



**UNIVERSITI MALAYSIA TERENGGANU**  
**FACULTY OF COMPUTER SCIENCE AND MATHEMATICS**

**CSM3023 – WEB BASED APPLICATION DEVELOPMENT**  
**LAB REPORT 8 – JSP : AN MVC EXAMPLE WITH SERVLETS AND JSP**

**PREPARED BY:**  
**HARINATUL MUFLIHUN BINTI HASNUL MUNAWAR**  
**( S67604 )**

**PREPARED FOR:**  
**DR. MOHAMAD NOR BIN HASSAN**

**BACHELOR OF COMPUTER SCIENCE (MOBILE COMPUTING)**  
**WITH HONOURS**  
**SEMESTER II 2023/2024**

## TASK 1 : CREATING MVC DATABASE WEB APPLICATION IN JSP AND SERVLETS – FOR CREATE, READ, UPDATE, DELETE

COMPANY database schema.

```
1 • CREATE DATABASE IF NOT EXISTS Company;
2 • USE Company;
3
4 • CREATE TABLE IF NOT EXISTS employees (
5     id INT NOT NULL auto_increment,
6     Name varchar(60),
7     Email varchar(50),
8     Position varchar(15),
9     primary key (id)
10 )
```

EmployeeDAO.java

```
5 package com.DAO;
6
7 import java.sql.Connection;
8 import java.sql.DriverManager;
9 import java.sql.PreparedStatement;
10 import java.sql.ResultSet;
11 import java.sql.SQLException;
12 import java.util.ArrayList;
13 import java.util.List;
14
15 import com.Model.Employee;
16
17 public class EmployeeDAO {
18     private String jdbcURL = "jdbc:mysql://localhost:3306/company"; // Corrected URL
19     private String jdbcUsername = "root";
20     private String jdbcPassword = "admin";
21
22     private static final String INSERT_EMPLOYEES_SQL = "INSERT INTO employees (name, email, position) VALUES (?, ?, ?);";
23     private static final String SELECT_EMPLOYEE_BY_ID = "SELECT id, name, email, position FROM employees WHERE id = ?";
24     private static final String SELECT_ALL_EMPLOYEES = "SELECT * FROM employees";
25     private static final String DELETE_EMPLOYEES_SQL = "DELETE FROM employees WHERE id = ?;";
26     private static final String UPDATE_EMPLOYEES_SQL = "UPDATE employees SET name = ?, email = ?, position = ? WHERE id = ?;";
27
28     public EmployeeDAO() {}
```

```

30 protected Connection getConnection() {
31     Connection connection = null;
32     try {
33         Class.forName("com.mysql.cj.jdbc.Driver");
34         connection = DriverManager.getConnection(jdbcURL, jdbcUsername, jdbcPassword);
35         System.out.println("Database connected!");
36     } catch (SQLException e) {
37         e.printStackTrace();
38     } catch (ClassNotFoundException e) {
39         e.printStackTrace();
40     }
41     return connection;
42 }
43
44 public void insertEmployee(Employee employee) throws SQLException {
45     System.out.println(INSERT_EMPLOYEES_SQL);
46     try (Connection connection = getConnection();
47         PreparedStatement preparedStatement = connection.prepareStatement(INSERT_EMPLOYEES_SQL)) {
48         preparedStatement.setString(1, employee.getName());
49         preparedStatement.setString(2, employee.getEmail());
50         preparedStatement.setString(3, employee.getPosition());
51         System.out.println(preparedStatement);
52         preparedStatement.executeUpdate();
53     } catch (SQLException e) {
54         printSQLException(e);
55     }
56 }

```

```

78 public List<Employee> selectAllEmployee() {
79     List<Employee> employees = new ArrayList<>();
80     try (Connection connection = getConnection();
81         PreparedStatement preparedStatement = connection.prepareStatement(SELECT_ALL_EMPLOYEES)) {
82         System.out.println(preparedStatement);
83         ResultSet rs = preparedStatement.executeQuery();
84
85         while (rs.next()) {
86             int id = rs.getInt("id");
87             String name = rs.getString("name");
88             String email = rs.getString("email");
89             String position = rs.getString("position");
90             employees.add(new Employee(id, name, email, position));
91         }
92     } catch (SQLException e) {
93         printSQLException(e);
94     }
95     return employees;
96 }

```

```

98 public boolean deleteEmployee(int id) throws SQLException {
99     boolean rowDeleted;
100     try (Connection connection = getConnection();
101         PreparedStatement statement = connection.prepareStatement(DELETE_EMPLOYEES_SQL)) {
102         statement.setInt(1, id);
103         rowDeleted = statement.executeUpdate() > 0;
104     }
105     return rowDeleted;
106 }
107
108 public boolean updateEmployee(Employee employee) throws SQLException {
109     boolean rowUpdated;
110     try (Connection connection = getConnection();
111         PreparedStatement statement = connection.prepareStatement(UPDATE_EMPLOYEES_SQL)) {
112         statement.setString(1, employee.getName());
113         statement.setString(2, employee.getEmail());
114         statement.setString(3, employee.getPosition());
115         statement.setInt(4, employee.getId());
116
117         rowUpdated = statement.executeUpdate() > 0;
118     }
119     return rowUpdated;
120 }

```

```

122 private void printSQLException(SQLException ex) {
123     for (Throwable e : ex) {
124         if (e instanceof SQLException) {
125             e.printStackTrace(System.err);
126             System.err.println("SQLState: " + ((SQLException) e).getSQLState());
127             System.err.println("Error Code: " + ((SQLException) e).getErrorCode());
128             System.err.println("Message: " + e.getMessage());
129             Throwable t = ex.getCause();
130             while (t != null) {
131                 System.out.println("Cause: " + t);
132                 t = t.getCause();
133             }
134         }
135     }
136 }
137 }

```

## Employee.java

```

5 package com.Model;
6
7 public class Employee {
8     protected int id;
9     protected String name;
10    protected String email;
11    protected String position;
12
13    public Employee() {}
14
15    public Employee (String name, String email, String position) {
16        super();
17        this.name = name;
18        this.email = email;
19        this.position = position;
20    }
21
22    public Employee(int id, String name, String email, String position){
23        super();
24        this.id = id;
25        this.name = name;
26        this.email = email;
27        this.position = position;
28    }
29
30    public void setId(int id) {
31        this.id = id;
32    }
33 }

```

```

34  public void setName(String name) {
35      this.name = name;
36  }
37
38  public void setEmail(String email) {
39      this.email = email;
40  }
41
42  public void setPosition(String position) {
43      this.position = position;
44  }
45
46  public int getId() {
47      return id;
48  }
49
50  public String getName() {
51      return name;
52  }
53
54  public String getEmail() {
55      return email;
56  }
57
58  public String getPosition() {
59      return position;
60  }
61
62  }

```

## employeeServlet.java

```

5  package com.WEB;
6
7  import java.io.IOException;
8  import java.sql.SQLException;
9  import java.util.List;
10
11  import java.io.PrintWriter;
12  import java.sql.Connection;
13  import java.sql.DriverManager;
14  import java.sql.PreparedStatement;
15  import jakarta.servlet.ServletException;
16  import jakarta.servlet.annotation.WebServlet;
17  import jakarta.servlet.http.HttpServlet;
18  import jakarta.servlet.http.HttpServletRequest;
19  import jakarta.servlet.http.HttpServletResponse;
20
21  import com.DAO.EmployeeDAO;
22  import com.Model.Employee;
23  import jakarta.servlet.RequestDispatcher;
24
25  @WebServlet("/")
26  public class EmployeeServlet extends HttpServlet {
27
28      private EmployeeDAO employeeDAO;
29
30      public void init() {
31          employeeDAO = new EmployeeDAO();
32      }

```

```

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);
    }
}

```

```

66     }
67 }
68
69 private void listEmployee (HttpServletRequest request, HttpServletResponse response)
70     throws SQLException, IOException, ServletException {
71     List <Employee> listEmployee = employeeDAO.selectAllEmployee();
72     request.setAttribute ("listEmployee", listEmployee);
73     RequestDispatcher dispatcher = request.getRequestDispatcher ("EmployeeList.jsp");
74     dispatcher.forward (request, response);
75 }
76
77 private void showNewForm (HttpServletRequest request, HttpServletResponse response)
78     throws ServletException, IOException {
79     RequestDispatcher dispatcher= request.getRequestDispatcher("EmployeeForm.jsp");
80     dispatcher.forward (request, response);
81 }
82
83 private void showEditForm (HttpServletRequest request, HttpServletResponse response)
84     throws SQLException, ServletException, IOException {
85     int id = Integer.parseInt(request.getParameter("id"));
86     Employee existingEmployee = employeeDAO.selectEmployee(id);
87     RequestDispatcher dispatcher = request.getRequestDispatcher ("EmployeeForm.jsp");
88     request.setAttribute ("employee", existingEmployee);
89     dispatcher.forward(request, response);
90 }
91

```

```

92 private void insertEmployee (HttpServletRequest request, HttpServletResponse response)
93     throws SQLException, IOException {
94     String name = request.getParameter("name");
95     String email = request.getParameter("email");
96     String position = request.getParameter("position");
97     Employee newEmployee = new Employee(name, email, position);
98     employeeDAO.insertEmployee (newEmployee);
99     response.sendRedirect ("list");
100 }
101
102 private void updateEmployee (HttpServletRequest request, HttpServletResponse response)
103     throws SQLException, IOException {
104     int id = Integer.parseInt(request.getParameter("id"));
105     String name= request.getParameter("name");
106     String email = request.getParameter("email");
107     String position = request.getParameter("position");
108
109     Employee employee = new Employee(id, name, email, position);
110     employeeDAO.updateEmployee(employee);
111     response.sendRedirect ("list");
112 }
113
114 private void deleteEmployee (HttpServletRequest request, HttpServletResponse response)
115     throws SQLException, IOException {
116     int id = Integer.parseInt(request.getParameter("id"));
117     employeeDAO.deleteEmployee(id);
118     response.sendRedirect ("list");
119 }
120 }

```

## employeeForm.jsp

```

8 <%@page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>
9 <%@taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
10 <!DOCTYPE html>
11 <html>
12 <head>
13     <title>Employee Management Application</title>
14     <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/css/bootstrap.min.css">
15     <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.7.1/jquery.min.js"></script>
16     <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/js/bootstrap.min.js"></script>
17 </head>
18 <body>
19     <header>
20         <nav class="navbar navbar-expand-md navbar-dark" style="background-color: tomato">
21             <div>
22                 <a href="" class="navbar-brand"> Employee Management App </a>
23             </div>
24
25             <ul class="navbar-nav">
26                 <li><a href="<%=request.getContextPath()%>/list" class="nav-link">Employees</a></li>
27             </ul>
28         </nav>
29     </header>
30     <br>
31     <div class="container col-md-5">
32         <div class="card">
33             <div class="card-body">
34                 <c:if test="{employee != null}">
35                     <form action="update" method="post">
36                 </c:if>
37                 <c:if test="{employee == null}">
38                     <form action="insert" method="post">
39                 </c:if>

```

```

41 |
42 |
43 |
44 |
45 |
46 |
47 |
48 |
49 |
50 |
51 |
52 |
53 |
54 |
55 |
56 |
57 |
58 |
59 |
60 |
61 |
62 |
63 |
64 |
65 |
66 |
67 |
68 |
69 |
70 |
71 |
72 |
73 |
74 |
75 |
76 |
77 |
78 |
79 |
80 |
81 |
82 |

```

```

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

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

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

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

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

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

```

## employeeList.jsp

```

7 |
8 |
9 |
10 |
11 |
12 |
13 |
14 |
15 |
16 |
17 |
18 |
19 |
20 |
21 |
22 |
23 |
24 |
25 |
26 |
27 |
28 |
29 |
30 |
31 |
32 |
33 |
34 |
35 |
36 |

```

```

<@page contentType="text/html" pageEncoding="UTF-8"%>
<@taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<!DOCTYPE html>
<html>
<head>
  <title>Employee Management Application</title>
  <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/css/bootstrap.min.css">
  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.7.1/jquery.min.js"></script>
  <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/js/bootstrap.min.js"></script>
</head>
<body>
  <header>
    <nav class="navbar navbar-expand-md navbar-dark" style="background-color: tomato">
      <div>
        <a href="" class="navbar-brand">Employee Management App </a>
      </div>
      <ul class="navbar-nav">
        <li><a href="<%=request.getContextPath()%>/list" class="nav-link">Employees</a></li>
      </ul>
    </header>
    <br>
    <div class="row">
      <div class="container">
        <h3 class="text-center">List of Employees</h3>
        <hr>
        <div class="container text-left">
          <a href="<%=request.getContextPath()%>/new" class="btn btn-success">Add New Employee</a>
        </div>
        <br>

```



```

37 |         <table class="table table-bordered">
38 |             <thead>
39 |                 <tr>
40 |                     <th>ID</th>
41 |                     <th>Name</th>
42 |                     <th>Email</th>
43 |                     <th>Position</th>
44 |                     <th>Actions</th>
45 |                 </tr>
46 |             </thead>
47 |             <tbody>
48 |                 <c:forEach var="employee" items="${listEmployee}">
49 |                     <tr>
50 |                         <td>
51 |                             <c:out value="${employee.id}" />
52 |                         </td>
53 |                         <td>
54 |                             <c:out value="${employee.name}" />
55 |                         </td>
56 |                         <td>
57 |                             <c:out value="${employee.email}" />
58 |                         </td>
59 |                         <td>
60 |                             <c:out value="${employee.position}" />
61 |                         </td>
62 |                         <td>
63 |                             <a href="edit?id=<c:out value="${employee.id}" />">Edit</a> &nbsp; &nbsp; &nbsp; &nbsp; &nbsp;
64 |                             <a href="delete?id=<c:out value="${employee.id}" />">Delete</a></td>
65 |                         </tr>
66 |                     </c:forEach>
67 |                 </tbody>
68 |             </table>
69 |
70 |         </div>
71 |     </div>
72 | </body>
73 | </html>

```

## Error.jsp

```

7 | <%@page contentType="text/html" pageEncoding="UTF-8"%>
8 | <!DOCTYPE html>
9 | <html>
10 |     <head>
11 |         <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
12 |         <title>Error page</title>
13 |     </head>
14 |     <body>
15 |         <center>
16 |             <h1>Error</h1>
17 |             <h2><%=exception.getMessage() %><br/></h2>
18 |         </center>
19 |     </body>
20 | </html>
21 |

```

## Index.jsp

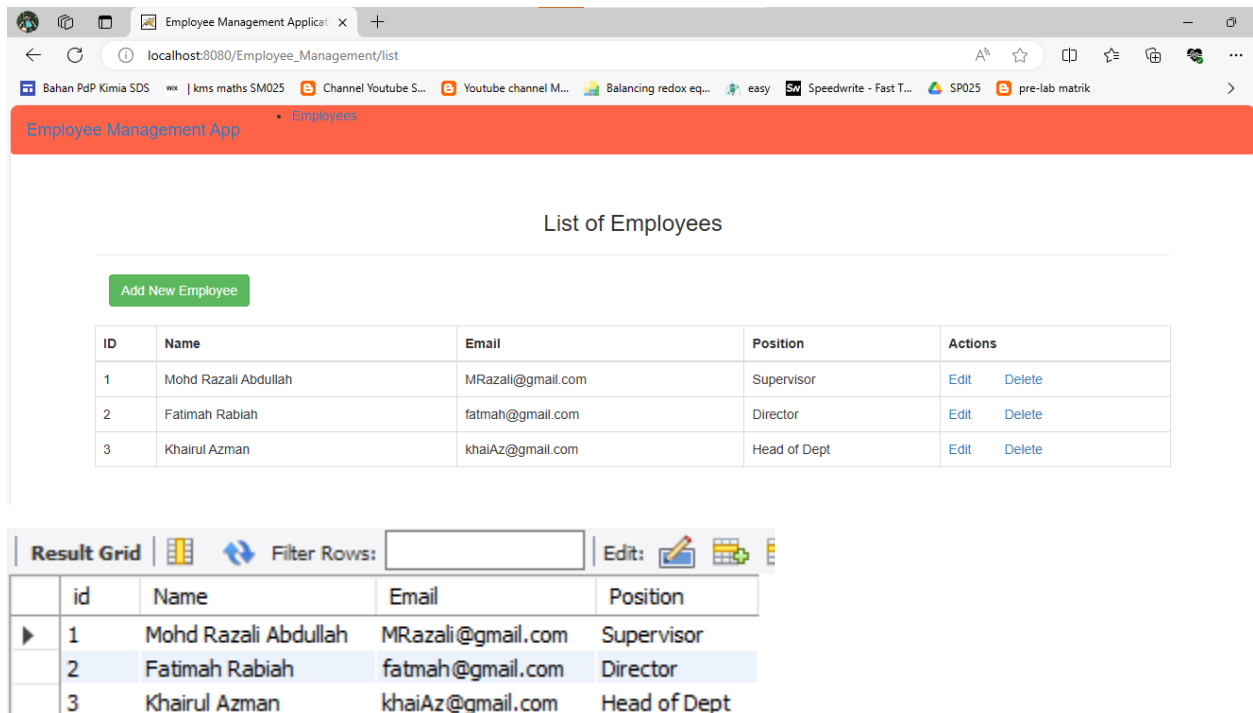
```

7 | <%@page contentType="text/html" pageEncoding="UTF-8"%>
8 | <!DOCTYPE html>
9 | <html>
10 |     <head>
11 |         <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
12 |         <title>User Management Application</title>
13 |         <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/css/bootstrap.min.css">
14 |         <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.7.1/jquery.min.js"></script>
15 |         <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/js/bootstrap.min.js"></script>
16 |     </head>
17 |     <body>
18 |         <h1>Application MVC system for Employee Management</h1><br>
19 |
20 |         <ul>
21 |             <li><a href="http://localhost:8080/Employee Management/list">All Employee List</a></li>
22 |             <li><a href="http://localhost:8080/Employee Management/new">Add a New Employee</a></li>
23 |             <li><a href="http://localhost:8080/Employee Management/list">Edit Employee</a></li>
24 |         </ul>
25 |     </body>
26 | </html>

```

The output.

- This is the list data that I insert.

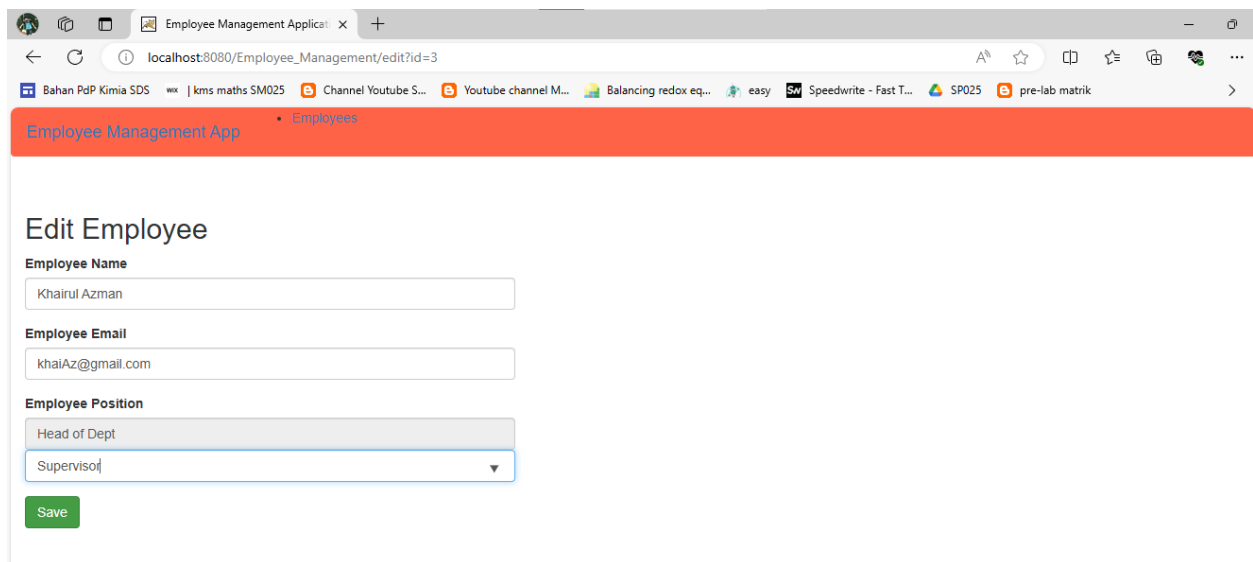


The screenshot shows a web browser window with the URL `localhost:8080/Employee_Management/list`. The page title is "Employee Management App" and the breadcrumb is "Employees". The main heading is "List of Employees". Below the heading is a green button labeled "Add New Employee". A table lists three employees:

ID	Name	Email	Position	Actions
1	Mohd Razali Abdullah	MRazali@gmail.com	Supervisor	<a href="#">Edit</a> <a href="#">Delete</a>
2	Fatimah Rabiah	fatmah@gmail.com	Director	<a href="#">Edit</a> <a href="#">Delete</a>
3	Khairul Azman	khaiAz@gmail.com	Head of Dept	<a href="#">Edit</a> <a href="#">Delete</a>

Below the table is a "Result Grid" section with a "Filter Rows:" input field and an "Edit:" button. The grid shows the same three employees with their IDs, names, emails, and positions.

- The data that I edit



The screenshot shows a web browser window with the URL `localhost:8080/Employee_Management/edit?id=3`. The page title is "Employee Management App" and the breadcrumb is "Employees". The main heading is "Edit Employee". Below the heading are three form fields:

**Employee Name**

**Employee Email**

**Employee Position**

Below the form fields is a green button labeled "Save".

- The data after edit

Employee Management App

Employees

### List of Employees

[Add New Employee](#)

ID	Name	Email	Position	Actions
1	Mohd Razali Abdullah	MRazali@gmail.com	Supervisor	<a href="#">Edit</a> <a href="#">Delete</a>
2	Fatimah Rabiah	fatmah@gmail.com	Director	<a href="#">Edit</a> <a href="#">Delete</a>
3	Khairul Azman	khaiAz@gmail.com	Supervisor	<a href="#">Edit</a> <a href="#">Delete</a>

	id	Name	Email	Position
▶	1	Mohd Razali Abdullah	MRazali@gmail.com	Supervisor
	2	Fatimah Rabiah	fatmah@gmail.com	Director
	3	Khairul Azman	khaiAz@gmail.com	Supervisor

- The data when I click delete button.

Employee Management App

Employees

### List of Employees

[Add New Employee](#)

ID	Name	Email	Position	Actions
1	Mohd Razali Abdullah	MRazali@gmail.com	Supervisor	<a href="#">Edit</a> <a href="#">Delete</a>
3	Khairul Azman	khaiAz@gmail.com	Supervisor	<a href="#">Edit</a> <a href="#">Delete</a>

	id	Name	Email	Position
▶	1	Mohd Razali Abdullah	MRazali@gmail.com	Supervisor
	3	Khairul Azman	khaiAz@gmail.com	Supervisor

- The output add new employee.

Employee Management App • Employees

## Add New Employee

**Employee Name**

**Employee Email**

**Employee Position**

[Save](#)

Employee Management App • Employees

## List of Employees

[Add New Employee](#)

ID	Name	Email	Position	Actions
1	Mohd Razali Abdullah	MRazali@gmail.com	Supervisor	<a href="#">Edit</a> <a href="#">Delete</a>
3	Khairul Azman	khaiAz@gmail.com	Supervisor	<a href="#">Edit</a> <a href="#">Delete</a>
4	Laila a/p Din	laila@gmail.com	Head of Dept	<a href="#">Edit</a> <a href="#">Delete</a>

	id	Name	Email	Position
▶	1	Mohd Razali Abdullah	MRazali@gmail.com	Supervisor
	3	Khairul Azman	khaiAz@gmail.com	Supervisor
	4	Laila a/p Din	laila@gmail.com	Head of Dept

## 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

Carshop scheme.

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

---

## CarDAO.java

```
5 package com.DAO;
6
7 import java.sql.Connection;
8 import java.sql.DriverManager;
9 import java.sql.PreparedStatement;
10 import java.sql.ResultSet;
11 import java.sql.SQLException;
12 import java.util.ArrayList;
13 import java.util.List;
14
15 import com.Model.Car;
16
17 public class CarDAO {
18     private String jdbcURL = "jdbc:mysql://localhost:3306/carshop";
19     private String jdbcUsername = "root";
20     private String jdbcPassword = "admin";
21
22     private static final String INSERT_CAR_SQL = "INSERT INTO CarPricelist (Brand, Model, Cylinder, Price) VALUES (?, ?, ?, ?)";
23     private static final String SELECT_CAR_BY_ID = "SELECT Car_id, Brand, Model, Cylinder, Price FROM CarPricelist WHERE Car_id = ?";
24     private static final String SELECT_ALL_CAR = "SELECT * FROM CarPricelist";
25     private static final String DELETE_CAR_SQL = "DELETE FROM CarPricelist WHERE Car_id = ?";
26     private static final String UPDATE_CAR_SQL = "UPDATE CarPricelist SET Brand = ?, Model = ?, Cylinder = ?, Price = ? WHERE Car_id = ?";
27
28     public CarDAO() {}
29
30     protected Connection getConnection() {
31         Connection connection = null;
32         try {
33             Class.forName("com.mysql.cj.jdbc.Driver");
34             connection = DriverManager.getConnection(jdbcURL, jdbcUsername, jdbcPassword);
35         } catch (SQLException | ClassNotFoundException e) {
36             e.printStackTrace();
37         }
38         return connection;
39     }
40
41     public void insertCar(Car car) throws SQLException {
42         System.out.println(INSERT_CAR_SQL);
43         try (Connection connection = getConnection();
44             PreparedStatement preparedStatement = connection.prepareStatement(INSERT_CAR_SQL)) {
45             preparedStatement.setString(1, car.getBrand());
46             preparedStatement.setString(2, car.getModel());
47             preparedStatement.setString(3, car.getCylinder());
48             preparedStatement.setString(4, car.getPrice());
49             System.out.println(preparedStatement);
50             preparedStatement.executeUpdate();
51         } catch (SQLException e) {
52             printSQLException(e);
53         }
54     }
55
56     public Car selectCar(int car_id) {
57         Car car = null;
58         try (Connection connection = getConnection();
59             PreparedStatement preparedStatement = connection.prepareStatement(SELECT_CAR_BY_ID)) {
60             preparedStatement.setInt(1, car_id);
61             System.out.println(preparedStatement);
62             ResultSet rs = preparedStatement.executeQuery();
63             while (rs.next()) {
64                 String brand = rs.getString("Brand");
65                 String model = rs.getString("Model");
66                 String cylinder = rs.getString("Cylinder");
67                 String price = rs.getString("Price");
68                 car = new Car(car_id, brand, model, cylinder, price);
69             }
70         } catch (SQLException e) {
71             printSQLException(e);
72         }
73     }
74 }
```

```

73     return car;
74 }
75
76 public List<Car> selectAllCar() {
77     List<Car> cars = new ArrayList<>();
78     try (Connection connection = getConnection();
79         PreparedStatement preparedStatement = connection.prepareStatement(SELECT_ALL_CAR)) {
80         System.out.println(preparedStatement);
81         ResultSet rs = preparedStatement.executeQuery();
82         while (rs.next()) {
83             int car_id = rs.getInt("Car_id");
84             String brand = rs.getString("Brand");
85             String model = rs.getString("Model");
86             String cylinder = rs.getString("Cylinder");
87             String price = rs.getString("Price");
88             cars.add(new Car(car_id, brand, model, cylinder, price));
89         }
90     } catch (SQLException e) {
91         printSQLException(e);
92     }
93     return cars;
94 }
95
96 public boolean deleteCar(int car_id) throws SQLException {
97     boolean rowDeleted;
98     try (Connection connection = getConnection();
99         PreparedStatement statement = connection.prepareStatement(DELETE_CAR_SQL)) {
100         statement.setInt(1, car_id);
101         rowDeleted = statement.executeUpdate() > 0;
102     }
103     return rowDeleted;
104 }
105
106 public boolean updateCar(Car car) throws SQLException {
107     boolean rowUpdated;
108     try (Connection connection = getConnection();
109         PreparedStatement statement = connection.prepareStatement(UPDATE_CAR_SQL)) {
110         statement.setString(1, car.getBrand());
111         statement.setString(2, car.getModel());
112         statement.setString(3, car.getCylinder());
113         statement.setString(4, car.getPrice());
114         statement.setInt(5, car.getCar_id());
115         rowUpdated = statement.executeUpdate() > 0;
116     }
117     return rowUpdated;
118 }
119
120 private void printSQLException(SQLException ex) {
121     for (Throwable e : ex) {
122         if (e instanceof SQLException) {
123             e.printStackTrace(System.err);
124             System.err.println("SQLState: " + ((SQLException) e).getSQLState());
125             System.err.println("Error Code: " + ((SQLException) e).getErrorCode());
126             System.err.println("Message: " + e.getMessage());
127             Throwable t = ex.getCause();
128             while (t != null) {
129                 System.out.println("Cause: " + t);
130                 t = t.getCause();
131             }
132         }
133     }
134 }
135 }

```

CarServlet.java

```

5 package com.WEB;
6
7 import java.io.IOException;
8 import java.sql.SQLException;
9 import java.util.List;
10 import jakarta.servlet.RequestDispatcher;
11 import jakarta.servlet.ServletException;
12 import jakarta.servlet.annotation.WebServlet;
13 import jakarta.servlet.http.HttpServlet;
14 import jakarta.servlet.http.HttpServletRequest;
15 import jakarta.servlet.http.HttpServletResponse;
16
17 import com.DAO.CarDAO;
18 import com.Model.Car;
19
20 @WebServlet("/")
21 public class CarServlet extends HttpServlet {
22
23     private CarDAO carDAO;
24
25     public void init() {
26         carDAO = new CarDAO();
27     }
28
29     protected void doPost(HttpServletRequest request, HttpServletResponse response)
30         throws ServletException, IOException {
31         doGet(request, response);
32     }
33
34     protected void doGet(HttpServletRequest request, HttpServletResponse response)
35         throws ServletException, IOException {
36         String action = request.getServletPath();
37         try {
38             switch (action) {
39
40                 case "/new":
41                     showNewForm(request, response);
42                     break;
43                 case "/insert":
44                     insertCar(request, response);
45                     break;
46                 case "/delete":
47                     deleteCar(request, response);
48                     break;
49                 case "/edit":
50                     showEditForm(request, response);
51                     break;
52                 case "/update":
53                     updateCar(request, response);
54                     break;
55                 default:
56                     listCar(request, response);
57                     break;
58             }
59         } catch (SQLException ex) {
60             throw new ServletException(ex);
61         }
62     }
63
64     private void listCar(HttpServletRequest request, HttpServletResponse response)
65         throws ServletException, IOException {
66         List<Car> listCar = carDAO.selectAllCar();
67         request.setAttribute("listCar", listCar);
68         RequestDispatcher dispatcher = request.getRequestDispatcher("carList.jsp");
69         dispatcher.forward(request, response);
70     }
71 }

```



```

71 private void showNewForm(HttpServletRequest request, HttpServletResponse response)
72     throws ServletException, IOException {
73     RequestDispatcher dispatcher = request.getRequestDispatcher("carForm.jsp");
74     dispatcher.forward(request, response);
75 }
76
77 private void showEditForm(HttpServletRequest request, HttpServletResponse response)
78     throws SQLException, ServletException, IOException {
79     int car_id = Integer.parseInt(request.getParameter("car_id"));
80     Car existingCar = carDAO.selectCar(car_id);
81     RequestDispatcher dispatcher = request.getRequestDispatcher("carForm.jsp");
82     request.setAttribute("car", existingCar);
83     dispatcher.forward(request, response);
84 }
85
86 private void insertCar(HttpServletRequest request, HttpServletResponse response)
87     throws SQLException, IOException {
88     String brand = request.getParameter("brand");
89     String model = request.getParameter("model");
90     String cyclinder = request.getParameter("cyclinder");
91     String price = request.getParameter("price");
92     Car newCar = new Car(brand, model, cyclinder, price);
93     carDAO.insertCar(newCar);
94     response.sendRedirect("list");
95 }
96
97 private void updateCar(HttpServletRequest request, HttpServletResponse response)
98     throws SQLException, IOException {
99     int car_id = Integer.parseInt(request.getParameter("car_id"));
100     String brand = request.getParameter("brand");
101     String model = request.getParameter("model");
102     String cyclinder = request.getParameter("cyclinder");
103     String price = request.getParameter("price");
104
105     Car car = new Car(car_id, brand, model, cyclinder, price);
106     carDAO.updateCar(car);
107     response.sendRedirect("list");
108 }
109
110 private void deleteCar(HttpServletRequest request, HttpServletResponse response)
111     throws SQLException, IOException {
112     int car_id = Integer.parseInt(request.getParameter("car_id"));
113     carDAO.deleteCar(car_id);
114     response.sendRedirect("list");
115 }

```

## Car.java

```

5 package com.Model;
6
7 /**
8  *
9  * @author User
10 */
11
12 public class Car {
13     private int Car_id;
14     private String Brand;
15     private String Model;
16     private String Cyclinder;
17     private String Price;
18
19     // Constructor for insert
20     public Car(String Brand, String Model, String Cyclinder, String Price) {
21         this.Brand = Brand;
22         this.Model = Model;
23         this.Cyclinder = Cyclinder;
24         this.Price = Price;
25     }
26
27     // Constructor for update and select by ID
28     public Car(int Car_id, String Brand, String Model, String Cyclinder, String Price) {
29         this.Car_id = Car_id;
30         this.Brand = Brand;
31         this.Model = Model;
32         this.Cyclinder = Cyclinder;
33         this.Price = Price;
34     }

```

```
36 // Getters and Setters
37 public int getCar_id() {
38     return Car_id;
39 }
40
41 public void setCar_id(int Car_id) {
42     this.Car_id = Car_id;
43 }
44
45 public String getBrand() {
46     return Brand;
47 }
48
49 public void setBrand(String Brand) {
50     this.Brand = Brand;
51 }
52
53 public String getModel() {
54     return Model;
55 }
56
57 public void setModel(String model) {
58     this.Model = Model;
59 }
60
61 public String getCyclinder() {
62     return Cyclinder;
63 }
64
65 public void setCyclinder(String Cyclinder) {
66     this.Cyclinder = Cyclinder;
67 }
68
69 public String getPrice() {
70     return Price;
71 }
72
73 public void setPrice(String price) {
74     this.Price = Price;
75 }
76 }
```

## carList.jsp

```
7 <%@page contentType="text/html" pageEncoding="UTF-8"%>
8 <%@taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
9 <!--DOCTYPE html-->
10 <html>
11 <head>
12 <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
13 <title>Car Shop Management Application</title>
14 <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/css/bootstrap.min.css">
15 </head>
16 <body>
17 <div class="container">
18 <h2>Car List</h2>
19 <table class="table">
20 <thead>
21 <tr>
22 <th>ID</th>
23 <th>Brand</th>
24 <th>Model</th>
25 <th>Cylinder</th>
26 <th>Price</th>
27 <th>Actions</th>
28 </tr>
29 </thead>
30 <tbody>
31 <c:forEach var="car" items="${listCar}">
32 <tr>
33 <td>${car.car_id}</td>
34 <td>${car.brand}</td>
35 <td>${car.model}</td>
36 <td>${car.cylinder}</td>
37 <td>${car.price}</td>
38 <td>
39 <a href="edit?car_id=${car.car_id}" class="btn btn-info">Edit</a>
40 <a href="delete?car_id=${car.car_id}" class="btn btn-danger">Delete</a>
41 </td>
42 </tr>
43 </c:forEach>
44 </tbody>
45 </table>
46 <a href="new" class="btn btn-primary">Add New Car</a>
47 </div>
48 </body>
49 </html>
```

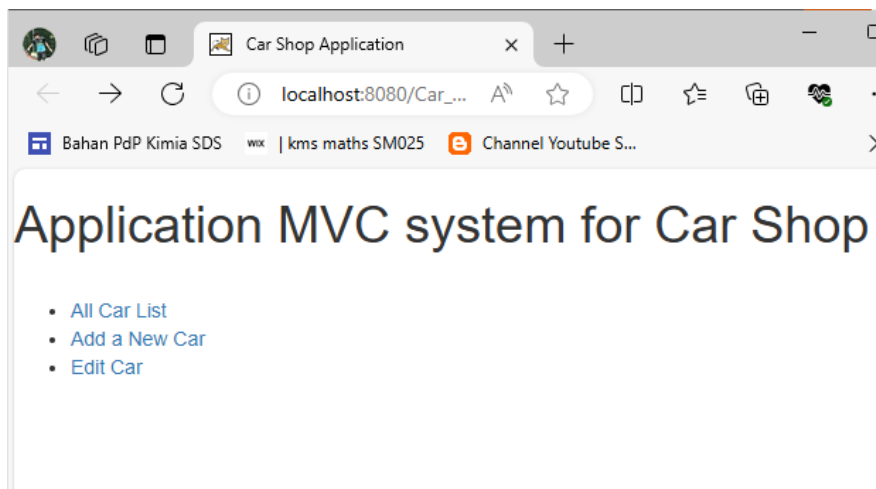
## carForm.jsp

```
7 <%@page contentType="text/html" pageEncoding="UTF-8"%>
8 <%@taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
9 <!DOCTYPE html>
10 <html>
11 <head>
12 <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
13 <title>Car Shop Management Application</title>
14 <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/css/bootstrap.min.css">
15 </head>
16 <body>
17 <div class="container">
18 <h2>${car == null ? 'Add New Car' : 'Edit Car'}</h2>
19 <form action="${car == null ? 'insert' : 'update'}" method="post">
20 <c:if test="${car != null}">
21 <input type="hidden" name="car_id" value="${car.car_id}">
22 </c:if>
23 <div class="form-group">
24 <label>Brand</label>
25 <input type="text" name="brand" class="form-control" value="${car != null ? car.brand : ''}" required>
26 </div>
27 <div class="form-group">
28 <label>Model</label>
29 <input type="text" name="model" class="form-control" value="${car != null ? car.model : ''}" required>
30 </div>
31 <div class="form-group">
32 <label>Cylinder</label>
33 <input type="text" name="cylinder" class="form-control" value="${car != null ? car.cylinder : ''}" required>
34 </div>
35 <div class="form-group">
36 <label>Price</label>
37 <input type="text" name="price" class="form-control" value="${car != null ? car.price : ''}" required>
38 </div>
39 <button type="submit" class="btn btn-primary">Submit</button>
40 </form>
41 </div>
42 </body>
43 </html>
```

## carIndex.jsp

```
1 <!--
2 Document : carIndex
3 Created on : Jun 14, 2024, 1:20:10 AM
4 Author : User
5 -->
6
7 <%@page contentType="text/html" pageEncoding="UTF-8"%>
8 <!DOCTYPE html>
9 <html>
10 <head>
11 <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
12 <title>Car Shop Application</title>
13 <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/css/bootstrap.min.css">
14 <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.7.1/jquery.min.js"></script>
15 <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/js/bootstrap.min.js"></script>
16 </head>
17 <body>
18 <h1>Application MVC system for Car Shop</h1><br>
19
20 <ul>
21 <li><a href="http://localhost:8080/Car Shop/listcar">All Car List</a></li>
22 <li><a href="http://localhost:8080/Car Shop/new">Add a New Car</a></li>
23 <li><a href="http://localhost:8080/Car Shop/listcar">Edit Car</a></li>
24 </ul>
25 </body>
26 </html>
```

The output.



- Add new car

## Add New Car

**Brand**

**Model**

**Cyclinder**

**Price**

# Car List

ID	Brand	Model	Cyclinder	Price	Actions	
6	Proton	X70	6	980000.0	Edit	Delete
7	Honda	Jazz	4	156800.0	Edit	Delete
8	Produa	Axia	3	45000.0	Edit	Delete
9	Produa	Myvi	3	55000.0	Edit	Delete

Add New Car

	car_id	brand	model	cylinder	price
▶	6	Proton	X70	6	980000
	7	Honda	Jazz	4	156800
	8	Produa	Axia	3	45000
	9	Produa	Myvi	3	55000

- Edit car

## Edit Car

**Brand**

Produa

**Model**

Axia

**Cyclinder**

5

**Price**

45000.0

Submit

## Car List

ID	Brand	Model	Cyclinder	Price	Actions	
6	Proton	X70	6	980000.0	Edit	Delete
7	Honda	Jazz	4	156800.0	Edit	Delete
8	Produa	Axia	5	45000.0	Edit	Delete
9	Produa	Myvi	3	55000.0	Edit	Delete

Add New Car

	car_id	brand	model	cylinder	price
▶	6	Proton	X70	6	980000
	7	Honda	Jazz	4	156800
	8	Produa	Axia	5	45000
	9	Produa	Myvi	3	55000

- Delete car

## Car List

ID	Brand	Model	Cyclinder	Price	Actions	
6	Proton	X70	6	980000.0	Edit	Delete
7	Honda	Jazz	4	156800.0	Edit	Delete
8	Produa	Axia	5	45000.0	Edit	Delete

Add New Car

	car_id	brand	model	cyclinder	price
▶	6	Proton	X70	6	980000
	7	Honda	Jazz	4	156800
	8	Produa	Axia	5	45000