



UNIVERSITI MALAYSIA TERENGGANU

CSM3023 – WEB PROGRAMMING 2

BACHELOR OF COMPUTER SCIENCE (MOBILE COMPUTING) WITH HONORS

LAB 8 – An MVC Example with Servlets and JSP

SEMESTER II 2023/2024

Prepared for:

DR MOHAMAD NOR BIN HASSAN

Prepared by:

MUHAMMAD ARIF HAIKAL BIN SALLEHUDDIN

(S66355)

Link Github :

<https://github.com/arifhaikal2001/CSM3023-LAB-8.git>

Task 1 - Creating MVC Database Web Application in JSP and Servlets – for Create, Read, Update, Delete.

Code :

index.jsp

```
<%--
  Document   : index
  Created on : 15 Jun 2024, 11:58:46 am
  Author      : ARIF HAIKAL
--%>

<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <title>User Management Application</title>
    <link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css">
  </head>
  <body>
    <h1>Application MVC system for Employee Management</h1><br>

    <ul>
      <li><a href="http://localhost:8080/Employee_Management/list">All Employee
List</a></li>
      <li><a href="http://localhost:8080/Employee_Management/new">Add a New
Employee</a></li>
      <li><a href="http://localhost:8080/Employee_Management/list">Edit
Employee</a></li>
    </ul>
  </body>
</html>
```

EmployeeForm.jsp

```
<%--
  Document   : EmployeeForm
  Created on : 15 Jun 2024, 11:58:20 am
  Author    : ARIF HAIKAL
--%>

<%@page contentType="text/html" pageEncoding="UTF-8"%>
<%@taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<!DOCTYPE html>
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <title>Employee Management Application</title>
    <link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css">

    <script>
      function updatePosition() {
        var selectedPosition = document.getElementById("position").value;
        document.getElementById("displayPosition").value = selectedPosition;
      }
    </script>
  </head>
  <body>
    <header>
      <nav class="navbar navbar-expand-md navbar-dark" style="background-color:
tomato">
        <div>
          <a href="" class="navbar-brand"> Employee Management App </a>
        </div>

        <ul class="navbar-nav">
          <li><a href="<%=request.getContextPath()%>/list" class="nav-
link">Employees</a></li>
        </ul>
      </nav>
    </header>
    <br>
    <div class="container col-md-5">
      <div class="card">
        <div class="card-body">
          <c:if test="${employee != null}">
            <form action="update" method="post">
          </c:if>
          <c:if test="${employee == null}">
            <form action="insert" method="post">
          </c:if>

          <h2>
            <c:if test="${employee != null}">
              Edit Employee
            </c:if>
            <c:if test="${employee == null}">
```

```

        Add New Employee
    </c:if>
</h2>

<c:if test="{employee != null}">
    <input type="hidden" name="id" value="<c:out value='{employee.id}' />" />
</c:if>

    <fieldset class="form-group">
        <label>Employee Name</label><input type="text" value="<c:out
value='{employee.name}' />"
                                class="form-control" name="name"
required="required">
    </fieldset>

    <fieldset class="form-group">
        <label>Employee Email</label><input type="text" value="<c:out
value='{employee.email}' />"
                                class="form-control" name="email">
    </fieldset>

    <fieldset class="form-group">
        <label>Employee Position</label>
        <input type="text" id="displayPosition" value="<c:out
value='{employee.position}' />" class="form-control" readonly>
        <input list="positionList" id="position" class="form-control" name="position"
onchange="updatePosition()" >
        <datalist id="positionList">
            <option value="Manager">
            <option value="Head of Dept">
            <option value="Supervisor">
            <option value="Director">
        </datalist>
    </fieldset>

    <button type="submit" class="btn btn-success">Save</button>
</form>
</div>
</div>
</body>
</html>

```

EmployeeList.jsp

```
<%--
  Document   : EmployeeList
  Created on : 15 Jun 2024, 11:58:28 am
  Author    : ARIF HAIKAL
--%>

<%@page contentType="text/html" pageEncoding="UTF-8"%>
<%@taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<!DOCTYPE html>
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <title>Employee Management System</title>
    <link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css">
  </head>
  <body>

    <header>
      <nav class="navbar navbar-expand-md navbar-dark" style="background-color:
tomato">
        <div>
          <a href="" class="navbar-brand"> Employee Management App </a>
        </div>

        <ul class="navbar-nav">
          <li><a href="<%=request.getContextPath()%>/list" class="nav-
link">Employees</a></li>
        </ul>
      </nav>
    </header>
    <br>

    <div class="row">
      <div class="container">
        <h3 class="container">List of Employees</h3>
        <hr>
        <div class="container text-left">
          <a href="<%=request.getContextPath()%>/new" class="btn btn-
success">Add New Employee</a>
        </div>
        <br>
        <table class="table table-bordered">
          <thead>
            <tr>
              <th>ID</th>
              <th>Name</th>
              <th>Email</th>
              <th>Position</th>
              <th>Actions</th>
            </tr>
          </thead>
          <tbody>
```


EmployeeDAO.java

```
package com.DAO;

import java.sql.*;
import java.sql.SQLException;
import java.util.*;
import com.Model.Employee;

/**
 *
 * @author ARIF HAIKAL
 */
public class EmployeeDAO {

    Connection connection = null;
    private String jdbcURL = "jdbc:mysql://localhost:3306/EmployeeManagement";
    private String jdbcUsername = "root";
    private String jdbcPassword = "admin@123";

    private static final String INSERT_EMPLOYEES_SQL = "INSERT INTO
employees(name, email, position) VALUES (?, ?, ?);";
    private static final String SELECT_EMPLOYEE_BY_ID = "select id,name,email,position
from employees where id=?";
    private static final String SELECT_ALL_EMPLOYEES = "select * from employees";
    private static final String DELETE_EMPLOYEES_SQL = "delete from employees where
id = ?;";
    private static final String UPDATE_EMPLOYEES_SQL = "update employees set name
= ?,email= ?, position= ? where id = ?;";

    public EmployeeDAO(){

        protected Connection getConnection(){
            Connection connection = null;
            try{
                Class.forName("com.mysql.jdbc.Driver");
                connection = DriverManager.getConnection(jdbcURL, jdbcUsername,
jdbcPassword);
                System.out.println("Database connected!");
            }catch(SQLException e){
                e.printStackTrace();
            }catch(ClassNotFoundException e){
                e.printStackTrace();
            }
            return connection;
        }

        public void insertEmployee(Employee employee) throws SQLException{
            System.out.println(INSERT_EMPLOYEES_SQL);
            try(Connection connection = getConnection(); PreparedStatement
preparedStatement =
                connection.prepareStatement(INSERT_EMPLOYEES_SQL)){
                preparedStatement.setString(1, employee.getName());
                preparedStatement.setString(2, employee.getEmail());
                preparedStatement.setString(3, employee.getPosition());
            }
        }
    }
}
```

```

        System.out.println(preparedStatement);
        preparedStatement.executeUpdate();
    }catch(SQLException e){
        printSQLException(e);
    }
}

public Employee selectEmployee(int id){
    Employee employee = null;
    // Step 1: Establishing a Connection
    try(Connection connection = getConnection());
        // Step 2: Create a statement using connection
        PreparedStatement preparedStatement =
connection.prepareStatement(SELECT_EMPLOYEE_BY_ID);{
        preparedStatement.setInt(1, id);
        System.out.println(preparedStatement);
        ResultSet rs = preparedStatement.executeQuery();

        while(rs.next()){
            String name = rs.getString("name");
            String email = rs.getString("email");
            String position = rs.getString("position");
            employee = new Employee(id, name, email, position);
        }
    }catch (SQLException e){
        printSQLException(e);
    }
    return employee;
}

public List < Employee > selectAllEmployees(){
    List < Employee > employees = new ArrayList < > ();
    try (Connection connection = getConnection());

        PreparedStatement preparedStatement =
connection.prepareStatement(SELECT_ALL_EMPLOYEES);{
        System.out.println(preparedStatement);
        ResultSet rs = preparedStatement.executeQuery();

        while(rs.next()){
            int id = rs.getInt("id");
            String name = rs.getString("name");
            String email = rs.getString("email");
            String position = rs.getString("position");
            employees.add(new Employee(id, name, email, position));
        }
    }catch(SQLException e){
        printSQLException(e);
    }
    return employees;
}

public boolean deleteEmployee(int id) throws SQLException{
    boolean rowDeleted;
    try(Connection connection = getConnection(); PreparedStatement statement =

```



```

        connection.prepareStatement(DELETE_EMPLOYEES_SQL);{
        statement.setInt(1, id);
        rowDeleted = statement.executeUpdate() > 0;
    }
    return rowDeleted;
}

public boolean updateEmployee(Employee employee) throws SQLException{
    boolean rowUpdated;
    try(Connection connection = getConnection(); PreparedStatement statement =
        connection.prepareStatement(UPDATE_EMPLOYEES_SQL);){
        statement.setString(1, employee.getName());
        statement.setString(2, employee.getEmail());
        statement.setString(3, employee.getPosition());
        statement.setInt(4, employee.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();
            }
        }
    }
}
}
}
}
}
}
}
}

```

Employee.java

```
package com.Model;

/**
 *
 * @author ARIF HAIKAL
 */
public class Employee {

    protected int id;
    protected String name;
    protected String email;
    protected String position;

    public Employee() {}

    public Employee(String name, String email, String position) {
        super();
        this.name = name;
        this.email = email;
        this.position = position;
    }

    public Employee(int id, String name, String email, String position) {
        this.id = id;
        this.name = name;
        this.email = email;
        this.position = position;
    }

    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 getPosition() {  
    return position;  
}  
  
public void setPosition(String position) {  
    this.position = position;  
}  
}
```

EmployeeServlet.java

```
package com.WEB;

import jakarta.servlet.RequestDispatcher;
import java.io.IOException;
import java.io.PrintWriter;
import jakarta.servlet.ServletException;
import jakarta.servlet.annotation.WebServlet;
import jakarta.servlet.http.HttpServlet;
import jakarta.servlet.http.HttpServletRequest;
import jakarta.servlet.http.HttpServletResponse;
import java.sql.SQLException;
import java.util.List;

import com.DAO.EmployeeDAO;
import com.Model.Employee;

/**
 *
 * @author ARIF HAIKAL
 */

@WebServlet("/")
public class EmployeeServlet extends HttpServlet {

    /**
     * Processes requests for both HTTP GET and POST
     * methods.
     *
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error occurs
     * @throws IOException if an I/O error occurs
     */
    private EmployeeDAO employeeDAO;

    @Override
    public void init(){
        employeeDAO = new EmployeeDAO();
    }

    // <editor-fold defaultstate="collapsed" desc="HttpServlet methods. Click on the + sign
    on the left to edit the code.">
    /**
     * Handles the HTTP GET method.
     *
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error occurs
     * @throws IOException if an I/O error occurs
     */
    @Override
    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":
            insertEmployee(request, response);
            break;
        case "/delete":
            deleteEmployee(request, response);
            break;
        case "/edit":
            showEditForm(request, response);
            break;
        case "/update":
            updateEmployee(request, response);
            break;
        default:
            listEmployee(request, response);
            break;
    }
} catch(SQLException ex){
    throw new ServletException(ex);
}

}

private void listEmployee(HttpServletRequest request, HttpServletResponse response)
    throws SQLException, IOException, ServletException{
    List < Employee > listEmployee = employeeDAO.selectAllEmployees();
    request.setAttribute("listEmployee", listEmployee);
    RequestDispatcher dispatcher = request.getRequestDispatcher("EmployeeList.jsp");
    dispatcher.forward(request, response);
}

private void showNewForm(HttpServletRequest request, HttpServletResponse
response)
    throws ServletException, IOException{
    RequestDispatcher dispatcher =
request.getRequestDispatcher("EmployeeForm.jsp");
    dispatcher.forward(request, response);
}

private void showEditForm(HttpServletRequest request, HttpServletResponse
response)
    throws SQLException, ServletException, IOException{
    int id = Integer.parseInt(request.getParameter("id"));
    Employee existingEmployee = employeeDAO.selectEmployee(id);
    RequestDispatcher dispatcher =
request.getRequestDispatcher("EmployeeForm.jsp");
    request.setAttribute("employee", existingEmployee);
    dispatcher.forward(request, response);
}

```

```

private void insertEmployee(HttpServletRequest request, HttpServletResponse
response)
    throws SQLException, IOException{
    String name = request.getParameter("name");
    String email = request.getParameter("email");
    String position = request.getParameter("position");
    Employee newEmployee = new Employee(name, email, position);
    employeeDAO.insertEmployee(newEmployee);
    response.sendRedirect("list");
}

private void updateEmployee(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 position = request.getParameter("position");

    Employee employee = new Employee(id, name, email, position);
    employeeDAO.updateEmployee(employee);
    response.sendRedirect("list");
}

private void deleteEmployee(HttpServletRequest request, HttpServletResponse
response)
    throws SQLException, IOException{
    int id = Integer.parseInt(request.getParameter("id"));
    employeeDAO.deleteEmployee(id);
    response.sendRedirect("list");
}

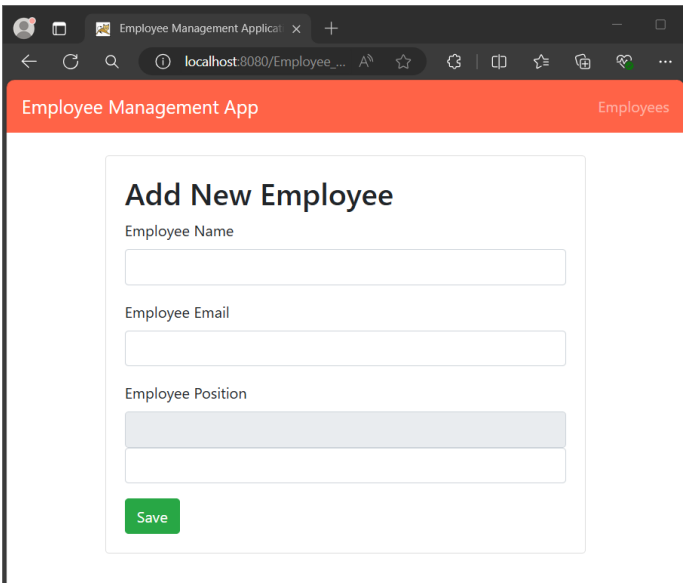
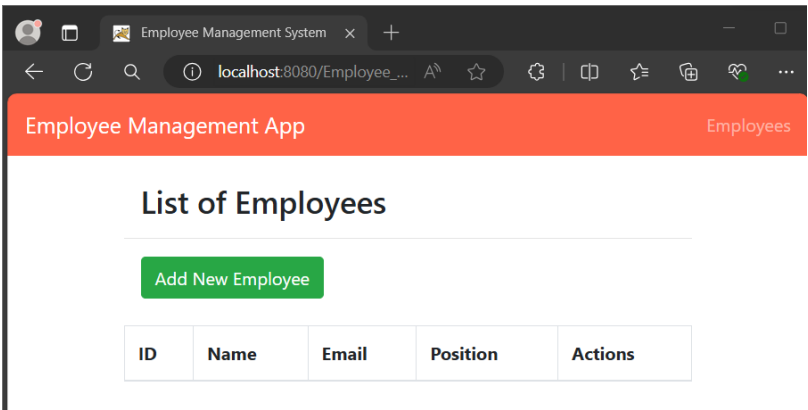
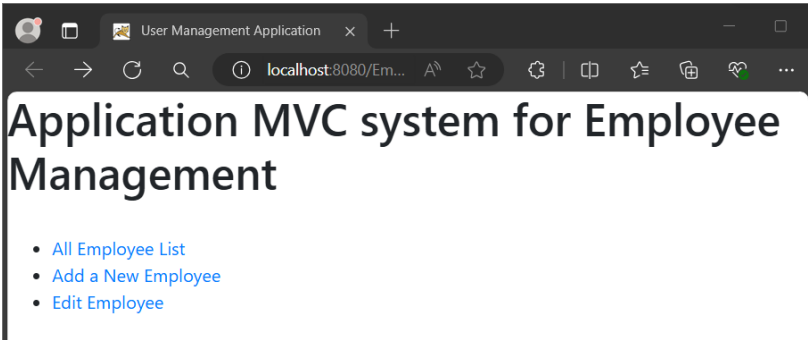
/**
 * Handles the HTTP <code>POST</code> method.
 *
 * @param request servlet request
 * @param response servlet response
 * @throws ServletException if a servlet-specific error occurs
 * @throws IOException if an I/O error occurs
 */
@Override
protected void doPost(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
    doGet(request, response);
}

/**
 * Returns a short description of the servlet.
 *
 * @return a String containing servlet description
 */
@Override
public String getServletInfo() {
    return "Short description";
}

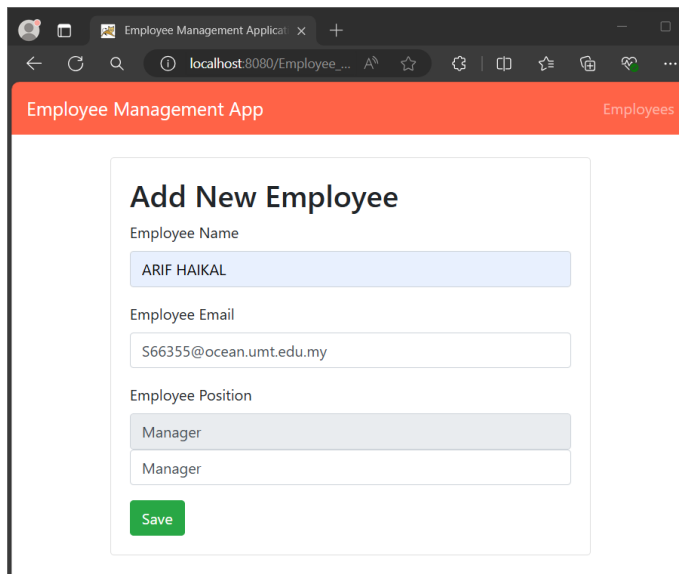
```

```
}// </editor-fold>
}
```

Output :



Add new Employee :



The screenshot shows a web browser window with the title 'Employee Management App'. The address bar shows 'localhost:8080/Employee_...'. The page has a red header bar with 'Employee Management App' on the left and 'Employees' on the right. The main content area is titled 'Add New Employee'. It contains four input fields: 'Employee Name' with the value 'ARIF HAIKAL', 'Employee Email' with the value 'S66355@ocean.umd.edu.my', and 'Employee Position' with the value 'Manager'. Below the input fields is a green 'Save' button.

Employee Name

ARIF HAIKAL

Employee Email

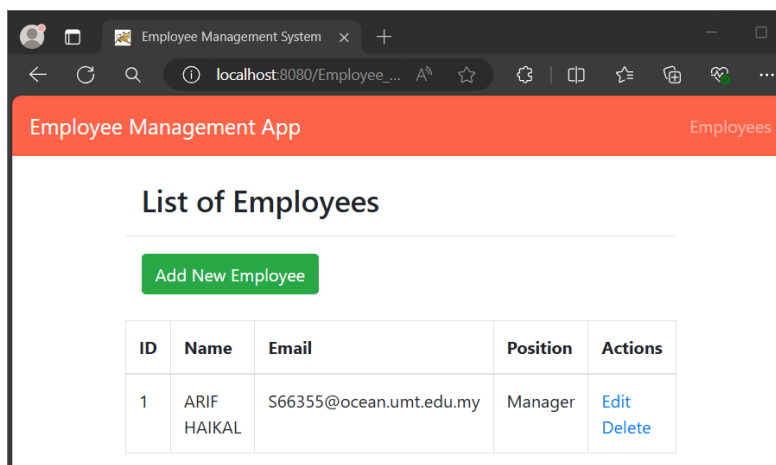
S66355@ocean.umd.edu.my

Employee Position

Manager

Manager

Save



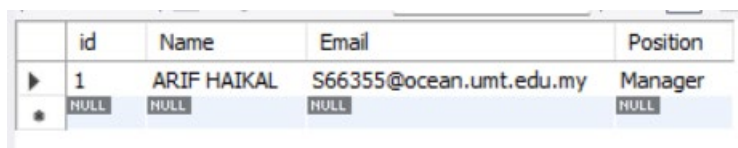
The screenshot shows a web browser window with the title 'Employee Management System'. The address bar shows 'localhost:8080/Employee_...'. The page has a red header bar with 'Employee Management App' on the left and 'Employees' on the right. The main content area is titled 'List of Employees'. Below the title is a green 'Add New Employee' button. Below the button is a table with 5 columns: ID, Name, Email, Position, and Actions. The table contains one row with the following data: ID: 1, Name: ARIF HAIKAL, Email: S66355@ocean.umd.edu.my, Position: Manager, and Actions: Edit, Delete.

List of Employees

Add New Employee

ID	Name	Email	Position	Actions
1	ARIF HAIKAL	S66355@ocean.umd.edu.my	Manager	Edit Delete

Data in Workbench :



The screenshot shows a table with 5 columns: id, Name, Email, and Position. The table contains one row with the following data: id: 1, Name: ARIF HAIKAL, Email: S66355@ocean.umd.edu.my, and Position: Manager. Below the table is a row with the values NULL, NULL, NULL, and NULL.

	id	Name	Email	Position
	1	ARIF HAIKAL	S66355@ocean.umd.edu.my	Manager
	NULL	NULL	NULL	NULL

Exercise

Code :

index.jsp

```
<%--
  Document   : index
  Created on : 15 Jun 2024, 1:23:47 pm
  Author      : ARIF HAIKAL
--%>
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <title>Car Shop Management Application</title>
    <link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css">
  </head>
  <body>
    <h1>Car Shop Management System</h1><br>

    <ul>
      <li><a href="http://localhost:8080/Carshop_Management/listcar">All Car
List</a></li>
      <li><a href="http://localhost:8080/Carshop_Management/new">Add a New
Car</a></li>
      <li><a href="http://localhost:8080/Carshop_Management/listcar">Edit Car</a></li>
    </ul>
  </body>
</html>
```

CarForm.jsp

```
<%--
  Document   : CarForm
  Created on : 15 Jun 2024, 1:23:21 pm
  Author    : ARIF HAIKAL
--%>

<%@page contentType="text/html" pageEncoding="UTF-8"%>
<%@taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<!DOCTYPE html>
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <title>Car Shop 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">

    <script>
      function updateBrand() {
        var selectedBrand = document.getElementById("brand").value;
        document.getElementById("displayBrand").value = selectedBrand;
      }
    </script>
  </head>
  <body>
    <header>
      <nav class="navbar navbar-expand-md navbar-dark" style="background-color:
tomato">
        <div>
          <a href="" class="navbar-brand"> Car Management Application </a>
        </div>

        <ul class="navbar-nav">
          <li><a href="<%=request.getContextPath()%>/listcar" class="nav-
link">Cars</a></li>
        </ul>
      </nav>
    </header>
    <br>
    <div class="container col-md-5">
      <div class="card">
        <div class="card-body">
          <c:if test="${car != null}">
            <form action="update" method="post">
          </c:if>
          <c:if test="${car == null}">
            <form action="insert" method="post">
          </c:if>

          <h2>
            <c:if test="${car != null}">
```

```

        Edit Car
    </c:if>
    <c:if test="${car == null}">
        Add New Car
    </c:if>
</h2>

<c:if test="${car != null}">
    <input type="hidden" name="car_id" value="<c:out value='${car.id}' />" />
</c:if>

    <fieldset class="form-group">
        <label>Car Brand</label>
        <input type="text" id="displayBrand" value="<c:out value='${car.brand}' />"
class="form-control" readonly>
        <input list="carList" id="brand" class="form-control" name="brand"
onchange="updateBrand()" >
        <datalist id="carList">
            <option value="Proton">
            <option value="Perodua">
            <option value="Toyota">
            <option value="Honda">
            <option value="Nissan">
            <option value="Ford">
            <option value="Hyundai">
            <option value="Kia">
        </datalist>
    </fieldset>

    <fieldset class="form-group">
        <label>Car Model</label><input type="text" value="<c:out
value='${car.model}' />"
                                class="form-control" name="model"
required="required">
    </fieldset>

    <fieldset class="form-group">
        <label>Engine Cylinder</label><input type="text" value="<c:out
value='${car.cylinder}' />"
                                class="form-control" name="cylinder">
    </fieldset>

    <fieldset class="form-group">
        <label>Car Price</label><input type="text" value="<c:out
value='${car.price}' />"
                                class="form-control" name="price">
    </fieldset>

    <button type="submit" class="btn btn-success">Save</button>
</form>
</div>
</div>
</div>
</body>
</html>

```

CarList.jsp

```
<%--
  Document   : CarList
  Created on : 15 Jun 2024, 1:23:30 pm
  Author    : ARIF HAIKAL
--%>

<%@page contentType="text/html" pageEncoding="UTF-8"%>
<%@taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<!DOCTYPE html>
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <title>Car Shop 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="" class="navbar-brand"> Car Shop Management App </a>
          </div>

          <ul class="navbar-nav">
            <li><a href="<%=request.getContextPath()%>/listcar" class="nav-
link">Cars</a></li>
          </ul>
        </nav>
      </header>
      <br>

      <div class="row">
        <div class="container">
          <h3 class="container">List of Cars</h3>
          <hr>
          <div class="container text-left">
            <a href="<%=request.getContextPath()%>/new" class="btn btn-
success">Add New Car</a>
          </div>
          <br>
          <table class="table table-bordered">
            <thead>
              <tr>
                <th>ID</th>
                <th>Brand</th>
                <th>Model</th>
                <th>Engine Cylinder</th>
                <th>Price</th>
            </thead>
          </table>
        </div>
      </div>
    </body>
  </html>
--%>
```

[illegible]

error.jsp

```
<%--
  Document   : error
  Created on : 15 Jun 2024, 1:23:40 pm
  Author    : ARIF HAIKAL
--%>

<%@page contentType="text/html" pageEncoding="UTF-8" isErrorPage="true" %>
<!DOCTYPE html>
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <title>Error page</title>
  </head>
  <body>
    <center>
      <h1>Error</h1>
      <h2><%=exception.getMessage() %><br/></h2>
    </center>
  </body>
</html>
```

CarServlet.java

```
package com.Controller;

import jakarta.servlet.RequestDispatcher;
import java.io.IOException;
import java.io.PrintWriter;
import jakarta.servlet.ServletException;
import jakarta.servlet.annotation.WebServlet;
import jakarta.servlet.http.HttpServlet;
import jakarta.servlet.http.HttpServletRequest;
import jakarta.servlet.http.HttpServletResponse;
import java.sql.SQLException;
import java.util.List;

import com.DAO.CarDAO;
import com.Model.Car;

/**
 *
 * @author ARIF HAIKAL
 */

@WebServlet("/")
public class CarServlet extends HttpServlet {

    /**
     * Processes requests for both HTTP GET and POST
     * methods.
     *
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error occurs
     * @throws IOException if an I/O error occurs
     */
    private CarDAO carDAO;

    @Override
    public void init(){
        carDAO = new CarDAO();
    }

    // <editor-fold defaultstate="collapsed" desc="HttpServlet methods. Click on the + sign
    on the left to edit the code.">
    /**
     * Handles the HTTP GET method.
     *
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error occurs
     * @throws IOException if an I/O error occurs
     */
    @Override
    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":
            insertCar(request, response);
            break;
        case "/delete":
            deleteCar(request, response);
            break;
        case "/edit":
            showEditForm(request, response);
            break;
        case "/update":
            updateCar(request, response);
            break;
        default:
            listCar(request, response);
            break;
    }
} catch(SQLException ex){
    throw new ServletException(ex);
}

}

private void listCar(HttpServletRequest request, HttpServletResponse response)
    throws SQLException, IOException, ServletException{
    List < Car > listCar = carDAO.selectAllCars();
    request.setAttribute("listCar", listCar);
    RequestDispatcher dispatcher = request.getRequestDispatcher("CarList.jsp");
    dispatcher.forward(request, response);
}

private void showNewForm(HttpServletRequest request, HttpServletResponse
response)
    throws ServletException, IOException{
    RequestDispatcher dispatcher = request.getRequestDispatcher("CarForm.jsp");
    dispatcher.forward(request, response);
}

private void showEditForm(HttpServletRequest request, HttpServletResponse
response)
    throws SQLException, ServletException, IOException{
    int id = Integer.parseInt(request.getParameter("car_id"));
    Car existingCar = carDAO.selectCar(id);
    RequestDispatcher dispatcher = request.getRequestDispatcher("CarForm.jsp");
    request.setAttribute("car", existingCar);
    dispatcher.forward(request, response);
}

private void insertCar(HttpServletRequest request, HttpServletResponse response)
    throws SQLException, IOException{

```



```

        String brand = request.getParameter("brand");
        String model = request.getParameter("model");
        int cylinder = Integer.parseInt(request.getParameter("cylinder"));
        double price = Double.parseDouble(request.getParameter("price"));
        Car car = new Car(brand, model, cylinder, price);
        carDAO.insertCar(car);
        response.sendRedirect("listcar");
    }

    private void updateCar(HttpServletRequest request, HttpServletResponse response)
        throws SQLException, IOException{
        int id = Integer.parseInt(request.getParameter("car_id"));
        String brand = request.getParameter("brand");
        String model = request.getParameter("model");
        int cylinder = Integer.parseInt(request.getParameter("cylinder"));
        double price = Double.parseDouble(request.getParameter("price"));
        Car car = new Car(brand, model, cylinder, price);
        carDAO.updateCar(car);
        response.sendRedirect("listcar");
    }

    private void deleteCar(HttpServletRequest request, HttpServletResponse response)
        throws SQLException, IOException{
        int id = Integer.parseInt(request.getParameter("car_id"));
        carDAO.deleteCar(id);
        response.sendRedirect("listcar");
    }

    /**
     * Handles the HTTP <code>POST</code> method.
     *
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error occurs
     * @throws IOException if an I/O error occurs
     */
    @Override
    protected void doPost(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
        doGet(request, response);
    }

    /**
     * Returns a short description of the servlet.
     *
     * @return a String containing servlet description
     */
    @Override
    public String getServletInfo() {
        return "Short description";
    } // </editor-fold>
}

```

CarDAO.java

```
package com.DAO;

import java.sql.*;
import java.sql.SQLException;
import java.util.*;

import com.Model.Car;

/**
 *
 * @author ARIF HAIKAL
 */
public class CarDAO {

    Connection connection = null;
    private String jdbcURL = "jdbc:mysql://localhost:3306/CarshopManagement";
    private String jdbcUsername = "root";
    private String jdbcPassword = "admin@123";

    private static final String INSERT_CARS_SQL = "INSERT INTO carpricelist(brand,
model, cylinder, price) VALUES (?, ?, ?, ?)";
    private static final String SELECT_CAR_BY_ID = "select car_id, brand, model, cylinder,
price from carpricelist where car_id=?";
    private static final String SELECT_ALL_CARS = "select * from carpricelist";
    private static final String DELETE_CARS_SQL = "delete from carpricelist where car_id
= ?";
    private static final String UPDATE_CARS_SQL = "update carpricelist set brand =
?,model= ?, cylinder= ?, price= ? where car_id = ?";

    public CarDAO(){}

    protected Connection getConnection(){
        Connection connection = null;
        try{
            Class.forName("com.mysql.jdbc.Driver");
            connection = DriverManager.getConnection(jdbcURL, jdbcUsername,
jdbcPassword);
            System.out.println("Database connected!");
        }catch(SQLException e){
            e.printStackTrace();
        }catch(ClassNotFoundException e){
            e.printStackTrace();
        }
        return connection;
    }

    public void insertCar(Car car) throws SQLException{
        System.out.println(INSERT_CARS_SQL);
        try(Connection connection = getConnection(); PreparedStatement
preparedStatement =
            connection.prepareStatement(INSERT_CARS_SQL)){
            preparedStatement.setString(1, car.getBrand());
            preparedStatement.setString(2, car.getModel());
```

```

        preparedStatement.setInt(3, car.getCylinder());
        preparedStatement.setDouble(4, car.getPrice());
        System.out.println(preparedStatement);
        preparedStatement.executeUpdate();
    }catch(SQLException e){
        printSQLException(e);
    }
}

public Car selectCar(int id){
    Car car = null;
    // Step 1: Establishing a Connection
    try(Connection connection = getConnection();
        // Step 2: Create a statement using connection
        PreparedStatement preparedStatement =
connection.prepareStatement(SELECT_CAR_BY_ID)){
        preparedStatement.setInt(1, id);
        System.out.println(preparedStatement);
        ResultSet rs = preparedStatement.executeQuery();

        while(rs.next()){
            String brand = rs.getString("brand");
            String model = rs.getString("model");
            int cylinder = rs.getInt("cylinder");
            double price = rs.getDouble("price");
            car = new Car(id, brand, model, cylinder, price);
        }
    }catch (SQLException e){
        printSQLException(e);
    }
    return car;
}

public List < Car > selectAllCars(){
    List <Car> cars = new ArrayList <>();
    try (Connection connection = getConnection();

        PreparedStatement preparedStatement =
connection.prepareStatement(SELECT_ALL_CARS)){
        System.out.println(preparedStatement);
        ResultSet rs = preparedStatement.executeQuery();

        while(rs.next()){
            int id = rs.getInt("car_id");
            String brand = rs.getString("brand");
            String model = rs.getString("model");
            int cylinder = rs.getInt("cylinder");
            double price = rs.getDouble("price");
            cars.add(new Car(id, brand, model, cylinder, price));
        }
    }catch(SQLException e){
        printSQLException(e);
    }
    return cars;
}

```

```

public boolean deleteCar(int id) throws SQLException{
    boolean rowDeleted;
    try(Connection connection = getConnection(); PreparedStatement statement =
        connection.prepareStatement(DELETE_CARS_SQL)){
        statement.setInt(1, id);
        rowDeleted = statement.executeUpdate() > 0;
    }
    return rowDeleted;
}

public boolean updateCar(Car car) throws SQLException{
    boolean rowUpdated;
    try(Connection connection = getConnection(); PreparedStatement statement =
        connection.prepareStatement(UPDATE_CARS_SQL)){
        statement.setString(1, car.getBrand());
        statement.setString(2, car.getModel());
        statement.setInt(3, car.getCylinder());
        statement.setDouble(4, car.getPrice());
        statement.setInt(5, car.getCar_id());

        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();
            }
        }
    }
}
}

```

Car.java

```
package com.Model;

/**
 *
 * @author ARIF HAIKAL
 */
public class Car {

    protected int car_id;
    protected String brand;
    protected String model;
    protected int cylinder;
    protected double price;

    public Car(){}

    public Car(String brand, String model, int cylinder, double price) {
        super();
        this.brand = brand;
        this.model = model;
        this.cylinder = cylinder;
        this.price = price;
    }

    public Car(int car_id, String brand, String model, int cylinder, double price) {
        this.car_id = car_id;
        this.brand = brand;
        this.model = model;
        this.cylinder = cylinder;
        this.price = price;
    }

    public int getCar_id() {
        return car_id;
    }

    public void setCar_id(int car_id) {
        this.car_id = car_id;
    }

    public String getBrand() {
        return brand;
    }

    public void setBrand(String brand) {
        this.brand = brand;
    }

    public String getModel() {
        return model;
    }
}
```

```
public void setModel(String model) {  
    this.model = model;  
}  
  
public int getCylinder() {  
    return cylinder;  
}  
  
public void setCylinder(int cylinder) {  
    this.cylinder = cylinder;  
}  
  
public double getPrice() {  
    return price;  
}  
  
public void setPrice(double price) {  
    this.price = price;  
}  
}
```

Output :

