



Week 8

# An MVC Example with Servlets and JSP

Web Programming 2

---

Name: NURHASLINDA BINTI  
BAHARUDDIN

Matric #: S67383

Semester: 4

Lab: 8

Demonstrator: SIR ARIZAL

---



Lecturers

## Task 1

com.Model

### Employee.java

```
11 public class Employee {
12     private int id;
13     private String name;
14     private String email;
15     private String position;
16
17     public Employee() {
18     }
19
20     public Employee (String name, String email, String position){
21         super();
22         this.name = name;
23         this.email = email;
24         this.position = position;
25     }
26
27     public Employee (int id, String name , String email, String position){
28         super();
29         this.id = id;
30         this.name = name;
31         this.email = email;
32         this.position = position;
33     }
34
35
36     public int getId() {
37         return id;
38     }
39
40     public void setId(int id) {
41         this.id = id;
42     }
43
44     public String getName() {
45         return name;
46     }
47
48     public void setName(String name) {
49         this.name = name;
50     }
51
52     public String getEmail() {
53         return email;
54     }
55
56     public void setEmail(String email) {
57         this.email = email;
58     }
59
60     public String getPosition() {
61         return position;
62     }
63
64     public void setPosition(String position) {
65         this.position = position;
66     }
67 }
```

com.DAO

EmployeeDAO.java

```
21 public class EmployeeDAO {
22     Connection connection=null;
23     private String jdbcURL = "jdbc:mysql://localhost:3306/company";
24     private String jdbcUsername = "root";
25     private String jdbcPassword = "admin";
26
27     private static final String INSERT_EMPLOYEES_SQL = "INSERT INTO employees (name, email, position) VALUES (?, ?, ?)";
28     private static final String SELECT_EMPLOYEE_BY_ID = "SELECT id, name, email, position from employees where id=?";
29     private static final String SELECT_ALL_EMPLOYEES = "SELECT * FROM employees";
30     private static final String DELETE_EMPLOYEES_SQL = "DELETE FROM employees where id = ?";
31     private static final String UPDATE_EMPLOYEES_SQL = "UPDATE employees set name = ?, email = ?, position = ? where id = ?";
32
33     public EmployeeDAO() {}
34
35     protected Connection getConnection(){
36         Connection connection = null;
37         try{
38             Class.forName( className: "com.mysql.jdbc.Driver");
39             connection = DriverManager.getConnection(url:jdbcURL , user: jdbcUsername , password:jdbcPassword);
40         } catch (ClassNotFoundException e){
41             e.printStackTrace();
42         } catch (SQLException e){
43             e.printStackTrace();
44         }
45         return connection;
46     }
47
48     public void insertEmployee(Employee employee) throws SQLException{
49         System.out.println(x: INSERT_EMPLOYEES_SQL);
50         try (Connection connection = getConnection();
51             PreparedStatement preparedStatement = connection.prepareStatement( sql:INSERT_EMPLOYEES_SQL)){
52
53             preparedStatement.setString( parameterIndex: 1, x:employee.getName());
54             preparedStatement.setString( parameterIndex: 2, x:employee.getEmail());
55             preparedStatement.setString( parameterIndex: 3, x:employee.getPosition());
56             System.out.println(x:preparedStatement);
57             preparedStatement.executeUpdate();
58         } catch (SQLException e) {
59             printSQLException( ex:e);
60         }
61     }
62
63     public Employee selectEmployee(int id) {
64         Employee employee = null;
65         try (Connection connection = getConnection();
66             PreparedStatement preparedStatement = connection.prepareStatement( sql:SELECT_EMPLOYEE_BY_ID)){
67             preparedStatement.setInt( parameterIndex: 1, x:id);
68             System.out.println(x:preparedStatement);
69             ResultSet rs = preparedStatement.executeQuery();
70             while (rs.next()) {
71                 String name = rs.getString( columnLabel: "name");
72                 String email = rs.getString( columnLabel: "email");
73                 String position = rs.getString( columnLabel: "position");
74                 employee = new Employee(id, name, email, position);
75             }
76         } catch (SQLException e) {
77             printSQLException( ex:e);
78         }
79         return employee;
80     }
81 }
```

```

83 public List<Employee> selectAllEmployees() {
84     List<Employee> employees = new ArrayList<>();
85     try (Connection connection = getConnection();
86         PreparedStatement preparedStatement = connection.prepareStatement("sql:SELECT_ALL_EMPLOYEES");) {
87         System.out.println(x:preparedStatement);
88         ResultSet rs = preparedStatement.executeQuery();
89         while (rs.next()) {
90             int id = rs.getInt(columnLabel: "id");
91             String name = rs.getString(columnLabel: "name");
92             String email = rs.getString(columnLabel: "email");
93             String position = rs.getString(columnLabel: "position");
94             employees.add(new Employee(id, name, email, position));
95         }
96     } catch (SQLException e) {
97         printSQLException(ex: e);
98     }
99     return employees;
100 }
101
102 public boolean deleteEmployee(int id) throws SQLException{
103     boolean rowDeleted;
104     try (Connection connection = getConnection();
105         PreparedStatement preparedStatement = connection.prepareStatement("sql:DELETE_EMPLOYEES_SQL");) {
106         preparedStatement.setInt(parameterIndex: 1, x: id);
107         rowDeleted = preparedStatement.executeUpdate() > 0;
108     }
109     return rowDeleted;
110 }

```

```

111
112 public boolean updateEmployee(Employee employee) throws SQLException{
113     boolean rowUpdated;
114     try (Connection connection = getConnection();
115         PreparedStatement preparedStatement = connection.prepareStatement("sql:UPDATE_EMPLOYEES_SQL");) {
116         preparedStatement.setString(parameterIndex: 1, x: employee.getName());
117         preparedStatement.setString(parameterIndex: 2, x: employee.getEmail());
118         preparedStatement.setString(parameterIndex: 3, x: employee.getPosition());
119         preparedStatement.setInt(parameterIndex: 4, x: employee.getId());
120         rowUpdated = preparedStatement.executeUpdate() > 0;
121     }
122     return rowUpdated;
123 }
124
125 private void printSQLException(SQLException ex) {
126     for(Throwable e : ex){
127         if(e instanceof SQLException){
128             e.printStackTrace(System.err);
129             System.out.println("SQLState : " + ((SQLException) e).getSQLState());
130             System.out.println("Error Code : " + ((SQLException) e).getErrorCode());
131             System.out.println("Message : " + e.getMessage());
132             Throwable t = ex.getCause();
133             while(t != null){
134                 System.out.println("Cause: " + t);
135                 t = t.getCause();
136             }
137         }
138     }
139 }
140 }

```

## Com.WEB

### EmployeeServlet.java

```
25 @WebServlet("/")
26 public class EmployeeServlet extends HttpServlet {
27
28     private EmployeeDAO employeeDAO;
29
30     public void init() {
31         employeeDAO = new EmployeeDAO();
32     }
33
34     @Override
35     protected void doPost(HttpServletRequest request, HttpServletResponse response)
36         throws ServletException, IOException {
37         doGet(request, response);
38     }
39
40     @Override
41     protected void doGet(HttpServletRequest request, HttpServletResponse response)
42         throws ServletException, IOException {
43         String action = request.getServletPath();
44
45         try {
46             switch(action) {
47                 case "/new":
48                     showNewForm(request, response);
49                     break;
50                 case "/insert":
51                     insertEmployee(request, response);
52                     break;
53                 case "/delete":
54                     deleteEmployee(request, response);
55                     break;
56                 case "/edit":
57                     showEditForm(request, response);
58                     break;
59                 case "/update":
60                     updateEmployee(request, response);
61                     break;
```

```
62                 default:
63                     listEmployee(request, response);
64                     break;
65             }
66         } catch (SQLException ex) {
67             throw new ServletException( rootCause: ex );
68         }
69     }
70
71     private void listEmployee(HttpServletRequest request, HttpServletResponse response)
72         throws SQLException, ServletException, IOException {
73         List<Employee> listEmployee = employeeDAO.selectAllEmployees();
74         request.setAttribute( name: "listEmployee", o: listEmployee);
75         RequestDispatcher dispatcher = request.getRequestDispatcher( path: "employeeList.jsp");
76         dispatcher.forward(request, response);
77     }
78
79     private void showNewForm(HttpServletRequest request, HttpServletResponse response)
80         throws ServletException, IOException {
81         RequestDispatcher dispatcher = request.getRequestDispatcher( path: "employeeForm.jsp");
82         dispatcher.forward(request, response);
83     }
84
85     private void showEditForm(HttpServletRequest request, HttpServletResponse response)
86         throws SQLException, ServletException, IOException {
87         int id = Integer.parseInt( s: request.getParameter( name: "id"));
88         Employee existingEmployee = employeeDAO.selectEmployee(id);
89         RequestDispatcher dispatcher = request.getRequestDispatcher( path: "employeeForm.jsp");
90         request.setAttribute( name: "employee", o: existingEmployee);
91         dispatcher.forward(request, response);
92     }
93 }
```

```

94     private void insertEmployee(HttpServletRequest request, HttpServletResponse response)
95     {
96         throws SQLException, IOException {
97             String name = request.getParameter( name: "name");
98             String email = request.getParameter( name: "email");
99             String position = request.getParameter( name: "position");
100             Employee newEmployee = new Employee(name, email, position);
101             employeeDAO.insertEmployee( employee: newEmployee);
102             response.sendRedirect( location: "list");
103         }
104     }
105
106     private void updateEmployee(HttpServletRequest request, HttpServletResponse response)
107     {
108         throws SQLException, IOException {
109             int id = Integer.parseInt( s: request.getParameter( name: "id"));
110             String name = request.getParameter( name: "name");
111             String email = request.getParameter( name: "email");
112             String position = request.getParameter( name: "position");
113             Employee employee = new Employee(id, name, email, position);
114             employeeDAO.updateEmployee(employee);
115             response.sendRedirect( location: "list");
116         }
117     }
118
119     private void deleteEmployee(HttpServletRequest request, HttpServletResponse response)
120     {
121         throws SQLException, IOException {
122             int id = Integer.parseInt( s: request.getParameter( name: "id"));
123             employeeDAO.deleteEmployee(id);
124             response.sendRedirect( location: "list");
125         }
126     }
127 }

```

## index.jsp

```

7     <%@page contentType="text/html" pageEncoding="UTF-8"%>
8     <!DOCTYPE html>
9     <html>
10     <head>
11         <title>User Management Application</title>
12         <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.min.css"
13             integrity="sha384-MCW98/STnGE8fJT3GXwEOngsV7Zt27NXXF0aoApmYm8liuXoPkFOJwJ8ERdknLPMO" crossorigin="anonymous">
14     </head>
15     <body>
16         <h1>Application MVC system for Employee Management</h1>
17         <br>
18         <ul>
19             <li><a href="http://localhost:8080/Employee Management/list">All Employee List</a></li>
20             <li><a href="http://localhost:8080/Employee Management/new">Add a Employee List</a></li>
21             <li><a href="http://localhost:8080/Employee Management/list">Edit Employee</a></li>
22         </ul>
23     </body>
24 </html>
25
26

```

## employeeForm.jsp

```

7     <%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>
8     <%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
9
10     <!DOCTYPE html>
11     <html>
12     <head>
13         <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
14         <title>Employee Management Application</title>
15         <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.min.css"
16             integrity="sha384-MCW98/STnGE8fJT3GXwEOngsV7Zt27NXXF0aoApmYm8liuXoPkFOJwJ8ERdknLPMO" crossorigin="anonymous">
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                 <ul class="navbar-nav">
25                     <li><a href="<%=request.getContextPath()%>/list" class="nav-link">Employees</a></li>
26                 </ul>
27             </nav>
28         </header>

```

```

29 <br>
30 <div class="container col-md-5">
31   <div class="card">
32     <div class="card-body">
33       <c:set var="formAction" value="${employee != null ? 'update' : 'insert'}" />
34       <form action="${formAction}" method="post">
35
36         <h2>
37           <c:if test="${employee != null}">
38             Edit Employee
39           </c:if>
40           <c:if test="${employee == null}">
41             Add New Employee
42           </c:if>
43         </h2>
44
45         <c:if test="${employee != null}">
46           <input type="hidden" name="id" value="<c:out value='${employee.id}' />" />
47         </c:if>
48
49         <fieldset class="form-group">
50           <label>Employee Name</label>
51           <input type="text" value="<c:out value='${employee.name}' />" class="form-control" name="name" required="required">
52         </fieldset>
53
54         <fieldset class="form-group">
55           <label>Employee E-mail</label>
56           <input type="text" value="<c:out value='${employee.email}' />" class="form-control" name="email">
57         </fieldset>
58
59         <fieldset class="form-group">
60           <label>Employee Position</label>
61           <input type="text" value="<c:out value='${employee.position}' />" class="form-control" readonly>
62           <input list="positionList" id="position" class="form-control" name="position">
63           <datalist id="positionList">
64             <option value="Manager">
65             <option value="Head of Dept">
66             <option value="Supervisor">
67             <option value="Director">
68             <option value="INA">
69           </datalist>
70         </fieldset>
71
72         <button type="submit" class="btn btn-success">Save</button>
73       </form>
74     </div>
75   </div>
76 </div>
77 </body>
78 </html>
79

```

## employeeList.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>Employee Management Application</title>
14     <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.min.css"
15           integrity="sha384-MCW98/ST567F6SPjO16176Pk7f3d7xM0nPW+98z9M9U2ADYG87389M3VihR8" crossorigin="anonymous">
16   </head>
17   <body>
18     <header>
19       <nav class="navbar navbar-expand-md navbar-dark" style="background-color:tomato">
20         <div>
21           <a href="#" class="navbar-brand">Employee Management App </a>
22         </div>
23
24         <ul class="navbar-nav">
25           <li><a href="<%=request.getContextPath()%>/list" class="nav-link">Employees</a></li>
26         </ul>
27       </nav>
28     </header>

```





Output:  
Create

localhost / 127.0.0.1 / compan...Employee Management Appli...+  
localhost:8080/Employee\_Management/employeeform.jsp

Employee Management AppEmployees

Add New Employee

Employee Name  
Cristiano Ronaldo

Employee E-mail  
cr7@gmail.com

Employee Position  
Director

Save

View

localhost / 127.0.0.1 / compan...Employee Management Appli...+  
localhost:8080/Employee\_Management/list

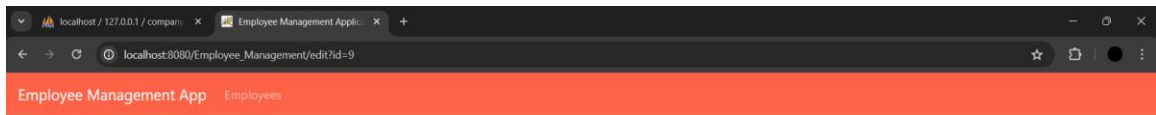
Employee Management AppEmployees

List of Employees

Add New Employee

ID	Name	Email	Position	Action
1	muh yus	myus@project164.com	INA	<a href="#">Edit</a> <a href="#">Delete</a>
2	yus yus	yusyus@project164.com	Director	<a href="#">Edit</a> <a href="#">Delete</a>
3	muhayus	c@ict.net	Director	<a href="#">Edit</a> <a href="#">Delete</a>
4	yusro	Bambang@gmail.co.id	Head of Dept	<a href="#">Edit</a> <a href="#">Delete</a>
7	Fouad	fouad@gmail.com	Supervisor	<a href="#">Edit</a> <a href="#">Delete</a>
8	Lim Jun Eng	limjun7@gmail.com	Head of Dept	<a href="#">Edit</a> <a href="#">Delete</a>
9	Cristiano Ronaldo	cr7@gmail.com	Director	<a href="#">Edit</a> <a href="#">Delete</a>

Update



### List of Employees

Add New Employee

ID	Name	Email	Position	Action
1	muh yus	myus@project164.com	INA	<a href="#">Edit</a> <a href="#">Delete</a>
2	yus yus	yusus@project164.com	Director	<a href="#">Edit</a> <a href="#">Delete</a>
3	muhayus	c@ict.net	Director	<a href="#">Edit</a> <a href="#">Delete</a>
4	yusro	Bambang@gmail.co.id	Head of Dept	<a href="#">Edit</a> <a href="#">Delete</a>
7	Fouad	fouad@gmail.com	Supervisor	<a href="#">Edit</a> <a href="#">Delete</a>
8	Lim Jun Eng	limjun7@gmail.com	Head of Dept	<a href="#">Edit</a> <a href="#">Delete</a>
9	C. Ronaldo	cronaldo7@gmail.com	Manager	<a href="#">Edit</a> <a href="#">Delete</a>

Name, email and position of employee number 9 had been changed

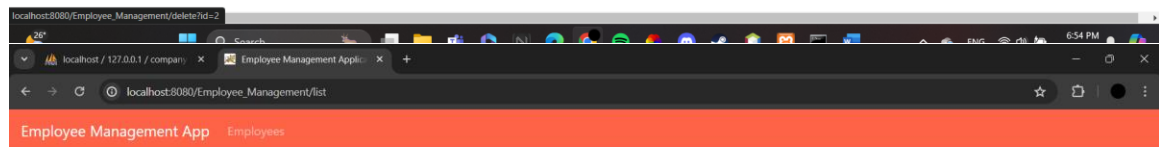
## Delete (Employee no.2)



### List of Employees

Add New Employee

ID	Name	Email	Position	Action
1	muh yus	myus@project164.com	INA	<a href="#">Edit</a> <a href="#">Delete</a>
2	yus yus	yusyus@project164.com	Director	<a href="#">Edit</a> <a href="#">Delete</a>
3	muhayus	c@ict.net	Director	<a href="#">Edit</a> <a href="#">Delete</a>
4	yusro	Bambang@gmail.co.id	Head of Dept	<a href="#">Edit</a> <a href="#">Delete</a>
7	Fouad	fouad@gmail.com	Supervisor	<a href="#">Edit</a> <a href="#">Delete</a>
8	Lim Jun Eng	limjun7@gmail.com	Head of Dept	<a href="#">Edit</a> <a href="#">Delete</a>
9	C. Ronaldo	cronaldo7@gmail.com	Manager	<a href="#">Edit</a> <a href="#">Delete</a>



### List of Employees

Add New Employee

ID	Name	Email	Position	Action
1	muh yus	myus@project164.com	INA	<a href="#">Edit</a> <a href="#">Delete</a>
3	muhayus	c@ict.net	Director	<a href="#">Edit</a> <a href="#">Delete</a>
4	yusro	Bambang@gmail.co.id	Head of Dept	<a href="#">Edit</a> <a href="#">Delete</a>
7	Fouad	fouad@gmail.com	Supervisor	<a href="#">Edit</a> <a href="#">Delete</a>
8	Lim Jun Eng	limjun7@gmail.com	Head of Dept	<a href="#">Edit</a> <a href="#">Delete</a>
9	C. Ronaldo	cronaldo7@gmail.com	Manager	<a href="#">Edit</a> <a href="#">Delete</a>

## Exercise

### com.Model

#### Car.java

```
11 public class Car {
12     private int carId;
13     private String brand;
14     private String model;
15     private int cylinder;
16     private double price;
17
18     public Car() {
19     }
20
21     public Car (String brand, String model, int cylinder, double price) {
22         super();
23         this.brand = brand;
24         this.model = model;
25         this.cylinder = cylinder;
26         this.price = price;
27     }
28
29     public Car (int carId, String brand, String model, int cylinder, double price) {
30         super();
31         this.carId = carId;
32         this.brand = brand;
33         this.model = model;
34         this.cylinder = cylinder;
35         this.price = price;
36     }
37 }
```

```
38     public int getCarId() {
39         return carId;
40     }
41
42     public void setCarId(int carId) {
43         this.carId = carId;
44     }
45
46     public String getBrand() {
47         return brand;
48     }
49
50     public void setBrand(String brand) {
51         this.brand = brand;
52     }
53
54     public String getModel() {
55         return model;
56     }
57
58     public void setModel(String model) {
59         this.model = model;
60     }
```

```
62     public int getCylinder() {
63         return cylinder;
64     }
65
66     public void setCylinder(int cylinder) {
67         this.cylinder = cylinder;
68     }
69
70     public double getPrice() {
71         return price;
72     }
73
74     public void setPrice(double price) {
75         this.price = price;
76     }
77
78
79 }
80 }
```

## com.DAO CarDAO.java

```
21 public class CarDAO {
22     Connection connection=null;
23     private String jdbcURL = "jdbc:mysql://localhost:3306/carshop";
24     private String jdbcUsername = "root";
25     private String jdbcPassword = "admin";
26
27     private static final String INSERT_CARS_SQL = "INSERT INTO carpricelist (brand, model, cyclinder, price) VALUES (?, ?, ?, ?)";
28     private static final String SELECT_CARS_BY_ID = "SELECT carid, brand, model, cyclinder, price from carpricelist where carid=?";
29     private static final String SELECT_ALL_CARS = "SELECT * FROM carpricelist";
30     private static final String DELETE_CARS_SQL = "DELETE FROM carpricelist where carid = ?";
31     private static final String UPDATE_CARS_SQL = "UPDATE carpricelist set brand = ?, model = ?, cyclinder = ?, price = ? where carid = ?";
32
33     public CarDAO() {}
34
35     protected Connection getConnection(){
36         Connection connection = null;
37         try{
38             Class.forName("com.mysql.jdbc.Driver");
39             connection = DriverManager.getConnection(uri:jdbcURL , user:jdbcUsername , password:jdbcPassword);
40         } catch (ClassNotFoundException e) {
41             e.printStackTrace();
42         } catch (SQLException e){
43             e.printStackTrace();
44         }
45         return connection;
46     }
47
48     public void insertCar(Car car) throws SQLException{
49         System.out.println(x:INSERT_CARS_SQL);
50         try (Connection connection = getConnection();
51             PreparedStatement preparedStatement = connection.prepareStatement(sql:INSERT_CARS_SQL);){
52
53             preparedStatement.setString(parameterIndex: 1, x:car.getBrand());
54             preparedStatement.setString(parameterIndex: 2, x:car.getModel());
55             preparedStatement.setInt(parameterIndex: 3, x:car.getCyclinder());
56             preparedStatement.setDouble(parameterIndex: 4, x:car.getPrice());
57             System.out.println(x:preparedStatement);
58             preparedStatement.executeUpdate();
59         } catch (SQLException e) {
60             printSQLException(e);
61         }
62     }
63
64     public Car selectCar(int carId) {
65         Car car = null;
66         try (Connection connection = getConnection();
67             PreparedStatement preparedStatement = connection.prepareStatement(sql:SELECT_CARS_BY_ID);){
68             preparedStatement.setInt(parameterIndex: 1, x:carId);
69             System.out.println(x:preparedStatement);
70             ResultSet rs = preparedStatement.executeQuery();
71             while (rs.next()) {
72                 String brand = rs.getString(columnLabel: "brand");
73                 String model = rs.getString(columnLabel: "model");
74                 int cyclinder = rs.getInt(columnLabel: "cyclinder");
75                 double price = rs.getDouble(columnLabel: "price");
76                 car = new Car(carId, brand, model, cyclinder, price);
77             }
78         } catch (SQLException e) {
79             printSQLException(e);
80         }
81         return car;
82     }
83 }
```

```

85 public List<Car> selectAllCars() {
86     List<Car> cars = new ArrayList<>();
87     try (Connection connection = getConnection();
88         PreparedStatement preparedStatement = connection.prepareStatement(sql: SELECT_ALL_CARS);) {
89         System.out.println(x: preparedStatement);
90         ResultSet rs = preparedStatement.executeQuery();
91         while (rs.next()) {
92             int carId = rs.getInt(columnLabel: "carId");
93             String brand = rs.getString(columnLabel: "brand");
94             String model = rs.getString(columnLabel: "model");
95             int cylcylinder = rs.getInt(columnLabel: "cylcylinder");
96             double price = rs.getDouble(columnLabel: "price");
97             cars.add(new Car(carId, brand, model, cylcylinder, price));
98         }
99     } catch (SQLException e) {
100         printSQLException(ex: e);
101     }
102     return cars;
103 }
104
105 public boolean deleteCar(int carId) throws SQLException{
106     boolean rowDeleted;
107     try (Connection connection = getConnection();
108         PreparedStatement preparedStatement = connection.prepareStatement(sql: DELETE_CARS_SQL);) {
109         preparedStatement.setInt(parameterIndex: 1, x: carId);
110         rowDeleted = preparedStatement.executeUpdate() > 0;
111     }
112     return rowDeleted;
113 }
114
115 public boolean updateCar(Car car) throws SQLException{
116     boolean rowUpdated;
117     try (Connection connection = getConnection();
118         PreparedStatement preparedStatement = connection.prepareStatement(sql: UPDATE_CARS_SQL);) {
119         preparedStatement.setString(parameterIndex: 1, x: car.getBrand());
120         preparedStatement.setString(parameterIndex: 2, x: car.getModel());
121         preparedStatement.setInt(parameterIndex: 3, x: car.getCylcylinder());
122         preparedStatement.setDouble(parameterIndex: 4, x: car.getPrice());
123         preparedStatement.setInt(parameterIndex: 5, x: car.getCarId());
124         rowUpdated = preparedStatement.executeUpdate() > 0;
125     }
126     return rowUpdated;
127 }
128
129 private void printSQLException(SQLException ex) {
130     for(Throwable e : ex) {
131         if(e instanceof SQLException) {
132             e.printStackTrace(System.err);
133             System.out.println("SQLState : " + ((SQLException) e).getSQLState());
134             System.out.println("Error Code : " + ((SQLException) e).getErrorCode());
135             System.out.println("Message : " + e.getMessage());
136             Throwable t = ex.getCause();
137             while(t != null) {
138                 System.out.println("Cause: " + t);
139                 t = t.getCause();
140             }
141         }
142     }
143 }
144 }

```

## com.WEB CarServlet.java

```
23 @WebServlet("/")
24 public class CarServlet extends HttpServlet {
25
26     private CarDAO carDAO;
27
28     public void init() {
29         carDAO = new CarDAO();
30     }
31
32     @Override
33     protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
34         doGet(request, response);
35     }
36
37     @Override
38     protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
39         String action = request.getServletPath();
40
41         try {
42             switch (action) {
43                 case "/new":
44                     showNewForm(request, response);
45                     break;
46                 case "/insert":
47                     insertCar(request, response);
48                     break;
49                 case "/delete":
50                     deleteCar(request, response);
51                     break;
52                 case "/edit":
53                     showEditForm(request, response);
54                     break;
55                 case "/update":
56                     updateCar(request, response);
57                     break;
58                 default:
59                     listCar(request, response);
60                     break;
61             }
62         } catch (SQLException ex) {
63             throw new ServletException("rootCause:ex");
64         }
65     }
66
67     private void listCar(HttpServletRequest request, HttpServletResponse response) throws SQLException, ServletException, IOException {
68         List<Car> listCar = carDAO.selectAllCars();
69         request.setAttribute("listCar", listCar);
70         RequestDispatcher rd = request.getRequestDispatcher("path:carList.jsp");
71         rd.forward(request, response);
72     }
73
74     private void showNewForm(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
75         RequestDispatcher rd = request.getRequestDispatcher("path:carForm.jsp");
76         rd.forward(request, response);
77     }
78
79     private void showEditForm(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
80         int carId = Integer.parseInt(request.getParameter("carId"));
81         Car existingCar = carDAO.selectCar(carId);
82         RequestDispatcher rd = request.getRequestDispatcher("path:carForm.jsp");
83         request.setAttribute("car", existingCar);
84         rd.forward(request, response);
85     }
86
87     private void insertCar(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
88         String brand = request.getParameter("brand");
89         String model = request.getParameter("model");
90         int cylinder = Integer.parseInt(request.getParameter("cylinder"));
91         double price = Double.parseDouble(request.getParameter("price"));
92         Car newCar = new Car(brand, model, cylinder, price);
93         carDAO.insertCar(newCar);
94         response.sendRedirect("location:list");
95     }
```

```

97     private void updateCar(HttpServletRequest request, HttpServletResponse response) throws SQLException, IOException {
98         int carId = Integer.parseInt(s:request.getParameter(name:"carId"));
99         String brand = request.getParameter(name:"brand");
100        String model = request.getParameter(name:"model");
101        int cylinder = Integer.parseInt(s:request.getParameter(name:"cylinder"));
102        double price = Double.parseDouble(s:request.getParameter(name:"price"));
103
104        Car car = new Car(carId, brand, model, cylinder, price);
105        carDAO.updateCar(car);
106        response.sendRedirect(location:"list");
107    }
108
109    private void deleteCar(HttpServletRequest request, HttpServletResponse response) throws SQLException, IOException {
110        int carId = Integer.parseInt(s:request.getParameter(name:"carId"));
111        carDAO.deleteCar(carId);
112        response.sendRedirect(location:"list");
113    }
114 }

```

## CarForm.jsp

```

10 <html>
11 <head>
12 <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
13 <title>Car Management Application</title>
14 <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css">
15 <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/bootstrap.bundle.min.js"></script>
16 </head>
17
18 <body>
19 <header>
20 <nav class="navbar navbar-expand-md navbar-dark" style="background-color: #6F7378">
21 <div>
22 <a href="carForm.jsp" class="navbar-brand">Car Manager</a>
23 </div>
24 <ul class="navbar-nav">
25 <li><a href="<%=request.getContextPath()%>/list" class="nav-link">Car List</a></li>
26 </ul>
27 </nav>
28 </header>
29 <br>
30 <div class="container col-md-8">
31 <div class="card">
32 <div class="card-body">
33 <:set var="formAction" value="{car != null ? 'update' : 'insert'}" />
34 <form action="{formAction}" method="post">
35
36 <h2>
37 <:if test="{car != null}">
38 Edit Car
39 </:if>
40 <:if test="{car == null}">
41 Add New Car
42 </:if>
43 </h2>
44
45 <:if test="{car != null}">
46 <input type="hidden" name="carId" value="{car.id value='{car.carId}' />"/>
47 </:if>
48
49 <fieldset class="form-group">
50 <label>Car Brand</label>
51 <input type="text" value="{car.brand value='{car.brand}' />"/> class="form-control" name="brand" required="required"/>
52 </fieldset>
53
54 <fieldset class="form-group">
55 <label>Car Model</label>
56 <input type="text" value="{car.model value='{car.model}' />"/> class="form-control" name="model" required="required"/>
57 </fieldset>
58
59 <fieldset class="form-group">
60 <label>Car cylinder</label>
61 <input type="text" value="{car.cylinder value='{car.cylinder}' />"/> class="form-control" name="cylinder" required="required"/>
62 </fieldset>
63
64 <fieldset class="form-group">
65 <label>Car Price</label>
66 <input type="text" value="{car.price value='{car.price}' />"/> class="form-control" name="price" required="required"/>
67 </fieldset>
68 <br>
69 <button type="submit" class="btn btn-success">Save</button>
70 </form>
71 </div>
72 </div>
73 </div>
74 </body>
75 </html>

```



## CarList.jsp

[illegible]

Output

Create

localhost / 127.0.0.1 / company

Car Management Application

localhost:8080/Car\_Shop/carform.jsp

Car ManagerCar List

Add New Car

Car Brand

Proton

Car Model

X50

Car Cyclinder

3

Car Price

100000

Save

View

localhost / 127.0.0.1 / company

Car Management Application

localhost:8080/Car\_Shop/list

Car ManagerCar List

List of Cars

Add New Car

Car ID	Brand	Model	Cyclinder	Price	
1	Proton	Saga	2	20000.0	<a href="#">Edit</a> <a href="#">Delete</a>
4	Perodua	Bezza	2	12000.0	<a href="#">Edit</a> <a href="#">Delete</a>
5	Proton	X50	3	100000.0	<a href="#">Edit</a> <a href="#">Delete</a>

Update



### Edit Car

Car Brand

Perodua

Car Model

Bezza

Car Cyclinder

2

Car Price

14000.0

Save



List of Cars

Add New Car

Car ID	Brand	Model	Cyclinder	Price	
1	Proton	Saga	2	20000.0	<a href="#">Edit</a> <a href="#">Delete</a>
4	Perodua	Bezza	2	14000.0	<a href="#">Edit</a> <a href="#">Delete</a>
5	Proton	X50	3	100000.0	<a href="#">Edit</a> <a href="#">Delete</a>

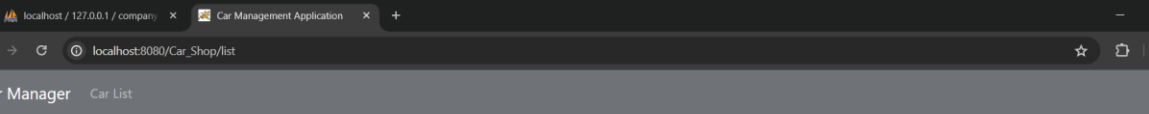
Delete (Car 1 Proton Saga)



List of Cars

Add New Car

Car ID	Brand	Model	Cyclinder	Price	
1	Proton	Saga	2	20000.0	<a href="#">Edit</a> <a href="#">Delete</a>
4	Perodua	Bezza	2	14000.0	<a href="#">Edit</a> <a href="#">Delete</a>
5	Proton	X50	3	100000.0	<a href="#">Edit</a> <a href="#">Delete</a>



### List of Cars

Add New Car

Car ID	Brand	Model	Cylinder	Price	
4	Perodua	Bezza	2	14000.0	<a href="#">Edit</a> <a href="#">Delete</a>
5	Proton	X50	3	100000.0	<a href="#">Edit</a> <a href="#">Delete</a>