1. **Aim of the Microproject:**

The **Online Bookshop Management System** is a Java Swing-based application designed to facilitate book management with a user-friendly interface. It includes essential functionalities such as **user authentication**, **book addition**, **student registration**, **book issuing**, and **returns**. Users can interact with a **graphical interface** featuring buttons, tables, and background images for a seamless experience. The system ensures restricted access with a login mechanism and provides an intuitive navigation system using a **tabbed layout**.

This project focuses on **efficient book management**, allowing users to **add, issue, return, and view books** dynamically. It incorporates **logout functionality** for session management and aims to enhance usability with an appealing design. Future enhancements may include **data persistence using databases** to store book and student records. This microproject serves as a **practical implementation** of Java Swing concepts, event handling, and object-oriented programming, making it a great learning experience for beginners.

1. **Proposed Methodology:**

The proposed **Online Bookshop Management System** follows a structured approach, starting with **user authentication** to restrict access. The login system verifies credentials before granting entry to the main interface, which is designed using **Java Swing** components like JFrame, JTabbedPane, and JTable. Users can navigate through different tabs, including **Home, Books Available, and Functional Panels**, ensuring an intuitive and organized UI. The **book management system** allows users to **view, add, issue, and return books**, with book details displayed in a dynamic table for easy access.

Additional features include **student registration**, enabling users to store student details before issuing books. The **return book feature** ensures proper tracking, while the **logout option** securely ends the session. The UI is enhanced with **background images, color-coded buttons, and structured layouts** for a visually appealing experience. Future improvements may include **data persistence using ArrayLists or a database**, making the system more robust and scalable for real-world applications.

1. **Action Plan :**

|  |  |  |  |
| --- | --- | --- | --- |
| **NO.** | **Activity** | **Description** | **Dates** |
| 1 | Requirement Analysis | Understand features: login (admin/student), registration, book issue, view | 04/03/2025 |
| 2 | Design GUI using Java Swing | Design screens for welcome, login (tabs), book details, registration, issue book | 11/03/2025 |
| 3 | Setup MySQL Database | Create database bookshop\_db and tables: users, books, students, issued\_books | 26/03/2025 |
| 4 | Connect Java with MySQL using JDBC | Use DriverManager and PreparedStatement for DB access | 27/03/2025 |
| 5 | Implement Login and Role-based Access | Admin and Student login with validation; load relevant panels dynamically | 01/04/2025 |
| 6 | Add Book and Delete Book (Admin Only) | Admin can insert/delete book records into/from DB | 04/04/2025 |
| 7 | Student Registration & Book Issue | Students can register and issue books (update quantity) | 07/04/2025 |
| 8 | View Issued Books & Student Info | Admin can view registered students and their issued books | 11/04/2025 |
| 9 | Test all Functionalities | Check all features, validate inputs, test DB updates, fix UI bugs | 12/04/2025 |

1. **Brief Description of Micro project :**

**4.1 Introduction:**

The **Online Bookshop Management System** is a **Java Swing-based** application designed to manage books efficiently. It includes a **secure login system** to restrict access, followed by a **graphical interface** with multiple tabs for different functionalities. Users can **view available books, add new books, register students, issue books, and process returns** using interactive forms and tables. The system ensures smooth navigation with a **JTabbedPane**, allowing users to switch between different sections seamlessly.

Key features include **book management with a JTable**, **student registration**, and a **return system** that helps track issued books. A **logout option** is provided for session management, ensuring security. The UI is enhanced with **background images, structured layouts, and color-coded buttons** for a better user experience.

**4.2 Used Concept:**

The Online Bookshop Management System uses Java Swing for GUI, featuring JTabbedPane for navigation, JTable for book management, and JOptionPane for user interactions. It includes user authentication, event handling for actions like adding, issuing, and returning books, and layout managers for structured UI design. The system ensures secure logout and session handling, with a modular OOP approach for better maintainability.

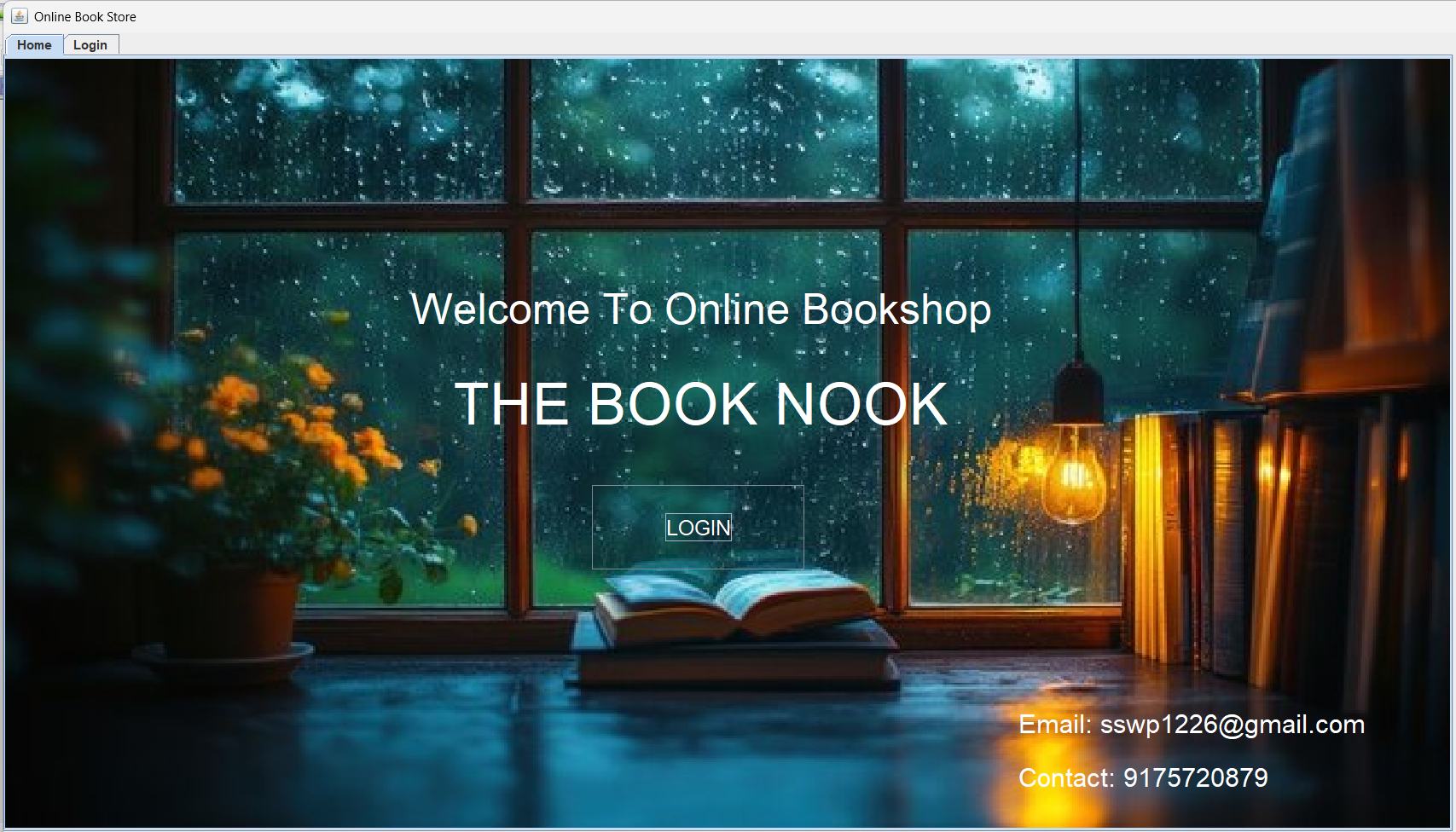
Concepts Used in the Code:

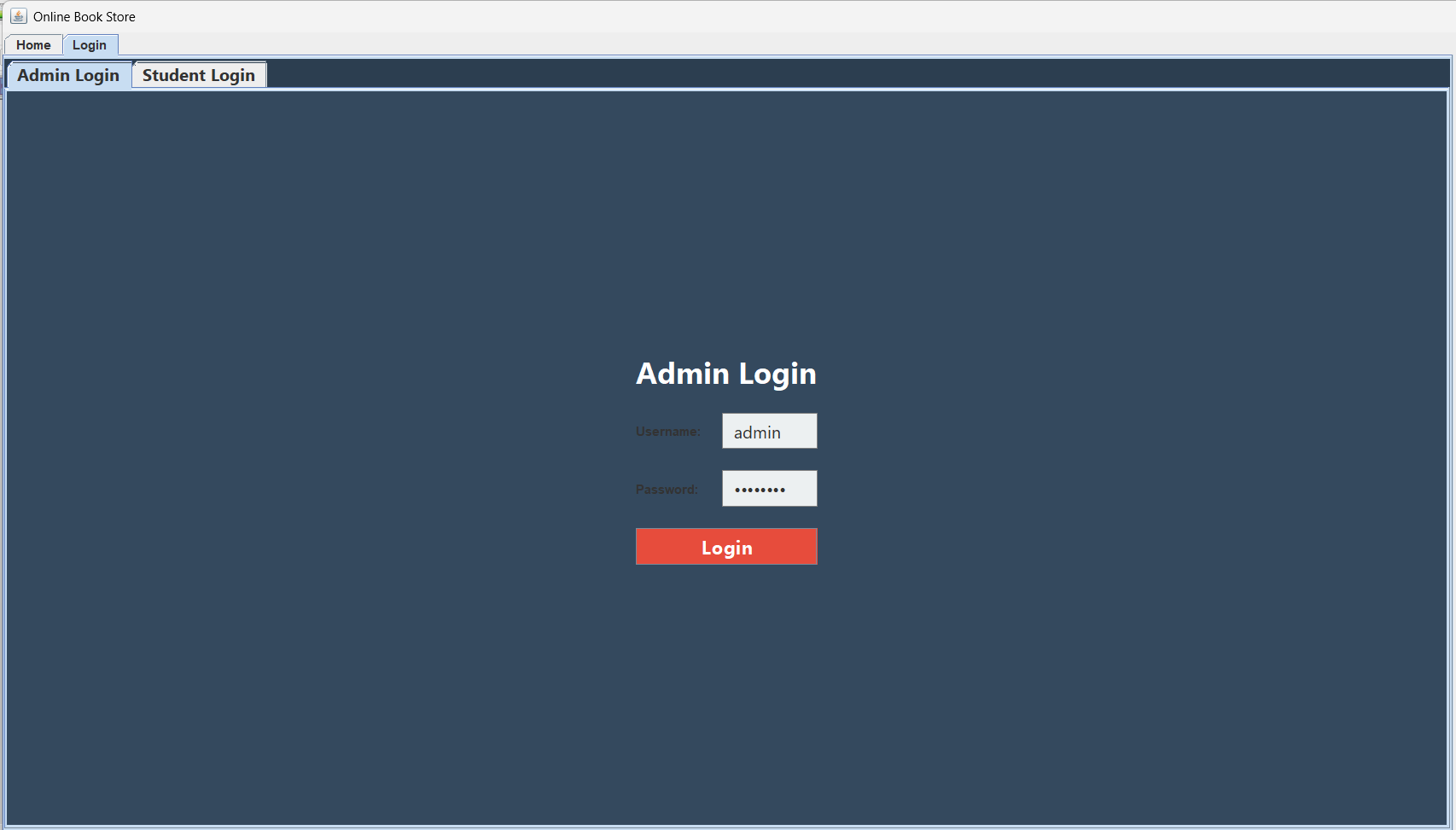
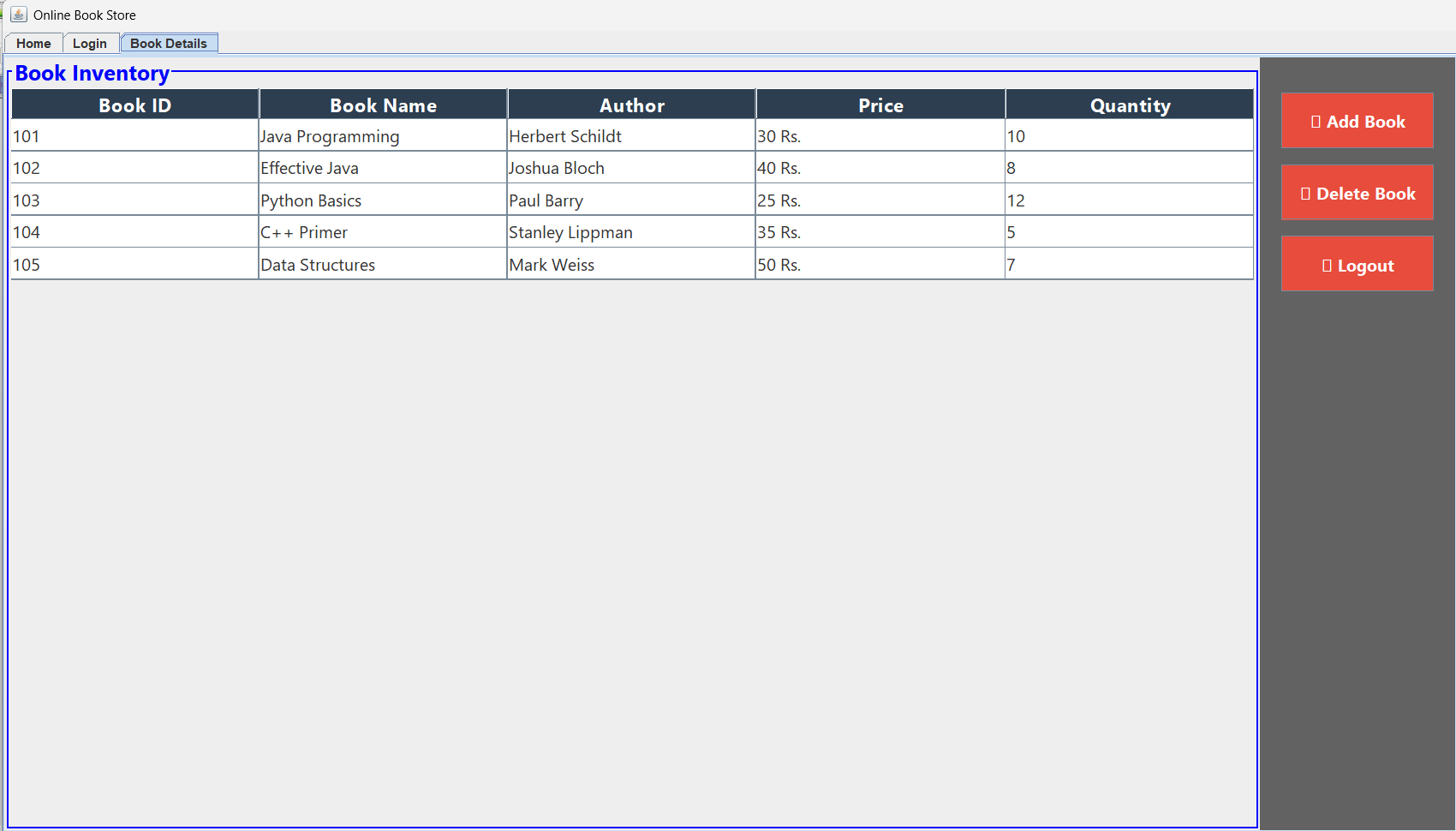
1. Java Swing for GUI Development – Used JFrame, JPanel, JLabel, JButton, JTable, JTabbedPane, etc., to create a user-friendly interface.
2. Event Handling – Implemented action listeners (ActionListener) to handle button clicks for login, book addition, issuing, returning, and logout functionality.
3. JTabbedPane for Navigation – Used JTabbedPane to organize different sections like Home, Books Available, and Functional Panels for easy navigation.
4. JTable for Data Display – Displayed books in a table format using JTable and DefaultTableModel to show book details dynamically.
5. User Authentication – Implemented a login system with JTextField and JPasswordField to restrict access to authorized users.
6. GridLayout and BorderLayout – Used layout managers (GridLayout, BorderLayout, null layout) to arrange components effectively.
7. JOptionPane for Dialogs – Displayed confirmation messages, error alerts, and logout prompts using JOptionPane.showMessageDialog() and JOptionPane.showConfirmDialog().
8. Encapsulation & Object-Oriented Programming (OOP) – Organized the code into classes (LoginFrame, Frame\_EX2, StudentRegistrationFrame, etc.), ensuring modularity and readability.
9. Dynamic UI Updates – Enabled real-time interaction, such as adding books, issuing, and returning them, by modifying table data and updating UI elements dynamically.
10. Logout & Session Handling – Implemented a logout feature that prompts the user and closes the session securely using SwingUtilities.getWindowAncestor().dispose().

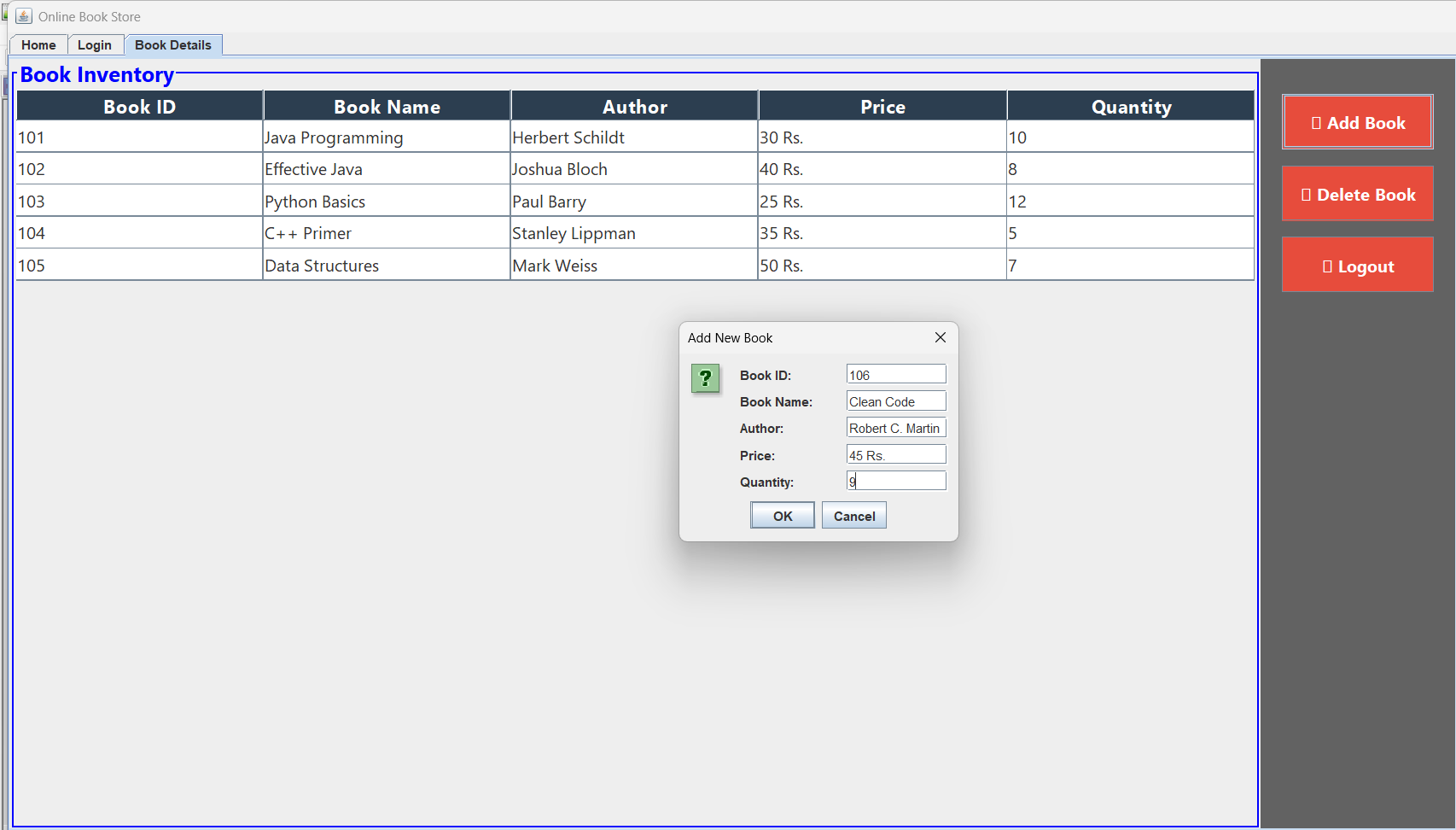
**4.3 Programs or code of relevant concepts and output:**

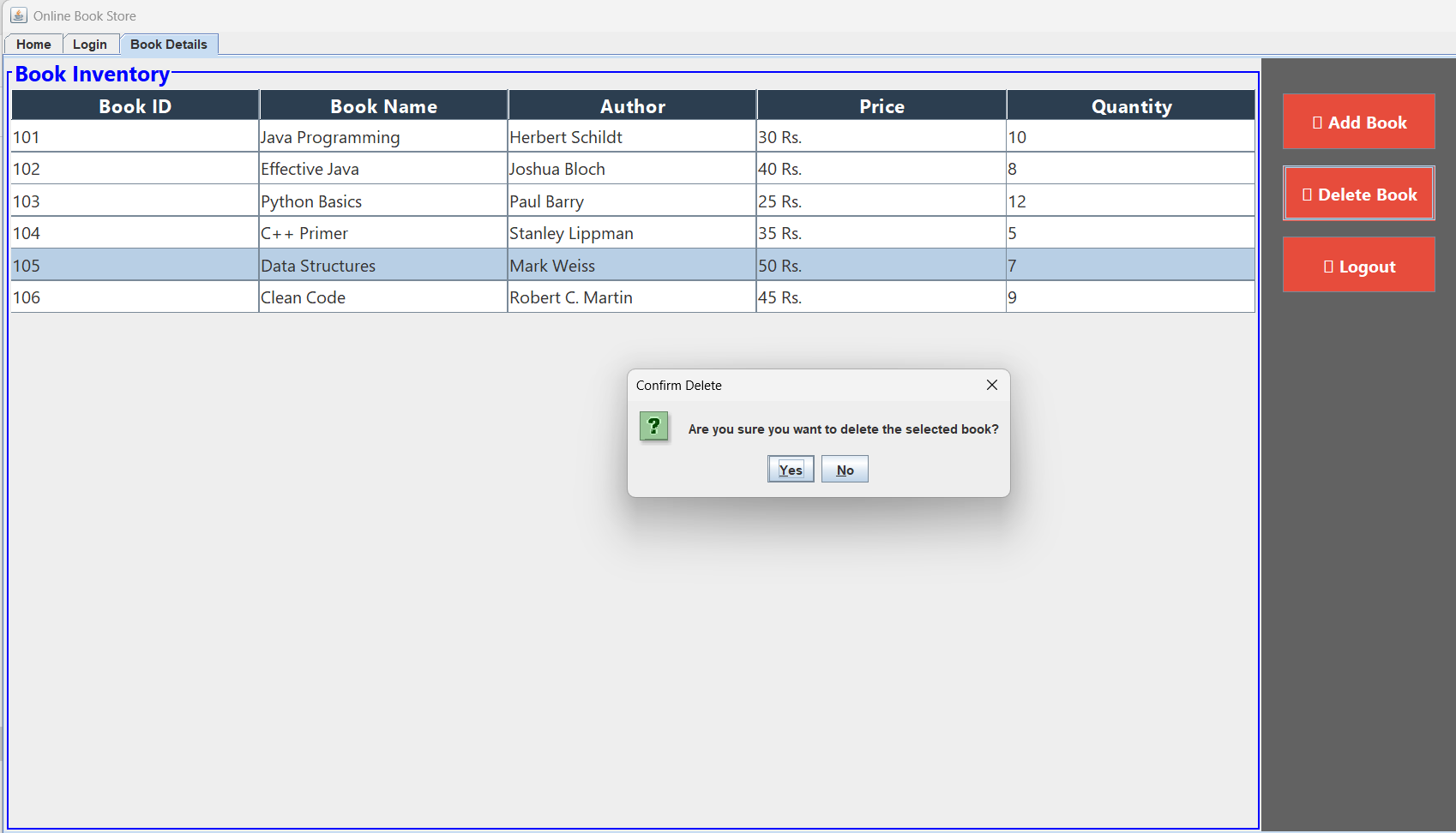
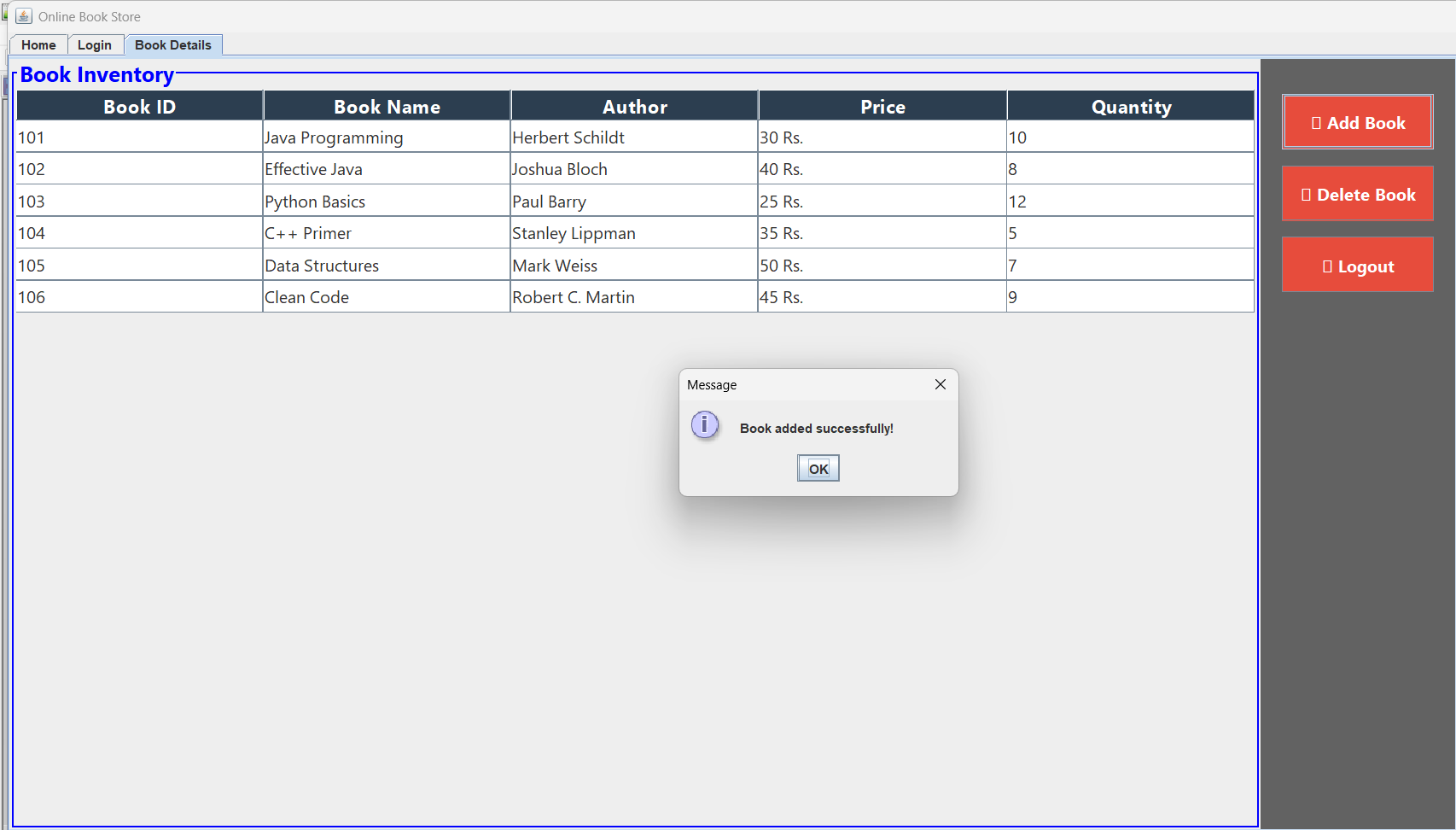
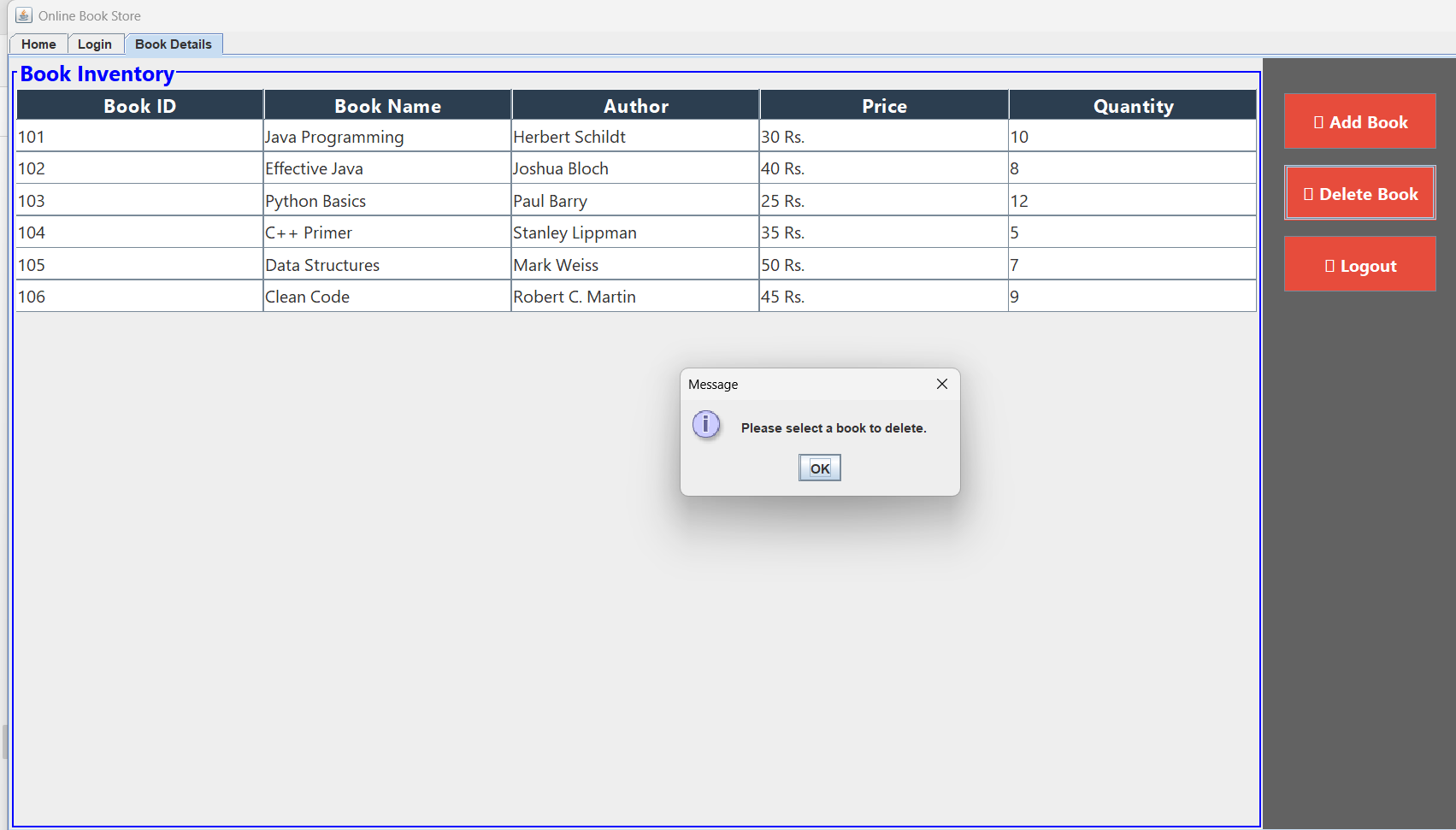
|  |
| --- |
| import javax.swing.\*;  import javax.swing.table.DefaultTableModel;  import java.awt.\*;  import java.awt.event.\*;  public class Bookshop {  static JTabbedPane tabbedPane;  static int booksTabIndex = -1;  static boolean isAdminLoggedIn = false;  static DefaultTableModel bookTableModel = new DefaultTableModel(  new Object[][]{  {"101", "Java Programming", "Herbert Schildt", "30 Rs.", "10"},  {"102", "Effective Java", "Joshua Bloch", "40 Rs.", "8"},  {"103", "Python Basics", "Paul Barry", "25 Rs.", "12"},  {"104", "C++ Primer", "Stanley Lippman", "35 Rs.", "5"},  {"105", "Data Structures", "Mark Weiss", "50 Rs.", "7"}  },  new Object[]{"Book ID", "Book Name", "Author", "Price", "Quantity"}  );  public static void main(String[] args) {  SwingUtilities.invokeLater(() -> new Bookshop().initialize());  }  public void initialize() {  JFrame frame = new JFrame("Online Book Store");  frame.setSize(1500, 800);  frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);  frame.setLayout(null);  tabbedPane = new JTabbedPane();  tabbedPane.setBounds(0, 0, 1360, 750);  frame.add(tabbedPane);  JPanel homePanel = createImagePanel("b2.jpg");  homePanel.add(new JLabel("Home Tab Content"));  JLabel hlabel = new JLabel("Welcome To Online Bookshop");  hlabel.setFont(new Font("Arial", Font.PLAIN, 40));  hlabel.setForeground(Color.WHITE);  hlabel.setBounds(380, 160, 1000, 150);  homePanel.add(hlabel);  JLabel lb = new JLabel("THE BOOK NOOK");  lb.setFont(new Font("Arial", Font.PLAIN, 55));  lb.setForeground(Color.WHITE);  lb.setBounds(420, 250, 1000, 150);  homePanel.add(lb);  JLabel hl = new JLabel("Email: sswp1226@gmail.com");  hl.setFont(new Font("Arial", Font.PLAIN, 25));  hl.setForeground(Color.WHITE);  hl.setBounds(950, 550, 1000, 150);  homePanel.add(hl);  JLabel Jl = new JLabel("Contact: 9175720879");  Jl.setFont(new Font("Arial", Font.PLAIN, 25));  Jl.setForeground(Color.WHITE);  Jl.setBounds(950, 600, 1000, 150);  homePanel.add(Jl);  JButton b = new JButton("LOGIN");  b.setFont(new Font("Arial", Font.PLAIN, 20));  b.setForeground(Color.WHITE);  b.setBounds(550, 400, 200, 80);  b.setContentAreaFilled(false);  homePanel.add(b);  b.addActionListener(e -> {  if (getTabIndexByTitle("Login") == -1) {  tabbedPane.addTab("Login", new LoginPanel());  }  tabbedPane.setSelectedIndex(getTabIndexByTitle("Login"));  });  tabbedPane.addTab("Home", homePanel);  frame.setVisible(true);  }  private int getTabIndexByTitle(String title) {  for (int i = 0; i < tabbedPane.getTabCount(); i++) {  if (tabbedPane.getTitleAt(i).equalsIgnoreCase(title)) {  return i;  }  }  return -1;  }  public JPanel createBooksPanel() {  JPanel panel = new JPanel(new BorderLayout());  JLabel background = new JLabel(new ImageIcon("b2.jpg"));  background.setLayout(new BorderLayout());  panel.add(background);  JTable booksTable = new JTable(Bookshop.bookTableModel);  booksTable.setRowHeight(30);  booksTable.setFont(new Font("Segoe UI", Font.PLAIN, 16));  booksTable.getTableHeader().setFont(new Font("Segoe UI", Font.BOLD, 18));  booksTable.getTableHeader().setBackground(new Color(44, 62, 80));  booksTable.getTableHeader().setForeground(Color.WHITE);  JScrollPane scrollPane = new JScrollPane(booksTable);  scrollPane.setBorder(BorderFactory.createTitledBorder(  BorderFactory.createLineBorder(Color.BLUE, 2),  "Book Inventory",  0, 0,  new Font("Segoe UI", Font.BOLD, 20),  Color.BLUE  ));  background.add(scrollPane, BorderLayout.CENTER);  JPanel sidePanel = new JPanel(new GridLayout(10, 1, 15, 15));  sidePanel.setBackground(new Color(0, 0, 0, 150));  sidePanel.setBorder(BorderFactory.createEmptyBorder(30, 20, 30, 20));  Font buttonFont = new Font("Segoe UI", Font.BOLD, 16);  if (isAdminLoggedIn) {  JButton addBookBtn = new JButton("➕ Add Book");  styleSideButton(addBookBtn, buttonFont);  addBookBtn.addActionListener(e -> {  JTextField id = new JTextField();  JTextField name = new JTextField();  JTextField author = new JTextField();  JTextField price = new JTextField();  JTextField qty = new JTextField();  JPanel inputPanel = new JPanel(new GridLayout(5, 2, 5, 5));  inputPanel.add(new JLabel("Book ID:")); inputPanel.add(id);  inputPanel.add(new JLabel("Book Name:")); inputPanel.add(name);  inputPanel.add(new JLabel("Author:")); inputPanel.add(author);  inputPanel.add(new JLabel("Price:")); inputPanel.add(price);  inputPanel.add(new JLabel("Quantity:")); inputPanel.add(qty);  int result = JOptionPane.showConfirmDialog(null, inputPanel, "Add New Book", JOptionPane.OK\_CANCEL\_OPTION);  if (result == JOptionPane.OK\_OPTION) {  if (id.getText().isEmpty() || name.getText().isEmpty() || author.getText().isEmpty() ||  price.getText().isEmpty() || qty.getText().isEmpty()) {  JOptionPane.showMessageDialog(null, "All fields are required!");  } else {  bookTableModel.addRow(new Object[]{id.getText(), name.getText(), author.getText(), price.getText(), qty.getText()});  JOptionPane.showMessageDialog(null, "Book added successfully!");  }  }  });  sidePanel.add(addBookBtn);  JButton deleteBookBtn = new JButton("🗑️ Delete Book");  styleSideButton(deleteBookBtn, buttonFont);  deleteBookBtn.addActionListener(e -> {  int selectedRow = booksTable.getSelectedRow();  if (selectedRow >= 0) {  int confirm = JOptionPane.showConfirmDialog(null, "Are you sure you want to delete the selected book?", "Confirm Delete", JOptionPane.YES\_NO\_OPTION);  if (confirm == JOptionPane.YES\_OPTION) {  bookTableModel.removeRow(selectedRow);  JOptionPane.showMessageDialog(null, "Book deleted successfully!");  }  } else {  JOptionPane.showMessageDialog(null, "Please select a book to delete.");  }  });  sidePanel.add(deleteBookBtn);  } else {  JButton studentRegBtn = new JButton("👤 Student Registration");  styleSideButton(studentRegBtn, buttonFont);  studentRegBtn.addActionListener(e -> new StudentRegistrationFrame());  sidePanel.add(studentRegBtn);  JButton issueBookBtn = new JButton("📚 Issue Books");  styleSideButton(issueBookBtn, buttonFont);  issueBookBtn.addActionListener(e -> new IssueBookFrame());  sidePanel.add(issueBookBtn);  }  JButton logoutButton = new JButton("🚪 Logout");  styleSideButton(logoutButton, buttonFont);  logoutButton.addActionListener(e -> {  int confirm = JOptionPane.showConfirmDialog(null, "Are you sure you want to logout?", "Logout", JOptionPane.YES\_NO\_OPTION);  if (confirm == JOptionPane.YES\_OPTION) {  JOptionPane.showMessageDialog(null, "Logged out successfully!");  ((JFrame) SwingUtilities.getWindowAncestor(tabbedPane)).dispose();  new Bookshop().initialize();  }  });  sidePanel.add(logoutButton);  background.add(sidePanel, BorderLayout.EAST);  return panel;  }  private void styleSideButton(JButton button, Font font) {  button.setFont(font);  button.setForeground(Color.WHITE);  button.setBackground(new Color(231, 76, 60));  button.setFocusPainted(false);  }  private JPanel createImagePanel(String imagePath) {  JPanel panel = new JPanel() {  @Override  protected void paintComponent(Graphics g) {  super.paintComponent(g);  ImageIcon imageIcon = new ImageIcon(imagePath);  g.drawImage(imageIcon.getImage(), 0, 0, getWidth(), getHeight(), null);  }  };  panel.setLayout(null);  return panel;  }  static class LoginPanel extends JPanel {  public LoginPanel() {  setLayout(new BorderLayout());  setBackground(new Color(44, 62, 80));  JTabbedPane loginTabs = new JTabbedPane();  loginTabs.setFont(new Font("Segoe UI", Font.BOLD, 16));  JPanel adminPanel = createLoginPanel("Admin Login", "admin", "admin123", true);  JPanel studentPanel = createLoginPanel("Student Login", "", "", false);  loginTabs.addTab("Admin Login", adminPanel);  loginTabs.addTab("Student Login", studentPanel);  add(loginTabs, BorderLayout.CENTER);  }  private JPanel createLoginPanel(String title, String validUser, String validPass, boolean isAdmin) { JPanel panel = new JPanel(new GridBagLayout());  panel.setBackground(new Color(52, 73, 94));  GridBagConstraints gbc = new GridBagConstraints();  gbc.insets = new Insets(10, 10, 10, 10);  gbc.fill = GridBagConstraints.HORIZONTAL;  JLabel loginTitle = new JLabel(title);  loginTitle.setFont(new Font("Segoe UI", Font.BOLD, 28));  loginTitle.setForeground(Color.WHITE);  gbc.gridx = 0;  gbc.gridy = 0;  gbc.gridwidth = 2;  panel.add(loginTitle, gbc);  JTextField userField = new JTextField();  JPasswordField passField = new JPasswordField();  styleTextField(userField);  styleTextField(passField);  gbc.gridwidth = 1;  gbc.gridy++;  panel.add(new JLabel(isAdmin ? "Username:" : "Student ID:"), gbc);  gbc.gridx = 1;  panel.add(userField, gbc);  gbc.gridx = 0;  gbc.gridy++;  panel.add(new JLabel("Password:"), gbc);  gbc.gridx = 1;  panel.add(passField, gbc);  JButton loginBtn = new JButton("Login");  styleButton(loginBtn);  gbc.gridx = 0;  gbc.gridy++;  gbc.gridwidth = 2;  panel.add(loginBtn, gbc);  loginBtn.addActionListener(e -> {  String user = userField.getText().trim();  String pass = new String(passField.getPassword()).trim();  if (isAdmin) {  if (user.equals(validUser) && pass.equals(validPass)) {  Bookshop.isAdminLoggedIn = true;  } else {  JOptionPane.showMessageDialog(this, "Invalid Admin Credentials!", "Error", JOptionPane.ERROR\_MESSAGE);  return;  }  } else {  if (!user.isEmpty() && !pass.isEmpty()) {  Bookshop.isAdminLoggedIn = false;  } else {  JOptionPane.showMessageDialog(this, "Student ID and Password cannot be empty!", "Error", JOptionPane.ERROR\_MESSAGE);  return;  }  }  JOptionPane.showMessageDialog(this, title + " Successful!");  if (Bookshop.booksTabIndex == -1) {  JPanel booksPanel = new Bookshop().createBooksPanel();  Bookshop.tabbedPane.addTab("Book Details", booksPanel);  Bookshop.booksTabIndex = Bookshop.tabbedPane.getTabCount() - 1;  }  Bookshop.tabbedPane.setSelectedIndex(Bookshop.booksTabIndex);  });  return panel;  }  private void styleTextField(JTextField field) {  field.setFont(new Font("Segoe UI", Font.PLAIN, 16));  field.setBorder(BorderFactory.createCompoundBorder(  BorderFactory.createLineBorder(Color.GRAY, 1),  BorderFactory.createEmptyBorder(5, 10, 5, 10)  ));  field.setBackground(new Color(236, 240, 241));  }  private void styleButton(JButton button) {  button.setFont(new Font("Segoe UI", Font.BOLD, 18));  button.setBackground(new Color(231, 76, 60));  button.setForeground(Color.WHITE);  button.setFocusPainted(false);  }  } static class StudentRegistrationFrame extends JFrame {  public StudentRegistrationFrame() {  setTitle("Student Registration");  setSize(400, 400);  setLocationRelativeTo(null);  setLayout(new GridLayout(6, 2, 10, 10));  JLabel idLabel = new JLabel("Student ID:");  JTextField idField = new JTextField();  JLabel nameLabel = new JLabel("Student Name:");  JTextField nameField = new JTextField();  JLabel courseLabel = new JLabel("Course:");  JTextField courseField = new JTextField();  JLabel branchLabel = new JLabel("Branch:");  JTextField branchField = new JTextField();  JButton registerButton = new JButton("Register");  add(idLabel); add(idField);  add(nameLabel); add(nameField);  add(courseLabel); add(courseField);  add(branchLabel); add(branchField);  add(new JLabel()); add(registerButton);  registerButton.addActionListener(e -> {  if (!idField.getText().trim().isEmpty() &&  !nameField.getText().trim().isEmpty() &&  !courseField.getText().trim().isEmpty() &&  !branchField.getText().trim().isEmpty()) {  JOptionPane.showMessageDialog(this, "Student Registered Successfully!");  dispose();  } else { JOptionPane.showMessageDialog(this, "Please fill all fields!", "Error", JOptionPane.ERROR\_MESSAGE);  }  });  setVisible(true);  } }  static class IssueBookFrame extends JFrame {  public IssueBookFrame() {  setTitle("Issue Book");  setSize(400, 400);  setLocationRelativeTo(null);  setLayout(new GridLayout(6, 2, 10, 10));  JLabel studentIdLabel = new JLabel("Student ID:");  JTextField studentIdField = new JTextField();  JLabel bookIdLabel = new JLabel("Book ID:");  JTextField bookIdField = new JTextField();  JLabel issueDateLabel = new JLabel("Issue Date:");  JTextField issueDateField = new JTextField();  JButton issueButton = new JButton("Issue");  add(studentIdLabel); add(studentIdField);  add(bookIdLabel); add(bookIdField);  add(issueDateLabel); add(issueDateField);  add(new JLabel()); add(issueButton);  issueButton.addActionListener(e -> {  String studentId = studentIdField.getText().trim();  String bookId = bookIdField.getText().trim();  String issueDate = issueDateField.getText().trim();  if (studentId.isEmpty() || bookId.isEmpty() || issueDate.isEmpty()) {  JOptionPane.showMessageDialog(this, "Please fill all fields!", "Error", JOptionPane.ERROR\_MESSAGE);  return;  }  DefaultTableModel model = Bookshop.bookTableModel;  boolean bookFound = false;  for (int i = 0; i < model.getRowCount(); i++) {  if (model.getValueAt(i, 0).toString().equals(bookId)) {  int qty = Integer.parseInt(model.getValueAt(i, 4).toString());  if (qty > 0) { model.setValueAt(String.valueOf(qty - 1), i, 4);  bookFound = true;  JOptionPane.showMessageDialog(this, "Book Issued Successfully to Student ID: " + studentId);  dispose();  } else {  JOptionPane.showMessageDialog(this, "Book is out of stock.", "Unavailable", JOptionPane.WARNING\_MESSAGE);  return;  } break;  }  } if (!bookFound) {  JOptionPane.showMessageDialog(this, "Book ID not found!", "Error", JOptionPane.ERROR\_MESSAGE);  }  }); setVisible(true);  }  }  } |

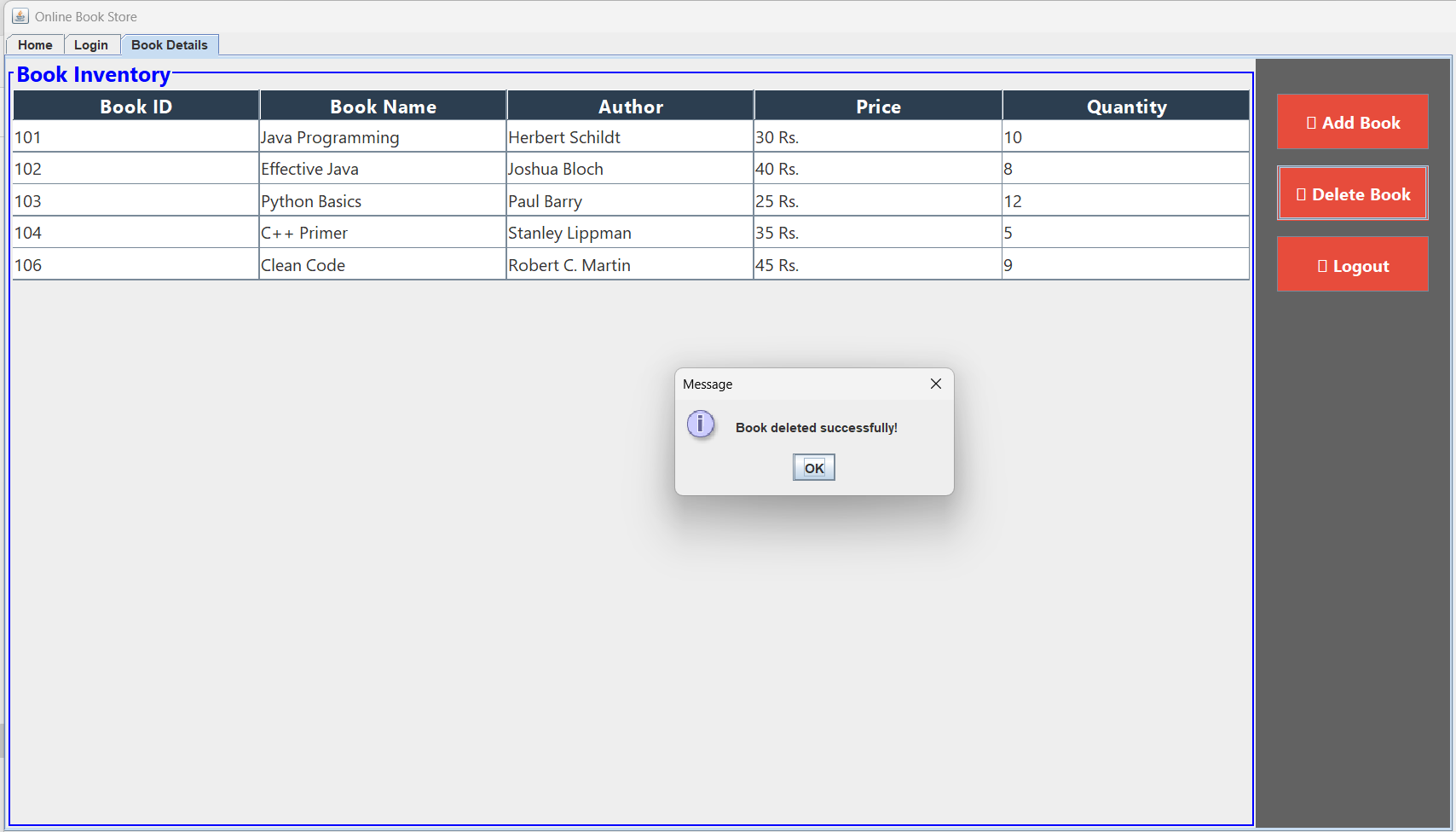
1. **Output:**

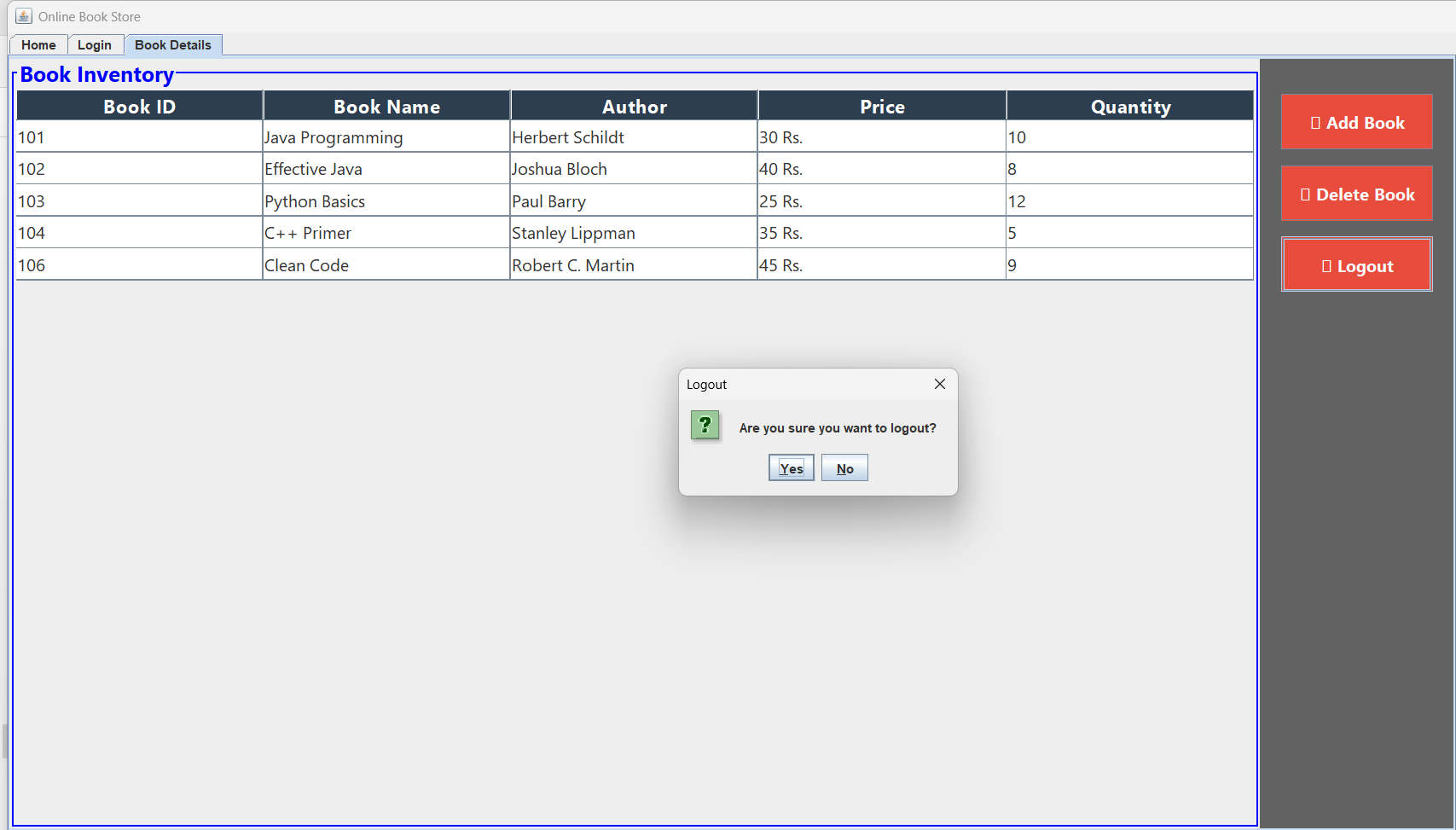
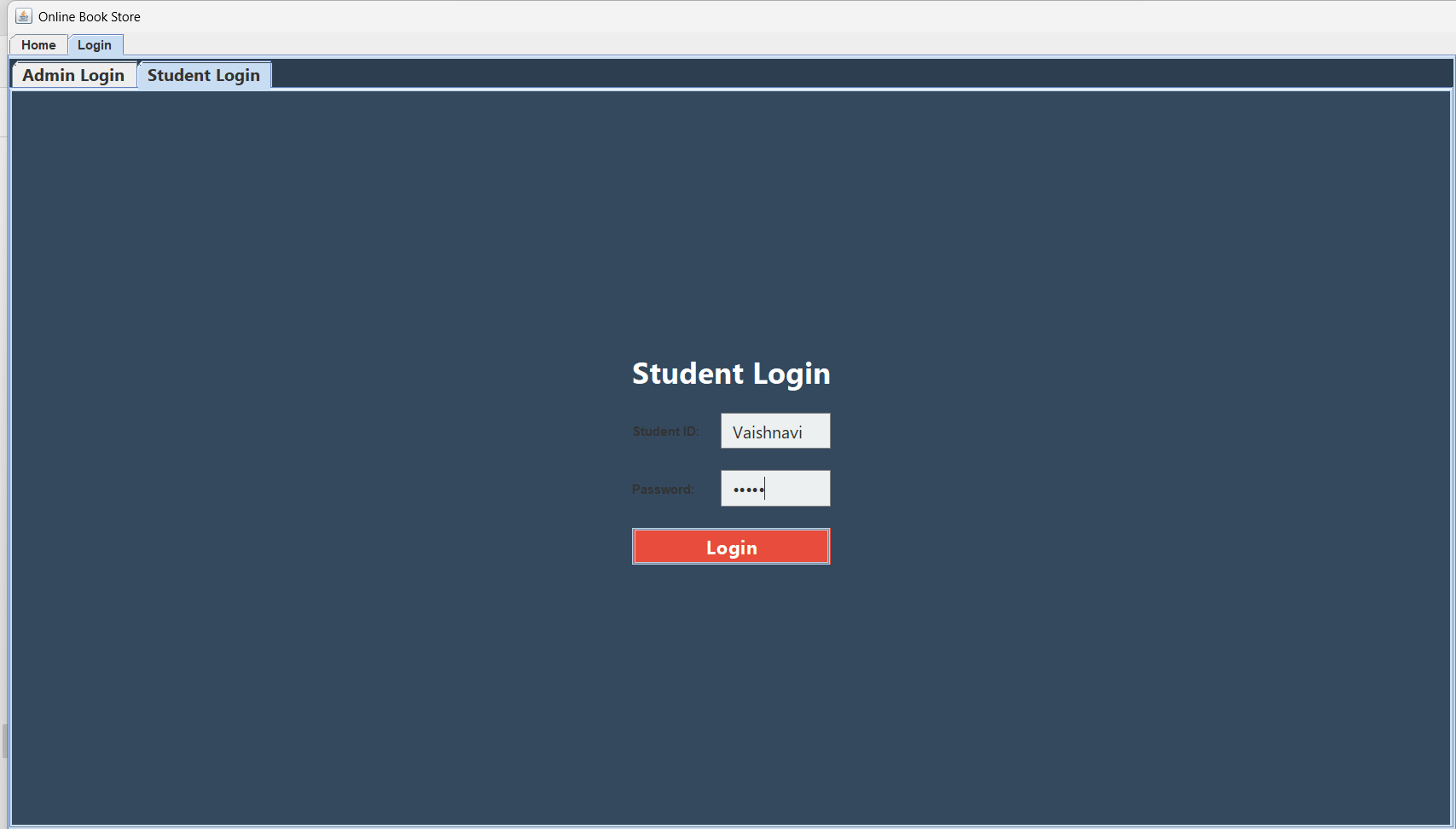
****

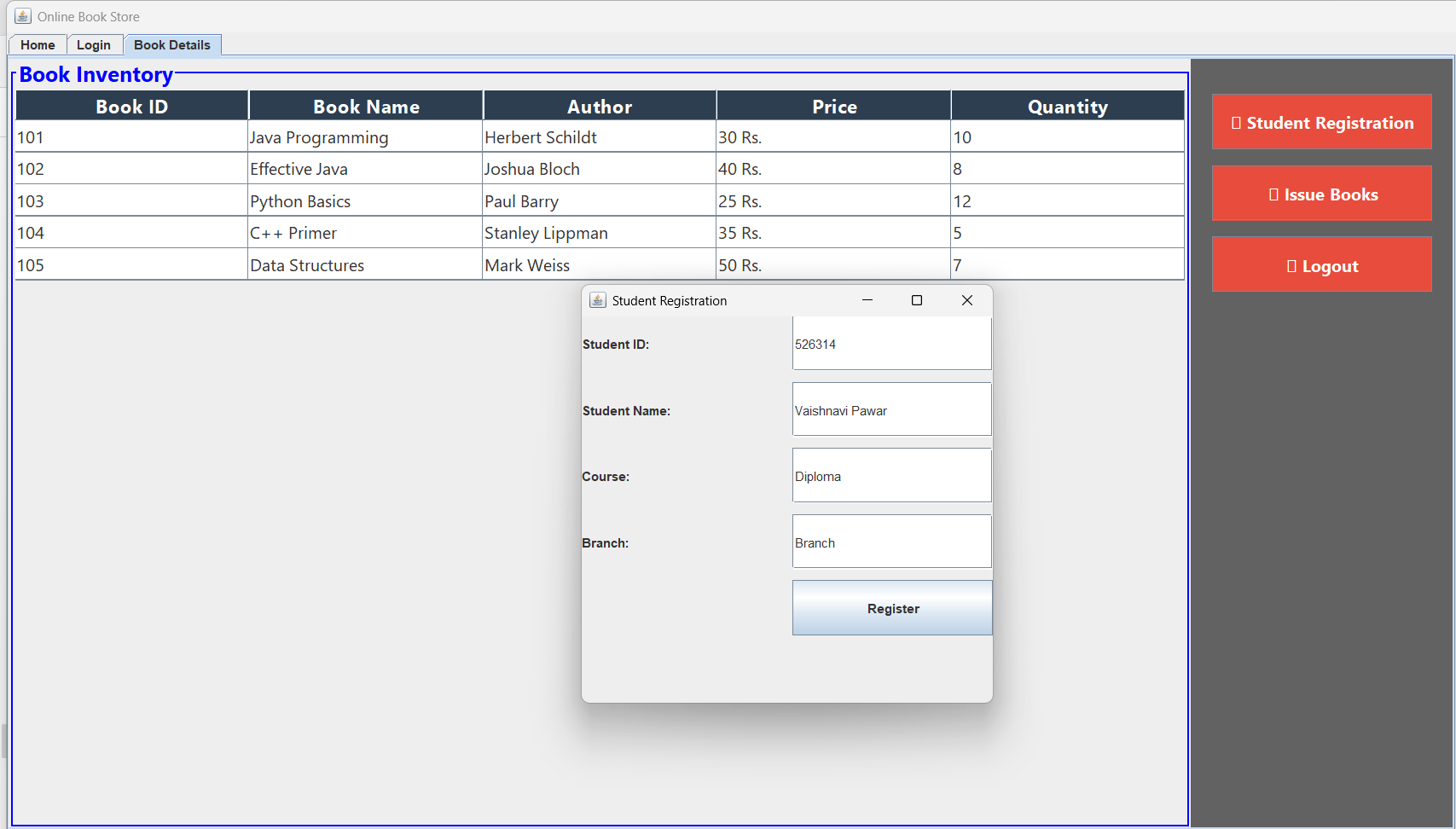


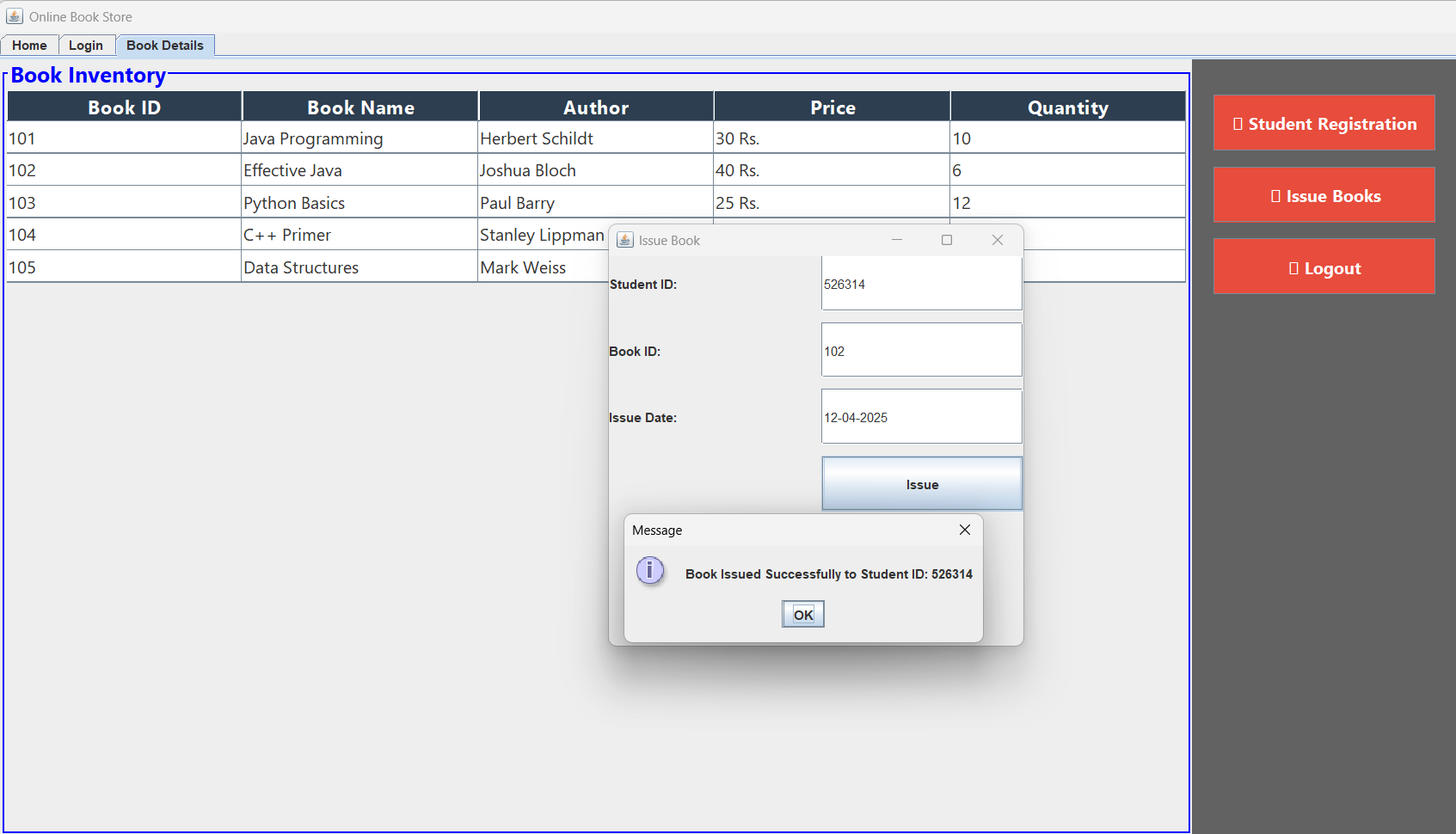
`











1. **Advantages:**

Here are four advantages of the Bookshop Management System:

1. User-Friendly Interface – Built with Java Swing, featuring an intuitive GUI with easy navigation using JTabbedPane.
2. Efficient Book Management – Users can add, issue, and return books easily using JTable, ensuring proper record-keeping.
3. Secure Authentication – A login system ensures that only authorized users can access the bookshop functionalities.
4. Time-Saving & Automated – Automates book transactions, reducing manual work and making operations faster.
5. Structured & Scalable Design – Uses Object-Oriented Programming (OOP) principles for better maintainability and future expansion.
6. Session Management & Logout – Provides a secure logout option, ensuring controlled user access and data safety.
7. Interactive & Informative – Uses JOptionPane for alerts, confirmations, and error handling, improving user interaction.
8. **Disadvantages:**

Here are four disadvantages of the Bookshop Management System:

* 1. No Data Persistence – Currently, the system does not use a database or file storage, meaning all data is lost when the program is closed.
  2. Limited User Roles – There is no distinction between admin and regular users, reducing control over system access and permissions.
  3. Basic Security Measures – The login system is hardcoded with credentials, making it insecure and vulnerable to unauthorized access.
  4. Static Book Data – Books are displayed using a JTable with predefined values, limiting the ability to dynamically manage inventory.
  5. No Search & Filtering – Users cannot search or filter books, making it harder to find specific titles quickly.

1. **Applications:**

Real-Life Applications of the Online Bookshop Management System:

1. Libraries & Educational Institutions – Can be used in school, college, and university libraries to manage book borrowing, returns, and student registrations.
2. Online & Physical Bookstores – Helps bookshops and retail stores keep track of book inventory, sales, and customer transactions.
3. E-Libraries & Digital Book Management – Can be extended to manage digital books (PDFs, eBooks) for online reading platforms.
4. Corporate & Office Libraries – Companies with internal libraries can use this system for managing employee book borrowing and returns.
5. Public Libraries – Government or community public libraries can implement this system for automating book records and reducing manual work.

1. **Conclusion:**

The Online Bookshop Management System provides an efficient and user-friendly platform for managing books, issuing, returning, and user authentication. Built using Java Swing, it offers basic book management functionalities but lacks data persistence and advanced search features. With future enhancements like database integration, role-based access, and UI improvements, this system can be transformed into a fully automated and scalable solution for libraries and bookstores.

1. **Reference:**
2. <https://www.geeksforgeeks.org/bookshop-management-system-using-file-handling/>
3. <https://www.scribd.com/doc/124378514/Book-Shop-Management-System-Documentation>
4. <https://sites.google.com/site/ignoubcafinalyearprojects/project-report/book-shop-management-system-project-report>