

Exercise 1:

Student Management System

[+ Add New Student](#)

ID	Student Code	Full Name	Email	Major	Created At	Actions
5	SV005	David Wilson	david.w@email.com	Computer Science	2025-11-08 10:31:24.0	Edit Delete
4	SV004	Sarah Davis	sarah.d@email.com	Data Science	2025-11-08 10:31:24.0	Edit Delete
3	SV003	Michael Brown	michael.b@email.com	Software Engineering	2025-11-08 10:31:24.0	Edit Delete
2	SV002	Emily Johnson	emily.j@email.com	Information Technology	2025-11-08 10:31:24.0	Edit Delete
1	SV001	John Smith	john.smith@email.com	Computer Science	2025-11-08 10:31:24.0	Edit Delete

```
<% if (request.getParameter("message") != null) { %>
    <div class="message success">
        <%= request.getParameter("message") %>
    </div>
<% } %>

<% if (request.getParameter("error") != null) { %>
    <div class="message error">
        <%= request.getParameter("error") %>
    </div>
<% } %>
```

If the message is empty and the error isn't empty then the user will get an error notification

```

try {
    Class.forName("com.mysql.cj.jdbc.Driver");

    conn = DriverManager.getConnection(
        "jdbc:mysql://localhost:3306/student_management",
        "root",
        "root"
    );

    stmt = conn.createStatement();
    String sql = "SELECT * FROM students ORDER BY id DESC";
    rs = stmt.executeQuery(sql);

    while (rs.next()) {
        int id = rs.getInt("id");
        String studentCode = rs.getString("student_code");
        String fullName = rs.getString("full_name");
        String email = rs.getString("email");
        String major = rs.getString("major");
        Timestamp createdAt = rs.getTimestamp("created_at");
    }
}

```

This line of code attempts to access the student_management sql file with the username of root and password of root, then assigns the corresponding values to a string or timestamp.

```

<%
    }
} catch (ClassNotFoundException e) {
    out.println("<tr><td colspan='7'>Error: JDBC Driver not found!</td></tr>");
    e.printStackTrace();
} catch (SQLException e) {
    out.println("<tr><td colspan='7'>Database Error: " + e.getMessage() + "</td></tr>");
    e.printStackTrace();
} finally {
    try {
        if (rs != null) rs.close();
        if (stmt != null) stmt.close();
        if (conn != null) conn.close();
    } catch (SQLException e) {
        e.printStackTrace();
    }
}
%>

```

catch (ClassNotFoundException) Handles missing JDBC driver errors

catch (SQLException) Handles database-related errors

finally

Closes database resources safely
(ResultSet, Statement, Connection)

Exercise 2:

On pressing Add New student:

+ Add New Student

Student Code *
SV005

Full Name *
d

Email
d@gmail.com

Major
Computer Science

Save Student **Cancel**

On pressing save student:

Student Management System

Student added successfully

+ Add New Student

ID	Student Code	Full Name	Email	Major
6	SV005	d	d@gmail.com	Com
4	SV004	Sarah Davisdd	sarah.d@email.com	Data
3	SV003	Michael Brown	michael.b@email.com	Soft
2	SV002	Emily Johnson	emily.j@email.com	Info
1	SV001	John Smith	john.smith@email.com	Com

A new student was added

Pressing the cancel button would just return the user to the main page again

```

<form action="process_add.jsp" method="POST">
    <div class="form-group">
        <label for="student_code">Student Code</label>
        <input type="text" id="student_code" name="student_code" placeholder="e.g., SV001" required="required" pattern="[A-Z]{2}[0-9]{3,}" title="Format: 2 uppercase letters followed by 3 digits" />
    </div>

    <div class="form-group">
        <label for="full_name">Full Name</label>
        <input type="text" id="full_name" name="full_name" placeholder="Enter full name" required="required" />
    </div>

    <div class="form-group">
        <label for="email">Email</label>
        <input type="email" id="email" name="email" placeholder="student@email.com" required="required" />
    </div>

    <div class="form-group">
        <label for="major">Major</label>
        <input type="text" id="major" name="major" placeholder="e.g., Computer Science" required="required" />
    </div>

    <button type="submit" class="btn-submit">Add Student</button>

```

Add_student.jsp just has a form to send input information to process_add.jsp

```

String studentCode = request.getParameter("student_code");
String fullName = request.getParameter("full_name");
String email = request.getParameter("email");
String major = request.getParameter("major");

if (studentCode == null || studentCode.trim().isEmpty() ||
    fullName == null || fullName.trim().isEmpty()) {
    response.sendRedirect("add_student.jsp?error=Required fields are empty");
    return;
}

Connection conn = null;
PreparedStatement pstmt = null;

try {
    Class.forName("com.mysql.cj.jdbc.Driver");
    conn = DriverManager.getConnection(
        "jdbc:mysql://localhost:3306/student_management",
        "root",
        "root"
    );

    String sql = "INSERT INTO students (student_code, full_name, email, major)";
    pstmt = conn.prepareStatement(sql);
    pstmt.setString(1, studentCode);
    pstmt.setString(2, fullName);
    pstmt.setString(3, email);
    pstmt.setString(4, major);
}

```

```
        if (rowsAffected > 0) {
            response.sendRedirect("list_students.jsp");
        } else {
            response.sendRedirect("add_student.jsp?error=1");
        }
    } catch (ClassNotFoundException e) {
        response.sendRedirect("add_student.jsp?error=2");
        e.printStackTrace();
    } catch (SQLException e) {
        String errorMsg = e.getMessage();
        if (errorMsg.contains("Duplicate entry")) {
            response.sendRedirect("add_student.jsp?error=3");
        } else {
            response.sendRedirect("add_student.jsp?error=4");
        }
        e.printStackTrace();
    } finally {
        try {
            if (stmt != null) stmt.close();
            if (conn != null) conn.close();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}
```

Process_add.jsp enters the input into the database, sending errors if any input is missing, a driver isn't found or if information is duplicate

Duplicate error:

Add New Student

Student code already exists

Student Code *

e.g., SV001

Full Name *

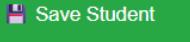
Enter full name

Email

student@email.com

Major

e.g., Computer Science

 Save Student  Cancel

Empty:

Student Code *

e.g., SV001

Full Name * ! Please fill out this field

Enter full name

Email

student@email.com

Major

e.g., Computer Science

 Save Student Cancel

Exercise 3:

 **Edit Student Information**

Student Code

SV005
Cannot be changed

Full Name *

d

Email

d@gmail.com

Major

Computer Science

 Update Cancel

After edit:

A screenshot of a web-based student management system. The page has a light green header bar with a blue button labeled '+ Add New Student'. Below the header is a table with a blue header row containing columns for ID, Student Code, Full Name, Email, and Major. The table contains five rows of student data. The last row, which corresponds to the 'Wrong id' test case, has an ID of 1 and a student code of SV001, but the full name is listed as 'John Smith'.

ID	Student Code	Full Name	Email	Major
6	SV005	d444444	d@gmail.com	Computer Science
4	SV004	Sarah Davisdd	sarah.d@email.com	Data Science
3	SV003	Michael Brown	michael.b@email.com	Software Engineering
2	SV002	Emily Johnson	emily.j@email.com	Information Systems
1	SV001	John Smith	john.smith@email.com	Computer Science

Wrong id:

A screenshot of a web-based student management system. The browser address bar shows 'localhost:8080/StudentManagement/list_students.jsp?error=Student%20not%20found'. The page title is 'Student Management System'. A pink error message box displays 'Student not found'. Below the error message is a blue header bar with a '+ Add New Student' button. A table with a blue header row follows, listing five student records. The first record has an ID of 6 and a student code of SV005, with the full name listed as 'Sarah Davisdd'.

ID	Student Code	Full Name	Email	Major
6	SV005	Sarah Davisdd	d@gmail.com	Computer Science
4	SV004	Sarah Davisdd	sarah.d@email.com	Data Science
3	SV003	Michael Brown	michael.b@email.com	Software Engineering
2	SV002	Emily Johnson	emily.j@email.com	Information Systems

Blank name:

Edit Student Information

Student Code

Cannot be changed

Full Name *

Email

Please fill out this field

Major

Update Cancel

Process_edit.jsp and edit_student.jsp works similarly to adding, one is a form and one changes information in the database

Exercise 4:

Student Management System

+ Add New Student

localhost:8080 says
Are you sure?

OK Cancel

ID	Student Code	Full Name	Email	Major	Created At	Actions
6	SV005	Sarah Davisdd	d@gmail.com	Computer Science	2025-11-08 11:02:36.0	Edit Delete
4	SV004	Sarah Davisdd	sarah.d@email.com	Data Science	2025-11-08 10:31:24.0	Edit Delete
3	SV003	Michael Brown	michael.b@email.com	Software Engineering	2025-11-08 10:31:24.0	Edit Delete
2	SV002	Emily Johnson	emily.j@email.com	Information Technology	2025-11-08 10:31:24.0	Edit Delete
1	SV001	John Smith	john.smith@email.com	Computer Science	2025-11-08 10:31:24.0	Edit Delete

Student Management System

Student deleted successfully

[+ Add New Student](#)

ID	Student Code	Full Name	Email	Major	Created At	Actions
4	SV004	Sarah Davisdd	sarah.d@email.com	Data Science	2025-11-08 10:31:24.0	Edit Delete
3	SV003	Michael Brown	michael.b@email.com	Software Engineering	2025-11-08 10:31:24.0	Edit Delete
2	SV002	Emily Johnson	emily.j@email.com	Information Technology	2025-11-08 10:31:24.0	Edit Delete
1	SV001	John Smith	john.smith@email.com	Computer Science	2025-11-08 10:31:24.0	Edit Delete

Cancel:

Student Management System

localhost:8080 says
Are you sure?

Student deleted successfully

[OK](#) [Cancel](#)

[+ Add New Student](#)

ID	Student Code	Full Name	Email	Major	Created At	Actions
4	SV004	Sarah Davisdd	sarah.d@email.com	Data Science	2025-11-08 10:31:24.0	Edit Delete
3	SV003	Michael Brown	michael.b@email.com	Software Engineering	2025-11-08 10:31:24.0	Edit Delete
2	SV002	Emily Johnson	emily.j@email.com	Information Technology	2025-11-08 10:31:24.0	Edit Delete
1	SV001	John Smith	john.smith@email.com	Computer Science	2025-11-08 10:31:24.0	Edit Delete

Student Management System

Student deleted successfully

[+ Add New Student](#)

ID	Student Code	Full Name	Email	Major	Created At	Actions
4	SV004	Sarah Davisdd	sarah.d@email.com	Data Science	2025-11-08 10:31:24.0	Edit Delete
3	SV003	Michael Brown	michael.b@email.com	Software Engineering	2025-11-08 10:31:24.0	Edit Delete
2	SV002	Emily Johnson	emily.j@email.com	Information Technology	2025-11-08 10:31:24.0	Edit Delete
1	SV001	John Smith	john.smith@email.com	Computer Science	2025-11-08 10:31:24.0	Edit Delete

The student remains

```
PreparedStatement pstmt = null;
```

```
try {
    Class.forName("com.mysql.cj.jdbc.Driver");
    conn = DriverManager.getConnection(
        "jdbc:mysql://localhost:3306/student_management",
        "root",
        "root"
    );

    String sql = "DELETE FROM students WHERE id = ?";
    pstmt = conn.prepareStatement(sql);
    pstmt.setInt(1, studentId);

    int rowsAffected = pstmt.executeUpdate();

    if (rowsAffected > 0) {
        response.sendRedirect("list_students.jsp?message=Student deleted successfully");
    } else {
        response.sendRedirect("list_students.jsp?error=Student not found");
    }
}

} catch (SQLException e) {
    if (e.getMessage().contains("foreign key constraint")) {
        response.sendRedirect("list_students.jsp?error=Cannot delete: has related records");
    } else {
        response.sendRedirect("list_students.jsp?error=Database error");
    }
}
```

The jsp file attempts to delete the correlating file from the sql database, report to the user if it encounters any errors while doing so such as if it has foreign key constraints