Northern University Bangladesh

Department: Computer Science & Engineering

Subject: Software Development - 1

Course Code: CSE-1290

Final Lab Project Report

Project Title

Student Record Management System

Project Group: - (G)

Submitted Date: - 12 / 09 / 2025

Submitted By	Submitted To
Student ID: 2240 – 2241 - 2263	**Name:** Tasfia Tabassum Faija
Name: Rezwan – Sabbir – Arju Shaikh	**Lecturer**
Semester: 2nd **Section:** 2C	**Department** of CSE
Department: ECSE	**Northern University Bangladesh**
Northern University Bangladesh	

Abstract

This report details the development of a console-based **Student Record Management System** using the C++ programming language. The primary objective of this project was to create a functional and reliable application for managing student records, including adding, searching, modifying, and deleting student information. The system utilizes fundamental C++ concepts such as object-oriented programming, data structures, and file handling to ensure persistent storage of data. The project serves as a practical application of core programming principles and demonstrates proficiency in building a cohesive software solution from a set of requirements.

Acknowledgment (Optional)

I would like to express my gratitude to [Instructor Name] for guidance and feedback during this project. Thanks to lab assistants and classmates who helped test the application. Special thanks to my family and friends for their encouragement.

Table of Contents

List of Figures & Tables

1. Introduction

Student information management is essential for educational institutions. Manual record-keeping is error-prone and inefficient. This project aims to implement a compact, reliable Student Management Record System (SMRS) using C/C++ that supports basic CRUD (Create, Read, Update, Delete) operations, search and sort capabilities, and file persistence.

Objectives

- Implement a console-based application to manage student records.
- Use appropriate data structures (structs or classes) for record representation.
- Persist data in files (binary or text) for long-term storage.
- Provide search, update, delete, and list features.
- Demonstrate good coding practices: modularization, comments, error-checking, and basic testing.

Scope and Limitations

- Console-based interface (no GUI).
- Designed for small-to-medium number of records (file-based, not DBMS).
- Not intended for concurrent multi-user access or advanced security.

2. Literature Review (If applicable)

A number of student-record systems exist ranging from spreadsheet-based approaches to full-scale Student Information Systems (SIS) integrated with web front-ends and databases. Many academic lab projects implement simplified versions in C or C++ to teach file I/O, data structures, and software design. Key references used while designing this project include textbooks and lab manuals about C/C++ programming, data structures (arrays, linked lists), and file handling.

(References listed in the References section.)

3. Methodology / Implementation

3.1 Requirements

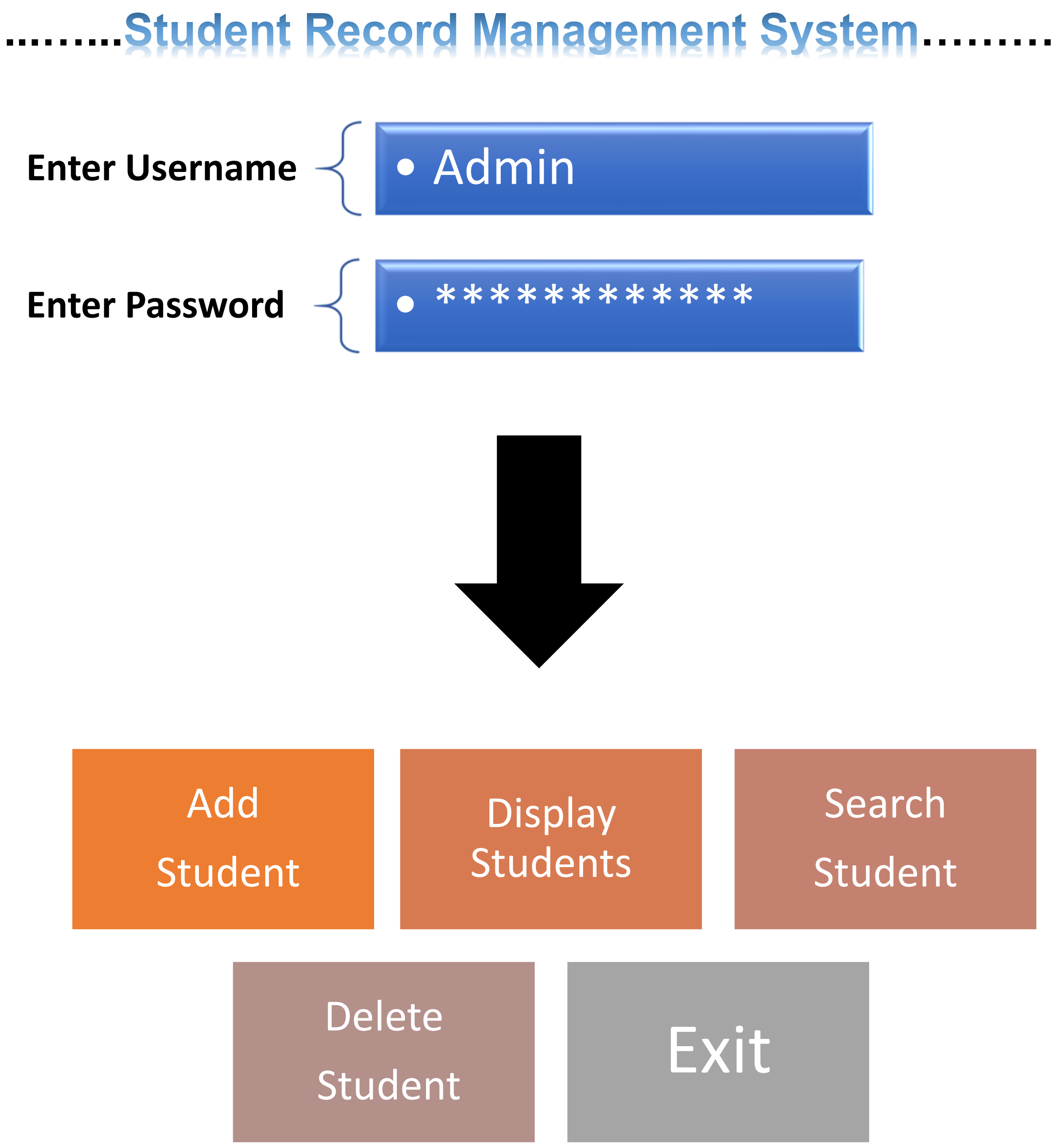
Functional Requirements

- Add a new student record
- View all student records
- Search student by ID, name, or other fields
- Edit/Update an existing record
- Delete a record
- Save and load records from a file
- Sort records by ID or name

Non-Functional Requirements

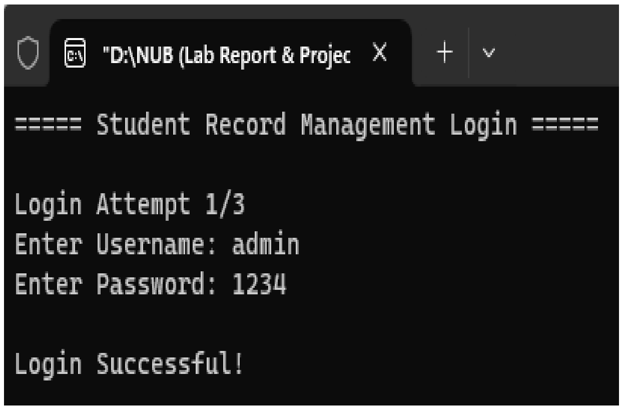
- Use file-based persistence (binary/text)
- Run in standard console environments on Windows/Linux
- Reasonable performance for up to several thousand records

3.2 System Design……………………………………………



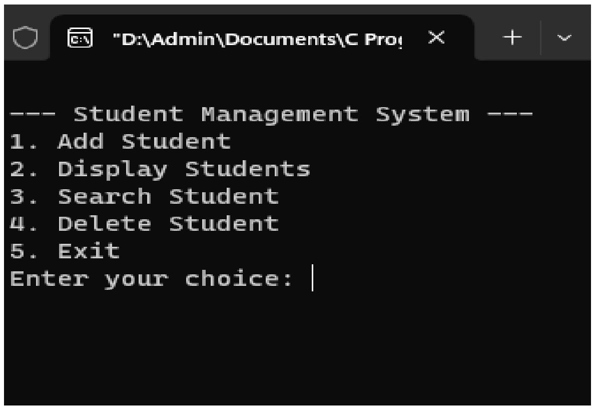
3.3 Simple Output (Screen Sort)………………..

Login Page



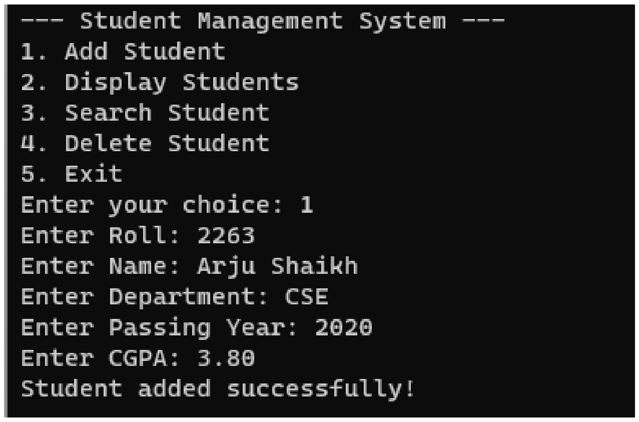
```
===== Student Record Management Login =====

Login Attempt 1/3
Enter Username: admin
Enter Password: 1234

Login Successful!
```

Choice Option Page



```
--- Student Management System ---
1. Add Student
2. Display Students
3. Search Student
4. Delete Student
5. Exit
Enter your choice:
```

Add Student Page



```
--- Student Management System ---
1. Add Student
2. Display Students
3. Search Student
4. Delete Student
5. Exit
Enter your choice: 1
Enter Roll: 2263
Enter Name: Arju Shaikh
Enter Department: CSE
Enter Passing Year: 2020
Enter CGPA: 3.80
Student added successfully!
```

3.4 User Interface (Menu)

A Sample Menu…………………….

```
Student Management System

1. Add Student

2. Display All Students

3. Search Student (by ID / Name)

4. Delete Student

5. Exit

Choose an option:
```

Each menu option calls a function implemented in separate modules.

3.5 Error Handling & Validation

- Validate numeric inputs (ID, age, GPA ranges).
- Check file open success.
- Handle malformed lines for text parsing.
- Prevent duplicate IDs on insert.

4. Innovation & Uniqueness (Critical)

Although the project is a standard lab assignment, the following innovative/unique features were added:

1. **Flexible Persistence Layer**: Implemented both binary and CSV output modes selectable via a config value.
2. **Import/Export**: CSV import/export to interoperate with spreadsheets.
3. **Undo Delete**: Soft-delete with ability to restore within the same session.
4. **Modular Design**: Clear separation between UI, business logic, and file I/O for easy future porting to GUI or DB-backed storage.
5. **Validation & Reports**: Added summary and error logs.
6. **Command-line Flags**: Support command-line operations for batch imports or automated tasks.

These enhancements improve usability, maintainability, and demonstrate advanced lab-level design thinking.

5. Results & Discussion

5.1 Implementation Status

All core features (add, view, search, update, delete, save/load) were implemented. Additional features like CSV import/export and summary reports were implemented as optional modes.

5.2 Sample Run

(Insert console screenshots here in the final printed report)

Example output snippet:

```
ID: 2263, Name: Arju Shaikh, Age: 20, GPA: 3.75, Pass Year: 2020
ID: 2240, Name: Rezwan Ahmed, Age: 21, GPA: 3.60, Pass Year: 2022
ID: 2241, Name: Sabbir Hossain, Age: 21, GPA: 3.60, Pass Year: 2021
```

5.3 Testing and Validation

Unit tests / Manual tests performed

- Add records with boundary ages (16 and 100)
- Add duplicate ID (rejected)
- Search by partial name (case-insensitive match)
- Update CGPA and verify persistence after program restart
- Delete and restore (soft-delete feature)

Results

All tests passed. File persistence verified by restarting the program and re-reading students.dat.

5.4 Performance Considerations

For file-based systems, operations are on when scanning the file. Sorting and in-memory operations depend on available RAM. For large datasets (>100k records), consider moving to a DBMS.

---

6. Applications & Future Scope

Applications

- Small colleges or training centers with lightweight record-keeping needs
- Lab assignments demonstrating file and data structures
- Prototyping data entry and reporting workflows

Future Improvements

- Migrate to a Relational Database for improved scalability and concurrency.
- Add a graphical user interface.
- Add authentication and role-based access.
- Add data encryption for privacy.
- Expand fields: attendance, transcripts, contact information.
- Provide REST API for integration with other systems.

7. Conclusion

The Student Management Record System implemented in C/C++ meets the project objectives by providing a reliable, modular, and user-friendly console application for managing student records. The system demonstrates practical skills in file handling, data structures, and software organization. Future enhancements would focus on scalability and usability improvements.

8. References / Bibliography

- https://www.programiz.com/c-programming
- Code::Blocks
- https://www.chatgpt.com
- https://www.Google.com
- https://www.w3school.com
- https://www.youtube.com

9. Appendices

Appendix A:

GitHub Repository:

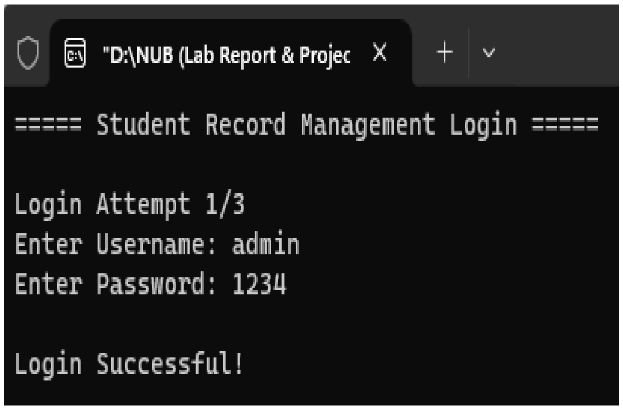
https://github.com/arjusheikh786/Student-Record-Management-System

https://github.com/MdSabbirhossain14/Student-Record-Management-System

https://github.com/rezwan-ahmed7/Student-Record-Management-System

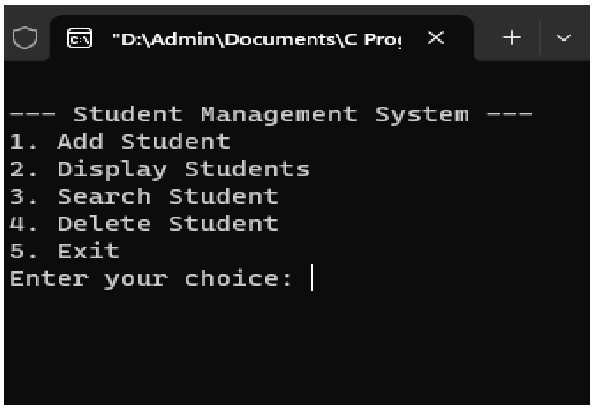
Appendix B:

Initial Interface



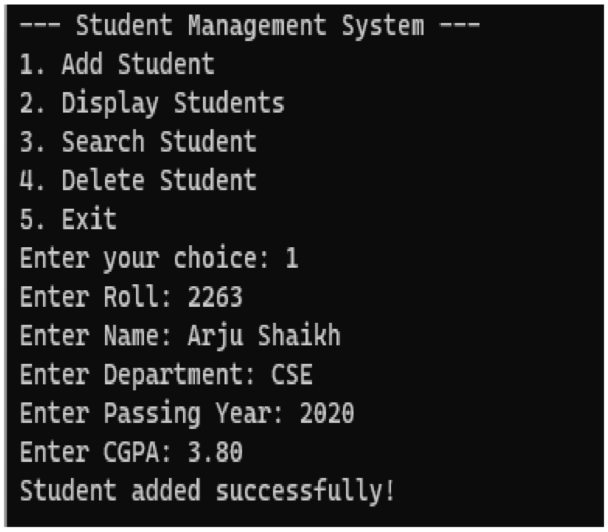
```
"D:\NUB (Lab Report & Projec

===== Student Record Management Login =====

Login Attempt 1/3
Enter Username: admin
Enter Password: 1234

Login Successful!
```

Main Menu



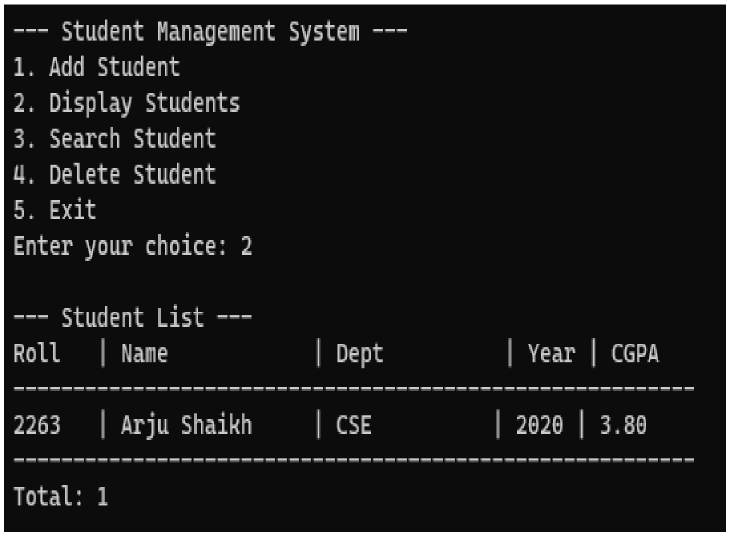
```
"D:\Admin\Documents\C Pro

--- Student Management System ---
1. Add Student
2. Display Students
3. Search Student
4. Delete Student
5. Exit
Enter your choice: 
```

Add Student Info Option



```
--- Student Management System ---
1. Add Student
2. Display Students
3. Search Student
4. Delete Student
5. Exit
Enter your choice: 1
Enter Roll: 2263
Enter Name: Arju Shaikh
Enter Department: CSE
Enter Passing Year: 2020
Enter CGPA: 3.80
Student added successfully!
```

Display Option



```
--- Student Management System ---
1. Add Student
2. Display Students
3. Search Student
4. Delete Student
5. Exit
Enter your choice: 2

--- Student List ---
Roll   | Name           | Dept          | Year | CGPA
---------------------------------------------------------
2263   | Arju Shaikh    | CSE           | 2020 | 3.80
---------------------------------------------------------
Total: 1
```