✅ Lab Sheet: Java Web Application Using MVC with Database

Module: Development of Enterprise Applications (DEA)
Topic: Developing a Java Web Application Using MVC Pattern with MySQL Database
Duration: 2 Hours

🎯 Learning Objectives

By the end of this lab, students will be able to:

- Implement a simple web application using the MVC design pattern.
- Connect and interact with a MySQL database using JDBC.
- Perform basic CRUD operations from a JSP/Servlet-based interface.
- Understand separation of concerns in MVC architecture.

📘 Scenario: Student Management System

Develop a web application that allows users to register students into a MySQL database.
The user should be able to:

- Add new students
- View a list of registered students

🛠️ Tasks

1. Set up MySQL Database

Create a database named studentdb and a table students:

sql

```sql
CREATE DATABASE studentdb;


CREATE TABLE students (
  id INT AUTO_INCREMENT PRIMARY KEY,
  name VARCHAR(100),
  email VARCHAR(100),
  course VARCHAR(100)
);
```

---

2. Project Structure (Following MVC)

markdown

CopyEdit

```
WebContent/
  ├── studentForm.jsp
  ├── studentList.jsp

src/
 └── com.nsbm.model/
   └── Student.java

 └── com.nsbm.dao/
   └── StudentDAO.java

 └── com.nsbm.controller/
   └── StudentServlet.java
```

---

3. Implementation Details

🔹 **Student.java – JavaBean (Model)**

java

```
public class Student {
  private int id;
  private String name;
  private String email;
  private String course;

  // Getters and setters
}
```

◆ **StudentDAO.java – Database Access (Model)**

Handles DB connection and operations like insert, list:

java

```
public class StudentDAO {
  private String jdbcURL = "jdbc:mysql://localhost:3306/studentdb";
  private String jdbcUsername = "root";
  private String jdbcPassword = "";

  public void insertStudent(Student student) {
    // JDBC code to insert student into DB
  }

  public List<Student> listStudents() {
    // JDBC code to fetch all students
  }
}
```

◆ **StudentServlet.java – Controller**

Handles form submission and data forwarding.

java

```
@WebServlet("/StudentServlet")
public class StudentServlet extends HttpServlet {
  private StudentDAO studentDAO;

  public void init() {
    studentDAO = new StudentDAO();
  }
```

```
    protected void doPost(HttpServletRequest request, HttpServletResponse response) {
        // Handle insert
    }

    protected void doGet(HttpServletRequest request, HttpServletResponse response) {
        // Handle list
    }
}
```

🔹 **studentForm.jsp – View (Input Form)**

jsp

```
<form action="StudentServlet" method="post">
    Name: <input type="text" name="name"/><br/>
    Email: <input type="text" name="email"/><br/>
    Course: <input type="text" name="course"/><br/>
    <input type="submit" value="Register"/>
</form>
```

🔹 **studentList.jsp – View (Display List)**

jsp

```
<c:forEach var="student" items="${studentList}">
    <p>${student.name} - ${student.email} - ${student.course}</p>
</c:forEach>
```

📁 **Deliverables**

- Functional MVC-based web application
- Screenshots of:
    - Student registration form
    - Student list page

- Source code of:
    - JSPs
    - Servlet
    - DAO
    - JavaBean

🧠 Extension Tasks (Optional)

- Add update and delete operations
- Add input validation and exception handling
- Use JSTL for better presentation

✅ Evaluation Criteria

Criteria	**Marks**
Database Integration	15
MVC Architecture Usage	15
Form and Display Page	10
DAO and JDBC Implementation	10
Code Structure and Cleanliness	10
Total	**60**