



**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:**

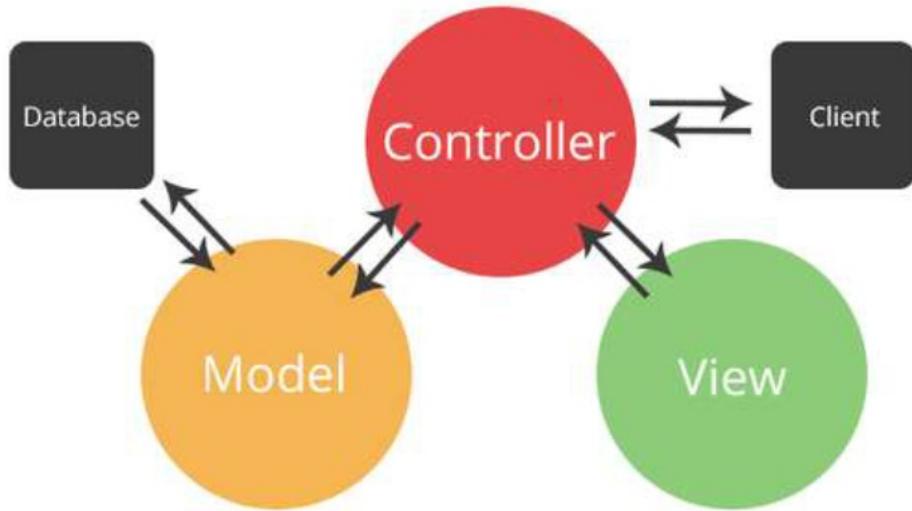
DR. MOHAMAD NOR HASSAN

**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 mengedarkan manual ini tanpa kebenaran rasmi daripada penulis.

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

### Instruction:

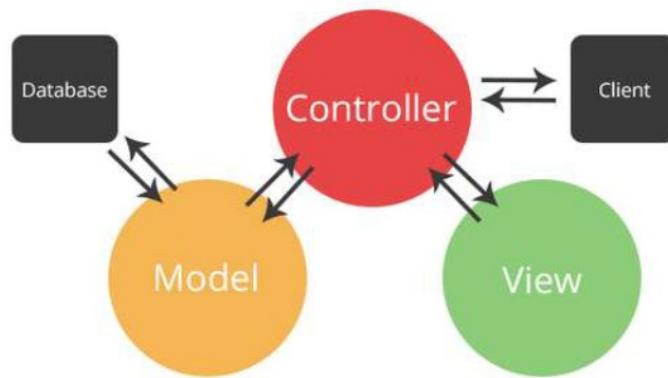
This laboratory manual is for use by the students of the Faculty of Ocean Engineering Technology and Informatics (FTKKI), Universiti Malaysia Terengganu only.

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 (v) 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

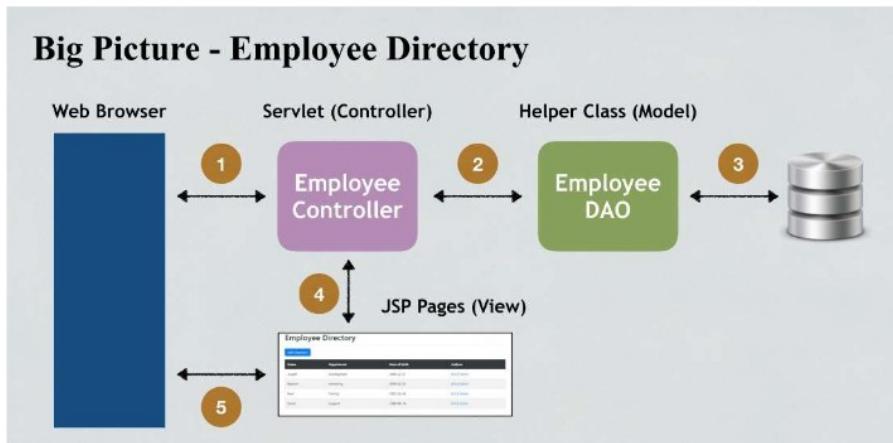
- Minimizes HTML code in Servlet no more: `out.println(...)` in Servlet code.
- Minimize Java business logic in JSPs no more large scriptlets in JSP code
- It separates the presentation layer from the business layer
- The Controller performs the action of invoking the Model and sending data to View
- 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.

```

1 CREATE DATABASE IF NOT EXISTS Company;
2 USE Company;
3
4 CREATE TABLE IF NOT EXISTS employees (
5   id INT NOT NULL AUTO_INCREMENT,
6   Name VARCHAR(60),
7   Email VARCHAR(50),
8   Position VARCHAR(15),
9   PRIMARY KEY (id)
10 )

```

**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

```
1  /*  
2   * To change this license header, choose License Headers in Project Properties.  
3   * To change this template file, choose Tools | Templates  
4   * and open the template in the editor.  
5   */  
6  package com.DAO;  
7  
8  import java.sql.Connection;  
9  import java.sql.DriverManager;  
10 import java.sql.PreparedStatement;  
11 import java.sql.ResultSet;  
12 import java.sql.SQLException;  
13 import java.util.ArrayList;  
14 import java.util.List;  
15  
16 import com.Model.Employee;  
17  
18 public class EmployeeDAO {  
19     Connection connection = null;  
20     private String jdbcURL = "jdbc:mysql://localhost:3306/company";  
21     private String jdbcUsername = "yusro";  
22     private String jdbcPassword = "admin";  
23  
24     private static final String INSERT_EMPLOYEES_SQL = "INSERT INTO employees (name, email, position) VALUES " +  
25         " (?, ?, ?);"  
26  
27     private static final String SELECT_EMPLOYEE_BY_ID = "select id,name,email,position from employees where id =?";  
28     private static final String SELECT_ALL_EMPLOYEES = "select * from employees";  
29     private static final String DELETE_EMPLOYEES_SQL = "delete from employees where id = ?";
```

```
30     private static final String UPDATE_EMPLOYEES_SQL = "update employees set name = ?,email= ?, position =? where id = ?";  
31  
32     public EmployeeDAO() {}  
33  
34     protected Connection getConnection() {  
35         Connection connection = null;  
36         try {  
37             Class.forName("com.mysql.jdbc.Driver");  
38             connection = DriverManager.getConnection(jdbcURL, jdbcUsername, jdbcPassword);  
39         } catch (SQLException e) {  
40             // TODO Auto-generated catch block  
41             e.printStackTrace();  
42         } catch (ClassNotFoundException e) {  
43             // TODO Auto-generated catch block  
44             e.printStackTrace();  
45         }  
46         return connection;  
47     }  
48  
49     public void insertEmployee(Employee employee) throws SQLException {  
50         System.out.println(INSERT_EMPLOYEES_SQL);  
51         // try-with-resource statement will auto close the connection.  
52         try (Connection connection = getConnection(); PreparedStatement preparedStatement =  
53             connection.prepareStatement(INSERT_EMPLOYEES_SQL)) {  
54             preparedStatement.setString(1, employee.getName());  
55             preparedStatement.setString(2, employee.getEmail());  
56             preparedStatement.setString(3, employee.getPosition());  
57             System.out.println(preparedStatement);  
58             preparedStatement.executeUpdate();  
59         } catch (SQLException e) {  
60             printSQLException(e);  
61         }  
62     }  
63  
64     public Employee selectEmployee(int id) {  
65         Employee employee = null;  
66         // Step 1: Establishing a Connection  
67         try (Connection connection = getConnection();  
68             // Step 2:Create a statement using connection object  
69             PreparedStatement preparedStatement = connection.prepareStatement(SELECT_EMPLOYEE_BY_ID);) {  
70             preparedStatement.setInt(1, id);  
71             System.out.println(preparedStatement);  
72             // Step 3: Execute the query or update query  
73             ResultSet rs = preparedStatement.executeQuery();  
74  
75             // Step 4: Process the ResultSet object.  
76             while (rs.next()) {  
77                 String name = rs.getString("name");  
78                 String email = rs.getString("email");  
79                 String position = rs.getString("position");  
80                 employee = new Employee(id, name, email, position);  
81             }  
82         } catch (SQLException e) {  
83             printSQLException(e);  
84         }  
85         return employee;  
86     }
```

```

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
93         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
117     public boolean deleteEmployee(int id) throws SQLException {
118         boolean rowDeleted;
119         try (Connection connection = getConnection(); PreparedStatement statement =
120             connection.prepareStatement(DELETE_EMPLOYEES_SQL)) {
121             statement.setInt(1, id);
122             rowDeleted = statement.executeUpdate() > 0;
123         }
124         return rowDeleted;
125     }
126
127     public boolean updateEmployee(Employee employee) throws SQLException {
128         boolean rowUpdated;
129         try (Connection connection = getConnection(); PreparedStatement statement =
130             connection.prepareStatement(UPDATE_EMPLOYEES_SQL)) {
131             statement.setString(1, employee.getName());
132             statement.setString(2, employee.getEmail());
133             statement.setString(3, employee.getPosition());
134             statement.setInt(4, employee.getId());
135
136             rowUpdated = statement.executeUpdate() > 0;
137         }
138         return rowUpdated;
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 = e.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

```
1  /*
2  * To change this license header, choose License Headers in Project Properties.
3  * To change this template file, choose Tools | Templates
4  * and open the template in the editor.
5  */
6 package com.Model;
7
8 public class Employee {
9     protected int id;
10    protected String name;
11    protected String email;
12    protected String position;
13
14    public Employee() {}
15
16    public Employee(String name, String email, String position) {
17        super();
18        this.name = name;
19        this.email = email;
20        this.position = position;
21    }
22
23    public Employee(int id, String name, String email, String position) {
24        super();
25        this.id = id;
26        this.name = name;
27        this.email = email;
28        this.position = position;
29    }
30}
```

```
30
31    public int getId() {
32        return id;
33    }
34    public void setId(int id) {
35        this.id = id;
36    }
37    public String getName() {
38        return name;
39    }
40    public void setName(String name) {
41        this.name = name;
42    }
43    public String getEmail() {
44        return email;
45    }
46    public void setEmail(String email) {
47        this.email = email;
48    }
49    public String getPosition() {
50        return position;
51    }
52    public void setPosition(String position) {
53        this.position = position;
54    }
55}
56
```

## EmployeeServlet.java

Name the package as com.WEB

```
1  /*
2  * To change this license header, choose License Headers in Project Properties.
3  * To change this template file, choose Tools | Templates
4  * and open the template in the editor.
5  */
6 package com.WEB;
7
8 import java.io.IOException;
9 import java.sql.SQLException;
10 import java.util.List;
11
12 import javax.servlet.RequestDispatcher;
13 import javax.servlet.ServletException;
14 import javax.servlet.annotation.WebServlet;
15 import javax.servlet.http.HttpServlet;
16 import javax.servlet.http.HttpServletRequest;
17 import javax.servlet.http.HttpServletResponse;
18
19 import com.DAO.EmployeeDAO;
20 import com.Model.Employee;
21
22 @WebServlet("/")
23 public class EmployeeServlet extends HttpServlet {
24     // private static final long serialVersionUID = 1L;
25     private EmployeeDAO employeeDAO;
26
27     public void init() {
28         employeeDAO = new EmployeeDAO();
29     }
30 }
```

```
31     protected void doPost(HttpServletRequest request, HttpServletResponse response)
32         throws ServletException, IOException {
33             doGet(request, response);
34         }
35
36     protected void doGet(HttpServletRequest request, HttpServletResponse response)
37         throws ServletException, IOException {
38         String action = request.getServletPath();
39
40         try {
41             switch (action) {
42                 case "/new":
43                     showNewForm(request, response);
44                     break;
45                 case "/insert":
46                     insertEmployee(request, response);
47                     break;
48                 case "/delete":
49                     deleteEmployee(request, response);
50                     break;
51                 case "/edit":
52                     showEditForm(request, response);
53                     break;
54                 case "/update":
55                     updateEmployee(request, response);
56                     break;
57                 default:
58                     listEmployee(request, response);
59                     break;
60             }
61         } catch (SQLException ex) {
62             throw new ServletException(ex);
63         }
64     }
65
66     private void listEmployee(HttpServletRequest request, HttpServletResponse response)
67         throws SQLException, IOException, ServletException {
68         List<Employee> listEmployee = employeeDAO.selectAllEmployees();
69         request.setAttribute("listEmployee", listEmployee);
70         RequestDispatcher dispatcher = request.getRequestDispatcher("employeeList.jsp");
71         dispatcher.forward(request, response);
72     }
73
74     private void showNewForm(HttpServletRequest request, HttpServletResponse response)
75         throws ServletException, IOException {
76         RequestDispatcher dispatcher = request.getRequestDispatcher("employeeForm.jsp");
77         dispatcher.forward(request, response);
78     }
79
80     private void showEditForm(HttpServletRequest request, HttpServletResponse response)
81         throws SQLException, ServletException, IOException {
82         int id = Integer.parseInt(request.getParameter("id"));
83         Employee existingEmployee = employeeDAO.selectEmployee(id);
84         RequestDispatcher dispatcher = request.getRequestDispatcher("employeeForm.jsp");
85         request.setAttribute("employee", existingEmployee);
86         dispatcher.forward(request, response);
87     }
88 }
```

```

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-gg0yR0iXcbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUhcW7x5JyoRxT2M2wLT" 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
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}">

```

```

31             <form action="insert" method="post">
32             </c:if>
33
34             <h2>
35                 <c:if test="${employee != null}">
36                     Edit Employee
37                 <c:if>
38                     <c:if test="${employee == null}">
39                         Add New Employee
40                     </c:if>
41             </h2>
42
43             <c:if test="${employee != null}">
44                 <input type="hidden" name="id" value=">" />
45             </c:if>
46
47             <fieldset class="form-group">
48                 <label>Employee Name</label> <input type="text" value="

```

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

```

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
4   <!DOCTYPE html>
5   <html>
6
7       <head>
8           <title>Employee Management Application</title>
9           <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css"
10               integrity="sha384-ggOvR0iXChMGv3Xipma34MD+dH/1fQ784j16cYi4JTQUOhcWr7x9JvoRxT2MZWIT" crossorigin="anonymous">
11
12       </head>
13
14       <body>
15
16           <header>
17               <nav class="navbar navbar-expand-md navbar-dark" style="background-color: tomato">
18                   <div>
19                       <a href="#" class="navbar-brand"> Employee Management App </a>
20                   </div>
21
22                   <ul class="navbar-nav">
23                       <li><a href="<%=request.getContextPath()%>/list" class="nav-link">Employees</a></li>
24                   </ul>
25               </nav>
26           <br>
27

```

```

27 |         <div class="row">
28 |             <!-- <div class="alert alert-success" *ngIf='message'>{{message}}</div> -->
29 |
30 |             <div class="container">
31 |                 <h3 class="text-center">List of Employees</h3>
32 |                 <hr>
33 |                 <div class="container text-left">
34 |                     <a href="<%=request.getContextPath()%>/new" class="btn btn-success">Add New Employee</a>
35 |                 <br>
36 |                 <table class="table table-bordered">
37 |                     <thead>
38 |                         <tr>
39 |                             <th>ID</th>
40 |                             <th>Name</th>
41 |                             <th>Email</th>
42 |                             <th>Position</th>
43 |                             <th>Actions</th>
44 |                         </tr>
45 |                     </thead>
46 |
47 |                     <tbody>
48 |                         <!-- for (Todo node: nodes) { -->
49 |                         <c:forEach var="employee" items="${listEmployee}">
50 |                             <tr>
51 |                                 <td>
52 |                                     <c:out value="${employee.id}" />
53 |                                 </td>
54 |                                 <td>
55 |                                     <c:out value="${employee.name}" />
56 |                                 </td>
57 |                                 <td>
58 |                                     <c:out value="${employee.email}" />
59 |                                 </td>
60 |                                 <td>
61 |                                     <c:out value="${employee.position}" />
62 |                                 </td>
63 |                                 <td>
64 |                                     <a href="Edit &ampnbsp&ampnbsp&ampnbsp
65 |                                     <a href="Delete</td>
66 |                             </tr>
67 |                         </c:forEach>
68 |                     </tbody>
69 |                 </table>
70 |             </div>
71 |         </div>
72 |     </body>
73 | </html>

```

#### 4. File error.jsp

```

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

```

#### 5. File index.jsp

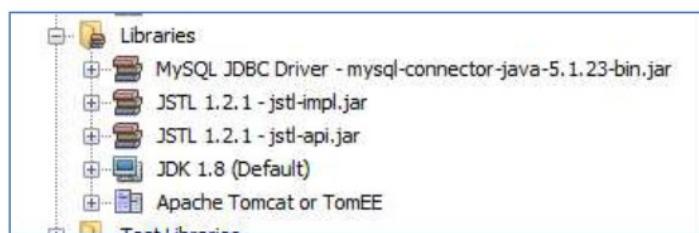
Contents of the Index.jsp:

```

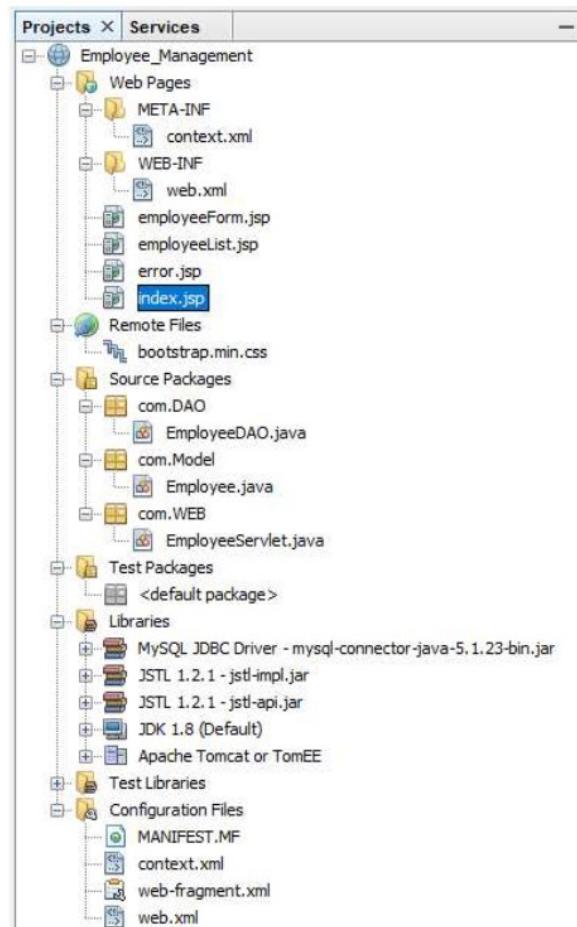
1  <%@page contentType="text/html" pageEncoding="UTF-8"%>
2  <!DOCTYPE html>
3  <html>
4      <head>
5          <title>User Management Application</title>
6          <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css"
7              integrity="sha384-9gOvR0iXChMo3Xipma34HD+dH/lfQ784/16cY/iJTOUhchW-7x9JvcRxT2NzwlT" crossorigin="anonymous">
8      </head>
9
10     <body>
11         <h1>Application MVC system for Employee Management</h1><br>
12
13         <ul>
14             <li> <a href="http://localhost:8080/Employee Management/list"> All Employee List </a></li>
15             <li> <a href="http://localhost:8080/Employee Management/new"> Add a New Employee </a></li>
16             <li> <a href="http://localhost:8080/Employee Management/list"> Edit Employee </a></li>
17         </ul>
18
19     </body>
20 </html>
21

```

## Step 5 - Add these libraries:



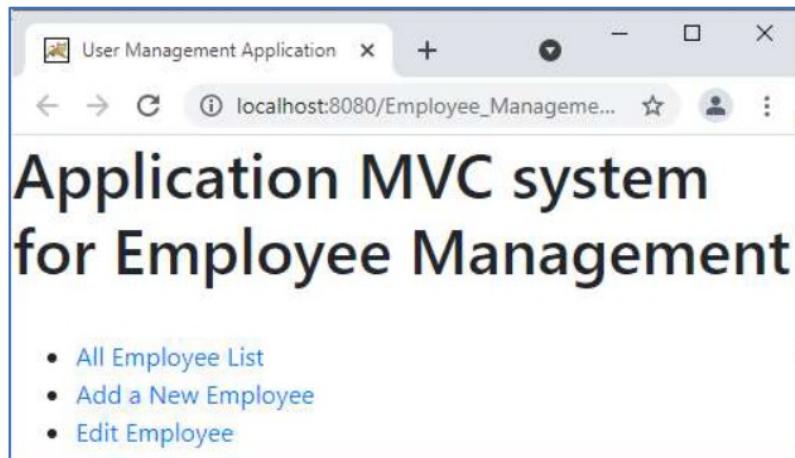
Finally, the Project schema should be like this:



## 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:



Employee Management App Employees

## List of Employees

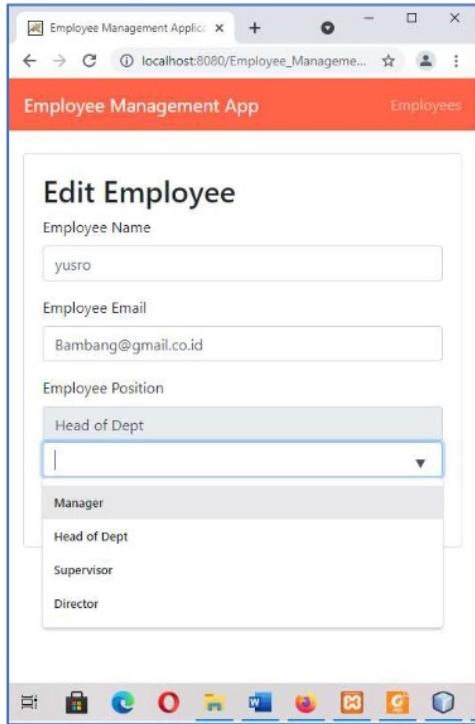
Add New Employee

ID	Name	Email	Position	Actions
1	muh yus	myus@project164.com	INA	Edit Delete
3	yus yus	yusyus@project164.com	Director	Edit Delete
4	muhyus	c@ict.net	Director	Edit Delete
5	yusro	Bambang@gmail.co.id	Head of Dept	Edit Delete

localhost:8080/Employee\_Management/edit?id=1

Type here to search

08:01 06/06/2021



## Exercise

Using this database schema, 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),
    Cylinder 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

```
<%--  
    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:if>  
  
                <fieldset class="form-group">  
                    <label>Employee Name</label><input type="text" value=""  
                        class="form-control" name="name" required="required">  
                </fieldset>  
  
                <fieldset class="form-group">  
                    <label>Employee Email</label><input type="text" value=""  
                        class="form-control" name="email">  
                </fieldset>  
62  
63  
64  
65                <fieldset class="form-group">  
66                    <label>Employee Position</label>  
67                    <input type="text" id="displayPosition" value="" class="form-control" readonly>  
68                    <input list="positionList" id="position" class="form-control" name="position" onchange="updatePosition()">  
69                    <datalist id="positionList">  
70                        <option value="Manager">  
71                        <option value="Head of Dept">  
72                        <option value="Supervisor">  
73                        <option value="Director">  
74                    </datalist>  
75                </fieldset>  
76  
77                    <button type="submit" class="btn btn-success">Save</button>  
78                </form>  
79            </div>  
80        </div>  
81    </body>  
82 </html>
```

## EmployeeList.jsp

```
1  <%--  
2   Document : EmployeeList  
3   Created on : 6 Jun 2024, 10:49:46 pm  
4   Author : rynaa  
5 --%>  
6  
7  <%@page contentType="text/html" pageEncoding="UTF-8"%>  
8  <%@taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>  
9  <!DOCTYPE html|  
0  <html>  
1  <head>  
2   <title>Employee Management Application</title>  
3   <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/css/bootstrap.min.css">  
4   <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.7.1/jquery.min.js"></script>  
5   <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/js/bootstrap.min.js"></script>  
6  </head>  
7  <body>  
8   <header>  
9    <nav class="navbar navbar-expand-md navbar-dark" style="background-color: tomato">  
0    <div>  
1     <a href="#" class="navbar-brand">Employee Management App </a>  
2    </div>  
3  
4    <ul class="navbar-nav">  
5     <li><a href="<%=request.getContextPath()%>/list" class="nav-link">Employees</a></li>  
6    </ul>  
7   </header>  
8   <br>  
9   <div class="row">  
0    <div class="container">  
1     <h3 class="text-center">List of Employees</h3>  
2  
32    <hr>  
33    <div class="container text-left">  
34     <a href="<%=request.getContextPath()%>/new" class="btn btn-success">Add New Employee</a>  
35    <br>  
36    <br>  
37    <table class="table table-bordered">  
38     <thead>  
39      <tr>  
40       <th>ID</th>  
41       <th>Name</th>  
42       <th>Email</th>  
43       <th>Position</th>  
44       <th>Actions</th>  
45      </tr>  
46     </thead>  
47  
48     <tbody>  
49      <c:forEach var="employee" items="${listEmployee}">  
50        <tr>  
51          <td>  
52            <c:out value="${employee.id}" />  
53          </td>  
54          <td>  
55            <c:out value="${employee.name}" />  
56          </td>  
57          <td>  
58            <c:out value="${employee.email}" />  
59          </td>  
60          <td>  
61            <c:out value="${employee.position}" />  
62          </td>  
63  
64          <td>  
65            <a href="edit?id=<c:out value="${employee.id}" />">Edit</a>        
66            <a href="delete?id=<c:out value="${employee.id}" />">Delete</a></td>  
67        </tr>  
68      </c:forEach>  
69    </tbody>  
70  </table>  
71  </div>  
72  </div>  
73  </body>  
74  </html>
```

### FileError.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

```
<%-->
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

```
1  /*
2   * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
3   * Click nbfs://nbhost/SystemFileSystem/Templates/JSP_Servlet/Servlet.java to edit this template
4   */
5  package com.DAO;
6
7  import java.sql.Connection;
8  import java.sql.DriverManager;
9  import java.sql.PreparedStatement;
10 import java.sql.ResultSet;
11 import java.sql.SQLException;
12 import java.util.ArrayList;
13 import java.util.List;
14
15 import com.Model.Employee;
16
17 public class EmployeeDAO {
18     private String jdbcURL = "jdbc:mysql://localhost:3306/company"; // Corrected URL
19     private String jdbcUsername = "root";
20     private String jdbcPassword = "admin";
21
22     private static final String INSERT_EMPLOYEES_SQL = "INSERT INTO employees (name, email, position) VALUES (?, ?, ?);";
23     private static final String SELECT_EMPLOYEE_BY_ID = "SELECT id, name, email, position FROM employees WHERE id = ?";
24     private static final String SELECT_ALL_EMPLOYEES = "SELECT * FROM employees";
25     private static final String DELETE_EMPLOYEES_SQL = "DELETE FROM employees WHERE id = ?;";
26     private static final String UPDATE_EMPLOYEES_SQL = "UPDATE employees SET name = ?, email = ?, position = ? WHERE id = ?;";
27
28     public EmployeeDAO() {}
29
30     protected Connection getConnection() {
31         Connection connection = null;
32
33         try {
34             Class.forName("com.mysql.cj.jdbc.Driver");
35             connection = DriverManager.getConnection(jdbcURL, jdbcUsername, jdbcPassword);
36             System.out.println("Database connected!");
37         } catch (SQLException e) {
38             e.printStackTrace();
39         } catch (ClassNotFoundException e) {
40             e.printStackTrace();
41         }
42         return connection;
43     }
44
45     public void insertEmployee(Employee employee) throws SQLException {
46         System.out.println(INSERT_EMPLOYEES_SQL);
47         try (Connection connection = getConnection()) {
48             PreparedStatement preparedStatement = connection.prepareStatement(INSERT_EMPLOYEES_SQL);
49             preparedStatement.setString(1, employee.getName());
50             preparedStatement.setString(2, employee.getEmail());
51             preparedStatement.setString(3, employee.getPosition());
52             System.out.println(preparedStatement);
53             preparedStatement.executeUpdate();
54         } catch (SQLException e) {
55             printSQLException(e);
56         }
57     }
58
59     public Employee selectEmployee(int id) {
60         Employee employee = null;
61         try (Connection connection = getConnection()) {
62             PreparedStatement preparedStatement = connection.prepareStatement(SELECT_EMPLOYEE_BY_ID);
63             preparedStatement.setInt(1, id);
64         }
```

```

63         System.out.println(preparedStatement);
64         ResultSet rs = preparedStatement.executeQuery();
65
66         while (rs.next()) {
67             String name = rs.getString("name");
68             String email = rs.getString("email");
69             String position = rs.getString("position");
70             employee = new Employee(id, name, email, position);
71         }
72     } catch (SQLException e) {
73         printSQLException(e);
74     }
75     return employee;
76 }
77
78 public List<Employee> selectAllEmployee() {
79     List<Employee> employees = new ArrayList<>();
80     try (Connection connection = getConnection();
81          PreparedStatement preparedStatement = connection.prepareStatement(SELECT_ALL_EMPLOYEES)) {
82         System.out.println(preparedStatement);
83         ResultSet rs = preparedStatement.executeQuery();
84
85         while (rs.next()) {
86             int id = rs.getInt("id");
87             String name = rs.getString("name");
88             String email = rs.getString("email");
89             String position = rs.getString("position");
90             employees.add(new Employee(id, name, email, position));
91         }
92     } catch (SQLException e) {
93         printSQLException(e);
94     }
95     return employees;
96 }
97
98 public boolean deleteEmployee(int id) throws SQLException {
99     boolean rowDeleted;
100    try (Connection connection = getConnection();
101        PreparedStatement statement = connection.prepareStatement(DELETE_EMPLOYEES_SQL)) {
102        statement.setInt(1, id);
103        rowDeleted = statement.executeUpdate() > 0;
104    }
105    return rowDeleted;
106 }
107
108 public boolean updateEmployee(Employee employee) throws SQLException {
109     boolean rowUpdated;
110     try (Connection connection = getConnection();
111          PreparedStatement statement = connection.prepareStatement(UPDATE_EMPLOYEES_SQL)) {
112         statement.setString(1, employee.getName());
113         statement.setString(2, employee.getEmail());
114         statement.setString(3, employee.getPosition());
115         statement.setInt(4, employee.getId());
116
117         rowUpdated = statement.executeUpdate() > 0;
118     }
119     return rowUpdated;
120 }
121
122 private void printSQLException(SQLException ex) {
123     for (Throwable e : ex) {
124         if (e instanceof SQLException) {
125             if (e instanceof SQLException) {
126                 e.printStackTrace(System.err);
127                 System.err.println("SQLState: " + ((SQLException) e).getSQLState());
128                 System.err.println("Error Code: " + ((SQLException) e).getErrorCode());
129                 System.err.println("Message: " + e.getMessage());
130                 Throwable t = e.getCause();
131                 while (t != null) {
132                     System.out.println("Cause: " + t);
133                     t = t.getCause();
134                 }
135             }
136         }
137     }
}

```

## Employee.java

```
1  /*
2   * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
3   * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
4   */
5  package com.Model;
6
7  public class Employee {
8      protected int id;
9      protected String name;
10     protected String email;
11     protected String position;
12
13     public Employee() {}
14
15     public Employee (String name, String email, String position) {
16         super();
17         this.name = name;
18         this.email = email;
19         this.position = position;
20     }
21
22     public Employee(int id, String name, String email, String position){
23         super();
24         this.id = id;
25         this.name = name;
26         this.email = email;
27         this.position = position;
28     }
29
30     public void setId(int id) {
31         this.id = id;
32     }
33
34     public void setName(String name) {
35         this.name = name;
36     }
37
38     public void setEmail(String email) {
39         this.email = email;
40     }
41
42     public void setPosition(String position) {
43         this.position = position;
44     }
45
46     public int getId() {
47         return id;
48     }
49
50     public String getName() {
51         return name;
52     }
53
54     public String getEmail() {
55         return email;
56     }
57
58     public String getPosition() {
59         return position;
60     }
61 }
```

## EmployeeServlet.java

```
1  /*
2   * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
3   * Click nbfs://nbhost/SystemFileSystem/Templates/JSP_Servlet/Servlet.java to edit this template
4   */
5  package com.WEB;
6
7  import java.io.IOException;
8  import java.sql.SQLException;
9  import java.util.List;
10
11 import java.io.PrintWriter;
12 import java.sql.Connection;
13 import java.sql.DriverManager;
14 import java.sql.PreparedStatement;
15 import jakarta.servlet.ServletException;
16 import jakarta.servlet.annotation.WebServlet;
17 import jakarta.servlet.http.HttpServlet;
18 import jakarta.servlet.http.HttpServletRequest;
19 import jakarta.servlet.http.HttpServletResponse;
20
21 import com.DAO.EmployeeDAO;
22 import com.Model.Employee;
23 import jakarta.servlet.RequestDispatcher;
24
25 @WebServlet("/")
26 public class EmployeeServlet extends HttpServlet {
27
28     private EmployeeDAO employeeDAO;
29
30     public void init() {
31         employeeDAO = new EmployeeDAO();
32     }
33
34     protected void doPost(HttpServletRequest request, HttpServletResponse response)
35             throws ServletException, IOException {
36         doGet(request, response);
37     }
38
39     protected void doGet(HttpServletRequest request, HttpServletResponse response)
40             throws ServletException, IOException {
41         String action = request.getServletPath();
42
43         try {
44             switch (action) {
45                 case "/new":
46                     showNewForm(request, response);
47                     break;
48                 case "/insert":
49                     insertEmployee(request, response);
50                     break;
51                 case "/delete":
52                     deleteEmployee(request, response);
53                     break;
54                 case "/edit":
55                     showEditForm(request, response);
56                     break;
57                 case "/update":
58                     updateEmployee(request, response);
59                     break;
60                 default:
61                     listEmployee(request, response);
62                     break;
63             }
64         } catch (Exception e) {
65             e.printStackTrace();
66         }
67     }
68
69     private void showNewForm(HttpServletRequest request, HttpServletResponse response)
70             throws ServletException, IOException {
71         RequestDispatcher dispatcher = request.getRequestDispatcher("newEmployee.jsp");
72         dispatcher.forward(request, response);
73     }
74
75     private void insertEmployee(HttpServletRequest request, HttpServletResponse response)
76             throws ServletException, IOException {
77         String name = request.getParameter("name");
78         String address = request.getParameter("address");
79         String phone = request.getParameter("phone");
80
81         Employee employee = new Employee(name, address, phone);
82
83         employeeDAO.insertEmployee(employee);
84
85         response.sendRedirect("list");
86     }
87
88     private void deleteEmployee(HttpServletRequest request, HttpServletResponse response)
89             throws ServletException, IOException {
90         String id = request.getParameter("id");
91
92         employeeDAO.deleteEmployee(id);
93
94         response.sendRedirect("list");
95     }
96
97     private void showEditForm(HttpServletRequest request, HttpServletResponse response)
98             throws ServletException, IOException {
99         String id = request.getParameter("id");
100
101        Employee employee = employeeDAO.getEmployee(id);
102
103        request.setAttribute("employee", employee);
104
105        RequestDispatcher dispatcher = request.getRequestDispatcher("editEmployee.jsp");
106        dispatcher.forward(request, response);
107    }
108
109    private void updateEmployee(HttpServletRequest request, HttpServletResponse response)
110            throws ServletException, IOException {
111        String id = request.getParameter("id");
112        String name = request.getParameter("name");
113        String address = request.getParameter("address");
114        String phone = request.getParameter("phone");
115
116        Employee employee = new Employee(id, name, address, phone);
117
118        employeeDAO.updateEmployee(employee);
119
120        response.sendRedirect("list");
121    }
122
123    private void listEmployee(HttpServletRequest request, HttpServletResponse response)
124            throws ServletException, IOException {
125        List<Employee> listEmployee = employeeDAO.listEmployee();
126
127        request.setAttribute("listEmployee", listEmployee);
128
129        RequestDispatcher dispatcher = request.getRequestDispatcher("listEmployee.jsp");
130        dispatcher.forward(request, response);
131    }
132}
```

```
63         }
64     } catch (SQLException ex) {
65         throw new ServletException(ex);
66     }
67 }
68
69 private void listEmployee (HttpServletRequest request, HttpServletResponse response)
70     throws SQLException, IOException, ServletException {
71     List <Employee> listEmployee = employeeDAO.selectAllEmployee();
72     request.setAttribute ("listEmployee", listEmployee);
73     RequestDispatcher dispatcher = request.getRequestDispatcher ("EmployeeList.jsp");
74     dispatcher.forward (request, response);
75 }
76
77 private void showNewForm (HttpServletRequest request, HttpServletResponse response)
78     throws ServletException, IOException {
79     RequestDispatcher dispatcher= request.getRequestDispatcher("EmployeeForm.jsp");
80     dispatcher. forward (request, response);
81 }
82
83 private void showEditForm (HttpServletRequest request, HttpServletResponse response)
84     throws SQLException, ServletException, IOException {
85     int id = Integer.parseInt(request.getParameter("id"));
86     Employee existingEmployee = employeeDAO.selectEmployee(id);
87     RequestDispatcher dispatcher = request.getRequestDispatcher ("EmployeeForm.jsp");
88     request.setAttribute ("employee", existingEmployee);
89     dispatcher.forward(request, response);
90 }
91
92 private void insertEmployee (HttpServletRequest request, HttpServletResponse response)
93     throws SQLException, IOException {
94     String name = request.getParameter("name");
95     String email = request.getParameter("email");
96     String position = request.getParameter("position");
97     Employee newEmployee = new Employee(name, email, position);
98     employeeDAO.insertEmployee (newEmployee);
99     response.sendRedirect ("list");
100 }
101
102 private void updateEmployee (HttpServletRequest request, HttpServletResponse response)
103     throws SQLException, IOException {
104     int id = Integer.parseInt(request.getParameter("id"));
105     String name= request.getParameter("name");
106     String email = request.getParameter("email");
107     String position = request.getParameter("position");
108
109     Employee employee = new Employee(id, name, email, position);
110     employeeDAO.updateEmployee(employee);
111     response.sendRedirect("list");
112 }
113
114 private void deleteEmployee (HttpServletRequest request, HttpServletResponse response)
115     throws SQLException, IOException {
116     int id = Integer.parseInt(request.getParameter("id"));
117     employeeDAO.deleteEmployee(id);
118     response.sendRedirect ("list");
119 }
120 }
```

## OUTPUT

The screenshot shows a web browser window titled "Employee Management App". The URL is "localhost:8080/Employee\_Management/list". The page displays a table titled "List of Employees" with two rows of data:

ID	Name	Email	Position	Actions
1	Abu	abu@gmail.com	Supervisor	Edit Delete
2	arina malek	arina@gmail.com	Director	Edit Delete

A green button labeled "Add New Employee" is located at the top left of the table.

The screenshot shows a web browser window titled "Employee Management App". The URL is "localhost:8080/Employee\_Management/edit?id=1". The page displays an "Edit Employee" form with the following fields:

- Employee Name:** Abu
- Employee Email:** abu@gmail.com
- Employee Position:** A dropdown menu currently set to "Supervisor", with other options available: Manager, Head of Dept, Supervisor, and Director.

## EXERCISE

### carForm.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>Cylinder</label>  
                        <input type="text" name="cylinder" class="form-control" value="${car != null ? car.cylinder : ''}" required>  
                    </div>  
                    <div class="form-group">  
                        <label>Price</label>  
                        <input type="text" name="price" class="form-control" value="${car != null ? car.price : ''}" required>  
                    </div>  
                    <button type="submit" class="btn btn-primary">Submit</button>  
                </div>  
            </form>  
        </div>  
    </body>  
</html>
```

### carIndex.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

```
<%--  
    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>Cylinder</th>  
                        <th>Price</th>  
                        <th>Actions</th>  
                    </tr>  
                </thead>  
                <tbody>  
                    <c:forEach var="car" items="${listCar}">  
                        <c:forEach var="car" items="${listCar}">  
                            <tr>  
                                <td>${car.car_id}</td>  
                                <td>${car.brand}</td>  
                                <td>${car.model}</td>  
                                <td>${car.cylinder}</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>  
                    </c:forEach>  
                </tbody>  
            </table>  
            <a href="new" class="btn btn-primary">Add New Car</a>  
        </div>  
    </body>  
</html>
```

## Error 1.jsp

```
1 <%--  
2     Document      : error_1  
3     Created on   : 14 Jun 2024, 1:52:03 am  
4     Author        : rynaa  
5 --%>  
6  
7 <%@page contentType="text/html" pageEncoding="UTF-8"%>  
8 <!DOCTYPE html>  
9 <html>  
10    <head>  
11        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">  
12        <title>Error page</title>  
13    </head>  
14    <body>  
15        <center>  
16            <h1>Error</h1>  
17            <h2><%=exception.getMessage()%><br/></h2>  
18        </center>  
19    </body>  
20 </html>
```

## carDAO.java

```
1 /*  
2  * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license  
3  * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template  
4 */  
5 package com.DAO;  
6  
7 import java.sql.Connection;  
8 import java.sql.DriverManager;  
9 import java.sql.PreparedStatement;  
10 import java.sql.ResultSet;  
11 import java.sql.SQLException;  
12 import java.util.ArrayList;  
13 import java.util.List;  
14  
15 import com.Model.Car;  
16  
17 public class CarDAO {  
18     private String ..... = "jdbc:mysql://localhost:3306/carshop";  
19     private String jdbcUsername = "root";  
20     private String ..... = "admin";  
21  
22     private static final String INSERT_CAR_SQL = "INSERT INTO CarPricelist (Brand, Model, Cyclinder, Price) VALUES (?, ?, ?, ?);";  
23     private static final String SELECT_CAR_BY_ID = "SELECT Car_id, Brand, Model, Cyclinder, Price FROM CarPricelist WHERE Car_id = ?";  
24     private static final String SELECT_ALL_CAR = "SELECT * FROM CarPricelist";  
25     private static final String DELETE_CAR_SQL = "DELETE FROM CarPricelist WHERE Car_id = ?";  
26     private static final String UPDATE_CAR_SQL = "UPDATE CarPricelist SET Brand = ?, Model = ?, Cyclinder = ?, Price = ? WHERE Car_id = ?";  
27  
28     public CarDAO() {}  
29  
30     protected Connection getConnection() {  
31         Connection connection = null;
```

```
31     connection = connection();
32     try {
33         Class.forName("com.mysql.cj.jdbc.Driver");
34         connection = DriverManager.getConnection(dbURL, jdbcUsername, jdbcPassword);
35     } catch (SQLException | ClassNotFoundException e) {
36         e.printStackTrace();
37     }
38     return connection;
39 }
40
41     public void insertCar(Car car) throws SQLException {
42         System.out.println(INSERT_CAR_SQL);
43         try (Connection connection = getConnection();)
44             PreparedStatement preparedStatement = connection.prepareStatement(INSERT_CAR_SQL)) {
45                 preparedStatement.setString(1, car.getBrand());
46                 preparedStatement.setString(2, car.getModel());
47                 preparedStatement.setString(3, car.getCylinder());
48                 preparedStatement.setString(4, car.getPrice());
49                 System.out.println(preparedStatement);
50                 preparedStatement.executeUpdate();
51             } catch (SQLException e) {
52                 printSQLException(e);
53             }
54     }
55
56     public Car selectCar(int car_id) {
57         Car car = null;
58         try (Connection connection = getConnection();)
59             PreparedStatement preparedStatement = connection.prepareStatement(SELECT_CAR_BY_ID)) {
60                 preparedStatement.setInt(1, car_id);
61                 System.out.println(preparedStatement);
62             ResultSet rs = preparedStatement.executeQuery();
63             while (rs.next()) {
64                 String brand = rs.getString("Brand");
65                 String model = rs.getString("Model");
66                 String cylinder = rs.getString("Cylinder");
67                 String price = rs.getString("Price");
68                 car = new Car(car_id, brand, model, cylinder, price);
69             }
70         } catch (SQLException e) {
71             printSQLException(e);
72         }
73         return car;
74     }
75
76     public List<Car> selectAllCar() {
77         List<Car> cars = new ArrayList<>();
78         try (Connection connection = getConnection();)
79             PreparedStatement preparedStatement = connection.prepareStatement(SELECT_ALL_CAR)) {
80                 System.out.println(preparedStatement);
81                 ResultSet rs = preparedStatement.executeQuery();
82                 while (rs.next()) {
83                     int car_id = rs.getInt("Car_id");
84                     String brand = rs.getString("Brand");
85                     String model = rs.getString("Model");
86                     String cylinder = rs.getString("Cylinder");
87                     String price = rs.getString("Price");
88                     cars.add(new Car(car_id, brand, model, cylinder, price));
89                 }
90             } catch (SQLException e) {
91                 printSQLException(e);
92             }
93         return cars;
94     }
95
96     public boolean deleteCar(int car_id) throws SQLException {
97         boolean rowDeleted;
98         try (Connection connection = getConnection();)
99             PreparedStatement statement = connection.prepareStatement(DELETE_CAR_SQL)) {
100                 statement.setInt(1, car_id);
101                 rowDeleted = statement.executeUpdate() > 0;
102             }
103         return rowDeleted;
104     }
105
106     public boolean updateCar(Car car) throws SQLException {
107         boolean rowUpdated;
108         try (Connection connection = getConnection();)
109             PreparedStatement statement = connection.prepareStatement(UPDATE_CAR_SQL)) {
110                 statement.setString(1, car.getBrand());
111                 statement.setString(2, car.getModel());
112                 statement.setString(3, car.getCylinder());
113                 statement.setString(4, car.getPrice());
114                 statement.setInt(5, car.getCar_id());
115                 rowUpdated = statement.executeUpdate() > 0;
116             }
117         return rowUpdated;
118     }
119
120     private void printSQLException(SQLException ex) {
121         for (Throwable e : ex) {

```

```
122     if (e instanceof SQLException) {
123         e.printStackTrace(System.err);
124         System.err.println("SQLState: " + ((SQLException) e).getSQLState());
125         System.err.println("Error Code: " + ((SQLException) e).getErrorCode());
126         System.err.println("Message: " + e.getMessage());
127         Throwable t = ex.getCause();
128         while (t != null) {
129             System.out.println("Cause: " + t);
130             t = t.getCause();
131         }
132     }
133 }
134 }
```

Car.java

```
1  /*
2   * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
3   * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
4   */
5  package com.Model;
6
7  /**
8   *
9   * @author User
10  */
11
12 public class Car {
13     private int Car_id;
14     private String Brand;
15     private String Model;
16     private String Cyclinder;
17     private String Price;
18
19     // Constructor for insert
20     public Car(String Brand, String Model, String Cyclinder, String Price) {
21         this.Brand = Brand;
22         this.Model = Model;
23         this.Cyclinder = Cyclinder;
24         this.Price = Price;
25     }
26
27     // Constructor for update and select by ID
28     public Car(int Car_id, String Brand, String Model, String Cyclinder, String Price) {
29         this.Car_id = Car_id;
30         this.Brand = Brand;
31         this.Model = Model;
32         this.Cyclinder = Cyclinder;
33         this.Price = Price;
34     }
35
36     // Getters and Setters
37     public int getCar_id() {
38         return Car_id;
39     }
40
41     public void setCar_id(int Car_id) {
42         this.Car_id = Car_id;
43     }
44
45     public String getBrand() {
46         return Brand;
47     }
48
49     public void setBrand(String Brand) {
50         this.Brand = Brand;
51     }
52
53     public String getModel() {
54         return Model;
55     }
56
57     public void setModel(String model) {
58         this.Model = Model;
59     }
60
61     public String getCyclinder() {
62 }
```

```
60
61     public String getCyclinder() {
62         return Cyclinder;
63     }
64
65     public void setCyclinder(String Cyclinder) {
66         this.Cyclinder = Cyclinder;
67     }
68
69     public String getPrice() {
70         return Price;
71     }
72
73     public void setPrice(String price) {
74         this.Price = Price;
75     }
76 }
```

### carServlet.java

```
1  /*
2  * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
3  * Click nbfs://nbhost/SystemFileSystem/Templates/JSP_Servlet/Servlet.java to edit this template
4  */
5 package com.WEB;
6
7 import java.io.IOException;
8 import java.sql.SQLException;
9 import java.util.List;
10 import jakarta.servlet.RequestDispatcher;
11 import jakarta.servlet.ServletException;
12 import jakarta.servlet.annotation.WebServlet;
13 import jakarta.servlet.http.HttpServlet;
14 import jakarta.servlet.http.HttpServletRequest;
15 import jakarta.servlet.http.HttpServletResponse;
16
17 import com.DAO.CarDAO;
18 import com.Model.Car;
19
20 @WebServlet("/")
21 public class CarServlet extends HttpServlet {
22
23     private CarDAO carDAO;
24
25     public void init() {
26         carDAO = new CarDAO();
27     }
28
29     protected void doPost(HttpServletRequest request, HttpServletResponse response)
30             throws ServletException, IOException {
31         doGet(request, response);
32     }
33
34     protected void doGet(HttpServletRequest request, HttpServletResponse response)
35             throws ServletException, IOException {
36         String action = request.getServletPath();
37         try {
38             switch (action) {
39                 case "/new":
40                     showNewForm(request, response);
41                     break;
42                 case "/insert":
43                     insertCar(request, response);
44                     break;
45                 case "/delete":
46                     deleteCar(request, response);
47                     break;
48                 case "/edit":
49                     showEditForm(request, response);
50                     break;
51                 case "/update":
52                     updateCar(request, response);
53                     break;
54                 default:
55                     listCar(request, response);
56                     break;
57             }
58         } catch (SQLException ex) {
59             throw new ServletException(ex);
60         }
61     }
62 }
```

```
62     private void listCar(HttpServletRequest request, HttpServletResponse response)
63         throws SQLException, IOException, ServletException {
64         List<Car> listCar = carDAO.selectAllCar();
65         request.setAttribute("listCar", listCar);
66         RequestDispatcher dispatcher = request.getRequestDispatcher("carList.jsp");
67         dispatcher.forward(request, response);
68     }
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);
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

ID	Brand	Model	Cylinder	Price	Actions
1	Axia	2.0	300	30000.0	<button>Edit</button> <button>Delete</button>
2	TOYOTA	hilux	200	1500.9	<button>Edit</button> <button>Delete</button>

[Add New Car](#)

### Edit Car

Brand

Model

Cylinder

Price

Submit

### Add New Car

Brand

Model

Cylinder

Price

Submit