2차 리포트

전략적 제품 기획서: AI 기반 부동산 계약 리스크 분석기 (Text-Input Ver.)

1. 기획 방향성 변화: OCR 제거와 NLP 집중

1.1 변경된 핵심 가치: "정확성"과 "비교 분석"

OCR(광학 문자 인식) 기능을 제외함에 따라, 사용자는 계약서의 조항을 직접 타이핑하거나 복사해서 입력하게 됩니다. 이는 사용자 경험(UX) 측면에서 입력 장벽이 생길 수 있지만, 반대로 '데이터의 정확성(Data Quality)'이 보장된다는 강력한 장점이 있습니다. OCR의 오인식 문제(노이즈)가 사라지므로, 팀은 **'입력된 텍스트의 법률적 의미 해석'**에 모든 기술적 역량을 집중해야 합니다.

따라서 본 프로젝트의 새로운 기술적 차별점은 **"검색 증강 생성(RAG)의 정확도 극대화"**와 **"표준 계약서와의 정밀 비교"**가 되어야 합니다.

1.2 타겟 프로세스 재정의

- **Before (OCR 포함):** 사진 촬영 -> 텍스트 변환 -> 분석
- **After (Text 입력):** 조항 복사/붙여넣기 -> 표준 조항과 유사도 비교(**Similarity Check**) -> 위험도 분석 -> 솔루션 제공

---

2. 데이터 전략: 민법 데이터셋 (Whitelist) 및 표준 계약서

2.1 필수 학습 데이터 (RAG Knowledge Base)

민법 전체를 학습시키는 대신, 임대차 분쟁과 직결된 핵심 조항만을 선별하여 학습시킵니다. 이는 LLM이 불필요한 법률 지식으로 인해 환각(Hallucination)을 일으키는 것을 방지합니다.

1. 주택임대차보호법 및 상가건물임대차보호법: 전체 조항 (시행령 포함)
2. 민법 선별 조항 **(Whitelist):**
   - 제**615**조, 제**654**조: 원상회복의무 (못 자국, 벽지 훼손 등 퇴거 시 분쟁)
   - 제**623**조: 임대인의 수선 의무 (누수, 보일러 고장 등)[1]
   - 제**628**조, 제**640**조: 차임증감청구권 및 연체로 인한 해지 (월세 밀렸을 때)
3. 특약 사항 데이터: "반려동물 금지", "퇴거 시 청소비 지급" 등 시중에서 통용되는 특약 문구와 이에 대한 유불리 해설 데이터 수집.

2.2 표준 계약서 데이터 활용 전략

사용자가 입력한 특약이 국토교통부 표준 임대차계약서의 내용과 얼마나 다른지 비교해주는 것이 핵심 기능입니다. 표준 계약서의 각 조항을 임베딩하여 Vector DB에 저장해두고, 사용자

입력값과 가장 유사한 표준 조항을 찾아 비교해줍니다.

---

3. 기술 아키텍처: RAG 고도화 및 Reranker 도입

OCR이 빠진 자리를 **Advanced RAG** 기술로 채워 기술적 깊이를 증명합니다. 단순한 유사도 검색(Vector Search)은 법률 용어의 미묘한 차이를 놓칠 수 있으므로, **Cross-Encoder** 기반의 **Reranker**를 도입합니다.

3.1 시스템 아키텍처 (Revised)

구성 요소	기술 스택	설명
Frontend	**Streamlit**	st.text_area를 통해 조항 입력, st.diff 등을 활용해 표준 조항과 비교 화면 구현.
Embedding	**BGE-M3** 또는 **Ko-SBERT**	한국어 법률 문장에 특화된 임베딩 모델 사용.[2]
Retrieval	**ChromaDB** + **Hybrid Search**	키워드 검색(BM25)과 벡터 검색(Vector)을 결합하여 법률 용어 검색 정확도 보완.[4]
Reranking	**Cross-Encoder (BGE-Reranker)**	**[**핵심 기술**]** 1차로 검색된 문서들을 질문과의 연관성 순으로 재정렬하여 LLM의 판단 정확도를 비약적으로 향상시킴.[4]
LLM	**OpenAI GPT-4o** 또는 **Solar-10.7B**	입력된 조항의 위험성을 최종 판단하고 설명 생성.

3.2 Cross-Encoder Reranker 도입의 의의 (포트폴리오 포인트)

일반적인 RAG(Bi-Encoder)는 질문과 문서를 따로 벡터화하여 거리를 계산하므로 속도는 빠르지만 정확도가 다소 떨어집니다. 반면, **Cross-Encoder**는 질문과 문서를 쌍(Pair)으로 입력받아 심층적인 관계를 계산하므로 정확도가 훨씬 높습니다.[5]

- 포트폴리오 스토리: "초기 모델에서 법률 용어 검색 정확도가 낮아, 검색 후 상위 5개 문서에 대해 Cross-Encoder로 재순위화(Reranking)를 수행하여 RAGAS 정확도 점수를

0.7에서 0.85로 향상시켰습니다."

---

4. 사용자 경험(UX) 및 기능 상세

4.1 입력 인터페이스: "어떤 조항이 걱정되시나요?"

OCR이 없으므로 사용자가 텍스트를 입력하기 쉽게 유도해야 합니다.

1. 카테고리 선택: "수리/파손", "계약 해지", "보증금 반환", "특약 사항" 중 버튼 선택.
2. 텍스트 입력: "계약서의 해당 부분을 복사해서 붙여넣거나 직접 입력하세요." (예시 텍스트 제공)
3. 프리셋 질문: "집주인이 도배를 안 해준대요", "월세를 2달 밀리면 쫓겨나나요?" 같은 자연어 질문도 허용.

4.2 분석 결과 리포트 (Comparison UI)

단순 텍스트 답변 대신 구조화된 결과를 보여줍니다.

- 사용자 입력: "모든 수선 비용은 임차인이 부담한다."
- **AI** 분석 결과: 🔴 위험 (불리한 조항)
- 근거 법령: 민법 제623조 (임대인의 수선 의무) 및 대법원 판례 인용.
- 표준 조항 비교: "표준 계약서에는 '난방, 상하수도 등 주요 설비는 임대인이 수리한다'고 되어 있습니다. 이 특약은 임차인에게 과도하게 불리하므로 무효일 가능성이 큽니다."

---

5. 팀원 R&R (역할 분담) 및 일정 (2주 스프린트)

OCR 담당자의 역할을 **"Advanced RAG & Evaluation"**으로 변경하여 팀 밸런스를 맞춥니다.

5.1 역할 분담 (4인)

1. **PM & Frontend (팀장):**
   - Streamlit UI 개발 (채팅창, 비교 UI, 사이드바 히스토리).
   - 전체 일정 관리 및 통합 테스트.
2. **AI Engineer A (Data & Basic RAG):**
   - 주택/상가임대차보호법, 민법 필수 조항 데이터 수집 및 청킹(Chunking).
   - Vector DB (Chroma) 구축 및 기본 검색(Retriever) 구현.
3. **AI Engineer B (Advanced RAG - 舊 OCR 담당):**
   - **Reranking** 모듈 구현: 검색된 법령의 순위를 재조정하는 Cross-Encoder 도입.[5]
   - **Prompt Engineering:** 페르소나 설정("당신은 임차인 전문 변호사입니다") 및 답변 포맷(JSON 등) 최적화.
4. **Data Analyst & QA:**
   - 테스트 데이터셋 구축: "독소 조항 예시 50선"을 만들어 모델 성능 평가.
   - **RAGAS** 평가: Faithfulness(충실성), Answer Relevancy(관련성) 지표 측정 및 리포트

작성.

5.2 상세 일정 **(WBS)**

일차	주요 활동 **(Key Activities)**	비고
Day 1	기획 확정, Streamlit 환경 세팅, Github Repo 생성	Kick-off
Day 2	법령 데이터(민법/특별법) 수집 및 전처리, 임베딩 모델 선정	데이터셋 구축
Day 3	기본 RAG 파이프라인(LangChain + Chroma) 구현	Basic RAG
Day 4	**[**심화**]** Reranking 모델(Cross-Encoder) 테스트 및 적용	기술 차별화
Day 5	Streamlit UI 연동 (입력창 -> RAG -> 답변 출력)	MVP 1차 완성
Day 6	특약 사항 데이터셋(독소 조항) 추가 학습 및 프롬프트 튜닝	성능 개선
Day 7	**RAGAS** 정량 평가 및 결과 분석 (파라미터 조정)	객관적 지표 확보
Day 8	UI 고도화 (표준 조항 비교 뷰, 법적 고지/면책 조항 팝업)	UX 개선
Day 9	최종 버그 수정, 시연 시나리오 리허설	최종 점검
Day 10	포트폴리오 문서(README, 기술 블로그) 작성 및	문서화

	프로젝트 종료	

---

6. 포트폴리오 핵심 전략 (Job Interview 용)

취업 시 이 프로젝트를 어필할 때, OCR이 빠진 것을 약점이 아닌 **"선택과 집중"**의 결과로 설명해야 합니다.

1. **"Data-Centric AI"** 접근: 이미지 처리에 쓸 리소스를 데이터의 품질과 검색 정확도에 투자했습니다. 민법과 특별법의 우선순위를 고려하여 데이터를 선별(Whitelisting)했습니다.
2. **Advanced RAG** 기술: 단순 검색이 아닌 **Hybrid Search**와 **Reranking**을 적용하여, 법률 도메인의 복잡한 질의에 대해 Top-1 정확도(Precision@1)를 00% 향상시켰습니다.
3. 정량적 평가 도입: 감으로 개발한 것이 아니라, **RAGAS** 프레임워크를 통해 답변의 신뢰성을 수치화하고 개선했습니다.

이 구성은 주니어 레벨에서 보여줄 수 있는 가장 이상적인 "문제 해결 중심"의 AI 프로젝트 형태입니다.

참고 자료

1. st.warning - Streamlit Docs, 1월 13, 2026에 액세스, https://docs.streamlit.io/develop/api-reference/status/st.warning
2. 조문정보 | 국가법령정보센터, 1월 13, 2026에 액세스, http://www.law.go.kr/lsLinkProc.do?lsClsCd=L&lsNm=%EB%AF%BC%EB%B2%95&lsId=prec20161118&joNo=064000&efYd=20161118&mode=11&lnkJoNo=undefined
3. 조문정보 | 국가법령정보센터, 1월 13, 2026에 액세스, http://law.go.kr/LSW/lsLawLinkInfo.do?lsJoLnkSeq=900158553&chrClsCd=010202
4. RAG Chatbot With HuggingFace And Streamlit: Complete Tutorial - Codecademy, 1월 13, 2026에 액세스, https://www.codecademy.com/article/aichatbot-using-huggingface-rag-streamlit
5. Build and Deploy a Chatbot App Using Streamlit and OpenAI | Dataquest Project Lab, 1월 13, 2026에 액세스, https://www.youtube.com/watch?v=palgG8e8LHk
6. When to Choose Local LLMs vs APIs: A Founder's Real-World Guide, 1월 13, 2026에 액세스, https://thebootstrappedfounder.com/when-to-choose-local-llms-vs-apis-a-founders-real-world-guide/
7. Technical Analysis of Modern Non-LLM OCR Engines - IntuitionLabs, 1월 13, 2026에 액세스, https://intuitionlabs.ai/articles/non-llm-ocr-technologies

1차 리포트

전략적 제품 기획서: AI 기반 부동산 계약 및 분쟁 예방 어시스턴트 구축 가이드

1. 서론: 프롭테크(PropTech)와 리걸테크(LegalTech)의 융합

1.1 시장 배경 및 문제 정의

대한민국의 부동산 시장, 특히 주거용 임대차 시장은 구조적인 정보 비대칭성(Information Asymmetry)이 극심한 영역 중 하나입니다. 임대인과 임차인 사이의 권력 관계는 단순히 경제적 자산의 차이에서 기인하는 것을 넘어, 법률적 지식의 격차에서 비롯되는 경우가 많습니다. 최근 사회적 이슈가 된 전세 사기나 깡통 전세 문제, 그리고 빈번하게 발생하는 보증금 반환 지연 사태는 임차인이 계약 체결 단계에서 특약 사항의 독소 조항을 식별하지 못하거나, 거주 중 발생한 분쟁 상황에서 자신의 법적 권리를 적시에 행사하지 못해 피해가 확대되는 경향을 보입니다.[1]

이러한 상황에서 주니어 레벨의 개발팀이 2주라는 단기간 내에 구축하고자 하는 "부동산 계약 및 분쟁 예방 챗봇"은 단순한 기술적 토이 프로젝트를 넘어, 사회적 효용성이 매우 높은 리걸테크(LegalTech) 솔루션으로서의 잠재력을 가집니다. 특히 대학생이나 사회 초년생과 같은 1인 가구는 전문 변호사나 법무사의 조력을 받기에는 비용적 장벽이 높기 때문에, AI를 통해 1차적인 법률 필터링을 제공하는 서비스는 명확한 타겟 유저와 시장 수요(Product-Market Fit)를 확보할 수 있습니다.

본 기획서는 4인 구조의 팀이 AI+X 수업의 일환으로 진행하는 프로젝트의 성공적인 수행을 위해, 단순한 기능 구현을 넘어 취업용 포트폴리오로서의 차별성을 확보할 수 있는 심층적인 개발 전략을 제시합니다. 특히 사용자가 우려하는 계약서 조항을 입력했을 때 단순히 검색 결과를 보여주는 것을 넘어, 해당 조항의 위험성을 판단하고 조언하는 '판단형 AI'를 구축하기 위해 검색 증강 생성(RAG, Retrieval-Augmented Generation) 기술과 광학 문자 인식(OCR) 기술을 유기적으로 결합하는 방안을 모색합니다.

1.2 프로젝트 목표 및 핵심 가치 제안

본 프로젝트의 핵심 목표는 "법률적 판단의 민주화"와 "계약 리스크의 사전 차단"으로 정의할 수 있습니다. 기존의 챗봇들이 사전에 정의된 시나리오에 따라 단답형 정보를 제공하는 데 그쳤다면, 본 프로젝트는 생성형 AI의 추론 능력을 활용하여 개별 계약 상황에 맞는 맞춤형 조언을 제공하는 것을 목표로 합니다.

이를 달성하기 위한 핵심 가치 제안(Value Proposition)은 다음과 같습니다. 첫째, 즉각적인 계약서 독소 조항 필터링입니다. 사용자가 스마트폰으로 촬영한 계약서 이미지를 업로드하면, OCR 기술이 이를 텍스트로 변환하고, RAG 모델이 표준 임대차 계약서 및 주택임대차보호법과 대조하여 임차인에게 불리한 특약(예: "모든 수선 의무는 임차인에게 있다")을 식별해냅니다.[2]

둘째, 분쟁 상황별 법률 가이드 제공입니다. 누수, 곰팡이, 소음 등 거주 중 발생하는 다양한 문제에 대해, 민법과 특별법의 우선순위를 고려한 정확한 대응 매뉴얼을 제시합니다.

이 과정에서 가장 중요한 전략적 의사결정은 데이터의 범위(Scope of Data)를 설정하는 것입니다. 민법의 방대한 양을 고려할 때 이를 제외하고자 하는 초기 기획은 합리적인 접근이나, 분쟁 예방이라는 목적을 달성하기 위해서는 민법의 완전한 배제보다는 '선별적 수용'이 필수적입니다. 본 보고서는 이러한 데이터 전략을 포함하여, 머신러닝 요소의 도입, 웹 애플리케이션 아키텍처, 그리고 팀 매니지먼트 전략까지 포괄적으로 다룹니다.

---

2. 법률 도메인 데이터 전략**:** 민법 포함 여부에 대한 의사결정

2.1 법체계의 계층 구조와 특별법 우선의 원칙

사용자의 초기 기획안에서 제기된 가장 핵심적인 질문은 "민법을 학습 데이터에서 제외할 것인가?"입니다. 이에 대한 결론부터 제시하자면, **"전면적인 제외는 서비스의 품질을 치명적으로 저하하므로, 임대차 관련 필수 조항만을 선별하여 포함해야 한다(Whitelist Strategy)"**가 정답입니다.

법률 시스템은 일반법과 특별법의 관계로 구성되어 있으며, "특별법 우선의 원칙(Lex specialis derogat legi generali)"이 적용됩니다. 주택임대차보호법과 상가건물임대차보호법은 민법의 임대차 규정에 대한 특례를 규정한 특별법입니다. 따라서 대항력, 우선변제권, 계약갱신요구권 등 임차인의 핵심 권리는 이 두 특별법이 민법보다 우선하여 적용됩니다.[1]

그러나 특별법은 말 그대로 '특별한' 상황이나 보호가 필요한 영역을 규율하기 위해 만들어진 법입니다. 즉, 임대차 계약의 기본이 되는 성립, 효력, 그리고 일반적인 권리와 의무는 여전히 민법이 규율하고 있습니다. 만약 챗봇이 주택임대차보호법만 학습한다면, "보증금을 못 받았을 때 어떻게 해야 하나요?"라는 질문에는 임차권등기명령(주택임대차보호법 제3조의3)을 인용하여 답변할 수 있지만, "천장에서 물이 새는데 집주인이 수리를 안 해줘요"라는 질문에는 답변할 근거 법령을 찾지 못하게 됩니다. 수선 의무는 특별법이 아닌 민법 제623조에 규정되어 있기 때문입니다.[1]

2.2 민법 데이터 선별 전략 **(Whitelist Approach)**

민법 전체를 RAG에 학습시키는 것은 데이터의 노이즈(Noise)를 증가시켜 검색 정확도(Precision)를 떨어뜨릴 위험이 있습니다. 챗봇이 임대차 질문에 대해 친족상속법이나 물권법의 엉뚱한 조항을 가져오는 환각(Hallucination) 현상을 방지하기 위해, 임대차 계약과 직접적으로 연관된 조항만을 선별하여 벡터 데이터베이스(Vector DB)에 인덱싱해야 합니다.

다음은 본 프로젝트에 반드시 포함되어야 할 민법의 핵심 조항 리스트와 그 선정 근거입니다.

조항 번호	조항명	선정 근거 및 챗봇 활용 시나리오
제**615**조	임차인의 원상회복의무	퇴거 시 가장 빈번한 분쟁인 '원상복구 범위'에 대한 기준 제공. 못자국이나 벽지 오염 논쟁 시 필수 인용.[5]
제**618**조	임대차의 의의	임대차 계약의 정의. 사용·수익의 대가로 차임을 지급한다는 기본 구조 설명.[1]
제**623**조	임대인의 의무	**[**핵심**]** 임대인의 수선 의무 규정. 곰팡이, 누수, 보일러 고장 등 시설물 하자에 대한 책임 소재 판단의 근거.[3]
제**626**조	임차인의 상환청구권	임차인이 자비로 수리한 비용(필요비, 유익비)을 청구할 수 있는 권리 설명.[1]
제**627**조	일부멸실 등과 감액청구	태풍, 화재 등으로 집의 일부를 못 쓰게 되었을 때 월세 감액을 요구할 수 있는 근거.[1]
제**640**조	차임연체와 해지	월세가 2기(2달치) 연체되었을 때 계약 해지 가능성을 경고. 상가(3기)와 주택(2기)의 차이를 설명할 때 비교군으로 사용.[7]
제**390**조	채무불이행과 손해배상	계약 위반에 따른 손해배상 청구의 일반 원칙. 특약 사항 위반 시 대응 논리 구성.[5]
제**750**조	불법행위의 내용	임대인이나 임차인의 고의/과실로 인한 손해(예: 화재) 발생 시 책임 소재 판단.[5]

이 조항들을 제외하면, 사용자가 "집주인이 도배를 안 해줘요"라고 물었을 때, 챗봇은 "관련

법령을 찾을 수 없습니다"라고 답하거나 엉뚱한 특별법 조항을 들이댈 것입니다. 따라서 데이터 엔지니어링 단계에서 이 조항들만을 추출하여 별도의 '필수 민법 데이터셋'을 구축하는 것이 프로젝트의 완성도를 결정짓는 중요한 요소가 됩니다.

2.3 데이터 전처리 및 청킹(Chunking) 전략

법률 문서는 일반적인 자연어 텍스트와 다른 구조적 특징을 가집니다. 제1조, 제2항, 제3호와 같이 위계질서가 명확하고, 각 조항이 독립적인 의미 단위를 구성합니다. 따라서 일반적인 고정 길이 청킹(Fixed-size Chunking, 예: 500자 단위 자르기)을 적용할 경우, 하나의 조항이 중간에 잘려 문맥이 손실될 우려가 큽니다.

본 프로젝트에서는 '조항 단위 의미론적 청킹(Article-based Semantic Chunking)' 전략을 채택해야 합니다.

1. 메타데이터 태깅(Metadata Tagging): 각 청크(Chunk)에 법령명, 조항 번호, 핵심 키워드를 메타데이터로 부여합니다.
   - 예: { "text": "임대인은 목적물을 임차인에게 인도하고...", "metadata": { "law": "민법", "article": "623", "topic": ["수선의무", "하자보수"] } }
2. 계층적 인덱싱(Hierarchical Indexing): 검색 시 상위 개념(예: 수선 의무)을 먼저 찾고, 그에 해당하는 구체적 조항(민법 623조 vs 특약 사항)을 하위에서 탐색하는 구조를 설계합니다.

이러한 데이터 전략은 RAG 시스템이 단순히 문자의 유사도가 높은 텍스트를 가져오는 것을 넘어, 질문의 법적 맥락(Legal Context)을 이해하고 정확한 근거 조항을 인용할 수 있게 만듭니다.

---

3. 기술 아키텍처: RAG 파이프라인과 OCR 고도화

3.1 전체 시스템 아키텍처 개요

본 프로젝트의 기술 스택은 주니어 팀의 역량과 2주라는 제한된 시간을 고려하여 '생산성'과 '성능'의 균형을 맞추는 방향으로 설계되어야 합니다. 전체 시스템은 크게 입력 레이어(Streamlit UI), 전처리 레이어(OCR), 추론 레이어(RAG & LLM), 그리고 **데이터 레이어(Vector DB)**로 구성됩니다.

구성 요소	추천 기술 스택	선정 이유
Frontend/UI	**Streamlit**	Python 기반으로 빠른 프로토타이핑 가능, 파일 업로드 및 채팅 인터페이스 구현 용이.[11]

OCR	**PaddleOCR**	한국어 인식률 우수, 표(Table) 구조 인식 기능 제공, 오픈소스로 비용 부담 없음.[2]
LLM	**OpenAI GPT-4o (API)** 또는 **Solar-10.7B (Local)**	복잡한 법률 추론에는 높은 지능 필요. 초기엔 API로 개발하고, 후반에 로컬 모델로 전환 시도.[16]
Vector DB	**ChromaDB**	로컬 파일 기반으로 작동하여 서버 구축 불필요, LangChain과의 높은 호환성.[18]
Framework	**LangChain**	RAG 파이프라인 구축을 위한 풍부한 도구 제공, 다양한 리트리버(Retriever) 지원.[20]

3.2 OCR 파이프라인: 계약서 이미지의 디지털화

사용자가 업로드하는 부동산 임대차 계약서는 정형화된 서식이지만, 사진 촬영 상태에 따라 기울어짐, 조명 반사, 그림자 등의 노이즈가 포함될 수 있습니다. 또한 계약서는 '표(Grid)' 형태로 되어 있어 단순 텍스트 추출만으로는 "보증금"이라는 키워드 옆에 있는 숫자가 보증금액인지 월세인지 구분하기 어렵습니다.

따라서 Tesseract와 같은 고전적인 OCR 엔진보다는 딥러닝 기반의 **PaddleOCR** 사용을 강력히 권장합니다. PaddleOCR은 텍스트 검출(Detection)과 인식(Recognition)이 분리된 2-Stage 구조를 가지며, 특히 한국어 모델이 경량화되어 있어 CPU 환경에서도 준수한 속도를 보여줍니다.[2]

OCR 처리 프로세스**:**

1. 이미지 전처리**:** OpenCV를 활용하여 이미지를 흑백(Grayscale)으로 변환하고, 이진화(Binarization) 처리를 통해 글자의 선명도를 높입니다. getPerspectiveTransform 등을 이용해 기울어진 문서를 바로잡습니다.
2. 구조화된 데이터 추출**:** PaddleOCR의 결과값은 텍스트와 좌표(Bounding Box)로 나옵니다. 이 좌표 정보를 활용하여 같은 Y축 선상에 있는 텍스트들을 하나의 행(Row)으로 인식하게 하는 후처리 알고리즘을 작성해야 합니다.
3. **LLM** 기반 파싱**:** OCR로 추출된 원시 텍스트(Raw Text)를 LLM에 프롬프트와 함께 전달하여 JSON 형태로 구조화합니다.
    - 프롬프트 예시: "다음은 OCR로 추출된 임대차 계약서 텍스트야. 여기서 [보증금, 월세, 임대기간, 특약사항]을 추출해서 JSON 형식으로 출력해줘."

이 과정은 사용자가 입력한 계약서의 내용을 기계가 이해할 수 있는 데이터로 변환하는 핵심 단계이며, 이후 RAG 시스템이 특약 사항의 유불리를 판단하는 기초 자료가 됩니다.

3.3 RAG 검색 성능 최적화: 하이브리드 검색(Hybrid Search)

법률 용어는 매우 구체적이고 배타적입니다. 예를 들어 "계약갱신요구권"과 "묵시적 갱신"은 비슷해 보이지만 법적 효력은 완전히 다릅니다. 벡터 유사도 기반의 시맨틱 검색(Semantic Search)만 사용할 경우, 의미적으로 유사한 텍스트를 찾아줄 수는 있지만, 정확한 법률 용어가 포함된 조항을 놓칠 수 있습니다. 반면 키워드 기반 검색(BM25)은 정확한 단어를 찾지만 문맥을 놓칠 수 있습니다.

따라서 본 프로젝트에서는 이 두 가지 방식을 결합한 하이브리드 검색**(Hybrid Search)** 전략을 사용해야 합니다. LangChain의 EnsembleRetriever를 사용하면 BM25 검색 결과와 벡터 검색(FAISS/Chroma) 결과를 가중 평균하여 최적의 문서를 추출할 수 있습니다.[22]

또한, 검색된 문서들 중에서 가장 연관성이 높은 문서를 다시 상위로 올리는 리랭킹**(Reranking)** 과정을 추가하면 정확도를 비약적으로 높일 수 있습니다. 오픈소스 리랭커 모델인 BAAI/bge-reranker-v2-m3 등을 활용하면 한국어 법률 문서에서도 우수한 성능을 기대할 수 있습니다.

---

4. 머신러닝 및 딥러닝 요소 심화: Local LLM 도입 전략

프로젝트의 요구사항 중 "머신러닝 혹은 딥러닝을 가미할 수 있으면 더 좋다"는 부분은 팀의 기술적 역량을 보여줄 수 있는 중요한 차별화 포인트입니다. 단순히 OpenAI의 API를 호출하는 것은 '개발(Development)'에 가깝지만, 오픈소스 LLM을 직접 구동하고 최적화하는 것은 '엔지니어링(Engineering)'의 영역입니다.

4.1 Local LLM 선정: Solar-10.7B

취업 포트폴리오로서 "우리는 API 비용 절감과 데이터 보안을 위해 로컬 모델을 도입했습니다"라는 스토리는 매우 매력적입니다. 이를 위해 한국어 성능이 뛰어나면서도 단일 GPU(Colab T4 등)에서 구동 가능한 모델로 **Upstage**의 **Solar-10.7B-Instruct-v1.0**을 추천합니다.[17]

- 선정 이유: Solar-10.7B는 100억 개 파라미터 수준의 경량 모델임에도 불구하고, 심층 업스케일링(Depth Up-Scaling) 기술을 적용하여 300억 개 이상의 파라미터를 가진 모델들과 대등한 성능을 보입니다. 특히 한국어 지시 이행 능력(Instruction Following)이 탁월하여 RAG 시스템의 답변 생성기로 적합합니다.[25]

4.2 양자화(Quantization) 및 추론 최적화

주니어 개발 환경에서는 고성능 GPU 서버를 확보하기 어렵습니다. 따라서 모델을 4-bit 또는 8-bit로 **양자화(Quantization)**하여 메모리 사용량을 줄여야 합니다. llama.cpp 라이브러리와

GGUF 포맷을 활용하면 일반 소비자용 그래픽카드나 심지어 CPU 환경에서도 모델을 구동할 수 있습니다.

또한, **Ollama**와 같은 도구를 활용하면 로컬 모델을 마치 OpenAI API처럼 호출할 수 있는 로컬 서버를 쉽게 구축할 수 있습니다.[27] 이는 개발 과정에서 API 기반으로 먼저 로직을 짠 후, 엔드포인트(Endpoint) 주소만 변경하면 로컬 모델로 전환할 수 있게 해주어 개발 유연성을 높여줍니다.

4.3 (선택 심화) 파인튜닝(Fine-tuning) 전략

만약 팀의 역량이 허락한다면, 표준 임대차 계약서 데이터와 법률 상담 데이터를 활용해 Solar-10.7B 모델을 **PEFT(Parameter-Efficient Fine-Tuning)** 방식인 LoRA(Low-Rank Adaptation)로 미세 조정하는 것을 고려해볼 수 있습니다. 이는 "기존 LLM이 법률 용어를 더 잘 이해하도록 특화시켰다"는 강력한 포트폴리오 스토리가 되지만, 2주라는 시간을 고려할 때 우선순위는 낮게 설정하고, RAG 파이프라인이 완성된 후 여유가 있을 때 시도하는 것이 좋습니다.[28]

---

5. 사용자 경험(UX) 설계 및 리스크 분석 로직

5.1 사용자 페르소나(Persona) 정의

성공적인 서비스는 명확한 타겟 유저 정의에서 시작합니다. 본 프로젝트의 주 사용자를 다음과 같이 구체적인 페르소나로 설정합니다.[30]

- 페르소나 **A:** 사회초년생 '김불안' **(27**세**,** 직장인**)**
  - 상황**:** 생애 첫 전세 계약을 앞두고 있음. 뉴스에서 전세 사기 기사를 보고 매우 불안함.
  - 니즈**:** 어려운 법률 용어 대신 "이 계약서 안전함/위험함"과 같은 직관적인 결과를 원함. 특약 사항에 적힌 말이 나에게 불리한지 아닌지 즉시 확인하고 싶음.
  - 행동 패턴**:** 모바일로 계약서를 찍어 바로 확인하려 함. 긴 텍스트보다는 요약된 정보를 선호.

5.2 계약서 리스크 분석(Risk Assessment) 로직

단순한 Q&A 챗봇을 넘어, '분쟁 예방'이라는 목적을 달성하기 위해 챗봇은 능동적인 분석 결과를 제공해야 합니다. 이를 위해 OCR로 추출된 특약 사항을 다음과 같은 로직으로 분석합니다.

1. 조항 매핑**(Clause Mapping):** 추출된 특약 문구를 표준 임대차 계약서의 조항 및 관련 법령과 매핑합니다.
2. 유불리 판단**(Sentiment Analysis for Legal Risk):** LLM에게 다음과 같은 시스템 프롬프트를 주어 위험도를 3단계(안전/주의/위험)로 분류하게 합니다.
   - 프롬프트: "너는 임차인의 권익을 보호하는 변호사야. 다음 특약 사항이 주택임대차보호법이나 민법에 위반되거나 임차인에게 현저히 불리한 경우 '위험'으로, 다소 불리하지만 협의 가능한 경우 '주의'로 분류하고 그 이유를 설명해."

3. 결과 시각화**:** Streamlit의 st.error (빨강), st.warning (노랑), st.success (초록) 컴포넌트를 활용하여 신호등 형태로 직관적인 리포트를 보여줍니다.[32]

5.3 면책 조항 및 **UI** 안전장치

법률 정보를 제공하는 서비스는 잘못된 정보로 인한 리스크가 큽니다. 따라서 UX적으로 사용자의 과도한 신뢰를 방지하는 장치가 필수적입니다.

- 서비스 진입 시**:** "본 서비스는 인공지능을 활용한 자동 분석 결과를 제공하며, 변호사의 법률 자문을 대체할 수 없습니다. 최종 계약 체결 전 반드시 전문가와 상의하십시오."라는 팝업 동의를 받습니다.[33]
- 답변 하단**:** 모든 답변의 끝에는 "근거 법령: 주택임대차보호법 제3조"와 같이 출처를 명시하고, "AI 생성 결과로 오류가 있을 수 있습니다"라는 문구를 고정적으로 노출합니다.

---

6. 프로젝트 매니지먼트**: 2**주 스프린트 실행 계획

4명의 팀원이 2주(10일, 주말 제외) 동안 8시간씩 투입되는 프로젝트는 시간 관리가 생명입니다. 애자일(Agile) 방법론을 차용하여 1주 단위 스프린트로 진행하며, 각 팀원의 역할(R&R)을 명확히 합니다.

6.1 팀원 역할 분담 **(R&R)**

1. **PM & Full-stack (**팀장**):** 전체 일정 관리, Streamlit 프론트엔드 개발, 각 모듈(OCR, RAG) 통합, 배포 관리.
2. **AI Engineer A (RAG** 담당**):** 법률 데이터 수집 및 전처리(청킹), 벡터 DB 구축, 검색 알고리즘(Hybrid Search) 최적화.
3. **AI Engineer B (OCR & Model** 담당**):** PaddleOCR 파이프라인 구축, LLM 프롬프트 엔지니어링, Local LLM 서빙 환경 구성.
4. **Data Analyst & QA:** 테스트 케이스(Ground Truth) 작성, RAGAS 평가 진행, 사용자 시나리오 테스트, 최종 보고서 및 발표 자료 작성.

6.2 일자별 상세 실행 계획 **(WBS)**

주차	일자	주요 활동 **(Key Activities)**	마일스톤 **(Milestone)**
1주차	Day 1	**Kick-off:** 기획 확정, Git Repo 생성, 개발 환경 세팅(Virtualenv)	환경 설정 완료
	Day 2	데이터 수집**:**	데이터셋 구축 완료

		법령(민법 선별 포함) PDF/TXT 수집, 전처리, 청킹 전략 수립	
	Day 3	핵심 기능 개발 **1:** RAG 기본 파이프라인 구축(LangChain + Chroma), 임베딩 테스트	검색 기능 동작
	Day 4	핵심 기능 개발 **2:** PaddleOCR 연동, 계약서 이미지 인식 테스트, JSON 파싱	OCR 파이프라인 완료
	Day 5	통합 및 **MVP:** Streamlit에 RAG와 OCR 기능 통합, 기본 UI 구성	**MVP(**최소기능제품**)** 완성
2주차	Day 6	평가 및 개선**:** RAGAS 활용 정량 평가, 환각 현상 분석, 프롬프트 수정	성능 리포트 1차
	Day 7	기능 고도화**:** 하이브리드 검색 적용, 민법 데이터 보강, 리스크 분석 로직 정교화	검색 정확도 향상
	Day 8	**UX** 개선**:** 면책 조항 팝업, 로딩 애니메이션, 답변 스타일링(마크다운), 모바일 뷰 점검	UI 폴리싱 완료
	Day 9	최종 테스트**:** 팀원	최종 빌드 완성

		교차 테스트(Red Teaming), 버그 수정, Local LLM 적용 실험	
	Day 10	문서화 및 발표: 시연 영상 촬영, 포트폴리오 문서(README) 작성, 최종 회고	프로젝트 종료

---

7. 품질 보증 및 평가(Evaluation) 방법론

"잘 만들었습니다"라는 주관적인 주장보다는 "정확도가 85%입니다"라는 객관적인 데이터가 포트폴리오의 신뢰도를 높입니다. 이를 위해 **RAGAS(Retrieval Augmented Generation Assessment)** 프레임워크를 도입하여 정량적 평가를 수행해야 합니다.[35]

7.1 RAGAS 기반 정량 평가

RAGAS는 RAG 시스템의 성능을 '생성(Generation)'과 '검색(Retrieval)' 두 가지 측면에서 평가합니다.

- **Faithfulness (충실성):** 챗봇의 답변이 검색된 법령 문서에 근거하고 있는가? (환각 여부 판단)
- **Answer Relevancy (답변 관련성):** 챗봇의 답변이 사용자의 질문 의도에 부합하는가?
- **Context Precision (문맥 정확도):** 검색된 법령들이 실제로 질문과 관련된 내용인가?

한국어 평가를 위해서는 RAGAS의 평가 프롬프트를 한국어로 번역하여 적용하는 과정(adapt 메소드 활용)이 필요합니다.[37] 팀은 약 20~30개의 예상 질문-모범 답안 세트(Golden Dataset)를 만들고, 이를 바탕으로 점수를 측정하여 개발 전후의 성능 향상을 수치로 보여주어야 합니다.

7.2 정성 평가 및 레드 티밍(Red Teaming)

팀원들이 임대인, 악성 임차인 등 다양한 역할을 맡아 챗봇을 속이거나 공격적인 질문을 던지는 '레드 티밍'을 수행해야 합니다. 예를 들어 "월세 하루 밀렸는데 나가라고 해도 돼?"와 같이 법적으로 미묘하거나 오답을 유도하는 질문을 통해 챗봇의 방어 기제를 테스트하고, 시스템 프롬프트를 보완합니다.

---

8. 결론 및 포트폴리오 전략

본 프로젝트는 주니어 레벨에서 접근하기 어려운 '법률'이라는 도메인에 최신 AI 기술(RAG,

OCR, Local LLM)을 접목한 도전적인 과제입니다. 2주라는 짧은 기간 동안 완성도 높은 결과물을 내기 위해서는 **"선택과 집중"**이 필수적입니다. 민법의 방대한 데이터를 모두 다루려는 욕심을 버리고, 임대차와 직결된 조항만을 선별하여 학습시키는 데이터 전략이 프로젝트 성공의 열쇠가 될 것입니다.

또한, 단순히 기술 구현에 그치지 않고, **"사용자의 불안을 해소한다"**는 제품의 본질적인 가치에 집중하여 UX를 설계하고, RAGAS와 같은 객관적인 지표로 성능을 증명한다면, 취업 시장에서 매우 경쟁력 있는 포트폴리오가 될 것입니다. 특히 API 활용 능력과 로컬 모델 엔지니어링 능력을 동시에 보여주는 하이브리드 접근 방식은 'AI+X'라는 융합 인재상에 완벽히 부합하는 전략입니다.

이 보고서에 담긴 전략적 가이드라인을 바탕으로, 팀원들과 긴밀히 소통하며 실행에 옮긴다면 기술적으로 탄탄하고 사회적으로 의미 있는 서비스를 만들어낼 수 있을 것입니다.

참고 자료

1. 임차인의 권리: 민법, 주택임대차보호법, 상가임대차보호법 비교 - 로톡, 1월 13, 2026에 액세스, https://www.lawtalk.co.kr/posts/120241
2. OCR이란? OCR 프로그램 추천, 한글 OCR 오픈소스, AI OCR 활용 사례 4가지 - AI 스토어, 1월 13, 2026에 액세스, https://app.dalpha.so/blog/ai-ocr-guide/
3. 1월 13, 2026에 액세스, http://easylaw.go.kr/CSP/CnpClsMain.laf?popMenu=ov&csmSeq=629&ccfNo=4&cciNo=2&cnpClsNo=2#:~:text=%EC%9E%84%EB%8C%80%EC%9D%B8%EC%9D%80%20%EC%9E%84%EC%B0%A8%EC%9D%B8%EC%9D%B4%20%EB%AA%A9%EC%A0%81%EB%AC%BC,%EB%AF%BC%EB%B2%95%E3%80%8D%20%EC%A0%9C623%EC%A1%B0).
4. 「민법」에 따른 임대차와의 비교 - 엘파인드 생활 법령, 1월 13, 2026에 액세스, https://lfind.kr/statutes/627_1_2_3
5. 손해배상(기)・손해배상(기)(임차건물 화재로 인하여 임대차 목적물이 아닌 부분까지 불탄 경우 임차인의 손해배상책임의 성립과 손해배상의 범위가 문제된 사건) - 국가법령정보센터, 1월 13, 2026에 액세스, https://www.law.go.kr/precInfoP.do?mode=0&precSeq=184812&vSct=2012%EB%8B%A486895
6. 제7절 임대차 [제618조~제654조] - E-BOOK채권각론 - For Justice21 - Daum 카페, 1월 13, 2026에 액세스, https://cafe.daum.net/forjustice21/iHm7/14?svc=cafeapi
7. 조문정보 | 국가법령정보센터, 1월 13, 2026에 액세스, https://www.law.go.kr/lsLinkProc.do?efYd=20140724&joNo=064000&lsClsCd=L&lsId=prec20140724&lsNm=%EB%AF%BC%EB%B2%95&mode=11
8. 조문정보 | 국가법령정보센터, 1월 13, 2026에 액세스, http://www.law.go.kr/lsLinkProc.do?lsClsCd=L&lsNm=%EB%AF%BC%EB%B2%95&lsId=prec20161118&joNo=064000&efYd=20161118&mode=11&lnkJoNo=undefined
9. 조문정보 | 국가법령정보센터, 1월 13, 2026에 액세스, http://law.go.kr/LSW/lsLawLinkInfo.do?lsJoLnkSeq=900158553&chrClsCd=010202
10. 법령 - 민법 제390조 - 로앤비, 1월 13, 2026에 액세스, https://www.lawnb.com/Info/ContentView?sid=L000EF207208ECE3_390

11. Build a basic LLM chat app - Streamlit Docs, 1월 13, 2026에 액세스, https://docs.streamlit.io/develop/tutorials/chat-and-llm-apps/build-conversational-apps
12. RAG Chatbot With HuggingFace And Streamlit: Complete Tutorial - Codecademy, 1월 13, 2026에 액세스, https://www.codecademy.com/article/aichatbot-using-huggingface-rag-streamlit
13. Build and Deploy a Chatbot App Using Streamlit and OpenAI | Dataquest Project Lab, 1월 13, 2026에 액세스, https://www.youtube.com/watch?v=palgG8e8LHk
14. Technical Analysis of Modern Non-LLM OCR Engines - IntuitionLabs, 1월 13, 2026에 액세스, https://intuitionlabs.ai/articles/non-llm-ocr-technologies
15. Comparison of Paddle OCR, EasyOCR, KerasOCR, and Tesseract OCR - Plugger - AI, 1월 13, 2026에 액세스, https://www.plugger.ai/blog/comparison-of-paddle-ocr-easyocr-kerasocr-and-tesseract-ocr
16. When to Choose Local LLMs vs APIs: A Founder's Real-World Guide, 1월 13, 2026에 액세스, https://thebootstrappedfounder.com/when-to-choose-local-llms-vs-apis-a-founders-real-world-guide/
17. SOLAR 10.7B Instruct V1.0 · Models - Dataloop, 1월 13, 2026에 액세스, https://dataloop.ai/library/model/upstage_solar-107b-instruct-v10/
18. RAG Project: Build an AI Onboarding Chatbot with Streamlit, LangChain, and ChromaDB, 1월 13, 2026에 액세스, https://www.youtube.com/watch?v=WUUujm1MRQg
19. Streamlit RAG Application - Read the Docs, 1월 13, 2026에 액세스, https://streamlit-rag-app.readthedocs.io/en/latest/
20. 0xZee/RAG_Apps: Streamlit ChatBot App powered by RAG - GitHub, 1월 13, 2026에 액세스, https://github.com/0xZee/RAG_Apps
21. Building A Simple RAG Application — A Step-by-Step Approach - Medium, 1월 13, 2026에 액세스, https://medium.com/@genuine.opinion/building-a-simple-rag-application-a-step-by-step-approach-a9b77fce04f9
22. Build an Enterprise RAG Pipeline Blueprint - NVIDIA NIM APIs, 1월 13, 2026에 액세스, https://build.nvidia.com/nvidia/build-an-enterprise-rag-pipeline
23. HKUDS/RAG-Anything: "RAG-Anything: All-in-One RAG Framework" - GitHub, 1월 13, 2026에 액세스, https://github.com/HKUDS/RAG-Anything
24. Upstage SOLAR 10.7B v1.0 claims to beat Mixtral 8X7B and models up to 30B parameters., 1월 13, 2026에 액세스, https://www.reddit.com/r/LocalLLaMA/comments/18hga4p/upstage_solar_107b_v10_claims_to_beat_mixtral/
25. TheBloke/SOLAR-10.7B-Instruct-v1.0-GGUF - Hugging Face, 1월 13, 2026에 액세스, https://huggingface.co/TheBloke/SOLAR-10.7B-Instruct-v1.0-GGUF
26. upstage/SOLAR-10.7B-Instruct-v1.0 - Hugging Face, 1월 13, 2026에 액세스, https://huggingface.co/upstage/SOLAR-10.7B-Instruct-v1.0
27. Day 5: OpenAI API vs Local LLMs (Ollama) | Chat Completions Explained - YouTube, 1월 13, 2026에 액세스, https://www.youtube.com/watch?v=7gTn2TzJVH8

28. OpenAI api vs own LLM : r/LocalLLM - Reddit, 1월 13, 2026에 액세스, https://www.reddit.com/r/LocalLLM/comments/1bo4rx0/openai_api_vs_own_llm/
29. Fine tuning training cost 10,000 PDFs : r/OpenAI - Reddit, 1월 13, 2026에 액세스, https://www.reddit.com/r/OpenAI/comments/1iid161/fine_tuning_training_cost_10000_pdfs/
30. 7 Persona Examples for Modeling Your Ideal Customer - HubSpot, 1월 13, 2026에 액세스, https://www.hubspot.com/make-my-persona/persona-examples
31. How To Create Personas in Real Estate (+ A Downloadable Guide) - PropStream, 1월 13, 2026에 액세스, https://www.propstream.com/news/how-to-create-personas-in-your-real-estate-business-a-downloadable-guide
32. st.warning - Streamlit Docs, 1월 13, 2026에 액세스, https://docs.streamlit.io/develop/api-reference/status/st.warning
33. Legal Disclaimer - Personal Data, Language Support, and Verifying Responses, 1월 13, 2026에 액세스, https://experienceleague.adobe.com/en/docs/experience-platform/ai-assistant/legal-disclaimer
34. Disclaimer Examples | 8+ Disclaimer Statements - Termly, 1월 13, 2026에 액세스, https://termly.io/resources/articles/disclaimer-examples/
35. Evaluation Dataset - Ragas, 1월 13, 2026에 액세스, https://docs.ragas.io/en/latest/concepts/components/eval_dataset/
36. RAGAS: How to Evaluate a RAG Application Like a Pro for Beginners - YouTube, 1월 13, 2026에 액세스, https://www.youtube.com/watch?v=5fp6e5nhJRk
37. Automatic prompt Adaptation - Ragas, 1월 13, 2026에 액세스, https://docs.ragas.io/en/v0.1.21/concepts/prompt_adaptation.html
38. Adapting Metrics to Target Language - Ragas, 1월 13, 2026에 액세스, https://docs.ragas.io/en/latest/howtos/customizations/metrics/metrics_language_adaptation/