AIO 1기 · FINAL PROJECT · GUIDE

WBS 잘 쓰는 법

해야 할 일을 빠짐없이 쪼개고, 끝까지 관리하기

쪼개기 → 추정하기 → 배치하기 → 관리하기

WHAT IS WBS

WBS는 해야 할 일 전체의 지도입니다

W

Work

프로젝트가 만들어 내야 할 결과물과 그에 필요한 일

B

Breakdown

관리할 수 있는 크기가 될 때까지 쪼갠다

S

Structure

위에서 아래로 이어지는 계층 구조로 정리한다

WBS는 먼저 **“무엇을”**을 정합니다. 일정과 담당자는 그 위에 얹는 정보입니다.

WHY WBS

WBS가 있으면 달라지는 네 가지

빠진 일이 보인다

테스트, 문서, 발표 준비처럼 잊기 쉬운 일을 미리 찾는다

진행이 보인다

“거의 다 했어요” 대신 몇 개 중 몇 개가 끝났는지로 말한다

공정하게 나눈다

누가 무엇을 맡았는지 분명해지고 개인 기여가 기록된다

기간을 추정할 수 있다

작게 쪼갠 일은 얼마나 걸릴지 예측하기 쉽다

WBS VS OTHERS

할 일 목록, 간트 차트와 무엇이 다른가

도구	답하는 질문	한계
할 일 목록	지금 뭘 해야 하지?	전체 범위와 구조가 안 보인다
WBS	**무엇을 다 해야 끝나지?**	날짜와 순서는 따로 얹어야 한다
간트 차트	언제, 어떤 순서로 하지?	WBS 없이 그리면 일이 빠진다
칸반 보드	지금 어디까지 왔지?	전체 계획 대비 진척은 약하다

순서는 **WBS로 범위 확정 → 간트로 일정 배치 → 칸반으로 매일 진행 관리**입니다.

RULE #1

100% 규칙 — 빠짐없이, 겹침없이

하위 항목을 모두 더하면 정확히 상위 항목 전체가 되어야 합니다.

- 빠진 일이 있으면 → 계획에 없는 일이 나중에 터진다
- 겹치는 일이 있으면 → 두 사람이 같은 일을 하거나 아무도 안 한다
- 범위 밖의 일이 있으면 → 그 일은 WBS에서 지운다

2. 백엔드 API = 100%

2.1 API 설계	2.2 DB 스키마
2.3 인증 기능	2.4 대화 API

네 개를 다 끝내면 2번이 끝난다

RULE #2

위는 결과물(명사), 아래는 작업(동사)

상위 레벨 = 결과물

끝났을 때 손에 쥐는 것

인증 기능
API 설계 문서
배포 환경

최하위 레벨 = 작업

그 결과물을 만들기 위해 하는 일

JWT 발급 API 구현
엔드포인트 목록 작성
Dockerfile 작성

"백엔드 공부하기"처럼 결과물이 없는 항목은 작업이 아닙니다. **무엇이 남는지**로 바꿔 쓰세요.

EXAMPLE · AI 상담 챗봇 서비스

계층 구조 예시

LEVEL 1 · AI 상담 챗봇 서비스

1 기획 | 2 백엔드 API | 3 AI 에이전트 | 4 화면 | 5 배포·테스트 | 6 프로젝트 관리

LEVEL 3 · 작업 패키지 (2번 아래)

- 2.1 API 설계 문서
- 2.2 DB 스키마
- **2.3 인증 기능**
- 2.4 대화 API

LEVEL 4 · 작업 (2.3 아래)

- 2.3.1 회원가입 API 구현
- 2.3.2 JWT 발급·검증 구현
- 2.3.3 인증 테스트 작성

HOW SMALL

얼마나 잘게 쪼갤까

1~3일

한 작업의 권장 크기 — 하루 단위로 진행을 확인할 수 있게

이 질문에 모두 “예”면 그만 쪼갭니다

- 한 사람이 맡을 수 있는가?
- 얼마나 걸릴지 감이 오는가?
- 끝났는지 누구나 확인할 수 있는가?
- 다른 사람에게 설명 없이 넘길 수 있는가?

너무 크면 — “백엔드 개발” 같은 항목은 몇 주째 진행 중

너무 작으면 — 관리하는 데 일하는 시간보다 더 든다

NAMING

작업 이름은 동사 + 대상 + 범위

모호한 이름	바로 착수할 수 있는 이름
로그인	이메일 로그인 API 구현 (JWT 발급 포함)
DB 작업	사용자·대화 테이블 ERD 작성
프롬프트 개선	상담 에이전트 시스템 프롬프트 v2 작성 및 10개 질문 테스트
배포	백엔드 Docker 이미지 빌드 스크립트 작성
회의	주간 회고 진행 및 회의록 공유 (매주 금)

이름만 보고 다른 팀원이 “무엇을, 어디까지” 할지 알 수 있어야 합니다.

DEFINITION OF DONE

“완료”의 뜻을 팀이 미리 합의합니다

팀 공통 완료 기준 (예시)

- ✓ 코드가 main 브랜치에 머지됨
- ✓ 다른 팀원 1명이 리뷰함
- ✓ 테스트를 통과함
- ✓ 관련 문서나 README가 갱신됨

작업별 완료 기준

2.3.2 JWT 발급·검증 구현

→ 로그인 시 토큰이 발급되고, 토큰 없는 요청은 401로 거부되며, 테스트 3개가 통과한다

완료 기준이 없으면 “90% 했어요”가 몇 주씩 이어집니다.

WBS FIELDS

WBS 표에 들어갈 칸

ID	작업	산출물	담당	예상	선행	기간	상태
2.2.1	사용자·대화 테이블 ERD 작성	ERD 이미지	A	1일	1.3	11/16	완료
2.3.1	회원가입 API 구현	PR	B	1.5일	2.2.1	11/17–18	진행
2.3.2	JWT 발급·검증 구현	PR	B	2일	2.3.1	11/18–19	할 일
2.3.3	인증 테스트 작성	테스트 코드	C	1일	2.3.2	11/20	할 일

필수 ID · 작업 · 담당 · 기간 · 상태

있으면 좋음 산출물 · 예상 공수 · 선행 작업 · 실제 소요

ESTIMATION

기간은 세 번 물어보고 정합니다

낙관 (O)
모든 게 잘 풀리면
1일

보통 (M)
평소대로라면
2일

비관 (P)
막히면
6일

예상 = (O + 4M + P) ÷ 6
(1 + 8 + 6) ÷ 6
2.5일

- 추정은 실제로 그 일을 할 사람이 합니다
- 처음 써 보는 기술은 비관값을 넉넉하게 잡습니다
- 팀 전체 일정에는 따로 버퍼를 둡니다 (개별 작업에 숨기지 않기)

DEPENDENCIES

순서와 의존성 — 가장 긴 길을 찾기



화면 디자인 (병렬 진행) ← 백엔드를 기다리지 않고 동시에 할 수 있는 일

핵심 경로

이어진 작업 중 가장 긴 줄. 여기가 늦으면 전체가 늦습니다

병렬 작업

모의 데이터·가짜 API로 의존성을 끊으면 동시에 진행할 수 있습니다

SCHEDULING

일정은 마감에서 거꾸로 얹습니다

1	**고정된 날짜부터 표시**	산출물 마감 · 발표일 · 휴일 · 수업 없는 날
2	**마감 앞에 검토 시간**	마감 하루 전은 팀 내부 리뷰와 수정
3	**핵심 경로부터 배치**	의존성 순서대로, 병렬 작업은 빈자리에
4	**마일스톤 정하기**	“무엇이 돌아가는가”로 정의 (예: 로그인 후 대화 가능)
5	**버퍼 남기기**	발표 직전 주는 새 기능 없이 안정화와 리허설

ASSIGNMENT

담당은 한 명, 부담은 고르게

담당 1명 + 검토 1명

“다 같이”는 아무도 책임지지 않는다는 뜻입니다

사람별 합계 확인

주별로 예상 공수를 더해 한 사람에게 몰리지 않게

강점과 성장을 섞기

잘하는 일 + 배우고 싶은 일을 함께 맡기

모두가 핵심 작업 하나씩

문서·관리만 맡는 사람이 없게 — 개인 기여의 근거

TOOLS

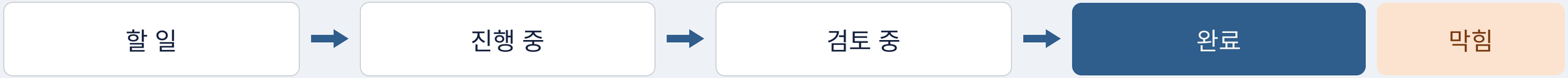
GitHub Projects로 WBS 운영하기

WBS 요소	GitHub Projects에서
작업	이슈 1개 = 작업 1개
WBS ID · 단계	라벨 또는 사용자 정의 필드
담당 · 상태	Assignee · Status 필드
기간	시작일 · 종료일 필드 → Roadmap 뷰
마일스톤	Milestone
완료 증거	이슈에 연결된 PR

같은 데이터를 **Table**(WBS) · **Roadmap**(일정) · **Board**(칸반) 뷰로 바꿔 볼 수 있습니다. 스프레드시트로 시작해도 괜찮습니다.

KEEP IT ALIVE

WBS는 한 번 쓰고 끝나는 문서가 아닙니다



매일

스탠드업에서 상태 갱신, 막힘은 바로 공유

매주

계획 대비 실제 비교, 다음 주 작업 잘게 쪼개기

바뀔 때

범위·일정 변경은 이유와 함께 기록

COMMON MISTAKES

흔한 실수와 처방

실수	처방
날짜부터 그린다	결과물로 먼저 쪼개고 일정은 나중에
“백엔드 개발” 같은 거대 작업	1~3일 단위로 다시 쪼개기
회의·문서·발표 준비가 없다	‘프로젝트 관리’ 묶음을 따로 만들기
담당자가 “전원”	담당 1명 + 검토 1명
테스트·버그 수정 시간이 없다	기능마다 테스트 작업 넣고 버퍼 따로 두기
처음 만들고 방치	매일 상태 갱신, 매주 계획 대비 실제 점검

SELF CHECK

WBS 셀프 체크리스트

- ✓ 하위를 다 끝내면 상위가 끝난다
- ✓ 겹치는 작업이 없다
- ✓ 작업이 1~3일 크기다
- ✓ 작업 이름만으로 할 일이 보인다
- ✓ 테스트·문서·발표 준비가 들어 있다
- ✓ 모든 작업에 담당자가 1명이다
- ✓ 한 사람에게 일이 몰리지 않았다
- ✓ 마감 앞에 검토 시간이 있다
- ✓ 핵심 경로를 알고 있다
- ✓ 버퍼가 따로 잡혀 있다

마지막 확인: 팀원 한 명이 갑자기 빠져도 누가 무엇을 이어받을지 보이는가?

잘 쪼갠 일은
반은 끝난 일입니다

빠짐없이 쪼개기 | 현실적으로 추정하기 | 거꾸로 배치하기 | 매일 갱신하기

Q & A