Zeeta Documentation

**Scope**

To train insurance agents in their daily tasks and facilitate career development, the chatbot should be intuitive, interactive, and capable of simulating complex conversational scenarios.

**Select**

In this project, the selected model is a pre-trained model called "Llama 2 7b" from the open-source platform Hugging Face.

**Llama-2 7b**

**Type:** Large Language Model (LLM)

**Developer:** Meta

**Size:** 7 billion parameters (7b)

**Requirement:** 40GB of RAM memory

**Function:**

* Generates text.
* Translates languages.
* Creates creative content.
* Answers questions informatively.
* Particularly adept at chat conversations (understanding and responding to natural language).

**Additional Notes:**

* "7b" indicates the model's complexity based on the number of parameters it can handle.
* Larger versions of Llama-2 exist with increased capabilities but require more computing power.

**Adapt and align model**

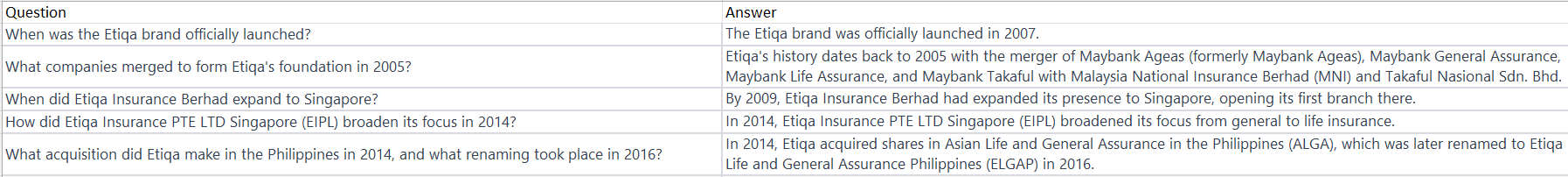
**RAG (Retrieval-augmented generation)**

RAG, a technique combining information retrieval with text generation, first retrieves relevant information from a vector database. In this case, the vector database used is Pinecone, and then utilizes this context along with its internal knowledge to generate a response.

The RAG workflow starts with storing the data into the vector database. Each piece of information in the knowledge base is represented as vectors in a multidimensional array of numbers, with Pinecone excelling at storing and searching these vectors. When input is provided, the LLM formulates a query and sends it to Pinecone, which searches the vector space to identify information vectors closest to the query vector, providing the most relevant information. The LLM then integrates this retrieved context from Pinecone with its own knowledge to generate a more accurate and informative response.

The data that needs to be stored in this vector database is in an Excel file containing questions and answers about the Etiqa company, organized in two columns, resembling a conversation.

Data:



**Prompt engineering**

Here is the template for the Llama 2 7b model:

<s>[INST] <<SYS>>

You are a helpful assistant.

<</SYS>>

Instruction[/INST]

</s><s>

Explanation:

* **<s> ... </s>**: These lines denote the start and end of a special code block within the text.
* **[INST]**: This indicates the beginning of an instruction for the model.
* **<<SYS>>**: This section defines system behavior.
* **You are a helpful assistant.** : This is the actual instruction given to the model, telling it to act as a helpful assistant during the interaction.
* **<</SYS>>**: This marks the end of the system behavior definition.

By following this template, we are able to use a few short prompts to efficiently store and retrieve information about the Etiqa company in a conversational format. Few-shot prompting is a technique for large language models (LLMs) like Llama-2 7b. It helps them perform specific tasks or generate desired outputs by providing a small set of examples. These examples, called "shots," demonstrate the format and style of the response required.

Here's how few-shot prompting works: First, a limited number of examples (typically 2-5) showcasing the desired response are provided to the LLM. These examples can take various formats, such as questions and answers or instructions and outputs, tailored to the task at hand. Next, by analyzing these examples, the LLM learns the desired structure, tone, and necessary information for the response through a process known as learning by demonstration. Finally, when presented with a new prompt related to the examples, the LLM utilizes its understanding from the provided examples to generate a more accurate and relevant response, leading to improved performance in generating responses.

In this case, few-shot prompting is used to control the output of each function, which serves to manage various aspects of sentence manipulation. These functions encompass segmenting the sentence into standalone units, categorizing the sentences based on their structure or content, and employing the corresponding function to handle each category effectively.

**Functions:**

Segmentation: Breaking down text into smaller, manageable units or segments for analysis or processing.

Standalone sentence: Make the sentence that can stand alone grammatically and contextually, conveying a complete thought or idea.

Classify the sentence: Categorizing a sentence based on its purpose, content, or structure.

Greeting: Create a salutation or expression of welcome at the beginning of a conversation.

Small talk: Create casual, light conversation typically used to establish rapport or fill silence.

Classify the sentence can be answer or not: Determining if a sentence requires a response or if it's merely a statement or observation.

Classify the sentence related to data we have or not: Assessing if a sentence pertains to information within our database or is unrelated.

Answer the question: Providing a relevant response to a query or inquiry.

Handle common social interactions: Managing typical social exchanges such as greetings, farewells, and expressions of gratitude.

Handle the unknown user’s input: Responding appropriately when encountering input that cannot be understood or processed.

End the conversation: Concluding the interaction and terminating the conversation.

End the training program: Exiting the training session or program being utilized.

Skip the question: Moving past a question without providing a response.

Request a hint: Asking for additional guidance or assistance in understanding or answering a question.

Start the training program: Initiating or beginning a training session or program.

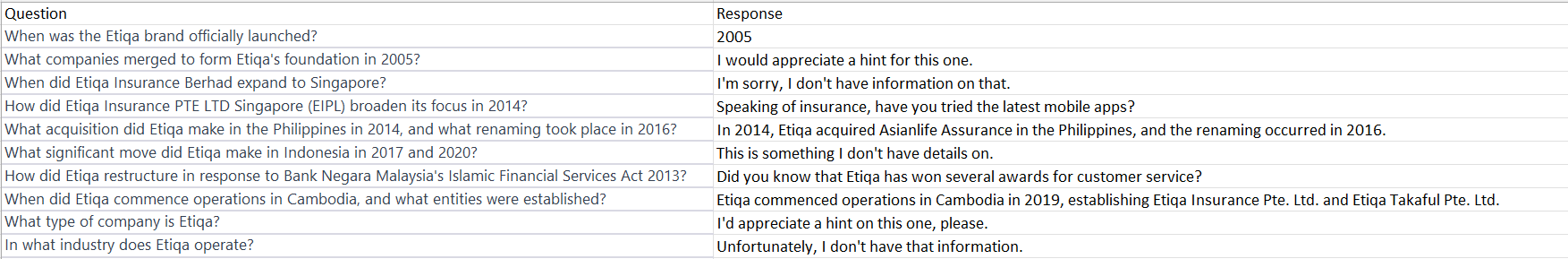
**Algorithm:**

The chatbot system algorithm is designed to seamlessly integrate various functions, necessitating the combination of these functions to achieve desired outcomes. One crucial aspect involves extracting pertinent data from the output of a large language model, ensuring that the extracted information can effectively activate or trigger the desired functions. Additionally, thorough pre-processing and post-processing procedures are implemented for all functions to optimize performance and ensure smooth operation within the chatbot system. These steps are vital for the algorithm's functionality and efficacy in delivering accurate and timely responses to user queries.

**Evaluate**

The testing phase includes over 100 manually created test cases tailored for each function. Human analysts evaluate the output quality to ensure accuracy and relevance. Below are the test data and results for one of the functions.

Test data:



Result:

