**SK**네트웍스 **Family AI** 과정 **14**기

모델배포 개발된 LLM 연동 웹 애플리케이션 

| 산출물 단계 | 모델배포 |
| --- | --- |
| 평가 산출물 | LLM 연동 웹 애플리케이션 |
| 제출 일자 | 2025.09.26 |
| 깃허브 경로 | https://github.com/skn-ai14-250409/SKN14-Final-2Team |
| 작성 팀원 | 강윤구,박빛나 |

**1.** 프로젝트 개요

● 프로젝트명: ScentPick - 향수 추천 LLM 챗봇

● 작성일: 2025-09-26

● 작성자: 강윤구,박빛나

**2.** 시스템 개요

● 목표: 향수 데이터셋 + LLM을 연동하여 사용자 취향·브랜드·가격 기반 향수 추천

● 개요: ScentPick은 AI를 활용하여 사용자에게 맞춤형 향수를 추천해주는 웹 서비스입니다. 이 프로젝트는 두 개의 주요 파트로 구성된 현대적인 웹 아키텍처를 따릅니다.

1. Django 웹 애플리케이션 (사용자 인터페이스 및 데이터 관리): 사용자가 직접 상호작용하는 모든 웹 페이지, 회원 관리, 향수 정보 조회, 즐겨찾기 등의 기능을 담당
2. FastAPI AI 서버 (AI 챗봇 및 추천 로직): 복잡한 AI 추론 및 추천 로직을 처리하는 백엔드 서버. Django는 사용자의 채팅 메시지를 이 서버로 보내 AI의 답변을 받아옴

● 페이지별 기능:

○ 메인 페이지: 프로젝트의 대문 역할을 하는 간단한 소개 페이지

○ 향수 목록: 모든 향수 데이터를 보여주는 카탈로그 페이지

* 브랜드, 성별, 농도, 메인 어코드 등 다양한 조건으로 필터링이 가능
* 검색 기능과 페이지네이션이 구현되어 있으며, 필터링 시 AJAX를 통해 페이지 새로고침 없이 상품 목록만 동적으로 업데이트 됨

○ 향수 상세: 개별 향수의 상세 정보(설명, 노트 구성, 용량 등)를 보여줌

* 사용자는 이 페이지에서 즐겨찾기 추가/제거, 좋아요/싫어요 피드백을 남길 수 있음 (AJAX로 처리)
* 노트별 이미지를 note\_translations.py 유틸을 통해 매칭하여 시각적으로 보여줌
* 성별/계절/시간대/노트 특성에 대한 분석 대시보드를 시각화하여 제공

○ AI 맞춤 추천: 날씨 기반 추천: 사용자의 현재 위치(위도/경도)를 받아와 날씨 API를 호출하고, 날씨에 맞는 어코드의 향수를 추천

* 계절 기반 추천: 현재 월(Month)과 로그인한 사용자의 성별을 기준으로 봄/여름/가을/겨울에 맞는 향수를 추천
* 향수 월드컵: 사용자가 '성별/어코드/시간대'를 선택하면 8개의 후보를 제시하고, 토너먼트 방식으로 이상형 월드컵을 진행하여 최종 취향의 향수를 발견하게 도움

○ AI 챗봇: 사용자가 자연어 질문으로 향수를 추천받는 핵심 페이지

* 대화 목록: 좌측 사이드바에서 이전 대화 목록을 불러올 수 있음
* 실시간 스트리밍: 사용자가 메시지를 보내면 chat\_stream\_api 뷰가 FastAPI의 스트리밍 엔드포인트를 호출. FastAPI는 AI 답변을 조각(chunk)으로 보내주고, Django는 이를 StreamingHttpResponse (SSE)를 통해 받아 프론트엔드에 실시간으로 타이핑되는 것처럼 보여줌
* 멀티모달: 이미지 업로드를 지원하여, 사용자가 올린 이미지를 기반으로 향수를 추천받을 수 있음

○ 마이페이지: 추천받은 향수 내역: AI 챗봇에게 추천받았던 모든 향수의 목록과 횟수를 집계하여 보여줌. 날짜, 브랜드, 이름, 횟수별로 정렬이 가능하며, 페이지네이션은 AJAX로 처리됨

* 선호/비선호 향수: 사용자가 '좋아요/싫어요'를 누른 향수 목록을 보여주며, 이 곳에서도 상태를 변경하거나 피드백을 삭제할 수 있음
* 즐겨찾기: 사용자가 즐겨찾기한 향수 목록을 보여주고, 해제할 수 있음
* 회원정보 수정/비밀번호 변경: 이메일, 출생연도, 성별 등 개인정보를 수정하고 프로필 사진을 변경할 수 있음음

○ 오프라인 매장: 카카오맵 API를 연동하여 '올리브영', '향수공방', '브랜드 매장' 등 오프라인 매장의 위치를 지도에 표시해줌

* 지도 이동 시 자동으로 주변 매장을 검색하며, 목록에서 특정 매장을 클릭하면 지도가 해당 위치로 이동

**3.** 시스템 아키텍처

● Django 아키텍처

○ 패턴: 표준 Django MVT(Model-View-Template) 구조

○ Models: Django ORM(Object Relational Mapping)을 통해 DB와 상호작용하는 계층

○ Views : 각 URL 요청에 대한 비즈니스 로직을 처리. 데이터베이스 조회, 필터링, 페이지네이션, FastAPI 서버와의 통신 등을 수행

○ Templates: 사용자에게 보여질 HTML 페이지를 렌더링

○ 인증 (uauth 앱, allauth 라이브러리): Django의 기본 인증 시스템과 django-allauth를 결합하여 로컬 회원가입/로그인뿐만 아니라 네이버, 카카오, 구글 소셜 로그인을 지원

○ 클라우드 서비스

* AWS S3: 사용자가 업로드하는 프로필 이미지와 같은 정적 파일을 저장하고 서빙
* AWS RDS: 운영 환경의 데이터베이스로 MySQL 호환 RDS를 사용

○ 보안: OAuth 소셜 로그인(Google/Kakao/Naver), HTTPS, 환경변수 기반 API 키 관리

● FastAPI 아키텍쳐

○ 프레임워크: 비동기 처리에 강한 FastAPI를 사용하여 AI 모델의 응답을 빠르게 처리

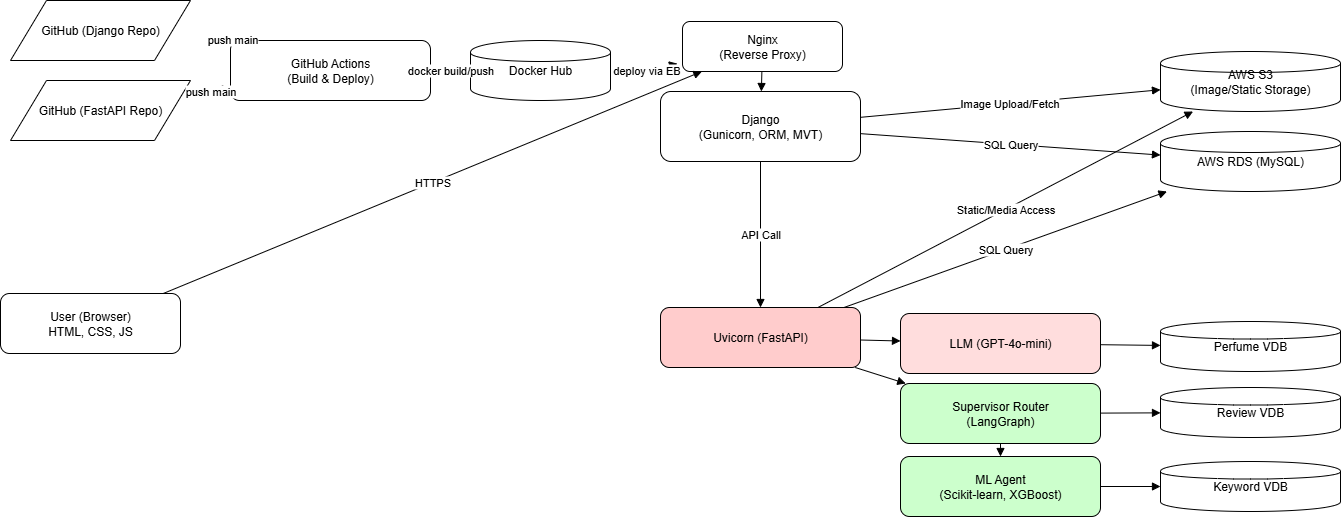
○ 핵심 AI 로직: LangGraph를 사용하여 상태(State)를 가진 대화형 에이전트 시스템을 구축. 이는 단순한 LLM 호출이 아니라, 사용자의 의도에 따라 각기 다른 전문 에이전트(노드)가 작업을 처리하는 구조

* supervisor\_node (라우터/감독관 에이전트) : 모든 사용자 요청을 가장 먼저 받는 관문
* 전문화된 에이전트
  + LLM\_parser\_node: 여러 조건이 포함된 복잡한 쿼리를 처리
  + ML\_agent\_node: 단순한 취향/분위기 기반 쿼리를 처리
  + review\_agent\_node: 향 취향과 가격 조건이 함께 들어온 쿼리를 전문적으로 처리. 리뷰 벡터 DB를 검색하여 사용자의 미묘한 취향을 파악하고, 가격 필터링을 거쳐 최종 추천을 생성
  + price\_agent\_node: 가격에만 초점을 맞춘 쿼리를 처리. 네이버 쇼핑 API를 호출하여 실시간 가격 정보를 제공
  + FAQ\_agent\_node: 향수 관련 지식 질문에 답변
  + multimodal\_agent\_node: 사용자가 이미지를 업로드했을 때 활성화. GPT-4o-mini의 이미지 분석 기능을 사용하여 이미지의 분위기, 색감, 스타일을 텍스트로 해석하고, 이를 기반으로 향수를 추천
  + rec\_echo\_node / memory\_echo\_node: 같은 대화목록 안에서 메타 질문을 처리하여 대화의 편의성을 높임
  + human\_fallback\_node: 향수와 관련 없거나 어떤 에이전트도 처리하기 힘든 질문에 대해 안내 메시지를 보내는 최후의 보루

○ 데이터베이스: Django와 동일한 RDS 데이터베이스를 공유. SQLAlchemy를 통해 대화 내용(conversations, messages)과 추천 로그(rec\_runs, rec\_candidates)를 직접 기록

○ 주요 AI 기술 스택

* LLM: OpenAI GPT-4o-mini를 사용하여 사용자 의도 분석(라우팅), 자연어 이해(파싱), 최종 답변 생성 등을 수행
* 임베딩 모델: 텍스트를 벡터로 변환하기 위해 text-embedding-ada-002 (OpenAI), paraphrase-multilingual-MiniLM-L12-v2 (HuggingFace) 등을 사용
* 벡터 데이터베이스: Pinecone을 사용하여 향수 정보, 리뷰 데이터 등을 벡터로 저장하고, 사용자 쿼리와 의미적으로 유사한 정보를 빠르게 찾아내는 RAG(검색 증강 생성)의 핵심 역할
  + perfume-vectordb: 향수 스펙 기반 벡터 DB
  + review-vectordb: 사용자 리뷰 기반 벡터 DB
  + keyword-vectordb: 키워드-어코드 매칭용 벡터 DB



**3-1.** 시스템 아키텍처 - ERD

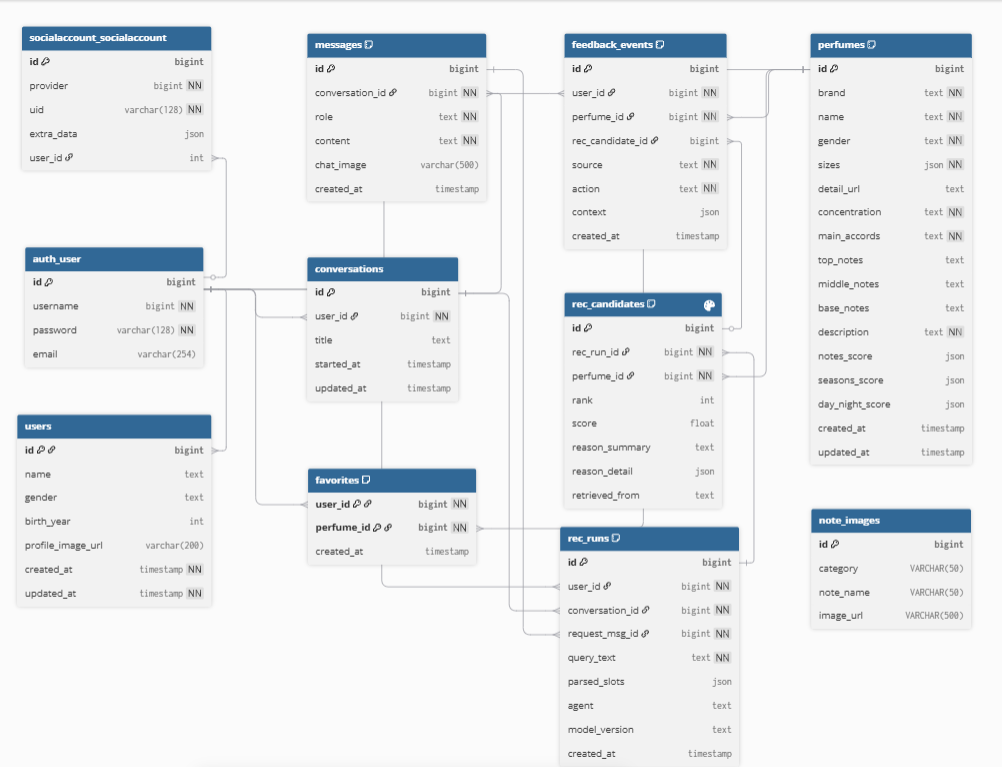
○ Perfume: 향수 기본 정보, 노트, 어코드, 각종 점수(계절, 시간대 등)를 저장

○ User, UserDetail: Django 기본 유저 모델을 확장하여 이름, 성별, 출생연도, 프로필 이미지 URL 등 추가 정보를 관리

○ Conversation, Message: 챗봇 대화 내용을 저장하여 사용자가 이전 대화 내역을 볼 수 있게 함

○ Favorite, FeedbackEvent: 사용자의 '즐겨찾기', '좋아요/싫어요' 같은 상호작용을 기록

○ RecRun, RecCandidate: AI가 추천을 실행할 때마다 어떤 쿼리로 어떤 후보들을 추천했는지 로깅하여 마이페이지의 '추천 내역'에 사용



**4.** 벡터 데이터베이스 연동

● 연동 목적: 향수 메타데이터와 임베딩을 기반으로, LLM이 적합한 향수를 추천하거나 가격·FAQ 응답을 생성

● Pinecone 인덱스 구성

○ 공통 스팩

* + 클라우드: AWS
  + 리전: us-east-1
  + 타입: Dense
  + 용량 모드: Serverless
  + 차원(Dimension): 1536
  + 임베딩 모델: OpenAI text-embedding-3-small
  + 거리(metric): cosine similarity

○ perfume-vectordb

* 향수 스펙 기반 벡터 DB
* 데이터 소스: 바이슈코 향수 사이트 스크래핑 향수 데이터 CSV (brand, name, concentration, gender, sizes 등)
* 저장 방식:
  + page\_content: description (텍스트 설명)
  + metadata: 브랜드, 이름, 농도, 성별, 사이즈, season\_score, day\_night\_score 등
* Record Count : 802

○ review-vectordb

* 사용자 리뷰 기반 벡터 DB
* 데이터 소스: 네이버 카페 ”향수사랑” <https://cafe.naver.com/perfumelove> 리뷰 데이터셋 (url, text)
* 저장 방식:
  + page\_content: 리뷰 본문 (text)
  + metadata: url + text 함께 저장 (검색 후 근거 확인용)
* Record Count : 6,591

○ keyword-vectordb

* 키워드-어코드 매칭용 벡터 DB
* 데이터 소스: 키워드 사전 CSV (note, accord 매핑 등)
* 저장 방식:
  + page\_content: CSV의 key:value 형태 문자열 (예: "note: Rose | accord: Floral")
  + metadata: id와 원본 텍스트
* Record Count : 53

**5**. 예외 처리

5.1 예외 처리

● 예상치 못한 상황에 대한 예외 처리

**Django (웹 서버)**

○ 외부 API 연동 실패 시

* 날씨 API 호출 실패(requests.RequestException) → "데이터 없음" 기본값 반환, 추천 페이지 전체는 정상 렌더링
* FastAPI 챗봇 API 호출 실패
  + requests.HTTPError: FastAPI 응답 오류 시 502 Bad Gateway 반환
  + 기타 Exception: 500 Internal Server Error 반환

○ 데이터베이스 및 조회 오류 처리

* query\_perfumes\_by\_accords: JSONField 조회 실패 시 → TextField 기반 fallback 쿼리 수행
* Conversation.DoesNotExist: 없는 대화 ID 요청 시 빈 대화 상태로 초기화
* User.DoesNotExist: 사용자 조회 실패 시 빈 데이터와 함께 안전한 페이지 렌더링

○ 파일 처리 및 사용자 입력값 오류 처리

* profile\_edit:
  + ValueError: birth\_year 숫자 변환 실패 → 오류 메시지 출력 후 다시 폼 렌더링
  + Exception: 이미지 처리·S3 업로드 오류 → "프로필 이미지 처리 오류" 메시지 노출, 다른 정보는 정상 저장

**FastAPI (AI 서버)**

○ 외부 API 연동 실패 시

* OpenAI API 호출 실패(get\_openai\_embedding) → 예외 로그 기록 후 raise 또는 기본값 반환
* Pinecone 검색 실패 → 오류 로그 기록 후 빈 리스트 반환
* 네이버 쇼핑 API 호출 실패(price\_tool) → {"error": ...} JSON 또는 "❌ 요청 오류" 반환

○ 데이터 파싱 및 형식 변환 오류 처리

* LLM 응답(JSON) 파싱 실패(json.loads) → 원본 사용자 쿼리를 fallback 값으로 사용
* 용량 변환 헬퍼(\_to\_int\_ml) → 변환 실패 시 None 반환

○ AI 에이전트 노드 안정성

* 각 노드(review\_agent\_node) 전체를 try...except로 감싸, 예기치 못한 오류 시 "추천 과정에서 오류가 발생했습니다"라는 일관된 에러 메시지 반환

○ 데이터베이스 무결성 보장

* django\_chat\_endpoint: DB 저장 과정 오류 발생 시 db.rollback() 수행, 데이터 정합성 유지
* get\_db: finally 블록에서 항상 db.close() 실행하여 세션 누수 방지

**6.** 성과 및 향후 발전

6.1 성과

● 단일 RAG 파이프라인에서 확장된 멀티에이전트 챗봇 구현

● 추천, 가격조회, FAQ, 회상 기능 통합

● Django 웹앱과 FastAPI + LangGraph 추천엔진 연동 성공

● 사용자 즐겨찾기·피드백 기록 관리 및 마이페이지 제공

● Fast API의 모듈화 덕분에 새로운 기능(예: '성분 기반 추천 에이전트')을 추가하고 싶을 때, 새로운 노드 파일과 프롬프트를 만들고 Supervisor의 라우팅 규칙만 수정하면 되어 확장성이 매우 뛰어남

6.2 향후 발전

● 하이브리드 검색(BM25 + 벡터 검색) 적용

● 추천 결과에 대해 설명 가능한 AI 기능 강화

● 더 많은 향수 데이터셋과 이미지 확장