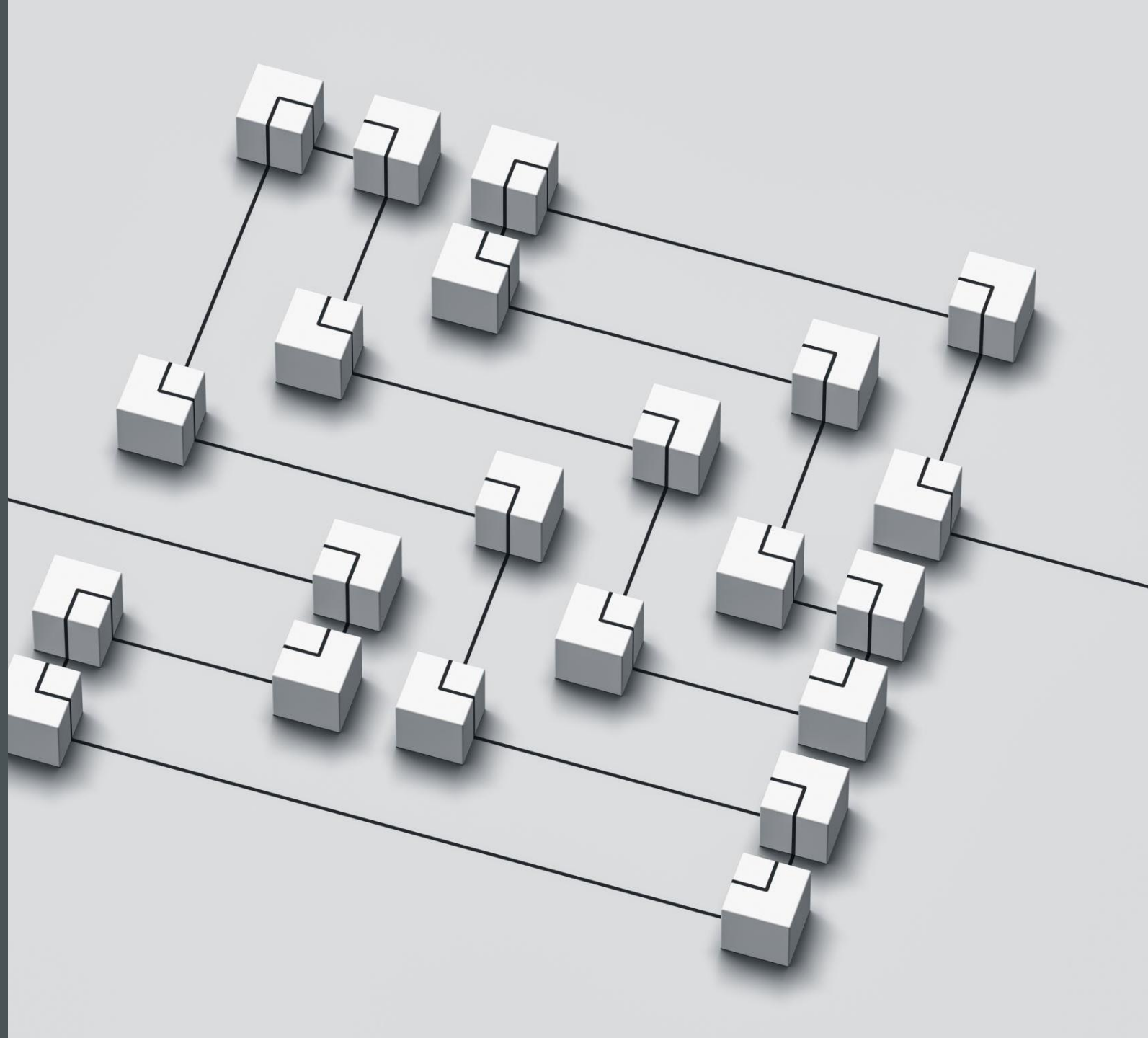
TASK 1: CREATE A HABIT TRACKING APP DLBDSOOFPP01 OBJECT ORIENTED AND FUNCTIONAL PROGRAMMING WITH PYTHON

Ayaulym Myrzatay

Github: https://github.com/ayaaiden/Ayaulym_Myrzatay_92003661_DLBDSOOFPP01_Phase_2



TOOLS AND TECHNOLOGIES USED IN THE PROJECT

Python 3.10

Built in Json module

Object – oriented programming

Functional Programming (FP)

Unit testing with the *unittest*

ARCHITECTURE OF THE PROJECT

Habit class: it represents a single habit (by the name, frequency, and completions)

HabitTracker class: it manages multilple habits

Main.py: CLI interface and logic control

Test_main.py: Unit testing for the core functions

KEY FEATURES OF THE PROJECT

Adding and listing of the habits (for example: daily or weekly)

Mark the habit as “done”

Filtering habits by the frequency

Track the completions timestamps

Persistent storage in the habits.json

FUCTIONAL PROGRAMMING

1. For the functional programming I have used map, filter, and lambda in

1.1. Filtering by the frequency

1.2. Mapping the habits by the names

2. It ensured the side – effect– free functions for the analytics

CLI FLOW

- Example of the Habit: Journaling
- Example of the Frequency: weekly

- After the completion marked and saved into the JSON

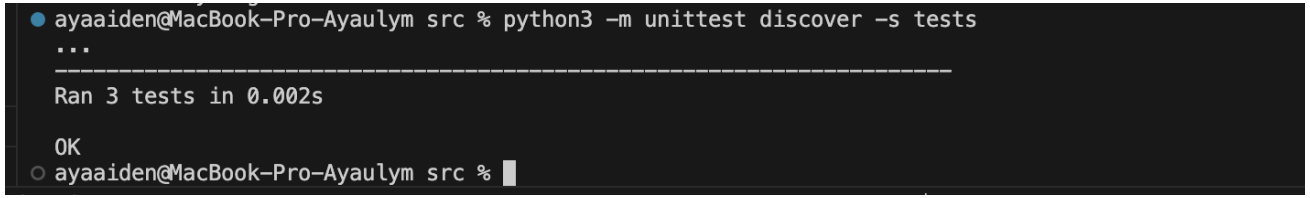
```
 Habit Tracker Menu
1. View all Habits
2. Add a New Habit
3. Mark Habit as Done
4. Filter Habits by frequency
5. Exit
```

```
{
    "name": "Journaling",
    "frequency": "weekly",
    "completions": []
}
```

TESTING AND VALIDATION IN THE PROJECT

- Unit tests:
- Creating the habit
- Adding and finding habits
- Marking them as “done”

Tests are located in the test/test_main.py
Which runs via python3 –m unittest discover –s tests



```
 Habit Tracker Menu
1. View all Habits
2. Add a New Habit
3. Mark Habit as Done
4. Filter Habits by frequency
5. Exit
 Enter the choice from 1 to 5:3
Select the habit to mark down:
1. Journaling
2. Running
Enter the habit number: 1
Marked 'Journaling ' as done.

 Habit Tracker Menu
1. View all Habits
2. Add a New Habit
3. Mark Habit as Done
4. Filter Habits by frequency
5. Exit
 Enter the choice from 1 to 5:3
Select the habit to mark down:
1. Journaling
2. Running
Enter the habit number: 2
Marked 'Running ' as done.

 Habit Tracker Menu
1. View all Habits
2. Add a New Habit
3. Mark Habit as Done
4. Filter Habits by frequency
5. Exit
 Enter the choice from 1 to 5:3
Select the habit to mark down:
1. Journaling
2. Running
Enter the habit number: 1
Marked 'Journaling ' as done.

 Habit Tracker Menu
1. View all Habits
2. Add a New Habit
3. Mark Habit as Done
4. Filter Habits by frequency
5. Exit
 Enter the choice from 1 to 5:
```

```
ayaaiden@MacBook-Pro-Ayaulym Python_project  % cd habit_tracker_project
ayaaiden@MacBook-Pro-Ayaulym habit_tracker_project % ls
docs    src
ayaaiden@MacBook-Pro-Ayaulym habit_tracker_project % cd src
ayaaiden@MacBook-Pro-Ayaulym src % python3 main.py

 Habit Tracker Menu
1. View all Habits
2. Add a New Habit
3. Mark Habit as Done
4. Filter Habits by frequency
5. Exit
 Enter the choice from 1 to 5:1
No habits found.

 Habit Tracker Menu
1. View all Habits
2. Add a New Habit
3. Mark Habit as Done
4. Filter Habits by frequency
5. Exit
 Enter the choice from 1 to 5:2
Enter the habit name: Journaling
Enter the frequency(f.e., daily, weekly): daily

 Habit Tracker Menu
1. View all Habits
2. Add a New Habit
3. Mark Habit as Done
4. Filter Habits by frequency
5. Exit
 Enter the choice from 1 to 5:2
Enter the habit name: Running
Enter the frequency(f.e., daily, weekly): weekly

 Habit Tracker Menu
1. View all Habits
2. Add a New Habit
3. Mark Habit as Done
4. Filter Habits by frequency
5. Exit
 Enter the choice from 1 to 5:1
Habit(name = Journaling , frequency = daily , completed =0 times)
Habit(name = Running , frequency = weekly, completed =0 times)

 Habit Tracker Menu
1. View all Habits
2. Add a New Habit
3. Mark Habit as Done
4. Filter Habits by frequency
5. Exit
 Enter the choice from 1 to 5:
```

SCREENSHOTS OF THE PROGRAMMING

WHAT I HAVE LEARNED & PRACTICED

Object - Oriented Programming (OOP)

File I/O and JSON serialization

Date/time handling with datetime

Exception handling

Unit testing with Python's unittest

CLI design and user input validation