Requirements Specification

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

Component Analysis

1) https://fahad-cprogramming.blogspot.com/2016/03/school-management-cpp-project-source-code.html
- School scheduling system made entirely in C++. Includes much of the needed functionality, but stores data in arrays instead of a dedicated database

2) https://github.com/GibbonEdu/core
- Gibbon is entirely open source and includes much of the needed functionality, and can be edited relatively easily with the supporting documentation on their website
- The downside of this software is that most of the code is in PHP or Javascript, which is outside our expertise and diverts heavily from C++

3) ⭐https://www.youtube.com/watch?v=nrB35Dh2xac&ab_channel=DJOamen
- This video is a tutorial which uses MySQL (not SQLite) and C++ to create a school management system
- Out of the three, this is the most fleshed out and similar to the project we are trying to create. There are some features missing, and most of the focus is on using SQL, but other than that most of the functionality is the same. Additionally, the tutorial comes with creating a GUI
- The only major downside is the video creator has paywalled the source code, so we must either follow the tutorial and create our own code or pay for the source code and modify it from there

Estimated duration:
5/19/23 - 6/2/23

Requirements Modification

*****Each number corresponds to the number in the previous step. This is also the step where a decision on what components to use will be made.**

1) Must modify the code to include databases. This would mean taking all of the functions which reference arrays and changing them to reference an SQL database. Additionally, there is some functionality missing, so it will need to be added manually or taken from some other source
2) Most of the needed functionality is here, and the software is free and open source. However, using this software will have a learning curve due to the fact that it is wholly new to us and is coded in PHP and Javascript, which are mostly unfamiliar to us. Furthermore, since most if not all of the functionality is already here, it begs the question if modifications even need to be done at all.
3) This tutorial is most similar to what we want out of the system, and is coded in familiar C++. All of the functionality is there and more, since it even includes extraneous information about users (such as DOB and address) and courses (such as department, dean, and general topics used in the course)

Estimated duration:
6/2/23 - 6/9/23

System Design with Reuse

Now with the components chosen, the system needs to be roughly molded in order to prepare for development, where these components will be used in the final design. Each requirement should be checked against the components being used. If the functionality exists to fulfill the requirement, we can move on. Otherwise, we will need to either write the code ourselves to achieve that functionality, or go searching for another existing program which does what we need. Once it is clear that all requirements can be fulfilled using the existing components, we can move on to the next step.

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

Development and Integration

Once all of the code has been modified to be able to fit all of the requirements, development on the final version can begin. This is where all of the components that have been repurposed will finally be put to use in fulfilling all of the requirements and goals.

Important things to keep in mind during this stage is if any of the functionality we thought was possible to add actually isn't, we need to go back and decide how to add it. Additionally, if we do need to add any new outside components, we need to make sure any components we use are fully open source so that there is no worry of legal trouble.

Estimated duration:
6/23/23 - 7/21/23

System Validation

The system should be validated through user and developer testing. All bugs and issues should be ironed out, and then the final version of the system can be shipped out.

Estimated duration:
Final two weeks of project