Requirement Definition

Services:
- Students will be able to add/drop courses, search for courses, and print their schedule
- Instructors will be able to print their schedule and class list, as well as search for courses
- Admins will be able to add/remove courses from the system, add/remove users, add/remove students from a course, and search/print rosters and courses

Constraints:
- Database must work for 100 students, 10 instructors, and 1 admin
- Users should not be able to access functions outside of their classification

Goals:
- System should provide a scheduling service which will allow students, instructors, and admins to add courses, search for courses, print schedules, etc.
- System should include multiple semesters, print-out of schedule, and scheduling preferences
- Needs both a database of users and a database of courses

Estimated duration:
5/11/23 - 5/19/23

System and Software Design

System will be coded in C++ with Visual Studio 2022 as the IDE of choice. Additionally, SQLite will be used as the means for creating the system's database.

The system will consist of a main file from where the code will be run, a base User file, and three additional files which inherit from User: Student, Instructor, and Admin. The unique functions for each of these types of users should be private so that they cannot be used outside of their intended classification of user.

Estimated duration:
5/19/23 - 5/26/23

Implementation and Unit Testing

To get a functional version of the system running, the first thing that needs to be done is setting up all of the required databases. Most of the functions in the system are meant to view or interface some database, and so they must exist in some form (even just filled with test items) before any of the functions can be tested. These databases should include:
- Users
- Courses
- Rosters (depending on implementation)

With these databases filled, the next step will be to implement each of the functions. Starting with one type of user and testing it before moving on to the next will be most efficient for avoiding mistakes. Each type of user will be completed and tested before starting the next step.

Estimated duration:
5/26/23 - 6/9/23

Integration and System Testing

Once all individual parts have been completed and tested, it is time to integrate the code into a genuine system and test the functionality there. There are multiple ways to tackle this problem.

The simplest way to achieve functionality would be to do all of the interfacing through the command window, the default output space for C++. From there, a user could follow the printed prompts and interface the system entirely through command line inputs.

A more complex way would be to use some type of GUI as an interface. There are libraries for C++, but also some other options include Javascript, .NET, and Qt.

Estimated duration:
6/9/23 - 6/30/23

Operation and Maintenance

Continue to use the system while documenting bugs and other issues. After enough time has passed or if a significantly important bug is found, return to the beginning phase and update all steps accordingly.

Estimated duration:
6/30/23 - End of semester