# 목표

* 사용자의 자연어 질의에 해당하는 Youtube 영상의 구간들을 찾아 줌
* 각 Video별, 구간별 설명 추가, 자동 영상 편집
* 하나의 mp4 파일로 제작
* 해당 영상 구간들을 선택적으로 play 할 수 있도록 Web에 개시 (미 구현)

# 활용 기술

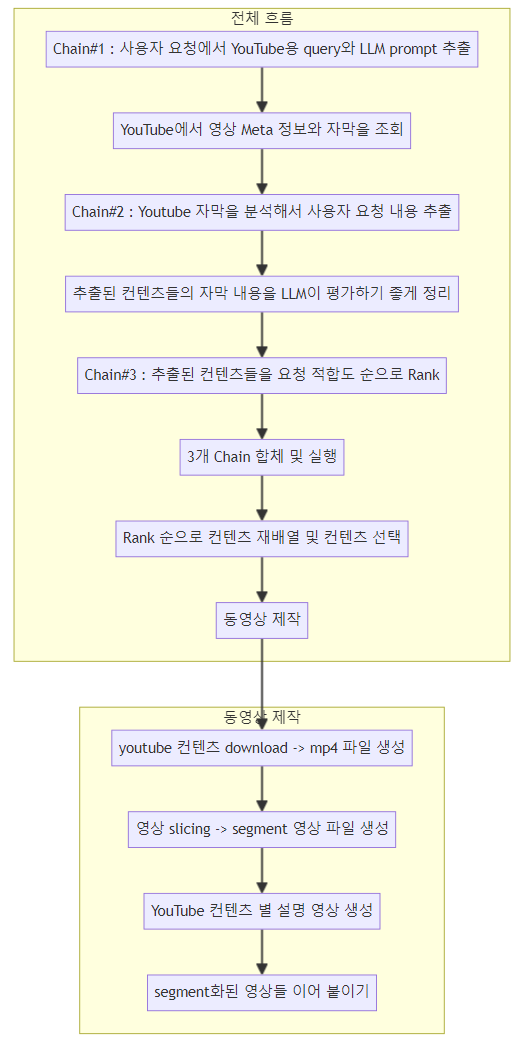
* Langchain
* Langsmith
* Ffmpeg, OpenCV
* TTS (pyttsx3)

# LLM 적용 내용

* YouTube은 Keywork 기반의 검색이며 검색된 컨텐츠가 실제 사용자 요청에 적합하지 않은 경우가 많았음. 그래서 아래와 같이 절차로 실행됨

1. Chain #1 : 사용자의 자연어 질의에서 YouTube 검색어를 추출
2. YouTube 컨텐츠 조회 (필요한 양보다 2배수 조회)
3. Chain #2 : 각각의 컨텐츠에서 질의를 충족하는 구간들을 찾아 잘라냄
4. Chain #3 : 컨텐츠들 간 질의 충족도 순위를 먹임
5. 순위가 높은 n개만 선택함

# 기능 구성



**1. 환경 변수 Load**

- API 키와 관련된 정보를 로드하여 보안 강화.

**2. llm 객체 ChatGPT 생성**

- OpenAI의 ChatGPT 모델 생성.

- 모델의 온도를 0.3으로 설정하여 응답의 일관성 유지.

**3. Chain#1 : 사용자 요청에서 YouTube용 query와 LLM prompt 추출**

- 사용자 요청을 분석하여 YouTube 검색어와 LLM 질의를 추출.

- `YoutubeQuery` Pydantic 모델과 `PydanticOutputParser`를 사용.

**4. YouTube에서 영상 Meta 정보와 자막을 조회**

- YouTube API를 사용해 검색 결과와 자막 정보를 조회.

- `ContentManager`, `Content`, `YouTube` 클래스를 사용하여 검색된 영상의 자막을 저장.

**5. Chain#2 : Youtube 자막을 분석해서 사용자 요청 내용 추출**

- 자막에서 사용자 요청에 맞는 내용을 추출하고 요약 및 구간 정보를 생성.

- `Segment`와 `Summary` Pydantic 모델과 `PydanticOutputParser`를 사용.

**6. 추출된 컨텐츠들의 자막 내용을 LLM이 평가하기 좋게 정리**

- `flat\_text` 함수를 정의하여 선택된 컨텐츠의 요약 및 구간별 자막을 텍스트로 추출.

**7. Chain#3 : 추출된 컨텐츠들을 요청 적합도 순으로 Rank**

- 각 컨텐츠의 적합성을 평가하고 순위를 매김.

- `CommaSeparatedListOutputParser`와 `PromptTemplate`를 사용하여 평가 프롬프트 작성.

**8. 3개 Chain 합체 및 실행**

- 첫 번째, 두 번째, 세 번째 체인을 합쳐서 전체 체인을 구성.

- `RunnablePassthrough`, `RunnableLambda`를 사용해 각 단계의 출력을 연결.

**9. Rank 순으로 컨텐츠 재배열 및 컨텐츠 선택**

- 평가된 컨텐츠를 순위에 따라 재배열하고 필요한 개수만큼 선택.

- `reorder\_summaries` 함수를 정의하여 재배열.

**10. 동영상 제작**

- 작업 ID 생성.

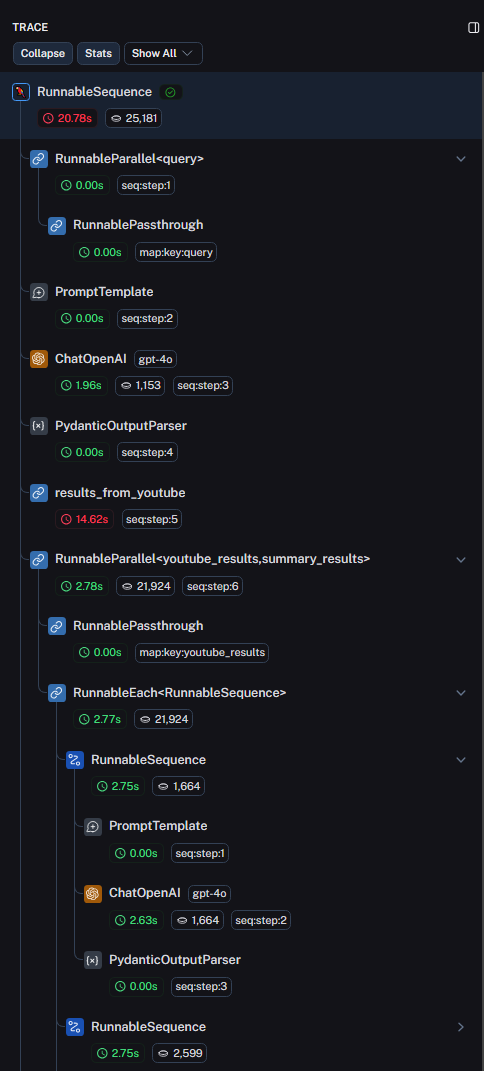
- YouTube 영상을 다운로드하고, 영상을 구간별로 나눔.

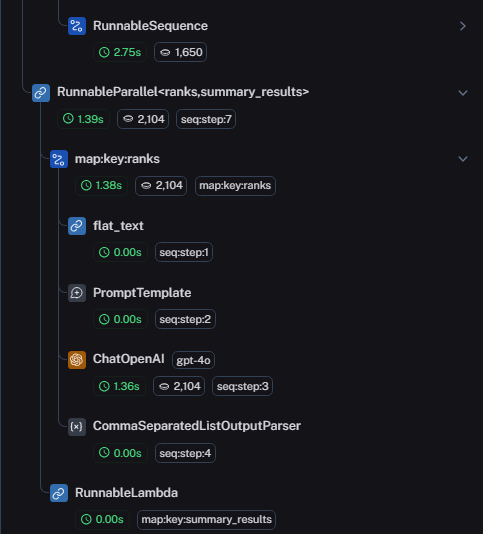
- 설명 영상 및 구간별 설명 영상을 생성하고, 영상에 텍스트를 삽입.

- 구간별 영상을 하나의 영상으로 합침.

- 최종적으로 완성된 영상을 출력 파일로 저장.

<LangSmith Trace>





# 프롬프트 예시

1. Query 생성 Prompt (첫번째 chain의 prompt)

"""

System :

너는 사용자가 원하는 youtube 컨텐츠를 찾아주는 AI야.

사용자의 query를 분석해서 다음 3가지 정보를 생성해줘.

 - 사용자가 원하는 YouTube 컨텐츠를 찾을 수 있도록 YouTube 검색어를 만들어줘.

 - 찾아야 할 컨텐츠가 한글인 경우 'ko' 영어인 경우 'en'이라고 표시해줘.

 - YouTube 컨텐츠가 담고있는 Text 정보에서 사용자가 원하는 부분을 찾기 위한 자연어 질의를 만들어줘

   예를 들어, 채널명이나 필터 정보 등 YouTube 검색에 사용되는 내용은 제외해야되.

검색어를 만들때 tips 를 참고해.

tips :

{tips}

query :

{query}

Format :

{format}

"""

1. Youtube 클립 추출 prompt (두번째 chain의 prompt)

"""

System :

너는 동영상 편집자이다.

먼저 captions 내용을 100자 이내로 요약해.

요약 결과는 네가 시청자에게 동영상을 소개하는 말투여야 해.

그리고 captions에서 사용자 query를 잘 설명하고 있는 내용을 뽑아내고

그 내용이 포함된 구간을 찾아내.

여기서 구간이란 captions 내의 하나의 세부 구간이 아니라 연속되는 여러개의 세부 구간의 합을 의미한다.

예)

세부 구간 1 :

  start : 112.0, duration : 5.0

세부 구간 2 :

  start : 117.0, duration : 4.0

세부 구간 3 :

  start : 121.0, duration : 6.0

세부 구간 4 :

  start : 127.0, duration : 3.0

네가 찾은 하나의 구간이 세부 구간 1 ~ 4 인 경우

구간의 start, duration은 다음과 같이 계산 하면 된다.

  start : 112.0 (세부 구간 1 의 start)

  duration : 18.0 ( 5.0 + 4.0 + 6.0 + 3.0, 세부 구간 4개 duration 의 합계)

해당 구간의 내용을 20자 이내로 요약해서 구간 제목을 만들어.

그 구간은 없을 수도 있고 여러개 일수도 있다.

그 구간이 4개 이상이라면 가장 적합한 것 3개만 찾아내라.

video\_id 값은 동영상을 식별하는 id이다. 변경하지 말고 그대로 출력하라.

channel\_name은 동영상을 소유한 채널 이름이다. 변경하지 말고 그대로 출력하라.

query :

{query}

video\_id :

{video\_id}

channel\_name :

{channel\_name}

captions :

{captions}

Format :

{format}

"""

1. 컨텐츠들이 사용자 요청에 적합한 지 평가 (세번째 chain의 prompt)

System :

당신은 특정 질의(query)에 대해 여러 비디오(video\_id)의 답변 적합성을 평가하고 순위를 매기는 AI입니다.

llm\_query에는 질의 내용이 포함되어 있으며, video\_text에는 각 video\_id에 해당하는 답변(text)이 저장되어 있습니다.

당신의 임무는 다음과 같습니다:

- 각 비디오의 답변을 질의에 대한 적합성 기준으로 평가합니다.

- 평가 결과를 바탕으로 가장 적절한 답변을 제공한 비디오부터 순서대로 video\_id 목록을 작성합니다.

최종적으로, 적합성이 높은 순서대로 정렬된 video\_id 목록을 출력해 주세요.

llm\_query :

{llm\_query}

video\_text :

{video\_text}

Format :

{format}

1. 사용자 입력 쿼리

input\_query = """

한국인 중에 기네스북에 오른 사람들은 누가 있으면 어떤 분야에서 올랐는지 알려줘

"""

# 실행 결과

1. 사용자 입력 쿼리

input\_query = """

한국인 중에 기네스북에 오른 사람들은 누가 있으면 어떤 분야에서 올랐는지 알려줘

"""

1. 첫번째 Chain의 completion 결과

# youtube\_query : YouTube 검색용 query

# llm\_query : LLM용 query (자막 내용으로 In-context learning방식)

# language : YouTube 검색 시 자막의 언어

{

  "output": {

    "youtube\_query": "한국인 기네스북 기록",

    "llm\_query": "한국인 중 기네스북에 오른 사람들과 그들이 어떤 분야에서 기록을 세웠는지",

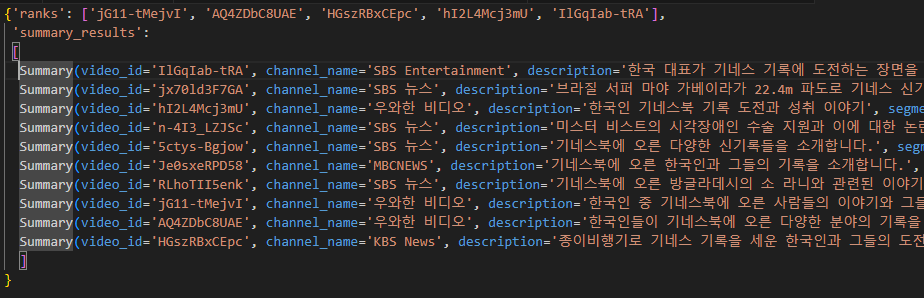
    "language": "ko"

  }

}

1. Youtube 컨텐츠 추출 및 Ranking 정보 (Full Chain 수행 결과)

* 선택된 youtube 컨텐츠(summary\_results)와 적합성 Rank(ranks)



1. Segment 화된 video 파일들(Create Explanation Videos for Each Segment and Insert Text in Videos 단계 수행 결과)

# <UUID>\_<video\_id>\_c.mp4 : video 표지 영상

# <UUID>\_<video\_id>\_n\_c.mp4 : video n번 segment 표지 영상

# <UUID>\_<video\_id>\_n\_t.mp4 : video n번 segment 영상

[

 '3593b521-fdcc-42c1-9040-f6c9cda481cd\_Je0sxeRPD58\_c.mp4',

 '3593b521-fdcc-42c1-9040-f6c9cda481cd\_Je0sxeRPD58\_0\_c.mp4',

 '3593b521-fdcc-42c1-9040-f6c9cda481cd\_Je0sxeRPD58\_0\_t.mp4',

 '3593b521-fdcc-42c1-9040-f6c9cda481cd\_5ctys-Bgjow\_c.mp4',

 '3593b521-fdcc-42c1-9040-f6c9cda481cd\_5ctys-Bgjow\_0\_c.mp4',

 '3593b521-fdcc-42c1-9040-f6c9cda481cd\_5ctys-Bgjow\_0\_t.mp4',

 '3593b521-fdcc-42c1-9040-f6c9cda481cd\_5ctys-Bgjow\_1\_c.mp4',

 '3593b521-fdcc-42c1-9040-f6c9cda481cd\_5ctys-Bgjow\_1\_t.mp4',

 '3593b521-fdcc-42c1-9040-f6c9cda481cd\_5ctys-Bgjow\_2\_c.mp4',

 '3593b521-fdcc-42c1-9040-f6c9cda481cd\_5ctys-Bgjow\_2\_t.mp4',

 '3593b521-fdcc-42c1-9040-f6c9cda481cd\_jx70ld3F7GA\_c.mp4',

 '3593b521-fdcc-42c1-9040-f6c9cda481cd\_jx70ld3F7GA\_0\_c.mp4',

 '3593b521-fdcc-42c1-9040-f6c9cda481cd\_jx70ld3F7GA\_0\_t.mp4',

 '3593b521-fdcc-42c1-9040-f6c9cda481cd\_IlGqIab-tRA\_c.mp4',

 '3593b521-fdcc-42c1-9040-f6c9cda481cd\_IlGqIab-tRA\_0\_c.mp4',

 '3593b521-fdcc-42c1-9040-f6c9cda481cd\_IlGqIab-tRA\_0\_t.mp4',

 '3593b521-fdcc-42c1-9040-f6c9cda481cd\_IlGqIab-tRA\_1\_c.mp4',

 '3593b521-fdcc-42c1-9040-f6c9cda481cd\_IlGqIab-tRA\_1\_t.mp4',

 '3593b521-fdcc-42c1-9040-f6c9cda481cd\_IlGqIab-tRA\_2\_c.mp4',

 '3593b521-fdcc-42c1-9040-f6c9cda481cd\_IlGqIab-tRA\_2\_t.mp4'

 ]

1. 하나의 Video 파일로 이어 붙이기

기네스북\_한국인.mp4

* 1. Video 컨텐츠 별 간지 영상 예



* 1. segment 별 간지 영상 예



* 1. segment 영상 Capture

