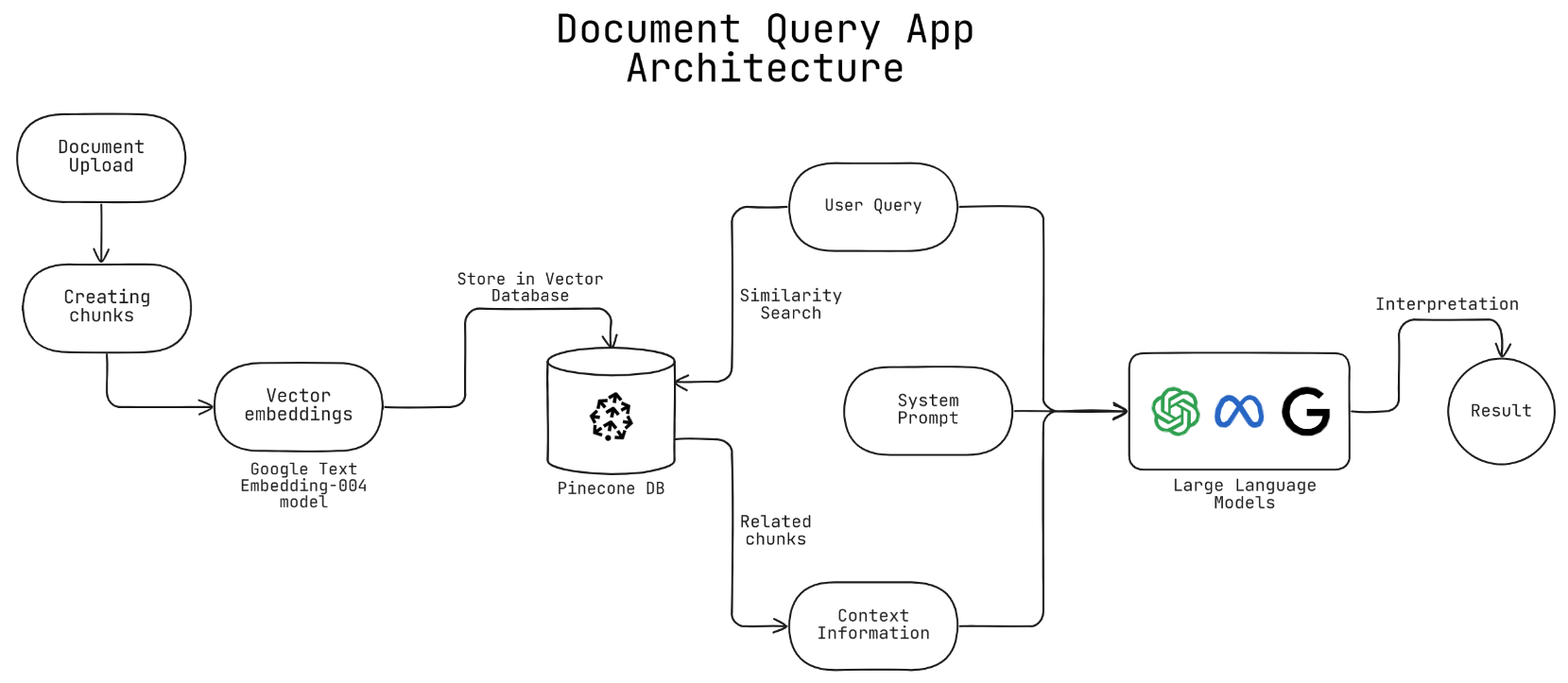
*Document Query App Architecture*

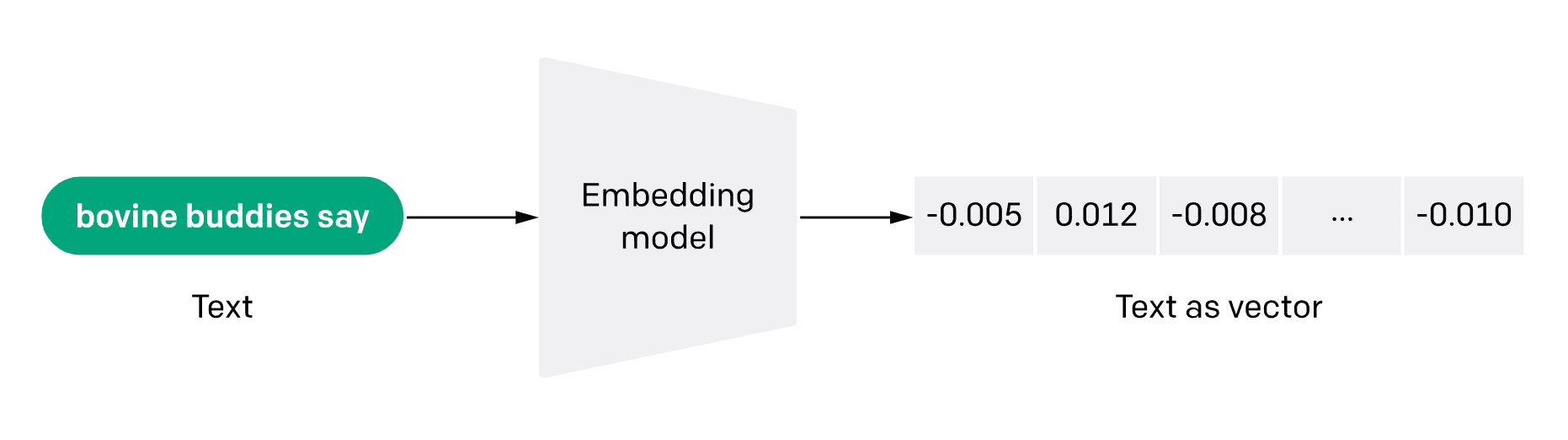
In this project, we aim to build a Document Query App using LLM. The architecture we have thought of for the app is as follows:



***COMPONENTS OF THE APPLICATION***

The main components of the application are:

1. **Document Chunker(Creating chunks)** : It will be a python function that will accept a file that will be uploaded by the user, read the contents of each page of the file, and convert in into chunks of some predefined size
2. **Vector Embeddings :** The next task of converting these chunks of data into vector embeddings will be handled using Google Text-Embedding-004 model API. This model accepts a string as input and converts it into a floating digit vector of **size 768**. These vectors will be used in performing similarity search.



1. **Pinecone Vector Database :** Pinecone is a vector database that will be used to store the vector embeddings that are created for all the chunks that are obtained from the user uploaded document. The vectors will be stored in the database using Pinecone API.
2. **System Prompt :**  The system prompt is given to a LLM model in order to provide it instructions as to how it should behave while giving responses. We can instruct the LLM model to respond to user queries only using the context information and not use any of its prior knowledge, as it can affect the accuracy of the responses.
3. **Context Information :** The user query is also converted into a vector using the same Google Embedding-004 model. This vector is then compared with the stored vector embeddings in Pinecone, using similarity search. The vectors from the database that match the query vector will certainly be of those chunks that have the required information that the user wants through the query, These chunks will be fetched and given to the LLM as context which it should refer to give its response.
4. **The Final Response:** The user query, system prompt and the context information, together will be sent to the LLM (Google Gemini Model) through its API. The LLM will use the provided context information and accordingly answer the user query, thereby achieving the desired goal.

***PROCESS AND STAGES OF EXECUTION***

1. **Document Upload and Chunking**
   * **The user uploads a document.**
   * **The Document Chunker function processes the document by reading its contents and dividing it into chunks of predefined sizes.**
2. **Vector Embedding Generation**
   * **Each chunk of text is passed through the Google Text-Embedding-004 model.**
   * **The model converts the text into vector embeddings (floating-point vectors of size 768).**
3. **Storing Embeddings in Pinecone**
   * **The generated vector embeddings are stored in the Pinecone Vector Database.**
   * **This allows for efficient similarity searches when processing queries.**
4. **User Query Processing**
   * **The user submits a query.**
   * **The query text is also converted into a vector embedding using the Google Text-Embedding-004 model.**
5. **Context Retrieval Using Similarity Search**
   * **The query vector is compared against stored vector embeddings in Pinecone.**
   * **The most relevant document chunks (with high similarity scores) are retrieved.**
6. **Generating the Response Using LLM**
   * **The system prompt, retrieved context information, and the user query are sent to the LLM (Google Gemini Model).**
   * **The LLM processes the input and generates a response based on the provided context.**
7. **Returning the Final Answer**
   * **The generated response is displayed to the user.**
   * **The process ensures that responses are based only on the uploaded documents, improving accuracy.**