JAVA PROJECT

```
Import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
import java.util.ArrayList;
import java.sql.*;

// ===== COURSE CLASS =====
class Course {
    private String code, title, instructor;
    private int credits, maxSeats;

    public Course(String code, String title, int credits, String instructor, int maxSeats) {
        this.code = code;
        this.title = title;
        this.credits = credits;
        this.instructor = instructor;
        this.maxSeats = maxSeats;
    }

    public String getCode() { return code; }
    public String getTitle() { return title; }
    public String getInstructor() { return instructor; }
    public int getCredits() { return credits; }
    public int getMaxSeats() { return maxSeats; }

    @Override
    public String toString() {
        return code + " - " + title + " (" + maxSeats + " seats)";
    }
}

// ===== STUDENT CLASS =====
class Student {
    private String studentID, name, email;

    public Student(String id, String name, String email) {
        this.studentID = id;
```

```
        this.name = name;
        this.email = email;
    }

    public String getId() { return studentID; }
    public String getName() { return name; }
    public String getEmail() { return email; }

    @Override
    public String toString() {
        return studentID + " - " + name + " (" + email + ")";
    }
}

// ===== MAIN GUI CLASS =====
public class StudentManagementSystemGUI extends JFrame {
    private ArrayList<Student> students = new ArrayList<>();
    private ArrayList<Course> courses = new ArrayList<>();
    private JComboBox<Student> studentBox;
    private JComboBox<Course> courseBox;
    private JTextArea output;

    public StudentManagementSystemGUI() {
        setTitle("Student Course Registration System (SQLite)");
        setSize(750, 600);
        setDefaultCloseOperation(EXIT_
        setLocationRelativeTo(null);

        JTabbedPane tabs = new JTabbedPane();
        Font font = new Font("SansSerif", Font.PLAIN, 14);

        // ===== STUDENT TAB =====
        JPanel studentPanel = new JPanel(new GridBagLayout());
        GridBagConstraints g = new GridBagConstraints();
        g.insets = new Insets(5, 5, 5, 5);
        g.fill = GridBagConstraints.HORIZONTAL;

        JTextField sID = new JTextField(15), sName = new JTextField(15), sEmail = new JTextField(15);
        JButton addStudent = new JButton(" Add Student");

        g.gridx = 0; g.gridy = 0; studentPanel.add(new JLabel("Student ID:"), g);
        g.gridx = 1; studentPanel.add(sID, g);
        g.gridx = 0; g.gridy = 1; studentPanel.add(new JLabel("Name:"), g);
        g.gridx = 1; studentPanel.add(sName, g);
        g.gridx = 0; g.gridy = 2; studentPanel.add(new JLabel("Email:"), g);
        g.gridx = 1; studentPanel.add(sEmail, g);
        g.gridx = 1; g.gridy = 3; studentPanel.add(addStudent, g);

        addStudent.addActionListener(_ -> {
```

```
        if (sID.getText().isEmpty() || sName.getText().isEmpty() || sEmail.getText().isEmpty()) {
            output.setText(" Please fill all fields!");
            return;
        }

        try (Connection conn = getConnection()) {
            String sql = "INSERT INTO students(studentID, name, email) VALUES(?, ?, ?)";
            PreparedStatement pst = conn.prepareStatement(sql);
            pst.setString(1, sID.getText());
            pst.setString(2, sName.getText());
            pst.setString(3, sEmail.getText());
            pst.executeUpdate();

            Student s = new Student(sID.getText(), sName.getText(), sEmail.getText());
            students.add(s);
            updateStudentCombo();
            output.setText(" Student Added: " + s);

            sID.setText(""); sName.setText(""); sEmail.setText("");
        } catch (SQLException ex) {
            output.setText("Error: " + ex.getMessage());
        }
    });

    // ===== COURSE TAB =====
    JPanel coursePanel = new JPanel(new GridBagLayout());
    GridBagConstraints c = new GridBagConstraints();
    c.insets = new Insets(5, 5, 5, 5);
    c.fill = GridBagConstraints.HORIZONTAL;

    JTextField cCode = new JTextField(15), cTitle = new JTextField(15), cCredits = new
JTextField(15),
            cInst = new JTextField(15), cSeats = new JTextField(15);
    JButton addCourse = new JButton("Add Course");

    c.gridx = 0; c.gridy = 0; coursePanel.add(new JLabel("Course Code:"), c);
    c.gridx = 1; coursePanel.add(cCode, c);
    c.gridx = 0; c.gridy = 1; coursePanel.add(new JLabel("Title:"), c);
    c.gridx = 1; coursePanel.add(cTitle, c);
    c.gridx = 0; c.gridy = 2; coursePanel.add(new JLabel("Credits:"), c);
    c.gridx = 1; coursePanel.add(cCredits, c);
    c.gridx = 0; c.gridy = 3; coursePanel.add(new JLabel("Instructor:"), c);
    c.gridx = 1; coursePanel.add(cInst, c);
    c.gridx = 0; c.gridy = 4; coursePanel.add(new JLabel("Max Seats:"), c);
    c.gridx = 1; coursePanel.add(cSeats, c);
    c.gridx = 1; c.gridy = 5; coursePanel.add(addCourse, c);

    addCourse.addActionListener(_ -> {
        if (cCode.getText().isEmpty() || cTitle.getText().isEmpty() || cCredits.getText().isEmpty()
```

```
                || cInst.getText().isEmpty() || cSeats.getText().isEmpty()) {
            output.setText(" Fill all fields!");
            return;
        }

        try (Connection conn = getConnection()) {
            String sql = "INSERT INTO courses(courseCode, title, credits, instructor, maxSeats)
VALUES(?,?,?,?,?)";
            PreparedStatement pst = conn.prepareStatement(sql);
            pst.setString(1, cCode.getText());
            pst.setString(2, cTitle.getText());
            pst.setInt(3, Integer.parseInt(cCredits.
            pst.setString(4, cInst.getText());
            pst.setInt(5, Integer.parseInt(cSeats.
            pst.executeUpdate();

            Course co = new Course(cCode.getText(), cTitle.getText(), Integer.parseInt(cCredits.
                  cInst.getText(), Integer.parseInt(cSeats.
            courses.add(co);
            updateCourseCombo();
            output.setText(" Course Added: " + co);

            cCode.setText(""); cTitle.setText(""); cCredits.setText(""); cInst.setText("");
cSeats.setText("");
        } catch (SQLException ex) {
            output.setText(" Error: " + ex.getMessage());
        }
    });

    // ===== REGISTRATION TAB =====
    JPanel regPanel = new JPanel(new GridBagLayout());
    GridBagConstraints r = new GridBagConstraints();
    r.insets = new Insets(5, 5, 5, 5);
    r.fill = GridBagConstraints.HORIZONTAL;

    studentBox = new JComboBox<>();
    courseBox = new JComboBox<>();
    JButton register = new JButton(" Register Course");

    r.gridx = 0; r.gridy = 0; regPanel.add(new JLabel("Select Student:"), r);
    r.gridx = 1; regPanel.add(studentBox, r);
    r.gridx = 0; r.gridy = 1; regPanel.add(new JLabel("Select Course:"), r);
    r.gridx = 1; regPanel.add(courseBox, r);
    r.gridx = 0; r.gridy = 2; regPanel.add(register, r);

    register.addActionListener(_ -> {
        Student s = (Student) studentBox.getSelectedItem();
        Course co = (Course) courseBox.getSelectedItem();
        if (s == null || co == null) {
```

```
            output.setText(" Select student and course!");
            return;
        }

        try (Connection conn = getConnection()) {
            String sql = "INSERT INTO registrations(studentID, courseCode, regDate) VALUES(?,?,?)";
            PreparedStatement pst = conn.prepareStatement(sql);
            pst.setString(1, s.getId());
            pst.setString(2, co.getCode());
            pst.setString(3, new java.sql.Date(System.
            pst.executeUpdate();

            output.setText(" Registration Successful: " + s.getName() + " -> " + co.getTitle());
        } catch (SQLException ex) {
            output.setText(" Error: " + ex.getMessage());
        }
    });

    // ===== OUTPUT AREA =====
    output = new JTextArea(8, 40);
    output.setEditable(false);
    output.setFont(font);
    output.setBorder(

    tabs.add(" Student Management", studentPanel);
    tabs.add(" Course Management", coursePanel);
    tabs.add(" Registration", regPanel);

    add(tabs, BorderLayout.CENTER);
    add(new JScrollPane(output), BorderLayout.SOUTH);

    // Initialize database + load data
    initDatabase();
    loadDataFromDB();

    setVisible(true);
}

private void updateStudentCombo() {
    studentBox.removeAllItems();
    for (Student s : students) studentBox.addItem(s);
}

private void updateCourseCombo() {
    courseBox.removeAllItems();
    for (Course c : courses) courseBox.addItem(c);
}

// ===== DATABASE METHODS =====
```

```
private Connection getConnection() throws SQLException {
    String url = "jdbc:sqlite:student_
    return DriverManager.getConnection(
}

private void initDatabase() {
    try (Connection conn = getConnection(); Statement stmt = conn.createStatement()) {
        stmt.executeUpdate("""
            CREATE TABLE IF NOT EXISTS students (
                studentID TEXT PRIMARY KEY,
                name TEXT NOT NULL,
                email TEXT NOT NULL
            )
        """);

        stmt.executeUpdate("""
            CREATE TABLE IF NOT EXISTS courses (
                courseCode TEXT PRIMARY KEY,
                title TEXT NOT NULL,
                credits INTEGER NOT NULL,
                instructor TEXT NOT NULL,
                maxSeats INTEGER NOT NULL
            )
        """);

        stmt.executeUpdate("""
            CREATE TABLE IF NOT EXISTS registrations (
                id INTEGER PRIMARY KEY AUTOINCREMENT,
                studentID TEXT NOT NULL,
                courseCode TEXT NOT NULL,
                regDate TEXT NOT NULL,
                FOREIGN KEY(studentID) REFERENCES students(studentID),
                FOREIGN KEY(courseCode) REFERENCES courses(courseCode)
            )
        """);
    } catch (SQLException ex) {
        JOptionPane.showMessageDialog(
    }
}

private void loadDataFromDB() {
    students.clear();
    courses.clear();

    try (Connection conn = getConnection()) {
        Statement stmt = conn.createStatement();

        // Load students
        ResultSet rs1 = stmt.executeQuery("SELECT * FROM students");
```

```
            while (rs1.next()) {
                Student s = new Student(rs1.getString("
                students.add(s);
            }
            updateStudentCombo();

            // Load courses
            ResultSet rs2 = stmt.executeQuery("SELECT * FROM courses");
            while (rs2.next()) {
                Course c = new Course(
                    rs2.getString("courseCode"),
                    rs2.getString("title"),
                    rs2.getInt("credits"),
                    rs2.getString("instructor"),
                    rs2.getInt("maxSeats")
                );
                courses.add(c);
            }
            updateCourseCombo();
        } catch (SQLException ex) {
            output.setText(" Error loading data: " + ex.getMessage());
        }
    }

    public static void main(String[] args) {
        SwingUtilities.invokeLater(
    }
}
```