MP 2 - Productivity Application: Self Quarantine

Abhinav Singh

May 2020

1 Introduction

The self-tracking technology is getting famous nowadays from calorie tracking applications to heart rates and blood pressure tracking applications like Instant Heart Rate which uses the camera lens to track heart rate and blood pressure, and wearable as Fitbit, which perform the same functions. The Application is based on the principle of self-tracking or self-monitoring to understand his/her productivity in a given time. The main idea is to analyze the user data in a given amount of time to come up with a statistic or output that can help the user to improve, maintain or change his/her current state by utilizing the output made by the application or a system.

The data utilized in self-tracking is "Personal data" and according to Information Commission Office(ICO) it can be classified in various categories, but as we are building an application which would be operable on an operating system like Mac OS, so Personal Data is information pertaining to an individual which can be identified or identifiable, and this identity can be as simple as an I.P. address too.

2 Motivation

There are many self-tracking applications for windows and I own a Mac-book which makes it a little bit hard to find a plethora of self-tracking apps. I found some applications like self-spy, which offers analytics platform for windows, and they claim they have proper Mac-OS working model, but the last commit on their GitHub anything regarding Mac-OS was done 5-7 years ago. So I thought I should work on my self-tracking application on Mac-OS only.

As I am in quarantine right now, I named it **Self-Quarantine** which is built on Python and the application is exclusively for Mac-OS. The problem was what is productive, and how can track productivity? How an application would decide that if an application is productive or not, for example, a messenger application works differently in different situations, if you are talking to your friend or lecturer it may or may not be productive. So the user should be in command to

dictate what application is productive for him/her or which application is not. So on this premise i started building my application.

3 Approach

The Application is designed to run in a given amount of time and record user data on an operating system to check how productive the user is according to his own set of rules, the apps the user finds productive and that he/she is not.

Checks applications that is currently being worked on:
The Application is designed for macOS, It records for how long the user was active on a given window of the application. Rather than checking all the applications that are currently running in the background, there too many background processes that are being utilized by the system and it would be really time-consuming to classify all of these applications as productive or non-productive. Moreover, there are zombie processes also known as defunct processes, which are done with their execution, but still, their entries are remaining in the process table and similar to that there are orphan processes, the processes which lost their parent-process that started that process in the first place.

Time Constraint:
The application lets the user choose the total running time for the application to produce and analyse the data in the given time frame in order to come up with results which would help the user to interpret his/her productivity. The time in application can be chosen in seconds, minutes or hours. To optimise the system a little bit the updation and insertion in the database is done in every 3 seconds.

Keyboard-activity monitor:
The application monitors keyboard activity but doesn't record the keystrokes. The sole purpose of a keyboard-activity monitor is to analyse if a user is active in a given amount of time and User interaction with Graphical User Interface or Command-line Interface is being done or not. The keyboard activity monitor is not a key-logger as it is not saving any of the keystrokes. It only concludes that an activity is being productive or not on the basis of keys pressed in a given amount of time.

Mouse-activity monitor:
The application monitors mouse activity, but as same as keyboard-activity monitor no data about which key is pressed is being recorded. The purpose of mouse activity monitor is to support keyboard activity, monitor, in cases where the user is reading an article or working on an application that requires only mouse usage for example Paint, the mouse-activity-monitor would check the last coordinates of the mouse pointer to the current one, to check if any activity happens or not and then accordingly deem the application productive or nonproductive for a given span of time.

Choosing Productive or Non-Productive Applications:
The task of deciding on which application is productive or which application is not is subjective. As some applications like Zoom, which is a web conferencing application can be used for talking to a friend or attending a meeting, but still, the user has to decide if the meeting was productive or talking to a friend, maybe the user gain more insight about a topic just by having a conversation with a friend rather than questions answered in the meeting. Same goes for media applications in which the user might be watching a movie or a lecture, and it might be possible the user learned more from the movie. So this application lets the user decide which application is productive for him/her.

Uses a NoSql database:
In the initial phase of the application design the data was getting stored in form of a dictionary(Implementation of Hash-map in python), and the system was appropriately fast, but after a while the because the system was inundated by too many processes of mouse listening, keyboard listening, and continuously updating dictionary values, the operating system started slowing down and in an hour it crashed. The system was generating the data for a specific amount of time and as soon as the session ended it was generating a visual interpretation of productivity and data was not being saved, but the user was able to save the visualizations.

The problem of lost data was fixed by using MongoDB as a backend database service, and even if the system crashed the user was able to retain the previous data. The system failed after 1.39 hours while trying to record 4 hours of work.

4 Visualisation and Results

Plotly library is used for this purpose to show results in a browser, which reads data from MongoDB database.

Analysis of 1.39 hours:

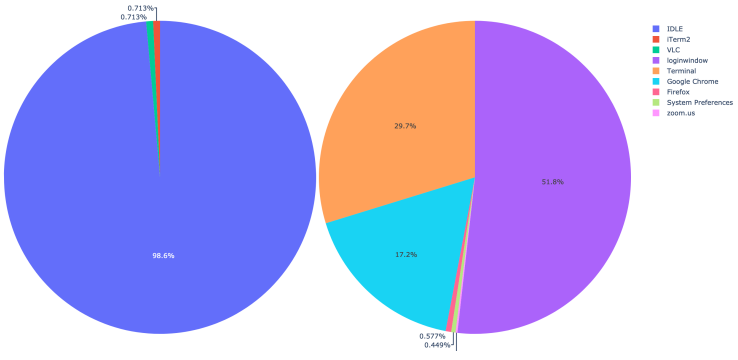

Figure 1:

Fig.1 is the result 1.39 hours of analysis, where the system crashed. Here the left figure is Time spent on productive applications and right figure is time spend on Non-productive applications.

Analysis of 10 min:
Figure2. signify the applications that user used or opened.

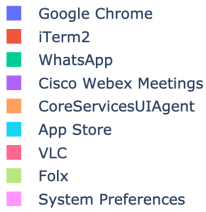

Figure 2:

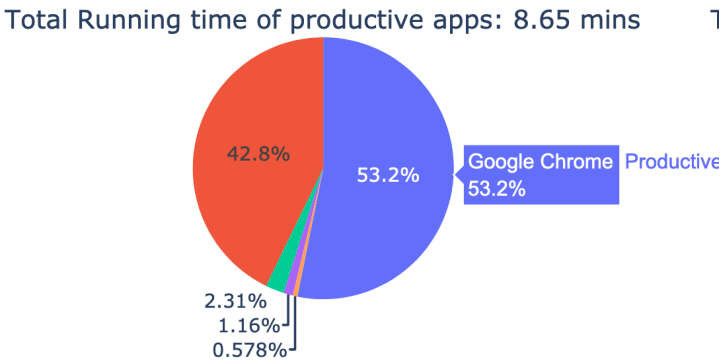

Figure 3:

Total Running time of productive applications:
In Figure 3. signify the total time spent in productive applications, and pie chart signifies the out of the time how much time spend on each application in percentage form.

Total Running time of Non-productive applications:
In Figure 4. signify the total time spent in Non-productive applications, and pie chart signifies the out of the time how much time spent on each application in percentage form.

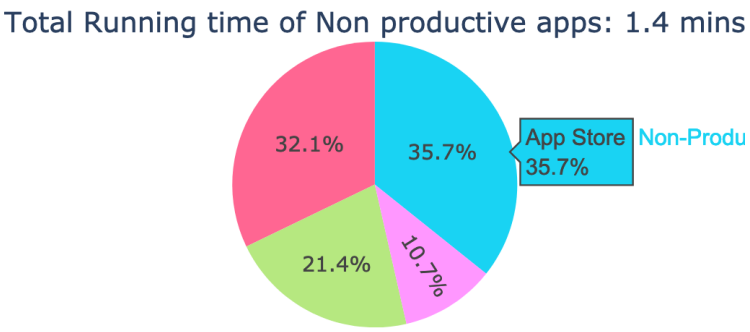

Figure 4:

Productivity of Productive applications:
In Figure 5. Signify the total time spent in Productive applications while being productive, which was calculated by mouse and keyboard listening, and pie chart signifies the of the time how much time spent on each productive application while being productive in percentage form.

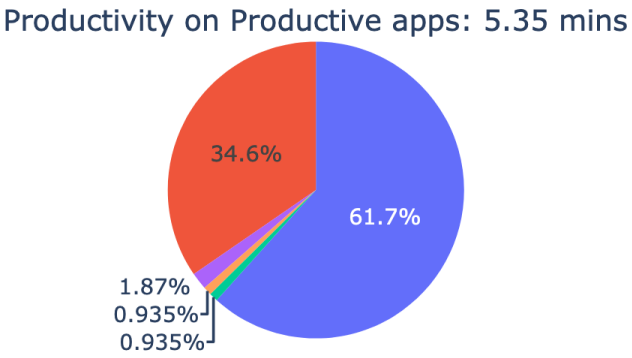

Figure 5:

Activity of Non-Productive applications: In Figure 6. Signify the total time spent on Non-Productive applications while showing some activity, which was calculated by mouse and keyboard listening, and pie chart signify the of the

time how much time spent on each Non-productive application while showing some activity in percentage form.

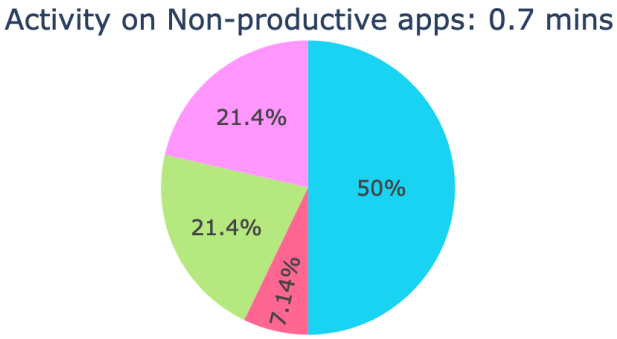

Figure 6:

5 How to run the program

for mounting the directory and running the code:
cd /Self_Quarantine/Self_Quarantine/
python3 mongoTest.py

let user choose if he/she wants to use old data or creating a collection/table for new session as shown in Figure 7.

let user choose if he/she wants to plot the existing data or create new ses-

Figure 7:

sion as shown in Figure 8.

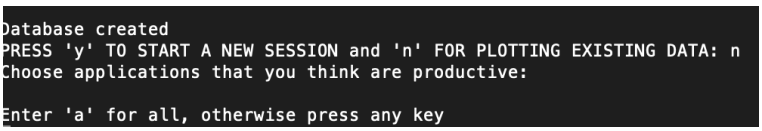

Figure 8:

Gives user to decide which app is productive and which is not, as shown in Figure 9.

```
Choose applications that you think are productive:

Enter 'a' for all, otherwise press any key
l
app name: Terminal
Enter 'a' for all, 'y' for yes and 'n' for no: y
Productive

app name: PyCharm
Enter 'a' for all, 'y' for yes and 'n' for no: y
Productive

app name: VLC
Enter 'a' for all, 'y' for yes and 'n' for no: n
Not Productive

app name: Music
Enter 'a' for all, 'y' for yes and 'n' for no: y
Productive

app name: Firefox
Enter 'a' for all, 'y' for yes and 'n' for no: y
Productive
```

Figure 9:

Application automatically shows result in the default browser as shown in Figure 10.

6 Limitations

This application is made on Python and for database it is using mongoDB. After a 10 minutes or so the operating system starts to slow down because of continuous insertion and updation in the database and processes like mouse and keyboard listener are running parallelly. So after a 1 hour the application doesn't crash itself, but user operating system does. Programming language which are low level like C can be better for this application or using libraries that are written in low level languages can be adequate too. Moreover, if the user wants to what was its data one month ago, user cannot find it, as a additional feature

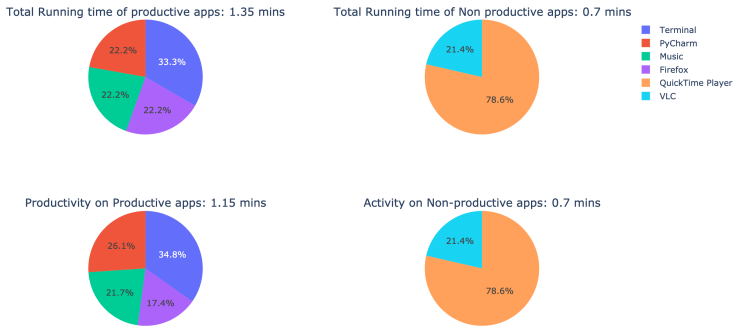

Figure 10:

like time stamp would be adequate rather than incrementing 3 seconds to the application dictionary value.

7 References

[1.]https://ico.org.uk/for-organisations/guide-to-data-protection/guide-to-the-general-data-protection-regulation-gdpr/key-definitions/what-is-personal-data/
[2.]https://en.wikipedia.org/wiki/Quantified_self
[3.]Koo, Sumin Fallon, Kristopher. (2018). Explorations of wearable technology for tracking self and others. Fashion and Textiles.5. 10.1186/s40691-017-0123-z.
[4.]https://en.wikipedia.org/wiki/Fitbit
[5.]https://www.azumio.com/s/instantheartrate/index.html
[6.]https://github.com/selfspy/selfspy
[7.]https://seleritysas.com/blog/2019/07/03/how-to-get-more-out-of-your-workforce-with-productivity-analytics/