



프로젝트 소개 (1)

PSL

PoliSupport Lab



링크: [GitHub Link](#)




















프로젝트 소개

프로젝트를 진행하게 된 배경과 목표, 기획 의도 등의 소개

- 프로젝트 주제 : “보험 약관 문서 기반 챗봇 시스템”
- 프로젝트 목표: LLM 언어 모델을 활용하여 보험 약관 문서 기반으로 자연어 질의응답이 가능한 웹 챗봇 개발

- **보장 범위 안내:** 사용자가 가입한 보험의 보장 내역을 한눈에 확인하고 이해할 수 있도록 분석 및 제공
- **보험 정보 접근성 개선:** 어려운 보험 약관을 쉽게 이해할 수 있도록 정리 및 시각화하여 사용자 경험 향상
- **맞춤형 보험 추천:** 보험에 처음 가입하려는 사용자가 자신의 니즈에 맞는 최적의 보험 상품을 찾을 수 있도록 지원
- 진행 기간 : 2025.03.31 ~ 04.22 (22일)
- 팀원 : 4명
- 주요 업무 : 질의 입력에 대한 답변(모델 결과값) 조절
 - 모델에 따른 답변 비교 및 모델 확정
 - 프롬프트 엔지니어링 및 파인튜닝을 통한 결과값 조절
- 활용 기술

항목	내용
개발 도구	VS Code  Docker  Docker Hub
개발 언어	 Python  HTML5
Vector DB	 FAISS
사용 모델	 GPT-4  LLaMA-3  Kanana-nano  LangChain
서버	AWS EC2  Django  Gunicorn  Nginx
추론 서버	 RunPod  FastAPI
데이터베이스	 MySQL
협업 도구	 GitHub  Notion



프로젝트 핵심 주제를 간단하게 한 문장으로 적어주세요!



프로젝트 진행과정 및 나의 역할

프로젝트 내용 및 진행과정을 간략하면서도 핵심적인 내용 위주로 서술해 주세요.

특히 업무 진행 순서대로 1 → 2 → 3 으로 구성해도 좋습니다.

1. 요구사항 정의 및 시스템 설계

요구사항 ID	요구사항명	요구사항 내용	중요도	난이도	기능 유형
req-001	채팅 시스템	1-1. 사용자는 사이드바에서 알고 싶은 보험사와 보험 종류를 체크 박스 형태로 선택 (중복 선택 가능) 보험사 종류: AXA, KB, Lotte, Hana 보험 종류: 자동차 보험, 일반 보험, 장기 보험 1-2. 선택한 보험사의 사이트의 링크를 사이드바에서 보여준다 2-1. 사용자가 질문하는 내용을 텍스트로 입력창에 작성 내용을 입력받는다. 2-2. 입력된 사용자의 텍스트를 말풍선 안에 넣어 화면에 출력한다. 2-3. 입력 받은 내용(사용자가 선택한 보험 정보, 사용자의 질문)을 llm모델에 연동하여 생성된 응답을 받는다. 2-4. 생성받은 응답 메시지를 말풍선에 넣어 화면에 출력한다. 2-5. 사용자는 '대화종료 및 다운로드'를 클릭하여, 대화를 초기화 하거나 대화내용을 다운로드 가능하다.	상	상	기능
req-002	대화 종료 후 사용자 피드백 받기	1. 사용자가 '대화 종료 및 다운로드'를 선택하면, 팝업 뜨면서 좋아요 & 싫어요 &종료 버튼을 누를 수 있다. 2. 피드백을 받으면 전체 질의 응답 메시지와 피드백 내용을 저장한다.	하	중	기능
req-003	전체 대화 내용을 사용자에게 파일 제공	1. 사용자에게 전체 대화에 대한 기록이 담긴 텍스트 파일을 제공한다.	하	하	기능

1) 챗봇 질의응답 기능

- 사용자는 보험 약관 관련 질문을 자유롭게 입력하고, 보험 약관 문서를 기반으로 답변을 제공.

2) 피드백 제출 기능

- 사용자는 대화를 종료 후에 챗봇 시스템에 대해 (좋아요&보통&싫어요)으로 피드백을 남길 수 있음.
- 이 피드백 데이터를 수집하여 향후 성능향상을 위한 목적으로 모델 학습에 사용될 수 있음.

3) 대화 기록 다운로드 기능

- 사용자는 챗봇 대화 내용을 다운로드 받을 수 있음.
- 이를 통해 상담 이력을 보관하거나 필요한 내용을 다시 확인할 수 있음.

2. 데이터 수집 및 전처리

데이터	약 500개 약관 PDF (약 2.5GB)
보험사별 분류	AXA 손해보험, KB 손해보험, 롯데 손해보험, 하나 손해보험

보험 종류별 분류	자동차보험, 일반보험, 장기(질병, 상해)보험, 저축 및 연금 보험, 기타보험
-----------	---

- 데이터 전처리 내용

- : 페이지 번호 제거
- : 목차 및 주소, QR 코드 등 불필요한 텍스트 제거
- : 다중 공백 및 불필요한 개행 제거
- : 이미지 내 텍스트 추출
- : 보험 종류별로 묶어 임베딩 처리 및 faiss DB 구축

3. 백엔드 환경 구축

```

chatbot_project/
├── manage.py           # Django 프로젝트 실행 관리 파일
├── .env
├──
├── chatbot_project/    # 프로젝트 설정 폴더
│   ├── __init__.py
│   ├── settings.py     # 앱 등록, static 경로, LLM 연동 등 설정
│   ├── urls.py         # 전체 URL 라우팅 설정
│   ├── asgi.py
│   └── wsgi.py
├──
├── chat/               # 챗봇 기능 전담 앱
│   ├── __init__.py
│   ├── apps.py
│   ├── models.py       # DB에 저장할 데이터 구조 정의. 예: 질문, 답변, 피드백
│   ├── urls.py         # /chat, /start, /feedback 등 URL들을 처리
│   ├── forms.py        # Django의 피드백 폼 처리
│   ├──
│   ├── entity/         # 💡 도메인 객체(모델) 정의
│   │   ├── __init__.py
│   │   ├── question.py # 질문 모델
│   │   ├── answer.py   # 답변 모델
│   │   └── feedback.py  # 피드백 모델
│   ├──
│   ├── repository/     # 💡 데이터 접근 및 저장 책임
│   │   └── __init__.py
├──

```

```

| | | — question_repository.py
| | | — answer_repository.py
| | | — feedback_repository.py
|
| | — service/ # 💡 비즈니스 로직 계층
| | | — __init__.py
| | | — chat_service.py
| | | — feedback_service.py
| | — controller # 사용자가 /chat에 들어오면 어떤 HTML을 보여줄지
| | | — views_chat.py
| | | — views_feedback.py
|
| | — migrations/ # DB에 적용할 변경사항을 저장하는 폴더
| | | — __init__.py
|
| | — templates/
| | | — chat/
| | | | — intro.html # 첫 페이지: 챗봇 사용법 설명 화면
| | | | — chat.html # 챗봇 질문/응답 화면 (모달 포함)
|
| — llm/ # LLM 관련 기능 (답변 생성, 벡터 검색 등)
| | — __init__.py
| | — apps.py
| | — urls.py
|
| | — model/ # 💡 LLM 관련 모델 코드
| | | — __init__.py
| | | — generator.py # 텍스트 생성기 (예: GPT, llama 등)
| | | — embedder.py # 임베딩 생성기 (예: HuggingFace)
|
| | — service/
| | | — __init__.py
| | | — llm_service.py # LLM 호출 로직 (프롬프트 구성 등)
| | | — search_service.py # 유사도 검색 및 결과 재구성
|
| | — controller/
| | | — views_llm.py # 답변 생성 로직 (LLM 모델)

```

```

|   |   └─ views_vectordb.py      # 벡터 검색 관련 로직 (faiss 불러오기, 유사도
|   |
|   └─ vector_db/                # 벡터DB 저장 폴더
|       └─ AXA_db
|           └─ axa_faiss_index.bin # FAISS 인덱스
|           └─ axa_docstore.pkl   # 원문 문서 정보 (id → 텍스트 등)
|       └─ KB_db
|           └─ kb_faiss_index.bin  # FAISS 인덱스
|           └─ kb_docstore.pkl     # 원문 문서 정보 (id → 텍스트 등)
|       └─ HANA_db
|           └─ hana_faiss_index.bin # FAISS 인덱스
|           └─ hana_docstore.pkl   # 원문 문서 정보 (id → 텍스트 등)
|       └─ LOTTE_db
|           └─ lotte_faiss_index.bin # FAISS 인덱스
|           └─ lotte_docstore.pkl   # 원문 문서 정보 (id → 텍스트 등)
|
└─ static/                        # 이미지, 아이콘 등 리소스 관리용 폴더
    └─ images/
    └─ icons/

```

4. 사용자 페이지 UI 테스트

[화면정의서.pdf](#)

5. AWS ec2 배포

✓ 아쉬웠던 점과 개선 방향

현 시점에서 가장 아쉬운 점은 **추론 모델의 고도화**입니다.

현재 시스템은 벡터 검색 기반의 RAG 구조에 의존하고 있으며, 문서 내 정보는 잘 추출하나, **사용자 질문의 의도를 다층적으로 해석하거나, 모호한 질문에 유연하게 대응하는 능력은 제한적이었습니다.**

향후에는 피드백 데이터를 기반으로 한 사용자 질의 분류 모델, **프롬프트 체인** 혹은 **Tool-augmented 구조**, 선택적 답변 생성 등 **추론 흐름을 정교화**하는 방향으로 고도화를 진행하

고자 합니다.

✓ 프로젝트 성과 및 배운점

보험 약관이라는 비정형 문서를 기반으로, LLM을 활용한 질의응답 챗봇 시스템을 직접 기획하고 구현하며, 실제 사용자 피드백 및 대화 기록 저장까지 가능한 **엔드투엔드 서비스**를 완성했습니다.

프로토타입 수준을 넘어, AWS EC2를 활용한 **서비스 배포**까지 직접 진행하면서 AI 솔루션의 **기획 → 구현 → 운영** 전 과정을 경험할 수 있었습니다.

특히, 다양한 보험사·보험종류 데이터를 분류하고 embedding 및 벡터DB로 구성한 과정은 향후 실무에서도 반복 가능한 중요한 자산이 되었습니다.

성과 : 복잡한 약관 문서를 대화형 서비스로 바꿔낸 AI 챗봇 구현

- **약관 문서 500개(2.5GB) → 벡터 임베딩 및 검색 가능 구조로 정제**
- **챗봇 피드백 수집 기능 및 대화 다운로드 기능 100% 구현**