LLM-powered FAQ-service with user query logging and retrieval

Objective

Design and **implement** a **simple backend service** in Python that:

- Integrates the **Mistral Large Language Model** via **Ollama** to run it locally (https://ollama.com/library/mistral).
- Provides an **API endpoint** to answer user queries using a given **knowledge base** (local file).
- Stores the **queries** and **model responses** in a local database.
- Provides **API endpoints** to retrieve past queries and their answers.

Deliverables

- Clean, modular and well documented Python source code.
- A GitHub repository containing the code and a README.md file with detailed setup instructions.
- Dockerfile.

Scenario

Your company is building an **AI-driven customer support assistant**. As a backend developer, your task is to **build a prototype backend service** that integrates **Mistral LLM** to answer customer questions using a predefined knowledge base.

Tasks

1. Build a REST API using Python via the use of FastAPI

- **Endpoint 1**: POST /ask
  - Accepts a JSON body with a user query {"question": "..."}
  - Uses the LLM to generate an answer based on the provided knowledge base.
  - Stores the query, answer, and timestamp in a database (SQLite).
  - Returns the answer as JSON {"answer": "..."}
- **Endpoint 2**: GET /history?n=35
  - Returns the last N questions and answers with timestamps.
  - On the request the user can set N as a query parameter. Use 10 as default value.

2. LLM Integration

- **Use** the **LLM to generate answers** from a small **knowledge base** (e.g., a .txt file, markdown, or structured FAQ document):
  - You may implement a simple RAG (retrieval-augmented generation) mechanism or context injection into the prompt

3. Prompt Engineering (Basic)

- Design a prompt that feeds the model the appropriate context (using, for example, a summarised knowledge base or top-matching FAQ).

- **Bonus**: Implement simple keyword matching or similarity search to improve relevance.

4. Persistence module

- Store user queries and model responses with timestamps in an SQLite database
- Write a simple ORM model (e.g., using SQLAlchemy or sqlite3 module).

Bonus Points

- Add basic logging and error handling

Knowledge Base Example (for reference)

You can provide a markdown or .txt file with content like:

Q: What is the refund policy?
A: Our refund policy allows customers to return products within 30 days...
Q: How can I contact support?
A: You can reach us via email at support@example.com...

Feel free to generate the knowledge base via the use of LLM but please specify the LLM of your choosing and the prompt in the README.md and the file itself in the code.

AI Coding Assistant Usage

The use of AI coding assistants such as **GitHub Copilot**, **Claude**, or **ChatGPT** is acceptable and encouraged for productivity.

Please specify in the README.md which tool(s) you used, and how they helped (e.g., generating boilerplate, debugging, prompt tuning).
