ELECTRONIC STORE

A PROJECT REPORT

Submitted by

ANSHUL JAYESHBHAI PATEL

210670107047

In partial fulfillment for the award of the degree of

BACHELOR OF ENGINEERING

In

Computer Engineering

SAL Institute of Technology and Engineering

Research, Ahmedabad

Gujarat Technological University, Ahmedabad

May, 2025

SAL Institute of Technology and Engineering Research

Opposite Science City, Sola, Ahmedabad, Gujarat – 380060

CERTIFICATE

This is to certify that the project report submitted along with the project entitled **Electronic Store** has been carried out by **Anshul Jayeshbhai Patel** under my guidance in partial fulfillment for the degree of Bachelor of Engineering in Information Technology, 8th Semester of Gujarat Technological University, Ahmedabad during the academic year 2024-25.

Dr. Krishna Hingrajiya

Internal Guide

Dr. Nimisha Patel

Head of Department

TECH IT EASY
smarter - simpler - faster

H.O RAJ CARRYING CARGO PVT. LTD.
OLD LATI BAJAR, O/S RAIPUR GATE,
RAIPUR, AHMEDABAD-380001
Email: techiteasy11@gmail.com
Phone: 9723601139

Date: 24th January 2025

Dear **Ms, Sakshi Panchal**

Subject: Internship Offer Letter

We are pleased to offer you an internship as a **Backend Developer** with **Tech IT Easy**, following your application and subsequent discussions with us.

Your internship is scheduled to commence on **20th January 2025**. We kindly request you to confirm your acceptance of this offer within three (3) days from the date of issuance of this letter.

Additionally, we would like to inform you that upon joining our organization, you will be required to sign a confidentiality agreement. This agreement ensures that you do not disclose any confidential information during or after your engagement with Tech IT Easy.

We are excited to have you join our team and look forward to a productive collaboration.

For, Tech IT Easy,

____

I hereby agree that I will perform my duties at Tech IT Easy.

Accepted __________________

GTU CERTIFICATE

SAL Institute of Technology and Engineering Research

Opposite Science City, Sola, Ahmedabad, Gujarat – 380060

DECLARATION

We hereby declare that the Internship / Project report submitted along with the Internship /Project entitled **Electronic Store** submitted in partial fulfillment for the degree of Bachelor of Engineering in Computer Engineering to Gujarat Technological University, Ahmedabad, is a bonafide record of original project work carried out by me / us at **SAL Institute of Technology and Engineering Research** under the supervision of **Prof. Krishna Hingrajiya** and that no part of this report has been directly copied from any students' reports or taken from any other source, without providing due reference.

Name of Student	Sign of Student
Anshul Patel(210670107047)	____________________

ACKNOWLEDGEMENT

I wish to express our sincere gratitude to our External guide **Mr. Deepak Vishwakarma** for continuously guiding me at the company and answering all my doubts with patience. I would also like to thank my Internal Guide **Prof. Krishna Hingrajiya** for helping us through our internship by giving us the necessary suggestions and advice along with their valuable co-ordination in completing this internship.

We also thank our parents, friends and all the members of the family for their precious support and encouragement which they had provided in completion of our work. In addition to that, we would also like to mention the company personals who gave us the permission to use and experience the valuable resources required for the internship.

Thus, in conclusion to the above said, I once again thank the staff members of **OPL Innovate** for their valuable support in completion of the project.

Thank You
Anshul Patel

ABSTRACT

The ***Electronic Store*** *is a full-stack e-commerce web application that enables users to register, browse, and purchase electronic products online. The backend is developed using* ***Spring Boot*** *and* ***Java****, ensuring a secure and scalable API-driven architecture, while the frontend is built with* ***Angular*** *for a dynamic and responsive user experience.* ***MySQL*** *serves as the database to manage product listings, user data, orders, and transactions. This project simulates a real-world e-commerce platform with features like user authentication, product browsing, cart management, order placement, and payment integration, offering a complete online shopping solution.*

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

HTML	HyperText Markup Language
CSS	Cascading Style Sheets
API	Application Programming Interface
OPP	Object Oriented Programming
DBMS	Database Management System
URL	Uniform Resource Locator
ORM	Object-Relational Mapping
CRUD	Create,Read,Update,Delete
SQL	Structured Query Language
HTTP	Hypertext Transfer Protocol
REST	Representational State Transfer
FTP	File Transfer Protocol
DFD	Data Flow Diagram
UE	User Experience
CRM	Customer Relationship Management
ER	Entity Relationship
KPI	Key Performance Indicator
CSV	Comma-Separated Values
XML	Extensible Markup Language
SDK	Software Development Kit
JSON	JavaScript Object Notation

LIST OF CONTENTS

CHAPTER 1: COMPANY OVERVIEW

1.1 ABOUT THE COMPANY

1.1.1 Who We Are:

OPL is a forward-thinking, innovation-driven digital infrastructure and solutions provider, empowering financial institutions, corporates, MSMEs, and individuals through cutting-edge technology. At our core, we believe **"Innovation Is Good"**, and we embody this philosophy by delivering futuristic, inclusive, and impactful financial solutions that create real change — for companies, communities, and the country.

1.1.2 What We Do:

OPL designs and delivers **customised, scalable, secure, and intelligent technology solutions** that transform the way financial services are accessed and delivered. From enabling fast and seamless credit assessment and disbursals to building white-label lending platforms, we help clients streamline operations, optimise decision-making, and accelerate growth. Whether it's helping large banks innovate or empowering farmers and MSMEs through platforms like Jansamarth — we build solutions that work for everyone.

1.1.3 Why Choose Us:

- Innovation in Every Layer – From strategy to execution, innovation is embedded in every step.
- Comprehensive, Scalable Tech – Microservices architecture, AI-powered engines, and real-time analytics ensure unmatched scalability and performance.
- Customised & White-label Ready – We build what *you* need: fast, flexible, future-ready solutions.

- Secure & Reliable – With 24x7 monitoring, encrypted environments, and defence-in-depth security, your data and services are always protected.
- People-Powered Excellence – A 270+ member team with 250+ years of cumulative experience across 17+ domains.

1.2 AIM AND OBJECTIVE OF THE INTERNSHIP

Aim:

The aim of this internship is to bridge the gap between academic knowledge and real-world application by engaging in the design, development, and deployment of a full-stack e-commerce web application. Through this internship, the goal is to enhance domain-specific skills in full-stack development, strengthen understanding of modern software architecture, and apply future-oriented skills such as scalable system design and API integration.The aim of the internship provides a direction to the activities, helps to focus on a result, and to assess the result achieved.

- Before going on the internship, two important factors guiding your development should be taken into account when formulating the aim:

1. Connecting what you have learned (theoretical and practical knowledge on your subject field) with actual work experience, in order to complement your field specific skills and learn new ones.
2. Apply and analyze at least one future skill.
    - There can be one or two aims, but both development of field specific skills as well as future skills have to be represented.

CHAPTER 2

OVERVIEW OF THE DEPARTMENTS OF ORGANIZATION AND LAYOUT OF PRODUCTION/PROCESS BEING CARRIED OUT IN COMPANY

2.1 DEPARTMENTS OVERVIEW:

Frontend Development Team:

- Designs and develops interactive user interfaces using Angular.
- Implements responsive layouts for product browsing, cart management, and user registration..
- Handles routing, state management, and dynamic data binding for a seamless user experience.

Backend Development Team:

- Develops secure RESTful APIs using Java and Spring Boot.
- Manages business logic for features like authentication, product management, and order processing.
- Ensures API integration with frontend and database layers for full functionality.

Quality Assurance (QA) Team:

- Conducts unit, integration, and end-to-end testing to ensure application stability.
- Uses tools like Postman for API testing and automation scripts for regression testing.
- Validates all functional flows including login, cart operations, and order placement.

Deployment & DevOps Team:

- Manages deployment pipelines and CI/CD integration.
- Ensures secure deployment on cloud platforms (like AWS).

UI/UX & Design Team:

- Focuses on creating intuitive and user-friendly designs.
- Ensures consistency across various components and optimizes mobile responsiveness.

Sales & Marketing:

- Focuses on client acquisition, partnerships, and promotions.
- Ensures customer satisfaction and service quality.

2.2 TECHNICAL SPECIFICATION OF TECHNOLOGIES USED IN DEPARTMENT

1. Frontend :

- Angular 15+, HTML5, CSS3, TypeScript.

2. Backend:

- Java 17, Spring Boot (REST APIs).

3. Database:

- MySQL 8 (Relational DBMS for structured data).

4. Tools:

- Postman (API Testing), Git (Version Control), VS Code, IntelliJ IDEA.

5. Security:

- Spring Security (Authentication & Authorization), HTTPS.

2.3 SCHEMATIC LAYOUT

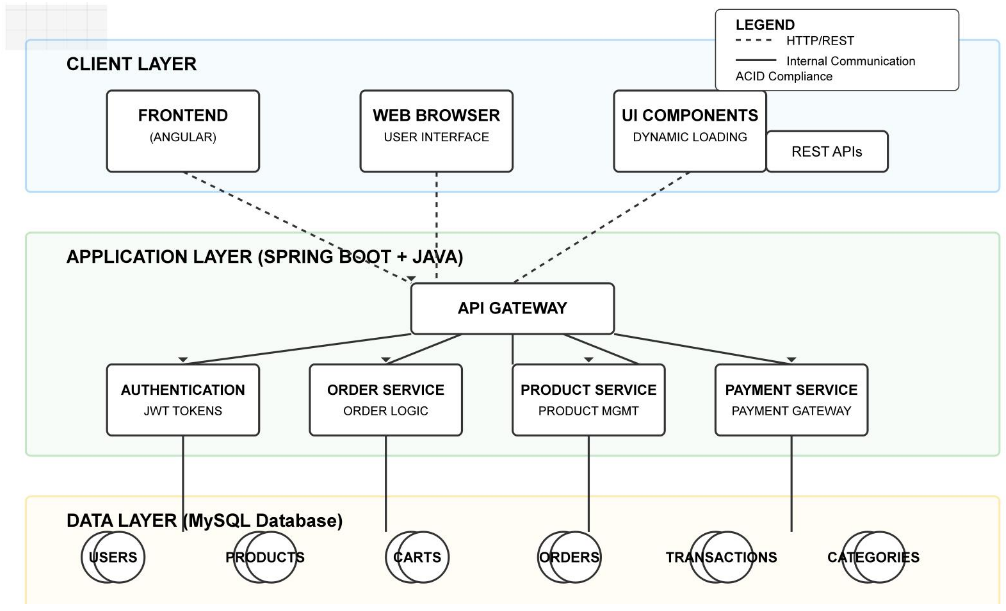


Fig 2.3.1 Sequence of Operation followed

System Flow Description:

The architecture of *The Electronic Store* is based on a **three-tier model:**

1. **Client Layer (Frontend – Angular)**

   i. User interacts via web browser.

   ii. UI dynamically loads product data, handles user inputs, and communicates with backend through REST APIs.

2. **Application Layer (Backend – Spring Boot + Java)**

   i. Handles user authentication, order logic, product management, and payment processing.

   ii. Processes requests, validates input, and returns responses to the frontend.

3. **Data Layer (MySQL Database)**

   i. Stores data on users, products, categories, carts, orders, and transactions.

   ii. Ensures ACID compliance and maintains relational integrity.

Authentication & Security

- JWT-based token management secures APIs.
- Role-based access control ensures only authorized users can perform restricted operations (e.g., admin functions).

Payment Integration

- Simulates or connects with payment gateways (e.g., Razorpay/PayPal) for order checkout.

2.4 DETAILED EXPLANATION OF EACH STAGE OF PRODUCTION

1. Planning

- Identified project scope: building a secure, scalable, and complete e-commerce solution.
- Listed key modules: user registration, product management, shopping cart, orders, and payments.
- Defined technology stack (Angular, Spring Boot, MySQL).

2. Requirements Analysis

- Drafted **Software Requirement Specification (SRS)** for each module.
- Defined user roles (admin, customer) and their access levels.
- Considered system requirements: concurrent users, load times, security compliance.

3. Design

- Designed frontend UI wireframes for product browsing, cart, and checkout pages.
- Defined REST API endpoints (e.g., /api/products, /api/orders, /api/auth/login).
- Created ER diagrams and relational database schema in MySQL.

4. Development

- **Frontend**: Angular components for Login, Product List, Cart, Checkout, Order Summary.

- **Backend**: Spring Boot REST APIs for login, registration, product and order CRUD operations.
- Integrated payment APIs and implemented input validation.
- Applied MVC architecture and dependency injection principles.

5. Testing

- Manual testing of frontend modules.
- API testing via **Postman** for authentication, order placement, and product listings.
- Performed unit tests for backend services and data layers.
- Ensured system handles edge cases (e.g., invalid logins, stock unavailability).

6. Deployment

- Deployed application on a cloud platform like AWS.
- Used Git for version control and CI/CD for smooth deployment.
- Secured backend using HTTPS and environment variables for database and payment keys.

CHAPTER 3: INTRODUCTION OF PROJECT

This chapter provides a comprehensive iThis chapter provides a comprehensive introduction to the Electronic Store e-commerce project. It includes the project summary, purpose, objectives, scope, technologies used, planning, justification, cost estimation, roles, dependencies, and scheduling timeline.

3.1 PROJECT SUMMARY

The **Electronic Store** is a full-stack web application designed to allow users to browse, purchase, and track electronic products online. It features a responsive frontend developed using **Angular**, a secure backend built using **Java and Spring Boot**, and a **MySQL** database to handle product listings, orders, and user information.

Users can register, log in, view products, add items to their cart, and place orders. Admins can manage product categories, add or update products, view all users, and oversee all orders via a comprehensive admin dashboard. The platform offers role-based access control (Admin vs User) and simulates the functionality of a real-world online electronics shopping system.

3.2 PURPOSE

1. **User-Friendly Shopping:** Provide a modern and efficient platform for users to shop for electronic goods with ease.
2. **Admin Control:** Empower admins with tools to manage the store—products, categories, orders, and users.
3. **Seamless Cart & Order System:** Enable real-time cart updates, order tracking, and checkout flows.
4. **Secure Login System:** Ensure data security and role-based authentication.
5. **Practice Full-Stack Skills:** Implement the project using industry-standard technologies to simulate real-time e-commerce workflows.

3.3 OBJECTIVES

- Implement a **secure registration and login system** for both users and admins.
- Build an **interactive UI** for browsing and filtering electronic products.
- Allow users to **add products to a cart**, **place orders**, and **track their order status**.
- Enable admins to:
  - Add/View/Edit/Delete product categories.
  - Add/View/Edit/Delete products.
  - View all users.
  - Monitor and manage all orders.
- Develop a responsive and dynamic UI using **Angular**.
- Ensure backend APIs are efficient, secure, and well-documented using **Spring Boot**.
- Use **MySQL** to manage all transactional and product-related data.
-

3.4 SCOPE

1. **Frontend Features:** The frontend interface, developed in Angular, provides a responsive and intuitive experience for users. The homepage displays a categorized product catalog, allowing users to filter products by type, brand, or price. Users can register or login using their credentials and start adding products to their shopping cart. The cart page allows quantity updates, product removal, and displays a detailed pricing summary. Upon checkout, users can place orders, which are recorded and shown on their order tracking page. The tracking page displays order status such as "Placed", "Shipped", "Out for Delivery", or "Delivered", offering full visibility into order progress.

2. **Admin Panel Features:** The admin panel serves as the backend interface for managing the store's data. Admin users can create, edit, or delete product categories, and manage the product catalog through a tabular view, which includes columns for product name, price, stock status, and category. Admins can also view the entire user database, which helps in monitoring user activity and handling support queries. Moreover, all orders placed by customers are

viewable in the admin dashboard, where status updates can be made, such as marking an order as shipped or delivered. This ensures centralized control over store operations and helps maintain data consistency.

3. **Security & Access:** Security is a foundational aspect of the *Electronic Store* project. The system uses JWT (JSON Web Token) based authentication to ensure that each session is securely verified and tamper-proof. Role-based access control (RBAC) is implemented, where each user is assigned a role such as Admin or User, and permissions are granted based on their role. This prevents unauthorized access to restricted features such as order management or product editing. Additionally, input validation and sanitization are enforced both at the client and server side to prevent common vulnerabilities like SQL Injection and XSS (Cross-Site Scripting). Passwords are stored using hashing algorithms, ensuring user credentials are secure. All API requests are authenticated and unauthorized access is gracefully handled through proper HTTP status responses and error messages.

4. **Scalability:** The system architecture of the *Electronic Store* is designed with scalability at its core. Built using Spring Boot and Angular, both frontend and backend are modular, enabling seamless extension of features without refactoring core components. For instance, integrating third-party payment gateways like Razorpay or PayPal can be done with minimal changes. The backend APIs are RESTful, making it easy to scale horizontally using load balancers. On the database level, MySQL can be configured for read replicas to handle increasing read requests. Furthermore, the application can support additional user roles beyond Admin and User, such as SuperAdmin, Support Staff, or Warehouse Manager, which can be implemented using Role-Based Access Control (RBAC) without major architectural changes. This flexibility ensures that the application can grow along with business needs, user base, and new functional requirements.

5. **Deployment:** The *Electronic Store* application is fully designed and tested for smooth deployment on modern cloud platforms like AWS (Amazon Web Services), Render, and Heroku. The backend, built using Spring Boot, is containerized using tools like Docker (if configured) to ensure consistency

across development and production environments. The frontend, developed in Angular, can be compiled and served using Nginx or hosted directly via services like Netlify or Vercel. Environment configurations are managed through property files and environment variables to securely handle API endpoints and database connections during deployment. The application structure supports CI/CD pipelines, allowing automated testing and deployment whenever new features are pushed. This readiness ensures the application can go live with minimal downtime and provide reliable service to users at scale.

3.5 TECHNOLOGY AND LITERATURE REVIEW

- **Frontend:** Angular
  - Component-based UI development
  - Reactive forms, routing, and state management
  - HTTP services to communicate with backend
- **Backend:** Spring Boot
  - REST API creation with CRUD operations
  - Security with Spring Security & JWT
  - Business logic handling for orders, products, and authentication
- **Database:** MySQL
  - Relational data storage
  - Tables: Users, Products, Categories, Orders, Cart
- **Authentication:** JWT (JSON Web Tokens)
  - Secure session management
  - Role-based access control
- **Tools & IDEs**:
  - VS Code (Frontend)
  - IntelliJ IDEA (Backend)

- MySQL Workbench
- Postman (API Testing)
- Git/GitHub (Version Control)

- **Literature Review Includes**:
  - Spring Boot REST API Design Best Practices
  - Angular Component Architecture
  - Secure Login with JWT in Spring Boot
  - Relational Database Schema Design for E-Commerce

3.6 PROJECT PLANNING

1. Requirement Gathering:

- Identify core features: user login, cart, product browsing, admin dashboard.
- Understand user roles and permissions.
- Plan database schema.

Deliverables:

- Functional specification document
- API endpoint list
- Database schema design

2. System Design:

- Design UI wireframes for frontend pages.
- Create backend API structure using Spring Boot.
- Plan entity relationships for MySQL (e.g., User → Orders → Products).

Deliverables:

- ER diagram
- UI mockups
- API route map

3. Database Setup:

- Create MySQL schema with relations.
- Sample data insertion for testing.

Deliverables:

  - SQL scripts for creating tables
  - Insert scripts for sample products, users

4. **Frontend and Backend Development:**

- Develop frontend modules (Angular):
  - Login, Register, Product List, Cart, Orders
- Develop backend modules (Spring Boot):
  - Auth API, Product API, Order API, Admin APIs

Deliverables:

  - Complete frontend & backend source code
  - Working APIs with Postman testing

5. Integration & Testing:

- Connect Angular with Spring Boot APIs.
- Unit and integration testing using JUnit (backend) and Jasmine (frontend).

Deliverables:

  - Test reports
  - Integrated frontend-backend system

6. Deployment & Documentation:

- Host backend and frontend on cloud or localhost.
- Create user manual and API documentation.

- Provide database backup scripts and deployment steps.

Deliverables:

- Live app demo (if hosted)
- Final project report
- Deployment guide

3.6.1 Project Justification

E-commerce is a booming industry, and learning how to build a real-world, scalable, and secure e-commerce application is invaluable. This project offers exposure to full-stack development, security practices, database management, and real-time data handling—making it a perfect educational tool and a stepping stone for real-world projects.

3.6.2 Cost estimation of the project

Resource	Estimated Cost
Developer Time (Student Work)	No cost
Tools & IDEs (Open Source)	Free
Database (MySQL)	Free
Deployment (if using cloud)	~₹500–₹1000/month (optional)
Documentation Tools	Free (Google Docs, MS Word)

3.6.3 Roles and Responsibilities

- **Frontend Developer**:
  - Develop UI components using Angular.
  - Integrate with backend APIs.
  - Handle routing, form validations, and UI feedback.

- **Backend Developer**:
    - Create REST APIs using Spring Boot.
    - Implement business logic and database interaction.
    - Ensure secure login with JWT.
- **Database Administrator**:
    - Design schema.
    - Create SQL scripts.
    - Optimize queries for performance.
- **QA/Tester**:
    - Write test cases.
    - Perform bug tracking and issue reporting.

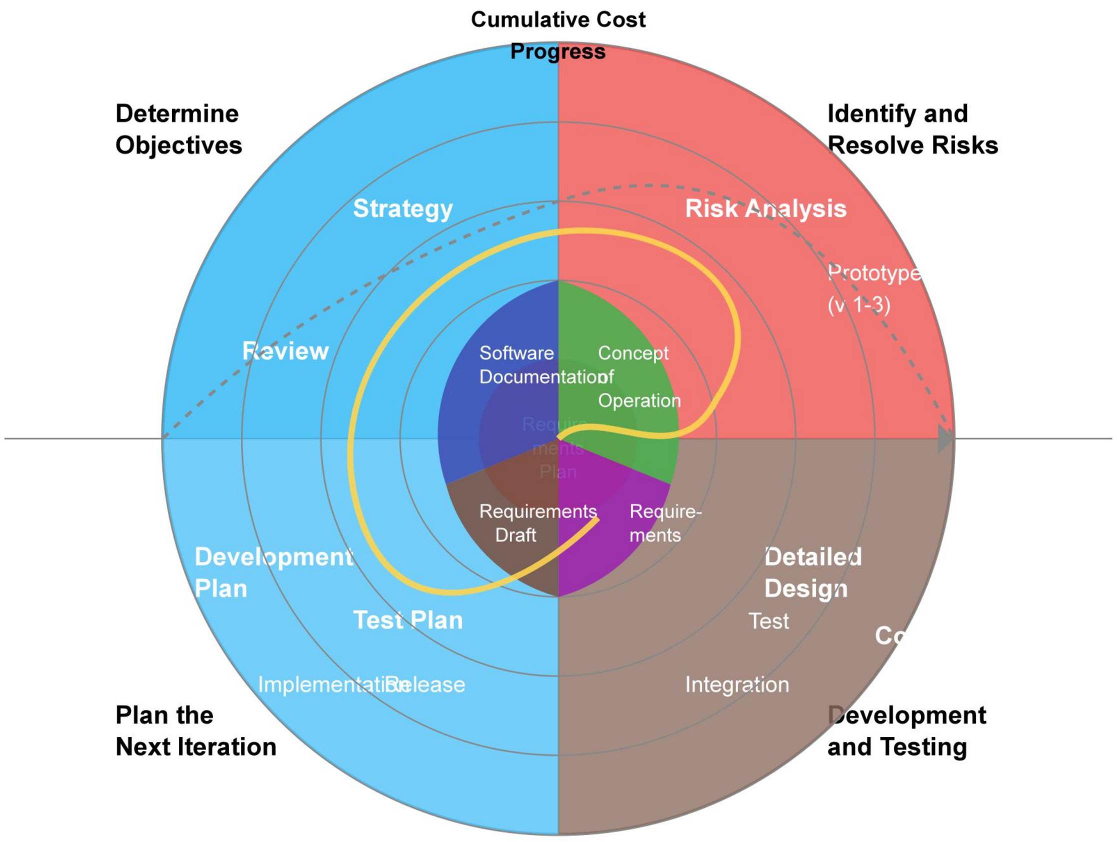
3.6.4 Group Dependencies

- **Frontend depends on**:
    - API readiness and structure
    - Proper error handling from backend
- **Backend depends on**:
    - Clear API documentation
    - Accurate database schema
- **Database team needs**:
    - Inputs from backend for table structure
    - Coordination with both frontend and backend for relational mapping

3.7 PROJECT SCHEDULING

Week 1	• Project Initialization. • Dependency Management. • Database Configuration.
Week 2	• Define entity classes and establish relationships for database consistency. • Implement repository layer for optimized data retrieval and transactions. • Develop API endpoints to facilitate CRUD operations efficiently.
Week 3	• Develop API controllers to expose service functionalities for frontend interactions. • Optimize business logic to improve execution efficiency. • Introduce structured error-handling mechanisms.
Week 4	• Security Implementation. • API Documentation. • API Endpoint Development.
Week 5	• Enhance security features with OAuth integration. • Implement logging mechanisms for monitoring API requests. • Optimize API response times to improve frontend interactions.
Week 6	• Angular Setup. • Bootstrap with Angular. • Angular App Structure. • Signup module.

Week 7	● Login Module. ● Creating Login form. ● Request backend for Login,NgRX Setup. ● Guard:Protecting normal user routes. ● Protecting admin routes using guard.
Week 8	● Admin Dashbord Section. ● Creating admin layout. ● Creating add category form. ● Creating HTTP interceptor. ● Create Category View,Delete Category.
Week 9	● Create User View Component. ● Update User Form with dynamic data. ● Complete user uodate with image. ● Implementing infinite scroll with user. ● Showing user to admin panel.
Week 10	● Create store page and showing live products. ● Filter category wise product. ● Showing Product Detail. ● Complete Cart Page Design. ● Sharing cart data to other component,Increase Decrease quantity button.
Week 11	●Loading orders form backend. ●Displaying order detail on modal. ●Creating Update order by admin. ●Creating view orders for user in single api call.
Week 12	●Made tbb_billing_statement API and changed the print positions. ●Made pod-receive-report API and get the all pod_receive_report data.



3.1 Spiral Model

CHAPTER 4: SYSTEM ANALYSIS

4.1 STUDY OF CURRENT SYSTEM

The current e-commerce systems are either paid solutions that often lack customization or are open-source solutions that do not provide adequate user experience and functionality for an electronic store. Many existing systems lack flexibility and do not offer a comprehensive solution for both the customer-facing experience and the administrative control needed by business owners. Furthermore, the lack of real-time data synchronization, cart management, and order tracking leads to a fragmented user experience and inefficiency. For example, users are unable to view the latest updates regarding product availability or order status, and admin features like inventory management and order processing often require manual intervention.

4.2 PROBLEMS AND WEAKNESS OF CURRENT SYSTEM

The most critical issues with the current system can be summarized as follows:

- **Lack of Admin Control:** Existing platforms do not provide sufficient administrative functionalities for managing products, user data, or orders. Admins often struggle to update the product catalog or manage orders without facing limitations in the user interface.
- **No Category-Based Filtering:** The lack of product categorization and filtering options makes it difficult for users to find relevant products, leading to frustration and abandonment. This is particularly problematic in an electronic store where products span various categories (e.g., laptops, mobile phones, accessories).
- **Poor UI/UX:** Many systems fail to provide an engaging and intuitive user interface. The user experience (UX) design is often complicated, leading to slow navigation, poor visual presentation, and a lack of trust in the platform. This can deter customers from completing their purchases.

4.3 REQUIREMENTS OF NEW SYSTEM

The new **Electronic Store** system addresses the shortcomings of the previous systems and introduces a variety of features:

- **Secure Authentication**: The system will include robust JWT-based authentication to ensure that users' data is secure. Passwords will be hashed and sensitive user data will be protected against unauthorized access.
- **Role-Based Access Control (RBAC)**: The system will provide role-based access, allowing differentiation between users and admins. This enables a secure and organized flow of control, ensuring that only authorized users can access certain features such as adding products or viewing sensitive user data.
- **Real-Time Cart and Order System**: The system will implement a real-time cart system, allowing users to instantly update their cart items, view prices, and apply promotions. Additionally, the order system will track the status of each order in real-time, providing customers with accurate and up-to-date information about their purchases.

4.4 SYSTEM FEASIBILITY

4.4.1 Contribution to Organization

The Electronic Store system is expected to significantly contribute to the organization by improving customer experience and business management. By implementing this system, the business will gain a comprehensive understanding of e-commerce workflows, which includes product management, user interaction, and order fulfillment. This system also provides opportunities for expanding the platform in the future, including the integration of additional features like payment gateways, inventory management, and real-time shipping data. Furthermore, it will facilitate full-stack development skills within the organization by using technologies like Spring Boot, Angular, and MySQL.

4.4.2 Technological Viability

The system is built using modern and widely adopted technologies, ensuring its scalability, security, and efficiency. Spring Boot ensures a scalable and flexible backend, while Angular provides a dynamic, single-page application frontend. MySQL is used for database management, offering robust features such as relational data handling, indexing, and transactions. These technologies are not only reliable but are also well-documented and have active communities, which makes them a feasible choice for building a robust e-commerce solution. Furthermore, these technologies allow seamless integration with third-party tools like payment gateways (Razorpay, PayPal) and shipping APIs.

4.4.3 Integration

The Electronic Store system is designed to integrate easily with various third-party services. For example, payment systems like Razorpay or PayPal can be seamlessly integrated to handle transactions, providing users with a variety of payment options. Additionally, the system can be integrated with external shipping APIs to automatically update users about the shipping status, such as tracking information and expected delivery dates. This integration with third-party APIs allows for easy enhancement of the system's capabilities as the business grows.

4.5 PROCESS IN NEW SYSTEM

The following process flow outlines the user experience in the new **Electronic Store** system:

- **User Registration/Login**: A user registers or logs into the system using a secure authentication process. The login page will also provide options for password recovery.
- **Product Selection and Cart Addition**: After logging in, the user can browse through various product categories, select the desired items, and add them to the shopping cart. Each product will display essential details such as price, description, and available stock.
- **Order Placement**: Once the user is ready, they can proceed to the checkout page, where they will provide shipping details, review their cart, and place the order. At this stage, the system will validate the order and calculate the total price (including taxes, discounts, etc.).
- **Admin Order Processing**: The admin will then process the order by updating its status (e.g., "Shipped," "Delivered") and manage inventory by adjusting stock levels accordingly.

4.6 FEATURES OF NEW SYSTEM

The **Electronic Store** system introduces several features that improve both user and admin experiences:

- **Responsive UI**: The user interface will be optimized for both desktop and mobile devices, ensuring a smooth and responsive experience regardless of the device being used.
- **Category & Product Management**: The admin can manage products and categories from the admin dashboard. This includes adding, editing, or deleting product entries, as well as managing product categories to ensure they are well-organized.
- **Order Tracking**: Both users and admins will be able to track orders in real-time. Users can view the status of their order (e.g., pending, shipped, delivered), while admins can manage and update the order statuses from the admin panel.
- **Real-Time Validation**: The system will include real-time validation for user inputs, such as checking if the email already exists during registration or ensuring that all required fields are filled in during the checkout process.

4.7 LIST MAIN MODULES

The new system will be divided into the following main modules, each serving a unique function:

- **Authentication Module**: This module handles user login and registration, including JWT token generation for secure authentication and session management.
- **Product Management Module**: Admins will use this module to add, update, or delete products from the store's catalog. It includes functionalities for managing product names, prices, descriptions, images, and stock levels.
- **Cart System Module**: Users can view, add, or remove products from their shopping cart. This module updates the cart in real-time, reflecting any changes made by the user.
- **Order Management Module**: This module tracks orders placed by users, allowing admins to update order status and manage shipments. It also includes functionalities for order cancellations and refunds.
- **Admin Dashboard Module**: This is the central hub for admin users to manage the entire store. It provides a comprehensive view of products, categories, users, and orders in a tabular format. Admins can also generate reports and monitor store performance from this dashboard.

CHAPTER 5: SYSTEM DESIGN

1.1 BLOCK DIAGRAM

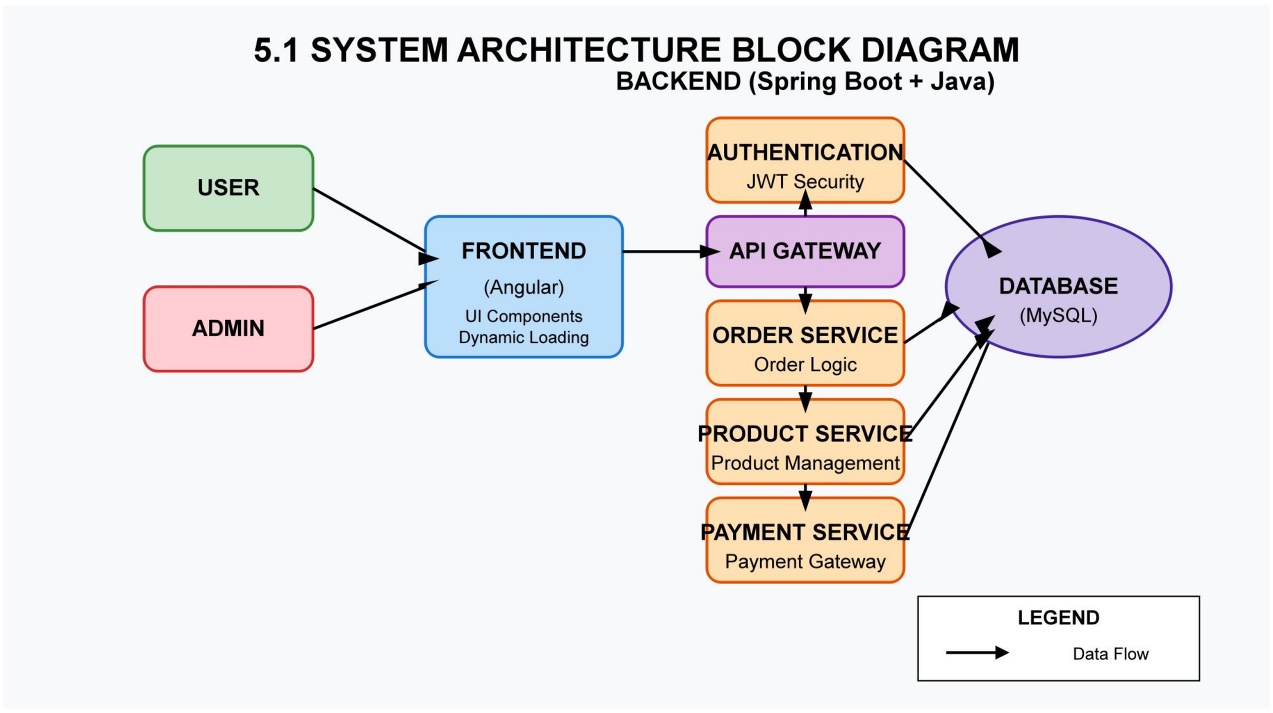


Fig 5.1 Block Diagram

This image shows a system architecture block diagram for a backend system built with Spring Boot and Java. Here's a description of the components:

The diagram is titled "5.1 SYSTEM ARCHITECTURE BLOCK DIAGRAM - BACKEND (Spring Boot + Java)" and illustrates the flow between different components:

1. **User & Admin**: At the left side, there are two user types represented by green and red boxes respectively.
2. **Frontend**: Both user types connect to a Frontend component (blue box) that uses Angular with UI Components and Dynamic Loading.
3. **API Gateway**: The Frontend connects to an API Gateway (purple box) which serves as the entry point to the backend services.
4. **Microservices**: The architecture includes three orange microservices:
    1. Authentication Service with JWT Security
    2. Order Service handling Order Logic
    3. Product Service for Product Management
5. **Database**: All these services connect to a MySQL Database (purple oval).

6. **Data Flow**: Black arrows throughout the diagram indicate the direction of data flow between components, as noted in the legend at the bottom right.

This represents a modern microservices-based architecture with clear separation of concerns, where the API Gateway routes requests from the frontend to the appropriate backend services, all of which interact with a central database.

1.2 FLOW CHART DIAGRAM

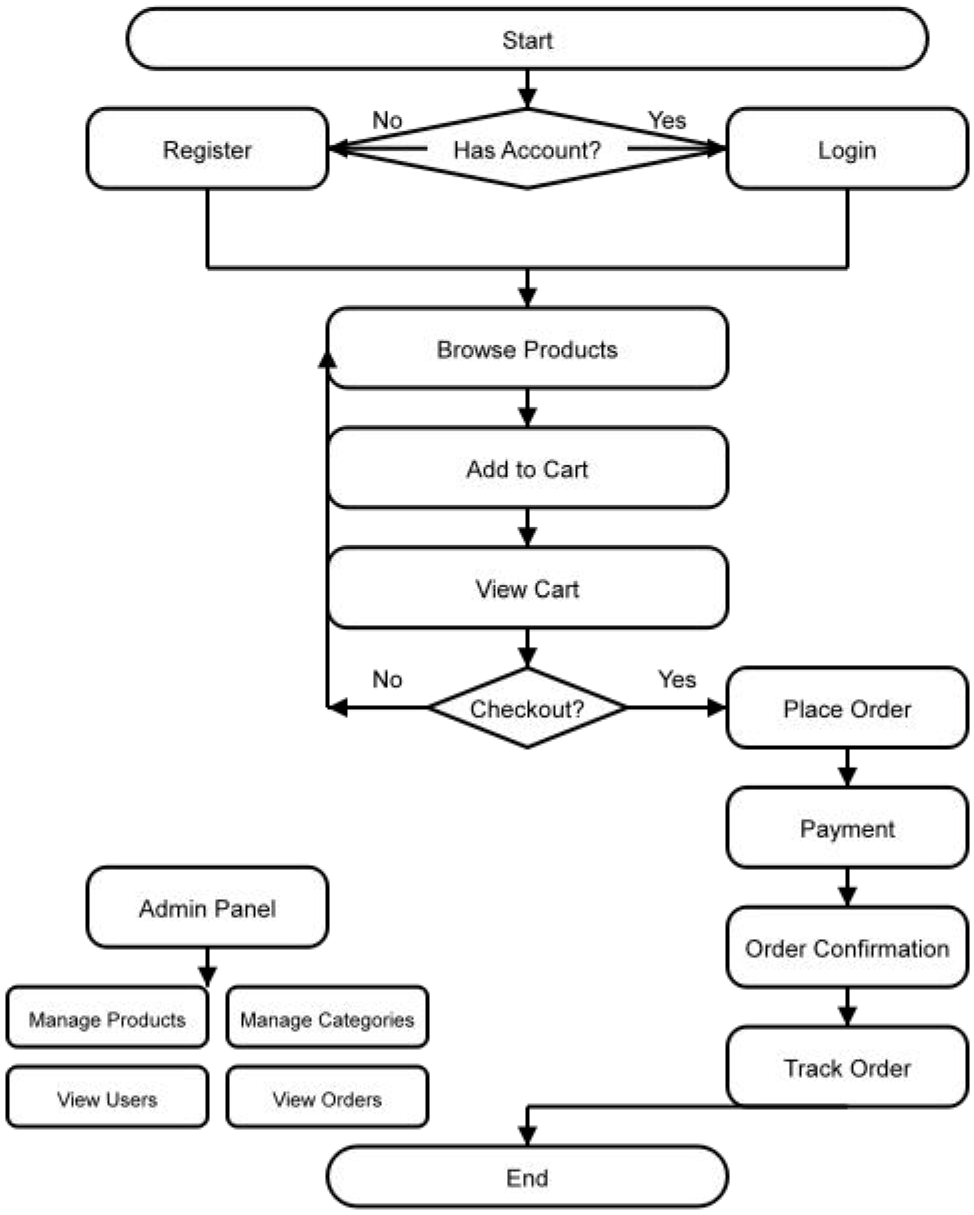


Fig 5.2 Flow Chart Diagram

1.3 ER DIAGRAM

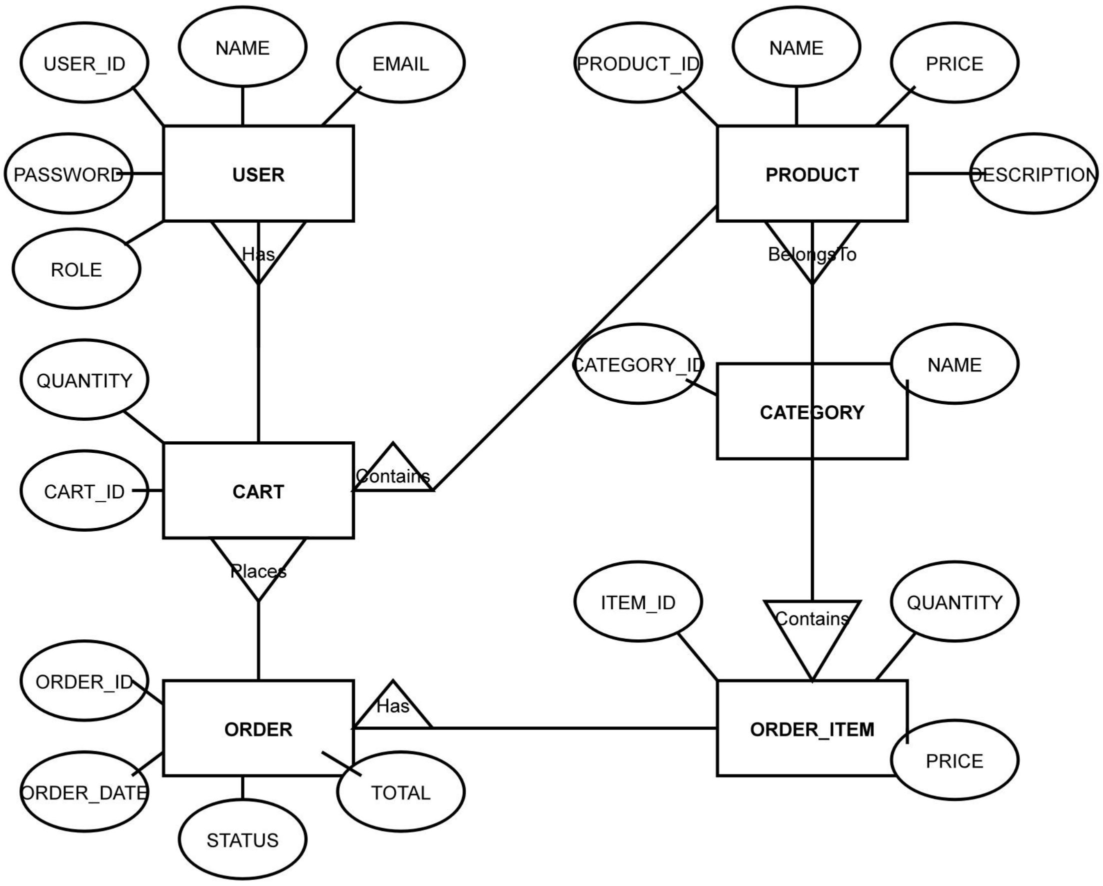


Fig 5.3 ER Diagram

1.4 USE CASE DIAGRAM

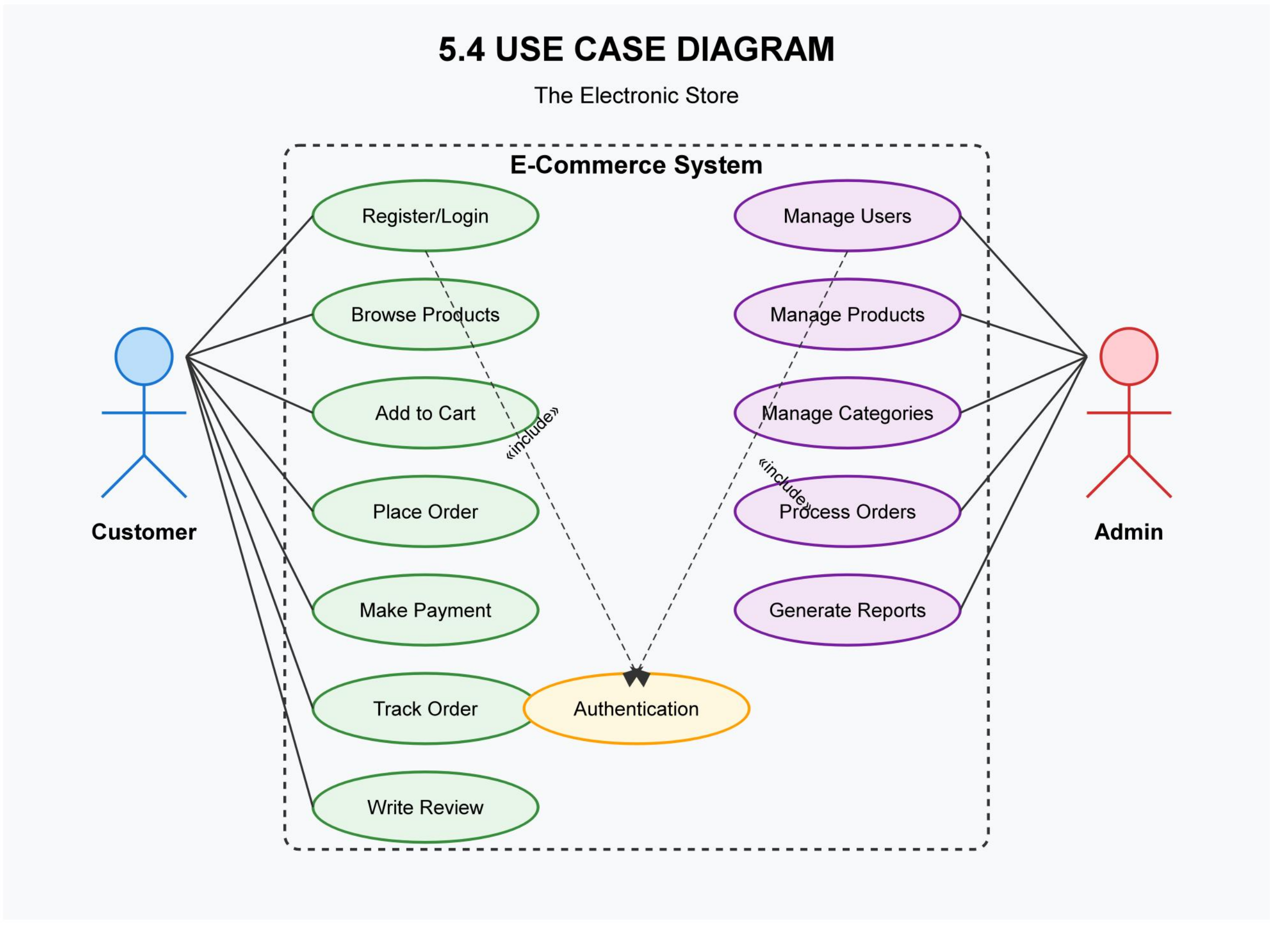


Fig 5.4 Use Case Diagram

1.5 SEQUENTIAL DIAGRAM

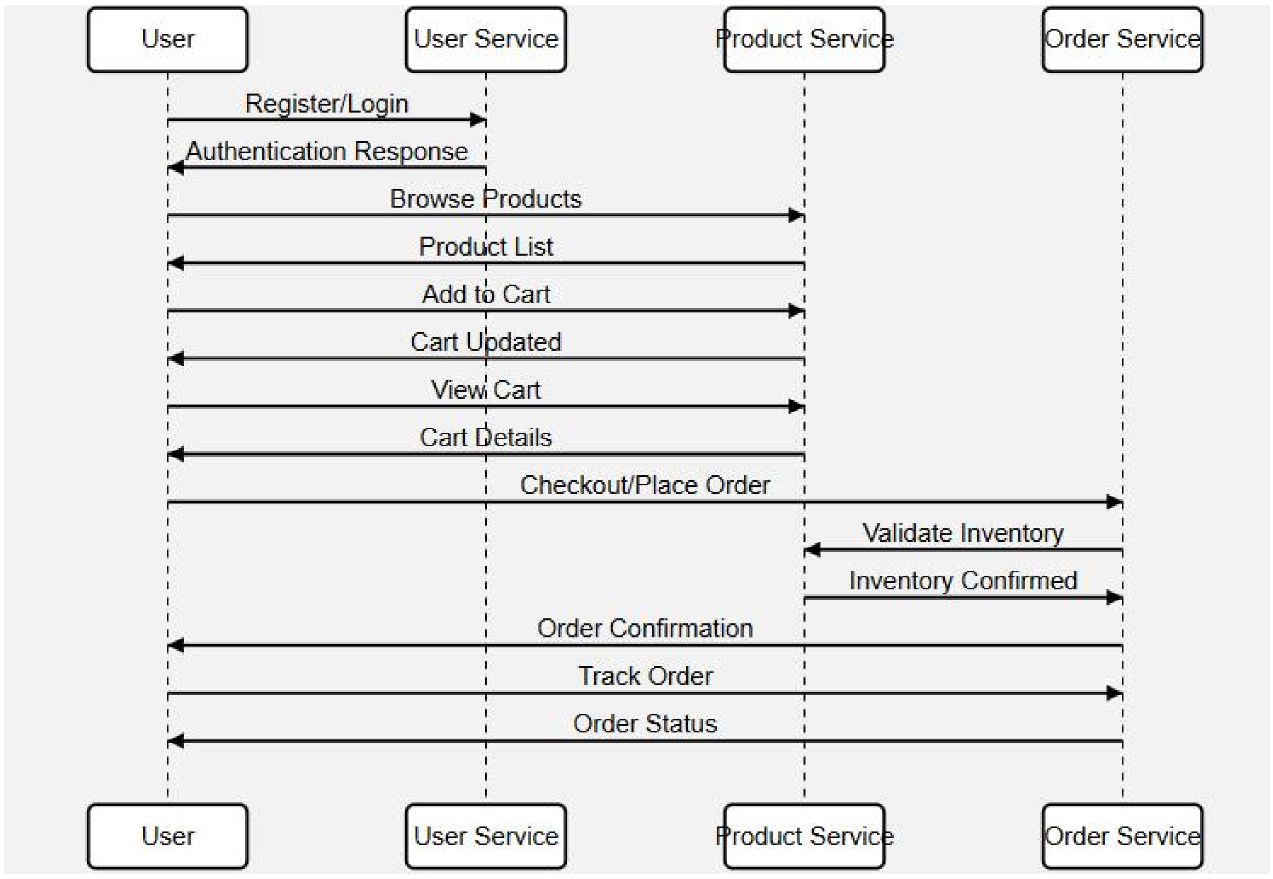


Fig 5.5 Sequential Diagram

CHAPTER 6: IMPLEMENTATION

6.1 IMPLEMENTATION PLATFORM

To develop a modern and scalable electronic commerce web application, a robust full-stack architecture has been used. Each component of the platform plays a vital role in delivering a seamless and responsive online shopping experience for users and administrators alike.

- **Frontend**: Developed using Angular, a powerful TypeScript-based framework that supports reactive forms, component-based architecture, and seamless routing. Angular provides a rich and responsive user interface for customers and admins.
- **Backend**: The backend is built using Spring Boot (Java). It offers a secure, RESTful API-based structure that handles business logic, user authentication, cart and order management, and role-based access control.
- **Database**: MySQL is used for relational data storage, efficiently managing tables such as users, products, categories, orders, and cart items.
- **Authentication**: Security is implemented using JWT (JSON Web Tokens) for maintaining stateless sessions and secure access to APIs. Admin and normal users are differentiated by roles.
- **Development Tools**:

  1. **Spring Tool Suite / IntelliJ** for backend development
  2. **Angular CLI & Visual Studio Code** for frontend development
  3. **MySQL Workbench** for database modeling and queries

6.2 MODULE SPECIFICATION

1. Authentication Module

This module allows users to register, log in, and maintain secure sessions using JWT-based authentication. Users are assigned roles—either **User** or **Admin**—at the time of registration or by admin update. Input validation and password hashing enhance security.

2. Product Management Module

Admins have access to a dedicated dashboard where they can:

- Add, update, or delete products
- Assign products to specific categories
- View all products in a tabular format with pagination and sorting

This module ensures that all product-related operations are handled dynamically via Spring Boot services and Angular UI.

3. Category Management Module

Admins can create and manage product categories to improve the browsing experience for users. Categories are displayed as filters on the product catalog page, making it easier for users to find specific electronic items.

4. User Module

This module manages all registered users. Admin can view the list of users, monitor activity, and manage roles if needed. The system ensures proper access control through role-based permission handling.

5. Cart Module

Users can add products to their cart, view and modify cart contents, and proceed to checkout. Cart data is managed in the backend with real-time synchronization between frontend and database.

6. Order Management Module

Users can place orders after finalizing their cart. Each order is tracked in the system with a unique order ID and status such as "Pending", "Shipped", or "Delivered". Admins can view all orders and update their statuses as needed.

7. Order Tracking Module

Users can track the status of their orders using their dashboard. This ensures transparency and enhances the post-purchase experience. Order statuses are updated by the admin through the backend.

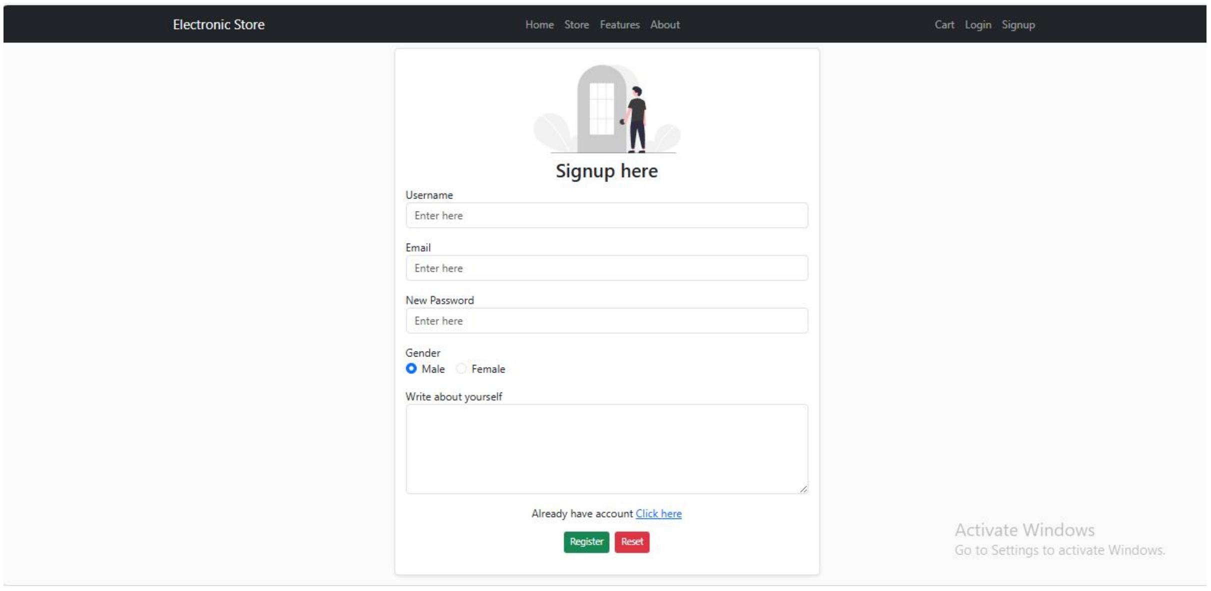
6.3 OUTCOMES

The successful implementation of this electronic store project demonstrates the effectiveness of full-stack development for e-commerce platforms. The key outcomes include:

- **Complete User Authentication** with secure role-based access control using JWT.
- **Functional Admin Dashboard** allowing seamless product and category management.
- **Responsive Product Catalog** with real-time filtering based on categories.
- **Efficient Cart System** supporting add/remove functionality, quantity updates, and price calculations.
- **Order Placement & Tracking** system allowing users to place and monitor orders effortlessly.
- **User-Friendly Interface** on both desktop and mobile through Angular’s responsive design.
- **Real-Time Communication** between frontend and backend through RESTful APIs.

This project successfully simulates an end-to-end e-commerce flow—from registration to checkout—enabling practical understanding of web app architecture, security, and database integration.

6.4 SCREENSHOTS



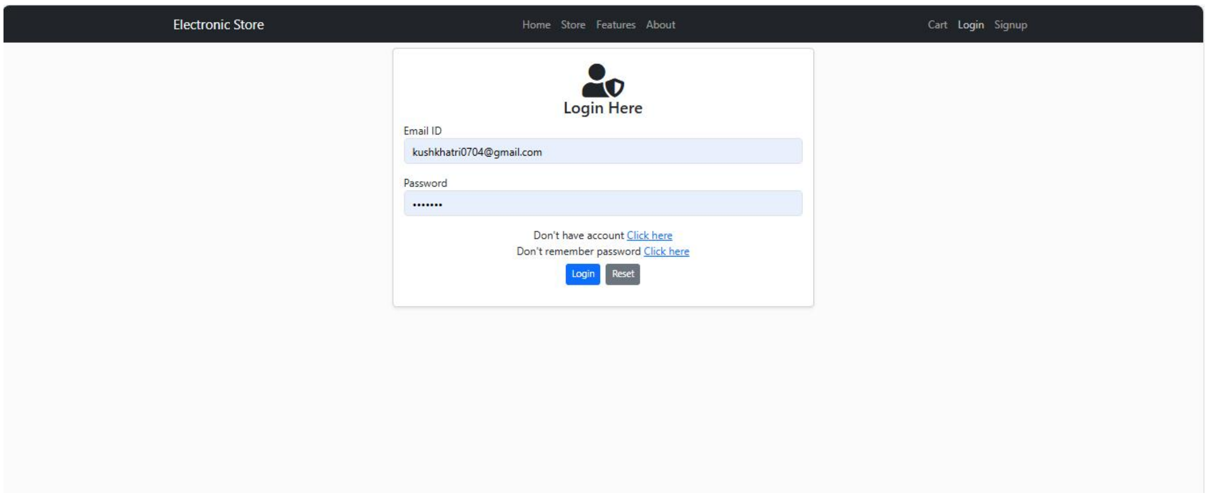
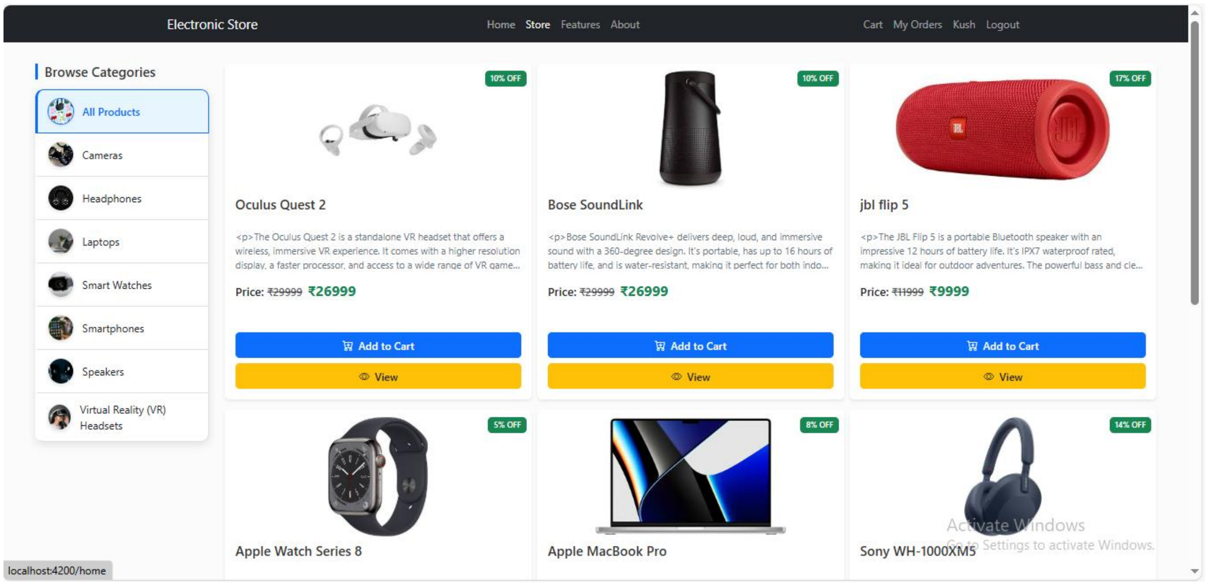


Fig 6.4.1 – User Registration and Login Page



Oculus Quest 2

<p>The Oculus Quest 2 is a standalone VR headset that offers a wireless, immersive VR experience. It comes with a higher resolution display, a faster processor, and access to a wide range of VR game...

Price: ~~₹29999~~ ₹26999

Bose SoundLink

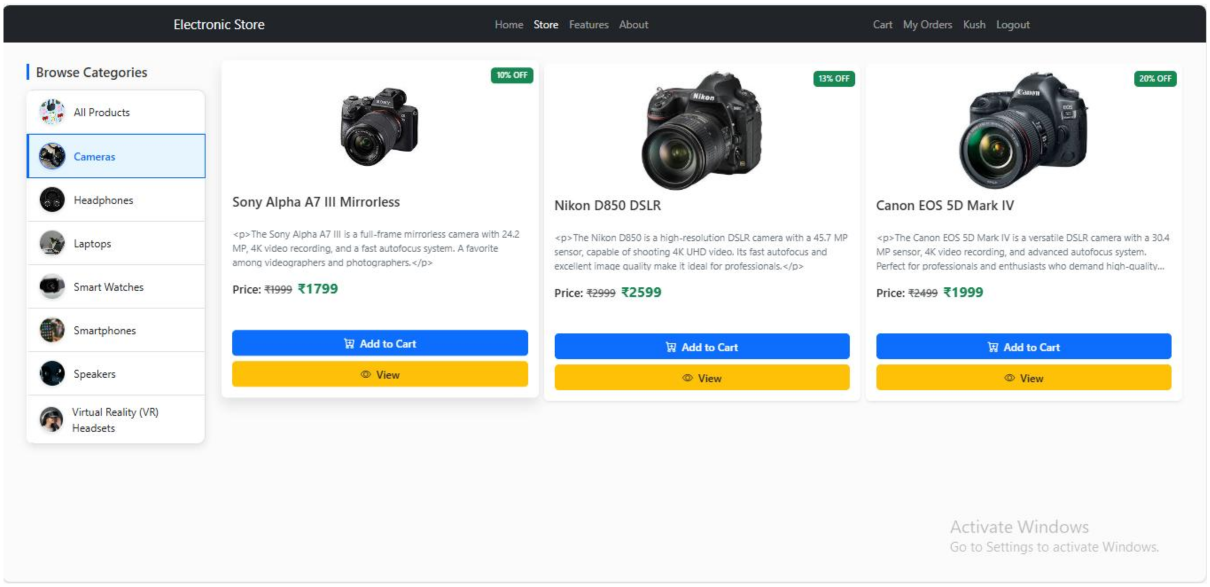
<p>Bose SoundLink Revolve+ delivers deep, loud, and immersive sound with a 360-degree design. It's portable, has up to 16 hours of battery life, and is water-resistant, making it perfect for both indo...

Price: ~~₹29999~~ ₹26999

jbl flip 5

<p>The JBL Flip 5 is a portable Bluetooth speaker with an impressive 12 hours of battery life. It's IPX7 waterproof rated, making it ideal for outdoor adventures. The powerful bass and cle...

Price: ~~₹11999~~ ₹9999



Sony Alpha A7 III Mirrorless

<p>The Sony Alpha A7 III is a full-frame mirrorless camera with 24.2 MP, 4K video recording, and a fast autofocus system. A favorite among videographers and photographers.</p>

Price: ~~₹1999~~ ₹1799

Nikon D850 DSLR

<p>The Nikon D850 is a high-resolution DSLR camera with a 45.7 MP sensor, capable of shooting 4K UHD video. Its fast autofocus and excellent image quality make it ideal for professionals.</p>

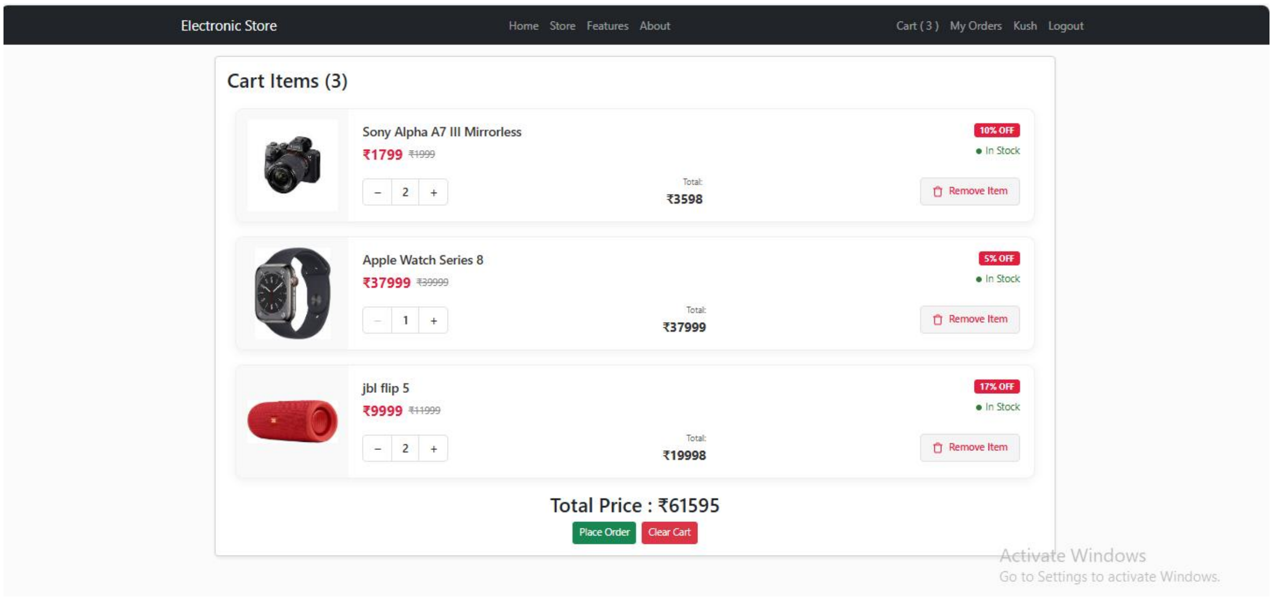
Price: ~~₹2999~~ ₹2599

Canon EOS 5D Mark IV

<p>The Canon EOS 5D Mark IV is a versatile DSLR camera with a 30.4 MP sensor, 4K video recording, and advanced autofocus system. Perfect for professionals and enthusiasts who demand high-quality...

Price: ~~₹2499~~ ₹1999

Fig 6.4.2 – Product Catalog with Category Filtering



Electronic Store — Home Store Features About — Cart (3) My Orders Kush Logout

Cart Items (3)

Sony Alpha A7 III Mirrorless
₹1799 ~~₹1999~~
10% OFF ● In Stock
– 2 + Total: ₹3598 Remove Item

Apple Watch Series 8
₹37999 ~~₹39999~~
5% OFF ● In Stock
– 1 + Total: ₹37999 Remove Item

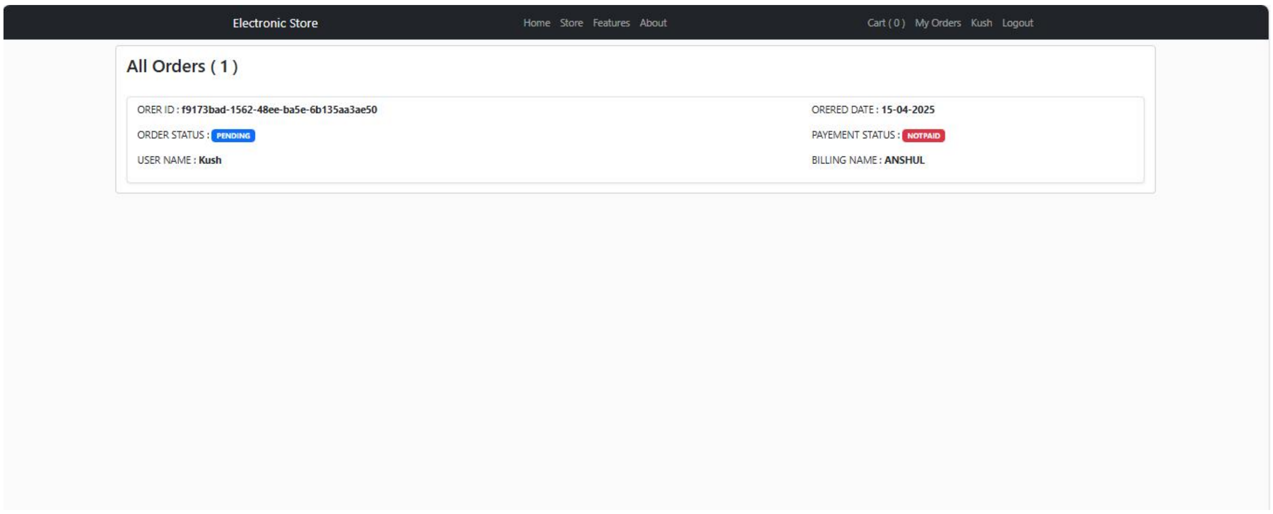
jbl flip 5
₹9999 ~~₹11999~~
17% OFF ● In Stock
– 2 + Total: ₹19998 Remove Item

Total Price : ₹61595

Place Order Clear Cart

Activate Windows
Go to Settings to activate Windows.

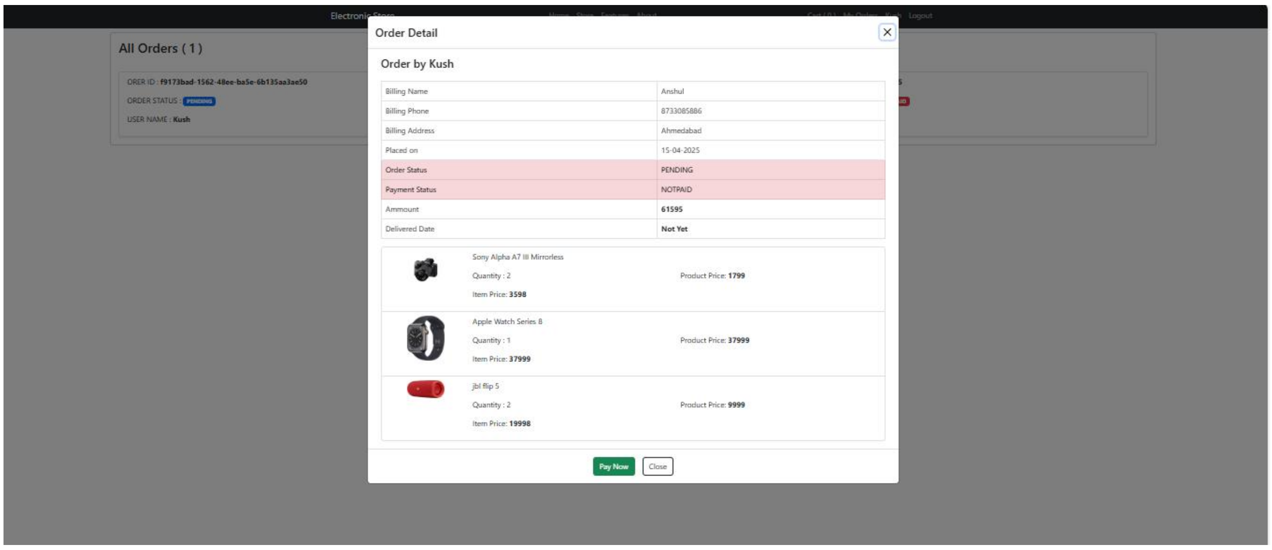
Fig 6.4.3 – Cart View with Product Quantity Controls



Electronic Store — Home Store Features About — Cart (0) My Orders Kush Logout

All Orders (1)

ORER ID : **f9173bad-1562-48ee-ba5e-6b135aa3ae50** ORERED DATE : **15-04-2025**
ORDER STATUS : PENDING PAYEMENT STATUS : NOTPAID
USER NAME : **Kush** BILLING NAME : **ANSHUL**



Order Detail

Order by Kush

Billing Name	Anshul
Billing Phone	8733085886
Billing Address	Ahmedabad
Placed on	15-04-2025
Order Status	PENDING
Payment Status	NOTPAID
Ammount	**61595**
Delivered Date	**Not Yet**

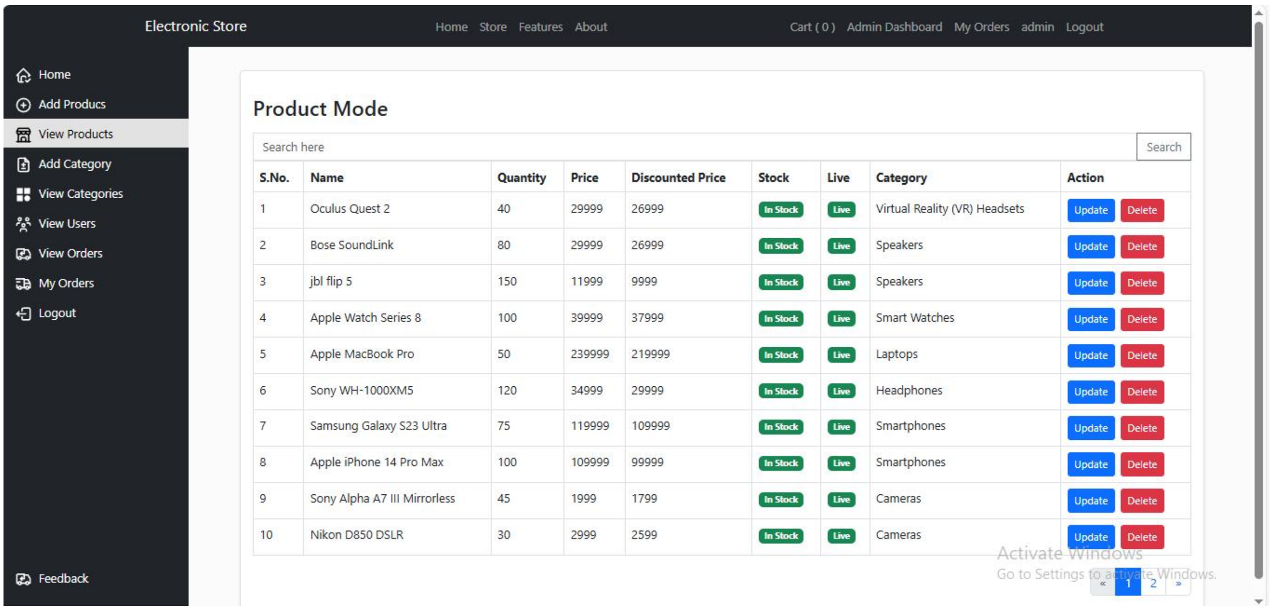
Sony Alpha A7 III Mirrorless
Quantity : 2 Product Price: **1799**
Item Price: **3598**

Apple Watch Series 8
Quantity : 1 Product Price: **37999**
Item Price: **37999**

jbl flip 5
Quantity : 2 Product Price: **9999**
Item Price: **19998**

Pay Now Close

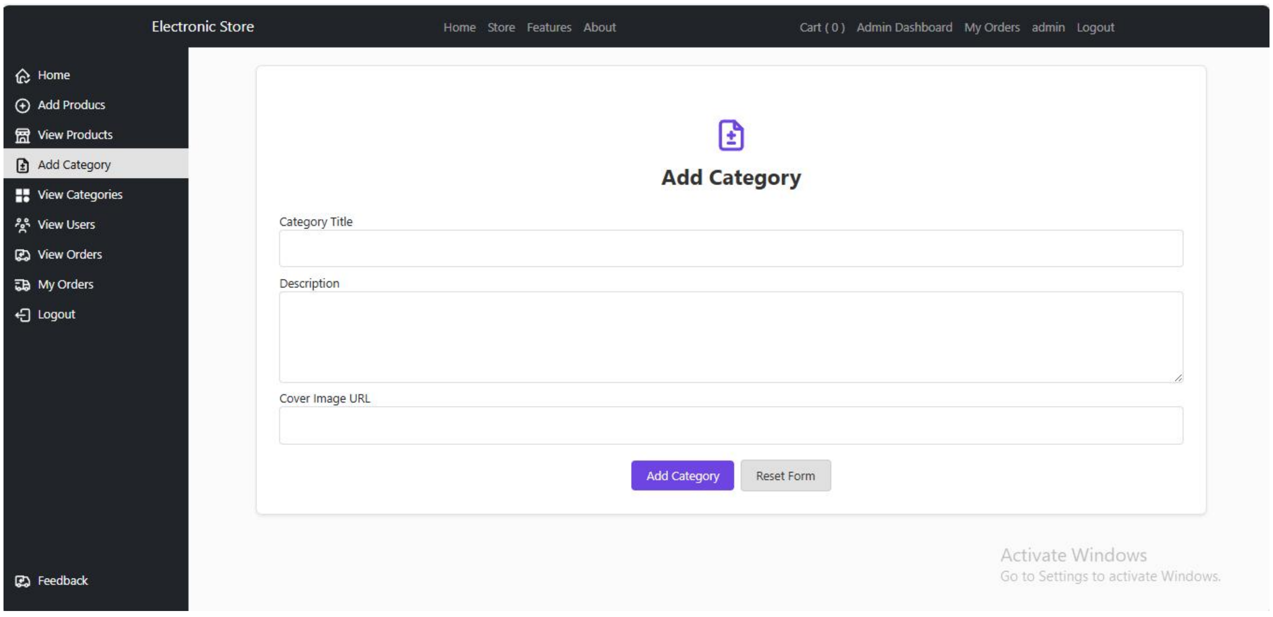
Fig 6.4.4 – Order Tracking Dashboard for Users



Product Mode

S.No.	Name	Quantity	Price	Discounted Price	Stock	Live	Category	Action
1	Oculus Quest 2	40	29999	26999	In Stock	Live	Virtual Reality (VR) Headsets	Update Delete
2	Bose SoundLink	80	29999	26999	In Stock	Live	Speakers	Update Delete
3	jbl flip 5	150	11999	9999	In Stock	Live	Speakers	Update Delete
4	Apple Watch Series 8	100	39999	37999	In Stock	Live	Smart Watches	Update Delete
5	Apple MacBook Pro	50	239999	219999	In Stock	Live	Laptops	Update Delete
6	Sony WH-1000XM5	120	34999	29999	In Stock	Live	Headphones	Update Delete
7	Samsung Galaxy S23 Ultra	75	119999	109999	In Stock	Live	Smartphones	Update Delete
8	Apple iPhone 14 Pro Max	100	109999	99999	In Stock	Live	Smartphones	Update Delete
9	Sony Alpha A7 III Mirrorless	45	1999	1799	In Stock	Live	Cameras	Update Delete
10	Nikon D850 DSLR	30	2999	2599	In Stock	Live	Cameras	Update Delete

Fig 6.4.5 – Admin Panel: Product Management Table



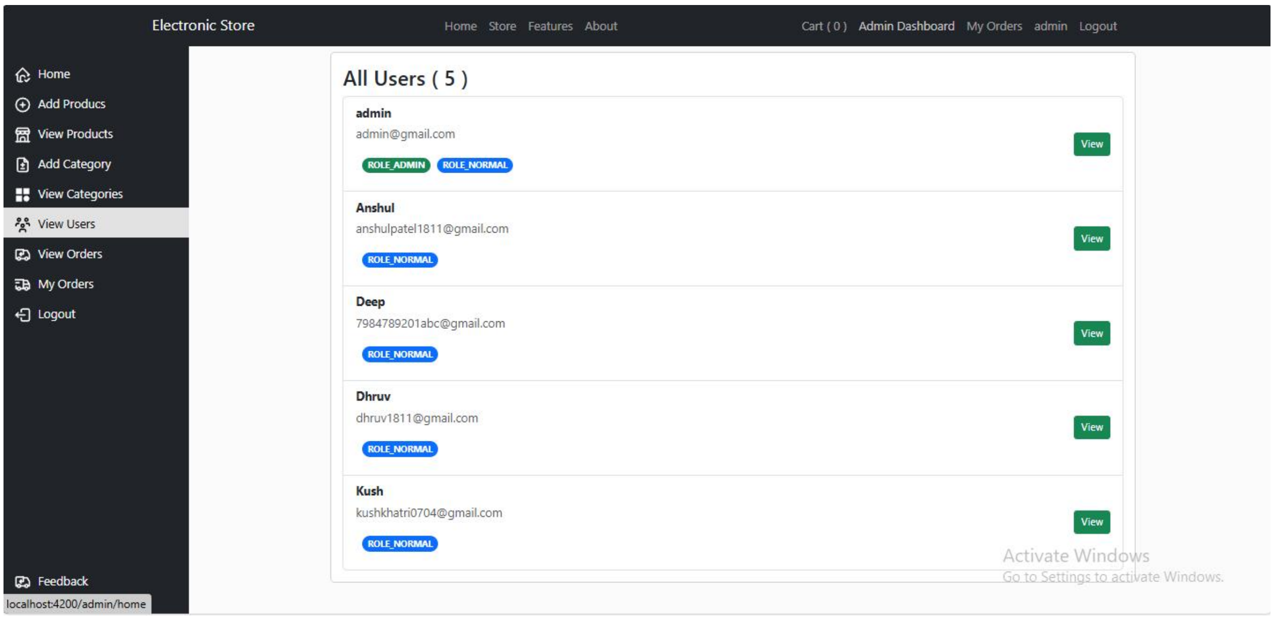
Add Category

Category Title

Description

Cover Image URL

Fig 6.4.5 – Admin Panel: Product Management Table



All Users (5)

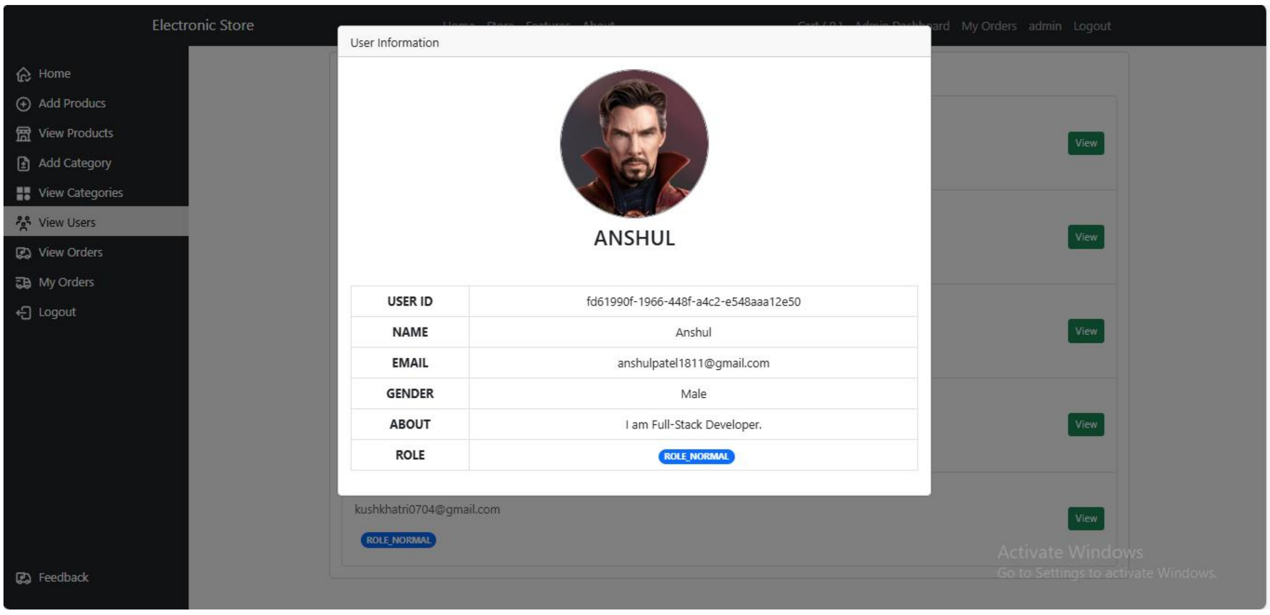
admin
admin@gmail.com
ROLE_ADMIN ROLE_NORMAL

Anshul
anshulpatel1811@gmail.com
ROLE_NORMAL

Deep
7984789201abc@gmail.com
ROLE_NORMAL

Dhruv
dhruv1811@gmail.com
ROLE_NORMAL

Kush
kushkhatri0704@gmail.com
ROLE_NORMAL

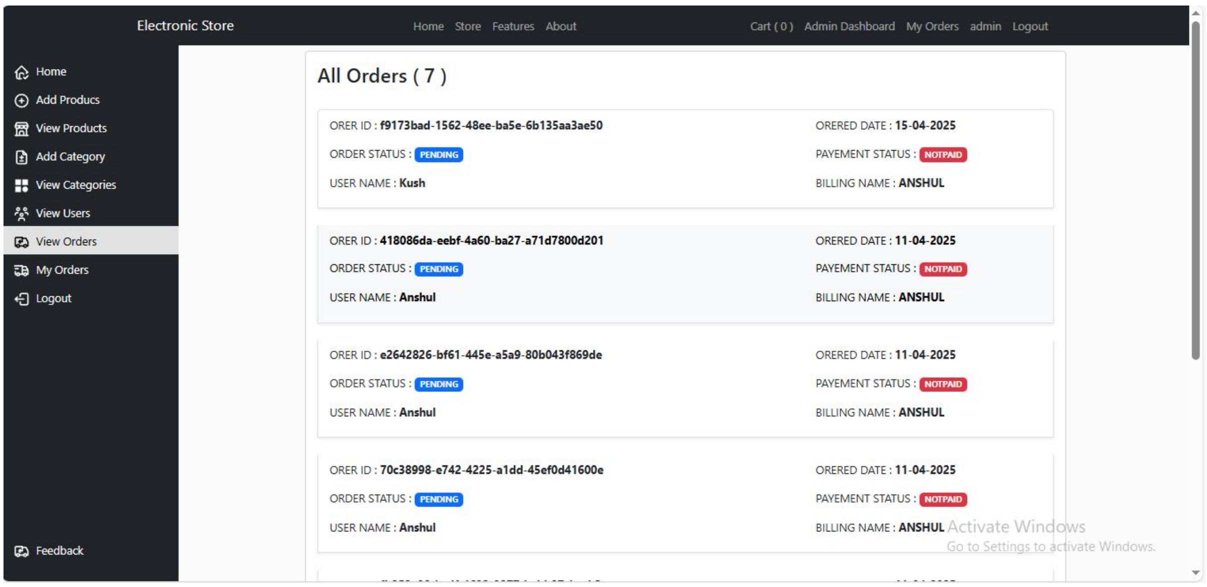


User Information

ANSHUL

USER ID	fd61990f-1966-448f-a4c2-e548aaa12e50
NAME	Anshul
EMAIL	anshulpatel1811@gmail.com
GENDER	Male
ABOUT	I am Full-Stack Developer.
ROLE	ROLE_NORMAL

Fig 6.4.7 – User Table View in Admin Dashboard



All Orders (7)

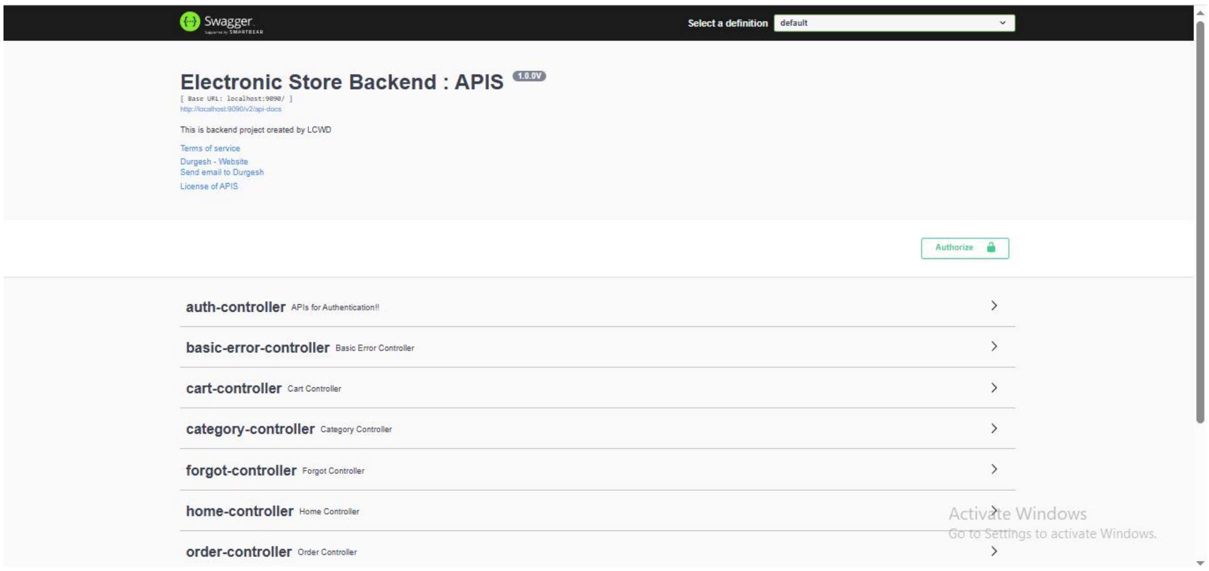
ORER ID : f9173bad-1562-48ee-ba5e-6b135aa3ae50
ORERED DATE : 15-04-2025
ORDER STATUS : PENDING
PAYEMENT STATUS : NOTPAID
USER NAME : Kush
BILLING NAME : ANSHUL

ORER ID : 418086da-eebf-4a60-ba27-a71d7800d201
ORERED DATE : 11-04-2025
ORDER STATUS : PENDING
PAYEMENT STATUS : NOTPAID
USER NAME : Anshul
BILLING NAME : ANSHUL

ORER ID : e2642826-bf61-445e-a5a9-80b043f869de
ORERED DATE : 11-04-2025
ORDER STATUS : PENDING
PAYEMENT STATUS : NOTPAID
USER NAME : Anshul
BILLING NAME : ANSHUL

ORER ID : 70c38998-e742-4225-a1dd-45ef0d41600e
ORERED DATE : 11-04-2025
ORDER STATUS : PENDING
PAYEMENT STATUS : NOTPAID
USER NAME : Anshul
BILLING NAME : ANSHUL

Fig 6.4.8 – Orders Management Table (Admin View)



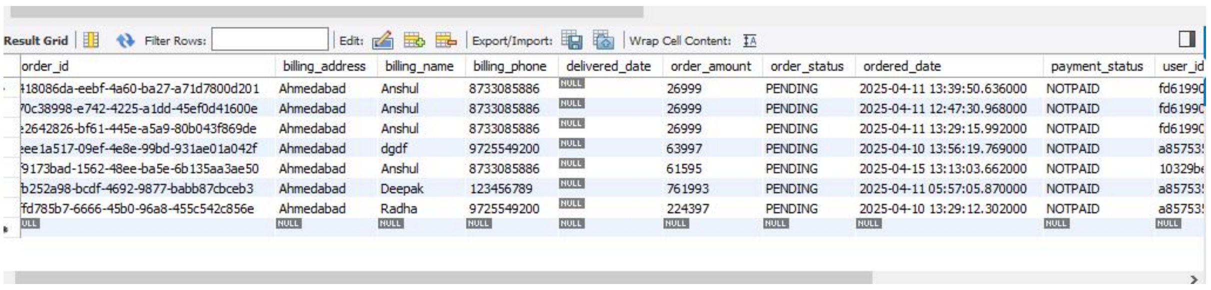
Electronic Store Backend : APIS 1.0.0V
[Base URL: localhost:9090/]
http://localhost:9090/v2/api-docs

This is backend project created by LCWD

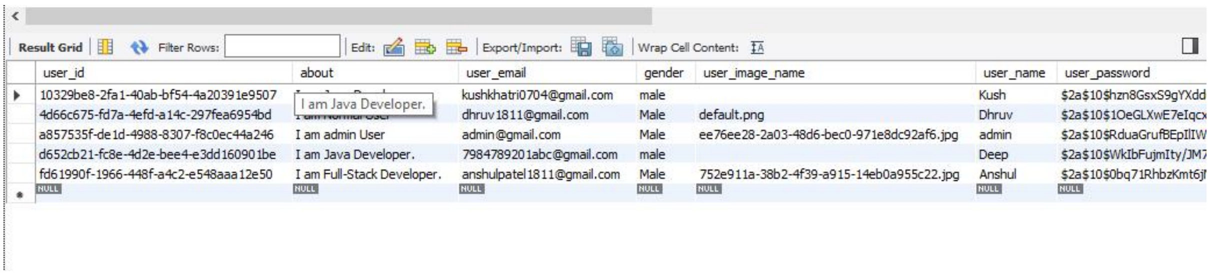
Terms of service
Durgesh - Website
Send email to Durgesh
License of APIS

auth-controller APIs for Authentication!!
basic-error-controller Basic Error Controller
cart-controller Cart Controller
category-controller Category Controller
forgot-controller Forgot Controller
home-controller Home Controller
order-controller Order Controller

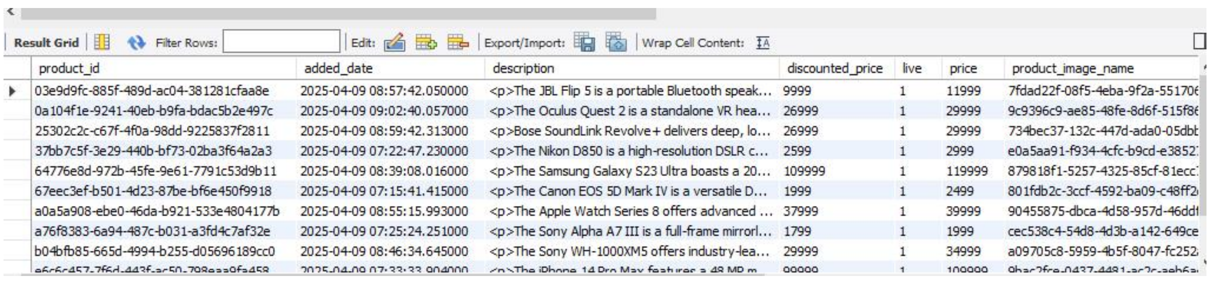
Fig 6.4.9 – API Testing via Swagger



order_id	billing_address	billing_name	billing_phone	delivered_date	order_amount	order_status	ordered_date	payment_status	user_id
418086da-eebf-4a60-ba27-a71d7800d201	Ahmedabad	Anshul	8733085886	NULL	26999	PENDING	2025-04-11 13:39:50.636000	NOTPAID	fd6199(
70c38998-e742-4225-a1dd-45ef0d41600e	Ahmedabad	Anshul	8733085886	NULL	26999	PENDING	2025-04-11 12:47:30.968000	NOTPAID	fd6199(
e2642826-bf61-445e-a5a9-80b043f869de	Ahmedabad	Anshul	8733085886	NULL	26999	PENDING	2025-04-11 13:29:15.992000	NOTPAID	fd6199(
eee1a517-09ef-4e8e-99bd-931ae01a042f	Ahmedabad	dgdf	9725549200	NULL	63997	PENDING	2025-04-10 13:56:19.769000	NOTPAID	a85753!
f9173bad-1562-48ee-ba5e-6b135aa3ae50	Ahmedabad	Anshul	8733085886	NULL	61595	PENDING	2025-04-15 13:13:03.662000	NOTPAID	10329be
fb252a98-bcdf-4692-9877-babb87cbceb3	Ahmedabad	Deepak	123456789	NULL	761993	PENDING	2025-04-11 05:57:05.870000	NOTPAID	a85753!
ffd785b7-6666-45b0-96a8-455c542c856e	Ahmedabad	Radha	9725549200	NULL	224397	PENDING	2025-04-10 13:29:12.302000	NOTPAID	a85753!
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL



user_id	about	user_email	gender	user_image_name	user_name	user_password
10329be8-2fa1-40ab-bf54-4a20391e9507	I am Java Developer.	kushkhatri0704@gmail.com	male		Kush	$2a$10$hzn8GsxS9gYXdd
4d66c675-fd7a-4efd-a14c-297fea6954bd	I am Normal User	dhruv1811@gmail.com	Male	default.png	Dhruv	$2a$10$1OeGLXwE7eIqcx
a857535f-de1d-4988-8307-f8c0ec44a246	I am admin User	admin@gmail.com	Male	ee76ee28-2a03-48d6-bec0-971e8dc92af6.jpg	admin	$2a$10$RduaGrufBEpIlIW
d652cb21-fc8e-4d2e-bee4-e3dd160901be	I am Java Developer.	7984789201abc@gmail.com	male		Deep	$2a$10$WkIbFujmIty/JM7
fd61990f-1966-448f-a4c2-e548aaa12e50	I am Full-Stack Developer.	anshulpatel1811@gmail.com	Male	752e911a-38b2-4f39-a915-14eb0a955c22.jpg	Anshul	$2a$10$0bq71RhbzKmt6j
NULL	NULL	NULL	NULL	NULL	NULL	NULL



product_id	added_date	description	discounted_price	live	price	product_image_name
03e9d9fc-885f-489d-ac04-381281cfaa8e	2025-04-09 08:57:42.050000	<p>The JBL Flip 5 is a portable Bluetooth speak...	9999	1	11999	7fdad22f-08f5-4eba-9f2a-55170€
0a104f1e-9241-40eb-b9fa-bdac5b2e497c	2025-04-09 09:02:40.057000	<p>The Oculus Quest 2 is a standalone VR hea...	26999	1	29999	9c9396c9-ae85-48fe-8d6f-515f8€
25302c2c-c67f-4f0a-98dd-9225837f2811	2025-04-09 08:59:42.313000	<p>Bose SoundLink Revolve+ delivers deep, lo...	26999	1	29999	734bec37-132c-447d-ada0-05dbb
37bb7c5f-3e29-440b-bf73-02ba3f64a2a3	2025-04-09 07:22:47.230000	<p>The Nikon D850 is a high-resolution DSLR c...	2599	1	2999	e0a5aa91-f934-4cfc-b9cd-e3852
64776e8d-972b-45fe-9e61-7791c53d9b11	2025-04-09 08:39:08.016000	<p>The Samsung Galaxy S23 Ultra boasts a 20...	109999	1	119999	879818f1-5257-4325-85cf-81ecc
67eec3ef-b501-4d23-87be-bf6e450f9918	2025-04-09 07:15:41.415000	<p>The Canon EOS 5D Mark IV is a versatile D...	1999	1	2499	801fdb2c-3ccf-4592-ba09-c48ff2
a0a5a908-ebe0-46da-b921-533e4804177b	2025-04-09 08:55:15.993000	<p>The Apple Watch Series 8 offers advanced ...	37999	1	39999	90455875-dbca-4d58-957d-46dd
a76f8383-6a94-487c-b031-a3fd4c7af32e	2025-04-09 07:25:24.251000	<p>The Sony Alpha A7 III is a full-frame mirrorl...	1799	1	1999	cec538c4-54d8-4d3b-a142-649ce
b04bfb85-665d-4994-b255-d05696189cc0	2025-04-09 08:46:34.645000	<p>The Sony WH-1000XM5 offers industry-lea...	29999	1	34999	a09705c8-5959-4b5f-8047-fc252
e6c6c457-7f6d-443f-ac50-798eaa9fa458	2025-04-09 07:33:33.904000	<p>The iPhone 14 Pro Max features a 48 MP m	99999	1	109999	9bac2fce-0437-4481-ac2c-aeb6a

Fig 6.4.10 – MySQL Database Tables for Orders, Users, Products

CHAPTER 7: TESTING

This chapter describes the testing methodology adopted for the Electronic Store project and presents the results derived during various stages of testing. The objective of testing is to ensure that the application is reliable, secure, and functions as intended across different user roles and scenarios.

7.1 TESTING TECHNIQUES:

◆ **Black Box Testing**:

Black box testing was used to validate the core functionalities of the system without analyzing the internal code. It focused on testing user flows such as:

- User registration and login/logout
- Product browsing and category filtering
- Add to cart functionality
- Order placement and order tracking
- Admin operations like adding/editing products and categories

All inputs were tested for valid and invalid data to verify correct outputs and error handling.

◆ **White Box Testing**:

White box testing involved examining internal code structures, logic, and database interactions. Key focus areas included:

- Validation logic for user input and authentication
- Product CRUD operations and data consistency
- Efficient database queries for large product sets
- Role-based access control and route protection

Spring Boot service classes and repository layers were tested to ensure accurate data manipulation.

7.2 TEST AUTOMATION:

- **Automated Testing Tools**: Automating repetitive tasks such as user login, product search, cart operations, and checkout to ensure consistent performance and faster regression cycles.
- **Automation Frameworks**: Tools like Selenium for frontend UI testing, Postman for API testing (authentication, cart, orders), and JUnit or Mockito for backend unit testing in the Spring Boot environment.

7.3 TEST ENVIRONMENT:

- **Environment Setup**:
  - Replicating a staging environment that mirrors the production setup (Spring Boot backend, Angular frontend, MySQL database).
  - Ensuring payment gateways, email servers (for password recovery), and third-party authentication modules are configured properly.
- **Dependencies Configuration**:
  - Setting up external APIs (like payment providers or OTP services).
  - Initializing MySQL with seed data for categories, products, and user roles.
  - Using Docker or environment files for consistent backend/frontend environments.

7.4 DEFECT MANAGEMENT:

- **Defect logging**:
  - Capturing issues like broken login, failed product add-to-cart, or payment failures.
  - Using tools such as JIRA, Trello, or Bugzilla for defect tracking and management.
- **Defect Resolution Process**:
  - Assigning bugs to relevant devs with steps to reproduce, actual vs expected behavior, and screenshots/logs.
  - Prioritizing bugs based on user impact (e.g., checkout failures > UI misalignment).

7.5 PERFORMANCE TESTING:

- **Load Testing**: Simulating hundreds of users logging in, browsing products, and placing orders simultaneously using tools like JMeter or Locust.

- **Stress Testing**: Testing how the system handles extreme load—like mass product imports or 1000+ concurrent checkouts.

- **Scalability Testing**: Ensuring the system scales effectively when expanding product catalogs, user base, and orders volume.

7.6 SECURITY TESTING:

- **Penetration Testing:** Simulating attacks like SQL injection, cross-site scripting (XSS), and session hijacking on login, cart, and admin modules.Verifying access control between normal users and admin roles.

- **Vulnerability Scanning:** Scanning APIs and frontend for security loopholes using tools like OWASP ZAP or SonarQube.Checking secure password storage, HTTPS usage, and CSRF protection.

7.7 USER ACCEPTANCE TESTING(UAT):

- **End-User Testing:** Conducting UAT with a group of users representing both roles (normal users and admins).Testing major features: product browsing, cart, orders, registration/login, and admin product management.
- **Feedback Integration:** Collecting feedback to ensure UX/UI is intuitive and features like password reset, product filtering, and payment are working as expected.Iteratively refining based on real user interaction before deployment.

CHAPTER 8
CONCLUSIONS AND DISCUSSIONS

8.1 OVERALL ANALYSIS OF INTERNSHIP

- The development and implementation of the **Electronic Store** web application proved to be a highly effective and practical solution in the domain of online retail and e-commerce. From technical, economic, and operational perspectives, the project demonstrates strong viability, aiming to deliver a seamless digital shopping experience for end-users while offering powerful administrative tools for store management.

- From a **technical standpoint**, the system leverages modern and scalable technologies such as **Spring Boot** for secure backend development, **Angular** for a dynamic frontend, and **MySQL** for reliable data management. This full-stack architecture promotes clean code separation, modularity, and performance optimization. The application supports key features such as user registration, login, product browsing, cart management, checkout, and payment integration—effectively simulating a real-world e-commerce platform.

- The **operational feasibility** is also high, with the system supporting real-life business logic such as role-based access (admin vs. user), secure payment handling, order tracking, and product/category management. These functionalities mirror the operations of modern e-commerce businesses, making the system highly usable and scalable for commercial deployment.

- Moreover, the modular structure ensures **scalability and maintainability**, enabling easy integration of additional features such as discount modules, customer reviews, and mobile app extensions in the future.

8.2 PHOTOGRAPH AND DATE OF SURPRISE VISIT BY INSTITUE MENTOR

Visit by institute mentor to the company on 24th March 2025

8.3 PROBLEMS ENCOUNTERED AND POSSIBLE SOLUTIONS

1. Complex Role Management

Problem:

Handling multiple user roles (admin and customer) securely while managing access to sensitive admin functionalities.

Solution:

Implemented JWT-based authentication with Spring Security, and established role-based access control to ensure that users only see the modules relevant to their role.

2. Cart and Checkout Synchronization

Problem:

Keeping the cart data consistent across sessions and devices while maintaining real-time updates.

Solution:

Developed cart logic using Angular services and session/local storage on the frontend, and ensured synchronization with the backend database using unique user IDs.

3. Payment Gateway Integration

Problem:

Integrating a secure and reliable payment system that works seamlessly with order processing.

Solution:

Used third-party payment APIs (like Razorpay/Stripe) with secure token handling and backend validation before confirming orders.

4. Product & Category Management

Problem:

Admin needed a user-friendly way to manage products, categories, images, and pricing.

Solution:

Created dynamic admin panels with form validations, image upload support, and proper data-binding using Angular's reactive forms and REST APIs.

5. Data Validation and Error Handling

Problem:

Ensuring that all user inputs and actions (e.g., registration, payments) are secure and error-free.

Solution:

Applied both frontend (Angular) and backend (Spring Boot) validations, with meaningful error messages and fallback mechanisms.

6. Frontend-Backend CORS Issues

Problem:

CORS errors occurred during the development phase due to the separation of frontend (Angular) and backend (Spring Boot) servers.

Solution:

Configured appropriate CORS settings in the Spring Boot application to allow cross-origin requests during development and production.

7. Search and Filtering Functionality

Problem:

Providing responsive and accurate product search and filtering based on categories, prices, etc.

Solution:

Implemented query-based filtering endpoints in the backend and dynamic filtering logic in Angular using observables and pagination.

8.4 SUMMARY OF INTERNSHIP

During my internship, I was involved in the end-to-end development of a **full-stack e-commerce application** named *Electronic Store*, designed to enable users to browse, purchase, and manage electronic products online. The application utilized **Spring Boot (Java)** for backend APIs, **MySQL** as the relational database, and **Angular** for an interactive and responsive frontend.

My contributions included:

- Designing the **database schema** for users, products, orders, and categories.
- Implementing **user authentication and role-based authorization** using JWT tokens.
- Building **RESTful APIs** for product management, cart operations, and order processing.
- Developing **admin dashboards** for managing products, categories, users, and viewing orders.
- Creating **UI components in Angular** for registration, login, product listing, and checkout.
- Integrating **payment gateways** and building secure transaction flows.
- Handling **file uploads** for product images and ensuring proper validation.
- Deploying the application and resolving deployment-level issues like CORS, environment configurations, and database connectivity.

This internship significantly enhanced my understanding of **full-stack development**, **real-world problem-solving**, and **secure application design**. It also provided hands-on experience in managing a full product development lifecycle—right from planning and architecture to testing and deployment.

8.5 LIMITATIONS AND FUTURE ENHANCEMENT

LIMITATIONS:

1. Lack of Real-Time Notifications: Currently, there are no real-time alerts for order confirmation, shipping status, or payment failures. Users must manually check the order status, which may reduce engagement or lead to confusion during high-demand periods.

2. Limited Payment Integration Options: The system is integrated with a basic payment gateway, which limits payment methods. Users may not be able to use alternative payment modes such as UPI, digital wallets, or Buy Now Pay Later (BNPL).

3. Basic Search and Filter Functionality: Product search and filtering are available but could be more intelligent (e.g., typo tolerance, personalized results, trending items). The lack of advanced filtering impacts user navigation and conversion rates.

4. No Mobile App: The absence of a dedicated mobile application may limit accessibility for users who prefer shopping through mobile platforms. While the web app is responsive, native mobile experiences are generally smoother.

5. No AI or Recommendation Engine: The current platform does not suggest products based on user behavior or purchase history, which limits upselling and personalization.

FUTURE ENHANCEMENT:

1. **AI-Based Product Recommendation System:**
   - Use machine learning to analyze user behavior and recommend products tailored to individual preferences.
   - This would enhance user engagement and drive more sales.

2. **Multi-Payment Gateway Integration:**
   - Add support for additional payment options like **PayPal, UPI, Google Pay, Razorpay**, etc.
   - Ensure fallback gateways are available in case one fails.

3. **Progressive Web App (PWA) or Mobile App:**
   - Develop a **mobile-first** Progressive Web App for better accessibility.
   - Optionally, create Android/iOS apps for a native shopping experience.

4. **Admin Analytics Dashboard:**
   - Implement an advanced analytics dashboard for the admin to view sales trends, top products, customer insights, and inventory performance using chart libraries like Chart.js or D3.js.

5. **Real-Time Notification System:**
   - Integrate with services like **Firebase Cloud Messaging** or **WebSocket** to provide real-time updates on orders, stock changes, and promotions.

6. **Customer Review and Rating Module:**
   - Allow verified users to rate and review products.
   - Helps improve transparency, product quality feedback, and user trust.

CONCLUSION

The Electronic Store project represents a fully functional and modular e-commerce platform built using Spring Boot, Angular, and MySQL. It provides essential features such as user authentication, product browsing, cart management, order placement, and admin functionalities like product and category management.

Through this project, a solid foundation for online shopping experiences was developed with a secure and scalable architecture. Role-based access, payment integration, and dynamic product handling make it a strong base for real-world e-commerce deployment.

The system is also designed with future scalability in mind. With potential enhancements like AI-driven recommendations, real-time notifications, and multi-payment integrations, this platform is well-positioned for real-world business expansion.

Overall, the project not only strengthened technical skills in full-stack development but also provided practical experience in building enterprise-grade applications aligned with modern digital commerce needs.

REFERENCES

- **Spring Boot Documentation**: https://spring.io/projects/spring-boot
- **Angular Official Guide**: https://angular.io/start
- **MySQL Tutorial (W3Schools)**: https://www.w3schools.com/mysql/default.asp
- **JWT Authentication**: https://jwt.io/introduction
- **REST API Design (Spring Boot)**: https://www.baeldung.com/rest-with-spring-series
- **Angular Reactive Forms**: https://angular.io/guide/reactive-forms