Online Bookstore Project Report

Team Members:

• Akhil Raj Vudukulla

• Sri Varsha Pasham

• Charitha Sri Shavva

• Pavani Krishna Priya Chava

Course: Advanced Application Programming in Java (CS-5220)

Semester: Spring 2025

Instructor: Anand Bapatla

University: University of Central Missouri

# 1. Introduction

As a group of four students enrolled in the Advanced Application Programming in Java course (CS-5220), we collaboratively developed an Online Bookstore web application. This project allowed us to apply concepts such as Servlets, JDBC, and MVC architecture in a practical, real-world scenario. This report outlines the design, development, and implementation details of the project.

# 2. Project Overview

The Online Bookstore is a Java-based web application that enables users to browse, purchase, and manage books online. The application supports both admin and customer roles. Customers can register, login, add books to the cart, and proceed to checkout, while admins have privileges to manage the book inventory.

# 3. Project Structure

We organized the project into logical packages and folders. The source files are located in 'src/main/java'. Other directories include configuration files, SQL scripts, and deployment utilities.  
  
• '.idea/' and '.settings/' – IDE configuration files  
• 'src/main/java/' – Java source code including models, services, servlets  
• 'setup/' – SQL script for database schema creation  
• 'scripts/' – Shell scripts to manage server execution  
• 'pom.xml' – Maven build and dependency file  
• 'Procfile', 'appspec.yaml' – Deployment configuration (AWS supported)

# 4. Key Java Components

## 4.1 Models

The model classes are used to represent core entities like books and users:  
• Book.java – Represents book details  
• User.java – Holds user credentials and profile information  
• Cart.java – Used for storing selected books for checkout  
• UserRole.java – Distinguishes between admin and customer roles

## 4.2 Services and Implementations

The service layer separates business logic:  
• BookService.java – Interface for managing books  
• UserService.java – Interface for user management  
These interfaces are implemented in the 'impl' subpackage using JDBC.

## 4.3 Utilities

To simplify development, we used utility classes:  
• DBUtil.java – Manages database connections  
• StoreUtil.java – Contains helper methods  
• DatabaseConfig.java – Stores database credentials

## 4.4 Servlets

We implemented multiple servlets to handle user and admin requests:  
• CustomerLoginServlet.java – Manages login  
• CustomerRegisterServlet.java – Handles user registration  
• AddBookServlet.java – Enables admin book addition  
• CartServlet.java – Manages shopping cart  
• CheckoutServlet.java – Processes checkout  
• ErrorHandlerServlet.java – Handles exceptions globally

# 5. Database Integration

The database integration was done using JDBC. The database schema includes tables for books, users, and cart items. The connection parameters are defined in DatabaseConfig.java. The script in 'setup/CreateDatastore.sql' is used to set up the initial database.

# 6. Build and Deployment

We used Maven as the build tool to manage dependencies and package the application into a WAR file. It can be deployed on Apache Tomcat by placing the WAR in the webapps directory. Shell scripts are available to start and stop the server.

# 7. Running the Application

To run the application:  
1. Execute the database script  
2. Configure credentials in DatabaseConfig.java  
3. Build the project with Maven  
4. Deploy the WAR file to Tomcat  
5. Access via http://localhost:8080/onlinebookstore/

# 8. Functional Summary

The project includes the following features:  
  
• User registration and login  
• Book catalog with search functionality  
• Shopping cart and checkout  
• Admin interface to add/update/delete books  
• Error and session management

# 9. Conclusion

This project helped us gain valuable experience in Java EE development, database integration, and full-stack application design. We collaborated effectively, dividing tasks such as database setup, backend development, and servlet logic implementation among the team.