**Solution Approach:**

**Document Question Answering:**

* For Question Answering RAG (Retreival Augmented Generation) is performed using Lang chain Framework
* Pinecone Vector database is utilized for storing document vector
* Open AI GPT-3.5 is used for response generation

**Named Entity Recognition (Medical Entities):**

* Prompt Engineering is implemented for extracting medical entities.
* Open AI GPT-3.5 is used for response generation

**Summarization:**

* Used a Pretrained T-5 model for medical summarization and further finetuned it using additional clinical data to get more accurate and precise summaries.
* Used that new Finetuned model to generate summary.

**Project Overview**

* **Objective:** Develop a Flask-based API for processing medical documents, answering user queries, extracting medical entities, and summarizing content.
* **Key Components:**
  + Flask: Web framework for building the API.
  + Langchain: Library for handling document loading, text splitting, embeddings, and retrieval.
  + Pinecone: Service for managing embeddings and performing document retrieval.
  + OpenAI GPT-3.5 Turbo: For question answering and named entity recognition.
  + Finetuned Transformer Model Used for document summarization.

**Prerequisites**

* **Dependencies:**
  + Install required Python packages using **pip install -r requirements.txt**.
  + Ensure Flask, Langchain, Pinecone, Transformers, and Werkzeug are installed.
* **API Keys:**
  + Set up Pinecone and OpenAI API keys as environment variables.

**Folder Structure**

* **uploads:** Directory for storing uploaded medical documents.

**API Endpoint**

**/answer Endpoint**

* **URL:** **http://127.0.0.1:5000/answer**

**Request Parameters**

* **file**: Medical document (allowed extensions: .docx)
* **query**: User query

**Response**

* JSON response containing:
  + **answer\_qa**: Answer to the user's query using GPT-3.5 Turbo.
  + **ner\_result**: Extracted medical entities using GPT-3.5 Turbo.
  + **summary\_result**: Document summary

**Usage Guidelines**

* **Document Format:**
  + Ensure medical documents are in the correct format (DOCX).

**Run the API**

* Execute **python app.py** to run the Flask application.