



Smart Medical Video AI Assistant

**By:
Saleh Al-Malki
Mohammed Bunahyah**

Project Goal:

To develop an intelligent assistant that can analyze YouTube videos (mainly medical content), transcribe their audio, store the extracted text in a vector database, and answer user questions related to the video through an interactive interface using OpenAI models.

Main Components of the Project:

1. Library Installation

Purpose: Install all required dependencies:

- yt-dlp for downloading YouTube videos.
- openai-whisper for speech-to-text.
- youtube-transcript-api for subtitles.
- langchain, chromadb, sentence-transformers for vector storage and retrieval.
- gradio for building the UI.
- langsmith for debugging and tracing.

2. Environment Setup & Library Imports

Purpose: Import necessary Python libraries such as:

- os, tempfile, pydub.
- Configure OpenAI API keys and environment variables for LangChain and LangSmith.

3. YouTube Video Processing

Component: YouTubeProcessor

Function:

- Extracts the video ID from a YouTube link.
- Downloads the audio as .mp3 using yt-dlp.
- If audio is unavailable, tries to fetch subtitles using the YouTubeTranscriptApi.

4. Speech-to-Text with Whisper

Component: TranscriptionProcessor

Function:

- Loads OpenAI's Whisper model ("base").
- Transcribes audio files to text.
- Saves the output into .txt files.

5. Vector Database Construction

Component: VectorDatabaseManager

Function:

- Splits the transcript into smaller chunks using RecursiveCharacterTextSplitter.
- Converts text chunks into embeddings using:
 - OpenAI Embeddings (if API key is provided), or
 - Local all-MiniLM-L6-v2 model.
- Stores text vectors and metadata in ChromaDB for fast retrieval.

6. Retriever Setup

Purpose:

- Enables semantic search by retrieving text chunks relevant to the user's question.
- Powered by the previously created vector database.

7. Question Answering System with LangChain

Tool Used: RetrievalQA from LangChain

How it works:

- A prompt template is used to format questions and context.
- When a user asks a question:
 - The retriever finds the most relevant chunks from the transcript.
 - The model (ChatGPT) generates an answer based on those chunks.

8. User Interface with Gradio

Features:

- Input field for YouTube link.
- Button to analyze the video.
- Input field for user questions.
- Area for displaying AI-generated answers.

Workflow:

- User inputs a video link → Audio is downloaded and transcribed → Text is embedded and stored → User can ask questions → Answers are generated from the video content.

9. Tools and Technologies Used

| Category | Tool/Library | Purpose |
|---------------------|--------------------------------|------------------------------------|
| Speech Processing | OpenAI Whisper | Transcribe audio to text |
| Language Model | OpenAI ChatGPT | Generate answers |
| Vector Database | ChromaDB | Store and retrieve vectorized text |
| Embedding Models | OpenAI / Sentence-Transformers | Convert text to embeddings |
| YouTube Handling | yt-dlp | Download audio from YouTube |
| Subtitle Extraction | YouTube Transcript API | Get subtitles as text |
| Interface | Gradio | Build interactive UI |
| Monitoring | LangSmith | Track and debug flows |

10. Final Output

An interactive AI assistant that:

- Takes a YouTube medical video link.
- Converts the video content into searchable text.
- Lets users ask questions related to the video.
- Answers intelligently using retrieved context and a powerful LLM.