Project Report : Fashion Query Support Chatbot

1. Aim

In this project, we aim to create a generative search enabled chatbot which supports user with appropriate choices against their queries and preferences. It also provides user-friendly answers and relevant information about the apparels like brand and price assisting the users to make decisions.

2. Design and Considerations

While designing the project, some key decisions have been made based on requirements and observations

Considerations:

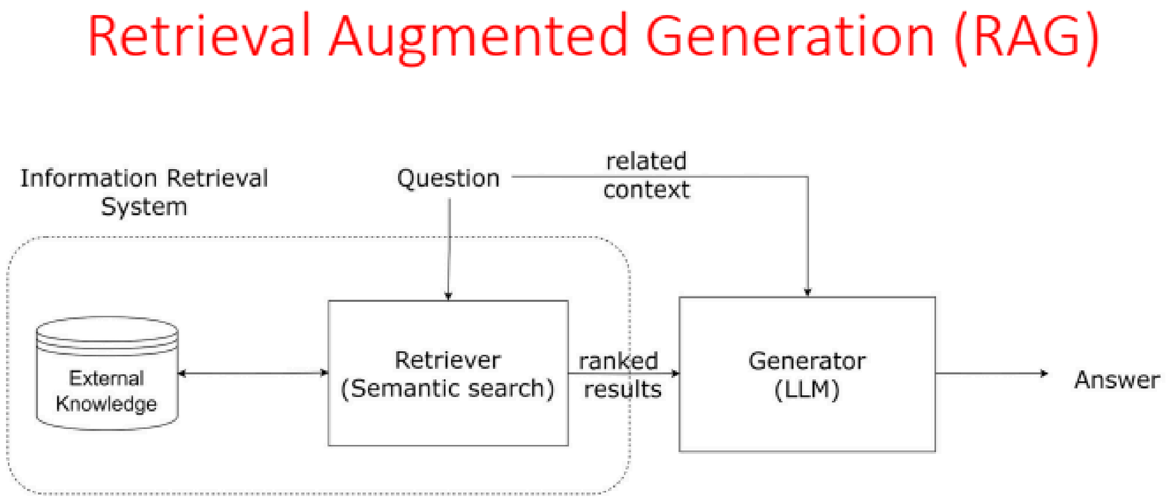
1. We need a vector DB, for storing and query purposes. So, we have chosen ChromaDB, which is supported thoroughly by Python and is open source.
2. We need a sentence embedding transformer for the embedding search. We use the embedding model: "all-MiniLM-L6-v2"
3. **Chunking:** As the concept of chunking is pretty much essential to break down the document and make sense of the embeddings, we ponder on the requirement. We have observed the description itself is very prominent and short. So, explicit chunking will be a trivial matter. So, chunking is not done.
4. We see the name and description provide enough to be treated as documents and help the search, so we use a combination of these 2 in the documents

Architectural Design:

The architecture consists of 3 main layers:

1. Embedding Layer

2. Search Layer
3. Generation Layer


Retrieval Augmented Generation (RAG)

Embedding Layer : This layer handles the preprocessing of the data input, creation of embeddings of the input documents and metadata and storing in the vector database. Uses Sentence Transformers from HuggingFace and ChromaDB

Search Layer: This layer uses the client of the chromaDB database created in the 1st Embedding step and additionally another database for caching purposes. It handles the querying and producing top 10 most appropriate results.

In this layer, we also rank the outputs based on distances and similarity to the query via a re-ranking mechanism (cross-encoder) offered by sentence Transformer library of Hugging Face.

Generation Layer: This is the final layer which generates the output. The final LLM based layer where the task and guidelines are defined. We design and provide a prompt indicating the query and source database to the LLM. The model used here is GPT-3.5-Turbo.

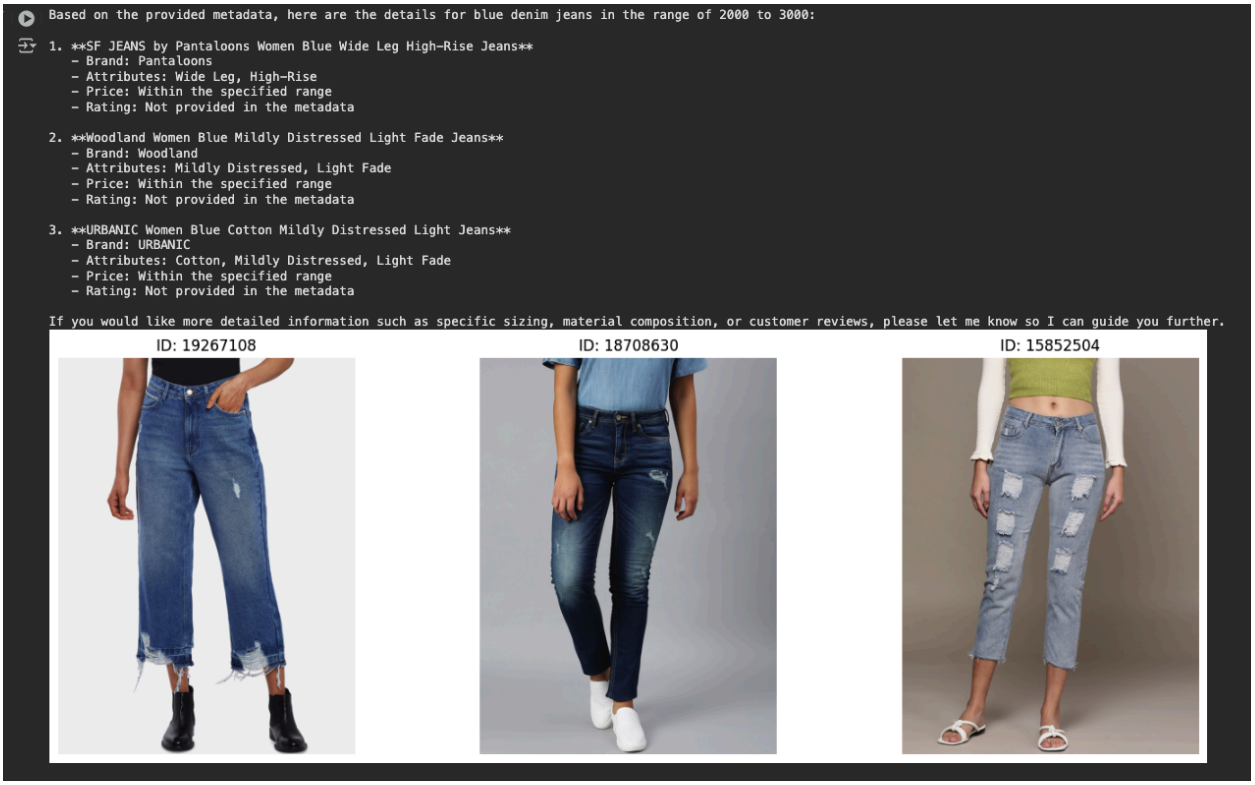
Input:

1. **The knowledge Database : https://www.kaggle.com/datasets/djagatiya/myntra-fashion-product-dataset**
2. **Query**

Output:

Top 3 suggestions

Example Query



```
Based on the provided metadata, here are the details for blue denim jeans in the range of 2000 to 3000:

1. **SF JEANS by Pantaloons Women Blue Wide Leg High-Rise Jeans**
   - Brand: Pantaloons
   - Attributes: Wide Leg, High-Rise
   - Price: Within the specified range
   - Rating: Not provided in the metadata

2. **Woodland Women Blue Mildly Distressed Light Fade Jeans**
   - Brand: Woodland
   - Attributes: Mildly Distressed, Light Fade
   - Price: Within the specified range
   - Rating: Not provided in the metadata

3. **URBANIC Women Blue Cotton Mildly Distressed Light Jeans**
   - Brand: URBANIC
   - Attributes: Cotton, Mildly Distressed, Light Fade
   - Price: Within the specified range
   - Rating: Not provided in the metadata

If you would like more detailed information such as specific sizing, material composition, or customer reviews, please let me know so I can guide you further.
```

Requirements to Run:

Please provide a file with the OpenAPI key and name it “OpenAI-Key.txt” and save it to base project.