Lab 5 Homework Report

Student: Lê Hoài Bảo
ID: ITCSIU22259

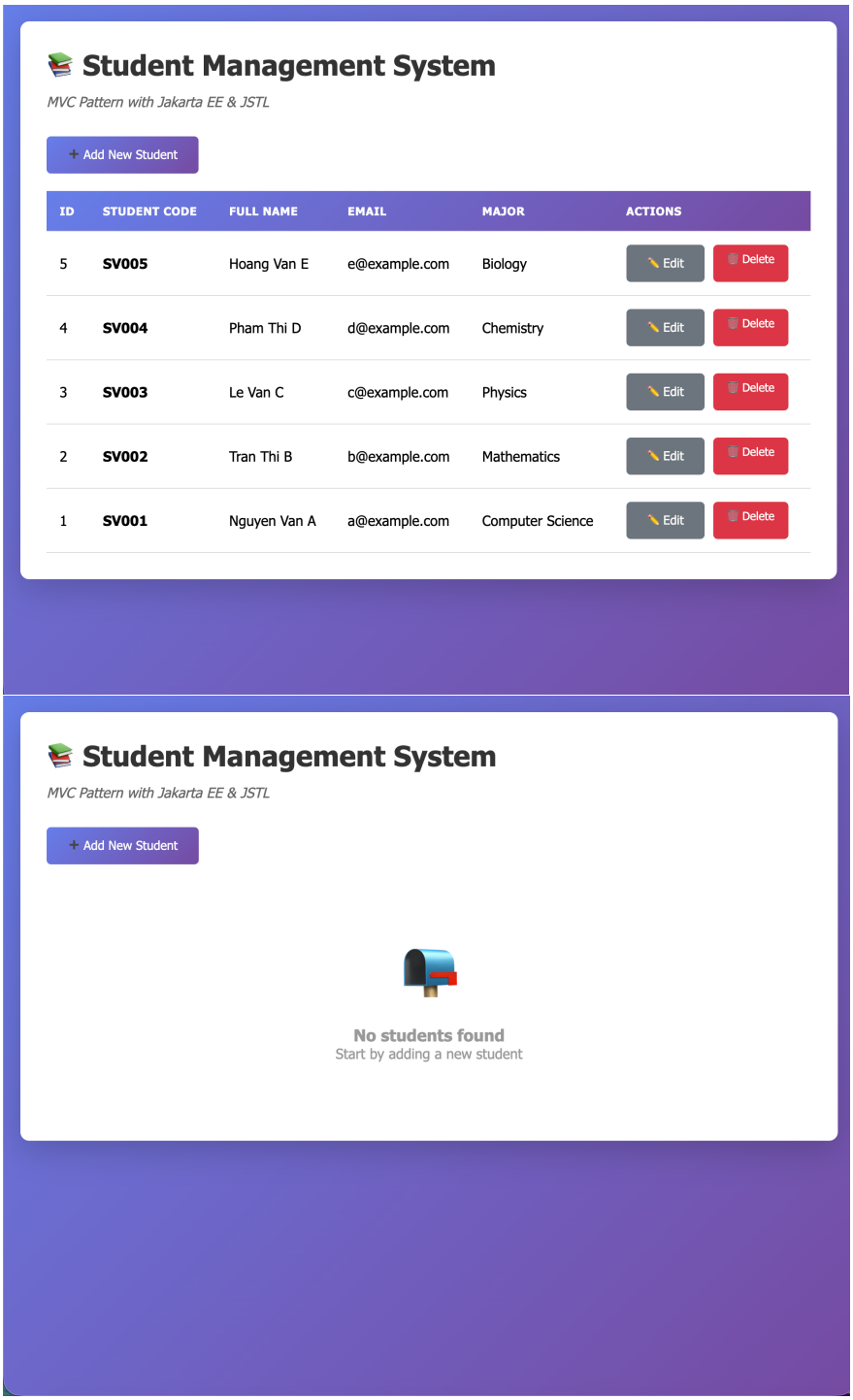
5. Seach student
5.1 StudentDAO.java

```java
public List<Student> searchStudents(String keyword) {
    List<Student> students = new ArrayList<>();

    // Handle null keyword to prevent errors (treat as empty string)
    if (keyword == null) {
        keyword = "";
    }

    // SQL Query: Search across 3 columns using OR logic
    String sql = "SELECT * FROM students WHERE student_code LIKE ? OR full_name LIKE ? OR email LIKE ? ORDER BY id D

    try (Connection conn = getConnection();
         PreparedStatement pstmt = conn.prepareStatement(sql)) {

        // Prepare the wildcard pattern
        String searchPattern = "%" + keyword + "%";

        // Fill all three placeholders (?) with the same pattern
        pstmt.setString(1, searchPattern); // Checks student_code
        pstmt.setString(2, searchPattern); // Checks full_name
        pstmt.setString(3, searchPattern); // Checks email

        try (ResultSet rs = pstmt.executeQuery()) {
            while (rs.next()) {
                Student student = new Student();
                student.setId(rs.getInt("id"));
                student.setStudentCode(rs.getString("student_code"));
                student.setFullName(rs.getString("full_name"));
                student.setEmail(rs.getString("email"));
                student.setMajor(rs.getString("major"));
                student.setCreatedAt(rs.getTimestamp("created_at"));
                students.add(student);
            }
        }
```

5.2 StudentController.java

```java
@Override
protected void doGet(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {

    String action = request.getParameter("action");

    if (action == null) {
        action = "list";
    }

    switch (action) {
        case "new":
            showNewForm(request, response);
            break;
        case "edit":
            showEditForm(request, response);
            break;
        case "delete":
            deleteStudent(request, response);
            break;
        case "search": // <--- NEW CASE ADDED HERE
            searchStudents(request, response);
            break;
        default:
            listStudents(request, response);
            break;
    }
}

// Search students
private void searchStudents(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {

    // 1. Get keyword parameter from the form
    String keyword = request.getParameter("keyword");

    List<Student> students;

    // 2. Decide which DAO method to call
    // If keyword is null or empty, just show all students
    if (keyword != null && !keyword.trim().isEmpty()) {
        students = studentDAO.searchStudents(keyword);
    } else {
        students = studentDAO.getAllStudents();
    }

    // 3. Set request attributes
    // We send the list of students found
    request.setAttribute("students", students);

    // We ALSO send the keyword back so we can keep it in the search box (sticky form)
    request.setAttribute("keyword", keyword);

    // 4. Forward to view
    RequestDispatcher dispatcher = request.getRequestDispatcher("student-list.jsp");
    dispatcher.forward(request, response);
}
```

5.3 update form in student-list.jsp



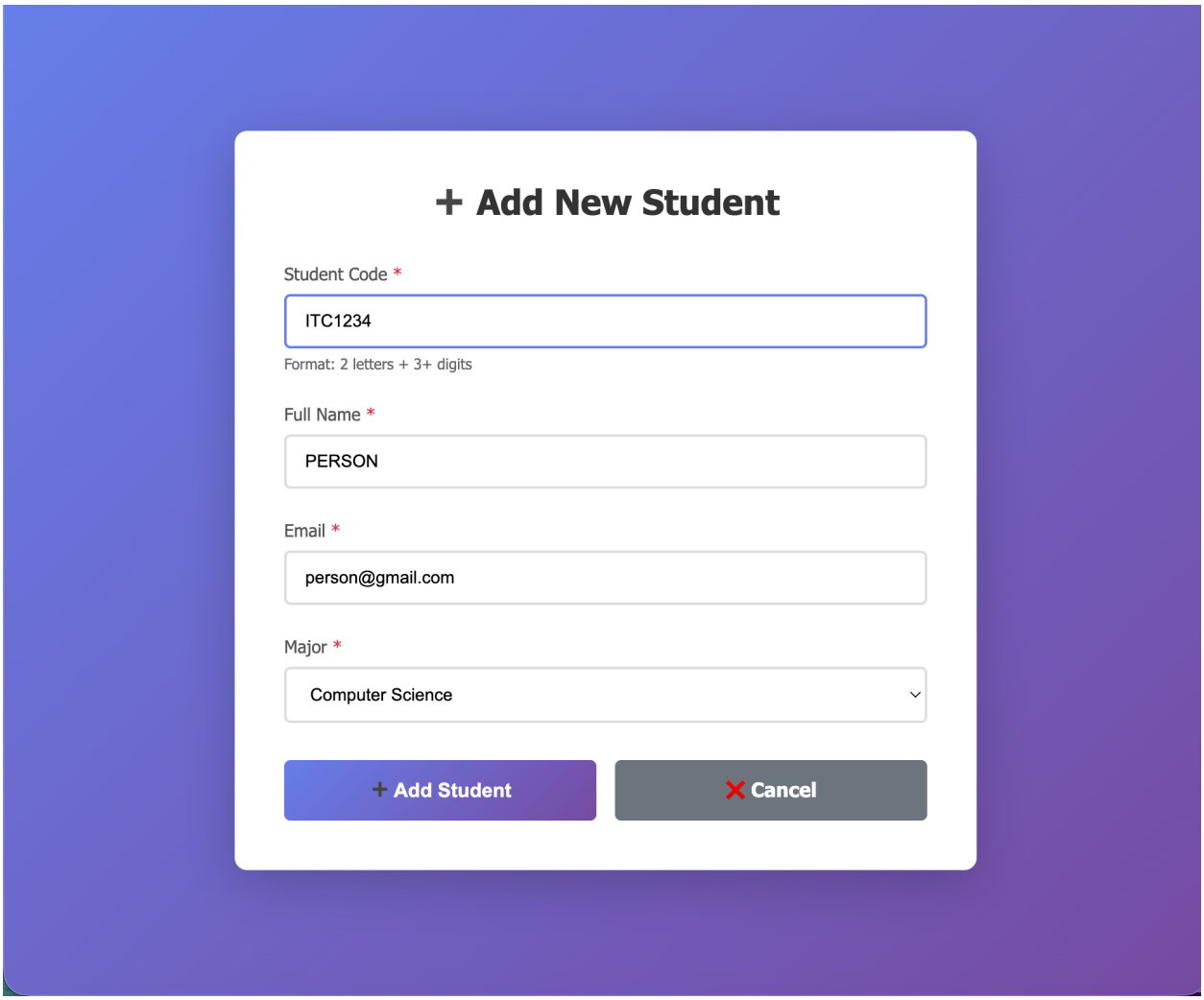
6 Validation

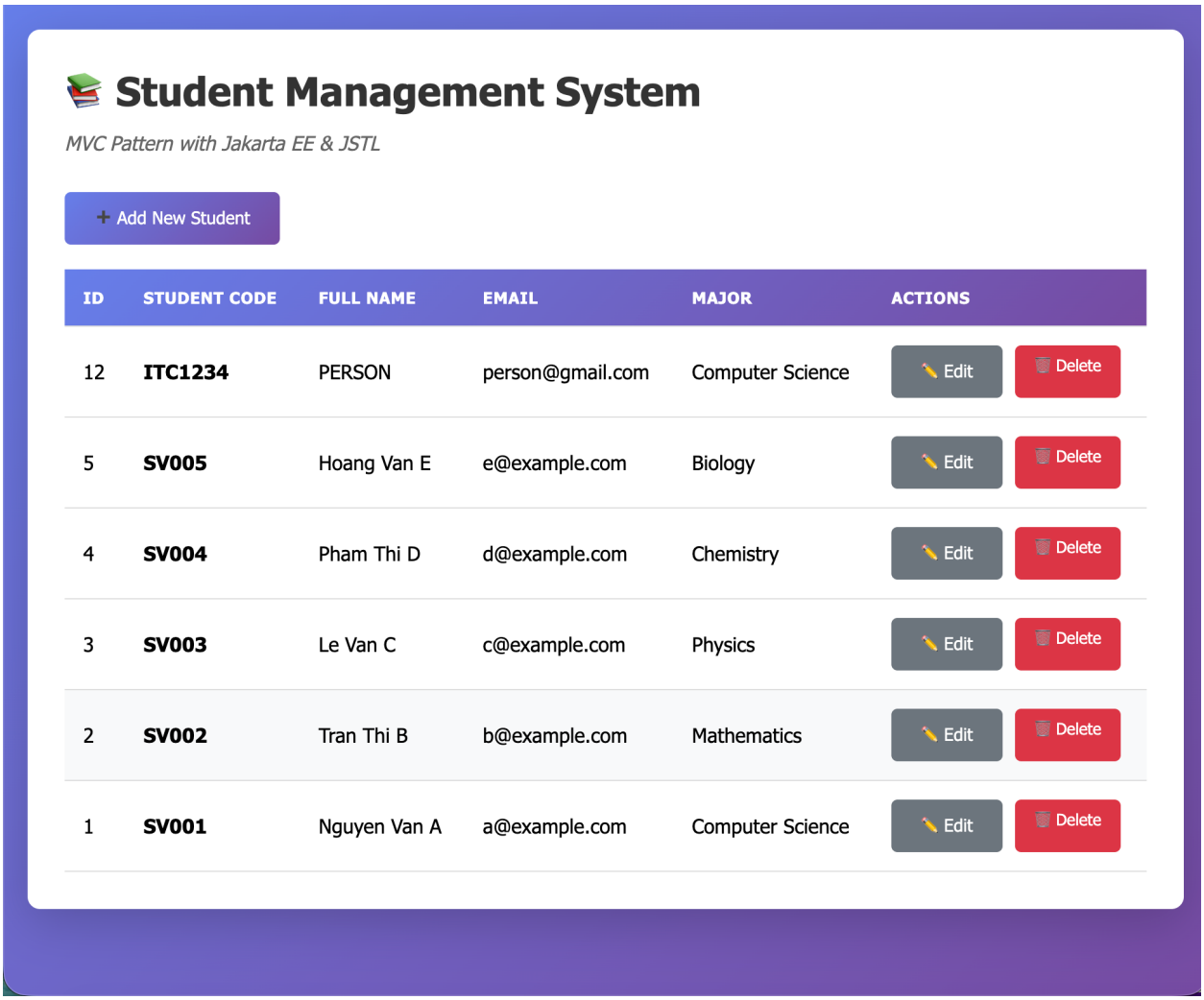
6.1 ValidateStudent

```java
private boolean validateStudent(Student student, HttpServletRequest request) {
    boolean isValid = true;

    // 1. Validate Student Code
    String code = student.getStudentCode();
    String codePattern = "[A-Z]{2}[0-9]{3,}"; // e.g., SV001, IT123

    if (code == null || code.trim().isEmpty()) {
        request.setAttribute("errorCode", "Student code is required");
        isValid = false;
    } else if (!code.matches(codePattern)) {
        request.setAttribute("errorCode", "Invalid format. Use 2 uppercase letters + 3+ digits (e.g., SV001)");
        isValid = false;
    }

    // 2. Validate Full Name
    String name = student.getFullName();
    if (name == null || name.trim().isEmpty()) {
        request.setAttribute("errorName", "Full name is required");
        isValid = false;
    } else if (name.trim().length() < 2) {
        request.setAttribute("errorName", "Name must be at least 2 characters");
        isValid = false;
    }

    // 3. Validate Email (Optional but must be valid format if provided)
    String email = student.getEmail();
    String emailPattern = "^[A-Za-z0-9+_.-]+@(.+)$";

    if (email != null && !email.trim().isEmpty()) {
        if (!email.matches(emailPattern)) {
            request.setAttribute("errorEmail", "Invalid email format");
```

6.2 student-form.jsp

6.3 css


+ Add New Student
Student Code *
ITC1234
Format: 2 letters + 3+ digits
Full Name *
PERSON
Email *
person@gmail.com
Major *
Computer Science
+ Add Student
Cancel


Student Management System
MVC Pattern with Jakarta EE & JSTL
+ Add New Student
ID
STUDENT CODE
FULL NAME
EMAIL
MAJOR
ACTIONS
12
ITC1234
PERSON
person@gmail.com
Computer Science
Edit
Delete
5
SV005
Hoang Van E
e@example.com
Biology
Edit
Delete
4
SV004
Pham Thi D
d@example.com
Chemistry
Edit
Delete
3
SV003
Le Van C
c@example.com
Physics
Edit
Delete
2
SV002
Tran Thi B
b@example.com
Mathematics
Edit
Delete
1
SV001
Nguyen Van A
a@example.com
Computer Science
Edit
Delete