**TA's 솔루션 해설**

* <https://github.com/fullstack-gpt-python/assignment-17/blob/main/app.py>
* <https://assignment-17-ypobm669xteqnbodqvatlg.streamlit.app/>

1. OpenAI API 키 입력 받기
   * st.text\_input을 사용하여 사용자의 OpenAI API 키를 입력받습니다.
   * 입력받은 API 키를 ChatOpenAI와 OpenAIEmbeddings 클래스를 사용할 때 openai\_api\_key 매개변수로 넘깁니다.
2. 사이트맵 로드
   * SitemapLoader를 사용하여 Cloudflare의 사이트맵을 로드합니다.
   * 이때, 특정 URL 규칙으로 필터링하여 해당 문서만 파싱합니다. (filter\_urls 이용)
   * 파싱된 문서에서 불필요한 부분(header와 footer)을 제거합니다. (parsing\_function 이용)
   * 사이트맵 로드에 대한 자세한 설명은 [공식 문서 (Sitemap)](https://python.langchain.com/v0.2/docs/integrations/document_loaders/sitemap/)를 참고하세요.
3. 임베딩 및 벡터 스토어 생성
   * 로드된 문서의 임베딩 과정을 거치고 FAISS를 사용하여 벡터 스토어를 생성합니다.
   * 벡터 스토어의 as\_retriever 메소드를 이용하여 retriever를 생성하고, 이것을 체인에서 사용합니다.
   * ([관련 공식 문서 (Vector store-backed retriever)](https://python.langchain.com/v0.1/docs/modules/data_connection/retrievers/vectorstore/))
4. 질문에 대한 답변 및 메시지 저장
   * 사용자가 질문을 하면 생성된 retriever와 적절한 프롬프트를 이용하여 답변을 생성하고 화면에 표시합니다.
   * 메시지 기록을 유지하기 위해 st.session\_state를 사용하여 메시지를 저장하고 불러옵니다.
   * ([Session State 공식 문서](https://docs.streamlit.io/library/api-reference/session-state))
5. 결론

* 특정 사이트의 사이트맵을 로드하여 문서를 파싱하고, 이에 대한 사용자의 질문에 적절한 답변을 생성하는 챗봇을 구현하는 챌린지였습니다.
* 이번 챌린지는 강의 코드를 약간만 수정하면 쉽게 제출할 수 있었습니다. 따라서 챌린지를 끝내고 바로 다음 과정으로 넘어가기보다는 코드를 분석하고 이해하는 데 집중하는 것이 중요합니다.