



11/09/2020

Informatics 370 Deliverable 4 – Technical Specification

The technical specification deliverable aims to give further information about technical aspects of system through. This document includes detailed technical use case diagrams and their corresponding narratives for describing the steps included in each use case. A complete set of dataflow diagrams accompanied by pseudo code provide a more thorough explanation of the system. To support the diagrams detailed descriptions of all the inputs and out outputs are provided and validated against the requirements.



Name (from left to right)	Surname	Student Number
Byron J.	Donald	17237913
Hansen	Li	17073708
Tapiwarufaro	Ndoro (Group Leader)	17210918
Cameron	King	17165904
Gareth	Phillips	17004162

Virtual Visionaries
GROUP 17



Table of Contents

1.	Deliverable Introduction	1
2.	Use Cases	2
2.1.	Introduction	2
2.2.	Full set of Use Case Diagrams	2
2.2.1.	Introduction	2
2.2.2.	Sub-system 1: User	3
2.2.3.	Sub-system 2: Mechanic Schedule.....	4
2.2.4.	Sub-system 3: Tracking Hardware	5
2.2.5.	Sub-system 4: Student	6
2.2.6.	Sub-system 5: Bicycle.....	7
2.2.7.	Sub-system 6: Bicycle Part	8
2.2.8.	Sub-system 7: Tracking Log.....	9
2.2.9.	Sub-system 8: Job Tasks.....	10
2.2.10.	Sub-system 9: School	11
2.2.11.	Sub-system 10: Job	12
2.2.12.	Sub-system 11: Reporting.....	13
2.2.13.	Sub-system 12: Notification	14
2.2.14.	Sub-system 13: Administration.....	15
2.2.15.	Conclusion.....	16
2.3.	Full set of Use Case Narratives.....	16
2.3.1.	Introduction	16
2.3.2.	Sub-system 1: User	17
2.3.3.	Sub-system 2: Mechanic Schedule.....	100
2.3.4.	Sub-system 3: Tracking Hardware	108
2.3.5.	Sub-system 4: Student	164
2.3.6.	Sub-system 5: Bicycle.....	237
2.3.7.	Sub-system 6: Bicycle Part	291
2.3.8.	Sub-system 7: Tracking	347
2.3.9.	Sub-system 8: Job Tasks.....	382
2.3.10.	Sub-system 9: School	435
2.3.11.	Sub-system 10: Job	469
2.3.12.	Sub-system 11: Reporting.....	499
2.3.13.	Sub-system 12: Notification	541
2.3.14.	Sub-system 13: Administration.....	548



2.3.15.	Conclusion.....	569
3.	Process-Oriented Design.....	570
3.1.	Introduction	570
3.2.	Diagrams	570
3.3.	Conclusion.....	570
4.	System Database Design	571
4.1.	Introduction	571
4.2.	Entity Relationship Diagram.....	572
4.3.	Size Estimation of Proposed Database	573
4.3.1.	Introduction	573
4.3.2.	Size Estimation	573
4.3.3.	5 Year Projection.....	575
4.3.4.	Conclusion.....	575
4.4.	Conclusion.....	576
5.	System Interface Design	577
5.1.	Introduction	577
5.2.	Input and Output Design.....	577
5.3.	Conclusion.....	577
6.	Package Design Specifications	578
6.1.	Test Specifications.....	578
6.1.1.	Introduction	578
6.1.2.	Test Plan.....	578
6.1.3.	Test Scenarios and Test Data: Add Bicycle.....	579
6.1.4.	Test Scenarios and Test Data: Assign a Bicycle to Student	583
6.1.5.	Test Scenarios and Test Data: Report Repair Job	586
6.1.6.	Test Scenarios and Test Data: Check-in Bicycle	593
6.1.7.	Test Scenarios and Test Data: View Jobs	595
6.1.8.	Test Scenarios and Test Data: Complete Job	597
6.1.9.	Conclusion.....	598
6.2.	Hardware and Software Requirements	599
6.2.1.	Introduction	599
6.2.2.	Hardware Requirements.....	599
6.2.3.	Software Requirements	600
6.2.4.	Conclusion.....	600
6.3.	Network / Web Layout Specifications	601
6.3.1.	Introduction	601



6.3.2.	Diagram.....	601
6.3.3.	Conclusion.....	601
6.4.	Validation	602
6.4.1.	Introduction	602
6.4.2.	Validation for each Data Flow.....	602
6.4.3.	Conclusion.....	625
7.	Deliverable Conclusion & Sign-off.....	626
7.1.	Conclusion.....	626
7.2.	Client Sign-off.....	626
7.3.	Group Sign-off.....	626
8.	Complexity Matrix.....	627



Table of Figures

Figure 1 Sub-system 1 Use Case Diagram	3
Figure 2 Sub-system 2 Use Case Diagram	4
Figure 3 Sub-system 3 Use Case Diagram	5
Figure 4 Sub-system 4 Use Case Diagram	6
Figure 5 Sub-system 5 Use Case Diagram	7
Figure 6 Sub-system 6 Use Case Diagram	8
Figure 7 Sub-system 7 Use Case Diagram	9
Figure 8 Sub-system 8 Use Case Diagram	10
Figure 9 Sub-system 9 Use Case Diagram	11
Figure 10 Sub-system 10 Use Case Diagram	12
Figure 11 Sub-system 11 Use Case Diagram	13
Figure 12 Sub-system 12 Use Case Diagram	14
Figure 13 Sub-system 13 Use Case Diagram	15
Figure 14 Entity Relationship Diagram	572
Figure 15 Network Layout Specification	601



1. Deliverable Introduction

The technical specification deliverable aims to give further information about the technical aspects of the system. This document includes detailed technical use case diagrams and their corresponding narratives for describing the steps included in each use case. A complete set of dataflow diagrams accompanied by pseudo-code provide a more thorough explanation of the system. To support the diagrams detailed descriptions of all the inputs and out outputs are provided and validated against the requirements.



2. Use Cases

2.1. Introduction

Use cases refer to the different processes within a system. They are found within individual Sub-systems which in turn makes up the entire system. They can, therefore, be seen as the building blocks of a system when it comes to the process within a system.

2.2. Full set of Use Case Diagrams

2.2.1. Introduction

Use case diagrams are used to demonstrate the interactions between the users and the system. This will include all the processes and sub-systems of our system as well as how the users interact with them and how they will interact with each other.

Lastly, Use case diagrams also shows what outputs the system generates if there are any.



2.2.2. Sub-system 1: User

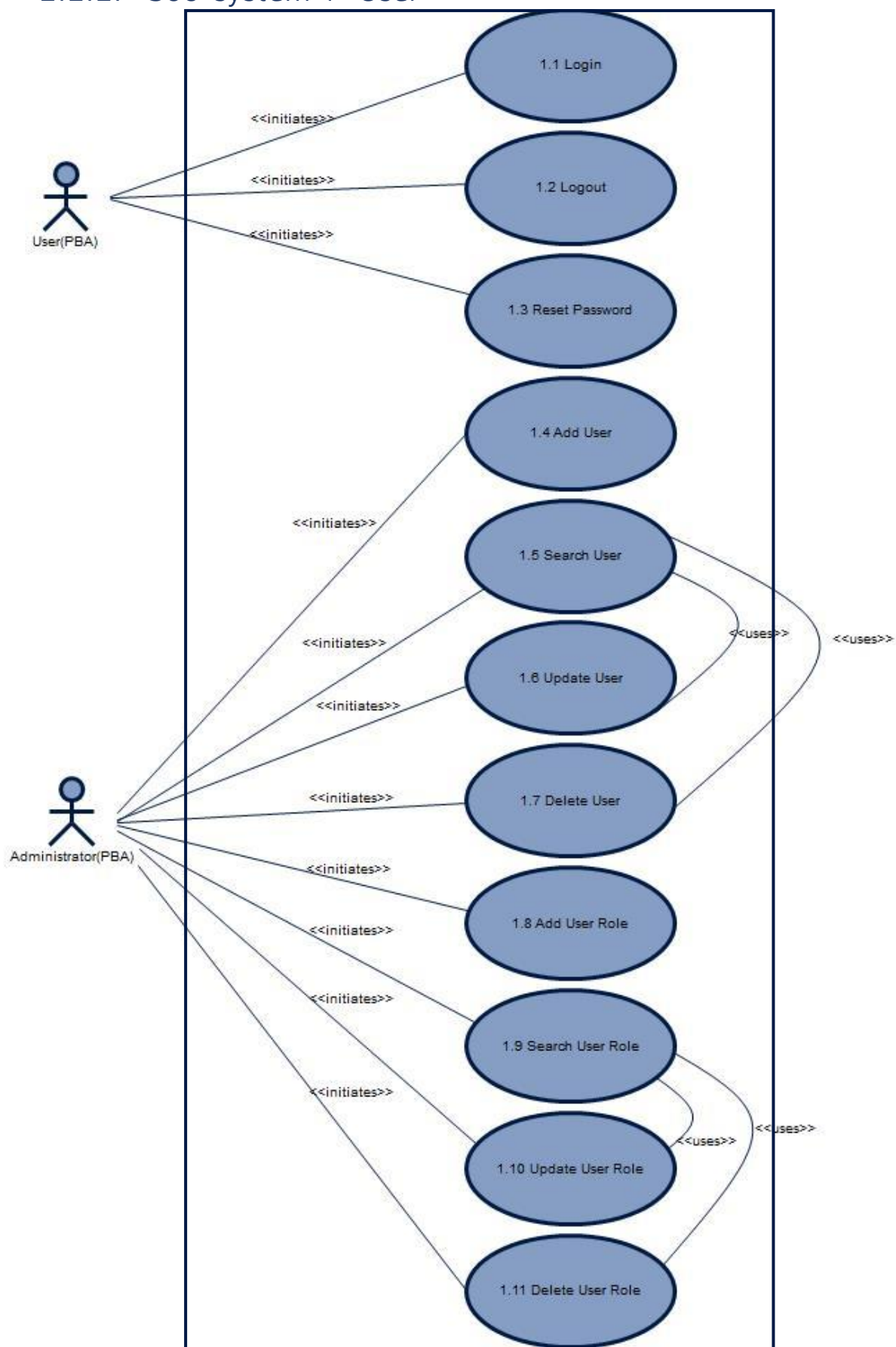


Figure 1 Sub-system 1 Use Case Diagram



2.2.3. Sub-system 2: Mechanic Schedule

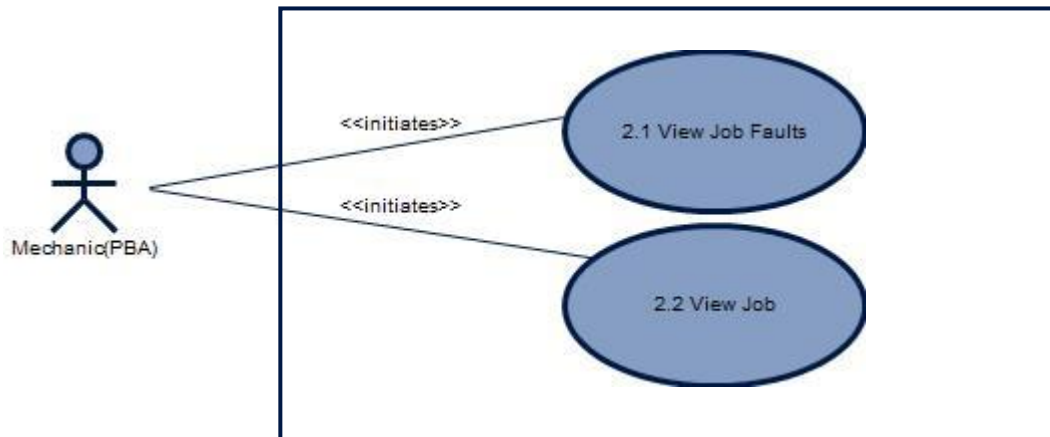


Figure 2 Sub-system 2 Use Case Diagram



2.2.4. Sub-system 3: Tracking Hardware

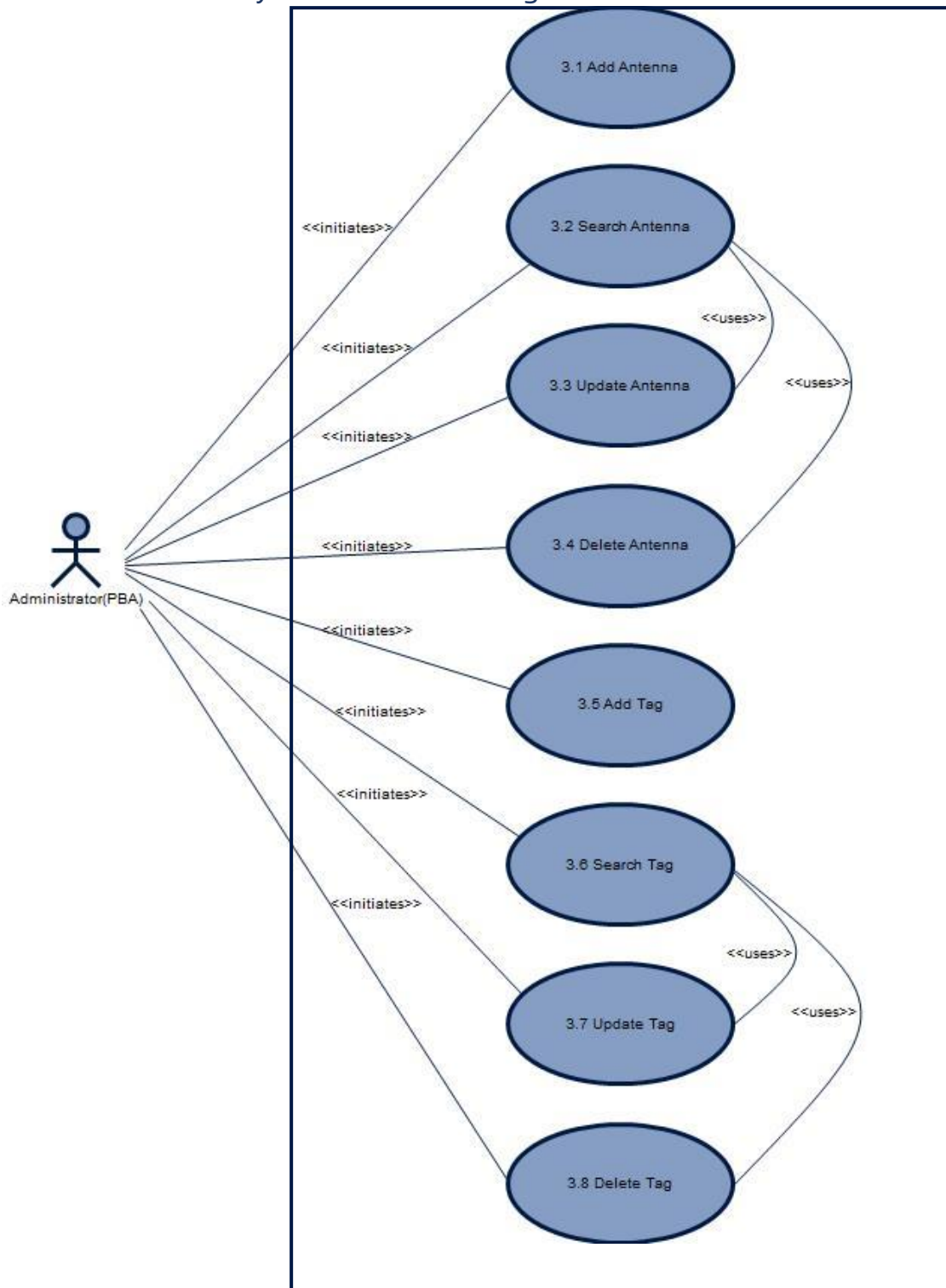


Figure 3 Sub-system 3 Use Case Diagram



2.2.5. Sub-system 4: Student

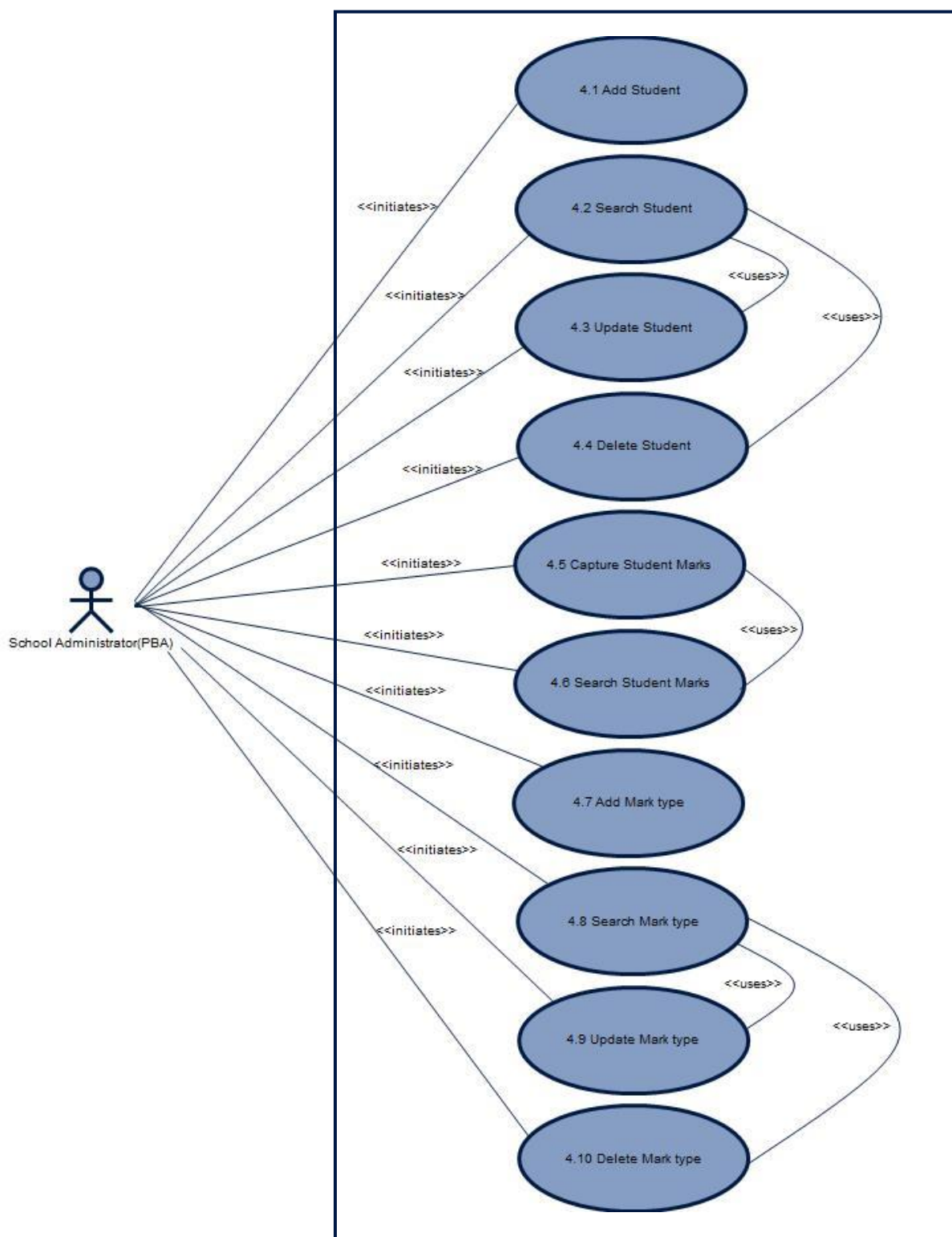


Figure 4 Sub-system 4 Use Case Diagram



2.2.6. Sub-system 5: Bicycle

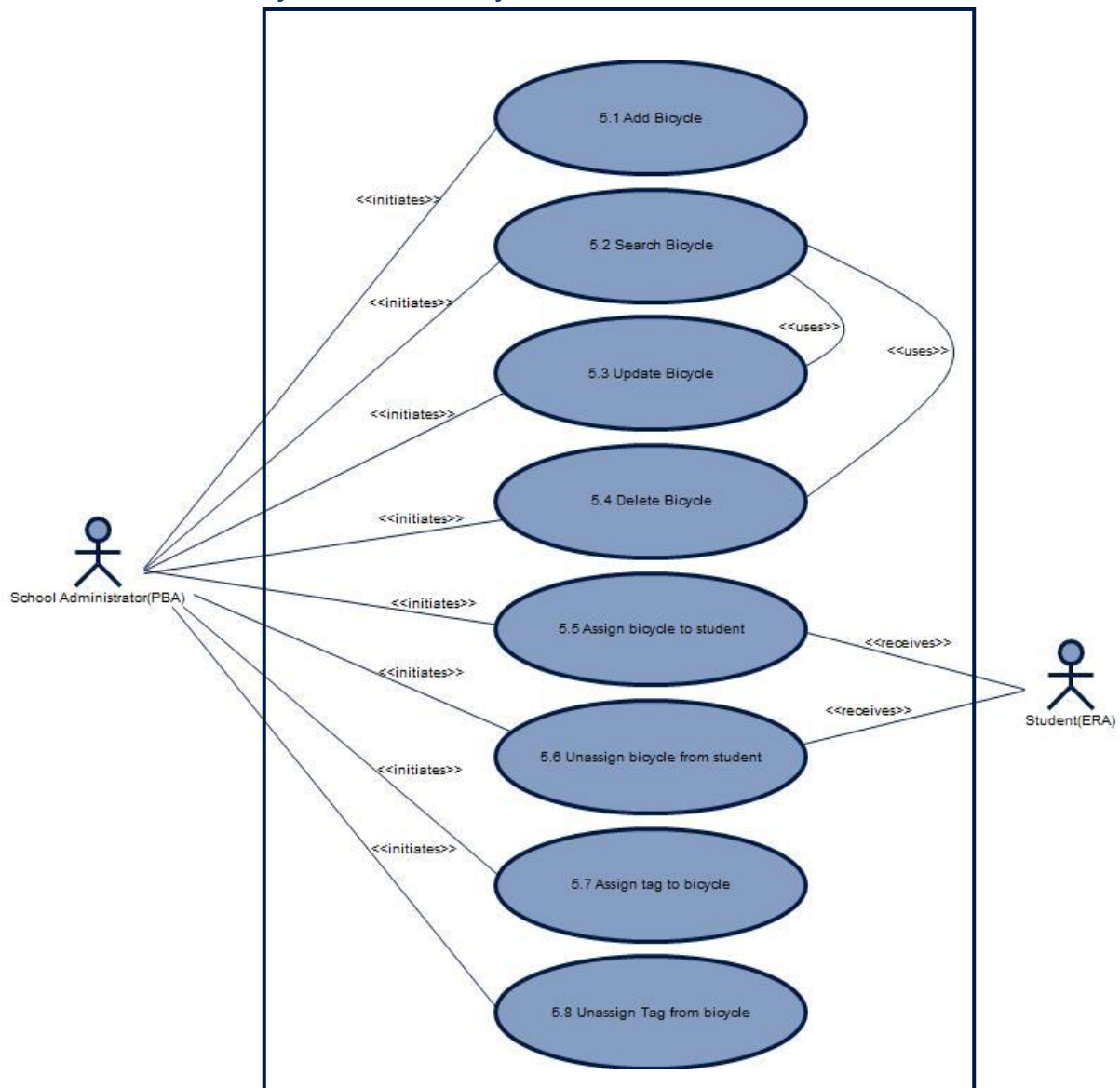


Figure 5 Sub-system 5 Use Case Diagram



2.2.7. Sub-system 6: Bicycle Part

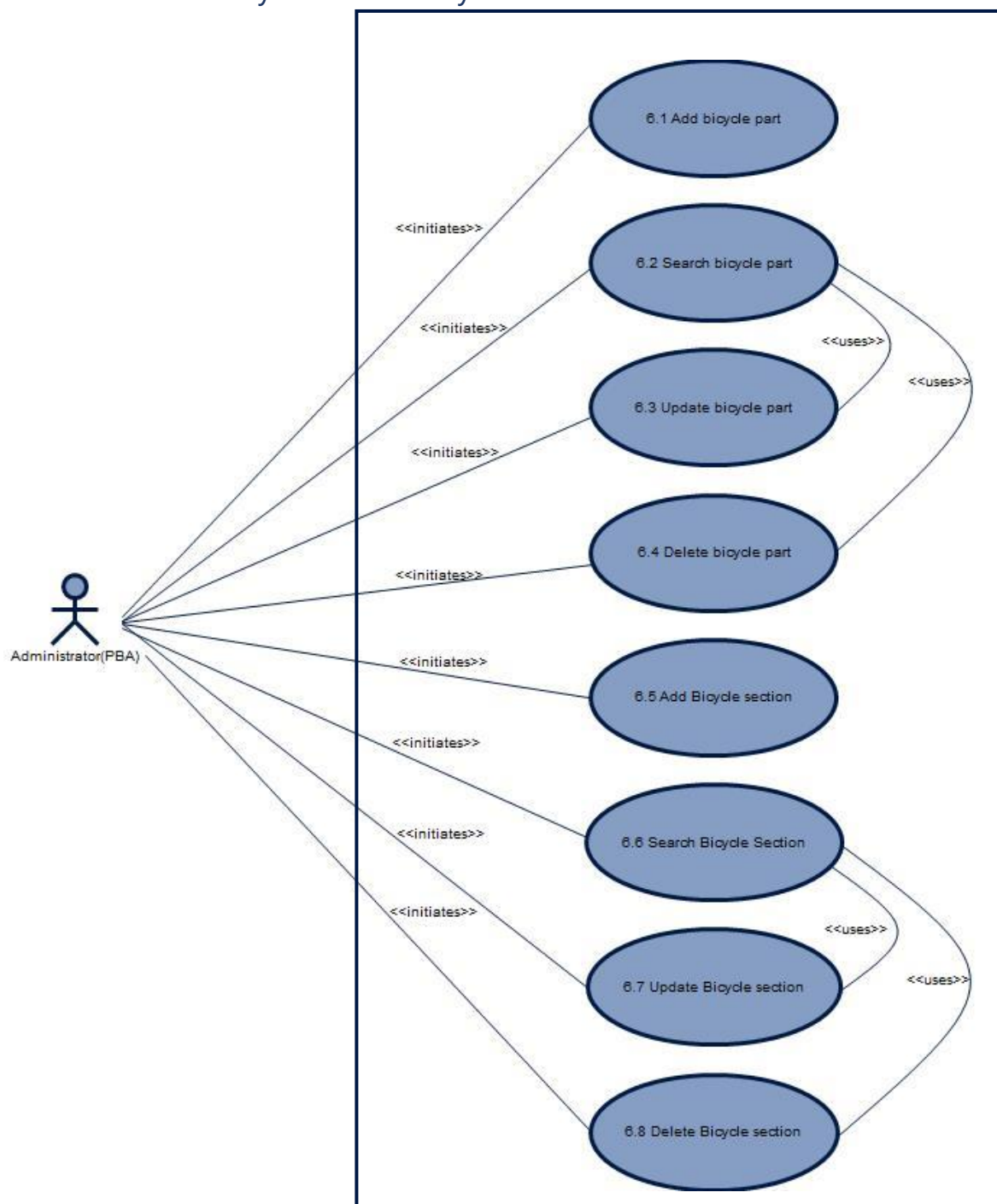


Figure 6 Sub-system 6 Use Case Diagram



2.2.8. Sub-system 7: Tracking Log

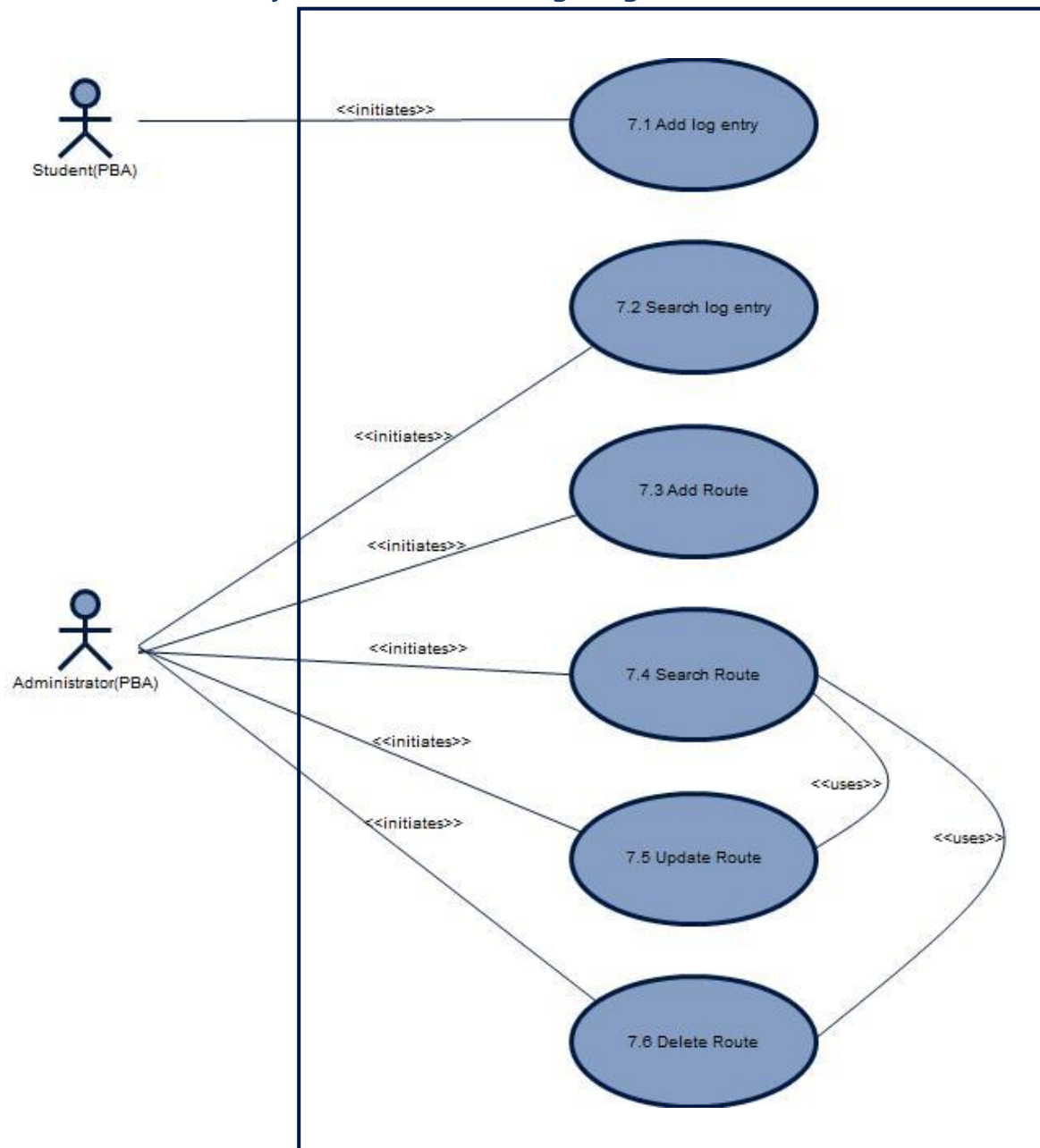


Figure 7 Sub-system 7 Use Case Diagram



2.2.9. Sub-system 8: Job Tasks

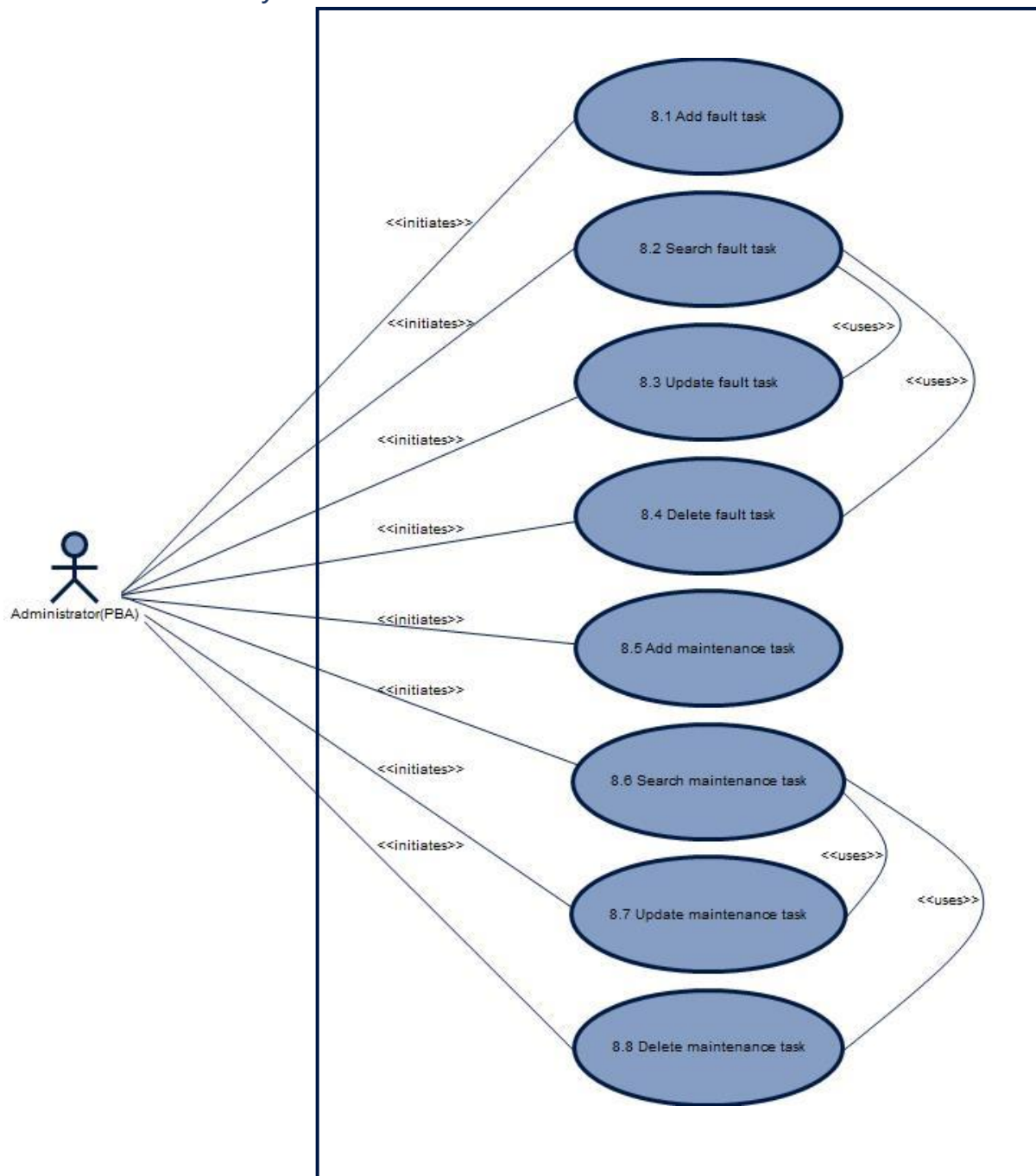


Figure 8 Sub-system 8 Use Case Diagram



2.2.10. Sub-system 9: School

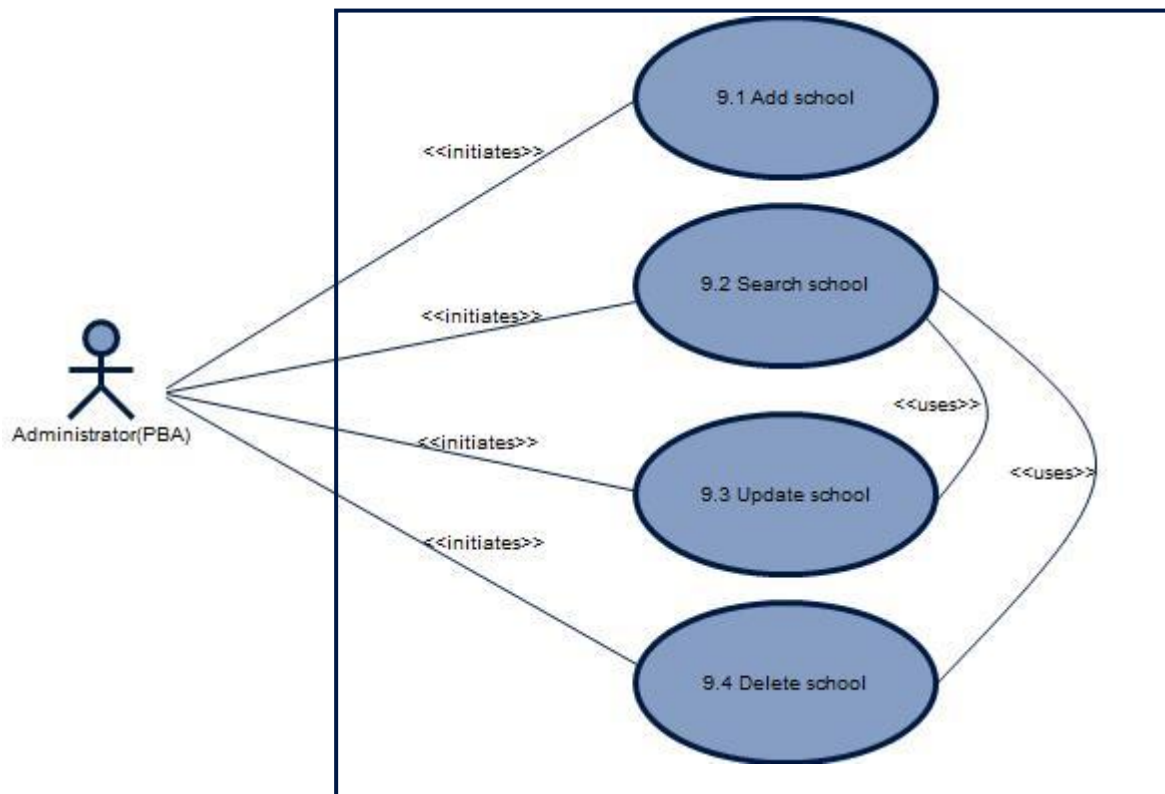


Figure 9 Sub-system 9 Use Case Diagram



2.2.11. Sub-system 10: Job

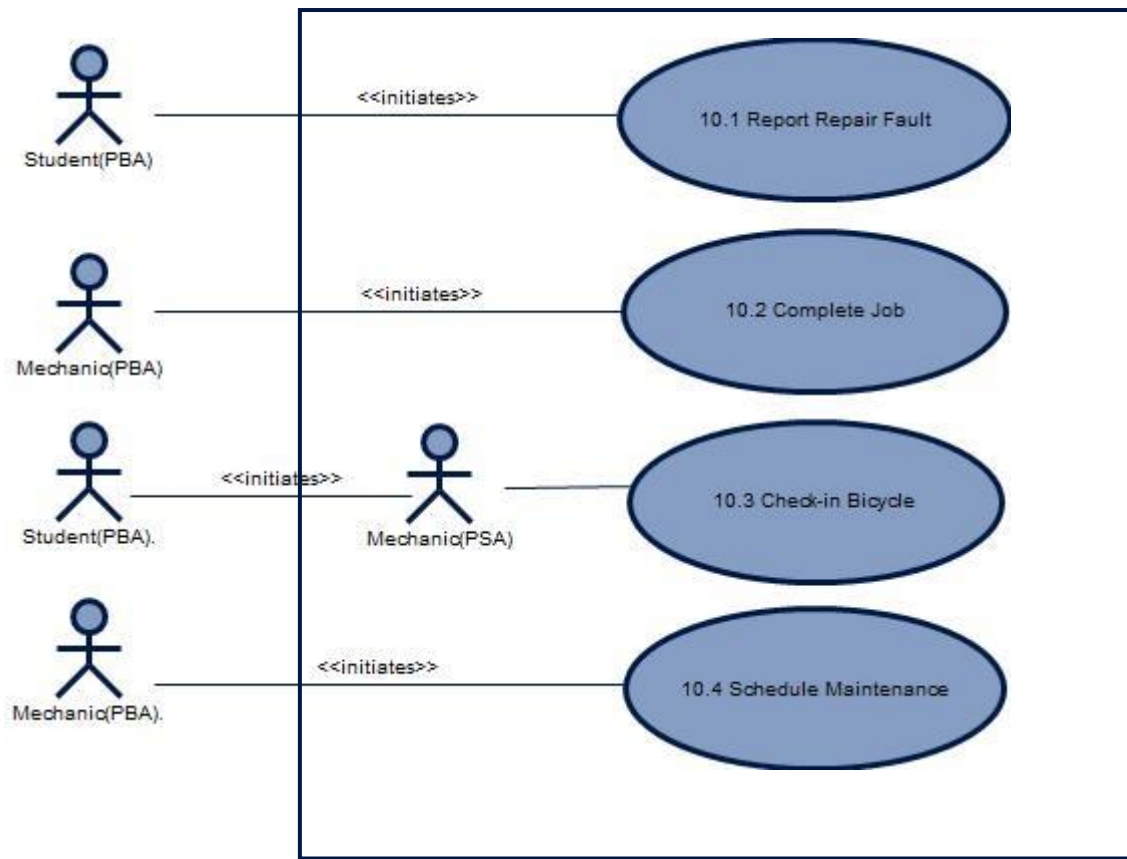


Figure 10 Sub-system 10 Use Case Diagram



2.2.12. Sub-system 11: Reporting

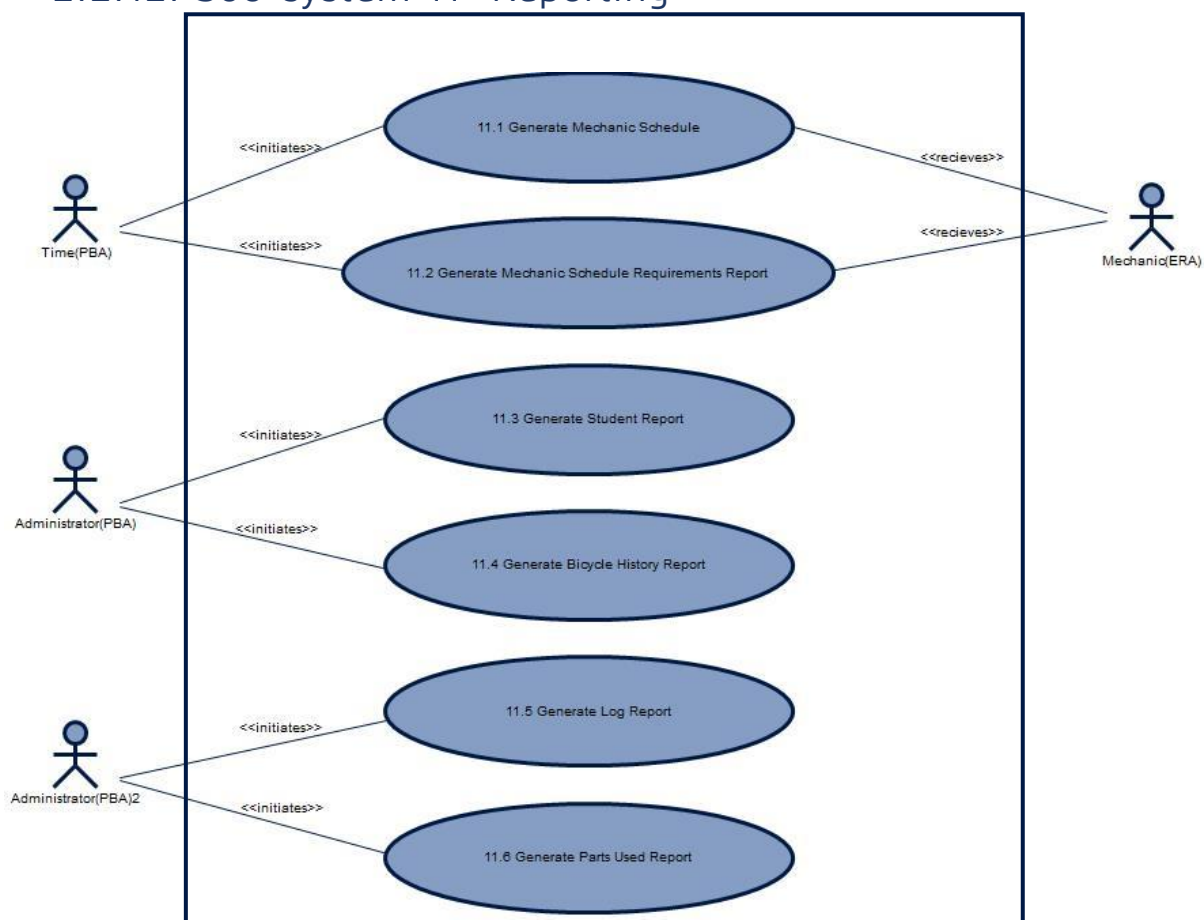


Figure 11 Sub-system 11 Use Case Diagram



2.2.13. Sub-system 12: Notification

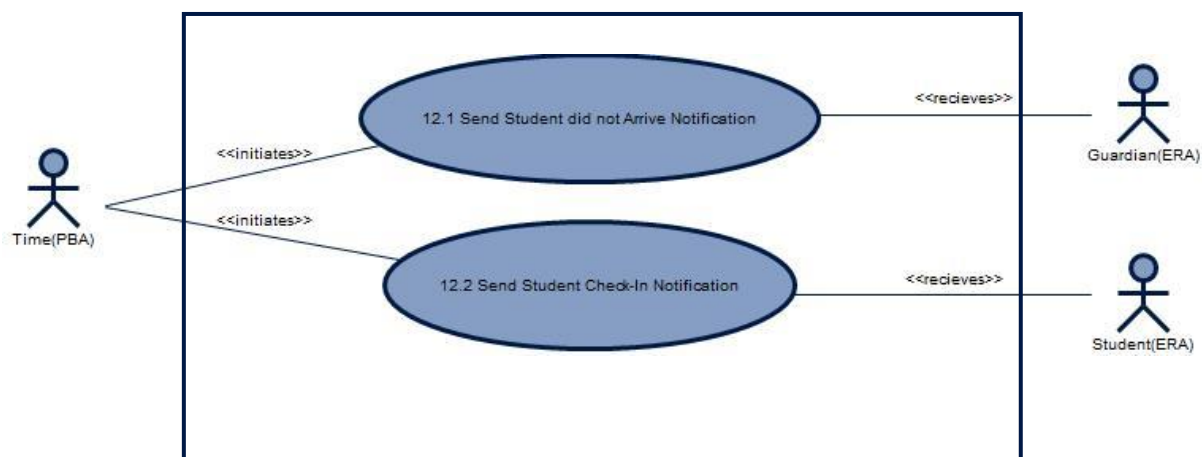


Figure 12 Sub-system 12 Use Case Diagram



2.2.14. Sub-system 13: Administration

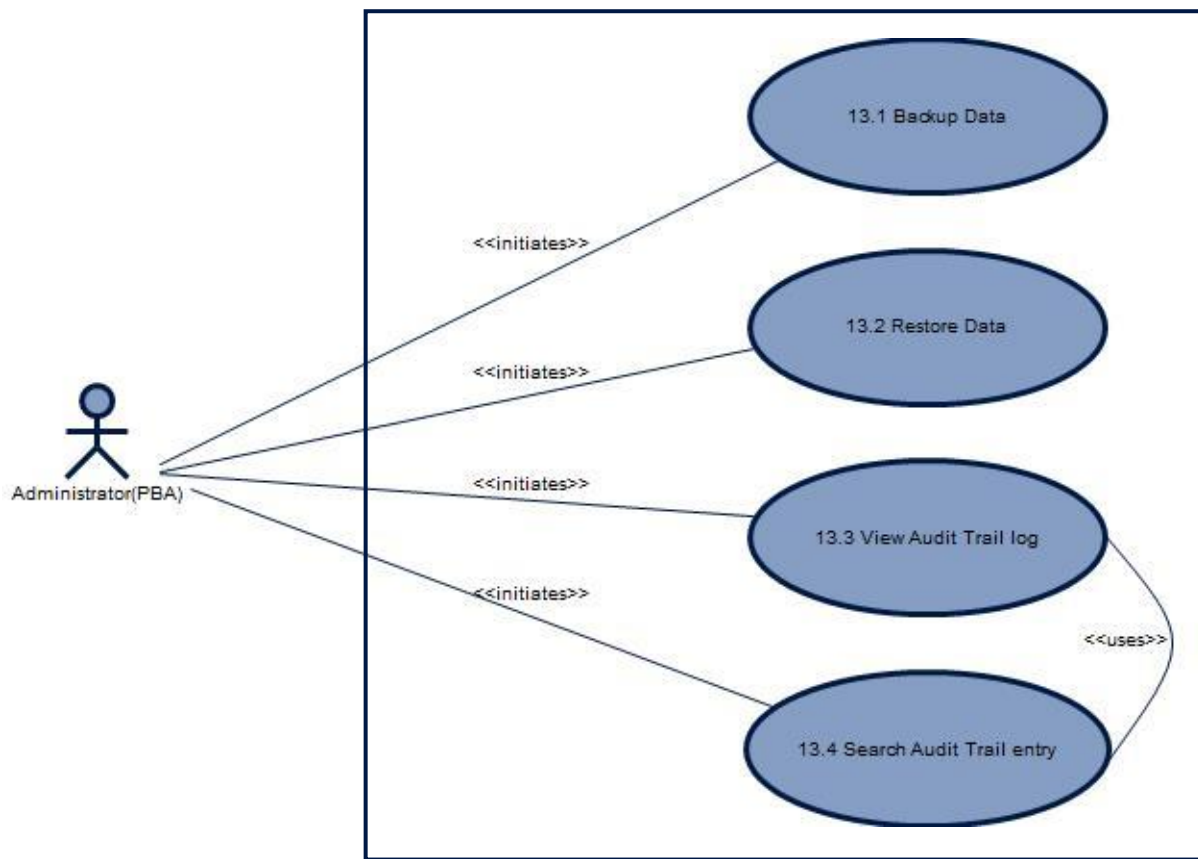


Figure 13 Sub-system 13 Use Case Diagram

2.2.15. Conclusion

In conclusion, we were able to gain a lot more insight into each of our processes by listing out each step in such a way that it is easily understandable. We also made various changes to our use case diagram after identifying some needed changes from the narratives.

2.3. Full set of Use Case Narratives

2.3.1. Introduction

Use case Narratives expand on Use case diagrams by going into a lot more detail about each Use case. This includes the steps involved in each use case as well as how the user interacts with the process. Lastly, the Use case narratives also assists in mentioning critical information such as business rules, what's changed in a system as well as triggers.

2.3.2. Sub-system 1: User

2.3.2.1. Login

System Name:	VVs
Author:	Hansen Li
Date:	18/06/2020
Version:	1.1

Use Case Name:	Login	Use Case Type:
Use Case ID:	1.1	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	User	
Primary System Actor	None	
Other Participating Actors:	None	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process of a user wanting to log into the system. The Use Case starts when the user navigates from the "Home Page" by selecting the "Login" option under the "Navigation" taskbar menu and clicks the "Login" option. The system then will load the "Login" form which will in turn allow the user to enter their login details. Once their details have been verified the use case will conclude when the User is granted access to system functionality based on their authorization level.	

Pre-Condition:	The User must be registered on the system				
Trigger:	The User requests to log in to the system				
Typical Course Of Events:	Actor Action	System Response			
		Manual Action	Automated Action		
	Step 1: The User from the “Homepage” screen selects the “Sidebar” button and then clicks on the “Login” option.		Step 2: The system Displays the “Login” screen and populates it with the following details:		
			Text/Label	Control Type	Additional Info
			Login	Label	Heading for the form displayed.
			Username	Label	Used to indicate what the corresponding Text-Box is used for.
			Password	Label	Used to indicate what the corresponding Text-Box is used for.
			Username	Text-Box	Used to capture text for the inputted Username.
Password	Text-Box	Used to capture text for the			

			inputted Password.						
			<table><tr><td>Login</td><td>Button</td><td>Verifies User Details and if correct, takes User to the "landing" page</td></tr><tr><td>Forgot Password</td><td>Link</td><td>Takes User to the Forgot Password Form</td></tr></table>	Login	Button	Verifies User Details and if correct, takes User to the "landing" page	Forgot Password	Link	Takes User to the Forgot Password Form
	Login	Button	Verifies User Details and if correct, takes User to the "landing" page						
	Forgot Password	Link	Takes User to the Forgot Password Form						
	Step 3: User inputs the following Login details: - Username - Password								
Step 4: The User clicks on the "Login" button. [alt]		Step 5: The system captures the entered details and ensures the Username and Password fields have been filled. [alt] <i>User table</i> <table><tr><th>Attribute</th><th>Validation</th></tr><tr><td>Username</td><td>1. Max 50 characters 2. Text characters only</td></tr><tr><td>Password</td><td> - Max 50 characters - Text characters only </td></tr></table>	Attribute	Validation	Username	1. Max 50 characters 2. Text characters only	Password	- Max 50 characters - Text characters only	
Attribute	Validation								
Username	1. Max 50 characters 2. Text characters only								
Password	- Max 50 characters - Text characters only								
		Step 6: The system uses a FIRESQL READ query to retrieve the following: [alt] <table><tr><th>Entity</th><th>Attributes</th></tr><tr><td>User</td><td>Username</td></tr></table>	Entity	Attributes	User	Username			
Entity	Attributes								
User	Username								

			User Password User UserTypeID UserType UserType The system will then compare the captured details to the details retrieved from the database and verifies they match.
			Step 7: The system will Return the Admin to the home screen with an authorization level of "Admin". [alt]
			Step 8: The system creates and saves the following in the database by using a FIRESQL INSERT query: AuditTrail Table AuditID DateTime TransactionDescription UserID
Alternate Courses:	Alt Step 4: The User clicks the Forgot Password link; the system will Invoke Use Case 1.3 Forgot Password.		
	Alt Step 5: System detects an Input error; the required fields have not been filled. The system will notify the User of the error with the following error Banner:		
	Text/Label "Please complete this field."	Control Type Label (with a light blue background color fill)	Additional Info Used to inform the User of an empty field "X" Icon Used to indicate an error result.

	Return to Step 3.									
	Alt Step 6: In the case of incorrect login details, the system notifies the User of the Verification error with the following error Banner:									
	<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>"The Login Details were incorrect, Please Try Again."</td><td>Banner popup</td><td>Used to inform the User of the result.</td></tr><tr><td>"X"</td><td>Icon</td><td>Used to indicate an error result.</td></tr></table>	Text/Label	Control Type	Additional Info	"The Login Details were incorrect, Please Try Again."	Banner popup	Used to inform the User of the result.	"X"	Icon	Used to indicate an error result.
	Text/Label	Control Type	Additional Info							
	"The Login Details were incorrect, Please Try Again."	Banner popup	Used to inform the User of the result.							
"X"	Icon	Used to indicate an error result.								
Redirect to Step 5.										
Alt Step 7a: The system will load the Mechanic Calendar Screen with an authorization level of "Mechanic".										
	Alt Step 7b: The system will Return the School Administrator to the home screen with an authorization level of "Admin".									
Conclusion:	The use case narrative concludes when the User has been successfully logged into the system.									
Post-Condition:	- The User will be granted access to system functionality based on their access level stored in the UserType table. - The AuditTrail table has been successfully updated.									
Business Rules	- Only registered users will be granted access to the system functionality. - Users must be granted different access permissions based on their User Type.									
Implementation Constraints and Specifications:	Passwords will be hashed and compared using the SHA256 algorithm.									

Assumptions:	• None
Open Issues:	• None

2.3.2.2. Logout

System Name:	VVs
Author:	Hansen Li
Date:	18/06/2020
Version:	1.1

Use Case Name:	Logout	Use Case Type:
Use Case ID:	1.2	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	User	
Primary System Actor	None	
Other Participating Actors:	None	
Other Interested Stakeholders:	None	
Description:	This use Technical use case narrative describes the event whereby a User requests to log out of the system. The User will click on the "Logout" option on the "Navigation" Taskbar. The use case concludes when the system notifies the User that they have been successfully logged out after they confirm it.	
Pre-Condition:	The User must be logged into the system.	
Trigger:	The User requests to log out of the system.	
Typical Course	Actor Action	System Response

Of Events:		Manual Action	Automated Action												
	Step 1: The User clicks the "Logout" option in the "Navigation" Taskbar		Step 2: The system Displays the popup screen and populates it with the following details:												
			<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Yes</td><td>Button</td><td>Logs the User out and returns them to the "Login" Page</td></tr><tr><td>No</td><td>Button</td><td>Returns User to "Homepage"</td></tr><tr><td>Are you sure you want to logout</td><td>Label</td><td>Displays a list of Bicycle Parts.</td></tr></table>	Text/Label	Control Type	Additional Info	Yes	Button	Logs the User out and returns them to the "Login" Page	No	Button	Returns User to "Homepage"	Are you sure you want to logout	Label	Displays a list of Bicycle Parts.
			Text/Label	Control Type	Additional Info										
			Yes	Button	Logs the User out and returns them to the "Login" Page										
	No	Button	Returns User to "Homepage"												
	Are you sure you want to logout	Label	Displays a list of Bicycle Parts.												
	Step 3: The User confirms the logout request and presses the "yes" button. [alt]		Step 4: The system Displays the "Login" screen and populates it with the following details:												
			<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Login</td><td>Label</td><td>Heading for the form displayed.</td></tr><tr><td>Username</td><td>Label</td><td>Used to indicate what the corresponding Text-Box is used for.</td></tr><tr><td>Password</td><td>Label</td><td>Used to indicate what the corresponding Text-Box is used for.</td></tr></table>	Text/Label	Control Type	Additional Info	Login	Label	Heading for the form displayed.	Username	Label	Used to indicate what the corresponding Text-Box is used for.	Password	Label	Used to indicate what the corresponding Text-Box is used for.
			Text/Label	Control Type	Additional Info										
			Login	Label	Heading for the form displayed.										
Username			Label	Used to indicate what the corresponding Text-Box is used for.											
Password			Label	Used to indicate what the corresponding Text-Box is used for.											

			Username	Text-Box	Used to capture text for the inputted Username.
			Password	Text-Box	Used to capture text for the inputted Password.
			Login	Button	Verifies User Details and if correct, takes User to the "landing" page
			Forgot Password	Link	Takes User to the Forgot Password Form
				Step 5: The system creates and saves the following in the database by using a FIRESQL INSERT query: AuditTrail Table AuditID DateTime TransactionDescription UserID	
Alternate Courses:	Alt Step 3: The User clicks "no", the Use Case is then terminated, and the User is returned to the homepage.				
Conclusion:	The use case narrative concludes when the system notifies the User that they have been successfully logged out of the system.				

Post-Condition:	- The User can no longer navigate the system or access functionality. - The AuditTrail table has been successfully updated.
Business Rules	The system will automatically logout a User after 10 minutes of inactivity.
Implementation Constraints and Specifications:	None
Assumptions:	None
Open Issues:	None

2.3.2.3. *Forgot Password*

System Name:	VVs
Author:	Hansen Li
Date:	18/06/2020
Version:	1.1

Use Case Name:	Forgot Password	Use Case Type:
Use Case ID:	1.3	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	User	
Primary System Actor	None	
Other Participating Actors:	None	
Other Interested Stakeholders:	None	
Description:	This use case narrative describes the event whereby a User requests to reset their password. The User will press the "Forgot Password" link and the system will prompt the user to enter their email address and notify them that they will be sent a reset password link. The email will instruct the User the follow the link and they will be prompted to input and confirm their new password. The Use Case will conclude when the system notifies the user that the password has been reset successfully.	
Pre-Condition:	The User must be registered on the system.	

Trigger:	The User requests to reset their password.																							
Typical Course Of Events:	Actor Action	System Response																						
		Manual Action	Automated Action																					
	Step 1: The User requests to reset their password and presses on the "Reset Password" link.		Step 2: The system Displays the "Forgot Password" screen and populates it with the following details:																					
			<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Forgot Password</td><td>Label</td><td>Heading for the form displayed.</td></tr><tr><td>Email Address</td><td>Label</td><td>Used to indicate what the corresponding Text-Box is used for.</td></tr><tr><td>"Please enter your email</td><td>Label</td><td>To give instructions</td></tr><tr><td>Email Address</td><td>Textbox</td><td>Used to capture the User's Email Address</td></tr><tr><td>Submit</td><td>Button</td><td>Clicking on this will send an Email to the Email the User inputted</td></tr><tr><td>Cancel</td><td>Button</td><td>The button will cancel the forgot password process.</td></tr></table>	Text/Label	Control Type	Additional Info	Forgot Password	Label	Heading for the form displayed.	Email Address	Label	Used to indicate what the corresponding Text-Box is used for.	"Please enter your email	Label	To give instructions	Email Address	Textbox	Used to capture the User's Email Address	Submit	Button	Clicking on this will send an Email to the Email the User inputted	Cancel	Button	The button will cancel the forgot password process.
			Text/Label	Control Type	Additional Info																			
			Forgot Password	Label	Heading for the form displayed.																			
			Email Address	Label	Used to indicate what the corresponding Text-Box is used for.																			
			"Please enter your email	Label	To give instructions																			
			Email Address	Textbox	Used to capture the User's Email Address																			
			Submit	Button	Clicking on this will send an Email to the Email the User inputted																			
			Cancel	Button	The button will cancel the forgot password process.																			

	Step 3: The User inputs their email address.							
	Step 4: The User presses the "Submit" button.		Step 5: The system captures the inputted email address and validates that the email address field is filled in and in the correct format. [alt] <i>User table</i> <table><tr><th>Attribute</th><th>Validation</th></tr><tr><td>Email Address</td><td>3. Max 50 characters 4. Text characters only</td></tr></table>	Attribute	Validation	Email Address	3. Max 50 characters 4. Text characters only	
	Attribute	Validation						
	Email Address	3. Max 50 characters 4. Text characters only						
			Step 6: The system uses a FIRESQL READ to retrieve the following: [alt] <table><tr><th>Entity</th><th>Attributes</th></tr><tr><td>User</td><td>UserID</td></tr><tr><td>User</td><td>Email</td></tr></table> The system verifies that the specified email is linked to a User	Entity	Attributes	User	UserID	User
Entity	Attributes							
User	UserID							
User	Email							
		Step 7: The system sends a Reset Password link to the Users email address.						
		Step 8: The system notifies the User that the Reset Password link has been sent with the following popup: <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Success</td><td>Heading</td><td>Heading for the popup</td></tr></table>	Text/Label	Control Type	Additional Info	Success	Heading	Heading for the popup
Text/Label	Control Type	Additional Info						
Success	Heading	Heading for the popup						

			Forgot Password Label Email sent	Summary for the popup displayed.
			OK Button	Used to accept the result
	Step 9: The User navigates to their Email and clicks on the "Reset Password" link.	Step 10: The system displays the Reset Password screen with the following details:		
			Text/Label	Control Type
			Reset Password	Label
			Heading for the form displayed.	
			Password	Label
			Used to indicate what the corresponding Text-Box is used for.	
			Confirm Password	Label
			Used to indicate what the corresponding Text-Box is used for.	
			Password	Textbox
			Textbox to capture the New Password	
			Confirm Password	Textbox
			Textbox to confirm the capture of the New Password	
			Reset Password	Button
			Clicking on this will reset the	

			password of the User					
	Step 11: The User inputs the new password details: - Password - Confirm Password							
	Step 12: The User clicks the "Reset Password" button		Step 13: The system captures the inputted password details and verifies that the required fields have been filled. The system will also verify that the following inputted fields match identically as well as meet the password specification criteria: [alt] <table><tr><th>Attribute</th><th>Validation</th></tr><tr><td>Password</td><td>5. Max 50 characters 6. Characters only</td></tr></table>	Attribute	Validation	Password	5. Max 50 characters 6. Characters only	
	Attribute	Validation						
	Password	5. Max 50 characters 6. Characters only						
		Step 14: The system uses a FIRESQL UPDATE query to update the following: <table><tr><th>Entity</th><th>Attributes</th></tr><tr><td>User</td><td>UserID</td></tr><tr><td>User</td><td>Password</td></tr></table>	Entity	Attributes	User	UserID	User	Password
Entity	Attributes							
User	UserID							
User	Password							
		Step 15: The system notifies the user that their password has been successfully updated with the following popup: [alt] <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Success</td><td>Label</td><td>Heading</td></tr></table>	Text/Label	Control Type	Additional Info	Success	Label	Heading
Text/Label	Control Type	Additional Info						
Success	Label	Heading						

			<table><tr><td>Password Successfully updated</td><td>Label</td><td>Description for the popup displayed.</td></tr><tr><td>OK</td><td>Button</td><td>Used to accept the result</td></tr></table>	Password Successfully updated	Label	Description for the popup displayed.	OK	Button	Used to accept the result			
	Password Successfully updated	Label	Description for the popup displayed.									
OK	Button	Used to accept the result										
			Step 16: The system creates and saves the following in the database by using a FIRESQL INSERT query: AuditTrail Table AuditID DateTime TransactionDescription UserID									
Alternate Courses:	Alt Step 5a: System detects an Input error; the required fields have not been filled. The system will notify the User of the error with the following error message: <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>"Please enter your email"</td><td>Label (with a light red background color fill)</td><td>Used to inform the User of an empty field</td></tr><tr><td>"X"</td><td>Button</td><td>Returns the user to the forgot password page</td></tr></table> Return to Step 3.			Text/Label	Control Type	Additional Info	"Please enter your email"	Label (with a light red background color fill)	Used to inform the User of an empty field	"X"	Button	Returns the user to the forgot password page
Text/Label	Control Type	Additional Info										
"Please enter your email"	Label (with a light red background color fill)	Used to inform the User of an empty field										
"X"	Button	Returns the user to the forgot password page										

	Alt Step 5b: System detects a verification error; the email was not found. The system will notify the User of the error with the following error message:									
	<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>"Error! Email is invalid, please try again."</td><td>Label (with a light red background color fill)</td><td>Used to inform the User of an empty field</td></tr><tr><td>"X"</td><td>Button</td><td>Returns the user to the forgot password page</td></tr></table>	Text/Label	Control Type	Additional Info	"Error! Email is invalid, please try again."	Label (with a light red background color fill)	Used to inform the User of an empty field	"X"	Button	Returns the user to the forgot password page
	Text/Label	Control Type	Additional Info							
	"Error! Email is invalid, please try again."	Label (with a light red background color fill)	Used to inform the User of an empty field							
	"X"	Button	Returns the user to the forgot password page							
Return to Step 3.										
Alt Step 13: The entered passwords are not identical, or they do meet the password specification criteria. The system will notify the User of the error. Return to Step 11.										
Alt Step 15: The password was not updated successfully; the system will notify the User of the error. Return to Step 11.										
Conclusion:	The use case narrative concludes when the system notifies the user that their password has been successfully updated.									
Post-Condition:	The Users password has been updated in the User table. The AuditTrail table has been successfully updated									
Business Rules	Only a User can reset their password.									
Implementation Constraints and Specifications:	- Passwords meet the following specifications: - Contain more than 7 characters - Contain an uppercase letter - Contain a number All passwords stored in the system are hashed.									

Assumptions:	None
Open Issues:	None

2.3.2.4. Add User

System Name:	VVs
Author:	Hansen Li
Date:	18/06/2020
Version:	1.1

Use Case Name:	Add User	Use Case Type:
Use Case ID:	1.4	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	Administrator	
Primary System Actor	None	
Other Participating Actors:	None	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process of an Administrator adding a new User to the system. The Use Case starts when the Administrator navigates by selecting the "User Management" option under the "User Management" Combo box in the Navigation bar. From there, they click the "Add User" button and the system will load the "User" screen. The system populates the relevant Combo boxes with information necessary for selection, and the corresponding labels, text boxes and buttons, icons, logos, and banners.	
Pre-Condition:	The Administrator must be logged into the system.	

Trigger:	The Administrator requests to add a new User to the system.			
Typical Course Of Events:	Actor Action	System Response		
		Manual Action	Automated Action	
	Step 1: The Administrator selects the “User Management” option under the “User Management” Combo box in the Navigation bar.		Step 2: The system Displays the “User Management” screen and populates it with the following details:	

	Step 3: The Administrator clicks on the "Add User" button.		Step 4: The system will load the "User" screen and displays it with the following controls and layout:		
			Text/Label	Control Type	Additional Info
			Menu	Menu-icon	Used to Navigate the side menu-bar.
			User	Label	Heading for the pop-up displayed.
			User Picture	Picture image	Displays the uploaded picture.
			Name	Label	Used to indicate what the corresponding text box is used for.
			Surname	Label	Used to indicate what the corresponding text box is used for.
			Email	Label	Used to indicate what the corresponding text box is used for.
			Username	Label	Used to indicate what the corresponding

			text box is used for.	
			User Type	Label
			Used to indicate what the corresponding Combo box is used for.	
			Phone Number	Label
			Used to indicate what the corresponding text box is used for.	
			Password	Label
			Used to indicate what the corresponding text box is used for.	
			Confirm Password	Label
			Used to indicate what the corresponding text box is used for.	
			Name	Text box
			Used to capture text for the Name details.	
			Surname	Text box
			Used to capture text for the Surname details.	

			Email	Text box	Used to capture text for the Email details.
			Username	Text box	Used to capture text for the Username details.
			User Type	Combo box	Used to Select the User Type and is Populated by contents from the ERD with options as indicated in the table below.
			Phone Number	Text Box	Used to capture text for the Phone Number details.
			Password	Password Text Box	Used to capture the User's Password Details
			Confirm Password	Password Text Box	Used to confirm that the User's password matches this Password
			Create	Button	Used to Save the new User Details.

			Cancel Button Used to cancel the action of adding a User The User Combo box options are populated by using a FIRESQL READ query with data from the following: <table><thead><tr><th>Entity</th><th>Attributes</th></tr></thead><tbody><tr><td>User Type</td><td>UserType</td></tr></tbody></table>	Entity	Attributes	User Type	UserType							
	Entity	Attributes												
	User Type	UserType												
Step 5: Administrator inputs the following User details: - Name - Surname - Email - Username - User Type - Phone Number														
Step 6: The Administrator clicks the “Continue” button. [alt]		Step 7: The system displays the following form next: [alt] <table><thead><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr></thead><tbody><tr><td>School Administrator</td><td>Label</td><td>Heading to show which option the user picked</td></tr><tr><td>Position</td><td>Label</td><td>Used to indicate what the corresponding Combo box is used for.</td></tr><tr><td>School</td><td>Label</td><td>Used to indicate what the</td></tr></tbody></table>	Text/Label	Control Type	Additional Info	School Administrator	Label	Heading to show which option the user picked	Position	Label	Used to indicate what the corresponding Combo box is used for.	School	Label	Used to indicate what the
Text/Label	Control Type	Additional Info												
School Administrator	Label	Heading to show which option the user picked												
Position	Label	Used to indicate what the corresponding Combo box is used for.												
School	Label	Used to indicate what the												

				corresponding Combo box is used for.						
			Position	Combo Box	Used to Select the Administrator Position and is Populated by contents from the ERD with options as indicated in the table below.					
			School	Combo Box	Used to Select the School and is Populated by contents from the ERD with options as indicated in the table below.					
			"Return"	Button	Used to go back to the create user form					
			"Create"	Button	Used to submit the form					
			The School and Position Combo box options are populated by using a FIRESQL READ query with data from the following:							
			<table><tr><th>Entity</th><th>Attributes</th></tr><tr><td>School</td><td>SchoolName</td></tr><tr><td>AdminSchool</td><td>Position</td></tr></table>		Entity	Attributes	School	SchoolName	AdminSchool	Position
Entity	Attributes									
School	SchoolName									
AdminSchool	Position									

			Step 8: The system captures and ensures that all of the fields are filled in and an option is selected in the Combo boxes. [alt]																
			Step 9: The system validates that the captured details are in the correct format: [alt]																
			<i>User table</i>																
			<table><tr><th>Attribute</th><th>Validation</th></tr><tr><td>Name</td><td>7. Max 50 characters 8. Text characters only</td></tr><tr><td>Surname</td><td> - Max 50 characters - Text characters only </td></tr><tr><td>Email</td><td> - Max 50 characters - Text characters only </td></tr><tr><td>Username</td><td> - Max 50 characters - Text characters only </td></tr><tr><td>User Type</td><td> - Option must be selected </td></tr><tr><td>Phone Number</td><td> - Max 50 characters - Text characters only </td></tr><tr><td>Password</td><td> - Max 50 characters </td></tr></table>	Attribute	Validation	Name	7. Max 50 characters 8. Text characters only	Surname	- Max 50 characters - Text characters only	Email	- Max 50 characters - Text characters only	Username	- Max 50 characters - Text characters only	User Type	Option must be selected	Phone Number	- Max 50 characters - Text characters only	Password	Max 50 characters
	Attribute	Validation																	
	Name	7. Max 50 characters 8. Text characters only																	
	Surname	- Max 50 characters - Text characters only																	
	Email	- Max 50 characters - Text characters only																	
	Username	- Max 50 characters - Text characters only																	
	User Type	Option must be selected																	
Phone Number	- Max 50 characters - Text characters only																		
Password	Max 50 characters																		
		<i>Admin Table</i>																	
		<table><tr><th>Attribute</th><th>Validation</th></tr><tr><td>School</td><td>9. An option must be selected</td></tr><tr><td>Position</td><td> An option must be selected </td></tr></table>	Attribute	Validation	School	9. An option must be selected	Position	An option must be selected											
Attribute	Validation																		
School	9. An option must be selected																		
Position	An option must be selected																		

			Step 10: The system verifies that no duplicate User exists on the system by comparing the following: [alt]															
			<table><tr><th>Input</th><th>Attribute</th><th>Entity</th></tr><tr><td>Username</td><td>Username</td><td>User</td></tr></table>	Input	Attribute	Entity	Username	Username	User									
	Input	Attribute	Entity															
	Username	Username	User															
			Step 11: The system prompts the Administrator to confirm to add the new User with the following:															
			<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Confirmation</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed.</td></tr><tr><td>"Are you sure you want to add this User?"</td><td>Dialog Box Message</td><td>Used to inform the Administrator of the action.</td></tr><tr><td>Yes</td><td>Button</td><td>Used to confirm the action.</td></tr><tr><td>No</td><td>Button</td><td>Used to decline the action.</td></tr></table>	Text/Label	Control Type	Additional Info	Confirmation	Dialog Box Title	Used to indicate what type of message is being displayed.	"Are you sure you want to add this User?"	Dialog Box Message	Used to inform the Administrator of the action.	Yes	Button	Used to confirm the action.	No	Button	Used to decline the action.
	Text/Label	Control Type	Additional Info															
	Confirmation	Dialog Box Title	Used to indicate what type of message is being displayed.															
	"Are you sure you want to add this User?"	Dialog Box Message	Used to inform the Administrator of the action.															
	Yes	Button	Used to confirm the action.															
No	Button	Used to decline the action.																

	Step 12: The Administrator clicks the “Yes” button. [alt]		Step 13: The User is assigned a unique UserID in the User table which is set to the next consecutive number after the previous UserID in the database, the system then creates and saves the new user in the User table. The following attributes are saved in the database by using a FIRESQL INSERT query: User UserID Name Surname Email Username Password PhoneNumber Password AdminSchool AdminSchoolID SchoolID Position		
			Step 14: The system notifies the Administrator that the User has been added successfully to the database with the following: [alt] <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr></table>	Text/Label	Control Type
Text/Label	Control Type	Additional Info			

			Success	Dialog Box Title	Used to indicate what type of message is being displayed.
			"User has been added successfully"	Dialog Box Message	Used to inform the Administrator of the result.
			Ok	Button	Used to accept the result.
				Step 15: The system creates and saves the following in the database by using a FIRESQL INSERT query: AuditTrail Table AuditID DateTime TransactionDescription UserID	
Alternate Courses:	Alt Step 6: The Administrator clicks the "Cancel" button. Terminate Use Case.				
	Alt Step 7: In the case of the administrator selecting the Mechanic UserType Option				
	Text/Label	Control Type	Additional Info		
	Mechanic	Label	Heading to show which option the user picked		
	Work Day	Label	Used to indicate what the corresponding		

			Combo box is used for.
School	Label		Used to indicate what the corresponding Combo box is used for.
Qualification	Label		Used to indicate what the corresponding Textbox is used for.
Work Day	Combo Box		Used to Select A day of the week which the mechanic will come
School	Combo Box		Used to Select the School and is Populated by contents from the ERD with options as indicated in the table below.
Qualification	Textbox		Use to show the Mechanics qualifications
"Return"	Button		Used to go back to the create user form

"Create" Button Used to submit the form

The School Combo box options are populated by using a FIRESQL READ query with data from the following:

Entity	Attributes
School	SchoolName
MechanicSchool	WorkDayID
Mechanic	Qualification

Alt Step 8a: In the case of an empty Text box, the system notifies the Administrator of the Input error below the input field with the following:

Text/Label	Control Type	Additional Info
"Please fill this field."	Label (with a light red background color fill)	Used to inform the Administrator of the result.
"X"	Icon	Used to indicate an error result.

Redirect to Step 5.

Alt Step 8b: In the case of a Combo box option not selected, the system notifies the Administrator of the Input error below the input field with the following:

Text/Label	Control Type	Additional Info
"Please select an option."	Label (with a light red background color fill)	Used to inform the Administrator of the result.
"X"	Icon	Used to indicate an error result.

	Redirect to Step 5.		
	Alt Step 9: In the case of invalid input being entered, the system notifies the Administrator of the Format error below the input field with the following:		
	Text/Label	Control Type	Additional Info
	"Invalid input."	Label (with a light red background color fill)	Used to inform the Administrator of the result.
	"X"	Icon	Used to indicate an error result.
	Redirect to Step 5.		
	Alt Step 10: The inputted User already exists in the system; the system will notify the Administrator of the duplication error with the following:		
	Text/Label	Control Type	Additional Info
	Error	Dialog Box Title	Used to indicate what type of message is being displayed.
	"The User already exists. Please try again."	Dialog Box Message	Used to inform the Administrator of the result.
	Ok	Button	Used to accept the result.
	Redirect to Step 5.		
	Alt Step 12: The Administrator declines to add the new User by clicking the "No" button. Redirect to Step 5.		

	Alt Step 14: The User was not added successfully to the database, the system will notify the Administrator of the save error with the following:		
	Text/Label	Control Type	Additional Info
	Failed	Dialog Box Title	Used to indicate what type of message is being displayed
	"Create unsuccessful. Please try again."	Dialog Box Message	Used to inform the Administrator of the result.
	Ok	Button	Used to accept the result.
	Redirect to Step 5.		
Conclusion:	The use case concludes when the Administrator has been notified that the User has been added successfully.		
Post-Condition:	• A new User is added to the system with the relevant details. • The AuditTrail table is successfully updated. • The User tables are successfully updated with the details being created.		
Business Rules	• Only an Administrator can add a User to the system.		
Implementation Constraints and Specifications:	• None		
Assumptions:	• None		
Open Issues:	• None		

2.3.2.5. Search User

System Name:	VVs
Author:	Hansen Li
Date:	18/06/2020
Version:	1.1

Use Case Name:	Search User	Use Case Type:
Use Case ID:	1.5	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	Administrator	
Primary System Actor	None	
Other Participating Actors:	None	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process of an Administrator searching a User on the system. The Use Case starts when the Administrator navigates by selecting the "User Management" option under the "User Management" Combo box in the Navigation bar. The system populates the table with the information of all the Users.	
Pre-Condition:	The Administrator must be logged into the system.	
Trigger:	The Administrator requests to search for a User on the system.	

Typical Course Of Events:	Actor Action	System Response																	
		Manual Action	Automated Action																
	Step 1: The Administrator selects the “User Management” option under the “User Management” Combo box in the Navigation bar.		Step 2: The system Displays the “User Management” screen and populates it with the following details:																
			<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Add User</td><td>Button</td><td>Used to load the “User” screen.</td></tr><tr><td>Search</td><td>Text-Box</td><td>Used for capturing Search criteria for the specific User.</td></tr><tr><td>User Management</td><td>Table</td><td>Displays a list of Users.</td></tr><tr><td>“Pen icon”</td><td>Button</td><td>Used to navigate to the specific User view.</td></tr></table>		Text/Label	Control Type	Additional Info	Add User	Button	Used to load the “User” screen.	Search	Text-Box	Used for capturing Search criteria for the specific User.	User Management	Table	Displays a list of Users.	“Pen icon”	Button	Used to navigate to the specific User view.
			Text/Label	Control Type	Additional Info														
			Add User	Button	Used to load the “User” screen.														
			Search	Text-Box	Used for capturing Search criteria for the specific User.														
			User Management	Table	Displays a list of Users.														
			“Pen icon”	Button	Used to navigate to the specific User view.														
			The table is populated by using a FIRESQL READ query with data from the following: (in the table we have a full name made up of FirstName and Surname)																
<table><tr><th>Entity</th><th>Attributes</th><th>Input Name</th></tr><tr><td>User</td><td>FirstName</td><td>First Name</td></tr><tr><td>User</td><td>Surname</td><td>Surname</td></tr><tr><td>UserType</td><td>UserType</td><td>User Type</td></tr></table>			Entity	Attributes	Input Name	User	FirstName	First Name	User	Surname	Surname	UserType	UserType	User Type					
Entity			Attributes	Input Name															
User	FirstName	First Name																	
User	Surname	Surname																	
UserType	UserType	User Type																	

	Step 3: The Administrator enters the search criteria that they wish to search by. They do so by typing in a keyword in the "Search" Text box.													
	Step 4: The Administrator clicks on the "Search" button.		Step 5: The system uses a FIRESQL READ query to READ the following details in the database: [alt] User UserID Name Surname Email Username UserType Password PhoneNumber											
			Step 6: The following details are loaded and used to populate the "User Management" table by using a FIRESQL READ query: <table><tr><th>Output</th><th>Attribute</th><th>Entity</th></tr><tr><td>UserType</td><td>UserType</td><td>UserType</td></tr><tr><td>Name</td><td>Name</td><td>User</td></tr><tr><td>Surname</td><td>Surname</td><td>User</td></tr></table>	Output	Attribute	Entity	UserType	UserType	UserType	Name	Name	User	Surname	Surname
Output	Attribute	Entity												
UserType	UserType	UserType												
Name	Name	User												
Surname	Surname	User												

	Step 7: The Administrator clicks on the "Pen icon" User Button in the "Action" table column.		Step 8: The System will load the "User" screen and displays it with the following controls and layout:		
			Text/Label	Control Type	Additional Info
			Menu	Menu-icon	Used to Navigate the side menu-bar.
			User	Label	Heading for the pop-up displayed.
			User Picture	Picture image	Displays the uploaded picture.
			Name	Label	Used to indicate what the corresponding text box is used for.
			Surname	Label	Used to indicate what the corresponding text box is used for.
			Email	Label	Used to indicate what the corresponding text box is used for.
			Username	Label	Used to indicate what the corresponding

			text box is used for.	
			User Type	Label
			Used to indicate what the corresponding Combo box is used for.	
			Phone Number	Label
			Used to indicate what the corresponding text box is used for.	
			Name	Text box
			Used to capture text for the Name details.	
			Surname	Text box
			Used to capture text for the Surname details.	
			Email	Text box
			Used to capture text for the Email details.	
			Username	Text box
			Used to capture text for the Username details.	
			User Type	Combo box
			Used to Select the User Type and is Populated by contents from the ERD with	

options as indicated in the table below.

Phone Number	Text Box	Used to capture text for the Phone Number details.
Update	Button	Used to Save the new User Details.
Delete	Button	Used to cancel the action of adding a User

The Bicycle Section Combo box options are populated with data from the following by using a FIRESQL READ query:

Entity	Attributes
User Type	UserType

The following fields are populated with data retrieved from the database where the UserID is the same as the one searched by using a FIRESQL READ query:

Output	Attribute	Entity
Name	Name	User
Surname	Surname	User

			Email	Email	User									
			Username	Username	User									
			User Type	UserType	UserType									
			Phone Number	PhoneNumber	User									
				Step 9: The system creates and saves the following in the database by using a FIRESQL INSERT query: AuditTrail Table AuditID DateTime TransactionDescription UserID										
Alternate Courses:	Alt Step 5: The specific User does not exist in the database. The system will notify the Administrator of this with the following popup:													
	<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>"User not found."</td><td>Label</td><td>Used to inform the Administrator of the result.</td></tr><tr><td>"Ok"</td><td>Button</td><td>Used to close the message</td></tr></table>					Text/Label	Control Type	Additional Info	"User not found."	Label	Used to inform the Administrator of the result.	"Ok"	Button	Used to close the message
	Text/Label	Control Type	Additional Info											
	"User not found."	Label	Used to inform the Administrator of the result.											
"Ok"	Button	Used to close the message												
Redirect to Step 3.														
Conclusion:	The use case narrative concludes when the searched User details are displayed.													
Post-Condition:	The User has been searched successfully.													

	• The AuditTrail table is successfully updated.
Business Rules	• Only an Administrator can search for a User on the system.
Implementation Constraints and Specifications:	• None
Assumptions:	• None
Open Issues:	• None

2.3.2.6. Update User

System Name:	VVs
Author:	Hansen Li
Date:	18/06/2020
Version:	1.1

Use Case Name:	Update User	Use Case Type:
Use Case ID:	1.6	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	Administrator	
Primary System Actor	None	
Other Participating Actors:	None	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process of an Administrator updating an existing User's details in the system. The Use Case starts when the Administrator navigates by selecting the "User Management" option under the "User Management" Combo box in the Navigation bar. From there, they search for the User they want to update and the system will load the "User" screen with the User details. The system populates the relevant Combo boxes with information of the searched User, and the Administrator can make their changes and save.	
Pre-Condition:	The Administrator must be logged into the system.	

Trigger:	The Administrator requests to update an existing User to the system.										
Typical Course Of Events:	Actor Action	System Response									
		Manual Action	Automated Action								
	Step 1: The Administrator selects the "User Management" option under the "User Management" Combo box in the Navigation bar.		Step 2: The System invokes Use Case 1.5 "Search User" and displays the "User" screen with the data of the searched User.								
	Step 3: Administrator updates the following User details: - Name - Surname - Email - Username - User Type - Phone Number										
	Step 4: The Administrator clicks the "Update" button. [alt]		Step 5: The system displays the following form next if the selected User Type was an Admin: [alt] <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>School Administrator</td><td>Label</td><td>Heading to show which option the user picked</td></tr><tr><td>Position</td><td>Label</td><td>Used to indicate what the corresponding Combo box is used for.</td></tr></table>	Text/Label	Control Type	Additional Info	School Administrator	Label	Heading to show which option the user picked	Position	Label
Text/Label	Control Type	Additional Info									
School Administrator	Label	Heading to show which option the user picked									
Position	Label	Used to indicate what the corresponding Combo box is used for.									

			School	Label	Used to indicate what the corresponding Combo box is used for.
			Position	Combo Box	Used to Select the Administrator Position and is Populated by contents from the ERD with options as indicated in the table below.
			School	Combo Box	Used to Select the School and is Populated by contents from the ERD with options as indicated in the table below.
			"Return"	Button	Used to go back to the Update user form
			"Update"	Button	Used to submit the form
			The School Combo box options are populated by using a FIRESQL READ query with data from the following:		
			Entity	Attributes	
			School	SchoolName	

			AdminSchool	Position																			
	Step 6: Administrator updates the following User details if needed: - School - Position		Step 7: The system captures and ensures that all of the fields are filled in and an option is selected in the Combo box. [alt]																				
			Step 8: The system validates that the captured details are in the correct format: [alt] User table <table><tr><th>Attribute</th><th>Validation</th></tr><tr><td>Name</td><td>10.Max 50 characters 11.Text characters only</td></tr><tr><td>Surname</td><td> - Max 50 characters - Text characters only </td></tr><tr><td>Email</td><td> - Max 50 characters - Text characters only </td></tr><tr><td>Username</td><td> - Max 50 characters - Text characters only </td></tr><tr><td>User Type</td><td> - Option must be selected </td></tr><tr><td>Phone Number</td><td> - Max 50 characters - Text characters only </td></tr></table> Admin Table <table><tr><th>Attribute</th><th>Validation</th></tr><tr><td>School</td><td>12.An option must be selected</td></tr><tr><td>Position</td><td> - An option must be selected </td></tr></table>		Attribute	Validation	Name	10.Max 50 characters 11.Text characters only	Surname	- Max 50 characters - Text characters only	Email	- Max 50 characters - Text characters only	Username	- Max 50 characters - Text characters only	User Type	Option must be selected	Phone Number	- Max 50 characters - Text characters only	Attribute	Validation	School	12.An option must be selected	Position
Attribute	Validation																						
Name	10.Max 50 characters 11.Text characters only																						
Surname	- Max 50 characters - Text characters only																						
Email	- Max 50 characters - Text characters only																						
Username	- Max 50 characters - Text characters only																						
User Type	Option must be selected																						
Phone Number	- Max 50 characters - Text characters only																						
Attribute	Validation																						
School	12.An option must be selected																						
Position	An option must be selected																						

			Step 9: The system prompts the Administrator to confirm the update of the User's details.															
			<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Confirmation</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed.</td></tr><tr><td>"Are you sure you want to update this User?"</td><td>Dialog Box Message</td><td>Used to inform the Administrator of the action.</td></tr><tr><td>Yes</td><td>Button</td><td>Used to confirm the action.</td></tr><tr><td>No</td><td>Button</td><td>Used to decline the action.</td></tr></table>	Text/Label	Control Type	Additional Info	Confirmation	Dialog Box Title	Used to indicate what type of message is being displayed.	"Are you sure you want to update this User?"	Dialog Box Message	Used to inform the Administrator of the action.	Yes	Button	Used to confirm the action.	No	Button	Used to decline the action.
Text/Label	Control Type	Additional Info																
Confirmation	Dialog Box Title	Used to indicate what type of message is being displayed.																
"Are you sure you want to update this User?"	Dialog Box Message	Used to inform the Administrator of the action.																
Yes	Button	Used to confirm the action.																
No	Button	Used to decline the action.																

	Step 10: The Administrator clicks the “Yes” button. [alt]		Step 11: The system then updates and saves the updated User details in the User table. The following attributes are updated in the database by using a FIRESQL UPDATE query: User UserID Name Surname Email Username UserType PhoneNumber AdminSchool SchoolID Position					
			Step 12: The system notifies the Administrator that the User has been updated successfully in the database with the following: [alt] <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Success</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed.</td></tr></table>	Text/Label	Control Type	Additional Info	Success	Dialog Box Title
Text/Label	Control Type	Additional Info						
Success	Dialog Box Title	Used to indicate what type of message is being displayed.						

			"The User has been updated successfully" Dialog Box Message Used to inform the Administrator of the result.
			Ok Button Used to accept the result.
			Step 13: The system updates and saves the following in the database by using a FIRESQL INSERT query: AuditTrail Table AuditID DateTime TransactionDescription UserID
Alternate Courses:	Alt Step 4a: The Administrator clicks the "Cancel" button. Terminate Use Case.		
	Alt Step 4b: In the case of an empty Text box, the system notifies the Administrator of the Input error below the input field with the following:		
	Text/Label "Please fill this field."	Control Type Label (with a light blue background color fill)	Additional Info Used to inform the Administrator of the result.
	"X"	Icon	Used to indicate an error result.
	Redirect to Step 3.		
	Alt Step 4c: In the case of a Combo box option not selected, the system notifies the Administrator of the Input error below the input field with the following:		

	Text/Label	Control Type	Additional Info
	"Please select an option."	Label (with a light blue background color fill)	Used to inform the Administrator of the result.
	"X"	Icon	Used to indicate an error result.
	Redirect to Step 3.		
	Alt Step 5: In the case of the administrator selecting the Mechanic UserType Option		
	Text/Label	Control Type	Additional Info
	Mechanic	Label	Heading to show which option the user picked
	Work Day	Label	Used to indicate what the corresponding Combo box is used for.
	School	Label	Used to indicate what the corresponding Combo box is used for.
	Qualification	Label	Used to indicate what the corresponding Textbox is used for.

Work Day	Combo Box	Used to Select A day of the week which the mechanic will come
School	Combo Box	Used to Select the School and is Populated by contents from the ERD with options as indicated in the table below.
Qualification	Textbox	Use to show the Mechanics qualifications
"Return"	Button	Used to go back to the Update user form
"Update"	Button	Used to submit the form

The School Combo box options are populated by using a FIRESQL READ query with data from the following:

Entity	Attributes
School	SchoolName
MechanicSchool	WorkDayID
Mechanic	Qualification

Alt Step 7: In the case of invalid input being entered, the system notifies the Administrator of the Format error below the input field with the following:

Text/Label	Control Type	Additional Info

	"Invalid input."	Label (with a light yellow background color fill)	Used to inform the Administrator of the result.												
	"X"	Icon	Used to indicate an error result.												
	Redirect to Step 3.														
	Alt Step 10: The Administrator declines to update the User by clicking the "No" button. Redirect to Step 4.														
	Alt Step 12: The User was not updated successfully in the database. The system will notify the Administrator of the update error with the following:														
	<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Failed</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed</td></tr><tr><td>"Update unsuccessful. Please try again."</td><td>Dialog Box Message</td><td>Used to inform the Administrator of the result.</td></tr><tr><td>Ok</td><td>Button</td><td>Used to accept the result.</td></tr></table>			Text/Label	Control Type	Additional Info	Failed	Dialog Box Title	Used to indicate what type of message is being displayed	"Update unsuccessful. Please try again."	Dialog Box Message	Used to inform the Administrator of the result.	Ok	Button	Used to accept the result.
	Text/Label	Control Type	Additional Info												
	Failed	Dialog Box Title	Used to indicate what type of message is being displayed												
	"Update unsuccessful. Please try again."	Dialog Box Message	Used to inform the Administrator of the result.												
	Ok	Button	Used to accept the result.												
Redirect to Step 3.															
Conclusion:	The use case concludes when the Administrator has been notified that the User has been updated successfully.														
Post-Condition:	• An existing User is updated in the system with the relevant details. • The AuditTrail table is successfully updated. • The User tables are successfully updated with the details being updated.														

Business Rules	• Only an Administrator can update a User to the system.
Implementation Constraints and Specifications:	• None
Assumptions:	• None
Open Issues:	• None

2.3.2.7. Delete User

System Name:	VVs
Author:	Hansen Li
Date:	18/06/2020
Version:	1.0

Use Case Name:	Delete User	Use Case Type:
Use Case ID:	1.7	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	Administrator	
Primary System Actor	None	
Other Participating Actors:	None	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process of an Administrator deleting an existing User's details from the system. The Use Case starts when the Administrator navigates by selecting the "User Management" option under the "User Management" Combo box in the Navigation bar. From there, they search for the User they want to delete and the system displays the "User" screen with the User details to be deleted. The system populates the relevant Combo boxes with information of the searched User, and the Administrator can make their deletion.	
Pre-Condition:	The Administrator must be logged into the system.	

Trigger:	The Administrator requests to delete an existing User from the system.			
Typical Course Of Events:	Actor Action	System Response		
		Manual Action	Automated Action	
	Step 1: The Administrator selects the "User Management" option under the "User Management" Combo box in the Navigation bar.		Step 2: The System invokes Use Case 1.5 "Search User" and displays the "User" screen with the data of the searched User.	
	Step 3: The Administrator clicks on the "Delete" button.[alt]		Step 4: The system prompts the Administrator to confirm the deletion of the User's details with the following:	
			Text/Label	Control Type
Confirmation			Dialog Box Title	Used to indicate what type of message is being displayed.
"Are you sure you want to delete this User?"			Dialog Box Message	Used to inform the Administrator of the action.
Yes			Button	Used to confirm the action.
No	Button	Used to decline the action.		

	Step 5: The Administrator clicks the “Yes” button. [alt]		Step 6: The system then removes the User details in the User table. The following attributes are deleted in the database by using a FIRESQL DELETE query: User UserID Name Surname Email Username UserType PhoneNumber AdminSchool AdminSchoolID Position SchoolID Position					
			Step 7: The system notifies the Administrator that the User has been deleted successfully from the database with the following: [alt] <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Success</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed.</td></tr></table>	Text/Label	Control Type	Additional Info	Success	Dialog Box Title
Text/Label	Control Type	Additional Info						
Success	Dialog Box Title	Used to indicate what type of message is being displayed.						

			"The User has been deleted successfully" Dialog Box Message Used to inform the Administrator of the result.
			Ok Button Used to accept the result.
			Step 8: The system updates and saves the following in the database by using a FIRESQL INSERT query: AuditTrail Table AuditID DateTime TransactionDescription UserID
Alternate Courses:	Alt Step 3: The Administrator clicks the "Cancel" button. Terminate Use Case.		
	Alt Step 5: The Administrator declines to delete the User by clicking the "No" button. Redirect to Step 2.		
	Alt Step 7: The Administrator is notified of the unsuccessful delete action with the following:		
	Text/Label Failed	Control Type Dialog Box Title	Additional Info Used to indicate what type of message is being displayed
	"Delete unsuccessful. Please try again."	Dialog Box Message	Used to inform the Administrator of the result.

	Ok Button Used to accept the result. Redirect to Step 2.
Conclusion:	The use case concludes when the Administrator has been notified that the User has been successfully deleted.
Post-Condition:	• An existing User is deleted from the system with all the relevant details being removed. • The AuditTrail table is successfully updated. • The User tables are successfully updated with the deleted details being removed.
Business Rules	• Only an Administrator can delete a User from the system.
Implementation Constraints and Specifications:	• None
Assumptions:	• None
Open Issues:	• None

2.3.2.8. Add User Type

System Name:	VVs
Author:	Hansen Li
Date:	18/06/2020
Version:	1.1

Use Case Name:	Add User Type	Use Case Type:
Use Case ID:	1.8	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	Administrator	
Primary System Actor	None	
Other Participating Actors:	None	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process of an Administrator adding a new User Type to the system. The Use Case starts when the Administrator navigates by selecting the "User Type Management" option under the "User Management" Combo box in the Navigation bar. From there, they click the "Add User Type" button and the system will load the "User Type" screen. The system populates the relevant Combo boxes with information necessary for selection, and the corresponding labels, text boxes and buttons, icons, logos, and banners.	
Pre-Condition:	The Administrator must be logged into the system.	

Trigger:	The Administrator requests to add a new User Type to the system.																	
Typical Course Of Events:	Actor Action	System Response																
		Manual Action	Automated Action															
	Step 1: The Administrator selects the “User Type” option under the “User Type Management” Combo box in the Navigation bar.		Step 2: The system Displays the “User Type Management” screen and populates it with the following details:															
			<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Add User Type</td><td>Button</td><td>Used to load the “User Type” screen.</td></tr><tr><td>Search</td><td>Text-Box</td><td>Used for capturing Search criteria for the specific User Type.</td></tr><tr><td>User Type Management</td><td>Table</td><td>Displays a list of User Types.</td></tr><tr><td>“Pen icon”</td><td>Button</td><td>Used to navigate to the specific User Type view.</td></tr></table>	Text/Label	Control Type	Additional Info	Add User Type	Button	Used to load the “User Type” screen.	Search	Text-Box	Used for capturing Search criteria for the specific User Type.	User Type Management	Table	Displays a list of User Types.	“Pen icon”	Button	Used to navigate to the specific User Type view.
			Text/Label	Control Type	Additional Info													
			Add User Type	Button	Used to load the “User Type” screen.													
			Search	Text-Box	Used for capturing Search criteria for the specific User Type.													
			User Type Management	Table	Displays a list of User Types.													
			“Pen icon”	Button	Used to navigate to the specific User Type view.													
			The table is populated by using a FIRESQL READ query with data from the following:															
			<table><tr><th>Entity</th><th>Attributes</th><th>Input Name</th></tr><tr><td>UserType</td><td>UserType</td><td>User Type</td></tr><tr><td>UserType</td><td>UserPermissions</td><td>User Permissions</td></tr></table>	Entity	Attributes	Input Name	UserType	UserType	User Type	UserType	UserPermissions	User Permissions						
			Entity	Attributes	Input Name													
UserType	UserType	User Type																
UserType	UserPermissions	User Permissions																

	Step 3: The Administrator clicks on the "Add User Type" button.		Step 4: The system will load the "User Type" screen and displays it with the following controls and layout:																					
			<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Menu</td><td>Menu-icon</td><td>Used to Navigate the side menu-bar.</td></tr><tr><td>User Type</td><td>Label</td><td>Heading for the pop-up displayed.</td></tr><tr><td>User Type</td><td>Label</td><td>Used to indicate what the corresponding text box is used for.</td></tr><tr><td>User Permissions</td><td>Label</td><td>Used to indicate what the corresponding Combo box is used for.</td></tr><tr><td>User Type</td><td>Text box</td><td>Used to capture text for the User Type Name details.</td></tr><tr><td>UserPermissions</td><td>Combo box</td><td>Used to Select the Permission level with this Part and is Populated by hardcoding.</td></tr></table>	Text/Label	Control Type	Additional Info	Menu	Menu-icon	Used to Navigate the side menu-bar.	User Type	Label	Heading for the pop-up displayed.	User Type	Label	Used to indicate what the corresponding text box is used for.	User Permissions	Label	Used to indicate what the corresponding Combo box is used for.	User Type	Text box	Used to capture text for the User Type Name details.	UserPermissions	Combo box	Used to Select the Permission level with this Part and is Populated by hardcoding.
			Text/Label	Control Type	Additional Info																			
			Menu	Menu-icon	Used to Navigate the side menu-bar.																			
			User Type	Label	Heading for the pop-up displayed.																			
			User Type	Label	Used to indicate what the corresponding text box is used for.																			
			User Permissions	Label	Used to indicate what the corresponding Combo box is used for.																			
			User Type	Text box	Used to capture text for the User Type Name details.																			
UserPermissions	Combo box	Used to Select the Permission level with this Part and is Populated by hardcoding.																						

			Create	Button	Used to Save the new User Type Details.						
			Cancel	Button	Used to cancel the action of adding a User Type.						
			Entity	Attributes	Input Name						
			UserType	UserType	User Type						
			UserType	UserPermissions	User Permissions						
	Step 5: Administrator inputs the following User Type details: - User Type - Permissions										
	Step 6: The Administrator clicks the "Create" button. [alt]				Step 7: The system captures and ensures that all of the fields are filled in and an option is selected in the Combo box. [alt]						
					Step 8: The system validates that the captured details are in the correct format: [alt]						
					User Type table <table><thead><tr><th>Attribute</th><th>Validation</th></tr></thead><tbody><tr><td>UserType</td><td>13.Max 50 characters 14.Text characters only</td></tr><tr><td>UserPermissions</td><td> A permission option must be selected </td></tr></tbody></table>	Attribute	Validation	UserType	13.Max 50 characters 14.Text characters only	UserPermissions	A permission option must be selected
	Attribute	Validation									
UserType	13.Max 50 characters 14.Text characters only										
UserPermissions	A permission option must be selected										
				Step 9: The system verifies that no duplicate User Type exists on the system by comparing the following: [alt]							

			<table><tr><td>Input</td><td>Attribute</td><td>Entity</td></tr><tr><td>User Type</td><td>UserType</td><td>UserType</td></tr></table>	Input	Attribute	Entity	User Type	UserType	UserType									
	Input	Attribute	Entity															
	User Type	UserType	UserType															
			Step 10: The system prompts the Administrator to confirm to add the new User Type with the following: <table><tr><td>Text/Label</td><td>Control Type</td><td>Additional Info</td></tr><tr><td>Confirmation</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed.</td></tr><tr><td>"Are you sure you want to add this User Type"</td><td>Dialog Box Message</td><td>Used to inform the Administrator of the action.</td></tr><tr><td>Yes</td><td>Button</td><td>Used to confirm the action.</td></tr><tr><td>No</td><td>Button</td><td>Used to decline the action.</td></tr></table>	Text/Label	Control Type	Additional Info	Confirmation	Dialog Box Title	Used to indicate what type of message is being displayed.	"Are you sure you want to add this User Type"	Dialog Box Message	Used to inform the Administrator of the action.	Yes	Button	Used to confirm the action.	No	Button	Used to decline the action.
	Text/Label	Control Type	Additional Info															
	Confirmation	Dialog Box Title	Used to indicate what type of message is being displayed.															
	"Are you sure you want to add this User Type"	Dialog Box Message	Used to inform the Administrator of the action.															
	Yes	Button	Used to confirm the action.															
	No	Button	Used to decline the action.															
	Step 11: The Administrator clicks the "Yes" button. [alt]		Step 12: The UserType is assigned a unique UserTypeID in the UserType table which is set to the next consecutive number after the previous UserTypeID in the database, the system then creates and saves the new User Type in the UserType table. The following attributes are saved in the database by using a FIRESQL INSERT query: UserType Table <table><tr><td>UserTypeID</td></tr><tr><td>Name</td></tr><tr><td>UserPermissions</td></tr></table>	UserTypeID	Name	UserPermissions												
UserTypeID																		
Name																		
UserPermissions																		

			Step 13: The system notifies the Administrator that the User Type has been added successfully to the database with the following: [alt]												
			<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Success</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed.</td></tr><tr><td>"User Type has been added successfully"</td><td>Dialog Box Message</td><td>Used to inform the Administrator of the result.</td></tr><tr><td>Ok</td><td>Button</td><td>Used to accept the result.</td></tr></table>	Text/Label	Control Type	Additional Info	Success	Dialog Box Title	Used to indicate what type of message is being displayed.	"User Type has been added successfully"	Dialog Box Message	Used to inform the Administrator of the result.	Ok	Button	Used to accept the result.
			Text/Label	Control Type	Additional Info										
			Success	Dialog Box Title	Used to indicate what type of message is being displayed.										
"User Type has been added successfully"	Dialog Box Message	Used to inform the Administrator of the result.													
Ok	Button	Used to accept the result.													
		Step 14: The system creates and saves the following in the database by using a FIRESQL INSERT query:													
		<table><tr><th>AuditTrail Table</th></tr><tr><td>AuditID</td></tr><tr><td>DateTime</td></tr><tr><td>TransactionDescription</td></tr><tr><td>UserID</td></tr></table>	AuditTrail Table	AuditID	DateTime	TransactionDescription	UserID								
AuditTrail Table															
AuditID															
DateTime															
TransactionDescription															
UserID															
Alternate Courses:	Alt Step 6: The Administrator clicks the "Cancel" button. Terminate Use Case.														
	Alt Step 7a: In the case of an empty Text box, the system notifies the Administrator of the Input error below the input field with the following:														
	<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr></table>	Text/Label	Control Type	Additional Info											
Text/Label	Control Type	Additional Info													

	"Please fill this field."	Label (with a light red background color fill)	Used to inform the Administrator of the result.
	"X"	Icon	Used to indicate an error result.
	Redirect to Step 5.		

Alt Step 7b: In the case of a Combo box option not selected, the system notifies the Administrator of the Input error below the input field with the following:

Text/Label	Control Type	Additional Info
"Please select an option."	Label (with a light red background color fill)	Used to inform the Administrator of the result.
"OK"	Button	Used to return to the screen

Redirect to Step 5.

Alt Step 8: In the case of invalid input being entered, the system notifies the Administrator of the Format error below the input field with the following:

Text/Label	Control Type	Additional Info
"Invalid input."	Label (with a light red background color fill)	Used to inform the Administrator of the result.
"OK"	Button	Used to return to the screen

Redirect to Step 5.

Alt Step 9: The inputted UserType already exists in the system; the system will notify the Administrator of the duplication error with the following:

Text/Label	Control Type	Additional Info
Error	Dialog Box Title	Used to indicate what type of message is being displayed.
"The User Type already exists. Please try again."	Dialog Box Message	Used to inform the Administrator of the result.
Ok	Button	Used to accept the result.

Redirect to Step 5.

Alt Step 11: The Administrator declines to add the new User Type by clicking the "No" button. Redirect to Step 5.

Alt Step 13: The User Type was not added successfully to the database, the system will notify the Administrator of the save error with the following:

Text/Label	Control Type	Additional Info
Failed	Dialog Box Title	Used to indicate what type of message is being displayed
"Create unsuccessful. Please try again."	Dialog Box Message	Used to inform the Administrator of the result.
Ok	Button	Used to accept the result.

	Redirect to Step 5.
Conclusion:	The use case concludes when the Administrator has been notified that the User Type has been added successfully.
Post-Condition:	• A new UserType is added to the system with the relevant details. • The AuditTrail table is successfully updated. • The User Type tables are successfully updated.
Business Rules	• Only an Administrator can add a Part to the system.
Implementation Constraints and Specifications:	• None
Assumptions:	• None
Open Issues:	• None

2.3.2.9. Search User Type

System Name:	VVs
Author:	Hansen Li
Date:	18/06/2020
Version:	1.1

Use Case Name:	Search User Type	Use Case Type:
Use Case ID:	1.9	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	Administrator	
Primary System Actor	None	
Other Participating Actors:	None	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process of an Administrator searching a User Type on the system. The Use Case starts when the Administrator navigates by selecting the "User Type Management" option under the "User Management" Combo box in the Navigation bar. The system populates the table with the information of all the bicycle parts.	
Pre-Condition:	The Administrator must be logged into the system.	
Trigger:	The Administrator requests to search for a User Type on the system.	

Typical Course Of Events:	Actor Action	System Response																	
		Manual Action	Automated Action																
	Step 1: The Administrator selects the "User Type Management" option under the "User Management" Combo box in the Navigation bar.		Step 2: The system Displays the "User Type Management" screen and populates it with the following details:																
			<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Add User Type</td><td>Button</td><td>Used to load the "User Type" screen.</td></tr><tr><td>Search</td><td>Text-Box</td><td>Used for capturing Search criteria for the specific User Type.</td></tr><tr><td>User Type Management</td><td>Table</td><td>Displays a list of User Types.</td></tr><tr><td>"Pen icon"</td><td>Button</td><td>Used to navigate to the specific User Type view.</td></tr></table>		Text/Label	Control Type	Additional Info	Add User Type	Button	Used to load the "User Type" screen.	Search	Text-Box	Used for capturing Search criteria for the specific User Type.	User Type Management	Table	Displays a list of User Types.	"Pen icon"	Button	Used to navigate to the specific User Type view.
			Text/Label	Control Type	Additional Info														
			Add User Type	Button	Used to load the "User Type" screen.														
			Search	Text-Box	Used for capturing Search criteria for the specific User Type.														
			User Type Management	Table	Displays a list of User Types.														
			"Pen icon"	Button	Used to navigate to the specific User Type view.														
			The table is populated by using a FIRESQL READ query with data from the following:																
<table><tr><th>Entity</th><th>Attributes</th><th>Input Name</th></tr><tr><td>UserType</td><td>UserType</td><td>User Type</td></tr><tr><td>UserType</td><td>UserPermissions</td><td>User Permissions</td></tr></table>			Entity	Attributes	Input Name	UserType	UserType	User Type	UserType	UserPermissions	User Permissions								
Entity	Attributes	Input Name																	
UserType	UserType	User Type																	
UserType	UserPermissions	User Permissions																	
Step 3: The Administrator enters the search criteria that they wish																			

	to search by. They do so by typing in a keyword in the "Search" Text box.														
	Step 4: The Administrator clicks on the "Search" button.		Step 5: The system uses a FIRESQL READ query to READ the following details in the database: [alt] UserType Table UserTypeID Name UserPermissions												
			Step 6: The following details are loaded and used to populate the "User Type Management" table by using a FIRESQL READ query: <table><tr><th>Output</th><th>Attribute</th><th>Entity</th></tr><tr><td>User Type ID</td><td>UserTypeID</td><td>UserType</td></tr><tr><td>UserType</td><td>UserType</td><td>UserType</td></tr><tr><td>UserPermissions</td><td>UserPermissions</td><td>UserType</td></tr></table>	Output	Attribute	Entity	User Type ID	UserTypeID	UserType	UserType	UserType	UserType	UserPermissions	UserPermissions	UserType
	Output	Attribute	Entity												
User Type ID	UserTypeID	UserType													
UserType	UserType	UserType													
UserPermissions	UserPermissions	UserType													
Step 7: The Administrator clicks on the "Pen icon" Bicycle Part Button in the "Action" table column.		Step 8: The System will load the "User Type" screen and displays it with the following controls and layout: <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr></table>	Text/Label	Control Type	Additional Info										
Text/Label	Control Type	Additional Info													

			Menu	Menu-icon	Used to Navigate the side menu-bar.
			User Type	Label	Heading for the pop-up displayed.
			UserTypeID	Label	Used to indicate what the corresponding text box is used for.
			Type	Label	Used to indicate what the corresponding text box is used for.
			User Permissions	Label	Used to indicate what the corresponding Combo box is used for.
			UserTypeID	Label	Used to display text for the User TypeID details.
			Type	Text box	Used to capture text for the User Type Name details.

			UserPermissions	Combo box	Used to Select the Permission level with this Part and is Populated by hardcoding.
			Update	Button	Used to Save the new User Type Details.
			Delete	Button	Used to cancel the action of adding a User Type.
			The following fields are populated with data retrieved from the database where the UserTypeID is the same as the one searched by using a FIRESQL READ query:		
			Output	Attribute	Entity
			User Type	UserType	UserType
			User Permissions	UserPermissions	UserType
			Step 9: The system creates and saves the following in the database by using a FIRESQL INSERT query:		
			AuditTrail Table		
			AuditID		
			DateTime		

			TransactionDescription UserID									
Alternate Courses:	Alt Step 5: The specific User Type does not exist in the database. The system will notify the Administrator of this with the following: <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>"User Type not found."</td><td>Label</td><td>Used to inform the Administrator of the result.</td></tr><tr><td>"Ok"</td><td>Button</td><td>Used to accept the result</td></tr></table> Redirect to Step 3.			Text/Label	Control Type	Additional Info	"User Type not found."	Label	Used to inform the Administrator of the result.	"Ok"	Button	Used to accept the result
Text/Label	Control Type	Additional Info										
"User Type not found."	Label	Used to inform the Administrator of the result.										
"Ok"	Button	Used to accept the result										
Conclusion:	The use case narrative concludes when the searched User Type details are displayed.											
Post-Condition:	- The User Type has been searched successfully. - The AuditTrail table is successfully updated.											
Business Rules	Only an Administrator can search for a User Type on the system.											
Implementation Constraints and Specifications:	None											
Assumptions:	None											
Open Issues:	None											

2.3.2.10. Update User Type

System Name:	VVs
Author:	Hansen Li
Date:	18/06/2020
Version:	1.1

Use Case Name:	Update User Type	Use Case Type:
Use Case ID:	1.10	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	Administrator	
Primary System Actor	None	
Other Participating Actors:	None	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process of an Administrator updating an existing User Type's details in the system. The Use Case starts when the Administrator navigates by selecting the "User Type Management" option under the "User Management" Combo box in the Navigation bar. From there, they search for the User Type they want to update and the system will load the "User Type" screen with the User Type details. The system populates the relevant Combo boxes with information of the searched User Type, and the Administrator can make their changes and save.	
Pre-Condition:	The Administrator must be logged into the system.	

Trigger:	The Administrator requests to update an existing User Type to the system.								
Typical Course Of Events:	Actor Action	System Response							
		Manual Action	Automated Action						
	Step 1: The Administrator selects the “User Type Management” option under the “User Management” Combo box in the Navigation bar.		Step 2: The System invokes Use Case 1.9 “Search User Type” and displays the “User Type” screen with the data of the searched User Type.						
	Step 3: Administrator updates the following User Type details: - Type - Permissions								
	Step 4: The Administrator clicks the “Update User Type” button. [alt]		Step 5: The system captures and ensures that all of the fields are filled in and an option is selected in the Combo box. [alt]						
			Step 6: The system validates that the captured details are in the correct format: [alt] User Type table <table><tr><th>Attribute</th><th>Validation</th></tr><tr><td>Type</td><td>15.Max 50 characters 16.Text characters only</td></tr><tr><td>UserPermissions</td><td> An option must be selected </td></tr></table>	Attribute	Validation	Type	15.Max 50 characters 16.Text characters only	UserPermissions	An option must be selected
	Attribute	Validation							
	Type	15.Max 50 characters 16.Text characters only							
UserPermissions	An option must be selected								
		Step 7: The system prompts the Administrator to confirm the update of the Bicycle Part’s details. <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr></table>	Text/Label	Control Type	Additional Info				
Text/Label	Control Type	Additional Info							

			Confirmation	Dialog Box Title	Used to indicate what type of message is being displayed.
			"Are you sure you want to update this User Type?"	Dialog Box Message	Used to inform the Administrator of the action.
			Yes	Button	Used to confirm the action.
			No	Button	Used to decline the action.
	Step 8: The Administrator clicks the "Yes" button. [alt]		Step 9: The system then updates and saves the updated User Type details in the UserType table. The following attributes are updated in the database by using a FIRESQL UPDATE query: UserType Table Type UserPermissions		
		Step 10: The system notifies the Administrator that the Bicycle Part has been updated successfully in the database with the following: [alt]			
			Text/Label	Control Type	Additional Info
			Success	Dialog Box Title	Used to indicate what type of message is being displayed.

			"The User Type has been updated successfully" Dialog Box Message Used to inform the Administrator of the result.									
			Ok Button Used to accept the result.									
			Step 11: The system updates and saves the following in the database by using a FIRESQL INSERT query: AuditTrail Table AuditID DateTime TransactionDescription UserID									
Alternate Courses:	Alt Step 4: The Administrator clicks the "Cancel" button. Terminate Use Case.											
	Alt Step 5a: In the case of an empty Text box, the system notifies the Administrator of the Input error below the input field with the following: <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>"Please fill this field."</td><td>Label (with a light red background color fill)</td><td>Used to inform the Administrator of the result.</td></tr><tr><td>"X"</td><td>Icon</td><td>Used to indicate an error result.</td></tr></table>			Text/Label	Control Type	Additional Info	"Please fill this field."	Label (with a light red background color fill)	Used to inform the Administrator of the result.	"X"	Icon	Used to indicate an error result.
	Text/Label	Control Type	Additional Info									
"Please fill this field."	Label (with a light red background color fill)	Used to inform the Administrator of the result.										
"X"	Icon	Used to indicate an error result.										
Redirect to Step 3.												

Alt Step 5b: In the case of a Combo box option not selected, the system notifies the Administrator of the Input error below the input field with the following:

Text/Label	Control Type	Additional Info
"Please select an option."	Label (with a light red background color fill)	Used to inform the Administrator of the result.
"X"	Icon	Used to indicate an error result.

Redirect to Step 3.

Alt Step 6: In the case of invalid input being entered, the system notifies the Administrator of the Format error below the input field with the following:

Text/Label	Control Type	Additional Info
"Invalid input."	Label (with a light red background color fill)	Used to inform the Administrator of the result.
"X"	Icon	Used to indicate an error result.

Redirect to Step 3.

Alt Step 8: The Administrator declines to update the User Type by clicking the "No" button. Redirect to Step 4.

Alt Step 10: The User Type was not updated successfully in the database, the system will notify the Administrator of the update error with the following:

Text/Label	Control Type	Additional Info
Failed	Dialog Box Title	Used to indicate what type of

	message is being displayed "Update unsuccessful. Please try again." Dialog Box Message Ok Button Used to inform the Administrator of the result. Used to accept the result. Redirect to Step 3.
Conclusion:	The use case concludes when the Administrator has been notified that the UserType has been updated successfully.
Post-Condition:	• An existing User Type is updated in the system with the relevant details. • The AuditTrail table is successfully updated. • The UserType table is successfully updated.
Business Rules	• Only an Administrator can update a User Type to the system.
Implementation Constraints and Specifications:	• None
Assumptions:	• None
Open Issues:	• None

2.3.2.11. Delete User Type

System Name:	VVs
Author:	Hansen Li
Date:	18/06/2020
Version:	1.1

Use Case Name:	Delete User Type	Use Case Type:
Use Case ID:	1.11	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	Administrator	
Primary System Actor	None	
Other Participating Actors:	None	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process of an Administrator deleting an existing User Type's details from the system. The Use Case starts when the Administrator navigates by selecting the "User Type Management" option under the "User Management" Combo box in the Navigation bar. From there, they search for the User Type they want to delete and the system displays the "User Type" screen with the User Type details to be deleted. The system populates the relevant Combo boxes with information of the searched User Type, and the Administrator can make their deletion.	
Pre-Condition:	The Administrator must be logged into the system.	

Trigger:	The Administrator requests to delete an existing User Type from the system.																
Typical Course Of Events:	Actor Action	System Response															
		Manual Action	Automated Action														
	Step 1: The Administrator selects the "User Type Management" option under the "User Management" Combo box in the Navigation bar.		Step 2: The System invokes Use Case 1.9 "Search User Type" and displays the "User Type" screen with the data of the searched User Type.														
	Step 3: The Administrator clicks on the "Delete User Type" button. [alt]		Step 4: The system prompts the Administrator to confirm the deletion of the User Type's details with the following:														
			<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Confirmation</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed.</td></tr><tr><td>"Are you sure you want to delete this User Type?"</td><td>Dialog Box Message</td><td>Used to inform the Administrator of the action.</td></tr><tr><td>Yes</td><td>Button</td><td>Used to confirm the action.</td></tr><tr><td>No</td><td>Button</td><td>Used to decline the action.</td></tr></table>	Text/Label	Control Type	Additional Info	Confirmation	Dialog Box Title	Used to indicate what type of message is being displayed.	"Are you sure you want to delete this User Type?"	Dialog Box Message	Used to inform the Administrator of the action.	Yes	Button	Used to confirm the action.	No	Button
Text/Label			Control Type	Additional Info													
Confirmation			Dialog Box Title	Used to indicate what type of message is being displayed.													
"Are you sure you want to delete this User Type?"	Dialog Box Message	Used to inform the Administrator of the action.															
Yes	Button	Used to confirm the action.															
No	Button	Used to decline the action.															

	Step 5: The Administrator clicks the "Yes" button. [alt]		Step 6: The system then removes the User Type details in the UserType table. The following attributes are deleted in the database by using a FIRESQL DELETE query: UserType Table Type UserPermissions												
			Step 7: The system notifies the Administrator that the User Type has been deleted successfully from the database with the following: [alt] <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Success</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed.</td></tr><tr><td>"The User Type has been deleted successfully"</td><td>Dialog Box Message</td><td>Used to inform the Administrator of the result.</td></tr><tr><td>Ok</td><td>Button</td><td>Used to accept the result.</td></tr></table>	Text/Label	Control Type	Additional Info	Success	Dialog Box Title	Used to indicate what type of message is being displayed.	"The User Type has been deleted successfully"	Dialog Box Message	Used to inform the Administrator of the result.	Ok	Button	Used to accept the result.
	Text/Label	Control Type	Additional Info												
Success	Dialog Box Title	Used to indicate what type of message is being displayed.													
"The User Type has been deleted successfully"	Dialog Box Message	Used to inform the Administrator of the result.													
Ok	Button	Used to accept the result.													
		Step 8: The system updates and saves the following in the database by using a FIRESQL INSERT query: AuditTrail Table AuditID DateTime													

			TransactionDescription UserID
Alternate Courses:	Alt Step 3: The Administrator clicks the "Cancel" button. Terminate Use Case.		
	Alt Step 5: The Administrator declines to delete the User Type by clicking the "No" button. Redirect to Step 2.		
	Alt Step 7: The Administrator is notified of the unsuccessful delete action with the following:		
	Text/Label	Control Type	Additional Info
	Failed	Dialog Box Title	Used to indicate what type of message is being displayed
	"Delete unsuccessful. Please try again."	Dialog Box Message	Used to inform the Administrator of the result.
	Ok	Button	Used to accept the result.
	Redirect to Step 2.		
Conclusion:	The use case concludes when the Administrator has been notified that the User Type has been successfully deleted.		
Post-Condition:	- An existing User Type is deleted from the system with all the relevant details being removed. - The AuditTrail table is successfully updated. - The UserType table is successfully updated with the deleted details being removed.		
Business Rules	Only an Administrator can delete a User Type from the system.		

Implementation Constraints and Specifications:	• None
Assumptions:	• None
Open Issues:	• None

2.3.3. Sub-system 2: Mechanic Schedule

2.3.3.1. View Jobs

System Name:	VVs
Author:	Byron Donald
Date:	22/06/2020
Version:	1.0

Use Case Name:	View Jobs	Use Case Type:
Use Case ID:	2.1	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	Mechanic	
Primary System Actor	None	
Other Participating Actors:	None	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process of a Mechanic viewing their jobs for a day on the system. The Use Case starts when the Mechanic navigates from the "Mechanic Calendar" screen by selecting the day that they wish to view. The system then loads the "JobList" screen and populates the table with the information of all the jobs scheduled for the selected day.	
Pre-Condition:	The Mechanic must be logged into the system.	

Trigger:	The Mechanic requests to view their jobs for the day on the system.																	
Typical Course Of Events:	Actor Action	System Response																
		Manual Action	Automated Action															
	Step 1: The Mechanic from the “Mechanic Calendar” screen selects the day they want to view from the “mechanicCalendar” and clicks on the day.		Step 2: The system Displays the “JobLists” screen and populates it with the following details:															
			<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>dateOfJob</td><td>Heading</td><td>Used to identify which day the jobs are for</td></tr><tr><td>Check-In Bicycle</td><td>Button</td><td>Used to load the “Check-In Scan” screen.</td></tr><tr><td>jobsTable</td><td>Table</td><td>Displays a list of Jobs for the day.</td></tr><tr><td>viewJob</td><td>Icon Link</td><td>Used to navigate to the specific jobs view.</td></tr><tr><td>expandJob</td><td>Icon Link</td><td>Used to view extra information on a job.</td></tr></table>	Text/Label	Control Type	Additional Info	dateOfJob	Heading	Used to identify which day the jobs are for	Check-In Bicycle	Button	Used to load the “Check-In Scan” screen.	jobsTable	Table	Displays a list of Jobs for the day.	viewJob	Icon Link	Used to navigate to the specific jobs view.
Text/Label	Control Type	Additional Info																
dateOfJob	Heading	Used to identify which day the jobs are for																
Check-In Bicycle	Button	Used to load the “Check-In Scan” screen.																
jobsTable	Table	Displays a list of Jobs for the day.																
viewJob	Icon Link	Used to navigate to the specific jobs view.																
expandJob	Icon Link	Used to view extra information on a job.																
			The table is populated with data from the following by using a FIRESQL READ query:															

			Entity	Attributes	Input Name
			RepairJob	RepairJobStatusID	
			RepairJob	RepairJobID	Repair Job ID
			RepairJobStatus	RepairJobStatusID	
			RepairJobStatus	RepairJobStatus	Repair Job Status
			RepairJob	DateReported	Date Reported
			RepairJob	DateScheduled	Date Scheduled
			RepairJob	Assigned	Assigned
			MaintenanceJob	MaintenanceJobID	ID
			MaintenanceJob	BicycleID	Bicycle ID
			MaintenanceJob	MaintenanceJobStatusID	Job Status
			MaintenanceJob	DateScheduled	Date Scheduled
			MaintenanceJob	Assigned	Assigned
			MaintenanceJobStatus	MaintenanceJobStatusID	
			MaintenanceJobStatus	MaintenanceJobStatus	Job Status

			Step 3: The system creates and saves the following in the database by using a FIRESQL INSERT query: AuditTrail Table AuditID DateTime TransactionDescription UserID
Alternate Courses:	None		
Conclusion:	The use case narrative concludes when the relevant jobs have been displayed.		
Post-Condition:	- The Jobs have been viewed successfully. - The AuditTrail table is successfully updated.		
Business Rules	Only an authorized user can view the jobs on the system.		
Implementation Constraints and Specifications:	The mechanics literacy level is a constraint as we need to design the system in such a way that a person with a low literacy level can use the system.		
Assumptions:	The mechanic will be trained to use the system.		
Open Issues:	None		

2.3.3.2. View Job List

System Name:	VVs
Author:	Byron Donald
Date:	22/06/2020
Version:	1.0

Use Case Name:	View Job List	Use Case Type:
Use Case ID:	2.2	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	Mechanic	
Primary System Actor	None	
Other Participating Actors:	None	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process of a Mechanic viewing the list of actions/activities required for a job on the system. The Use Case starts when the Mechanic navigates from the "JobList" screen by selecting the job that they wish to view. The system then loads the "FaultList" screen and populates the table with the information of all the actions/activities needed for the selected job.	
Pre-Condition:	The Mechanic must be logged into the system.	

Trigger:	The Mechanic requests to view the Job List for a specific job on the system.																	
Typical Course Of Events:	Actor Action	System Response																
		Manual Action	Automated Action															
	Step 1: The Mechanic from the “JobList” screen selects the job they want to view from the “jobsTable” table and clicks on the job.		Step 2: The system Displays the “FaultList” screen for a RepairJob and populates it with the following details: [alt]															
			<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Complete Job</td><td>Button</td><td>Used to complete the job as a whole.</td></tr><tr><td>jobNumber</td><td>Heading</td><td>Used to identify which job the list is for</td></tr><tr><td>Faults</td><td>Table</td><td>Displays a list of Tasks for the Job.</td></tr><tr><td>“Complete”</td><td>Icon Link</td><td>Used to complete a specific fault</td></tr></table>	Text/Label	Control Type	Additional Info	Complete Job	Button	Used to complete the job as a whole.	jobNumber	Heading	Used to identify which job the list is for	Faults	Table	Displays a list of Tasks for the Job.	“Complete”	Icon Link	Used to complete a specific fault
			Text/Label	Control Type	Additional Info													
Complete Job			Button	Used to complete the job as a whole.														
jobNumber	Heading	Used to identify which job the list is for																
Faults	Table	Displays a list of Tasks for the Job.																
“Complete”	Icon Link	Used to complete a specific fault																
The table is populated with data from the following by using a FIRESQL READ query:																		
<table><tr><th>Entity</th><th>Attributes</th><th>Input Name</th></tr><tr><td>FaultType</td><td>FaultType</td><td>Fault</td></tr><tr><td>JobFault</td><td>Notes</td><td>Fault</td></tr></table>	Entity	Attributes	Input Name	FaultType	FaultType	Fault	JobFault	Notes	Fault									
Entity	Attributes	Input Name																
FaultType	FaultType	Fault																
JobFault	Notes	Fault																

			JobFault	JobFaultStatusID	Status															
			Section	SectionName	Section															
			Part	PartName	Part															
			JobFaultStatus	JobFaultStatus	Complete															
			Step 3: The system creates and saves the following in the database by using a FIRESQL INSERT query: AuditTrail Table AuditID DateTime TransactionDescription UserID																	
Alternate Courses:	Alt Step 2: The system Displays the “MaintenanceList” screen for a MaintenanceJob and populates it with the following details:																			
	<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Complete Job</td><td>Button</td><td>Used to complete the job as a whole.</td></tr><tr><td>jobNumber</td><td>Heading</td><td>Used to identify which job the list is for</td></tr><tr><td>maintenanceTasks</td><td>Table</td><td>Displays a list of Tasks for the Job.</td></tr><tr><td>“Complete”</td><td>Icon Link</td><td>Used to complete a specific fault</td></tr></table>					Text/Label	Control Type	Additional Info	Complete Job	Button	Used to complete the job as a whole.	jobNumber	Heading	Used to identify which job the list is for	maintenanceTasks	Table	Displays a list of Tasks for the Job.	“Complete”	Icon Link	Used to complete a specific fault
	Text/Label	Control Type	Additional Info																	
	Complete Job	Button	Used to complete the job as a whole.																	
	jobNumber	Heading	Used to identify which job the list is for																	
	maintenanceTasks	Table	Displays a list of Tasks for the Job.																	
“Complete”	Icon Link	Used to complete a specific fault																		
The table is populated with data from the following by using a FIRESQL READ query:																				

	Entity	Attributes	Input Name
	MaintenanceTask	MaintenanceTask	Maintenance Task
	Maintenancelist	MaintenancelistStatusID	Status
	Part	PartName	Part
Conclusion:	The use case narrative concludes when the relevant actions/activities have been displayed.		
Post-Condition:	- The Job List has been viewed successfully. - The AuditTrail table is successfully updated.		
Business Rules	Only an authorized user can view the Job List on the system.		
Implementation Constraints and Specifications:	The mechanics literacy level is a constraint as we need to design the system in such a way that a person with a low literacy level can use the system.		
Assumptions:	The mechanic will be trained to use the system.		
Open Issues:	None		

2.3.4. Sub-system 3: Tracking Hardware

2.3.4.1. Add Antenna

System Name:	VVs
Author:	Cameron King
Date:	15/06/2020
Version:	1.0

Use Case Name:	Add Antenna	Use Case Type:
Use Case ID:	3.1	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	Administrator	
Primary System Actor	None	
Other Participating Actors:	None	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process of an Administrator adding a new Antenna to the system. The Use Case starts when the Administrator navigates by selecting the "Antenna Management" option under the "System Management" Combo box in the Navigation bar. From there, they click the "Add" button and the system will load the "Antenna" screen. The system populates the relevant Combo boxes with information necessary for selection, and the corresponding labels, text boxes and buttons, icons, logos, and banners.	

Pre-Condition:	The Administrator must be logged into the system.				
Trigger:	The Administrator requests to add a new Antenna to the system.				
Typical Course Of Events:	Actor Action	System Response			
		Manual Action	Automated Action		
	Step 1: The Administrator selects the “Antenna Management” option and under the “System Management” Combo box in the Navigation bar.		Step 2: The system Displays the “Antenna Management” screen and populates it with the following details:		
			Text/Label	Control Type	Additional Info
			Navigation	Navigation-bar	Used to Navigate the side site.
			Add	Button	Used to load the “Antenna” screen.
			Search	Text-Box	Used for capturing Search criteria for the specific Antenna.
			Search	Button	Used to initiate a search for Antennas.
			Antenna Management	Table	Displays a list of Antennas.
“Pen icon”			Button	Used to navigate to the specific Antenna view.	

			The table is populated by using a FIRESQL READ query with data from the following: <table><tr><th>Entity</th><th>Attributes</th><th>Input Name</th></tr><tr><td>Antenna</td><td>AntennaSerialNumber</td><td>Antenna Serial Number</td></tr><tr><td>Antenna</td><td>Co-ordinates</td><td>Co-ordinates</td></tr><tr><td>AntennaStatus</td><td>AntennaStatus</td><td>Status</td></tr></table> The FIRESQL READ query will display the AntennaStatus in the AntennaStatus table that corresponds with the AntennaStatusID in the Antenna and AntennaStatus tables.	Entity	Attributes	Input Name	Antenna	AntennaSerialNumber	Antenna Serial Number	Antenna	Co-ordinates	Co-ordinates	AntennaStatus	AntennaStatus	Status			
	Entity	Attributes	Input Name															
	Antenna	AntennaSerialNumber	Antenna Serial Number															
	Antenna	Co-ordinates	Co-ordinates															
	AntennaStatus	AntennaStatus	Status															
	Step 3: The Administrator clicks on the “Add” button.		Step 4: The system will load the “Antenna” screen and displays it with the following controls and layout: <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Navigation</td><td>Navigation-bar</td><td>Used to Navigate the side site.</td></tr><tr><td>Antenna</td><td>Label</td><td>Heading for the pop-up displayed.</td></tr><tr><td>Antenna Serial Number</td><td>Text box</td><td>Used to indicate what the corresponding text box is used for.</td></tr><tr><td>Co-ordinates</td><td>Text box</td><td>Used to indicate what the</td></tr></table>	Text/Label	Control Type	Additional Info	Navigation	Navigation-bar	Used to Navigate the side site.	Antenna	Label	Heading for the pop-up displayed.	Antenna Serial Number	Text box	Used to indicate what the corresponding text box is used for.	Co-ordinates	Text box	Used to indicate what the
	Text/Label	Control Type	Additional Info															
	Navigation	Navigation-bar	Used to Navigate the side site.															
	Antenna	Label	Heading for the pop-up displayed.															
	Antenna Serial Number	Text box	Used to indicate what the corresponding text box is used for.															
Co-ordinates	Text box	Used to indicate what the																

				corresponding text box is used for.	
			Date Installed	Date	Used to indicate what the corresponding text box is used for.
			Antenna Suburb	Combo box	Used to indicate what the corresponding Combo box is used for.
			Antenna Status	Combo box	Used to indicate what the corresponding Combo box is used for.
			Create	Button	Used to Save the new Antenna Details.
			Cancel	Button	Used to cancel the action of adding an Antenna.
			The Antenna Status Combo box options are populated by using a FIRESQL READ query with data from the following:		
			Entity	Attributes	

			AntennaStatus AntennaStatus The Antenna Suburb Combo box options are populated by using a FIRESQL READ query with data from the following: <table><tr><th>Entity</th><th>Attributes</th></tr><tr><td>Suburb</td><td>SuburbName</td></tr></table>	Entity	Attributes	Suburb	SuburbName
	Entity	Attributes					
	Suburb	SuburbName					
	Step 5: Administrator inputs the following Antenna details: • Antenna Serial Number • Co-ordinates • Date Installed • Suburb • Antenna Status						
Step 6: The Administrator clicks the "Create" button. [alt]		Step 7: The system captures and ensures that all of the fields are filled in and an option is selected in the Combo box. [alt]					
		Step 8: The system validates that the captured details are in the correct format: [alt] <i>Antenna table</i> <table><tr><th>Attribute</th><th>Validation</th></tr><tr><td>Co-Ordinates</td><td>17. Max 50 characters 18. The latitude component must be between -90 and 90. It may possess any number of decimalized digits, or</td></tr></table>	Attribute	Validation	Co-Ordinates	17. Max 50 characters 18. The latitude component must be between -90 and 90. It may possess any number of decimalized digits, or	
Attribute	Validation						
Co-Ordinates	17. Max 50 characters 18. The latitude component must be between -90 and 90. It may possess any number of decimalized digits, or						

			it may be a whole number. 19.The longitude component must be between -180 and 180. It may possess any number of decimalized digits, or it may be a whole number. DateInstalled - Max 10 characters - Format must follow: DD/MM/YYYY - Numeric characters and "/" or "-" only. AntennaStatus - Combo box option is selected Suburb - Combo box option is selected						
			Step 9: The system verifies that no duplicate Antenna exists on the system by comparing the following: [alt] <table><tr><th>Input</th><th>Attribute</th><th>Entity</th></tr><tr><td>Co-ordinates</td><td>Co-Ordinates</td><td>Antenna</td></tr></table>	Input	Attribute	Entity	Co-ordinates	Co-Ordinates	Antenna
	Input	Attribute	Entity						
Co-ordinates	Co-Ordinates	Antenna							
		Step 10: The system prompts the Administrator to confirm to add the new Antenna with the following: <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Confirmation</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed.</td></tr></table>	Text/Label	Control Type	Additional Info	Confirmation	Dialog Box Title	Used to indicate what type of message is being displayed.	
Text/Label	Control Type	Additional Info							
Confirmation	Dialog Box Title	Used to indicate what type of message is being displayed.							

			"Are you sure you want to add this Antenna?" Dialog Box Message Used to inform the Administrator of the action.
			Yes Button Used to confirm the action.
			No Button Used to decline the action.
	Step 11: The Administrator clicks the "Yes" button. [alt]		Step 12: The Antenna is assigned a unique AntennaID in the Antenna table which is set to the next consecutive number after the previous AntennaID in the database, the system then creates and saves the new Antenna in the Antenna table. The following attributes are saved in the database by using a FIRESQL INSERT query: Antenna Table AntennaID Co-ordinates AntennaStatusID DateInstalled SuburbID
			Step 13: The system notifies the Administrator that the Antenna has been added successfully to the database with the following: [alt] Text/Label Control Type Additional Info Success Dialog Box Title Used to indicate what type of

			message is being displayed.								
			<table><tr><td>"Antenna has been added successfully"</td><td>Dialog Box Message</td><td>Used to inform the Administrator of the result.</td></tr><tr><td>Ok</td><td>Button</td><td>Used to accept the result.</td></tr></table>	"Antenna has been added successfully"	Dialog Box Message	Used to inform the Administrator of the result.	Ok	Button	Used to accept the result.		
	"Antenna has been added successfully"	Dialog Box Message	Used to inform the Administrator of the result.								
Ok	Button	Used to accept the result.									
		Step 14: The system creates and saves the following in the database by using a FIRESQL INSERT query: <table><tr><td>AuditTrail Table</td></tr><tr><td>AuditTrailID</td></tr><tr><td>DateTime</td></tr><tr><td>TransactionDescription</td></tr><tr><td>UserID</td></tr></table>	AuditTrail Table	AuditTrailID	DateTime	TransactionDescription	UserID				
AuditTrail Table											
AuditTrailID											
DateTime											
TransactionDescription											
UserID											
Alternate Courses:	Alt Step 6: The Administrator clicks the "Cancel" button. Terminate Use Case.										
	Alt Step 7a: In the case of an empty Text box, the system notifies the Administrator of the Input error below the input field with the following:										
	<table><tr><td>Text/Label</td><td>Control Type</td><td>Additional Info</td></tr><tr><td>"Please fill this field."</td><td>Label (with a light red background color fill)</td><td>Used to inform the Administrator of the result.</td></tr><tr><td>"X"</td><td>Icon</td><td>Used to indicate an error result.</td></tr></table> Redirect to Step 5.			Text/Label	Control Type	Additional Info	"Please fill this field."	Label (with a light red background color fill)	Used to inform the Administrator of the result.	"X"	Icon
Text/Label	Control Type	Additional Info									
"Please fill this field."	Label (with a light red background color fill)	Used to inform the Administrator of the result.									
"X"	Icon	Used to indicate an error result.									

Alt Step 7b: In the case of a Combo box option not selected, the system notifies the Administrator of the Input error below the input field with the following:

Text/Label	Control Type	Additional Info
"Please select an option."	Label (with a light red background color fill)	Used to inform the Administrator of the result.
"X"	Icon	Used to indicate an error result.

Redirect to Step 5.

Alt Step 8: In the case of invalid input being entered, the system notifies the Administrator of the Format error below the input field with the following:

Text/Label	Control Type	Additional Info
"Invalid input."	Label (with a light red background color fill)	Used to inform the Administrator of the result.
"X"	Icon	Used to indicate an error result.

Redirect to Step 5.

Alt Step 9: The inputted Antenna already exists in the system; the system will notify the Administrator of the duplication error with the following:

Text/Label	Control Type	Additional Info
Error	Dialog Box Title	Used to indicate what type of message is being displayed.

	"The Antenna already exists. Please try again."	Dialog Box Message	Used to inform the Administrator of the result.
	Ok	Button	Used to accept the result.
	Redirect to Step 5.		
	Alt Step 11: The Administrator declines to add the new Antenna by clicking the "No" button. Redirect to Step 5.		
	Alt Step 13: The Antenna was not added successfully to the database, the system will notify the Administrator of the save error with the following:		
	Text/Label	Control Type	Additional Info
	Failed	Dialog Box Title	Used to indicate what type of message is being displayed
	"Create unsuccessful. Please try again."	Dialog Box Message	Used to inform the Administrator of the result.
	Ok	Button	Used to accept the result.
	Redirect to Step 5.		
Conclusion:	The use case concludes when the Administrator has been notified that the Antenna has been added successfully.		
Post-Condition:	- A new Antenna is added to the system with the relevant details. - The AuditTrail table is successfully updated. - The Antenna table is successfully updated.		

Business Rules	• Only an Administrator can add an Antenna to the system.
Implementation Constraints and Specifications:	• None
Assumptions:	• None
Open Issues:	• None

2.3.4.2. Search Antenna

System Name:	VVs
Author:	Cameron King
Date:	15/06/2020
Version:	1.0

Use Case Name:	Search Antenna	Use Case Type:
Use Case ID:	3.2	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	Administrator	
Primary System Actor	None	
Other Participating Actors:	None	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process of an Administrator searching an Antenna on the system. The Use Case starts when the Administrator navigates by selecting the "Antenna Management" option under the "System Management" Combo box in the Navigation bar. The system populates the table with the information of all the antennas.	
Pre-Condition:	The Administrator must be logged into the system.	

Trigger:	The Administrator requests to search for an Antenna on the system.				
Typical Course Of Events:	Actor Action	System Response			
		Manual Action	Automated Action		
	Step 1: The Administrator selects the “Antenna Management” option under the “System Management” Combo box in the Navigation bar.		Step 2: The system Displays the “Antenna Management” screen and populates it with the following details:		
			Text/Label	Control Type	Additional Info
			Navigation	Nav-bar	Used to Navigate the side site.
			Add	Button	Used to load the “Antenna” screen.
			Search	Text-Box	Used for capturing Search criteria for the specific Antenna.
			Search	Button	Used to initiate a search for Antennas.
			Antenna Management	Table	Displays a list of Antennas.
			“Pen icon”	Button	Used to navigate to the specific Antenna view.
“Back”			Button	Returns to Home Page.	

		The table is populated with data from the following by using a FIRESQL READ query: <table><tr><th>Entity</th><th>Attributes</th><th>Input Name</th></tr><tr><td>Antenna</td><td>AntennaSerialNu mber</td><td>Antenna Serial Number</td></tr><tr><td>Antenna</td><td>Co-ordinates</td><td>Co-ordinates</td></tr><tr><td>AntennaStatus</td><td>AntennaStatus</td><td>Status</td></tr></table> The FIRESQL READ query will display the AntennaStatus in the AntennaStatus table that corresponds with the AntennaStatusID in the Antenna and AntennaStatus tables.	Entity	Attributes	Input Name	Antenna	AntennaSerialNu mber	Antenna Serial Number	Antenna	Co-ordinates	Co-ordinates	AntennaStatus	AntennaStatus	Status
Entity	Attributes	Input Name												
Antenna	AntennaSerialNu mber	Antenna Serial Number												
Antenna	Co-ordinates	Co-ordinates												
AntennaStatus	AntennaStatus	Status												
Step 3: The Administrator enters the search criteria that they wish to search by. They do so by typing in a keyword in the "Search" Text box.														
Step 5: The Administrator clicks the search button.		Step 6: The system uses a FIRESQL READ query to READ the following details in the database: [alt] <table><tr><th>Entity</th><th>Attributes</th></tr><tr><td>Antenna</td><td>AntennaSerialNumber</td></tr></table>	Entity	Attributes	Antenna	AntennaSerialNumber								
Entity	Attributes													
Antenna	AntennaSerialNumber													

			Step 7: The following details are loaded and used to populate the “Antenna Management” table by using a FIRESQL READ query: <table><tr><th>Output</th><th>Attribute</th><th>Entity</th></tr><tr><td>Antenna Serial Number</td><td>AntennaSerialNumber</td><td>Antenna</td></tr><tr><td>Co-ordinates</td><td>Co-ordinates</td><td>Antenna</td></tr><tr><td> </td><td> </td><td> </td></tr><tr><td>Antenna Status</td><td>AntennaStatus</td><td>AntennaStatus</td></tr></table> The FIRESQL READ query will display the AntennaStatus in the AntennaStatus table that corresponds with the AntennaStatusID in the Antenna and AntennaStatus tables.	Output	Attribute	Entity	Antenna Serial Number	AntennaSerialNumber	Antenna	Co-ordinates	Co-ordinates	Antenna				Antenna Status	AntennaStatus	AntennaStatus
	Output	Attribute	Entity															
	Antenna Serial Number	AntennaSerialNumber	Antenna															
	Co-ordinates	Co-ordinates	Antenna															
Antenna Status	AntennaStatus	AntennaStatus																
Step 8: The Administrator clicks on the “Pen icon” Antenna Button in the “Action” table column.		Step 9: The System will load the “Antenna” screen and displays it with the following controls and layout: <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Navigation</td><td>Nav-bar</td><td>Used to Navigate the side site.</td></tr><tr><td>Antenna</td><td>Label</td><td>Heading for the pop-up displayed.</td></tr><tr><td>Antenna Serial Number</td><td>Textbox</td><td>Used to indicate what the corresponding read-only text box is used for.</td></tr></table>	Text/Label	Control Type	Additional Info	Navigation	Nav-bar	Used to Navigate the side site.	Antenna	Label	Heading for the pop-up displayed.	Antenna Serial Number	Textbox	Used to indicate what the corresponding read-only text box is used for.				
Text/Label	Control Type	Additional Info																
Navigation	Nav-bar	Used to Navigate the side site.																
Antenna	Label	Heading for the pop-up displayed.																
Antenna Serial Number	Textbox	Used to indicate what the corresponding read-only text box is used for.																

			Antenna Suburb	Combo box	Used to indicate what the corresponding combo box is used for.
			Antenna Status	Combo box	Used to indicate what the corresponding combo box is used for.
			Co-ordinates	Textbox	Used to indicate what the corresponding text box is used for.
			Date Installed	Date picker	Used to indicate what the corresponding date picker is used for.
			Delete	Button	Used to Delete the Antenna details.
			Update	Button	Used to update the Antenna details.
			The Antenna Status Combo box options are populated with data from the following by using a FIRESQL READ query:		

Entity	Attributes
AntennaStatus	AntennaStatus

The FIRESQL READ query will display the AntennaStatus in the AntennaStatus table that corresponds with the AntennaStatusID in the Antenna and AntennaStatus tables.

The Antenna Suburb Combo box options are populated by using a FIRESQL READ query with data from the following:

Entity	Attributes
Suburb	SuburbName

The FIRESQL READ query will display the SuburbName in the Suburb table that corresponds with the SuburbID in the Antenna and Suburb tables.

The following fields are populated with data retrieved from the database where the AntennaID is the same as the one searched by using a FIRESQL READ query:

Output	Attribute	Entity
Antenna Serial Number	AntennaSerialNumber	Antenna
Antenna Status	AntennaStatus	AntennaStatus
Suburb	SuburbID	Antenna
Co-ordinates	Co-ordinates	Antenna
Date Installed	DateInstalled	Antenna

			Step 10: The system creates and saves the following in the database by using a FIRESQL INSERT query: AuditTrail Table AuditTrailID DateTime TransactionDescription UserID									
Alternate Courses:	Alt Step 6: The specific Antenna does not exist in the database. The system will notify the Administrator of this with the following: <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>"Antenna not found."</td><td>Label</td><td>Used to inform the Administrator of the result.</td></tr><tr><td>"X"</td><td>Banner (with a light red background color fill)</td><td>Used to display the result message.</td></tr></table> Redirect to Step 3.			Text/Label	Control Type	Additional Info	"Antenna not found."	Label	Used to inform the Administrator of the result.	"X"	Banner (with a light red background color fill)	Used to display the result message.
Text/Label	Control Type	Additional Info										
"Antenna not found."	Label	Used to inform the Administrator of the result.										
"X"	Banner (with a light red background color fill)	Used to display the result message.										
Conclusion:	The use case narrative concludes when the searched Antenna details are displayed.											
Post-Condition:	• The Antenna has been searched successfully. • The AuditTrail table is successfully updated.											
Business Rules	• Only an Administrator can search for an Antenna on the system.											

Implementation Constraints and Specifications:	• None
Assumptions:	• None
Open Issues:	• None

2.3.4.3. Update Antenna

System Name:	VVs
Author:	Cameron King
Date:	15/06/2020
Version:	1.0

Use Case Name:	Update Antenna	Use Case Type:
Use Case ID:	6.3	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	Administrator	
Primary System Actor	None	
Other Participating Actors:	None	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process of an Administrator updating an existing Antenna's details in the system. The Use Case starts when the Administrator navigates by selecting the "Antenna Management" option under the "System Management" Combo box in the Navigation bar. From there, they search for the Antenna they want to update and the system will load the "Antenna" screen with the Antenna details. The system populates the relevant Combo boxes with information of the searched Antenna, and the Administrator can make their changes and save.	
Pre-Condition:	The Administrator must be logged into the system.	

Trigger:	The Administrator requests to update an existing Antenna in the system.									
Typical Course Of Events:	Actor Action	System Response								
		Manual Action	Automated Action							
	Step 1: The Administrator selects the “Antenna Management” option under the “System Management” Combo box in the Navigation bar.		Step 2: The System invokes Use Case 3.2 “Search Antenna” and displays the “Antenna” screen with the data of the searched Antenna.							
	Step 3: Administrator updates the following Antenna details: • Antenna Status • Co-ordinates • Suburb • Date Installed									
	Step 4: The Administrator clicks the “Update” button. [alt]		Step 5: The system captures and ensures that all of the fields are filled in and an option is selected in the Combo box. [alt]							
			Step 6: The system validates that the captured details are in the correct format: [alt] Antenna table <table><tr><th>Attribute</th><th>Validation</th></tr><tr><td>AntennaStatus</td><td>20.Combo box option is selected</td></tr><tr><td>SuburbID</td><td>21.Combo box option is selected</td></tr><tr><td>Co-ordinates</td><td> • Max 50 characters • The latitude component must be </td></tr></table>	Attribute	Validation	AntennaStatus	20.Combo box option is selected	SuburbID	21.Combo box option is selected	Co-ordinates
Attribute	Validation									
AntennaStatus	20.Combo box option is selected									
SuburbID	21.Combo box option is selected									
Co-ordinates	• Max 50 characters • The latitude component must be									

			between -90 and 90. It may possess any number of decimalized digits, or it may be a whole number. - The longitude component must be between -180 and 180. It may possess any number of DateInstalled - Format must follow: DD/MM/YYYY - Numeric characters and "/" or "-" only.														
			Step 7: The system prompts the Administrator to confirm the update of the Antenna's details. <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Confirmation</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed.</td></tr><tr><td>"Are you sure you want to update this Antenna?"</td><td>Dialog Box Message</td><td>Used to inform the Administrator of the action.</td></tr><tr><td>Yes</td><td>Button</td><td>Used to confirm the action.</td></tr><tr><td>No</td><td>Button</td><td>Used to decline the action.</td></tr></table>	Text/Label	Control Type	Additional Info	Confirmation	Dialog Box Title	Used to indicate what type of message is being displayed.	"Are you sure you want to update this Antenna?"	Dialog Box Message	Used to inform the Administrator of the action.	Yes	Button	Used to confirm the action.	No	Button
Text/Label	Control Type	Additional Info															
Confirmation	Dialog Box Title	Used to indicate what type of message is being displayed.															
"Are you sure you want to update this Antenna?"	Dialog Box Message	Used to inform the Administrator of the action.															
Yes	Button	Used to confirm the action.															
No	Button	Used to decline the action.															

	Step 8: The Administrator clicks the “Yes” button. [alt]		Step 9: The system then updates and saves the updated Antenna details in the Antenna table. The following attributes are updated in the database by using a FIRESQL UPDATE query: Antenna Table AntennaStatusID SuburbID Co-ordinates DateInstalled												
			Step 10: The system notifies the Administrator that the Antenna has been updated successfully in the database with the following: [alt] <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Success</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed.</td></tr><tr><td>“The Antenna has been updated successfully”</td><td>Dialog Box Message</td><td>Used to inform the Administrator of the result.</td></tr><tr><td>Ok</td><td>Button</td><td>Used to accept the result.</td></tr></table>	Text/Label	Control Type	Additional Info	Success	Dialog Box Title	Used to indicate what type of message is being displayed.	“The Antenna has been updated successfully”	Dialog Box Message	Used to inform the Administrator of the result.	Ok	Button	Used to accept the result.
	Text/Label	Control Type	Additional Info												
Success	Dialog Box Title	Used to indicate what type of message is being displayed.													
“The Antenna has been updated successfully”	Dialog Box Message	Used to inform the Administrator of the result.													
Ok	Button	Used to accept the result.													
		Step 11: The system updates and saves the following in the database by using a FIRESQL INSERT query: AuditTrail Table													

			AuditTrailID
			DateTime
			TransactionDescription
			UserID
Alternate Courses:	Alt Step 4: The Administrator clicks the "Cancel" button. Terminate Use Case.		
	Alt Step 5a: In the case of an empty Text box, the system notifies the Administrator of the Input error below the input field with the following:		
	Text/Label	Control Type	Additional Info
	"Please fill this field."	Label (with a light red background color fill)	Used to inform the Administrator of the result.
	"X"	Icon	Used to indicate an error result.
	Redirect to Step 3.		
Alternate Courses:	Alt Step 5b: In the case of a Combo box option not selected, the system notifies the Administrator of the Input error below the input field with the following:		
	Text/Label	Control Type	Additional Info
	"Please select an option."	Label (with a light red background color fill)	Used to inform the Administrator of the result.
	"X"	Icon	Used to indicate an error result.
		Redirect to Step 3.	

Alt Step 6: In the case of invalid input being entered, the system notifies the Administrator of the Format error below the input field with the following:

Text/Label	Control Type	Additional Info
"Invalid input."	Label (with a light red background color fill)	Used to inform the Administrator of the result.
"X"	Icon	Used to indicate an error result.

Redirect to Step 3.

Alt Step 8: The Administrator declines to update the Antenna by clicking the "No" button. Redirect to Step 4.

Alt Step 10: The Antenna was not updated successfully in the database, the system will notify the Administrator of the update error with the following:

Text/Label	Control Type	Additional Info
Failed	Dialog Box Title	Used to indicate what type of message is being displayed
"Update unsuccessful. Please try again."	Dialog Box Message	Used to inform the Administrator of the result.
"Help icon"	Button	Used to Load the Help document.
Ok	Button	Used to accept the result.

Redirect to Step 3.

Conclusion:	The use case concludes when the Administrator has been notified that the Antenna has been updated successfully.
Post-Condition:	• An existing Antenna is updated in the system with the relevant details. • The AuditTrail table is successfully updated. • The Antenna table is successfully updated.
Business Rules	• Only an Administrator can update an Antenna to the system.
Implementation Constraints and Specifications:	• None
Assumptions:	• None
Open Issues:	• None

2.3.4.4. Delete Antenna

System Name:	VVs
Author:	Cameron King
Date:	15/06/2020
Version:	1.0

Use Case Name:	Delete Antenna	Use Case Type:
Use Case ID:	3.4	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	Administrator	
Primary System Actor	None	
Other Participating Actors:	None	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process of an Administrator deleting an existing Antenna's details from the system. The Use Case starts when the Administrator navigates by selecting the "Antenna Management" option under the "System Management" Combo box in the Navigation bar. From there, they search for the Antenna they want to delete and the system displays the "Antenna" screen with the Antenna details to be deleted. The system populates the relevant Combo boxes with information of the searched Antenna, and the Administrator can make their deletion.	
Pre-Condition:	The Administrator must be logged into the system.	

Trigger:	The Administrator requests to delete an existing Antenna from the system.			
Typical Course Of Events:	Actor Action	System Response		
		Manual Action	Automated Action	
	Step 1: The Administrator selects the "Antenna Management" option under the "System Management" Combo box in the Navigation bar.		Step 2: The System invokes Use Case 3.2 "Search Antenna" and displays the "Antenna" screen with the data of the searched Antenna.	
	Step 3: The Administrator clicks on the "Delete" button.[alt]		Step 4: The system prompts the Administrator to confirm the deletion of the Antenna's details with the following:	
			Text/Label	Control Type
Confirmation			Dialog Box Title	Used to indicate what type of message is being displayed.
"Are you sure you want to delete this Antenna?"			Dialog Box Message	Used to inform the Administrator of the action.
		Yes	Button	Used to confirm the action.
		No	Button	Used to decline the action.

	Step 5: The Administrator clicks the “Yes” button. [alt]		Step 6: The system then removes the Antenna details in the Antenna table. The following attributes are deleted in the database by using a FIRESQL DELETE query: Antenna Table AntennaID AntennaStatusID SuburbID Co-ordinates DateInstalled												
			Step 7: The system notifies the Administrator that the Antenna has been deleted successfully from the database with the following: [alt] <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Success</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed.</td></tr><tr><td>“The Antenna has been deleted successfully”</td><td>Dialog Box Message</td><td>Used to inform the Administrator of the result.</td></tr><tr><td>Ok</td><td>Button</td><td>Used to accept the result.</td></tr></table>	Text/Label	Control Type	Additional Info	Success	Dialog Box Title	Used to indicate what type of message is being displayed.	“The Antenna has been deleted successfully”	Dialog Box Message	Used to inform the Administrator of the result.	Ok	Button	Used to accept the result.
	Text/Label	Control Type	Additional Info												
Success	Dialog Box Title	Used to indicate what type of message is being displayed.													
“The Antenna has been deleted successfully”	Dialog Box Message	Used to inform the Administrator of the result.													
Ok	Button	Used to accept the result.													
		Step 8: The system updates and saves the following in the database by using a FIRESQL INSERT query:													

				AuditTrail Table AuditTrailID DateTime TransactionDescription UserID														
Alternate Courses:	Alt Step 3: The Administrator clicks the "Cancel" button. Terminate Use Case.																	
	Alt Step 5: The Administrator declines to delete the Antenna by clicking the "No" button. Redirect to Step 2.																	
	Alt Step 7: The Administrator is notified of the unsuccessful delete action with the following:																	
	<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Failed</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed</td></tr><tr><td>"Delete unsuccessful. Please try again."</td><td>Dialog Box Message</td><td>Used to inform the Administrator of the result.</td></tr><tr><td>"Help icon"</td><td>Button</td><td>Used to Load the Help document.</td></tr><tr><td>Ok</td><td>Button</td><td>Used to accept the result.</td></tr></table>			Text/Label	Control Type	Additional Info	Failed	Dialog Box Title	Used to indicate what type of message is being displayed	"Delete unsuccessful. Please try again."	Dialog Box Message	Used to inform the Administrator of the result.	"Help icon"	Button	Used to Load the Help document.	Ok	Button	Used to accept the result.
	Text/Label	Control Type	Additional Info															
	Failed	Dialog Box Title	Used to indicate what type of message is being displayed															
	"Delete unsuccessful. Please try again."	Dialog Box Message	Used to inform the Administrator of the result.															
"Help icon"	Button	Used to Load the Help document.																
Ok	Button	Used to accept the result.																
Redirect to Step 2.																		
Conclusion:	The use case concludes when the Administrator has been notified that the Antenna has been successfully deleted.																	
Post-Condition:	An existing Antenna is deleted from the system with all the relevant details being removed.																	

	• The AuditTrail table is successfully updated. • The Antenna table is successfully updated with the deleted details being removed.
Business Rules	• Only an Administrator can delete an Antenna from the system.
Implementation Constraints and Specifications:	• None
Assumptions:	• None
Open Issues:	• None

2.3.4.5. Add Tag

System Name:	VVs
Author:	Cameron King
Date:	15/06/2020
Version:	1.0

Use Case Name:	Add Tag	Use Case Type:
Use Case ID:	3.5	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	Administrator	
Primary System Actor	None	
Other Participating Actors:	None	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process of an Administrator adding a new Tag to the system. The Use Case starts when the Administrator navigates by selecting the "Tag Management" option under the "Bicycle Management" Combo box in the Navigation bar. From there, they click the "Add Tag" button and the system will load the "Tag" screen. The system populates the relevant Combo boxes with information necessary for selection, and the corresponding labels, text boxes and buttons, icons, logos, and banners.	
Pre-Condition:	The Administrator must be logged into the system.	

Trigger:	The Administrator requests to add a new Tag to the system.				
Typical Course Of Events:	Actor Action	System Response			
		Manual Action	Automated Action		
	Step 1: The Administrator selects the “Tag Management” option and under the “System Management” Combo box in the Navigation bar.		Step 2: The system Displays the “Tag Management” screen and populates it with the following details:		
			Text/Label	Control Type	Additional Info
			Add	Button	Used to load the “Tag” screen.
			Search	Button	Used to search for a specific Tag.
			Search	Text-Box	Used for capturing Search criteria for the specific Tag.
			Tag Management	Table	Displays a list of Tags.
			“Pen icon”	Button	Used to navigate to the specific Tag view.
			The table is populated by using a FIRESQL READ query with data from the following:		
			Entity	Attributes	Input Name
			Tag	TagSerialNumber	Tag Serial Number
			Bicycle	BicycleCode	Bicycle Code

			TagStatus TagStatus Tag Status The FIRESQL READ query will display the TagStatus in the TagStatus table that corresponds with the TagStatusID in the Tag and TagStatus tables.																					
	Step 3: The Administrator clicks on the "Add Tag" button.		Step 4: The system will load the "Tag" screen and displays it with the following controls and layout: <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Menu</td><td>Menu-icon</td><td>Used to Navigate the side menu-bar.</td></tr><tr><td>Tag</td><td>Label</td><td>Heading for the pop-up displayed.</td></tr><tr><td>Tag Serial Number</td><td>Label</td><td>Used to indicate what the corresponding text box is used for.</td></tr><tr><td>Tag Serial Number</td><td>Text box</td><td>Used to input the Tag details.</td></tr><tr><td>Create Tag</td><td>Button</td><td>Used to Save the new Tag Details.</td></tr><tr><td>Cancel</td><td>Button</td><td>Used to cancel the action of adding a Tag.</td></tr></table>	Text/Label	Control Type	Additional Info	Menu	Menu-icon	Used to Navigate the side menu-bar.	Tag	Label	Heading for the pop-up displayed.	Tag Serial Number	Label	Used to indicate what the corresponding text box is used for.	Tag Serial Number	Text box	Used to input the Tag details.	Create Tag	Button	Used to Save the new Tag Details.	Cancel	Button	Used to cancel the action of adding a Tag.
	Text/Label	Control Type	Additional Info																					
	Menu	Menu-icon	Used to Navigate the side menu-bar.																					
	Tag	Label	Heading for the pop-up displayed.																					
	Tag Serial Number	Label	Used to indicate what the corresponding text box is used for.																					
	Tag Serial Number	Text box	Used to input the Tag details.																					
	Create Tag	Button	Used to Save the new Tag Details.																					
Cancel	Button	Used to cancel the action of adding a Tag.																						
Step 5: Administrator inputs the following Tag details: • Tag Serial Number																								

	Step 6: The Administrator clicks the “Create Tag” button. [alt]		Step 7: The system captures and ensures that the field is filled. [alt]											
			Step 8: The system validates that the captured details are in the correct format: [alt] Tag table<table><tr><th>Attribute</th><th>Validation</th></tr><tr><td>TagSerialNumber</td><td>22.Max 50 characters</td></tr></table>	Attribute	Validation	TagSerialNumber	22.Max 50 characters							
	Attribute	Validation												
	TagSerialNumber	22.Max 50 characters												
		Step 9: The system verifies that no duplicate Tag exists on the system by comparing the following: [alt] <table><tr><th>Input</th><th>Attribute</th><th>Entity</th></tr><tr><td>Tag Serial Number</td><td>Tag Serial Number</td><td>Tag</td></tr></table>	Input	Attribute	Entity	Tag Serial Number	Tag Serial Number	Tag						
Input	Attribute	Entity												
Tag Serial Number	Tag Serial Number	Tag												
		Step 10: The system prompts the Administrator to confirm to add the new Tag with the following: <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Confirmation</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed.</td></tr><tr><td>“Are you sure you want to add this Tag?”</td><td>Dialog Box Message</td><td>Used to inform the Administrator of the action.</td></tr><tr><td>Yes</td><td>Button</td><td>Used to confirm the action.</td></tr></table>	Text/Label	Control Type	Additional Info	Confirmation	Dialog Box Title	Used to indicate what type of message is being displayed.	“Are you sure you want to add this Tag?”	Dialog Box Message	Used to inform the Administrator of the action.	Yes	Button	Used to confirm the action.
Text/Label	Control Type	Additional Info												
Confirmation	Dialog Box Title	Used to indicate what type of message is being displayed.												
“Are you sure you want to add this Tag?”	Dialog Box Message	Used to inform the Administrator of the action.												
Yes	Button	Used to confirm the action.												

			No Button Used to decline the action.												
	Step 11: The Administrator clicks the “Yes” button. [alt]		Step 12: The system then creates and saves the new Tag in the Tag table. The TagStatusID is saved with the same TagStatusID from the TagStatus where the TagStatus is ‘Available’. The following attributes are saved in the database by using a FIRESQL INSERT query: Tag Table TagID TagSerialNumber TagStatusID												
			Step 13: The system notifies the Administrator that the Tag has been added successfully to the database with the following: [alt] <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Success</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed.</td></tr><tr><td>“Tag has been added successfully”</td><td>Dialog Box Message</td><td>Used to inform the Administrator of the result.</td></tr><tr><td>Ok</td><td>Button</td><td>Used to accept the result.</td></tr></table>	Text/Label	Control Type	Additional Info	Success	Dialog Box Title	Used to indicate what type of message is being displayed.	“Tag has been added successfully”	Dialog Box Message	Used to inform the Administrator of the result.	Ok	Button	Used to accept the result.
	Text/Label	Control Type	Additional Info												
Success	Dialog Box Title	Used to indicate what type of message is being displayed.													
“Tag has been added successfully”	Dialog Box Message	Used to inform the Administrator of the result.													
Ok	Button	Used to accept the result.													
		Step 14: The system creates and saves the following in the database by using a FIRESQL INSERT query:													

				AuditTrail Table
				AuditTrailID
				DateTime
				TransactionDescription
				UserID
Alternate Courses:	Alt Step 6: The Administrator clicks the "Cancel" button. Terminate Use Case.			
	Alt Step 7a: In the case of an empty Text box, the system notifies the Administrator of the Input error below the input field with the following:			
	Text/Label	Control Type	Additional Info	
	"Please fill this field."	Label (with a light red background color fill)	Used to inform the Administrator of the result.	
	"X"	Icon	Used to indicate an error result.	
	Redirect to Step 5.			
	Alt Step 8: In the case of invalid input being entered, the system notifies the Administrator of the Format error below the input field with the following:			
	Text/Label	Control Type	Additional Info	
	"Invalid input."	Label (with a light red background color fill)	Used to inform the Administrator of the result.	
	"X"	Icon	Used to indicate an error result.	
		Redirect to Step 5.		

Alt Step 9: The inputted Tag already exists in the system; the system will notify the Administrator of the duplication error with the following:

Text/Label	Control Type	Additional Info
Error	Dialog Box Title	Used to indicate what type of message is being displayed.
"The Tag already exists. Please try again."	Dialog Box Message	Used to inform the Administrator of the result.
"Help icon"	Button	Used to Load the Help document.
Ok	Button	Used to accept the result.

Redirect to Step 5.

Alt Step 11: The Administrator declines to add the new Tag by clicking the "No" button. Redirect to Step 5.

Alt Step 13: The Tag was not added successfully to the database, the system will notify the Administrator of the save error with the following:

Text/Label	Control Type	Additional Info
Failed	Dialog Box Title	Used to indicate what type of message is being displayed
"Create unsuccessful. Please try again."	Dialog Box Message	Used to inform the Administrator of the result.

	"Help icon"	Button	Used to Load the Help document.
	Ok	Button	Used to accept the result.
	Redirect to Step 5.		
Conclusion:	The use case concludes when the Administrator has been notified that the Tag has been added successfully.		
Post-Condition:	• A new Tag is added to the system with the relevant details. • The AuditTrail table is successfully updated. • The Tag table is successfully updated.		
Business Rules	• Only an Administrator can add a Tag to the system.		
Implementation Constraints and Specifications:	• None		
Assumptions:	• None		
Open Issues:	• None		

2.3.4.6. Search Tag

System Name:	VVs
Author:	Cameron King
Date:	15/06/2020
Version:	1.0

Use Case Name:	Search Tag	Use Case Type:
Use Case ID:	3.6	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	Administrator	
Primary System Actor	None	
Other Participating Actors:	None	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process of an Administrator searching a Tag on the system. The Use Case starts when the Administrator navigates by selecting the "Tag Management" option under the "Bicycle Management" Combo box in the Navigation bar. The system populates the table with the information of all the Tags.	
Pre-Condition:	The Administrator must be logged into the system.	

Trigger:	The Administrator requests to search for a Tag on the system.																										
Typical Course Of Events:	Actor Action	System Response																									
		Manual Action	Automated Action																								
	Step 1: The Administrator selects the "Tag Management" option under the "Bicycle Management" Combo box in the Navigation bar.		Step 2: The system Displays the "Tag Management" screen and populates it with the following details:																								
			<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Add a new Tag</td><td>Button</td><td>Used to load the "Tag" screen.</td></tr><tr><td>Assign or Unassign Tag</td><td>Button</td><td>Used to load the "Assign or Unassign Tag" screen.</td></tr><tr><td>Search</td><td>Text-Box</td><td>Used for capturing Search criteria for the specific Tag.</td></tr><tr><td>Tag Management</td><td>Table</td><td>Displays a list of Tags.</td></tr><tr><td>"Pen icon"</td><td>Button</td><td>Used to navigate to the specific Tag view.</td></tr><tr><td>"Back"</td><td>Button</td><td>Returns to Home Page.</td></tr><tr><td>"Help Icon"</td><td>Button</td><td>Used to Load the Help document.</td></tr></table>	Text/Label	Control Type	Additional Info	Add a new Tag	Button	Used to load the "Tag" screen.	Assign or Unassign Tag	Button	Used to load the "Assign or Unassign Tag" screen.	Search	Text-Box	Used for capturing Search criteria for the specific Tag.	Tag Management	Table	Displays a list of Tags.	"Pen icon"	Button	Used to navigate to the specific Tag view.	"Back"	Button	Returns to Home Page.	"Help Icon"	Button	Used to Load the Help document.
			Text/Label	Control Type	Additional Info																						
			Add a new Tag	Button	Used to load the "Tag" screen.																						
			Assign or Unassign Tag	Button	Used to load the "Assign or Unassign Tag" screen.																						
			Search	Text-Box	Used for capturing Search criteria for the specific Tag.																						
			Tag Management	Table	Displays a list of Tags.																						
			"Pen icon"	Button	Used to navigate to the specific Tag view.																						
			"Back"	Button	Returns to Home Page.																						
			"Help Icon"	Button	Used to Load the Help document.																						

			The table is populated with data from the following by using a FIRESQL READ query: <table><tr><th>Entity</th><th>Attributes</th><th>Input Name</th></tr><tr><td>Tag</td><td>TagID</td><td>Tag ID</td></tr><tr><td>TagStatus</td><td>TagStatus</td><td>Tag Status</td></tr></table> The FIRESQL READ query will display the TagStatus in the TagStatus table that corresponds with the TagStatusID in the Tag and TagStatus tables.	Entity	Attributes	Input Name	Tag	TagID	Tag ID	TagStatus	TagStatus	Tag Status
	Entity	Attributes	Input Name									
	Tag	TagID	Tag ID									
	TagStatus	TagStatus	Tag Status									
Step 3: The Administrator enters the search criteria that they wish to search by. They do so by typing in a keyword in the "Search" Text box.		Step 4: The system uses a FIRESQL READ query to READ the following details in the database: [alt] <table><tr><th>Entity</th><th>Attributes</th></tr><tr><td>Tag</td><td>TagID</td></tr></table>	Entity	Attributes	Tag	TagID						
Entity	Attributes											
Tag	TagID											
		Step 5: The following details are loaded and used to populate the "Tag Management" table by using a FIRESQL READ query: <table><tr><th>Output</th><th>Attribute</th><th>Entity</th></tr><tr><td>Tag ID</td><td>TagID</td><td>Tag</td></tr><tr><td>Tag Status</td><td>TagStatus</td><td>TagStatus</td></tr></table> The FIRESQL READ query will display the TagStatus in the TagStatus table that corresponds with the TagStatusID in the Tag and TagStatus tables.	Output	Attribute	Entity	Tag ID	TagID	Tag	Tag Status	TagStatus	TagStatus	
Output	Attribute	Entity										
Tag ID	TagID	Tag										
Tag Status	TagStatus	TagStatus										
Step 6: The Administrator clicks on the "Pen icon" Tag Button in the "Action" table column.		Step 7: The System will load the "Tag" screen and displays it with the following controls and layout: <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr></table>	Text/Label	Control Type	Additional Info							
Text/Label	Control Type	Additional Info										

			Menu	Menu-icon	Used to Navigate the side menu-bar.
			Tag	Label	Heading for the pop-up displayed.
			"Help icon"	Button	Used to Load the Help document.
			Tag ID	Label	Used to indicate what the corresponding read-only text box is used for.
			Tag Status	Label	Used to indicate what the corresponding read-only text box is used for.
			Date Installed	Label	Used to indicate what the corresponding date picker is used for.
			Delete Tag	Button	Used to Delete the Tag details.
			Update Tag	Button	Used to update the Tag details.

			the database where the TagID is the same as the one searched by using a FIRESQL READ query: <table><tr><th>Output</th><th>Attribute</th><th>Entity</th></tr><tr><td>Tag ID</td><td>TagID</td><td>Tag</td></tr><tr><td>Tag Status</td><td>TagStatus</td><td>TagStatus</td></tr><tr><td>Date Installed</td><td>DateInstalled</td><td>Tag</td></tr></table> The FIRESQL READ query will display the TagStatus in the TagStatus table that corresponds with the TagStatusID in the Tag and TagStatus tables.	Output	Attribute	Entity	Tag ID	TagID	Tag	Tag Status	TagStatus	TagStatus	Date Installed	DateInstalled	Tag
	Output	Attribute	Entity												
Tag ID	TagID	Tag													
Tag Status	TagStatus	TagStatus													
Date Installed	DateInstalled	Tag													
			Step 8: The system creates and saves the following in the database by using a FIRESQL INSERT query: <table><tr><th>AuditTrail Table</th></tr><tr><td>AuditTrailID</td></tr><tr><td>DateTime</td></tr><tr><td>TransactionDescription</td></tr><tr><td>UserID</td></tr></table>	AuditTrail Table	AuditTrailID	DateTime	TransactionDescription	UserID							
AuditTrail Table															
AuditTrailID															
DateTime															
TransactionDescription															
UserID															
Alternate Courses:	Alt Step 5: The specific Tag does not exist in the database. The system will notify the Administrator of this with the following: <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>"Tag not found."</td><td>Label</td><td>Used to inform the Administrator of the result.</td></tr><tr><td>"X"</td><td>Banner (with a light red</td><td>Used to display the result message.</td></tr></table>			Text/Label	Control Type	Additional Info	"Tag not found."	Label	Used to inform the Administrator of the result.	"X"	Banner (with a light red	Used to display the result message.			
Text/Label	Control Type	Additional Info													
"Tag not found."	Label	Used to inform the Administrator of the result.													
"X"	Banner (with a light red	Used to display the result message.													

	background color fill) Redirect to Step 3.
Conclusion:	The use case narrative concludes when the searched Tag details are displayed.
Post-Condition:	• The Tag has been searched successfully. • The AuditTrail table is successfully updated.
Business Rules	• Only an Administrator can search for a Tag on the system.
Implementation Constraints and Specifications:	• None
Assumptions:	• None
Open Issues:	• None

2.3.4.7. Update Tag

System Name:	VVs
Author:	Cameron King
Date:	15/06/2020
Version:	1.0

Use Case Name:	Update Tag	Use Case Type:
Use Case ID:	3.7	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	Administrator	
Primary System Actor	None	
Other Participating Actors:	None	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process of an Administrator updating an existing Tag's details in the system. The Use Case starts when the Administrator navigates by selecting the "Tag Management" option under the "Bicycle Management" Combo box in the Navigation bar. From there, they search for the Tag they want to update and the system will load the "Tag" screen with the Tag details. The system populates the relevant Combo boxes with information of the searched Tag, and the Administrator can make their changes and save.	
Pre-Condition:	The Administrator must be logged into the system.	

Trigger:	The Administrator requests to update an existing Tag in the system.						
Typical Course Of Events:	Actor Action	System Response					
		Manual Action	Automated Action				
	Step 1: The Administrator selects the “Tag Management” option under the “Bicycle Management” Combo box in the Navigation bar.		Step 2: The System invokes Use Case 3.6 “Search Tag” and displays the “Tag” screen with the data of the searched Tag.				
	Step 3: Administrator updates the following Tag details: Date Installed						
	Step 4: The Administrator clicks the “Update Tag” button. [alt]		Step 5: The system captures and ensures that all of the fields are filled in. [alt]				
			Step 6: The system validates that the captured details are in the correct format: [alt] Tag table <table><tr><th>Attribute</th><th>Validation</th></tr><tr><td>DateInstalled</td><td> - Max 10 characters - Year attribute must be no later than 2015. - Format must follow: DD/MM/YYYY 23.Numeric characters and “/” or “-” only.</td></tr></table>	Attribute	Validation	DateInstalled	- Max 10 characters - Year attribute must be no later than 2015. - Format must follow: DD/MM/YYYY 23.Numeric characters and “/” or “-” only.
	Attribute	Validation					
	DateInstalled	- Max 10 characters - Year attribute must be no later than 2015. - Format must follow: DD/MM/YYYY 23.Numeric characters and “/” or “-” only.					
		Step 7: The system prompts the Administrator to confirm the update of the Tag’s details. <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr></table>	Text/Label	Control Type	Additional Info		
Text/Label	Control Type	Additional Info					

			Confirmation	Dialog Box Title	Used to indicate what type of message is being displayed.
			"Are you sure you want to update this Tag?"	Dialog Box Message	Used to inform the Administrator of the action.
			Yes	Button	Used to confirm the action.
			No	Button	Used to decline the action.
	Step 8: The Administrator clicks the "Yes" button. [alt]		Step 9: The system then updates and saves the updated Tag details in the Tag table. The following attributes are updated in the database by using a FIRESQL UPDATE query: Tag Table DateInstalled		
			Step 10: The system notifies the Administrator that the Tag has been updated successfully in the database with the following: [alt]		
			Text/Label	Control Type	Additional Info
			Success	Dialog Box Title	Used to indicate what type of message is being displayed.
			"The Tag has been updated successfully"	Dialog Box Message	Used to inform the Administrator of the result.

			Ok	Button	Used to accept the result.									
			Step 11: The system updates and saves the following in the database by using a FIRESQL INSERT query: AuditTrail Table AuditTrailID DateTime TransactionDescription UserID											
Alternate Courses:	Alt Step 4: The Administrator clicks the "Cancel" button. Terminate Use Case.													
	Alt Step 5a: In the case of an empty date picker, the system notifies the Administrator of the Input error below the input field with the following:													
	<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>"Please fill this field."</td><td>Label (with a light red background color fill)</td><td>Used to inform the Administrator of the result.</td></tr><tr><td>"X"</td><td>Icon</td><td>Used to indicate an error result.</td></tr></table>					Text/Label	Control Type	Additional Info	"Please fill this field."	Label (with a light red background color fill)	Used to inform the Administrator of the result.	"X"	Icon	Used to indicate an error result.
	Text/Label	Control Type	Additional Info											
	"Please fill this field."	Label (with a light red background color fill)	Used to inform the Administrator of the result.											
"X"	Icon	Used to indicate an error result.												
Redirect to Step 3.														
Alt Step 6: In the case of invalid input being entered, the system notifies the Administrator of the Format error below the input field with the following:														
<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>"Invalid input."</td><td>Label (with a light red background color fill)</td><td>Used to inform the Administrator of the result.</td></tr></table>					Text/Label	Control Type	Additional Info	"Invalid input."	Label (with a light red background color fill)	Used to inform the Administrator of the result.				
Text/Label	Control Type	Additional Info												
"Invalid input."	Label (with a light red background color fill)	Used to inform the Administrator of the result.												

	"X"	Icon	Used to indicate an error result.															
	Redirect to Step 3.																	
	Alt Step 8: The Administrator declines to update the Tag by clicking the "No" button. Redirect to Step 4.																	
	Alt Step 10: The Tag was not updated successfully in the database, the system will notify the Administrator of the update error with the following:																	
	<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Failed</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed</td></tr><tr><td>"Update unsuccessful. Please try again."</td><td>Dialog Box Message</td><td>Used to inform the Administrator of the result.</td></tr><tr><td>"Help icon"</td><td>Button</td><td>Used to Load the Help document.</td></tr><tr><td>Ok</td><td>Button</td><td>Used to accept the result.</td></tr></table>			Text/Label	Control Type	Additional Info	Failed	Dialog Box Title	Used to indicate what type of message is being displayed	"Update unsuccessful. Please try again."	Dialog Box Message	Used to inform the Administrator of the result.	"Help icon"	Button	Used to Load the Help document.	Ok	Button	Used to accept the result.
	Text/Label	Control Type	Additional Info															
	Failed	Dialog Box Title	Used to indicate what type of message is being displayed															
"Update unsuccessful. Please try again."	Dialog Box Message	Used to inform the Administrator of the result.																
"Help icon"	Button	Used to Load the Help document.																
Ok	Button	Used to accept the result.																
Redirect to Step 3.																		
Conclusion:	The use case concludes when the Administrator has been notified that the Tag has been updated successfully.																	
Post-Condition:	• An existing Tag is updated in the system with the relevant details. • The AuditTrail table is successfully updated. • The Tag table is successfully updated.																	
Business Rules	• Only an Administrator can update a Tag in the system.																	

Implementation Constraints and Specifications:	• None
Assumptions:	• None
Open Issues:	• None

2.3.4.8. Delete Tag

System Name:	VVs
Author:	Cameron King
Date:	15/06/2020
Version:	1.0

Use Case Name:	Delete Tag	Use Case Type:
Use Case ID:	3.8	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	Administrator	
Primary System Actor	None	
Other Participating Actors:	None	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process of an Administrator deleting an existing Tag's details from the system. The Use Case starts when the Administrator navigates by selecting the "Tag Management" option under the "Bicycle Management" Combo box in the Navigation bar. From there, they search for the Tag they want to delete and the system displays the "Tag" screen with the Tag details to be deleted. The system populates the relevant Combo boxes with information of the searched Tag, and the Administrator can make their deletion.	
Pre-Condition:	The Administrator must be logged into the system.	

Trigger:	The Administrator requests to delete an existing Tag from the system.																	
Typical Course Of Events:	Actor Action	System Response																
		Manual Action	Automated Action															
	Step 1: The Administrator selects the "Tag Management" option under the "Bicycle Management" Combo box in the Navigation bar.		Step 2: The System invokes Use Case 3.6 "Search Tag" and displays the "Tag" screen with the data of the searched Tag.															
	Step 3: The Administrator clicks on the "Delete Tag" button.[alt]		Step 4: The system prompts the Administrator to confirm the deletion of the Tag's details with the following:															
			<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Confirmation</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed.</td></tr><tr><td>"Are you sure you want to delete this Tag?"</td><td>Dialog Box Message</td><td>Used to inform the Administrator of the action.</td></tr><tr><td>Yes</td><td>Button</td><td>Used to confirm the action.</td></tr><tr><td>No</td><td>Button</td><td>Used to decline the action.</td></tr></table>		Text/Label	Control Type	Additional Info	Confirmation	Dialog Box Title	Used to indicate what type of message is being displayed.	"Are you sure you want to delete this Tag?"	Dialog Box Message	Used to inform the Administrator of the action.	Yes	Button	Used to confirm the action.	No	Button
Text/Label			Control Type	Additional Info														
Confirmation			Dialog Box Title	Used to indicate what type of message is being displayed.														
"Are you sure you want to delete this Tag?"	Dialog Box Message	Used to inform the Administrator of the action.																
Yes	Button	Used to confirm the action.																
No	Button	Used to decline the action.																

	Step 5: The Administrator clicks the "Yes" button. [alt]		Step 6: The system then removes the Tag details in the Tag table. The following attributes are deleted in the database by using a FIRESQL DELETE query: Tag Table TagID TagStatusID DateInstalled												
			Step 7: The system notifies the Administrator that the Tag has been deleted successfully from the database with the following: [alt] <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Success</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed.</td></tr><tr><td>"The Tag has been deleted successfully"</td><td>Dialog Box Message</td><td>Used to inform the Administrator of the result.</td></tr><tr><td>Ok</td><td>Button</td><td>Used to accept the result.</td></tr></table>	Text/Label	Control Type	Additional Info	Success	Dialog Box Title	Used to indicate what type of message is being displayed.	"The Tag has been deleted successfully"	Dialog Box Message	Used to inform the Administrator of the result.	Ok	Button	Used to accept the result.
	Text/Label	Control Type	Additional Info												
Success	Dialog Box Title	Used to indicate what type of message is being displayed.													
"The Tag has been deleted successfully"	Dialog Box Message	Used to inform the Administrator of the result.													
Ok	Button	Used to accept the result.													
		Step 8: The system updates and saves the following in the database by using a FIRESQL INSERT query: AuditTrail Table AuditTrailID DateTime													

			TransactionDescription UserID
Alternate Courses:	Alt Step 3: The Administrator clicks the "Cancel" button. Terminate Use Case.		
	Alt Step 5: The Administrator declines to delete the Tag by clicking the "No" button. Redirect to Step 2.		
	Alt Step 7: The Administrator is notified of the unsuccessful delete action with the following:		
	Text/Label	Control Type	Additional Info
	Failed	Dialog Box Title	Used to indicate what type of message is being displayed
	"Delete unsuccessful. Please try again."	Dialog Box Message	Used to inform the Administrator of the result.
	"Help icon"	Button	Used to Load the Help document.
	Ok	Button	Used to accept the result.
	Redirect to Step 2.		
Conclusion:	The use case concludes when the Administrator has been notified that the Tag has been successfully deleted.		
Post-Condition:	• An existing Tag is deleted from the system with all the relevant details being removed. • The AuditTrail table is successfully updated. • The Tag table is successfully updated with the deleted details being removed.		
Business Rules	• Only an Administrator can delete a Tag from the system.		

Implementation Constraints and Specifications:	• None
Assumptions:	• None
Open Issues:	• None

2.3.5. Sub-system 4: Student

2.3.5.1. Add Student

System Name:	VVs
Author:	Gareth Phillips
Date:	15/06/2020
Version:	1.0

Use Case Name:	Add Student	Use Case Type:
Use Case ID:	4.1	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	School Administrator	
Primary System Actor	None	
Other Participating Actors:	None	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process of an Administrator adding a new Student to the system. The Use Case starts when the Administrator navigates by selecting the "Student Management" option under the "System Management" Combo box in the Navigation bar. From there, they click the "Add Student" button and the system will load the "Student" screen. The system populates the relevant Combo boxes with information necessary for selection, and the corresponding labels, text boxes and buttons, icons, logos, and banners.	

Pre-Condition:	The Administrator must be logged into the system.				
Trigger:	The Administrator requests to add a new Student to the system.				
Typical Course Of Events:	Actor Action	System Response			
		Manual Action	Automated Action		
	Step 1: The Administrator selects the "Student Management" option under the "Student Management" Combo box in the Navigation bar.		Step 2: The system Displays the "Student Management" screen and populates it with the following details:		
			Text/Label	Control Type	Additional Info
			Navigation	Navigation-bar	Used to Navigate the side site.
			Add	Button	Used to load the "Student" screen.
			Search	Button	Used to search for a Student.
			Search	Text-Box	Used for capturing Search criteria for the specific Student.
			Student Management	Table	Displays a list of Students.
"Pen icon"			Button	Used to navigate to the specific Student view.	

			The table is populated by using a FIRESQL READ query with data from the following: <table><tr><th>Entity</th><th>Attributes</th><th>Input Name</th></tr><tr><td>Student</td><td>StudentName</td><td>Name</td></tr><tr><td>Student</td><td>StudentSurname</td><td>Surname</td></tr><tr><td>Bicycle</td><td>BicycleCode</td><td>Bicycle Code</td></tr></table>	Entity	Attributes	Input Name	Student	StudentName	Name	Student	StudentSurname	Surname	Bicycle	BicycleCode	Bicycle Code						
	Entity	Attributes	Input Name																		
	Student	StudentName	Name																		
	Student	StudentSurname	Surname																		
	Bicycle	BicycleCode	Bicycle Code																		
	Step 3: The Administrator clicks on the "Add" button.		Step 4: The system will load the "Student" screen and displays it with the following controls and layout: <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Navigation</td><td>Navigation-bar</td><td>Used to Navigate the side site.</td></tr><tr><td>Student</td><td>Label</td><td>Heading for the pop-up displayed.</td></tr><tr><td>Name</td><td>Label</td><td>Used to indicate what the corresponding text box is used for.</td></tr><tr><td>Surname</td><td>Label</td><td>Used to indicate what the corresponding text box is used for.</td></tr><tr><td>Phone Number</td><td>Label</td><td>Used to indicate what the</td></tr></table>	Text/Label	Control Type	Additional Info	Navigation	Navigation-bar	Used to Navigate the side site.	Student	Label	Heading for the pop-up displayed.	Name	Label	Used to indicate what the corresponding text box is used for.	Surname	Label	Used to indicate what the corresponding text box is used for.	Phone Number	Label	Used to indicate what the
	Text/Label	Control Type	Additional Info																		
	Navigation	Navigation-bar	Used to Navigate the side site.																		
	Student	Label	Heading for the pop-up displayed.																		
	Name	Label	Used to indicate what the corresponding text box is used for.																		
Surname	Label	Used to indicate what the corresponding text box is used for.																			
Phone Number	Label	Used to indicate what the																			

					corresponding text box is used for.
			Name	Text box	Used to capture text for the Student Name details.
			Surname	Text box	Used to capture text for the Student Surname details.
			Phone Number	Text box	Used to capture text for the Student PhoneNumber details.
			Date of Birth	Date Picker	Used to select a students date of birth
			Village	Combo box	Used to Select the Student Village and is Populated by content from the ERD with options as indicated in the table below.
			School	Combo box	Used to Select the Student School and is

					Populated by content from the ERD with options as indicated in the table below.
			Guardian	Label	Heading for the pop-up displayed.
			Guardian Name	Text box	Used to capture text for the Guardian Name details.
			Guardian Email	Text box	Used to capture text for the Guardian Email details.
			Guardian Phone Number	Text box	Used to capture text for the Guardian Phone Number details.
			Create	Button	Facilitates creating a student.
			Cancel	Button	Facilitates uploading a picture.

			The Village Combo box options are populated by using a FIRESQL READ query with data from the following: <table><tr><th>Entity</th><th>Attributes</th></tr><tr><td>Village</td><td>VillageName</td></tr></table> The School Combo box options are populated by using a FIRESQL READ query with data from the following: <table><tr><th>Entity</th><th>Attributes</th></tr><tr><td>School</td><td>SchoolName</td></tr></table>	Entity	Attributes	Village	VillageName	Entity	Attributes	School	SchoolName
	Entity	Attributes									
	Village	VillageName									
	Entity	Attributes									
	School	SchoolName									
Step 5: Administrator inputs the following Student details:											
• Name • Surname • Phone Number • Date Of Birth • Village • School • Guardian Name • Guardian Email • Guardian Phone Number											
Step 6: The Administrator clicks the “Create” button. [alt]		Step 7: The system captures and ensures that all of the fields are filled in and an option is selected in the Combo box and a date has been selected. [alt]									
		Step 8: The system validates that the captured details are in the correct format: [alt]									
		<i>Student table</i> <table><tr><th>Attribute</th><th>Validation</th></tr></table>	Attribute	Validation							
Attribute	Validation										

			<table><tr><td>Name</td><td>24. Max 100 characters 25. Text characters only</td></tr><tr><td>Surname</td><td> - Max 100 characters - Text characters only </td></tr><tr><td>Phone Number</td><td> - Max 15 units </td></tr></table> <i>Guardian table</i> <table><tr><th>Attribute</th><th>Validation</th></tr><tr><td>Guardian Name</td><td> - Max 100 characters - Text characters only </td></tr><tr><td>Guardian Email</td><td> - Max 100 characters </td></tr><tr><td>Guardian Phone</td><td> - Max 15 units </td></tr></table>	Name	24. Max 100 characters 25. Text characters only	Surname	- Max 100 characters - Text characters only	Phone Number	Max 15 units	Attribute	Validation	Guardian Name	- Max 100 characters - Text characters only	Guardian Email	Max 100 characters	Guardian Phone	Max 15 units
	Name	24. Max 100 characters 25. Text characters only															
	Surname	- Max 100 characters - Text characters only															
	Phone Number	Max 15 units															
	Attribute	Validation															
	Guardian Name	- Max 100 characters - Text characters only															
	Guardian Email	Max 100 characters															
	Guardian Phone	Max 15 units															
			Step 9: The system verifies that no duplicate Students exists on the system by comparing the following: [alt] <table><tr><th>Input</th><th>Attribute</th><th>Entity</th></tr><tr><td>Student Name</td><td>StudentName</td><td>Student</td></tr><tr><td>Student Surname</td><td>StudentSurname</td><td>Student</td></tr></table>	Input	Attribute	Entity	Student Name	StudentName	Student	Student Surname	StudentSurname	Student					
	Input	Attribute	Entity														
Student Name	StudentName	Student															
Student Surname	StudentSurname	Student															
		Step 10: The system prompts the Administrator to confirm to add the new Student with the following: <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Confirmation</td><td>Dialog Box Title</td><td>Used to indicate what type of</td></tr></table>	Text/Label	Control Type	Additional Info	Confirmation	Dialog Box Title	Used to indicate what type of									
Text/Label	Control Type	Additional Info															
Confirmation	Dialog Box Title	Used to indicate what type of															

			message is being displayed.	
			"Are you sure you want to add this Student?"	Dialog Box Message
			Yes	Button
			No	Button

	Step 11: The Administrator clicks the “Yes” button. [alt]		Step 12: The Student is assigned a unique StudentID in the Student table which is set to the next consecutive number after the previous StudentID in the database, the system then creates and saves the new Student in the Student table. The following attributes are saved in the database by using a FIRESQL INSERT query: Student Table StudentID StudentName StudentSurname PhoneNumber StudentDoB VillageID SchoolID GuardianID StudentName

			Step 13: The Guardian is assigned a unique GuardianID in the Guardian table which is set to the next consecutive number after the previous GuardianID in the database, the system then creates and saves the new Guardian in the Guardian table. The following attributes are saved in the database by using a FIRESQL INSERT query: Guardian Table GuardianID GuardianName GuardianEmail GuardianPhone												
			Step 14: The system notifies the Administrator that the Student has been added successfully to the database with the following: [alt] <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Success</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed.</td></tr><tr><td>"Student has been added successfully"</td><td>Dialog Box Message</td><td>Used to inform the Administrator of the result.</td></tr><tr><td>Ok</td><td>Button</td><td>Used to accept the result.</td></tr></table>	Text/Label	Control Type	Additional Info	Success	Dialog Box Title	Used to indicate what type of message is being displayed.	"Student has been added successfully"	Dialog Box Message	Used to inform the Administrator of the result.	Ok	Button	Used to accept the result.
	Text/Label	Control Type	Additional Info												
Success	Dialog Box Title	Used to indicate what type of message is being displayed.													
"Student has been added successfully"	Dialog Box Message	Used to inform the Administrator of the result.													
Ok	Button	Used to accept the result.													
		Step 15: The system creates and saves the following in the database by using a FIRESQL INSERT query:													

				AuditTrail Table AuditTrailID DateTime TransactionDescription UserID									
Alternate Courses:	Alt Step 6: The Administrator clicks the "Cancel" button. Terminate Use Case.												
	Alt Step 8a: In the case of an empty Text box, the system notifies the Administrator of the Input error below the input field with the following:												
	<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>"Please fill this field."</td><td>Label (with a light red background color fill)</td><td>Used to inform the Administrator of the result.</td></tr><tr><td>"X"</td><td>Icon</td><td>Used to indicate an error result.</td></tr></table>				Text/Label	Control Type	Additional Info	"Please fill this field."	Label (with a light red background color fill)	Used to inform the Administrator of the result.	"X"	Icon	Used to indicate an error result.
	Text/Label	Control Type	Additional Info										
	"Please fill this field."	Label (with a light red background color fill)	Used to inform the Administrator of the result.										
	"X"	Icon	Used to indicate an error result.										
	Redirect to Step 5.												
	Alt Step 8b: In the case of a Combo box option not selected, the system notifies the Administrator of the Input error below the input field with the following:												
	<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>"Please select an option."</td><td>Label (with a light red background color fill)</td><td>Used to inform the Administrator of the result.</td></tr><tr><td>"X"</td><td>Icon</td><td>Used to indicate an error result.</td></tr></table>				Text/Label	Control Type	Additional Info	"Please select an option."	Label (with a light red background color fill)	Used to inform the Administrator of the result.	"X"	Icon	Used to indicate an error result.
	Text/Label	Control Type	Additional Info										
"Please select an option."	Label (with a light red background color fill)	Used to inform the Administrator of the result.											
"X"	Icon	Used to indicate an error result.											
Redirect to Step 5.													

Alt Step 11: In the case of invalid input being entered, the system notifies the Administrator of the Format error below the input field with the following:

Text/Label	Control Type	Additional Info
"Invalid input."	Label (with a light red background color fill)	Used to inform the Administrator of the result.
"X"	Icon	Used to indicate an error result.

Redirect to Step 5.

Alt Step 9: The inputted Student already exists in the system; the system will notify the Administrator of the duplication error with the following:

Text/Label	Control Type	Additional Info
Error	Dialog Box Title	Used to indicate what type of message is being displayed.
"The Student already exists. Please try again."	Dialog Box Message	Used to inform the Administrator of the result.
Ok	Button	Used to accept the result.

Redirect to Step 5.

Alt Step 11: The Administrator declines to add the new Student by clicking the "No" button. Redirect to Step 5.

Alt Step 14: The Student was not added successfully to the database, the system will notify the Administrator of the save error with the following:

	Text/Label	Control Type	Additional Info
	Failed	Dialog Box Title	Used to indicate what type of message is being displayed
	"Create unsuccessful. Please try again."	Dialog Box Message	Used to inform the Administrator of the result.
	Ok	Button	Used to accept the result.
	Redirect to Step 5.		
Conclusion:	The use case concludes when the Administrator has been notified that the Student has been added successfully.		
Post-Condition:	• A new Student is added to the system with the relevant details. • The AuditTrail table is successfully updated. • The Student table is successfully updated.		
Business Rules	• Only an Administrator can add a Student to the system.		
Implementation Constraints and Specifications:	• None		
Assumptions:	• None		
Open Issues:	• None		

2.3.5.2. Search Student

System Name:	VVs
Author:	Gareth Phillips
Date:	15/06/2020
Version:	1.0

Use Case Name:	Search Student	Use Case Type:
Use Case ID:	4.2	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	School Administrator	
Primary System Actor	None	
Other Participating Actors:	None	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process of an Administrator searching a Student on the system. The Use Case starts when the Administrator navigates by selecting the " Student Management" option under the "System Management" Combo box in the Navigation bar. The system populates the table with the information of all the Students.	
Pre-Condition:	The Administrator must be logged into the system.	
Trigger:	The Administrator requests to search for a Student on the system.	

Typical Course Of Events:	Actor Action	System Response			
		Manual Action	Automated Action		
	Step 1: The Administrator selects the “Student Management” option under the “System Management” Combo box in the Navigation bar.		Step 2: The system Displays the “Student Management” screen and populates it with the following details:		
			Text/Label	Control Type	Additional Info
			Add	Button	Used to load the “Student” screen.
			Search	Button	Used to search a student.
			Search	Text-Box	Used for capturing Search criteria for the specific Student.
			Student Management	Table	Displays a list of Students.
			“Pen icon”	Button	Used to navigate to the specific Student view.
			“Back”	Button	Returns to Home Page.
“Help Icon”			Button	Used to Load the Help document.	
		The table is populated by using a FIRESQL READ query with data from the following:			

			<table><tr><th>Entity</th><th>Attributes</th><th>Input Name</th></tr><tr><td>Student</td><td>StudentName</td><td>Name</td></tr><tr><td>Student</td><td>StudentSurname</td><td>Surname</td></tr><tr><td>Bicycle</td><td>BicycleCode</td><td>Bicycle Code</td></tr></table>	Entity	Attributes	Input Name	Student	StudentName	Name	Student	StudentSurname	Surname	Bicycle	BicycleCode	Bicycle Code
	Entity	Attributes	Input Name												
	Student	StudentName	Name												
	Student	StudentSurname	Surname												
	Bicycle	BicycleCode	Bicycle Code												
	Step 3: The Administrator enters the search criteria that they wish to search by. They do so by typing in a keyword in the "Search" Text box.														
	Step 4: The Administrator clicks the "search button".		Step 5: The system uses a FIRESQL READ query to READ the following details in the database: [alt] <table><tr><th>Entity</th><th>Attributes</th></tr><tr><td>Student</td><td>StudentName</td></tr><tr><td>Student</td><td>StudentSurname</td></tr><tr><td>Bicycle</td><td>BicycleCode</td></tr></table>	Entity	Attributes	Student	StudentName	Student	StudentSurname	Bicycle	BicycleCode				
	Entity	Attributes													
	Student	StudentName													
	Student	StudentSurname													
Bicycle	BicycleCode														
		Step 5: The following details are loaded and used to populate the "Student Management" table by using a FIRESQL READ query: <table><tr><th>Output</th><th>Attribute</th><th>Entity</th></tr><tr><td>Student Name</td><td>StudentName</td><td>Student</td></tr><tr><td>Student Surname</td><td>StudentSurname</td><td>Student</td></tr><tr><td>Bicycle Code</td><td>BicycleCode</td><td>Bicycle</td></tr></table>	Output	Attribute	Entity	Student Name	StudentName	Student	Student Surname	StudentSurname	Student	Bicycle Code	BicycleCode	Bicycle	
Output	Attribute	Entity													
Student Name	StudentName	Student													
Student Surname	StudentSurname	Student													
Bicycle Code	BicycleCode	Bicycle													

	Step 6: The Administrator clicks on the "Pen icon" Student Button in the "Action" table column.		Step 7: The System will load the "Student" screen and displays it with the following controls and layout:		
			Text/Label	Control Type	Additional Info
			Menu	Menu-icon	Used to Navigate the side menu-bar.
			Student	Label	Heading for the pop-up displayed.
			Choose Picture	Button	Facilitates uploading a picture.
			Student Picture	Picture Image	Displays the uploaded picture.
			Name	Label	Used to indicate what the corresponding text box is used for.
			Surname	Label	Used to indicate what the corresponding text box is used for.
			Phone Number	Label	Used to indicate what the corresponding

					text box is used for.
			Name	Text box	Used to capture text for the Student Name details.
			Surname	Text box	Used to capture text for the Student Surname details.
			Phone Number	Text box	Used to capture text for the Student PhoneNumber details.
			Date of Birth	Date Picker	Used to select a students date of birth
			Village	Combo box	Used to Select the Student Village and is Populated by content from the ERD with options as indicated in the table below.
			School	Combo box	Used to Select the Student School and is

			Populated by content from the ERD with options as indicated in the table below.		
			Guardian	Label	Used to indicate what the corresponding text boxes are used for.
			Guardian Name	Label	Used to indicate what the corresponding text box is used for.
			Guardian Email	Label	Used to indicate what the corresponding text box is used for.
			Guardian Phone Number	Label	Used to indicate what the corresponding text box is used for.
			Guardian Name	Text box	Used to capture text for the Guardian Name details.

			Guardian Email	Text box	Used to capture text for the Guardian Email details.
			Guardian Phone Number	Text box	Used to capture text for the Guardian Phone Number details.
			Delete	Button	Used to Delete a specific Student.
			Update	Button	Used to update Student.
			The following fields are populated with data retrieved from the database where the StudentID is the same as the one searched by using a FIRESQL READ query:		
			Output	Attribute	Entity
			StudentID	StudentID	Student
			Name	StudentName	Student
			Surname	StudentSurname	Student
			Phone Number	PhoneNumber	Student
			Date Of Birth	StudentDoB	Student
			Gender	StudentGender	Student

			Village	VillageID	Student									
			School	SchoolID	Student									
			Bicycle	BicycleID	Student									
			Guardian Name	GuardianName	Student									
			Guardian Email	GuardianEmail	Student									
			Guardian Phone	GuardianPhone	Student									
					Step 8: The system creates and saves the following in the database by using a FIRESQL INSERT query:									
AuditTrail Table														
AuditTrailID														
DateTime														
TransactionDescription														
UserID														
Alternate Courses:	Alt Step 5: The specific Student does not exist in the database. The system will notify the Administrator of this with the following:													
	<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>"Student not found."</td><td>Label</td><td>Used to inform the Administrator of the result.</td></tr><tr><td>"X"</td><td>Banner (with a light red background color fill)</td><td>Used to display the result message.</td></tr></table>					Text/Label	Control Type	Additional Info	"Student not found."	Label	Used to inform the Administrator of the result.	"X"	Banner (with a light red background color fill)	Used to display the result message.
	Text/Label	Control Type	Additional Info											
	"Student not found."	Label	Used to inform the Administrator of the result.											
"X"	Banner (with a light red background color fill)	Used to display the result message.												

	Redirect to Step 3.
Conclusion:	The use case narrative concludes when the searched Student details are displayed.
Post-Condition:	- The Student has been searched successfully. - The AuditTrail table is successfully updated.
Business Rules	Only an Administrator can search for a Student on the system.
Implementation Constraints and Specifications:	None
Assumptions:	None
Open Issues:	None

2.3.5.3. Update Student

System Name:	VVs
Author:	Gareth Phillips
Date:	15/06/2020
Version:	1.0

Use Case Name:	Update Student	Use Case Type:
Use Case ID:	4.3	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	School Administrator	
Primary System Actor	None	
Other Participating Actors:	None	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process of an Administrator updating an existing Student details in the system. The Use Case starts when the Administrator navigates by selecting the " Student Management" option under the "System Management" Combo box in the Navigation bar. From there, they search for the Student they want to update, the system then will load the "Student" screen with the Student details. The Administrator can then make their changes and save it. The system populates the relevant Combo boxes with information necessary for selection, and the corresponding labels, text boxes and buttons, icons, logos, and banners.	
Pre-Condition:	The Administrator must be logged into the system.	

Trigger:	The Administrator requests to update an existing Student to the system.			
Typical Course Of Events:	Actor Action	System Response		
		Manual Action	Automated Action	
	Step 1: The Administrator selects the “Student Management” option under the “System Management” Combo box in the Navigation bar.		Step 2: The System invokes Use Case 4.2 “Search Student” and displays the “Student” screen with the data of the searched Fault Task.	
	Step 3: Administrator updates the following Fault Task details: • Name • Surname • Phone Number • Date Of Birth • Gender • Village • School • Bicycle • Guardian Name • Guardian Email • Guardian Phone Number			
	Step 4: The Administrator clicks the “Update Student” button. [alt]		Step 5: The system captures and ensures that all of the fields are filled in and an option is selected in the Combo box and a date has been selected. [alt]	
			Step 6: The system validates that the captured details are in the correct format: [alt] Student table <table><thead><tr><th>Attribute</th><th>Validation</th></tr></thead></table>	Attribute
Attribute	Validation			

			<table><tr><td>Name</td><td>26.Max 100 characters 27.Text characters only</td></tr><tr><td>Surname</td><td> - Max 100 characters - Text characters only </td></tr><tr><td>Phone Number</td><td> - Max 15 units </td></tr></table> <i>Guardian table</i> <table><tr><th>Attribute</th><th>Validation</th></tr><tr><td>Guardian Name</td><td> - Max 100 characters - Text characters only </td></tr><tr><td>Guardian Email</td><td> - Max 100 characters </td></tr><tr><td>Guardian Phone</td><td> - Max 15 units </td></tr></table>	Name	26.Max 100 characters 27.Text characters only	Surname	- Max 100 characters - Text characters only	Phone Number	Max 15 units	Attribute	Validation	Guardian Name	- Max 100 characters - Text characters only	Guardian Email	Max 100 characters	Guardian Phone	Max 15 units
	Name	26.Max 100 characters 27.Text characters only															
	Surname	- Max 100 characters - Text characters only															
	Phone Number	Max 15 units															
	Attribute	Validation															
	Guardian Name	- Max 100 characters - Text characters only															
	Guardian Email	Max 100 characters															
	Guardian Phone	Max 15 units															
			Step 7: The system prompts the Administrator to confirm the update of the Students’ details. <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Confirmation</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed.</td></tr><tr><td>“Are you sure you want to update this Student?”</td><td>Dialog Box Message</td><td>Used to inform the Administrator of the action.</td></tr><tr><td>Yes</td><td>Button</td><td>Used to confirm the action.</td></tr></table>	Text/Label	Control Type	Additional Info	Confirmation	Dialog Box Title	Used to indicate what type of message is being displayed.	“Are you sure you want to update this Student?”	Dialog Box Message	Used to inform the Administrator of the action.	Yes	Button	Used to confirm the action.		
	Text/Label	Control Type	Additional Info														
Confirmation	Dialog Box Title	Used to indicate what type of message is being displayed.															
“Are you sure you want to update this Student?”	Dialog Box Message	Used to inform the Administrator of the action.															
Yes	Button	Used to confirm the action.															

			No	Button	Used to decline the action.
	Step 8: The Administrator clicks the "Yes" button. [alt]		Step 9: The system then updates and saves the updated Student details in the Student Table and the updated Guardian Details in the Guardian Table. The following attributes are updated in the database by using a FIRESQL UPDATE query: Guardian Table GuardianName GuardianEmail GuardianPhone Student Table StudentName StudentSurname PhoneNumber StudentDoB StudentGender VillageID SchoolID		

			BicycleID GuardianID												
			Step 10: The system notifies the Administrator that the Student has been updated successfully in the database with the following: [alt] <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Success</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed.</td></tr><tr><td>"The Student has been updated successfully"</td><td>Dialog Box Message</td><td>Used to inform the Administrator of the result.</td></tr><tr><td>Ok</td><td>Button</td><td>Used to accept the result.</td></tr></table>	Text/Label	Control Type	Additional Info	Success	Dialog Box Title	Used to indicate what type of message is being displayed.	"The Student has been updated successfully"	Dialog Box Message	Used to inform the Administrator of the result.	Ok	Button	Used to accept the result.
	Text/Label	Control Type	Additional Info												
	Success	Dialog Box Title	Used to indicate what type of message is being displayed.												
"The Student has been updated successfully"	Dialog Box Message	Used to inform the Administrator of the result.													
Ok	Button	Used to accept the result.													
		Step 11: The system updates and saves the following in the database by using a FIRESQL INSERT query: AuditTrail Table AuditTrailID DateTime TransactionDescription UserID													
Alternate Courses:	Alt Step 4: The Administrator clicks the "Cancel" button. Terminate Use Case.														

Alt Step 5a: In the case of an empty Text box, the system notifies the Administrator of the Input error below the input field with the following:

Text/Label	Control Type	Additional Info
"Please fill this field."	Label (with a light red background color fill)	Used to inform the Administrator of the result.
"X"	Icon	Used to indicate an error result.

Redirect to Step 3.

Alt Step 5b: In the case of a Combo box option not selected, the system notifies the Administrator of the Input error below the input field with the following:

Text/Label	Control Type	Additional Info
"Please select an option."	Label (with a light red background color fill)	Used to inform the Administrator of the result.
"X"	Icon	Used to indicate an error result.

Redirect to Step 3.

Alt Step 6: In the case of invalid input being entered, the system notifies the Administrator of the Format error below the input field with the following:

Text/Label	Control Type	Additional Info
"Invalid input."	Label (with a light red background color fill)	Used to inform the Administrator of the result.
"X"	Icon	Used to indicate an error result.

	Redirect to Step 3.															
	Alt Step 8: The Administrator declines to update the Fault Task by clicking the "No" button. Redirect to Step 4.															
	Alt Step 10: The Student was not updated successfully in the database, the system will notify the Administrator of the update error with the following:															
	<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Failed</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed</td></tr><tr><td>"Update unsuccessful. Please try again."</td><td>Dialog Box Message</td><td>Used to inform the Administrator of the result.</td></tr><tr><td>"Help icon"</td><td>Button</td><td>Used to Load the Help document.</td></tr><tr><td>Ok</td><td>Button</td><td>Used to accept the result.</td></tr></table>	Text/Label	Control Type	Additional Info	Failed	Dialog Box Title	Used to indicate what type of message is being displayed	"Update unsuccessful. Please try again."	Dialog Box Message	Used to inform the Administrator of the result.	"Help icon"	Button	Used to Load the Help document.	Ok	Button	Used to accept the result.
	Text/Label	Control Type	Additional Info													
	Failed	Dialog Box Title	Used to indicate what type of message is being displayed													
"Update unsuccessful. Please try again."	Dialog Box Message	Used to inform the Administrator of the result.														
"Help icon"	Button	Used to Load the Help document.														
Ok	Button	Used to accept the result.														
Redirect to Step 3.																
Conclusion:	The use case concludes when the Administrator has been notified that the Student has been updated successfully.															
Post-Condition:	• An existing Student is updated in the system with the relevant details. • The AuditTrail table is successfully updated. • The Student table is successfully updated. • The Guardian table is successfully updated.															
Business Rules	• Only an Administrator can update a Student on the system.															

Implementation Constraints and Specifications:	• None
Assumptions:	• None
Open Issues:	• None

2.3.5.4. Delete Student

System Name:	VVs
Author:	Gareth Phillips
Date:	15/06/2020
Version:	1.0

Use Case Name:	Delete Student	Use Case Type:
Use Case ID:	4.4	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	School Administrator	
Primary System Actor	None	
Other Participating Actors:	None	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process of an Administrator deleting an existing Student details from the system. The Use Case starts when the Administrator navigates by selecting the "Student Management" option under the "Student Management" Combo box in the Navigation bar. From there, they search for the Student they want to delete and the system displays the "Student" screen with the Student details to be deleted. The system populates the relevant Labels and corresponding text boxes with information of the searched Student, and the Administrator can make their deletion.	
Pre-Condition:	The Administrator must be logged into the system.	

Trigger:	The Administrator requests to delete an existing Student from the system.																	
Typical Course Of Events:	Actor Action	System Response																
		Manual Action	Automated Action															
	Step 1: The Administrator selects the "Student Management" option under the "System Management" Combo box in the Navigation bar.		Step 2: The System invokes Use Case 4.2 "Search Student" and displays the "Student" screen with the data of the searched Student.															
	Step 3: The Administrator clicks on the "Delete Student" button. [alt]		Step 4: The system prompts the Administrator to confirm the deletion of the Student details with the following:															
			<table><tr><td>Text/Label</td><td>Control Type</td><td>Additional Info</td></tr><tr><td>Confirmation</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed.</td></tr><tr><td>"Are you sure you want to delete this Student?"</td><td>Dialog Box Message</td><td>Used to inform the Administrator of the action.</td></tr><tr><td>Yes</td><td>Button</td><td>Used to confirm the action.</td></tr><tr><td>No</td><td>Button</td><td>Used to decline the action.</td></tr></table>		Text/Label	Control Type	Additional Info	Confirmation	Dialog Box Title	Used to indicate what type of message is being displayed.	"Are you sure you want to delete this Student?"	Dialog Box Message	Used to inform the Administrator of the action.	Yes	Button	Used to confirm the action.	No	Button
Text/Label			Control Type	Additional Info														
Confirmation			Dialog Box Title	Used to indicate what type of message is being displayed.														
"Are you sure you want to delete this Student?"			Dialog Box Message	Used to inform the Administrator of the action.														
Yes	Button	Used to confirm the action.																
No	Button	Used to decline the action.																

	Step 5: The Administrator clicks the "Yes" button. [alt]		Step 6: The system then removes the Student details in the Student table. The following attributes are deleted in the database by using a FIRESQL DELETE query: Student Table StudentID StudentName StudentSurname PhoneNumber StudentDoB StudentGender VillageID SchoolID BicycleID GuardianID Guardian Table GuardianID GuardianName GuardianEmail GuardianPhoneNumber

			Step 7: The system notifies the Administrator that the Student has been deleted successfully from the database with the following: [alt]												
			<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Success</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed.</td></tr><tr><td>"The Student has been deleted successfully"</td><td>Dialog Box Message</td><td>Used to inform the Administrator of the result.</td></tr><tr><td>Ok</td><td>Button</td><td>Used to accept the result.</td></tr></table>	Text/Label	Control Type	Additional Info	Success	Dialog Box Title	Used to indicate what type of message is being displayed.	"The Student has been deleted successfully"	Dialog Box Message	Used to inform the Administrator of the result.	Ok	Button	Used to accept the result.
			Text/Label	Control Type	Additional Info										
			Success	Dialog Box Title	Used to indicate what type of message is being displayed.										
"The Student has been deleted successfully"	Dialog Box Message	Used to inform the Administrator of the result.													
Ok	Button	Used to accept the result.													
		Step 8: The system updates and saves the following in the database by using a FIRESQL INSERT query:													
		<table><tr><th>AuditTrail Table</th></tr><tr><td>AuditTrailID</td></tr><tr><td>DateTime</td></tr><tr><td>TransactionDescription</td></tr><tr><td>UserID</td></tr></table>	AuditTrail Table	AuditTrailID	DateTime	TransactionDescription	UserID								
		AuditTrail Table													
AuditTrailID															
DateTime															
TransactionDescription															
UserID															
Alternate Courses:															
	Alt Step 3: The Administrator clicks the "Cancel" button. Terminate Use Case.														
	Alt Step 5: The Administrator declines to delete the Student by clicking the "No" button. Redirect to Step 2.														
	Alt Step 7: The Administrator is notified of the unsuccessful delete action with the following:														
	<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr></table>	Text/Label	Control Type	Additional Info											
Text/Label	Control Type	Additional Info													

	Failed	Dialog Box Title	Used to indicate what type of message is being displayed
	"Delete unsuccessful. Please try again."	Dialog Box Message	Used to inform the Administrator of the result.
	"Help icon"	Button	Used to Load the Help document.
	Ok	Button	Used to accept the result.
	Redirect to Step 2.		
Conclusion:	The use case concludes when the Administrator has been notified that the Student has been successfully deleted.		
Post-Condition:	• An existing Student is deleted from the system with all the relevant details being removed. • The AuditTrail table is successfully updated. • The Student table is successfully updated with the deleted details being removed.		
Business Rules	• Only an Administrator can delete a Student from the system.		
Implementation Constraints and Specifications:	• None		
Assumptions:	• None		
Open Issues:	• None		

2.3.5.5. Capture Student Mark

System Name:	VVs
Author:	Gareth Phillips
Date:	15/06/2020
Version:	1.0

Use Case Name:	Capture Student Mark	Use Case Type:
Use Case ID:	4.5	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	School Administrator	
Primary System Actor	None	
Other Participating Actors:	None	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process of an Administrator capturing a Student Mark to the system. The Use Case starts when the Administrator navigates by selecting the " Student Mark Management" option under the "System Management" Combo box in the Navigation bar. From there, they click the "Capture Student Mark" button and the system will load the "Capture Student Mark Task" screen. The system populates the corresponding labels, text boxes and buttons, icons, logos, and banners.	
Pre-Condition:	The Administrator must be logged into the system.	

Trigger:	The Administrator requests to add a new Maintenance Task to the system.																	
Typical Course Of Events:	Actor Action	System Response																
		Manual Action	Automated Action															
	Step 1: The Administrator selects the “Student Mark Management” option under the “Student Management” Combo box in the Navigation bar.		Step 2: The System invokes Use Case 4.6 “Search Student Mark” and displays the “Student Mark” screen with the data of the searched Student Mark.															
			Step 2: The system Displays the “Student Mark Management” screen and populates it with the following details: <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Search</td><td>Text-Box</td><td>Used for capturing Search criteria for the specific Student Mark.</td></tr><tr><td>Search</td><td>Button</td><td>Used to initiate a search.</td></tr><tr><td>Student Mark Management</td><td>Table</td><td>Displays a list of Student Marks.</td></tr><tr><td>Capture Mark</td><td>Button</td><td>Used to navigate to the specific Student Mark view.</td></tr></table>	Text/Label	Control Type	Additional Info	Search	Text-Box	Used for capturing Search criteria for the specific Student Mark.	Search	Button	Used to initiate a search.	Student Mark Management	Table	Displays a list of Student Marks.	Capture Mark	Button	Used to navigate to the specific Student Mark view.
	Text/Label	Control Type	Additional Info															
Search	Text-Box	Used for capturing Search criteria for the specific Student Mark.																
Search	Button	Used to initiate a search.																
Student Mark Management	Table	Displays a list of Student Marks.																
Capture Mark	Button	Used to navigate to the specific Student Mark view.																

			The table is populated by using a FIRESQL READ query with data from the following: <table><tr><th>Entity</th><th>Attributes</th><th>Input Name</th></tr><tr><td>Student Mark</td><td>Mark</td><td>Mark</td></tr><tr><td>Student</td><td>StudentName</td><td>Name</td></tr><tr><td>Student</td><td>StudentSurname</td><td>Surname</td></tr></table>	Entity	Attributes	Input Name	Student Mark	Mark	Mark	Student	StudentName	Name	Student	StudentSurname	Surname
	Entity	Attributes	Input Name												
	Student Mark	Mark	Mark												
	Student	StudentName	Name												
	Student	StudentSurname	Surname												
	Step 3: The Administrator clicks on the "Capture Mark" button.		Step 4: The system will load the "Capture Student Mark" screen and displays it with the following controls and layout: <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Student Mark</td><td>Label</td><td>Heading for the pop-up displayed.</td></tr><tr><td>Mark</td><td>Label</td><td>Used to indicate what the corresponding text box is used for.</td></tr><tr><td>Mark Type</td><td>Combo box</td><td>Used to Select the Mark Type and is Populated by content from the ERD with options as indicated in the table below.</td></tr></table>	Text/Label	Control Type	Additional Info	Student Mark	Label	Heading for the pop-up displayed.	Mark	Label	Used to indicate what the corresponding text box is used for.	Mark Type	Combo box	Used to Select the Mark Type and is Populated by content from the ERD with options as indicated in the table below.
	Text/Label	Control Type	Additional Info												
Student Mark	Label	Heading for the pop-up displayed.													
Mark	Label	Used to indicate what the corresponding text box is used for.													
Mark Type	Combo box	Used to Select the Mark Type and is Populated by content from the ERD with options as indicated in the table below.													

			Date DatePicker Used to select the date of Mark
			Mark Text box Used to capture text for the Student Mark details.
			Capture Button Used to capture the new Student Mark Details.
			Cancel Button Used to cancel the action of adding a Student Mark.
			The School Combo box options are populated by using a FIRESQL READ query with data from the following: EntityAttributes Mark TypeMarkType
	Step 5: Administrator inputs the following Student Mark details: • Mark Type • Date • Mark		
	Step 6: The Administrator clicks the “Capture” button. [alt]		Step 7: The system captures and ensures that all of the fields are filled in and an option is selected in the Combo box and a date has been selected. [alt]
			Step 8: The system validates that the captured details are in the correct format: [alt]

			<table><tr><th colspan="2">Mark Type <i>table</i></th></tr><tr><th>Attribute</th><th>Validation</th></tr><tr><td>Mark</td><td> - Max 150 characters - Text characters only </td></tr><tr><td>Date</td><td> - Date inputted </td></tr><tr><td>Mark Type</td><td> - Option selected </td></tr></table>	Mark Type <i>table</i>		Attribute	Validation	Mark	- Max 150 characters - Text characters only	Date	Date inputted	Mark Type	Option selected					
	Mark Type <i>table</i>																	
	Attribute	Validation																
	Mark	- Max 150 characters - Text characters only																
	Date	Date inputted																
	Mark Type	Option selected																
			Step 9: The system prompts the Administrator to confirm to capture the Student Mark with the following: <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Confirmation</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed.</td></tr><tr><td>"Are you sure you want to Capture this Student Mark?"</td><td>Dialog Box Message</td><td>Used to inform the Administrator of the action.</td></tr><tr><td>Yes</td><td>Button</td><td>Used to confirm the action.</td></tr><tr><td>No</td><td>Button</td><td>Used to decline the action.</td></tr></table>	Text/Label	Control Type	Additional Info	Confirmation	Dialog Box Title	Used to indicate what type of message is being displayed.	"Are you sure you want to Capture this Student Mark?"	Dialog Box Message	Used to inform the Administrator of the action.	Yes	Button	Used to confirm the action.	No	Button	Used to decline the action.
	Text/Label	Control Type	Additional Info															
	Confirmation	Dialog Box Title	Used to indicate what type of message is being displayed.															
	"Are you sure you want to Capture this Student Mark?"	Dialog Box Message	Used to inform the Administrator of the action.															
Yes	Button	Used to confirm the action.																
No	Button	Used to decline the action.																

	Step 10: The Administrator clicks the “Yes” button. [alt]		Step 11: The Student Mark is assigned a unique StudentMarkID in the Student Mark table which is set to the next consecutive number after the previous StudentMarkID in the database, the system then creates and saves the new Fault Task in the StudentMark table. The following attributes are saved in the database by using a FIRESQL INSERT query: Student Mark Table StudentMarkID StudentID MarkTypeID Date Mark								
			Step 12: The system notifies the Administrator that the Student Mark has been added successfully to the database with the following: [alt] <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Success</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed.</td></tr><tr><td>“Student Mark has been captured successfully”</td><td>Dialog Box Message</td><td>Used to inform the Administrator of the result.</td></tr></table>	Text/Label	Control Type	Additional Info	Success	Dialog Box Title	Used to indicate what type of message is being displayed.	“Student Mark has been captured successfully”	Dialog Box Message
Text/Label	Control Type	Additional Info									
Success	Dialog Box Title	Used to indicate what type of message is being displayed.									
“Student Mark has been captured successfully”	Dialog Box Message	Used to inform the Administrator of the result.									

			Ok Button Used to accept the result.									
			Step 13: The system creates and saves the following in the database by using a FIRESQL INSERT query: AuditTrail Table AuditTrailID DateTime TransactionDescription UserID									
Alternate Courses:	Alt Step 6: The Administrator clicks the "Cancel" button. Terminate Use Case.											
	Alt Step 7a: In the case of an empty Text box, the system notifies the Administrator of the Input error below the input field with the following:											
	<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>"Please fill this field."</td><td>Label (with a light red background color fill)</td><td>Used to inform the Administrator of the result.</td></tr><tr><td>"X"</td><td>Icon</td><td>Used to indicate an error result.</td></tr></table>			Text/Label	Control Type	Additional Info	"Please fill this field."	Label (with a light red background color fill)	Used to inform the Administrator of the result.	"X"	Icon	Used to indicate an error result.
	Text/Label	Control Type	Additional Info									
"Please fill this field."	Label (with a light red background color fill)	Used to inform the Administrator of the result.										
"X"	Icon	Used to indicate an error result.										
Redirect to Step 5.												
	Alt Step 7b: In the case of a Combo box option not selected, the system notifies the Administrator of the Input error below the input field with the following:											
	<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr></table>			Text/Label	Control Type	Additional Info						
Text/Label	Control Type	Additional Info										

	"Please select an option."	Label (with a light red background color fill)	Used to inform the Administrator of the result.
	"X"	Icon	Used to indicate an error result.
	Redirect to Step 5.		
	Alt Step 8: In the case of invalid input being entered, the system notifies the Administrator of the Format error below the input field with the following:		
	Text/Label	Control Type	Additional Info
	"Invalid input."	Label (with a light red background color fill)	Used to inform the Administrator of the result.
	"X"	Icon	Used to indicate an error result.
	Redirect to Step 3.		
	Alt Step 10: The Administrator declines to capture the Student Mark by clicking the "No" button. Redirect to Step 5.		
	Alt Step 12: The Maintenance Task was not added successfully to the database, the system will notify the Administrator of the save error with the following:		
	Text/Label	Control Type	Additional Info
	Failed	Dialog Box Title	Used to indicate what type of message is being displayed

	"Capture mark unsuccessful. Please try again."	Dialog Box Message	Used to inform the Administrator of the result.
	"Help icon"	Button	Used to Load the Help document.
	Ok	Button	Used to accept the result.
	Redirect to Step 5.		
Conclusion:	The use case concludes when the Administrator has been notified that the Student Mark has been added successfully.		
Post-Condition:	• A new Maintenance Task is added to the system with the relevant details. • The AuditTrail table is successfully updated. • The StudentMark table is successfully updated.		
Business Rules	• Only an Administrator can capture a Student Mark to the system.		
Implementation Constraints and Specifications:	• None		
Assumptions:	• None		
Open Issues:	• None		

2.3.5.6. Search Student Mark

System Name:	VVs
Author:	Gareth Phillips
Date:	15/06/2020
Version:	1.0

Use Case Name:	Search Student Mark	Use Case Type:
Use Case ID:	4.6	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	School Administrator	
Primary System Actor	None	
Other Participating Actors:	None	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process of an Administrator searching a Student Mark on the system. The Use Case starts when the Administrator navigates by selecting the "Student Mark Management" option under the "Student Management" Combo box in the Navigation bar. The system populates the table with the information of all the Student Mark.	
Pre-Condition:	The Administrator must be logged into the system.	
Trigger:	The Administrator requests to search for a Student Mark on the system.	

Typical Course Of Events:	Actor Action	System Response																	
		Manual Action	Automated Action																
	Step 1: The Administrator selects the "Student Mark Management" option under the "Student Management" Combo box in the Navigation bar.		Step 2: The system Displays the "Student Mark Management" screen and populates it with the following details:																
			<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Search</td><td>Text-Box</td><td>Used for capturing Search criteria for the specific Student Mark.</td></tr><tr><td>Search</td><td>Button</td><td>Used to initiate a search.</td></tr><tr><td>Student Mark Management</td><td>Table</td><td>Displays a list of Student Marks.</td></tr><tr><td>Capture Mark</td><td>Button</td><td>Used to navigate to the specific Student Mark view.</td></tr></table>		Text/Label	Control Type	Additional Info	Search	Text-Box	Used for capturing Search criteria for the specific Student Mark.	Search	Button	Used to initiate a search.	Student Mark Management	Table	Displays a list of Student Marks.	Capture Mark	Button	Used to navigate to the specific Student Mark view.
			Text/Label	Control Type	Additional Info														
			Search	Text-Box	Used for capturing Search criteria for the specific Student Mark.														
			Search	Button	Used to initiate a search.														
			Student Mark Management	Table	Displays a list of Student Marks.														
			Capture Mark	Button	Used to navigate to the specific Student Mark view.														
			The table is populated by using a FIRESQL READ query with data from the following:																
			<table><tr><th>Entity</th><th>Attributes</th><th>Input Name</th></tr><tr><td>Student Mark</td><td>Mark</td><td>Mark</td></tr><tr><td>Student</td><td>StudentName</td><td>Name</td></tr></table>		Entity	Attributes	Input Name	Student Mark	Mark	Mark	Student	StudentName	Name						
			Entity	Attributes	Input Name														
Student Mark	Mark	Mark																	
Student	StudentName	Name																	

			Student	StudentSurname	Surname
	Step 3: The Administrator enters the search criteria that they wish to search by. They do so by typing in a keyword in the "Search" Text box.				
	Step 4: The Administrator clicks the search button.		Step 5: The system uses a FIRESQL READ query to READ the following details in the database: [alt]		
			Entity	Attributes	Input Name
			Student Mark	Mark	Mark
		Student	StudentName	Name	
		Student	StudentSurname	Surname	
		Step 5: The following details are loaded and used to populate the "Student Mark Management" table by using a FIRESQL READ query:			
		Output	Attribute	Entity	
		Mark	Mark	Student Mark	
		Name	StudentName	Student	
		Surname	StudentSurname	Student	
		Step 6: The system creates and saves the following in the database by using a FIRESQL INSERT query:			
		AuditTrail Table			
		AuditTrailID			

			DateTime TransactionDescription UserID									
Alternate Courses:	Alt Step 4: In the case of invalid input being entered, the system notifies the Administrator of the Format error below the input field with the following: <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>"Invalid input."</td><td>Label (with a light red background color fill)</td><td>Used to inform the Administrator of the result.</td></tr><tr><td>"X"</td><td>Icon</td><td>Used to indicate an error result.</td></tr></table> Redirect to Step 3.			Text/Label	Control Type	Additional Info	"Invalid input."	Label (with a light red background color fill)	Used to inform the Administrator of the result.	"X"	Icon	Used to indicate an error result.
Text/Label	Control Type	Additional Info										
"Invalid input."	Label (with a light red background color fill)	Used to inform the Administrator of the result.										
"X"	Icon	Used to indicate an error result.										
Conclusion:	The use case narrative concludes when the searched Student Mark details are displayed.											
Post-Condition:	- The Student Mark has been searched successfully. - The AuditTrail table is successfully updated.											
Business Rules	Only an Administrator can search for a Student Mark on the system.											
Implementation Constraints and Specifications:	None											
Assumptions:	None											
Open Issues:	None											

2.3.5.7. Add Mark Type

System Name:	VVs
Author:	Gareth Phillips
Date:	15/06/2020
Version:	1.0

Use Case Name:	Add Mark Type	Use Case Type:
Use Case ID:	4.7	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	Administrator	
Primary System Actor	None	
Other Participating Actors:	None	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process of an Administrator adding a new Mark Type to the system. The Use Case starts when the Administrator navigates by selecting the "Mark Type Management" option under the "System Management" Combo box in the Navigation bar. From there, they click the "Add Mark Type" button and the system will load the "Mark Type" screen. The system populates the corresponding labels, text boxes and buttons, icons, logos, and banners.	
Pre-Condition:	The Administrator must be logged into the system.	

Trigger:	The Administrator requests to add a new Mark Type to the system.				
Typical Course Of Events:	Actor Action	System Response			
		Manual Action	Automated Action		
	Step 1: The Administrator selects the “Mark Type Management” option under the “Student Management” Combo box in the Navigation bar.		Step 2: The system Displays the “Mark Type Management” screen and populates it with the following details:		
			Text/Label	Control Type	Additional Info
			Add Mark Type	Button	Used to load the “Mark Type” screen.
			Search	Text-Box	Used for capturing Search criteria for the specific Mark Type.
			Mark Type Management	Table	Displays a list of Mark Types.
			“Pen icon”	Button	Used to navigate to the specific Mark Type view.
			The table is populated by using a FIRESQL READ query with data from the following:		
			Entity	Attributes	Input Name
MarkType			MarkType	Mark Type	
MarkType	Weighting	Weighting			

	Step 3: The Administrator clicks on the "Add Mark Type" button.		Step 4: The system will load the "Mark Type" screen and displays it with the following controls and layout:		
			Text/Label	Control Type	Additional Info
			Mark Type	Label	Heading for the pop-up displayed.
			Mark Type	Label	Used to indicate what the corresponding text box is used for.
			Mark Type	Text box	Used to capture text for the Mark Type details.
			Weighting	Label	Used to indicate what the corresponding text box is used for.
			Weighting	Text box	Used to capture text for the Weighting details.
			Create	Button	Used to Save the new Mark Type Details.
			Cancel	Button	Used to cancel the action of

			adding a Mark Type.								
	Step 5: Administrator inputs the following Mark Type details: - Mark Type - Weighting										
	Step 6: The Administrator clicks the "Create" button. [alt]		Step 7: The system captures and ensures that all of the fields are filled in.								
			Step 8: The system validates that the captured details are in the correct format: [alt] MarkType table <table><tr><th>Attribute</th><th>Validation</th></tr><tr><td>MarkType</td><td> - Max 25 characters </td></tr><tr><td>Weighting</td><td> - Numeric characters and "%" only </td></tr></table>	Attribute	Validation	MarkType	Max 25 characters	Weighting	Numeric characters and "%" only		
	Attribute	Validation									
	MarkType	Max 25 characters									
Weighting	Numeric characters and "%" only										
		Step 9: The system verifies that no duplicate Mark Type exists on the system by comparing the following: [alt] <table><tr><th>Input</th><th>Attribute</th><th>Entity</th></tr><tr><td>Mark Type</td><td>MarkType</td><td>Mark Type</td></tr><tr><td>Weighting</td><td>Weighting</td><td>Mark Type</td></tr></table>	Input	Attribute	Entity	Mark Type	MarkType	Mark Type	Weighting	Weighting	Mark Type
Input	Attribute	Entity									
Mark Type	MarkType	Mark Type									
Weighting	Weighting	Mark Type									
		Step 10: The system prompts the Administrator to confirm to add the new Mark Type with the following: <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Confirmation</td><td>Dialog Box Title</td><td>Used to indicate what type of</td></tr></table>	Text/Label	Control Type	Additional Info	Confirmation	Dialog Box Title	Used to indicate what type of			
Text/Label	Control Type	Additional Info									
Confirmation	Dialog Box Title	Used to indicate what type of									

			message is being displayed.					
			<table><tr><td>"Are you sure you want to add this Mark Type?"</td><td>Dialog Box Message</td><td>Used to inform the Administrator of the action.</td></tr></table>	"Are you sure you want to add this Mark Type?"	Dialog Box Message	Used to inform the Administrator of the action.		
	"Are you sure you want to add this Mark Type?"	Dialog Box Message	Used to inform the Administrator of the action.					
			<table><tr><td>Yes</td><td>Button</td><td>Used to confirm the action.</td></tr></table>	Yes	Button	Used to confirm the action.		
	Yes	Button	Used to confirm the action.					
		<table><tr><td>No</td><td>Button</td><td>Used to decline the action.</td></tr></table>	No	Button	Used to decline the action.			
No	Button	Used to decline the action.						
Step 11: The Administrator clicks the "Yes" button. [alt]		Step 12: The Mark Type is assigned a unique MarkTypeID in the Mark Type table which is set to the next consecutive number after the previous MarkType in the database, the system then creates and saves the new Fault Task in the MarkType table. The following attributes are saved in the database by using a FIRESQL INSERT query: Mark Type Table MarkTypeID MarkType Weighting						
		Step 13: The system notifies the Administrator that the Mark Type has been added successfully to the database with the following: [alt] <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Success</td><td>Dialog Box Title</td><td>Used to indicate what type of</td></tr></table>	Text/Label	Control Type	Additional Info	Success	Dialog Box Title	Used to indicate what type of
Text/Label	Control Type	Additional Info						
Success	Dialog Box Title	Used to indicate what type of						

			message is being displayed.		
			"Mark Type has been added successfully"	Dialog Box Message	Used to inform the Administrator of the result.
			Ok	Button	Used to accept the result.
			Step 14: The system creates and saves the following in the database by using a FIRESQL INSERT query:		
			AuditTrail Table		
			AuditTrailID		
			DateTime		
			TransactionDescription		
UserID					
Alternate Courses:	Alt Step 6: The Administrator clicks the "Cancel" button. Terminate Use Case.				
	Alt Step 7: In the case of an empty Text box, the system notifies the Administrator of the Input error below the input field with the following:				
	Text/Label	Control Type	Additional Info		
	"Please fill this field."	Label (with a light red background color fill)	Used to inform the Administrator of the result.		
	"X"	Icon	Used to indicate an error result.		
Redirect to Step 5.					

Alt Step 8: In the case of invalid input being entered, the system notifies the Administrator of the Format error below the input field with the following:

Text/Label	Control Type	Additional Info
"Invalid input."	Label (with a light red background color fill)	Used to inform the Administrator of the result.
"X"	Icon	Used to indicate an error result.

Redirect to Step 5.

Alt Step 9: The inputted Mark Type already exists in the system; the system will notify the Administrator of the duplication error with the following:

Text/Label	Control Type	Additional Info
Error	Dialog Box Title	Used to indicate what type of message is being displayed.
"The Mark Type already exists. Please try again."	Dialog Box Message	Used to inform the Administrator of the result.
"Help icon"	Button	Used to Load the Help document.
Ok	Button	Used to accept the result.

Redirect to Step 5.

Alt Step 11: The Administrator declines to add the new Mark Type by clicking the "No" button. Redirect to Step 5.

	Alt Step 13: The Mark Type was not added successfully to the database, the system will notify the Administrator of the save error with the following:		
	Text/Label	Control Type	Additional Info
	Failed	Dialog Box Title	Used to indicate what type of message is being displayed
	"Create unsuccessful. Please try again."	Dialog Box Message	Used to inform the Administrator of the result.
	"Help icon"	Button	Used to Load the Help document.
	Ok	Button	Used to accept the result.
	Redirect to Step 5.		
Conclusion:	The use case concludes when the Administrator has been notified that the Mark Type has been added successfully.		
Post-Condition:	- A new Mark Type is added to the system with the relevant details. - The AuditTrail table is successfully updated. - The MarkType table is successfully updated.		
Business Rules	Only an Administrator can add a Mark Type to the system.		
Implementation Constraints and Specifications:	None		
Assumptions:	None		

Open Issues:	• None

2.3.5.8. Search Mark Type

System Name:	VVs
Author:	Gareth Phillips
Date:	15/06/2020
Version:	1.0

Use Case Name:	Search Mark Type	Use Case Type:
Use Case ID:	4.8	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	School Administrator	
Primary System Actor	None	
Other Participating Actors:	None	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process of an Administrator searching a Mark Type on the system. The Use Case starts when the Administrator navigates by selecting the " Mark Type Management" option under the "System Management" Combo box in the Navigation bar. The system populates the table with the information of all the Mark Type.	
Pre-Condition:	The Administrator must be logged into the system.	
Trigger:	The Administrator requests to search for a Mark Type on the system.	

Typical Course Of Events:	Actor Action	System Response			
		Manual Action	Automated Action		
	Step 1: The Administrator selects the “Mark Type Management” option under the “System Management” Combo box in the Navigation bar.		Step 2: The system Displays the “Mark Type Management” screen and populates it with the following details:		
			Text/Label	Control Type	Additional Info
			Add Mark Type	Button	Used to load the “Mark Type” screen.
			Search	Text-Box	Used for capturing Search criteria for the specific Mark Type.
			Mark Type Management	Table	Displays a list of Mark Types.
			“Pen icon”	Button	Used to navigate to the specific Mark Type view.
			“Back”	Button	Returns to Home Page.
			“Help Icon”	Button	Used to Load the Help document.
The table is populated by using a FIRESQL READ query with data from the following:					
Entity	Attributes	Input Name			

			Mark Type	MarkType	Mark Type												
	Step 3: The Administrator enters the search criteria that they wish to search by. They do so by typing in a keyword in the "Search" Text box.		Step 4: The system uses a FIRESQL READ query to READ the following details in the database: [alt]														
			<table><tr><td>Entity</td><td>Attributes</td></tr><tr><td>Mark Type</td><td>Mark Type</td></tr></table>			Entity	Attributes	Mark Type	Mark Type								
	Entity	Attributes															
	Mark Type	Mark Type															
		Step 5: The following details are loaded and used to populate the "Mark Type Management" table by using a FIRESQL READ query:															
		<table><tr><td>Output</td><td>Attribute</td><td>Entity</td></tr><tr><td>Mark Type</td><td>MarkType</td><td>Mark Type</td></tr><tr><td>Weighting</td><td>Weighting</td><td>Mark Type</td></tr></table>			Output	Attribute	Entity	Mark Type	MarkType	Mark Type	Weighting	Weighting	Mark Type				
Output	Attribute	Entity															
Mark Type	MarkType	Mark Type															
Weighting	Weighting	Mark Type															
	Step 6: The Administrator clicks on the "Pen icon" Maintenance Task Button in the "Action" table column.		Step 7: The System will load the "Mark Type" screen and displays it with the following controls and layout:														
			<table><tr><td>Text/Label</td><td>Control Type</td><td>Additional Info</td></tr><tr><td>Mark Type</td><td>Label</td><td>Heading for the pop-up displayed.</td></tr><tr><td>Mark Type</td><td>Label</td><td>Used to indicate what the corresponding text box is used for.</td></tr><tr><td>Weighting</td><td>Label</td><td>Used to indicate what the corresponding</td></tr></table>			Text/Label	Control Type	Additional Info	Mark Type	Label	Heading for the pop-up displayed.	Mark Type	Label	Used to indicate what the corresponding text box is used for.	Weighting	Label	Used to indicate what the corresponding
Text/Label	Control Type	Additional Info															
Mark Type	Label	Heading for the pop-up displayed.															
Mark Type	Label	Used to indicate what the corresponding text box is used for.															
Weighting	Label	Used to indicate what the corresponding															

			text box is used for.		
			Mark Type	Text box	Used to display text for the Mark Type details.
			Weighting	Text box	Used to capture text for the Mark Type details.
			Delete	Button	Used to Delete the Mark Type Details.
			Update	Button	Used to Update the Mark Type Details.
			The following fields are populated with data retrieved from the database where the MarkTypeID is the same as the one searched by using a FIRESQL READ query:		
			Output	Attribute	Entity
			Weighting	Weighting	Mark Type
			Mark Type	MarkType	Mark Type
			Step 8: The system creates and saves the following in the database by using a FIRESQL INSERT query:		
			AuditTrail Table AuditTrailID DateTime		

			TransactionDescription UserID									
Alternate Courses:	Alt Step 4: The specific Mark Type does not exist in the database. The system will notify the Administrator of this with the following: <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>"Mark Type not found."</td><td>Label</td><td>Used to inform the Administrator of the result.</td></tr><tr><td>"X"</td><td>Banner (with a light red background color fill)</td><td>Used to display the result message.</td></tr></table> Redirect to Step 3.			Text/Label	Control Type	Additional Info	"Mark Type not found."	Label	Used to inform the Administrator of the result.	"X"	Banner (with a light red background color fill)	Used to display the result message.
Text/Label	Control Type	Additional Info										
"Mark Type not found."	Label	Used to inform the Administrator of the result.										
"X"	Banner (with a light red background color fill)	Used to display the result message.										
Conclusion:	The use case narrative concludes when the searched Mark Type details are displayed.											
Post-Condition:	- The Mark Type has been searched successfully. - The AuditTrail table is successfully updated.											
Business Rules	Only an Administrator can search for a Mark Type on the system.											
Implementation Constraints and Specifications:	None											
Assumptions:	None											
Open Issues:	None											

2.3.5.9. Update Mark Type

System Name:	VVs
Author:	Gareth Phillips
Date:	15/06/2020
Version:	1.0

Use Case Name:	Update Mark Type	Use Case Type:
Use Case ID:	4.9	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	School Administrator	
Primary System Actor	None	
Other Participating Actors:	None	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process of an Administrator updating an existing Maintenance Task's details in the system. The Use Case starts when the Administrator navigates by selecting the "Maintenance Task Management" option under the "System Management" Combo box in the Navigation bar. From there, they search for the Maintenance Task they want to update, the system then will load the "Maintenance Task" screen with the Maintenance Task details. The Administrator can then make their changes and save it.	
Pre-Condition:	The Administrator must be logged into the system.	

Trigger:	The Administrator requests to update an existing Maintenance Task to the system.								
Typical Course Of Events:	Actor Action	System Response							
		Manual Action	Automated Action						
	Step 1: The Administrator selects the "Maintenance Task Management" option under the "System Management" Combo box in the Navigation bar.		Step 2: The System invokes Use Case 4.8 "Search Mark Type" and displays the "Mark Type" screen with the data of the searched Mark Type.						
	Step 3: Administrator updates the following Maintenance Task details: Mark Type								
	Step 4: The Administrator clicks the "Update Mark Type" button. [alt]		Step 5: The system captures and ensures that all of the fields are filled in. [alt]						
			Step 6: The system validates that the captured details are in the correct format: [alt] Mark Type <i>table</i> <table><tr><th>Attribute</th><th>Validation</th></tr><tr><td>MarkType</td><td> - Max 25 characters </td></tr><tr><td>Weighting</td><td> - Numeric characters and "%" only </td></tr></table>	Attribute	Validation	MarkType	Max 25 characters	Weighting	Numeric characters and "%" only
	Attribute	Validation							
	MarkType	Max 25 characters							
Weighting	Numeric characters and "%" only								
		Step 7: The system prompts the Administrator to confirm the update of the Mark Type details. <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Confirmation</td><td>Dialog Box Title</td><td>Used to indicate what type of</td></tr></table>	Text/Label	Control Type	Additional Info	Confirmation	Dialog Box Title	Used to indicate what type of	
Text/Label	Control Type	Additional Info							
Confirmation	Dialog Box Title	Used to indicate what type of							

			message is being displayed.		
			“Are you sure you want to update this Mark Type?”	Dialog Box Message	Used to inform the Administrator of the action.
			Yes	Button	Used to confirm the action.
			No	Button	Used to decline the action.
	Step 8: The Administrator clicks the “Yes” button. [alt]		Step 9: The system then updates and saves the updated Mark Type details in the MarkType. The following attributes are updated in the database by using a FIRESQL UPDATE query: Mark Type Table MarkTypeID Weighting MarkType		
			Step 10: The system notifies the Administrator that the Mark Type has been updated successfully in the database with the following: [alt]		
			Text/Label	Control Type	Additional Info
			Success	Dialog Box Title	Used to indicate what type of message is being displayed.

			"The Mark Type has been updated successfully" Dialog Box Message Used to inform the Administrator of the result.
			Ok Button Used to accept the result.
			Step 11: The system updates and saves the following in the database by using a FIRESQL INSERT query: AuditTrail Table AuditTrailID DateTime TransactionDescription UserID
Alternate Courses:	Alt Step 4: The Administrator clicks the "Cancel" button. Terminate Use Case.		
	Alt Step 5a: In the case of an empty Text box, the system notifies the Administrator of the Input error below the input field with the following:		
	Text/Label "Please fill this field."	Control Type Label (with a light red background color fill)	Additional Info Used to inform the Administrator of the result.
	"X"	Icon	Used to indicate an error result.
	Redirect to Step 3.		

Alt Step 6: In the case of invalid input being entered, the system notifies the Administrator of the Format error below the input field with the following:

Text/Label	Control Type	Additional Info
"Invalid input."	Label (with a light red background color fill)	Used to inform the Administrator of the result.
"X"	Icon	Used to indicate an error result.

Redirect to Step 3.

Alt Step 8: The Administrator declines to update the Mark Type by clicking the "No" button. Redirect to Step 4.

Alt Step 10: The Mark Type was not updated successfully in the database; the system will notify the Administrator of the update error with the following:

Text/Label	Control Type	Additional Info
Failed	Dialog Box Title	Used to indicate what type of message is being displayed
"Update unsuccessful. Please try again."	Dialog Box Message	Used to inform the Administrator of the result.
"Help icon"	Button	Used to Load the Help document.
Ok	Button	Used to accept the result.

Redirect to Step 3.

Conclusion:	The use case concludes when the Administrator has been notified that the Mark Type has been updated successfully.
Post-Condition:	• An existing Mark Type is updated in the system with the relevant details. • The AuditTrail table is successfully updated. • The MarkType table is successfully updated.
Business Rules	• Only an Administrator can update a Mark Type to the system.
Implementation Constraints and Specifications:	• None
Assumptions:	• None
Open Issues:	• None

2.3.5.10. Delete Mark Type

System Name:	VVs
Author:	Gareth Phillips
Date:	15/06/2020
Version:	1.0

Use Case Name:	Delete Mark Type	Use Case Type:
Use Case ID:	4.10	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	School Administrator	
Primary System Actor	None	
Other Participating Actors:	None	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process of an Administrator deleting an existing Mark Type details from the system. The Use Case starts when the Administrator navigates by selecting the " Mark Type Management" option under the "System Management" Combo box in the Navigation bar. From there, they search for the Mark Type they want to delete, and the system displays the "Mark Type" screen with the Mark Type details to be deleted. The system populates the relevant Labels and corresponding text boxes with information of the searched Maintenance Task, and the Administrator can make their deletion.	
Pre-Condition:	The Administrator must be logged into the system.	

Trigger:	The Administrator requests to delete an existing Mark Type from the system.																
Typical Course Of Events:	Actor Action	System Response															
		Manual Action	Automated Action														
	Step 1: The Administrator selects the “Mark Type Management” option under the “System Management” Combo box in the Navigation bar.		Step 2: The System invokes Use Case 4.8 “Search Mark Type” and displays the “Mark Type” screen with the data of the searched Mark Type.														
	Step 3: The Administrator clicks on the “Delete Mark Type” button. [alt]		Step 4: The system prompts the Administrator to confirm the deletion of the Mark Type details with the following:														
			<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Confirmation</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed.</td></tr><tr><td>“Are you sure you want to delete this Mark Type?”</td><td>Dialog Box Message</td><td>Used to inform the Administrator of the action.</td></tr><tr><td>Yes</td><td>Button</td><td>Used to confirm the action.</td></tr><tr><td>No</td><td>Button</td><td>Used to decline the action.</td></tr></table>	Text/Label	Control Type	Additional Info	Confirmation	Dialog Box Title	Used to indicate what type of message is being displayed.	“Are you sure you want to delete this Mark Type?”	Dialog Box Message	Used to inform the Administrator of the action.	Yes	Button	Used to confirm the action.	No	Button
Text/Label			Control Type	Additional Info													
Confirmation			Dialog Box Title	Used to indicate what type of message is being displayed.													
“Are you sure you want to delete this Mark Type?”	Dialog Box Message	Used to inform the Administrator of the action.															
Yes	Button	Used to confirm the action.															
No	Button	Used to decline the action.															

	Step 5: The Administrator clicks the "Yes" button. [alt]		Step 6: The system then removes the Mark Type details in the MarkType table. The following attributes are deleted in the database by using a FIRESQL DELETE query: Mark Type Table MarkTypeID MarkType Weighting												
			Step 7: The system notifies the Administrator that the Mark Type has been deleted successfully from the database with the following: [alt] <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Success</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed.</td></tr><tr><td>"The Mark Type has been deleted successfully"</td><td>Dialog Box Message</td><td>Used to inform the Administrator of the result.</td></tr><tr><td>Ok</td><td>Button</td><td>Used to accept the result.</td></tr></table>	Text/Label	Control Type	Additional Info	Success	Dialog Box Title	Used to indicate what type of message is being displayed.	"The Mark Type has been deleted successfully"	Dialog Box Message	Used to inform the Administrator of the result.	Ok	Button	Used to accept the result.
	Text/Label	Control Type	Additional Info												
Success	Dialog Box Title	Used to indicate what type of message is being displayed.													
"The Mark Type has been deleted successfully"	Dialog Box Message	Used to inform the Administrator of the result.													
Ok	Button	Used to accept the result.													
		Step 8: The system updates and saves the following in the database by using a FIRESQL INSERT query: AuditTrail Table AuditTrailID													

			DateTime TransactionDescription UserID
Alternate Courses:	Alt Step 3: The Administrator clicks the "Cancel" button. Terminate Use Case.		
	Alt Step 5: The Administrator declines to delete the Mark Type by clicking the "No" button. Redirect to Step 2.		
	Alt Step 7: The Administrator is notified of the unsuccessful delete action with the following:		
	Text/Label	Control Type	Additional Info
	Failed	Dialog Box Title	Used to indicate what type of message is being displayed
	"Delete unsuccessful. Please try again."	Dialog Box Message	Used to inform the Administrator of the result.
	"Help icon"	Button	Used to Load the Help document.
	Ok	Button	Used to accept the result.
	Redirect to Step 2.		
Conclusion:	The use case concludes when the Administrator has been notified that the Mark Type has been successfully deleted.		
Post-Condition:	- An existing Mark Type is deleted from the system with all the relevant details being removed. - The AuditTrail table is successfully updated. - The MarkType Task table is successfully updated with the deleted details being removed.		

Business Rules	• Only an Administrator can delete a Mark Type from the system.
Implementation Constraints and Specifications:	• None
Assumptions:	• None
Open Issues:	• None

2.3.6. Sub-system 5: Bicycle

2.3.6.1. Add Bicycle

System Name:	VVs
Author:	Cameron King
Date:	15/06/2020
Version:	1.0

Use Case Name:	Add Bicycle	Use Case Type:
Use Case ID:	5.1	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	School Administrator	
Primary System Actor	None	
Other Participating Actors:	None	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process of an Administrator adding a new Bicycle to the system. The Use Case starts when the Administrator navigates by selecting the "Bicycle Management" option under the "Bicycle Management" Combo box in the Navigation bar. From there, they click the "Add Bicycle" button and the system will load the "Bicycle" screen. The system populates the relevant Combo boxes with information necessary for selection, and the corresponding labels, text boxes and buttons, icons, logos, and banners.	

Pre-Condition:	The Administrator must be logged into the system.		
Trigger:	The Administrator requests to add a new Bicycle to the system.		
Typical Course Of Events:	Actor Action	System Response	
		Manual Action	Automated Action
	Step 1: The Administrator selects the “Bicycle Management” option and under the “Bicycle Management” Combo box in the Navigation bar.		Step 2: The system Displays the “Bicycle Management” screen and populates it with the following details:
The table is populated by using a FIRESQL READ query with data from the following:			

			<table><tr><th>Entity</th><th>Attributes</th><th>Input Name</th></tr><tr><td>Bicycle</td><td>BicycleCode</td><td>Bicycle Code</td></tr><tr><td>Bicycle</td><td>BicycleMake</td><td>BicycleMake</td></tr><tr><td>BicycleStatus</td><td>BicycleStatus</td><td>Bicycle Status</td></tr><tr><td colspan="3">The FIRESQL READ query will display the BicycleStatus in the BicycleStatus table that corresponds with the BicycleStatusID in the Bicycle and BicycleStatus tables.</td></tr></table>	Entity	Attributes	Input Name	Bicycle	BicycleCode	Bicycle Code	Bicycle	BicycleMake	BicycleMake	BicycleStatus	BicycleStatus	Bicycle Status	The FIRESQL READ query will display the BicycleStatus in the BicycleStatus table that corresponds with the BicycleStatusID in the Bicycle and BicycleStatus tables.								
	Entity	Attributes	Input Name																					
	Bicycle	BicycleCode	Bicycle Code																					
	Bicycle	BicycleMake	BicycleMake																					
	BicycleStatus	BicycleStatus	Bicycle Status																					
	The FIRESQL READ query will display the BicycleStatus in the BicycleStatus table that corresponds with the BicycleStatusID in the Bicycle and BicycleStatus tables.																							
	Step 3: The Administrator clicks on the "Add" button.		<table><tr><td colspan="3">Step 4: The system will load the "Bicycle" screen and displays it with the following controls and layout:</td></tr><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Menu</td><td>Menu-icon</td><td>Used to Navigate the side menu-bar.</td></tr><tr><td>Bicycle</td><td>Label</td><td>Heading for the pop-up displayed.</td></tr><tr><td>Bicycle Make</td><td>Text Box</td><td>Used to indicate what the corresponding text box is used for.</td></tr><tr><td>Create</td><td>Button</td><td>Used to Save the new Bicycle Details.</td></tr><tr><td>Cancel</td><td>Button</td><td>Used to cancel the action of adding a Bicycle.</td></tr></table>	Step 4: The system will load the "Bicycle" screen and displays it with the following controls and layout:			Text/Label	Control Type	Additional Info	Menu	Menu-icon	Used to Navigate the side menu-bar.	Bicycle	Label	Heading for the pop-up displayed.	Bicycle Make	Text Box	Used to indicate what the corresponding text box is used for.	Create	Button	Used to Save the new Bicycle Details.	Cancel	Button	Used to cancel the action of adding a Bicycle.
	Step 4: The system will load the "Bicycle" screen and displays it with the following controls and layout:																							
	Text/Label	Control Type	Additional Info																					
	Menu	Menu-icon	Used to Navigate the side menu-bar.																					
Bicycle	Label	Heading for the pop-up displayed.																						
Bicycle Make	Text Box	Used to indicate what the corresponding text box is used for.																						
Create	Button	Used to Save the new Bicycle Details.																						
Cancel	Button	Used to cancel the action of adding a Bicycle.																						

	Step 5: Administrator inputs the following Bicycle details: - Bicycle Make - Bicycle Code								
	Step 6: The Administrator clicks the "Create" button. [alt]		Step 7: The system captures and ensures that the field is filled. [alt]						
			Step 8: The system validates that the captured details are in the correct format: [alt] Bicycle table <table><tr><th>Attribute</th><th>Validation</th></tr><tr><td>Bicycle Make</td><td>28.Max 50 characters</td></tr><tr><td>Date Added</td><td> - Max 10 characters - Year attribute must be no earlier than 2015. - Format must follow DD/MM/YYYY 29.Numeric characters and "/" or "-" only.</td></tr></table>	Attribute	Validation	Bicycle Make	28.Max 50 characters	Date Added	- Max 10 characters - Year attribute must be no earlier than 2015. - Format must follow DD/MM/YYYY 29.Numeric characters and "/" or "-" only.
	Attribute	Validation							
	Bicycle Make	28.Max 50 characters							
Date Added	- Max 10 characters - Year attribute must be no earlier than 2015. - Format must follow DD/MM/YYYY 29.Numeric characters and "/" or "-" only.								
		Step 9: The system verifies that no duplicate Part exists on the system by comparing the following: [alt] <table><tr><th>Input</th><th>Attribute</th><th>Entity</th></tr><tr><td>Bicycle Code</td><td>BicycleCode</td><td>BicycleCode</td></tr></table>	Input	Attribute	Entity	Bicycle Code	BicycleCode	BicycleCode	
Input	Attribute	Entity							
Bicycle Code	BicycleCode	BicycleCode							
		Step 10: The system prompts the Administrator to confirm to add the new Bicycle with the following: <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Confirmation</td><td>Dialog Box Title</td><td>Used to indicate what type of</td></tr></table>	Text/Label	Control Type	Additional Info	Confirmation	Dialog Box Title	Used to indicate what type of	
Text/Label	Control Type	Additional Info							
Confirmation	Dialog Box Title	Used to indicate what type of							

			message is being displayed.									
			<table><tr><td>"Are you sure you want to add this Bicycle?"</td><td>Dialog Box Message</td><td>Used to inform the Administrator of the action.</td></tr><tr><td>Yes</td><td>Button</td><td>Used to confirm the action.</td></tr><tr><td>No</td><td>Button</td><td>Used to decline the action.</td></tr></table>	"Are you sure you want to add this Bicycle?"	Dialog Box Message	Used to inform the Administrator of the action.	Yes	Button	Used to confirm the action.	No	Button	Used to decline the action.
	"Are you sure you want to add this Bicycle?"	Dialog Box Message	Used to inform the Administrator of the action.									
	Yes	Button	Used to confirm the action.									
	No	Button	Used to decline the action.									
Step 11: The Administrator clicks the "Yes" button. [alt]		Step 12: The Bicycle is assigned a unique BicycleID in the Bicycle table which is set to the next consecutive number after the previous BicycleID in the database, the system then creates and saves the new Bicycle in the Bicycle table. The BicycleStatusID in the Bicycle table is saved with the same BicycleStatusID from the BicycleStatus table where the BicycleStatus is 'Unavailable'. The following attributes are saved in the database by using a FIRESQL INSERT query: <table><tr><td>Bicycle Table</td></tr><tr><td>BicycleID</td></tr><tr><td>BicycleStatusID</td></tr><tr><td>BicycleMake</td></tr><tr><td>DateAdded</td></tr></table>		Bicycle Table	BicycleID	BicycleStatusID	BicycleMake	DateAdded				
Bicycle Table												
BicycleID												
BicycleStatusID												
BicycleMake												
DateAdded												
		Step 13: The system notifies the Administrator that the Tag has been added successfully to the database with the following: [alt] <table><tr><td>Text/Label</td><td>Control Type</td><td>Additional Info</td></tr><tr><td>Success</td><td>Dialog Box Title</td><td>Used to indicate what type of</td></tr></table>		Text/Label	Control Type	Additional Info	Success	Dialog Box Title	Used to indicate what type of			
Text/Label	Control Type	Additional Info										
Success	Dialog Box Title	Used to indicate what type of										

			message is being displayed.
		"Bicycle has been added successfully"	Dialog Box Message Used to inform the Administrator of the result.
		Ok	Button Used to accept the result.
			Step 14: The system creates and saves the following in the database by using a FIRESQL INSERT query: AuditTrail Table AuditTrailID DateTime TransactionDescription UserID
Alternate Courses:	Alt Step 6: The Administrator clicks the "Cancel" button. Terminate Use Case.		
	Alt Step 7a: In the case of an empty Text box, the system notifies the Administrator of the Input error below the input field with the following:		

Text/Label	Control Type	Additional Info
"Please fill this field."	Label (with a light red background color fill)	Used to inform the Administrator of the result.
"X"	Icon	Used to indicate an error result.

Redirect to Step 5.

Alt Step 8: In the case of invalid input being entered, the system notifies the Administrator of the Format error below the input field with the following:

Text/Label	Control Type	Additional Info
"Invalid input."	Label (with a light red background color fill)	Used to inform the Administrator of the result.
"X"	Icon	Used to indicate an error result.

Redirect to Step 5.

Alt Step 9: The inputted Part already exists in the system; the system will notify the Administrator of the duplication error with the following:

Text/Label	Control Type	Additional Info
Error	Dialog Box Title	Used to indicate what type of message is being displayed.
"The Bicycle already exists. Please try again."	Dialog Box Message	Used to inform the Administrator of the result.
"Help icon"	Button	Used to Load the Help document.
Ok	Button	Used to accept the result.

Redirect to Step 5.

Alt Step 11: The Administrator declines to add the new Bicycle by clicking the "No" button. Redirect to Step 5.

	Alt Step 13: The Bicycle was not added successfully to the database, the system will notify the Administrator of the save error with the following:		
	Text/Label	Control Type	Additional Info
	Failed	Dialog Box Title	Used to indicate what type of message is being displayed
	"Create unsuccessful. Please try again."	Dialog Box Message	Used to inform the Administrator of the result.
	"Help icon"	Button	Used to Load the Help document.
	Ok	Button	Used to accept the result.
	Redirect to Step 5.		
Conclusion:	The use case concludes when the Administrator has been notified that the Bicycle has been added successfully.		
Post-Condition:	- A new Bicycle is added to the system with the relevant details. - The AuditTrail table is successfully updated. - The Bicycle table is successfully updated.		
Business Rules	Only an Administrator can add a Bicycle to the system.		
Implementation Constraints and Specifications:	None		
Assumptions:	None		
Open Issues:	None		

2.3.6.2. Search Bicycle

System Name:	VVs
Author:	Cameron King
Date:	15/06/2020
Version:	1.0

Use Case Name:	Search Bicycle	Use Case Type:
Use Case ID:	5.2	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	School Administrator	
Primary System Actor	None	
Other Participating Actors:	None	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process of an Administrator searching a Bicycle on the system. The Use Case starts when the Administrator navigates by selecting the "Bicycle Management" option under the "Bicycle Management" Combo box in the Navigation bar. The system populates the table with the information of all the Bicycles.	
Pre-Condition:	The Administrator must be logged into the system.	
Trigger:	The Administrator requests to search for a Bicycle on the system.	

Typical Course Of Events:	Actor Action	System Response																							
		Manual Action	Automated Action																						
	Step 1: The Administrator selects the "Bicycle Management" option under the "Bicycle Management" Combo box in the Navigation bar.		Step 2: The system Displays the "Bicycle Management" screen and populates it with the following details:																						
			<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Navigation</td><td>Navigation-bar</td><td>Used to Navigate the side site.</td></tr><tr><td>Add</td><td>Button</td><td>Used to load the "Bicycle" screen.</td></tr><tr><td>Search</td><td>Button</td><td>Used to initiate the search.</td></tr><tr><td>Search</td><td>Text-Box</td><td>Used for capturing Search criteria for the specific Bicycle.</td></tr><tr><td>Bicycle Management</td><td>Table</td><td>Displays a list of Bicycles.</td></tr><tr><td>"Pen icon"</td><td>Button</td><td>Used to navigate to the specific Bicycle view.</td></tr></table>		Text/Label	Control Type	Additional Info	Navigation	Navigation-bar	Used to Navigate the side site.	Add	Button	Used to load the "Bicycle" screen.	Search	Button	Used to initiate the search.	Search	Text-Box	Used for capturing Search criteria for the specific Bicycle.	Bicycle Management	Table	Displays a list of Bicycles.	"Pen icon"	Button	Used to navigate to the specific Bicycle view.
			Text/Label	Control Type	Additional Info																				
			Navigation	Navigation-bar	Used to Navigate the side site.																				
			Add	Button	Used to load the "Bicycle" screen.																				
			Search	Button	Used to initiate the search.																				
			Search	Text-Box	Used for capturing Search criteria for the specific Bicycle.																				
			Bicycle Management	Table	Displays a list of Bicycles.																				
			"Pen icon"	Button	Used to navigate to the specific Bicycle view.																				
			The table is populated with data from the following by using a FIRESQL READ query:																						
			<table><tr><th>Entity</th><th>Attributes</th><th>Input Name</th></tr><tr><td>Bicycle</td><td>BicycleCode</td><td>Bicycle Code</td></tr><tr><td>Bicycle</td><td>BicycleMake</td><td>BicycleMake</td></tr></table>		Entity	Attributes	Input Name	Bicycle	BicycleCode	Bicycle Code	Bicycle	BicycleMake	BicycleMake												
			Entity	Attributes	Input Name																				
Bicycle	BicycleCode	Bicycle Code																							
Bicycle	BicycleMake	BicycleMake																							

			BicycleStatus BicycleStatus Bicycle Status The FIRESQL READ query will display the BicycleStatus in the BicycleStatus table that corresponds with the BicycleStatusID in the Bicycle and BicycleStatus tables.											
	Step 3: The Administrator enters the search criteria that they wish to search by. They do so by typing in a keyword in the "Search" Text box.													
	Step 4: The Administrator clicks the "Search" button.		Step 5: The system uses a FIRESQL READ query to READ the following details in the database: [alt] <table><tr><th>Entity</th><th>Attributes</th><th>Input Name</th></tr><tr><td>Bicycle</td><td>BicycleCode</td><td>Bicycle Code</td></tr><tr><td>BicycleStatus</td><td>BicycleStatus</td><td>Bicycle Status</td></tr></table>	Entity	Attributes	Input Name	Bicycle	BicycleCode	Bicycle Code	BicycleStatus	BicycleStatus	Bicycle Status		
	Entity	Attributes	Input Name											
Bicycle	BicycleCode	Bicycle Code												
BicycleStatus	BicycleStatus	Bicycle Status												
		Step 6: The following details are loaded and used to populate the "Bicycle Management" table by using a FIRESQL READ query: <table><tr><th>Output</th><th>Attribute</th><th>Entity</th></tr><tr><td>Bicycle</td><td>BicycleCode</td><td>Bicycle Code</td></tr><tr><td>Bicycle</td><td>BicycleMake</td><td>BicycleMake</td></tr><tr><td>BicycleStatus</td><td>BicycleStatus</td><td>Bicycle Status</td></tr></table> The FIRESQL READ query will display the BicycleStatus in the BicycleStatus table that corresponds with the BicycleStatusID in the Bicycle and BicycleStatus tables.	Output	Attribute	Entity	Bicycle	BicycleCode	Bicycle Code	Bicycle	BicycleMake	BicycleMake	BicycleStatus	BicycleStatus	Bicycle Status
Output	Attribute	Entity												
Bicycle	BicycleCode	Bicycle Code												
Bicycle	BicycleMake	BicycleMake												
BicycleStatus	BicycleStatus	Bicycle Status												

	Step 7: The Administrator clicks on the “Pen icon” Bicycle Button in the “Action” table column.		Step 8: The System will load the “Bicycle” screen and displays it with the following controls and layout:																					
			<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Navigation</td><td>Navigation-bar</td><td>Used to Navigate the side site.</td></tr><tr><td>Bicycle</td><td>Label</td><td>Heading for the pop-up displayed.</td></tr><tr><td>Bicycle Code</td><td>Label</td><td>Used to indicate what the corresponding read-only text box is used for.</td></tr><tr><td>Bicycle Status</td><td>Label</td><td>Used to indicate what the corresponding read-only text box is used for.</td></tr><tr><td>Bicycle Make</td><td>Label</td><td>Used to indicate what the corresponding text box is used for.</td></tr><tr><td>Date Added</td><td>Label</td><td>Used to indicate what the corresponding date picker is used for.</td></tr></table>	Text/Label	Control Type	Additional Info	Navigation	Navigation-bar	Used to Navigate the side site.	Bicycle	Label	Heading for the pop-up displayed.	Bicycle Code	Label	Used to indicate what the corresponding read-only text box is used for.	Bicycle Status	Label	Used to indicate what the corresponding read-only text box is used for.	Bicycle Make	Label	Used to indicate what the corresponding text box is used for.	Date Added	Label	Used to indicate what the corresponding date picker is used for.
			Text/Label	Control Type	Additional Info																			
			Navigation	Navigation-bar	Used to Navigate the side site.																			
			Bicycle	Label	Heading for the pop-up displayed.																			
			Bicycle Code	Label	Used to indicate what the corresponding read-only text box is used for.																			
			Bicycle Status	Label	Used to indicate what the corresponding read-only text box is used for.																			
			Bicycle Make	Label	Used to indicate what the corresponding text box is used for.																			
Date Added	Label	Used to indicate what the corresponding date picker is used for.																						

			DeleteButtonUsed to Delete the Bicycle details. UpdateButtonUsed to update the Bicycle details. The following fields are populated with data retrieved from the database where the BicycleID is the same as the one searched by using a FIRESQL READ query: <table><tr><th>Output</th><th>Attribute</th><th>Entity</th></tr><tr><td>Bicycle Code</td><td>BicycleCode</td><td>Bicycle</td></tr><tr><td>Bicycle Status</td><td>BicycleStatus</td><td>BicycleStatus</td></tr><tr><td>Bicycle Make</td><td>BicycleMake</td><td>Bicycle</td></tr><tr><td>Date Added</td><td>DateAdded</td><td>Bicycle</td></tr></table> The FIRESQL READ query will display the BicycleStatus in the BicycleStatus table that corresponds with the BicycleStatusID in the Bicycle and BicycleStatus tables.	Output	Attribute	Entity	Bicycle Code	BicycleCode	Bicycle	Bicycle Status	BicycleStatus	BicycleStatus	Bicycle Make	BicycleMake	Bicycle	Date Added	DateAdded	Bicycle
	Output	Attribute	Entity															
Bicycle Code	BicycleCode	Bicycle																
Bicycle Status	BicycleStatus	BicycleStatus																
Bicycle Make	BicycleMake	Bicycle																
Date Added	DateAdded	Bicycle																
			Step 9: The system creates and saves the following in the database by using a FIRESQL INSERT query: AuditTrail Table AuditTrailID DateTime TransactionDescription UserID															

Alternate Courses:	Alt Step 4: The specific Bicycle does not exist in the database. The system will notify the Administrator of this with the following:											
	<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>"Bicycle not found."</td><td>Label</td><td>Used to inform the Administrator of the result.</td></tr><tr><td>"X"</td><td>Banner (with a light red background color fill)</td><td>Used to display the result message.</td></tr></table>			Text/Label	Control Type	Additional Info	"Bicycle not found."	Label	Used to inform the Administrator of the result.	"X"	Banner (with a light red background color fill)	Used to display the result message.
	Text/Label	Control Type	Additional Info									
	"Bicycle not found."	Label	Used to inform the Administrator of the result.									
"X"	Banner (with a light red background color fill)	Used to display the result message.										
Redirect to Step 3.												
Conclusion:	The use case narrative concludes when the searched Bicycle details are displayed.											
Post-Condition:	• The Bicycle has been searched successfully. • The AuditTrail table is successfully updated.											
Business Rules	• Only an Administrator can search for a Bicycle on the system.											
Implementation Constraints and Specifications:	• None											
Assumptions:	• None											
Open Issues:	• None											

2.3.6.3. Update Bicycle

System Name:	VVs
Author:	Cameron King
Date:	15/06/2020
Version:	1.0

Use Case Name:	Update Bicycle	Use Case Type:
Use Case ID:	5.3	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	School Administrator	
Primary System Actor	None	
Other Participating Actors:	None	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process of an Administrator updating an existing Bicycle's details in the system. The Use Case starts when the Administrator navigates by selecting the "Bicycle Management" option under the "Bicycle Management" Combo box in the Navigation bar. From there, they search for the Bicycle they want to update, and the system will load the "Bicycle" screen with the Bicycle details. The system populates the relevant Combo boxes with information of the searched Bicycle, and the Administrator can make their changes and save.	
Pre-Condition:	The Administrator must be logged into the system.	

Trigger:	The Administrator requests to update an existing Bicycle in the system.							
Typical Course Of Events:	Actor Action	System Response						
		Manual Action	Automated Action					
	Step 1: The Administrator selects the "Bicycle Management" option under the "Bicycle Management" Combo box in the Navigation bar.		Step 2: The System invokes Use Case 5.2 "Search Bicycle" and displays the "Bicycle" screen with the data of the searched Bicycle.					
	Step 3: Administrator updates the following Bicycle details: - Date Added - Bicycle Make - Bicycle Status							
	Step 4: The Administrator clicks the "Update" button. [alt]		Step 5: The system captures and ensures that all of the fields are filled in. [alt]					
			Step 6: The system validates that the captured details are in the correct format: [alt] Bicycle table <table><tr><th>Attribute</th><th>Validation</th></tr><tr><td>DateAdded</td><td> - Max 10 characters - Year attribute must be no later than 2015. - Format must follow: DD/MM/YYYY </td></tr><tr><td>Bicycle Make</td><td> - Max 50 characters. </td></tr></table>	Attribute	Validation	DateAdded	- Max 10 characters - Year attribute must be no later than 2015. - Format must follow: DD/MM/YYYY	Bicycle Make
Attribute	Validation							
DateAdded	- Max 10 characters - Year attribute must be no later than 2015. - Format must follow: DD/MM/YYYY							
Bicycle Make	Max 50 characters.							

			Bicycle Status • Combo box option selected															
			Step 7: The system prompts the Administrator to confirm the update of the Bicycle's details. <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Confirmation</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed.</td></tr><tr><td>"Are you sure you want to update this Bicycle?"</td><td>Dialog Box Message</td><td>Used to inform the Administrator of the action.</td></tr><tr><td>Yes</td><td>Button</td><td>Used to confirm the action.</td></tr><tr><td>No</td><td>Button</td><td>Used to decline the action.</td></tr></table>	Text/Label	Control Type	Additional Info	Confirmation	Dialog Box Title	Used to indicate what type of message is being displayed.	"Are you sure you want to update this Bicycle?"	Dialog Box Message	Used to inform the Administrator of the action.	Yes	Button	Used to confirm the action.	No	Button	Used to decline the action.
	Text/Label	Control Type	Additional Info															
	Confirmation	Dialog Box Title	Used to indicate what type of message is being displayed.															
	"Are you sure you want to update this Bicycle?"	Dialog Box Message	Used to inform the Administrator of the action.															
	Yes	Button	Used to confirm the action.															
No	Button	Used to decline the action.																
Step 8: The Administrator clicks the "Yes" button. [alt]		Step 9: The system then updates and saves the updated Bicycle details in the Bicycle table. The following attributes are updated in the database by using a FIRESQL UPDATE query: Bicycle Table Bicycle Make Date Added																

			Step 10: The system notifies the Administrator that the Bicycle has been updated successfully in the database with the following: [alt] <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Success</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed.</td></tr><tr><td>"The Bicycle has been updated successfully"</td><td>Dialog Box Message</td><td>Used to inform the Administrator of the result.</td></tr><tr><td>Ok</td><td>Button</td><td>Used to accept the result.</td></tr></table>	Text/Label	Control Type	Additional Info	Success	Dialog Box Title	Used to indicate what type of message is being displayed.	"The Bicycle has been updated successfully"	Dialog Box Message	Used to inform the Administrator of the result.	Ok	Button	Used to accept the result.
	Text/Label	Control Type	Additional Info												
Success	Dialog Box Title	Used to indicate what type of message is being displayed.													
"The Bicycle has been updated successfully"	Dialog Box Message	Used to inform the Administrator of the result.													
Ok	Button	Used to accept the result.													
			Step 11: The system updates and saves the following in the database by using a FIRESQL INSERT query: AuditTrail Table AuditTrailID DateTime TransactionDescription UserID												
Alternate Courses:	Alt Step 4: The Administrator clicks the "Cancel" button. Terminate Use Case.														
	Alt Step 5a: In the case of an empty date picker, the system notifies the Administrator of the Input error below the input field with the following: <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr></table>			Text/Label	Control Type	Additional Info									
Text/Label	Control Type	Additional Info													

	"Please fill this field."	Label (with a light red background color fill)	Used to inform the Administrator of the result.
	"X"	Icon	Used to indicate an error result.
	Redirect to Step 3.		
	Alt Step 6: In the case of invalid input being entered, the system notifies the Administrator of the Format error below the input field with the following:		
	Text/Label	Control Type	Additional Info
	"Invalid input."	Label (with a light red background color fill)	Used to inform the Administrator of the result.
	"X"	Icon	Used to indicate an error result.
	Redirect to Step 3.		
	Alt Step 8: The Administrator declines to update the Bicycle by clicking the "No" button. Redirect to Step 4.		
	Alt Step 10: The Bicycle was not updated successfully in the database, the system will notify the Administrator of the update error with the following:		
	Text/Label	Control Type	Additional Info
	Failed	Dialog Box Title	Used to indicate what type of message is being displayed
	"Update unsuccessful. Please try again."	Dialog Box Message	Used to inform the Administrator of the result.

	"Help icon"	Button	Used to Load the Help document.
	Ok	Button	Used to accept the result.
	Redirect to Step 3.		
Conclusion:	The use case concludes when the Administrator has been notified that the Bicycle has been updated successfully.		
Post-Condition:	• An existing Bicycle is updated in the system with the relevant details. • The AuditTrail table is successfully updated. • The Bicycle table is successfully updated.		
Business Rules	• Only an Administrator can update a Bicycle in the system.		
Implementation Constraints and Specifications:	• None		
Assumptions:	• None		
Open Issues:	• None		

2.3.6.4. Delete Bicycle

System Name:	VVs
Author:	Cameron King
Date:	15/06/2020
Version:	1.0

Use Case Name:	Delete Bicycle	Use Case Type:
Use Case ID:	5.4	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	School Administrator	
Primary System Actor	None	
Other Participating Actors:	None	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process of an Administrator deleting an existing Bicycle's details from the system. The Use Case starts when the Administrator navigates by selecting the "Bicycle Management" option under the "Bicycle Management" Combo box in the Navigation bar. From there, they search for the Bicycle they want to delete and the system displays the "Bicycle" screen with the Bicycle details to be deleted. The system populates the relevant Combo boxes with information of the searched Bicycle, and the Administrator can make their deletion.	
Pre-Condition:	The Administrator must be logged into the system.	

Trigger:	The Administrator requests to delete an existing Bicycle from the system.			
Typical Course Of Events:	Actor Action	System Response		
		Manual Action	Automated Action	
	Step 1: The Administrator selects the "Bicycle Management" option under the "Bicycle Management" Combo box in the Navigation bar.		Step 2: The System invokes Use Case 5.2 "Search Bicycle" and displays the "Bicycle" screen with the data of the searched Bicycle.	
	Step 3: The Administrator clicks on the "Delete Bicycle" button.[alt]		Step 4: The system prompts the Administrator to confirm the deletion of the Bicycle's details with the following:	
			Text/Label	Control Type
Confirmation			Dialog Box Title	Used to indicate what type of message is being displayed.
"Are you sure you want to delete this Bicycle?"			Dialog Box Message	Used to inform the Administrator of the action.
Yes			Button	Used to confirm the action.
No	Button	Used to decline the action.		

	Step 5: The Administrator clicks the “Yes” button. [alt]		Step 6: The system then removes the Bicycle details in the Bicycle table. The following attributes are deleted in the database by using a FIRESQL DELETE query: Bicycle Table BicycleID BicycleStatusID StudentID BicycleMake DateAdded TagID												
			Step 7: The system notifies the Administrator that the Bicycle has been deleted successfully from the database with the following: [alt] <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Success</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed.</td></tr><tr><td>“The Bicycle has been deleted successfully”</td><td>Dialog Box Message</td><td>Used to inform the Administrator of the result.</td></tr><tr><td>Ok</td><td>Button</td><td>Used to accept the result.</td></tr></table>	Text/Label	Control Type	Additional Info	Success	Dialog Box Title	Used to indicate what type of message is being displayed.	“The Bicycle has been deleted successfully”	Dialog Box Message	Used to inform the Administrator of the result.	Ok	Button	Used to accept the result.
	Text/Label	Control Type	Additional Info												
Success	Dialog Box Title	Used to indicate what type of message is being displayed.													
“The Bicycle has been deleted successfully”	Dialog Box Message	Used to inform the Administrator of the result.													
Ok	Button	Used to accept the result.													
		Step 8: The system updates and saves the following in the database by using a FIRESQL INSERT query:													

				AuditTrail Table AuditTrailID DateTime TransactionDescription UserID	
Alternate Courses:	Alt Step 3: The Administrator clicks the "Cancel" button. Terminate Use Case.				
	Alt Step 5: The Administrator declines to delete the Bicycle by clicking the "No" button. Redirect to Step 2.				
	Alt Step 7: The Administrator is notified of the unsuccessful delete action with the following:				
	Text/Label	Control Type	Additional Info		
	Failed	Dialog Box Title	Used to indicate what type of message is being displayed		
	"Delete unsuccessful. Please try again."	Dialog Box Message	Used to inform the Administrator of the result.		
	"Help icon"	Button	Used to Load the Help document.		
	Ok	Button	Used to accept the result.		
	Redirect to Step 2.				
	Conclusion:	The use case concludes when the Administrator has been notified that the Bicycle has been successfully deleted.			
	Post-Condition:	• An existing Bicycle is deleted from the system with all the relevant details being removed. • The AuditTrail table is successfully updated.			

	The Bicycle table is successfully updated with the deleted details being removed.
Business Rules	Only an Administrator can delete a Bicycle from the system.
Implementation Constraints and Specifications:	None
Assumptions:	None
Open Issues:	None

2.3.6.5. Assign Bicycle to Student

System Name:	VVs
Author:	Cameron King
Date:	15/06/2020
Version:	1.0

Use Case Name:	Assign Bicycle to Student	Use Case Type:
Use Case ID:	5.5	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	School Administrator	
Primary System Actor	None	
Other Participating Actors:	None	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process of an Administrator assigning a Bicycle to a Student. The Use Case starts when the Administrator navigates by selecting the "Bicycle Assignment" option under the "Bicycle Management" Combo box in the Navigation bar. From there, they click the "Assign" button and the system will load the "Assign Bicycle to Student" screen. The system populates the relevant combo boxes with information necessary for selection, and the corresponding labels, links, and tables.	
Pre-Condition:	The Administrator must be logged into the system.	

Trigger:	The Administrator requests to assign a Bicycle to a Student to the system.				
Typical Course Of Events:	Actor Action	System Response			
		Manual Action	Automated Action		
	Step 1: The Administrator selects the “Bicycle Assignment” option and under the “Bicycle Management” Combo box in the Navigation bar.		Step 2: The system displays the “Bicycle Management” screen and populates it with the following details:		
			Text/Label	Control Type	Additional Info
			Navigation	Navigation bar	Used for navigation of the site.
			Search	Button	Used to search for a bicycle or student.
			Search	Text-Box	Used for capturing Search criteria for the specific Bicycle.
			Bicycle Assignment Management	Table	Displays a list of Bicycles.
			Assign	Button	Used to navigate to the specific Bicycle view.
			The table is populated by using a FIRESQL READ query with data from the following:		
Entity			Attributes	Input Name	
Bicycle	BicycleCode	Bicycle Code			

			<table><tr><td>Student</td><td>StudentName</td><td>Student Name</td></tr><tr><td>Student</td><td>StudentSurname</td><td>Student Surname</td></tr><tr><td>BicycleStatus</td><td>BicycleStatus</td><td>Bicycle Status</td></tr></table>	Student	StudentName	Student Name	Student	StudentSurname	Student Surname	BicycleStatus	BicycleStatus	Bicycle Status							
			Student	StudentName	Student Name														
			Student	StudentSurname	Student Surname														
			BicycleStatus	BicycleStatus	Bicycle Status														
	The FIRESQL READ query will display the BicycleStatus in the BicycleStatus table that corresponds with the BicycleStatusID in the Bicycle and BicycleStatus tables.																		
	The FIRESQL READ query will compare the StudentID's in the Student table with the StudentID's that appear in the BicycleStudent table. The Student combo box will display the StudentName and StudentSurname that do not appear in the BicycleStudent table and thus will be eligible to be assigned.																		
	Step 3: The Administrator clicks the "Assign" button.			Step 4: The system will load the "Assign Bicycle to Student" screen and displays it with the following controls and layout:															
				<table><tr><td>Text/Label</td><td>Control Type</td><td>Additional Info</td></tr><tr><td>Menu</td><td>Menu-icon</td><td>Used to Navigate the side menu-bar.</td></tr><tr><td>Assign Bicycle to Student</td><td>Label</td><td>Heading for the pop-up displayed.</td></tr><tr><td>"Help icon"</td><td>Button</td><td>Used to Load the Help document.</td></tr><tr><td>Bicycle Code</td><td>Text box</td><td>Used to indicate what the corresponding read-only text box is used for.</td></tr></table>	Text/Label	Control Type	Additional Info	Menu	Menu-icon	Used to Navigate the side menu-bar.	Assign Bicycle to Student	Label	Heading for the pop-up displayed.	"Help icon"	Button	Used to Load the Help document.	Bicycle Code	Text box	Used to indicate what the corresponding read-only text box is used for.
				Text/Label	Control Type	Additional Info													
				Menu	Menu-icon	Used to Navigate the side menu-bar.													
Assign Bicycle to Student				Label	Heading for the pop-up displayed.														
"Help icon"				Button	Used to Load the Help document.														
Bicycle Code	Text box	Used to indicate what the corresponding read-only text box is used for.																	

			Select Student to Assign Combo box Used to indicate what the corresponding combo box is used for. The Select Student to Assign Combo box options are populated with StudentName and StudentSurname that have not yet been assigned a bicycle by using a FIRESQL READ query with data from the following: <table><thead><tr><th>Entity</th><th>Attributes</th></tr></thead><tbody><tr><td>Student</td><td>StudentID</td></tr><tr><td>BicycleStudent</td><td>StudentID</td></tr><tr><td>Student</td><td>StudentName</td></tr><tr><td>Student</td><td>StudentSurname</td></tr></tbody></table> The FIRESQL READ query will compare the StudentID's in the Student table with the StudentID's that appear in the BicycleStudent table. The Student combo box will display the StudentName and StudentSurname that do not appear in the BicycleStudent table and thus will be eligible to be assigned.	Entity	Attributes	Student	StudentID	BicycleStudent	StudentID	Student	StudentName	Student	StudentSurname
	Entity	Attributes											
	Student	StudentID											
	BicycleStudent	StudentID											
	Student	StudentName											
Student	StudentSurname												
Step 5: The Administrator selects the following details:													
Select Student to Assign													
Step 6: The Administrator clicks the "Assign" button. [alt]													
		Step 7: The system validates that the captured details are in the correct format: [alt]											

			BicycleStudent table <table><tr><th>Attribute</th><th>Validation</th></tr><tr><td>StudentID</td><td>31.Option must be selected from the combo box.</td></tr></table>	Attribute	Validation	StudentID	31.Option must be selected from the combo box.											
	Attribute	Validation																
	StudentID	31.Option must be selected from the combo box.																
			Step 8: The system prompts the Administrator to confirm to assign the Bicycle to the Student with the following: <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Confirmation</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed.</td></tr><tr><td>"Are you sure you want to assign this Bicycle?"</td><td>Dialog Box Message</td><td>Used to inform the Administrator of the action.</td></tr><tr><td>Yes</td><td>Button</td><td>Used to confirm the action.</td></tr><tr><td>No</td><td>Button</td><td>Used to decline the action.</td></tr></table>	Text/Label	Control Type	Additional Info	Confirmation	Dialog Box Title	Used to indicate what type of message is being displayed.	"Are you sure you want to assign this Bicycle?"	Dialog Box Message	Used to inform the Administrator of the action.	Yes	Button	Used to confirm the action.	No	Button	Used to decline the action.
	Text/Label	Control Type	Additional Info															
	Confirmation	Dialog Box Title	Used to indicate what type of message is being displayed.															
	"Are you sure you want to assign this Bicycle?"	Dialog Box Message	Used to inform the Administrator of the action.															
	Yes	Button	Used to confirm the action.															
	No	Button	Used to decline the action.															

	Step 9: The Administrator clicks the “Yes” button. [alt]		Step 10: The system then updates and saves the updated Bicycle details in the BicycleStudent and Bicycle table. The BicycleStatusID is saved with the same BicycleStatusID from the BicycleStatus where the BicycleStatus is ‘Assigned’. The following attributes are updated in the database by using a FIRESQL UPDATE query: Bicycle Table BicycleStatusID BicycleStudent Table BicycleID StudentID DateAssigned								
			Step 11: The system notifies the Administrator that the Bicycle has been successfully assigned to the Student in the database with the following: [alt] <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Success</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed.</td></tr><tr><td>“Bicycle has been successfully assigned”</td><td>Dialog Box Message</td><td>Used to inform the Administrator of the result.</td></tr></table>	Text/Label	Control Type	Additional Info	Success	Dialog Box Title	Used to indicate what type of message is being displayed.	“Bicycle has been successfully assigned”	Dialog Box Message
Text/Label	Control Type	Additional Info									
Success	Dialog Box Title	Used to indicate what type of message is being displayed.									
“Bicycle has been successfully assigned”	Dialog Box Message	Used to inform the Administrator of the result.									

			Ok	Button	Used to accept the result.									
			Step 12: The system creates and saves the following in the database by using a FIRESQL INSERT query: AuditTrail Table AuditTrailID DateTime TransactionDescription UserID											
Alternate Courses:	Alt Step 6: The Administrator clicks the "Cancel" button. Terminate Use Case.													
	Alt Step 7a: In the case of an empty combo box, the system notifies the Administrator of the Input error below the input field with the following:													
	<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>"Please fill this field."</td><td>Label (with a light red background color fill)</td><td>Used to inform the Administrator of the result.</td></tr><tr><td>"X"</td><td>Icon</td><td>Used to indicate an error result.</td></tr></table>					Text/Label	Control Type	Additional Info	"Please fill this field."	Label (with a light red background color fill)	Used to inform the Administrator of the result.	"X"	Icon	Used to indicate an error result.
	Text/Label	Control Type	Additional Info											
	"Please fill this field."	Label (with a light red background color fill)	Used to inform the Administrator of the result.											
"X"	Icon	Used to indicate an error result.												
Redirect to Step 5.														
Alt Step 7b: In the case of invalid input being entered, the system notifies the Administrator of the Format error below the input field with the following:														
<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>"Invalid input."</td><td>Label (with a light red background color fill)</td><td>Used to inform the Administrator of the result.</td></tr></table>					Text/Label	Control Type	Additional Info	"Invalid input."	Label (with a light red background color fill)	Used to inform the Administrator of the result.				
Text/Label	Control Type	Additional Info												
"Invalid input."	Label (with a light red background color fill)	Used to inform the Administrator of the result.												

	"X"	Icon	Used to indicate an error result.															
	Redirect to Step 5.																	
	Alt Step 9: The Administrator declines to assign the Bicycle to the Student by clicking the "No" button. Redirect to Step 6.																	
	Alt Step 11: The Bicycle was not successfully assigned; the system will notify the Administrator of the error with the following:																	
	<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Failed</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed</td></tr><tr><td>"Assign unsuccessful. Please try again."</td><td>Dialog Box Message</td><td>Used to inform the Administrator of the result.</td></tr><tr><td>"Help icon"</td><td>Button</td><td>Used to Load the Help document.</td></tr><tr><td>Ok</td><td>Button</td><td>Used to accept the result.</td></tr></table>			Text/Label	Control Type	Additional Info	Failed	Dialog Box Title	Used to indicate what type of message is being displayed	"Assign unsuccessful. Please try again."	Dialog Box Message	Used to inform the Administrator of the result.	"Help icon"	Button	Used to Load the Help document.	Ok	Button	Used to accept the result.
	Text/Label	Control Type	Additional Info															
Failed	Dialog Box Title	Used to indicate what type of message is being displayed																
"Assign unsuccessful. Please try again."	Dialog Box Message	Used to inform the Administrator of the result.																
"Help icon"	Button	Used to Load the Help document.																
Ok	Button	Used to accept the result.																
Redirect to Step 5.																		
Conclusion:	The use case concludes when the Administrator has been notified that the Bicycle has been successfully assigned to a Student.																	
Post-Condition:	• A Bicycle has been successfully assigned to a Student. • The AuditTrail table is successfully updated. • The BicycleStudent and Bicycle table is successfully updated.																	

Business Rules	Only an Administrator can assign a Bicycle to a Student to the system.
Implementation Constraints and Specifications:	None
Assumptions:	None
Open Issues:	None

2.3.6.6. Unassign Bicycle from Student

System Name:	VVs
Author:	Cameron King
Date:	15/06/2020
Version:	1.0

Use Case Name:	Unassign Bicycle from Student	Use Case Type:
Use Case ID:	5.6	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	School Administrator	
Primary System Actor	None	
Other Participating Actors:	None	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process of an Administrator unassigning a Bicycle from a Student. The Use Case starts when the Administrator navigates by selecting the "Bicycle Assignment" option under the "Bicycle Management" Combo box in the Navigation bar. From there, they click the "Unassign" button. The system populates the relevant Combo boxes with information necessary for selection, and the corresponding labels, text boxes and buttons, icons, logos, and banners.	
Pre-Condition:	The Administrator must be logged into the system.	

Trigger:	The Administrator requests to add unassign a Bicycle from a Student.																							
Typical Course Of Events:	Actor Action	System Response																						
		Manual Action	Automated Action																					
	Step 1: The Administrator selects the “Bicycle Assignment” option and under the “Bicycle Management” Combo box in the Navigation bar.		Step 2: The system displays the “Bicycle Assignment” screen and populates it with the following details:																					
			<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Navigation</td><td>Navigation bar</td><td>Used for navigation of the site.</td></tr><tr><td>Search</td><td>Button</td><td>Used to search for a bicycle or student.</td></tr><tr><td>Search</td><td>Text-Box</td><td>Used for capturing Search criteria for the specific Bicycle.</td></tr><tr><td>Bicycle Assignment Management</td><td>Table</td><td>Displays a list of Bicycles.</td></tr><tr><td>“Pen icon”</td><td>Button</td><td>Used to navigate to the specific Bicycle view.</td></tr><tr><td>“Help Icon”</td><td>Button</td><td>Used to Load the Help document.</td></tr></table>	Text/Label	Control Type	Additional Info	Navigation	Navigation bar	Used for navigation of the site.	Search	Button	Used to search for a bicycle or student.	Search	Text-Box	Used for capturing Search criteria for the specific Bicycle.	Bicycle Assignment Management	Table	Displays a list of Bicycles.	“Pen icon”	Button	Used to navigate to the specific Bicycle view.	“Help Icon”	Button	Used to Load the Help document.
			Text/Label	Control Type	Additional Info																			
			Navigation	Navigation bar	Used for navigation of the site.																			
			Search	Button	Used to search for a bicycle or student.																			
			Search	Text-Box	Used for capturing Search criteria for the specific Bicycle.																			
			Bicycle Assignment Management	Table	Displays a list of Bicycles.																			
			“Pen icon”	Button	Used to navigate to the specific Bicycle view.																			
			“Help Icon”	Button	Used to Load the Help document.																			
			The table is populated by using a FIRESQL READ query with data from the following:																					

			<table><tr><th>Entity</th><th>Attributes</th></tr><tr><td>Student</td><td>StudentID</td></tr><tr><td>BicycleStudent</td><td>StudentID</td></tr><tr><td>Student</td><td>StudentName</td></tr><tr><td>Student</td><td>StudentSurname</td></tr></table>	Entity	Attributes	Student	StudentID	BicycleStudent	StudentID	Student	StudentName	Student	StudentSurname
			Entity	Attributes									
			Student	StudentID									
			BicycleStudent	StudentID									
			Student	StudentName									
Student	StudentSurname												
	The FIRESQL READ query will display the BicycleStatus in the BicycleStatus table that corresponds with the BicycleStatusID in the Bicycle and BicycleStatus tables.												
	The FIRESQL READ query will compare the StudentID's in the Student table with the StudentID's that appear in the BicycleStudent table. The Student combo box will display the StudentName and StudentSurname that do not appear in the BicycleStudent table and thus will be eligible to be assigned.												
Step 3: The Administrator clicks the "Unassign" button.		Step 4: The system prompts the Administrator to confirm to unassign the Bicycle from the Student with the following:											
		<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Confirmation</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed.</td></tr><tr><td>"Are you sure you want to unassign this Bicycle?"</td><td>Dialog Box Message</td><td>Used to inform the Administrator of the action.</td></tr></table>	Text/Label	Control Type	Additional Info	Confirmation	Dialog Box Title	Used to indicate what type of message is being displayed.	"Are you sure you want to unassign this Bicycle?"	Dialog Box Message	Used to inform the Administrator of the action.		
Text/Label	Control Type	Additional Info											
Confirmation	Dialog Box Title	Used to indicate what type of message is being displayed.											
"Are you sure you want to unassign this Bicycle?"	Dialog Box Message	Used to inform the Administrator of the action.											

			Yes	Button	Used to confirm the action.
			No	Button	Used to decline the action.
	Step 5: The Administrator clicks the "Yes" button. [alt]		Step 6: The system then updates and saves the updated Bicycle details in the BicycleStudent table.		
			The BicycleStudent Table BicycleID StudentID DateAssigned DateUnassigned Bicycle Table BicycleStatusID BicycleStatusID is saved with the same BicycleStatusID from the BicycleStatus where the BicycleStatus is 'Available'. The DateAssigned attribute is set to null and the DateUnassigned attribute is updated with the current date. The following attributes are updated in the database by using a FIRESQL UPDATE query		
			Step 7: The system notifies the Administrator that the Bicycle has been successfully unassigned from the Student in the database with the following: [alt]		

			<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Success</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed.</td></tr><tr><td>"Bicycle has been successfully unassigned"</td><td>Dialog Box Message</td><td>Used to inform the Administrator of the result.</td></tr><tr><td>Ok</td><td>Button</td><td>Used to accept the result.</td></tr></table>	Text/Label	Control Type	Additional Info	Success	Dialog Box Title	Used to indicate what type of message is being displayed.	"Bicycle has been successfully unassigned"	Dialog Box Message	Used to inform the Administrator of the result.	Ok	Button	Used to accept the result.
	Text/Label	Control Type	Additional Info												
	Success	Dialog Box Title	Used to indicate what type of message is being displayed.												
	"Bicycle has been successfully unassigned"	Dialog Box Message	Used to inform the Administrator of the result.												
	Ok	Button	Used to accept the result.												
		Step 8: The system creates and saves the following in the database by using a FIRESQL INSERT query: <table><tr><th>AuditTrail Table</th></tr><tr><td>AuditTrailID</td></tr><tr><td>DateTime</td></tr><tr><td>TransactionDescription</td></tr><tr><td>UserID</td></tr></table>	AuditTrail Table	AuditTrailID	DateTime	TransactionDescription	UserID								
AuditTrail Table															
AuditTrailID															
DateTime															
TransactionDescription															
UserID															
Alt Step 5: The Administrator declines to assign the Bicycle to the Student by clicking the "No" button. Redirect to Step 1.															
Alt Step 7: The Bicycle was not successfully unassigned from the Student, the system will notify the Administrator of the error with the following:															
		<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Failed</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed</td></tr></table>	Text/Label	Control Type	Additional Info	Failed	Dialog Box Title	Used to indicate what type of message is being displayed							
Text/Label	Control Type	Additional Info													
Failed	Dialog Box Title	Used to indicate what type of message is being displayed													

	"Unassign action unsuccessful. Please try again."	Dialog Box Message	Used to inform the Administrator of the result.
	"Help icon"	Button	Used to Load the Help document.
	Ok	Button	Used to accept the result.
	Redirect to Step 1.		
Conclusion:	The use case concludes when the Administrator has been notified that the Bicycle has been successfully unassigned from the Student.		
Post-Condition:	• A Bicycle has been successfully unassigned from the Student. • The AuditTrail table is successfully updated. • The BicycleStudent and Bicycle table is successfully updated.		
Business Rules	• Only an Administrator can assign a Bicycle to a Student to the system.		
Implementation Constraints and Specifications:	• None		
Assumptions:	• None		
Open Issues:	• None		

2.3.6.7. Assign Tag to Bicycle

System Name:	VVs
Author:	Cameron King
Date:	15/06/2020
Version:	1.0

Use Case Name:	Assign Tag to Bicycle	Use Case Type:
Use Case ID:	5.7	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	School Administrator	
Primary System Actor	None	
Other Participating Actors:	None	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process of an Administrator assigning a Tag to a Bicycle. The Use Case starts when the Administrator navigates by selecting the "Tag Assignment" or option under the "Bicycle Management" Combo box in the Navigation bar. From there, they click the "Unassign" button and the system will load the "Unassign Bicycle" screen. The system populates the relevant Combo boxes with information necessary for selection, and the corresponding labels, text boxes and buttons, icons, logos, and banners.	
Pre-Condition:	The Administrator must be logged into the system.	

Trigger:	The Administrator requests to assign a Tag to a Bicycle to the system.																				
Typical Course Of Events:	Actor Action	System Response																			
		Manual Action	Automated Action																		
	Step 1: The Administrator selects the “Tag Assignment” option and under the “Bicycle Management” Combo box in the Navigation bar.		Step 2: The system displays the “Tag Assignment” screen and populates it with the following details:																		
			<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Navigation</td><td>Navigation bar</td><td>Used for navigation of the site.</td></tr><tr><td>Search</td><td>Button</td><td>Used to search for a bicycle or student.</td></tr><tr><td>Search</td><td>Text-Box</td><td>Used for capturing Search criteria for the specific Bicycle.</td></tr><tr><td>Tag Assignment</td><td>Table</td><td>Displays a list of Bicycles.</td></tr><tr><td>“Pen icon”</td><td>Button</td><td>Used to navigate to the specific Bicycle view.</td></tr></table>	Text/Label	Control Type	Additional Info	Navigation	Navigation bar	Used for navigation of the site.	Search	Button	Used to search for a bicycle or student.	Search	Text-Box	Used for capturing Search criteria for the specific Bicycle.	Tag Assignment	Table	Displays a list of Bicycles.	“Pen icon”	Button	Used to navigate to the specific Bicycle view.
			Text/Label	Control Type	Additional Info																
			Navigation	Navigation bar	Used for navigation of the site.																
			Search	Button	Used to search for a bicycle or student.																
			Search	Text-Box	Used for capturing Search criteria for the specific Bicycle.																
			Tag Assignment	Table	Displays a list of Bicycles.																
			“Pen icon”	Button	Used to navigate to the specific Bicycle view.																
			The table is populated by using a FIRESQL READ query with data from the following:																		
			<table><tr><th>Entity</th><th>Attributes</th><th>Input Name</th></tr><tr><td>Tag</td><td>TagSerialNumber</td><td>Tag Serial Number</td></tr></table>	Entity	Attributes	Input Name	Tag	TagSerialNumber	Tag Serial Number												
Entity	Attributes	Input Name																			
Tag	TagSerialNumber	Tag Serial Number																			

			<table><tr><td>Bicycle</td><td>BicycleCode</td><td>Bicycle Code</td></tr><tr><td>TagStatus</td><td>TagStatus</td><td>Tag Status</td></tr></table> The FIRESQL READ query will display the TagStatus in the TagStatus table that corresponds with the TagStatusID in the Tag and TagStatus tables.	Bicycle	BicycleCode	Bicycle Code	TagStatus	TagStatus	Tag Status					
	Bicycle	BicycleCode	Bicycle Code											
	TagStatus	TagStatus	Tag Status											
	Step 3: The Administrator clicks on the "Assign" button.		Step 4: The system will load the "Assign Tag to Bicycle" screen and displays it with the following controls and layout: <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Navigation</td><td>Nav-bar</td><td>Used to Navigate the site.</td></tr><tr><td>Tag Serial Number</td><td>Text box</td><td>Used to indicate what the corresponding read-only text box is used for.</td></tr><tr><td>Select Bicycle to Assign</td><td>Combo box</td><td>Used to indicate what the corresponding combo box is used for.</td></tr></table> The FIRESQL READ query will display the TagStatus in the TagStatus table that corresponds with the TagStatusID in the Tag and TagStatus tables.	Text/Label	Control Type	Additional Info	Navigation	Nav-bar	Used to Navigate the site.	Tag Serial Number	Text box	Used to indicate what the corresponding read-only text box is used for.	Select Bicycle to Assign	Combo box
Text/Label	Control Type	Additional Info												
Navigation	Nav-bar	Used to Navigate the site.												
Tag Serial Number	Text box	Used to indicate what the corresponding read-only text box is used for.												
Select Bicycle to Assign	Combo box	Used to indicate what the corresponding combo box is used for.												

			The FIRESQL READ query will compare the TagID’s in the Tag table with the TagID’s that appear in the Bicycle table. The Bicycle to Assign combo box will display the BicycleCode and that do not appear in the Bicycle table and thus will be eligible to be assigned.								
	Step 5: The Administrator selects an option from the combo box Bicycle Code										
	Step 6: The Administrator clicks the “Assign” button.		Step 7: The system validates that the captured details are in the correct format: [alt] <i>Bicycle table</i><table><tr><th>Attribute</th><th>Validation</th></tr><tr><td>BicycleCode</td><td>Option must be selected from the combo box.</td></tr></table>	Attribute	Validation	BicycleCode	Option must be selected from the combo box.				
	Attribute	Validation									
BicycleCode	Option must be selected from the combo box.										
		Step 8: The system prompts the Administrator to confirm to assign the Tag to the Bicycle with the following: <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Confirmation</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed.</td></tr><tr><td>“Are you sure you want to assign this Bicycle?”</td><td>Dialog Box Message</td><td>Used to inform the Administrator of the action.</td></tr></table>	Text/Label	Control Type	Additional Info	Confirmation	Dialog Box Title	Used to indicate what type of message is being displayed.	“Are you sure you want to assign this Bicycle?”	Dialog Box Message	Used to inform the Administrator of the action.
Text/Label	Control Type	Additional Info									
Confirmation	Dialog Box Title	Used to indicate what type of message is being displayed.									
“Are you sure you want to assign this Bicycle?”	Dialog Box Message	Used to inform the Administrator of the action.									

			Yes	Button	Used to confirm the action.						
			No	Button	Used to decline the action.						
	Step 9: The Administrator clicks the “Yes” button. [alt]		Step 10: The system then updates and saves the updated Tag details in the Tag table. The TagStatusID is saved with the same TagStatusID from the TagStatus where the TagStatus is ‘Assigned’.								
			The BicycleStatusID is saved with the same BicycleStatusID from the BicycleStatus where the BicycleStatus is ‘Available’. The following attributes are updated in the database by using a FIRESQL UPDATE query:								
			Bicycle Table TagID BicycleStatusID								
			Tag Table TagStatusID DateInstalled								
			Step 11: The system notifies the Administrator that the Tag has been successfully assigned to the Bicycle in the database with the following: [alt]								
			<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Success</td><td>Dialog Box Title</td><td>Used to indicate what type of</td></tr></table>			Text/Label	Control Type	Additional Info	Success	Dialog Box Title	Used to indicate what type of
Text/Label	Control Type	Additional Info									
Success	Dialog Box Title	Used to indicate what type of									

				message is being displayed.								
			"Tag has been successfully assigned"	Dialog Box Message	Used to inform the Administrator of the result.							
			Ok	Button	Used to accept the result.							
				Step 12: The system creates and saves the following in the database by using a FIRESQL INSERT query: AuditTrail Table AuditTrailID DateTime TransactionDescription UserID								
Alternate Courses:	Alt Step 6: The Administrator clicks the "Cancel" button. Terminate Use Case.											
	Alt Step 7a: In the case of an empty combo box, the system notifies the Administrator of the Input error below the input field with the following:											
	<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>"Please fill this field."</td><td>Label (with a light red background color fill)</td><td>Used to inform the Administrator of the result.</td></tr><tr><td>"X"</td><td>Icon</td><td>Used to indicate an error result.</td></tr></table>	Text/Label	Control Type	Additional Info	"Please fill this field."	Label (with a light red background color fill)	Used to inform the Administrator of the result.	"X"	Icon	Used to indicate an error result.	Redirect to Step 5.	
Text/Label	Control Type	Additional Info										
"Please fill this field."	Label (with a light red background color fill)	Used to inform the Administrator of the result.										
"X"	Icon	Used to indicate an error result.										

Alt Step 7b: In the case of invalid input being entered, the system notifies the Administrator of the Format error below the input field with the following:

Text/Label	Control Type	Additional Info
"Invalid input."	Label (with a light red background color fill)	Used to inform the Administrator of the result.
"X"	Icon	Used to indicate an error result.

Redirect to Step 5.

Alt Step 9: The Administrator declines to assign the Tag to the Bicycle by clicking the "No" button. Redirect to Step 4.

Alt Step 11: The Tag was not successfully assigned, the system will notify the Administrator of the error with the following:

Text/Label	Control Type	Additional Info
Failed	Dialog Box Title	Used to indicate what type of message is being displayed
"Assign action unsuccessful. Please try again."	Dialog Box Message	Used to inform the Administrator of the result.
"Help icon"	Button	Used to Load the Help document.
Ok	Button	Used to accept the result.

Redirect to Step 4.

Conclusion:	The use case concludes when the Administrator has been notified that the Tag has been successfully assigned to a Bicycle.
Post-Condition:	• A Tag has been successfully assigned to a Bicycle. • The AuditTrail table is successfully updated. • The Tag and Bicycle table is successfully updated.
Business Rules	• Only an Administrator can assign a Tag to a Bicycle in the system.
Implementation Constraints and Specifications:	• None
Assumptions:	• None
Open Issues:	• None

2.3.6.8. Unassign Tag from Bicycle

System Name:	VVs
Author:	Cameron King
Date:	15/06/2020
Version:	1.0

Use Case Name:	Unassign Tag from Bicycle	Use Case Type:
Use Case ID:	5.8	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	School Administrator	
Primary System Actor	None	
Other Participating Actors:	None	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process of an Administrator unassigning a Tag from a Bicycle. The Use Case starts when the Administrator navigates by selecting the "Bicycle Management" option under the "Bicycle Management" Combo box in the Navigation bar. From there, they click the "Assign or Unassign Tag" button and the system will load the "Assign or Unassign Tag" screen. The system populates the relevant Combo boxes with information necessary for selection, and the corresponding labels, tables, buttons, icons, logos, and banners.	
Pre-Condition:	The Administrator must be logged into the system.	

Trigger:	The Administrator requests to add unassign a Bicycle from a Student.				
Typical Course Of Events:	Actor Action	System Response			
		Manual Action	Automated Action		
	Step 1: The Administrator selects the "Bicycle Management" option and under the "Bicycle Management" Combo box in the Navigation bar.		Step 2: The system displays the "Bicycle Management" screen and populates it with the following details:		
			Text/Label	Control Type	Additional Info
			Navigation	Navigation bar	Used for navigation of the site.
			Search	Button	Used to search for a bicycle or student.
			Search	Text-Box	Used for capturing Search criteria for the specific Bicycle.
			Bicycle Assignment Management	Table	Displays a list of Bicycles.
			"Pen icon"	Button	Used to navigate to the specific Bicycle view.
			"Help Icon"	Button	Used to Load the Help document.
			The table is populated by using a FIRESQL READ query with data from the following:		

			<table><tr><th>Entity</th><th>Attributes</th><th>Input Name</th></tr><tr><td>Tag</td><td>TagSerialNumber</td><td>Tag Serial Number</td></tr><tr><td>Bicycle</td><td>BicycleCode</td><td>Bicycle Code</td></tr><tr><td>TagStatus</td><td>TagStatus</td><td>Tag Status</td></tr></table>	Entity	Attributes	Input Name	Tag	TagSerialNumber	Tag Serial Number	Bicycle	BicycleCode	Bicycle Code	TagStatus	TagStatus	Tag Status			
	Entity	Attributes	Input Name															
	Tag	TagSerialNumber	Tag Serial Number															
	Bicycle	BicycleCode	Bicycle Code															
	TagStatus	TagStatus	Tag Status															
			The FIRESQL READ query will display the TagStatus in the TagStatus table that corresponds with the TagStatusID in the Tag and TagStatus tables.															
	Step 6: The Administrator clicks the “Unassign” button.		Step 7: The system prompts the Administrator to confirm to unassign the Tag from the Bicycle with the following: <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Confirmation</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed.</td></tr><tr><td>“Are you sure you want to unassign this Tag?”</td><td>Dialog Box Message</td><td>Used to inform the Administrator of the action.</td></tr><tr><td>Yes</td><td>Button</td><td>Used to confirm the action.</td></tr><tr><td>No</td><td>Button</td><td>Used to decline the action.</td></tr></table>	Text/Label	Control Type	Additional Info	Confirmation	Dialog Box Title	Used to indicate what type of message is being displayed.	“Are you sure you want to unassign this Tag?”	Dialog Box Message	Used to inform the Administrator of the action.	Yes	Button	Used to confirm the action.	No	Button	Used to decline the action.
	Text/Label	Control Type	Additional Info															
	Confirmation	Dialog Box Title	Used to indicate what type of message is being displayed.															
	“Are you sure you want to unassign this Tag?”	Dialog Box Message	Used to inform the Administrator of the action.															
Yes	Button	Used to confirm the action.																
No	Button	Used to decline the action.																

	Step 8: The Administrator clicks the "Yes" button. [alt]		Step 9: The system then updates and saves the updated Tag and Bicycle details in the Bicycle table. The TagStatusID in the Tag Table is saved with the same TagStatusID from the TagStatus table where the TagStatus is 'Available.' The BicycleStatusID in the Bicycle table is saved with the same BicycleStatusID from the BicycleStatus table where the BicycleStatus is 'Available' The DateInstalled attribute is set to null and the DateUninstalled attribute is updated with the current date. The following attributes are updated in the database by using a FIRESQL UPDATE query: Tag Table TagStatusID DateInstalled DateUninstalled Bicycle Table BicycleStatusID TagID
			Step 10: The system notifies the Administrator that the Tag has been successfully unassigned from the Student in the database with the following: [alt]

			Text/Label	Control Type	Additional Info						
			Success	Dialog Box Title	Used to indicate what type of message is being displayed.						
			"Tag has been successfully unassigned"	Dialog Box Message	Used to inform the Administrator of the result.						
			Ok	Button	Used to accept the result.						
			Step 11: The system creates and saves the following in the database by using a FIRESQL INSERT query:								
			AuditTrail Table								
			AuditTrailID								
			DateTime								
			TransactionDescription								
					UserID						
Alt Step 8: The Administrator declines to unassign the Tag from the Bicycle by clicking the "No" button. Redirect to Step 4.											
Alt Step 9: The Tag was not successfully unassigned from the Bicycle, the system will notify the Administrator of the error with the following:											
<table><tr><td>Text/Label</td><td>Control Type</td><td>Additional Info</td></tr><tr><td>Failed</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed</td></tr></table>						Text/Label	Control Type	Additional Info	Failed	Dialog Box Title	Used to indicate what type of message is being displayed
Text/Label	Control Type	Additional Info									
Failed	Dialog Box Title	Used to indicate what type of message is being displayed									

	"Unassign action unsuccessful. Please try again."	Dialog Box Message	Used to inform the Administrator of the result.
	"Help icon"	Button	Used to Load the Help document.
	Ok	Button	Used to accept the result.
	Redirect to Step 5.		
Conclusion:	The use case concludes when the Administrator has been notified that the Tag has been successfully unassigned from the Bicycle.		
Post-Condition:	• A Tag has been successfully unassigned from the Bicycle. • The AuditTrail table is successfully updated. • The Tag and Bicycle table is successfully updated.		
Business Rules	• Only an Administrator can unassign a Tag from a Bicycle to the system.		
Implementation Constraints and Specifications:	• None		
Assumptions:	• None		
Open Issues:	• None		

2.3.7. Sub-system 6: Bicycle Part

2.3.7.1. Add Bicycle Part

System Name:	VVs
Author:	Tapiwarufaro Ndoro
Date:	15/06/2020
Version:	1.0

Use Case Name:	Add Bicycle Part	Use Case Type:
Use Case ID:	6.1	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	Administrator	
Primary System Actor	None	
Other Participating Actors:	None	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process of an Administrator adding a new Bicycle Part to the system. The Use Case starts when the Administrator navigates by selecting the "Bicycle Part Management" option under the "Bicycle Management" Combo box in the Navigation bar. From there, they click the "+ icon" button and the system will load the "Bicycle Part" screen. The system populates the relevant Combo boxes with information necessary for selection, and the corresponding labels, text boxes and buttons, icons, logos, and banners.	

Pre-Condition:	The Administrator must be logged into the system.																				
Trigger:	The Administrator requests to add a new Bicycle Part to the system.																				
Typical Course Of Events:	Actor Action	System Response																			
		Manual Action	Automated Action																		
	Step 1: The Administrator selects the “Bicycle Part Management” option under the “Bicycle Management” Combo box in the Navigation bar.		Step 2: The system Displays the “Bicycle Part Management” screen and populates it with the following details:																		
			<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>+ icon</td><td>Button</td><td>Used to load the “Bicycle Part” screen.</td></tr><tr><td>Search</td><td>Text-Box</td><td>Used for capturing Search criteria for the specific Bicycle Part.</td></tr><tr><td>Bicycle Part Management</td><td>Table</td><td>Displays a list of Bicycle Parts.</td></tr><tr><td>“Pen icon”</td><td>Button</td><td>Used to navigate to the specific Bicycle Part view.</td></tr><tr><td>Search</td><td>Button</td><td>Used to perform the search action</td></tr></table>	Text/Label	Control Type	Additional Info	+ icon	Button	Used to load the “Bicycle Part” screen.	Search	Text-Box	Used for capturing Search criteria for the specific Bicycle Part.	Bicycle Part Management	Table	Displays a list of Bicycle Parts.	“Pen icon”	Button	Used to navigate to the specific Bicycle Part view.	Search	Button	Used to perform the search action
			Text/Label	Control Type	Additional Info																
			+ icon	Button	Used to load the “Bicycle Part” screen.																
			Search	Text-Box	Used for capturing Search criteria for the specific Bicycle Part.																
			Bicycle Part Management	Table	Displays a list of Bicycle Parts.																
			“Pen icon”	Button	Used to navigate to the specific Bicycle Part view.																
			Search	Button	Used to perform the search action																
The table is populated by using a FIRESQL READ query with data from the following:																					

			Entity	Attributes	Input Name
			Part	PartName	Part Name
			Part	PartDescription	Part Description
	Step 3: The Administrator clicks on the "+ icon" button.		Section	SectionName	Bicycle Section
			Step 4: The system will load the "Bicycle Part" screen and displays it with the following controls and layout:		
			Text/Label	Control Type	Additional Info
			Navigation	Navigation-bar	Used for Navigation of the site.
			Bicycle Part	Label	Heading for the pop-up displayed.
			Choose Picture	Button	Facilitates uploading a picture.
			Part Picture	Picture image	Displays the uploaded picture.
			Part Name	Label	Used to indicate what the corresponding text box is used for.
			Part Description	Label	Used to indicate what the corresponding

					text box is used for.
			Bicycle Section	Label	Used to indicate what the corresponding Combo box is used for.
			Part Name	Text box	Used to capture text for the Bicycle Part Name details.
			Part Description	Text box	Used to capture text for the Bicycle Part Description details.
			Bicycle Section	Combo box	Used to Select the Bicycle Section with this Part and is Populated by contents from the ERD with options as indicated in the table below.
			Create Part	Button	Used to Save the new Bicycle Part Details.
			Cancel	Button	Used to cancel the action of

			adding a Bicycle Part. The Bicycle Section Combo box options are populated by using a FIRESQL READ query with data from the following: <table><tr><th>Entity</th><th>Attributes</th></tr><tr><td>Section</td><td>SectionName</td></tr></table>	Entity	Attributes	Section	SectionName	
	Entity	Attributes						
	Section	SectionName						
	Step 5: Administrator inputs the following Bicycle Part details: • Part Name • Image • Part Description • Bicycle Section							
	Step 6: The Administrator clicks the “Create Part” button. [alt]		Step 7: The system captures and ensures that all of the fields are filled in and an option is selected in the Combo box. [alt]					
		Step 8: The system validates that the captured details are in the correct format: [alt] <i>Part table</i> <table><tr><th>Attribute</th><th>Validation</th></tr><tr><td>PartName</td><td>32.Max 50 characters 33.Text characters only</td></tr><tr><td>PartDescription</td><td> • Max 50 characters • Text characters only </td></tr></table>	Attribute	Validation	PartName	32.Max 50 characters 33.Text characters only	PartDescription	• Max 50 characters • Text characters only
Attribute	Validation							
PartName	32.Max 50 characters 33.Text characters only							
PartDescription	• Max 50 characters • Text characters only							
		Step 9: The system verifies that no duplicate Part exists on the system by comparing the following: [alt] <table><tr><th>Input</th><th>Attribute</th><th>Entity</th></tr></table>	Input	Attribute	Entity			
Input	Attribute	Entity						

			Part Name	PartName	Part
			Step 10: The system prompts the Administrator to confirm to add the new Bicycle Part with the following:		
			Text/Label	Control Type	Additional Info
			Confirmation	Dialog Box Title	Used to indicate what type of message is being displayed.
			"Are you sure you want to add this Bicycle Part?"	Dialog Box Message	Used to inform the Administrator of the action.
			Yes	Button	Used to confirm the action.
			No	Button	Used to decline the action.

	Step 11: The Administrator clicks the "Yes" button. [alt]		Step 12: The Bicycle Part is assigned a unique PartID in the Part table which is set to the next consecutive number after the previous PartID in the database, the system then creates and saves the new Bicycle Part in the Part table. The same is done for the SectionPart association. The following attributes are saved in the database by using a FIRESQL query: INSERT Part Table PartID PartName PartDescription SectionPart Table SectionPartID SectionID PartID								
			Step 13: The system notifies the Administrator that the Bicycle Part has been added successfully to the database with the following: [alt] <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Success</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed.</td></tr><tr><td>"Bicycle Part has been added successfully"</td><td>Dialog Box Message</td><td>Used to inform the Administrator of the result.</td></tr></table>	Text/Label	Control Type	Additional Info	Success	Dialog Box Title	Used to indicate what type of message is being displayed.	"Bicycle Part has been added successfully"	Dialog Box Message
Text/Label	Control Type	Additional Info									
Success	Dialog Box Title	Used to indicate what type of message is being displayed.									
"Bicycle Part has been added successfully"	Dialog Box Message	Used to inform the Administrator of the result.									

			Ok	Button	Used to accept the result.									
			Step 14: The system creates and saves the following in the database by using a FIRESQL INSERT query: AuditTrail Table AuditID DateTime TransactionDescription UserID											
Alternate Courses:	Alt Step 6: The Administrator clicks the "Cancel" button. Terminate Use Case.													
	Alt Step 7a: In the case of an empty Text box, the system notifies the Administrator of the Input error below the input field with the following:													
	<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>"Please fill this field."</td><td>Label (with a light red background color fill)</td><td>Used to inform the Administrator of the result.</td></tr><tr><td>"X"</td><td>Icon</td><td>Used to indicate an error result.</td></tr></table>					Text/Label	Control Type	Additional Info	"Please fill this field."	Label (with a light red background color fill)	Used to inform the Administrator of the result.	"X"	Icon	Used to indicate an error result.
	Text/Label	Control Type	Additional Info											
	"Please fill this field."	Label (with a light red background color fill)	Used to inform the Administrator of the result.											
"X"	Icon	Used to indicate an error result.												
Redirect to Step 5.														
Alt Step 7b: In the case of a Combo box option not selected, the system notifies the Administrator of the Input error below the input field with the following:														
<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr></table>					Text/Label	Control Type	Additional Info							
Text/Label	Control Type	Additional Info												

	"Please select an option."	Label (with a light red background color fill)	Used to inform the Administrator of the result.
	"X"	Icon	Used to indicate an error result.
	Redirect to Step 5.		
	Alt Step 8: In the case of invalid input being entered, the system notifies the Administrator of the Format error below the input field with the following:		
	Text/Label	Control Type	Additional Info
	"Invalid input."	Label (with a light red background color fill)	Used to inform the Administrator of the result.
	"X"	Icon	Used to indicate an error result.
	Redirect to Step 5.		
	Alt Step 9: The inputted Part already exists in the system; the system will notify the Administrator of the duplication error with the following:		
	Text/Label	Control Type	Additional Info
	Error	Dialog Box Title	Used to indicate what type of message is being displayed.
	"The Bicycle Part already exists. Please try again."	Dialog Box Message	Used to inform the Administrator of the result.

	"Help icon"	Button	Used to Load the Help document.
	Ok	Button	Used to accept the result.
	Redirect to Step 5.		
	Alt Step 11: The Administrator declines to add the new Part by clicking the "No" button. Redirect to Step 5.		
	Alt Step 13: The Part was not added successfully to the database, the system will notify the Administrator of the save error with the following:		
	Text/Label	Control Type	Additional Info
	Failed	Dialog Box Title	Used to indicate what type of message is being displayed
	"Create unsuccessful. Please try again."	Dialog Box Message	Used to inform the Administrator of the result.
	"Help icon"	Button	Used to Load the Help document.
	Ok	Button	Used to accept the result.
	Redirect to Step 5.		
Conclusion:	The use case concludes when the Administrator has been notified that the Part has been added successfully.		
Post-Condition:	- A new Part is added to the system with the relevant details. - The AuditTrail table is successfully updated.		

	The Part and BicycleSection tables are successfully updated.
Business Rules	Only an Administrator can add a Part to the system.
Implementation Constraints and Specifications:	None
Assumptions:	None
Open Issues:	None

2.3.7.2. Search Bicycle Part

System Name:	VVs
Author:	Tapiwarufaro Ndoro
Date:	15/06/2020
Version:	1.0

Use Case Name:	Search Bicycle Part	Use Case Type:
Use Case ID:	6.2	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	Administrator	
Primary System Actor	None	
Other Participating Actors:	None	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process of an Administrator searching a Bicycle Part on the system. The Use Case starts when the Administrator navigates by selecting the "Bicycle Part Management" option under the "Bicycle Management" Combo box in the Navigation bar. The system populates the table with the information of all the bicycle parts.	
Pre-Condition:	The Administrator must be logged into the system.	
Trigger:	The Administrator requests to search for a Bicycle Part on the system.	

Typical Course Of Events:	Actor Action	System Response																			
		Manual Action	Automated Action																		
	Step 1: The Administrator selects the "Bicycle Part Management" option under the "Bicycle Management" Combo box in the Navigation bar.		Step 2: The system Displays the "Bicycle Part Management" screen and populates it with the following details:																		
			<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>" + icon"</td><td>Button</td><td>Used to load the "Bicycle Part" screen.</td></tr><tr><td>Search</td><td>Text-Box</td><td>Used for capturing Search criteria for the specific Bicycle Part.</td></tr><tr><td>Bicycle Part Management</td><td>Table</td><td>Displays a list of Bicycle Parts.</td></tr><tr><td>"Pen icon"</td><td>Button</td><td>Used to navigate to the specific Bicycle Part view.</td></tr><tr><td>Search</td><td>Button</td><td>Used to perform the search action</td></tr></table>	Text/Label	Control Type	Additional Info	" + icon"	Button	Used to load the "Bicycle Part" screen.	Search	Text-Box	Used for capturing Search criteria for the specific Bicycle Part.	Bicycle Part Management	Table	Displays a list of Bicycle Parts.	"Pen icon"	Button	Used to navigate to the specific Bicycle Part view.	Search	Button	Used to perform the search action
			Text/Label	Control Type	Additional Info																
			" + icon"	Button	Used to load the "Bicycle Part" screen.																
			Search	Text-Box	Used for capturing Search criteria for the specific Bicycle Part.																
			Bicycle Part Management	Table	Displays a list of Bicycle Parts.																
			"Pen icon"	Button	Used to navigate to the specific Bicycle Part view.																
			Search	Button	Used to perform the search action																
			The table is populated with data from the following by using a FIRESQL READ query:																		
			<table><tr><th>Entity</th><th>Attributes</th><th>Input Name</th></tr><tr><td>Part</td><td>PartName</td><td>Part Name</td></tr><tr><td>Part</td><td>PartDescription</td><td>Part Description</td></tr></table>	Entity	Attributes	Input Name	Part	PartName	Part Name	Part	PartDescription	Part Description									
Entity	Attributes	Input Name																			
Part	PartName	Part Name																			
Part	PartDescription	Part Description																			

			Section	SectionName	Bicycle Section											
	Step 3: The Administrator enters the search criteria that they wish to search by. They do so by typing in a keyword in the "Search" Text box then click the "Search" button.		Step 4: The system uses a FIRESQL READ query to READ the following details in the database: [alt] <table><tr><th>Entity</th><th>Attributes</th></tr><tr><td>Part</td><td>PartName</td></tr></table>			Entity	Attributes	Part	PartName							
	Entity	Attributes														
	Part	PartName														
		Step 5: The following details are loaded and used to populate the "Bicycle Part Management" table by using a FIRESQL READ query: <table><tr><th>Output</th><th>Attribute</th><th>Entity</th></tr><tr><td>Part Name</td><td>PartName</td><td>Part</td></tr><tr><td>Part Description</td><td>PartDescription</td><td>Part</td></tr><tr><td>Bicycle Section</td><td>SectionName</td><td>Section</td></tr></table>			Output	Attribute	Entity	Part Name	PartName	Part	Part Description	PartDescription	Part	Bicycle Section	SectionName	Section
Output	Attribute	Entity														
Part Name	PartName	Part														
Part Description	PartDescription	Part														
Bicycle Section	SectionName	Section														
Step 6: The Administrator clicks on the "Pen icon" Bicycle Part Button in the "Action" table column.		Step 7: The System will load the "Bicycle Part" screen and displays it with the following controls and layout: <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Navigation</td><td>Navigation-bar</td><td>Used for Navigation of the site.</td></tr><tr><td>Bicycle Part</td><td>Label</td><td>Heading for the pop-up displayed.</td></tr><tr><td>Choose Picture</td><td>Button</td><td>Facilitates uploading a picture.</td></tr></table>			Text/Label	Control Type	Additional Info	Navigation	Navigation-bar	Used for Navigation of the site.	Bicycle Part	Label	Heading for the pop-up displayed.	Choose Picture	Button	Facilitates uploading a picture.
Text/Label	Control Type	Additional Info														
Navigation	Navigation-bar	Used for Navigation of the site.														
Bicycle Part	Label	Heading for the pop-up displayed.														
Choose Picture	Button	Facilitates uploading a picture.														

			Part Picture	Picture image	Displays the Bicycle Part picture.
			Part Name	Label	Used to indicate what the corresponding text box is used for.
			Part Description	Label	Used to indicate what the corresponding text box is used for.
			Bicycle Section	Label	Used to indicate what the corresponding Combo box is used for.
			Part Name	Text box	Used to capture text for the Bicycle Part Name details.
			Part Description	Text box	Used to capture text for the Bicycle Part Description details.
			Bicycle Section	Combo box	Used to Select the Bicycle Section with this

			Part and is Populated by contents from the ERD with options as indicated in the table below.														
			Delete	Button	Used to delete a Bicycle Part.												
			Update	Button	Used to update a Bicycle Part.												
			The Bicycle Section Combo box options are populated with data from the following by using a FIRESQL READ query:														
			<table><tr><th>Entity</th><th>Attributes</th></tr><tr><td>Section</td><td>SectionName</td></tr></table>			Entity	Attributes	Section	SectionName								
Entity	Attributes																
Section	SectionName																
			The following fields are populated with data retrieved from the database where the PartID is the same as the one searched by using a FIRESQL READ query:														
			<table><tr><th>Output</th><th>Attribute</th><th>Entity</th></tr><tr><td>Part Name</td><td>PartName</td><td>Part</td></tr><tr><td>Part Description</td><td>PartDescription</td><td>Part</td></tr><tr><td>Bicycle Section</td><td>SectionName</td><td>Section</td></tr></table>			Output	Attribute	Entity	Part Name	PartName	Part	Part Description	PartDescription	Part	Bicycle Section	SectionName	Section
			Output	Attribute	Entity												
			Part Name	PartName	Part												
			Part Description	PartDescription	Part												
Bicycle Section	SectionName	Section															
Step 8: The system creates and saves the following in the database by using a FIRESQL INSERT query:																	
<table><tr><th>AuditTrail Table</th></tr></table>			AuditTrail Table														
AuditTrail Table																	

			AuditID DateTime TransactionDescription UserID									
Alternate Courses:	Alt Step 4: The specific Bicycle Part does not exist in the database. The system will notify the Administrator of this with the following: <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>"Bicycle Part not found."</td><td>Label</td><td>Used to inform the Administrator of the result.</td></tr><tr><td>"X"</td><td>Banner (with a light red background color fill)</td><td>Used to display the result message.</td></tr></table> Redirect to Step 3.			Text/Label	Control Type	Additional Info	"Bicycle Part not found."	Label	Used to inform the Administrator of the result.	"X"	Banner (with a light red background color fill)	Used to display the result message.
Text/Label	Control Type	Additional Info										
"Bicycle Part not found."	Label	Used to inform the Administrator of the result.										
"X"	Banner (with a light red background color fill)	Used to display the result message.										
Conclusion:	The use case narrative concludes when the searched Bicycle Part details are displayed.											
Post-Condition:	- The Part has been searched successfully. - The AuditTrail table is successfully updated.											
Business Rules	Only an Administrator can search for a Bicycle Part on the system.											
Implementation Constraints and Specifications:	None											
Assumptions:	None											

Open Issues:	• None

2.3.7.3. Update Bicycle Part

System Name:	VVs
Author:	Tapiwarufaro Ndoro
Date:	15/06/2020
Version:	1.0

Use Case Name:	Update Bicycle Part	Use Case Type:
Use Case ID:	6.3	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	Administrator	
Primary System Actor	None	
Other Participating Actors:	None	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process of an Administrator updating an existing Bicycle Part's details in the system. The Use Case starts when the Administrator navigates by selecting the "Bicycle Part Management" option under the "Bicycle Management" Combo box in the Navigation bar. From there, they search for the Bicycle Part they want to update and the system will load the "Bicycle Part" screen with the Bicycle Part details. The system populates the relevant Combo boxes with information of the searched Bicycle Part, and the Administrator can make their changes and save.	
Pre-Condition:	The Administrator must be logged into the system.	

Trigger:	The Administrator requests to update an existing Bicycle Part to the system.								
Typical Course Of Events:	Actor Action	System Response							
		Manual Action	Automated Action						
	Step 1: The Administrator selects the “Bicycle Part Management” option under the “Bicycle Management” Combo box in the Navigation bar.		Step 2: The System invokes Use Case 6.2 “Search Bicycle Part” and displays the “Bicycle Part” screen with the data of the searched Bicycle Part.						
	Step 3: Administrator updates the following Bicycle Part details: • Part Name • Part Description • Bicycle Section								
	Step 4: The Administrator clicks the “Update” button. [alt]		Step 5: The system captures and ensures that all of the fields are filled in and an option is selected in the Combo box. [alt]						
			Step 6: The system validates that the captured details are in the correct format: [alt] Part table <table><tr><th>Attribute</th><th>Validation</th></tr><tr><td>PartName</td><td>34.Max 50 characters 35.Text characters only</td></tr><tr><td>PartDescription</td><td> • Max 50 characters • Text characters only </td></tr></table>	Attribute	Validation	PartName	34.Max 50 characters 35.Text characters only	PartDescription	• Max 50 characters • Text characters only
	Attribute	Validation							
PartName	34.Max 50 characters 35.Text characters only								
PartDescription	• Max 50 characters • Text characters only								
		Step 7: The system prompts the Administrator to confirm the update of the Bicycle Part’s details.							

			Text/Label	Control Type	Additional Info
			Confirmation	Dialog Box Title	Used to indicate what type of message is being displayed.
			"Are you sure you want to update this Bicycle Part?"	Dialog Box Message	Used to inform the Administrator of the action.
			Yes	Button	Used to confirm the action.
			No	Button	Used to decline the action.

	Step 8: The Administrator clicks the "Yes" button. [alt]		Step 9: The system then updates and saves the updated Bicycle Part details in the Part table and the SectionPart associative table. The following attributes are updated in the database by using a FIRESQL UPDATE query: Part Table PartID PartName PartDescription SectionPart Table SectionPartID SectionID PartID								
			Step 10: The system notifies the Administrator that the Bicycle Part has been updated successfully in the database with the following: [alt] <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Success</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed.</td></tr><tr><td>"The Bicycle Part has been</td><td>Dialog Box Message</td><td>Used to inform the Administrator of the result.</td></tr></table>	Text/Label	Control Type	Additional Info	Success	Dialog Box Title	Used to indicate what type of message is being displayed.	"The Bicycle Part has been	Dialog Box Message
Text/Label	Control Type	Additional Info									
Success	Dialog Box Title	Used to indicate what type of message is being displayed.									
"The Bicycle Part has been	Dialog Box Message	Used to inform the Administrator of the result.									

			updated successfully”									
			OkButtonUsed to accept the result.									
			Step 11: The system updates and saves the following in the database by using a FIRESQL INSERT query: AuditTrail Table AuditID DateTime TransactionDescription UserID									
Alternate Courses:	Alt Step 4: The Administrator clicks the “Cancel” button. Terminate Use Case.											
	Alt Step 5a: In the case of an empty Text box, the system notifies the Administrator of the Input error below the input field with the following:											
	<table><tr><td>Text/Label</td><td>Control Type</td><td>Additional Info</td></tr><tr><td>“Please fill this field.”</td><td>Label (with a light red background color fill)</td><td>Used to inform the Administrator of the result.</td></tr><tr><td>“X”</td><td>Icon</td><td>Used to indicate an error result.</td></tr></table>			Text/Label	Control Type	Additional Info	“Please fill this field.”	Label (with a light red background color fill)	Used to inform the Administrator of the result.	“X”	Icon	Used to indicate an error result.
	Text/Label	Control Type	Additional Info									
“Please fill this field.”	Label (with a light red background color fill)	Used to inform the Administrator of the result.										
“X”	Icon	Used to indicate an error result.										
Redirect to Step 3.												
	Alt Step 5b: In the case of a Combo box option not selected, the system notifies the Administrator of the Input error below the input field with the following:											
	<table><tr><td>Text/Label</td><td>Control Type</td><td>Additional Info</td></tr></table>			Text/Label	Control Type	Additional Info						
Text/Label	Control Type	Additional Info										

	"Please select an option."	Label (with a light red background color fill)	Used to inform the Administrator of the result.
	"X"	Icon	Used to indicate an error result.
	Redirect to Step 3.		
	Alt Step 6: In the case of invalid input being entered, the system notifies the Administrator of the Format error below the input field with the following:		
	Text/Label	Control Type	Additional Info
	"Invalid input."	Label (with a light red background color fill)	Used to inform the Administrator of the result.
	"X"	Icon	Used to indicate an error result.
	Redirect to Step 3.		
	Alt Step 8: The Administrator declines to update the Bicycle Part by clicking the "No" button. Redirect to Step 4.		
	Alt Step 10: The Bicycle Part was not updated successfully in the database, the system will notify the Administrator of the update error with the following:		
	Text/Label	Control Type	Additional Info
	Failed	Dialog Box Title	Used to indicate what type of message is being displayed
	"Update unsuccessful. Please try again."	Dialog Box Message	Used to inform the Administrator of the result.

	"Help icon"	Button	Used to Load the Help document.
	Ok	Button	Used to accept the result.
	Redirect to Step 3.		
Conclusion:	The use case concludes when the Administrator has been notified that the Bicycle Part has been updated successfully.		
Post-Condition:	• An existing Bicycle Part is updated in the system with the relevant details. • The AuditTrail table is successfully updated. • The Part and BicycleSection tables are successfully updated.		
Business Rules	• Only an Administrator can update a Bicycle Part to the system.		
Implementation Constraints and Specifications:	• None		
Assumptions:	• None		
Open Issues:	• None		

2.3.7.4. Delete Bicycle Part

System Name:	VVs
Author:	Tapiwarufaro Ndoro
Date:	15/06/2020
Version:	1.0

Use Case Name:	Delete Bicycle Part	Use Case Type:
Use Case ID:	6.4	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	Administrator	
Primary System Actor	None	
Other Participating Actors:	None	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process of an Administrator deleting an existing Bicycle Part's details from the system. The Use Case starts when the Administrator navigates by selecting the "Bicycle Part Management" option under the "Bicycle Management" Combo box in the Navigation bar. From there, they search for the Bicycle Part they want to delete and the system displays the "Bicycle Part" screen with the Bicycle Part details to be deleted. The system populates the relevant Combo boxes with information of the searched Bicycle Part, and the Administrator can make their deletion.	
Pre-Condition:	The Administrator must be logged into the system.	

Trigger:	The Administrator requests to delete an existing Bicycle Part from the system.			
Typical Course Of Events:	Actor Action	System Response		
		Manual Action	Automated Action	
	Step 1: The Administrator selects the "Bicycle Part Management" option under the "Bicycle Management" Combo box in the Navigation bar.		Step 2: The System invokes Use Case 6.2 "Search Bicycle Part" and displays the "Bicycle Part" screen with the data of the searched Bicycle Part.	
	Step 3: The Administrator clicks on the "Delete" button. [alt]		Step 4: The system prompts the Administrator to confirm the deletion of the Bicycle Part's details with the following:	
			Text/Label	Control Type
Confirmation			Dialog Box Title	Used to indicate what type of message is being displayed.
"Are you sure you want to delete this Bicycle Part?"			Dialog Box Message	Used to inform the Administrator of the action.
Yes			Button	Used to confirm the action.
No	Button	Used to decline the action.		

	Step 5: The Administrator clicks the "Yes" button. [alt]		Step 6: The system then removes the Bicycle Part details in the Part table and the corresponding SectionPart. The following attributes are deleted in the database by using a FIRESQL DELETE query: Part Table PartID PartName PartDescription SectionPart Table SectionPartID SectionID PartID								
			Step 7: The system notifies the Administrator that the Bicycle Part has been deleted successfully from the database with the following: [alt] <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Success</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed.</td></tr><tr><td>"The Bicycle Part has been deleted successfully"</td><td>Dialog Box Message</td><td>Used to inform the Administrator of the result.</td></tr></table>	Text/Label	Control Type	Additional Info	Success	Dialog Box Title	Used to indicate what type of message is being displayed.	"The Bicycle Part has been deleted successfully"	Dialog Box Message
Text/Label	Control Type	Additional Info									
Success	Dialog Box Title	Used to indicate what type of message is being displayed.									
"The Bicycle Part has been deleted successfully"	Dialog Box Message	Used to inform the Administrator of the result.									

			Ok	Button	Used to accept the result.															
			Step 8: The system updates and saves the following in the database by using a FIRESQL INSERT query: AuditTrail Table AuditID DateTime TransactionDescription UserID																	
Alternate Courses:	Alt Step 3: The Administrator clicks the "Cancel" button. Terminate Use Case.																			
	Alt Step 5: The Administrator declines to delete the Bicycle Part by clicking the "No" button. Redirect to Step 2.																			
	Alt Step 7: The Administrator is notified of the unsuccessful delete action with the following:																			
	<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Failed</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed</td></tr><tr><td>"Delete unsuccessful. Please try again."</td><td>Dialog Box Message</td><td>Used to inform the Administrator of the result.</td></tr><tr><td>"Help icon"</td><td>Button</td><td>Used to Load the Help document.</td></tr><tr><td>Ok</td><td>Button</td><td>Used to accept the result.</td></tr></table>					Text/Label	Control Type	Additional Info	Failed	Dialog Box Title	Used to indicate what type of message is being displayed	"Delete unsuccessful. Please try again."	Dialog Box Message	Used to inform the Administrator of the result.	"Help icon"	Button	Used to Load the Help document.	Ok	Button	Used to accept the result.
	Text/Label	Control Type	Additional Info																	
	Failed	Dialog Box Title	Used to indicate what type of message is being displayed																	
	"Delete unsuccessful. Please try again."	Dialog Box Message	Used to inform the Administrator of the result.																	
"Help icon"	Button	Used to Load the Help document.																		
Ok	Button	Used to accept the result.																		

	Redirect to Step 2.
Conclusion:	The use case concludes when the Administrator has been notified that the Bicycle Part has been successfully deleted.
Post-Condition:	• An existing Bicycle Part is deleted from the system with all the relevant details being removed. • The AuditTrail table is successfully updated. • The Part and BicycleSection tables are successfully updated with the deleted details being removed.
Business Rules	• Only an Administrator can delete a Bicycle Part from the system.
Implementation Constraints and Specifications:	• None
Assumptions:	• None
Open Issues:	• None

2.3.7.5. Add Bicycle Section

System Name:	VVs
Author:	Tapiwarufaro Ndoro
Date:	15/06/2020
Version:	1.0

Use Case Name:	Add Bicycle Section	Use Case Type:
Use Case ID:	6.5	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	Administrator	
Primary System Actor	None	
Other Participating Actors:	None	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process of an Administrator adding a new Bicycle Section to the system. The Use Case starts when the Administrator navigates by selecting the "Bicycle Section Management" option under the "Bicycle Management" Combo box in the Navigation bar. From there, they click the "Add Bicycle Section" button, and the system Will load the "Bicycle Section" screen. The system populates the relevant Text boxes with information necessary placeholders, and the corresponding labels, buttons, icons, logos, and banners.	
Pre-Condition:	The Administrator must be logged into the system.	

Trigger:	The Administrator requests to add a new Bicycle Section to the system.																				
Typical Course Of Events:	Actor Action	System Response																			
		Manual Action	Automated Action																		
	Step 1: The Administrator selects the “Bicycle Section Management” option under the “Bicycle Management” Combo box in the Navigation bar.		Step 2: The system Displays the “Bicycle Section Management” screen and populates it with the following details:																		
			<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>+ icon</td><td>Button</td><td>Used to load the “Bicycle Section” screen.</td></tr><tr><td>Search</td><td>Text-Box</td><td>Used for Searching a specific Bicycle Section.</td></tr><tr><td>Bicycle Section Management</td><td>Table</td><td>Displays a list of Bicycle Sections.</td></tr><tr><td>Pen icon</td><td>Button</td><td>Used to navigate to the specific Bicycle Section view.</td></tr><tr><td>Search</td><td>Button</td><td>Used to perform the search action.</td></tr></table>	Text/Label	Control Type	Additional Info	+ icon	Button	Used to load the “Bicycle Section” screen.	Search	Text-Box	Used for Searching a specific Bicycle Section.	Bicycle Section Management	Table	Displays a list of Bicycle Sections.	Pen icon	Button	Used to navigate to the specific Bicycle Section view.	Search	Button	Used to perform the search action.
			Text/Label	Control Type	Additional Info																
			+ icon	Button	Used to load the “Bicycle Section” screen.																
			Search	Text-Box	Used for Searching a specific Bicycle Section.																
			Bicycle Section Management	Table	Displays a list of Bicycle Sections.																
			Pen icon	Button	Used to navigate to the specific Bicycle Section view.																
			Search	Button	Used to perform the search action.																
			The table is populated by using a FIRESQL READ query with data from the following:																		
			<table><tr><th>Entity</th><th>Attributes</th><th>Input Name</th></tr></table>	Entity	Attributes	Input Name															
Entity	Attributes	Input Name																			

			Section	SectionName	Section Name
			Section	SectionDescription	Section Description
	Step 3: The Administrator clicks on the "+ icon" button.		Step 4: The system will load the "Bicycle Section" screen and displays it with the following controls and layout:		
			Text/Label	Control Type	Additional Info
			Navigation	Navigation-bar	Used for Navigation of the site.
			Bicycle Section	Label	Heading for the pop-up displayed.
			Choose Picture	Button	Facilitates uploading a picture.
			Section Picture	Picture image	Displays the uploaded picture.
			Section Name	Label	Used to indicate what the corresponding text box is used for.
			Section Description	Label	Used to indicate what the corresponding text box is used for.

			<table><tr><td>Section Name</td><td>Text box</td><td>Used to capture text for the Bicycle Section Name details.</td></tr><tr><td>Section Description</td><td>Text box</td><td>Used to capture text for the Bicycle Section Description details.</td></tr><tr><td>Create Section</td><td>Button</td><td>Used to Save the new Bicycle Section Details.</td></tr><tr><td>Cancel</td><td>Button</td><td>Used to cancel the action of adding a Bicycle Section.</td></tr></table>	Section Name	Text box	Used to capture text for the Bicycle Section Name details.	Section Description	Text box	Used to capture text for the Bicycle Section Description details.	Create Section	Button	Used to Save the new Bicycle Section Details.	Cancel	Button	Used to cancel the action of adding a Bicycle Section.
	Section Name	Text box	Used to capture text for the Bicycle Section Name details.												
	Section Description	Text box	Used to capture text for the Bicycle Section Description details.												
	Create Section	Button	Used to Save the new Bicycle Section Details.												
	Cancel	Button	Used to cancel the action of adding a Bicycle Section.												
Step 5: Administrator inputs the following Bicycle Section details: - Section Name - Section Description															
Step 6: The Administrator clicks the "Create Section" button. [alt]		Step 7: The system captures and ensures that all of the fields are filled. [alt]													
		Step 8: The system validates that the captured details are in the correct format: [alt] Section table <table><tr><th>Attribute</th><th>Validation</th></tr><tr><td>SectionName</td><td>36.Max 150 characters 37.Text characters only</td></tr></table>	Attribute	Validation	SectionName	36.Max 150 characters 37.Text characters only									
Attribute	Validation														
SectionName	36.Max 150 characters 37.Text characters only														

			SectionDescription • Max 150 characters • Text characters only														
			Step 9: The system verifies that no duplicate Section exists on the system by comparing the following: [alt] <table><tr><th>Input</th><th>Attribute</th><th>Entity</th></tr><tr><td>Section Name</td><td>SectionName</td><td>Section</td></tr></table>	Input	Attribute	Entity	Section Name	SectionName	Section								
	Input	Attribute	Entity														
	Section Name	SectionName	Section														
			Step 10: The system prompts the Administrator to confirm to add the new Bicycle Section with the following: <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Confirmation</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed.</td></tr><tr><td>"Are you sure you want to add this Bicycle Section?"</td><td>Dialog Box Message</td><td>Used to inform the Administrator of the action.</td></tr><tr><td>Yes</td><td>Button</td><td>Used to confirm the action.</td></tr><tr><td>No</td><td>Button</td><td>Used to decline the action.</td></tr></table>	Text/Label	Control Type	Additional Info	Confirmation	Dialog Box Title	Used to indicate what type of message is being displayed.	"Are you sure you want to add this Bicycle Section?"	Dialog Box Message	Used to inform the Administrator of the action.	Yes	Button	Used to confirm the action.	No	Button
Text/Label	Control Type	Additional Info															
Confirmation	Dialog Box Title	Used to indicate what type of message is being displayed.															
"Are you sure you want to add this Bicycle Section?"	Dialog Box Message	Used to inform the Administrator of the action.															
Yes	Button	Used to confirm the action.															
No	Button	Used to decline the action.															

	Step 11: The Administrator clicks the "Yes" button. [alt]		Step 12: The Bicycle Section is assigned a unique SectionID in the Section table which is set to the next consecutive number after the previous SectionID in the database, the system then creates and saves the new Bicycle Section in the Section table. The following attributes are saved in the database by using a FIRESQL INSERT query: Section Table SectionID SectionName SectionDescription												
			Step 13: The system notifies the Administrator that the Bicycle Section has been added successfully to the database with the following: [alt] <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Success</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed.</td></tr><tr><td>"Bicycle Section has been added successfully"</td><td>Dialog Box Message</td><td>Used to inform the Administrator of the result.</td></tr><tr><td>Ok</td><td>Button</td><td>Used to accept the result.</td></tr></table>	Text/Label	Control Type	Additional Info	Success	Dialog Box Title	Used to indicate what type of message is being displayed.	"Bicycle Section has been added successfully"	Dialog Box Message	Used to inform the Administrator of the result.	Ok	Button	Used to accept the result.
	Text/Label	Control Type	Additional Info												
Success	Dialog Box Title	Used to indicate what type of message is being displayed.													
"Bicycle Section has been added successfully"	Dialog Box Message	Used to inform the Administrator of the result.													
Ok	Button	Used to accept the result.													
		Step 14: The system creates and saves the following in the database by using a FIRESQL INSERT query: AuditTrail Table													

			AuditID
			DateTime
			TransactionDescription
			UserID
Alternate Courses:	Alt Step 6: The Administrator clicks the "Cancel" button. Terminate Use Case.		
	Alt Step 7: In the case of an empty Text box, the system notifies the Administrator of the Input error below the input field with the following:		
	Text/Label	Control Type	Additional Info
	"Please fill this field."	Label (with a light red background color fill)	Used to inform the Administrator of the result.
	"X"	Icon	Used to indicate an error result.
	Redirect to Step 5.		
	Alt Step 8: In the case of invalid input being entered, the system notifies the Administrator of the Format error below the input field with the following:		
	Text/Label	Control Type	Additional Info
	"Invalid input."	Label (with a light red background color fill)	Used to inform the Administrator of the result.
	"X"	Icon	Used to indicate an error result.
	Redirect to Step 5.		

Alt Step 9: The inputted Section already exists in the system; the system will notify the Administrator of the duplication error with the following:

Text/Label	Control Type	Additional Info
Error	Dialog Box Title	Used to indicate what type of message is being displayed.
"The Bicycle Section already exists. Please try again."	Dialog Box Message	Used to inform the Administrator of the result.
"Help icon"	Button	Used to Load the Help document.
Ok	Button	Used to accept the result.

Redirect to Step 5.

Alt Step 11: The Administrator declines to add the new Section by clicking the "No" button. Redirect to Step 5.

Alt Step 13: The Section was not added successfully to the database, the system will notify the Administrator of the save error with the following:

Text/Label	Control Type	Additional Info
Failed	Dialog Box Title	Used to indicate what type of message is being displayed

	"Create unsuccessful. Please try again."	Dialog Box Message	Used to inform the Administrator of the result.
	"Help icon"	Button	Used to Load the Help document.
	Ok	Button	Used to accept the result.
	Redirect to Step 5.		
Conclusion:	The use case concludes when the Administrator has been notified that the Section has been added successfully.		
Post-Condition:	• A new Section is added to the system with the relevant details. • The AuditTrail table is successfully updated. • The Section table is successfully updated.		
Business Rules	• Only an Administrator can add a Section to the system.		
Implementation Constraints and Specifications:	• None		
Assumptions:	• None		
Open Issues:	• None		

2.3.7.6. Search Bicycle Section

System Name:	VVs
Author:	Tapiwarufaro Ndoro
Date:	15/06/2020
Version:	1.0

Use Case Name:	Search Bicycle Section	Use Case Type:
Use Case ID:	6.6	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	Administrator	
Primary System Actor	None	
Other Participating Actors:	None	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process of an Administrator searching a Bicycle Section on the system. The Use Case starts when the Administrator navigates by selecting the "Bicycle Section Management" option under the "Bicycle Management" Combo box in the Navigation bar. The system populates the table with the information of all the bicycle Sections.	
Pre-Condition:	The Administrator must be logged into the system.	

Trigger:	The Administrator requests to search for a Bicycle Section on the system.																				
Typical Course Of Events:	Actor Action	System Response																			
		Manual Action	Automated Action																		
	Step 1: The Administrator selects the "Bicycle Section Management" option under the "Bicycle Management" Combo box in the Navigation bar.		Step 2: The system Displays the "Bicycle Section Management" screen and populates it with the following details:																		
			<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>" + icon"</td><td>Button</td><td>Used to load the "Bicycle Section" screen.</td></tr><tr><td>Search</td><td>Text-Box</td><td>Used for Searching a specific Bicycle Section.</td></tr><tr><td>Bicycle Section Management</td><td>Table</td><td>Displays a list of Bicycle Sections.</td></tr><tr><td>"Pen icon"</td><td>Button</td><td>Used to navigate to the specific Bicycle Section view.</td></tr><tr><td>"Search"</td><td>Button</td><td>Used to perform the search action.</td></tr></table>	Text/Label	Control Type	Additional Info	" + icon"	Button	Used to load the "Bicycle Section" screen.	Search	Text-Box	Used for Searching a specific Bicycle Section.	Bicycle Section Management	Table	Displays a list of Bicycle Sections.	"Pen icon"	Button	Used to navigate to the specific Bicycle Section view.	"Search"	Button	Used to perform the search action.
			Text/Label	Control Type	Additional Info																
			" + icon"	Button	Used to load the "Bicycle Section" screen.																
			Search	Text-Box	Used for Searching a specific Bicycle Section.																
			Bicycle Section Management	Table	Displays a list of Bicycle Sections.																
			"Pen icon"	Button	Used to navigate to the specific Bicycle Section view.																
			"Search"	Button	Used to perform the search action.																
			The table is populated with data from the following by using a FIRESQL READ query:																		
			<table><tr><th>Entity</th><th>Attributes</th><th>Input Name</th></tr></table>	Entity	Attributes	Input Name															
Entity	Attributes	Input Name																			

			<table><tr><td>Section</td><td>SectionName</td><td>Section Name</td></tr><tr><td>Section</td><td>SectionDescription</td><td>Section Description</td></tr></table>	Section	SectionName	Section Name	Section	SectionDescription	Section Description		
	Section	SectionName	Section Name								
	Section	SectionDescription	Section Description								
	Step 3: The Administrator enters the search criteria that they wish to search by. They do so by typing in a keyword in the "Search" Text box then click the "Search" button.		Step 4: The system uses a FIRESQL READ query to READ the following details in the database: [alt] <table><tr><td>Entity</td><td>Attributes</td></tr><tr><td>Section</td><td>SectionName</td></tr></table>	Entity	Attributes	Section	SectionName				
	Entity	Attributes									
Section	SectionName										
		Step 5: The following details are loaded and used to populate the "Bicycle Section Management" table by using a FIRESQL READ query: <table><tr><td>Output</td><td>Attribute</td><td>Entity</td></tr><tr><td>Section Name</td><td>SectionName</td><td>Section</td></tr><tr><td>Section Description</td><td>SectionDescription</td><td>Section</td></tr></table>	Output	Attribute	Entity	Section Name	SectionName	Section	Section Description	SectionDescription	Section
Output	Attribute	Entity									
Section Name	SectionName	Section									
Section Description	SectionDescription	Section									
Step 6: The Administrator clicks on the "Pen icon" Bicycle Section Button in the "Action" table column.		Step 7: The System will load the "Bicycle Section" screen and displays it with the following controls and layout: <table><tr><td>Text/Label</td><td>Control Type</td><td>Additional Info</td></tr><tr><td>Navigation</td><td>Navigation-bar</td><td>Used for Navigation of the site.</td></tr><tr><td>Bicycle Section</td><td>Label</td><td>Heading for the pop-up displayed.</td></tr></table>	Text/Label	Control Type	Additional Info	Navigation	Navigation-bar	Used for Navigation of the site.	Bicycle Section	Label	Heading for the pop-up displayed.
Text/Label	Control Type	Additional Info									
Navigation	Navigation-bar	Used for Navigation of the site.									
Bicycle Section	Label	Heading for the pop-up displayed.									

			Choose Picture	Button	Facilitates uploading a picture.
			Section Picture	Picture image	Displays the Bicycle Section picture.
			Section Name	Label	Used to indicate what the corresponding text box is used for.
			Section Description	Label	Used to indicate what the corresponding text box is used for.
			Section Name	Text box	Used to capture text for the Bicycle Section Name details.
			Section Description	Text box	Used to capture text for the Bicycle Section Description details.
			Delete	Button	Used to delete a Bicycle Section.
			Update	Button	Used to update a Bicycle Section.

			The following fields are populated with data retrieved from the database where the SectionID is the same as the one searched by using a FIRESQL READ query: <table><tr><th>Output</th><th>Attribute</th><th>Entity</th></tr><tr><td>Section Name</td><td>SectionName</td><td>Section</td></tr><tr><td>Section Description</td><td>SectionDescription</td><td>Section</td></tr></table>	Output	Attribute	Entity	Section Name	SectionName	Section	Section Description	SectionDescription	Section
	Output	Attribute	Entity									
Section Name	SectionName	Section										
Section Description	SectionDescription	Section										
			Step 8: The system creates and saves the following in the database by using a FIRESQL INSERT query: AuditTrail Table AuditID DateTime TransactionDescription UserID									
Alternate Courses:	Alt Step 4: The specific Bicycle Section does not exist in the database. The system will notify the Administrator of this with the following: <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>"Bicycle Section not found."</td><td>Label</td><td>Used to inform the Administrator of the result.</td></tr><tr><td>"X"</td><td>Banner (with a light red background color fill)</td><td>Used to display the result message.</td></tr></table>			Text/Label	Control Type	Additional Info	"Bicycle Section not found."	Label	Used to inform the Administrator of the result.	"X"	Banner (with a light red background color fill)	Used to display the result message.
	Text/Label	Control Type	Additional Info									
	"Bicycle Section not found."	Label	Used to inform the Administrator of the result.									
"X"	Banner (with a light red background color fill)	Used to display the result message.										
	Redirect to Step 3.											

Conclusion:	The use case narrative concludes when the searched Bicycle Section details are displayed.
Post-Condition:	• The Section has been searched successfully. • The AuditTrail table is successfully updated.
Business Rules	• Only an Administrator can search for a Bicycle Section on the system.
Implementation Constraints and Specifications:	• None
Assumptions:	• None
Open Issues:	• None

2.3.7.7. Update Bicycle Section

System Name:	VVs
Author:	Tapiwarufaro Ndoro
Date:	15/06/2020
Version:	1.0

Use Case Name:	Update Bicycle Section	Use Case Type:
Use Case ID:	6.7	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	Administrator	
Primary System Actor	None	
Other Participating Actors:	None	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process of an Administrator updating an existing Bicycle Section's details in the system. The Use Case starts when the Administrator navigates by selecting the "Bicycle Section Management" option under the "Bicycle Management" Combo box in the Navigation bar. From there, they search for the Bicycle Section they want to update, and the system will load the "Bicycle Section" screen with the Bicycle Section details. The system populates the relevant Text boxes with information of the searched Bicycle Section, and the Administrator can make their changes and save.	
Pre-Condition:	The Administrator must be logged into the system.	

Trigger:	The Administrator requests to update an existing Bicycle Section to the system.								
Typical Course Of Events:	Actor Action	System Response							
		Manual Action	Automated Action						
	Step 1: The Administrator selects the "Bicycle Section Management" option under the "Bicycle Management" Combo box in the Navigation bar.		Step 2: The System invokes Use Case 6.6 "Search Bicycle Section" and displays the "Bicycle Section" screen with the data of the searched Bicycle Section.						
	Step 3: Administrator updates the following Bicycle Section details: • Section Name • Section Description								
	Step 4: The Administrator clicks the "Update" button. [alt]		Step 5: The system captures and ensures that all of the fields are filled. [alt]						
			Step 6: The system validates that the captured details are in the correct format: [alt] Section table <table><tr><th>Attribute</th><th>Validation</th></tr><tr><td>SectionName</td><td>38.Max 50 characters 39.Text characters only</td></tr><tr><td>SectionDescription</td><td> • Max 50 characters • Text characters only </td></tr></table>	Attribute	Validation	SectionName	38.Max 50 characters 39.Text characters only	SectionDescription	• Max 50 characters • Text characters only
	Attribute	Validation							
SectionName	38.Max 50 characters 39.Text characters only								
SectionDescription	• Max 50 characters • Text characters only								
		Step 7: The system prompts the Administrator to confirm the update of the Bicycle Section's details. <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr></table>	Text/Label	Control Type	Additional Info				
Text/Label	Control Type	Additional Info							

			Confirmation	Dialog Box Title	Used to indicate what type of message is being displayed.
			"Are you sure you want to update this Bicycle Section?"	Dialog Box Message	Used to inform the Administrator of the action.
			Yes	Button	Used to confirm the action.
			No	Button	Used to decline the action.
	Step 8: The Administrator clicks the "Yes" button. [alt]		Step 9: The system then updates and saves the updated Bicycle Section details in the Section table. The following attributes are updated in the database by using a FIRESQL UPDATE query: Section Table SectionID SectionName SectionDescription		
		Step 10: The system notifies the Administrator that the Bicycle Section has been updated successfully in the database with the following: [alt]			
			Text/Label	Control Type	Additional Info
			Success	Dialog Box Title	Used to indicate what type of

			message is being displayed.								
			<table><tr><td>"The Bicycle Section has been updated successfully"</td><td>Dialog Box Message</td><td>Used to inform the Administrator of the result.</td></tr><tr><td>Ok</td><td>Button</td><td>Used to accept the result.</td></tr></table>	"The Bicycle Section has been updated successfully"	Dialog Box Message	Used to inform the Administrator of the result.	Ok	Button	Used to accept the result.		
	"The Bicycle Section has been updated successfully"	Dialog Box Message	Used to inform the Administrator of the result.								
Ok	Button	Used to accept the result.									
		Step 11: The system updates and saves the following in the database by using a FIRESQL INSERT query: <table><tr><td>AuditTrail Table</td></tr><tr><td>AuditID</td></tr><tr><td>DateTime</td></tr><tr><td>TransactionDescription</td></tr><tr><td>UserID</td></tr></table>		AuditTrail Table	AuditID	DateTime	TransactionDescription	UserID			
AuditTrail Table											
AuditID											
DateTime											
TransactionDescription											
UserID											
Alternate Courses:	Alt Step 4: The Administrator clicks the "Cancel" button. Terminate Use Case.										
	Alt Step 5: In the case of an empty Text box, the system notifies the Administrator of the Input error below the input field with the following:										
	<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>"Please fill this field."</td><td>Label (with a light red background color fill)</td><td>Used to inform the Administrator of the result.</td></tr><tr><td>"X"</td><td>Icon</td><td>Used to indicate an error result.</td></tr></table>	Text/Label	Control Type	Additional Info	"Please fill this field."	Label (with a light red background color fill)	Used to inform the Administrator of the result.	"X"	Icon	Used to indicate an error result.	
Text/Label	Control Type	Additional Info									
"Please fill this field."	Label (with a light red background color fill)	Used to inform the Administrator of the result.									
"X"	Icon	Used to indicate an error result.									

	Redirect to Step 3.		
	Alt Step 6: In the case of invalid input being entered, the system notifies the Administrator of the Format error below the input field with the following:		
	Text/Label	Control Type	Additional Info
	"Invalid input."	Label (with a light red background color fill)	Used to inform the Administrator of the result.
	"X"	Icon	Used to indicate an error result.
	Redirect to Step 3.		
	Alt Step 8: The Administrator declines to update the Bicycle Section by clicking the "No" button. Redirect to Step 4.		
	Alt Step 10: The Bicycle Section was not updated successfully in the database; the system will notify the Administrator of the update error with the following:		
	Text/Label	Control Type	Additional Info
	Failed	Dialog Box Title	Used to indicate what type of message is being displayed
	"Update unsuccessful. Please try again."	Dialog Box Message	Used to inform the Administrator of the result.
	"Help icon"	Button	Used to Load the Help document.
	Ok	Button	Used to accept the result.

	Redirect to Step 3.
Conclusion:	The use case concludes when the Administrator has been notified that the Bicycle Section has been updated successfully.
Post-Condition:	• An existing Bicycle Section is updated in the system with the relevant details. • The AuditTrail table is successfully updated. • The Section table is successfully updated.
Business Rules	• Only an Administrator can update a Bicycle Section in the system.
Implementation Constraints and Specifications:	• None
Assumptions:	• None
Open Issues:	• None

2.3.7.8. Delete Bicycle Section

System Name:	VVs
Author:	Tapiwarufaro Ndoro
Date:	15/06/2020
Version:	1.0

Use Case Name:	Delete Bicycle Section	Use Case Type:
Use Case ID:	6.8	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	Administrator	
Primary System Actor	None	
Other Participating Actors:	None	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process of an Administrator deleting an existing Bicycle Section's details from the system. The Use Case starts when the Administrator navigates by selecting the "Bicycle Section Management" option under the "Bicycle Management" Combo box in the Navigation bar. From there, they search for the Bicycle Section they want to delete, and the system displays the "Bicycle Section" screen with the Bicycle Section details to be deleted. The system populates the relevant Text boxes with information of the searched Bicycle Section, and the Administrator can make their deletion.	
Pre-Condition:	The Administrator must be logged into the system.	

Trigger:	The Administrator requests to delete an existing Bicycle Section from the system.			
Typical Course Of Events:	Actor Action	System Response		
		Manual Action	Automated Action	
	Step 1: The Administrator selects the "Bicycle Section Management" option under the "Bicycle Management" Combo box in the Navigation bar.		Step 2: The System invokes Use Case 6.6 "Search Bicycle Section" and displays the "Bicycle Section" screen with the data of the searched Bicycle Section.	
	Step 3: The Administrator clicks on the "Delete" button. [alt]		Step 4: The system prompts the Administrator to confirm the deletion of the Bicycle Section's details with the following:	
			Text/Label	Control Type
Confirmation			Dialog Box Title	Used to indicate what type of message is being displayed.
"Are you sure you want to delete this Bicycle Section?"			Dialog Box Message	Used to inform the Administrator of the action.
Yes			Button	Used to confirm the action.
No	Button	Used to decline the action.		

	Step 5: The Administrator clicks the "Yes" button. [alt]		Step 6: The system then removes the Bicycle Section details in the Section table. The following attributes are deleted in the database by using a FIRESQL DELETE query: Section Table SectionID SectionName SectionDescription												
			Step 7: The system notifies the Administrator that the Bicycle Section has been deleted successfully from the database with the following: [alt] <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Success</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed.</td></tr><tr><td>"The Bicycle Section has been deleted successfully"</td><td>Dialog Box Message</td><td>Used to inform the Administrator of the result.</td></tr><tr><td>Ok</td><td>Button</td><td>Used to accept the result.</td></tr></table>	Text/Label	Control Type	Additional Info	Success	Dialog Box Title	Used to indicate what type of message is being displayed.	"The Bicycle Section has been deleted successfully"	Dialog Box Message	Used to inform the Administrator of the result.	Ok	Button	Used to accept the result.
	Text/Label	Control Type	Additional Info												
Success	Dialog Box Title	Used to indicate what type of message is being displayed.													
"The Bicycle Section has been deleted successfully"	Dialog Box Message	Used to inform the Administrator of the result.													
Ok	Button	Used to accept the result.													
		Step 8: The system updates and saves the following in the database by using a FIRESQL INSERT query: AuditTrail Table AuditID													

			DateTime TransactionDescription UserID
Alternate Courses:	Alt Step 3: The Administrator clicks the "Cancel" button. Terminate Use Case.		
	Alt Step 5: The Administrator declines to delete the Bicycle Section by clicking the "No" button. Redirect to Step 2.		
	Alt Step 7: The Administrator is notified of the unsuccessful delete action with the following:		
	Text/Label	Control Type	Additional Info
	Failed	Dialog Box Title	Used to indicate what type of message is being displayed
	"Delete unsuccessful. Please try again."	Dialog Box Message	Used to inform the Administrator of the result.
	"Help icon"	Button	Used to Load the Help document.
	Ok	Button	Used to accept the result.
	Redirect to Step 2.		
Conclusion:	The use case concludes when the Administrator has been notified that the Bicycle Section has been successfully deleted.		
Post-Condition:	- An existing Bicycle Section is deleted from the system with all the relevant details being removed. - The AuditTrail table is successfully updated.		

	The Section table is successfully updated with the deleted details being removed.
Business Rules	Only an Administrator can delete a Bicycle Section from the system.
Implementation Constraints and Specifications:	None
Assumptions:	None
Open Issues:	None

2.3.8. Sub-system 7: Tracking

2.3.8.1. Add Log Entry

System Name:	VVs
Author:	Gareth Phillips
Date:	15/06/2020
Version:	1.0

Use Case Name:	Add Tracking	Use Case Type:
Use Case ID:	7.1	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS User Requirements	System Design: ☒
Primary Business Actor	Student	
Primary System Actor	None	
Other Participating Actors:	None	
Other Interested Stakeholders:	Administrator	
Description:	This Technical Use Case Narrative describes the process whereby a student passes through an area that is within the antenna range which in turn creates an entry in the log. The use case concludes when the log entry has been successfully added.	
Pre-Condition:	The Tag is registered on the system.	

Trigger:	The Tag is within the Antenna range.		
Typical Course Of Events:	Actor Action	System Response	
		Manual Action	Automated Action
	Step 1: The student passes through an area that is within an antenna's area with their tag.		Step 2: The Antenna captures the Tags details Attribute TagID DateTime
			Step 3: The system captures and saves the following attributes in the database by using a FIRESQL INSERT query: [alt] Log Table TagID AntennalID DateTime
			Step 4: The system creates and saves the following in the database by using a FIRESQL INSERT query: AuditTrail Table AuditTrailID DateTime TransactionDescription UserID
Alternate Courses:	Alt Step 3: Tag READ error. Terminate Use Case.		

Conclusion:	The use case concludes once a log entry has been successfully saved to the database.
Post-Condition:	• A new log entry has been successfully added to the system. • The AuditTrail table is successfully updated. • The Log table is successfully updated.
Business Rules	• Tag must be in the range of an antenna.
Implementation Constraints and Specifications:	• None
Assumptions:	• The antenna and system are connected.
Open Issues:	• None

2.3.8.2. Search Log Entry

System Name:	VVs
Author:	Gareth Phillips
Date:	15/06/2020
Version:	1.0

Use Case Name:	Search Log	Use Case Type:
Use Case ID:	7.2	Business Requirements: ☐
Priority:	Medium	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	Administrator	
Primary System Actor	None	
Other Participating Actors:	None	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process of an Administrator searching a Log Entry on the system. The Use Case starts when the Administrator navigates by selecting the " Log Management" option under the "System Management" Combo box in the Navigation bar. The system populates the table with the information of all the Maintenance Task.	
Pre-Condition:	The Administrator must be logged into the system.	
Trigger:	The Administrator requests to search for a Log Entry on the system.	

Typical Course Of Events:	Actor Action	System Response																	
		Manual Action	Automated Action																
	Step 1: The Administrator selects the “Log Management” option under the “System Management” Combo box in the Navigation bar.		Step 2: The system Displays the “Log Management” screen and populates it with the following details:																
			<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Search</td><td>Text-Box</td><td>Used for capturing Search criteria for the specific Log Entry.</td></tr><tr><td>Log Management</td><td>Table</td><td>Displays a list of Log Entries.</td></tr><tr><td>“Back”</td><td>Button</td><td>Returns to Home Page.</td></tr><tr><td>“Help Icon”</td><td>Button</td><td>Used to Load the Help document.</td></tr></table>		Text/Label	Control Type	Additional Info	Search	Text-Box	Used for capturing Search criteria for the specific Log Entry.	Log Management	Table	Displays a list of Log Entries.	“Back”	Button	Returns to Home Page.	“Help Icon”	Button	Used to Load the Help document.
			Text/Label	Control Type	Additional Info														
			Search	Text-Box	Used for capturing Search criteria for the specific Log Entry.														
			Log Management	Table	Displays a list of Log Entries.														
			“Back”	Button	Returns to Home Page.														
			“Help Icon”	Button	Used to Load the Help document.														
			The table is populated by using a FIRESQL READ query with data from the following:																
			<table><tr><th>Entity</th><th>Attributes</th><th>Input Name</th></tr><tr><td>Log</td><td>TagID</td><td>TagID</td></tr><tr><td>Log</td><td>BicycleID</td><td>BicycleID</td></tr><tr><td>Log</td><td>StudentID</td><td>StudentID</td></tr><tr><td>Log</td><td>DateTime</td><td>Time</td></tr></table>		Entity	Attributes	Input Name	Log	TagID	TagID	Log	BicycleID	BicycleID	Log	StudentID	StudentID	Log	DateTime	Time
			Entity	Attributes	Input Name														
			Log	TagID	TagID														
			Log	BicycleID	BicycleID														
Log	StudentID	StudentID																	
Log	DateTime	Time																	

			Log	DateTime	Date																	
	Step 3: The Administrator enters the search criteria that they wish to search by. They do so by typing in a keyword in the "Search" Text box.																					
	Step 4: The Administrator clicks the "Search" button.		Step 5: The system uses a FIRESQL READ query to READ the following details in the database: [alt] Log Table TagID BicycleID StudentID DateTime																			
			Step 5: The following details are loaded and used to populate the "Log Management" table by using a FIRESQL READ query: <table><tr><th>Output</th><th>Attribute</th><th>Entity</th></tr><tr><td>TagID</td><td>TagID</td><td>Log</td></tr><tr><td>BicycleID</td><td>BicycleID</td><td>Log</td></tr><tr><td>StudentID</td><td>StudentID</td><td>Log</td></tr><tr><td>Time</td><td>DateTime</td><td>Log</td></tr><tr><td>Date</td><td>DateTime</td><td>Log</td></tr></table>			Output	Attribute	Entity	TagID	TagID	Log	BicycleID	BicycleID	Log	StudentID	StudentID	Log	Time	DateTime	Log	Date	DateTime
Output	Attribute	Entity																				
TagID	TagID	Log																				
BicycleID	BicycleID	Log																				
StudentID	StudentID	Log																				
Time	DateTime	Log																				
Date	DateTime	Log																				

			Step 6: The system creates and saves the following in the database by using a FIRESQL INSERT query: AuditTrail Table AuditTrailID DateTime TransactionDescription UserID									
Alternate Courses:	Alt Step 4: The specific Log Entry does not exist in the database. The system will notify the Administrator of this with the following: <table><thead><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr></thead><tbody><tr><td>"Log Entry not found."</td><td>Label</td><td>Used to inform the Administrator of the result.</td></tr><tr><td>"X"</td><td>Banner (with a light red background color fill)</td><td>Used to display the result message.</td></tr></tbody></table> Redirect to Step 3.			Text/Label	Control Type	Additional Info	"Log Entry not found."	Label	Used to inform the Administrator of the result.	"X"	Banner (with a light red background color fill)	Used to display the result message.
Text/Label	Control Type	Additional Info										
"Log Entry not found."	Label	Used to inform the Administrator of the result.										
"X"	Banner (with a light red background color fill)	Used to display the result message.										
Conclusion:	The use case narrative concludes when the searched Log Entry details are displayed.											
Post-Condition:	- The Log Details has been searched successfully. - The AuditTrail table is successfully updated.											
Business Rules	Only an Administrator can search for a Log Entry on the system.											

Implementation Constraints and Specifications:	• None
Assumptions:	• None
Open Issues:	• None

2.3.8.3. Add Route

System Name:	VVs
Author:	Hansen Li
Date:	18/06/2020
Version:	1.1

Use Case Name:	Add Route	Use Case Type:
Use Case ID:	7.3	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	Administrator	
Primary System Actor	None	
Other Participating Actors:	None	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process of an Administrator adding a new Route to the system. The Use Case starts when the Administrator navigates by selecting the "Route Management" option under the "Systems Management" Combo box in the Navigation bar. From there, they click the "Add Route" button and the system will load the "Route" screen. The system populates the relevant Combo boxes with information necessary for selection, and the corresponding labels, text boxes and buttons, icons, logos, and banners.	
Pre-Condition:	The Administrator must be logged into the system.	

Trigger:	The Administrator requests to add a new Route to the system.																	
Typical Course Of Events:	Actor Action	System Response																
		Manual Action	Automated Action															
	Step 1: The Administrator selects the “Route Management” option under the “Systems Management” Combo box in the Navigation bar.		Step 2: The system Displays the “Route Management” screen and populates it with the following details:															
			<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Add Route</td><td>Button</td><td>Used to load the “Route” screen.</td></tr><tr><td>Search</td><td>Text-Box</td><td>Used for capturing Search criteria for the specific Route.</td></tr><tr><td>Route Management</td><td>Table</td><td>Displays a list of Routes.</td></tr><tr><td>“Pen icon”</td><td>Button</td><td>Used to navigate to the specific Route view.</td></tr></table>	Text/Label	Control Type	Additional Info	Add Route	Button	Used to load the “Route” screen.	Search	Text-Box	Used for capturing Search criteria for the specific Route.	Route Management	Table	Displays a list of Routes.	“Pen icon”	Button	Used to navigate to the specific Route view.
			Text/Label	Control Type	Additional Info													
			Add Route	Button	Used to load the “Route” screen.													
			Search	Text-Box	Used for capturing Search criteria for the specific Route.													
			Route Management	Table	Displays a list of Routes.													
			“Pen icon”	Button	Used to navigate to the specific Route view.													
			The table is populated by using a FIRESQL READ query with data from the following:															
<table><tr><th>Entity</th><th>Attributes</th><th>Input Name</th></tr><tr><td>Route</td><td>RouteID</td><td>Route ID</td></tr><tr><td>School</td><td>SchoolName</td><td>School Name</td></tr></table>			Entity	Attributes	Input Name	Route	RouteID	Route ID	School	SchoolName	School Name							
Entity	Attributes	Input Name																
Route	RouteID	Route ID																
School	SchoolName	School Name																
Step 3: The Administrator clicks on the “Add Route” button.		Step 4: The system will load the “Route” screen and displays it with the following controls and layout:																

			Text/Label	Control Type	Additional Info
			Menu	Menu-icon	Used to Navigate the side menu-bar.
			Route	Label	Heading for the pop-up displayed.
			School	Label	Used to indicate what the corresponding text box is used for.
			Antenna	Label	Heading to display which antenna you are adding
			Antenna position	Label	Used to indicate what the corresponding text box is used for.
			Antenna ID	Label	Used to indicate what the corresponding text box is used for.
			School	Combo box	Used to select the school

			Antenna Position	Text box	Used to Display the Antenna position					
			Antenna ID	Combo box	Used to select the Antenna					
			Add Antenna	Button	Used to create another antenna for this Route					
			Create Route	Button	Used to Save the new Route Details.					
			Cancel	Button	Used to cancel the action of adding a Route					
				The Route Combo box options are populated by using a FIRESQL READ query with data from the following:						
				<table><tr><th>Entity</th><th>Attributes</th></tr><tr><td>School</td><td>SchoolName</td></tr></table>			Entity	Attributes	School	SchoolName
	Entity	Attributes								
	School	SchoolName								
				<table><tr><th>Entity</th><th>Attributes</th></tr><tr><td>Antenna</td><td>AntennaID</td></tr></table>			Entity	Attributes	Antenna	AntennaID
Entity	Attributes									
Antenna	AntennaID									
Step 5: Administrator inputs the following Antenna details:										
- School - Antenna Position										

	Antenna ID										
	Step 6: The Administrator clicks the "Create Route" button. [alt]		Step 7: The system captures and ensures that all of the fields are filled in and an option is selected in the Combo boxes. [alt]								
			Step 8: The system validates that the captured details are in the correct format: [alt] Route table<table><tr><th>Attribute</th><th>Validation</th></tr><tr><td>School</td><td> - Option must be selected </td></tr><tr><td>AntennaID</td><td> - Option must be selected </td></tr></table>	Attribute	Validation	School	Option must be selected	AntennaID	Option must be selected		
	Attribute	Validation									
	School	Option must be selected									
AntennaID	Option must be selected										
		Step 9: The system verifies that no duplicate Route exists on the system by comparing the following: [alt] <table><tr><th>Input</th><th>Attribute</th><th>Entity</th></tr><tr><td>Route ID</td><td>RouteID</td><td>Antenna</td></tr></table>	Input	Attribute	Entity	Route ID	RouteID	Antenna			
Input	Attribute	Entity									
Route ID	RouteID	Antenna									
		Step 10: The system prompts the Administrator to confirm to add the new Antenna with the following: <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Confirmation</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed.</td></tr><tr><td>"Are you sure you want to add this Route?"</td><td>Dialog Box Message</td><td>Used to inform the Administrator of the action.</td></tr></table>	Text/Label	Control Type	Additional Info	Confirmation	Dialog Box Title	Used to indicate what type of message is being displayed.	"Are you sure you want to add this Route?"	Dialog Box Message	Used to inform the Administrator of the action.
Text/Label	Control Type	Additional Info									
Confirmation	Dialog Box Title	Used to indicate what type of message is being displayed.									
"Are you sure you want to add this Route?"	Dialog Box Message	Used to inform the Administrator of the action.									

			Yes Button Used to confirm the action.							
			No Button Used to decline the action.							
	Step 11: The Administrator clicks the "Yes" button. [alt]		Step 12: The Route is assigned a unique RouteID in the Route table which is set to the next consecutive number after the previous RouteID in the database, the system then creates and saves the new Route in the Route table. The following attributes are saved in the database by using a FIRESQL INSERT query: RouteAntenna RouteAntennaID RouteID AntennaID Position Route RouteID SchoolID							
			Step 13: The system notifies the Administrator that the Route has been added successfully to the database with the following: [alt] <table><thead><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr></thead><tbody><tr><td>Success</td><td>Dialog Box Title</td><td>Used to indicate what type of</td></tr></tbody></table>			Text/Label	Control Type	Additional Info	Success	Dialog Box Title
Text/Label	Control Type	Additional Info								
Success	Dialog Box Title	Used to indicate what type of								

			message is being displayed.
			"Route has been added successfully" Dialog Box Message Used to inform the Administrator of the result.
			Ok Button Used to accept the result.
			Step 14: The system creates and saves the following in the database by using a FIRESQL INSERT query:
			AuditTrail Table AuditID DateTime TransactionDescription UserID
	Step 15: [alt] The Admin is finished		
Alternate Courses:	Alt Step 6a: The Administrator clicks the "Cancel" button. Terminate Use Case.		
	Alt Step 7a: In the case of an empty Text box, the system notifies the Administrator of the Input error below the input field with the following:		
	Text/Label "Please fill this field."	Control Type Label (with a light red background color fill)	Additional Info Used to inform the Administrator of the result.

	Error	Dialog Box Title	Used to indicate what type of message is being displayed.
	"The Route already exists. Please try again."	Dialog Box Message	Used to inform the Administrator of the result.
	Ok	Button	Used to accept the result.
	Redirect to Step 5.		
	Alt Step 11: The Administrator declines to add the new Route by clicking the "No" button. Redirect to Step 5.		
	Alt Step 13: The Route was not added successfully to the database, the system will notify the Administrator of the save error with the following:		
	Text/Label	Control Type	Additional Info
	Failed	Dialog Box Title	Used to indicate what type of message is being displayed
	"Create unsuccessful. Please try again."	Dialog Box Message	Used to inform the Administrator of the result.
	Ok	Button	Used to accept the result.
Redirect to Step 5.			
	Alt Step 15: The Administrator clicks the "Add Antenna" button. And return to step 5 to enter another Antenna's details		

Conclusion:	The use case concludes when the Administrator has been notified that the Route has been added successfully.
Post-Condition:	• A new Route is added to the system with the relevant details. • The AuditTrail table is successfully updated. • The Route tables are successfully updated with the details being created.
Business Rules	• Only an Administrator can add a Route to the system.
Implementation Constraints and Specifications:	• None
Assumptions:	• None
Open Issues:	• None

2.3.8.4. Search Route

System Name:	VVs
Author:	Hansen Li
Date:	18/06/2020
Version:	1.1

Use Case Name:	Search Route	Use Case Type:
Use Case ID:	7.4	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	Administrator	
Primary System Actor	None	
Other Participating Actors:	None	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process of an Administrator searching a Route on the system. The Use Case starts when the Administrator navigates by selecting the "Route Management" option under the "System Management" Combo box in the Navigation bar. The system populates the table with the information of all the Routes.	
Pre-Condition:	The Administrator must be logged into the system.	
Trigger:	The Administrator requests to search for a Route on the system.	

Typical Course Of Events:	Actor Action	System Response																	
		Manual Action	Automated Action																
	Step 1: The Administrator selects the “Route Management” option under the “System Management” Combo box in the Navigation bar.		Step 2: The system Displays the “Route Management” screen and populates it with the following details:																
			<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Add Route</td><td>Button</td><td>Used to load the “Route” screen.</td></tr><tr><td>Search</td><td>Text-Box</td><td>Used for capturing Search criteria for the specific Route.</td></tr><tr><td>Route Management</td><td>Table</td><td>Displays a list of Routes.</td></tr><tr><td>“Pen icon”</td><td>Button</td><td>Used to navigate to the specific User view.</td></tr></table>		Text/Label	Control Type	Additional Info	Add Route	Button	Used to load the “Route” screen.	Search	Text-Box	Used for capturing Search criteria for the specific Route.	Route Management	Table	Displays a list of Routes.	“Pen icon”	Button	Used to navigate to the specific User view.
			Text/Label	Control Type	Additional Info														
			Add Route	Button	Used to load the “Route” screen.														
			Search	Text-Box	Used for capturing Search criteria for the specific Route.														
			Route Management	Table	Displays a list of Routes.														
			“Pen icon”	Button	Used to navigate to the specific User view.														
			The table is populated by using a FIRESQL READ query with data from the following:																
			<table><tr><th>Entity</th><th>Attributes</th><th>Input Name</th></tr><tr><td>Route</td><td>RouteID</td><td>Route ID</td></tr><tr><td>Route</td><td>SchoolID</td><td>School ID</td></tr><tr><td>School</td><td>SchoolName</td><td>School Name</td></tr></table>		Entity	Attributes	Input Name	Route	RouteID	Route ID	Route	SchoolID	School ID	School	SchoolName	School Name			
			Entity	Attributes	Input Name														
			Route	RouteID	Route ID														
			Route	SchoolID	School ID														
School	SchoolName	School Name																	
Step 4: The system uses a FIRESQL READ query to READ the following details in the database: [alt]																			

	to search by. They do so by typing in a keyword in the "Search" Text box and click the search button.		Route RouteID SchoolName														
			Step 5: The following details are loaded and used to populate the "User Management" table by using a FIRESQL READ query: <table><tr><th>Output</th><th>Attribute</th><th>Entity</th></tr><tr><td>Route ID</td><td>RouteID</td><td>Route</td></tr><tr><td>School Name</td><td>SchoolName</td><td>School</td></tr></table>	Output	Attribute	Entity	Route ID	RouteID	Route	School Name	SchoolName	School					
	Output	Attribute	Entity														
Route ID	RouteID	Route															
School Name	SchoolName	School															
Step 6: The Administrator clicks on the "Pen icon" Route Button in the "Action" table column.		Step 7: The system will load the "Route" screen and displays it with the following controls and layout: <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Menu</td><td>Menu-icon</td><td>Used to Navigate the side menu-bar.</td></tr><tr><td>Route</td><td>Label</td><td>Heading for the pop-up displayed.</td></tr><tr><td>School</td><td>Label</td><td>Used to indicate what the corresponding text box is used for.</td></tr><tr><td>Antenna</td><td>Label</td><td>Heading to display which</td></tr></table>	Text/Label	Control Type	Additional Info	Menu	Menu-icon	Used to Navigate the side menu-bar.	Route	Label	Heading for the pop-up displayed.	School	Label	Used to indicate what the corresponding text box is used for.	Antenna	Label	Heading to display which
Text/Label	Control Type	Additional Info															
Menu	Menu-icon	Used to Navigate the side menu-bar.															
Route	Label	Heading for the pop-up displayed.															
School	Label	Used to indicate what the corresponding text box is used for.															
Antenna	Label	Heading to display which															

			antenna you are adding		
			Antenna position	Label	Used to indicate what the corresponding text box is used for.
			Antenna ID	Label	Used to indicate what the corresponding text box is used for.
			School	Combo box	Used to select the school
			Antenna Position	Text box	Used to Display the Antenna position
			Antenna ID	Combo box	Used to select the Antenna
			Update Route	Button	Used to update details for this Route
			Delete Route	Button	Used to Delete Route Details.
			Cancel	Button	Used to cancel the action of adding a Route

			The Route Combo box options are populated by using a FIRESQL READ query with data from the following: <table><tr><th>Entity</th><th>Attributes</th></tr><tr><td>School</td><td>SchoolName</td></tr></table> <table><tr><th>Entity</th><th>Attributes</th></tr><tr><td>Antenna</td><td>AntennaID</td></tr></table>	Entity	Attributes	School	SchoolName	Entity	Attributes	Antenna	AntennaID	
	Entity	Attributes										
School	SchoolName											
Entity	Attributes											
Antenna	AntennaID											
			Step 8: The system creates and saves the following in the database by using a FIRESQL INSERT query: <table><tr><th>AuditTrail Table</th></tr><tr><td>AuditID</td></tr><tr><td>DateTime</td></tr><tr><td>TransactionDescription</td></tr><tr><td>UserID</td></tr></table>	AuditTrail Table	AuditID	DateTime	TransactionDescription	UserID				
AuditTrail Table												
AuditID												
DateTime												
TransactionDescription												
UserID												
Alternate Courses:	Alt Step 4: The specific Route does not exist in the database. The system will notify the Administrator of this with the following: <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>"Route not found."</td><td>Label</td><td>Used to inform the Administrator of the result.</td></tr><tr><td>"X"</td><td>Banner (with a light red</td><td>Used to display the result message.</td></tr></table>			Text/Label	Control Type	Additional Info	"Route not found."	Label	Used to inform the Administrator of the result.	"X"	Banner (with a light red	Used to display the result message.
Text/Label	Control Type	Additional Info										
"Route not found."	Label	Used to inform the Administrator of the result.										
"X"	Banner (with a light red	Used to display the result message.										

	background color fill) Redirect to Step 3.
Conclusion:	The use case narrative concludes when the searched Route details are displayed.
Post-Condition:	• The Route has been searched successfully. • The AuditTrail table is successfully updated.
Business Rules	• Only an Administrator can search for a Route on the system.
Implementation Constraints and Specifications:	• None
Assumptions:	• None
Open Issues:	• None

2.3.8.5. Update Route

System Name:	VVs
Author:	Hansen Li
Date:	18/06/2020
Version:	1.1

Use Case Name:	Update Route	Use Case Type:
Use Case ID:	7.5	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	Administrator	
Primary System Actor	None	
Other Participating Actors:	None	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process of an Administrator updating an existing Route's details in the system. The Use Case starts when the Administrator navigates by selecting the "Route Management" option under the "System Management" Combo box in the Navigation bar. From there, they search for the Route they want to update and the system will load the "Route" screen with the Route details. The system populates the relevant Combo boxes with information of the searched Route, and the Administrator can make their changes and save.	
Pre-Condition:	The Administrator must be logged into the system.	

Trigger:	The Administrator requests to update an existing Route to the system.								
Typical Course Of Events:	Actor Action	System Response							
		Manual Action	Automated Action						
	Step 1: The Administrator selects the “Route Management” option under the “System Management” Combo box in the Navigation bar.		Step 2: The System invokes Use Case 13.6 “Search Route” and displays the “Route” screen with the data of the searched route.						
	Step 3: Administrator updates the following User details: - School Selected - Antenna Selected								
	Step 4: The Administrator clicks the “Update” button. [alt]		Step 5: The system captures and ensures that all of the fields are filled in and an option is selected in the Combo box. [alt]						
			Step 6: The system validates that the captured details are in the correct format: [alt] Route table <table><tr><th>Attribute</th><th>Validation</th></tr><tr><td>School</td><td> - Option must be selected </td></tr><tr><td>AntennaID</td><td> - Option must be selected </td></tr></table>	Attribute	Validation	School	Option must be selected	AntennaID	Option must be selected
	Attribute	Validation							
	School	Option must be selected							
AntennaID	Option must be selected								
		Step 7: The system prompts the Administrator to confirm the update of the Route’s details. <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr></table>	Text/Label	Control Type	Additional Info				
Text/Label	Control Type	Additional Info							

			Confirmation	Dialog Box Title	Used to indicate what type of message is being displayed.
			"Are you sure you want to update this Route?"	Dialog Box Message	Used to inform the Administrator of the action.
			Yes	Button	Used to confirm the action.
			No	Button	Used to decline the action.
	Step 8: The Administrator clicks the "Yes" button. [alt]		Step 9: The system then updates and saves the updated Route details in the Route table. The following attributes are updated in the database by using a FIRESQL UPDATE query: RouteAntenna RouteAntennaID RouteID AntennaID Position Route RouteID SchoolID		

			Step 10: The system notifies the Administrator that the Route has been updated successfully in the database with the following: [alt]												
			<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Success</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed.</td></tr><tr><td>"The Route has been updated successfully"</td><td>Dialog Box Message</td><td>Used to inform the Administrator of the result.</td></tr><tr><td>Ok</td><td>Button</td><td>Used to accept the result.</td></tr></table>	Text/Label	Control Type	Additional Info	Success	Dialog Box Title	Used to indicate what type of message is being displayed.	"The Route has been updated successfully"	Dialog Box Message	Used to inform the Administrator of the result.	Ok	Button	Used to accept the result.
	Text/Label	Control Type	Additional Info												
	Success	Dialog Box Title	Used to indicate what type of message is being displayed.												
	"The Route has been updated successfully"	Dialog Box Message	Used to inform the Administrator of the result.												
Ok	Button	Used to accept the result.													
			Step 11: The system updates and saves the following in the database by using a FIRESQL INSERT query:												
			<table><tr><th>AuditTrail Table</th></tr><tr><td>AuditID</td></tr><tr><td>DateTime</td></tr><tr><td>TransactionDescription</td></tr><tr><td>UserID</td></tr></table>	AuditTrail Table	AuditID	DateTime	TransactionDescription	UserID							
AuditTrail Table															
AuditID															
DateTime															
TransactionDescription															
UserID															
Alternate Courses:	Alt Step 4: The Administrator clicks the "Cancel" button. Terminate Use Case.														
	Alt Step 5a: In the case of an empty Text box, the system notifies the Administrator of the Input error below the input field with the following:														
			<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr></table>	Text/Label	Control Type	Additional Info									
Text/Label	Control Type	Additional Info													

	"Please fill this field."	Label (with a light red background color fill)	Used to inform the Administrator of the result.
	"X"	Icon	Used to indicate an error result.
	Redirect to Step 3.		
	Alt Step 5b: In the case of a Combo box option not selected, the system notifies the Administrator of the Input error below the input field with the following:		
	Text/Label	Control Type	Additional Info
	"Please select an option."	Label (with a light red background color fill)	Used to inform the Administrator of the result.
	"X"	Icon	Used to indicate an error result.
	Redirect to Step 3.		
	Alt Step 6: In the case of invalid input being entered, the system notifies the Administrator of the Format error below the input field with the following:		
	Text/Label	Control Type	Additional Info
	"Invalid input."	Label (with a light red background color fill)	Used to inform the Administrator of the result.
	"X"	Icon	Used to indicate an error result.
	Redirect to Step 3.		
	Alt Step 8: The Administrator declines to update the Route by clicking the "No" button. Redirect to Step 4.		

	Alt Step 10: The Route was not updated successfully in the database, the system will notify the Administrator of the update error with the following:		
	Text/Label	Control Type	Additional Info
	Failed	Dialog Box Title	Used to indicate what type of message is being displayed
	"Update unsuccessful. Please try again."	Dialog Box Message	Used to inform the Administrator of the result.
	Ok	Button	Used to accept the result.
	Redirect to Step 3.		
Conclusion:	The use case concludes when the Administrator has been notified that the Route has been updated successfully.		
Post-Condition:	• An existing Route is updated in the system with the relevant details. • The AuditTrail table is successfully updated. • The Route and RouteAntenna tables are successfully updated with the details being updated.		
Business Rules	• Only an Administrator can update a Route on the system.		
Implementation Constraints and Specifications:	• None		
Assumptions:	• None		
Open Issues:	• None		

2.3.8.6. Delete Route

System Name:	VVs
Author:	Hansen Li
Date:	18/06/2020
Version:	1.1

Use Case Name:	Delete Route	Use Case Type:
Use Case ID:	7.6	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	Administrator	
Primary System Actor	None	
Other Participating Actors:	None	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process of an Administrator deleting an existing Route's details from the system. The Use Case starts when the Administrator navigates by selecting the "Route Management" option under the "System Management" Combo box in the Navigation bar. From there, they search for the Route they want to delete and the system displays the "Route" screen with the Route details to be deleted. The system populates the relevant Combo boxes with information of the searched Route, and the Administrator can make their deletion.	
Pre-Condition:	The Administrator must be logged into the system.	

Trigger:	The Administrator requests to delete an existing Route from the system.																	
Typical Course Of Events:	Actor Action	System Response																
		Manual Action	Automated Action															
	Step 1: The Administrator selects the "Route Management" option under the "System Management" Combo box in the Navigation bar.		Step 2: The System invokes Use Case 13.6 "Search Route" and displays the "Route" screen with the data of the searched route.															
	Step 3: The Administrator clicks on the "Delete" button. [alt]		Step 4: The system prompts the Administrator to confirm the deletion of the Route's details with the following:															
			<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Confirmation</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed.</td></tr><tr><td>"Are you sure you want to delete this Route?"</td><td>Dialog Box Message</td><td>Used to inform the Administrator of the action.</td></tr><tr><td>Yes</td><td>Button</td><td>Used to confirm the action.</td></tr><tr><td>No</td><td>Button</td><td>Used to decline the action.</td></tr></table>		Text/Label	Control Type	Additional Info	Confirmation	Dialog Box Title	Used to indicate what type of message is being displayed.	"Are you sure you want to delete this Route?"	Dialog Box Message	Used to inform the Administrator of the action.	Yes	Button	Used to confirm the action.	No	Button
Text/Label			Control Type	Additional Info														
Confirmation			Dialog Box Title	Used to indicate what type of message is being displayed.														
"Are you sure you want to delete this Route?"			Dialog Box Message	Used to inform the Administrator of the action.														
Yes	Button	Used to confirm the action.																
No	Button	Used to decline the action.																

	Step 5: The Administrator clicks the “Yes” button. [alt]		Step 6: The system then removes the Route details in the Route table. The following attributes are deleted in the database by using a FIRESQL DELETE query: RouteAntenna RouteAntennaID RoutelD AntennaID Position Route RoutelD SchoolID											
			Step 7: The system notifies the Administrator that the Route has been deleted successfully from the database with the following: [alt] <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Success</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed.</td></tr><tr><td>“The Route has been deleted successfully”</td><td>Dialog Box Message</td><td>Used to inform the Administrator of the result.</td></tr><tr><td>Ok</td><td>Button</td><td>Used to accept the result.</td></tr></table>	Text/Label	Control Type	Additional Info	Success	Dialog Box Title	Used to indicate what type of message is being displayed.	“The Route has been deleted successfully”	Dialog Box Message	Used to inform the Administrator of the result.	Ok	Button
Text/Label	Control Type	Additional Info												
Success	Dialog Box Title	Used to indicate what type of message is being displayed.												
“The Route has been deleted successfully”	Dialog Box Message	Used to inform the Administrator of the result.												
Ok	Button	Used to accept the result.												

			Step 8: The system updates and saves the following in the database by using a FIRESQL INSERT query: <table><tr><th>AuditTrail Table</th></tr><tr><td>AuditID</td></tr><tr><td>DateTime</td></tr><tr><td>TransactionDescription</td></tr><tr><td>UserID</td></tr></table>	AuditTrail Table	AuditID	DateTime	TransactionDescription	UserID							
AuditTrail Table															
AuditID															
DateTime															
TransactionDescription															
UserID															
Alternate Courses:	Alt Step 3: The Administrator clicks the "Cancel" button. Terminate Use Case.														
	Alt Step 5: The Administrator declines to delete the Route by clicking the "No" button. Redirect to Step 2.														
	Alt Step 7: The Administrator is notified of the unsuccessful delete action with the following:														
	<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Failed</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed</td></tr><tr><td>"Delete unsuccessful. Please try again."</td><td>Dialog Box Message</td><td>Used to inform the Administrator of the result.</td></tr><tr><td>Ok</td><td>Button</td><td>Used to accept the result.</td></tr></table>	Text/Label	Control Type	Additional Info	Failed	Dialog Box Title	Used to indicate what type of message is being displayed	"Delete unsuccessful. Please try again."	Dialog Box Message	Used to inform the Administrator of the result.	Ok	Button	Used to accept the result.		
	Text/Label	Control Type	Additional Info												
	Failed	Dialog Box Title	Used to indicate what type of message is being displayed												
"Delete unsuccessful. Please try again."	Dialog Box Message	Used to inform the Administrator of the result.													
Ok	Button	Used to accept the result.													
Redirect to Step 2.															
Conclusion:	The use case concludes when the Administrator has been notified that the Route has been successfully deleted.														
Post-Condition:	• An existing Route is deleted from the system with all the relevant details being removed.														

	• The AuditTrail table is successfully updated. • The Route and RouteAntenna tables are successfully updated with the deleted details being removed.
Business Rules	• Only an Administrator can delete a Route from the system.
Implementation Constraints and Specifications:	• None
Assumptions:	• None
Open Issues:	• None

2.3.9. Sub-system 8: Job Tasks

2.3.9.1. Add Fault Task

System Name:	VVs
Author:	Gareth Phillips
Date:	15/06/2020
Version:	1.0

Use Case Name:	Add Fault Task	Use Case Type:
Use Case ID:	8.1	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	Administrator	
Primary System Actor	None	
Other Participating Actors:	None	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process of an Administrator adding a new Fault Task to the system. The Use Case starts when the Administrator navigates by selecting the " Fault Task Management" option under the "System Management" Combo box in the Navigation bar. From there, they click the "Add Fault Task" button and the system will load the "Fault Task" screen. The system populates the corresponding labels, text boxes and buttons, icons, logos, and banners.	

Pre-Condition:	The Administrator must be logged into the system.																							
Trigger:	The Administrator requests to add a new Fault Task to the system.																							
Typical Course Of Events:	Actor Action	System Response																						
		Manual Action	Automated Action																					
	Step 1: The Administrator selects the "Fault Task Management" option under the "System Management" Combo box in the Navigation bar.		Step 2: The system Displays the "Fault Task Management" screen and populates it with the following details:																					
			<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>" + Icon "</td><td>Button</td><td>Used to load the "Fault Task" screen.</td></tr><tr><td>Search</td><td>Text-Box</td><td>Used for capturing Search criteria for the specific Fault Task.</td></tr><tr><td>Fault Task Management</td><td>Table</td><td>Displays a list of Fault tasks.</td></tr><tr><td>"Pen icon"</td><td>Button</td><td>Used to navigate to the specific Fault Task view.</td></tr><tr><td>"Search"</td><td>Button</td><td>Used to perform search Action.</td></tr><tr><td>"Help Icon"</td><td>Button</td><td>Used to Load the Help document.</td></tr></table>	Text/Label	Control Type	Additional Info	" + Icon "	Button	Used to load the "Fault Task" screen.	Search	Text-Box	Used for capturing Search criteria for the specific Fault Task.	Fault Task Management	Table	Displays a list of Fault tasks.	"Pen icon"	Button	Used to navigate to the specific Fault Task view.	"Search"	Button	Used to perform search Action.	"Help Icon"	Button	Used to Load the Help document.
			Text/Label	Control Type	Additional Info																			
			" + Icon "	Button	Used to load the "Fault Task" screen.																			
			Search	Text-Box	Used for capturing Search criteria for the specific Fault Task.																			
			Fault Task Management	Table	Displays a list of Fault tasks.																			
			"Pen icon"	Button	Used to navigate to the specific Fault Task view.																			
"Search"			Button	Used to perform search Action.																				
"Help Icon"	Button	Used to Load the Help document.																						

			The table is populated by using a FIRESQL READ query with data from the following: <table><tr><th>Entity</th><th>Attributes</th><th>Input Name</th></tr><tr><td>Fault Task</td><td>FaultTask</td><td>Fault Task</td></tr><tr><td>Fault Task</td><td>FaultTaskDescription</td><td>FaultTask Description</td></tr></table>	Entity	Attributes	Input Name	Fault Task	FaultTask	Fault Task	Fault Task	FaultTaskDescription	FaultTask Description					
	Entity	Attributes	Input Name														
	Fault Task	FaultTask	Fault Task														
	Fault Task	FaultTaskDescription	FaultTask Description														
	Step 3: The Administrator clicks on the "+ Icon" button.		Step 4: The system will load the "Fault Task" screen and displays it with the following controls and layout: <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Navigation</td><td>Navigation-bar</td><td>Used to Navigate the side menu-bar.</td></tr><tr><td>Fault Task</td><td>Label</td><td>Heading for the pop-up displayed.</td></tr><tr><td>Fault Task</td><td>Label</td><td>Used to indicate what the corresponding text box is used for.</td></tr><tr><td>Fault Task Description</td><td>Label</td><td>Used to indicate what the corresponding text box is used for.</td></tr></table>	Text/Label	Control Type	Additional Info	Navigation	Navigation-bar	Used to Navigate the side menu-bar.	Fault Task	Label	Heading for the pop-up displayed.	Fault Task	Label	Used to indicate what the corresponding text box is used for.	Fault Task Description	Label
Text/Label	Control Type	Additional Info															
Navigation	Navigation-bar	Used to Navigate the side menu-bar.															
Fault Task	Label	Heading for the pop-up displayed.															
Fault Task	Label	Used to indicate what the corresponding text box is used for.															
Fault Task Description	Label	Used to indicate what the corresponding text box is used for.															

			Fault Task	Text box	Used to capture text for the Fault Task details.				
			Fault Task Description	Text box	Used to capture text for the Fault Task Description details.				
			Create Fault Task	Button	Used to Save the new Fault Task Details.				
			Cancel	Button	Used to cancel the action of adding a Fault Task.				
	Step 5: Administrator inputs the following Fault Task details: - Fault Task - Fault Task Description								
	Step 6: The Administrator clicks the “Create Fault Task” button. [alt]				Step 7: The system captures and ensures that all of the fields are filled in.[alt]				
					Step 8: The system validates that the captured details are in the correct format: [alt] Fault Task table <table><thead><tr><th>Attribute</th><th>Validation</th></tr></thead><tbody><tr><td>FaultTask</td><td>40.Max 50 characters</td></tr></tbody></table>	Attribute	Validation	FaultTask	40.Max 50 characters
Attribute	Validation								
FaultTask	40.Max 50 characters								

			FaultTaskDescription • Max 150 characters • Text characters only														
			Step 9: The system verifies that no duplicate Fault Task exists on the system by comparing the following: [alt] <table><tr><th>Input</th><th>Attribute</th><th>Entity</th></tr><tr><td>Fault Task</td><td>FaultTask</td><td>Fault Task</td></tr></table>	Input	Attribute	Entity	Fault Task	FaultTask	Fault Task								
	Input	Attribute	Entity														
	Fault Task	FaultTask	Fault Task														
			Step 10: The system prompts the Administrator to confirm to add the new Fault Task with the following: <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Confirmation</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed.</td></tr><tr><td>"Are you sure you want to add this Fault Task?"</td><td>Dialog Box Message</td><td>Used to inform the Administrator of the action.</td></tr><tr><td>Yes</td><td>Button</td><td>Used to confirm the action.</td></tr><tr><td>No</td><td>Button</td><td>Used to decline the action.</td></tr></table>	Text/Label	Control Type	Additional Info	Confirmation	Dialog Box Title	Used to indicate what type of message is being displayed.	"Are you sure you want to add this Fault Task?"	Dialog Box Message	Used to inform the Administrator of the action.	Yes	Button	Used to confirm the action.	No	Button
Text/Label	Control Type	Additional Info															
Confirmation	Dialog Box Title	Used to indicate what type of message is being displayed.															
"Are you sure you want to add this Fault Task?"	Dialog Box Message	Used to inform the Administrator of the action.															
Yes	Button	Used to confirm the action.															
No	Button	Used to decline the action.															

	Step 11: The Administrator clicks the “Yes” button. [alt]		Step 12: The Fault Task is assigned a unique FaultTaskID in the Fault Task table which is set to the next consecutive number after the previous FaultTaskID in the database, the system then creates and saves the new Fault Task in the FaultTask table. The following attributes are saved in the database by using a FIRESQL INSERT query: Fault Task Table FaultTaskID FaultTask FaultTaskDescription												
			Step 13: The system notifies the Administrator that the Fault Task has been added successfully to the database with the following: [alt] <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Success</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed.</td></tr><tr><td>“Fault Task has been added successfully”</td><td>Dialog Box Message</td><td>Used to inform the Administrator of the result.</td></tr><tr><td>Ok</td><td>Button</td><td>Used to accept the result.</td></tr></table>	Text/Label	Control Type	Additional Info	Success	Dialog Box Title	Used to indicate what type of message is being displayed.	“Fault Task has been added successfully”	Dialog Box Message	Used to inform the Administrator of the result.	Ok	Button	Used to accept the result.
	Text/Label	Control Type	Additional Info												
Success	Dialog Box Title	Used to indicate what type of message is being displayed.													
“Fault Task has been added successfully”	Dialog Box Message	Used to inform the Administrator of the result.													
Ok	Button	Used to accept the result.													
		Step 14: The system creates and saves the following in the database by using a FIRESQL INSERT query: AuditTrail Table													

			AuditTrailID
			DateTime
			TransactionDescription
			UserID
Alternate Courses:	Alt Step 6: The Administrator clicks the "Cancel" button. Terminate Use Case.		
	Alt Step 7: In the case of an empty Text box, the system notifies the Administrator of the Input error below the input field with the following:		
	Text/Label	Control Type	Additional Info
	"Please fill this field."	Label (with a light red background color fill)	Used to inform the Administrator of the result.
	"X"	Icon	Used to indicate an error result.
	Redirect to Step 5.		
	Alt Step 8: In the case of invalid input being entered, the system notifies the Administrator of the Format error below the input field with the following:		
	Text/Label	Control Type	Additional Info
	"Invalid input."	Label (with a light red background color fill)	Used to inform the Administrator of the result.
	"X"	Icon	Used to indicate an error result.
		Redirect to Step 5.	

Alt Step 9: The inputted Fault Task already exists in the system; the system will notify the Administrator of the duplication error with the following:

Text/Label	Control Type	Additional Info
Error	Dialog Box Title	Used to indicate what type of message is being displayed.
"The Fault Task already exists. Please try again."	Dialog Box Message	Used to inform the Administrator of the result.
"Help icon"	Button	Used to Load the Help document.
Ok	Button	Used to accept the result.

Redirect to Step 5.

Alt Step 11: The Administrator declines to add the new Fault Task by clicking the "No" button. Redirect to Step 5.

Alt Step 13: The Fault Task was not added successfully to the database, the system will notify the Administrator of the save error with the following:

Text/Label	Control Type	Additional Info
Failed	Dialog Box Title	Used to indicate what type of message is being displayed
"Create unsuccessful. Please try again."	Dialog Box Message	Used to inform the Administrator of the result.

	"Help icon"	Button	Used to Load the Help document.
	Ok	Button	Used to accept the result.
	Redirect to Step 5.		
Conclusion:	The use case concludes when the Administrator has been notified that the Fault Task has been added successfully.		
Post-Condition:	• A new Fault Task is added to the system with the relevant details. • The AuditTrail table is successfully updated. • The FaultTask table is successfully updated.		
Business Rules	• Only an Administrator can add a Fault Task to the system.		
Implementation Constraints and Specifications:	• None		
Assumptions:	• None		
Open Issues:	• None		

2.3.9.2. Search Fault Task

System Name:	VVs
Author:	Gareth Phillips
Date:	15/06/2020
Version:	1.0

Use Case Name:	Search Fault Task	Use Case Type:
Use Case ID:	8.2	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	Administrator	
Primary System Actor	None	
Other Participating Actors:	None	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process of an Administrator searching a Fault Task on the system. The Use Case starts when the Administrator navigates by selecting the " Fault Task Management" option under the "System Management" Combo box in the Navigation bar. The system populates the table with the information of all the Fault Task.	
Pre-Condition:	The Administrator must be logged into the system.	
Trigger:	The Administrator requests to search for a Fault Task on the system.	

Typical Course Of Events:	Actor Action	System Response			
		Manual Action	Automated Action		
	Step 1: The Administrator selects the “Fault Task Management” option under the “System Management” Combo box in the Navigation bar.		Step 2: The system Displays the “Fault Task Management” screen and populates it with the following details:		
			Text/Label	Control Type	Additional Info
			+ Icon	Button	Used to load the “Fault Task” screen.
			Search	Text-Box	Used for capturing Search criteria for the specific Fault Task.
			Fault Task Management	Table	Displays a list of Fault tasks.
			Pen icon	Button	Used to navigate to the specific Fault Task view.
			Search ”	Button	Used to perform the search action.
			The table is populated by using a FIRESQL READ query with data from the following:		
Entity			Attributes	Input Name	
Fault Task	FaultTask	Parts Needed			

			Fault Task	FaultTaskDescription	FaultTask Description								
	Step 3: The Administrator enters the search criteria that they wish to search by. They do so by typing in a keyword in the "Search" Text box. Then click the "Search" Button												
			Step 4: The system uses a FIRESQL READ query to READ the following details in the database: [alt] <table><tr><th>Entity</th><th>Attributes</th></tr><tr><td>Fault Task</td><td>FaultTask</td></tr></table>			Entity	Attributes	Fault Task	FaultTask				
	Entity	Attributes											
	Fault Task	FaultTask											
		Step 5: The following details are loaded and used to populate the "Fault Task Management" table by using a FIRESQL READ query: <table><tr><th>Output</th><th>Attribute</th><th>Entity</th></tr><tr><td>Fault Task</td><td>PartsNeeded</td><td>Fault Task</td></tr><tr><td>Fault Task Description</td><td>FaultTaskDescription</td><td>Fault Task</td></tr></table>			Output	Attribute	Entity	Fault Task	PartsNeeded	Fault Task	Fault Task Description	FaultTaskDescription	Fault Task
Output	Attribute	Entity											
Fault Task	PartsNeeded	Fault Task											
Fault Task Description	FaultTaskDescription	Fault Task											
Step 6: The Administrator clicks on the "Pen icon" Fault Task Button in the "Action" table column.		Step 7: The System will load the "Fault Task" screen and displays it with the following controls and layout: <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Navigation</td><td>Navigation-Bar</td><td>Used for Navigation of the site.</td></tr></table>			Text/Label	Control Type	Additional Info	Navigation	Navigation-Bar	Used for Navigation of the site.			
Text/Label	Control Type	Additional Info											
Navigation	Navigation-Bar	Used for Navigation of the site.											

			Fault Task	Label	Heading for the pop-up displayed.
			Fault Task	Label	Used to indicate what the corresponding text box is used for.
			Fault Task Description	Label	Used to indicate what the corresponding text box is used for.
			Fault Task	Text box	Used to capture text for the Fault Task details.
			Fault Task Description	Text box	Used to capture text for the Fault Task Description details.
			Delete	Button	Used to Delete the Fault Task Details.
			Update	Button	Used to Update the Fault Task Details.

			the database where the FaultTaskID is the same as the one searched by using a FIRESQL READ query: <table><tr><th>Output</th><th>Attribute</th><th>Entity</th></tr><tr><td>FaultTaskID</td><td>FaultTaskID</td><td>Fault Task</td></tr><tr><td>Fault Task</td><td>FaultTask</td><td>Fault Task</td></tr><tr><td>Fault Task Description</td><td>FaultTaskDescription</td><td>Fault Task</td></tr></table>	Output	Attribute	Entity	FaultTaskID	FaultTaskID	Fault Task	Fault Task	FaultTask	Fault Task	Fault Task Description	FaultTaskDescription	Fault Task
	Output	Attribute	Entity												
FaultTaskID	FaultTaskID	Fault Task													
Fault Task	FaultTask	Fault Task													
Fault Task Description	FaultTaskDescription	Fault Task													
			Step 8: The system creates and saves the following in the database by using a FIRESQL INSERT query: AuditTrail Table AuditTrailID DateTime TransactionDescription UserID												
Alternate Courses:	Alt Step 4: The specific Fault Task does not exist in the database. The system will notify the Administrator of this with the following: <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>"Fault Task not found."</td><td>Label</td><td>Used to inform the Administrator of the result.</td></tr><tr><td>"X"</td><td>Banner (with a light red background color fill)</td><td>Used to display the result message.</td></tr></table>			Text/Label	Control Type	Additional Info	"Fault Task not found."	Label	Used to inform the Administrator of the result.	"X"	Banner (with a light red background color fill)	Used to display the result message.			
Text/Label	Control Type	Additional Info													
"Fault Task not found."	Label	Used to inform the Administrator of the result.													
"X"	Banner (with a light red background color fill)	Used to display the result message.													

	Redirect to Step 3.
Conclusion:	The use case narrative concludes when the searched Fault Task details are displayed.
Post-Condition:	• The Fault Task has been searched successfully. • The AuditTrail table is successfully updated.
Business Rules	• Only an Administrator can search for a Fault Task on the system.
Implementation Constraints and Specifications:	• None
Assumptions:	• None
Open Issues:	• None

2.3.9.3. Update Fault Task

System Name:	VVs
Author:	Gareth Phillips
Date:	15/06/2020
Version:	1.0

Use Case Name:	Update Fault Task	Use Case Type:
Use Case ID:	8.3	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	Administrator	
Primary System Actor	None	
Other Participating Actors:	None	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process of an Administrator updating an existing Fault Task's details in the system. The Use Case starts when the Administrator navigates by selecting the " Fault Task Management" option under the "System Management" Combo box in the Navigation bar. From there, they search for the Fault Task they want to update, the system then will load the "Fault Task" screen with the Fault Task details. The Administrator can then make their changes and save it.	
Pre-Condition:	The Administrator must be logged into the system.	

Trigger:	The Administrator requests to update an existing Fault Task to the system.								
Typical Course Of Events:	Actor Action	System Response							
		Manual Action	Automated Action						
	Step 1: The Administrator selects the “Fault Task Management” option under the “System Management” Combo box in the Navigation bar.		Step 2: The System invokes Use Case 8.2 “Search Fault Task” and displays the “Fault Task” screen with the data of the searched Fault Task.						
	Step 3: Administrator updates the following Fault Task details: - Fault Task - Fault Task Description								
	Step 4: The Administrator clicks the “Update” button. [alt]		Step 5: The system captures and ensures that all of the fields are filled in. [alt]						
			Step 6: The system validates that the captured details are in the correct format: [alt] Fault Task <i>table</i> <table><thead><tr><th>Attribute</th><th>Validation</th></tr></thead><tbody><tr><td>FaultTask</td><td>41.Max 50 characters</td></tr><tr><td>FaultTaskDescription</td><td> - Max 150 characters - Text characters only </td></tr></tbody></table>	Attribute	Validation	FaultTask	41.Max 50 characters	FaultTaskDescription	- Max 150 characters - Text characters only
	Attribute	Validation							
FaultTask	41.Max 50 characters								
FaultTaskDescription	- Max 150 characters - Text characters only								
		Step 7: The system prompts the Administrator to confirm the update of the Fault Task’s details. <table><thead><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr></thead></table>	Text/Label	Control Type	Additional Info				
Text/Label	Control Type	Additional Info							

			Confirmation	Dialog Box Title	Used to indicate what type of message is being displayed.
			"Are you sure you want to update this Fault Task?"	Dialog Box Message	Used to inform the Administrator of the action.
			Yes	Button	Used to confirm the action.
			No	Button	Used to decline the action.
	Step 8: The Administrator clicks the "Yes" button. [alt]		Step 9: The system then updates and saves the updated Fault Task details in the FaultTask. The following attributes are updated in the database by using a FIRESQL UPDATE query: Fault Task Table FaultTaskID FaultTask FaultTaskDescription		
		Step 10: The system notifies the Administrator that the Fault Task has been updated successfully in the database with the following: [alt]			
			Text/Label	Control Type	Additional Info
			Success	Dialog Box Title	Used to indicate what type of

			message is being displayed.								
			<table><tr><td>"The Fault Task has been updated successfully"</td><td>Dialog Box Message</td><td>Used to inform the Administrator of the result.</td></tr><tr><td>Ok</td><td>Button</td><td>Used to accept the result.</td></tr></table>	"The Fault Task has been updated successfully"	Dialog Box Message	Used to inform the Administrator of the result.	Ok	Button	Used to accept the result.		
	"The Fault Task has been updated successfully"	Dialog Box Message	Used to inform the Administrator of the result.								
Ok	Button	Used to accept the result.									
		Step 11: The system updates and saves the following in the database by using a FIRESQL INSERT query: <table><tr><td>AuditTrail Table</td></tr><tr><td>AuditTrailID</td></tr><tr><td>DateTime</td></tr><tr><td>TransactionDescription</td></tr><tr><td>UserID</td></tr></table>		AuditTrail Table	AuditTrailID	DateTime	TransactionDescription	UserID			
AuditTrail Table											
AuditTrailID											
DateTime											
TransactionDescription											
UserID											
Alternate Courses:	Alt Step 4: The Administrator clicks the "Cancel" button. Terminate Use Case.										
	Alt Step 5a: In the case of an empty Text box, the system notifies the Administrator of the Input error below the input field with the following:										
	<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>"Please fill this field."</td><td>Label (with a light red background color fill)</td><td>Used to inform the Administrator of the result.</td></tr><tr><td>"X"</td><td>Icon</td><td>Used to indicate an error result.</td></tr></table>	Text/Label	Control Type	Additional Info	"Please fill this field."	Label (with a light red background color fill)	Used to inform the Administrator of the result.	"X"	Icon	Used to indicate an error result.	
Text/Label	Control Type	Additional Info									
"Please fill this field."	Label (with a light red background color fill)	Used to inform the Administrator of the result.									
"X"	Icon	Used to indicate an error result.									

Redirect to Step 3.

Alt Step 6: In the case of invalid input being entered, the system notifies the Administrator of the Format error below the input field with the following:

Text/Label	Control Type	Additional Info
"Invalid input."	Label (with a light red background color fill)	Used to inform the Administrator of the result.
"X"	Icon	Used to indicate an error result.

Redirect to Step 3.

Alt Step 8: The Administrator declines to update the Fault Task by clicking the "No" button. Redirect to Step 4.

Alt Step 10: The Fault Task was not updated successfully in the database, the system will notify the Administrator of the update error with the following:

Text/Label	Control Type	Additional Info
Failed	Dialog Box Title	Used to indicate what type of message is being displayed
"Update unsuccessful. Please try again."	Dialog Box Message	Used to inform the Administrator of the result.
"Help icon"	Button	Used to Load the Help document.
Ok	Button	Used to accept the result.

	Redirect to Step 3.
Conclusion:	The use case concludes when the Administrator has been notified that the Fault Task has been updated successfully.
Post-Condition:	• An existing Fault Task is updated in the system with the relevant details. • The AuditTrail table is successfully updated. • The FaultTask table is successfully updated.
Business Rules	• Only an Administrator can update a Fault Task to the system.
Implementation Constraints and Specifications:	• None
Assumptions:	• None
Open Issues:	• None

2.3.9.4. Delete Fault Task

System Name:	VVs
Author:	Gareth Phillips
Date:	15/06/2020
Version:	1.0

Use Case Name:	Delete Fault Task	Use Case Type:
Use Case ID:	8.4	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	Administrator	
Primary System Actor	None	
Other Participating Actors:	None	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process of an Administrator deleting an existing Fault Task's details from the system. The Use Case starts when the Administrator navigates by selecting the " Fault Task Management" option under the "System Management" Combo box in the Navigation bar. From there, they search for the Fault Task they want to delete and the system displays the "Fault Task" screen with the Fault Task details to be deleted. The system populates the relevant Labels and corresponding text boxes with information of the searched Fault Task, and the Administrator can make their deletion.	
Pre-Condition:	The Administrator must be logged into the system.	

Trigger:	The Administrator requests to delete an existing Fault Task from the system.				
Typical Course Of Events:	Actor Action	System Response			
		Manual Action	Automated Action		
	Step 1: The Administrator selects the "Fault Task Management" option under the "System Management" Combo box in the Navigation bar.		Step 2: The System invokes Use Case 8.2 "Search Fault Task" and displays the "Fault Task" screen with the data of the searched Maintenance Task.		
	Step 3: The Administrator clicks on the "Delete" button. [alt]		Step 4: The system prompts the Administrator to confirm the deletion of the Fault Task's details with the following:		
			Text/Label	Control Type	Additional Info
			Confirmation	Dialog Box Title	Used to indicate what type of message is being displayed.
"Are you sure you want to delete this Fault Task?"			Dialog Box Message	Used to inform the Administrator of the action.	
Yes			Button	Used to confirm the action.	
No	Button	Used to decline the action.			

	Step 5: The Administrator clicks the “Yes” button. [alt]		Step 6: The system then removes the Fault Task details in the FaultTask table. The following attributes are deleted in the database by using a FIRESQL DELETE query: Fault Task Table FaultTaskID FaultTask FaultTaskDescription												
			Step 7: The system notifies the Administrator that the Fault Task has been deleted successfully from the database with the following: [alt] <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Success</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed.</td></tr><tr><td>“The Fault Task has been deleted successfully”</td><td>Dialog Box Message</td><td>Used to inform the Administrator of the result.</td></tr><tr><td>Ok</td><td>Button</td><td>Used to accept the result.</td></tr></table>	Text/Label	Control Type	Additional Info	Success	Dialog Box Title	Used to indicate what type of message is being displayed.	“The Fault Task has been deleted successfully”	Dialog Box Message	Used to inform the Administrator of the result.	Ok	Button	Used to accept the result.
	Text/Label	Control Type	Additional Info												
Success	Dialog Box Title	Used to indicate what type of message is being displayed.													
“The Fault Task has been deleted successfully”	Dialog Box Message	Used to inform the Administrator of the result.													
Ok	Button	Used to accept the result.													
		Step 8: The system updates and saves the following in the database by using a FIRESQL INSERT query: AuditTrail Table AuditTrail													

			DateTime TransactionDescription UserID
Alternate Courses:	Alt Step 3: The Administrator clicks the "Cancel" button. Terminate Use Case.		
	Alt Step 5: The Administrator declines to delete the Fault Task by clicking the "No" button. Redirect to Step 2.		
	Alt Step 7: The Administrator is notified of the unsuccessful delete action with the following:		
	Text/Label	Control Type	Additional Info
	Failed	Dialog Box Title	Used to indicate what type of message is being displayed
	"Delete unsuccessful. Please try again."	Dialog Box Message	Used to inform the Administrator of the result.
	"Help icon"	Button	Used to Load the Help document.
	Ok	Button	Used to accept the result.
	Redirect to Step 2.		
Conclusion:	The use case concludes when the Administrator has been notified that the Fault Task has been successfully deleted.		
Post-Condition:	- An existing Fault Task is deleted from the system with all the relevant details being removed. - The AuditTrail table is successfully updated. - The FaultTask table is successfully updated with the deleted details being removed.		

Business Rules	• Only an Administrator can delete a Fault Task from the system.
Implementation Constraints and Specifications:	• None
Assumptions:	• None
Open Issues:	• None

2.3.9.5. Add Maintenance Task

System Name:	VVs
Author:	Gareth Phillips
Date:	15/06/2020
Version:	1.0

Use Case Name:	Add Maintenance Task	Use Case Type:
Use Case ID:	8.5	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	Administrator	
Primary System Actor	None	
Other Participating Actors:	None	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process of an Administrator adding a new Maintenance Task to the system. The Use Case starts when the Administrator navigates by selecting the "Maintenance Task Management" option under the "System Management" Combo box in the Navigation bar. From there, they click the "Add Maintenance Task" button and the system will load the "Maintenance Task" screen. The system populates the corresponding labels, text boxes and buttons, icons, logos, and banners.	
Pre-Condition:	The Administrator must be logged into the system.	

Trigger:	The Administrator requests to add a new Maintenance Task to the system.																				
Typical Course	Actor Action	System Response																			
Of Events:		Manual Action	Automated Action																		
	Step 1: The Administrator selects the “Maintenance Task Management” option under the “System Management” Combo box in the Navigation bar.		Step 2: The system Displays the “Maintenance Task Management” screen and populates it with the following details:																		
		<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>“+ Icon”</td><td>Button</td><td>Used to load the “Maintenance Task” screen.</td></tr><tr><td>Search</td><td>Text-Box</td><td>Used for capturing Search criteria for the specific Maintenance Task.</td></tr><tr><td>Maintenance Task Management</td><td>Table</td><td>Displays a list of Maintenance Task.</td></tr><tr><td>“Pen icon”</td><td>Button</td><td>Used to navigate to the specific Maintenance Task view.</td></tr><tr><td>Search</td><td>Button</td><td>Used to perform the search action</td></tr></table>		Text/Label	Control Type	Additional Info	“+ Icon”	Button	Used to load the “Maintenance Task” screen.	Search	Text-Box	Used for capturing Search criteria for the specific Maintenance Task.	Maintenance Task Management	Table	Displays a list of Maintenance Task.	“Pen icon”	Button	Used to navigate to the specific Maintenance Task view.	Search	Button	Used to perform the search action
Text/Label	Control Type	Additional Info																			
“+ Icon”	Button	Used to load the “Maintenance Task” screen.																			
Search	Text-Box	Used for capturing Search criteria for the specific Maintenance Task.																			
Maintenance Task Management	Table	Displays a list of Maintenance Task.																			
“Pen icon”	Button	Used to navigate to the specific Maintenance Task view.																			
Search	Button	Used to perform the search action																			

			The table is populated by using a FIRESQL READ query with data from the following: <table><tr><th>Entity</th><th>Attributes</th><th>Input Name</th></tr><tr><td>Maintenance Task</td><td>MaintenanceTask</td><td>Maintenance Task</td></tr><tr><td>Maintenance Task</td><td>MaintenanceTask Description</td><td>MaintenanceTask Description</td></tr></table>	Entity	Attributes	Input Name	Maintenance Task	MaintenanceTask	Maintenance Task	Maintenance Task	MaintenanceTask Description	MaintenanceTask Description						
	Entity	Attributes	Input Name															
	Maintenance Task	MaintenanceTask	Maintenance Task															
	Maintenance Task	MaintenanceTask Description	MaintenanceTask Description															
	Step 3: The Administrator clicks on the "+ Icon" button.		Step 4: The system will load the "Maintenance Task" screen and displays it with the following controls and layout: <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Navigation</td><td>Menu-icon</td><td>Used for Navigation of the site.</td></tr><tr><td>Maintenance Task</td><td>Label</td><td>Heading for the pop-up displayed.</td></tr><tr><td>Maintenance Task Description</td><td>Label</td><td>Used to indicate what the corresponding text box is used for.</td></tr><tr><td>Maintenance Task Description</td><td>Text box</td><td>Used to capture text for the Maintenance Task Description details.</td></tr></table>	Text/Label	Control Type	Additional Info	Navigation	Menu-icon	Used for Navigation of the site.	Maintenance Task	Label	Heading for the pop-up displayed.	Maintenance Task Description	Label	Used to indicate what the corresponding text box is used for.	Maintenance Task Description	Text box	Used to capture text for the Maintenance Task Description details.
	Text/Label	Control Type	Additional Info															
	Navigation	Menu-icon	Used for Navigation of the site.															
	Maintenance Task	Label	Heading for the pop-up displayed.															
	Maintenance Task Description	Label	Used to indicate what the corresponding text box is used for.															
	Maintenance Task Description	Text box	Used to capture text for the Maintenance Task Description details.															

			<table><tr><td>Maintenance Group</td><td>Label</td><td>Used to Select the Maintenance Group for this maintenance Task and is Populated by contents from the ERD with options as indicated in the table below.</td></tr><tr><td>Create Maintenance Task</td><td>Button</td><td>Used to Save the new Maintenance Task Details.</td></tr><tr><td>Cancel</td><td>Button</td><td>Used to cancel the action of adding a Maintenance Task.</td></tr></table> The Maintenance Group Combo box options are populated with data from the following by using a FIRESQL READ query: <table><tr><th>Entity</th><th>Attributes</th></tr><tr><td>MaintenanceType</td><td>MaintenanceType</td></tr></table>	Maintenance Group	Label	Used to Select the Maintenance Group for this maintenance Task and is Populated by contents from the ERD with options as indicated in the table below.	Create Maintenance Task	Button	Used to Save the new Maintenance Task Details.	Cancel	Button	Used to cancel the action of adding a Maintenance Task.	Entity	Attributes	MaintenanceType	MaintenanceType
	Maintenance Group	Label	Used to Select the Maintenance Group for this maintenance Task and is Populated by contents from the ERD with options as indicated in the table below.													
Create Maintenance Task	Button	Used to Save the new Maintenance Task Details.														
Cancel	Button	Used to cancel the action of adding a Maintenance Task.														
Entity	Attributes															
MaintenanceType	MaintenanceType															
	Step 5: Administrator inputs the following Maintenance Task details: - Maintenance Task Description - Maintenance Type															

	Step 6: The Administrator clicks the “Create Maintenance Task” button. [alt]		Step 7: The system captures and ensures that all of the fields are filled in and an option is selected in the Combo box. [alt]								
			Step 8: The system validates that the captured details are in the correct format: [alt] Maintenance Task <i>table</i><table><tr><th>Attribute</th><th>Validation</th></tr><tr><td>MaintenanceTask</td><td>• Max 150 characters</td></tr><tr><td>Description</td><td>• Text characters only</td></tr></table>	Attribute	Validation	MaintenanceTask	• Max 150 characters	Description	• Text characters only		
	Attribute	Validation									
	MaintenanceTask	• Max 150 characters									
Description	• Text characters only										
		Step 9: The system verifies that no duplicate Maintenance Task exists on the system by comparing the following: [alt] <table><tr><th>Input</th><th>Attribute</th><th>Entity</th></tr><tr><td>Maintenance Task Description</td><td>MaintenanceTaskDescription</td><td>Maintenance Task</td></tr></table>	Input	Attribute	Entity	Maintenance Task Description	MaintenanceTaskDescription	Maintenance Task			
Input	Attribute	Entity									
Maintenance Task Description	MaintenanceTaskDescription	Maintenance Task									
		Step 10: The system prompts the Administrator to confirm to add the new Maintenance Task with the following: <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Confirmation</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed.</td></tr><tr><td>“Are you sure you want to add this Maintenance Task?”</td><td>Dialog Box Message</td><td>Used to inform the Administrator of the action.</td></tr></table>	Text/Label	Control Type	Additional Info	Confirmation	Dialog Box Title	Used to indicate what type of message is being displayed.	“Are you sure you want to add this Maintenance Task?”	Dialog Box Message	Used to inform the Administrator of the action.
Text/Label	Control Type	Additional Info									
Confirmation	Dialog Box Title	Used to indicate what type of message is being displayed.									
“Are you sure you want to add this Maintenance Task?”	Dialog Box Message	Used to inform the Administrator of the action.									

			Yes Button Used to confirm the action.								
			No Button Used to decline the action.								
	Step 11: The Administrator clicks the “Yes” button. [alt]		Step 12: The Maintenance Task is assigned a unique MaintenanceTaskID in the Maintenance Task table which is set to the next consecutive number after the previous MaintenanceTaskID in the database, the system then creates and saves the new Fault Task in the MaintenanceTask table. The following attributes are saved in the database by using a FIRESQL INSERT query: Maintenance Task Table MaintenanceTaskID MaintenanceTaskDescription								
			Step 13: The system notifies the Administrator that the Maintenance Task has been added successfully to the database with the following: [alt] <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Success</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed.</td></tr><tr><td>“Maintenance Task has been added successfully”</td><td>Dialog Box Message</td><td>Used to inform the Administrator of the result.</td></tr></table>	Text/Label	Control Type	Additional Info	Success	Dialog Box Title	Used to indicate what type of message is being displayed.	“Maintenance Task has been added successfully”	Dialog Box Message
Text/Label	Control Type	Additional Info									
Success	Dialog Box Title	Used to indicate what type of message is being displayed.									
“Maintenance Task has been added successfully”	Dialog Box Message	Used to inform the Administrator of the result.									

			Ok Button Used to accept the result.									
			Step 14: The system creates and saves the following in the database by using a FIRESQL INSERT query: AuditTrail Table AuditTrailID DateTime TransactionDescription UserID									
Alternate Courses:	Alt Step 6: The Administrator clicks the "Cancel" button. Terminate Use Case.											
	Alt Step 7: In the case of an empty Text box, the system notifies the Administrator of the Input error below the input field with the following:											
	<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>"Please fill this field."</td><td>Label (with a light red background color fill)</td><td>Used to inform the Administrator of the result.</td></tr><tr><td>"X"</td><td>Icon</td><td>Used to indicate an error result.</td></tr></table>			Text/Label	Control Type	Additional Info	"Please fill this field."	Label (with a light red background color fill)	Used to inform the Administrator of the result.	"X"	Icon	Used to indicate an error result.
	Text/Label	Control Type	Additional Info									
"Please fill this field."	Label (with a light red background color fill)	Used to inform the Administrator of the result.										
"X"	Icon	Used to indicate an error result.										
Redirect to Step 5.												
	Alt Step 7b: In the case of a Combo box option not selected, the system notifies the Administrator of the Input error below the input field with the following:											
	<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr></table>			Text/Label	Control Type	Additional Info						
Text/Label	Control Type	Additional Info										

	"Please select an option."	Label (with a light red background color fill)	Used to inform the Administrator of the result.
	"X"	Icon	Used to indicate an error result.
	Redirect to Step 5.		
	Alt Step 8: In the case of invalid input being entered, the system notifies the Administrator of the Format error below the input field with the following:		
	Text/Label	Control Type	Additional Info
	"Invalid input."	Label (with a light red background color fill)	Used to inform the Administrator of the result.
	"X"	Icon	Used to indicate an error result.
	Redirect to Step 5.		
	Alt Step 9: The inputted Maintenance Task already exists in the system; the system will notify the Administrator of the duplication error with the following:		
	Text/Label	Control Type	Additional Info
	Error	Dialog Box Title	Used to indicate what type of message is being displayed.
	"The Maintenance Task already	Dialog Box Message	Used to inform the Administrator of the result.

	exists. Please try again."		
	"Help icon"	Button	Used to Load the Help document.
	Ok	Button	Used to accept the result.
	Redirect to Step 5.		
	Alt Step 11: The Administrator declines to add the new Maintenance Task by clicking the "No" button. Redirect to Step 5.		
	Alt Step 13: The Maintenance Task was not added successfully to the database, the system will notify the Administrator of the save error with the following:		
	Text/Label	Control Type	Additional Info
	Failed	Dialog Box Title	Used to indicate what type of message is being displayed
	"Create unsuccessful. Please try again."	Dialog Box Message	Used to inform the Administrator of the result.
	"Help icon"	Button	Used to Load the Help document.
	Ok	Button	Used to accept the result.
	Redirect to Step 5.		
	Conclusion:		
	The use case concludes when the Administrator has been notified that the Maintenance Task has been added successfully.		

Post-Condition:	• A new Maintenance Task is added to the system with the relevant details. • The AuditTrail table is successfully updated. • The MaintenanceTask table is successfully updated.
Business Rules	• Only an Administrator can add a Maintenance Task to the system.
Implementation Constraints and Specifications:	• None
Assumptions:	• None
Open Issues:	• None

2.3.9.6. Search Maintenance Task

System Name:	VVs
Author:	Gareth Phillips
Date:	15/06/2020
Version:	1.0

Use Case Name:	Search Maintenance Task	Use Case Type:
Use Case ID:	8.6	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	Administrator	
Primary System Actor	None	
Other Participating Actors:	None	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process of an Administrator searching a Maintenance Task on the system. The Use Case starts when the Administrator navigates by selecting the " Maintenance Task Management" option under the "System Management" Combo box in the Navigation bar. The system populates the table with the information of all the Maintenance Task.	
Pre-Condition:	The Administrator must be logged into the system.	
Trigger:	The Administrator requests to search for a Maintenance Task on the system.	

Typical Course Of Events:	Actor Action	System Response																				
		Manual Action	Automated Action																			
	Step 1: The Administrator selects the "Maintenance Task Management" option under the "System Management" Combo box in the Navigation bar.		Step 2: The system Displays the "Maintenance Task Management" screen and populates it with the following details:																			
			<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>" + Icon"</td><td>Button</td><td>Used to load the "Maintenance Task" screen.</td></tr><tr><td>Search</td><td>Text-Box</td><td>Used for capturing Search criteria for the specific Maintenance Task.</td></tr><tr><td>Maintenance Task Management</td><td>Table</td><td>Displays a list of Maintenance Task.</td></tr><tr><td>"Pen icon"</td><td>Button</td><td>Used to navigate to the specific Maintenance Task view.</td></tr><tr><td>Search</td><td>Button</td><td>Used to perform the search action.</td></tr></table>		Text/Label	Control Type	Additional Info	" + Icon"	Button	Used to load the "Maintenance Task" screen.	Search	Text-Box	Used for capturing Search criteria for the specific Maintenance Task.	Maintenance Task Management	Table	Displays a list of Maintenance Task.	"Pen icon"	Button	Used to navigate to the specific Maintenance Task view.	Search	Button	Used to perform the search action.
			Text/Label	Control Type	Additional Info																	
			" + Icon"	Button	Used to load the "Maintenance Task" screen.																	
			Search	Text-Box	Used for capturing Search criteria for the specific Maintenance Task.																	
			Maintenance Task Management	Table	Displays a list of Maintenance Task.																	
			"Pen icon"	Button	Used to navigate to the specific Maintenance Task view.																	
			Search	Button	Used to perform the search action.																	
The table is populated by using a FIRESQL READ query with data from the following:																						

			<table><tr><th>Entity</th><th>Attributes</th><th>Input Name</th></tr><tr><td>Maintenance Task</td><td>MaintenanceTask Description</td><td>MaintenanceTask Description</td></tr><tr><td>MaintenanceType</td><td>MaintenanceType</td><td>Maintenance Type</td></tr></table>	Entity	Attributes	Input Name	Maintenance Task	MaintenanceTask Description	MaintenanceTask Description	MaintenanceType	MaintenanceType	Maintenance Type
	Entity	Attributes	Input Name									
	Maintenance Task	MaintenanceTask Description	MaintenanceTask Description									
	MaintenanceType	MaintenanceType	Maintenance Type									
Step 3: The Administrator enters the search criteria that they wish to search by. They do so by typing in a keyword in the "Search" Text box and then click the search button.		Step 4: The system uses a FIRESQL READ query to READ the following details in the database: [alt] <table><tr><th>Entity</th><th>Attributes</th></tr><tr><td>Maintenance Task</td><td>MaintenanceTaskDescription</td></tr><tr><td>MaintenanceType</td><td>MaintenanceType</td></tr></table>	Entity	Attributes	Maintenance Task	MaintenanceTaskDescription	MaintenanceType	MaintenanceType				
Entity	Attributes											
Maintenance Task	MaintenanceTaskDescription											
MaintenanceType	MaintenanceType											
		Step 5: The following details are loaded and used to populate the "Maintenance Task Management" table by using a FIRESQL READ query: <table><tr><th>Output</th><th>Attribute</th><th>Entity</th></tr><tr><td>Maintenance Task Description</td><td>MaintenanceTaskDescription</td><td>Maintenance Task</td></tr><tr><td>MaintenanceType</td><td>MaintenanceType</td><td>Maintenance Type</td></tr></table>	Output	Attribute	Entity	Maintenance Task Description	MaintenanceTaskDescription	Maintenance Task	MaintenanceType	MaintenanceType	Maintenance Type	
Output	Attribute	Entity										
Maintenance Task Description	MaintenanceTaskDescription	Maintenance Task										
MaintenanceType	MaintenanceType	Maintenance Type										
Step 6: The Administrator clicks on the "Pen icon" Maintenance Task Button in the "Action" table column.		Step 7: The System will load the "Maintenance Task" screen and displays it with the following controls and layout: <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr></table>	Text/Label	Control Type	Additional Info							
Text/Label	Control Type	Additional Info										

			Navigation	Navigation-Bar	Used for Navigation of the site.
			Maintenance Task	Label	Heading for the pop-up displayed.
			Maintenance Group	Label	Used to Select the Maintenance Group for this maintenance Task and is Populated by contents from the ERD with options as indicated in the table below.
			Maintenance Task Description	Label	Used to indicate what the corresponding text box is used for.
			Maintenance Task Description	Text box	Used to capture text for the Maintenance Task Description details.
			Delete Maintenance Task	Button	Used to Delete the Maintenance Task Details.

			Update Maintenance Task Button Used to Update the Maintenance Task Details. The following fields are populated with data retrieved from the database where the MaintenanceTaskID is the same as <table><thead><tr><th>Output</th><th>Attribute</th><th>Entity</th></tr></thead><tbody><tr><td>MaintenanceTaskID</td><td>MaintenanceTaskID</td><td>Maintenance Task</td></tr><tr><td>Maintenance Task Description</td><td>MaintenanceTaskDescription</td><td>Maintenance Task</td></tr></tbody></table> the one searched by using a FIRESQL READ query: The Maintenance Group Combo box options are populated with data from the following by using a FIRESQL READ query: <table><thead><tr><th>Entity</th><th>Attributes</th></tr></thead><tbody><tr><td>MaintenanceType</td><td>MaintenanceType</td></tr></tbody></table>	Output	Attribute	Entity	MaintenanceTaskID	MaintenanceTaskID	Maintenance Task	Maintenance Task Description	MaintenanceTaskDescription	Maintenance Task	Entity	Attributes	MaintenanceType	MaintenanceType
	Output	Attribute	Entity													
MaintenanceTaskID	MaintenanceTaskID	Maintenance Task														
Maintenance Task Description	MaintenanceTaskDescription	Maintenance Task														
Entity	Attributes															
MaintenanceType	MaintenanceType															
			Step 8: The system creates and saves the following in the database by using a FIRESQL INSERT query: <table><thead><tr><th>AuditTrail Table</th></tr></thead><tbody><tr><td>AuditTrailID</td></tr><tr><td>DateTime</td></tr><tr><td>TransactionDescription</td></tr><tr><td>UserID</td></tr></tbody></table>	AuditTrail Table	AuditTrailID	DateTime	TransactionDescription	UserID								
AuditTrail Table																
AuditTrailID																
DateTime																
TransactionDescription																
UserID																

Alternate Courses:	Alt Step 4: The specific Maintenance Task does not exist in the database. The system will notify the Administrator of this with the following: <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>"Maintenance Task not found."</td><td>Label</td><td>Used to inform the Administrator of the result.</td></tr><tr><td>"X"</td><td>Banner (with a light red background color fill)</td><td>Used to display the result message.</td></tr></table> Redirect to Step 3.	Text/Label	Control Type	Additional Info	"Maintenance Task not found."	Label	Used to inform the Administrator of the result.	"X"	Banner (with a light red background color fill)	Used to display the result message.
Text/Label	Control Type	Additional Info								
"Maintenance Task not found."	Label	Used to inform the Administrator of the result.								
"X"	Banner (with a light red background color fill)	Used to display the result message.								
Conclusion:	The use case narrative concludes when the searched Maintenance Task details are displayed.									
Post-Condition:	• The Maintenance Task has been searched successfully. • The AuditTrail table is successfully updated.									
Business Rules	• Only an Administrator can search for a Maintenance Task on the system.									
Implementation Constraints and Specifications:	• None									
Assumptions:	• None									
Open Issues:	• None									

2.3.9.7. Update Maintenance Task

System Name:	VVs
Author:	Gareth Phillips
Date:	15/06/2020
Version:	1.0

Use Case Name:	Update Maintenance Task	Use Case Type:
Use Case ID:	8.7	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	Administrator	
Primary System Actor	None	
Other Participating Actors:	None	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process of an Administrator updating an existing Maintenance Task's details in the system. The Use Case starts when the Administrator navigates by selecting the "Maintenance Task Management" option under the "System Management" Combo box in the Navigation bar. From there, they search for the Maintenance Task they want to update, the system then will load the "Maintenance Task" screen with the Maintenance Task details. The Administrator can then make their changes and save it.	
Pre-Condition:	The Administrator must be logged into the system.	

Trigger:	The Administrator requests to update an existing Maintenance Task to the system.						
Typical Course Of Events:	Actor Action	System Response					
		Manual Action	Automated Action				
	Step 1: The Administrator selects the “Maintenance Task Management” option under the “System Management” Combo box in the Navigation bar.		Step 2: The System invokes Use Case 8.6 “Search Maintenance Task” and displays the “Maintenance Task” screen with the data of the searched Maintenance Task.				
	Step 3: Administrator updates the following Maintenance Task details: - Maintenance Task Description - Maintenance Type						
	Step 4: The Administrator clicks the “Update Maintenance Task” button. [alt]		Step 5: The system captures and ensures that all of the fields are filled in and an option is selected in the Combo box. [alt]				
			Step 6: The system validates that the captured details are in the correct format: [alt] Maintenance Task <i>table</i> <table><thead><tr><th>Attribute</th><th>Validation</th></tr></thead><tbody><tr><td>MaintenanceTaskDescription</td><td> - Max 150 characters - Text characters only </td></tr></tbody></table>	Attribute	Validation	MaintenanceTaskDescription	- Max 150 characters - Text characters only
	Attribute	Validation					
MaintenanceTaskDescription	- Max 150 characters - Text characters only						
		Step 7: The system prompts the Administrator to confirm the update of the Maintenance Task’s details. <table><thead><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr></thead></table>	Text/Label	Control Type	Additional Info		
Text/Label	Control Type	Additional Info					

			Confirmation	Dialog Box Title	Used to indicate what type of message is being displayed.
			"Are you sure you want to update this Maintenance Task?"	Dialog Box Message	Used to inform the Administrator of the action.
			Yes	Button	Used to confirm the action.
			No	Button	Used to decline the action.
	Step 8: The Administrator clicks the "Yes" button. [alt]		Step 9: The system then updates and saves the updated Maintenance Task details in the MaintenanceTask. The following attributes are updated in the database by using a FIRESQL UPDATE query: Maintenance Task Table MaintenanceTaskID MaintenanceTaskDescription MaintenanceType		
		Step 10: The system notifies the Administrator that the Maintenance Task has been updated successfully in the database with the following: [alt]			
			Text/Label	Control Type	Additional Info

			Success	Dialog Box Title	Used to indicate what type of message is being displayed.
			"The Maintenance Task has been updated successfully"	Dialog Box Message	Used to inform the Administrator of the result.
			Ok	Button	Used to accept the result.
			Step 11: The system updates and saves the following in the database by using a FIRESQL INSERT query: AuditTrail Table AuditTrailID DateTime TransactionDescription UserID		
Alternate Courses:	Alt Step 4: The Administrator clicks the "Cancel" button. Terminate Use Case.				
	Alt Step 5a: In the case of an empty Text box, the system notifies the Administrator of the Input error below the input field with the following:				
	Text/Label	Control Type	Additional Info		
	"Please fill this field."	Label (with a light red background color fill)	Used to inform the Administrator of the result.		

	Text/Label	Control Type	Additional Info
	Failed	Dialog Box Title	Used to indicate what type of message is being displayed
	"Update unsuccessful. Please try again."	Dialog Box Message	Used to inform the Administrator of the result.
	"Help icon"	Button	Used to Load the Help document.
	Ok	Button	Used to accept the result.
	Redirect to Step 3.		
Conclusion:	The use case concludes when the Administrator has been notified that the Maintenance Task has been updated successfully.		
Post-Condition:	• An existing Maintenance Task is updated in the system with the relevant details. • The AuditTrail table is successfully updated. • The MaintenanceTask table is successfully updated.		
Business Rules	• Only an Administrator can update a Maintenance Task to the system.		
Implementation Constraints and Specifications:	• None		
Assumptions:	• None		
Open Issues:	• None		

2.3.9.8. Delete Maintenance Task

System Name:	VVs
Author:	Gareth Phillips
Date:	15/06/2020
Version:	1.0

Use Case Name:	Delete Maintenance Task	Use Case Type:
Use Case ID:	8.8	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	Administrator	
Primary System Actor	None	
Other Participating Actors:	None	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process of an Administrator deleting an existing Maintenance Task's details from the system. The Use Case starts when the Administrator navigates by selecting the "Maintenance Task Management" option under the "System Management" Combo box in the Navigation bar. From there, they search for the Maintenance Task they want to delete and the system displays the "Maintenance Task" screen with the Maintenance Task details to be deleted. The system populates the relevant Labels and corresponding text boxes with information of the searched Maintenance Task, and the Administrator can make their deletion.	
Pre-Condition:	The Administrator must be logged into the system.	

Trigger:	The Administrator requests to delete an existing Maintenance Task from the system.																
Typical Course Of Events:	Actor Action	System Response															
		Manual Action	Automated Action														
	Step 1: The Administrator selects the “Maintenance Task Management” option under the “System Management” Combo box in the Navigation bar.		Step 2: The System invokes Use Case 8.6 “Search Maintenance Task” and displays the “Maintenance Task” screen with the data of the searched Maintenance Task.														
	Step 3: The Administrator clicks on the “Delete Maintenance Task” button. [alt]		Step 4: The system prompts the Administrator to confirm the deletion of the Maintenance Task’s details with the following:														
			<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Confirmation</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed.</td></tr><tr><td>“Are you sure you want to delete this Maintenance Task?”</td><td>Dialog Box Message</td><td>Used to inform the Administrator of the action.</td></tr><tr><td>Yes</td><td>Button</td><td>Used to confirm the action.</td></tr><tr><td>No</td><td>Button</td><td>Used to decline the action.</td></tr></table>	Text/Label	Control Type	Additional Info	Confirmation	Dialog Box Title	Used to indicate what type of message is being displayed.	“Are you sure you want to delete this Maintenance Task?”	Dialog Box Message	Used to inform the Administrator of the action.	Yes	Button	Used to confirm the action.	No	Button
Text/Label			Control Type	Additional Info													
Confirmation			Dialog Box Title	Used to indicate what type of message is being displayed.													
“Are you sure you want to delete this Maintenance Task?”	Dialog Box Message	Used to inform the Administrator of the action.															
Yes	Button	Used to confirm the action.															
No	Button	Used to decline the action.															

	Step 5: The Administrator clicks the “Yes” button. [alt]		Step 6: The system then removes the Maintenance Task details in the MaintenanceTask table. The following attributes are deleted in the database by using a FIRESQL DELETE query: Maintenance Task Table MaintenanceTaskID MaintenanceTaskDescription MaintenanceType												
			Step 7: The system notifies the Administrator that the Maintenance Task has been deleted successfully from the database with the following: [alt] <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Success</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed.</td></tr><tr><td>“The Maintenance Task has been deleted successfully”</td><td>Dialog Box Message</td><td>Used to inform the Administrator of the result.</td></tr><tr><td>Ok</td><td>Button</td><td>Used to accept the result.</td></tr></table>	Text/Label	Control Type	Additional Info	Success	Dialog Box Title	Used to indicate what type of message is being displayed.	“The Maintenance Task has been deleted successfully”	Dialog Box Message	Used to inform the Administrator of the result.	Ok	Button	Used to accept the result.
	Text/Label	Control Type	Additional Info												
Success	Dialog Box Title	Used to indicate what type of message is being displayed.													
“The Maintenance Task has been deleted successfully”	Dialog Box Message	Used to inform the Administrator of the result.													
Ok	Button	Used to accept the result.													
		Step 8: The system updates and saves the following in the database by using a FIRESQL INSERT query: AuditTrail Table													

			AuditTrailID DateTime TransactionDescription UserID
Alternate Courses:	Alt Step 3: The Administrator clicks the "Cancel" button. Terminate Use Case.		
	Alt Step 5: The Administrator declines to delete the Maintenance Task by clicking the "No" button. Redirect to Step 2.		
	Alt Step 7: The Administrator is notified of the unsuccessful delete action with the following:		
	Text/Label	Control Type	Additional Info
	Failed	Dialog Box Title	Used to indicate what type of message is being displayed
	"Delete unsuccessful. Please try again."	Dialog Box Message	Used to inform the Administrator of the result.
	"Help icon"	Button	Used to Load the Help document.
	Ok	Button	Used to accept the result.
	Redirect to Step 2.		
Conclusion:	The use case concludes when the Administrator has been notified that the Maintenance Task has been successfully deleted.		
Post-Condition:	An existing Maintenance Task is deleted from the system with all the relevant details being removed.		

	• The AuditTrail table is successfully updated. • The Maintenance Task table is successfully updated with the deleted details being removed.
Business Rules	• Only an Administrator can delete a Maintenance Task from the system.
Implementation Constraints and Specifications:	• None
Assumptions:	• None
Open Issues:	• None

2.3.10. Sub-system 9: School

2.3.10.1. Add School

System Name:	VVs
Author:	Tapiwarufaro Ndoro
Date:	15/06/2020
Version:	1.0

Use Case Name:	Add School	Use Case Type:
Use Case ID:	9.1	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	Administrator	
Primary System Actor	None	
Other Participating Actors:	None	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process of an Administrator adding a new School to the system. The Use Case starts when the Administrator navigates by selecting the "School Management" option under the "System Management" Combo box in the Navigation bar. From there, they click the "Add School" button and the system will load the "School" screen. The system populates the relevant Combo and Text boxes with information necessary for selection, and the corresponding labels, buttons, icons, logos, and banners.	

Pre-Condition:	The Administrator must be logged into the system.																				
Trigger:	The Administrator requests to add a new School to the system.																				
Typical Course Of Events:	Actor Action	System Response																			
		Manual Action	Automated Action																		
	Step 1: The Administrator selects the “School Management” option under the “System Management” Combo box in the Navigation bar.		Step 2: The system Displays the “School Management” screen and populates it with the following details:																		
			<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>+ icon</td><td>Button</td><td>Used to load the “School” screen.</td></tr><tr><td>Search</td><td>Text-Box</td><td>Used for capturing Search criteria for the specific School.</td></tr><tr><td>School Management</td><td>Table</td><td>Displays a list of Schools.</td></tr><tr><td>“Pen icon”</td><td>Button</td><td>Used to navigate to the specific School view.</td></tr><tr><td>Search</td><td>Button</td><td>Used to perform the search action.</td></tr></table>	Text/Label	Control Type	Additional Info	+ icon	Button	Used to load the “School” screen.	Search	Text-Box	Used for capturing Search criteria for the specific School.	School Management	Table	Displays a list of Schools.	“Pen icon”	Button	Used to navigate to the specific School view.	Search	Button	Used to perform the search action.
			Text/Label	Control Type	Additional Info																
			+ icon	Button	Used to load the “School” screen.																
			Search	Text-Box	Used for capturing Search criteria for the specific School.																
			School Management	Table	Displays a list of Schools.																
			“Pen icon”	Button	Used to navigate to the specific School view.																
			Search	Button	Used to perform the search action.																
The table is populated by using a FIRESQL READ query with data from the following:																					
<table><tr><th>Entity</th><th>Attributes</th><th>Input Name</th></tr><tr><td>School</td><td>SchoolName</td><td>School Name</td></tr></table>	Entity	Attributes	Input Name	School	SchoolName	School Name															
Entity	Attributes	Input Name																			
School	SchoolName	School Name																			

			Village	VillageName	Village
	Step 3: The Administrator clicks on the "+ icon" button.		Step 4: The system will load the "School" screen and displays it with the following controls and layout:		
			Text/Label	Control Type	Additional Info
			Navigation	Navigation-bar	Used for Navigation of the site.
			School	Label	Heading for the pop-up displayed.
			School Name	Label	Used to indicate what the corresponding text box is used for.
			School Email	Label	Used to indicate what the corresponding text box is used for.
			School Telephone	Label	Used to indicate what the corresponding text box is used for.
			Country	Label	Used to indicate what the corresponding

					Combo box is used for.
			Province	Label	Used to indicate what the corresponding Combo box is used for.
			City	Label	Used to indicate what the corresponding Combo box is used for.
			Suburb	Label	Used to indicate what the corresponding Combo box is used for.
			Village	Label	Used to indicate what the corresponding Combo box is used for.
			School Name	Text box	Used to capture text for the School Name details.
			School Email	Text box	Used to capture text for the School Email details.

			School Telephone	Text box	Used to capture text for the School Telephone details.
			Country	Label	Used to Select the Country for this School and is Populated by contents from the ERD with options as indicated in the table below.
			Province	Label	Used to Select the Province for this School and is Populated by contents from the ERD with options as indicated in the table below.
			City	Label	Used to Select the City for this School and is Populated by contents from the ERD with options as indicated in the table below.
			Suburb	Label	Used to Select the Suburb for this School and is

			Populated by contents from the ERD with options as indicated in the table below.	
			Village	Label
			Used to Select the Village for this School and is Populated by contents from the ERD with options as indicated in the table below.	
			Create School	Button
			Used to Save the new School Details.	
			Cancel	Button
			Used to cancel the action of adding a School.	
			The Country Combo box options are populated with data from the following by using a FIRESQL READ query:	
			Entity	Attributes
			Country	CountryName
			The Province Combo box options are populated with data from the following by using a FIRESQL READ query:	
			Entity	Attributes

		ProvinceProvinceName The City Combo box options are populated with data from the following by using a FIRESQL READ query: <table><tr><th>Entity</th><th>Attributes</th></tr><tr><td>City</td><td>CityName</td></tr></table> The Suburb Combo box options are populated with data from the following by using a FIRESQL READ query: <table><tr><th>Entity</th><th>Attributes</th></tr><tr><td>Suburb</td><td>SuburbName</td></tr></table> The Village Combo box options are populated with data from the following by using a FIRESQL READ query: <table><tr><th>Entity</th><th>Attributes</th></tr><tr><td>Village</td><td>VillageName</td></tr></table>	Entity	Attributes	City	CityName	Entity	Attributes	Suburb	SuburbName	Entity	Attributes	Village	VillageName
Entity	Attributes													
City	CityName													
Entity	Attributes													
Suburb	SuburbName													
Entity	Attributes													
Village	VillageName													
Step 5: Administrator inputs the following School details: - School Name - School Email - School Telephone - Country - Province - City - Suburb - Village														

	Step 6: The Administrator clicks the “Create School” button. [alt]		Step 7: The system captures and ensures that all of the fields are filled in and an option is selected in the Combo box. [alt]								
			Step 8: The system validates that the captured details are in the correct format: [alt] School table <table><tr><th>Attribute</th><th>Validation</th></tr><tr><td>SchoolName</td><td>42.Max 50 characters 43.Text characters only</td></tr><tr><td>SchoolEmail</td><td>44.Max 50 characters 45.Text characters only</td></tr><tr><td>SchoolTelephone</td><td> - Max 50 characters - Text characters only </td></tr></table>	Attribute	Validation	SchoolName	42.Max 50 characters 43.Text characters only	SchoolEmail	44.Max 50 characters 45.Text characters only	SchoolTelephone	- Max 50 characters - Text characters only
	Attribute	Validation									
	SchoolName	42.Max 50 characters 43.Text characters only									
SchoolEmail	44.Max 50 characters 45.Text characters only										
SchoolTelephone	- Max 50 characters - Text characters only										
		Step 9: The system verifies that no duplicate School exists on the system by comparing the following: [alt] <table><tr><th>Input</th><th>Attribute</th><th>Entity</th></tr><tr><td>School Name</td><td>SchoolName</td><td>School</td></tr></table>	Input	Attribute	Entity	School Name	SchoolName	School			
Input	Attribute	Entity									
School Name	SchoolName	School									
		Step 10: The system prompts the Administrator to confirm to add the new School with the following: <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Confirmation</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed.</td></tr><tr><td>“Are you sure you want to add this School?”</td><td>Dialog Box Message</td><td>Used to inform the Administrator of the action.</td></tr></table>	Text/Label	Control Type	Additional Info	Confirmation	Dialog Box Title	Used to indicate what type of message is being displayed.	“Are you sure you want to add this School?”	Dialog Box Message	Used to inform the Administrator of the action.
Text/Label	Control Type	Additional Info									
Confirmation	Dialog Box Title	Used to indicate what type of message is being displayed.									
“Are you sure you want to add this School?”	Dialog Box Message	Used to inform the Administrator of the action.									

			<table><tr><td>Yes</td><td>Button</td><td>Used to confirm the action.</td></tr><tr><td>No</td><td>Button</td><td>Used to decline the action.</td></tr></table>	Yes	Button	Used to confirm the action.	No	Button	Used to decline the action.		
	Yes	Button	Used to confirm the action.								
	No	Button	Used to decline the action.								
	Step 11: The Administrator clicks the “Yes” button. [alt]		Step 12: The School is assigned a unique SchoolID in the School table which is set to the next consecutive number after the previous SchoolID in the database, the system then creates and saves the new School in the School table. The following attributes are saved in the database by using a FIRESQL INSERT query: <table><tr><td>School Table</td></tr><tr><td>SchoolID</td></tr><tr><td>SchoolName</td></tr><tr><td>SchoolEmail</td></tr><tr><td>SchoolTelephone</td></tr></table>	School Table	SchoolID	SchoolName	SchoolEmail	SchoolTelephone			
School Table											
SchoolID											
SchoolName											
SchoolEmail											
SchoolTelephone											
		Step 13: The system notifies the Administrator that the School has been added successfully to the database with the following: [alt] <table><tr><td>Text/Label</td><td>Control Type</td><td>Additional Info</td></tr><tr><td>Success</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed.</td></tr><tr><td>“School has been added successfully”</td><td>Dialog Box Message</td><td>Used to inform the Administrator of the result.</td></tr></table>	Text/Label	Control Type	Additional Info	Success	Dialog Box Title	Used to indicate what type of message is being displayed.	“School has been added successfully”	Dialog Box Message	Used to inform the Administrator of the result.
Text/Label	Control Type	Additional Info									
Success	Dialog Box Title	Used to indicate what type of message is being displayed.									
“School has been added successfully”	Dialog Box Message	Used to inform the Administrator of the result.									

			Ok Button Used to accept the result.									
			Step 14: The system creates and saves the following in the database by using a FIRESQL INSERT query: AuditTrail Table AuditID DateTime TransactionDescription UserID									
Alternate Courses:	Alt Step 6: The Administrator clicks the "Cancel" button. Terminate Use Case.											
	Alt Step 7a: In the case of an empty Text box, the system notifies the Administrator of the Input error below the input field with the following:											
	<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>"Please fill this field."</td><td>Label (with a light red background color fill)</td><td>Used to inform the Administrator of the result.</td></tr><tr><td>"X"</td><td>Icon</td><td>Used to indicate an error result.</td></tr></table>			Text/Label	Control Type	Additional Info	"Please fill this field."	Label (with a light red background color fill)	Used to inform the Administrator of the result.	"X"	Icon	Used to indicate an error result.
	Text/Label	Control Type	Additional Info									
	"Please fill this field."	Label (with a light red background color fill)	Used to inform the Administrator of the result.									
"X"	Icon	Used to indicate an error result.										
Redirect to Step 5.												
Alt Step 7b: In the case of a Combo box option not selected, the system notifies the Administrator of the Input error below the input field with the following:												
<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr></table>			Text/Label	Control Type	Additional Info							
Text/Label	Control Type	Additional Info										

	"Please select an option."	Label (with a light red background color fill)	Used to inform the Administrator of the result.
	"X"	Icon	Used to indicate an error result.
	Redirect to Step 5.		
	Alt Step 8: In the case of invalid input being entered, the system notifies the Administrator of the Format error below the input field with the following:		
	Text/Label	Control Type	Additional Info
	"Invalid input."	Label (with a light red background color fill)	Used to inform the Administrator of the result.
	"X"	Icon	Used to indicate an error result.
	Redirect to Step 5.		
	Alt Step 9: The inputted School already exists in the system; the system will notify the Administrator of the duplication error with the following:		
	Text/Label	Control Type	Additional Info
	Error	Dialog Box Title	Used to indicate what type of message is being displayed.
	"The School already exists. Please try again."	Dialog Box Message	Used to inform the Administrator of the result.

	"Help icon"	Button	Used to Load the Help document.
	Ok	Button	Used to accept the result.
	Redirect to Step 5.		
	Alt Step 11: The Administrator declines to add the new School by clicking the "No" button. Redirect to Step 5.		
	Alt Step 13: The School was not added successfully to the database, the system will notify the Administrator of the save error with the following:		
	Text/Label	Control Type	Additional Info
	Failed	Dialog Box Title	Used to indicate what type of message is being displayed
	"Create unsuccessful. Please try again."	Dialog Box Message	Used to inform the Administrator of the result.
	"Help icon"	Button	Used to Load the Help document.
	Ok	Button	Used to accept the result.
	Redirect to Step 5.		
Conclusion:	The use case concludes when the Administrator has been notified that the School has been added successfully.		
Post-Condition:	- A new School is added to the system with the relevant details. - The AuditTrail table is successfully updated.		

	The School table is successfully updated.
Business Rules	Only an Administrator can add a School to the system.
Implementation Constraints and Specifications:	None
Assumptions:	None
Open Issues:	None

2.3.10.2. Search School

System Name:	VVs
Author:	Tapiwarufaro Ndoro
Date:	15/06/2020
Version:	1.0

Use Case Name:	Search School	Use Case Type:
Use Case ID:	9.2	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	Administrator	
Primary System Actor	None	
Other Participating Actors:	None	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process of an Administrator searching a School on the system. The Use Case starts when the Administrator navigates by selecting the "School Management" option under the "System Management" Combo box in the Navigation bar. The system populates the table with the information of all the Schools.	
Pre-Condition:	The Administrator must be logged into the system.	
Trigger:	The Administrator requests to search for a School on the system.	

Typical Course Of Events:	Actor Action	System Response																				
		Manual Action	Automated Action																			
	Step 1: The Administrator selects the “School Management” option under the “System Management” Combo box in the Navigation bar.		Step 2: The system Displays the “School Management” screen and populates it with the following details:																			
			<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>+ icon</td><td>Button</td><td>Used to load the “School” screen.</td></tr><tr><td>Search</td><td>Text-Box</td><td>Used for capturing Search criteria for the specific School.</td></tr><tr><td>School Management</td><td>Table</td><td>Displays a list of Schools.</td></tr><tr><td>“Pen icon”</td><td>Button</td><td>Used to navigate to the specific School view.</td></tr><tr><td>Search</td><td>Button</td><td>Used to perform the search action.</td></tr></table>		Text/Label	Control Type	Additional Info	+ icon	Button	Used to load the “School” screen.	Search	Text-Box	Used for capturing Search criteria for the specific School.	School Management	Table	Displays a list of Schools.	“Pen icon”	Button	Used to navigate to the specific School view.	Search	Button	Used to perform the search action.
			Text/Label	Control Type	Additional Info																	
			+ icon	Button	Used to load the “School” screen.																	
			Search	Text-Box	Used for capturing Search criteria for the specific School.																	
			School Management	Table	Displays a list of Schools.																	
			“Pen icon”	Button	Used to navigate to the specific School view.																	
			Search	Button	Used to perform the search action.																	
			The table is populated with data from the following by using a FIRESQL READ query:																			
			<table><tr><th>Entity</th><th>Attributes</th><th>Input Name</th></tr><tr><td>School</td><td>SchoolName</td><td>School Name</td></tr><tr><td>Village</td><td>VillageName</td><td>Village</td></tr></table>		Entity	Attributes	Input Name	School	SchoolName	School Name	Village	VillageName	Village									
			Entity	Attributes	Input Name																	
			School	SchoolName	School Name																	
Village	VillageName	Village																				

	Step 3: The Administrator enters the search criteria that they wish to search by. They do so by typing in a keyword in the "Search" Text box then click the "Search" button.		Step 4: The system uses a FIRESQL READ query to READ the following details in the database: [alt] <table><tr><th>Entity</th><th>Attributes</th></tr><tr><td>School</td><td>SchoolName</td></tr></table>	Entity	Attributes	School	SchoolName											
	Entity	Attributes																
	School	SchoolName																
		Step 5: The following details are loaded and used to populate the "School Management" table by using a FIRESQL READ query: <table><tr><th>Output</th><th>Attribute</th><th>Entity</th></tr><tr><td>School Name</td><td>SchoolName</td><td>School</td></tr><tr><td>Village</td><td>VillageName</td><td>Village</td></tr></table>	Output	Attribute	Entity	School Name	SchoolName	School	Village	VillageName	Village							
Output	Attribute	Entity																
School Name	SchoolName	School																
Village	VillageName	Village																
	Step 6: The Administrator clicks on the "Pen icon" School Button in the "Action" table column.		Step 7: The System will load the "School" screen and displays it with the following controls and layout: <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Navigation</td><td>Navigation-bar</td><td>Used for Navigation of the site.</td></tr><tr><td>School</td><td>Label</td><td>Heading for the pop-up displayed.</td></tr><tr><td>School Name</td><td>Label</td><td>Used to indicate what the corresponding text box is used for.</td></tr><tr><td>School Email</td><td>Label</td><td>Used to indicate what the</td></tr></table>	Text/Label	Control Type	Additional Info	Navigation	Navigation-bar	Used for Navigation of the site.	School	Label	Heading for the pop-up displayed.	School Name	Label	Used to indicate what the corresponding text box is used for.	School Email	Label	Used to indicate what the
Text/Label	Control Type	Additional Info																
Navigation	Navigation-bar	Used for Navigation of the site.																
School	Label	Heading for the pop-up displayed.																
School Name	Label	Used to indicate what the corresponding text box is used for.																
School Email	Label	Used to indicate what the																

				corresponding text box is used for.
			School Telephone	Label Used to indicate what the corresponding text box is used for.
			Country	Label Used to indicate what the corresponding Combo box is used for.
			Province	Label Used to indicate what the corresponding Combo box is used for.
			City	Label Used to indicate what the corresponding Combo box is used for.
			Suburb	Label Used to indicate what the corresponding Combo box is used for.
			Village	Label Used to indicate what the

					corresponding Combo box is used for.
			School Name	Text box	Used to capture text for the School Name details.
			School Email	Text box	Used to capture text for the School Email details.
			School Telephone	Text box	Used to capture text for the School Telephone details.
			Country	Label	Used to Select the Country for this School and is Populated by contents from the ERD with options as indicated in the table below.
			Province	Label	Used to Select the Province for this School and is Populated by contents from the ERD with options

			as indicated in the table below.	
			City	Label
			Used to Select the City for this School and is Populated by contents from the ERD with options as indicated in the table below.	
			Suburb	Label
			Used to Select the Suburb for this School and is Populated by contents from the ERD with options as indicated in the table below.	
			Village	Label
			Used to Select the Village for this School and is Populated by contents from the ERD with options as indicated in the table below.	
			Delete	Button
			Used to delete a School.	
			Update	Button
			Used to update a School.	

The Country Combo box options are populated with data from the following by using a FIRESQL READ query:

Entity	Attributes
Country	CountryName

The Province Combo box options are populated with data from the following by using a FIRESQL READ query:

Entity	Attributes
Province	ProvinceName

The City Combo box options are populated with data from the following by using a FIRESQL READ query:

Entity	Attributes
City	CityName

The Suburb Combo box options are populated with data from the following by using a FIRESQL READ query:

Entity	Attributes
Suburb	SuburbName

The Village Combo box options are populated with data from the following by using a FIRESQL READ query:

Entity	Attributes
Village	VillageName

			The following fields are populated with data retrieved from the database where the SchoolID is the same as the one searched and data linked to it by using a FIRESQL READ query: <table><tr><th>Output</th><th>Attribute</th><th>Entity</th></tr><tr><td>SchoolID</td><td>SchoolID</td><td>School</td></tr><tr><td>School Name</td><td>SchoolName</td><td>School</td></tr><tr><td>School Email</td><td>SchoolEmail</td><td>School</td></tr><tr><td>School Telephone</td><td>SchoolTelephone</td><td>School</td></tr><tr><td>Country</td><td>CountryName</td><td>Country</td></tr><tr><td>Province</td><td>ProvinceName</td><td>Province</td></tr><tr><td>City</td><td>CityName</td><td>City</td></tr><tr><td>Suburb</td><td>SuburbName</td><td>Suburb</td></tr><tr><td>Village</td><td>VillageName</td><td>Village</td></tr></table>	Output	Attribute	Entity	SchoolID	SchoolID	School	School Name	SchoolName	School	School Email	SchoolEmail	School	School Telephone	SchoolTelephone	School	Country	CountryName	Country	Province	ProvinceName	Province	City	CityName	City	Suburb	SuburbName	Suburb	Village	VillageName	Village
	Output	Attribute	Entity																														
	SchoolID	SchoolID	School																														
	School Name	SchoolName	School																														
	School Email	SchoolEmail	School																														
	School Telephone	SchoolTelephone	School																														
	Country	CountryName	Country																														
	Province	ProvinceName	Province																														
	City	CityName	City																														
	Suburb	SuburbName	Suburb																														
Village	VillageName	Village																															
		Step 8: The system creates and saves the following in the database by using a FIRESQL INSERT query: AuditTrail Table AuditID DateTime TransactionDescription UserID																															

Alternate Courses:	Alt Step 4: The specific School does not exist in the database. The system will notify the Administrator of this with the following:		
	Text/Label	Control Type	Additional Info
	"School not found."	Label	Used to inform the Administrator of the result.
	"X"	Banner (with a light red background color fill)	Used to display the result message.
	Redirect to Step 3.		
Conclusion:	The use case narrative concludes when the searched School details are displayed.		
Post-Condition:	- The School has been searched successfully. - The AuditTrail table is successfully updated.		
Business Rules	Only an Administrator can search for a School on the system.		
Implementation Constraints and Specifications:	None		
Assumptions:	None		
Open Issues:	None		

2.3.10.3. Update School

System Name:	VVs
Author:	Tapiwarufaro Ndoro
Date:	15/06/2020
Version:	1.0

Use Case Name:	Update School	Use Case Type:
Use Case ID:	9.3	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	Administrator	
Primary System Actor	None	
Other Participating Actors:	None	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process of an Administrator updating an existing School's details in the system. The Use Case starts when the Administrator navigates by selecting the "School Management" option under the "System Management" Combo box in the Navigation bar. From there, they search for the School they want to update, and the system will load the "School" screen with the School details. The system populates the relevant Combo and Text boxes with information of the searched School, and the Administrator can make their changes and save.	
Pre-Condition:	The Administrator must be logged into the system.	

Trigger:	The Administrator requests to update an existing School to the system.								
Typical Course Of Events:	Actor Action	System Response							
		Manual Action	Automated Action						
	Step 1: The Administrator selects the “School Management” option under the “System Management” Combo box in the Navigation bar.		Step 2: The System invokes Use Case 9.2 “Search School” and displays the “School” screen with the data of the searched School.						
	Step 3: Administrator updates the following School details: • School Name • School Email • School Telephone • Country • Province • City • Suburb • Village								
	Step 4: The Administrator clicks the “Update” button. [alt]		Step 5: The system captures and ensures that all of the fields are filled in and an option is selected in the Combo box. [alt]						
			Step 6: The system validates that the captured details are in the correct format: [alt] School table <table><tr><th>Attribute</th><th>Validation</th></tr><tr><td>SchoolName</td><td>46.Max 50 characters 47.Text characters only</td></tr><tr><td>SchoolEmail</td><td>48.Max 50 characters</td></tr></table>	Attribute	Validation	SchoolName	46.Max 50 characters 47.Text characters only	SchoolEmail	48.Max 50 characters
	Attribute	Validation							
SchoolName	46.Max 50 characters 47.Text characters only								
SchoolEmail	48.Max 50 characters								

			49.Text characters only SchoolTelephone • Max 50 characters • Text characters only															
			Step 7: The system prompts the Administrator to confirm the update of the School's details. <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Confirmation</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed.</td></tr><tr><td>"Are you sure you want to update this School?"</td><td>Dialog Box Message</td><td>Used to inform the Administrator of the action.</td></tr><tr><td>Yes</td><td>Button</td><td>Used to confirm the action.</td></tr><tr><td>No</td><td>Button</td><td>Used to decline the action.</td></tr></table>	Text/Label	Control Type	Additional Info	Confirmation	Dialog Box Title	Used to indicate what type of message is being displayed.	"Are you sure you want to update this School?"	Dialog Box Message	Used to inform the Administrator of the action.	Yes	Button	Used to confirm the action.	No	Button	Used to decline the action.
	Text/Label	Control Type	Additional Info															
	Confirmation	Dialog Box Title	Used to indicate what type of message is being displayed.															
	"Are you sure you want to update this School?"	Dialog Box Message	Used to inform the Administrator of the action.															
	Yes	Button	Used to confirm the action.															
	No	Button	Used to decline the action.															
	Step 8: The Administrator clicks the "Yes" button. [alt]		Step 9: The system then updates and saves the updated School details in the School table. The following attributes are updated in the database by using a FIRESQL UPDATE query: School Table SchoolID SchoolName SchoolEmail SchoolTelephone															

			Step 10: The system notifies the Administrator that the School has been updated successfully in the database with the following: [alt] <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Success</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed.</td></tr><tr><td>"The School has been updated successfully"</td><td>Dialog Box Message</td><td>Used to inform the Administrator of the result.</td></tr><tr><td>Ok</td><td>Button</td><td>Used to accept the result.</td></tr></table>	Text/Label	Control Type	Additional Info	Success	Dialog Box Title	Used to indicate what type of message is being displayed.	"The School has been updated successfully"	Dialog Box Message	Used to inform the Administrator of the result.	Ok	Button	Used to accept the result.
	Text/Label	Control Type	Additional Info												
Success	Dialog Box Title	Used to indicate what type of message is being displayed.													
"The School has been updated successfully"	Dialog Box Message	Used to inform the Administrator of the result.													
Ok	Button	Used to accept the result.													
			Step 11: The system updates and saves the following in the database by using a FIRESQL INSERT query: AuditTrail Table AuditID DateTime TransactionDescription UserID												
Alternate Courses:	Alt Step 4: The Administrator clicks the "Cancel" button. Terminate Use Case.														
	Alt Step 5a: In the case of an empty Text box, the system notifies the Administrator of the Input error below the input field with the following: <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr></table>			Text/Label	Control Type	Additional Info									
Text/Label	Control Type	Additional Info													

	"Please fill this field."	Label (with a light red background color fill)	Used to inform the Administrator of the result.									
	"X"	Icon	Used to indicate an error result.									
	Redirect to Step 3.											
	Alt Step 5b: In the case of a Combo box option not selected, the system notifies the Administrator of the Input error below the input field with the following:											
	<table> <tr> <th>Text/Label</th> <th>Control Type</th> <th>Additional Info</th> </tr> <tr> <td>"Please select an option."</td> <td>Label (with a light red background color fill)</td> <td>Used to inform the Administrator of the result.</td> </tr> <tr> <td>"X"</td> <td>Icon</td> <td>Used to indicate an error result.</td> </tr> </table>			Text/Label	Control Type	Additional Info	"Please select an option."	Label (with a light red background color fill)	Used to inform the Administrator of the result.	"X"	Icon	Used to indicate an error result.
Text/Label	Control Type	Additional Info										
"Please select an option."	Label (with a light red background color fill)	Used to inform the Administrator of the result.										
"X"	Icon	Used to indicate an error result.										
Redirect to Step 3.												
Alt Step 6: In the case of invalid input being entered, the system notifies the Administrator of the Format error below the input field with the following:												
	<table> <tr> <th>Text/Label</th> <th>Control Type</th> <th>Additional Info</th> </tr> <tr> <td>"Invalid input."</td> <td>Label (with a light red background color fill)</td> <td>Used to inform the Administrator of the result.</td> </tr> <tr> <td>"X"</td> <td>Icon</td> <td>Used to indicate an error result.</td> </tr> </table>			Text/Label	Control Type	Additional Info	"Invalid input."	Label (with a light red background color fill)	Used to inform the Administrator of the result.	"X"	Icon	Used to indicate an error result.
Text/Label	Control Type	Additional Info										
"Invalid input."	Label (with a light red background color fill)	Used to inform the Administrator of the result.										
"X"	Icon	Used to indicate an error result.										
Redirect to Step 3.												
Alt Step 8: The Administrator declines to update the School by clicking the "No" button. Redirect to Step 4.												

	Alt Step 10: The School was not updated successfully in the database, the system will notify the Administrator of the update error with the following:		
	Text/Label	Control Type	Additional Info
	Failed	Dialog Box Title	Used to indicate what type of message is being displayed
	"Update unsuccessful. Please try again."	Dialog Box Message	Used to inform the Administrator of the result.
	"Help icon"	Button	Used to Load the Help document.
	Ok	Button	Used to accept the result.
	Redirect to Step 3.		
Conclusion:	The use case concludes when the Administrator has been notified that the School has been updated successfully.		
Post-Condition:	- An existing School is updated in the system with the relevant details. - The AuditTrail table is successfully updated. - The School table is successfully updated.		
Business Rules	Only an Administrator can update a School to the system.		
Implementation Constraints and Specifications:	None		
Assumptions:	None		

Open Issues:	• None

2.3.10.4. Delete School

System Name:	VVs
Author:	Tapiwarufaro Ndoro
Date:	15/06/2020
Version:	1.0

Use Case Name:	Delete School	Use Case Type:
Use Case ID:	9.4	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	Administrator	
Primary System Actor	None	
Other Participating Actors:	None	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process of an Administrator deleting an existing School's details from the system. The Use Case starts when the Administrator navigates by selecting the "School Management" option under the "System Management" Combo box in the Navigation bar. From there, they search for the School they want to delete, and the system displays the "School" screen with the School details to be deleted. The system populates the relevant Combo and Text boxes with information of the searched School, and the Administrator can make their deletion.	
Pre-Condition:	The Administrator must be logged into the system.	

Trigger:	The Administrator requests to delete an existing School from the system.				
Typical Course Of Events:	Actor Action	System Response			
		Manual Action	Automated Action		
	Step 1: The Administrator selects the "School Management" option under the "System Management" Combo box in the Navigation bar.		Step 2: The System invokes Use Case 9.2 "Search School" and displays the "School" screen with the data of the searched School.		
	Step 3: The Administrator clicks on the "Delete" button. [alt]		Step 4: The system prompts the Administrator to confirm the deletion of the School's details with the following:		
			Text/Label	Control Type	Additional Info
Confirmation			Dialog Box Title	Used to indicate what type of message is being displayed.	
"Are you sure you want to delete this School?"			Dialog Box Message	Used to inform the Administrator of the action.	
Yes	Button	Used to confirm the action.			
No	Button	Used to decline the action.			

	Step 5: The Administrator clicks the "Yes" button. [alt]		Step 6: The system then removes the School details in the School table. The following attributes are deleted in the database by using a FIRESQL DELETE query: School Table SchoolID SchoolName SchoolEmail SchoolTelephone												
			Step 7: The system notifies the Administrator that the School has been deleted successfully from the database with the following: [alt] <table><thead><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr></thead><tbody><tr><td>Success</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed.</td></tr><tr><td>"The School has been deleted successfully"</td><td>Dialog Box Message</td><td>Used to inform the Administrator of the result.</td></tr><tr><td>Ok</td><td>Button</td><td>Used to accept the result.</td></tr></tbody></table>	Text/Label	Control Type	Additional Info	Success	Dialog Box Title	Used to indicate what type of message is being displayed.	"The School has been deleted successfully"	Dialog Box Message	Used to inform the Administrator of the result.	Ok	Button	Used to accept the result.
	Text/Label	Control Type	Additional Info												
Success	Dialog Box Title	Used to indicate what type of message is being displayed.													
"The School has been deleted successfully"	Dialog Box Message	Used to inform the Administrator of the result.													
Ok	Button	Used to accept the result.													
		Step 8: The system updates and saves the following in the database by using a FIRESQL INSERT query:													

				AuditTrail Table AuditID DateTime TransactionDescription UserID														
Alternate Courses:	Alt Step 3: The Administrator clicks the "Cancel" button. Terminate Use Case.																	
	Alt Step 5: The Administrator declines to delete the School by clicking the "No" button. Redirect to Step 2.																	
	Alt Step 7: The Administrator is notified of the unsuccessful delete action with the following:																	
	<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Failed</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed</td></tr><tr><td>"Delete unsuccessful. Please try again."</td><td>Dialog Box Message</td><td>Used to inform the Administrator of the result.</td></tr><tr><td>"Help icon"</td><td>Button</td><td>Used to Load the Help document.</td></tr><tr><td>Ok</td><td>Button</td><td>Used to accept the result.</td></tr></table>			Text/Label	Control Type	Additional Info	Failed	Dialog Box Title	Used to indicate what type of message is being displayed	"Delete unsuccessful. Please try again."	Dialog Box Message	Used to inform the Administrator of the result.	"Help icon"	Button	Used to Load the Help document.	Ok	Button	Used to accept the result.
	Text/Label	Control Type	Additional Info															
	Failed	Dialog Box Title	Used to indicate what type of message is being displayed															
	"Delete unsuccessful. Please try again."	Dialog Box Message	Used to inform the Administrator of the result.															
"Help icon"	Button	Used to Load the Help document.																
Ok	Button	Used to accept the result.																
Redirect to Step 2.																		
Conclusion:	The use case concludes when the Administrator has been notified that the School has been successfully deleted.																	
Post-Condition:	• An existing School is deleted from the system with all the relevant details being removed.																	

	• The AuditTrail table is successfully updated. • The School table is successfully updated with the deleted details being removed.
Business Rules	• Only an Administrator can delete a School from the system.
Implementation Constraints and Specifications:	• None
Assumptions:	• None
Open Issues:	• None

2.3.11. Sub-system 10: Job

2.3.11.1. Report Repair Job

System Name:	VVs
Author:	Byron Donald
Date:	22/06/2020
Version:	1.0

Use Case Name:	Report Repair Job	Use Case Type:
Use Case ID:	10.1	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	Student	
Primary System Actor	None	
Other Participating Actors:	None	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process of a student reporting the faults that apply to the bicycle and the system then creates a new job entry. The use case starts when the student clicks the "Report Repair Job" button on the "Homepage" screen. From there, the system loads the "QR Scan" screen and prompts the student to scan their QR code after which the system loads the "Bicycle Blueprint" screen. The student will then click the section of the bicycle that the fault appears on followed by the system displaying all the parts that appear in the selected section on the "Section Blueprint" screen. This allows the student to select the specific part where the fault	

	is located so that the system can display the “Part Faults” screen where the student selects and confirms the fault that appears on their bicycle. The system will then load the “Report List” screen where the student can choose to either report more faults or submit the job.											
Pre-Condition:	The student reports the fault at the school.											
Trigger:	The student requests to report a new fault on the system.											
Typical Course Of Events:	Actor Action	System Response										
		Manual Action	Automated Action									
	Step 1: The Student from the “Homepage” screen clicks the “Report Fault” Button.		Step 2: The system Displays the “QR Scan” screen and populates it with the following details:									
			<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>“Please scan your QR Code to continue”</td><td>Heading</td><td>Used to inform the Student of the action that needs to be done.</td></tr><tr><td>QR-scan.gif</td><td>Image</td><td>Used to demonstrate what must be done</td></tr></table>	Text/Label	Control Type	Additional Info	“Please scan your QR Code to continue”	Heading	Used to inform the Student of the action that needs to be done.	QR-scan.gif	Image	Used to demonstrate what must be done
			Text/Label	Control Type	Additional Info							
“Please scan your QR Code to continue”	Heading	Used to inform the Student of the action that needs to be done.										
QR-scan.gif	Image	Used to demonstrate what must be done										
Step 3: The Student scans their QR Code with the required device.		Step 4: The repair job is assigned a unique RepairJobID in the RepairJob table which is set to the next consecutive number after the previous RepairJobID in the database, the system then creates and saves the new RepairJob in the RepairJob table. The following attributes are saved in the database by using a FIRESQL INSERT query:										

			RepairJob RepairJobID BicycleID StudentID RepairJobStatusID DateReported Assigned DateScheduled												
			Step 5: The system loads the "Bicycle Blueprint" screen and displays it with the following controls and layout: <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>"Please select the section of the bicycle where your fault appears"</td><td>Heading</td><td>Used to inform the user of what must be done on the screen.</td></tr><tr><td>Bicycle-blueprint</td><td>Picture Image</td><td>The blueprint of the bicycle that allows the student to select a section.</td></tr><tr><td>Unknown</td><td>Button</td><td>Used to report an unknown fault.</td></tr></table>	Text/Label	Control Type	Additional Info	"Please select the section of the bicycle where your fault appears"	Heading	Used to inform the user of what must be done on the screen.	Bicycle-blueprint	Picture Image	The blueprint of the bicycle that allows the student to select a section.	Unknown	Button	Used to report an unknown fault.
	Text/Label	Control Type	Additional Info												
	"Please select the section of the bicycle where your fault appears"	Heading	Used to inform the user of what must be done on the screen.												
	Bicycle-blueprint	Picture Image	The blueprint of the bicycle that allows the student to select a section.												
	Unknown	Button	Used to report an unknown fault.												

		The Bicycle Blueprint options are populated by using a FIRESQL READ query with data from the following: <table><tr><th>Entity</th><th>Attributes</th></tr><tr><td>Section</td><td>SectionName</td></tr></table>	Entity	Attributes	Section	SectionName															
Entity	Attributes																				
Section	SectionName																				
Step 6: The student clicks on the section of the bicycle where their problem is located. [alt]		Step 7: The system loads the "Section Blueprint" screen and displays it with the following controls and layout: <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>"Please select the part from the _____ where your fault appears"</td><td>Heading</td><td>Used to inform the user of what must be done on the screen.</td></tr><tr><td>Part Picture</td><td>Picture Image</td><td>Displays the image of all the relevant parts.</td></tr><tr><td>Part Name</td><td>Label</td><td>Displays the name of all the relevant parts.</td></tr><tr><td>Unknown</td><td>Button</td><td>Used to report an unknown fault.</td></tr></table> The Section Blueprint options are populated by using a FIRESQL READ query with data from the following: <table><tr><th>Entity</th><th>Attributes</th></tr><tr><td>Part</td><td>PartName</td></tr></table>	Text/Label	Control Type	Additional Info	"Please select the part from the _____ where your fault appears"	Heading	Used to inform the user of what must be done on the screen.	Part Picture	Picture Image	Displays the image of all the relevant parts.	Part Name	Label	Displays the name of all the relevant parts.	Unknown	Button	Used to report an unknown fault.	Entity	Attributes	Part	PartName
Text/Label	Control Type	Additional Info																			
"Please select the part from the _____ where your fault appears"	Heading	Used to inform the user of what must be done on the screen.																			
Part Picture	Picture Image	Displays the image of all the relevant parts.																			
Part Name	Label	Displays the name of all the relevant parts.																			
Unknown	Button	Used to report an unknown fault.																			
Entity	Attributes																				
Part	PartName																				

	Step 8: The student clicks on the Part of the bicycle where their problem is located. [alt]		Step 9: The system loads the “Part Faults” screen and displays it with the following controls and layout:															
			<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>“Please select the fault that appears on the _____”</td><td>Heading</td><td>Used to inform the user of what must be done on the screen.</td></tr><tr><td>Fault Picture</td><td>Picture Image</td><td>Displays the image of all the relevant Faults.</td></tr><tr><td>Fault Name</td><td>Label</td><td>Displays the name of all the relevant Faults.</td></tr><tr><td>Unknown</td><td>Button</td><td>Used to report an unknown fault.</td></tr></table>	Text/Label	Control Type	Additional Info	“Please select the fault that appears on the _____”	Heading	Used to inform the user of what must be done on the screen.	Fault Picture	Picture Image	Displays the image of all the relevant Faults.	Fault Name	Label	Displays the name of all the relevant Faults.	Unknown	Button	Used to report an unknown fault.
	Text/Label	Control Type	Additional Info															
	“Please select the fault that appears on the _____”	Heading	Used to inform the user of what must be done on the screen.															
	Fault Picture	Picture Image	Displays the image of all the relevant Faults.															
	Fault Name	Label	Displays the name of all the relevant Faults.															
	Unknown	Button	Used to report an unknown fault.															
			The Parts Faults options are populated by using a FIRESQL READ query with data from the following:															
			<table><tr><th>Entity</th><th>Attributes</th></tr><tr><td>FaultType</td><td>FaultType</td></tr></table>	Entity	Attributes	FaultType	FaultType											
	Entity	Attributes																
FaultType	FaultType																	
	Step 10: The student clicks on the fault that they wish to report. [alt]		Step 11: The system prompts the Student to confirm their selection with the following:															
			<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Confirmation</td><td>Dialog Box Title</td><td>Used to indicate what type of</td></tr></table>	Text/Label	Control Type	Additional Info	Confirmation	Dialog Box Title	Used to indicate what type of									
Text/Label	Control Type	Additional Info																
Confirmation	Dialog Box Title	Used to indicate what type of																

			message is being displayed.		
			"Are you sure you want to select a ____?"	Dialog Box Message	Used to inform the Student of the action.
			Yes	Button	Used to confirm the action.
			No	Button	Used to decline the action.
Step 12: The Student clicks the "Yes" button. [alt]			Step 13: The JobFault is assigned a unique JobFaultID in the JobFault table which is set to the next consecutive number after the previous JobFaultID in the database, the system then creates and saves the new JobFault in the JobFault table. The following attributes are saved in the database by using a FIRESQL INSERT query: JobFault JobFaultID PartFaultID SectionPartID RepairJobID JobFaultStatusID SectionID Notes		

			Step 14: The system notifies the Student that the Fault has been added successfully to the database with the following: [alt]			
			Text/Label	Control Type	Additional Info	
			Successful	Dialog Box Title	Used to indicate what type of message is being displayed.	
			"The Fault was added successfully"	Dialog Box Message	Used to inform the Student of the result.	
	Okay	Button	Used to accept the result.			
				Step 15: The system loads the "Report List" screen and displays it with the following controls and layout:		
				Text/Label	Control Type	Additional Info
				Add a new Fault	Icon Link	Used to load the "Bicycle Blueprint" screen.
				Report Job	Button	Used to finalize the job entry.
				Job Faults	Table	Displays a list of Faults for the job.
Delete Fault	Icon Link	Used to remove a fault from the table.				

			The table is populated with data from the following by using a FIRESQL READ query: <table><tr><th>Entity</th><th>Attributes</th><th>Input Name</th></tr><tr><td>FaultType</td><td>FaultType</td><td>Fault</td></tr><tr><td>JobFault</td><td>Notes</td><td>Fault</td></tr><tr><td>Section</td><td>SectionName</td><td>Section</td></tr><tr><td>Part</td><td>PartName</td><td>Part</td></tr></table>	Entity	Attributes	Input Name	FaultType	FaultType	Fault	JobFault	Notes	Fault	Section	SectionName	Section	Part	PartName	Part
	Entity	Attributes	Input Name															
	FaultType	FaultType	Fault															
	JobFault	Notes	Fault															
	Section	SectionName	Section															
	Part	PartName	Part															
	Step 16: The Student clicks the "Report Job" button. [alt]		Step 17: The system assigns the job to a mechanic and creates a new Check-In entry Check-In Table Check-InID CheckInDate Check-InStatusID JobTypeID MechanicSchoolID The system notifies the Student that the Faults have been added successfully to the database with the following: [alt] <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Successful</td><td>Dialog Box Title</td><td>Used to indicate what type of</td></tr></table>	Text/Label	Control Type	Additional Info	Successful	Dialog Box Title	Used to indicate what type of									
	Text/Label	Control Type	Additional Info															
	Successful	Dialog Box Title	Used to indicate what type of															

			message is being displayed.		
			"The Faults Have been Reported successfully"	Dialog Box Message	Used to inform the Student of the result.
			Okay	Button	Used to accept the result.
			Step 18: The system creates and saves the following in the database by using a FIRESQL INSERT query: AuditTrail Table AuditID DateTime TransactionDescription UserID		
Alternate Courses:	Alt Step 6a: The student is unaware of the section where the problem is located. The Student clicks the "Unknown" Button.				
	Alt Step 6b: The System loads the "UnknownFault" modal with the following:				

	Text/Label	Control Type	Additional Info
	Unknown Fault	Modal Title	Used to indicate what modal is being displayed
	"Please enter any additional information."	Heading	Used to inform the Student of what can be done.
	unknownDetails	Text Field	Used to provide the student with a means to input additional information
	Submit	Button	Used to submit the JobFault with unknown attributes.
	Proceed to Step 13.		
	Alt Step 8a: The student is unaware of the part where the problem is located. The Student clicks the "Unknown" Button.		
	Alt Step 8b: The System loads the "UnknownFault" modal with the following:		

	Text/Label	Control Type	Additional Info
	Unknown Fault	Modal Title	Used to indicate what modal is being displayed
	"Please enter any additional information."	Heading	Used to inform the Student of what can be done.
	unknownDetails	Text Field	Used to provide the student with a means to input additional information
	Submit	Button	Used to submit the JobFault with unknown attributes.
	Proceed to Step 13.		
	Alt Step 10a: The student is unaware of the fault for the specified part. The Student clicks the "Unknown" Button.		
	Alt Step 10b: The System loads the "UnknownFault" modal with the following:		

	Text/Label	Control Type	Additional Info
	Unknown Fault	Modal Title	Used to indicate what modal is being displayed
	"Please enter any additional information."	Heading	Used to inform the Student of what can be done.
	unknownDetails	Text Field	Used to provide the student with a means to input additional information
	Submit	Button	Used to submit the JobFault with unknown attributes.
Proceed to Step 13.			
Alt Step 12: The Student clicks the "No" button. Redirect to Step 5.			
Alt Step 14: The Fault was not added successfully to the database, the system will notify the Student of the save error with the following:			
	Text/Label	Control Type	Additional Info
	Failed	Dialog Box Title	Used to indicate what type of message is being displayed
	"Fault reported unsuccessful. Please try again."	Dialog Box Message	Used to inform the Student of the result.
	Okay	Button	Used to accept the result.

	Redirect to Step 5.												
	Alt Step 16: The Student has more to report and clicks the "Add a new Fault" Button. Redirect to Step 5.												
	Alt Step 17: The Fault was not added successfully to the database, the system will notify the Student of the save error with the following:												
	<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Failed</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed</td></tr><tr><td>"Job reported unsuccessful. Please try again."</td><td>Dialog Box Message</td><td>Used to inform the Student of the result.</td></tr><tr><td>Okay</td><td>Button</td><td>Used to accept the result.</td></tr></table>	Text/Label	Control Type	Additional Info	Failed	Dialog Box Title	Used to indicate what type of message is being displayed	"Job reported unsuccessful. Please try again."	Dialog Box Message	Used to inform the Student of the result.	Okay	Button	Used to accept the result.
	Text/Label	Control Type	Additional Info										
Failed	Dialog Box Title	Used to indicate what type of message is being displayed											
"Job reported unsuccessful. Please try again."	Dialog Box Message	Used to inform the Student of the result.											
Okay	Button	Used to accept the result.											
Redirect to Step 15.													
Conclusion:	The use case concludes when the student has been notified that the Job and faults have been reported successfully.												
Post-Condition:	• A new Job is added to the system with the relevant details. • The AuditTrail table is successfully updated. • The ReportFault and ReportLine tables are successfully updated.												
Business Rules	• Only registered students can report a fault. • Only registered bikes can be reported.												
Implementation Constraints and Specifications:	• Students can only report at school because they do not have personal devices												

Assumptions:	• The student knows how to interact with the GUI. • The system will retrieve the entries only picked up by the Antenna at that school on that day, at the back end.
Open Issues:	• None

2.3.11.2. Complete Job

System Name:	VVs
Author:	Byron Donald
Date:	22/06/2020
Version:	1.0

Use Case Name:	Complete Job	Use Case Type:
Use Case ID:	10.2	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	Mechanic	
Primary System Actor	None	
Other Participating Actors:	None	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process of a Mechanic completing a job on the system. Once all the Faults have been fixed the mechanic then clicks the "Complete Job" button.	
Pre-Condition:	The Mechanic must be viewing the Job List. The Mechanic must be logged into the system.	
Trigger:	The Mechanic requests to complete a Fault on the System.	

Typical Course Of Events:	Actor Action	System Response														
		Manual Action	Automated Action													
	Step 1: The Mechanic from the "Job List" screen clicks the "Complete Job" button on the "Fault List" Screen.		Step 2: The system then updates and saves the updated Repair Job details in the Repair Job table. The following attributes are updated in the database by using a SQL UPDATE query: RepairJob Table RepairJobStatusID													
			Step 3: The system notifies the Mechanic that the Job has been completed in the database with the following: [alt] <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Successful</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed.</td></tr><tr><td>"The Job was completed successfully"</td><td>Dialog Box Message</td><td>Used to inform the Mechanic of the result.</td></tr><tr><td>Okay</td><td>Button</td><td>Used to accept the result.</td></tr></table>		Text/Label	Control Type	Additional Info	Successful	Dialog Box Title	Used to indicate what type of message is being displayed.	"The Job was completed successfully"	Dialog Box Message	Used to inform the Mechanic of the result.	Okay	Button	Used to accept the result.
	Text/Label	Control Type	Additional Info													
	Successful	Dialog Box Title	Used to indicate what type of message is being displayed.													
"The Job was completed successfully"	Dialog Box Message	Used to inform the Mechanic of the result.														
Okay	Button	Used to accept the result.														
		Step 4: The system updates and saves the following in the database by using a FIRESQL INSERT query: AuditTrail Table AuditID														

			DateTime TransactionDescription UserID
Alternate Courses:	Alt Step 1: The Mechanic from the "Maintenance List" screen clicks the "Complete Job" button on the "Maintenance List" Screen.		
	Alt Step 2: The system then updates and saves the updated Repair Job details in the Repair Job table. The following attributes are updated in the database by using a SQL UPDATE query: MaintenanceJob Table MaintenanceJobStatusID		
Conclusion:	The use case concludes when the Mechanic has been notified that the Job has been completed successfully.		
Post-Condition:	- The Job is completed on the system with the relevant details. - The AuditTrail table is successfully updated.		
Business Rules	Only registered users can complete a job		
Implementation Constraints and Specifications:	None		
Assumptions:	- The Mechanic knows how to interact with the GUI. - The complete job button can only be clicked when all the faults have been completed/fixed.		
Open Issues:			

2.3.11.3. Check-in Bicycle

System Name:	VVs
Author:	Byron Donald
Date:	22/06/2020
Version:	1.0

Use Case Name:	Check-In Bicycle	Use Case Type:
Use Case ID:	10.3	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	Student	
Primary System Actor	Mechanic	
Other Participating Actors:	None	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process where a student arrives with their bicycle to be checked-in by the mechanic for the current job. The Use Case starts when the student arrives with their bicycle and hands it over to the mechanic who then clicks the "Check-In Bicycle" button on the "Jobs" screen. The system then prompts the mechanic to scan the bicycles QR Code through the "QR Scan" Screen after which the system then searches for any relevant jobs and verifies if they are eligible for Check-In. After which the system will mark the relevant bicycle as arrived for a job on the system.	
Pre-Condition:	The student has a bicycle that is registered to the student with them physically.	

	The mechanic is logged into the system and has the "Jobs" screen open.											
Trigger:	The Student arrives with their Bicycle and requests to check it in.											
Typical Course Of Events:	Actor Action	System Response										
		Manual Action	Automated Action									
	Step 1: The Student arrives with their bicycle and hands it over to the mechanic.	Step 2: The Mechanic from the "Jobs" screen clicks the "Check-In Bicycle" button.	Step 3: The system Displays the "QR Scan" screen and populates it with the following details:									
			<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>"Please scan your QR Code to continue"</td><td>Heading</td><td>Used to inform the Student of the action that needs to be done.</td></tr><tr><td>QR-scan.gif</td><td>Image</td><td>Used to demonstrate what must be done</td></tr></table>	Text/Label	Control Type	Additional Info	"Please scan your QR Code to continue"	Heading	Used to inform the Student of the action that needs to be done.	QR-scan.gif	Image	Used to demonstrate what must be done
			Text/Label	Control Type	Additional Info							
"Please scan your QR Code to continue"	Heading	Used to inform the Student of the action that needs to be done.										
QR-scan.gif	Image	Used to demonstrate what must be done										
	Step 4: The Mechanic scans the QR Code with the required device. [alt]	Step 5: The system captures and READS the following details:										
		Step 6: The system READS the following details:										

			<table><tr><td>RepairJob</td><td>Check-InID</td></tr><tr><td>RepairJob</td><td>BicycleID</td></tr><tr><td>MaintenanceJob</td><td>Check-InID</td></tr><tr><td>MaintenanceJob</td><td>BicycleID</td></tr><tr><td>Check-In</td><td>Check-InStatusID</td></tr><tr><td>Check-In</td><td>CheckInDate</td></tr></table> Where the BicycleID is equal to the BicycleID from STEP 5 and the DateScheduled is equal to the Current Date. [alt]	RepairJob	Check-InID	RepairJob	BicycleID	MaintenanceJob	Check-InID	MaintenanceJob	BicycleID	Check-In	Check-InStatusID	Check-In	CheckInDate
	RepairJob	Check-InID													
	RepairJob	BicycleID													
	MaintenanceJob	Check-InID													
	MaintenanceJob	BicycleID													
	Check-In	Check-InStatusID													
	Check-In	CheckInDate													
			Step 7: The system then updates and saves the updated Check-In details in the Check-In table to reflect that the bicycle has been successfully Checked In. The following attributes are updated in the database by using a FIRESQL UPDATE query: <table><tr><td>Check-In Table</td></tr><tr><td>CheckInDate</td></tr><tr><td>Check-InStatusID</td></tr></table>	Check-In Table	CheckInDate	Check-InStatusID									
	Check-In Table														
	CheckInDate														
Check-InStatusID															
		Step 8: The system notifies the Mechanic that the Bicycle has been Checked-in Successfully in the database with the following: <table><tr><td>Text/Label</td><td>Control Type</td><td>Additional Info</td></tr></table>	Text/Label	Control Type	Additional Info										
Text/Label	Control Type	Additional Info													

			Success	Dialog Box Title	Used to indicate what type of message is being displayed.
			"The Bicycle has been Checked-in successfully"	Dialog Box Message	Used to inform the Mechanic of the result.
			Ok	Button	Used to accept the result.
			Step 9: The Mechanic notifies the student that their bicycle has been checked in.	Step 10: The system creates and saves the following in the database by using a FIRESQL INSERT query: AuditTrail Table AuditID DateTime TransactionDescription UserID	
Alternate Courses:	Alt Step 4: The scan was invalid; the system will notify the Mechanic of the error with the following:				
	Text/Label	Control Type	Additional Info		
	Error	Dialog Box Title	Used to indicate what type of message is being displayed		

	"Scan was invalid. Please try again."	Dialog Box Message	Used to inform the Student of the result.
	Okay	Button	Used to accept the result.
	Redirect to Step 3.		
	Alt Step 6a: The Bicycle is not Due Today; the system will notify the Mechanic of the error with the following:		
	Text/Label	Control Type	Additional Info
	Failed	Dialog Box Title	Used to indicate what type of message is being displayed
	"Bicycle is not due today."	Dialog Box Message	Used to inform the Mechanic of the result.
	Create Job	Button	Used to Invoke Use Case 10.1 Report Repair Job.
	Okay	Button	Used to accept the result.
	The Student does not wish to create a new Repair Job. Terminate Use Case.		
	Alt Step 6b: The Bicycle is not Due Today; the system will notify the Mechanic of the error with the following:		
	Text/Label	Control Type	Additional Info
	Failed	Dialog Box Title	Used to indicate what type of

	message is being displayed	
	"Bicycle is not due today."	Dialog Box Message
	Create Job	Button
	Cancel	Button
Used to inform the Mechanic of the result.		
Used to Invoke Use Case 10.1 Report Repair Job.		
Used to cancel the check-in process.		
The Student wishes to create a new Repair Job.		
Mechanic clicks the "Create Job" button. Invoking Use Case 10.1 Report Repair Job. Continue from Step 6		
Conclusion:	The mechanic has the bicycle and has updated the relevant job to Received.	
Post-Condition:	- The Bicycle is ready to be repaired. - The AuditTrail table is successfully updated.	
Business Rules	The bicycle must be registered to a valid student and the details much match.	
Implementation Constraints and Specifications:	Students must physically check-in their bicycles.	
Assumptions:	Students check-in their bicycles at school on a specific date.	
Open Issues:	None	

2.3.11.4. *Schedule Maintenance*

System Name:	VVs
Author:	Byron Donald
Date:	22/06/2020
Version:	1.0

Use Case Name:	Schedule Maintenance	Use Case Type:
Use Case ID:	10.4	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	Mechanic	
Primary System Actor	None	
Other Participating Actors:	None	
Other Interested Stakeholders:	Administrator	
Description:	This Technical Use Case Narrative describes the process of the system scheduling the next type of maintenance as soon as the mechanic marks the current maintenance job as complete. The Use Case starts when the mechanic finishes Use Case Narrative 10.2 for a maintenance job.	
Pre-Condition:	Maintenance Types are set to have their fixed frequencies. The mechanic is logged into the system.	

Trigger:	The mechanic completes a maintenance Job												
Typical Course Of Events:	Actor Action	System Response											
		Manual Action	Automated Action										
	Step 1: The Mechanic marks the current Maintenance Job as complete.		Step 2: The System uses a FIRESQL READ query to read the following:										
			<table><tr><th>Entity</th><th>Attributes</th></tr><tr><td>MaintenanceList</td><td>MaintenanceJobID</td></tr><tr><td>MaintenanceList</td><td>MaintenanceListStatusID</td></tr><tr><td>MaintenanceList</td><td>MaintenanceListID</td></tr><tr><td>MaintenanceJob</td><td>MaintenanceJobID</td></tr></table>	Entity	Attributes	MaintenanceList	MaintenanceJobID	MaintenanceList	MaintenanceListStatusID	MaintenanceList	MaintenanceListID	MaintenanceJob	MaintenanceJobID
			Entity	Attributes									
			MaintenanceList	MaintenanceJobID									
MaintenanceList			MaintenanceListStatusID										
MaintenanceList	MaintenanceListID												
MaintenanceJob	MaintenanceJobID												
Where the MaintenanceJobID is equal to the MaintenanceJobID from the Completed Job and the MaintenanceListStatusID is set to complete.													
		Step 3: The System uses a FIRESQL READ query to read the following:											
		<table><tr><th>Entity</th><th>Attributes</th></tr><tr><td>MaintenanceType</td><td>MaintenanceTypeID</td></tr><tr><td>MaintenanceType</td><td>Frequency</td></tr></table>	Entity	Attributes	MaintenanceType	MaintenanceTypeID	MaintenanceType	Frequency					
		Entity	Attributes										
MaintenanceType	MaintenanceTypeID												
MaintenanceType	Frequency												

			Where the MaintenanceTypeID is equal to the MaintenanceTypeID from the retrieved MaintenanceList Entry.				
			Step 4: The system uses the retrieved Frequency to calculate the next Maintenance date for that Maintenance Type based on the current Date and stores it in a temporary variable.				
			Step 5: The System uses a FIRESQL READ query to read the following:				
			<table><tr><th>Entity</th><th>Attributes</th></tr><tr><td>MaintenanceJob</td><td>MaintenanceJobID</td></tr></table>	Entity	Attributes	MaintenanceJob	MaintenanceJobID
			Entity	Attributes			
MaintenanceJob			MaintenanceJobID				
<table><tr><td>MaintenanceJob</td><td>BicycleID</td></tr></table>	MaintenanceJob	BicycleID					
MaintenanceJob	BicycleID						
<table><tr><td>MaintenanceJob</td><td>DateScheduled</td></tr></table>	MaintenanceJob	DateScheduled					
MaintenanceJob	DateScheduled						
			Where the DateScheduled is equal to the Date calculated in STEP 4 and the BicycleID is equal to the BicycleID from the Completed Job. [alt]				
			Step 6: The system then creates a new MaintenanceList Entry in the MaintenanceList table for the retrieved MaintenanceJob Entry in STEP 5. The maintenance list is assigned a unique MaintenanceListID in the MaintenanceList table which is set to the next consecutive number after the previous MaintenanceListID in the database, the system then creates and saves the new ScheduledMaintenance in the MaintenanceJob table. The following attributes with values				

			retrieved from previous steps are saved in the database by using a FIRESQL INSERT query: <table><tr><th>MaintenanceList</th></tr><tr><td>MaintenanceListID</td></tr><tr><td>MaintenanceJobID</td></tr><tr><td>MaintenanceListStatusID</td></tr><tr><td>D</td></tr><tr><td>MaintenanceTypeID</td></tr></table>	MaintenanceList	MaintenanceListID	MaintenanceJobID	MaintenanceListStatusID	D	MaintenanceTypeID		
	MaintenanceList										
	MaintenanceListID										
MaintenanceJobID											
MaintenanceListStatusID											
D											
MaintenanceTypeID											
			Step 7: The system then updates and saves the updated MaintenanceList details in the MaintenanceList table for the MaintenanceList entry retrieved in STEP 2. The following attribute is updated to NextTypeScheduled in the database by using a FIRESQL UPDATE query: <table><tr><th>MaintenanceList Table</th></tr><tr><td>MaintenanceListStatusID</td></tr></table>	MaintenanceList Table	MaintenanceListStatusID						
MaintenanceList Table											
MaintenanceListStatusID											
			Step 8: The System uses a FIRESQL READ query to read the following: <table><tr><th>Entity</th><th>Attributes</th></tr><tr><td>MaintenanceList</td><td>MaintenanceJobID</td></tr><tr><td>MaintenanceList</td><td>MaintenanceListStatusID</td></tr><tr><td>MaintenanceList</td><td>MaintenanceListID</td></tr></table>	Entity	Attributes	MaintenanceList	MaintenanceJobID	MaintenanceList	MaintenanceListStatusID	MaintenanceList	MaintenanceListID
Entity	Attributes										
MaintenanceList	MaintenanceJobID										
MaintenanceList	MaintenanceListStatusID										
MaintenanceList	MaintenanceListID										

			MaintenanceJob MaintenanceJobID Where the MaintenanceJobID is equal to the MaintenanceJobID from the Completed Job and the MaintenanceListStatusID is set to complete to ensure that no additional entries are found. [alt]
			Step 9: The system assigns the job to a mechanic, and creates and saves the following in the database by using a FIRESQL INSERT query: Check-In Table Check-InID CheckInDate Check-InStatusID JobTypeID MechanicSchoolID
			Step 10: The system creates and saves the following in the database by using a FIRESQL INSERT query: AuditTrail Table AuditID DateTime TransactionDescription UserID
Alternate Courses:	Alt Step 5: No maintenance job entries are found. The system then creates a new MaintenanceJob Entry.		

	The MaintenanceJob is assigned a unique MaintenanceJobID in the MaintenanceJob table which is set to the next consecutive number after the previous MaintenanceJobID in the database, the system then creates and saves the new ScheduledMaintenance in the MaintenanceJob table. The following attributes with values retrieved from previous steps are saved in the database by using a FIRESQL INSERT query: MaintenanceJob MaintenanceJobID BicycleID MaintenanceJobStatusID DateScheduled Assigned Proceeds to Step 6.
	Alt Step 8: The system retrieves an additional entry from the FIRESQL READ. Repeat process starting from Step 3.
Conclusion:	The next maintenance job is created and stored successfully.
Post-Condition:	- The Maintenance Job is created and assigned. - The AuditTrail table is successfully updated. - The MaintenanceJob table is successfully updated.
Business Rules	Maintenance must be scheduled based on a frequency
Implementation Constraints and Specifications:	None

Assumptions:	• The first set of Maintenance Jobs are scheduled when the bicycle is assigned to a student
Open Issues:	• None

2.3.12. Sub-system 11: Reporting

2.3.12.1. *Generate Mechanic Schedule*

System Name:	VVs
Author:	Tapiwarufaro Ndoro
Date:	15/06/2020
Version:	1.0

Use Case Name:	Generate Mechanic Schedule	Use Case Type:
Use Case ID:	11.1	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	Time (Period)	
Primary System Actor	None	
Other Participating Actors:	Mechanic (ERA)	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process of a Mechanic Schedule being generated by the system after a certain specified period. This Mechanic Schedule is then sent to the Mechanic and can be viewed by the Administrator too. Both users can download the Mechanic Schedule as a PDF. This use case details the controls displayed and populated by the system.	
Pre-Condition:	There must be pending activities for a particular mechanic to carry out.	

Trigger:	After a certain period, passes, the system is prompted to generate a Mechanic schedule.													
Typical Course Of Events:	Actor Action	System Response												
		Manual Action	Automated Action											
	Step 1: After a certain period in time passes, the system is prompted to generate a Mechanic schedule.		Step 2: The system uses a FIRESQL READ query to read the following: <table><tr><th>Entity</th><th>Attributes</th></tr><tr><td>BusinessRule</td><td>Period</td></tr></table> This data is used to prompt the generation of the “Mechanic Schedule for that duration (e.g., 1 week)”	Entity	Attributes	BusinessRule	Period							
	Entity	Attributes												
BusinessRule	Period													
		Step 3: The system generates the “Mechanic Schedule” with all the following for the specific Mechanic: <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Navigation</td><td>Navigation-bar</td><td>Used for Navigation of the site.</td></tr><tr><td>Start Date</td><td>Label</td><td>Used for indicating the start point of this Mechanic schedule period</td></tr><tr><td>End Date</td><td>Label</td><td>Used for indicating the endpoint of this Mechanic schedule period</td></tr></table>	Text/Label	Control Type	Additional Info	Navigation	Navigation-bar	Used for Navigation of the site.	Start Date	Label	Used for indicating the start point of this Mechanic schedule period	End Date	Label	Used for indicating the endpoint of this Mechanic schedule period
Text/Label	Control Type	Additional Info												
Navigation	Navigation-bar	Used for Navigation of the site.												
Start Date	Label	Used for indicating the start point of this Mechanic schedule period												
End Date	Label	Used for indicating the endpoint of this Mechanic schedule period												

			Mechanic Schedule	Table	Displays a list of the Mechanic jobs pending.
			Download Report	Button	Used to download a PDF of the report.
			The table is populated by using a FIRESQL READ query with data from the following:		
			Entity	Attributes	Input Name
			Mechanic Schedule	MechanicScheduleID	Job Code
			BicycleStudent	BicycleID	Bicycle Code
			Student	StudentName & StudentSurname	Student Name
			MaintenanceTask & FaultTask	MaintenanceTask Description or FaultTaskDescription	Job Description
			Student	PhoneNumber	Student Contact
			MechanicSchedule	DateScheduled	Date Scheduled
			Step 4: The system sends the Mechanic Schedule report via email to the mechanic. A FIRESQL READ query used on the following to retrieve contact details:		
			Entity	Attributes	

			User	Email												
	Step 5: User (Mechanic or Administrator) clicks on the "Download Report" button. [alt]		Step 6: The system generates a PDF download file and displays the following: [alt]													
			<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Success</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed.</td></tr><tr><td>"Mechanic Schedule Report has been downloaded successfully."</td><td>Dialog Box Message</td><td>Used to inform the User of the result.</td></tr><tr><td>Ok</td><td>Button</td><td>Used to accept the result.</td></tr></table>		Text/Label	Control Type	Additional Info	Success	Dialog Box Title	Used to indicate what type of message is being displayed.	"Mechanic Schedule Report has been downloaded successfully."	Dialog Box Message	Used to inform the User of the result.	Ok	Button	Used to accept the result.
			Text/Label	Control Type	Additional Info											
			Success	Dialog Box Title	Used to indicate what type of message is being displayed.											
"Mechanic Schedule Report has been downloaded successfully."	Dialog Box Message	Used to inform the User of the result.														
Ok	Button	Used to accept the result.														
		Step 6: The system creates and saves the following in the database by using a FIRESQL INSERT query:														
		<table><tr><th>AuditTrail Table</th></tr><tr><td>AuditID</td></tr><tr><td>DateTime</td></tr><tr><td>TransactionDescription</td></tr><tr><td>UserID</td></tr></table>		AuditTrail Table	AuditID	DateTime	TransactionDescription	UserID								
AuditTrail Table																
AuditID																
DateTime																
TransactionDescription																
UserID																
Alternate Courses:	Alt Step 5: The user closes the Mechanic Schedule. Terminate Use Case.															
	Alt Step 6: There was an error in generating a download file for the User. Display the following:															

	Text/Label	Control Type	Additional Info
	Error	Dialog Box Title	Used to indicate what type of message is being displayed.
	"Download failed. Please try again."	Dialog Box Message	Used to inform the User of the result.
	"Help icon"	Button	Used to Load the Help document.
	Ok	Button	Used to accept the result.
	Redirect to Step 5.		
Conclusion:	The use case concludes when the generated Mechanic Schedule is successfully emailed or downloaded.		
Post-Condition:	- The generated Mechanic Schedule remains fixed until the next set is generated. - The AuditTrail table is updated.		
Business Rules	Only time can trigger the generation of this Mechanic Schedule. A request to view the Mechanic Schedule will read this generated Mechanic Schedule.		
Implementation Constraints and Specifications:	None		
Assumptions:	The Mechanic will be able to view the Mechanic Schedule, the same as the Administrator.		
Open Issues:	None		

2.3.12.2. Generate Mechanic Schedule Requirements Report

System Name:	VVs
Author:	Tapiwarufaro Ndoro
Date:	15/06/2020
Version:	1.0

Use Case Name:	Generate Mechanic Schedule Requirements Report	Use Case Type:
Use Case ID:	11.2	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	Time (Period)	
Primary System Actor	None	
Other Participating Actors:	Mechanic (ERA)	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process of a Mechanic Schedule Requirements Report being generated by the system after a certain specified period. This Mechanic Schedule Requirements is then sent to the Mechanic and can be viewed by the Administrator too. Both users can download the Mechanic Schedule Requirements Report linked to a Mechanic Schedule as a PDF. This use case details the controls displayed and populated by the system.	
Pre-Condition:	There must be pending Mechanic Schedule activities for a particular Mechanic to carry out before generating any requirements report	

	• A Mechanic Schedule must have been generated.										
Trigger:	After a certain period, passes, the system is prompted to generate a Mechanic Schedule Requirements Report linked to a Mechanic Schedule.										
Typical Course Of Events:	Actor Action	System Response									
		Manual Action	Automated Action								
	Step 1: After a certain period in time passes, the system is prompted to generate a Mechanic Schedule Requirements Report.		Step 2: The system uses a FIRESQL READ query to read the following: <table><tr><td>Entity</td><td>Attributes</td></tr><tr><td>BusinessRule</td><td>Period</td></tr></table> This data is used to prompt the generation of the “Mechanic Schedule Requirements Report for that duration (e.g., 1 week)”	Entity	Attributes	BusinessRule	Period				
	Entity	Attributes									
BusinessRule	Period										
		Step 3: The system generates the “Mechanic Schedule Requirements Report” for the specific mechanic and displays all the following: <table><tr><td>Text/Label</td><td>Control Type</td><td>Additional Info</td></tr><tr><td>Navigation</td><td>Navigation-bar</td><td>Used for Navigation of the site.</td></tr><tr><td>Start Date</td><td>Label</td><td>Used for indicating the start point of this Mechanic Schedule Requirements Report period</td></tr></table>	Text/Label	Control Type	Additional Info	Navigation	Navigation-bar	Used for Navigation of the site.	Start Date	Label	Used for indicating the start point of this Mechanic Schedule Requirements Report period
Text/Label	Control Type	Additional Info									
Navigation	Navigation-bar	Used for Navigation of the site.									
Start Date	Label	Used for indicating the start point of this Mechanic Schedule Requirements Report period									

			End Date	Label	Used for indicating the endpoint of this Mechanic Schedule Requirements Report period
			Mechanic Schedule Requirements Report	Table	Displays a list of the Maintenance jobs pending.
			Download Report	Button	Used to download a PDF of the report.
			The table is populated by using a FIRESQL READ query with data from the following:		
			Entity	Attributes	Input Name
			MechanicSchedule	MechanicScheduleID	Job Code
			BicycleStudent	BicycleID	Bicycle Code
			Student	StudentName & StudentSurname	Student Name
			MaintenanceTask & FaultTask	MaintenanceTask Description or FaultTaskDescription	Job Description
			FaultType	PartNeeded	Parts Needed

			MechanicSchedule DateScheduled Date Scheduled e A sub-table is generated at run-time to show the following: <table><tr><th>Output</th><th>Attribute</th><th>Entity</th></tr><tr><td>Parts Needed</td><td>PartNeeded (Groups together the Parts Needed in the main table displayed)</td><td>FaultType</td></tr><tr><td>Quantity</td><td>(Count the number of occurrences)</td><td>None</td></tr></table>	Output	Attribute	Entity	Parts Needed	PartNeeded (Groups together the Parts Needed in the main table displayed)	FaultType	Quantity	(Count the number of occurrences)	None
	Output	Attribute	Entity									
	Parts Needed	PartNeeded (Groups together the Parts Needed in the main table displayed)	FaultType									
Quantity	(Count the number of occurrences)	None										
		Step 4: The system sends the Mechanic Schedule Requirements Report via email to the mechanic. A FIRESQL READ query used on the following to retrieve contact details: <table><tr><th>Entity</th><th>Attributes</th></tr><tr><td>User</td><td>Email</td></tr></table>	Entity	Attributes	User	Email						
Entity	Attributes											
User	Email											
Step 5: User (Mechanic or Administrator) clicks on the "Download Report" button. [alt]		Step 6: The system generates a PDF download file and displays the following: [alt] <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Success</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed.</td></tr></table>	Text/Label	Control Type	Additional Info	Success	Dialog Box Title	Used to indicate what type of message is being displayed.				
Text/Label	Control Type	Additional Info										
Success	Dialog Box Title	Used to indicate what type of message is being displayed.										

			"Mechanic Schedule Requirements Report has been downloaded successfully." Dialog Box Message Used to inform the User of the result.
			Ok Button Used to accept the result.
			Step 7: The system creates and saves the following in the database by using a FIRESQL INSERT query: AuditTrail Table AuditID DateTime TransactionDescription UserID
Alternate Courses:	Alt Step 5: The user closes the Mechanic Schedule Requirements Report. Terminate Use Case.		
	Alt Step 6: There was an error in generating a download file for the User. Display the following:		
	Text/Label Error	Control Type Dialog Box Title	Additional Info Used to indicate what type of message is being displayed.
	"Download failed. Please try again."	Dialog Box Message	Used to inform the User of the result.

	"Help icon"	Button	Used to Load the Help document.
	Ok	Button	Used to accept the result.
	Redirect to Step 5.		
Conclusion:	The use case concludes when the generated Mechanic Schedule Requirements Report is successfully emailed or downloaded.		
Post-Condition:	- The generated Mechanic Schedule Requirements Report remains fixed until the next set is generated. - The AuditTrail table is updated.		
Business Rules	Only time can trigger the generation of this Mechanic Schedule Requirements Report. A request to view the Mechanic Schedule Requirements Report will read this generated Mechanic Schedule Requirements Report.		
Implementation Constraints and Specifications:	None		
Assumptions:	The Mechanic will be able to view the Mechanic Schedule Requirements Report, the same as the Administrator.		
Open Issues:	None		

2.3.12.3. Generate Student Report

System Name:	VVs
Author:	Tapiwarufaro Ndoro
Date:	15/06/2020
Version:	1.0

Use Case Name:	Generate Student Report	Use Case Type:
Use Case ID:	11.3	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	Administrator	
Primary System Actor	None	
Other Participating Actors:	None	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process of a Student Report being generated by the system after the Administrators request. The Administrator enters their Year criteria and School Term for the Student Report generation. A Chart is displayed with summaries of the Pass Rate and overall Attendance percentages per school. Tables with Control Breaks per school are also displayed. The Administrator can download this report. This use case narrative details the controls displayed and populated by the system.	
Pre-Condition:	The Administrator must be logged into the system.	

Trigger:	The Administrator requests to Generate a Student Report.			
Typical Course Of Events:	Actor Action	System Response		
		Manual Action	Automated Action	
	Step 1: The Administrator navigates from the "Homepage" screen via the Navigation Bar in the System Management Combo box, to the "Reporting" screen and clicks the "Generate Student Report" button.		Step 2: The System will load the "Student Report" screen with the following controls:	

			Combo box is used for.						
			<table><tr><td>"Year"</td><td>Combo box</td><td>Used to select the Year criteria of the report generation period.</td></tr><tr><td>"School Term"</td><td>Combo box</td><td>Used to select the School term criteria of the report generation period.</td></tr></table>	"Year"	Combo box	Used to select the Year criteria of the report generation period.	"School Term"	Combo box	Used to select the School term criteria of the report generation period.
	"Year"	Combo box	Used to select the Year criteria of the report generation period.						
	"School Term"	Combo box	Used to select the School term criteria of the report generation period.						
			The "Year" Combo box is populated with Years from the First Year with data available for generation in the system, thus not allowing for invalid selections. The "School Term" Combo box options are populated by using a FIRESQL READ query with data from the following: <table><tr><th>Entity</th><th>Attributes</th></tr><tr><td>MarkType</td><td>MarkType</td></tr></table>	Entity	Attributes	MarkType	MarkType		
Entity	Attributes								
MarkType	MarkType								
Step 3: The administrator selects the Year and School Term criteria by choosing the valid options available in the Combo boxes.									
Step 4: The Administrator clicks the "Generate Report" button. [alt]		Step 5: The system captures the desired criteria and ensures that both the "year" and "School Term" options are selected in the Combo boxes [alt]							

Step 6: The system uses the criteria to generate and display a Chart after extracting data using a FIRESQL READ query with the following:

Output	Attribute	Entity
School Name	SchoolName	School
"Mark %"	(Calculated as the overall pass rate for that school)	StudentMark
"Attendance %"	(Calculated as an average of all the student's Attendance % in that school)	Attendance

And tables populated by a FIRESQL READ query displaying control breaks with the following:

Entity	Attributes	Input Name
Student	StudentName	Student Name
BicycleStudent	BicycleID	Bicycle Code
Student	(Calculated from the Student DoB)	Age
Attendance	(Calculated from the count of Date where they were present over total school days)	Attendance %

			StudentMark	Mark	Average Mark &
			Running totals are displayed at the top and end of each table as follows:		
			Output	Attribute	Entity
			"Name of School"	SchoolName	School
			"Total Students:"	(Count the number of StudentID's of students in the school table on display)	Student
			"Average School Attendance"	(Calculated as an average of all the student's Attendance % in that school)	Attendance
			"Overall School Average"	(Calculated as an average of all the student's Mark % in that school)	StudentMark
			The "Download Report" Button becomes active [alt]		
	Step 7: The Administrator clicks the "Download Report" button. [alt]		Step 8: The system generates a PDF download file and displays the following: [alt]		
			Text/Label	Control Type	Additional Info

			Success Dialog Box Title Used to indicate what type of message is being displayed.
			"Student Report has been downloaded successfully." Dialog Box Message Used to inform the User of the result.
			Ok Button Used to accept the result.
			Step 9: The system creates and saves the following in the database by using a FIRESQL INSERT query: AuditTrail Table AuditID DateTime TransactionDescription UserID
Alternate Courses:	Alt Step 4: The Administrator closes the Student Report. Terminate Use Case.		
	Alt Step 5: In the case of an unselected Combo box, the system notifies the Administrator of the Input error below the input field with the following:		
	Text/Label "Please select an option."	Control Type Label (with a light red background color fill)	Additional Info Used to inform the Administrator of the result.

	"X"	Icon	Used to indicate an error result.
	Redirect to Step 3.		
	Alt Step 6: There was an error in generating a Student Report. Display the following:		
	Text/Label	Control Type	Additional Info
	Failed	Dialog Box Title	Used to indicate what type of message is being displayed.
	"Student Report failed to generate. Please try again."	Dialog Box Message	Used to inform the User of the result.
	"Help icon"	Button	Used to Load the Help document.
	Ok	Button	Used to accept the result.
	Redirect to Step 3.		
	Alt Step 7: The Administrator closes the Student Report. Terminate Use Case.		
	Alt Step 8: There was an error in generating a download file for the Administrator. Display the following:		
	Text/Label	Control Type	Additional Info
	Error	Dialog Box Title	Used to indicate what type of message is being displayed.

	"Download failed. Please try again."	Dialog Box Message	Used to inform the User of the result.
	"Help icon"	Button	Used to Load the Help document.
	Ok	Button	Used to accept the result.
	Redirect to Step 5.		
Conclusion:	The use case concludes when the Administrator successfully Generates a Student Report, Views it, and/or Downloads it.		
Post-Condition:	- The downloaded Student Report gets saved in the Administrators device. - The AuditTrail table is successfully updated.		
Business Rules	Only an Administrator can Generate a Student Report.		
Implementation Constraints and Specifications:	None		
Assumptions:	None		
Open Issues:	None		

2.3.12.4. Generate Bicycle History Report

System Name:	VVs
Author:	Tapiwarufaro Ndoro
Date:	15/06/2020
Version:	1.0

Use Case Name:	Generate Bicycle History Report	Use Case Type:
Use Case ID:	11.4	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	Administrator	
Primary System Actor	None	
Other Participating Actors:	None	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process of a Bicycle History Report being generated by the system after the Administrators request. The Administrator enters their School criteria and Job Type for the Bicycle History Report generation. Tables with Control Breaks per school are also displayed. The Administrator can download this report. This use case narrative details the controls displayed and populated by the system.	
Pre-Condition:	The Administrator must be logged into the system.	
Trigger:	The Administrator requests to Generate a Bicycle History Report.	

Typical Course Of Events:	Actor Action	System Response		
		Manual Action	Automated Action	
	Step 1: The Administrator navigates from the "Homepage" screen via the Navigation Bar in the System Management Combo box, to the "Reporting" screen and clicks the "Generate Bicycle History Report" button.		Step 2: The System will load the "Bicycle History Report" screen with the following controls:	
			Text/Label	Control Type
			Navigation	Navigation-bar
			Download Report	Button (Disabled until a report is generated)
			Generate Report	Button
			Bicycle History Report	Banner
			Choose School:	Label
			Choose Job Type:	Label

			Combo box is used for.			
			<table><tr><td>"School"</td><td>Combo box</td><td>Used to select the School criteria of the report generation period.</td></tr></table>	"School"	Combo box	Used to select the School criteria of the report generation period.
	"School"	Combo box	Used to select the School criteria of the report generation period.			
			<table><tr><td>"Job type"</td><td>Combo box</td><td>Used to select the Job Type criteria of the report generation period.</td></tr></table>	"Job type"	Combo box	Used to select the Job Type criteria of the report generation period.
	"Job type"	Combo box	Used to select the Job Type criteria of the report generation period.			
		The "Job Type" Combo box is populated with either "Fault Task" or "Maintenance Task" The "School" Combo box options are populated by using a FIRESQL READ query with data from the following: <table><tr><th>Entity</th><th>Attributes</th></tr><tr><td>School</td><td>School Name</td></tr></table>	Entity	Attributes	School	School Name
Entity	Attributes					
School	School Name					
Step 3: The administrator selects the Job Type and School criteria by choosing the valid options available in the Combo boxes.						
Step 4: The Administrator clicks the "Generate Report" button. [alt]		Step 5: The system captures the desired criteria and ensures that both the "School" and "Job Type" options are selected in the Combo boxes [alt]				

Step 6: The system uses the criteria to generate the report after extracting data using a FIRESQL READ query. Tables populated by a FIRESQL READ query displaying control breaks with the following:

Entity	Attributes	Input Name
ScheduledMaintenance & RepairJob	Scheduled MaintenanceID & JobID	Job Code
(Depends on the selected Job Type)	(Depends on the selected Job Type)	
Student	StudentName & StudentSurname	Student Name
FaultTask & MaintenanceTask Description	FaultTaskDescription & MaintenanceTask Description	Job Description
MechanicJobStatus	(System determined by the date the Job Status was set to "Complete")	Date Completed

Running totals are displayed at the top and end of each table as follows:

Output	Attribute	Entity
"Bicycle Code"	BicycleID	Bicycle

			<table><tr><td>"Status"</td><td>BicycleStatus</td><td>BicycleStatus</td></tr><tr><td>"Total Jobs:"</td><td>(Count the number of rows in the table on display)</td><td>None</td></tr><tr><td>"Date Added"</td><td>DateScheduled</td><td>MechanicSchedule</td></tr></table>	"Status"	BicycleStatus	BicycleStatus	"Total Jobs:"	(Count the number of rows in the table on display)	None	"Date Added"	DateScheduled	MechanicSchedule		
	"Status"	BicycleStatus	BicycleStatus											
	"Total Jobs:"	(Count the number of rows in the table on display)	None											
	"Date Added"	DateScheduled	MechanicSchedule											
		The "Download Report" Button becomes active [alt]												
Step 7: The Administrator clicks the "Download Report" button. [alt]		Step 8: The system generates a PDF download file and displays the following: [alt] <table><tr><td>Text/Label</td><td>Control Type</td><td>Additional Info</td></tr><tr><td>Success</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed.</td></tr><tr><td>"Bicycle History Report has been downloaded successfully."</td><td>Dialog Box Message</td><td>Used to inform the User of the result.</td></tr><tr><td>Ok</td><td>Button</td><td>Used to accept the result.</td></tr></table>	Text/Label	Control Type	Additional Info	Success	Dialog Box Title	Used to indicate what type of message is being displayed.	"Bicycle History Report has been downloaded successfully."	Dialog Box Message	Used to inform the User of the result.	Ok	Button	Used to accept the result.
Text/Label	Control Type	Additional Info												
Success	Dialog Box Title	Used to indicate what type of message is being displayed.												
"Bicycle History Report has been downloaded successfully."	Dialog Box Message	Used to inform the User of the result.												
Ok	Button	Used to accept the result.												
		Step 9: The system creates and saves the following in the database by using a FIRESQL INSERT query: <table><tr><td>AuditTrail Table</td></tr><tr><td>AuditID</td></tr></table>	AuditTrail Table	AuditID										
AuditTrail Table														
AuditID														

Alternate Courses:			DateTime
			TransactionDescription
			UserID
	Alt Step 4: The Administrator closes the Bicycle History Report. Terminate Use Case.		
	Alt Step 5: In the case of an unselected Combo box, the system notifies the Administrator of the Input error below the input field with the following:		
	Text/Label	Control Type	Additional Info
	"Please select an option."	Label (with a light red background color fill)	Used to inform the Administrator of the result.
	"X"	Icon	Used to indicate an error result.
	Redirect to Step 3.		
	Alt Step 6: There was an error in generating a Bicycle History Report. Display the following:		
	Text/Label	Control Type	Additional Info
	Failed	Dialog Box Title	Used to indicate what type of message is being displayed.
	"Bicycle History Report failed to generate. Please try again."	Dialog Box Message	Used to inform the User of the result.
	"Help icon"	Button	Used to Load the Help document.

	Ok	Button	Used to accept the result.
	Redirect to Step 3.		
	Alt Step 7: The Administrator closes the Bicycle History Report. Terminate Use Case.		
	Alt Step 8: There was an error in generating a download file for the Administrator. Display the following:		
	Text/Label	Control Type	Additional Info
	Error	Dialog Box Title	Used to indicate what type of message is being displayed.
	"Download failed. Please try again."	Dialog Box Message	Used to inform the User of the result.
	"Help icon"	Button	Used to Load the Help document.
	Ok	Button	Used to accept the result.
	Redirect to Step 5.		
Conclusion:	The use case concludes when the Administrator successfully Generates a Bicycle History Report, Views it, and/or Downloads it.		
Post-Condition:	• The downloaded Bicycle History Report gets saved in the Administrators device. • The AuditTrail table is successfully updated.		
Business Rules	• Only an Administrator can Generate a Bicycle History Report.		

Implementation Constraints and Specifications:	• None
Assumptions:	• None
Open Issues:	• None

2.3.12.5. Generate Log Report

System Name:	VVs
Author:	Tapiwarufaro Ndoro
Date:	15/06/2020
Version:	1.0

Use Case Name:	Generate Log Report	Use Case Type:
Use Case ID:	11.5	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	Administrator	
Primary System Actor	None	
Other Participating Actors:	None	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process of a Log Report being generated by the system after the Administrators request. The Administrator enters their date criteria for the Log Report generation. Tables with Control Breaks per Antenna are also displayed. The Administrator can download this report. This use case narrative details the controls displayed and populated by the system.	
Pre-Condition:	The Administrator must be logged into the system.	
Trigger:	The Administrator requests to Generate a Log Report.	

Typical Course Of Events:	Actor Action	System Response		
		Manual Action	Automated Action	
	Step 1: The Administrator navigates from the "Homepage" screen via the Navigation Bar in the System Management Combo box, to the "Reporting" screen and clicks the "Generate Log Report" button.		Step 2: The System will load the "Log Report" screen with the following controls:	
			Text/Label	Control Type
			Navigation	Navigation-bar
			Download Report	Button (Disabled until a report is generated)
			Generate Report	Button
			Log Report	Banner
			Choose Start Date:	Label
			Choose End Date:	Label

			Date-Picker is used for.						
			<table><tr><td>"Start Date"</td><td>Date-Picker</td><td>Used to select the start date of the report generation period.</td></tr><tr><td>"End Date"</td><td>Date-Picker</td><td>Used to select the end date of the report generation period.</td></tr></table>	"Start Date"	Date-Picker	Used to select the start date of the report generation period.	"End Date"	Date-Picker	Used to select the end date of the report generation period.
	"Start Date"	Date-Picker	Used to select the start date of the report generation period.						
	"End Date"	Date-Picker	Used to select the end date of the report generation period.						
			The Date-Pickers are populated with Dates already within acceptable starting date and ending date ranges, not allowing for invalid selections.						
Step 3: The administrator selects the Start and End dates by choosing the valid options available in the Date-Pickers.									
Step 4: The Administrator clicks the "Generate Report" button. [alt]		Step 5: The system captures the desired date criteria and ensures that both the "Start" and "End" Dates are selected in the Date-Pickers. [alt]							
		Step 6: The system uses the criteria to generate the report after extracting data using a FIRESQL READ query. Tables populated by a FIRESQL READ query displaying control breaks with the following:							
		<table><tr><td>Entity</td><td>Attributes</td><td>Input Name</td></tr></table>	Entity	Attributes	Input Name				
Entity	Attributes	Input Name							

			Tag	TagID	Tag Serial Number
			Bicycle	BicycleID	Bicycle Code
			Student	StudentName & StudentSurname	Student Name
			Log	Time (Extracts the Time only)	Time
			Log	DateTime (Extracts the Date only)	Date
			Running totals are displayed at the top and end of each table as follows:		
			Output	Attribute	Entity
			"Antenna Serial Number"	AntennaID	Antenna
			"Status"	AntennaStatus	AntennaStatus
			"Total Tags:"	(Count the number of rows in the table on display)	None
			"Coordinates"	Co-ordinates	Antenna
			The "Download Report" Button becomes active [alt]		

	Step 7: The Administrator clicks the “Download Report” button. [alt]		Step 8: The system generates a PDF download file and displays the following: [alt] <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Success</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed.</td></tr><tr><td>“Log Report has been downloaded successfully.”</td><td>Dialog Box Message</td><td>Used to inform the User of the result.</td></tr><tr><td>Ok</td><td>Button</td><td>Used to accept the result.</td></tr></table>	Text/Label	Control Type	Additional Info	Success	Dialog Box Title	Used to indicate what type of message is being displayed.	“Log Report has been downloaded successfully.”	Dialog Box Message	Used to inform the User of the result.	Ok	Button	Used to accept the result.
	Text/Label	Control Type	Additional Info												
Success	Dialog Box Title	Used to indicate what type of message is being displayed.													
“Log Report has been downloaded successfully.”	Dialog Box Message	Used to inform the User of the result.													
Ok	Button	Used to accept the result.													
			Step 9: The system creates and saves the following in the database by using a FIRESQL INSERT query: AuditTrail Table AuditID DateTime TransactionDescription UserID												
Alternate Courses:	Alt Step 4: The Administrator closes the Log Report. Terminate Use Case.														
	Alt Step 5: In the case of an unselected Date from the Date-Picker, the system notifies the Administrator of the Input error below the input field with the following: <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr></table>			Text/Label	Control Type	Additional Info									
Text/Label	Control Type	Additional Info													

	"Please select a date."	Label (with a light red background color fill)	Used to inform the Administrator of the result.
	"X"	Icon	Used to indicate an error result.
	Redirect to Step 3.		
	Alt Step 6: There was an error in generating a Log Report. Display the following:		
	Text/Label	Control Type	Additional Info
	Failed	Dialog Box Title	Used to indicate what type of message is being displayed.
	"Log Report failed to generate. Please try again."	Dialog Box Message	Used to inform the User of the result.
	"Help icon"	Button	Used to Load the Help document.
	Ok	Button	Used to accept the result.
	Redirect to Step 3.		
	Alt Step 7: The Administrator closes the Log Report. Terminate Use Case.		
	Alt Step 8: There was an error in generating a download file for the Administrator. Display the following:		
	Text/Label	Control Type	Additional Info
	Error	Dialog Box Title	Used to indicate what type of

			message is being displayed.
	"Download failed. Please try again."	Dialog Box Message	Used to inform the User of the result.
	"Help icon"	Button	Used to Load the Help document.
	Ok	Button	Used to accept the result.
	Redirect to Step 5.		
Conclusion:	The use case concludes when the Administrator successfully Generates a Log Report, Views it, and/or Downloads it.		
Post-Condition:	- The downloaded Log Report gets saved in the Administrators device. - The AuditTrail table is successfully updated.		
Business Rules	Only an Administrator can Generate a Log Report.		
Implementation Constraints and Specifications:	None		
Assumptions:	None		
Open Issues:	None		

2.3.12.6. Generate Parts Used Report

System Name:	VVs
Author:	Tapiwarufaro Ndoro
Date:	15/06/2020
Version:	1.0

Use Case Name:	Generate Parts Used Report	Use Case Type:
Use Case ID:	11.6	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	Administrator	
Primary System Actor	None	
Other Participating Actors:	None	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process of a Parts Used Report being generated by the system after the Administrators request. The Administrator enters their date criteria for the Parts Used Report generation and then a single table with the data is displayed. The Administrator can also download this report. This use case details the controls displayed and populated by the system.	
Pre-Condition:	The Administrator must be logged into the system.	
Trigger:	The Administrator requests to Generate a Parts Used Report.	

Typical Course Of Events:	Actor Action	System Response		
		Manual Action	Automated Action	
	Step 1: The Administrator navigates from the "Homepage" screen via the Navigation Bar in the System Management Combo box, to the "Reporting" screen and clicks the "Generate Parts Used Report" button.		Step 2: The System will load the "Parts Used Report" screen with the following controls:	
			Text/Label	Control Type
			Navigation	Navigation-bar
			Download Report	Button (Disabled until a report is generated)
			Generate Report	Button
			Parts Used Report	Banner
			Choose Start Date:	Label
			Choose End Date:	Label

			Date-Picker is used for.						
			<table><tr><td>"Start Date"</td><td>Date-Picker</td><td>Used to select the start date of the report generation period.</td></tr><tr><td>"End Date"</td><td>Date-Picker</td><td>Used to select the end date of the report generation period.</td></tr></table>	"Start Date"	Date-Picker	Used to select the start date of the report generation period.	"End Date"	Date-Picker	Used to select the end date of the report generation period.
	"Start Date"	Date-Picker	Used to select the start date of the report generation period.						
	"End Date"	Date-Picker	Used to select the end date of the report generation period.						
			The Date-Pickers are populated with Dates already within acceptable starting date and ending date ranges, not allowing for invalid selections.						
Step 3: The administrator selects the Start and End dates by choosing the valid options available in the Date-Pickers.									
Step 4: The Administrator clicks the "Generate Report" button. [alt]		Step 5: The system captures the desired date criteria and ensures that both the "Start" and "End" Dates are selected in the Date-Pickers. [alt]							
		Step 6: The system uses the duration between the "Start" and "End" Dates to generate a Table view of the "Parts Used Report". The table is populated using a FIRESQL READ query for all data within the criteria and displays the following: [alt]							
		<table><tr><td>Entity</td><td>Attributes</td><td>Input Name</td></tr></table>	Entity	Attributes	Input Name				
Entity	Attributes	Input Name							

			RepairJob & ScheduledMaintenance	(Job Type determined by back-end criteria)	Job
			BicycleStudent	BicycleID	Bicycle Code
			Student	StudentName & StudentSurname	Student Name
			FaultTask & MaintenanceTask Description	FaultTaskDescription & MaintenanceTask Description	Description
			JobFaultTask	PartUsed (Where the Boolean value is True), display relevant Parts	Parts Used
			MechanicJobStatus	(System determined by the date the Job Status was set to "Complete")	Date Completed
			A sub-table is generated at run-time to show the following:		
			Output	Attribute	Entity
			Parts Used	Parts Used (Groups together the Parts Used in	ReportFaultTask

			the main table displayed <table><tr><td>Quantity</td><td>(Count the number of occurrences)</td><td>None</td></tr></table> The "Download Report" Button becomes active	Quantity	(Count the number of occurrences)	None									
	Quantity	(Count the number of occurrences)	None												
	Step 7: The Administrator clicks the "Download Report" button. [alt]		Step 8: The system generates a PDF download file and displays the following: [alt] <table><tr><td>Text/Label</td><td>Control Type</td><td>Additional Info</td></tr><tr><td>Success</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed.</td></tr><tr><td>"Parts Used Report has been downloaded successfully."</td><td>Dialog Box Message</td><td>Used to inform the User of the result.</td></tr><tr><td>Ok</td><td>Button</td><td>Used to accept the result.</td></tr></table>	Text/Label	Control Type	Additional Info	Success	Dialog Box Title	Used to indicate what type of message is being displayed.	"Parts Used Report has been downloaded successfully."	Dialog Box Message	Used to inform the User of the result.	Ok	Button	Used to accept the result.
	Text/Label	Control Type	Additional Info												
Success	Dialog Box Title	Used to indicate what type of message is being displayed.													
"Parts Used Report has been downloaded successfully."	Dialog Box Message	Used to inform the User of the result.													
Ok	Button	Used to accept the result.													
		Step 9: The system creates and saves the following in the database by using a FIRESQL INSERT query: AuditTrail Table AuditID DateTime TransactionDescription													

			UserID															
Alternate Courses:	Alt Step 4: The Administrator closes the Parts Used Report. Terminate Use Case.																	
	Alt Step 5: In the case of an unselected Date from the Date-Picker, the system notifies the Administrator of the Input error below the input field with the following:																	
	<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>"Please select a date."</td><td>Label (with a light red background color fill)</td><td>Used to inform the Administrator of the result.</td></tr><tr><td>"X"</td><td>Icon</td><td>Used to indicate an error result.</td></tr></table>			Text/Label	Control Type	Additional Info	"Please select a date."	Label (with a light red background color fill)	Used to inform the Administrator of the result.	"X"	Icon	Used to indicate an error result.						
	Text/Label	Control Type	Additional Info															
	"Please select a date."	Label (with a light red background color fill)	Used to inform the Administrator of the result.															
	"X"	Icon	Used to indicate an error result.															
	Redirect to Step 3.																	
	Alt Step 6: There was an error in generating a Parts Used Report. Display the following:																	
	<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Failed</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed.</td></tr><tr><td>"Parts Used Report failed to generate. Please try again."</td><td>Dialog Box Message</td><td>Used to inform the User of the result.</td></tr><tr><td>"Help icon"</td><td>Button</td><td>Used to Load the Help document.</td></tr><tr><td>Ok</td><td>Button</td><td>Used to accept the result.</td></tr></table>			Text/Label	Control Type	Additional Info	Failed	Dialog Box Title	Used to indicate what type of message is being displayed.	"Parts Used Report failed to generate. Please try again."	Dialog Box Message	Used to inform the User of the result.	"Help icon"	Button	Used to Load the Help document.	Ok	Button	Used to accept the result.
	Text/Label	Control Type	Additional Info															
Failed	Dialog Box Title	Used to indicate what type of message is being displayed.																
"Parts Used Report failed to generate. Please try again."	Dialog Box Message	Used to inform the User of the result.																
"Help icon"	Button	Used to Load the Help document.																
Ok	Button	Used to accept the result.																

	Redirect to Step 3.															
	Alt Step 7: The Administrator closes the Parts Used Report. Terminate Use Case.															
	Alt Step 8: There was an error in generating a download file for the Administrator. Display the following:															
	<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Error</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed.</td></tr><tr><td>"Download failed. Please try again."</td><td>Dialog Box Message</td><td>Used to inform the User of the result.</td></tr><tr><td>"Help icon"</td><td>Button</td><td>Used to Load the Help document.</td></tr><tr><td>Ok</td><td>Button</td><td>Used to accept the result.</td></tr></table>	Text/Label	Control Type	Additional Info	Error	Dialog Box Title	Used to indicate what type of message is being displayed.	"Download failed. Please try again."	Dialog Box Message	Used to inform the User of the result.	"Help icon"	Button	Used to Load the Help document.	Ok	Button	Used to accept the result.
	Text/Label	Control Type	Additional Info													
	Error	Dialog Box Title	Used to indicate what type of message is being displayed.													
"Download failed. Please try again."	Dialog Box Message	Used to inform the User of the result.														
"Help icon"	Button	Used to Load the Help document.														
Ok	Button	Used to accept the result.														
Redirect to Step 5.																
Conclusion:	The use case concludes when the Administrator successfully Generates a Parts Used Report, Views it, and/or Downloads it.															
Post-Condition:	• The downloaded Parts Used Report gets saved in the Administrators device. • The AuditTrail table is successfully updated.															
Business Rules	• Only an Administrator can Generate a Parts Used Report.															
Implementation Constraints and Specifications:	• None															

Assumptions:	• None
Open Issues:	• None

2.3.13. Sub-system 12: Notification

2.3.13.1. *Send Student did not Arrive notification*

System Name:	VVs
Author:	Hansen Li
Date:	18/06/2020
Version:	1.1

Use Case Name:	Send Student did not arrive notification	Use Case Type:
Use Case ID:	12.1	Business Requirements: ☐
Priority:	Low	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	Time (PBA)	
Primary System Actor	None	
Other Participating Actors:	Student (ERA)	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process of the System sending a notification to a specific student's parents if they did not arrive at school	
Pre-Condition:	The student and their respective parents are registered on the system	
Trigger:	The student has not arrived for school	

Typical Course Of Events:	Actor Action	System Response												
		Manual Action	Automated Action											
	Step 1: The student has not arrived for school.		Step 2: The system uses a FIRESQL READ from the following table to identify if a student has not arrived by checking the following: <table><tr><th>Entity</th><th>Attributes</th></tr><tr><td>Log</td><td>LogID</td></tr><tr><td>Log</td><td>DateTime</td></tr><tr><td>Log</td><td>TagID</td></tr><tr><td>Log</td><td>AntennaID</td></tr></table>	Entity	Attributes	Log	LogID	Log	DateTime	Log	TagID	Log	AntennaID	
	Entity	Attributes												
	Log	LogID												
	Log	DateTime												
	Log	TagID												
	Log	AntennaID												
			Step 3: If a log entry does not have a corresponding log entry then a notification must be sent out											
			Step 4: System uses a FIRESQL READ to read data from the following tables: <table><tr><th>Entity</th><th>Attributes</th></tr><tr><td>Bicycle</td><td>BicycleID</td></tr><tr><td>Student</td><td>StudentName</td></tr><tr><td>Student</td><td>StudentSurname</td></tr><tr><td>Guardian</td><td>GuardianName</td></tr><tr><td>Guardian</td><td>GuardianPhoneN umber</td></tr></table>	Entity	Attributes	Bicycle	BicycleID	Student	StudentName	Student	StudentSurname	Guardian	GuardianName	Guardian
Entity	Attributes													
Bicycle	BicycleID													
Student	StudentName													
Student	StudentSurname													
Guardian	GuardianName													
Guardian	GuardianPhoneN umber													
		Step 5: The system generates the notification												

			Step 6: The system sends an SMS to the parents with the following details: [alt]						
			<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Notification heading</td><td>Label</td><td>Heading for the form displayed.</td></tr><tr><td>Message</td><td>Paragraph</td><td>Used to provide the parents with the necessary details</td></tr></table>	Text/Label	Control Type	Additional Info	Notification heading	Label	Heading for the form displayed.
Text/Label	Control Type	Additional Info							
Notification heading	Label	Heading for the form displayed.							
Message	Paragraph	Used to provide the parents with the necessary details							
			Step 7: The system creates and saves the following in the database by using a FIRESQL INSERT query: AuditTrail Table AuditID DateTime TransactionDescription UserID						
Alternate Courses:	Alt Step 6: The SMS was not sent successfully; the system will notify the Administrator. Return to Step 1.								
Conclusion:	The use case narrative concludes when the SMS is sent to the parents								
Post-Condition:	The system has sent an SMS to the parents								
Business Rules	The SMS must only be sent out if the child hasn't arrived								
Implementation Constraints and Specifications:	None								

Assumptions:	The parent's details are correct
Open Issues:	None

2.3.13.2. *Send Check-in notification*

System Name:	VVs
Author:	Hansen Li
Date:	18/06/2020
Version:	1.1

Use Case Name:	Send Student Check-In notification	Use Case Type:
Use Case ID:	12.2	Business Requirements: ☐
Priority:	Low	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	Time (PBA)	
Primary System Actor	None	
Other Participating Actors:	Student (ERA)	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process of the System sending a notification to a specific student's bike that needs to be taken checked in to the mechanic.	
Pre-Condition:	The student's bike requires a maintenance	
Trigger:	A week before the maintenance date arrives	
Typical Course	Actor Action	System Response

Of Events:		Manual Action	Automated Action																
	Step 1: A week before a specific bicycle's check-in date arrives		Step 2: The system uses a FIRESQL READ from the following tables to get the information of the student with the bicycle due for a check-in: <table><tr><th>Entity</th><th>Attributes</th></tr><tr><td>BicycleStudent</td><td>BicycleID</td></tr><tr><td>BicycleStudent</td><td>StudentID</td></tr></table> <table><tr><th>Entity</th><th>Attributes</th></tr><tr><td>Student</td><td>StudentName</td></tr><tr><td>Student</td><td>PhoneNumber</td></tr></table> <table><tr><th>Entity</th><th>Attributes</th></tr><tr><td>Bicycle</td><td>BicycleID</td></tr></table>	Entity	Attributes	BicycleStudent	BicycleID	BicycleStudent	StudentID	Entity	Attributes	Student	StudentName	Student	PhoneNumber	Entity	Attributes	Bicycle	BicycleID
	Entity	Attributes																	
	BicycleStudent	BicycleID																	
BicycleStudent	StudentID																		
Entity	Attributes																		
Student	StudentName																		
Student	PhoneNumber																		
Entity	Attributes																		
Bicycle	BicycleID																		
		Step 3: System generates the notification																	
		Step 4: [alt] The system sends the SMS to the student with the following details: <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Notification heading</td><td>Label</td><td>Heading for the form displayed.</td></tr><tr><td>Message</td><td>Paragraph</td><td>Used to provide the Student with</td></tr></table>	Text/Label	Control Type	Additional Info	Notification heading	Label	Heading for the form displayed.	Message	Paragraph	Used to provide the Student with								
Text/Label	Control Type	Additional Info																	
Notification heading	Label	Heading for the form displayed.																	
Message	Paragraph	Used to provide the Student with																	

			the necessary details
			Step 5: The system creates and saves the following in the database by using a FIRESQL INSERT query: AuditTrail Table AuditID DateTime TransactionDescription UserID
Alternate Courses:	Alt Step 4: The SMS was not sent successfully; the system will notify the Administrator. Return to Step 1.		
Conclusion:	The use case narrative concludes when the SMS is sent to the student		
Post-Condition:	The system has sent an SMS to the student		
Business Rules	The SMS must be sent a week before the check-in is due		
Implementation Constraints and Specifications:	None		
Assumptions:	The student's phone number that is saved in the database is correct		
Open Issues:	None		

2.3.14. Sub-system 13: Administration

2.3.14.1. Backup Data

System Name:	VVs
Author:	Cameron King
Date:	15/06/2020
Version:	1.0

Use Case Name:	Backup Data	Use Case Type:
Use Case ID:	13.1	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	Administrator	
Primary System Actor	None	
Other Participating Actors:	None	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process of an Administrator backing up the system data. The Use Case starts when the Administrator navigates by selecting the "System Data Management" option under the "System Management" Combo box in the Navigation bar. From there, they click the "Backup System" button, and the system requests confirmation of the backup.	
Pre-Condition:	The Administrator must be logged into the system.	

Trigger:	The Administrator requests to backup the system.			
Typical Course Of Events:	Actor Action	System Response		
		Manual Action	Automated Action	
	Step 1: The Administrator selects the “System Data Management” option and under the “System Management” Combo box in the Navigation bar.		Step 2: The system Displays the “System Backup and Restore” screen and populates it with the following details:	

			Entity	Attributes	Input Name
			Backup (External backup database)	BackupID	Backup ID
			Backup (External backup database)	DateTime	Date Time
	Step 3: The Administrator clicks on the "Backup System" button.		Step 4: The system prompts the Administrator to confirm the system backup with the following:		
			Text/Label	Control Type	Additional Info
			Confirmation	Dialog Box Title	Used to indicate what type of message is being displayed.
			"Are you sure you want to Backup the system?"	Dialog Box Message	Used to inform the Administrator of the action.
			Yes	Button	Used to confirm the action.
			No	Button	Used to decline the action.

	Step 5: The Administrator clicks the "Yes" button. [alt]		Step 6: The system performs the backup of all system data, ensuring data integrity. The system saves the backup data and saves the backup securely with the following file naming attributes: <table><tr><th>Entity</th><th>Attributes</th><th>Input Name</th></tr><tr><td>Backup (External backup database)</td><td>BackupID</td><td>Backup ID</td></tr><tr><td>Backup (External backup database)</td><td>DateTime</td><td>Date Time</td></tr></table>	Entity	Attributes	Input Name	Backup (External backup database)	BackupID	Backup ID	Backup (External backup database)	DateTime	Date Time		
	Entity	Attributes	Input Name											
	Backup (External backup database)	BackupID	Backup ID											
Backup (External backup database)	DateTime	Date Time												
		Step 7: The system notifies the Administrator that the system has been successfully backed up. [alt] <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Success</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed.</td></tr><tr><td>"System has been backed up successfully."</td><td>Dialog Box Message</td><td>Used to inform the Administrator of the result.</td></tr><tr><td>Ok</td><td>Button</td><td>Used to accept the result.</td></tr></table>	Text/Label	Control Type	Additional Info	Success	Dialog Box Title	Used to indicate what type of message is being displayed.	"System has been backed up successfully."	Dialog Box Message	Used to inform the Administrator of the result.	Ok	Button	Used to accept the result.
Text/Label	Control Type	Additional Info												
Success	Dialog Box Title	Used to indicate what type of message is being displayed.												
"System has been backed up successfully."	Dialog Box Message	Used to inform the Administrator of the result.												
Ok	Button	Used to accept the result.												
		Step 8: The system creates and saves the following in the database by using a FIRESQL INSERT query: <table><tr><th>AuditTrail Table</th></tr><tr><td>AuditTrailID</td></tr></table>	AuditTrail Table	AuditTrailID										
AuditTrail Table														
AuditTrailID														

			DateTime
			TransactionDescription
			UserID
	Alt Step 5: The Administrator declines to Backup the system by clicking the "No" button. Redirect to Step 2.		
	Alt Step 7: The system backup failed, the system will notify the Administrator of the backup error with the following:		
	Text/Label	Control Type	Additional Info
	Failed	Dialog Box Title	Used to indicate what type of message is being displayed
	"Backup unsuccessful. Please try again"	Dialog Box Message	Used to inform the Administrator of the result.
	"Help icon"	Button	Used to Load the Help document.
	Ok	Button	Used to accept the result.
	Redirect to Step 2.		
Conclusion:	The use case concludes when the Administrator has been notified that the system has been backed up successfully.		
Post-Condition:	- The system has been backed up successfully. - The AuditTrail table is successfully updated.		
Business Rules	- Only an Administrator can back up the system. - The system will be automatically backed up weekly.		

	• The backup will be encrypted.
Implementation Constraints and Specifications:	• None
Assumptions:	• None
Open Issues:	• None

2.3.14.2. Restore Data

System Name:	VVs
Author:	Cameron King
Date:	15/06/2020
Version:	1.0

Use Case Name:	Restore Data	Use Case Type:
Use Case ID:	13.2	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	Administrator	
Primary System Actor	None	
Other Participating Actors:	None	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process of an Administrator restoring the system data. The Use Case starts when the Administrator navigates by selecting the "System Data Management" option under the "System Management" Combo box in the Navigation bar. From there, they click the "Restore" link and the system requests confirmation of the system restore.	
Pre-Condition:	The Administrator must be logged into the system.	
Trigger:	The Administrator requests to restore the system.	

Typical Course Of Events:	Actor Action	System Response																							
		Manual Action	Automated Action																						
	Step 1: The Administrator selects the "System Data Management" option and under the "System Management" Combo box in the Navigation bar.		Step 2: The system Displays the "System Backup and Restore" screen and populates it with the following details:																						
			<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Backup System</td><td>Button</td><td>Used to perform the system backup action.</td></tr><tr><td>Search</td><td>Text-Box</td><td>Used for capturing Search criteria for the specific backup file.</td></tr><tr><td>Restore Management</td><td>Table</td><td>Displays a list of backup files.</td></tr><tr><td>"Restore"</td><td>Link</td><td>Used to perform the system restore action.</td></tr><tr><td>"Back"</td><td>Button</td><td>Returns to Home Page.</td></tr><tr><td>"Help Icon"</td><td>Button</td><td>Used to Load the Help document.</td></tr></table>		Text/Label	Control Type	Additional Info	Backup System	Button	Used to perform the system backup action.	Search	Text-Box	Used for capturing Search criteria for the specific backup file.	Restore Management	Table	Displays a list of backup files.	"Restore"	Link	Used to perform the system restore action.	"Back"	Button	Returns to Home Page.	"Help Icon"	Button	Used to Load the Help document.
			Text/Label	Control Type	Additional Info																				
			Backup System	Button	Used to perform the system backup action.																				
			Search	Text-Box	Used for capturing Search criteria for the specific backup file.																				
			Restore Management	Table	Displays a list of backup files.																				
			"Restore"	Link	Used to perform the system restore action.																				
			"Back"	Button	Returns to Home Page.																				
			"Help Icon"	Button	Used to Load the Help document.																				
			The table is populated by using a FIRESQL READ query with data from the following:																						
<table><tr><th>Entity</th><th>Attributes</th><th>Input Name</th></tr></table>		Entity	Attributes	Input Name																					
Entity	Attributes	Input Name																							

			Backup (External backup database) BackupID Backup ID															
			Backup (External backup database) DateTime Date Time															
	Step 3: The Administrator clicks on the "Restore" link.		Step 4: The system prompts the Administrator to confirm the system backup with the following:															
			<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Confirmation</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed.</td></tr><tr><td>"Are you sure you want to Restore the system?"</td><td>Dialog Box Message</td><td>Used to inform the Administrator of the action.</td></tr><tr><td>Yes</td><td>Button</td><td>Used to confirm the action.</td></tr><tr><td>No</td><td>Button</td><td>Used to decline the action.</td></tr></table>	Text/Label	Control Type	Additional Info	Confirmation	Dialog Box Title	Used to indicate what type of message is being displayed.	"Are you sure you want to Restore the system?"	Dialog Box Message	Used to inform the Administrator of the action.	Yes	Button	Used to confirm the action.	No	Button	Used to decline the action.
	Text/Label	Control Type	Additional Info															
	Confirmation	Dialog Box Title	Used to indicate what type of message is being displayed.															
	"Are you sure you want to Restore the system?"	Dialog Box Message	Used to inform the Administrator of the action.															
	Yes	Button	Used to confirm the action.															
	No	Button	Used to decline the action.															
	Step 5: The Administrator clicks the "Yes" button. [alt]		Step 6: The system performs the system restore from the Backup file that was selected, this is verified by comparing the with the following attributes:															
		<table><tr><th>Entity</th><th>Attributes</th><th>Input Name</th></tr><tr><td>Backup (External backup database)</td><td>BackupID</td><td>Backup ID</td></tr></table>	Entity	Attributes	Input Name	Backup (External backup database)	BackupID	Backup ID										
Entity	Attributes	Input Name																
Backup (External backup database)	BackupID	Backup ID																

			Backup (External backup database)	Date Time	Date Time												
			Step 7: The system notifies the Administrator that the system has been successfully backed up. [alt]														
			<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Success</td><td>Dialog Box Title</td><td>Used to indicate what type of message is being displayed.</td></tr><tr><td>"System has been restored successfully. Please restart the application."</td><td>Dialog Box Message</td><td>Used to inform the Administrator of the result.</td></tr><tr><td>Ok</td><td>Button</td><td>Used to accept the result.</td></tr></table>			Text/Label	Control Type	Additional Info	Success	Dialog Box Title	Used to indicate what type of message is being displayed.	"System has been restored successfully. Please restart the application."	Dialog Box Message	Used to inform the Administrator of the result.	Ok	Button	Used to accept the result.
			Text/Label	Control Type	Additional Info												
			Success	Dialog Box Title	Used to indicate what type of message is being displayed.												
	"System has been restored successfully. Please restart the application."	Dialog Box Message	Used to inform the Administrator of the result.														
	Ok	Button	Used to accept the result.														
			Step 8: The system creates and saves the following in the database by using a FIRESQL INSERT query:														
			<table><tr><td>AuditTrail Table</td></tr><tr><td>AuditTrailID</td></tr><tr><td>DateTime</td></tr><tr><td>TransactionDescription</td></tr><tr><td>UserID</td></tr></table>			AuditTrail Table	AuditTrailID	DateTime	TransactionDescription	UserID							
			AuditTrail Table														
AuditTrailID																	
DateTime																	
TransactionDescription																	
UserID																	
Alt Step 5: The Administrator declines to Restore the system by clicking the "No" button. Redirect to Step 2.																	
Alt Step 7: The system Restore failed, the system will notify the Administrator of the restore error with the following:																	

	Text/Label	Control Type	Additional Info
	Failed	Dialog Box Title	Used to indicate what type of message is being displayed
	"Restore unsuccessful. Please try again"	Dialog Box Message	Used to inform the Administrator of the result.
	"Help icon"	Button	Used to Load the Help document.
	Ok	Button	Used to accept the result.
	Redirect to Step 2.		
Conclusion:	The use case concludes when the Administrator has been notified that the system has been restored successfully.		
Post-Condition:	- The system has been restored successfully. - The AuditTrail table is successfully updated.		
Business Rules	Only an Administrator can restore the system.		
Implementation Constraints and Specifications:	None		
Assumptions:	None		
Open Issues:	None		

2.3.14.3. View Audit Trail Log

System Name:	VVs
Author:	Cameron King
Date:	15/06/2020
Version:	1.0

Use Case Name:	View Audit Trail Log	Use Case Type:
Use Case ID:	13.3	Business Requirements: ☐
Priority:	High	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	Administrator	
Primary System Actor	None	
Other Participating Actors:	None	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process of an Administrator requesting to view the Audit Trail Log of the system. The Use Case starts when the Administrator navigates by selecting the "Audit Trail Management" option under the "System Management" Combo box in the Navigation bar. The system populates the table with the information of all the audit trail entries.	
Pre-Condition:	The Administrator must be logged into the system.	
Trigger:	The Administrator requests to view the Audit Trail Log of the system.	

Typical Course Of Events:	Actor Action	System Response																				
		Manual Action	Automated Action																			
	Step 1: The Administrator selects the "Audit Trail Management" option under the "System Management" Combo box in the Navigation bar.		Step 2: The System invokes Use Case13.4 "Search Audit Trail Entry" and displays the "Audit Trail management" screen with the data of the searched Maintenance Task.																			
			Step 2: The system Displays the "Audit Trail Management" screen and populates it with the following details: <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>View Audit Trail Log</td><td>Button</td><td>Used to load the "Audit Trail Log" screen.</td></tr><tr><td>Search</td><td>Text-Box</td><td>Used for capturing Search criteria for the specific Audit Trail entry.</td></tr><tr><td>Search</td><td>Button</td><td>Used to initiate the search.</td></tr><tr><td>Audit Trail Management</td><td>Table</td><td>Displays a list of Audit Trail entries.</td></tr><tr><td>View Button</td><td>Button</td><td>Used to navigate to the specific Audit Trail entry view.</td></tr></table>		Text/Label	Control Type	Additional Info	View Audit Trail Log	Button	Used to load the "Audit Trail Log" screen.	Search	Text-Box	Used for capturing Search criteria for the specific Audit Trail entry.	Search	Button	Used to initiate the search.	Audit Trail Management	Table	Displays a list of Audit Trail entries.	View Button	Button	Used to navigate to the specific Audit Trail entry view.
Text/Label	Control Type	Additional Info																				
View Audit Trail Log	Button	Used to load the "Audit Trail Log" screen.																				
Search	Text-Box	Used for capturing Search criteria for the specific Audit Trail entry.																				
Search	Button	Used to initiate the search.																				
Audit Trail Management	Table	Displays a list of Audit Trail entries.																				
View Button	Button	Used to navigate to the specific Audit Trail entry view.																				

			The table is populated with data from the following by using a FIRESQL READ query: <table><tr><th>Entity</th><th>Attributes</th><th>Input Name</th></tr><tr><td>AuditTrail</td><td>AuditTrailID</td><td>Audit Trail ID</td></tr><tr><td>AuditTrail</td><td>DateTime</td><td>Date Time</td></tr><tr><td>AuditTrail</td><td>TransactionType</td><td>Transaction Type</td></tr></table>	Entity	Attributes	Input Name	AuditTrail	AuditTrailID	Audit Trail ID	AuditTrail	DateTime	Date Time	AuditTrail	TransactionType	Transaction Type												
	Entity	Attributes	Input Name																								
	AuditTrail	AuditTrailID	Audit Trail ID																								
	AuditTrail	DateTime	Date Time																								
	AuditTrail	TransactionType	Transaction Type																								
	Step 3: The Administrator clicks on the “View Audit Trail Log” button.		Step 4: The System will load the “Audit Trail Log” screen and displays it with the following controls and layout: <table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>Audit Trail Log</td><td>Table</td><td>Displays a list of all the Audit Trail entries.</td></tr></table> The table is populated with data from the following by using a FIRESQL READ query: <table><tr><th>Entity</th><th>Attributes</th><th>Input Name</th></tr><tr><td>AuditTrail</td><td>AuditTrailID</td><td>Audit Trail ID</td></tr><tr><td>AuditTrail</td><td>DateTime</td><td>Date Time</td></tr><tr><td>UserType</td><td>UserType</td><td>User</td></tr><tr><td>AuditTrail</td><td>TransactionType</td><td>Transaction Type</td></tr><tr><td></td><td></td><td></td></tr></table>	Text/Label	Control Type	Additional Info	Audit Trail Log	Table	Displays a list of all the Audit Trail entries.	Entity	Attributes	Input Name	AuditTrail	AuditTrailID	Audit Trail ID	AuditTrail	DateTime	Date Time	UserType	UserType	User	AuditTrail	TransactionType	Transaction Type			
	Text/Label	Control Type	Additional Info																								
	Audit Trail Log	Table	Displays a list of all the Audit Trail entries.																								
	Entity	Attributes	Input Name																								
	AuditTrail	AuditTrailID	Audit Trail ID																								
AuditTrail	DateTime	Date Time																									
UserType	UserType	User																									
AuditTrail	TransactionType	Transaction Type																									
		Step 5: The system creates and saves the following in the database by using a FIRESQL INSERT query: AuditTrail Table																									

Alternate Courses:			AuditTrailID
			DateTime
			TransactionDescription
			UserID
	Alt Step 4: The Audit Trail log failed to load. The system will notify the Administrator of this with the following:		
	Text/Label	Control Type	Additional Info
	"Audit Trail Log could not be loaded. Please try again"	Label	Used to inform the Administrator of the result.
	"X"	Banner (with a light red background color fill)	Used to display the result message.
	Redirect to Step 3.		
Conclusion:	The use case narrative concludes when the searched Antenna details are displayed.		
Post-Condition:	- The Audit Trail log has been displayed successfully. - The AuditTrail table is successfully updated.		
Business Rules	Only an Administrator can view the Audit Trail log of the system.		
Implementation Constraints and Specifications:	None		
Assumptions:	None		
Open Issues:	None		

2.3.14.4. Search Audit Trail Entry

System Name:	VVs
Author:	Cameron King
Date:	15/06/2020
Version:	1.0

Use Case Name:	Search Audit Trail Entry	Use Case Type:
Use Case ID:	13.4	Business Requirements: ☐
Priority:	Medium	System Analysis: ☐
Source:	VVS Prototype Screens	System Design: ☒
Primary Business Actor	Administrator	
Primary System Actor	None	
Other Participating Actors:	None	
Other Interested Stakeholders:	None	
Description:	This Technical Use Case Narrative describes the process of an Administrator searching an Audit Trail entry on the system. The Use Case starts when the Administrator navigates by selecting the "Audit Trail Management" option under the "System Management" Combo box in the Navigation bar. The system populates the table with the information of all the Audit Trail entries.	
Pre-Condition:	The Administrator must be logged into the system.	
Trigger:	The Administrator requests to search for an Audit Trail entry on the system.	

Typical Course Of Events:	Actor Action	System Response																	
		Manual Action	Automated Action																
	Step 1: The Administrator selects the "Audit Trail Management" option under the "System Management" Combo box in the Navigation bar.		Step 2: The system Displays the "Audit Trail Management" screen and populates it with the following details:																
			<table><tr><th>Text/Label</th><th>Control Type</th><th>Additional Info</th></tr><tr><td>View Audit Trail Log</td><td>Button</td><td>Used to load the "Audit Trail Log" screen.</td></tr><tr><td>Search</td><td>Text-Box</td><td>Used for capturing Search criteria for the specific Audit Trail entry.</td></tr><tr><td>Audit Trail Management</td><td>Table</td><td>Displays a list of Audit Trail entries.</td></tr><tr><td>View</td><td>Button</td><td>Used to navigate to the specific Audit Trail entry view.</td></tr></table>		Text/Label	Control Type	Additional Info	View Audit Trail Log	Button	Used to load the "Audit Trail Log" screen.	Search	Text-Box	Used for capturing Search criteria for the specific Audit Trail entry.	Audit Trail Management	Table	Displays a list of Audit Trail entries.	View	Button	Used to navigate to the specific Audit Trail entry view.
			Text/Label	Control Type	Additional Info														
			View Audit Trail Log	Button	Used to load the "Audit Trail Log" screen.														
			Search	Text-Box	Used for capturing Search criteria for the specific Audit Trail entry.														
			Audit Trail Management	Table	Displays a list of Audit Trail entries.														
			View	Button	Used to navigate to the specific Audit Trail entry view.														
			The table is populated with data from the following by using a FIRESQL READ query:																
			<table><tr><th>Entity</th><th>Attributes</th><th>Input Name</th></tr><tr><td>AuditTrail</td><td>AuditTrailID</td><td>Audit Trail ID</td></tr><tr><td>AuditTrail</td><td>DateTime</td><td>Date Time</td></tr><tr><td>UserType</td><td>UserType</td><td>User</td></tr></table>		Entity	Attributes	Input Name	AuditTrail	AuditTrailID	Audit Trail ID	AuditTrail	DateTime	Date Time	UserType	UserType	User			
			Entity	Attributes	Input Name														
AuditTrail	AuditTrailID	Audit Trail ID																	
AuditTrail	DateTime	Date Time																	
UserType	UserType	User																	

			AuditTrail TransactionType Transaction Type								
	Step 3: The Administrator enters the search criteria that they wish to search by. They do so by typing in a keyword in the “Search” Text box.										
	Step 4: The Administrator clicks the search button.		Step 5: The system validates that the captured details are in the correct format: AuditTrail table <table><tr><th>Attribute</th><th>Validation</th></tr><tr><td>AuditTrailID</td><td>Max 10 characters. Numeric characters only.</td></tr><tr><td>UserID</td><td>Max 10 characters. Numeric characters only.</td></tr><tr><td>TransactionType</td><td>Max 50 characters.</td></tr></table>	Attribute	Validation	AuditTrailID	Max 10 characters. Numeric characters only.	UserID	Max 10 characters. Numeric characters only.	TransactionType	Max 50 characters.
	Attribute	Validation									
	AuditTrailID	Max 10 characters. Numeric characters only.									
UserID	Max 10 characters. Numeric characters only.										
TransactionType	Max 50 characters.										
		Step 6: The system uses a FIRESQL READ query to READ the following details in the database: [alt] <table><tr><th>Entity</th><th>Attributes</th></tr><tr><td>AuditTrail</td><td>AuditTrailID</td></tr><tr><td>AuditTrail</td><td>DateTime</td></tr><tr><td>AuditTrail</td><td>TransactionType</td></tr></table>	Entity	Attributes	AuditTrail	AuditTrailID	AuditTrail	DateTime	AuditTrail	TransactionType	
Entity	Attributes										
AuditTrail	AuditTrailID										
AuditTrail	DateTime										
AuditTrail	TransactionType										
		Step 7: The following details are loaded and used to populate the “Audit Trail Management” table by using a FIRESQL READ query:									

			Output	Attribute	Entity
			Audit Trail ID	AuditTrailID	AuditTrail
			Date Time	DateTime	AuditTrail
			Transaction Type	TransactionType	AuditTrail
	Step 8: The Administrator clicks on the "View" Audit Trail entry Button in the "Action" table column.		Step 9: The System will load the "Audit Trail Entry" screen and displays it with the following controls and layout:		
			Text/Label	Control Type	Additional Info
			Menu	Menu-icon	Used to Navigate the side menu-bar.
			Audit Trail Entry	Label	Heading for the pop-up displayed.
			Audit Trail ID	Label	Used to indicate what the corresponding read-only text box is used for.
			User	Label	Used to indicate what the corresponding read-only text box is used for.
			Date Time	Label	Used to indicate what the corresponding read-only text box is used for.

			<table><tr><td>Transaction Type</td><td>Label</td><td>Used to indicate what the corresponding read-only text box is used for.</td></tr><tr><td>Back to search results</td><td>Button</td><td>Used to navigate back to Step 6</td></tr></table> The following fields are populated with data retrieved from the database where the AuditTrailID is the same as the one searched by using a FIRESQL READ query: <table><tr><th>Output</th><th>Attribute</th><th>Entity</th></tr><tr><td>Audit Trail ID</td><td>AuditTrailID</td><td>AuditTrail</td></tr><tr><td>User</td><td>UserType</td><td>UserType</td></tr><tr><td>Date Time</td><td>DateTime</td><td>AuditTrail</td></tr><tr><td>Transaction Type</td><td>TransactionType</td><td>AuditTrail</td></tr></table>	Transaction Type	Label	Used to indicate what the corresponding read-only text box is used for.	Back to search results	Button	Used to navigate back to Step 6	Output	Attribute	Entity	Audit Trail ID	AuditTrailID	AuditTrail	User	UserType	UserType	Date Time	DateTime	AuditTrail	Transaction Type	TransactionType	AuditTrail
	Transaction Type	Label	Used to indicate what the corresponding read-only text box is used for.																					
Back to search results	Button	Used to navigate back to Step 6																						
Output	Attribute	Entity																						
Audit Trail ID	AuditTrailID	AuditTrail																						
User	UserType	UserType																						
Date Time	DateTime	AuditTrail																						
Transaction Type	TransactionType	AuditTrail																						
			Step 10: The system creates and saves the following in the database by using a FIRESQL INSERT query: AuditTrail Table AuditTrailID DateTime TransactionDescription UserID																					
Alternate Courses:	Alt Step 5: The specific Audit Trail entry does not exist in the database. The system will notify the Administrator of this with the following:																							

	Text/Label	Control Type	Additional Info
	"Audit Trail entry not found."	Label	Used to inform the Administrator of the result.
	"X"	Banner (with a light red background color fill)	Used to display the result message.
	Redirect to Step 3.		
Conclusion:	The use case narrative concludes when the searched Audit Trail entry details are displayed.		
Post-Condition:	- The Audit Trail entry has been searched successfully. - The AuditTrail table is successfully updated.		
Business Rules	Only an Administrator can search for an Audit Trail entry on the system.		
Implementation Constraints and Specifications:	None		
Assumptions:	None		
Open Issues:	None		

2.3.15. Conclusion

In conclusion, we were able to gain a lot more insight into each of our processes by listing out each step in such a way that it is easily understandable. We also made various changes to our use case diagram after identifying some needed changes from the narratives.

3. Process-Oriented Design

3.1. Introduction

Process models can vary in terms of the amount of information shown largely depending on what type of diagram is used. Process models show all the inputs and outputs of a process which means that it gets all its information from the use case narratives. Process models are a useful way to identify all the individual flows of data and information in and out of the system.

3.2. Diagrams

Please find all Diagrams associated with Process-Orientated design in Appendix A.

3.3. Conclusion

After reviewing all our Data Flow Diagrams, we can now fully gain an understanding of how our processes work.

4. System Database Design

4.1. Introduction

The entity-relationship diagram (ERD for short) is all the data stores of the system and essentially forms the backbone of the system as all data is going to be stored in tables that are created using the ERD. The ERD is a useful way to map out all the data stores which the system will have and allows us to see the relationships between them so that when we design the database, we will already have a blueprint for it.

4.2. Entity Relationship Diagram

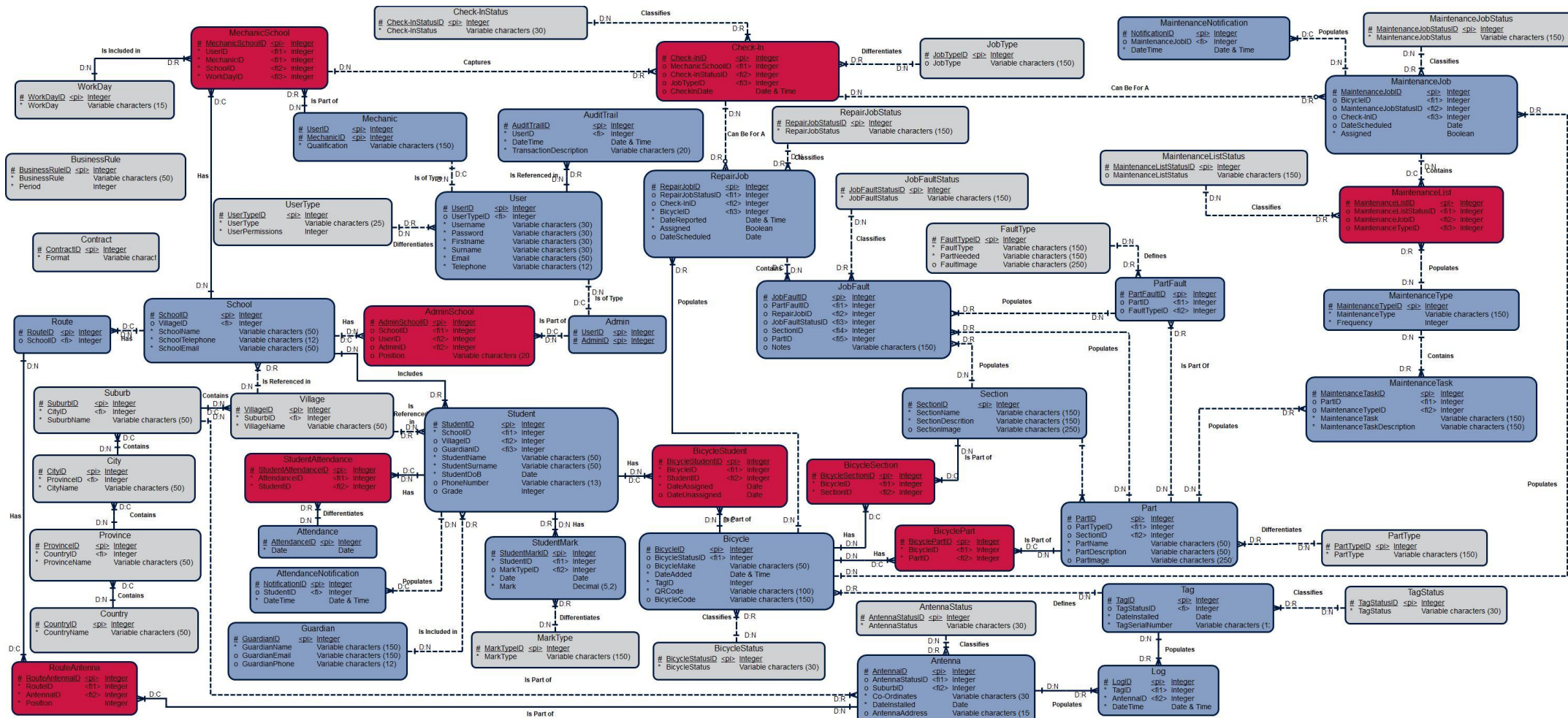


Figure 14 Entity Relationship Diagram

4.3. Size Estimation of Proposed Database

4.3.1. Introduction

The size estimation is used to give us an idea of how much space would be needed on the devices that our system will run on. To calculate the total size of the database I started by calculating an estimated size of an individual entity for each table using [MySQL Data Type Storage Requirements](#) as seen in the table below. Once these values were calculated we then used the expected number of rows for each table to calculate the total size of each entity then adding them together, we got the total size of our proposed database.

4.3.2. Size Estimation

Entity Name	Entity Size (in bytes)	Expected Rows	Total Size (in bytes)
Admin	8	10	80
AdminSchool	37	10	370
Antenna	46	50	2300
AntennaStatus	35	3	105
Attendance	7	201	1407
AttendanceNotification	14	480	6720
AuditTrail	35	1440000	50400000
Bicycle	170	75	12750
BicyclePart	12	25	300
BicycleSection	12	8	96
BicycleStatus	35	3	105
BicycleStudent	18	70	1260
BusinessRule	59	5	295
Check-In	22	1000	22000
Check-InStatus	35	4	140
City	59	70	4130
Contract	255	3	765
Country	55	1	55
FaultType	306	50	15300

Entity Name	Entity Size (in bytes)	Expected Rows	Total Size (in bytes)
Guardian	319	80	25520
JobFault	175	150	26250
JobFaultStatus	155	2	310
JobType	155	2	310
Log	18	1407000	25326000
MaintenanceJob	21	900	18900
MaintenanceJobStatus	155	3	465
MaintenanceList	16	1620	25920
MaintenanceListStatus	155	2	310
MaintenanceNotification	14	950	13300
MaintenanceTask	314	35	10990
MaintenanceType	159	3	477
MarkType	155	2	310
Mechanic	159	4	636
MechanicSchool	20	10	200
Part	110	18	1980
PartFault	12	60	720
PartType	155	5	775
Province	59	9	531
RepairJob	27	100	2700
RepairJobStatus	155	3	465
Route	8	20	160
RouteAntenna	16	60	960
School	123	5	615
Section	306	5	1530
Student	139	100	13900
StudentAttendance	12	18000	216000
StudentMark	20	200	4000

Entity Name	Entity Size (in bytes)	Expected Rows	Total Size (in bytes)
Suburb	59	300	17700
Tag	11	75	825
TagStatus	35	3	105
User	196	18	3528
UserType	34	3	102
Village	59	100	5900
WorkDay	20	5	100
Database Total			76205822

4.3.3. 5 Year Projection

For our 5-year projection, we went on the assumption that the log and AuditTrail entities would not keep entries for more than two years so after the first two years the rate of growth would get reduced by a large amount and would only increase based on new entries. We also accounted for a 30% overhead on the total calculated above as seen in the estimated size for the current period below.

Time	Estimated Growth	Estimated Size
Current		99067568,6
1 year	0,9	188228380,3
2 year	0,9	357633922,6
3 year	0,1	393397314,9
4 year	0,1	432737046,4
5 year	0,1	476010751

476010751

4.3.4. Conclusion

Based on the Estimation and 5-year Projection the total amount of bytes needed to store our database in the long run would be 480MB. So for us to host our database it would be essential for us to ensure that our package or plan would be able to support the size of our database while at the same time ensuring that the devices used to run our system will have the minimum amount of space to be able to store

local temporary data for at least a year. Therefore, they will need roughly 100MB of space on the remote devices.

4.4. Conclusion

After designing our ERD we now have a much better understanding of the system overall and its technical aspects.

5. System Interface Design

5.1. Introduction

This section will be showing all the prototype screens and all their corresponding specifications and description. The Report Policy is also included to illustrate and show the matrix of the Design Principles adhered to in the Output Design. Other interfaces and their technical descriptions are also indicated in this section, in attempts to give a full picture of what interfaces the user will use to interact with the system.

5.2. Input and Output Design

Please find all Images associated with Input and Output design in Appendix B.

5.3. Conclusion

At the end of all this, the user should now know what to expect in the final product and ultimately how they will interact with the system at every given step. This will also assist and make the work easier for us as we transition to the actual development of the product.

6. Package Design Specifications

6.1. Test Specifications

6.1.1. Introduction

The purpose of a Test Plan is to test the system by providing it with a variety of test data and ensuring that the system can handle the data which the user inputs in such a way that the data entered into the system is accurate and consistent.

6.1.2. Test Plan

The Bikes4ERP program is mostly web-based and below we have constructed a test plan to conduct a variety of tests to ensure that the data entered into the system is of the required accuracy and consistency. We have selected 6 core use cases which will each have their Test scenarios which can be broken down into their test steps/procedures which will contain the input, output data as well as the expected results. This will allow us to perform an in-depth analysis of any vulnerabilities that may occur in our system and allow us to catch these problems earlier on in the project development stage.

The 6 core use cases we chose are:

1. Add Bicycle
2. Assign Bicycle
3. Report Repair Job
4. Check-In
5. View Jobs
6. Complete Job

6.1.3. Test Scenarios and Test Data: Add Bicycle

Test Scenario:	1. Add Bicycle
Tester Name:	
Tester Qualifications:	
Scenario Description:	This requirement describes the process of a bicycle being added to the system by a system user. The user initiates the process by capturing all the bicycle details and adding the bicycle.
Scenario Pre-Condition(s):	- The Bicycle is not already available in the system. - The Bicycle has a unique QR Code assigned to it.

Test Case:	Test response on Adding a bicycle to the system.

Test Step/Procedure:	1.1. Click '+ icon' Button to add a new bicycle.	Pass	Fail												
Correct Test Data:	None														
Partially Correct Test Data:	None														
Incorrect Test Data:	None														
Expected Results:	- The system displays the "Bicycle" screen. - Screen will display the following: <table><tr><th>Text/Label</th><th>Control Type</th></tr><tr><td>Create Bicycle</td><td>Button</td></tr><tr><td>Cancel</td><td>Button</td></tr><tr><td>Date Added</td><td>Date Picker</td></tr><tr><td>Bicycle Make</td><td>Text box</td></tr><tr><td>QR Code</td><td>Link</td></tr></table>	Text/Label	Control Type	Create Bicycle	Button	Cancel	Button	Date Added	Date Picker	Bicycle Make	Text box	QR Code	Link		
Text/Label	Control Type														
Create Bicycle	Button														
Cancel	Button														
Date Added	Date Picker														
Bicycle Make	Text box														
QR Code	Link														
Actual Results:															
Overall Outcome:															

Test Step/Procedure:	1.2. Provide "Bicycle Make", "Date Added" and "QR code Link" then click "Create Bicycle" Button	Pass	Fail
Correct Test Data:	- Bicycle Make: Qhubeka - Date Added: 12/06/2018 - QR Code: Link...		

Partially Correct Test Data:	- Bicycle Make: M - Date Added: 10/03/2020 - QR Code: Link...		
Incorrect Test Data:	- Bicycle Make: #NoBrandName23 - Date Added: 15/12/2024		
Expected Results:	The system displays a confirmation modal with the message <i>"Are you sure you want to add this Bicycle"</i> with "Yes" and "No" Buttons.		
Actual Results:			
Overall Outcome:			

Test Step/Procedure:	1.3. Click "Yes" Button	Pass	Fail														
Correct Test Data:	None																
Partially Correct Test Data:	None																
Incorrect Test Data:	None																
Expected Results:	System Creates a new "Bicycle" entry in the database. <table><tr><th>Bicycle</th><th>Attribute</th></tr><tr><td>BicycleID</td><td>10123</td></tr><tr><td>BicycleMake</td><td>Qhubeka</td></tr><tr><td>DateAdded</td><td>12/06/2018</td></tr><tr><td>QR Code</td><td>Link...</td></tr></table> <table><tr><th>BicycleStatus</th><th>Attribute</th></tr><tr><td>BicycleStatus</td><td>Available</td></tr></table> And displays a success modal with the message "Bicycle has been added successfully." and an "OK" Button.	Bicycle	Attribute	BicycleID	10123	BicycleMake	Qhubeka	DateAdded	12/06/2018	QR Code	Link...	BicycleStatus	Attribute	BicycleStatus	Available		
Bicycle	Attribute																
BicycleID	10123																
BicycleMake	Qhubeka																
DateAdded	12/06/2018																
QR Code	Link...																
BicycleStatus	Attribute																
BicycleStatus	Available																
Actual Results:																	
Overall Outcome:																	

Test Step/Procedure:	1.4. Click the "OK" Button	Pass	Fail

Correct Test Data:	None																
Partially Correct Test Data:	None																
Incorrect Test Data:	None																
Expected Results:	The system closes the "Bicycle" Screen and displays the "Bicycle Management" screen. With the following controls: <table><tr><th>Text/Label</th><th>Control Type</th></tr><tr><td>Navigation</td><td>Navigation-bar</td></tr><tr><td>" + icon"</td><td>Button</td></tr><tr><td>Search</td><td>Text-Box</td></tr><tr><td>Search</td><td>Button</td></tr><tr><td>Bicycle Management</td><td>Table</td></tr><tr><td>"Pen icon"</td><td>Button</td></tr></table>	Text/Label	Control Type	Navigation	Navigation-bar	" + icon"	Button	Search	Text-Box	Search	Button	Bicycle Management	Table	"Pen icon"	Button		
Text/Label	Control Type																
Navigation	Navigation-bar																
" + icon"	Button																
Search	Text-Box																
Search	Button																
Bicycle Management	Table																
"Pen icon"	Button																
Actual Results:																	
Overall Outcome:																	

Test Step/Procedure:	1.5. Click "Yes" Button [alt]	Pass	Fail
Correct Test Data:	None		
Partially Correct Test Data:	None		
Incorrect Test Data:	None		
Expected Results:	System displays an error modal with the message "<i>Bicycle already exists. Please try again.</i>" and an "OK" Button.		
Actual Results:			
Overall Outcome:			

Test Step/Procedure:	1.6. Click "Yes" Button [alt]	Pass	Fail
Correct Test Data:	None		
Partially Correct Test Data:	None		

Incorrect Test Data:	None		
Expected Results:	System displays an error modal with the message "An error has occurred, please try again." and "OK" Button.		
Actual Results:			
Overall Outcome:			

Test Step/Procedure:	1.7. Click “No” Button [alt]	Pass	Fail														
Correct Test Data:	None																
Partially Correct Test Data:	None																
Incorrect Test Data:	None																
Expected Results:	System Cancels the action of adding a new bicycle and loads the Bicycle Management screen. With the following controls: <table><tr><th>Text/Label</th><th>Control Type</th></tr><tr><td>Navigation</td><td>Navigation-bar</td></tr><tr><td>“+ icon”</td><td>Button</td></tr><tr><td>Search</td><td>Text-Box</td></tr><tr><td>Search</td><td>Button</td></tr><tr><td>Bicycle Management</td><td>Table</td></tr><tr><td>“Pen icon”</td><td>Button</td></tr></table>	Text/Label	Control Type	Navigation	Navigation-bar	“+ icon”	Button	Search	Text-Box	Search	Button	Bicycle Management	Table	“Pen icon”	Button		
Text/Label	Control Type																
Navigation	Navigation-bar																
“+ icon”	Button																
Search	Text-Box																
Search	Button																
Bicycle Management	Table																
“Pen icon”	Button																
Actual Results:																	
Overall Outcome:																	

6.1.4. Test Scenarios and Test Data: Assign a Bicycle to Student

Test Scenario:	2. Assign Bicycle to Student
Tester Name:	
Tester Qualifications:	
Scenario Description:	This requirement describes the process of an available bicycle being assigned to a student by a system user. The system user initiates the process by selecting to assign a bicycle to a student without a bicycle assigned to them and completing the assignment.
Scenario Pre-Condition(s):	- Student doesn't have a Bicycle assigned to them. - Bicycle status is "Available". - Bicycle is in working condition.

Test Case:	Test response on assigning a bicycle to a student.

Test Step/Procedure:	2.1 Click 'Assign' Button on the available Bicycle	Pass	Fail
Correct Test Data:	None		
Partially Correct Test Data:	None		
Incorrect Test Data:	None		
Expected Results:	The system displays the "Assign Bicycle to Student" screen.		
Actual Results:			
Overall Outcome:			

Test Step/Procedure:	2.2 Select Student to Assign	Pass	Fail
Correct Test Data:	None		
Partially Correct Test Data:	None		
Incorrect Test Data:	Student already has a bicycle assigned to them		
Expected Results:	System updates the following in the database:		
	<table><tr><td>Bicycle</td><td>Attribute</td></tr></table>		
Bicycle	Attribute		

	BicycleID	10123		
	BicycleStatus	1 ("Assigned")		
	BicycleStudent	Attribute		
	BicycleStudentID	56		
	BicycleID	10123		
	StudentID	568		
	DateAssigned	02/07/2020		
	System displays a success modal with the message "Bicycle has been assigned successfully" an "OK" Button.			
Actual Results:	•			
Overall Outcome:				

Test Step/Procedure:	2.3 Click the “OK” Button	Pass	Fail														
Correct Test Data:	None																
Partially Correct Test Data:	None																
Incorrect Test Data:	None																
Expected Results:	The system closes the “Assign Bicycle to Student” Screen and displays the “Bicycle Management” screen. With the following controls: <table> <tr> <th>Text/Label</th> <th>Control Type</th> </tr> <tr> <td>Navigation</td> <td>Navigation-bar</td> </tr> <tr> <td>“+ icon”</td> <td>Button</td> </tr> <tr> <td>Search</td> <td>Text-Box</td> </tr> <tr> <td>Search</td> <td>Button</td> </tr> <tr> <td>Bicycle Management</td> <td>Table</td> </tr> <tr> <td>“Pen icon”</td> <td>Button</td> </tr> </table>	Text/Label	Control Type	Navigation	Navigation-bar	“+ icon”	Button	Search	Text-Box	Search	Button	Bicycle Management	Table	“Pen icon”	Button		
Text/Label	Control Type																
Navigation	Navigation-bar																
“+ icon”	Button																
Search	Text-Box																
Search	Button																
Bicycle Management	Table																
“Pen icon”	Button																
Actual Results:																	

Overall Outcome:		

Test Step/Procedure:	2.4 Click the "OK" Button [alt]	Pass	Fail
Correct Test Data:	None		
Partially Correct Test Data:	None		
Incorrect Test Data:	None		
Expected Results:	System displays an error modal with the message <i>"An error has occurred, please try again."</i> and an "OK" Button.		
Actual Results:			
Overall Outcome:			

Test Step/Procedure:	2.5 Click the "OK" Button [alt]	Pass	Fail
Correct Test Data:	None		
Partially Correct Test Data:	None		
Incorrect Test Data:	None		
Expected Results:	The system refreshes the "Assign Bicycle to Student" screen.		
Actual Results:			
Overall Outcome:			

6.1.5. Test Scenarios and Test Data: Report Repair Job

Test Scenario:	3. Report Repair Job
Tester Name:	
Tester Qualifications:	
Scenario Description:	This requirement describes the process of a bicycle fault being reported by a student. The student initiates the process by scanning the Bicycle QR Code and selecting the Bicycle Section and Part(s) on the screen that need to be fixed. After capturing all that, the next step is selecting the possible faults for the parts. After completing that selection, they submit the job. The scheduled date and time when they should bring their Bicycle for repair is displayed.
Scenario Pre-Condition(s):	• Student has Bicycle assigned to them. • The bicycle is present with the rightful student.

Test Case:	Test response on reporting a Repair Job for a student's bicycle.

Test Step/Procedure:	3.1 Click 'Report Repair Job' Navigation Bar option	Pass	Fail
Correct Test Data:	• None		
Partially Correct Test Data:	• None		
Incorrect Test Data:	• None		
Expected Results:	• The system displays the "Scan Bicycle QR Code" screen.		
Actual Results:	•		
Overall Outcome:			

Test Step/Procedure:	3.2 Scan 'Bicycle QR Code' using a required device.	Pass	Fail
Correct Test Data:	• Valid Bicycle QR Code scan		
Partially Correct Test Data:	• None		
Incorrect Test Data:	• Scan a picture of something that is not a QR Code • Scan a QR Code that is not recognized in the system		

Expected Results:	- System displays the "Full Bicycle Blueprint" screen - The Sections on the blueprint must be selectable.		
Actual Results:			
Overall Outcome:			

Test Step/Procedure:	3.3 Click on the Bicycle Section with the faulty part.	Pass	Fail
Correct Test Data:	Select the Front Wheel		
Partially Correct Test Data:	None		
Incorrect Test Data:	No Section selected		
Expected Results:	System displays the Parts in the selected Section		
Actual Results:			
Overall Outcome:			

Test Step/Procedure:	3.4 Click on the "Unknown" Button. [alt]	Pass	Fail
Correct Test Data:	None		
Partially Correct Test Data:	None		
Incorrect Test Data:	None		
Expected Results:	The system displays an input field where students can capture details of what is wrong if they can't tell which section is faulty.		
Actual Results:			
Overall Outcome:			

Test Step/Procedure:	3.5 Click on the "Submit" button. [alt]	Pass	Fail
Correct Test Data:	Sufficient valid text captured.		
Partially Correct Test Data:	Valid data but not sufficiently descriptive of the problem.		

Incorrect Test Data:	No Text captured in the input field		
Expected Results:	The system displays the Fault list indicating that the Section and Part with the Fault are "Unknown".		
Actual Results:			
Overall Outcome:			

Test Step/Procedure:	3.6 Click on the faulty Bicycle Part.	Pass	Fail
Correct Test Data:	Select the Spokes		
Partially Correct Test Data:	None		
Incorrect Test Data:	No Part selected		
Expected Results:	The system displays the Fault list for the selected Bicycle Part.		
Actual Results:			
Overall Outcome:			

Test Step/Procedure:	3.7 Click on the "Unknown" Button. [alt]	Pass	Fail
Correct Test Data:	None		
Partially Correct Test Data:	None		
Incorrect Test Data:	None		
Expected Results:	The system displays an input field where students can capture details of what is wrong if they can't tell which part in the section is faulty.		
Actual Results:			
Overall Outcome:			

Test Step/Procedure:	3.8 Click on the "Submit" button. [alt]	Pass	Fail
Correct Test Data:	Sufficient valid text captured.		

Partially Correct Test Data:	Valid data but not sufficiently descriptive of the problem.		
Incorrect Test Data:	No Text captured in the input field		
Expected Results:	The system displays the Fault list indicating the Section and that the Part with the Fault is "Unknown".		
Actual Results:			
Overall Outcome:			

Test Step/Procedure:	3.9 Click on the Fault that applies to the selected Bicycle Part.	Pass	Fail										
Correct Test Data:	Select the “Bent Spokes”												
Partially Correct Test Data:	None												
Incorrect Test Data:	No-Fault is selected												
Expected Results:	System displays the Fault List screen with the following controls and a summary of faults reported in this job:												
	<table><tr><th>Text/Label</th><th>Control Type</th></tr><tr><td>Job Faults</td><td>Table</td></tr><tr><td>Add Fault</td><td>Button</td></tr><tr><td>Finalize Job</td><td>Button</td></tr><tr><td>Delete Fault</td><td>Icon Link</td></tr></table>			Text/Label	Control Type	Job Faults	Table	Add Fault	Button	Finalize Job	Button	Delete Fault	Icon Link
	Text/Label			Control Type									
	Job Faults			Table									
	Add Fault			Button									
	Finalize Job			Button									
Delete Fault	Icon Link												
Actual Results:													
Overall Outcome:													

Test Step/Procedure:	3.10 Click on the "Unknown" Button. [alt]	Pass	Fail
Correct Test Data:	None		
Partially Correct Test Data:	None		
Incorrect Test Data:	None		
Expected Results:	The system displays an input field where students can capture details of what is wrong		

	if they can't tell what the exact fault is with the part.		
Actual Results:	•		
Overall Outcome:			

Test Step/Procedure:	3.11 Click on the "Submit" button. [alt]	Pass	Fail
Correct Test Data:	• Sufficient valid text captured.		
Partially Correct Test Data:	• Valid data but not sufficiently descriptive of the problem.		
Incorrect Test Data:	• No Text captured in the input field		
Expected Results:	• The system displays the Fault list indicating the Section and that the Part whose Fault is "Unknown".		
Actual Results:	•		
Overall Outcome:			

Test Step/Procedure:	3.12 Click “Finalize Job” button	Pass	Fail														
Correct Test Data:	None																
Partially Correct Test Data:	None																
Incorrect Test Data:	None																
Expected Results:	System adds the following details to the database:																
	<table><tr><th>JobFault</th><th>Attribute</th></tr><tr><td>ReportLineID</td><td>1</td></tr><tr><td>PartFaultID</td><td>3</td></tr><tr><td>SectionPartID</td><td>4</td></tr><tr><td>FaultReportID</td><td>5</td></tr><tr><td>ReportLineStatusID</td><td>6</td></tr><tr><td>Notes</td><td></td></tr></table>	JobFault	Attribute	ReportLineID	1	PartFaultID	3	SectionPartID	4	FaultReportID	5	ReportLineStatusID	6	Notes			
	JobFault	Attribute															
	ReportLineID	1															
	PartFaultID	3															
	SectionPartID	4															
	FaultReportID	5															
	ReportLineStatusID	6															
	Notes																
<table><tr><th>RepairJob</th><th>Attribute</th></tr></table>	RepairJob	Attribute															
RepairJob	Attribute																

	JobID	1		
	BicycleID	3		
	StudentID	4		
	RepairJobStatusID	5		
	DateReported	06/05/2020		
	ScheduledDate	22/05/2020		
	The system displays a success modal with the message <i>"Repair Job has been Successfully reported"</i> with an "Okay" button. The system also displays the Date of the Scheduled Repair Job appointment.			
Actual Results:				
Overall Outcome:				

Test Step/Procedure:	3.13 Click “Delete Fault” button [alt]	Pass	Fail																						
Correct Test Data:	None																								
Partially Correct Test Data:	None																								
Incorrect Test Data:	None																								
Expected Results:	System removes the following entry from the current Job:																								
	<table><tr><th>JobFault</th><th>Attribute</th></tr><tr><td>ReportLineID</td><td>1</td></tr><tr><td>PartFaultID</td><td>3</td></tr><tr><td>SectionPartID</td><td>4</td></tr><tr><td>FaultReportID</td><td>5</td></tr><tr><td>ReportLineStatusID</td><td>6</td></tr><tr><td>Notes</td><td>None</td></tr><tr><th>RepairJob</th><th>Attribute</th></tr><tr><td>JobID</td><td>1</td></tr><tr><td>BicycleID</td><td>3</td></tr><tr><td>StudentID</td><td>4</td></tr></table>			JobFault	Attribute	ReportLineID	1	PartFaultID	3	SectionPartID	4	FaultReportID	5	ReportLineStatusID	6	Notes	None	RepairJob	Attribute	JobID	1	BicycleID	3	StudentID	4
	JobFault			Attribute																					
	ReportLineID			1																					
	PartFaultID			3																					
	SectionPartID			4																					
	FaultReportID			5																					
	ReportLineStatusID			6																					
	Notes			None																					
	RepairJob			Attribute																					
	JobID			1																					
	BicycleID			3																					
	StudentID			4																					

	RepairJobStatusID	5		
	DateReported	06/05/2020		
	ScheduledDate	22/05/2020		
	The system displays a success modal with the message <i>"Repair Job has been Successfully reported"</i> with an "Okay" button. The system also displays the Date of the Scheduled Repair Job appointment.			
Actual Results:				
Overall Outcome:				

Test Step/Procedure:	3.14 Click "Add Fault" button [alt]	Pass	Fail
Correct Test Data:	None		
Partially Correct Test Data:	None		
Incorrect Test Data:	None		
Expected Results:	- System displays the "Full Bicycle Blueprint" screen - The Sections on the blueprint must be selectable.		
Actual Results:			
Overall Outcome:			

6.1.6. Test Scenarios and Test Data: Check-in Bicycle

Test Scenario:	4. Check-in bicycle
Tester Name:	
Tester Qualifications:	
Scenario Description:	This requirement describes the process of a bicycle being checked in on the day it is due for either a maintenance or a repair. The Student will take their bike to the mechanic which will in turn scan the QR code on the bike to check the bike in.
Scenario Pre-Condition(s):	- Student had previously successfully booked for a maintenance/repair - The mechanic is logged in

Test Case:	Test response on checking in a bicycle

Test Step/Procedure:	4.1 Click 'Check-in bicycle' button	Pass	Fail
Correct Test Data:	None		
Partially Correct Test Data:	None		
Incorrect Test Data:	None		
Expected Results:	The system displays the "Scan Bicycle QR Code" screen.		
Actual Results:			
Overall Outcome:			

Test Step/Procedure:	4.2 Scan 'Bicycle QR Code' using a required device.	Pass	Fail
Correct Test Data:	None		
Partially Correct Test Data:	None		
Incorrect Test Data:	- Scan a picture of something that is not a QR Code - Scan a QR Code that is not recognized in the system		
Expected Results:	System updates the bicycle's status to checked-in		
JobFault Attribute			

	CheckedInStatus True A success message is shown to inform the mechanic that the bicycle was successfully checked in		
Actual Results:			
Overall Outcome:			
Test Step/Procedure:	4.3 Click the "Ok" button	Pass	Fail
Correct Test Data:	None		
Partially Correct Test Data:	None		
Incorrect Test Data:	None		
Expected Results:	The Mechanic is redirected to the checked-in bicycle's jobs		
Actual Results:			
Overall Outcome:			

6.1.7. Test Scenarios and Test Data: View Jobs

Test Scenario:	5. View Jobs
Tester Name:	
Tester Qualifications:	
Scenario Description:	This requirement describes the process of a Mechanic accessing his calendar to view jobs by clicking on the "View Calendar" option in the navigation bar. By clicking on a specific date in the calendar he will be able to see the Jobs he has for that day and he can click on a specific job for even more details.
Scenario Pre-Condition(s):	The Mechanic is logged in

Test Case:	Test response on viewing Mechanic Jobs

Test Step/Procedure:	5.1 Click 'View Calendar' Navigation Bar option	Pass	Fail
Correct Test Data:	None		
Partially Correct Test Data:	None		
Incorrect Test Data:	None		
Expected Results:	The system displays the "Mechanic's Calendar" screen.		
Actual Results:			
Overall Outcome:			

Test Step/Procedure:	5.2 The mechanic clicks on a specific date on the calendar.	Pass	Fail
Correct Test Data:	None		
Partially Correct Test Data:	The mechanic clicks on the incorrect date		
Incorrect Test Data:	None		
Expected Results:	The System displays the "View Jobs" page with a list of all the jobs for the date selected		
Actual Results:			

Overall Outcome:		

Test Step/Procedure:	5.3 Click on a specific job	Pass	Fail
Correct Test Data:	None		
Partially Correct Test Data:	None		
Incorrect Test Data:	None		
Expected Results:	The Mechanic will see more details of that specific job when he clicks on the item		
Actual Results:			
Overall Outcome:			

6.1.8. Test Scenarios and Test Data: Complete Job

Test Scenario:	6. Complete Job (actually completing task)
Tester Name:	
Tester Qualifications:	
Scenario Description:	This requirement describes the process of the Mechanic Completing each task within a job and once that is done, he can mark the job as being completed
Scenario Pre-Condition(s):	- The Job has incomplete tasks - The mechanic is logged in

Test Case:	Test response on completing a job

Test Step/Procedure:	6.1 Click 'View Job' for a specific job	Pass	Fail
Correct Test Data:	None		
Partially Correct Test Data:	None		
Incorrect Test Data:	None		
Expected Results:	The system displays the "View Job" screen.		
Actual Results:			
Overall Outcome:			

Test Step/Procedure:	6.2 Check all the checkboxes of tools that he used	Pass	Fail								
Correct Test Data:	None										
Partially Correct Test Data:	Mechanic checked the incorrect boxes										
Incorrect Test Data:	Mechanic didn't select any of the boxes										
Expected Results:	PartFault is updated with the selected parts that the mechanic used: <table><tr><th>PartFault</th><th>Attribute</th></tr><tr><td>PartFaultID</td><td>12</td></tr><tr><td>PartID</td><td>2</td></tr><tr><td>FaultTypeID</td><td>4</td></tr></table>	PartFault	Attribute	PartFaultID	12	PartID	2	FaultTypeID	4		
PartFault	Attribute										
PartFaultID	12										
PartID	2										
FaultTypeID	4										

	The JobFault’s status is updated to complete: Lastly, if this was the last task in the job that needed to be completed then the “Complete Job” button on the “View Job” screen will be enabled. <table><tr><td>JobFault</td><td>Attribute</td></tr><tr><td>Status</td><td>Complete</td></tr></table>	JobFault	Attribute	Status	Complete		
	JobFault	Attribute					
	Status	Complete					
Actual Results:	•						
Overall Outcome:							

Test Step/Procedure:	6.3 Click on "Complete Job" button	Pass	Fail
Correct Test Data:	None		
Partially Correct Test Data:	None		
Incorrect Test Data:	None		
Expected Results:	Repair Job has its following updated: RepairJob Attribute RepairJobStatus Complete		
Actual Results:			
Overall Outcome:			

6.1.9. Conclusion

With the above documentation, it is now a lot clearer how we intend to test our system using the stipulated and detailed test plans above.

6.2. Hardware and Software Requirements

6.2.1. Introduction

The following tables give a detailed overview of the Hardware and Software requirements that will be required for our system to run at full efficiency and effectiveness.

6.2.2. Hardware Requirements

Hardware	Client-Side		Server Side	
	Minimum Requirements	Recommended Requirements	Minimum Requirements	Recommended Requirements
Input Devices	RFID Antenna RFID Tag QR Scanner Keyboard Mouse	RFID Antenna RFID Tag QR Scanner Keyboard Mouse	Keyboard Mouse	Keyboard Mouse
Output Devices	VGA Display Monitor	VGA Display Monitor Printer	VGA Display Monitor	VGA Display Monitor Printer
CPU	1GHz 32-bit (x86) Processor	Intel Core i3 2.4GHz 32-bit(x86) Processor	1GHz 32-bit (x86) Processor	Intel Core i3 2.4GHz 32-bit(x86) Processor
HDD(Storage)	100	200+	100	200+
RAM	2GB	4GB	2GB	4GB
Graphics	UHD 630 @ 1.10GHz	Intel Integrated graphics	UHD 630 @ 1.10GHz	Intel Integrated graphics
Networking Hardware	10Mbps Built-In NIC	100/1000Mbps Built-In NIC	10MbpsBuilt-In NIC	100/1000Mbps Built-In NIC
Internet Connection	1mb/s – 4mb/s Upload and Download speed	4mb/s – 10mb/s Upload and Download speed	1mb/s – 4mb/s Upload and Download speed	4mb/s – 10mb/s Upload and Download speed

6.2.3. Software Requirements

Software	Client-Side		Server Side	
	Minimum Requirements	Recommended Requirements	Minimum Requirements	Recommended Requirements
Operating System	Microsoft Windows 7	Microsoft Windows 10	Microsoft Windows 7	Microsoft Windows 10
Browser	Microsoft Edge/Google Chrome	Google Chrome	Microsoft Edge/Google Chrome	Google Chrome
Email Provider	Gmail	Gmail		
Database Management System			Cloud Firebase	Cloud Firebase
Web Server			Microsoft IIS	Microsoft IIS
PDF Viewer	Google Chrome/ Microsoft Edge	Google Chrome	Google Chrome/ Microsoft Edge	Google Chrome
Development Environment		Visual Studio Code		Visual Studio Code

6.2.4. Conclusion

With that information, a clearer picture of the Hardware and Software requirements needed for the VVs system to operate at full capacity has been provided.

6.3. Network / Web Layout Specifications

6.3.1. Introduction

The following diagram gives an overview of the intended layout of our system, which includes various components such as servers, firewalls, routers, and workstations.

6.3.2. Diagram

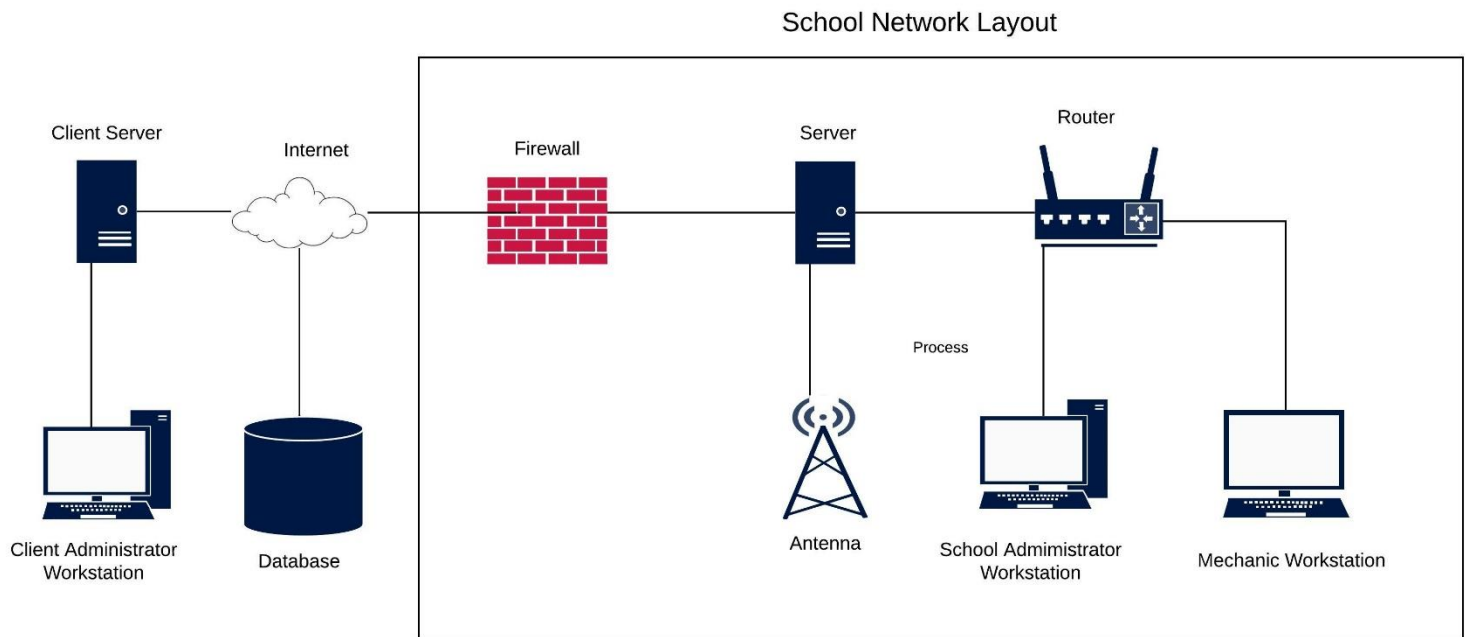


Figure 15 Network Layout Specification

6.3.3. Conclusion

Through the following diagram, you can understand and visualize how we intend to set up the layout of our system physically.

6.4. Validation

6.4.1. Introduction

The validation lists every requirement in our system. It shows which use cases, processes on the primitive level of the Data Flow Diagrams, entities in the Entity Relational Diagram satisfy the requirements individually to detail. In short, the table below validates the need and usage of each Requirement per subsystem.

6.4.2. Validation for each Data Flow

Sub System	Use Case	Process (DFD)	Entities	Screen
1. User	1.1 Login	1.1.1 Display User Login Screen	None	Login
		1.1.2 Capture and check User login details have been filled	None	Login
		1.1.3 Verify User Login details	User UserType	Login
		1.1.4 Display Homepage Screen	AuditTrail	Home
	1.2 Logout	1.2.1 Display Logout confirmation pop-up	None	Navigation Bar
		1.2.2 Display Login Screen details	AuditTrail	Login
	1.3 Forgot Password	1.3.1 Display Forgot Password Screen	None	Forgot Password
		1.3.2 Capture and Validate User Email Address fields are filled	None	Forgot Password
		1.3.3 Verify inputted Email Address details	User	Forgot Password
		1.3.4 Send Reset password link	None	Forgot Password
		1.3.5 Display Reset password Screen	None	Reset Password
		1.3.6 Capture and Verify new User password details	None	Reset Password
		1.3.7 Update Password. Display Reset password Successful notification	User AuditTrail	Reset Password
	1.4 Add User	1.4.1 Display User Management Screen	User UserType	User Management
		1.4.2 Display Add User Screen	UserType School AdminSchool	Create User

Sub System	Use Case	Process (DFD)	Entities	Screen
		1.4.3 Capture User details and check that all fields are filled in and an option selected in Combo box	None	Create User
		1.4.4 Validate User details	None	Create User
		1.4.5 Verify User details	User	Create User
		1.4.6 Generate User ID and Save User details	User AdminSchool	Create User Create User
		1.4.7 Display Create Successful Notification	AuditTrail	Create User
	1.5 Search User	1.5.1 Display User management Screen	User UserType	User Management
		1.5.2 Capture User Search criteria details	None	User Management
		1.5.3 Display Searched User Results	User UserType	User Management
		1.5.4 Display Searched User details Screen	User UserType AuditTrail	View User
	1.6 Update User	1.6.1 Display Searched User Screen	School AdminSchool	View User
		1.6.2 Capture User details and check for empty fields and for unselected combo box	None	View User
		1.6.3 Validate updated User details	None	View User
		1.6.4 Save updated User details	User AdminSchool	View User
		1.6.5 Display Update Successful Notification	AuditTrail	View User
	1.7 Delete User	1.7.1 Display Searched User Screen details	None	View User
		1.7.2 Delete Searched User details	AuditTrail User	View User
		1.7.3 Display User Deleted successfully notification	AuditTrail	View User

Sub System	Use Case	Process (DFD)	Entities	Screen
	1.8 Add User Type	1.8.1 Display User Type Management Screen	UserType	User Type Management
		1.8.2 Display Add User Type Screen	None	Create User Type
		1.8.3 Capture User Type details and check that all fields are filled in and an option selected in Combo box	None	Create User Type
		1.8.4 Validate User Type details	None	Create User Type
		1.8.5 Verify User Type Details	UserType	Create User Type
		1.8.6 Generate User Type ID and Save User Type details	UserType	Create User Type
		1.8.7 Display Create Successful Notification	AuditTrail	Create User Type
	1.9 Search User Type	1.9.1 Display User Type Management Screen	UserType	User Type Management
		1.9.2 Capture User Type Search criteria details	None	User Type Management
		1.9.3 Display Searched User Type Results	UserType	User Type Management
		1.9.4 Display Searched User Type details Screen	UserType AuditTrail	View User Type
	1.10 Update User Type	1.10.1 Display Searched User Type Screen	None	View User Type
		1.10.2 Capture User Type details and check for empty fields and for unselected combo box	None	View User Type
		1.10.3 Validate updated User Type details	UserType	View User Type
		1.10.4 Save updated User Type details	UserType	View User Type
		1.10.5 Display Update Successful Notification	AuditTrail	View User Type
	1.11 Delete User Type	1.11.1 Display Searched User Type Screen	None	View User Type

Sub System	Use Case	Process (DFD)	Entities	Screen
		1.11.2 Delete Searched User Type details	UserType	View User Type
		1.11.3 Display User Type Deleted successfully notification	AuditTrail	View User Type
2. Mechanic Schedule	2.1 View Job	2.1.1 Display Job Screen	Check-In MaintenanceJob RepairJob Check-InStatus AuditTrail	Job
	2.2 View Job List	2.2.1 Display Job List Screen	FaultType JobFault Section Part JobFaultStatus AuditTrail	Job List
3. Tracking Hardware	3.1 Add Antenna	3.1.1 Display Antenna Management Screen	Antenna AntennaStatus	Antenna management
		3.1.2 Display Add Antenna Screen	None	Create Antenna
		3.1.3 Capture Antenna details and check that all fields are filled in and an option selected in Combo box	None	Create Antenna
		3.1.4 Validate Antenna details	None	Create Antenna
		3.1.5 Verify Antenna details	Antenna	Create Antenna
		3.1.6 Generate Antenna ID and Save Antenna details	Antenna	Create Antenna
		3.1.7 Display Create Successful Notification	AuditTrail	Create Antenna
	3.2 Search Antenna	3.2.1 Display Antenna management Screen	Antenna AntennaStatus	Antenna management
		3.2.2 Capture Antenna Search criteria details	None	Antenna management
		3.2.3 Display Searched Antenna Results	Antenna AntennaStatus	Antenna management
		3.2.4 Display Searched Antenna Details Screen	Antenna AntennaStatus AuditTrail	View Antenna

Sub System	Use Case	Process (DFD)	Entities	Screen
	3.3 Update Antenna	3.3.1 Display Searched Antenna screen	None	View Antenna
		3.3.2 Capture Antenna details and check for empty fields and for unselected combo box	None	View Antenna
		3.3.3 Validate updated Antenna details	Antenna	View Antenna
		3.3.4 Save updated Antenna details	Antenna	View Antenna
		3.3.5 Display Update Successful Notification	AuditTrail	View Antenna
	3.4 Delete Antenna	3.4.1 Display Searched Antenna screen	None	View Antenna
		3.4.2 Delete Searched Antenna details	Antenna AuditTrail	View Antenna
		3.4.3 Display Antenna Deleted notification	AuditTrail	View Antenna
	3.5 Add Tag	3.5.1 Display Tag Management Screen	Tag TagStatus	Tag Management
		3.5.2 Display Add Tag screen	None	Create Tag
		3.5.3 Capture Tag details and check for empty fields	None	Create Tag
		3.5.4 Validate Tag details	None	Create Tag
		3.5.5 Verify Tag Details	Tag	Create Tag
		3.5.6 Generate Tag ID and Save Tag details	Tag	Create Tag
		3.5.7 Display Add Tag Successful Notification	AuditTrail	Create Tag
	3.6 Search Tag	3.6.1 Display Tag management screen	None	Tag Management
		3.6.2 Capture Tag Search criteria details	None	Tag Management
		3.6.3 Display Searched Tag results	Tag TagStatus	Tag Management
		3.6.4 Display Searched Tag Screen	Tag TagStatus AuditTrail	View Tag

Sub System	Use Case	Process (DFD)	Entities	Screen
	3.7 Update Tag	3.7.1 Display Searched Tag Screen	None	View Tag
		3.7.2 Capture Updated Tag details and check for empty fields	None	View Tag
		3.7.3 Validate updated Tag details	None	View Tag
		3.7.4 Save updated Tag details	Tag TagStatus	View Tag
		3.7.5 Display Update Successful Notification	AuditTrail	View Tag
	3.8 Delete Tag	3.8.1 Display Searched Tag Screen	Tag	View Tag
		3.8.2 Delete Searched Tag details	Tag	View Tag
		3.8.3 Display Tag Deleted successfully notification	AuditTrail	View Tag
4. Student	4.1 Add Student	4.1.1 Display Student Management Screen	Student	Student Management
		4.1.2 Display Add Student Screen	None	Create Student
		4.1.3 Capture Student details and check that all fields are filled in and an option selected in Combo box	None	Create Student
		4.1.4 Validate Student details	None	Create Student
		4.1.5 Verify Student Details	Student	Create Student
		4.1.6 Generate Student ID and Save Student details	Student	Create Student
		4.1.7 Display Create Successful Notification	AuditTrail	Create Student
	4.2 Search Student	4.2.1 Display Student management Screen	Student	Student Management
		4.2.2 Capture Student Search criteria details	None	Student Management
		4.2.3 Display Searched Student Results	Student	Student Management

Sub System	Use Case	Process (DFD)	Entities	Screen
		4.2.4 Display Searched Student Screen	Student AuditTrail	View Student
	4.3 Update Student	4.3.1 Display Searched Student details	None	View Student
		4.3.2 Capture Student details and check for empty fields and for unselected combo box	None	View Student
		4.3.3 Validate updated Student details	None	View Student
		4.3.4 Save updated Student details	Student	View Student
		4.3.5 Display Update Successful Notification	AuditTrail	View Student
	4.4 Delete Student	4.4.1 Display Searched Student Screen	None	View Student
		4.4.2 Delete Searched Student details	Student	View Student
		4.4.3 Display Student Deleted Successfully Notification	AuditTrail	View Student
	4.5 Capture Student Mark	4.5.1 Display Student Mark management Screen	Student StudentMark	Student Mark Management
		4.5.2 Display Capture Student Mark details	Student StudentMark	Capture Student Mark
		4.5.3 Capture Student Mark details and check that all fields are filled in and an option selected in Combo box	None	Capture Student Mark
		4.5.4 Validate Student Mark details	None	Capture Student Mark
		4.5.5 Generate StudentMarkID and Save Student details	StudentMark	Capture Student Mark
		4.5.6 Display Capture Mark Successful Notification	AuditTrail	Capture Student Mark

Sub System	Use Case	Process (DFD)	Entities	Screen
	4.6 Search Student Marks	4.6.1 Display Student Mark management Screen	Student StudentMark	Student Mark Management
		4.6.2 Capture Student Mark Search criteria details	None	Student Mark Management
		4.6.3 Display Searched Student Mark Results	Student StudentMark AuditTrail	Student Mark Management
	4.7 Add Mark Type	4.7.1 Display Mark Type Management Screen	MarkType	Mark Type Management
		4.7.2 Display Add Mark Type Screen	None	Create Mark Type
		4.7.3 Captures Mark Type details and check that all fields are filled in	None	Create Mark Type
		4.7.4 Validate Mark Type details	None	Create Mark Type
		4.7.5 Verify Mark Type Details	MarkType	Create Mark Type
		4.7.6 Generate MarkTypeID and Save Mark Type details	MarkType	Create Mark Type
		4.7.7 Display Create Mark Type Successful Notification	AuditTrail	Create Mark Type
	4.8 Search Mark Type	4.8.1 Display Mark Type Management Screen	MarkType	Mark Type Management
		4.8.2 Capture Mark Type Search criteria details	None	Mark Type Management
		4.8.3 Display Searched Student Results	MarkType	Mark Type Management
		4.8.4 Display Searched Mark Type Screen	MarkType AuditTrail	View Mark Type
	4.9 Update Mark Type	4.9.1 Display Searched Mark Type Screen	None	View Mark Type
		4.9.2 Capture Mark Type details and check for empty fields	None	View Mark Type
		4.9.3 Validate updated Mark Type details	MarkType	View Mark Type

Sub System	Use Case	Process (DFD)	Entities	Screen
		4.9.4 Save updated Mark Type details	MarkType	View Mark Type
		4.9.5 Display Update Successful Notification	AuditTrail	View Mark Type
	4.10 Delete Mark Type	4.10.1 Display Searched Mark Type details	None	View Mark Type
		4.10.2 Delete Searched Mark Type details	MarkType	View Mark Type
		4.10.3 Display Mark Type Deleted successfully notification	AuditTrail	View Mark Type
5. Bicycle	5.1 Add Bicycle	5.1.1 Display Bicycle Management Screen	Bicycle BicycleStatus	Bicycle Management
		5.1.2 Display Add Bicycle Screen	None	Create Bicycle
		5.1.3 Captured Bicycle details and check that all fields are filled in	None	Create Bicycle
		5.1.4 Validate Bicycle details	None	Create Bicycle
		5.1.5 Verify Bicycle details	Bicycle	Create Bicycle
		5.1.6 Generate BicycleID and Save Bicycle details	Bicycle	Create Bicycle
		5.1.7 Display Create Successful Notification	AuditTrail	Create Bicycle
	5.2 Search Bicycle	5.2.1 Display Bicycle management Screen	Bicycle BicycleStatus	Bicycle Management
		5.2.2 Capture Bicycle Search criteria details	None	Bicycle Management
		5.2.3 Display Searched Bicycle Results	Bicycle BicycleStatus	Bicycle Management
		5.2.4 Display Searched Bicycle Screen	Bicycle BicycleStatus AuditTrail	View Bicycle
	5.3 Update Bicycle	5.3.1 Display Searched Bicycle details	None	View Bicycle
		5.3.2 Capture Bicycle details and check for empty fields	None	View Bicycle

Sub System	Use Case	Process (DFD)	Entities	Screen
		5.3.3 Validate Updated Bicycle details	None	View Bicycle
		5.3.4 Save Updated Bicycle details	Bicycle	View Bicycle
		5.3.5 Display Update Successful Notification	AuditTrail	View Bicycle
	5.4 Delete Bicycle	5.4.1 Display Searched Bicycle details	None	View Bicycle
		5.4.2 Delete Searched Bicycle details	Bicycle	View Bicycle
		5.4.3 Display Bicycle Deleted successfully notification	AuditTrail	View Bicycle
	5.5 Assign A Bicycle To Student	5.5.1 Display Bicycle management Screen	Bicycle Student BicycleStatus	Bicycle Management
		5.5.2 Display Assign Bicycle to Student Screen	Student BicycleStudent	Assign Bicycle to Student
		5.5.3 Capture and Validate Assign Bicycle to student details	None	Assign Bicycle to Student
		5.5.4 Update and Save inputted details	Bicycle BicycleStudent	Assign Bicycle to Student
		5.5.5 Display Assign Successful notification	AuditTrail	Assign Bicycle to Student
	5.6 Unassign Bicycle From Student	5.6.1 Display Bicycle Assignment Screen	Student BicycleStudent	Bicycle Assignment
		5.6.2 Update and Save Unassign Bicycle details	Bicycle BicycleStatus BicycleStudent Student	Bicycle Assignment
		5.6.3 Display Unassign Successful notification	AuditTrail	Bicycle Assignment
	5.7 Assign Tag To Bicycle	5.7.1 Display Tag assignment Screen	Bicycle Tag	Tag Assignment
		5.7.2 Display Assign Tag to Bicycle Screen	Bicycle Tag	Assign Tag to Bicycle
		5.7.3 Capture and Validate Assign Tag to Bicycle details	None	Assign Tag to Bicycle

Sub System	Use Case	Process (DFD)	Entities	Screen
		5.7.4 Update and Save inputted details	Bicycle Tag	Assign Tag to Bicycle
		5.7.5 Display Assign Successful notification	AuditTrail	Assign Tag to Bicycle
	5.8 Unassign Tag from Bicycle	5.8.1 Display Tag Assignment Screen	None	Tag Assignment
		5.8.2 Update and Save Unassign Tag details	Bicycle Tag	Tag Assignment
		5.8.3 Display Unassign Successful notification	AuditTrail	Tag Assignment
6. Bicycle Part	6.1 Add Bicycle Part	6.1.1 Display Bicycle Part Management Screen	Part Section	Bicycle Part Management
		6.1.2 Display Add Bicycle Part Screen	None	Create Bicycle Part
		6.1.3 Capture Bicycle Part details and check that all fields are filled in and an option selected in Combo box	None	Create Bicycle Part
		6.1.4 Validate Bicycle Part details	None	Create Bicycle Part
		6.1.5 Verify Bicycle Part details	Part	Create Bicycle Part
		6.1.6 Generate PartID and Save Bicycle Part details	Part	Create Bicycle Part
		6.1.7 Display Add Successful Notification	AuditTrail	Create Bicycle Part
	6.2 Search Bicycle Part	6.2.1 Display Bicycle Part Management Screen	Part Section	Bicycle Part Management
		6.2.2 Capture Bicycle Part Search criteria details	None	Bicycle Part Management
		6.2.3 Display Searched Bicycle Part Results	Part Section	Bicycle Part Management
		6.2.4 Display Searched Bicycle Part Screen	Part Section AuditTrail	View Bicycle Part
	6.3 Update Bicycle Part	6.3.1 Display Searched Bicycle Part Screen	None	View Bicycle Part
		6.3.2 Capture Bicycle Part details	None	View Bicycle Part

Sub System	Use Case	Process (DFD)	Entities	Screen
		and check for empty fields and for unselected combo box		
		6.3.3 Validate Updated Bicycle Part details	None	View Bicycle Part
		6.3.4 Save Updated Bicycle Part details	Part	View Bicycle Part
		6.3.5 Display Update Successful notification	AuditTrail	View Bicycle Part
	6.4 Delete BicyclePart	6.4.1 Display Searched Bicycle Part details	None	View Bicycle Part
		6.4.2 Delete Searched Bicycle Part details	Part	View Bicycle Part
		6.4.3 Display Bicycle Part deleted successfully notification	AuditTrail	View Bicycle Part
	6.5 Add Bicycle Section	6.5.1 Display Bicycle Section Management Screen	Section	Bicycle Section Management
		6.5.2 Display Add Bicycle Section Screen	None	Create Bicycle Section
		6.5.3 Capture Bicycle Section details and check that all fields are filled in	None	Create Bicycle Section
		6.5.4 Validate Bicycle Section details	None	Create Bicycle Section
		6.5.5 Verify Bicycle Section details	Section	Create Bicycle Section
		6.5.6 Generate SectionID and Save Bicycle Section details	Section	Create Bicycle Section
		6.5.7 Display Add Successful Notification	AuditTrail	Create Bicycle Section
	6.6 Search Bicycle Section	6.6.1 Display Bicycle Section Management Screen	Section	Bicycle Section Management
		6.6.2 Capture Bicycle Section Search criteria details	None	Bicycle Section Management

Sub System	Use Case	Process (DFD)	Entities	Screen
		6.6.3 Display Searched Bicycle Section details	Section	Bicycle Section Management
		6.6.4 Display Searched Bicycle Section Screen	BicycleSection	View Bicycle Section
	6.7 Update Bicycle Section	6.7.1 Display Searched Bicycle Section Screen	None	View Bicycle Section
		6.7.2 Capture Bicycle Section details and check for empty fields	None	View Bicycle Section
		6.7.3 Validate Updated Bicycle Section details	None	View Bicycle Section
		6.7.4 Save Updated Bicycle Section details	Section	View Bicycle Section
		6.7.5 Display Update Successful notification	AuditTrail	View Bicycle Section
	6.8 Delete Bicycle Section	6.8.1 Display Searched Bicycle Section details	None	View Bicycle Section
		6.8.2 Delete Searched Bicycle Section details	Section	View Bicycle Section
		6.8.3 Display Bicycle Section deleted successfully notification	AuditTrail	View Bicycle Section
7. Tracking	7.1 Add Log Entry	7.1.1 Capture tag details	None	None
		7.1.2 [ALT] Tag read error termination	None	None
		7.1.2 Create and Save Log entry details	Log AuditTrail	None
	7.2 Search Log Entry	7.2.1 Display Log management Screen	None	Log Management
		7.2.2 Capture Log entry Search criteria details	None	Log Management
		7.2.3 Display Searched Log Entry Screen	Log AuditTrail	Log Management
	7.3 Add Route	7.3.1 Display Route Management Screen	Route School	Route Management
		7.3.2 Display Add Route Screen	None	Create Route

Sub System	Use Case	Process (DFD)	Entities	Screen
		7.3.3 Capture Route details and check that all fields are filled in and an option selected in Combo box	None	Create Route
		7.3.4 Validate Route details	None	Create Route
		7.3.5 Verify Route Details	Route	Create Route
		7.3.6 Generate RouteID and RouteAntenna and Save the corresponding details	Route RouteAntenna	Create Route
		7.3.7 Display Create Successful Notification	AuditTrail	Create Route
	7.4 Search Route	7.4.1 Display Route management Screen	Route School	Route Management
		7.4.2 Capture Route Search criteria details	None	Route Management
		7.4.3 Display Searched Route Results	Route School	Route Management
		7.4.4 Display Searched Route Screen	Route AuditTrail	View Route
	7.5 Update Route	7.5.1 Display Searched Route details	None	View Route
		7.5.2 Capture Route details and check for empty fields and for unselected combo box	None	View Route
		7.5.3 Validate updated Route details	None	View Route
		7.5.4 Save updated Route details	Route RouteAntenna	View Route

Sub System	Use Case	Process (DFD)	Entities	Screen
		7.5.5 Display Update Successful Notification	AuditTrail	View Route
	7.6 Delete Route	7.6.1 Display Searched Route Screen	None	View Route
		7.6.2 Delete Searched Route details	Route RouteAntenna	View Route
		7.6.3 Display Route Deleted successfully Notification	AuditTrail	View Route
8. Job Tasks	8.1 Add Fault Task	8.1.1 Display Fault Task Management Screen	FaultTask	Fault Task Management
		8.1.2 Display Add Fault Task Screen	None	Create Fault Task
		8.1.3 Capture Fault task details and check for empty fields	None	Create Fault Task
		8.1.4 Validate Fault task details	None	Create Fault Task
		8.1.5 Verify Fault Task details	FaultTask	Create Fault Task
		8.1.6 Generate Fault ID and Save FaultTask details	FaultTask	Create Fault Task
		8.1.7 Display Add Successful Notification	AuditTrail	Create Fault Task
	8.2 Search Fault Task	8.2.1 Display Fault Task Management Screen	FaultTask	Fault Task Management
		8.2.2 Capture Fault Task Search criteria details	None	Fault Task Management
		8.2.3 Display Searched Fault Task Results	FaultTask	Fault Task Management
		8.2.4 Display Fault Task details Screen	FaultTask AuditTrail	View Fault Task
	8.3 Update Fault Task	8.3.1 Display Searched Fault Task screen	None	View Fault Task
		8.3.2 Capture Fault task details and check for empty fields	None	View Fault Task
		8.3.3 Validate Updated Fault Task details	None	View Fault Task

Sub System	Use Case	Process (DFD)	Entities	Screen
		8.3.4 Save updated Fault Task details	FaultTask	View Fault Task
		8.3.5 Display Update Successful Notification	AuditTrail	View Fault Task
	8.4 Delete Fault Task	8.4.1 Display Searched Fault Task screen	None	View Fault Task
		8.4.2 Delete Searched Fault Task details	FaultTask	View Fault Task
		8.4.3 Display Fault Task deleted successfully notification	AuditTrail	View Fault Task
	8.5 Add Maintenance Task	8.5.1 Display Maintenance Task Management Screen	MaintenanceTask	Maintenance Task Management
		8.5.2 Display Add Maintenance Task screen	None	Create Maintenance Task
		8.5.3 Capture Maintenance Task details and check for empty field	None	Create Maintenance Task
		8.5.4 Validate Maintenance task details	None	Create Maintenance Task
		8.5.5 Verify Maintenance Task details	MaintenanceTask	Create Maintenance Task
		8.5.6 Generate MaintenanceTask ID and Save MaintenanceTask details	MaintenanceTask	Create Maintenance Task
		8.5.7 Display Add Successful Notification	AuditTrail	Create Maintenance Task
	8.6 Search Maintenance Task	8.6.1 Display MaintenanceTask Management Screen	MaintenanceTask MaintenanceType	Maintenance Task Management
		8.6.2 Capture Maintenance Task Search criteria details	None	Maintenance Task Management
		8.6.3 Display Searched Maintenance Task Results	MaintenanceTask MaintenanceGroup	Maintenance Task Management
		8.6.4 Display Maintenance Task Details Screen	MaintenanceTask AuditTrail	View Maintenance Task

Sub System	Use Case	Process (DFD)	Entities	Screen
	8.7 Update Maintenance Task	8.7.1 Display Searched Maintenance Task screen	None	View Maintenance Task
		8.7.2 Capture Maintenance task details and check for empty fields	None	View Maintenance Task
		8.7.3 Validate Updated Maintenance Task details	None	View Maintenance Task
		8.7.4 Save Updated Maintenance Task details	MaintenanceTask	View Maintenance Task
		8.7.5 Display Update Successful notification	AuditTrail	View Maintenance Task
	8.8 Delete Maintenance Task	8.8.1 Display Searched Maintenance Task screen	None	View Maintenance Task
		8.8.2 Delete Searched Maintenance Task details	MaintenanceTask	View Maintenance Task
		8.8.3 Display Maintenance Task deleted successfully notification	AuditTrail	View Maintenance Task
9. School	9.1 Add School	9.1.1 9.1.1 Display School Management Screen	School	School Management
		9.1.2 Display Add School Screen	None	Create School
		9.1.3 Capture School details and check that all fields are filled in and an option selected in Combo box	None	Create School
		9.1.4 Validate School details	None	Create School
		9.1.5 Verify School Details	School	Create School
		9.1.6 Generate SchoolID and Save School details	School	Create School
		9.1.7 Display Create School	AuditTrail	Create School

Sub System	Use Case	Process (DFD)	Entities	Screen
		Successful Notification		
	9.2 Search School	9.2.1 Display School management screen	School	School Management School Management
		9.2.2 Capture School Search criteria details	None	School Management
		9.2.3 Display Searched School Results	School	School Management
		9.2.4 Display Searched School Screen	School AuditTrail	School Management
	9.3 Update School	9.3.1 Display Searched School screen	None	View School
		9.3.2 Capture School details and check for empty fields and for unselected combo box	None	View School
		9.3.3 Validate Updated School details	None	View School
		9.3.4 Save updated School details	School	View School
		9.3.5 Display Update Successful Notification	AuditTrail	View School
	9.4 Delete School	9.4.1 Display Searched School screen	None	View School
		9.4.2 Delete Searched School details	School	View School
		9.4.3 Display School deleted successfully Notification	AuditTrail	View School
10. Job	10.1 Report Repair Job	10.1.1 Display QR Scan Screen	None	QR Scan
		10.1.2 Generate RepairJobID and Save Repair Job details	RepairJob	QR Scan

Sub System	Use Case	Process (DFD)	Entities	Screen
		10.1.3 Display Full Bicycle Blueprint Screen	Section	Bicycle Blueprint
		10.1.3 [ALT] Display Unknown Fault Screen	None	Unknown fault
		10.1.4 Display Section Blueprint Screen	Part	Section Blueprint
		10.1.4 [ALT] Display Unknown Fault Screen	None	Unknown fault
		10.1.5 Display Part Faults Screen	FaultType	Part Fault
		10.1.5 [ALT] Display Unknown Fault Screen	None	Unknown fault
		10.1.6 Generate JobFaultID and Save JobFault details	JobFault	Part Fault
		10.1.7 Display Fault List Screen details	FaultType JobFault Part Section	Fault List
		10.1.8 Assign Job and Create Check-In entry	Check-In	Fault List
		10.1.9 Display Fault Reported Successfully notification	AuditTrail	Fault List
	10.2 Complete Job	10.2.1 Display Fix Fault Screen	FaultTask	Fix Fault
		10.2.2 Save and Update ReportFaultTask and ReportLine details	ReportLine ReportFaultTask	Fix Fault
		10.2.3 Check if all Faults are completed	None	Fix Fault
		10.2.4 Display Job Completed successfully notification	AuditTrail	Fix Fault
	10.3 Check-In Bicycle	10.3.1 Receive Student Bicycle	None	None
		10.3.2 Display QR Scan Screen	None	QR Scan

Sub System	Use Case	Process (DFD)	Entities	Screen
		10.3.3 Capture and Retrieve Check-In details	Bicycle RepairJob MaintenanceJob Check-In	QR Scan
		10.3.3A [ALT] Display Bicycle not due error Screen	None	QR Scan
		10.3.3B [ALT] Notify Student of unsuccessful check-In	None	QR Scan
		10.3.4 Update and Save Check-In details	Check-In	QR Scan
		10.3.5 Display Check-in Successful Notification	None	QR Scan
		10.3.6 Notify Student of Successful check-in	AuditTrail	QR Scan
	10.4 Report Maintenance	10.4.1 Generate Next Maintenance date details	MaintenanceList MaintenanceJob MaintenanceType	None
		10.4.2 Generate MaintenanceListID and Save MaintenanceList details	MaintenanceList MaintenanceJob	None
		10.4.3 Update Past Maintenance List Status	MaintenanceList	None
		10.4.4 Assign Job to Mechanic	Check-In AuditTrail	None
11. Reporting	11.1 Generate Mechanic Schedule	11.1.1 Generate and View Mechanic Schedule	BusinessRule MechanicSchedule Student BicycleStudent MaintenanceTask FaultTask	Mechanic Schedule
		11.1.2 Generate Mechanic Schedule Email	User	Mechanic Schedule
		11.2.3 Generate PDF download file	AuditTrail	Mechanic Schedule
	11.2 Generate Mechanic	11.2.1 Generate and View	BusinessRule MechanicSchedule Student	Mechanic Schedule

Sub System	Use Case	Process (DFD)	Entities	Screen
	Schedule Requirements Report	Mechanic Schedule	FaultType BicycleStudent MaintenanceTask FaultTask	
		11.2.2 Generate Mechanic Schedule Requirements Email	User	Mechanic Schedule
		11.2.3 Generate PDF download file	AuditTrail	Mechanic Schedule
	11.3 Generate Student Report	11.3.1 Display Student report Screen	Mark Type	Student Report
		11.3.2 Capture desired criteria and check that an option in the combo box is selected	None	Student Report
		11.3.3 Generate Chart details	Student BicycleStudent Attendance StudentMark	Student Report
		11.3.4 Display Chart and Control breaks details	None	Student Report
		11.3.5 Generate PDF download file	AuditTrail	Student Report
	11.4 Generate Bicycle History Report	11.4.1 Display Bicycle History report Screen	School	Bicycle History Report
		11.4.2 Capture desired criteria and check that an option in the combo box is selected	None	Bicycle History Report
		11.4.3 Generate Report details	Student ScheduledMaintenance RepairJob FaultTask MaintenanceTask MechanicJobStatus	Bicycle History Report
		11.4.4 Display Bicycle History Report details	AuditTrail	Bicycle History Report
		11.4.5 Generate PDF download file	AuditTrail	Bicycle History Report

Sub System	Use Case	Process (DFD)	Entities	Screen
	11.5 Generate Log Report	11.5.1 Display Log report Screen	None	Log Report
		11.5.2 Capture desired criteria and check that a Start and end date are selected	None	Log Report
		11.5.3 Generate Log Report details	Student Bicycle Tag Log	Log Report
		11.5.4 Display Log Report details	None	Log Report
		11.5.5 Generate PDF download file	AuditTrail	Log Report
	11.6 Generate Parts Used Report	11.6.1 Display Parts Used Report Screen	None	Parts Used Report
		11.6.2 Capture desired criteria and check that a Start and end date are selected	None	Parts Used Report
		11.6.3 Generate Parts Used Report details	FaultTask MaintenanceTask RepairJob ScheduledMaintenance Student BicycleStudent JobFaultTask MechanicJobStatus	Parts Used Report
		11.6.4 Display Parts Used Report details	None	Parts Used Report
		11.6.5 Generate PDF download file	AuditTrail	Parts Used Report
12. Notification	12.1 Send Student did not arrive notification	12.1.1 Identify a Student who hasn't arrived at school	Log	None
		12.1.2 Retrieve Student details	Bicycle Student Guardian	None
		12.1.3 Generate and Send Student alert notification SMS	AuditTrail	None
	12.2 Send Student	12.2.1 Retrieve Bicycle due for Maintenance	Student Bicycle BicycleStudent	None

Sub System	Use Case	Process (DFD)	Entities	Screen
	Check-In notification	12.2.2 Generate Maintenance notification details	None	None
		12.2.3 Maintenance notification SMS details	AuditTrail	None
13. Administrator	13.1 Backup Data	13.1.1 Display System Backup and Restore Screen	BackupFile	System Backup and Restore
		13.1.2 Perform system backup	BackupFile	System Backup and Restore
		13.1.3 Display Backup successful notification	AuditTrail	System Backup and Restore
	13.2 Restore Data	13.2.1 Display System Backup and Restore Screen	BackupFile	System Backup and Restore
		13.2.2 Perform System Restore	BackupFile	System Backup and Restore
		13.2.3 Display Restore Successful notification	AuditTrail	System Backup and Restore
	13.3 View Audit Trail Log	13.3.1 Display Audit Trail Management Screen	None	Audit Trail Management
		13.3.2 Display Audit trail Log Screen	AuditTrail	Audit Trail Log
	13.4 Search Audit Trail Entry	13.4.1 Display Audit Trail Management Screen	UserType AuditTrail	Audit Trail Management
		13.4.2 Capture and validate Audit Trail search criteria	None	Audit Trail Management
		13.4.3 Display searched Audit Trail Results	AuditTrail	Audit Trail Management
		13.4.3 Display Audit Trail details	UserType AuditTrail	Audit Trail Entry

6.4.3. Conclusion

That table has now painted a clearer picture and explains the system requirements and their usage at the lowest level of the system to detail. This is important because those Processes are the moving parts to what our system is intended to do.

7. Deliverable Conclusion & Sign-off

7.1. Conclusion

This concludes the second deliverable of the proposed system. This deliverable is purposed to provide a broad overview of the functional specifications of each Functional Requirement, along with descriptions, data models, and data that go along with them. The goal is that now you can fully envision how the core parts of our system will work and what data will be flowing through it.

7.2. Client Sign-off

I, _____, on behalf of EPI-USE, hereby confirm that I understand and agree to the stated scope of work, specifications, diagrams and requirements set out in this document – Visual Visionaries Deliverable 3 and 4 Technical Specification.

Date: _____

Signature: _____

7.3. Group Sign-off

We, as Group 17 Visual Visionaries, hereby agree and declare that each member in the group contributed equally and fully to this document, Deliverable 3 and 4 Technical Specification for INF 370 2020.

Name: _____

Signature: _____

Date: _____

Name: _____

Signature: _____

Date: _____

Name: _____

Signature: _____

Date: _____

Name: _____

Signature: _____

Date: _____

Name: _____

Signature: _____

Date: _____

8. Complexity Matrix

INF370 - Complexity Matrix - 2020				
Topic	Level		Marks	MAX
1. Special GUI	For online applications: Responsive web design		3	42
	For desktop applications: Form design according to design principles (Schneiderman's golden rule on navigation applies here)			
	Appropriate use of grids/tables		3	
	Appropriate use of tabs/links		3	
	Use of graphs in an appropriate business context		4	
	The storage and display of graphical information, like photos with a good business reason		3	
	Working e-mail automatically generated from the database in an appropriate business context		2	
	SMS messages automatically generated from the system in an appropriate business context		2	
	Extensive user-friendly search facility		3	
	At least one use of a tree to display data from the database		3	
	Able to dynamically modify a data tree structure and in doing so adjusting the data in the database		4	
	At least one use of a calendar view of data (not a date/time picker; not a plug-in such as Google calendar)		3	
	Uploading a file into the system with appropriate business reason		3	
	The use of audio/video in an appropriate business context		3	
	At least one use of an administrator configurable timer in an appropriate business context		3	
2. Database access	At least 30 tables used (4 member groups) or 40 tables used (5 member groups)		6	15
	Full referential integrity on all tables		6	
	At least one use of master-detail table relationships (Schneiderman's golden rule on system status applies here)		3	
3. Reports	At least 3 simple list reports in a reporting tool (no control breaks, no graphs, single table)		3	15

	At least 2 transactional report with 2 or more control breaks (with heading and calculated values/totals, multiple tables)		6	
	At least 1 report with adjustable criteria		3	
	At least 1 management report using a graph		3	
4. Flexibility	All data that can change in the future should not be hardcoded but maintained in a sub-module of the system (e.g. Lookup tables)		6	12
	Some business rules are not hardcoded but maintained in a sub-module of the system.		6	
5. Error handling	All system-generated errors are trapped, and consistent, user-friendly error messages are displayed		6	12
	Appropriate data validation on all input fields		6	
6. Help	At least one menu item or other control that opens up a complete help document (HTML, PDF, Help-file)		3	15
	Extensive context-sensitive help. E.g. calling Help on a specific screen/function will automatically open the specific help for that screen/function.		6	
	Search Facility on Help		3	
	Extensive use of hints		3	
7. Security	Logon screen with user ID and password and fixed user profiles		3	13
	Applying two-factor authentication with applicable business reasons.		3	
	Encrypted passwords in database		1	
	Flexible user profiles (i.e. you can dynamically add user profiles that will enable/disable access to certain parts of the system)		6	
8. Audit Trail	An audit trail of all transactions in the system showing at least date, time, user, transaction type, critical data (such as amount and quantity of transaction)		6	9
	Able to search the audit trail on any of the following: date, user, transaction type		3	
9. Deployment	For a desktop application: Fully functional installation disks that take care of application installation requirements (install and uninstall)		3	15
	For an online application: Deployment of application to a publicly accessible web server		3	
	For a mobile application: Deployment to an App Market place (such as the Play Store or the AppStore)		6	
	Deployment of the database to a remote database server		3	

10. Backup and Restore	A backup and restore subsystem exist that backup/restore all data (the system may exit during restore)		3	3
11. Import/Export Data	Able to open Word or Excel and automatically place data in it based on the parameters provided (with a good business reason)		6	9
	XML or JSON: At least 1 XML or JSON file for Importing or Exporting of data (with good business reason)		3	
12. External INPUT device	Simple Link to an external INPUT device using plug-and-play technology, such as a swipe card reader, bar code reader, etc. or a native component such as a QR reader, a GPS component, etc.		3	18
	Loose Link to an external INPUT device using device-specific software. Data or images must seamlessly be stored in the database, but device-specific software is visible to the user. (This could include a digital camera, scanner, voice recording device, thump print reader, etc.)		6	
	Tight Link to an external INPUT device using device-specific software. Data or images must seamlessly be stored in the database, but device-specific software is not visible to the user. (This could include a digital camera, scanner, voice recording device, thump print reader, etc.)		9	
13. External APPLICATION / Services	Integrate an existing web service into your application (with good business reason)		3	9
	A fully functional link to an installed external application system exists and the interface must be shown to work on the external system. Note that this excludes Microsoft Office Applications		6	
14. Multiplatform processing for an appropriate business reason	Appropriate business use of static views on an alternative platform.		3	27
	Appropriate use of dynamic views on an alternative platform (i.e. data is displayed from the system's database)		3	
	Appropriate use of substantial dynamic views on an alternative platform (i.e. both reading and writing data from the system's database)		9	
	Uploading a file through an alternative platform onto the system's database.		3	
	Substantial processing on a third platform (i.e. both reading and writing data from the system's database)		9	

15. Programming Principles	The use of a data layer to facilitate interaction between your database and your business layer		3	12
	The use of an API to facilitate interaction between your business layer and your presentation layer		6	
	Comprehensive use of stored procedures and/or triggers and/or jobs.		3	
16. An innovative addition to the system	Any very advanced innovative addition to the system (e.g. machine learning, AI, blockchain, text mining, IoT, etc.)		9	9
	Maximum Complexity Marks			235
	Our Projects Complexity Marks			166