92133053 – ORIBHUNUEBHO, HAPPINESS

OOFPP

DEVELOPMENT PHASE

PROJECT OVERVIEW

The created application is a CLI based backend that handles multiple features for habit tracking:

- Create habits
- Read/list habits
- Update habits
- Delete habits
- Analyze habit streaks

All features are implemented using Object-oriented and functional programing in Python v3.13 with a database as data storage mechanism.

```
uv run src\main.py

=========================================================
            🎯 HABIT TRACKER - Build Better Habits 🎯
=========================================================
Type 'help' to see available commands


> help


    |                    AVAILABLE COMMANDS                    |
    |                                                          |
    |  create      - Create a new habit                        |
    |  list        - List all habits                           |
    |  complete    - Mark a habit as completed                 |
    |  delete      - Delete a habit                            |
    |  update      - Update habit details                      |
    |  analyze     - View habit analytics                      |
    |  summary     - Display overall statistics                |
    |  help        - Show this help menu                       |
    |  exit        - Exit the application                      |

>
```
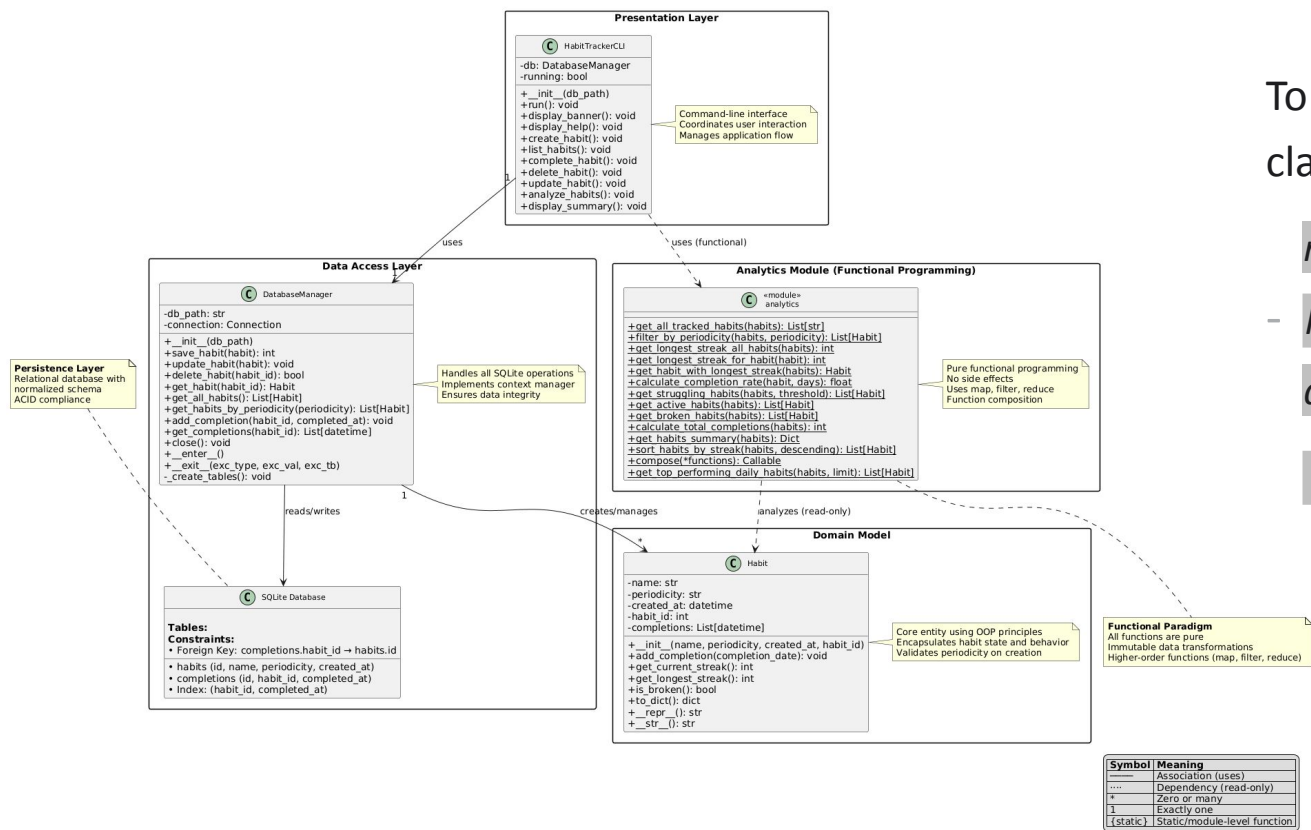
IU INTERNATIONAL UNIVERSITY OF APPLIED SCIENCES

To follow the object-oriented principle, I implemented multiple classes which are represented by the following files:

- *main.py* – used for CLI interaction
- *habit.py* – defines the Habit class
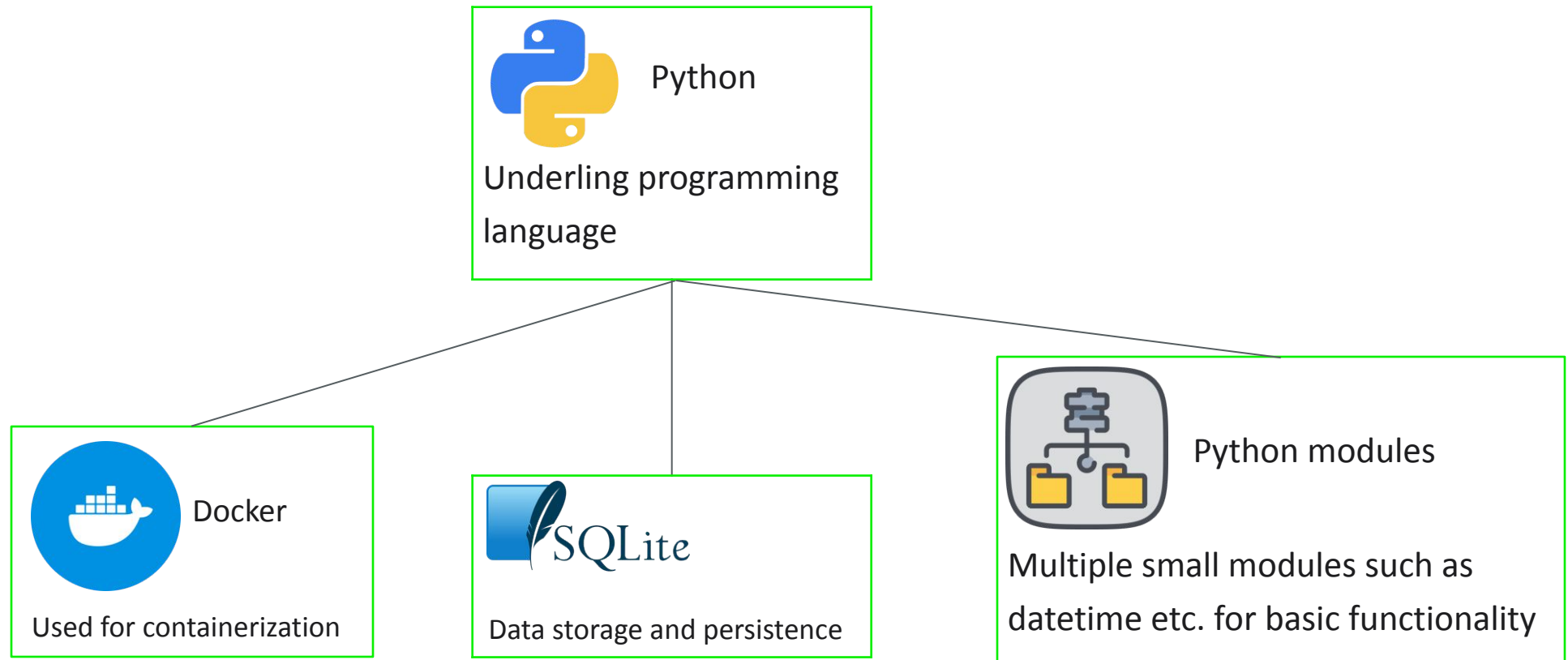- *analytics.py* – used for the analytics functions
- *database.py* – enables database integration

Presentation Layer

HabitTrackerCLI

-db: DatabaseManager
-running: bool

+__init__(db_path)
+run(): void
+display_banner(): void
+display_help(): void
+create_habit(): void
+list_habits(): void
+complete_habit(): void
+delete_habit(): void
+update_habit(): void
+analyze_habits(): void
+display_summary(): void

Command-line interface
Coordinates user interaction
Manages application flow

uses / uses (functional)

Data Access Layer

DatabaseManager

-db_path: str
-connection: Connection

+__init__(db_path)
+save_habit(habit): int
+update_habit(habit): void
+delete_habit(habit_id): bool
+get_habit(habit_id): Habit
+get_all_habits(): List[Habit]
+get_habits_by_periodicity(periodicity): List[Habit]
+add_completion(habit_id, completed_at): void
+get_completions(habit_id): List[datetime]
+close(): void
+__enter__()
+__exit__(exc_type, exc_val, exc_tb)
-_create_tables(): void

Handles all SQLite operations
Implements context manager
Ensures data integrity

Persistence Layer
Relational database with normalized schema
ACID compliance

SQLite Database

Tables:
Constraints:
• Foreign Key: completions.habit_id → habits.id

• habits (id, name, periodicity, created_at)
• completions (id, habit_id, completed_at)
• Index: (habit_id, completed_at)

reads/writes

creates/manages

Analytics Module (Functional Programming)

«module»
analytics

+get_all_tracked_habits(habits): List[str]
+filter_by_periodicity(habits, periodicity): List[Habit]
+get_longest_streak_all_habits(habits): int
+get_longest_streak_for_habit(habit): int
+get_habit_with_longest_streak(habits): Habit
+calculate_completion_rate(habit, days): float
+get_struggling_habits(habits, threshold): List[Habit]
+get_active_habits(habits): List[Habit]
+get_broken_habits(habits): List[Habit]
+calculate_total_completions(habits): int
+get_habits_summary(habits): Dict
+sort_habits_by_streak(habits, descending): List[Habit]
+compose(*functions): Callable
+get_top_performing_daily_habits(habits, limit): List[Habit]

Pure functional programming
No side effects
Uses map, filter, reduce
Function composition

analyzes (read-only)

Functional Paradigm
All functions are pure
Immutable data transformations
Higher-order functions (map, filter, reduce)

Domain Model

Habit

-name: str
-periodicity: str
-created_at: datetime
-habit_id: int
-completions: List[datetime]

+__init__(name, periodicity, created_at, habit_id)
+add_completion(completion_date): void
+get_current_streak(): int
+get_longest_streak(): int
+is_broken(): bool
+to_dict(): dict
+__repr__(): str
+__str__(): str

Core entity using OOP principles
Encapsulates habit state and behavior
Validates periodicity on creation

Symbol	Meaning
———	Association (uses)
····	Dependency (read-only)
*	Zero or many
1	Exactly one
{static}	Static/module-level function

Separation of concerns enables independent development and testing of each component.

TOOLS & FRAMEWORKS

Multiple tools and frameworks were used which allowed a quick and efficient design and implementation:

USER INTERACTION - BASICS

Start and help menu

List, create, and mark habit(s)

List and delete habit(s)

✦ Every interaction (create, delete, edit) can be done with either the initial letter or the whole word.

USER INTERACTION - ANALYTICS

Show analytics of habits

Shows the longest streak of all habits

Lists each habit by its corresponding periodicity

View all habits with each current streak

Show the streak for a defined habit

View top 5 performing habits

Currently, there is only one habit

```
> analyze

📊 Habit Analytics
--------------------------------------------------------------
1. List all tracked habits
2. List habits by periodicity (daily/weekly)
3. Show longest streak across all habits
4. Show longest streak for specific habit
5. Show struggling habits (completion rate < 50%)
6. Show top performers
7. Back to main menu

Select option (1-7): 1

📋 All tracked habits (1):
   1. Drink 1 gallon of water

> analyzw
❌ Unknown command: 'analyzw'. Type 'help' for available commands.

> analyze

📊 Habit Analytics
--------------------------------------------------------------
1. List all tracked habits
2. List habits by periodicity (daily/weekly)
3. Show longest streak across all habits
4. Show longest streak for specific habit
5. Show struggling habits (completion rate < 50%)
6. Show top performers
7. Back to main menu

Select option (1-7): 6

🌟 Top 5 performers:
   1. Drink 1 gallon of water - 0 daily streak
```

APPENDIX

- The software is licensed under the MIT license which allows everyone with a copy to modify, merge, publish, distribute, etc. copies of the software.

- The source code can be found in the public GitHub repository:

  https://github.com/herthiqal/OOFPP_Habits

- Documentation is available as a "README" and comments in the code itself