**Assignment: Chatbot**

**Submitted By: Rajesh Kumar Sharma**

**Date: 18-Oct-24**

**Assignment:** **To develop a chatbot for an enterprise that should support the customer's 24/7**

* **Use any open-source model. Chatbot should be restricted to specific industry**
* **Deploy your chatbot in streamlit UI**
* **Submit the code, and detailed project report**

**Chatbot:** Artificial intelligence is used to construct a computer program known as "a chatbot" that simulates human chats with users. It employs a technique known as NLP to comprehend the user's inquiries and offer pertinent information. Chatbots have various functions in customer service, information retrieval, and personal support.

**Limitations of Chatbot:** Chatbot have some limitations to be aware of:

* Lack of semantic understanding: Chatbots may require assistance comprehending the discourse, which could result in misinterpretation or incorrect responses.
* Dependency on training [data:](https://www.simplilearn.com/what-is-data-article) The calibre and volume of training data greatly impact the chatbot performance. There may be a need for more accurate or biased training data, which can result in incorrect responses.
* Handling complicated queries: Chatbots could encounter questions beyond simple pattern matching and call for greater comprehension or deductive reasoning.

**Setting-up environment:**

1. Go to Anaconda and create the environment.

2. Open the terminal

3. Type the command to install streamlit - ***pip install streamlit***

4. Test is the installation worked - ***streamlit hello***

5. Below page opens up:

A screenshot of a computer

Description automatically generated

Similarly, we have to install langchain and langchain\_community as well.

***pip install langchain\_community***

***pip install langchain***

***Alternatively,*** if we want the installations to do in one shot, we can prepare a text file: *requirements.txt* with the below contents:

Streamlt

Langchain

Langchain\_community

Run the below command for the installation:

***pip install -r requirements.txt***

Before we run the chatbot code, we have to make sure that we have downloaded the llama model on your local machine and its running.

Open the terminal and run the following command

***ollama run llama2***



**Running the streamlt code:**

***streamlit run file\_name.py***

**Code Link:** <https://github.com/rajeshksharmasls/GenAI/blob/master/GenAI/Chatbot/chatbot_streamlit_opensourcellm.py>

**Dataset:** *None*

**Process:** Below is the summarization of the code:

Importing Libraries:

Streamlit is used for building a simple web interface.

LangChain's ChatPromptTemplate is used to structure chatbot prompts.

Ollama is used to initialize the Llama2 model for generating chatbot responses.

Defining the Chatbot Prompt:

A prompt template is defined with a system message ("You are a helpful assistant...") and a placeholder for the user input (formatted as "Question: {question}").

This template structures the interaction between the user and the chatbot.

Initializing the Llama2 Model:

The Llama2 model is loaded through Ollama to handle text generation based on user queries.

Creating a Response Chain:

The prompt and the model are combined into a chain using |, which will generate responses when invoked with user input.

Retail-Specific Inquiry Handling:

The function get\_retail\_response() handles specific retail-related queries:

It responds to common queries such as product availability, order status, promotions, store locations, and return policies with predefined answers.

If no retail-specific match is found, the function defaults to the Llama2 model for a generic response using the chain.invoke() method.

Streamlit Web App Setup:

Title: The app title is set to "Retail Industry Chatbot (Llama2 with LangChain)".

User Input: Users can type a question in a text input field.

Conversation History: If it's the user's first interaction, a conversation history is initialized in the session state.

Send Button: When the user clicks "Send", the chatbot processes the input:

The user's message is appended to the conversation history.

The chatbot's response (either predefined or generated by Llama2) is also appended to the history.

Display History: The conversation history is displayed, showing both user input and chatbot responses.

**Snapshots of the conversations with the Chatbot:**

*Query Prompt:*

A screenshot of a chatbot

Description automatically generated

*Order Status:*

A screenshot of a chatbot

Description automatically generated

*Return Policy:*

A screenshot of a chatbot

Description automatically generated

*Generic Response:*

**

**Conclusion:** As witnessed, the Chatbot is able to get the specific responses to the user queries for the stores or e-commerce from retail industry. Even the generic queries are being answered by the chatbot.

The Chatbot can be further improved by doing the following:

Integrating with APIs**:** Chatbot can be connected to the actual retail systems (inventory, order tracking) to provide real-time information.

Improve NLP:Chatbot can be further fine-tuned or switch to more advanced models like GPT-Neo for better handling of complex queries.

Deployment: Chatbot can be deployed on platforms like Streamlit Cloud or Heroku for public access.