Habit Tracker Project Abstract

GitHub Repository: https://github.com/topgun101/oofpp_habits_project

Project Overview

The Habit Tracker application was developed as a CLI-based tool to enable users to define, track, and analyze habits with various periodicities (daily and weekly). The application allows users to create and delete habits, log completions, and review habit performance over time. Using Python and SQLite, the application ensures data persistence while offering flexibility and simplicity for user data management.

Technical Approach

The project was developed using Python with a modular structure, ensuring separation of key functionalities:

1. **Habit Management (habit.py):** Habits can be added and deleted by the user. Each habit has a task specification and periodicity (daily or weekly). Example habits provide ideas of what can be tracked with the application and can be used to test functionalities.
2. **Habit Completion Tracking (completion.py):** Users can mark a habit as completed, which logs the completion date. This completion history is crucial for tracking habit streaks and identifying broken habits.
3. **Analytics Module (analytics.py):** An analytics module was implemented to enable users to retrieve current habits, list habits by periodicity, check for broken habits, and determine the longest streaks. Functional programming practices were used in the analytics module to ensure a clean and efficient structure.
4. **Database Integration (db.py):** SQLite was chosen to ensure persistent storage of habits and completions. Each habit is stored with a unique ID, name, description, and periodicity, while completions are timestamped events that associate with each habit.
5. **Command-Line Interface (CLI):** The Python library “click” was used to create a command-line interface to provide an intuitive CLI for users to interact with the app.
6. **Automated Testing (test_habit_tracker.py)**: By using “pytest”, unit tests can be run, that ensure consistent functionality across the application, including testing for duplicate habit prevention, broken habit detection, and longest streak calculation. To ensure reliable testing, a fixed set of test data was introduced using a function that simulates consistent daily and weekly habit completion data.

Achievements and Highlights

The modular design approach worked exceptionally well, allowing each feature to be developed and tested independently. Using SQLite for persistent data storage proved effective, enabling smooth data handling for working with the app and testing functions. Implementing the command-line interface with click provided a straightforward and user-friendly experience.

Challenges and Pitfalls

A significant challenge was ensuring data consistency, especially with randomized test data, which initially led to inconsistent test results. Addressing this required a shift to fixed test data, which added reliability but took extra time to implement. Additionally, the calculation of broken habits required careful handling of date comparisons, especially when daily and weekly habits had inconsistent completion records. This complexity initially led to misidentifying active habits as broken, necessitating a refined approach with consistent date logic.

Key Achievements and Features

- **Separation of Core Functionalities**: The app is divided into three main modules: habit management, completion tracking, and analytics. This separation allows each module to operate independently, making it easier to maintain and enhancing code readability.
- **Comprehensive Analytics Module**: The analytics module adds significant value by allowing users to quickly assess their progress, longest streaks, and identify areas that need improvement.
- **Robust Testing Environment**: With isolated test and production databases, the app's modular structure supports unit testing for each function independently. This structure reduces the chance of unintended interactions, ensures data integrity, and makes the application more reliable.

Conclusion

The Habit Tracker project delivered a robust, CLI-based habit tracking solution with comprehensive analytics, persistent storage, and reliable automated testing. Through modular design, clean analytics implementation, and a dedicated test database, the application is ready for further extension, including the potential for a GUI or external API integration. This project lays a solid foundation for ongoing development and demonstrates effective application of object-oriented and functional programming techniques to solve a real-world problem.