Week 13. Java Database Connectivity & Servlet API and JDBC

# Library Management System: Book Checkout and Return

**Objective:** Build a basic Library Management System where users can checkout and return books.

**Setup**: Create a database named **LibraryDB** with tables **Books** (columns: book\_id, title, author, is\_available) and **Users** (columns: user\_id, name, email).

# Tasks:

* 1. Write a JDBC program that allows users to view all available books.
  2. Implement a **checkoutBook()** method that takes a **user\_id** and **book\_id** and updates the **is\_available** column in the **Books** table.
  3. Implement a returnBook() method that updates is\_available back to true. **Challenges**: Handle cases where a book is not available for checkout or the user tries to return a book they haven't checked out.

CREATE DATABASE LibraryDB;

USE LibraryDB;

CREATE TABLE Users (

user\_id INT PRIMARY KEY,

name VARCHAR(100),

email VARCHAR(100)

);

CREATE TABLE Books (

book\_id INT PRIMARY KEY,

title VARCHAR(100),

author VARCHAR(100),

is\_available BOOLEAN DEFAULT TRUE

);

import java.sql.\*;

public class LibraryManagementSystem {

static final String DB\_URL = "jdbc:mysql://localhost:3306/LibraryDB";

static final String USER = "root";

static final String PASS = "your\_password";

public static void main(String[] args) {

try (Connection conn = DriverManager.getConnection(DB\_URL, USER, PASS)) {

System.out.println("Connected to the database.");

viewAvailableBooks(conn);

checkoutBook(conn, 1, 101);

returnBook(conn, 1, 101);

} catch (SQLException e) {

e.printStackTrace();

}

}

public static void viewAvailableBooks(Connection conn) throws SQLException {

String query = "SELECT \* FROM Books WHERE is\_available = TRUE";

try (Statement stmt = conn.createStatement();

ResultSet rs = stmt.executeQuery(query)) {

System.out.println("Available Books:");

while (rs.next()) {

System.out.printf("ID: %d | Title: %s | Author: %s%n",

rs.getInt("book\_id"),

rs.getString("title"),

rs.getString("author"));

}

}

}

public static void checkoutBook(Connection conn, int userId, int bookId)

throws SQLException {

String checkQuery = "SELECT is\_available FROM Books WHERE book\_id = ?";

String updateQuery = "UPDATE Books SET is\_available = FALSE WHERE book\_id = ?";

try (PreparedStatement checkStmt = conn.prepareStatement(checkQuery)) {

checkStmt.setInt(1, bookId);

ResultSet rs = checkStmt.executeQuery();

if (rs.next()) {

boolean isAvailable = rs.getBoolean("is\_available");

if (isAvailable) {

try (PreparedStatement updateStmt = conn.prepareStatement(updateQuery)) {

updateStmt.setInt(1, bookId);

updateStmt.executeUpdate();

System.out.println("Book " + bookId + " checked out by User " + userId);

}

} else {

System.out.println("Book " + bookId + " is not available.");

}

} else {

System.out.println("Book with ID " + bookId + " does not exist.");

}

}

}

public static void returnBook(Connection conn, int userId, int bookId)

throws SQLException {

String checkQuery = "SELECT is\_available FROM Books WHERE book\_id = ?";

String updateQuery = "UPDATE Books SET is\_available = TRUE WHERE book\_id = ?";

try (PreparedStatement checkStmt = conn.prepareStatement(checkQuery)) {

checkStmt.setInt(1, bookId);

ResultSet rs = checkStmt.executeQuery();

if (rs.next()) {

boolean isAvailable = rs.getBoolean("is\_available");

if (!isAvailable) {

try (PreparedStatement updateStmt = conn.prepareStatement(updateQuery)) {

updateStmt.setInt(1, bookId);

updateStmt.executeUpdate();

System.out.println("Book " + bookId + " returned by User " + userId);

}

} else {

System.out.println("Book " + bookId + " is already available (not checked out).");

}

} else {

System.out.println("Book with ID " + bookId + " does not exist.");

}

}

}

}

# Employee Database Management System

**Objective:** Create a console-based Java application to manage employee records in an SQLite or MySQL database.

# Tasks:

* Establish a connection to the database.
* Implement CRUD operations for employee records (e.g., **addEmployee**, **viewEmployee**, **updateEmployee**, and **deleteEmployee**).
* Use PreparedStatement to avoid SQL injection.
* Add basic error handling to capture potential SQL exceptions.

**Expected Outcome:** Practice managing database connections, performing CRUD operations, and handling exceptions.

CREATE DATABASE EmployeeDB;

USE EmployeeDB;

CREATE TABLE Employees (

id INT PRIMARY KEY AUTO\_INCREMENT,

name VARCHAR(100),

department VARCHAR(100),

email VARCHAR(100)

);

import java.sql.\*;

import java.util.Scanner;

public class EmployeeManagementSystem {

static final String DB\_URL = "jdbc:mysql://localhost:3306/EmployeeDB";

static final String USER = "root";

static final String PASS = "your\_password";

public static void main(String[] args) {

try (Connection conn = DriverManager.getConnection(DB\_URL, USER, PASS);

Scanner scanner = new Scanner(System.in)) {

System.out.println("Connected to EmployeeDB");

while (true) {

System.out.println("\n1. Add Employee\n2. View Employee\n3. Update Employee\n4. Delete Employee\n5. Exit");

System.out.print("Enter your choice: ");

int choice = scanner.nextInt();

scanner.nextLine();

switch (choice) {

case 1 -> addEmployee(conn, scanner);

case 2 -> viewEmployee(conn, scanner);

case 3 -> updateEmployee(conn, scanner);

case 4 -> deleteEmployee(conn, scanner);

case 5 -> {

System.out.println("Exiting...");

return;

}

default -> System.out.println("Invalid choice.");

}

}

} catch (SQLException e) {

e.printStackTrace();

}

}

public static void addEmployee(Connection conn, Scanner scanner) throws SQLException {

System.out.print("Enter name: ");

String name = scanner.nextLine();

System.out.print("Enter department: ");

String dept = scanner.nextLine();

System.out.print("Enter email: ");

String email = scanner.nextLine();

String query = "INSERT INTO Employees (name, department, email) VALUES (?, ?, ?)";

try (PreparedStatement stmt = conn.prepareStatement(query)) {

stmt.setString(1, name);

stmt.setString(2, dept);

stmt.setString(3, email);

stmt.executeUpdate();

System.out.println("Employee added successfully.");

}

}

public static void viewEmployee(Connection conn, Scanner scanner) throws SQLException {

System.out.print("Enter Employee ID to view: ");

int id = scanner.nextInt();

String query = "SELECT \* FROM Employees WHERE id = ?";

try (PreparedStatement stmt = conn.prepareStatement(query)) {

stmt.setInt(1, id);

ResultSet rs = stmt.executeQuery();

if (rs.next()) {

System.out.println("ID: " + rs.getInt("id"));

System.out.println("Name: " + rs.getString("name"));

System.out.println("Department: " + rs.getString("department"));

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

} else {

System.out.println("Employee not found.");

}

}

}

public static void updateEmployee(Connection conn, Scanner scanner) throws SQLException {

System.out.print("Enter Employee ID to update: ");

int id = scanner.nextInt();

scanner.nextLine();

System.out.print("Enter new name: ");

String name = scanner.nextLine();

System.out.print("Enter new department: ");

String dept = scanner.nextLine();

System.out.print("Enter new email: ");

String email = scanner.nextLine();

String query = "UPDATE Employees SET name = ?, department = ?, email = ? WHERE id = ?";

try (PreparedStatement stmt = conn.prepareStatement(query)) {

stmt.setString(1, name);

stmt.setString(2, dept);

stmt.setString(3, email);

stmt.setInt(4, id);

int rowsUpdated = stmt.executeUpdate();

if (rowsUpdated > 0) {

System.out.println("Employee updated successfully.");

} else {

System.out.println("Employee not found.");

}

}

}

public static void deleteEmployee(Connection conn, Scanner scanner) throws SQLException {

System.out.print("Enter Employee ID to delete: ");

int id = scanner.nextInt();

String query = "DELETE FROM Employees WHERE id = ?";

try (PreparedStatement stmt = conn.prepareStatement(query)) {

stmt.setInt(1, id);

int rowsDeleted = stmt.executeUpdate();

if (rowsDeleted > 0) {

System.out.println("Employee deleted successfully.");

} else {

System.out.println("Employee not found.");

}

}

}

}

# Building a Mini Student Database with CRUD Operations

**Objective**: Develop a mini database application to manage student records, with Create, Read, Update, and Delete (CRUD) functionality

# Requirements:

* Create a **Student** table in a relational database (e.g., MySQL, PostgreSQL) with columns for **student\_id**, **name**, **age**, and **grade**.
* Implement a Java class, **StudentDAO**, using JDBC to:
  1. Insert new student records.
  2. Retrieve all student records or by **student\_id**.
  3. Update existing student records by **student\_id**.
  4. Delete a student record by **student\_id**.
* Use **PreparedStatement** for all SQL queries to prevent SQL injection.

CREATE DATABASE StudentDB;

USE StudentDB;

CREATE TABLE Students (

student\_id INT PRIMARY KEY AUTO\_INCREMENT,

name VARCHAR(100),

age INT,

grade VARCHAR(10)

);

import java.sql.\*;

import java.util.ArrayList;

import java.util.List;

public class StudentDAO {

static final String DB\_URL = "jdbc:mysql://localhost:3306/StudentDB";

static final String USER = "root";

static final String PASS = "your\_password";

public void insertStudent(String name, int age, String grade) {

String sql = "INSERT INTO Students (name, age, grade) VALUES (?, ?, ?)";

try (Connection conn = DriverManager.getConnection(DB\_URL, USER, PASS);

PreparedStatement stmt = conn.prepareStatement(sql)) {

stmt.setString(1, name);

stmt.setInt(2, age);

stmt.setString(3, grade);

stmt.executeUpdate();

System.out.println("Student inserted successfully.");

} catch (SQLException e) {

e.printStackTrace();

}

}

public void getAllStudents() {

String sql = "SELECT \* FROM Students";

try (Connection conn = DriverManager.getConnection(DB\_URL, USER, PASS);

PreparedStatement stmt = conn.prepareStatement(sql);

ResultSet rs = stmt.executeQuery()) {

while (rs.next()) {

System.out.printf("ID: %d | Name: %s | Age: %d | Grade: %s%n",

rs.getInt("student\_id"),

rs.getString("name"),

rs.getInt("age"),

rs.getString("grade"));

}

} catch (SQLException e) {

e.printStackTrace();

}

}

public void getStudentById(int id) {

String sql = "SELECT \* FROM Students WHERE student\_id = ?";

try (Connection conn = DriverManager.getConnection(DB\_URL, USER, PASS);

PreparedStatement stmt = conn.prepareStatement(sql)) {

stmt.setInt(1, id);

ResultSet rs = stmt.executeQuery();

if (rs.next()) {

System.out.printf("ID: %d | Name: %s | Age: %d | Grade: %s%n",

rs.getInt("student\_id"),

rs.getString("name"),

rs.getInt("age"),

rs.getString("grade"));

} else {

System.out.println("Student not found.");

}

} catch (SQLException e) {

e.printStackTrace();

}

}

public void updateStudent(int id, String name, int age, String grade) {

String sql = "UPDATE Students SET name = ?, age = ?, grade = ? WHERE student\_id = ?";

try (Connection conn = DriverManager.getConnection(DB\_URL, USER, PASS);

PreparedStatement stmt = conn.prepareStatement(sql)) {

stmt.setString(1, name);

stmt.setInt(2, age);

stmt.setString(3, grade);

stmt.setInt(4, id);

int rows = stmt.executeUpdate();

if (rows > 0) {

System.out.println("Student updated successfully.");

} else {

System.out.println("Student not found.");

}

} catch (SQLException e) {

e.printStackTrace();

}

}

public void deleteStudent(int id) {

String sql = "DELETE FROM Students WHERE student\_id = ?";

try (Connection conn = DriverManager.getConnection(DB\_URL, USER, PASS);

PreparedStatement stmt = conn.prepareStatement(sql)) {

stmt.setInt(1, id);

int rows = stmt.executeUpdate();

if (rows > 0) {

System.out.println("Student deleted successfully.");

} else {

System.out.println("Student not found.");

}

} catch (SQLException e) {

e.printStackTrace();

}

}

}

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

StudentDAO dao = new StudentDAO();

Scanner scanner = new Scanner(System.in);

while (true) {

System.out.println("\n1. Insert Student\n2. View All Students\n3. View Student by ID\n4. Update Student\n5. Delete Student\n6. Exit");

System.out.print("Enter your choice: ");

int choice = scanner.nextInt();

scanner.nextLine();

switch (choice) {

case 1 -> {

System.out.print("Name: ");

String name = scanner.nextLine();

System.out.print("Age: ");

int age = scanner.nextInt();

scanner.nextLine();

System.out.print("Grade: ");

String grade = scanner.nextLine();

dao.insertStudent(name, age, grade);

}

case 2 -> dao.getAllStudents();

case 3 -> {

System.out.print("Enter Student ID: ");

int id = scanner.nextInt();

dao.getStudentById(id);

}

case 4 -> {

System.out.print("Enter Student ID to update: ");

int id = scanner.nextInt();

scanner.nextLine();

System.out.print("New Name: ");

String name = scanner.nextLine();

System.out.print("New Age: ");

int age = scanner.nextInt();

scanner.nextLine();

System.out.print("New Grade: ");

String grade = scanner.nextLine();

dao.updateStudent(id, name, age, grade);

}

case 5 -> {

System.out.print("Enter Student ID to delete: ");

int id = scanner.nextInt();

dao.deleteStudent(id);

}

case 6 -> {

System.out.println("Exiting...");

return;

}

default -> System.out.println("Invalid choice.");

}

}

}

}

# User Registration and Login System

**Objective:** Develop a basic user management system where users can register, log in, and view their profile information.

# Requirements:

* Create a MySQL database with a **users** table containing fields like **user\_id**, **username**, **password**, **email**, and **created\_at**.
* Use a registration servlet to handle new user sign-ups. Validate data before storing it in the database using JDBC.
* Implement a login servlet to authenticate users based on their username and password.
* Redirect successful logins to a profile servlet, which displays the user’s information retrieved from the database.

CREATE DATABASE UserDB;

USE UserDB;

CREATE TABLE users (

user\_id INT AUTO\_INCREMENT PRIMARY KEY,

username VARCHAR(50) UNIQUE NOT NULL,

password VARCHAR(100) NOT NULL,

email VARCHAR(100),

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

import java.sql.\*;

public class DBUtil {

private static final String URL = "jdbc:mysql://localhost:3306/UserDB";

private static final String USER = "root";

private static final String PASS = "your\_password";

public static Connection getConnection() throws SQLException {

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

}

}

import java.io.\*;

import java.sql.\*;

import javax.servlet.\*;

import javax.servlet.http.\*;

public class RegisterServlet extends HttpServlet {

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

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

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

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

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

String query = "INSERT INTO users (username, password, email) VALUES (?, ?, ?)";

PreparedStatement stmt = conn.prepareStatement(query);

stmt.setString(1, username);

stmt.setString(2, password);

stmt.setString(3, email);

int result = stmt.executeUpdate();

if (result > 0) {

res.sendRedirect("login.html");

} else {

res.sendRedirect("register.html");

}

} catch (SQLException e) {

e.printStackTrace();

res.sendRedirect("register.html");

}

}

}

import java.io.\*;

import java.sql.\*;

import javax.servlet.\*;

import javax.servlet.http.\*;

public class LoginServlet extends HttpServlet {

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

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

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

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

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

PreparedStatement stmt = conn.prepareStatement(query);

stmt.setString(1, username);

stmt.setString(2, password);

ResultSet rs = stmt.executeQuery();

if (rs.next()) {

HttpSession session = req.getSession();

session.setAttribute("user\_id", rs.getInt("user\_id"));

res.sendRedirect("profile");

} else {

res.sendRedirect("login.html");

}

} catch (SQLException e) {

e.printStackTrace();

res.sendRedirect("login.html");

}

}

}

import java.io.\*;

import java.sql.\*;

import javax.servlet.\*;

import javax.servlet.http.\*;

public class ProfileServlet extends HttpServlet {

protected void doGet(HttpServletRequest req, HttpServletResponse res) throws ServletException, IOException {

HttpSession session = req.getSession(false);

if (session == null || session.getAttribute("user\_id") == null) {

res.sendRedirect("login.html");

return;

}

int userId = (int) session.getAttribute("user\_id");

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

String query = "SELECT \* FROM users WHERE user\_id = ?";

PreparedStatement stmt = conn.prepareStatement(query);

stmt.setInt(1, userId);

ResultSet rs = stmt.executeQuery();

PrintWriter out = res.getWriter();

res.setContentType("text/html");

if (rs.next()) {

out.println("<h2>Welcome, " + rs.getString("username") + "</h2>");

out.println("<p>Email: " + rs.getString("email") + "</p>");

out.println("<p>Joined on: " + rs.getString("created\_at") + "</p>");

} else {

out.println("<p>User not found.</p>");

}

} catch (SQLException e) {

e.printStackTrace();

}

}

}

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

<input type="text" name="username" placeholder="Username" required><br>

<input type="password" name="password" placeholder="Password" required><br>

<input type="email" name="email" placeholder="Email"><br>

<button type="submit">Register</button>

</form>

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

<input type="text" name="username" placeholder="Username" required><br>

<input type="password" name="password" placeholder="Password" required><br>

<button type="submit">Login</button>

</form>

<web-app>

<servlet>

<servlet-name>RegisterServlet</servlet-name>

<servlet-class>RegisterServlet</servlet-class>

</servlet>

<servlet-mapping>

<servlet-name>RegisterServlet</servlet-name>

<url-pattern>/register</url-pattern>

</servlet-mapping>

<servlet>

<servlet-name>LoginServlet</servlet-name>

<servlet-class>LoginServlet</servlet-class>

</servlet>

<servlet-mapping>

<servlet-name>LoginServlet</servlet-name>

<url-pattern>/login</url-pattern>

</servlet-mapping>

<servlet>

<servlet-name>ProfileServlet</servlet-name>

<servlet-class>ProfileServlet</servlet-class>

</servlet>

<servlet-mapping>

<servlet-name>ProfileServlet</servlet-name>

<url-pattern>/profile</url-pattern>

</servlet-mapping>

</web-app>

# Product Management System with Search Functionality

**Objective:** Build a product catalog management system that allows users to add, update, delete, and search for products.

# Requirements:

* Set up a **products** table in MySQL with fields like **product\_id**, **product\_name**, **description**, **price**, and **category**.
* Develop servlets for adding, updating, and deleting products, using JDBC for database interactions.
* Implement a search servlet that allows users to filter products by name, category, or price range.
* Display search results dynamically on a JSP page.

**Key Concepts Covered:** Data filtering, servlet-JDBC CRUD operations, JSP integration, MVC pattern.

CREATE DATABASE ProductDB;

USE ProductDB;

CREATE TABLE products (

product\_id INT AUTO\_INCREMENT PRIMARY KEY,

product\_name VARCHAR(100),

description TEXT,

price DECIMAL(10,2),

category VARCHAR(50)

);

import java.sql.\*;

public class DBUtil {

private static final String URL = "jdbc:mysql://localhost:3306/ProductDB";

private static final String USER = "root";

private static final String PASS = "your\_password";

public static Connection getConnection() throws SQLException {

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

}

}

import javax.servlet.\*;

import javax.servlet.http.\*;

import java.io.\*;

import java.sql.\*;

public class AddProductServlet extends HttpServlet {

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

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

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

double price = Double.parseDouble(req.getParameter("price"));

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

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

String query = "INSERT INTO products (product\_name, description, price, category) VALUES (?, ?, ?, ?)";

PreparedStatement stmt = conn.prepareStatement(query);

stmt.setString(1, name);

stmt.setString(2, description);

stmt.setDouble(3, price);

stmt.setString(4, category);

stmt.executeUpdate();

res.sendRedirect("products.jsp");

} catch (SQLException e) {

e.printStackTrace();

}

}

}

import javax.servlet.\*;

import javax.servlet.http.\*;

import java.io.\*;

import java.sql.\*;

public class UpdateProductServlet extends HttpServlet {

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

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

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

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

double price = Double.parseDouble(req.getParameter("price"));

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

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

String query = "UPDATE products SET product\_name = ?, description = ?, price = ?, category = ? WHERE product\_id = ?";

PreparedStatement stmt = conn.prepareStatement(query);

stmt.setString(1, name);

stmt.setString(2, description);

stmt.setDouble(3, price);

stmt.setString(4, category);

stmt.setInt(5, id);

stmt.executeUpdate();

res.sendRedirect("products.jsp");

} catch (SQLException e) {

e.printStackTrace();

}

}

}

import javax.servlet.\*;

import javax.servlet.http.\*;

import java.io.\*;

import java.sql.\*;

public class DeleteProductServlet extends HttpServlet {

protected void doGet(HttpServletRequest req, HttpServletResponse res) throws ServletException, IOException {

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

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

String query = "DELETE FROM products WHERE product\_id = ?";

PreparedStatement stmt = conn.prepareStatement(query);

stmt.setInt(1, id);

stmt.executeUpdate();

res.sendRedirect("products.jsp");

} catch (SQLException e) {

e.printStackTrace();

}

}

}

import javax.servlet.\*;

import javax.servlet.http.\*;

import java.io.\*;

import java.sql.\*;

import java.util.\*;

public class SearchProductServlet extends HttpServlet {

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

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

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

String minPriceStr = req.getParameter("minPrice");

String maxPriceStr = req.getParameter("maxPrice");

List<String[]> products = new ArrayList<>();

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

String query = "SELECT \* FROM products WHERE product\_name LIKE ? AND category LIKE ? AND price BETWEEN ? AND ?";

PreparedStatement stmt = conn.prepareStatement(query);

stmt.setString(1, "%" + name + "%");

stmt.setString(2, "%" + category + "%");

stmt.setDouble(3, minPriceStr.isEmpty() ? 0 : Double.parseDouble(minPriceStr));

stmt.setDouble(4, maxPriceStr.isEmpty() ? 99999 : Double.parseDouble(maxPriceStr));

ResultSet rs = stmt.executeQuery();

while (rs.next()) {

String[] row = {

rs.getString("product\_id"),

rs.getString("product\_name"),

rs.getString("description"),

rs.getString("price"),

rs.getString("category")

};

products.add(row);

}

} catch (SQLException e) {

e.printStackTrace();

}

req.setAttribute("results", products);

RequestDispatcher rd = req.getRequestDispatcher("products.jsp");

rd.forward(req, res);

}

}

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

<html>

<body>

<h2>Search Products</h2>

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

<input type="text" name="name" placeholder="Product Name">

<input type="text" name="category" placeholder="Category">

<input type="text" name="minPrice" placeholder="Min Price">

<input type="text" name="maxPrice" placeholder="Max Price">

<input type="submit" value="Search">

</form>

<h2>Add Product</h2>

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

<input type="text" name="name" placeholder="Name" required>

<input type="text" name="description" placeholder="Description" required>

<input type="number" step="0.01" name="price" placeholder="Price" required>

<input type="text" name="category" placeholder="Category" required>

<input type="submit" value="Add">

</form>

<h2>Search Results</h2>

<table border="1">

<tr><th>ID</th><th>Name</th><th>Description</th><th>Price</th><th>Category</th><th>Action</th></tr>

<%

List<String[]> results = (List<String[]>) request.getAttribute("results");

if (results != null) {

for (String[] row : results) {

%>

<tr>

<td><%= row[0] %></td>

<td><%= row[1] %></td>

<td><%= row[2] %></td>

<td><%= row[3] %></td>

<td><%= row[4] %></td>

<td>

<form action="delete" method="get" style="display:inline;">

<input type="hidden" name="id" value="<%= row[0] %>">

<input type="submit" value="Delete">

</form>

</td>

</tr>

<%

}

}

%>

</table>

</body>

</html>

<web-app>

<servlet>

<servlet-name>AddProductServlet</servlet-name>

<servlet-class>AddProductServlet</servlet-class>

</servlet>

<servlet-mapping>

<servlet-name>AddProductServlet</servlet-name>

<url-pattern>/add</url-pattern>

</servlet-mapping>

<servlet>

<servlet-name>UpdateProductServlet</servlet-name>

<servlet-class>UpdateProductServlet</servlet-class>

</servlet>

<servlet-mapping>

<servlet-name>UpdateProductServlet</servlet-name>

<url-pattern>/update</url-pattern>

</servlet-mapping>

<servlet>

<servlet-name>DeleteProductServlet</servlet-name>

<servlet-class>DeleteProductServlet</servlet-class>

</servlet>

<servlet-mapping>

<servlet-name>DeleteProductServlet</servlet-name>

<url-pattern>/delete</url-pattern>

</servlet-mapping>

<servlet>

<servlet-name>SearchProductServlet</servlet-name>

<servlet-class>SearchProductServlet</servlet-class>

</servlet>

<servlet-mapping>

<servlet-name>SearchProductServlet</servlet-name>

<url-pattern>/search</url-pattern>

</servlet-mapping>

</web-app>