**TITLE: Digital Library System Project**

**Section 1: Project Setup & Environment Configuration**

**1.1 Install Required Tools and Dependencies**

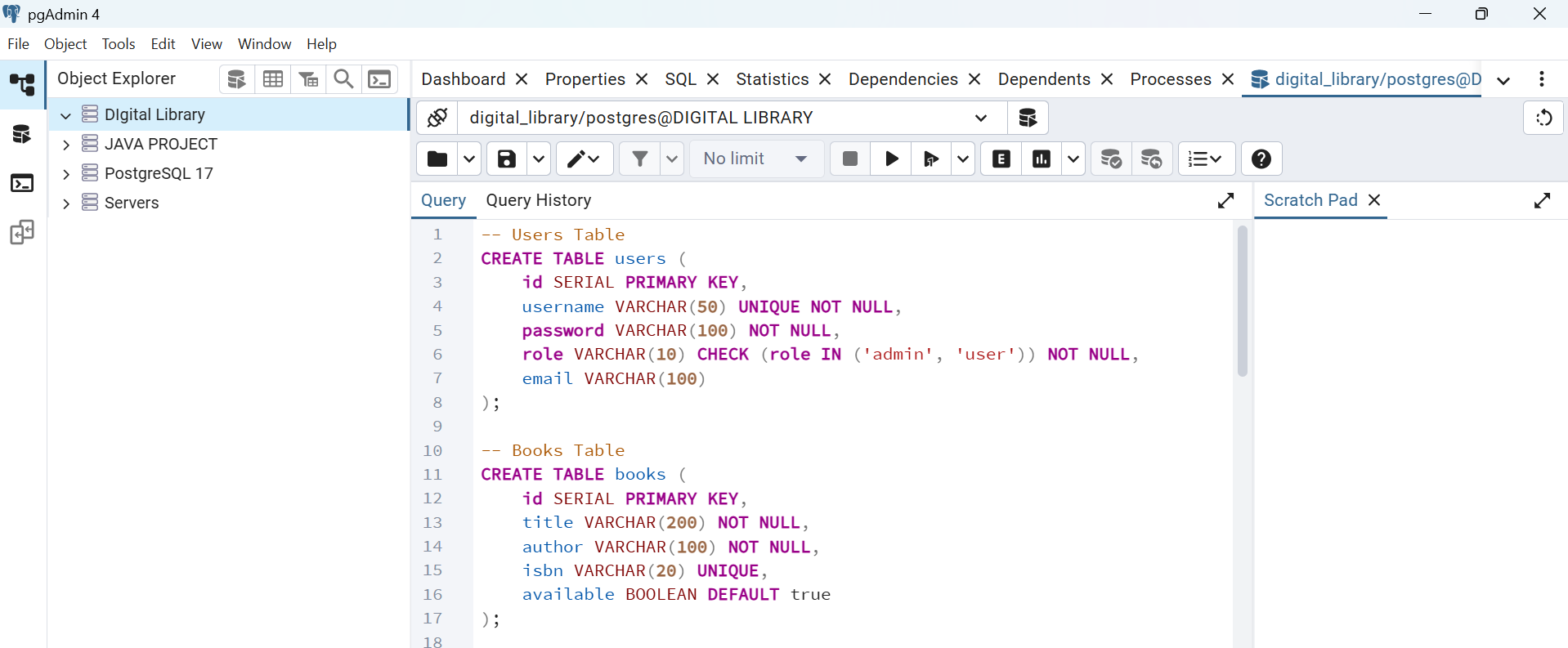
**• NetBeans IDE (with Java & Tomcat)**

**• PostgreSQL + pgAdmin**

**• PostgreSQL JDBC Driver (postgresql-42.7.3.jar)**

* **javax.servlet-api-4.0.1.jar**
* **javax.servlet.jsp-api-2.3.3.jar**
* **jstl-1.2.jar**
* **postgresql-42.7.3.jar**

**1.2 Create Database in pgAdmin**

****

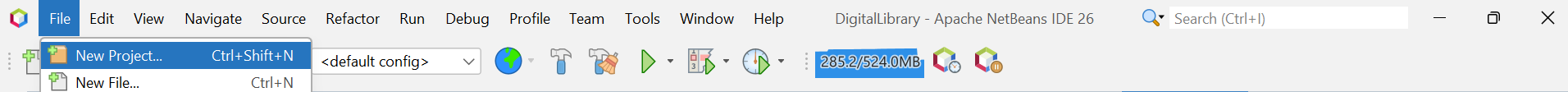
****

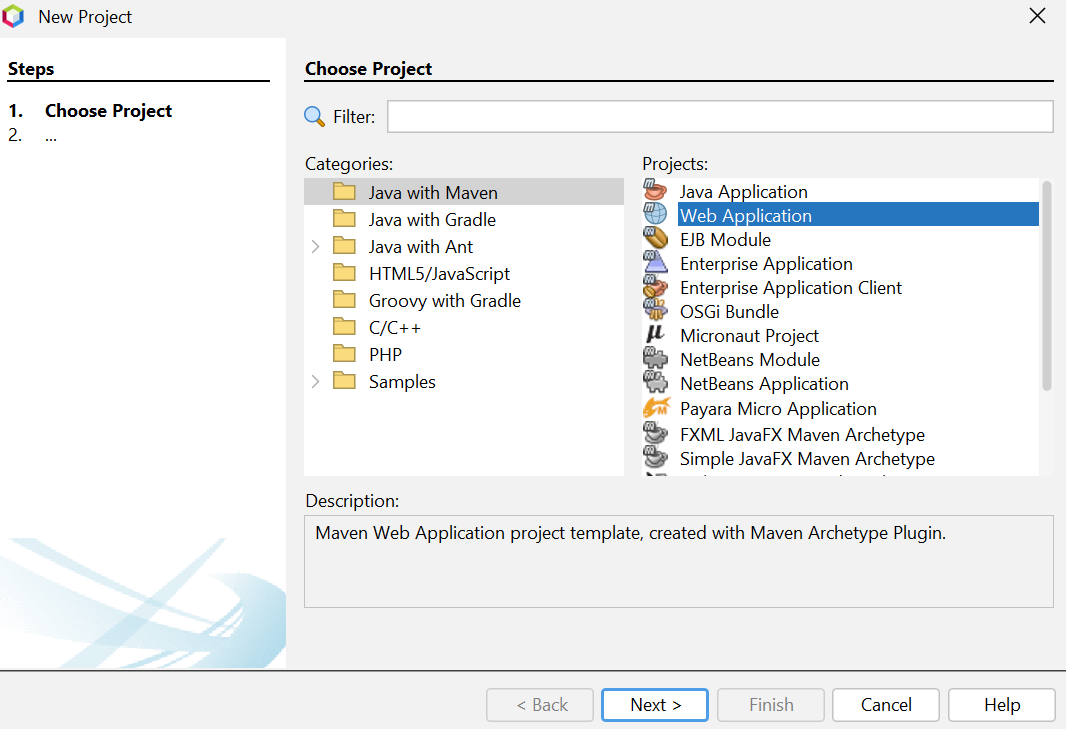
**1.3 NetBeans Project Setup**

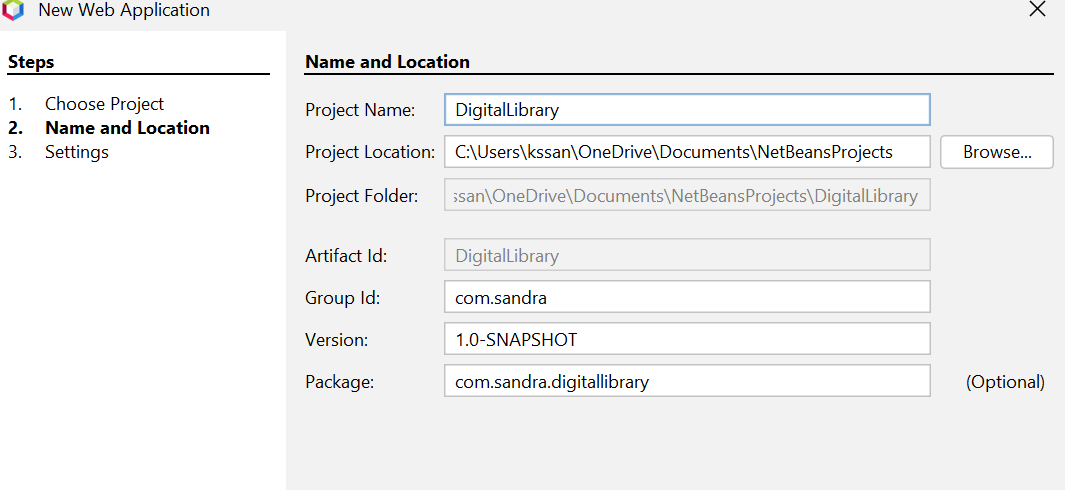
**File → New Project → Java with Maven → Web Application**

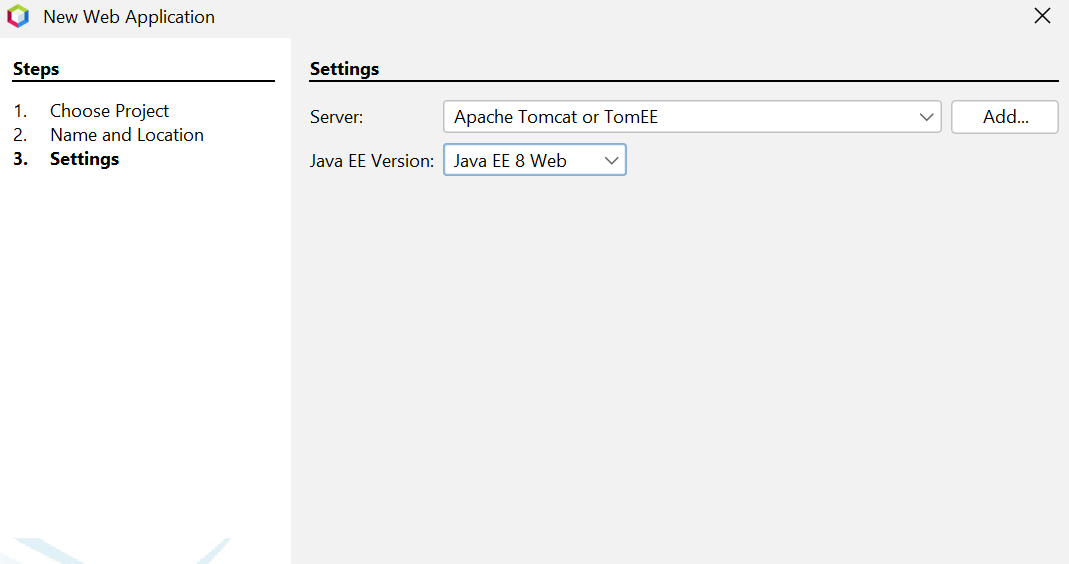
**Project Name: DigitalLibrary**

**Server: Apache Tomcat**

****

****

****

****

**1.4 Configure Maven Dependencies (pom.xml)**

<project xmlns="http://maven.apache.org/POM/4.0.0"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0

http://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>com.sandra</groupId>

<artifactId>DigitalLibrary</artifactId>

<version>1.0-SNAPSHOT</version>

<packaging>war</packaging>

<name>DigitalLibrary</name>

<properties>

<maven.compiler.source>11</maven.compiler.source>

<maven.compiler.target>11</maven.compiler.target>

<project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>

<failOnMissingWebXml>false</failOnMissingWebXml>

</properties>

<dependencies>

<!-- Servlet API (provided by Tomcat) -->

<dependency>

<groupId>javax.servlet</groupId>

<artifactId>javax.servlet-api</artifactId>

<version>4.0.1</version>

<scope>provided</scope>

</dependency>

<!-- JSP API -->

<dependency>

<groupId>javax.servlet.jsp</groupId>

<artifactId>javax.servlet.jsp-api</artifactId>

<version>2.3.3</version>

<scope>provided</scope>

</dependency>

<!-- JSTL -->

<dependency>

<groupId>javax.servlet</groupId>

<artifactId>jstl</artifactId>

<version>1.2</version>

</dependency>

<!-- PostgreSQL JDBC Driver -->

<dependency>

<groupId>org.postgresql</groupId>

<artifactId>postgresql</artifactId>

<version>42.7.3</version>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-compiler-plugin</artifactId>

<version>3.11.0</version>

<configuration>

<source>11</source>

<target>11</target>

</configuration>

</plugin>

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-war-plugin</artifactId>

<version>3.3.2</version>

<configuration>

<failOnMissingWebXml>false</failOnMissingWebXml>

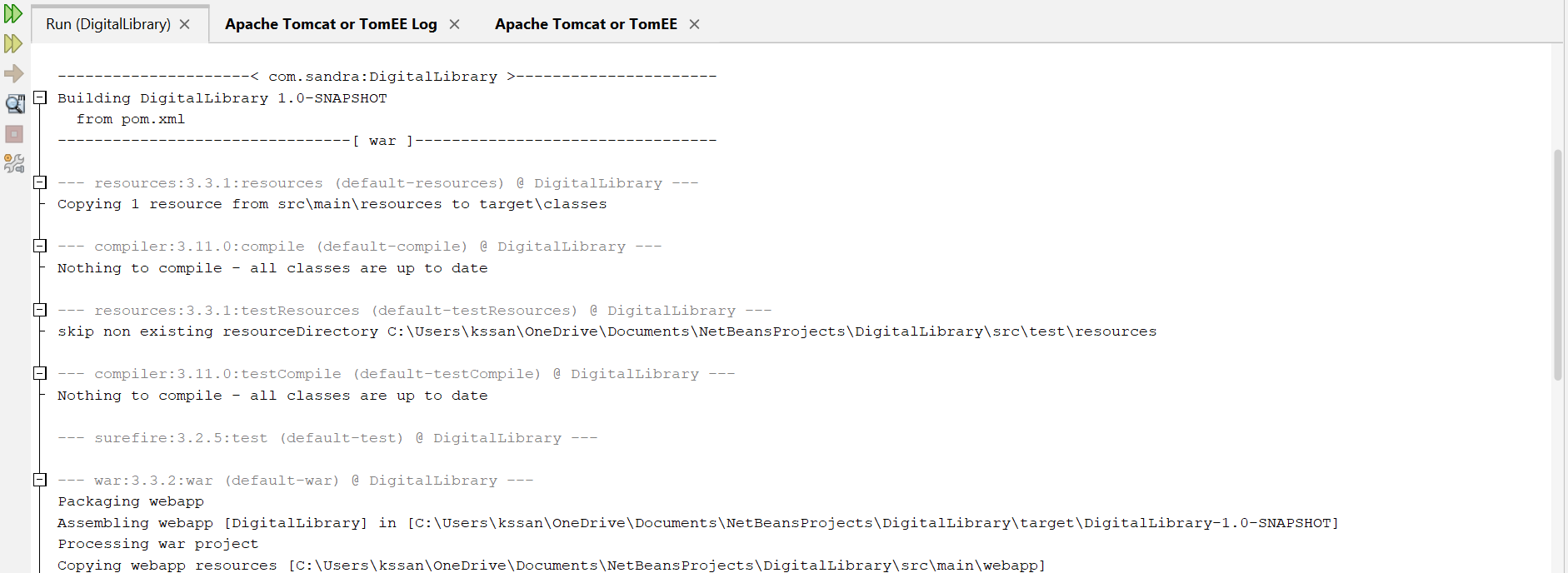
</configuration>

</plugin>

</plugins>

</build>

</project>





**Section 2: Database Connection & Core Classes**

**2.1 Database Connection Class (DatabaseConnection/DatabaseConnection.java)**

/\*

\* Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license

\* Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template

\*/

package DatabaseConnection;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.SQLException;

public class DatabaseConnection {

private static final String URL = "jdbc:postgresql://localhost:5432/digital\_library";

private static final String USER = "\_\_\_\_\_\_"; // your pgAdmin username

private static final String PASSWORD = "\_\_\_\_\_"; // your pgAdmin password

static {

try {

Class.forName("org.postgresql.Driver");

} catch (ClassNotFoundException e) {

e.printStackTrace();

}

}

public static Connection getConnection() throws SQLException {

return DriverManager.getConnection(URL, USER, PASSWORD);

}

}



**2.2 Model Classes**

**2.2.1 User Model (model/User.java)**

package model;

public class User {

private int id;

private String username;

private String password;

private String role;

private String email;

private String lastLogin; // For admin to see when user last logged in

// Default constructor

public User() {}

// Constructor with parameters

public User(String username, String password, String role, String email) {

this.username = username;

this.password = password;

this.role = role;

this.email = email;

}

// Getters and Setters

public int getId() {

return id;

}

public void setId(int id) { // This must be implemented!

this.id = id;

}

public String getUsername() {

return username;

}

public void setUsername(String username) {

this.username = username;

}

public String getPassword() {

return password;

}

public void setPassword(String password) {

this.password = password;

}

public String getRole() {

return role;

}

public void setRole(String role) {

this.role = role;

}

public String getEmail() {

return email;

}

public void setEmail(String email) {

this.email = email;

}

public String getLastLogin() {

return lastLogin;

}

public void setLastLogin(String lastLogin) {

this.lastLogin = lastLogin;

}

}

**2.2.2 Book Model (model/Book.java)**

/\*

\* Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license

\* Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template

\*/

package model;

public class Book {

private int id;

private String title;

private String author;

private String isbn;

private boolean available;

public Book() {}

public Book(String title, String author, String isbn) {

this.title = title;

this.author = author;

this.isbn = isbn;

this.available = true;

}

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public String getTitle() {

return title;

}

public void setTitle(String title) {

this.title = title;

}

public String getAuthor() {

return author;

}

public void setAuthor(String author) {

this.author = author;

}

public String getIsbn() {

return isbn;

}

public void setIsbn(String isbn) {

this.isbn = isbn;

}

public boolean isAvailable() {

return available;

}

public void setAvailable(boolean available) {

this.available = available;

}

}

**Section 3: Data Access Objects (DAO)**

**3.1 User DAO (dao/UserDAO.java)**

package dao;

import DatabaseConnection.DatabaseConnection; // Fixed import path

import model.User;

import java.sql.\*;

import java.util.ArrayList;

import java.util.List;

public class UserDAO {

// Get all users (for admin)

public List<User> getAllUsers() {

List<User> users = new ArrayList<>();

String sql = "SELECT \* FROM users ORDER BY id";

try (Connection conn = DatabaseConnection.getConnection();

Statement stmt = conn.createStatement();

ResultSet rs = stmt.executeQuery(sql)) {

while (rs.next()) {

User user = new User();

user.setId(rs.getInt("id"));

user.setUsername(rs.getString("username"));

user.setPassword(rs.getString("password"));

user.setRole(rs.getString("role"));

user.setEmail(rs.getString("email"));

Timestamp lastLogin = rs.getTimestamp("last\_login");

user.setLastLogin(lastLogin != null ? lastLogin.toString() : "Never");

users.add(user);

}

} catch (SQLException e) {

e.printStackTrace(); // Added error logging

}

return users;

}

public User authenticate(String username, String password) {

String sql = "SELECT \* FROM users WHERE username = ? AND password = ?";

try (Connection conn = DatabaseConnection.getConnection();

PreparedStatement stmt = conn.prepareStatement(sql)) {

stmt.setString(1, username);

stmt.setString(2, password);

ResultSet rs = stmt.executeQuery();

if (rs.next()) {

User user = new User();

user.setId(rs.getInt("id"));

user.setUsername(rs.getString("username"));

user.setPassword(rs.getString("password"));

user.setRole(rs.getString("role"));

user.setEmail(rs.getString("email"));

Timestamp lastLogin = rs.getTimestamp("last\_login");

user.setLastLogin(lastLogin != null ? lastLogin.toString() : "Never");

return user;

}

} catch (SQLException e) {

e.printStackTrace();

}

return null; // Login failed

}

// Register new user

public boolean registerUser(String username, String password, String email) {

String sql = "INSERT INTO users (username, password, role, email) VALUES (?, ?, 'user', ?)";

try (Connection conn = DatabaseConnection.getConnection();

PreparedStatement stmt = conn.prepareStatement(sql)) {

stmt.setString(1, username);

stmt.setString(2, password);

stmt.setString(3, email);

int rowsAffected = stmt.executeUpdate();

return rowsAffected > 0;

} catch (SQLException e) {

e.printStackTrace(); // Added error logging

}

return false; // Registration failed

}

}

**3.2 Book DAO (dao/BookDAO.java)**

package dao;

import DatabaseConnection.DatabaseConnection;

import model.Book;

import java.sql.\*;

import java.util.ArrayList;

import java.util.List;

public class BookDAO {

// FIXED: Get all NON-DELETED books (for admin to manage ALL books, including borrowed ones)

public List<Book> getAllBooks() {

List<Book> books = new ArrayList<>();

// Admin should see ALL books (available or borrowed) as long as they're not deleted

String sql = "SELECT \* FROM books";

try (Connection conn = DatabaseConnection.getConnection();

Statement stmt = conn.createStatement();

ResultSet rs = stmt.executeQuery(sql)) {

while (rs.next()) {

Book book = new Book();

book.setId(rs.getInt("id"));

book.setTitle(rs.getString("title"));

book.setAuthor(rs.getString("author"));

book.setIsbn(rs.getString("isbn"));

book.setAvailable(rs.getBoolean("available"));

books.add(book);

}

} catch (SQLException e) {

// Log the exact error for debugging

System.err.println("ERROR in getAllBooks(): " + e.getMessage());

}

return books;

}

// Search books (for user dashboard - only AVAILABLE books)

public List<Book> searchBooks(String keyword) {

List<Book> books = new ArrayList<>();

// Only show AVAILABLE books to users

String sql = "SELECT \* FROM books WHERE deleted = false AND available = true AND (title ILIKE ? OR author ILIKE ?)";

try (Connection conn = DatabaseConnection.getConnection();

PreparedStatement stmt = conn.prepareStatement(sql)) {

String likeKeyword = "%" + keyword + "%";

stmt.setString(1, likeKeyword);

stmt.setString(2, likeKeyword);

ResultSet rs = stmt.executeQuery();

while (rs.next()) {

Book book = new Book();

book.setId(rs.getInt("id"));

book.setTitle(rs.getString("title"));

book.setAuthor(rs.getString("author"));

book.setIsbn(rs.getString("isbn"));

book.setAvailable(rs.getBoolean("available"));

books.add(book);

}

} catch (SQLException e) {

}

return books;

}

// Add new book

public boolean addBook(Book book) {

String sql = "INSERT INTO books (title, author, isbn) VALUES (?, ?, ?)";

try (Connection conn = DatabaseConnection.getConnection();

PreparedStatement stmt = conn.prepareStatement(sql)) {

stmt.setString(1, book.getTitle());

stmt.setString(2, book.getAuthor());

stmt.setString(3, book.getIsbn());

int rows = stmt.executeUpdate();

System.out.println("DEBUG: Added " + rows + " book(s)"); // Debug

return rows > 0;

} catch (SQLException e) {

System.err.println("ERROR in addBook(): " + e.getMessage());

}

return false;

}

// Soft delete (mark as deleted)

public boolean deleteBook(int id) {

String sql = "UPDATE books SET deleted = true WHERE id = ?";

try (Connection conn = DatabaseConnection.getConnection();

PreparedStatement stmt = conn.prepareStatement(sql)) {

stmt.setInt(1, id);

int rows = stmt.executeUpdate();

System.out.println("DEBUG: Deleted " + rows + " book(s)"); // Debug

return rows > 0;

} catch (SQLException e) {

System.err.println("ERROR in deleteBook(): " + e.getMessage());

}

return false;

}

// Get book by ID

public Book getBookById(int id) {

String sql = "SELECT \* FROM books WHERE id = ?";

try (Connection conn = DatabaseConnection.getConnection();

PreparedStatement stmt = conn.prepareStatement(sql)) {

stmt.setInt(1, id);

ResultSet rs = stmt.executeQuery();

if (rs.next()) {

Book book = new Book();

book.setId(rs.getInt("id"));

book.setTitle(rs.getString("title"));

book.setAuthor(rs.getString("author"));

book.setIsbn(rs.getString("isbn"));

book.setAvailable(rs.getBoolean("available"));

return book;

}

} catch (SQLException e) {

}

return null;

}

// Update book

public boolean updateBook(Book book) {

String sql = "UPDATE books SET title = ?, author = ?, isbn = ? WHERE id = ?";

try (Connection conn = DatabaseConnection.getConnection();

PreparedStatement stmt = conn.prepareStatement(sql)) {

stmt.setString(1, book.getTitle());

stmt.setString(2, book.getAuthor());

stmt.setString(3, book.getIsbn());

stmt.setInt(4, book.getId());

return stmt.executeUpdate() > 0;

} catch (SQLException e) {

}

return false;

}

}

**Section 4: Servlets (Backend Controllers)**

**4.1 Login Servlet (com.sandra.digitallibrary/LoginServlet.java)**

package com.sandra.digitallibrary;

import dao.UserDAO;

import model.User;

import javax.servlet.\*;

import javax.servlet.http.\*;

import javax.servlet.annotation.WebServlet;

import java.io.IOException;

import java.sql.Connection;

import java.sql.PreparedStatement;

@WebServlet("/LoginServlet")

public class LoginServlet extends HttpServlet {

@Override

protected void doPost(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

// Get username and password from form

String username = request.getParameter("username");

String password = request.getParameter("password");

System.out.println("DEBUG: Login attempt - Username: " + username + ", Password: " + password);

// Authenticate user

UserDAO userDAO = new UserDAO();

User user = userDAO.authenticate(username, password);

if (user != null) {

System.out.println("DEBUG: Login successful for " + user.getUsername());

// Update last\_login timestamp

try (Connection conn = DatabaseConnection.DatabaseConnection.getConnection()) {

PreparedStatement ps = conn.prepareStatement(

"UPDATE users SET last\_login = CURRENT\_TIMESTAMP WHERE id = ?");

ps.setInt(1, user.getId());

ps.executeUpdate();

} catch (Exception e) {

e.printStackTrace();

}

// Create session

HttpSession session = request.getSession();

session.setAttribute("user", user);

if ("admin".equals(user.getRole())) {

response.sendRedirect("admin\_dashboard.jsp");

} else {

response.sendRedirect("user\_dashboard.jsp");

}

} else {

System.out.println("DEBUG: Login failed for " + username);

request.setAttribute("error", "Invalid username or password");

request.getRequestDispatcher("login.jsp").forward(request, response);

}

}

}

**4.2 Register Servlet (com.sandra.digitallibrary/RegisterServlet.java)**

package com.sandra.digitallibrary;

import dao.UserDAO;

import javax.servlet.\*;

import javax.servlet.http.\*;

import javax.servlet.annotation.WebServlet;

import java.io.IOException;

@WebServlet("/RegisterServlet")

public class RegisterServlet extends HttpServlet {

@Override

protected void doPost(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

// Log what we received

String username = request.getParameter("username");

String password = request.getParameter("password");

String email = request.getParameter("email");

System.out.println("REGISTER ATTEMPT:");

System.out.println("Username: " + username);

System.out.println("Password: " + password);

System.out.println("Email: " + email);

UserDAO userDAO = new UserDAO();

boolean success = userDAO.registerUser(username, password, email);

if (success) {

System.out.println(" Registration successful!");

response.sendRedirect("login.jsp?registered=1");

} else {

System.out.println("Registration failed!");

request.setAttribute("error", "Registration failed. Username may exist.");

request.getRequestDispatcher("register.jsp").forward(request, response);

}

}

}

**4.3 Book Management Servlets**

**4.3.1 Add Book Servlet (com.sandra.digitallibrary/AddBookServlet.java)**

package com.sandra.digitallibrary;

import dao.BookDAO;

import model.Book;

import javax.servlet.\*;

import javax.servlet.http.\*;

import javax.servlet.annotation.WebServlet;

import java.io.IOException;

@WebServlet("/AddBookServlet")

public class AddBookServlet extends HttpServlet {

@Override

protected void doPost(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

// Get parameters

String title = request.getParameter("title");

String author = request.getParameter("author");

String isbn = request.getParameter("isbn");

// Validate required fields

if (title == null || title.trim().isEmpty()) {

request.setAttribute("error", "Title is required!");

request.getRequestDispatcher("add\_book.jsp").forward(request, response);

return;

}

if (author == null || author.trim().isEmpty()) {

request.setAttribute("error", "Author is required!");

request.getRequestDispatcher("add\_book.jsp").forward(request, response);

return;

}

// Log what we received

System.out.println("ADD BOOK REQUEST:");

System.out.println("Title: " + title);

System.out.println("Author: " + author);

System.out.println("ISBN: " + isbn);

Book book = new Book(title, author, isbn);

BookDAO bookDAO = new BookDAO();

boolean success = bookDAO.addBook(book);

if (success) {

System.out.println(" Book added successfully!");

response.sendRedirect("admin\_books.jsp?added=1");

} else {

System.out.println("Failed to add book!");

request.setAttribute("error", "Failed to add book. ISBN may already exist.");

request.getRequestDispatcher("add\_book.jsp").forward(request, response);

}

}

}

**4.3.2 Delete Book Servlet (com.sandra.digitallibrary/DeleteBookServlet.java)**

package com.sandra.digitallibrary;

import dao.BookDAO;

import javax.servlet.\*;

import javax.servlet.http.\*;

import javax.servlet.annotation.WebServlet;

import java.io.IOException;

@WebServlet("/DeleteBookServlet")

public class DeleteBookServlet extends HttpServlet {

@Override

protected void doGet(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

// Log what we received

int id = Integer.parseInt(request.getParameter("id"));

System.out.println("DELETE BOOK REQUEST: ID = " + id);

BookDAO bookDAO = new BookDAO();

boolean success = bookDAO.deleteBook(id);

if (success) {

System.out.println(" Book deleted successfully!");

response.sendRedirect("admin\_books.jsp?deleted=1");

} else {

System.out.println("Failed to delete book!");

request.setAttribute("error", "Failed to delete book.");

request.getRequestDispatcher("admin\_books.jsp").forward(request, response);

}

}

}

**4.4 Borrow/Return Servlets**

**4.4.1 Borrow Book Servlet (com.sandra.digitallibrary/BorrowBookServlet.java)**

package com.sandra.digitallibrary;

import dao.BookDAO;

import model.User;

import javax.servlet.\*;

import javax.servlet.http.\*;

import javax.servlet.annotation.WebServlet;

import java.io.IOException;

import java.sql.\*;

@WebServlet("/BorrowBookServlet")

public class BorrowBookServlet extends HttpServlet {

@Override

protected void doGet(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

User user = (User) request.getSession().getAttribute("user");

if (user == null) {

response.sendRedirect("login.jsp");

return;

}

int bookId = Integer.parseInt(request.getParameter("id"));

try (Connection conn = DatabaseConnection.DatabaseConnection.getConnection()) {

// Check if book is available

PreparedStatement check = conn.prepareStatement(

"SELECT available FROM books WHERE id = ?");

check.setInt(1, bookId);

ResultSet rs = check.executeQuery();

if (rs.next() && rs.getBoolean("available")) {

// Mark book as unavailable

PreparedStatement updateBook = conn.prepareStatement(

"UPDATE books SET available = false WHERE id = ?");

updateBook.setInt(1, bookId);

updateBook.executeUpdate();

// Record borrow with current date

PreparedStatement insertBorrow = conn.prepareStatement(

"INSERT INTO borrow\_records (user\_id, book\_id, borrow\_date) VALUES (?, ?, CURRENT\_DATE)");

insertBorrow.setInt(1, user.getId());

insertBorrow.setInt(2, bookId);

insertBorrow.executeUpdate();

response.sendRedirect("user\_dashboard.jsp?borrowed=1");

} else {

response.sendRedirect("book\_details.jsp?id=" + bookId + "&error=unavailable");

}

} catch (Exception e) {

e.printStackTrace();

response.sendRedirect("user\_dashboard.jsp?error=server");

}

}

}

**4.4.2 Return Book Servlet (com.sandra.digitallibrary/ReturnBookServlet.java)**

package com.sandra.digitallibrary;

import dao.BookDAO;

import model.User;

import javax.servlet.\*;

import javax.servlet.http.\*;

import javax.servlet.annotation.WebServlet;

import java.io.IOException;

import java.sql.\*;

@WebServlet("/ReturnBookServlet")

public class ReturnBookServlet extends HttpServlet {

@Override

protected void doGet(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

User user = (User) request.getSession().getAttribute("user");

if (user == null) {

response.sendRedirect("login.jsp");

return;

}

int borrowId = Integer.parseInt(request.getParameter("id"));

try (Connection conn = DatabaseConnection.DatabaseConnection.getConnection()) {

// Check if user actually borrowed this book

PreparedStatement check = conn.prepareStatement(

"SELECT \* FROM borrow\_records WHERE id = ? AND user\_id = ? AND return\_date IS NULL");

check.setInt(1, borrowId);

check.setInt(2, user.getId());

ResultSet rs = check.executeQuery();

if (rs.next()) {

// Mark book as available

PreparedStatement updateBook = conn.prepareStatement(

"UPDATE books SET available = true WHERE id = ?");

updateBook.setInt(1, rs.getInt("book\_id"));

updateBook.executeUpdate();

// Record return date

PreparedStatement updateBorrow = conn.prepareStatement(

"UPDATE borrow\_records SET return\_date = CURRENT\_DATE WHERE id = ?");

updateBorrow.setInt(1, borrowId);

updateBorrow.executeUpdate();

// Redirect with success message

response.sendRedirect("my\_borrowed\_books.jsp?returned=1");

} else {

response.sendRedirect("my\_borrowed\_books.jsp?error=not\_borrowed");

}

} catch (Exception e) {

e.printStackTrace();

response.sendRedirect("my\_borrowed\_books.jsp?error=server");

}

}

}

**Section 5: JSP Pages (Frontend Views)**

**5.1 Header & Footer**

**5.1.1 Header (header.jsp)**

<%@ page contentType="text/html;charset=UTF-8" language="java" %>

<%@ page import="model.User" %>

<%

User currentUser = (User) session.getAttribute("user");

boolean isAdmin = false;

if (currentUser != null && "admin".equals(currentUser.getRole())) {

isAdmin = true;

}

// Get current page URI for active link highlighting

String uri = request.getRequestURI();

boolean isUserDashboard = uri.contains("user\_dashboard.jsp");

boolean isMyBorrowedBooks = uri.contains("my\_borrowed\_books.jsp");

boolean isAdminDashboard = uri.contains("admin\_dashboard.jsp");

boolean isAdminBooks = uri.contains("admin\_books.jsp");

boolean isManageUsers = uri.contains("manage\_users.jsp");

boolean isBorrowHistory = uri.contains("borrow\_history.jsp");

%>

<!DOCTYPE html>

<html>

<head>

<meta charset="UTF-8">

<title>Digital Library</title>

<!-- Fixed: removed extra spaces -->

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css" rel="stylesheet">

<style>

body { padding-top: 56px; }

.navbar { box-shadow: 0 2px 4px rgba(0,0,0,0.1); }

</style>

</head>

<body>

<nav class="navbar navbar-expand-lg navbar-dark bg-primary fixed-top">

<div class="container">

<a class="navbar-brand" href="<%= isAdmin ? "admin\_dashboard.jsp" : (currentUser != null ? "my\_borrowed\_books.jsp" : "login.jsp") %>">

DIGITAL LIBRARY

</a>

<div class="navbar-nav ms-auto">

<%

if (currentUser != null) {

if (isAdmin) {

%>

<a class="nav-link text-white <%= isAdminDashboard ? "fw-bold" : "" %>" href="admin\_dashboard.jsp">Dashboard</a>

<a class="nav-link text-white <%= isAdminBooks ? "fw-bold" : "" %>" href="admin\_books.jsp">Manage Books</a>

<a class="nav-link text-white <%= isManageUsers ? "fw-bold" : "" %>" href="manage\_users.jsp">Manage Users</a>

<a class="nav-link text-white <%= isBorrowHistory ? "fw-bold" : "" %>" href="borrow\_history.jsp">Borrow History</a>

<%

} else {

%>

<a class="nav-link text-white <%= isMyBorrowedBooks ? "fw-bold" : "" %>" href="my\_borrowed\_books.jsp">My Books</a>

<a class="nav-link text-white <%= isUserDashboard ? "fw-bold" : "" %>" href="user\_dashboard.jsp">Available Books</a>

<%

}

%>

<span class="navbar-text text-white me-3">

Hello, <%= currentUser.getUsername() %>

</span>

<a class="nav-link text-white" href="logout.jsp">Logout</a>

<%

}

%>

</div>

</div>

</nav>

**5.1.2 Footer (footer.jsp)**

<%@ page contentType="text/html;charset=UTF-8" language="java" %>

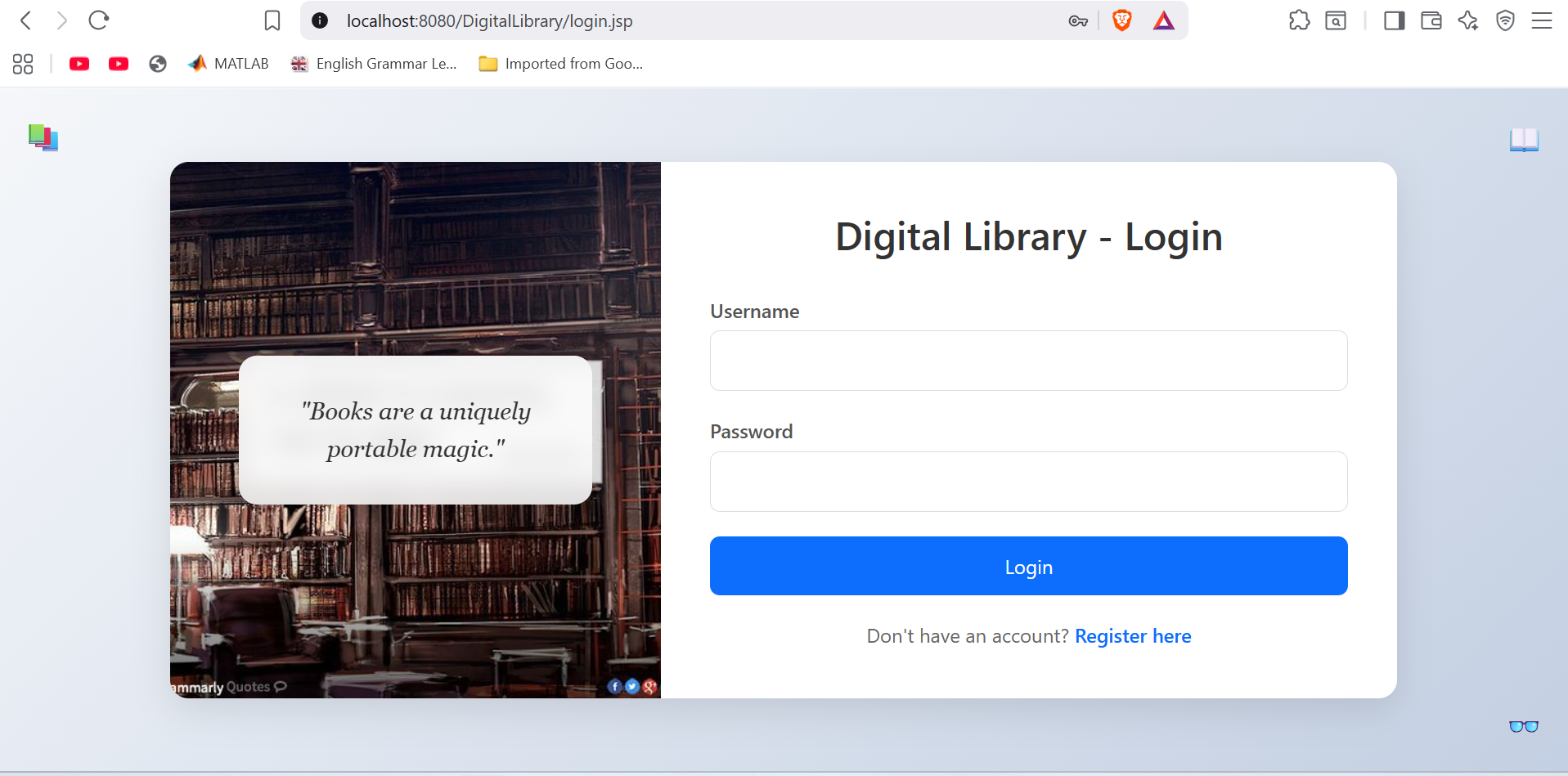
<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.bundle.min.js"></script>

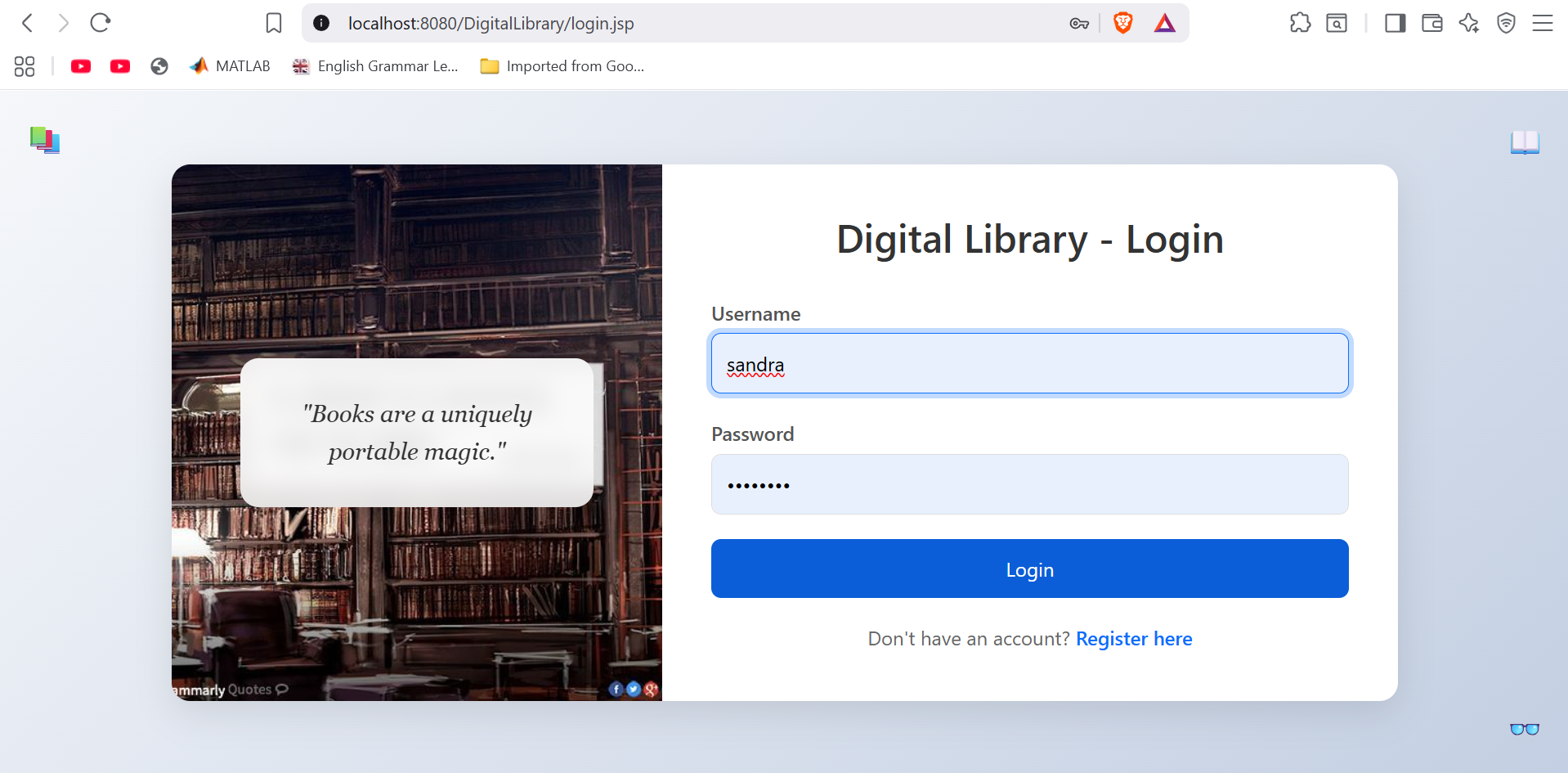
</body>

</html>

**5.2 Authentication Pages**

**5.2.1 Login Page (login.jsp)**

****

****

<%@ page contentType="text/html;charset=UTF-8" language="java" %>

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<title>Digital Library - Login</title>

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css" rel="stylesheet">

<style>

body {

font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;

background: linear-gradient(135deg, #f5f7fa 0%, #c3cfe2 100%);

min-height: 100vh;

margin: 0;

padding: 0;

display: flex;

align-items: center;

justify-content: center;

position: relative;

}

.login-container {

display: flex;

width: 90%;

max-width: 1000px;

box-shadow: 0 10px 30px rgba(0,0,0,0.1);

border-radius: 15px;

overflow: hidden;

background: white;

}

.login-image {

width: 40%;

/\* YOUR PINTEREST IMAGE URL \*/

background: url('https://i.pinimg.com/564x/40/cc/22/40cc229415acfff1801b8c9dc2015c77.jpg') no-repeat center center;

background-size: cover;

position: relative;

display: flex;

align-items: center;

justify-content: center;

padding: 20px;

}

.quote-box {

background: rgba(255,255,255,0.85);

padding: 30px;

border-radius: 15px;

text-align: center;

max-width: 80%;

backdrop-filter: blur(10px);

}

.quote-text {

font-size: 1.2rem;

font-style: italic;

color: #333;

line-height: 1.6;

font-family: 'Georgia', serif;

}

.login-form {

width: 60%;

padding: 40px;

display: flex;

flex-direction: column;

justify-content: center;

}

.login-title {

font-size: 2rem;

font-weight: 600;

color: #333;

margin-bottom: 30px;

text-align: center;

}

.form-group {

margin-bottom: 20px;

}

.form-label {

font-weight: 500;

color: #555;

margin-bottom: 5px;

}

.form-control {

padding: 12px;

font-size: 1rem;

border: 1px solid #ddd;

border-radius: 8px;

transition: border-color 0.3s ease;

}

.form-control:focus {

border-color: #0d6efd;

box-shadow: 0 0 0 0.25rem rgba(13, 110, 253, 0.25);

}

.btn-login {

background: #0d6efd;

color: white;

border: none;

padding: 12px 20px;

font-size: 1rem;

border-radius: 8px;

cursor: pointer;

transition: background 0.3s ease;

width: 100%;

}

.btn-login:hover {

background: #0b5ed7;

}

.register-link {

text-align: center;

margin-top: 20px;

color: #666;

}

.register-link a {

color: #0d6efd;

text-decoration: none;

font-weight: 500;

}

.register-link a:hover {

text-decoration: underline;

}

/\* Corner Icons \*/

.top-left, .top-right {

position: absolute;

top: 20px;

font-size: 1.5rem;

color: #666;

}

.top-left {

left: 20px;

}

.top-right {

right: 20px;

}

/\* Responsive \*/

@media (max-width: 768px) {

.login-container {

flex-direction: column;

width: 95%;

}

.login-image {

width: 100%;

height: 200px;

}

.login-form {

width: 100%;

padding: 30px;

}

.login-title {

font-size: 1.5rem;

}

}

</style>

</head>

<body>

<!-- Corner Icons -->

<div class="top-left">📚</div>

<div class="top-right">📖</div>

<div class="login-container">

<!-- Left Side: Your Pinterest Book Image + Quote -->

<div class="login-image">

<div class="quote-box">

<div class="quote-text">

"Books are a uniquely portable magic."

</div>

</div>

</div>

<!-- Right Side: Login Form -->

<div class="login-form">

<h2 class="login-title">Digital Library - Login</h2>

<%

String error = (String) request.getAttribute("error");

if (error != null) {

out.println("<div class='alert alert-danger'>" + error + "</div>");

}

%>

<form method="post" action="LoginServlet">

<div class="form-group">

<label for="username" class="form-label">Username</label>

<input type="text" id="username" name="username" class="form-control" required>

</div>

<div class="form-group">

<label for="password" class="form-label">Password</label>

<input type="password" id="password" name="password" class="form-control" required>

</div>

<button type="submit" class="btn-login">Login</button>

</form>

<div class="register-link">

Don't have an account? <a href="register.jsp">Register here</a>

</div>

</div>

</div>

<!-- Bottom Icon -->

<div style="position: fixed; bottom: 20px; right: 20px; font-size: 1.5rem; color: #666;">👓</div>

</body>

</html>

**5.2.2 Registration Page (register.jsp)**

****

<%@ page contentType="text/html;charset=UTF-8" language="java" %>

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<title>Digital Library - Register</title>

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css" rel="stylesheet">

<style>

body {

font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;

background: linear-gradient(135deg, #f5f7fa 0%, #c3cfe2 100%);

min-height: 100vh;

margin: 0;

padding: 0;

display: flex;

align-items: center;

justify-content: center;

position: relative;

}

.register-container {

display: flex;

width: 90%;

max-width: 1000px;

box-shadow: 0 10px 30px rgba(0,0,0,0.1);

border-radius: 15px;

overflow: hidden;

background: white;

}

.register-image {

width: 40%;

/\* YOUR GOOGLE IMAGE URL \*/

background: url('https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcTyfzeitXQYpIrNqsHaiEBkgkIEPG1KCKkisvZEqR0lR4X0An5z5n4OJHmulw7XFJzDGOY&usqp=CAU') no-repeat center center;

background-size: cover;

position: relative;

display: flex;

align-items: center;

justify-content: center;

padding: 20px;

color: white;

}

.quote-box {

text-align: center;

padding: 30px;

max-width: 80%;

backdrop-filter: blur(10px);

}

.quote-text {

font-size: 1.1rem;

font-style: italic;

line-height: 1.6;

font-family: 'Georgia', serif;

text-shadow: 0 2px 4px rgba(0,0,0,0.3);

}

.register-form {

width: 60%;

padding: 40px;

display: flex;

flex-direction: column;

justify-content: center;

}

.register-title {

font-size: 2rem;

font-weight: 600;

color: #333;

margin-bottom: 30px;

text-align: center;

}

.form-group {

margin-bottom: 20px;

}

.form-label {

font-weight: 500;

color: #555;

margin-bottom: 5px;

}

.form-control {

padding: 12px;

font-size: 1rem;

border: 1px solid #ddd;

border-radius: 8px;

transition: border-color 0.3s ease;

}

.form-control:focus {

border-color: #0d6efd;

box-shadow: 0 0 0 0.25rem rgba(13, 110, 253, 0.25);

}

.btn-register {

background: #198754;

color: white;

border: none;

padding: 12px 20px;

font-size: 1rem;

border-radius: 8px;

cursor: pointer;

transition: background 0.3s ease;

width: 100%;

}

.btn-register:hover {

background: #157347;

}

.login-link {

text-align: center;

margin-top: 20px;

color: #666;

}

.login-link a {

color: #0d6efd;

text-decoration: none;

font-weight: 500;

}

.login-link a:hover {

text-decoration: underline;

}

/\* Corner Icons \*/

.top-left, .top-right {

position: absolute;

top: 20px;

font-size: 1.5rem;

color: #666;

}

.top-left {

left: 20px;

}

.top-right {

right: 20px;

}

/\* Bottom Quotes \*/

.bottom-quotes {

position: fixed;

bottom: 20px;

left: 50%;

transform: translateX(-50%);

text-align: center;

color: #666;

font-style: italic;

font-size: 0.9rem;

max-width: 80%;

}

/\* Responsive \*/

@media (max-width: 768px) {

.register-container {

flex-direction: column;

width: 95%;

}

.register-image {

width: 100%;

height: 200px;

}

.register-form {

width: 100%;

padding: 30px;

}

.register-title {

font-size: 1.5rem;

}

.bottom-quotes {

font-size: 0.8rem;

max-width: 90%;

}

}

</style>

</head>

<body>

<!-- Corner Icons -->

<div class="top-left">📚</div>

<div class="top-right">📖</div>

<div class="register-container">

<!-- Left Side: Image + Quote -->

<div class="register-image">

<div class="quote-box">

<div class="quote-text">

"A reader lives a thousand lives before he dies."

</div>

</div>

</div>

<!-- Right Side: Registration Form -->

<div class="register-form">

<h2 class="register-title">Register</h2>

<%

String error = (String) request.getAttribute("error");

if (error != null) {

out.println("<div class='alert alert-danger'>" + error + "</div>");

}

%>

<form method="post" action="RegisterServlet">

<div class="form-group">

<label for="username" class="form-label">Username</label>

<input type="text" id="username" name="username" class="form-control" placeholder="Choose a unique name" required>

</div>

<div class="form-group">

<label for="password" class="form-label">Password</label>

<input type="password" id="password" name="password" class="form-control" placeholder="Create a secure password" required>

</div>

<div class="form-group">

<label for="email" class="form-label">Email</label>

<input type="email" id="email" name="email" class="form-control" placeholder="Enter your email" required>

</div>

<button type="submit" class="btn-register">Register</button>

</form>

<div class="login-link">

Already have an account? <a href="login.jsp">Login</a>

</div>

</div>

</div>

<!-- Bottom Quotes -->

<div class="bottom-quotes">

<div>"The only thing you absolutely have to know is the location of the library."</div>

</div>

<!-- Bottom Icon -->

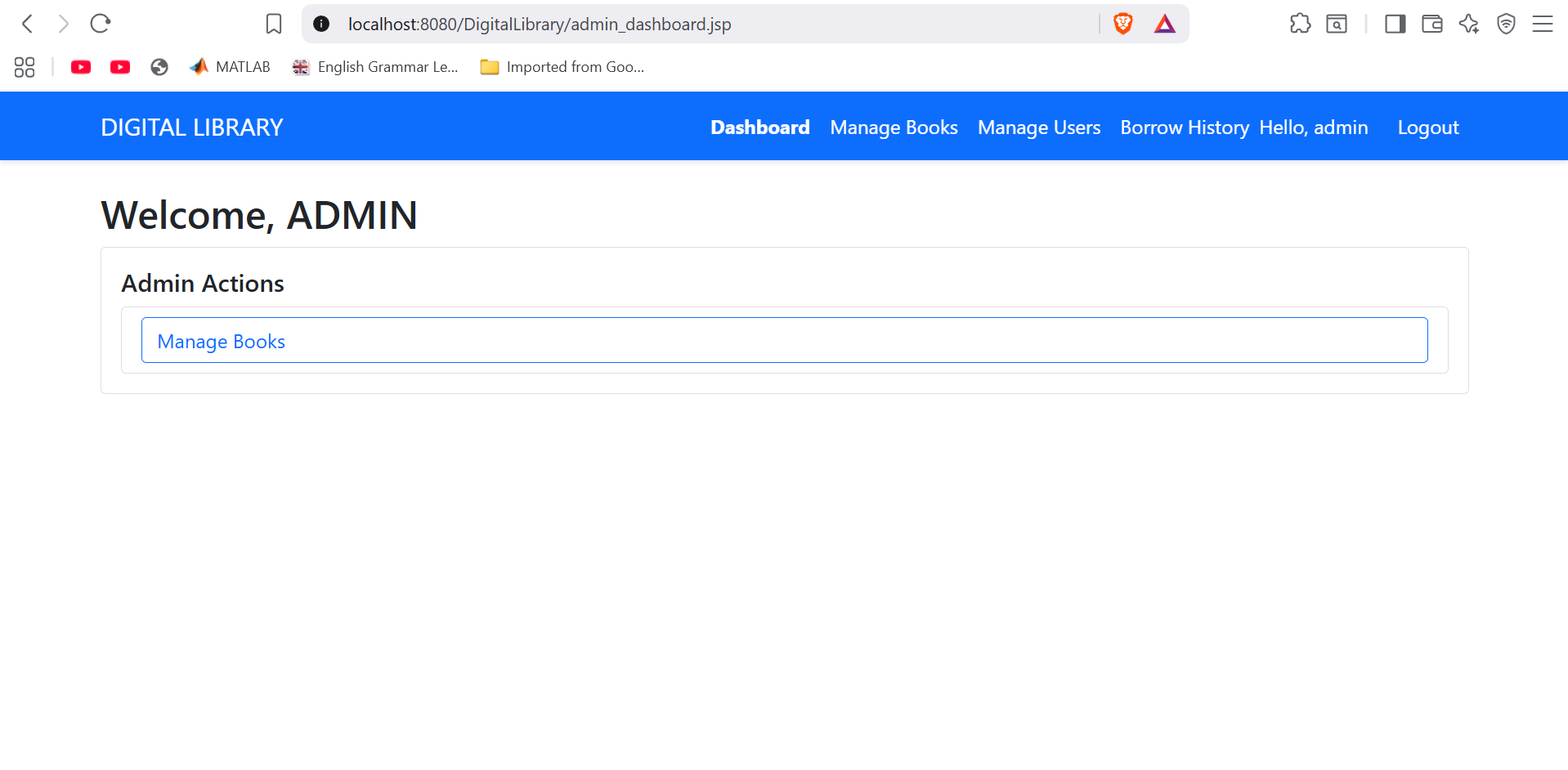
<div style="position: fixed; bottom: 20px; right: 20px; font-size: 1.5rem; color: #666;">👓</div>

</body>

</html>

**5.3 Admin Pages**

**5.3.1 Admin Dashboard (admin\_dashboard.jsp)**

****

<%@ page contentType="text/html;charset=UTF-8" language="java" %>

<jsp:include page="header.jsp" />

<div class="container mt-4">

<h2>Welcome, ADMIN</h2>

<div class="card">

<div class="card-body">

<h5 class="card-title">Admin Actions</h5>

<ul class="list-group">

<li class="list-group-item">

<a href="admin\_books.jsp" class="btn btn-outline-primary w-100 text-start">Manage Books</a>

</li>

</ul>

</div>

</div>

</div>

<jsp:include page="footer.jsp" />

**5.3.2 Manage Books (admin\_books.jsp)**

<%@ page import="dao.BookDAO, model.Book, java.util.List" %>

<%

if (session.getAttribute("user") == null || !"admin".equals(((model.User) session.getAttribute("user")).getRole())) {

response.sendRedirect("login.jsp");

return;

}

%>

<jsp:include page="header.jsp" />

<div class="container mt-4">

<div class="d-flex justify-content-between align-items-center mb-4">

<h2>Manage Books</h2>

<a href="add\_book.jsp" class="btn btn-success">+ Add New Book</a>

</div>

<%

// Show success messages

String msg = request.getParameter("added");

String delMsg = request.getParameter("deleted");

if ("1".equals(msg)) {

out.println("<div class='alert alert-success alert-dismissible fade show' role='alert'>" +

" Book added successfully!" +

"<button type='button' class='btn-close' data-bs-dismiss='alert'></button></div>");

}

if ("1".equals(delMsg)) {

out.println("<div class='alert alert-success alert-dismissible fade show' role='alert'>" +

" Book deleted!" +

"<button type='button' class='btn-close' data-bs-dismiss='alert'></button></div>");

}

%>

<%

try {

BookDAO bookDAO = new BookDAO();

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

// DEBUG: Print to NetBeans Output

System.out.println("DEBUG: admin\_books.jsp loaded " + books.size() + " books");

if (books.isEmpty()) {

%>

<div class="alert alert-info">

<h4>No books found</h4>

<p>Click <strong>"+ Add New Book"</strong> to add your first book to the library.</p>

</div>

<%

} else {

%>

<div class="table-responsive">

<table class="table table-bordered table-hover align-middle">

<thead class="table-dark">

<tr>

<th>ID</th>

<th>Title</th>

<th>Author</th>

<th>ISBN</th>

<th>Available</th>

<th>Actions</th>

</tr>

</thead>

<tbody>

<%

for (Book b : books) {

%>

<tr>

<td><%= b.getId() %></td>

<td><%= b.getTitle() %></td>

<td><%= b.getAuthor() %></td>

<td><%= b.getIsbn() %></td>

<td>

<% if (b.isAvailable()) { %>

<span class="badge bg-success">Available</span>

<% } else { %>

<span class="badge bg-warning text-dark">Borrowed</span>

<% } %>

</td>

<td>

<a href="DeleteBookServlet?id=<%= b.getId() %>"

class="btn btn-danger btn-sm"

onclick="return confirm('Are you sure you want to delete this book?')">

Delete

</a>

</td>

</tr>

<%

}

%>

</tbody>

</table>

</div>

<%

}

} catch (Exception e) {

e.printStackTrace();

%>

<div class="alert alert-danger">

<h4> Database Error!</h4>

<p>Check NetBeans Output for details.</p>

<pre><%= e.getMessage() %></pre>

</div>

<%

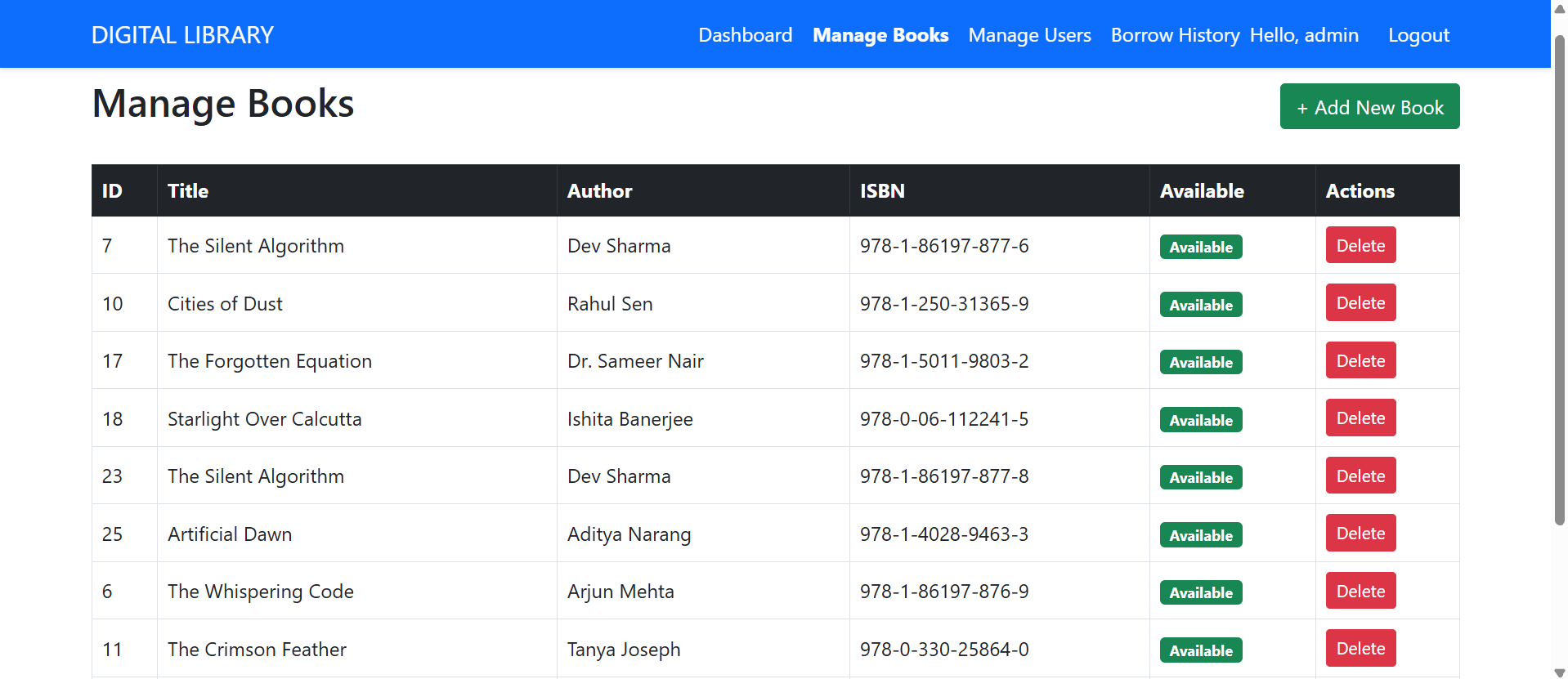
}

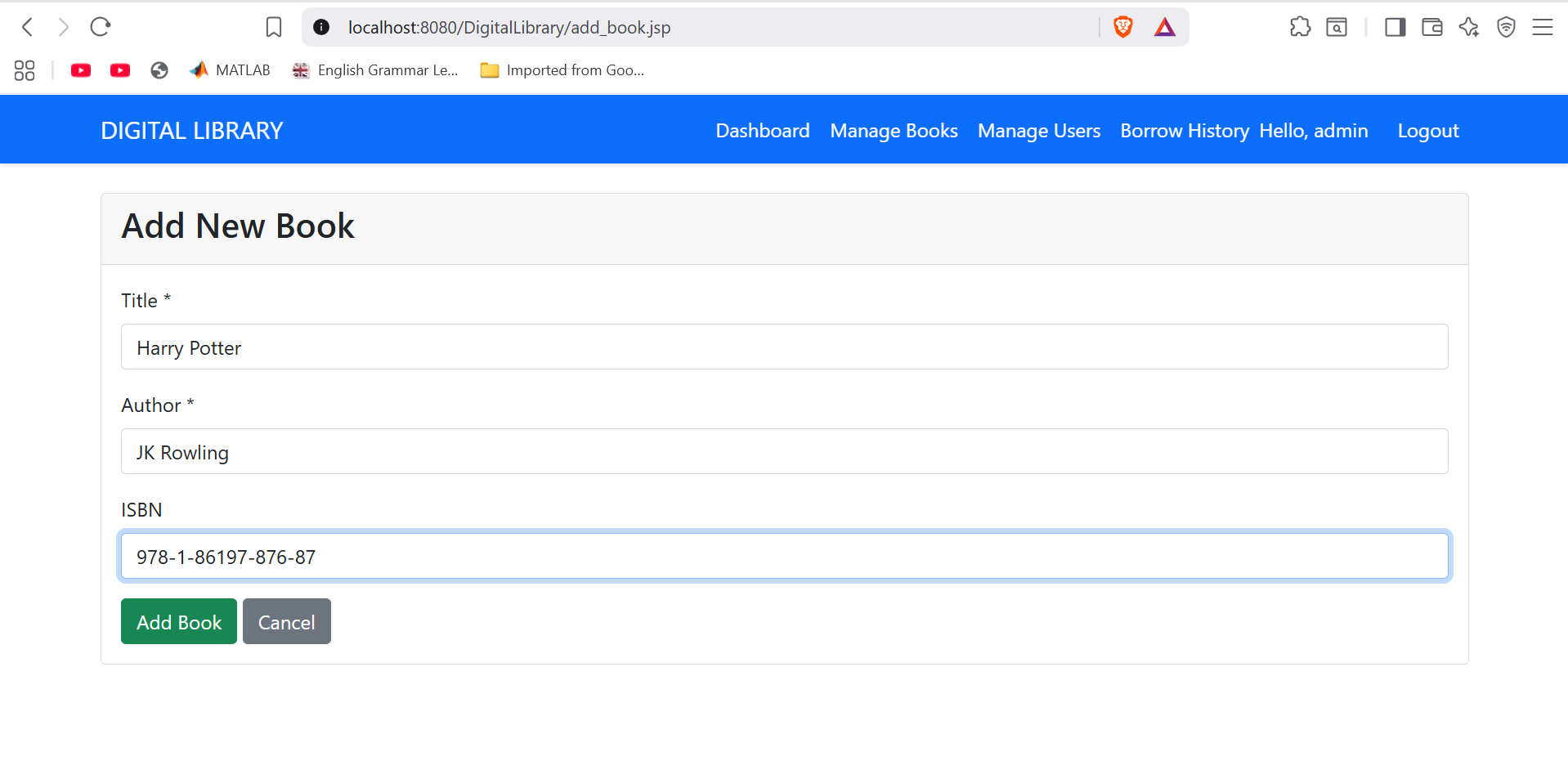
%>

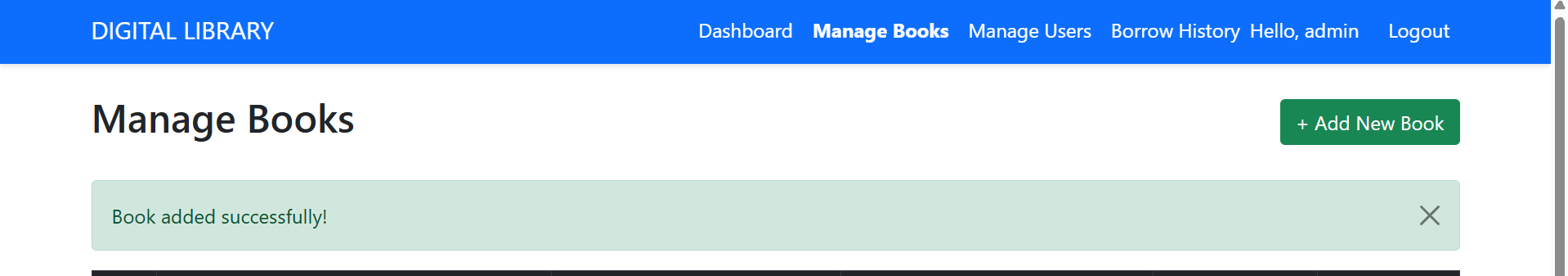
</div>

<jsp:include page="footer.jsp" />

**5.3.3 Add Book (add\_book.jsp)**

****

****

****

<%@ page contentType="text/html;charset=UTF-8" language="java" %>

<jsp:include page="header.jsp" />

<div class="container mt-4">

<div class="card">

<div class="card-header">

<h3>Add New Book</h3>

</div>

<div class="card-body">

<%

String error = (String) request.getAttribute("error");

if (error != null) {

out.println("<div class='alert alert-danger'>" + error + "</div>");

}

%>

<form method="post" action="AddBookServlet">

<div class="mb-3">

<label class="form-label">Title \*</label>

<input type="text" name="title" class="form-control"

value="<%= request.getParameter("title") != null ? request.getParameter("title") : "" %>"

required>

</div>

<div class="mb-3">

<label class="form-label">Author \*</label>

<input type="text" name="author" class="form-control"

value="<%= request.getParameter("author") != null ? request.getParameter("author") : "" %>"

required>

</div>

<div class="mb-3">

<label class="form-label">ISBN</label>

<input type="text" name="isbn" class="form-control"

value="<%= request.getParameter("isbn") != null ? request.getParameter("isbn") : "" %>">

</div>

<button type="submit" class="btn btn-success">Add Book</button>

<a href="admin\_books.jsp" class="btn btn-secondary">Cancel</a>

</form>

</div>

</div>

</div>

<jsp:include page="footer.jsp" />

**5.3.4 Manage Users (manage\_users.jsp)**

****

<%@ page import="dao.UserDAO, model.User, java.util.List" %>

<%

if (session.getAttribute("user") == null || !"admin".equals(((model.User) session.getAttribute("user")).getRole())) {

response.sendRedirect("login.jsp");

return;

}

%>

<jsp:include page="header.jsp" />

<div class="container mt-4">

<h2> Manage Users</h2>

<div class="table-responsive">

<table class="table table-bordered table-hover align-middle">

<thead class="table-dark">

<tr>

<th>ID</th>

<th>Username</th>

<th>Email</th>

<th>Role</th>

<th>Last Login</th>

</tr>

</thead>

<tbody>

<%

UserDAO userDAO = new UserDAO();

List<User> users = userDAO.getAllUsers();

for (User u : users) { // ? 'u' is now defined here!

%>

<tr>

<td><%= u.getId() %></td>

<td><%= u.getUsername() %></td>

<td><%= u.getEmail() %></td>

<td>

<% if ("admin".equals(u.getRole())) { %>

<span class="badge bg-danger">Admin</span>

<% } else { %>

<span class="badge bg-primary">User</span>

<% } %>

</td>

<td><%= u.getLastLogin() %></td> <!-- ? Now 'u' is resolved -->

</tr>

<%

}

%>

</tbody>

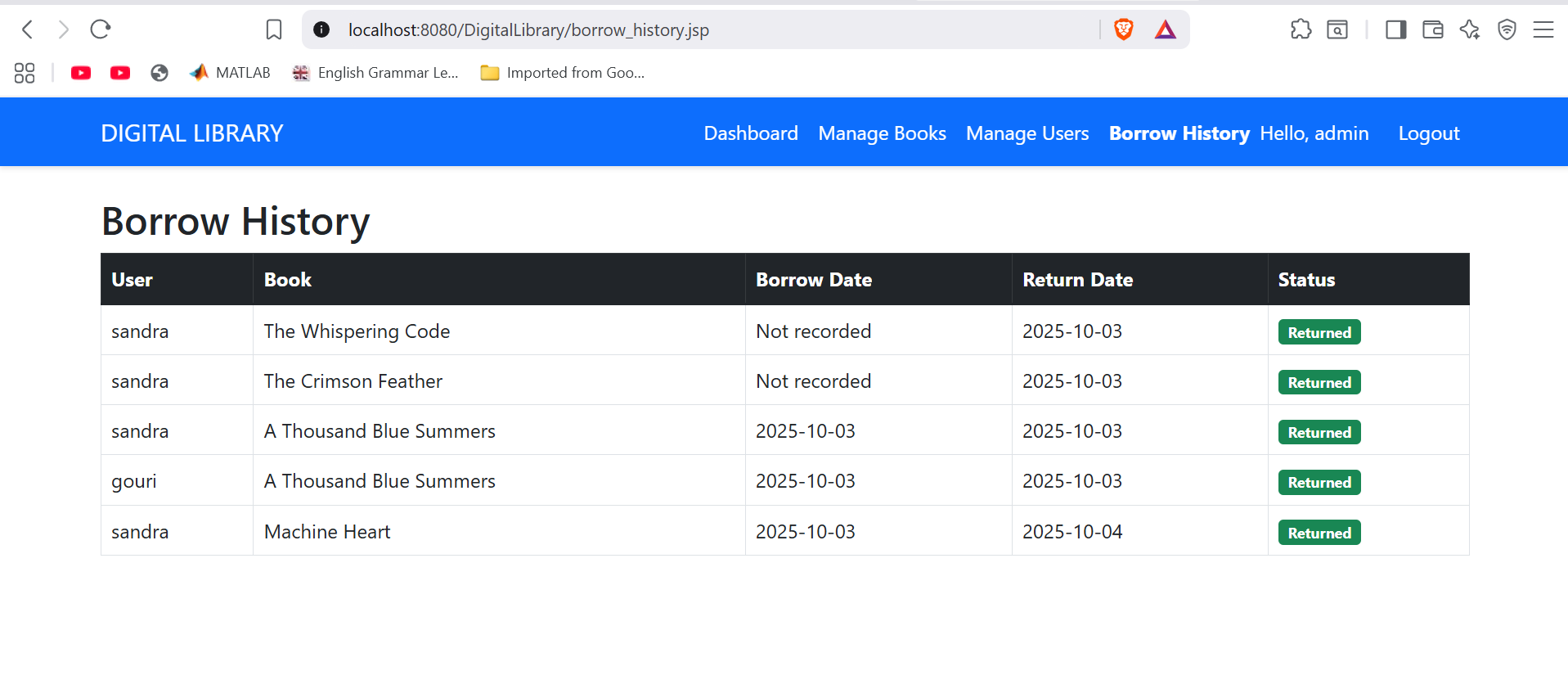
</table>

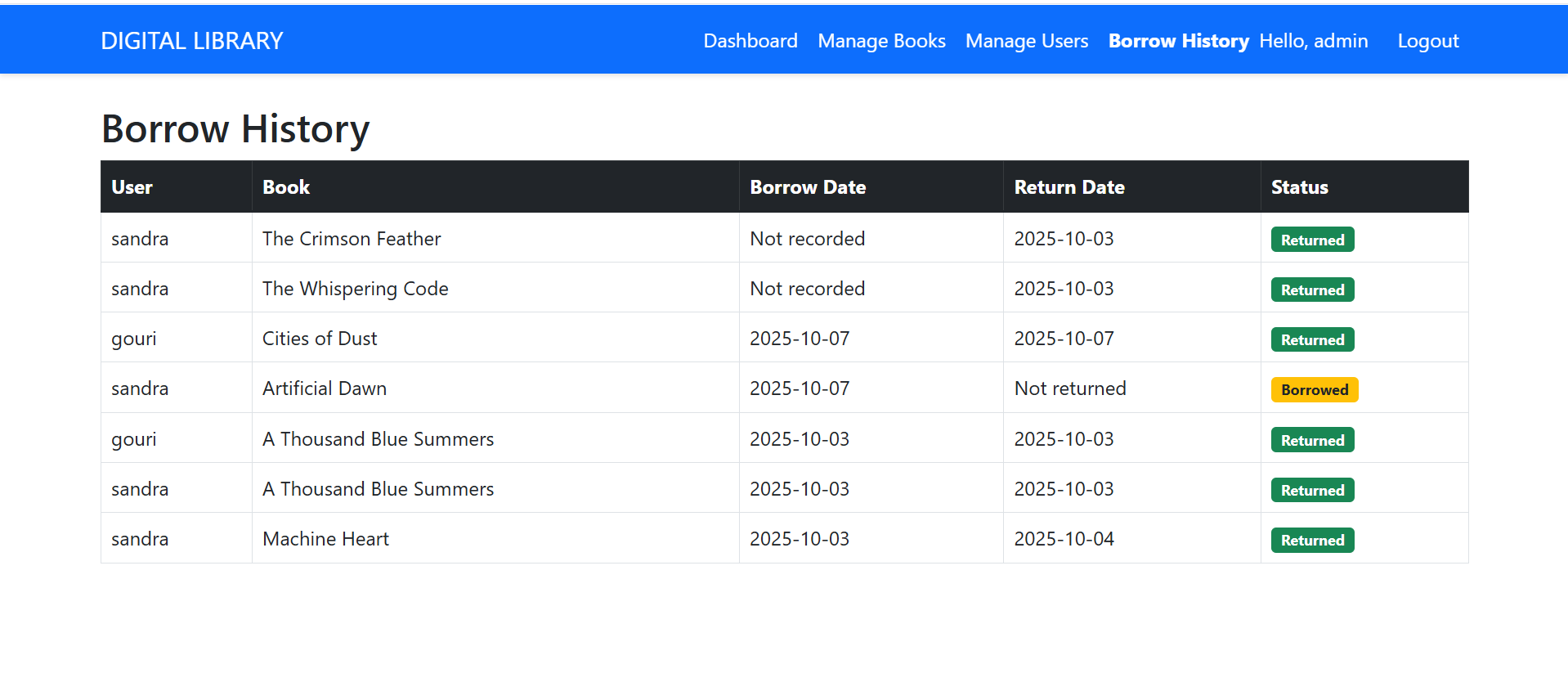
</div>

</div>

<jsp:include page="footer.jsp" />

**5.3.5 Borrow History (borrow\_history.jsp)**

****

****

<%@ page import="java.sql.\*, java.util.\*" %>

<%@ page import="DatabaseConnection.DatabaseConnection" %>

<%

if (session.getAttribute("user") == null || !"admin".equals(((model.User) session.getAttribute("user")).getRole())) {

response.sendRedirect("login.jsp");

return;

}

%>

<jsp:include page="header.jsp" />

<div class="container mt-4">

<h2>Borrow History</h2>

<div class="table-responsive">

<table class="table table-bordered table-hover">

<thead class="table-dark">

<tr>

<th>User</th>

<th>Book</th>

<th>Borrow Date</th>

<th>Return Date</th>

<th>Status</th>

</tr>

</thead>

<tbody>

<%

try (Connection conn = DatabaseConnection.getConnection()) {

String sql = "SELECT u.username, b.title, br.borrow\_date, br.return\_date " +

"FROM borrow\_records br " +

"JOIN users u ON br.user\_id = u.id " +

"JOIN books b ON br.book\_id = b.id " +

"ORDER BY br.borrow\_date DESC";

Statement stmt = conn.createStatement();

ResultSet rs = stmt.executeQuery(sql);

while (rs.next()) {

%>

<tr>

<td><%= rs.getString("username") %></td>

<td><%= rs.getString("title") %></td>

<td><%= rs.getDate("borrow\_date") != null ? rs.getDate("borrow\_date") : "Not recorded" %></td>

<td><%= rs.getDate("return\_date") != null ? rs.getDate("return\_date") : "Not returned" %></td>

<td>

<% if (rs.getDate("return\_date") == null) { %>

<span class="badge bg-warning text-dark">Borrowed</span>

<% } else { %>

<span class="badge bg-success">Returned</span>

<% } %>

</td>

</tr>

<%

}

} catch (SQLException e) {

e.printStackTrace();

%>

<tr>

<td colspan="5" class="text-danger">

<strong>Error loading borrow history:</strong> <%= e.getMessage() %>

</td>

</tr>

<%

}

%>

</tbody>

</table>

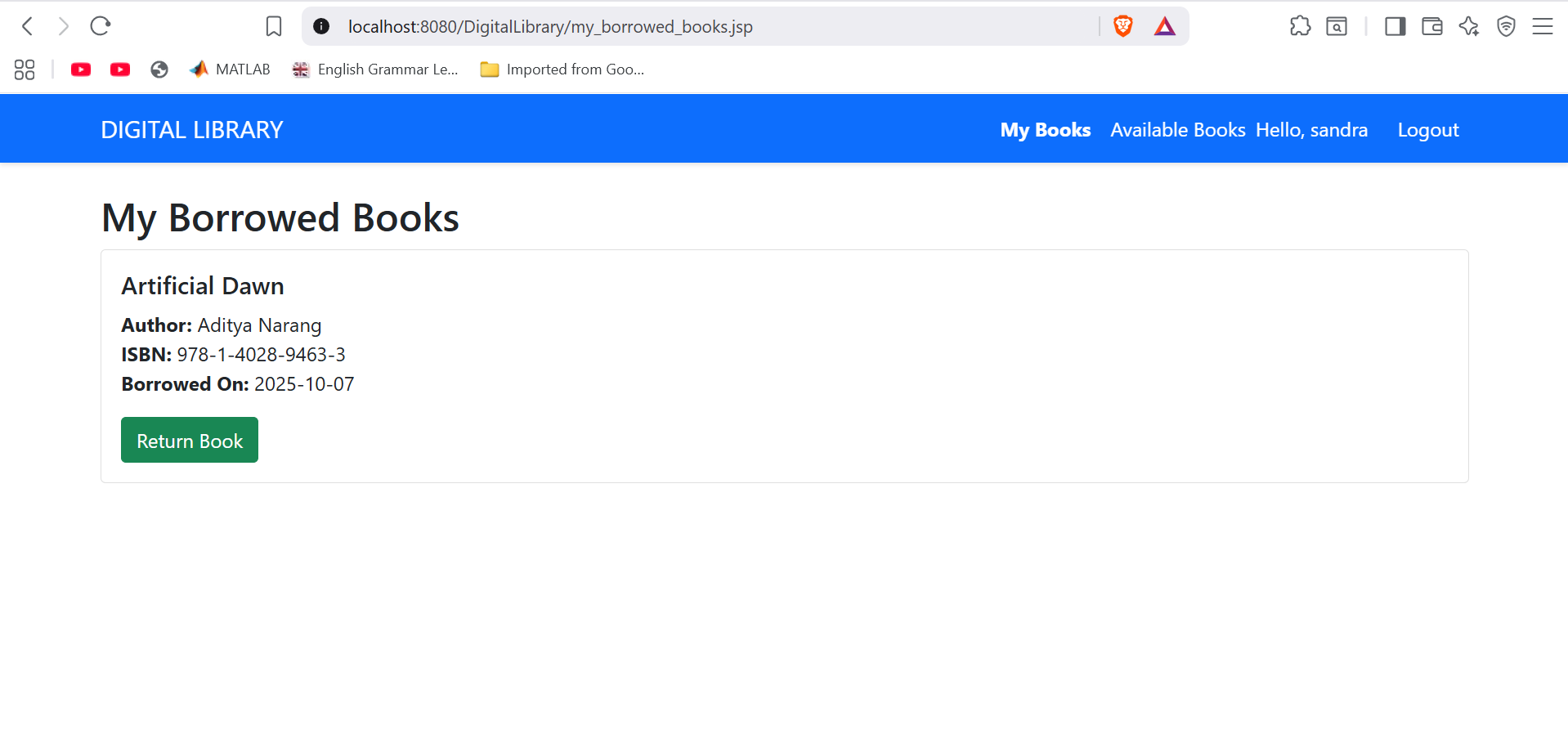
</div>

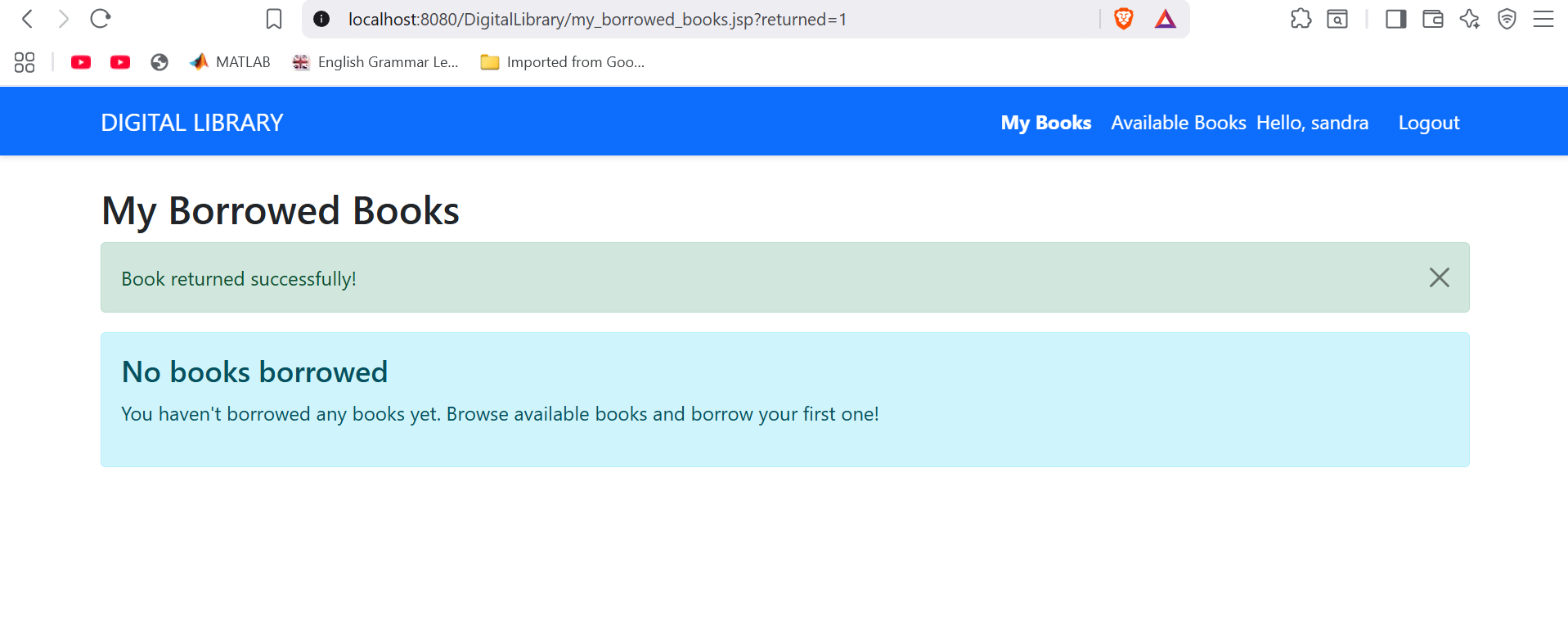
</div>

<jsp:include page="footer.jsp" />

**5.4 User Pages**

**5.4.1 My Borrowed Books (my\_borrowed\_books.jsp)**

****

****

<%@page import="model.User"%>

<%@ page import="java.sql.\*, java.util.\*" %>

<%

if (session.getAttribute("user") == null) {

response.sendRedirect("login.jsp");

return;

}

User currentUser = (User) session.getAttribute("user");

%>

<jsp:include page="header.jsp" />

<div class="container mt-4">

<h2>My Borrowed Books</h2>

<%

String returned = request.getParameter("returned");

if ("1".equals(returned)) {

out.println("<div class='alert alert-success alert-dismissible fade show' role='alert'>" +

" Book returned successfully!" +

"<button type='button' class='btn-close' data-bs-dismiss='alert'></button></div>");

}

String error = request.getParameter("error");

if ("not\_borrowed".equals(error)) {

out.println("<div class='alert alert-warning alert-dismissible fade show' role='alert'>" +

"? This book was not borrowed by you." +

"<button type='button' class='btn-close' data-bs-dismiss='alert'></button></div>");

}

if ("server".equals(error)) {

out.println("<div class='alert alert-danger alert-dismissible fade show' role='alert'>" +

" Server error. Please try again later." +

"<button type='button' class='btn-close' data-bs-dismiss='alert'></button></div>");

}

%>

<%

try (Connection conn = DatabaseConnection.DatabaseConnection.getConnection()) {

String sql = "SELECT b.title, b.author, b.isbn, br.borrow\_date, br.id as borrow\_id " +

"FROM borrow\_records br " +

"JOIN books b ON br.book\_id = b.id " +

"WHERE br.user\_id = ? AND br.return\_date IS NULL";

PreparedStatement stmt = conn.prepareStatement(sql);

stmt.setInt(1, currentUser.getId());

ResultSet rs = stmt.executeQuery();

boolean hasBorrowedBooks = false;

while (rs.next()) {

hasBorrowedBooks = true;

%>

<div class="card mb-3">

<div class="card-body">

<h5 class="card-title"><%= rs.getString("title") %></h5>

<p class="card-text">

<strong>Author:</strong> <%= rs.getString("author") %><br>

<strong>ISBN:</strong> <%= rs.getString("isbn") %><br>

<strong>Borrowed On:</strong> <%= rs.getDate("borrow\_date") != null ? rs.getDate("borrow\_date") : "Unknown" %>

</p>

<a href="ReturnBookServlet?id=<%= rs.getInt("borrow\_id") %>"

class="btn btn-success"> Return Book</a>

</div>

</div>

<%

}

if (!hasBorrowedBooks) {

%>

<div class="alert alert-info">

<h4>No books borrowed</h4>

<p>You haven't borrowed any books yet. Browse available books and borrow your first one!</p>

</div>

<%

}

} catch (SQLException e) {

e.printStackTrace();

%>

<div class="alert alert-danger">

<h4>Error loading borrowed books!</h4>

<p>Please try again later.</p>

</div>

<%

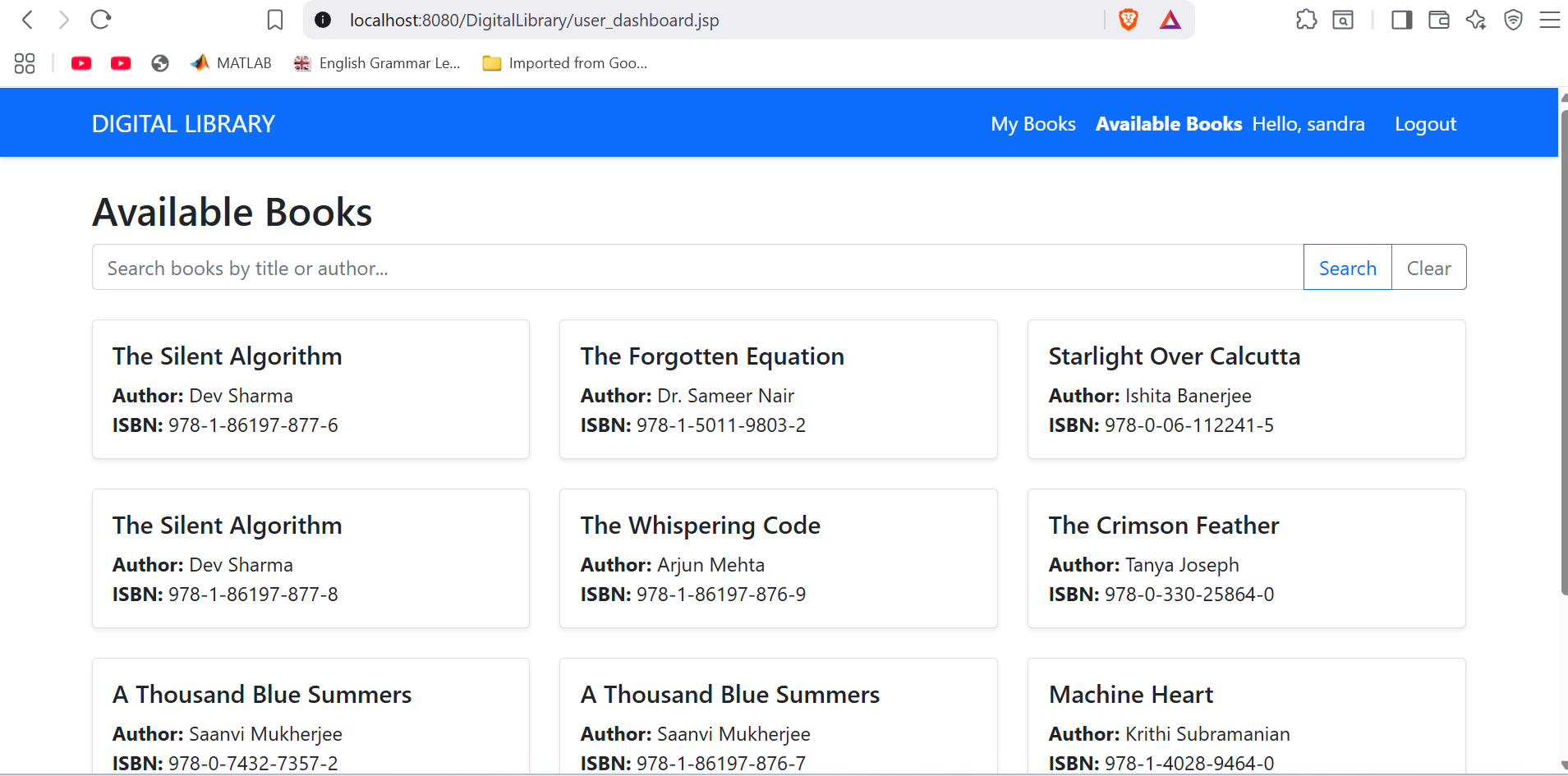
}

%>

</div>

<jsp:include page="footer.jsp" />

**5.4.2 Available Books (user\_dashboard.jsp)**

****

<%@ page import="dao.BookDAO, model.Book, java.util.List" %>

<%

// Redirect if not logged in

if (session.getAttribute("user") == null) {

response.sendRedirect("login.jsp");

return;

}

%>

<jsp:include page="header.jsp" />

<div class="container mt-4">

<h2>Available Books</h2>

<%

String returned = request.getParameter("returned");

if ("1".equals(returned)) {

out.println("<div class='alert alert-success alert-dismissible fade show' role='alert'>" +

" Book returned successfully!" +

"<button type='button' class='btn-close' data-bs-dismiss='alert'></button></div>");

}

%>

<%

// Show success message if book was borrowed

String borrowed = request.getParameter("borrowed");

if ("1".equals(borrowed)) {

out.println("<div class='alert alert-success alert-dismissible fade show' role='alert'>" +

" Book borrowed successfully!" +

"<button type='button' class='btn-close' data-bs-dismiss='alert'></button></div>");

}

String error = request.getParameter("error");

if ("unavailable".equals(error)) {

out.println("<div class='alert alert-warning alert-dismissible fade show' role='alert'>" +

"? This book is no longer available." +

"<button type='button' class='btn-close' data-bs-dismiss='alert'></button></div>");

}

%>

<!-- Search Bar -->

<form method="get" class="mb-4">

<div class="input-group">

<input type="text" name="search" class="form-control"

placeholder="Search books by title or author..."

value="<%= request.getParameter("search") != null ? request.getParameter("search") : "" %>">

<button class="btn btn-outline-primary" type="submit">Search</button>

<a href="user\_dashboard.jsp" class="btn btn-outline-secondary"> Clear</a>

</div>

</form>

<%

try {

BookDAO bookDAO = new BookDAO();

List<Book> books;

String search = request.getParameter("search");

if (search != null && !search.trim().isEmpty()) {

books = bookDAO.searchBooks(search.trim());

} else {

books = bookDAO.getAllBooks();

}

boolean hasAvailableBooks = false;

for (Book b : books) {

if (b.isAvailable()) {

hasAvailableBooks = true;

break;

}

}

if (!hasAvailableBooks) {

%>

<div class="alert alert-info">

<h4> No books available</h4>

<p>There are no books in the library right now. Please check back later!</p>

</div>

<%

} else {

%>

<div class="row">

<%

for (Book b : books) {

if (b.isAvailable()) {

%>

<div class="col-md-6 col-lg-4 mb-4">

<div class="card h-100 shadow-sm">

<div class="card-body d-flex flex-column">

<!-- Link to book details -->

<h5 class="card-title">

<a href="book\_details.jsp?id=<%= b.getId() %>" class="text-decoration-none text-dark">

<%= b.getTitle() %>

</a>

</h5>

<p class="card-text flex-grow-1">

<strong>Author:</strong> <%= b.getAuthor() %><br>

<strong>ISBN:</strong> <%= b.getIsbn() %>

</p>

</div>

</div>

</div>

<%

}

}

%>

</div>

<%

}

} catch (Exception e) {

e.printStackTrace();

%>

<div class="alert alert-danger">

<h4> Error loading books!</h4>

<p>Please try again later.</p>

</div>

<%

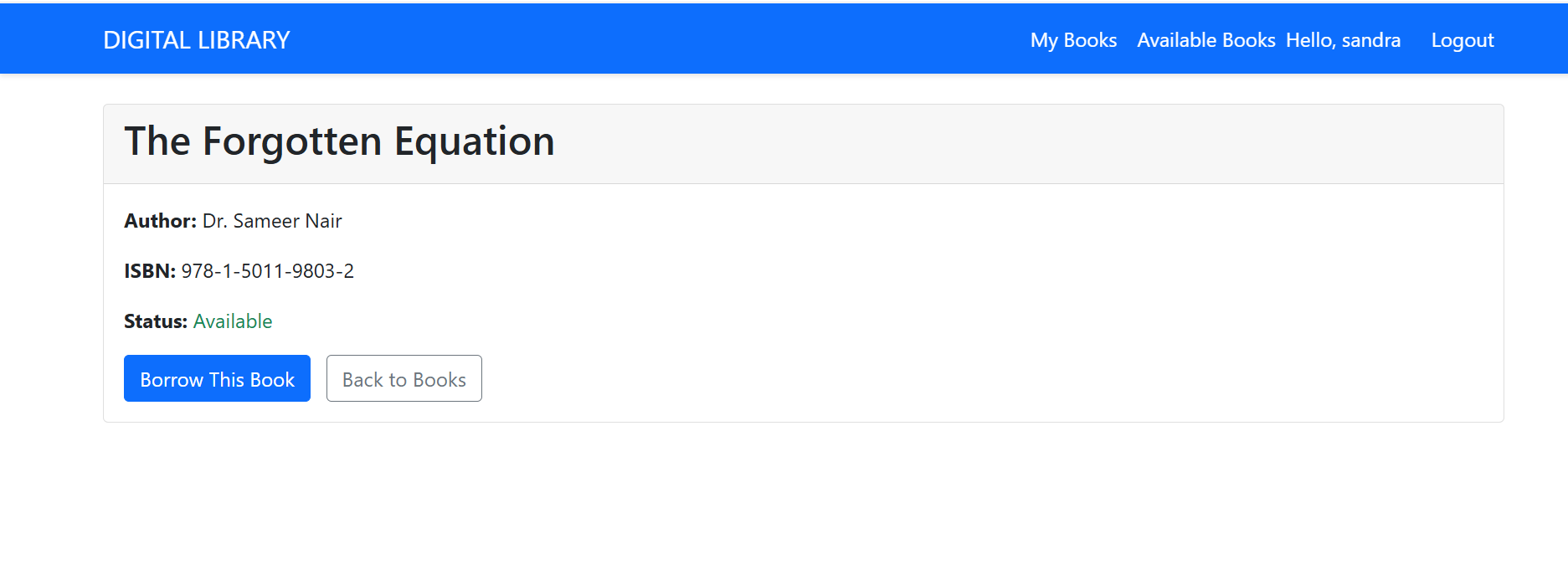
}

%>

</div>

<jsp:include page="footer.jsp" />

**5.4.3 Book Details (book\_details.jsp)**

****

<%@page import="java.sql.SQLException"%>

<%@page import="java.sql.ResultSet"%>

<%@page import="java.sql.PreparedStatement"%>

<%@page import="java.sql.Connection"%>

<%@ page import="dao.BookDAO, model.Book, model.User, DatabaseConnection.DatabaseConnection" %>

<%

if (session.getAttribute("user") == null) {

response.sendRedirect("login.jsp");

return;

}

int bookId = Integer.parseInt(request.getParameter("id"));

BookDAO bookDAO = new BookDAO();

Book book = bookDAO.getBookById(bookId);

if (book == null) {

response.sendRedirect("user\_dashboard.jsp");

return;

}

%>

<jsp:include page="header.jsp" />

<div class="container mt-4">

<div class="card">

<div class="card-header">

<h2><%= book.getTitle() %></h2>

</div>

<div class="card-body">

<p><strong>Author:</strong> <%= book.getAuthor() %></p>

<p><strong>ISBN:</strong> <%= book.getIsbn() %></p>

<p><strong>Status:</strong>

<%= book.isAvailable() ? "<span class='text-success'>Available</span>" : "<span class='text-danger'>Borrowed</span>" %>

</p>

<%

boolean canBorrow = book.isAvailable();

boolean isBorrowed = false;

String borrowStatus = "";

// Check if current user borrowed this book

if (!canBorrow) {

User currentUser = (User) session.getAttribute("user");

if (currentUser != null) {

try (Connection conn = DatabaseConnection.getConnection()) {

PreparedStatement ps = conn.prepareStatement(

"SELECT \* FROM borrow\_records WHERE user\_id = ? AND book\_id = ? AND return\_date IS NULL");

ps.setInt(1, currentUser.getId());

ps.setInt(2, book.getId());

ResultSet rs = ps.executeQuery();

if (rs.next()) {

isBorrowed = true;

borrowStatus = "You borrowed this book on " + rs.getDate("borrow\_date");

}

} catch (SQLException e) {

e.printStackTrace();

}

}

}

%>

<% if (canBorrow) { %>

<a href="BorrowBookServlet?id=<%= book.getId() %>" class="btn btn-primary">Borrow This Book</a>

<% } else if (isBorrowed) { %>

<a href="ReturnBookServlet?id=<%= book.getId() %>" class="btn btn-success"> Return This Book</a>

<p class="mt-2 text-muted"><%= borrowStatus %></p>

<% } else { %>

<button class="btn btn-secondary" disabled> Already Borrowed</button>

<% } %>

<a href="user\_dashboard.jsp" class="btn btn-outline-secondary ms-2"> Back to Books</a>

</div>

</div>

</div>

<jsp:include page="footer.jsp" />

**5.5 Utility Pages**

**5.5.1 Logout (logout.jsp)**

<%

session.invalidate();

response.sendRedirect("login.jsp");

%>

**Section 6: Testing & Deployment**

**6.1 Testing the Application**

Run the project in NetBeans

Test Login with:

Username: admin

Password: admin123

Test Registration with new user

Test Book Management (add/delete books)

Test Borrow/Return functionality

Test Admin Features (manage users, view history)

**6.2 Common Issues & Solutions**

Database Connection Errors: Verify PostgreSQL password in DatabaseConnection.java

404 Errors: Ensure servlet annotations (@WebServlet) are present

JSP Compilation Errors: Check imports and package structure

Login Failures: Trim whitespace from username/password inputs

**6.3 Deployment**

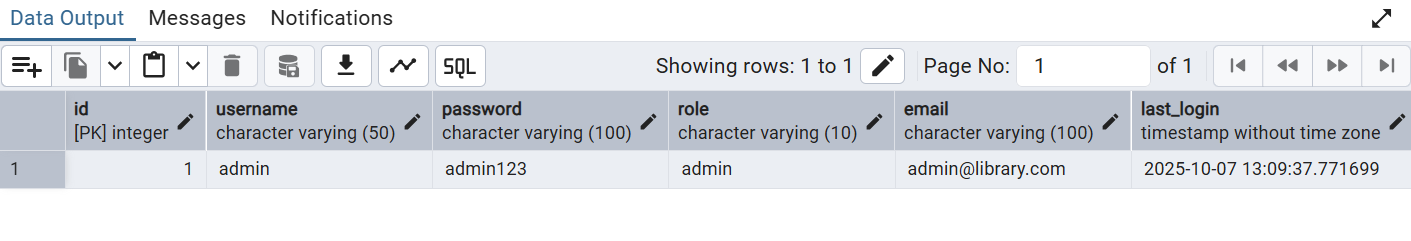
Build WAR file: Right-click project → Clean and Build

Deploy to Tomcat: Copy target/DigitalLibrary-1.0-SNAPSHOT.war to tomcat/webapps/

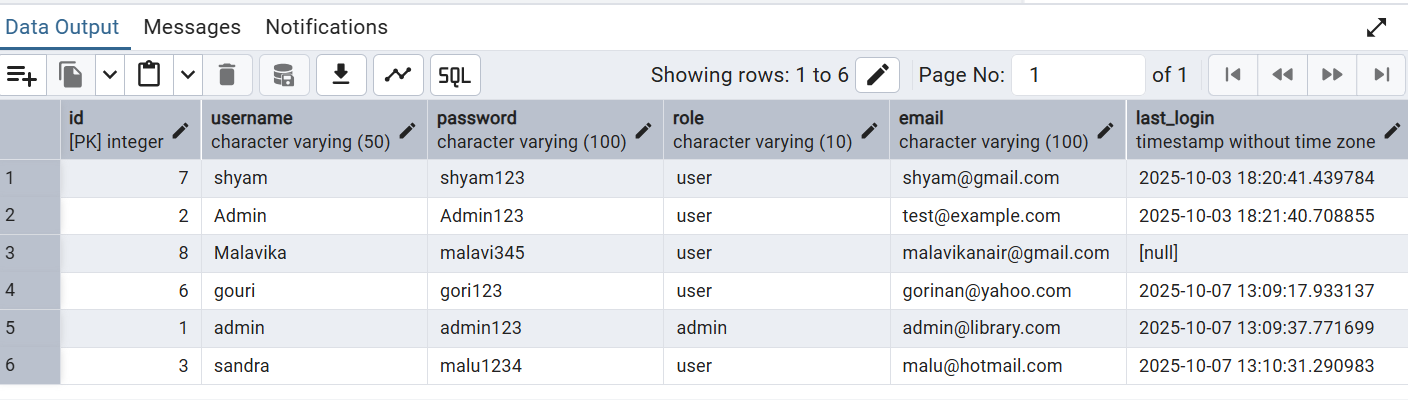
Access Application: <http://localhost:8080/DigitalLibrary-1.0-SNAPSHOT/>

**Section 7: Database in pgAdmin 4**

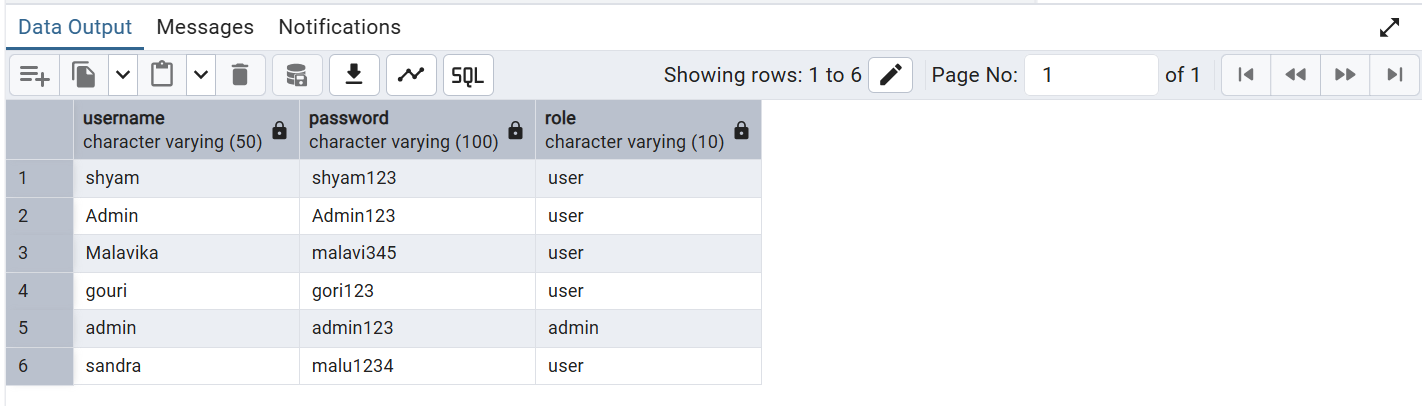
1. SELECT \* FROM users WHERE username = 'admin';

****

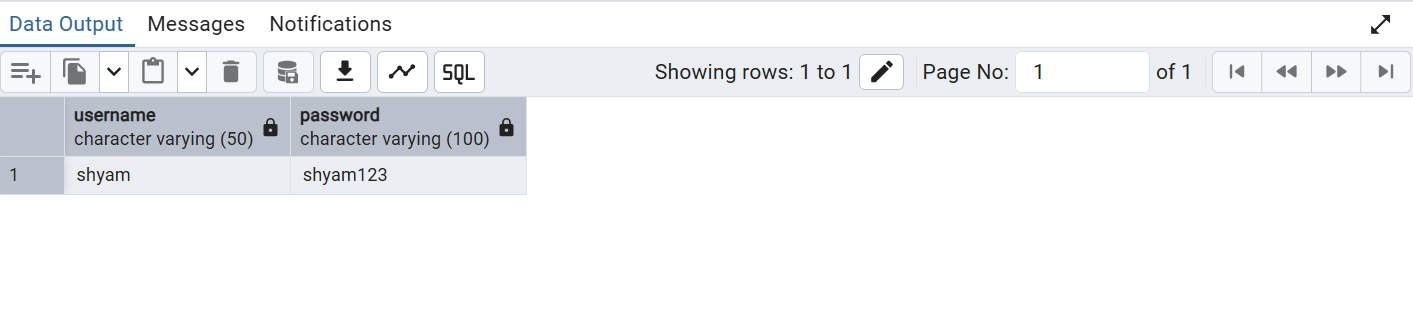
2. SELECT \* FROM users;



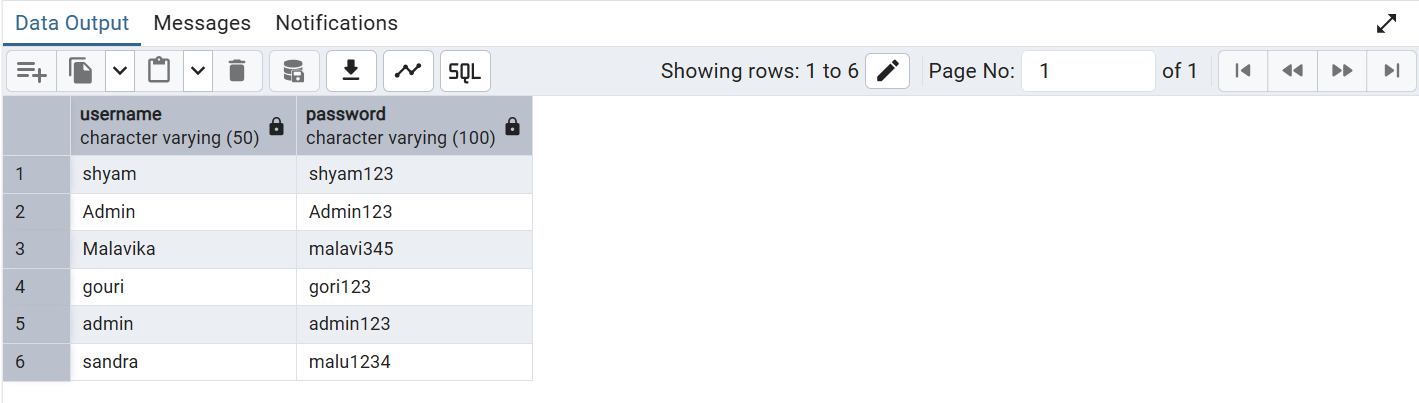
3. SELECT username, password, role FROM users;



4. SELECT username, password FROM users WHERE username = 'shyam';



5. SELECT username, password FROM users;



6. SELECT username, LENGTH(username) FROM users;

