E-Commerce Store

A MINI-PROJECT REPORT

Submitted by

Akhil Kumaran S 240701026

Aldrine Linjoe S 240701033

in partial fulfillment of the award of the degree

of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

RAJALAKSHMI ENGINEERING COLLEGE, CHENNAI

An Autonomous Institute

CHENNAI NOVEMBER 2025

BONAFIDE CERTIFICATE

Certified that this project **"E-COMMERCE PRODUCT AND CATALOUGE**

E-Commerce Store" is the Bonafide work of

"Akhil Kumaran S, Aldrine linjoe" who carried out the project work under my supervision.

SIGNATURE

Deepa B

ASSISTANT PROFESSOR SG

Dept. of Computer Science and Eng,

Rajalakshmi Engineering College

Chennai

This mini project report is submitted for the viva voce examination to be held on _____

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ABSTRACT

The **"E-Commerce Desktop Application"** is a Java-based project built to simulate the core functionality of a retail e-commerce platform using a graphical user interface (GUI set as Java application's UI). The overriding aim of the system is to provide an intuitive and responsive buying-selling experience using desktop-based Java Swing technology. The application supports two primary users: **Admin** for product and inventory management, and **Customer** for browsing products, managing cart items, and placing orders with simple checkout management.

This system supports user authentication, product browsing, cart operations, order placement, and secure interaction between UI and persistent storage. It extensively incorporates Java Database Connectivity (JDBC) to perform **CRUD operations** (Create, Read, Update, Delete) with a MySQL backend. User data, product listings, and purchase history are stored in normalized database tables, ensuring reliability and scalability.

The project successfully demonstrates an end-to-end lifecycle of a minimal online store through a desktop model, which makes it suitable for small-scale or offline retail environments. Additionally, the application’s modular architecture supports future enhancements such as payment gateway integration, image-based product UI, and invoice generation.

ACKNOWLEDGEMENT

We express our sincere thanks to our beloved and honorable chairman **MR. S. MEGANATHAN** and the chairperson DR. **M. THANGAM MEGANATHAN** for their timely support and encouragement.

We are greatly indebted to our respected and honorable principal **Dr. S.N. MURUGESAN** for his able support and guidance.

No words of gratitude will suffice for the unquestioning support extended to us by our Head of The Department **Dr. E.M. MALATHY** and our Deputy Head of The Department **Dr. J. MANORANJINI for** being ever supporting force during our project work

We also extend our sincere and hearty thanks to our internal guide **Deepa B,** for her valuable guidance and motivation during the completion of this project.

Our sincere thanks to our family members, friends and other staff members of computer science engineering.

1. AKHIL KUMARAN S

2. ALDRINE LINJOE S

TABLE OF CONTENTS

CHAPTER NO. TITLE PAGE NO

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION

The growth of e-commerce has transformed the retail industry, giving rise to highly efficient systems for inventory management, product display, and user interaction. This project, titled **"E-Commerce Desktop Application using Java Swing"**, provides a basic yet extendable version of a desktop shopping system where users can browse products in a visual UI, interact with a graphical shopping cart, and place orders. Meanwhile, the administrator can log in to manage products (add, delete, update, view). The application is built using **Java Swing** for the user interface and **MySQL** for database management, connected via **JDBC** API — ensuring seamless data interaction between client and databases..

1.2 SCOPE OF THE WORK

The scope of this system includes:

- **Customer Module**: Register and login securely, view products, filter by category, add items to cart, and simulate checkout.
- **Admin Module**: Secure login access, product management (CRUD), inventory updates, and catalog control.
- **Database Integration**: Stable connectivity with MySQL using JDBC for all backend functionality (users, products, and cart).
- **User Interface**: Java Swing-based desktop GUI supporting form-based login, registration, shopping, and product control.

This scope is particularly ideal for college-level Java projects and reflects real-world system behavior.

1.3 PROBLEM STATEMENT

Modern businesses face limitations when relying on manual retail management, leading to poor customer experience, inefficient sales tracking, and slow inventory updates. Traditional systems lack a way for small merchants or shop owners to computerize their sales process. This project solves the problem by implementing an automated, user-friendly digital interface to simulate an authentic e-commerce environment using basic computing resources.

1.4 AIM AND OBJECTIVES OF THE PROJECT

Aim:

The primary aim is to design and implement a fully functional e-commerce desktop application using Java Swing and MySQL that supports real-time product browsing, cart interaction, and admin-level product management.

Objectives:

☑ To implement secure authentication for both customer and admin roles

☑ To integrate a GUI-based product list, cart system, and product editor

☑ To implement database operations via JDBC for persistent data storage

☑ To design a shopping-like interface with accurate stock deductions

☑ To create a scalable modular architecture ready for feature extension like invoicing, reporting or image uploads

CHAPTER 2

SYSTEM SPECIFICATIONS

2.1 HARDWARE SPECIFICATIONS

Component	Detailed Specification
Processor	2.0 GHz Dual-Core or higher
RAM	4 GB DDR3/DDR4 Minimum
Storage	100 MB free disk space
Network	Stable Internet or LAN connection

2.2 SOFTWARE SPECIFICATIONS (Updated for MySQL)

Component	Detailed Specification
Operating System	Windows, macOS, or Linux
Programming Language	Java SE Development Kit (JDK) 17+
GUI Library	java. swing

Database Management	MySQL Server (v8.0 or later)
Connector	MySQL Connector/J (JDBC Driver)
Database Utility	XAMPP, WAMP, or MySQL Workbench

CHAPTER 3

MODULE DESCRIPTION

The application is structured into four primary modules: Data Model, Data Access Layer (DAO), Database Utility, and User Interfaces (UI).

3.1 DATA MODEL MODULE

- **Product.java**: Acts as the blueprint for every item in the catalog. It encapsulates data fields (id, name, price, stock, category, image Path) and includes constructors, getters, and setters for data manipulation.

3.2 DATA ACCESS LAYER (DAO) MODULE

This module contains the core business logic for database interaction.

- **ProductDAO.java**: Manages all interactions with the products table.
  - **Methods Implemented:** add Product, getAllProducts, getProductById, update Product, delete Product.

- **Critical Stock Management:** Includes the reduce Stock (int productid, int quantity) method, which is transactional and crucial for preventing overselling.
- **Filtering:** Includes get Categories () and getProductsFiltered (String category, String search Term) to support dynamic catalog browsing.

3.3 DATABASE UTILITY MODULE

- **DBConnection.java**: A singleton class responsible for managing the SQLite connection.
  - **Core Function:** Provides a static method get Connection() to obtain a live database connection.
  - **Initialization:** Contains initialize Database(), which runs on application start to create the products and cart items tables and insert initial sample data.

3.4 USER INTERFACE(UI) MODULE

The UI Module is built using **Java Swing** and comprises three main frames that manage the user experience and interaction.

1. **Welcome Frame**: The system entry point that initializes the database and directs users to either the Customer or Admin access points.

2. **Product Frame (Customer View)**: This is the main catalog, displaying products in a dynamic Jtable. It features a custom **Image Renderer** for visuals and includes **filtering** (by category) and **search** functionality. Crucially, it uses a custom Button Editor to handle "Add to Cart" actions while reflecting real-time stock status (e.g., showing "Out of Stock").

3. **Admin Frame (Management View)**: This secure panel is dedicated to inventory control. It allows administrators to select products from the table and perform **CRUD** (Create, Read, Update, Delete) operations via dedicated input fields and action buttons, ensuring data synchronization with the MySQL database.

CHAPTER 4

SAMPLE CODING

DBCONNECTION CODE:

```
package com.ecommerce;

import javax.swing.*;

public class ECommerceApp {
    public static final String DB_URL = "jdbc:mysql://localhost:3306/shop_db";
    public static final String DB_USER = "root";
    public static final String DB_PASSWORD = "Akhil@13";

    public static int currentUserId = -1;
    public static String currentUsername = "";
    public static String currentUserRole = "";

    public static void main(String[] args) {
        SwingUtilities.invokeLater(() -> new LoginPage());
    }

    public static Connection getConnection() throws SQLException {
        try {
            Class.forName("com.mysql.cj.jdbc.Driver");
            return DriverManager.getConnection(DB_URL, DB_USER, DB_PASSWORD);
        } catch (ClassNotFoundException e) {
            throw new SQLException("MySQL JDBC Driver not found");
        }
    }
```

WELCOMEFRAME CODE:

```
package com.ecommerce;

import javax.swing.*;
import java.awt.*;
import java.sql.*;

public class RegisterPage extends
JFrame {
    private JTextField
usernameField, emailField;
    private JPasswordField
passwordField,
confirmPasswordField;

    public RegisterPage() {
        setTitle("User
Registration");
        setSize(450, 450);

setDefaultCloseOperation(DISPOSE_ON
_CLOSE);

setLocationRelativeTo(null);

        JPanel mainPanel = new
JPanel(new BorderLayout(10, 10));

mainPanel.setBorder(BorderFactory.c
reateEmptyBorder(20, 30, 20, 30));
        mainPanel.setBackground(new
Color(240, 248, 255));

        // Title Panel
        JPanel titlePanel = new
JPanel();

titlePanel.setBackground(new
Color(240, 248, 255));
        JLabel titleLabel = new
JLabel("📝 Create New Account",
SwingConstants.CENTER);
        titleLabel.setFont(new
Font("Arial", Font.BOLD, 26));

titleLabel.setForeground(new
Color(25, 25, 112));
        titlePanel.add(titleLabel);
        mainPanel.add(titlePanel,
BorderLayout.NORTH);

        // Form Panel
        JPanel formPanel = new
JPanel(new GridLayout(5, 2, 10,
15));
```

```
        formPanel.setBackground(new
Color(240, 248, 255));

formPanel.setBorder(BorderFactory.c
reateEmptyBorder(20, 10, 20, 10));

        // Username
        JLabel userLabel = new
JLabel("Username:");
        userLabel.setFont(new
Font("Arial", Font.BOLD, 14));
        formPanel.add(userLabel);

        usernameField = new
JTextField();
        usernameField.setFont(new
Font("Arial", Font.PLAIN, 14));

formPanel.add(usernameField);

        // Email
        JLabel emailLabel = new
JLabel("Email:");
        emailLabel.setFont(new
Font("Arial", Font.BOLD, 14));
        formPanel.add(emailLabel);

        emailField = new
JTextField();
        emailField.setFont(new
Font("Arial", Font.PLAIN, 14));
        formPanel.add(emailField);

        // Password
        JLabel passLabel = new
JLabel("Password:");
        passLabel.setFont(new
Font("Arial", Font.BOLD, 14));
        formPanel.add(passLabel);

        passwordField = new
JPasswordField();
        passwordField.setFont(new
Font("Arial", Font.PLAIN, 14));

formPanel.add(passwordField);

        // Confirm Password
        JLabel confirmLabel = new
JLabel("Confirm Password:");
        confirmLabel.setFont(new
Font("Arial", Font.BOLD, 14));

formPanel.add(confirmLabel);
```

```
        confirmPasswordField = new
JPasswordField();

confirmPasswordField.setFont(new
Font("Arial", Font.PLAIN, 14));

formPanel.add(confirmPasswordField)
;

        // Info Label
        JLabel infoLabel = new
JLabel("(New users will be
registered as 'User' role)");
        infoLabel.setFont(new
Font("Arial", Font.ITALIC, 11));

infoLabel.setForeground(Color.GRAY)
;
        formPanel.add(infoLabel);
        formPanel.add(new
JLabel(""));

        mainPanel.add(formPanel,
BorderLayout.CENTER);

        // Button Panel
        JPanel buttonPanel = new
JPanel(new
FlowLayout(FlowLayout.CENTER, 15,
10));

buttonPanel.setBackground(new
Color(240, 248, 255));

        JButton registerBtn = new
JButton("Register");
        registerBtn.setFont(new
Font("Arial", Font.BOLD, 15));

registerBtn.setBackground(new
Color(144, 238, 144));

registerBtn.setForeground(Color.BLA
CK);

registerBtn.setPreferredSize(new
Dimension(140, 45));

registerBtn.setFocusPainted(false);

registerBtn.addActionListener(e ->
register());

        JButton cancelBtn = new
JButton("Cancel");
        cancelBtn.setFont(new
Font("Arial", Font.BOLD, 15));
```

```
        cancelBtn.setBackground(new
Color(255, 182, 193));

cancelBtn.setForeground(Color.BLACK
);

cancelBtn.setPreferredSize(new
Dimension(140, 45));

cancelBtn.setFocusPainted(false);

cancelBtn.addActionListener(e ->
dispose());


buttonPanel.add(registerBtn);
        buttonPanel.add(cancelBtn);

        mainPanel.add(buttonPanel,
BorderLayout.SOUTH);

        add(mainPanel);
        setVisible(true);
    }

    private void register() {
        String username =
usernameField.getText().trim();
        String email =
emailField.getText().trim();
        String password = new
String(passwordField.getPassword())
;
        String confirmPassword =
new
String(confirmPasswordField.getPass
word());

        // Validation
        if (username.isEmpty() ||
email.isEmpty() ||
password.isEmpty() ||
confirmPassword.isEmpty()) {

JOptionPane.showMessageDialog(this,
                "⚠ Please fill
all fields!",
                "Missing
Information",

JOptionPane.WARNING_MESSAGE);
            return;
        }

        if (username.length() < 3)
{
```

```
JOptionPane.showMessageDialog(this,
                "Username must be
at least 3 characters long!",
                "Invalid Username",

JOptionPane.WARNING_MESSAGE);
            return;
        }

        if (password.length() < 6)
{

JOptionPane.showMessageDialog(this,
                "Password must be
at least 6 characters long!",
                "Weak Password",

JOptionPane.WARNING_MESSAGE);
            return;
        }

        if
(!password.equals(confirmPassword))
{

JOptionPane.showMessageDialog(this,
                "❌ Passwords
don't match!\nPlease re-enter your
password.",
                "Password
Mismatch",

JOptionPane.ERROR_MESSAGE);
            return;
        }

        // Register user in
database
        try (Connection conn =
ECommerceApp.getConnection()) {
            String query = "INSERT
INTO users (username, password,
role) VALUES (?, ?, 'user')";
            PreparedStatement pst =
conn.prepareStatement(query);
            pst.setString(1,
username);
            pst.setString(2,
password);

            pst.executeUpdate();


JOptionPane.showMessageDialog(this,
```

```
                "☑ Registration
Successful!\n\n" +
                "Username: " +
username + "\n" +
                "Role: User\n\n" +
                "You can now login
with your credentials.",
                "Success",

JOptionPane.INFORMATION_MESSAGE);

            dispose();

        } catch (SQLException ex) {
            if
(ex.getMessage().contains("Duplicat
e entry")) {

JOptionPane.showMessageDialog(this,
                    "✖ Username
already exists!\n\nPlease choose a
different username.",
                    "Registration
Failed",

JOptionPane.ERROR_MESSAGE);
            } else {

JOptionPane.showMessageDialog(this,
                    "⚠ Database
Error!\n\n" + ex.getMessage(),
                    "Error",

JOptionPane.ERROR_MESSAGE);
            }
        }
    }
}}
```

CHAPTER 5

SCREEN SHOTS

Fig 5.1 WELCOME PAGE

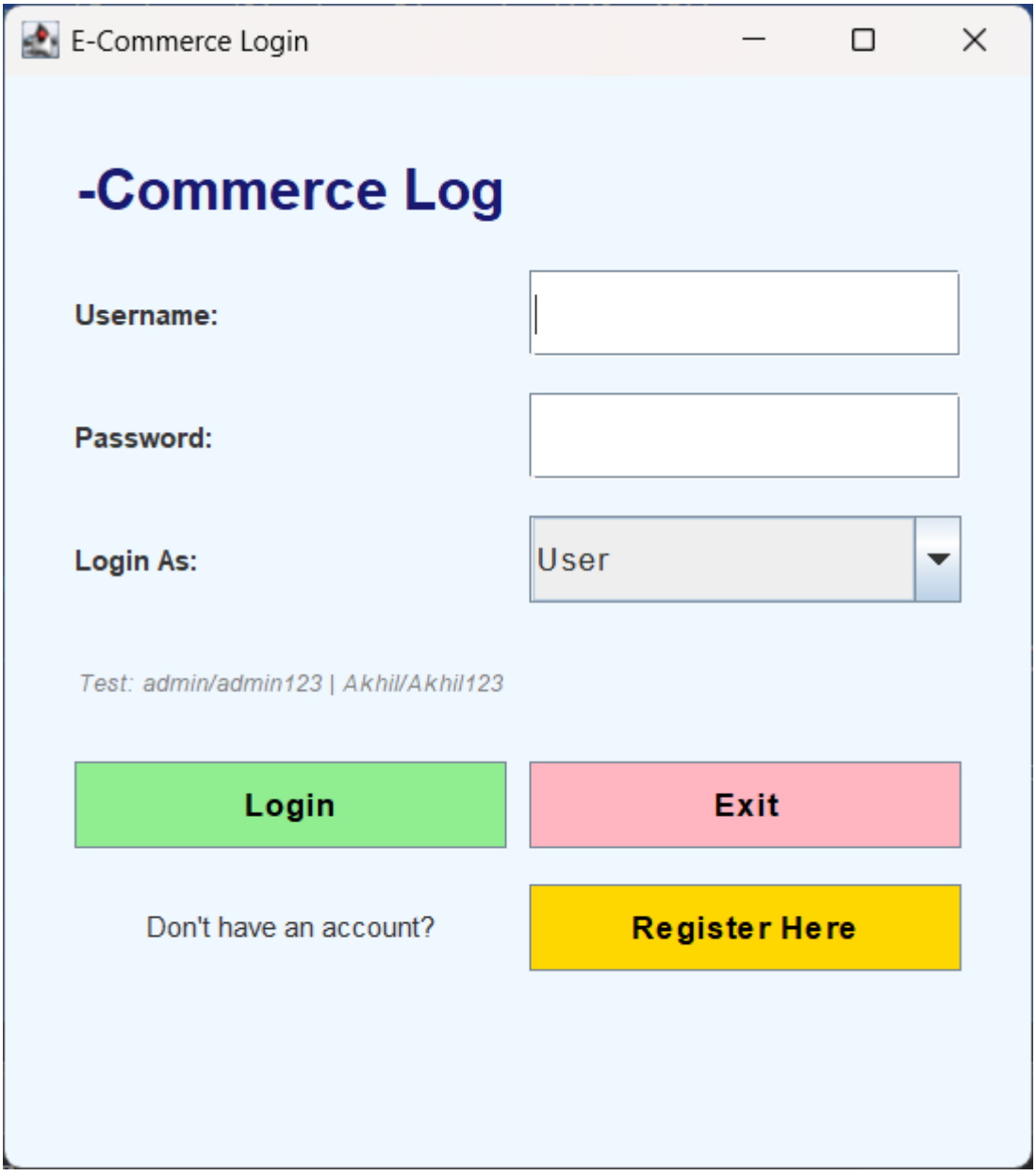

Fig 5.3 ADMIN LOGIN

Fig 5.4 PRODUCT FRAME

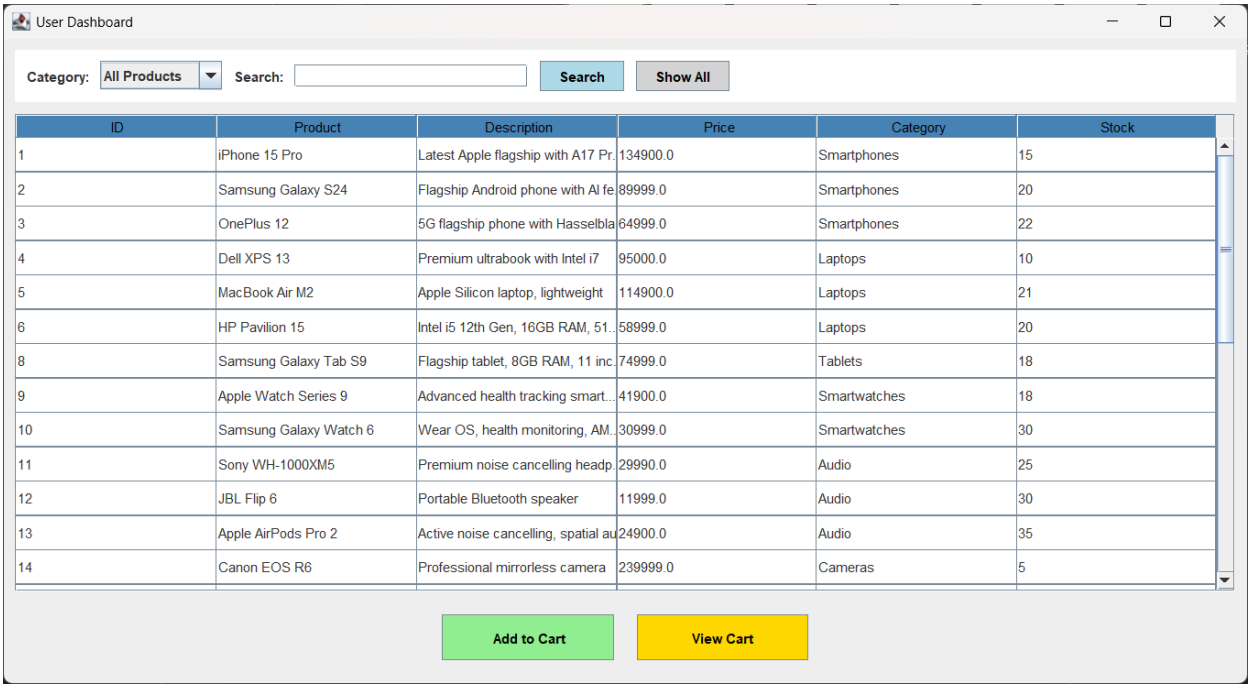

Fig 5.5 CART FRAME

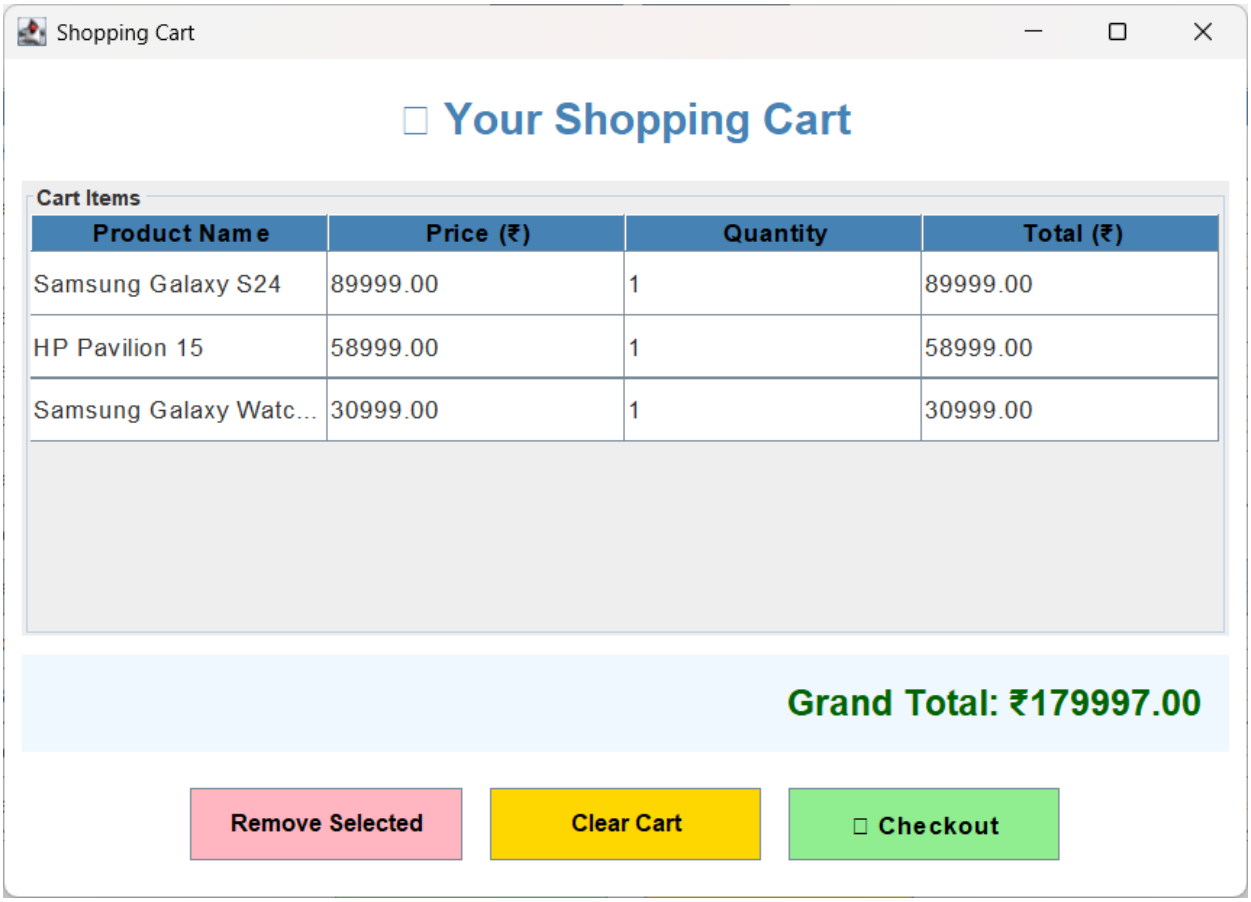

Fig 5.6 CHECKOUT FRAME

CHAPTER 6

CONCLUSION AND FUTURE ENHANCEMENT

CONCLUSION

The E-Commerce Desktop Application built using **Java Swing** and **MySQL (JAVA-JDBC)** successfully fulfills the core requirements of a basic digital retail solution. The system offers two distinct user interfaces — a **customer module** for browsing and shopping, and an **admin module** for complete product management.

By integrating GUI design principles with MySQL-backed JDBC connectivity, the application ensures smooth interaction between frontend components and backend data operations. Features such as login and registration, cart management simulation, and robust CRUD-based inventory control contribute to a seamless e-commerce experience.

The project successfully demonstrates principles of modular design, data persistence, and event-driven programming. It is highly suitable for a standalone environment such as college projects, small shops, or offline demos, and provides a solid foundation for further feature-rich expansion.

FUTURE ENHANCEMENT

To evolve this project into a more comprehensive retail platform, the following future enhancements are recommended:

Even though the current version of the application meets its foundational goals, the following improvements can be added to make it more aligned with modern e-commerce platforms:

1. **Complete Checkout and Payment Flow**
   Implement full cart-to-order placement with quantity updates, payment simulation (or gateway integration), and order confirmation.
2. **Order Management and History Modules**
   Add a dedicated order table and history feature so users and admins can track purchases, downloads invoices, and manage order statuses.
3. **User Profile and Security Enhancements**
   Migrate to **hashed password storage** using SHA-256 or bcrypt; allow users to update personal data, reset passwords, and upload profile pictures.
4. **Product Image and Multimedia Support**
   Extend the admin panel to allow image uploads per product; display images via `JLabel` or scaled icon renderer in JTable.
5. **Reporting and Analytics**
   Provide admin reports such as sales summary, most-viewed items, low-stock alerts, and category-wise revenue insights using SQL aggregation functions.
6. **Multi-Platform Expansion**
   Convert the existing desktop implementation to a JavaFX UI or migrate to a web-based framework such as Spring Boot + JSP/HTML for broader accessibility.

REFERENCES

1. https://docs.oracle.com/javase/8/docs/api/javax/swing/packagesummary.html

2. https://docs.oracle.com/javase/tutorial/jdbc/index.html

3. https://dev.mysql.com/doc/refman/8.0/en/

4. https://dev.mysql.com/doc/connector-j/8.0/en/

5. https://www.oracle.com/java/technologies/data-access-object.html