Chaitrali Katkar

NUID : 002811805

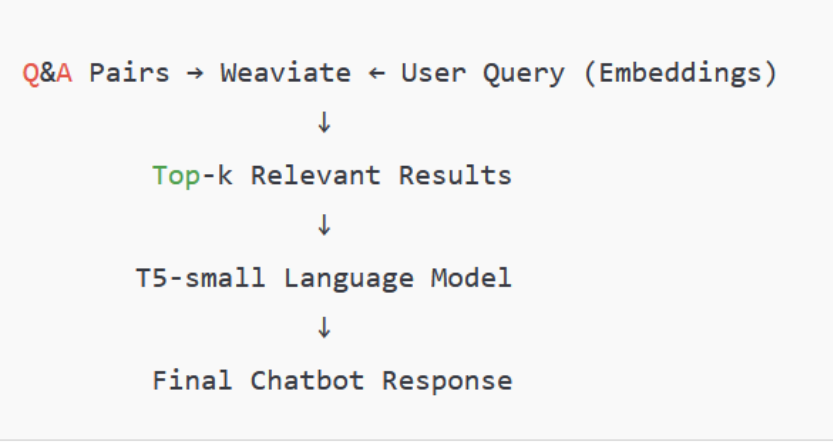
Title: Personal Finance Chatbot with RAG (Retrieval-Augmented Generation)

1. System Architecture Diagram

The architecture involves three main components:

1. **Data Ingestion:** Q&A pairs are ingested into Weaviate with embeddings generated using SentenceTransformer.
2. **Retrieval:** User queries are encoded into embeddings and matched with stored vectors in Weaviate using semantic similarity.
3. **Response Generation**: Top retrieved results are passed to a T5-small language model to generate natural-sounding responses.

Diagram:



2. Implementation Details

Technologies Used:

- **Weaviate**: Vector database for semantic search.
- **SentenceTransformer**: Model for embedding generation (all-MiniLM-L6-v2).
- **Hugging Face Transformers**: T5-small language model for response generation.
- **Streamlit**: Chatbot UI for user interaction.

Key Steps:

1. **Data Upload**:
    - Q&A pairs are loaded from a JSON file and stored in Weaviate with custom embeddings.
2. **Retrieval**:
    - User query → Embedding → Weaviate vector search → Top-k matching results.
3. **Response Generation**:
    - Retrieved Q&A pairs are formatted as context.
    - Passed to the T5-small model to generate a dynamic response.
4. **Chatbot UI**:
    - Built using Streamlit for easy user interaction.

3. Performance Metrics

Retrieval Accuracy:

- Evaluated by the relevance of Q&A pairs returned from Weaviate.

Response Coherence:

- Measured subjectively by comparing generated responses to actual answers.

Example Test:

User Query	Retrieved Answer	Generated Response
"How much did I spend on food?"	"You spent $85 on Snacks."	"You spent $85 on snacks."
"What was my income last month?"	"Your income was $1,500."	"You earned $1,500."

4. Challenges and Solutions

1. **Challenge**: Integrating Weaviate for Vector Search.
   Solution: Used REST API endpoints and ensured proper schema setup.
2. **Challenge**: Generating coherent responses.
   Solution: Combined retrieval results into a context and passed them to the **T5-small** model.
3. **Challenge**: Managing edge cases (e.g., ambiguous or invalid queries).
   Solution: Added query validation and fallback responses in the Streamlit UI.

5. Future Improvements

1. **Enhanced Language Models**: Use GPT-4 or larger LLMs for more accurate and natural responses.
2. **Memory Management**: Add conversation history for context-aware replies.
3. **UI Improvements**: Provide response explanations or data visualizations.
4. **Real-Time Data Integration**: Pull live financial updates for dynamic responses.