Conception Phase

Written by Emre Ertürk

My goal is to deliver a habit tracker app that is easy to use and allows the user to store their progress.

General Idea

Visualization

The main issue with non-GUI apps is that the graphical user interface is non-existent. Therefore, my solution must have some form of basic visualizations. I choose a basic data frame as my solution because it is easy to read and understand. As seen in Table 1, the index of the data frame will be the habit itself, and the remaining columns will display the specifications of each habit.

Habit Name	Specification in less than 10 words	Are you currently tracking this habit? [y/n]	Daily or Weekly	Current Streak	Longest Streak
Drink more water	Living a hydrated life	y	Daily	3	10
Code daily	improve my coding skills	y	Daily	12	12
Learn Turkish	strenghten my mother tongue	y	Weekly	13	17
Learn Georgian	strenghten my father's language	n			
Workout	Move my body for mental & physical health	y	Daily	33	33

Table 1: Data frame of habits

Saving the data

Next, saving the data frame is crucial to track the analytics with historical data and to let the user interact with their own data frame between sessions. I decided to use CSV files as this type of file is the easiest to use. I have experience using CSV files and think they can be used for this task.

Command Line Interface (CLI)

Building a way to give users to interact with the program to add, delete, and manage habits is crucial. I have decided to use a simple input prompt interface to make this happen.

```
What is the habit you want to track? Workout
Describe your habit: aasd
What is you goal in days? 60
```

Figure 1: CLI of the app will be within Terminal through an input function

Organization of code

The code will be organized based on OOP and FP standards. Meaning that classes with methods will be created. I think that a two-class code is sufficient to provide an easy-to-read code but still compact. The 'HabitTracker' class includes all methods and attributes related to showing, editing, and managing the habits. All adding, deleting, and tracking methods will be stored there. The 'Analytics' class will include all analytics methods which will be displayed as a separate line of output in the terminal. Some analytics will be already shown in the main data frame, but I will explicitly program them as separate outputs as well.

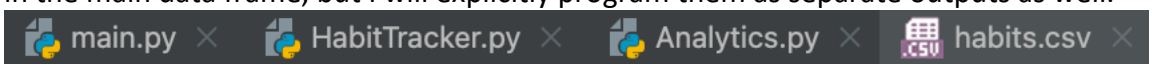

Figure 2: Screenshot PyCharm: 2 classes and main.py

How habits are tracked

Within the manage method, the user gets asked if they have completed the task for today. The algorithm will add '+1' to the current streak. If the habit is broken by non-checking the field, 'current streak' will be set to 0 and the historical record will be saved in 'longest streak'.

Modules & Tools

- Pandas
- Tabulate
- CSV/JSON
- Classes [methods/attributes]
- Docstrings

The main modules used in this program will be based on pandas, tabulate, and built-in module CSV/JSON. Pandas for easy analysis of the data frame, tabulate to display data frame more elegantly, and CSV/JSON to write and store files with the updated habit data frame.
Classes provide a means of bundling data and functionality together which will make the code easier to read for the client.

Organization: Flowchart

Figure 4 describes the entire concept in a visual way. All major tasks, functions, and relationships are displayed there.

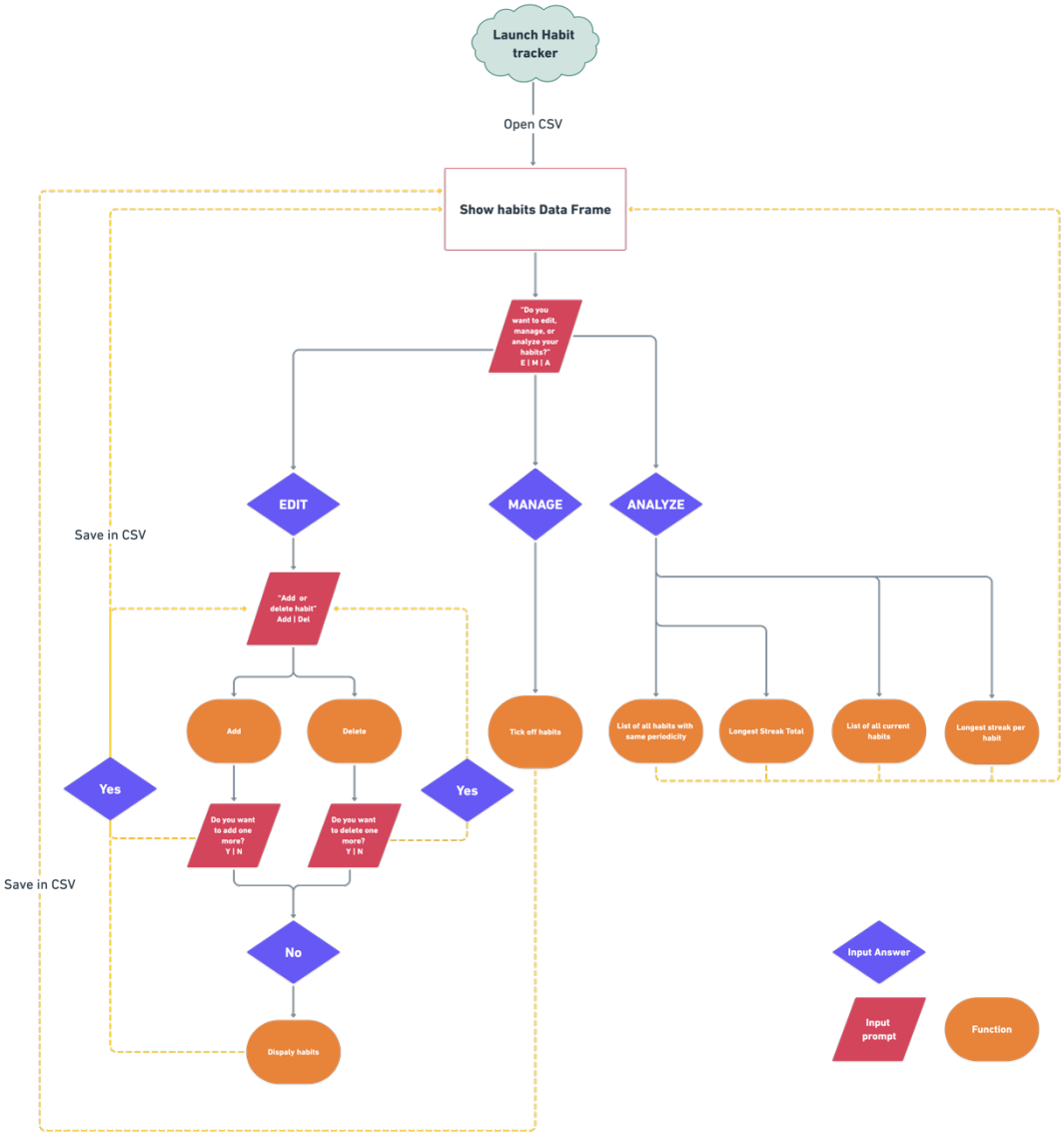

Figure 3 - Flowchart Habit Tracker App

Possible issues

There are two main issues that I see could unfold while programming the habit tracker.

1. Using a CSV may be a convenience for me, but JSON would allow me to store the habits in class which would make the adding, deleting, and managing aspect easier. I may have to reconsider that choice.
2. The input function could cause issues as there are better CLI modules out there. Again, it just is super convenient.