**A Project Report On­­­­­**

**Juta.com**

**Academic Year 2024-25**

**MCA SEM 4**

**Submitted & Affiliated To**

****

**SARDAR VALLABHBHAI GLOBAL UNIVERSITY**

**Under the guidance of**

**Prof. Priyank Nahar**

**Submitted By:**

|  |  |
| --- | --- |
| **Niravkumar Tank** | **23CI2110123** |
| **Karmarajsinh Vaghela** | **23CI2110130** |
| **Payal Pithadiya** | **23CI2110123** |

### **Acknowledgment**

We would like to express our heartfelt gratitude to **Dr. Priyank Nahar**, our internal project guide, for his continuous support, insightful guidance, and valuable feedback throughout the development of this project. His expertise and encouragement have played a crucial role in shaping this project.

Finally, we would like to appreciate the support of our **college, faculty members, and teammates** for their contributions, collaboration, and dedication, which helped us successfully complete this project.

**Team Members:**

**Niravkumar Tank** (23CI2110123)

**Karmarajsinh Vaghela** (23CI2110130)

**Payal Pithadiya** (23CI2110095)

# **Table Of Contents**

|  |  |
| --- | --- |
| **COMPANY PROFILE** | 1 |
| **OVERVIEW OF COMPANY** | 2 |
| 1. Introduction | 2 |
| 1. Services Provided by Elsner | 2 |
| **PROJECT DEFINITION** | 3 |
| 1. Introduction | 3 |
| 1. Problem Specification | 3 |
| 1. Aim & Objectives | 3 |
| **TOOLS & TECHNOLOGY** | 4 |
| 1. Developer | 4 |
| 1. Hardware Requirements | 4 |
| 1. Software Requirements | 4 |
| 1. Client | 4 |
| 1. Hardware Requirements | 4 |
| 1. Software Requirements | 5 |
| **SYSTEM ANALYSIS** | 6 |
| 1. FEASIBILITY STUDIES | 6 |
| 1. Economic Feasibility | 6 |
| 1. Technical Feasibility | 6 |
| 1. Operational Feasibility | 7 |
| **SYSTEM REQUIREMENT** | 8 |
| 1. Functional Requirements | 8 |
| 1. Technical Requirements | 8 |
| 1. System Qualities | 9 |
| 1. Constraints and Assumptions | 9 |
| **OBJECT ORIENTED APPROACH** | 11 |
| 1. Class Diagram | 11 |
| 1. Use Case Diagram | 12 |
| 1. Activity Diagram | 13 |
| 1. Sequence Diagram | 15 |
| **DATA DICTIONARY** | 17 |
| **SCREENSHOTS OF THE SYSTEM** | 24 |
| **REPORTS** | 40 |
| **TEST CASES** | 42 |
| **LEARNING DURING THE JOURNEY OF THE PROJECT** | 49 |
| **CODING CONVENTIONS** | 50 |
| 1. General Guidelines | 50 |
| 1. Frontend (React.js) Conventions | 51 |
| 1. Backend (Spring Boot - Java) Conventions | 52 |
| 1. API Naming Conventions | 52 |
| **FUTURE ENHANCEMENTS** | 54 |
| **BIBLIOGRAPHY** | 54 |
| **APPENDIX** | 55 |
| 1. Planning of the Project Using Waterfall Model | 55 |
| 1. Scheduling of the Project According to Gantt Search | 56 |

# **Project Profile**

|  |  |
| --- | --- |
| **Title** | **Details** |
| **Project Title** | Juta.com |
| **Project In-short** | The **E-Commerce Shoe Management System** is a web-based platform designed to manage shoes, customers, orders, payments, and inventory efficiently. |
| **Project Goal** | The **E-Commerce Shoe Management System** aims to provide a secure, efficient, and user-friendly platform for managing shoes, customers, orders, payments, and inventory while ensuring seamless operations and accurate data handling. |
| **Project Category** | Web Application |
| **Development Tools and Technologies** | **Frontend:** Angular  **Backend:** .NET 8 Core Web API (C#, REST APIs) **Database:** MS SQL Server (Stored Procedures) |
| **Developed By** | **Niravkumar Tank** (23CI2110123)  **Karmarajsinh Vaghela** (23CI2110130)  **Payal Pithadiya** (23CI2110095) |
| **Institute Guidance by** | Prof. Priyank Nahar |
| **Organization Guidance by** |  |

# **Company Profile**

### **Prudent Corporate Advisory Services Ltd - IT Sector**

#### **Company Profile**

**Name**: Prudent Corporate Advisory Services Ltd - IT Division

**Website**: [https://prudentcorporate.com](https://prudentcorporate.com" \t "_new)

**Size**: 200+ Employees (IT & Financial Services)

**Email**: support@prudentcorporate.com

**Contact No**: +91 79 66000000

**Headquarter**: Ahmedabad, India

**Other Branches**: Mumbai, Delhi, Bangalore, Hyderabad

#### **About the IT Sector**

The **IT sector of Prudent Corporate Advisory Services Ltd** is at the forefront of **financial technology (FinTech), digital transformation, and cybersecurity**. The company integrates advanced **software solutions, AI-driven analytics, and secure cloud infrastructure** to offer seamless digital services for wealth management, stock trading, and financial advisory.

Prudent’s IT division ensures a **secure, scalable, and data-driven approach** for financial services, leveraging **automation, and AI-powered risk assessment** to enhance customer experience and operational efficiency.

#### **IT Services Provided by Prudent**

**FinTech Solutions** – Development of digital platforms for investment, mutual funds, and stock trading.  
 **AI-Powered Financial Analytics** – Advanced data analytics for smart investment decisions.  
 **Cybersecurity & Compliance** – Implementation of high-level security protocols and regulatory compliance.  
 **Cloud Computing & Infrastructure** – Cloud-based platforms for seamless and scalable financial operations.  
**Automated Trading & Risk Management** – AI-driven risk assessments and automated trading solutions.

#### **Address**

Prudent Corporate Advisory Services Ltd  
3rd Floor, Prudent House, Panjrapole Cross Road, Ambawadi, Ahmedabad - 380015, Gujarat, India.

By investing in **cutting-edge technology and digital solutions**, Prudent’s IT sector plays a vital role in **enhancing financial services, ensuring data security, and optimizing digital experiences for clients across India and beyond**.

# **Project Definition**

## **Introduction**

The **E-Commerce Shoes Management System(juta.com)** is a web-based platform designed to provide a seamless shopping experience for customers looking to purchase shoes online. The system offers an intuitive interface for users to browse products, add them to the cart, and complete purchases securely.

Built using **Angular** for the front-end, **.NET Core Web API** for the back-end, and **MS SQL Server** as the database, the system ensures smooth integration between components, efficient inventory management, and a secure transaction process.

The platform includes three types of users:

**Admin** – Manages inventory, categories, orders, payments, and users.

**Registered Users** – Can register, log in, manage profiles, search for products, add to cart, order products, and provide reviews.

**Visitors** – Can browse and search for products without registration.

This system ensures an organized and efficient way of managing shoe inventory and transactions.

## **Problem Specification**

Traditional shoe businesses often face challenges such as inventory mismanagement, inefficient order processing, and poor user experience. Some of the key challenges addressed by this system include:

Lack of an online platform for a diverse range of shoe collections.

Inefficient inventory tracking leading to stock shortages or excess supply.

Difficulty in managing user data, orders, and payments manually.

Limited product search and filtering options, making it hard for users to find the right products.

Security risks in payment transactions without a dedicated system.

## **Aim & Objectives**

### **Aims**

Develop a secure and user-friendly e-commerce platform for shoes.

Ensure efficient inventory and order management with role-based access.

Enhance product search and filtering to improve user experience.

Integrate secure payment processing to facilitate smooth transactions.

Enable an admin panel to manage products, categories, orders, and users.

### **Objectives**

Implement user authentication and authorization for secure access.

Develop a shopping cart and checkout system for seamless purchases.

Enable CRUD operations for product, category, order, and review management.

Generate real-time reports on sales, orders, and user activity.

Ensure responsive design for a smooth shopping experience on all devices.

## **Tools & Technology**

### **Developer Requirements**

#### **Hardware Requirements**

**Processor:** Intel Core i5 (10th Gen or later) / AMD Ryzen 5 or higher

**RAM:** Minimum 8GB (Recommended: 16GB)

**Storage:** SSD with at least 256GB (Recommended: 512GB or more)

**Internet Connection:** Stable broadband for development and testing

#### **Software Requirements**

**Operating System:** Windows 10/11, macOS, or Linux (Ubuntu 20.04 or later)

**Development Tools:**

Visual Studio Code – For Angular front-end development

Visual Studio – For .NET Core Web API development

Postman – For API testing

**Database:** MS SQL Server for structured data storage

**Build & Package Managers:**

Node.js & npm – For Angular dependencies

NuGet – For .NET dependencies

### **Client Requirements**

#### **Hardware Requirements**

**Processor:** Intel Core i3 (8th Gen or later) / AMD Ryzen 3 or higher

**RAM:** Minimum 4GB (Recommended: 8GB)

**Storage:** Minimum 128GB (Recommended: SSD with 256GB or more)

**Internet Connection:** Stable broadband or Wi-Fi connection

#### **Software Requirements**

**Operating System:** Windows 10/11, macOS, or Linux

**Web Browser:** Latest version of Chrome, Firefox, or Edge

**PDF Viewer:** Adobe Acrobat Reader (for viewing order invoices and reports)

### **System Analysis**

#### **Feasibility Study**

A feasibility study ensures that the system is practical, cost-effective, and meets all requirements.

### **Technical Feasibility**

The system's technical feasibility is evaluated based on the following aspects:

#### **1. Technology Stack**

**Frontend:** Angular for an interactive and dynamic UI.

**Backend:** .NET Core Web API for robust business logic and data processing.

**Database:** MS SQL Server for structured product, order, and user data.

#### **2. System Performance & Scalability**

Supports multiple users accessing the platform simultaneously.

MS SQL Server ensures efficient data retrieval and storage.

Angular provides a smooth and responsive user experience.

#### **3. Integration & Compatibility**

RESTful APIs enable seamless communication between frontend and backend.

The system is accessible on modern web browsers (Chrome, Firefox, Edge, Safari).

### **Economic Feasibility**

Economic feasibility ensures that the system provides high value at a reasonable cost while reducing long-term expenses.

#### **1. Development & Implementation Cost**

Uses open-source technologies such as Angular and .NET Core, reducing licensing costs.

Can be deployed on cloud platforms or on-premise servers based on budget constraints.

#### **2. Operational Cost Savings**

Reduces manual inventory management and streamlines order processing.

Saves costs on administrative work by automating product and order tracking.

#### **3. Long-Term Benefits**

One-time development investment with minimal maintenance costs.

Scalable architecture allows for future expansion.

### **Operational Feasibility**

Operational feasibility focuses on how well the system meets user needs and integrates into existing workflows.

#### **1. User-Friendly Interface**

Customers can browse, search, and purchase shoes effortlessly.

Admins can manage product inventory efficiently.

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

**Admin**: Manages products, categories, orders, and user details.

**Users**: Browse products, place orders, and manage their accounts.

**Visitors**: Can view products and search for items but need to register to make purchases.

#### **3. Reporting & Order Tracking**

Generates detailed order reports for business analysis.

Tracks user purchase history for better customer insights.

## **System Requirements**

### **Functional Requirements**

The E-Commerce Shoe Website includes various functionalities for efficient product and order management.

#### **1. Admin Module**

Login authentication.

Manage products, categories, and orders.

View user details and payment information.

#### **2. User Module**

Registration and login.

Manage profile and order history.

Browse and filter products, add to cart, and complete purchases.

#### **3. Visitor Module**

View product catalog.

Search and filter products.

#### **4. Shopping & Payment Module**

Add/remove products from the cart.

Secure checkout and payment processing.

#### **5. Review Module**

Users can leave reviews for purchased products.

### **Technical Requirements**

#### **1. Frontend Requirements**

**Technology:** Angular

**Packages:** FormsModule, HttpClientModule, RouterModule

**Browser Compatibility:** Chrome, Firefox, Edge, Safari

#### **2. Backend Requirements**

**Technology:** .NET Core Web API

**Libraries:** ADO.NET

**API Communication:** RESTful APIs

#### **3. Database Requirements**

**Database:** MS SQL Server

#### **4. Security Requirements**

**Role-Based Access Control (RBAC) for Admin and Users**

## **System Qualities**

The Election Management System (EMS) is designed to be reliable, secure, and efficient. Below are the key system qualities that ensure its effectiveness:

|  |  |
| --- | --- |
| Quality | Feature |
| Reliability | Ensures accurate product and order data management without failures. |
| Security | Implements Role-based access control. |
| Scalability | |  | | --- | |  |  |  | | --- | | Supports growing product catalog and increasing user base. | |
| Performance | Optimized database queries for fast data retrieval. |
| Maintainability | Modular architecture allows easy updates and enhancements. |
| Usability | Provides a user-friendly interface for customers and admins. |
| Transparency | |  | | --- | |  |  |  | | --- | | Tracks order history and user actions for accountability | |

## **Constraints and Assumptions**

### **Constraints (System Limitations)**

**User Authentication** – Only registered users can place orders.

**Role-Based Access** – Admins have full control, while users can only manage their accounts.

**Internet Dependency** – The system requires an active internet connection.

### **Assumptions (Pre-Conditions for System Functionality)**

**Users have basic knowledge of online shopping.**

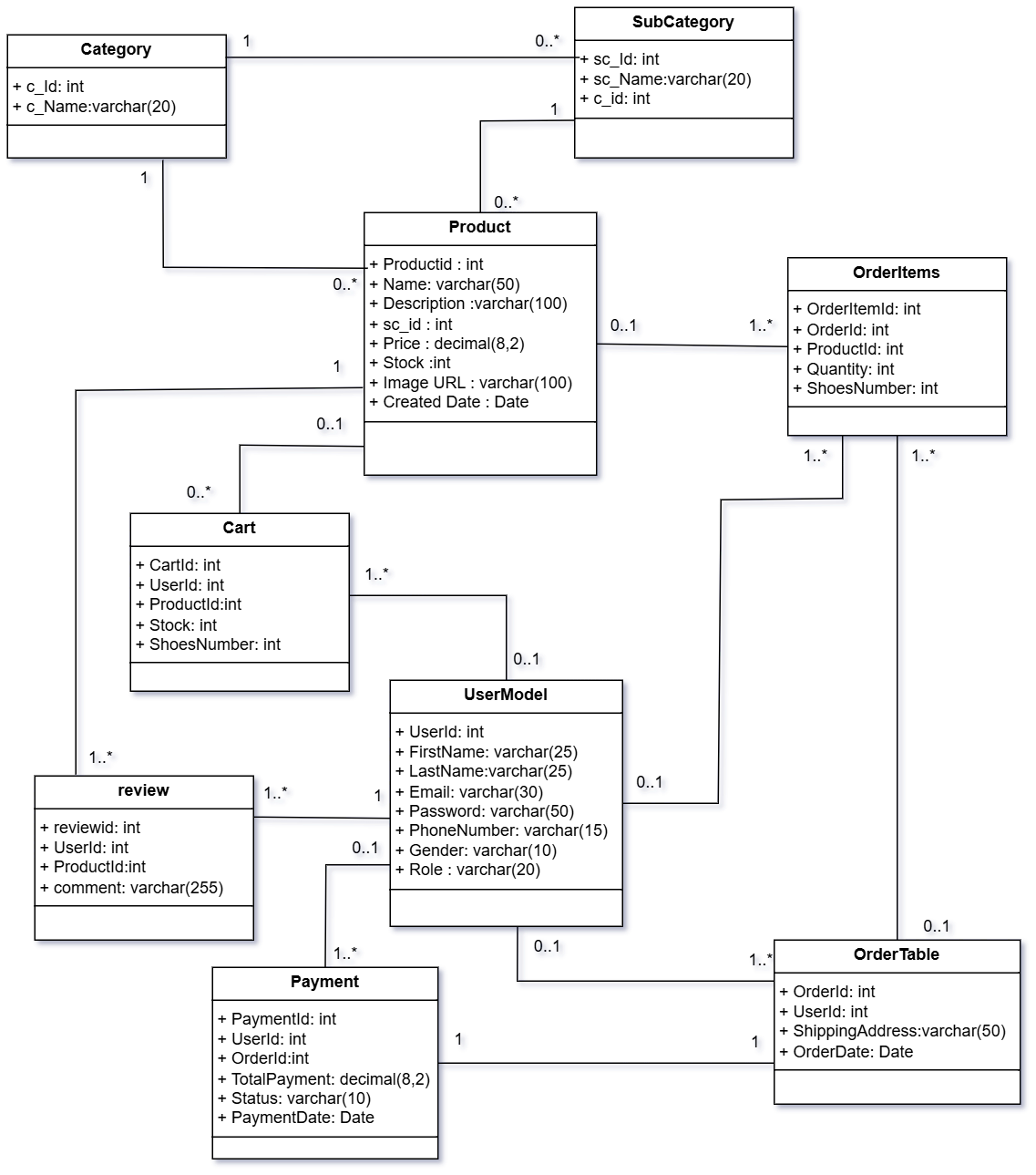
**Valid product data is entered into the system.**

**Secure payment gateway integration is available.**

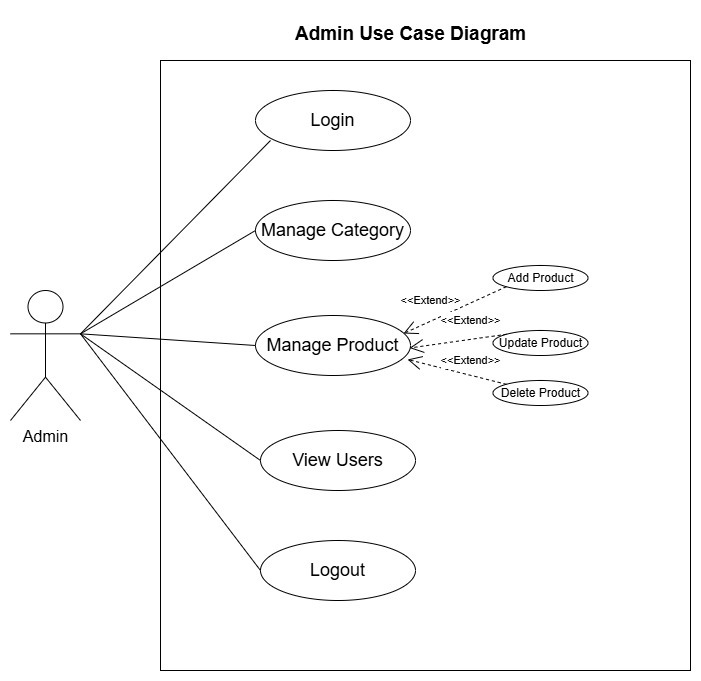
**Regular maintenance is performed to keep the system updated.**

# **Digrams**

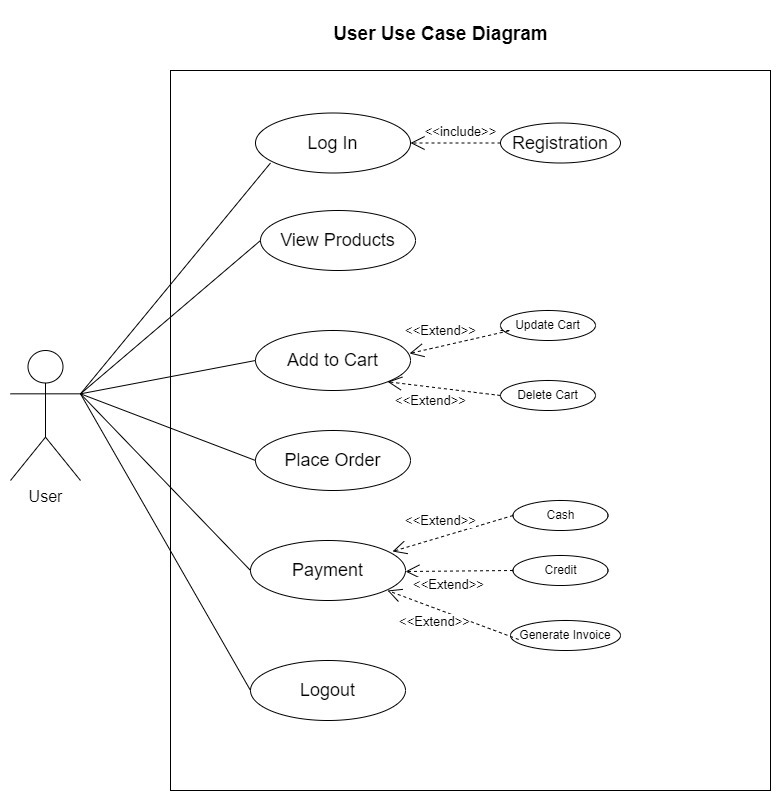
## **Class Diagram**



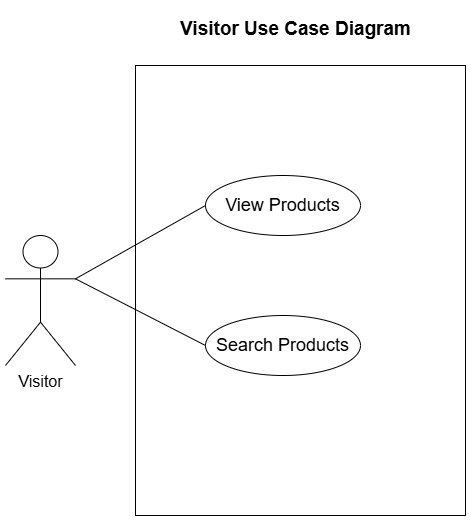
## **Use Case Diagram For Admin**



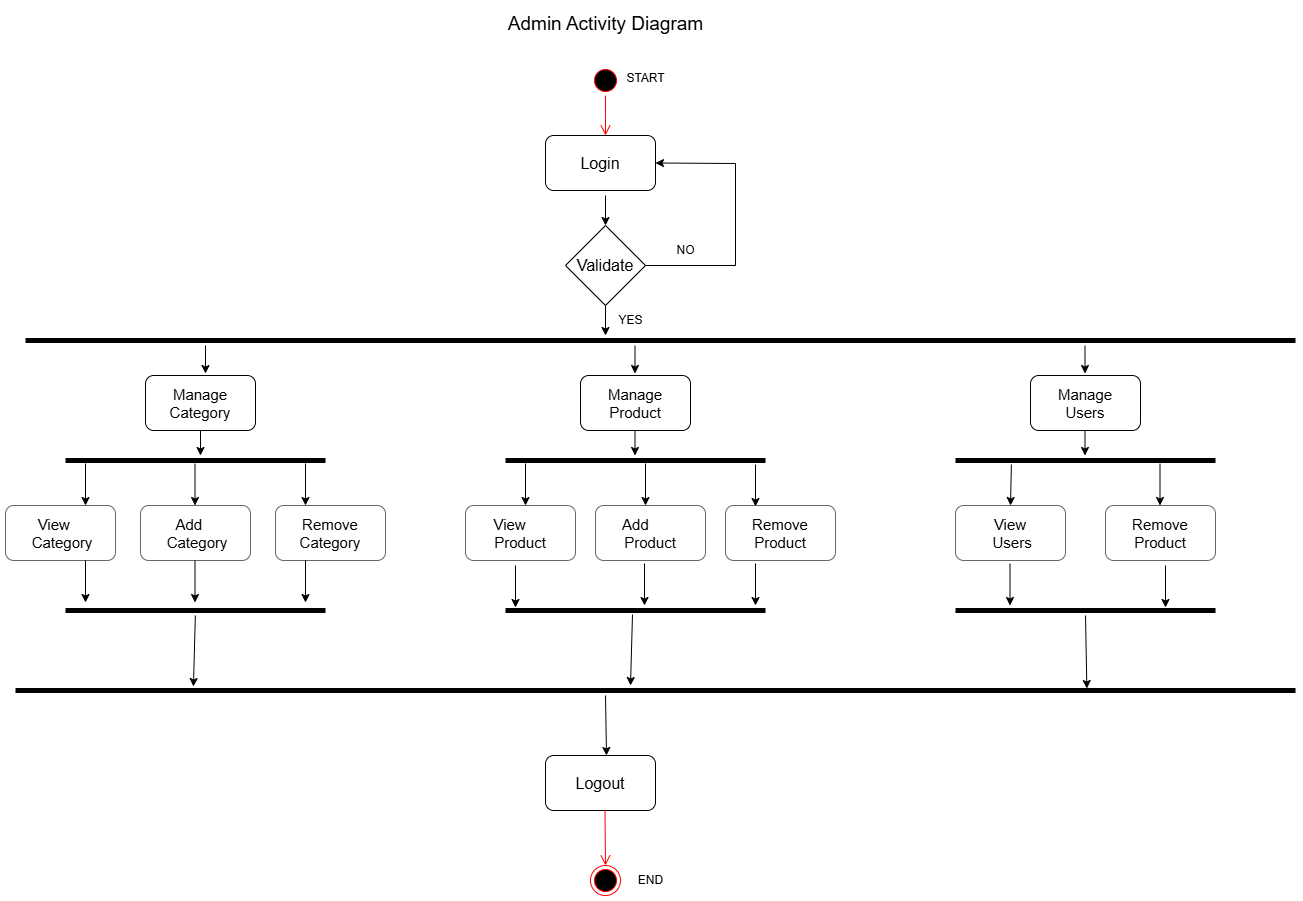
## **Use Case Diagram For User**



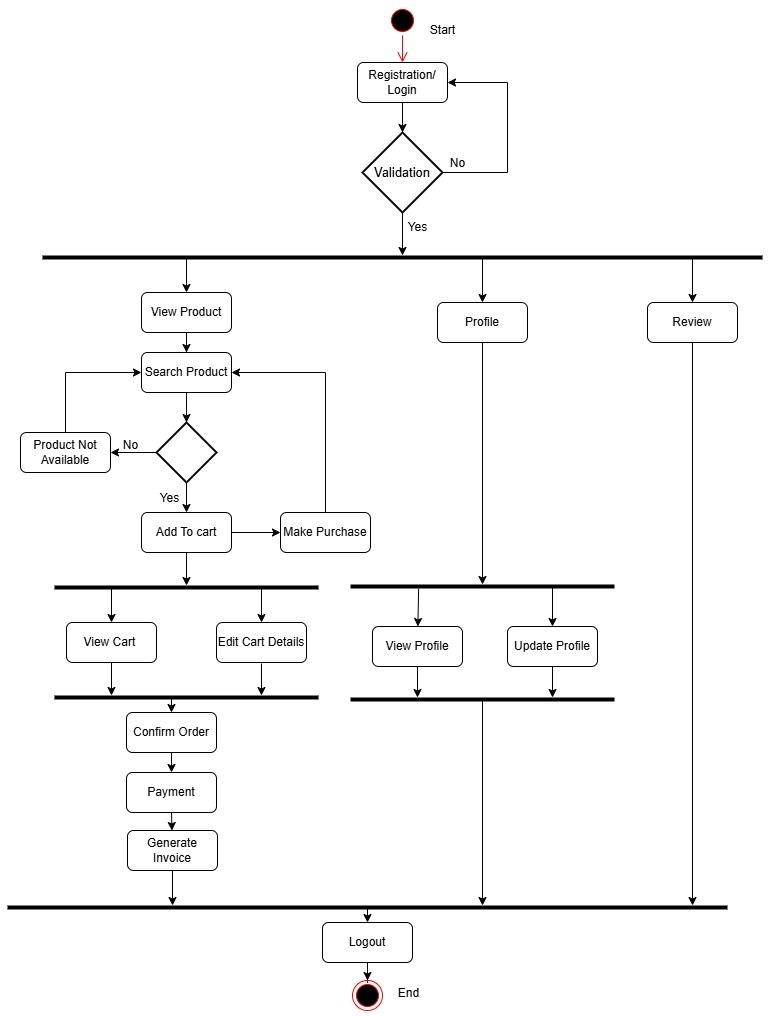
## **Use Case Diagram For visitor**



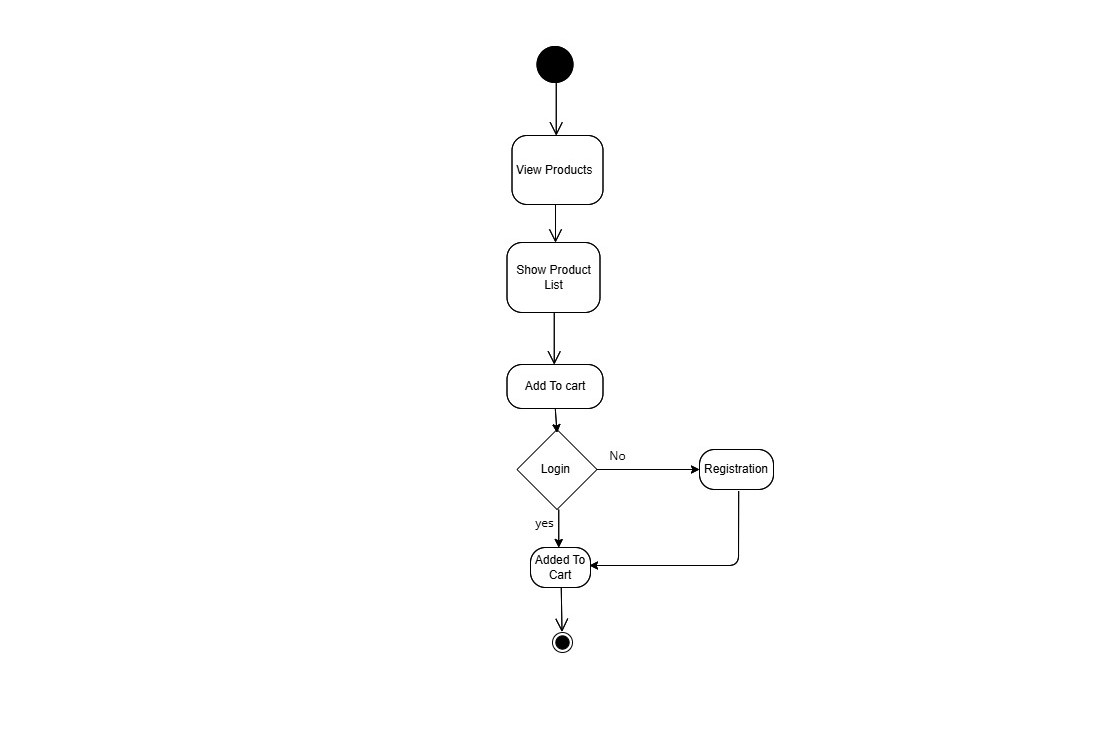
### Admin Activity Diagram



### User Activity Diagram

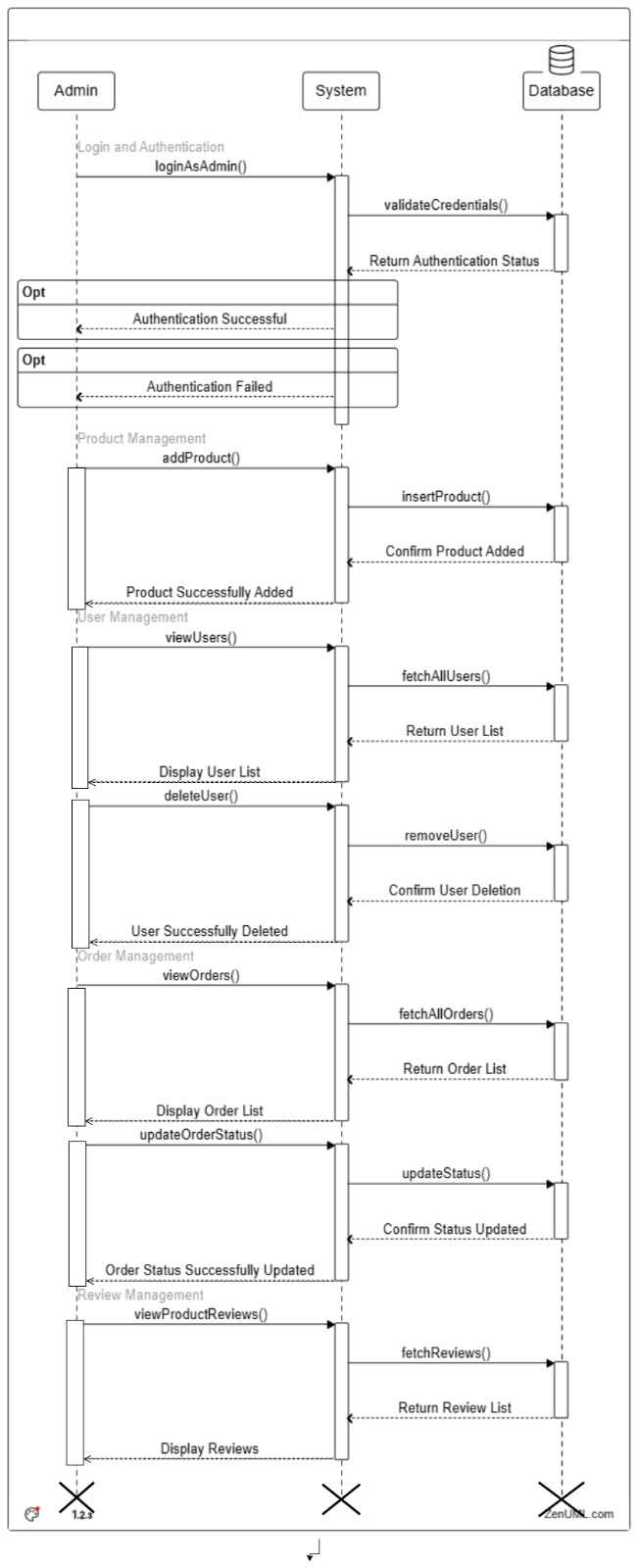


### Visitor Activity Diagram

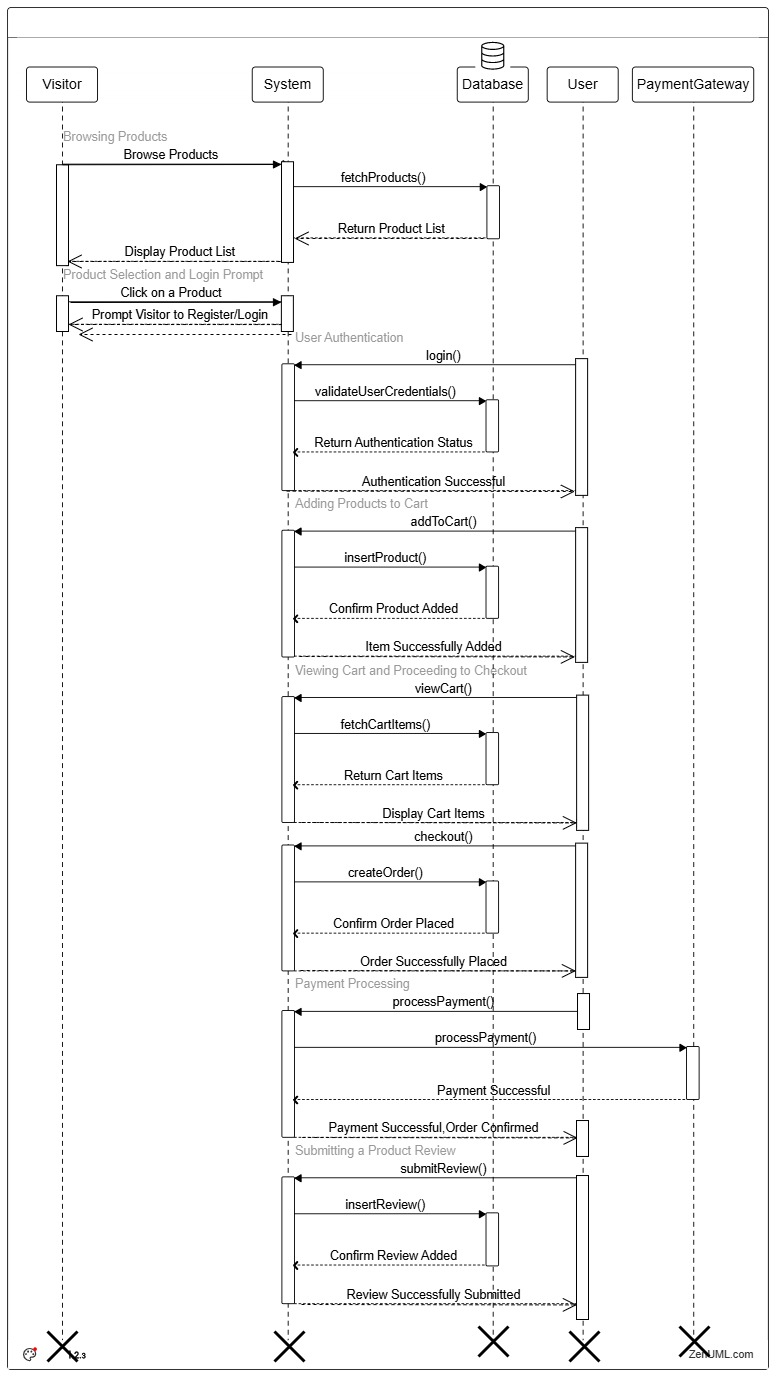


## **Sequence Diagram**

### Admin Sequence Diagram



### User Sequence Diagram



# **Data Dictionary**

# **User Table**

|  |  |  |
| --- | --- | --- |
| Column Name | Data Type | Description |
| UserId | int | Primary Key, Identity column for User |
| FirstName | varchar(25) | User's first name (cannot be null) |
| LastName | varchar(25) | User's last name (cannot be null) |
| Email | varchar(30) | User's email (cannot be null) |
| Password | varchar(50) | User's password (cannot be null) |
| PhoneNumber | varchar(15) | User's phone number (cannot be null) |
| Gender | varchar(10) | User's gender |
| Role | varchar(20) | Role of the user, default is 'User' |

# **Product Table**

|  |  |  |
| --- | --- | --- |
| Column Name | Data Type | Description |
| ProductId | int | Primary Key, Identity column for Product |
| Name | varchar(50) | Name of the product (cannot be null) |
| Description | varchar(100) | Description of the product (cannot be null) |
| c\_id | Int | Foreign Key to the Category table (cannot be null) |
| Price | decimal(8, 2) | Price of the product (cannot be null) |
| Stock | Int | Stock quantity available (cannot be null) |
| ImageUrl | varchar(100) | URL of the product image |
| CreatedDate | Date | Date the product was created, default to current date |

# **Order Table**

|  |  |  |
| --- | --- | --- |
| Column Name | Data Type | Description |
| OrderId | int | Primary Key, Identity column for the Order |
| UserId | int | Foreign Key referencing UserModel(UserId) |
| ShippingAddress | varchar(100) | Shipping address for the order (cannot be null) |
| OrderDate | date | Date when the order was created, default to current date |
| Status | varchar(10) | Status of the order, default is 'false' |

# **OrderItems Table**

|  |  |  |
| --- | --- | --- |
| Column Name | Data Type | Description |
| OrderItemId | int | Primary Key, Identity column for the Order Item |
| OrderId | int | Foreign Key referencing OrderTable(OrderId) |
| ProductId | int | Foreign Key referencing Product(ProductId) |
| Quantity | int | Quantity of the product in the order (cannot be null) |
| ShoesNumber | int | ShoesNumber (cannot be null) |

# **Cart Table**

|  |  |  |
| --- | --- | --- |
| Column Name | Data Type | Description |
| CartId | int | Primary Key, Identity column for Cart |
| UserId | int | Foreign Key referencing UserModel(UserId) |
| ProductId | int | Foreign Key referencing Product(ProductId) |
| Stock | int | Quantity of the product in the cart (cannot be null) |
| ShoesNumber | int | ShoesNumber (cannot be null) |

# **Category Table**

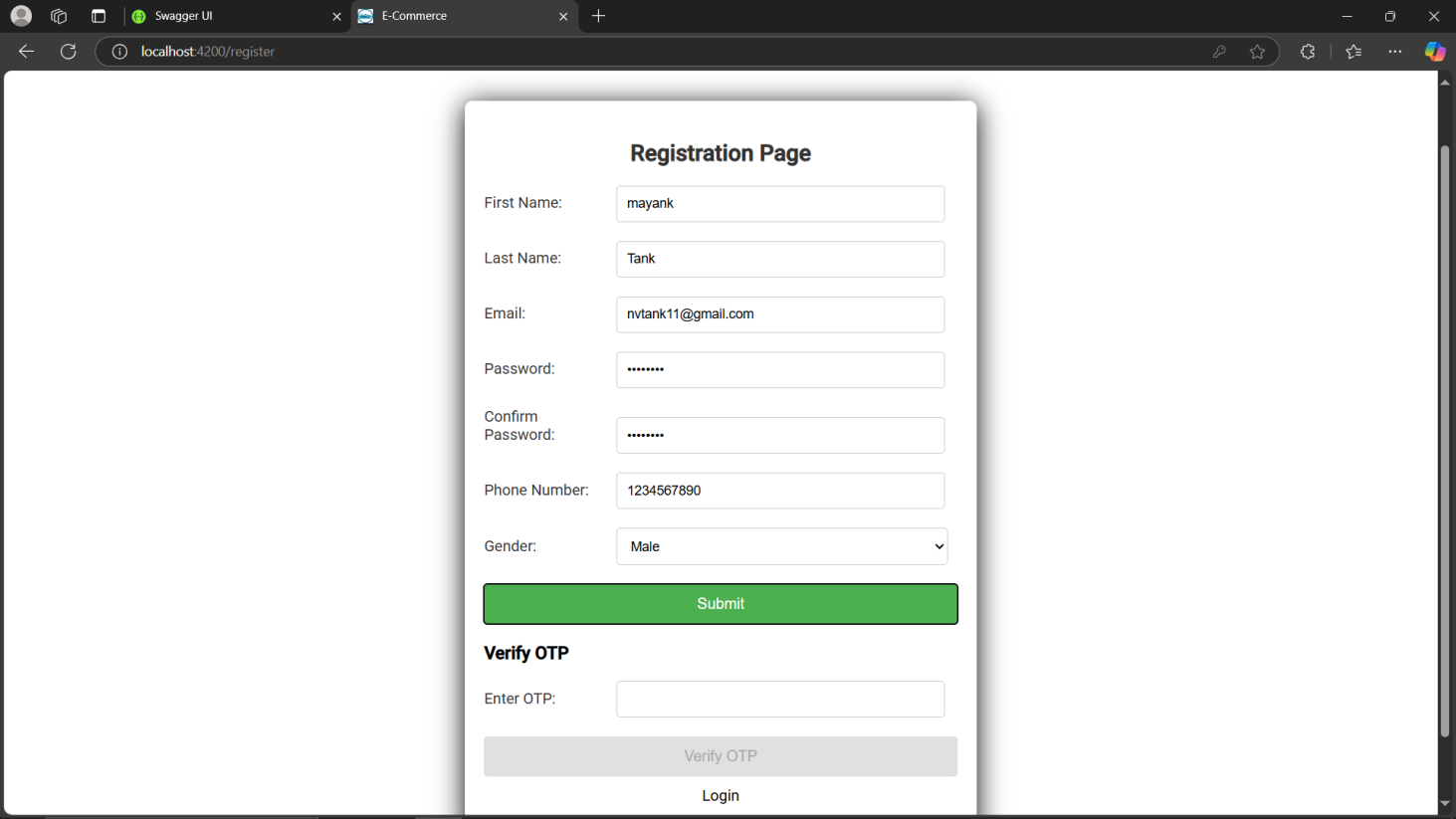
|  |  |  |
| --- | --- | --- |
| Column Name | Data Type | Description |
| c\_id | int | Primary Key, Identity column for Category |
| c\_Name | varchar(20) | Name of the category (cannot be null) |

# **Sub-Category Table**

|  |  |  |
| --- | --- | --- |
| Column Name | Data Type | Description |
| sc\_id | int | Primary Key, Identity column for Subcategory |
| sc\_Name | varchar(20) | Name of the Subcategory (cannot be null) |

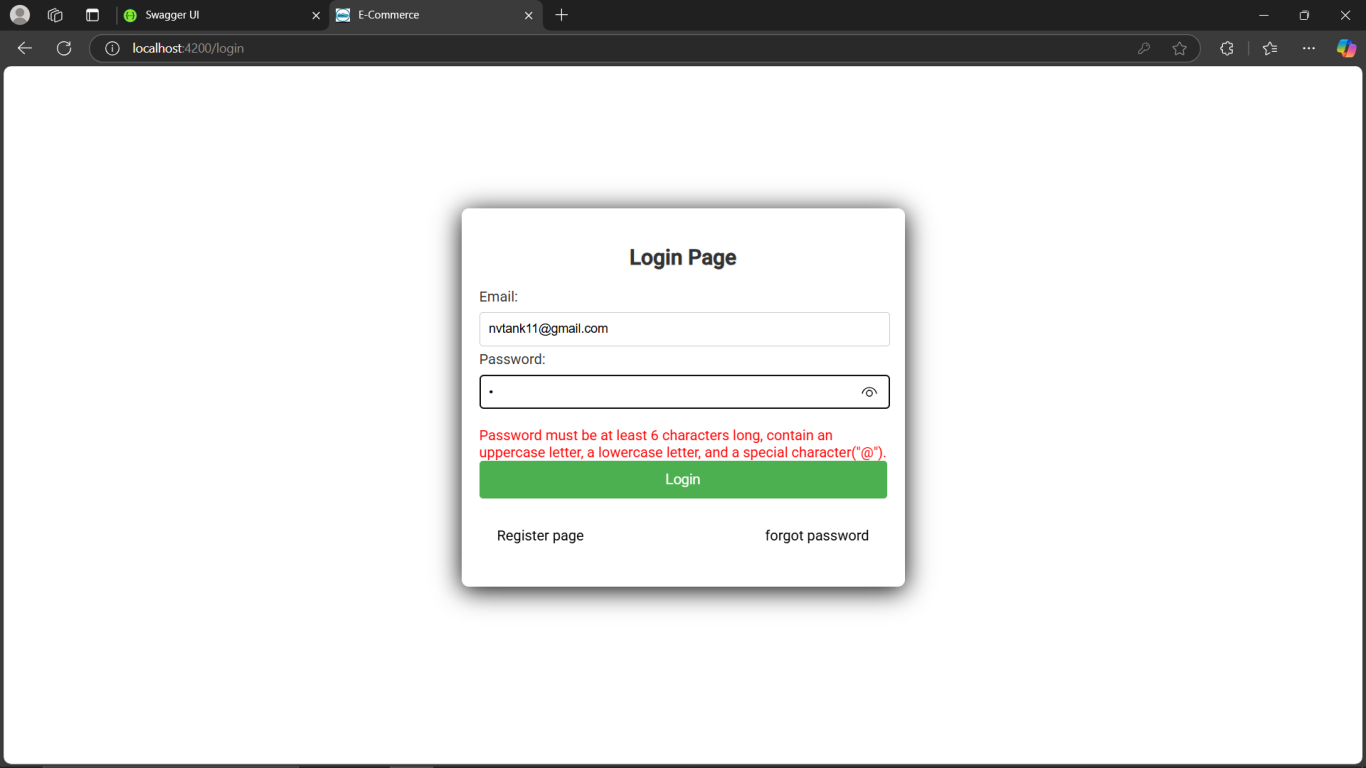
# **Screen Shots of The System**

## Registration page

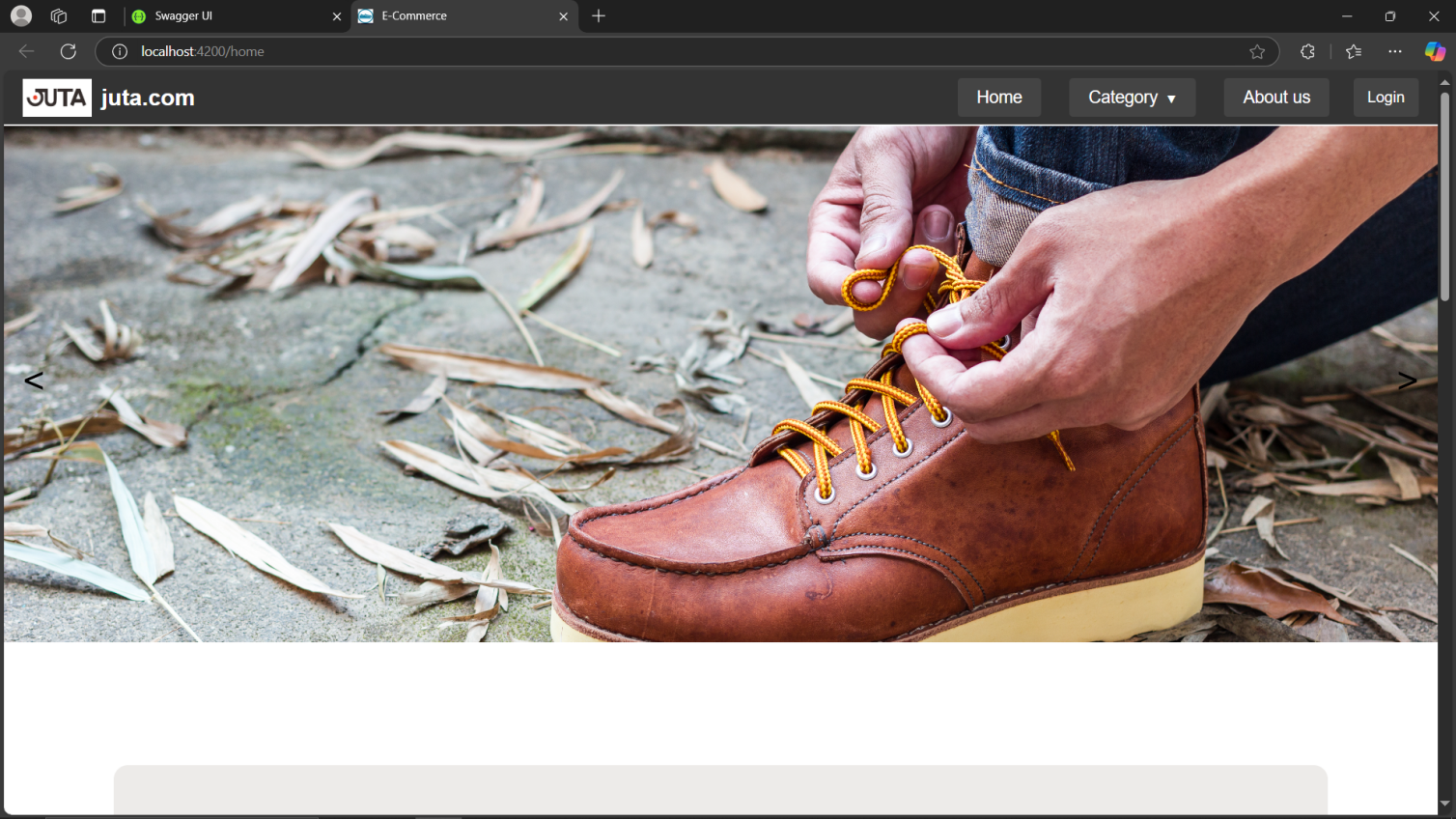


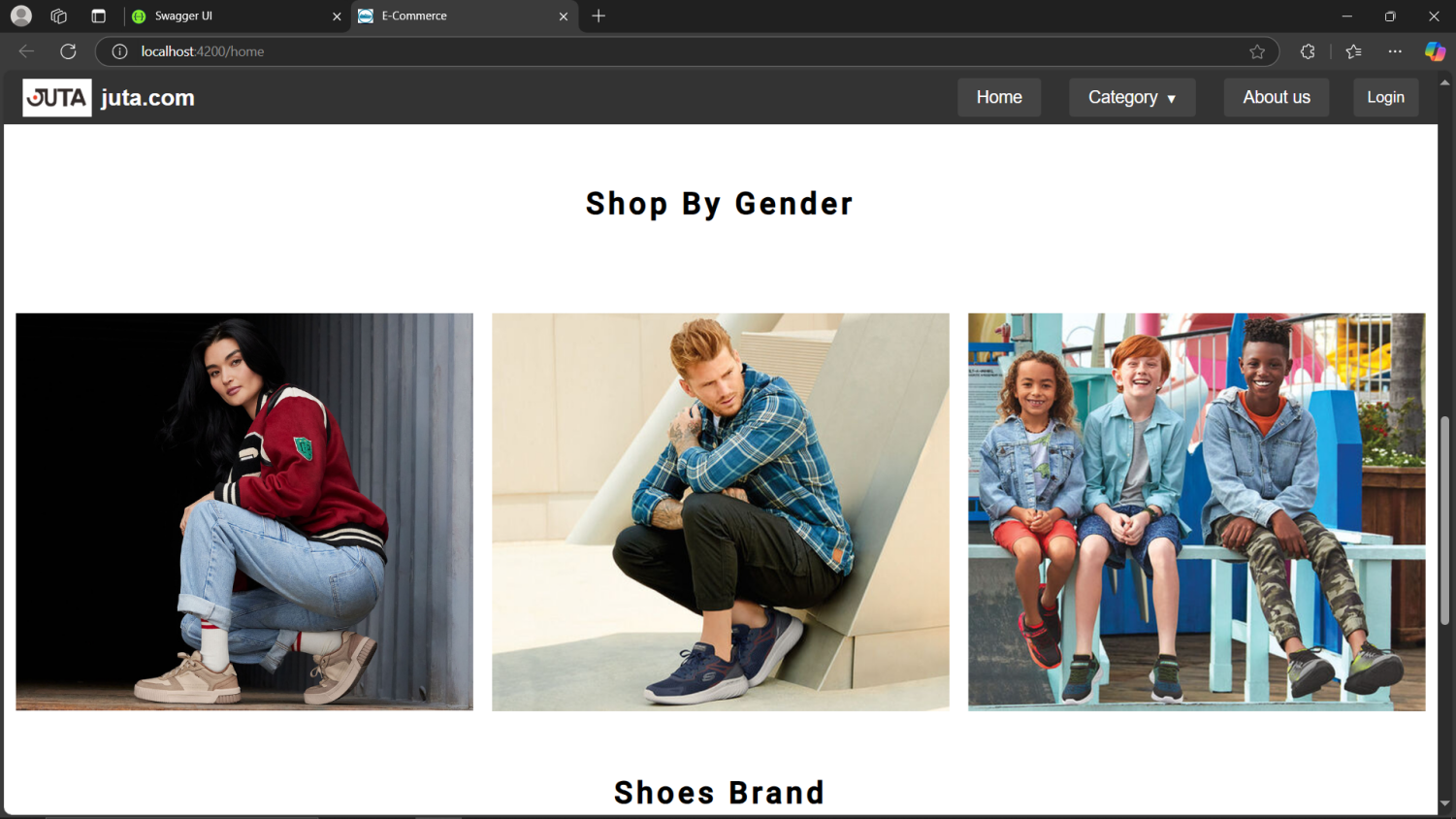
## 

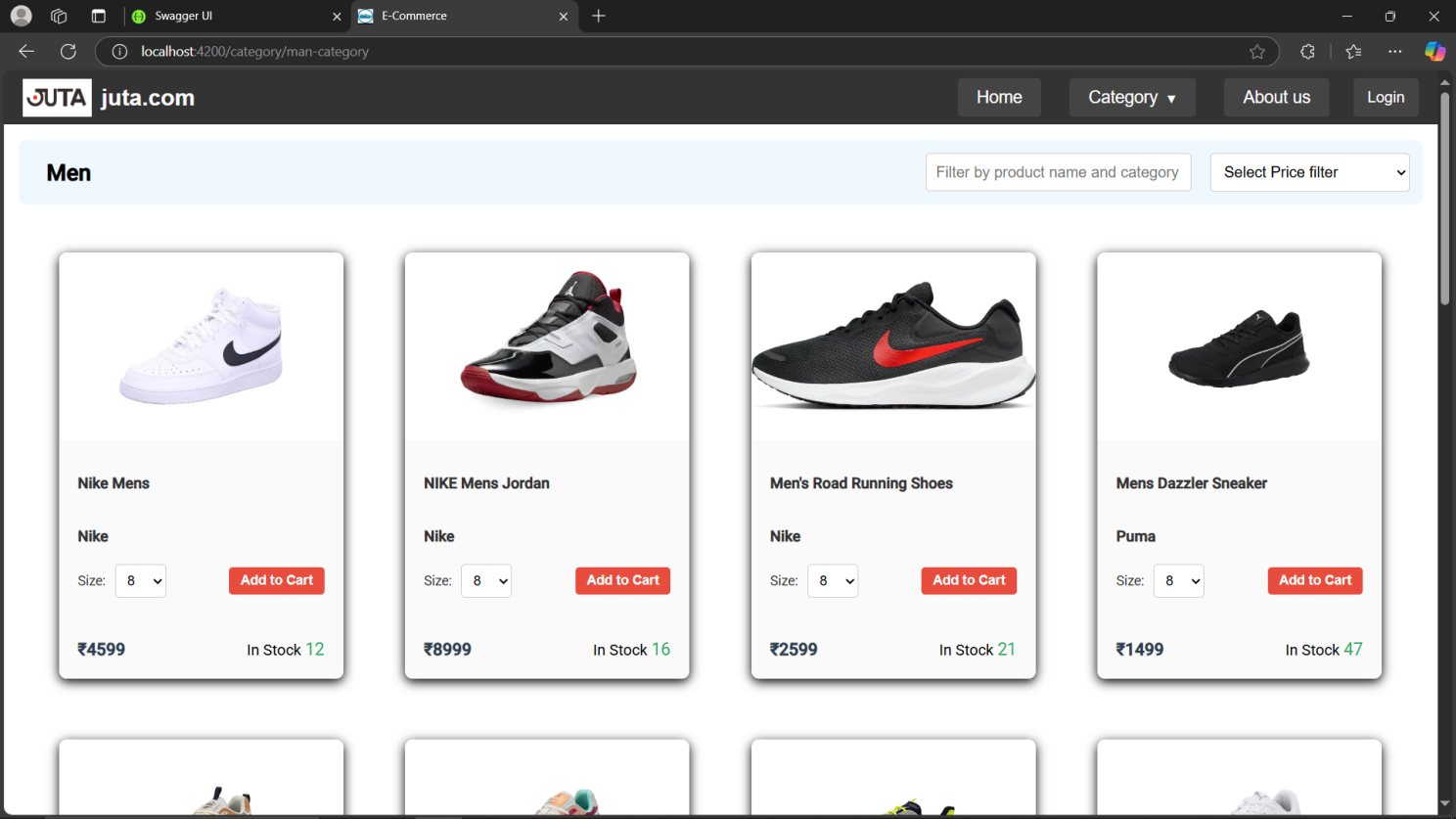
## Login page

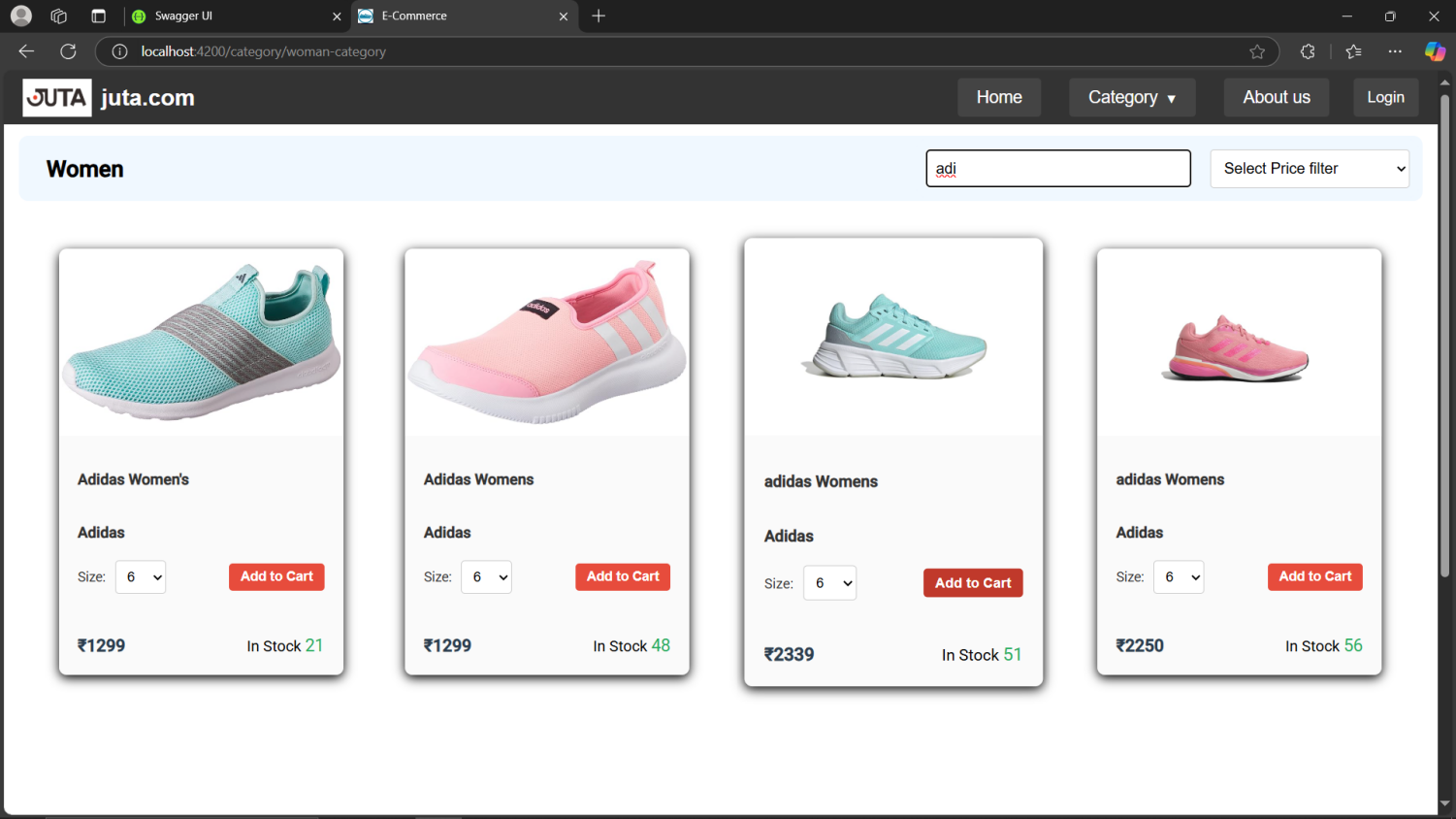


## Home page

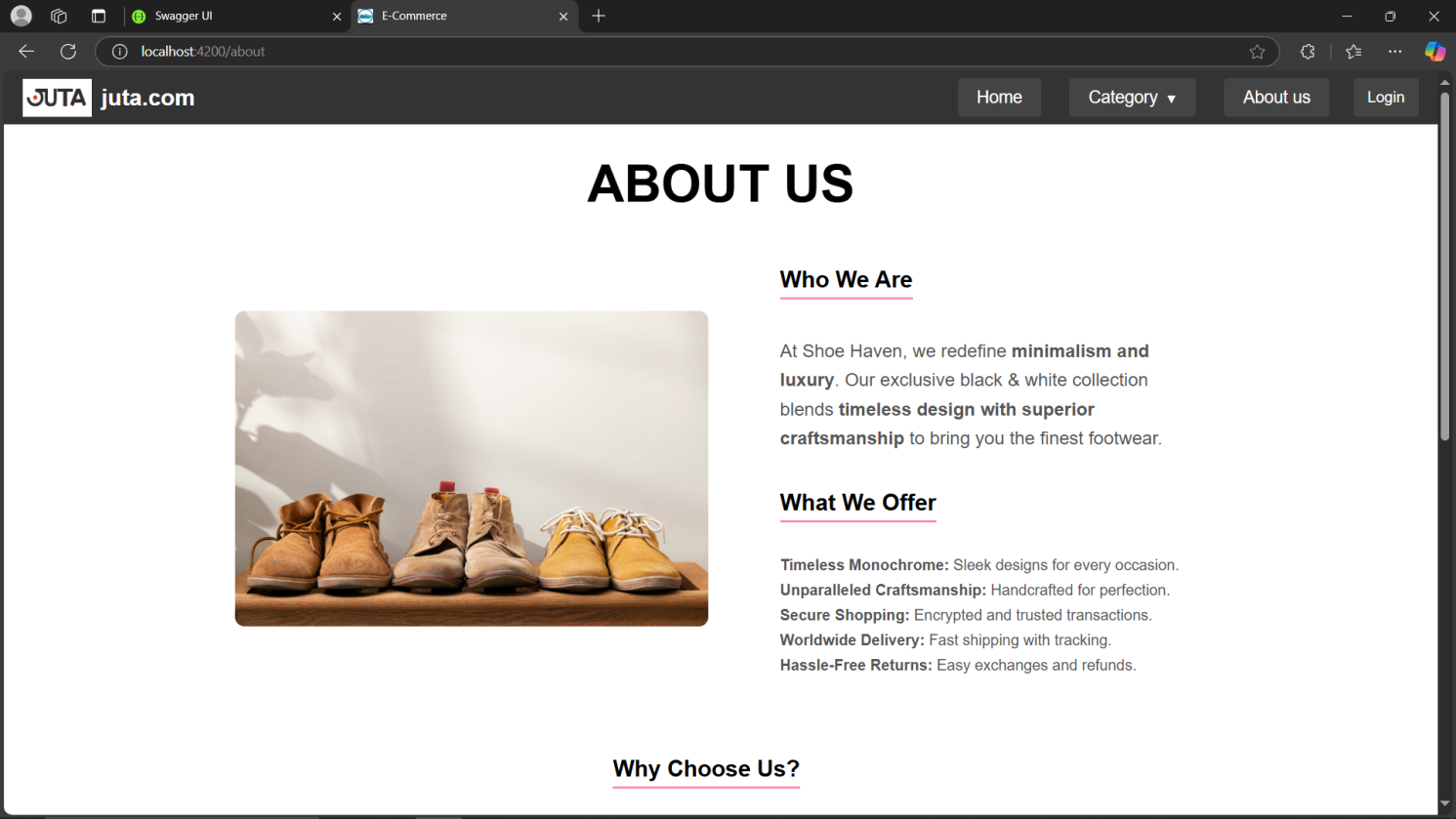




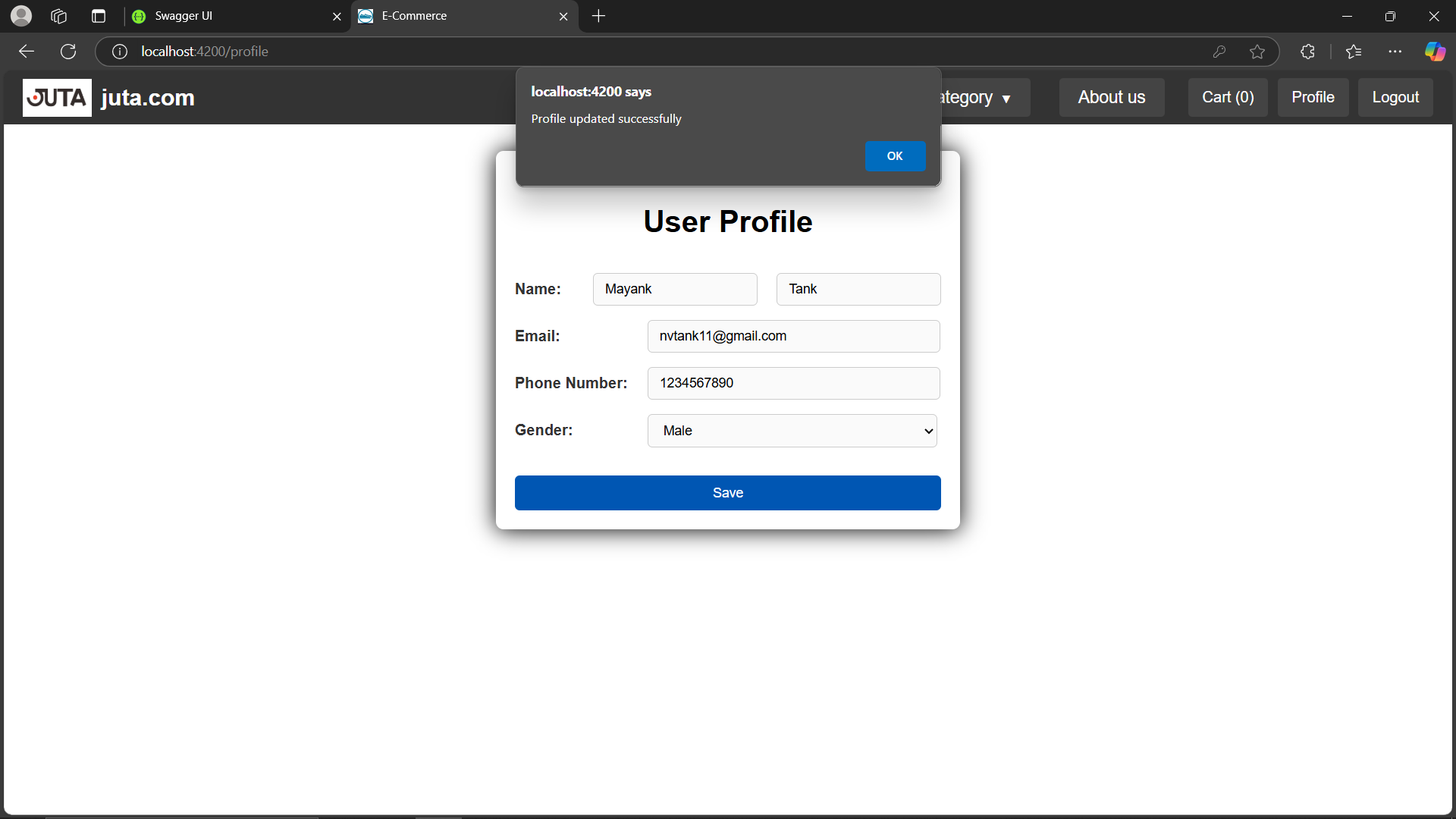


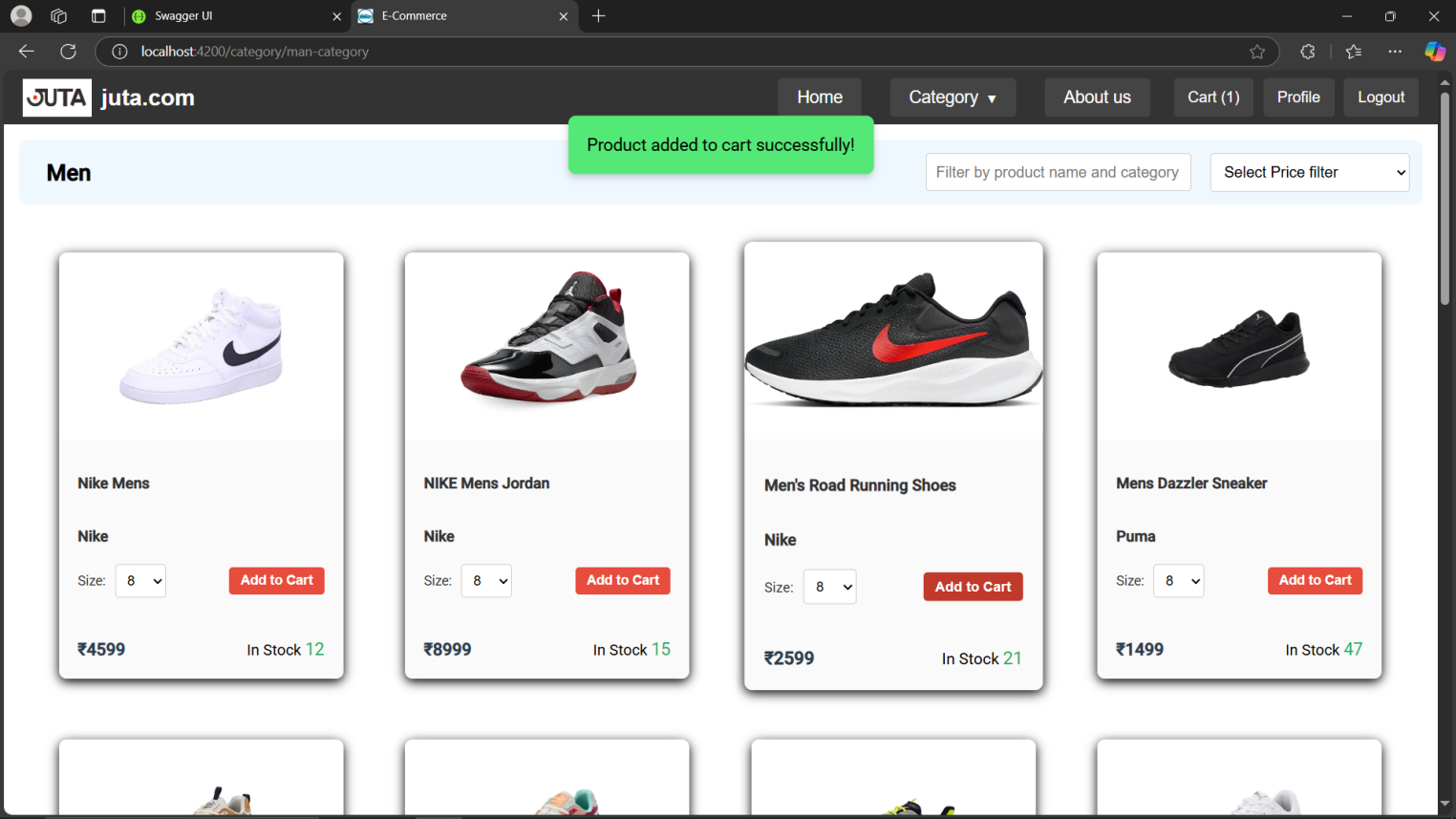


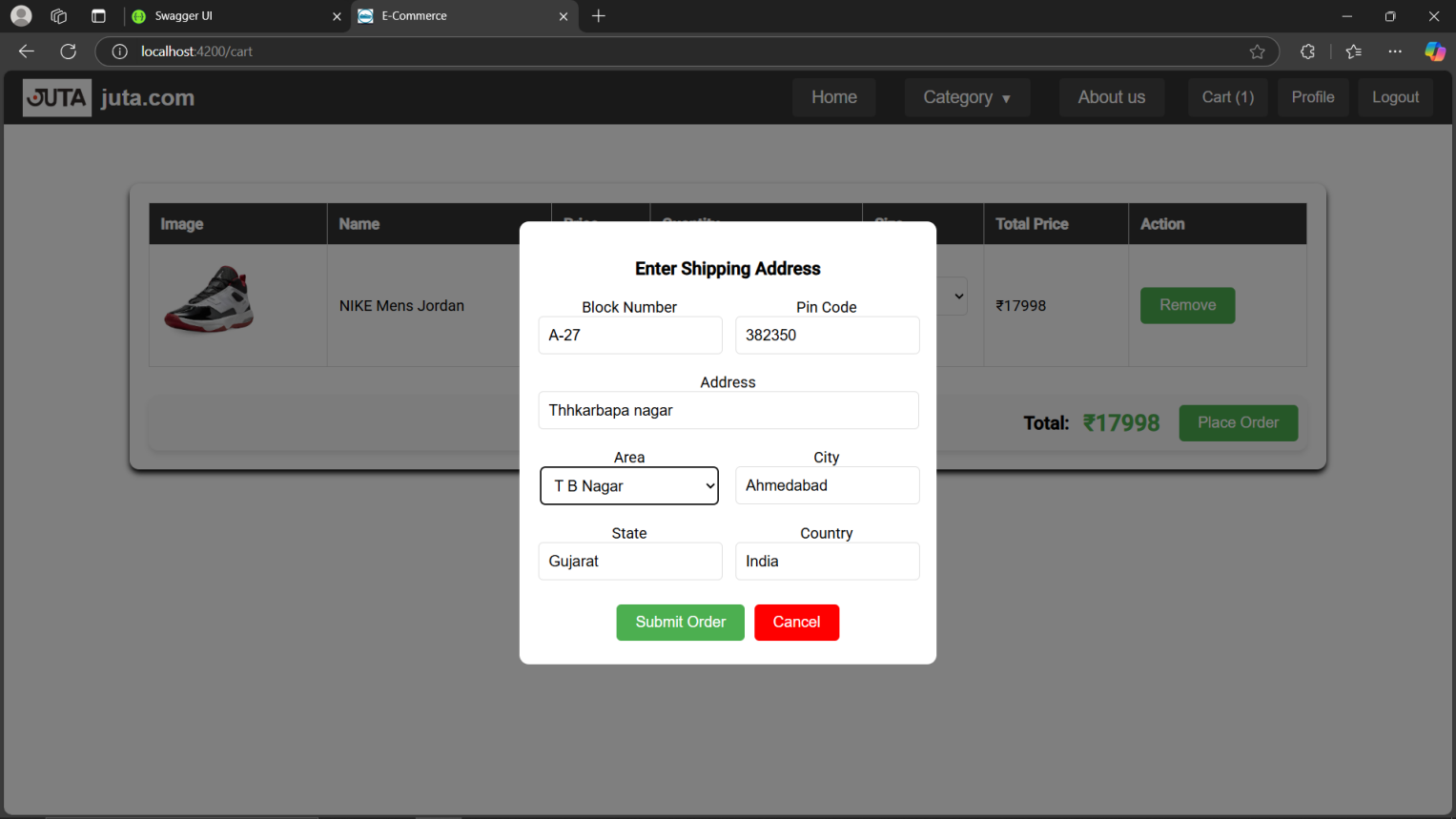
## About-Us page

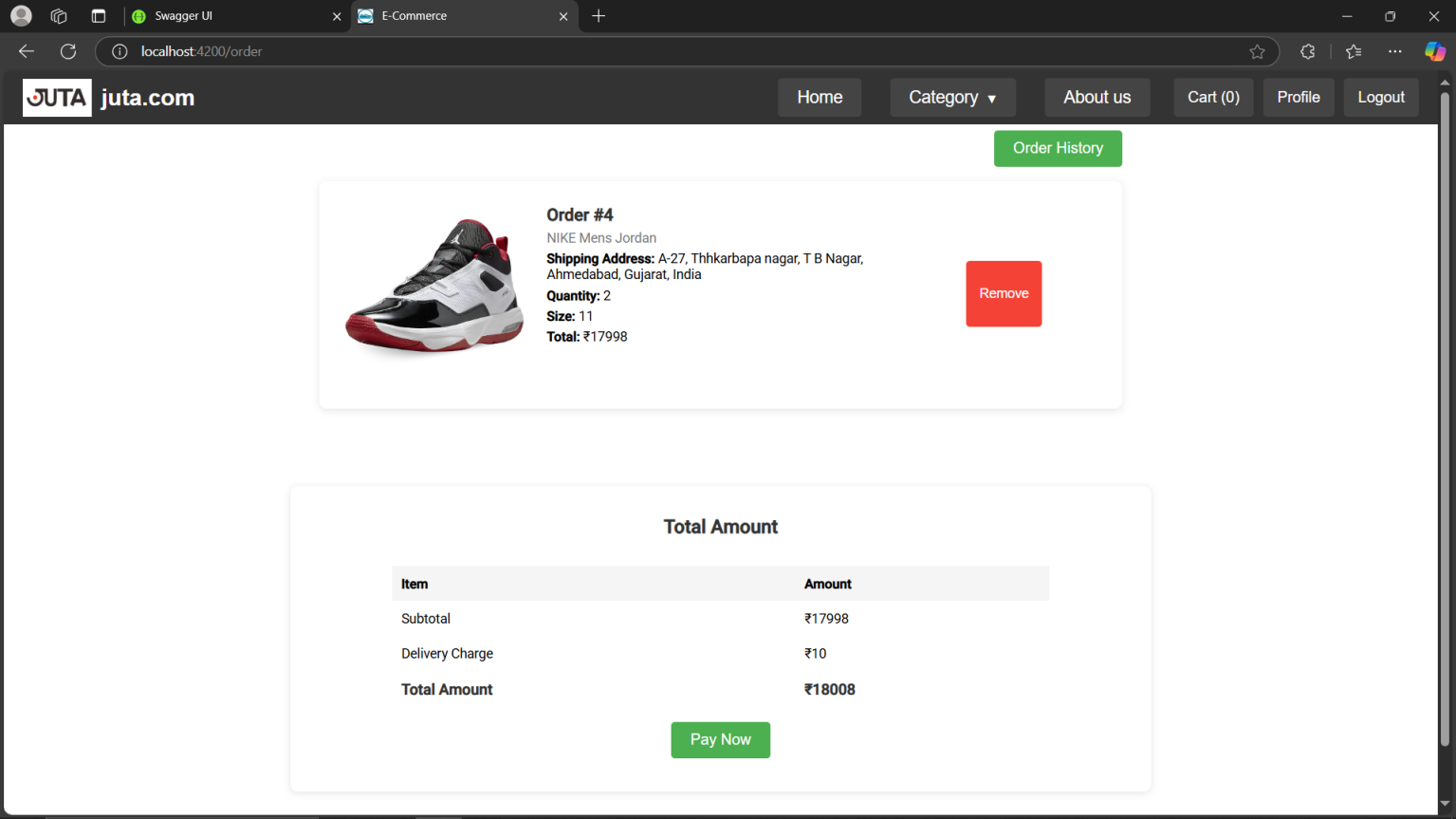


## Admin page

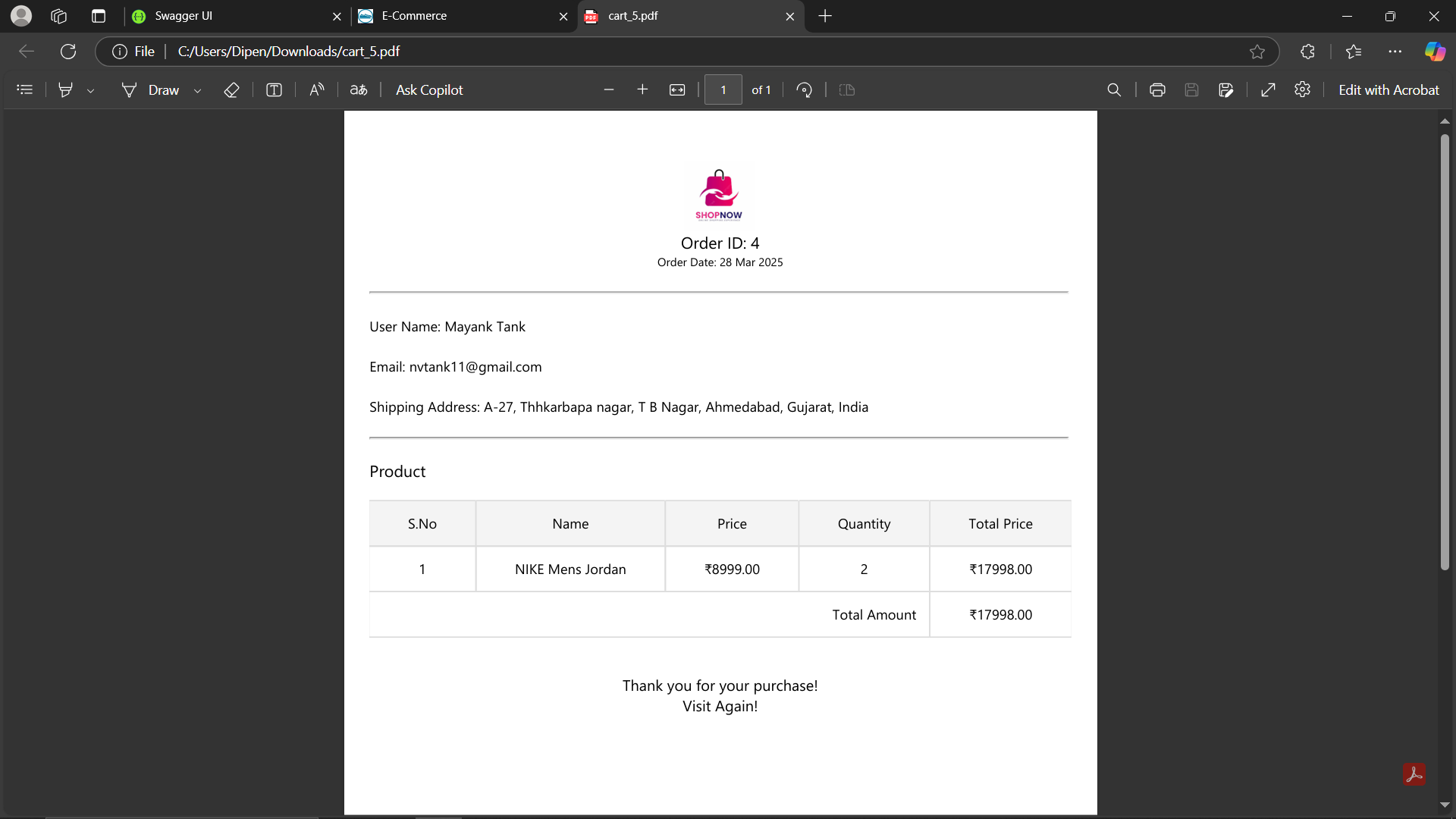








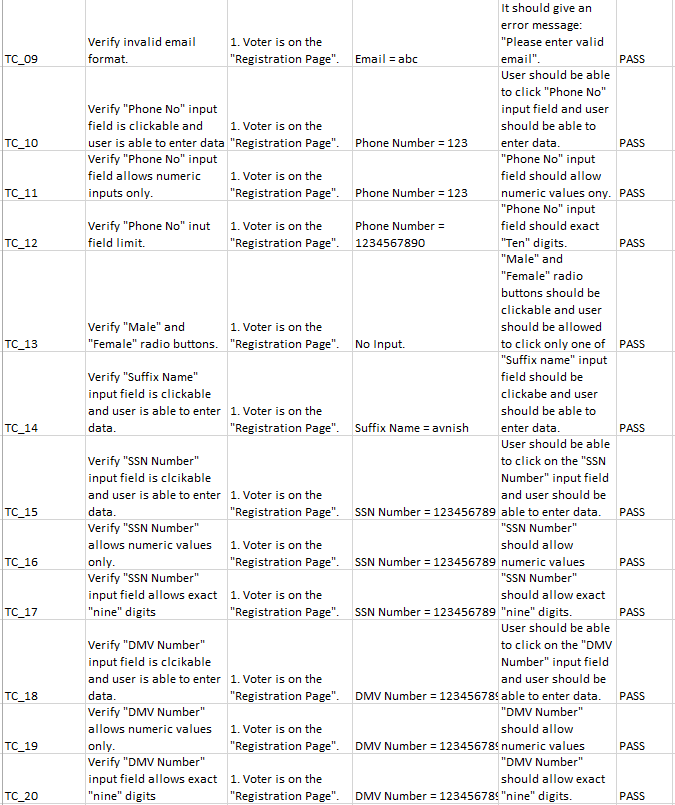
## Generate Bill

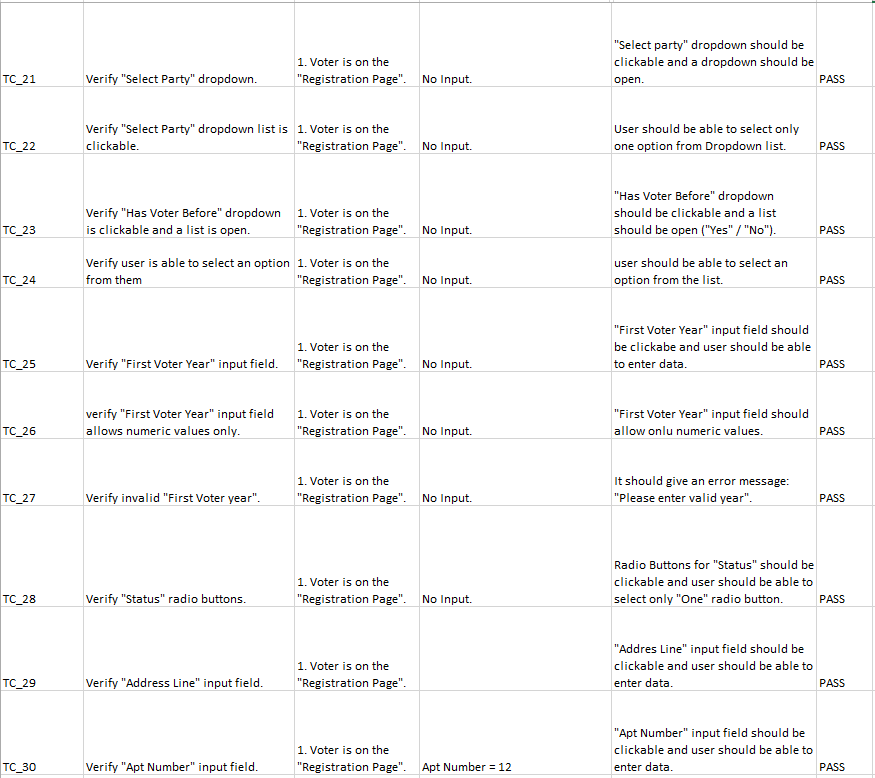


# **Test Case Report**

* Voter Module







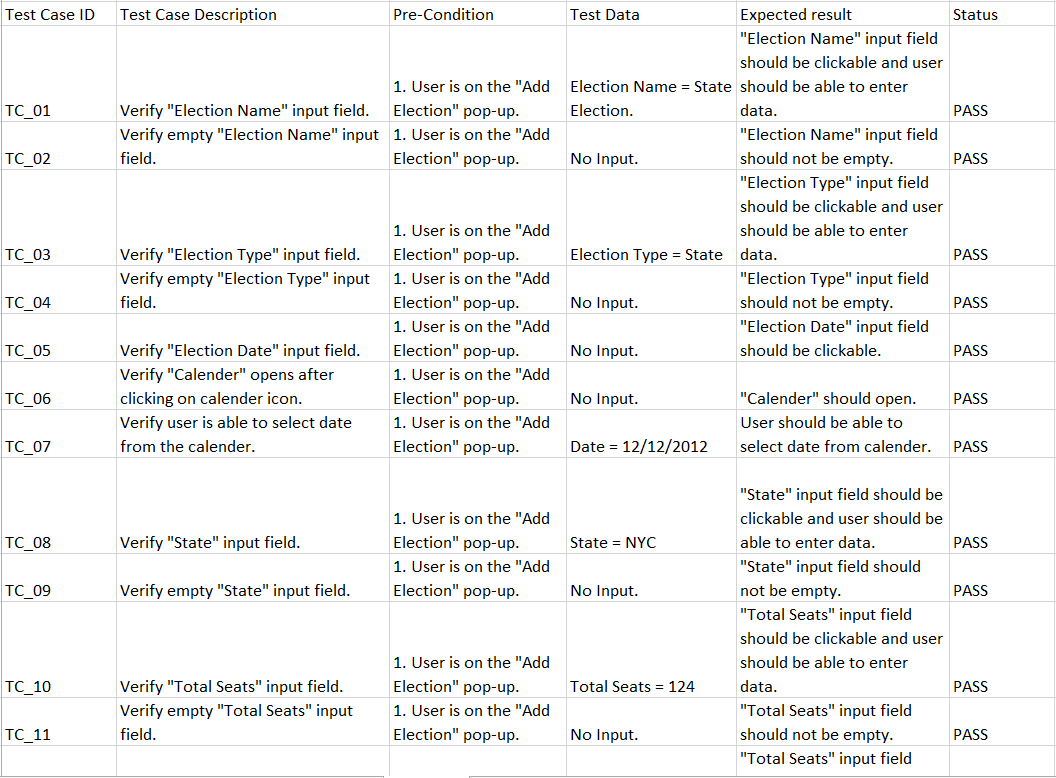


* Party Module





* Election Module



# **Learning During the Journey of the Project**

· **Full-Stack Development** – Implemented a seamless integration of **Angular (frontend)** and **.NET Core Web API (backend)** to create a dynamic and responsive shopping experience.

· **Database Management** – Worked with **MS SQL Server** to efficiently store and manage product, user, and order data.

· **RESTful API Design** – Developed secure and well-structured APIs for smooth communication between the frontend and backend.

· **Role-Based Access Control** – Implemented different access levels for **admins, registered users, and visitors** to ensure secure operations.

· **Error Handling & Debugging** – Improved skills in exception handling, debugging, and performance optimization for better system reliability.

· **Agile Development Practices** – Followed structured **planning, coding, testing, and review cycles** to maintain project quality.

· **Security Best Practices** – Implemented **authentication, authorization, and secure payment methods** to safeguard user data.

· **Shopping & Order Management** – Developed key features like **cart management, order processing, and payment integration** for a seamless shopping experience.

· **Admin Panel Development** – Created an intuitive **admin dashboard** to manage products, categories, orders, and users efficiently.

· **Team Collaboration & Problem-Solving** – Enhanced **communication, teamwork, and problem-solving skills** while working on different project modules.

# **Bibliography**

The following resources were referenced while developing the **Election Management System (EMS)**:

· **Official Documentation**

* **.NET Tutorials** – For backend development using .NET Core Web API and RESTful API implementation.

**Angular Documentation** – For frontend development and building dynamic, component-based UI applications.

**MS SQL Server Documentation** – For relational database management and query optimization.

· **Web Articles & Tutorials**

**.NET Tutorials** – Guides on developing secure and efficient Web APIs using .NET Core.

**Angular Blogs & Tutorials** – Step-by-step explanations of Angular concepts, including services, routing, and state management.

**Trusted SQL Server Resources** – Best practices for database design, indexing, and query performance optimization.

# **Appendix**

## **Planning of the Project Using Waterfall Model**

The Waterfall Model is a linear and sequential software development approach where each phase must be completed before moving to the next. It is best suited for projects with well-defined requirements and minimal expected changes.

Phases of the Waterfall Model:

1. Requirement Gathering: Collecting and documenting project requirements.
2. System Design: Creating architecture, UI, and database design.
3. Implementation: Writing code and developing the system based on design.
4. Testing: Identifying and fixing bugs to ensure functionality.
5. Deployment: Releasing the system for real-world use.
6. Maintenance: Performing updates and resolving issues after deployment.



## **Scheduling of the Project According to Gantt Search**

