서예찬 포트폴리오

1. 소개

안녕하세요, 신입 개발자 서예찬입니다.

저는 팀에서 든든한 존재가 되는 개발자를 꿈꿉니다. "등이 넓은 사람은 믿음직하다"는 말을 떠올리며, 맡은 일을 책임감 있게 완수하고, 동료들이 믿고 의지할 수 있는 개발자로 성장하고 싶습니다.

2. 기술 스택

● LLM Models & AI Frameworks: OpenAI GPT-4o, Hugging Face Transformers, LangChain, LangGraph

● AI & ML Frameworks: scikit-learn, PEFT(LoRA), Prompt Engineering

● Vector DB & Data: Qdrant, FAISS, ChromaDB, pandas, numpy

● Backend & API: Python, Django, REST API, FastAPI (기초), JSON, bash

● Programming Languages: Python, JavaScript, SQL

● Frontend & Web: HTML5, CSS3, JavaScript, Django Template

● DevOps & Cloud: Docker, AWS EC2, RunPod, GitHub Actions

● Tools & Others: VS Code, Jupyter, Google Colab, Postman, Draw.io

3. 프로젝트 경험

[Senpick – 감정 및 상황 기반 대화형 선물 추천 챗봇]

- GitHub : https://github.com/SKNETWORKS-FAMILY-AICAMP/SKN09-FINAL-2Team

작업 기간: 2025.04.23 ~ 2025.06.20

팀 구성: 5명

프로젝트 개요: LLM 기반 대화형 인터페이스를 통해 사용자 감정, 관계, 스타일, 예산에 맞는 선물 추천 시스템 개발

주요 업무 및 역할:

- 상황 정보 추론: LangGraph FSM을 설계하여 감정, 스타일, 예산 추출 및 불충분 조건 시 질문 유도

- 프롬프트 설계: Tool 선택 전략과 응답 구조 통제

- 도구 통합: MySQL(RAG Tool), 네이버 API(Naver Tool) 설계 및 연결

- 프론트엔드: Django + JS 기반 단계별 입력 UI 및 실시간 응답 구현

- 데이터 저장: 추천 이력 관리 및 사용자 입력 DB 구조화

배운 점:

이번 프로젝트를 통해 기술 구현을 넘어서 사용자 감정과 경험을 고려한 시스템 설계가 얼마나 중요한지 깊이 깨달았습니다.

초기에는 감정이나 맥락을 반영하지 않는 기존 추천 시스템의 한계로 인해 사용자 참여도와 만족도가 낮았는데, 이를 해결하기 위해 감정 기반 대화 흐름과 맞춤형 추천 알고리즘을 직접 설계하며 사용자 중심의 서비스를 구현해냈습니다.

특히 LangGraph 기반 FSM으로 대화 흐름을 명확히 제어하면서, 사용자의 감정·스타일·예산이 충분히 수집되지 않으면 도구 호출을 제한하는 구조를 만들 수 있었고, 이는 **UX**와 추천 정확도 모두를 향상시키는 결과로 이어졌습니다.

이 과정을 통해 기술적으로는 FSM 제어 방식과 LLM Tool Agent 구조에 대한 깊은 이해를 쌓을 수 있었고, 동시에 사용자 피드백을 데이터로 수집·반영하는 전체 개선 사이클의 중요성을 실무적으로 체감하는 계기가 되었습니다.

[LLM 기반 대출 상담 챗봇]

- GitHub : https://github.com/SKNETWORKS-FAMILY-AICAMP/SKN09-4th-3Team

작업 기간: 2025.04.17 ~ 2025.04.22

팀 구성: 4명

프로젝트 개요: KoAlpaca 모델을 LoRA 기반으로 파인튜닝하여 금융 분야 대화형 챗봇 모델 구축

주요 업무 및 역할:

- 금융 문서 PDF 수집 및 JSONL Q&A 형태로 정제

- KoGPT2 → KoAlpaca 전환, LoRA 기반 파인튜닝 및 RunPod/Colab 실험

- ChromaDB 기반 검색 시스템 구현 및 챗봇 연동

- AWS 배포, Docker, Gunicorn, nginx 적용

배운 점:

프로젝트 초기에 데이터 구조가 문서화되어 있지 않아 학습 파이프라인이 멈추는 경험을 하면서, 데이터 명세가 사소해 보여도 전체 시스템의 안정성과 직결된다는 점을 깨달았습니다.

또한 CUDA 오류나 OOM 이슈처럼 한 번에 파악하기 어려운 문제를 반복적으로 겪으며, 문제를 예측하고 빠르게 원인을 좁혀가는 디버깅 루틴을 갖추는 것이 개발자로서 얼마나 중요한 역량인지 실감했습니다.

이후에는 작은 설정 하나도 체크리스트화하고, 실험 로그와 오류 대응 과정을 모두 문서화하며 보다 체계적인 개발 습관을 갖게 되었습니다.