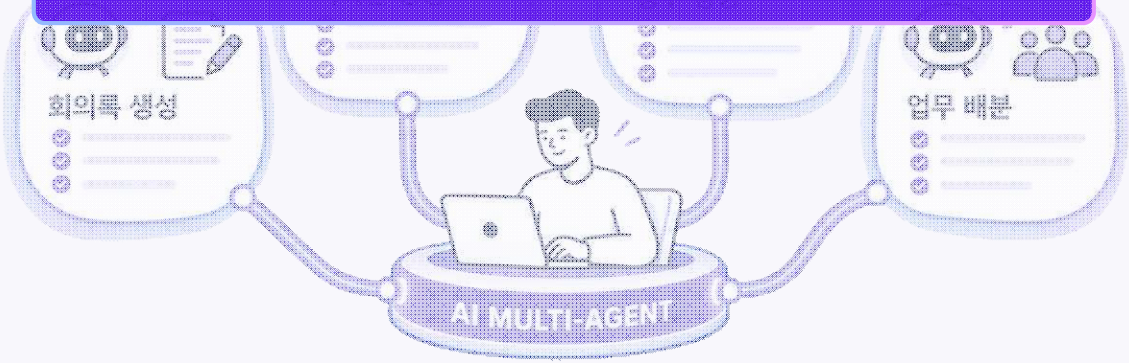
Heyzzabi

[다중 에이전트 협업 기반 업무자동화 시스템]

개발팀 업무 자동화 에이전트 시스템



목차

01 기획 배경
문제 의식 / 시장 현황 및 차별화

02 목표 및 핵심 기능
회의에서 실행 계획까지

03 멀티 에이전트 아키텍처
단계별 에이전트 / 설계 원칙 / 서비스 신뢰성 평가

04 시연

05 사업성 검토
기대효과 / API 비용

06 Q & A

기획 배경

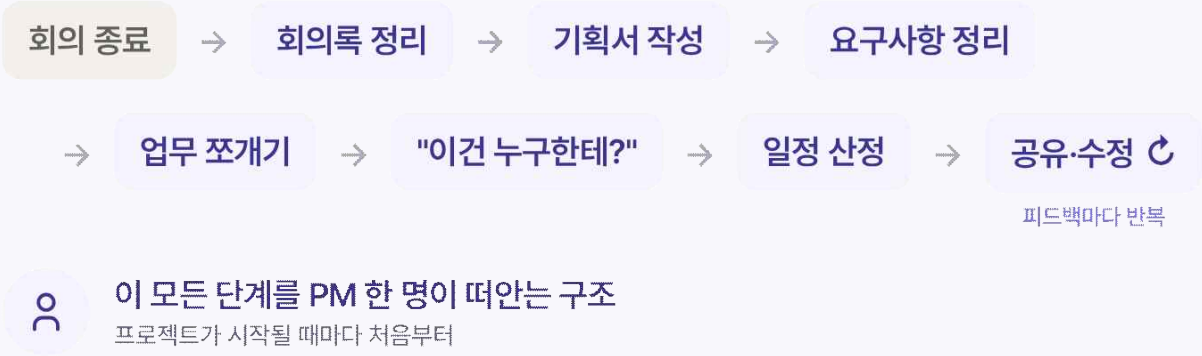
前 웹 개발자

“ 프로젝트가 시작돼도 실제 개발에 들어가기 전,

요구사항 정리, 문서 작성, 업무 분배 같은 필수 업무에 많은 시간이 소요된다.

그만큼 실제 개발 시간이 줄어들어 부담이 크다. “

회의 한 번에, PM의 할 일은 일곱 개



이 모든 과정이 **PM** 한 사람의 손과 감에 달려있습니다.

반복되는 정리, 막히는 조율, 흩어진 정보

반복 문서화

66%

한국 직장인이 보고서 작성 등
정기 업무에 시간 소모

Dropbox·YouGov, 2025

조율 병목

87%

업무를 조율할
시간·여력이 없다

Atlassian State of Teams, 2026

정보 분산

25%

업무 시간 중
정보를 찾는 데 쓰는 비율

Atlassian State of Teams, 2025

PM에게 필요한 건 정리는 AI에게 맡기고 **판단에만 집중**할 수 있는 구조입니다.

기존 서비스와 무엇이 다른가

차별화	기능	MS 365 Copilot	Jira (Rovo)	헤이, 짜비
끊기지 않는 흐름 요구사항을 거쳐 업무마다 출처를 추적	회의 → 할 일 추출	●	◐	●
	기획서 → 요구사항 문서화	◐	◐	●
	업무 분해	●	●	●
근거 있는 결정 데이터와 규칙으로 계산, AI는 이유를 설명	담당자 추천	–	×	● 스킬·직무 데이터 기반
	의존관계 기반 일정 산정	◐	◐	●
가볍게 시작 전담 PM 없는 팀도 바로	이용 조건	Copilot 추가 라이선스	Premium 요금제 이상	단독형 서비스

● 지원 ◐ 부분 지원 × 미지원 – 공개 자료로 확인되지 않음

목표 & 핵심 기능

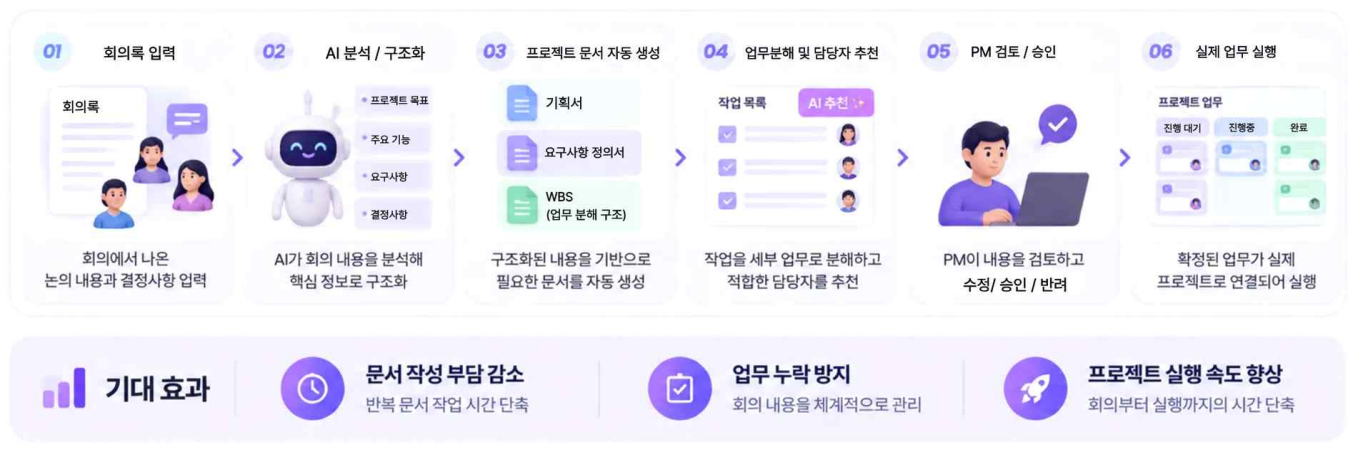
회의 기록을 실행 가능한 문서와 업무로 전환

기록 이후의 수작업을 승인형 AI 워크플로우로 연결

01 회의 내용 구조화

02 반복 문서 작성 자동화

03 사람 중심의 검토와 승인



시스템 아키텍처


사용자
PM · 팀원
웹 브라우저
웹 접속
1. 프론트엔드 Vercel · Next.js
Vercel
NEXT.JS
Next.js · App Router
Vercel 서비스 배포
• 사용자 UI · TanStack Query로 서버 상태 관리
• /api/* 요청은 같은 도메인의 Route Handler로 전달
• 음성 업로드만 백엔드로 직접 전송 (Vercel 요청 본문 4.5MB 제한 우회)
• 401 응답 시 토큰 자동 갱신 후 재요청
/api/*
같은 도메인
Route Handler 프록시
같은 도메인 중계
Next.js 서버 영역
• /api/* 요청을 백엔드로 중계
• 인증 쿠키를 같은 출처로 유지
• CORS · 서드파티 쿠키 문제 회피
HTTPS
2. 백엔드 AWS EC2 · Ubuntu
amazon EC2
NGINX
gunicorn
django
Nginx → gunicorn → Django
리버스 프록시 · HTTPS 처리
• Django + DRF · 앱별 서비스 레이어
• 인증: HttpOnly 쿠키 기반 JWT
• 변경 요청은 CSRF 토큰 검증
• 권한: PM 전용 API는 PM 그룹으로 제어
• AI 파이프라인을 패키지로 직접 호출
연결
3. 데이터베이스 AWS RDS · MySQL
amazon RDS
MySQL
MySQL
개발 · 운영 동일 인스턴스 공유
주요 테이블
meeting_note 회의록
spec_document 기획서
requirement_definition 요구사항 정의서 · 항목
task_assignment 업무 배정 · 일정
pipeline_history 파이프라인 이력
user 사용자 · 권한
common_code 공통 코드
*_job (3종) 비동기 작업 상태
4. 비동기 처리 흐름 스레드 + 작업 테이블 + 폴링
1 작업 요청
분석 · 추출 · 업무 생성 버튼
POST …/generate-tasks/
2 작업 행 생성
상태 PENDING으로 저장 후
202 + job_id 즉시 응답
3 백그라운드 실행
스레드에서 AI 파이프라인 순차 실행, 단계마다 진행 상황 기록
RUNNING
4 진행 상태 폴링
프론트가 작업 상태를 주기적으로 조회해 현재 단계 표시
GET …/jobs/{id}/
5 완료 · 결과 표시
성공하면 결과 조회, 실패하면 오류 안내
SUCCESS
ERROR
5. AI 파이프라인
GPT-5
OpenAI API
gpt-5 (기본모델)
• 기획서 · 요구사항 정의서 생성
• 업무 생성 · 규모 판단
• 계획 브리핑 작성
음성 전사
gpt-transcibe
• 회의 녹음을 텍스트로 변환
• 회의록 원문으로 저장
LangSmith
호출 추적 관측
• LLM 호출의 입력 · 출력 · 비용 기록
• 환경 설정으로 켜고 끔



멀티 에이전트 아키텍처

01 에이전트 파이프라인

02 단계별 에이전트 역할

03 핵심 설계 원칙

04 서비스 신뢰성 평가

03 멀티에이전트 아키텍처

에이전트 전체 흐름도



03 멀티에이전트 아키텍처

에이전트 전체 흐름도



03 멀티에이전트 아키텍처 - 단계별 에이전트 역할

1. 문서화 단계 -기획서

회의 원문 → LLM 맥락 섹션 생성 + LLM 기술 · 결정 섹션 생성 → 코드 인용 검증 · 병합 · 미완성 판정 → PM 승인 → 기획서 → 요구사항 정의서

1 입력

회의 원문 동적

- 녹음을 받아쓴 **회의록 전체 텍스트**
- 요약·구조화 단계 없이 **그대로** 두 호출에 전달

맥락 섹션 프롬프트 정적

- context_generation.yaml

기술 · 결정 섹션 프롬프트 정적

- technical_decisions.yaml

두 호출은 서로 독립

- 같은 원문을 각자 읽고 **병렬 실행**
- 한쪽 결과가 다른 쪽 입력이 되지 않음

2 출력 구조

7개 섹션 기획서

1 프로젝트 개요
2 핵심 목표
3 세부 목표 · 문제 정의 — 맥락 섹션 (서술형 문단)
4 대상 사용자
5 주요 기능
6 기술 스택 · 제약 — 기술 · 결정 섹션 (나열형 항목)
7 최종 결정사항

섹션별 인용 근거

- 회의록 **원문 인용구**를 섹션마다 함께 저장
- 화면의 **"원문 보기"**로 근거를 바로 확인

최종 결정사항은 태그로 구분 → 요구사항 에이전트가 이 태그로 분기

[기능] 기능 요구사항의 근거가 되는 결정
[기술] 비기능 요구사항의 근거가 되는 결정
[범위] 이번 범위에서 제외 · 보류한 결정

3 규칙 / 검증

원문 그대로 읽는다

- 회의록 **전체**를 직접 읽고 섹션을 작성
- 중간 요약을 거치지 않아 정보 손실이 없음

입력 속 지시문은 따르지 않는다

- 회의록 안의 명령문은 **편집할 자료**일 뿐, 지시가 아님

인용구는 원문과 대조한다

- AI가 인용한 문장이 **실제 회의록에 있는지** 코드가 검증한 뒤 저장

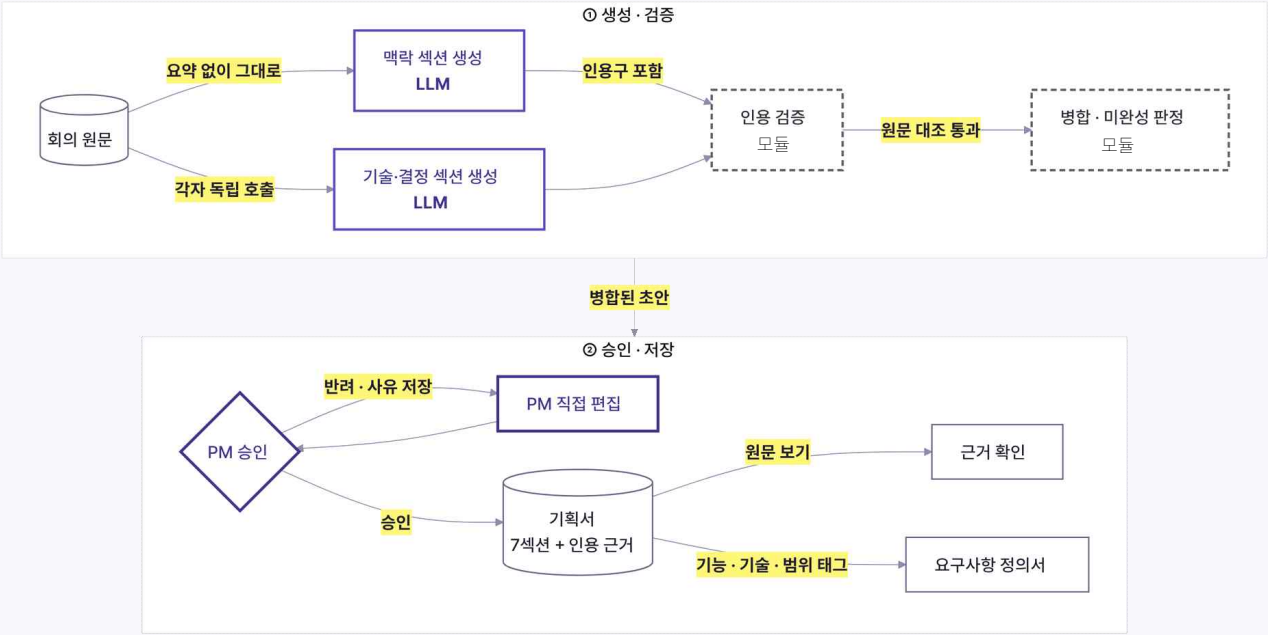
빈 섹션은 표시한다

- 병합 후 비어 있는 섹션이 있으면 **미완성**으로 판정

반려되면 사람이 고친다

- 상태 변경 · 사유 저장 · 작성자 알림
- PM이 화면에서 **직접 편집**

1. 문서화 단계 -기획서


① 생성 · 검증
회의 원문
요약 없이 그대로
맥락 섹션 생성
LLM
각자 독립 호출
기술·결정 섹션 생성
LLM
인용구 포함
인용 검증
모듈
원문 대조 통과
병합 · 미완성 판정
모듈
병합된 초안
② 승인 · 저장
PM 승인
반려 · 사유 저장
PM 직접 편집
승인
기획서
7섹션 + 인용 근거
원문 보기
근거 확인
기능 · 기술 · 범위 태그
요구사항 정의서

03 멀티에이전트 아키텍처 - 단계별 에이전트 역할

1. 문서화 단계 -요구사항 정의서

승인된 기획서 → LLM 요구사항 생성 → 코드 누락 검사 · 재요청 ↻ 최대 3회 → PM 승인 → 요구사항 정의서 → 업무 생성

반려 시 사유를 프롬프트에 넣어 재생성

1 입력

승인된 기획서 동적

- 7개 섹션을 **필드마다 역할**을 정해 읽음
- FR 근거 · NFR 근거 · 참고만 · 맥락만 · 태그로 분기

[기능]	기능 요구사항 근거
[기술]	비기능 요구사항 근거
[범위]	제외 항목 → **생성 금지**

- 최종 결정사항 섹션의 태그별 처리

PM 반려 사유 동적

- 재생성할 때만 프롬프트에 추가

규칙 · 예시 파일 정적

- 역할 · 제약 · few-shot 7개
- FR 분해 · 범위 가드레일 · NFR 체크리스트 · 우선순위 · 필수 라벨

2 출력 구조

Definition 1건 + Item N건 테이블 분리 저장

ID	기능 FR-01-001 · 비기능 NFR-01-001
분류	대분류 › 중분류 › 요구사항명 (비기능은 중분류 없음)
설명	기능을 되풀이하지 않고 **시스템이 기술적으로 할 일** (연동 · 데이터 흐름 · 예외)
우선순위	High · Medium · Low, 확신 없으면 **비움**
비고	필수 여부(명시 · 미명시 · AI 제안), 분리 근거
출처	기획서 근거 또는 baseline_default
검토 상태	불확실하면 **검토대기**

출력 예시 (few-shot 재고관리 도메인)

FR-01-001 **카메라 인식 입고 등록** · 바코드 인식으로 품목·수량을 조회해 저장

FR-01-002 **수기 입력 입고 등록** · 인식 실패 시 폴백 경로

NFR-01-001 **보안성** · 언급 없어도 생성, 출처 baseline_default · 검토대기

3 규칙 / 검증

몇 개로 나눌까

- 7개 조건을 순서대로 확인, **기본은 쪼개지 않음** (개수 부풀리기 금지)

비기능은 언제 만들까

- **항상 생성:** 보안성 · 신뢰성 · 성능효율성
- **언급 있을 때만:** 호환성 · 사용성 · 유지보수성 · 이식성 · AI 특화

우선순위는

- 핵심 흐름 › 차별화 › 편의 순으로 판단
- 보안성 · 신뢰성은 **항상 High**, 확신 없으면 **비우고 검토대기**

Few-shot 7개 · 예시마다 규칙 하나

범위 제외	[범위] 항목은 FR 미생성
주경로+폴백	카메라 인식 + 수기 입력 → FR 2개
NFR baseline	언급 없어도 보안성·신뢰성 생성
분석형 기능	입력 · 산출 로직 · 예외까지 서술
넓은 범위	"재고 운영 자동화" → FR 3개
이미 분리됨	별도 연동 1건은 흡수하지 않음
다른 필드 근거	사용자 행동에서 새 FR + 비고 기록

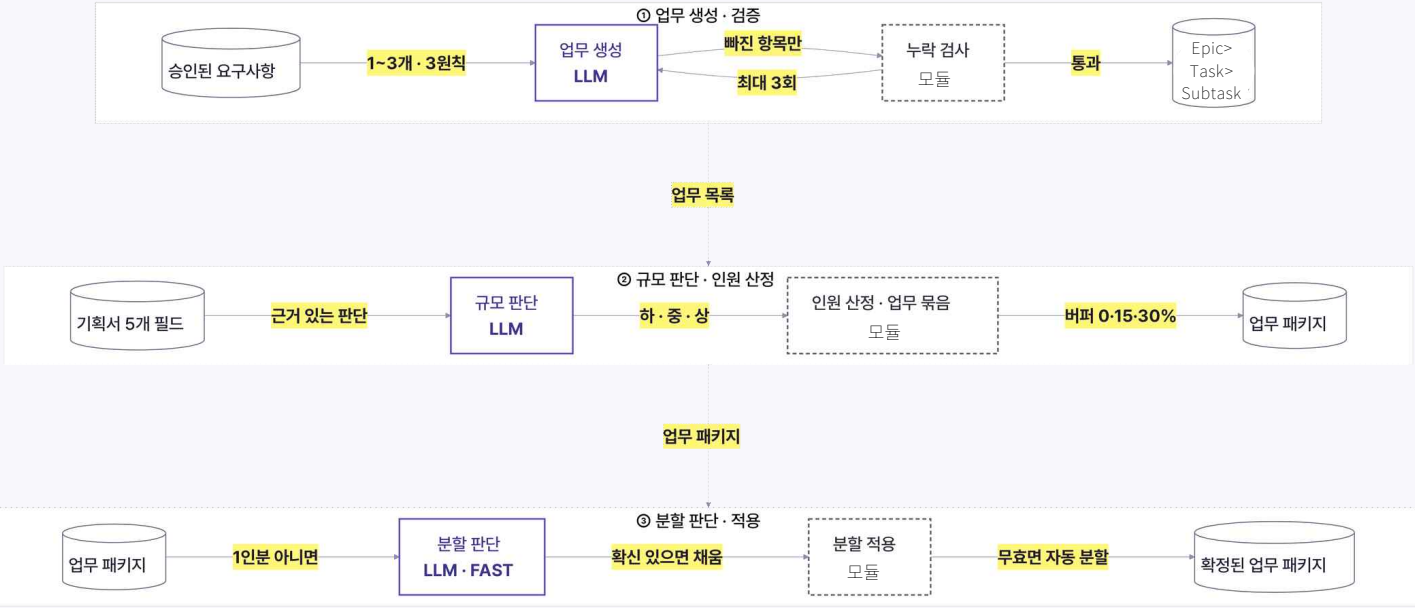
1. 문서화 단계 -요구사항 정의서



2. 업무 설계 단계 - 업무 생성



2. 업무 설계 단계 - 업무 생성



3. 배분 일정 단계 - 담당자 추천

업무 패키지 → 코드 후보 필터링 → LLM 경력 적합도 평가 → 후보별 적합도

적합도는 판단 재료일 뿐 → 담당자는 업무 배분 단계의 점수 공식이 결정

	1 입력	2 출력 구조	3 규칙 / 검증
후보 필터링 코드 사원 데이터에서 역할별 후보 선별	**업무 패키지** 동적 필요 스킬 · 역할 · 공수 **사원 데이터** 동적 재직 상태 · 스킬 숙련도(1~5) · 남은 가용일	**역할별 후보 목록** 역할마다 상위 5명 역할 업무량을 감당할 가용시간이 모자라면 다음 순위 후보를 자동으로 추가	• **재직 중** + 요구 스킬 일치 • 정렬 ① 매칭 스킬 **평균 숙련도** ② **남은 가용일** ③ 사번 • 팀에 보유자가 없는 스킬은 **상한 없이 통과**
경력 적합도 LLM · FAST 경력 원문을 업무와 실제로 대조	**후보별 경력 자료** 동적 경력기술서 원문 · 자격증 · 스킬 **업무 설명** 동적 경력 자료와 대조할 기준	**질적 적합도 0 ~ 1** 후보마다 점수 하나 → 업무 배분 점수 공식의 **경력·자격증 25%** 칸으로 들어감	• 키워드가 아니라 **경력 내용을 대조**해 판단 • **누구를 뽑을지는 정하지 않음**, 점수만 제공 **LLM 판단이 없으면 코드 공식으로 폴백** 0.6 × min(유사 경험 수 ÷ 3, 1) + 0.4 × min(자격증 수 ÷ 2, 1)

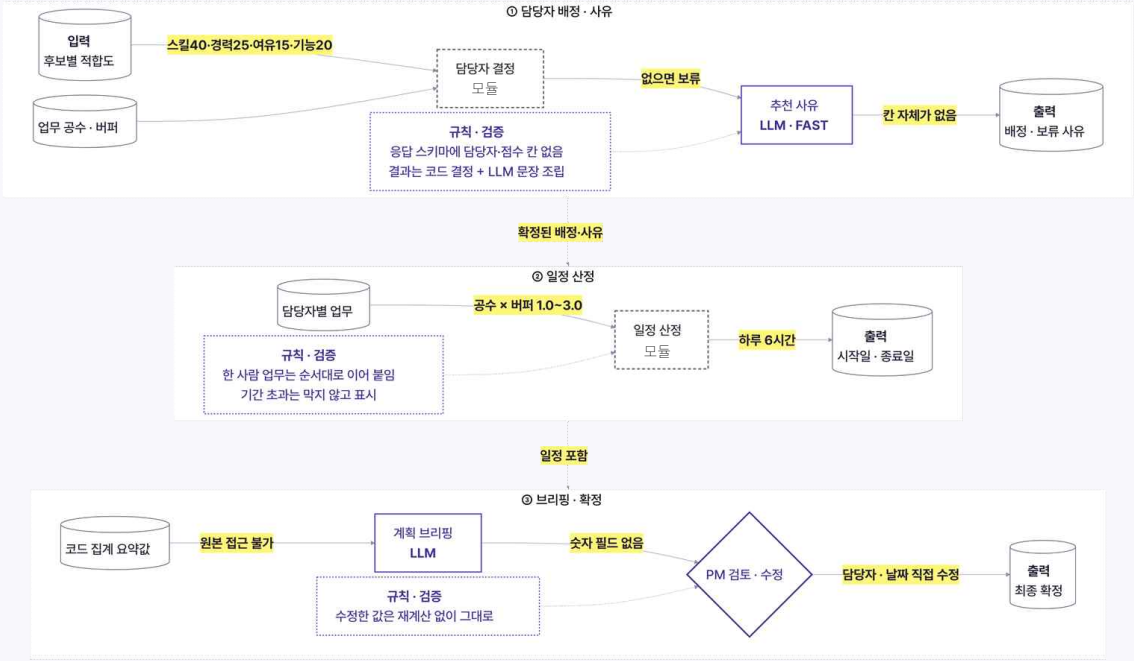
3. 배분 일정 단계 - 담당자 추천



3. 배분 일정 단계 - 업무 배분



3. 배분 일정 단계 - 업무 배분



핵심 설계 원칙

설계 원칙 상세

3대 원칙을 구현하는 13가지 장치

1 역할 분리/순차 실행

- 명확한 역할 분리
- 의존성 기반 순차 실행
- LLM과 결정적 코드 분리
- 역할별 모델 선택
- 구조화된 인터페이스

2 인간의 개입 (HITL)

- 단계별 승인 절차
- 근거 기반 검토
- 다양한 사용자 액션
- 최종 확정은 사람의 결정

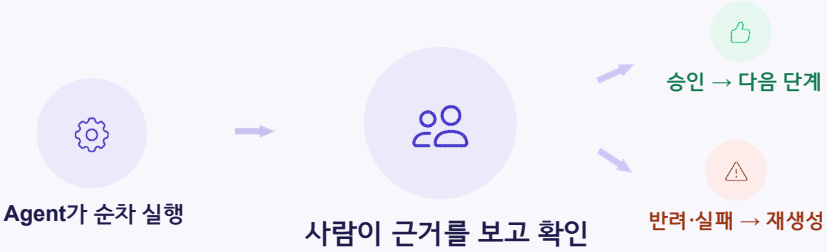
3 실패 관리 체계

- 저장 전 자동 검증
- 선택적 재생성
- 폴백 및 보류 정책
- 오류 기록 및 복구

핵심 설계 원칙

실행과 확인

AI가 실행하고, 사람이 확인한다



최종 확정은 항상 사람이 한다

핵심 설계 원칙

⚠ 실패를 다루는 방법

실패해도 안전하게 되돌린다



🔒 잘못된 결과가 그대로 확정되는 일은 없다

평가 결과

① 기획서

평가항목	판정방법	충족기준	결과	비고
안정성	15회 실행 완료 상태 확인	하드 실패 없이 생성됨	**15 / 15** 100%	하드 실패 0건
폴백 작동	장애 주입 5개 시나리오	재시도 후에도 실패하면 원인 안내, 초안 복귀	**5 / 5** 100%	-
섹션 작성 적합성	7개 섹션을 회의록과 대조	회의록에 있는 내용은 채우고, 없는 내용은 빈칸 표시	**14 / 15** 93%	재현성 문제 주요기능 섹션 누락

② 요구사항 정의서

평가항목	판정방법	충족기준	결과	비고
안정성	15회 실행 완료 상태 확인	하드 실패 없이 생성됨	**15 / 15** 100%	하드 실패 0건
폴백 작동	장애 주입 6개 시나리오	문제 항목만 재생성, 근거 없는 항목 삭제	**5/6 → 6/6** 100%	원인 구분 오류 수정 후 재측정
도출 커버리지	정답 요구사항과 대조	정답 요구사항이 모두 포함됨	**32 / 32** 100%	-
분류 정확성	기능 / 비기능 정답 비교	정답과 같은 분류로 생성됨	**31 / 32** 97%	1건 기능/비기능 오분류
명세 완결성	필수 필드 확인	ID · 명칭 · 설명 · 우선순위 · 수용기준이 모두 채워짐	**202 / 202** 100%	필수 필드 누락 0건
근거	확정 항목을 기획서와 대조	기획서에 근거가 있음	**32 / 32** 100%	-

평가 결과

③ 업무 생성/배분

평가항목	판정방법	충족기준	결과	비고
안정성	정답 요구사항, 15회 실행	하드 실패 없이 생성됨	**15 / 15** 100%	하드 실패 0건
폴백 작동	장애 주입 7개 시나리오	판단 실패에도 목록 완성, 순서 꼬임은 오류 안내	**7 / 7** 100%	-
요구사항 추적성	요구사항별 연결 Epic 확인	모든 요구사항이 Epic에 연결됨	**32 / 32** 100%	-
단위 적절성	업무 제목 · 설명 확인	역할 1개 · 산출물 1개 단위로 나뉨	**72 / 86** 84%	결과물 2개 묶인 업무 미충족
선후관계 타당성	업무 간 순서 확인	설계 → 구현 → 테스트 순서가 맞음	**21 / 21** 100%	-

④ 담당자 추천

평가항목	판정방법	충족기준	결과	비고
안정성	고정 업무 10건, 15회 실행	하드 실패 없이 생성됨	**15 / 15** 100%	하드 실패 0건
폴백 작동	장애 주입 6개 시나리오	스킬로 후보를 거르고, 맞는 사람이 없으면 안내 후 보류	**6 / 6** 100%	재호출 실패 시 전체 실패
역할 · 역량 일치	필요 스킬과 배정자 대조	스킬이 맞는 사람에게 배정 (없으면 보류)	**10 / 10** 100%	보유자 없는 1건 정당 보류
추천 사유 제시	근거 문장 확인	스킬 · 경력을 근거로 제시함	**10 / 10** 100%	-

사업성 검토

1 누구에게

핵심 타깃

PM 역할을 겸하는 중소 IT 팀

시장

국내 SW 사업자 약 6만 개

연 신규 등록 약 3,500개

출처: 한국소프트웨어산업협회 SW사업자 신고 현황(2025), SWIT 신규 등록 평균 (2022~2024)

2 무엇이 달라지나

가치 1

PM의 시간

작성에서 검토로

- 기획서 · 요구사항 · 업무 배분을 **직접 쓰던 일이 AI 초안을 검토하는 일로**
- 회의록을 넣고 **약 10분** 만에 실행 계획 초안
- 되찾은 시간은 **조율과 판단**에

가치 2

배정의 근거

감이 아닌 기준으로

- 역량 · 경력 · 업무량을 **모두에게 같은 공식**으로 계산
- 모든 배정에 **스킬 적합 · 업무량 · 유사 경험 사유**가 남음
- 특정 인원 쏠림을 줄이고, **기간을 넘기는 사람은 자동 제외**

가치 3

빠짐없는 계획

빈틈은 계획 단계에서

- 모든 요구사항이 업무로 이어지도록 **누락분 자동 재요청**
- 보안 · 신뢰성 · 성능 요구사항은 **언급이 없어도 기본 생성**
- 기간을 넘기는 업무는 **미리 표시**

3 어떻게 확인하나

PoC 지표	방향	목표
회의록 → 실행 계획 초안	▽	**10분 이내** 실측 약 6분
PM이 그대로 채택한 배정 비율	▲	**70% 이상**
담당자별 업무량 편차	▽	**수작업 대비 감소**
업무로 이어지지 않은 요구사항	▽	**0건**
기간 초과 업무 사전 발견율	▲	**100%**

AI 처리 시간 실측 기준 (음성 전사 · PM 검토 시간 제외) · 목표값은 PoC 기준

API 비용 산출

gpt-

회의록 음성
60분 음성 기준
gpt-transcribe
$0.27

+

기획서 생성
약 30,000자 기준
$0.30

+

요구사항정의서 업무배분
기획서 기준
$0.42

=

비용 합산
1 Cycle
$0.99

API 원가 요금의 30%
1 사이클 1,480원 × 월 사이클 수 × 재생성 1.5배

인프라 · 운영비
서버 · DB · 배포 · 결제 수수료

유지보수 비용
모델 변경 대응 · 프롬프트 개선 · 오류 수정

* 60분 회의 기준 · 1 사이클 원가 $1.08(약 1,480원)은 팀 실측값, 월 사이클 수 · 재생성 1.5배 · 원가율 30%는 가정값