A Synopsis Report on

Spare Part Inventory Management System

Submitted in partial fulfilment of the requirements of the degree of

PG Diploma in Advance Computing

by

Vikrant Pramod Kambli (250243120083)

Prasad Ravindra Patil (250243120054)

Sagar Ashok Mali (250243120027)

Tejas Mahendra Jadhav (250243120079)

Table of Contents :

Sr. No.	Title	Page No.
1	Introduction	5
2	Problem Definition & Scope	7
2.1	Problem Definition	7
2.2	Goals and Objectives	9
2.3	Main Constraints and Outcomes	11
3	Software Requirement Specifications	13
3.1	Purposed System	13
3.2	Scope	16
4	System Modules	18
5	Performance Requirements	22
5.1	H/W Requirements and S/W Requirements	24
6	UML Diagrams	26
6.1	DFD	26
6.2	ERD	27
6.3	Use Case Diagram	28
6.4	Class Diagram	29
6.5	Sequence Diagram	30
6.6	Activity Diagram	31
6.7	Deployment Diagram	32
6.8	System Architecture	33
7	Database Design	36
8	Test Cases	40
9	Screenshot's	42
10	References	53

1. INTRODUCTION TO PROJECT

The web-based "Spare Part Management System" project aims to simulate the essential features of an online platform for managing and purchasing automobile or machinery spare parts. This system allows users to register on the website, enabling them to browse, purchase spare parts, and raise service-related complaints if needed. The system provides registration functionality for various types of users, allowing each to create a personalized account.

Once logged in, users can explore available spare parts, add them to the cart, place orders, track deliveries, and provide feedback or ratings on purchased items. The system includes three types of logins: User, Admin, and Delivery Personnel, each assigned with specific roles and responsibilities.

- Admin Module: This module plays a crucial role in managing the overall operations of the platform. It allows the administrator to monitor inventory, handle user queries, update product listings, and ensure smooth functioning of the system.
- Customer Module: This module is designed to provide customers with a user-friendly interface to search for spare parts, check availability, place orders, and manage their profiles and order history.
- Seller Module: This section focuses on streamlining the seller add category and edit category of ordered spare parts, ensuring timely and accurate deliveries to the users.

At "Spare Part Management System," our mission is to offer a reliable, efficient, and user-friendly platform for sourcing genuine spare parts. Whether for repair, maintenance, or replacement, the system is designed to ensure customers get exactly what they need, with ease and transparency.

The Spare Part Management System is set to modernize the spare parts distribution process by combining automation with convenience, helping both businesses and individual users manage their spare part needs efficiently and effectively.

2. Problem Definition & Scope

2.1. Problem Definition

The automobile spare parts industry faces numerous challenges that arise due to a lack of proper coordination between manufacturers, shippers, sellers, and end customers. The traditional system is mostly manual or disconnected, leading to issues such as:

- **Poor Inventory Management**: Sellers and companies often struggle with real-time stock updates, leading to delayed deliveries or order cancellations.
- **Inefficient Order Tracking:** Customers are unable to track their orders, and sellers rely heavily on manual processes for delivery updates.
- **Lack of Role-based Control:** There is no centralized system where each user (Admin, Shipper, Seller, Customer) can perform tasks based on their role.
- **Delayed Communication:** In traditional workflows, there is a communication gap between sellers and shipping partners.
- **Lack of Customer Trust:** Without Aadhaar verification, customers may feel insecure during registration and payment.

The absence of an integrated solution creates operational inefficiencies, delays, customer dissatisfaction, and loss of revenue.
SparePartWala aims to address these issues by providing a centralized web-based portal where each user interacts through their respective role with ease and clarity, ensuring transparency, efficiency, and security.

- **Scope of the Project**

The scope of this project is to develop a full-fledged web-based application named SparePartWala, which streamlines the management of automobile spare parts and ensures

seamless coordination between different stakeholders:

- **Multi-role Access:** The system will allow login access for four major users — Company (Manufacturer), Shipper, Seller, and Customer — each with customized dashboards and functionalities.
- **Single-Time Registration:** Users will register once using their personal details including Aadhaar number and photograph. These details will be reused in login and identity verification.
- **Inventory and Order Management:** Sellers can manage their inventory, update product information, and track stock levels. Orders will be processed automatically based on availability.
- **Order Tracking and Shipment Updates**: Customers will be able to track their orders in real-time, and shippers can update delivery status directly in the system.
- **Secure and Transparent System:** Aadhaar integration and photo-based identity ensure trustworthy transactions and user verification.
- The system will be scalable for adding more users or roles in the future and extendable to mobile applications.

The ultimate goal is to automate the spare parts distribution process, enhance efficiency, reduce manual dependencies, and build a transparent and trustworthy platform for all participants in the automobile spare part supply chain.

2.2 Goals & Objectives

- **Goals**

The primary goal of the SparePartWala system is to develop a centralized, secure, and role-based web application that streamlines the workflow of the automobile spare parts industry—from registration and inventory management to order placement—while enhancing operational efficiency, transparency, and user experience.

- **Objectives**

To achieve the defined goal, the project outlines the following specific objectives:

1. **Centralized Platform Development**

- Design and implement a web-based system that enables three user roles—Admin, Seller, and Customer—to interact through a unified platform with dedicated functionalities.

2. **User Registration and Authentication**

- Implement one-time secure user registration for all roles.
- Develop role-based login mechanisms to ensure secure and controlled access to the system.

3. **Inventory and Product Management**

- Enable sellers to add, update, or remove spare part listings.

- Ensure real-time inventory tracking to maintain availability and prevent stock mismatches.

4. **Order Placement and Processing**

- Allow customers to browse spare parts, add items to the cart, and place orders.
- Notify sellers upon order placement for timely processing.

5. **Admin Monitoring and Role Management**

- Provide the Admin with capabilities to monitor platform activities, manage user roles, resolve issues, and ensure system security and compliance.

6. **Improve Communication and Reduce Delays**

- Facilitate real-time system notifications to streamline interactions between customers, sellers, and administrators.

7. **Scalability and Maintainability**

- Develop the application with a modular and extensible architecture to support future enhancements such as mobile application support, payment gateway integration, or product recommendations.

2.3 Major Constraints & Outcomes

Key Constraints

1. **System Architecture & Performance**

- Real-time syncing between inventory, orders, and tracking.
- Must scale smoothly with increasing users and traffic.

2. **Security & Access Control**

- Role-based authentication for Admin, Seller, and Customer.
- Protection against unauthorized data access.

3. **External Dependency Risks**

- Downtime in third-party APIs (email/SMS/maps) can affect core modules.
- Reliable data backup and recovery for large volumes.

Expected Outcomes :

1. **Streamlined Multi-role Operations**

- Automated workflows for each user role.
- Easy collaboration among Admin ,sellers and customers.

2. **Improved Customer & Seller Experience**

- Real-time updates and tracking across the system.
- Intuitive and responsive interface for all devices.

3. **Future-ready Platform**

- Scalable for mobile app or feature upgrades.
- Can integrate additional modules like payment gateway.

3. Software Requirement Specification

3.1 Purposed System

The proposed system, SparePartWala, is a web-based Spare Parts Inventory Management platform developed using React for the frontend and Spring Boot for the backend. It focuses on centralizing operations between three main users Admin, Seller, and Customer to ensure smooth, transparent, and efficient management of spare parts inventory, ordering, and distribution.

Key Features :

Multi-Role Authentication and Dashboard Access:
The system supports separate login mechanisms for Admin, Seller, and Customer roles. After authentication, each user is redirected to a customized dashboard relevant to their responsibilities. Admins manage system-wide activities, Sellers handle product listings and stock, while Customers can search and order spare parts.

Spare Parts Inventory and Stock Monitoring (Seller Module):
Sellers can add, update, or delete spare part records from their dashboard. The system tracks stock in real-time, enabling sellers to monitor inventory levels and avoid overselling or order cancellations due to out-of-stock items.

Order Placement and History (Customer Module):
Customers can browse the catalog of available spare parts, place orders securely, and view their order history. The system also allows users to check the current status of their orders, providing better transparency in the purchase process.

Admin Control Panel and Analytics:
The Admin dashboard offers control over users, product categories, and seller management. It

includes system statistics and analytics such as total orders, active sellers, top-selling items, and more, aiding better business decisions.

User-Friendly Responsive Interface:

The entire application is responsive and works seamlessly across devices. The interface is designed to be intuitive and accessible, offering a clean experience to all stakeholders, whether on desktop or mobile.

Role-Based Security and Access Control:

Each user type has specific access permissions. Admins can manage users and view all transactions, while Sellers and Customers have restricted access to features relevant to their operations. This ensures data security and operational integrity.

3.2 Scope

The proposed system “SparePartWala” aims to provide a unified web-based platform that enables seamless interaction among four major stakeholders: Admin (Manufacturer), Seller, Shipper, and Customer. The system is designed to bring automation and transparency in the automobile spare parts supply chain. The following points describe the scope of this system in detail:

1. **User Registration and Login**

The system shall provide secure registration functionality using Aadhaar number verification, ensuring authenticity of users. Each user will register according to their role—Admin, Seller, Shipper, or Customer—and will receive role-specific login access. Proper password hashing, email verification, and login session management will be implemented.

2. **Role-Based Dashboards**

Every user will have access to a customized dashboard that provides relevant features based on their role. The Admin dashboard will include access to user management, order monitoring, and reporting. The Seller dashboard will provide inventory control and order status management. The Shipper dashboard will allow viewing of assigned deliveries and updating shipment status. The Customer dashboard will support browsing products, placing orders, and viewing order history.

3. **Inventory Management (Seller Panel)**

Sellers will be able to manage spare part listings by adding, updating, or deleting products along with prices and available stock. This panel will allow sellers to monitor which items are selling

the most and receive notifications when stocks are low. They will also be able to view and track orders placed by customers for their products.

4. **Product Browsing and Ordering (Customer Panel)**

Customers can explore available spare parts using a category-wise or vehicle-wise search system. The system will provide detailed product descriptions, prices, and seller ratings. Customers will be able to add items to the cart, place orders securely, and view their order history along with real-time delivery status tracking.

5. **Order Management (Admin Panels)**

The Admin will have access to all orders placed on the platform and will assign delivery tasks to available shippers. Shippers can update the order status at each stage such as Picked, In Transit, and Delivered. This ensures full transparency and real-time tracking for both admin and customers.

6. **Security and Validation**

Robust input validation will be implemented across all forms to prevent SQL injection and ensure data integrity. Role-based access control will prevent unauthorized data access. Aadhaar number and email verification will be performed during registration to prevent fraudulent accounts. HTTPS encryption will be enforced throughout the site for secure communication.

7. **Analytics and Reporting (Admin Panel)**

The Admin will be able to generate analytical reports such as sales trends, product demand analysis, order fulfilment rate, and user registration metrics. These reports will help in making data-driven decisions to optimize supply chain operations.

4. System Modules

4.1 Admin Module

The Admin module serves as the backbone of the system, granting administrators complete control over the application's configuration and workflow.

- **User Management**
    - Approve or reject Seller registrations.
    - Monitor registered Customers.
    - Manage user roles and permissions (Spring Security-based role handling).
    - Built using Spring Boot Controller, Service, and Repository layers.
- **Category & Master Data Management**
    - Create and update product categories for sellers.
    - Maintain reference tables using MySQL**.**
- **Order Supervision**
    - Monitor all orders across the system.
    - View seller-wise and customer-wise order history.
- **Reporting & Analytics**
    - Generate reports like sales trends, inventory logs, and user activity using REST APIs.
    - Display data in visual form using React.js charts (e.g., Chart.js or Recharts if used).
- **Technical Stack:**
    - Spring Boot (Java 17), MySQL, Spring Data JPA
    - RESTful APIs, JSON for backend communication
    - Role-based access via Spring Security

4.2 Seller Module

This module enables spare part sellers to handle their own digital storefront within the platform.

- **Product Management**
  - Add new spare part listings with attributes (name, category, price, quantity, description).
  - Update and delete existing products via secure endpoints.
  - Product form validation done using React Hooks and controlled components.
- **Inventory Control**
  - View and update stock availability in real time.
  - Prevent over-ordering by syncing with MySQL inventory records.
- **Order Handling**
  - View incoming customer orders.
  - Change order status to "Ready for Dispatch" via PUT API.
- **Technical Stack:**
  - Frontend: React.js (useState, useEffect, Axios)
  - Backend: Spring Boot, JPA Repositories
  - Data Format: JSON
  - State handled via React Hooks or Context API (if applicable)

4.3 Customer Module

The Customer module is designed for end-users (buyers) to explore and purchase spare parts effortlessly.

- **Product Browsing**
  - Browse by category, search by name/brand.
  - View detailed product information and availability.
- **Order Placement**
  - Add products to cart (if implemented) or directly order.
  - React handles cart state and Axios sends POST requests to the backend.
- **Order History & Tracking**
  - View past orders and track the status (Pending, Ready for Dispatch, Delivered).
  - Dynamic rendering of order data using React and REST API.
- **User Profile Management**
  - Update personal information securely.
  - Session data stored using JWT Tokens (if implemented).
- **Technical Stack:**
  - Frontend: React.js (React Router, Axios for API calls)
  - Backend: Spring Boot controllers and services
  - Persistent data in MySQL (customer, order tables)

4.4 Authentication & Authorization Module

This module ensures that each user accesses only their respective parts of the application securely.

- **User Registration & Login**
  - All users register with username, password, email (stored in encrypted format using BCrypt).
  - Secure login using JWT-based authentication (if used) or Spring Security with sessions.
- **Role-Based Access Control**
  - Admin, Seller, and Customer roles are defined in the database.
  - Access control configured via Spring Security's @PreAuthorize, filters, and role checks.
- **Session & Token Management**
  - Secure sessions or tokens are used to keep users authenticated.
  - React stores tokens using localStorage or sessionStorage.
- **Technical Stack:**
  - Backend: Spring Security, BCryptPasswordEncoder
  - React: Protected Routes, token-based redirection
  - MySQL stores user credentials securely

5. Requirement

- **Functional Requirements**

1. **User Registration and Authentication:**

   The system shall allow Admins, Sellers, and Customers to register and log in with secure credentials. Each user will be redirected to a role-specific dashboard after login.

2. **Role-Based Access Control:**

   The system shall provide access to features based on user roles:

   - **Admin:** Full system access and control
   - **Seller:** Manage inventory and process orders
   - **Customer:** Browse products and place orders

3. **Product & Inventory Management (Seller):**

   Sellers shall be able to add new spare parts, update stock levels, set prices, and delete or deactivate products as needed.

4. **Product Browsing and Search (Customer):**

   Customers shall be able to browse spare parts by category, brand, or keyword search. Product listings should include images, descriptions, price, and availability.

5. **Order Placement and Management:**

   Customers shall be able to place orders for available products. Once placed, the system shall update the inventory and notify the relevant seller.

6. **Order Processing (Seller):**

   Sellers shall be able to view new orders, mark them as “Ready for Dispatch”, and monitor order status.

7. **Admin Panel Functionalities:**

Admin shall be able to:

- Approve/reject seller registrations
- Manage product categories
- Monitor overall system activity
- View and generate reports on users, inventory, and sales

8. **Responsive UI Design:**

The application shall be usable across various screen sizes (desktop, tablet, mobile), ensuring a consistent experience.

9. **Database Operations**

All data (users, products, orders, inventory) shall be stored and managed in a MySQL database using Spring Data JPA.

- **<u>Non -Functional Requirements:</u>**

Following Non-Functional Requirements will be there in the insurance to the internet:

1. Secure access to user's confidential data.
2. 24X7 availability.
3. Better component design to get better performance at peak time.
4. Flexible service-based architecture will be highly desirable for future extension. Non-Functional requirements define system properties andconstraints.

Various other Non-Functional Requirements are:

- Security
- Reliability
- Maintainability
- Portability
- Extensibility
- Reusability
- Compatibility

5.1.1 Hardware Requirements

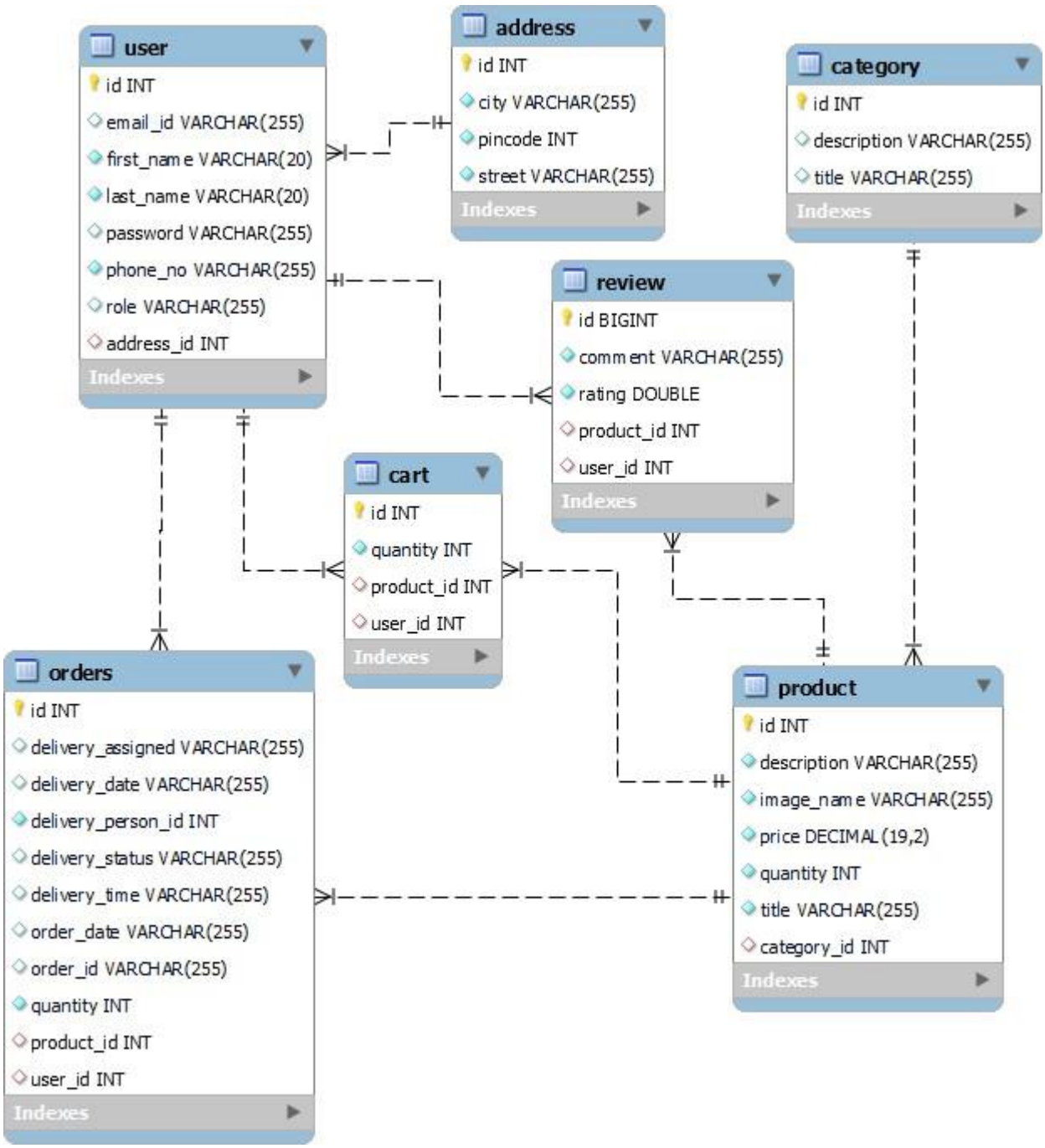
- Intel Core i3 processor or equivalent (minimum)
- **8 GB RAM (minimum), 16 GB recommended for optimal performance**
- **500 GB HDD or 256 GB SSD for data and code storage**
- 13-inch or larger display with at least 1366×768 resolution
- Standard input devices (keyboard and mouse)
- Stable internet connection for API usage, Git, and deployment

5.1.2 Software Requirements

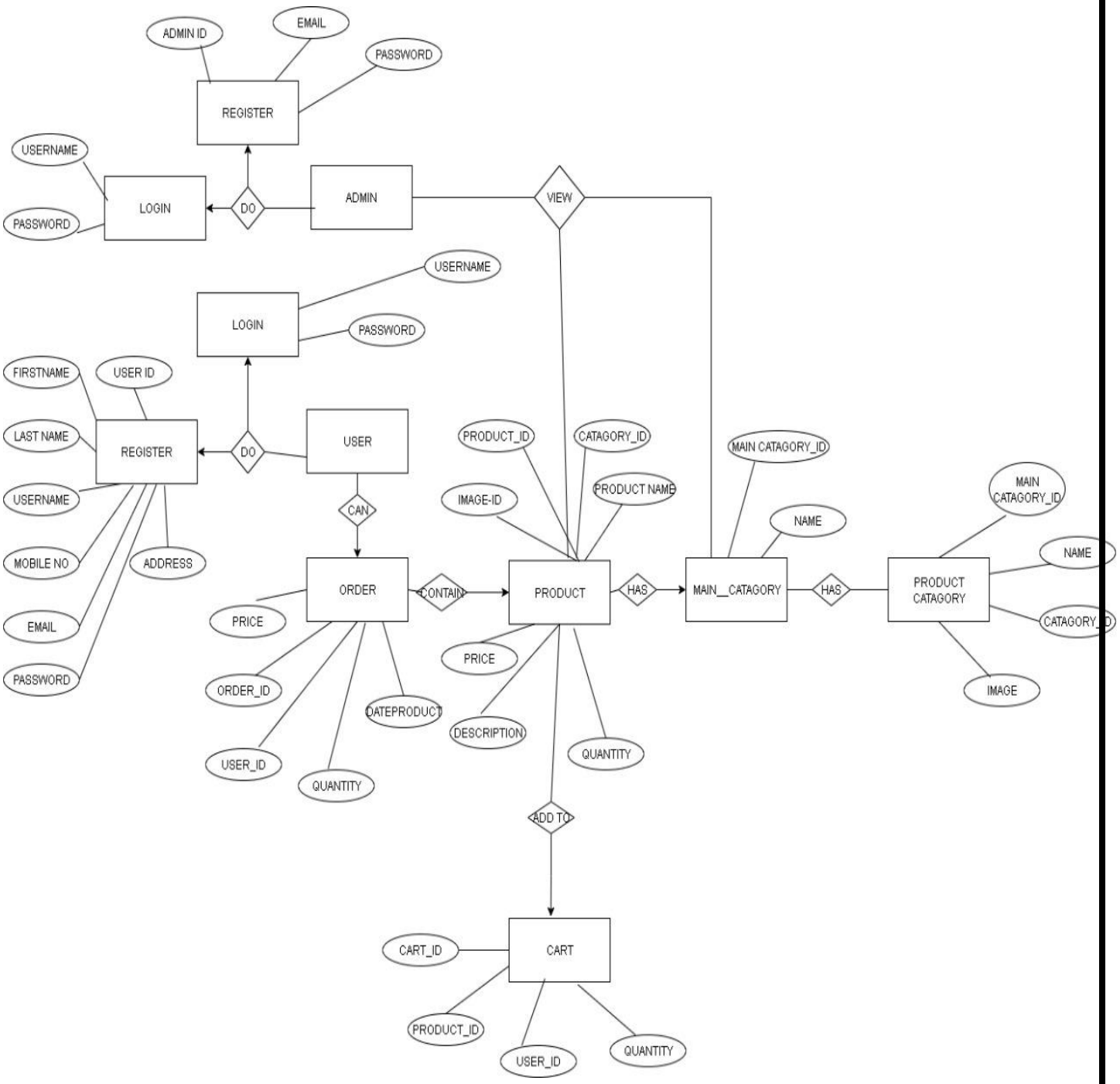
- Windows 10/11, Linux, or macOS operating system
- Visual Studio Code for frontend development using React.js
- Eclipse IDE for backend development using Spring Boot
- Node.js and npm for running and building the React frontend
- Java JDK 17 or higher for compiling and executing Spring Boot applications
- MySQL database for storing user, inventory, and order data
- Apache Maven for backend build and dependency management
- Git and GitHub for version control and collaboration
- Postman for testing RESTful APIs
- Google Chrome or Mozilla Firefox for UI testing and validation

6. UML Diagrams

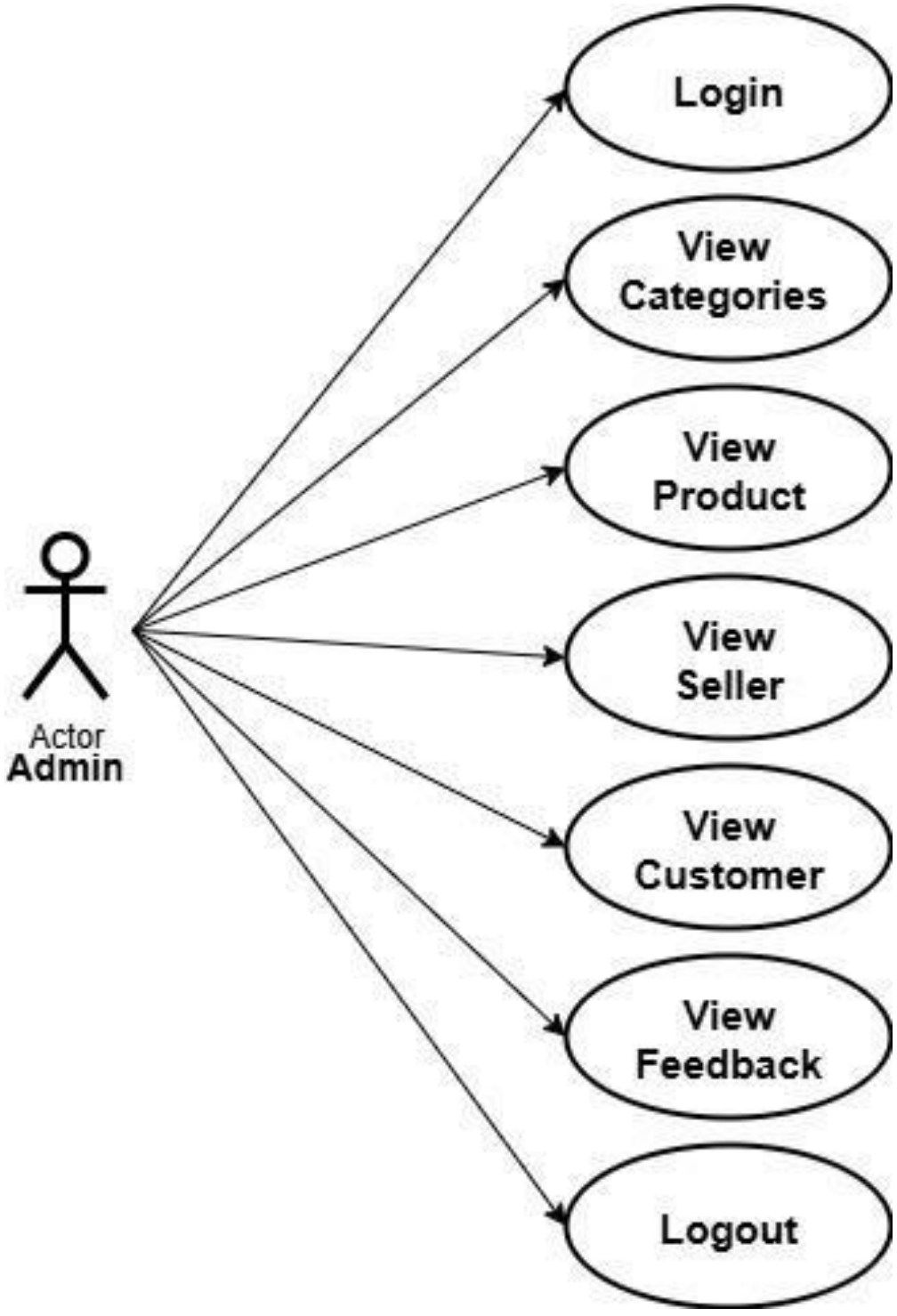
6.1 Data Flow Diagram (DFD)

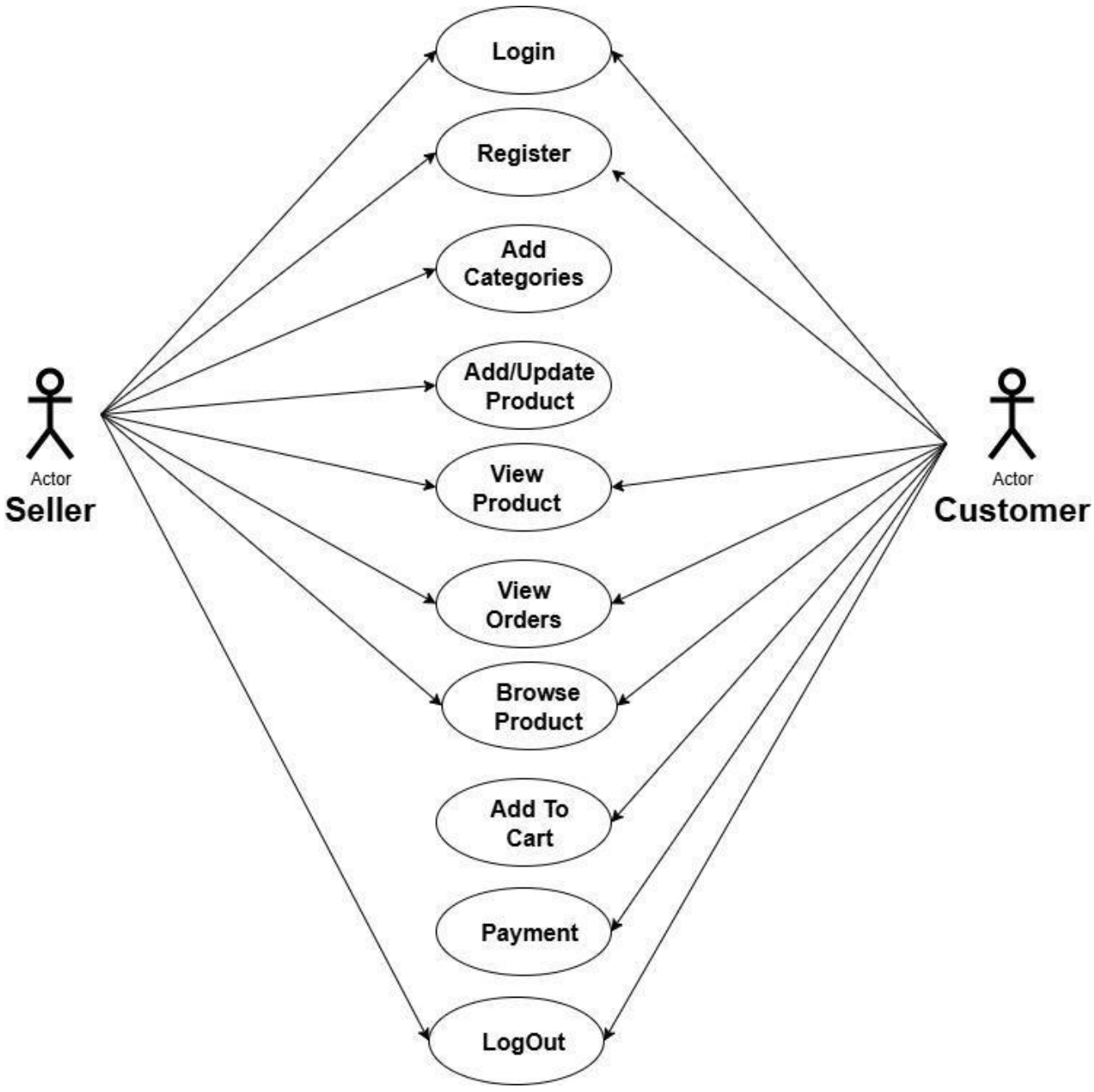


6.2 Entity-Relationship Diagram (ERD)

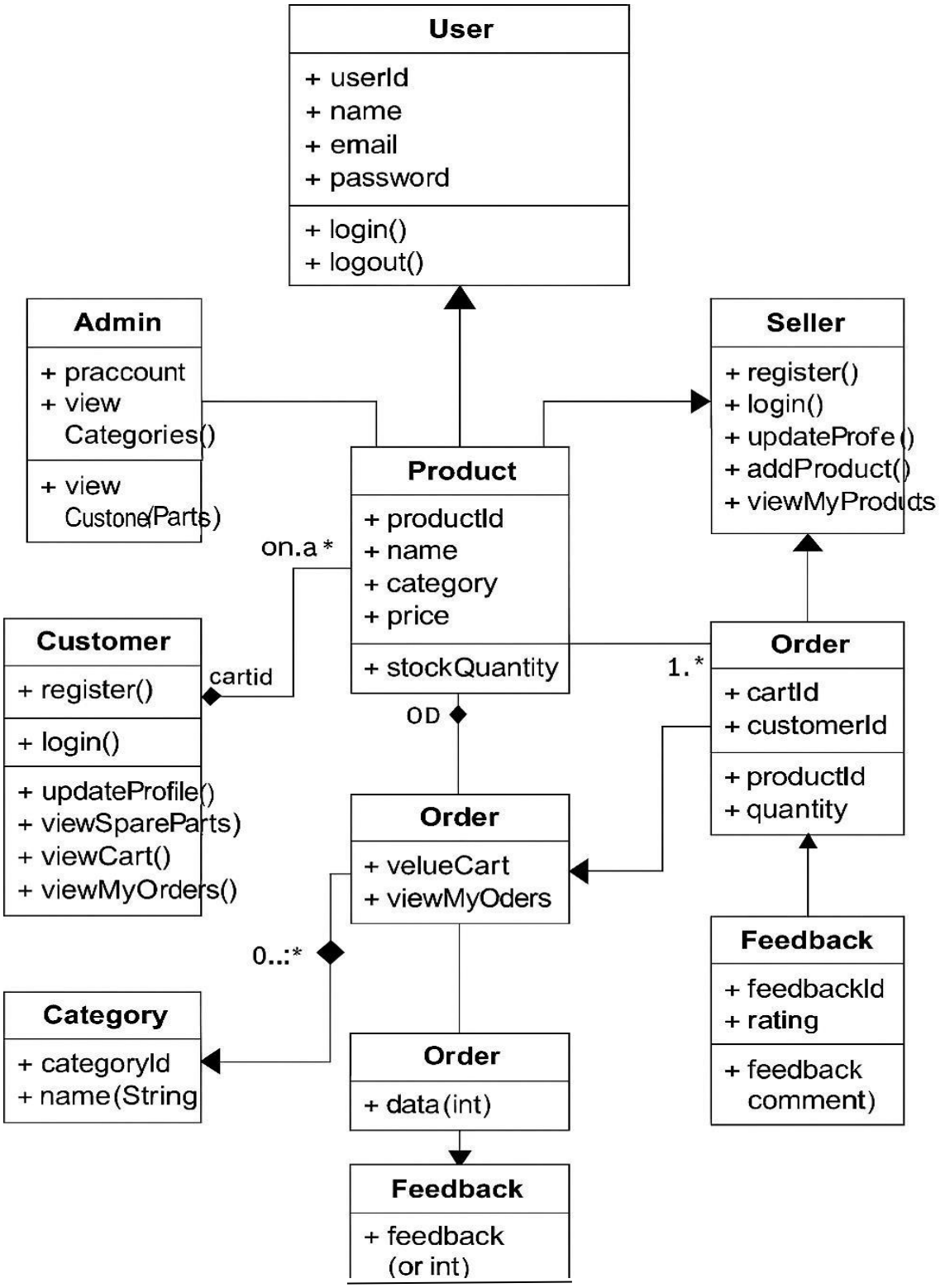


6.3 Use Case Diagram

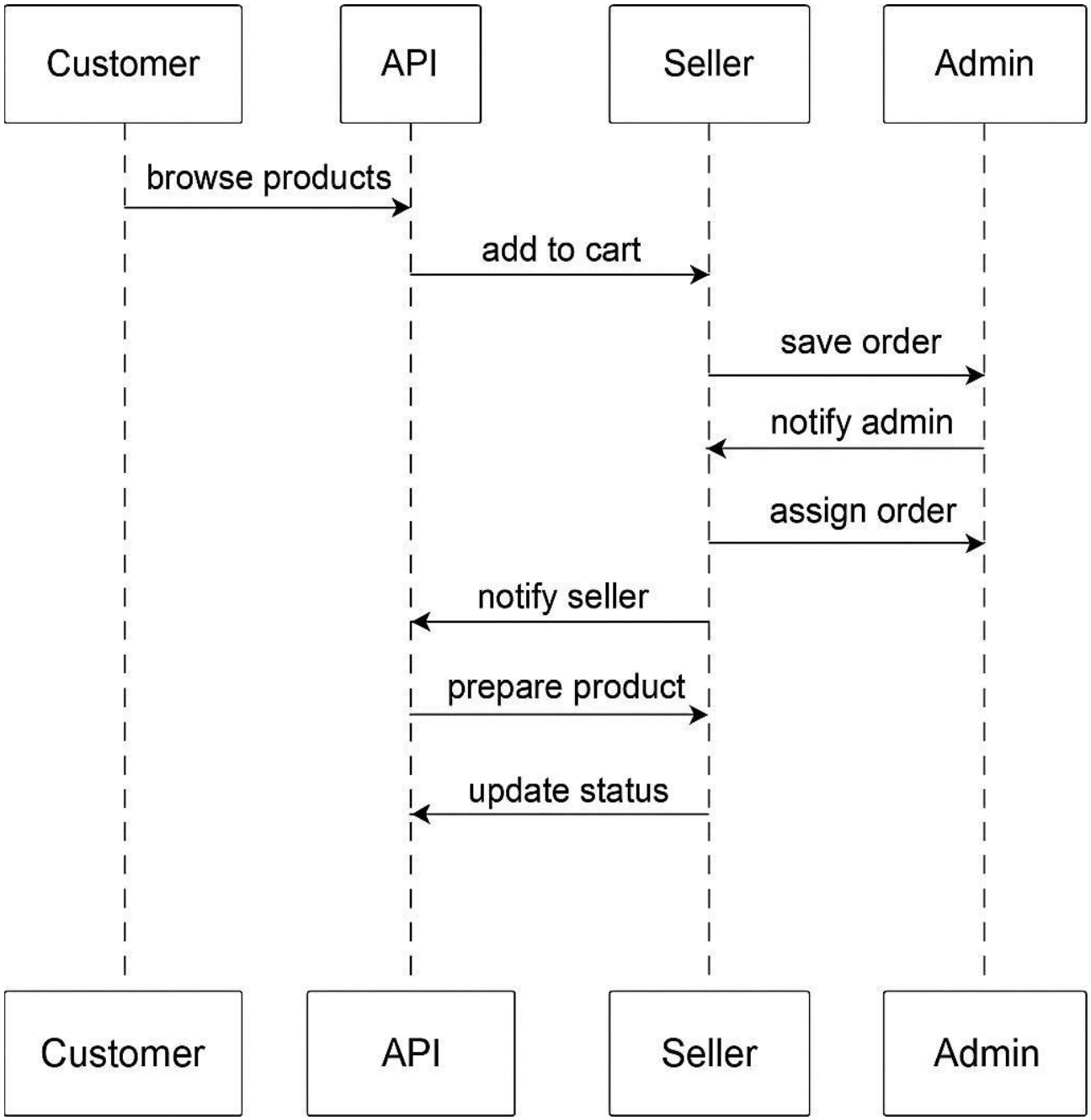



Login
Register
Add Categories
Add/Update Product
View Product
View Orders
Browse Product
Add To Cart
Payment
LogOut
Actor
Seller
Actor
Customer

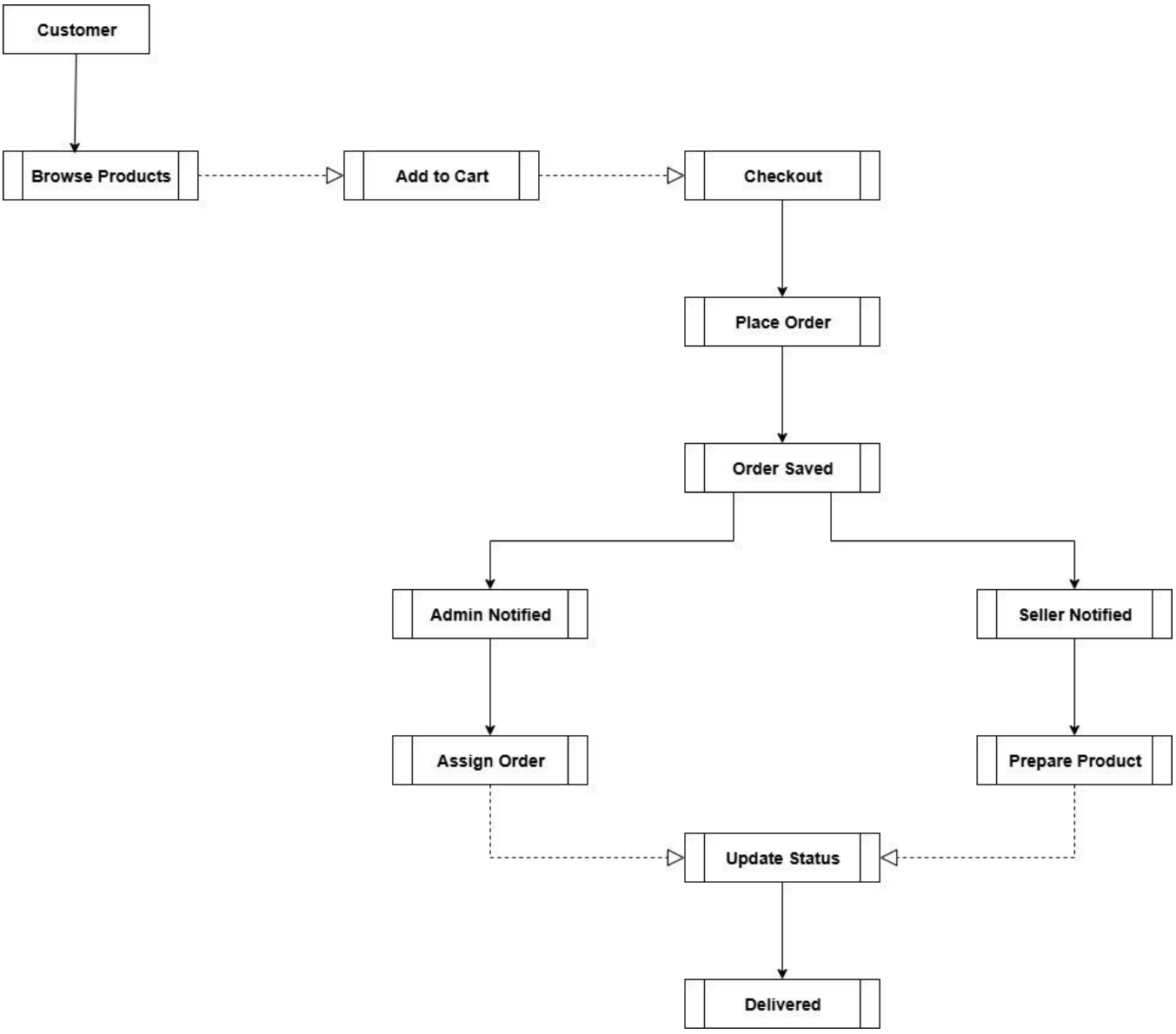
6.4 Class Diagram



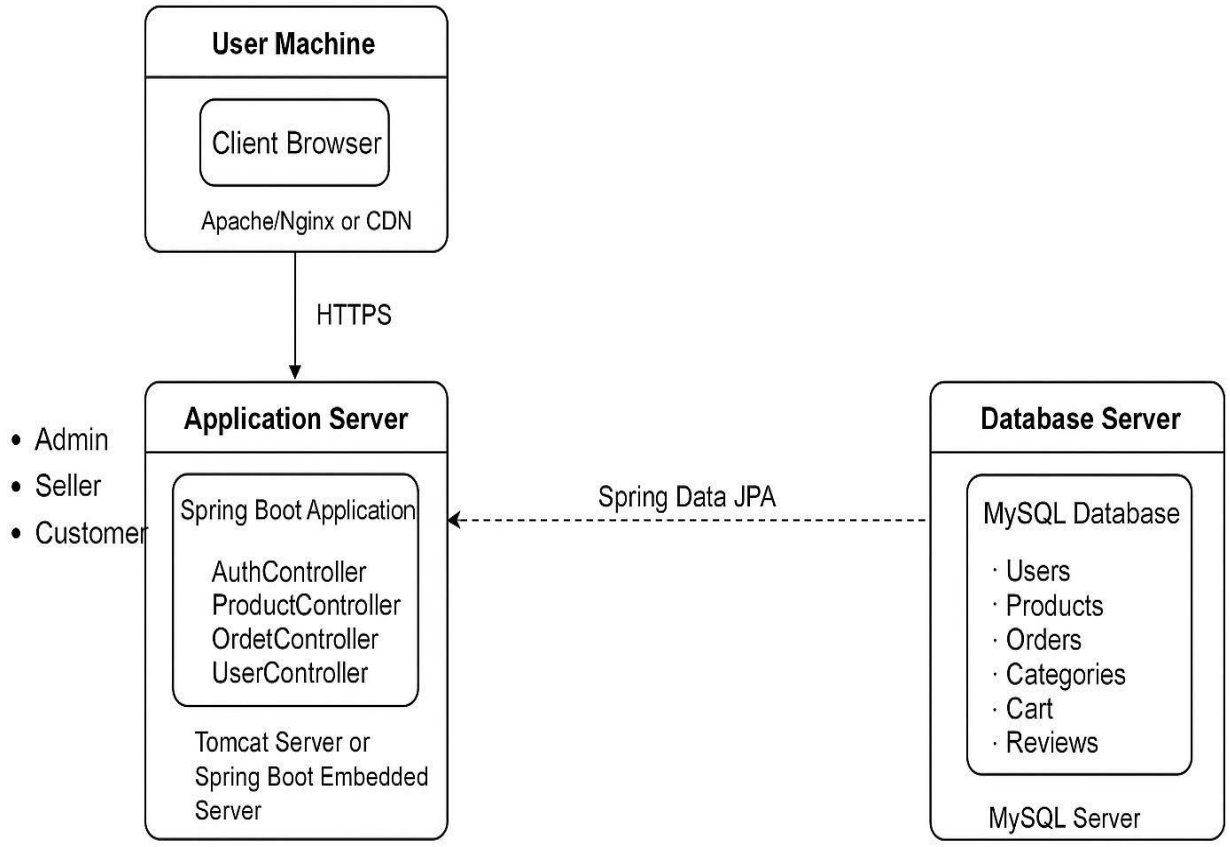
6.5 Sequence Diagram



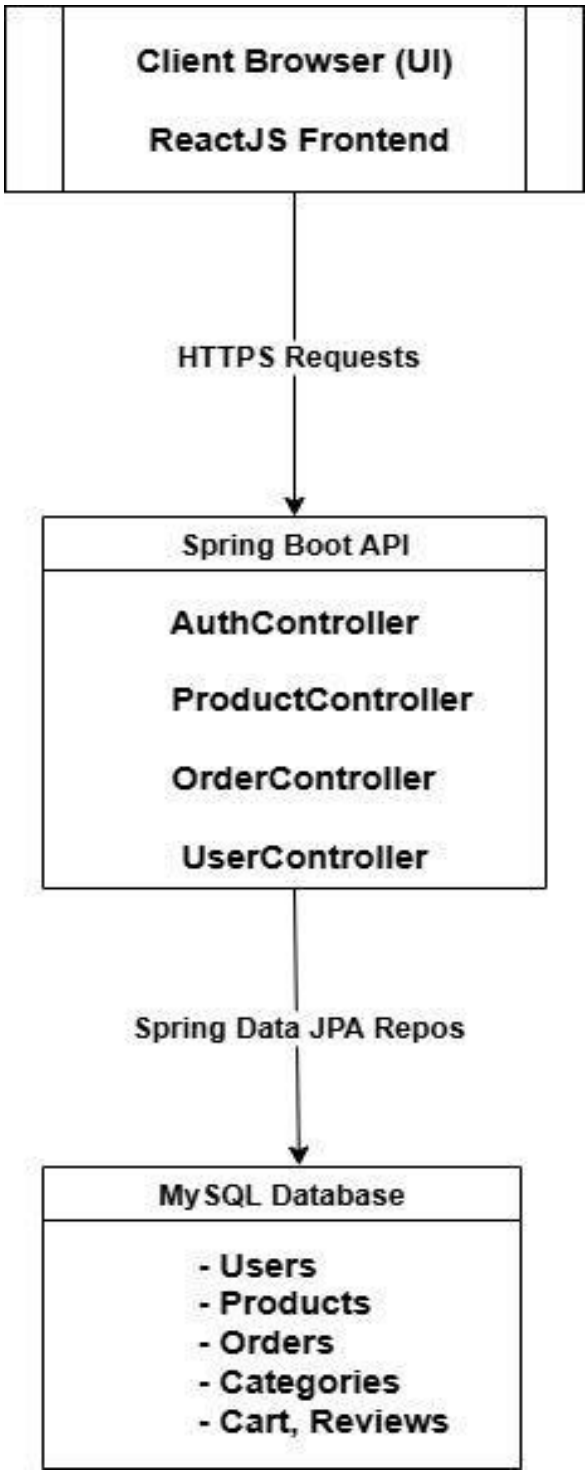
6.6 Activity Diagram



6.7 Deployment Diagram



6.7 System Architecture Diagram



7. Database Design

The following table structures depict the database design.

Table 1 : Customer

Key Type/ Constraint	Column Name	Data Type	Length	Allow Null (1=Yes;0=No)
2	id	INT	-	0
1	email_id	varchar	255	1
0	first_name	varchar	20	0
0	lae	varchar	20	0
0	password	varchar	255	1
0	phone_no	varchar	255	0
0	role	varchar	255	1
1	address_id	INT	-	1

Table 2 : address

Key Type/ Constraint	Column Name	Data Type	Length	Allow Null (1=Yes;0=No)
2	id	INT	-	0
0	city	varchar	255	0
0	pincode	INT	-	0
0	street	varchar	255	0

Table 3 : Category

Key Type/ Constraint	Column Name	Data Type	Length	Allow Null (1=Yes;0=No)
2	id	INT	-	0
0	description	varchar	255	0
0	title	varchar	255	0

Table 4 : Product

Key Type/ Constraint	Column Name	Data Type	Length	Allow Null (1=Yes;0=No)
2	id	INT	-	0
0	description	varchar	255	0
0	image_name	varchar	255	0
0	price	decimal	(19,2)	0
0	quantity	INT	-	0
0	title	varchar	255	0
1	category_id	INT	-	1

Table 5 : Review

Key Type/ Constraint	Column Name	Data Type	Length	Allow Null (1=Yes;0=No)
2	id	bigint	-	0
0	comment	varchar	255	0
0	rating	double	255	0
1	product_id	INT	255	1
1	user_id	INT	255	1

Table 6 : Cart

Key Type/ Constraint	Column Name	Data Type	Length	Allow Null (1=Yes;0=No)
2	id	INT	-	0
0	quantity	varchar	255	0
1	product_id	INT	255	1
1	user_id	INT	255	1

Table 7 : Orders

Key Type/ Constraint	Column Name	Data Type	Length	Allow Null (1=Yes;0=No)
2	id	INT	-	0
0	delivery_assigned	varchar	255	1
0	delivery_date	varchar	255	1
0	delivery_person_id	INT	-	0
0	delivery_status	varchar	255	1
0	delivery_time	varchar	255	1
0	order_date	varchar	255	1
0	order_id	varchar	255	1
0	quantity	INT	-	0
1	product_id	INT	-	1
1	user_id	INT	-	1

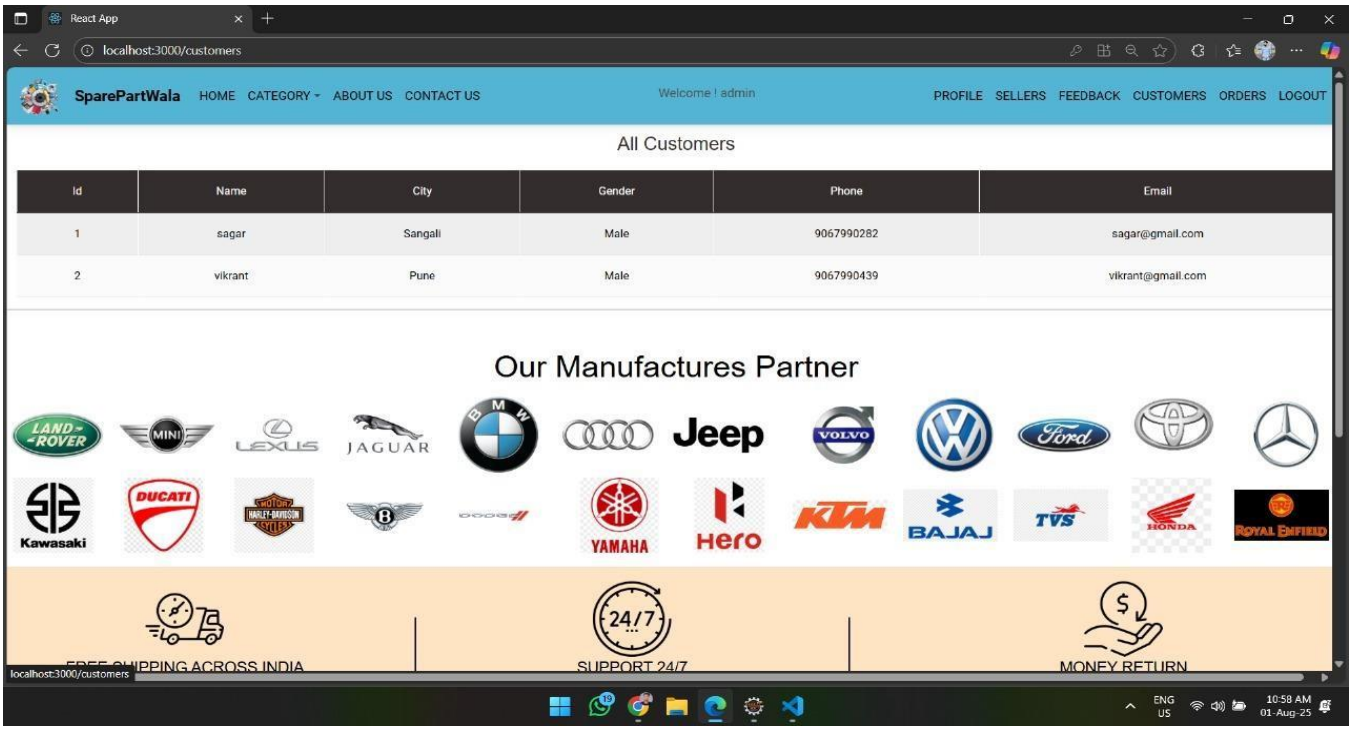
7. Test Cases

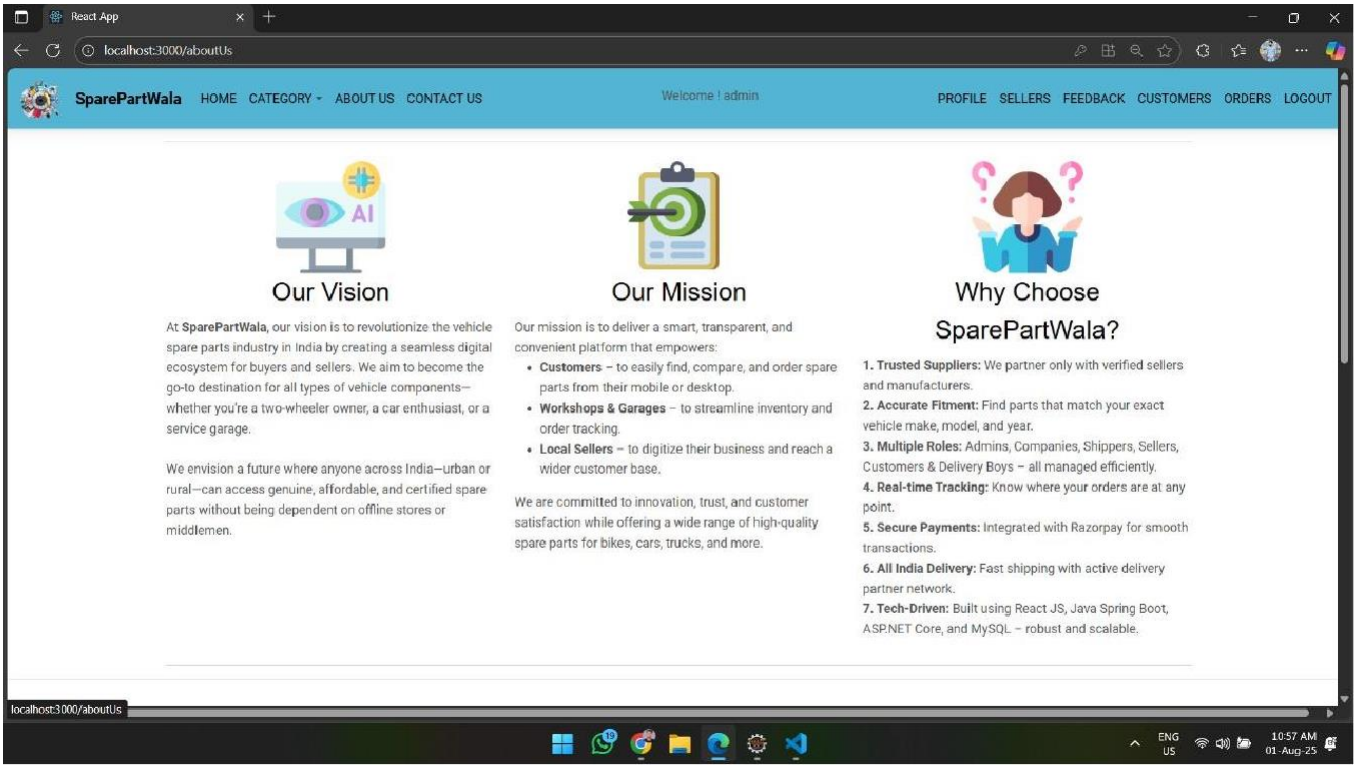
GENERAL TESTING :

SR-NO	TEST CASE	EXPECTD RESULT	ACTUAL RESULT	ERROR MESSAGE
1	Register Page	Redirected to Nextpage	OK	Nothing
2	Login Page	Pop-up will come	Ok	Please enter username and password again .
3	Costumer Register	Register a Costumer	Ok	Nothing
4	Checking login or not	Costumer is logged in or not	Ok	Nothing
5	Add Product	Add Information about Product	Ok	Nothing
6	Order Details	Get all orders	Ok	Nothing
7	Add to cart	Test adding products to cart	Ok	Nothing
8	Security (SQL Injection)	Test SQL injection vulnerability	Ok	Nothing
9	Mobile Responsiveness	Test website on mobile devices	Ok	Nothing
10	Logout	It will logout from userprofile.	Ok	Nothing

8. Screenshots

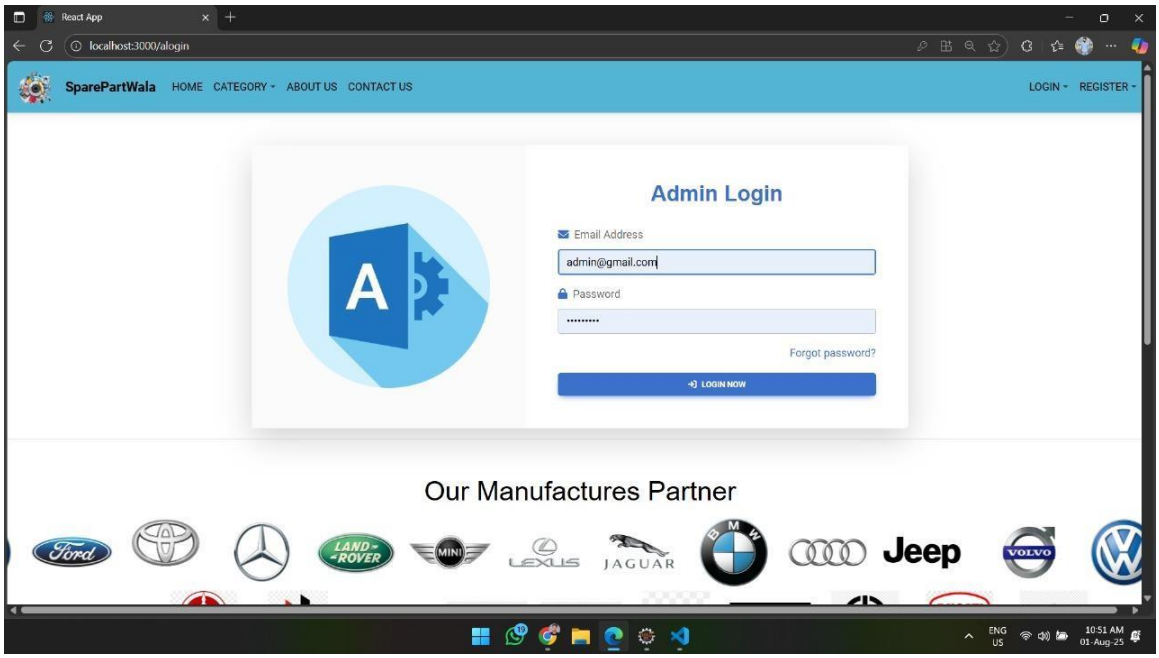
Homepage :



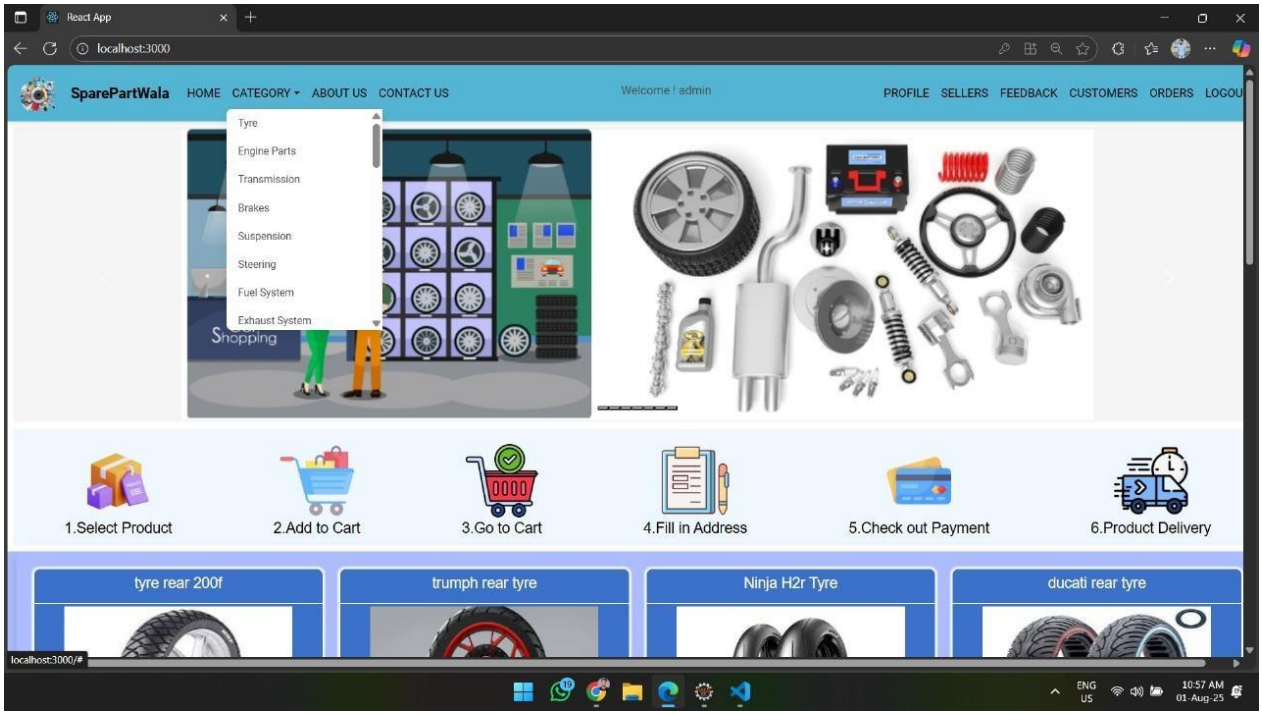


Admin

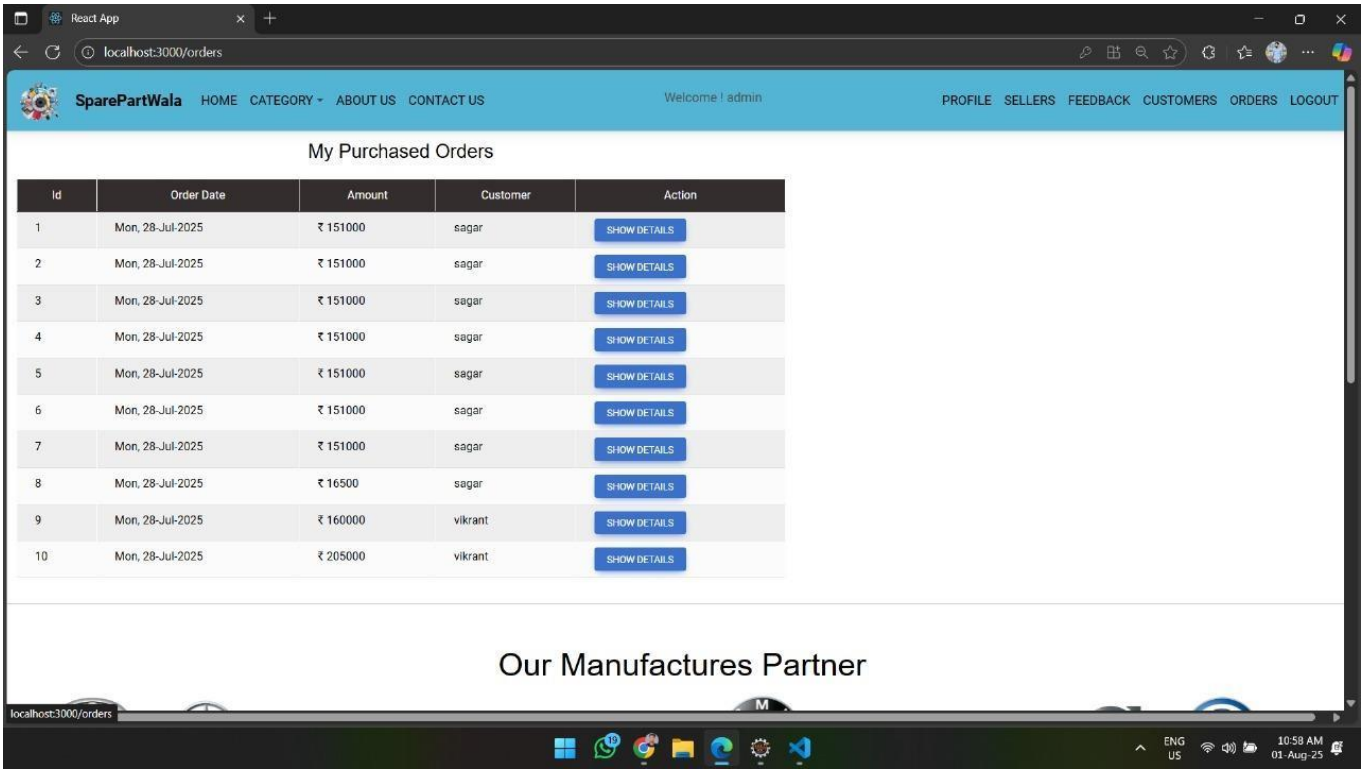
1. Admin Login



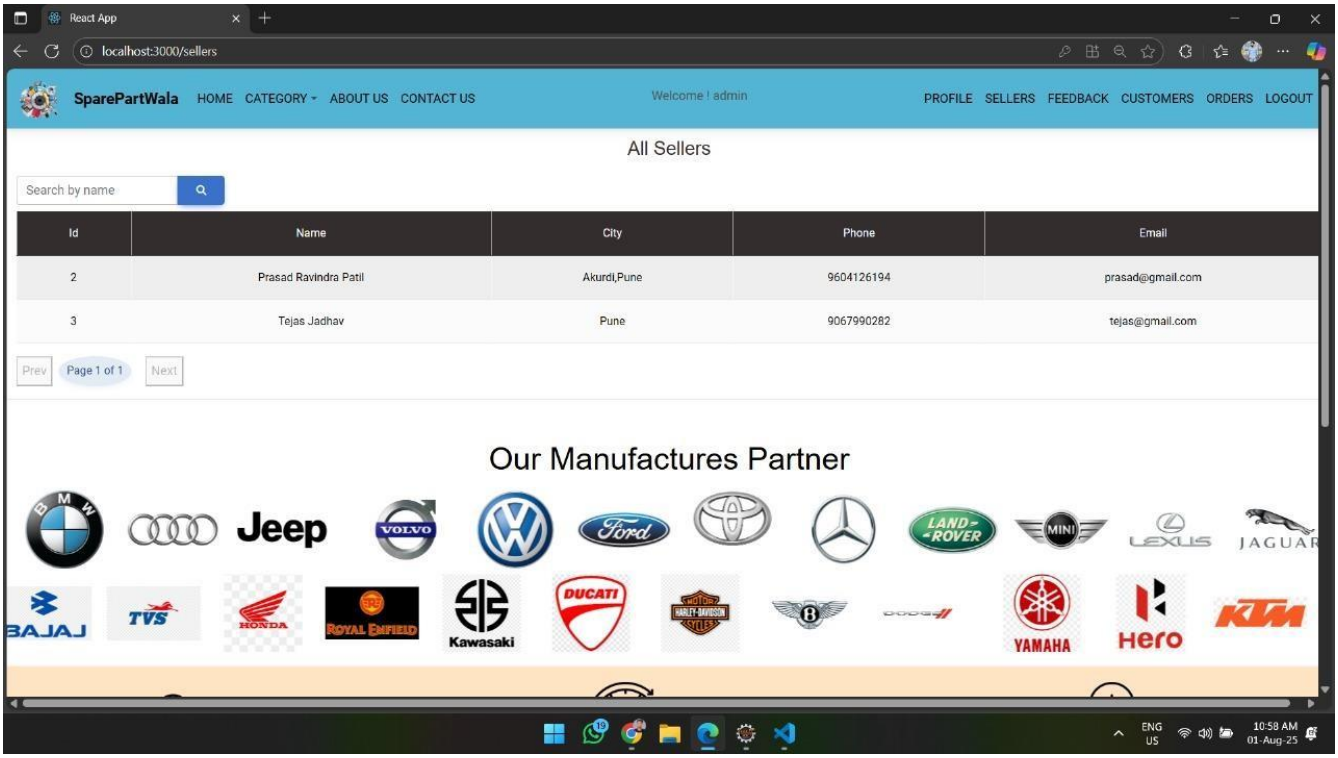
2. Admin page



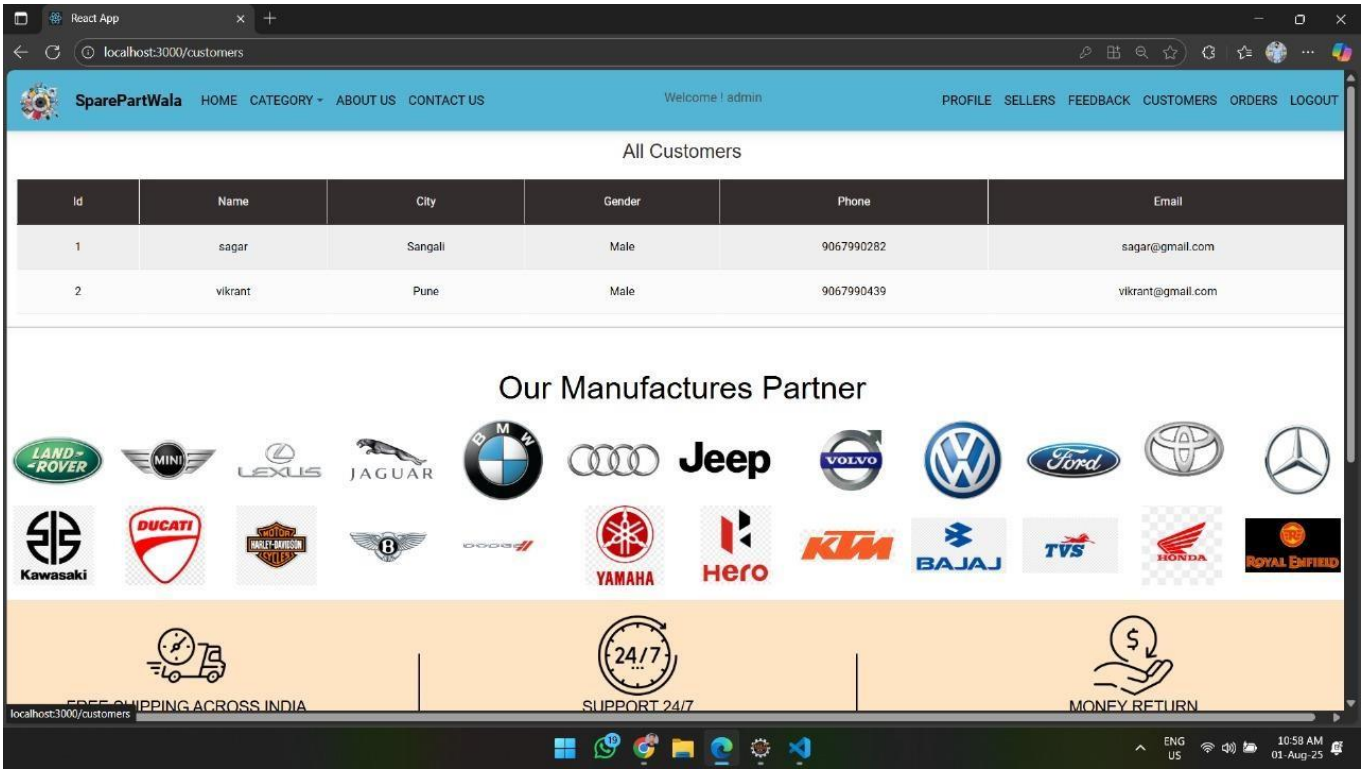
3. All orders details



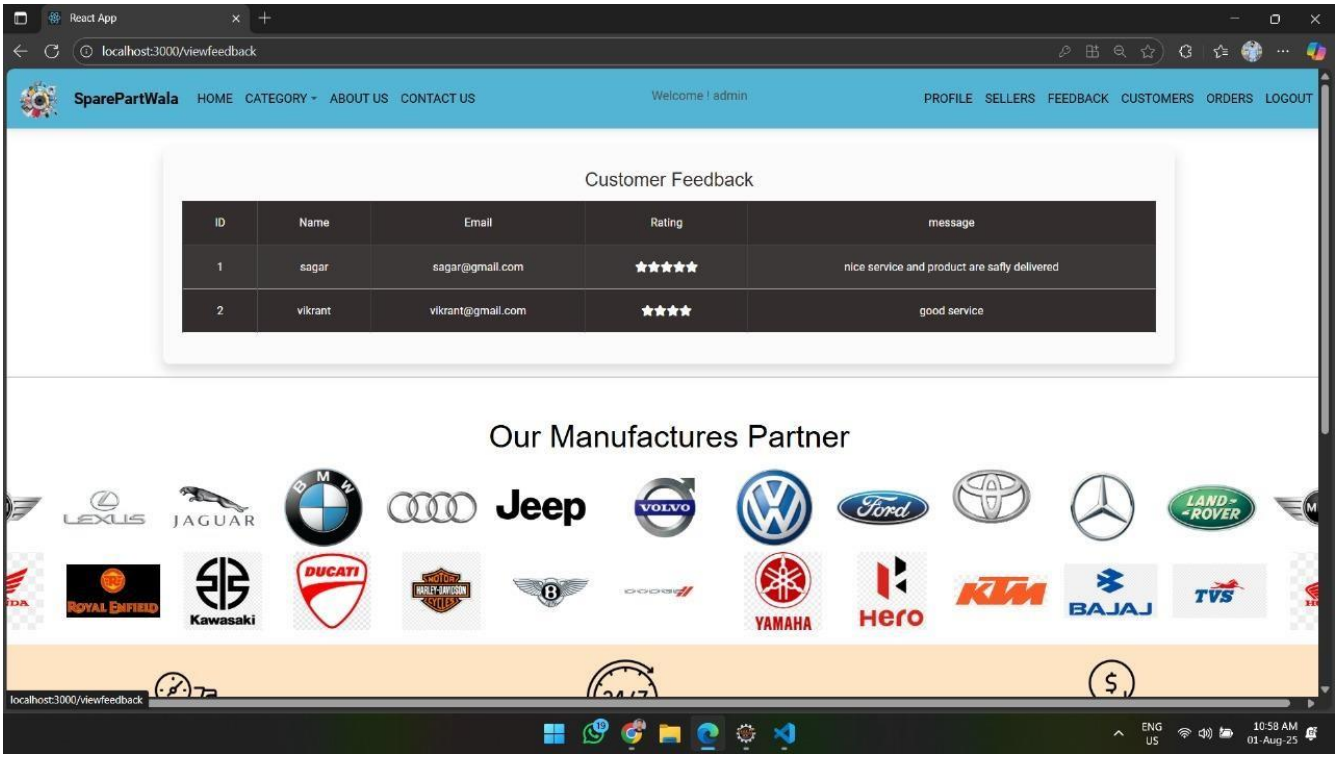
3. Sellers list



4. All Coustomers

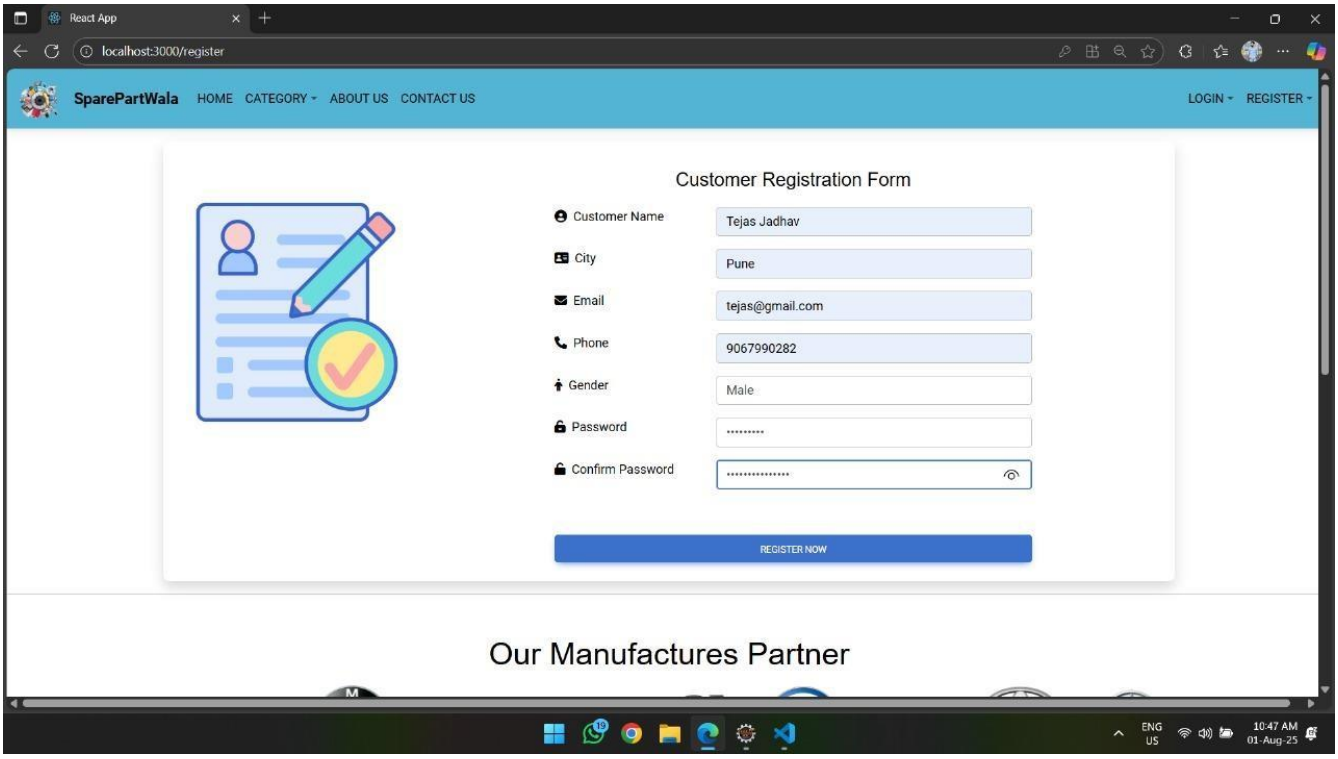


5. Coustomers Feedback

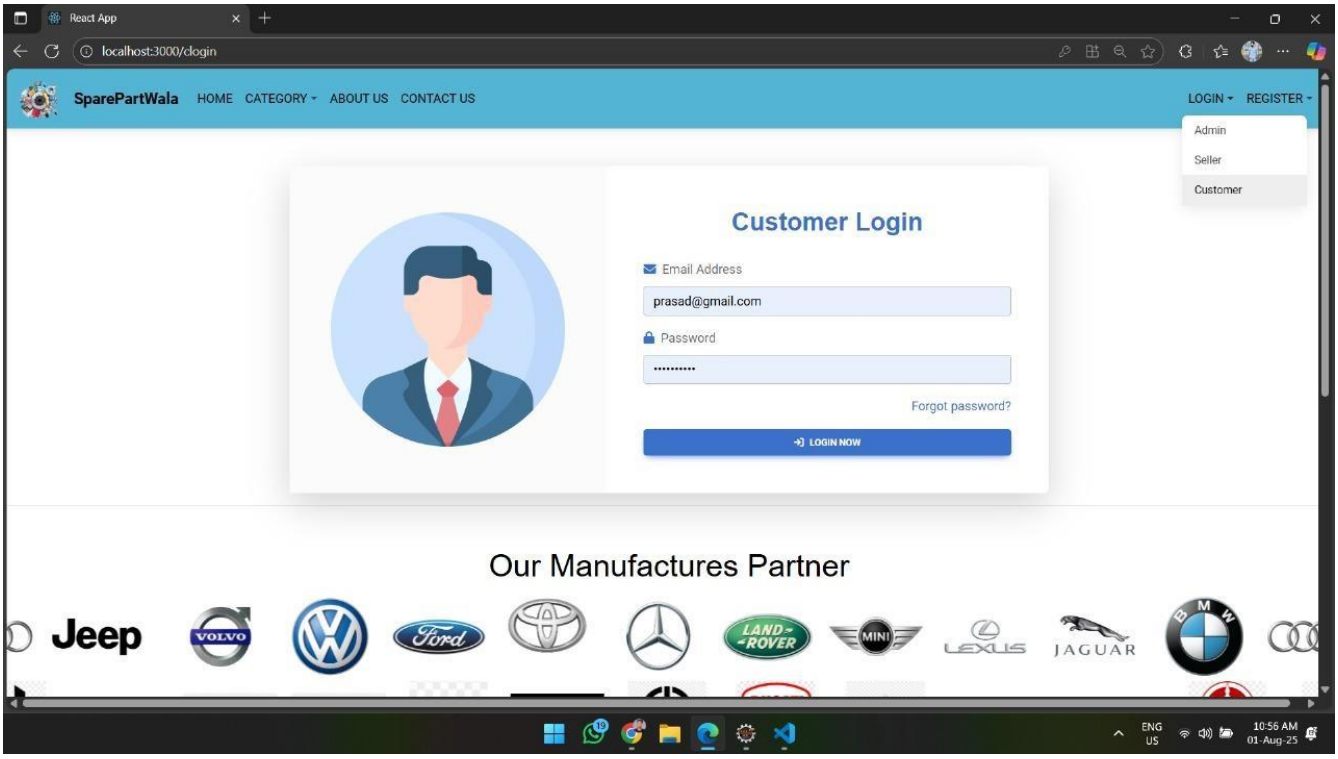


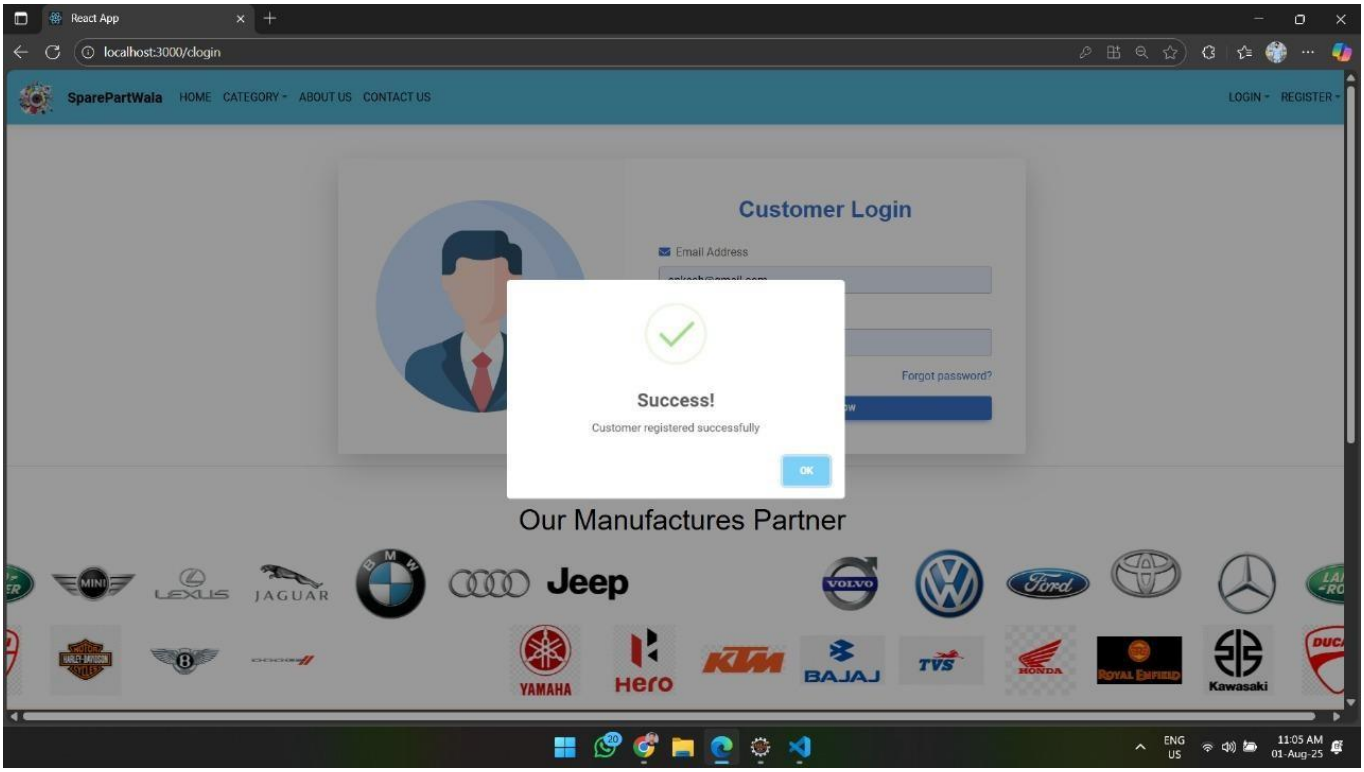
Coustomer

1. Coustomer Registration Form

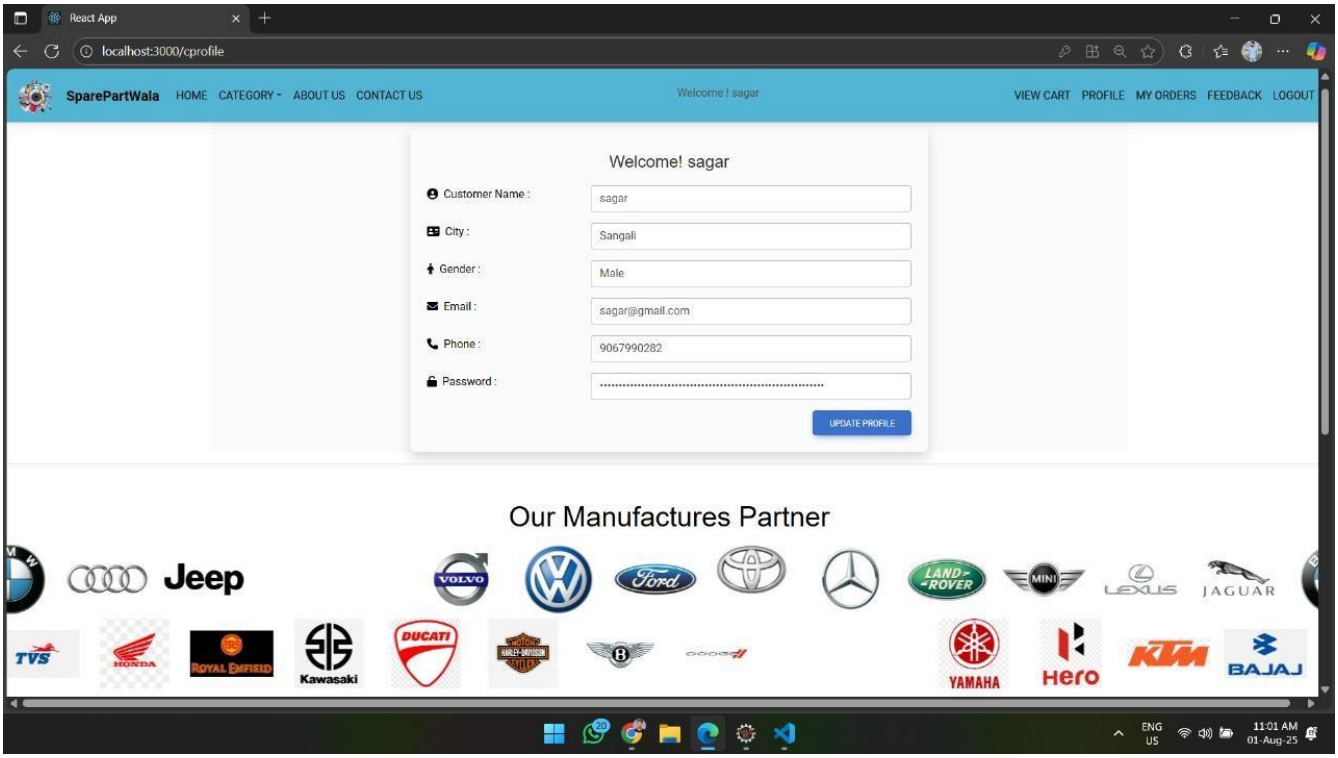


2. Coustomer Login

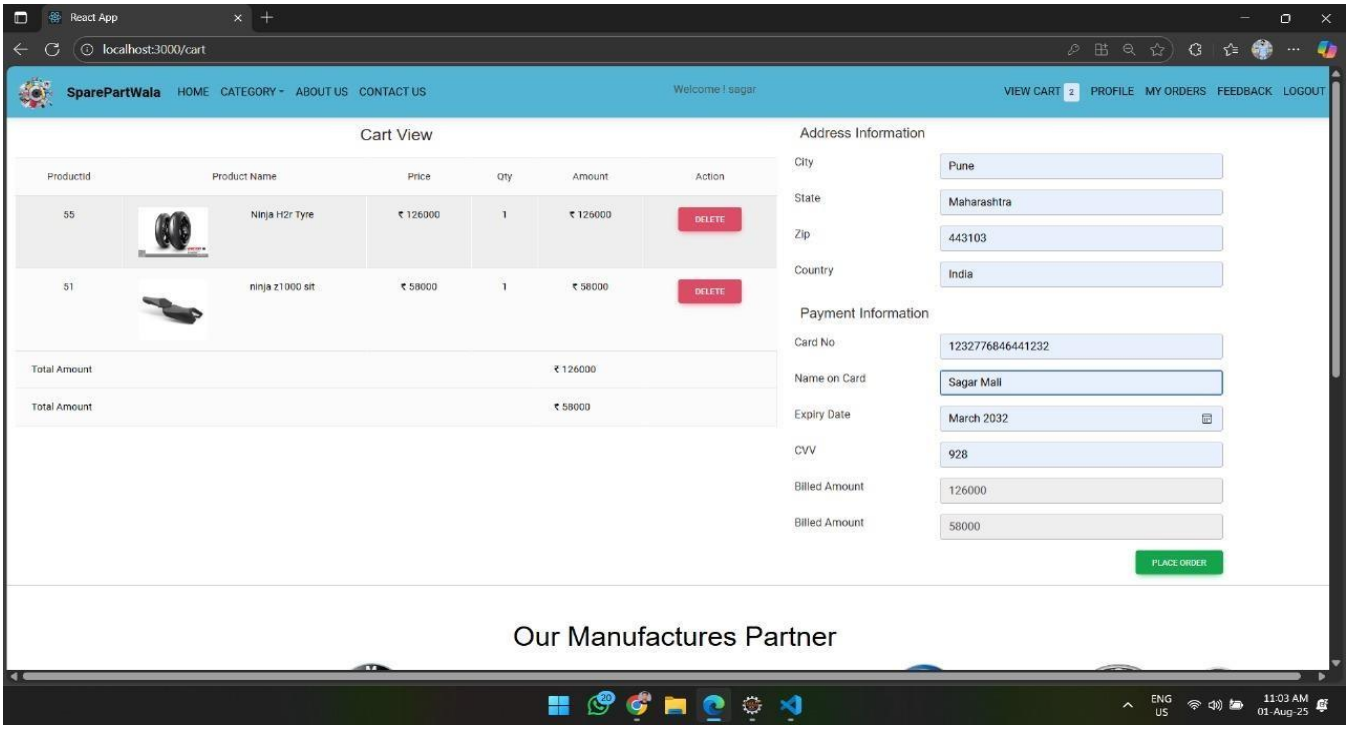




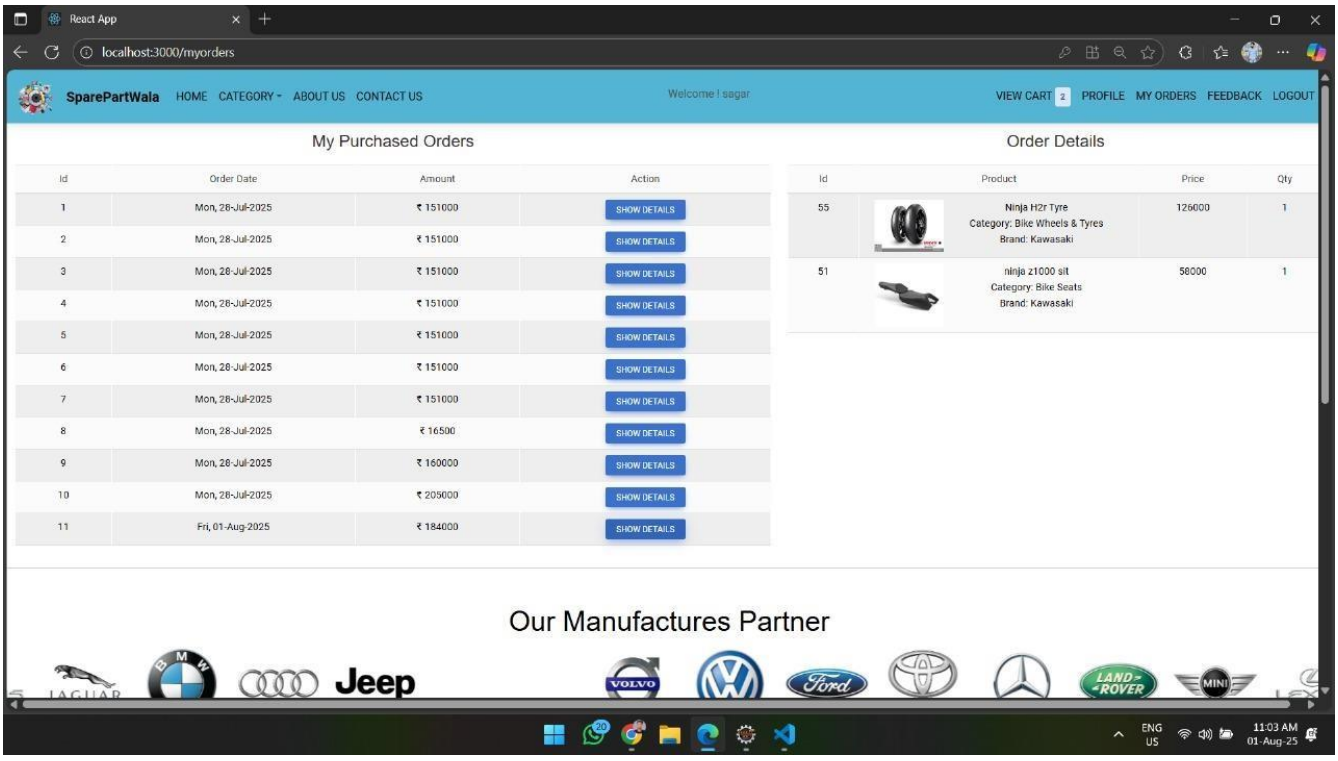
3. Coustomer Update profile



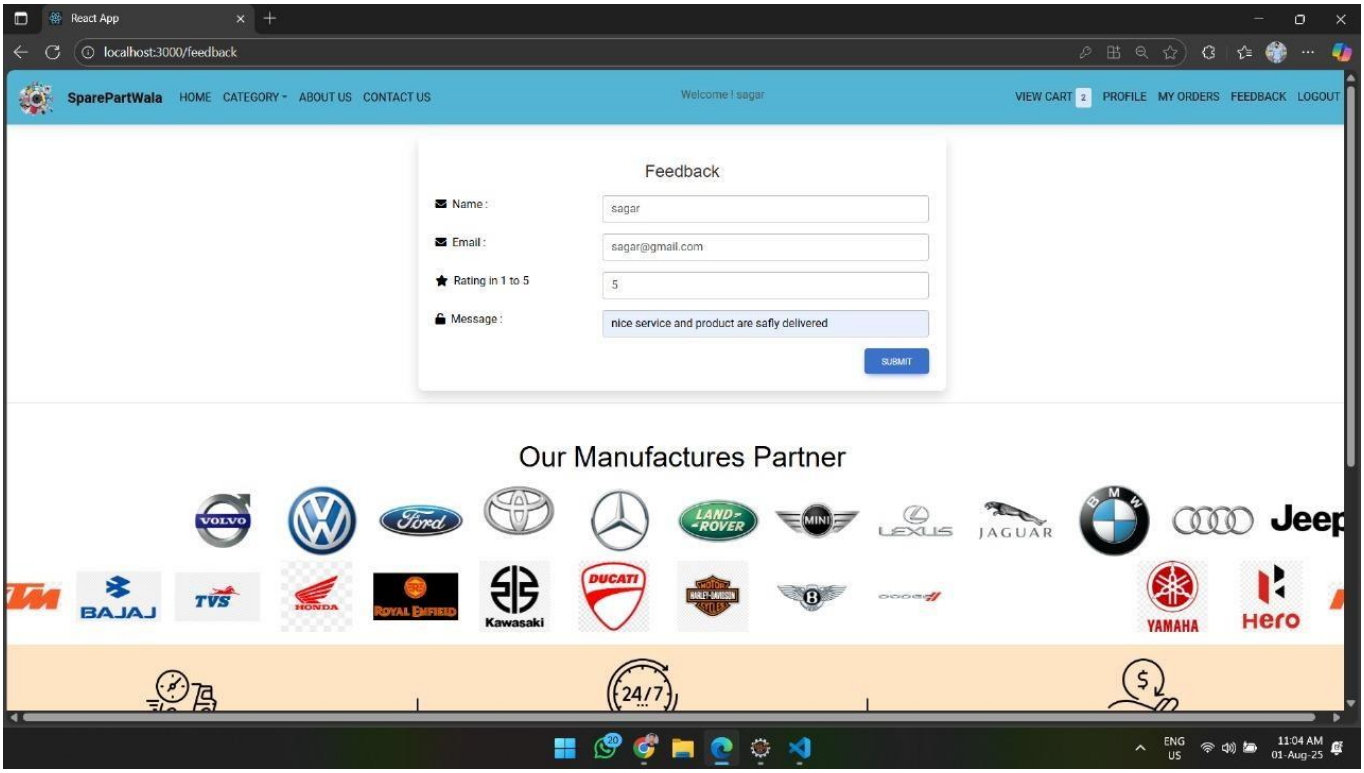
4. View Cart



4. My Purchase Orders

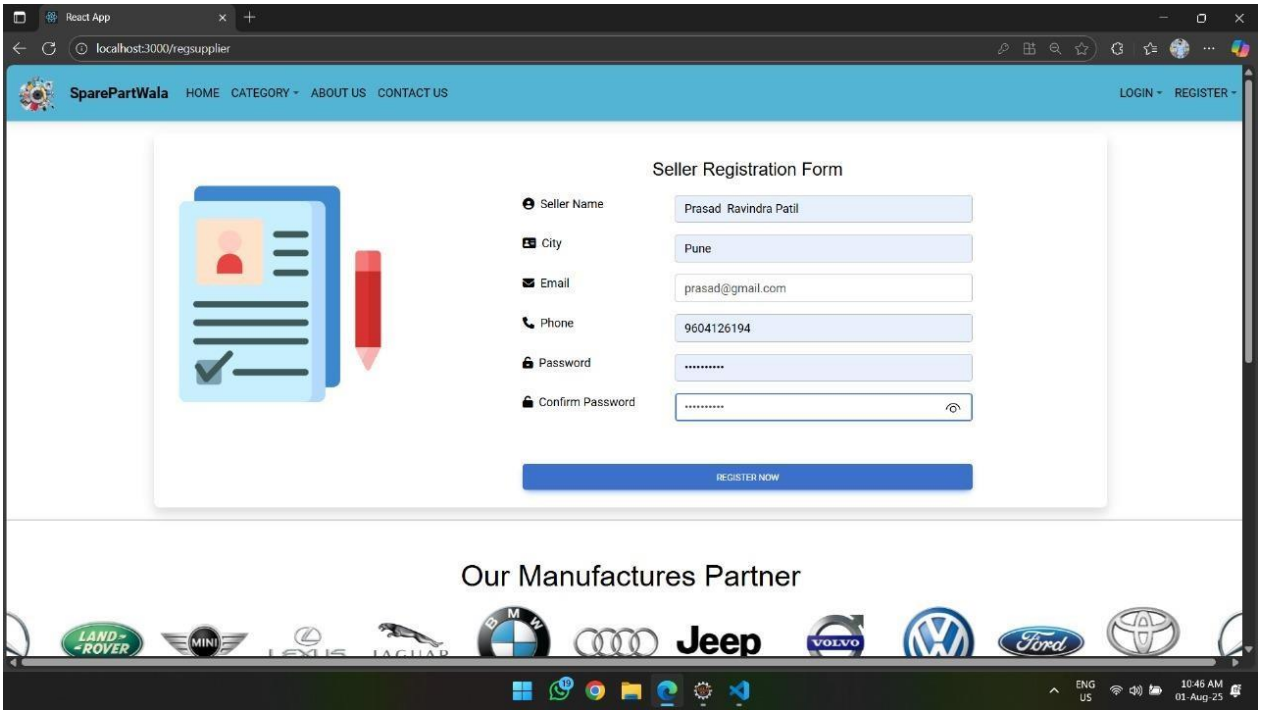


5. Feedback paage

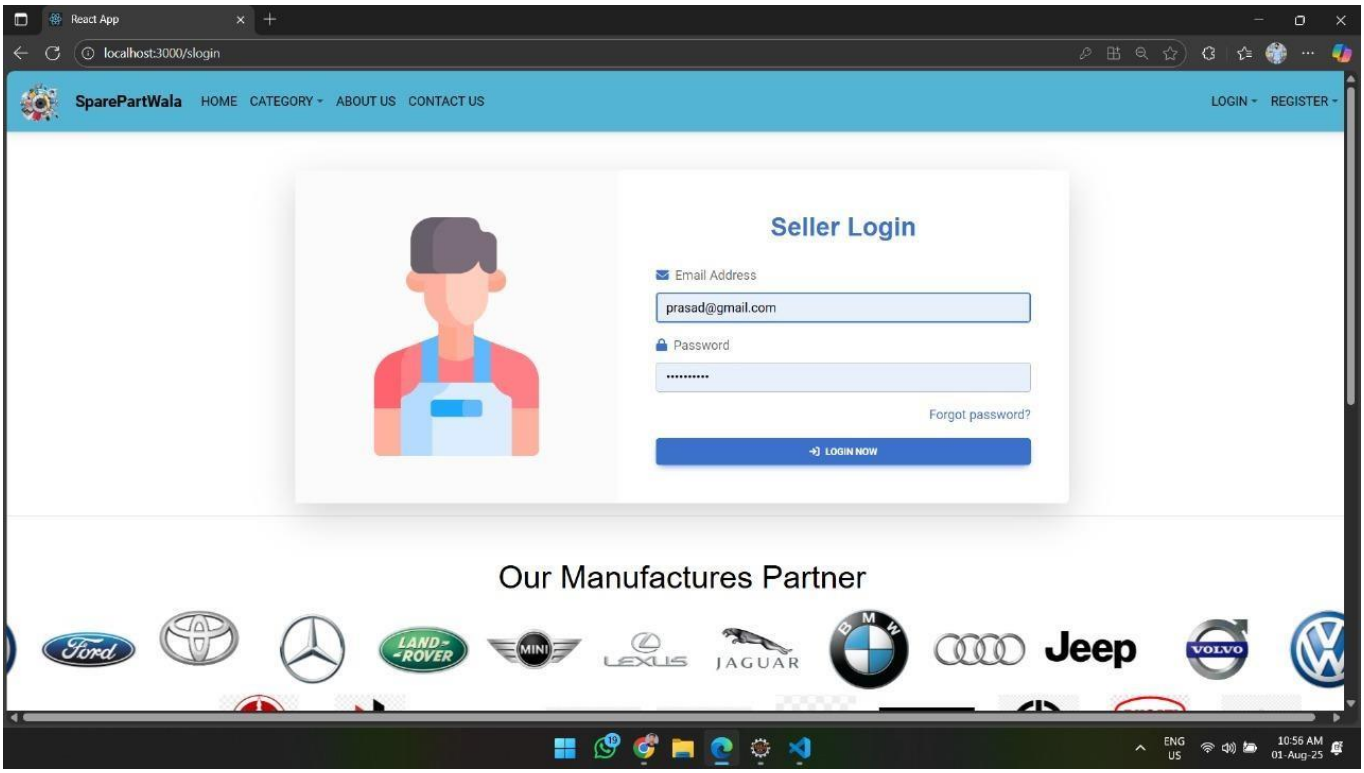


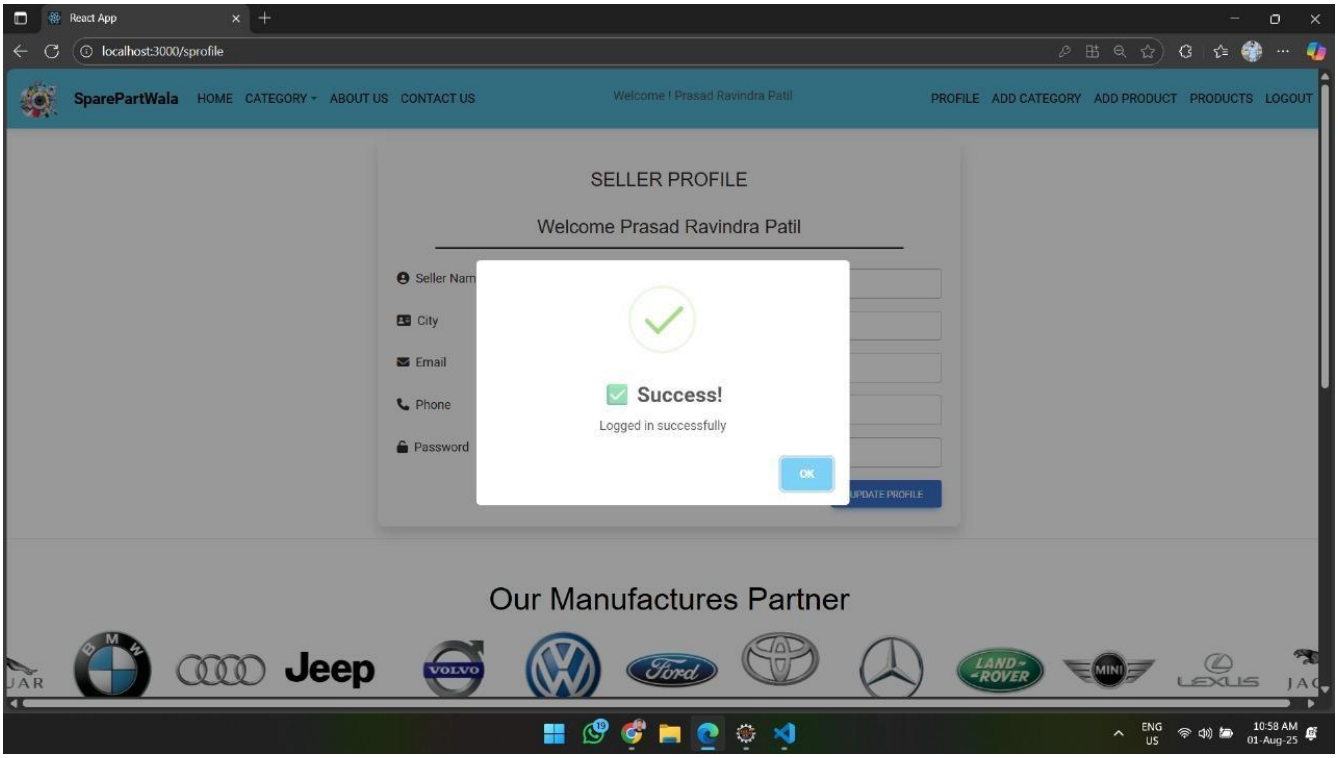
Seller

1. Seller Registration Form

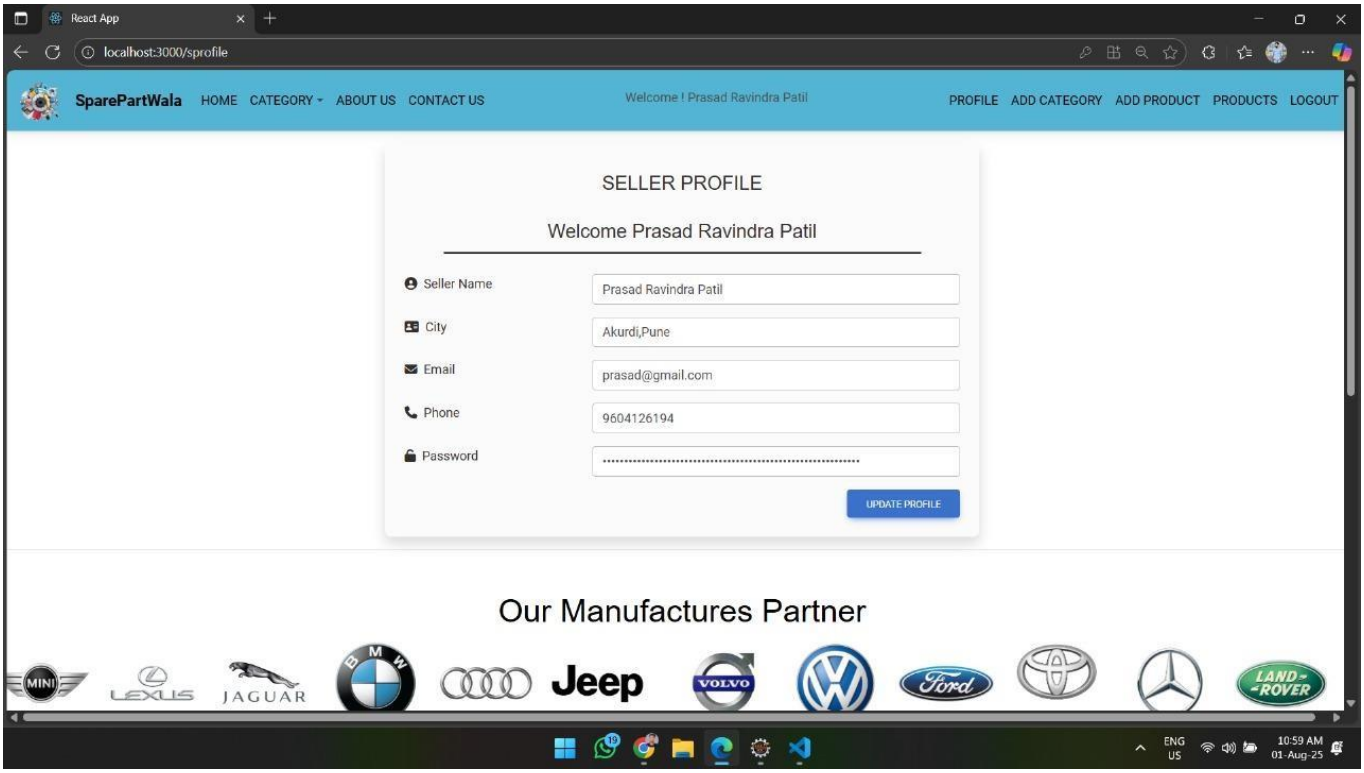


2. Seller Login Form

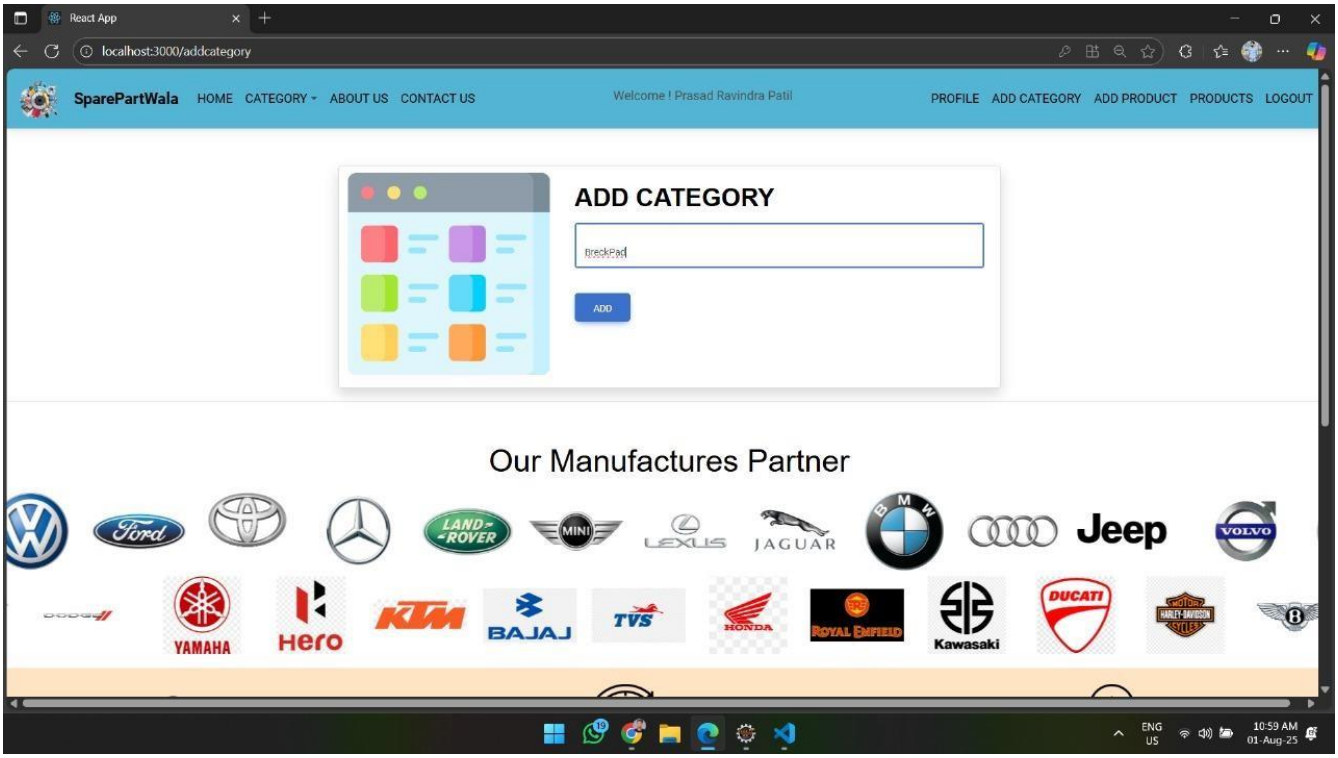




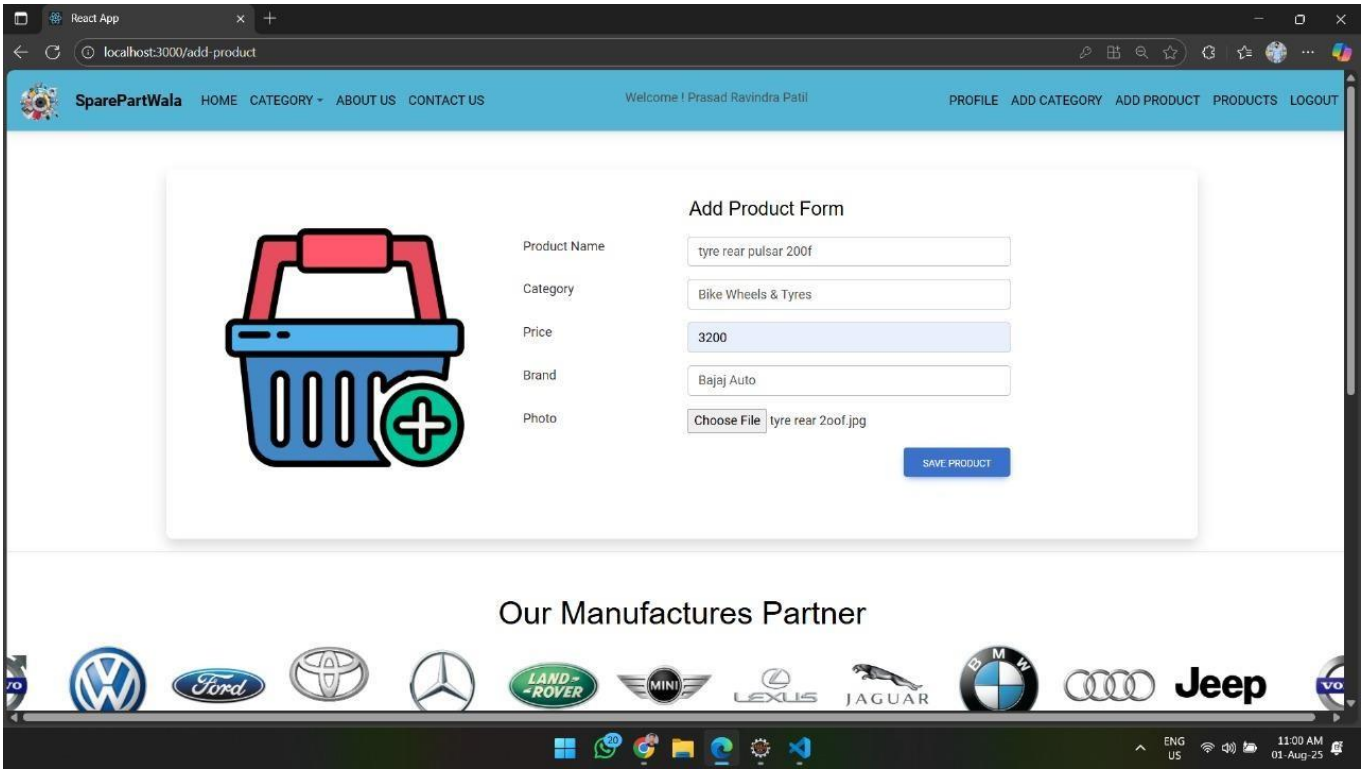
3. Seller Profile



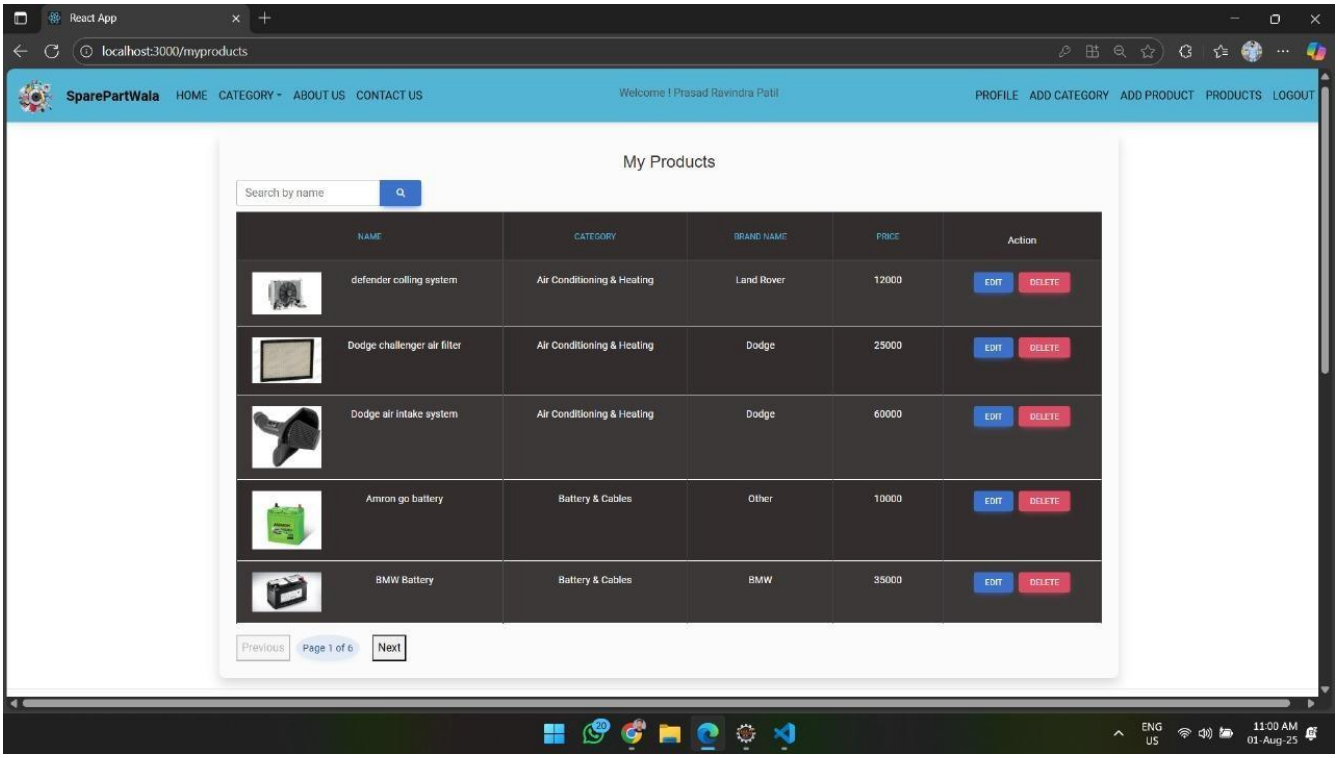
4. Add Category



5. Add Product Form



6. My Products



10. References

[1] ReactJS Documentation. [Online]. Available: https://reactjs.org

[2] Spring Boot Documentation. [Online]. Available: https://spring.io/projects/spring-boot

[3] GeeksforGeeks, “Spring Boot and ReactJS Tutorials.” [Online]. Available: https://www.geeksforgeeks.org

[4] YouTube, “Full Stack Development with Spring Boot and ReactJS.” [Online]. Available: https://www.youtube.com

[5] ChatGPT, OpenAI. "AI-based assistance for code generation and documentation," [Online]. Available: https://chat.openai.com

[6] W3Schools, “HTML, CSS, and JavaScript Tutorials.” [Online]. Available: https://www.w3schools.com

[7] Stack Overflow, “Developer Community for Troubleshooting and Code Help.” [Online]. Available: https://stackoverflow.com

[8] MySQL Documentation. [Online]. Available: https://dev.mysql.com/doc/

[9] MDN Web Docs, “JavaScript and Web Technology Guides.” [Online]. Available: https://developer.mozilla.org

[10] Draw.io, “Free Online UML and Architecture Diagram Tool.” [Online]. Available: https://www.draw.io