## Abstract

### Core Components

1. **InteractiveMenu**: This class handles the user interface, allowing users to navigate through the application using arrow keys and select options with the Enter key. It's called by the TUI.py for interacting with the user.
2. **HabitTrackerDB**: This class manages the SQLite3 db, which stores all the data related to users, habits, and completions. It has methods for creating, clearing, and filling the table with debug data.
3. **Analytics**: This class provides methods to retrieve and analyze data from the database, like getting all habits for a user, habits by periodicity, and calculating the longest streaks for habits.
4. **Completions**: This class manages the completion records for habits, adding, retrieving, updating, and deleting completions. It also includes a method to calculate the longest streak for a habit for later analysis.
5. **Habit**: This class handles the creation, retrieval, updating, and deletion of habits. It also provides methods to get habits by user and periodicity for later analysis.
6. **User**: This class manages adding, retrieving, updating, and deleting users.
7. **TUI**: This class actually handles calling the different functions and showing the results in a human-readable way. these are then fed into `main.py` to see. Makes heavy use of `InteractiveMenu` for interaction and `Colorama` for coloring outputs.

### What Went Well

- **User Interface**: Using `prompt_toolkit` for the TUI was a great idea, as we now no longed needed to type in numbers for running the program. The arrow-key navigation and color-coded options made the app easier to use for non-programming folks.
- **Database Management**: SQLite3 proved to be an excellent choice for the database engine. Its lightweight and serverless nature made it easy to integrate and manage without additional configuration.
- **Habit and User**: Implementing the classes as they are now was pretty painless, as we pretty much grabbed the methods from the `HabitTrackerDB` class and moved them to `Habit` and `User`.

### Challenges and Pitfalls

- **Database Schema Design**: Initially, the project started with a blob-based database, which caused trouble when trying to do Analytics and recalling data. I decided to handle part of the analytics process using SQL, so I had to migrate to a Table based approach.
- **User Interface Evolution**: The UI began as a number-based TUI and was later upgraded to use `prompt_toolkit` for a more interactive experience. This transition made the software more accessible but required a lot of refactoring
- **Database Operations Refactoring**: Originally, all database operations (SQL) were centralized in the `DataPersistence` class, making the `User`, `Habit`, and `Completions` classes mere boilerplate code. The SQL operations were later distributed across multiple classes, improving code organization and maintainability.
- **Error Handling**: Implementing robust error handling was a complex task. I went one by one through the TUI in order to try to break it, and a debug menu was added to facilitate faster debugging.
- **Analytics Implementation**: The analytics module underwent multiple iterations, initially using only SQL, then only Python, and finally a mix of both. This hybrid approach gave the best and most consistent performance.

### Value Adds

- **Interactive TUI**: Built using `prompt_toolkit`, it gives a more intuitive UX than using python's `input()` command for navigation.
- **Comprehensive Analytics**: The analytics module is fantastic for understanding the data in the database, and gives out pretty interesting results. The most useful function in my opinion is the `getLongestStreakAllHabits(self, user_id)`
- **Database Operations**: The HabitTrackerDB class does a great job managing the SQLite3 database. I'm particularly proud of the `fill_tables(self)` method, cause I managed to make a loop that simulates semi-realistic habit completion for filling the database with example data.

### Conclusion

This Habit Tracker is now robust and complete enough for full evaluation. It was a hard road to walk, but I've done it, and I hope it fulfills all the requeriments. The project can be accessed at cloned at https://github.com/alonsoburon/habit_tracker.git