W03
로드맵 설계하기

안전하게 완성하는 전략

내가 만든 자료는 충분히 완벽한가?

- “정답을 알려줘~”: 안좋은 결과를 가져온다
- AI는 내가 무엇을 원하는지에 큰 관심이 없다
- 제일 좋은 방법은 스스로 계속 문제를 돌아보게 하는 것

만약 서로 다른 두 에이전트가 있다면?

- 기존의 자료를 모두 입력하고 이렇게 진행하는 것이 최선인지 검토하게 한다.
- 검토 후 피드백한 내용을 다시 원 에이전트에게 검토하게 한다.
- 검토한 내용 중 받아들이는 부분을 개선하게 한다.

질문 a. 의사결정 검토 - 개선하기

“이 자료들이 목적을 달성하는데 최선인지 분석해보고, 개선할 점들에 대한 피드백을 작성해줘"

“이 자료에 대해 이런 피드백이 제안되었어. 검토해보고 동의하는 부분에 대해서 기존 문서의 내용을 개선해줘"

- AI가 작성한 문서에 대해 이렇게 상호 검토 과정을 거치면 완성도가 더욱 올라간다.
- 서로 다른 두 종류의 에이전트를 활용하면 높은 피드백 효고를 낼 수 있다.
- 하나의 에이전트를 사용하더라도 히스토리를 공유하지 않는 채팅을 사용하면 충분히 검토가 가능하다.

지금 그대로 개발을 맡기면 어떻게 될까?

AI에게 "이거 만들어줘"라고 하면?

- 어디서부터 시작할지 모른다
- 한번에 너무 많은 것을 만들려고 한다
- 만들다가 앞서 만든 것과 충돌이 발생한다
- 뭐가 완성된 건지 확인할 방법이 없다

"AI는 전체를 한번에 완벽하게 만드는 능력이 없다"

모든것을 다 정해주면 어떻게 될까?

함수 하나하나, 변수 이름까지 다 정해주면?

- 문서 작성이 개발보다 오래 걸린다
- AI가 더 나은 방법을 제안할 기회를 뺏는다
- 기술적 변경이 생기면 문서 전체를 다시 써야 한다
- 정작 중요한 '왜 이렇게 만드는가'는 빠지게 된다

"너무 구체적이면 오히려 유연성을 잃는다"

실수없는 개발을 하기 위해서

AI 에이전트 개발에서 실수가 발생하는 이유

- 맥락을 잃어버린다: 대화가 길어지면 처음 의도를 잊는다
- 순서가 꼬인다: 먼저 만들어야 할 것을 나중에 만든다
- 완료 기준이 없다: 뭐가 끝난 건지 판단할 수 없다
- 검증이 안 된다: 제대로 동작하는지 확인할 방법이 없다

해결책: 단계를 나누고, 각 단계마다 검증한다

"한번에 다 만들지 말고, 검증 가능한 단위로 쪼개서 만든다"

프로젝트 관리 기법의 활용

소프트웨어 개발은 오래전부터 '어떻게 하면 실수 없이 만들 수 있을까'를 고민해왔다

WBS (Work Breakdown Structure)

- 큰 프로젝트를 작은 작업 단위로 분해하는 방법
- 각 단위는 명확한 결과물과 완료 기준을 가진다

이것을 AI 에이전트 개발에 적용하면?

- AI가 한번에 처리할 수 있는 크기로 분해
- 각 단계마다 검증 가능한 기준 설정
- 순서와 의존성을 명확히 정의

"검증된 프로젝트 관리 기법을 AI 개발에 적용한다"

로드맵이란?

로드맵: 목적지까지 가는 전체 경로를 보여주는 지도

- 어디서 시작해서 어디로 가는가 (시작점과 목표)
- 어떤 순서로 진행하는가 (단계와 흐름)
- 각 단계에서 무엇을 완성하는가 (산출물)
- 언제쯤 도착하는가 (일정 감각)

AI 에이전트 개발에서 로드맵의 역할

- AI에게 "지금 여기를 만들어"라고 정확히 지시할 수 있다
- 어디까지 완성됐는지 추적할 수 있다
- 문제가 생겼을 때 어디서 발생했는지 파악할 수 있다

"로드맵이 있으면 길을 잃지 않는다"

마일스톤

마일스톤(Milestone): 프로젝트의 주요 이정표

- "여기까지 오면 의미 있는 무언가가 완성된다"
- 검증 가능한 결과물이 있다
- 다음 단계로 넘어가도 되는지 판단할 수 있다

MVP 기준 일반적인 규모: 3~5개

- 마일스톤 1: 사용자 인증 시스템 완성
- 마일스톤 2: 핵심 기능 동작
- 마일스톤 3: MVP 출시 가능 상태

"마일스톤은 '여기서 한번 확인하자'라는 체크포인트"

질문 1. 마일스톤 로드맵 작성하기

“당신은 세계 최고 수준의 소프트웨어 아키텍트이자 PM입니다. /XXX/XXX 폴더의 주어진 자료를 분석하여 MVP 개발 프로젝트를 완수하기 위한 최상위 전략인 'Milestones'를 수립합니다.”

- 프로젝트를 관리 가능한 큰 덩어리(Milestone)로 나눕니다.
- 각 Milestone은 명확한 목표와 산출물을 가져야 합니다.
- MVP 구축에 제일 효과적인 방식으로 논리적 그룹으로 나눕니다.
- 의존성을 고려하여 Milestone 간의 순서를 명확하게 정합니다.

질문 b. 의사결정 이해하기

“개발에 대해 OOO정도의 이해수준을 가진 사람에게 지금과 같이 로드맵의 각 단계를 수립하게 된 내용에 대해, 이 프로젝트를 끝까지 이어나가는데 문제가 없는 수준으로 친절하게 설명합니다.”

- 선택 1. 이렇게 진행하는 것이 왜 목표 달성에 제일 최적인지에 대해 설명합니다.
- 선택 2. 이렇게 결정하게 된 기술적, 맥락적 이유와 배경에 대해 설명합니다.
- 선택 3. 이렇게 진행하는 것의 장점과 단점, 기회와 위험에 대해 설명합니다.

이후 모르는 단어와 개념들에 대해 추가질문을 통해 해당 의사결정을 완벽하게 이해합니다.

다른 의사결정들에도 이런 과정을 모두 적용합니다.

기능 단위로 개발하면 생기는 문제

서비스를 만드는 과정 != 기능을 만드는 과정

- "로그인 만들고 → 회원가입 만들고 → 대시보드 만들고"로 진행하면?
- DB 스키마 설계 → 기능이 아니다
- 프로젝트 초기 세팅 → 기능이 아니다
- 인프라/배포 환경 구축 → 기능이 아니다

기능만 보면 이런 작업들이 빠진다

- 나중에 "아, 이거 먼저 했어야 했네" 하고 돌아가야 함
- "기능이 아닌 작업도 포함할 수 있는 단위가 필요하다"

워크 패키지

워크 패키지(Work Package): 마일스톤을 달성하기 위한 작업 묶음

워크 패키지는 중립적인 개념

- 사용자 기능일 수도 있고
- 기술적 작업일 수도 있고
- 문서화 작업일 수도 있다

마일스톤당 일반적인 규모: 3~7개

"워크 패키지는 '이것들을 만들면 마일스톤이 완성된다'의 목록"

질문 2. 워크 패키지 로드맵 작성하기

“당신은 세계 최고 수준의 테크 리드입니다. 주어진 Milestone을 달성하기 위해 필요한 구체적인 단계인 'Work Package’로 분해하십시오.”

- Milestone의 목표를 실행 가능한 단위로 구체화합니다.
- Vertical Slicing: 가능한 한 각 Workpack이 사용자 가치를 전달하도록 구성하십시오.
- 완결성: 각 Workpack이 끝나면 소프트웨어는 '동작하는' 상태여야 합니다.

시나리오

시나리오(Scenario): 하나의 행동과 결과를 검증하는 단위

BDD(Behavior-Driven Development)에서 온 개념

- Given: 이런 상태에서
- When: 이런 행동을 하면
- Then: 이런 결과가 나온다

왜 시나리오 단위로 나누는가?

- AI 에이전트가 "이것만 만들어"라고 지시받을 수 있는 크기
- 만들고 나서 "됐는지 안 됐는지" 바로 확인 가능
- 실패해도 어디서 실패했는지 명확함

시나리오

크기 판단 기준

- "로그인 기능 구현" → 너무 크다, 분해 필요
- "올바른 credential로 로그인 시 토큰 반환" → 적정
- "이메일 정규식 체크 함수" → 너무 작다, AI가 알아서 처리

"시나리오는 AI에게 지시하고 검증할 수 있는 최소 단위"

질문 3. 시나리오 로드맵 작성하기

당신은 세계 최고 수준의 시니어 개발자입니다. Work Package를 Scenario로 분해하십시오.

- 하나의 Scenario = 하나의 행동 = 하나의 검증
- 성공 케이스와 실패 케이스를 별도 Scenario로 분리
- 선행 의존성이 있으면 명시
- 출력형식: Scenario [WP번호]-[순번]: [행동 기반 명칭]
    - Given: [사전 상태/조건]
    - When: [실행할 행동]
    - Then: [기대 결과]
    - 선행 Scenario: [의존성 있으면 번호, 없으면 "없음"]

질문 3. 시나리오 로드맵 작성하기

예시: Work Package: 사용자 로그인

- Scenario 1-1: 올바른 credential로 로그인 시 토큰 반환
  - Given: 등록된 사용자 존재
  - When: 올바른 이메일/비밀번호로 로그인 요청
  - Then: JWT 토큰 반환, 200 OK
  - 선행: 없음
- Scenario 1-2: 잘못된 비밀번호로 로그인 시 실패
  - Given: 등록된 사용자 존재
  - When: 잘못된 비밀번호로 로그인 요청
  - Then: 401 Unauthorized, 에러 메시지 반환
  - 선행: 없음

모든 로드맵을 한번에 생성하지 말자

왜 전체를 한번에 만들면 안 되는가?

- 마일스톤 5개 × 워크 패키지 5개 × 시나리오 10개 = 250개 시나리오
- 한번에 만들면 앞부분 만들 때 결정한 것이 뒷부분과 충돌
- 실제 개발하면서 계획이 바뀌는데, 이미 만든 로드맵 전체를 수정해야 함

권장하는 방식

- 마일스톤 전체를 먼저 수립 (큰 그림)
- 첫 번째 마일스톤만 워크 패키지로 분해
- 첫 번째 워크 패키지만 시나리오로 분해
- 개발 완료 후 다음 단계 분해

"한 단계 앞만 구체화하고, 나머지는 큰 그림만 유지한다"

오늘 해결해야 할 문제

1. 앞서 작성한 서비스 개요, PRD, 기술 스택, 시스템 구조를 바탕으로 마일스톤 로드맵을 작성합니다. 각 마일스톤이 왜 이 순서로 배치되었는지 이해합니다.
2. 첫번째 마일스톤을 워크 패키지로 분해합니다. 기능뿐 아니라 기술적 작업(DB 설계, 환경 구축 등)도 빠짐없이 포함되었는지 확인합니다.
3. 하나의 워크 패키지를 선택해 시나리오로 분해해 봅니다. Given-When-Then 형식으로 작성하고, 이것이 검증 가능한 단위인지 스스로 판단해 봅니다.
4. 로컬에 프로젝트 폴더를 하나 생성하고 /<project>/docs/roadmap 폴더를 생성해 생성한 첫 로드맵 파일들을 해당 폴더에 계층적으로 저장해 놓습니다.
5. (선택) 해당 프로젝트 폴더 전체를 내 github repo에 커밋합니다.

Q&A