4413 Team Project
Deliverable 1

Document Sign-Off

Name (Position)	Signature	Date
Adil Guluzade (...)	A.G.	02/14/2025
Taha Asim	T.A.	02/14/2025
Amir Ahmadnasab	A.A.	02/14/2025
Hulya Yasar	H.Y.	02/14/2025

Contents

1 Introduction

1.1 Purpose

The purpose of this document is to talk about the specifications and structure of the online store application made for the automotive company. It will explain the functionalities of the application, how each component interacts with each other, our process of developing the application and our testing procedure.

1.2 Overview

In this design document, we show our project's structure, development approach and other design decisions. In section 2 we talk about major design decisions like the architecture, structure of the program, the APIs we will use etc. In section 3, we show the use case diagram for the application showing how the user and the administrator will be able to interact with the application and all the features they'll be able to use. Section 4 shows the sequence diagram. For the sequence diagram, we show a depiction of 3 separate use cases showing how the objects will interact with each other over time. The 5th section shows a visual depiction of the applications architecture with two different diagrams, one showing a component diagram of the front end and another of the backend. Another diagram in the architecture section shows the package diagram depicting the classes and packages we would need to make for the application. In the activity plan, we outline the project backlog, sprint planning, and meeting logs to track team progress. Finally, in the test-driven development section, we make test cases, talk about the testing procedure and mention the expected outcome of the test. We do this for all aspects of the application to make sure it is not lacking in any department that is working the way it is required.

1.3 Resources - References

What is a component diagram? (n.d.). https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-component-diagram/

Bus, L., & Eccam. (n.d.). *Reqview Software Requirements Specification Example*. ReqView Software Requirements Specification Example. https://www.reqview.com/papers/ReqView-Example_Software_Requirements_Specification_SRS_Document.pdf

2 Major Design Decisions

Architectural Pattern Choice

For the development of the online store for electric vehicles, we have chosen a multi-tier architecture to separate concerns and improve maintainability. The system is divided into:

1. Presentation Layer (Front-end): Developed using React.js, providing a responsive and interactive UI.
2. Business Logic Layer (Back-end): Implemented with Spring Boot (Java), processing requests and enforcing business rules.
3. Data Layer: A cloud-hosted SQL database for storing vehicle details, user information, orders, and transactions.
4. API Gateway: A RESTful API acting as an intermediary between the front-end and back-end, handling authentication and security.

This architecture enhances scalability, security, and performance while ensuring a clear separation of concerns.

Design Pattern Choices

The design patterns chosen align with the system's architecture and modularity, as seen in the package diagram:

1. Model-View-Controller (MVC) - The User Interface Layer (View) interacts with the Controller Layer (API Gateway) and Business Logic Layer (Model), ensuring the separation of concerns.
2. Observer Pattern - The Analytics & Reporting module (Sales Report Service, User Activity Tracking) observes events from the Business Logic Layer (e.g., Order Processing, Payment Processing) to generate real-time updates.
3. Factory Pattern - Payment Processing follows the Factory Pattern to dynamically handle different payment methods.
4. Singleton Pattern - Authentication & Authorization in the Security Layer uses the Singleton Pattern to manage session validation centrally.
5. Data Access Object (DAO) Pattern - The Data Access Layer (Database Service) abstracts database interactions, keeping business logic separate from data operations.

These patterns enhance maintainability, scalability, and system efficiency.

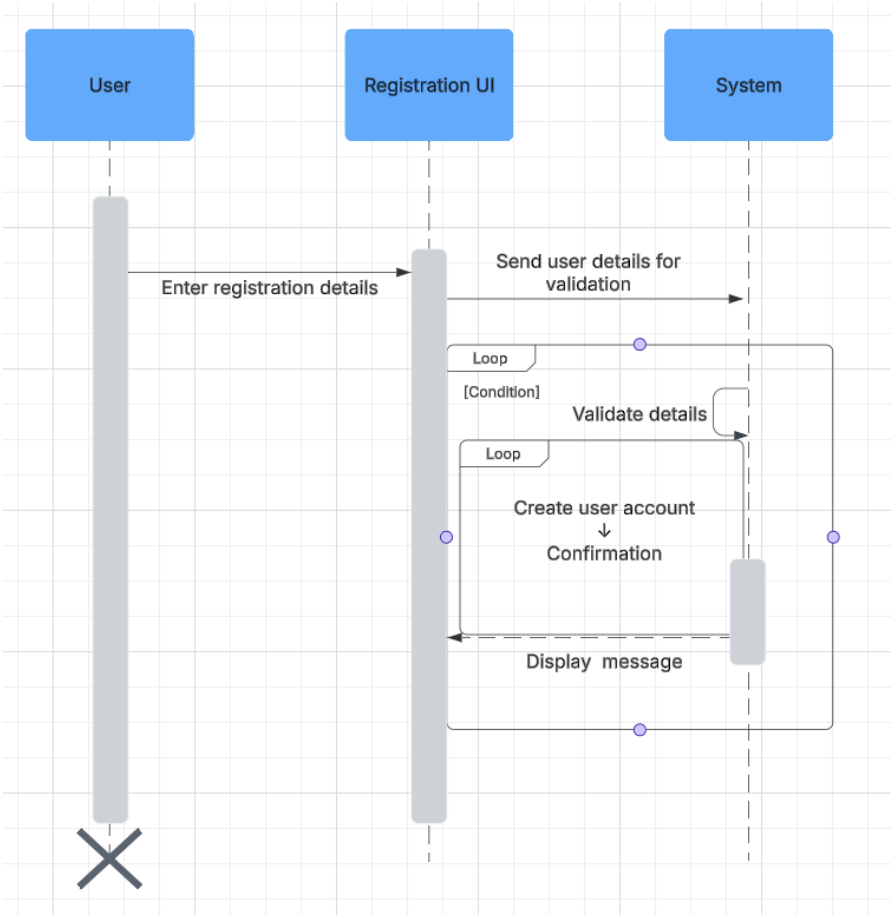
Modularization Criteria

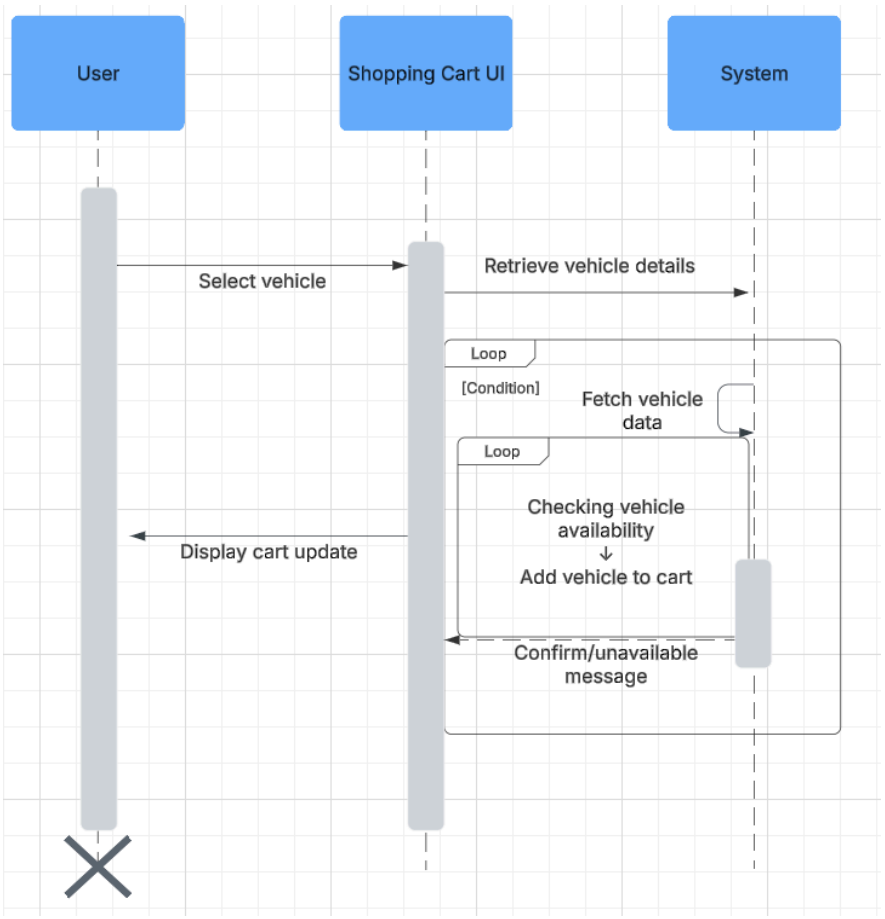
Our Modularization criteria is going to be based around high cohesion and low coupling meaning we will make sure we will keep each class as for their own defined purpose and divide each function into a module such as:

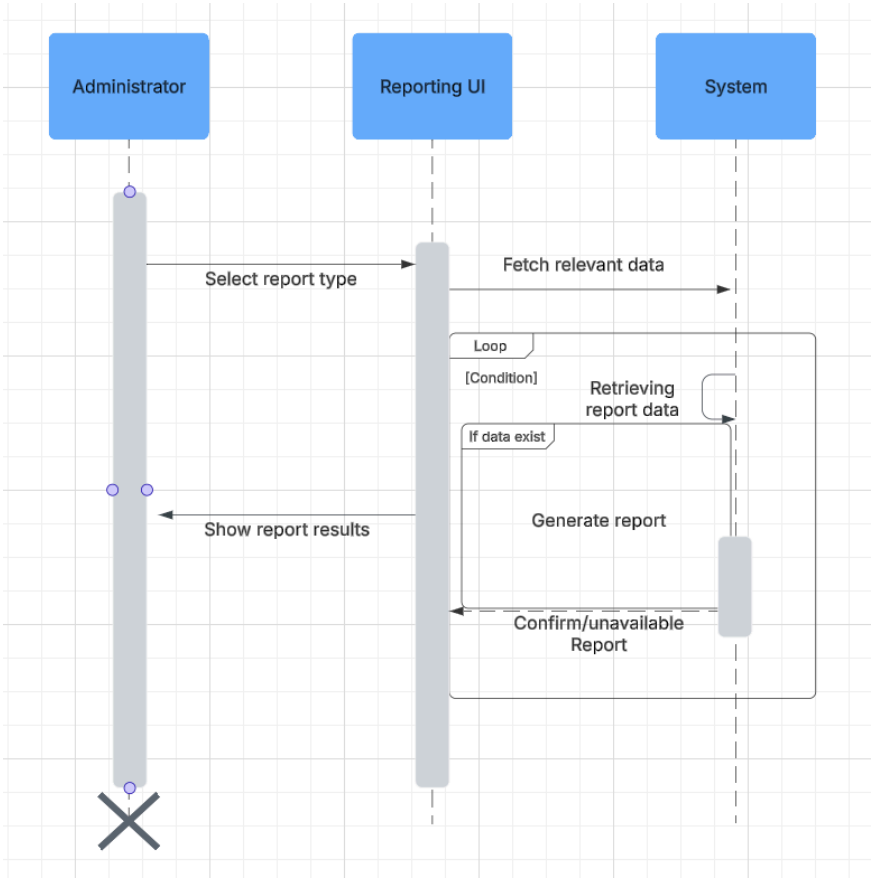
1. User management is going to take care of user registration, logging in and signing off.
2. product catalogue will allow people to look at different vehicles, customize it, view more details of it etc.
3. Cart will take care of editing, managing and deleting products from the cart.
4. report generation will take care of running analysis and provide the administrators with accurate application usage and sales reports.

3 Use case Diagram

4 Sequence Diagrams

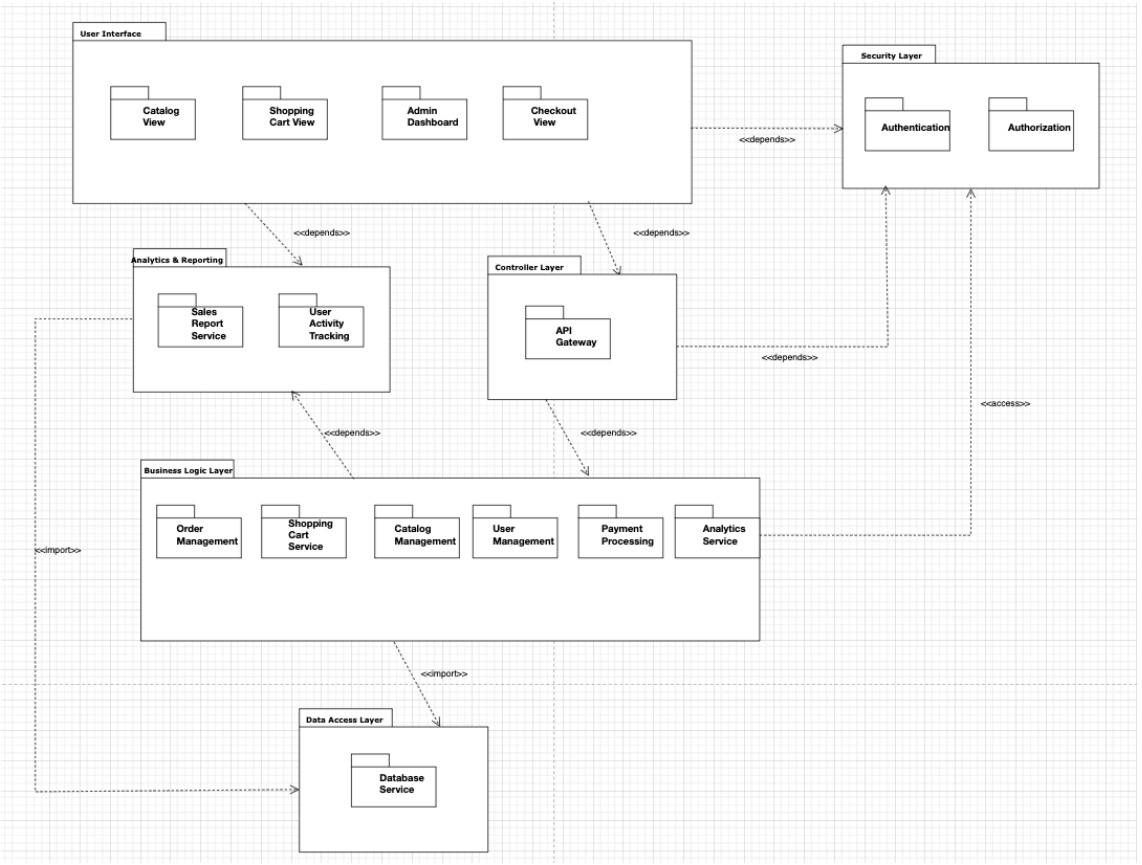




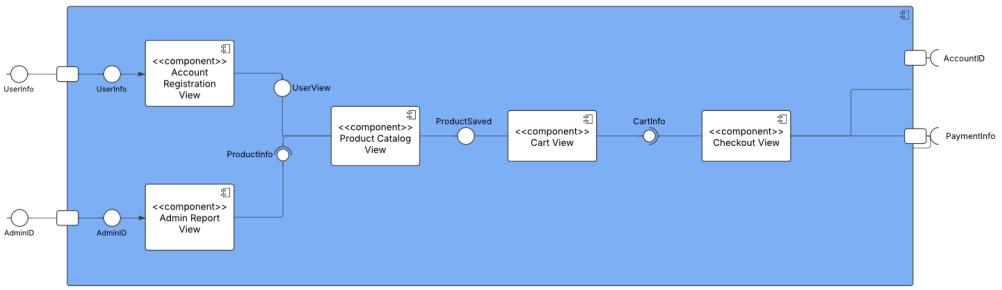


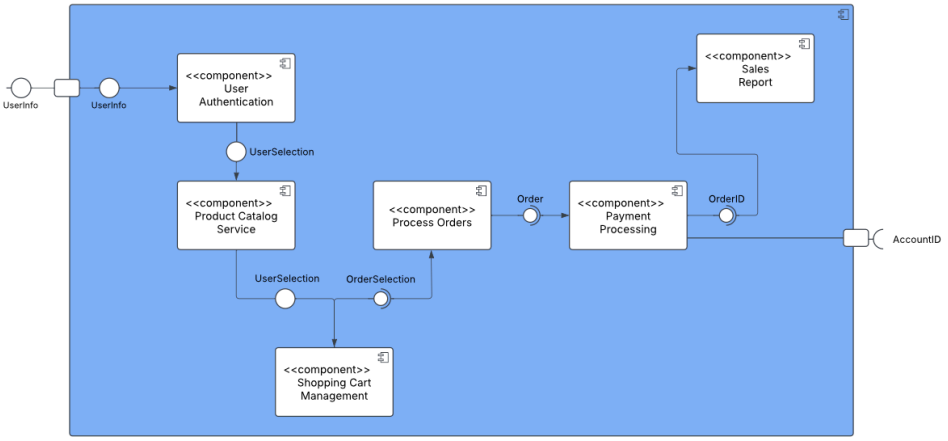
5 Architecture

PACKAGE DIAGRAM



COMPONENT DIAGRAM





Modules			
Module Name	**Description**	**Exposed Interface Names**	**Interface Description**
M1: Product Catalog	Manage product details on the home page of the e-commerce website	M1:I1 Product display screen	M1:I1 displays all the our products on the product display screen where users can scroll through all of the options available
M2: Process Orders	Handles the processing of orders that users make, such as adding products to the shopping cart	M2:I1 Process Orders	M2:I3 After clicking on check-out on the shopping cart screen, users are asked to enter all necessary information in order to create an order
M3: User Authentication	Manages the authentication of user log-in/sign-up	M3:I1 User Log-in M3:I2 User Sign-up	M3:I1 displays a log-in screen for the user to enter the email/username and password associated with their account M3:I2 displays a sign-up screen for the user to enter their account details such as name, email, password, etc.
M4: Shopping Cart Management	Manages and holds all the products saved by	M4:I1 Shopping cart screen	M4:I1 This display will show users what products they

	the user in a shopping cart		have saved in their shopping cart
M5: Payment Processing	Handles payments when users try to purchase products	M5:I1 Payment Process	M5:I1 When the user is in the process order screen, they will be asked to provide a payment in order to process their order
M6: Sales Report	Adds all the sales made after users make a payment, then generates a sales report for the admin	M6:I1 Admin sales report	M6:I1 After each sale, the sales order information gets sent to the database and the admin will be able to access the reports at any given time

Interfaces		
Interface Name	**Operations**	**Operation Descriptions**
M1:I1	<String> M1:I1 productName() used by M2, M4, M5, M6 <Image> M1:I1 productImage(int x) used by M2 and M4 <Integer> M1:I1 productPrice() used by M2, M4, M5, M6	M1:I1 productName(): This operation displays the names (strings) of each product on display M1:I1 productImage(int x): Along with the name of the product, an image will be displayed of said product M1:I1 productPrice(): Along with the name of the product, a price will be displayed which will be required to display the price of each product
M2:I1	<String> M2:I1 productName() used by M4, M5, M6 <Integer> M2:I1 productPrice() used by M4, M5, M6	M2:I1 productName(): This operation displays the names (strings) of each product on display when processing orders M2:I1 productPrice(): Along with the name of the product, a price will be displayed which will be required in order to process orders
M3:I1	<String> M3:I1 logInInfo(string user, string password) used by M3, M4, M5, M6	M3:I1 logInInfo(user, password): This operation requests the username and password in order for the user to log into their account
M3:I2	<void> M3:I2 signUpInfo(string name, string email, string username, string password) used by M3, M4, M5, M6	M3:I2 signUpInfo(name, email, username, password): This operation is where the user enters required information in order to create an account for them
M4:I1	<String> M4:I1 productName() used by M5, M6 <Image> M4:I1 productImage() <Integer> M4:I1 productPrice() used by M5, M6	M4:I1 productName() Within the shopping cart, the name of the product will be displayed M4:I1 productImage() Along with the name of the product, the image will also be displayed

		M4:I1 productPrice() Along with the name and image, the price is required to be shown
M5:I1	<Integer> M5:I1 paymentProcess(int cardNum, int exp, int secCode, string name) used by M6	M5:I1 paymentProcess() In order to process the payment, user will be required to provide a payment option along with the necessary details listed in the parameter
M6:I1	<Table> salesReport()	M6:I1 salesReport() The admin will be able to view the reports at any given time

This section will be revised in deliverables 2 & 3 to have an updated document at the end of the term.

6 Activities Plan

1.1 Project Backlog and Sprint Backlog

In this Section, and assuming you follow a Scrum process model, provide a list of product backlog items so that you can select items for your Sprint backlog. Make sure the product backlog list and the tasks in each product backlog item are consistent with the Gantt Chart in Section 6.1. above.

1.2 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**
Adil, Hulya, Taha, Amir	Jan 25	We discussed and reviewed the project's requirements. Made sure everyone understood it and assigned everyone the UML diagrams.
Adil, Hulya, Taha	Feb 8	We compared and checked each other's diagrams and talked about test cases and how many test cases we may need to make for certain functionalities.
Taha, Amir	Feb 13	Double-checked through all the test cases and made sure all of the functions had test cases.

1.3 Project Backlog and Sprint Backlog

Task	Priority	Due Date (Flexible)
User Authentication Screen	High	Mar 7, 2025
Design Vehicle Catalogue Page	High	Mar 7, 2025
Sales Report Screen for Admin	Medium	Mar 7, 2025
Implement Shopping Cart	High	Mar 26, 2025
Develop a Payment Checkout Screen	High	Mar 26, 2025
Generate a Loan Calculator	Low	Mar 26, 2025
Create Chatbot	Medium	Apr 8, 2025
Implement Feature to Customize Cars	Medium	Apr 8, 2025
Design a Screen to View Hot Deals	Medium	Apr 8, 2025

2 Test Driven Development

Test cases will be provided in the form of a table as follows:

Test ID	Test1A
Category	Testing the program's ability to save new valid IDs in database
Requirements Coverage	Successful-registration
Initial Condition	The login information is valid and can update the database
Procedure	1. The user clicks register 2. The user provides a valid user name 3. The user provides a valid password 4. The user logs-in into the system and is presented with the main UI page.
Expected Outcome	It shows you the main website page
Notes	Passwords and username should be at least 7 characters long. numbers and special characters all allowed

Test ID	Test1B
Category	Testing the programs ability to give error when invalid ID is entered
Requirements Coverage	Failed-registration

Initial Condition	The login ID information is invalid
Procedure	1. The user clicks register 2. The user provides a invalid user name 3. The user provides a valid password 4. The user stays on the registration page
Expected Outcome	It gives an error message
Notes	Password and username should be at least 7 characters long.

Test ID	Test1C
Category	Testing the programs ability to give error when invalid password is entered
Requirements Coverage	Failed-registration
Initial Condition	The login password information is invalid
Procedure	1. The user clicks register 2. The user provides a valid user name 3. The user provides a invalid password 4. The user stays on the registration page
Expected Outcome	It gives an error message
Notes	Password and username should be at least 7 characters long.

Test ID	Test1D
Category	Testing the programs ability to sign out
Requirements Coverage	Successful-Signing-out
Initial Condition	You are currently logged in.
Procedure	1. The user clicks logout button 2. We see the sign in/registration page.
Expected Outcome	User is back to the main sign in page
Notes	No notes.

Test Cases created by Hulya Yasar

Test ID	Test2A
Category	Checkout Process

Requirements Coverage	Successful-Checkout
Initial Condition	The user has items in the shopping cart and valid payment details.
Procedure	1. Click "Checkout" button 2. Enter valid shipping details 3. Enter valid credit card details 4. Click "Submit Payment"
Expected Outcome	The system displays an error: "Payment declined: Expired card."
Notes	The system should not allow payments with expired cards.

Test ID	Test2B
Category	Checkout Process
Requirements Coverage	Invalid-Payment
Initial Condition	The user has items in the shopping cart but enters an expired credit card.
Procedure	1. Click "Checkout" button 2. Enter shipping details 3. Enter expired credit card details 4. Click "Submit Payment"
Expected Outcome	The system displays an error: "Payment declined: Expired card."
Notes	The system should not allow payments with expired cards.

Test ID	Test2C
Category	Checkout Process
Requirements Coverage	Insufficient-Funds
Initial Condition	The user has items in the shopping cart but their credit card does not have enough balance.
Procedure	1. Click "Checkout" button 2. Enter shipping details 3. Enter credit card details with insufficient funds 4. Click "Submit Payment"
Expected Outcome	The system displays an error: "Payment declined: Insufficient funds."
Notes	Users should be able to try a different payment method after a failed attempt.

Test ID	Test2D
Category	Checkout Process
Requirements Coverage	Billing-Address-Error
Initial Condition	The user has items in the shopping cart but enters an incorrect billing address.

Procedure	1. Click "Checkout" button 2. Enter incorrect billing address 3. Enter valid credit card details 4. Click "Submit Payment"
Expected Outcome	The system displays an error: "Billing address does not match card details."
Notes	Users should be able to edit and retry their payment information.

Test ID	Test3A
Category	Running reports
Requirements Coverage	Vehicle-sales report
Initial Condition	The report gets made regardless of the number of sales
Procedure	1. The user clicks generate report which gives a drop down menu 2. The user selects the month they want to see the report of 3. The user then presses generate 4. The report gets automatically downloaded
Expected Outcome	The months report gets downloaded
Notes	If the month has no sales, the application will download a report with no data.

Test ID	Test3B
Category	Running reports
Requirements Coverage	Application-usage report
Initial Condition	The report gets made regardless of how much the application has been used.
Procedure	1. The user clicks generate report which gives a drop down menu 2. The user selects the month they want to see the report of 3. The user then presses generate 4. The report gets automatically downloaded
Expected Outcome	The months report gets downloaded
Notes	If the month has no application usage, the application will download the report with no data.

Test ID	Test4A
Category	List vehicles from catalogue

Test ID	Test4A
Requirements Coverage	catalogue-listing
Initial Condition	It will only show available cars.
Procedure	1. The user clicks on show cars button in the main page 2. The catalogue page opens and shows all the available cars
Expected Outcome	The catalogue page with cars show up
Notes	If the company has no available, no car will show up

Test cases by Adil

Test ID	Test5A
Category	Sort Vehicles by price
Requirements Coverage	User can sort the vehicles in the catalogue
Initial Condition	The user will see a random list of vehicles available
Procedure	1. User sees the available vehicle catalogue 2. User clicks on sort and selects whether to sort high-to-low or low-to-high
Expected Outcome	After sorting, the vehicles will show in the order that the user has selected
Notes	User should be able to sort the vehicles available by price

Test ID	Test5B
Category	Remove products from Shopping Cart
Requirements Coverage	User can remove products from their shopping cart
Initial Condition	User already has products in his cart
Procedure	1. User has product stored in the cart 2. User presses the remove button 3. The product has been removed from cart
Expected Outcome	The cart no longer has the product saved.

Test ID	Test5B
Notes	If the cart does not have any products at all, the delete button will not do anything.

Test ID	Test 5E
Category	add products from Shopping Cart
Requirements Coverage	User can add products from their shopping cart
Initial Condition	It does not matter how many products the cart has.
Procedure	1. User opens a product page. 2. User clicks on the “add to cart” button. 3. User now has an item in the cart.
Expected Outcome	The cart now contains the newly added item.
Notes	You can add new items to cart even if the cart already has an item.

Test ID	Test5C
Category	Chatbot
Requirements Coverage	User can ask chatbot questions without error
Initial Condition	User will be able to ask chatbot a question and get a response
Procedure	1. User is on the home page of the website 2. User clicks on the chatbot and enters a message/question 3. Chatbot responds with the proper message
Expected Outcome	Chatbot is able to efficiently answer the users questions
Notes	User should be able able to ask chatbot a question and receive a proper and helpful response

Test ID	Test5D
Category	Loan Calculator
Requirements Coverage	User can use the loan calculator to calculate the cost of the vehicle
Initial Condition	The user is presented with a screen showing the car chosen to view
Procedure	1. User clicks on a vehicle to view fully 2. User scrolls and clicks on loan calculator

Test ID	Test5D
	3. User enters valid input data and retrieves necessary data
Expected Outcome	User is able to enter valid input data and retrieve the data necessary to keep track of
Notes	User can use the loan calculator and keep track of their expense for the potential future vehicle

Test cases by Amir

Test ID	Test6D
Category	Write and Rate Reviews
Requirements Coverage	Allow users to write text reviews and give star ratings
Initial Condition	User is signed in as a registered customer User has previously purchased or viewed a particular vehicle
Procedure	1. The user navigates to the vehicle's detail or purchase history page. 2. The user clicks "Write a Review." 3. The user enters a review comment and selects a star rating (1 to 5). 4. The user clicks "Submit Review."
Expected Outcome	The review and rating are saved and displayed under the vehicle's review section. A confirmation message appears indicating the review was submitted successfully.
Notes	System should validate that the text review and star rating are provided. If missing, the system shows an error prompting the user to complete all fields.

Test ID	Test6C
Category	Select Vehicle Customization Options
Requirements Coverage	User can customize a chosen vehicle before adding it to the cart
Initial Condition	User is signed in as a registered customer User has opened the detail page of a vehicle that offers custom options (e.g., color, trim level)
Procedure	1. The user navigates to the vehicle's detail page. 2. The user selects desired customization options (e.g., color, interior, package). 3. The user clicks "Apply Customization."
Expected Outcome	The page updates to show the selected customization(s) and recalculates price if needed. The user can then add the customized vehicle to the cart.
Notes	If the customization is invalid (e.g., an unavailable color), the system should display an error or disable that option.

Test ID	Test6B
Category	Compare Vehicles
Requirements Coverage	Compare two or more vehicles' features and specs
Initial Condition	User is signed in as a registered customer At least two vehicles exist in the catalog
Procedure	1. The user navigates to the vehicle catalog page. 2. The user selects two (or more) vehicles by clicking a "Compare" checkbox or button. 3. The user clicks "Compare Selected Vehicles."
Expected Outcome	The system displays a comparison table showing vehicle specifications (price, mileage, features, etc.) side by side.
Notes	If fewer than two vehicles are selected, the system should prompt the user to select at least two.

Test ID	Test7A
Category	Filter Vehicles
Requirements Coverage	User can filter electric vehicles by brand, shape, model year, and vehicle history
Initial Condition	User is signed in as a registered customer Vehicle catalogue has a variety of electric vehicles with different brands, shapes, model years, and accident/damage histories
Procedure	1. The user navigates to the “Catalogue” or “Show Cars” page. 2. The user opens the filter panel (e.g., a sidebar or dropdown). 3. The user selects one or more filter criteria (e.g., Brand: “Tesla,” Shape: “SUV,” Model Year: “2023,” Vehicle History: “No reported damages”). 4. The user clicks “Apply Filter.”
Expected Outcome	The page displays only the vehicles that match **all** selected filter criteria. If no vehicles match, the page should display a “No vehicles found” message.
Notes	Filters should be applied cumulatively. The user should be able to clear or reset the filters to see **all** vehicles again.

Test ID	Test8A
Category	View Hot deals
Requirements Coverage	Viewing hot deals when there are none
Initial Condition	There are no hot deals available
Procedure	1. User is on the main page and clicks on view hot deals button. 2. User is given a prompt saying there are no hot deals available at the moment.
Expected Outcome	Given a prompt saying no there are no hot deals at the moment
Notes	Cars will be categorized and if there are no cars showing meaning no car has been put into that category.

Test ID	Test8B
Category	View Hot deals
Requirements Coverage	Viewing hot deals when they are available
Initial Condition	There are hot deals available
Procedure	1. User is on the main page and clicks on view hot deals button. 2. User is taken to a different webpage showing only the cars that are currently part of the hot deals.
Expected Outcome	Seeing a new page with cars that the company is giving a hot deal on.
Notes	Cars will be categorized and if cars are showing, it means the company is offering a hot deal on the car.

Test ID	Test 9A
Category	Viewing Car details
Requirements Coverage	Viewing extended car details
Initial Condition	The viewer can only see the basic car details
Procedure	1. User is on the car page but is only looking at the basic details in the information section and presses extend button 2. The information area opens up further and more car details are shown
Expected Outcome	Information area extends in size and displays more car details
Notes	Initially everyone will only be looking at the basic car details unless they extend it. It will also have the button to compress it.

Deliverable 2

Section A:

Design Patterns Used

Model-View-Controller (MVC):

Applied across the UI, backend services, and data management. React handles the view layer and MongoDB manages data models.

DAO (Data Access Object) Pattern:

Used for database operations to abstract and encapsulate all access to the data source, ensuring business logic does not directly interact with the database.

Singleton Pattern:

Applied in user session and authentication services to ensure a single shared instance is used across components.

Architectural Patterns

Layered Architecture:

The system is organized in multiple layers (Frontend, Controller, Service, Persistence). This separation of concerns increases maintainability and simplifies future development.

Quality Attributes

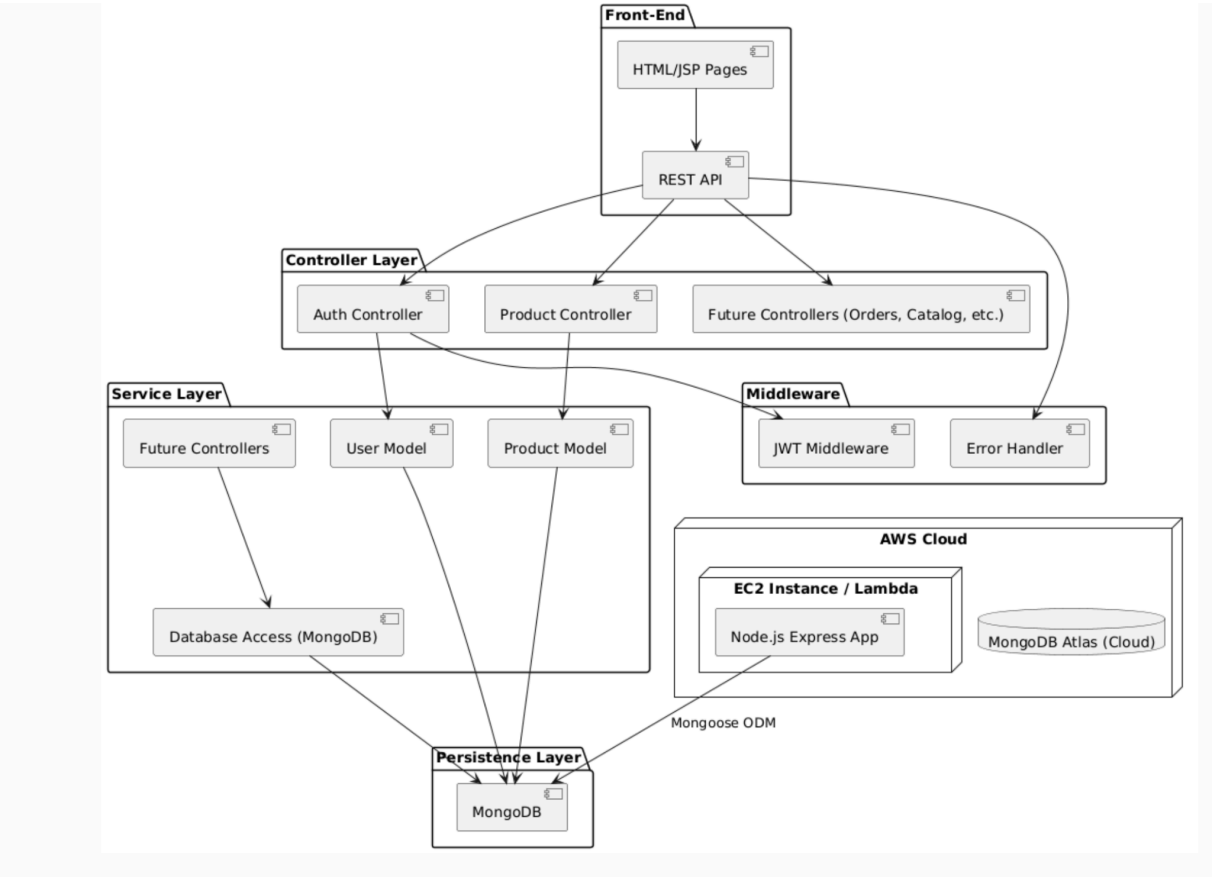
Security: JWT for secure authentication, HTTPS enforcement, role-based access in backend APIs

Performance: Stateless backend, optimized API endpoints, minimal coupling between services

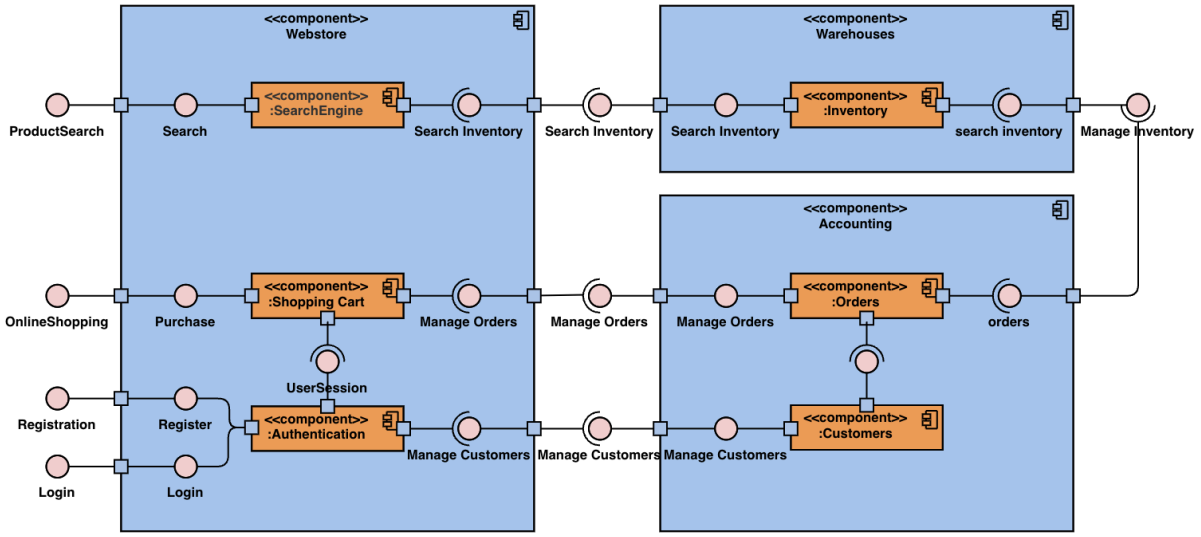
Maintainability: Modular codebase, layered architecture, use of design patterns

Availability: Database replication in MongoDB Atlas

Deployment Diagram



Component Diagram of Webstore



Section B:

Activity Plan and contribution

Taha Asim: Server environment setup, database deployment, login, user authorization, basic front end development and report writing

Adil Guluzade: Cart development and report writing

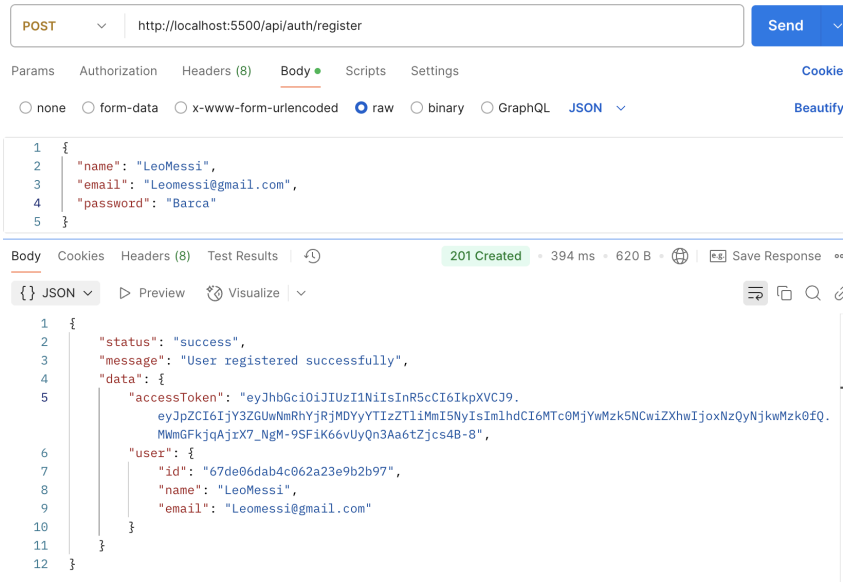
Amir Ahmadnasab: Product catalogue development and report writing

Hulya Yasar: Diagram design and report writing.

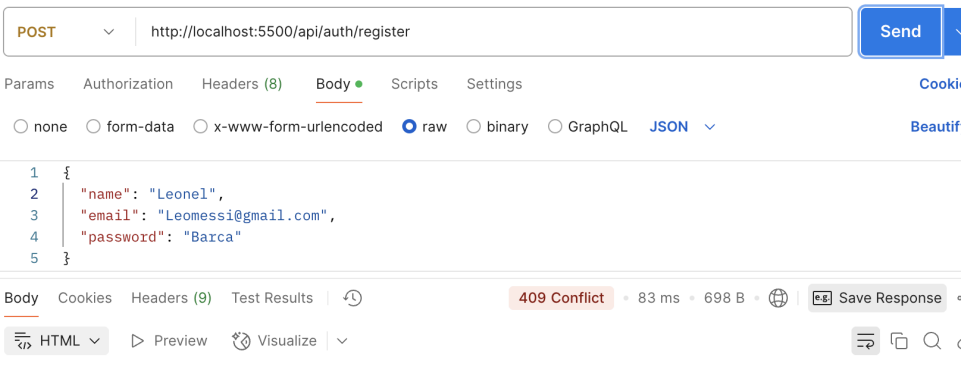
Meeting date	Meeting attendees	Meeting purpose
Feb 21, 2025	Hulya, Taha, Adil	Read through the deliverable requirements and understand what needs to be done.
Feb 27, 2025	Hulya, Taha Adil	Divide the deliverable requirements in parts and assign to each member equally. Discussed technologies and resources we need to use for the project.

Section C:

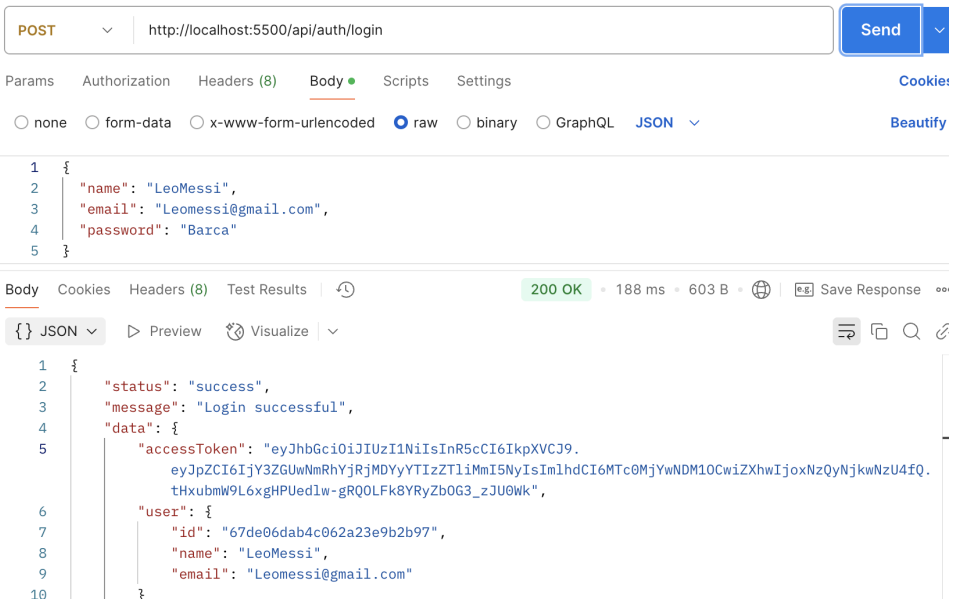
Test successful Registration:



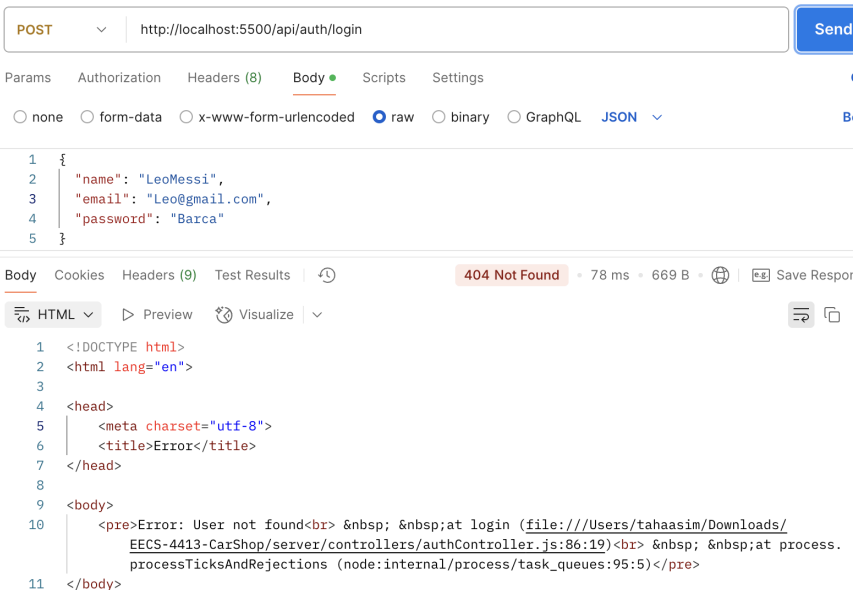
Test failed registration due the user email already being in use:

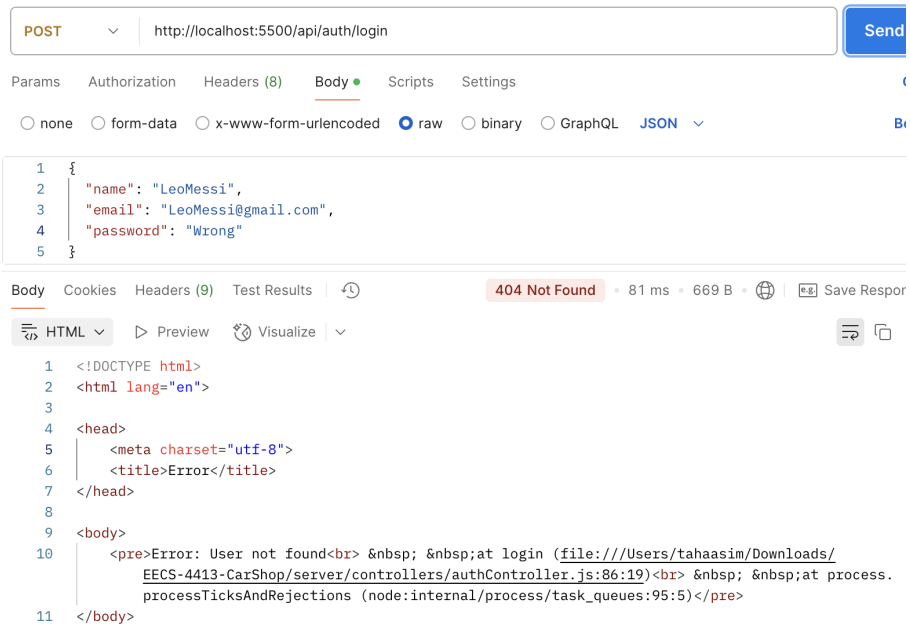


Test for successful login:

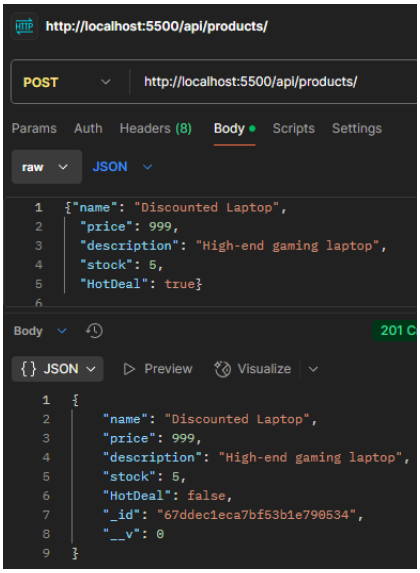


Test for failed login due to incorrect credentials:

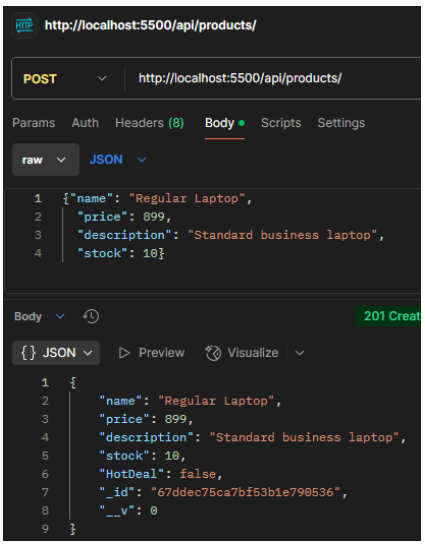




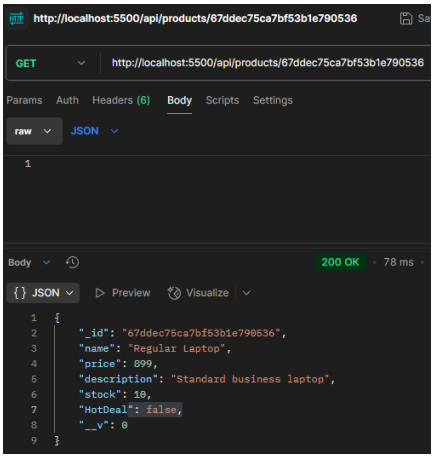
Test 1 Create Product:



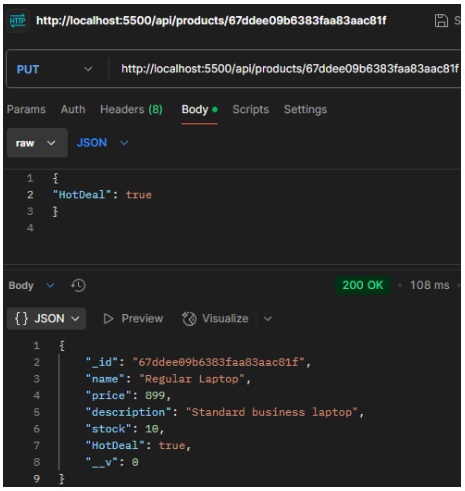
Test 2 Without HotDeal (Should Work):



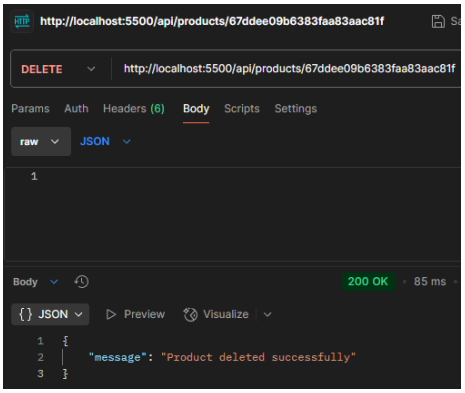
Test 3 Get Product by ID:



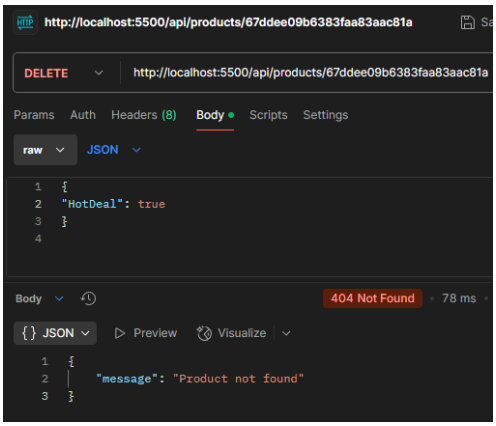
Test 4 Update Product:



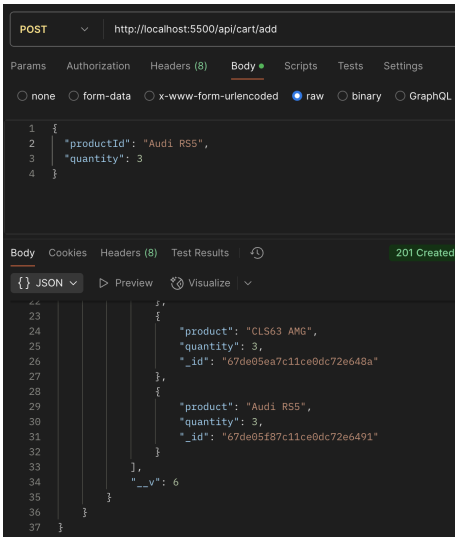
Test 5 Delete Product:



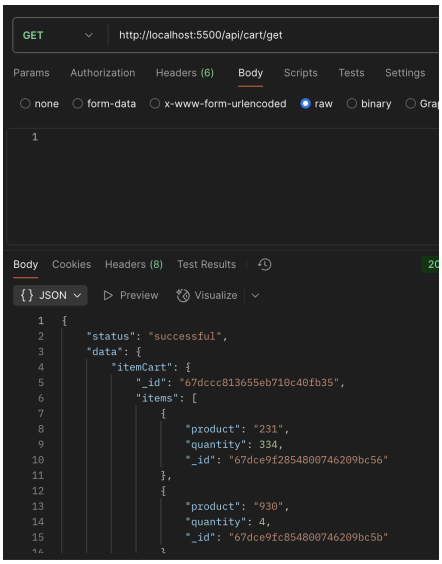
Test 5 Delete Unavailable Product:



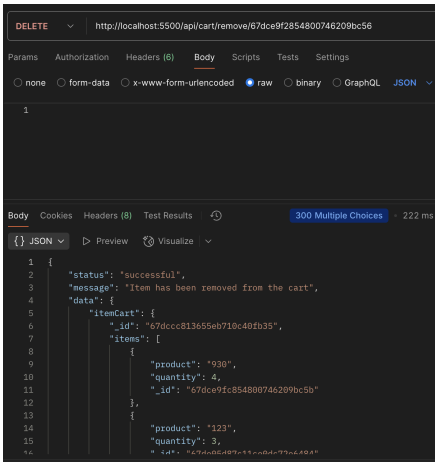
Test 6 Add a product:



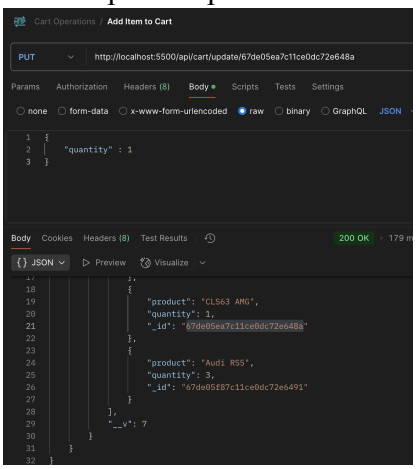
Test 7 Get the products:



Test 8 Remove a product:



Test 9 Update a product:



Section D:

When looking at the REST principles, we used them and implemented them deeply into our server side program to make sure our website stays as simple, easy to understand, scalable

and performs as smoothly as possible. When developing our program's server, we focused on keeping the server and client side of our application separate. We have a database layer that handles the applications connection to MongoDB, we have routes.js to define the API endpoints, JWT.js, a separate layer to take care of authentication. We ensured that all the necessary authentication details were present in every request when using the JWT tokens. We have consistent JSON responses to success and error in entries. We used standard HTTP methods like POST to authenticate and register new users.
We plan on deploying our application on the aws cloud services and we are using JavaScript primarily to build the backend of our application. To build a chatbot, we seeked online resources such as Jotform who provide services that can allow the programmer to build a simple but effective chatbot that can then be integrated into our application.