E-COMMERCE PLATFORM

A PROJECT REPORT

Submitted by

BHARATH BABU (2303811710421017)

in partial fulfillment of requirements for the award of the course

CGB1201 - JAVA PROGRAMMING

In

COMPUTER SCIENCE AND ENGINEERING

K. RAMAKRISHNAN COLLEGE OF TECHNOLOGY

(An Autonomous Institution, affiliated to Anna University Chennai and Approved by AICTE, New Delhi)

SAMAYAPURAM – 621 112

NOVEMBER- 2024

K. RAMAKRISHNAN COLLEGE OF TECHNOLOGY (AUTONOMOUS)

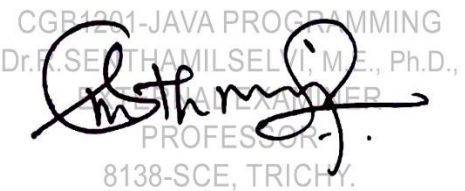
SAMAYAPURAM – 621 112

BONAFIDE CERTIFICATE

Certified that this project report on **"E-COMMERCE PLATFORM"** is the bonafide work of **BHARATH BABU (2303811710421017)** who carried out the project work during the academic year 2024 - 2025 under my supervision.

SIGNATURE	**SIGNATURE**
Dr.A.Delphin Carolina Rani, M.E.,Ph.D.,	Mr. M. Saravanan, M.E.,
HEAD OF THE DEPARTMENT	**SUPERVISOR**
PROFESSOR	ASSISTANT PROFESSOR
Department of CSE	Department of CSE
K.Ramakrishnan College of Technology (Autonomous)	K.Ramakrishnan College of Technology (Autonomous)
Samayapuram–621112.	Samayapuram–621112.

Submitted for the viva-voce examination held on 02.12.2024



INTERNAL EXAMINER EXTERNAL EXAMINER

DECLARATION

I declare that the project report on **"E-COMMERCE PLATFORM"** is the result of original work done by us and best of our knowledge, similar work has not been submitted to **"ANNA UNIVERSITY CHENNAI"** for the requirement of Degree of **BACHELOR OF ENGINEERING**. This project report is submitted on the partial fulfilment of the requirement of the completion of the course **CGB1201 - JAVA PROGRAMMING.**

.

Signature

BHARATH BABU

Place: Samayapuram

Date: 02.12.2024

ACKNOWLEDGEMENT

It is with great pride that I express our gratitude and in-debt to our institution "**K.Ramakrishnan College of Technology (Autonomous)**", for providing us with the opportunity to do this project.

I glad to credit honourable chairman **Dr. K. RAMAKRISHNAN**, **B.E.,** for having provided for the facilities during the course of our study in college.

I would like to express our sincere thanks to our beloved Executive Director **Dr. S. KUPPUSAMY, MBA, Ph.D.,** for forwarding to our project and offering adequate duration in completing our project.

I would like to thank **Dr. N. VASUDEVAN, M.Tech., Ph.D.,** Principal, who gave opportunity to frame the project the full satisfaction.

I whole heartily thanks to **Dr. A. DELPHIN CAROLINA RANI, M.E.,Ph.D.,** Head of the department, **COMPUTER SCIENCE AND ENGINEERING** for providing her encourage pursuing this project.

I express our deep expression and sincere gratitude to our project supervisor **Mr. M. SARAVANAN, M.E.,** Department of **COMPUTER SCIENCE AND ENGINEERING,** for his incalculable suggestions, creativity, assistance and patience which motivated us to carry out this project.

I render our sincere thanks to Course Coordinator and other staff members for providing valuable information during the course.

I wish to express our special thanks to the officials and Lab Technicians of our departments who rendered their help during the period of the work progress.

VISION OF THE INSTITUTION

To serve the society by offering top-notch technical education on par with global standards

MISSION OF THE INSTITUTION

- Be a center of excellence for technical education in emerging technologies by exceeding the needs of the industry and society.
- Be an institute with world class research facilities
- Be an institute nurturing talent and enhancing the competency of students to transform them as all-round personality respecting moral and ethical values

VISION OF DEPARTMENT

To be a center of eminence in creating competent software professionals with research and innovative skills.

MISSION OF DEPARTMENT

M1: Industry Specific: To nurture students in working with various hardware and software platforms inclined with the best practices of industry.

M2: Research: To prepare students for research-oriented activities.

M3: Society: To empower students with the required skills to solve complex technological problems of society.

PROGRAM EDUCATIONAL OBJECTIVES

1. PEO1: Domain Knowledge

To produce graduates who have strong foundation of knowledge and skills in the field of Computer Science and Engineering.

2. PEO2: Employability Skills and Research

To produce graduates who are employable in industries/public sector/research organizations or work as an entrepreneur.

3. PEO3: Ethics and Values

To develop leadership skills and ethically collaborate with society to tackle real-world challenges.

PROGRAM SPECIFIC OUTCOMES (PSOs)

PSO 1: Domain Knowledge

To analyze, design and develop computing solutions by applying foundational concepts of Computer Science and Engineering.

PSO 2: Quality Software

To apply software engineering principles and practices for developing quality software for scientific and business applications.

PSO 3: Innovation Ideas

To adapt to emerging Information and Communication Technologies (ICT) to innovate ideas and solutions to existing/novel problems

PROGRAM OUTCOMES (POs)

Engineering students will be able to:

1. **Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
2. **Problem analysis:** Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences
3. **Design/development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations
4. **Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions

5. **Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations

6. **The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice

7. **Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development

8. **Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

9. **Individual and team work:** Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

10. **Communication:** Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

11. **Project management and finance:** Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

12. **Life-long learning:** Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

ABSTRACT

This project is a Java-based e-commerce application implemented using the Swing framework. It provides an interactive graphical user interface (GUI) that allows users to browse products, add selected items to a shopping cart, and proceed to checkout. The application features a modular design with key functionalities for product management, cart operations, discount application, and order placement.The platform includes a product catalog where users can view details such as product ID, name, and price. A shopping cart feature enables users to add items, view the selected items, and calculate the total cost. A discount system is incorporated, offering a 20% discount when a laptop is purchased along with a small accessory like headphones or a mouse. The GUI includes panels for product display and cart management, with intuitive buttons for actions such as adding to the cart and proceeding to checkout.The application demonstrates essential concepts of object-oriented programming, event-driven programming, and GUI design in Java. It is designed as a scalable and user-friendly foundation for e-commerce platforms, with potential extensions like database integration, user authentication, and advanced discount features.

ABSTRACT WITH POs AND PSOs MAPPING

CO 5 : BUILD JAVA APPLICATIONS FOR SOLVING REAL-TIME PROBLEMS.

ABSTRACT	POs MAPPED	PSOs MAPPED
This project is a Java-based e-commerce application implemented using the Swing framework. It provides an interactive graphical user interface (GUI) that allows users to browse products, add selected items to a shopping cart, and proceed to checkout. The application features a modular design with key functionalities for product management, cart operations, discount application, and order placement.The platform includes a product catalog where users can view details such as product ID, name, and price. A shopping cart feature enables users to add items, view the selected items, and calculate the total cost. A discount system is incorporated, offering a 20% discount when a laptop is purchased along with a small accessory like headphones or a mouse. The GUI includes panels for product display and cart management, with intuitive buttons for actions such as adding to the cart and proceeding to checkout.The application demonstrates essential concepts of object-oriented programming, event-driven programming, and GUI design in Java. It is designed as a scalable and user-friendly foundation for e-commerce platforms, with potential extensions like database integration, user authentication, and advanced discount features.	**PO1 -3** **PO2 -3** **PO3 -3** **PO4 -3** **PO5 -3** **PO6 -3** **PO7 -3** **PO8 -3** **PO9 -3** **PO10 -3** **PO11-3** **PO12 -3**	**PSO1 -3** **PSO2 -3** **PSO3 -3**

Note: 1- Low, 2-Medium, 3- High

TABLE OF CONTENTS

CHAPTER 1
INTRODUCTION

1.1Objective

The objective of this program is to create an interactive E-commerce application using Java Swing that mimics the functionality of an online shopping platform. Users can browse a curated list of products, add selected items to a shopping cart, and proceed to checkout with features like discount eligibility for bundled purchases and multiple payment method options. The application employs object-oriented programming principles, incorporating classes, objects, and custom exceptions to handle invalid product selections and payment errors seamlessly. The GUI design focuses on user-friendliness, offering real-time feedback through confirmation and error pop-ups, as well as recommendations for additional products. This program not only serves as a functional prototype for real-world e-commerce platforms but also highlights key programming skills such as graphical interface design, event handling, and the implementation of business logic for calculating totals, discounts, and order confirmation processes.

1.2Overview

This program is a Java-based E-commerce application designed with a graphical user interface (GUI) using Java Swing. It provides functionalities for users to view a list of products, add items to a shopping cart, and proceed to checkout with an interactive and intuitive interface. The program includes features such as real-time product recommendations, discount eligibility for bundled purchases (e.g., buying a Laptop with a small accessory like a Mouse or Headphones), and a dynamic cart display. It incorporates object-oriented programming principles, using classes to represent products and custom exceptions to handle invalid actions, such as selecting nonexistent products or invalid payment methods. The checkout process allows users to choose from multiple payment options, and the system provides appropriate feedback, including confirmation of successful orders and error messages when necessary. This code demonstrates the integration of GUI design, business logic, and exception handling to create a seamless shopping experience, making it an excellent example of a functional and interactive desktop application.

1.3 Java Programming Concepts

The basic concepts of Object-Oriented Programming (OOP) are:

- **Encapsulation:** Bundling data and methods in a class while restricting access to protect data integrity.
- **Abstraction**: Hiding implementation details and showing only essential features to simplify usage.
- **Inheritance:** Enabling a class to inherit properties and methods from another class, promoting code reuse**.**
- **Polymorphism:** Allowing one interface to represent multiple forms, such as method overriding or overloading.
- **Classes and Objects:** Classes are blueprints for creating objects, which are instances holding specific data and behaviors.
- **Association, Aggregation, and Composition**: Relationships between classes, where objects interact, depend, or exist within one another.

Project related concepts

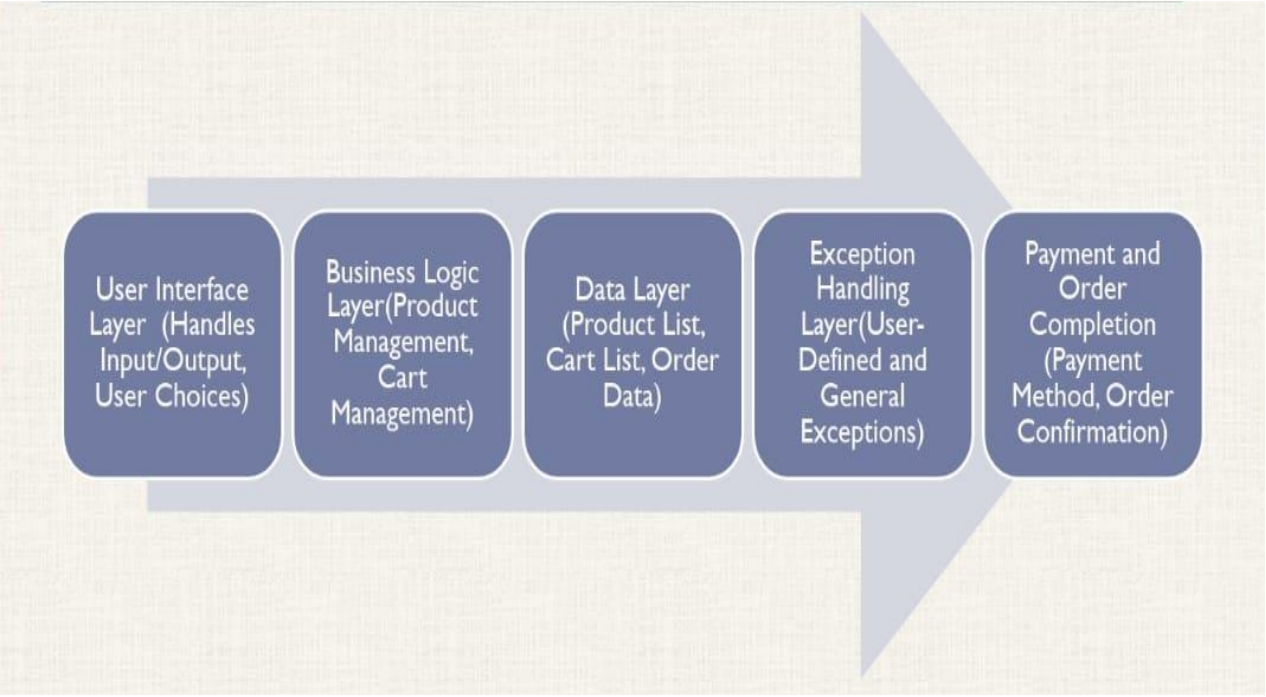
- **Scope and Requirements:** Clearly defining the project's objectives, deliverables, and user requirements to ensure alignment with stakeholder expectations.
- **Planning and Execution:** Creating a detailed roadmap with timelines, resource allocation, and risk management to guide development and achieve milestones effectively.
- **Testing and Deployment**: Verifying the system's functionality, reliability, and usability through rigorous testing before delivering the final product to users.
- **Quality Assurance (QA):** Ensures that the product meets the required standards and is free from defects. QA includes processes for verifying that the software is reliable, secure, and user-friendly.
- **Project Closure:** The final phase of the project, which involves reviewing the project outcomes, documenting lessons learned, and ensuring all deliverables have been met before officially closing the project.

CHAPTER 2
PROJECT METHODOLOGY

2.1Proposed Work

The proposed work for this e-commerce application involves enhancing the user experience by adding more advanced features such as a product search and filter system, enabling users to easily find products based on categories, price range, or ratings. Additionally, integrating real payment gateways like PayPal or Razorpay for secure transactions and implementing a user authentication system to manage user accounts and order history will significantly improve the platform. A multi-language and multi-currency support feature will be added to cater to a global audience. The application will also be optimized for mobile responsiveness to ensure smooth usage on different devices. Further, an admin dashboard for product management and order tracking will be incorporated, alongside the implementation of advanced discount schemes and loyalty programs for frequent buyers. These improvements will make the platform more robust, scalable, and user-friendly.

2.2 Block Diagram



CHAPTER 3
MODULE DESCRIPTION

3.1 Data Model Module

The Data Model Module (represented by the Product class) is essential in this e-commerce application as it defines the structure and attributes of the products available for sale. It holds the fundamental details, such as the product ID, name, and price, ensuring that all product-related data is encapsulated in a standardized format. By using this module, we can efficiently manage and manipulate product information throughout the application. This structure enables the ECommercePlatform class to easily access and display product details, while also facilitating the cart management process. The toString() method in the Product class ensures that each product can be represented clearly in the UI, providing an organized way to handle product data that can be reused and updated as needed.

3.2 Business Logic Module

The Business Logic Module (represented by the ECommercePlatform class) is crucial for managing the core functionality of the e-commerce application. It handles all the essential operations, such as maintaining the product catalog, adding products to the shopping cart, calculating the total price, and applying any applicable discounts. This module encapsulates the business rules of the platform, ensuring that the shopping experience remains consistent and accurate. By separating the logic from the user interface, the ECommercePlatform class ensures that changes to business rules, such as discount calculations or product management, can be easily implemented without affecting the user interface. This separation also promotes reusability, maintainability, and scalability of the application, as the logic is independent of the user-facing components.

3.3 Application Launcher Module

The Application Launcher Module (represented by the Main class) serves as the entry point for the e-commerce application. It is responsible for initializing and setting up the application by creating an instance of the ECommercePlatform class, which contains the product data and business logic. The module then triggers the graphical user interface (GUI) by launching the ECommerceGUI class, passing the platform data to it. This separation ensures that the application's core logic is independent of the user interface, making it easier to manage and update each part. Essentially, the Application Launcher Module ensures that all components of the application are initialized properly before the user can interact with the

system.

3.4 UI Elements Module

The UI Elements Module (comprising Swing components like JFrame, JPanel, JList, JButton, and JScrollPane) is used to create the graphical interface of the e-commerce application, providing a visually interactive platform for the user. These components are essential for displaying the product list, shopping cart, and allowing users to interact with the application through buttons, lists, and other elements. The JFrame serves as the main window, while JPanel organizes content within the window. JList is used to show the available products and the contents of the cart, and JButton allows users to perform actions like adding products to the cart or checking out. The JScrollPane ensures that long lists of products or cart items can be scrolled easily. Using these UI components ensures a user-friendly experience, making the application intuitive and easy to navigate.

3.5 Interaction Module

The Interaction Module (represented by event listeners and action handlers, such as those implemented with ActionListener for buttons) is responsible for capturing and handling user inputs and interactions with the graphical interface. This module ensures that the application responds to user actions, such as selecting products, adding items to the cart, and proceeding to checkout. For example, when a user clicks the "Add to Cart" or "Checkout" button, the corresponding action listeners are triggered, executing the appropriate logic such as updating the cart or displaying a confirmation message. By handling these interactions separately, the Interaction Module allows the application to maintain a clear separation of concerns, ensuring that the UI is responsive and the business logic is processed seamlessly in response to user actions.

3.6 Exception Handling Module

The Exception Handling Module is responsible for managing errors and exceptional cases that may occur during the execution of the application. In this code, it handles situations like invalid product selection (e.g., when a product ID does not exist) and invalid actions such as incorrect user inputs during checkout. For example, when a user tries to add a product to the cart that does not exist, the InvalidProductException is thrown, and when an invalid payment method is selected during checkout, the InvalidPaymentMethodException is triggered. This module ensures that the application remains stable by catching exceptions and providing meaningful feedback to the user, By using custom exceptions,ensuring that any unforeseen issues are addressed gracefully without crashing the application.

CHAPTER 4
CONCLUSION & FUTURE SCOPE

4.1 CONCLUSION

In conclusion, this e-commerce application effectively demonstrates the integration of Object-Oriented Programming principles, modular design, and user-friendly interaction within a real-world shopping platform. By utilizing distinct modules for data management, business logic, UI elements, user interaction, and exception handling, the code ensures a robust and maintainable structure. The use of custom exceptions and event-driven programming enhances the application's reliability and responsiveness, providing users with a smooth shopping experience. Overall, the code exemplifies a well-organized, scalable solution for building interactive applications, while also allowing for easy updates and expansions in the future.

4.2 FUTURE SCOPE

The future scope of this e-commerce application is vast, with several key areas for expansion and enhancement. One major improvement would be integrating a secure user authentication and account management system, allowing users to create profiles, track order history, and receive personalized recommendations based on their preferences and behavior. Incorporating a database (such as MySQL or MongoDB) would allow for dynamic product listings, real-time inventory updates, and efficient order management, making the system more robust and scalable. Additionally, adding support for multiple advanced payment gateways, such as PayPal, Razorpay, and others, would ensure a secure and seamless checkout process. Implementing machine learning algorithms for personalized product recommendations could significantly enhance user engagement. Expanding to mobile platforms by creating native mobile applications or progressive web apps would provide users with a more flexible and convenient shopping experience. Real-time inventory management would ensure product availability is always up-to-date, and integrating features like order tracking, payment history, and refund mechanisms would improve customer satisfaction. Furthermore, refining the UI/UX to be more responsive, interactive, and user-friendly across all devices would help improve the overall shopping experience. These improvements would not only make the application more feature-rich and scalable but also improve its ability to handle a larger user base, ultimately transforming it into a competitive and efficient e-commerce platform.

REFERENCES

Java Books:

1. **"Head First Java" by Kathy Sierra and Bert Bates (O'Reilly Media):** A beginner-friendly introduction to Java programming, covering fundamental concepts.
2. **"Effective Java" by Joshua Bloch (Addison-Wesley):** A more advanced book that offers practical advice on writing robust, maintainable, and efficient Java code.
3. **"Java: The Complete Reference" by Herbert Schildt (McGraw-Hill Education):** A comprehensive guide covering the Java language from basics to advanced topics.

Websites:

1. **Oracle Java Documentation:** The official documentation for the Java language, providing details about core libraries, Java APIs, and JVM-related information.
2. **W3Schools Java Tutorial:** A beginner-friendly site that covers all fundamental Java topics with examples and interactive exercises.
3. **GeeksforGeeks Java Programming:** A comprehensive collection of articles and tutorials on Java programming and algorithms.

YouTube Links:

1. **Java Programming - Programming with Mosh:** A popular YouTube tutorial series for learning Java from the basics.
2. **Java Tutorial for Beginners - FreeCodeCamp:** A detailed video tutorial that covers Java basics, object-oriented programming, and more.
3. **Java Brains:** A YouTube channel offering high-quality tutorials on Java, web development, and frameworks like Spring.

APPENDIX A
(SOURCE CODE)

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
import java.util.ArrayList;
import java.util.List;

class Product {
    int id;
    String name;
    double price;

    Product(int id, String name, double price) {
        this.id = id;
        this.name = name;
        this.price = price;
    }

    @Override
    public String toString() {
        return id + " - " + name + " (₹" + price + ")";
    }
}

class ECommercePlatform {
    List<Product> products = new ArrayList<>();
    List<Product> cart = new ArrayList<>();

    void addProduct(int id, String name, double price) {
        products.add(new Product(id, name, price));
    }
```

```
    List<Product> getProducts() {
        return products;
    }

    void addToCart(int productId) throws Exception {
        Product product = products.stream()
            .filter(p -> p.id == productId)
            .findFirst()
            .orElseThrow(() -> new Exception("Product with ID " + productId + " not found."));
        cart.add(product);
    }

    List<Product> getCart() {
        return cart;
    }

    double calculateTotal() {
        return cart.stream().mapToDouble(p -> p.price).sum();
    }

    String checkDiscount() {
        boolean hasLaptop = cart.stream().anyMatch(p -> p.name.equalsIgnoreCase("Laptop"));
        boolean hasSmallProduct = cart.stream().anyMatch(p -> p.name.equalsIgnoreCase("Headphones") ||
p.name.equalsIgnoreCase("Mouse"));

        if (hasLaptop && hasSmallProduct) {
            return "You get a 20% discount for buying a Laptop with one small product.";
        }
        return null;
    }

    double applyDiscount(double total) {
        boolean hasLaptop = cart.stream().anyMatch(p -> p.name.equalsIgnoreCase("Laptop"));
```

```
        boolean hasSmallProduct = cart.stream().anyMatch(p -> p.name.equalsIgnoreCase("Headphones") ||
p.name.equalsIgnoreCase("Mouse"));

        if (hasLaptop && hasSmallProduct) {
            return total * 0.8;
        }
        return total;
    }

    void clearCart() {
        cart.clear();
    }
}

public class Main {
    public static void main(String[] args) {
        ECommercePlatform platform = new ECommercePlatform();

        // Adding products
        platform.addProduct(1, "Laptop", 65999.0);
        platform.addProduct(2, "Smartphone", 39999.0);
        platform.addProduct(3, "Headphones", 2499.0);
        platform.addProduct(4, "Tablet", 24999.0);
        platform.addProduct(5, "Smartwatch", 15999.0);
        platform.addProduct(6, "Gaming Console", 32999.0);
        platform.addProduct(7, "Bluetooth Speaker", 3999.0);
        platform.addProduct(8, "Monitor", 12499.0);
        platform.addProduct(9, "Keyboard", 3299.0);
        platform.addProduct(10, "Mouse", 1599.0);

        SwingUtilities.invokeLater(() -> new Main().initializeUI(platform));
    }

    private void initializeUI(ECommercePlatform platform) {
```

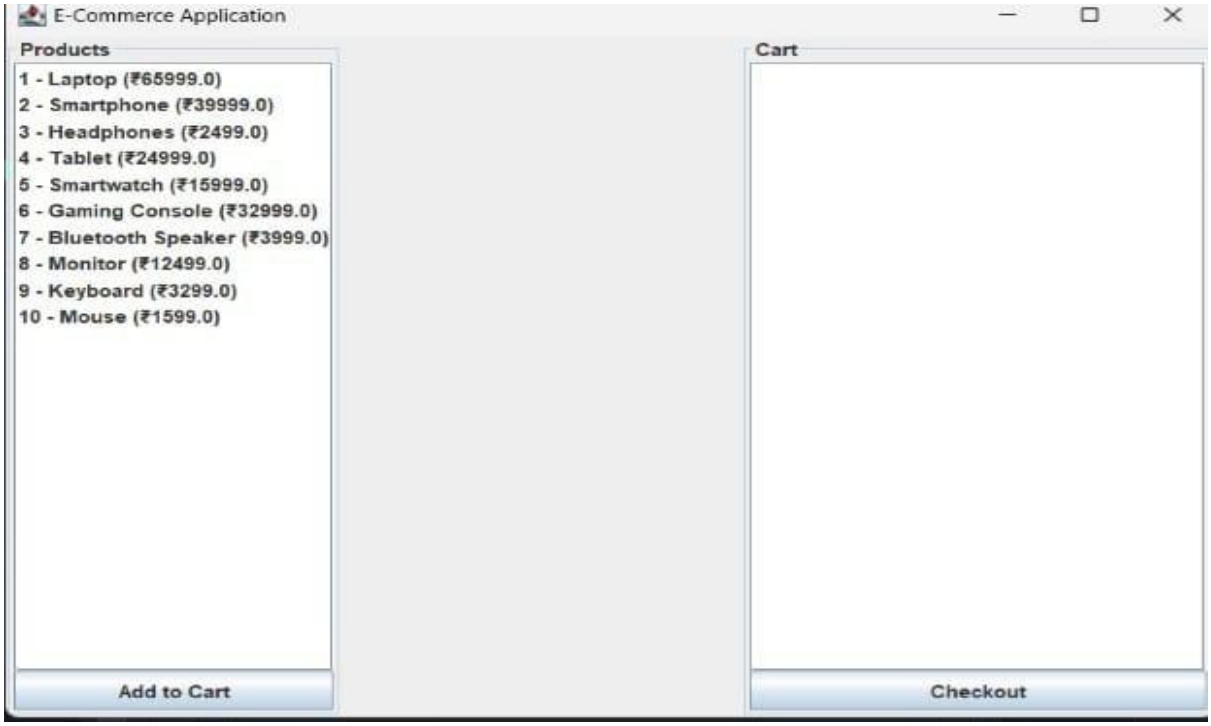
```
JFrame frame = new JFrame("E-Commerce Application");
frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
frame.setSize(700, 500);
frame.setLayout(new BorderLayout());

// Product list panel
JPanel productPanel = new JPanel(new BorderLayout());
productPanel.setBorder(BorderFactory.createTitledBorder("Products"));

DefaultListModel<String> productListModel = new DefaultListModel<>();
JList<String> productList = new JList<>(productListModel);
platform.getProducts().forEach(p -> productListModel.addElement(p.toString()));

JScrollPane productScrollPane = new JScrollPane(productList);
productPanel.add(productScrollPane, BorderLayout.CENTER);

JButton addToCartButton = new JButton("Add to Cart");
productPanel.add(addToCartButton, BorderLayout.SOUTH);

// Cart panel
JPanel cartPanel = new JPanel(new BorderLayout());
cartPanel.setBorder(BorderFactory.createTitledBorder("Cart"));

DefaultListModel<String> cartListModel = new DefaultListModel<>();
JList<String> cartList = new JList<>(cartListModel);
JScrollPane cartScrollPane = new JScrollPane(cartList);
cartPanel.add(cartScrollPane, BorderLayout.CENTER);

JButton checkoutButton = new JButton("Checkout");
cartPanel.add(checkoutButton, BorderLayout.SOUTH);

// Add panels to frame
frame.add(productPanel, BorderLayout.WEST);
frame.add(cartPanel, BorderLayout.EAST);
```

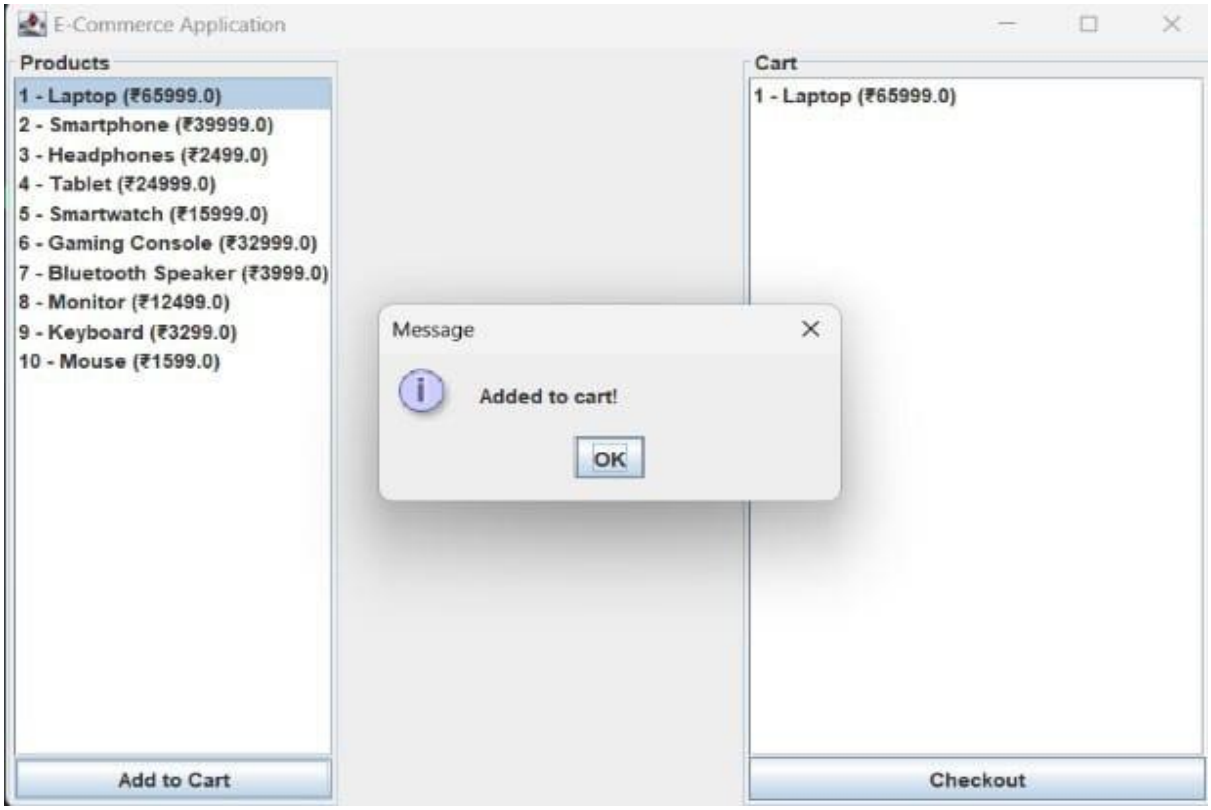
```
        // Event handlers
        addToCartButton.addActionListener(e -> {
            String selected = productList.getSelectedValue();
            if (selected != null) {
                int productId = Integer.parseInt(selected.split(" - ")[0]);
                try {
                    platform.addToCart(productId);
                    cartListModel.addElement(selected);
                    JOptionPane.showMessageDialog(frame, "Added to cart!");
                } catch (Exception ex) {
                    JOptionPane.showMessageDialog(frame, ex.getMessage(), "Error",
JOptionPane.ERROR_MESSAGE);
                }
            } else {
                JOptionPane.showMessageDialog(frame, "Please select a product to add!", "Warning",
JOptionPane.WARNING_MESSAGE);
            }
        });

        checkoutButton.addActionListener(e -> {
            if (cartListModel.isEmpty()) {
                JOptionPane.showMessageDialog(frame, "Your cart is empty!", "Warning",
JOptionPane.WARNING_MESSAGE);
                return;
            }

            double total = platform.calculateTotal();
            String discountMessage = platform.checkDiscount();
            if (discountMessage != null) {
                JOptionPane.showMessageDialog(frame, discountMessage, "Discount",
JOptionPane.INFORMATION_MESSAGE);
                total = platform.applyDiscount(total);
            }
```

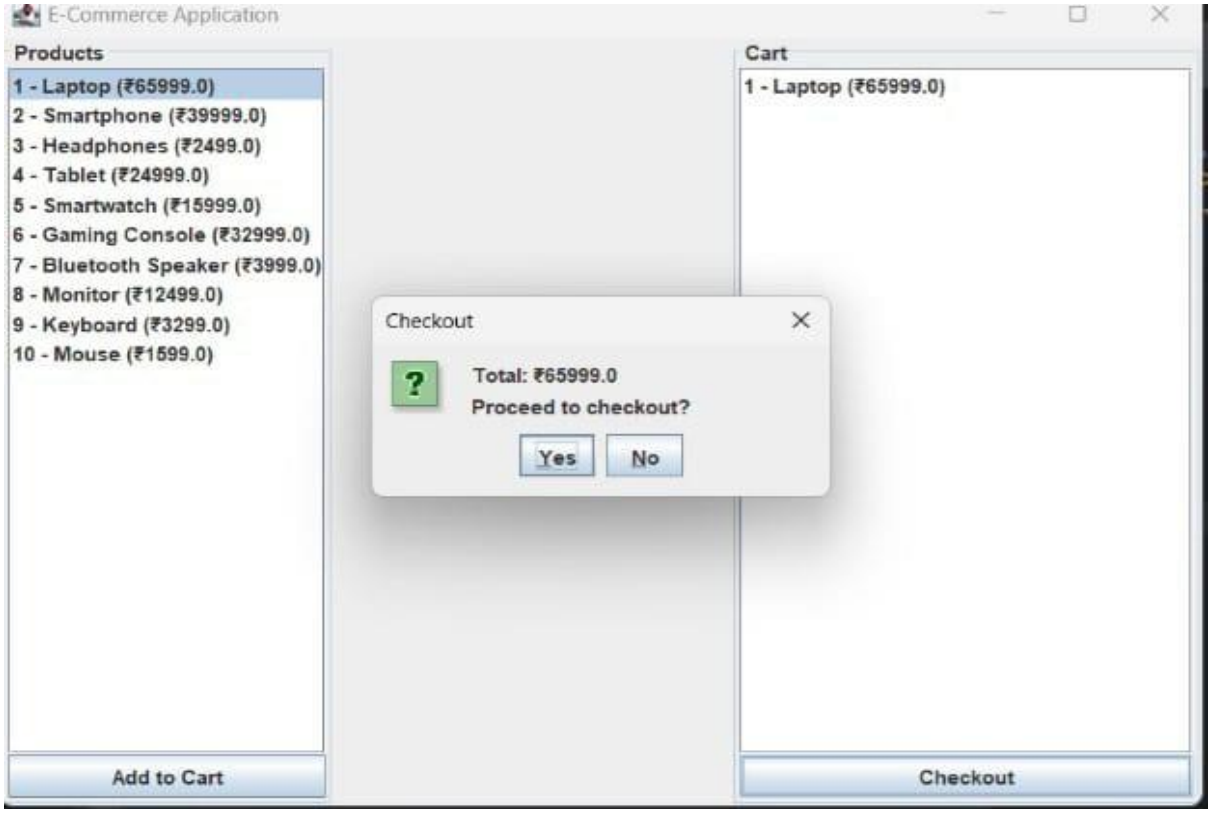
```
        int result = JOptionPane.showConfirmDialog(frame, "Total: ₹" + total + "\nProceed to checkout?",
"Checkout", JOptionPane.YES_NO_OPTION);
        if (result == JOptionPane.YES_OPTION) {
            JOptionPane.showMessageDialog(frame, "Order placed successfully!\nThank you for shopping with
us!", "Success", JOptionPane.INFORMATION_MESSAGE);
            platform.clearCart();
            cartListModel.clear();
        }
    });

    frame.setVisible(true);
  }
}
```

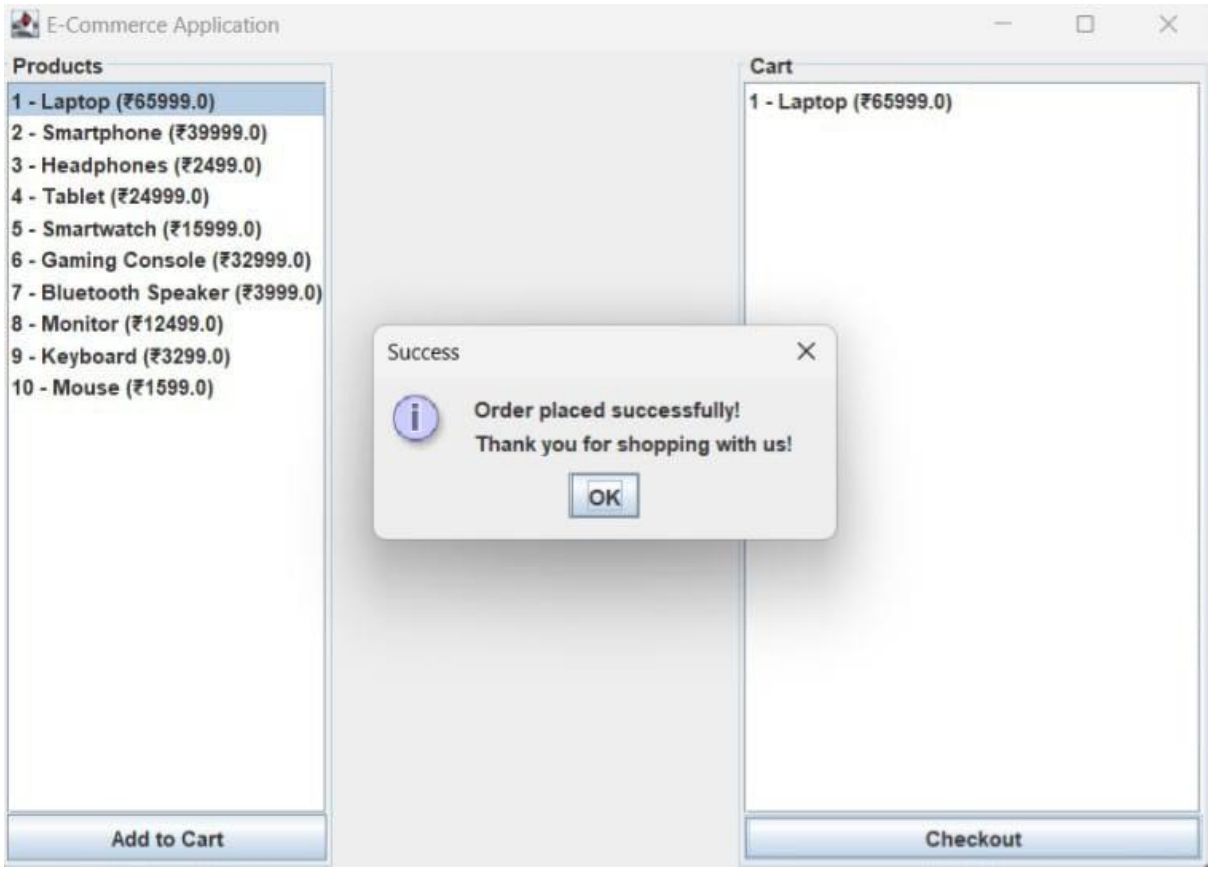
APPENDIX B

(SCREENSHOTS)






E-Commerce Application
Products
1 - Laptop (₹65999.0)
2 - Smartphone (₹39999.0)
3 - Headphones (₹2499.0)
4 - Tablet (₹24999.0)
5 - Smartwatch (₹15999.0)
6 - Gaming Console (₹32999.0)
7 - Bluetooth Speaker (₹3999.0)
8 - Monitor (₹12499.0)
9 - Keyboard (₹3299.0)
10 - Mouse (₹1599.0)
Add to Cart
Cart
1 - Laptop (₹65999.0)
Checkout
Checkout
Total: ₹65999.0
Proceed to checkout?
Yes
No


E-Commerce Application
Products
1 - Laptop (₹65999.0)
2 - Smartphone (₹39999.0)
3 - Headphones (₹2499.0)
4 - Tablet (₹24999.0)
5 - Smartwatch (₹15999.0)
6 - Gaming Console (₹32999.0)
7 - Bluetooth Speaker (₹3999.0)
8 - Monitor (₹12499.0)
9 - Keyboard (₹3299.0)
10 - Mouse (₹1599.0)
Add to Cart
Cart
1 - Laptop (₹65999.0)
Checkout
Success
Order placed successfully!
Thank you for shopping with us!
OK