LG 부트캠프 9기

프로젝트 결과보고서

FOBI: File-Oriented Bot Interface

목차 기반 문서 탐색 서비스

데이터 처리 업무 자동화반 7팀

김민 김하늘 천용태 최재원

목차구성

CONTENTS COMPOSITION

1. 주제 및 결과 요약

프로젝트 주제

설계 문서에 대한 이해도가 낮은 사람도 쉽게 문서의 구조를 이해할 수 있도록

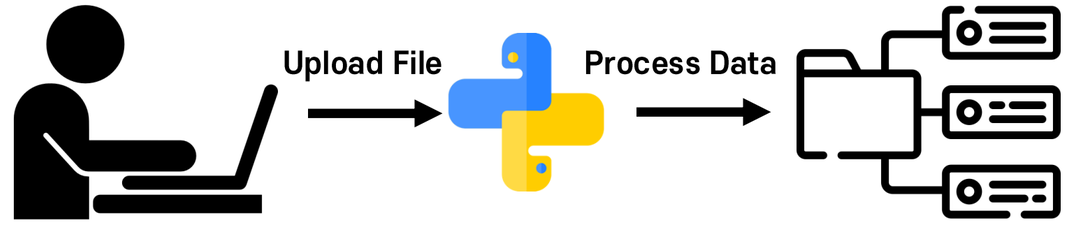
“목차 기반”으로 문서 탐색을 도와주는 서비스

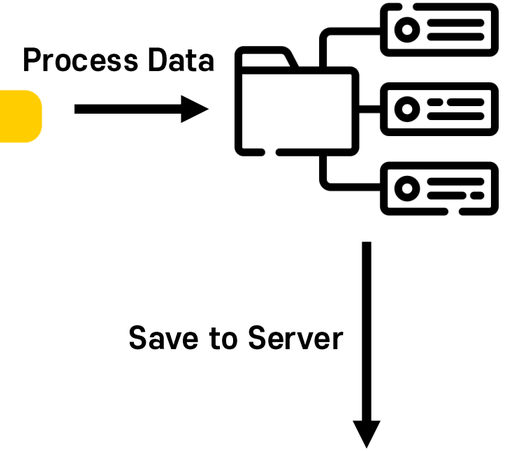
프로젝트 배경

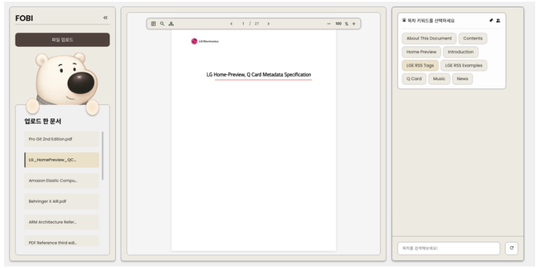
- 방대한 설계 문서를 이해하는데 겪는 어려움 해소
- 문서 관련 담당자와 히스토리를 찾느라 쏟는 시간을 줄이기

1. 주제 및 결과 요약

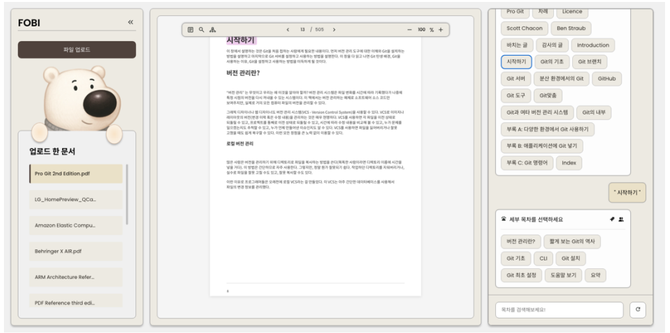
프로젝트 구성



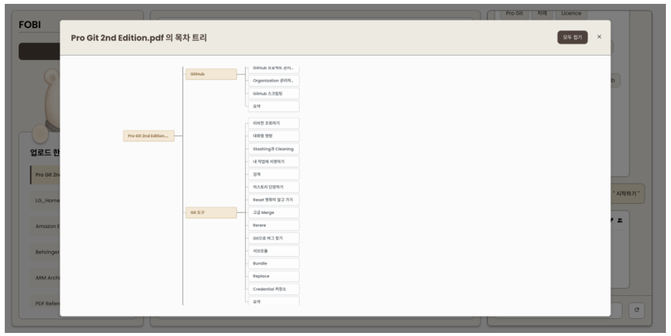




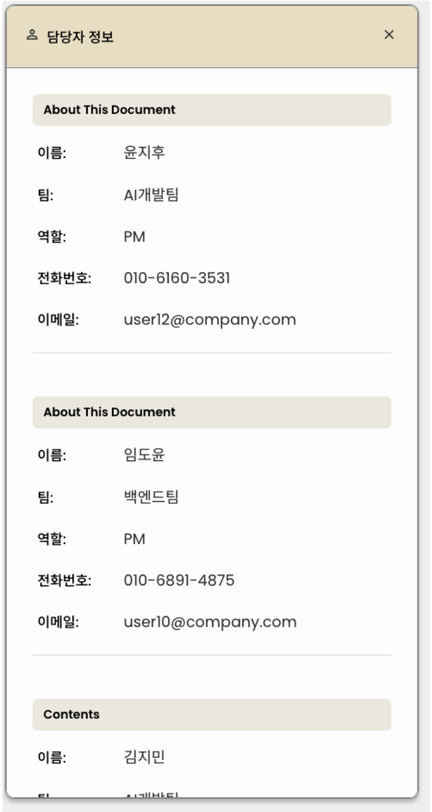
프로젝트 결과



키워드 챗봇 & 문서 뷰어



키워드 시각화 트리



담당자 &이슈 확인

2. 요구사항 분석 및 시스템 명세

요구사항 분석	시스템 명세
1 목차 키워드 추출을 통해 목차 구조 파악 및 관련 정보를 제공해야 함	• 텍스트 추출 후 목차 (TOC, “Contents”) 항목 인식, 키워드 추출 • Rule-based Parsing 알고리즘 적용 및 PyMuPDF 연동 • 관련 담당자, 이슈 정보 Mock 데이터 제공
2 추출된 목차 키워드들을 사용자가 한 눈에 파악할 수 있도록 제공해야 함	• React-d3-tree 라이브러리를 이용해 트리 뷰(Tree View) 형태로 문서 구조를 표현 • 계층적 목차를 챗봇 기반 탐색 인터페이스로 시각화
3 사용자가 키워드를 선택 시, 문서에서 해당 키워드를 탐색하여 UI 요소를 추가해야 함	• React-pdf 라이브러리로 모든 텍스트 레이어 내 키워드 탐색 • 모든 페이지 동시 검색해 각 레이어에서 TreeWalker를 사용해 모든 텍스트 노드를 찾아 검색어와 일치하는지 확인

2. 요구사항 분석 및 시스템 명세

	요구사항 분석	시스템 명세
4	사용자가 키워드 검색 시, 실시간으로 검색 결과 제공 및 UI 반영	서버에서 업로드된 파일의 키워드 추출 결과를 전송하면, 해당 데이터로 클라이언트 측에서 filter 수행
5	업로드한 문서를 사용자가 관리할 수 있어야 함	문서 데이터는 서버의 API를 통해 관리되며, 작업(로드/수정/삭제)마다 적절한 API 엔드포인트를 호출
6	브라우저 안에서 실제 PDF 뷰어처럼 사용하도록, 표준적인 모든 기능을 제공해야 함	PDF.js 커스터마이징 : PDF 뷰어 컴포넌트 구현, 기본 pdf 렌더링, 뷰어 모드 전환, 텍스트 스크롤링/검색 및 하이라이트, 페이지 네이게이션, 확대/축소 기능

3. 개발 목표 및 개발 결과

개발 목표

- Pdf 파일에서 목차 키워드를 추출해 Rule-Based 챗봇 기능 제공
- 데이터 처리 및 키워드 추출 결과 전송을 위한 서버 연동
- 문서 구조 파악을 위한 키워드 트리 제공
- 키워드에 해당하는 사내 담당자 & 관련 이슈 정보 제공

개발 결과

✅ PDF 내 목차 키워드를 추출을 통한 Rule-Based 챗봇 기능 구현 완료

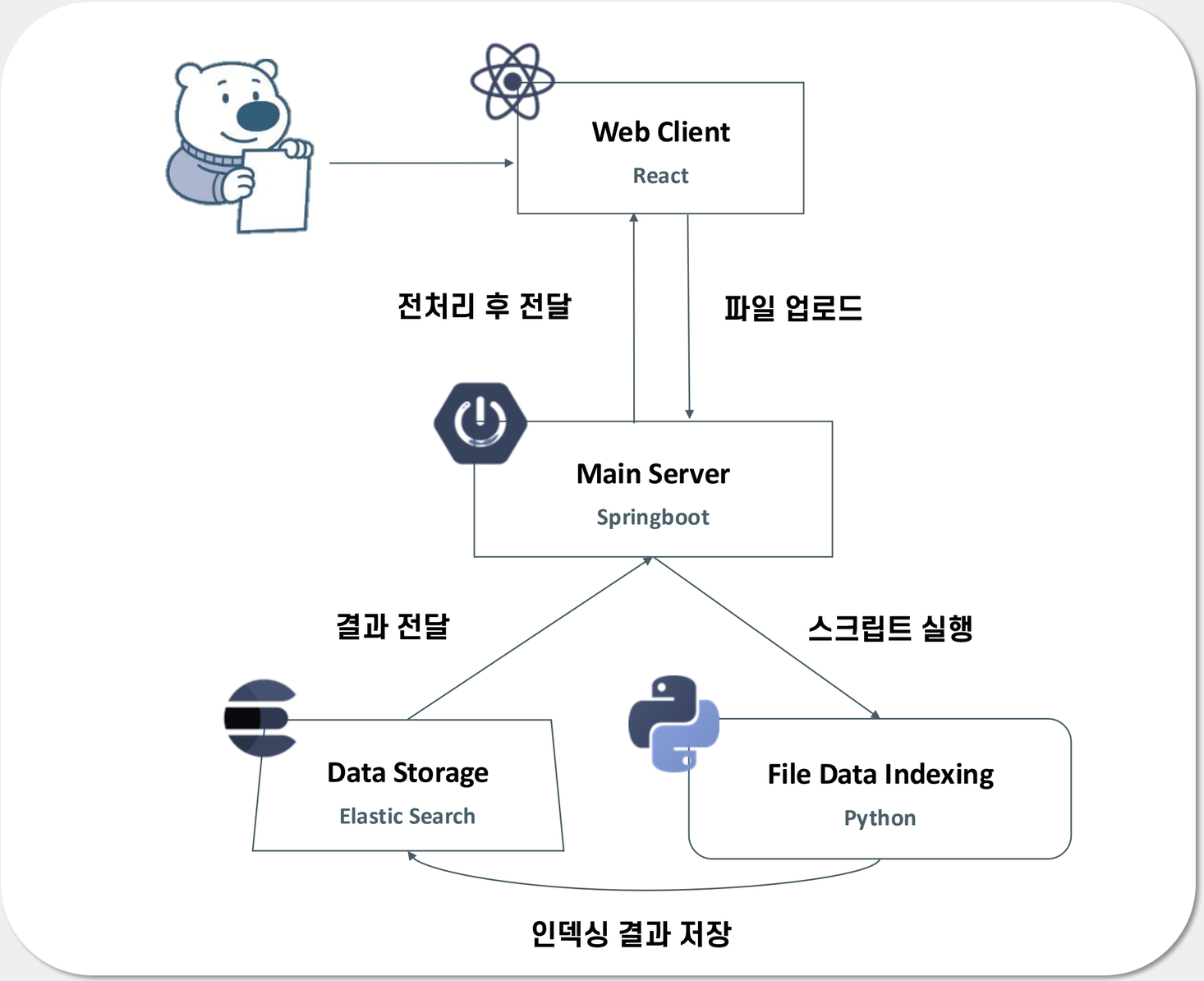
✅ 데이터 전처리 서버 연동 구현 완료

✅ 키워드 시각화 트리 구현 완료

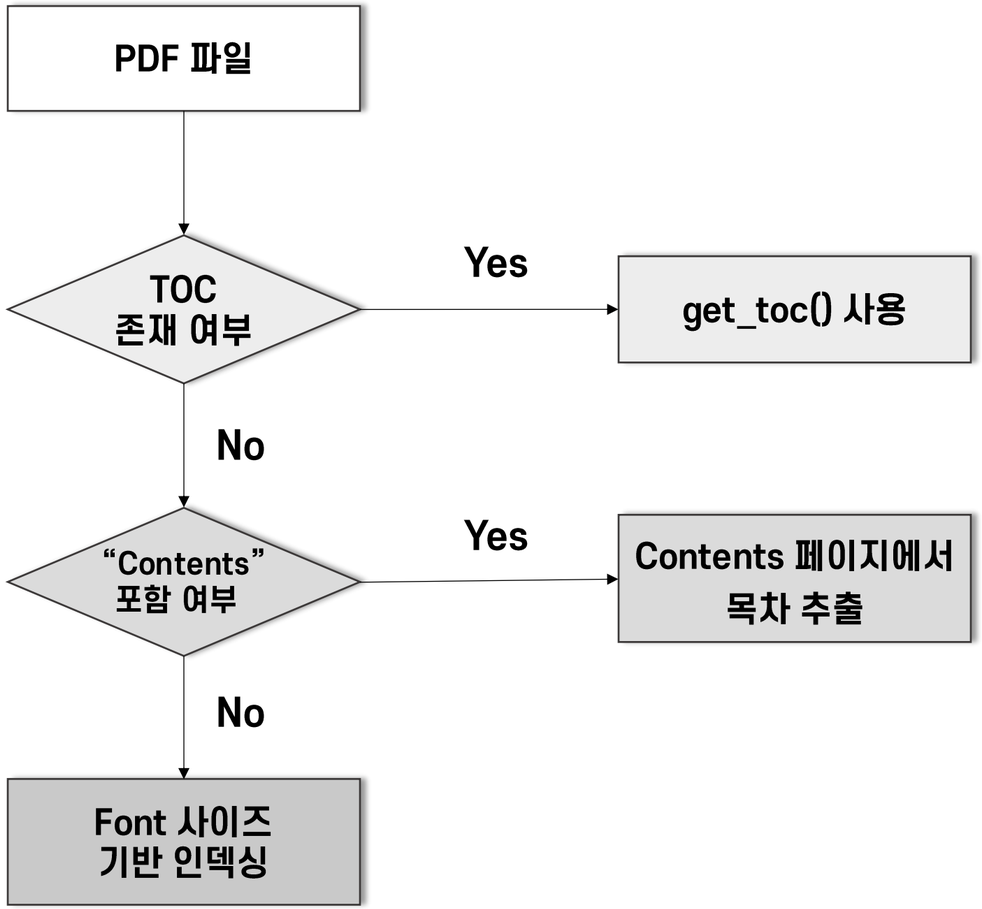
✅ 사내 데이터 접근 문제로 실제 데이터는 사용하지 못하였으나, Mock 데이터를 사용해 전체 기능 구현 완료

4. 핵심 기술

아키텍처 구조도



4. 핵심 기술 – 키워드 인덱싱



Indexing flow

1. **PDF 파일 입력**
2. **TOC 존재 여부 우선 확인**
   - PyMuPDF 라이브러리 제공 get_toc() 함수로 목차 추출
   - PDF 파일 내부에 저장된 목차를 읽어오는 함수
3. **TOC가 없으면, “Contents” 페이지 탐색**
   - “Contents”가 있는 페이지부터 1~2 페이지를 탐색해 목차 추출
4. **“Contents” 도 없다면, font 크기 기준 탐색**
   - 전체 문서에 대해 문서 구조를 분석해 목차 생성

4. 핵심 기술 – 키워드 인덱싱

“Contents” 기반 인덱싱

분석 기준

“Contents” 키워드 및 들여쓰기

적용 범위

“Contents” 검색해 “Contents” 등장 페이지를 contents_page로 설정

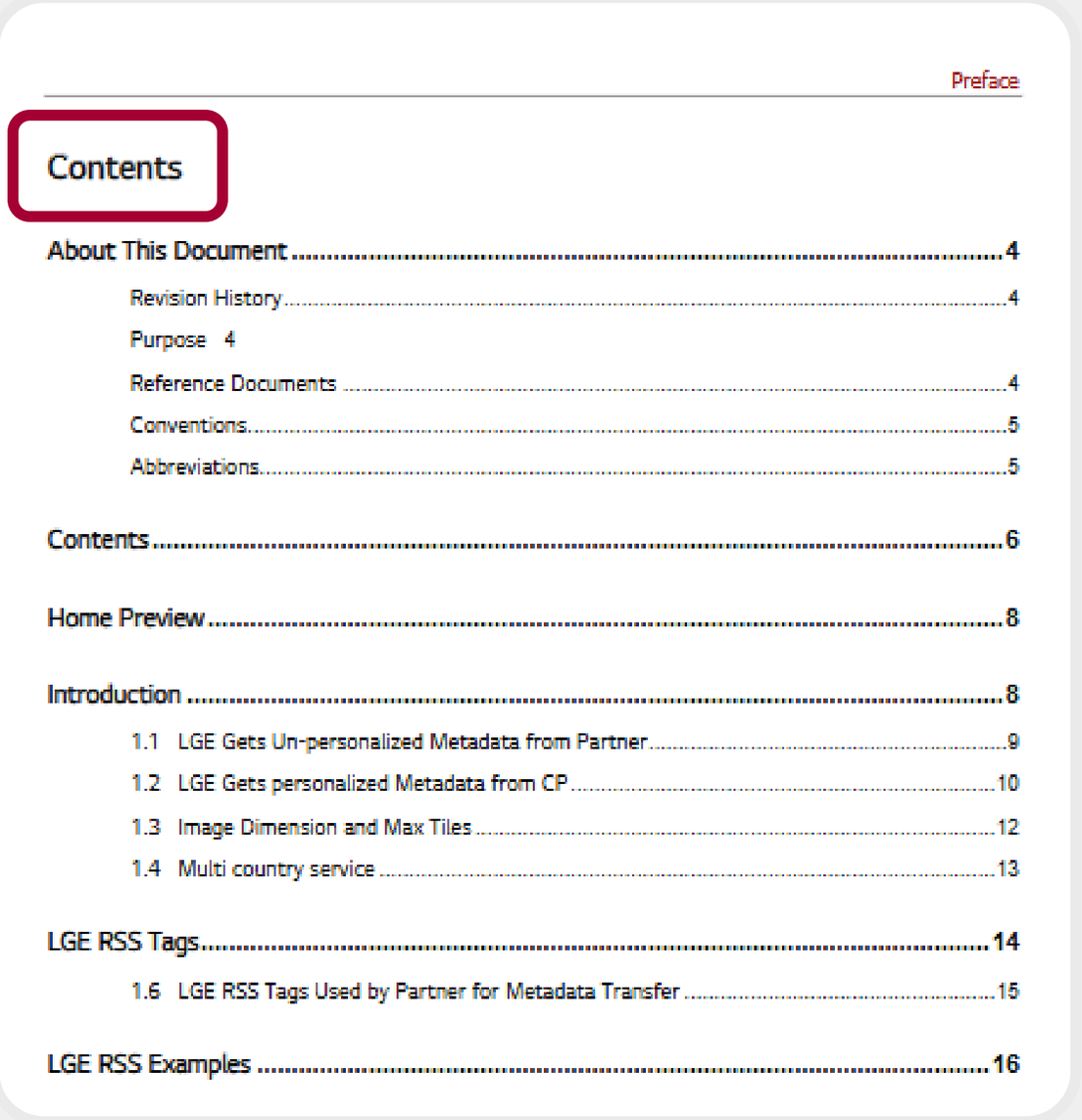
↓

다른 섹션 제목이 등장하면 contents_ends로 설정
없으면 contents_page+2페이지 까지 탐색

처리 방식

텍스트 블록 내 각 라인 순회하며

1. **들여쓰기 깊이(x0 좌표)**를 기반으로 목차 레벨 추정
   - 들여쓰기가 클수록 하위 레벨로 간주
2. 텍스트 내용에서 마지막 숫자를 **페이지 번호**로 인식
3. 제목에서 점이나 페이지 번호 제거해 순수 제목 텍스트 추출



Preface

Contents

4. 핵심 기술 – 키워드 인덱싱

Font 사이즈 기반 인덱싱

분석 기준

폰트 크기

적용 범위

모든 페이지를 대상으로 각 페이지에서 텍스트 블록을 읽고
Span 단위로 폰트 크기 (size)와 텍스트 수집

처리 방식

전체 수집된 폰트 크기에서

1. 가장 많이 등장한 크기를 본문 폰트 크기로 추정
   - Counter.most_common(1)[0]
2. 본문 폰트 크기보다 큰 폰트를 사용한 keyword로 추출
3. 큰 폰트 크기 순서대로 목차 레벨 부여

4. 핵심 기술 – PDF Viewer

React-pdf

기본적인 PDF 렌더링과 페이지 네비게이션 구현

장점

- React 컴포넌트 기반의 간단한 PDF 렌더링
- 기본적인 PDF 뷰어 기능 제공
- 상대적으로 가벼운 구현

한계

- react-pdf는 텍스트 레이어만 제공
  - 실제 pdf 원문 이미지와 텍스트 레이어가 일치하지 않아서 검색 및 하이라이트 기능 등에 한계
- 대용량 PDF 처리 시 메모리 관리가 제한적
- 제한된 커스터마이징 옵션

+

PDF.js

고급 PDF 기능 지원으로 복잡한 상호작용과 고급 기능에 사용

React-pdf 라이브러리의 한계 보완

- PDF 스펙의 모든 기능 지원
- 더 많은 커스터마이징 옵션
- 더 나은 성능과 메모리 관리

사용

- 텍스트 검색 및 하이라이트
- 페이지 내 검색 결과 네비게이션
- 페이지 렌더링 최적화
- 메모리 관리 및 성능 최적화
- PDF 메타데이터 처리

시연 영상

https://youtu.be/564nu0UX5uE

5. 결과 분석 및 기대 효과

결과 분석

- 인덱싱 고도화 후 메타데이터(TOC)가 없는 파일에 대한 파일 전처리도 가능해 짐
  - 기존 최상위 키워드 1개만 추출가능 -> 하위 3계층 목차까지 추출가능
- 500P 분량의 PDF 파일 업로드 후 페이지 랜더링까지 5.4s 소요

기대 효과

- 사내 프로그램으로 대외비 문서도 업로드 하여 사용 가능
- 목차 키워드 및 트리 제공으로 문서 구조에 대한 이해도 증진에 기여
- 실제 업무 담당자 정보와 Issue 정보를 사용할 수 있게 되면 비약적인 업무 소요 시간 단축 가능
- 사내 AI인 엘지니의 기능 중 하나로 추가 가능

6. 향후 연구 과제

AI를 활용한 서비스 고도화

- AI를 활용한 본문 요약 기능 추가로 사용자 편의성 개선
- 번역 기능 추가로 다국어 지원

사내 데이터와의 연계

- 실제 문서 상 업무를 담당하는 담당자 정보와 해당 업무와 관련된 이슈 정보를 제공해 업무 소요 시간 획기적 단축

범용성

- PDF 외의 확장자를 가진 파일(워드 등)로 대상 파일 지원을 넓혀 사용성 개선

7. 프로젝트 수행 후기

현업에 실제로 필요한 주제를 선정하여 프로젝트를 수행해서 재밌었습니다.
생성형 AI 기반의 프로젝트 경험은 처음이어서, 효율적으로 프로그래밍하는데 많은 도움이 되었습니다.

다른 부서의 연구원 분들과 현업에서의 경험을 나누며 이를 프로젝트 아이디어로 이어가는 과정 자체가 값진 과정이었던 것 같습니다.

기획부터 개발까지 현업에서 필요로 했던 서비스를 팀원들과 함께 개발할 수 있어서 의미 있는 시간이었습니다.

실습 기반 과정의 데이터 처리 및 시각화 학습과, 단기간이었지만 배운 내용을 기반으로 프로젝트를 진행할 수 있어 좋은 시간이었습니다.

감사합니다 !!