Habit Tracker

DEVELOPMENT & REFLECTION

Mohamed Haizoun

Scope & Acceptance Fit

- Create, track, and analyze daily/weekly habits with duplicate-safe check-offs & streaks.
- CLI actions: create, list, checkoff, analytics with clear, friendly output.
- Persistence: SQLite (default) + JSON (portable), full CRUD, schema + index.
- Analytics (pure functions): longest/current streaks, inactive habits, completion rates.
- Deliverable matches brief: customer-facing PDF (5–10 slides) with visuals & tool choices.

Design & Implementation

- I IMPLEMENTED THE CORE DOMAIN MODEL: HABIT (ID, NAME, PERIODICITY, CREATED_AT, CHECK_OFFS) AND PERIODICITY (DAILY/WEEKLY). THE MODEL BLOCKS DUPLICATE CHECK-OFFS PER DAY/WEEK AND COMPUTES CURRENT AND LONGEST STREAKS.

- I BUILT DUAL PERSISTENCE: SQLITE (DEFAULT) WITH TABLES HABITS AND CHECK_OFFS PLUS AN INDEX, AND JSON (PORTABLE) WITH SAFE FILE INITIALIZATION. FULL CRUD AND A GUARDED ADD_CHECK_OFF PATH ENSURE INTEGRITY.

- I IMPLEMENTED AN ANALYTICS MODULE AS PURE FUNCTIONS: LIST ALL HABITS/BY PERIODICITY, LONGEST STREAK OVERALL, CURRENT STREAKS, INACTIVITY DETECTION, COMPLETION RATES, AND SUMMARY STATS.

- I EXPOSED A CLI WITH CLEAR, VISUAL OUTPUT (CLICK + RICH): COMMANDS CREATE, DELETE, CHECKOFF, LIST, AND ANALYTICS, INCLUDING SUCCESS PANELS AND TABLES.

- I ADDED REALISTIC SAMPLE DATA: 5 PREDEFINED HABITS (3 DAILY, 2 WEEKLY) WITH ~4 WEEKS OF CHECK-OFFS AND A HELPER TO POPULATE A REPOSITORY.

Core Flows

• I CREATE A HABIT BY ENTERING A NAME AND CHOOSING DAILY OR WEEKLY; THE APP VALIDATES, GENERATES AN ID, PERSISTS IT, AND RECORDS THE CREATION TIME.

• I CHECK OFF A HABIT AT A SPECIFIC DATE/TIME; DUPLICATES IN THE SAME DAY/WEEK ARE BLOCKED, AND STREAKS UPDATE IMMEDIATELY.

• I LIST AND FILTER HABITS (ALL, DAILY, WEEKLY) AND SEE CURRENT/BEST STREAKS AND TOTAL COMPLETIONS FOR EACH.

• I ANALYZE PROGRESS WITH REQUIRED FUNCTIONS: ALL HABITS, BY PERIODICITY, LONGEST STREAK OVERALL, AND LONGEST STREAK FOR A GIVEN HABIT (SHOWN AS CLEAR CLI OUTPUT).

• I CAN PRELOAD 5 PREDEFINED HABITS WITH ~4 WEEKS OF CHECK-OFFS TO DEMO THE FULL FLOW END-TO-END.

CLI EXAMPLES:

HABIT-TRACKER CREATE "EXERCISE" DAILY
HABIT-TRACKER CHECKOFF <HABIT_ID>
HABIT-TRACKER LIST --PERIODICITY DAILY
HABIT-TRACKER ANALYTICS

Architecture (Layers & Responsibilities)

• **MODELS** — **HABIT** (ID, NAME, PERIODICITY, CREATED_AT, CHECK_OFFS) AND **PERIODICITY** (DAILY/WEEKLY). THE MODEL PREVENTS DUPLICATE CHECK-OFFS PER DAY/WEEK AND EXPOSES **ADD_CHECK_OFF()**, **GET_CURRENT_STREAK()**, AND **GET_LONGEST_STREAK().**

• **PERSISTENCE** — DUAL BACKENDS:
– **SQLITEHABITREPOSITORY (DEFAULT):** TABLES **HABITS(ID, NAME, PERIODICITY, CREATED_AT)** AND **CHECK_OFFS(HABIT_ID, CHECK_OFF_TIME)** + INDEX ON **HABIT_ID**, TRANSACTION-SAFE CRUD.

– **JSONHABITREPOSITORY** (PORTABLE): STRUCTURED JSON FILE WITH SAFE INITIALIZATION AND CRUD.
BOTH BACKENDS SHARE THE SAME API AND A GUARDED **ADD_CHECK_OFF(HABIT_ID, TIME)** PATH TO AVOID DUPLICATES.

• **ANALYTICS (FUNCTIONAL)** — PURE FUNCTIONS THAT OPERATE ON IN-MEMORY HABIT OBJECTS:
LIST_ALL_HABITS, LIST_HABITS_BY_PERIODICITY, GET_LONGEST_STREAK_OVERALL, GET_LONGEST_STREAK_PER_HABIT, GET_CURRENT_STREAKS, GET_HABITS_WITHOUT_RECENT_ACTIVITY(DAYS), GET_COMPLETION_RATE_BY_PERIODICITY, GET_HABIT_STATISTICS.

• **INTERFACE (CLI)** — CLEAN COMMANDS WITH FRIENDLY, VISUAL OUTPUT:
CREATE, DELETE, CHECKOFF, LIST [--PERIODICITY DAILY|WEEKLY], ANALYTICS.
USES **CLICK** FOR COMMANDS AND **RICH** FOR TABLES/PANELS.

• **SAMPLE DATA (BOOTSTRAP)** — HELPER TO POPULATE 5 PREDEFINED HABITS (3 DAILY, 2 WEEKLY) WITH ~4 WEEKS OF REALISTIC CHECK-OFFS FOR DEMOS AND TESTS.

ANALYTICS

- **LISTS:** ALL HABITS (SORTED BY CREATION DATE) AND BY PERIODICITY (DAILY/WEEKLY).

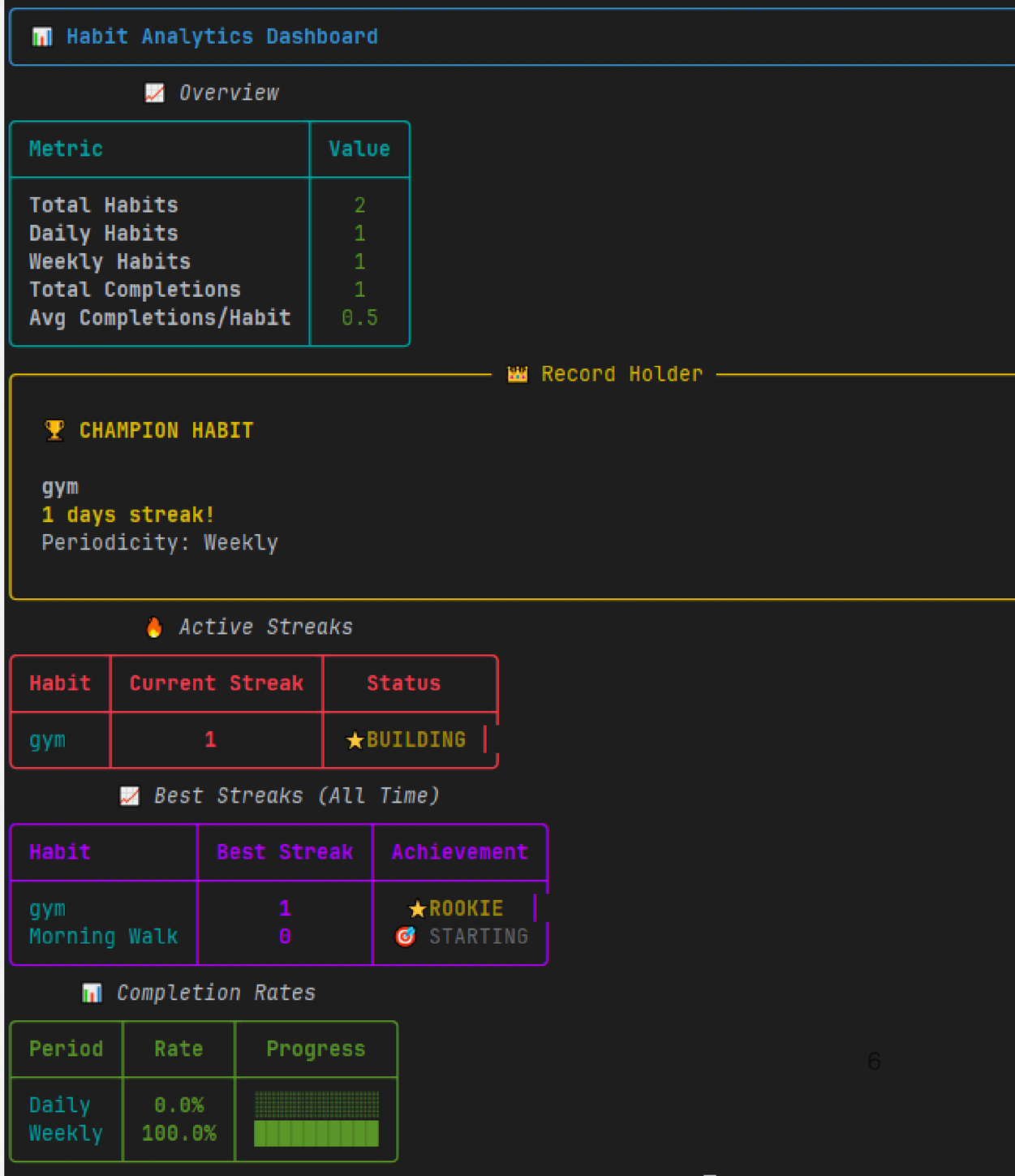
- **STREAK KPIS:** LONGEST STREAK OVERALL, LONGEST STREAK PER HABIT, AND CURRENT STREAK PER HABIT.

- **ACTIVITY GAPS:** HABITS WITHOUT RECENT ACTIVITY IN THE LAST N DAYS (CONFIGURABLE).

- **COMPLETION RATES:** DAILY VS WEEKLY COMPLETION RATE BASED ON EXPECTED VS COMPLETED CHECK-OFFS (CAPPED AT 100%).

- **SUMMARY STATS:** TOTALS, AVERAGES, AND COUNTS (TOTAL HABITS, CHECK-OFFS, LONGEST STREAK OVERALL, #HABITS WITH ACTIVE STREAKS).

- **CLI ANALYTICS VIEW:** A CHAMPION HABIT PANEL, ACTIVE-STREAKS TABLE WITH BADGES, AND COMPLETION-RATE PROGRESS BARS FOR QUICK SCANNING.



Evidence of Quality (Tests & Reliability)

• I WROTE UNIT TESTS FOR EVERY CORE LAYER.

– **MODELS:** CREATION, CUSTOM IDS, DUPLICATE PREVENTION (DAILY & WEEKLY), CURRENT/LONGEST STREAKS, AND SERIALIZATION.

– **ANALYTICS:** LISTS (ALL/BY PERIODICITY), LONGEST STREAK OVERALL, PER-HABIT STREAKS, CURRENT STREAKS, INACTIVITY, COMPLETION RATES, AND SUMMARY STATS.

– **PERSISTENCE (SQLITE & JSON):** CRUD, UPDATE WITH CHECK-OFFS, DELETE, GUARDED ADD_CHECK_OFF, AND TEMP-FILE ISOLATION.

– **SAMPLE DATA:** 5 PREDEFINED HABITS (3 DAILY, 2 WEEKLY), REALISTIC ~4-WEEK HISTORIES, REPO POPULATION, AND SUMMARY TEXT.

• **HOW I RUN THEM:** PYTEST -Q (FAST, ISOLATED, REPEATABLE).

• **CLI RELIABILITY**: COMMANDS ARE TYPED & VALIDATED (CREATE, DELETE, CHECKOFF, LIST, ANALYTICS) WITH CLEAR OUTPUT FOR SUCCESS/ERRORS.

• **WHY THIS MATTERS FOR USERS:** CORE LOGIC IS CORRECT, STREAK MATH IS TRUSTWORTHY, AND DATA STAYS CONSISTENT ACROSS SQLITE/JSON BACKENDS.