



LeafWheels v0.6





Design Document

Version:	0.6
Print Date:	June 13, 2025
Release Date:	June 13, 2025
Release State:	Draft
Approval State:	Distributed
Approved by:	Prabhjot Singh, Manali Bisht Tarang Patel, Zhenxu Wang
Prepared by:	Prabhjot Singh, Manali Bisht Tarang Patel, Zhenxu Wang
Reviewed by:	Prabhjot Singh, Manali Bisht Tarang Patel, Zhenxu Wang
Path Name:	leafwheels/
File Name:	leafwheels-design-document
Document No:	6

Document Change Control

Version	Date	Authors	Summary of Changes
1	May 19, 2025	Manali Bisht	Create initial design document
2	June 3, 2025	Prabhjot Singh,Manali Bisht Tarang Patel, Zhenxu Wang	Major design decision made.
3	June 9, 2025	Prabhjot Singh,Manali Bisht Tarang Patel, Zhenxu Wang	Add initial use case diagram, add initial sequence diagram, add scrum planning.
4	June 11, 2025	Manali Bisht, Tarang Patel	Add updated use case diagram.
5	June 12, 2025	Prabhjot Singh,Manali Bisht Tarang Patel, Zhenxu Wang	Add updated sequence diagrams, add updated architecture diagrams, add TTD development tables.
6	June 13, 2025	Prabhjot Singh,Manali Bisht Tarang Patel, Zhenxu Wang	Finalize design document.

Document Sign-Off

Name (Position)	Signature	Date
Prabhjot Singh- 219971126		June 13, 2025
Manali Bisht-219241009		June 13, 2025
Tarang Patel-220175758		June 13, 2025
Zhenxu Wang-218587295		June 13, 2025

Contents

1 Introduction..... 4

1.1 Purpose..... 4

1.2 Overview..... 4

1.3 Resources - References..... 4

2 Major Design Decisions..... 5

3 Use case Diagram..... 6

4 Sequence Diagrams..... 7

5 Architecture..... 10

6 Activities Plan..... 14

6.1 Project Backlog and Sprint Backlog..... 14

6.2 Group Meeting Logs.....21

7 Test Driven Development..... 22

1 Introduction

1.1 Purpose

This document presents the design of **LeafWheels**, an e-commerce web application that allows users to browse, compare, and purchase electric vehicles online. The system supports core customer features such as vehicle catalog browsing, filtering, loan calculation, account management, checkout, and post-purchase tracking. Administrative functions like review moderation and analytics are also included. It describes the system's intended functionality, user interactions, modular architecture, and verification approach using test-driven development. As a small-scale project, this document contains the full set of design elements necessary to guide development and ensure the system meets all functional and non-functional requirements.

1.2 Overview

This document is organized to present a structured and comprehensive design of the LeafWheels application. It begins with a high-level summary of major design decisions, followed by UML diagrams for use cases and system interactions. Architecture is presented through modular decomposition and interface definitions. Agile development planning is described through the product backlog, sprint backlog, Gantt chart, and meeting logs. Finally, a full set of test cases is provided to support test-driven development and verify requirement coverage.

1.3 Resources - References

- [Jira Project Board](#)
- EECS4413 Team Project Specification - on [eclass](#)
- EECS4413 Deliverable 1 Instructions PDF - on [eclass](#)

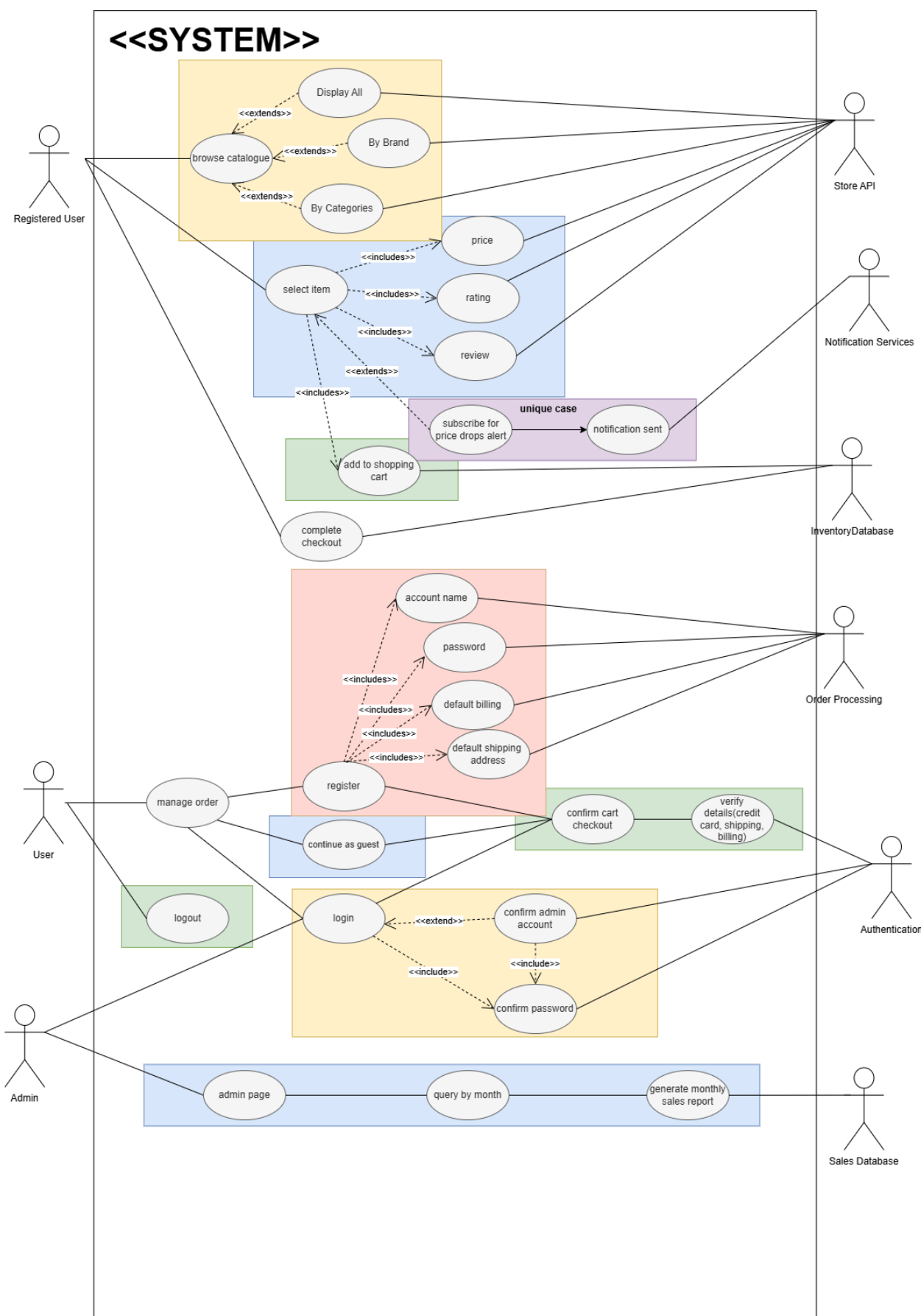
2 Major Design Decisions

Decision to Make	Decision	Notes
Tools & Technologies		
Frontend Framework	React.js	
Backend Framework	Spring Boot	
Dependency Management	Maven, npm, yarn	
Frontend-backend communication	HTTP through REST API for majority of the features	Websocket for chatbot
SQL Database	MySQL	H2 DB for development
NoSQL Database	MongoDB, Redis	May not need NoSQL
File Storage	Cloud provider's bucket	
VCS	GitHub	Repo link here
Containerization	Docker	
Deployment	Cloud provider's container orchestration	
Task Distribution	Jira	Scrum board link here
Notification System	Cloud provider's pub/sub	
Architectural Design		
System Architecture	MVC	
View Layer	React based web app	
Controller Layer	Spring Boot RESTful API	
Model Layer	Service and DAO components	
Design Patterns		
Database Interaction	DAO, Spring Boot JPA as ORM	Data Access Object
Object Creation & DI	Singleton, Factory	
Notification	Publisher/Subscriber	
Controller	Facade	
Other Design Decisions		
Modularization Criteria	<p>Each module should only do one distinct responsibility. Communication between modules should use REST API for low coupling (except Chatbot using Websocket with external API).</p> <p>We have divided our service components based on single-responsibility principle and utilize Spring Boot feature to provide a high cohesion, low coupling and robusting system</p>	
Scalability & Maintainability	<p>We plan to deploy our backend container separately from the frontend component, so that backend modules can scale independently. Cloud container orchestration will provide horizontal scaling, Kubernetes can be employed if needed in future.</p> <p>API versioning strategy is considered for future backward compatibility.</p>	

3 Use case Diagram

Provide a use case diagram depicting the features your e-commerce system will support. You should have at least six use cases in your use case diagram. You should have a maximum of 20 use cases in your use case diagram.

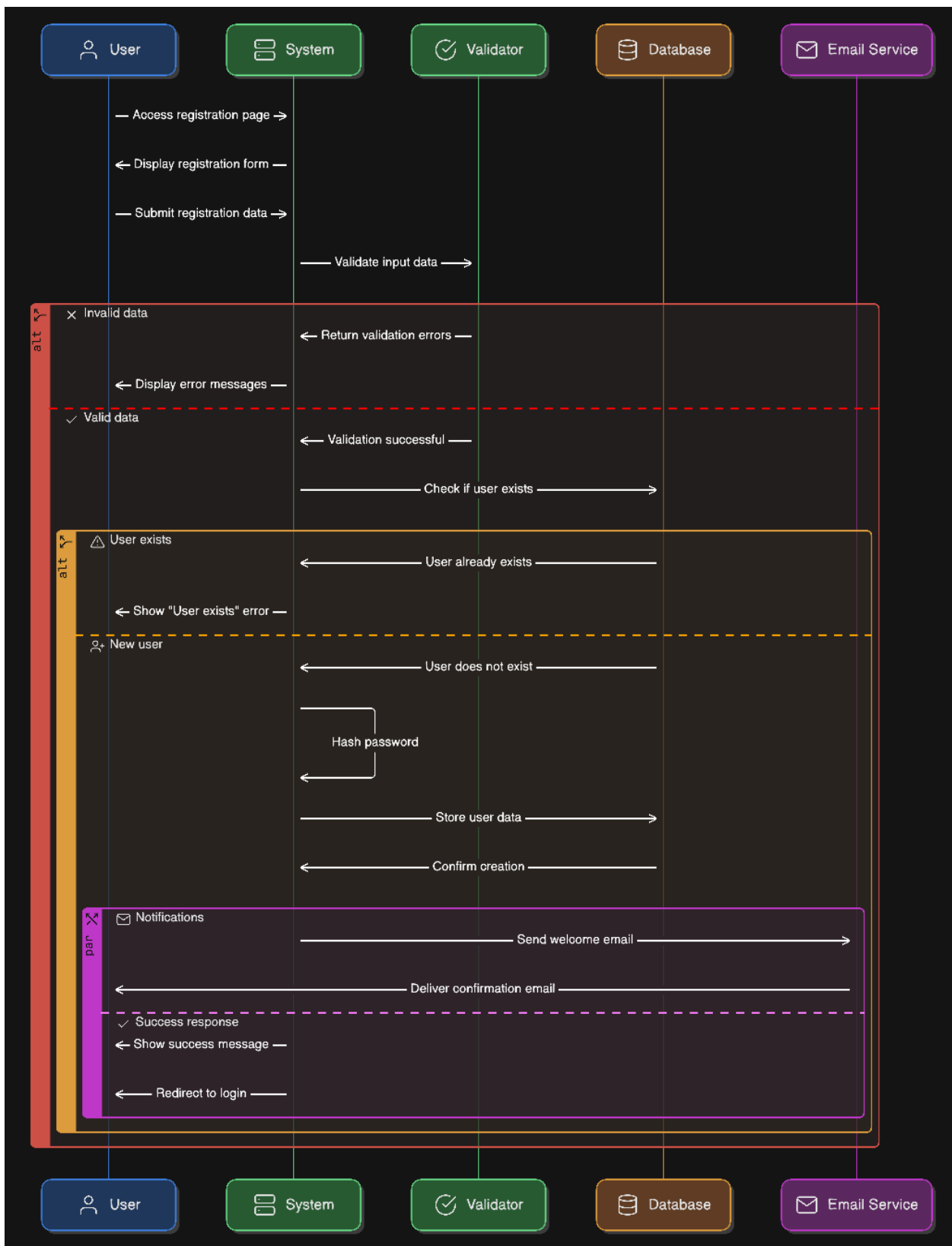
Link:
https://cdn.discordapp.com/attachments/1247369225505341453/1383143712195870790/Untitled_Diagram-Page-1.drawio.png?ex=684db876&is=684c66f6&hm=0a40f73e249a455afae9fd7e467fb900f39add1dfc61c1a2d926f507e9d28c0f



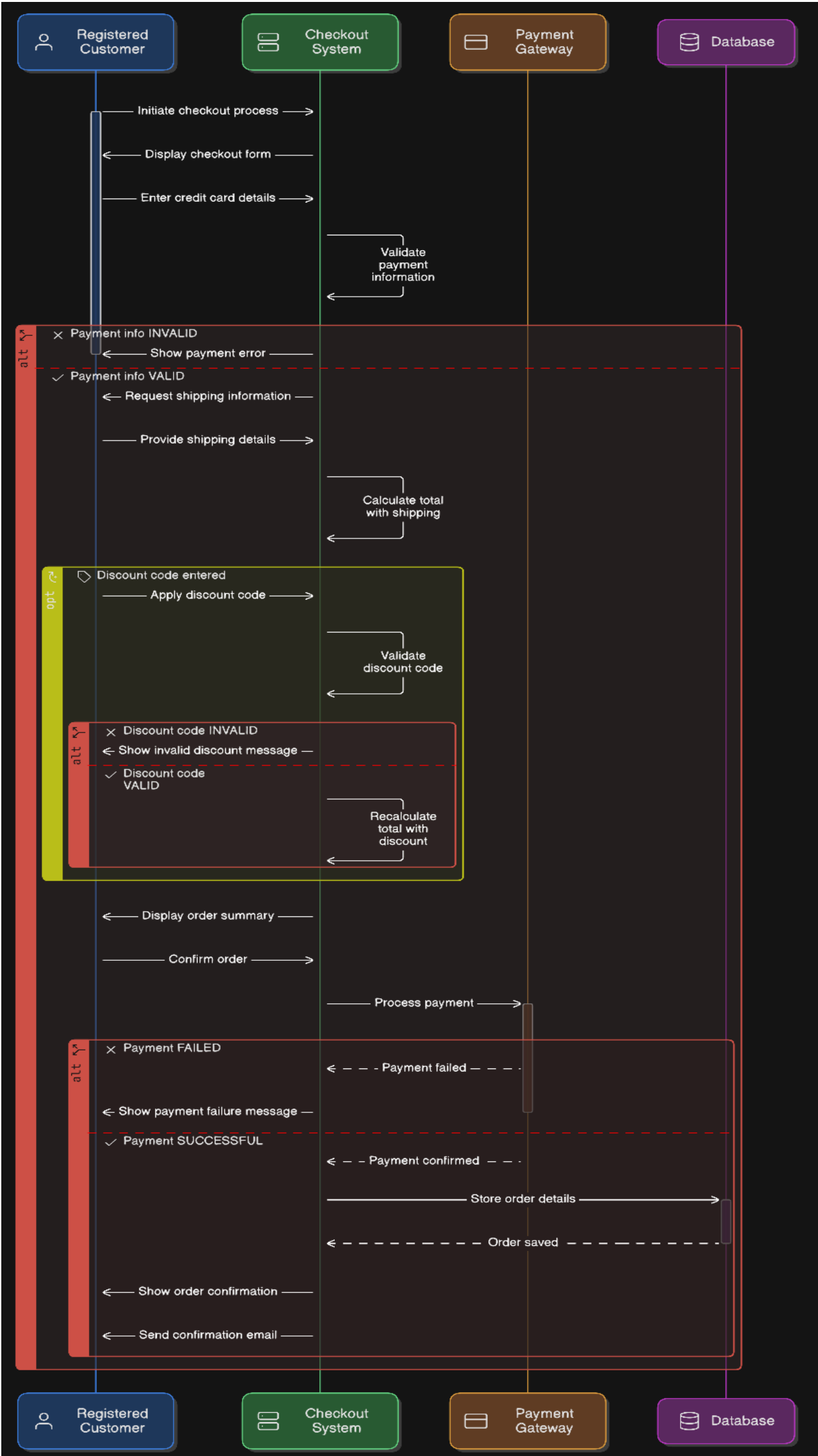
4 Sequence Diagrams

Pick three use cases depicted in your use case diagram and provide the corresponding three sequence diagrams. One of the three use cases should describe your original use case (i.e., distinguished use case). Make sure you identify and associate each sequence diagram with the proper use case by maintaining unique Identifiers for use cases.

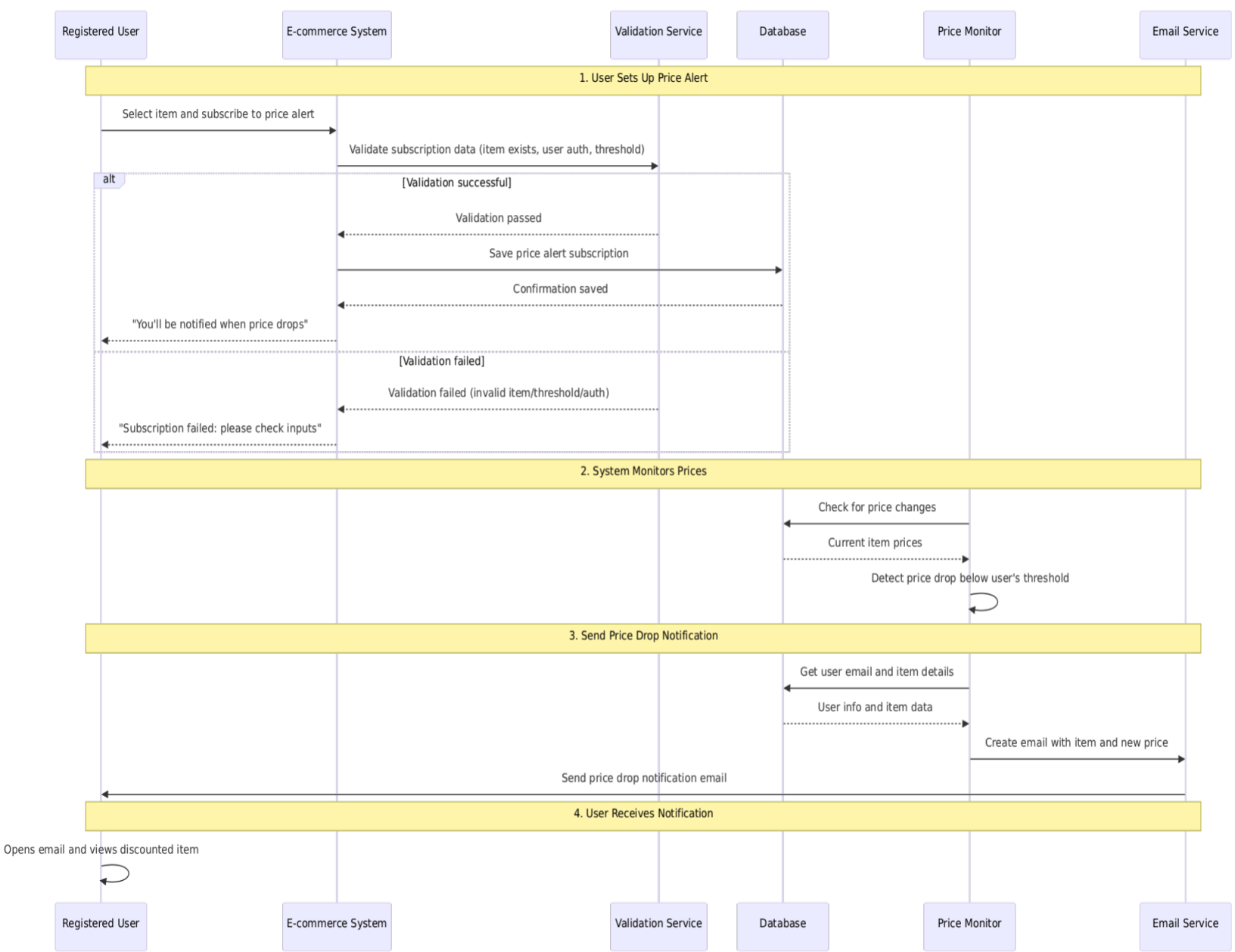
Register Use case Sequence Diagram:



Complete Checkout Process:

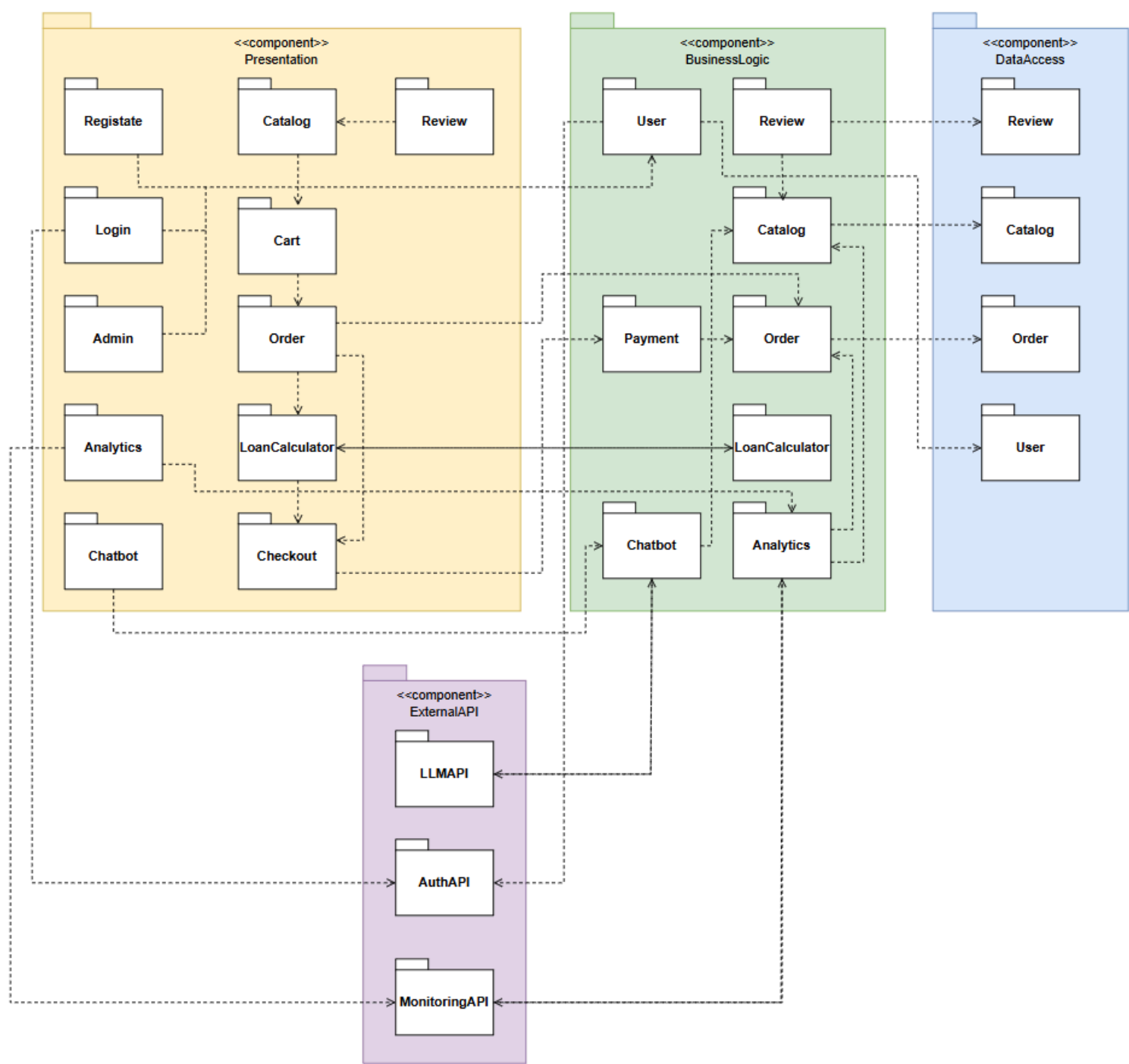


Subscribe to Price Drop alerts (Original Use Case) sequence Diagram:

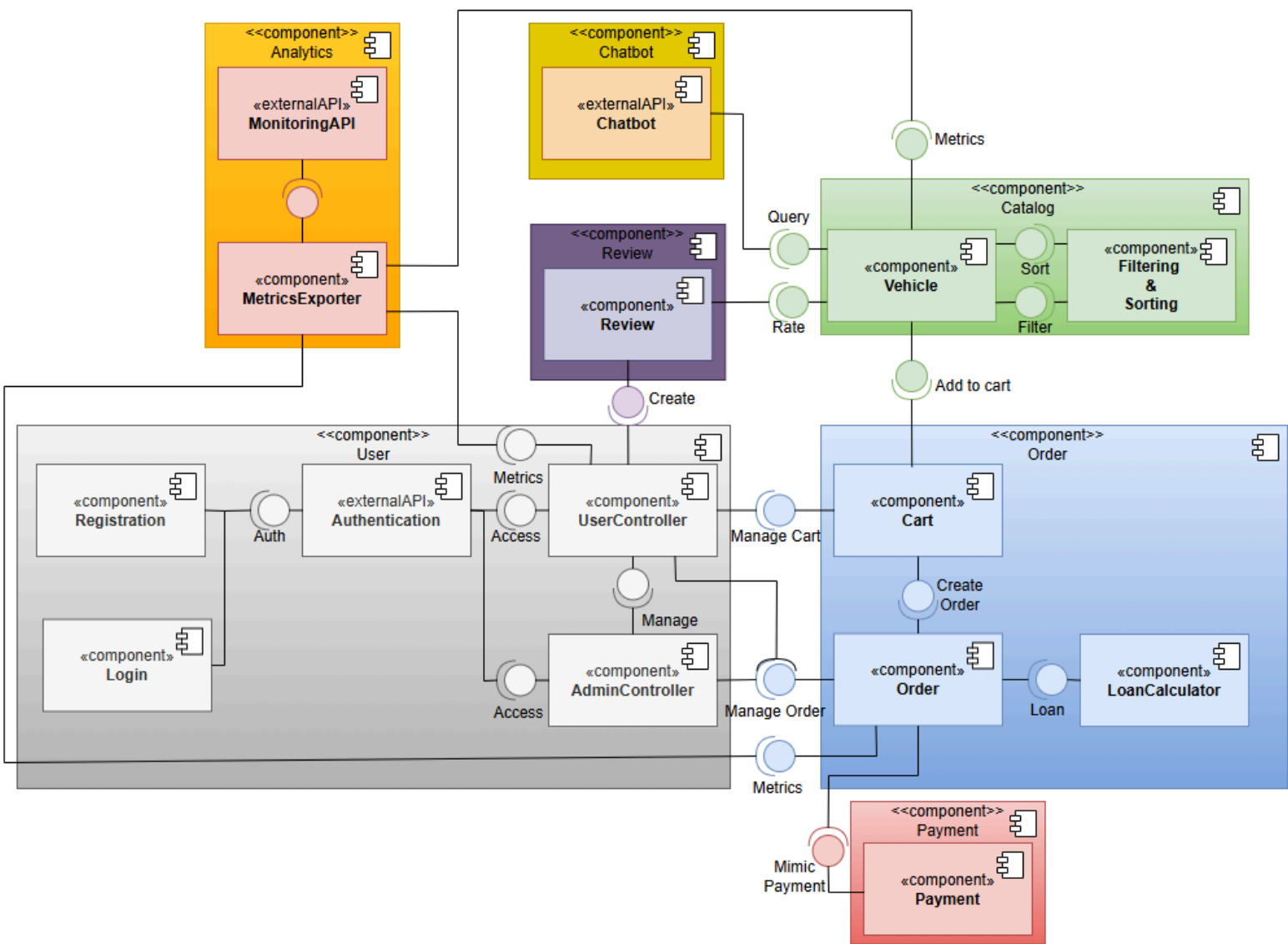


5 Architecture

Decomposition of the System



Component Diagram



Modules			
Module Name	Description	Exposed Interface Names	Interface Description
Catalog	Store and manage vehicle information.	Catalog:ReviewInterface	Catalog:ReviewInterface pull up user review and star rating.
User	Create, authenticate and store user information.	User:AnalyticsAPI User:ReviewAPI	User:AnalyticsAPI admin user access analytics data. User:ReviewAPI user create and admin user delete reviews and star rating.
Order	Handle vehicle order.	Order:CartAPI Order:PaymentAPI	Order:CartAPI generate order based on cart. Order:PaymentAPI mimic payment process base on order.

Interfaces		
Interface Name	Operations	Operation Descriptions
Catalog:ReviewInterface	<Review> ReviewInterface: getReview(UUID)	Catalog:ReviewInterface getReview(UUID): Extract Review by vehicle's UUID.
User:AnalyticsInterface	<Stats> AnalyticsInterface: getAnalytics() getAnalytics(Timestamp, Timestamp)	User:AnalyticsInterface getAnalytics(), getAnalytics(Timestamp, Timestamp): Extract Stats of default (last 30 days) or given time duration
User:ReviewInterface	<boolean> ReviewInterface: addReview(Review) <boolean> ReviewInterface deleteReviewById(UUID)	User:ReviewInterface addReview(Review): Store user's input review, return true if addition success.. User:ReviewInterface deleteReviewById(UUID): Admin user delete a review by UUID, return true if deletion success.

Order:CartInterface	<Order>CartInterface: addOrder(Order)	Order:CartInterface addOrder(Order): Store user's input order, return true if addition success.
Order:PaymentInterface	<Boolean> PaymentInterface: pay(Order)	Order:PaymentInterface pay(Order): Mimic the payment process of given order, return true if payment success.

6 Activities Plan

6.1 Project Backlog and Sprint Backlog

Project backlog: also available [here](#) on Jira.

Issue key	Summary	Assignee	Reporter	Status	Sprint
EVWEB-6	Finalize Deliverable 1 Document		Mason Wang	To Do	Deliverable 1 Sprint 1
EVWEB-7	Deliverable 1 - Introduction		Mason Wang	To Do	Deliverable 1 Sprint 1
EVWEB-8	Deliverable 1 - Major Design Decisions		Mason Wang	To Do	Deliverable 1 Sprint 1
EVWEB-9	Deliverable 1 - Use Case Diagram 1	Manali Bisht	Mason Wang	In Review	Deliverable 1 Sprint 1
EVWEB-10	Deliverable 1 - Use Case Diagram 2	Prabhjot Singh	Mason Wang	In Progress	Deliverable 1 Sprint 1
EVWEB-11	Deliverable 1 - Use Case Diagram 3	Tarang	Mason Wang	To Do	Deliverable 1 Sprint 1
EVWEB-12	Deliverable 1 - Sequence Diagram 1	Tarang	Mason Wang	In Review	Deliverable 1 Sprint 1
EVWEB-13	Deliverable 1 - Sequence Diagram 2	Tarang	Mason Wang	In Review	Deliverable 1 Sprint 1
EVWEB-14	Deliverable 1 - Sequence Diagram 3 (original use case)		Mason Wang	To Do	Deliverable 1 Sprint 1
EVWEB-15	Deliverable 1 - Architecture	Mason Wang	Mason Wang	Done	Deliverable 1 Sprint 1
EVWEB-16	Deliverable 1 - 6.1 Project Backlog and Sprint Backlog Items	Mason Wang	Mason Wang	Done	Deliverable 1 Sprint 1
EVWEB-17	Deliverable 1 - 6.1 Gantt Chart	Mason Wang	Mason Wang	In Progress	Deliverable 1 Sprint 1
EVWEB-19	Deliverable 1 - 6.2 Group Meeting logs		Mason Wang	To Do	Deliverable 1 Sprint 1
EVWEB-20	Delicerable 1 - TDD test case 1 to 4	Mason Wang	Mason Wang	Done	Deliverable 1 Sprint 1
EVWEB-21	Delicerable 1 - TDD test case 5 to 8	Prabhjot Singh	Mason Wang	In Progress	Deliverable 1 Sprint 1
EVWEB-22	Delicerable 1 - TDD test case 9 to 12	Tarang	Mason Wang	To Do	Deliverable 1 Sprint 1
EVWEB-23	Delicerable 1 - TDD test case 13 to 16	Manali Bisht	Mason Wang	To Do	Deliverable 1 Sprint 1
EVWEB-33	GH Repo Setup	Mason Wang	Mason Wang	Done	Deliverable 1 Sprint 1

EVWEB-38	Deliverable 1 - Use Case Diagram 4		Mason Wang	To Do	Deliverable 1 Sprint 1
EVWEB-39	Deliverable 1 - Use Case Diagram 5		Mason Wang	To Do	Deliverable 1 Sprint 1
EVWEB-40	Deliverable 1 - Use Case Diagram 6 (original use case)		Mason Wang	To Do	Deliverable 1 Sprint 1
EVWEB-34	Catalog @Entity		Mason Wang	To Do	Deliverable 2 Sprint 1
EVWEB-35	Catalog @Repository		Mason Wang	To Do	Deliverable 2 Sprint 1
EVWEB-36	Catalog @Service		Mason Wang	To Do	Deliverable 2 Sprint 1
EVWEB-37	Catalog @Controller		Mason Wang	To Do	Deliverable 2 Sprint 1
EVWEB-41	Catalog Overall Design		Mason Wang	To Do	Deliverable 2 Sprint 1
EVWEB-42	Catalog Domain Design		Mason Wang	To Do	Deliverable 2 Sprint 1
EVWEB-43	User Overall Design		Mason Wang	To Do	Deliverable 2 Sprint 1
EVWEB-44	User Domain Design		Mason Wang	To Do	Deliverable 2 Sprint 1
EVWEB-45	User @Entity		Mason Wang	To Do	Deliverable 2 Sprint 1
EVWEB-46	User @Repository		Mason Wang	To Do	Deliverable 2 Sprint 1
EVWEB-47	User @Service		Mason Wang	To Do	Deliverable 2 Sprint 1
EVWEB-48	User @Controller		Mason Wang	To Do	Deliverable 2 Sprint 1
EVWEB-49	Review Overall Design		Mason Wang	To Do	Deliverable 2 Sprint 1
EVWEB-50	Review Domain Design		Mason Wang	To Do	Deliverable 2 Sprint 1
EVWEB-51	Review @Entity		Mason Wang	To Do	Deliverable 2 Sprint 1
EVWEB-52	Review @Repository		Mason Wang	To Do	Deliverable 2 Sprint 1
EVWEB-53	Review @Service		Mason Wang	To Do	Deliverable 2 Sprint 1
EVWEB-54	Review @Controller		Mason Wang	To Do	Deliverable 2 Sprint 1
EVWEB-55	Analytics Service Overall Design Desicion		Mason Wang	To Do	Deliverable 2 Sprint 1

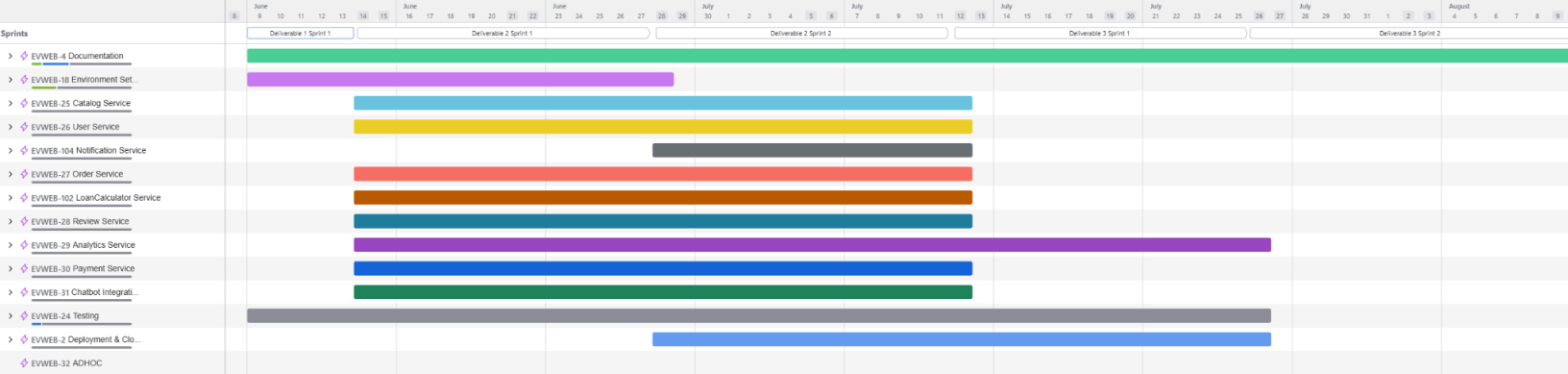
EVWEB-56	Payment Overall Design		Mason Wang	To Do	Deliverable 2 Sprint 1
EVWEB-57	Payment @Service		Mason Wang	To Do	Deliverable 2 Sprint 1
EVWEB-58	Payment @Controller		Mason Wang	To Do	Deliverable 2 Sprint 1
EVWEB-59	Chatbot Design Desicion		Mason Wang	To Do	Deliverable 2 Sprint 1
EVWEB-64	Order Overall Design		Mason Wang	To Do	Deliverable 2 Sprint 1
EVWEB-65	Order Domain Design		Mason Wang	To Do	Deliverable 2 Sprint 1
EVWEB-66	Order @Entity		Mason Wang	To Do	Deliverable 2 Sprint 1
EVWEB-67	Order @Repository		Mason Wang	To Do	Deliverable 2 Sprint 1
EVWEB-68	Order @Service		Mason Wang	To Do	Deliverable 2 Sprint 1
EVWEB-69	Order @Controller		Mason Wang	To Do	Deliverable 2 Sprint 1
EVWEB-70	Catalog Overview Initial UI		Mason Wang	To Do	Deliverable 2 Sprint 1
EVWEB-71	User Login Initial UI		Mason Wang	To Do	Deliverable 2 Sprint 1
EVWEB-73	Catalog - Single Vehicle Detail UI		Mason Wang	To Do	Deliverable 2 Sprint 1
EVWEB-76	Payment Initial UI		Mason Wang	To Do	Deliverable 2 Sprint 1
EVWEB-77	Order - Order Creation UI		Mason Wang	To Do	Deliverable 2 Sprint 1
EVWEB-79	Catalog - Springboot Components Unit Test		Mason Wang	To Do	Deliverable 2 Sprint 1
EVWEB-80	User - Springboot Components Unit Test		Mason Wang	To Do	Deliverable 2 Sprint 1
EVWEB-81	Review - Springboot Components Unit Test		Mason Wang	To Do	Deliverable 2 Sprint 1
EVWEB-82	Payment - Springboot Components Unit Test		Mason Wang	To Do	Deliverable 2 Sprint 1
EVWEB-83	Order - Springboot Components Unit Test		Mason Wang	To Do	Deliverable 2 Sprint 1
EVWEB-84	Catalog Service Layer Unit Test		Mason Wang	To Do	Deliverable 2 Sprint 1
EVWEB-85	User Service Layer Unit Test		Mason Wang	To Do	Deliverable 2 Sprint 1

EVWEB-86	Review Service Layer Unit Test		Mason Wang	To Do	Deliverable 2 Sprint 1
EVWEB-87	Payment Service Layer Unit Test		Mason Wang	To Do	Deliverable 2 Sprint 1
EVWEB-88	Order Service Layer Unit Test		Mason Wang	To Do	Deliverable 2 Sprint 1
EVWEB-97	LoanCalculator @Service		Mason Wang	To Do	Deliverable 2 Sprint 1
EVWEB-98	LoanCalculator @Controller		Mason Wang	To Do	Deliverable 2 Sprint 1
EVWEB-100	LoanCalculator - Springboot Components Unit Test		Mason Wang	To Do	Deliverable 2 Sprint 1
EVWEB-103	LoanCalculator Service Layer Unit Test		Mason Wang	To Do	Deliverable 2 Sprint 1
EVWEB-128	GH - Frontend Environment Setup		Mason Wang	To Do	Deliverable 2 Sprint 1
EVWEB-129	Backend Environment Onboarding		Mason Wang	To Do	Deliverable 2 Sprint 1
EVWEB-130	Frontend Environment Onboarding		Mason Wang	To Do	Deliverable 2 Sprint 1
EVWEB-60	Chatbot Backend Integration		Mason Wang	To Do	Deliverable 2 Sprint 2
EVWEB-61	Chatbot Initial UI Component		Mason Wang	To Do	Deliverable 2 Sprint 2
EVWEB-62	Docker Setup		Mason Wang	To Do	Deliverable 2 Sprint 2
EVWEB-63	Cloud Provider Choices & Desicion		Mason Wang	To Do	Deliverable 2 Sprint 2
EVWEB-72	Review Addition Initial UI		Mason Wang	To Do	Deliverable 2 Sprint 2
EVWEB-74	Catalog - Compare Vehicles UI		Mason Wang	To Do	Deliverable 2 Sprint 2
EVWEB-75	Catalog - Filter & Sort UI Components		Mason Wang	To Do	Deliverable 2 Sprint 2
EVWEB-78	Order - Order Management UI		Mason Wang	To Do	Deliverable 2 Sprint 2
EVWEB-89	User/Review Integration Test		Mason Wang	To Do	Deliverable 2 Sprint 2
EVWEB-90	User/Order Integration Test		Mason Wang	To Do	Deliverable 2 Sprint 2
EVWEB-91	User/Payment/Order Integration Test		Mason Wang	To Do	Deliverable 2 Sprint 2
EVWEB-92	Finalize Deliverable 2 Document		Mason Wang	To Do	Deliverable 2 Sprint 2

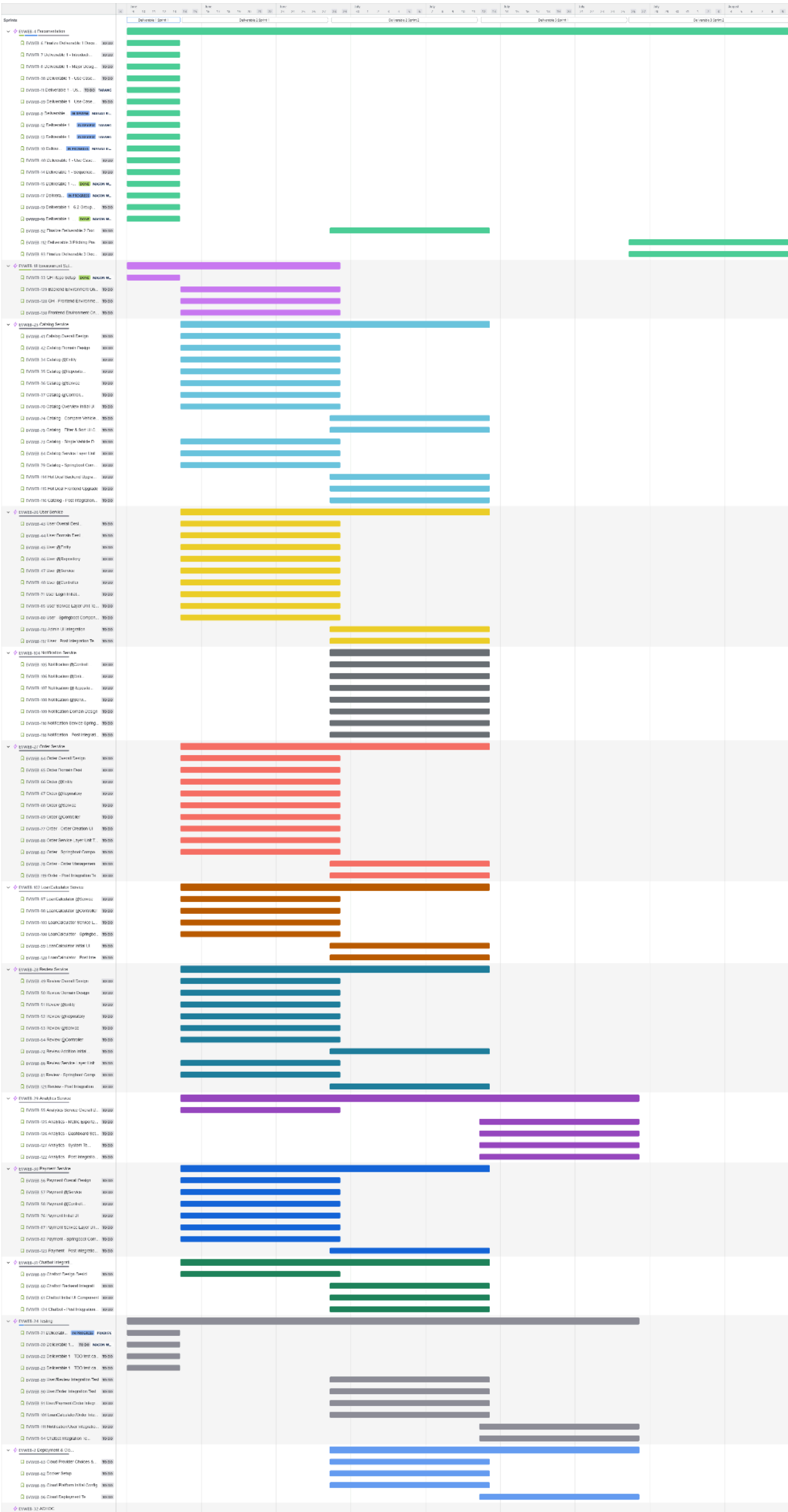
EVWEB-95	Cloud Platform Initial Config		Mason Wang	To Do	Deliverable 2 Sprint 2
EVWEB-99	LoanCalculator Initial UI		Mason Wang	To Do	Deliverable 2 Sprint 2
EVWEB-101	LoanCalculator/Order Integration Test		Mason Wang	To Do	Deliverable 2 Sprint 2
EVWEB-105	Notification @Controller		Mason Wang	To Do	Deliverable 2 Sprint 2
EVWEB-106	Notification @Entity		Mason Wang	To Do	Deliverable 2 Sprint 2
EVWEB-107	Notification @Repository		Mason Wang	To Do	Deliverable 2 Sprint 2
EVWEB-108	Notification @Service		Mason Wang	To Do	Deliverable 2 Sprint 2
EVWEB-109	Notification Domain Design		Mason Wang	To Do	Deliverable 2 Sprint 2
EVWEB-110	Notification Service Springboot Components Unit Test		Mason Wang	To Do	Deliverable 2 Sprint 2
EVWEB-113	Admin UI Integration		Mason Wang	To Do	Deliverable 2 Sprint 2
EVWEB-114	Hot Deal Backend Upgrade		Mason Wang	To Do	Deliverable 2 Sprint 2
EVWEB-115	Hot Deal Frontend Upgrade		Mason Wang	To Do	Deliverable 2 Sprint 2
EVWEB-116	Catalog - Post Integration Test Upgrade Plan		Mason Wang	To Do	Deliverable 2 Sprint 2
EVWEB-117	User - Post Integration Test Upgrade Plan		Mason Wang	To Do	Deliverable 2 Sprint 2
EVWEB-118	Notification - Post Integration Test Upgrade Plan		Mason Wang	To Do	Deliverable 2 Sprint 2
EVWEB-119	Order - Post Integration Test Upgrade Plan		Mason Wang	To Do	Deliverable 2 Sprint 2
EVWEB-120	LoanCalculator - Post Integration Test Upgrade Plan		Mason Wang	To Do	Deliverable 2 Sprint 2
EVWEB-121	Review - Post Integration Test Upgrade Plan		Mason Wang	To Do	Deliverable 2 Sprint 2
EVWEB-123	Payment - Post Integration Test Upgrade Plan		Mason Wang	To Do	Deliverable 2 Sprint 2
EVWEB-124	Chatbot - Post Integration Test Upgrade Plan		Mason Wang	To Do	Deliverable 2 Sprint 2
EVWEB-94	Chatbot Integration Test		Mason Wang	To Do	Deliverable 3 Sprint 1

EVWEB-96	Cloud Deployment Test		Mason Wang	To Do	Deliverable 3 Sprint 1
EVWEB-111	Notification/User Integration Test		Mason Wang	To Do	Deliverable 3 Sprint 1
EVWEB-122	Analytics - Post Integration Test Upgrade Plan		Mason Wang	To Do	Deliverable 3 Sprint 1
EVWEB-125	Analytics - Metric Exporter Integration		Mason Wang	To Do	Deliverable 3 Sprint 1
EVWEB-126	Analytics - Dashboard Setup		Mason Wang	To Do	Deliverable 3 Sprint 1
EVWEB-127	Analytics - System Test		Mason Wang	To Do	Deliverable 3 Sprint 1
EVWEB-93	Finalize Deliverable 3 Document		Mason Wang	To Do	Deliverable 3 Sprint 2
EVWEB-112	Deliverable 3 Pitching Preparation		Mason Wang	To Do	Deliverable 3 Sprint 2

Gantt chart for Epics: (available [here](#) for a clearer view)



Gantt chart for all stories: (available [here](#) for a clearer view)



6.2 Group Meeting Logs

Present Group Members	Meeting Date	Issues Discussed / Resolved
All	May 20, 2025	<ul style="list-style-type: none">- Going through document requirements, divided tasks- Discussed system features and agreed on using an MVC-based designy- Start gantt chart diagram on jira Task Assignments: <ul style="list-style-type: none">- Mason: test cases, and Architecture- Manali: Use Case Diagrams and test cases- Tarang: Sequence Diagrams and test cases- Prabhjot: Use Case Diagrams and test cases
All	May 29, 2025	<ul style="list-style-type: none">- Confirmed diagrams are sufficient- Reviewed progress on diagrams and architecture- Finalized the original use case- Individual work on test cases
All	June 9, 2025	<ul style="list-style-type: none">- Final touch ups and report revising- Confirmed all test cases and diagrams are complete- Reviewed final report structure for submission

7 Test Driven Development

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

Test ID	T01
Category	User – User login
Requirements Coverage	User-AuthenticationAPI-01
Initial Condition	User is on the login page
Procedure	<div>1. The user clicks “SSO” login</div> <div>2. The user is redirected to SSO provider</div> <div>3. The user select SSO login account with SSO provider</div>
Expected Outcome	SSO provider redirects user back to home page, user logged in, with access to order page and cart
Notes	System should create and store the user’s profile if the SSO profile has never logged into the app

Test ID	T02
Category	Payment - Process payment
Requirements Coverage	Payment-pay-success-01
Initial Condition	User has at least one in-stock vehicle item in cart
Procedure	<div>1. The user clicks “Checkout” button</div> <div>2. The user enters their credit card and billing address information on the checkout page</div> <div>3. The user clicks the "Pay" button</div>
Expected Outcome	User is redirected to payment success page, with order number and confirmation number on the page
Notes	Credit card number must under valid credit card number format Billing address is required to proceed the payment Since this payment system mimics real payment, 1⁄3 of the transactions will be marked as <i>failed</i> by the system

Test ID	T03
Category	Payment - Process payment
Requirements Coverage	Payment-pay-fail-not-enough-balance
Initial Condition	User has at least one in-stock vehicle item in cart

Procedure	<ol style="list-style-type: none"> 1. The user clicks “Checkout” button 2. The user enters their credit card and billing address information on the checkout page 3. The user clicks the "Pay" button
Expected Outcome	A pop up message in red background with “payment not proceed” is showing on the checkout page
Notes	Only some “blacklisted” credit card numbers(predefined) will trigger expected outcome

Test ID	T04
Category	Review - Delete review
Requirements Coverage	Review-admin-delete-review-success
Initial Condition	Admin user is on the vehicle detail page, and there is at least one review for that vehicle
Procedure	<ol style="list-style-type: none"> 1. Admin user clicks on a review and is brought to a the single review details pop up window 2. Admin user clicks on “Delete” button 3. Admin user clicks on “confirm” button
Expected Outcome	The review details pop up window is closed, that review is no longer visible on the vehicle reviews section
Notes	<p>This action will make that review deleted permanently</p> <p>Make sure to add some reviews back after this test</p>

Test ID	T05
Category	Authentication – Evaluation of user credentials stored in database
Requirements Coverage	UC2a-Successful-User-Login
Initial Condition	User is on the login page, and has a registered account in the system
Procedure	<ol style="list-style-type: none"> 1. The user enters their email address 2. The user enters their password 3. The user clicks the "Login" button
Expected Outcome	The login form closes, user is logged in and redirected to the page they were on before initiating login
Notes	<p>Email and password must match a registered user</p> <p>Password must be alphanumeric (no special characters)</p> <p>If no prior page is recorded, redirect to the homepage</p>

Test ID	T06
Category	Authentication – Evaluation of login with incorrect credentials
Requirements Coverage	UC2b-Login-Failure-Handling
Initial Condition	User is on the login page and has a registered email in the system
Procedure	<ol style="list-style-type: none">1. The user enters a valid email address2. The user enters an incorrect password3. The user clicks the "Login" button
Expected Outcome	An error message is displayed indicating incorrect credentials; user remains on the login page
Notes	Account should not be locked on the first failure A retry option must be available

Test ID	T07
Category	Catalog – Viewing available electric vehicles
Requirements Coverage	UC5-View-Vehicles
Initial Condition	User is on the homepage or catalog page of the site. The system has vehicle listings in the database.
Procedure	<ol style="list-style-type: none">1. The user opens the homepage or clicks “Browse Vehicles”2. The system retrieves and displays the list of available electric vehicles
Expected Outcome	The page shows all vehicles currently in the catalog, including at least: Vehicle image, Model, Brand, Price, Availability.
Notes	Only vehicles with quantity > 0 are shown by default Vehicles can be new or used (condition should be displayed)

Test ID	T08
Category	Catalog – Filtering and sorting electric vehicles
Requirements Coverage	UC6-Filter-And-Sort-Vehicles
Initial Condition	User is on the vehicle catalog page. The system has multiple vehicles with varying prices, brands, and mileage.
Procedure	<ol style="list-style-type: none">1. The user selects a brand filter2. The user selects a sort option3. The system applies the filter and sort criteria
Expected Outcome	Only vehicles of the selected brand are shown, sorted in ascending order by price.
Notes	If no results match the filter, a “No vehicles found” message is shown Sorting and filtering should be combinable

Test ID	T09
Category	Shopping Cart – Add item to cart
Requirements Coverage	UC7-Add-To-Cart
Initial Condition	User is logged in and is viewing the vehicle catalog. At least one vehicle is available in stock.
Procedure	<ol style="list-style-type: none">1. The user clicks on a vehicle to view its details2. The user clicks the “Add to Cart” button3. The user navigates to the cart page
Expected Outcome	The selected vehicle is displayed in the cart with correct price, quantity = 1, and total cost
Notes	Vehicle must have quantity > 0 to be added If the vehicle is already in the cart, quantity should increase instead of creating a duplicate entry

Test ID	T10
Category	Checkout – Completing purchase with valid information
Requirements Coverage	UC8-Complete-Checkout
Initial Condition	User is logged in and has at least one vehicle in the shopping cart
Procedure	<ol style="list-style-type: none"> 1. The user navigates to the cart 2. The user clicks “Proceed to Checkout” 3. The user enters valid shipping and billing information 4. The user enters a valid credit card number 5. The user clicks “Confirm Order”
Expected Outcome	The order is processed successfully, a confirmation message is shown, and the cart is cleared
Notes	<p>Credit card must be valid per system rules (2 of every 3 succeed)</p> <p>Shipping address must be complete</p> <p>Vehicle quantities in stock are decremented after order</p>

Test ID	T11
Category	Catalog – Viewing details of a selected electric vehicle
Requirements Coverage	UC9-View-Vehicle-Details
Initial Condition	User is on the vehicle catalog page. At least one vehicle is available in the system with full detail data stored.
Procedure	<ol style="list-style-type: none"> 1. The user clicks on a vehicle card 2. The system loads the vehicle details page
Expected Outcome	<p>The details page displays the following:</p> <p>Vehicle image, Brand and model, Year, Price, Mileage, Condition, Vehicle history</p>
Notes	<p>If a history report is not available, the system should show “No history report available”</p> <p>Price and mileage should be shown in user-friendly format</p>

Test ID	T12
Category	Order – Track existing order status
Requirements Coverage	UC10-Track-Order
Initial Condition	User is logged in and has placed at least one order previously
Procedure	<ol style="list-style-type: none"> 1. The user navigates to their order history page 2. The user selects an order to view 3. The system retrieves the current status of the selected order
Expected Outcome	The system displays the order status along with the vehicle details and order number
Notes	<p>The system should prevent access to orders not associated with the logged-in user</p> <p>If no orders exist, show a friendly message: “You have no orders yet”</p>

Test ID	T13
Category	Finance – Monthly payment estimation via loan calculator
Requirements Coverage	UC11-Use-Loan-Calculator
Initial Condition	User is on a vehicle details page or loan calculator page
Procedure	<ol style="list-style-type: none"> 1. The user enters the following into the calculator form: <ul style="list-style-type: none"> • Vehicle price • Down payment • Interest rate • Loan duration 2. The user clicks the “Calculate” button
Expected Outcome	The system displays the estimated monthly payment amount, clearly labeled and rounded to two decimal places
Notes	<p>The system should validate that all inputs are positive numbers</p> <p>If fields are missing or invalid, display a warning instead of calculating</p>

Test ID	T14
Category	Catalog – Compare selected electric vehicles
Requirements Coverage	UC12-Compare-Vehicles
Initial Condition	User is on the vehicle catalog page. At least two vehicles are available and selectable for comparison.
Procedure	<ol style="list-style-type: none">1. The user selects two or more vehicles using checkboxes or a “Compare” toggle2. The user clicks the “Compare Vehicles” button
Expected Outcome	A comparison view is shown in a table or side-by-side layout, including: Model, brand, price, mileage, Condition, Ratings
Notes	If only one vehicle is selected, the compare button should be disabled Support up to 3 vehicles for comparison on standard screens

Test ID	T15
Category	Support – Chatbot navigation assistance
Requirements Coverage	UC13-Chatbot-Help
Initial Condition	User is on any page where the chatbot icon is accessible (
Procedure	<ol style="list-style-type: none">1. The user clicks on the chatbot icon2. The chatbot interface opens3. The user types: “Show me all Tesla vehicles under \$80,000”4. The chatbot responds with filtered results or a direct link to a filtered catalog view
Expected Outcome	Chatbot provides a meaningful response or navigational action based on user query
Notes	The chatbot should handle at least basic keyword-based commands If input is unrecognized, show “Sorry, I didn’t understand that”

Test ID	T16
Category	Shopping Cart – Duplicate item handling
Requirements Coverage	UC7-Add-To-Cart
Initial Condition	User is logged in and already has a specific vehicle in the cart
Procedure	<ol style="list-style-type: none">1. The user browses the catalog2. The user clicks “Add to Cart” on the same vehicle again3. The user views the cart
Expected Outcome	The quantity of the vehicle in the cart is incremented to 2, not duplicated as a separate entry.
Notes	Inventory stock must still be available Cart should visually reflect quantity rather than duplicating entries

Test ID	T17
Category	User – Registration (create new account)
Requirements Coverage	UC1-Register-Success
Initial Condition	User is on the registration page; the email address to be used is not yet registered in the system.
Procedure	<ol style="list-style-type: none">1. The user enters a first name and last name.2. The user enters a unique email address3. The user creates an alphanumeric password4. The user clicks the “Register” button.
Expected Outcome	The account is created successfully. The user is redirected to the login page and a confirmation message “Registration successful” is displayed.
Notes	All fields are mandatory; email must follow standard format Passwords must be alphanumeric; special characters are not allowed System stores the new profile in the user database and prevents duplicate future registrations with the same email.