LAB 8

An MVC Example with Servlets and JSP

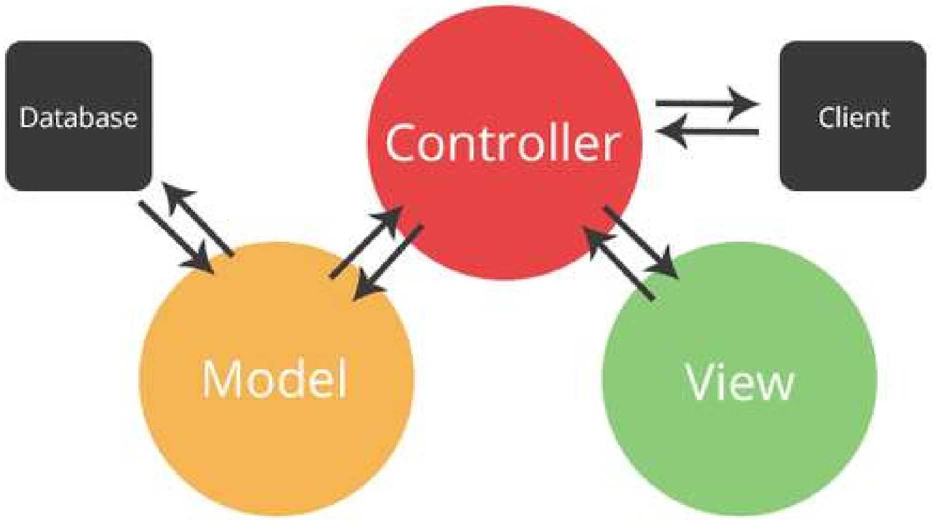


Table of Contents

Arahan:

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

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

Instruction:

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

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

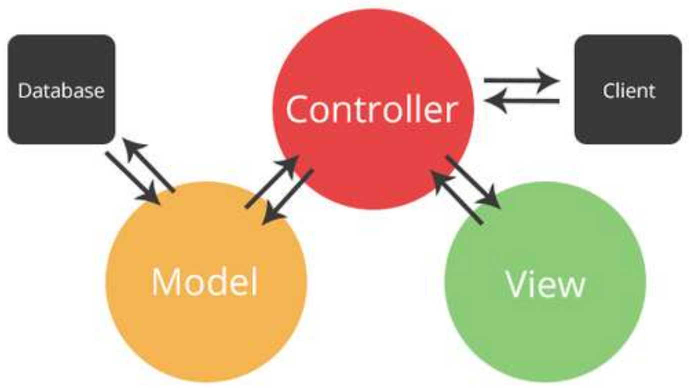
Creating MVC Database Web Application in JSP and Servlets – for Create, Read, Update, Delete

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

- Model - Model represents an object or JAVA POJO carrying data. It can also have logic to update controller if its data changes.

- View - View represents the visualization of the data that model contains.

- Controller - Controller acts on both model and view. It controls the data flow into model object and updates the view whenever data changes. It keeps view and model separate.



Benefits of MVC in JSP and Servlet Web Application

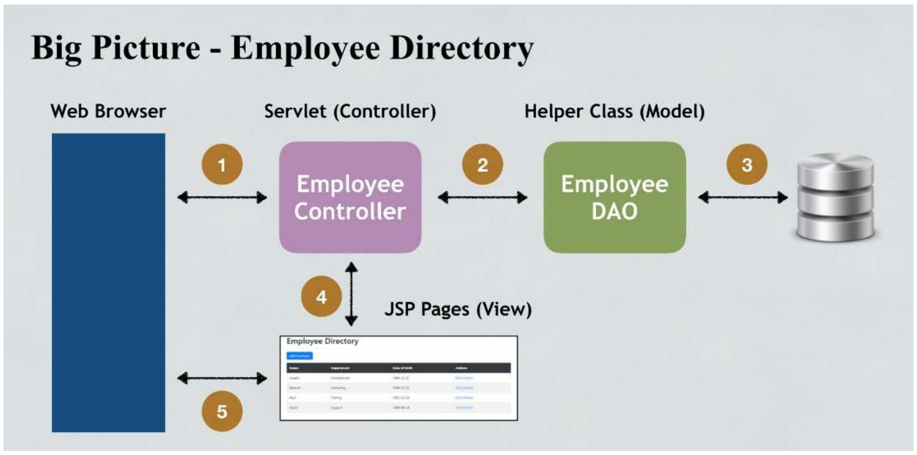
- Minimizes HTML code in Servlet no more: out.println(…) in Servlet code.
- Minimize Java business logic in JSPs no more large scriptlets in JSP code
- It separates the presentation layer from the business layer
- The Controller performs the action of invoking the Model and sending data to View
- The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

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

Steps Involved in the Application

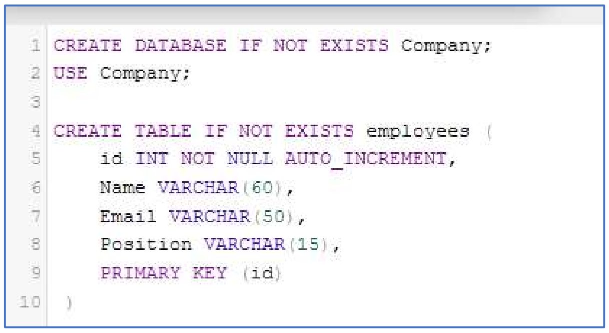
Basically, there are 4 main steps involved in this application that are given below:

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



Step by step of Application development:

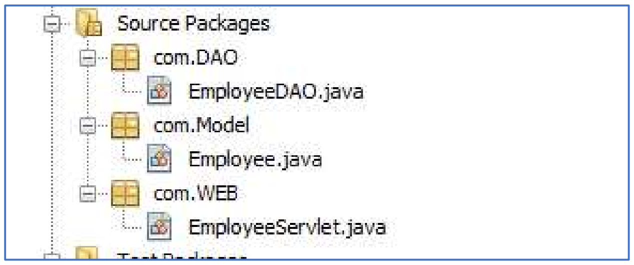
Step 1 - Create table employees in COMPANY database schema.



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

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

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

Step 3 - Create three Java class that representing :

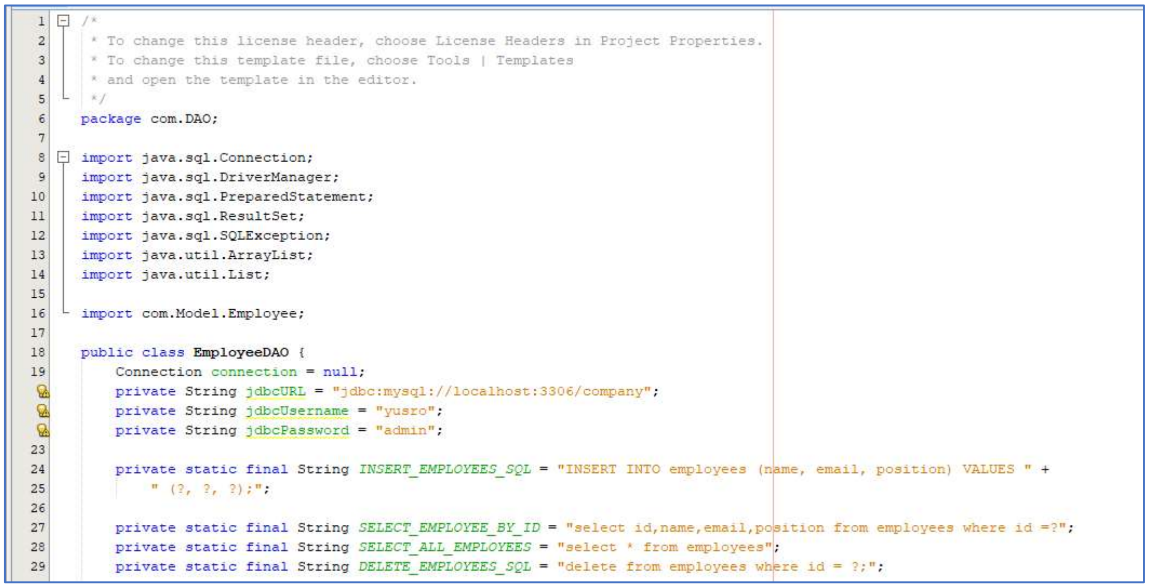


```
Source Packages
    com.DAO
        EmployeeDAO.java
    com.Model
        Employee.java
    com.WEB
        EmployeeServlet.java
```

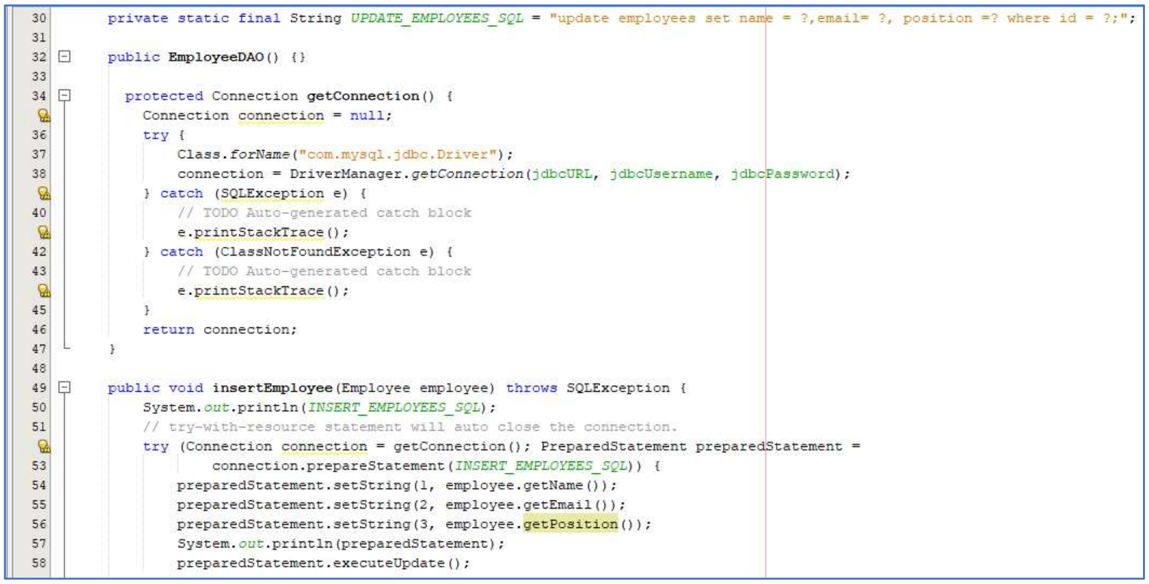
- EmployeeDAO.java (act as a Data Access Object (DAO) and to open /close database connection),
- Employee.java (act as a JavaBeans to represent business object), and
- EmployeeServlet.java (act to perform CRUD process)

EmployeeDAO.java

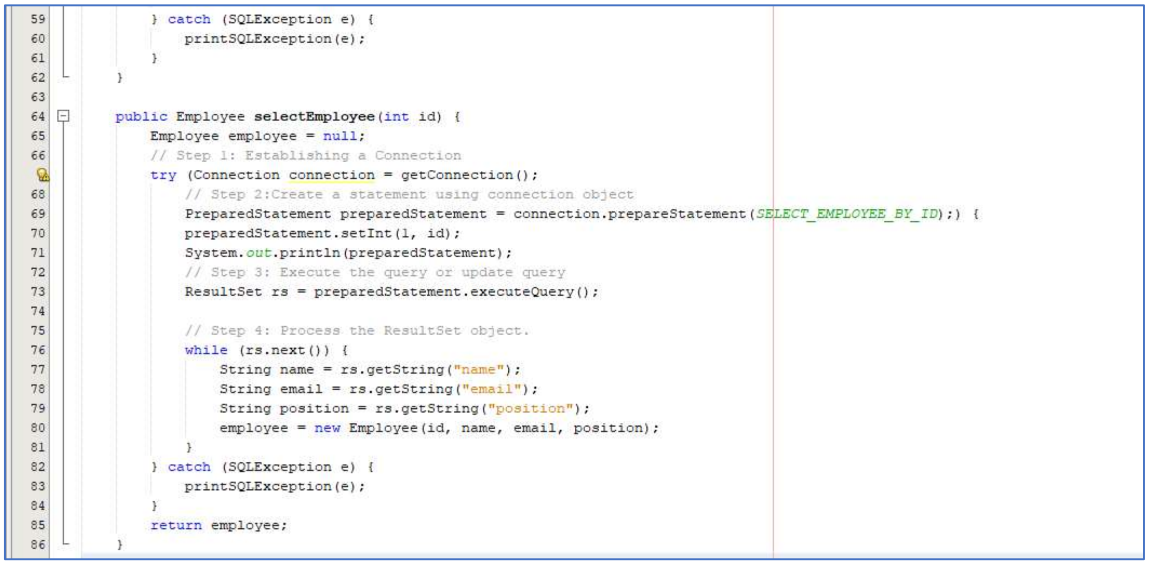
Name the package as com.DAO



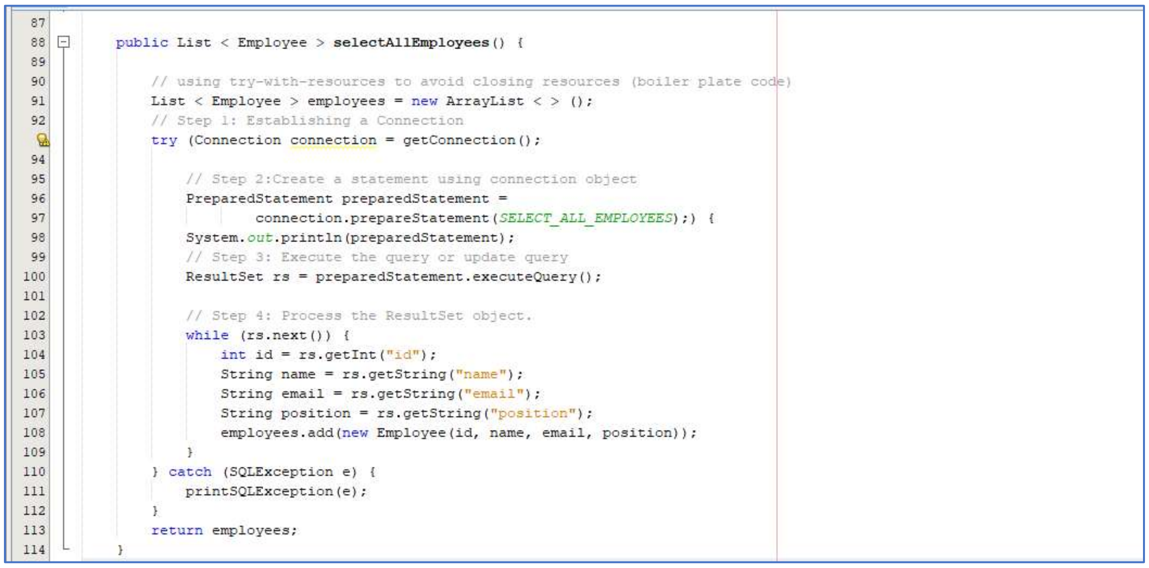
```java
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package com.DAO;

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 = ?;";
```



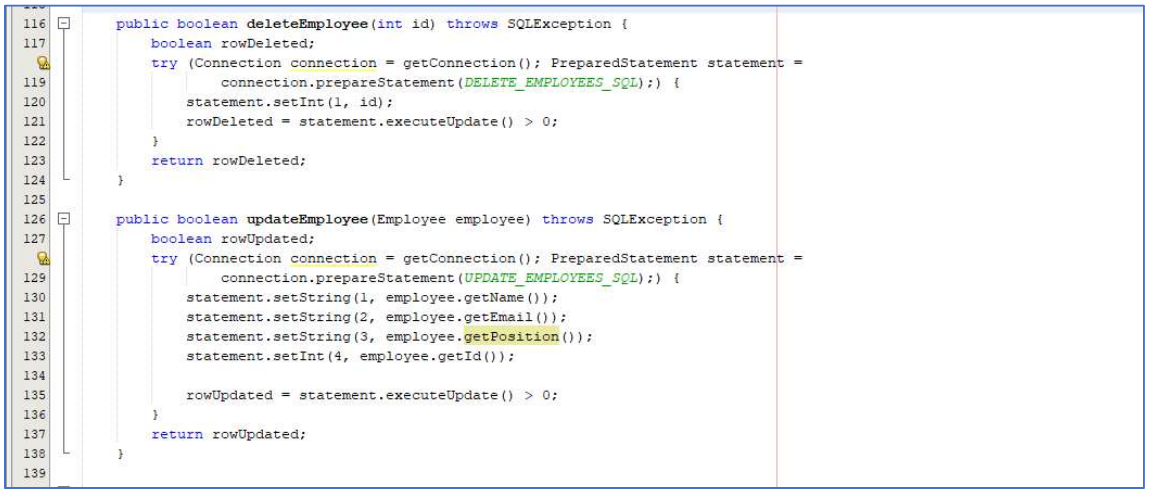
```java
    private static final String UPDATE_EMPLOYEES_SQL = "update employees set name = ?,email= ?, position =? where id = ?;";

    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();
```



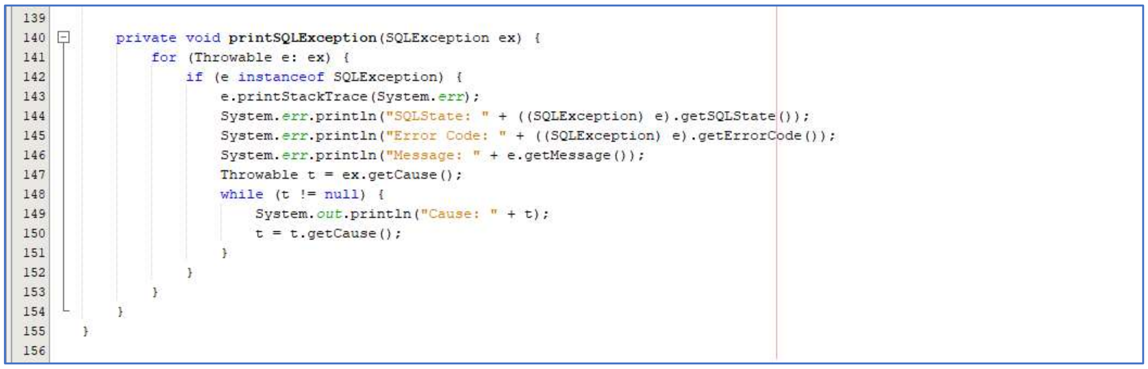
```java
        } catch (SQLException e) {
            printSQLException(e);
        }
    }

    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

    public List < Employee > selectAllEmployees() {

        // using try-with-resources to avoid closing resources (boiler plate code)
        List < Employee > employees = new ArrayList < > ();
        // Step 1: Establishing a Connection
        try (Connection connection = getConnection();

            // Step 2:Create a statement using connection object
            PreparedStatement preparedStatement =
                    connection.prepareStatement(SELECT_ALL_EMPLOYEES);) {
            System.out.println(preparedStatement);
            // Step 3: Execute the query or update query
            ResultSet rs = preparedStatement.executeQuery();

            // Step 4: Process the ResultSet object.
            while (rs.next()) {
                int id = rs.getInt("id");
                String name = rs.getString("name");
                String email = rs.getString("email");
                String position = rs.getString("position");
                employees.add(new Employee(id, name, email, position));
            }
        } catch (SQLException e) {
            printSQLException(e);
        }
        return employees;
    }
```



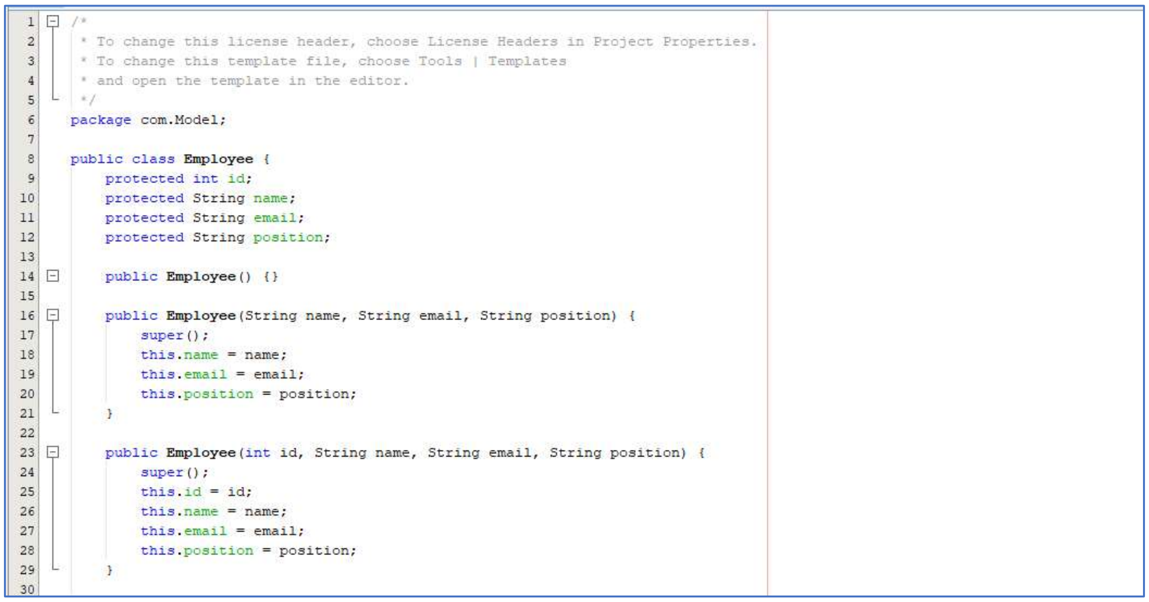
```java
    public boolean deleteEmployee(int id) throws SQLException {
        boolean rowDeleted;
        try (Connection connection = getConnection(); PreparedStatement statement =
                connection.prepareStatement(DELETE_EMPLOYEES_SQL);) {
            statement.setInt(1, id);
         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

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

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

```



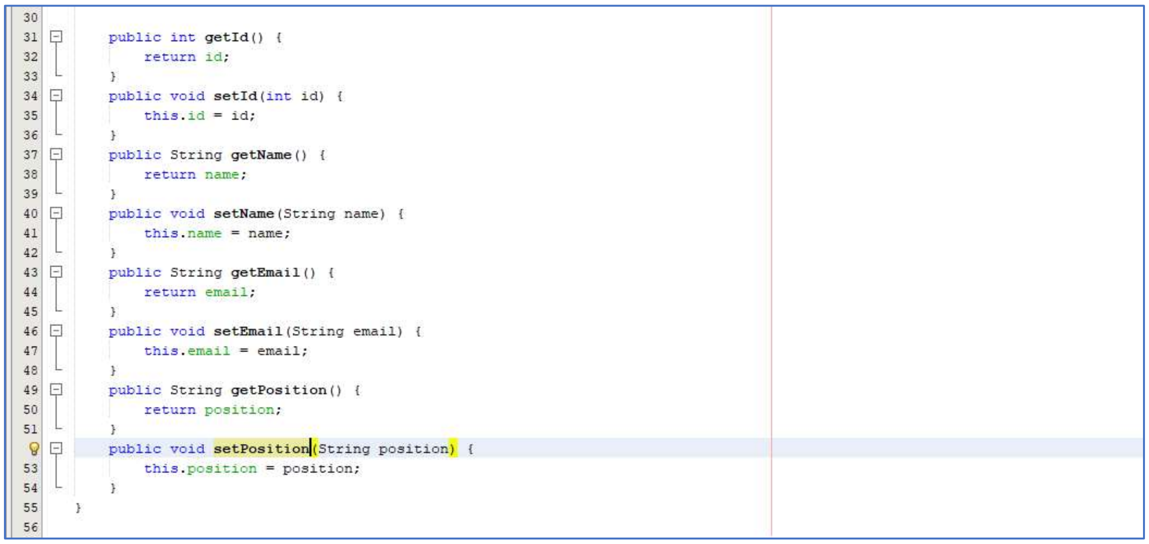
```java

    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

Name the package as com.Model



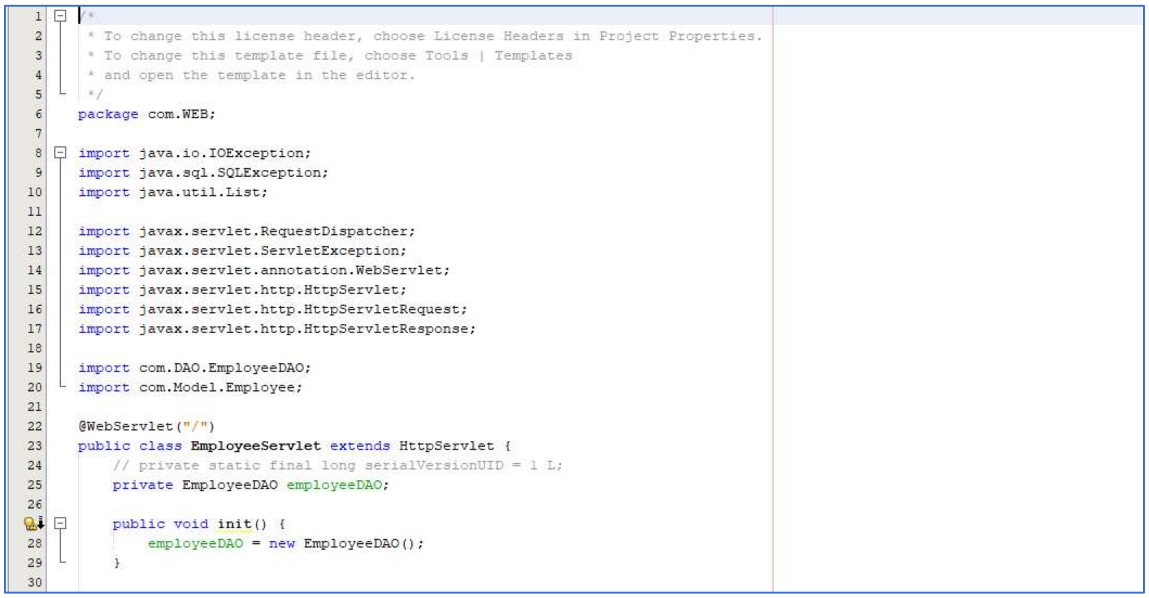
```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package com.Model;

public class Employee {
    protected int id;
    protected String name;
    protected String email;
    protected String position;

    public Employee() {}

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

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

```



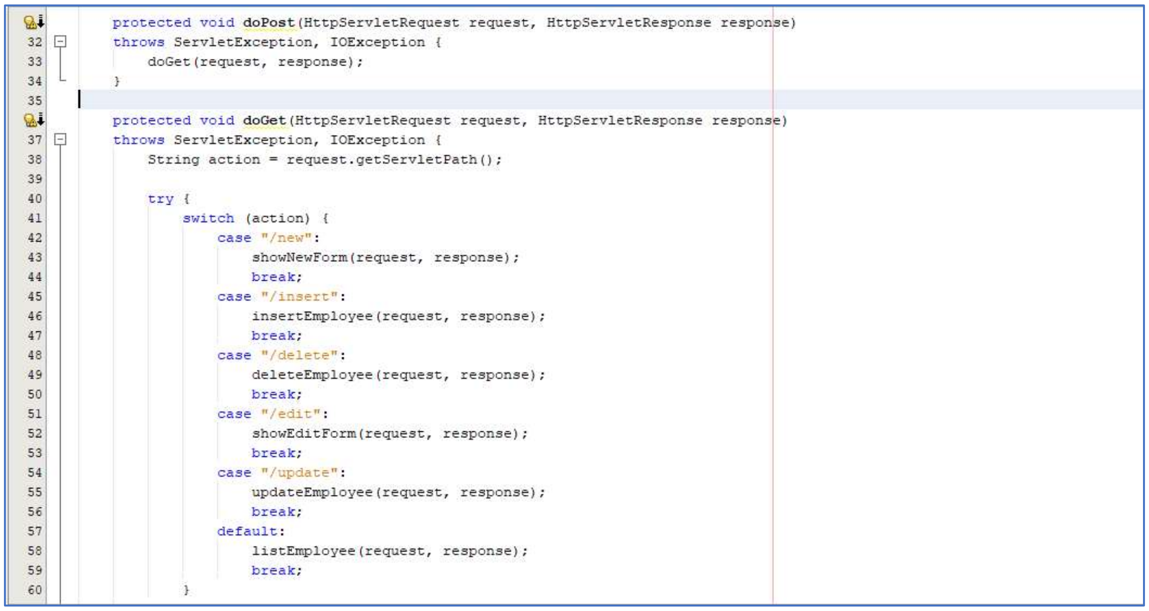
```

    public int getId() {
        return id;
    }
    public void setId(int id) {
        this.id = id;
    }
    public String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }
    public String getEmail() {
        return email;
    }
    public void setEmail(String email) {
        this.email = email;
    }
    public String getPosition() {
        return position;
    }
    public void setPosition(String position) {
        this.position = position;
    }
}

```

EmployeeServlet.java

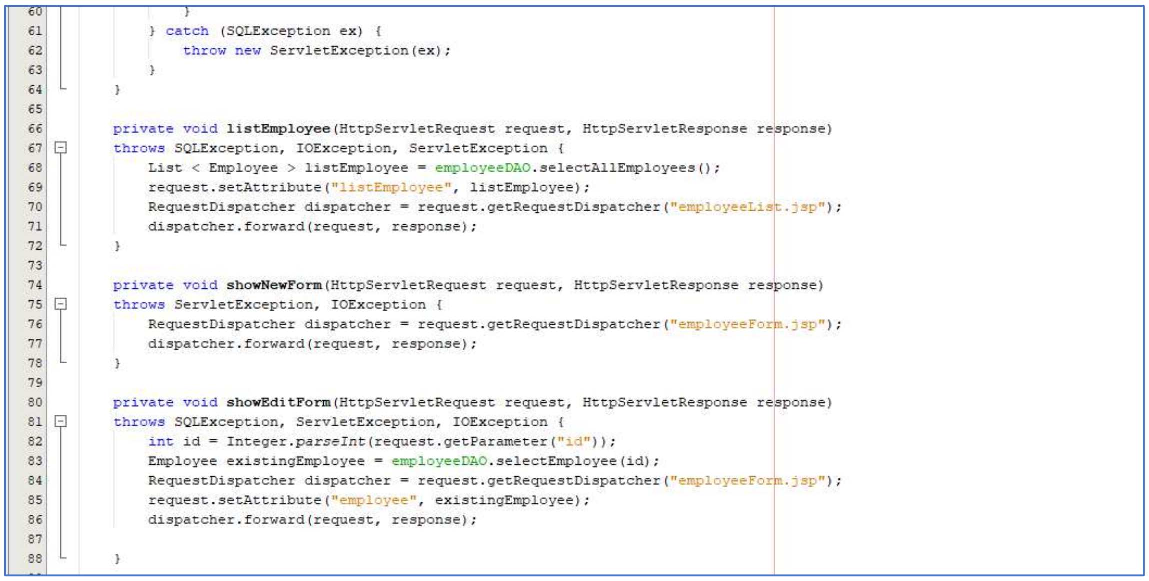
Name the package as com.WEB



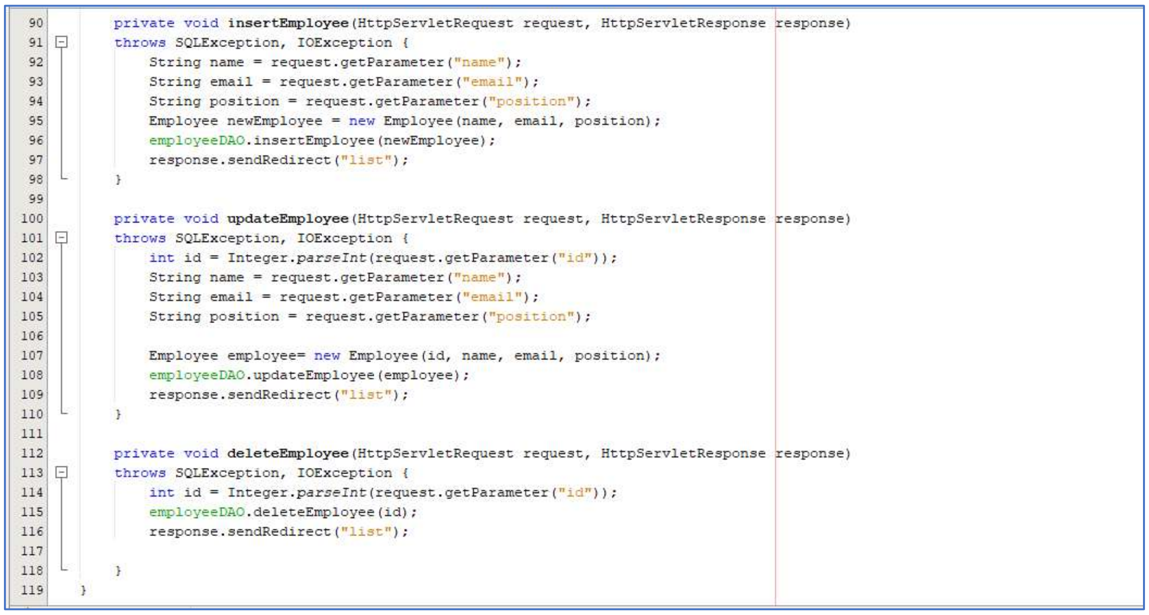
```java
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package com.WEB;

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

import javax.servlet.RequestDispatcher;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

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

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

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

```



```java
    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;
            }
```

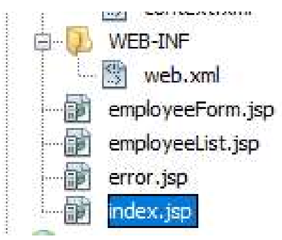


```java
        } catch (SQLException ex) {
            throw new ServletException(ex);
        }
    }

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

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

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

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

    }
}
```

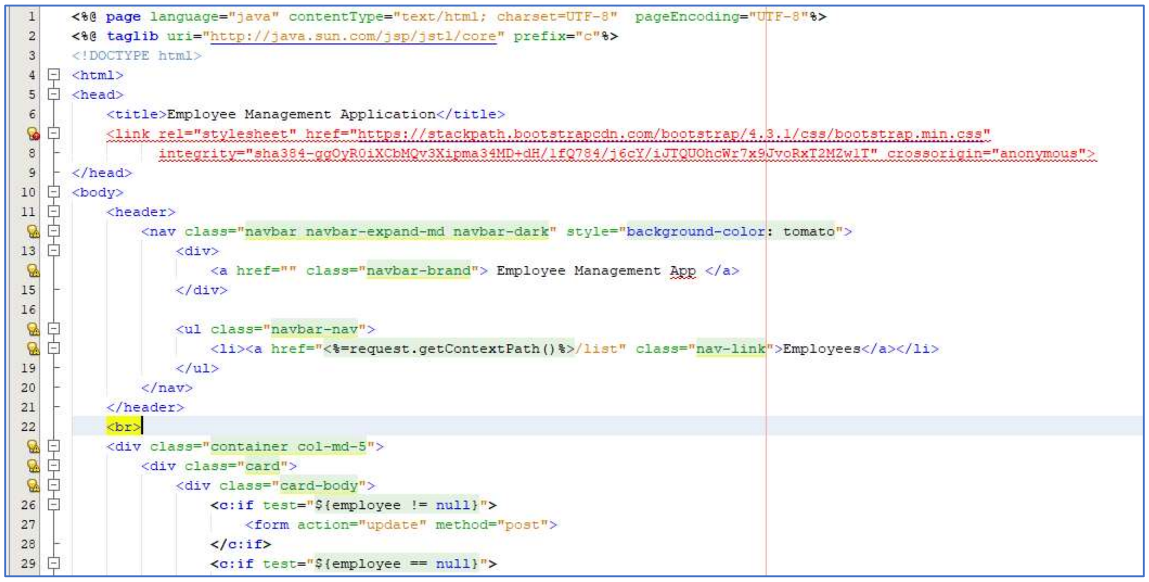
Step 4 - Create these files:



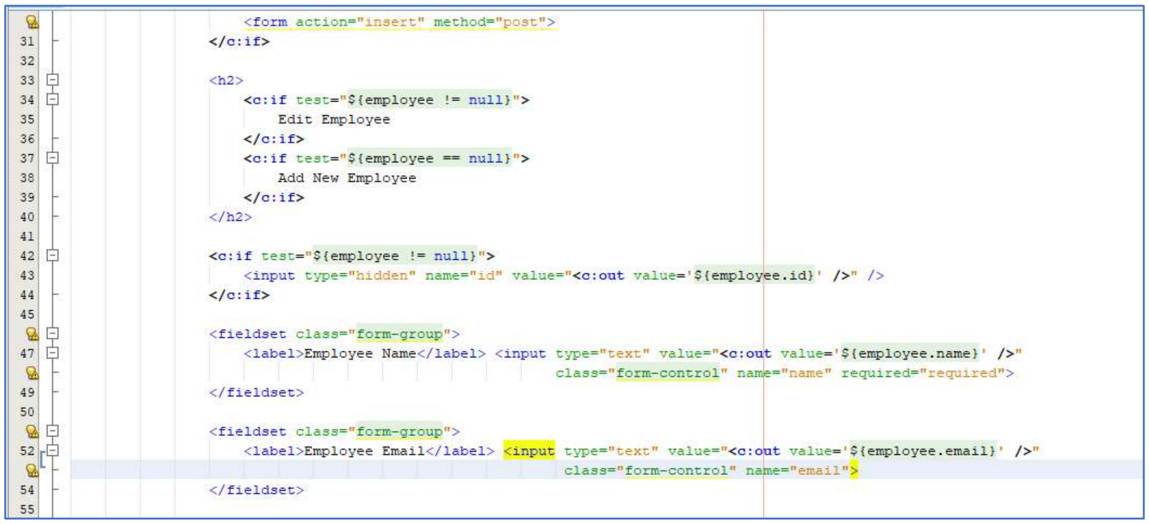
1. File web.xml

   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.

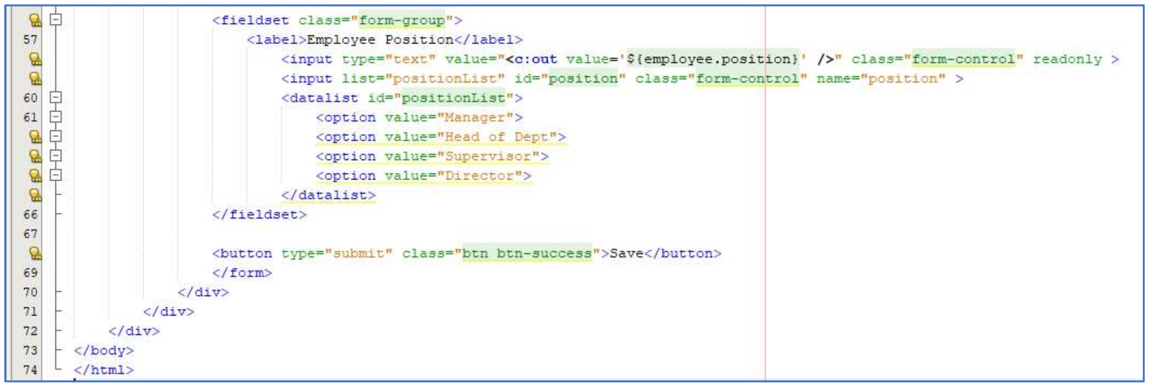
2. File EmployeeForm.jsp (used for Add and Edit/Update process)



```jsp
<%@ page language="java" contentType="text/html; charset=UTF-8"  pageEncoding="UTF-8"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<!DOCTYPE html>
<html>
<head>
    <title>Employee Management Application</title>
    <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css"
          integrity="sha384-ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T" crossorigin="anonymous">
</head>
<body>
    <header>
        <nav class="navbar navbar-expand-md navbar-dark" style="background-color: tomato">
            <div>
                <a href="" class="navbar-brand"> Employee Management App </a>
            </div>

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

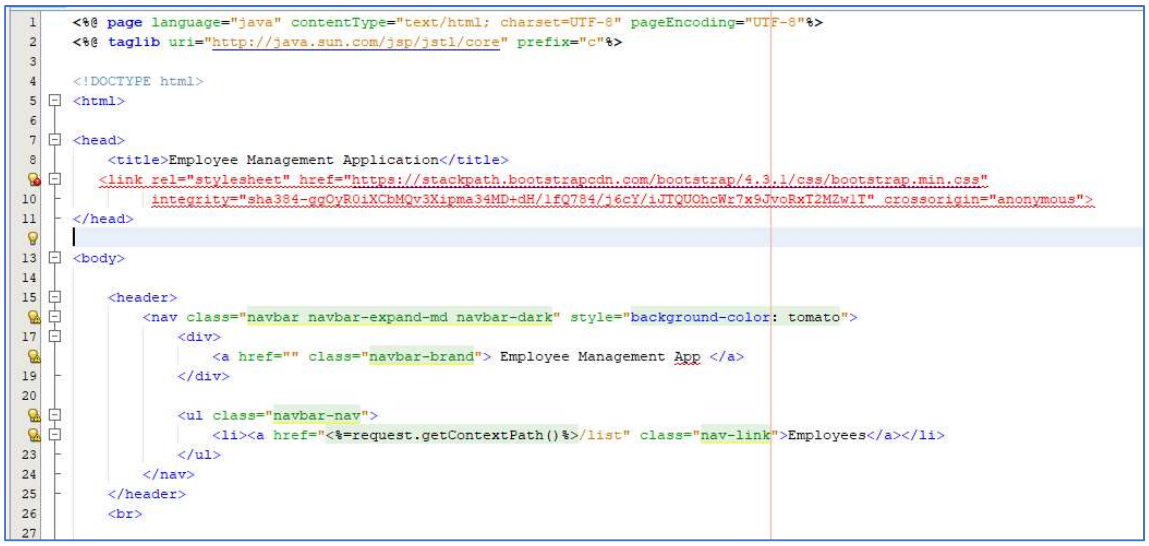


```
                    <form action="insert" method="post">
                </c:if>

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

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

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

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

```

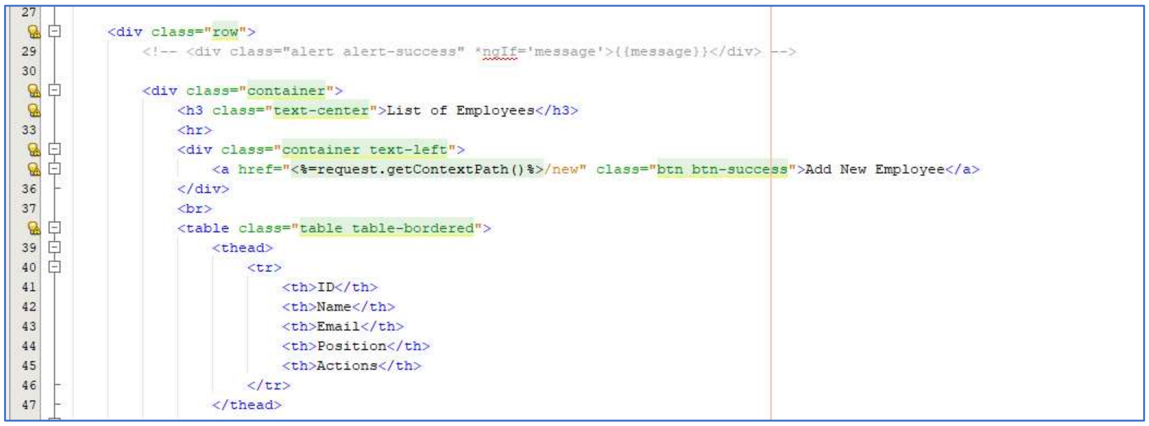


```
                <fieldset class="form-group">
                    <label>Employee Position</label>
                        <input type="text" value="<c:out value='${employee.position}' />" class="form-control" readonly >
                        <input list="positionList" id="position" class="form-control"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

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

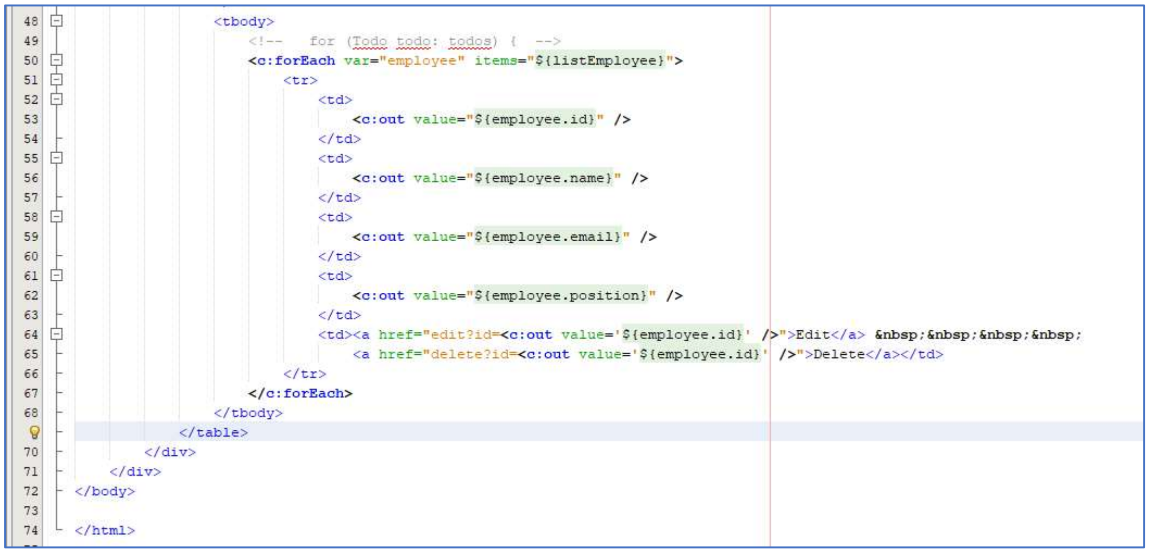
3. File EmployeeList.jsp (used for displaying all employee records)



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

<!DOCTYPE html>
<html>

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

<body>

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

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

```

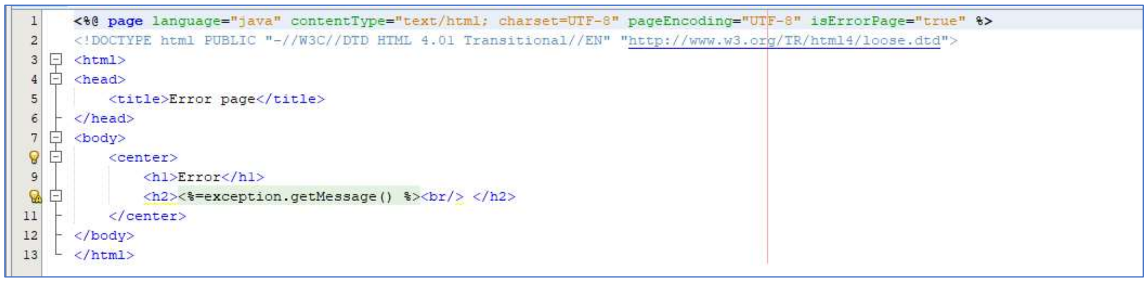


```
    <div class="row">
        <!-- <div class="alert alert-success" *ngIf='message'>{{message}}</div> -->

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



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

</html>
```

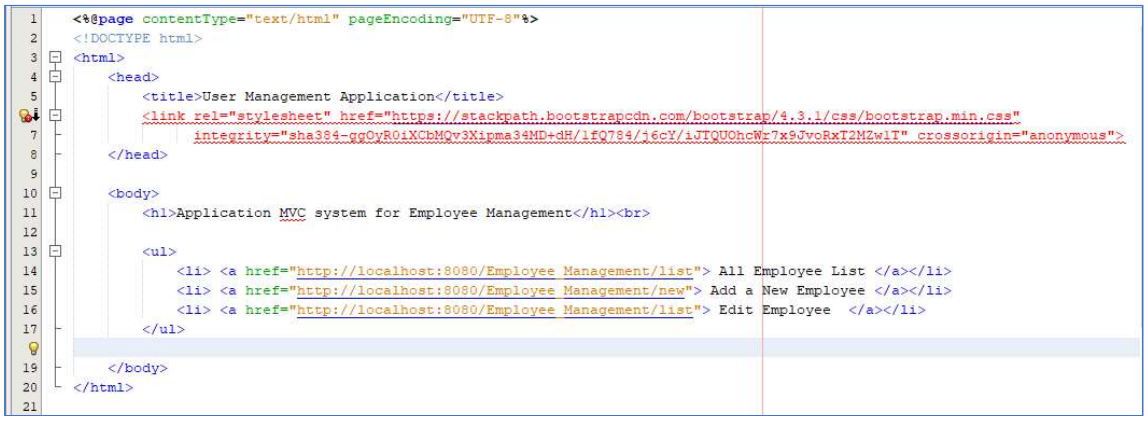
4. File error.jsp



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

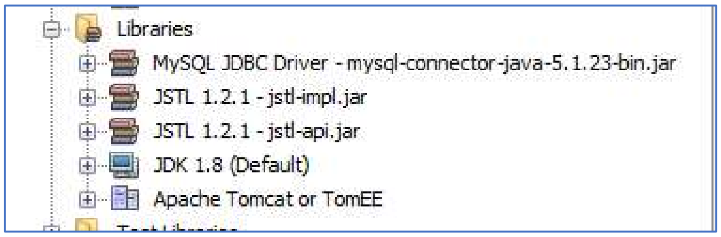
5. File index.jsp

Contents of the Index.jsp:



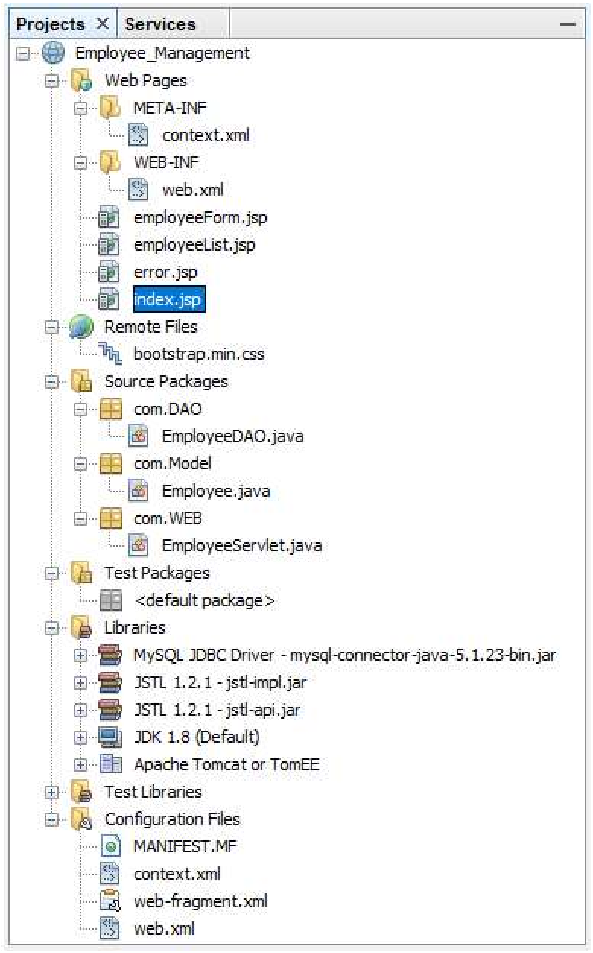
```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
    <head>
        <title>User Management Application</title>
        <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css"
              integrity="sha384-ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T" crossorigin="anonymous">
    </head>

    <body>
        <h1>Application MVC system for Employee Management</h1><br>

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

    </body>
</html>
```

Step 5 - Add these libraries:



```
Libraries
    MySQL JDBC Driver - mysql-connector-java-5.1.23-bin.jar
    JSTL 1.2.1 - jstl-impl.jar
    JSTL 1.2.1 - jstl-api.jar
    JDK 1.8 (Default)
    Apache Tomcat or TomEE
```

Finally, the Project schema should be like this:

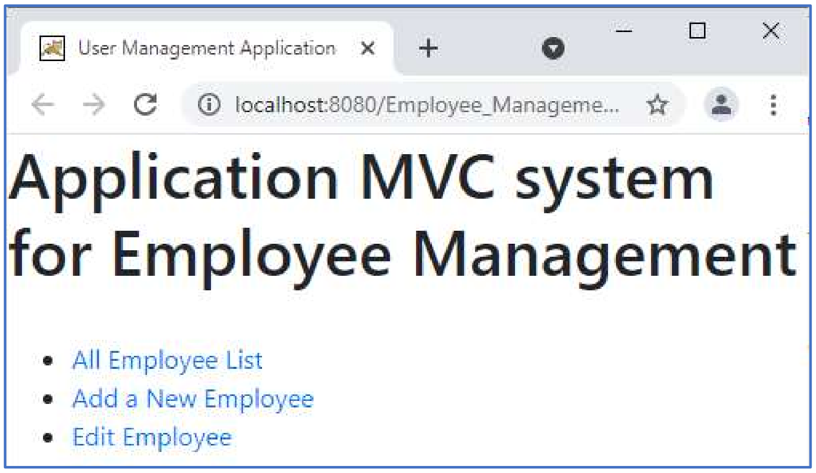


```
Projects   Services
Employee_Management
    Web Pages
        META-INF
            context.xml
        WEB-INF
            web.xml
        employeeForm.jsp
        employeeList.jsp
        error.jsp
        index.jsp
    Remote Files
        bootstrap.min.css
    Source Packages
        com.DAO
            EmployeeDAO.java
        com.Model
            Employee.java
        com.WEB
            EmployeeServlet.java
    Test Packages
        <default package>
    Libraries
        MySQL JDBC Driver - mysql-connector-java-5.1.23-bin.jar
        JSTL 1.2.1 - jstl-impl.jar
        JSTL 1.2.1 - jstl-api.jar
        JDK 1.8 (Default)
        Apache Tomcat or TomEE
    Test Libraries
    Configuration Files
        MANIFEST.MF
        context.xml
        web-fragment.xml
        web.xml
```

Step 6 - Running the program and try CRUD process:

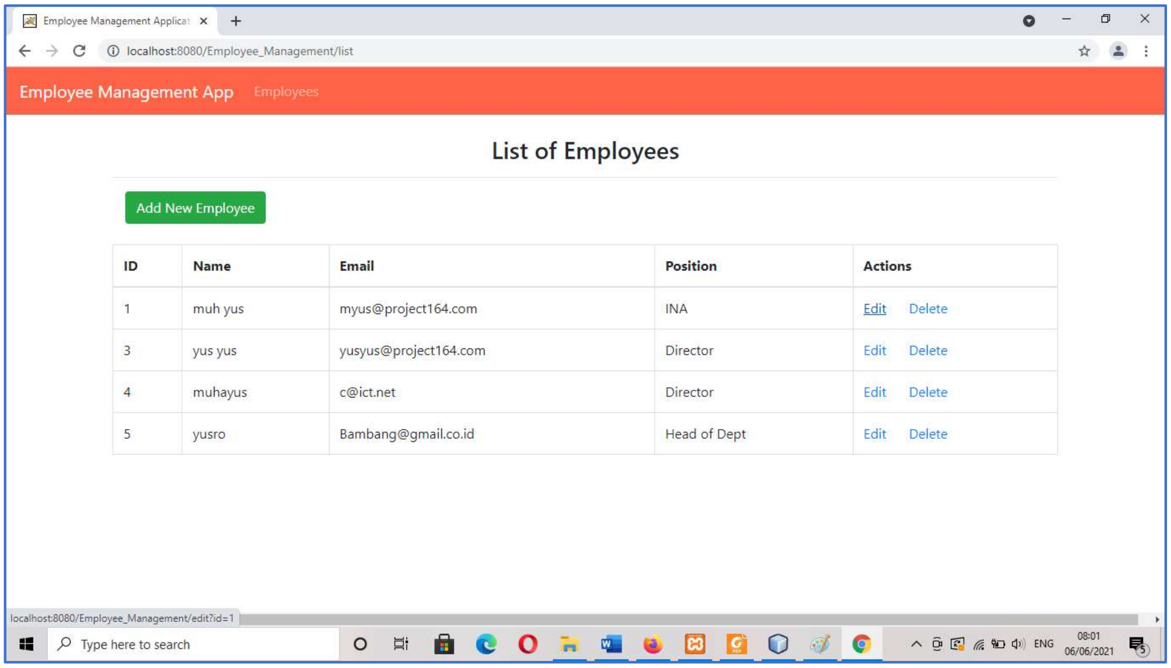
1. Run index.jsp page.
2. Click List All User button to show all records.
3. Click Add User button to create new record.
4. Click hyperlink Update to do edit/update an existing record.
5. Click hyperlink Delete to do delete an existing record.

The result:


User Management Application
localhost:8080/Employee_Manageme...

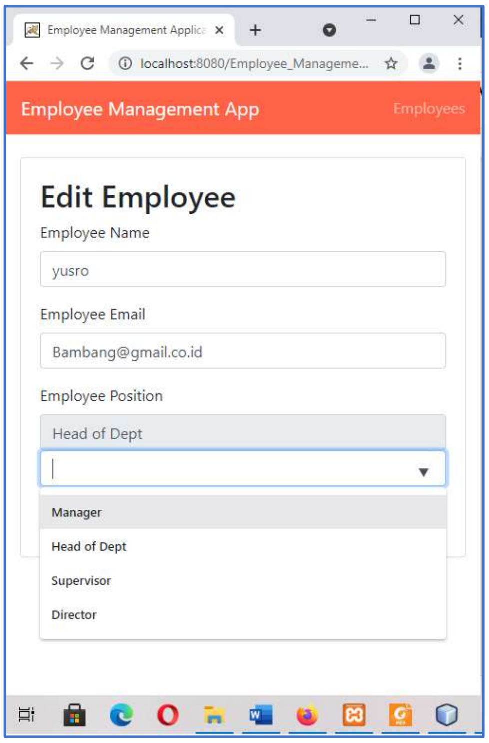
Application MVC system for Employee Management

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


Employee Management Applicat
localhost:8080/Employee_Management/list
Employee Management App
Employees
Add New Employee
localhost:8080/Employee_Management/edit?id=1
Type here to search
ENG
08:01
06/06/2021

List of Employees

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


Employee Management App

Edit Employee

Employee Name: yusro

Employee Email: Bambang@gmail.co.id

Employee Position: Head of Dept

Employee Management Applica
localhost:8080/Employee_Manageme...
Employees
Manager
Head of Dept
Supervisor
Director

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