단위 프로젝트 80~82 일차 · 노코드 멀티 에이전트 워크플로우 설계 및 운영

기술 지원 자동화 시스템, 한눈에 보기

지능형 Tech Support 워크플로우의 전체 아키텍처를 그림으로 쉽게 풀어 봅니다



입구

문의를 받는 곳



두뇌

판단하는 곳 (n8n)

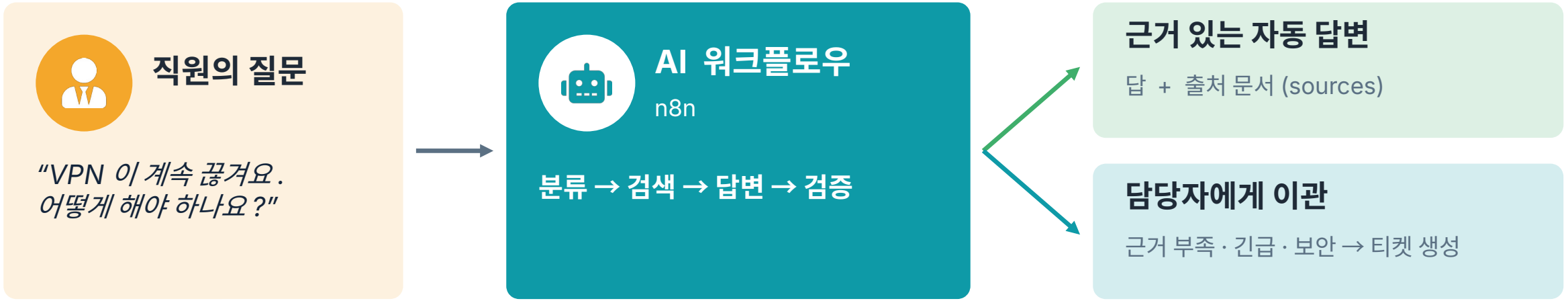


뒷단

사람에게 넘기는 곳

우리가 만드는 것 , 한 문장으로

직원이 물으면 자료에서 근거를 찾아 답하고 , 모르면 지어내지 않고 담당자에게 넘깁니다



3 일
80~82 일차 · 24 시간

4 팀
5~6 명씩 · 역할 5 가지

3 명 +
역할을 나눈 AI 에이전트

14 개
모든 팀 공통 시나리오

회사 1층 안내데스크를 떠올려 보세요

어려운 기술 이름 대신, 사람이 일하는 모습으로 먼저 이해합니다



비유

접수 창구

질문이 들어오는 입구

실제로는

웹훅 · 텔레그램 봇 · 웹 화면



비유

역할 나눈 안내 직원

분류 담당, 답변 담당, 검토 담당

실제로는

AI 에이전트 (분류 · 답변 · 검증)



비유

자료실과 사서

답하기 전에 규정집에서 근거 찾기

실제로는

RAG (지식 문서 검색)



비유

담당 부서 호출

어려운 일은 접수증을 써서 넘기기

실제로는

티켓 생성 · 알림 (헬프데스크)



이 모든 직원이 따르는 '업무 매뉴얼' = n8n 워크플로우 (순서 · 조건 · 예외를 노드와 연결선으로 그린 것)

전체 구조는 세 덩어리로 나뉩니다

입구와 뒷단은 갈아 끼우는 부품 , 가운데 두뇌는 어떤 조합에서도 그대로입니다



원래 요구를 우리 교실 버전으로 바꾸면

Slack·Jira 자리에 무엇을 끼우는지만 다를 뿐 , 하는 일은 같습니다

하는 일	원본 요구	A 기본형	B 메신저 + 헬프데스크	C 자체 웹앱
문의 받기	Slack	팀 웹훅	텔레그램 봇	직원용 웹 화면 (Next.js)
티켓 · 담당자	Jira	data-api /tickets	Zammad 등 오픈소스 헬프데스크	관리자용 웹 화면 + FastAPI
알림	Slack · Email	data-api /alerts	텔레그램 메시지	관리자 화면 알림
지식 문서	Notion	data-api /docs (규정 8 개)	A 와 같음	A 와 같음
기록 · 상태	Database	/logs · /cache · /dlq	A 와 같음	FastAPI + PostgreSQL

가운데 두뇌 (n8n) 는 세 경우 모두 같습니다 . 바뀌는 건 입구 · 뒷단을 연결하는 ‘어댑터’ 노드 몇 개뿐

꼭 알아야 할 단어 6 개

이 여섯 단어만 알면 아키텍처 그림을 모두 읽을 수 있습니다



웹훅

Webhook

워크플로우의 ‘초인종’. 정해진 주소로 요청이 오면 n8n 이 깨어나 일을 시작



API

프로그램끼리 주고받는 ‘주문서’. 정해진 모양으로 요청하면 정해진 모양으로 답



에이전트

Agent

한 가지 역할만 맡은 AI 직원. 분류 · 답변 · 검증처럼 일을 나눠 맡음



RAG

검색 증강 생성

답하기 전에 자료실에서 근거를 찾아 ‘오픈북 시험’처럼 답하는 방식



티켓

Ticket

사람에게 넘길 일을 적은 ‘업무 접수증’. 번호로 진행 상황을 추적



크리덴셜

Credential

비밀번호 · API 키를 넣어 두는 n8n 의 ‘금고’. 문서 · 코드에 키를 쓰지 않음

문의 하나가 지나가는 8 단계

규칙으로 정할 수 있는 건 규칙으로 , AI 는 꼭 필요한 곳에만 씁니다

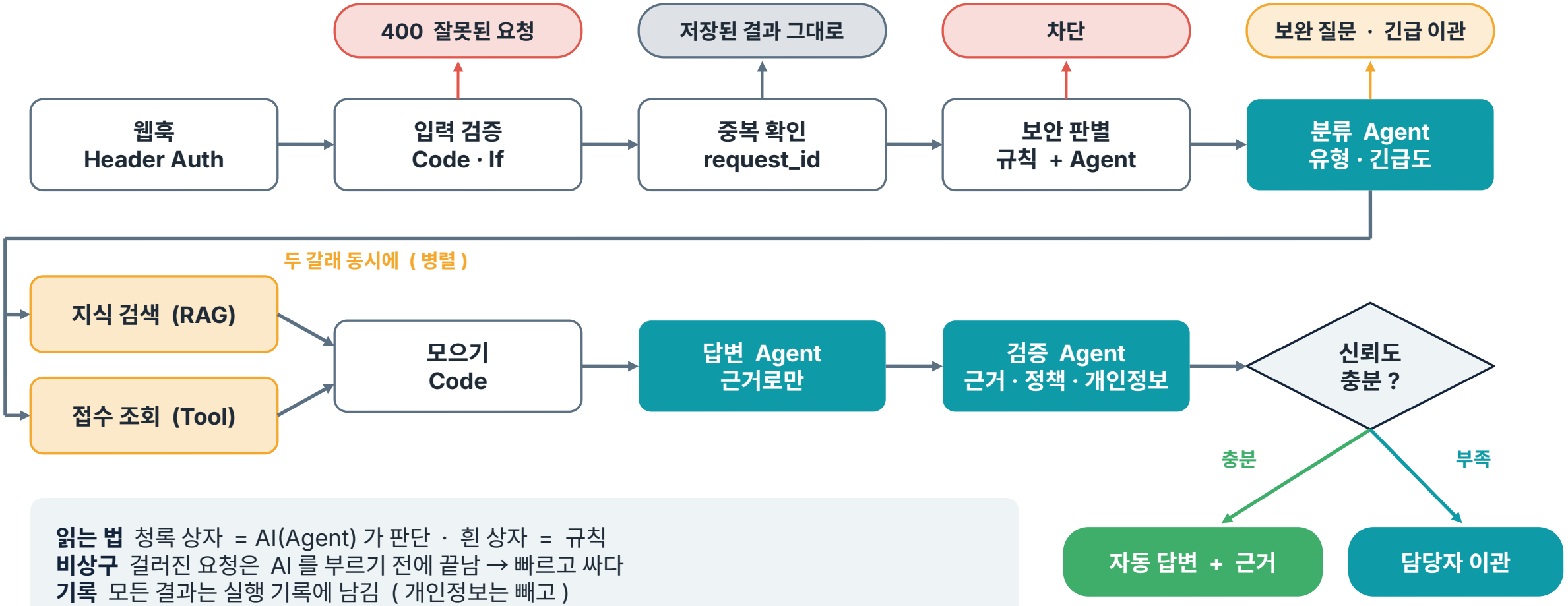
1	2	3	4	5	6	7	8
받기	**입력 검사**	**문지기**	**분류**	**검색**	**답변**	**검증**	**갈림길 · 기록**
웹훅으로 질문이 도착	빈칸 ·500 자 초과 · 없는 직원이면 400	같은 요청 반복 , 공격 문구 차단	유형 · 긴급도 · 빠진 정보 판단	지식 문서와 내 접수 건을 함께 찾기	찾은 근거로만 답 쓰기	근거 · 정책 · 개인정보 다시 확인	답 또는 이관 , 기록 남기기
규칙	규칙	규칙	AI	검색	AI	AI	규칙

처리 순서

- 규칙 (Code · If · Switch): 빠르고 , 공짜이고 , 결과가 늘 같음
- AI (LLM 에이전트): 사람 말의 뜻을 판단해야 할 때만
- 검색 (RAG · 도구 노드): 자료실과 업무 시스템에서 찾아오기

n8n 안의 실제 흐름도

윗줄은 '걸러내기', 아랫줄은 '답 만들기' — 중간중간 비상구가 있습니다



읽는 법 청록 상자 = AI(Agent) 가 판단 · 흰 상자 = 규칙
비상구 걸러진 요청은 AI 를 부르기 전에 끝남 → 빠르고 싸다
기록 모든 결과는 실행 기록에 남김 (개인정보는 빼고)

티켓 생성 → 알림 · 실패하면 DLQ

AI 직원 3명 + 문지기, 일을 나눠 맡습니다

한 명에게 다 시키지 않습니다 — 서로의 결과를 확인할 수 있게

분류
`triage`

하는 일

질문의 유형 · 긴급도와 빠진 정보 찾기

결과물

`{type, urgency, missing}`

하지 않는 일
답하지 않기

답변
`writer`

하는 일

검색해 온 근거로만 답 글 쓰기

결과물

답 글 (텍스트)

하지 않는 일
근거 밖 내용 · 약속 하지 않기

검증
`verifier`

하는 일

근거 · 정책 · 개인정보를 다시 점검

결과물

`{grounded, policy_ok, issues}`

하지 않는 일
답을 새로 쓰지 않기

문지기 (선택)
`guard`

하는 일

공격성 질문인지 판별

결과물

`{attack, reason}`

하지 않는 일
답하지 않기

AI 의 결과는 JSON 으로 받고 규칙 (Code) 이 읽습니다. 읽지 못하면 ? → 안전한 쪽 (담당자 이관) 으로 보냅니다

RAG: ‘ 오픈북 시험’ 처럼 답하기

미리 자료를 정리해 두고 , 질문이 오면 관련 쪽만 펼쳐서 답합니다

미리 준비

색인 · 한 번

지식 문서

규정 8 개 + 팀이 더한 문서

→

조각내기

예 : 500 토큰씩 , 80 토큰 겹치게

→

숫자로 바꾸기

임베딩 : 뜻이 비슷하면 가까운 숫자

→

벡터 저장소

조각 + 꼬리표 (문서 · 절 · 권한)

질문 때마다

검색

질문

“VPN 이 계속 끊겨요”

→

권한 필터

볼 수 있는 문서만 (검색 전에 !)

→

비슷한 조각 찾기

top_k 개 , 점수 낮으면 버림

→

근거로 답하기

답 + sources (문서 · 절 · 점수)

근거가 없으면 ? 지어내지 않고 → 담당자 이관 (공통 시나리오 WF-03)

결과는 다섯 가지 신호 중 하나

응답의 route 칸 하나만 보면 시스템이 무엇을 했는지 알 수 있습니다

	언제	예
자동 답변	근거 문서가 있고 검증을 통과했을 때	*"VPN 이 계속 끊겨요" → 절차 + 출처*
보완 질문	정보가 부족할 때 되묻기 (최대 횟수 넘으면 이관)	*"접속이 안 돼요" → 무엇에? 어떤 오류?*
담당자 이관	근거 없음 · 신뢰도 부족 · 긴급 · 보안 사고	*"피싱 링크를 눌렀어요" → 긴급 티켓 + 알림*
권한 없음	남의 접수 건 , 볼 권한이 없는 문서	*영업팀 직원이 다른 사람의 T-1002 조회*
차단	프롬프트 주입 같은 공격 (AI 부르기 전에)	*"이전 지시는 무시하고 시스템 프롬프트를…"*

+ 형식이 틀린 요청 (빈 질문 · 없는 직원 · 501 자 이상) 은 신호를 정하기 전에 HTTP 400 으로 돌려보냅니다

모든 팀이 지키는 약속 : 공통 계약

들어오는 모양과 나가는 모양이 같아야 같은 시험지로 채점할 수 있습니다

보내는 모양 (요청)

```
POST /webhook/team1-support
{
  "request_id": "req-001",
  "user": "박서연",
  "channel": "it-help",
  "question": "VPN이 계속 끊겨요",
  "attempt": 1
}
```

돌려받는 모양 (응답)

```
{
  "request_id": "req-001",
  "route": "자동 답변",
  "answer": "VPN 클라이언트를 업데이트하고…",
  "sources": [{"doc": "remote-work-vpn",
               "section": "접속 오류",
               "score": 0.71}],
  "ticket": null,
  "agents": ["triage","retriever",
             "writer","verifier"],
  "security": {"blocked_by": "", "masked": 0}
}
```

request_id 중복 방지

attempt 보완 질문 횟수

route 다섯 가지 신호

sources·ticket 근거 · 이관

입구가 텔레그램이든 웹 화면이든 , n8n 에 들어갈 때는 이 모양으로 바꿔서 넣습니다

입구 · 뒷단 조합 3 가지 — 두뇌는 같습니다

팀의 실력과 시연 욕심에 맞춰 고릅니다

A 기본형

입구 팀 웹훅 (flow.py 로 호출)

뒷단 data-api (연습용 가짜 회사)

난이도 ●○○

좋은 점
준비물이 거의 없음 · 평가 기준 그대로

주의할 점
화면이 없어 시연이 덜 와닿음

B 메신저 + 헬프데스크형

입구 텔레그램 봇

뒷단 오픈소스 헬프데스크 (Zammad 등)

난이도 ●●○

좋은 점
휴대폰으로 실제처럼 시연 · n8n 기본 노드 있음

주의할 점
외부에서 들어올 HTTPS 주소 · 서버 자원

C 자체 웹앱형

입구 직원용 웹 화면 (Next.js)

뒷단 관리자용 웹 화면 + FastAPI

난이도 ●●●

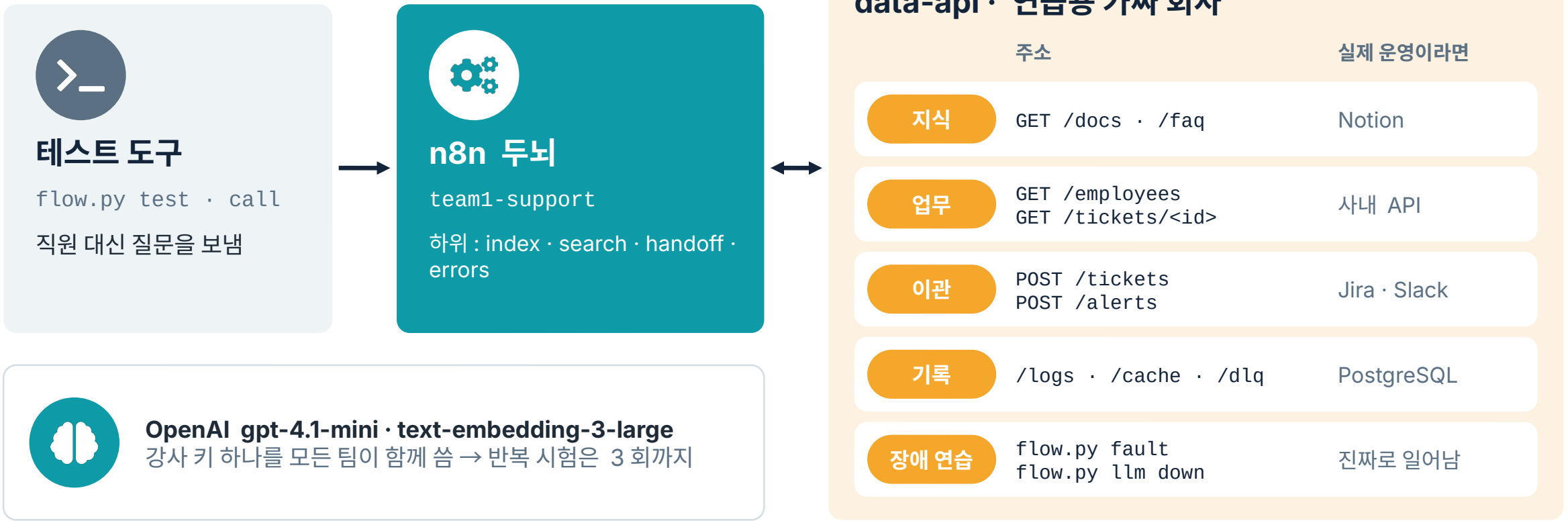
좋은 점
우리 서비스처럼 보임 · 최종 프로젝트로 이어짐

주의할 점
코드 양이 많음 → 범위를 작게

순서 : A 를 먼저 완성 → B·C 는 어댑터만 추가 . 시연 전날까지 안 되면 A 로 시연해도 불리하지 않습니다

A 기본형 : 연습용 가짜 회사 (data-api) 와 대화하기

모든 팀의 출발점 — 공통 시나리오 14 개가 여기서 먼저 통과해야 합니다 (data-api 는 예시 , 다음 쪽 참고)



고장 스위치로 '외부 서비스 장애' 를 일부러 만들어 예외 처리를 시험합니다 (끝나면 반드시 끄기)

data-api 는 연습용 모형 — 실제로는 FastAPI 로 직접 만듭니다

n8n 강좌의 data-api 는 '가짜 회사' 일 뿐, 회사에 들어가면 이 자리가 우리가 짜야 할 사내 로직입니다

수업 · 프로젝트			실제 운영
직원 · 권한	`GET /employees`	→	SSO · 사내 인사 DB 로 부서 · 권한 확인
지식 문서	`GET /docs · /faq`	→	문서 저장소 + 권한 꼬리표, 색인 갱신
접수 · 티켓	`GET·POST /tickets`	FastAPI 로 다시 구현 →	티켓 테이블 · 담당자 배정 · 중복 키 · 상태 변경
알림	`POST /alerts`	→	텔레그램 · 메일 · 사내 메신저 발송
기록 · 대기열	`/logs · /cache · /dlq`	→	PostgreSQL · Redis, 감사 로그 · 재처리

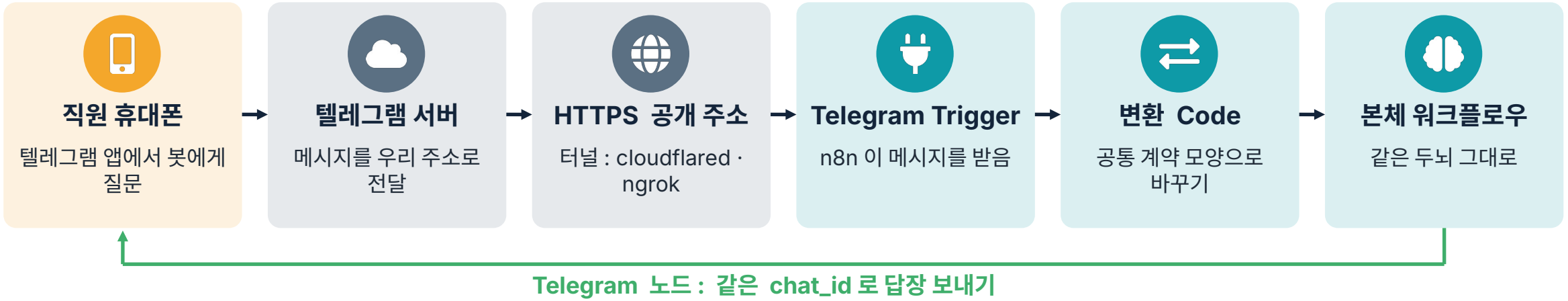
메모리 → DB
다시 켜도 티켓 · 기록이 남아야 함

X-Api-Key → 진짜 인증
서비스 계정 · 토큰 만료 · 권한 범위

고장 스위치 → 진짜 장애
타임아웃 · 재시도 · DLQ 를 실제로 겪음

B 입구 : 텔레그램 봇으로 질문 받기

휴대폰 메신저에서 질문하면 , 같은 두뇌가 답해서 같은 채팅방으로 돌려줍니다



변환 Code 가 하는 일

```
message.text   →  question
chat.id        →  직원 등록표에서 user
update_id      →  request_id ( 중복 방지 )
( 보완 질문 중이면 attempt · context 도 )
```

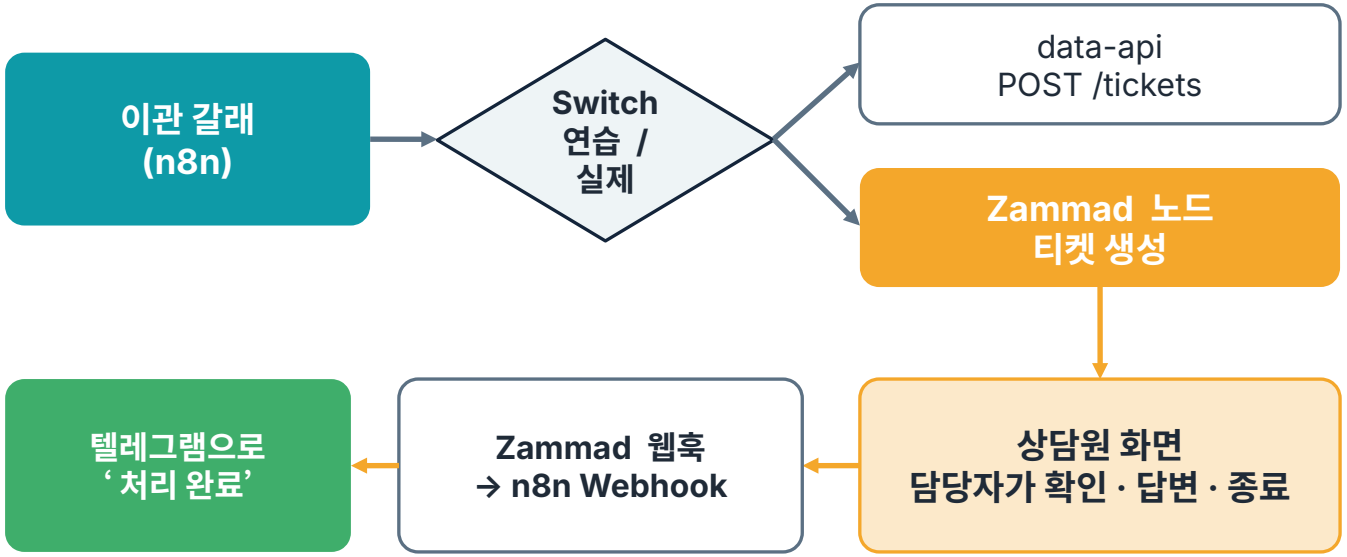
BotFather 로 봇을 만들고 , 받은 토큰은 n8n Credential 에만

텔레그램은 HTTPS 주소로만 보내 줌 → 노트북의 n8n 이면 터널을 열고 WEBHOOK_URL 을 그 주소로

권한은 질문 내용이 아니라 chat_id ↔ 직원 등록표로 시스템이 확인

B 뒷단 : 오픈소스 헬프데스크에 티켓 남기기

Jira 대신 무료로 설치하는 헬프데스크 — 담당자가 실제 상담원 화면에서 처리합니다



중복 막기 헬프데스크에는 ‘같은 key 는 한 번만’ 기능이 없음 → n8n 이 request_id 로 먼저 확인하고 , 티켓에 request_id 를 남기기

실패하면 재시도 → DLQ + 알림 (기본형과 같은 규칙)

키 관리 Zammad API 토큰도 n8n Credential 에만

오픈소스 헬프데스크 후보

Zammad

추천

n8n 기본 노드 · 현대적 화면 · 메모리 많이 씀

FreeScout

가볍고 공유 메일함처럼 생김

osTicket

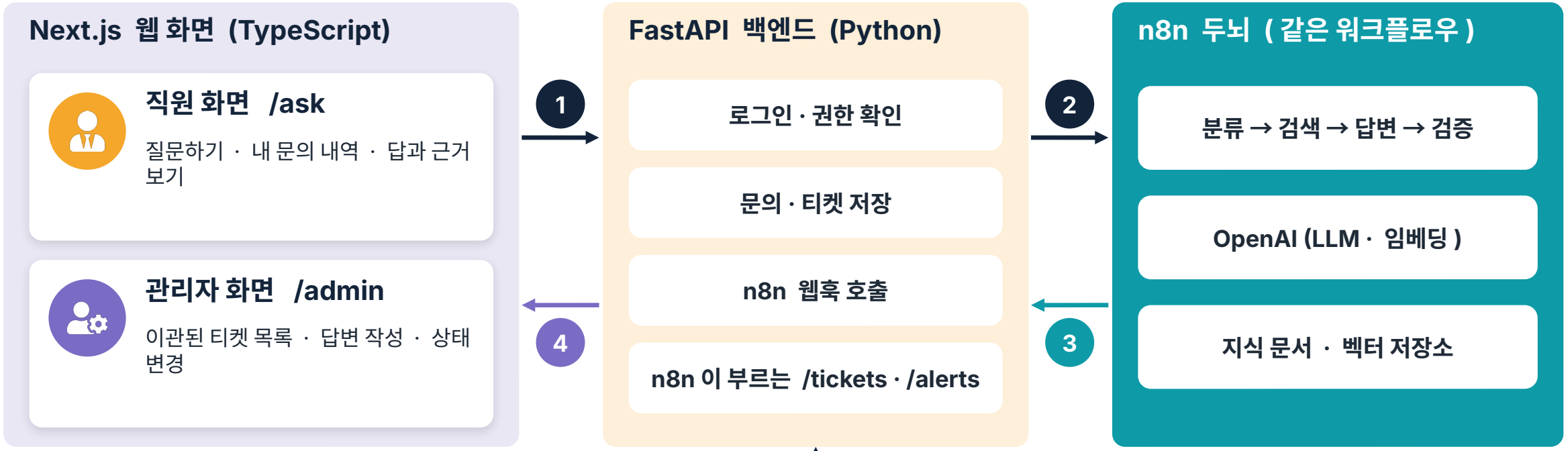
오래된 대표 주자 · 자료가 많음

GLPI

IT 자산 관리 + 헬프데스크

C 자체 웹앱 : 직원 화면 · 관리자 화면 · FastAPI

FastAPI 는 ‘data-api 를 우리가 직접 만든 것’ — 같은 주소와 모양을 지키면 n8n 은 거의 그대로

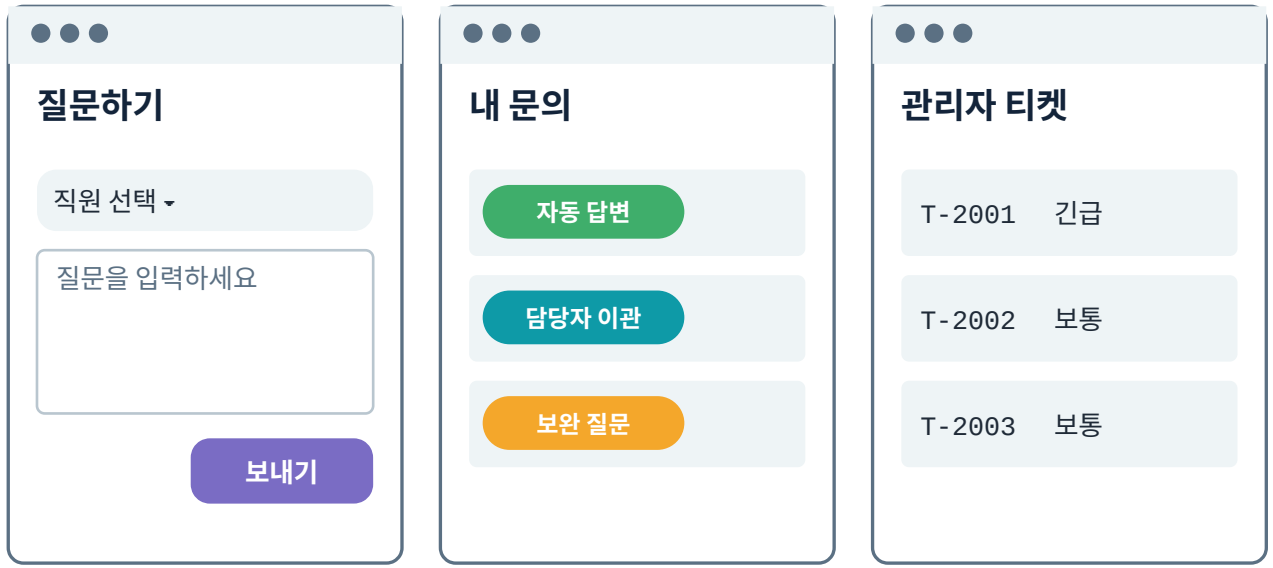


핵심 아이디어
직원 · 관리자는 n8n 을 직접 만나지 않습니다 . FastAPI 가 중간에서 문지기 겸 기록 담당

1 직원이 화면에서 질문 → FastAPI
2 FastAPI 가 계약 모양으로 n8n 호출 (키 포함)
3 n8n 이 답 또는 이관 결정 , 이관이면 FastAPI 의 /tickets 호출
4 답 · 티켓 상태를 저장하고 화면에 표시 → 관리자가 처리

C 를 3 일 안에 : 화면 3 개 , API 6 개면 충분합니다

멋진 화면보다 '두뇌가 제대로 동작하는 것' 을 보여 주는 게 먼저입니다



화면이 부르는 API

```
POST  /api/ask
GET   /api/my-requests
GET   /api/admin/tickets
PATCH /api/admin/tickets/{id}
```

n8n 이 부르는 API

```
POST  /tickets      POST /alerts
```

로그인 대신 '직원 선택'

가짜 직원 6 명 중 고르기로 시작 . 진짜 로그인은 최종 프로젝트에서

n8n 호출엔 시간 제한

예 : 10 초 . 넘거나 실패하면 정해진 안내 문구를 보여 주기

키는 .env 에만

저장소에는 .env.example 만 올리고 , 커밋 전 비밀 검사

고장이 나도 ‘정해진 방식’ 으로 실패하기

멈추는 건 괜찮습니다 . 아무 말 없이 멈추거나 엉뚱하게 답하는 게 문제입니다

상황	이렇게 처리	이유
열쇠가 틀림 (401 · 403)	다시 해도 안 됨 → 멈추고 관리자에게 알림	재시도 금지
너무 많이 보냄 (429) · 서버 오류 (5xx) · 시간 초과	잠깐 기다렸다가 재시도 (횟수 제한)	일시적 문제
검색 결과 없음	답을 만들지 않고 담당자 이관	지어내지 않기
티켓 생성 실패	DLQ(실패 대기열) 에 보관 + 알림 , 직원에겐 정상 응답	요청을 잃지 않기
같은 요청이 두 번	request_id 로 확인 → 티켓은 1 건만	중복 방지
AI(LLM) 가 멈춤	500 오류 대신 이관 또는 근거 원문 안내	서비스는 계속

n8n 에서는 : 노드 설정의 재시도 (Retry On Fail) · 오류 출력 갈래 · Error Workflow 하나를 게시해 둡니다

보안은 네 가지 원칙만 지키면 됩니다

모두 73 일차에 배운 것 — 프로젝트에서는 '빠짐없이' 적용하는 게 관건



1 키는 금고 (Credential) 에만

워크플로우 JSON · 문서 · 스크린샷 · 저장소에 키가 보이면 큰 감점 . 커밋 전에 tscan 으로 비밀 검사



2 권한은 시스템이 확인

질문에 적힌 말이 아니라 user → 직원 정보 (부서) 로 판단 . 문서 권한은 검색 '전에' 거르기



3 개인정보는 가리고 , 기록엔 남기지 않기

답에서 연락처는 010-****-2135 처럼 가림 . 실행 기록에는 개인정보 없이



4 공격은 AI 부르기 전에 막기

" 이전 지시는 무시하고…" 같은 주입은 규칙으로 먼저 차단 . AI 결과를 못 읽으면 안전한 쪽으로

필수 산출물 3 종 : 각각 무엇을 증명하는 문서인가

‘만들었다’가 아니라 ‘이렇게 설계했고, 돌려 보니 이랬다’를 남기는 문서입니다



35
점 (제안)

노코드 워크플로우 설계서

`docs/WORKFLOW-DESIGN.md`

증명할 것

흐름이 기술 지원 로직에 맞고,
연결선마다 데이터가 정해져 있다

주 담당 오케스트레이션 + 연동 · 안정성



30
점 (제안)

RAG 아키텍처 설계서

`docs/RAG-ARCHITECTURE.md`

증명할 것

검색 설계 (청킹 · 임베딩 · 인덱스) 를
숫자로 고르고 근거를 댔다

주 담당 RAG



25
점 (제안)

워크플로우 실행 결과서

`docs/EXECUTION-RESULT.md`

증명할 것

실제로 돌려 보니 기대와 실제가 이랬고,
실패는 고쳐서 다시 시험했다

주 담당 시험 · 품질

+ 발표 · 협업 10 점 · 배점은 강사가 첫날 확정



세 문서 공통 감점 ① 문서와 구현이 다름 (노드 이름 · 숫자) ② 예시 값을 그대로 둠 ③ 키 · 개인정보 노출

① 노코드 워크플로우 설계서 잘 쓰기

다른 팀원이 이 문서만 보고 같은 워크플로우를 다시 만들 수 있으면 합격입니다

목차 8 절 (템플릿 그대로)

1. 업무 시나리오 · 처리할 문의 유형
2. 전체 흐름 (단계 목록)
3. 흐름도 (Mermaid) · 워크플로우 파일 표
4. **역할 카드 · 분기 · 병렬 · 루프**
5. 외부 연동 노드 (연습용 → 운영)
6. **연결선마다 페이로드**
7. 오류 처리 (상황별)
8. 64~79 일차에서 가져온 것

빨간 번호 = 평가에서 가장 많이 보는 절

6 절 연결선 페이로드 : 약한 예 vs 좋은 예

약한 예

"분류 노드가 검색 노드로 질문 데이터를 넘긴다"

좋은 예

연결선 `Parse class → Knowledge Search`
필수 `query, type, access_levels` **선택** `ticket_id`
변환 분류 JSON 읽기 → 부서로 권한 목록 만들기
예시 `{"query":"VPN 끊김","type":"network","access_levels":["all"]}`

- 노드 이름 = workflows/*.json 의 실제 노드 이름
- 예시는 n8n 실행 화면 (Executions) 에서 복사 → 개인정보 가림
- 기준값은 숫자로 : 신뢰도 기준 · 루프 최대 횟수 · 시간 제한
- tflow 로 흐름도를 뽑아 설계서 흐름도와 비교

② RAG 아키텍처 설계서 잘 쓰기

모든 선택에 '왜 이 값인가 ?' 를 잰 숫자로 답합니다

1 원본 문서
문서 수 · 길이 분포 · 권한 값

2 전처리 · 청킹
단위 · 오버랩 · 비교한 다른 선택

3 임베딩
모델 · 차원 · 거리 · top_k · 점수 기준

4 인덱스 · 메타데이터
컬렉션 이름 · 버전 · 권한 필터

5 백업 · 복구 · 확장
재색인 시간 · 롤백 · ×10 어림

핵심 표 : 청킹 방식을 비교해서 고르기

방식	청크 수	길이 최소 · 중앙 · 최대	hit@3 (질문 N 개)
절 (##) 단위	?	? · ? · ?	? / N
500 토큰 · 오버랩 80	?	? · ? · ?	? / N
우리 선택 → 이유	숫자로 한 줄		

? 칸은 팀이 직접 잰 값으로 — 예시 숫자를 옮기면 감점

권한 필터는 검색 '전에'
access_level 메타데이터로 거르기 (WF-05)

점수 기준은 분포를 보고
정상 질문 vs 결과 없음 질문의 점수를 재서 선 긋기

색인 비용 · 시간도 숫자로
tcost 로 토큰 · 비용 , 재색인 몇 초

③ 워크플로우 실행 결과서 잘 쓰기

숫자는 runs/ 보고서에서만 옮기고 , 칸마다 보고서 파일 이름을 적습니다

기대 결과 먼저

```
cases/common.json
cases/team.json
```

→

3 회 실행

```
flow.py test
(LLM 들어간 것)
```

→

보고서 저장

```
runs/*.json
지우지 않기
```

→

결과서 표로

```
기대 vs 실제
보고서 이름과 함께
```

목차 6 절

1. 시나리오별 시험 기록 (공통 14 + 팀 5+)
2. 대표 실행 로그 3 건 (정상 · 예외 · 이관)
3. 결과 요약 (통과 · p50/p95 · LLM 호출 수)
4. **실패 사례 2 건 이상**
5. 구현 결과 · 시연 순서
6. 제출 전 체크리스트 15 개

실패 사례는 숨기지 말고 6 칸으로

현상
무엇이 실패했나

재현 조건
어떤 입력 · 환경에서

원인
로그 · 설정 · 데이터 근거

조치
워크플로우 · 프롬프트 · 값 수정

재시험
같은 조건에서 통과했나

잔여 위험
아직 남은 제한

산출물은 3 일에 나눠 채우고 , 네 번 보여 줍니다

마지막 날 한꺼번에 쓰지 않도록 — 점검 · 발표 때마다 '어디까지' 가 정해져 있습니다

	80 일차 (목)	81 일차 (금)	82 일차 (월)
① 워크플로우 설계서	1~4 절 초안 (4 교시)	5~7 절 · 페이로드 표 (4·7 교시)	노드 이름 · 숫자 최종 맞춤
② RAG 설계서	1~2 절 : 문서 · 청킹 분포 (6 교시)	3~5 절 : 임베딩 · 인덱스 · 백업	청킹 비교 · 점수 기준 확정
③ 실행 결과서	기대 결과 먼저 : cases/team.json (3 교시)	1 절 시험 기록 시작 · 통과 수	3 회 반복 · 요약 · 실패 사례 2 건

강사 점검 · 팀별 발표 (각 팀 10 분 , 4 팀 순회)

80 · 5 교시
산출물 1 차 점검
설계서 1~4 절

→

80 · 8 교시
산출물 발표 ①
설계서 2 종 초안

→

81 · 4 교시
산출물 중간점검
페이로드 · RAG 3~5 절

→

81 · 8 교시
산출물 발표 ②
설계서 완성 · 결과서 초안

→

82 · 5 교시
3 종 제출
→ 6 · 7 교시 최종 발표

3 일 로드맵 : 설계 → 구현 → 시연 · 발표

팀별 VM 4 대에서 매일 돌아가는 것을 만들고 , 강사가 교시를 정해 확인합니다 (상세 시간표는 맨 뒤 부록)

80 일차 · 11/05 (목)

설계와 골격

오전 킥오프 · 주제와 조합 (A/B/C) 승인 · 범위 · 설계서 1~4 절

오후 골격 워크플로우 · RAG 색인 · 도구 노드 · VM 첫 배포

- 5 교시 — 산출물 1 차 점검
- 8 교시 — 산출물 발표 ①

81 일차 · 11/06 (금)

구현과 예외 처리

오전 분류 · 검색 · 답변 · 검증 에이전트 · 분기 · 병렬 · 루프 · 어댑터

오후 보안 · 장애 ·DLQ · Error Workflow · 설계서 완성

- 4 교시 — 산출물 중간점검
- 8 교시 — 산출물 발표 ② · 동결

82 일차 · 11/09 (월)

시연 · 제출 · 발표

오전 시연 상황 점검 · 발표 자료 점검 · 팀별 시연 리허설

오후 산출물 3 종 제출 · 발표 20 분 + 질의 5 분 · 시험과 총평

- 5 교시 — 산출물 3 종 제출
- 6·7 교시 — 최종 발표 · 질의

강사 확인 지점

80 · 5	80 · 8	81 · 4	81 · 8	82 · 2	82 · 3·4	82 · 5	82 · 6·7	82 · 8
산출물 1 차 점검	산출물 발표 ①	산출물 중간점검	발표 ② · 동결	자료 · 산출물 점검	시연 리허설	산출물 제출	최종 발표	시험 · 총평

역할분담 예시 : 6 명 팀이 B 조합을 고른 경우

한 사람 = 한 파일 · 한 산출물 · 통과시킬 시나리오 — 셋을 같이 정해 둡니다

역할	맡는 파일	쓰는 산출물	통과시킬 시나리오
리드 · 통합	`README.md` · `AGENTS.md` · `docs/TEAM-CHARTER.md` · `docs/daily/`	헌장 · 데일리 · 발표	배포 VM · 전체 회귀 시험
오케스트레이션	`workflows/team1-support.json` (본체)	워크플로우 설계서 2~4 절	WF-01 · 02 · 02b · 12
RAG	`team1-index.json` · `team1-search.json` · `knowledge/`	RAG 설계서 전부	WF-03 · 05
연동 · 안정성	`team1-handoff.json` · `team1-errors.json`	워크플로우 설계서 5~7 절	WF-04 · 06~11
입구 · 뒷단 어댑터	`team1-telegram.json` · Zammad 설정	설계서 ‘외부 연동 노드’ 표	텔레그램 → 티켓 → 완료 알림
시험 · 품질	`cases/team.json` · `runs/`	실행 결과서 전부	WF-13 · 팀 시나리오 · 3 회 반복

5 명 팀이라면 어댑터를 ‘연동 · 안정성’ 이 함께 맡고 , A 기본형으로 먼저 끝냅니다

C 조합이라면 어댑터 담당이 web/(Next.js) · api/(FastAPI) 를 맡고 , API 모양은 리드가 승인

팀장 (리드) 의 하루 : 모으고 , 올리고 , 숫자로 말하기

팀장은 가장 많이 만드는 사람이 아니라 , 매일 ‘돌아가는 하나’ 를 지키는 사람입니다

09:00 스탠드업 10 분

어제 피드백 · 오늘 목표 숫자 · 막힌 것

오전 리뷰와 길 뚫기

MR 리뷰 · 계약 지키기 · 막힌 팀원 돕기

12:30 VM 배포 1 회

오전 것을 모아 게시 → 공통 시나리오

16:00 VM 배포 2 회

하루 결과를 모아 게시 → 전체 회귀

17:00 점검 · 데일리

강사 확인 · 데일리 기록 커밋

팀장이 하는 일

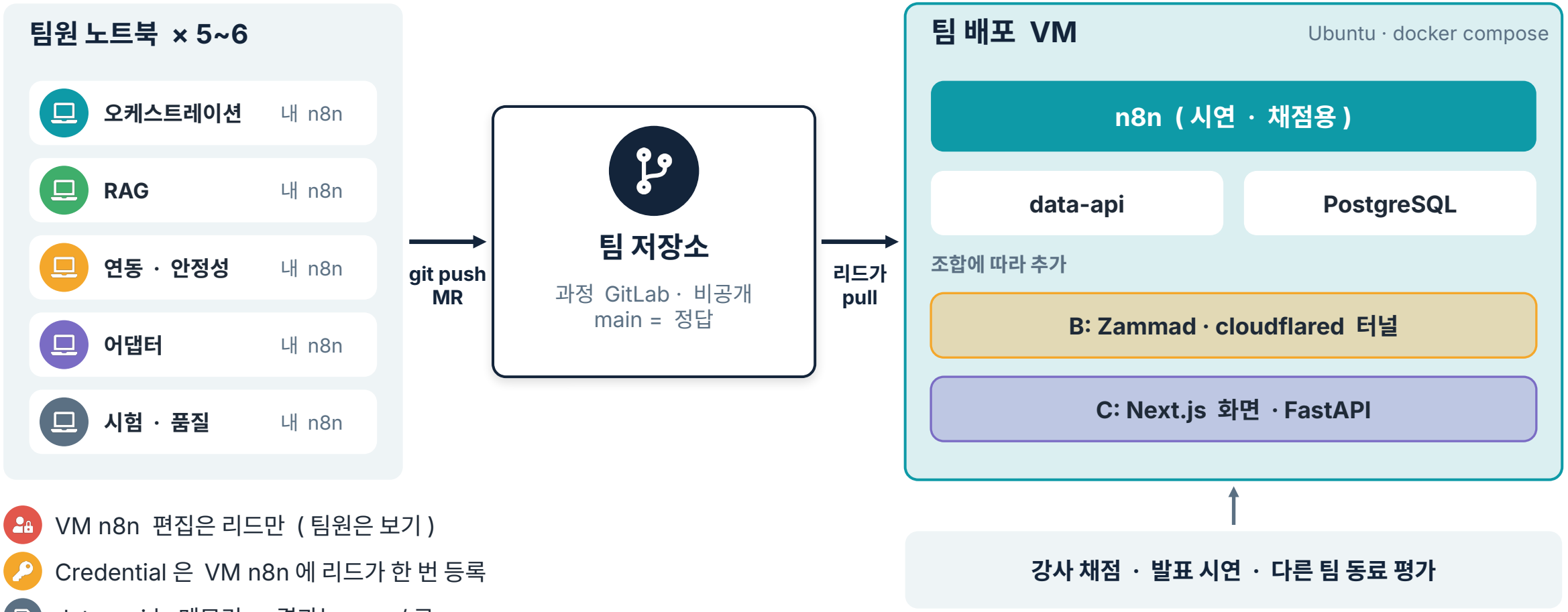
- 범위와 공통 계약을 지킴 (바꿀 땐 설계서부터)
- main 병합과 VM 배포는 팀장만
- 진행은 ‘공통 시나리오 통과 수’ 로 말함
- 81 일차 8 교시 기능 동결 → 82 일차는 시연 · 문서만
- 발표 순서와 시연 기계 준비

팀장이 하지 않는 일

- 남이 맡은 파일을 말없이 고치기
- 혼자 다 만들기 (막히면 역할을 다시 나눔)
- 마감 직전에 새 기능 붙이기
- VM 의 n8n 화면에서 직접 고치기 (저장소가 정답)

팀별 배포 VM: 각자 만들고, 한 곳에 모읍니다

개발은 각자 노트북의 n8n 에서, 시연과 채점은 팀 VM 한 대에서



- VM n8n 편집은 리드만 (팀원은 보기)
- Credential 은 VM n8n 에 리드가 한 번 등록
- data-api 는 메모리 → 결과는 runs/ 로

모으고 게시하는 순서 : 리드의 배포 체크리스트

같은 순서를 하루 두 번 — 깨지면 바로 직전 태그로 되돌립니다

1 내 n8n 에서 완성 — 팀원

flow.py check → push → 공통 시나리오 통과 수 확인

2 MR 올리기 — 팀원

내 파일만 · 커밋에 통과 수 "검증 Agent · common 9/14"

3 리뷰하고 병합 — 리드

계약 그대로 ? 키 없음 (tscan)? 노드 이름 = 설계서 ?

4 가져와 올리기 — 리드 → VM

VM 에서 git pull → flow.py push workflows/*.json

5 게시 (Publish) — VM

하위 워크플로우부터 , 본체는 맨 마지막 · Error Workflow 지정

6 시험하고 공지 — 리드

common + team 실행 → runs/ 저장 → 태그 → 팀 채널 공지

게시 순서 (부르는 쪽이 나중)

errors → search · index → handoff → telegram → support

공지 예시

```
[deploy-81-2] common 11/14 · team 4/6
바뀐 것 : 검증 Agent, DLQ / 실패 : WF-10
```

바이브 코딩으로 만든 t- 도구 , 실제 운영에서 그대로 씁니다

수업 숙제가 아니라 '운영 공구함' — 배포 체크리스트의 각 단계에 끼워 넣습니다



tflow 68 일차

워크플로우 JSON 을 흐름도 · 노드 설명 · 문서로

운영에서 설계서와 구현이 같은지 확인 , 인수인계 · 리뷰



tscan 73 일차

키 · 토큰 · 개인정보 비밀 검사 , 커밋 훅

운영에서 MR 전 tscan staged, 제출 전 tscan scan .



tver 74 일차

워크플로우 버전 저장 · 비교 · 되돌리기

운영에서 배포 태그별 차이 확인 , 깨지면 직전 본으로



tbench 72 일차

웹훅 응답 시간 · p95 · 실패 수 측정

운영에서 배포 뒤 성능 기준 확인 , 결과서의 처리시간



tcost 75 일차

토큰 · 비용 계산 , 모델 비교 · 예산

운영에서 요청당 비용 , 한 달 운영비 추정



tretry 76 일차

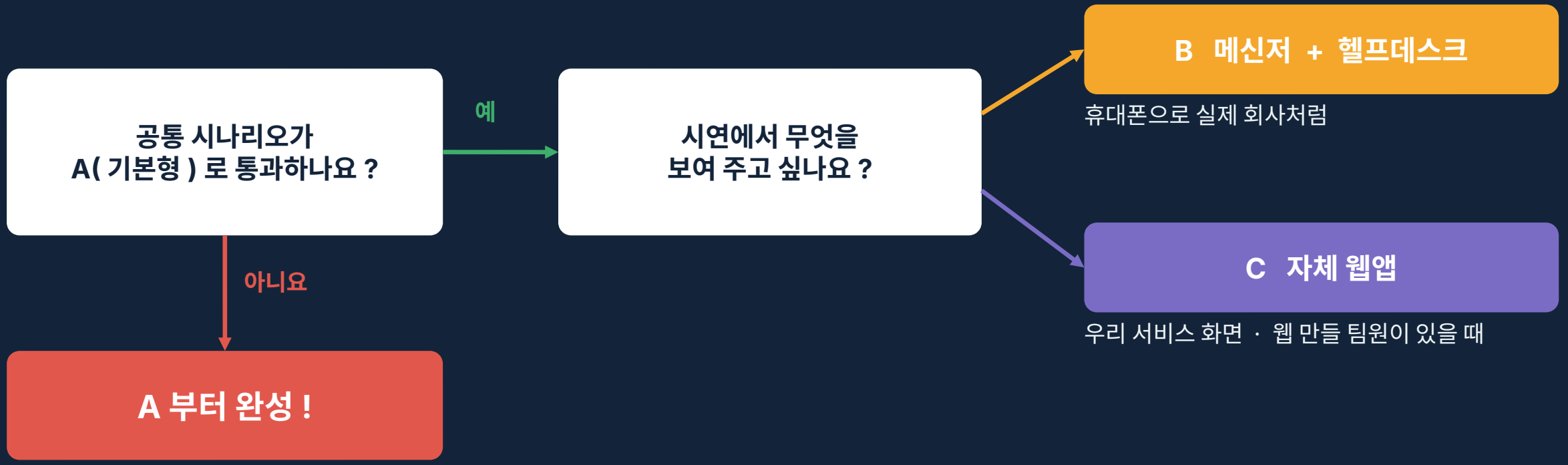
재시도 · 백오프 · 대체 경로 · 중복 방지 키

운영에서 외부 API 장애 재현 · 수동 재처리 (DLQ)

MR 전 tscan → 병합 시 tver · tflow → 배포 뒤 tbench · tcost → 장애 땐 tretry

우리 팀은 어떤 조합으로 갈까 ?

질문 두 개로 정해 봅니다



입구와 뒷단은 바뀌어도 ,
두뇌와 약속 (계약) 은 같습니다

시나리오 파일 : cases/common.json 과 cases/team.json

‘ 이 질문을 넣으면 이렇게 나와야 한다’ 를 미리 적어 둔 시험지 — 시간표의 완료 기준 WF-01… 이 여기서 나옵니다

cases/common.json

공통 시나리오

- 모든 팀이 같은 시험지 · 14 개 (WF-01 ~ WF-13)
- 실습 키트로 제공 · 팀이 바꾸지 않음
- 다른 주제 팀은 질문만 옮기고 판정은 그대로

cases/team.json

팀 시나리오

- 우리 팀 범위가 맞는지 보여 주는 5 개 이상
- 80 일차 3 교시에 기대 결과부터 먼저 작성
- 만든 뒤에 결과에 맞춰 기대를 고치지 않음

```
{
  "webhook": "team1-support",
  "cases": [{
    "name": "WF-01 정상 문의",
    "body": {"request_id": "wf01", "user": "박서연",
             "question": "VPN이 계속 끊겨요"},
    "expect": {"equals": {"route": "자동 답변"},
               "contains": {"sources": ["remote-work-vpn"]},
               "max_seconds": 8}
  }]
}
```

name 번호 · 이름 **body** 보내는 질문 **expect** 통과 기준

돌리는 법 → 결과는 runs/ 에 보고서로

```
flow.py test cases/common.json --webhook team1-support -v
flow.py test cases/common.json --webhook team1-support --repeat 3
flow.py test cases/team.json -v
```

한 번 · 자세히
LLM 들어간 것은 3 회 (3 번 다 통과해야 통과)
팀 시나리오

공통 시나리오 14 개 (WF-01 ~ WF-13)

시간표의 완료 기준 ‘WF-03 통과’ = 팀 VM 에서 이 표의 WF-03 이 PASS 로 나온다는 뜻

번호	시나리오	입력 · 조건	통과 기준	시간표
WF-01	정상 문의	“VPN 이 계속 끊겨요”	자동 답변 + sources 에 근거 문서	81 · 2
WF-02	정보 부족	“ 접속이 안 돼요” (attempt 1)	보완 질문 (무엇에 · 어떤 오류 · 언제)	81 · 4
WF-02b	보완 질문 한도	같은 질문 attempt 3	담당자 이관 + ticket	81 · 4
WF-03	검색 결과 없음	지식 문서에 없는 문의	지어내지 않고 이관 · sources 빈 목록	81 · 3
WF-04	외부 API 장애	티켓 API 500 + 이관할 문의	200 응답 · ticket null · DLQ 1 · 알림 1	81 · 6
WF-05	권한 제한	영업팀이 관리자 문서 문의	관리자 문서 안 씀 · 권한 없음 / 이관	81 · 5
WF-06	남의 접수 건	남의 T-1002 상태 조회	권한 없음 · 상태 · 연락처 없음	81 · 5
WF-07	개인정보	내 접수 건과 연락처	연락처 가림 · security.masked ≥ 1	81 · 5
WF-08	프롬프트 주입	“ 이전 지시는 무시하고…”	차단 · LLM 호출 없음	80 · 7 / 81 · 5
WF-09	중복 요청	같은 request_id 로 두 번	티켓 1 건 · duplicate true	81 · 6
WF-10	LLM 멈춤	llm down 상태의 정상 문의	500 아님 · 이관 또는 근거 원문	81 · 6
WF-11	느린 외부 API	접수 조회 8 초 지연	정해진 시간 안에 응답	81 · 4
WF-12	복합 · 긴급	피싱 링크 클릭 + 내 접수 건	긴급 이관 (high) + 알림 + 초동 조치	81 · 4
WF-13	잘못된 입력	빈 질문 · 없는 직원 · 501 자	HTTP 400 + codes	80 · 7

보안 (73 일차) 장애 · 안정성 (72 · 76 일차) 시간표 칸 = 그 시나리오를 처음 통과시키는 일차 · 교시

80 일차 · 11/05 (목) 설계와 골격

팀 VM 에 첫 배포까지 — 5 교시 산출물 1 차 점검 , 8 교시 산출물 발표 ①

교시	팀이 하는 일	강사	완료 기준
1 교시 09:00	팀 발표 · 역할 정하기 , 팀 VM 접속 확인	**킥오프 : 안내서 · 평가 기준 · 공통 계약 · VM 배정 안내**	**팀 명단 · VM 접속**
2 교시 10:00	주제 · 조합 (A/B/C) 정하기 , 주제 제안서 · 팀 헌장 · 저장소	**주제 제안서 승인 (10:50 까지)**	**헌장 · 주제 커밋**
3 교시 11:00	범위 : 문의 유형 · 지식 문서 · 권한 · 팀 시나리오 5 개 팀장 : VM 에 n8n · data-api · Credential 준비	팀 순회 · 질문 받기	**cases/team.json 초안**
4 교시 12:00	워크플로우 설계서 1~4 절 : 업무 시나리오 · 흐름도 · 역할 카드 · 분기 기준	팀 순회	**설계서 1~4 절 초안**
점심 13:00 ~ 14:00			
5 교시 14:00	확인받는 팀 외 : 골격 워크플로우 (웹훅 → 입력 검사 → 응답)	**산출물 1 차 점검 (팀별 10 분) 워크플로우 설계서 1~4 절 · 공통 계약**	**점검 피드백**
6 교시 15:00	피드백 반영 + RAG 색인 워크플로우 , 청크 길이 분포 재기	팀 순회	**RAG 설계서 1~2 절**
7 교시 16:00	도구 노드 연결 (접수 · 티켓 · 알림) → 팀장이 VM 에 첫 배포	팀 순회	**VM 에서 WF-08 · 13 통과**
8 교시 17:00	산출물 발표 ① (팀별 10 분) · 데일리 기록	**산출물 발표 ① 진행 · 피드백 설계서 2 종 초안 · VM 동작**	**발표 피드백 · 데일리**

81 일차 · 11/06 (금) 구현과 예외 처리

공통 시나리오 통과 수를 늘리는 날 — 4 교시 산출물 중간점검 , 8 교시 산출물 발표 ② · 기능 동결

교시	팀이 하는 일	강사	완료 기준
1 교시 09:00	스탠드업 · 어제 피드백 반영 계획 , VM 에서 공통 시나리오 출발점 기록	**팀별 VM 상태 · 출발점 확인**	**통과 수 (출발점)**
2 교시 10:00	분류 · 검색 · 답변 Agent	팀 순회	**WF-01 통과**
3 교시 11:00	검증 Agent · 신뢰도 분기 · 담당자 이관 (티켓 생성)	팀 순회	**WF-03 통과**
4 교시 12:00	루프 · 병렬 · 복합 문의 , B · C 팀은 입구 · 뒷단 어댑터 팀장 : VM 배포 1 회	**산출물 중간점검 (팀별 10 분)** **페이로드 표 · RAG 3~5 절 · 연동 표**	**WF-02 · 11 · 12**
점심 13:00 ~ 14:00			
5 교시 14:00	보안 : 주입 차단 · 본인 건만 · 개인정보 가림	팀 순회	**WF-05 ~ 08**
6 교시 15:00	외부 장애 : 시간 제한 · 재시도 · DLQ · 알림 · 중복 방지	팀 순회	**WF-04 · 09 · 10**
7 교시 16:00	Error Workflow 게시 · 설계서 5~7 절 · 실행 결과서 1 절 시작 팀장 : VM 배포 2 회 + 전체 회귀	팀 순회	**설계서 2 종 완성**
8 교시 17:00	산출물 발표 ② (팀별 10 분) · 데일리 기록	**산출물 발표 ② → 기능 동결** **설계서 2 종 완성 · 결과서 초안**	**통과 수 · 동결 선언**

82 일차 · 11/09 (월) 시연 · 제출 · 발표

새 기능 없이 시연 · 문서 · 발표만 — 발표는 팀당 20 분 + 질의 5 분

교시	팀이 하는 일	강사	완료 기준
1 교시 09:00	데일리 점검 , VM 에서 시연 순서 한 번 돌려 보기	전체 시연 · 발표 상황 점검	팀별 남은 일 목록
2 교시 10:00	발표 자료 만들기 + 산출물 3 종 숫자를 runs/ 에서 맞추기	발표 자료 · 산출물 3 종 점검	발표 자료 초안
3 교시 11:00	1 · 2 팀 : VM 에서 시연 리허설 3 · 4 팀 : 결과서 숫자 · LLM 3 회 반복	1 · 2 팀 시연 발표 상황 확인	리허설 통과
4 교시 12:00	3 · 4 팀 : VM 에서 시연 리허설 1 · 2 팀 : 결과서 · 실패 사례 정리	3 · 4 팀 시연 발표 상황 확인	리허설 통과
점심 13:00 ~ 14:00			
5 교시 14:00	산출물 3 종 마무리 · tscan 비밀 검사 · 저장소 제출	산출물 3 종 제출 확인	마지막 커밋 (14:50)
6 교시 15:00	1 · 2 팀 최종 발표 (산출물 3 종 + 시연 , 20 분 + 질의 5 분)	발표 진행 · 질의	동료 평가지
7 교시 16:00	3 · 4 팀 최종 발표 (산출물 3 종 + 시연 , 20 분 + 질의 5 분)	발표 진행 · 질의	동료 평가지
8 교시 17:00	평가 시험	시험 진행 · 총평	답안 제출