**SAMPLE SET**

**Part 2: Interactive QA Bot Interface**

**Overview**

This project aims to develop an interactive Question Answering (QA) bot that allows users to upload PDF documents and ask questions based on their content. The bot integrates a Retrieval-Augmented Generation (RAG) model, using Pinecone for document retrieval and the Cohere API for answer generation.

**Approach**

1. **Architecture Design**:
   * **Frontend**: Streamlit interface for document uploads and user queries.
   * **Backend**: Pinecone as a vector database for similarity search and Cohere API for generating responses.
2. **Key Components**:
   * **Text Extraction**: Utilized PyPDF2 to extract text from PDFs.
   * **Embedding Generation**: Employed SentenceTransformer for creating vector embeddings.
   * **Vector Storage**: Stored embeddings in Pinecone for efficient retrieval.
   * **Answer Generation**: Used Cohere API to generate answers based on retrieved content.

**Implementation Steps**

1. **Pinecone Initialization**: Checked and created an index for storing embeddings.
2. **File Upload**: Implemented a Streamlit file uploader for PDF documents.
3. **Embedding Process**: Generated and upserted embeddings into Pinecone.
4. **Query Handling**: Generated query embeddings and retrieved relevant document segments.
5. **Answer Generation**: Combined retrieved segments to formulate prompts for the Cohere API.
6. **Display Results**: Showed retrieved text and generated answers in the Streamlit interface.

**Challenges and Solutions**

1. **PDF Variability**: Addressed inconsistent text extraction by implementing checks for empty text.
2. **Embedding Management**: Prevented ID conflicts by using unique identifiers for each document.
3. **Coherence in Responses**: Improved response quality by refining context aggregation and prompt formulation.
4. **Performance**: Enhanced retrieval speed by limiting the number of segments retrieved.

**Conclusion**

The project effectively combines document retrieval and generation to create a functional interactive QA bot. Future enhancements may include better error handling, support for various document types, and further user experience improvements in the interface.