Knowledge Base Usage Report

Project Title: City Info Retrieval API
Purpose: To provide intelligent answers to user queries based on a city knowledge base using a Retrieval-Augmented Generation (RAG) system.

❖ Data Flow and Processing Overview

1. Knowledge Base Loading

- **Source**: JSON file (JSON_FILE_PATH) containing city-related information.
- **Step**: load_and_prepare_vectorstore()
- **Action**: Converts city data into vector embeddings for semantic search.

2. RAG Chain Setup

- **Step**: setup_rag_chain(vectorstore)
- **Action**: Links a retriever (vectorstore) with a language model to enable question-answering based on retrieved content.

3. Handling User Queries

- **Endpoint**: POST /query
- **Model**: QueryRequest
- **Action**:
    - Accepts a user question.
    - Checks for cached results.
    - If not cached:
        - Retrieves relevant content using the vectorstore.
        - Generates a response via the RAG chain.
        - Caches the response.

4. Searching the Knowledge Base

- **Endpoint**: POST /search
- **Action**: Returns the top-k semantically relevant entries from the vectorstore based on user input.

5. Accessing Raw Knowledge Base

- **Endpoint**: GET /data

- **Action**: Allows users to view all original knowledge base entries

Tools Implemented

- FastAPI: Web framework for API routes
- Pydantic: Validates data models
- LangChain and RAG: Handles retrieval and generation
- Vectorstore: Enables semantic document search
- AsyncIO + ThreadPoolExecutor: Improves responsiveness
- Hashlib + LRU Cache: Speeds up repeated query results