B.SC. Data Science
Eric Anton Hauck
Matriculation Number 102305503
May 2025

Object Oriented and Functional Programming with Python

(DLBDSOOFPP01)
Task: Create a Habit Tracking App

ABSTRACT

Project goal

The goal of the course is to program a habit tracking application backend utilizing a command line interface (CLI) for user interaction. Further it is a goal to introduce students with the basic programming paradigms of object oriented and functional programming with python. During the project students also need to familiarize themselves with self-organized and project-oriented workflows. Likewise correct code and detailed instructions and comments are important.

Core features

The habit tracker allows users to create habits with customizable periodicities (daily, weekly, monthly or custom intervals). Users can mark habits as completed, track their progress with streaks, and analyze their habits using various metrics such as longest streaks and habits grouped by periodicity.
The app is built using Python 3.7+, with SQLite as the database for persistent data storage. It follows a modular, object-oriented design with classes for habit_manager, habits and db_manager. The command-line interface is implemented using questionary, providing an interactive user experience. Core functionalities are validated with pytest.

Requirements

The project meets all acceptance criteria, including the ability to create habits with daily and weekly periodicities, predefined habits with example data, and robust analytics. The CLI provides a clean and intuitive interface, and the app is fully documented with a detailed README and inline docstrings.

Process

Starting the project, I viewed the content in the video library on MyCampus and familiarized myself with the concept of object oriented programming. Beginning with phase 1 I created the conceptual framework of my habit tracker and after its review I quickly started programming. I used the example on MyCampus as a rough basis for my project and started building the required functionality on top. During the programming process I came to the realization that I needed to differ on some points from my original concept, as I found better solutions to my initial ideas. I additionally decided to add another class called db_manager which handles the database connection and can be passed into the other classes habit and habit_manager via dependency injection. This allowed me to have a test dataset that can be loaded into the program for real testing. Also I decided it would be best to store the period as integers instead of strings as

initially described in my concept. This allows easier calculations and also enabled me to add the feature of custom periods. Important analytics like current_streak and longest_streak are in difference to my initial concept, written in the database and updated upon starting the program. This way I would not need to write complex analytics functions that looked at the whole completions table and calculate these every time anew. During the process some additional design/definition questions needed to be addressed. What happens if a user completes a habit more than once in a period? I choose to not add additional completions to the streak. This is due to my definition of a period cycle. If the period is a week starting on Monday and the user completes the habit, the next period starts on the next Monday and then the user has again a week time to complete the habit.
After programming the functionality of the habit tracker, I started to build the CLI. I needed to add some additional helper functions to e.g. convert the period into a more readable text “daily” and “weekly” or functions to validate the users input. During this time I also did some optimizations on the functions when needed.
As a last step I commented my code and created the README and requirements.

Challenges and Achievements

My biggest struggle was the way to approach the project. I have some background in programming in general but building an application was something new. On top of that there was alot new input to unpack. Like building a CLI, Git, database design and object oriented thinking. I tried tackling that by just starting a sandbox program and trying different things out. When I felt confident, I started to build the habit tracker. In retro perspective I feel like this was the right approach for me. I have gradually built up more knowledge and confidence in programming with python. In the beginning I felt like I needed to find “the solution” to the habit tracker but I began to realize that there are many and I can design the solution. I think I have built a robust and simple habit tracker and am especially proud of my progress. What went not so well? I underestimated the value of planning the production of code. Sometimes I sat down and didn’t know where to continue from yesterday. Then I started to write a to-do list with various importance levels that I could revisit. This greatly improved my production and organization. Also I sparsely commented the code during production and added detailed explanation later. I think that on this approach I can improve.