Electric Vehicle E-Commerce Website

EEFCS 4413,

Software Design Document (SDD)

Adnan Fahad Faizi

Arian Ghazanfariyan

Selena Nguyen

Tirth Patel

Release Date:	
Release State:	
Approval State:	
Approved by:	
Prepared by:	
Reviewed by:	
Path Name:	
File Name:	EECS4413-team project SDD
Document No:	1

Document Change Control

Version	Date	Authors	Summary of Changes
1.0	9/26/2023	Selena Nguyen	Started adding to the Group Meeting Logs.
1.1	9/28/2023	Selena Nguyen	Assigned the use case diagrams, sequence diagrams, component and package diagrams, and GANTT chart to different team members. Updated the Group Meeting Logs.
1.2	09/30/2023	Adnan Faizi	Updated the layout of the document
1.3	10/17/2023	Tirth Patel	Started adding 5 Test cases in TDD section
1.4	10/19/2023	Adnan Faizi	Added component diagram, package diagram and completed module tables.
1.5	10/19/2023	Arian Ghazanfariyan	Added five test cases for catalog view, filtering, listing and sorting
1.6	10/20/2023	Adnan Faizi, Selena Nguyen	Added Use case diagram and completed the document for submission of deliverable 1

Document Sign-Off

Name (Position)	Signature	Date
Adnan Fahad Faizi	Adnan Fahad Faizi	9/28/2023
Tirth Patel	Tirth Patel	9/28/2023
Arian Ghazanfariyan	Arian Ghazanfariyan	9/28/2023
Selena Nguyen	Selena Nguyen	9/28/2023

Table of Contents

1 INTRODUCTION 4
1.1 Purpose 4
1.2 Overview 4
1.3 Resources – References 4
2 MAJOR DESIGN DECISIONS 4
3 USE CASE DIAGRAM 6
3.1 Use Case Diagram 6
4 SEQUENCE DIAGRAMS 6
4.1 Sign In 6
4.2 List electric vehicles available in the catalogue 7
4.3 Feedback form (Original use case) 7
***5* ARCHITECTURE 8**
5.1 Package Diagram 8
.... 9
5.2 Component Diagram 9
6 ACTIVITIES PLAN 11
6.1 Gantt Chart 11
6.2 Project Backlog and Sprint Backlog 11
6.3 Group Meeting Logs 12
7 TEST DRIVEN DEVELOPMENT 13

1 Introduction

1.1 Purpose

This project is an E-Commerce web application that aims to provide a platform for purchasing electric vehicles online. The project is being built by considering agile development methodologies over three sprints that include in-depth requirement analysis and design for the first sprint, implementation of server-side for second sprint and implementation of client-side for third sprint while also considering a test-driven development approach into consideration. Furthermore, this group project was assigned as part of a 4th year course (EECS 4413) at Lassonde of Engineering, York University.

1.2 Overview

The Electric Vehicle E-Commerce website application is implemented using a three-tier architecture consisting of presentation tier, business tier and data tier. The Business tier is further divided into a MVC pattern for the application to be more modular and cohesive. On the home page, visitors can navigate to different sections of the website and view available deals. The website also allows them to browse the catalog, sort and filter the vehicles based on different criterias and view the specifications of each vehicle in more detail; they can also use a loan calculator to determine their monthly payments. Additionally, users are also able to compare multiple vehicles and select different customization options for a chosen vehicle. There are two different users for the website, admins and registered customers. Customers must have an account in order to rate vehicles, submit reviews or complete a purchase. Admins must also go through the necessary authentication to run reports on vehicles sales and application usage. Authenticated users can add vehicles to the shopping cart, remove vehicles from the shopping cart and edit the vehicles in the cart. Lastly, the website also features a chatbot that can help to answer customers' basic questions in order to provide a better user experience.

1.3 Resources – References

1. https://github.com/Faizi-AdnanFahad/EECS4413_Group_Project_WS
2. https://github.com/Faizi-AdnanFahad/EECS4413_Group_Project_WS/blob/main/Diagrams/UML%20Package%20Diagram-Page-4.jpg
3. https://github.com/Faizi-AdnanFahad/EECS4413_Group_Project_WS/blob/main/Diagrams/component%20Diagram%20Final%20Services%20TBA.drawio.png

2 Major Design Decisions

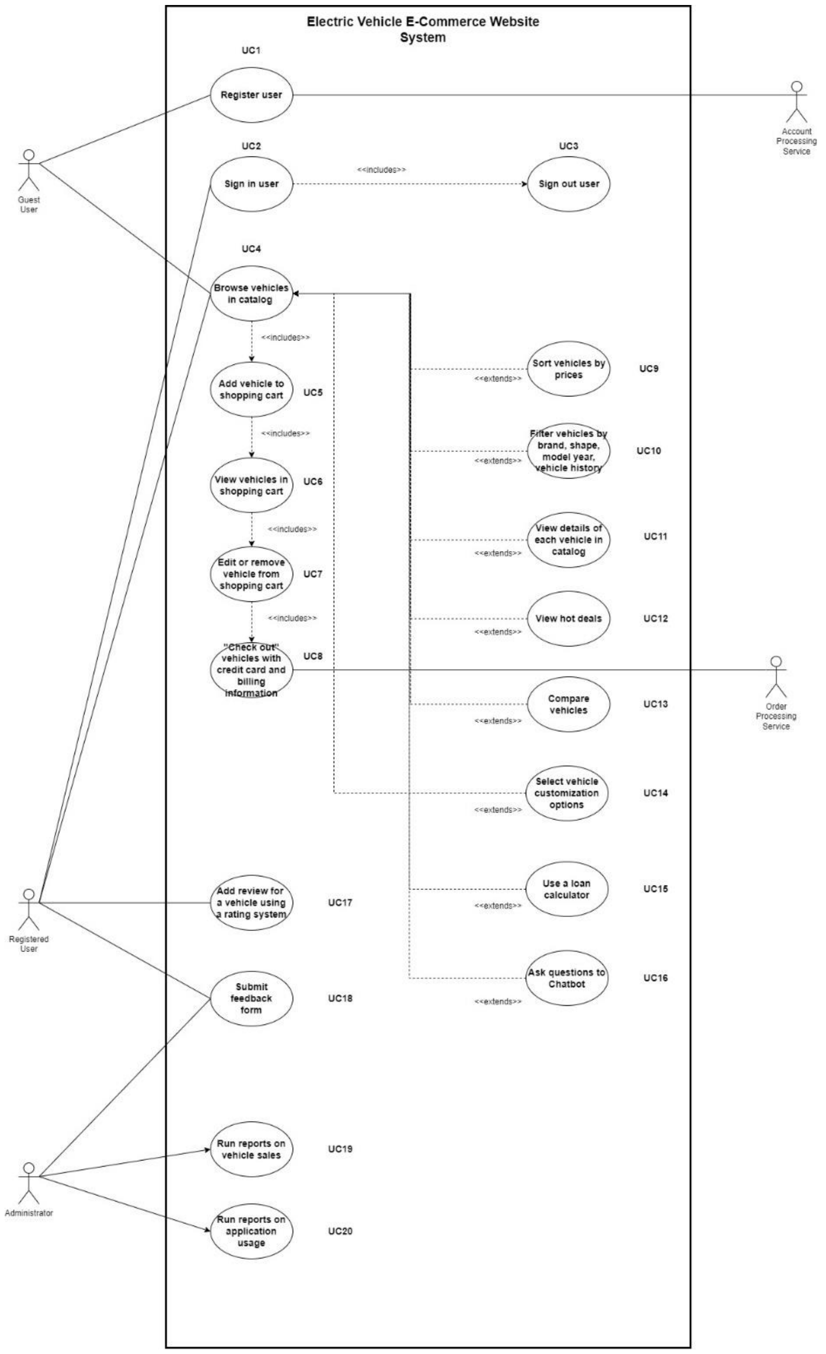
We have used a three-tier architecture consisting of presentation tier, business tier and data tier. The Business tier is further divided into a MVC pattern that helps keep our application further modularized and cohesive. The model communicates with the database through a layer called a data-access object that consists of 5 cohesive components that are able to retrieve information from the database and update tables in the database as required. Controller component in business tier communicates with both the view and the model by getting user information from the view and processing them accordingly. It then creates the necessary objects of the model to process user requests. Each module in the controller is extremely cohesive and is represented in the component diagram with more detail. Lastly, the view component consists of a sequence of views that may be HTML or dynamic jsp pages to present the GUI components to the user and interacts with the presentation view of the application consisting of HTML index pages, CSS styles and JavaScripts.

With the current information at hand, the application uses HTML, CSS, JavaScript and ReactJS for front-end, Java EE as the middleware and a SQL database for back-end.

3 Use case Diagram

3.1 Use Case Diagram

3.2 https://github.com/Faizi-AdnanFahad/EECS4413_Group_Project_WS/blob/main/Diagrams/Use%20Case%20Diagram%20-%20Selena.jpg



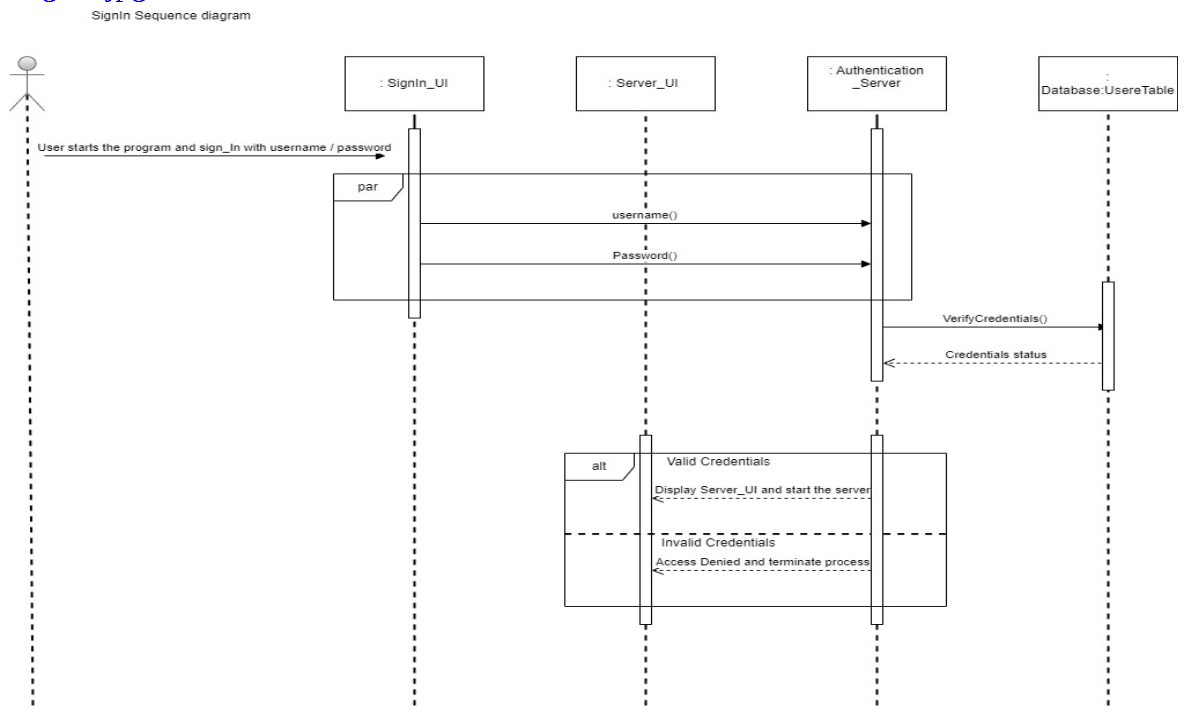
Electric Vehicle E-Commerce Website System
UC1
Register user
Account Processing Service
Guest User
UC2
Sign in user
<<includes>>
UC3
Sign out user
UC4
Browse vehicles in catalog
<<includes>>
Add vehicle to shopping cart
UC5
<<includes>>
View vehicles in shopping cart
UC6
<<includes>>
Edit or remove vehicle from shopping cart
UC7
<<includes>>
"Check out" vehicles with credit card and billing information
UC8
Order Processing Service
Sort vehicles by prices
UC9
<<extends>>
Filter vehicles by brand, shape, model year, vehicle history
UC10
<<extends>>
View details of each vehicle in catalog
UC11
<<extends>>
View hot deals
UC12
<<extends>>
Compare vehicles
UC13
<<extends>>
Select vehicle customization options
UC14
<<extends>>
Use a loan calculator
UC15
<<extends>>
Ask questions to Chatbot
UC16
<<extends>>
Registered User
Add review for a vehicle using a rating system
UC17
Submit feedback form
UC18
Administrator
Run reports on vehicle sales
UC19
Run reports on application usage
UC20

4 Sequence Diagrams

4.1 UC2: Sign In

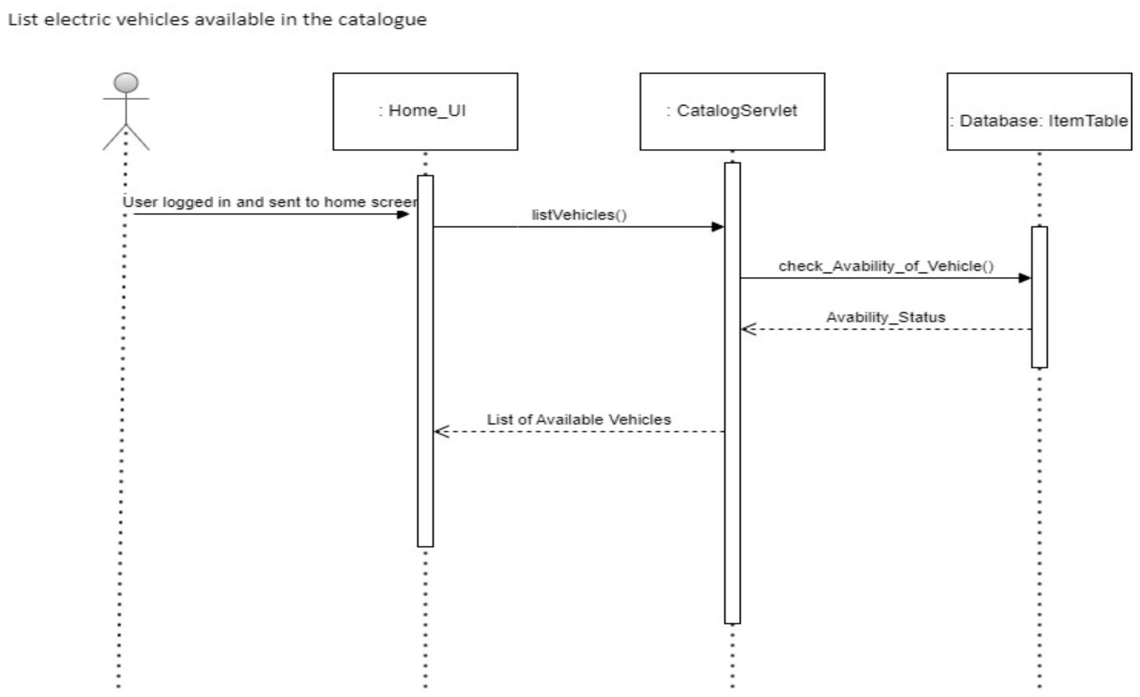
https://github.com/Faizi-AdnanFahad/EECS4413_Group_Project_WS/blob/main/Diagrams/Sequence%20Diagram-

Page-1.jpg



4.2 UC4: Browse electric vehicles available in the catalogue

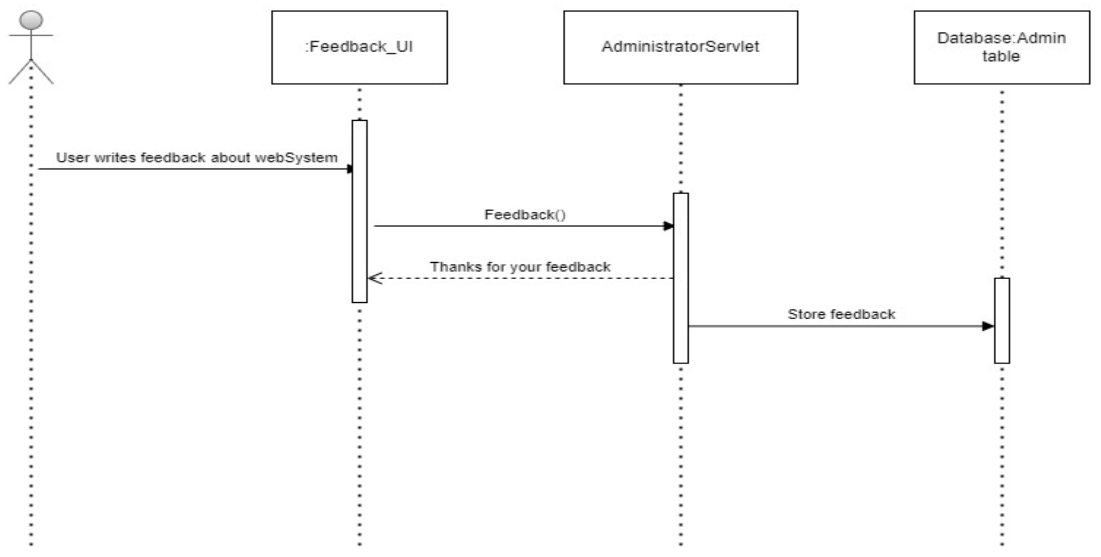
https://github.com/Faizi-AdnanFahad/EECS4413_Group_Project_WS/blob/main/Diagrams/Sequence%20Diagram-Page-2.jpg


List electric vehicles available in the catalogue
: Home_UI
: CatalogServlet
: Database: ItemTable
User logged in and sent to home screen
listVehicles()
check_Avability_of_Vehicle()
Avability_Status
List of Available Vehicles

4.3 UC18: Feedback form (Original use case)

https://github.com/Faizi-AdnanFahad/EECS4413_Group_Project_WS/blob/main/Diagrams/Sequence%20Diagram-Page-3.jpg

Feedback form Sequence diagram

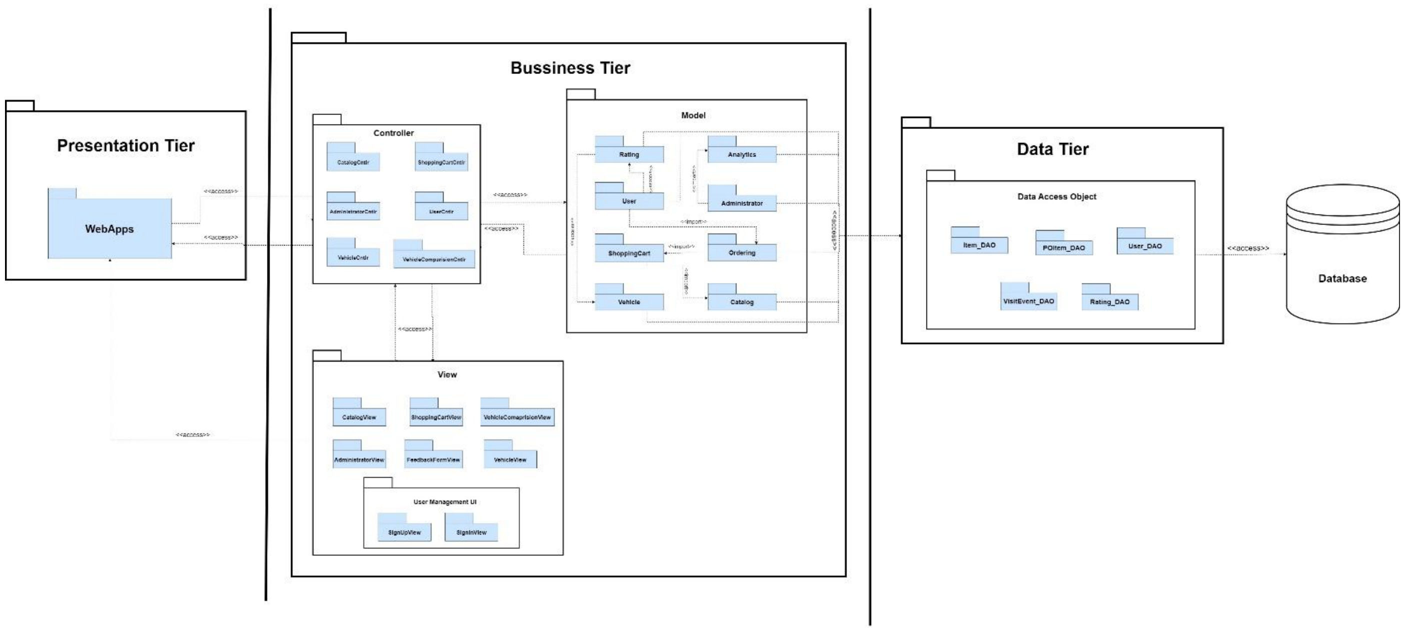


5 Architecture

5.1 Package Diagram

Link to the package diagram in GitHub: https://github.com/Faizi-AdnanFahad/EECS4413_Group_Project_WS/blob/main/Diagrams/UML%20Package%20Diagram-Page-4.jpg

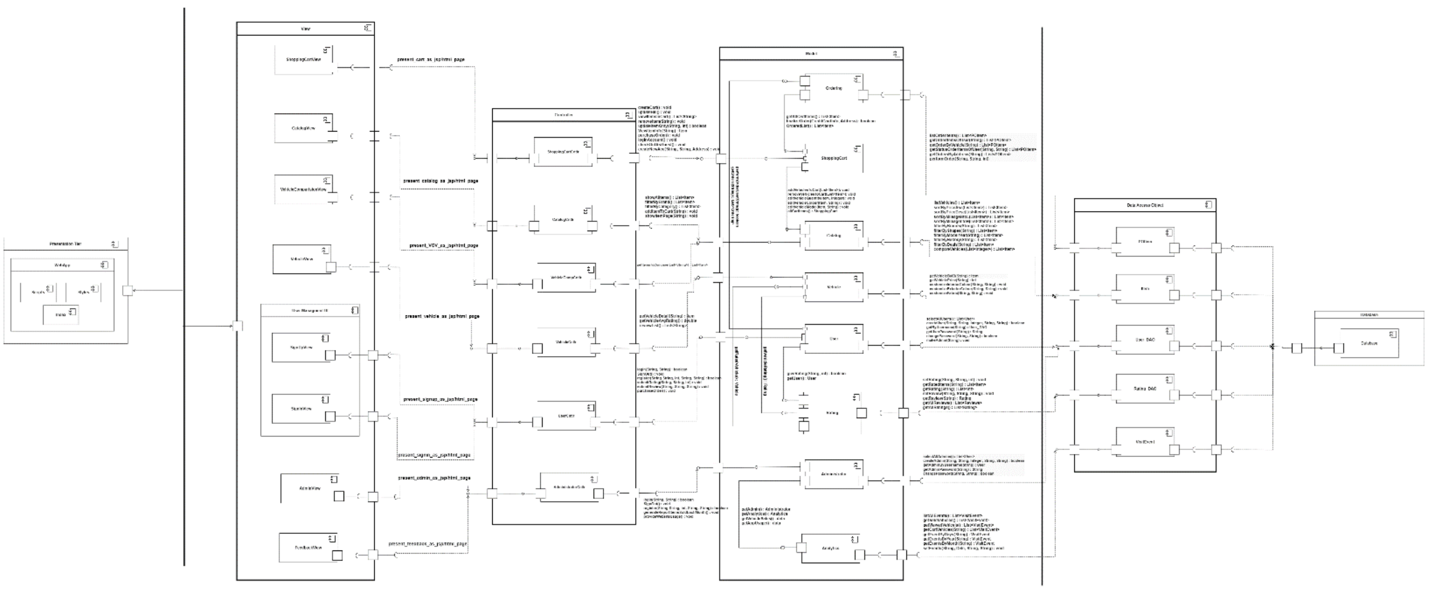
Electric Vehicles E-Commerce Website Package Diagram



5.2 Component Diagram

Link to the Component diagram in GitHub: https://github.com/Faizi-AdnanFahad/EECS4413_Group_Project_WS/blob/main/Diagrams/component%20Diagram%20Final%20Services%20TBA.drawio.png

Electric Vehicles E-Commerce Website Component Diagram



Modules			
Module Name	**Description**	**Exposed Interface Names**	**Interface Description**
UserCntlr	“A module that consist of handling Authentication of users, their rating and ordering vehicles.”	UserCntlr:IAuthServer UserCntlr:IReview	UserCntlr:IAuthServer : “Provides an interface to authenticate a user with passed credentials against the database.” UserCntlr:IReview “Provides an interface to submit reviews and ratings by users to different vehicles”

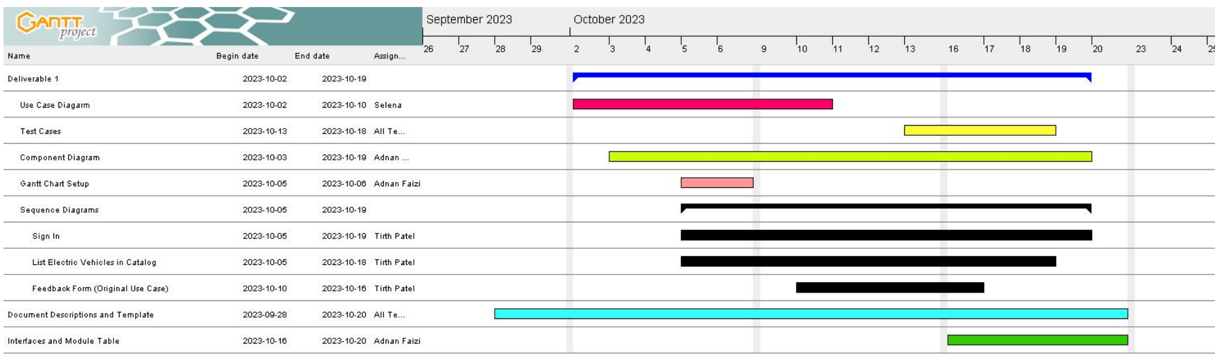
Catalog	"Provides interfaces for shopping cart, ordering, vehicles and more to perform operations that would include browsing through the catalog and selecting and performing the necessary operations."	Catalog:IFilter Catalog: ISorting	Catalog:IFilter : "This interface provides with operation that will filter the catalog based on different criterias such as vehicle shape, brand, mode and so on." Catalog: ISorting : "Provides with operations to sort the catalog based on different criteria and values."
ShoppingCart	"This component provides interfaces that deals CRUD operation of items from the catalog and purchasing and checkout operations."	ShoppingCart: IOrder ShoppingCart: ICatalogOperation	ShoppingCart: IOrder : "Provides information with purchasing and checking out of items from the shopping cart" ShoppingCart: ICatalogOperation : "Provides information about the operations of shopping cart in catalog such as CRUD operations."

Interfaces		
Interface Name	**Operations**	**Operation Descriptions**
UserCntlr:IAuthServer	<void> IAuthServer:login(String, String) used by Ordering, Rating <boolean> IAuthServer:register(String, String, int, String, String) used by Ordering, Rating	<void> IAuthServer:login(String, String) : "Given a username and a password, logs in the user as a normal user of the website." IAuthServer:register(String, String, int, String, String) : "Given a firstname, lastname, age, contact information and address, registers the user as normal users in the website."
UserCntlr:IReview	<void> UserCntlr:IReview:submitRating(String, String, int) used by Vehicle <void> UserCntlr:IReview:submitReview(String, String, String) used by Vehicle	<void> UserCntlr:IReview:submitRating(String, String, int) : "Given a userId, a vehicleID and a rating, the operation creates a rating for the vehicle." <void> UserCntlr:IReview:submitReview(String, String, String) : "Given a userId, a vehicleID and a review description, the operation creates a review for the vehicle."
Catalog:ISorting	<List<Item>> Catalog:ISorting:sortByPriceAsc(List<Item>) used by Catalog <List<Item>> Catalog:ISorting:sortByPriceDesc(List<Item>) used by Catalog <List<Item>> Catalog:ISorting:sortByMileageHtoL(List<Item>) used by Catalog	<List<Item>> Catalog:ISorting:sortByPriceAsc(List<Item>) : "Sorts the items in catalog by their prices from lowest to highest" <List<Item>> Catalog:ISorting:sortByPriceDesc(List<Item>) : "Sorts the items in catalog by their prices from highest to lowest" <List<Item>> Catalog:ISorting:sortByMileageHtoL(List<Item>) : "Sorts the items in catalog by their mileage from highest to lowest"

	<List<Item>> Catalog:ISorting:sortByMilageLtoH(List<Item>) used by Catalog	<List<Item>> Catalog:ISorting:sortByMileageHtoL(List<Item>) : “Sorts the items in catalog by their mileage from lowest to highest”
Catalog:IFilter	<List<Item>> Catalog:IFilter:filterByBrands(String) used by Catalog <List<Item>> Catalog:IFilter:filterByShapes(String) used by Catalog <List<Item>> Catalog:IFilter:filterByDeals(String) used by Catalog	<List<Item>> Catalog:IFilter:filterByBrands(String) : “Filters the items in catalog by their brands.” <List<Item>> Catalog:IFilter:filterByShapes(String) : “Filters the items in catalog by their Shapes.” <List<Item>> Catalog:IFilter:filterByDeals(String) : “Filters the items in catalog by their Deals.”
ShoppingCart: ICatalogOperation	<void> ShoppingCart: ICatalogOperation:addVehiclesToCart(List<Item>) used by Catalog, ShoppingCart, Ordering <void> ShoppingCart:ICatalogOperation :removeVehiclesFromCart(List<Item>) used by Catalog, ShoppingCart, Ordering	<void> addVehiclesToCart(List<Item>) : “Adds the requested vehicles from the catalog to the list of shopping cart of the user.” <void> removeVehiclesFromCart(List<Item>) : “Removes the requested vehicles from the shopping of the user.”
ShoppingCart: IOrder	<List<Item>> ShoppingCart: ICatalogOperation:getAllCartItems() used by POItem, Ordering Boolean ShoppingCart:IOrder: finalizeOrder(CreditCartInfo, Address) used by POItem, Ordering	<List<Item>> ShoppingCart: ICatalogOperation:getAllCartItems() : “gets all the vehicles in cart to purchase.” ShoppingCart:IOrder: finalizeOrder(CreditCartInfo, Address) : “finalizes the purchase from the list of selected items in the shopping cart by providing the address and the credit cart information.”

6 Activities Plan

6.1 Gantt Chart



Name	Begin date	End date	Assign...
Deliverable 1	2023-10-02	2023-10-19	
Use Case Diagarm	2023-10-02	2023-10-10	Selena
Test Cases	2023-10-13	2023-10-18	All Te...
Component Diagram	2023-10-03	2023-10-19	Adnan ...
Gantt Chart Setup	2023-10-05	2023-10-06	Adnan Faizi
Sequence Diagrams	2023-10-05	2023-10-19	
Sign In	2023-10-05	2023-10-19	Tirth Patel
List Electric Vehicles in Catalog	2023-10-05	2023-10-18	Tirth Patel
Feedback Form (Original Use Case)	2023-10-10	2023-10-16	Tirth Patel
Document Descriptions and Template	2023-09-28	2023-10-20	All Te...
Interfaces and Module Table	2023-10-16	2023-10-20	Adnan Faizi

6.2 Project Backlog and Sprint Backlog

Deliverable 1

~~- Use Case Diagrams~~
~~- 16 Test Cases~~

- ~~Component Diagrams~~
- ~~Gantt Chart Setup~~
- ~~Sequence Diagrams~~
  - ~~Sign in~~
  - ~~List electric vehicles available in catalog~~
  - ~~Feedback Form (Original Use Case)~~
- ~~Document Descriptions and Template~~
- ~~Interface and Module Table~~

Deliverable 2

- TBD

Deliverable 3

- TBD

6.3 Group Meeting Logs

In this Section you write minutes of each meeting, listing the attendance, what the topics of discussion in the meeting were, any decisions that were made, and which team members were assigned which tasks. These minutes must be submitted with the project report in each deliverable and will provide input to be used for the overall assessment of the project.

Present Group Members	**Meeting Date**	**Issues Discussed / Resolved**
Adnan Fahad Faizi Tirth Patel Selena Nguyen	9/26/2023 (8:00 PM – 8:30 PM)	• Read the project description • Planned our next meeting • Assigned Tirth to make the GANTT diagram • Assigned Selena to fix up the rest of the Project Report template and add to the Group Meeting Logs
Adnan Fahad Faizi Tirth Patel Arian Ghazanfariyan Selena Nguyen	9/28/2023 (5:30 PM – 6:15 PM)	• Chose 6 use cases that we would make use case diagrams for • Chose which 3 use cases we would make sequence diagrams for • Assigned which members should work on the use case diagrams, sequence diagrams, component and package diagrams, and GANTT chart • All members will work on the Report Template • Adnan: Package Diagram, Component Diagram (with Arian) • Arian: Component Diagram (with Adnan) • Tirth: Sequence Diagrams • Selena: Use Case Diagrams, GANTT Chart, Group Meeting Logs
Adnan Faizi Arian Ghazanfariyan	10/13/2023 (8:00 PM – 9:00 PM)	• Discussed the structure of the package and component diagram and planned our work for these items accordingly.
Adnan Faizi	10/15/2023	• Divided up the rest of the work and assigned 4 Test cases to each of the members.

Arian Ghazanfariyan Selena Nguyen	(10:00 AM – 10: 30 AM)	• Discussed what still needs to be done and planned an in-person meeting for Thursday October 19th • Came up with our own use case: Feedback form
Adnan Fahad Faizi Tirth Patel Arian Ghazanfariyan Selena Nguyen	10/19/2023 (5:30 PM – 7:30 PM)	• Reviewed and finalized everything • Added test cases for 4 use cases (Register, Sign in, Sign out, and Chatbot answers)

7 Test Driven Development

Test ID	Test_shoppingCart_01
Category	Addition of vehicles from catalog to the shopping cart.
Requirements Coverage	UC9-Successful-vehicle-addition-to-Shopping-cart
Initial Condition	The system has been initiated while showing the catalog of vehicles.
Procedure	*1. The user is able to see an add button beside each vehicle in the catalog.* *2. The user clicks on the add button for a vehicle.* *3. The user goes to the carts page by clicking on go to cart button.*
Expected Outcome	The user is able to see the vehicle added to the list of vehicles in the shopping cart.
Notes	The user should only try to add one item at a time to the cart.

Test ID	Test_shoppingCart_02
Category	Editing vehicles in the shopping cart
Requirements Coverage	UC10-Successful-vehicle-edit-in-the-Shopping-cart
Initial Condition	The system has been initiated while showing the user is in the shopping cart page.
Procedure	*1. The user browsers from the list of items in the shopping cart and finds the vehicle that needs to be edited.* *2. The user increases/decreases the vehicles quantity or changes the color, type, or model of the vehicle.* *3. The user waits for the page to be refreshed or refreshes the page manually.*
Expected Outcome	The user is able to see the changes made on the vehicle such as the quantity has changed, or the model/type/color has changed.

Notes	*The user already has a one or more vehicles in the shopping cart.*

Test ID	Test_shoppingCart_03
Category	Removing vehicles in the shopping cart
Requirements Coverage	UC10-Successful-vehicle-removal-in-the-Shopping-cart
Initial Condition	The system has been initiated while showing the user is in the shopping cart page.
Procedure	*1. The user browsers from the list of items in the shopping cart and finds the vehicle that needs to be removed.* *2. The user clicks on the remove button on the side of the vehicle in the shopping cart or changes the quantity of the item to 0.* *3. The user waits for the page to be refreshed or refreshes the page manually.*
Expected Outcome	The user is able to see that the vehicle is no longer in the list of vehicles in the shopping cart.
Notes	*The user already has a one or more vehicles in the shopping cart.*

Test ID	Test_checkout_01
Category	Checking out an item with credit card information and other relevant information
Requirements Coverage	UC11-Successful-checkout-of-an-order
Initial Condition	The system has been initiated while showing that the user is in the checkout page checking out the items to purchase.
Procedure	*1. The user selects one or more items in the shopping cart.* 2. The user enters its credit card information. 3. The user enters its shipping information. 4. The user clicks on the confirm order button to complete the purchase.
Expected Outcome	The user’s card information and order request has been processed and the user is presented with a confirmation message that the order has been successfully placed.
Notes	*The user enters valid credit-card and shipping information.*

Test ID	*Test_ViewHotDeal_01*
Category	*Updated hot deals*
Requirements Coverage	*UC12-Successful-Getting-Hot-Deals*
Initial Condition	*The system has been initiated while showing great hot deals to Users.*
Procedure	*1. The user opens system in browser.* *2. The system refreshes the home page and shows updated tending deals to users.*

	3 The user browsers from the list of hot deals available in the system and selects items they like.
Expected Outcome	*The user is able to see hot deals available right now in the system*
Notes	*The system should be updated with hot available deals.*

Test ID	*Test_LoanCalculator_01*
Category	*Using a loan calculator*
Requirements Coverage	*UC13-Successful-Getting-Result-In-Loan-Calculator*
Initial Condition	*The system has been initiated while showing that the user is in the loan calculator section.*
Procedure	*1. The user clicks on loan calculator button and the system refreshes and opens loan calculator.* *2. The user adds the required details to calculate their loan price.* *3.The user enters vehicle price.* *4. The user enters down payment and other required information and clicks on calculate loan.*
Expected Outcome	*The user is able to see how much the estimated amount of payment they need to make each month.*
Notes	*The user enters valid information in the loan calculator to calculate the estimate amount of payment.*

Test ID	*Test_CompareVehicle_01*
Category	*Be able to compare vehicle*
Requirements Coverage	*UC14-Successful-in-Comparing-Vehicle*
Initial Condition	*The system has been initiated while showing that the user is in the comparing vehicle page.*
Procedure	*1. The user selects vehicles that they like to compare each other with (from the available vehicle list).* *2. The user enters vehicle model information.* *3 The user clicks the compare button and gets a result with detailed information.*
Expected Outcome	*The user is able to see the good and bad things about a vehicle from selected vehicles.*
Notes	*The user selects and enters valid information of vehicle's*

Test ID	*Test_VehicleCustomization_01*
Category	*Be able to customize the vehicle*
Requirements Coverage	*UC15-Successful-in-Customizing-Vehicle*
Initial Condition	*The system has been initiated while showing that the user is in the vehicle customization page when customizing vehicle.*
Procedure	*1. The user selects a vehicle that they want to make any customization.* *2. The user selects the customization parts available in the vehicle that they can customize.*

	3. The user adds their custom type in vehicle. *4. The user clicks the customize button and gets a view of their customized vehicle and the option to finalize their custom vehicle.*
Expected Outcome	*The user is able to view their custom vehicle after clicking the customize button.*
Notes	*The user enters valid information about the vehicle they like to customize.*

Test ID	*Test_feedbackform_01*
Category	*Be able to receive feedback through feedback form*
Requirements Coverage	*UC16-Successful-Reciving-feedback*
Initial Condition	*The system has been initiated while showing that the user is in the feedback form page and can add feedback.*
Procedure	*1. The user can enter text in feedback section.* *2. The user clicks on submit and form is stored in admin table.*
Expected Outcome	*The user is able to see the return message “Thanks for your feedback”.*
Notes	*The user is able to enter text in form*

Test ID	*Test_listCatalog_01*
Category	*Catalog*
Requirements Coverage	*UC M1 browse the catalog and see All, By Brand, By Categories products.*
Initial Condition	*Catalog is populated*
Procedure	*User selects "Browse All Products"*
Expected Outcome	*The system displays a list of all products in the catalog*
Notes	*User sees a list of all products in the catalog.*

Test ID	*Test_listCatalog_02*
Category	*Catalog*
Requirements Coverage	*UC MI: Browse the catalog and see By Brand products*
Initial Condition	*Catalog is populated*
Procedure	*User selects "Browse by Brand"*
Expected Outcome	*The system displays a list of products by the brand*
Notes	*Only one brand products are shown.*

Test ID	*Test_sorting_01*
Category	*Sorting*
Requirements Coverage	*UC M2: select an item and see the detailed information for that item (price, ratings, etc.).*

Initial Condition	*Catalog with electric vehicles*
Procedure	*User selects "Sort by Price (Ascending)"*
Expected Outcome	*The system sorts electric vehicles by ascending price*
Notes	*Vehicles are listed from the lowest price to the highest.*
Test ID	*Test_sorting_02*
Category	*Sorting*
Requirements Coverage	*UC M2: select an item and see the detailed information for that item (price, ratings, etc.).*
Initial Condition	*Catalog with electric vehicles*
Procedure	*User selects "Sort by Mileage (High to Low)"*
Expected Outcome	*The system sorts electric vehicles by descending mileage*
Notes	*Vehicles are listed from highest mileage to lowest.*

Test ID	*Test_filter_01*
Category	*Catalog*
Requirements Coverage	*Filter electric vehicles by brands*
Initial Condition	*Catalog is populated*
Procedure	*User selects "Filter by Brand:"*
Expected Outcome	*The system displays electric vehicles filtered by the brand*
Notes	*Only relevant brand products are shown.*

Test ID	*Test_RegisterUser_01*
Category	*Register user*
Requirements Coverage	*UC17-Successful-User-Registration*
Initial Condition	*The system has been initiated while showing that the user is on the registration (account creation) page.*
Procedure	*1. The user opens the system in their browser.* *2. The user is on the Home page and can see the Register button to register a new account.* *3. The user clicks on the Register button to create a new account.* *4. The user fills in their username, first and last name, password, age, and address.* *5. The system validates whether the username, first name, last name, password, age, and address, are valid.* *6. The user waits for the page to verify the information.* *7. The user clicks the Register button.*
Expected Outcome	*The system refreshes the page and shows feedback that the user has successfully registered an account. The feedback shows the message, "Thank you for registering!"*

Notes	*The user is able to fill in text on the registration page.*

Test ID	*Test_SignIn_01*
Category	*Sign in user*
Requirements Coverage	*UC18-Successful-User-Sign-In*
Initial Condition	*The system has been initiated while showing that the user is on the sign in page, and already has an account.*
Procedure	*1. The user opens the system in their browser.* *2. The user is on the home page and can see the Sign in button to sign into a new account.* *3. The user clicks on the Sign in button to login to their account.* *4. The user fills in their username and password.* *5. The system validates whether the username and password are valid.* *6. The user clicks the Sign in button to login to the system.* *7. The user waits for the page to load to sign them in.* *8. The system successfully signs them in.* *9. The system redirects the user to the home page.*
Expected Outcome	*The system shows the home page, and that the user is logged into their account.*
Notes	*The user is able to fill in text on the Sign in page.*

Test ID	*Test_SignOut_01*
Category	*Sign out user*
Requirements Coverage	*UC19-Successful-User-Sign-Out*
Initial Condition	*The system has been initiated while showing the button to sign out, and the user already has an account and is logged into the system.*
Procedure	*1. The user has the system open in their browser and is already logged in.* *2. The user can see the Sign out button.* *3. The user clicks on the Sign out button to sign out of their account.* *4. The user waits for the page to load to sign them out.* *5. The system successfully signs them out.* *6. The system redirects the user to the Sign in page.*
Expected Outcome	*The user is successfully signed out of their account and is redirected to the Sign in page.*
Notes	*The user is able to click the Sign out button.*

Test ID	*Test_ChatbotAnswer_01*
Category	*Ask questions to a chatbot able to answer basic questions and help customers navigate to a specific vehicle or support page*

Requirements Coverage	*UC20-Successful-Chatbot-Basic-Answer*
Initial Condition	*The system has been initiated while showing that the user is on the Catalog page and has the chatbot popup tab visible.*
Procedure	*1. The user has the system open in their browser and is already logged in.* *2. The user is on the Catalog page and can see the Chatbot popup tab.* *3. The user enters text to write a message to the Chatbot's input text field, such as, "Where can I change my password?"* *4. The Chatbot is able to successfully give a response to the user's message and tells them where they can change their password.* *5. The Chatbot sends a message back to the user that they can change their password through the Change Password button in their user account settings.*
Expected Outcome	*The Chatbot sends a message back to the user about where they can navigate to the Change Password button.*
Notes	*The user is able to enter text into the Chatbot's text input*

Test ID	*Test_ChatbotAnswer_02*
Category	*Ask questions to a chatbot able to answer basic questions and help customers navigate to a specific vehicle or support page*
Requirements Coverage	*UC21-Successful-Chatbot-Basic-Navigation-Vehicle*
Initial Condition	*The system has been initiated while showing that the user is on the Catalog page and has the chatbot popup tab visible.*
Procedure	*1. The user has the system open in their browser and is already logged in.* *2. The user is on the Catalog page and can see the Chatbot popup tab.* *3. The user enters text to write a message to the Chatbot's input text field.* *4. The Chatbot is able to successfully give a response to the user's message and help them navigate to a specific vehicle by sending them a link to the specific vehicle.*
Expected Outcome	*The Chatbot sends a message back to the user that includes a link about a specific vehicle.*
Notes	*The user is able to enter text into the Chatbot's text input.*

Test ID	*Test_ChatbotAnswer_03*
Category	*Ask questions to a chatbot able to answer basic questions and help customers navigate to a specific vehicle or support page*

Requirements Coverage	*UC22-Successful-Chatbot-Basic-Navigation-Vehicle*
Initial Condition	*The system has been initiated while showing that the user is on the Catalog page and has the chatbot popup tab visible.*
Procedure	*1. The user has the system open in their browser and is already logged in.* *2. The user is on the Catalog page and can see the Chatbot popup tab.* *3. The user enters text to write a message to the Chatbot's input text field.* *4. The Chatbot is able to successfully give a response to the user's message and help them navigate to a specific vehicle by sending them a link to the support page.*
Expected Outcome	*The Chatbot sends a message back to the user a link to a support page.*
Notes	*The user is able to enter text into the Chatbot's text input.*