

NAME: LEWINS MUREITHI

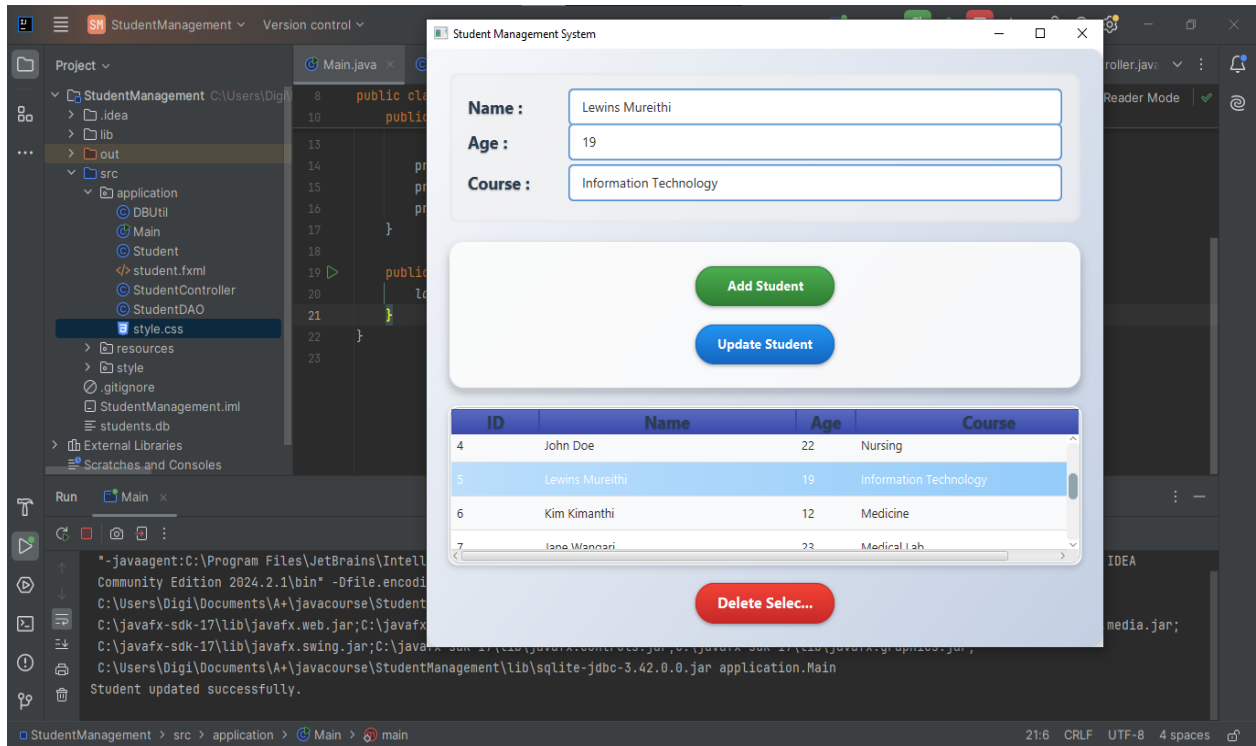
REG NUMBER: G127/1415/2023

COURSE: INFORMATION TECHNOLOGY

YEAR: 3 SEMESTER 1

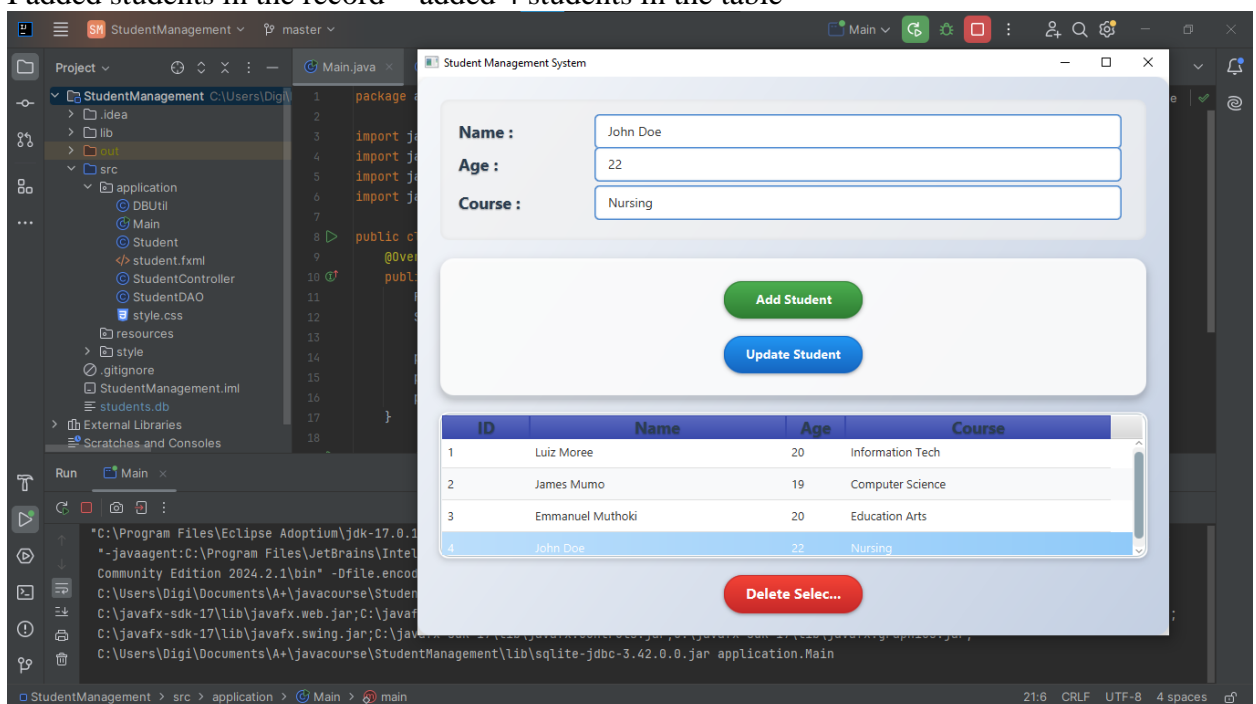
ACADEMIC YEAR: 2025/2026

1.User Interface



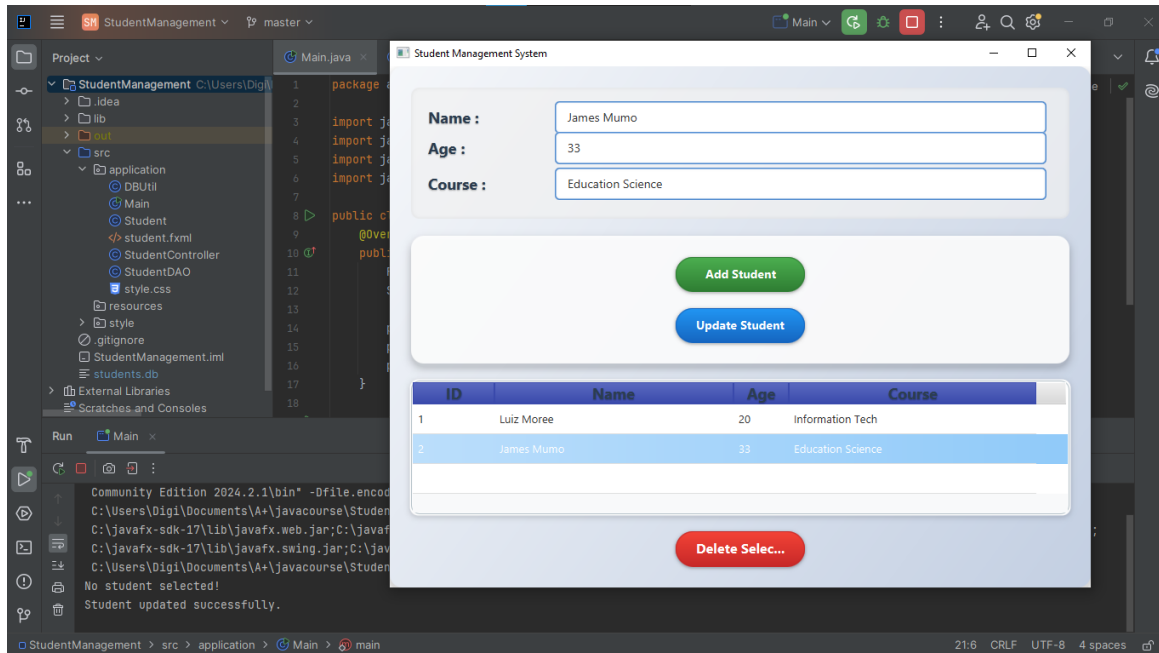
2. Adding students

I added students in the record - added 4 students in the table



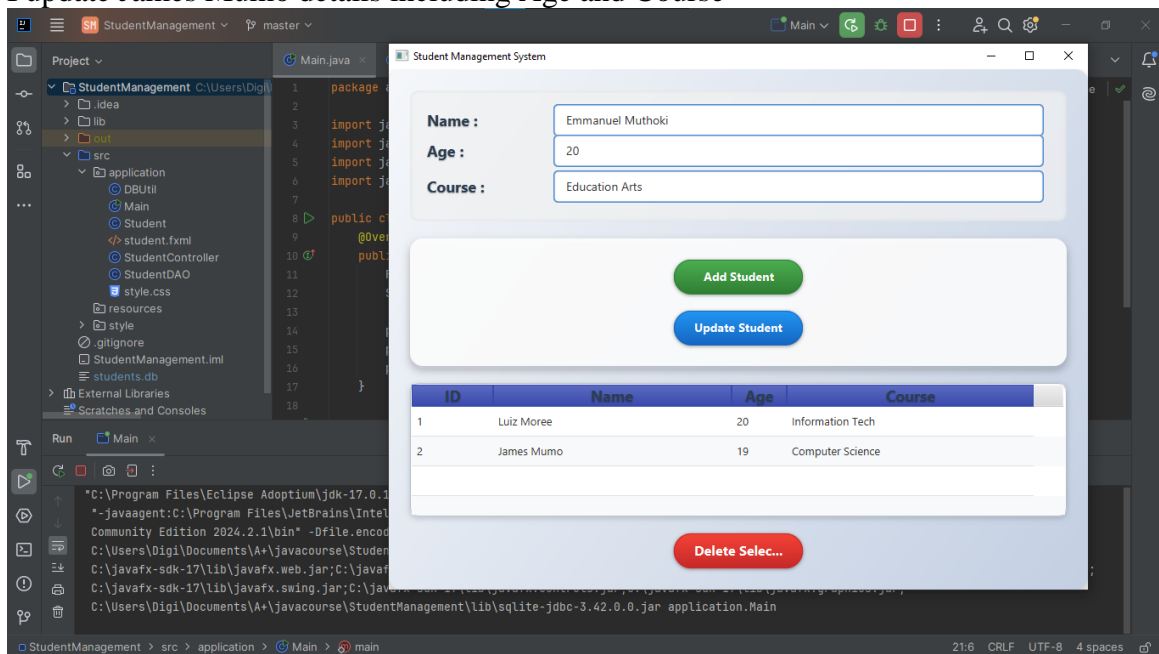
3. Deleting students in the list.

I deleted 2 student from the table – number 3 and 4



4. Updated student details in the table

I update James Mumo details including Age and Course



5. Code for update/Edit functionality

A) StudentController.java

```
package application;

import javafx.collections.ObservableList;
import javafx.fxml.FXML;
import javafx.scene.control.*;

public class StudentController {

    @FXML private TextField txtName;
    @FXML private TextField txtAge;
    @FXML private TextField txtCourse;

    @FXML private TableView<Student> tableView;
    @FXML private TableColumn<Student, Integer> colId;
    @FXML private TableColumn<Student, String> colName;
    @FXML private TableColumn<Student, Integer> colAge;
    @FXML private TableColumn<Student, String> colCourse;

    @FXML
    public void initialize() {
        colId.setCellValueFactory(cell -> new
        javafx.beans.property.SimpleIntegerProperty(cell.getValue().getId()).asObject());
        colName.setCellValueFactory(cell -> new
        javafx.beans.property.SimpleStringProperty(cell.getValue().getName()));
        colAge.setCellValueFactory(cell -> new
        javafx.beans.property.SimpleIntegerProperty(cell.getValue().getAge()).asObject());
        colCourse.setCellValueFactory(cell -> new
        javafx.beans.property.SimpleStringProperty(cell.getValue().getCourse()));

        // Add row selection listener for auto-filling fields
        tableView.setOnMouseClicked(event -> onRowSelect());

        loadStudents();
    }

    private void loadStudents() {
        try {
            ObservableList<Student> list = StudentDAO.getStudents();
            tableView.setItems(list);
        } catch (Exception e) {
```

```

        e.printStackTrace();
    }
}

```

@FXML

```

private void addStudent() {
    try {
        String name = txtName.getText();
        int age = Integer.parseInt(txtAge.getText());
        String course = txtCourse.getText();

        StudentDAO.addStudent(name, age, course);
        loadStudents();
        clearFields();
    } catch (Exception e) {
        e.printStackTrace();
    }
}

```

@FXML

```

private void deleteStudent() {
    Student selected = tableView.getSelectionModel().getSelectedItem();
    if (selected != null) {
        try {
            StudentDAO.deleteStudent(selected.getId());
            loadStudents();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}

```

@FXML

```

private void onRowSelect() {
    Student = tableView.getSelectionModel().getSelectedItem();
    if (student != null) {
        txtName.setText(student.getName());
        txtAge.setText(String.valueOf(student.getAge()));
        txtCourse.setText(student.getCourse());
    }
}

```

@FXML

```

private void updateStudent() {
    Student selected = tableView.getSelectionModel().getSelectedItem();
    if (selected == null) {

```

```

        System.out.println("No student selected!");
        return;
    }

    try {
        String name = txtName.getText();
        int age = Integer.parseInt(txtAge.getText());
        String course = txtCourse.getText();

        Student updatedStudent = new Student(selected.getId(), name, age, course);

        StudentDAO.updateStudent(updatedStudent);

        loadStudents();
        clearFields();
    } catch (NumberFormatException e) {
        System.out.println("Please enter a valid age!");
        e.printStackTrace();
    } catch (Exception e) {
        e.printStackTrace();
    }
}

private void clearFields() {
    txtName.clear();
    txtAge.clear();
    txtCourse.clear();
}
}

```

b) StudentDAO.java

```

package application;

import javafx.collections.FXCollections;
import javafx.collections.ObservableList;

import java.sql.*;

public class StudentDAO {

```

```

public static void addStudent(String name, int age, String course) throws SQLException {
    String sql = "INSERT INTO students(name, age, course) VALUES(?, ?, ?)";
    try (Connection conn = DBUtil.getConnection();
        PreparedStatement stmt = conn.prepareStatement(sql)) {
        stmt.setString(1, name);
        stmt.setInt(2, age);
        stmt.setString(3, course);
        stmt.executeUpdate();
    }
}

```

```

public static ObservableList<Student> getStudents() throws SQLException {
    ObservableList<Student> list = FXCollections.observableArrayList();
    String sql = "SELECT * FROM students";

    try (Connection conn = DBUtil.getConnection();
        Statement stmt = conn.createStatement();
        ResultSet rs = stmt.executeQuery(sql)) {

        while (rs.next()) {
            list.add(new Student(
                rs.getInt("id"),
                rs.getString("name"),
                rs.getInt("age"),
                rs.getString("course")
            ));
        }
    }
    return list;
}

```

```

public static void deleteStudent(int id) throws SQLException {
    String sql = "DELETE FROM students WHERE id=?";
    try (Connection conn = DBUtil.getConnection();
        PreparedStatement stmt = conn.prepareStatement(sql)) {
        stmt.setInt(1, id);
        stmt.executeUpdate();
    }
}

```

```

public static void updateStudent(Student student) {
    String sql = "UPDATE students SET name = ?, age = ?, course = ? WHERE id = ?";

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

```

```
stmt.setString(1, student.getName());  
stmt.setInt(2, student.getAge());  
stmt.setString(3, student.getCourse());  
stmt.setInt(4, student.getId());  
  
stmt.executeUpdate();  
System.out.println("Student updated successfully.");  
  
} catch (SQLException e) {  
    e.printStackTrace();  
}  
}  
}
```