الأكاديمية السعودية الرقمية
SAUDI DIGITAL ACADEMY

IRON HACK

Project Documentation: VidTalk

AI-Powered YouTube Learning Assistant

By :

Azzam AlHarbi

Ahmed AlSubhi

For :

AI Engineering Bootcamp

Contents

Introduction :

This project is a comprehensive AI-powered learning assistant that transforms YouTube videos into structured, queryable knowledge. The system downloads educational YouTube videos, extracts and transcribes audio, chunks and indexes the text, and then builds a retrieval-based Q&A and quiz generation system with conversational agents.
It leverages tools like yt_dlp, ffmpeg, OpenAI Whisper, LangChain, FAISS, and OpenAI GPT models to build a self-contained intelligent tutor capable of understanding, retrieving, and testing user knowledge.

Setup Instructions

1. Clone the repository.
2. Install dependencies using pip install -r requirements.txt.
3. Add your OpenAI API key and ideogram.ai as environment variable or .env file.
4. Run the app with app.py.

Usage Guide

1. Launch the app.
2. Select input: YouTube link, upload file, or search.
3. Ask questions or request quizzes/mind maps.
4. Use voice or text input.

Repository Structure :

```
Project_VidTalk/
 ├── static/
|    ├── styles.css
|   └── script.js
|
 ├── templates/
|   └── index.html
|
 ├── app.py
 ├── core.py
 ├── requirements.txt
 ├── Dockerfile
 ├── runtime.txt
 ├── Project Documentation.pdf
└── README.md
```

Project Objectives :

- Convert YouTube videos into a searchable knowledge base.
- Allow learners to query content as if asking a teacher.
- Generate multiple-choice quizzes to validate understanding.
- Create visual mind maps for enhanced comprehension.
- Support learning via YouTube playlist links and keyword-based search.
- Enable real-time audio responses for accessibility.

Key Challenges :

1. **Audio Transcription Accuracy**: Achieving high transcription quality across diverse accents, background noise, and low-quality audio.
2. **Contextual Chunking**: Designing chunking strategies to maintain semantic context, using overlapping text segments.
3. **Scalable Retrieval**: Building FAISS vector indices that handle multiple videos efficiently and allow low-latency retrieval.
4. **Multi-User Memory Management**: Implementing user-specific memory buffers to distinguish learning sessions across users and maintain contextual continuity.
5. **Quiz Reliability**: Ensuring generated quizzes are accurate, relevant, and strictly based on the embedded content.
6. **Video Duration Validation**: Restricting processing to videos under 30 minutes using the is_valid_duration(url) function to ensure quick processing and manageable workloads.

System Workflow :

Input Options:

- Search YouTube by topic
- Paste a YouTube playlist link
- Paste a single YouTube video link
- Upload a video file Video Processing:
- Download audio using yt_dlp
- Convert to .mp3 using ffmpeg
- Transcribe audio using OpenAI Whisper Text Chunking:
- Split transcript into overlapping chunks using LangChain's

RecursiveCharacterTextSplitter Embedding:

- Embed chunks via OpenAIEmbeddings
- Store vectors in a FAISS index Interactive Features:
- Answer questions with RetrievalQA
- Quiz generation
- Mind map creation
- Text-to-speech responses

Core Functions Explained :

download_audio(youtube_url, output_path)

Downloads the best available audio from a YouTube link using yt_dlp.

extract_audio_with_ffmpeg(video_path, audio_output_path)

Extracts audio from a video file using ffmpeg, useful for uploaded videos.

extract_youtube_links(text)

Uses regex to extract YouTube links from search results.

transcribe_audio_openai(file_path)

Transcribes .mp3 audio files using OpenAI's Whisper (whisper-1 model).

split_text(text, chunk_size, chunk_overlap)

Splits text into manageable overlapping chunks for better context retention.

process_videos_and_store(links)

End-to-end processing: download → transcribe → chunk → embed → store.

get_agent_for_session(user_id)

Loads an agent that remembers each user's conversations and can retrieve information from FAISS vectorstore.

learning_function(query)

Uses the Retrieval QA chain to return relevant answers and source documents from embedded video chunks.

generate_quiz_from_text(_)

Asks the LLM to create 10 multiple-choice or true/false questions from the embedded video content.

get_quiz_answers(_)

Returns the last generated quiz and its answers.

generate_speech()

Uses gpt-4o-mini-tts model from OpenAI to read the responses loudly.

Agent Design :

Agent Type

The system uses LangChain's initialize_agent method with the ConversationalAgent type. This agent is equipped to handle open-ended dialogue while selectively calling custom tools based on user input. It is designed to maintain conversational flow while using retrieval-based knowledge grounded in YouTube content.

LLM Used

The agent uses OpenAI's GPT-4o as the core Large Language Model (LLM) to interpret user inputs, generate responses, and produce questions or summaries. GPT-4o provides higher reasoning ability and better comprehension across complex educational topics.

Memory

The agent uses ConversationBufferMemory to store and recall the history of user queries and answers throughout the session. This enables the assistant to respond in a contextual and coherent manner, building a natural tutoring experience.

LangChain Tools and Descriptions :

Tool Name	Description
LearningTool	Retrieves factual answers from transcribed YouTube content using RetrievalQA.
QuizTool	Generates multiple-choice or true/false quizzes based on embedded video knowledge.
QuizAnswerTool	Retrieves the most recent quiz and its corresponding answers.
MindMapImageTool	Summarizes content and generates a mind map using an external image API.
SummarizeTool	Creates short summaries of video content to give learners quick overviews.
YouTubeSearchTool	Searches YouTube for videos based on user-entered topics.

Extra Functions :

Visual Mind Map Generation

Summarizes indexed chunks and creates a visual mind map using a third-party image generation API (ideogram.ai). This enhances comprehension.

Text-to-Speech

OpenAI gpt-4o-mini-tts model is used to convert text answers into speech using the function generate_speech.

Speech-To-Text

Uses Javascript's webkitSpeechRecognition() function.

Example Use Case :

1. User selects **Option 1** and enters the topic: *"introduction to machine learning"*.
2. Tool uses YouTubeSearchTool to fetch top 3 video links.
3. Each video is downloaded, transcribed, and stored.
4. User asks: *"What is supervised learning?"*
5. The system answers with relevant content and citations from video chunks.
6. User says: *"Test me"*
7. A quiz is generated with 10 questions.
8. Optionally, a mind map is generated and shared via a link.

Technologies Used :

Component	**Tool/Library**
Audio Download	yt_dlp
Audio Transcription	OpenAI Whisper
Text Chunking	LangChain Text Splitter
Embedding	OpenAI Embeddings
Vector Store	FAISS
Q&A	LangChain RetrievalQA

Agent System	LangChain Tools & Agents
Visualization	External API (e.g., ideogram.ai ‘)
Component	**Tool/Library**
Text-to-Speech	OpenAI gpt-4o-mini-tts
YouTube Search	langchain_community.tools.YouTubeSearchTool

Future Improvements :

- Add multilingual support for Arabic, Spanish, etc.
- Support long lectures and videos.

Conclusion :

VidTalk bridges the gap between passive YouTube watching and active learning by turning educational content into interactive experiences. Through AI agents, visual tools, and structured assessments, it helps learners better understand and retain complex material in a conversational way.