Olivia (Olive) Shannon

CMPT 305L

Software Development 1

Final Project Writeup

When I first started to decide what I wanted my project to accomplish, I was a bit paralyzed by how broad a range of topics I could choose from. But I knew that whatever I made, I wanted it to be something that was actually useful for myself and other people. I've always struggled with staying organized and being productive because of some pretty severe focus problems, so naturally, this was the first issue that came to mind that seemed tackle-able with the help of technology. The final idea I decided on, a program that calculates and tracks a user's productivity over time, was inspired by a lack of a tool like that in my own life; I use the notes tool on my iPhone daily to make to-do lists for tasks I need to complete each day, and check them off as I go, but there's no feature that would automatically calculate and record my progress in completing the tasks.

Initially, I was unsure of how to actually structure such a program, but after our classes on Object-Oriented Programming and data structures, I realized that it would require at the very minimum multiple classes, and some kind of database to store the information. Since I don't know PHP or SQL, my best option for connecting a functional, external database was the Google Sheets API. After deciding on this path, I was able to fully diagram and plan out what I intended my final project to look like.

In terms of requirements, a command line, the gradle build system, an internet connection, a functioning browser, and the Google Sheets API library would be required to successfully build and run it.

There would be a main class, ProductivityTracker, where the code to collect user input and determine what other classes to run would be located. In this class, the program would collect user entered input for the data fields (day, month, list of tasks, how many there were total, and how many the user actually completed), and use these as parameters for an object instantiated using the constructor DateObject class. From there, the main method would call the GoogleSheets external class, where the methods required for interacting with the database were written. The method addEntry() would add the instance of the DateObject created previously to the first unfilled row in the spreadsheet database. After this, the user could tell the program through command line input to calculate their weekly productivity, monthly productivity, or display the task list for a specific entry date. The first two options would run methods in the GoogleSheets class that would retrieve a certain amount of the most recent entries' this.productivity attribute (either 7 or 30), add them to an array, then perform the basic algebra required to determine their total productivity. Displaying task list would retrieve and display the this.taskList attribute for the database entry they specified. All of these functions and the overall functionality of the GoogleSheets class require multiple dependencies and a myriad of jars and libraries to run, which are all included in the downloadable Google Sheets java database.

However, this was not where my project ultimately ended up. I successfully coded the structure of the project and the command line functionality, and all that was left was to integrate the code from the Google Sheets API- this is where I hit a wall, so to speak. I followed the tutorial and understood the concepts behind integrating it into the code, but after days of poring over the documentation and Stack Overflow, I was still completely stumped by the actual code in it, as well as how to integrate it. Although I managed to successfully download and run their sample version of it, there was no guide on integrating it into another application, and attempting to figure this out without a guide led me down a rabbit-hole of documentation references that seemed to have no clear answer. I included all the code for this first attempt in the directory for this project inside the folder “p2 first draft.”

After it became clear that successfully integrating the API would take a much longer amount of time than I had left until the due date, I decided to cut my losses and code a second version of the project, one that didn’t require the Sheets API, so that I could still display the functionality of my code and its intended structure, albeit without some of the functions I would have liked to include. This version has two classes: ProductivityTracker, the main class, and the DateObject constructor class. The program prompts the user for input and returns their daily productivity. After this, it gives the user two options, executed from an if/if else/else statement depending on the user input. The first option takes user input for total tasks that were to have been completed that week, and the total amount of tasks that were, and returns their weekly productivity with the method getWeeklyProductivity(). The second option takes user input and firstly calculates their productivity for the first and last days of the week, then calculates and returns their change in productivity over the course of the week via the method getProductivityChange(). There is a second if/if else/else statement nested in this if/else code block that handles formatting the answer depending on if their productivity change was negative, positive, or the same.

What this version of the project lacks in robust functionality, however, it makes up for in ease of use and adaptability. Unlike the previous draft, this project requires only a computer's command line and whatever JDK is installed on the operating system. It has no external dependencies or libraries required for it to function, and only possesses 2 classes; one being the main class, and one serving as an object constructor. This means it requires little perquisites for setup like installing build programs or downloading libraries, and is straightforward to use.

Although I am disappointed that the final, functional version of my project doesn't have all of the features I intended for it, I believe that my project is a genuinely useful tool, and, more importantly, a solid starting point for me to develop it further into a robust, user-friendly program that can help people like myself visualize and monitor their productivity over time.