Gabriel Martinez-Amezaga, Kaz Tam, Gabriella Best, Andisha Safdariyan, Danish Safdariyan

WIP Report

CS 3110: Data Structures and Functional Programming

March 27, 2024

**WIP Report**

Vision

To reiterate what we reported in MS1, we seek to create a niche calendar program that has various functions and a consistent, aesthetically pleasing, graphical user interface. A specific goal we have in mind is to create three main uses: a regular calendar, a schedule manager, and a task manager. The regular calendar will include normal functionality such as creating repeating events or notating important dates. The schedule manager will be integrated with Cornell scheduler (as in, importing a file exported from scheduler), and be very similar to the implementation of scheduler. The task manager will be a normal task manager that keeps track of impending tasks. The purpose of this program is to provide a time management method that is easy to use and convenient as well.

Some of us within the group are currently frustrated with the different calendar and task management options available to us. More often than not, a calendar app does not encompass what we want to use out of it (ex: different views that include different events/schedules). Additionally, we want to combine this calendar functionality with an integrated task manager window in the program as well. The goal is to create a computer program for a task management system and organizational system that encompasses different uses and is simple to use.

**WIP Report Continued**

Summary of Progress

During MS1, we created a vision for what we want our program to do. For MS2, we wanted to do two essential things that would set the foundation for the rest of the project. First, we wanted to figure out a data structure to represent calendar data. For that we decided on using a simple dictionary/map that currently has the structure of [(k , v)] where the keys are tuples with column and row, and the values are dates. In other terms, the structure is [( (row \* col) , date)]. This is fine for our current implementation, but we don’t foresee using this exact implementation for the rest of the project. Now second, we wanted to find a GUI library that we would work with for the project. After experimenting with the OCaml graphics library and deciding it would not work long term, we settled on using Bogue. The link to Bogue is below: <https://github.com/sanette/bogue/blob/master/README.md>.

For our WIP, we implemented a basic GUI using Bogue and represented the map/dictionary on that GUI. This is just a basic GUI and representation that will definitely not stay the same long term, but we wanted to establish a basic foundation to work off of, which we did.