**MAHATMA EDUCATION SOCIETY’S**

**PILLAI COLLEGE OF ARTS, COMMERCE & SCIENCE**

**(Autonomous)**

**NEW PANVEL**

PROJECT REPORT ON

**“Library / Book Store Management System”**

IN PARTIAL FULFILLMENT OF

**BACHELOR OF COMPUTER SCIENCE**

**SEMESTER V ( 2023-24 )**

PROJECT GUIDE

**PROF. SHUBHANGI PAWAR**

SUBMITTED BY

**OWAIS AHMED ALTAF SHAIKH**

ROLL NO: **8881**

**1. Introduction:**

The Library/Bookstore Management System is a comprehensive and user-friendly application designed to streamline library and bookstore operations. This system leverages a set of efficient stored procedures and functions to facilitate essential CRUD (Create, Read, Update, Delete) functionalities, ensuring the seamless management of books, authors, and customers. It enables librarians and bookstore owners to effortlessly add, edit, and organize their book collections, track customer records, and simplify the process of borrowing or purchasing books. With a focus on simplicity and functionality, this system empowers both librarians and bookstore managers to efficiently maintain and enhance their literary inventories while providing an improved experience for patrons and book enthusiasts alike.

**2. Objectives:**

Book Management: Enable Admin to add, remove and edit Books, including details such as Bookname, AuthorName and price .

Database Integration: Utilize MySQL (Pl/Sql) for storing and managing Books

User-Friendly Interface: Create a decent interface and easy to use.

Scalability: Design the system to handle growth and adapt to future enhancements.

**3. System Requirements:**

Server: GlassFish Server 4.1.1

Java: JDK (Java Development Kit) 8.

Development Tools: Netbeans IDE 8.2.

Database: MySQL V5.7(PL/SQL).

Web Technologies: JSP, HTML, CSS.

Database Connectivity: JDBC (Java Database Connectivity) driver for MySQL.

Web Browser: Chrome, Firefox, or any modern browser.

**Code:**

**DatabaseUtils.jsp**

<%@page contentType="text/html" pageEncoding="UTF-8"%>

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

<%!

static Connection getConnection() throws SQLException, ClassNotFoundException {

String url = "jdbc:mysql://localhost:3307/cadbms";

String username = "root";

String password = "root"

Class.forName("com.mysql.jdbc.Driver");

return DriverManager.getConnection(url, username, password); }%>

**View.jsp**

<%@page contentType="text/html" pageEncoding="UTF-8"%>

<%@include file="DatabaseUtils.jsp" %>

<!DOCTYPE html>

<html>

<body>

<%

Connection conn = null;

CallableStatement cstmt = null;

ResultSet rs = null;

try {

conn = getConnection();

String sql = "{CALL GetAllBooks()}";

cstmt = conn.prepareCall(sql);

rs = cstmt.executeQuery();

while (rs.next()) {

int id = rs.getInt(1);

String bookName = rs.getString(2);

String authorName = rs.getString(3);

double price = rs.getDouble(4);

out.println("<tr>");

out.println("<td>" + id + "</td>");

out.println("<td>" + bookName + "</td>");

out.println("<td>" + authorName + "</td>");

out.println("<td>" + price + "</td>");

out.println("</tr>");

}

} catch (Exception e) {

e.printStackTrace();

} finally {

try {

if (rs != null) rs.close();

if (cstmt != null) cstmt.close();

if (conn != null) conn.close();

} catch (SQLException e) {

e.printStackTrace();

}

}

%>

</tbody>

</table>

</div>

</body>

</html>

**Insert.jsp**

<%@page contentType="text/html" pageEncoding="UTF-8"%>

<%@include file="DatabaseUtils.jsp" %>

<!DOCTYPE html>

<html>

<body>

<%

if (request.getMethod().equals("POST")) {

Connection conn = null;

CallableStatement cstmt = null;

ResultSet rs = null;

try {

conn = getConnection();

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

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

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

String sql = "{CALL InsertBookWithSeq(?, ?, ?)}"; // Assuming you have a stored procedure for insertion with a sequence-generated ID

cstmt = conn.prepareCall(sql);

cstmt.setString(1, bookname);

cstmt.setString(2, authorname);

cstmt.setInt(3, price);

cstmt.executeUpdate();

// Get the last inserted ID using the same connection

rs = cstmt.executeQuery("SELECT LAST\_INSERT\_ID() AS inserted\_id");

if (rs.next()) {

int lastInsertedId = rs.getInt("inserted\_id");

if (lastInsertedId > 0) {

out.println("<p class=\"text-success\">Insertion successful</p>");

} else {

out.println("<p class=\"text-danger\">Insertion failed</p>");

}

}

} catch (Exception e) {

e.printStackTrace();

out.println("<p class=\"text-danger\">An error occurred during insertion</p>");

} finally {

try {

if (rs != null) rs.close();

if (cstmt != null) cstmt.close();

if (conn != null) conn.close();

} catch (SQLException e) {

e.printStackTrace();

}

}

}

%>

</div>

</div>

</body>

</html>

**Update.jsp**

<%@page contentType="text/html" pageEncoding="UTF-8"%>

<%@include file="DatabaseUtils.jsp" %>

<!DOCTYPE html>

<html>

<body>

<%

if (request.getMethod().equals("POST")) {

Connection conn = null;

CallableStatement cstmt = null;

try {

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

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

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

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

conn = getConnection();

String sql = "{CALL UpdateBook(?, ?, ?, ?)}";

cstmt = conn.prepareCall(sql);

cstmt.setString(1, id);

cstmt.setString(2, newBookName);

cstmt.setString(3, newAuthorName);

cstmt.setInt(4, newPrice);

int rowsUpdated = cstmt.executeUpdate();

if (rowsUpdated > 0) {

out.println("<p class=\"text-success\">Update successful</p>");

} else {

// Check if update failed due to invalid ID

sql = "SELECT COUNT(\*) FROM books WHERE id = ?";

cstmt = conn.prepareCall(sql);

cstmt.setString(1, id);

ResultSet rs = cstmt.executeQuery();

rs.next();

int rowCount = rs.getInt(1);

if (rowCount == 0) {

out.println("<p class=\"text-danger\">Update failed: Invalid ID</p>");

} else {

out.println("<p class=\"text-danger\">Update failed</p>");

}

}

} catch (Exception e) {

e.printStackTrace();

out.println("<p class=\"text-danger\">An error occurred during update</p>");

} finally {

try {

if (cstmt != null) cstmt.close();

if (conn != null) conn.close();

} catch (SQLException e) {

e.printStackTrace();

}

}

}

%>

</div>

</div>

</body>

</html>

**Delete.jsp**

<%@page contentType="text/html" pageEncoding="UTF-8"%>

<%@include file="DatabaseUtils.jsp" %>

<!DOCTYPE html>

<html>

<body>

<%

if (request.getMethod().equals("POST")) {

Connection conn = null;

PreparedStatement pstmt = null;

try {

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

conn = getConnection();

String sql = "SELECT DeleteBookByID(?) AS result";

pstmt = conn.prepareStatement(sql);

pstmt.setString(1, id);

ResultSet rs = pstmt.executeQuery();

rs.next();

boolean deleteResult = rs.getBoolean("result");

if (deleteResult) {

out.println("<p class=\"text-success\">Delete successful</p>");

} else {

out.println("<p class=\"text-danger\">No row deleted. Make sure the ID exists.</p>");

}

} catch (Exception e) {

e.printStackTrace();

out.println("<p class=\"text-danger\">An error occurred during delete</p>");

} finally {

try {

if (pstmt != null) pstmt.close();

if (conn != null) conn.close();

} catch (SQLException e) {

e.printStackTrace();

}

}

}

%>

</div>

</div>

</body>

</html>

**5)Procedures and fucnctions using plsql in mysql**

**InsertBookwithseq**

DELIMITER //

CREATE PROCEDURE InsertBookWithSeq(IN bookName VARCHAR(255), IN authorName VARCHAR(255), IN bookPrice INT)

BEGIN

INSERT INTO book (bookname, authorname, price) VALUES (bookName, authorName, bookPrice);

SELECT LAST\_INSERT\_ID() AS inserted\_id;

END;

//

DELIMITER ;

**UpdateBook**

DELIMITER //

CREATE PROCEDURE UpdateBook(

IN book\_id INT,

IN new\_book\_name VARCHAR(255),

IN new\_author\_name VARCHAR(255),

IN new\_price DECIMAL(10, 2)

)

BEGIN

UPDATE books

SET bookname = new\_book\_name,

authorname = new\_author\_name,

price = new\_price

WHERE id = book\_id;

END;

//

DELIMITER ;

**DeleteBookById**

DELIMITER //

CREATE FUNCTION DeleteBookByID(book\_id INT)

RETURNS BOOLEAN

BEGIN

DECLARE deleted\_rows INT;

DELETE FROM books WHERE id = book\_id;

SET deleted\_rows = ROW\_COUNT();

IF deleted\_rows > 0 THEN

RETURN TRUE;

ELSE

RETURN FALSE;

END IF;

END;

//

DELIMITER ;

**GetAllBooks**

DELIMITER //

CREATE PROCEDURE GetAllBooks()

BEGIN

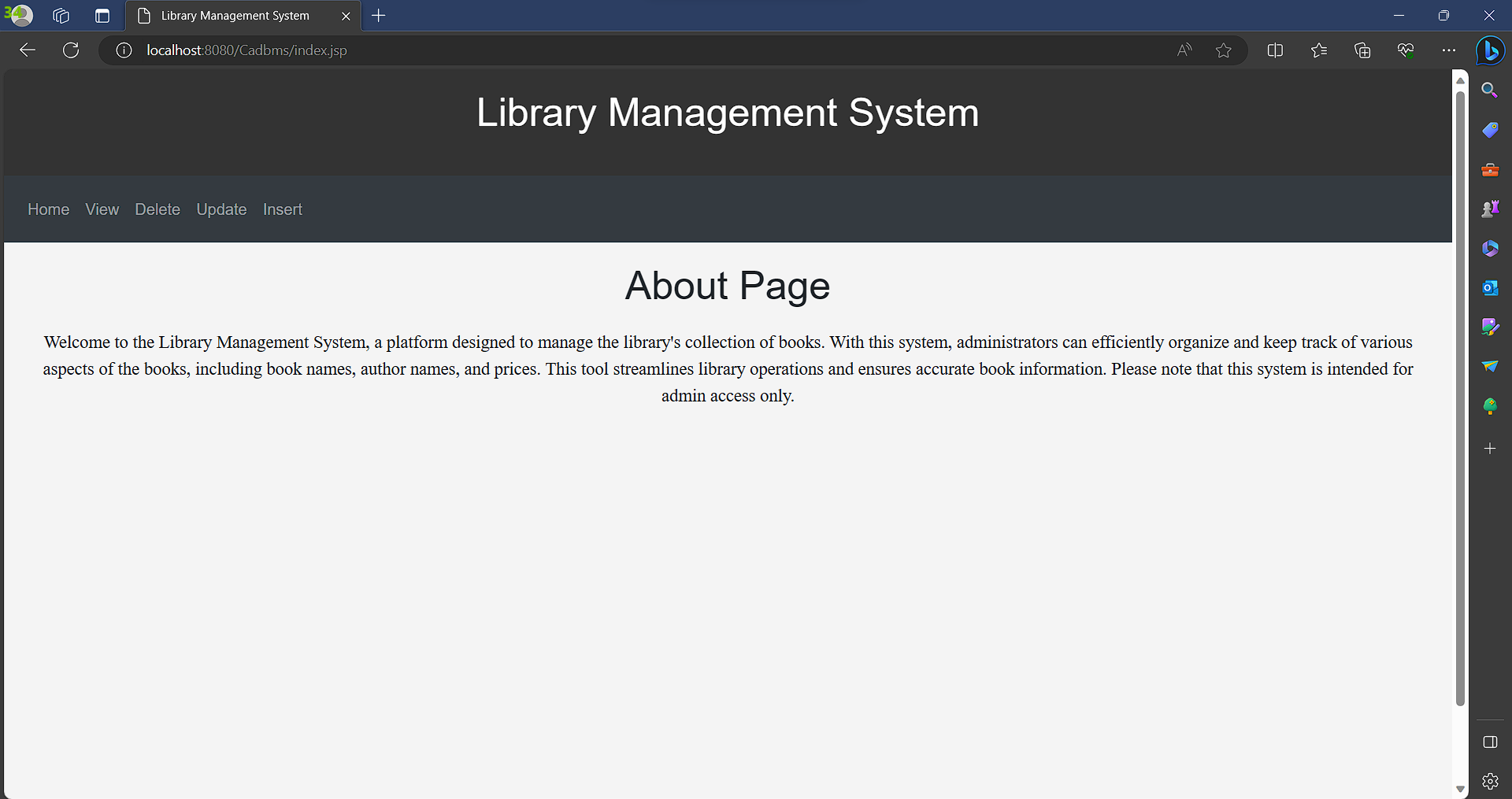
SELECT \* FROM books;

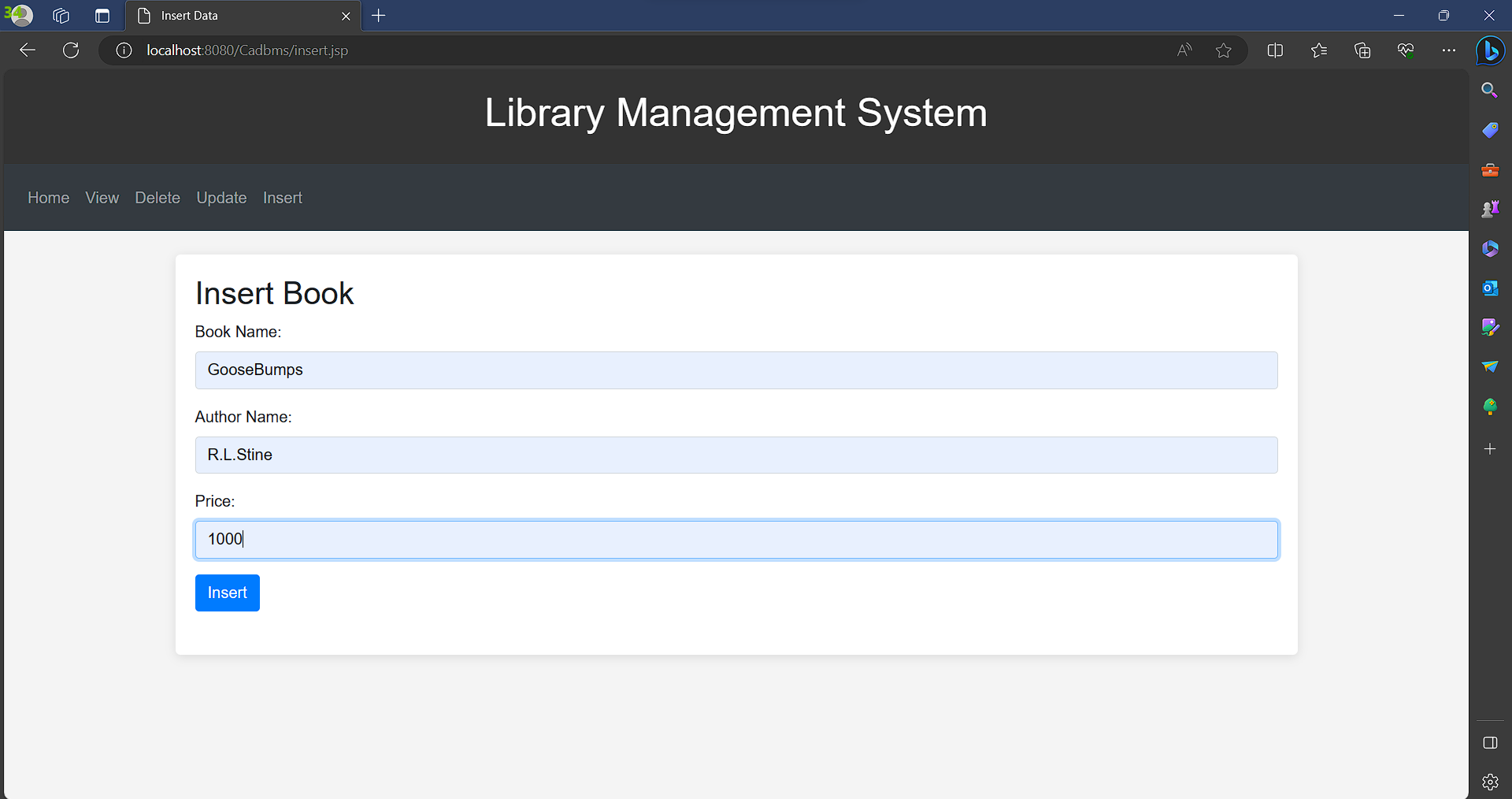
END;

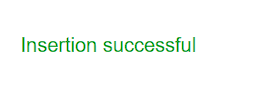
//

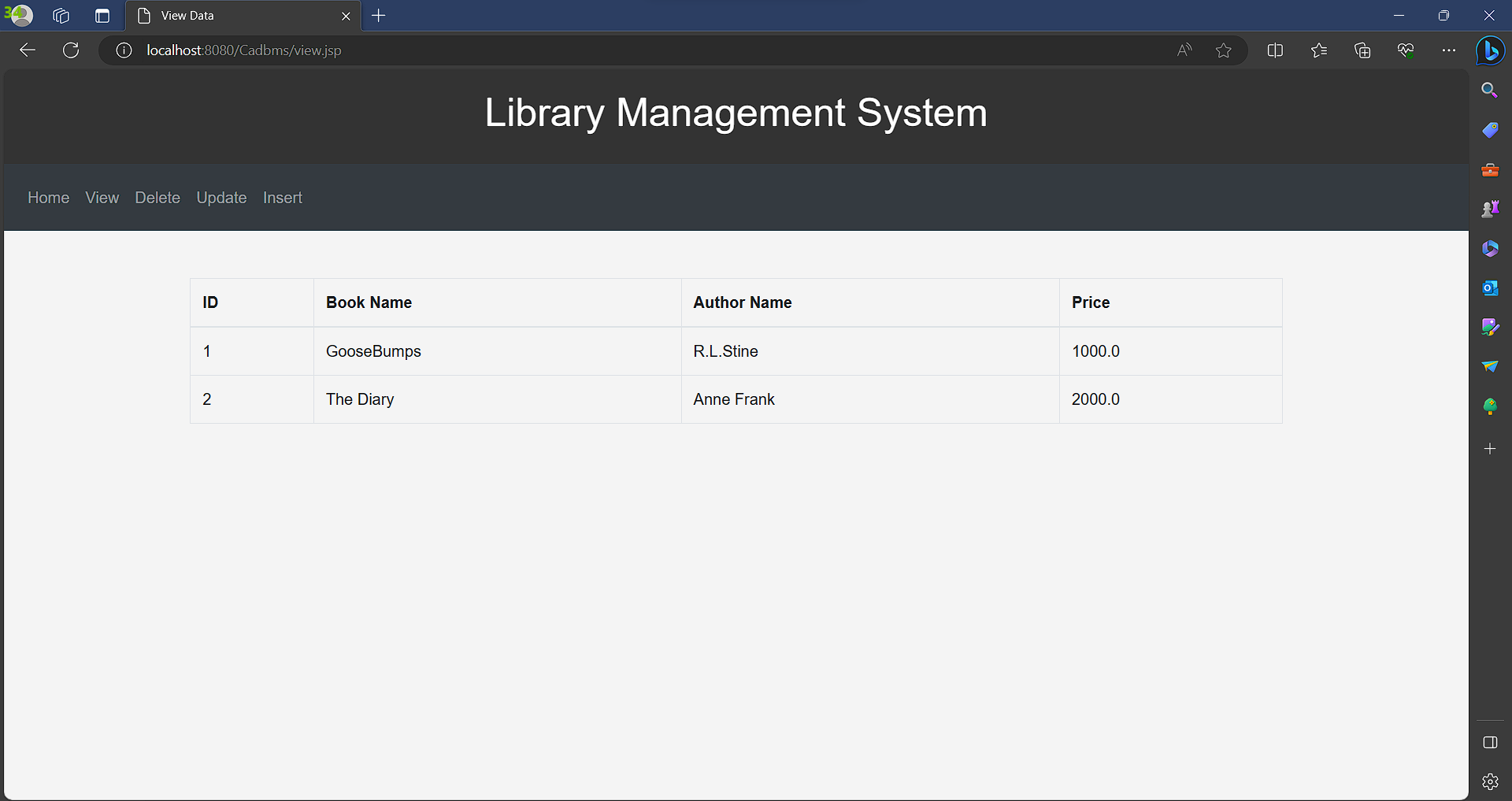
DELIMITER ;

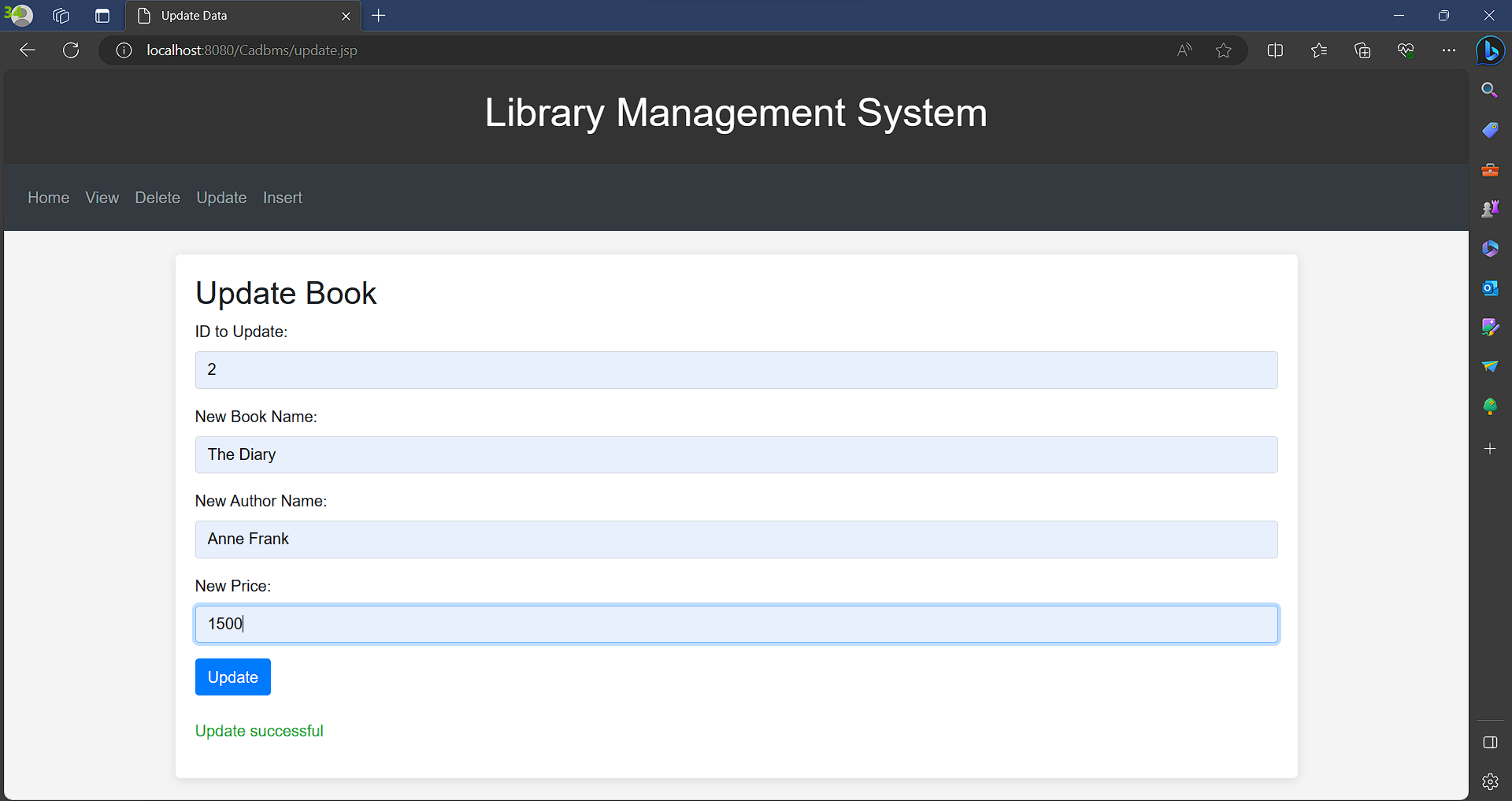
**4. Code with Output Overview:**

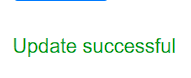
****

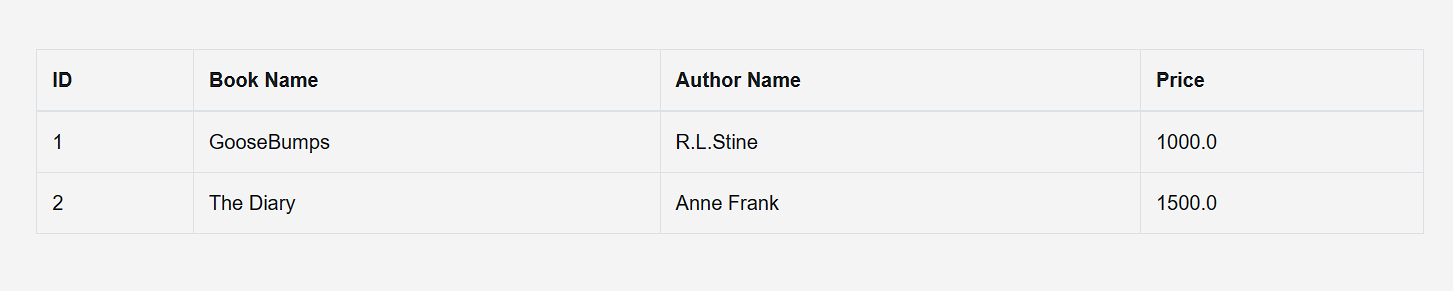
****

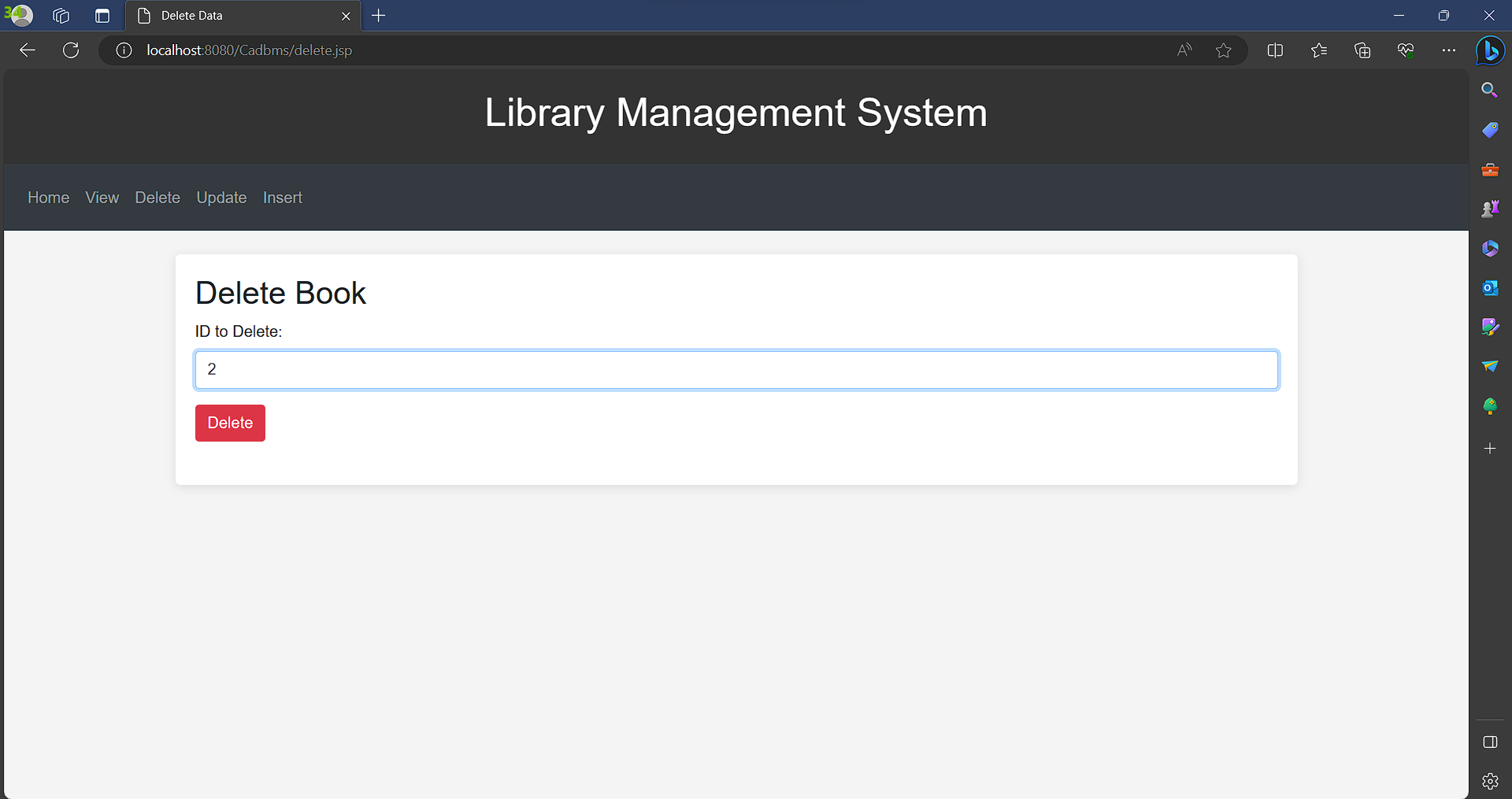
****

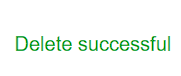
****

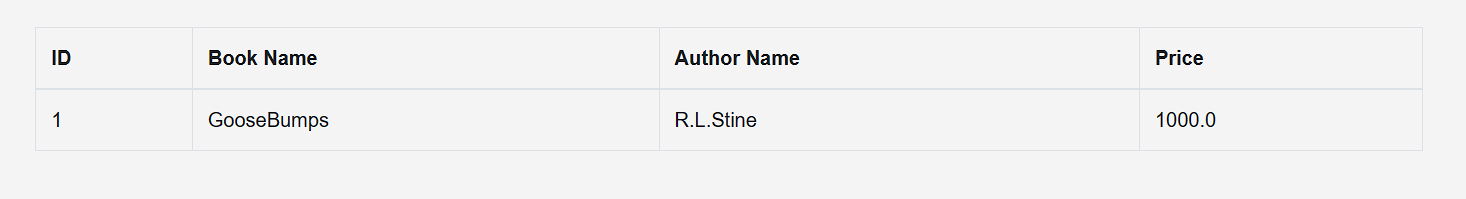
****

****

****

****

****

****