ONLINE BOOK BUYING AND SELLING PORTAL

A PROJECT REPORT

Submitted by

GAYATHRI M J(8115U23EC027)

In partial fulfillment of requirements for the award of the course

EGB1201 - JAVA PROGRAMMING

in

ELECTRONICS AND COMMUNICATION ENGINEERING

K. RAMAKRISHNAN COLLEGE OF ENGINEERING
(An Autonomous Institution, affiliated to Anna University Chennai and Approved by AICTE, New Delhi)

SAMAYAPURAM – 621 112

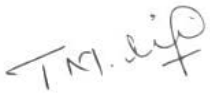
DECEMBER - 2024

K. RAMAKRISHNAN COLLEGE OF ENGINEERING (AUTONOMOUS)

SAMAYAPURAM – 621 111

BONAFIDE CERTIFICATE

Certified that this project report on **"ONLINE BOOK BUYING AND SELLING PORTAL"** is the bonafide work of **GAYATHRI M J (8115U23EC027)** who carried out the project work during the academic year 2024 - 2025 under my supervision.



SIGNATURE

Dr. T. M. NITHYA, M.E.,Ph.D.,

HEAD OF THE DEPARTMENT

ASSOCIATE PROFESSOR

Department of CSE

K.Ramakrishnan College of Engineering
(Autonomous)

Samayapuram – 621 112.

SIGNATURE

Mr.V.KUMARARAJA,M.E.,(Ph.D.,),

SUPERVISOR

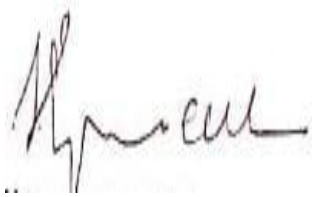
ASSISTANT PROFESSOR

Department of CSE

K.Ramakrishnan College of Engineering
(Autonomous)

Samayapuram – 621 112.

Submitted for the viva-voce examination held on 06/12/2024



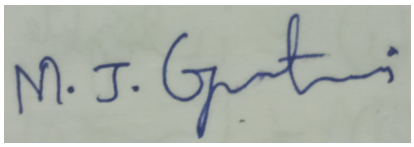
INTERNAL EXAMINER

EXTERNAL EXAMINER

DECLARATION

I declare that the project report on **"ONLINE BOOK BUYING AND SELLING PORTAL"** 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 **EGB1201 - JAVA PROGRAMMING.**

Signature



GAYATHRI M J

Place:Samayapuram

Date:06/12/2024

ACKNOWLEDGEMENT

It is with great pride that I express our gratitude and in-debt to our institution **"K.Ramakrishnan College of Engineering (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. D. SRINIVASAN, B.E, M.E., Ph.D.,** Principal, who gave opportunity to frame the project the full satisfaction.

I whole heartily thanks to **Dr. T. M. NITHYA, 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. V. KUMARARAJA, M.E., (Ph.D.,) ,** 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 achieve a prominent position among the top technical institutions.

MISSION OF THE INSTITUTION

- M1: To bestow standard technical education par excellence through state of the art infrastructure, competent faculty and high ethical standards.
- M2: To nurture research and entrepreneurial skills among students in cutting edge technologies.
- M3: To provide education for developing high-quality professionals to transform the society.

VISION OF DEPARTMENT

To create eminent professionals of Computer Science and Engineering by imparting quality education.

MISSION OF DEPARTMENT

M1: To provide technical exposure in the field of Computer Science and Engineering through state of the art infrastructure and ethical standards.

M2: To engage the students in research and development activities in the field of Computer Science and Engineering.

M3: To empower the learners to involve in industrial and multi-disciplinary projects for addressing the societal needs.

PROGRAM EDUCATIONAL OBJECTIVES

Our graduates shall

- ✔ PEO1: Analyse, design and create innovative products for addressing social needs.
- ✔ PEO2: Equip themselves for employability, higher studies and research.
- ✔ PEO3: Nurture the leadership qualities and entrepreneurial skills for their successful career.

PROGRAM SPECIFIC OUTCOMES (PSOs)

- ✔ PSO1: Apply the basic and advanced knowledge in developing software, hardware and firmware solutions addressing real life problems.
- ✔ PSO2: Design, develop, test and implement product-based solutions for their career enhancement.

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

The Java program is a console-based simulation of an Online Book Buying and Selling Portal that employs object-oriented programming principles to provide a seamless experience for listing, browsing, and purchasing books. The Book class serves as the fundamental building block, encapsulating essential attributes like title, author, price, and seller, along with an overridden toString() method for formatted output. The Bookstore class acts as the core management system, using a dynamic ArrayList to store and handle the inventory of books. It offers methods to add books for sale, display available books, and facilitate purchases by removing selected books from the inventory. The BookPortal class ties these components together with a user-friendly interface that guides users through various operations using a menu-driven structure. Input validation ensures robust interaction, while dynamic inventory management allows real-time updates during transactions. This program exemplifies principles like encapsulation, modularity, and abstraction, demonstrating practical implementation of object-oriented techniques to create an efficient and interactive book trading system.

ABSTRACT WITH POs AND PSOs MAPPING

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

ABSTRACT	POs MAPPED	PSOs MAPPED
The Java program is a console-based simulation of an Online Book Buying and Selling Portal that employs object-oriented programming principles to provide a seamless experience for listing, browsing, and purchasing books. The Book class serves as the fundamental building block, encapsulating essential attributes like title, author, price, and seller, along with an overridden toString() method for formatted output. The Bookstore class acts as the core management system, using a dynamic ArrayList to store and handle the inventory of books. It offers methods to add books for sale, display available books, and facilitate purchases by removing selected books from the inventory.	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

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

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

1.1 Objective

- The objective of this program is to develop a fully functional Online Book Buying and Selling Portal that serves as a simple and interactive platform for listing and purchasing books. This portal facilitates seamless transactions between sellers and buyers through a menu-driven interface. Sellers can list books by providing details such as the title, author, price, and their name, making the books available to potential buyers.view detailed information about each listing, and purchase a book by selecting its number from the list.

- The program is designed with the primary aim of simulating a basic e-commerce workflow while providing a hands-on learning experience in Java programming. It effectively demonstrates the practical application of core Java concepts, including object-oriented programming (OOP), exception handling to ensure smooth operation, addressing scenarios such as invalid inputs or book selection errors.

- Beyond its basic functionality, the portal serves as a foundation for more complex applications. It could be extended with features like user authentication, advanced search and filtering options, or even payment integration, making it relevant for small-scale projects or educational environments. This program bridges the gap between theoretical knowledge and practical implementation, allowing users to gain confidence in working with Java to build real-world systems.

1.2 Overview

The Online Book Buying and Selling Portal provides a user-friendly, console-based interface for buyers and sellers to interact with the system. It is organized into three main components, each contributing to the seamless functioning of the application: This class serves as the blueprint for representing individual books in the portal.It provides methods for adding new books to the store, browsing the list of available books, and facilitating purchases by removing sold books from the inventory. By centralizing these operations, the Bookstore class ensures that all transactions are managed efficiently and accurately, reflecting real-time updates to the inventory.

1.3 Java Programming Concepts

- ✔ This program leverages several fundamental Java programming concepts, showcasing their practical use in developing an interactive and efficient system for book management. Below is a detailed explanation of the key concepts. The program introduces the Book class to encapsulate all relevant attributes and behaviors related to a book, such as its title, author, price, and seller.Instances of the Book class represent individual books.
- ✔ The design emphasizes object-oriented principles by treating books as real-world objects with properties and actions.The program follows the principle of encapsulation by grouping related data (attributes) and operations (methods) together in the Book and Bookstore classes.

- ✔ This ensures a modular codebase, where implementation details are hidden from external entities, making the system easier to maintain and extend.The Bookstore class utilizes the ArrayList<Book> from Java's collections framework to dynamically store and manage a list of books available for sale.This approach provides flexibility, as books can be added, removed, or modified without fixed size limitations, improving scalability and efficiency.

- ✔ The program employs the Scanner class to capture user input for operations like adding books, browsing inventory, and purchasing books.It incorporates mechanisms to handle various data types, such as strings for book titles, integers for menu selection, and doubles for prices, ensuring accurate data entry and processing.The program uses control structures such as conditional statements (if-else) and loops (do-while) to manage user interactions and ensure smooth operation of the system.

- ✔ For instance, the do-while loop enables continuous menu display and user interaction until the user chooses to exit the application.The toString() method is overridden in the Book class to provide a customized string representation of book details.This allows the program to display book information in a user-friendly and readable format when browsing the available inventory.

- ✔ By combining these Java programming concepts, the program demonstrates a structured, efficient, and interactive approach to solving real-world problems, such as managing an online book trading system. The implementation reflects best practices in object -oriented programming, input handling, and system design, making it a valuable learning example

CHAPTER 2

PROJECT METHODOLOGY

2.1 Proposed Work

- The proposed work focuses on the development of an Online Book Buying and Selling Portal using Java, which provides a streamlined platform for users to list books for sale, browse available options, and purchase books efficiently. The design prioritizes functionality, user experience, and scalability, making it a practical demonstration of Java programming principles in action. The system is built upon three key components:

- This class encapsulates all essential details of a book, such as its title, author, price, and seller's name. It serves as the foundational entity for the application, representing individual books in the system. By bundling book-related data and behaviors, the Book class follows the principles of encapsulation, ensuring that book attributes and operations are logically grouped and managed.

- The Bookstore class handles the dynamic inventory of books using an ArrayList<Book>. This structure allows for real-time addition, browsing, and removal of books, ensuring flexible and efficient inventory management. The class also includes methods for displaying book listings in a user-friendly format, enabling seamless interaction. Its implementation highlights the versatility of Java’s collections framework and promotes scalability by accommodating an unlimited number of books.

2.2 Block Diagram

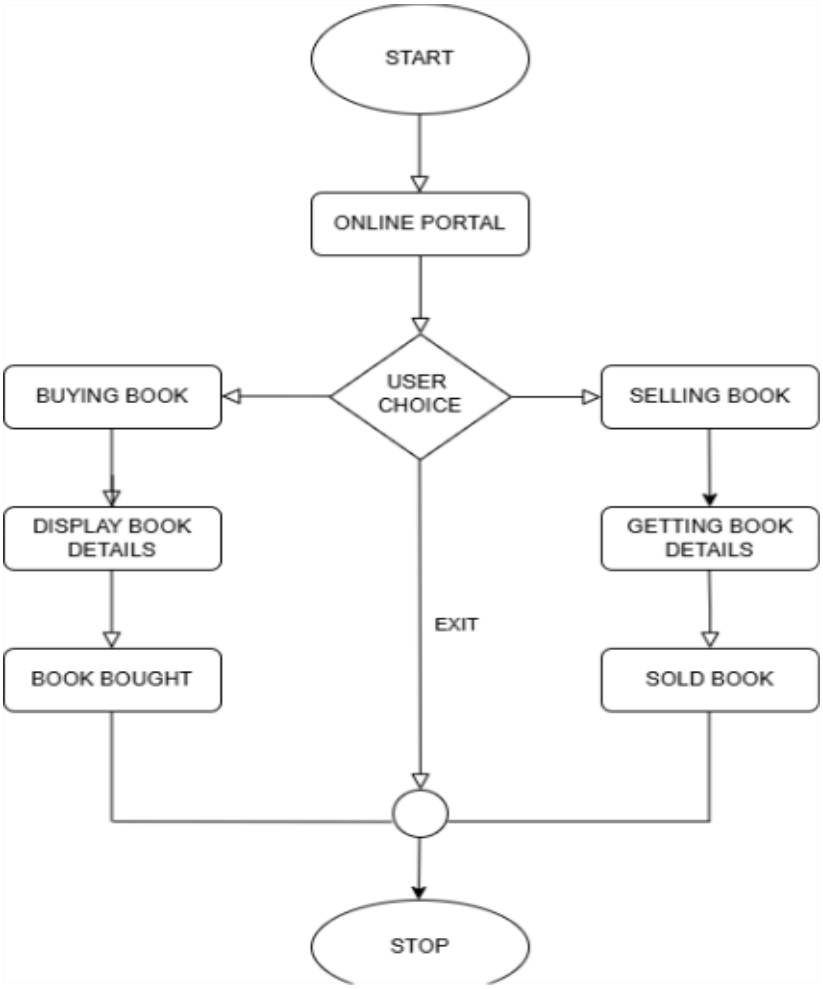


Fig 2.1 Online Book Buying and Selling Portal Block Diagram

CHAPTER 3

MODULE DESCRIPTION

3.1 Selling Book:

The Selling Books module allows users (sellers) to list and manage books for sale on the online portal. It provides the tools for sellers to showcase their books, set prices, manage inventory, and complete transactions with buyers.

3.2 Buying Book:

The Buying Books Module allows users (buyers) to browse, search, and purchase books from the online portal. This module facilitates an intuitive shopping experience, providing features such as book discovery, secure checkout, and payment processing.

3.3 Browsing Data:

The Browsing Data module captures and processes information about users' interactions with the online book portal, helping to enhance the user experience, personalize recommendations, and improve business insights.

3.4 Managing Data:

The Managing Data module focuses on the organization, storage, and retrieval of all critical data related to books, users (buyers and sellers), transactions, and inventory within the online book portal.

3.5 Displaying the Data:

The Displaying Data module shows book details, user profiles, and order statuses in an organized format. It helps users easily navigate through search results, track orders, and view personalized recommendations.

CHAPTER 4

CONCLUSION & FUTURE SCOPE

4.1 CONCLUSION

- ✔ The Online Book Buying and Selling Portal stands as a practical demonstration of how core Java programming concepts can be applied to create a functional and userfriendly system for managing book transactions. The project provides a streamlined, interactive experience for users, enabling them to list books for sale, browse available options, and purchase books with ease.

- ✔ At the core of the system is a robust object-oriented design, which ensures a modular, maintainable, and scalable architecture. Key principles such as encapsulation, modularity, and reusability have been carefully incorporated. The Book class represents individual books, encapsulating essential attributes such as the title, author, price, and seller, while the Bookstore class handles the management of books through dynamic operations like adding, viewing, and removing books.

- ✔ A strong emphasis has been placed on error handling and user experience to make the system robust and reliable. Input validation mechanisms safeguard against invalid or incorrect data entries, ensuring smooth program execution. Users are guided by clear prompts and feedback messages, which enhance the overall usability of the application.

4.2 FUTURE SCOPE

- The Online Book Buying and Selling Portal has significant potential for expansion and enhancement, paving the way for advanced features and improved user experiences. Below are the key areas for future development. Implementing a relational database management system (RDBMS) such as MySQL or PostgreSQL, or a NoSQL solution like MongoDB, would allow for persistent storage of book details, user accounts, and transaction histories.
- This would ensure data retention even after the application is closed, Features like searching by title, author, price range, genre, or seller, along with filtering options, would enhance usability and improve navigation within the system.Adding a user authentication system would enable personalized experiences. Sellers could manage their book listings, and buyers could view their purchase history. Secure login mechanisms using hashed passwords or multi-factor authentication (MFA) would ensure user data privacy and security.
- Transitioning to a GUI using libraries like JavaFX or Swing would make the platform more visually appealing and accessible. A GUI-based interface would attract a broader audience and improve overall user interaction by offering intuitive menus, buttons, and forms.Adding payment gateway support, such as PayPal, Stripe, or UPI, would simulate real-world e-commerce platforms. This feature would enable buyers to complete transactions seamlessly and securely, mimicking realtime monetary exchanges.

APPENDIX A (SOURCE CODE)

```
import java.awt.*;
import java.awt.event.*;
import java.util.ArrayList;

public class BookPortalAWT {
    public static void main(String[] args) {
        Frame frame = new Frame("Online Book Buying & Selling Portal");
        frame.setSize(500, 400);
        frame.setLayout(new FlowLayout());

        // Components
        Label heading = new Label("Welcome to the Book Portal");
        heading.setFont(new Font("Arial", Font.BOLD, 16));
        frame.add(heading);

        // Buttons
        Button buyButton = new Button("Buy a Book");
        Button sellButton = new Button("Sell a Book");
        Button chooseButton = new Button("Choose Book");
        Button addBookButton = new Button("Add Book to Portal");
        frame.add(buyButton);
        frame.add(sellButton);
        frame.add(chooseButton);
        frame.add(addBookButton);

        // TextArea to display output
```

```
TextArea outputArea = new TextArea(10, 40);
frame.add(outputArea);

// Choice for book selection (initially empty)
Choice bookChoice = new Choice();
frame.add(bookChoice);

// TextFields for selling book details (book name and price)
TextField bookNameField = new TextField(40);
TextField bookPriceField = new TextField(10);
bookNameField.setEnabled(false); // Initially disabled
bookPriceField.setEnabled(false); // Initially disabled
frame.add(bookNameField);
frame.add(bookPriceField);

// List to store books and their prices
ArrayList<String> booksList = new ArrayList<>();
ArrayList<String> booksPriceList = new ArrayList<>();

// Predefined books and prices
String[] predefinedBooks = {
    "Java Programming",
    "Python for Beginners",
    "Data Structures",
    "Web Development"
};
String[] predefinedPrices = {
    "$30",
    "$25",
```

```
            "$20",
            "$35"
        };

        // Add predefined books to the portal and Choice dropdown
        for (String book : predefinedBooks) {
            booksList.add(book);
        }
        for (String price : predefinedPrices) {
            booksPriceList.add(price);
        }

        // Add predefined books to the Choice dropdown
        for (String book : predefinedBooks) {
            bookChoice.add(book);
        }

        // Event Listeners
        buyButton.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                outputArea.setText("You chose to buy a book.\nAvailable Books:\n");
                if (booksList.isEmpty()) {
                    outputArea.append("No books available for sale.");
                } else {
                    for (int i = 0; i < booksList.size(); i++) {
                        outputArea.append(booksList.get(i) + " - " + booksPriceList.get(i) + "\n");
                    }
```

```
            }
            bookChoice.setEnabled(true); // Enable book choice when buying
            bookNameField.setEnabled(false); // Disable book name input
            bookPriceField.setEnabled(false); // Disable book price input
        }
    });

    sellButton.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent e) {
            outputArea.setText("You chose to sell a book.\nPlease provide the name and price of the book you want to sell.");
            bookChoice.setEnabled(false); // Disable book choice when selling
            bookNameField.setEnabled(true); // Enable book name input
            bookPriceField.setEnabled(true); // Enable book price input
        }
    });

    chooseButton.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent e) {
            String selectedBook = bookChoice.getSelectedItem();
            int index = booksList.indexOf(selectedBook);
            String selectedBookPrice = (index != -1) ? booksPriceList.get(index) : "Unknown";
            outputArea.setText("You have successfully ordered the book: " + selectedBook + " - Price: " + selectedBookPrice);
        }
    });
```

```
        // Add new book to the portal
        addBookButton.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                String bookName = bookNameField.getText();
                String bookPrice = bookPriceField.getText();

                if (!bookName.isEmpty() && !bookPrice.isEmpty()) {
                    // Add the book and price to the lists
                    booksList.add(bookName);
                    booksPriceList.add(bookPrice);

                    // Add the book name to the Choice dropdown
                    bookChoice.add(bookName);

                    // Clear input fields
                    bookNameField.setText("");
                    bookPriceField.setText("");

                    // Display success message and updated list
                    StringBuilder successMessage = new StringBuilder("Your book has been successfully added to the portal.\n\nUpdated list of books:\n");
                    for (int i = 0; i < booksList.size(); i++) {
                        successMessage.append(booksList.get(i)).append(" - ").append(booksPriceList.get(i)).append("\n");
                    }
                    outputArea.setText(successMessage.toString());
                } else {
                    outputArea.setText("Please enter both book name and price.");
```

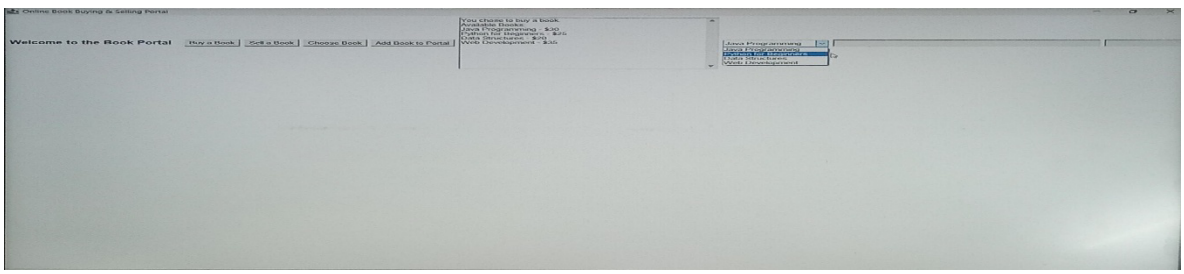
```
            }
        }
    });

    // Close Window
    frame.addWindowListener(new WindowAdapter() {
        public void windowClosing(WindowEvent we) {
            System.exit(0);
        }
    });

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

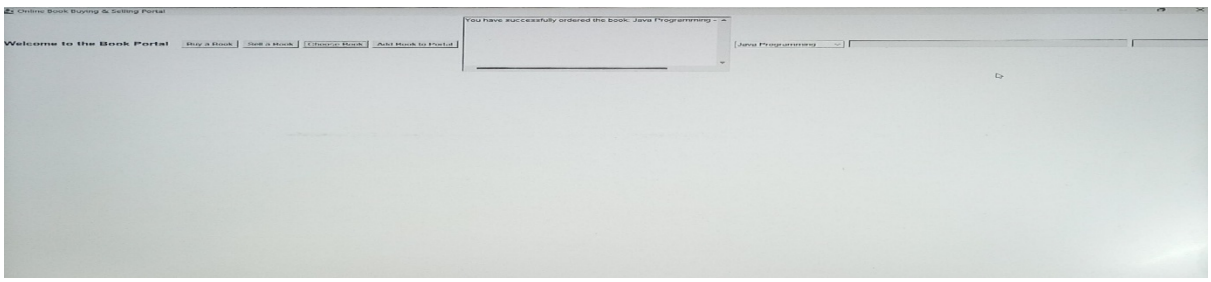
APPENDIX B(SCREENSHOTS)

5.1 BUYING BOOK:

Buy a Book Output:

1. Buy the Book button clicked.
2. Searching for book: Java Programming.
3. Purchase confirmed. Order placed successfully!

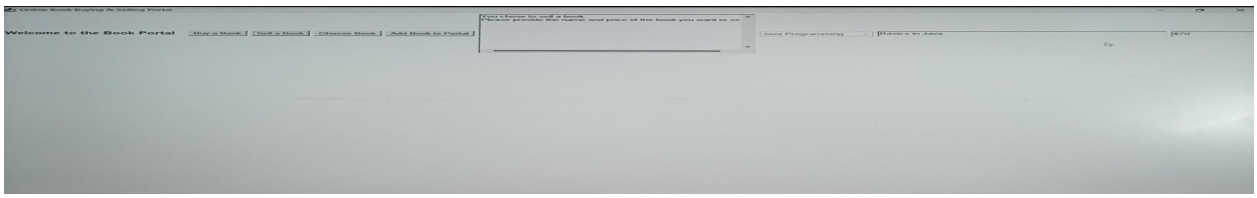


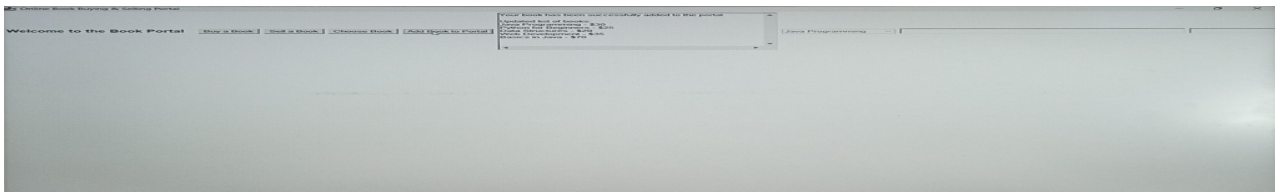


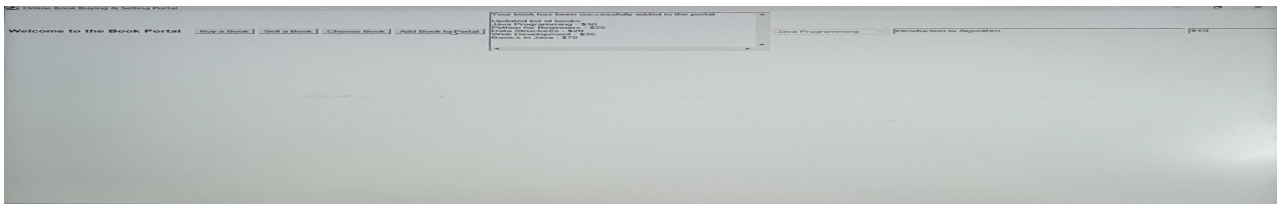
5.2 SELLING BOOK:

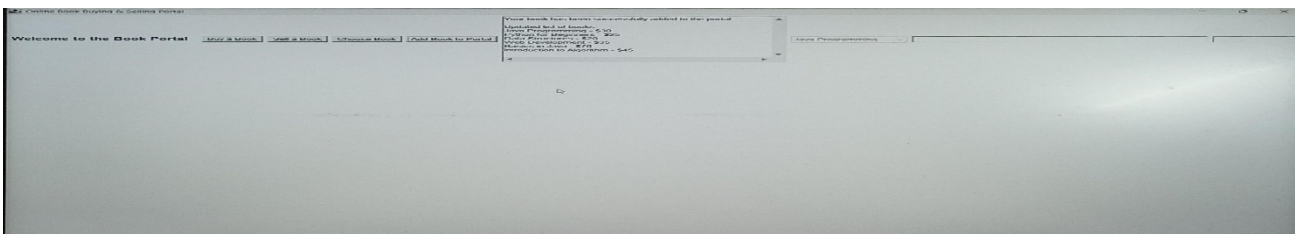
Sell a Book Output:

1. Sell a Book button clicked.
2. Book added: Effective Java by Joshua Bloch for $50.
3. Book listed successfully for sale.









REFERENCE

1. Spring Framework Documentation:
Spring Framework - Building Web Applications

2. Java EE Documentation for E-commerce:
Java EE - Building Web Applications

3. "Building an Online Bookstore with Java" by GeeksforGeeks:
Building an Online Bookstore with Java

4. "Online Bookstore in Java" by JournalDev:
Java Online Bookstore Tutorial

5. "Web Development with Node.js and Express" by Ethan Brown:
Web Development with Node.js and Express

6. "PHP and MySQL for Dynamic Web Sites" by Larry Ullman:
PHP and MySQL Tutorial

7. "Designing E-commerce Websites" : Shopify E-commerce Guide

8. "HTML & CSS: Design and Build Websites" by Jon Duckett:
HTML & CSS: Design and Build Websites

9. "E-commerce Websites" by MDN Web Docs: MDN Web Docs - E-commerce Development

10. "Stripe Documentation for E-commerce Integration":Stripe Documentation for Payment Integration .