RAG 기반 LLM 서비스 구현

Streamlit 기반 웹 서비스 구현하기

충북대학교 의과대학 박 승

충북대학교
CHUNGBUK NATIONAL UNIVERSITY

학습 목표

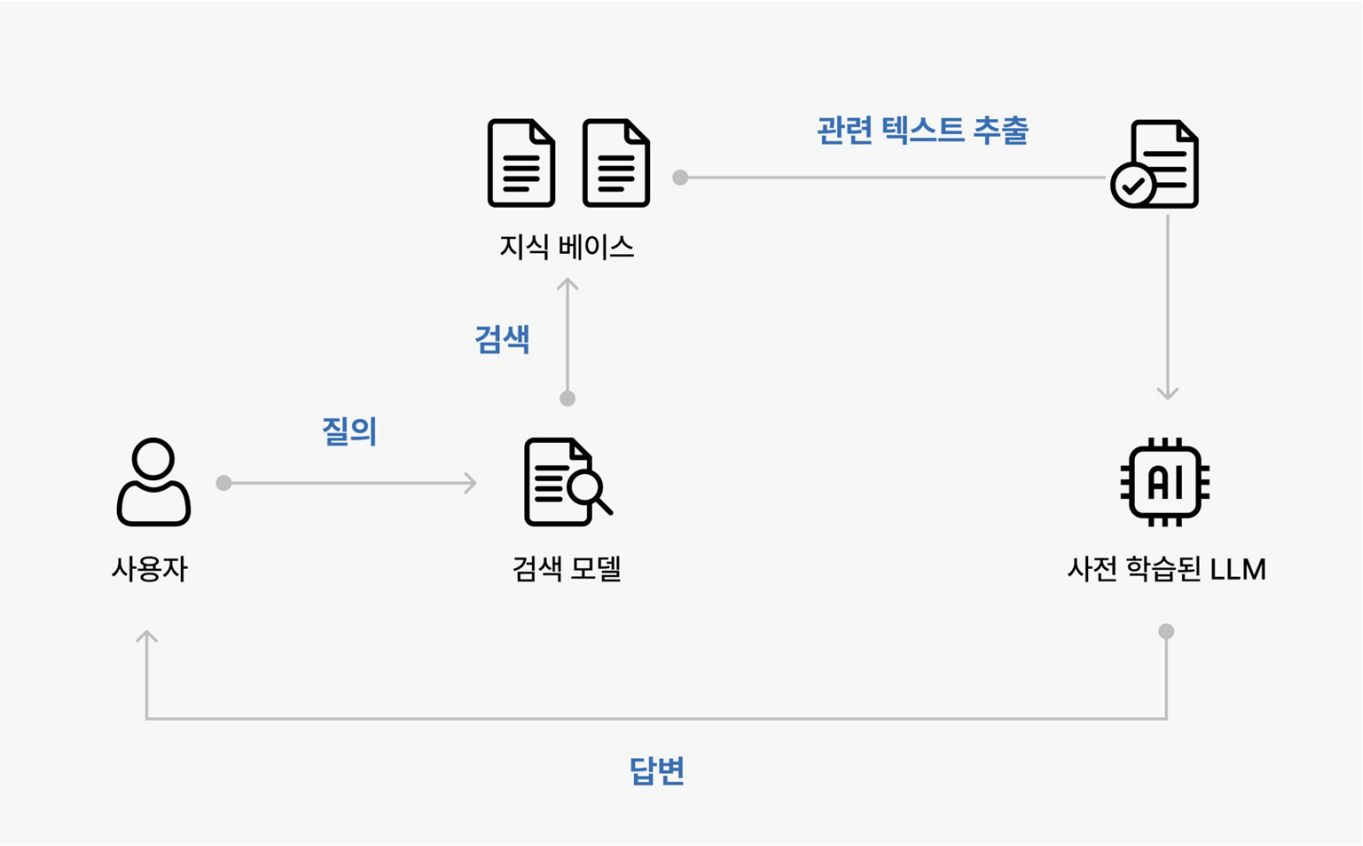
1. 검색 증강 생성 기법에 대해 이해한다.

2. 검색 증강 생성 기법을 활용한 다양한 웹 서비스를 개발한다.

CHUNGBUK
NATIONAL UNIVERSITY

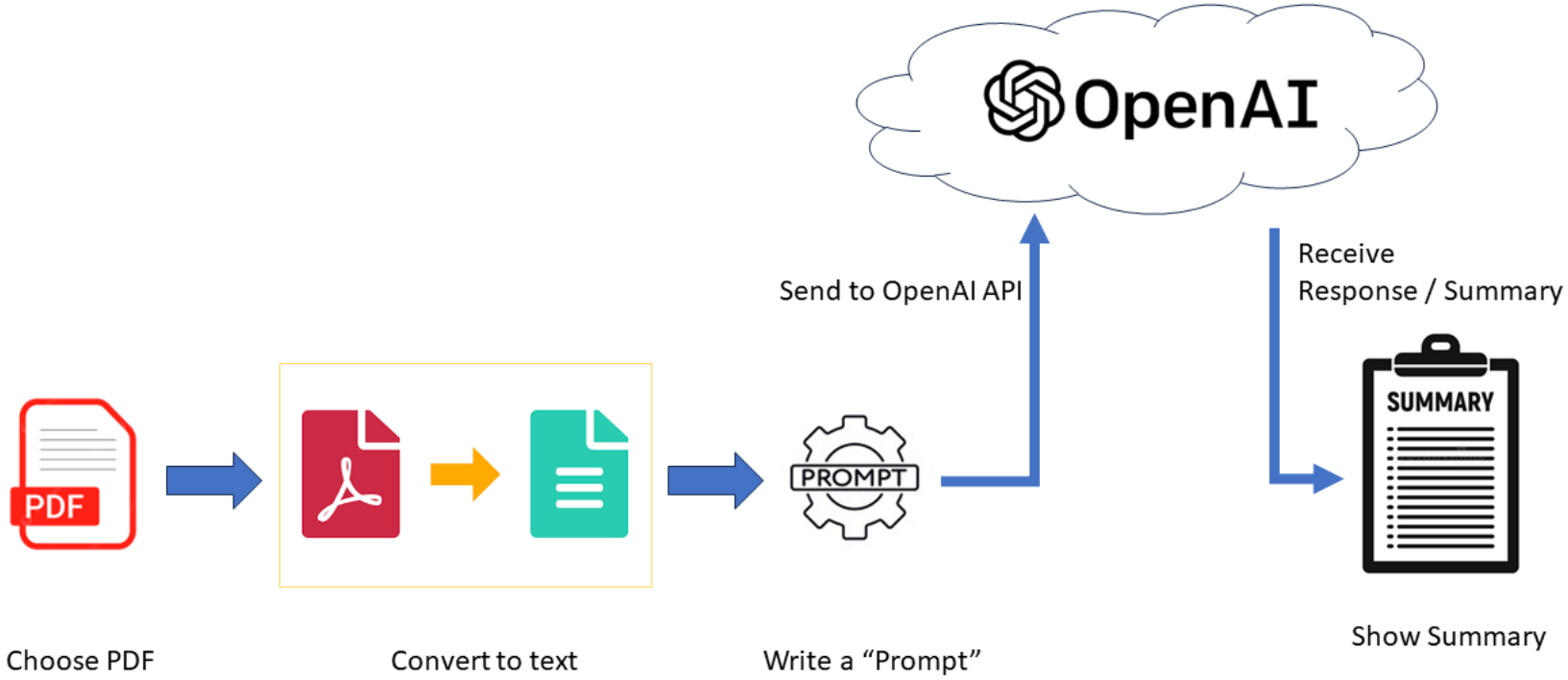
검색 증강 생성 (Retrieval Augmented Generation)

- 대규모 언어 모델(LLM)의 출력을 향상시키기 위해 외부 지식 베이스를 참조하는 기술



검색 증강 생성 (Retrieval Augmented Generation)

- 실습 1) PDF 요약 서비스
  - 메인 프레임워크

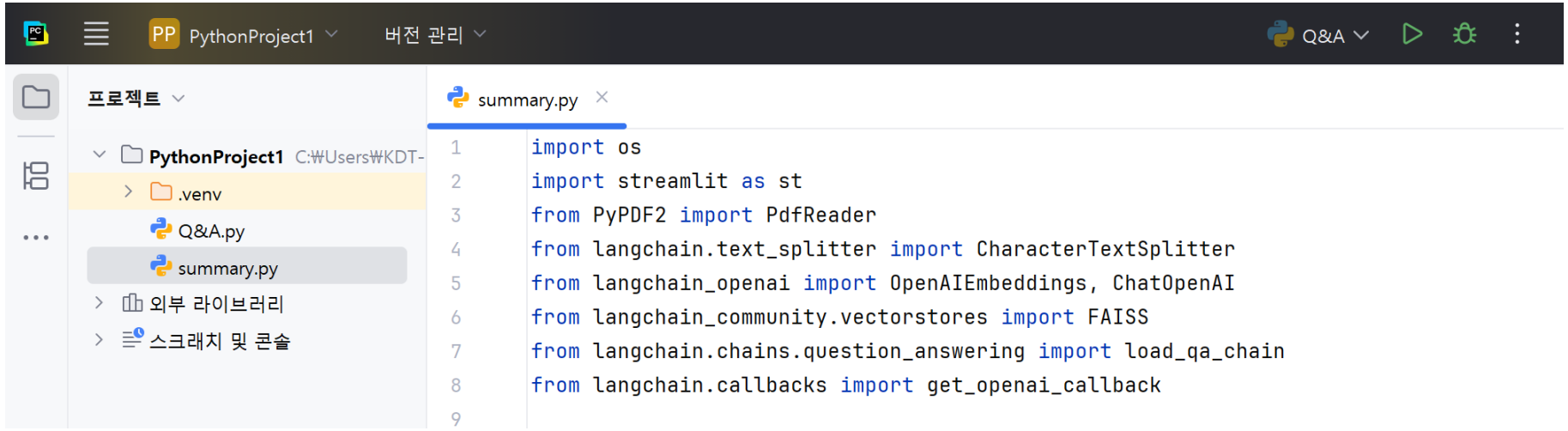


  - 설치: pip install streamlit PyPDF2 langchain langchain-openai langchain-community openai faiss-cpu

검색 증강 생성 (Retrieval Augmented Generation)

실습 1) PDF 요약 서비스

- 1. PyCharm 실행 및 summary.py 파일 생성



검색 증강 생성 (Retrieval Augmented Generation)

실습 1) PDF 요약 서비스

- 2. 텍스트 분할, 임베딩, DB 저장 파트 구현

```
10      # OpenAI API 키 설정
11      os.environ["OPENAI_API_KEY"] = "sk-proj-0V0VDnhzv6-1R2n-1OD1_voC0oi_AlFWZpcb-ItyRvJaf
12
13      # 텍스트를 분할하여 임베딩 후 FAISS 벡터 DB 생성
14      def process_text(text):  1개의 사용 위치
15          text_splitter = CharacterTextSplitter(
16              separator="\n",
17              chunk_size=1000,
18              chunk_overlap=200,
19              length_function=len
20          )
21          chunks = text_splitter.split_text(text)
22          embeddings = OpenAIEmbeddings(model="text-embedding-ada-002")
23          documents = FAISS.from_texts(chunks, embeddings)
24          return documents
```

검색 증강 생성 (Retrieval Augmented Generation)

실습 1) PDF 요약 서비스

- 3. Streamlit 기반 웹 어플리케이션 구현

```python
# Streamlit 메인 애플리케이션
def main():  1개의 사용 위치
    st.title("GPT-4 기반 PDF 요약기")
    st.divider()

    # PDF 파일 업로드
    pdf = st.file_uploader('PDF 파일을 업로드하세요.', type='pdf')

    if pdf:
        # PDF 텍스트 추출
        pdf_reader = PdfReader(pdf)
        text = ""
        for page in pdf_reader.pages:
            text += page.extract_text()

        # 텍스트 처리 후 벡터 DB 생성
        documents = process_text(text)

        # 요약 쿼리
        query = "업로드된 PDF 파일의 내용을 약 3~5 문장으로 요약해주세요."

        if query:
            # 벡터 유사도 기반 문서 검색
            docs = documents.similarity_search(query, k=3)

            # GPT-4 모델로 질의응답 체인 설정
            llm = ChatOpenAI(model="gpt-4", temperature=0.1)
            chain = load_qa_chain(llm, chain_type='stuff')

            # 요약 생성 및 API 사용 비용 출력
            with get_openai_callback() as cost:
                response = chain.run(input_documents=docs, question=query)
                print(cost)

            st.subheader("요약 결과:")
            st.write(response)

# 애플리케이션 실행
if __name__ == '__main__':
    main()
```

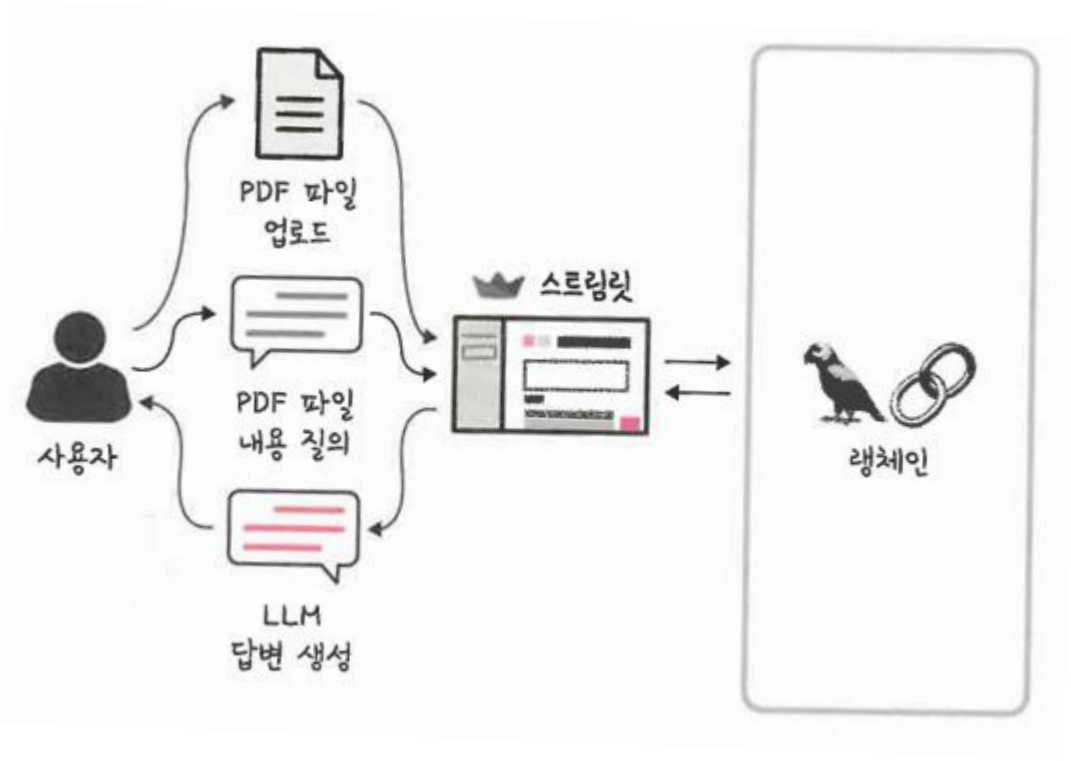
검색 증강 생성 (Retrieval Augmented Generation)

실습 1) PDF 요약 서비스

- 4. PyCharm 터미널 내 streamlit run summary.py 를 통해 실행

검색 증강 생성 (Retrieval Augmented Generation)

- 실습 2) PDF 질의응답 서비스
  - 메인 프레임워크

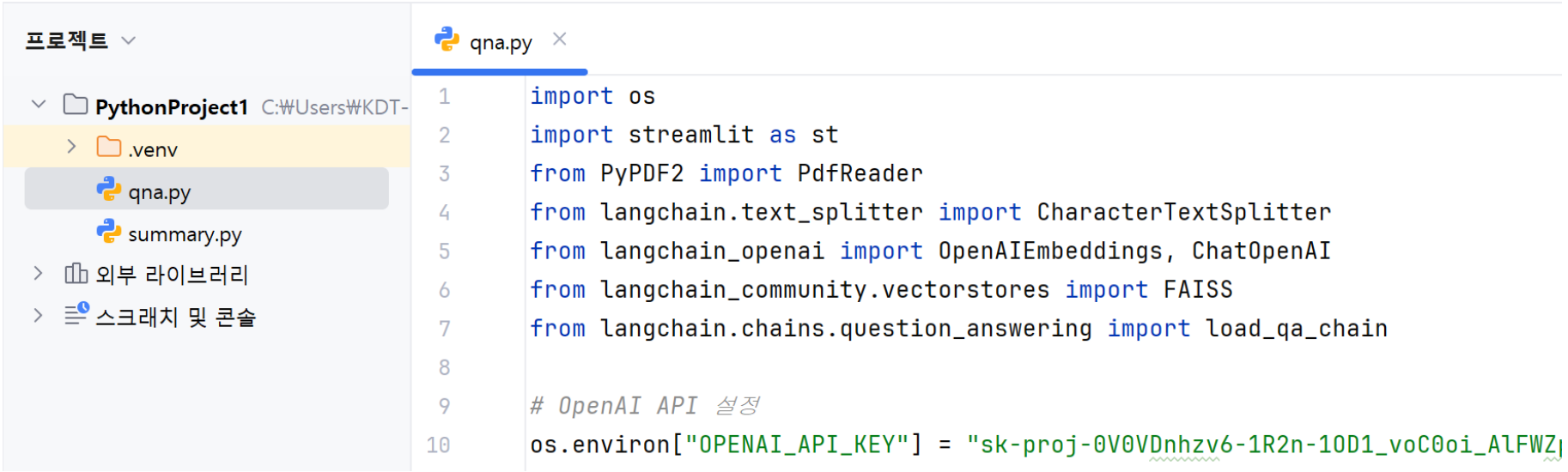


  - 설치: pip install streamlit PyPDF2 langchain langchain-openai langchain-community openai faiss-cpu

검색 증강 생성 (Retrieval Augmented Generation)

실습 2) PDF 질의응답 서비스

- 1. PyCharm 실행 및 qna.py 파일 생성



검색 증강 생성 (Retrieval Augmented Generation)

실습 2) PDF 질의응답 서비스

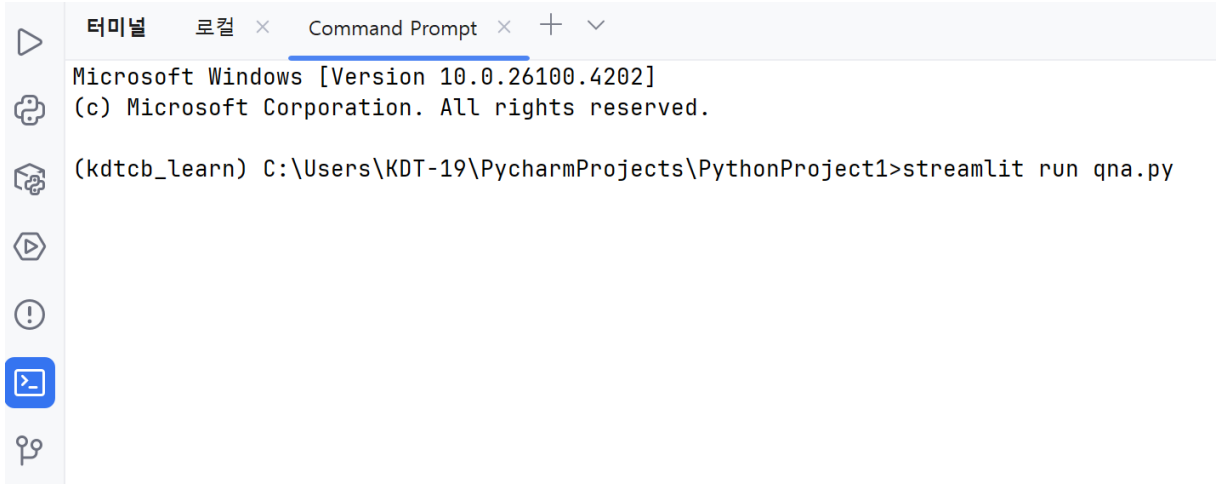
- 2. Streamlit 기반 웹 서비스 구현

```python
# Streamlit 제목 설정
st.title("PDF 기반 GPT-4 질의응답")

# PDF 파일 업로드
pdf = st.file_uploader("PDF 파일을 업로드하세요.", type='pdf')

if pdf:
    # PDF에서 텍스트 추출
    reader = PdfReader(pdf)
    text = ""
    for page in reader.pages:
        text += page.extract_text()

    # 텍스트를 작은 단위(chunk)로 분할
    splitter = CharacterTextSplitter(
        separator="\n",
        chunk_size=1000,
        chunk_overlap=200,
        length_function=len
    )
    chunks = splitter.split_text(text)

    # 임베딩을 이용한 벡터 저장소 생성
    embeddings = OpenAIEmbeddings()
    vector_store = FAISS.from_texts(chunks, embeddings)
```

```python
    # 임베딩을 이용한 벡터 저장소 생성
    embeddings = OpenAIEmbeddings()
    vector_store = FAISS.from_texts(chunks, embeddings)

    # 사용자 질의 입력
    query = st.text_input("PDF 내용에 대해 질문하세요:")

    if query:
        # 유사도 검색으로 관련 chunk 추출
        docs = vector_store.similarity_search(query, k=3)

        # GPT-4 모델로 질의응답 체인 로드
        llm = ChatOpenAI(model_name="gpt-4", temperature=0)
        qa_chain = load_qa_chain(llm, chain_type="stuff")

        # 질의에 대한 답변 생성
        response = qa_chain.run(input_documents=docs, question=query)

        # 결과 표시
        st.subheader("답변 결과")
        st.write(response)
```

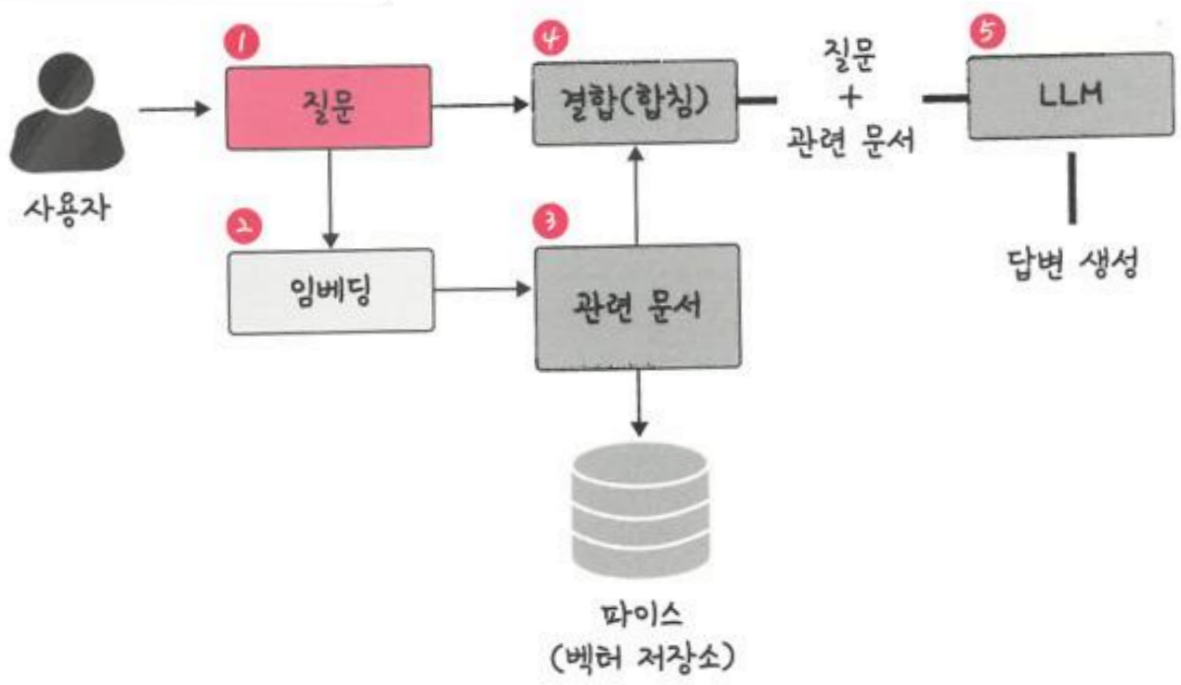
검색 증강 생성 (Retrieval Augmented Generation)

- 실습 2) PDF 질의응답 서비스
  - 3. PyCharm 터미널 내 streamlit run qna.py 를 통해 실행



검색 증강 생성 (Retrieval Augmented Generation)

- 실습 3) PDF 기반 대화형 서비스
  - 메인 프레임워크

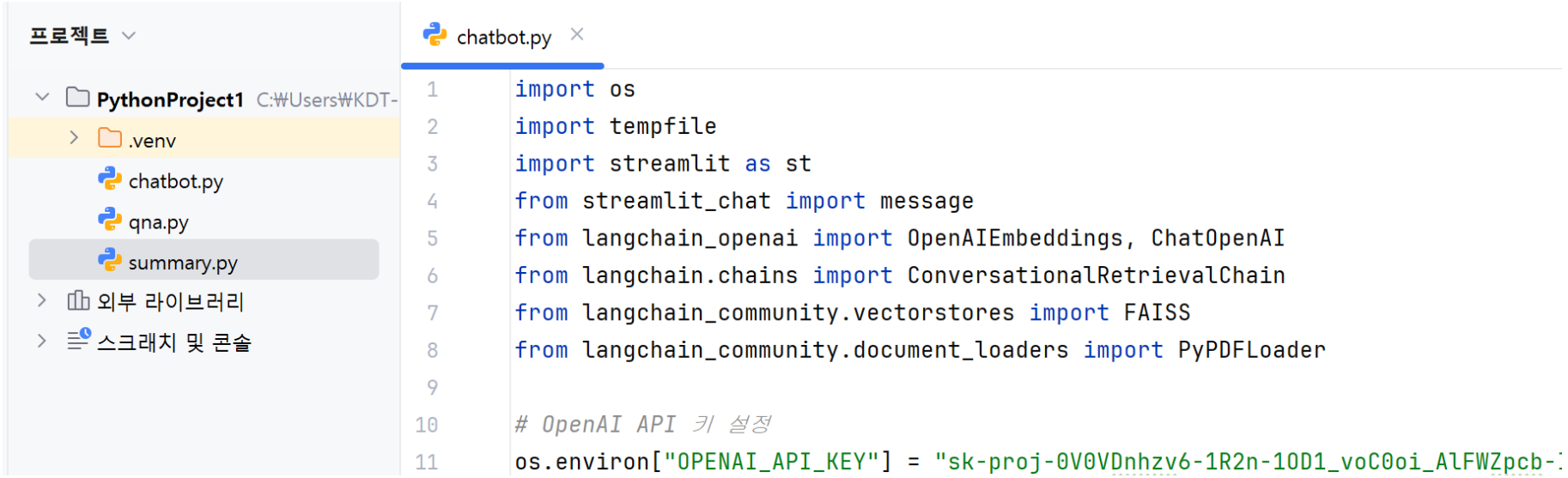


  - 설치: pip install streamlit streamlit-chat langchain langchain-openai langchain-community openai PyPDF2 faiss-cpu

검색 증강 생성 (Retrieval Augmented Generation)

실습 3) PDF 기반 대화형 서비스

- 1. PyCharm 실행 및 chatbot.py 파일 생성



검색 증강 생성 (Retrieval Augmented Generation)

실습 3) PDF 기반 대화형 서비스

- 2. Streamlit 기반 웹 서비스 구현

```python
# 사이드바에서 PDF 파일 업로드
uploaded_file = st.sidebar.file_uploader("PDF 파일을 업로드하세요.", type="pdf")

if uploaded_file:
    # 임시파일로 PDF 저장 후 로딩
    with tempfile.NamedTemporaryFile(delete=False) as tmp_file:
        tmp_file.write(uploaded_file.getvalue())
        tmp_file_path = tmp_file.name

    # PDF에서 문서 로드
    loader = PyPDFLoader(tmp_file_path)
    data = loader.load()

    # 문서 임베딩 및 벡터 DB 생성
    embeddings = OpenAIEmbeddings(model="text-embedding-ada-002")
    vectors = FAISS.from_documents(data, embeddings)

    # GPT-4 기반 대화형 검색 체인 설정
    chain = ConversationalRetrievalChain.from_llm(
        llm=ChatOpenAI(temperature=0.0, model_name='gpt-4'),
        retriever=vectors.as_retriever()
    )
```

```python
# 대화 처리 함수
def conversational_chat(query):  1개의 사용 위치
    result = chain({
        "question": query,
        "chat_history": st.session_state['history']
    })
    st.session_state['history'].append((query, result["answer"]))
    return result["answer"]

# 세션 상태 초기화
if 'history' not in st.session_state:
    st.session_state['history'] = []

if 'generated' not in st.session_state:
    st.session_state['generated'] = [f"{uploaded_file.name}에 관해 질문주세요."]

if 'past' not in st.session_state:
    st.session_state['past'] = ["안녕하세요!"]

# 사용자 입력 컨테이너
response_container = st.container()
container = st.container()
```

검색 증강 생성 (Retrieval Augmented Generation)

실습 3) PDF 기반 대화형 서비스

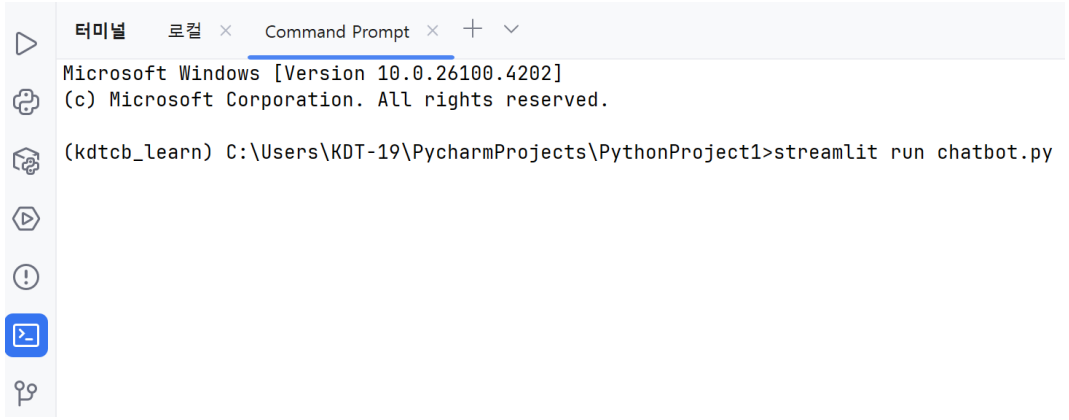
- 2. Streamlit 기반 웹 서비스 구현

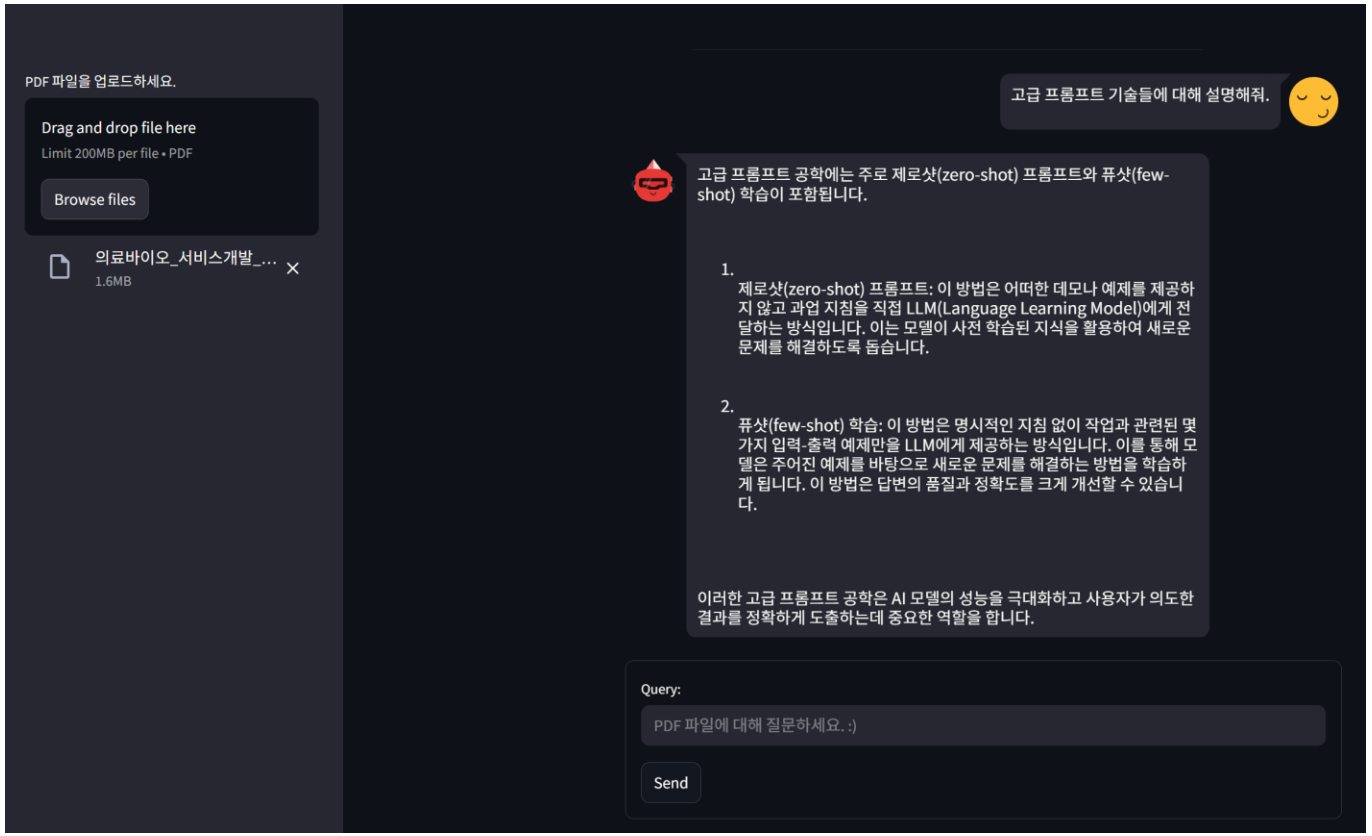
```
59      with container:
60          with st.form(key='Conv_Question', clear_on_submit=True):
61              user_input = st.text_input(
62                  "Query:",
63                  placeholder="PDF 파일에 대해 질문하세요. :)",
64                  key='input'
65              )
66              submit_button = st.form_submit_button(label='Send')
67
68          if submit_button and user_input:
69              output = conversational_chat(user_input)
70              st.session_state['past'].append(user_input)
71              st.session_state['generated'].append(output)
72
73      # 채팅 기록 출력
74      if st.session_state['generated']:
75          with response_container:
76              for i in range(len(st.session_state['generated'])):
77                  message(st.session_state["past"][i], is_user=True, key=f'{i}_user', avatar_style="fun-emoji", seed="Nala")
78                  message(st.session_state["generated"][i], key=str(i), avatar_style="bottts", seed="Fluffy")
79
```

검색 증강 생성 (Retrieval Augmented Generation)

실습 3) PDF 기반 대화형 서비스

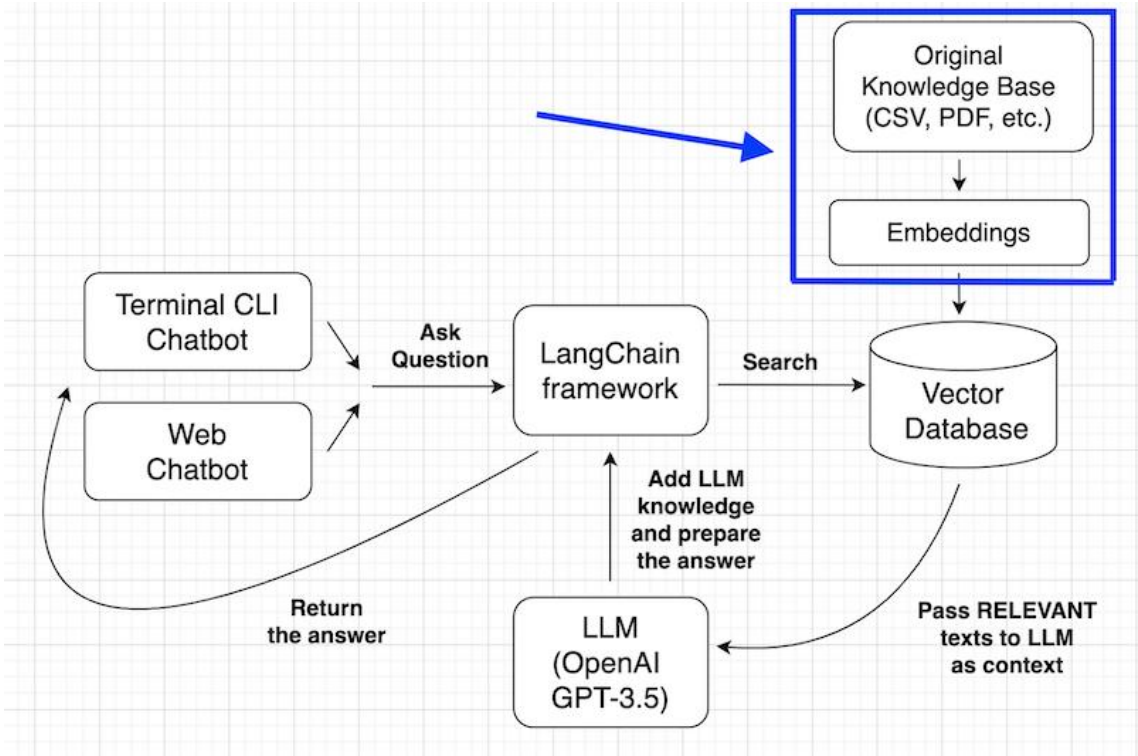
- 3. PyCharm 터미널 내 streamlit run chatbot.py 를 통해 실행





검색 증강 생성 (Retrieval Augmented Generation)

- 실습 4) CSV 파일 분석 서비스
  - 메인 프레임워크



  - 설치: pip install streamlit pandas langchain langchain-openai langchain-experimental openai

검색 증강 생성 (Retrieval Augmented Generation)

실습 4) CSV 파일 분석 서비스

- 1. PyCharm 실행 및 csv_reader.py 파일 생성

검색 증강 생성 (Retrieval Augmented Generation)

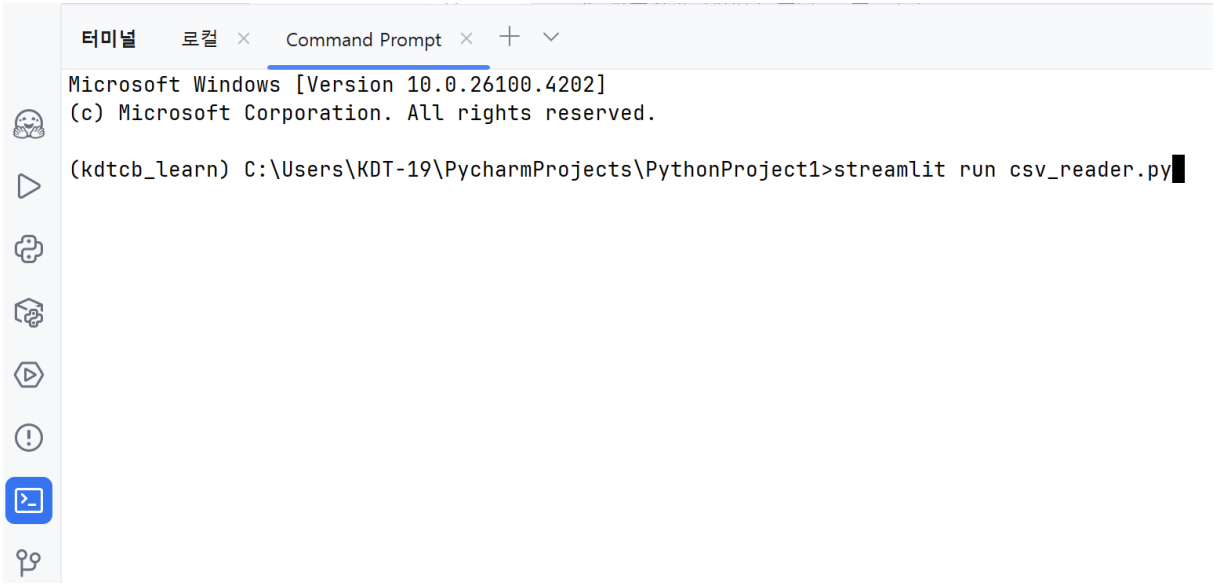
실습 4) CSV 파일 분석 서비스

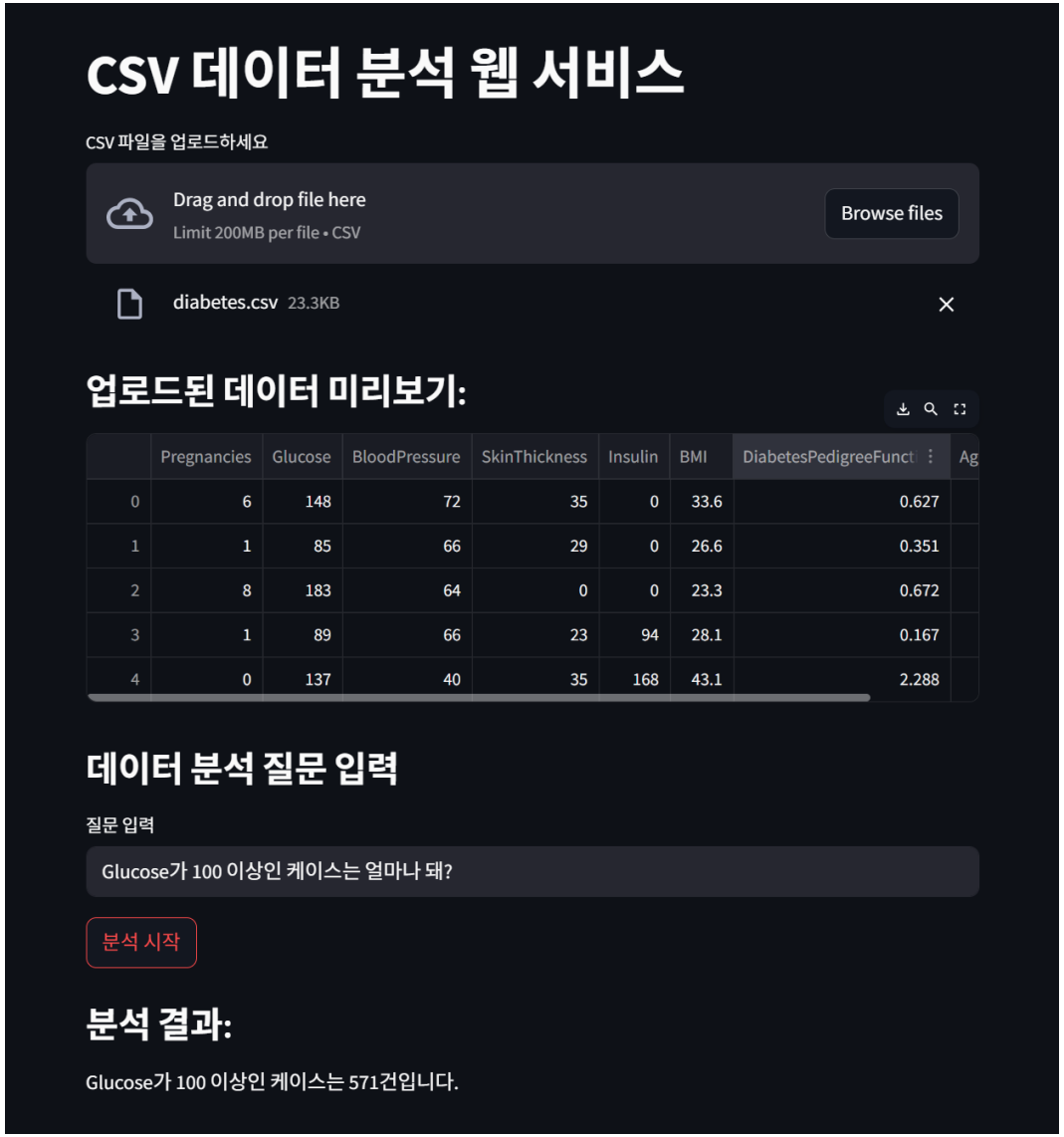
```python
# Streamlit 앱 제목 설정
st.title("CSV 데이터 분석 웹 서비스")

# CSV 파일 업로드
uploaded_file = st.file_uploader("CSV 파일을 업로드하세요", type="csv")

if uploaded_file:
    # CSV 파일을 데이터프레임으로 읽기
    df = pd.read_csv(uploaded_file)
    st.write("### 업로드된 데이터 미리보기:")
    st.dataframe(df.head())

    # LangChain 데이터프레임 분석 에이전트 생성
    agent = create_pandas_dataframe_agent(
        ChatOpenAI(temperature=0, model='gpt-4o'),
        df,
        verbose=False,
        agent_type=AgentType.OPENAI_FUNCTIONS,
        allow_dangerous_code=True
    )
```

```python
    # 사용자가 데이터 분석 질문 입력
    st.write("### 데이터 분석 질문 입력")
    user_query = st.text_input("질문 입력", placeholder="질문을 입력하세요.")

    # 분석 수행 및 결과 출력
    if st.button("분석 시작"):
        if user_query:
            with st.spinner("분석 중..."):
                result = agent.run(user_query)
            st.write("### 분석 결과:")
            st.write(result)
```

검색 증강 생성 (Retrieval Augmented Generation)

실습 4) CSV 파일 분석 서비스





감사합니다

Q&A

충북대학교
CHUNGBUK NATIONAL UNIVERSITY