**Task 9. JSP**

Build an User Management web application that manages a collection of users with the basic feature: list, insert, update, delete (or CURD operations - Create, Update, Read and Delete) using JSP.

1. Create an Eclipse Dynamic Web Project
2. Add Dependencies
3. Project Structure
4. MySQL Database Setup
5. Create a JavaBean - User.java
6. Create a UserDAO.java
7. Create a UserServlet.java
8. Creating User Listing JSP Page - user-list.jsp
9. Create a User Form JSP Page - user-form.jsp
10. Creating Error JSP page
11. Deploying and Testing the Application Demo

**MySQL Database Setup**

CREATE DATABASE 'demo';

USE demo;

create table users (

id int(3) NOT NULL AUTO\_INCREMENT,

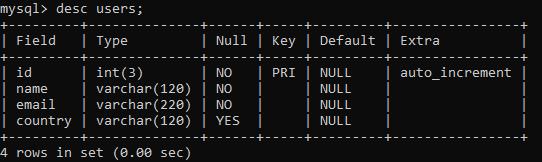
name varchar(120) NOT NULL,

email varchar(220) NOT NULL,

country varchar(120),

PRIMARY KEY (id)

);



**Create a JavaBean - User.java**

public class User {

protected int id;

protected String name;

protected String email;

protected String country;

public User() {}

public User(String name, String email, String country) {

super();

this.name = name;

this.email = email;

this.country = country;

}

public User(int id, String name, String email, String country) {

super();

this.id = id;

this.name = name;

this.email = email;

this.country = country;

}

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getEmail() {

return email;

}

public void setEmail(String email) {

this.email = email;

}

public String getCountry() {

return country;

}

public void setCountry(String country) {

this.country = country;

}

}

**Create a UserDAO.java**

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.util.ArrayList;

import java.util.List;

import net.javaguides.usermanagement.model.User;

public class UserDAO {

private String jdbcURL = "jdbc:mysql://localhost:3306/demo?useSSL=false";

private String jdbcUsername = "root";

private String jdbcPassword = "root";

private static final String INSERT\_USERS\_SQL = "INSERT INTO users" + " (name, email, country) VALUES " +

" (?, ?, ?);";

private static final String SELECT\_USER\_BY\_ID = "select id,name,email,country from users where id =?";

private static final String SELECT\_ALL\_USERS = "select \* from users";

private static final String DELETE\_USERS\_SQL = "delete from users where id = ?;";

private static final String UPDATE\_USERS\_SQL = "update users set name = ?,email= ?, country =? where id = ?;";

public UserDAO() {}

protected Connection getConnection() {

Connection connection = null;

try {

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

connection = DriverManager.getConnection(jdbcURL, jdbcUsername, jdbcPassword);

} catch (SQLException e) {

// TODO Auto-generated catch block

e.printStackTrace();

} catch (ClassNotFoundException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

return connection;

}

public void insertUser(User user) throws SQLException {

System.out.println(INSERT\_USERS\_SQL);

// try-with-resource statement will auto close the connection.

try (Connection connection = getConnection(); PreparedStatement preparedStatement = connection.prepareStatement(INSERT\_USERS\_SQL)) {

preparedStatement.setString(1, user.getName());

preparedStatement.setString(2, user.getEmail());

preparedStatement.setString(3, user.getCountry());

System.out.println(preparedStatement);

preparedStatement.executeUpdate();

} catch (SQLException e) {

printSQLException(e);

}

}

public User selectUser(int id) {

User user = null;

// Step 1: Establishing a Connection

try (Connection connection = getConnection();

// Step 2:Create a statement using connection object

PreparedStatement preparedStatement = connection.prepareStatement(SELECT\_USER\_BY\_ID);) {

preparedStatement.setInt(1, id);

System.out.println(preparedStatement);

// Step 3: Execute the query or update query

ResultSet rs = preparedStatement.executeQuery();

// Step 4: Process the ResultSet object.

while (rs.next()) {

String name = rs.getString("name");

String email = rs.getString("email");

String country = rs.getString("country");

user = new User(id, name, email, country);

}

} catch (SQLException e) {

printSQLException(e);

}

return user;

}

public List < User > selectAllUsers() {

// using try-with-resources to avoid closing resources (boiler plate code)

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

// Step 1: Establishing a Connection

try (Connection connection = getConnection();

// Step 2:Create a statement using connection object

PreparedStatement preparedStatement = connection.prepareStatement(SELECT\_ALL\_USERS);) {

System.out.println(preparedStatement);

// Step 3: Execute the query or update query

ResultSet rs = preparedStatement.executeQuery();

// Step 4: Process the ResultSet object.

while (rs.next()) {

int id = rs.getInt("id");

String name = rs.getString("name");

String email = rs.getString("email");

String country = rs.getString("country");

users.add(new User(id, name, email, country));

}

} catch (SQLException e) {

printSQLException(e);

}

return users;

}

public boolean deleteUser(int id) throws SQLException {

boolean rowDeleted;

try (Connection connection = getConnection(); PreparedStatement statement = connection.prepareStatement(DELETE\_USERS\_SQL);) {

statement.setInt(1, id);

rowDeleted = statement.executeUpdate() > 0;

}

return rowDeleted;

}

public boolean updateUser(User user) throws SQLException {

boolean rowUpdated;

try (Connection connection = getConnection(); PreparedStatement statement = connection.prepareStatement(UPDATE\_USERS\_SQL);) {

statement.setString(1, user.getName());

statement.setString(2, user.getEmail());

statement.setString(3, user.getCountry());

statement.setInt(4, user.getId());

rowUpdated = statement.executeUpdate() > 0;

}

return rowUpdated;

}

private void printSQLException(SQLException ex) {

for (Throwable e: ex) {

if (e instanceof SQLException) {

e.printStackTrace(System.err);

System.err.println("SQLState: " + ((SQLException) e).getSQLState());

System.err.println("Error Code: " + ((SQLException) e).getErrorCode());

System.err.println("Message: " + e.getMessage());

Throwable t = ex.getCause();

while (t != null) {

System.out.println("Cause: " + t);

t = t.getCause();

}

}

}

}

}

**Create a UserServlet.java**

import java.io.IOException;

import java.sql.SQLException;

import java.util.List;

import javax.servlet.RequestDispatcher;

import javax.servlet.ServletException;

import javax.servlet.annotation.WebServlet;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import net.javaguides.usermanagement.dao.UserDAO;

import net.javaguides.usermanagement.model.User;

@WebServlet("/")

public class UserServlet extends HttpServlet {

private static final long serialVersionUID = 1 L;

private UserDAO userDAO;

public void init() {

userDAO = new UserDAO();

}

protected void doPost(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

doGet(request, response);

}

protected void doGet(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

String action = request.getServletPath();

try {

switch (action) {

case "/new":

showNewForm(request, response);

break;

case "/insert":

insertUser(request, response);

break;

case "/delete":

deleteUser(request, response);

break;

case "/edit":

showEditForm(request, response);

break;

case "/update":

updateUser(request, response);

break;

default:

listUser(request, response);

break;

}

} catch (SQLException ex) {

throw new ServletException(ex);

}

}

private void listUser(HttpServletRequest request, HttpServletResponse response)

throws SQLException, IOException, ServletException {

List < User > listUser = userDAO.selectAllUsers();

request.setAttribute("listUser", listUser);

RequestDispatcher dispatcher = request.getRequestDispatcher("user-list.jsp");

dispatcher.forward(request, response);

}

private void showNewForm(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

RequestDispatcher dispatcher = request.getRequestDispatcher("user-form.jsp");

dispatcher.forward(request, response);

}

private void showEditForm(HttpServletRequest request, HttpServletResponse response)

throws SQLException, ServletException, IOException {

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

User existingUser = userDAO.selectUser(id);

RequestDispatcher dispatcher = request.getRequestDispatcher("user-form.jsp");

request.setAttribute("user", existingUser);

dispatcher.forward(request, response);

}

private void insertUser(HttpServletRequest request, HttpServletResponse response)

throws SQLException, IOException {

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

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

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

User newUser = new User(name, email, country);

userDAO.insertUser(newUser);

response.sendRedirect("list");

}

private void updateUser(HttpServletRequest request, HttpServletResponse response)

throws SQLException, IOException {

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

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

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

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

User book = new User(id, name, email, country);

userDAO.updateUser(book);

response.sendRedirect("list");

}

private void deleteUser(HttpServletRequest request, HttpServletResponse response)

throws SQLException, IOException {

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

userDAO.deleteUser(id);

response.sendRedirect("list");

}

}

**Creating User Listing JSP Page - user-list.jsp**

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

pageEncoding="UTF-8"%>

<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>

<html>

<head>

<title>User Management Application</title>

<link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css" integrity="sha384-ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T" crossorigin="anonymous">

</head>

<body>

<header>

<nav class="navbar navbar-expand-md navbar-dark" style="background-color: tomato">

<div>

<a href="https://www.javaguides.net" class="navbar-brand"> User

Management App </a>

</div>

<ul class="navbar-nav">

<li><a href="<%=request.getContextPath()%>/list" class="nav-link">Users</a></li>

</ul>

</nav>

</header>

<br>

<div class="row">

<!-- <div class="alert alert-success" \*ngIf='message'>{{message}}</div> -->

<div class="container">

<h3 class="text-center">List of Users</h3>

<hr>

<div class="container text-left">

<a href="<%=request.getContextPath()%>/new" class="btn btn-success">Add

New User</a>

</div>

<br>

<table class="table table-bordered">

<thead>

<tr>

<th>ID</th>

<th>Name</th>

<th>Email</th>

<th>Country</th>

<th>Actions</th>

</tr>

</thead>

<tbody>

<!-- for (Todo todo: todos) { -->

<c:forEach var="user" items="${listUser}">

<tr>

<td>

<c:out value="${user.id}" />

</td>

<td>

<c:out value="${user.name}" />

</td>

<td>

<c:out value="${user.email}" />

</td>

<td>

<c:out value="${user.country}" />

</td>

<td><a href="edit?id=<c:out value='${user.id}' />">Edit</a> &nbsp;&nbsp;&nbsp;&nbsp; <a href="delete?id=<c:out value='${user.id}' />">Delete</a></td>

</tr>

</c:forEach>

<!-- } -->

</tbody>

</table>

</div>

</div>

</body>

</html>

**Create a User Form JSP Page - user-form.jsp**

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

pageEncoding="UTF-8"%>

<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>

<html>

<head>

<title>User Management Application</title>

<link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css" integrity="sha384-ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T" crossorigin="anonymous">

</head>

<body>

<header>

<nav class="navbar navbar-expand-md navbar-dark" style="background-color: tomato">

<div>

<a href="https://www.javaguides.net" class="navbar-brand"> User Management App </a>

</div>

<ul class="navbar-nav">

<li><a href="<%=request.getContextPath()%>/list" class="nav-link">Users</a></li>

</ul>

</nav>

</header>

<br>

<div class="container col-md-5">

<div class="card">

<div class="card-body">

<c:if test="${user != null}">

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

</c:if>

<c:if test="${user == null}">

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

</c:if>

<caption>

<h2>

<c:if test="${user != null}">

Edit User

</c:if>

<c:if test="${user == null}">

Add New User

</c:if>

</h2>

</caption>

<c:if test="${user != null}">

<input type="hidden" name="id" value="<c:out value='${user.id}' />" />

</c:if>

<fieldset class="form-group">

<label>User Name</label> <input type="text" value="<c:out value='${user.name}' />" class="form-control" name="name" required="required">

</fieldset>

<fieldset class="form-group">

<label>User Email</label> <input type="text" value="<c:out value='${user.email}' />" class="form-control" name="email">

</fieldset>

<fieldset class="form-group">

<label>User Country</label> <input type="text" value="<c:out value='${user.country}' />" class="form-control" name="country">

</fieldset>

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

</form>

</div>

</div>

</div>

</body>

</html>

**Creating Error JSP page**

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

pageEncoding="UTF-8" isErrorPage="true" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"

"http://www.w3.org/TR/html4/loose.dtd">

<html>

<head>

<title>Error</title>

</head>

<body>

<center>

<h1>Error</h1>

<h2><%=exception.getMessage() %><br/> </h2>

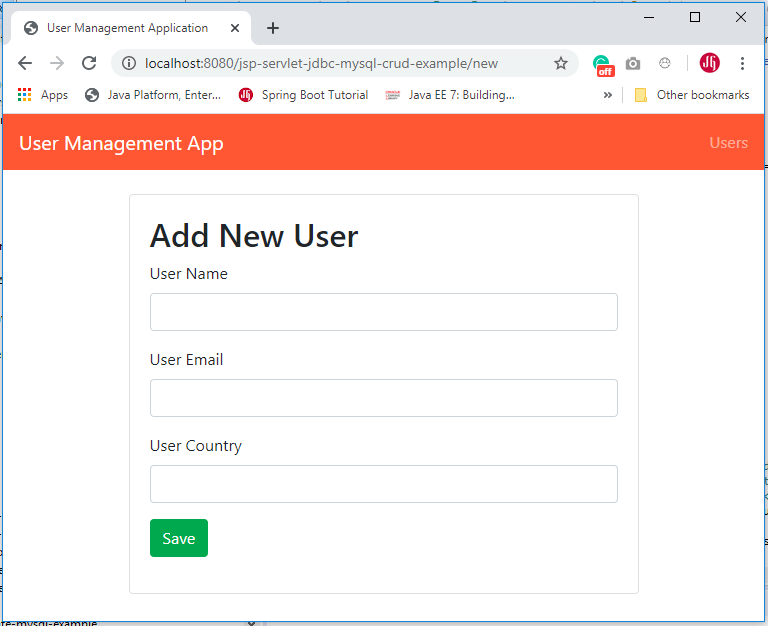
</center>

</body>

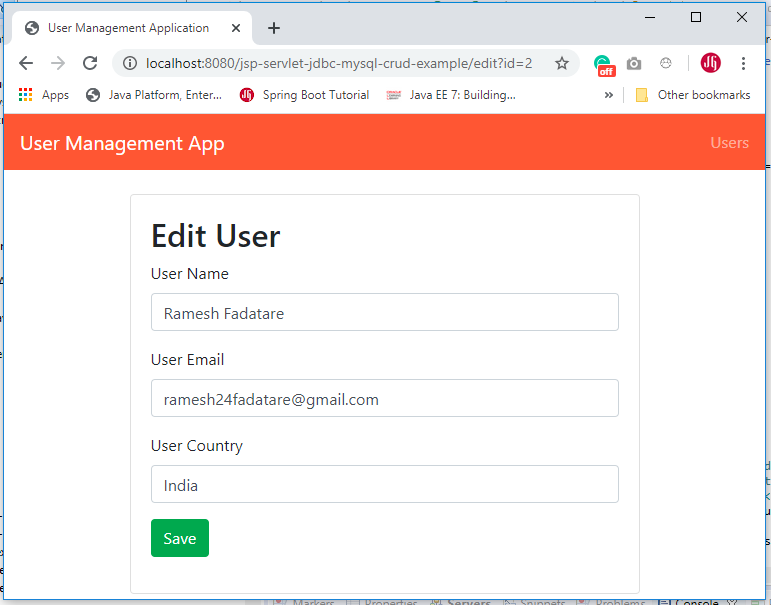
</html>

Output:

Create New User



Edit User



List of all Users

