RAG가 해결하는 문제
언어모델(LLM)의 한계: 파라미터 안에 든 지식은 최신성이 떨어지고, 출처를 제시하기 어렵고, 길게 기억하지도 못해요.
RAG의 아이디어: “답을 만들기 전에 외부 지식베이스에서 관련 문서를 찾아 컨텍스트로 넣어준다.”
→ 최신성, 출처 제시, 도메인 특화 정확도가 크게 좋아지고, 파라미터 재학습 없이 지식만 교체·증분 추가 가능.
큰 그림(아키텍처 한눈에)
지식 수집/정제 → PDF/웹/DB 등에서 문서 수집, 전처리(클린·분할·메타데이터)
인덱싱 → 임베딩 생성(문서→벡터), 벡터DB/색인(BM25 등) 저장
질의 처리 → 사용자의 질문을 임베딩화, 검색(k, 필터), 재순위화
컨텍스트 구성 → 길이 제한 안에서 중요한 조각을 압축·선택
생성 → LLM 프롬프트에 컨텍스트 + 지시문을 넣어 답변 생성(인용/출처 포함)
평가/로그/옵스 → 품질(정확성/충실성), 비용/지연 시간, 관측·모니터링, 캐싱·업데이트
1. 데이터 레이어(수집·정제·분할)
수집(Sources)
문서: PDF, DOCX, Notion, Confluence, 이메일, 코드 리포지토리
구조화 데이터: SQL/CSV/스프레드시트
반구조화: HTML, Markdown, 로그
멀티모달: 이미지/표/도식(나중에 멀티모달 RAG에서 사용)
정제(Cleaning)
텍스트 추출 시 레거시 인코딩/하이픈/머리말·꼬리말 제거
중복 문서/버전 정리(최신만 남기거나 버전 필드 메타데이터화)
보안·개인정보 마스킹(PII/비식별화)
분할(Chunking)
왜 분할? 검색과 컨텍스트 효율성, 정확한 재현을 위해.
토큰 기준 분할: 예) 300~600 token, 10~20% 오버랩(문맥 연결)
의미 기반 분할: 제목/소제목/문단 경계, 문서 구조(Heading, 리스트, 표) 반영
표/코드는 통째로 한 덩어리로 취급(깨지면 의미 상실)
메타데이터(문서ID, 제목, 섹션, 날짜, 권한, 태그) 꼭 저장 → 필터형 검색에 필수
2. 인덱싱(Embeddings & Search)
임베딩(Embeddings)
목적: 텍스트 의미를 벡터로 매핑하여 “의미 유사성” 기반 검색 가능하게.
언어: 한국어를 많이 쓴다면 다국어 임베딩 모델을 기본값으로.
인퍼런스 세부팁
문서 임베딩과 질의 임베딩이 같은 공간(호환 모델)인지 확인
정규화(L2 normalize)로 코사인 유사도 일관성 확보
배치 임베딩 + 캐시(중복 텍스트/중복 크롤링 방지)
색인(Indexes)
벡터 검색(ANN): HNSW, IVF, PQ 등 근사 최근접(메모리 ↔ 정확도 ↔ 지연시간 트레이드오프)

키워드 검색(BM25): 정확한 키워드 매칭·희귀어·고유명사에 강함
하이브리드 검색: λ·BM25 + (1-λ)·벡터 가중 합산 or 단계 결합(먼저 벡터, 다음 키워드 필터 등)
→ 초기 버전부터 강추: 의미+키워드 장점을 동시에.
재순위화(Re-ranking)
첫 단계에서 k=50200 회수 → Cross-Encoder로 상위 520개 정밀 재정렬
효과: 섬세한 상호작용(질의-문서 문장 단위 매칭)로 정확성↑
비용: 런타임 비용/지연 증가 → 오프라인 튜닝으로 k와 cutoff 최적화
3. 질의 엔진(쿼리 가공·확장·분해)
쿼리 재작성(Query Rewriting): 사용자의 짧은/모호한 질문을 더 검색 친화적으로 확장
동의어·약어 확장, 한국어/영어 교차 확장, 오탈자 보정
질의확장(PRF/HyDE 등): 예시 답변/키워드로 리콜(Recall) 증가
셀프쿼리(Self-Query): LLM이 메타데이터 필터(기간/저자/부서)를 스스로 추출
멀티홉 분해(Decomposition): "①A를 구하고 ②A로 B를 구해 ③최종 답" 같이 단계 쪼개 검색
비즈니스 필터: 접근 권한/조직/테넌트 분리(Zero-Trust 관점)
4. 컨텍스트 구성(압축·선정·길이 관리)
LLM 컨텍스트 창은 유한 → Top-k + 다양성(MMR) + 중복 제거
컨텍스트 압축: 문서가 길면 요약/키 포인트 추출("query-focused summarization")
세밀한 스루풋 팁
긴 문서는 "문단 요약 인덱스"를 별도로 만들어 1차 압축
동일 문서의 인접 청크는 붙여 넣기(오버랩 중복 제거)
출처 표기용으로 원문 스팬(문서ID·페이지·행번호) 보관
5. 생성 단계(프롬프트·템플릿·출처)
프롬프트 패턴
Stuff: 그대로 다 넣기(간단/작은 컨텍스트에 적합)
Map-Reduce: 각 청크 요약(Map) → 통합(Reduce)
Refine: 초기 초안 → 새 증거로 점진 보강
증거지향 지시문 예시
"아래 컨텍스트에서만 답하세요. 확실치 않으면 '모름'이라 말하고, 사용한 문서를 [각주]로 표기하세요."
"핵심 bullet → 근거 인용 → 한줄 결론" 형식 고정
인용/출처
각 문장/단락 끝에 [문서명 §섹션] 식 표기
사용자가 클릭해 원문을 확인할 수 있게 ID/URL 매핑 유지
안전장치
Hallucination 방지: 컨텍스트 밖이면 "모름/재질의 유도"
정확성 우선 모드: 온도 낮춤, 디코딩 보수적, 금지어(추측 표현) 필터
6. 평가(Eval) - 오프라인·온라인
오프라인
검색 품질: Recall@k, Precision@k, nDCG, MRR

→ 라벨링(정답 문서 집합) 만들어 두면 튜닝이 쉬움
생성 품질:
충실성(Faithfulness): 출처에 근거했는가?
정확성/일치(Answer Correctness): 정답과 의미론적 일치 여부
유용성(Helpfulness): 구조·명료성·행동가능성
골든세트 구축 팁
“질문-정답-근거 문서 스팬” 50~200쌍부터 시작
주기적 리그레션 테스트(인덱스 업데이트 시 품질 흔들림 감시)
온라인
A/B 테스트: 클릭률, 재질의율, 대화 종료까지 걸린 턴 수
피드백 버튼: “맞음/틀림/애매” + 자유 텍스트 + 출처 유효성 체크
7. 운영(옵스) – 지연·비용·캐싱·관측
지연 시간 슬라이스: 임베딩(질의), 검색(ANN), 재순위화, LLM 디코딩
캐싱
쿼리→결과 컨텍스트 캐시(유사 쿼리 해시)
임베딩 캐시(문서 업데이트 빈도 낮을수록 효과↑)
완성 캐시(자주 묻는 질문)
동시성/배치: 검색 병렬화, 스트리밍 답변
관측: 질문 분포, 실패 패턴, 상위 무효 출처, 슬로우 쿼리 추적
인덱스 수명주기: 증분 업데이트(append-only), 재인덱싱 창구(야간 배치), 롤백 포인트
8. 보안·권한·컴플라이언스
문서 권한을 검색 시점에 반영: per-tenant index, row-level security
감사 추적: 누가 언제 어떤 문서를 근거로 보았는지
개인정보/민감정보: 마스킹·K-익명화, 안전 필터, 보존 주기
온프레미스/가둬진 VPC: 데이터 유출 경로 차단
9. 고급 RAG 주제
하이브리드·다단계 검색
Stage 1: 고리콜(벡터+k 큰 값)
Stage 2: 메타데이터 필터 + 키워드 재검색
Stage 3: 크로스엔코더 재랭크 → 컨텍스트 압축
멀티홉·그래프 RAG
문서 간 엔티티·관계(인물-기관-날짜)를 그래프로 엮어 연결 추론
“A의 임기 → B 정책 변화 → C 지표 영향” 같은 연결형 질문에 강함
멀티모달 RAG
OCR/표/도식 구조 인식 → 텍스트와 함께 색인
이미지 캡션·도표 데이터 추출 → “이 표를 설명해줘 + 출처” 가능
에이전틱 RAG(도구 사용)
LLM이 검색/코드실행/웹조회/API콜을 플래닝하고 결과를 재귀적으로 흡수
계산/조회가 필요한 답을 더 정확하게(예: 환율, 일정, SQL 집계)
구조화 출력
JSON 스키마/함수 호출로 폼 채우기 (FAQ 추출, 정책 비교표, 일정표 생성 등)

메모리 RAG(대화 장기 컨텍스트)
세션 메모리(요약) + 개인 노트(보안 영역) + 문서 RAG 결합
10. 실패 모드 & 디버깅 체크리스트
증상 → 원인 → 처방 빠르게 매칭하기
관련 문서 못 찾음 → 분할 과도/메타데이터 누락/임베딩 불일치
분할 크기·오버랩 조정, 제목/섹션 보존, 다국어 임베딩 확인
키워드 질의 약함 → 하이브리드 검색/PRF/동의어 사전
틀린 문서가 상위 랭크 → 재랭커 도입, BM25 비중 상향, k 조정
답은 맞는데 출처 없음 → 컨텍스트 압축 시 인용 텍스트 유지
환각(추측) → 지시문 강화, 온도↓, 컨텍스트 밖 답변 금지, “모름” 허용
최신성 부족 → 증분 크롤링·리인덱싱 주기·TTL/버전 정책
지연 높음 → k↓, ANN 파라미터 조정, 재랭커 배치 최소화, 스트리밍
11. 하이퍼파라미터 가이드(초기값 예시)
Chunk size: 300~600 tokens, overlap 10~20%
Top-k(초회수): 50200 → 재랭커 후 Top-m 520
하이브리드 λ: 0.3~0.5에서 스윕
MMR 다이버시티 람다: 0.3~0.7
스코어 컷오프: 임계값 설정해 “무리한 답변 금지”
(도메인/모델/문서 길이에 따라 꼭 오프라인 튜닝으로 확정하세요)
12. 프롬프트 템플릿 예시(한국어)
[시스템]
너는 신중하고 근거지향적인 도우미야.
- 제공된 컨텍스트에서만 답해.
- 근거가 없으면 "자료가 충분하지 않습니다"라고 말해.
- 사용한 근거에는 [문서명 §섹션]을 붙여.
[사용자 질문]
{question}
[검색 컨텍스트(인용 가능)]
{contexts}
[요구 형식]
- 핵심 요약(3줄)
- 상세 설명(필요시 단계별)
- 참고: [문서명 §섹션], [문서명 §페이지]
13. 파이프라인 의사코드(라이브러리 불문, 개념형)

```
def build_index(docs):
    cleaned = clean(docs)
    chunks = chunk(cleaned, size=500, overlap=0.15, respect_structure=True)
    embeddings = embed(chunks)  # multilingual-safe
    vector_db.upsert(chunks, embeddings)  # with metadata
def retrieve(query, filters=None):
    q = rewrite_query(query)
```

```
    q_emb = embed([q])
    cand = vector_db.search(q_emb, top_k=120, filters=filters)  # hybrid optional
    reranked = cross_encoder_rerank(q, cand, top_m=15)
    return compress_for_context(q, reranked, budget_tokens=4000)
def generate_answer(question):
    contexts = retrieve(question, filters=permissions_for(user))
    prompt = render_prompt(question, contexts)
    return llm(prompt, temperature=0.2, max_tokens=800)
# usage
# build_index(docs)  # offline/batch
# answer = generate_answer("정책 변경 전후 차이를 비교해줘")
```

14. 적용 분야(실전 시나리오)

엔터프라이즈 검색/사내 위키 비서: 조직/권한별 필터 + 최신 문서 우선

고객지원 코파일럿: 티켓·FAQ·매뉴얼 검색 → 정확한 단계별 답변 + 인용

개발자 도우미: 코드베이스/ADR/릴리즈노트 RAG → PR 리뷰 요약·레거시 API 맵

교육/연구 요약: 논문·교재·강의노트에서 근거 추출, 비교표 자동 작성

법/정책/규정 조회: 버전·시점 필터("2023년 5월 기준") + 조항 인용

데이터 분석 메모: 쿼리 템플릿·대시보드 설명서에서 즉시 조언

멀티모달: 기술 사양 PDF의 도표/회로도까지 읽고 근거 제시

의료·금융처럼 고위험 도메인은 항상 인간 검토("Human-in-the-loop")를 필수로!

15. RAG vs 파인튜닝 vs 하이브리드

RAG: 최신성/출처/빠른 업데이트에 최적.

파인튜닝(LoRA 등): 스타일·포맷·작업 절차를 모델 내부에 학습시키는 데 유리.

하이브리드: RAG로 사실, FT로 표현·절차를 보강 → 가장 견고.

16. 비용·성능 최적화 치트시트

k 줄이기(리콜 떨어지면 재랭커 도입으로 상쇄)

ANN 파라미터(efSearch, nprobe 등) 조절

재랭커 배치를 트래픽 상위 쿼리에만 적용

컨텍스트 압축으로 토큰 절감

캐시 레이어 3종(임베딩/검색결과/완성)

증분 인덱싱 + 야간 대규모 재인덱싱

자주 쓰는 문서는 요약 인덱스 병행(두 단계 검색)

17. 품질 높이는 실전 팁 10

하이브리드 검색부터 시작

분할 시 제목/표/리스트 구조 보존

컨텍스트에 원문 인용 스팬 포함

"모름"을 허용하는 지시문

메타데이터 필터(날짜/부서/문서종류) 적극 활용

재랭커로 상위 5~15개 정밀 선별

출시 전 골든세트 최소 수십 쌍 라벨

질문 분해로 멀티홉 해결

업데이트 파이프라인 자동화(증분/버전/롤백)
관측 대시보드로 품질·지연·비용 상시 추적
18. “가상 PDF로 RAG 테스트” 빠른 실험 설계
데이터: 여러분이 만든 테스트 PDF 3~5개(주제·난이도 다르게)
라벨링: 문서별로 5~10개 질문과 “근거 문장 스팬” 작성
평가 루프
하이브리드 검색 켜고 Recall@50 체크
재랭커 추가 후 nDCG/정확성 비교
프롬프트에 인용 강제 → 충실성 점검
성공 기준:
회수 단계 Recall@50 ≥ 0.9
최종 답 충실성 ≥ 0.85, 재질의율 ≤ 10%
19. 자주 묻는 질문(FAQ)
Q. PDF 표/이미지가 많아요.
A. OCR+테이블 인식으로 텍스트화 후, 표는 CSV·키-값 요약도 함께 색인. 표 전체를 하나의 청크로 유지하세요.
Q. 한국어/영어 섞인 문서인데 검색이 약해요.
A. 다국어 임베딩 + 키워드 하이브리드. 쿼리 재작성 때 영문 키워드 동의어를 자동 추가.
Q. 최신 문서만 보고 싶어요.
A. 메타데이터 date>=… 필터, 그리고 재랭킹에서 “recency prior”를 가중치로 반영.
Q. 답이 길어 퍼포먼스가 떨어져요.
A. 컨텍스트를 줄이거나 요약 인덱스를 먼저 검색 → 필요한 섹션만 확장 로딩.
20. 체크리스트(런칭 전 최종 점검)
- [ ] 데이터 정제/분할 전략 합의(크기·오버랩·구조)
- [ ] 임베딩·색인 파이프라인 재현성/버전 관리
- [ ] 하이브리드 검색 + 재랭커 기본 탑재
- [ ] 프롬프트에 인용·“모름” 규칙 포함
- [ ] 골든세트 기반 오프라인 지표 산출
- [ ] 관측 대시보드(지연·비용·정확성) 준비
- [ ] 증분 업데이트·롤백 플랜
- [ ] 권한/보안/감사 로그 점검