RAJALAKSHMI
ENGINEERING COLLEGE
An AUTONOMOUS Institution
Affiliated to ANNA UNIVERSITY, Chennai

ONLINE BOOK STORE

A MINI PROJECT REPORT

Submitted by

NISHANTHINI S	**231501113**
NIKSHITHA H	**231501109**
NIMAL M G	**231501110**

In partial fulfillment for the award of the degree of

BACHELOR OF

TECHNOLOGY IN

ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING

RAJALAKSHMI ENGINEERING COLLEGE (AUTONOMOUS)

THANDALAM

CHENNAI-602105

2024 - 2025

BONAFIDE CERTIFICATE

Certified that this project report "**ONLINE BOOK STORE**" is the Bonafide work of **"NISHANTHINI S (231501113), NIKSHITHA H (231501109), NIMAL M G (231501110)"** who carried out the project work under my supervision.

Submitted for the Practical Examination held on _____

SIGNATURE

Mr. U. Kumaran,
Assistant Professor (SS)
AIML,
Rajalakshmi Engineering College,
(Autonomous),
Thandalam, Chennai - 602 105

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ABSTRACT

The Online Book Store is a web-based application developed using Java, JDBC, and Servlets to purchase books online and administrators to manage book inventory. The system features two distinct interfaces: an admin portal for managing books, pricing, and inventory, and a customer portal for browsing and purchasing books. Administrators can add new books, modify prices, and monitor stock levels, while customers can register accounts, browse available books, add items to their cart, and complete purchases securely.

The front-end is built using HTML, CSS, JavaScript, and Bootstrap to ensure a responsive and user-friendly experience. The back-end utilizes Java Servlets for processing requests and JDBC for database operations with MySQL. The system maintains comprehensive transaction records and generates payment receipts for completed orders. This streamlined solution demonstrates the practical implementation of Java web technologies while providing essential e-commerce functionality.

TABLE OF CONTENTS

I. INTRODUCTION

1.1 INTRODUCTION

In today's digital age, online shopping has become an integral part of our daily lives. The Online Book Store project is a web-based solution designed to provide a seamless book-buying experience for customers while offering efficient inventory management for administrators. Built using Java web technologies, including JDBC and Servlets, the system offers a secure and user-friendly platform for both book sellers and buyers.

The project features two main interfaces: an administrative panel for managing inventory and sales, and a customer portal for browsing and purchasing books. Using modern web technologies like HTML5, CSS3, and Bootstrap, the system ensures a responsive design accessible across various devices. This practical implementation demonstrates the integration of database management, web development, and e-commerce functionalities in a real-world application.

1.2 OBJECTIVES

1. **Primary Objectives**
    - Create a user-friendly online platform for buying and selling books
    - Develop a secure system for managing book inventory and sales
    - Implement efficient user authentication and authorization
    - Provide automated transaction processing and receipt generation
2. **Business Objectives**
    - Streamline book selling operations
    - Reduce manual inventory management efforts
    - Increase accessibility for customers
    - Maintain accurate sales records.

1.3 MODULES

1. **Admin Module**
    - Login/Authentication
    - Book Management
        - Add new books
        - Update book details
        - Remove books
        - Modify prices
    - Inventory Control
        - Stock management
        - Sales tracking
2. **User Module**
    - Registration/Login

- Book Browsing
  - View available books
  - Search functionality
- Shopping Cart
  - Add/remove books
  - Update quantities
- Purchase Processing
  - Checkout system
  - Payment handling
  - Receipt generation

3. **Database Module**
   - User data management
   - Book inventory records
   - Transaction history
   - Order processing
4. **Security Module**
   - User authentication
   - Session management
   - Secure data transmission
   - Access control

II. SURVEY OF TECHNOLOGIES

2.1 SOFTWARE DESCRIPTION

ECLIPSE

Eclipse is written mostly in Java and its primary use is for developing Java applications,Eclipse is an integrated development environment (IDE) used in computer programming. It contains a base workspace and an extensible plug-in system for customizing the environment. It is the second-most-popular IDE for Java development, and, until 2016, was the most popular. Eclipse is written mostly in Java and its primary use is for developing Java applications.

Eclipse Enterprise Edition (EE) is a package for developers who work with Java and web applications. It includes tools for:

- Java
- JavaScript
- TypeScript
- JavaServer Pages and Faces
- Web Services
- Maven and Gradle
- Git

Eclipse EE is a version of Eclipse that comes with tools to make it easier to write server code. For example, you can compile and run a server by pressing the play button.

2.2 LANGUAGES

2.2.1 MySQL

MySQL is an open-source relational database management system (RDBMS). A relational database organizes data into one or more data tables in which data may be related to each other; these relations help structure the data. SQL is a language that programmers use to create, modify and extract data from the relational database, as well as control user access to the database. In

addition to relational databases and SQL, an RDBMS like MySQL works with an operating system to implement a relational database in a computer's storage system, manages users, allows for network access and facilitates testing database integrity and creation of backups.

2.2.2 JAVA

Java is a set of computer software and specifications that provides a software platform for developing application software and deploying it in a cross-platform computing environment. Java is used in a wide variety of computing platforms from embedded devices and mobile phones to enterprise servers and supercomputers. Java applets, which are less common than standalone Java applications, were commonly run in secure, sandboxed environments to provide many features of native applications through being embedded in HTML pages.

2.2.3 HTML

HTML, or Hypertext Markup Language, is the standard language used to create web pages. It defines the structure and content of web documents using tags and attributes to format text, embed images, create links, and build interactive elements. HTML facilitates communication between web browsers and servers, making it a crucial skill for web developers.

HTML was invented by Tim Berners-Lee, a physicist at CERN, in 1990. His goal was to create a simple way to share and access documents over the Internet. Since its inception, HTML has evolved significantly, becoming the foundation of web development. When working with HTML, you use a simple code structure that includes tags and attributes to build the layout of a webpage.

2.2.4 CSS

CSS, which stands for Cascading Style Sheets, is a language in web development that enhances the presentation of HTML elements. By applying styles like color, layout, and spacing, CSS makes web pages visually appealing and responsive to various screen sizes. CSS is designed to enable the separation of content and presentation, including layout, colors, and fonts. This separation can improve content accessibility, since the content can be written without concern for

its presentation; provide more flexibility and control in the specification of presentation characteristics; enable multiple web pages to share formatting by specifying the relevant CSS in a separate .css file, which reduces complexity and repetition in the structural content; and enable the .css file to be cached to improve the page load speed between the pages that share the file and its formatting.

2.2.5 JAVASCRIPT

JavaScript, often abbreviated as JS, is a programming language and core technology of the Web, alongside HTML and CSS. 99% of websites use JavaScript on the client side for webpage behavior. JavaScript is a high-level, often just-in-time compiled language that conforms to the ECMAScript standard. It has dynamic typing, prototype-based object-orientation, and first-class functions. It is multi-paradigm, supporting event-driven, functional, and imperative programming styles. It has application programming interfaces (APIs) for working with text, dates, regular expressions, standard data structures, and the Document Object Model (DOM).

III.REQUIREMENTS AND ANALYSIS

3.1 REQUIREMENT SPECIFICATION

User Requirements

The system requirement in the online bookstore management focuses on the ability to search for books by title, author, or genre by the customer. It also includes user account management, viewing book details, and making purchases.

System Requirements

There should be a database backup of the online bookstore system. The operating system should be Windows 10 or a higher version of Windows.

3.2 HARDWARE AND SOFTWARE REQUIREMENTS

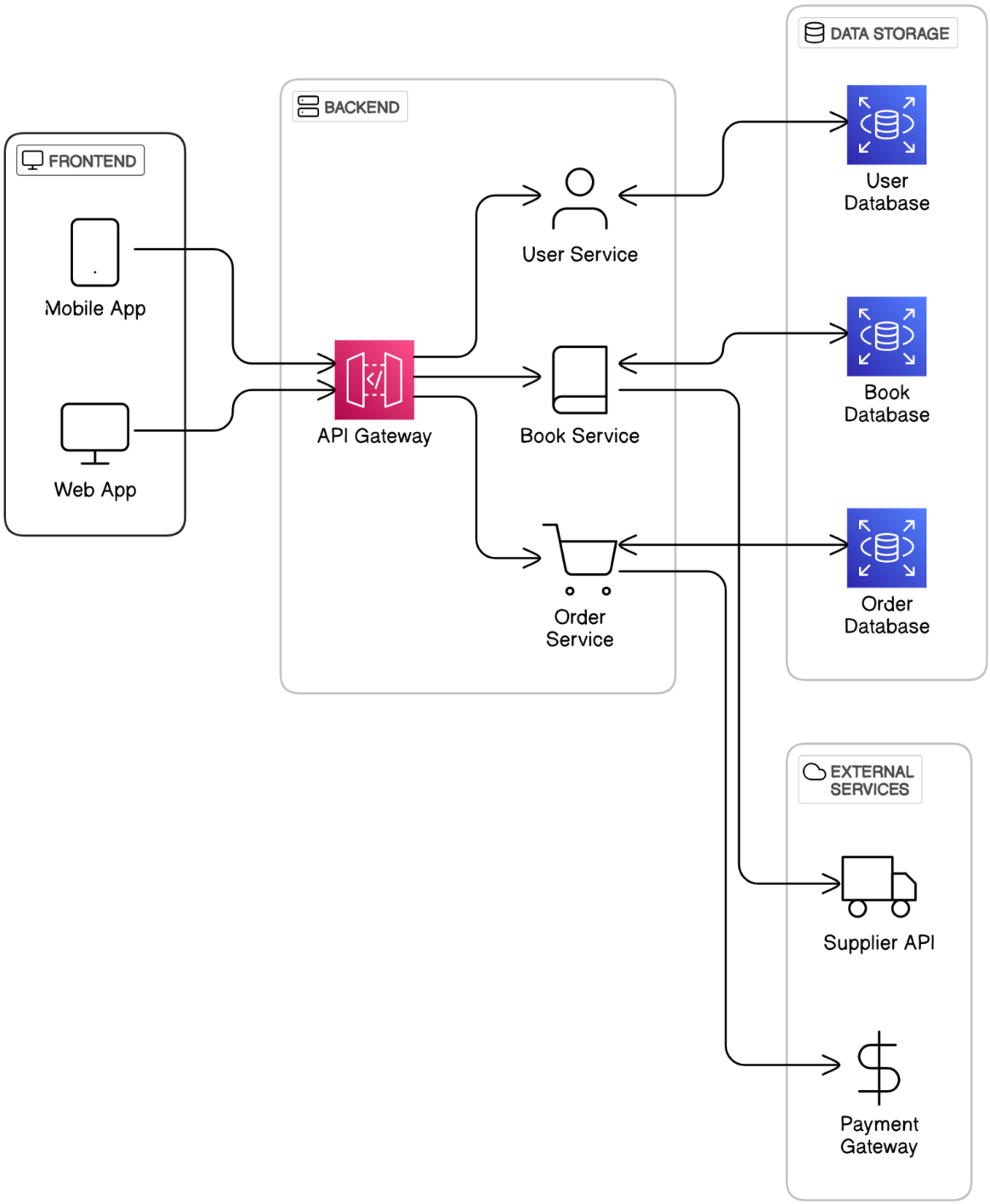
Software Requirements

- Operating System: Windows 10
- Front End: HTML, CSS, JavaScript
- Back End: Java, MySQL

Hardware Requirements

- Desktop PC or Laptop
- Printer (optional)
- Operating System: Windows 10
- Intel® Core™ i3-6006U CPU @ 2.00GHz or higher
- 4.00 GB RAM or higher
- 64-bit operating system, x64 based processor
- Monitor Resolution: 1024 x 768 or higher
- Keyboard and Mouse

Online Book Store Architecture



3.4 DATA DICTIONARY

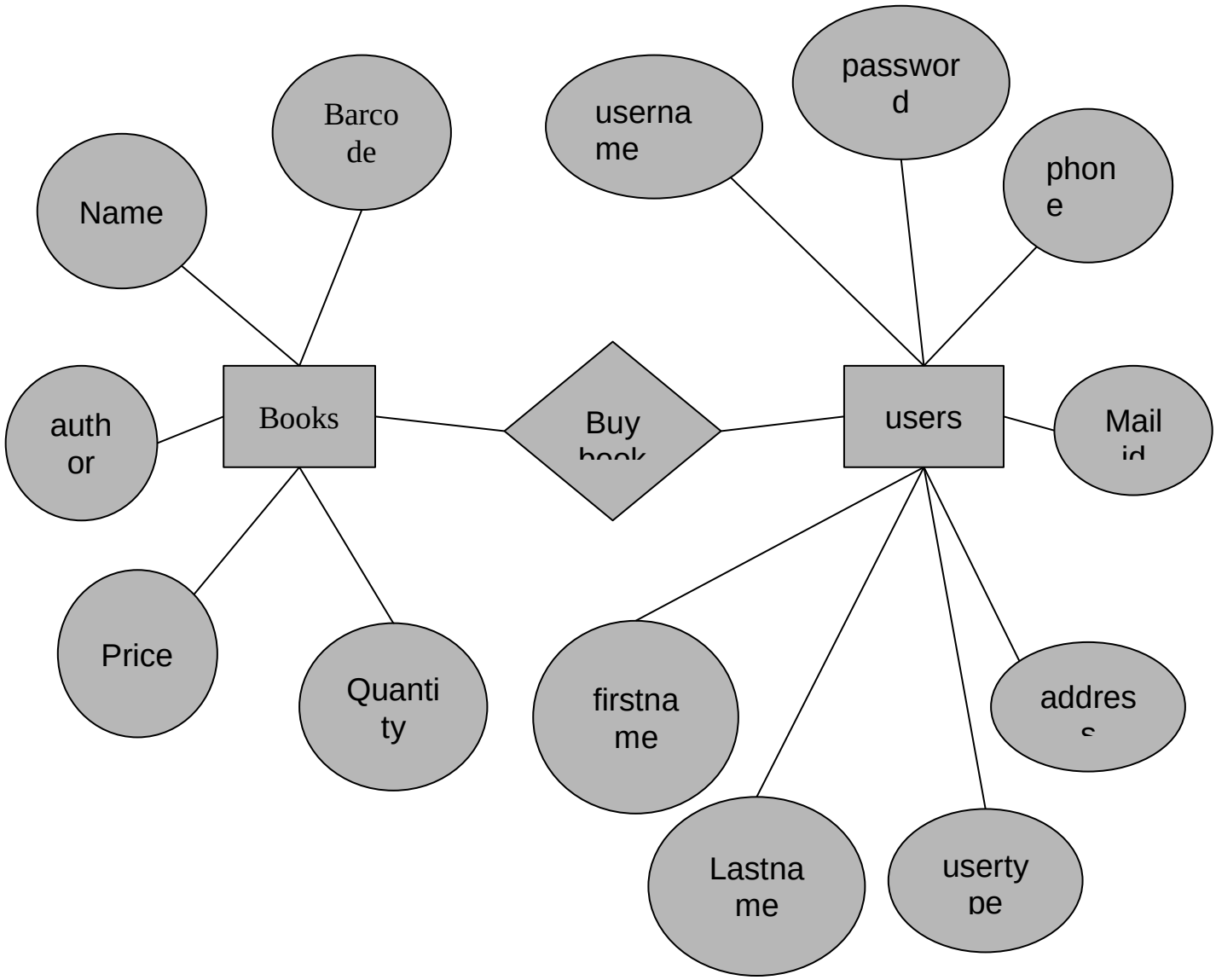
BOOK TABLE

	Field	Type	Null	Key	Default	Extra
▶	barcode	varchar(100)	NO	PRI	NULL	
	name	varchar(100)	YES		NULL	
	author	varchar(100)	YES		NULL	
	price	int	YES		NULL	
	quantity	int	YES		NULL	

USER TABLE

	Field	Type	Null	Key	Default	Extra
▶	username	varchar(100)	NO	PRI	NULL	
	password	varchar(100)	YES		NULL	
	firstname	varchar(100)	YES		NULL	
	lastname	varchar(100)	YES		NULL	
	address	text	YES		NULL	
	phone	varchar(100)	YES		NULL	
	mailid	varchar(100)	YES		NULL	
	usertype	int	YES		NULL	

3.5 ER DIAGRAM



IV. PROGRAM CODE

DATABASE

```
CREATE DATABASE if not exists onlinebookstore;
\c onlinebookstore
CREATE TABLE if not exists books
 (
   barcode   VARCHAR(100) PRIMARY KEY,
   name      TEXT NOT NULL,
   author    VARCHAR(100) NOT NULL,
   price     INT,
   quantity  REAL
 );

 CREATE TABLE if not exists users
 (
   username  VARCHAR(100) PRIMARY KEY,
   password  VARCHAR(100) NOT NULL,
   firstname VARCHAR(100) NOT NULL,
   lastname  VARCHAR(100) NOT NULL,
   address   TEXT NOT NULL,
   phone     VARCHAR(100) NOT NULL,
   mailid    VARCHAR(100) NOT NULL,
   usertype  INT
 );
```

APPLICATION PROPERTIES

db.driver=com.mysql.cj.jdbc.Driver

db.host=jdbc:mysql://localhost

db.port=3306

db.name=onlinebookstore

db.username=root

db.password=root

CUSTOMER REGISTER

```
package servlets;


import java.io.IOException;
import java.io.PrintWriter;


import javax.servlet.RequestDispatcher;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;


import com.bittercode.constant.BookStoreConstants;
import com.bittercode.constant.ResponseCode;
import com.bittercode.constant.db.UsersDBConstants;
import com.bittercode.model.User;
import com.bittercode.model.UserRole;
```

```
import com.bittercode.service.UserService;
import com.bittercode.service.impl.UserServiceImpl;

public class CustomerRegisterServlet extends HttpServlet {

    UserService userService = new UserServiceImpl();

    public void service(HttpServletRequest req, HttpServletResponse res) throws IOException,
ServletException {
        PrintWriter pw = res.getWriter();
        res.setContentType(BookStoreConstants.CONTENT_TYPE_TEXT_HTML);

        String pWord = req.getParameter(UsersDBConstants.COLUMN_PASSWORD);
        String fName = req.getParameter(UsersDBConstants.COLUMN_FIRSTNAME);
        String lName = req.getParameter(UsersDBConstants.COLUMN_LASTNAME);
        String addr = req.getParameter(UsersDBConstants.COLUMN_ADDRESS);
        String phNo = req.getParameter(UsersDBConstants.COLUMN_PHONE);
        String mailId = req.getParameter(UsersDBConstants.COLUMN_MAILID);
        User user = new User();
        user.setEmailId(mailId);
        user.setFirstName(fName);
        user.setLastName(lName);
        user.setPassword(pWord);
        user.setPhone(Long.parseLong(phNo));
        user.setAddress(addr);
```

```
        try {
            String respCode = userService.register(UserRole.CUSTOMER, user);
            System.out.println(respCode);
            if (ResponseCode.SUCCESS.name().equalsIgnoreCase(respCode)) {
                RequestDispatcher rd = req.getRequestDispatcher("CustomerLogin.html");
                rd.include(req, res);
                pw.println("<table class=\"tab\"><tr><td>User Registered
Successfully</td></tr></table>");
            } else {
                RequestDispatcher rd = req.getRequestDispatcher("CustomerRegister.html");
                rd.include(req, res);
                pw.println("<table class=\"tab\"><tr><td>" + respCode + "</td></tr></table>");
                pw.println("Sorry for interruption! Try again");
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

CUSTOMER LOGIN

```
package servlets;


import java.io.IOException;
import java.io.PrintWriter;
```

```
import javax.servlet.RequestDispatcher;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import com.bittercode.constant.BookStoreConstants;
import com.bittercode.constant.db.UsersDBConstants;
import com.bittercode.model.User;
import com.bittercode.model.UserRole;
import com.bittercode.service.UserService;
import com.bittercode.service.impl.UserServiceImpl;

public class CustomerLoginServlet extends HttpServlet {

    UserService authService = new UserServiceImpl();

    public void doPost(HttpServletRequest req, HttpServletResponse res) throws IOException,
ServletException {
        PrintWriter pw = res.getWriter();
        res.setContentType(BookStoreConstants.CONTENT_TYPE_TEXT_HTML);
        String uName = req.getParameter(UsersDBConstants.COLUMN_USERNAME);
        String pWord = req.getParameter(UsersDBConstants.COLUMN_PASSWORD);
        User user = authService.login(UserRole.CUSTOMER, uName, pWord, req.getSession());
```

```
        try {


            if (user != null) {


                RequestDispatcher rd = req.getRequestDispatcher("CustomerHome.html");

                rd.include(req, res);

                pw.println("    <div id=\"topmid\"><h1>Welcome to Online <br>Book
Store</h1></div>\r\n"

                        + "    <br>\r\n"

                        + "    <table class=\"tab\">\r\n"

                        + "        <tr>\r\n"

                        + "            <td><p>Welcome "+user.getFirstName()+", Happy Learning
!!</p></td>\r\n"

                        + "        </tr>\r\n"

                        + "    </table>");


            } else {


                RequestDispatcher rd = req.getRequestDispatcher("CustomerLogin.html");

                rd.include(req, res);

                pw.println("<table class=\"tab\"><tr><td>Incorrect UserName or
PassWord</td></tr></table>");

            }


        } catch (Exception e) {

            e.printStackTrace();
```

```
        }
    }
}
```

ADD BOOK

```
package servlets;

import java.io.IOException;
import java.io.PrintWriter;
import java.util.UUID;

import javax.servlet.RequestDispatcher;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import com.bittercode.constant.BookStoreConstants;
import com.bittercode.constant.db.BooksDBConstants;
import com.bittercode.model.Book;
import com.bittercode.model.UserRole;
import com.bittercode.service.BookService;
import com.bittercode.service.impl.BookServiceImpl;
import com.bittercode.util.StoreUtil;

public class AddBookServlet extends HttpServlet {
```

```
    BookService bookService = new BookServiceImpl();


    public void service(HttpServletRequest req, HttpServletResponse res) throws IOException,
ServletException {
        PrintWriter pw = res.getWriter();
        res.setContentType(BookStoreConstants.CONTENT_TYPE_TEXT_HTML);


        if (!StoreUtil.isLoggedIn(UserRole.SELLER, req.getSession())) {
            RequestDispatcher rd = req.getRequestDispatcher("SellerLogin.html");
            rd.include(req, res);
            pw.println("<table class=\"tab\"><tr><td>Please Login First to
Continue!!</td></tr></table>");
            return;
        }


        String bName = req.getParameter(BooksDBConstants.COLUMN_NAME);
        RequestDispatcher rd = req.getRequestDispatcher("SellerHome.html");
        rd.include(req, res);
        StoreUtil.setActiveTab(pw, "addbook");
        pw.println("<div class='container my-2'>");
        if(bName == null || bName.isBlank()) {
            //render the add book form;
            showAddBookForm(pw);
            return;
        } //else process the add book
```

```
        try {

            String uniqueID = UUID.randomUUID().toString();

            String bCode = uniqueID;

            String bAuthor = req.getParameter(BooksDBConstants.COLUMN_AUTHOR);

            double bPrice =
Integer.parseInt(req.getParameter(BooksDBConstants.COLUMN_PRICE));

            int bQty =
Integer.parseInt(req.getParameter(BooksDBConstants.COLUMN_QUANTITY));



            Book book = new Book(bCode, bName, bAuthor, bPrice, bQty);

            String message = bookService.addBook(book);

            if ("SUCCESS".equalsIgnoreCase(message)) {

                pw.println(

                        "<table class=\"tab\"><tr><td>Book Detail Updated Successfully!<br/>Add More
Books</td></tr></table>");

            } else {

                pw.println("<table class=\"tab\"><tr><td>Failed to Add Books! Fill up
CareFully</td></tr></table>");

                //rd.include(req, res);

            }

        } catch (Exception e) {

            e.printStackTrace();

            pw.println("<table class=\"tab\"><tr><td>Failed to Add Books! Fill up
CareFully</td></tr></table>");

        }

    }
```

```
    private static void showAddBookForm(PrintWriter pw) {

        String form = "<table class=\"tab my-5\" style=\"width:40%;\">\r\n"

                + "        <tr>\r\n"

                + "            <td>\r\n"

                + "                <form action=\"addbook\" method=\"post\">\r\n"

                + "                    <!-- <label for=\"bookCode\">Book Code : </label><input
type=\"text\" name=\"barcode\" id=\"bookCode\" placeholder=\"Enter Book Code\"
required><br/> -->\r\n"

                + "                    <label for=\"bookName\">Book Name : </label> <input type=\"text\"
name=\"name\" id=\"bookName\" placeholder=\"Enter Book's name\" required><br/>\r\n"

                + "                    <label for=\"bookAuthor\">Book Author : </label><input type=\"text\"
name=\"author\" id=\"bookAuthor\" placeholder=\"Enter Author's Name\" required><br/>\r\n"

                + "                    <label for=\"bookPrice\">Book Price : </label><input type=\"number\"
name=\"price\" placeholder=\"Enter the Price\" required><br/>\r\n"

                + "                    <label for=\"bookQuantity\">Book Qnty : </label><input
type=\"number\" name=\"quantity\" id=\"bookQuantity\" placeholder=\"Enter the quantity\"
required><br/>\r\n"

                + "                    <input class=\"btn btn-success my-2\" type=\"submit\" value=\" Add
Book \">\r\n"

                + "                </form>\r\n"

                + "            </td>\r\n"

                + "        </tr>  \r\n"

                + "        <!-- <tr>\r\n"

                + "            <td><a href=\"index.html\">Go Back To Home Page</a></td>\r\n"

                + "        </tr> -->\r\n"

                + "    </table>";

        pw.println(form);

    }
```

}

BUY BOOK

```
package servlets;


import java.io.IOException;

import java.io.PrintWriter;

import java.util.List;


import javax.servlet.RequestDispatcher;

import javax.servlet.ServletException;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;


import com.bittercode.constant.BookStoreConstants;

import com.bittercode.model.Book;

import com.bittercode.model.UserRole;

import com.bittercode.service.BookService;

import com.bittercode.service.impl.BookServiceImpl;

import com.bittercode.util.StoreUtil;


public class BuyBooksServlet extends HttpServlet {

    BookService bookService = new BookServiceImpl();
```

```
    public void doPost(HttpServletRequest req, HttpServletResponse res) throws IOException,
ServletException {

        PrintWriter pw = res.getWriter();

        res.setContentType(BookStoreConstants.CONTENT_TYPE_TEXT_HTML);

        if (!StoreUtil.isLoggedIn(UserRole.CUSTOMER, req.getSession())) {

            RequestDispatcher rd = req.getRequestDispatcher("CustomerLogin.html");

            rd.include(req, res);

            pw.println("<table class=\"tab\"><tr><td>Please Login First to
Continue!!</td></tr></table>");

            return;

        }

        try {

            List<Book> books = bookService.getAllBooks();

            RequestDispatcher rd = req.getRequestDispatcher("CustomerHome.html");

            rd.include(req, res);

            StoreUtil.setActiveTab(pw, "cart");

            pw.println("<div class=\"tab hd brown \">Books Available In Our Store</div>");

            pw.println("<div class=\"tab\"><form action=\"buys\" method=\"post\">");

            pw.println("<table>\r\n" +

                    "                    <tr>\r\n" +

                    "                        <th>Books</th>\r\n" +

                    "                        <th>Code</th>\r\n" +

                    "                        <th>Name</th>\r\n" +

                    "                        <th>Author</th>\r\n" +

                    "                        <th>Price</th>\r\n" +

                    "                        <th>Avail</th>\r\n" +
```

```
            "                           <th>Qty</th>\r\n" +
            "                       </tr>");
        int i = 0;
        for (Book book : books) {
            String bCode = book.getBarcode();
            String bName = book.getName();
            String bAuthor = book.getAuthor();
            double bPrice = book.getPrice();
            int bAvl = book.getQuantity();
            i = i + 1;
            String n = "checked" + Integer.toString(i);
            String q = "qty" + Integer.toString(i);
            pw.println("<tr>\r\n" +
                    "                               <td>\r\n" +
                    "                                   <input type=\"checkbox\" name=" + n + "
value=\"pay\">\r\n" + // Value is
                                                                    // made equal
                                                                    // to bcode
                    "                               </td>");
            pw.println("<td>" + bCode + "</td>");
            pw.println("<td>" + bName + "</td>");
            pw.println("<td>" + bAuthor + "</td>");
            pw.println("<td>" + bPrice + "</td>");
            pw.println("<td>" + bAvl + "</td>");
            pw.println("<td><input type=\"text\" name=" + q + " value=\"0\" text-
align=\"center\"></td></tr>");
```

```
            }
            pw.println("</table>\r\n" + "<input type=\"submit\" value=\" PAY NOW \">" + "<br/>"
+
                    "  </form>\r\n" +
                    "  </div>");
            // pw.println("<div class=\"tab\"><a href=\"AddBook.html\">Add More
            // Books</a></div>");
        } catch (Exception e) {
            e.printStackTrace();
        }
    }

}
```

CART

```
package servlets;

import java.io.IOException;
import java.io.PrintWriter;
import java.util.ArrayList;
import java.util.List;

import javax.servlet.RequestDispatcher;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
```

```
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;

import com.bittercode.constant.BookStoreConstants;
import com.bittercode.model.Book;
import com.bittercode.model.Cart;
import com.bittercode.model.UserRole;
import com.bittercode.service.BookService;
import com.bittercode.service.impl.BookServiceImpl;
import com.bittercode.util.StoreUtil;

public class CartServlet extends HttpServlet {

    BookService bookService = new BookServiceImpl();

    public void service(HttpServletRequest req, HttpServletResponse res) throws IOException,
ServletException {
        PrintWriter pw = res.getWriter();
        res.setContentType(BookStoreConstants.CONTENT_TYPE_TEXT_HTML);

        // Check if Customer is logged In
        if (!StoreUtil.isLoggedIn(UserRole.CUSTOMER, req.getSession())) {
            RequestDispatcher rd = req.getRequestDispatcher("CustomerLogin.html");
            rd.include(req, res);
```

```
            pw.println("<table class=\"tab\"><tr><td>Please Login First to
Continue!!</td></tr></table>");

            return;

        }

        try {

            // Add/Remove Item from the cart if requested

            // store the comma separated bookIds of cart in the session

            StoreUtil.updateCartItems(req);


            HttpSession session = req.getSession();

            String bookIds = "";

            if (session.getAttribute("items") != null)

                bookIds = (String) session.getAttribute("items");// read comma separated bookIds from
session


            RequestDispatcher rd = req.getRequestDispatcher("CustomerHome.html");

            rd.include(req, res);


            // Set the active tab as cart

            StoreUtil.setActiveTab(pw, "cart");


            // Read the books from the database with the respective bookIds

            List<Book> books = bookService.getBooksByCommaSeperatedBookIds(bookIds);

            List<Cart> cartItems = new ArrayList<Cart>();

            pw.println("<div id='topmid' style='background-color:grey'>Shopping Cart</div>");

            pw.println("<table class=\"table table-hover\" style='background-color:white'>\r\n"
```

```
            + "  <thead>\r\n"
            + "    <tr style='background-color:black; color:white;'>\r\n"
            + "      <th scope=\"col\">BookId</th>\r\n"
            + "      <th scope=\"col\">Name</th>\r\n"
            + "      <th scope=\"col\">Author</th>\r\n"
            + "      <th scope=\"col\">Price/Item</th>\r\n"
            + "      <th scope=\"col\">Quantity</th>\r\n"
            + "      <th scope=\"col\">Amount</th>\r\n"
            + "    </tr>\r\n"
            + "  </thead>\r\n"
            + "  <tbody>\r\n");
        double amountToPay = 0;
        if (books == null || books.size() == 0) {
            pw.println("    <tr style='background-color:green'>\r\n"
                    + "      <th scope=\"row\" colspan='6' style='color:yellow; text-align:center;'> No
Items In the Cart </th>\r\n"
                    + "    </tr>\r\n");
        }
        for (Book book : books) {
            int qty = (int) session.getAttribute("qty_" + book.getBarcode());
            Cart cart = new Cart(book, qty);
            cartItems.add(cart);
            amountToPay += (qty * book.getPrice());
            pw.println(getRowData(cart));
        }
```

```
        // set cartItems and amountToPay in the session
        session.setAttribute("cartItems", cartItems);
        session.setAttribute("amountToPay", amountToPay);

        if (amountToPay > 0) {
            pw.println("    <tr style='background-color:green'>\r\n"
                    + "      <th scope=\"row\" colspan='5' style='color:yellow; text-align:center;'>
Total Amount To Pay </th>\r\n"
                    + "      <td colspan='1' style='color:white; font-
weight:bold'><span>&#8377;</span> "
                    + amountToPay
                    + "</td>\r\n"
                    + "    </tr>\r\n");
        }
        pw.println("  </tbody>\r\n"
                + "</table>");
        if (amountToPay > 0) {
            pw.println("<div style='text-align:right; margin-right:20px;'>\r\n"
                    + "<form action=\"checkout\" method=\"post\">"
                    + "<input type='submit' class=\"btn btn-primary\" name='pay' value='Proceed to
Pay &#8377; "
                    + amountToPay + "'/></form>"
                    + "    </div>");
        }
    } catch (Exception e) {
```

```
            e.printStackTrace();
        }
    }

    public String getRowData(Cart cart) {
        Book book = cart.getBook();
        return "    <tr>\r\n"
                + "      <th scope=\"row\">" + book.getBarcode() + "</th>\r\n"
                + "      <td>" + book.getName() + "</td>\r\n"
                + "      <td>" + book.getAuthor() + "</td>\r\n"
                + "      <td><span>&#8377;</span> " + book.getPrice() + "</td>\r\n"
                + "      <td><form method='post' action='cart'><button type='submit'
name='removeFromCart' class=\"glyphicon glyphicon-minus btn btn-danger\"></button> "
                + "<input type='hidden' name='selectedBookId' value='" + book.getBarcode() + "'/>"
                + cart.getQuantity()
                + " <button type='submit' name='addToCart' class=\"glyphicon glyphicon-plus btn btn-
success\"></button></form></td>\r\n"
                + "      <td><span>&#8377;</span> " + (book.getPrice() * cart.getQuantity()) +
"</td>\r\n"
                + "    </tr>\r\n";
    }
}
```

REMOVE BOOK

```
package servlets;


import java.io.IOException;
```

```
import java.io.PrintWriter;


import javax.servlet.RequestDispatcher;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;


import com.bittercode.constant.ResponseCode;
import com.bittercode.model.UserRole;
import com.bittercode.service.BookService;
import com.bittercode.service.impl.BookServiceImpl;
import com.bittercode.util.StoreUtil;


public class RemoveBookServlet extends HttpServlet {


    BookService bookService = new BookServiceImpl();


    public void service(HttpServletRequest req, HttpServletResponse res) throws IOException,
ServletException {
        PrintWriter pw = res.getWriter();
        res.setContentType("text/html");
        if (!StoreUtil.isLoggedIn(UserRole.SELLER, req.getSession())) {
            RequestDispatcher rd = req.getRequestDispatcher("SellerLogin.html");
            rd.include(req, res);
```

```
            pw.println("<table class=\"tab\"><tr><td>Please Login First to
Continue!!</td></tr></table>");

            return;

        }


        try {

            String bookId = req.getParameter("bookId");

            RequestDispatcher rd = req.getRequestDispatcher("SellerHome.html");

            rd.include(req, res);

            StoreUtil.setActiveTab(pw, "removebook");

            pw.println("<div class='container'>");

            if (bookId == null || bookId.isBlank()) {

                // render the remove book form;

                showRemoveBookForm(pw);

                return;

            } // else continue


            String responseCode = bookService.deleteBookById(bookId.trim());

            if (ResponseCode.SUCCESS.name().equalsIgnoreCase(responseCode)) {

                pw.println("<table class=\"tab my-5\"><tr><td>Book Removed
Successfully</td></tr></table>");

                pw.println(

                        "<table class=\"tab\"><tr><td><a href=\"removebook\">Remove more
Books</a></td></tr></table>");


            } else {
```

```
            pw.println("<table class=\"tab my-5\"><tr><td>Book Not Available In The
Store</td></tr></table>");

            pw.println(

                  "<table class=\"tab\"><tr><td><a href=\"removebook\">Remove more
Books</a></td></tr></table>");

         }

         pw.println("</div>");

      } catch (Exception e) {

         e.printStackTrace();

         pw.println("<table class=\"tab\"><tr><td>Failed to Remove Books! Try
Again</td></tr></table>");

      }

   }


   private static void showRemoveBookForm(PrintWriter pw) {

      String form = "<form action=\"removebook\" method=\"post\" class='my-5'>\r\n"

            + "      <table class=\"tab\">\r\n"

            + "      <tr>\r\n"

            + "         <td>\r\n"

            + "            <label for=\"bookCode\">Enter BookId to Remove </label>\r\n"

            + "            <input type=\"text\" name=\"bookId\" placeholder=\"Enter Book Id\"
id=\"bookCode\" required>\r\n"

            + "            <input class=\"btn btn-danger my-2\" type=\"submit\" value=\"Remove
Book\">\r\n"

            + "         </td>\r\n"

            + "      </tr>\r\n"

            + "\r\n"
```

```
            + "        </table>\r\n"
            + "    </form>";
        pw.println(form);
    }
}
```

CHECKOUT

```
package servlets;

import java.io.IOException;
import java.io.PrintWriter;

import javax.servlet.RequestDispatcher;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import com.bittercode.constant.BookStoreConstants;
import com.bittercode.model.UserRole;
import com.bittercode.util.StoreUtil;

public class CheckoutServlet extends HttpServlet {
    public void doPost(HttpServletRequest req, HttpServletResponse res) throws IOException,
ServletException {
        PrintWriter pw = res.getWriter();
```

```
        res.setContentType(BookStoreConstants.CONTENT_TYPE_TEXT_HTML);
        if (!StoreUtil.isLoggedIn(UserRole.CUSTOMER, req.getSession())) {
            RequestDispatcher rd = req.getRequestDispatcher("CustomerLogin.html");
            rd.include(req, res);
            pw.println("<table class=\"tab\"><tr><td>Please Login First to
Continue!!</td></tr></table>");
            return;
        }
        try {

            RequestDispatcher rd = req.getRequestDispatcher("payment.html");
            rd.include(req, res);
            StoreUtil.setActiveTab(pw, "cart");
            pw.println("Total Amount<span class=\"price\" style=\"color: black\"><b>&#8377; "
                    + req.getSession().getAttribute("amountToPay")
                    + "</b></span>");

            pw.println("<input type=\"submit\" value=\"Pay & Place Order\" class=\"btn\">"
                    + "</form>");

            pw.println("</div>\r\n"
                    + " </div>\r\n"
                    + " </div>\r\n"
                    + " </div>");
        } catch (Exception e) {
```

```
            e.printStackTrace();
        }
    }

}
```

PROCESS PAYMENT

```
package servlets;

import java.io.IOException;
import java.io.PrintWriter;
import java.util.List;

import javax.servlet.RequestDispatcher;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;

import com.bittercode.constant.BookStoreConstants;
import com.bittercode.model.Book;
import com.bittercode.model.Cart;
import com.bittercode.model.UserRole;
import com.bittercode.service.BookService;
import com.bittercode.service.impl.BookServiceImpl;
```

```
import com.bittercode.util.StoreUtil;


public class ProcessPaymentServlet extends HttpServlet {


    BookService bookService = new BookServiceImpl();


    @SuppressWarnings("unchecked")

    public void service(HttpServletRequest req, HttpServletResponse res) throws IOException,
ServletException {

        PrintWriter pw = res.getWriter();

        res.setContentType(BookStoreConstants.CONTENT_TYPE_TEXT_HTML);

        if (!StoreUtil.isLoggedIn(UserRole.CUSTOMER, req.getSession())) {

            RequestDispatcher rd = req.getRequestDispatcher("CustomerLogin.html");

            rd.include(req, res);

            pw.println("<table class=\"tab\"><tr><td>Please Login First to
Continue!!</td></tr></table>");

            return;

        }

        try {


            RequestDispatcher rd = req.getRequestDispatcher("CustomerHome.html");

            rd.include(req, res);

            StoreUtil.setActiveTab(pw, "cart");

            pw.println("<div id='topmid' style='background-color:grey'>Your Orders</div>");

            pw.println("<div class=\"container\">\r\n"

                    + "        <div class=\"card-columns\">");
```

```
        HttpSession session = req.getSession();
        List<Cart> cartItems = null;
        if (session.getAttribute("cartItems") != null)
            cartItems = (List<Cart>) session.getAttribute("cartItems");
        for (Cart cart : cartItems) {
            Book book = cart.getBook();
            double bPrice = book.getPrice();
            String bCode = book.getBarcode();
            String bName = book.getName();
            String bAuthor = book.getAuthor();
            int availableQty = book.getQuantity();
            int qtToBuy = cart.getQuantity();
            availableQty = availableQty - qtToBuy;
            bookService.updateBookQtyById(bCode, availableQty);
            pw.println(this.addBookToCard(bCode, bName, bAuthor, bPrice, availableQty));
            session.removeAttribute("qty_" + bCode);
        }
        session.removeAttribute("amountToPay");
        session.removeAttribute("cartItems");
        session.removeAttribute("items");
        session.removeAttribute("selectedBookId");
        pw.println("</div>\r\n"
                + "    </div>");
    } catch (Exception e) {
        e.printStackTrace();
```

```
    }
  }

  public String addBookToCard(String bCode, String bName, String bAuthor, double bPrice, int
bQty) {
    String button = "<a href=\"#\" class=\"btn btn-info\">Order Placed</a>\r\n";
    return "<div class=\"card\">\r\n"
        + "            <div class=\"row card-body\">\r\n"
        + "              <img class=\"col-sm-6\" src=\"logo.png\" alt=\"Card image cap\">\r\n"
        + "              <div class=\"col-sm-6\">\r\n"
        + "                <h5 class=\"card-title text-success\">" + bName + "</h5>\r\n"
        + "                <p class=\"card-text\">\r\n"
        + "                Author: <span class=\"text-primary\" style=\"font-weight:bold;\"> "
+ bAuthor
        + "</span><br>\r\n"
        + "                </p>\r\n"
        + "                \r\n"
        + "              </div>\r\n"
        + "            </div>\r\n"
        + "            <div class=\"row card-body\">\r\n"
        + "              <div class=\"col-sm-6\">\r\n"
        + "                <p class=\"card-text\">\r\n"
        + "                <span style='color:blue;'>Order Id: ORD" + bCode + "TM
</span>\r\n"
        + "                <br><span class=\"text-danger\">Item Yet to be
Delivered</span>\r\n"
        + "                </p>\r\n"
```

```
            + "                </div>\r\n"
            + "                <div class=\"col-sm-6\">\r\n"
            + "                    <p class=\"card-text\">\r\n"
            + "                    Amout Paid: <span style=\"font-weight:bold; color:green\">
&amp;#8377; " + bPrice
            + " </span>\r\n"
            + "                    </p>\r\n"
            + button
            + "                </div>\r\n"
            + "            </div>\r\n"
            + "        </div>";
    }
}
```

RECEIPT

```
package servlets;


import java.io.IOException;
import java.io.PrintWriter;
import java.util.List;


import javax.servlet.RequestDispatcher;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
```

```
import javax.servlet.http.HttpServletResponse;


import com.bittercode.constant.BookStoreConstants;
import com.bittercode.model.Book;
import com.bittercode.model.UserRole;
import com.bittercode.service.BookService;
import com.bittercode.service.impl.BookServiceImpl;
import com.bittercode.util.StoreUtil;


public class ReceiptServlet extends HttpServlet {
   BookService bookService = new BookServiceImpl();


   //NOT_IN_USED
   public void service(HttpServletRequest req, HttpServletResponse res) throws IOException,
ServletException {
      PrintWriter pw = res.getWriter();
      res.setContentType(BookStoreConstants.CONTENT_TYPE_TEXT_HTML);
      if (!StoreUtil.isLoggedIn(UserRole.CUSTOMER, req.getSession())) {
         RequestDispatcher rd = req.getRequestDispatcher("CustomerLogin.html");
         rd.include(req, res);
         pw.println("<table class=\"tab\"><tr><td>Please Login First to
Continue!!</td></tr></table>");
         return;
      }
      try {
         List<Book> books = bookService.getAllBooks();
```

```
int i = 0;
RequestDispatcher rd = req.getRequestDispatcher("CustomerHome.html");
rd.include(req, res);
StoreUtil.setActiveTab(pw, "cart");
pw.println("<div class=\"tab\">Your order status is as below</div>");
pw.println(
      "<div class=\"tab\">\r\n" + "                <table>\r\n" + "
<tr>\r\n" + "                                \r\n"
            + "                                <th>Book Code</th>\r\n" + "
<th>Book Name</th>\r\n"
            + "                                <th>Book Author</th>\r\n" + "
<th>Book Price</th>\r\n"
            + "                                <th>Quantity</th><br/>\r\n" + "
<th>Amount</th><br/>\r\n"
            + "                        </tr>");
double total = 0.0;
for (Book book : books) {
   double bPrice = book.getPrice();
   String bCode = book.getBarcode();
   String bName = book.getName();
   String bAuthor = book.getAuthor();
   int bQty = book.getQuantity();
   i = i + 1;

   String qt = "qty" + Integer.toString(i);
   int quantity = Integer.parseInt(req.getParameter(qt));
   try {
```

```
                String check1 = "checked" + Integer.toString(i);
                String getChecked = req.getParameter(check1);
                if (bQty < quantity) {
                    pw.println(
                            "</table><div class=\"tab\" style='color:red;'>Please Select the Qty less than Available Books Quantity</div>");
                    break;
                }

                if (getChecked.equals("pay")) {
                    pw.println("<tr><td>" + bCode + "</td>");
                    pw.println("<td>" + bName + "</td>");
                    pw.println("<td>" + bAuthor + "</td>");
                    pw.println("<td>" + bPrice + "</td>");
                    pw.println("<td>" + quantity + "</td>");
                    double amount = bPrice * quantity;
                    total = total + amount;
                    pw.println("<td>" + amount + "</td></tr>");
                    bQty = bQty - quantity;
                    System.out.println(bQty);
                    bookService.updateBookQtyById(bCode, bQty);
                }
            } catch (Exception e) {
            }
        }
```

```
            pw.println("</table><br/><div class='tab'>Total Paid Amount: " + total + "</div>");
//            String fPay = req.getParameter("f_pay");
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

ERROR HANDLER

```
package servlets;

import java.io.IOException;
import java.io.PrintWriter;
import java.util.Optional;

import javax.servlet.RequestDispatcher;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import com.bittercode.constant.ResponseCode;
import com.bittercode.model.StoreException;
import com.bittercode.model.UserRole;
import com.bittercode.util.StoreUtil;
```

```
public class ErrorHandlerServlet extends HttpServlet {


   public void service(HttpServletRequest req, HttpServletResponse res) throws IOException,
ServletException {

      PrintWriter pw = res.getWriter();

      res.setContentType("text/html");


      // Fetch the exceptions

      Throwable throwable = (Throwable) req.getAttribute("javax.servlet.error.exception");

      Integer statusCode = (Integer) req.getAttribute("javax.servlet.error.status_code");

      String servletName = (String) req.getAttribute("javax.servlet.error.servlet_name");

      String requestUri = (String) req.getAttribute("javax.servlet.error.request_uri");

      String errorMessage = ResponseCode.INTERNAL_SERVER_ERROR.getMessage();

      String errorCode = ResponseCode.INTERNAL_SERVER_ERROR.name();


      if (statusCode == null)

         statusCode = 0;

      Optional<ResponseCode> errorCodes =
ResponseCode.getMessageByStatusCode(statusCode);

      if (errorCodes.isPresent()) {

         errorMessage = errorCodes.get().getMessage();

         errorCode = errorCodes.get().name();

      }


      if (throwable != null && throwable instanceof StoreException) {

         StoreException storeException = (StoreException) throwable;
```

```
    if (storeException != null) {
        errorMessage = storeException.getMessage();
        statusCode = storeException.getStatusCode();
        errorCode = storeException.getErrorCode();
        storeException.printStackTrace();
    }
}

System.out.println("======ERROR TRIGGERED========");
System.out.println("Servlet Name: " + servletName);
System.out.println("Request URI: " + requestUri);
System.out.println("Status Code: " + statusCode);
System.out.println("Error Code: " + errorCode);
System.out.println("Error Message: " + errorMessage);
System.out.println("============================");

if (StoreUtil.isLoggedIn(UserRole.CUSTOMER, req.getSession())) {
    RequestDispatcher rd = req.getRequestDispatcher("CustomerHome.html");
    rd.include(req, res);
    StoreUtil.setActiveTab(pw, "home");
    showErrorMessage(pw, errorCode, errorMessage);

} else if (StoreUtil.isLoggedIn(UserRole.SELLER, req.getSession())) {
    RequestDispatcher rd = req.getRequestDispatcher("SellerHome.html");
    rd.include(req, res);
```

```
            StoreUtil.setActiveTab(pw, "home");
            showErrorMessage(pw, errorCode, errorMessage);

        } else {
            RequestDispatcher rd = req.getRequestDispatcher("index.html");
            rd.include(req, res);
            pw.println("<script>"
                    + "document.getElementById('topmid').innerHTML=';'
                    + "document.getElementById('happy').innerHTML=';'
                    + "</script>");
            showErrorMessage(pw, errorCode, errorMessage);
        }

    }

    private void showErrorMessage(PrintWriter pw, String errorCode, String errorMessage) {
        pw.println("<div class='container my-5'>"
                + "<div class=\"alert alert-success\" role=\"alert\" style='max-width:450px; text-
align:center; margin:auto;'>\r\n"
                + "  <h4 class=\"alert-heading\">"
                + errorCode
                + "</h4>\r\n"
                + "  <hr>\r\n"
                + "  <p class=\"mb-0\">"
                + errorMessage
```

```
            + "</p>\r\n"
            + "</div>"
            + "</div>");

    }

}
```

SELLER LOGIN

```
package servlets;

import java.io.IOException;
import java.io.PrintWriter;

import javax.servlet.RequestDispatcher;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import com.bittercode.constant.BookStoreConstants;
import com.bittercode.constant.db.UsersDBConstants;
import com.bittercode.model.User;
import com.bittercode.model.UserRole;
import com.bittercode.service.UserService;
import com.bittercode.service.impl.UserServiceImpl;
```

```
public class SellerLoginServlet extends HttpServlet {


    UserService userService = new UserServiceImpl();


    public void doPost(HttpServletRequest req, HttpServletResponse res) throws IOException,
ServletException {
        PrintWriter pw = res.getWriter();
        res.setContentType(BookStoreConstants.CONTENT_TYPE_TEXT_HTML);
        String uName = req.getParameter(UsersDBConstants.COLUMN_USERNAME);
        String pWord = req.getParameter(UsersDBConstants.COLUMN_PASSWORD);
        try {
            User user = userService.login(UserRole.SELLER, uName, pWord, req.getSession());
            if (user != null) {
                RequestDispatcher rd = req.getRequestDispatcher("SellerHome.html");


                rd.include(req, res);
                pw.println("    <div id=\"topmid\"><h1>Welcome to Online <br>Book
Store</h1></div>\r\n"
                        + "    <br>\r\n"
                        + "    <table class=\"tab\">\r\n"
                        + "        <tr>\r\n"
                        + "            <td><p>Welcome "+user.getFirstName()+", Happy Learning
!!</p></td>\r\n"
                        + "        </tr>\r\n"
                        + "    </table>");
```

```
            } else {

                RequestDispatcher rd = req.getRequestDispatcher("SellerLogin.html");
                rd.include(req, res);
                pw.println("<div class=\"tab\">Incorrect UserName or PassWord</div>");
            }

        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

STORE BOOK

```
package servlets;

import java.io.IOException;
import java.io.PrintWriter;
import java.util.List;

import javax.servlet.RequestDispatcher;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
```

```
import com.bittercode.model.Book;

import com.bittercode.model.UserRole;

import com.bittercode.service.BookService;

import com.bittercode.service.impl.BookServiceImpl;

import com.bittercode.util.StoreUtil;


public class StoreBookServlet extends HttpServlet {


    // book service for database operations and logics

    BookService bookService = new BookServiceImpl();


    public void service(HttpServletRequest req, HttpServletResponse res) throws IOException,
ServletException {

        PrintWriter pw = res.getWriter();

        res.setContentType("text/html");


        // Check if the customer is logged in, or else return to login page

        if (!StoreUtil.isLoggedIn(UserRole.SELLER, req.getSession())) {

            RequestDispatcher rd = req.getRequestDispatcher("SellerLogin.html");

            rd.include(req, res);

            pw.println("<table class=\"tab\"><tr><td>Please Login First to
Continue!!</td></tr></table>");

            return;

        }

        try {
```

```
        // Add/Remove Item from the cart if requested
        // store the comma separated bookIds of cart in the session
        // StoreUtil.updateCartItems(req);

        RequestDispatcher rd = req.getRequestDispatcher("SellerHome.html");
        rd.include(req, res);
        pw.println("<div class='container'>");
        // Set the active tab as cart
        StoreUtil.setActiveTab(pw, "storebooks");

        // Read the books from the database with the respective bookIds
        List<Book> books = bookService.getAllBooks();
        pw.println("<div id='topmid' style='background-color:grey'>Books Available In the
Store</div>");
        pw.println("<table class=\"table table-hover\" style='background-color:white'>\r\n"
              + "  <thead>\r\n"
              + "    <tr style='background-color:black; color:white;'>\r\n"
              + "      <th scope=\"col\">BookId</th>\r\n"
              + "      <th scope=\"col\">Name</th>\r\n"
              + "      <th scope=\"col\">Author</th>\r\n"
              + "      <th scope=\"col\">Price</th>\r\n"
              + "      <th scope=\"col\">Quantity</th>\r\n"
              + "      <th scope=\"col\">Action</th>\r\n"
              + "    </tr>\r\n"
```

```
                + "  </thead>\r\n"
                + "  <tbody>\r\n");
        if (books == null || books.size() == 0) {
            pw.println("    <tr style='background-color:green'>\r\n"
                    + "      <th scope=\"row\" colspan='6' style='color:yellow; text-align:center;'> No Books Available in the store </th>\r\n"
                    + "    </tr>\r\n");
        }
        for (Book book : books) {
            pw.println(getRowData(book));
        }

        pw.println("  </tbody>\r\n"
                + "</table></div>");

    } catch (Exception e) {
        e.printStackTrace();
    }
}

public String getRowData(Book book) {
    return "    <tr>\r\n"
            + "      <th scope=\"row\">" + book.getBarcode() + "</th>\r\n"
            + "      <td>" + book.getName() + "</td>\r\n"
            + "      <td>" + book.getAuthor() + "</td>\r\n"
```

```
            + "      <td><span>&#8377;</span> " + book.getPrice() + "</td>\r\n"
            + "      <td>"
            + book.getQuantity()
            + "      </td>\r\n"
            + "      <td><form method='post' action='updatebook'>"
            + "          <input type='hidden' name='bookId' value='" + book.getBarcode() + "'/>"
            + "          <button type='submit' class=\"btn btn-success\">Update</button>"
            + "          </form>"
            + "    </tr>\r\n";
    }

}
```

UPDATE BOOK

```
package servlets;

import java.io.IOException;
import java.io.PrintWriter;

import javax.servlet.RequestDispatcher;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
```

```
import com.bittercode.constant.BookStoreConstants;

import com.bittercode.constant.ResponseCode;

import com.bittercode.constant.db.BooksDBConstants;

import com.bittercode.model.Book;

import com.bittercode.model.UserRole;

import com.bittercode.service.BookService;

import com.bittercode.service.impl.BookServiceImpl;

import com.bittercode.util.StoreUtil;


public class UpdateBookServlet extends HttpServlet {

   BookService bookService = new BookServiceImpl();


   public void service(HttpServletRequest req, HttpServletResponse res) throws IOException,
ServletException {

      PrintWriter pw = res.getWriter();

      res.setContentType(BookStoreConstants.CONTENT_TYPE_TEXT_HTML);


      if (!StoreUtil.isLoggedIn(UserRole.SELLER, req.getSession())) {

         RequestDispatcher rd = req.getRequestDispatcher("SellerLogin.html");

         rd.include(req, res);

         pw.println("<table class=\"tab\"><tr><td>Please Login First to
Continue!!</td></tr></table>");

         return;

      }


      RequestDispatcher rd = req.getRequestDispatcher("SellerHome.html");
```

```
        rd.include(req, res);

        StoreUtil.setActiveTab(pw, "storebooks");

        pw.println("<div class='container my-2'>");


        try {

            if (req.getParameter("updateFormSubmitted") != null) {

                String bName = req.getParameter(BooksDBConstants.COLUMN_NAME);

                String bCode = req.getParameter(BooksDBConstants.COLUMN_BARCODE);

                String bAuthor = req.getParameter(BooksDBConstants.COLUMN_AUTHOR);

                double bPrice =
Double.parseDouble(req.getParameter(BooksDBConstants.COLUMN_PRICE));

                int bQty =
Integer.parseInt(req.getParameter(BooksDBConstants.COLUMN_QUANTITY));


                Book book = new Book(bCode, bName, bAuthor, bPrice, bQty);

                String message = bookService.updateBook(book);

                if (ResponseCode.SUCCESS.name().equalsIgnoreCase(message)) {

                    pw.println(

                            "<table class=\"tab\"><tr><td>Book Detail Updated
Successfully!</td></tr></table>");

                } else {

                    pw.println("<table class=\"tab\"><tr><td>Failed to Update
Book!!</td></tr></table>");

                    // rd.include(req, res);

                }


                return;
```

```
            }

            String bookId = req.getParameter("bookId");

            if (bookId != null) {
                Book book = bookService.getBookById(bookId);
                showUpdateBookForm(pw, book);
            }

        } catch (Exception e) {
            e.printStackTrace();
            pw.println("<table class=\"tab\"><tr><td>Failed to Load Book
data!!</td></tr></table>");
        }
    }

    private static void showUpdateBookForm(PrintWriter pw, Book book) {
        String form = "<table class=\"tab my-5\" style=\"width:40%;\">\r\n"
                + "        <tr>\r\n"
                + "            <td>\r\n"
                + "                <form action=\"updatebook\" method=\"post\">\r\n"
                + "                    <label for=\"bookCode\">Book Code : </label><input type=\"text\"
name=\"barcode\" id=\"bookCode\" placeholder=\"Enter Book Code\" value="'"
                + book.getBarcode() + "' readonly><br/>"
                + "                    <label for=\"bookName\">Book Name : </label> <input type=\"text\"
name=\"name\" id=\"bookName\" placeholder=\"Enter Book's name\" value="'"
```

```
            + book.getName() + "' required><br/>\r\n"

            + "                <label for=\"bookAuthor\">Book Author : </label><input type=\"text\"
name=\"author\" id=\"bookAuthor\" placeholder=\"Enter Author's Name\" value='"

            + book.getAuthor() + "' required><br/>\r\n"

            + "                <label for=\"bookPrice\">Book Price : </label><input type=\"number\"
name=\"price\" placeholder=\"Enter the Price\" value='"

            + book.getPrice() + "' required><br/>\r\n"

            + "                <label for=\"bookQuantity\">Book Qnty : </label><input
type=\"number\" name=\"quantity\" id=\"bookQuantity\" placeholder=\"Enter the quantity\"
value='"

            + book.getQuantity() + "' required><br/>\r\n"

            + "                <input class=\"btn btn-success my-2\" type=\"submit\"
name='updateFormSubmitted' value=\" Update Book \">\r\n"

            + "            </form>\r\n"

            + "          </td>\r\n"

            + "       </tr>  \r\n"

            + "    </table>";

      pw.println(form);

   }

}
```

VIEW BOOK

```
package servlets;


import java.io.IOException;

import java.io.PrintWriter;

import java.util.List;
```

```
import javax.servlet.RequestDispatcher;

import javax.servlet.ServletException;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import javax.servlet.http.HttpSession;


import com.bittercode.model.Book;

import com.bittercode.model.UserRole;

import com.bittercode.service.BookService;

import com.bittercode.service.impl.BookServiceImpl;

import com.bittercode.util.StoreUtil;


public class ViewBookServlet extends HttpServlet {


    // book service for database operations and logics

    BookService bookService = new BookServiceImpl();


    public void service(HttpServletRequest req, HttpServletResponse res) throws IOException,
ServletException {

        PrintWriter pw = res.getWriter();

        res.setContentType("text/html");


        // Check if the customer is logged in, or else return to login page
```

```
        if (!StoreUtil.isLoggedIn(UserRole.CUSTOMER, req.getSession())) {
            RequestDispatcher rd = req.getRequestDispatcher("CustomerLogin.html");
            rd.include(req, res);
            pw.println("<table class=\"tab\"><tr><td>Please Login First to
Continue!!</td></tr></table>");
            return;
        }
        try {


            // Read All available books from the database
            List<Book> books = bookService.getAllBooks();


            // Default Page to load data into
            RequestDispatcher rd = req.getRequestDispatcher("CustomerHome.html");
            rd.include(req, res);


            // Set Available Books tab as active
            StoreUtil.setActiveTab(pw, "books");


            // Show the heading for the page
            pw.println("<div id='topmid' style='background-color:grey'>Available Books"
                    + "<form action=\"cart\" method=\"post\" style='float:right; margin-right:20px'>"
                    + "<input type='submit' class=\"btn btn-primary\" name='cart'
value='Proceed'/></form>"
                    + "</div>");
            pw.println("<div class=\"container\">\r\n"
```

```
                + "        <div class=\"card-columns\">");

            // Add or Remove items from the cart, if requested
            StoreUtil.updateCartItems(req);

            HttpSession session = req.getSession();
            for (Book book : books) {

                // Add each book to display as a card
                pw.println(this.addBookToCard(session, book));

            }

            // Checkout Button
            pw.println("</div>"
                    + "<div style='float:auto'><form action=\"cart\" method=\"post\">"
                    + "<input type='submit' class=\"btn btn-success\" name='cart' value='Proceed to
Checkout'/></form>"
                    + "    </div>");

        } catch (Exception e) {
            e.printStackTrace();
        }
    }
```

```
    public String addBookToCard(HttpSession session, Book book) {
        String bCode = book.getBarcode();
        int bQty = book.getQuantity();

        // Quantity of the current book added to the cart
        int cartItemQty = 0;
        if (session.getAttribute("qty_" + bCode) != null) {
            // Quantity of each book in the cart will be added in the session prefixed with
            // 'qty_' following with bookId
            cartItemQty = (int) session.getAttribute("qty_" + bCode);
        }

        // Button To Add/Remove item from the cart
        String button = "";
        if (bQty > 0) {
            // If no items in the cart, show add to cart button
            // If items is added to the cart, then show +, - button to add/remove more items
            button = "<form action=\"viewbook\" method=\"post\">"
                    + "<input type='hidden' name = 'selectedBookId' value = " + bCode + ">"
                    + "<input type='hidden' name='qty_" + bCode + "' value='1'/>"
                    + (cartItemQty == 0
                            ? "<input type='submit' class=\"btn btn-primary\" name='addToCart'
value='Add To Cart'/></form>"
                            : "<form method='post' action='cart'>"
                                    + "<button type='submit' name='removeFromCart' class=\"glyphicon
glyphicon-minus btn btn-danger\"></button> "
```

```
                    + "<input type='hidden' name='selectedBookId' value='" + bCode + "'/>"
                    + cartItemQty
                    + " <button type='submit' name='addToCart' class=\"glyphicon glyphicon-
plus btn btn-success\"></button></form>")
            + "";
    } else {
        // If available Quantity is zero, show out of stock button
        button = "<p class=\"btn btn-danger\">Out Of Stock</p>\r\n";
    }

    // Bootstrap card to show the book data
    return "<div class=\"card\">\r\n"
        + "            <div class=\"row card-body\">\r\n"
        + "                <img class=\"col-sm-6\" src=\"logo.png\" alt=\"Card image cap\">\r\n"
        + "                <div class=\"col-sm-6\">\r\n"
        + "                    <h5 class=\"card-title text-success\">" + book.getName() +
"</h5>\r\n"
        + "                    <p class=\"card-text\">\r\n"
        + "                    Author: <span class=\"text-primary\" style=\"font-weight:bold;\"> "
        + book.getAuthor()
        + "</span><br>\r\n"
        + "                    </p>\r\n"
        + "                    \r\n"
        + "                </div>\r\n"
        + "            </div>\r\n"
        + "            <div class=\"row card-body\">\r\n"
```

```
            + "                 <div class=\"col-sm-6\">\r\n"
            + "                     <p class=\"card-text\">\r\n"
            + "                     <span>Id: " + bCode + "</span>\r\n"
            + (bQty < 20 ? "<br><span class=\"text-danger\">Only " + bQty + " items
left</span>\r\n"
                    : "<br><span class=\"text-success\">Trending</span>\r\n")
            + "                     </p>\r\n"
            + "                 </div>\r\n"
            + "                 <div class=\"col-sm-6\">\r\n"
            + "                     <p class=\"card-text\">\r\n"
            + "                     Price: <span style=\"font-weight:bold; color:green\"> &#8377; "
            + book.getPrice()
            + " </span>\r\n"
            + "                     </p>\r\n"
            + button
            + "                 </div>\r\n"
            + "             </div>\r\n"
            + "         </div>";
    }
}
```

LOGOUT

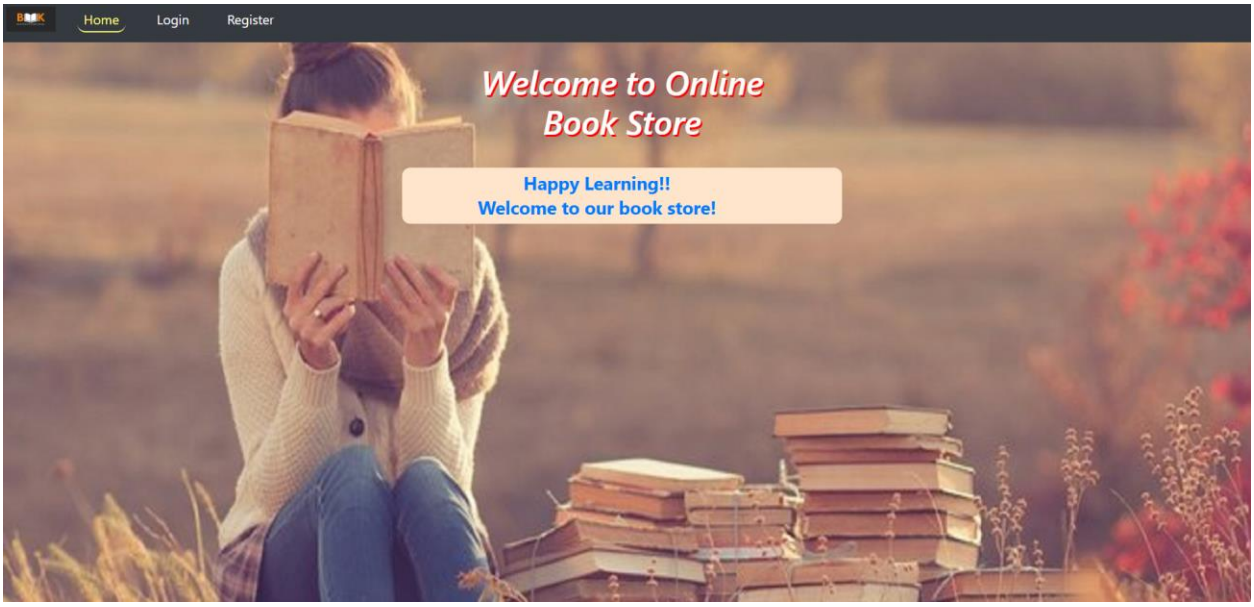
```
package servlets;


import java.io.IOException;
```

```
import java.io.PrintWriter;


import javax.servlet.RequestDispatcher;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;


import com.bittercode.constant.BookStoreConstants;
import com.bittercode.service.UserService;
import com.bittercode.service.impl.UserServiceImpl;


public class LogoutServlet extends HttpServlet {


    UserService authService = new UserServiceImpl();


    public void doGet(HttpServletRequest req, HttpServletResponse res) throws IOException,
ServletException {
        PrintWriter pw = res.getWriter();
        res.setContentType(BookStoreConstants.CONTENT_TYPE_TEXT_HTML);
        try {


            boolean logout = authService.logout(req.getSession());


            RequestDispatcher rd = req.getRequestDispatcher("CustomerLogin.html");
```

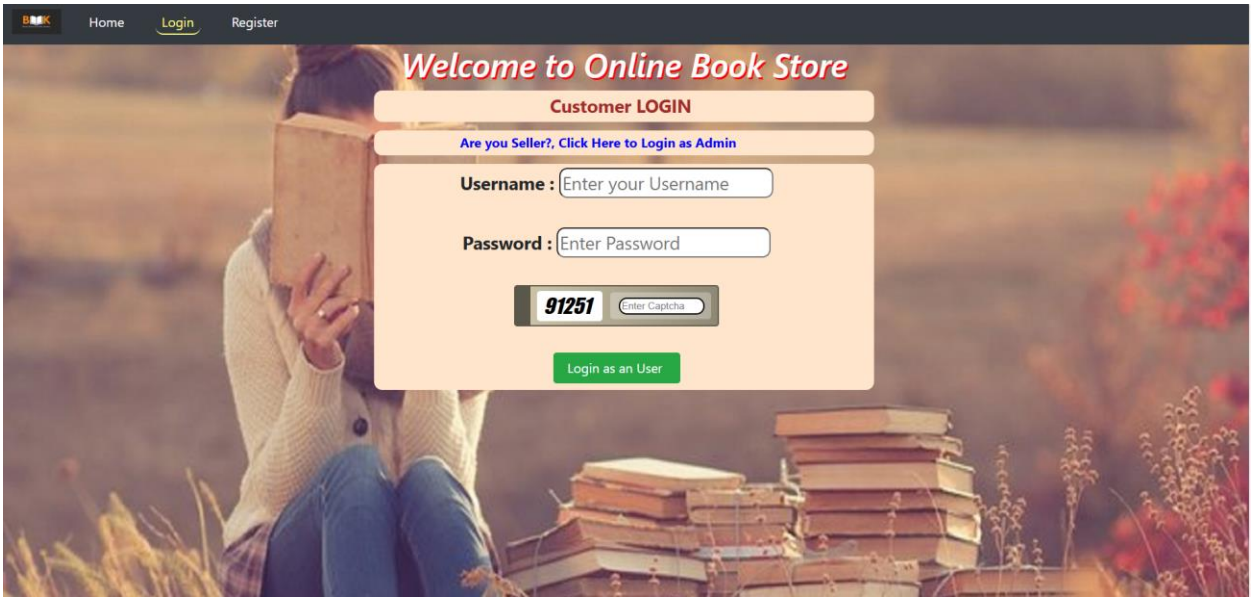
```
            rd.include(req, res);
//              StoreUtil.setActiveTab(pw, "logout");
            if (logout) {
                pw.println("<table class=\"tab\"><tr><td>Successfully logged
out!</td></tr></table>");
            }

        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

V. RESULT AND DISCUSSION

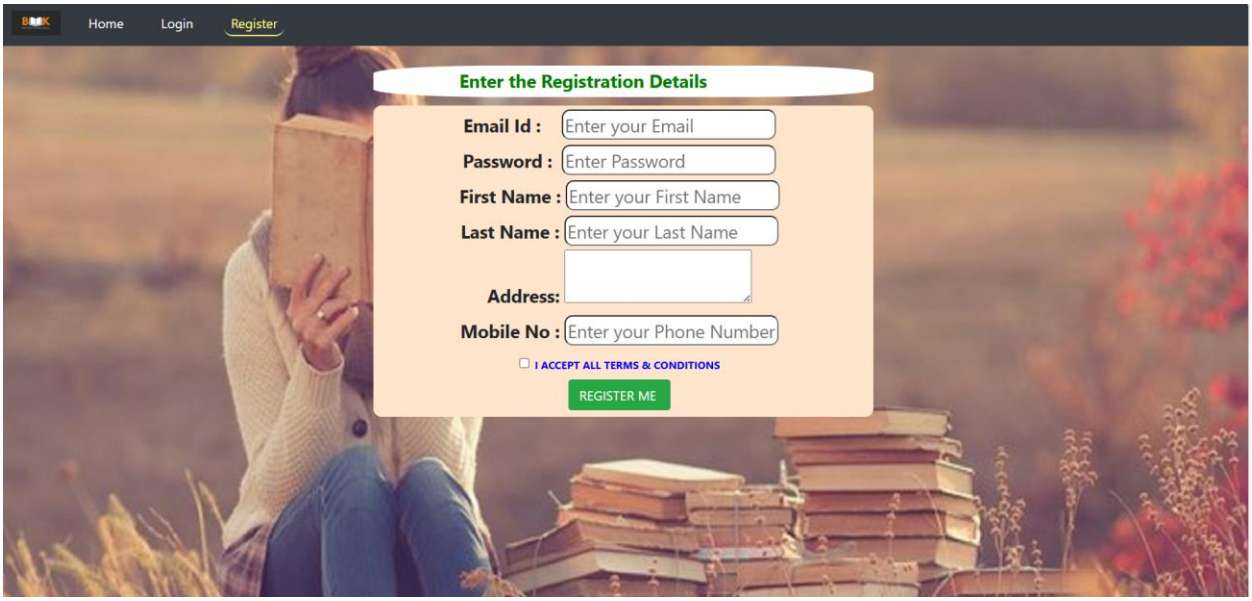
HOME PAGE:



LOGIN PAGE:

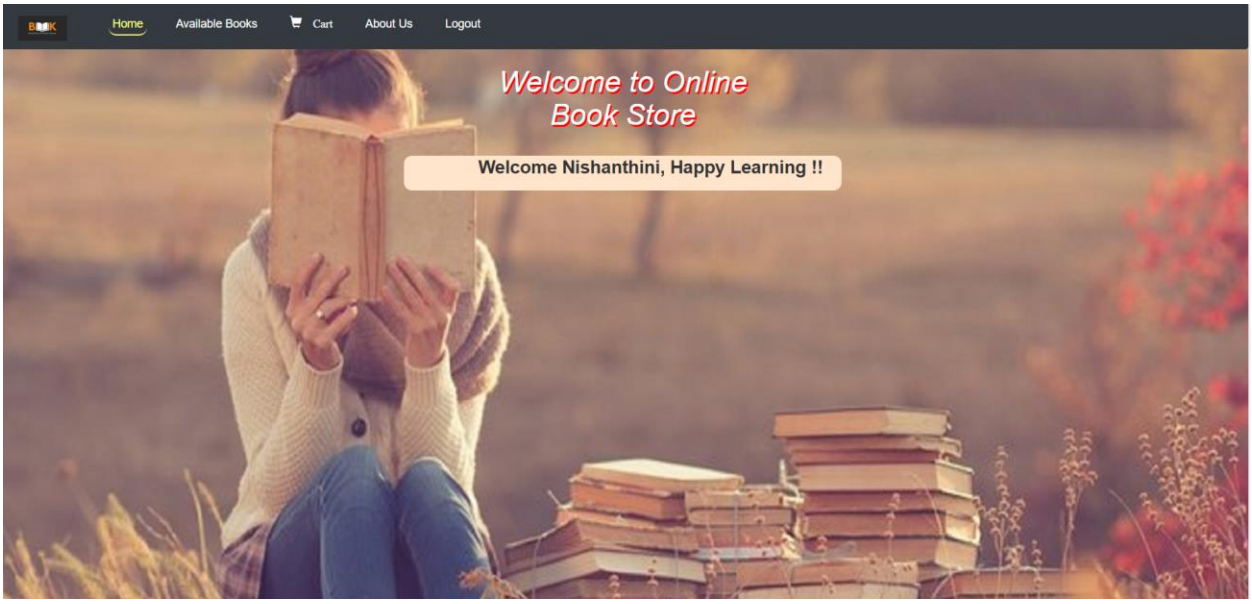


REGISTRATION PAGE:

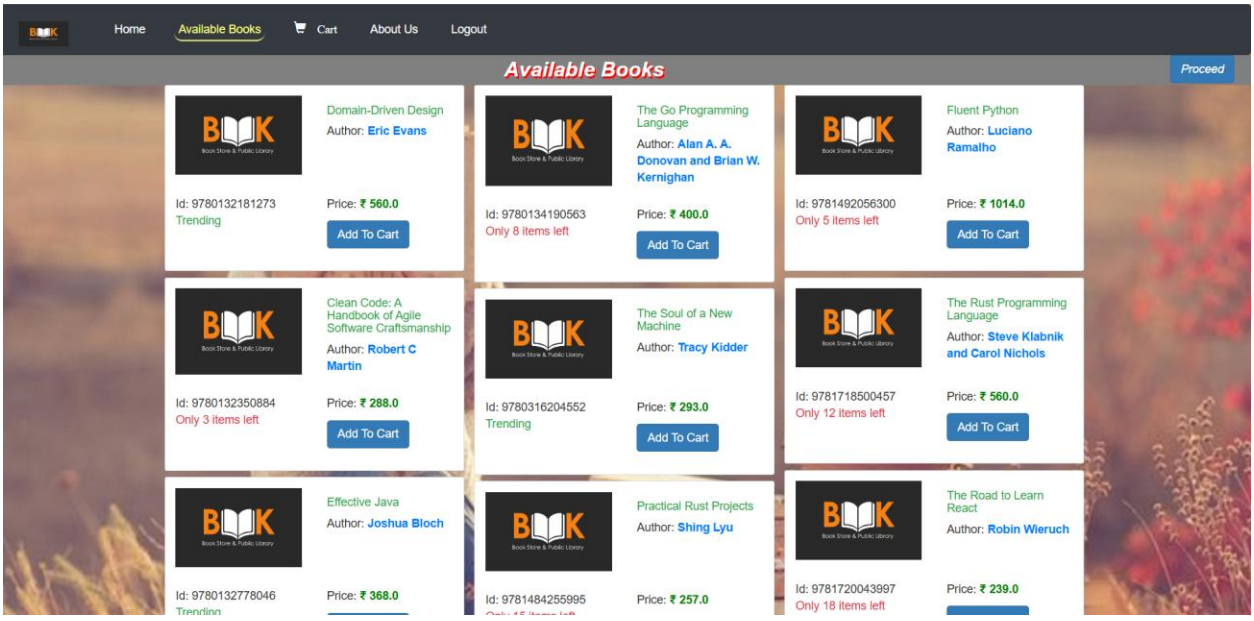


LOGIN PAGE AFTER REGISTRATION:

WELCOME PAGE:

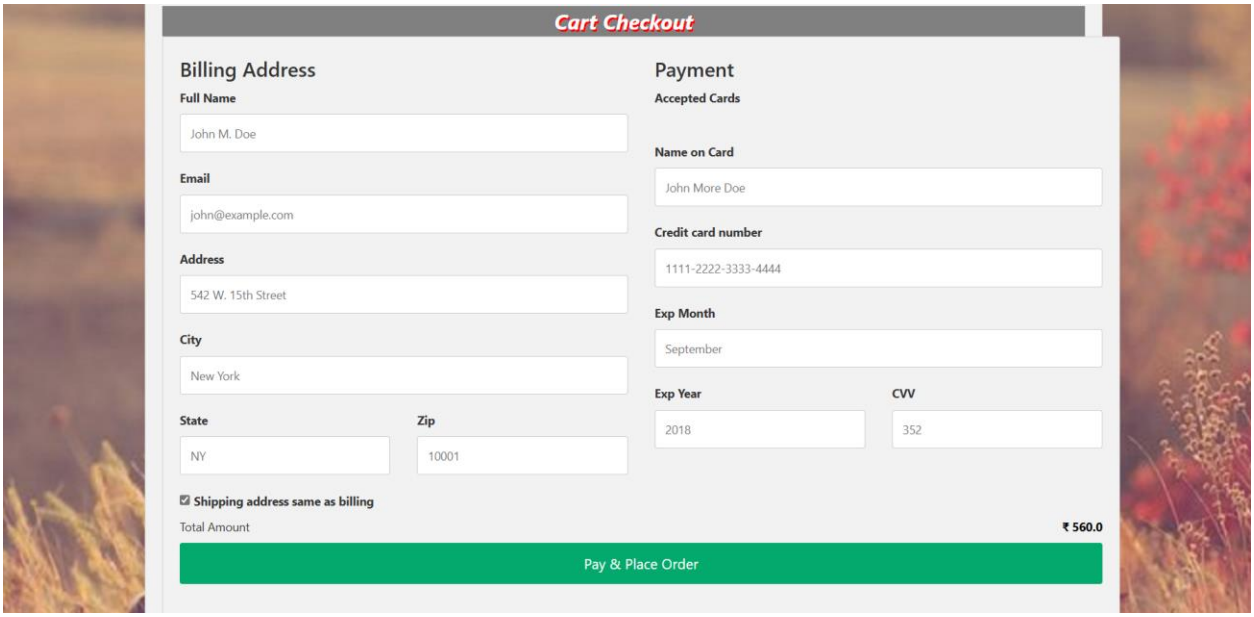


AVAILABLE BOOKS:



CART PAGE:

BILLING ADDRESS AND PAYMENT PAGE:



RESULT

The online book store web application developed using Java, JDBC, and Servlets appears to be a comprehensive and well-designed system that meets the stated requirements. The key features and functionalities implemented in the application include:

1. Selling books online
2. Maintaining book selling history
3. Adding and managing books by the admin
4. User-friendly interface for both admins and regular users
5. Implementation of HTTP Servlets in Java for the backend development
6. Separation of admin and user functionalities with appropriate access control

The application utilizes a robust technology stack comprising Java, JDBC, Servlets for the backend, and HTML, CSS, JavaScript, and Bootstrap for the frontend. The use of a relational database like MySQL is a suitable choice for managing the book and user data.

DISCUSSION

The successful implementation of this online book store web application demonstrates the developer's proficiency in Java, JDBC, Servlets, and web development principles. The separation of admin and user functionalities, along with the inclusion of features like book management, user registration, and payment receipts, showcases a well-thought-out and comprehensive approach to the problem domain.

The use of HTTP Servlets for the backend development is a appropriate choice, as it allows for the implementation of server-side logic and the handling of client-server communication in a scalable and secure manner. The integration of JDBC for database interactions ensures efficient data management and persistence.

The front-end technologies employed, such as HTML, CSS, JavaScript, and Bootstrap, provide a user-friendly and visually appealing interface, which is crucial for the success of an e-commerce

application. The combination of a robust backend and an intuitive frontend creates a well-balanced and functional web application.

While the current implementation meets the stated requirements, there are opportunities for further enhancements and improvements. Some potential areas for development include:

1. Implementing more advanced user authentication and authorization mechanisms, such as those provided by frameworks like Spring Security, to ensure the security and integrity of the system.
2. Integrating a payment gateway, such as PayPal or Stripe, to enable seamless and secure online transactions for the users.
3. Adding search and filtering functionality to the book catalog, allowing users to easily find and browse the available books.
4. Incorporating a user review and rating system, which can help customers make informed purchasing decisions and provide valuable feedback to the book store.
5. Implementing a recommendation system based on user browsing and purchase history, which can enhance the user experience and increase sales.

By addressing these potential improvements, the online book store web application can become even more robust, user-friendly, and commercially viable as an e-commerce platform.

Overall, the successful completion of this mini-project demonstrates the developer's ability to apply their knowledge of Java, JDBC, Servlets, and web development to create a functional and feature-rich online book store. This project can serve as a solid foundation for further enhancements and expansion, showcasing the developer's skills and potential for building more complex and sophisticated web applications.

VI. CONCLUSION

The online book store web application developed using Java, JDBC, and Servlets is a well-designed and feature-rich system that meets the stated requirements. The application provides a robust and user-friendly platform for selling books online, managing book inventory, and handling user interactions.

The key strengths of this mini-project include:

1. Comprehensive Functionality: The application covers a wide range of features, including book management, user registration and authentication, order processing, and payment receipts. This breadth of functionality demonstrates the developer's understanding of the problem domain and their ability to deliver a complete solution.
2. Separation of Concerns: The clear separation of admin and user functionalities, with appropriate access control, showcases a well-structured and secure application design. This approach ensures that the system can be easily maintained and scaled as the business requirements evolve.
3. Robust Technology Stack: The use of Java, JDBC, and Servlets for the backend, combined with HTML, CSS, JavaScript, and Bootstrap for the frontend, provides a solid and scalable technology foundation. This stack allows for the implementation of complex business logic and the creation of a user-friendly interface.
4. Potential for Expansion: While the current implementation meets the stated requirements, there are opportunities for further enhancements, such as integrating advanced authentication mechanisms, payment gateways, search and filtering capabilities, and recommendation systems. These improvements can help the application become more commercially viable and competitive in the e-commerce market.

In conclusion, this online book store web application is a well-executed mini-project that demonstrates the developer's proficiency in Java, JDBC, Servlets, and web development. The application's comprehensive functionality, secure design, and robust technology stack make it a promising foundation for further development and expansion into a commercially viable e-

commerce platform. With continued improvements and innovative features, this project has the potential to become a successful and competitive online book store solution.

VII.REFERENCES

- https://www.researchgate.net/publication/314783622_The_Online_Bookstore
- https://shop4books.co.in/?srsltid=AfmBOop3f3UNNUZ5E9Wmt58CYabh9zrWxysJ1SYTDPtrDDzZyNv7jtVf
- https://www.researchgate.net/publication/380729587_ONLINE_BOOK_STORE_MANAGEMENT_SYSTEM_PROJECT_REPORT
- https://www.scribd.com/presentation/390566986/Online-Book-Store-PPT
- http://utpedia.utp.edu.my/id/eprint/24035/1/Online%20Book%20Store%20System.pdf