Project Guide: Final Assignment C# Application

1. Introduction

This project is a desktop application built using C# and SQLite. It is designed to manage a college or institute's academic system including users, courses, marks, timetable, and more. The application provides a user-friendly interface for administrators to efficiently manage data related to students, lecturers, staff, and courses.

2. Overview

The application consists of several modules, including:

- User management (Admin, Staff, Lecturer, Student)
- Course management
- Marks management
- Timetable scheduling

Each module allows CRUD operations (Create, Read, Update, Delete) and is integrated with an SQLite database backend for persistent storage.

3. Functions

- **User Registration & Management**: Register new users with roles, update and delete users.
- **Course Management**: Add, update, and delete courses.
- **Marks Management**: Add, update, and delete students' marks for various courses and subjects.

- **Timetable Module**: Manage and display course schedules.

4. Key Features

- Role-based user management.
- Data validation and error handling in forms.
- Dynamic data grids for easy viewing and selection.
- Modular design for easy maintenance and scalability.
- Integration with SQLite database for lightweight and efficient storage.

5. Technologies Used

- **Programming Language:** C#
- **Framework:** .NET Windows Forms
- **Database:** SQLite
- **Development Environment:** Visual Studio
- **Version Control:** Git & GitHub

6. Challenges

- Handling asynchronous database operations without blocking UI.
- Managing state and data consistency across multiple user controls.
- Implementing secure password storage and role management.
- UI responsiveness and validation of user inputs.

7. Solutions

- Used synchronous SQLite commands with proper exception handling for simplicity.
- Designed modular user controls to encapsulate functionality.

- Planned role-based access at the UI level (basic approach).
- Added validation checks and confirmation dialogs to prevent invalid operations.

8. Future Updates

- Implement password hashing and improved security.
- Add user authentication and authorization features.
- Expand timetable features to support more complex scheduling.
- Create reports and analytics for marks and attendance.
- Refactor UI using modern frameworks like WPF or React Native.

9. Conclusion

This project demonstrates the fundamentals of desktop application development with database integration. It provides a practical solution for managing academic data and can be extended further to include advanced features and better security practices

10. References

- OpenAI ChatGPT (GPT-4) — Assisted in coding, debugging, and project planning.
- Microsoft Docs — Official documentation for C# and SQLite.
- Stack Overflow — Community Q&A for programming challenges.
- GitHub — Version control and project hosting platform.