**Introduction**

The focus of this project was to create and implement an application that would simulate a course scheduling and registration system. My application is able to create rooms, courses and students; as well as enroll those students into courses and display them on screen. The application uses a GUI interface to allow the user to navigate through the different options available.

**Requirements**

The application is required to contain the following functionalities.

- The user must be able to specify the number of rooms available for courses.

- The user must be able to add or delete rooms.

- The user must be able to add or delete students.

- The user must be able to add or delete courses.

- Rooms can be created by specifying the number, and max amount of courses.

- Courses can be created by specifying name, room number, hour, and participants.

- Students can be created by specifying their name and phone number.

- The application can display all courses, all courses in a certain room, all courses at a certain time, and all courses being attended by a single student.

- A student cannot attend more than one course at one time slot, and a room cannot contain more than one course at one time slot.

**Design**

These are the classes used in the application and each of their functionalities.

AddCourseFrame

A JFrame for GUI allowing the user to add a course to the schedule by specifying its name, room number, time, and participants.

AddRoomFrame

A JFrame for GUI allowing the user to add a room to the schedule by specifying its number and amount of courses.

AddStudentFrame

A JFrame for GUI allowing the user to add a student to the schedule by specifying their name and phone number.

DeleteCourseFrame

A JFrame for GUI allowing the user to delete a course by specifying its name.

DeleteRoomFrame

A JFrame for GUI allowing the user to delete a room by specifying its number.

DeleteStudentFrame

A JFrame for GUI allowing the user to delete a student by specifying their name.

EnrollStudentFrame

A JFrame for GUI allowing the user to enroll a student in a course by specifying the student name and the course name.

UnenrollStudentFrame

A JFrame for GUI allowing the user to unenroll a student in a course by specifying the student name and the course name.

MainFrame

The main JFrame for GUI allowing the user to navigate and access all the different options for the schedule. It also allows the user to display any courses using the time or students as well as save and import files.

SetNumberOfRooms

A JFrame for GUI requiring the user to enter the max number of rooms available for scheduling.

Course

The Course class, which implements all the code relating to courses such as adding, deleting, and displaying courses. Manages time and is also is able to manipulate some information from the Student and Room classes.

Student

The Student class, which implements all the code relating to students such as adding, deleting, enrolling, and unenrolling students. Is able to manipulate some information from the Course and Room classes.

Room

The Room class, which implements all the code relating to rooms, such as adding, deleting, and setting the maximum number of rooms. Is able to manipulate some information from the Student and Course classes.

**Challenges**

The main challenges I faced with this project were simply finding a system that worked. I was thinking of doing everything from the main method, but felt that there should be a much more efficient way of doing things. Problem was I couldn’t think of a way I could keep things simple and consistent. I finally decided to go with array list which worked out perfectly. It was much easier to grab information and store it between classes this way, and helped keep me focused and organized.

For the second obstacle, it was figuring out the differences between static and non-static variables and methods. I put a lot of time in messing around with the code to see what I could set as static in order to access the information in other classes, without creating a mess of problems within the class itself. After I figured this out, it was a lot easier to synchronize the classes together and get my system functioning.

Lastly, the final obstacle was time. Even with a good idea of what I needed to do, creating the backbone of the application took a lot longer than I expected. It took a full 3 days and nights without break where all I did was simply eat, sleep and code. Staying focused was the hardest part, but in the end I was able get all the pieces working and create a working application which was easy to use. The system I set up was so simple, it only took around 20 minutes to set up a command line interface for the program. Not mentioning that it made building the GUI much easier and take up less than 20% of the time I spent building the heart of the application.

**Shortcomings**

Although I spent a lot of time on the project, it wasn’t perfect. To start, the program doesn’t account for extra whitespace. For example if a user entered a course with the name CMPSCI122 with an extra space at the end of it and then went to delete that course entering the same name but without the space. The application would report that the course doesn’t exist do to the missing space. This could lead to a lot of confusion, and I wish I knew how to fix this.

Next I also wish that my GUI interface worked a bit more efficiently when adding, deleting, or displaying information. As I have it now it can display courses at the click of a button, but it only shows the course name. I would have liked to display all the information relating to the course and more. I also believe the program would be much better if the user could have that list of courses and add or delete them by simply selecting them. I don’t have enough knowledge or time to implement these features, but maybe sometime in the future I will come back and add them when I have gained more experience.

**Conclusion**

Overall, I am very pleased with my work that went into creating this project. There are many features I feel can be added to improve this application as I stated in the shortcomings section. I learned more from this project than any other I have done before and feel a tiny bit more confident in my programming potential and skill. I believe this will be another stepping stone for my career as a Software Engineer, and I’m looking forward to seeing what I can accomplish in the future using what I learned.