비개발자가 AI 코딩 도구로 체중 기록과 등교 옷차림 고민을 토스 미니앱으로 만들어 푼 기록

매일 되풀이되는 작은 불편을 앱 하나로 풀어 보고 싶은데, 코드를 몰라서 멈춰 있는 분께 드리는 기록입니다. AI 코딩 도구에 코드보다 계획서를 먼저 쓰게 하는 순서, 계획서에 꼭 넣어야 할 줄(개인 정보, 판정하지 않기, 데이터가 모자랄 때), 그리고 바로 옮겨 쓸 수 있는 단계표와 요청문을 담았어요.

블로그·SNS 채널 '수달빌더'(the-cyclopedia.com)를 운영하는 비개발자가 토스 앱인토스(Apps in Toss)에 생활 미니앱(Mini App)을 AI 코딩 도구로 만들어 올린 기록입니다. 앱인토스는 토스 앱 안에서 열리는 작은 앱을 만들어 올리는 플랫폼이에요. 이 글은 건강 기록과 아침 옷차림을 다루는 두 앱, 앱인토스 미니앱 - 체중 다이어리와 앱인토스 미니앱 - 날씨 코디를 중심으로 씁니다. 설명과 이미지는 앱 폴더의 계획서, 소스 코드, 점검 기록, 콘솔 등록용 화면에서 옮겼습니다.

① 어떤 상황에서 AI를 활용했나요?

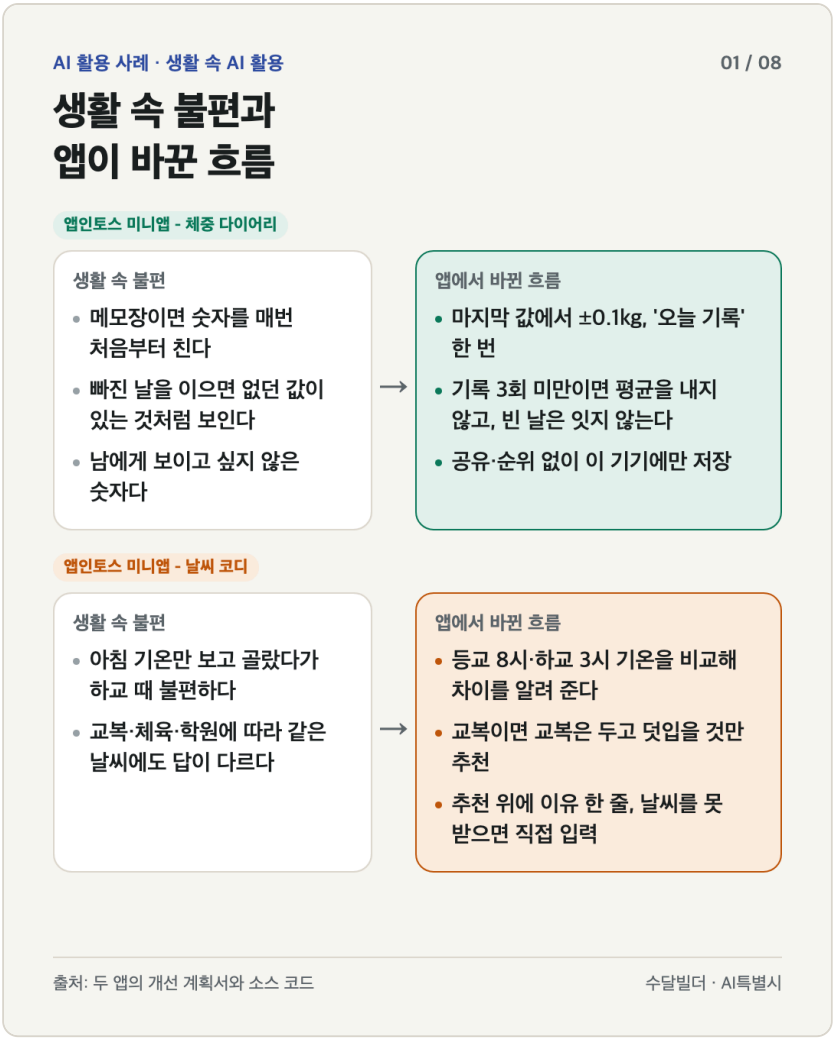
체중을 적는 일과 아침에 옷을 고르는 일에서 시작했어요

첫 번째는 체중 기록입니다. 매일 적어 두고 흐름을 보고 싶을 때 걸리는 점이 몇 가지 있어요. 메모장에 적는다면 숫자를 매번 처음부터 쳐야 하고, 빠진 날을 그래프에서 이어 버리면 그 사이에도 값이 있었던 것처럼 보이죠. 체중은 개인 데이터라서 공유나 순위를 붙이기도 조심스럽고요.

체중 다이어리는 원래 앱인토스에 올라가 있던 앱이고, 예전 설명에는 BMI, 목표까지 남은 kg, 다이어트 격려 문구가 함께 들어 있었어요. 개선 계획서는 방향을 이렇게 정했습니다. "하루 변동을 평가하거나 의료 판정으로 오해할 표현을 걷어내고, 기록 속도와 장기 흐름을 핵심 가치로 둔다."

두 번째는 아침 옷차림이에요. 날씨 코디 계획서는 상황을 이렇게 적었습니다. "중·고등학생은 아침 기온만 보고 옷을 골랐다가 하교 시간의 일교차, 비·바람·습도, 교실 냉방 때문에 불편해진다. 동시에 교복 여부, 체육 수업, 학원, 주말 약속처럼 같은 날씨에도 상황별 답이 달라진다." 기온 하나로 옷을 추천하던 앱을 등교 전에 10초 안에 코디를 정하고, 입어 본 느낌까지 기록하는 앱으로 바꾸는 게 목표였어요.

두 앱을 합치면 화면, 기기 저장, 한국 시간 날짜 계산, 날씨 데이터 받기, 그리고 무언가 실패했을 때의 화면까지 갖춰야 합니다. 비개발자인 운영자가 이걸 직접 짜기는 어려워서, 계획서 쓰기부터 제작과 점검까지 AI 코딩 도구에 단계로 나눠 맡겼어요. 사람은 아이디어나 고칠 점을 주고, 계획서를 승인하고, 자기 휴대폰으로 써 본 뒤 내보낼지 정하는 자리에 남았습니다.



이미지 1. 생활 속 불편과 앱이 바꾼 흐름

② 어떤 AI를 어떻게 활용했나요?

계획서를 쓰는 AI와 만드는 AI를 따로 두었어요

쓴 AI 도구는 Anthropic의 Claude Code와 OpenAI의 Codex입니다. 둘 다 파일을 읽고 고치고 명령을 실행하는 AI 코딩 에이전트(AI Coding Agent)인데, 도구보다 역할로 나눠 썼어요. 기획 담당 세션은 계획서만 쓰고 코드와 앱 설정 파일은 건드리지 않습니다. 제작 담당 세션은 승인된 계획만 구현하고, 계획 자체가 마음에 들지 않으면 고치지 않고 멈춘 뒤 운영자에게 보고해요.

체중 다이어리는 Claude Code가 계획서를 쓰고 Codex가 만들었고, 날씨 코디는 Codex가 계획서를 쓰고 Claude Code가 만들었습니다. 계획서 첫머리와 작업 보드에 담당 도구가 적혀 있어요. 기획 담당은 운영자가 준 아이디어나 고칠 점을 바탕으로 설계하고, 요약을 보고한 뒤 승인을 기다립니다.

앱 하나는 이 순서로 지나갑니다.

1) 계획서(기획 담당 AI): 앱마다 계획서 한 장을 씁니다. 체중 다이어리 계획서는 맨 위 '제작자용 요약(Builder Fast Spec)' 표에 화면 골격, 핵심 화면, 계산 규칙, 공유 여부, 광고 위치, 알림을 한 칸씩 적고, 그 아래에 첫 60초 동안 사용자가 누르는 순서, 빈 화면·저장 실패·인터넷 끊김 때의 문구, 넣지 않을 기능을 적어요.

2) 승인(운영자): 계획서가 승인 폴더에 들어가야 다음으로 넘어갑니다. 검사 스크립트의 만들기·빌드 명령은 승인된 계획서가 하나 있을 때만 돌아가요.

3) 제작(제작 담당 AI): 작업은 항상 context 명령으로 시작합니다. 앱 번호를 넣으면 그 앱의 계획서와 중심 파일만 '다음에 읽을 것'으로 돌려줘서, 다른 앱 폴더나 빌드 결과물을 훑지 않게 했어요.

4) 규칙 검사(check): 자주 나오는 규칙은 스크립트가 코드에서 직접 찾습니다. 날짜를 세계 표준시 기준으로 잘라 쓰는 코드가 있으면 "한국 시간 0시부터 8시 59분 사이에는 어제 날짜가 나온다"는 이유와 대신 쓸 함수를 붙여 오류로 내요. 체중 다이어리 계획서도 하루 기록을 한국 시간 기준 하나로 정하고 공용 날짜 함수를 쓰게 했습니다.

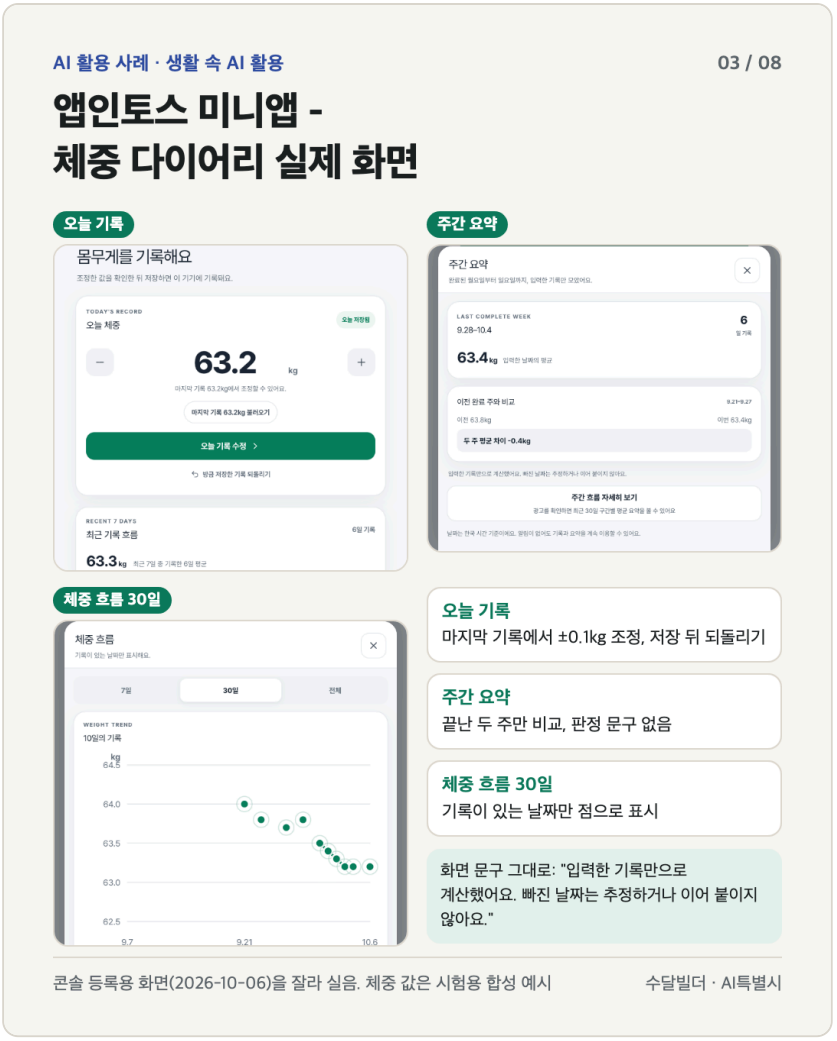
5) 점검 담당 서브에이전트(Subagent): Claude Code 서브에이전트로 정의한 점검 담당 셋이 앱을 봅니다. 이름·설명이 약속한 것이 화면에 있는지(약속 대조), 앱을 실제로 띄워 모든 화면과 실패 경로가 동작하는지(구동 검증, Playwright 브라우저 도구 사용), 검수 반려 사유가 남아 있지 않은지(출시 전 점검)를 각각 확인해요. 판정형 두 담당은 Claude Opus, 체크리스트형인 출시 전 점검은 Claude Sonnet으로 정의 파일에 모델을 적어 두었습니다.

6) 판정 기록: 판정은 소스 파일의 지문(Fingerprint)과 함께 남깁니다. 점검 뒤에 코드를 고치면 기록이 지금 코드와 맞지 않게 되고, 검사가 지문을 바꾼 파일을 짚어 줘요. 셋 중 하나라도 기록이 없으면 출시용 검사(check --release)를 통과하지 못합니다.

7) 테스트와 출시: 검사를 통과하면 번들(Bundle)을 만들어 운영자 휴대폰으로 테스트 버전을 보냅니다. 운영자가 직접 써 보고 "테스트 OK"라고 말하기 전에는 검수 요청을 넣지 않아요. 검수 요청은 그 말을 들은 뒤에만 운영 담당 AI가 넣고, 검수 승인 뒤 출시 버튼은 운영자가 콘솔에서 누릅니다.



이미지 2. 앱 하나를 맡는 AI 구성과 사람이 정하는 자리



이미지 3. 앱인토스 미니앱 - 체중 다이어리 실제 화면

③ 활용 결과 어떤 변화가 있었나요?

체중은 두 번 눌러 적고, 옷은 등교 전에 정해요

체중 다이어리는 앱을 열면 마지막 기록이 미리 채워져 있습니다. 빼기·더하기 버튼으로 0.1kg씩 맞추고 '오늘 기록'을 누르면 끝이에요. 계획서가 정한 기준이 "최소 2회 조작"이었고, 어제 기록이 있으면 '어제와 같은 값' 버튼이 따로 나옵니다. 같은 날 다시 저장하면 기록이 하나 더 생기지 않고 그날 값이 바뀌며, 저장 직후에는 '방금 저장한 기록 되돌리기'가 보여요.

평균과 그래프는 입력한 값만 씁니다. 최근 7일 기록이 3개보다 적으면 평균 대신 "기록이 3회 모이면 7일 평균을 보여드려요"가 나오고, 평균을 보여 줄 때도 "최근 7일 중 기록한 6일 평균"처럼 며칠 치인지 같이 적어요. 그래프는 기록한 날만 점으로 찍고, 날짜가 이어지지 않는 두 점은 선으로 잇지 않습니다. 주간 비교는 월요일부터 일요일까지 다 지난 두 주만 놓고, 두 주 모두 3회 이상 기록했을 때만 차이를 보여 줘요. 차이 옆에 잘했다거나 분발하라는 말은 붙이지 않습니다.

날씨 코디는 시간대별 예보에서 오전 8시와 오후 3시에 가장 가까운 기온을 뽑아 등교와 하교를 비교합니다. 차이가 5도 이상이면 "등교 13°C → 하교 22°C" 같은 안내가 뜨고, 8도 이상이면 "일교차가 커요, 겉옷을 챙기세요"가 붙어요.

교복을 고르면 교복은 그대로 두고 덧입을 것만 추천하는데, 12도 이상 20도 미만이면 조끼나 가디건, 5도 이상 12도 미만이면 가디건·바람막이·두꺼운 양말을 고르는 식입니다.

추천 위에는 이유가 한 줄 붙습니다. 체육이 있는 날이면 "등교 13°C, 하교 22°C로 일교차가 커요 · 체육복으로 갈아입을 걸 감안해 여벌 상의나 겉옷을 하나 더 챙기세요"처럼요. 위치 권한을 허용하지 않아도 주요 도시 14곳 가운데 골라 쓸 수 있고, 날씨를 받아 오지 못하면 기온과 비 여부를 직접 고르는 화면으로 바뀌어요. 계획서에 적은 "마지막 날씨를 현재 값처럼 표시하지 않는다"는 규칙을 구현한 부분입니다.

운영자 쪽에서 바뀐 점도 있습니다.

1) 손이 가는 자리가 정해졌어요. 아이디어나 고칠 점 주기, 계획서 승인, 휴대폰으로 테스트 버전 써 본 뒤 "테스트 OK" 말하기, 출시 버튼 누르기입니다. 나머지는 기획 담당, 제작 담당, 점검 담당, 운영 담당 AI와 검사 스크립트가 차례로 넘겨받아요.

2) 확인한 내용이 기록으로 남습니다. 체중 다이어리의 약속 대조 담당은 2026년 10월 6일 판정에 "홈 최근 7일 6회/63.3kg 평균, 완료 주 6회/63.4kg·이전 63.8kg·차이 -0.4kg로 표시 일치, 30일 그래프 실제 10점과 날짜/kg 축", "누락값 생성·감량 평가 없음"이라고 적었어요. 이 체중 값은 시험용으로 넣은 합성 예시입니다.

3) 구동 확인도 실제로 눌러 본 흐름으로 남습니다. 날씨 코디 제작 기록에는 위치 거절 뒤 직접 지역 고르기, 날씨 조회 실패 뒤 직접 기온 입력, 교복·사복 전환, 옷장 체크에 따른 '대체 제안' 표시, 코디 저장부터 착용 피드백까지 Playwright로 실제 동작을 확인했다고 적혀 있어요.

두 앱은 앱인토스에 출시된 미니앱입니다(체중 다이어리는 계획서에, 날씨 코디는 제작 기록에 콘솔 상태 '열림(OPEN)'으로 적혀 있음). 체중 다이어리 개선판은 위 순서대로 번들을 만들어 운영자 휴대폰에 테스트 버전으로 보냈고, 날씨 코디 개선판도 테스트 버전을 보낸 뒤 검수 단계에 들어갔어요.

AI 활용 사례 · 생활 속 AI 활용 04 / 08

AI에게 시킨 지시와 그 결과

체중 다이어리

계획서에 적은 지시

"최근 7일 평균은 최근 7개 달력 날짜의 실제 기록만 사용하고, 3개 미만이면 평균 대신 … 표시한다. 빠진 날은 점을 잇지 않고 추정·보간하지 않는다."

↓ 제작 담당 AI가 구현

```
store/weightState.ts
function average(records: WeightRecord[]): number | null {
  if (records.length < 3) return null;
  return records.reduce((total, record) => total + record.weight, 0) / records.length;
}
```

화면 문구 "기록이 3회 모이면 7일 평균을 보여드려요."
"기록이 있는 날짜만 표시해요."

날씨 코디

계획서에 적은 지시

"교복 모드는 조끼·가디건·바람막이·양말·우산 같은 조절 가능한 요소에 집중한다."

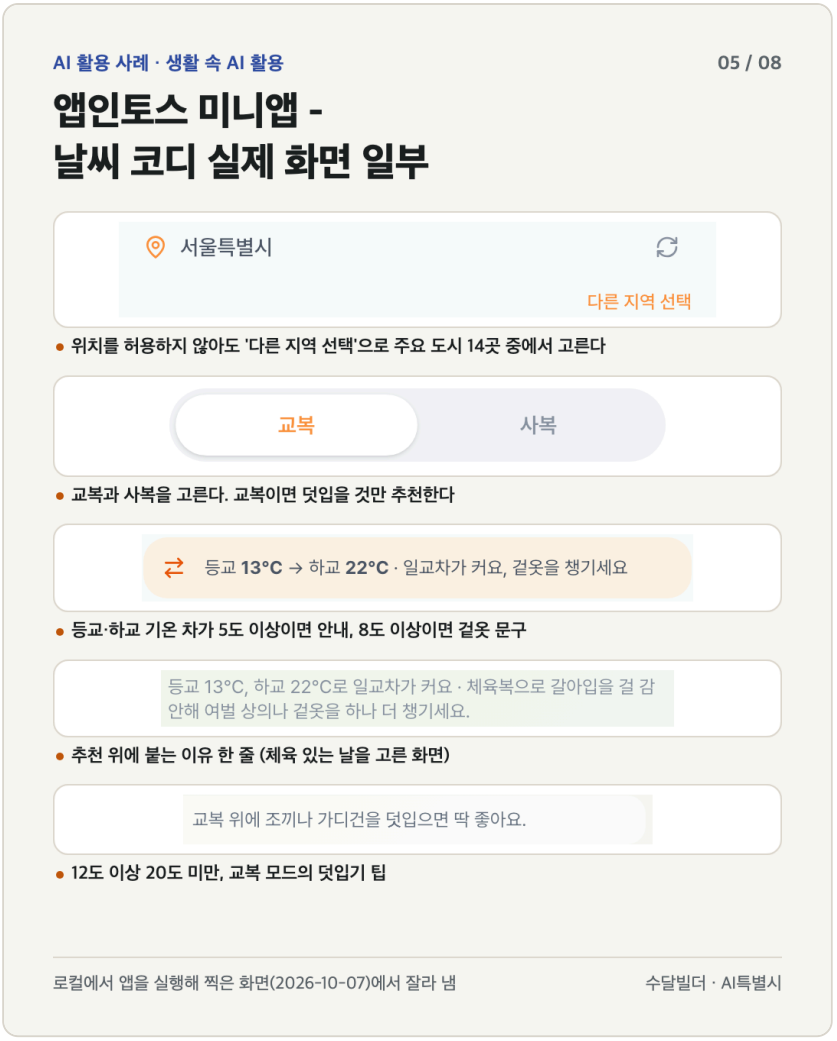
↓ 제작 담당 AI가 구현

```
data/outfitRules.ts (첫 두 단계)
if (temp >= 20) {
  return { items: [], tip: '교복 셔츠만으로 충분해요.' };
}
if (temp >= 12) {
  return { items: [ITEMS.vest, ITEMS.cardigan], tip: '교복 위에 조끼나 가디건을 덧입으면 딱 좋아요.' };
}
```

화면 문구 "교복 위에 조끼나 가디건을 덧입으면 딱 좋아요."

계획서 원문 줄 → 실제 소스 코드 → 앱 화면 문구 수달빌더 · AI특별시

이미지 4. AI에게 시킨 지시와 그 결과



이미지 5. 앱인토스 미니앱 - 날씨 코디 실제 화면 일부



이미지 6. 점검 담당 에이전트가 남긴 실제 판정 기록

④ 나만의 방식 또는 개선 포인트는 무엇인가요?

계획서에 '넣지 않을 것'과 '모자랄 때'를 먼저 적었어요

AI에게 기능을 늘려 달라고 하기 전에, 넣지 않을 것과 데이터가 모자랄 때 할 일을 계획서에 먼저 적었습니다. 만드는 AI는 계획서에 적힌 대로 구현하니, 이 칸이 곧 지킬 규칙이 돼요.

1) 넣지 않을 것: 체중 다이어리 계획서는 BMI 수치와 판정 라벨, 내일 체중 맞히기, 리그·배지·연속 기록, 기본 공유 카드, 다른 사람과의 순위를 뺐습니다. 적어 둔 이유는 "사용자 체중은 개인 데이터이며 비교·경쟁이 제품 가치를 높인다는 근거가 없다"예요. 날씨 코디 계획서는 체형 평가, 성별 고정 코디, 과소비·브랜드 구매 압박, 외모 비하 표현을 금지했고 쇼핑몰 연결과 사진 업로드는 범위 밖으로 두었어요.

2) 모자랄 때: 기록이 3회 미만이면 평균을 내지 않고, 빠진 날짜는 추정하거나 이어 붙이지 않습니다. 날씨를 못 받으면 직접 입력으로 바꾸고, 미세먼지는 실제로 값이 올 때만 카드를 보여 주고 조회에 실패하면 카드를 숨겨요.

3) 저장이 안 됐을 때: 체중 다이어리는 기기에 쓰지 못하면 "기기에 저장하지 못했어요"를 띄우고, 저장에 성공하기 전에는 저장됐다고 표시하지 않습니다. 기존 기록을 읽지 못한 경우에도 원래 저장값을 지우거나 기본값으로 덮어쓰지 않게 했어요.



이미지 7. 계획서에 먼저 적은 지킬 줄

윤리와 책임을 지키려고 둔 장치

- 개인 정보: 체중 기록은 이 기기에만 저장되고 다른 기기와 자동으로 동기화되지 않는다고 화면에 밝힙니다. 알림 문구와 잠금화면 미리보기에는 체중값·증감량·목표 잔여량을 넣지 않고, 분석 이벤트에는 기록 여부만 보내도록 계획서에 적었어요. 날씨 코디는 위치를 시·구 수준으로만 보여 주고, 공유 카드에는 위치명·학교명 없이 날씨·코디 요약·한 줄 팁만 담습니다. 옷장에서도 브랜드, 가격, 신체 치수는 받지 않아요.

- 판정처럼 읽히지 않게: 체중 차이는 색으로 좋고 나쁨을 가르지 않고, 목표 체중은 설정에서 직접 정한 사람에게만 보입니다. 날씨 코디의 스타일 무드는 깔끔, 캐주얼, 힙, 편안 네 가지로, 성별이나 체형이 아니라 핏과 색, 소재로 설명해요. 계획서에 주 사용자를 13~18세로 적었기 때문에 둔 기준입니다.

- 묻는 순서: 알림 동의는 앱을 열자마자 묻지 않습니다. 날씨 코디는 첫 코디를 저장한 뒤에야 '평일 아침 코디 알림 받기' 카드가 나타나요.

- 사람 확인 단계(HITL, Human-in-the-Loop): 계획서 승인, 휴대폰 테스트 뒤 "테스트 OK" 말하기, 출시 버튼은 운영자가 합니다. 검수 요청은 운영자가 테스트를 마쳤다고 말한 뒤에만 AI가 넣어요. 날씨 코디는 앱 정보(스크린샷·설명·카테고리) 검토 요청을 내기 전에 콘솔 확약 사항 5개를 운영자에게 보여 주고 명시적인 동의를 받은 뒤 냈다고 제작 기록에 적혀 있습니다.

- 데이터 출처: 날씨 코디의 날씨와 대기질 값은 Open-Meteo API에서 받습니다. 이 글과 이미지에 나온 체중 값은 시험용으로 넣은 합성 예시이고, 날씨 화면은 2026년 10월 7일 로컬에서 앱을 실행해 찍은 화면이에요.

- AI 생성 표시: 이 출품작의 이미지는 실제 앱 화면과 HTML로 직접 그린 도식이고, AI로 생성한 그림은 없습니다.

만들며 정한 점

1) 체중 다이어리의 등록 정보 초안(app.spec.json)에서 앱 설명과 검색 키워드의 BMI 판정, 감량 효과, 격려 표현을 걷어내고 "체중을 기록하고 최근 흐름을 차분히 확인해요"로 좁혔습니다. 콘솔 앱 정보에는 휴대폰 테스트를 마친 뒤 반영하기로 했어요.

2) 날씨 코디는 계획서의 핵심 화면에 맞춰 하단 탭을 오늘·시간별·주간·팁에서 오늘·옷장·기록·가이드로 바꾸고, 시간별 날씨는 홈 안의 작은 띠로 남겼어요.

3) 기획 단계의 점검 담당 셋(제품 방향, 약속, 디자인)은 한 번에 같이 돌리고, 지적이 서로 부딪히면 어떻게 판단했는지 적습니다. 하나씩 차례로 돌리면 지적이 나올 때마다 계획서를 다시 쓰게 되기 때문이에요.

4) 점검은 알려진 수정을 다 끝낸 뒤에 부르고, 지적을 고쳐 지문이 바뀌면 그 지적을 낸 담당만 다시 돌립니다. 작업 규칙에는 "재실행이 진짜 비용이다"라고 적어 두었어요.

⑤ 다른 사람도 따라 할 수 있나요?

불편 한 줄에서 시작하는 순서

앱인토스에 올리지 않고 내 휴대폰에서만 쓰는 작은 웹 페이지로 만들어도 순서는 같습니다. 이렇게 시작해 보길 권해요.

1) 불편을 한 문장으로 적습니다. 누가, 언제, 무엇 때문에 불편한지까지요. "중·고등학생은 아침 기온만 보고 옷을 골랐다가 하교 때 불편해진다"처럼요.

2) AI에게 코드보다 계획서를 먼저 받습니다. 아래 요청문을 그대로 써도 돼요.

"아래 불편을 푸는 휴대폰용 작은 앱의 계획서를 써 줘. 코드는 아직 쓰지 마. 첫 60초 동안 사용자가 누르는 순서, 계산 규칙(데이터가 모자랄 때 어떻게 할지 포함), 빈 화면·저장 실패·인터넷 끊김 때 보여 줄 문구, 넣지 않을 기능과 그 이유를 표로 정리해 줘. 개인 정보는 기기 밖으로 보내지 않는 쪽으로 짜 줘."

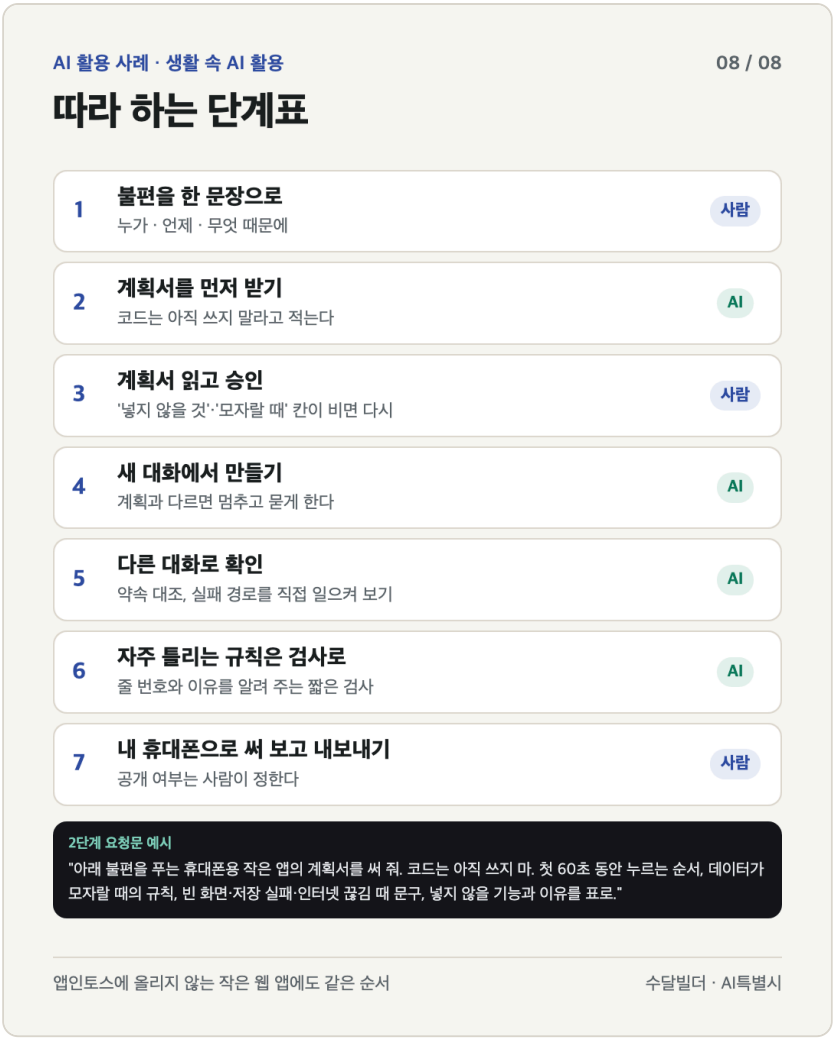
3) 계획서를 직접 읽고 승인합니다. 넣지 않을 것과 모자랄 때 칸이 비어 있으면 채워 달라고 다시 요청해요.

4) 새 대화(다른 AI 도구여도 좋아요)에 승인한 계획서만 주고 만들게 합니다. "계획과 다르게 하고 싶으면 고치지 말고 멈춰서 물어봐 줘"를 함께 적어요.

5) 확인도 다른 대화에 맡깁니다. "계획서가 약속한 것이 화면에 다 있는지 하나씩 대조해서 없는 것만 알려 줘"라고 하고, 저장 실패나 인터넷 끊김 같은 실패 경로도 직접 일으켜 보게 해요.

6) 자주 틀리는 규칙은 검사로 바꿉니다. 날짜를 한국 시간으로 구하는지처럼 코드에서 찾을 수 있는 규칙이면, 찾아서 줄 번호와 이유를 알려 주는 짧은 검사를 AI에게 같이 만들어 달라고 하면 돼요.

7) 내 휴대폰으로 직접 써 본 뒤에 내보냅니다. 내보내기와 공개 여부는 사람이 정합니다.



이미지 8. 따라 하는 단계표

체중이나 옷차림이 아니어도 물 마시기, 경조사비, 교통비처럼 "적어 두고 다시 찾는" 생활 기록이면 같은 표를 그대로 쓸 수 있어요. 칸에 들어갈 내용만 바꾸면 됩니다.

AI 활용 표시와 출처

- 이 출품 글의 초안과 도식은 Claude Code로 만들었고, 문장마다 앱 폴더의 원본 파일과 대조했습니다. 운영자가 최종 내용을 확인하고 제출합니다.

- 이미지 1, 2, 7, 8: 두 앱의 계획서, 작업 규칙 파일, 역할 정의 파일, 검사 스크립트에 적힌 내용으로 HTML 도식을 그렸습니다.

- 이미지 3: 체중 다이어리 개선판의 콘솔 등록용 화면 3장(2026년 10월 6일 생성)입니다. 체중 값은 시험용 합성 예시예요.

- 이미지 4: 계획서 원문 줄과, 그 줄을 구현한 실제 소스 코드·화면 문구를 나란히 옮겼습니다.

- 이미지 5: 날씨 코디를 로컬에서 실행해 찍은 화면(2026년 10월 7일)에서 일부를 잘라 실었습니다.

- 이미지 6: 점검 담당 에이전트가 남긴 판정 기록 파일에서 옮겼고, 파일 지문 목록은 줄였습니다.

- 날씨 데이터: Open-Meteo(open-meteo.com) 예보·대기질 API