PROJECT PLAN - TIME MANAGEMENT

1. Personal Information

- Title: Time Management
- Name: Nguyen Duc Tam
- Student Number: 906968
- Study Program: Digital Systems and Design
- Study Year and Date: 2021-2024

2. General description and difficulty level

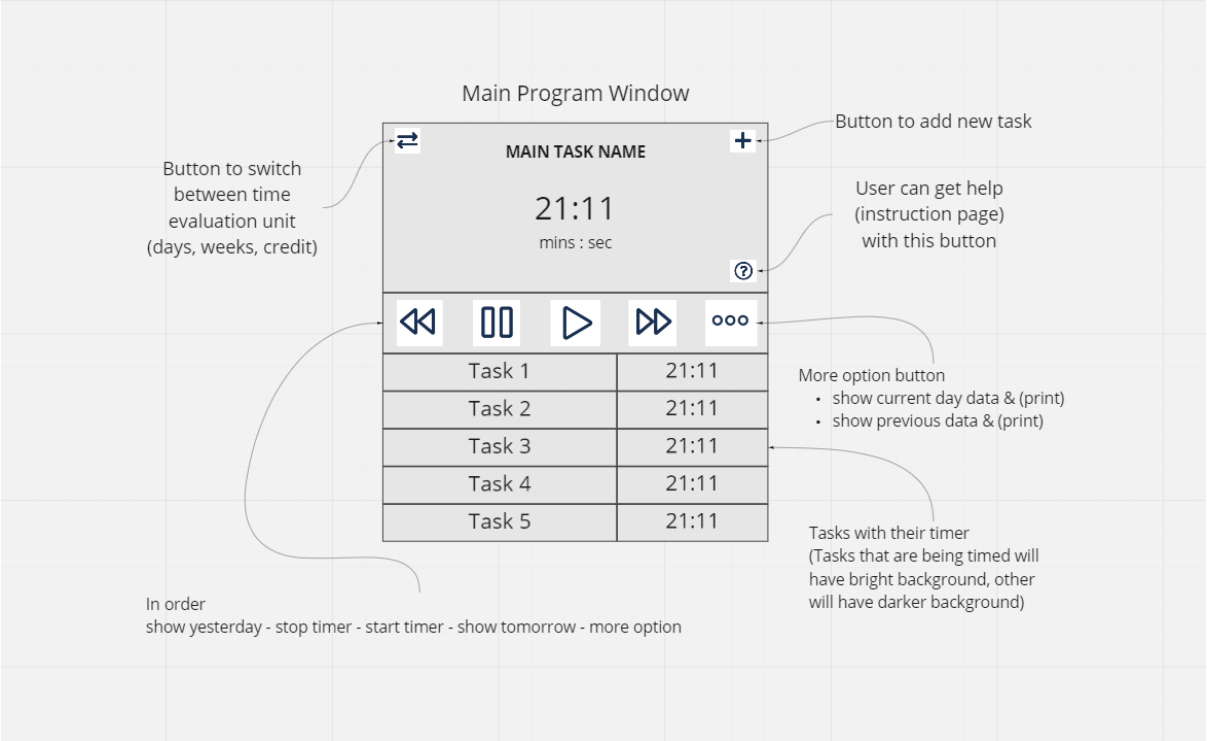
2.1. General Description

- Time Management is a program that allows the user to time their activities, with several functionality and flexibility to customize based on user's need and favor.
- Unique feature: Help function and Statistics for user data, Pomodoro Technique reminder

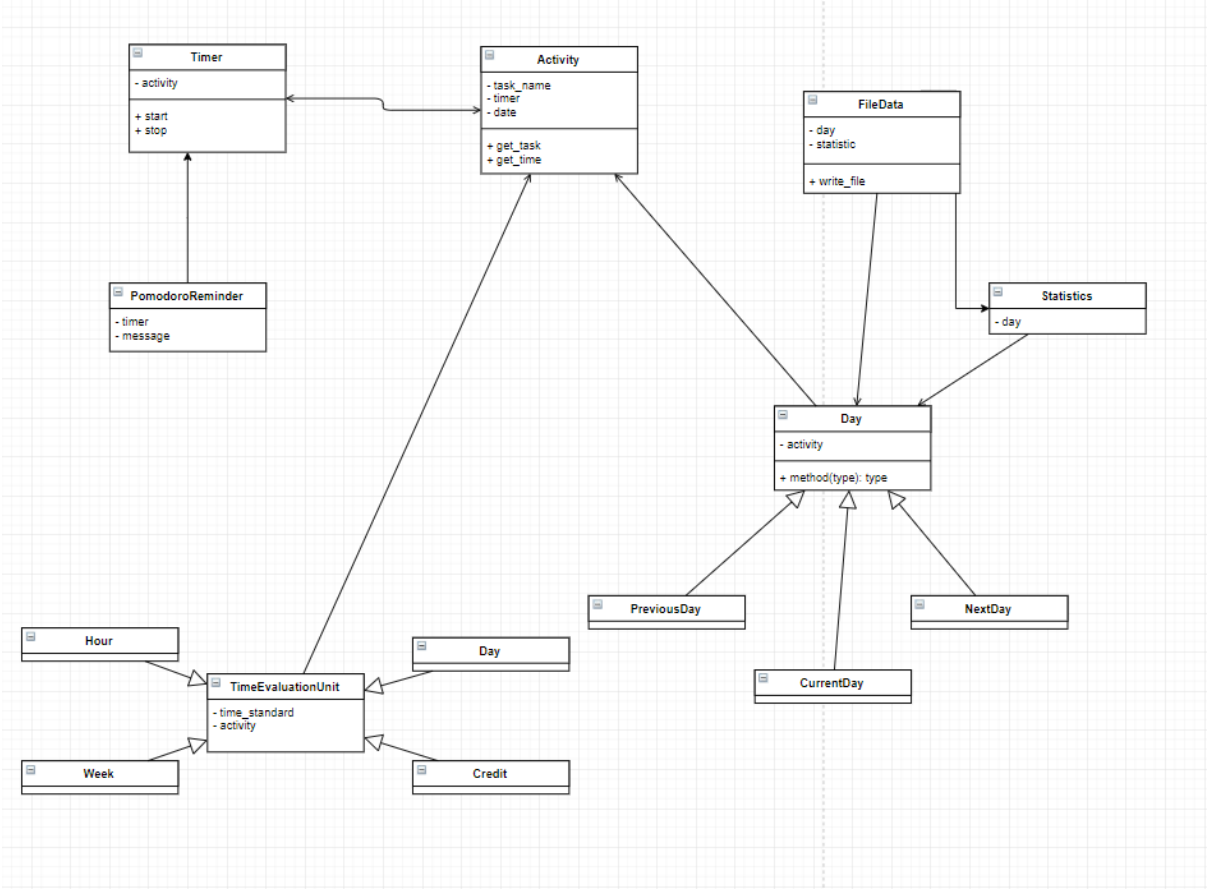
2.2. Difficulty Level - Hard

- Graphical interface
- Creation of an activity based on user input
- The amount of added activities can be maxed out at 5
- Taking time for activities
- The activity that is being timed should be distinct
- Multitasking, multiple courses can be timed at the same timer
- Picking a new activity to time interrupts and saves the previous activity's timer
- Timer's time is shown on screen
- Switching courses/replacing can be done with the graphical interface
- Print time management data of current day
- Print time management data from previous days from a file
- Saving time management data into a file
- Unit Tests for at least part of the program

3. Use case description and draft of the user interface



4. Program's structure plan



5. Data Structures

Python predefined data structures:

- List
- Dictionary

6. Files and file formats

- File will be used when the user wants to save the timer of the tasks.
- File will be loaded if the user wants to see the data from previous days.
- File formats:
    - CSV

7. Algorithms

- Basic converter in different time evaluation units (e.g., Mins - Hours - Days - Weeks - Credits).
- Some statistics math formula to sum up the timer data of the user in a day (e.g., so that the user can know how many percentages a single task takes up, total time spent, average time per task).

8. Testing plan

- UI test (the graphical interface is working correctly or not, i.e.,the 8 buttons in the picture in section 3)

- Different functionality test (e.g., does the start/stop function works well, the function saves timer data into file, the function reads input file, etc.)
- Max 5 activity can be added
- Unittest for at least one part of the program

9. Libraries and other tools

- PyQt5: to create graphical user interface
- datetime: to get information about the date
- numpy, pandas: to store the time of the tasks, and probably use for visualizing when user want to print the data
- scipy: to show some statistics about the time data of the user (total time, average, how much time does each task comprise)
- pyqtgraph: to create graph and chart to visualize data for user when printing

10. Schedule

- 25.02: return the plan
- 28.02: first meeting with TA, ask questions
- Probably around 10.03: start coding the program (I need to finish round 5 & 6 first to learn things, then I'll start coding the project)
- 25.03: checkpoint 1. Add the classes (objects) of the program
- 15.04: checkpoint 2. Add the Unittest for the "backend" (code) and start creating a graphical user interface
- 06.05 14:00: deadline. Upload all the code and the documentation on Gitlab before the deadline

11. Literature references and links

- Basics in Programming Y2 (Course material): https://plus.cs.aalto.fi/y2/2022/
- PyQt5 Documentation: https://doc.qt.io/
- datetime: https://docs.python.org/3/library/datetime.html
- numpy: https://numpy.org/
- pandas: https://pandas.pydata.org/
- scipy: https://docs.scipy.org/doc/scipy/reference/
- pyqtgraph: https://pyqtgraph.readthedocs.io/en/latest/