

## "AI 기반 문서 관리 + 협업 플랫폼"

단순 요약 툴 → 팀 단위로 사용하는 문서 관리 시스템

### 추가 기능 제안

#### 1. 팀/프로젝트 관리

- 팀 생성 및 멤버 초대
- 프로젝트별 문서 그룹화
- 역할 기반 권한 관리 (읽기/쓰기/관리자)

백엔드 포인트: 복잡한 권한 관리, 멀티 테넌시 설계

#### 2. 문서 버전 관리 + 협업

- 같은 문서의 여러 버전 관리
- 버전 간 비교 (diff)
- 공동 편집 히스토리
- 댓글/피드백 기능

백엔드 포인트: 이벤트 소싱, 버전 관리 전략, 낙관적 락

#### 3. 스마트 검색 + 추천

- 전체 문서 검색 (Elasticsearch)
- 태그 자동 생성 (AI)
- 관련 문서 추천
- 문서 간 연관성 분석

백엔드 포인트: 검색 엔진 연동, 추천 알고리즘, 캐싱 전략

#### 4. 대시보드 + 분석

- 팀별 문서 통계
- 카테고리별 분포
- 사용자 활동 로그
- 문서 트렌드 분석

백엔드 포인트: 배치 처리, 집계 쿼리 최적화, 통계 데이터 설계

#### 5. 알림 시스템

- 새 문서 업로드 알림
- 멘션 알림
- 요약 완료 알림 (비동기)

백엔드 포인트: 이벤트 기반 아키텍처, 알림 큐 관리

#### 6. 외부 연동

- Google Drive 연동
- Slack 알림 연동
- Webhook으로 다른 시스템 연동

백엔드 포인트: OAuth 구현, 외부 API 연동, 에러 핸들링

### 기술 스택 제안

Backend:

- Spring Boot 3.x

- Spring Security (JWT 인증)
- JPA + QueryDSL (복잡한 쿼리)
- Redis (캐싱, Rate Limit, 세션)
- RabbitMQ or Kafka (AI 처리 비동기화)
- Elasticsearch (검색)
- Spring Batch (통계 집계)

AI:

- OpenAI API or Claude API

DB:

- MariaDB (메인)
- Redis (캐시/세션)
- Elasticsearch (검색)

Infra:

- Docker + Docker Compose
- GitHub Actions (CI/CD)
- AWS S3 (파일 저장)

면접에서 어필할 수 있는 포인트

마이크로서비스 지향 설계 (모놀리식이지만 모듈화)

이벤트 기반 아키텍처 (비동기 처리)

성능 최적화 (쿼리 튜닝, 캐싱 전략)

확장 가능한 구조 (멀티 테넌시, 플러그인 구조)

테스트 코드 (단위/통합/E2E)

API 문서화 (Swagger/Spring REST Docs)

결론

원래 아이디어를 기본 MVP로 시작하되, 위 기능들을 단계적으로 추가하는 걸 추천해요.

Phase 1 (2주): 기본 요약 기능 + 비동기 처리

Phase 2 (2주): 팀 관리 + 권한

Phase 3 (2주): 검색 + 버전 관리

Phase 4 (1주): 대시보드 + 통계

1단계: 프로젝트 기본 세팅

Spring Boot 4.0.1 프로젝트 생성

Docker로 MariaDB 실행

application.yml 설정

기본 패키지 구조 설계

2단계: 인증/회원 기능

User 테이블 설계

Spring Security 설정

JWT 발급/검증

회원가입 API

로그인/로그아웃 API

3단계: 핵심 도메인 설계

Document(문서) 테이블

Team(팀) 테이블

TeamMember(팀 멤버) 테이블

Category(카테고리) 테이블

테이블 간 관계(FK) 설정

현실적인 ERD (간단 버전)

User (사용자)

- id
- email
- password
- username
- created\_at

Team (팀)

- id
- name
- owner\_id (FK: User)
- created\_at

TeamMember (팀 멤버)

- id
- team\_id (FK: Team)
- user\_id (FK: User)
- role (OWNER/MEMBER/VIEWER)
- joined\_at

Document (문서)

- id
- title
- original\_filename
- file\_path
- file\_size
- upload\_user\_id (FK: User)
- team\_id (FK: Team)
- category (MEETING/PLAN/ETC)
- status (PROCESSING/COMPLETED/FAILED)
- created\_at
- updated\_at

DocumentSummary (요약 결과)

- id
- document\_id (FK: Document)
- summary\_text

└── ai\_model\_version  
└── created\_at  
└── updated\_at