JULIEN GODFROY

POC : CUSTOMER SERVICE CHATBOT

ILIAD RECRUITMENT PROCESS: ML LLM ENGINEER

Contexte [FR]:

“

Ce projet est réalisé dans le cadre de l’épreuve technique du processus de recrutement du groupe Iliad, pour le **poste de ML LLM Engineer**.

Le projet consiste à créer un ChatBot basé sur un LLM, capable de répondre aux questions des clients de Free. L’objectif de ce projet est d’aider les techniciens du Support Client à répondre efficacement aux demandes.

Toutes les technologies et les LLM évoqués dans ce projet sont entièrement **open-sources**.

Ce rapport technique est rédigé en anglais, pour respecter les conventions.

”

Table of Contents

Project Overview **4**

System Architecture & features **4**

Data Preparation 5

Vectorized Data Creation 5

Similarity Search 5

Chatbot Interaction 5

Web Interface 6

Conversation memory 6

Conclusion **6**

References **Error! Bookmark not defined.**

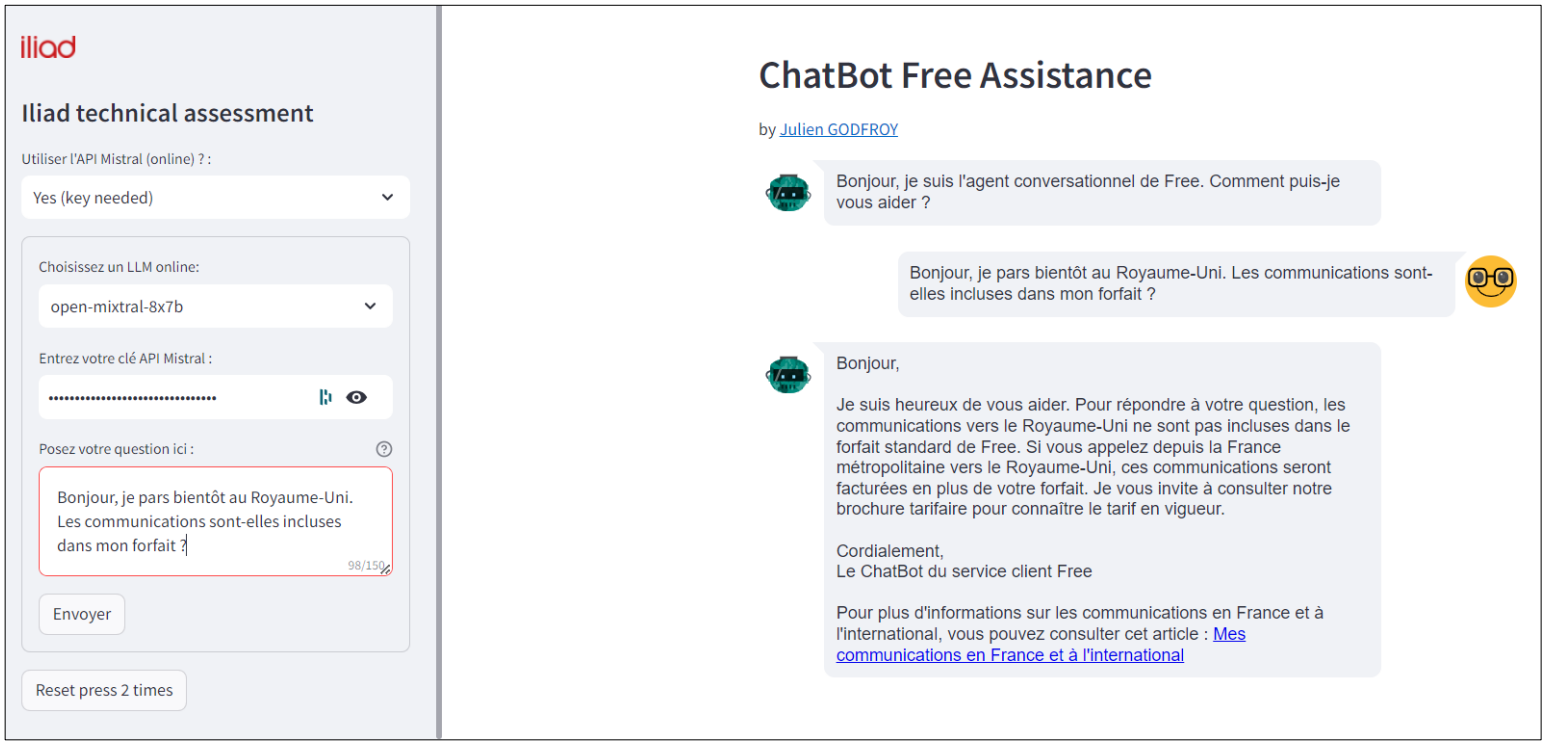
Project Overview

The Free Customer Service Chatbot is developed as a PoC to demonstrate the potential of automated customer service solutions in handling user queries efficiently. The chatbot operates on a foundation of articles available in the "customer service" section of Free's website, ensuring a comprehensive knowledge base. The main technologies include:

LangChain: Used for processing natural language, enabling the chatbot to understand and generate human-like responses.

ChromaDB: Facilitates the vectorization of textual data, allowing for the efficient retrieval of documents similar to user queries.

Ollama and Mistral API: Provides the flexibility to choose between local and online models for generating responses, catering to different operational needs. Note that every selected models are open-**source and usable in commercial purpose.**



System Architecture & features

Data Preparation

The prepare_df(file) function processes the SQLite database containing customer service articles. This step involves removing duplicate entries, calculating the length of content, and merging the title and content of each article to create a unified representation.

Vectorized Data Creation

Utilizing ChromaDB and the load_vectorized_data(df, model="all-MiniLM-L12-v2", recreate=False) function, the prepared data is vectorized. This process embeds the articles in a high-dimensional space, enabling the system to perform similarity searches effectively. The function supports both the creation of new vectorized data and the loading of existing data.

Similarity Search

The search_similar_documents(db, query, k=2) function searches the vectorized data for articles that are most similar to the user's query. This step is crucial for finding the most relevant information to base the chatbot's response on. If the similarity score is above a defined threshold, the information is not returned. This, to avoid to any unnecessary element in the prompt.

Chatbot Interaction

Interaction with the chatbot is facilitated through functions like query(...) and main(...), which manage the generation of responses based on similar documents found and the conversation history. This approach ensures that the chatbot can provide contextually relevant answers.

Web Interface

A Streamlit web interface allows users to interact with the chatbot easily. The interface includes options for LLM selection, API key input for the Mistral API, and conversation history management, enhancing the user experience. For this POC, the UI makes it easier the switch between models and configuration.

Conversation memory

Upon accessing the Streamlit application, users can engage with the chatbot by typing queries and receiving context-aware responses. The system's ability to maintain a conversation history ensures that each response is informed by previous interactions, significantly enhancing the relevance and accuracy of the information provided.

Model choice and configuration

For **local execution**, users can use Ollama by running ollama pull <modelname> to download and use various models directly on their machine. Local execution offers the advantage of **GPU acceleration**, significantly enhancing performance for computationally intensive tasks. However, running models locally can be very slow depending on the hardware capabilities.

Alternatively, users have the option to use **online** open-source Mistral models via **the Mistral API**, eliminating the need to run models locally and manage hardware constraints. This approach requires an API key, which can be easily inputted into the system, providing a seamless and efficient way to access powerful models without local computational overhead.

Conclusion

The Free Customer Service Chatbot PoC exemplifies the potential of integrating modern NLP and document vectorization technologies to create an effective automated customer service solution. By leveraging a comprehensive knowledge base and advanced computational models, the chatbot demonstrates a significant capacity for improving the efficiency and quality of customer service interactions.