

KNOU RAG 챗봇

AI 학사정보 시스템 - 학생 생활을 도와주는 똑똑
한 챗봇



목차

- 1 개발 배경 및 목적
- 2 개발 도구
- 3 주요 기능 소개
- 4 시스템 구조
- 5 내용 및 방법
- 6 실행 예시
- 7 결론

01. 개발 배경 및 목적



문제 상황

공지사항을 놓치는 공지사항



해결 목표

RAG를 활용한 정확한 학사 정보 제공

개발 배경

평소 AI, 특히 RAG 기반 LLM 구축에 관심이 많아 직접 만들어보고 싶었습니다.

그러다 문득, 학기 중이 아닐 때 중요한 방통대 학사 공지를 놓치는 경우가 생각났습니다.

LLM에 RAG를 더하면 쉽게 확인할 수 있겠다는 생각이 들었습니다.

개발 목적

정확한 학사 정보 제공

- 1 RAG (Retrieval-Augmented Generation) 기술을 활용하여 데이터 속에서 필요한 정보를 정확하게 찾아내어 답변함으로써, 정보의 신뢰성을 높입니다.

정보 접근성 및 편의성 향상

- 2 LLM(대규모 언어 모델) 기반의 자연어 처리 능력을 통해 마치 대화하듯이 쉽고 편리하게 학사 정보를 얻을 수 있도록 돕습니다.

학습 경험과 실질적 기여

- 3 인공지능 분야의 최신 기술을 실제 서비스에 적용하는 경험을 쌓는 동시에, 한국방송통신대학교 학생들의 학사 생활에 실질적인 도움을 제공합니다.

02. 개발 도구



개발 언어 및 라이브러리

개발 언어

- Python

웹 프레임워크

- FastAPI
- Uvicorn
- Pydantic

유ти리티

- python-dotenv

코드 관리

- GitHub



AI 모델 및 데이터 처리

대규모 언어 모델(LLM)

- 텍스트 생성 모델
 - Gemini-1.5-Flash
- 임베딩 모델
 - Google-Text-Embedding 004

벡터 데이터베이스

- ChromaDB
- Numpy

데이터 전처리

- Playwright
- Requests
- BeautifulSoup4



시스템 배포 및 운영 인프라

컨테이너 가상화

- Docker
- Supervisor (FastAPI+Cron)

배포 환경

- 미니PC (우분투 리눅스 서버 OS)

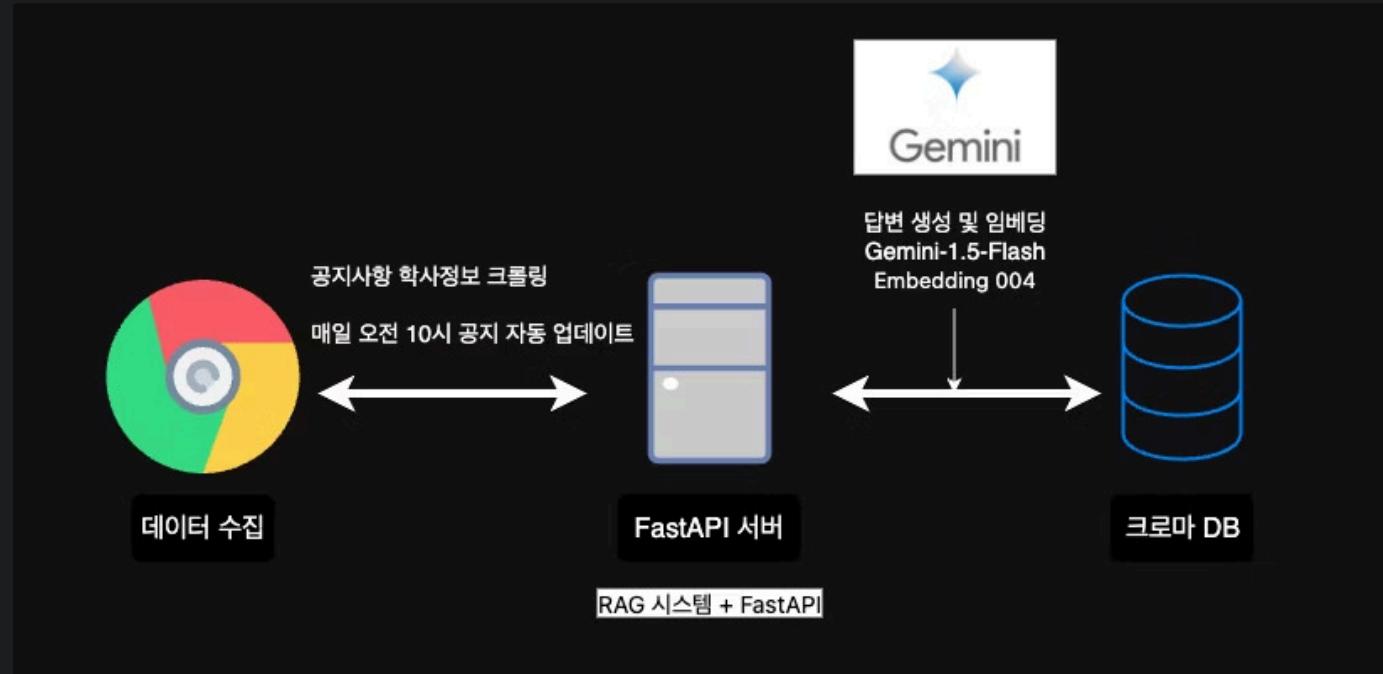
네트워킹

- Cloudflare 터널링
- 도메인 적용 (가비아 구매)

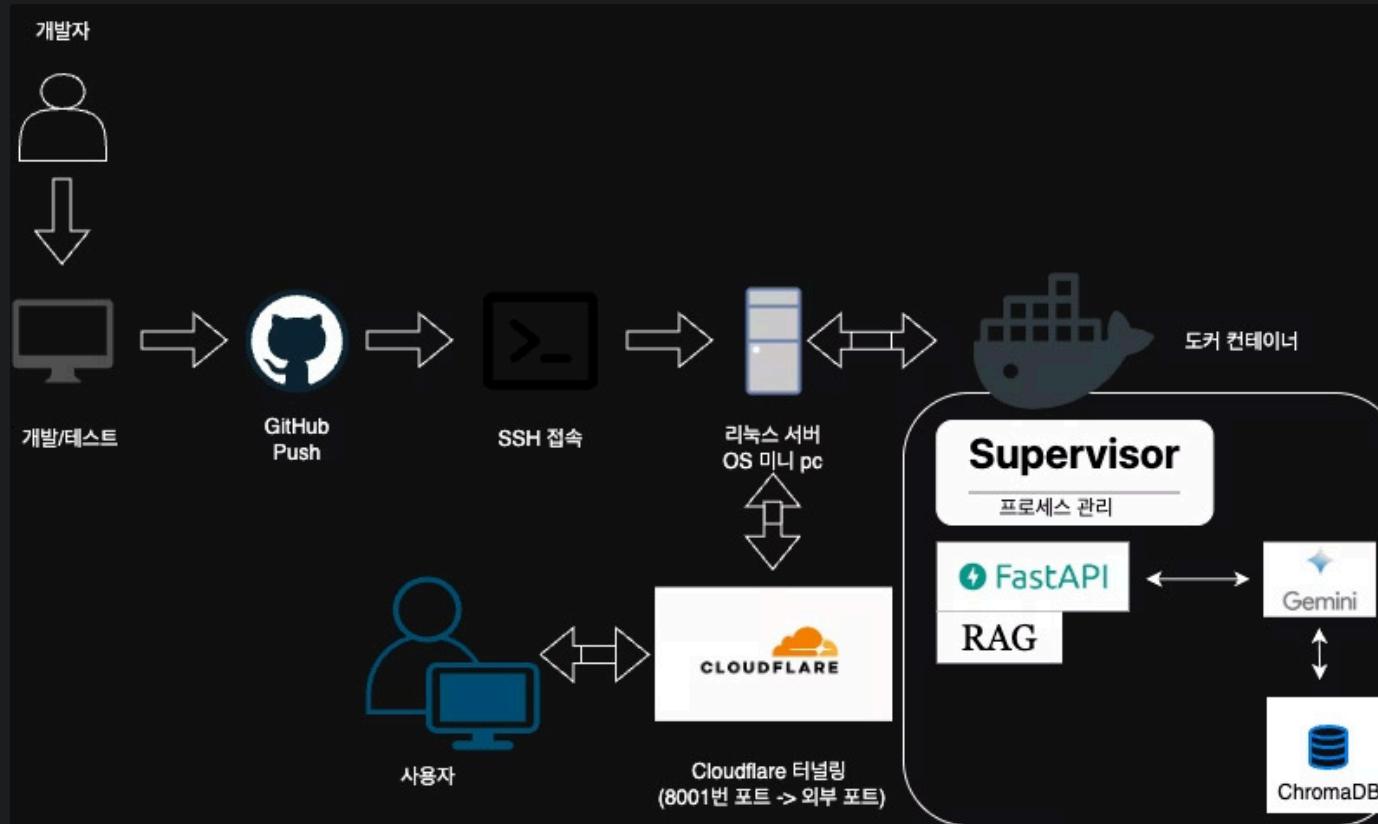
자동화

- 오전 10시 크롤링 스크립트 자동 실행
- 로그 관리

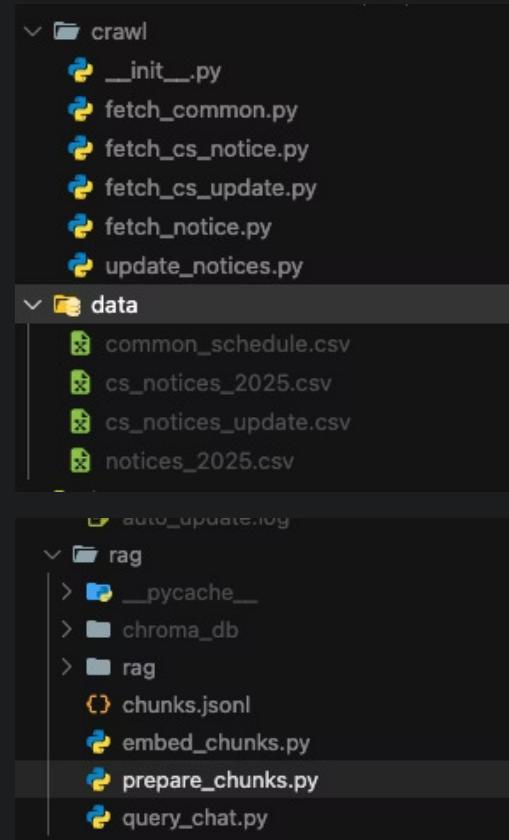
03. 주요 기능 소개



04. 시스템 구조



05. 내용 및 방법 (데이터 생성 및 처리)



- 크롤링 코드를 실행합니다.
- data 디렉토리에 있는 CSV 형식으로 저장된 각각의 데이터를 청크화 시킵니다.

입력 데이터 예시:

[CSV 행]
title: "2025학년도 2학기 등록금 납부 안내"
date: "2025-07-16"
content: "1. 납부기간: 2025.7.25~7.31 ..."

↓

청크화 결과:

```
{  
  "id": "notice_a1b2c3d4e5f6_0",  
  "text": "[2025학년도 2학기 등록금 납부 안내]"}
```

2025-07-16

```
1. 납부기간: 2025.7.25~7.31...,  
  "date": "2025-07-16",  
  "title": "2025학년도 2학기 등록금 납부 안내",  
  "type": "notice",  
  "source": "https://..."}
```

}

- 청크 저장:

- 형식: JSONL (각 줄이 JSON 객체)
- 위치: rag/chunks.jsonl
- ID 체계: {파일종류}_{UUID}_{청크번호}

05. 내용 및 방법 (데이터 임베딩)

```
def main():
    print("Gemini 임베딩 활용 소스입니다...")
    # API 키 설정
    if not GEMINI_API_KEY or GEMINI_API_KEY == "your_api_key_here":
        print("Gemini API KEY를 설정하세요!")
        exit(1)
    print(f"API 키: {GEMINI_API_KEY} 설정되었습니다!")

    # 초기화 및 데이터 설정
    embedding_func = GeminiEmbeddingFunction()
    client = PersistenceClient(path=HOMA_DB)

    try:
        collection = client.get_collection(
            name=COLLECTION_NAME,
            embedding_function=embedding_func
        )
        existing_ids = set(collection.get()["ids"])
        print(f"기존 ID 목록: {list(existing_ids)}가 있습니다")
        except:
            collection = client.create_collection(
                name=COLLECTION_NAME,
                embedding_function=embedding_func
            )
            existing_ids = set()

        print(f"새로운 ID 목록: {existing_ids}")

        # 청크 처리
        new_chunks = []
        for chunk in chunks:
            if chunk["id"] not in existing_ids:
                new_chunks.append(chunk)
        print(f"새로운 청크 수: {len(new_chunks)}개입니다...")

    if not new_chunks:
        print("새로운 청크가 없습니다. 종료합니다.")
        return

    # 배치 단위로 청크 처리
    batch_size = len(new_chunks)
    for i in range(0, len(new_chunks), batch_size):
        batch = new_chunks[i:i+batch_size]

        # 데이터셋 생성
        documents = [chunk["text"] for chunk in batch]
        # 메타데이터 생성
        metadata = [(k, v) for k, v in chunk.items() if k not in ["text", "id"]]
        for chunk in batch:
            ids = [chunk["id"]]

        print(f"배치 크기: ({batch_size}x{batch_size}) 개입니다...")

        # ChromaDB에 추가 (임베딩 수행)
        collection.add_documents(
            documents=documents,
            metadatas=metadata,
            ids=ids
        )

        print("모든 청크가 성공적으로 처리되었습니다!")

    final_count = len(collection.get()["ids"])
    print(f"최종 청크 수: {final_count}개!")

if __name__ == "__main__":
    main()
```

- Gemini 임베딩 설정:
 - 모델: models/text-embedding-004
 - 출력: 768차원 벡터
 - 저장소: ChromaDB (벡터 데이터베이스)
- 임베딩 처리 과정:
 - a. 환경 설정
 - i. Gemini API 키 확인
 - ii. ChromaDB 연결 설정
 - b. 청크 처리
 - i. chunks.jsonl 파일 읽기
 - ii. 기존 임베딩 ID 확인
 - iii. 신규 청크 필터링
 - c. 배치 처리 • 배치 크기: 10개씩 처리 • 각 청크별 처리:
 - 텍스트 → 768차원 벡터 변환
 - 메타데이터 분리
 - ChromaDB에 저장
- 임베딩 저장 구조:
 - 벡터: 텍스트의 의미를 담은 768차원 숫자 배열.
 - 메타데이터: 날짜, 제목, 유형, 출처 정보
 - ID: 청크 고유 식별자

📝 임베딩 처리 예시:

[입력 청크]

```
{
    "id": "notice_a1b2c3d4e5f6_0",
    "text": "[공지사항 제목]\n날짜\n내용...",
    "date": "2025-07-16",
    "type": "notice"
}
```

↓

[임베딩 변환]

- 텍스트 → Gemini 모델 → 768차원 벡터
 - 벡터 예시: [0.123, -0.456, 0.789, ...]
- ↓
- [ChromaDB 저장]
- 벡터: 768차원 임베딩
 - 메타데이터: {"date": "2025-07-16", "type": "notice"}
 - ID: "notice_a1b2c3d4e5f6_0"

05. 내용 및 방법 (검색 및 응답 생성 프로세스)

검색 최적화 기법

- **쿼리 전처리**
 - 일반 용어 → 공식 용어 변환 (예: '학비' → '등록금')
 - 키워드 확장: 유사어 및 관련어 포함
- **하이브리드 검색 (4단계)**
 - 벡터 검색: 의미 기반 유사성 탐색
 - 키워드 검색: 정확한 텍스트 매칭
 - RRF(Reciprocal Rank Fusion)를 통한 결과 통합
 - 날짜 가중치 적용: 최신 정보 우선 배치
- **가중치 시스템**
 - 제목 일치: 5배 가중치 부여
 - 정확한 구문 일치: 10점 보너스 추가
 - 최신 문서 (1년 이내): 최대 1.5배 가중치 적용
 - 오래된 문서 (1년 이상): 0.3배 가중치로 조정

응답 생성 프로세스

- **컨텍스트 구성**
 - 검색된 관련 문서들을 통합하여 컨텍스트 생성
 - 메타데이터 (날짜, 제목 등)를 포함하여 정보 보강
- **프롬프트 구조화**
 - 마크다운 형식을 사용하여 응답 가독성 향상
 - 핵심 요약을 가장 먼저 제공
 - 정보를 섹션별로 명확하게 구분
 - 중요 정보를 강조하여 전달
- **응답 품질 관리**
 - 참고 문서에 기반한 정확한 답변 제공
 - 생성된 답변의 정확성 검증
 - 사용자 친화적인 가독성 최적화

💡 검색 프로세스 예시

[사용자 질문] "2025학년도 2학기 등록금 납부 방법이 어떻게 되나요?"

[검색 프로세스]

1. **쿼리 전처리:** '등록금 납부' → '등록금 납부 안내', 키워드 확장 (등록금, 납부, 수납, 등록비 등)
2. **하이브리드 검색:** 의미 유사성 및 키워드 매칭을 통해 관련 문서 검색, 최신 공지 우선 적용

06. 실행 예시

KNOU RAG CHATBOT
AI 학사 정보 시스템

인녕하세요! 저는 한국방송통신대학교 AI 챗봇입니다.
궁금한 점을 무엇이든 물어보세요.

최신 등록금 관련 공지사항 알리워

핵심 요약 (2025년 7월 16일 공지)
2025학년도 2학기 한국방송통신대학교 등록금 납부 안내가 2025년 7월 16일 공지되었습니다. 등록 기간, 납부 방법(가상계좌, 카드, 은행 방문), 선택적 경비 납부 등의 정보가 포함되어 있습니다.

등록 기간
1/7 등록 기간
신입생, 전입생, 시간제 등록금: 7.28.(월) ~ 7.31.(목)
재학생, 복학생, 재입학생(수강신청 완료자): 정규 등록(7.28.7.31), 최종 등록(8.20~8.22)
재학생, 복학생, 재입학생(수강신청 완료자): 정규 등록(7.28.7.31), 최종 등록(8.20.22)
수강신청 미완료 시 등록 대상자로 생성되지 않음
학비감면 신청자는 추가 또는 최종 등록 기간에 감면된 등록금으로 등록

고지서 조회 및 출력
등록 시작일(각 기간의 첫째 날) 09:30부터 출력 가능 (PC에서만 가능)
홈페이지 로그인 → MyKnou 학사정보 → 등록 → 등록금 조회/납부 → "고지서 인쇄" 버튼 클릭
등록금 납부 의무 이후에는 고지서 출력 불가

등록금 납부 방법

여기에 질문을 입력하세요... ➤

[일반] 2025학년도 2학기 학생회 임원봉사단 운영 계획 알림	학생과	2025.07.17
[일반] 기업체 창설인 허용 안내(2025년 7월 2차)	장학복지팀	2025.07.17
[등록] 2025학년도 2학기 등록금 납부 안내	재무과	2025.07.16
[전공] [25.2학기] 마이크로전공 수강신청 및 이수신청 안내	학사운영과	2025.07.16

1. 등록 기간 (기간의 날짜 불가)				
구분	정규 등록	추가 등록	최종 등록	비고
신입생, 전입생, 시간제 등록금	7.28.(월) ~ 7.31.(목)	8.6.(수) ~ 8.8.(금)	8.20.(수) ~ 8.22.(금)	- 신입생 및 학비감면신청자는 초기 등록기간에 임원봉사단 운영 계획 알림 등록금 납부 안내 장학복지팀 허용 안내 마이크로전공 수강신청 및 이수신청 학사운영과 허용 안내 국가학장금, 북한이탈주민 장학금 등록금으로 등록 (신청한 기초생활수급자, 저소득층, 장애인, 국가학장금, 북한이탈주민)
재학생, 복학생, 재입학생 (수강신청 완료자)	7.28.(월) ~ 7.31.(목)	X	8.20.(수) ~ 8.22.(금)	- 수강신청을 완료하지 않은 등록 대상자로 생성되지 않음 - 재입학생 및 학비감면신청자는 최종 등록 기간에 감면된 등록금으로 등록

2. 고지서 조회 및 출력

- 출력 시간: 등록 시작하는 날 09:30부터 (작업은 70분이면 가능)
- 강도: 홈페이지 로그인 → MyKnou 학사정보 → 등록 → 등록금 조회/납부 → "고지서 인쇄" 버튼 클릭
※ 등록금 납부 작업 후에는 등록금 고지서 출력 절대 불가(하므로 필요 시, 등록금 납부 전 미리 출력/보관)

07. 결론

1. 결론

RAG와 벡터DB 활용

사용할 수 있는 서비스를 만들어서 보람있는 경험이었습니다.

1

개선 방안 도입

날짜 가중치, 용어 정규화, 키워드 확장, 하이브리드 검색 등을 통해 정확도를 높였습니다.

2

데이터 처리의 어려움

단순 임베딩만으로는 한계가 있었습니다. 최신 정보 우선 제공, 일반/공식 용어 매칭 등의 문제가 있었습니다.

3

1 응답 생성 핵심

정보 나열이 아닌 마크다운을 활용한 구조화된 형태로 제공

2 배운 점

AI 기술 실제 적용 과정의 문제 해결 경험, 사용자 경험 개선 기술 습득

3 향후 계획

이를 바탕으로 더 나은 서비스 개발 추진