

# **DEPARTMENT OF COMPUTER SCIENCE AND TECHNOLOGY**

## **2025/2026 SEMESTER 1**

### **CAT 2 ASSIGNMENT**

#### **APPLICATION**

---

JavaFX GUI program created with SceneBuilder and connected to a database using JDBC.

Student Management System where you can:

- Add students (Name, Age, Course).
- View student records in a **TableView**.
- Delete a student.

#### **Step 1: Database Setup**

We'll use **SQLite** (lightweight, no server needed).

-- Create database file: students.db

```
CREATE TABLE students (
    id INTEGER PRIMARY KEY AUTOINCREMENT,
    name TEXT NOT NULL,
    age INTEGER NOT NULL,
    course TEXT NOT NULL
);
```

Save this as students.db in your project folder.

#### **Step 2: Project Structure**

StudentManagement/

```
  |- src/
  |  |- application/
  |  |  |- Main.java
  |  |  |- DBUtil.java
```

```
| |   └─ Student.java  
| |   └─ StudentDAO.java  
| |   └─ StudentController.java  
| └ resources/  
|   └ student.fxml (designed using SceneBuilder)  
└ students.db
```

### Step 3: Student Model

```
package application;  
  
public class Student {  
  
    private int id;  
  
    private String name;  
  
    private int age;  
  
    private String course;  
  
  
    public Student(int id, String name, int age, String course) {  
  
        this.id = id;  
  
        this.name = name;  
  
        this.age = age;  
  
        this.course = course;  
  
    }  
  
  
    // Getters & Setters  
  
    public int getId() { return id; }  
  
    public String getName() { return name; }  
  
    public int getAge() { return age; }  
  
    public String getCourse() { return course; }  
  
}
```

### Step 4: Database Utility

```

package application;

import java.sql.*;

public class DBUtil {

    private static final String URL = "jdbc:sqlite:students.db";


    public static Connection getConnection() throws SQLException {
        return DriverManager.getConnection(URL);
    }

}

```

### **Step 5: Data Access Object (DAO)**

```

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();
    }
}

```

#### **Step 6: SceneBuilder FXML (student.fxml)**

In **SceneBuilder**, design a GUI:

##### **VBox → GridPane (for inputs)**

- TextField: txtName
- TextField: txtAge
- TextField: txtCourse
- Button: btnAdd

##### **TableView (fx:id=tableView)**

- TableColumn: colId

- TableColumn: colName
- TableColumn: colAge
- TableColumn: colCourse

**Button: btnDelete**

Save as **student.fxml**.

**Step 7: Controller**

```
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())));
    }
}
```

```

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();
        txtName.clear(); txtAge.clear(); txtCourse.clear();
    } 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();
}
}
}
}
}

```

### **Step 8: Main Class**

```

package application;

import javafx.application.Application;
import javafx.fxml.FXMLLoader;
import javafx.scene.Scene;
import javafx.stage.Stage;

public class Main extends Application {

    @Override
    public void start(Stage primaryStage) throws Exception {
        FXMLLoader loader = new FXMLLoader(getClass().getResource("/student.fxml"));
        Scene scene = new Scene(loader.load());
        primaryStage.setTitle("Student Management System");
        primaryStage.setScene(scene);
        primaryStage.show();
    }

    public static void main(String[] args) {
        launch(args);
    }
}

```

### **Features in this program:**

- GUI designed in **SceneBuilder**.
- Database interaction using **JDBC**.
- Add, display, and delete student records.
- Uses **TableView** to show live data.

### **ASSIGNMENT TASK:**

**Extend** this application with **update functionality** (editing student details) so that the system becomes fully **CRUD**?

**(20 MARKS)**

### **INSTRUCTIONS**

1. Create a word processor document (.docx)
2. Your document should have:
  - a) Cover page that clearly shows your registration number, name and course program, academic year and semester.
  - b) Screenshot for all user interfaces of your application
  - c) Copy and paste code for update/Edit functionality
3. Your final document should be converted to **.PDF format**, renamed to your registration number then emailed to [pkasyoka@seku.ac.ke](mailto:pkasyoka@seku.ac.ke)
4. The subject of your email should be SIT311\_ASSIGNMENT\_yourRegNo (e.g SIT311\_ASSIGNMENT\_G126\_9090\_2050)  
(If you do not format the subject of your email, your assignment will not be received)
5. DEADLINE for submitting your work via email is 5<sup>th</sup> January 2026.