

AI 챗봇 시스템 중간 발표

2025 메디앙 시스템 학기 계절학기 인턴십

이가은

개요

- 프로젝트 설명
- RAG 설명
- 개발에 사용된 스택
- 주요 고려점
- 개발 진행 이력
- 시연
- 잘 구현된 부분 & 개선이 필요한 부분
- 주요 코드 설명
- 향후 개발 계획
- Q & A

프로젝트 설명

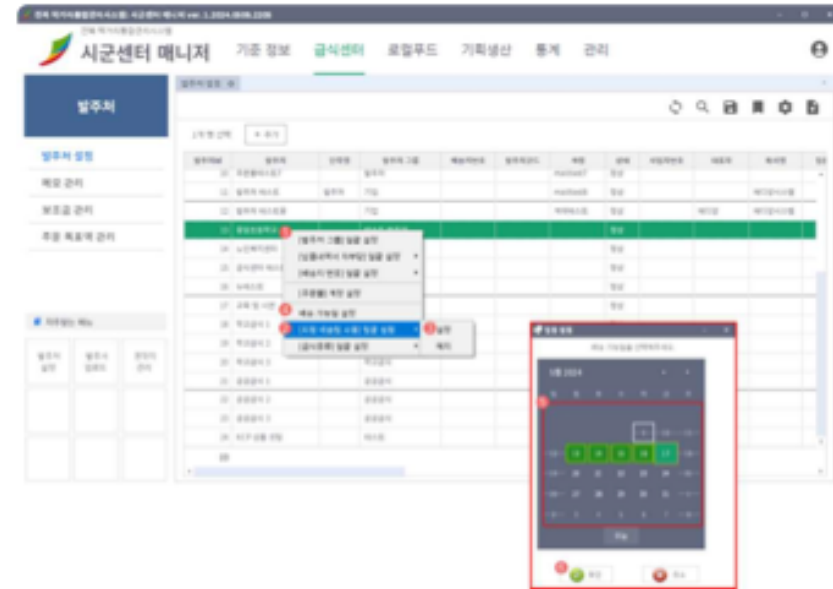
- AI 챗봇 시스템
- 사용자가 질의하면 PDF 형식의 문서 내용을 바탕으로 답변(텍스트 + 이미지)을 제공합니다.

☺ 메뉴얼 챗봇

PDF 메뉴얼에 대한 질문을 입력해 보세요.

🔴 발주처별 지정 배송일은 어떻게 설정하나요?

🛒 발주처 설정 & 단가 등록 메뉴얼에 따르면, 발주처별 지정 배송일 설정을 위해 발주처 설정에서 해당 발주처를 마우스 우클릭하고 [지정 배송일] 일괄 설정을 클릭합니다. 그런 다음 배송 가능일을 선택하고 "확인" 버튼을 눌러 완료합니다.



관련 이미지

PDF 메뉴얼에 대해 궁금한 점을 입력해 주세요...

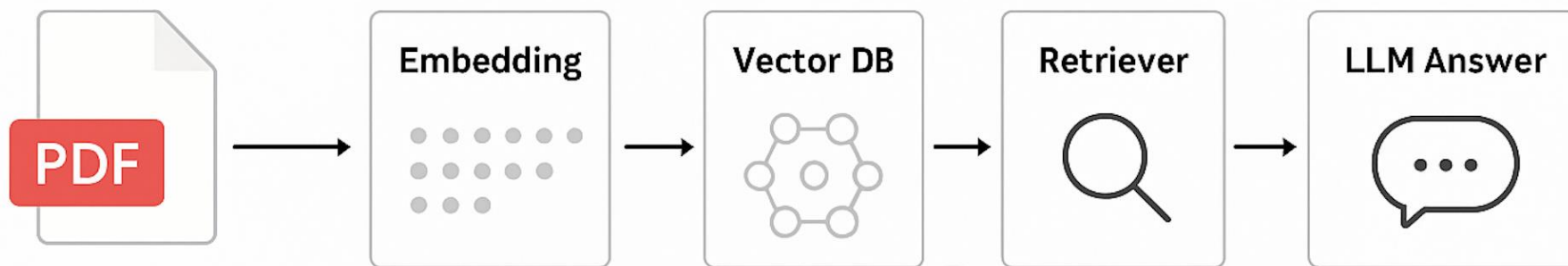
RAG (Retrieval-Augmented Generation)

- RAG는 검색 기반 생성 방식으로, 외부의 지식을 검색하여 답변을 생성하는 방식입니다.

- 1. **Retrieval** : 데이터를 가져오는 것
- 2. **Augmented** : 정보를 아는 것 처럼
- 3. **Generation** : 답변을 생성

- RAG 파이프 라인

문서 load → 분할 split → 임베딩 embedding → Vectort DB에 저장
→ 인덱싱 및 검색 → 유사도 높은 문장 추출 Retrieval → LLM 응답 생성



RAG 파이프라인 (주요 5단계 설명)

1. **데이터 로드** (Load Data) : RAG에 사용할 데이터를 불러옵니다.
2. **텍스트 분할** (Text Split) : 불러오는 데이터를 작은 크기의 단위(chunk)로 분할하는 과정입니다.
3. **인덱싱** (Indexing) : 분할된 텍스트를 검색 가능한 형태로 만드는 단계입니다.
4. **검색** (Retrieval) : 사용자의 질문이나 주어진 문맥에서 가장 관련된 정보를 찾아내는 과정입니다.
5. **생성** (Generation) : 검색된 정보를 바탕으로 사용자의 질문에 답변을 생성하는 최종 단계입니다.

RAG 방식을 사용하는 이유

1. **풍부한 정보 제공** : RAG 모델은 검색을 통해 얻은 외부 데이터를 활용하여 보다 풍부한 정보를 제공합니다.
2. **실시간 정보 반영** : 최신 데이터를 반영하여 변하는 정보에 대응할 수 있습니다.
3. **환각 방지** : 검색을 통해 실제 데이터에 기반한 답변을 생성함으로써, 환각 현상이 발생할 위험이 줄어듭니다.

개발에 사용된 스택

- **Python** : 사용된 언어
- **PyMuPDF**, Tesseract OCR : PDF 로드, 텍스트 및 이미지 추출
- **Langchain** : RAG 체인 구성 (LLM 프레임워크)
- **OpenAI GPT-4o** : 질의 응답 생성
- **Chroma** Vector DB : 임베딩 저장, 검색
- **Streamlit** : UI 구현


주요 고려점


1. **히스토리 기반 Chain** – 대화 맥락 반영으로 자연스러운 응답.
2. **메타데이터 트리 구조화** – 섹션-서브섹션 계층 구조 설계.
3. **정규식 기반 파싱** – 자동화된 문서 분리 및 인덱싱.
4. **이미지 경로 자동화** – 텍스트 + 이미지 동시 출력 구현.


히스토리 기반 Chain

대화의 맥락을 반영하여 자연스러운 응답을 생성한다.


 발주처 등록은 어떻게 하나요?


 발주처 설정 & 단가 등록 메뉴얼에 따르면, 급식센터 - 발주처 - 발주처 설정에서 발주처 등록을 하려면 발주처 설정에서 등록 버튼을 클릭하고, 발주처 명을 입력한 후 발주처 그룹을 선택하고 필요한 기타 정보를 입력한 뒤 저장 버튼을 클릭하면 됩니다.


 기타 정보를 반드시 입력해야 하나요?

 발주처 설정 & 단가 등록 메뉴얼에 따르면, 기타 정보는 반드시 입력하지 않으셔도 됩니다.


 발주처 등록은 어떻게 하나요?

 발주처 설정 & 단가 등록 메뉴얼에 따르면, 발주처 등록은 급식센터 - 발주처 - 발주처 설정에서 '추가' 버튼을 클릭하여 진행합니다. 발주처 명과 그룹을 선택하고, 필요한 경우 기타 다른 정보를 입력한 후 '저장' 버튼을 클릭하여 완료합니다.


 기타 정보를 반드시 입력해야 하나요?

 모르겠습니다.

히스토리 ○

-  : 기존 정보를 바탕으로 대답을 잘 해준다.

히스토리 x

-  : 맥락이 끊긴다

메타데이터 트리 구조화

```
[
  {
    "title": "공공급식 단가 관리",
    "section": "IV",
    "section_title": "발주처 설정 & 단가 등록",
    "subsection": "1",
    "subsection_title": "발주처 그룹 등록",
    "text": "",
    "page_number": 37,
    "image_path": ["images/section4-1_page37_img1.png"]
  },
  {
    "title": "공공급식 단가 관리",
    "section": "IV",
    "section_title": "발주처 설정 & 단가 등록",
    "subsection": "2",
    "subsection_title": "발주처 등록",
    "page_number": 38,
    "image_path": ["images/section4-2_page38_img1.png"]
  },
  {
    "title": "공공급식 단가 관리",
    "section": "IV",
    "section_title": "발주처 설정 & 단가 등록",
    "subsection": "3",
    "subsection_title": "발주처 계정 발급",
    "page_number": 39,
    "image_path": ["images/section4-3_page39_img1.png"]
  }
]
```

- 메타데이터의 트리 구조화
- 섹션-서브섹션 계층 구조 설계.
- 트리 구조 관계를 설정하여 메타데이터를 추출해 벡터에 저장한다.

정규식 기반 파싱

▶ Section IV: 발주처 설정 & 단가 등록

[1] Page: 1

```
section_num : IV
section_title : 발주처 설정 & 단가 등록
subsection_num :
subsection_title :
has_image : False
image_path : None
page_content : ...
```

[2] Page: 2

```
section_num : IV
section_title : 발주처 설정 & 단가 등록
subsection_num : 1
subsection_title : 발주처 그룹 등록
has_image : True
image_path : ./extracted_images/page_2.jpeg
page_content : ...
```

[3] Page: 3

```
section_num : IV
section_title : 발주처 설정 & 단가 등록
subsection_num : 2
subsection_title : 발주처 등록
has_image : True
image_path : ./extracted_images/page_3.jpeg
page_content : ...
```

- 문서 속 구조화된 메타데이터를 자동으로 추출합니다.
- 벡터 데이터베이스에 저장해 인덱싱을 합니다.

```
regex_pattern = r"^M{0,3}(CM|CD|D?C{0,3})(XC|XL|L?X{0,3})(IX|IV|V?I{0,3})$"
```

```
import re
```

```
print(str(bool(re.match(regex_pattern, input()))))
```

↑ 로마 숫자 정규식

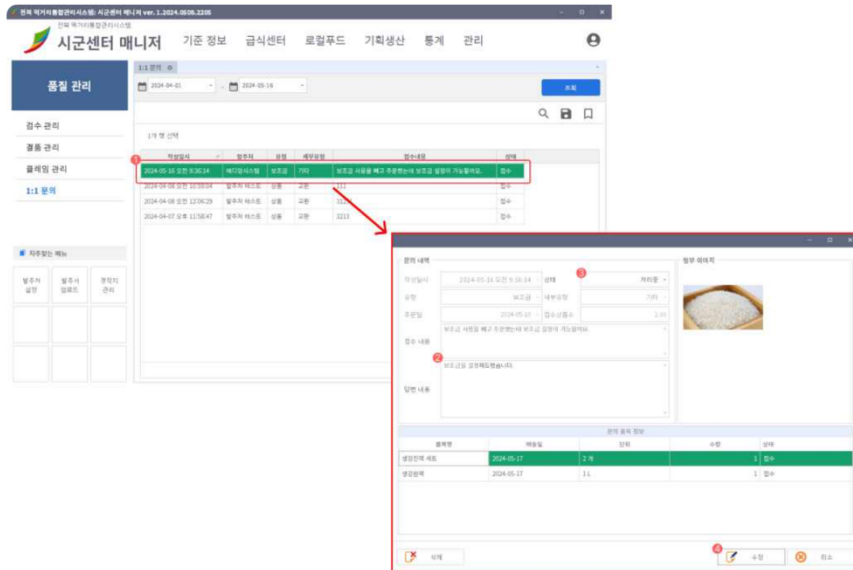
이미지 경로 자동화



공급자 계정 권한 부여는 어떻게 하나요?



메뉴얼에 따르면, 공급자 계정 권한 부여는 급식센터의 설정에서 공급자 계정 관리로 이동하여 진행합니다. 권한을 부여할 공급자를 클릭하고, 권한 부여할 메뉴의 체크박스를 선택한 후, 접근을 허용할 매장을 선택하면 됩니다.



관련 이미지

- 텍스트 + 이미지를 동시에 출력합니다.

개발 진행 이력 : 타임 라인

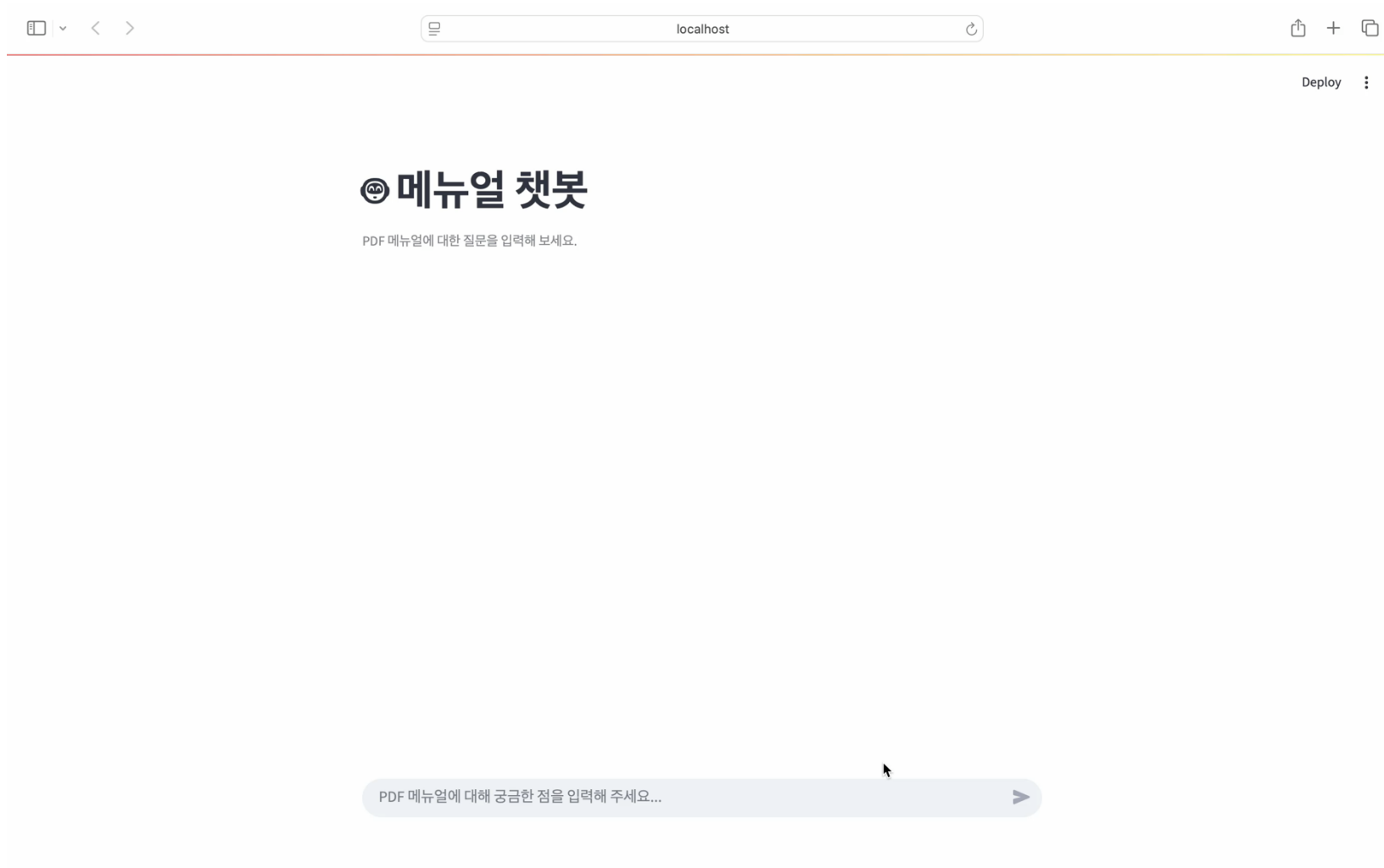
- 1주차: PDF 로더 & 텍스트 분할
- 2주차: 임베딩 생성 + Chroma 벡터 DB 구축 + RAG 체인 구현 & 테스트
- 3주차: 이미지 메타데이터 매칭 + Streamlit UI 완성

자세한 내용 : [노션 : 2025 여름 현장 실습/인턴개발일지](#)

개발 진행 이력 : 문제 - 해결 - 결과

문제 (Problem)	해결 (Solution)	결과 (Result)
이미지 정보가 메타 데이터에 반영되지 않음	Parsing 단계에서 이미지 메타데이터 추가 및 구조화	메타데이터 자동 추출 및 이미지 출력 가능
<code>page_number</code> 누락으로 페이지 인덱싱 불가	PDF Parsing 단계에서 <code>page_number</code> 추가하여 페이지 번호 반환	정확한 페이지 참조 가능
LLM이 이미지 경로를 반환하지 못함	<code>chain.invoke()</code> 에서 context를 직접 관리하여 <code>image_path</code> 반환	텍스트 + 이미지 동시 응답
PDF 양이 많아질수록 검색 정확도 저하	섹션/서브섹션 기반 정규식 파싱으로 문서 계층 구조화	섹션 기반 검색 정확도 향상
섹션/서브섹션 구분 불가	<code>split_by_section_and_subsection</code> 함수 작성 및 인덱싱 개선	트리 구조 기반 효율적 검색 및 확장성 확보

시연, 시연 영상



잘 구현된 부분

1. 텍스트 + 이미지 동시 출력 가능 (Streamlit UI 완성)
2. 섹션.서브섹션 기반 검색 정확도 향상
3. 메타데이터 자동 추출 및 구조화 성공

개선이 필요한 부분

1.로딩 중 이전 이미지가 노출됨

- 새로운 질문 대기 중 기존 이미지가 남아 있는 문제 → streamlit 출력 형식

2.유사도 검색 정확도 저하

- 6.1, 6.2, 6.3과 같은 복잡한 subsection에서 검색 실패. → 코드 수정 중.

3.숫자 기반 subsection 인식 오류

- 예: "수주 단가 일괄 등록 (8.1)"을 물으면 8번 내용을 답함. → 코드 수정 중.

4.다중 PDF 미지원

- 현재 구조는 단일 PDF 전용 → 멀티 PDF 환경 코드 수정 중.

5.여러 이미지가 있을 때 1장만 출력

- 다중 이미지 처리 로직 미비. → 코드 수정 중.

주요 코드 설명

- 프로젝트 파일 구조

파일명	역할 설명
<code>llm.py</code>	LangChain 기반 RAG 처리, 이미지 추출, 검색 등 핵심 기능
<code>chat.py</code>	Streamlit UI, 채팅 인터페이스, 세션 관리
<code>config.py</code>	GPT 응답 유도용 예시 문장 설정
<code>.env</code>	OpenAI API 키

주요 코드 로직 설명 (llm.py)

1. 데이터 로드 (Data Load)

1. PyMuPDFLoader로 PDF 텍스트와 이미지 추출
2. extract_images_from_pdf()로 이미지 경로 메타데이터화

2. 텍스트 분할 (Text Split)

1. split_by_section_and_subsection()로 섹션/서브섹션 단위 문서화
2. page_number, section_num, subsection_num 메타데이터 생성

3. 인덱싱 (Indexing)

1. OpenAIEmbeddings로 벡터화 후 Chroma 벡터 DB에 저장

4. 검색 (Retrieval)

1. db.as_retriever()로 유사 문서 검색
2. 히스토리 기반 create_history_aware_retriever() 사용

5. 생성 (Generation)

1. ChatOpenAI 모델로 답변 생성
2. create_retrieval_chain()으로 검색+생성을 연결

향후 개발 목표 (2주)

1. API화 목표 (프론트엔드)

- FastAPI를 활용해 백엔드 → API 서버화
- Streamlit UI를 유지하면서 외부 호출 가능한 API로 확장

2. 백엔드 문제점 해결

- 메타데이터 구조 개선 (섹션·서브섹션 정교화)
- 다중 PDF 환경 지원 (멀티 문서 검색 및 통합)
- 이미지 처리 개선 (다중 이미지 대응)

3. 배포 및 테스트

- 배포 후 테스트 및 성능 검증

Q & A

감사합니다.