***Role Base Student Management System***

Submitted by: Muhammad Ali Naqvi | Neeraj Khemani

Course title: Object Oriented Programming

Instructor: Sir Shafique Ur Rehman

Submission: 11th May 2025

## Executive Summary:

The goal of this project is to create a terminal-based "Role Based Student Management System" using C++ to address the problem of student data management. This system uses role-based access control to allow for faster and more efficient management of student records. Student data can be created, read, updated, and deleted (CRUD operations).  
The project fundamentally applies Object-Oriented Programming (OOP) concepts like encapsulation, inheritance, and polymorphism, but also blends in functional programming approaches to maximize the flexibility of C++. Additionally, third-party C++ libraries are employed to enhance the functionality, resulting in a modern and versatile management system.

## Introduction:

### Background:

Managing large amounts of student data manually leads to errors, inefficiencies, and data breaches. Traditional systems do not scale well and require significant administrative effort. By utilizing C++ and a terminal-based UI, this project offers a lightweight, flexible, and secure alternative that leverages both OOP design principles and the functional capabilities of C++.

## Goals:

* Design a multi-role system for secure student data management.
* Implement full CRUD operations based on user roles.
* Provide a smooth terminal-based user interface using Windows API (windows.h) and console interaction (conio.h).
* Minimize manual effort while ensuring data security and integrity.

## Project Description:

### System Components:

* **Admin Role**:  
  Full control; can create and manage all user types (Moderators, Faculty, Students).
* **Moderator Role**:  
  Can create and manage student records.
* **Faculty Role**:  
  Can upload or update student academic records (e.g., marks, subjects).
* **Student Role**:  
  Can only view their own data, no edit permissions.

### Terminal UI:

* **Windows.h** is used for console window manipulation (cursor position, colors, etc.).
* **Conio.h** provides keyboard input functions (getch(), etc.) for smooth interaction without needing to press Enter.

## Innovation:

* **Role-based Access Control**: Users are only allowed operations permitted by their role, improving security.
* **Blended Programming Paradigm**: Combination of OOP and functional programming to maximize C++ capabilities.
* **Terminal UX Enhancements**: Rich terminal interaction without external GUIs, creating a lightweight but powerful interface.
* **Extensibility**: Future roles and permissions can be easily added without disrupting the system.

## Implementation and Development:

### Development Process:

* **Class Design**:
  + Base User class with derived classes: *Admin, Moderator, Faculty, Student*.
  + Student records are handled using file operations to ensure data persistence.
* **Role Management**:
  + Admin creates/modifies users, Moderator manages students, Faculty uploads results, and Students view their data.
* **Data Handling**:
  + CRUD operations for students performed by authorized users.
  + Secure file operations for storing and retrieving student records.

### Tools & Technologies:

* Language: C++
* Libraries: iostream, fstream , windows.h, conio.h
* Platform: Windows Terminal

## Team Contributions:

[Muhammad Ali]:

* Implementation of role based system.
* Include/Src management of C++ code.
* Implementation of main function.

[Neeraj Khemani]:

* Filing handling for data management.
* Debugging and code optimization.
* Implementation of main function.