Course Name: Object Oriented and Functional Programming with Python

Course Code: DLBDSOOFPP01

Student Name: Mohamed Haizoun

Matriculation No.: 376364

ABSTRACT

Habit Tracker Application

Introduction

A straightforward program to help folks keep tabs on daily or weekly routines is the habit tracker. In today's busy world, sticking to good practices can be tough. My habit tracker aims to fix that by giving a simple setup for handling habits. Built in Python, it uses object-oriented ideas for the main parts, functional methods for reviews, and a command line setup for ease. The tracker focuses on being easy to handle, letting users add, mark, and check their habits without fuss. Its key aim is to build better habits and boost daily output. By showing progress clearly, the tracker pushes users to stay on track and reach their aims. Beyond just output, it aids in better life changes and time use for anyone looking to build new routines.

Technical Approach

Python served as the base for the habit tracker, known for its clear code and quick build times. For keeping data, options include SQLite for structured saves or JSON for simpler files. The setup follows object-oriented rules to keep code neat and reusable. Main files include **models.py** for the Habit class, **analytics.py** for review functions, **persistence.py** for saving, and **cli.py** for user talks.

The Habit class in **models.py** holds details like name, cycle (daily or weekly), start date, and mark times. It has ways to add marks without repeats, figure current runs, and longest runs. For example, it checks same days or weeks to avoid extras.

Analytics.py has clean functions that don't change data, like finding top runs across habits or finish rates by cycle. These take habit lists and return maps or numbers.

Persistence.py offers a base for saving, with SQLite or JSON choices to add, get, or change habits.

Cli.py uses Rich for nice menus, letting users add habits, mark them, remove, or see stats.

Sample_data.py makes test habits with real-looking marks over four weeks from September 7, 2025, to October 5, 2025. This helps show how it works.

In action, users run the program for a menu to handle habits and view details like runs or rates.

Challenges and Pitfalls

Building went okay in spots. The object setup for Habit worked well, making adds like run counts simple without mess. Functional reviews were easy to check since they just use inputs.

But making test data real was hard. First tries gave odd patterns from fixed rules, like day mods. Needed changes for better odds, 80 percent workdays and 66 off days for daily.

Date rules for cycles surprised me. Weekly checks for same or next weeks needed exact math on Mondays, causing early count mistakes. Stopping repeat marks had issues with times, fixed by using dates and ordering.

Command line tools like Rich had small setup hitches, but made it nicer without big problems.

Looking back, some issues stretched time, but taught a lot on planning and fixes.

Achievements

Several parts stand out that add real help.

Run Tracking: It figures ongoing and top runs past marks, showing gains. With tests, daily like Brush teeth got ongoing 5 and top 5, weekly 4 each.

Stats Views: Gives finish rates by cycle, like 79 percent daily over 29 days and 80 percent weekly, plus full views to spot strong or weak spots.

Saving Choices: Switch between SQLite or JSON keeps data safe over runs.

Menu Setup: Nice tables and choices make it easy and fun, better than plain.

Test Setup: Ready habits with likely marks show use fast, with 23 marks per daily and 4 weekly. These make it a solid tool for habit handling.

Conclusion
Overall, building the habit tracker was a good practice that showed Python's use in real tasks,

covering models, reviews, saves, and user sides. It meets a need in fast lives by aiding routine tracks. With flexible cycles, clear stats, and easy use, it helps growth and better time. Though challenges came up, the end result shows problem fixes and steady work. Lessons on dates, data, and interfaces improved my skills. The tracker's success comes from planned fixes and useful adds.

GitHub Link: **https://github.com/Nemesis-hub/Habit-Tracker-App.git**