Project Specs: 75 Hard Habit Tracker

Heather Lewin

HCI 5840

Fall 2024

General Description of the Project

This project will develop a habit tracking application specifically designed for users participating in the 75 Hard Challenge.

The challenge involves daily requirements over a 75 day time frame:

- two 45-minute workouts (one must be outdoors)
- drinking a gallon of water
- reading 10 pages of a non-fiction book
- following a diet with no cheat meals
- taking a progress photo

The app will help users track these daily tasks and visualize their progress over time, offering motivation and accountability.

The app will provide a streamlined interface for users to log their daily tasks, visually track their progress, and keep track of their streak over the entire challenge period.

Initially, the app will run as a command-line interface (CLI) where users can log their daily habits by responding to prompts. Eventually, it will evolve into a graphical user interface (GUI) for a more user-friendly experience. The GUI will be built using Tkinter.

External Mechanisms

- **Tkinter**: A Python library for creating desktop GUIs.
- **Database**: Initially, habit data will be stored locally in a CSV file. Later, this will be upgraded to a database solution (e.g., SQLite) for better scalability and data handling.

GUI

- The app will have a simple tkinter-based GUI to provide users with an intuitive visual interface
- Graphs will be added to visualize progress over time

Task Vignettes (User Activity Flow)

Task 1: Daily Habit Logging

When the user opens the app, they will see the task list for the 75 Hard habits. For each habit (e.g., exercise, water intake, diet adherence), the user will input whether the task has been completed for the day.

- **User Configurable Fields**:
    - Default: Completed status for each habit: Yes/No.

    - Configurable: Water tracking in increments; inclusion of picture; type of exercise; intensity of exercise; time spent; location of exercise; calories; protein; sugar; etc.
- **User Activity**:
    - The user opens the app and is prompted with the habit checklist and days in current streak completion
    - They mark each habit as completed; default is not completed. Add additional details if so configured
    - The app logs the data when user hits "save progress" button
- **Technical Details:**
    - Default input fields: checkbox for completed tasks
    - Configurable input fields: entry boxes for text, buttons for amount of water consumed
    - Data storage: Log daily data into a CSV file or database.

Task 2: Streak Tracking and Visualization

The app will show users a visual representation of their current streak for the 75 Hard challenge. The streak data will be calculated based on the user's progress logs and displayed either a number or in a calendar.

- **User Configurable Fields**:
    - Start date of the challenge.
    - Type of graph outputs they want
- **User Activity**:
    - The app displays the total number of days completed and the current streak.
    - If any days were missed, the streak resets, and the app informs the user.
    - Specific habits will have graphs or visualizations to aid users in understanding their progress, these will be configurable by the user. They can be used by clicking an icon in the habit bar
- **Technical Details:**
    - Default input fields: checkbox for completed tasks
    - Configurable input fields: entry boxes for text, buttons for amount of water consumed
    - Data storage: Log daily data into a CSV file or database.

Technical "Flow"

The app's data flow consists of three main blocks: user input, data storage, and output (display/notifications). The primary components will interact as follows:

1. **User Input**: The user provides daily habit data, inputs configured data, and views streaks or other visualizations.
2. **Data Storage**: User habit data will be saved to a CSV file or database.
3. **Output/Display**:
   - **GUI**: In the GUI version, data will be displayed using Tkinter widgets such as labels, buttons, progress bars, and calendars.
   - **Graphs:** Graphs and visualizations will be pulled from the database or CSV files, and displayed on screen

Data Structures:

- **User habit logs**: A CSV file or database table will store daily task data (date, task name, completion status, and any other configurable data).

Core Functions:

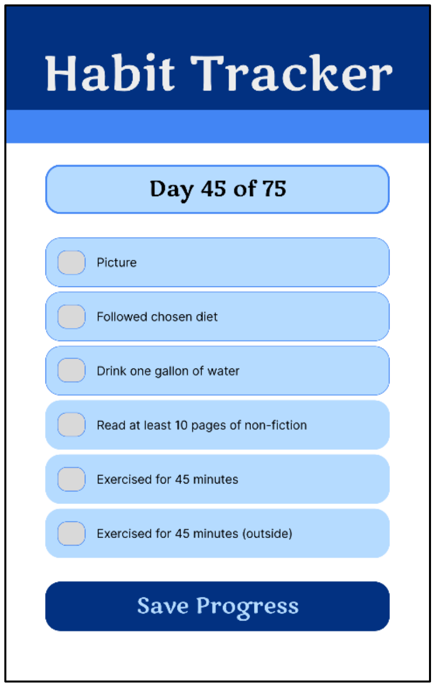
- log_habit_completion(): Collects and stores user data for the day's habits.
- calculate_streak(): Calculates the user's current streak based on the data.

Final (Self) Assessment

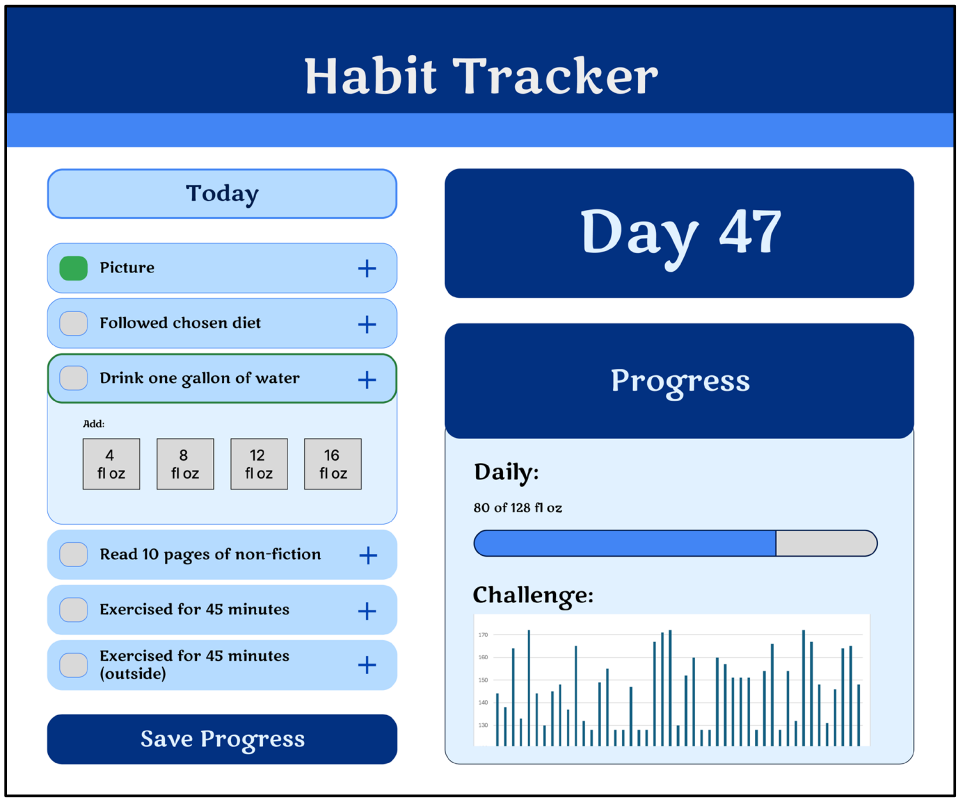
I feel confident that I can implement the default functions and data storage techniques as I've worked with these before. One challenge will be formatting the GUI using Tkinter and connecting the functions to the GUI, as I have less experience with that library. Displaying visualizations should be straightforward but I will add that as a secondary objective once everything else is functional. I think that the configurable options will result in the biggest challenges as they would add more interactions, data storage, and layout changes.

The most critical problem to solve will be building the GUI in a way that is both functional and user-friendly, and correctly syncing the data storage. I may need guidance on advanced Tkinter features to accomplish what I envision.

Default version of the Habit Tracker.


Habit Tracker
Day 45 of 75
Picture
Followed chosen diet
Drink one gallon of water
Read at least 10 pages of non-fiction
Exercised for 45 minutes
Exercised for 45 minutes (outside)
Save Progress

Tracker with User Configuration and Progress charts



GitHub Repository

hslewin/HabitTracker: hci5840 semester project (github.com)