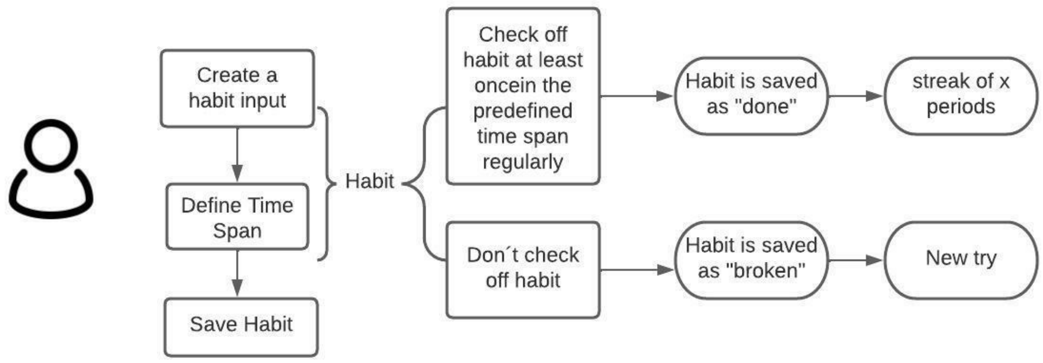
Conception Phase

As this is a very user oriented app which means that the user creates his own tasks and checks them off regularly but also has the ability to analyze them it is crucial to make it as user-friendly and simple to understand as possible.

The following visualization shows the app from the user´s perspective – in other words how the user proceeds when creating and checking off habits.



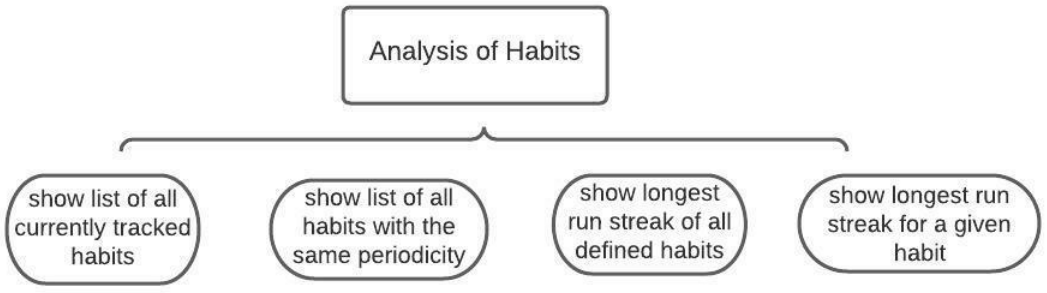
First of all, the user can decide between several option which are: Create a new habit, Choose a predefined habit, check a habit, Analyze habits or Exit. These choices are offered using the questionary library.

When creating a habit the user can type in a chosen sentence or word which is implemented with the ask( ) function also from the questionary library. When choosing the option "Choose a predefined habit" the user can choose between 5 different habits using the select( ) function by questionary library. The predefined habits are stored in a database in another file which is connected to the main.py file by using the import function. For both the user also has to type in a timespan which is implemented again using ask( ) function.

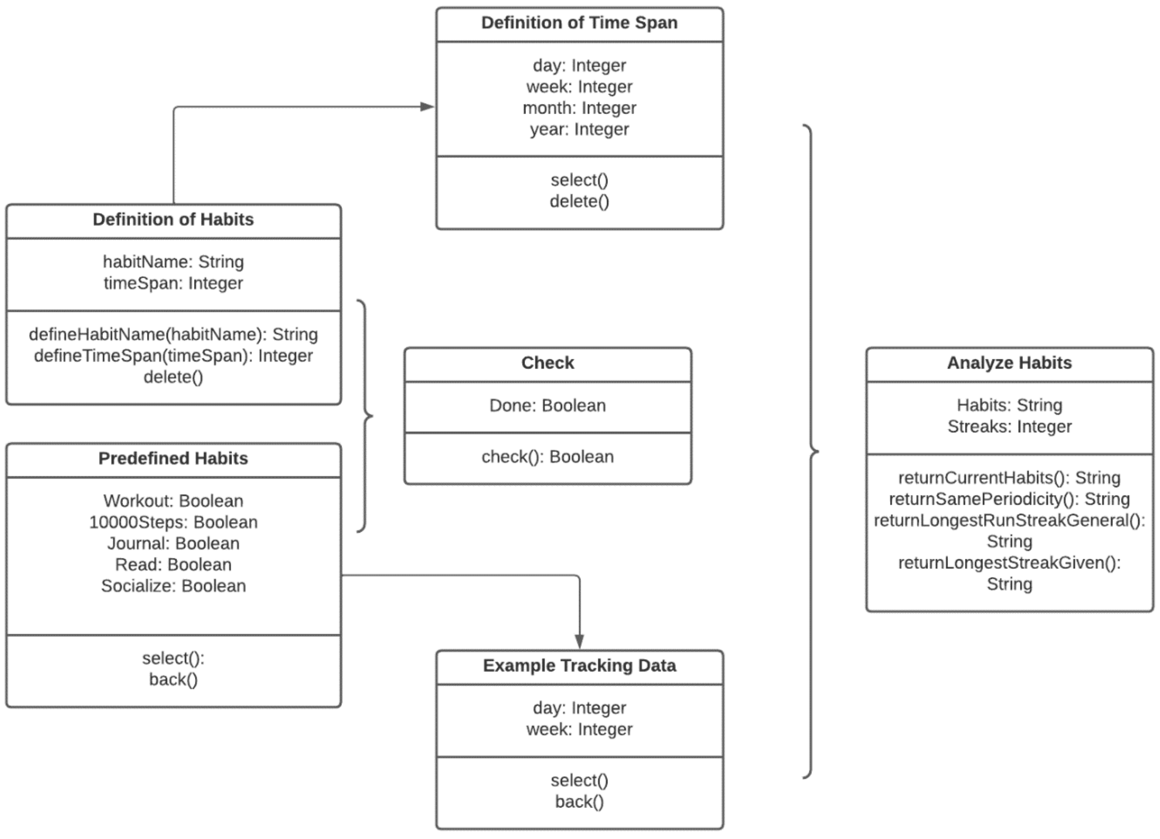
When choosing the "Check a habit" option the user has to choose a habit they defined/chose earlier to check it off using the select( ) function mentioned above. The checking off of the habits will be connected to a increment( ) function which will be stored in the file "check.py". This function will increment each time that the user chooses a habit as checked. If the user misses it, the function will turn back to 0 and start it over again.

Choosing the "analyze habits" option brings the user to another list of choices which each are connected to different function which are stored in the file called "analyze.py". Each of the functions returns the analyzed data. The function use the data from the database which is stored in the "db.py" file and connected via the import function to the "analyze.py" file.

Last option is "Exit". When the user chooses it the programs ends. The entire list of choices is covered into a loop that exists the program as soon as the user´s choice is "Exit".


Analysis of Habits
show list of all currently tracked habits
show list of all habits with the same periodicity
show longest run streak of all defined habits
show longest run streak for a given habit

To show the technical point of view it makes sense to use UML Diagrams where each functionality is divided into functions.



While coding the application one must not forget to regularly test it to check if everything runs correctly before getting too many issues. It is more time efficient to solve the errors while in process of coding. To test the classes for this project the pytest library will be used.

To separate testing a file named “test_project.py” will be created. In this file the check.py file as well as the db.py file are going to be imported. Inside the class TestHabitTracker a setup_method( ) function will be created which is run before the tests themselves and is used to create a database. Then, a new test database will be connected and some input will be written into different functions of the program to test them.

It is important to close the database after the tests and then delete it using it the teardown_method( ).

Other than questionary and pytest a few other requirements will be needed to implement the application. One of them is the datetime library which will be used to define the timespan for the habits. The other one is the sqlite3 library which will be needed for creating and working with databases and using SQL for that.