Final Report: Student Record System

Team Members:

Nistha Aryal

Mohammad Nemer

Course: COMPSCI-2

Instructor: Oke Onwuka

Date: 5/12/2025

Project Background and Inspiration

Being Computer Science students, we were eager to do a project that would not only demonstrate our understanding of major C++ concepts but also be of practical use. We developed a Student Record Management System, which is an essential application for student record organization in schools and colleges. The system simulates the functionality of a simple database where users can insert, display, search, and sort student records using a C++ console application.

After receiving feedback from our professor to make the project more complex, we developed our application further by adding a Graphical User Interface (GUI) through Windows Forms. This made the project more interactive, user-friendly, and sophisticated.

Design and Implementation

Our project is structured in a modular manner, with several C++ files to maintain the organization of the code:

• main.cpp – Handles the menu and main loop of the console version.

•\tstudent.h – Defines the student struct and all function prototypes.

• student.cpp – Wraps fundamental features such as the addition, retrieval, display, save, and reload of student records.

• students.txt – Stores student information persistently between executions.

We then included:

• Mainform.cpp / Mainform.h – Controls the behavior and structure of the Windows Forms graphical user interface.

Key Features Implemented

• Add, view, search, and sort student records.

• File I/O via .txt files for data loading and saving.

• Sorting student records by GPA using Bubble Sort.

• GUI form to input student information and display it immediately.

• All functionality modularized into header/source files.

Applied C++ Concepts

• Structures and Pointers – To deal with linked lists and dynamic data.

• File Streams – Employed for persistent read/write operations.

• Sorting Algorithms – Bubble Sort used for sorting students based on GPA.

• Event-based GUI logic – Introduced through button click handlers in Windows Forms.

Issues Encountered

We encountered some difficulties, especially in executing GPA sorting and file handling debugging. The file writes and reads initially did not work because of formatting problems. It also took some learning to add the GUI, such as event handling and control binding, but we managed to do this by dividing tasks into functions and testing each one extensively.

Lessons Learned

This project gave us practical experience in C++ beyond typical textbook exercises. We developed skills in modular programming, debugging, data structures, and stream input/output. We learned version control (GitHub), user interface design, and the implementation of theoretical class concepts to real software development through teamwork.

Potential Developments

• Implement functions to update and delete student records.

• Improve error handling and input validation.

• Enhance the GUI by switching to Qt or enabling file-based logs.

• Investigate class-based object orientation (instead of structs). • Implement student ID uniqueness checks and GPA validation.

Conclusion

By incorporating a graphical user interface into our console application, we not only fulfilled the minimum requirements of the project but also developed a product that we can be really proud of. In doing this, we learned lessons in teamwork, planning, debugging, and software design principles. The completed application is a worthy contribution to our software development portfolios.