ONLINE SHOPPING CART SYSTEM

A PROJECT REPORT

Submitted by

SAKTHI SWATHI M	**2303811710422135**
SHAJATHI BEE M	**2303811710422141**
SHAMIRTHA J	**2303811710422142**
SHARMILY S	**2303811710422144**

in partial fulfillment of the requirements for the award degree of

Bachelor in Engineering

CSB1303 – OBJECT ORIENTED ANALYSIS AND DESIGN

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

K. RAMAKRISHNAN COLLEGE OF TECHNOLOGY

(AUTONOMOUS)

SAMAYAPURAM – 621112

DECEMBER 2025

K. RAMAKRISHNAN COLLEGE OF TECHNOLOGY

(AUTONOMOUS)

SAMAYAPURAM - 621112

BONAFIDE CERTIFICATE

The work embodied in the present project report entitled **"ONLINE SHOPPING CART SYSTEM"** has been carried out by the students **SAKTHI SWATHI M, SHAJATHI BEE M, SHAMIRTHA J, SHARMILY S.** The work reported herein is original and we declare that the project is our own work, except where specifically acknowledged, and has not been copied from other sources or been previously submitted for assessment.

Date of Viva Voce: ……

Mrs. V. KALPANA M.E., (Ph.D.,)

SUPERVISOR

Assistant Professor

Department of CSE

K. Ramakrishnan College of Technology (Autonomous)

Samayapuram – 621 112

Mr. R. RAJAVARMAN M.E., (Ph.D.,)

HEAD OF THE DEPARTMENT

Assistant Professor (Sr. Grade)

Department Of CSE

K. Ramakrishnan College of Technology (Autonomous)

Samayapuram – 621 112.

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ABSTRACT

The **Online Shopping Cart System** is an object-oriented e-commerce application developed using Java to provide an efficient and user-friendly online shopping platform. It allows customers to browse products, view detailed descriptions, compare items, add products to a cart, and place orders without visiting a physical store. The system integrates a database using SQL to maintain real-time product availability, user information, and order details. It automates traditional manual processes such as billing, stock checking, and order tracking, thereby reducing human effort and errors. The project follows Object-Oriented Analysis and Design (OOAD) principles such as classes, objects, inheritance, and encapsulation, which makes the system modular, scalable, and easy to maintain. Secure user authentication ensures the protection of customer data and transaction information. Admin modules support product management, category updates, and stock monitoring. Overall, the system provides a smooth, secure, and structured shopping experience and demonstrates the effectiveness of Java and OOAD concepts in building modern e-commerce applications.

KEY WORDS:

Online Shopping Cart System, E-commerce Application, Java, Object-Oriented Analysis and Design (OOAD), SQL Database, User Authentication, Product Management, Shopping Cart, Order Processing, Stock Management, Secure Transactions.

ACKNOWLEDGEMENT

We thank our **Dr. N. Vasudevan**, Principal, for his valuable suggestions and support during the course of our research work.

We thank our **Mr. R. Rajavarman,** Head of the Department, Assistant Professor (Sr. Grade), Department of Computer Science and Engineering, for his valuable suggestions and support during the course of our research work.

We wish to record our deep sense of gratitude and profound thanks to our Guide **Mrs. V. Kalpana,** Assistant Professor, Department of Computer Science and Engineering, for her keen interest, inspiring guidance, constant encouragement with our work during all stages, to bring this thesis into fruition.

We are extremely indebted to our project coordinator **Mrs. V. Kalpana,** Department of Computer Science and Engineering, for her valuable suggestions and support during the course of our research work.

We also thank the faculty and non-teaching staff members of the Computer Science and Engineering, K. Ramakrishnan College of Technology, Samayapuram, for their valuable support throughout the course of our research work.

Finally, we thank our parents, friends and our well wishes for their kind support.

SIGNATURE

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

ABBREVIATIONS		DEFINITION
SQL	-	Structured Query Language
GB	-	Gigabyte
RAM	-	Random Access Memory
HTML	-	Hyper Text Markup Language
OS	-	Operating System
UI	-	User Interface
CSS	-	Cascading Style Sheets
HTTPS	-	Hyper Text Transfer Protocol Secure
JDK	-	Java Development Kit

CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION TO DOMAIN

The Online Shopping Cart System is a modern platform that transforms traditional shopping into a convenient online experience using object-oriented concepts. It allows users to browse products, compare prices, view details, and purchase items easily from home. Built using Java for backend logic and SQL for database management, the system ensures accurate product information and smooth operations. Online shopping has become essential today because it saves time, reduces manual work, and offers 24/7 accessibility. Customers can add items to their cart, modify them, and proceed to secure checkout. Administrators can manage products, stock, and orders through a structured interface. OOAD principles and diagrams like use-case and class diagrams help maintain clarity in system design. This introduction highlights the importance of a well-organized structure for future updates and scalability. It also explains how secure login, smooth navigation, and reliable data handling make the system efficient. Overall, it prepares the reader to understand the methodology and modules used in developing this e-commerce application.

1.2 PROBLEM DESCRIPTION

In traditional shopping, customers must visit physical stores to search for products, compare prices, and make purchases, which is often time-consuming and inconvenient. Store owners also struggle with manual stock management, billing errors, and difficulty tracking customer orders. As online shopping grows, there is a need for a system that automates these tasks and provides a simple, fast, and secure way to buy and manage products digitally. The problem is to develop an online shopping cart system that allows users to browse items, add them to a cart, and place orders easily, while giving administrators efficient control over products, inventory, and sales records.

1.3 OBJECTIVE OF THE PROJECT

The main objective of the Online Shopping Cart System is to provide users with a smooth, convenient, and secure online shopping experience. It automates manual activities like browsing products, adding items to the cart, checking stock, billing, and tracking orders. The system reduces workload for both customers and administrators by offering real-time product availability, accurate order updates, and secure login features. Using OOAD principles and Java, the project ensures scalability, flexibility, and easy future enhancements. It also aims to help administrators manage products, stock, and pricing efficiently through a structured backend database. By simplifying checkout, improving data accuracy, and supporting multiple users simultaneously, the system enhances overall shopping efficiency. Ultimately, the objective is to build a reliable and user-friendly e-commerce platform that demonstrates the effective use of OOAD and Java technologies.

1.4 SCOPE OF THE PROJECT

The scope of the Online Shopping Cart System covers the development of a complete online shopping platform that allows customers to browse products, view detailed descriptions, compare items, add them to a cart, and place orders securely. It includes real-time inventory management, where product availability is updated instantly based on user actions. The admin module is designed to manage product categories, pricing, stock levels, and order records efficiently. The system also ensures secure user authentication, smooth navigation, and an intuitive interface so users can shop easily. It supports communication between the frontend and the database to keep all information accurate and up to date. The scope further extends to automating key processes such as billing, order tracking, and stock updates, reducing manual effort. Designed with OOAD principles, the project allows easy scalability for future features like online payments, discount management, product reviews, and personalized recommendations. Overall, the scope includes delivering a reliable, scalable, and user-friendly e-commerce solution.

CHAPTER 2

SYSTEM REQUIREMENT SPECIFICATION (SRS)

2.1 FUNCTIONAL REQUIREMENTS

2.1.1 USER ACCOUNT MANAGEMENT

The system must allow new users to register with their basic details and create secure accounts. Registered users should be able to log in using validated credentials. Passwords and input fields must be checked to avoid errors or unauthorized access. Users should have access to their personal profiles and account settings. The system ensures only verified users can use customer features.

2.1.2 PRODUCT BROWSING AND SEARCH

Users must be able to view all available products with details such as name, price, image, and description. Products should be organized into categories for easy browsing. The system must allow keyword-based product search. Filtering options like price, category, and popularity must be available. All product information must be fetched accurately from the database.

2.1.3 SHOPPING CART AND ORDER PROCESSING

The system must allow users to add products to their cart, update quantities, and remove items easily. The cart must calculate the total price automatically. Users should be able to review cart details before checkout. The system verifies items and user details during order placement. A confirmed order should be recorded in the database.

2.1.4 SECURE PAYMENT AND ORDER TRACKING

Users must be able to make secure payments through available payment methods. The system validates payment details before processing transactions. After payment, the system must confirm the order status. Users should be able to track their orders, including Processing, Shipped, and Delivered. Order history must be accessible to all registered users.

2.1.5 ADMIN PRODUCT AND ORDER MANAGEMENT

Only authorized admins should be able to log in through secure admin authentication. Admins can add, edit, or delete product details at any time. Product updates must reflect immediately in the system. The system must maintain accurate and complete order records for admin monitoring.

2.2 NON-FUNCTIONAL REQUIREMENTS

2.2.1 PERFORMANCE REQUIREMENTS

The system must load product pages quickly and handle multiple users at the same time without slowing down. Search and filter responses should be fast and accurate. The database should retrieve product and order information efficiently. Cart updates must happen instantly when users add or remove items. Overall system performance should support smooth and uninterrupted shopping.

2.2.2 SECURITY REQUIREMENTS

All user and admin logins must be protected using secure authentication. Passwords should be stored in encrypted form. Payment details must be processed safely without exposing sensitive data. Unauthorized access to admin pages should be strictly blocked. The system must follow standard security practices to prevent data breaches.

2.2.3 USABILITY REQUIREMENTS

The user interface should be simple, clear, and easy for all users to understand. Navigation between pages must be smooth and consistent. Buttons, text, and product layouts should be readable and user-friendly. Error messages should be clear and helpful. The system should provide a pleasant shopping experience for both new and experienced users.

2.2.4 RELIABILITY REQUIREMENTS

The system must work consistently without frequent crashes or failures. Order and payment processes should always produce correct results. Data stored in the database must remain accurate and consistent. The system should recover smoothly from minor failures. Availability of the platform should be high to support continuous use.

2.2.5 SCALABILITY REQUIREMENTS

The system must support future growth in the number of users, products, and orders. It should be easy to add new modules or features without major changes. Increased traffic should not affect the system's performance. The database must handle larger amounts of data smoothly. The architecture should allow upgrades when needed.

2.3 HARDWARE REQUIREMENTS

2.3.1 SERVER HARDWARE

The server requires at least an Intel i3 or higher processor to ensure smooth system operations. A minimum of 8 GB RAM is necessary to handle multiple user requests efficiently. Around 250 GB of storage space is needed to store product images, user data, and order records. A stable network card should be available for uninterrupted online connectivity. The server must support scalability for future upgrades.

2.3.2 CLIENT SYSTEM

End users should have a basic laptop or desktop with at least 4 GB RAM for smooth browsing. A dual-core processor is sufficient to run the online shopping system without lag. The client device should support modern web browsers such as Chrome, Firefox, or Edge. A display with standard resolution is enough to view product catalogs clearly. Reliable internet access is essential for uninterrupted shopping activity.

2.3.3 MOBILE DEVICE

The system should be accessible on smartphones running Android or iOS. A device with at least 2 GB RAM is recommended for proper page loading. The phone must have an updated browser that supports HTML5. A stable mobile internet connection (4G or above) is required for quick responses. The display should be capable of showing content legibly in mobile-friendly layouts.

2.3.4 NETWORK REQUIREMENTS

A high-speed internet connection is necessary for both users and admins to access the system. The network must support stable data transfer with minimal interruptions. Bandwidth should be sufficient to handle multiple users simultaneously. Low latency is important for fast loading of product pages and transactions. Secure network protocols must be supported to protect sensitive data.

2.3.5 NETWORK REQUIREMENTS

An external storage system is needed to maintain periodic backups of database files. This backup unit should have enough capacity to store multiple versions of user, product, and order data. The backup device must support fast read/write operations for quick recovery. Regular backup schedules should be maintained to avoid data loss. The system should allow restoring data efficiently in case of failure.

2.4 SOFTWARE REQUIREMENTS

2.4.1. OPERATING SYSTEM

The system can run on Windows, Linux, or macOS operating systems. The server is recommended to use a stable OS like Windows Server or Ubuntu. The client side works smoothly on any modern OS used by customers. The operating system must support modern browsers and Java applications. Regular updates should be enabled for better security and performance.

2.4.2 PROGRAMMING LANGUAGES

Java is required for developing the backend logic of the Online Shopping Cart System. It provides platform independence and strong object-oriented features. HTML, CSS, and JavaScript are used for designing the frontend interface. These languages ensure a clean, responsive, and user-friendly UI. The combination of Java and web technologies supports a smooth and interactive system.

2.4.3 DATABASE SOFTWARE

MySQL or any relational database system is needed to store user, product, and order data. The database must support SQL queries for efficient data management. Features like indexing, relationships, and constraints ensure data accuracy. The server should allow secure access to the database with proper credentials. Backup and restore features must be available for data safety.

2.4.4 WEB BROWSER

Clients must use modern browsers like Google Chrome, Mozilla Firefox, Microsoft Edge, or Safari. These browsers support HTML5, CSS3, and JavaScript, ensuring proper UI display. A browser with fast rendering helps load product pages

quickly. Security features like HTTPS support are necessary for safe transactions. Regular browser updates improve compatibility and system performance.

2.4.5 ADDITIONAL SOFTWARE TOOLS

Java Development Kit (JDK) is required for compiling and running Java applications. A web server tool like Apache Tomcat is needed to deploy the project. Database management tools like phpMyAdmin or MySQL Workbench help manage data easily. A code editor such as Eclipse, IntelliJ IDEA, or VS Code can be used for development. Antivirus software is recommended for safe and uninterrupted system usage.

2.5 USER CHARACTERTICS

The users of the Online Shopping Cart System include general customers and administrators with basic computer and internet skills. Customers are expected to know simple web navigation such as browsing, searching, and adding items to the cart. They do not require any technical expertise to use the system. Administrators, however, should have a moderate understanding of product management, stock control, and order handling. Overall, the system is designed to be user-friendly so that both beginners and experienced users can operate it easily.

2.6 CONSTRAINTS

The Online Shopping Cart System must operate within certain constraints, such as requiring a stable internet connection for users to access the platform. The system depends on browser compatibility, so only modern browsers are supported. Hardware limitations like low device memory or slow processors may affect performance. The system must also follow security constraints, ensuring encrypted data storage and safe payment processing. Additionally, all modules must function within the capabilities of the selected database and server environment.

CHAPTER 3

ANALYSIS AND DESIGN

3.1 USE CASE DIAGRAM

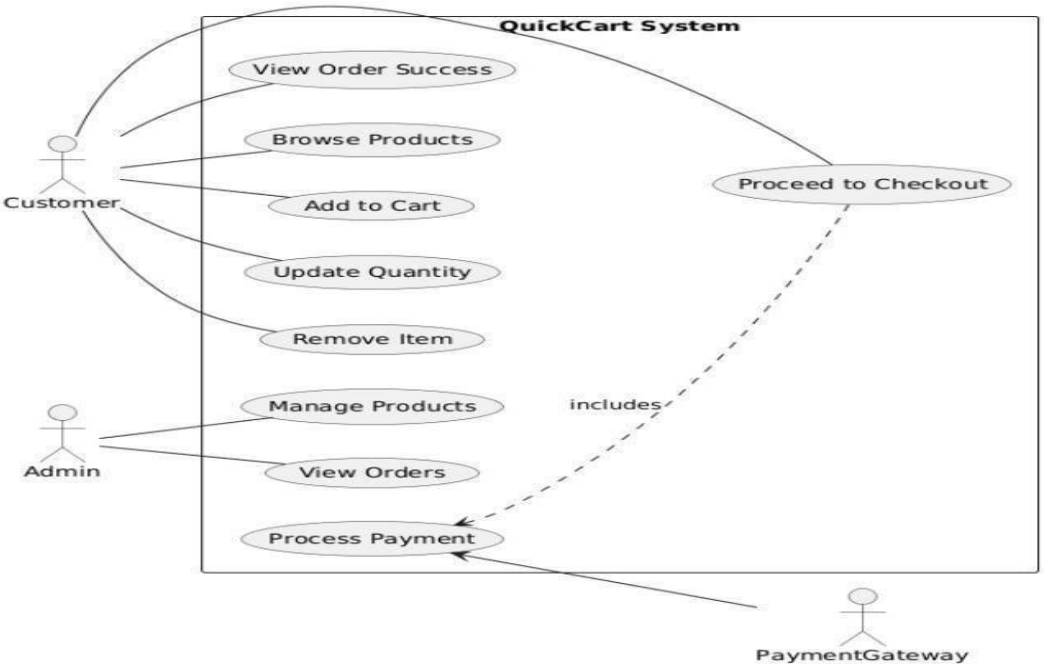


Figure 3.1 USE CASE DIAGRAM

3.1.1 USE CASE DIAGRAM DESCRIPTION

The Use Case diagram of the Quick Cart system represents the interaction between users and the system's core functionalities. The primary actors include the Customer, the Admin, and an external Payment Gateway. The Customer performs activities such as browsing products, adding items to the cart, viewing the cart, updating or removing items, proceeding to checkout, and making payments. The Admin controls product management activities like adding, editing, and maintaining product details. The Payment Gateway acts as an external system responsible for processing online payments. Overall, the Use Case diagram visually explains how users connect with the system and what operations are available to them.

3.2 ACTIVITY DIAGRAM

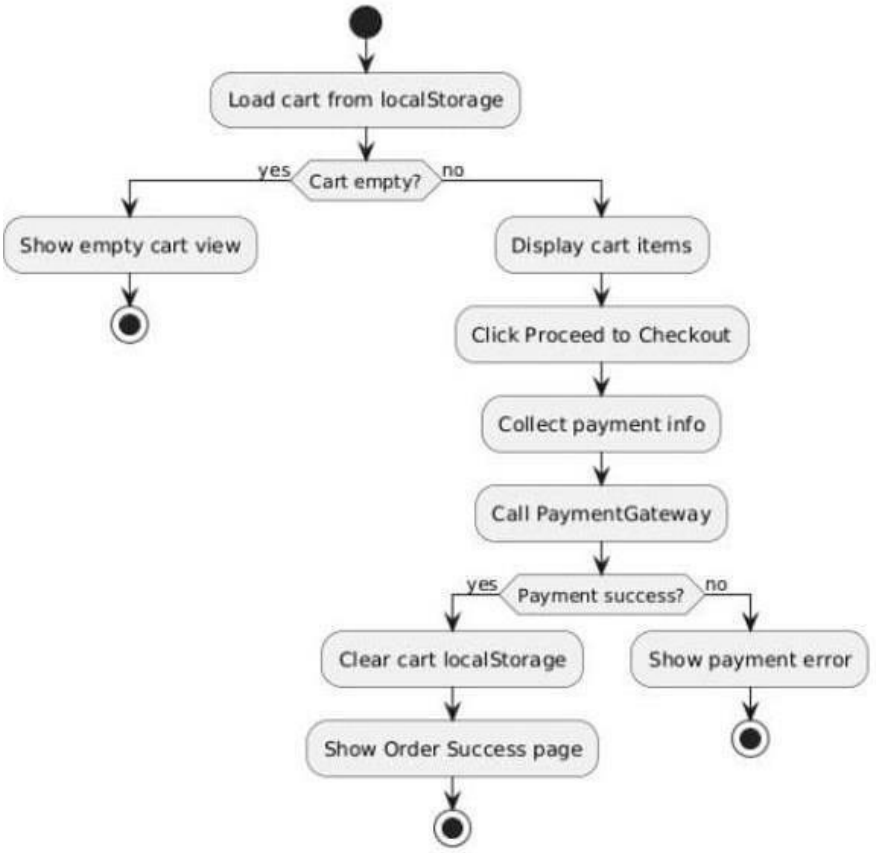


Figure 3.2 ACTIVITY DIAGRAM

3.2.1 ACTIVITY DIAGRAM DESCRIPTION

The Activity Diagram describes the workflow involved in adding products to the cart. The process begins when the system displays available items to the user, after which they choose a product to add. The system checks if the selected item already exists in the cart. If it exists, the quantity is incremented; otherwise, a new item entry is created. The updated cart is then saved to local storage, and the system reflects the new cart count to the user. This diagram conveys the flow of decision making and the continuous movement of activities until the operation completes successfully.

3.3 CLASS DIAGRAM

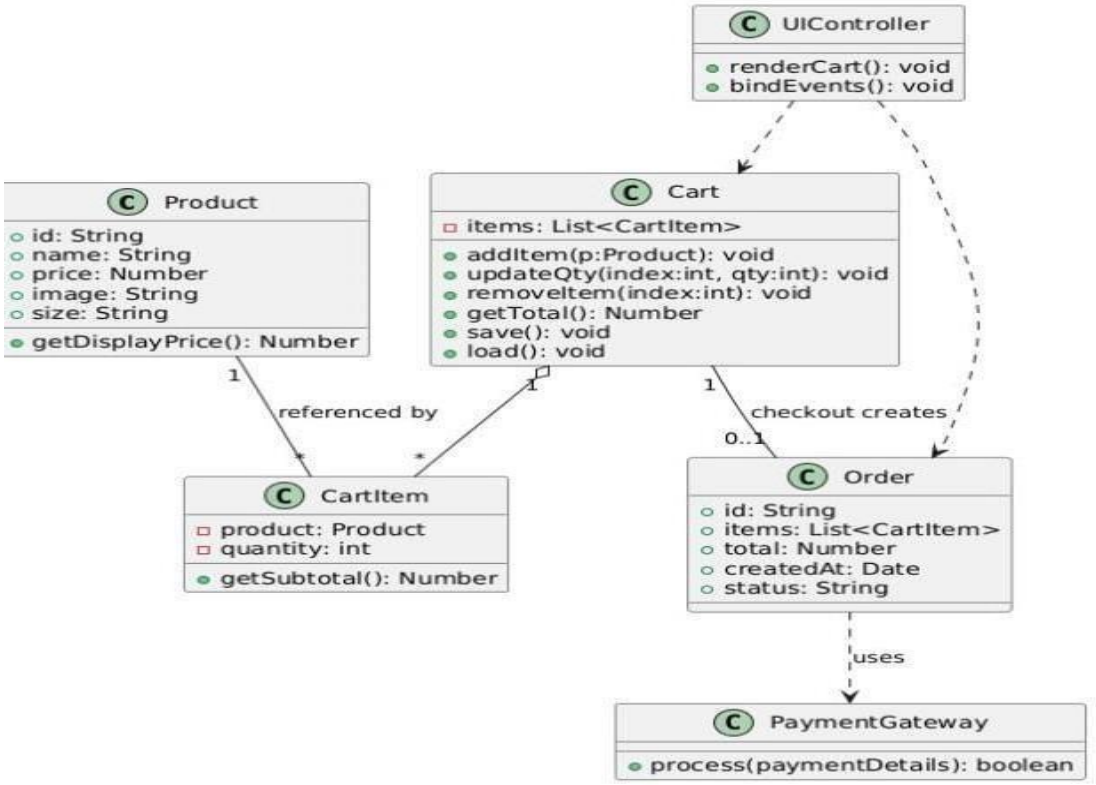


Figure 3.3 CLASS DIAGRAM

3.3.1 CLASS DIAGRAM DESCRIPTION

The Class Diagram for Quick Cart explains the structural design of the system by representing classes, attributes, and methods. The core classes include Item, Cart Manager, Local Storage Handler, Home Page, and Cart Page. The Item class stores product details such as name, price, image, and quantity, while Cart Manager acts as the central logic controller responsible for adding items, removing items, updating quantities, and calculating the total price. Local Storage Handler manages data storage within the browser, ensuring cart contents persist across sessions. Meanwhile, Home Page and Cart Page represent the UI where the customer interacts with the system. This diagram reflects a modular and maintainable design with high cohesion and low coupling.

3.4 SEQUENCE DIAGRAM

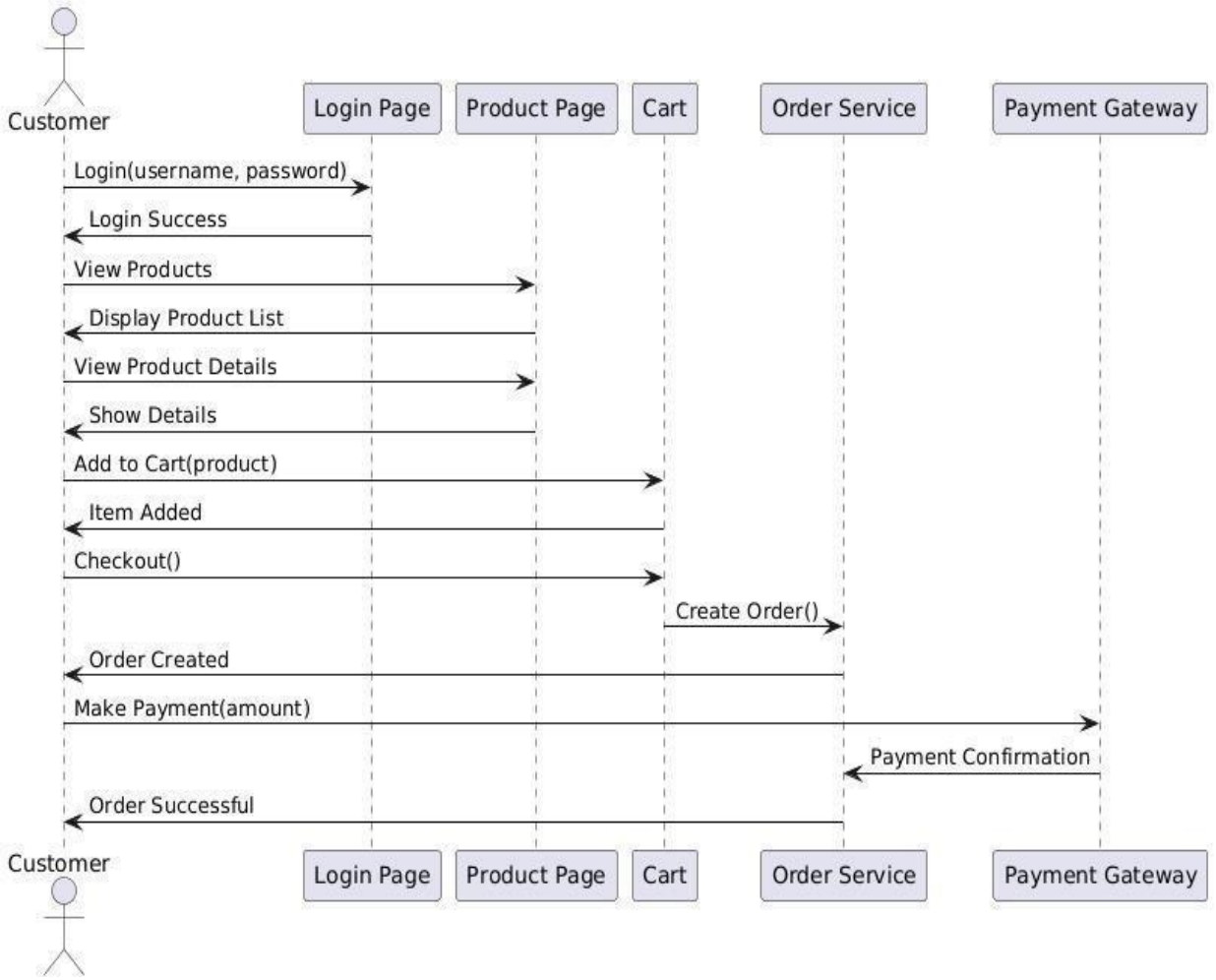


Figure 3.4 SEQUENCE DIAGRAM

3.4.1 SEQUENCE DIAGRAM DESCRIPTION

The Sequence Diagram illustrates the step-by-step interaction flow when an item is added to the cart and viewed later. When a customer presses the "Add to Cart" button, the request first reaches the Cart Manager, which either adds a new item or increases the quantity of an existing one. The updated cart data is then saved using Local Storage Handler. When the user opens the Cart Page, it requests stored data through the Cart Manager and displays the list of items back to the user. This diagram clearly highlights the chronological communication between UI components, business logic, and storage modules.

3.5 STATE MACHINE DIAGRAM

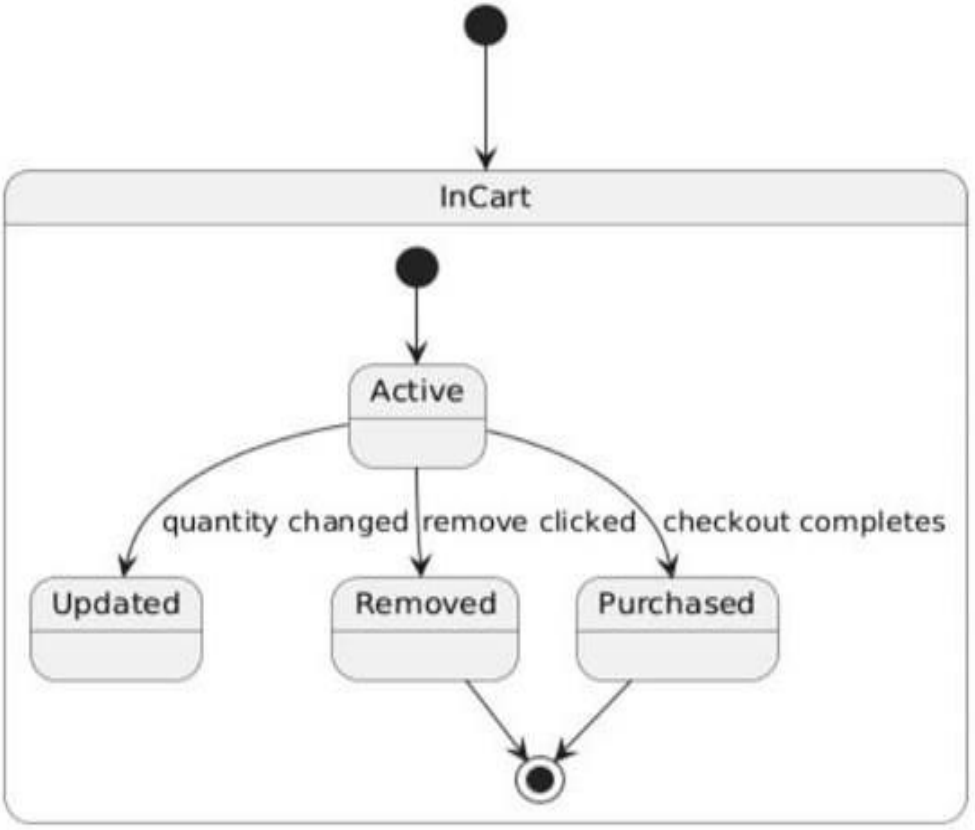


Figure 3.5 STATE MACHINE DIAGRAM

3.5.1 STATE MACHINE DIAGRAM DESCRIPTION

The State Machine Diagram represents the different states through which a cart item transitions during the shopping process. An item first enters the system when it is added to the cart, and may move into the 'Quantity Updated' state if the user increases or decreases the number of units. If the user removes the item, it transitions into the 'Removed' state. Finally, when checkout is completed, cart items enter a 'Purchased' or 'Order Completed' state and are cleared from the cart. This diagram helps visualize how the status of an item changes based on user actions.

3.6 COMPONENT DIAGRAM

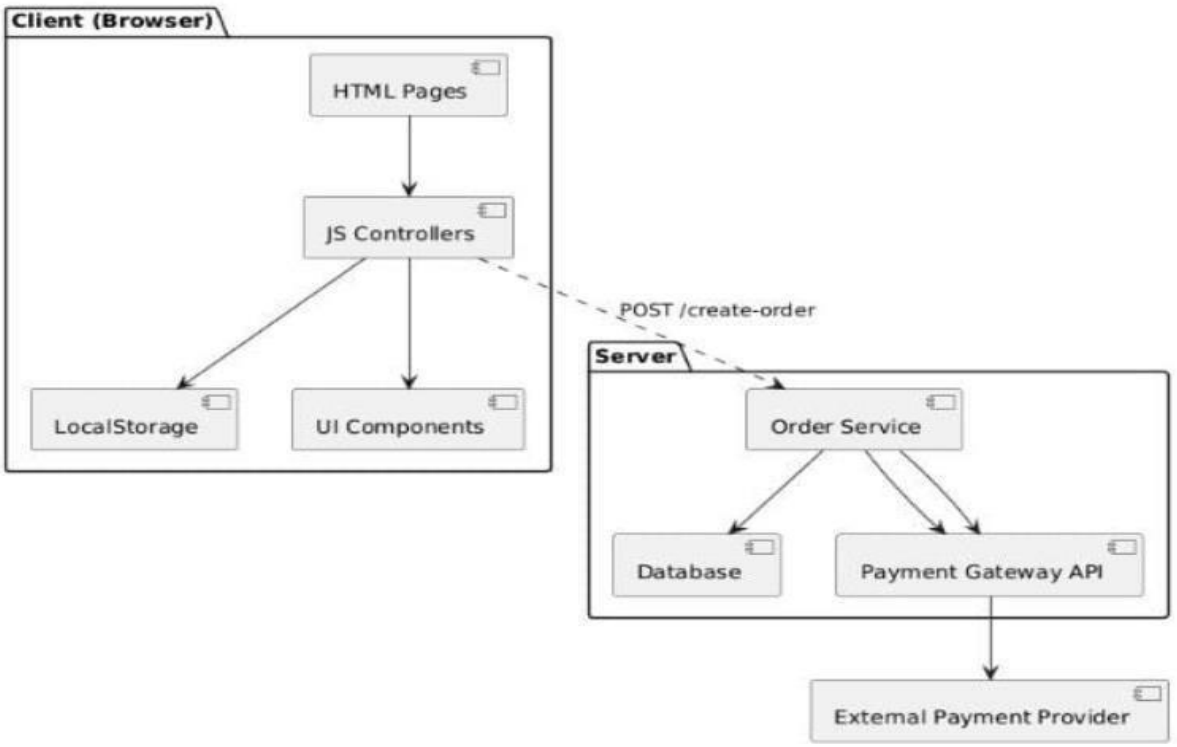


Figure 3.6 COMPONENT DIAGRAM

3.6.1 COMPONENT DIAGRAM DESCRIPTION

The Component Diagram focuses on how different modules of Quick Cart are structured and interact. The system is divided into three major components: the UI component, the Business Logic component, and the Storage component. The UI contains pages like Home Page and Cart Page where the user performs actions. The Business Logic component implements the core functionality through Cart Manager and pricing strategies. The Storage component handles saving and retrieving cart data using Local Storage Handler. The connection of UI → Logic → Storage demonstrates a clean architecture that is easy to scale and maintain.

3.7 DEPLOYMENT DIAGRAM

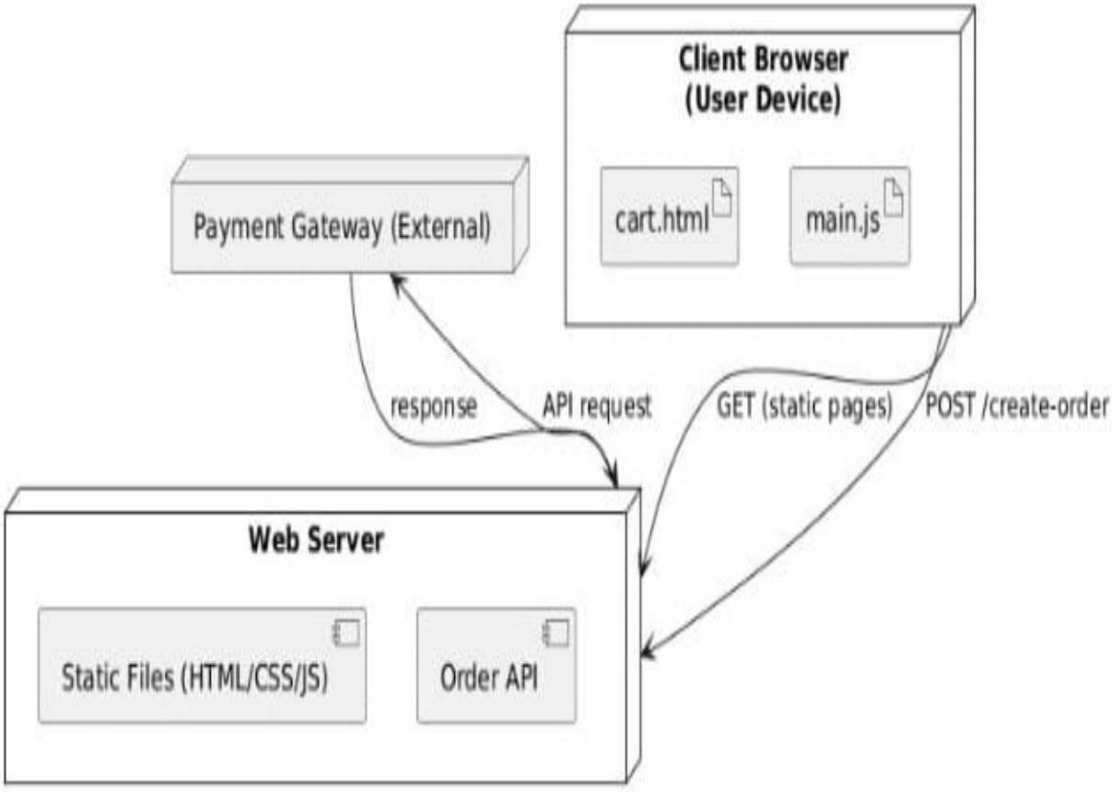


Figure 3.7 DEPLOYMENT DIAGRAM

3.7.1 DEPLOYMENT DIAGRAM DESCRIPTION

The Deployment Diagram represents how the system operates in a real runtime environment. The Quick Cart application runs entirely inside a web browser, where HTML, CSS, and JavaScript files are executed. Local Storage within the browser serves as the storage environment for cart data. Optionally, an admin backend or product database can run on a cloud or local server for managing inventory. A third-party Payment Gateway server handles online transactions during checkout. This diagram depicts how the system is physically deployed and accessed by users on various devices.

3.8 PACKAGE DIAGRAM

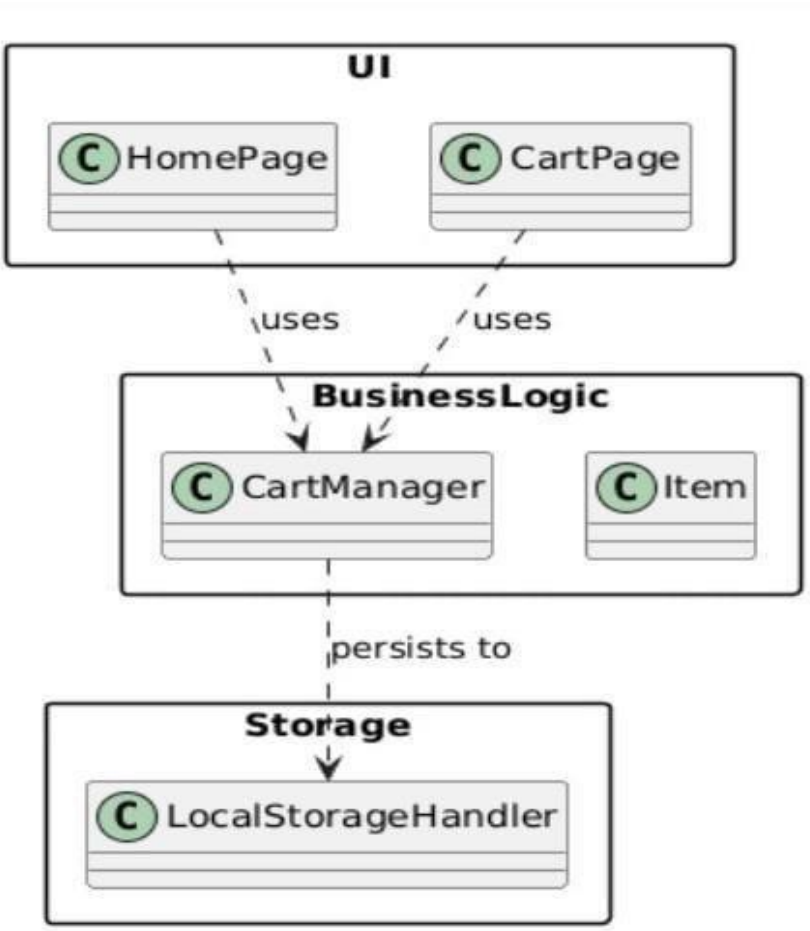


Figure 3.8 PACKAGE DIAGRAM

3.8 PACKAGE DIAGRAM DESCRIPTION

The Package Diagram groups system elements into logical modules to simplify understanding and maintenance. The Quick Cart system contains three primary packages: the UI package, the Business Logic package, and the Storage package. The UI package includes Home Page and Cart Page, responsible for displaying products and enabling user interactions. The Business Logic package contains Cart Manager and Item classes which process cart operations and pricing. The Storage package includes Local Storage Handler which stores cart data persistently. The dependency flow UI → Business Logic → Storage ensures separation of concerns, resulting in a scalable and well-organized architecture.

3.9 DESIGN PATTERNS USED

The system applies multiple design patterns and GRASP principles to ensure a well-structured and maintainable design. The Controller pattern is implemented through the *Cart Manager*, which acts as an intermediary between the UI layer and the business logic, handling all cart-related user actions such as adding, updating, and removing items. The Information Expert pattern is used by the *Item* class, as it holds essential data like price and quantity and is therefore responsible for providing accurate information for calculations. Low Coupling is achieved by clearly separating the UI, business logic, and storage layers, allowing each layer to evolve independently without affecting others. High Cohesion is maintained by assigning each class a single, well-defined responsibility, such as managing cart operations or handling data persistence. In addition, the Layered Architecture pattern organizes the system into distinct layers, improving readability, scalability, and ease of maintenance.

CHAPTER 4

IMPLEMENTATION

4.1 MODULE DESCRIPTION

The Online Shopping Cart System is designed using a modular approach to ensure smooth operation and easy management. It consists of several interconnected modules, each responsible for a specific function within the system. The User Module manages customer registration, secure login, profile updates, product browsing, cart handling, and access to order history, providing a personalized shopping experience. The Admin Module controls backend operations such as product management, category maintenance, stock updates, and order monitoring to keep the system organized and up to date. The Product Catalog Module displays available products with detailed information and supports searching and filtering for easy navigation. The Shopping Cart Module allows users to add, modify, and remove items before checkout while calculating the total cost. The Payment Module handles secure transaction processing, validates payment details, and updates order status after successful payment. The Order Management Module manages order creation, tracking, status updates, and maintains order history for both users and administrators. Together, these modules work in coordination to deliver a secure, efficient, and user-friendly online shopping experience.

4.1.1 USER MODULE

The User Module manages all customer-related activities in the Online Shopping Cart System. It begins with user registration, allowing customers to create an account using their personal details. After registration, users can log in securely with their credentials. The module verifies and validates login information to prevent unauthorized access. It allows customers to update their profile information whenever needed. Users can browse products, view details, and add items to their shopping cart. The module keeps track of the user session to maintain cart items until checkout. It also provides access to order history for reviewing past purchases. SQL stores and retrieves all user information securely.

4.1.2 ADMIN MODULE

The Admin Module handles all backend management activities of the Online Shopping Cart System. It provides a secure login for administrators to access the system. Admins can add, update, and delete products along with their details. The module allows managing product categories and stock levels. Admins can also view and monitor all customer orders. Order status can be updated to processing, shipped, or delivered. The module ensures all product changes reflect immediately in the system. It helps admins maintain a clean and organized catalog for users. SQL supports storing all administrative actions. Overall, the Admin Module keeps the system updated and well-managed.

4.1.3 PRODUCT CATALOG MODULE

The Product Catalog Module displays all available products in an organized manner. It lists items with details like name, description, price, and image. Users can browse products easily through categories. The module allows filtering and searching for specific products. It retrieves product data from the database using SQL queries. Only available products are shown to ensure accuracy. The module connects with the Admin Module for real-time product updates. It also links to the Shopping Cart Module when a user selects an item. The catalog is designed to be attractive and user-friendly. Overall, it serves as the main interface for product viewing.

4.1.4 SHOPPING CART MODULE

The Shopping Cart Module allows users to temporarily store items before purchasing. Customers can add products to the cart with one click. The module calculates the total cost based on item quantity and price. Users can update or remove items anytime. It keeps cart data active throughout the session. The module communicates with the Product Catalog to verify stock levels. SQL is used to save cart data for logged-in users. It provides a clear view of selected items before checkout. After payment, the cart is automatically cleared. Overall, the Shopping Cart Module helps manage items efficiently before purchase.

4.1.5 PAYMENT MODULE

The Payment Module handles all transaction-related processes in the system. It verifies the total bill before initiating payment. The module validates user-entered payment details securely. It interacts with a payment gateway for authorization. After successful payment, it updates the order status. The module records each transaction in the database. It prevents duplicate payments through validation checks. Error messages are displayed for failed transactions. A digital receipt is generated after payment confirmation. Overall, the Payment Module ensures secure and accurate purchase processing.

4.1.6 ORDER MANAGEMENT MODULE

The Order Management Module handles all customer order processing in the system. After payment, a unique order ID is generated for each purchase. The module maintains order details such as items and total cost. Customers can view their order status through their account. Admins can update the order status to shipped or delivered. The module updates product stock once an order is placed. SQL stores all order records securely for future reference. It supports viewing complete order history for both users and admins. The module ensures accurate processing of every order. Overall, the Order Management Module manages all order-related activities efficiently.

4.2 TECHNOLOGY DESCRIPTION

The Online Shopping Cart System is built using Java for backend processing due to its security and object-oriented features. The frontend uses HTML, CSS, and JavaScript to provide a responsive and user-friendly interface. MySQL is used to store and manage user, product, cart, and order data efficiently. The system is deployed using a web server such as Apache Tomcat, with security ensured through input validation, password hashing, and session management. Overall, this technology stack offers a secure, stable, and scalable e-commerce solution.

CHAPTER 5

TESTING

5.1 TESTING STRATEGY

The testing strategy for the Online Shopping Cart System focuses on ensuring that all modules function correctly, securely, and efficiently before deployment. The process begins with unit testing, where each individual component—such as user login, product display, shopping cart, and payment functions—is tested separately to identify internal errors. Integration testing is then performed to ensure that these modules work together smoothly, especially in areas where data flows between components like cart to payment or admin updates to product catalog. System testing evaluates the entire application in a real-time environment to verify performance, security, usability, and reliability under normal and high user loads. Finally, user acceptance testing is conducted to confirm that the system meets user expectations and functional requirements. This multi-level testing strategy ensures that the system is stable, error-free, and ready for actual use.

5.2 SAMPLE TEST CASES

- **Test Case 1 – User Login Validation**
  Objective: Verify that only valid users can log into the system.
  Input: Username = “user01”, Password = “user123”
  Expected Output: Redirect to user homepage.
  Actual Result: Successful login.
  Status: Pass
- **Test Case 2 – Add Product to Cart**
  Objective: Verify that the user can add a selected product to the shopping cart.
  Input: Click “Add to Cart” for Product ID = 101
  Expected Output: Product added to cart and cart count increases.
  Actual Result: Product successfully added.
  Status: Pass

- **Test Case 3 – Product Search Functionality**
  Objective: Verify that the search bar returns correct matching products.
  Input: Search keyword = “Shirt”
  Expected Output: Display list of products containing “Shirt”.
  Actual Result: Correct products displayed.
  Status: Pass
- **Test Case 4 – Payment Processing Validation**
  Objective: Check if payment is processed successfully with valid card details.
  Input: Card No = “1234 5678 9012 3456”, CVV = “123”, Amount = ₹1500
  Expected Output: Payment successful and order ID generated.
  Actual Result: Payment completed and order generated.
  Status: Pass
- **Test Case 5 – Admin Adding New Product**
  Objective: Verify that an admin can add a new product into the system.
  Input: Product Name = “Saree”, Price = “₹899”, Category = “Dress”
  Expected Output: Product added and visible in product list.
  Actual Result: Product successfully added.
  Status: Pass

5.3 TEST RESULTS

The testing of the Online Shopping Cart System produced successful results across all major modules, confirming that the system performs reliably and meets the required specifications. User login and registration functions worked correctly, allowing only valid users to access the platform. Product browsing, search, and filtering features responded quickly and displayed accurate results. The shopping cart module performed as expected, updating item quantities and calculating totals without errors. Payment processing was validated successfully with correct transaction handling and order generation. Admin operations such as adding products, updating stock, and managing orders also functioned smoothly. No critical defects were found during system testing, and minor issues detected were fixed immediately. Overall, the test results indicate that the system is stable, user- friendly, and ready for deployment.

CHAPTER 6

CONCLUSION AND FUTURE ENHANCEMENT

6.1 CONCLUSION

The Online Shopping Cart System effectively demonstrates how Object-Oriented Analysis and Design principles can be applied to build a reliable and efficient e-commerce platform. It provides customers with a smooth, secure, and convenient way to browse products, manage their cart, and make purchases anytime and anywhere. The system overcomes the limitations of traditional shopping by automating tasks such as billing, stock updating, and order tracking, thereby reducing manual errors. With Java ensuring stability and scalability and an SQL database maintaining accurate, structured data, the system offers strong backend support and secure user authentication. Each module—such as Admin Management, Product Catalog, Shopping Cart, Payment, and Order Management— works together to deliver a seamless shopping experience.

The project also highlights how automation improves business efficiency and reduces workload for both customers and administrators. Thanks to OOAD principles, the system is easy to maintain and expand in the future. Overall, the project meets its goals of providing a secure, user- friendly, and well-organized online shopping environment while proving the effectiveness of OOAD in modern e-commerce development.

6.2 FUTURE SCOPE

The Online Shopping Cart System has wide potential for future upgrades that can improve performance and user experience. The system can integrate multiple secure payment gateways, support product reviews, ratings, discounts, and offer personalized product recommendations using AI. Features like wishlist, faster checkout, multi-language support, and social media login can enhance accessibility. Admins can benefit from advanced analytics, automatic stock alerts, and improved inventory management. The platform can also include vendor- based selling, return/refund modules, and automated shipment updates. Overall, these enhancements can transform the system into a more secure, scalable, and user-friendly e-commerce solution.

APPENDIX A
SOURCE CODE

Cart.html :

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1.0"/>
  <title>QuickCart</title>
  <link rel="icon" type="image/phg" href="placeholder/icon.png">
  <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css"
rel="stylesheet">
  <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/bootstrap.bundle.min.js
"></script>
  <link rel="stylesheet" href="css/styles.css" />
  <!-- for font -->
  <link rel="preconnect" href="https://fonts.googleapis.com">
  <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
  <link
 href="https://fonts.googleapis.com/css2?family=Poppins:ital,wght@0,100;0,
200;0,300;0,400;0,500;0,600;0,700;0,800;0,900;1,100;1,200;1,300;1,400;1,500;1
,600;1,700;1,800;1,900&display=swap"
    rel="stylesheet">
    <!-- foricon -->
  <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap-
icons@1.10.5/font/bootstrap-icons.css">
  <!-- for font -->
  <link rel="preconnect" href="https://fonts.googleapis.com">
 <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
 <link
```

```
href="https://fonts.googleapis.com/css2?family=Raleway:ital,wght@0,100..900;
1,100..900&display=swap" rel="stylesheet">
    <style>
     #divs {
      position: relative;
      background-image: url("placeholder/images1.png");
      background-repeat: no-repeat;
      background-size: cover;
      background-position: center;
      height: 80vh;
      opacity: 0;
      transform: translateY(-100px); /* start above view */
      transition: opacity 1.5s ease, transform 1.5s ease;
    }
    #divs.visible {
      opacity: 1;
      transform: translateY(0); /* slide down into view */
    }
    .fixed-btn {
      position: absolute;
      bottom: 120px;
      left: 30%;
      transform: translateX(-50%);
      padding: 14px 30px;
      font-size: 18px;
      font-weight: bold;
      background: linear-gradient(90deg, #00c6ff, #0072ff);
      color: white;
      border: none;
      border-radius: 10px;
      text-align: center;
      text-decoration: none;
```

```
  z-index: 1;
  box-shadow: 0 0 10px #00c6ff;
  transition: 0.3s;
}
.fixed-btn:hover {
  transform: translateX(-50%) scale(1.05);
}
    body {
      background: #a1d7eb;
      font-family: 'Poppins', sans-serif;
      padding-top: 70px;
    }
    h2 {
      margin-bottom: 30px;
    }
    table {
      width: 100%;
      background: white;
      border-radius: 4px;
      overflow: hidden;
      box-shadow: 0 0 8px rgb(0 0 0 / 0.1);
    }
    thead th {
      background-color: #1f1f1f;
      color: white;
      font-weight: 700;
    }
    thead th {
      padding: 12px 15px;
      text-align: left;
    }
    tbody tr {
```

```
  border-bottom: 1px solid #ddd;
}
tbody tr:last-child {
  border-bottom: none;
}
.product-info {
  display: flex;
  align-items: center;
  gap: 15px;
}
.cart-img {
  width: 60px;
  height: 60px;
  object-fit: cover;
  border-radius: 4px;
  border: 1px solid #ddd;
}
.product-details {
  font-size: 14px;
}
.product-name {
  font-weight: 700;
  margin-bottom: 4px;
  color: #222;
}
.product-size {
  font-size: 12px;
  color: #777;
}
.quantity-input {
  width: 50px;
  height: 32px;
```

```
  padding: 5px 8px;
  border: 1px solid #ccc;
  border-radius: 4px;
  font-size: 14px;
  text-align: center;
}
td, th {
  vertical-align: middle !important;
}
.price, .subtotal {
  font-weight: 600;
  font-size: 15px;
  color: #222;
  text-align: right;
}
.remove-btn {
  background-color: #dc3545;
  border: none;
  color: white;
  padding: 6px 14px;
  font-size: 14px;
  border-radius: 4px;
  cursor: pointer;
  transition: background-color 0.2s ease-in-out;
}
.remove-btn:hover {
  background-color: #b02a37;
}
.total-box {
  margin-top: 25px;
  text-align: right;
  font-size: 18px;
```

```
  font-weight: 600;
  color: #2978ff;
}
.checkout-btn {
  background-color: #27784e;
  color: white;
  border: none;
  padding: 10px 22px;
  border-radius: 5px;
  font-size: 16px;
  cursor: pointer;
  float: right;
  margin-top: 10px;
  box-shadow: 0 4px 8px rgb(39 120 78 / 0.4);
  transition: background-color 0.3s ease;
}
.checkout-btn:hover {
  background-color: #1d5c3a;
}
/* Responsive */
@media(max-width: 576px) {
  .product-info {
    flex-direction: column;
    align-items: flex-start;
  }
  .price, .subtotal, .remove-btn {
    text-align: left;
  }
  .total-box, .checkout-btn {
    text-align: center;
    float: none;
  }
```

```
      .checkout-btn {
        width: 100%;
        margin-top: 15px;
      }
    }
    #emptyCart{
      position: relative;
      z-index: 1;
      background: rgba(138, 50, 50, 0.1);
      border-radius: 16px;
      padding: 50px 20px;
      text-align: center;
      color: rgb(207, 55, 55);
      box-shadow: 0 8px 32px rgba(0, 0, 0, 0.4);
      backdrop-filter: blur(10px);
      border: 1px solid rgba(239, 190, 190, 0.2);
      width: 50%;
    }
    .btn {
  background-color: #f35780;
  font-size: 16px;
  color: white;
  border: none;
}
  </style>
</head>
<body>
<nav class="navbar navbar-expand-md fixed-top">
  <div class="container-fluid">

    <!-- Logo -->
    <a class="navbar-brand d-flex align-items-center" href="#">
```

```
          <img src="placeholder/logo.png" alt="Logo" style="height: 40px; width:
50px; margin-right: 5px;">
        </a>
         <h2 style="text-align: justify; font-family: poppins, sans-serif; color:
aliceblue; margin-right: 5px;">QuickCart </h2>
        <!-- Toggler for Mobile -->
        <button class="navbar-toggler navbar-light  " type="button" data-bs-
toggle="collapse" data-bs-target="#mainNavbar">
          <span  class="navbar-toggler-icon"></span>
        </button>
        <!-- Collapsible Navbar Content -->
        <div class="collapse navbar-collapse" id="mainNavbar">
          <!-- LEFT MENU -->
          <!-- CENTER SEARCH BAR -->
      <form class="search-form" onsubmit="searchProducts(event)">
       <div class="search-container">
         <span class="search-icon">
          <svg xmlns="http://www.w3.org/2000/svg" width="16" height="16"
fill="#6c757d" class="bi bi-search" viewBox="0 0 16 16">
            <path d="M11.742 10.344a6.5 6.5 0 1 0-1.397 1.397h-.001l3.85 3.85a1
1 0 0 0 1.415-1.414l-3.85-3.85zm-5.242 1.397a5.5 5.5 0 1 1
                  0-11 5.5 5.5 0 0 1 0 11z"/>
          </svg>
        </span>
        <input id="searchInput" class="form-control search-input" type="search"
placeholder="Search for products, brands and more">
       </div>
      </form>
          <!-- for cart and product -->
          <ul class="navbar-nav nav-underline ms-auto">
            <li class="nav-item "><a class="nav-link"
href="men.html">Men</a></li>
```

```
            <li class="nav-item "><a class="nav-link"
href="women.html">Women</a></li>
            <li class="nav-item "><a class="nav-link"
href="kids.html">Kids</a></li>
           <li class="nav-item active"><a class="nav-link"
href="products.html">Home</a></li>
            <li class="nav-item">
        <a class="nav-link position-relative" href="cart.html">
        Cart
        <span id="cart-count"
            class="position-absolute top-0 start-100 translate-middle badge
rounded-pill bg-danger"
            style="font-size: 0.75rem; padding: 4px 6px;">
          0
        </span>
      </a></li>
      </ul>
        </div>
      </div>
      </nav>
      <br>
      <br>
      <div class="container">
        <h2 class="text-center fw-bold " style="font-family:'Franklin Gothic
Medium', 'Arial Narrow', Arial, sans-serif;">Your Shopping Cart</h2>
       <center>
        <div id="emptyCart" class="empty-box d-none" >
          <h3>Your cart is empty ☹</h3>
          <br>
          <a href="products.html" class="btn">Continue Shopping</a>
        </div>
      </center>
```

```
<table class="table" id="cartTable">
  <thead>
    <tr>
      <th>Product</th>
      <th>Quantity</th>
      <th>Price</th>
      <th>Subtotal</th>
      <th>Remove</th>
    </tr>
  </thead>
  <tbody id="cartBody"></tbody>
</table>
<div class="total-box">
  Total: ₹<span id="totalAmount">0</span>
</div>
<button class="checkout-btn" onclick="proceedToCheckout()">Proceed to Checkout</button>
</div>
<script>
// Sample product data structure for demo; replace with your actual cart data in localStorage
// Load cart
function loadCart() {
  const cart = JSON.parse(localStorage.getItem("cartItems")) || [];
  const body = document.getElementById("cartBody");
  const emptyBox = document.getElementById("emptyCart");
  const table = document.getElementById("cartTable");
  const cartCount = document.getElementById("cart-count");
  const totalBox = document.querySelector(".total-box");
  const checkoutBtn = document.querySelector(".checkout-btn");
  body.innerHTML = "";
  // If cart empty
```

```
        if (cart.length === 0) {
          emptyBox.classList.remove("d-none");
          table.classList.add("d-none");
          totalBox.style.display = "none";
          checkoutBtn.style.display = "none";
          cartCount.innerText = "0";
          return;
        }
        emptyBox.classList.add("d-none");
        table.classList.remove("d-none");
        totalBox.style.display = "block";
        checkoutBtn.style.display = "inline-block";
        cartCount.innerText = cart.reduce((acc, item) => acc + item.quantity, 0);
        let total = 0;
        cart.forEach((item, index) => {
          const subtotal = item.quantity * item.price;
          total += subtotal;
          body.innerHTML += `
            <tr>
              <td>
                <div class="product-info">
                  <img src="${item.image}" alt="${item.name}" class="cart-img" />
                  <div class="product-details">
                    <div class="product-name">${item.name}</div>
                    <div class="product-size">Size: ${item.size || 'M'}</div>
                  </div>
                </div>
              </td>
              <td>
                <input type="number" min="1" class="quantity-input"
value="${item.quantity}" onchange="updateQty(${index}, this.value)" />
              </td>
```

```
            <td class="price">₹${item.price.toFixed(2)}</td>
            <td class="subtotal">₹${subtotal.toFixed(2)}</td>
            <td>
              <button class="remove-btn"
onclick="removeItem(${index})">Remove</button>
            </td>
          </tr>
        `;
      });
      document.getElementById("totalAmount").innerText = total.toFixed(2);
    }
    // Change quantity using input
    function updateQty(index, value) {
      let cart = JSON.parse(localStorage.getItem("cartItems")) || [];
      const qty = parseInt(value);
      if (isNaN(qty) || qty < 1) return;
      cart[index].quantity = qty;
      localStorage.setItem("cartItems", JSON.stringify(cart));
      loadCart();
    }
    // Remove item
    function removeItem(index) {
      let cart = JSON.parse(localStorage.getItem("cartItems")) || [];
      cart.splice(index, 1);
      localStorage.setItem("cartItems", JSON.stringify(cart));
      loadCart();
    }
    function proceedToCheckout() {
       // Redirect or further checkout logic here
        // 1. Clear cart table
       const cartTableBody = document.getElementById("cart-items");
       if (cartTableBody) {
```

```
            cartTableBody.innerHTML = ""; // removes all rows
        }
        // 2. Reset cart count
        const cartCount = document.getElementById("cart-count");
        if (cartCount) {
            cartCount.innerText = "0";
        }
        // 3. Clear localStorage cart data (if you used it)
        localStorage.removeItem("cartItems");
        // 4. Redirect to next page
       window.location.href = "order-success.html";
    }
    // Initialize cart display on page load
    loadCart();
    </script>
    </body>
    </html>
```

main.js :

```
// Attach event listener to all add-to-cart buttons
document.addEventListener("DOMContentLoaded", function () {
    const buttons = document.querySelectorAll(".add-to-cart");
    buttons.forEach(button => {
        button.addEventListener("click", function (e) {
            e.preventDefault();
            // Get product card
            const card = button.closest(".card");
            const title = card.querySelector(".card-title").innerText;
            const image = card.querySelector("img").getAttribute("src");
            // Extract price automatically from text: "Price : ₹999"
            const priceText = card.querySelector(".card-text:nth-of-
type(2)").innerText;
```

```
            const price = parseInt(priceText.replace(/[^0-9]/g, ""));
            // Product object
            const product = {
                name: title,
                image: image,
                price: price,
                quantity: 1
            };
            addToCart(product);
        });
    });
    updateCartCount();
});
function addToCart(product) {
    let cart = JSON.parse(localStorage.getItem("cartItems")) || [];
    // Check if product already exists
    const existingItem = cart.find(item => item.name === product.name);
    if (existingItem) {
        existingItem.quantity += 1;
    } else {
        cart.push(product);
    }
    localStorage.setItem("cartItems", JSON.stringify(cart));
    updateCartCount();
}
// Update cart badge
function updateCartCount() {
    const cart = JSON.parse(localStorage.getItem("cartItems")) || [];
    const total = cart.reduce((sum, item) => sum + item.quantity, 0);
    const badge = document.getElementById("cart-count");
    if (badge) badge.textContent = total;
}
```

product.html :

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1.0"/>
  <title>QuickCart</title>
  <link rel="icon" type="image/phg" href="placeholder/icon.png">
  <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css"
rel="stylesheet">
  <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/bootstrap.bundle.min.js
"></script>
  <link rel="stylesheet" href="css/styles.css" />
  <!-- for font -->
  <link rel="preconnect" href="https://fonts.googleapis.com">
  <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
  <link
href="https://fonts.googleapis.com/css2?family=Poppins:ital,wght@0,100;0,200;
0,300;0,400;0,500;0,600;0,700;0,800;0,900;1,100;1,200;1,300;1,400;1,500;1,600
;1,700;1,800;1,900&display=swap"
    rel="stylesheet">
     <!-- foricon -->
    <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap-
icons@1.10.5/font/bootstrap-icons.css">
    <!-- for font -->
    <link rel="preconnect" href="https://fonts.googleapis.com">
<link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
<link
href="https://fonts.googleapis.com/css2?family=Raleway:ital,wght@0,100..900;
```

```
1,100..900&display=swap" rel="stylesheet">
  </head>
<style>
 #divs {
  position: relative;
  background-image: url("placeholder/images1.png");
  background-repeat: no-repeat;
  background-size: cover;
  background-position: center;
  height: 80vh;
  opacity: 0;
  transform: translateY(-100px); /* start above view */
  transition: opacity 1.5s ease, transform 1.5s ease;
}
#divs.visible {
  opacity: 1;
  transform: translateY(0); /* slide down into view */
}
.fixed-btn {
  position: absolute;
  bottom: 120px;
  left: 30%;
  transform: translateX(-50%);
  padding: 14px 30px;
  font-size: 18px;
  font-weight: bold;
  background: linear-gradient(90deg, #00c6ff, #0072ff);
  color: white;
  border: none;
  border-radius: 10px;
  text-align: center;
  text-decoration: none;
```

```
  z-index: 1;
  box-shadow: 0 0 10px #00c6ff;
  transition: 0.3s;
}
.fixed-btn:hover {
  transform: translateX(-50%) scale(1.05);
}
</style>
<body>
  <div class="container-fluid">
<nav class="navbar navbar-expand-md  fixed-top">
  <div class="container-fluid">
    <!-- Logo -->
    <a class="navbar-brand d-flex align-items-center" href="#">
      <img src="placeholder/logo.png" alt="Logo" style="height: 40px; width:
50px; margin-right: 5px;">
    </a>
     <h2 style="text-align: justify; font-family: poppins, sans-serif; color:
aliceblue; margin-right: 5px;">QuickCart </h2>
    <!-- Toggler for Mobile -->
    <button class="navbar-toggler navbar-light  " type="button" data-bs-
toggle="collapse" data-bs-target="#mainNavbar">
      <span  class="navbar-toggler-icon"></span>
    </button>
    <!-- Collapsible Navbar Content -->
    <div class="collapse navbar-collapse" id="mainNavbar">
      <!-- LEFT MENU -->
      <!-- CENTER SEARCH BAR -->
 <form class="search-form" onsubmit="searchProducts(event)">
  <div class="search-container">
    <span class="search-icon">
      <svg xmlns="http://www.w3.org/2000/svg" width="16" height="16"
```

```
fill="#6c757d" class="bi bi-search" viewBox="0 0 16 16">
        <path d="M11.742 10.344a6.5 6.5 0 1 0-1.397 1.397h-.001l3.85 3.85a1 1 0
0 0 1.415-1.414l-3.85-3.85zm-5.242 1.397a5.5 5.5 0 1 1
                0-11 5.5 5.5 0 0 1 0 11z"/>
      </svg>
    </span>
    <input id="searchInput" class="form-control search-input" type="search"
placeholder="Search for products, brands and more">
  </div>
</form>
      <!-- for cart and product -->
      <ul class="navbar-nav nav-underline ms-auto">
        <li class="nav-item "><a class="nav-link" href="men.html">Men</a></li>
        <li class="nav-item "><a class="nav-link"
href="women.html">Women</a></li>
        <li class="nav-item "><a class="nav-link" href="kids.html">Kids</a></li>
       <li class="nav-item active"><a class="nav-link"
href="products.html">Home</a></li>
        <li class="nav-item">
  <a class="nav-link position-relative" href="cart.html">
  Cart
  <span id="cart-count"
        class="position-absolute top-0 start-100 translate-middle badge rounded-pill
bg-danger"
        style="font-size: 0.75rem; padding: 4px 6px;">
  </span>
</a></li>
</ul>
    </div>
  </div>
</nav>
</div>
```

```
<br><br><br>
  <!-- poster -->
 <div class="container-fluid banner py-5" id="divs">
  <div class="row justify-content-center">
   <div class="col-12 col-sm-6 col-md-4 text-center">
    <a href="products.html" class="btn btn-primary fixed-btn">Buy Now</a>
   </div>
  </div>
</div>
<div class="container py-5">
  <div class="row text-center">
   <!-- Feature 1 -->
   <div class="col-6 col-md-3 mb-4">
    <i class="bi bi-truck display-4 text-primary"></i>
    <h5 class="mt-3 fw-bold">Free Shipping</h5>
    <p class="text-muted small">On all orders above ₹499</p>
   </div>
   <!-- Feature 2 -->
   <div class="col-6 col-md-3 mb-4">
    <i class="bi bi-shield-lock display-4 text-primary"></i>
    <h5 class="mt-3 fw-bold">Secure Payment</h5>
    <p class="text-muted small">100% safe and encrypted</p>
   </div>
   <!-- Feature 3 -->
   <div class="col-6 col-md-3 mb-4">
    <i class="bi bi-cash-coin display-4 text-primary"></i>
    <h5 class="mt-3 fw-bold">Money Back</h5>
    <p class="text-muted small">Easy 7-day return policy</p>
   </div>
   <!-- Feature 4 -->
   <div class="col-6 col-md-3 mb-4">
    <i class="bi bi-headset display-4 text-primary"></i>
```

```
      <h5 class="mt-3 fw-bold">24/7 Support</h5>
      <p class="text-muted small">Friendly customer service</p>
    </div>
  </div>
</div>
<!-- offer -->
<div class="container mt-4">
  <div class="row g-3">
    <!-- Left Side Large Banner -->
    <div class="col-md-6">
      <div class="offer-banner" style="height: 300px;">
        <img src="placeholder/cart1.png" alt="Women's Style">
        <div class="offer-content">
          <h3 class="fw-bold text-dark-shadow">Women's Style</h3>
          <p class="mb-2">Up to 70% Off</p>
          <div class="btn-center">
  <a href="#" class="btn btn-primary add-to-cart">Add To Cart</a>
</div>
        </div>
      </div>
    </div>
    <!-- Right Side Small Banners -->
    <div class="col-md-6">
      <div class="row g-3">
        <!-- Handbag -->
        <div class="col-6">
          <div class="offer-banner" style="height: 140px;">
            <span class="discount-badge">25% OFF</span>
            <img src="placeholder/women/Wcart17.png" alt="Handbag">
            <div class="offer-content">
              <h6 class="fw-bold text-dark-shadow ">Frog</h6>
              <div class="btn-center">
```

```
  <a href="#" class="btn btn-primary add-to-cart">Add To Cart</a>
</div>
          </div>
         </div>
        </div>

        <!-- Watch -->
        <div class="col-6">
         <div class="offer-banner" style="height: 140px;">
          <span class="discount-badge">40% OFF</span>
          <img src="placeholder/kids/Kcart6.png" alt="Watch">
          <div class="offer-content">
           <h6 class="fw-bold text-dark-shadow">T shirt</h6>
           <div class="btn-center">
  <a href="#" class="btn btn-primary add-to-cart">Add To Cart</a>
</div>
          </div>
         </div>
        </div>
        <!-- Backpack -->
        <div class="col-12">
         <div class="offer-banner" style="height: 150px;">
          <img src="placeholder/men/Mcart1.png" alt="Backpack">
          <div class="offer-content">
           <h6 class="fw-bold text-dark-shadow">Shirt</h6>
           <p class="small">Min. 40-60% Off</p>
           <div class="btn-center">
  <a href="#" class="btn btn-primary add-to-cart">Add To Cart</a>
</div>
          </div>
         </div>
        </div>
```

```
    </div>
   </div>
  </div>
</div>
 <br><br>
   <!-- container 1 -->
<div class="container mt-4">
   <h2 style="text-align: center; font-family: Raleway, sans-serif; font-weight:
bold;">All Products</h2><br>
   <!-- row 1 -->
<div class="row g-3">
   <div class="col-12 col-sm-6 col-md-4 col-lg-3">
     <div class="card product-card">
       <img src="placeholder/women/Wcart11.png" class="card-img-top product-
img" alt="Product Image">
       <div class="card-body">
         <h5 class="card-title">Floral Summer Dress</h5>
         <p class="card-text">Star Rating  :★ ★ ★ ☆ ☆</p>
         <p class="card-text">Price : ₹999</p>
         <div class="btn-center">
           <a href="#" class="btn btn-primary add-to-cart">Add To Cart</a>
         </div>
       </div>
     </div>
   </div>
   <div class="col-12 col-sm-6 col-md-4 col-lg-3">
     <div class="card product-card">
       <img src="placeholder/cart2.png" class="card-img-top product-img"
alt="Product Image">
       <div class="card-body">
         <h5 class="card-title">Floral Traditional Dress</h5>
         <p class="card-text">Star Rating :★ ★ ★ ★ ★</p>
```

```
        <p class="card-text">Price : ₹999</p>
          <div class="btn-center">
            <a href="#" class="btn btn-primary add-to-cart">Add To Cart</a>
          </div>
        </div>
      </div>
    <div class="col-12 col-sm-6 col-md-4 col-lg-3">
      <div class="card product-card">
        <img src="placeholder/cart3.png" class="card-img-top product-img"
alt="Product Image">
        <div class="card-body">
          <h5 class="card-title">Butterfly Frock</h5>
          <p class="card-text">Star Rating :★ ★ ★ ★ ☆</p>
        <p class="card-text">Price : ₹999</p>
          <div class="btn-center">
            <a href="#" class="btn btn-primary add-to-cart">Add To Cart</a>
          </div>
        </div>
      </div>
    </div>
    <div class="col-12 col-sm-6 col-md-4 col-lg-3">
      <div class="card product-card">
        <img src="placeholder/cart4.png" class="card-img-top product-img"
alt="Product Image">
        <div class="card-body">
          <h5 class="card-title">Green Kurti</h5>
          <p class="card-text">Star Rating :★ ★ ★ ★ ★</p>
        <p class="card-text">Price : ₹999</p>
          <div class="btn-center">
            <a href="#" class="btn btn-primary add-to-cart">Add To Cart</a>
          </div>
```

```
        </div>
      </div>
    </div>
</div>
<br>
        <!-- row 2 -->
 <div class="row">
    <div class="col-12 col-sm-6 col-md-4 col-lg-3">
      <div class="card product-card">
        <img src="placeholder/cart5.png" class="card-img-top product-img"
alt="Product Image">
        <div class="card-body">
          <h5 class="card-title">Traditional Saree</h5>
          <p class="card-text">Star Rating :★ ★ ★ ★ ☆</p>
        <p class="card-text">Price : ₹999</p>
          <div class="btn-center">
            <a href="#" class="btn btn-primary add-to-cart">Add To Cart</a>
          </div>
        </div>
      </div>
    </div>
    <div class="col-12 col-sm-6 col-md-4 col-lg-3">
      <div class="card product-card">
        <img src="placeholder/cart6.png" class="card-img-top product-img"
alt="Product Image">
        <div class="card-body">
          <h5 class="card-title">Full Kurti</h5>
          <p class="card-text">Star Rating :★ ★ ★ ★ ☆</p>
        <p class="card-text">Price : ₹999</p>
          <div class="btn-center">
            <a href="#" class="btn btn-primary add-to-cart">Add To Cart</a>
          </div>
```

```
      </div>
    </div>
  </div>
  <div class="col-12 col-sm-6 col-md-4 col-lg-3">
    <div class="card product-card">
      <img src="placeholder/cart7.png" class="card-img-top product-img"
alt="Product Image">
      <div class="card-body">
        <h5 class="card-title">Party Tranding</h5>
        <p class="card-text">Star Rating :★ ★ ★ ★ ★</p>
      <p class="card-text">Price : ₹999</p>
        <div class="btn-center">
          <a href="#" class="btn btn-primary add-to-cart">Add To Cart</a>
        </div>
      </div>
    </div>
  </div>
  <div class="col-12 col-sm-6 col-md-4 col-lg-3">
    <div class="card product-card">
      <img src="placeholder/cart8.png" class="card-img-top product-img"
alt="Product Image">
      <div class="card-body">
        <h5 class="card-title">Half Saree</h5>
        <p class="card-text">Star Rating :★ ★ ★ ★ ☆</p>
      <p class="card-text">Price : ₹999</p>
        <div class="btn-center">
          <a href="#" class="btn btn-primary add-to-cart">Add To Cart</a>
        </div>
      </div>
    </div>
  </div>
</div>
```

```
</div>
   <!-- container 2-->
<div class="container mt-4">
   <!-- <h1>Men Collections</h1> -->
   <!-- row 1 -->
<div class="row g-3">
   <div class="col-12 col-sm-6 col-md-4 col-lg-3">
     <div class="card product-card">
       <img src="placeholder/men/Mcart1.png" class="card-img-top product-img"
alt="Product Image">
       <div class="card-body">
         <h5 class="card-title">Rayon Panel Shirt</h5>
         <p class="card-text">Star Rating :★ ★ ★ ★ ☆</p>
       <p class="card-text">Price : ₹999</p>
         <div class="btn-center">
           <a href="#" class="btn btn-primary add-to-cart">Add To Cart</a>
         </div>
       </div>
     </div>
   </div>
   <div class="col-12 col-sm-6 col-md-4 col-lg-3 ">
     <div class="card product-card">
       <img src="placeholder/men/Mcart2.png" class="card-img-top product-img"
alt="Product Image">
       <div class="card-body">
         <h5 class="card-title">Trendy Shirt & Pants</h5>
         <p class="card-text">Star Rating :★ ★ ★ ★ ☆</p>
       <p class="card-text">Price : ₹999</p>
         <div class="btn-center">
           <a href="#" class="btn btn-primary add-to-cart">Add To Cart</a>
         </div>
       </div>
```

```
      </div>
    </div>
    <div class="col-12 col-sm-6 col-md-4 col-lg-3">
      <div class="card product-card">
        <img src="placeholder/men/Mcart3.png" class="card-img-top product-img"
alt="Product Image">
        <div class="card-body">
          <h5 class="card-title">Lion Embroidered Designer Shirt</h5>
          <p class="card-text">Star Rating :★ ★ ★ ★ ★</p>
        <p class="card-text">Price : ₹999</p>
          <div class="btn-center">
            <a href="#" class="btn btn-primary add-to-cart">Add To Cart</a>
          </div>
        </div>
      </div>
    </div>
    <div class="col-12 col-sm-6 col-md-4 col-lg-3">
      <div class="card product-card">
        <img src="placeholder/men/Mcart4.png" class="card-img-top product-img"
alt="Product Image">
        <div class="card-body">
          <h5 class="card-title">Shirt Slim Fit Cotton</h5>
          <p class="card-text">Star Rating :★ ★ ★ ★ ☆</p>
        <p class="card-text">Price : ₹999</p>
          <div class="btn-center">
            <a href="#" class="btn btn-primary add-to-cart">Add To Cart</a>
          </div>
        </div>
      </div>
    </div>
</div>
<br>
```

```
        <!-- row 2 -->
 <div class="row">
    <div class="col-12 col-sm-6 col-md-4 col-lg-3 ">
      <div class="card product-card">
        <img src="placeholder/men/Mcart5.png" class="card-img-top product-img"
alt="Product Image">
        <div class="card-body">
          <h5 class="card-title">Pure Cotton Casual Shirt</h5>
          <p class="card-text">Star Rating :★ ★ ★ ★ ☆</p>
        <p class="card-text">Price : ₹999</p>
          <div class="btn-center">
            <a href="#" class="btn btn-primary add-to-cart">Add To Cart</a>
          </div>
        </div>
      </div>
    </div>
    <div class="col-12 col-sm-6 col-md-4 col-lg-3">
      <div class="card product-card">
        <img src="placeholder/men/Mcart6.png" class="card-img-top product-img"
alt="Product Image">
        <div class="card-body">
          <h5 class="card-title"> Full Sleeves Shirt </h5>
          <p class="card-text">Star Rating :★ ★ ★ ★ ☆</p>
        <p class="card-text">Price : ₹999</p>
          <div class="btn-center">
            <a href="#" class="btn btn-primary add-to-cart">Add To Cart</a>
          </div>
        </div>
      </div>
    </div>
    <div class="col-12 col-sm-6 col-md-4 col-lg-3">
      <div class="card product-card">
```

```
        <img src="placeholder/men/Mcart7.png" class="card-img-top product-img"
alt="Product Image">
        <div class="card-body">
          <h5 class="card-title">White Shirt Combo Pack</h5>
          <p class="card-text">Star Rating :★ ★ ★ ★ ★</p>
        <p class="card-text">Price : ₹999</p>
          <div class="btn-center">
            <a href="#" class="btn btn-primary add-to-cart">Add To Cart</a>
          </div>
        </div>
      </div>
    </div>
    <div class="col-12 col-sm-6 col-md-4 col-lg-3">
      <div class="card product-card">
        <img src="placeholder/men/Mcart8.png" class="card-img-top product-img"
alt="Product Image">
        <div class="card-body">
          <h5 class="card-title">Slim Fit Casual Shirt</h5>
          <p class="card-text">Star Rating :★ ★ ★ ★ ☆</p>
        <p class="card-text">Price : ₹999</p>
          <div class="btn-center">
            <a href="#" class="btn btn-primary add-to-cart">Add To Cart</a>
          </div>
        </div>
      </div>
    </div>
</div>
</div>
    <!-- container 3-->
<div class="container mt-4">
    <!-- <h1>Kids Collections</h1> -->
    <!-- row 1 -->
```

```
<div class="row g-3">
   <div class="col-12 col-sm-6 col-md-4 col-lg-3">
     <div class="card product-card">
       <img src="placeholder/kids/Kcart1.png" class="card-img-top product-img"
alt="Product Image">
       <div class="card-body">
         <h5 class="card-title">Rose embroidery ball gown</h5>
         <p class="card-text">Star Rating :★ ★ ★ ★ ☆</p>
       <p class="card-text">Price : ₹999</p>
         <div class="btn-center">
           <a href="#" class="btn btn-primary add-to-cart">Add To Cart</a>
         </div>
       </div>
     </div>
   </div>
   <div class="col-12 col-sm-6 col-md-4 col-lg-3 ">
     <div class="card product-card">
       <img src="placeholder/kids/Kcart2.png" class="card-img-top product-img"
alt="Product Image">
       <div class="card-body">
         <h5 class="card-title">Traditional Gown</h5>
         <p class="card-text">Star Rating :★ ★ ★ ★ ☆</p>
       <p class="card-text">Price : ₹999</p>
         <div class="btn-center">
           <a href="#" class="btn btn-primary add-to-cart">Add To Cart</a>
         </div>
       </div>
     </div>
   </div>
   <div class="col-12 col-sm-6 col-md-4 col-lg-3">
     <div class="card product-card">
       <img src="placeholder/kids/Kcart3.png" class="card-img-top product-img"
```

```
alt="Product Image">
        <div class="card-body">
          <h5 class="card-title">StarAndDaisy Party Dress</h5>
          <p class="card-text">Star Rating :★ ★ ★ ★ ☆</p>
        <p class="card-text">Price : ₹999</p>
          <div class="btn-center">
            <a href="#" class="btn btn-primary add-to-cart">Add To Cart</a>
          </div>
        </div>
      </div>
    </div>
    <div class="col-12 col-sm-6 col-md-4 col-lg-3">
      <div class="card product-card">
        <img src="placeholder/kids/Kcart4.png" class="card-img-top product-img"
alt="Product Image">
        <div class="card-body">
          <h5 class="card-title">Apparel Baby Wear</h5>
          <p class="card-text">Star Rating :★ ★ ★ ★ ☆</p>
        <p class="card-text">Price : ₹999</p>
          <div class="btn-center">
            <a href="#" class="btn btn-primary add-to-cart">Add To Cart</a>
          </div>
        </div>
      </div>
    </div>
</div>
<br>
        <!-- row 2 -->
 <div class="row">
    <div class="col-12 col-sm-6 col-md-4 col-lg-3 ">
      <div class="card product-card">
        <img src="placeholder/kids/Kcart5.png" class="card-img-top product-img"
```

```
alt="Product Image">
        <div class="card-body">
          <h5 class="card-title">Best kids coat</h5>
          <p class="card-text">Star Rating :★ ★ ★ ★ ☆</p>
        <p class="card-text">Price : ₹999</p>
          <div class="btn-center">
            <a href="#" class="btn btn-primary add-to-cart">Add To Cart</a>
          </div>
        </div>
      </div>
    </div>
    <div class="col-12 col-sm-6 col-md-4 col-lg-3">
      <div class="card product-card">
        <img src="placeholder/kids/Kcart6.png" class="card-img-top product-img"
alt="Product Image">
        <div class="card-body">
          <h5 class="card-title"> T Shirt </h5>
          <p class="card-text">Star Rating :★ ★ ★ ★ ☆</p>
        <p class="card-text">Price : ₹999</p>
          <div class="btn-center">
            <a href="#" class="btn btn-primary add-to-cart">Add To Cart</a>
          </div>
        </div>
      </div>
    </div>
    <div class="col-12 col-sm-6 col-md-4 col-lg-3">
      <div class="card product-card">
        <img src="placeholder/kids/Kcart7.png" class="card-img-top product-img"
alt="Product Image">
        <div class="card-body">
          <h5 class="card-title">Traditional wear</h5>
          <p class="card-text">Star Rating :★ ★ ★ ★ ☆</p>
```

```
        <p class="card-text">Price : ₹999</p>
          <div class="btn-center">
            <a href="#" class="btn btn-primary add-to-cart">Add To Cart</a>
          </div>
        </div>
      </div>
    </div>
    <div class="col-12 col-sm-6 col-md-4 col-lg-3">
      <div class="card product-card">
        <img src="placeholder/kids/Kcart8.png" class="card-img-top product-img"
alt="Product Image">
        <div class="card-body">
          <h5 class="card-title">full Sleeves T Shirt</h5>
          <p class="card-text">Star Rating :★ ★ ★ ★ ☆</p>
        <p class="card-text">Price : ₹999</p>
          <div class="btn-center">
            <a href="#" class="btn btn-primary add-to-cart">Add To Cart</a>
          </div>
        </div>
      </div>
    </div>
  <br>
<div class="container mt-4">
  <div class="row g-3">
    <!-- Men's Fashion -->
    <div class="col-md-6">
      <div class="promo-banner">
        <img src="placeholder/cart1.png" alt="Men's Fashion">
        <div class="promo-text">
          <p class="text-light mb-1">Weekend Sale</p>
          <h5>Men's Fashion</h5>
          <p>Flat <span class="text-info fw-bold">75% OFF</span></p>
```

```
          <button class="shop-now-btn">Shop Now</button>
        </div>
      </div>
    </div>
    <!-- Women's Wear -->
    <div class="col-md-6">
      <div class="promo-banner">
        <img src="placeholder/men/Mcart1.png" alt="Women's Wear">
        <div class="promo-text">
          <p class="text-light mb-1">Fashion Style</p>
          <h5>Women's Wear</h5>
          <p>Min. <span class="text-info fw-bold">30–75% OFF</span></p>
          <button class="shop-now-btn">Shop Now</button>
        </div>
      </div>
    </div>
  </div>
</div>
</body>
<script>
  window.addEventListener("load", () => {
    const banner = document.getElementById("divs");
    setTimeout(() => {
    banner.classList.add("visible");
      // Scroll to it smoothly after it appears
      banner.scrollIntoView({ behavior: "smooth", block: "start" });
    }, 300); // short delay before showing
  });
</script>
<script src="js/main.js"></script>
</html>
```

style.css :

```
.img1{
      width: 60px;
      height: 80px;
    }
    h6{
      color: rgb(208, 214, 227);
      font-family: 'Franklin Gothic Medium', 'Arial Narrow', Arial, sans-serif;
    }
    .navbar{
      background-color:#000000;
      height: 60px;
      font-weight: 700;
      font-size: 18px;
    }
    .nav-link{
      color:rgb(253, 255, 255)
    }
    .nav-link:hover{
      color: white;
    }
      .nav-item.active {
  background-color: #506f9c;
  color: white;
  border-radius: 6px;
}
 .product-card {
  border-radius: 10px;
  box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
  overflow: hidden;
}
```

```
.product-card img {
  height: 220px;
  object-fit: cover;
}
.card-title {
  font-size: 16px;
  font-weight: 600;
}
.card-text {
  font-size: 14px;
  margin-bottom: 10px;
}
.btn-sm {
  padding: 6px 12px;
  font-size: 14px;
}
  #cart-count {
  font-size: 0.75rem;
  padding: 5px 7px;
}
 .product-card img {
      height: 200px;
      object-fit: cover;
    }
  .search-form {
  max-width: 500px;
  margin: auto;
}
.search-container {
  position: relative;
}
.search-input {
  padding-left: 2.2rem;
```

```
  height: 45px;
  width: 500px;
  border-radius: 10px;
  border: 1px solid #ddd;
}
.search-icon {
  position: absolute;
  top: 50%;
  left: 12px;
  transform: translateY(-50%);
  pointer-events: none;
  color: #aaa;
}
 #divs {
  position: relative;
  background-image: url("placeholder/images1.png");
  background-repeat: no-repeat;
  background-size: cover;
  background-position: center;
  height: 80vh;
  opacity: 0;
  transform: translateY(-100px); /* start above view */
  transition: opacity 1.5s ease, transform 1.5s ease;
}
#divs.visible {
  opacity: 1;
  transform: translateY(0); /* slide down into view */
}
.fixed-btn {
  position: absolute;
  bottom: 120px;
  left: 30%;
  transform: translateX(-50%);
  padding: 14px 30px;
```

```
  font-size: 18px;
  font-weight: bold;
  background: linear-gradient(90deg, #00c6ff, #0072ff);
  color: white;
  border: none;
  border-radius: 10px;
  text-align: center;
  text-decoration: none;
  z-index: 1;
  box-shadow: 0 0 10px #00c6ff;
  transition: 0.3s;
}
.fixed-btn:hover {
  transform: translateX(-50%) scale(1.05);
}
  .promo-banner {
    position: relative;
    border-radius: 10px;
    overflow: hidden;
    color: #fff;
  }
  .promo-banner img {
    width: 100%;
    height: 200px;
    object-fit: cover;
    filter: brightness(0.7);
  }
  .promo-text {
    position: absolute;
    top: 20%;
    left: 10%;
    z-index: 2;
  }
```

```
.promo-text h5 {
  font-weight: bold;
}
.promo-text p {
  margin: 0;
  font-size: 0.9rem;
}
.shop-now-btn {
  background-color: #fff;
  color: #000;
  padding: 5px 12px;
  border-radius: 20px;
  font-weight: 500;
  font-size: 0.85rem;
  margin-top: 10px;
  border: none;
}
.offer-banner {
  position: relative;
  color: white;
  overflow: hidden;
  border-radius: 8px;
}
.offer-banner img {
  width: 100%;
  height: 100%;
  object-fit: cover;
  filter: brightness(0.75);
}
.offer-content {
  position: absolute;
  top: 15%;
  left: 10%;
```

```
  z-index: 2;
}
.discount-badge {
  position: absolute;
  top: 10px;
  left: 10px;
  background-color: #27517e;
  color: white;
  padding: 4px 8px;
  font-size: 0.75rem;
  font-weight: bold;
  border-radius: 4px;
}
.shop-btn {
  background: white;
  border: none;
  padding: 6px 14px;
  border-radius: 20px;
  font-size: 0.9rem;
  font-weight: 500;
  margin-top: 10px;
}
.text-dark-shadow {
  color: #fff;
  text-shadow: 0 0 5px #000;
}
```

APPENDIX B
SCREENSHOTS

6.4.1 HOME PAGE

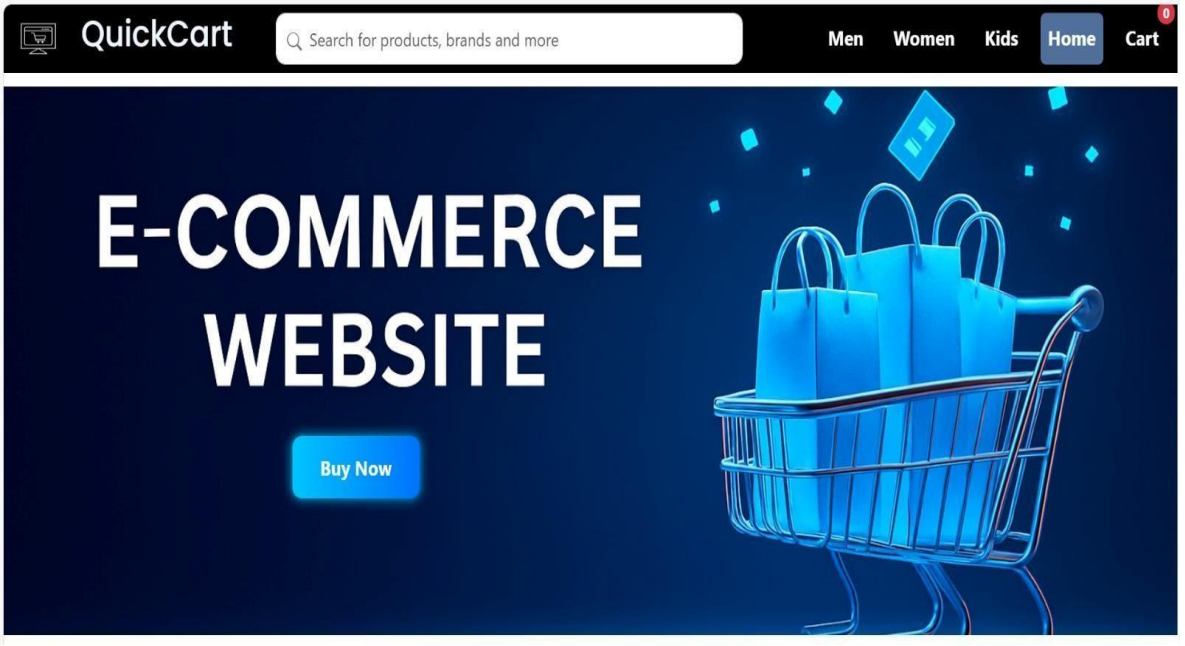
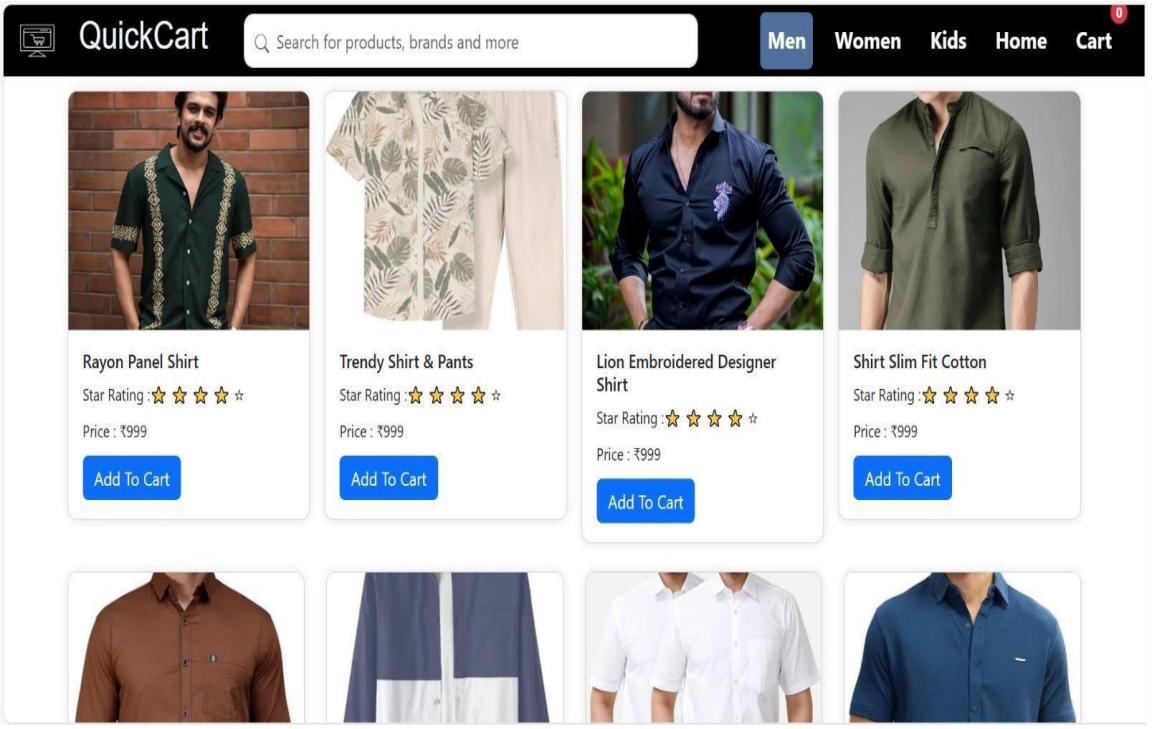
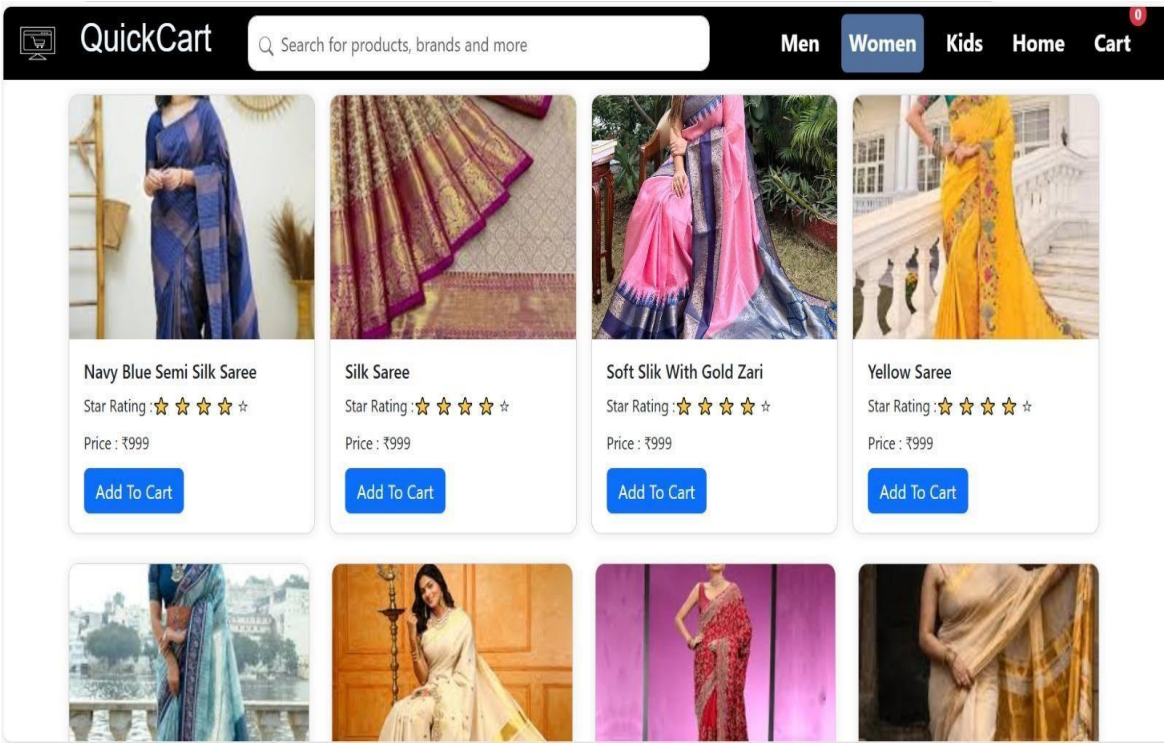


Figure 6.1 Home Page

6.4.2 PRODUCTS





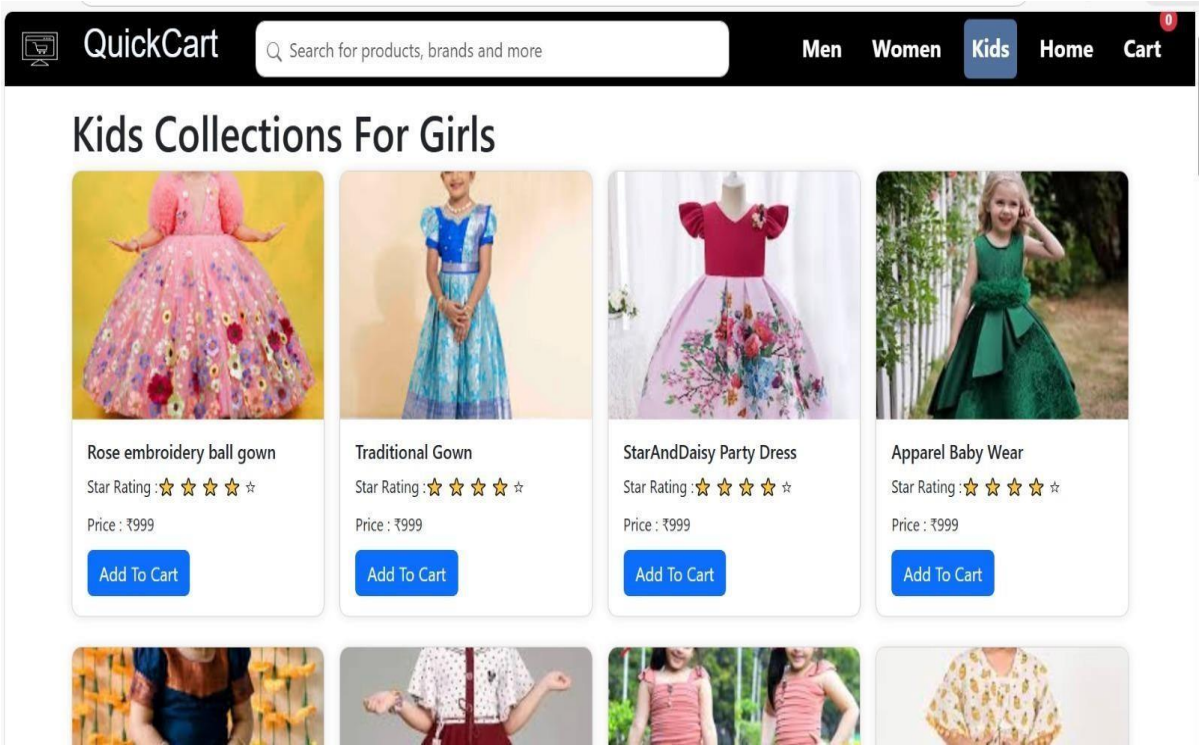


Figure 6.2 Products

6.4.3 CART

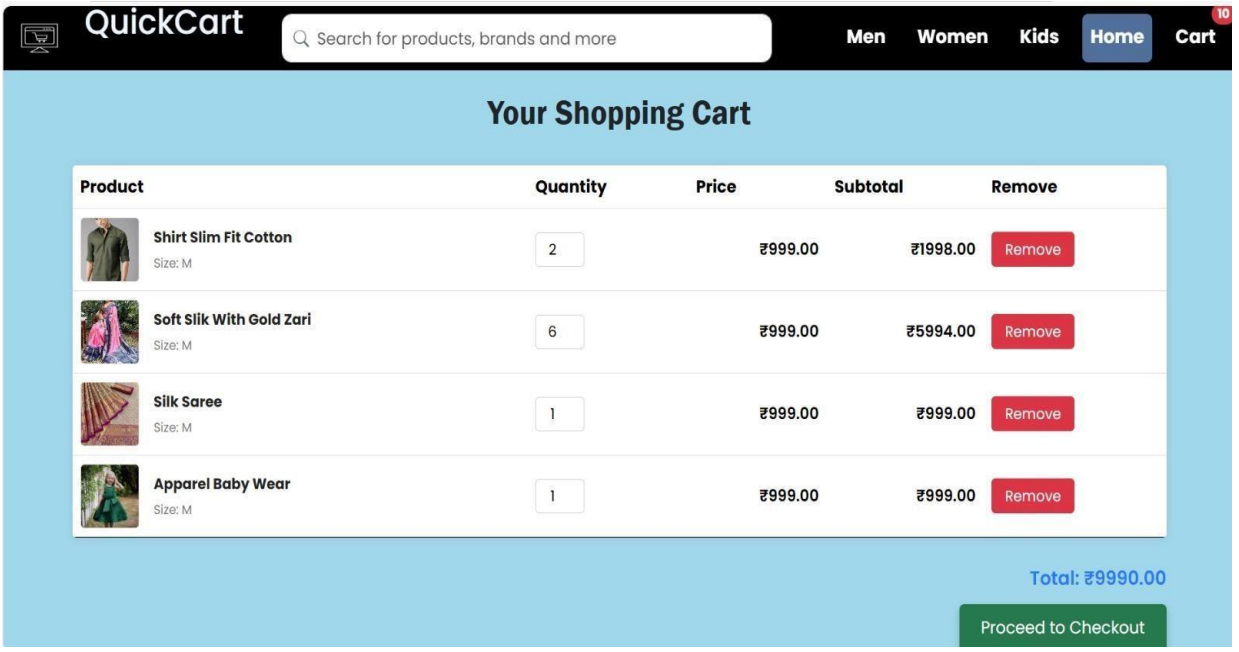


Figure 6.3 Cart

6.4.4 EMPTY CART

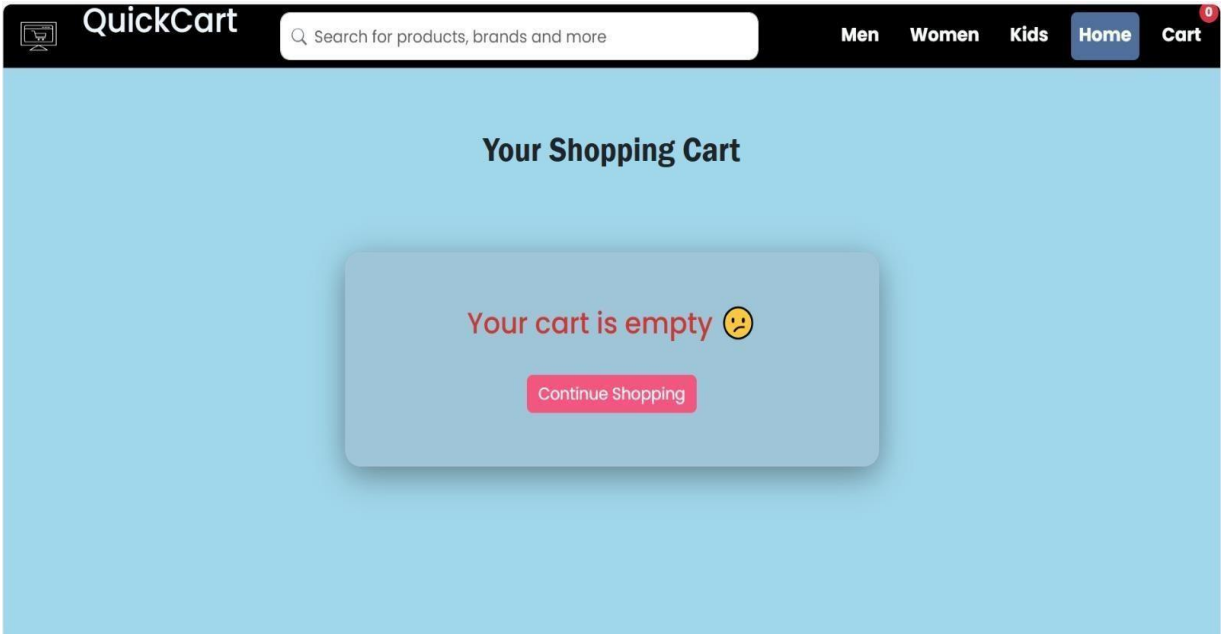


Figure 6.4 Empty Cart

6.4.5 END CART

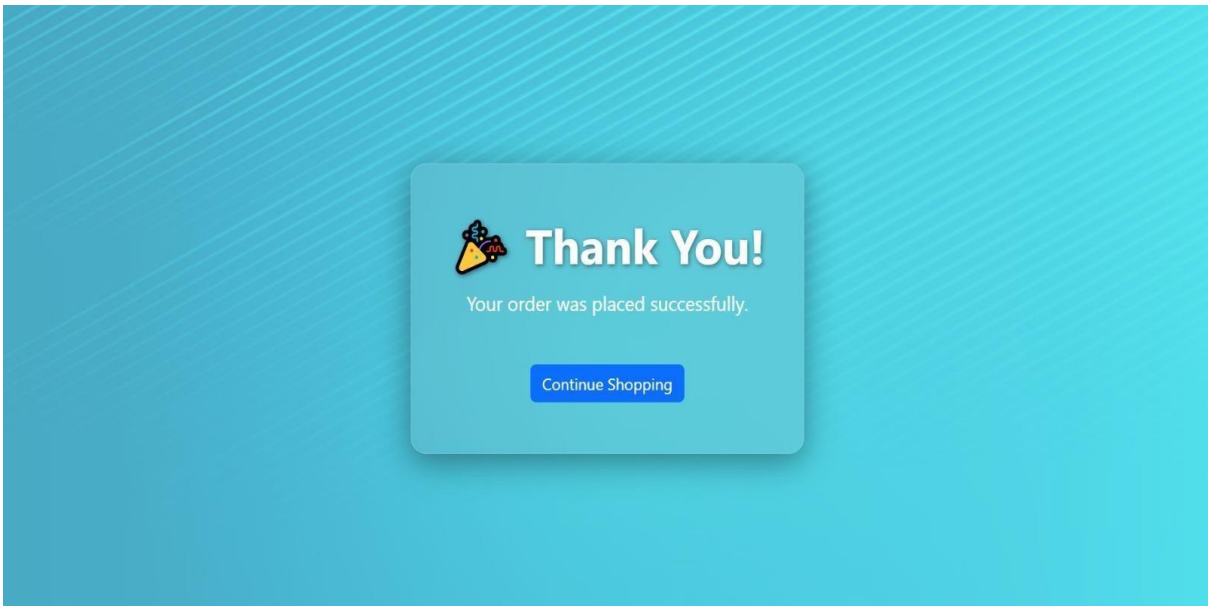


Figure 6.5 End Page

REFERENCES

BOOKS:

1. A. Silberschatz, H. Korth, And S. Sudarshan, *Database System Concepts*, 6th Ed., Mcgraw-Hill, 2010.
2. Apache Software Foundation, "Apache Tomcat — Application Server Documentation," 2023.
3. E. Gamma, R. Helm, R. Johnson, And J. Vlissides, *Design Patterns: Elements Of Reusable Object-Oriented Software*, Addison-Wesley, 1994.
4. G. Booch, J. Rumbaugh, And I. Jacobson, *The Unified Modeling Language User Guide*, Addison-Wesley, 2005.
5. J. Bloch, *Effective Java*, 3rd Ed., Addison-Wesley, 2018.

WEBSITES:

1. KonaKart – Enterprise Java eCommerce and shopping cart platform. (Provides insight into complex cart systems and frameworks)
   https://en.wikipedia.org/wiki/KonaKart
2. nopCommerce – Open-source eCommerce platform with built-in shopping cart features.
   https://www.nopcommerce.com/
3. OpenCart – PHP-based online store management system and shopping cart software.
   https://www.opencart.com/
4. Spree Commerce – Open-source headless e-commerce platform (includes cart functionality).
   https://spreecommerce.org/