**TA's 솔루션 해설**

* <https://github.com/fullstack-gpt-python/assignment-16/blob/main/app.py>
* <https://assignment-16-jyjgz9gtyuywgwrq3dbaan.streamlit.app/>

1. OpenAI API 키 입력 받기
   * st.text\_input을 사용하여 유저로부터 OpenAI API 키를 입력받습니다.
   * 입력받은 API 키는 ChatOpenAI 클래스를 사용할 때 openai\_api\_key 매개변수로 넘깁니다.
2. 퀴즈 생성 함수 스키마 정의
   * 함수 호출에 사용할 퀴즈 생성 함수를 정의합니다.
   * 이 함수는 ChatOpenAI를 초기화할 때 bind 하여 llm이 이 함수를 필수로 사용할 수 있도록 지정합니다.
3. 파일 업로드 및 위키피디아
   * st.selectbox를 사용하여 파일을 기반으로 퀴즈를 만들지, 위키피디아를 기반으로 퀴즈를 만들지 선택할 수 있도록 합니다.
   * 파일을 선택한다면 st.file\_uploader를 사용하여 유저로부터 파일을 입력받고 해당 파일을 CharacterTextSplitter를 이용하여 split 합니다.
   * 위키피디아를 선택한다면 st.text\_input를 사용하여 유저로부터 topic을 입력받고, 해당 주제를 기반으로 문서를 검색합니다.
4. 난이도 선택
   * st.selectbox를 사용하여 난이도를 EASY, HARD 중에 하나를 선택할 수 있도록 합니다.
5. 퀴즈 생성
   * 파일 혹은 위키피디아를 통해 만들어진 문서(docs)와 선택된 난이도(level)을 이용하여 퀴즈를 생성합니다.
   * 함수 호출을 활용하여 만들어진 응답에서 매개변수를 추출하고 json으로 로드하여 파이썬에서 사용할 수 있는 객체로 변환합니다.
   * 만들어진 퀴즈를 st.radio 등을 이용하여 화면에 표시하여 유저가 풀 수 있도록 합니다.
6. 퀴즈 만점일 때 화면에 풍선 표시
   * 퀴즈의 총 개수를 question\_count, 유저가 맞춘 퀴즈의 개수를 success\_count로 합니다.
   * question\_count == success\_count라면 유저가 모든 문제를 맞추었다는 의미이므로 st.balloons를 사용하여 화면에 풍선을 표시합니다.
7. 결론

* 파일 업로드 및 위키피디아 검색을 통해 문서를 로드하고, 로드된 문서를 기반으로 함수 호출을 활용하여 퀴즈를 생성하는 과정을 구현해보는 챌린지였습니다.