Brance AI engineer Task

Name:Hrishikesh Chougule

Linkedin Profile: https://www.linkedin.com/in/hrishikesh-chougule-7937401b6

Date Challenge Received:22-7-23

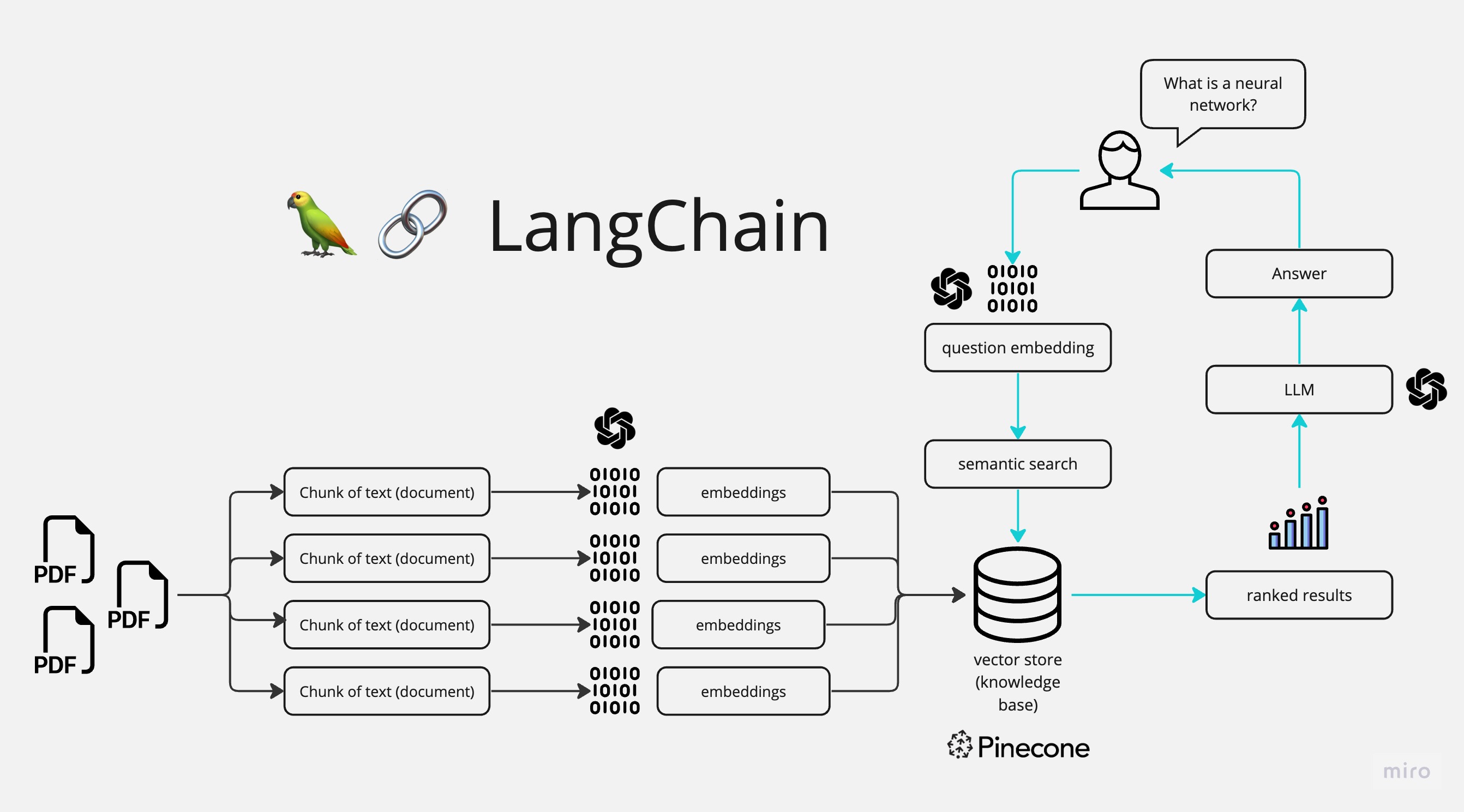
Date Solution Delivered:24-7-23



1. Problem Statement

Making a chatbot implementing the rag module finding out about the various embedding techniques applying the best one and feeding the data to any given llm to generate meaningful content.

2. Approach



3. Solution

* **streamlit**: This library is used to build interactive web applications with Python.
* **dotenv**: This library is used to load environment variables from a file named **.env**.
* **langchain**: This appears to be a custom library for natural language processing and conversational AI.
* Helper Functions: The code defines several helper functions:
* **read\_text\_file(file\_path)**: This function reads the contents of a text file specified by **file\_path** and returns the content as a string.
* **get\_text\_chunks(text)**: This function takes a large text and splits it into smaller chunks using a custom text splitter from the **langchain** library. The purpose of chunking the text is likely to manage memory and computational resources when processing large documents.
* **get\_vectorstore(text\_chunks)**: This function creates a vector store using the **OpenAIEmbeddings** model from the **langchain** library. The vector store is a data structure that allows for efficient similarity search between texts. It uses vector embeddings to represent text chunks in a high-dimensional space, where similarity between chunks is measured by the distance between their corresponding vectors.
* **get\_conversation\_chain(vectorstore)**: This function sets up a conversation chain, combining a language model (**ChatOpenAI**) with a retriever based on the vector store. This enables the system to retrieve relevant information from the knowledge document based on user queries.
* **handle\_userinput(user\_question)**: This function handles the user's input (question) and generates a response using the previously set up conversation chain. The response is then displayed on the web interface.
* Main Function: The **main()** function is the entry point for the Streamlit web application. It performs the following tasks:
* It sets the title and icon for the web page using **st.set\_page\_config()**.
* It initializes two session state variables, **st.session\_state.conversation** and **st.session\_state.chat\_history**, which will be used to store conversation data and history across user interactions.
* It displays the main header and provides a text input for users to ask questions about their documents.
* When the user enters a question and clicks the "Process" button, the application starts processing the user's query.

4. Future Scope

Looking for free alternatives for apis  
adding additional features like voice to text

Evaluation of answers and extending the model to beyond one document