**Design Process**

**Emerging Tech**

**Staff Schedule**

Jacob Heiser

Full Sail: Scheduler is a program design to optimize Full Sail University current systems that oversees the scheduling of staff to the many courses that need to be taught. Full Sail has had an issue with many staff being overly burdened with the workload assigned to them, and other staff have the exact opposite issue of having no work. Using the provided data that is used to create the current schedules for staff, this program finds new ways to compare the data together to make a unique and fair way of scheduling staff automatically with minimal manual adjustments.

During the creation of this program, I was challenged to develop, design, and problem solve multiple tasks. My first set of tasks were to identify my client’s problem(s) and understand the provided data. Technically write and design the logic to the program and code. Developing my own database system, creating a custom process to efficiently utilize staff. Developing a way to manually adjust the automation of staff utilization, and the creation of multiple new reports to identify potential future problems with Full Sail’s scheduling system.

**Identifying the problem and understanding the data:** The first step was to understand the issue. Why does my client needs this, how is it affecting them, what should it be doing? Understanding their issue gives me insights on what direction to approach. I also need to know what data I would be provided with and what its current uses are. To play with that data and see what uses I can find for it that are not being utilized currently. With Full Sail not being able to utilize their staff efficiently, and a bunch of data to see the courses, staff identifiers, and more I had several ideas on how to approach.

**Writing the Logic:** Writing a rough draft to get my ideas together, I crafted a flow chart to visual the order of the steps needed to be taken. How to separate the program into different parts, different tasks to tackle one bit at a time. With the flow chart, I was able to design a final draft of my logic and begin working.

Seeing the logic, I would need to break the program into several different parts each responsible for its own role in the program by feeding it data and receiving part of the final data.

**A custom Database:** Looking at the data given, there is only so much that could be done in its base form. To organize the data, and compare data in more complicated and advanced ways, I needed to make my own database and combine certain data together. Rather than using MySQL, I did it all in python making my own database, limiting the number of outside resources needed. With a custom database, I can more efficiently compare data together, and manipulate data in a way that was not possible beforehand.

**Custom Process to Utilize staff:** By creating a database, I was able to compare data and loop through staff, and courses that needed to be taught in a new light. To better assign staff to teach, I created a leveling system of 7 levels to fairly distribute the workload across all staff. Looping through the levels, with plenty of error checking to find possible issues and notify management before it becomes an issue.

**Manual adjustments:** With the error checking, I thought about the fact that the program will never be perfect and that there may be need to make manual adjustment. Creating a way for management to see the current staff assignments and giving them the ability make changes when an exception occurs.

**Refining a UI:** A program would be hard to use without a user interface to interact on. I was struggling to design one. Seeing my teammate take his own approach inspired me to design the interface the way I did. It gave me hints on how to lay things out in an easy-to-read way. Then all that was left was for me to program the elements to use the created code. Now I have a nice UI that can automate a scheduling system.

**The Final Outputs:** Through many hours of debugging and testing the code, I have noticed several issues with the data that Full Sail provided. Not only did I think to error test that data and possible related data, I also thought it be beneficial to create additional reports and data regarding the different issues that could occur. Courses that could not be assigned, staff that were not teaching, courses with unusual amounts of students.

Overall, I really challenged myself to create and solve as many of the problems that Full Sail was and would be facing as a result of poor data management and efficiently assigning their staff. Of course the main focus was to solve Full Sail’s main concern of Staffing, and solving other tasks would be less concerning but a good benefit if time allows. I managed to not just efficiently solve the issue at hand while it could still be improved upon, I managed to identify several other issues as a result of my progress and approach solutions to a few of them.