SK네트웍스 **Family AI**과정 **6**기

모델링 및 평가 시스템 아키텍처

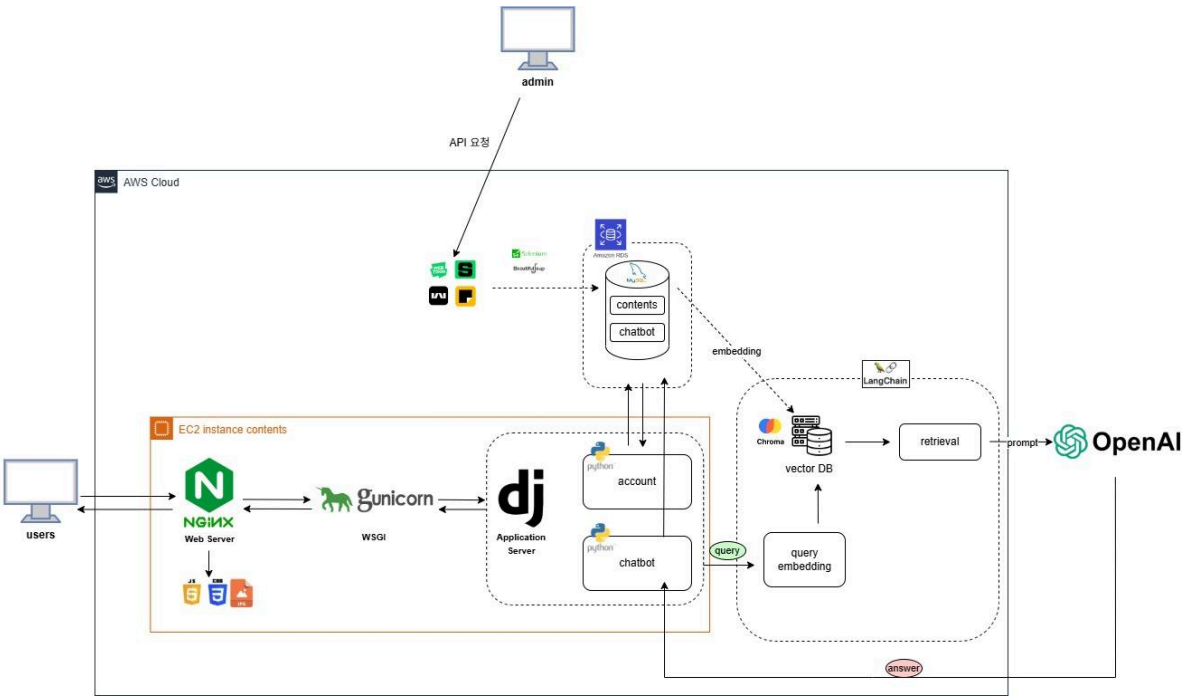
☐ 개요

- 산출물 단계 : 모델링 및 평가
- 평가 산출물 : 시스템 아키텍처
- 제출 일자 : 2025.02.28
- 깃허브 경로 : https://github.com/SKNETWORKS-FAMILY-AICAMP/SKN06-FINAL-2Team
- 작성 팀원 : 이세화, 안형진

개요	본 시스템 아키텍처는 Django 기반 애플리케이션과 AI 챗봇을 통합하여 사용자에게 적절한 웹툰 추천 목록을 제공하는 데 초점을 맞추고 있습니다. 벡터 DB를 활용한 효율적인 검색과 LLM을 통한 자연어 응답 생성을 통해, 사용자 질문에 대한 최적의 답변을 제공하는 구조로 설계되었습니다. **서비스 개요** ● 이 시스템은 사용자에게 LLM을 이용한 웹툰 또는 웹소설을 추천하는 서비스를 위해 구성되었습니다. ● 사용자가 질문을 하면 DB에 저장되어 있는 카카오, 네이버 웹툰/웹소설의 정보와 대조하여 가장 알맞는 작품을 추천합니다. ● 작품마다 별점, 조회수 등 여러 요인으로 작품의 score를 책정하고, 이를 추천에 반영하여 LLM에 올바른 추천을 할 수 있도록 하였습니다. **데이터베이스 및 기술 스택** ● 추천 시스템은 AWS EC2에서 Django 기반 웹 서버를 통해 운영됩니다. ● 데이터는 크롤링 또는 API를 활용하여 주기적으로 최신 상태를 유지합니다. ● 대용량 데이터 처리를 위해 벡터 데이터베이스를 활용하여 LLM이 한번에 처리 할 수 있게 합니다.

구성 요소	### 사용자 인터페이스 (Frontend)
	- **기능**: 웹 기반 UI를 제공하여 사용자와 챗봇이 상호 작용할 수 있도록 함
	- **사용 기술**: HTML, CSS, JavaScript
	- **역할**: 사용자의 입력을 웹 서버로 전송하고, 서버에서 반환된 응답을 표시함
	
	### Web 서버
	- **기능**: 사용자의 요청을 처리하고 wsgi 서버로 전달
	- **사용 기술**: Nginx
	- **역할**: 정적 파일(HTML, CSS, JS 등)을 제공하고, 애플리케이션 서버와 연동하여 사용자 요청을 처리함
	
	### WSGI 서버
	- **기능**: 웹 서버로부터의 요청을 해석해 애플리케이션(django)으로 전달
	- **사용 기술**: Gunicorn
	- **역할**: Django 애플리케이션을 실행
	
	### Application 서버
	- **기능**: 업무 로직을 처리하고 데이터베이스와 상호작용
	- **사용 기술**: Django (Python 기반)
	- **구성 요소**:
	  - **Account 서비스**: 회원 계정 관리(회원가입,로그인/로그아웃,회원정보 수정, 회원 탈퇴)
	  - **Chatbot 서비스**: 웹툰/웹소설 추천 챗봇으로 사용자에게 적절한 작품을 추천하고 저장
	
	### 데이터베이스
	- **기능**: 챗봇의 데이터(콘텐츠 및 대화 기록) 저장 및 관리
	- **사용 기술**: Amazon RDS (MySQL)
	- **저장 데이터**:
	  - contents schema: 웹툰/웹소설 정보 데이터(제목, 별점 등),
	  - chatbot schema: 사용자 대화 기록, 사용자 정보,
	
	### 벡터 데이터베이스
	- **기능**: 문서의 임베딩 벡터를 저장하고 검색하여 관련 문서를 빠르게 찾음
	- **사용 기술**: ChromaDB
	- **구성 요소**:
	  - **Query Embedding**: 사용자 입력을 벡터로 변환
	  - **Vector DB**: 문서 임베딩을 저장하고 검색

	○ **Retrieval**: LangChain을 사용하여 관련 문서 검색 ## LLM 모델 ● **기능**: 검색된 작품을 기반으로 LLM을 활용하여 응답을 생성 ● **사용 기술**: 1. OpenAI GPT 모델("gpt-4o") 2. LangChain (LLM 프레임워크) ● **흐름**: 1. 사용자의 입력을 벡터로 변환 2. 벡터 DB에서 관련 문서를 검색 3. 검색된 문서를 LangChain이 LLM으로 전달하여 적절한 응답을 생성 4. 생성된 응답을 사용자에게 반환 ## AWS 클라우드 ● **EC2 인스턴스**: Nginx, gunicorn, Django 호스팅 ● **Amazon RDS**: MySQL 데이터베이스 관리
데이터 흐름	## 전체 시스템 데이터 흐름 1. 사용자가 웹 브라우저를 통해 질문을 입력 2. 동적 애플리케이션 서버(Django)로 전달 3. Django의 Chatbot 서비스가 사용자의 질문을 받아 Query -> Embedding 벡터 DB(ChromaDB)에서 관련 문서 검색 4. 검색된 문서를 기반으로 LangChain이 OpenAI LLM에 프롬프트를 전달하여 응답 생성 5. 응답이 Django 애플리케이션 서버를 통해 사용자에게 반환됨 6. 사용자와의 대화 내역이 Amazon RDS 'chatbot' 데이터베이스 아래에 저장됨


admin
API 요청
AWS Cloud
contents
chatbot
embedding
LangChain
EC2 instance contents
Chroma
vector DB
retrieval
prompt
OpenAI
users
NGINX
Web Server
gunicorn
WSGI
Application
Server
account
chatbot
query
query
embedding
answer