Expense tracker:

**Assumptions:**

* **Language/Framework**: Python 3.3 with built-in modules (Json, os, datetime) for zero dependencies. CLI chosen over API/UI for quick setup and focus on core logic.
* **Data Model**: Expenses as list of dicts: **{'amount': float, 'date': str (YYYY-MM-DD), 'note': str, 'category': str}**. Stored in JSON for easy persistence (human-readable, no DB setup needed).
* **Validation**: Amount > 0 (float), date in YYYY-MM-DD (no future dates enforced), notes/categories trimmed but not length-limited. Indices are 1-based for user-friendliness.
* **Error Handling**: Try-except for parsing/file ops; user-friendly messages. No crashes on invalid input—prompts retry implicitly via loop.
* **Edge Cases**: Empty list handled; filters return empty if no matches; file corruption starts fresh.
* **Good-to-Haves Scope**: Categories optional; summaries use simple dict grouping (O(n) time, fine for small datasets); monthly grouping uses date prefix (YYYY-MM).

**Design Decisions:**

* **Class-Based**: **ExpenseTracker** encapsulates state (expenses list) and methods for clean separation (load/save as core ops).
* **CLI Loop**: Infinite loop in **main()** with command parsing for interactivity. Commands are case-insensitive and simple.
* **Persistence**: Auto-save on mutations (add/update/delete) to ensure data safety.
* **Performance**: In-memory list for speed. suitable for hundreds of expenses.

By- Chhavi Jain

9414552783

Jainchhavi1111@gmail.com