Habit tracker application (HTA) concept

Problem

To implement a new habit one needs to repeat it many times. But the same action releases less and less dopamine leading to depletion of motivation. To overcome this one needs to track emotional level and modify something in the habit to make it “new enough” and restore dopamine level. HTA allows record and edit habits, “check-offs” and get progress info to make timely decisions and ensure successful habit implementation.

Context view

User has three stories (replicated in the diagram below):

- Manage habits. Habit should be
  - registered,
  - archived if not pursued any more,
  - modified in description to make it “new enough”,
  - dropped completely if registered by mistake or has no valuable history
- Manage check-offs:
  - register new check-off
  - drop check-off registered by mistake
- Analyze history to see which habits are doing well and which need intervention.

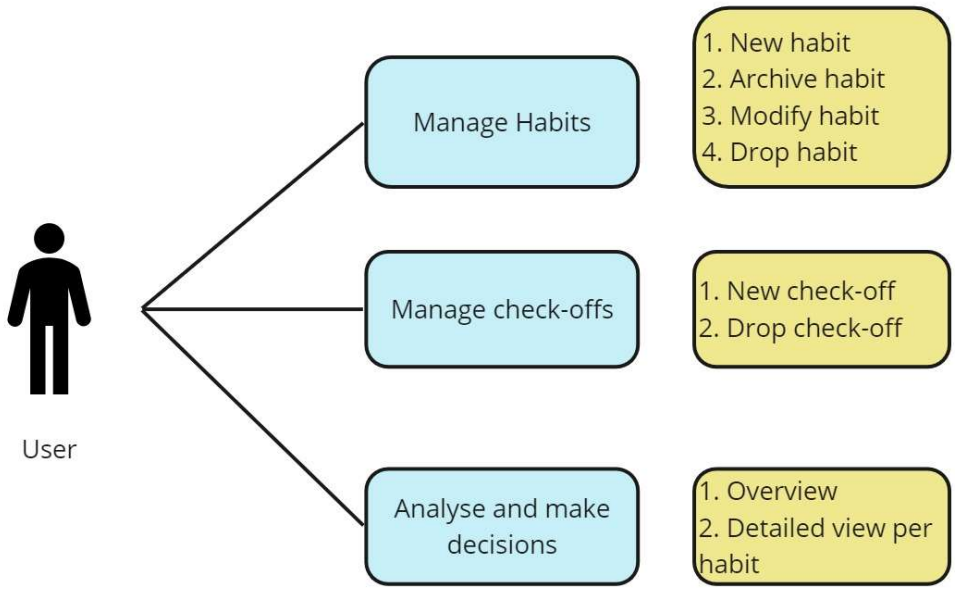

Figure 1: Context diagram

Logical view
We build two immutable data classes: Habit and CheckOff. They can be NamedTuples or @dataclass objects. Creating and saving new instances to a JSON file can be done directly, without loading their histories (collections).

But for many other manipulations we will need collections. And they should be mutable for modification and sorting. Hence we should build two more classes to manage each of two collections: HabitManager and CheckOffManager. As several methods (load, save and collection creation) will be the same, one class may inherit another.

Analytics block will be designed by functions without creating additional classes. User API - menus built with a standard *print* and *input* commands. Below is the UML class diagram.

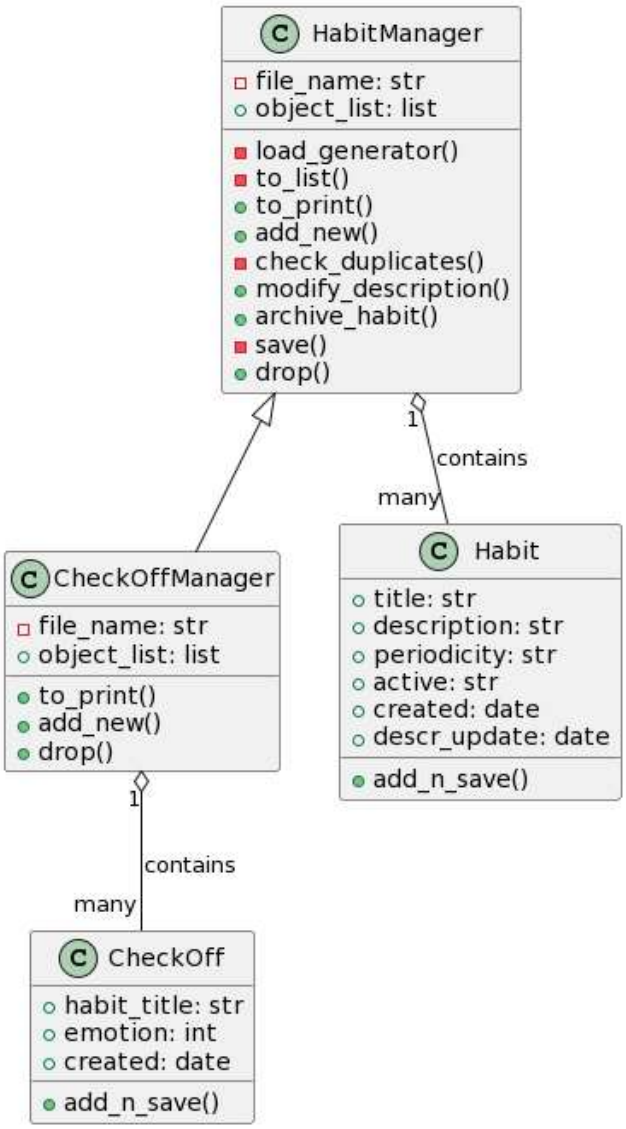

Figure 2: Class diagram.