UNIVERSITI MALAYSIA TERENGGANU

CSM3023

BACHELOR OF COMPUTER SCIENCE (MOBILE COMPUTING) WITH HONORS

LAB 8

SEMESTER II 2023/2024

Prepared for:

DR. MOHAMMAD NOR HASSAN

Prepared by:

HUSNA ZAHIRA BINTI RUZELI (S67554)

(K1)

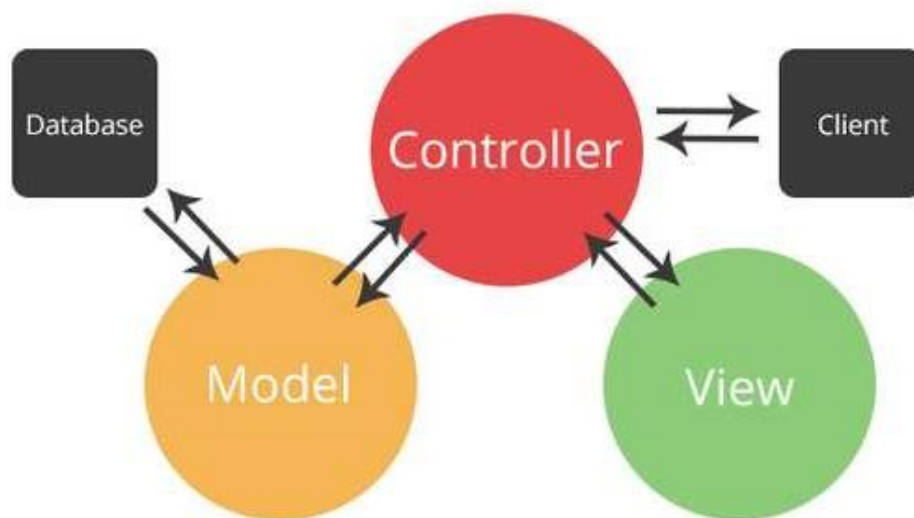# An MVC Example with Servlets and JSP

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.
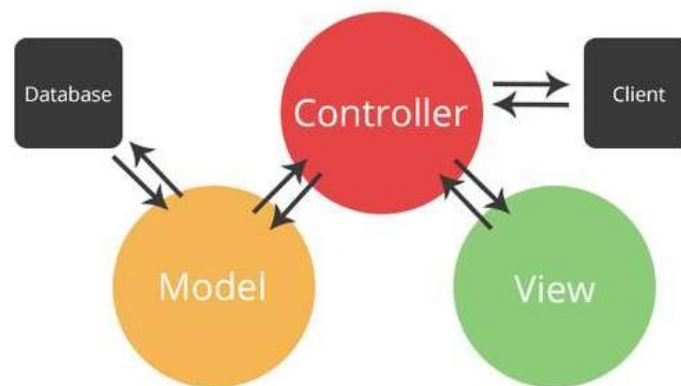

Benefits of MVC in JSP and Servlet Web Application

- 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.

Big Picture - Employee Directory
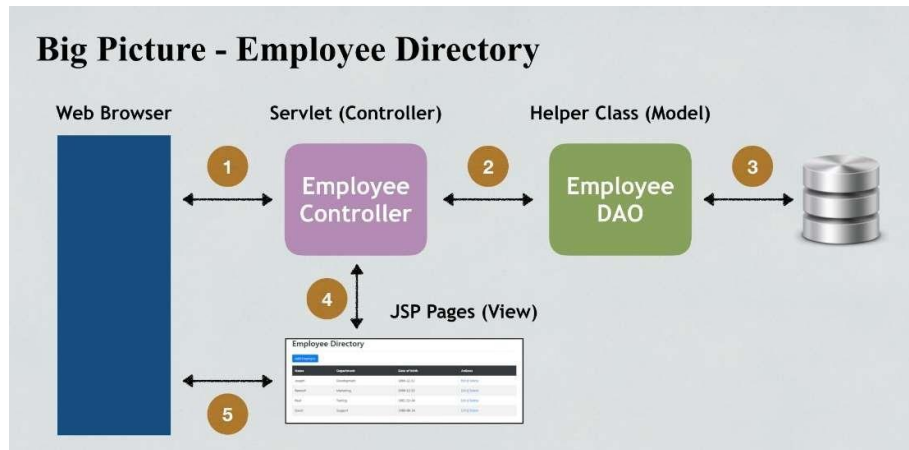
Step by step of Application development:

Step 1 - Create table employees in COMPANY database schema.

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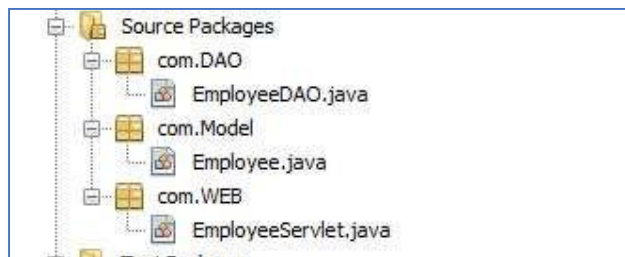
- EmployeeDAO.java (act as a Data Access Object (DAO) and to open /close database connection),
- Employee.java (act as a JavaBeans to represent business object), and
- EmployeeServlet.java (act to perform CRUD process)

EmployeeDAO.java

Name the package as com.DAO

```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;
    }
```

```java
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   }
```

```java
116   public boolean deleteEmployee(int id) throws SQLException {
117       boolean rowDeleted;
118       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;
128       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
```

```java
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;
    }
```

```java
    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);
    }
```

```java
 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)

```jsp
 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">
31      </c:if>
32
33      <h2>
34          <c:if test="${employee != null}">
35              Edit Employee
36          </c:if>
37          <c:if test="${employee == null}">
38              Add New Employee
39          </c:if>
40      </h2>
41
42      <c:if test="${employee != null}">
43          <input type="hidden" name="id" value="<c:out value='${employee.id}' />" />
44      </c:if>
45
46      <fieldset class="form-group">
47          <label>Employee Name</label> <input type="text" value="<c:out value='${employee.name}' />"
48                                               class="form-control" name="name" required="required">
49      </fieldset>
50
51      <fieldset class="form-group">
52          <label>Employee Email</label> <input type="text" value="<c:out value='${employee.email}' />"
53                                               class="form-control" name="email">
54      </fieldset>
55
```

```
56      <fieldset class="form-group">
57          <label>Employee Position</label>
58              <input type="text" value="<c:out value='${employee.position}' />" class="form-control" readonly >
59              <input list="positionList" id="position" class="form-control" name="position" >
60              <datalist id="positionList">
61                  <option value="Manager">
62                  <option value="Head of Dept">
63                  <option value="Supervisor">
64                  <option value="Director">
65              </datalist>
66      </fieldset>
67
68      <button type="submit" class="btn btn-success">Save</button>
69      </form>
70      </div>
71      </div>
72      </div>
73  </body>
74  </html>
```

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-ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T" crossorigin="anonymous">
11  </head>
12
13  <body>
14
15      <header>
16          <nav class="navbar navbar-expand-md navbar-dark" style="background-color: tomato">
17              <div>
18                  <a href="" class="navbar-brand"> Employee Management App </a>
19              </div>
20
21              <ul class="navbar-nav">
22                  <li><a href="<%=request.getContextPath()%>/list" class="nav-link">Employees</a></li>
23              </ul>
24          </nav>
25      </header>
26      <br>
27
```

```
27      <div class="row">
29          <!-- <div class="alert alert-success" *ngIf='message'>{{message}}</div> -->
30
            <div class="container">
                <h3 class="text-center">List of Employees</h3>
33              <hr>
                <div class="container text-left">
                    <a href="<%=request.getContextPath()%>/new" class="btn btn-success">Add New Employee</a>
36              </div>
37              <br>
                <table class="table table-bordered">
39                  <thead>
40                      <tr>
41                          <th>ID</th>
42                          <th>Name</th>
43                          <th>Email</th>
44                          <th>Position</th>
45                          <th>Actions</th>
46                      </tr>
47                  </thead>
48                  <tbody>
49                      <!--    for (Todo todo: todos) {  -->
50                      <c:forEach var="employee" items="${listEmployee}">
51                          <tr>
52                              <td>
53                                  <c:out value="${employee.id}" />
54                              </td>
55                              <td>
56                                  <c:out value="${employee.name}" />
57                              </td>
58                              <td>
59                                  <c:out value="${employee.email}" />
60                              </td>
61                              <td>
62                                  <c:out value="${employee.position}" />
63                              </td>
64                              <td><a href="edit?id=<c:out value='${employee.id}' />">Edit</a>     
65                                  <a href="delete?id=<c:out value='${employee.id}' />">Delete</a></td>
66                          </tr>
67                      </c:forEach>
68                  </tbody>
69              </table>
70          </div>
71      </div>
72  </body>
73
74  </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:

```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-ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T" 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:
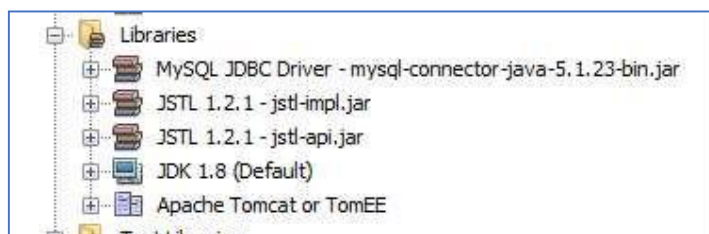
Finally, the Project schema should be like this:
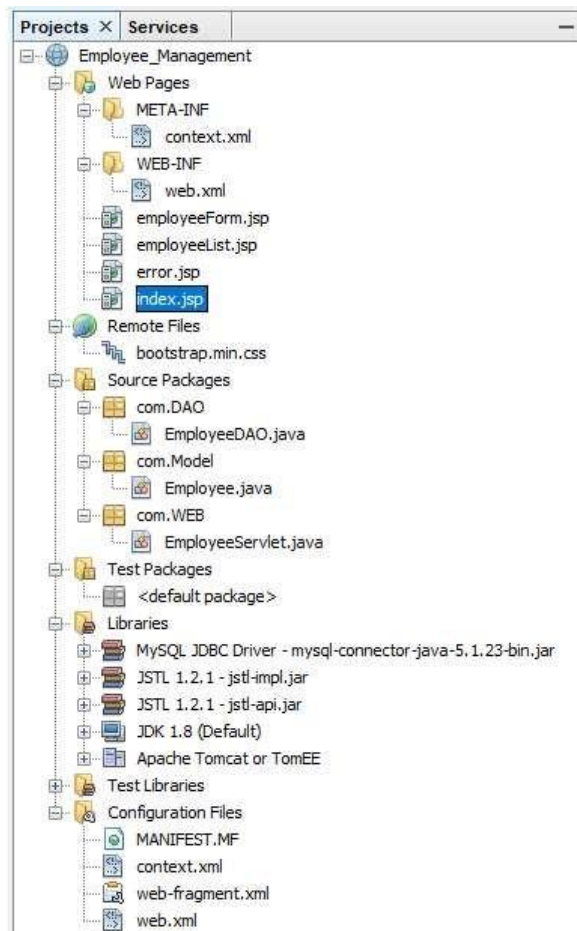
Step 6 - Running the program and try CRUD process:

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:
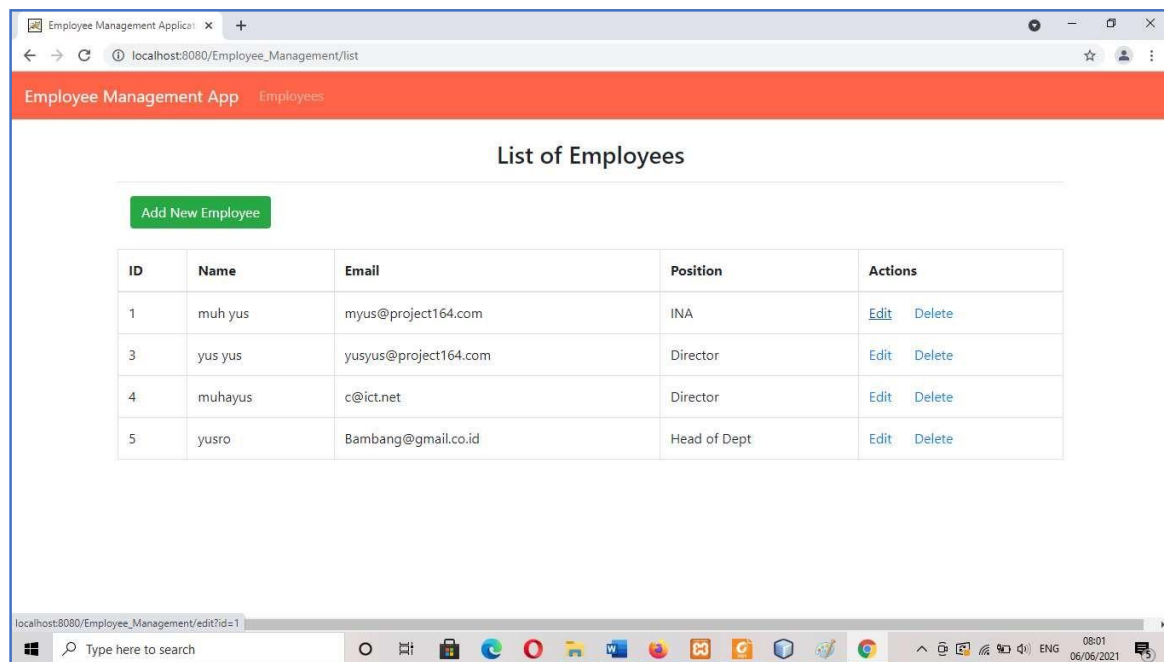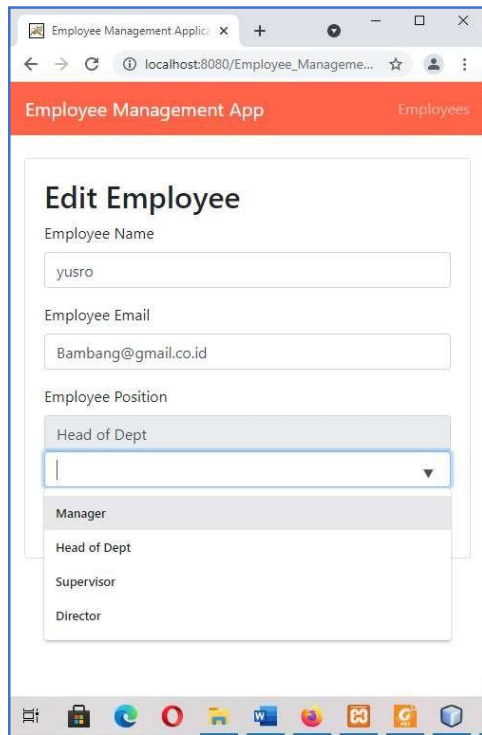
Code:
Employees Table(mySQL):

```sql
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  );
```

EmployeeDAO.java

```java
package com.DAO;

/**
 *
 * @author USER
 */
import java.sql.*;
import java.sql.SQLException;
import java.util.*;
import com.Model.Employee;

public class EmployeeDAO {
    Connection connection = null;
    private String jdbcURL = "jdbc:mysql://localhost:3306/company";
    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;
        // Step 1: Establishing a Connection
        try(Connection connection = getConnection();
            // Step 2: Create a statement using connection
            PreparedStatement preparedStatement = connection.prepareStatement(SELECT_EMPLOYEE_BY_ID);){
            preparedStatement.setInt(1, id);
            System.out.println(preparedStatement);
            ResultSet rs = preparedStatement.executeQuery();

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

```java
            return employee;
    }

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

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

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

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

    public boolean updateEmployee(Employee employee) throws SQLException{
        boolean rowUpdated;


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

            rowUpdated = statement.executeUpdate() > 0;
        }
        return rowUpdated;
    }

    private void printSQLException(SQLException ex){
        for(Throwable e: ex){
            if(e instanceof SQLException){
                e.printStackTrace(System.err);
                System.err.println("SQLState: " + ((SQLException) e).getSQLState());
                System.err.println("Error Code: " + ((SQLException) e).getErrorCode());
                System.err.println("Message: " + e.getMessage());
                Throwable t = ex.getCause();
                while(t != null){
                    System.out.println("Cause: " + t);
                    t = t.getCause();
                }
            }
        }
    }
}
```

Employee.java

```java
package com.Model;

/**
 *
 * @author USER
 */
public class Employee {
    protected int id;
    protected String name;
    protected String email;
    protected String position;

    public Employee() {}

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

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

    public int getId() {
        return id;
    }

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

    public 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

```java
package com.WEB;

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

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

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

```java
     */
    private EmployeeDAO employeeDAO;

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

    // <editor-fold defaultstate="collapsed" desc="HttpServlet methods. Click on the + sign on the left to edit the code.">
    /**
     * Handles the HTTP <code>GET</code> method.
     *
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error occurs
     * @throws IOException if an I/O error occurs
     */
    @Override
    protected void 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;
```

```java
    @Override
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        String action = request.getServletPath();

        try{
            switch(action){
                case "/new":
                    showNewForm(request, response);
                    break;
                case "/insert":
                    insertEmployee(request, response);
                    break;
                case "/delete":
                    deleteEmployee(request, response);
                    break;
                case "/edit":
                    showEditForm(request, response);
                    break;
                case "/update":
                    updateEmployee(request, response);
                    break;
                default:
                    listEmployee(request, response);
                    break;
            }
        }catch(SQLException ex){
            throw new ServletException(ex);
        }
    }

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


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

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

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

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

    private void updateEmployee(HttpServletRequest request, HttpServletResponse response)
            throws SQLException, IOException{
        int id = Integer.parseInt(request.getParameter("id"));
```

```java
    private void updateEmployee(HttpServletRequest request, HttpServletResponse response)
            throws SQLException, IOException{
        int id = Integer.parseInt(request.getParameter("id"));
        String name = request.getParameter("name");
        String email = request.getParameter("email");
        String position = request.getParameter("position");

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

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


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

    /**
     * Returns a short description of the servlet.
```

```java
        int id = Integer.parseInt(request.getParameter("id"));
        employeeDAO.deleteEmployee(id);
        response.sendRedirect("list");
    }


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

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

}
```

EmployeeForm.jsp

```jsp
<%--
    Document    : EmployeeForm
    Created on  : Jun 15, 2024, 9:53:39 PM
    Author      : S67554
--%>

<%@page contentType="text/html" pageEncoding="UTF-8"%>
<%@taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<!DOCTYPE html>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
        <title>Employee Management Application</title>
        <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css"
            integrity="sha384-ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T" crossorigin="anonymous">

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

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


            </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>
```

```jsp
                    <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 : Jun 15, 2024, 9:54:13 PM
    Author     : S67554
--%>

<%@page contentType="text/html" pageEncoding="UTF-8"%>
<%@taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<!DOCTYPE html>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
        <title>Employee Management System</title>
        <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css"
              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="container">
                <h3 class="container">List of Employees</h3>
                <hr>
                <div class="container text-left">
                    <a href="<%=request.getContextPath()%>/new" class="btn btn-success">Add New Employee</a>
```

```
                        <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="${employee.position}" />
                                    </td>
                                    <td><a href="edit?id=<c:out value='${employee.id}' />">Edit</a>     
                                        <a href="delete?id=<c:out value='${employee.id}' />">Delete</a>
                                    </td>
                                </tr>
                            </c:forEach>
                        </tbody>
                    </table>
                </div>
            </div>
```

Error.jsp

```
<%--
    Document    : error
    Created on  : Jun 15, 2024, 9:54:29 PM
    Author      : S67554
--%>

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

Index.jsp

```
<%--
    Document    : index
    Created on  : Jun 15, 2024, 9:54:41 PM
    Author      : S67554
--%>

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

Pom.xml

```xml
<dependencies>
    <dependency>
        <groupId>jakarta.platform</groupId>
        <artifactId>jakarta.jakartaee-api</artifactId>
        <version>${jakartaee}</version>
        <scope>provided</scope>
    </dependency>
    <!-- https://mvnrepository.com/artifact/com.mysql/mysql-connector-j -->
    <dependency>
        <groupId>com.mysql</groupId>
        <artifactId>mysql-connector-j</artifactId>
        <version>8.3.0</version>
    </dependency>
    <dependency>
        <groupId>jakarta.servlet.jsp.jstl</groupId>
        <artifactId>jakarta.servlet.jsp.jstl-api</artifactId>
        <version>2.0.0</version>
    </dependency>
    <dependency>
        <groupId>org.glassfish.web</groupId>
        <artifactId>jakarta.servlet.jsp.jstl</artifactId>
        <version>2.0.0</version>
    </dependency>
</dependencies>
```

Output:

Application MVC system for Employee Management

- All Employee List
- Add a New Employee
- Edit Employee

All Employee List:

Employee Management App	Employees

List of Employees

Add New Employee

ID	Name	Email	Position	Actions
2	Husna Zahira	husnazahira03@gmail.com	Manager	Edit Delete
3	Bidin Jamal	jamal991@gmail.com	Head of Dept	Edit Delete

Add New Employee:

Add New Employee

Employee Name

Employee Email

Employee Position

Save

Edit Employee:

List of Employees

Add New Employee

ID	Name	Email	Position	Actions
2	Husna Zahira	husnazahira03@gmail.com	Manager	Edit Delete
3	Bidin Jamal	jamal991@gmail.com	Head of Dept	Edit Delete

Add New Record:

Add New Employee

Employee Name

Husna Zahira

Employee Email

husnazahira03@gmail.com

Employee Position

Manager

Manager

Save

List of Employees

Add New Employee

ID	Name	Email	Position	Actions
1	Husna Zahira	husnazahira03@gmail.com	Manager	Edit Delete

Result Grid | 🔢 🔃 Filter Rows: [   ] | Edit: ✏️

	id	Name	Email	Position
▶	1	Husna Zahira	husnazahira03@gmail.com	Manager
*	NULL	NULL	NULL	NULL

Edit Record:

List of Employees

Add New Employee

ID	Name	Email	Position	Actions
1	Husna Zahira	husnazahira03@gmail.com	Manager	Edit Delete

Edit Employee

Employee Name

Husna Zahira Ruzeli

Employee Email

husnazahira03@gmail.com

Employee Position

Supervisor

Supervisor

Save

List of Employees

Add New Employee

ID	Name	Email	Position	Actions
1	Husna Zahira Ruzeli	husnazahira03@gmail.com	Supervisor	Edit Delete

	id	Name	Email	Position
▶	1	Husna Zahira Ruzeli	husnazahira03@gmail.com	Supervisor
*	NULL	NULL	NULL	NULL

Delete Record:

Employee Management App   Employees

List of Employees

Add New Employee

ID	Name	Email	Position	Actions
1	Husna Zahira Ruzeli	husnazahira03@gmail.com	Supervisor	Edit Delete

Employee Management App   Employees

List of Employees

Add New Employee

ID	Name	Email	Position	Actions

Result Grid | Filter Rows:

	id	Name	Email	Position
*	NULL	NULL	NULL	NULL

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

Code:
Carshop Table(mySQL):

```
 1 ●    CREATE DATABASE if not EXISTS carshop;
 2 ●    USE carshop;
 3
 4 ● ⊖  CREATE TABLE if not EXISTS CarPricelist (
 5          Car_id INT NOT NULL AUTO_INCREMENT,
 6          Brand VARCHAR(15),
 7          Model VARCHAR(30),
 8          Cylinder INT,
 9          Price DOUBLE,
10          PRIMARY KEY (Car_id)
11      );
```

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.controller;

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

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

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


     * @throws IOException if an I/O error occurs
     */
    private CarDAO carDAO;

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

    // <editor-fold defaultstate="collapsed" desc="HttpServlet methods. Click on the + sign on the left to edit the code.">
    /**
     * Handles the HTTP <code>GET</code> method.
     *
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error occurs
     * @throws IOException if an I/O error occurs
     */
    @Override
    protected void 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);
```

```java
        @Override
        protected void doGet(HttpServletRequest request, HttpServletResponse response)
                throws ServletException, IOException {
            String action = request.getServletPath();

            try{
                switch(action){
                    case "/new":
                        showNewForm(request, response);
                        break;
                    case "/insert":
                        insertCar(request, response);
                        break;
                    case "/delete":
                        deleteCar(request, response);
                        break;
                    case "/edit":
                        showEditForm(request, response);
                        break;
                    case "/update":
                        updateCar(request, response);
                        break;
                    default:
                        listCar(request, response);
                        break;
                }
            }catch(SQLException ex){
                throw new ServletException(ex);
            }
        }

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

CarDAO.java

```java
package com.DAO;

/**
 *
 * @author USER
 */
import java.sql.*;
import java.sql.SQLException;
import java.util.*;
import com.model.Car;

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

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

    public CarDAO(){}

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

```java
            }
            return connection;
        }

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

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

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

```java
                double price = rs.getDouble("price");
                car = new Car(id, brand, model, cylinder, price);
            }
        }catch (SQLException e){
            printSQLException(e);
            }
        return car;
    }

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

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

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

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

```java
            try(Connection connection = getConnection(); PreparedStatement statement =
                    connection.prepareStatement(DELETE_CARS_SQL);){
                statement.setInt(1, id);
                rowDeleted = statement.executeUpdate() > 0;
            }
            return rowDeleted;
    }

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

            rowUpdated = statement.executeUpdate() > 0;
        }
        return rowUpdated;
    }

    private void printSQLException(SQLException ex){
        for(Throwable e: ex){
            if(e instanceof SQLException){
                e.printStackTrace(System.err);
                System.err.println("SQLState: " + ((SQLException) e).getSQLState());
                System.err.println("Error Code: " + ((SQLException) e).getErrorCode());
                System.err.println("Message: " + e.getMessage());
                Throwable t = ex.getCause();
                while(t != null){
                    System.out.println("Cause: " + t);
                    t = t.getCause();
                }
            }
        }
    }
```

Car.java

```java
 */
package com.model;

/**
 *
 * @author USER
 */
public class Car {
    protected int car_id;
    protected String brand;
    protected String model;
    protected int cylinder;
    protected double price;

    public Car(){}

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

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


    public int getCar_id() {
        return car_id;
```

```java
    public int getCar_id() {
        return car_id;
    }

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

    public String getBrand() {
        return brand;
    }

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

    public String getModel() {
        return model;
    }

    public void setModel(String model) {
        this.model = model;
    }

    public int getCylinder() {
        return cylinder;
    }

    public void setCylinder(int cylinder) {
        this.cylinder = cylinder;
    }

    public double getPrice() {
        return price;
    }
```

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

    public String getModel() {
        return model;
    }

    public void setModel(String model) {
        this.model = model;
    }

    public int getCylinder() {
        return cylinder;
    }

    public void setCylinder(int cylinder) {
        this.cylinder = cylinder;
    }

    public double getPrice() {
        return price;
    }

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

CarForm.jsp

```jsp
<%--
    Document   : CarForm
    Created on : Jun 16, 2024, 12:21:54 AM
    Author     : S67554
--%>

<%@page contentType="text/html" pageEncoding="UTF-8"%>
<%@taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<!DOCTYPE html>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
        <title>Car Shop Management Application</title>
        <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css"
              integrity="sha384-ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T" crossorigin="anonymous">

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

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


                </nav>
            </header>
            <br>
            <div class="container col-md-5">
                <div class="card">
                    <div class="card-body">
                        <c:if test="${car != null}">
                            <form action="update" method="post">
                        </c:if>
                        <c:if test="${car == null}">
                            <form action="insert" method="post">
                        </c:if>

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

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

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

```
<datalist id="carList">
    <option value="Proton">
    <option value="Perodua">
    <option value="Toyota">
    <option value="Honda">
    <option value="Nissan">
    <option value="Ford">
    <option value="Hyundai">
    <option value="Kia">
</datalist>
</fieldset>

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

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

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

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

CarList.jsp

```
<%--
    Document   : CarList
    Created on : Jun 16, 2024, 12:22:04 AM
    Author     : S67554
--%>

<%@page contentType="text/html" pageEncoding="UTF-8"%>
<%@taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<!DOCTYPE html>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
        <title>Car Shop Management Application</title>
        <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css"
              integrity="sha384-ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T" crossorigin="anonymous">
    </head>
    <body>

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

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

        <div class="row">
            <div class="container">
                <h3 class="container">List of Cars</h3>
                <hr>
                <div class="container text-left">
                    <a href="<%=request.getContextPath()%>/new" class="btn btn-success">Add New Car</a>
```

```jsp
                <a href="<%=request.getContextPath()%>/new" class="btn btn-success">Add New Car</a>
            </div>
            <br>
            <table class="table table-bordered">
                <thead>
                    <tr>
                        <th>ID</th>
                        <th>Brand</th>
                        <th>Model</th>
                        <th>Engine Cylinder</th>
                        <th>Price</th>
                        <th>Actions</th>
                    </tr>
                </thead>
                <tbody>
                    <c:forEach var="car" items="${listCar}">
                        <tr>
                            <td>
                                <c:out value="${car.car_id}" />
                            </td>
                            <td>
                                <c:out value="${car.brand}" />
                            </td>
                            <td>
                                <c:out value="${car.model}" />
                            </td>
                            <td>
                                <c:out value="${car.cylinder}" />
                            </td>
                            <td>
                                <c:out value="${car.price}" />
                            </td>
                            <td><a href="edit?car_id=<c:out value='${car.car_id}' />">Edit</a>     
                                <a href="delete?car_id=<c:out value='${car.car_id}' />">Delete</a>
                            </td>
                        </tr>
                    </c:forEach>



                        <th>Engine Cylinder</th>
                        <th>Price</th>
                        <th>Actions</th>
                    </tr>
                </thead>
                <tbody>
                    <c:forEach var="car" items="${listCar}">
                        <tr>
                            <td>
                                <c:out value="${car.car_id}" />
                            </td>
                            <td>
                                <c:out value="${car.brand}" />
                            </td>
                            <td>
                                <c:out value="${car.model}" />
                            </td>
                            <td>
                                <c:out value="${car.cylinder}" />
                            </td>
                            <td>
                                <c:out value="${car.price}" />
                            </td>
                            <td><a href="edit?car_id=<c:out value='${car.car_id}' />">Edit</a>     
                                <a href="delete?car_id=<c:out value='${car.car_id}' />">Delete</a>
                            </td>
                        </tr>
                    </c:forEach>
                </tbody>
            </table>
        </div>
    </div>
</body>
</html>
```

Error.jsp

```jsp
<%--
    Document    : error
    Created on  : Jun 16, 2024, 12:22:12 AM
    Author      : S67554
--%>

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

Index.jsp

```jsp
<%--
    Document    : index
    Created on  : Jun 16, 2024, 12:22:20 AM
    Author      : S67554
--%>

<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
        <title>Car Shop Management Application</title>
        <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css"
              integrity="sha384-ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T" crossorigin="anonymous">
    </head>
    <body>
        <h1>Car Shop Management System</h1><br>

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

Output:

Car Shop Management System

- All Car List
- Add a New Car
- Edit Car

All Car List:

List of Cars

Add New Car

ID	Brand	Model	Engine Cylinder	Price	Actions
2	Perodua	Myvi	3	45000.0	Edit Delete
3	Proton	Saga	4	60000.0	Edit Delete

Add New Car:

Add New Car

Car Brand

Car Model

Engine Cylinder

Car Price

Save

Edit Car:

List of Cars

Add New Car

ID	Brand	Model	Engine Cylinder	Price	Actions
2	Perodua	Myvi	3	45000.0	Edit Delete
3	Proton	Saga	4	60000.0	Edit Delete

Result Grid | Filter Rows:

	Car_id	Brand	Model	Cylinder	Price
▶	2	Perodua	Myvi	3	45000
	3	Proton	Saga	4	60000
*	NULL	NULL	NULL	NULL	NULL