UNIVERSITI MALAYSIA TERENGGANU

CSM3023 WEB BASED APPLICATION DEVELOPMENT (K1)

BACHELOR OF COMPUTER SCIENCE (MOBILE COMPUTING) WITH HONORS

LAB 8 – AN MV EXAMPLE WITH SERVLET AND JSP
SEMESTER 4 2023/2024

Prepared for:

DR. MOHAMAD NOR HASSAN

Prepared by:

NUR ARINA BINTI ABDUL MALEK (S65361)

LAB 8

An MVC Example with Servlets and JSP

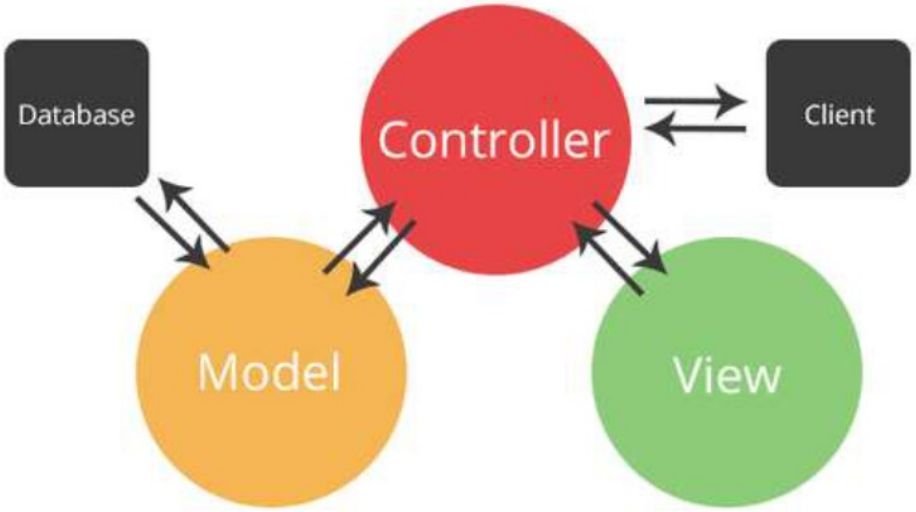


Table of Contents

Arahan:

Manual makmal ini adalah untuk kegunaan pelajar-pelajar Fakulti Teknologi Kejuruteraan Kelautan dan Informatik - FTKKI, Universiti Malaysia Terengganu (UMT) sahaja. Tidak dibenarkan mencetak dan mengedar manual ini tanpa kebenaran rasmi daripada penulis.

Sila ikuti langkah demi langkah sebagaimana yang dinyatakan di dalam manual. Tandakan (√) setiap langkah yang telah selesai dibuat dan tulis kesimpulan bagi setiap aktiviti yang telah selesai dijalankan.

Instruction:

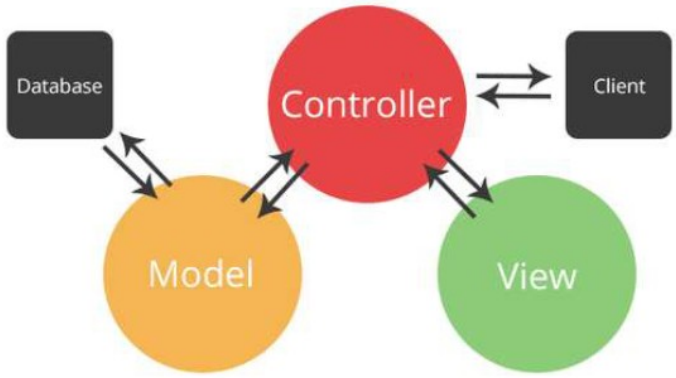
This laboratory manual is for use by the students of the Faculty of Ocean Engineering Technology and Informatics (FTKKI), Universiti Malaysia Terengganu only.

It is not permissible to print and distribute this manual without the official authorisation of the author. Please follow step by step as described in the manual. Tick (√) each step completed and write the conclusions for each completed activity.

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

MVC Pattern stands for Model-View-Controller Pattern. This pattern is used to separate application's concerns.

- Model - Model represents an object or JAVA POJO carrying data. It can also have logic to update controller if its data changes.
- View - View represents the visualization of the data that model contains.
- Controller - Controller acts on both model and view. It controls the data flow into model object and updates the view whenever data changes. It keeps view and model separate.



Benefits of MVC in JSP and Servlet Web Application

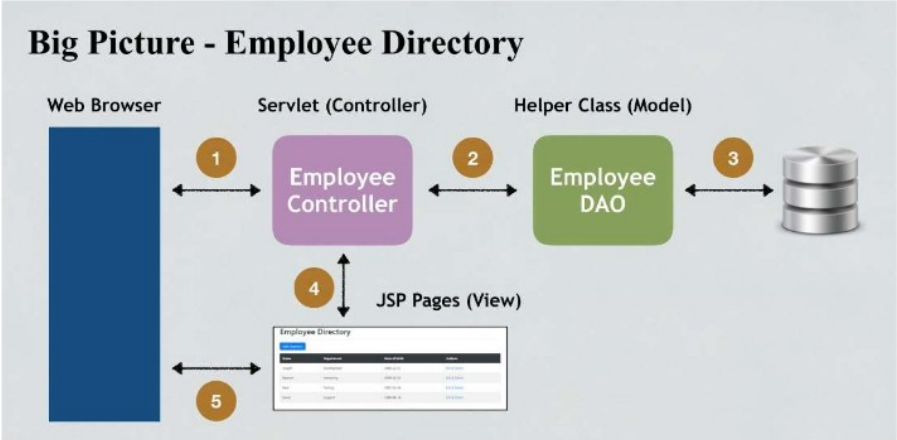
- Minimizes HTML code in Servlet no more: out.println(…) in Servlet code.
- Minimize Java business logic in JSPs no more large scriptlets in JSP code
- It separates the presentation layer from the business layer
- The Controller performs the action of invoking the Model and sending data to View
- The Model is not even aware that it is used by some web application or a desktop application

In this Lab8 activity, student will follow step by step to create a MVC application using JSP, Servlet and MySQL to create, read, update, and delete (CRUD) the student records into the database.

Steps Involved in the Application

Basically, there are 4 main steps involved in this application that are given below:

- Capture the employee records and store it into the database.
- Fetch the employee records from the database and display it on the JSP.
- Update the existing employee records into the database.
- Delete the employee records from the database.



Step by step of Application development:

Step 1 - Create table employees in COMPANY database schema.

```
CREATE DATABASE IF NOT EXISTS Company;
USE Company;

CREATE TABLE IF NOT EXISTS employees (
    id INT NOT NULL AUTO_INCREMENT,
    Name VARCHAR(60),
    Email VARCHAR(50),
    Position VARCHAR(15),
    PRIMARY KEY (id)
 )
```

Step 2 - Create new web application project, named as Employee_Management.

Step 3 - Create three Java class that representing :

- EmployeeDAO.java (act as a Data Access Object (DAO) and to open /close database connection),
- Employee.java (act as a JavaBeans to represent business object), and
- EmployeeServlet.java (act to perform CRUD process)

EmployeeDAO.java

Name the package as com.DAO

```java
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package com.DAO;

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 com.Model.Employee;

public class EmployeeDAO {
    Connection connection = null;
    private String jdbcURL = "jdbc:mysql://localhost:3306/company";
    private String jdbcUsername = "yusro";
    private String jdbcPassword = "admin";

    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 = ?;";
```

```java
    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);
        } catch (SQLException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        } catch (ClassNotFoundException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
        return connection;
    }

    public void insertEmployee(Employee employee) throws SQLException {
        System.out.println(INSERT_EMPLOYEES_SQL);
        // try-with-resource statement will auto close the connection.
        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();
```

```java
        } 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 object
            PreparedStatement preparedStatement = connection.prepareStatement(SELECT_EMPLOYEE_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 position = rs.getString("position");
                employee = new Employee(id, name, email, position);
            }
        } catch (SQLException e) {
            printSQLException(e);
        }
        return employee;
    }
```

```
87
88      public List < Employee > selectAllEmployees() {
89
90          // using try-with-resources to avoid closing resources (boiler plate code)
91          List < Employee > employees = new ArrayList < > ();
92          // Step 1: Establishing a Connection
            try (Connection connection = getConnection();
94
95              // Step 2:Create a statement using connection object
96              PreparedStatement preparedStatement =
97                      connection.prepareStatement(SELECT_ALL_EMPLOYEES);) {
98              System.out.println(preparedStatement);
99              // Step 3: Execute the query or update query
100             ResultSet rs = preparedStatement.executeQuery();
101
102             // Step 4: Process the ResultSet object.
103             while (rs.next()) {
104                 int id = rs.getInt("id");
105                 String name = rs.getString("name");
106                 String email = rs.getString("email");
107                 String position = rs.getString("position");
108                 employees.add(new Employee(id, name, email, position));
109             }
110         } catch (SQLException e) {
111             printSQLException(e);
112         }
113         return employees;
114     }
```

```
116     public boolean deleteEmployee(int id) throws SQLException {
117         boolean rowDeleted;
            try (Connection connection = getConnection(); PreparedStatement statement =
119                 connection.prepareStatement(DELETE_EMPLOYEES_SQL);) {
120             statement.setInt(1, id);
121             rowDeleted = statement.executeUpdate() > 0;
122         }
123         return rowDeleted;
124     }
125
126     public boolean updateEmployee(Employee employee) throws SQLException {
127         boolean rowUpdated;
            try (Connection connection = getConnection(); PreparedStatement statement =
129                 connection.prepareStatement(UPDATE_EMPLOYEES_SQL);) {
130             statement.setString(1, employee.getName());
131             statement.setString(2, employee.getEmail());
132             statement.setString(3, employee.getPosition());
133             statement.setInt(4, employee.getId());
134
135             rowUpdated = statement.executeUpdate() > 0;
136         }
137         return rowUpdated;
138     }
139
```

```
139
140     private void printSQLException(SQLException ex) {
141         for (Throwable e: ex) {
142             if (e instanceof SQLException) {
143                 e.printStackTrace(System.err);
144                 System.err.println("SQLState: " + ((SQLException) e).getSQLState());
145                 System.err.println("Error Code: " + ((SQLException) e).getErrorCode());
146                 System.err.println("Message: " + e.getMessage());
147                 Throwable t = ex.getCause();
148                 while (t != null) {
149                     System.out.println("Cause: " + t);
150                     t = t.getCause();
151                 }
152             }
153         }
154     }
155 }
156
```

Employee.java

Name the package as com.Model

```java
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package com.Model;

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) {
        super();
        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

Name the package as com.WEB

```java
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package com.WEB;

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

@WebServlet("/")
public class EmployeeServlet extends HttpServlet {
    // private static final long serialVersionUID = 1 L;
    private EmployeeDAO employeeDAO;

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

    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":
                    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);

    }
```

```
90      private void insertEmployee(HttpServletRequest request, HttpServletResponse response)
91      throws SQLException, IOException {
92          String name = request.getParameter("name");
93          String email = request.getParameter("email");
94          String position = request.getParameter("position");
95          Employee newEmployee = new Employee(name, email, position);
96          employeeDAO.insertEmployee(newEmployee);
97          response.sendRedirect("list");
98      }
99
100     private void updateEmployee(HttpServletRequest request, HttpServletResponse response)
101     throws SQLException, IOException {
102         int id = Integer.parseInt(request.getParameter("id"));
103         String name = request.getParameter("name");
104         String email = request.getParameter("email");
105         String position = request.getParameter("position");
106
107         Employee employee= new Employee(id, name, email, position);
108         employeeDAO.updateEmployee(employee);
109         response.sendRedirect("list");
110     }
111
112     private void deleteEmployee(HttpServletRequest request, HttpServletResponse response)
113     throws SQLException, IOException {
114         int id = Integer.parseInt(request.getParameter("id"));
115         employeeDAO.deleteEmployee(id);
116         response.sendRedirect("list");
117
118     }
119 }
```

Step 4 - Create these files:

1. File web.xml

   Java web applications use a deployment descriptor file to determine how URLs map to servlets, which URLs require authentication, and other information. This file is named web.xml, and resides in the app's WAR under the WEB-INF/ directory. web.xml is part of the servlet standard for web applications.

2. File EmployeeForm.jsp (used for Add and Edit/Update process)

```
1   <%@ page language="java" contentType="text/html; charset=UTF-8"  pageEncoding="UTF-8"%>
2   <%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
3   <!DOCTYPE html>
4   <html>
5   <head>
6       <title>Employee Management Application</title>
7       <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css"
8           integrity="sha384-ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T" crossorigin="anonymous">
9   </head>
10  <body>
11      <header>
12          <nav class="navbar navbar-expand-md navbar-dark" style="background-color: tomato">
13              <div>
14                  <a href="" class="navbar-brand"> Employee Management App </a>
15              </div>
16
17              <ul class="navbar-nav">
18                  <li><a href="<%=request.getContextPath()%>/list" class="nav-link">Employees</a></li>
19              </ul>
20          </nav>
21      </header>
22      <br>
23      <div class="container col-md-5">
24          <div class="card">
25              <div class="card-body">
26                  <c:if test="${employee != null}">
27                      <form action="update" method="post">
28                  </c:if>
29                  <c:if test="${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" value="<c:out value='${employee.position}' />" class="form-control" readonly >
                        <input list="positionList" id="position" class="form-control" name="position" >
                        <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>
    </div>
</body>
</html>
```

3. File EmployeeList.jsp (used for displaying all employee records)

```
<%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>

<!DOCTYPE html>
<html>

<head>
    <title>Employee 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"> 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="alert alert-success" *ngIf='message'>{{message}}</div> -->

        <div class="container">
            <h3 class="text-center">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>
                    <!--   for (Todo todo: todos) {  -->
                    <c:forEach var="employee" items="${listEmployee}">
                        <tr>
                            <td>
                                <c:out value="${employee.id}" />
                            </td>
                            <td>
                                <c:out value="${employee.name}" />
                            </td>
                            <td>
                                <c:out value="${employee.email}" />
                            </td>
                            <td>
                                <c:out value="${employee.position}" />
                            </td>
                            <td><a href="edit?id=<c:out value='${employee.id}' />">Edit</a>     
                                <a href="delete?id=<c:out value='${employee.id}' />">Delete</a></td>
                        </tr>
                    </c:forEach>
                </tbody>
            </table>
        </div>
    </div>
</body>

</html>
```

4. File error.jsp

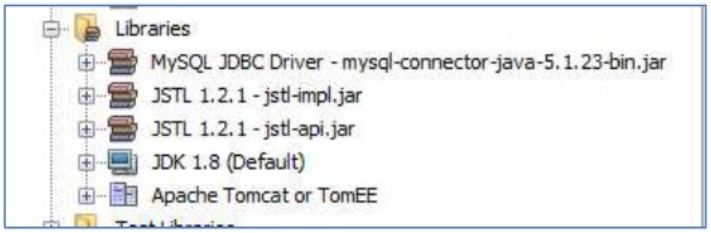
```
<%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8" isErrorPage="true" %>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
    <title>Error page</title>
</head>
<body>
    <center>
        <h1>Error</h1>
        <h2><%=exception.getMessage() %><br/> </h2>
    </center>
</body>
</html>
```

5. File index.jsp

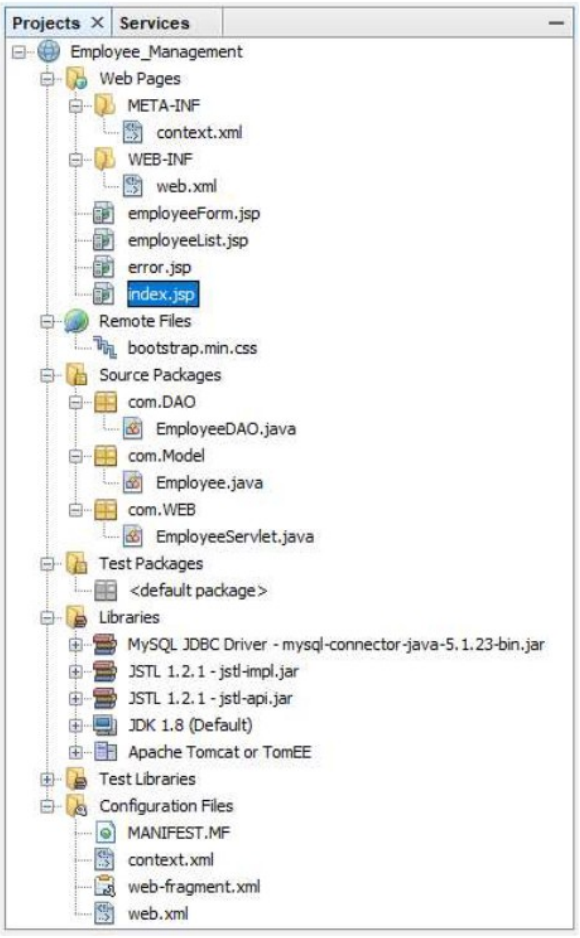
   Contents of the Index.jsp:

```jsp
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<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>
        <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>
```

Step 5 - Add these libraries:



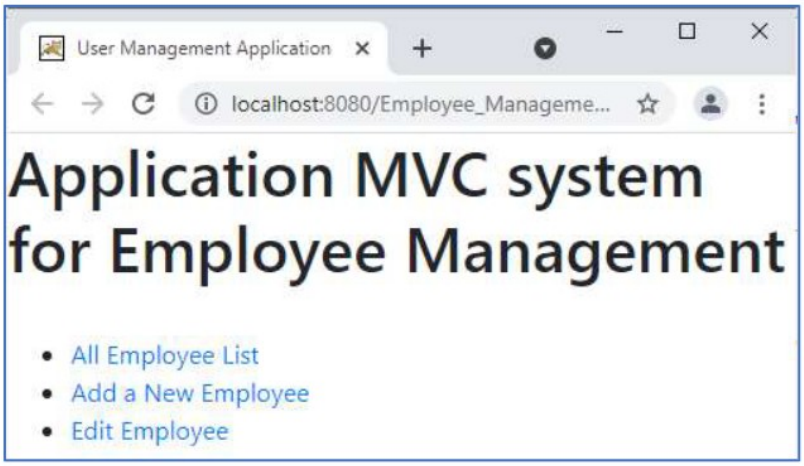
Finally, the Project schema should be like this:

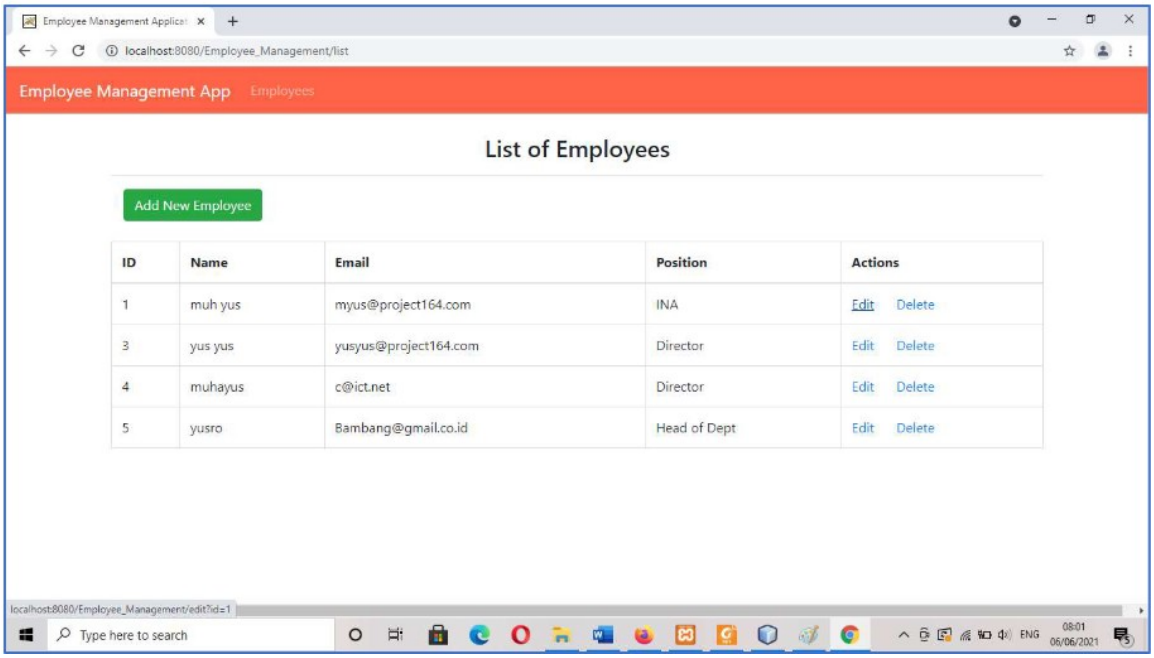


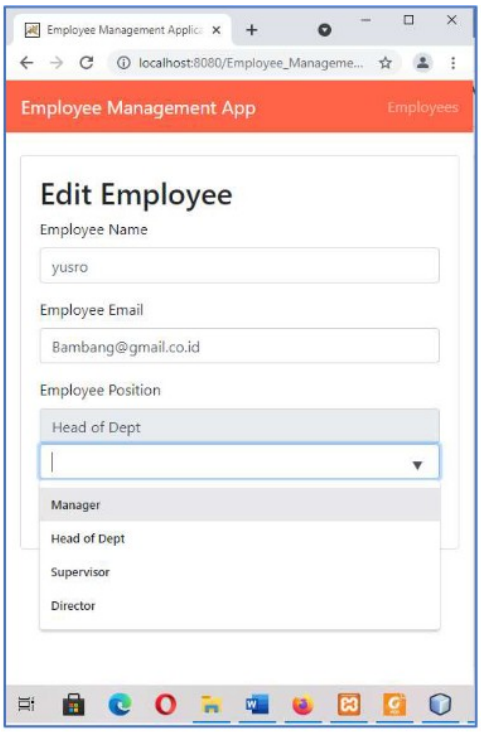
Step 6 - Running the program and try CRUD process:

1. Run index.jsp page.
2. Click List All User button to show all records.
3. Click Add User button to create new record.
4. Click hyperlink Update to do edit/update an existing record.
5. Click hyperlink Delete to do delete an existing record.

The result:







Exercise

Using this database shema, please create MVC Application [CRUD] for Car Shop, using JSP, Servlet, and MySQL.

```
CREATE DATABASE if not EXISTS carshop;
USE carshop;

CREATE TABLE if not EXISTS CarPricelist(
    Car_id INT NOT NULL AUTO_INCREMENT,
    Brand VARCHAR(15),
    Model VARCHAR(30),
    Cyclinder INT,
    Price DOUBLE,
    PRIMARY KEY (Car_id)
);
```

The application should be able to handle these activities:

1. View all data
2. Add new data
3. Edit/Update current data
4. Delete the specific data

MY WORK
EmployeeForm.jsp

```jsp
<%--
    Document   : EmployeeForm
    Created on : 5 Jun 2024, 5:01:18 pm
    Author     : rynaa
--%>


<%@page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>
<%@taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<!DOCTYPE html>
<html>
<head>
    <title>Employee Management Application</title>
    <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/css/bootstrap.min.css">
  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.7.1/jquery.min.js"></script>
  <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/js/bootstrap.min.js"></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>
    </div>
</body>
</html>
```

EmployeeList.jsp

```jsp
<%--
    Document   : EmployeeList
    Created on : 6 Jun 2024, 10:49:46 pm
    Author     : rynaa
--%>

<%@page contentType="text/html" pageEncoding="UTF-8"%>
<%@taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<!DOCTYPE html>
<html>
<head>
    <title>Employee Management Application</title>
    <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/css/bootstrap.min.css">
    <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.7.1/jquery.min.js"></script>
    <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/js/bootstrap.min.js"></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>
    </header>
    <br>
    <div class="row">
        <div class="container">
            <h3 class="text-center">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>
                    <c:forEach var="employee" items="${listEmployee}">
                        <tr>
                            <td>
                                <c:out value="${employee.id}" />
                            </td>
                            <td>
                                <c:out value="${employee.name}" />
                            </td>
                            <td>
                                <c:out value="${employee.email}" />
                            </td>
                            <td>
                                <c:out value="${employee.position}" />
                            </td>
                            </td>
                            <td><a href="edit?id=<c:out value="${employee.id}" />">Edit</a>        
                                <a href="delete?id=<c:out value="${employee.id}" />">Delete</a></td>
                        </tr>
                    </c:forEach>
                    </tbody>
                </table>
            </div>
        </div>
    </body>
    </html>
```

FileError.jsp

```jsp
<%--
    Document   : FileError
    Created on : 7 Jun 2024, 1:45:04 am
    Author     : rynaa
--%>

<%@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 page</title>
</head>
<body>
    <center>
        <h1>Error</h1>
        <h2><-exception.getMessage() ><br/> </h2>
    </center>
</body>
</html>
```

Index.jsp

```jsp
<%--
    Document   : Index
    Created on : 7 Jun 2024, 8:49:57 pm
    Author     : rynaa
--%>

<%@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://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/css/bootstrap.min.css">
        <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.7.1/jquery.min.js"></script>
        <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/js/bootstrap.min.js"></script>
    </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>
```

EmployeeDAO.java

```java
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/JSP_Servlet/Servlet.java to edit this template
 */
package com.DAO;

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 com.Model.Employee;

public class EmployeeDAO {
    private String jdbcURL = "jdbc:mysql://localhost:3306/company"; // Corrected URL
    private String jdbcUsername = "root";
    private String jdbcPassword = "admin";

    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.cj.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;
        try (Connection connection = getConnection();
             PreparedStatement preparedStatement = connection.prepareStatement(SELECT_EMPLOYEE_BY_ID)) {
            preparedStatement.setInt(1, id);
```

com.DAO.EmployeeDAO › getConnection › connection

```
            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> selectAllEmployee() {
        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) {
                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

```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 com.Model;

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){
        super();
        this.id = id;
        this.name = name;
        this.email = email;
        this.position = position;
    }

    public void setId(int id) {
        this.id = id;
```

com.Model.Employee

```java
        this.id = id;
    }

    public void setName(String name) {
        this.name = name;
    }

    public void setEmail(String email) {
        this.email = email;
    }

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

    public int getId() {
        return id;
    }

    public String getName() {
        return name;
    }

    public String getEmail() {
        return email;
    }

    public String getPosition() {
        return position;
    }

}
```

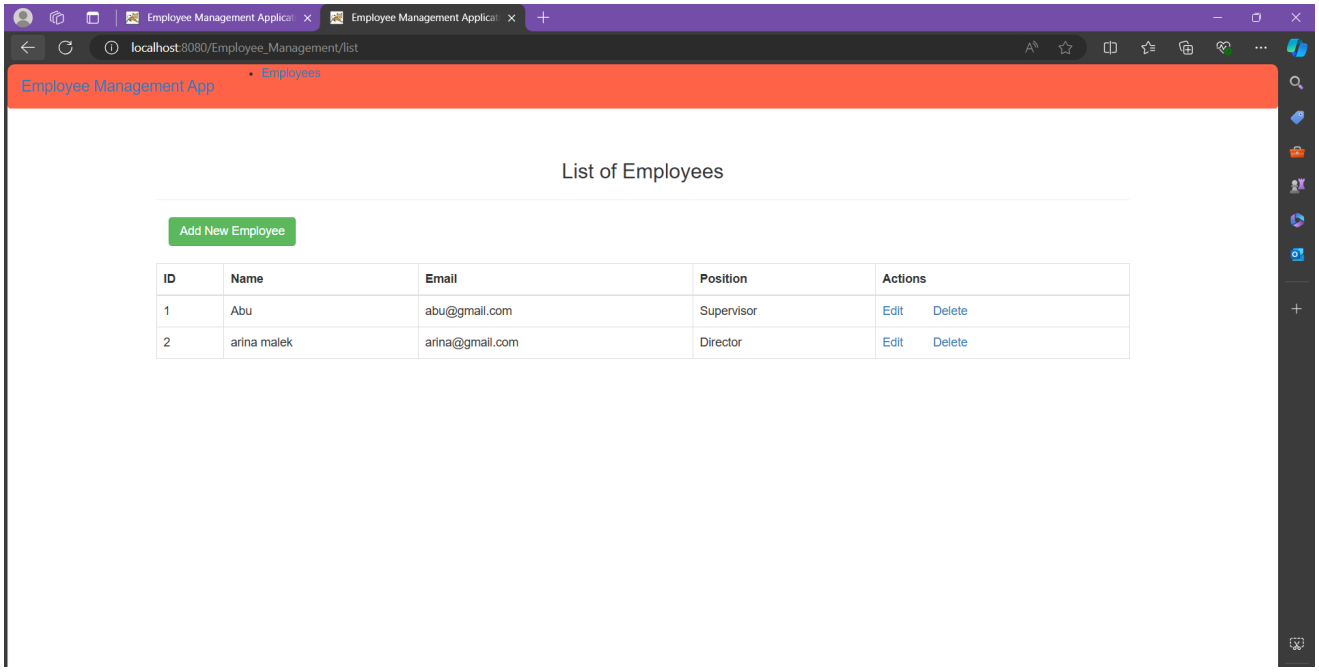
EmployeeServlet.java

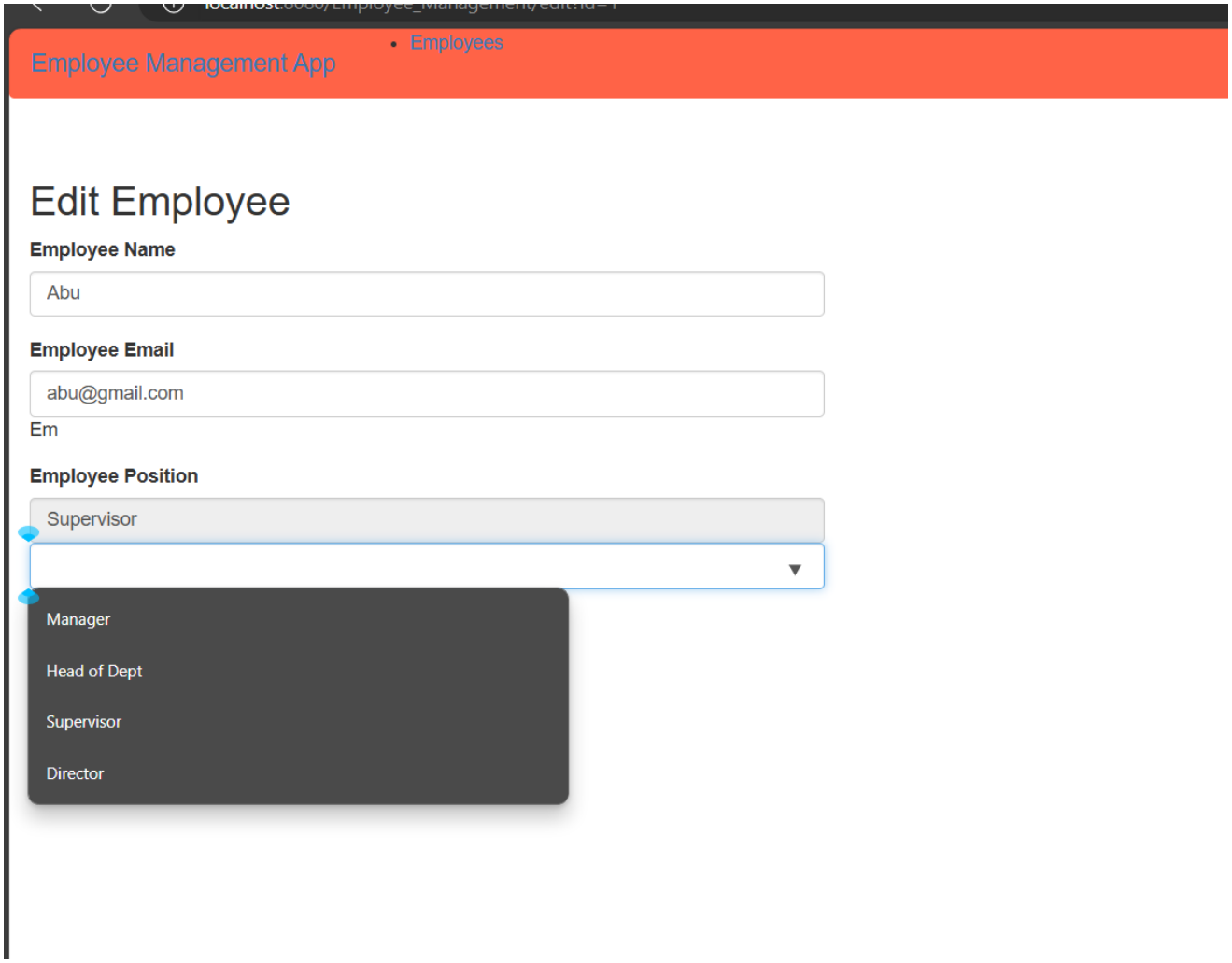
```java
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/JSP_Servlet/Servlet.java to edit this template
 */
package com.WEB;

import java.io.IOException;
import java.sql.SQLException;
import java.util.List;

import java.io.PrintWriter;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
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 com.DAO.EmployeeDAO;
import com.Model.Employee;
import jakarta.servlet.RequestDispatcher;

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

    private EmployeeDAO employeeDAO;

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

    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":
                        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.selectAllEmployee();
        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 {
```

```
            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");
    }
}
```

OUTPUT


Employee Management Applicat
Employee Management Applicat
localhost:8080/Employee_Management/list
Employee Management App
Employees
List of Employees
Add New Employee
ID
Name
Email
Position
Actions
1
Abu
abu@gmail.com
Supervisor
Edit
Delete
2
arina malek
arina@gmail.com
Director
Edit
Delete


Employee Management App
Employees
Edit Employee
Employee Name
Abu
Employee Email
abu@gmail.com
Em
Employee Position
Supervisor
Manager
Head of Dept
Supervisor
Director

EXERCISE
carForm.jsp

```jsp
<%--
    Document   : carForm
    Created on : 12 Jun 2024, 1:50:03 am
    Author     : rynaa
--%>

<%@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://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/css/bootstrap.min.css">
    </head>
    <body>
        <div class="container">
            <h2>${car == null ? 'Add New Car' : 'Edit Car'}</h2>
            <form action="${car == null ? 'insert' : 'update'}" method="post">
                <c:if test="${car != null}">
                    <input type="hidden" name="car_id" value="${car.car_id}">
                </c:if>
                <div class="form-group">
                    <label>Brand</label>
                    <input type="text" name="brand" class="form-control" value="${car != null ? car.brand : ''}" required>
                </div>
                <div class="form-group">
                    <label>Model</label>
                    <input type="text" name="model" class="form-control" value="${car != null ? car.model : ''}" required>
                </div>
                <div class="form-group">
                    <div class="form-group">
                        <label>Cyclinder</label>
                        <input type="text" name="cyclinder" class="form-control" value="${car != null ? car.cyclinder : ''}" required>
                    </div>
                    <div class="form-group">
                        <label>Price</label>
                        <input type="text" name="price" class="form-control" value="${car != null ? car.price : ''}" required>
                    </div>
                    <button type="submit" class="btn btn-primary">Submit</button>
                </form>
            </div>
        </body>
    </html>
```

carIndex.jsp

```jsp
<%--
    Document   : carIndex
    Created on : 13 Jun 2024, 1:51:10 am
    Author     : rynaa
--%>

<%@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 Application</title>
        <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/css/bootstrap.min.css">
        <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.7.1/jquery.min.js"></script>
        <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/js/bootstrap.min.js"></script>
    </head>
    <body>
        <h1>Application MVC system for Car Shop</h1><br>

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

```

carList.jsp

```jsp
<%--
    Document   : carList
    Created on : 13 Jun 2024, 2:34:34 am
    Author     : rynaa
--%>

<%@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://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/css/bootstrap.min.css">
    </head>
    <body>
        <div class="container">
            <h2>Car List</h2>
            <table class="table">
                <thead>
                    <tr>
                        <th>ID</th>
                        <th>Brand</th>
                        <th>Model</th>
                        <th>Cyclinder</th>
                        <th>Price</th>
                        <th>Actions</th>
                    </tr>
                </thead>
                <tbody>
                    <c:forEach var="car" items="${listCar}">
                        <tr>
                            <td>${car.car_id}</td>
                            <td>${car.brand}</td>
                            <td>${car.model}</td>
                            <td>${car.cyclinder}</td>
                            <td>${car.price}</td>
                            <td>
                                <a href="edit?car_id=${car.car_id}" class="btn btn-info">Edit</a>
                                <a href="delete?car_id=${car.car_id}" class="btn btn-danger">Delete</a>
                            </td>
                        </tr>
                    </c:forEach>
                </tbody>
            </table>
            <a href="new" class="btn btn-primary">Add New Car</a>
        </div>
    </body>
</html>
```

Error_1.jsp

```jsp
<%--
    Document   : error_1
    Created on : 14 Jun 2024, 1:52:03 am
    Author     : rynaa
--%>

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

carDAO.java

```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 com.DAO;

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 com.Model.Car;

public class CarDAO {
    private String jdbcURL = "jdbc:mysql://localhost:3306/carshop";
    private String jdbcUsername = "root";
    private String jdbcPassword = "admin";

    private static final String INSERT_CAR_SQL = "INSERT INTO CarPricelist (Brand, Model, Cyclinder, Price) VALUES (?, ?, ?, ?);";
    private static final String SELECT_CAR_BY_ID = "SELECT Car_id, Brand, Model, Cyclinder, Price FROM CarPricelist WHERE Car_id = ?";
    private static final String SELECT_ALL_CAR = "SELECT * FROM CarPricelist";
    private static final String DELETE_CAR_SQL = "DELETE FROM CarPricelist WHERE Car_id = ?;";
    private static final String UPDATE_CAR_SQL = "UPDATE CarPricelist SET Brand = ?, Model = ?, Cyclinder = ?, Price = ? WHERE Car_id = ?;";

    public CarDAO() {}

    protected Connection getConnection() {
        Connection connection = null;
```

```
        try {
            Class.forName("com.mysql.cj.jdbc.Driver");
            connection = DriverManager.getConnection(jdbcURL, jdbcUsername, jdbcPassword);
        } catch (SQLException | ClassNotFoundException e) {
            e.printStackTrace();
        }
        return connection;
    }

    public void insertCar(Car car) throws SQLException {
        System.out.println(INSERT_CAR_SQL);
        try (Connection connection = getConnection();
             PreparedStatement preparedStatement = connection.prepareStatement(INSERT_CAR_SQL)) {
            preparedStatement.setString(1, car.getBrand());
            preparedStatement.setString(2, car.getModel());
            preparedStatement.setString(3, car.getCyclinder());
            preparedStatement.setString(4, car.getPrice());
            System.out.println(preparedStatement);
            preparedStatement.executeUpdate();
        } catch (SQLException e) {
            printSQLException(e);
        }
    }

    public Car selectCar(int car_id) {
        Car car = null;
        try (Connection connection = getConnection();
             PreparedStatement preparedStatement = connection.prepareStatement(SELECT_CAR_BY_ID)) {
            preparedStatement.setInt(1, car_id);
            System.out.println(preparedStatement);
```

com.DAO.CarDAO › getConnection › connection

```
            ResultSet rs = preparedStatement.executeQuery();
            while (rs.next()) {
                String brand = rs.getString("Brand");
                String model = rs.getString("Model");
                String cyclinder = rs.getString("Cyclinder");
                String price = rs.getString("Price");
                car = new Car(car_id, brand, model, cyclinder, price);
            }
        } catch (SQLException e) {
            printSQLException(e);
        }
        return car;
    }

    public List<Car> selectAllCar() {
        List<Car> cars = new ArrayList<>();
        try (Connection connection = getConnection();
             PreparedStatement preparedStatement = connection.prepareStatement(SELECT_ALL_CAR)) {
            System.out.println(preparedStatement);
            ResultSet rs = preparedStatement.executeQuery();
            while (rs.next()) {
                int car_id = rs.getInt("Car_id");
                String brand = rs.getString("Brand");
                String model = rs.getString("Model");
                String cyclinder = rs.getString("Cyclinder");
                String price = rs.getString("Price");
                cars.add(new Car(car_id, brand, model, cyclinder, price));
            }
        } catch (SQLException e) {
            printSQLException(e);
        }
        return cars;
    }

    public boolean deleteCar(int car_id) throws SQLException {
        boolean rowDeleted;
        try (Connection connection = getConnection();
             PreparedStatement statement = connection.prepareStatement(DELETE_CAR_SQL)) {
            statement.setInt(1, car_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_CAR_SQL)) {
            statement.setString(1, car.getBrand());
            statement.setString(2, car.getModel());
            statement.setString(3, car.getCyclinder());
            statement.setString(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

```
/*
 * 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 com.Model;

/**
 *
 * @author User
 */

public class Car {
    private int Car_id;
    private String Brand;
    private String Model;
    private String Cyclinder;
    private String Price;

    // Constructor for insert
    public Car(String Brand, String Model, String Cyclinder, String Price) {
        this.Brand = Brand;
        this.Model = Model;
        this.Cyclinder = Cyclinder;
        this.Price = Price;
    }

    // Constructor for update and select by ID
    public Car(int Car_id, String Brand, String Model, String Cyclinder, String Price) {
        this.Car_id = Car_id;
        this.Brand = Brand;
        this.Model = Model;
        this.Cyclinder = Cyclinder;
        this.Price = Price;
    }

    // Getters and Setters
    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 String getCyclinder() {
```

```java

    public String getCyclinder() {
        return Cyclinder;
    }

    public void setCyclinder(String Cyclinder) {
        this.Cyclinder = Cyclinder;
    }

    public String getPrice() {
        return Price;
    }

    public void setPrice(String price) {
        this.Price = Price;
    }
}
```

carServlet.java

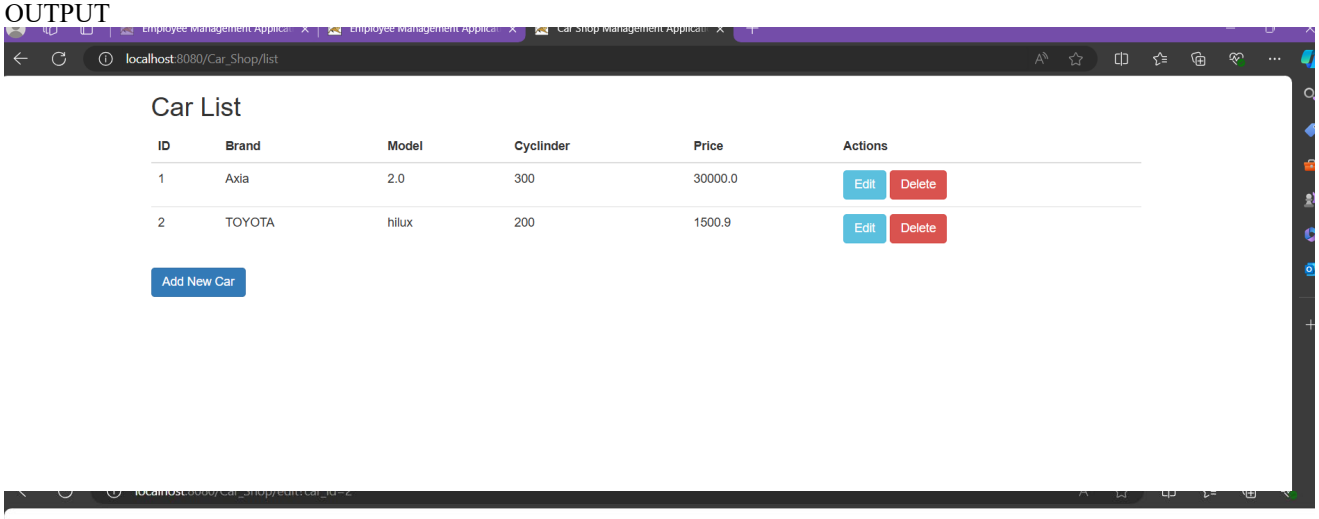
```java
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/JSP_Servlet/Servlet.java to edit this template
 */
package com.WEB;

import java.io.IOException;
import java.sql.SQLException;
import java.util.List;
import jakarta.servlet.RequestDispatcher;
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 com.DAO.CarDAO;
import com.Model.Car;

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

    private CarDAO carDAO;

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

    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":
                    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);
        }
    }
```

```
62
63      private void listCar(HttpServletRequest request, HttpServletResponse response)
64              throws SQLException, IOException, ServletException {
65          List<Car> listCar = carDAO.selectAllCar();
66          request.setAttribute("listCar", listCar);
67          RequestDispatcher dispatcher = request.getRequestDispatcher("carList.jsp");
68          dispatcher.forward(request, response);
69      }
70
71      private void showNewForm(HttpServletRequest request, HttpServletResponse response)
72              throws ServletException, IOException {
73          RequestDispatcher dispatcher = request.getRequestDispatcher("carForm.jsp");
74          dispatcher.forward(request, response);
75      }
76
77      private void showEditForm(HttpServletRequest request, HttpServletResponse response)
78              throws SQLException, ServletException, IOException {
79          int car_id = Integer.parseInt(request.getParameter("car_id"));
80          Car existingCar = carDAO.selectCar(car_id);
81          RequestDispatcher dispatcher = request.getRequestDispatcher("carForm.jsp");
82          request.setAttribute("car", existingCar);
83          dispatcher.forward(request, response);
84      }
85
86      private void insertCar(HttpServletRequest request, HttpServletResponse response)
87              throws SQLException, IOException {
88          String brand = request.getParameter("brand");
89          String model = request.getParameter("model");
90          String cyclinder = request.getParameter("cyclinder");
91          String price = request.getParameter("price");
92          Car newCar = new Car(brand, model, cyclinder, price);
```

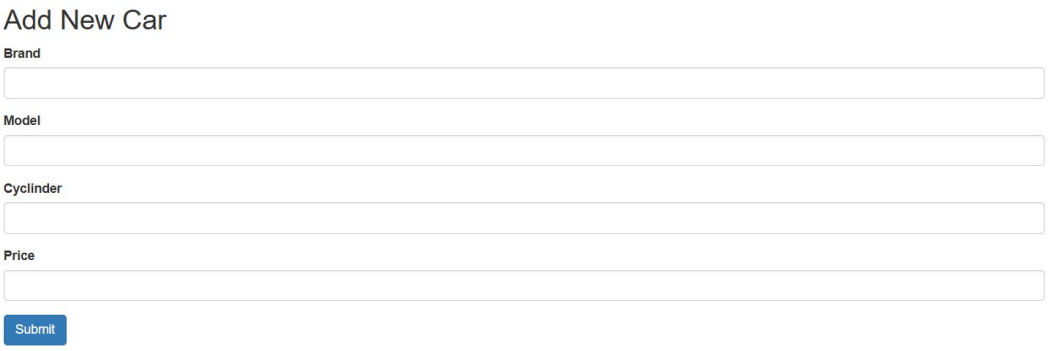
com.WEB.CarServlet > doGet >

```
92          Car newCar = new Car(brand, model, cyclinder, price);
93          carDAO.insertCar(newCar);
94          response.sendRedirect("list");
95      }
96
97      private void updateCar(HttpServletRequest request, HttpServletResponse response)
98              throws SQLException, IOException {
99          int car_id = Integer.parseInt(request.getParameter("car_id"));
100         String brand = request.getParameter("brand");
101         String model = request.getParameter("model");
102         String cyclinder = request.getParameter("cyclinder");
103         String price = request.getParameter("price");
104         Car car = new Car(car_id, brand, model, cyclinder, price);
105         carDAO.updateCar(car);
106         response.sendRedirect("list");
107     }
108
109     private void deleteCar(HttpServletRequest request, HttpServletResponse response)
110             throws SQLException, IOException {
111         int car_id = Integer.parseInt(request.getParameter("car_id"));
112         carDAO.deleteCar(car_id);
113         response.sendRedirect("list");
114     }
115 }
```

OUTPUT


localhost:8080/Car_Shop/list
Car List
ID
Brand
Model
Cyclinder
Price
Actions
1
Axia
2.0
300
30000.0
Edit
Delete
2
TOYOTA
hilux
200
1500.9
Edit
Delete
Add New Car


Edit Car
Brand
TOYOTA
Model
hilux
Cyclinder
200
Price
1500.9
Submit


Add New Car
Brand
Model
Cyclinder
Price
Submit