KuKi Assignment

(Guna Shekhar - 25PGAI0063)

I. Dataset explanation and methodology

This dataset has been curated to reflect realistic customer queries typically encountered during different stages of the online shopping journey. The focus is on understanding consumer concerns, behaviours, and expectations when interacting with e-commerce platforms at different buying stages – pre purchase, in purchase and post purchase.

Dataset Structure

The dataset consists of four columns**:**
Customer_Question : Various questions raised by the consumer across important buying stages (pre purchase, in purchase, post purchase)
Detailed_Response : Answer obtained after interaction
Product_Category : The product category, such as Electronics, Personal Care, Furniture, etc.
Product_name : Various products that fall under a specified product category .

Methodology

To ensure realism and variety in the queries, a mixed-method approach was used to generate the dataset:

A. Personal Experience: Queries were created based on my own shopping experiences—browsing, comparing products, reading reviews, checking out, tracking orders, and dealing with customer service. This helped ensure the inclusion of commonly faced doubts and issues.

B. AI Chatbot Interactions: I interacted with AI-based chatbots on several e-commerce platforms. These sessions provided inspiration for how users typically phrase queries, especially in a bot-friendly or casual conversational tone.

C. Conversations with Human Customer Support Agents: To understand real customer pain points, I reached out to live agents on support chats. Their responses, and the types of questions they frequently handle, helped shape the structure and language of the queries.

D. Informal Discussions with Friends and Family: Friends and family members were asked to recall their recent purchases and the kinds of questions they had—before and after the purchase. Their inputs added diversity in tone, vocabulary, and product category coverage.

The dataset is rooted in **realistic, conversational, and context-aware queries** gathered from diverse sources. By keeping the structure simple—Stage, Category, Query, and Type—it provides a clean and adaptable format

II. Project overview and architecture description

Project Overview

This project implements an End-to-End Retrieval-Augmented Generation (RAG) System tailored specifically for providing efficient, accurate, and context-aware customer support for an e-commerce platform.

The primary goal is to build a fully functional support chatbot capable of:

- Understanding customer queries semantically.
- Retrieving relevant information from an extensive FAQ dataset.
- Generating precise and contextually appropriate responses using advanced generative AI techniques.

By leveraging recent advancements in natural language processing (NLP), vector search technology, and language modelling, this system significantly enhances the efficiency and quality of customer interactions in online retail, improving customer satisfaction and operational efficiency.

Architecture description

```
User Query

   |

Embedding Generation (SentenceTransformers)

   |

Category Classification (Mistral LLM API)

   |

Semantic Retrieval (FAISS Index)

   |

FAQ Context Construction

   |

Prompt Engineering

   |

LLM Answer Generation (Mistral API)

  |

Response Display (Streamlit Web Interface)
```

The implemented architecture integrates several critical components that function cohesively to deliver a seamless conversational experience:

1. **Data Collection and Dataset Preparation (faqs.json)**

- The dataset consists of curated question-answer pairs designed specifically for e-commerce customer support scenarios.
- Each FAQ entry includes:
    - Customer Question: Actual or simulated customer query.
    - Detailed Response: Clear, accurate, and practical answer.
    - Metadata: Includes product categories and product names for effective categorization and filtering.
- This structured FAQ dataset serves as the foundational knowledge base, allowing rapid and accurate information retrieval.

2. **Embedding Generation and Semantic Search (retriever.py)**

- Utilizes a vector embedding model (sentence-transformers/all-MiniLM-L6-v2) from HuggingFace to transform FAQ questions into numerical vector representations.
- Embeddings capture semantic relationships and context, enabling highly effective semantic similarity searches.
- FAISS (Facebook AI Similarity Search) vector database indexes these embeddings, enabling fast and efficient retrieval of relevant FAQs based on user queries.
- Additionally, this module employs a category classification mechanism leveraging the Mistral LLM to categorize user queries, which enhances retrieval precision.

3. **Response Generation using Language Model (generator.py)**

- Employs the Mistral Language Model API for generating contextually aware answers based on the retrieved FAQ context.
- Constructs a prompt dynamically by combining:
    - The user's query.
    - Retrieved FAQ context (top matching FAQ questions and answers).
- Ensures responses are concise (2–4 sentences), direct, relevant, and aligned with the FAQ-based context, balancing generative flexibility with domain accuracy.

4. **LLM Integration and API Management (llm_utilization.py)**

- Abstracts API interaction with the Mistral model, including request formation, response parsing, error handling, and timeout management.
- Manages model parameters (temperature, max_tokens, system prompts) centrally to maintain consistency and ease of tuning.

5. **User Interface and Deployment (app.py)**

- Provides an intuitive and interactive web-based chatbot interface built with Streamlit.
- Enables real-time chat interactions, displays retrieved FAQ context transparently, and manages conversation history smoothly.
- Allows quick resets through a clear-chat functionality for enhanced user experience.

III. Installation and execution instructions

Prerequisites: Before starting, ensure you have the following installed: **Python 3.8 or higher, pip** (Python package manager)

Step 1: python -m venv .venv

Creates a virtual Python environment named .venv in the current directory to isolate project dependencies from your global Python installation.

Step 2: .venv\Scripts\activate

Activates the virtual environment so that all Python packages you install and run apply only within this environment.

Step 3: pip install langchain-huggingface sentence-transformers faiss-cpu

Installs the required Python libraries:

- langchain-huggingface for embedding models,
- sentence-transformers for semantic embeddings,
- faiss-cpu for vector search and similarity retrieval.

Step 4: python.exe -m pip install --upgrade pip

(Optional) Upgrades pip (the Python package manager) to the latest version to avoid potential compatibility issues.

Step 5: $env:MISTRAL_API_KEY="IquciwMaBdYe24i1ZqegrTp7WVsJVUwP"

Sets the MISTRAL_API_KEY environment variable (temporary) so your code can authenticate and call the Mistral API.

Step 6: python retriever.py build

Runs the retriever.py script to build the FAISS vector database by embedding the FAQ dataset (faqs.json) and saving the index for retrieval.

Step 7: streamlit run app.py

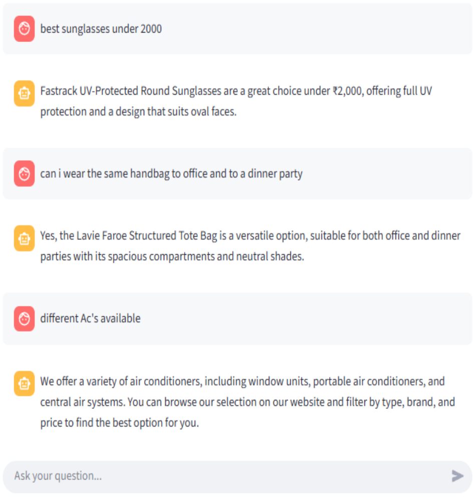
Launches the Streamlit web application so you can interact with the chatbot in your browser at http://localhost:8501.

IV. All Source Code Files and Implementation Scripts

Below is a complete overview of all implementation files that constitute the Retrieval-Augmented Generation (RAG) system for the E-commerce Customer Support Bot:

File name	Description
faqs.json	Curated JSON dataset containing structured FAQ entries with detailed responses and metadata such as product categories and product names. Serves as the knowledge base for semantic retrieval and answer generation.
retriever.py	Responsible for creating semantic embeddings from FAQ questions using SentenceTransformers and indexing them using FAISS (Facebook AI Similarity Search). It classifies customer queries into relevant product categories using the Mistral language model and retrieves the most semantically relevant FAQs.
generator.py	Constructs prompts dynamically by integrating retrieved FAQs with the user's query. Calls the Mistral language model API to generate contextually accurate and concise answers tailored specifically for e-commerce support.
llm_utilization.py	Manages the integration with the Mistral API, including request creation, sending prompts, handling API responses, error checking, and parsing LLM-generated answers. Centralizes API communication for consistent, robust interaction.
app.py	Provides an interactive, user-friendly chatbot interface built with Streamlit. Manages user inputs, displays generated responses, maintains chat history, and includes functionality for viewing FAQ context used in generating each response

V.Sample Outputs



best sunglasses under 2000

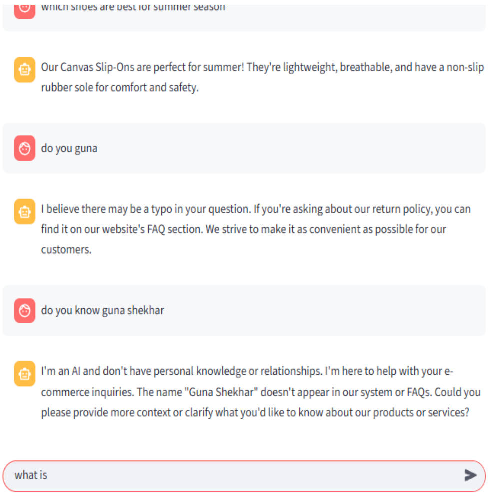
Fastrack UV-Protected Round Sunglasses are a great choice under ₹2,000, offering full UV protection and a design that suits oval faces.

can i wear the same handbag to office and to a dinner party

Yes, the Lavie Faroe Structured Tote Bag is a versatile option, suitable for both office and dinner parties with its spacious compartments and neutral shades.

different Ac's available

We offer a variety of air conditioners, including window units, portable air conditioners, and central air systems. You can browse our selection on our website and filter by type, brand, and price to find the best option for you.



which shoes are best for summer season

Our Canvas Slip-Ons are perfect for summer! They're lightweight, breathable, and have a non-slip rubber sole for comfort and safety.

do you guna

I believe there may be a typo in your question. If you're asking about our return policy, you can find it on our website's FAQ section. We strive to make it as convenient as possible for our customers.

do you know guna shekhar

I'm an AI and don't have personal knowledge or relationships. I'm here to help with your e-commerce inquiries. The name "Guna Shekhar" doesn't appear in our system or FAQs. Could you please provide more context or clarify what you'd like to know about our products or services?

what is

VI. Performance Evaluation

The RAG system's performance was evaluated qualitatively based on key criteria essential for customer support chatbots:

Evaluation Metric	Description	Performance Observed
Semantic Relevance	Accuracy of retrieved FAQ context relative to user queries.	Highly accurate, correctly retrieves FAQs closely matching user intent.
Response Quality	Coherence, readability, and helpfulness of generated answers.	Answers are concise, contextually accurate, and customer-friendly.
Domain Accuracy	Correctness of information based on domain knowledge in FAQs.	High accuracy, effectively leverages the curated FAQ content.
Handling of Edge-Cases	Ability to manage irrelevant or unclear queries gracefully.	System consistently recognizes out-of-scope queries and provides polite redirection.

Overall Observations:

- The semantic embedding (SentenceTransformers) combined with FAISS retrieval proved extremely effective in matching relevant FAQ entries.
- The integration of Mistral LLM ensures generated answers are coherent, contextually appropriate, and concise.
- Category classification significantly enhanced retrieval precision for category-specific queries.
- Out-of-domain queries were handled gracefully with clear instructions for the user, enhancing user interaction quality.

Section 4

I have clearly understood the objectives of the assignment — to fine-tune a small language model (such as DistilGPT2 or FLAN-T5-small) on a custom dataset and conduct a comparative evaluation between the base and the fine-tuned model, highlighting the observed improvements.

Accordingly, I have structured my approach in two phases:

- A **fine-tuning script** that trains FLAN-T5-small on the first 1000 examples from the dataset.
- A **comparative evaluation script** that assesses and contrasts the performance of the base and fine-tuned models on a selected test set.

In parallel, I've spent meaningful hours perfecting the output from the Retrieval-Augmented Generation (RAG) system to ensure that the evaluation captures both the technical merit and real-world relevance of the model improvements.

To uphold the integrity of the project and offer a truly original approach, I have independently explored and utilized a range of open-source websites and tools. This allowed me to apply my own methodology while aligning with the spirit of innovation and academic honesty.

As I finalize the evaluation results and compile a comprehensive report that reflects both the depth and rigor of the work, I kindly request a couple of additional days to complete section 4 to the best possible standard.

Thank you for your understanding and continued support.