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

```java
public class Employee {
    private int id;
    private String name;
    private String email;
    private 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;
    }
}
```

com.DAO
EmployeeDAO.java

```java
public class EmployeeDAO {
    Connection connection=null;
    private String jdbcURL = "jdbc:mysql://localhost:3306/company";
    private String jdbcUsername = "root";
    private String jdbcPassword = "admin";

    private static final String INSERT_EMPLOYEES_SQL = "INSERT INTO employees (name, email, position) VALUES (?, ?, ?)";
    private static final String SELECT_EMPLOYEE_BY_ID = "SELECT id, name, email, position from employees where id=?";
    private static final String SELECT_ALL_EMPLOYEES = "SELECT * FROM employees";
    private static final String DELETE_EMPLOYEES_SQL = "DELETE FROM employees where id = ?";
    private static final String UPDATE_EMPLOYEES_SQL = "UPDATE employees set name = ?, email = ?, position = ? where id = ?";

    public EmployeeDAO() {}

    protected Connection getConnection(){
        Connection connection = null;
        try{
            Class.forName( className: "com.mysql.jdbc.Driver");
            connection = DriverManager.getConnection( url: jdbcURL , user: jdbcUsername ,  password: jdbcPassword);
        } catch (ClassNotFoundException e){
            e.printStackTrace();
        } catch (SQLException e){
            e.printStackTrace();
        }
        return connection;
    }
```

```java
    public void insertEmployee(Employee employee) throws SQLException{
        System.out.println( x: INSERT_EMPLOYEES_SQL);
        try (Connection connection = getConnection();
            PreparedStatement preparedStatement = connection.prepareStatement( sql: INSERT_EMPLOYEES_SQL);){

            preparedStatement.setString( parameterIndex: 1,  x: employee.getName());
            preparedStatement.setString( parameterIndex: 2,  x: employee.getEmail());
            preparedStatement.setString( parameterIndex: 3,  x: employee.getPosition());
            System.out.println( x: preparedStatement);
            preparedStatement.executeUpdate();
        } catch (SQLException e) {
            printSQLException( ex: e);
        }
    }

    public Employee selectEmployee(int id) {
        Employee employee = null;
        try (Connection connection = getConnection();
            PreparedStatement preparedStatement = connection.prepareStatement( sql: SELECT_EMPLOYEE_BY_ID);){
            preparedStatement.setInt( parameterIndex: 1,  x: id);
            System.out.println( x: preparedStatement);
            ResultSet rs = preparedStatement.executeQuery();
            while (rs.next()) {
                String name = rs.getString( columnLabel: "name");
                String email = rs.getString( columnLabel: "email");
                String position = rs.getString( columnLabel: "position");
                employee = new Employee(id, name, email, position);
            }
        } catch (SQLException e) {
            printSQLException( ex: e);
        }
        return employee;
    }
```

```
    public List<Employee> selectAllEmployees() {
        List<Employee> employees = new ArrayList<>();
        try (Connection connection = getConnection();
            PreparedStatement preparedStatement = connection.prepareStatement( sql: SELECT_ALL_EMPLOYEES);){
            System.out.println( x: preparedStatement);
            ResultSet rs = preparedStatement.executeQuery();
            while (rs.next()) {
                int id = rs.getInt( columnLabel: "id");
                String name = rs.getString( columnLabel: "name");
                String email = rs.getString( columnLabel: "email");
                String position = rs.getString( columnLabel: "position");
                employees.add(new Employee(id, name, email, position));
            }
        } catch (SQLException e) {
            printSQLException( ex: e);
        }
        return employees;
    }

    public boolean deleteEmployee(int id) throws SQLException{
        boolean rowDeleted;
        try (Connection connection = getConnection();
            PreparedStatement preparedStatement = connection.prepareStatement( sql: DELETE EMPLOYEES SQL);){
            preparedStatement.setInt( parameterIndex: 1,  x: id);
            rowDeleted = preparedStatement.executeUpdate() > 0;
        }
        return rowDeleted;
    }
```

```

    public boolean updateEmployee(Employee employee) throws SQLException{
        boolean rowUpdated;
        try (Connection connection = getConnection();
            PreparedStatement preparedStatement = connection.prepareStatement( sql: UPDATE_EMPLOYEES_SQL);){
            preparedStatement.setString( parameterIndex: 1,  x: employee.getName());
            preparedStatement.setString( parameterIndex: 2,  x: employee.getEmail());
            preparedStatement.setString( parameterIndex: 3,  x: employee.getPosition());
            preparedStatement.setInt( parameterIndex: 4,  x: employee.getId());
            rowUpdated = preparedStatement.executeUpdate() > 0;
        }
        return rowUpdated;
    }

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

Com.WEB

EmployeeServlet.java

```java
@WebServlet("/")
public class EmployeeServlet extends HttpServlet {

    private EmployeeDAO employeeDAO;

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

    @Override
    protected void doPost(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        doGet(request, response);
    }

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

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

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

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

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

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

        private void updateEmployee(HttpServletRequest request, HttpServletResponse response)
                throws SQLException, IOException {
            int id = Integer.parseInt( s: request.getParameter( name: "id"));
            String name = request.getParameter( name: "name");
            String email = request.getParameter( name: "email");
            String position = request.getParameter( name: "position");
            Employee employee = new Employee(id, name, email, position);
            employeeDAO.updateEmployee(employee);
            response.sendRedirect( location: "list");
        }

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

```

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/4.1.3/css/bootstrap.min.css"
              integrity="sha384-MCw98/SFnGE8fJT3GXwEOngsV7Zt27NXFoaoApmYm81iuXoPkFOJwJ8ERdknLPMO" 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 Employee List</a></li>
            <li><a href="http://localhost:8080/Employee_Management/list">Edit Employee</a></li>
        </ul>
    </body>
</html>

```

employeeForm.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>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <title>Employee Management Application</title>
    <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.min.css"
          integrity="sha384-MCw98/SFnGE8fJT3GXwEOngsV7Zt27NXFoaoApmYm81iuXoPkFOJwJ8ERdknLPMO" 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>
```

```jsp
<br>
<div class="container col-md-5">
    <div class="card">
        <div class="card-body">
            <c:set var="formAction" value="${employee != null ? 'update' : 'insert'}" />
            <form action="${formAction}" method="post">

                <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 E-mail</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">
                            <option value="INA">
                        </datalist>
                    </fieldset>

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

employeeList.jsp

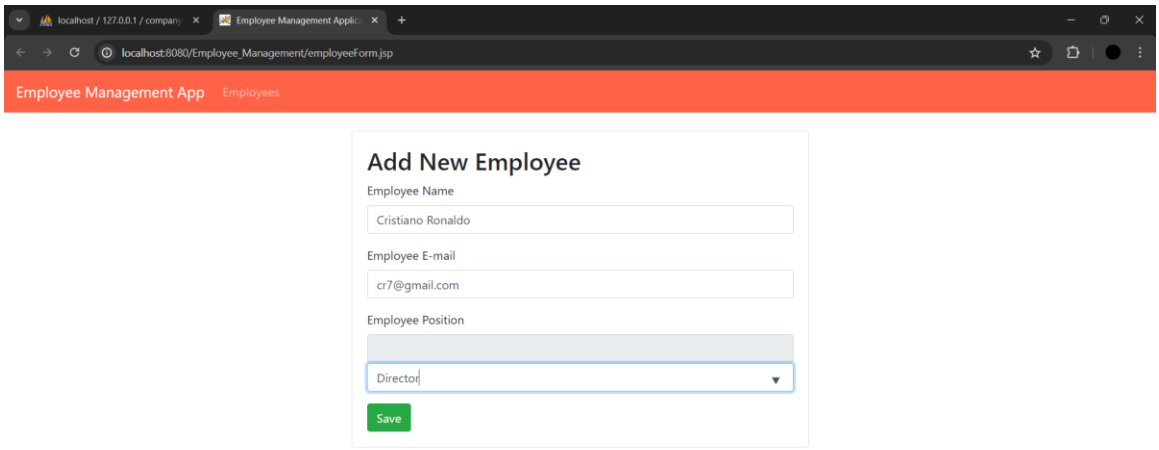
```jsp
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<!DOCTYPE html>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
        <title>Employee Management Application</title>
        <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.min.css"
              integrity="sha384-MCw98/SFnGE8fJT3GXwEOngsV7Zt27NXFoaoApmYm81iuXoPkFOJwJ8ERdknLPMO" 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="col">
            <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>Action</th>
                        </tr>
                    </thead>
                    <tbody>
                        <c:forEach var="employee" items="${listEmployee}">
                            <tr>
                                <td>
                                    <c:out value="${employee.id}"/>
                                </td>
                                <td>
                                    <c:out value="${employee.name}"/>
                                </td>
                                <td>
                                    <c:out value="${employee.email}"/>
                                </td>
                                <td>
                                    <c:out value="${employee.position}"/>
                                </td>
                                <td><a href="edit?id=<c:out value='${employee.id}'/>">Edit</a>     
                                    <a href="delete?id=<c:out value='${employee.id}'/>">Delete</a></td>
                            </tr>
                        </c:forEach>
                    </tbody>
                </table>
            </div>
        </div>
    </body>
</html>
```

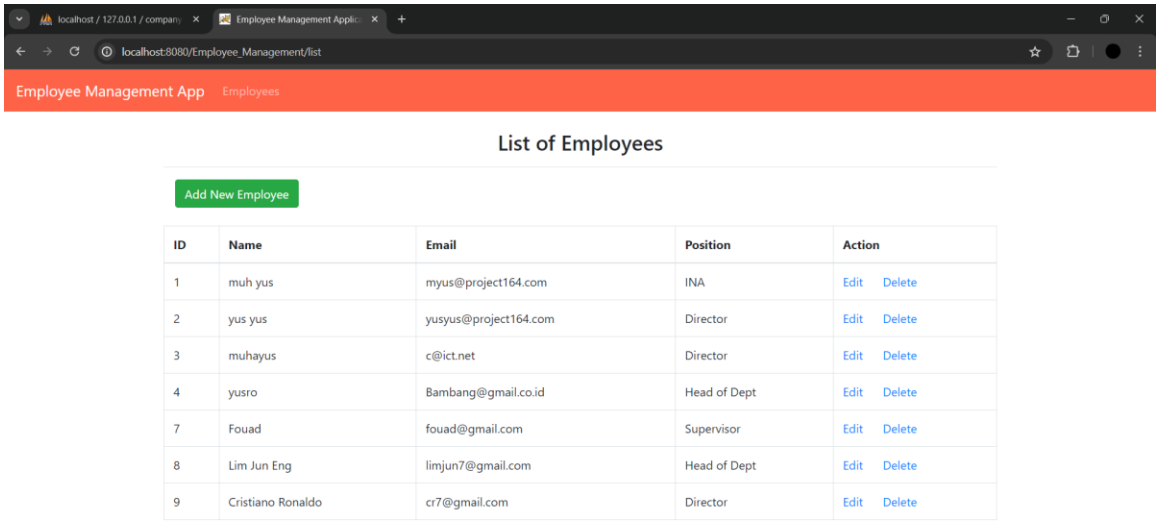
Error.jsp

```
<%@page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8" isErrorPage="true"%>
<!DOCTYPE html>
<html>
    <head>

        <title>Error Page</title>
    </head>
    <body>
    <center>
        <h1>Error</h1>
        <h2><%=exception.getMessage()%><br/></h2>
    </center>
    </body>
</html>
```

Output:
Create

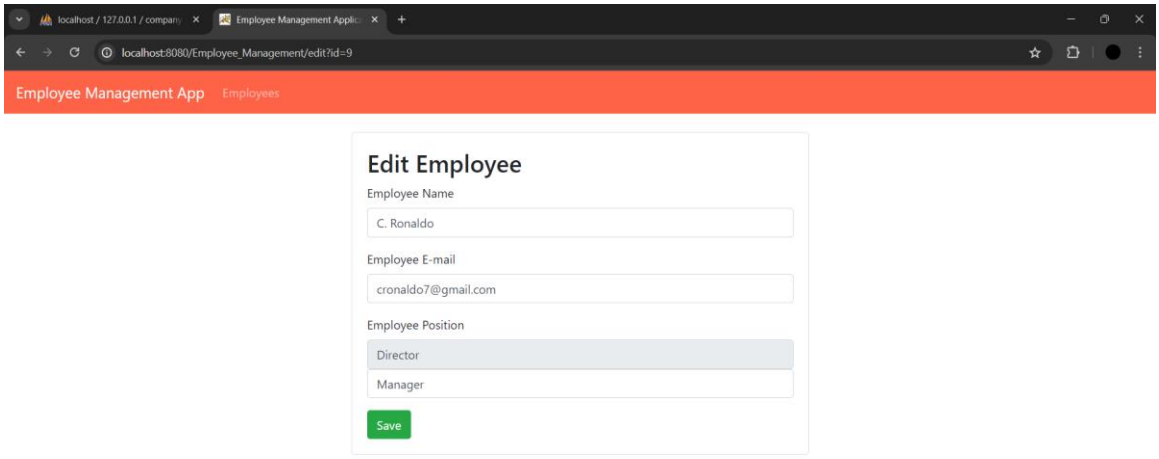


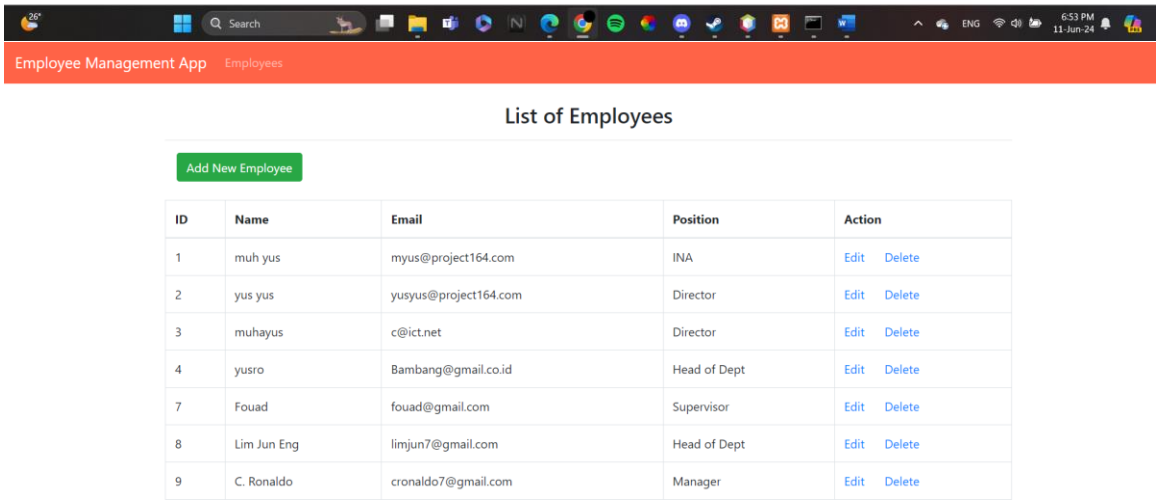
View



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

Update





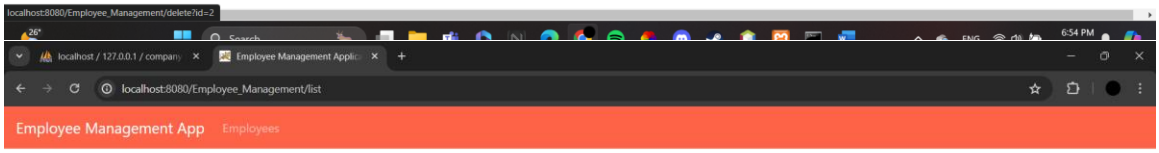
Name, email and position of employee number 9 had been changed

Delete (Employee no.2)

List of Employees

Add New Employee

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



List of Employees

Add New Employee

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

Exercise
com.Model
Car.java

```java
public class Car {
    private int carId;
    private String brand;
    private String model;
    private int cyclinder;
    private double price;

    public Car(){
    }

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

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

```java
    public int getCarId() {
        return carId;
    }

    public void setCarId(int carId) {
        this.carId = carId;
    }

    public String getBrand() {
        return brand;
    }

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

    public String getModel() {
        return model;
    }

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

```java
    public int getCyclinder() {
        return cyclinder;
    }

    public void setCyclinder(int cyclinder) {
        this.cyclinder = cyclinder;
    }

    public double getPrice() {
        return price;
    }

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


}
```

com.DAO
CarDAO.java

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

    private static final String INSERT_CARS_SQL = "INSERT INTO carpricelist (brand, model, cyclinder, price) VALUES (?, ?, ?, ?)";
    private static final String SELECT_CARS_BY_ID = "SELECT carid, brand, model, cyclinder, price from carpricelist where carid=?";
    private static final String SELECT_ALL_CARS = "SELECT * FROM carpricelist";
    private static final String DELETE_CARS_SQL = "DELETE FROM carpricelist where carid = ?";
    private static final String UPDATE_CARS_SQL = "UPDATE carpricelist set brand = ?, model = ?, cyclinder = ?, price = ? where carid = ?";

    public CarDAO() {}

    protected Connection getConnection(){
        Connection connection = null;
        try{
            Class.forName( className: "com.mysql.jdbc.Driver");
            connection = DriverManager.getConnection( url: jdbcURL , user: jdbcUsername ,  password: jdbcPassword);
        } catch (ClassNotFoundException e){
            e.printStackTrace();
        } catch (SQLException e){
            e.printStackTrace();
        }
        return connection;
    }
```

```java
    public void insertCar(Car car) throws SQLException{
        System.out.println( x: INSERT_CARS_SQL);
        try (Connection connection = getConnection();
            PreparedStatement preparedStatement = connection.prepareStatement( sql: INSERT_CARS_SQL);){

            preparedStatement.setString( parameterIndex: 1,  x: car.getBrand());
            preparedStatement.setString( parameterIndex: 2,  x: car.getModel());
            preparedStatement.setInt( parameterIndex: 3,  x: car.getCyclinder());
            preparedStatement.setDouble( parameterIndex: 4,  x: car.getPrice());
            System.out.println( x: preparedStatement);
            preparedStatement.executeUpdate();
        } catch (SQLException e) {
            printSQLException( ex: e);
        }
    }

    public Car selectCar(int carId) {
        Car car = null;
        try (Connection connection = getConnection();
            PreparedStatement preparedStatement = connection.prepareStatement( sql: SELECT_CARS_BY_ID);){
            preparedStatement.setInt( parameterIndex: 1,  x: carId);
            System.out.println( x: preparedStatement);
            ResultSet rs = preparedStatement.executeQuery();
            while (rs.next()) {
                String brand = rs.getString( columnLabel: "brand");
                String model = rs.getString( columnLabel: "model");
                int cyclinder = rs.getInt( columnLabel: "cyclinder");
                double price = rs.getDouble( columnLabel: "price");
                car = new Car(carId, brand, model, cyclinder, price);
            }
        } catch (SQLException e) {
            printSQLException( ex: e);
        }
        return car;
    }
```

```java
    public List<Car> selectAllCars() {
        List<Car> cars = new ArrayList<>();
        try (Connection connection = getConnection();
            PreparedStatement preparedStatement = connection.prepareStatement( sql: SELECT_ALL_CARS);){
            System.out.println( x: preparedStatement);
            ResultSet rs = preparedStatement.executeQuery();
            while (rs.next()) {
                int carId = rs.getInt( columnLabel: "carid");
                String brand = rs.getString( columnLabel: "brand");
                String model = rs.getString( columnLabel: "model");
                int cyclinder = rs.getInt( columnLabel: "cyclinder");
                double price = rs.getDouble( columnLabel: "price");
                cars.add(new Car(carId, brand, model, cyclinder, price));
            }
        } catch (SQLException e) {
            printSQLException( ex: e);
        }
        return cars;
    }

    public boolean deleteCar(int carId) throws SQLException{
        boolean rowDeleted;
        try (Connection connection = getConnection();
            PreparedStatement preparedStatement = connection.prepareStatement( sql: DELETE_CARS_SQL);){
            preparedStatement.setInt( parameterIndex: 1,  x: carId);
            rowDeleted = preparedStatement.executeUpdate() > 0;
        }
        return rowDeleted;
    }
```

```java
    public boolean updateCar(Car car) throws SQLException{
        boolean rowUpdated;
        try (Connection connection = getConnection();
            PreparedStatement preparedStatement = connection.prepareStatement( sql: UPDATE_CARS_SQL);){
            preparedStatement.setString( parameterIndex: 1,  x: car.getBrand());
            preparedStatement.setString( parameterIndex: 2,  x: car.getModel());
            preparedStatement.setInt( parameterIndex: 3,  x: car.getCyclinder());
            preparedStatement.setDouble( parameterIndex: 4,  x: car.getPrice());
            preparedStatement.setInt( parameterIndex: 5,  x: car.getCarId());
            rowUpdated = preparedStatement.executeUpdate() > 0;
        }
        return rowUpdated;
    }

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

com.WEB
CarServlet.java

```java
@WebServlet("/")
public class CarServlet extends HttpServlet {

    private CarDAO carDAO;

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

    @Override
    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
        doGet(request, response);
    }

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

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

    private void listCar(HttpServletRequest request, HttpServletResponse response)throws SQLException, ServletException, IOException{
        List<Car> listCar = carDAO.selectAllCars();
        request.setAttribute( name:"listCar",  o:listCar);
        RequestDispatcher rd = request.getRequestDispatcher( path:"carList.jsp");
        rd.forward(request, response);
    }

    private void showNewForm(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException{
        RequestDispatcher rd = request.getRequestDispatcher( path:"carForm.jsp");
        rd.forward(request, response);
    }

    private void showEditForm(HttpServletRequest request, HttpServletResponse response) throws SQLException, ServletException, IOException{
        int carId = Integer.parseInt( s:request.getParameter( name:"carId"));
        Car existingCar = carDAO.selectCar(carId);
        RequestDispatcher rd = request.getRequestDispatcher( path:"carForm.jsp");
        request.setAttribute( name:"car",  o:existingCar);
        rd.forward(request, response);
    }

    private void insertCar(HttpServletRequest request, HttpServletResponse response) throws SQLException, IOException{
        String brand = request.getParameter( name:"brand");
        String model = request.getParameter( name:"model");
        int cyclinder = Integer.parseInt( s:request.getParameter( name:"cyclinder"));
        double price = Double.parseDouble( s:request.getParameter( name:"price"));
        Car newCar = new Car (brand, model, cyclinder, price);
        carDAO.insertCar( car:newCar);
        response.sendRedirect( location:"list");
    }
```

```
    private void updateCar(HttpServletRequest request, HttpServletResponse response)throws SQLException, IOException{
        int carId = Integer.parseInt( s:request.getParameter( name:"carId"));
        String brand = request.getParameter( name:"brand");
        String model = request.getParameter( name:"model");
        int cyclinder = Integer.parseInt( s:request.getParameter( name:"cyclinder"));
        double price = Double.parseDouble( s:request.getParameter( name:"price"));

        Car car = new Car(carId, brand, model, cyclinder, price);
        carDAO.updateCar(car);
        response.sendRedirect( location:"list");
    }

    private void deleteCar(HttpServletRequest request, HttpServletResponse response)throws SQLException, IOException{
        int carId = Integer.parseInt( s:request.getParameter( name:"carId"));
        carDAO.deleteCar(carId);
        response.sendRedirect( location:"list");
    }
}
```

CarForm.jsp

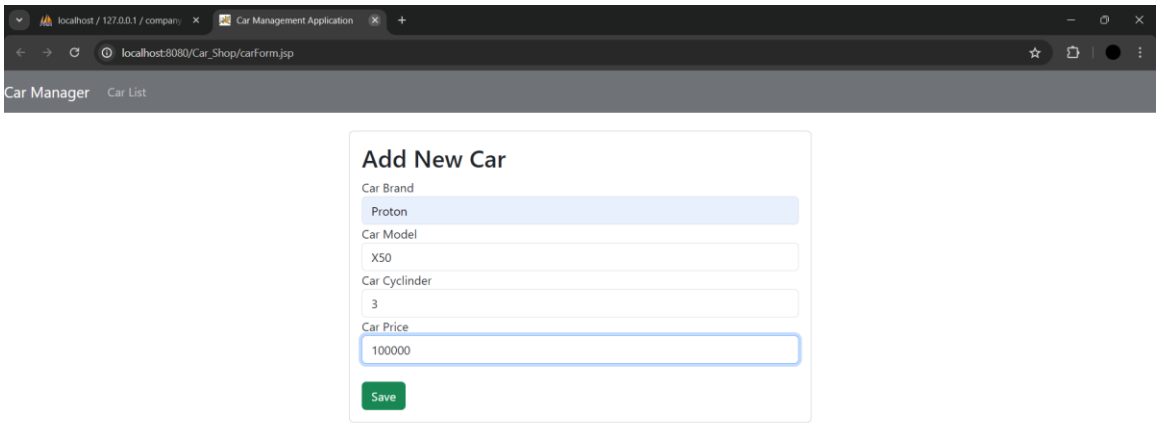
```
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
        <title>Car Management Application</title>
        <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css">
        <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/bootstrap.bundle.min.js"></script>
    </head>

    <body>
        <header>
            <nav class="navbar navbar-expand-md navbar-dark" style="background-color: #6P7378">
                <div>
                    <a href="carForm.jsp" class="navbar-brand">Car Manager</a>
                </div>
                <ul class="navbar-nav">
                    <li><a href="<%=request.getContextPath()%>/list" class="nav-link">Car List</a></li>
                </ul>
            </nav>
        </header>
        <br>
        <div class="container col-md-5">
            <div class="card">
                <div class="card-body">
                    <c:set var="formAction" value="${car != null ? 'update' : 'insert'}" />
                    <form action="${formAction}" method="post">

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

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

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

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

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

                    <fieldset class="form-group">
                        <label>Car Price</label>
                        <input type="text" value="<c:out value='${car.price}'/>" class="form-control" name="price" required="required"/>
                    </fieldset>
                    <br>
                    <button type="submit" class="btn btn-success">Save</button>
                </form>
            </div>
        </div>
    </div>
</body>
</html>
```

CarList.jsp

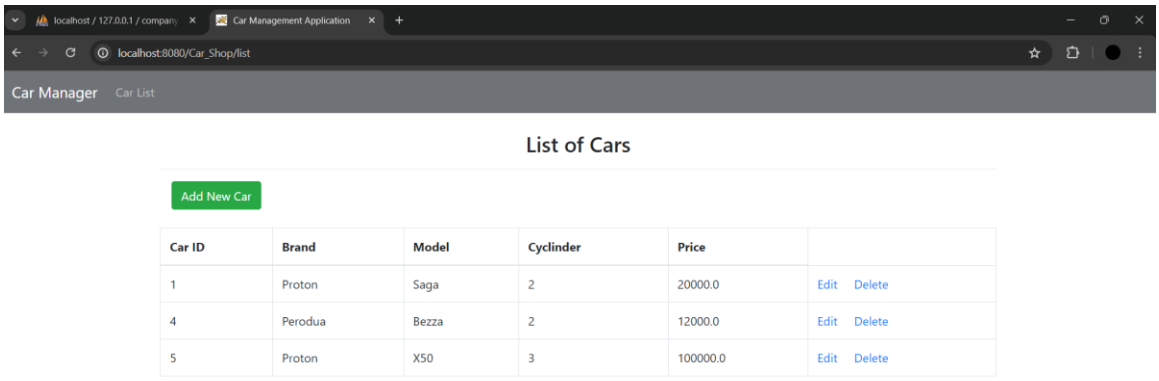
```jsp
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<!DOCTYPE html>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
        <title>Car Management Application</title>
        <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.min.css"
              integrity="sha384-MCw98/SFnGE8fJT3GXwEOngsV7Zt27NXFoaoApmYm81iuXoPkFOJwJ8ERdknLPMO" crossorigin="anonymous">
    </head>
    <body>
        <header>
            <nav class="navbar navbar-expand md navbar-dark" style="background-color: #6F7378">
                <div>
                    <a href="carForm.jsp" class="navbar-brand">Car Manager</a>
                </div>

                <ul class="navbar-nav">
                    <li><a href="<%request.getContextPath();%>/list" class="nav-link">Car List</a></li>
                </ul>
            </nav>
        </header>
        <br>
        <div class="row">
            <div class="container">
                <h3 class="text-center">List of Cars</h3>
                <hr>
                <div class="container text-left">
                    <a href="<%=request.getContextPath()%>/new" class="btn btn-success">Add New Car</a>
                </div>
                <br>
                <table class="table table-bordered">
                    <thead>
                        <tr>
                            <th>Car ID</th>
                            <th>Brand</th>
                            <th>Model</th>
                            <th>Cyclinder</th>
                            <th>Price</th>
                        </tr>
                    </thead>
                <tbody>
                    <c:forEach var="car" items="${listCar}">
                        <tr>
                            <td>
                                <c:out value="${car.carId}"/>
                            </td>

                            <td>
                                <c:out value="${car.brand}"/>
                            </td>

                            <td>
                                <c:out value="${car.model}"/>
                            </td>

                            <td>
                                <c:out value="${car.cyclinder}"/>
                            </td>

                            <td>
                                <c:out value="${car.price}"/>
                            </td>

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

Output

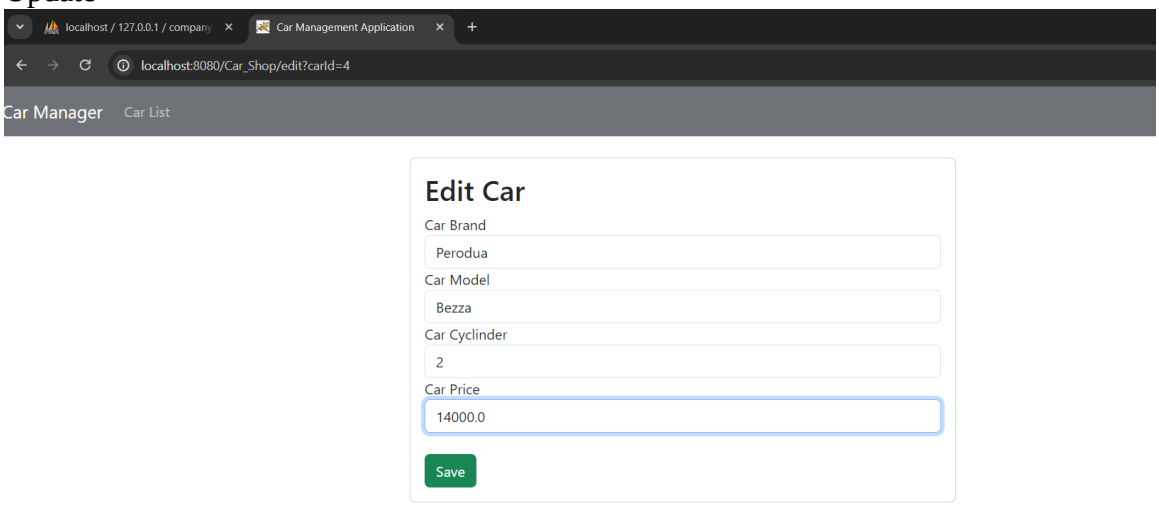
Create

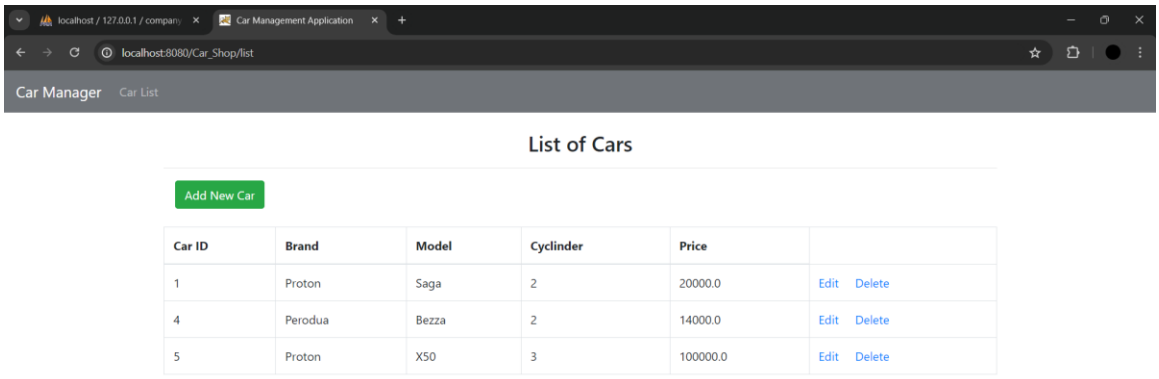


View

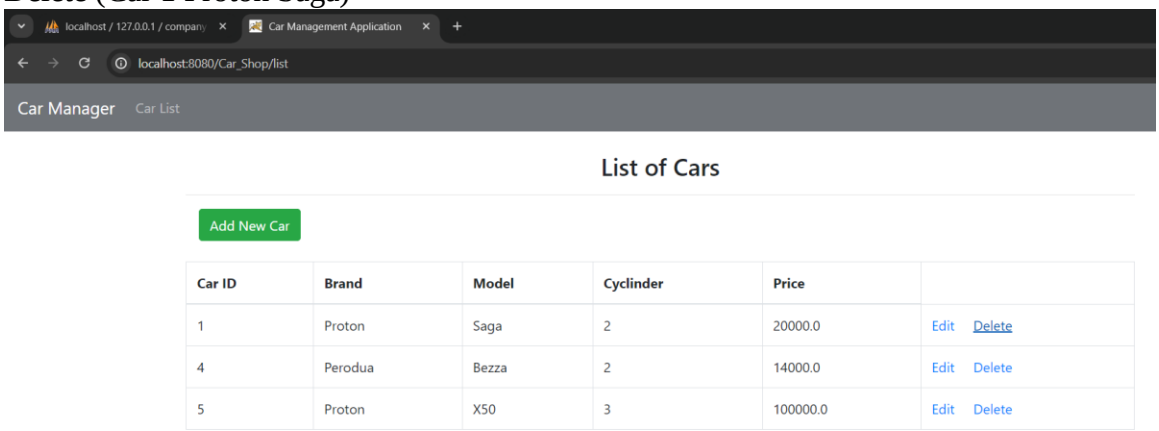


Update





Delete (Car 1 Proton Saga)



List of Cars

Add New Car

Car ID	Brand	Model	Cyclinder	Price	
4	Perodua	Bezza	2	14000.0	Edit Delete
5	Proton	X50	3	100000.0	Edit Delete