Dr. Babasaheb Ambedkar Marathwada University, Chh. Sambhajinagar

G. S. Mandal's

MARATHWADA INSTITUTE OF TECHNOLOGY
CIDCO, Chh. Sambhajinagar

A project report

E-Cart-Projects

Submitted by –

Miss. Yashoda Raghubar Gautam

MSc.(CS) – 2nd Sem.

Guided by –

Dr. Mahendra Kondekar Sir

In the fulfillment of the degree

Master of Science (Computer Science) 1st Year (**2nd Sem.)**
Department of CS [PG]
Academic Year :- 2024 - 25

Acknowledgement

It gives me proud privilege to complete this project work. This is the only page where I have the opportunity to express my emotions and gratitude from the bottom of my heart.

It is my great pleasure in expressing sincere and deep gratitude towards my **guide Dr. Mahendra Kondekar,** Marathwada Institute of Technology, Cidco, Chh. Sambhajinagar, for his valuable and firm suggestions, guidance and constant support throughout this work. I am thankful to for providing me various resources and infrastructure facilities.

I also offer my most sincere thanks to Principle **Dr. M.H.Kondekar**, **Principal**, Marathwada Institute of Technology, Cidco, Chh. Sambhajinagar, my colleagues and staff members of Computer Science and Management Department, Marathwada Institute of Technology, Cidco for cooperation provided by them in many ways.

Last but not the least I am thankful to my Company boss/guide Mr. [------] for his valuable guidance to complete my work.

Mr. Rajesh Nikam
MSc(CS)

Certificate

This is to certify that Mr. Rajesh Nikam have successfully completed the project entitled "**E-Cart – An Online Shopping System**" in the fulfillment of the degree 'Master of Science (Computer Science)' in the academic year 2024-25 in the Department of Computer Science & Information Technology[PG].

During the project work, he/she has done the work very sincerely.

HOD
(Prof. S.A.Vyavahare)

Project Guide
(Prof. Mahendra Kondekar)

External Examiner

Principal
(Dr. M.H.Kondekar)

Abstract

The **E-Cart** project is a Java-based online shopping system designed to provide users with a seamless and efficient e-commerce experience. This system enables customers to browse through products, add them to

their shopping cart, and place orders with ease. It also provides administrators with tools to manage products, view customer orders, and monitor the overall system activity.

The primary aim of this project is to simulate a real-time e-commerce platform that simplifies the online buying and selling process. It enhances user interaction, reduces manual operations, and improves transaction accuracy.

Technologies used in this project include:

- **Java** for backend development
- **JDBC (Java Database Connectivity)** for database interaction
- **MySQL** as the relational database
- **HTML/CSS** for the front-end design
- **Apache Tomcat** (if using JSP/Servlets for web deployment)

Key features of the system:

- User registration and login
- Product browsing and detailed views
- Shopping cart management
- Order placement and confirmation
- Admin panel for product and order management

This project demonstrates the integration of Java with MySQL for building a full-stack application and can be extended with features like payment gateway integration and real-time tracking.

Table of Contents

1. Introduction

1.1 Overview

In today's digital era, online shopping has revolutionized the retail industry by offering convenience, variety, and competitive pricing to consumers. The E-cart project is a Java-based application integrated with a MySQL database that aims to simulate an online shopping platform. It allows users to browse products, add them to a cart, and proceed to checkout, providing an intuitive and user-friendly experience. This system is designed to replicate real-world e-commerce functionalities such as user authentication, product management, and order processing.

1.2 Objectives

The main objectives of the E-cart project are:

- To develop a fully functional e-commerce platform using Java and MySQL.
- To provide users with the ability to register, login, and manage their profile.
- To allow customers to browse, search, and filter products based on categories.
- To implement a shopping cart system where users can add or remove items.
- To enable secure order placement and generate order summaries.
- To provide admin functionalities for product and user management.

1.3 Scope of the Project

The scope of this project includes the development of both the front-end and back-end systems of the E-cart application. The application will support:

- A customer-facing interface for product selection and checkout.
- An admin panel for managing inventory and viewing user activity.
- Integration with a MySQL database to store and retrieve information related to users, products, and transactions.
- Basic security features such as password hashing and session management. The project is limited to essential e-commerce functionalities and does not include payment gateway integration or real-time delivery tracking.

2. System Requirements

2.1 Hardware Requirements

To ensure smooth development, testing, and execution of the E-cart application, the following hardware components are recommended:

- **Processor:** Intel Core i3 or higher
- **RAM:** Minimum 4 GB (8 GB recommended for better performance)
- **Hard Disk:** Minimum 250 GB of free storage space
- **Monitor:** 14" or larger display with a resolution of at least 1366x768
- **Input Devices:** Keyboard and Mouse
- **Network:** Internet connectivity for testing database connectivity and updates

2.2 Software Requirements

The project uses Java and MySQL for development and requires the following software components:

- **Operating System:** Windows 10 / Linux / macOS
- **Java Development Kit (JDK):** Version 8 or above
- **IDE:** Eclipse / IntelliJ IDEA / NetBeans (any one)
- **Database:** MySQL Server 5.7 or higher
- **Web Server (optional):** Apache Tomcat (if developing with JSP/Servlet)
- **Browser:** Google Chrome / Mozilla Firefox (for admin or user interface preview)
- **Other Tools:**
    - XAMPP/WAMP (if using PHP for any module)
    - MySQL Workbench (for database management)

3. Technology Stack

3.1 Java

Java is the core programming language used for developing the back-end logic of the E-cart application. Its object-oriented features, platform independence, and rich library support make it ideal for building robust and scalable applications. Java handles the main business logic, user interactions, and integration with the database.

3.2 MySQL

MySQL is an open-source relational database management system used to store all the application's data, such as user credentials, product details, order history, and transaction records. It is chosen for its reliability, ease of use, and efficient data handling capabilities.

3.3 JDBC (Java Database Connectivity)

JDBC is the API that connects Java applications with the MySQL database. It enables the execution of SQL queries directly from the Java code, allowing seamless data operations like insert, update, delete, and retrieve. JDBC ensures secure and efficient communication between the application and the database.

3.4 HTML/CSS (if applicable)

If the application includes a user interface built with web technologies, HTML is used to structure the web pages, while CSS is applied for styling and enhancing the visual appearance. These technologies ensure a responsive and user-friendly design for both customers and administrators interacting with the system through a browser.

4. System Analysis

4.1 Existing System

In traditional retail and small-scale businesses, the existing system for product sales is manual or semi-automated. Customers have to visit physical stores, which limits their shopping flexibility. Some local stores may maintain inventory in spreadsheets or offline software, lacking the capability for online shopping or centralized data access. These systems are time-consuming, prone to errors, and lack features like real-time product availability, user accounts, and order tracking.

4.2 Proposed System

The proposed system is an online E-cart platform developed using Java and MySQL, aiming to digitalize the shopping experience. This system allows users to:

- Browse and search for products
- Add products to a cart
- Place orders securely
- View order history and profiles
  Additionally, the admin can manage products, view orders, and monitor customer activities. The proposed system ensures data accuracy, user convenience, and better business scalability. It also reduces manual errors and enhances customer satisfaction by offering 24/7 shopping access.

4.3 Feasibility Study

The feasibility study evaluates whether the proposed system is viable from different perspectives:

- **Technical Feasibility:** The project utilizes widely-used technologies like Java, MySQL, and JDBC, which are supported by ample documentation and community support. The required hardware and software are readily available.
- **Economic Feasibility:** The cost of development is minimal as open-source tools are used. No high-end infrastructure or commercial licenses are required, making the system affordable for small businesses.
- **Operational Feasibility:** The system is user-friendly and easy to operate for both customers and administrators. Minimal training is required, and the UI can be customized for different business needs.
- **Time Feasibility:** The development timeline is reasonable, and the system can be implemented in phases, starting with core features and gradually expanding.

5. System Design

The System Design phase focuses on the blueprint of the E-cart application, highlighting its functionality, structure, and data flow. This section includes various diagrams that illustrate the interaction between users and the system, data storage, process decomposition, object-oriented structure, and system architecture.

5.1 Use Case Diagrams

The Use Case Diagram visually represents the interactions between various types of users (actors) and the system. It identifies all major functionalities and how users interact with them.

Actors:

- Customer
- Administrator

Use Cases:

- Register/Login
- Browse Products
- Add to Cart
- Place Order
- View Orders
- Manage Products (Admin)
- View Users (Admin)

5.2 ER (Entity-Relationship) Diagram

The ER Diagram depicts the logical structure of the database. It shows the entities involved in the system and their relationships.

Main Entities:

- Users
- Products
- Cart
- Orders
- Order_Details
- Admin

Relationships:

- A user can place multiple orders.
- An order can contain multiple products (many-to-many via Order_Details).
- Admin can manage all product records.

5.3 Data Flow Diagrams (DFD)

Level 0 – Context Diagram: Shows the system as a single process with external entities like User and Admin interacting with it.

Level 1 – Decomposition of Main Process: Breaks down the system into:

- User Login/Registration
- Product Browsing
- Cart Management
- Order Processing
- Admin Controls

Level 2 – Detailed Flow: Further details specific sub-processes like:

- Validating login credentials
- Updating cart
- Storing order details into the database
- Fetching product inventory for admin edits

5.4 Class Diagrams

The Class Diagram outlines the object-oriented structure of the system. It includes classes, their attributes, methods, and relationships.

Key Classes:

- **User**: userID, username, password, email → register(), login()
- **Product**: productID, name, price, category → getDetails(), updateStock()
- **Cart**: cartID, userID, items → addItem(), removeItem(), calculateTotal()
- **Order**: orderID, userID, orderDate → placeOrder(), getOrderSummary()
- **Admin**: adminID, privileges → manageProduct(), viewUsers()

Relationships: Inheritance (Admin → User), Aggregation (Cart has Products), etc.

5.5 Architecture Diagram

The Architecture Diagram illustrates the high-level structure of the system. A typical **Three-Tier Architecture** is followed:

- **Presentation Layer (Client Side)**: HTML, CSS, JS, JSP – Responsible for user interface.
- **Business Logic Layer (Server Side)**: Java Servlets – Processes input, handles logic.
- **Data Layer (Database)**: MySQL – Stores and manages data persistently.

These layers interact in a modular fashion, ensuring separation of concerns and ease of maintenance.

6. Modules Description

The E-cart system is divided into several functional modules to ensure clarity, maintainability, and scalability. Each module focuses on a specific aspect of the system and contributes to the overall functionality.

6.1 User Module

This module manages all operations related to the customers of the system.

Features:

- **Registration:** New users can create an account by providing their details like name, email, password, etc.
- **Login:** Registered users can log in using their credentials. Validation ensures secure access.
- **View Products:** Users can browse all available products, search by category or keyword, and view detailed product information.

6.2 Admin Module

The Admin Module allows administrators to manage the product inventory and monitor user activity.

Features:

- **Add Products:** Admin can add new products with details such as name, price, quantity, and category.
- **Edit Products:** Admin can update existing product information.
- **Delete Products:** Admin can remove outdated or unavailable products from the catalog.
- **View Orders:** Admin can see a list of all user orders, including order details, user info, and purchase history.

6.3 Cart Module

This module handles shopping cart operations for users.

Features:

- **Add to Cart:** Users can add one or more products to their shopping cart.
- **Remove from Cart:** Users can remove unwanted items from the cart.
- **Checkout:** Once satisfied with the cart contents, users can proceed to checkout, confirming their selected items and quantity.

6.4 Order Module

The Order Module facilitates the final steps in the e-commerce process, managing payments and order confirmation.

Features:

- **Payment (Simulated):** Although no real payment gateway is integrated, a mock payment option is provided for simulation.
- **Order Placement:** After payment, the order is confirmed, stored in the database, and an order summary is generated.
- **Order History:** Users can view their past orders and details.

7. Database Design

The E-cart database is built using MySQL and is designed to store all essential data required for user management, product handling, cart operations, and order processing.

7.1 Table: `users`

Column Name	Data Type	Description
user_id	INT (PK, AUTO_INCREMENT)	Unique ID for each user
username	VARCHAR(50)	User's name
email	VARCHAR(100)	Email address
password	VARCHAR(255)	Hashed password

Sample Data:

user_id	username	email	password
1	yashoda	yashoda@gmail.com	******

7.2 Table: `admin`

Column Name	Data Type	Description
admin_id	INT (PK, AUTO_INCREMENT)	Unique ID for admin
username	VARCHAR(50)	Admin username
password	VARCHAR(255)	Admin password

Sample Data:

admin_id	username	password
1	admin	admin123

7.3 Table: `products`

Column Name	Data Type	Description
product_id	INT (PK, AUTO_INCREMENT)	Unique product ID
name	VARCHAR(100)	Product name
description	TEXT	Product description
price	DECIMAL(10,2)	Product price
quantity	INT	Available stock
category	VARCHAR(50)	Product category (e.g. Electronics, Clothing)

Sample Data:

product_id	name	price	quantity	category
1	Smartwatch	1999.00	10	Electronics
2	T-shirt	499.00	20	Clothing

7.4 Table: `cart`

Column Name	Data Type	Description
cart_id	INT (PK, AUTO_INCREMENT)	Unique cart ID
user_id	INT (FK)	Refers to the `users` table
product_id	INT (FK)	Refers to the `products` table
quantity	INT	Quantity selected by the user

Sample Data:

cart_id	user_id	product_id	quantity
1	1	2	2

7.5 Table: `orders`

Column Name	Data Type	Description
order_id	INT (PK, AUTO_INCREMENT)	Unique order ID
user_id	INT (FK)	Refers to the `users` table
order_date	DATETIME	Date and time of order
total_price	DECIMAL(10,2)	Total order amount
status	VARCHAR(50)	Order status (e.g. Pending, Shipped, Delivered)

Sample Data:

order_id	user_id	order_date	total_price	status
101	1	2025-04-13 12:30:00	2998.00	Pending

Let me know if you'd like:

- SQL queries to **create these tables**
- **Insert statements** for the sample data
- An **ER diagram** based on this schema

Or should we continue to the **Implementation or Testing** section next?

8. Implementation

This section provides an overview of the practical development process, highlighting important Java classes, database connectivity, and visual interfaces. It demonstrates how the frontend and backend components of the E-cart system are integrated.

8.1 Key Java Classes

Below are some of the core Java classes used in the project:

- **`User.java`**
  Represents the user entity. Contains fields like `userId`, `username`, `email`, and `password`. Includes getter/setter methods and constructors.
- **`Product.java`**
  Represents products. Includes fields like `productId`, `name`, `price`, `description`, and `category`.
- **`Cart.java`**
  Maintains a list of products selected by the user. Includes methods to add/remove items and calculate totals.
- **`Order.java`**
  Handles order placement, storing date, total amount, and user reference.
- **`DBConnection.java`**
  Utility class for establishing and managing a database connection using JDBC.
- **`LoginServlet.java`** / **`RegisterServlet.java`** / **`AddToCartServlet.java`** / **`PlaceOrderServlet.java`**
  Java Servlets to handle user actions, such as login, registration, cart updates, and placing orders.

8.2 MySQL Connectivity using JDBC

A reusable JDBC connection utility:

```
java
// DBConnection.java
import java.sql.Connection;
import java.sql.DriverManager;

public class DBConnection {
    private static final String URL =
"jdbc:mysql://localhost:3306/ecart";
    private static final String USER = "root";
    private static final String PASSWORD = "your_password";

    public static Connection getConnection() {
        Connection conn = null;
        try {
            Class.forName("com.mysql.cj.jdbc.Driver");
            conn = DriverManager.getConnection(URL, USER,
PASSWORD);
        } catch (Exception e) {
            System.out.println("Database connection failed: " +
e.getMessage());
        }
        return conn;
    }
}
```

8.3 Screenshots with Code Explanation (UI + Backend)

You should include key UI screenshots along with relevant code snippets in your report/presentation. Here's how to structure them:

Login Page (UI)

Code Snippet (HTML/JSP):

```
html

<form action="LoginServlet" method="post">
    <input type="text" name="email" placeholder="Email">
    <input type="password" name="password"
placeholder="Password">
    <button type="submit">Login</button>
</form>
```

Backend (LoginServlet.java):

```
java
CopyEdit
protected void doPost(HttpServletRequest request,
HttpServletResponse response)
throws ServletException, IOException {
    String email = request.getParameter("email");
    String password = request.getParameter("password");

    Connection conn = DBConnection.getConnection();
    PreparedStatement ps = conn.prepareStatement("SELECT * FROM
users WHERE email=? AND password=?");
    ps.setString(1, email);
    ps.setString(2, password);
    ResultSet rs = ps.executeQuery();



if (rs.next()) {
        response.sendRedirect("home.jsp");
    } else {
        response.sendRedirect("login.jsp?error=true");
    }
}
```

Product List Page (UI + Backend)

Screenshot: Display of products with “Add to Cart” buttons.

Code (Servlet/Controller):

```
java
CopyEdit
PreparedStatement ps = conn.prepareStatement("SELECT * FROM
products");
ResultSet rs = ps.executeQuery();
while(rs.next()) {
    out.println("<div>" + rs.getString("name") + " - " +
rs.getDouble("price") + "</div>");
}
```

Explanation: On checkout, data is stored in the `orders` and `order_details` tables.

9. Testing

Proper testing ensures the system functions as expected under various conditions. Both unit and integration testing were performed to validate individual components and their interactions.

9.1 Unit Testing

Each module and function was tested individually:

- **Login validation** tested with correct and incorrect credentials.
- **Add to cart** function tested with various product and quantity combinations.
- **Order placement** tested with valid and invalid inputs.

9.2 Integration Testing

Ensured proper communication between:

- Frontend and backend
- Backend and database
- User module and order module (e.g., checking order history after placing an order)

9.3 Sample Test Cases

Test Case ID	Description	Input	Expected Output	Result
TC001	User Login – Valid	Email + correct password	Redirect to homepage	Pass
TC002	User Login – Invalid	Email + wrong password	Show error message	Pass
TC003	Add to Cart	Product ID + quantity	Item added to cart	Pass
TC004	Place Order	Cart items + payment info	Order confirmed	Pass

10. Result & Discussion

10.1 Features Achieved

- User Registration and Login
- Product browsing with category support
- Add to Cart & Checkout
- Admin: Add/Edit/Delete Products
- Order management and storage

10.2 Challenges Faced

- JDBC connection errors during initial testing
- Handling cart persistence across sessions
- Displaying dynamic data using JSP with backend integration

10.3 Improvements Over Time

- Optimized SQL queries for faster loading
- Separated logic from JSP pages using MVC structure
- Improved form validation using JavaScript

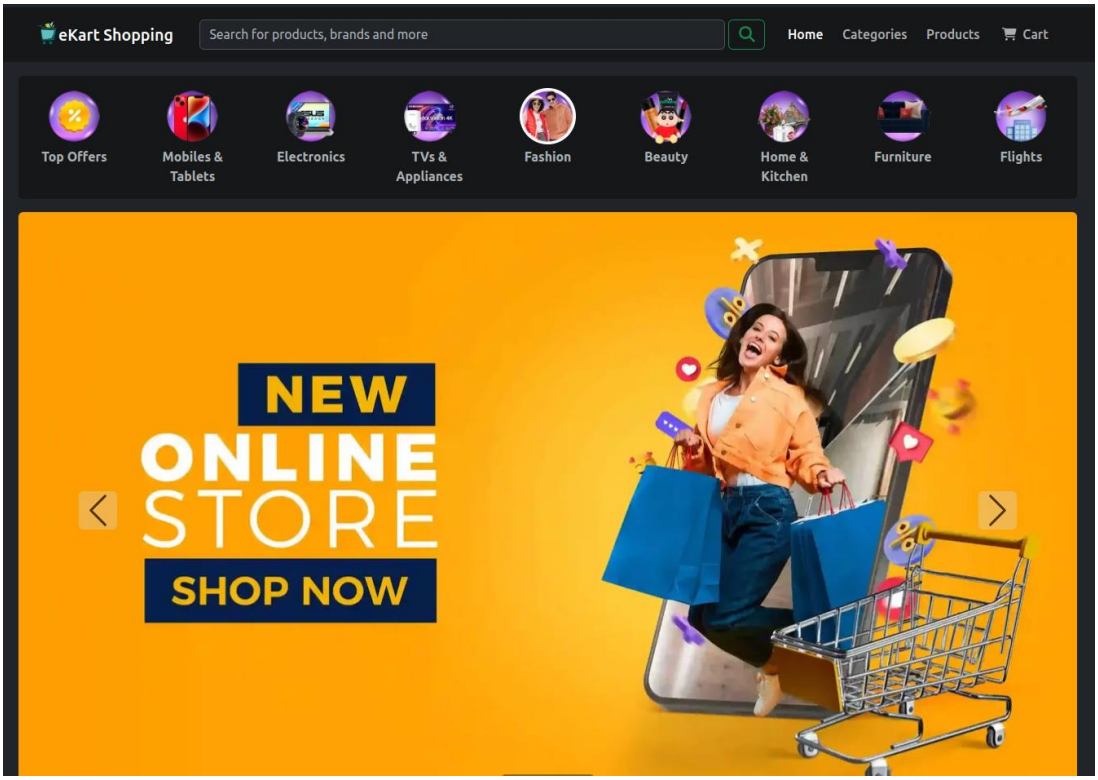
11. Future Scope

To make the system more robust and feature-rich, the following enhancements can be considered:

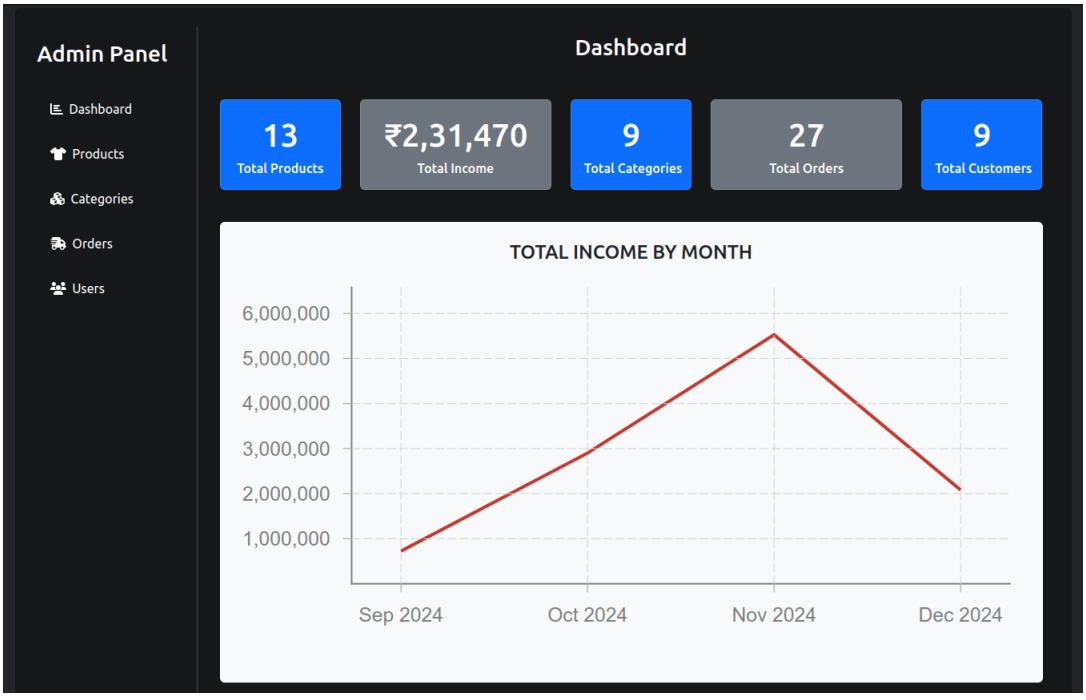
- **Payment Gateway Integration:** Incorporate Razorpay, PayPal, or UPI for real payments.
- **Real-time Order Tracking:** Allow users to view the delivery status of their orders.
- **Mobile App Version:** Create an Android/iOS version for mobile shopping.
- **Wishlist Feature:** Let users save items for later.
- **Product Reviews and Ratings:** Enhance user trust and product selection.

12. Output Screen-shorts

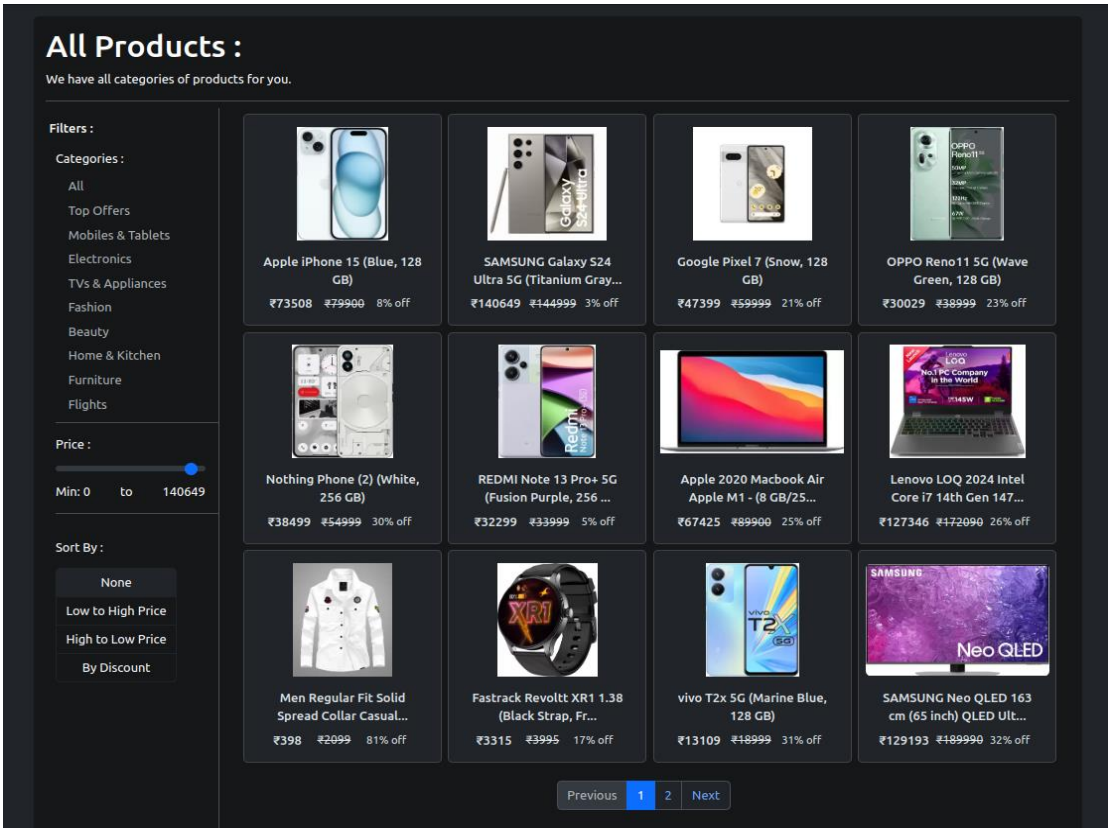
eKart_shopping_home



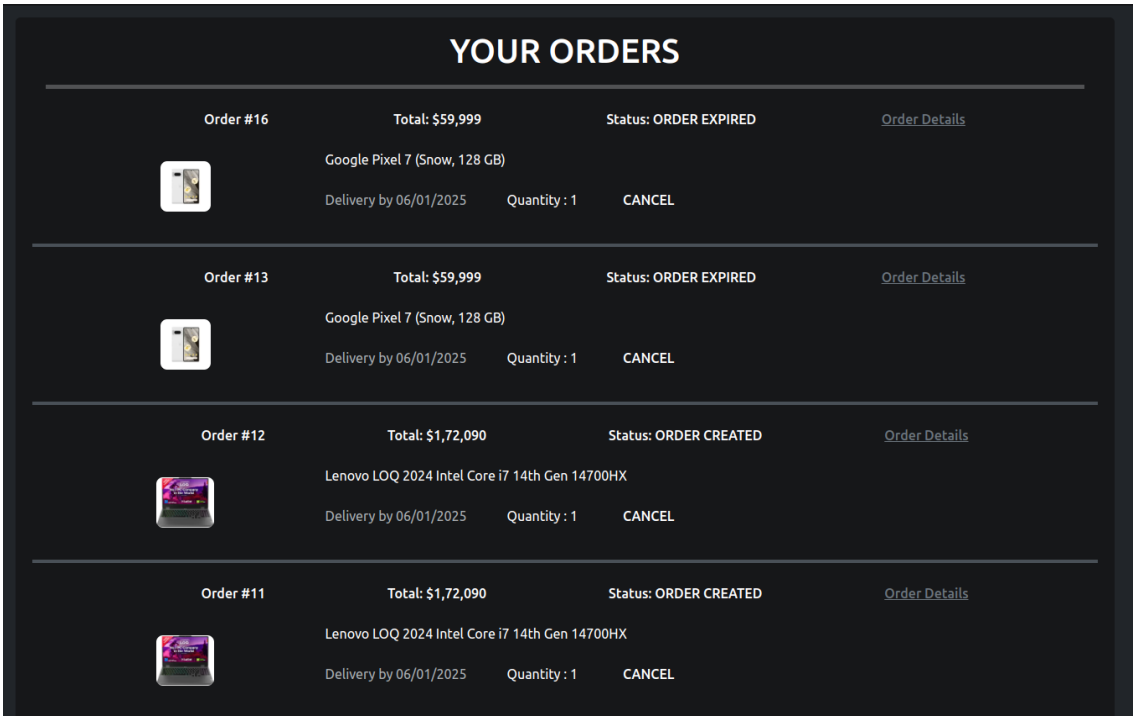
eKart_shopping_admin-panel



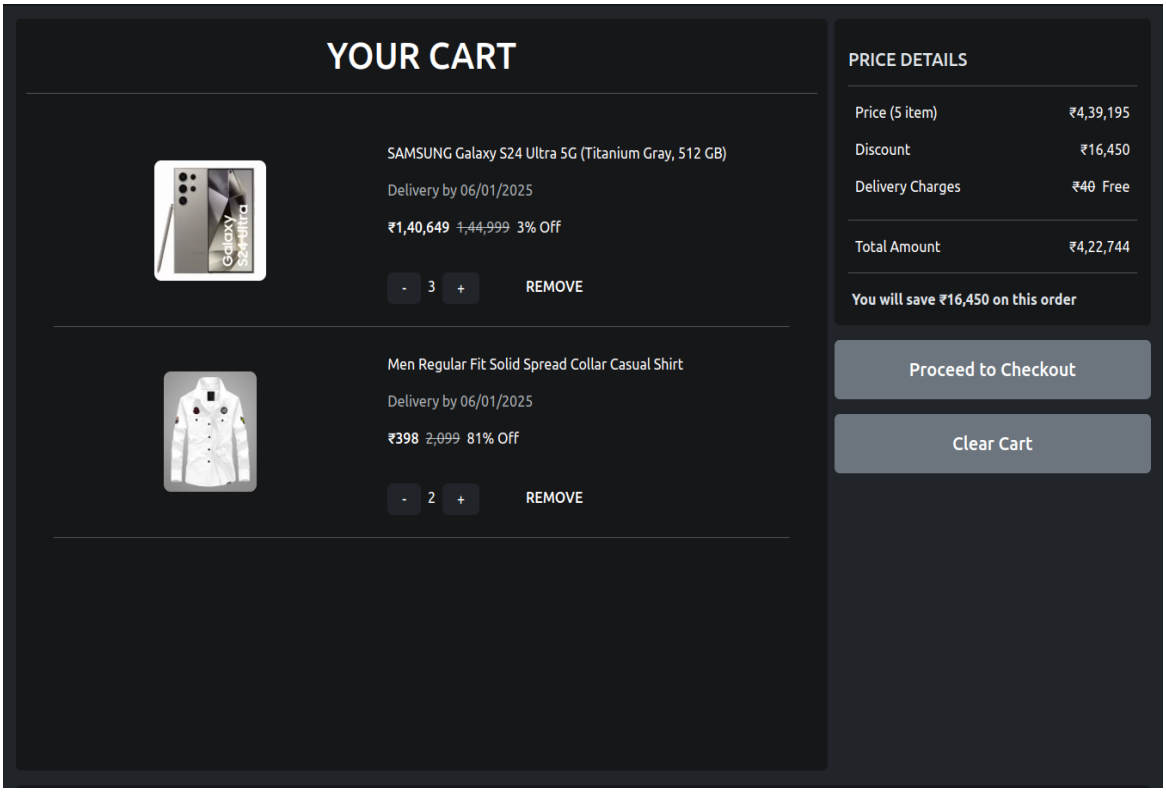
eKart_shopping_all-products



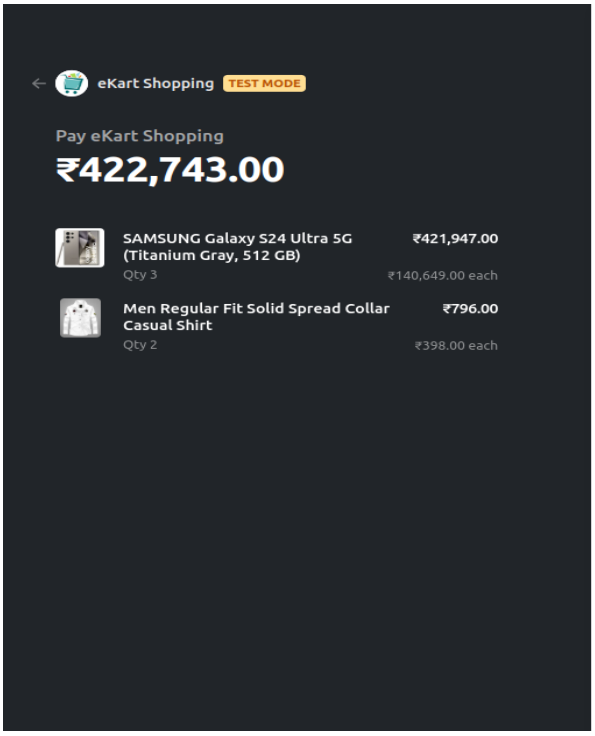
eKart_shopping_orders

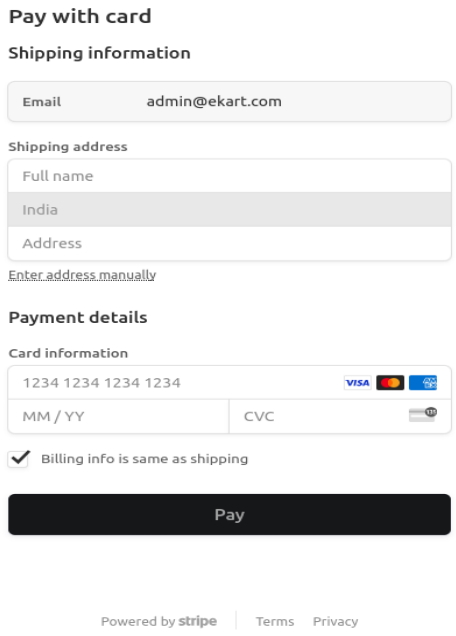


eKart_shopping_cart



eKart_shopping_payment-gateway





ekartdb-backend-ER-diagram

13. Conclusion

The E-cart project successfully simulates the core functionalities of an online shopping system.

It provides a user-friendly interface, secure login, dynamic product listings, and order placement with admin controls.

Through Java, MySQL, and web technologies, the system showcases a complete e-commerce workflow suitable for academic and small business applications.

14. References

1. Oracle Java Documentation – https://docs.oracle.com/javase/
2. MySQL Official Docs – http://1.https//dev.mysql.com/doc/
3. JDBC Tutorial – https://www.javatpoint.com/java-jdbc
4. W3Schools HTML/CSS – http://1.https//www.w3schools.com/
5. GitHub Profile – https://github.com/Yashoda84
6. Stack Overflow – Community posts for debugging and best practices

---

Let me know if you'd like this whole thing compiled into a **Word or PDF report**, or if you need help with generating **cover page, table of contents, or acknowledgement section**.

Java Course Certificate…