**Web Application Development – Lab 4: JSP + MYSQL - CRUD OPERATIONS**

**Name: Dang Thanh Lam – ITITDK23039**

## 1. Executive Summary

The initiative focused on deploying a JSP-based Student Management System leveraging Apache Tomcat/TomEE as the execution environment and MySQL as the persistence layer. Core deliverables included end-to-end CRUD functionality, JDBC integration, server-side validation, error handling, and successful deployment using NetBeans. The outcome met baseline functional requirements with a stable runtime footprint and consistent transactional behaviors.

## 2. Project Scope

The scope covered:

- Establishing a functional MySQL schema for student records.
- Implementing JSP pages for Create, Read, Update, and Delete workflows.
- Enabling JDBC connectivity using MySQL Connector/J.
- Deploying the application to Tomcat/TomEE through NetBeans.
- Validating operational performance across all CRUD paths.

## 3. System Architecture

The solution followed a classic three-tier structure:

### Presentation Layer

- JSP pages:
  list_students.jsp, add_student.jsp, process_add.jsp,
  edit_student.jsp, process_edit.jsp, delete_student.jsp

### Application Logic

- JSP-embedded JDBC routines
- PreparedStatement-driven SQL execution
- Form validation (client-side and server-side)

**Data Layer**

- MySQL database: student_management
- Table: students (id, student_code, full_name, email, major, created_at)

**4. Implementation Overview**

**4.1 List Functionality (SELECT)**

- Retrieval executed via SELECT * FROM students ORDER BY id DESC.
- ResultSet iterated to populate dynamic HTML tables.
- NULL-safe handling implemented for optional attributes.
- Action links routed to edit and delete endpoints.

**4.2 Create Functionality (INSERT)**

- Data entry handled through add_student.jsp.
- POST submission routed to process_add.jsp.
- Validations enforced for mandatory attributes.
- SQL executed through PreparedStatement to mitigate injection vectors.

**4.3 Update Functionality (UPDATE)**

- Existing student details pre-fetched and pre-populated in edit_student.jsp.
- Readonly constraints applied to immutable fields (student_code).
- Update executed through parameterized SQL with WHERE clause targeting id.

**4.4 Delete Functionality (DELETE)**

- Deletion logic implemented in delete_student.jsp.
- Server returns contextual messages based on affected rows.
- Foreign key constraint scenarios handled and redirected with structured errors.

**5. Database Schema Design**

**Table: students**

| Column | Type | Attributes |
|---|---|---|
| id | INT | PK, Auto-Increment |
| student_code | VARCHAR | Unique, Not Null |
| full_name | VARCHAR | Not Null |
| email | VARCHAR | Nullable |
| major | VARCHAR | Nullable |
| created_at | TIMESTAMP | Default CURRENT_TIMESTAMP |

Schema supports straightforward extensions and maintains atomic consistency across CRUD operations.

**6. Deployment Process**

- Environment: NetBeans 12 + Apache Tomcat/TomEE
- Workflow:
    1. Clean and Build
    2. Deploy
    3. Validate context path mapping (/StudentManagement)
- Addressed deployment constraints involving Windows service permissions and temporary directory access.

**7. Testing and Validation**

Executed functional validation across:

- Form submission flows
- Input validation (client/server side)
- Database consistency checks
- Error redirection and messaging
- CRUD lifecycle integrity

Outcomes confirmed that the solution operates within expected parameters.

**8. Issues Encountered and Resolutions**

**Issue: Deployment failure due to restricted Windows service permissions.**

**Resolution:** Adjusted TomEE/Tomcat account privileges or executed under LocalSystem.

**Issue: JDBC driver not found.**

**Resolution:** Added MySQL Connector/J manually into project libraries.

**Issue: SQL errors during CRUD workflow.**

**Resolution:** Standardized PreparedStatement usage and validated schema alignment.

**9. Conclusion**

The Student Management System was successfully developed, integrated, and deployed. The final solution delivers reliable end-user functionality, aligns with JDBC best practices, and demonstrates operational compliance with core CRUD requirements. The system is stable, extensible, and suitable for academic demonstration and practical application.