



학부생 연구실 인턴 활동보고서

인적사항	소속학과	수학과	성명	김연범
	학번	202000826	학년	4학년
전체 활동 기간	2025. 3. 1. ~ 2025. 8. 31 (6개월)			
해당월 활동 기간	2025. 3. 1. ~ 2025. 3. 31 (1개월)			
활동 내용	<ul style="list-style-type: none"> - 랭체인(Langchain)의 기본 개념 및 구조 공부 - OpenAI의 GPT API를 활용한 Closed source LLM 기반 기본 RAG 시스템 구현 및 테스트 - Retriever를 활용해 기본 RAG 시스템에 Memory 기능 추가 및 테스트 - 허깅페이스 플랫폼의 EEVE 모델을 활용한 Open source LLM 기반 Q&A 챗봇 구현 및 테스트 			
활동 성과 및 소감	<p>a) 활동 성과</p> <ul style="list-style-type: none"> - 랭체인(Langchain)의 개념과 구조에 대한 이해 및 활용 능력 향상 - Python의 langchain 라이브러리를 활용한 Closed & Open source LLM 기반 Q&A 챗봇 직접 구현 <p>b) 소감</p> <ul style="list-style-type: none"> - 이미 학습된 임베딩 모델과 LLM을 활용해 Chroma DB를 직접 만들고, 이를 활용해 챗봇을 만드는 과정이 인상 깊고 흥미롭게 다가왔음. 			
평가	성실성	업무능력	활용효과	
	상 . 중 . 하	상 . 중 . 하	상 . 중 . 하	
<p>* 활동사진 첨부</p> <p style="text-align: center;">2025년 3월 19일</p> <p style="text-align: right;">보고자 : 김연범 </p> <p style="text-align: right;">담당교수(지도교수) : 김경식 </p> <p>충남대학교 소프트웨어중심대학사업단장 귀하</p>				

활동사진(2장)

```
from langchain import hub
from langchain.document_loaders import PyPDFLoader
from langchain.text_splitters import RecursiveCharacterTextSplitter
from langchain_openai import OpenAIEmbeddings
from langchain_chroma import Chroma
from langchain_openai import ChatOpenAI
from langchain_core.runnables import RunnablePassthrough
from langchain_core.output_parsers import StrOutputParser
import os

# OpenAI 의 API 키 등록
os.environ["OPENAI_API_KEY"] = "sk-proj-950hCG9R30ttg75609m7AAN_Le2ij20LHmv-Doh2cU49A15zK06xrsaBGA-HtBt2jmlbjxUapbT"

# 원본 PDF 파일 로드
loader = PyPDFLoader("헌법.pdf")
pages = loader.load_and_split()

# PDF 파일을 1000자 청크로 분할
text_splitter = RecursiveCharacterTextSplitter(
    chunk_size=1000, chunk_overlap=100
)
docs = text_splitter.split_documents(pages)

# ChromaDB에 청크들을 벡터 임베딩으로 저장 (OpenAI 임베딩 모델 활용)
vectorstore = Chroma.from_documents(docs, OpenAIEmbeddings(
    model="text-embedding-3-small"
), persist_directory="RAG_EX/Chroma_DB")
retriever = vectorstore.as_retriever()

# GPT-4o-mini 모델 선언
llm = ChatOpenAI(model="gpt-4o-mini")

# Langchain Hub에서 RAG 프롬프트 로딩
prompt = hub.pull('rlm/rag-prompt')

# chain 구축
rag_chain = (
    {"context": retriever, "question": RunnablePassthrough()}
    | prompt
    | llm
    | StrOutputParser()
)

# 테스트
answer = rag_chain.invoke("국회의원의 의무는 뭐야?")
print(answer)

print(rag_chain.get_graph().print_ascii())
```

