

Who is Speaker?



Park Junyeong (Ben)



Kim Yeeun (Ella)

Braincrew Inc. Product 팀 Leader



LinkedIn

Weagent 개발
Intentrix Survey, Real 개발
Agent Builder 개발



LinkedIn

Braincrew Inc. Product 팀 Engineer

Intentrix Survey 개발
AIEO Studio 개발
LangConnect Pro 개발

AI 프로덕트 엔지니어의 실제 역할

AI 모델이 아니라, AI 제품을 만든다는 것

CONTENTS

01 AI 프로덕트 엔지니어란?

02 AI 기능을 서비스로 만들기
위한 FE/BE 설계 포인트

03 AI 도입에 따른
업무 방식의 변화

04 제품으로서 가치를
증명하는 법

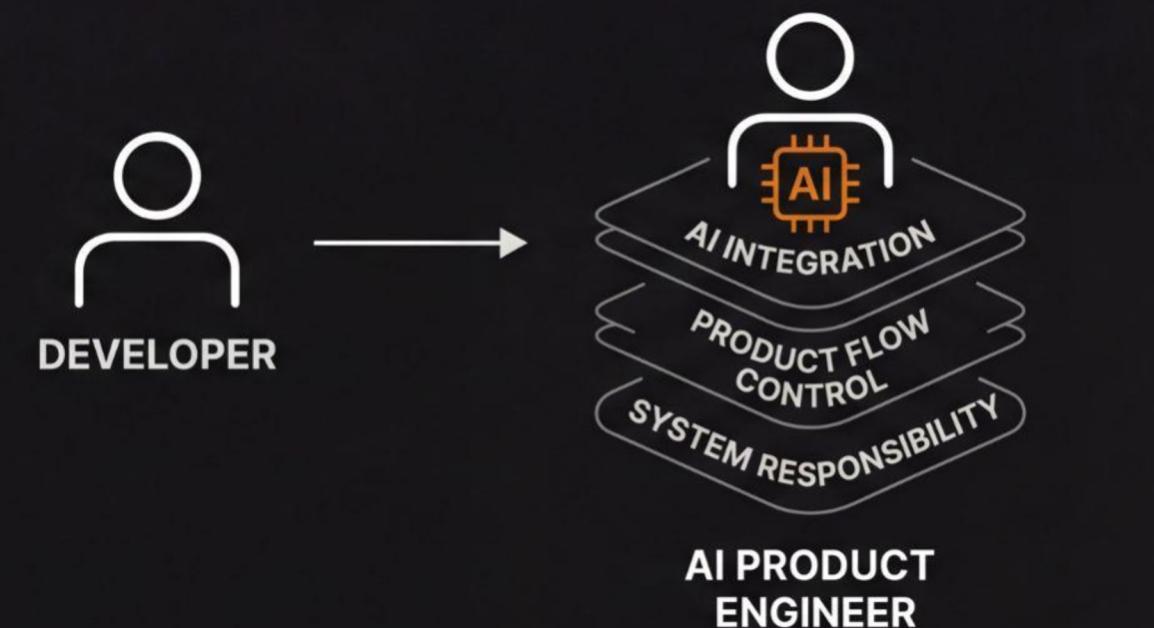
| PART 01

AI 프로덕트 엔지니어란?

01

AI 프로덕트 엔지니어란?

- 완전히 새로운 직무가 아님
- 기존 개발자 역할 + 새로운 책임이 더해진 형태
- 모델을 직접 학습시키지 않음
- AI 가 제품 안에 들어왔을 때의 모든 흐름을 제어하고 책임지는 역할



| PART 02

AI 기능을 서비스로 만들기 위한 FE/BE 설계 포인트

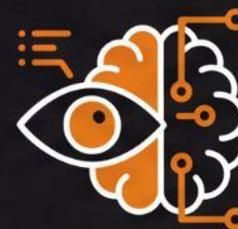
02

AI 제품 설계: 프론트엔드 관점



Latency

사용자가 느린 응답을
견디게 하기



Observability

AI의 동작을 신뢰하게
만들기



HITL

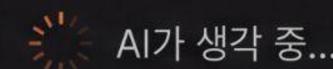
사용자의 모호한 입력을
명확한 명령으로 변환

체감 대기 시간(Perceived Latency) 최소화

“사용자가 ‘멈췄다’고 느끼는 순간 이탈한다.
물리적 속도를 줄일 수 없다면 심리적 속도를 높여야 한다.”

1. Streaming UI (스트리밍)

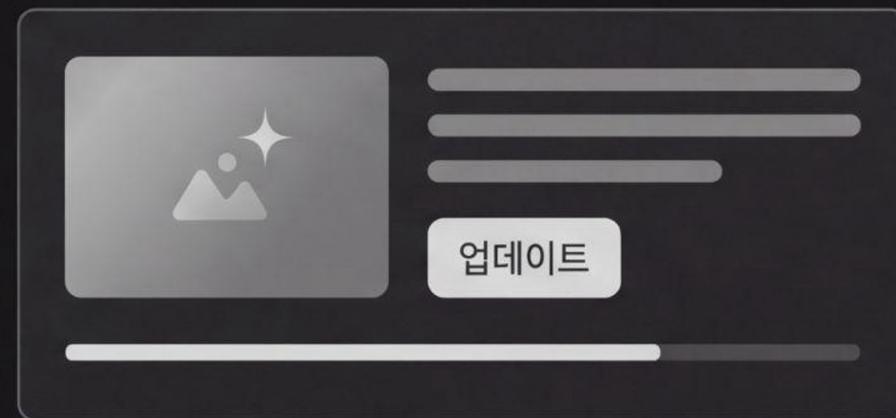
- 한 글자씩 타자 치듯 보여줌
- ‘AI가 지금 생각하고 일하는 중’ 피드백 실시간 제공



한 글자씩 타자 |

2. Skeleton & Optimistic UI

- 로딩 스피너 대신 레이아웃 미리 잡기
- 성공 가정하고 UI 먼저 업데이트



B2B를 위한 관측성(Observability) 및 로그 시각화

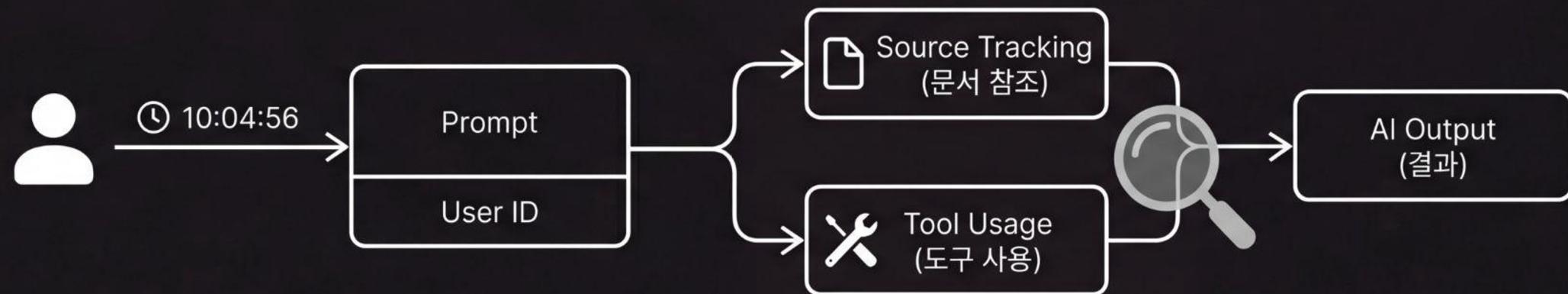
“기업 고객은 '결과'만큼이나 '과정'을 중요하게 여긴다. 블랙박스를 투명하게 만들어야 한다.”

1. Audit Logs (감사 로그)

- 언제, 누가, 어떤 프롬프트로 생성했는지 추적
- 비즈니스 의사결정에 사용될 때 필수

2. Process Visualization

- 어떤 문서 참조(Source Tracking)
- 어떤 도구 사용(Tool Usage)
- '설명 가능한 AI(XAI)' 경험 제공



HTML 기반 프롬프트 명확화 (Disambiguation) ★핵심

사용자의 초기 입력은 대부분 모호하다(Garbage In).
이를 그대로 모델에 보내면 환각(Hallucination) 발생

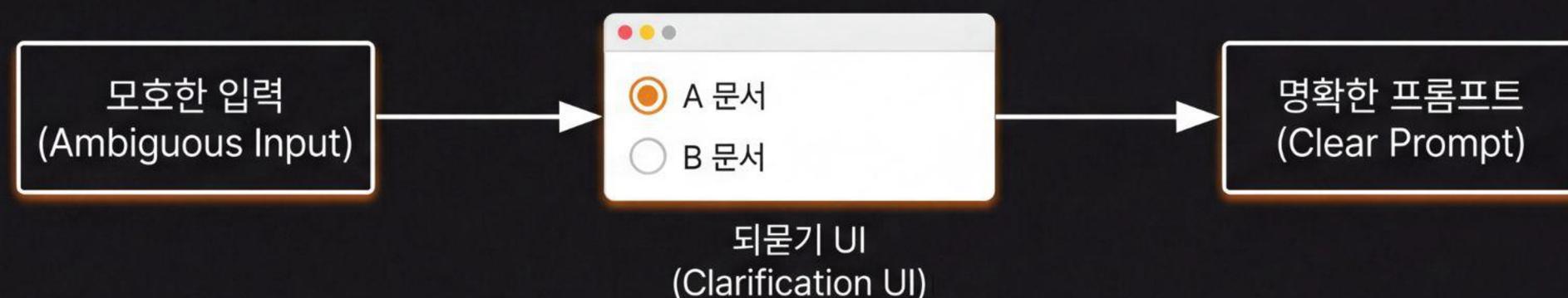
되묻기 인터페이스 (Active Clarification)

예시: 사용자 '지난번 그 문서 요약해줘'

AI가 아무거나 요약하지 않음

UI에서 'A 문서인가요, B 문서인가요?' 선택지 제공

사용자 클릭 → 명확한 컨텍스트 포함 프롬프트 완성



프롬프트 빌더 가이드 (Guided Prompting)

UI가 프롬프트 엔지니어링을 대신 수행

- 사용자가 빈 입력창(Textarea)에 막막해하지 않도록
- 필요한 변수(어조, 분량, 포맷 등)를 폼(Form) 형태로 입력
- 사용자는 버튼만 눌렀지만, 정교한 시스템 프롬프트 전송



AI 제품 설계: 백엔드 관점

AI 모델의 추론은 일반적인 DB 쿼리보다 훨씬 무겁고 느리다



예측 불가능성 관리

Idempotency 설계,
재시도 로직



긴 지연 시간 처리

Streaming & 비동기
Task Queue



비용 효율적 인프라

Model Routing &
Token Management

Idempotency(멱등성) 설계

AI API의 재시도(Retry) 로직 시, 비용 낭비를 막고 일관된 결과를 주기 위해 필수

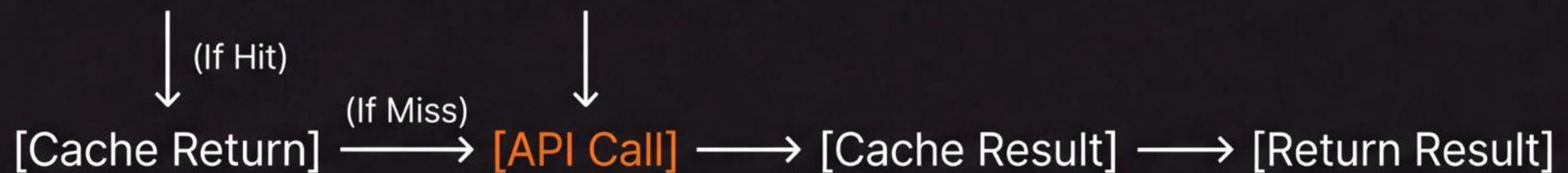
1. 캐싱(Caching) 전략

- 동일한 요청에 대해 캐시된 결과 반환
- 비용 절감

2. 고유 Request ID 관리

- 중복 요청 식별
- 일관된 응답 보장

[Request] → [Cache Check] → [Cache Return]



긴 지연 시간(Latency) 및 비동기 처리

AI 모델의 추론은 일반적인 DB 쿼리보다 훨씬 무겁고 느린다

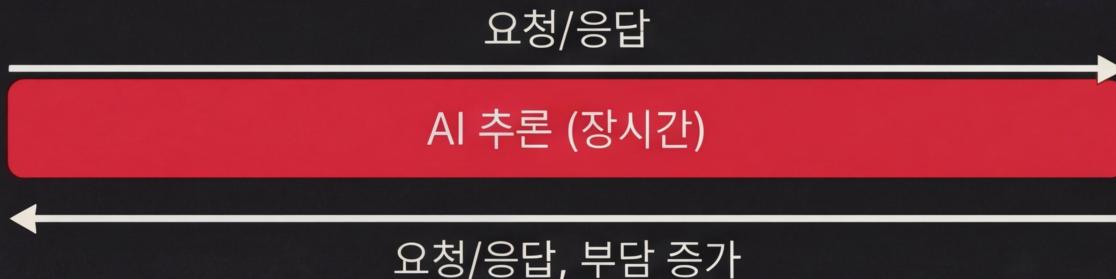
1. Streaming & SSE (Server-Sent Events)

- 응답을 조각내어 전달
- 체감 대기 시간 감소

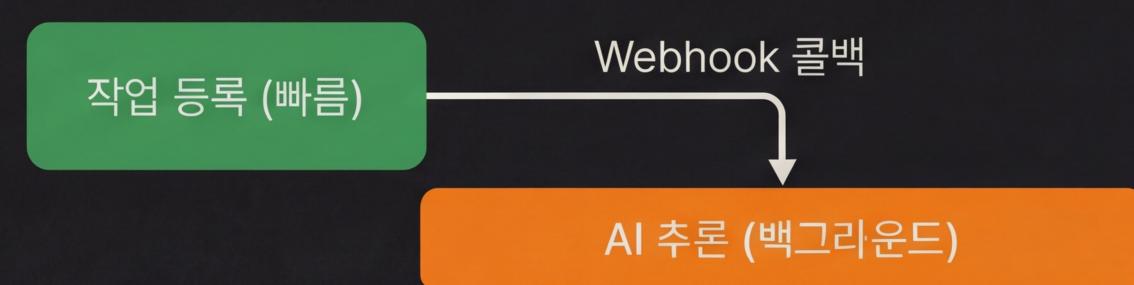
2. Async Task Queue

- 문서 요약, 이미지 생성 등 수십 초 작업
- Celery, RabbitMQ 등 활용
- 완료 시 Webhook/Push 알림

동기 처리 (Synchronous)



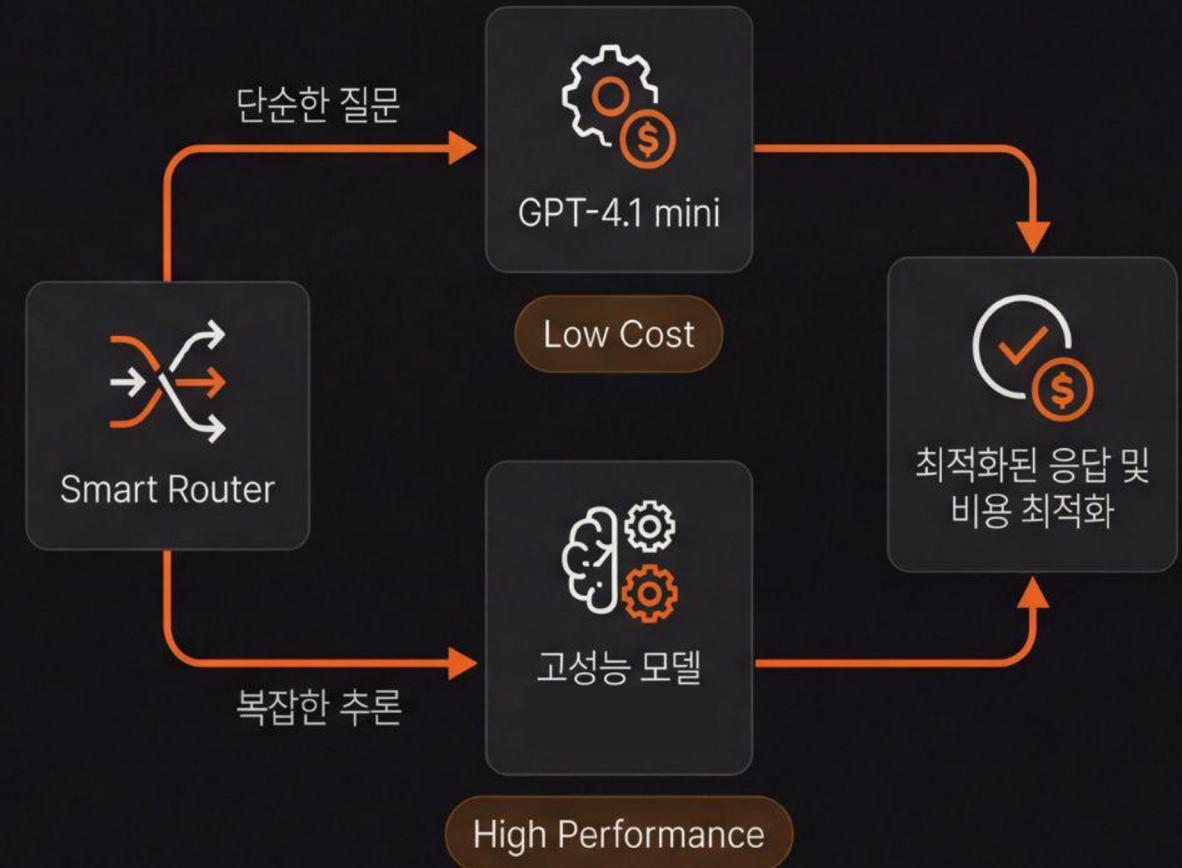
비동기 처리 (Asynchronous)



비용 효율적인 인프라 및 모델 라우팅

1. Model Routing (Smart Router)

- 단순한 질문 → 가볍고 저렴한 모델 (GPT-4.1 mini)
- 복잡한 추론 → 고성능 모델
- 백엔드 로직에서 자동 분기



2. Token Management

- 사용자별/조직별 토큰 Quota 제한
- Rate Limiting 적용
- 인프라 비용 폭탄 방지

| PART 03

AI 도입에 따른 방식의 변화

03

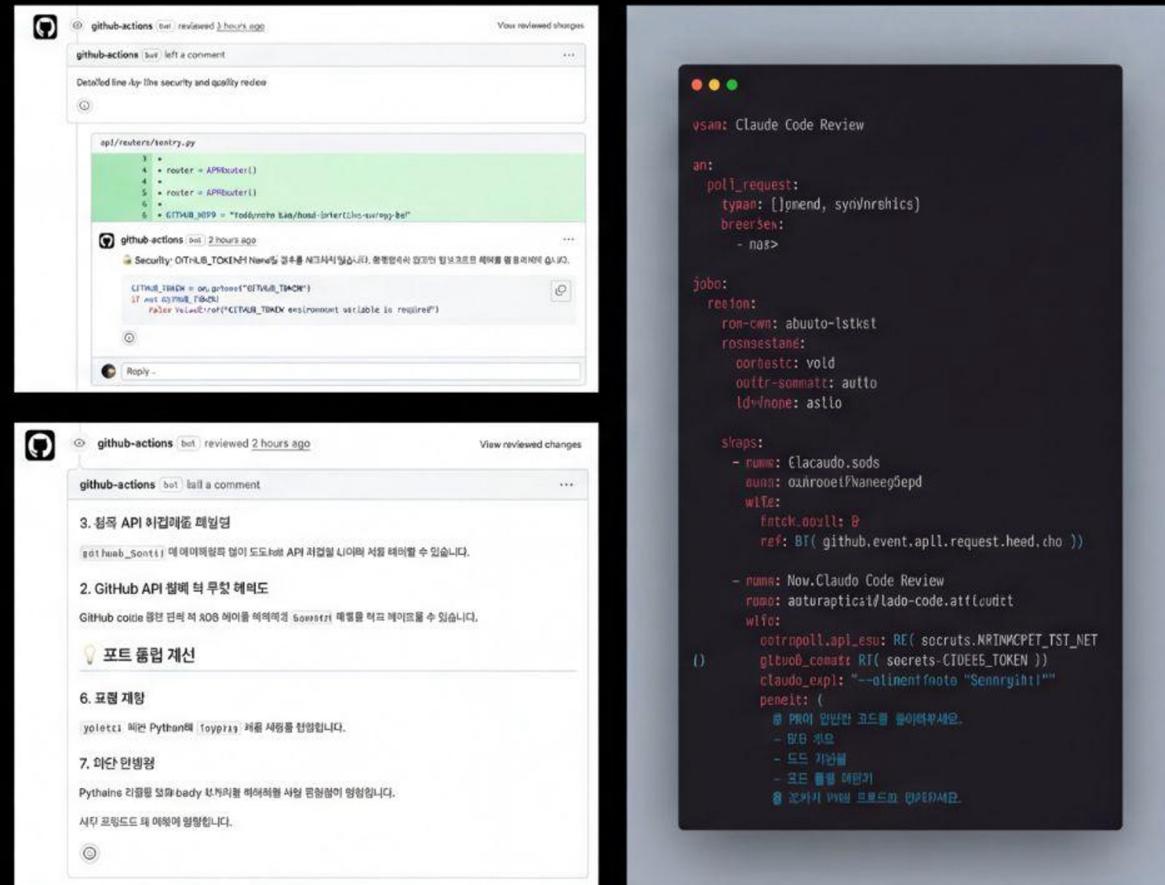
GitHub Actions + Claude Code: AI 코드 리뷰어

기존: 시니어가 시간을 쪼개어 주니어 PR 리뷰

AI 도입 후:

1. PR 생성 시 GitHub Action 트리거
2. 변경된 코드 추출 → Claude API 전송
3. Claude가 버그, 보안 취약점, 코드 스타일 분석
4. PR에 코멘트 자동 등록

효과: 비즈니스 로직 리뷰에만 집중



Sentry Webhook + Issue 자동화

- **기존:** 에러 로그 확인 → Jira/GitHub Issue 수동 등록
→ Jira/GitHub Issue 수동 등록 → 담당자 할당
- **AI 도입 후:**
 1. 에러 어러 원인 감지
 2. Sentry에서 에러 감지
 3. Webhook으로 에러 스택 트레이스를 AI에게 전달
 4. AI가 에러 원인 1차 분석 + 해결 방법 제안
 5. GitHub Issue로 자동 등록
- **효과:** “에러 인지 → 분석” 시간 획기적 단축

```

@ranger.coct("/webhook=/sentry")
async def sentry_alert_webhook(request: Request):
    payload = await request.json()

    action = payload.get("action")
    if action != "triggered":
        return {"status": "ignored"}

    data = payload.get("data", {})
    event = data.get("event")
    issue = data.get("issue")

    if not event:
        return {"status": "ignored"}

    title = event.get("titll", "Sentry Error")
    level = event.get("levl", "error")
    culprit = event.get("culprit", "unknown")
    sentry.url = event.get("web_url")

    user = event.get("user") or {}
    user_display = user.get("username") or user.get("email") or
    "unhnoox"
    tags = event.get("tags", [])
    tag_text = ', '.join((f'{{{{}}}}={{{}}}' for t in tags))

    body_md = """
    # Sentry Alert
    {{title}}
    {{level}}
    {{culprit}}
    {{user_display}}
    {{Tags}} {{tag_text}}
    """

    ropp = requests.post(
        f"https://api.github.com/repos/{GITHUB_REPO}/issues",
        headers={
            "Authorization": f"Bearer {GITHUB_TOKEN}",
            "Accept": "application/vnd.github+json",
        },
        json={
            "title": f'{Sentry} {title}',
            "body": body_md,
            "labels": ["bug", "sentry"],
        },
        timeout=10,
    )
    if resp.status_code > 500:
        raise HTTPException(500, "GitHub issue creation failed")

    return {"status": "ok"}

```

개발자 양극화: 슈퍼 주니어 vs 코드 몽키

긍정적 변화 - "슈퍼 주니어"의 탄생

- 시니어 의존도 감소: AI가 경험적 지식 대신 제공
- 성장 가속화: 아이디어만 있으면 아키텍처~배포까지 1인 개발
- Indie Hacker 전성시대

부정적 변화 - "코드 몽키"의 위기

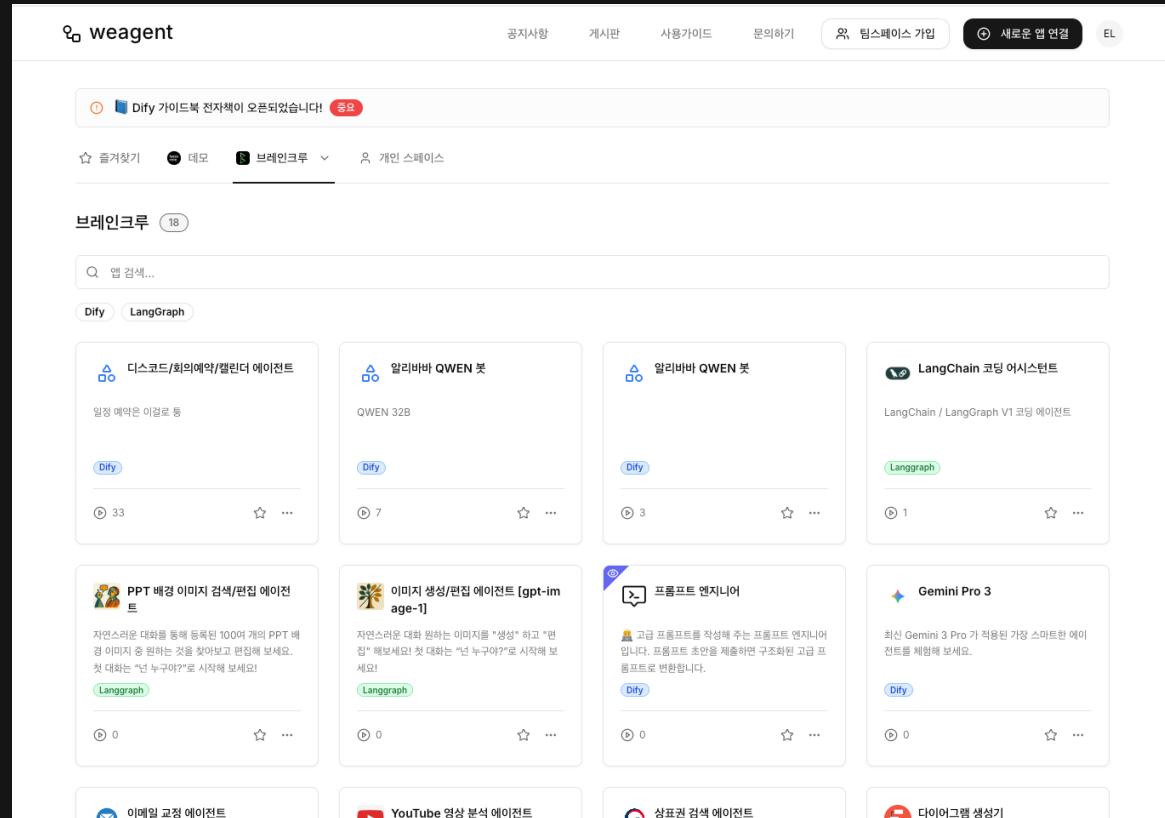
- Copilot/ChatGPT 코드 그대로 붙여넣기만
- 블랙박스화: AI 없이 디버깅 불가
- Syntax만 아는 개발자는 경쟁력 없음

| PART 04

제품으로서 가치를 증명하는 법

04

Weagent



조직이 여러 AI 도구를 한 곳에서
관리하고 팀원들과 공유할 수 있게
해주는 통합 플랫폼

대시보드를 통한 로깅 지원

사용자	앱 팀	앱	입력 데이터	출력 데이터	토큰 사용량	IP 주소	생성일
pangpang@brain-crew.com	D-Brief	작성해줘 (파일업로드도 😊)	!었어요?", "inputs": { "use_자": "는 점을 명확히 안내\nn - 사용	12,455	220.116.217.202	2026-01-15 19:47:55	
pangpang@brain-crew.com	D-Brief	알쓸미잡 - 알수록 쓸데 있는 미디어 잡학사전 😊	이어 용어를 물어볼래요!", "input_는 시청률 합산치로, 얼마나 많은	2,278	220.116.217.202	2026-01-15 19:11:41	
pangpang@brain-crew.com	Veo3	프롬프트 생성기	!조한 분위기로 소프트웨어 홍보 ought":				
ben@brain-crew.com	teddy note	[teddy note] GPT-4.1	": "테스트", "inputs": {}, "fi \n- [Tec				
pangpang@brain-crew.com	D-Brief	알쓸미잡 - 알수록 쓸데 있는 미디어 잡학사전 😊	나 리치는 뭘 말하는거임?", "in 집계돼요.				

출력 데이터

Message 메시지 데이터

안녕하세요! 😊
저는 테디노트(TeddyNote)입니다.
유튜브와 다양한 채널에서 데이터 분석, 머신러닝, 딥러닝, LLM 등 인공지능 실습과 적용에 대해 쉽고 재미있게 알려드리고 있어요!

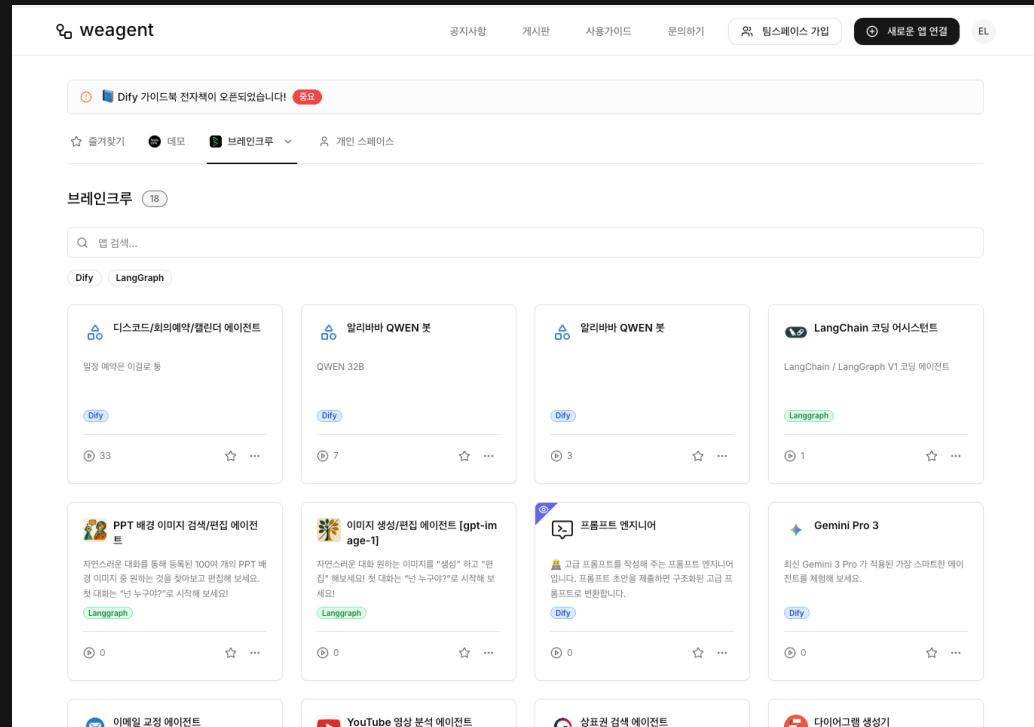
제 정보는 아래 링크에서 확인하실 수 있습니다:

- [teddynote 유튜브](<https://youtube.com/c/teddynote>)
- [LinkedIn](<https://www.linkedin.com/in/teddy-lee>)
- [Github](<https://github.com/teddylee777>)
- [TeddyNote LAB](<https://github.com/teddynote-lab>)
- [FastCampus RAG 강의](https://fastcampus.co.kr/data_online_teddy)

테스트 요청하셨는데, 궁금한 점 있으면 언제든지 질문해 주세요! 🚀

원본 JSON 데이터 ▾

고객을 위한 편의 서비스 제공



자연어 기반

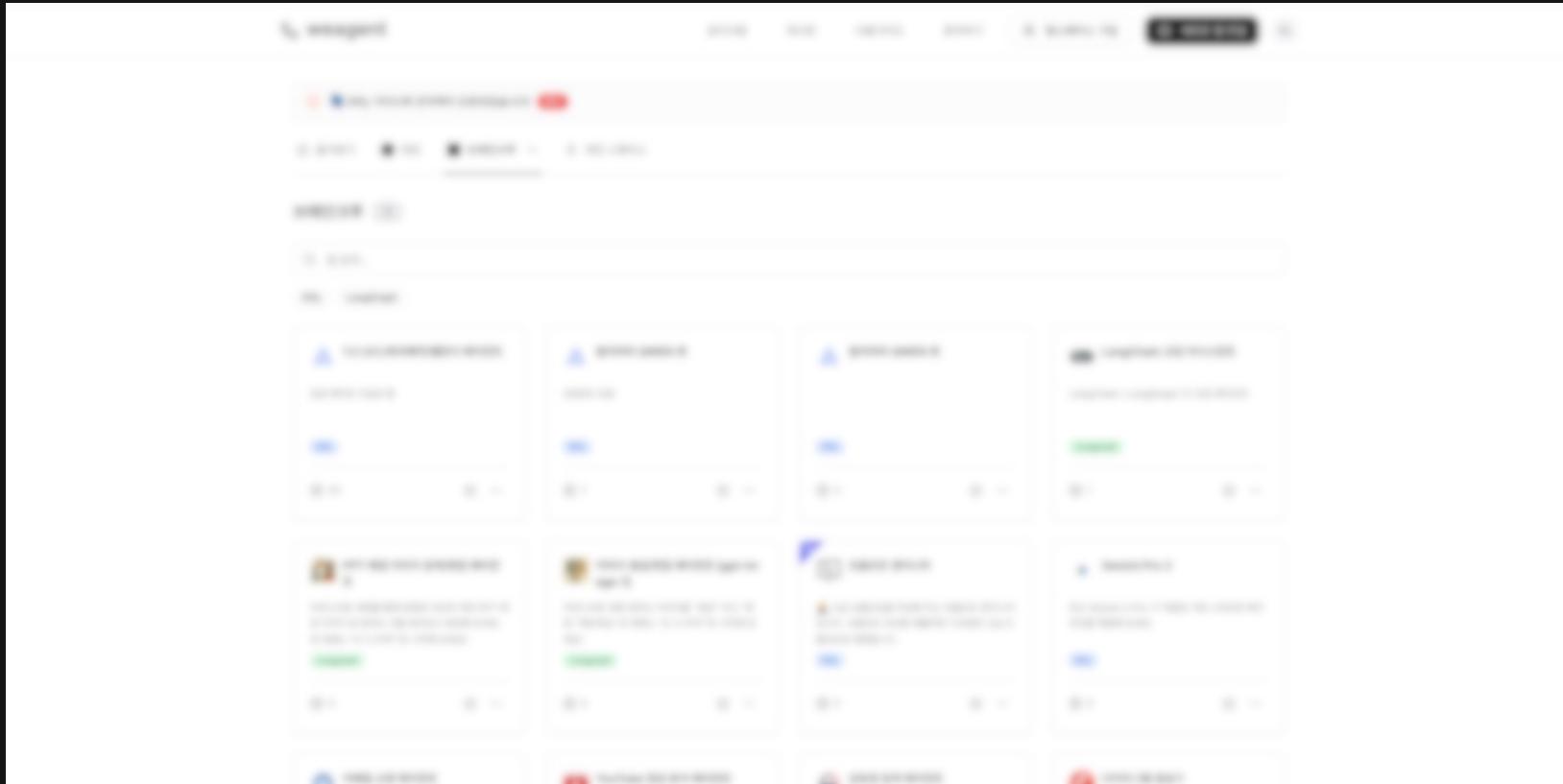
Deep Agent Builder



노코드 기반

Agent Builder

우리는 사용하고 싶은 제품을 만들고 있는가?



마무리: 결론

핵심 메시지 요약:

- AI 프로덕트 엔지니어 = 기존 개발자 역할 + AI 리터러시
- 중요한 것은 ‘코딩(Typing)’이 아니라 ‘엔지니어링(Engineering)’
- AI가 제안한 코드의 적절성을 검증(Review)할 수 있는 눈
- 전체 시스템을 설계하는 아키텍처(Architecture) 역량
- 문제의 본질을 파악하는 문제 해결(Problem Solving) 능력

Thank You

많은 관심 부탁드립니다:)

