**Student Management System (SMS) desktop application.**

**Task description**

This project is to develop a functional **Desktop Student Management System (SMS)** using **C# (.NET)** and **SQLite3**. The primary goal is to create a simple yet efficient application for managing student records, while demonstrating teamwork and technical skills.

**2. Group Members and Roles**

1. **IRUTABOSE Yoramu**
   * **Email:** [petitirutabyoseyoramu@gmail.com](mailto:petitirutabyoseyoramu@gmail.com)
   * **Phone:** +250781014782
   * **Reg Number:** 23/22136
   * **Role:** Team Lead & Deployment Manager
   * **Responsibilities:** Oversee project progress, coordinate team activities, manage documentation submission & Compile final project files, package deliverables, and ensure proper file naming.
2. **MASONGA SHEMA Prince** 
   * **Email:** [masongashemap@gmail.com](mailto:masongashemap@gmail.com)
   * **Phone:** +250786781268
   * **Reg Number:** 23/20861
   * **Role:** Frontend Developer
   * **Responsibilities:** Design and implement GUI using WinForms and Razor Views.
3. **IZABAYO HARANIRA Jean Luc Severin** 
   * **Email:** [izabayojeanlucseverin@gmail.com](mailto:izabayojeanlucseverin@gmail.com)
   * **Phone:** +250790635888
   * **Reg Number:** 23/22493
   * **Role:** Backend Developer
   * **Responsibilities:** Handle SQLite database integration and CRUD operations.
4. **ISHIMWE Kevin** 
   * **Email:**  ishimwekevin108@gmail.com
   * **Phone:** +250726373441
   * **Reg number:** 23/22823
   * **Role:** Quality Assurance (QA) Analyst
   * **Responsibilities:** Test application features, ensure functionality, submit screenshots.
5. **ISHIMWE HABYARIMANA Regis** 
   * **Phone:** +250789149102
   * **Role:** Documentation Specialist
   * **Responsibilities:** Prepare explanation documents, summarize challenges, and include implementation details.

**3. Application Overview**

The **Student Management System** (SMS) enables users to:

* Add, View, Edit, and Delete student records.
* Maintain a database of students, including fields like **StudentID**, **Name**, **Age**, and **Grade**.
* Navigate through a clean, user-friendly graphical interface.
* Use an installer for easy deployment.

**4. Development Process**

**4.1 Project Initialization**

* Created a GitHub repository to share the project files.
* Defined team roles and responsibilities.
* Scheduled milestones for database setup, GUI design, testing, and deployment.

**4.2 Database Setup**

* Developed a SQLite3 database named StudentDB.sqlite.
* Created a table named Students with the following structure:

|  |  |  |
| --- | --- | --- |
| **Field** | **Data Type** | **Description** |
| StudentID | INTEGER (PK) | Auto-incremented ID |
| Name | TEXT | Student's full name |
| Age | INTEGER | Student's age |
| Grade | TEXT | Student's grade |

* Inserted sample data for testing.

**4.3 GUI Design**

* Developed a **Main Form** with a data grid to display student records.
* Created forms for **Adding/Editing Students**.
* Added buttons for CRUD operations: **Add, Edit, Delete, Refresh List**.
* Ensured a responsive layout using WinForms docking and anchoring.

**4.4 CRUD Logic Implementation**

* We used parameterized queries to implement CRUD functionalities:
  + **Create:** Inserted new records into the database.
  + **Read:** Fetched all student records and displayed them in a data grid.
  + **Update:** Edited and updated existing records.
  + **Delete:** Removed records without affecting the database integrity.

**4.5 Quality Assurance**

* We tested the application for:
  + **Functionality:** Ensured CRUD operations are working correctly.
  + **Usability:** Verified user-friendliness of the interface.
  + **Edge Cases:** Checked validation for empty fields and invalid inputs.
* Documented results with screenshots.

**4.6 Deployment**

* Compiled the project in **Release Mode**.
* Packaged the application using **Inno Setup**.
* Included the StudentDB.sqlite file and created a configuration file for the database connection.
* Tested the installer on multiple machines for compatibility.

**5. Challenges and Solutions**

1. **Database Connection Issues:**
   * **Challenge:** Initial database file path caused errors.
   * **Solution:** Used relative paths in the configuration file.
2. **GUI Responsiveness:**
   * **Challenge:** Forms did not scale properly on different screen sizes.
   * **Solution:** Used anchoring and docking features in WinForms.
3. **Validation Errors:**
   * **Challenge:** Invalid inputs caused crashes.
   * **Solution:** Added input validation and error messages.

**6. Screenshots and Features**

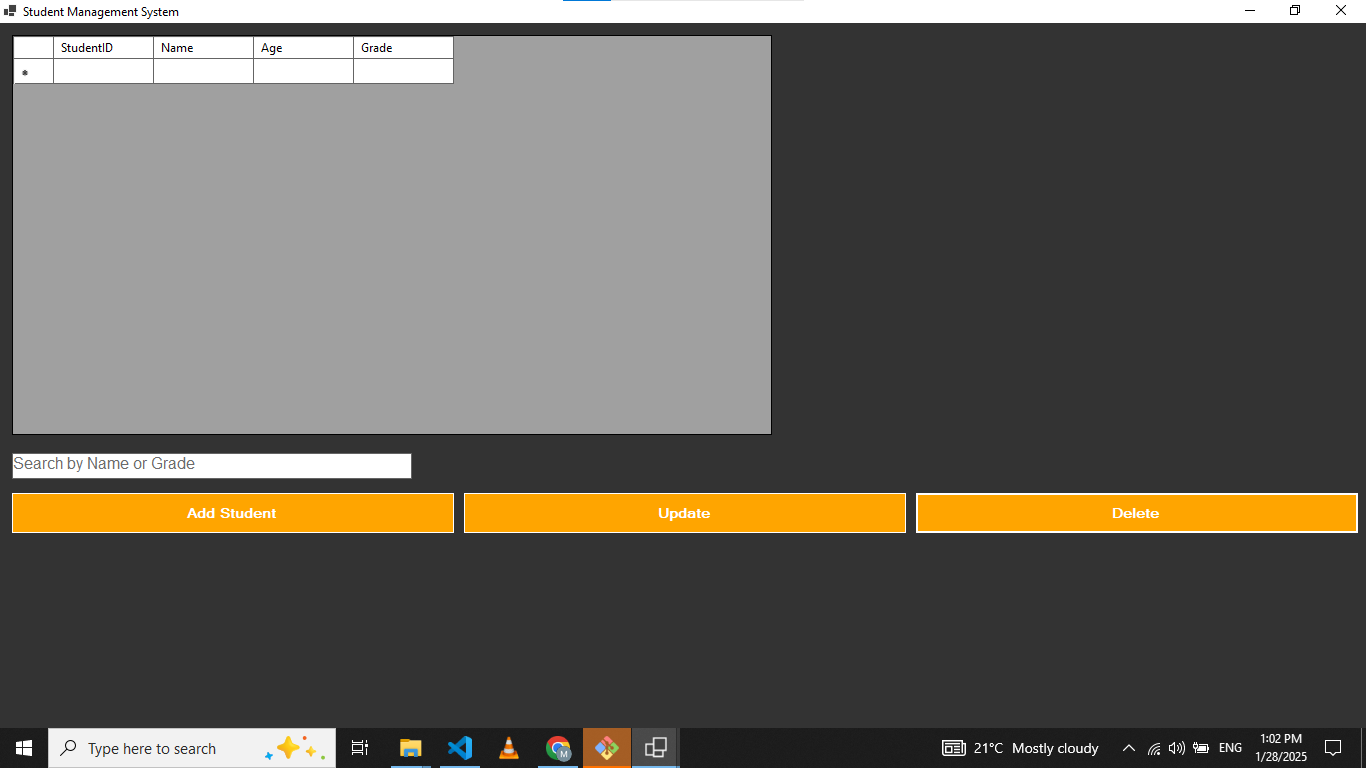
**Main Form**

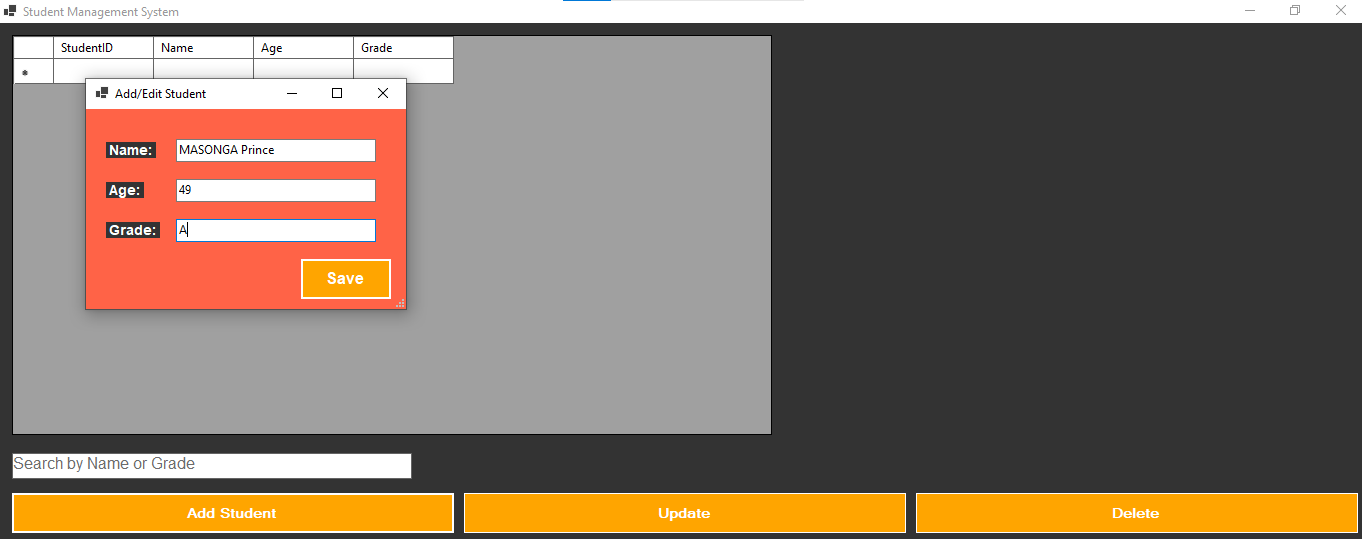
* Displays all student records in a grid view.
* Buttons for **Add, Edit, Delete**, and **Refresh List**.

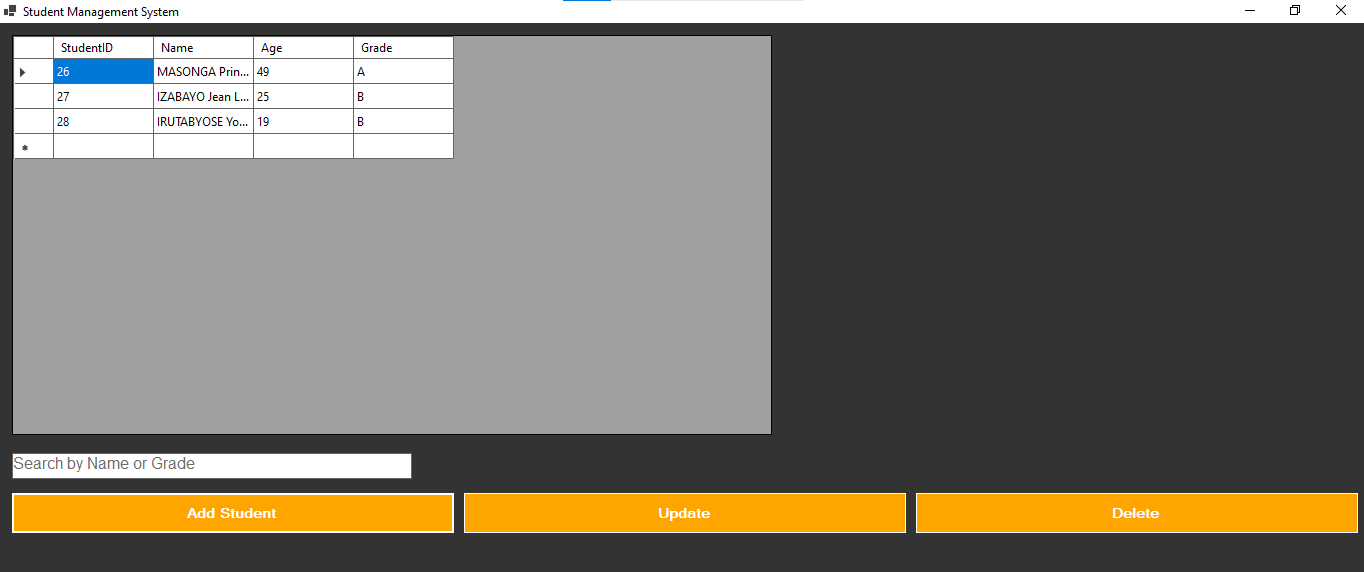
**Add/Edit Form**

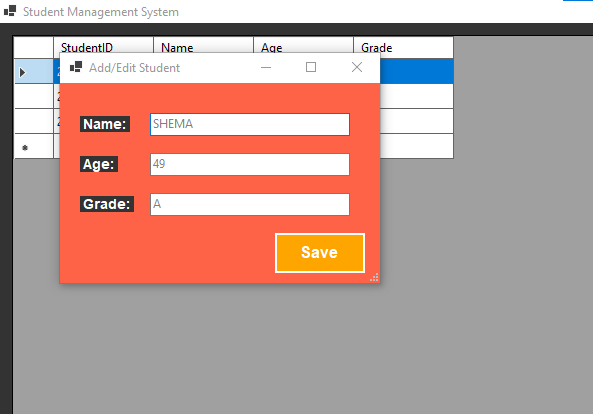
* Input fields for student details.
* Validation to ensure required fields are filled.

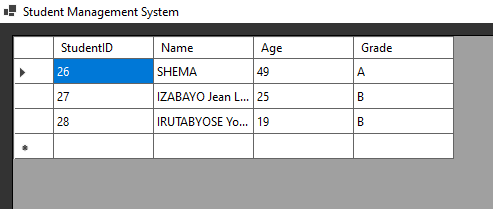
*Dashboard*



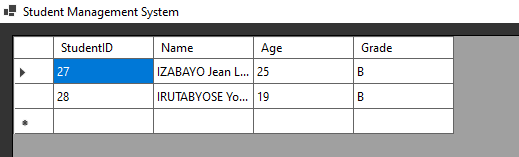
**Add student**

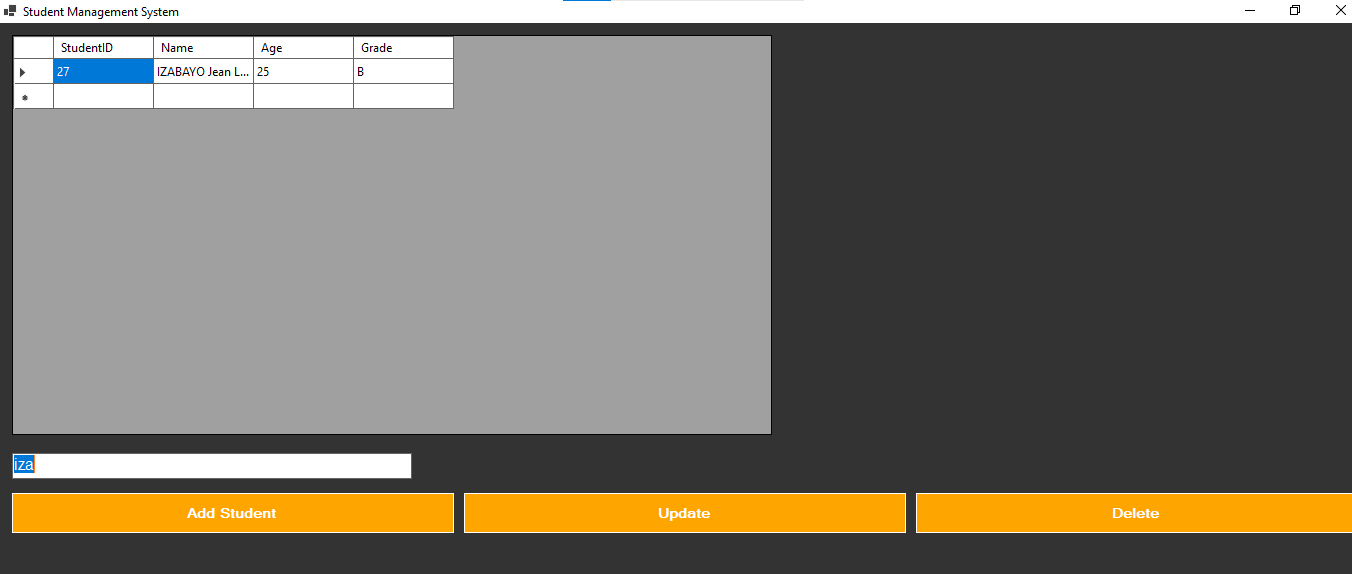
**View students**

**Update/edit Students**

View updated students

**Delete student**



Real-time Search student

1. **Github link:**

https://github.com/Masonga-Dev/Student\_Management\_System.git

1. **YouTube Presentation Link:**

https://youtu.be/V7gmIy304vY