

**SK네트웍스 Family AI 과정 15기  
 모델배포 개발된 LLM 연동 웹 애플리케이션**



| **산출물 단계** | 모델배포 |
| --- | --- |
| **평가 산출물** | LLM 연동 웹 애플리케이션 |
| **제출 일자** |  |
| **깃허브 경로** | 깃허브 주소 기재 |
| **작성 팀원** |  |

### 1. 프로젝트 개요

* 프로젝트명:
* 작성일:
* 작성자:
* 버전:
* 승인관리자:

### 2. 시스템 개요

* 목표: 이 애플리케이션은 벡터 데이터베이스와 LLM을 연동하여 효율적인 검색 및 응답을 제공하며, 사용자가 빠르게 정확한 정보를 찾을 수 있도록 합니다.
* 기능:  
  + 벡터 데이터베이스와 LLM 연동
  + 효율적인 프롬프트 최적화
  + 안전한 API 키 관리 및 예외 처리

### 3. 시스템 아키텍처

* 백엔드:  
  + 언어: Python
  + 프레임워크: Flask / Django
  + LLM 연동: OpenAI GPT-4 (또는 다른 LLM)
  + 벡터 데이터베이스: Pinecone / FAISS / Weaviate
  + 데이터베이스: PostgreSQL / MongoDB
  + 기술 스택:  
    - API 관리: Flask RESTful
    - 보안: OAuth 2.0, HTTPS
* 프론트엔드:  
  + 프레임워크: React / Vue.js
  + 상호작용: AJAX 요청을 통한 비동기 통신
  + UI: Material-UI / TailwindCSS

### 4. LLM 연동 및 벡터 데이터베이스 구현

#### 4.1 벡터 데이터베이스와 LLM 연동

* 연동 목적: LLM이 벡터 데이터베이스에 저장된 정보를 바탕으로 적절한 응답을 생성하도록 연동.
* 프롬프트: LLM에 제공되는 프롬프트는 사용자가 요청한 정보에 대한 명확한 답변을 유도하도록 설계되었습니다. 예를 들어, 사용자가 "오늘의 날씨"를 물어볼 때, 적절한 데이터베이스 쿼리를 실행하여 관련 데이터를 LLM에 전달합니다.

# 예시 코드: LLM과 벡터 데이터베이스 연동

import openai

from vector\_database import get\_similar\_documents # 벡터 DB에서 유사한 문서 가져오기

openai.api\_key = "YOUR\_API\_KEY"

def generate\_response(query):

# 벡터 DB에서 유사한 문서 검색

similar\_docs = get\_similar\_documents(query)

# LLM에 전달할 프롬프트 구성

prompt = f"다음 문서들에서 '{query}'와 관련된 정보를 찾으세요: {similar\_docs}"

# LLM 응답 생성

response = openai.Completion.create(

model="gpt-4",

prompt=prompt,

max\_tokens=150

)

return response['choices'][0]['text']

* 프롬프트 최적화:  
  + 프롬프트의 길이를 최소화하고, 데이터베이스에서 가져온 정보를 간결하게 요약하여 LLM에 전달.
  + LLM이 요구 사항에 맞는 최적의 결과를 도출할 수 있도록 주기적으로 테스트하고 최적화.

### 5. 예외 처리 및 보안

#### 5.1 예외 처리

* 예상치 못한 상황에 대한 예외 처리:  
  + API 요청이 실패하거나, 벡터 데이터베이스가 응답하지 않을 때 적절한 오류 메시지를 사용자에게 표시.
  + 예외 처리 코드 예시:

try:

response = generate\_response(user\_query)

except openai.error.OpenAIError as e:

print(f"OpenAI API 오류: {e}")

return "서비스 이용에 문제가 발생했습니다. 잠시 후 다시 시도해주세요."

except Exception as e:

print(f"알 수 없는 오류: {e}")

return "예상치 못한 오류가 발생했습니다. 고객센터에 문의해주세요."

* 보안:  
  + LLM API 키는 환경 변수를 사용하여 코드에서 직접 노출되지 않도록 합니다.
  + 환경 변수 예시:

export OPENAI\_API\_KEY="your-openai-api-key"

import os

openai.api\_key = os.getenv("OPENAI\_API\_KEY")

#### 5.2 보안 관리

* 보안에 민감한 정보 (예: API 키, 데이터베이스 연결 정보 등)는 모두 환경 변수나 비밀 관리 시스템을 사용하여 보호됩니다.
* 환경 변수를 사용하여 코드에 민감한 정보를 노출시키지 않도록 관리합니다.

### 6. 코드 모듈화 및 주석

#### 6.1 모듈화

* 코드는 각 기능을 별도의 모듈로 분리하여 유지보수를 용이하게 합니다.  
  + 예시: llm\_integration.py, vector\_database.py, error\_handling.py 등으로 모듈화하여 필요한 부분만 호출.

#### 6.2 주석

* 주석 예시:

python

복사편집

# LLM 응답을 생성하는 함수

def generate\_response(query):

"""

주어진 쿼리에 대해 LLM을 사용하여 응답을 생성한다.

:param query: 사용자 입력 쿼리

:return: LLM의 응답

"""

try:

# 벡터 DB에서 유사한 문서 검색

similar\_docs = get\_similar\_documents(query)

# 프롬프트 구성

prompt = f"다음 문서들에서 '{query}'와 관련된 정보를 찾으세요: {similar\_docs}"

# LLM에 응답 요청

response = openai.Completion.create(

model="gpt-4", # 사용되는 LLM 모델

prompt=prompt,

max\_tokens=150

)

return response['choices'][0]['text']

except Exception as e:

# 예외 처리

handle\_error(e)