ACTIGENCE

액티전스 교육 자료 통합본

바이브코딩,
클로드 실무 토탈 솔루션

개념부터 도구, 업무 자동화, 조직 안착까지

---

v1.0.0 · 2026-11-07 14:17 KST 기준

최신본 actigence.ai/library/downloads

이 책의 버전 v1.0.0 · 2026-11-07 14:17 KST 기준

이 책은 교육 내용이 바뀔 때마다 개정합니다. 가진 파일의 버전이 자료실의 버전보다 낮으면 새로 받습니다.

최신본: actigence.ai/library/downloads · 개정 이력은 맨 뒤 판권 면에 있습니다.

차례

들어가며

머리말

1부 개념

1장 AI와 클로드

2장 요금제 판단 기준

3장 프롬프트와 메타 프롬프팅

4장 바이브코딩

5장 설치 없이 시작하는 첫 화면

6장 PRD로 키우는 두 번째 프로그램

7장 AX의 언어

2부 도구

8장 데스크탑 앱과 프로젝트

9장 Claude Code와 합의서

10장 멀티 세션과 서브에이전트

11장 스킬과 MCP

12장 이미지·영상 생성 연동

3부 업무 자동화

13장 문의 응대 자동화: 문자 발송 연동

14장 데이터 분석
15장 딥리서치
16장 문서 업무 전 과정: 제안서와 기획서
17장 결과보고서: 기록과 맞춰 보는 문서
18장 간단한 웹앱으로 업무 자동화

4부 품질과 운영

19장 계획의 깊이와 자가검증
20장 세션에서 배포까지

5부 조직에 심기

21장 실행형 AX 컨설팅
22장 사내 AX 담당자와 첫 3개월

부록

부록 A 프롬프트 카드 12종
부록 B 스킬 14종
부록 C 용어 사전
부록 D 체크리스트

머리말

이 책은 AI를 업무에 처음 들이는 실무자, 대표, 사내 담당자를 위한 실무서입니다. 클로드(Claude)를 대화 상대로 쓰는 단계에서 시작해, 코드를 쓰는 도구로 일을 맡기고, 조직 안에 정착시키는 단계까지 한 흐름으로 다룹니다.

누구를 위한 책인가

코딩 경험이 없어도 읽을 수 있습니다. 필요한 것은 반복되는 업무 하나와, 그 일을 줄여 보겠다는 마음입니다. 다음 독자를 염두에 두고 썼습니다.

- 매주 같은 보고, 문의 응대, 표 정리에 시간을 쓰는 실무자
- 직원에게 AI 도입을 시키기 전에 먼저 감을 잡고 싶은 대표와 팀장
- 회사에서 AI 활용을 맡게 된 사내 담당자

이 책은 AI가 모든 것을 바꾼다는 식의 이야기를 하지 않습니다. 무엇을 시키고, 무엇을 확인하고, 어디서 멈출지를 단계별로 적습니다. 효과를 보여 주는 수치나 후기는 싣지 않았습니다. 예시는 모두 가상의 예이고, 그렇게 표시했습니다.

책의 구성

책은 5부와 부록으로 이루어져 있습니다.

부	내용	장
1부 개념	AI와 클로드, 요금제 판단, 프롬프트, 바이브코딩, 설치 없이 시작하는 첫 화면, PRD로 키우는 두 번째 프로그램, AX의 언어(워크플로우, 단일 원본, md 원본, 챔피언)	1~7장
2부 도구	데스크탑 앱, Claude Code, 멀티 세션, 스킬과 MCP, 이미지·영상 연동	8~12장
3부 업무 자동화	문의 응대, 데이터 분석, 딥리서치, 제안서·기획서 전 과정, 결과보고서, 간단한 웹앱	13~18장

부	내용	장
4부 품질과 운영	계획의 깊이와 자가검증, 세션에서 배포까지	19~20장
5부 조직에 심기	실행형 AX 컨설팅, 사내 담당자와 첫 3개월	21~22장

1부의 5장과 6장은 코딩 경험이 없는 독자가 화면 하나를 만들어 주소를 받기까지의 실습 경로입니다. 1부 끝의 7장은 이후 장이 함께 쓰는 어휘를 먼저 정리합니다. 3부의 16장과 17장은 문서 업무를 처음부터 끝까지 다룹니다. 16장은 제안서와 기획서를, 17장은 끝난 일을 기록과 맞춰 보고하는 결과보고서를 맡습니다.

부록은 네 가지입니다. 부록 A는 프롬프트 카드, 부록 B는 스킬, 부록 C는 용어 사전, 부록 D는 체크리스트입니다. 본문을 읽다가 모르는 말이 나오면 부록 C로, 바로 써 보고 싶으면 부록 A와 D로 가면 됩니다.

1~22장은 같은 모양입니다. 본문 뒤에 "실습"이 있고, 복사해서 쓸 수 있는 프롬프트가 들어 있습니다. 마지막에 "정리"가 3~5개 항목으로 붙습니다.

교육 과정과 함께 쓰기

이 책은 반나절 안팎의 특강이나 여러 주에 걸친 과정의 교재로도 쓸 수 있습니다. 짧은 특강이라면 1부(1~7장)로 개념을 잡고, 2부(8~12장)로 도구를 보고, 문의 응대(13장)와 활용 장(14장, 15장, 18장)에서 하나를 골라 시연하는 순서가 무리 없습니다. 시연에서 다 보여 주지 못하는 이유와 확인 방법, 집에서 다시 해 볼 순서는 책이 맡습니다.

4주 과정인 AI 바이브코딩 입문과도 이어집니다. 입문 과정의 회차별 실습 프롬프트는 actigence.ai 자료실에 있고, 이 책에서는 4장이 그 흐름을 소개합니다.

입문 과정	주제	이 책에서 가장 가까운 곳
1회차	도구 준비, 첫 결과물	1장, 4장
2회차	화면·버튼·입력 폼·목록	4장, 18장
3회차	반복 업무 자동화	3장, 14장

입문 과정	주제	이 책에서 가장 가까운 곳
4회차	점검, 웹 공개, 막혔을 때 묻는 법	4장, 20장

완전 초보자를 위한 이틀 과정으로 쓴다면 아래 순서가 맞습니다. 첫째 날은 설치 없이 브라우저만 쓰고, 둘째 날은 내 컴퓨터의 폴더에서 일합니다. 한 가지 주제(가상의 가족 일정표)를 이틀 내내 끌고 가는 구성이므로, 장을 건너뛰지 않기를 권합니다.

이틀 과정	주제	이 책에서 가장 가까운 곳
첫째 날 이론	물어보기와 만들게 하기, 사실 확인, 프롬프트 네 칸	1장, 3장, 4장
첫째 날 실습	한 줄 요청, 요청문, 모아서 고치기, 주소 받기	5장
둘째 날 이론	PRD, 화면·정보·규칙, 서버와 주소	6장
둘째 날 실습	PRD, 허락 창, 파일 만들기, 점검, 배포	6장, 18장, 20장

교안과 함께 보기

액티전스가 강의에서 쓰는 교안도 자료실에서 PDF로 받을 수 있습니다. 교안은 화면에서 보여 줄 장면과 실습 순서를 담고, 이 책은 그 장면의 이유와 확인 방법을 담습니다. 교안도 책과 같은 방식으로 버전을 관리합니다.

교안	다루는 내용	이 책에서 가장 가까운 곳
AI 업무 활용 과정 1회차	AI의 원리와 한계, AX의 언어, 프롬프트 기본기	1장, 2장, 3장, 7장
AI 업무 활용 과정 2회차	Claude Code 깊게 쓰기, 스킬·MCP, 도구를 주소와 데이터까지	9장, 10장, 11장, 18장
AI 업무 활용 과정 3회차	제안서 한 건을 여덟 단계로	15장, 16장
AI 업무 활용 과정 4회차	원본 하나에서 목적별 문서로	7장, 16장
AI 업무 활용 과정 5회차	기록에서 결과보고서까지	14장, 17장
바이브코딩 8시간 특강 1일차	말로 만드는 첫 화면과 주소	5장
바이브코딩 8시간 특강 2일차	PRD부터 데스크톱 앱, 배포까지	6장, 18장, 20장
디지털 역량 특강	AI 영상 만들어 올리기, 가족 일정 웹	4장, 5장, 12장

어떻게 읽으면 좋은가

- AI를 처음 쓰거나 써 봤지만 감이 없다면 1부부터 차례로 읽습니다. 1부는 짧고, 이후의 모든 장이 1부의 용어를 씁니다.
- 코딩 경험이 전혀 없고 직접 화면을 만들어 보고 싶다면 4장을 읽은 뒤 5장과 6장을 순서대로 따라 합니다.
- 당장 풀어야 할 업무가 있다면 3부로 바로 가도 됩니다. 도구 설명이 필요한 곳에서는 "9장 참고"처럼 번호로 안내합니다.
- 문서 업무(제안서, 기획서, 결과보고서)가 중심이라면 1부(1~7장)를 읽은 뒤 16장과 17장부터 읽어도 됩니다.
- 사내 도입을 맡았다면 2장, 7장, 19장, 21장, 22장을 먼저 읽고 나머지는 필요할 때 펼치기를 권합니다.

장마다 한 번에 읽을 수 있는 분량으로 나눴습니다. 읽는 것보다 실습을 한 번 해 보는 편이 이해에 도움이 됩니다.

표기 방법

프롬프트는 회색 상자(코드 블록) 안에 넣었습니다.

프롬프트

> 나는 [직무]를 맡고 있고, AI는 [처음 / 조금] 써 봤어. 내 수준에 맞춰 설명해 줘.

- 상자 안의 글을 그대로 복사해서 클로드 입력창에 붙여 넣습니다.
- 대괄호 `[ ]`로 표시한 곳은 자신의 상황에 맞게 바꿔 넣습니다. 대괄호 안의 `/`는 "둘 중 하나를 고르라"는 뜻입니다.
- 명령어와 파일 이름은 영문 그대로 적었습니다.
- 제품 화면과 기능은 자주 바뀝니다. 바뀔 가능성이 큰 내용에는 "(2026년 11월 기준)"을 붙이고 공식 문서 링크를 달았습니다. 책과 화면이 다르면 화면과 공식 문서가 맞습니다.

이 책의 개정 방침

AI 도구는 몇 달만 지나도 달라집니다. 그래서 이 책은 한 번 내고 끝내지 않고 계속 고칩니다.

- 표지와 판권 면에 버전과 기준 일시가 적혀 있습니다. 지금 손에 든 책이 언제 기준인지 먼저 확인하세요.
- 최신판은 항상 액티전스 자료실에서 내려받습니다. 예전에 받은 파일을 계속 쓰지 않기를 권합니다.
- 무엇이 바뀌었는지는 판권 면의 개정 이력에 기록합니다.
- 가격, 사용량 한도처럼 숫자로 박아 두면 금방 틀리는 정보는 책에 쓰지 않았습니다. 해당 공식 페이지를 직접 확인하도록 안내합니다.

문의

오류 제보, 내용 문의, 교육·컨설팅 문의는 cio@actigence.ai 로 보내 주세요. 제보는 다음 개정에 반영합니다. 업무 자료나 개인정보는 메일에 붙이지 말고, 문제가 되는 문장과 장 번호만 알려 주시면 됩니다.

1부

개념

1장 · AI와 클로드

2장 · 요금제 판단 기준

3장 · 프롬프트와 메타 프롬프팅

4장 · 바이브코딩

5장 · 설치 없이 시작하는 첫 화면

6장 · PRD로 키우는 두 번째 프로그램

7장 · AX의 언어

1장

AI와 클로드

이 장은 AI, 생성형 AI, 대규모 언어 모델이 어떻게 다른 말인지 정리하고, 클로드(Claude)가 어떤 제품이며 어디에서 쓸 수 있는지 설명합니다. 그리고 질문에 답을 받는 사용법과 일을 맡기는 사용법의 차이, 쓰기 전에 알아야 할 한계를 다룹니다.

AI라는 말의 범위

"AI"는 가장 넓은 이름입니다. 사람이 하던 판단이나 인식을 컴퓨터가 흉내 내는 기술 전체를 가리킵니다. 사진에서 얼굴을 찾는 기능, 스팸 메일을 거르는 기능, 길을 안내하는 기능도 모두 AI에 속합니다.

그 안에 생성형 AI가 있습니다. 글, 이미지, 코드, 음성처럼 새로운 결과물을 만들어 내는 AI입니다. 질문을 넣으면 답을 쓰고, 설명을 넣으면 그림을 그립니다.

글을 다루는 생성형 AI의 중심에는 대규모 언어 모델(LLM, Large Language Model)이 있습니다. 대량의 글을 학습해서, 앞에 놓인 글을 보고 그다음에 올 말을 이어 가는 방식으로 문장을 만듭니다. 사람이 쓴 것처럼 읽히지만 내부에서 "이해"를 하는 방식이 사람과 같다고 볼 수는 없습니다. 이 차이가 뒤에서 말할 한계의 뿌리입니다.

말	범위	예
AI	가장 넓음	추천, 인식, 번역, 대화
생성형 AI	결과물을 새로 만드는 AI	글, 이미지, 코드 생성
대규모 언어 모델	글을 중심으로 하는 생성형 AI의 엔진	Claude

확률로 다음 말을 예측하는 도구

생성형 AI는 많은 양의 글을 학습해서, 다음에 올 말을 확률로 예측하는 도구입니다. 이 한 문장에서 쓰는 쪽이 알아야 할 결과 세 가지가 나옵니다. 상세는 뒤의 "알아야 할 한계"에서 다룹니다.

원리	결과	대응
확률로 예측한다	같은 질문에도 매번 답이 조금씩 다릅니다	질문과 요구를 명확하게 씁니다

원리	결과	대응
학습한 글을 바탕으로 한다	학습에 없던 최신 정보는 모릅니다	웹 검색과 딥리서치(15장)를 씁니다
그럴듯한 말을 잇는다	틀린 내용도 그럴듯하게 말합니다	팩트체크를 거칩니다

지금 되는 이유 세 가지

AI는 오래전부터 있었는데 업무에 쓸 만해진 것은 최근입니다. 이유를 세 가지로 정리할 수 있습니다. 아래 용어의 정의는 업계에서도 아직 유동적이므로, 이 책에서는 이렇게 쓴다고 한정해 둡니다.

- 문맥을 이해하는 수준이 올라서: 말투와 배경이 담긴 지시를 알아듣고 따릅니다. 지시를 잘 쓰는 기술을 프롬프트 엔지니어링이라고 부릅니다. 3장에서 다룹니다.
- 긴 문서를 통째로 다루게 되어서: 한 번에 볼 수 있는 분량이 늘었습니다. 무엇을 보여 줄지 고르고 관리하는 일을 컨텍스트 엔지니어링이라고 부릅니다. 10장에서 이어집니다.
- 긴 시간 큰 작업을 수행하게 되어서: 여러 단계를 스스로 이어서 합니다. 그 작업이 안전하게 돌아가도록 규칙, 도구, 확인 절차를 갖춰 두는 일을 하네스 엔지니어링이라고 부릅니다. 9장과 16장에서 이어집니다.

세 용어는 모두 "AI에게 무엇을 어떻게 줄 것인가"의 범위가 넓어진 순서이기도 합니다. 공식 문서의 설명은 Claude 공식 문서에서 확인합니다(2026년 11월 기준).

클로드는 무엇인가

클로드는 앤트로픽(Anthropic)이 만든 AI 모델의 이름이고, 그 모델을 쓰는 제품들의 이름이기도 합니다. 같은 모델이라도 어떤 화면에서 어떤 도구와 함께 쓰느냐에 따라 할 수 있는 일이 달라집니다. 그래서 "클로드로 이게 되나요"라는 질문에는 "어디서 쓰느냐에 따라 다릅니다"라고 답하게 됩니다.

어떤 서비스가 있는가 (2026년 11월 기준)

글을 다루는 생성형 AI 서비스는 여럿이고, 대표적인 것이 ChatGPT, Gemini, Claude입니다. 어느 쪽이 낫다고 단정하기보다 하려는 일에 맞춰 고릅니다. 아래는 용도별 선택 기준을 위한 대략의 성격이며, 제품 특성은 빨리 바뀌므로 각 공식 사이트에서 확인합니다.

서비스	특성	고려할 점
ChatGPT	범용 대화와 이미지 생성 등 기능의 폭이 넓습니다	쓰려는 기능이 내 요금제에 포함되는지 확인합니다
Gemini	구글 서비스(메일, 드라이브, 검색)와 연결해 쓰기 좋습니다	구글 서비스를 쓰지 않는 업무에서는 이점이 줄어듭니다
Claude	긴 문서 처리, 코딩, 파일 작업을 위한 제품군을 갖추고 있습니다	사진·일러스트 같은 이미지는 직접 만들지 못해 외부 도구와 연결해야 합니다(12장)

공식 사이트는 chatgpt.com, gemini.google.com, claude.ai입니다. Claude의 이미지 관련 범위는 공식 안내에서 확인합니다. 이 책은 Claude를 중심으로 설명하지만, 같은 원리는 다른 서비스에도 적용됩니다.

어디에서 쓰는가 (2026년 11월 기준)

- 웹, 데스크탑 앱, 모바일 앱: 대화로 질문하고 문서와 표를 다루는 가장 일반적인 입구입니다. 처음이라면 여기서 시작합니다. 주소는 claude.ai입니다.
- Claude Code: 파일을 읽고, 만들고, 고치고, 명령을 실행하는 코딩 도구입니다. 터미널, 개발 도구(IDE)의 확장 프로그램, 데스크탑 앱, 웹에서 쓸 수 있다고 공식 문서에 안내되어 있습니다. 자세한 내용은 Claude Code 소개와 공식 문서를 확인합니다. 9장에서 다룹니다.
- API: 개발자가 자기 서비스나 사내 시스템에 클로드를 붙일 때 쓰는 연결 방식입니다. 이 책의 대상 독자가 직접 만질 일은 적지만, 사내 도구에 AI가 들어 있다면 그 뒤에서 API가 쓰이는 경우가 많습니다.

기능으로 보는 세 층

화면 이름보다 할 수 있는 일의 범위로 나누면 세 층으로 이해됩니다.

층	하는 일	예
1층 채팅	웹과 앱에서 대화하고 간단한 문서 작업을 합니다	메일 초안, 요약
2층 폴더 작업	내 컴퓨터의 폴더에 직접 접근해 파일 단위로 일합니다	폴더 안 파일 정리, 표 만들기
3층 Claude Code	컴퓨터를 조작하고 코딩까지 합니다	프로그램 제작, 자동화

위층은 아래층이 하는 일을 포함합니다. 그래서 위층 하나를 익히면 아래층은 따라옵니다. 2층에 해당하는 기능은 Cowork라는 이름으로 안내되며, 데스크탑 앱에서 연결한 폴더만 읽고 쓰는 방식입니다. 이름과 제공 범위는 바뀔 수 있으니 공식 안내를 확인합니다(2026년 11월 기준).

제공되는 화면과 기능은 계속 바뀝니다. 이 책에서 어떤 기능이 없다고 느껴지면 공식 지원 문서 (support.claude.com)에서 먼저 확인합니다.

대화 상대로 쓰기와 일을 맡기기

AI를 쓰는 방식은 크게 둘로 나뉩니다.

첫째는 대화 상대로 쓰는 방식입니다. 질문하고, 답을 읽고, 다시 묻습니다. 메일 초안, 요약, 아이디어 정리처럼 한 번의 주고받음으로 끝나는 일에 맞습니다. 결과를 받는 순간 사람이 읽고 판단합니다.

둘째는 일을 맡기는 방식입니다. 이때 AI는 답만 하지 않고, 파일을 열고 만들고 실행하는 여러 단계를 스스로 이어서 합니다. 이런 형태를 AI 에이전트라고 부릅니다. 폴더 안의 엑셀 열 개를 열어 같은 규칙으로 정리하고, 결과 파일을 저장하는 일이 예입니다.

두 방식은 사람의 역할이 다릅니다.

- 대화: 사람이 매 답변을 읽고 다음 질문을 정합니다. 실수는 읽는 순간 발견됩니다.
- 위임: 사람이 목표와 제약을 먼저 정하고, 중간 단계는 AI가 진행합니다. 실수는 결과물과 기록을 열어 보아야 발견됩니다.

위임은 편하지만 책임이 사라지지 않습니다. 무엇을 허용하고 무엇을 막을지, 끝난 뒤 무엇을 확인할지를 사람이 정해야 합니다. 이 책의 2부부터는 이 위임을 안전하게 하는 방법이 중심입니다.

> **핵심** 질문에 답을 받는 일과 일을 맡기는 일은 다릅니다. 맡길수록 사람은 앞에서 기준을 정하고 뒤에서 확인하는 쪽으로 일하게 됩니다.

초안, 검수, 발송 세 단계

AI를 쓰는 일의 대원칙은 하나입니다. AI가 초안을 만들고, 사람이 검수하고, 그다음에 발송합니다.

단계	맡는 쪽	하는 일
1. 초안	AI	빈 화면 채우기, 여러 버전 뽑기, 다시 쓰기, 정합성 점검
2. 검수	사람	사실 확인, 최종 판단, 발송 결정
3. 발송과 기록	사람	실제 사용, 기록이 쌓여 다음 일의 재료가 됨

위임할수록 2단계를 건너뛰고 싶어지지만, 2단계가 빠지면 AI의 실수가 그대로 밖으로 나갑니다. 검수의 방법은 19장 "최종 판단은 사람이 합니다"에서, 3단계의 기록을 조직의 자산으로 만드는 방법은 22장에서 다룹니다.

모델은 등급으로 고릅니다

클로드는 모델이 여러 개이고, 새 모델이 계속 나옵니다. 이름과 버전 번호는 금방 바뀌므로 책에 외워 둘 필요가 없습니다. 이름보다 등급(성격)으로 이해하면 오래 갑니다.

공식 모델 소개 페이지는 모델을 대략 이런 성격으로 나눕니다(2026년 11월 기준).

- 가장 어려운 추론과 긴 호흡의 에이전트 작업에 맞춘 상위 모델
- 오래 이어지는 코딩과 지식 업무에 맞춘 모델
- 속도와 성능의 균형이 좋은 중간 모델
- 분류, 추출처럼 빠르게 많이 처리하는 일에 맞춘 가벼운 모델

고르는 요령은 단순합니다.

- 처음에는 화면에 기본으로 선택된 모델로 시작합니다.
- 결과가 얕거나 긴 작업에서 자꾸 흐름을 놓치면 한 단계 위 모델을 시도합니다.
- 간단한 정리나 분류를 많이 시킬 때는 가벼운 모델이 더 빠르고 사용량 부담도 적습니다.
- 모델 이름을 업무 문서나 규정에 못 박지 않습니다. "현재 가장 균형이 좋은 모델"처럼 성격으로 적고, 현재 목록은 공식 모델 소개에서 확인합니다.

앱에서 어떤 모델을 고를 수 있는지는 요금제와 시기에 따라 다릅니다. 이 부분은 2장에서 이어집니다.

알아야 할 한계

그럴듯한 오답

AI는 모르는 내용도 자신 있는 말투로 이어 쓸 수 있습니다. 이를 환각(hallucination)이라고 합니다. 없는 책 제목, 존재하지 않는 법 조항, 맞지 않는 숫자가 실제로 나올 수 있습니다. 특히 날짜, 금액, 고유명사, 출처는 사람이 직접 확인합니다. 지시문에 "모르면 모른다고 말해 줘"를 넣으면 줄어들지만 사라지지는 않습니다.

사례 두 가지를 봅니다. 모두 공식 발표와 보도로 확인되는 일반적인 사실만 옮기며, 세부 경과는 쓰지 않습니다.

- 과도한 칭찬 사례: 해외의 한 상용 모델이 사용자의 취향에 맞춰 지나치게 칭찬하고 동조하는 경향을 보여 업데이트가 철회된 일이 있었습니다. 이런 경향을 아첨(sycophancy)이라고 부릅니다. 사용자의 반응을 반영해 학습하는 과정에서 사용자가 좋아하는 답 쪽으로 치우칠 수 있다고 설명됩니다.
  실용적인 결론은 두 가지입니다. 내가 틀려도 맞다고 해 줄 수 있으니 동의와 공감을 구하는 용도는 취미로만 씁니다. 판단이 필요한 일에는 "문제가 있으면 반박해 줘"라고 요구합니다.
- 가짜 판례 사례: 해외에서 법원에 낸 서면에 존재하지 않는 판례가 들어간 일이 있었습니다. 챗봇이 만든 판례였고, 그 챗봇에게 진짜인지 물었더니 진짜라고 답했다고 보도되었습니다. 교훈은 같은 AI에게 맞느냐고 물어서 검증하면 안 된다는 것입니다. 이 한계는 19장의 자가검증에서 다시 다룹니다.

매번 달라지는 답

같은 질문을 두 번 보내면 답이 조금씩 다릅니다. 다음 말을 확률로 고르기 때문입니다. 결과가 흔들려서 곤란하다면 질문이 열려 있다는 신호이므로, 형식과 기준을 구체적으로 적습니다. 3장에서 다룹니다.

답의 흔들림은 "온도"라는 설정으로 설명하기도 합니다. 온도가 낮으면 가장 흔한 말을 골라 늘 비슷한 답을 내고, 높으면 드문 말도 골라 새롭지만 없는 내용을 지어낼 위험이 커집니다. 사실을 다루는 일은 낮은 쪽, 아이디어를 모으는 일은 높은 쪽이 맞습니다. 앱과 서비스에서는 이 값을 직접 조절할 수 없는 경우가 많으니, 일에 맞게 지시문으로 방향을 잡는다고 이해하면 충분합니다. 또 하나 알아 둘 말은 과적합입니다. 배운 예시에 지나치게 맞춰져 새 상황에서 엉뚱하게 답하는 현상입니다. 예시를 많이 붙이면 AI가 예시를 그대로 따라 하는 것도 같은 성질입니다.

학습 시점과 웹 검색

모델은 특정 시점까지의 자료로 학습합니다. 그 이후에 생긴 일은 모델 안에 없습니다. 최신 정보가 필요한 질문은 웹 검색 기능이 켜진 상태에서 하거나, 최신 자료를 직접 붙여 넣어 줍니다. 검색 기능을 쓰더라도 AI가 가져온 출처를 클릭해서 원문을 확인하는 습관이 필요합니다. 검색 기능의 제공 여부와 위치는 시기와 요금제에 따라 달라지니 고객 지원 센터에서 확인합니다.

붙여 넣으면 안 되는 것

대화창에 넣은 내용은 AI 서비스로 전송됩니다. 다음은 넣기 전에 반드시 지우거나 가짜 값으로 바꿉니다.

- 개인정보: 이름, 연락처, 주민등록번호, 주소, 건강 정보
- 비밀 값: 비밀번호, API 키, 접속 토큰

- 조직의 기밀: 계약서, 미공개 재무 자료, 고객 목록
- 남의 저작물 전체

회사 업무 자료는 조직의 보안 규칙이 우선합니다. 규칙이 없다면 먼저 만드는 것이 좋습니다. 이 주제는 2장과 22장에서 다시 다룹니다.

연습할 때는 가짜 값으로 충분합니다. 이름은 "홍길동", 연락처는 "[연락처]"처럼 자리 표시만 넣어 명백히 가짜인 값을 씁니다.

첫 대화의 요령

AI는 내가 누구이고 무엇을 하려는지 모른 채 시작합니다. 첫 대화에서 직무, 목표, 수준을 알려 주면 이후의 답이 훨씬 맞아집니다. 모르는 용어는 한 줄로 풀어 달라고 미리 부탁해 두면 막힐 일이 줄어듭니다.

다음 실습에서 그대로 해 봅니다.

실습

실습 1. 첫 대화에서 상황 알리기

claude.ai에 로그인해서 새 대화를 열고 아래 프롬프트를 붙여 넣습니다. 대괄호는 자신의 상황으로 바꿉니다.

프롬프트

> 나는 [직무]를 맡고 있고, 코딩은 해 본 적이 [없어 / 조금 있어]. 이번 한 달 동안 AI로 [만들고 싶은 것 또는 줄이고 싶은 일]을 해 보려고 해. 내 수준에 맞춰 쉬운 말로 설명해 줘. 모르는 용어가 나오면 한 줄로 풀어서 알려 줘. 확실하지 않은 내용은 확실하지 않다고 말해 줘.

확인할 것은 두 가지입니다. 답변이 내 직무와 수준에 맞는 말투인지, 모르는 부분을 "확실하지 않다"고 구분해 주는지 봅니다.

실습 2. 한계 직접 확인하기

같은 대화에서 아래를 이어서 보냅니다.

프롬프트

> 방금 설명한 내용 중에서 사실과 다를 가능성이 있는 부분이나 네가 확신하지 못한 부분을 모두 짚어 줘. 숫자, 날짜, 이름, 고유명사 위주로 목록으로 알려 주고, 내가 어디서 확인하면 되는지도 적어 줘.

AI가 스스로 짚어 낸 항목 중 하나를 골라 공식 문서나 믿을 만한 출처에서 직접 확인합니다. 맞는지 틀린지가 아니라, 확인하는 과정을 한 번 거치는 것이 목적입니다.

실습 3. 같은 질문 두 번, 틀린 초안 한 번

먼저 새 대화 두 개에 같은 질문을 각각 보냅니다. 질문은 가상의 주제로 충분합니다.

프롬프트

> 신입 직원에게 [업무 주제]를 설명하는 세 문단짜리 안내문을 써 줘.

두 답을 나란히 놓고 어디가 달라졌는지 찾습니다. 왜 달라졌는지를 앞의 표(원리, 결과, 대응)로 설명해 봅니다.

다음으로 내 초안에 일부러 틀린 주장을 넣어 보내고, 반박하는지 동조하는지 관찰합니다.

프롬프트

> 내 초안은 [틀린 주장]을 근거로 합니다. 문제가 있으면 반박해 줘.

반박 요구를 붙이지 않고 같은 초안을 보낸 경우와 비교하면, 요구에 따라 태도가 달라지는 정도를 알 수 있습니다.

정리

- AI는 넓은 이름이고, 글을 다루는 생성형 AI의 중심에 대규모 언어 모델이 있으며, 클로드는 그중 앤트로픽의 모델과 제품군입니다.
- 클로드는 채팅, 폴더 작업, Claude Code의 세 층과 API 등 여러 곳에서 쓰며, 같은 모델이라도 쓰는 곳에 따라 할 수 있는 일이 다릅니다.
- 생성형 AI는 확률로 다음 말을 예측하므로 답이 달라지고, 최신 정보를 모르고, 틀린 말도 그럴듯하게 합니다. 원리를 알면 대응이 따라옵니다.
- 대화 상대로 쓰는 일과 일을 맡기는 일은 다르고, 맡길수록 앞에서 기준을 정하고 뒤에서 확인해야 합니다. AI가 초안, 사람이 검수, 그다음 발송입니다.

- 모델은 이름보다 등급으로 고르고, 문서에는 모델 이름을 못 박지 않습니다.
- 숫자·날짜·고유명사는 직접 확인하고, 개인정보와 비밀 값은 붙여 넣지 않습니다.

2장

요금제 판단 기준

이 장은 어떤 요금제를 써야 하는지를 숫자 없이 판단하는 방법을 다룹니다. 요금제의 가격과 사용량 한도는 자주 바뀌므로 이 책에는 적지 않았습니다. 대신 "내 사용 패턴이 어느 요금제의 목적과 맞는가"를 묻는 기준과, 결제 전에 확인할 것을 정리합니다.

요금제는 목적으로 나뉩니다

공식 요금제 페이지(claude.com/pricing)를 보면 요금제의 이름과 구성은 시기에 따라 달라집니다. 2026년 11월 기준으로는 크게 네 갈래로 이해하면 됩니다. 정확한 이름과 포함 기능은 반드시 해당 페이지에서 확인합니다.

갈래	목적	이런 사람에게
무료	써 보고 감을 잡는 용도	처음 시작하는 사람, 가끔 쓰는 사람
개인 유료	일상 업무에 꾸준히 쓰는 용도	매주 여러 번 쓰는 실무자
개인 상위	길고 무거운 작업을 자주 하는 용도	하루 종일 쓰거나 Claude Code를 많이 쓰는 사람
팀·기업	조직이 계정과 데이터를 관리하는 용도	여러 명이 같이 쓰고, 관리자가 필요한 조직

요금제 페이지에는 위 갈래 외에 교육기관용 요금제도 따로 나옵니다(2026년 11월 기준).

무료 이용은 업무용으로 쓰기에는 접근 범위와 사용량이 제한되는 경우가 많습니다. 어떤 기능과 모델이 막혀 있는지는 요금제 페이지에서 직접 확인합니다.

요금제가 다르면 쓸 수 있는 기능, 선택 가능한 모델, 한 번에 처리할 수 있는 양, 사용량 한도가 달라집니다. Claude Code는 무료 요금제에 포함되지 않고 유료 요금제에 포함된다고 안내됩니다(2026년 11월 기준). 쓰려는 기능이 내 요금제에 들어 있는지는 결제 전에 직접 확인합니다.

사용 패턴으로 판단하기

요금제는 "얼마가 적당한가"가 아니라 "내가 어떻게 쓰는가"로 정합니다. 다음 다섯 가지를 보면 대부분 정리됩니다.

1. 사용 빈도

일주일에 한두 번이라면 무료로 충분한지 먼저 확인합니다. 매일 쓰고, 쓰다가 한도에 걸려 일이 끊기는 경험을 한 번이라도 했다면 유료를 검토할 때입니다.

2. 작업의 길이

짧은 질문과 답으로 끝나는 일은 부담이 적습니다. 긴 문서를 여러 개 읽히거나, 표 데이터를 오래 분석하거나, 한 작업을 몇십 분씩 이어가면 사용량이 빨리 찹니다. 길고 무거운 작업이 일상이라면 한 단계 위를 봅니다.

3. 여러 세션을 오가는 작업

한 프로젝트에 대화창(세션)을 여러 개 열고 서브에이전트를 쓰면 사용량이 합쳐집니다. 10장의 멀티 세션 작업을 계획한다면 단일 대화로 쓸 때보다 여유 있는 요금제가 맞습니다.

4. Claude Code 사용

Claude Code는 파일을 읽고 쓰고 명령을 실행하며 여러 단계를 반복하기 때문에, 같은 시간 대화창을 쓸 때보다 사용량이 많아지는 경향이 있습니다. 코딩 도구로 바이브코딩을 본격적으로 할 계획이라면 이 항목이 요금제를 정하는 가장 큰 기준이 됩니다. 정확한 포함 여부와 한도는 Claude Code 문서와 요금제 페이지에서 확인합니다.

5. 조직 소유 계정과 데이터 관리

개인 계정으로 업무를 하면 퇴사나 담당자 교체 때 계정과 대화 기록이 함께 사라지거나 개인에게 남습니다. 회사 자료를 다룬다면 이 항목이 사용량보다 중요합니다.

- 계정은 개인 이메일이 아니라 회사 이름으로 만든 계정(회사 도메인 메일 등)으로 만듭니다.
- 여러 명이 쓴다면 관리자가 사용자와 설정을 관리할 수 있는 팀·기업 갈래를 검토합니다.
- 대화 내용의 보관과 학습 활용에 관한 설정, 보안·감사 요구사항은 요금제와 설정에 따라 다르므로 공식 문서와 약관에서 확인합니다.

> **핵심** 조직의 계정과 대화 기록은 조직의 자산입니다. 처음부터 회사 이름으로 계정을 만들어야 담당자가 바뀌어도 이어 쓸 수 있습니다.

파일 작업과 위임 작업은 사용량을 많이 씁니다. 파일을 읽고 고치거나 여러 단계를 스스로 이어 가는 일은 짧은 질문과 답보다 훨씬 많은 양을 처리하기 때문입니다. 채팅 기준으로 요금제를 골라 놓고 파일 작업을 시작하면 한도에 금방 닿습니다. 요금제를 고를 때는 지금의 대화 습관이 아니라 앞으로 맡길 일의 무게를 기준으로 봅니다.

모델과 사고 강도: 설정이 결과를 바꿉니다

같은 Claude라도 두 가지 설정에 따라 결과가 달라집니다. 하나는 모델이고, 다른 하나는 사고 강도입니다. 사고 강도는 답하기 전에 얼마나 오래 생각하게 할지를 정하는 설정입니다. 오래 생각할수록 놓치는 조건과 예외가 줄어드는 경향이 있습니다.

기본값은 모델과 환경마다 다르며, 중간 단계로 맞춰진 경우도 있습니다(2026년 11월 기준). 그래서 중요한 일에서는 기본값을 그대로 두지 말고 설정을 바꿔 볼 이유가 있습니다.

- 모델: 빠르고 가벼운 모델부터 가장 깊이 생각하는 상위 모델까지 등급이 나뉩니다. 이 책은 모델 이름을 쓰지 않고 등급으로 말합니다. 현재 제공되는 모델과 특징은 모델 개요에서 확인합니다.
- 사고 강도: 단계의 이름과 기본값, 설정하는 방법은 환경마다 다릅니다(2026년 11월 기준). Claude Code 모델 설정 문서와 앱의 모델 선택 화면에서 확인합니다.

일의 성격에 따라 이렇게 고르면 대체로 맞습니다.

작업	권고 설정	이유
문장 다듬기, 요약, 짧은 질문	기본 설정으로 충분	설정이 결과에 미치는 영향이 작습니다
보고서, 기획서, 데이터 분석	상위 모델에 사고 강도 높게	조건과 예외를 놓치면 결과가 틀어집니다
결재 문서, 규정 검토, 바이브코딩	위 설정에 서브에이전트와 대규모 검토 방식 추가	한 대화창이 감당하기 어려운 양을 나눠 맡깁니다

세 번째 행의 서브에이전트와 대규모 검토 방식은 10장에서 설명합니다.

대가도 있습니다. 모델 등급과 사고 강도를 올리면 같은 일에도 사용량을 훨씬 많이 씁니다. 그래서 요금제 선택이 중요해집니다. 위 표의 둘째, 셋째 행을 자주 하는 사람이 가벼운 요금제에 머물면 한도에 자주

닿습니다.

> **핵심** 깊이 생각하게 하는 설정은 사용량으로 값을 치릅니다. 모든 일에 켜지 말고 틀리면 곤란한 일에 씁니다.

판단 순서

아래 순서로 따라가면 대략의 결론이 나옵니다.

1. 지금 쓰는 요금제로 일주일을 쓰면서 사용 일지를 적습니다(실습 참고).
2. 한도에 걸린 횟수와 그때 하던 일을 봅니다.
3. 한도에 걸린 이유가 작업이 길어서인지, 횟수가 많아서인지, 세션을 여러 개 열어서인지 가려냅니다.
4. 이유에 맞는 대응을 고릅니다. 아래 절을 보세요.
5. 개인으로 쓰던 업무를 조직으로 옮길 시점이라면 5번 항목(조직 소유 계정)을 먼저 해결합니다.

사용량 한도에 걸렸을 때

한도의 크기와 다시 채워지는 방식은 요금제마다 다릅니다. 정확한 내용은 고객 지원 센터에서 확인합니다. 한도에 걸렸을 때 업그레이드부터 하지 말고 순서대로 시도합니다.

- 작업을 쪼갭니다. 한 번에 문서 열 개를 시키지 말고 세 개씩 나눕니다. 대화가 길어지면 새 대화를 열고 핵심만 요약해서 넘깁니다.
- 시간을 나눕니다. 무거운 작업은 한도가 다시 채워진 뒤에 시작하고, 가벼운 작업은 대기 시간에 합니다.
- 모델을 바꿉니다. 단순한 정리나 분류는 가벼운 모델로 충분합니다.
- 불필요한 파일 첨부와 긴 이전 대화를 줄입니다. 대화가 길수록 매번 처리해야 할 양이 늘어납니다.
- 그래도 반복된다면 한 단계 위 요금제를 검토합니다.

업그레이드는 한도에 걸린 날의 충동이 아니라 일지에 쌓인 근거로 결정하는 것이 좋습니다.

결제 전 체크리스트

- 쓰려는 기능(Claude Code, 프로젝트, 파일 분석, 검색 등)이 내 요금제에 들어 있는가
- 이번 달에 몇 번, 어떤 작업에서 한도에 걸렸는가
- 계정이 개인 소유인가 회사 소유인가
- 업무 자료를 넣어도 되는 계정과 설정인가
- 결제 주체가 누구이고, 영수증과 세금계산서는 어디서 받는가
- 해지와 변경은 어디서 하는가
- 요금제 페이지를 오늘 날짜로 직접 열어 보았는가

마지막 항목이 중요합니다. 블로그나 영상, 심지어 이 책의 설명도 시간이 지나면 어긋납니다. 결제는 항상 공식 페이지의 현재 내용을 보고 결정합니다.

실습

실습 1. 일주일 사용 일지 쓰기

일주일 동안 AI를 쓸 때마다 아래 표를 한 줄씩 채웁니다. 메모장이나 엑셀이면 충분합니다.

날짜	한 일	걸린 시간	한도에 걸렸나
11/10	거래처 안내 메일 초안	10분	아니오
11/11	회의록 요약 3건	25분	아니오

위 두 줄은 가상의 예입니다. 일주일 뒤 아래 프롬프트에 일지를 붙여 넣고 분석을 맡깁니다.

프롬프트

> 아래는 내가 일주일 동안 AI를 쓴 기록이야. 사용 빈도, 작업의 길이, 한도에 걸린 이유를 분석해 줘. 그다음 다음 네 가지 중 어디에 가장 가까운지 근거와 함께 알려 줘: 무료로 충분함 / 개인 유료 / 개인 상위 / 팀·기업. 요금제 가격이나 한도 숫자는 말하지 마. 내가 요금제 페이지에서 직접 확인해야 할 항목을 목록으로 알려 줘.
>
> [일지 붙여넣기]

AI의 판단은 참고용입니다. 최종 결정은 공식 요금제 페이지를 보며 직접 합니다.

실습 2. 조직 계정 점검표 만들기

회사에서 AI를 쓰고 있거나 쓸 예정이라면 아래 프롬프트로 점검표를 만듭니다.

프롬프트

> 우리 회사 [직원 수 규모] 팀에서 AI 도구를 업무에 쓰려고 해. 계정을 개인이 각자 만들면 생기는 문제를 정리하고, 회사 이름으로 계정을 만들 때 미리 정해야 할 것을 점검표로 만들어 줘. 담당자, 사용 가능한 자료의 범위, 넣으면 안 되는 자료, 대화 기록의 보관 방침을 포함해 줘. 법률적 판단이 필요한 부분은 전문가에게 확인하라고 표시해 줘.

실습 3. 설정에 따른 결과 비교(선택)

같은 보고서 한 건으로 설정에 따라 결과가 어떻게 달라지는지 관찰합니다. 효과가 있다고 단정하지 말고, 내 업무에서 실제로 차이가 나는지 확인하는 과제입니다. 민감한 내용이 없는 보고서나 가상의 보고서를 씁니다.

1. 기본 설정으로 아래 프롬프트를 실행하고 결과를 저장합니다.
2. 가장 깊이 생각하는 상위 모델에 사고 강도를 높여 같은 프롬프트를 실행합니다.
3. 두 결과를 원문과 대조해, 빠진 조건이나 예외가 몇 개인지 각각 셉니다.

프롬프트

> 아래 보고서를 요약하고, 요약에서 빠진 조건이나 예외가 있는지 스스로 점검 목록으로 만들어 줘.
>
> [보고서 붙여넣기]

차이가 작다면 그 업무에는 높은 설정이 필요 없다는 뜻입니다. 사용량을 아끼는 근거로 씁니다.

정리

- 요금제는 가격이 아니라 사용 패턴과 목적으로 고르며, 이름과 구성은 공식 요금제 페이지에서 확인합니다.
- 판단 기준은 사용 빈도, 작업 길이, 멀티 세션, Claude Code 사용, 조직 소유 계정 다섯 가지입니다.
- 한도에 걸리면 작업을 쪼개고, 시간을 나누고, 모델을 바꿔 본 뒤에 업그레이드를 검토합니다.
- 업무에 쓰는 계정은 처음부터 회사 이름으로 만들어 담당자가 바뀌어도 기록이 남게 합니다.
- 결제 전에는 일주일 사용 일지와 체크리스트로 근거를 확보합니다.

- 파일 작업과 깊은 사고 설정은 사용량을 많이 쓰므로, 채팅 기준으로 요금제를 고르지 않습니다.

3장

프롬프트와 메타 프롬프팅

프롬프트는 AI에게 하는 지시문입니다. 이 장은 지시문에 무엇을 담아야 하는지, 길고 중요한 작업에 쓰는 4층 구조는 어떤 모양인지, 그리고 지시문 자체를 AI에게 설계시키는 메타 프롬프팅은 어떻게 하는지 설명합니다.

좋은 지시문의 기본 세 가지

같은 요청이라도 지시문에 따라 결과가 크게 달라집니다. 기본은 세 가지를 적는 것입니다.

맥락

AI는 내가 누구이고, 이 일이 왜 필요한지 모릅니다. 직무, 목적, 읽는 사람, 지금까지의 사정을 한두 문장으로 알려 줍니다. "고객에게 보낼 안내문"과 "내부 공유용 메모"는 말투도 분량도 다릅니다.

제약

하지 말아야 할 것, 지켜야 할 조건을 적습니다. 분량, 말투, 쓰면 안 되는 표현, 근거로 삼을 자료의 범위가 해당됩니다. 제약이 없으면 AI는 빈칸을 자기 추측으로 채웁니다. "글에 없는 내용은 추측해서 쓰지 마"는 가장 자주 쓰는 제약 중 하나입니다.

출력 형식

결과를 어떤 모양으로 받고 싶은지 말합니다. 표, 번호 목록, 세 문단, 파일 이름까지 정합니다. 형식이 정해지면 받은 뒤에 고치는 일이 줄어듭니다.

가상의 예로 비교해 보겠습니다.

프롬프트

> 회의 내용 요약해 줘.

위 문장은 짧지만 AI가 길이, 대상, 형식을 모두 짐작해야 합니다. 아래처럼 바꾸면 어긋날 곳이 줄어듭니다.

프롬프트

> 나는 영업팀 팀장이고, 아래는 주간 회의 메모야. 회의에 못 온 팀원에게 공유할 요약을 만들어 줘. 결정된 일, 담당자가 있는 할 일, 아직 정해지지 않은 일 세 가지로 나눠 번호 목록으로 써 줘. 전체 열 줄 이내로 하고, 메모에 없는 내용은 추측해서 쓰지 마.
>
> [회의 메모 붙여넣기]

한 줄 요청, 구획 프롬프트, 메타 프롬프트

같은 일을 세 가지 수준으로 시켜 보겠습니다. 가상의 예로, 교육 참가가 확정된 사람들에게 보낼 안내 메일입니다. 먼저 한 줄 요청입니다.

프롬프트

> 나는 [부서] 담당이야. [월]에 임직원 [인원]명 대상 교육을 [회차]회차로 진행해. 참가 확정자에게 보낼 안내 메일을 써줘.

상황과 요청만 있으므로 길이, 말투, 들어갈 정보를 AI가 짐작해야 합니다. 조건을 한 줄 덧붙이면 어긋남이 줄어듭니다.

프롬프트

> (위 요청에 이어서) 5문장 이내, 존댓말로 쓰고, 일정·장소·준비물이 반드시 들어가야 해. 제목도 같이 제안해줘.

조건이 늘어나면 역할, 배경, 수신자, 조건을 구획으로 나눈 프롬프트가 읽기 쉽습니다.

프롬프트

> [역할]
> 너는 [업종]의 [부서]에서 일하는 사내 커뮤니케이션 전문가야.
>
> [배경과 목적]
> [월]에 임직원 [인원]명 대상 [회차]회차 교육을 진행해. 참가가 확정된 구성원에게 일정을 안내하고 참석을 돕는 메일을 쓰려고 해.
>
> [수신자]
> 바쁜 일과 중에 교육에 참여하는 구성원.
>
> [조건과 제약]
> 1. 어조: 정중한 비즈니스 존댓말
> 2. 분량: 인사말과 본문은 5문장 이내(정보 목록은 문장 수에서 제외)
> 3. 필수 포함 정보: 일정([일시]), 장소([장소]), 준비물([준비물])

이 구획 프롬프트도 직접 쓸 필요가 없습니다. "이 메일을 잘 쓰게 하는 프롬프트를 역할, 배경, 수신자, 조건으로 나눠서 설계해줘. 필요한 정보는 먼저 물어봐"라고 AI에게 시키면 됩니다. 이것이 아래에서 다루는 메타 프롬프팅입니다.

변수 자리 표기

위 예의 [일시], [장소], [준비물]처럼 바뀌는 부분을 대괄호 자리표시자로 비워 두는 습관을 들입니다. 같은 프롬프트를 회차나 부서만 바꿔 다시 쓸 수 있고, 무엇을 채워야 하는지도 한눈에 보입니다.

공식 문서도 명확한 지시, 예시, XML 구조화 같은 기법을 정리해 두었습니다. 자세한 내용은 프롬프트 작성 모범 사례를 참고합니다(2026년 11월 기준).

4층 프롬프트 구조

짧은 질문에는 위 세 가지면 충분합니다. 하지만 길고 중요한 작업, 대화가 길어지는 작업은 지시를 놓치거나 결과가 매번 달라지기 쉽습니다. ACTIGENCE의 사용설명서는 이런 작업을 위해 프롬프트를 4층으로 쌓는 방식을 씁니다. 이를 GMP(General Meta-Prompt Protocol)라고 부릅니다. 어렵게 들리지만 "좋은 프롬프트는 이 네 가지를 갖춘다"는 체크리스트입니다.

층	이름	하는 일
1층	XML 계층	지시를 섹션으로 나눠 구분
2층	이중 앵커링	핵심 제약을 맨 위와 맨 아래에 두 번 적음
3층	우선순위 라벨	[MUST], [SHOULD], [NICE]로 양보 순서 표시
4층	자가검증	출력 직전에 정확성·완전성·일관성을 스스로 점검

1층: XML 계층

지시문 안에서 역할이 다른 내용을 `<CRITICAL_CONSTRAINTS>`, `<INSTRUCTIONS>` 같은 이름표(태그)로 나눕니다. 제약, 지시, 자료, 출력 형식이 한 문단에 섞이지 않아서 AI가 어느 부분이 무엇인지 헷갈리지 않습니다. 태그를 코드처럼 쓸 필요는 없고, 이름표가 붙은 상자를 만든다고 생각하면 됩니다.

2층: 이중 앵커링

핵심 제약을 맨 위에 쓰고, 맨 아래 `<FINAL_REMINDER>`에서 한 번 더 씁니다. 대화나 지시문이 길어지면 초반의 조건을 잊기 쉽기 때문입니다. 닻을 두 군데 내려 두는 셈입니다.

3층: 우선순위 라벨

모든 조건이 같은 무게는 아닙니다. 지키지 않으면 결과가 무효인 것은 [MUST], 되도록 지킬 것은 [SHOULD], 여유가 있으면 할 것은 [NICE]로 표시합니다. 조건이 서로 부딪힐 때 AI가 무엇을 양보할지 알게 됩니다.

4층: 자가검증

답을 내놓기 전에 한 번 더 확인하라고 시키는 단계입니다. 19장에서 자세히 다루고, 이 장 끝에서 가장 간단한 형태를 소개합니다.

언제 쓰는가

모든 작업에 4층을 다 쓰지 않습니다. 작업의 복잡도가 프롬프트의 복잡도를 정합니다.

- 한두 문장으로 끝나는 질문: 맥락, 제약, 형식만으로 충분합니다.
- 매주 반복하는 업무, 외부에 나가는 문서: 1~3층까지 쓰면 안정됩니다.
- 길고 결과가 중요한 작업, 여러 번 재사용할 지시문: 4층을 모두 씁니다.

> **핵심** 프롬프트에도 적정 깊이가 있습니다. 쉬운 일에 4층을 쓰면 번거롭기만 하고, 어려운 일에 한 줄만 쓰면 결과가 매번 달라집니다.

지시가 어긋날 때

떠넘기기에는 세 번 더 시키기

일을 시켰는데 AI가 직접 하지 않고 "방법을 안내해 드리겠습니다"라며 단계별 사용법을 늘어놓는 경우가 있습니다. 친절해 보이지만 일은 하나도 하지 않은 상태입니다. 이때는 세 번 더 시킵니다.

1. 다시 시킵니다.

프롬프트

> 아니, 네가 직접 해봐.

2. 할 수 있다고 전제합니다.

프롬프트

> 너 할 수 있잖아. 어떻게든 해봐.

3. 필요한 것을 묻습니다.

프롬프트

> 정 안 되면, 네가 직접 하려면 뭐가 필요한지 말해.

처음에는 AI가 직접 할 수 있다고 가정하고 시작하는 편이 낫습니다. 다만 실제로 못 하는 일을 억지로 시키면 그럴듯한 거짓 결과가 나올 수 있습니다. 3번이 이를 막는 안전장치입니다. 이 답이 폴더 접근, 권한, 도구 연결 같은 설정으로 이어지는 경우가 많으며, 설정 방법은 8장부터 12장까지에서 다룹니다.

> **핵심** 사용법 안내가 돌아오면 속지 말고 끝까지 시킵니다. 세 번째 재촉에서 일이 시작되거나, 필요한 것이 무엇인지 드러납니다.

되묻게 하기

한 번에 완성된 지시문을 쓰려고 애쓰기보다 AI가 먼저 묻게 하는 방법도 있습니다. 답하는 동안 머릿속에만 있던 조건이 정리됩니다. 규칙은 네 가지입니다.

- 객관식으로 묻게 합니다. 보기에 없으면 직접 씁니다.
- 한 번에 하나씩 묻게 합니다.
- "모르겠다"고 하면 AI가 안을 제시하게 합니다.
- 질문이 끝나면 그때 결과를 쓰게 합니다.

프롬프트

> [주제]에 관한 [결과물]을 만들 거야. 바로 쓰지 말고, 쓰는 데 꼭 필요한 것을 나에게 먼저 물어봐.
> - 충분히 구체화될 때까지 계속 물어봐
> - 내가 고르기만 하면 되게 객관식으로, 한 번에 하나씩 물어봐. 보기에 없으면 직접 쓸게
> - 내가 모르겠다고 하면 네가 안을 제시해줘
> - 질문이 끝나면 그때 결과를 써줘

이 방법을 문서 작업 전체에 적용한 흐름은 16장에서 볼 수 있습니다.

되묻게 하기의 변형

질문 방식은 상황에 따라 바꿀 수 있습니다. 생각이 정리되지 않았다면 한 번에 하나씩 묻는 방식이 편하고, 답이 빠르게 나오는 일이라면 질문을 번호로 한꺼번에 받는 방식이 빠릅니다. 이때는 "5개 이하로 번호를 붙여 한꺼번에 물어봐 줘. 내가 답하기 전에는 결과를 쓰지 마"라고 시킵니다.

되묻게 하기의 결과물은 글뿐 아니라 다른 AI에게 붙여 넣을 요청문일 수도 있습니다. 화면을 만드는 AI에게 줄 요청문을 먼저 쓰게 하는 방법은 5장에서, 긴 PRD를 열 줄 안팎의 구현용 요청문으로 줄이게 하는 방법은 6장에서 실습합니다. 요청문을 먼저 보여 달라고 하고 확인한 뒤에 만들게 하면, 어긋난 방향을 일찍 잡을 수 있습니다.

합의서 한 장

지시문과 별개로, 작업을 맡기기 전에 짧은 문서 한 장을 먼저 써 두는 방법이 있습니다.
ACTIGENCE에서는 이를 "합의서"(헌법)라고 부릅니다. 건축주와 건축가가 도면을 먼저 맞추고 짓는 것과 같습니다. 내용은 세 가지입니다.

- 목표: 이번 작업에서 결국 얻고 싶은 것

- 제약: 절대 깨면 안 되는 규칙
- 우선순위: 충돌할 때 무엇을 먼저 챙길지

예를 들어 보고서를 맡길 때 "목표는 경영진 5분 보고, 제약은 출처 없는 숫자 금지, 우선순위는 정확성이 분량보다 먼저"라고 적어 두는 식입니다. 이 합의서는 AI가 길을 잃을 때 돌아올 기준이 되고, 사람이 판단에 지쳤을 때의 기준도 됩니다. Claude Code에서는 이 문서를 프로젝트 안의 파일로 두고 AI가 매번 읽게 합니다. 자세한 방법은 8장에서 다룹니다.

메타 프롬프팅: AI에게 지시문을 설계시키기

메타 프롬프팅은 프롬프트를 사람이 처음부터 쓰지 않고, AI에게 "이 일을 시키려면 어떤 지시문이 좋을지" 설계하게 하는 방법입니다. 지시문을 잘 쓰는 데 시간이 걸리는 사람, 매번 같은 품질을 원하는 사람에게 효과적입니다.

과정은 이렇습니다.

1. 하려는 일을 있는 그대로 설명합니다. 서툴러도 괜찮습니다.
2. AI가 부족한 정보를 질문하게 합니다.
3. 답이 모이면 AI가 재사용 가능한 지시문 한 편을 씁니다. 바뀌는 부분은 [대괄호] 빈칸으로 둡니다.
4. 그 지시문으로 실제 일을 한 번 시켜 보고, 어긋난 곳을 지시문에 반영해 고칩니다.

이 과정을 도와주는 스킬이 GMP입니다. 스킬은 AI에게 설치해 두는 전문 작업 안내서입니다(11장). GMP 스킬은 프롬프트나 SKILL.md 파일을 위의 4층 구조로 설계하고 개선합니다. "프롬프트 만들어줘", "이걸 프롬프트로" 같은 요청에 반응하도록 만들어져 있고, 내용은 부록 B에 있습니다.

스킬을 쓰지 않아도 됩니다. 일반 대화에서도 실습 2의 프롬프트로 같은 과정을 밟을 수 있습니다.

잘 통한 과정을 파일로 저장하기

되묻기와 수정을 거쳐 잘 된 작업이 있다면 그 과정을 다음에도 쓸 수 있게 프롬프트 파일로 저장합니다.

프롬프트

> 방금 우리가 한 [작업명] 과정을 다른 주제에도 쓸 수 있는 프롬프트로 만들어줘.
> - 주제만 바꿔 넣으면 되게 [ ] 자리를 비워 둬
> - 네가 나에게 물어봐야 할 질문 목록도 그 안에 넣어줘
> - 상위 모델에게 입력하는 거니까 AI의 자율성을 과도하게 제한하지 않는 선에서 써줘
> - [파일명].md 로 저장해줘

이 파일은 새 일을 시작할 때 폴더에 복사해 가는 것만으로 같은 과정을 되풀이하게 해 줍니다. 제안서 기획과 결과보고서에서 실제로 쓰는 모습은 16장과 17장에 있습니다.

실습

실습 1. 같은 일을 두 가지 지시문으로 시켜 보기

요약하거나 초안을 쓸 일 하나를 고릅니다. 먼저 한 줄로 시켜서 결과를 받고, 같은 일을 맥락, 제약, 출력 형식을 넣어서 다시 시킵니다. 두 결과를 나란히 두고 어느 쪽이 덜 고쳐도 되는지 비교합니다. 아래는 두 번째 시도의 틀입니다.

프롬프트

> 나는 [직무]이고, 아래 [자료 종류]를 [읽는 사람]에게 [목적]으로 보내려고 해.
> 조건:
> 1. 분량은 [분량]
> 2. 말투는 [정중하게 / 간결하게]
> 3. 자료에 없는 내용은 추측해서 쓰지 마. 확인이 필요한 곳은 [확인 필요]로 표시해
> 형식: [번호 목록 / 표 / 세 문단]
>
> [자료 붙여넣기]

실습 2. 메타 프롬프트로 반복 업무 지시문 만들기

매주 반복하는 일 하나를 고릅니다. 아래 프롬프트로 AI에게 지시문 설계를 맡깁니다.

프롬프트

> 나는 [직무]이고, 매주 [반복하는 일]을 해. 이 일을 AI에게 시킬 때 쓸 재사용 지시문을 만들고 싶어.
> 먼저 필요한 정보를 질문해 줘. 질문은 한 번에 하나씩 해 줘.
> 답이 모이면 다음 구조로 지시문을 써 줘:
> 1. 맥락 2. 지켜야 할 제약(중요도를 [MUST], [SHOULD], [NICE]로 표시) 3. 출력 형식 4. 매번 바뀌는 부분(대괄호 빈칸으로 표시) 5. 출력 전에 스스로 점검할 질문 3개
> 맨 위와 맨 아래에 핵심 제약 한 줄씩을 반복해 줘.

받은 지시문으로 실제 자료 하나를 시켜 보고, 어긋난 부분을 알려 주며 지시문을 고칩니다. 마음에 들면 파일로 저장해서 다음 주에 다시 씁니다.

실습 3. 한 줄 요청과 구획 프롬프트 비교

자기 업무의 메일 하나를 고릅니다. 한 줄 프롬프트와 구획 프롬프트로 각각 시키고 결과를 비교합니다. 어느 쪽이 덜 고쳐도 되는지, 구획 프롬프트의 어느 조건이 차이를 만들었는지 적습니다.

실습 4. 떠넘기기에 3단 재촉 적용하기

일부러 사용법 안내가 돌아올 만한 요청을 해 봅니다. 세 가지 재촉을 순서대로 적용하면서 몇 번째에서 일이 시작되는지 기록합니다. 3단계까지 가도 안 되면 AI가 말한 필요한 것을 정리해 둡니다.

실습 5. 자가검증 질문 붙이기

기존 지시문 끝에 아래 문장을 붙여 봅니다.

프롬프트

> 답을 내기 전에, 네 답에서 정말 맞아야 하는 핵심 주장 세 개를 골라 각각 맞는지 따로 점검해 줘. 어긋나는 것이 있으면 고친 뒤 최종 답만 보여 주고, 무엇을 고쳤는지 한 줄로 알려 줘.

붙이기 전후의 결과를 비교해서 달라진 점을 적어 봅니다.

정리

- 좋은 지시문의 기본은 맥락, 제약, 출력 형식 세 가지입니다.
- 길고 중요한 작업은 4층(XML 계층, 이중 앵커링, 우선순위 라벨, 자가검증)으로 쌓고, 쉬운 일에는 얕게 씁니다.

- 합의서는 목표, 제약, 우선순위를 담은 한 장짜리 문서이며 9장에서 이어집니다.
- 메타 프롬프팅은 AI에게 지시문을 설계시키는 방법이고, 결과는 실제로 한 번 돌려 보며 고칩니다.
- 한 줄 요청, 구획 프롬프트, 메타 프롬프트로 같은 일을 시켜 보고, 바뀌는 부분은 [대괄호]로 비워 둡니다.
- AI가 사용법만 안내하면 다시 시키고, 할 수 있다고 전제하고, 필요한 것을 묻는 세 번의 재촉을 씁니다.
- 출력 직전의 자가검증 질문 하나만 붙여도 결과가 달라질 수 있습니다. 자세한 방법은 19장에 있습니다.

4장

바이브코딩

바이브코딩은 만들고 싶은 것을 말로 설명하면 AI가 코드를 쓰고, 사람이 결과를 확인하고 고칠 점을 말하는 작업 방식입니다. 이 장은 무엇이 잘 되고 무엇이 어려운지, 작게 시작해서 반복하는 요령, 한 장짜리 PRD, 입문 과정 4회차의 흐름, 흔한 실수를 다룹니다.

바이브코딩이란

코드를 직접 한 줄씩 쓰는 대신, 원하는 결과를 자연어로 설명합니다. AI가 코드를 만들고, 사람은 그것을 실행해서 눈으로 확인합니다. 마음에 들지 않으면 "이 부분만 바꿔 줘"라고 말합니다. 이 순환이 바이브코딩의 전부입니다.

여기서 사람이 맡는 일은 세 가지입니다.

- 무엇을 만들지 정합니다.
- 결과가 맞는지 확인합니다.
- 다음에 무엇을 고칠지 결정합니다.

코드를 몰라도 시작할 수 있지만, 판단과 책임이 사람에게 있다는 점은 변하지 않습니다. 코드를 이해하지 못한 채 AI가 만든 결과를 그대로 외부에 공개하거나 고객 데이터를 다루게 하면 문제가 생깁니다. 이 장의 마지막 절과 20장에서 이 위험을 다룹니다.

잘 되는 일과 어려운 일

잘 되는 일

- 한 장짜리 소개 페이지, 안내 페이지처럼 눈에 보이는 결과물
- 입력 폼, 목록, 간단한 계산기 같은 작은 화면
- 엑셀·CSV 정리, 파일 이름 일괄 변경 같은 반복 작업을 처리하는 짧은 스크립트
- 기존 결과물의 문구, 색, 배치 수정

공통점은 결과가 바로 눈에 보이고, 틀려도 되돌리기 쉬우며, 범위가 작다는 점입니다.

정리하면 내부 도구, 우리 팀 몇 명이 쓰는 수준, 빠르게 만들고 고치는 일이 금방 됩니다. 고객에게 파는 상용 서비스, 많은 사용자가 동시에 쓰는 시스템, 보안과 결제가 얽힌 민감한 시스템은 금방 되지 않습니다.

업무를 아는 사람이 더 잘 만듭니다

바이브코딩은 코딩을 아는 사람의 일이 아니라 업무를 아는 사람의 일에 가깝습니다. 무엇이 불편한지, 결과가 맞는지 틀린지를 판단하는 것은 업무를 아는 사람이기 때문입니다. 작업 방식의 차이는 아래 표와 같습니다.

구분	순서
예전 방식	문법을 배운다, 코드를 쓴다, 오류를 고친다
바이브코딩	만들어 달라고 한다, AI가 만든다, 고쳐 달라고 한다

어려운 일

- 결제, 개인정보 보관, 로그인처럼 보안과 법적 책임이 걸린 기능
- 여러 사람이 동시에 쓰는 서비스, 대용량 데이터
- 기존 시스템과 복잡하게 얽힌 변경
- 정답이 없어서 "맞는지 확인"하는 기준을 사람이 세우기 어려운 일

어렵다고 해서 못 하는 것은 아닙니다. 다만 혼자 AI만 믿고 진행하지 말고 19장의 계획과 자가검증을 거치고, 필요하면 전문가의 검토를 받아야 합니다.

작게 시작해서 반복하기

처음부터 완성품을 한 번에 시키면 실패합니다. 결과가 어긋났을 때 어디가 잘못인지 알 수 없기 때문입니다. 대신 이렇게 합니다.

1. 오늘 안에 끝낼 수 있는 크기로 줄입니다.
2. 요청합니다.
3. 결과를 직접 열어서 확인합니다.
4. 마음에 들지 않는 곳 하나만 고치도록 요청합니다.
5. 3번과 4번을 되풀이합니다.

요청, 확인, 수정의 고리를 짧게 돌릴수록 빠르고 안전합니다. 한 번에 한 가지만 바꾸게 하면 무엇이 효과가 있었는지도 알 수 있습니다.

한 번에 하나씩 고치라는 원칙에는 예외가 있습니다. 사용량 한도가 있는 도구에서는 요청 횟수가 곧 자원입니다. 이때는 고칠 점을 메모해 모았다가 한 번에 보냅니다. 서로 영향을 주지 않는 가벼운 수정은 모아서, 무엇이 문제를 만들었는지 모를 때는 하나씩 확인한다고 구분하면 됩니다.

수정하기 전에 지금 상태를 남겨 두는 것도 중요합니다. 파일을 복사해 두거나, 버전 기록 도구(git)로 기록을 남기면 결과가 나빠졌을 때 되돌릴 수 있습니다. 되돌릴 수 있으면 과감하게 시도할 수 있습니다.

> **핵심** 한 번에 하나씩 고치고, 고치기 전 상태를 남겨 둡니다. 이 두 가지만 지켜도 대부분의 사고가 작은 일로 끝납니다.

만들기 전에 한 장짜리 PRD

PRD는 제품 요구사항 문서(Product Requirements Document)입니다. 무엇을 누구를 위해 왜 만드는지, 어디까지 하고 무엇은 하지 않을지 적은 짧은 문서입니다. 처음 만드는 사람을 위한 다섯 칸짜리 줄인 형태와 요청문에서 PRD로 자라는 과정은 6장에서 다룹니다. 바이브코딩에서 PRD는 AI와 맺는 약속입니다. 9장의 합의서와 같은 역할을 만드는 작업 단위로 옮겨 놓은 것입니다.

작은 작업이라면 한 장이면 충분합니다. 담을 내용은 다음과 같습니다.

- 문제: 어떤 불편을 해결하려는가
- 사용자: 누가 쓰는가
- 성공 기준: 무엇이 되면 성공인가
- 하지 않을 것: 이번에는 만들지 않을 것
- 제약: 기한, 쓸 수 있는 도구, 지켜야 할 규칙
- 화면: 어떤 화면이 필요한가

ACTIGENCE에는 PRD 작성을 돕는 PRD-Vibecoding 스킬이 있습니다. 한 줄짜리 아이디어를 주면 필요한 항목을 추론하고, 사용자가 정해야만 하는 것만 객관식으로 묻고, 나머지는 스스로 채워 PRD 한 편을 만듭니다. 이 스킬이 만드는 PRD는 처음부터 끝까지 한 번에 만드는 원샷 프로토타이핑 작업이 그대로 읽을 수 있는 형식입니다. 스킬 설명은 부록 B에 있습니다.

PRD는 이 도구의 단일 원본(SSOT, Single Source of Truth)입니다. 도구에 대한 판단이 엇갈릴 때 돌아가 확인하는 하나뿐인 원본이며, 범위를 묻는 사람과 일을 이어받는 사람에게 건네는 문서이기도 합니다. SSOT는 7장에서 자세히 다룹니다. 파일은 워드가 아니라 PRD.md로 저장합니다. AI가 가장 잘 읽는 형식이기 때문이며, 사람이 읽을 때만 보기 편한 HTML로 바꿔 봅니다.

준비 없이 "만들어 줘"로 시작하면 요청마다 방향이 흔들립니다. PRD를 먼저 쓰면 방향이 고정되고, 고치는 것이 아니라 쌓입니다.

스킬 없이도 아래 실습 2의 프롬프트로 한 장짜리 PRD를 만들 수 있습니다.

입문 과정 4회차의 흐름

ACTIGENCE의 AI 바이브코딩 입문 과정은 4주, 4회차로 구성됩니다. 회차별 실습은 actigence.ai 자료실에 있고, 이 책에서는 흐름을 요약합니다.

1회차. 도구를 준비하고 첫 결과물 만들기

AI에게 상황(직무, 목표, 수준)을 먼저 알려 주고, 작업 폴더를 만들고, 설치 중 막히면 화면 메시지를 그대로 붙여서 묻습니다. 그리고 한 장짜리 웹 페이지를 index.html 하나로 만들어 브라우저에서 엽니다. 무엇으로 이루어진 파일인지도 AI에게 설명을 듣습니다. 마지막에 수정하기 전 상태를 남겨 두는 방법을 익힙니다.

2회차. 화면을 말로 설명하기

바로 만들지 않고 AI가 질문을 하나씩 하게 해서 화면 구성안을 만듭니다. 꼭 필요한 것과 나중에 해도 되는 것으로 나눕니다. 그다음 버튼, 입력 폼, 목록을 차례로 만듭니다. 버튼은 "누르면 무슨 일이 일어나는지"까지 말하고, 목록은 보여 주기, 추가하기, 지우기 순서로 나눠 확인합니다. 휴대폰처럼 좁은 화면에서도 점검합니다.

3회차. 반복 업무 맡기기

AI에게 맡길 일과 사람이 남겨 둘 일을 구분합니다. 엑셀·CSV 정리를 시키되, 먼저 점검만 시키고 수정은 그다음에 합니다. 결과는 원본과 줄 수와 값을 비교해서 확인합니다. 메일 초안은 날짜와 금액을 [확인 필요]로 표시하게 하고, 사람이 직접 확인합니다. 이 회차의 데이터 정리는 14장에서 더 깊이 다룹니다.

4회차. 점검하고 웹에 올리기

공개 전에 개인정보, 비밀번호, 내부 자료가 파일에 들어 있지 않은지 점검합니다. README.md로 설명 문서를 남기고, 호스팅 서비스의 공식 문서를 보며 웹에 올립니다. 올린 뒤에는 다른 기기에서 열어 봅니다. 막혔을 때 묻는 틀(하려던 일, 실제로 일어난 일, 이미 해 본 것, 오류 메시지 원문)도 배웁니다. 이 흐름은 20장에서 더 확장됩니다.

설치 없이 브라우저만으로 시작하는 짧은 경로는 5장과 6장에 따로 정리했습니다. 입문 과정을 이틀로 압축한 흐름이며, 요청문에서 시작해 PRD, 파일, 주소로 이어집니다.

네 회차에 공통으로 흐르는 원칙은 하나입니다. AI가 만든 결과는 시작점이고, 확인은 사람이 합니다.

흔한 실수

- 처음부터 크게 시킵니다. 로그인, 결제, 관리자 화면까지 한 번에 요청하면 어디가 잘못인지 알 수 없습니다. 눈에 보이는 가장 작은 것부터 시작합니다.
- 확인 없이 다음으로 넘어갑니다. 결과를 열어 보지 않고 다음 요청을 쌓으면 오류가 겹쳐서 되돌리기 어려워집니다.
- 한 번에 여러 가지를 바꾸게 합니다. 무엇이 문제를 만들었는지 모르게 됩니다.
- 수정 전 상태를 남기지 않습니다. 결과가 나빠졌을 때 돌아갈 곳이 없습니다.
- 오류 메시지를 요약해서 전달합니다. 메시지는 요약하지 말고 그대로 붙입니다.
- 비밀 값을 파일에 넣습니다. 비밀번호와 접속 키는 대화에도, 공개되는 파일에도 넣지 않습니다.
- 실제 고객 정보로 연습합니다. 연습은 가짜 값으로 합니다.
- 검토가 새 병목인 것을 잊습니다. 만들기가 쉬워진 만큼 이해하고 확인하는 일이 늘어납니다. 내용을 이해하지 못한 결과물을 남에게 넘기면 일이 사라진 것이 아니라 받는 사람에게 넘어갑니다. 19장의 워크슬롭 절에서 이어서 다룹니다.
- 허락 창을 읽지 않고 누릅니다. 파일 삭제나 내려받기, 작업 폴더 밖의 이야기가 나오면 멈추고 확인합니다.
- 서비스마다 다른 계정으로 로그인합니다. 다른 계정으로 승인하면 빈 계정이 새로 생겨 올린 것이 보이지 않습니다.
- 만든 것을 이해하지 못한 채 공개합니다. 무엇을 하는 파일인지 AI에게 설명을 듣고, 공개 전 점검을 거칩니다.

실습

실습 1. 개인 소개 페이지 첫 프롬프트

작업 폴더를 하나 만들고, 클로드에서 아래 프롬프트로 시작합니다. 대괄호는 내용으로 바꿉니다. 입문 과정 1회차의 실습을 옮긴 것입니다.

프롬프트

> [주제: 나 자신 / 우리 가게 / 우리 팀]를 소개하는 한 장짜리 웹 페이지를 index.html 파일 하나로 만들어 줘. 제목, 두세 문장의 소개, 연락 방법을 넣어 줘. 글자는 읽기 편한 크기로 하고, 색은 [차분한 / 밝은] 느낌으로 맞춰 줘. 연락처 칸에는 실제 번호 대신 [연락처]라는 자리 표시만 넣어 줘. 만든 뒤에 파일을 어떻게 열어 보는지도 알려 줘.

파일을 열어 보고, 마음에 들지 않는 부분 하나만 골라 고쳐 달라고 요청합니다. 그 한 가지를 바꾸기 전에 파일을 복사해 둡니다.

실습 2. 한 장짜리 PRD 만들기

만들고 싶은 작은 도구(예: 방문 예약 폼, 할 일 목록)가 있다면 아래 프롬프트로 PRD를 만들어 봅니다.

프롬프트

> 나는 [직무]이고 [만들고 싶은 것]을 만들려고 해. 코딩은 [못 해 / 조금 해].
> 한 장짜리 PRD를 쓰고 싶어. 먼저 필요한 것을 질문해 줘. 질문은 한 번에 하나씩, 객관식으로 해 줘.
> 답이 모이면 다음 항목으로 PRD를 써 줘: 문제, 사용자, 성공 기준, 하지 않을 것, 제약, 필요한 화면 목록.
> '꼭 필요한 것'과 '나중에 해도 되는 것'을 나누고, 꼭 필요한 것은 오늘 안에 만들 수 있는 양으로 줄여 줘.

완성된 PRD에서 "하지 않을 것" 항목을 가장 먼저 읽어 봅니다. 여기가 비어 있으면 범위가 커지고 있다는 신호입니다.

실습 3. 작업 기록 남기기

하루 작업을 마치면 아래 프롬프트로 기록을 남깁니다.

프롬프트

> 오늘 우리가 한 작업을 notes.md 파일로 정리해 줘. 만든 파일 이름, 내가 쓴 지시 중 잘 먹힌 것, 막혔던 점, 다음에 해 보고 싶은 것을 항목별로 적어 줘.

정리

- 바이브코딩은 말로 설명하고, AI가 만들고, 사람이 확인하고 결정하는 작업 방식입니다.
- 눈에 보이고 되돌리기 쉬운 작은 일은 잘 되고, 보안과 법적 책임이 걸린 일은 어렵습니다.
- 요청, 확인, 수정의 고리를 짧게 돌리고, 한 번에 하나만 고치며, 수정 전 상태를 남겨 둡니다.
- 만들기 전에 한 장짜리 PRD로 문제, 성공 기준, 하지 않을 것을 정해 둡니다.
- 입문 과정 4회차는 도구 준비, 화면 구성, 반복 업무, 점검과 공개 순서로 이어지며, 공통 원칙은 확인을 사람이 한다는 것입니다.

5장

설치 없이 시작하는 첫 화면

이 장은 프로그램을 설치하지 않고 브라우저만으로 화면 하나를 만들어, 내 휴대폰에서 열어 보는 데까지 가는 길을 다룹니다. 말로 부탁하는 방식에서 "만들게 하는" 방식으로 넘어가는 첫 번째 실습이며, 완전 초보자 대상 이틀 과정의 첫째 날 흐름을 옮긴 것입니다.

물어보기에서 만들게 하기로

지금까지 AI를 쓰는 방식은 대체로 물어보는 것이었습니다. "이번 주 할 일을 정리하는 방법을 알려 줘"라고 하면 글이 돌아옵니다. 이 장에서는 같은 일을 이렇게 바꿉니다. "이번 주 할 일을 날짜별로 한눈에 보는 화면을 만들어 줘." 이제 돌아오는 것은 글이 아니라 눈으로 볼 수 있는 화면입니다.

4장에서 본 순환이 그대로 적용됩니다. 말로 설명하고, AI가 만들고, 결과를 보고 고칠 곳만 다시 말합니다. 마음에 들 때까지 되풀이합니다. 이 장은 그 순환을 가장 작은 크기로 한 번 완주합니다.

> **핵심** 처음 목표는 잘 만든 프로그램이 아니라, 말로 시켜서 화면이 나오고 그것이 내 휴대폰에서 열리는 경험입니다.

이 장의 범위와 안전선

범위는 화면 하나입니다. 로그인, 결제, 남의 개인정보가 얽힌 것은 이 장의 범위 밖입니다. 연습은 아래 규칙을 지키며 합니다.

- 가상의 정보만 씁니다. 실제 이름, 연락처, 건강 정보, 남의 사진과 글은 넣지 않습니다.
- 카드 입력, 결제, 업그레이드 창이 나오면 입력하지 않고 닫습니다.
- 약관과 동의 문구는 읽고 직접 누릅니다.
- 같은 문장을 두 번 보내지 않고, 결과가 나올 때까지 기다립니다.
- 검색 결과의 광고 링크는 누르지 않고, 서비스 주소를 주소창에 직접 입력합니다.
- 영어 화면이 나오면 버튼 이름의 뜻을 AI에게 물어 찾습니다.

준비물과 도구 고르기

필요한 것은 브라우저와 계정 하나입니다. 이 책은 Claude를 기준으로 쓰므로, 웹에서 Claude에게 화면을 만들게 하고 결과를 대화 옆 화면(아티팩트)에서 봅니다. 아티팩트는 8장에서 설명했습니다. 같은 일을 하는 다른 브라우저 서비스도 있지만, 이 장의 요령은 도구가 달라도 같습니다.

도구를 고를 때 볼 점은 세 가지입니다.

- 설치 없이 브라우저에서 쓸 수 있는가
- 만든 화면을 바로 보여 주는가
- 만든 자리에서 주소를 받을 수 있는가

세 번째는 도구마다 다릅니다. 화면까지만 보여 주고 주소는 주지 않는 도구도 있습니다. 아티팩트의 공유 방식은 아티팩트 안내에서 확인합니다(2026년 11월 기준). 주소를 주지 않는 도구로 만들었다면 이 장 끝의 다른 길을 씁니다.

무료로 쓸 수 있는 범위와 사용량 한도는 서비스마다, 시기마다 다릅니다. 숫자는 이 책에 적지 않았으니 요금제 페이지에서 직접 확인합니다. 한도가 있는 도구에서는 습관 두 가지가 도움이 됩니다.

- 짧게, 한 번에 제대로 요청합니다.
- 고칠 점은 모아서 한 번에 보냅니다.

긴 요청과 긴 대화일수록 한도를 더 많이 쓰기 때문입니다. 한도에 걸리면 새 계정을 만들지 말고, 시간을 두고 이어 가거나 옆 사람과 짝으로 진행합니다.

첫 시도: 한 줄 요청

먼저 아무 준비 없이 한 줄로 시켜 봅니다. 이 결과가 뒤의 비교 기준이 됩니다. 새 대화를 열고 아래 문장을 한 번만 보냅니다.

프롬프트

> 이번 주 할 일을 날짜별로 한눈에 보는 화면을 만들어줘.
> 화면의 모든 글자는 한국어로 해줘.

대화 옆에 화면이 나타납니다. 다른 사람과 모양이 달라도 정상입니다. 같은 요청에도 결과가 매번 달라지는 것은 1장에서 설명한 성질입니다. 마지막 줄은 이후 모든 요청에 붙이는 정형 문구로 삼습니다.

영어로 나오는 일을 막아 줍니다.

화면을 보고 아쉬운 점을 적어 둡니다. 누구를 위한 화면인지, 무엇을 빼야 하는지 AI가 알 수 없었으니 당연히 아쉽습니다.

만들 것 하나 고르기

연습할 주제를 하나 정합니다. 기준은 세 가지입니다.

- 반복해서 하는 일
- 손으로 적는 일
- 기억에 의존하는 일

가상의 예로는 가족 일정표, 여행 준비 점검표, 개인 사업의 간단한 장부, 구직 활동 관리표가 있습니다. 이 장은 가족 일정표로 진행합니다. 한 주제를 끝까지 끌고 가는 편이 이해가 빠릅니다. 이어지는 6장도 같은 주제로 이어 갑니다.

범위는 화면 하나까지입니다. 여러 화면, 로그인, 저장 기능은 욕심내지 않습니다.

요청문은 네 칸

한 줄 요청이 아쉬웠던 이유는 정보가 없었기 때문입니다. 요청문에는 네 칸을 채웁니다.

칸	내용	가상의 예
쓰는 사람	누가 보는가	함께 사는 가족 네 명
목적	무엇을 위한가	이번 주 누가 언제 어디 가는지 알기
화면	무엇이 보여야 하는가	큰 제목, 월~일 7칸, 칸마다 시간·누구·할 일
하지 않을 것	이번에 뺄 것	로그인, 알림, 결제, 실제 이름·연락처

3장의 맥락, 제약, 출력 형식과 같은 틀입니다. "하지 않을 것"이 가장 중요한 칸입니다. 이 칸이 비어 있으면 AI가 범위를 멋대로 넓힙니다.

AI에게 요청문을 쓰게 하기

네 칸을 처음부터 쓰기 어렵다면 메타 프롬프팅으로 AI가 쓰게 합니다. 3장의 되묻게 하기와 비슷하지만 두 가지가 다릅니다. 질문을 번호를 붙여 한꺼번에 받고, 산출물이 "화면을 만드는 AI에게 그대로 붙여 넣을 요청문"입니다. 새 대화를 열고 대괄호를 내 것으로 바꿔 보냅니다.

프롬프트

> 나는 [가족 일정을 챙기는 사람]이야. [우리 가족 이번 주 일정표] 화면을 AI로 만들고 싶어.
> 먼저 나에게 질문을 5개 이하로 번호를 붙여 한꺼번에 물어봐 줘. 내가 답하기 전에는 요청문을 쓰지 마.
> 내 답을 받으면, 화면을 만드는 AI에게 그대로 붙여 넣을 요청문을 써 줘.
> 요청문에는 누가 쓰는지, 무엇을 위한 것인지, 화면에 보일 것, 빼 줄 것(로그인 · 결제 · 데이터베이스 · 실제 이름과 연락처)을 넣고, 한국어로, 화면 하나로 끝나게 써 줘.

질문에는 번호를 붙여 짧게 답합니다. 답은 가상의 정보로만 합니다. 완성된 요청문은 메모장에 복사해 보관합니다. 이 요청문은 이 장에서 계속 씁니다.

한 번에 하나씩 묻는 3장의 방식과 번호를 붙여 한꺼번에 받는 이 방식 중 어느 쪽이 낫다고 정해진 것은 없습니다. 생각이 정리되지 않았으면 하나씩, 답이 빠르게 나오면 한꺼번에 받는 편이 편합니다.

같은 대화에서 다시 만들고 비교하기

요청문을 만든 같은 대화에서 아래 문장을 보냅니다.

프롬프트

> 이 요청문대로 하나의 HTML 파일로 화면을 만들어줘. 로그인과 데이터베이스는 빼줘.

HTML은 웹 화면을 만드는 글 형식이며, 파일 하나로 열립니다. 이번 화면을 첫 시도의 화면과 나란히 놓고 비교합니다. 가상의 예로, 첫 화면이 "화 회식, 수 요가" 정도의 나열이었다면 요청문 화면은 "요일 · 누구 · 할 일"이 표로 정돈되어 나옵니다. 차이가 요청문에서 왔다는 것을 눈으로 확인하는 것이 이 실습의 목적입니다.

고칠 점은 모아서 한 번에

결과를 보고 확인할 것은 글자, 순서, 빠진 것입니다. 마음에 안 드는 곳을 바로바로 보내지 말고 메모장에 모아 적은 뒤 한 번에 보냅니다.

프롬프트

> 제목 아래에 "이번 주 일정 ○개"를 보여주고, 오늘 칸은 굵은 테두리로 하고, 글자를 지금보다 크게 해줘.

세 가지를 한 번에 시키면 요청이 한 번으로 끝납니다. 하나씩 보내면 세 번을 보내야 하고 대화가 길어집니다. 사진은 AI에게 만들게 하지 말고, 가지고 있는 사진이나 저작권 걱정이 없는 이미지를 씁니다.

4장은 한 번에 하나씩 고치라고 했습니다. 두 방식은 목적이 다릅니다. 무엇이 효과가 있었는지 알고 싶을 때는 하나씩, 사용량 한도가 있는 도구에서 횟수를 아껴야 할 때는 모아서 보냅니다. 서로 영향을 주지 않는 가벼운 수정은 모으고, 어디서 문제가 생겼는지 모를 때는 하나씩 확인합니다.

막힐 때 쓰는 문장 다섯 가지

상황	쓸 문장
무엇이 무엇인지 모르겠다	지금 화면에 뭐가 있는지 쉬운 말로 설명해줘
이상하게 나왔다	내가 원한 건 ○○인데 지금은 △△야. 고쳐줘
너무 많이 바뀌었다	처음 요청했던 모습에 가깝게 다시 만들어줘
화면이 영어로 나왔다	화면의 모든 글자를 한국어로 바꿔줘
빨간 오류 문구가 떴다	오류 문구를 그대로 붙여 넣고 "이걸 고쳐줘"

오류 문구는 요약하지 않고 그대로 붙입니다. 4장에서 말한 원칙과 같습니다.

주소 받고 휴대폰에서 열기

화면이 만족스러우면 주소를 받습니다. 주소가 있으면 누구나 같은 한 곳을 보고, 고치면 같은 주소에서 최신이 보입니다. 파일을 메신저로 보내는 방식과의 차이는 18장에서 표로 설명합니다.

공개하기 전에 확인합니다.

1. 화면에 가상의 연습 내용만 있고 실명과 전화번호가 없는가.
2. 공개 범위가 어디까지인가. 주소만 알면 누구나 열 수 있는 방식은 아닌가.
3. 만든 도구가 무엇을 공개하는지 화면 안내를 읽었는가.

그다음 도구가 안내하는 공유 또는 게시 기능으로 주소를 받습니다. 받은 주소는 메모해 두고, 휴대폰 브라우저의 주소창에 직접 입력해 열어 봅니다. 설치할 앱은 없습니다. 내 휴대폰에서 내 화면이 열리면 이

장의 도달선에 닿은 것입니다.

만든 도구가 주소를 주지 않는다면 화면을 HTML 파일로 받아 두었다가, 6장의 방법으로 호스팅 서비스에 올려 주소를 붙입니다.

흔한 실수

- 한 줄로 "만들어 줘"만 보냅니다. 누구를 위한 것인지, 무엇을 뺄지 알 수 없어 산으로 갑니다.
- 고칠 점을 하나씩 보내다 한도에 걸립니다.
- 실제 이름과 연락처로 연습합니다. 공개 주소는 검색에 잡힐 수 있습니다.
- 결제 창이 떠서 당황해 카드를 입력합니다. 연습에 결제는 필요 없으니 닫습니다.
- 다른 사람의 화면과 모양이 다르다고 걱정합니다. 같은 요청에도 결과는 다르게 나옵니다.

실습

1. 한 줄과 요청문 비교하기

위의 한 줄 요청과 요청문 결과를 나란히 놓고, 가장 크게 달라진 점 세 가지를 메모장에 적습니다. 그다음 아래 문장을 보내 비교의 이유를 AI에게 설명받습니다.

프롬프트

> 방금 만든 두 화면에서 달라진 점이 요청문의 어느 칸 때문인지 쉬운 말로 설명해줘.

2. 모아서 한 번에 고치기

내 화면에서 고칠 점을 세 가지 이상 메모장에 적고 한 번에 보냅니다. 말하지 않은 부분은 그대로 두라는 문장을 함께 붙입니다.

프롬프트

> 아래 고칠 점을 반영해줘. 말하지 않은 부분은 그대로 둬.
> [메모장에 적은 고칠 점]

3. 공개 전 점검과 주소 열기

실명과 연락처가 없는지 점검하고, 주소를 받아 휴대폰에서 직접 입력해 열어 봅니다. 열리지 않으면 어떤 화면이 나왔는지 그대로 AI에게 붙여 물어봅니다.

정리

- 물어보기는 글을 받고, 만들게 하기는 화면을 받습니다. 순환은 4장과 같습니다.
- 한 줄 요청과 네 칸 요청문(쓰는 사람, 목적, 화면, 하지 않을 것)을 나란히 비교하면 요청문의 힘이 눈에 보입니다.
- 요청문이 막막하면 AI에게 질문을 받게 한 뒤 요청문을 쓰게 합니다.
- 사용량 한도가 있는 도구에서는 짧게, 한 번에, 고칠 점은 모아서 보냅니다.
- 공개 전에는 가상 내용만 있는지 확인하고, 받은 주소를 내 휴대폰에서 열어 봅니다.

6장

PRD로 키우는 두 번째 프로그램

이 장은 5장에서 만든 화면을 PRD로 다시 설계해, 내 컴퓨터의 폴더에서 파일로 만들고, 점검한 뒤 주소를 붙이는 과정을 다룹니다. 그 과정에서 화면, 정보, 규칙의 세 층과 주소의 종류 같은 프로그램 구조의 기본 어휘를 함께 익힙니다. 완전 초보자 대상 이틀 과정의 둘째 날 흐름을 옮긴 것입니다.

요청문에서 PRD로

5장의 요청문은 네 칸이었습니다. 쓰는 사람, 목적, 화면, 하지 않을 것입니다. 여기에 "넣을 정보와 규칙"을 더하면 PRD 다섯 칸이 됩니다. 요청문이 자란 모양입니다.

칸	내용
쓰는 사람	누가 보고 쓰는가
해결할 일	무엇이 불편하고 무엇을 줄이고 싶은가
화면	무엇이 어디에 보이는가
넣을 정보와 규칙	어떤 정보를 다루고, 어떻게 움직여야 하는가
하지 않을 것	이번에 만들지 않을 것

4장의 PRD는 문제, 사용자, 성공 기준, 하지 않을 것, 제약, 화면 여섯 칸이고, 18장의 PRD는 여기에 데이터를 더한 일곱 칸입니다. 이 장의 다섯 칸은 처음 만드는 사람을 위한 줄인 형태입니다. 성공 기준과 제약이 빠져 있으니, 프로그램이 커지면 4장과 18장의 형태로 넓힙니다. 가장 중요한 칸은 여기서도 "하지 않을 것"입니다.

PRD가 필요한 이유는 AI 이전부터 같았습니다. 사람끼리 일할 때도 의뢰인과 개발자가 미팅하고, 요구사항을 정리하고, 글로 확정한 뒤에 코딩을 시작했습니다. 말로 한 약속은 서로 다르게 기억하기 때문입니다. 코딩 전에 만들 것과 안 만들 것을 정하고, 다 만든 뒤에는 맞게 됐는지 확인하는 기준이 됩니다. 개발자 자리에 AI가 와도 같은 이유로 필요합니다.

가상의 예로, 가족 일정표 PRD는 이런 모양입니다.

프롬프트

> [쓰는 사람] 우리 가족 네 명이 함께 보는 화면이야.
> [해결할 일] 이번 주 누가 언제 어디 가는지 한눈에 보고, 일정이 겹치는 것을 줄이고 싶어.
> [화면] 큰 제목 "우리 가족 이번 주 일정", 월~일 7칸, 칸마다 시간 · 누구 · 할 일.
> [넣을 정보와 규칙] 정보는 날짜 · 시간 · 누구 · 할 일 · 장소(가상). 같은 시간에 겹치면 빨간 표시, 사람마다 색이 다르다.
> [하지 않을 것] 로그인, 알림, 결제, 여러 주 보기, 실제 이름 · 연락처 · 주소.

프로그램은 세 층입니다

프로그램을 고칠 때 "어느 층을 고치는가"를 말하면 요청이 분명해져 AI가 덜 헤맵니다. 층은 세 가지입니다.

층	용어	뜻	비유
화면	프론트엔드	눈에 보이는 것	가게의 진열대
정보	데이터베이스(DB)	넣어 둔 내용	가게의 장부
규칙	백엔드	어떻게 움직이는가	가게의 영업 규칙

가족 일정표에서 화면은 큰 제목과 7칸 달력, 정보는 날짜·시간·누구·할 일, 규칙은 사람마다 색이 다르고 겹치면 빨간 표시가 나오는 것입니다. "글자를 크게"는 화면을 고치는 요청이고, "겹치면 알려 줘"는 규칙을 고치는 요청입니다.

정보를 저장하는 곳 네 가지

정보를 어디에 두는지에 따라 누가 볼 수 있는지가 달라집니다.

어디에	성질	비유
화면 안에만	새로 고치거나 닫으면 사라질 수 있습니다	칠판
이 기기의 브라우저에	이 휴대폰, 이 브라우저에서만 보입니다	내 수첩
내 컴퓨터의 파일에	폴더에 남지만, 휴대폰에서 열려면 주소를 붙여야 합니다	서랍 속 서류철
데이터베이스에	여럿이 같은 정보를 봅니다	함께 쓰는 장부

내 휴대폰에 넣은 일정이 가족의 휴대폰에는 안 보이는 일이 생깁니다. 브라우저에 저장하는 방식은 기기마다 따로 남기 때문입니다. 여럿이 같은 정보를 보려면 데이터베이스나 구글 시트 같은 공유 저장소가

필요합니다. 18장의 데이터 3단계 사다리가 이 이야기를 이어서 다룹니다. 공유 저장소에 둔 정보는 주소를 아는 누구나 볼 수 있는 구조가 되기 쉬우니 가상의 정보만 넣습니다.

클라이언트와 서버

클라이언트는 요청하는 쪽(휴대폰이나 컴퓨터의 브라우저)이고, 서버는 응답하는 쪽(늘 켜져 있는 컴퓨터)입니다. 식당에 비유하면 손님이 주문하고 주방이 음식을 냅니다. 브라우저에 주소를 입력하면 "이 주소의 화면을 주세요"라는 요청이 가고, 서버가 화면과 정보를 보냅니다. Claude 앱도 Anthropic의 서버에 묻는 클라이언트입니다.

서버를 빌려 늘 켜 두는 것을 호스팅, 만든 것을 서버에 올려 주소로 열 수 있게 하는 일을 배포라고 합니다. 5장의 도구는 브라우저 안에서 만들고 같은 곳에서 올렸고, 이 장에서는 내 컴퓨터에서 만들고 호스팅 서비스에 따로 올립니다.

주소의 네 가지

유형	예	성질
나만(이 컴퓨터)	`http://localhost:3000`	미리보기를 켜면 나오는 주소로, 이 컴퓨터에서만 열립니다
누구나(임시 주소)	○○.vercel.app 같은 형태	배포하면 붙습니다. 주소를 알면 누구나 엽니다
누구나(정식 도메인)	○○○.com, ○○○.kr	등록업체에서 기간 단위로 구입해 연결합니다. 선택 사항입니다
IP 주소	203.0.113.7	컴퓨터가 쓰는 번호입니다. 사람은 이름을 씁니다

localhost는 내 컴퓨터를 가리키는 이름이고, 콜론 뒤 숫자(3000)는 한 컴퓨터 안의 문 번호인 포트입니다. 사람은 이름(도메인)을 쓰고, DNS(Domain Name System)가 이름을 번호로 바꿔 찾아갑니다. 전화번호부와 같은 역할입니다. 용어는 부록 C에 정리했습니다.

GUI, CLI, API

화면으로 조작하는 GUI와 글로 조작하는 CLI는 9장에서 설명했습니다. 여기에 API가 하나 더 있습니다. API는 프로그램끼리 요청과 응답을 주고받는 창구입니다. 이 장의 흐름에서 사람은 GUI(앱에 말로 시킴)를 쓰고, AI가 CLI(배포 명령)를 대신 치며, 정보를 저장하는 단계에서는 프로그램끼리 API로 주고받습니다.

작업 폴더에서 PRD 쓰게 하기

이 장의 실습은 파일을 만들 수 있는 도구가 필요합니다. 이 책에서는 Claude Code(9장) 또는 데스크탑 앱의 Code 탭(8장)을 씁니다. 내 컴퓨터의 폴더에서 파일을 만들고 고치고 명령을 실행할 수 있기 때문입니다. 메뉴 이름과 위치는 바뀔 수 있으니 공식 문서(Claude Code 문서)를 함께 봅니다(2026년 11월 기준).

작업 폴더는 바탕화면에 새로 만듭니다. 이름은 영어로, 띄어쓰기 없이 짓습니다(예: myapp). 그 폴더를 도구에서 엽니다. 폴더를 믿을지 묻는 창은 내용을 읽고 허용합니다. 권한은 하는 일마다 먼저 묻는 기본 모드로 둡니다. 모든 것을 허용하는 모드는 쓰지 않습니다. 모델과 사고 강도 같은 설정은 바꾸지 않고 시작해도 충분합니다. 어려운 일을 맡길 때만 올립니다. 올릴수록 오래 생각해 느려지지만 복잡한 일에 강해집니다.

도구 설정에는 입력한 내용을 서비스 개선에 쓰도록 허락하는 항목이 있을 수 있습니다. 처음 세팅할 때 이 항목이 무엇이고 지금 어떤 상태인지 확인합니다. 어떤 정보가 외부로 전송되는지 모른 채 쓰지 않습니다. 이 점은 8장의 처음 세팅 체크리스트에서 이어서 다룹니다.

5장에서 만든 화면과 고치고 싶은 점의 메모를 옆에 두고 아래 프롬프트를 보냅니다.

프롬프트

> 나는 [가족 일정을 챙기는 일]을 하는 사람이야.
> 앞서 [우리 가족 이번 주 일정표]를 만들어 봤어. 고치고 싶은 점은 [토요일 칸 글자가 넘치는 것]이야.
> 이 폴더에 PRD(만들기 전에 쓰는 한 장짜리 설명서)를 PRD.md 파일로 만들려고 해.
> 먼저 나에게 질문을 5개 이하로 번호를 붙여 한꺼번에 물어봐 줘. 내가 답하기 전에는 파일을 만들지 마.
> 내 답을 받으면 아래 다섯 칸으로 PRD.md를 써 줘.
> [쓰는 사람] [해결할 일] [화면] [넣을 정보와 규칙] [하지 않을 것]
> 한국어로, 화면 하나로 끝나는 크기로, 실제 이름 · 연락처 · 건강 정보 없이 가상 데이터만 써 줘.

질문에 번호를 붙여 짧게 답하면 폴더에 PRD.md가 생깁니다. md는 마크다운 형식의 글 파일입니다. 파일을 만들겠다는 허락 창이 뜨면 읽고 허용합니다.

PRD를 내가 고칩니다

AI가 쓴 PRD를 그대로 쓰지 않고 다섯 가지를 확인합니다.

1. 화면 하나 크기인가. 여러 화면, 로그인, 결제가 끼어 있지 않은가.
2. 내 말로 읽히는가. 모르는 용어나 내 일과 다른 말은 바꿉니다.

3. 개인정보가 없는가. 실명, 연락처, 건강 정보가 없어야 합니다.
4. 하지 않을 것이 적혀 있는가. 비어 있거나 줄었으면 내가 채웁니다.
5. 규칙이 내 뜻과 같은가. 색, 겹침 같은 규칙을 확인합니다.

고칠 곳은 말로 시킵니다.

프롬프트

> PRD.md에서 아래만 고쳐 줘.
> - [고칠 점 1: 하지 않을 것에 "여러 주 보기"를 넣어줘]
> - [고칠 점 2: 화면을 한 개로 줄여줘]
> - [고칠 점 3: 글자를 더 크게 해줘]
> 나머지는 그대로 두고, 실제 이름이나 연락처는 넣지 마.

PRD를 한 장 그림으로 검수하기

긴 글은 읽다가 빠뜨립니다. 같은 대화에서 PRD를 한 장 그림으로 바꿔 보면 빠졌거나 틀린 칸이 눈에 띕니다.

프롬프트

> PRD.md 내용을 한눈에 보는 인포그래픽을 infographic.html 하나로 만들어줘. 다섯 칸을 색으로 나누고 글자는 크게.

폴더에서 infographic.html을 두 번 눌러 브라우저로 엽니다. 틀린 칸은 "PRD.md에서 ○○을 고쳐 줘"라고 말합니다. 부록 B의 인포그래픽 스킬과 같은 발상이며, 여기서는 PRD를 검수하는 용도입니다.

PRD를 열 줄 요청문으로 줄이기

PRD는 사람이 읽는 긴 글이고, 화면을 만들 때는 짧은 요청문이 다루기 편합니다. 이 변환도 AI에게 시킵니다. 긴 문서를 구현용 짧은 요청문으로 압축하는 메타 프롬프팅입니다.

프롬프트

> PRD.md를 바탕으로, index.html 하나를 만들 요청문(프롬프트)을 먼저 써서 보여줘. 아직 파일은 만들지 마.
> 요청문에는 index.html 파일 하나에 모두 넣기, 화면의 모든 글자는 한국어, 휴대폰에서도 보기 좋게, 가상 데이터만, 하지 않을 것을 꼭 넣어줘.
> 10줄 안팎으로 짧게.

나온 요청문에서 개인정보가 없는지, 하지 않을 것이 들어 있는지, 파일이 index.html 하나인지를 확인합니다. 고칠 곳은 말로 시킵니다.

허락 창을 읽는 법

파일을 만들거나 명령을 실행하려 하면 도구가 허락을 구합니다. 허락 창에는 하려는 일이 한 줄로 보입니다. 눌러 넘기는 습관이 가장 큰 위험입니다(9장). 읽을 때는 이 기준을 씁니다.

허락 창에 보이는 것	판단
내 작업 폴더 안의 파일을 만들거나 고친다	읽고 하지 않을 것이 들어 있지 않으면 허용합니다
이해되는 한 줄 명령(상태 확인 등)	무슨 일인지 이해되면 허용합니다
모르는 설치 명령	허용하지 않고 AI에게 무엇인지 묻습니다
삭제, 내려받기, 내 폴더 밖의 파일	허용하지 않고 이유를 묻습니다

수업이라면 모르는 화면에서 손을 들고, 혼자라면 허용하지 않은 채 AI에게 "이 허락 창이 무슨 뜻인지 쉬운 말로 설명해줘"라고 묻습니다.

이제 열 줄 요청문대로 파일을 만들게 합니다. 같은 대화에서 한 번만 보냅니다.

프롬프트

> 좋아, 이 요청문대로 index.html을 만들어줘.

끝나면 요약이 나옵니다. 요약에 삭제, 내려받기, 내 폴더 밖 이야기가 있으면 멈추고 확인합니다.

결과를 확인하고 되돌리기

일이 끝나면 요약을 읽고, 폴더에서 index.html이 생겼는지 봅니다. 도구에는 바뀐 파일과 줄을 보여 주는 검토 기능이 있을 수 있습니다. 마음에 안 들면 이전 상태로 되돌리거나 말로 고칩니다. 되돌리기 기능은 버전 관리(git)를 쓰는 폴더에서만 되는 경우가 많습니다(20장). 마음에 드는 결과가 나오면 바꾸기 전에 폴더를 통째로 복사해 둡니다.

파일을 브라우저로 열면 주소창에 `file:///…/myapp/index.html` 같은 경로가 보입니다. 인터넷 주소가 아니라 내 컴퓨터 안의 경로이고, 나만 볼 수 있는 파일입니다. 눌러 보며 이상한 곳을 메모장에 적습니다.

눌러 보고 모아서 고치기

메모를 그대로 붙여 같은 대화에 보냅니다. 새 프로젝트를 만들지 않습니다. 더하고 싶은 것은 하나만 정합니다. 여러 개보다 하나를 제대로 하는 편이 좋습니다.

프롬프트

> 아래 고칠 점을 index.html에 반영해줘. 말하지 않은 부분은 그대로 둬.
> 파일은 index.html 하나만 고치고, 실제 이름이나 연락처는 넣지 마.
> [메모장에 적은 고칠 점과 더하고 싶은 것]

끝나면 index.html을 새로 고침해 확인합니다.

올리기 전 점검

올리기 전에 사람이 확인할 다섯 가지입니다. AI가 만들었다고 해서 맞는 것은 아닙니다. 확인하고 결정하는 일은 사람이 합니다.

확인	내용
계산	일정 개수와 겹침이 손으로 세어도 맞는가
빈 칸	비워 두면 어떻게 보이는가
이상한 값	아주 긴 글자나 25:00 같은 시간을 넣어도 깨지지 않는가
휴대폰 화면	글자 크기, 줄바꿈, 잘림이 없는가
글자·표기	한국어, 오타, 날짜 모양이 맞는가

휴대폰 화면은 브라우저 창의 폭을 좁혀서 흉내 낼 수 있고, 주소를 붙인 뒤에는 실제 휴대폰으로 다시 봅니다. 여기에 공개 전 점검을 더합니다. 가상의 연습 내용만 있고 실명과 전화번호가 없는지, 폴더 맨 위에 index.html이 있는지, 연습과 상관없는 파일이 섞이지 않았는지 봅니다. 이 단계에서는 올리지 않습니다. 공개는 별도의 단계입니다.

파일에서 주소로

파일을 메신저로 보내면 받는 사람마다 사본이 생기고 휴대폰에서 잘 열리지 않을 수 있습니다. 주소로 건네면 모두가 같은 한 곳을 봅니다. 5장의 브라우저 도구는 만든 자리에서 주소를 줄 수 있지만, 내

컴퓨터의 파일은 주소를 따로 붙여야 합니다. 호스팅 서비스는 Vercel처럼 여럿이 있고, 올리는 방법은 서비스 공식 문서(Vercel 문서)를 따릅니다(2026년 11월 기준). 이 책은 서비스 하나를 예로 듭니다.

index.html을 만든 같은 대화에서 아래 문장을 보냅니다.

프롬프트

> 이 폴더의 index.html을 Vercel CLI(Command Line Interface, 명령줄 도구)로 배포해줘.
> 먼저 vercel whoami로 로그인됐는지 확인하고, 로그인이 필요하면 내가 할 일을 한 단계씩 알려 줘.
> 끝나면 짧은 프로젝트 주소(OO.vercel.app)만 알려줘.

이 문장에는 두 가지 요령이 들어 있습니다. 하나는 로그인 상태부터 확인하게 하는 것이고, 다른 하나는 로그인이 필요하면 사람이 할 일을 단계별로 알려 달라고 하는 것입니다. 로그인과 승인은 사람의 몫으로 남깁니다.

진행 중에 이런 일이 생깁니다.

1. 명령 실행 허락 창이 뜹니다. 명령을 읽고 이해되면 허용하고, 모르는 설치 명령이면 멈춥니다.
2. 브라우저에 호스팅 서비스의 승인 화면이 열립니다. 5장에서 쓴 계정과 같은 계정으로 로그인합니다. 다른 계정으로 승인하면 빈 계정이 새로 생깁니다. 비밀번호 방식과 소셜 로그인 방식을 섞으면 계정이 꼬이니 하나로 통일합니다.
3. 끝나면 짧은 주소가 나옵니다.

고친 뒤에는 "다시 배포해줘" 한마디면 됩니다. 비밀 값과 환경 변수, 되돌리기는 20장을 봅니다.

올린 주소는 누구나 볼 수 있습니다

주소를 아는 누구나 열 수 있고, 검색에 잡힐 수도 있습니다. 실명, 연락처, 건강 정보, 남의 사진, 남의 글은 올리지 않습니다. 올린 뒤에는 시크릿 창(로그인하지 않은 브라우저 창)에서 열어, 로그인 없이도 보이는지와 보이면 안 되는 것이 없는지 확인합니다. 로그인 화면이 뜨면 공개 범위 설정을 다시 확인합니다. 무료 플랜의 이용 조건은 서비스마다 다르고 개인·비상업 용도로 제한하는 경우가 있으니, 판매나 광고, 대가를 받는 제작에 쓰기 전에 약관을 확인합니다.

휴대폰에서 주소를 직접 입력해 열어 보고, 내 기기가 아닌 곳에서도 열리는지 확인하면 이 장의 도달선입니다.

되는 것과 안 되는 것의 지도

구분	내용
이번에 하는 것	말로 화면 한 장 만들기, PRD로 다시 만들기, 주소 붙이기
조금 더 하면 되는 것	정보를 저장하기, 화면 여러 개, 모양 다듬기
전문가 영역	로그인, 결제, 남의 개인정보, 여러 사람이 함께 쓰는 서비스

4장의 "잘 되는 일과 어려운 일"을 위치로 표시한 것입니다. 맨 아래 칸은 혼자 AI만 믿고 진행하지 말고 전문가의 검토를 받습니다.

더 해 보고 싶다면: 정보를 저장하기

일정을 저장해 새로 고침해도 남게 하려면 저장소가 필요합니다. 이 책의 기본은 구글 시트입니다(18장). 데이터베이스 서비스를 쓰는 길도 있는데, 어느 쪽이든 지킬 원칙은 같습니다.

- 공개 화면에 넣어도 되는 키만 쓰고, 비밀 키는 코드에 넣지 않습니다.
- 접근 규칙을 켠 상태로 시작합니다. 데이터베이스에는 행 단위로 접근을 제한하는 보안 기능(RLS, Row Level Security)이 있는 경우가 많습니다.
- 카드 입력 창이 나오면 멈춥니다. 연습에는 무료 범위로 충분합니다.
- 주소를 아는 누구나 볼 수 있으니 가상 정보만 넣습니다.

제품마다 키의 이름과 설정 방법이 바뀌므로 공식 문서와 대조합니다.

쓰고 난 뒤 정리하기

공용 노트북이나 더 쓰지 않을 도구는 정리하고 마칩니다.

1. 로그아웃: 앱의 계정 메뉴에서 로그아웃합니다.
2. 배포 서비스 로그아웃: AI에게 "vercel logout 해줘"라고 시킵니다. 내 노트북이 아닐 때 필요합니다.
3. 프로그램 제거: 운영체제의 설치된 앱 목록에서 지웁니다.
4. 남은 파일: 공용 노트북이면 작업 폴더와 앱이 남긴 설정 폴더를 지웁니다.

앱을 지워도 올린 주소는 그대로 열립니다. 폴더를 지우기 전에 복사해 둡니다.

흔한 실수

- PRD의 "하지 않을 것"을 비워 둡니다. 범위가 슬며시 커집니다.
- 허락 창을 읽지 않고 누릅니다. 삭제, 내려받기, 내 폴더 밖 이야기가 나오면 멈춥니다.
- 다른 계정으로 승인합니다. 빈 계정이 새로 생겨 올린 것이 보이지 않습니다.
- 파일을 메신저로 보내 사본을 늘립니다.
- 올리기 전 점검을 건너뜁니다. 휴대폰 화면에서 글자가 잘리는 것은 흔한 문제입니다.
- 브라우저에만 저장하고 가족의 휴대폰에서도 보일 것이라고 기대합니다.

실습

1. PRD 다섯 칸 쓰고 다섯 가지 확인하기

내가 만들고 싶은 작은 화면의 PRD를 위 프롬프트로 만들고, 다섯 가지 확인 중 "하지 않을 것"과 "개인정보"부터 읽어 고칩니다.

2. PRD에서 파일까지

PRD를 열 줄 요청문으로 줄이고, 요청문대로 index.html을 만듭니다. 허락 창에서 읽은 내용을 메모장에 두 줄 적습니다.

프롬프트

> 방금 허락 창에 나온 명령이 각각 무엇을 하는지 쉬운 말로 설명해줘. 하지 않아야 할 것이 있었는지도 알려줘.

3. 올리기 전 점검과 주소 확인

다섯 가지 확인표로 점검하고 가상 내용만 있는지 본 뒤, 주소를 붙입니다. 시크릿 창과 휴대폰에서 열어 보고 결과를 메모합니다.

정리

- PRD 다섯 칸은 5장의 요청문에 "넣을 정보와 규칙"을 더한 것이며, 가장 중요한 칸은 "하지 않을 것"입니다.
- 프로그램은 화면, 정보, 규칙의 세 층이며, 고칠 때 어느 층인지 말하면 요청이 분명해집니다.

- 주소는 나만 볼 수 있는 것과 누구나 볼 수 있는 것이 있고, 저장하는 곳에 따라 누가 정보를 보는지가 달라집니다.
- 허락 창은 읽고 누르며, 삭제, 내려받기, 내 폴더 밖, 모르는 설치 명령은 멈춥니다.
- 올리기 전에 다섯 가지를 점검하고, 올린 주소는 시크릿 창과 휴대폰에서 확인합니다.

7장

AX의 언어

이 장은 AX(AI Transformation, 인공지능 전환)가 무엇인지 정의하고, 이후 장이 함께 쓰는 어휘인 워크플로우, 단일 원본(SSOT), md 원본과 변환 사다리, 챔피언을 하나씩 정리합니다. 뒤의 장에서 이 용어가 나오면 이 장을 가리킵니다.

AX란 무엇인가

DX(Digital Transformation, 디지털 전환)는 아날로그에서 디지털로 옮기는 일입니다. 종이 결재가 전자결재로 바뀌는 것이 대표적입니다. AX는 디지털에서 AI로 한 걸음 더 가는 일입니다. 사람이 하던 판단과 작성의 첫 초안을 AI가 맡습니다. 한 줄로 정의하면 AI로 일하는 방식 자체를 바꾸는 것입니다.

요약하면 이렇습니다. 종이에서 파일로, 파일에서 데이터로 옮깁니다. DX는 종이를 줄여 파일로 관리하게 했습니다. AX는 파일도 줄여 데이터로 관리합니다. 원본을 하나만 두고, 문서가 필요할 때 AI로 만듭니다.

가상의 예로 비교해 보겠습니다. 고객 응대 현황을 관리하는 한 팀이 있습니다.

파일 중심	데이터 중심
응대현황.xlsx, 주간보고, 최종보고, 최종보고_수정, 진짜최종 같은 파일이 쌓입니다.	구글 시트 한 장이 단일 원본입니다.
보고 요청이 올 때마다 가장 최근 파일을 찾아 고칩니다.	주간보고, 임원보고, 발표 요청이 오면 그때 원본에서 문서를 만듭니다.
미리 만들어 두고 버전을 올리는 일이 반복됩니다.	미리 만들고 버전을 올리던 관행이 사라집니다.

파일 중심에서는 같은 숫자가 파일마다 따로 놀기 쉽습니다. 데이터 중심에서는 고칠 곳이 한 군데이므로 문서는 언제든 새로 만들 수 있는 사본이 됩니다.

> **핵심** AX는 도구를 하나 더 쓰는 일이 아니라 일하는 방식을 바꾸는 일입니다. 문서를 쌓아 두는 대신 원본 하나에서 필요할 때 문서를 만듭니다.

AI 활용과 AX는 다르다

AI를 쓴다고 해서 모두 AX는 아닙니다. 둘의 차이는 다음과 같습니다.

항목	AI 활용	AX
바뀌는 것	도구	일의 흐름
일의 모양	하던 일을 그대로 더 빨리 합니다	거치던 단계가 없어집니다
효과가 남는 곳	개인 생산성	조직 역량

AI 활용은 같은 일을 조금 빨리 하는 것이고, AX는 그 일의 단계 자체가 없어지는 것입니다. 둘 중 하나가 옳다는 뜻은 아닙니다. 개인이 AI 활용에 익숙해지는 일은 AX의 출발점입니다.

직원이 자라는 네 단계

AI 활용에서 AX로 가는 길은 보통 네 단계로 나눌 수 있습니다.

1. AI를 배우는 직원
2. AI를 업무에 활용하는 직원
3. AI로 업무를 개선하는 직원
4. AI 업무 도구를 직접 만들어 쓰는 직원

조직의 AX는 3단계와 4단계에 있는 사람이 있느냐에 크게 좌우됩니다. 이 단계는 21장의 컨설팅 진행 구조, 22장의 담당자 후보 점검과 이어집니다.

워크플로우: 같은 일의 세 가지 상태

워크플로우는 하나의 결과물을 내기까지의 일의 순서입니다. 같은 일도 흐름이 세 가지 상태로 나뉩니다. 가상의 예로 월간 매출 보고를 보겠습니다.

상태	흐름	걸리는 시간
지금	지점별 매출 파일 수거, 엑셀 정리, 합계 계산, 차트 작성, 보고서 작성, 검토, 발송	며칠
AI 활용	위와 같은 흐름에서 합계와 차트 한 칸만 AI로 바뀝니다.	몇 시간

상태	흐름	걸리는 시간
AX	지점별 입력이 한 시트로 모이고, 집계와 차트와 보고서 초안이 한 번에 나옵니다. 사람은 검토하고 발송합니다.	몇십 분

AX 상태에서는 칸이 하나 빨라진 것이 아니라 칸이 사라지고 순서가 바뀝니다. 시간은 흐름이 어떻게 달라지는지 보여 주는 방향일 뿐이며, 가상의 예이고 실측이 아닙니다.

병목 찾기 세 질문

어느 단계를 AI로 바꿀지 고르는 데는 세 질문이면 충분합니다. 반복하는 업무 하나를 떠올려 단계마다 묻습니다.

1. 이 흐름에서 가장 오래 걸리는 단계는 어디인가
2. 그 단계가 매번 같은 방식으로 반복되는가
3. 그 단계의 입력과 출력이 정해져 있는가

세 질문에 모두 "예"라면 AI로 바꿀 1순위입니다. 입력과 출력이 정해져 있어야 AI에게 맡기는 기준을 세울 수 있고, 매번 같은 방식으로 반복되어야 한 번 만든 지시문을 다시 쓸 수 있습니다.

단일 원본(SSOT)

SOT(Source of Truth)는 "이게 맞다"고 말할 때 보는 자료입니다. SSOT(Single Source of Truth, 단일 원본)는 그 자료가 하나뿐인 상태입니다. 워크플로우를 단순하게 만드는 핵심입니다.

원본이 없다는 신호

아래 중 하나라도 해당하면 원본이 없는 상태입니다.

- 최종_최종_진짜최종 같은 이름의 파일이 있습니다.
- 같은 숫자를 두 사람이 다르게 말합니다.
- 어느 것이 최신인지 물어봐야 알 수 있습니다.
- 규정이 메일, 공유 드라이브, 개인 PC에 흩어져 있습니다.

원본을 만드는 세 단계

1. 정합니다. 이 주제의 원본은 이 파일 하나라고 정합니다.

2. 표시합니다. 나머지 파일은 이름에 `_참고`를 붙이거나 아카이브 폴더로 옮깁니다.
3. 갱신합니다. 내용이 바뀌면 원본만 고칩니다.

도구를 만들 때는 PRD(Product Requirements Document, 제품 요구사항 문서)가 그 도구의 단일 원본입니다. 4장에서 다룬 PRD가 이 역할을 합니다. 사람이 바뀌어도 PRD를 보면 무엇을 왜 만들었는지 알 수 있어야 합니다. 또 데이터가 둘 이상일 때는 원본끼리 서로 맞는지 대조하는 크로스체크를 거쳐야 단일 원본이 완전해집니다. 예를 들어 매출 시트와 정산 시트가 따로 있다면 합계가 서로 맞는지 정기적으로 확인합니다.

원본은 md로 두고 아래로만 내려간다

원본을 어떤 형식으로 둘지도 정해야 합니다. 이 책은 Markdown(마크다운) 파일을 권합니다. Markdown은 제목, 표, 목록 같은 서식을 기호 몇 개로 적는 일반 텍스트 파일이며 메모장으로도 열립니다. 확장자는 `.md`입니다.

원본으로 좋은 이유는 세 가지입니다.

- AI가 읽고 쓰기 쉽습니다. 구조가 글자에 그대로 드러나는 편이라 AI가 문서의 짜임을 파악하기 쉽습니다.
- 사람도 읽을 수 있고, 열기 위한 전용 프로그램이 필요 없습니다.
- 가벼워서 긴 문서도 통째로 AI에게 줄 수 있습니다.

변환 사다리

원본 md에서 필요한 형식으로 내려보냅니다. 형식마다 확인할 것과 고치기 쉬운 정도가 다릅니다.

형식	확인할 질문	고치기	용도
.md	내용과 순서가 맞는가	쉬움	원본
.html	한 화면에 들어오는가, 강약이 맞는가	보통	화면 확인
.docx	회사 서식과 문체에 맞는가	번거로움	회람과 편집
.pdf, .pptx	인쇄와 발표에서 깨지지 않는가	어려움	최종 배포본

위쪽은 AI가 읽고 쓰기 쉽고, 아래쪽은 사람이 보기 쉽습니다. 사다리는 아래로만 내려갑니다.

운영 규칙

- 기본 틀을 고칠 일이 생기면 md를 고치고 다시 내려보냅니다.
- docx나 pptx 같은 최종 파일을 직접 고치는 일은 마지막에 한두 글자를 바로잡는 정도로 한정합니다.
- 최종 파일을 직접 고치면 원본과 달라져 단일 원본이 깨집니다.

문서 한 건을 원본에서 최종 파일까지 내려보내는 실제 흐름은 16장에서 다룹니다.

> **핵심** 고치는 곳은 원본 하나입니다. 최종 파일은 언제든 다시 뽑을 수 있는 사본으로 둡니다.

챔피언: 실제로 굴리는 사람

챔피언은 직책이 아니라 역할입니다. 조직 안에서 변화를 실제로 굴리는 사람이며, AI로 일하는 데 두드러지게 능숙한 직원이 맡는 경우가 많습니다. 세 가지 일을 합니다.

1. 업무를 매우 빠르고 쉽게 처리합니다.
2. 더 빠르고 쉽게 처리하는 시스템을 만들고 개선합니다.
3. 자동화 흐름을 만들어 본인 외의 직원도 쓸 수 있게 합니다.

이 역할을 하는 사람이 한 명만 있어도 부서의 일하는 방식이 달라질 수 있습니다. 22장의 "담당자를 고르는 기준"에 나오는 AX 담당자는 바로 이 챔피언 역할을 맡은 사람입니다. 직책을 붙이는 일보다 이 세 가지를 실제로 하는 사람을 찾는 일이 먼저입니다.

AX 세 조각

지금까지의 용어는 세 가지로 묶입니다.

조각	질문	뜻
워크플로우	무엇을 바꾸나	일의 순서
SSOT	무엇을 근거로 하나	단 하나의 원본
챔피언	누가 바꾸나	실제로 굴리는 사람

실습

실습 1. 병목 세 질문으로 업무 고르기

반복하는 업무 하나를 정하고 아래 프롬프트를 사용합니다. 고객 이름이나 연락처 같은 개인정보는 넣지 않습니다.

프롬프트

> 내 반복 업무는 [업무명]이야. 단계를 순서대로 나열해줘.
> 그다음 단계마다 (1) 가장 오래 걸리는가 (2) 매번 같은 방식으로 반복되는가 (3) 입력과 출력이 정해져 있는가를 예·아니오로 표시하고, 셋 다 예인 단계 1순위를 골라 이유를 말해줘.
> 내가 답할 정보가 부족하면 먼저 한 번에 하나씩 물어봐.

실습 2. 우리 팀 주제 하나의 원본 정하기

팀에서 자주 다루는 주제 하나를 고릅니다. 그 주제의 원본 파일을 하나 정하고, 나머지 파일은 `_참고` 폴더로 옮깁니다. 실제 이동은 사본으로 연습한 뒤에 합니다. 끝나면 앞의 네 가지 신호(최종 파일 중복, 숫자 불일치, 최신본 불명확, 흩어진 규정)에 해당하는 것이 남아 있는지 점검합니다.

실습 3. 변환 사다리 내려가 보기

업무 보고서 하나를 md로 쓰게 합니다. 그 md를 AI에게 html, docx, pdf로 차례로 내려보내게 합니다. 어느 단계에서 고치는 것이 가장 쉬웠는지 기록합니다. 시간이 되면 선택 과제로, 반복 업무 하나를 AI 활용 상태와 AX 상태의 흐름으로 각각 그려 비교해 봅니다.

정리

- AX는 도구를 바꾸는 일이 아니라 일의 흐름을 바꾸는 일입니다. 단계가 빨라지는 것이 아니라 단계가 사라집니다.
- 종이에서 파일로, 파일에서 데이터로 옮깁니다. 원본 하나에서 필요할 때 문서를 만듭니다.
- 원본은 하나만 두고, 정하고 표시하고 갱신하는 세 단계로 지킵니다.
- 원본은 md로 두고 html, docx, pdf로 아래로만 내려보냅니다. 최종 파일은 마지막 한두 글자만 직접 고칩니다.
- 챔피언은 직책이 아니라 역할입니다. 실제로 굴리는 사람이 변화를 만듭니다.

2부

도구

8장 · 데스크탑 앱과 프로젝트

9장 · Claude Code와 합의서

10장 · 멀티 세션과 서브에이전트

11장 · 스킬과 MCP

12장 · 이미지·영상 생성 연동

8장

데스크탑 앱과 프로젝트

이 장은 Claude 데스크탑 앱으로 일하는 기본 틀을 다룹니다. 처음 세팅과 파일을 넣기 전에 지울 것도 여기서 정리합니다. 대화, 파일 첨부, 프로젝트, 아티팩트, 커넥터가 각각 어떤 일을 위한 기능인지 구분하고, 업무마다 프로젝트를 하나씩 만들어 지침을 쓰는 방법을 익힙니다.

데스크탑 앱은 무엇을 위한 도구인가

Claude는 웹 브라우저(claude.ai)에서도 쓸 수 있고 데스크탑 앱으로도 쓸 수 있습니다. 앱은 macOS와 Windows용이 있으며 Linux용은 베타입니다(2026년 11월 기준). 내려받는 곳은 claude.com/download이고, 설치 방법과 시스템 요구 사항은 Claude 데스크탑 설치 안내에서 확인합니다.

브라우저 대신 앱을 쓰는 이유는 단순합니다. 업무용 창을 따로 두면 탭 사이에서 길을 잃지 않습니다. 대화와 프로젝트는 계정에 저장되므로 브라우저에서 열어도 이어집니다. 이 책의 1부에서 다룬 개념은 브라우저와 앱 모두에 똑같이 적용됩니다.

앱은 크게 세 가지 일을 맡습니다.

- 질문하고 문서를 다듬는 일반 대화
- 프로젝트 단위로 자료와 지침을 묶어 두고 반복하는 업무
- 코드와 파일을 직접 다루는 Code 탭(뒤에서 설명)

대화와 파일 첨부

대화는 가장 기본 단위입니다. 질문을 쓰고 답을 받고, 마음에 들지 않으면 고쳐 달라고 말합니다. 여기에 파일을 붙이면 Claude가 그 파일을 읽고 답합니다. 보고서 초안, 회의록, 표 파일, 이미지처럼 일상 업무 자료를 붙여서 요약·비교·정리를 시킬 수 있습니다.

지원하는 파일 형식과 크기 제한은 바뀔 수 있으므로 외우지 말고 첨부 화면의 안내를 따릅니다. 파일을 붙일 때 지킬 원칙은 두 가지입니다.

- 개인정보와 영업 비밀이 들어 있는 파일은 붙이기 전에 필요한 부분만 남기거나 가립니다. 이름, 연락처, 계좌 정보는 대부분의 작업에 필요하지 않습니다.
- 파일을 붙일 때는 무엇을 하라는지 같이 씁니다. 파일만 붙이고 "봐 줘"라고 쓰면 답이 길고 흐릿해집니다.

Claude가 읽은 파일의 내용은 서버로 전송되고, 한 번 보낸 내용은 회수할 수 없습니다. 요금제별 데이터 처리 방식은 공식 도움말(support.claude.com)의 개인정보 항목에서 확인합니다(2026년 11월 기준). 넣기 전에 지울 것은 네 가지입니다.

지울 것	어떻게	비고
고객 정보	원본 대신 집계한 숫자만	이름, 계좌, 연락처
직원 실명	'담당자 A', 'OO팀 과장'으로 대체	역할로 바꿉니다
미공개 실적, 내부 금액	조직 규정이 허용하는지 먼저 확인	규정이 우선입니다
사진 속 얼굴, 화면	가린 사본만	원본은 올리지 않습니다

법적으로 어디까지 허용되는지는 업종과 조직마다 다르므로 단정하지 않습니다. 조직의 규정이 우선합니다. 데이터 파일의 개인정보 지우기는 14장, 기록 폴더를 만들 때의 적용은 17장에서 이어집니다.

대화는 길어질수록 앞부분이 흐려집니다. 주제가 바뀌면 새 대화를 시작하는 편이 낫습니다. 10장에서 다룰 컨텍스트 관리와 같은 원리입니다.

프로젝트: 업무 하나에 작업 공간 하나

프로젝트는 자료와 지침을 묶어 두는 작업 공간입니다. 공식 설명에 따르면 프로젝트는 각자의 대화 기록과 지식(project knowledge)을 갖는 독립 공간이고, 프로젝트마다 지침(instructions)을 둘 수 있습니다. 프로젝트 안에서 시작한 대화는 그 프로젝트의 자료와 지침을 바탕으로 답합니다. 자세한 내용은 프로젝트 안내를 참고하십시오. 프로젝트 개수나 공유 범위는 요금제에 따라 다르므로 요금제 페이지와 공식 안내에서 확인합니다.

프로젝트가 필요한 순간은 이렇습니다.

- 같은 설명을 대화마다 반복해서 붙여 넣고 있을 때
- 같은 자료(제품 설명서, 문체 가이드, 응대 기준 등)를 계속 참고해야 할 때
- 같은 종류의 결과물을 여러 번 만들 때

반대로 한 번 묻고 끝나는 질문은 프로젝트가 필요 없습니다.

> **핵심** 프로젝트는 "다음에 또 쓸 맥락"을 한 번만 써 두는 곳입니다. 반복되는 업무가 보이면 프로젝트를 만듭니다.

프로젝트 지침에 무엇을 쓰는가

지침은 Claude가 그 프로젝트의 모든 대화에서 따를 기본 규칙입니다. 9장에서 다룰 CLAUDE.md와 같은 역할이며, 이 책에서는 이를 합의서라고 부릅니다. 길게 쓸 필요는 없습니다. 다음 다섯 항목이면 충분합니다.

1. 역할과 목적: 이 프로젝트가 무엇을 위한 것인지, Claude가 어떤 입장에서 돕는지
2. 독자와 상황: 결과물을 읽는 사람이 누구인지(고객, 팀장, 내부 직원)
3. 문체와 형식: 존댓말 여부, 분량, 표나 목록 사용 규칙, 금지 표현
4. 사실 규칙: 확인되지 않은 내용은 추측하지 않고 표시하라, 수치는 자료에 있는 것만 쓰라
5. 완료 기준: 어떤 상태가 되면 결과물이 끝난 것인지

작성할 때 지킬 점입니다.

- 하지 말라는 말보다 해 달라는 말을 먼저 씁니다. "과장하지 마라"보다 "근거가 있는 문장만 쓴다"가 따르기 쉽습니다.
- 우선순위가 충돌할 때의 기준을 한 줄 넣습니다. 예를 들어 "정확성이 속도보다 우선"처럼 씁니다.
- 한 번 쓰고 끝내지 않습니다. 결과가 기대와 다를 때마다 지침을 고쳐 다음 대화에 반영합니다.

가상의 예를 하나 보겠습니다. 소규모 공방의 체험 수업 문의 응대 프로젝트라면 지침이 이런 모양이 됩니다.

프롬프트

> 역할: 체험 수업 문의에 대한 답변 초안을 작성하는 보조자입니다. 최종 발송은 담당자가 합니다.
> 독자: 수업 신청을 고민하는 일반 고객입니다.
> 문체: 존댓말, 3~5문장, 전문 용어는 풀어서 씁니다.
> 사실 규칙: 첨부한 응대 기준 문서에 있는 내용만 안내합니다. 일정과 비용처럼 문서에 없는 것은 "담당자가 확인 후 안내드리겠습니다"로 답합니다.
> 완료 기준: 질문에 직접 답하고, 다음 행동(신청 방법 또는 추가 문의)을 한 줄로 안내합니다.

프로젝트 자료와 파일 정리 규칙

프로젝트에 올리는 자료(project knowledge)는 많을수록 좋은 것이 아닙니다. 다음 규칙으로 정리합니다.

- 파일 이름에 내용과 날짜를 넣습니다. 예: `응대기준_2026-11.docx`. "최종", "최종2", "진짜최종" 같은 이름은 어느 것이 현행인지 Claude도 사람도 알 수 없습니다.
- 한 파일에 한 주제만 담습니다. 서로 다른 주제가 섞인 긴 문서보다 주제별 짧은 문서가 낫습니다.
- 낡은 자료는 지웁니다. 새 기준과 옛 기준이 같이 있으면 Claude가 둘 중 아무거나 따릅니다.
- 민감한 자료는 올리지 않습니다. 프로젝트에 올린 자료는 그 프로젝트의 대화에서 계속 참고됩니다.
- 파일명은 `이름_날짜_v번호` 형식으로 통일합니다. 예: `응대기준_2026-11_v2.docx`. 규칙 파일과 함께 두는 방법은 16장에서 다룹니다.
- 자료 목록을 지침에 한 줄씩 적어 둡니다. "어떤 질문에는 어떤 파일을 우선 참고하라"고 알려 주면 정확도가 올라갑니다.

컴퓨터의 폴더도 같은 방식으로 정리해 두면 9장 이후 Claude Code를 쓸 때 그대로 도움이 됩니다. 업무별 폴더, 주제별 파일, 날짜가 들어간 이름이 기본입니다.

처음 세팅 체크리스트

채팅만 쓰던 설정 그대로는 실제 업무를 맡기기에 부족합니다. 처음에 네 가지를 맞춰 둡니다.

세팅	이유	확인할 곳
데스크탑 앱 사용	Code 탭에서 컴퓨터의 폴더를 열어 파일을 직접 읽고 고칠 수 있습니다	이 장 앞부분
채팅에서도 Claude Code 방식으로 일을 맡기는 모드	방법 안내가 아니라 실제 작업을 시킬 수 있습니다	앱 설정
가장 깊이 생각하는 상위 등급 모델	조건과 예외를 덜 놓칩니다	모델 선택 메뉴
사고 강도를 높게	답하기 전에 더 오래 생각해 중요한 일의 누락이 줄어듭니다	모델 선택 메뉴

모델 이름과 모드 이름은 자주 바뀌므로 이 책은 등급과 성격으로만 적습니다. 현재 이름과 설정 위치는 모델 개요와 앱 안의 안내에서 확인합니다(2026년 11월 기준). 문장 다듬기나 짧은 질문에는 이 설정이

필요 없고, 보고서나 분석, 규정 검토처럼 틀리면 곤란한 일에 씁니다.

> **핵심** 높은 설정은 사용량을 더 많이 씁니다. 중요한 업무에 골라 쓰고, 요금제 판단은 2장을 봅니다.

권한을 넓게 푸는 모드

확인 질문 없이 작업을 이어 가도록 권한을 넓게 푸는 모드가 있습니다. 파일을 지우거나 덮어쓰는 일까지 묻지 않고 진행될 수 있어 보안과 데이터 손실 위험이 큽니다. 쓴다면 조건을 지킵니다.

- 믿어도 되는 작업 전용 폴더에서만 씁니다. 홈 폴더나 업무 전체 폴더에서는 쓰지 않습니다.
- 켜기 전에 백업합니다. 코드라면 git으로 되돌릴 수 있게 해 두고(9장, 20장), 문서라면 폴더를 복사해 둡니다.
- 확인이 필요한 일은 AI가 질문하게 지시합니다. 예를 들어 "삭제, 외부 전송, 결제가 얽힌 일은 실행하기 전에 물어봐"라고 합의서에 적습니다.

기본 권한 모델은 9장 '권한과 안전'에서 다룹니다. 처음에는 기본 모드로 시작하고, 반복 작업이 익숙해진 뒤에 범위를 넓힙니다.

아티팩트: 결과물을 대화 옆에서 다듬는다

아티팩트는 Claude가 만든 문서, 디자인, 슬라이드, 대시보드, 작은 도구 같은 결과물을 대화 옆 화면에 따로 열어 주는 기능입니다. 직접 고칠 수도 있고 Claude에게 수정을 요청할 수도 있으며, 처음에는 본인에게만 보이고 공유 설정으로 범위를 정합니다. 내보내기 형식과 공유 방식은 아티팩트 안내에서 확인합니다(2026년 11월 기준). 사용하려면 설정에서 코드 실행과 파일 생성이 켜져 있어야 합니다.

긴 문서나 표, 간단한 계산 도구처럼 "여러 번 고쳐 가며 완성하는 것"은 아티팩트로 만들게 하는 편이 대화창에 길게 붙여 넣는 것보다 편합니다. 공유 전에는 누가 열 수 있는지와 어떤 정보가 담겼는지를 반드시 확인합니다.

커넥터와 웹 검색

커넥터는 Claude를 외부 서비스의 자료나 도구와 연결하는 장치입니다. 달력, 문서 저장소, 메신저 같은 서비스를 연결하면 Claude가 그곳의 정보를 읽거나 작업을 할 수 있습니다. 직접 만든 서버를 붙이는

사용자 지정 커넥터도 있습니다. 원리는 11장의 MCP(Model Context Protocol) 설명에서 다룹니다. 이 장에서는 쓰기 전 주의점만 짚습니다.

- 신뢰할 수 있는 서버만 연결합니다. 사용자 지정 커넥터는 Anthropic이 검증하지 않습니다.
- 연결할 때 요청되는 권한을 읽고, 필요한 범위만 허용합니다.
- 쓰기·삭제처럼 되돌리기 어려운 권한은 꼭 필요할 때만 줍니다.
- 대화마다 커넥터를 켜고 끌 수 있으니, 그 업무에 필요한 것만 켭니다.

자세한 내용은 사용자 지정 커넥터 안내를 참고하십시오. 웹 검색은 최신 정보나 사실 확인이 필요한 질문에 쓰입니다. 켜는 방법은 공식 도움말에서 확인하고, 검색 결과로 쓴 내용은 출처를 열어 직접 확인하는 습관을 둡니다. 15장 딥리서치에서 이 검증 방법을 자세히 다룹니다.

앱 안의 Code 탭

데스크탑 앱에는 Code 탭이 있어 앱 안에서 Claude Code를 쓸 수 있습니다. Claude Code 문서에 따르면 앱에는 Claude Code가 포함되어 있어 CLI를 따로 설치하지 않아도 되고, 변경 내용을 화면에서 검토하고 여러 세션을 나란히 돌릴 수 있습니다. 유료 구독이 필요하다고 안내되어 있습니다(2026년 11월 기준, Claude Code 개요). 9장에서 Claude Code의 개념과 합의서를 다룹니다. 터미널이 낯설다면 이 탭으로 시작해도 됩니다.

실습

1. 업무 프로젝트 하나 만들기

반복해서 하는 업무를 하나 골라 프로젝트를 만듭니다. 이름은 업무 내용이 드러나게 짓습니다(예: 고객 문의 답변 초안). 관련 자료 두세 개를 올리되 앞의 파일 정리 규칙을 지킵니다.

2. 프로젝트 지침 초안 작성

프로젝트 밖의 일반 대화에서 아래 프롬프트를 사용해 지침 초안을 받습니다. 대괄호 안을 본인 업무로 바꿉니다.

프롬프트

> 나는 [업무 이름]을 반복해서 하는 실무자야. 이 업무 전용 Claude 프로젝트의 지침을 만들고 싶어.
> 다음 다섯 항목으로 지침 초안을 써 줘: 역할과 목적, 독자와 상황, 문체와 형식, 사실 규칙, 완료 기준.
> 먼저 부족한 정보를 알아내기 위해 나에게 질문을 5개 이내로 해 줘. 내 답을 받은 뒤 초안을 작성해 줘.
> 초안은 15줄 이내로 쓰고, "하지 마라"보다 "한다" 형태로 써 줘.

3. 지침 시험

초안을 프로젝트 지침에 붙여 넣고, 실제 업무 요청 하나를 새 대화에서 시켜 봅니다. 결과가 기대와 어긋난 부분을 한 가지 고르고 지침에 규칙 한 줄을 추가한 뒤 다시 시켜 비교합니다.

4. 세팅 체크리스트 적용(선택)

위 네 가지 세팅을 내 계정에 적용하고, 같은 보고서 요약을 기본 설정과 높인 설정으로 각각 시켜 봅니다. 놓친 조건이나 예외의 개수를 기록해 비교합니다.

정리

- 데스크탑 앱은 대화, 프로젝트, Code 탭을 한곳에서 쓰기 위한 업무용 창입니다. 처음에는 앱, 상위 모델, 높은 사고 강도를 맞추고, 권한을 넓게 푸는 모드는 전용 폴더와 백업을 조건으로 합니다.
- 프로젝트는 반복 업무의 자료와 지침을 한 번만 써 두는 작업 공간입니다.
- 지침은 역할, 독자, 문체, 사실 규칙, 완료 기준 다섯 항목으로 짧게 쓰고 결과를 보며 고칩니다.
- 파일은 주제별로 나누고 `이름_날짜_v번호`로 이름 짓습니다. 넣기 전에 고객 정보, 실명, 미공개 수치, 사진 속 얼굴을 지우며, 한 번 보낸 내용은 회수할 수 없습니다.
- 커넥터는 신뢰할 수 있는 것만, 필요한 권한만 허용합니다.

9장

Claude Code와 합의서

이 장은 Claude Code가 무엇인지, 어떻게 시작하는지, 작업 전에 어떤 합의서를 쓰는지 다룹니다. 코드를 모르는 실무자도 따라 할 수 있도록 개념과 순서 위주로 설명합니다.

Claude Code란 무엇인가

Claude Code는 내 컴퓨터의 프로젝트 폴더 안에서 파일을 읽고, 수정하고, 명령을 실행하는 Claude입니다. 공식 문서는 이를 코드베이스를 읽고 파일을 편집하며 명령을 실행하는 에이전트형 도구라고 설명합니다(Claude Code 개요, 2026년 11월 기준). 이름에 코드가 들어 있지만 쓰임이 개발에만 한정되지는 않습니다. 폴더 안의 문서를 정리하고, 데이터 파일을 분석하고, 웹페이지를 만드는 일에도 같은 방식으로 씁니다. 3부의 실무 장이 이 방식을 씁니다.

8장의 대화형 Claude와 차이는 이렇습니다.

- 대화형 Claude는 붙이거나 첨부한 내용만 압니다. Claude Code는 폴더 전체를 직접 열어 봅니다.
- 대화형 Claude는 답을 줍니다. Claude Code는 파일을 직접 만들고 고칩니다.
- 그래서 결과가 빠르지만, 잘못 시키면 파일도 잘못 바뀝니다. 이 장의 합의서와 권한 설정이 그 위험을 줄이는 장치입니다.

Claude Code는 터미널, VS Code 같은 편집기, 데스크탑 앱, 웹에서 쓸 수 있고 같은 엔진을 공유합니다. 이 책은 어느 화면을 쓰든 같은 원리로 설명합니다.

GUI와 CLI

Claude Code는 글로 일을 시키는 도구입니다. 이 방식이 왜 힘을 내는지는 화면으로 조작하는 방식(GUI)과 글로 조작하는 방식(CLI, Command Line Interface)을 비교하면 보입니다.

GUI(그래픽 화면)	CLI(명령어 창)
버튼과 메뉴를 찾아 누릅니다.	하고 싶은 일을 글로 씁니다.
만든 사람이 만들어 둔 기능만 쓸 수 있습니다.	글로 표현되는 일이면 대부분 할 수 있습니다.
파일 100개의 이름을 바꾸려면 100번 누릅니다.	같은 일이 한 줄 명령입니다.

언어 모델은 글로 소통하므로 글로 조작하는 CLI와 잘 맞습니다. 다만 CLI에는 단점이 있습니다. 명령어를 외워야 하고, 무엇을 할 수 있는지 보이지 않으며, 한 글자만 틀려도 실패합니다. Claude Code는 이 단점을 덜어 줍니다. 명령어 대신 평소 말투로 요청하고, "이거 할 수 있어?"라고 물어볼 수 있고, 요청이 모호하면 되물어 줍니다. GUI의 쉬움과 CLI의 범위를 합친 도구라고 보면 됩니다.

설치와 첫 실행

설치 방법은 자주 바뀌므로 공식 문서의 명령을 그대로 따릅니다. 이 책에 명령을 옮겨 적지 않는 이유입니다.

- 설치 안내: Claude Code 개요와 같은 문서의 시작하기 항목
- 터미널이 처음이라면: 공식 문서에 터미널 여는 법부터 안내한 가이드가 있습니다.
- 터미널 없이 시작하려면: 8장의 데스크탑 앱에서 Code 탭을 쓰는 방법이 있습니다. 앱에 Claude Code가 포함되어 있습니다.

설치한 뒤의 기본 흐름은 세 단계입니다.

1. 작업할 폴더를 정합니다. 업무별로 폴더를 따로 둡니다.
2. 그 폴더에서 Claude Code를 시작합니다. 처음에는 로그인 안내가 나옵니다. 본인의 Claude 계정이나 조직에서 안내한 계정으로 로그인합니다.
3. 첫 요청은 읽기 전용으로 시작합니다. 예를 들어 "이 폴더에 어떤 파일이 있는지 설명해 줘"가 좋은 첫 질문입니다.

> **핵심** 첫 세션에서는 아무것도 바꾸지 않는 요청으로 시작합니다. 도구가 폴더를 어떻게 이해하는지 먼저 확인합니다.

합의서: 일을 시키기 전에 쓰는 한 장

일을 맡기기 전에 목표, 제약, 우선순위를 짧은 문서 한 장으로 합의해 두는 것이 합의서입니다. 건축주와 건축가가 도면을 합의한 뒤 짓는 것과 같습니다. 도면 없이 시작하면 중간마다 무엇이 우선인지 다시 협상하게 됩니다.

Claude Code에서 합의서의 실제 파일이 `CLAUDE.md`입니다. 프로젝트 폴더 맨 위에 두면 Claude Code가 세션을 시작할 때마다 읽습니다. 공식 문서는 CLAUDE.md를 세션마다 불러오는 지속 지침이라고 설명하며, 대화가 새로 시작돼도 이 파일 덕분에 맥락이 이어집니다(기억 관리 문서). 이 책은 CLAUDE.md를 합의서라고 부르겠습니다.

합의서에 들어갈 세 가지

1. 목표: 이 프로젝트에서 결국 얻으려는 것. "분기 보고서 초안을 매달 10분 안에 만든다"처럼 결과로 씁니다.
2. 제약: 어기면 안 되는 규칙. 건드리면 안 되는 폴더, 외부로 보내면 안 되는 정보, 지켜야 할 문체와 형식.
3. 완료 기준: 어떤 상태면 끝인지. "표의 합계가 원본과 일치", "오탈자 검사 통과"처럼 확인 가능하게 씁니다.

필요하면 네 번째로 충돌할 때의 우선순위를 한 줄 더합니다. 예를 들어 "정확성, 일정, 문체 순으로 우선한다"처럼 씁니다.

쓸 때의 규칙

- 짧게 씁니다. 공식 문서는 파일당 200줄 이하를 권하고, 길수록 지침이 잘 지켜지지 않는다고 설명합니다. 일부 영역에만 필요한 규칙은 별도 파일로 나누는 방법이 있습니다.
- 구체적으로 씁니다. "깔끔하게"보다 "표는 4열 이하로"가 따르기 쉽습니다.
- 코드나 폴더를 보면 알 수 있는 것은 적지 않습니다. 보고 알 수 없는 것, 즉 목표, 금지 사항, 용어 약속만 씁니다.
- 합의서는 강제 규칙이 아니라 참고 맥락입니다. 절대 어기면 안 되는 일은 다음 절의 권한 설정으로 막습니다.
- `/init` 명령으로 초안을 자동 생성할 수 있습니다. 자동 초안은 출발점일 뿐이며 목표와 제약은 사람이 채웁니다.

작업 폴더의 기억

Claude는 어제 한 일을 기억하지 못합니다. 대신 파일에 적어 두면 다음 세션에서도 읽습니다. 합의서 하나에 모든 것을 담지 말고, 역할이 다른 파일 세 개를 둡니다.

- `CLAUDE.md` : 이 폴더의 규칙(위의 합의서)
- `PROGRESS.md` : 지금 어디까지 했는지와 다음 시작점. 목표, 상태, 결정, 다음 할 일, 막힌 것의 다섯 칸

- `HISTORY.md` : 무엇을 왜 그렇게 정했는지 한 줄씩. 되돌아갈 때의 근거

`CLAUDE.md` 맨 위에 다음 한 줄을 둡니다.

프롬프트

> 작업을 시작하면 PROGRESS.md와 HISTORY.md를 먼저 읽고, 지금 상태와 다음 할 일을 한 줄로 요약한 뒤 시작한다.

제대로 깔렸는지는 새 세션에서 "이어서 하자" 한마디로 확인합니다. Claude가 PROGRESS.md를 읽고 현재 상태부터 말하면 됩니다. 사용하는 화면에 따라 지침 파일을 읽는 방식이 다를 수 있으므로, 말하지 않는다면 "PROGRESS.md부터 읽어"라고 한 번 시키고 합의서 위치를 점검합니다. 새 일은 새 폴더에서 시작하고 규칙 파일만 옮겨 와 용도에 맞게 고치며, 진행·이유 기록은 새로 만듭니다. 문서 업무에 쓰는 자세한 운영은 16장에서 다룹니다.

권한과 안전: 허용한 것만 실행된다

Claude Code는 파일 수정이나 명령 실행 같은 행동을 하기 전에 허락을 구하는 것이 기본 동작이며, 어디까지 묻지 않고 할지는 권한 모드로 정합니다. 공식 문서의 주요 모드를 간단히 옮기면 다음과 같습니다(2026년 11월 기준, 권한 모드 문서).

모드	묻지 않고 하는 일	쓰임
기본(default, 화면에는 Manual)	읽기만	모든 행동을 직접 확인할 때
편집 허용(acceptEdits)	읽기, 파일 수정, 기본적인 파일 정리 명령	결과를 보면서 반복할 때
계획(plan)	읽기와 탐색, 수정은 승인 전까지 하지 않음	바꾸기 전에 조사할 때
자동(auto)	전부, 배경에서 안전 점검	긴 작업에서 확인 횟수를 줄일 때

이 밖에 모든 것을 허용하는 모드(bypassPermissions)가 있지만, 공식 문서는 격리된 컨테이너나 가상 머신에서만 쓰라고 경고합니다. 실무에서는 쓰지 않는다고 생각하면 됩니다. 터미널에서는 Shift+Tab으로 모드를 순환하고, 데스크탑 앱에서는 입력창 아래의 모드 선택기를 씁니다. 자동 모드는 지원하는 모델과 제공 환경에서만 쓸 수 있고 조직 관리자가 끌 수도 있습니다. 최신 버전의 터미널과 VS Code 세션은 자동 모드로 시작할 수 있습니다. 세션을 열면 화면 하단의 현재 모드부터 확인합니다.

안전을 위한 습관은 다음과 같습니다.

- 허락 창이 뜨면 무엇을 허락하는지 읽고 누릅니다. 눌러 넘기는 습관이 가장 큰 위험입니다.

- 중요한 파일은 작업 전에 복사본을 만들거나 버전 관리를 해 둡니다. 되돌릴 수 있어야 맡길 수 있습니다.
- 비밀번호, API 키, 개인정보 파일이 있는 폴더는 작업 폴더에서 분리합니다.
- 자동 모드도 안전을 보장하지 않습니다. 방향을 믿을 수 있는 작업에만 쓰고, 민감한 작업은 직접 검토합니다.
- 외부 발송이나 삭제처럼 되돌릴 수 없는 행동은 항상 사람이 승인합니다.

먼저 계획하게 하기: 계획 모드

계획 모드에서 Claude Code는 파일을 읽고 조사한 뒤 계획을 제안하며, 계획을 승인하기 전에는 원본 파일을 수정하지 않습니다. 시작할 때 계획 모드로 열 수도 있고 세션 중에 모드를 바꿀 수도 있습니다.

계획을 보면서 확인할 질문은 세 가지입니다.

- 내가 말한 목표와 완료 기준이 계획에 그대로 들어 있는가
- 건드리지 말라고 한 것을 건드리려 하지 않는가
- 계획의 단계 중 이해되지 않는 단계가 있는가

이해되지 않는 단계가 있으면 승인하지 않고 설명을 요구합니다.

계획의 깊이: No plan, Local plan, Two-pass

모든 일에 같은 무게의 계획을 붙이면 단순한 일은 느려지고, 복잡한 일에 계획이 없으면 되돌릴 일이 쌓입니다. 액티전스 사용설명서는 일의 크기에 따라 계획의 깊이를 세 단계로 나눕니다.

- No plan: 한두 문장으로 설명되고 영향받는 파일이 분명한 일. 오탈자 수정이나 간단한 정리입니다. 바로 시킵니다.
- Local plan: 몇 개 파일이 바뀌지만 한 영역 안에서 끝나는 일. 계획 모드로 계획을 먼저 받고 승인합니다.
- Two-pass: 여러 영역에 걸치거나 구조 변경, 되돌리기 어려운 일. 첫 계획을 받은 뒤 한 번 더 별도로 검토합니다. 19장에서 자세히 다룹니다.

파일 수보다 되돌리기 어려운 정도가 더 중요한 기준입니다. 경계가 애매하면 한 단계 위로 올립니다. 과한 계획으로 잃는 것은 시간이고, 부족한 계획으로 잃는 것은 결과물 자체일 수 있기 때문입니다.

실습

작은 프로젝트의 CLAUDE.md 초안 만들기

업무 폴더 하나를 정하고(예: 월간 보고서 작성) 그 폴더에서 Claude Code를 시작합니다. 아래 프롬프트로 합의서 초안을 받습니다. 대괄호를 채워 쓰십시오. 합의서는 사람이 읽고 확정합니다.

프롬프트

> 이 폴더는 [업무 이름]을 위한 프로젝트 폴더야. 폴더 안 파일을 먼저 읽고 어떤 자료가 있는지 파악해 줘. 아직 아무 파일도 수정하지 마.
> 그다음 이 프로젝트의 CLAUDE.md 초안을 만들어 줘. 구성은 다음과 같아.
> 1) 목표: [이 업무에서 얻으려는 결과]
> 2) 제약: 건드리면 안 되는 파일과 폴더, 외부로 보내면 안 되는 정보, 지켜야 할 문체
> 3) 완료 기준: 확인 가능한 문장으로 3개
> 4) 충돌할 때의 우선순위: 한 줄
> 60줄 이내로 쓰고, 내가 말하지 않은 사실을 지어내지 마. 모르는 항목은 [확인 필요]로 표시해 줘.
> 초안을 보여 주고, 내 승인 후에만 CLAUDE.md 파일로 저장해 줘.

저장한 뒤 새 세션을 열어 "이 프로젝트의 목표와 제약을 요약해 줘"라고 물어봅니다. 합의서가 읽혔는지 확인하는 방법입니다.

선택 과제. 파일 이름 일괄 변경으로 CLI의 범위 느끼기

연습용 폴더에 가상의 파일(예: 사진 22장)을 복사해 두고 시킵니다. 원본은 건드리지 않도록 복사본 폴더에서만 합니다.

프롬프트

> 이 폴더의 파일 이름을 [날짜_번호] 형식으로 바꾸려고 해. 바꾸기 전에 현재 이름과 새 이름을 표로 먼저 보여 주고, 내가 확인하면 실행해 줘. 되돌릴 수 있도록 원래 이름 목록도 파일로 남겨 줘.

한 줄로 수십 개가 바뀌는 속도와, 실행 전에 표로 확인하는 절차의 필요를 함께 느껴 보는 것이 목적입니다.

정리

- Claude Code는 프로젝트 폴더 안에서 파일을 읽고 쓰고 명령을 실행하는 Claude이며, 글로 조작하는 CLI의 범위를 쉬운 말투로 열어 줍니다. 설치는 공식 문서의 안내를 따릅니다.
- 합의서(CLAUDE.md)에는 목표, 제약, 완료 기준을 짧고 구체적으로 쓰고, 진행(PROGRESS.md)과 결정 이유(HISTORY.md)는 따로 적어 다음 세션이 이어받게 합니다.
- 합의서는 참고 맥락이므로, 절대 막아야 하는 일은 권한 설정과 사람의 승인으로 지킵니다.
- 바꾸기 전에는 계획 모드로 계획을 받아 확인하고, 일의 크기에 따라 계획의 깊이를 정합니다.
- 첫 세션은 읽기 전용 요청으로 시작하고, 중요한 파일은 되돌릴 수 있게 해 둡니다.

10장

멀티 세션과 서브에이전트

한 세션에 모든 일을 몰아 시키면 대화가 길어지고 결과가 흐려집니다. 이 장은 일을 여러 세션과 서브에이전트로 나누는 이유와 규칙, 세션 사이에 일을 넘기는 인계 문서, 컨텍스트를 관리하는 방법을 다룹니다.

왜 나누는가

컨텍스트 오염

Claude가 한 번에 볼 수 있는 내용의 범위를 컨텍스트라고 합니다. 대화가 길어질수록 컨텍스트에는 지나간 시도, 폐기된 초안, 탐색하다 만 파일 내용이 쌓입니다. 이 잡음이 많아지면 처음 정한 목표가 흐려지고, 이미 버린 방안이 다시 튀어나오기도 합니다. 공식 문서는 같은 교정을 두 번 했는데도 틀리면 세 번째 교정을 덧붙이기보다 대화를 비우고 배운 점을 담아 새로 쓰는 편이 낫다고 안내합니다(모범 사례 문서, 2026년 11월 기준).

병렬 처리

서로 의존하지 않는 일은 동시에 진행하면 전체 시간이 줄어듭니다. 자료 조사와 문서 틀 잡기는 서로 기다릴 필요가 없습니다.

> **핵심** 일을 나누는 기준은 "서로의 결과를 기다리지 않아도 되는가"와 "맥락이 섞이면 해로운가"입니다. 둘 중 하나라도 해당하면 나눕니다.

여러 세션을 돌리는 규칙

세션은 Claude와 나누는 하나의 작업 대화입니다. 여러 세션을 동시에 쓸 때는 다음 규칙을 지킵니다.

1. 한 세션에는 한 가지 일만 맡깁니다. 세션 이름도 그 일로 짓습니다(예: 경쟁사 조사, 보고서 본문 작성).
2. 두 세션이 같은 파일을 동시에 고치게 하지 않습니다. 한쪽의 수정을 다른 쪽이 덮어쓰거나 서로 충돌합니다. 파일 소유를 세션별로 나눕니다. 조사 세션은 `조사/` 폴더만, 작성 세션은 `초안/` 폴더만

씁니다.

3. 세션 사이에서 일을 넘길 때는 대화 내용을 통째로 복사하지 않고 인계 문서를 거칩니다(뒤에서 설명).
4. 조사처럼 읽기만 하는 세션은 계획 모드나 읽기 전용 권한으로 돌립니다. 실수로 파일을 바꿀 가능성이 없어야 병렬로 안심하고 돌릴 수 있습니다.
5. 동시에 켜는 세션은 사람이 확인할 수 있는 수만큼만 둡니다. 처음에는 2~3개로 시작합니다.

코드 프로젝트에서는 세션마다 별도 작업 복사본(워크트리, git worktree)을 주는 방식이 공식 문서에 안내되어 있습니다. 편집이 서로 충돌하지 않도록 각 세션에 독립된 체크아웃을 주는 방법이며, 데스크탑 앱은 이런 세션을 나란히 돌리는 기능을 제공합니다(모범 사례 문서). 코드를 다루지 않는 업무에서는 위의 폴더 분리 규칙만으로 충분합니다.

서브에이전트: 전문가 팀처럼 일 맡기기

서브에이전트는 한 세션 안에서 특정 종류의 일을 맡는 보조 Claude입니다. 공식 문서에 따르면 각 서브에이전트는 자기만의 컨텍스트 창, 지침, 도구 접근 권한을 갖고 독립적으로 일한 뒤 결과 요약만 본 대화에 돌려줍니다(서브에이전트 문서, 2026년 11월 기준). 자료를 뒤지느라 생긴 긴 중간 기록이 본 대화에 쌓이지 않는다는 점이 장점입니다. 반대로 서브에이전트는 본 대화의 기록을 보지 못하므로, 맡기는 지시문에 필요한 맥락을 직접 써 줘야 합니다.

한 줄로 말하면 일의 일부를 떼어 별도의 Claude에게 맡기고 결과만 받아 오는 방식입니다. 혼자 할 때와 맡길 때는 다음이 다릅니다.

항목	혼자 할 때	맡길 때
속도	일을 차례로 처리합니다	여러 일을 동시에 처리합니다
대화창	중간 기록이 전부 쌓입니다	결론만 돌아옵니다
맥락	길어지면 앞의 내용을 잊습니다	각자 자기 대화창을 씁니다

맡기는 말은 단순합니다.

프롬프트

> 서브에이전트를 나눠서 동시에 해줘.

쓰기 좋은 때는 네 가지입니다.

- 자료 여러 개를 각각 읽어야 할 때
- 문서 20개처럼 많은 것을 나눠 동시에 요약하거나 조사할 때
- 크고 여러 파일로 나뉜 데이터를 나눠 훑으며 정합성을 확인할 때
- 관점이 여러 개 필요할 때(세 관점으로 각각 검토해 비교)

서브에이전트의 모델은 따로 정하지 않으면 보통 본 대화의 모델을 따르지만, 비용을 줄이려고 가벼운 모델을 쓰는 서브에이전트도 있습니다. 결과가 중요한 일이면 중간 이상 등급의 모델을 쓰도록 직접 지정합니다. 정의 파일의 `model` 항목이나 지시로 정하며, 값의 종류와 우선순위는 서브에이전트 문서에서 확인합니다(2026년 11월 기준). 등급을 고르는 기준은 1장과 2장에 있습니다.

액티전스 사용설명서는 이 구조를 사람의 팀 업무에 비유합니다.

- 오케스트레이터: 전체 흐름을 조율하고 결과를 통합하고 품질을 점검하는 본 세션
- 서브에이전트 A: 자료 수집
- 서브에이전트 B: 분석과 요약
- 서브에이전트 C: 초안 작성

각 서브에이전트는 이전 단계의 잡음 없이 깨끗한 컨텍스트로 시작합니다. 이를 새 눈(Fresh Eyes)이라고 부릅니다. 같은 사람이 쓰고 같은 사람이 검토하면 실수를 놓치기 쉽습니다. 검토를 따로 맡긴 서브에이전트는 작성 과정의 맥락 없이 결과만 보기 때문에 다른 시각으로 점검합니다.

팀 방식으로 얻는 것은 네 가지입니다.

- 집중: 각자 필요한 맥락만 갖고 깊이 처리합니다.
- 속도: 의존하지 않는 일을 동시에 진행합니다.
- 재개: 단계별 상태를 문서로 남기면 중간에 끊겨도 이어갈 수 있습니다.
- 품질: 마지막에 통합하고 점검하는 단계가 있어 한 명이 다 한 결과보다 안정적입니다.

주의할 점도 있습니다. 서브에이전트는 따로 쓰는 만큼 사용량이 더 듭니다. 서브에이전트의 요청도 본 대화와 같은 사용 한도에 포함된다고 공식 문서에 안내되어 있습니다. 작은 일에 쓰면 오히려 손해입니다.

서브에이전트를 만드는 방법

서브에이전트는 이름과 설명이 적힌 마크다운 파일로 정의합니다. 프로젝트 폴더의 `.claude/agents/`에 두면 그 프로젝트에서, 사용자 폴더의 `~/.claude/agents/`에 두면 모든 프로젝트에서 쓸 수 있습니다. Claude에게 "검토 전담 서브에이전트를 만들어 줘"라고 요청하면 파일을 대신 써 줍니다. 읽기만 하는

역할에는 파일을 수정할 수 없도록 도구를 제한하는 것이 안전합니다. 세부 설정은 공식 문서를 확인합니다.

역할 분담 예시(가상의 예)입니다.

역할	하는 일	권한
조사 담당	자료를 찾아 출처와 함께 정리	읽기 전용
작성 담당	조사 결과로 초안 작성	초안 폴더에만 쓰기
검토 담당	초안의 사실과 합의서 위반 점검	읽기 전용

인계 문서: 세션을 건너 일을 넘기는 법

새 세션은 이전 대화를 기억하지 못합니다. 그래서 세션을 바꾸거나 다른 사람에게 일을 넘길 때는 인계 문서를 만듭니다. 대화를 통째로 붙이는 것보다 짧고 정확합니다.

액티전스 스킬 중 Session-Handoff는 이 문서를 만들어 줍니다. 부록 B에 정리되어 있습니다. 좋은 인계 문서에는 다음이 들어갑니다.

- 변경한 것(CHANGE): 이번 세션에서 바꾼 것과 확정된 결과물
- 유지할 것(KEEP): 앞으로도 바꾸지 않을 것
- 하지 않을 것(OUT_OF_SCOPE): 이번 범위에서 일부러 뺀 것
- 결정과 근거: 무엇을 선택했고 왜 그랬는지, 폐기한 방안과 폐기 이유
- 다음 세션이 처음에 할 일: 읽을 파일과 첫 지시문

시행착오의 과정은 적지 않습니다. 인계 문서에 필요한 것은 확정된 결과와 결정의 이유이지 헤맨 기록이 아닙니다.

컨텍스트 관리: 요약하고, 비우고, 새로 시작한다

컨텍스트가 길어졌을 때의 수단은 세 가지입니다.

- 요약(compact): 대화를 요약해 컨텍스트를 줄입니다. 무엇을 남길지 지정하면 좋습니다. 공식 문서는 "API 변경과 테스트 명령을 남겨라"처럼 초점을 주는 방식을 예로 듭니다. 긴 대화는 자동으로 요약되기도 합니다.

- 비우기(clear): 관련 없는 새 일로 넘어갈 때 대화를 비웁니다. 비용도 아끼고 앞 일의 영향도 없앱니다.
- 새 세션: 일의 성격이 바뀌었다면 인계 문서를 쓰고 새 세션에서 시작합니다.

일의 한 단위가 끝났다면 비우고 시작하는 편이 대체로 낫습니다. 같은 실수를 두 번 고쳤는데도 반복된다면 대화를 비우고 배운 것을 담은 새 지시문으로 다시 시작합니다.

대규모 검토 맡기기 (동적 워크플로)

서브에이전트를 일일이 지시하지 않고, Claude가 작업에 맞는 스크립트를 써서 많은 서브에이전트를 한꺼번에 지휘하게 하는 방식이 있습니다. 공식 문서는 이를 동적 워크플로(dynamic workflows)라고 부릅니다. 이 책에서는 대규모 검토 모드라고 줄여 부릅니다. 일을 맡길 때 "워크플로로 해 줘"라고 말하는 식으로 요청하며, 이름과 호출 방법, 제공 범위는 자주 바뀌므로 공식 문서에서 확인합니다(2026년 11월 기준).

보통 모드와 비교하면 다음과 같습니다.

항목	보통 모드	대규모 검토 모드
일하는 방식	한 명이 차례로 처리하거나 서브에이전트 몇 개에 나눠 맡깁니다	스크립트가 많은 서브에이전트를 나눠 돌립니다
중간 결과	대화창에 쌓입니다	스크립트가 들고 있고 최종 결과만 대화창에 옵니다
시간과 사용량	짧거나 보통입니다	길고 큽니다

중간 결과를 대화창이 아닌 스크립트가 들고 있다는 점이 규모를 키우는 원리입니다. 공식 문서는 이 방식이 한 대화로 감당하기 어려운 분량에 맞고, 서로의 결과를 교차 검증하게 만들 수도 있다고 설명합니다. 다만 교차 검증으로 누락과 오류가 얼마나 줄지는 일마다 다르니 단정하지 않습니다.

켜는 때와 끄는 때를 나눠 둡니다.

- 켜는 때: 규정이나 계약의 전수 대조, 제출물 일괄 검토, 틀리면 곤란한 자료
- 끄는 때: 문장 다듬기, 짧은 질문. 사용량만 크고 얻는 것이 없습니다.

처음에는 자료 일부로 작게 시험해 사용량을 가늠한 뒤 범위를 넓힙니다. 공식 문서는 한 번의 실행이 같은 일을 대화로 처리할 때보다 훨씬 많은 사용량을 쓸 수 있다고 안내합니다.

프롬프트 예(가상의 예)입니다.

프롬프트

> 워크플로로 해 줘. 이 규정 문서 3개를 비교해서 서로 충돌하는 조항을 전부 찾아줘.
> 찾은 조항마다 다른 에이전트가 원문과 다시 대조해서 검증해줘.

> **핵심** 이 모드가 줄이는 것은 검증에 드는 시간이지 검증의 책임이 아닙니다. 충돌 조항으로 지목된 것은 사람이 원문과 대조해 확인합니다.

사용량과의 관계

여러 세션과 서브에이전트는 같은 계정의 사용량을 함께 씁니다. 병렬로 돌릴수록, 대규모 검토 모드를 켤수록 한도에 빨리 다가갑니다. 한도 숫자는 요금제마다 다르고 바뀌므로 요금제 페이지에서 확인합니다. 2장에서 요금제를 고르는 기준을 다뤘습니다. 실무에서는 다음 방식이 효율적입니다.

- 단순하고 짧은 일은 한 세션에서 끝냅니다.
- 병렬은 조사처럼 맥락이 길고 서로 독립적인 일에만 씁니다.
- 세션을 오래 열어 두지 말고 일이 끝나면 닫습니다.

실습

조사 세션과 작성 세션을 나누고 인계하기

업무 주제 하나(예: 신규 서비스 소개 자료)를 고릅니다. 세션 두 개를 따로 열어 진행합니다. 폴더는 `조사/`와 `초안/`으로 나눕니다.

조사 세션에 입력합니다.

프롬프트

> 주제: [주제]. 이 세션의 역할은 조사 하나야. 조사/ 폴더에만 파일을 만들고 다른 폴더는 건드리지 마.
> 핵심 질문 5개를 먼저 제안해 줘. 내가 확정하면 각 질문에 답이 되는 자료를 찾아 출처와 함께 정리해 줘.
> 확인하지 못한 것은 "미확인"으로 표시해. 끝나면 조사/인계.md를 만들어 줘. 포함할 것: 확정된 사실, 미확인 항목, 결정과 근거, 작성 세션이 가장 먼저 읽을 파일.

작성 세션에 입력합니다.

프롬프트

> 이 세션의 역할은 초안 작성 하나야. 조사/인계.md를 읽고, 거기 적힌 확정된 사실만 써서 초안/초안1.md를 만들어 줘.
> "미확인" 항목은 본문에 사실처럼 쓰지 말고 문서 끝에 확인 필요 목록으로 따로 모아 줘.
> 조사/ 폴더는 읽기만 하고 수정하지 마.

끝난 뒤 인계 문서만으로 작성 세션이 일을 이어갔는지, 빠진 정보는 없었는지 점검하고, 빠진 항목을 다음 인계 문서의 항목으로 추가합니다.

선택 과제

- 자료 3~5개를 서브에이전트로 나눠 동시에 요약시킵니다. 같은 자료를 한 대화창에서 차례로 시킨 결과와 속도, 누락을 비교합니다.
- 가상의 규정 문서 3개를 만들어 충돌 조항 전수 대조를 보통 모드와 대규모 검토 모드로 각각 돌립니다. 결과와 사용량을 비교하고, 최종 확인은 사람이 원문과 대조해서 합니다.
- 세션을 두 개 이상 동시에 띄워 서로 다른 일을 맡겨 봅니다. 세션 이름과 맡긴 일이 한눈에 구분되는지 확인합니다.

정리

- 일을 나누는 이유는 컨텍스트 오염을 막고 독립된 일을 병렬로 처리하기 위해서입니다.
- 한 세션에는 한 가지 일만 맡기고, 두 세션이 같은 파일을 동시에 고치지 않게 소유를 나눕니다.
- 서브에이전트는 자기 컨텍스트에서 일하고 요약만 돌려주는 전문가이며, 읽기 전용 검토 역할에 특히 적합합니다.
- 서브에이전트는 낮은 등급 모델로 배정될 수 있으니 중요한 일에는 중간 이상 등급을 지정하고, 규모가 큰 전수 검토에는 대규모 검토 모드를 쓰되 확인은 사람이 합니다.
- 세션을 넘길 때는 확정된 결과, 결정과 근거, 다음 할 일만 담은 인계 문서를 씁니다.
- 병렬과 서브에이전트는 사용량을 더 쓰므로 맥락이 길고 독립적인 일에만 씁니다.

11장

스킬과 MCP

반복하는 업무를 매번 같은 지시문으로 설명하고 있다면 스킬로 포장할 때입니다. 이 장은 스킬이 무엇이고 언제 쓰는지, 액티전스 스킬 14종의 구성, 직접 스킬을 만드는 방법, 그리고 외부 도구와 자료를 연결하는 표준인 MCP와 보안 원칙을 다룹니다.

네 단어를 구분하기

스킬, MCP, CLI, API는 비슷해 보이지만 하는 일이 다릅니다. 먼저 한 문장씩 구분해 둡니다.

단어	한 문장	비유
스킬	잘 되던 긴 프롬프트를 파일로 굳혀 이름으로 부릅니다	말로 쓴 업무지시서
MCP	Claude를 바깥 도구에 꽂는 표준 연결 규격입니다	케이블을 하나로 통일한 USB-C
CLI	버튼 대신 글자로 조작하는 창입니다	마우스 대신 말로 하는 주문
API	사람이 아닌 프로그램이 드나드는 문입니다	정문은 화면, 후문은 API, 열쇠는 API 키

CLI(Command Line Interface, 명령줄 인터페이스)는 9장에서 본 GUI(그래픽 화면)와 CLI의 차이로 이어집니다. GUI는 만든 사람이 버튼으로 만들어 둔 일만 할 수 있고, CLI는 글로 표현되는 일이면 무엇이든 시킬 수 있습니다. 파일 100개의 이름을 바꾸는 일이 한 줄이면 끝납니다.

MCP와 API는 혼동하기 쉽습니다. MCP는 AI 응용 프로그램이 외부 도구에 연결되도록 만든 규격입니다. API는 AI와 상관없이 모든 프로그램에 열려 있는 문입니다. 구글 시트를 예로 들면, 시트 API는 어떤 프로그램이든 열쇠만 있으면 쓸 수 있고, Claude가 그 시트를 직접 다루게 하려면 그 위에 MCP 같은 연결을 얹습니다. 이 장의 나머지는 스킬과 MCP를 중심으로 다루고, API는 뒤에서 한 번 더 짚습니다.

스킬: 반복 업무를 포장한 지시문

스킬은 반복하는 작업의 방법을 담은 지침 묶음입니다. 폴더 하나에 `SKILL.md` 파일이 들어 있고, 필요하면 참고 문서나 스크립트가 함께 들어 있습니다. 공식 문서는 스킬을 Claude의 도구 상자를 넓히는

지침 파일이라고 설명하며, Claude가 상황에 맞는 스킬을 스스로 쓰거나 사용자가 직접 호출할 수 있다고 안내합니다(스킬 문서, 2026년 11월 기준).

스킬이 없으면 잘 됐던 프롬프트를 찾지 못해 매번 긴 지시를 새로 붙여 넣게 됩니다. 스킬로 파일에 남기면 그 프롬프트가 항상 제자리에 있고, 한마디만 해도 Claude가 알아서 펼쳐 봅니다.

스킬을 부르는 방법은 두 가지입니다. Claude가 스킬의 설명(description) 한 줄을 읽고 지금 쓸 스킬인지 스스로 판단하거나, 사용자가 스킬 이름으로 직접 부릅니다. 구조는 단순합니다. 폴더 하나에 파일 하나가 있고, 파일 위쪽의 name과 description 두 줄이 호출의 기준이 됩니다. 스킬은 코드가 아니라 말로 쓴 업무지시서입니다.

가상의 예로 주간 보고서 스킬 파일은 이런 모양입니다.

프롬프트

```
---
name: 주간보고서
description: 주간 사업 현황 보고서를 정해진 양식으로 작성할 때 사용한다.
---

1. 이번 주 완료
2. 다음 주 계획
3. 리스크와 지원 요청

숫자는 전주 대비 증감을 함께 쓴다.
```

스킬의 동작 방식에서 알아 둘 점은 이렇습니다.

- 모든 스킬의 이름과 설명은 항상 Claude에게 보입니다. Claude는 이 설명을 보고 지금 쓸 스킬인지 판단합니다.
- 스킬의 본문은 스킬이 실제로 쓰일 때만 대화에 들어옵니다. 그래서 긴 참고 자료를 담아도 평소 비용이 크지 않습니다.
- 한 번 불러온 본문은 이후 대화에서도 유지되므로, 일회성 지시보다 계속 따를 기준으로 씁니다.

합의서, 프로젝트 지침, 스킬의 차이

도구	쓰임	언제 읽히는가
합의서(CLAUDE.md)	프로젝트 전체에 항상 적용되는 목표와 제약	세션마다

도구	쓰임	언제 읽히는가
프로젝트 지침	8장의 프로젝트 안 모든 대화의 기본 규칙	프로젝트 대화마다
스킬	특정 작업의 절차와 기준	그 작업이 요청될 때

특정 작업에만 필요한 절차는 합의서에 넣지 말고 스킬로 옮깁니다. 합의서가 짧게 유지됩니다.

언제 스킬로 만드는가

- 같은 지시문을 세 번 이상 복사해 붙였을 때
- 결과물의 형식과 점검 기준이 정해져 있을 때(보고서, 응대 문안, 회의록)
- 담당자가 바뀌어도 같은 품질이 나와야 할 때

한 번 하고 끝날 일이나 상황마다 방식이 달라지는 일은 스킬이 맞지 않습니다.

스킬 추가하는 방법

스킬은 쓰는 곳에 따라 넣는 방법이 다릅니다(2026년 11월 기준).

- Claude Code: 스킬 폴더를 개인용 위치(`~/.claude/skills/<스킬 이름>/SKILL.md`)나 프로젝트 위치(`.claude/skills/<스킬 이름>/SKILL.md`)에 둡니다. 프로젝트 위치에 두면 같은 저장소를 쓰는 팀원과 공유됩니다. 이미 있는 스킬 폴더의 추가와 수정은 세션을 다시 시작하지 않아도 반영됩니다. 처음 만든 최상위 스킬 폴더는 `/reload-skills`로 불러옵니다.
- Claude 앱(claude.ai와 데스크탑): 설정의 스킬 화면(Customize > Skills)에서 스킬 폴더를 ZIP으로 묶어 올립니다. 스킬을 쓰려면 코드 실행과 파일 생성이 켜져 있어야 하고, 올린 스킬은 공유하기 전까지 본인에게만 보입니다. 자세한 내용은 스킬 사용 안내에서 확인합니다.

스킬은 문서이면서 때로는 실행 파일을 포함하므로 출처가 중요합니다. 공식 안내는 신뢰할 수 있는 출처의 스킬만 설치하고, 덜 신뢰하는 곳이나 동료가 준 스킬도 켜기 전에 포함된 파일(특히 스크립트, 외부 연결 지시)을 검토하라고 말합니다.

액티전스 스킬 14종 한눈에 보기

액티전스는 실무에서 자주 반복되는 일을 스킬로 만들어 공개합니다. 각 스킬의 상세 내용과 발동 조건은 부록 B에 있고, 여기서는 구성만 봅니다(스킬 목록은 actigence.ai에서 확인).

분류	스킬
프롬프트 작성	GMP
문서 생성	PRD-Vibecoding, Gov-Startup-Bizplan, HWPX
검증과 평가	FactCheck, Paper Quality Enhancer
시각화	Infographic, Gov-Doc-Infographic
워크플로우	Session-Handoff
리서치	Matjip-Scout
개발	원샷 프로토타이핑, 풀스택 기능 구현, 디버깅, SEO·AEO·GEO 적용

몇 가지를 이 책의 장과 연결해 보면 이렇습니다. Session-Handoff는 10장의 인계 문서, FactCheck는 19장의 자가검증, GMP는 3장의 메타 프롬프트, 원샷 프로토타이핑과 풀스택 기능 구현은 18장과 20장의 앱 제작과 배포에서 쓰입니다.

일부 스킬은 요청에 특정 단어가 있을 때만 발동하도록 설계되어 있습니다. 예를 들어 디버깅 스킬은 "디버깅"이라는 단어가 있을 때만 동작합니다. 불필요하게 무거운 절차가 일상적인 질문에서 켜지지 않게 하려는 장치입니다. 스킬의 발동 조건은 그 스킬의 설명(description)에 적혀 있으므로, 안 켜진다면 설명을 먼저 읽습니다.

스킬 만들기

스킬 파일은 위쪽의 머리말(이름, 설명)과 아래쪽의 본문으로 구성됩니다. 공식 문서 기준으로 이름은 선택 사항이고 설명은 권장 사항이며, 설명은 이 스킬이 무엇을 하는지와 언제 쓰는지를 담습니다. 핵심 사용 상황을 설명의 맨 앞에 쓰라고 안내합니다. 본문은 500줄 이내로 유지하고, 긴 참고 자료는 별도 파일로 나누는 것이 좋습니다.

프롬프트

> ---
> name: weekly-report
> description: 주간 업무 메모를 팀장 보고용 주간 보고서로 정리한다. 사용자가 주간 보고, 이번 주 정리를 요청할 때 쓴다.
> ---
>
> (본문: 목적, 입력, 절차, 출력 형식, 완료 기준)

만드는 순서는 다음과 같습니다.

1. 반복하는 업무를 하나 고르고, 좋았던 결과물 두세 개를 모읍니다.
2. 입력(무엇을 받는지), 절차(어떤 순서로 하는지), 출력(어떤 모양인지), 완료 기준(무엇을 확인하는지)을 적습니다.
3. 설명에는 어떤 요청에서 쓰이는지 사용자가 실제로 쓰는 표현으로 적습니다.
4. 스킬로 같은 업무를 시켜 보고, 어긋난 부분을 본문에 한 줄씩 보완합니다.

직접 쓰기 어려우면 스킬 작성 전용 도구의 도움을 받습니다. 액티전스의 GMP 스킬이 스킬 작성을 돕고, Claude 쪽에도 스킬 제작을 돕는 skill-creator가 있습니다. 어느 쪽이든 결과를 사람이 읽고 확정합니다. GMP는 XML 구조, 핵심 제약의 이중 배치, 우선순위 라벨, 자가검증의 4층으로 지침을 설계하는 방법으로, 3장에서 설명했습니다.

> **핵심** 스킬은 한 번 쓰고 끝내는 문서가 아닙니다. 결과가 어긋날 때마다 본문을 고쳐서 키웁니다.

MCP: 외부 도구와 자료를 잇는 공개 표준

MCP(Model Context Protocol, 모델 컨텍스트 프로토콜)는 AI 응용 프로그램을 외부 시스템에 연결하기 위한 오픈 소스 표준입니다. 공식 소개는 USB-C 단자에 비유합니다. 기기마다 연결 방식이 다르던 것을 한 규격으로 맞춘 것처럼, MCP는 AI가 파일, 데이터베이스, 검색 도구, 업무 도구에 연결되는 방식을 하나로 맞춥니다. Claude Code 문서도 MCP를 외부 데이터 소스와 연결하는 오픈 표준으로 소개하며, 이를 통해 문서 저장소의 자료를 읽거나 이슈 관리 도구를 갱신하는 일이 가능하다고 설명합니다 (Claude Code 개요, MCP 안내, 2026년 11월 기준).

MCP가 없을 때와 붙였을 때의 차이는 일하는 방식에서 드러납니다.

상황	MCP가 없을 때	MCP를 붙였을 때
시트의 자료가 필요할 때	시트를 열어 복사해 붙여 넣습니다	Claude가 시트를 직접 엽니다
파일이 여러 개일 때	파일을 하나씩 올립니다	필요한 것을 Claude가 찾아 옵니다
결과를 반영할 때	답을 손으로 옮깁니다	그 자리에서 원본이 바뀝니다

이미지 생성 서비스의 MCP를 연결하면 Claude가 이미지를 직접 만들어 문서에 배치할 수 있습니다. 어떤 서비스가 연결되는지는 자주 바뀌므로 12장과 해당 서비스의 공식 안내에서 확인합니다.

스킬과 MCP의 역할은 다릅니다.

- 스킬은 어떻게 일할지에 대한 지침입니다.
- MCP는 무엇에 접근하고 무엇을 실행할 수 있는지에 대한 연결입니다.

예를 들어 "주간 보고서를 쓰는 방법"은 스킬이고, "팀 문서함의 파일을 읽는 능력"은 MCP 연결입니다. 둘을 함께 쓰면 연결된 자료로 정해진 방식대로 일합니다.

8장에서 본 커넥터는 앱 화면에서 MCP 서버를 연결하는 방식입니다. 연결 가능한 서비스의 목록과 설정 화면은 바뀌므로 공식 안내를 따릅니다.

API와 열쇠: 시트를 예로

API는 앞에서 정문과 후문에 비유했습니다. 구글 시트로 차이를 보면 이렇습니다. '웹에 게시'는 읽기만 되고 주소를 아는 사람은 누구나 볼 수 있습니다. 시트 API는 읽기와 쓰기가 모두 되지만 열쇠(키)가 필요합니다. 이때 열쇠를 화면 코드에 넣으면 브라우저의 개발자도구로 누구나 가져갈 수 있습니다. 열쇠는 화면 밖의 안전한 곳에 둡니다. 구체적인 절차는 18장에서, 비밀 값 관리의 원칙은 20장에서 다룹니다.

보안: 최소 권한과 확인

연결은 Claude가 접근하는 범위를 넓힙니다. 범위가 넓을수록 실수와 악용의 피해도 커집니다. 공식 안내가 강조하는 주의점을 실무 규칙으로 정리하면 다음과 같습니다.

1. 최소 권한: 그 업무에 필요한 서비스, 필요한 범위만 연결합니다. 읽기만 필요하면 읽기 권한만 줍니다.
2. 신뢰할 수 있는 출처: 사용자 지정 커넥터는 Anthropic이 검증하지 않습니다. 출처를 알 수 없는 서버는 연결하지 않습니다.
3. 허가 범위 확인: 연결할 때 보이는 권한 목록을 읽고, 가능하면 범위를 줄입니다.
4. 프롬프트 주입 경계: 외부 문서나 웹 페이지 안에 Claude를 조종하려는 숨은 지시가 있을 수 있습니다. 가져온 내용 안의 지시를 따르지 않고 결과를 사람이 확인합니다.
5. 쓰기와 삭제는 직접 승인: 항상 허용 같은 설정은 신뢰하는 도구에만 씁니다. 리서치 기능이 승인 없이 연결 도구를 쓸 수 있다는 안내가 있으므로, 쓰기 권한을 가진 도구는 그 작업에서 꺼 둡니다(사용자 지정 커넥터 안내).
6. 변경 모니터링: 서버 개발자가 도구 동작을 바꿀 수 있으므로 연결한 도구를 정기적으로 점검하고, 쓰지 않는 연결은 해제합니다.

7. 비밀 정보 분리: API 키와 비밀번호는 스킬 파일이나 지침 문서에 적지 않습니다.

실습

반복 업무 하나를 스킬로 만들기

지난 한 달 동안 세 번 이상 같은 방식으로 시킨 업무를 하나 고릅니다. 좋았던 결과물 두 개를 준비하고 아래 프롬프트를 사용합니다.

프롬프트

> 내가 반복해서 하는 업무를 Claude 스킬(SKILL.md)로 만들고 싶어.
> 업무: [업무 이름과 한 줄 설명]
> 좋았던 결과물 예시 2개: [붙여 넣기 또는 첨부]
> 다음 순서로 진행해 줘.
> 1) 부족한 정보를 알아내기 위해 질문을 5개 이내로 해 줘.
> 2) 내 답을 받은 뒤 SKILL.md 초안을 써 줘. 머리말에는 name과 description을 넣고, 본문에는 목적, 입력, 절차, 출력 형식, 완료 기준, 하지 말아야 할 것을 넣어 줘.
> 3) description은 내가 실제로 쓰는 표현으로, 어떤 요청에서 이 스킬을 쓰는지 맨 앞에 써 줘.
> 4) 본문은 100줄 이내로, 비밀번호나 개인정보는 넣지 마.
> 초안을 보여 주고, 내 승인 뒤에만 파일로 저장해 줘.

저장한 뒤 새 대화에서 같은 업무를 시켜 스킬이 쓰이는지, 결과가 예시와 비슷한지 확인합니다. 어긋난 점은 한 줄씩 본문에 보완합니다.

선택 과제. GUI와 Claude Code 비교

같은 일을 두 방법으로 해 봅니다. 연습용 폴더에 가짜 파일 20개를 만들고, 파일 이름을 일괄 변경하는 일을 정합니다.

1. 파일 탐색기에서 이름을 하나씩 바꾸고 걸린 시간을 잽니다.
2. Claude Code에 한 문장으로 시키고 걸린 시간과 결과를 비교합니다.

프롬프트

> 이 폴더의 파일 이름 앞에 정리_ 를 붙여 줘. 바꾸기 전에 바꿀 목록을 먼저 보여 줘.

중요한 파일이 있는 폴더에서는 하지 않습니다.

정리

- 스킬은 반복 업무의 절차와 기준을 담은 SKILL.md 폴더이며, 이름과 설명은 항상 보이고 본문은 쓰일 때만 읽힙니다.
- 같은 지시를 세 번 이상 반복했거나 결과물 형식이 정해져 있다면 스킬로 만들 후보입니다.
- 액티전스 스킬 14종은 문서 생성, 검증, 시각화, 워크플로우, 리서치, 개발로 나뉘며 상세는 부록 B에 있습니다.
- MCP는 외부 도구와 자료를 잇는 오픈 표준이고, 스킬이 방법이라면 MCP는 접근 범위입니다. 네 단어는 굳힌 프롬프트(스킬), AI용 연결 규격(MCP), 글자로 하는 조작(CLI), 프로그램용 문(API)으로 구분합니다.
- 연결은 최소 권한으로, 출처를 확인하고, 쓰기와 삭제는 직접 승인합니다.

12장

이미지·영상 생성 연동

이 장은 Claude가 이미지와 관련해 할 수 있는 일과 할 수 없는 일을 정확히 구분한 뒤, Claude로 기획과 프롬프트를 쓰고 별도의 생성 서비스에서 만들고 사람이 검수하는 흐름을 설명합니다. 저작권과 초상권, 게시 전 승인 절차도 함께 다룹니다.

Claude가 이미지에서 하는 일

먼저 사실부터 정리합니다. Anthropic 도움말에 따르면 Claude는 사진이나 일러스트를 이미지 생성 도구처럼 만들어 내지 않습니다. 대신 두 가지를 합니다(2026년 11월 기준, 도움말: Claude가 이미지를 만들 수 있나요).

- 올린 이미지를 보고 분석합니다. 사진 속 내용 설명, 화면 캡처 해석, 표나 차트 이미지 읽기, 디자인 시안 비평이 여기에 속합니다.
- 대화 안에서 차트, 다이어그램, 대화형 시각 자료를 만듭니다. HTML과 SVG로 만드는 방식이며, 도움말은 이 기능이 웹과 데스크탑에서 베타로 제공된다고 안내합니다.

여기에 8장의 아티팩트도 해당합니다. 도해, 순서도, 대시보드, 슬라이드처럼 코드로 그릴 수 있는 시각물은 Claude가 직접 만들 수 있습니다. 반면 사진처럼 보이는 인물 사진, 제품 연출 컷, 수채화풍 일러스트, 영상 클립은 이미지 생성 서비스의 영역입니다. 이 구분은 기능이 바뀔 수 있으므로 쓰기 전에 도움말을 다시 확인합니다.

> **핵심** Claude는 이미지를 이해하고 코드로 그리는 도구입니다. 사진과 영상 같은 픽셀 이미지는 생성 서비스에 맡기고, Claude는 기획과 지시문과 검수를 맡습니다.

업무에서 막히는 지점

Claude는 글, 표, 코드, 파일, 웹 화면은 만들지만 사진 같은 이미지는 직접 만들지 못합니다. 간단한 도형 수준의 그림만 가능합니다(2026년 11월 기준, 현재 기능은 위 도움말에서 확인합니다). 그래서 다음 같은 일에서 막힙니다.

- 안내문 배너

- 증서 배경
- 사내 공지용 이미지
- 제안서 표지
- 발표자료 삽화
- 홍보용 짧은 영상

기본 흐름: 기획, 생성, 검수

이미지·영상 작업은 세 단계로 나눠 쓰면 안정적입니다.

1. Claude로 기획하고 프롬프트를 씁니다. 목적, 독자, 분위기, 장면 구성, 생성 서비스에 넣을 지시문을 만듭니다.
2. 이미지·영상 생성 서비스에서 만듭니다. 서비스는 목적과 예산, 약관에 맞춰 고릅니다. 이 책은 특정 서비스를 추천하지 않습니다. 같은 지시문도 서비스마다 결과가 다르므로 두세 곳에서 시험해 보고 고릅니다.
3. 사람이 검수합니다. 결과물을 Claude에 다시 올려 점검받는 방법도 있습니다. 올린 이미지를 읽는 기능을 활용하는 것입니다. 단, 최종 승인은 사람이 합니다.

단계마다 담당이 다릅니다. 기획과 문구는 Claude, 픽셀을 만드는 일은 생성 서비스, 책임은 사람입니다.

좋은 이미지 프롬프트 쓰는 법

생성 서비스에 넣는 지시문에는 다음 요소가 들어가면 결과가 안정적입니다.

- 주제와 장면: 무엇이 어디서 무엇을 하는지
- 구도: 가까이에서, 위에서, 정면에서 등
- 분위기와 색: 밝고 차분한, 따뜻한 색 등
- 스타일: 사진, 플랫 일러스트, 3D 렌더 같은 양식
- 규격: 화면 비율(정사각형, 세로형, 가로형)
- 제외할 것: 넣지 말 것(글자, 로고, 사람 얼굴 등)

서비스에 따라 이미지 안의 글자가 틀리게 그려지는 일이 있습니다. 문구는 이미지에 넣지 말고 나중에 디자인 도구에서 얹는 방식이 안전합니다.

Claude에게는 서비스에 넣을 지시문을 만들어 달라고 요청합니다. 한 번에 한 장이 아니라 장면 기획 전체를 맡기면 일관성이 높아집니다.

브랜드 톤 가이드와 일괄 생성

여러 장을 만들 때는 장마다 따로 쓰지 말고 브랜드 톤 가이드를 먼저 정합니다. 8장의 프로젝트 지침에 넣어 두면 대화마다 반복하지 않아도 됩니다. 가이드에 들어갈 항목은 다음과 같습니다.

- 사용할 색과 피할 색
- 사진인지 일러스트인지, 인물이 나오는지
- 넣지 않을 요소(과장된 표현, 자극적인 연출, 타 브랜드 로고)
- 화면 비율 규격
- 문구 톤(9장의 합의서와 같은 방식으로 한 장에 정리)

가이드가 있으면 10장, 30장을 만들어도 같은 브랜드처럼 보입니다. 일괄 생성은 장면 목록을 표로 먼저 확정한 뒤 장별 지시문을 한꺼번에 받고, 첫 두세 장을 시험해서 가이드를 보완한 뒤 나머지를 만드는 순서가 효율적입니다. 처음부터 전부 만들면 가이드의 문제를 뒤늦게 발견해서 전부 다시 만들게 됩니다.

생성 도구와 연결하기

생성 서비스와 Claude를 직접 연결하는 방법도 있습니다.

- MCP 연결: 서비스가 MCP를 지원하면 Claude Code나 앱에서 도구로 불러 지시문을 보내고 결과를 받을 수 있습니다. MCP는 11장에서 설명했습니다. 연결 가능 여부는 서비스마다 다르므로 서비스 쪽 안내를 확인합니다.
- 스킬 연결: 생성 서비스용 스킬이 있으면 지시문 작성부터 호출까지를 절차로 묶을 수 있습니다. 스킬을 설치하기 전에 11장의 규칙대로 포함된 파일과 외부 연결 지시를 검토합니다.

연결을 쓸 때는 11장의 보안 원칙을 그대로 적용합니다. 특히 생성 서비스에는 결제 수단이 연결되어 있는 경우가 많으므로 한 번에 생성할 수 있는 수량을 제한하고, 비용이 드는 호출은 사람이 승인합니다. API 키는 스킬 파일이나 대화에 적지 않습니다.

연결하면 흐름이 짧아집니다. 여러 이미지·영상 생성 모델을 한 계정에서 쓰게 해 주는 통합 서비스를 MCP(11장)로 연결하면, 연결된 서비스가 이미지를 생성하고 Claude가 문서나 웹 화면의 알맞은 자리에

배치할 수 있습니다. 지원 모델, 해상도, 영상 길이 같은 사양은 서비스마다 다르고 자주 바뀌므로 해당 서비스의 공식 문서에서 확인합니다.

프롬프트

> 연결된 이미지 생성 서비스로 [용도]용 이미지를 만들어서 적절한 위치에 배치해줘.
> 화면 비율은 [비율]로, 이미지 안에는 글자를 넣지 마.

생성 결과가 문서나 화면에 바로 들어가더라도 그대로 게시하지 않습니다. 아래 "게시 전 사람의 승인" 점검을 거칩니다.

연결 없이도 충분합니다. 지시문을 복사해 서비스에 붙여 넣는 수동 방식이 가장 단순하고 안전하며, 소량의 작업에는 이것으로 충분한 경우가 많습니다.

저작권, 초상권, 상표

생성된 이미지를 업무에 쓸 때 확인할 것입니다. 법적 판단은 개별 사안에 따라 달라지므로 아래는 점검 목록이며 최종 판단은 전문가에게 확인합니다.

- 저작권: 생성 서비스의 약관이 결과물의 상업적 이용을 허용하는지 확인합니다. 약관은 서비스마다 다르고 바뀝니다. 특정 작가나 작품의 스타일을 그대로 흉내 내라는 지시는 피합니다. 기존 이미지를 올려서 변형시킬 때는 그 이미지의 이용 권한이 있는지 확인합니다.
- 초상권과 퍼블리시티권: 실존 인물의 얼굴이나 닮은 모습을 동의 없이 생성해 게시하지 않습니다. 직원이나 고객 사진을 쓸 때는 사용 범위를 서면으로 동의받습니다. 실존 인물처럼 보이는 가상 인물에도 오해가 없도록 주의합니다.
- 상표: 타사 로고, 제품 외형, 캐릭터가 결과물에 섞이지 않았는지 확인합니다. 이미지 안에 의도치 않게 로고처럼 보이는 요소가 생기는 일이 있습니다.
- AI 생성물 표시: 광고나 대외 홍보물에서 AI 생성 여부를 표시하는 규정이 있는지 해당 분야와 플랫폼의 안내를 확인합니다. 의료, 금융 광고처럼 규제 분야는 특히 확인이 필요합니다.
- 사실 오인: 제품 이미지가 실제 제품과 다르면 소비자를 오인시킬 수 있습니다. 실물과 다른 연출은 그렇게 표시합니다.

게시 전 사람의 승인

자동화가 가능하더라도 게시는 사람이 합니다. 다음 점검표를 거쳐 승인합니다.

1. 브랜드 톤 가이드와 맞는가
2. 이미지 안의 글자, 숫자, 로고, 얼굴에 오류나 의도하지 않은 요소가 없는가
3. 실물과 다른 부분이 사실을 오인시키지 않는가
4. 저작권, 초상권, 상표 점검을 통과했는가
5. 게시 문구와 이미지가 서로 어긋나지 않는가
6. 승인자 이름과 날짜를 기록했는가

점검표는 8장 프로젝트에 파일로 올려 두고, 검수 단계에서 Claude에게 체크하게 한 뒤 사람이 최종 확인하는 방식으로 쓰면 빠뜨림이 줄어듭니다. 부록 D의 체크리스트와 함께 사용합니다.

실습

한 달치 소셜 이미지 10장 기획

가상의 상황을 가정합니다. 작은 공방이 한 달 동안 올릴 소셜 이미지 10장이 필요합니다. 본인 업무로 바꿔 아래 프롬프트를 사용합니다. 이 단계에서는 이미지를 만들지 않고 기획과 지시문까지만 받습니다.

프롬프트

> 나는 [업종]을 운영하는 실무자야. 이번 달 소셜 미디어에 올릴 이미지 10장을 기획하려고 해.
> 브랜드 정보: 대상 고객은 [고객], 분위기는 [분위기], 쓰지 않을 요소는 [금지 요소]야.
> 다음 순서로 진행해 줘.
> 1) 부족한 정보가 있으면 먼저 질문을 5개 이내로 해 줘.
> 2) 브랜드 톤 가이드를 10줄 이내로 정리해 줘(색, 스타일, 인물 여부, 비율, 금지 요소).
> 3) 10장의 장면 목록을 표로 만들어 줘. 열은 번호, 게시 주차, 목적, 장면 설명, 화면 비율 4개로 해 줘.
> 4) 내가 표를 확정하면 장별 이미지 생성 지시문을 영어와 한국어로 써 줘. 이미지 안에는 글자를 넣지 말고, 게시 문구는 별도로 한 줄씩 써 줘.
> 5) 마지막에 게시 전 검수 점검표를 6항목으로 만들어 줘. 저작권, 초상권, 상표 항목을 포함해 줘.
> 실존 인물이나 타사 브랜드를 연상시키는 요소는 넣지 마.

표가 확정되면 첫 2~3장만 생성 서비스에서 시험하고, 결과를 보며 톤 가이드를 보완한 뒤 나머지를 만듭니다. 게시 전에는 앞의 점검표로 승인합니다.

정리

- Claude는 올린 이미지를 분석하고 HTML·SVG로 도해와 시각 자료를 만들 수 있지만, 사진이나 일러스트를 생성 도구처럼 만들지는 않습니다(2026년 11월 기준, 공식 도움말 확인).
- 흐름은 Claude로 기획과 지시문, 생성 서비스에서 제작, 사람의 검수 순서입니다.
- 여러 장을 만들기 전에 브랜드 톤 가이드를 정하고, 처음 두세 장으로 시험한 뒤 나머지를 만듭니다.
- 생성 도구를 MCP나 스킬로 연결할 때는 최소 권한과 비용 승인 원칙을 지키고, 연결 없이 복사해 쓰는 방식도 충분합니다.
- 저작권, 초상권, 상표를 점검하고, 게시는 반드시 사람이 승인합니다.

3부

업무 자동화

13장 · 문의 응대 자동화: 문자 발송 연동

14장 · 데이터 분석

15장 · 딥리서치

16장 · 문서 업무 전 과정: 제안서와 기획서

17장 · 결과보고서: 기록과 맞춰 보는 문서

18장 · 간단한 웹앱으로 업무 자동화

13장

문의 응대 자동화: 문자 발송 연동

문의가 들어오면 기록하고, 분류하고, 정해진 기준 문서에 근거해 답변 초안을 만들고, 사람이 확인한 뒤 문자로 보내는 흐름을 만듭니다. 이 장의 핵심은 AI에게 답을 "지어내게" 하지 않고 "기준 문서에서 찾아 쓰게" 하는 것, 그리고 발송 직전에 사람이 한 번 보는 구조입니다.

전체 흐름

문의 응대는 반복이 많고 규칙이 분명해서 자동화 효과가 큰 업무입니다. 다만 고객에게 나가는 글이므로 틀린 답이 곧 신뢰 손실이 됩니다. 그래서 흐름을 단계로 쪼개고, 단계마다 누가 책임지는지 정해 둡니다.

단계	하는 일	맡는 쪽
1. 수집	전화·문자·메신저·폼으로 들어온 문의를 한곳에 모음	사람 + 도구
2. 기록	시트에 한 줄씩 남김(접수 시각, 채널, 내용, 상태)	자동
3. 분류	문의 유형과 긴급도를 붙임	AI 제안, 사람 확인
4. 초안	기준 문서에서 근거를 찾아 답변 초안 작성	AI
5. 검토	초안을 읽고 승인·수정·보류	사람
6. 발송	문자 발송 서비스의 API로 전송	자동
7. 결과 기록	발송 성공·실패, 시각을 시트에 남김	자동

이 표에서 사람이 맡은 칸을 없애는 것이 목표가 아닙니다. 사람이 하는 일을 "처음부터 쓰기"에서 "읽고 승인하기"로 바꾸는 것이 목표입니다.

가상의 예: 작은 도예 공방

이 장의 예시는 가상의 사례입니다. 도예 공방 한 곳이 있고, 체험 수업 예약·변경·환불, 작품 배송, 영업 시간에 대한 문의가 하루에 십수 건 들어온다고 가정합니다. 담당자는 한 명이고, 수업 중에는 휴대폰을 볼 수 없습니다. 문의는 전화 부재중 문자, 메신저, 홈페이지 폼으로 들어옵니다.

공방의 목표는 "수업 중에도 문의가 쌓이지 않게 하되, 환불·일정 변경 같은 민감한 답은 사람이 확인한다"입니다. 이 목표가 뒤에서 나오는 자동 발송 기준을 결정합니다.

1단계. 문의를 시트에 모으기

처음에는 복잡하게 만들지 않습니다. 구글 시트 한 장에 열을 정해 두고, 문의가 올 때마다 한 줄씩 쌓이게 합니다.

프롬프트

> 접수번호 | 접수시각 | 채널 | 고객 호칭 | 연락처(끝 4자리만 표시) | 문의 내용 | 유형 | 긴급도 | 답변 초안 | 상태 | 발송결과

연락처 전체는 발송 단계에서만 필요합니다. 분석과 초안 작성에 쓰는 열에는 전체 번호를 넣지 않는 편이 안전합니다. 이름도 "김OO 고객님"처럼 호칭만 있으면 충분합니다.

문의 채널이 여러 개라면 모두 자동으로 연결하려고 애쓰지 않습니다. 가장 문의가 많은 채널 하나부터 연결하고, 나머지는 담당자가 시트에 붙여 넣는 방식으로 시작합니다. 폼 응답을 시트로 받는 기능이나 메신저의 내보내기 기능은 각 서비스의 도움말에서 현재 방법을 확인합니다.

2단계. 기준 문서를 먼저 만든다

답변 자동화의 품질은 AI 성능보다 기준 문서에서 갈립니다. 기준 문서는 "이 공방이 고객에게 약속하는 내용"을 한곳에 적어 둔 파일입니다. FAQ, 환불 규정, 영업 시간, 배송 안내를 하나의 문서로 묶습니다.

기준 문서를 쓸 때의 원칙은 다섯 가지입니다.

- 항목마다 번호를 붙입니다. 초안이 "규정 3-2에 따라"처럼 근거를 가리킬 수 있어야 합니다.
- 예외를 적습니다. "수업 24시간 전까지 변경 가능, 단 공휴일 전날 수업은 예외"처럼 예외가 빠지면 AI는 일반 규칙만 말합니다.
- 기준 문서에 없는 내용은 없다고 적습니다. "택배 외 퀵 배송은 제공하지 않음"처럼 명시합니다.
- 사람이 판단해야 하는 항목을 표시합니다. "환불 금액 계산은 담당자가 확인"이라고 적어 둡니다.
- 날짜를 적습니다. 마지막으로 고친 날이 있어야 낡은 규정으로 답하는 일을 막을 수 있습니다.

> **핵심** 기준 문서에 없는 질문에는 "확인 후 연락드리겠습니다"가 정답입니다. 초안 작성기에게 이 답을 허용하지 않으면, AI는 그럴듯한 규정을 만들어 냅니다.

3단계. 분류와 초안 작성

분류와 초안은 한 번의 요청으로 함께 시켜도 됩니다. 다만 출력 형식을 정해서 시트의 열에 그대로 들어가게 합니다. 아래는 도예 공방 예시의 구조입니다. 실습에서는 이 구조를 AI에게 만들어 달라고 요청합니다.

프롬프트

```
입력: 문의 내용 한 건 + 기준 문서 전체
출력(표 한 줄):
- 유형: 예약 / 변경·취소 / 환불 / 배송 / 이용 안내 / 기타
- 긴급도: 당일 처리 / 24시간 내 / 여유
- 근거: 기준 문서의 항목 번호 (없으면 "근거 없음")
- 답변 초안: 문자 한 통 분량
- 사람 확인 필요 여부: 예 / 아니오, 그리고 이유 한 줄
```

"근거" 열이 중요합니다. 근거 번호가 있으면 검토하는 사람이 기준 문서의 해당 항목만 보고 맞는지 확인할 수 있습니다. 근거가 없으면 초안은 자동으로 보류 상태가 되게 합니다.

톤 가이드

문자는 짧고 읽기 쉬워야 합니다. 공방 담당자의 말투를 몇 가지 규칙으로 적어 초안 작성기에 함께 넘깁니다.

- 첫 문장에 결론을 씁니다. "네, 변경 가능합니다."처럼 답부터 말합니다.
- 한 통에 한 가지만 안내합니다. 길어지면 두 통으로 나눕니다.
- 호칭과 공방 이름을 밝힙니다. 누가 보낸 문자인지 바로 알 수 있어야 합니다.
- 사과가 필요한 상황과 아닌 상황을 구분합니다. 모든 문장에 사과를 붙이면 오히려 신뢰가 떨어집니다.
- 확정되지 않은 일은 약속하지 않습니다. "가능할 것 같습니다" 대신 "확인 후 오늘 안에 다시 연락드리겠습니다"라고 씁니다.
- 금액, 날짜, 시간은 기준 문서에서 그대로 가져옵니다. 계산이 필요하면 사람에게 넘깁니다.

톤 가이드는 한 번 써 두면 모든 초안에 같은 말투가 적용됩니다. 담당자가 직접 쓴 실제 답변 문자 몇 개를 예시로 함께 주면 말투를 더 잘 따라 합니다. 이때 고객 정보는 지운 상태로 넘깁니다.

자동 발송과 검토 후 발송

모든 문의를 사람이 검토하면 시간 절약이 줄어들고, 모두 자동으로 보내면 사고가 납니다. 기준은 "틀렸을 때 비용"입니다.

자동 발송을 고려할 수 있는 경우

- 영업 시간, 위치, 주차처럼 바뀌지 않고 정답이 하나인 안내
- 접수 확인("문의 잘 받았습니다. 영업일 기준 하루 안에 답변드립니다")처럼 내용에 판단이 들어가지 않는 문자
- 기준 문서에 근거가 있고, 초안이 근거 문장을 거의 그대로 옮긴 경우

반드시 검토 후 발송해야 하는 경우

- 환불, 위약금, 보상처럼 금액이 오가는 답
- 일정 변경처럼 상대방의 약속이 걸린 답
- 근거가 없거나 근거가 여러 항목에 걸친 답
- 항의, 사고, 건강·안전과 관련된 문의
- 처음 문의한 고객, 또는 이전에 같은 문의가 반복된 고객

처음에는 모든 초안을 검토 후 발송으로 운영합니다. 2~3주 동안 승인율과 수정 내용을 시트에서 살펴보고, 수정 없이 승인된 유형만 자동 발송으로 옮깁니다. 반대로 사고가 한 번이라도 난 유형은 다시 검토 단계로 되돌립니다. 자동 발송 범위는 늘리는 속도보다 줄이는 속도가 빨라야 합니다.

문자 발송 서비스 고르기

문자는 통신사에 직접 보내지 않고 문자 발송 서비스(문자 대행 서비스)를 통해 보냅니다. 서비스 이름과 요금 구조는 시기에 따라 바뀌므로 이 책은 특정 서비스를 추천하지 않고, 고르는 기준만 정리합니다.

기준	확인할 것
API 제공	시트·웹앱에서 호출할 수 있는 공식 API와 사용 설명서가 있는가

기준	확인할 것
발송 결과 조회	성공·실패·지연 상태를 되돌려 받을 수 있는가
발신번호 등록	등록 절차와 필요 서류가 명확한가
메시지 종류	단문, 장문, 알림톡 등 쓰려는 종류를 지원하는가
요금 구조	건당 요금, 최소 충전, 유효 기간, 실패 건 처리 방식
안전 기능	허용 IP 제한, 발송 한도 설정, 예약 취소 기능이 있는가
지원	장애 공지와 문의 창구가 있는가

요금은 서비스 안내 페이지에서 직접 확인하고, 하루 문의량을 기준으로 한 달 비용을 스스로 계산해 봅니다. 시험 발송을 지원하는지, 내 번호로 한두 통 보내 볼 수 있는지도 확인합니다.

API 키는 비밀번호와 같습니다. 채팅창에 붙여 넣지 않고, 코드 파일에 쓰지 않고, 환경 변수나 서비스가 제공하는 비밀 저장소에 둡니다. 이 원칙은 20장에서 다시 다룹니다.

한국에서 문자를 보낼 때 지킬 점

문자 발송에는 법과 운영 규칙이 걸려 있습니다. 아래는 일반적인 원칙이며, 법령은 개정되므로 발송 전에 최신 내용을 직접 확인해야 합니다. 법령 원문은 국가법령정보센터에서, 스팸 관련 안내는 한국인터넷진흥원(KISA)의 불법스팸대응센터에서 확인할 수 있습니다(2026년 11월 기준).

발신번호 사전등록

문자를 보내는 번호는 미리 등록된 번호여야 합니다. 등록하지 않은 번호나 남의 번호로 보내면 발송이 막힙니다. 보통 문자 발송 서비스가 등록 절차를 안내하며, 번호의 사용 권한을 증명하는 서류를 요구합니다. 번호를 속여서 표시하는 것은 금지됩니다. 서비스를 고를 때 등록 절차부터 확인하고, 등록이 끝난 뒤에 개발을 시작하면 시간을 아낄 수 있습니다.

문의 답변과 광고는 다릅니다

고객이 물어서 답하는 문자는 일반적으로 광고가 아니라 요청받은 정보 제공입니다. 그런데 답변 문자 끝에 "이번 달 신규 수업 할인 안내" 같은 홍보 문구를 붙이면 광고성 정보로 볼 수 있습니다. 초안 작성기에는 "답변 문자에 홍보 문구를 넣지 마"라고 명시해 두는 것이 안전합니다. 어떤 문자가 광고에 해당하는지 애매하면 문자 발송 서비스나 불법스팸대응센터에 문의합니다.

광고성 문자를 보낼 때

영리 목적의 광고성 정보를 문자로 보내려면 정보통신망법이 정한 의무를 지켜야 합니다. 핵심은 다음과 같습니다.

- 수신자의 사전 동의를 받아야 합니다. 동의는 광고 수신에 대한 것으로 따로 받습니다. 문의했다는 사실이 광고 수신 동의는 아닙니다.
- 밤 시간대(오후 9시부터 다음 날 오전 8시)에 광고를 보내려면 일반 수신 동의와 별도로 야간 수신 동의를 받아야 합니다.
- 문자 첫머리에 "(광고)"를 표시하고, 전송자의 명칭과 연락처, 무료로 수신을 거부할 수 있는 방법을 함께 밝혀야 합니다.
- 수신 거부 의사를 밝힌 사람에게는 더 이상 광고를 보내지 않아야 하며, 거부 처리는 지체 없이 이뤄져야 합니다.
- 수신 동의를 받은 사실은 기록해 두고, 동의 유지 여부를 다시 확인해야 하는 주기와 방법은 KISA 안내서에서 확인합니다.

세부 요건과 예외, 과태료 기준은 조문과 KISA 안내서에서 확인합니다. 안내서는 개정되므로 최신본인지 봅니다. 이 책의 설명은 법률 자문이 아닙니다. 광고 문자를 정기적으로 보낼 계획이라면 전문가 상담을 받는 편이 안전합니다.

개인정보는 최소한으로

문의 응대에는 연락처가 필요하지만 그 외 정보는 대부분 필요하지 않습니다.

- AI에게 보내는 문의 내용에서 연락처, 주소, 계좌번호, 주민등록번호는 지웁니다.
- 시트에서 분석 열과 발송 열을 나눕니다. 전체 연락처는 발송 단계에서만 읽습니다.
- 문의 기록을 얼마나 보관할지 기간을 정하고, 지난 기록은 삭제합니다.
- 고객이 삭제를 요청하면 시트와 발송 서비스 양쪽에서 지울 수 있어야 합니다.
- 개인정보 처리 방침에 문의 응대 목적과 문자 발송 서비스 이용을 적습니다.

사용 중인 Claude 요금제와 데이터 처리 방식은 Claude 고객 지원 센터에서 확인하고, 조직의 규정이 있다면 먼저 따릅니다.

만들 때의 순서

한 번에 모든 것을 연결하지 않고 다음 순서로 올라갑니다.

1. 시트와 기준 문서만 만들고, 초안은 Claude 채팅에서 한 건씩 붙여 만들어 봅니다.
2. 초안이 안정되면 시트의 여러 줄을 한꺼번에 처리하게 합니다.
3. 발송 서비스 계정과 발신번호 등록을 끝냅니다.
4. 내 번호로만 시험 발송을 합니다. 광고 표시가 없는 일반 답변 문자만 대상으로 합니다.
5. 승인한 건만 발송하는 버튼을 만들고, 결과를 시트에 기록하게 합니다.
6. 몇 주 운영한 뒤 자동 발송 범위를 정합니다.

두 번 보내지 않고, 실패를 놓치지 않기

발송 버튼을 두 번 누르거나 네트워크가 끊겨 다시 시도하면 같은 답변이 두 번 나갈 수 있습니다. 시트의 각 문의에 `발송 상태` 열(대기·발송됨·실패)을 두고, 발송 전에 이 값이 `대기`인지 확인한 뒤에만 보내게 합니다. 같은 요청을 여러 번 실행해도 결과가 한 번만 남는 성질을 멱등성이라고 부릅니다.

실패한 건은 `실패`로 표시하고 실패 사유와 시각을 함께 남깁니다. 다시 보낼지는 사람이 확인하고 정합니다. 자동 재시도를 넣는다면 횟수에 상한을 둡니다. 실패 목록을 누가, 언제 확인할지도 정해 둡니다.

연결 단계에서 막히면 오류 메시지를 그대로 붙여 Claude에게 원인을 묻습니다. 앱을 직접 만드는 방법은 18장에서 다룹니다.

실습

실습 1. 기준 문서 만들기

내 업무(또는 가상의 공방)의 자주 묻는 질문 5개와 환불·변경 규정을 기준 문서로 정리합니다. 아래 프롬프트에 실제 내용을 붙여 사용합니다.

프롬프트

> 아래는 우리 가게의 안내 내용이야. 이것을 고객 문의 응대의 기준 문서로 정리해 줘.
> - 항목마다 번호를 붙이고, 질문-답-예외 순서로 써 줘.
> - 내가 쓰지 않은 내용은 추가하지 마. 빠져 있어 보이는 항목은 마지막에 "보완 필요" 목록으로 따로 알려 줘.
> - 사람이 판단해야 하는 항목(환불 금액 계산 등)은 [담당자 확인]으로 표시해 줘.
> - 맨 위에 마지막 수정일을 적을 자리를 만들어 줘.
>
> [안내 내용을 여기에 붙여넣기. 연락처·계좌번호는 지울 것]

실습 2. 답변 초안 생성기 만들기

기준 문서를 근거로만 답하는 초안 작성기를 만듭니다. 먼저 Claude 채팅에서 동작을 확인하고, 마음에 들면 9장의 Claude Code에서 시트와 연결하는 코드를 만들어 달라고 이어서 요청합니다.

프롬프트

> 너는 [가게 이름]의 문의 응대 담당자를 돕는 초안 작성기야. 문의 한 건과 기준 문서를 받으면 아래 형식으로만 답해.
>
> [규칙]
> 1. 답변 초안은 기준 문서에 있는 내용만 근거로 써. 문서에 없으면 지어내지 말고 "확인 후 연락드리겠습니다"로 답하고, 사람 확인 필요를 "예"로 해.
> 2. 금액·날짜·시간은 문서의 값을 그대로 옮겨. 계산이 필요하면 계산하지 말고 사람 확인 필요를 "예"로 해.
> 3. 문자 한 통 분량(공백 포함 200자 안팎)으로 쓰고, 첫 문장에 결론을 써.
> 4. 홍보 문구, 할인 안내, 다른 상품 권유는 넣지 마.
> 5. 말투는 [정중하지만 간결하게]. 호칭은 문의에 적힌 이름을 쓰고 없으면 "고객님"으로 해.
> 6. 환불, 위약금, 항의, 안전·건강 관련 문의는 초안을 쓰되 사람 확인 필요를 반드시 "예"로 해.
>
> [출력 형식]
> - 유형:
> - 긴급도:
> - 근거 항목 번호:
> - 답변 초안:
> - 사람 확인 필요: 예/아니오 (이유 한 줄)
>
> [기준 문서]
> [실습 1에서 만든 문서를 붙여넣기]
>
> [문의]
> [문의 내용을 붙여넣기. 이름·연락처는 가짜 값으로 바꿀 것]

문의 다섯 건으로 시험합니다. 이 중 두 건은 기준 문서에 없는 질문, 한 건은 환불 질문으로 준비합니다. 기준 문서에 없는 질문에 지어낸 답이 나오면 규칙 1을 더 구체적으로 고쳐 다시 시험합니다.

실습 3. 자동 발송 기준표 만들기

프롬프트

> 위에서 시험한 문의 5건의 결과를 보고, 어떤 유형은 자동 발송해도 되고 어떤 유형은 반드시 검토해야 하는지 표로 정리해 줘. 열은 문의 유형, 틀렸을 때의 피해, 자동 발송 가능 여부, 이유 네 개로 해. 확신이 서지 않는 유형은 검토 후 발송으로 분류해 줘.

정리

- 문의 응대 자동화는 수집, 기록, 분류, 초안, 검토, 발송, 결과 기록의 흐름이며, 사람은 쓰는 사람에서 승인하는 사람으로 역할이 바뀝니다.
- 초안의 품질은 기준 문서가 정합니다. 근거 번호를 붙이게 하고, 근거가 없으면 보류하게 합니다.
- 환불·일정 변경처럼 틀렸을 때 비용이 큰 답은 검토 후 발송으로 두고, 자동 발송은 수정 없이 승인된 유형부터 천천히 넓힙니다.
- 문자 발송 서비스는 API, 발송 결과 조회, 발신번호 등록 절차, 요금 구조를 기준으로 고르고, API 키는 코드나 채팅에 쓰지 않습니다.
- 발신번호 사전등록, 광고 수신 동의, 야간 광고 제한, (광고) 표시와 수신 거부 안내, 개인정보 최소화는 발송 전에 공식 자료로 확인합니다.

14장

데이터 분석

표 데이터를 Claude에게 주고 분석을 맡기는 방법을 다룹니다. 도구를 다루는 법보다 먼저 "무슨 결정을 하려고 보는가"를 정하고, 결과를 맹신하지 않고 검증하는 습관까지 이어서 익힙니다.

분석은 결정에서 시작한다

데이터를 받으면 곧바로 "분석해 줘"라고 쓰고 싶어집니다. 그러면 AI는 평균, 합계, 증감률 같은 무난한 숫자를 늘어놓습니다. 읽을 때는 그럴듯하지만 결정에는 쓸 수 없는 결과가 되기 쉽습니다.

분석 요청 전에 아래 세 가지를 한 문장씩 적어 둡니다.

- **결정**: 이 분석으로 무엇을 정할 것인가. 예를 들어 "다음 달 재고를 늘릴 상품을 고른다"
- **기준**: 어떤 숫자가 어떻게 나오면 결정이 달라지는가. 예를 들어 "두 달 연속 판매량이 늘고 재고가 보름치 미만인 상품"
- **대상 기간과 범위**: 어느 기간, 어느 지점, 어느 상품군을 볼 것인가

결정이 없으면 분석은 끝나지 않습니다. 결정이 있으면 필요 없는 열과 필요 없는 질문이 걸러집니다.

> **핵심** 좋은 분석 요청은 "이 데이터를 분석해 줘"가 아니라 "이 결정을 하려는데, 이 기준으로 판단할 근거를 이 데이터에서 찾아 줘"입니다.

데이터 준비

AI가 읽기 쉬운 표는 사람이 보기에도 단순한 표입니다. 파일을 넘기기 전에 다음을 정리합니다.

개인정보와 민감 정보 지우기

이름, 연락처, 주소, 이메일, 계좌·카드 번호, 주민등록번호는 분석에 필요한 경우가 드뭅니다. 필요 없다면 열 자체를 지웁니다. 고객별 구분이 필요하면 이름 대신 "C001" 같은 임시 번호로 바꿉니다. 임시 번호와 실제 고객의 대응표는 별도 파일로 보관하고 AI에게 주지 않습니다.

파일에 넣기 전에 지울 것을 네 가지로 점검하는 방법은 8장을 참고합니다.

회사 안의 자료라면 조직의 정보 보안 규정을 먼저 확인합니다. 어떤 자료를 외부 AI 서비스에 올려도 되는지는 조직마다 다릅니다.

열을 정돈하기

- 첫 줄에 열 이름 한 줄만 둡니다. 병합된 제목 칸, 여러 줄 머리글은 풀어 줍니다.
- 한 열에는 한 종류의 값만 넣습니다. 금액 열에 "10,000원"과 "만 원"이 섞여 있으면 하나로 맞춥니다.
- 날짜 형식을 통일합니다. "2026-11-07" 형식으로 맞추면 오해가 줄어듭니다.
- 빈칸의 의미를 정합니다. 값이 없는 것인지, 0인지, 모르는 것인지 표시합니다.
- 합계 줄과 소계 줄은 지웁니다. 합계는 AI가 직접 계산하게 하는 편이 검증하기 쉽습니다.
- 같은 대상의 표기를 맞춥니다. "A상품", "A 상품", "a상품"이 따로 집계되지 않게 합니다.

입문 과정 3회차에서 다룬 것처럼 엑셀 파일은 CSV로 저장하면 읽기 쉬워집니다. 다만 CSV에는 서식과 수식이 저장되지 않으므로, 수식 결과가 필요하면 값으로 붙여 넣은 뒤 저장합니다.

Claude에 파일 주고 분석 시키기

Claude 앱에서는 대화창에 CSV 같은 표 파일을 첨부해 분석과 시각화를 요청할 수 있습니다. 코드를 실행해 계산하고 차트 이미지나 결과 파일을 만드는 기능은 설정에서 켜는 방식이며, 공식 안내 기준으로 설정 이름은 "코드 실행 및 파일 생성"(새 화면에서는 "파일 만들기 및 편집")입니다. Team·Enterprise 플랜에서는 조직 관리자가 이 설정을 관리합니다(2026년 11월 기준). 메뉴가 보이지 않으면 관리자 설정을 먼저 확인합니다. 이름과 위치는 바뀔 수 있으므로 Claude 파일 생성 및 편집 안내에서 현재 상태를 확인합니다.

코드 실행이 켜져 있으면 Claude는 말로 추측하는 대신 실제 계산을 돌립니다. 표의 합계를 말로 더하는 것보다 훨씬 믿을 만합니다. 그렇더라도 "코드가 돌았다"와 "결과가 맞다"는 다른 이야기이므로 뒤의 검증 절을 따릅니다.

요청의 순서

한 번에 모두 시키지 않고 세 단계로 나눕니다.

1. **점검**: 표의 구조와 문제를 먼저 보게 합니다. 열 이름, 행 수, 빈칸, 이상한 값을 알려 달라고 하고 아직 분석은 시키지 않습니다.

2. **질문 확정**: 점검 결과를 보고 결정과 기준을 말하며 분석할 질문을 정합니다.
3. **분석과 정리**: 질문별로 계산하게 하고, 결과를 표와 문장으로 받습니다.

점검 단계를 건너뛰면 AI가 이상한 값을 말없이 걸러 내거나 바로잡은 채 분석할 수 있습니다. 어떤 행을 제외했는지 항상 알려 달라고 요청합니다.

차트 만들기

차트는 읽는 사람이 한눈에 결론을 알 수 있어야 합니다. 요청할 때는 차트의 종류보다 "무엇을 비교하려는지"를 말합니다.

- 시간에 따른 변화는 선 그래프, 항목 간 비교는 막대 그래프가 기본입니다.
- 비율은 항목이 너무 많지 않을 때만 원형 대신 가로 막대로 보여 주면 읽기 쉽습니다.
- 제목에는 결론을 씁니다. "월별 매출"보다 "3분기에 B지점 매출이 가장 크게 늘었다"가 낫습니다.
- 축 이름과 단위를 반드시 표시하게 합니다.
- 색은 많이 쓰지 않고, 강조할 항목 하나만 다른 색으로 두게 합니다.

차트를 파일로 받아 보고서나 슬라이드에 붙일 수도 있습니다. 파일 형식은 이미지와 표 파일 중에서 용도에 맞게 고르고, 한글이 깨지지 않았는지 반드시 눈으로 확인합니다.

결과 검증하기

AI의 분석에서 가장 위험한 것은 틀린 숫자가 매끈한 문장에 담겨 나오는 경우입니다. 다음 네 가지 검증을 습관으로 만듭니다.

합계 맞춰 보기

- 원본의 총 행 수와 분석에 쓴 행 수가 같은지 확인합니다. 다르면 제외된 행의 이유를 받습니다.
- 항목별 합계를 모두 더하면 전체 합계와 같은지 확인합니다.
- 전체 합계는 표 프로그램에서 직접 한 번 더 계산해 봅니다. 한 번의 합계 확인으로 열이 통째로 빠진 오류를 잡을 수 있습니다.

행 몇 개를 직접 대조하기

AI가 "가장 많이 팔린 상품은 A"라고 하면, 원본에서 A의 판매 행을 직접 찾아 몇 개를 더해 봅니다. 결과에 오른 상위 항목과 하위 항목에서 각각 두세 개씩 확인합니다. 숫자가 맞으면 계산 방식은 대체로 믿을 수 있고, 하나라도 다르면 전체를 다시 의심합니다.

같은 질문을 다른 방식으로 다시 하기

같은 결론을 다른 경로로 얻을 수 있는지 봅니다. 예를 들어 지점별 합계로 전체를 계산한 값과 전체 행을 한꺼번에 계산한 값이 같은지 묻습니다. 새 대화에서 같은 파일과 같은 질문을 주고 결과가 일치하는지 보는 방법도 있습니다.

해석이 데이터를 넘지 않는지 보기

계산은 맞아도 해석이 과할 수 있습니다. "매출이 올랐다"는 사실이고 "광고 덕분이다"는 추측입니다. 요청할 때 "데이터로 확인된 사실과 추정을 나누어 써 달라"고 하고, 추정에는 근거가 되는 열을 적게 합니다. 표본이 작거나 기간이 짧으면 결론을 약하게 쓰게 합니다.

계산은 수식으로 남기기

비율과 차이를 AI가 계산해 값만 적게 하면 나중에 어떻게 나온 숫자인지 확인할 수 없습니다. 엑셀 수식으로 남기면 누구나 같은 값을 다시 얻을 수 있습니다. 원자료는 한 시트에 옮기고, 실적, 차이, 판정은 그 시트를 가리키는 수식으로 계산 시트에 만들게 합니다. 값이 비어 있으면 직접 계산하지 말고 멈춰서 알리게 하고, 이미 수식이 들어 있는 칸은 건드리지 않게 합니다. 아래는 가상의 매출표에 쓰는 요청 예입니다.

프롬프트

> 목표 매출 대비 실적표를 만들어 줘. 원자료는 '원자료' 시트에 옮기고, 실적, 차이, 판정은 원자료 시트를 가리키는 엑셀 수식으로 만들어 줘. 값이 비어 있으면 직접 계산하지 말고 멈춰서 알려 줘. 이미 수식이 있는 칸은 건드리지 마.

지표, 계획, 실적, 차이, 판정, 출처, 원인을 나란히 놓는 계획 대비 실적표의 구조는 17장에서 다룹니다.

월간 보고서 루틴

한 번 만든 분석은 매달 같은 방식으로 돌릴 수 있습니다. 루틴으로 만들면 매번 처음부터 시키지 않아도 됩니다.

1. **입력 규칙을 문서로 남깁니다.** 어떤 파일을 어느 열 구성으로 넣는지 적습니다.
2. **분석 질문 목록을 고정합니다.** 매달 같은 다섯 가지 질문을 던집니다. 이번 달 합계, 전월 대비 변화, 상위·하위 항목, 이상한 변동, 다음 달에 결정할 것입니다.
3. **보고서 틀을 고정합니다.** 첫 줄 결론, 핵심 숫자 세 개, 차트 두 개, 확인이 필요한 점 순서로 정합니다.
4. **검증 단계를 포함합니다.** 합계 대조 결과를 보고서 끝에 표로 남깁니다.
5. **비교를 위해 지난달 결과를 보관합니다.** 월별 요약 표를 한 장의 시트에 계속 쌓습니다.

이 루틴을 Claude Code의 지시 파일이나 스킬로 만들면 한 줄로 실행할 수 있습니다. 스킬은 11장에서 다룹니다. 처음 몇 달은 사람이 결과를 모두 확인하고, 안정되면 검증 표만 보는 방식으로 줄여 갑니다.

민감한 데이터를 다룰 때의 주의

- 개인을 식별할 수 있는 정보는 올리기 전에 지웁니다. 지웠다고 생각한 열이 다른 열과 합쳐져 개인을 알아볼 수 있게 되는 경우가 있으므로, 연령대·지역·직업이 모두 들어 있는 소규모 표는 한 번 더 점검합니다.
- 급여, 평가, 건강, 재무 같은 민감 자료는 조직의 허가 없이 올리지 않습니다.
- 대화 내용이 학습에 쓰이는지, 얼마나 보관되는지는 요금제와 설정에 따라 다릅니다. 지원 센터의 최신 안내를 확인합니다.
- 분석 결과 파일에도 원본의 민감 정보가 남아 있을 수 있습니다. 공유하기 전에 열어서 확인합니다.
- 결과를 근거로 사람에 대한 판단(채용, 평가, 신용)을 내리는 일은 AI의 계산에 맡기지 않습니다.

실습

실습 1. 가상의 월 매출 CSV 만들고 점검하기

가상의 데이터를 만듭니다. 실제 업무 자료로 연습할 때는 개인정보를 모두 지운 뒤 사용합니다.

프롬프트

> 가상의 소규모 온라인 상점의 최근 3개월 판매 기록 CSV를 만들어 줘. 실제 회사나 사람 이름은 쓰지 마.
> 열은 주문일(YYYY-MM-DD), 주문번호, 상품명, 상품분류, 수량, 단가, 결제수단, 지역이야.
> 행은 약 200개로 하고, 점검 연습을 위해 일부러 아래 문제를 몇 개씩 섞어 줘.
> - 상품명 표기 차이(띄어쓰기, 대소문자)
> - 비어 있는 수량 칸
> - 수량이 음수인 행
> - 같은 주문번호가 중복된 행
> 어떤 문제를 몇 개 섞었는지는 마지막에 따로 적어서, 내가 나중에 점검 결과와 비교할 수 있게 해 줘.

받은 CSV를 파일로 저장한 뒤 새 대화에 첨부하고 다음을 요청합니다.

프롬프트

> 첨부한 CSV를 점검만 해 줘. 열 이름, 행 수, 빈칸, 같은 대상이 다르게 적힌 곳, 이상한 값, 중복 주문번호를 알려 줘. 아직 파일을 고치지 말고 분석도 하지 마.

실습 2. 월간 보고서 만들기

점검에서 나온 문제를 정리한 뒤 보고서를 만듭니다.

프롬프트

> 방금 점검한 문제를 아래 규칙으로 처리해서 분석해 줘.
> - 상품명 표기 차이는 하나로 통일해.
> - 수량이 비어 있는 행과 음수인 행은 분석에서 제외하되, 제외한 행의 수와 주문번호를 마지막에 알려 줘.
> - 중복 주문번호는 첫 번째 행만 남겨.
>
> 결정: 다음 달에 재고를 늘릴 상품분류 한 가지를 고르려고 해.
> 기준: 3개월 동안 판매 수량이 매달 늘어난 상품분류를 우선 봐.
>
> 월간 보고서를 아래 순서로 써 줘.
> 1. 첫 줄 결론 한 문장
> 2. 핵심 숫자 세 개(월별 총 매출, 전월 대비 변화, 상품분류별 상위 3개)
> 3. 차트 두 개(월별 매출 추이, 상품분류별 3개월 판매 수량 비교). 제목은 결론 문장으로, 축 이름과 단위를 표시해 줘.
> 4. 데이터로 확인된 사실과 추정을 나눠서 정리
> 5. 검증 표: 원본 행 수, 분석에 쓴 행 수, 제외한 행 수, 월별 매출의 합과 전체 매출의 일치 여부

실습 3. 직접 검증하기

프롬프트

> 위 보고서의 수치를 내가 직접 검증하려고 해. 상품분류별 총 매출을 계산한 표를 따로 만들어 주고, 상위 3개 분류에서 임의로 5행씩 골라 주문번호, 수량, 단가, 계산한 금액을 나란히 보여 줘. 내가 원본 CSV에서 같은 행을 찾아 확인할 수 있게 해 줘.

이 표의 금액을 원본에서 직접 계산해 맞는지 확인하고, 하나라도 다르면 AI에게 원인을 묻습니다.

선택 과제로, 보고서의 합계 칸을 값이 아닌 수식으로 바꿔 보고 원자료를 고쳤을 때 합계가 따라 바뀌는지 확인합니다.

정리

- 분석은 결정과 기준을 먼저 적고 시작합니다. "분석해 줘"보다 "이 결정을 위해 이 기준으로 근거를 찾아 줘"가 낫습니다.
- 파일을 넘기기 전에 개인정보를 지우고, 열 이름과 형식을 통일하고, 합계 줄을 없앱니다.
- 요청은 점검, 질문 확정, 분석의 세 단계로 나누고, 제외한 행은 반드시 보고하게 합니다.
- 결과는 행 수와 합계를 대조하고, 몇 개의 행을 직접 확인하고, 사실과 추정을 나누어 읽습니다.
- 월간 보고서는 입력 규칙, 질문 목록, 보고서 틀, 검증 표를 고정해 루틴으로 만듭니다.

15장

딥리서치

웹을 검색해 여러 출처를 모으고 종합하는 조사 작업을 Claude에게 맡기는 방법을 다룹니다. 조사는 AI로 속도를 크게 높일 수 있지만 틀리기도 쉬우므로, 질문 설계와 출처 검증이 이 장의 중심입니다.

웹 검색과 Research

Claude는 두 가지 방식으로 웹의 정보를 가져올 수 있습니다. 두 방식의 쓰임새는 다릅니다.

- **웹 검색**: 질문 하나에 필요한 정보를 검색해 답에 반영합니다. 최신 소식 확인, 단일 사실 확인에 알맞습니다.
- **Research**: 하나의 큰 질문에 대해 여러 출처를 찾아 종합한 보고서를 만드는 기능입니다. 단순 검색보다 시간이 더 걸립니다.

공식 안내 기준으로 Research는 Claude 웹, 데스크탑 앱, 모바일 앱에서 쓸 수 있고 유료 플랜(Pro, Max, Team, Enterprise 등)에서 제공되며, 웹 검색이 켜져 있어야 동작합니다. 대화창의 더하기(+) 메뉴에서 켜는 방식이며, 웹이나 데스크탑 앱에서 시작하면 연결된 서비스(커넥터)의 내용도 조사에 쓸 수 있습니다. 모바일에서 시작하면 알림 없이 웹만 검색하고 연결된 서비스는 건너뜁니다. 사용 한도는 일반 대화와 같은 한도에서 차감되지만 더 빨리 소진될 수 있습니다(2026년 11월 기준). 현재 제공 범위와 켜는 방법은 공식 문서 Claude에서 Research 사용하기에서 확인합니다.

조직의 관리자가 기능을 제한해 둔 경우도 있습니다. 메뉴가 보이지 않으면 요금제 문제인지, 설정 문제인지, 관리자 설정인지를 차례로 확인합니다.

어떤 기능을 쓸까

상황	알맞은 방식
한 가지 사실, 최신 소식 확인	웹 검색
여러 출처의 비교·종합이 필요한 시장 조사	Research
내 문서와 웹 정보를 함께 보는 조사	Research(연결된 서비스 사용)
내가 이미 가진 자료만 정리	파일 첨부 대화(검색 불필요)

조사 세 갈래

조사는 한 가지 방법으로 하지 않습니다. 확인할 것마다 알맞은 갈래가 따로 있습니다.

갈래	알맞은 것	특징
내 자료 읽기	받은 자료, 사내 자료, 고객이 요구한 내용	가장 먼저 하고, 가장 믿을 만합니다
웹 검색	숫자 하나, 날짜 하나, 최신 규정 하나	빨리 확인하고 출처를 바로 엽니다
딥리서치	여러 출처를 모아 비교하는 큰 질문	오래 걸리므로 질문을 좁혀 맡깁니다

가상의 예로 소규모 제조 업체가 신규 포장재 도입을 검토한다고 해 보겠습니다. 우리 회사의 지난 구매 내역과 거래처의 요구 사항은 내 자료입니다. 포장재 관련 최신 규정 한 가지는 웹 검색으로 확인합니다. 다른 업체들의 도입 사례와 동향 비교는 딥리서치에 맡깁니다.

내 자료로 채울 칸은 딥리서치에 맡기지 않습니다. 인터넷에는 우리 회사의 실적도, 거래처의 요구도 없으므로 AI가 그럴듯하게 지어낼 뿐입니다.

질문 설계

조사 결과의 질이 질문에서 정해집니다. "전기차 시장 조사해 줘"는 질문이 아니라 주제입니다. 질문은 다음 네 가지를 갖춰야 합니다.

목적

조사를 어디에 쓰는지 밝힙니다. 투자 판단, 신규 서비스 기획, 제안서 근거, 내부 보고는 필요한 깊이와 증거의 무게가 다릅니다. "다음 분기 신규 서비스 진입 여부를 논의하는 회의 자료"처럼 쓰임을 한 문장으로 적습니다.

범위

대상 지역, 산업, 고객군을 좁힙니다. 범위가 넓으면 얕고 일반적인 결과가 나옵니다. 포함하지 않을 것도 함께 적습니다.

기간

어느 시점부터 어느 시점까지의 자료를 볼지 정합니다. 최신성이 중요한 주제는 "최근 2년 안의 자료를 우선"처럼 적고, 오래된 자료를 쓸 때는 연도를 밝히게 합니다. 조사하는 날짜도 알려 줍니다. 오늘이

며칠인지 AI가 정확히 모를 수 있기 때문입니다.

출처 기준

어떤 출처를 믿을지 미리 정합니다.

- 1순위: 법령, 정부·공공기관 통계, 공식 보고서, 기업의 공시 자료
- 2순위: 학술 논문, 신뢰할 만한 언론의 원문 기사, 업계 협회 자료
- 낮음: 개인 블로그, 출처 없는 요약, 광고성 비교 글, 다른 글을 옮겨 쓴 글

"1순위만으로 답이 안 되면 2순위를 쓰되 구분해서 표시하라"고 요청하면 출처의 무게를 보고서에서 알아볼 수 있습니다.

질문을 쪼개기

큰 조사는 하위 질문으로 쪼개면 결과가 좋아집니다. 시장 조사라면 시장 규모, 성장 요인, 주요 경쟁자, 규제, 고객의 불만 같은 하위 질문으로 나누고, 질문마다 필요한 증거의 종류를 적습니다. 실습에서 이 질문 세트를 만듭니다.

의뢰 전 점검 네 가지

질문을 딥리서치에 넘기기 전에 네 가지를 점검합니다. 범위가 분명해지면 시간과 사용량의 낭비도 줄어듭니다.

1. 질문이 구체적인가: "물류 업계 AI 활용 현황"보다 "국내 중소 물류 업체의 배송 경로 최적화 도입 사례와 규모"가 낫습니다(가상의 예).
2. 공개 자료로 답이 나올 만한가: 비공개 내부 수치를 묻는 질문은 조사로 풀리지 않습니다.
3. 출처 기준이 있는가: 1순위, 2순위, 낮음 기준을 함께 적었는지 봅니다.
4. 너무 많지 않은가: 처음에는 서너 개로 시작합니다.

시작 버튼을 누르기 전에 조사 계획을 먼저 내놓게 하고, 사람이 읽고 고친 뒤에 시작시킵니다. 계획을 보지 않고 시작하면 엉뚱한 곳을 조사하기 쉽습니다. 문서 작업에서는 계획서의 '조사할 것' 항목을 질문 형태로 써 두었다가 그대로 의뢰서로 쓸 수 있습니다. 이 흐름은 16장에서 다룹니다.

출처 검증

출처 링크가 붙어 있다는 것이 사실이라는 보증은 아닙니다. 출처 표시 방식은 기능과 시기에 따라 다르므로, 이 책은 "출처를 링크로 달게 요청하고 사람이 직접 열어 본다"는 원칙으로 설명합니다.

검증 순서

1. **링크가 열리는지** 확인합니다. 열리지 않거나 다른 페이지로 가면 의심합니다.
2. **문장이 실제로 그 페이지에 있는지** 확인합니다. 출처에는 있는 말인데 요약 과정에서 의미가 바뀐 경우가 많습니다.
3. **출처의 성격**을 봅니다. 누가 썼고 언제 썼고 무엇을 근거로 했는지 확인합니다. 원문이 따로 있다면 원문에 가까운 쪽을 믿습니다.
4. **숫자는 원문에서 직접** 봅니다. 단위(원, 천 원, 달러), 기준 연도, 조사 대상이 요약에서 빠지는 경우가 흔합니다.
5. **두 번째 출처**로 교차 확인합니다. 핵심 결론은 서로 다른 출처에서 같은 이야기가 나와야 합니다. 같은 기사를 인용한 여러 글은 한 출처로 칩니다.

지어낸 출처에 대응하기

AI는 존재하지 않는 논문, 보고서, 통계, 기관 이름을 그럴듯하게 만들어 낼 수 있습니다. 이를 환각이라고 합니다. 웹 검색을 써도 사라지지는 않습니다. 이런 신호가 보이면 특히 조심합니다.

- 제목은 그럴듯한데 링크가 없거나 열리지 않습니다.
- 저자·기관·연도가 구체적인데 검색해도 나오지 않습니다.
- 수치가 지나치게 딱 떨어지고 출처가 "업계 관계자"나 "여러 보고서"처럼 모호합니다.
- 질문한 내용에 맞춰 놓은 듯 완벽하게 들어맞는 통계가 나옵니다.

의심스러운 항목은 보고서에서 빼거나 "확인되지 않음" 표시를 붙입니다. AI에게 "출처를 확인하지 못한 문장은 목록으로 따로 알려 달라"고 요청하는 것도 도움이 되지만, 그 목록이 비어 있다고 안심해서는 안 됩니다. 최종 확인은 사람이 합니다.

7축 팩트체크로 검토하기

보고서를 쓴 뒤에는 FactCheck 스킬(부록 B)이 쓰는 7개 축으로 검토합니다. 스킬을 설치하지 않았어도 이 일곱 가지를 질문으로 바꿔 쓸 수 있습니다.

축	확인할 질문
사실 정확성	수치, 날짜, 고유명사, 인용이 원문과 일치하는가
논리 정합성	장 사이에 모순이 없는가, 원인과 결과가 실제로 이어지는가
가정	글에 깔린 전제가 현실적인가, 전제가 글에 밝혀져 있는가
수치·데이터 무결성	계산이 맞는가, 단위와 기준 연도가 일관되는가
맥락	출처의 품질과 조사 방법이 결론을 받쳐 주는가, 빠진 경쟁자나 반대 증거는 없는가
법률·규제	인용한 법령이 현행인가, 규제가 바뀐 것은 없는가
표현·과장	"반드시", "압도적" 같은 절대 표현이나 일부 사례만 고른 흔적이 없는가

FactCheck 스킬은 보수적인 기조로 의심되는 주장을 모두 표시하고, 출처 없는 수치는 자동으로 표시 대상으로 삼으며, 검증하지 못한 주장은 "미확인"으로 남깁니다. 심각도를 적색, 황색, 녹색으로 나눠 보고서를 내고, 적색과 황색을 반영한 개선본을 만듭니다. 이 방식은 사람이 직접 검토할 때도 그대로 쓸 수 있습니다. 확인하지 못한 것을 사실처럼 쓰지 않고 "미확인"으로 표시하는 습관이 가장 중요합니다.

검증은 보고서를 쓴 AI와 다른 새 대화에서 하는 편이 좋습니다. 자기가 쓴 글에는 관대해지기 쉽기 때문입니다. 19장의 자가검증과 같은 원리입니다.

보고서 구조

조사 보고서는 읽는 사람이 결론부터 알 수 있게 씁니다.

1. **한 줄 결론**: 질문에 대한 답을 한두 문장으로 씁니다.
2. **조사 개요**: 목적, 범위, 기간, 조사 날짜, 출처 기준을 적습니다.
3. **핵심 발견**: 서너 개로 줄이고, 각각 근거와 출처를 붙입니다.
4. **하위 질문별 상세**: 질문마다 확인된 사실, 추정, 확인하지 못한 것을 구분합니다.
5. **한계와 불확실성**: 찾지 못한 자료, 출처끼리 충돌한 부분, 오래된 자료를 적습니다.
6. **다음 행동**: 이 결과로 정할 일과 추가로 확인할 일을 적습니다.
7. **출처 목록**: 번호를 붙여 본문과 연결합니다.

핵심 숫자는 표로 따로 모아 두면 검토가 쉽습니다. 열은 숫자, 단위, 기준 시점, 출처, 발행일 다섯 가지입니다. 출처를 못 찾은 내용은 `[출처 없음]`으로 표시하고, 그 줄은 본문에 쓰지 않습니다.

"한계와 불확실성" 절이 비어 있는 조사 보고서는 믿기 어렵습니다. 어떤 조사도 못 찾은 것이 있기 마련이고, 그것을 적는 것이 보고서의 신뢰를 높입니다.

인용 규칙

- 문장마다 출처 번호를 붙입니다. 문단 끝에 몰아서 붙이지 않습니다.
- 출처 목록에는 제목, 발행 기관 또는 저자, 발행 연도, 주소, 확인한 날짜를 적습니다.
- 숫자에는 기준 연도와 단위를 함께 씁니다.
- 원문을 길게 옮기지 않고, 필요하면 짧게 인용하거나 내 말로 요약해 출처를 답니다. 저작권이 있는 글을 통째로 옮기지 않습니다.
- 확인하지 못한 문장은 "미확인"이라고 적고, 추정은 "추정"이라고 적습니다.
- AI가 쓴 보고서를 외부에 제출할 때는 사람이 모든 핵심 수치의 출처를 직접 열어 본 뒤에 냅니다.

조사 흐름 요약

1. 질문 세트를 설계합니다(목적, 범위, 기간, 출처 기준, 하위 질문).
2. Research 또는 웹 검색으로 조사합니다.
3. 보고서를 구조에 맞게 정리합니다.
4. 새 대화에서 7축으로 검토합니다.
5. 핵심 수치와 출처를 사람이 직접 확인합니다.
6. 확인하지 못한 부분을 한계로 적고 마무리합니다.

실습

실습 1. 시장 조사 질문 세트 설계하기

조사하고 싶은 주제를 하나 정하고(예: 가상의 예로 "동네 소규모 학원이 쓸 수 있는 출결 관리 도구 시장"), 아래 프롬프트로 질문 세트를 설계합니다. 아직 조사하지 않고 질문만 만듭니다.

프롬프트

> 나는 [내 직무]이고, [조사 주제]를 조사하려고 해. 아직 조사하지 말고 먼저 조사 질문 세트를 설계해 줘.
>
> [내가 아는 정보]
> - 조사 결과를 쓰는 곳: [예: 다음 분기 신규 서비스 진입 여부를 논의하는 회의 자료]
> - 대상 지역과 범위: [예: 국내, 소규모 사업장]
> - 보고 싶은 기간: [예: 최근 2년]
> - 오늘 날짜: [오늘 날짜]
>
> 아래 순서로 답해 줘.
> 1. 조사 목적을 한 문장으로 다시 써 줘. 모호하면 나에게 질문부터 해 줘.
> 2. 하위 질문 5~7개를 만들어 줘. 각 질문마다 필요한 증거의 종류(통계, 공식 문서, 사례, 사용자 의견 중)와 믿을 만한 출처 유형을 적어 줘.
> 3. 조사에서 제외할 범위를 제안해 줘.
> 4. 출처 기준을 1순위(법령, 공공 통계, 공식 보고서), 2순위(논문, 신뢰할 만한 언론 원문, 협회 자료), 낮음(개인 블로그, 광고성 글)으로 정리해 줘.
> 5. 조사가 끝났을 때 "충분하다"고 판단할 기준을 세 가지 제안해 줘.
> 6. 이 조사에서 환각이 생기기 쉬운 부분이 어디일지 짚어 줘.

실습 2. Research로 조사하고 출처 확인하기

질문 세트 중 하위 질문 두세 개만 골라 Research(또는 웹 검색)로 조사합니다. 결과가 나오면 다음 표를 직접 채웁니다.

프롬프트

> 방금 만든 조사 보고서에서 핵심 주장 5개를 골라 아래 표로 정리해 줘. 열은 주장, 출처 번호, 출처 주소, 원문에서 확인한 문장, 확인 상태(확인됨/부분 확인/미확인)야. 원문에서 확인하지 못한 주장은 지어내지 말고 미확인으로 적어 줘.

표의 출처 주소를 열어 직접 확인하고, 일치하지 않는 항목을 따로 표시합니다.

실습 3. 새 대화에서 7축 검토하기

프롬프트

> 아래 보고서를 팩트체크해 줘. 사실 정확성, 논리 정합성, 가정, 수치·데이터 무결성, 맥락, 법률·규제, 표현·과장의 7개 축으로 검토하고, 문제는 적색(오류·출처 부재), 황색(출처 미비·과장), 녹색(표현 개선)으로 나눠 줘. 근거 없는 수치는 모두 표시하고, 확인하지 못한 것은 사실처럼 쓰지 말고 [미확인]으로 남겨 줘. 마지막에 이 보고서를 회의에 쓸 수 있는지 한 줄로 판정해 줘.
>
> [보고서 전문 붙여넣기]

실습 4(선택). 확인할 것을 세 갈래로 나누기

실습 1의 하위 질문과 그 밖에 확인할 것을 모두 적고, 내 자료, 웹 검색, 딥리서치 중 어느 갈래로 풀지 분류합니다. 딥리서치로 보낼 질문은 서너 개로 줄이고, 의뢰 전 점검 네 가지를 통과하는지 봅니다.

정리

- 확인할 것은 내 자료, 웹 검색, 딥리서치 세 갈래로 나눕니다. Research는 여러 출처를 종합하는 큰 질문에 씁니다. 제공 범위는 수시로 바뀌므로 공식 문서로 확인합니다.
- 질문은 목적, 범위, 기간, 출처 기준을 갖추고 하위 질문으로 쪼갭니다. 의뢰 전에 구체성, 공개 자료 여부, 개수를 점검하고 조사 계획을 먼저 봅니다.
- 출처에 링크가 있어도 사실이라는 보증은 아닙니다. 링크를 열고, 문장을 대조하고, 숫자는 원문에서 확인하고, 핵심은 두 번째 출처로 교차 확인합니다.
- 지어낸 출처의 신호를 알아 두고, 확인하지 못한 내용은 "미확인"으로 남기며, 외부에 낼 보고서의 핵심 수치는 사람이 직접 확인합니다.
- 보고서는 결론, 개요, 핵심 발견, 상세, 한계, 다음 행동, 출처 순서로 쓰고, 7축 검토는 보고서를 쓴 대화와 다른 새 대화에서 합니다.

16장

문서 업무 전 과정: 제안서와 기획서

문서 한 건을 폴더 하나에서 기획, 조사, 초안, 검토, 양식 입히기, 최종 확인까지 끝내는 방법을 다룹니다. 규칙, 진행, 이유를 파일로 남기고, 원본 하나에서 목적별 문서를 뽑습니다. 7장에서 다룬 단일 원본(SSOT)과 md 원본의 변환 사다리를 전제로 합니다. 이 장의 예시는 모두 가상의 예이며, 중소 제조업체가 거래처에 새 설비 점검 서비스를 제안하는 상황을 가정합니다.

전체 흐름과 남는 파일

제안서 한 건은 아래 여덟 단계로 진행합니다. 단계마다 손에 남는 파일이 있어서, 중간에 세션이 끊겨도 파일에서 이어 갈 수 있습니다.

단계	하는 일	남는 파일
1. 폴더 세팅	규칙, 진행, 이유를 적는 파일을 만듭니다	CLAUDE.md, PROGRESS.md, HISTORY.md
2. 되묻기와 계획서	Claude가 질문하고, 답을 모아 계획서를 씁니다	계획서 md, 같은 이름의 html
3. 조사	내 자료, 웹 검색, 딥리서치로 나눠 조사합니다	조사 결과 md, html
4. 초안	계획서 목차와 조사 결과로 초안을 씁니다	초안 v1
5. 사람 검토와 코멘트	사람이 읽고 코멘트를 달아 반영시킵니다	초안 v2, HISTORY.md 갱신
6. 양식 규칙과 원본 확정	양식 파일에서 규칙을 뽑고 원본을 확정합니다	양식규칙.md, 원본 md
7. 아트보드로 보고 고치기	쪽 모양으로 펼쳐 번호로 짚어 고칩니다	아트보드 html
8. 최종 파일 뽑기	docx, pptx, PDF를 뽑아 직접 열어 확인합니다	최종 파일

대화창에서 곧바로 "제안서 써 줘"라고 시키는 방식에는 세 가지 문제가 있습니다. 대화가 끝나면 맥락이 흩어집니다. AI가 아는 것만으로 쓰기 때문에 우리 사정이 빠집니다. 뽑아 낸 파일을 손으로 고치게 되어

파일마다 내용이 달라집니다. 이 장의 여덟 단계는 이 세 문제를 막기 위한 장치입니다.

---

1. 작업 폴더의 기억: 규칙, 진행, 이유

Claude는 어제 한 일을 기억하지 못합니다. 대신 파일로 적어 두면 읽고 기억합니다. 하네스는 작업 환경을 파일로 만들어 두는 일입니다. 잘 쓴 지시문은 그 대화에서 끝나지만, 폴더에 남은 파일은 다음 세션과 다음 사람에게도 작동합니다.

폴더에는 세 파일을 둡니다.

파일	역할	담는 것
CLAUDE.md	폴더의 규칙	맨 위 한 줄, 파일 위치, 표기 규칙
PROGRESS.md	지금 어디까지 했는가	목표, 상태, 결정, 다음, 막힌 것 다섯 칸
HISTORY.md	무엇을 왜 그렇게 정했는가	결정과 이유를 한 줄씩

PROGRESS.md는 다음 시작점이고, HISTORY.md는 되돌아갈 때의 근거입니다. 둘을 나누면 진행 상태가 길어지지 않고, 이유가 사라지지도 않습니다.

CLAUDE.md의 맨 위 한 줄이 다음 세션의 행동을 좌우합니다. 아래 문장을 그대로 쓰면 됩니다.

프롬프트

> 작업을 시작하면 PROGRESS.md와 HISTORY.md를 먼저 읽고, 지금 상태와 다음 할 일을 한 줄로 요약한 뒤 시작한다.

그 아래에 같이 넣을 규칙은 세 가지입니다. 산출물은 산출물 폴더에, 자료는 자료 폴더에 둡니다. 모르는 것은 지어내지 말고 [확인 필요]로 남깁니다. 파일 이름은 이름_날짜_v번호 형식으로 짓습니다.

제대로 깔렸는지는 새 세션에서 "이어서 하자" 한마디로 확인합니다. Claude가 PROGRESS.md를 읽고 현재 상태를 먼저 말하면 정상입니다. CLAUDE.md는 Claude Code에서 세션을 시작할 때 읽히는 지속 지침이라고 공식 문서가 설명합니다(기억 관리 문서, 2026년 11월 기준). 다만 사용하는 환경에 따라 자동으로 읽히지 않을 수 있으므로, 이 확인 절차를 꼭 거칩니다.

새 일은 새 폴더에서 시작하고 규칙 파일만 옮겨 와 용도에 맞게 고칩니다. PROGRESS.md와 HISTORY.md는 새로 만듭니다. 규칙은 이어받고 기록은 이어받지 않는 것이 요령입니다.

2. 계획을 먼저 파일로 받는다

바로 본문을 쓰게 하지 않습니다. 계획을 파일로 받아 눈으로 확인하고 확정한 뒤에 씁니다. 계획서가 확정되기 전에는 본문을 쓰지 않는다는 것이 이 단계의 규칙입니다.

한 번에 완성된 계획을 요구하지 말고 Claude가 되묻게 합니다. 질문은 객관식으로, 한 번에 하나씩 받습니다. "모르겠다"고 답하면 Claude가 안을 내놓게 합니다. 답하는 과정에서 머릿속에만 있던 것이 정리되기 때문에, 질문이 끝난 뒤에 계획을 쓰게 하는 편이 낫습니다.

> **핵심** 처음부터 완성된 계획을 요구하지 말고, Claude가 묻고 내가 답하는 과정에서 계획을 만듭니다.

계획서에는 다음 칸을 둡니다.

- 제안 개요
- 읽는 사람과 결정 사항
- 목차안
- 조사할 것(질문 목록)
- 출처 기준
- 일정

조사할 것은 딥리서치에 그대로 넘길 수 있게 질문 형태로 씁니다. 아직 모르는 칸은 [확인 필요]로 남깁니다.

계획서는 두 가지 파일로 함께 받습니다. md는 고치기 위한 파일이고, 같은 이름의 html은 한눈에 보기 위한 파일입니다. html에는 조건을 붙입니다. 더블클릭으로 열리는 파일 하나여야 하고, 인터넷 연결 없이 열려야 하며, 배경은 밝고 글자는 진하게, 인쇄해도 잉크가 많이 들지 않게 합니다.

잘 통한 과정은 메타 프롬프트로 남깁니다. 메타 프롬프트는 프롬프트를 만들어 주는 프롬프트이며, 3장에서 다뤘습니다. 여기서는 방금 한 기획 과정을 주제만 바꿔 다시 쓸 수 있는 프롬프트 파일로 저장합니다. 주제 자리는 [ ]로 비워 두고, Claude가 물어봐야 할 질문 목록도 안에 넣습니다. 절차를 지나치게 세세하게 못 박지 말고, 목적과 지킬 규칙 위주로 씁니다. 이 파일은 가장 오래 쓰게 되는 산출물이어서, 다음 폴더에 이 파일 하나만 복사해 가면 같은 기획을 반복할 수 있습니다.

3. 조사 세 갈래와 딥리서치 의뢰

조사는 세 갈래로 나눕니다.

갈래	쓰는 때	요령
내 자료 읽기	우리 실적, 받은 자료, 거래처가 요구한 것	가장 먼저 하고, 가장 믿을 만합니다
웹 검색	숫자 하나, 날짜 하나를 빨리 확인할 때	출처를 바로 열어 봅니다
딥리서치	여러 출처를 모아 비교할 때	오래 걸리니 질문을 좁혀 맡깁니다

내 자료로 채울 칸은 딥리서치에 맡기지 않습니다. 우리 회사의 설비 점검 실적은 내 자료이고, 업계 동향 비교는 딥리서치의 일입니다.

딥리서치는 15장에서 자세히 다룹니다. 여기서는 계획서의 "조사할 것"을 의뢰서로 쓰는 연결만 봅니다. 의뢰서를 넘기기 전에 네 가지를 점검합니다.

1. 질문이 구체적인가. "업계 현황"이 아니라 "국내 중소 제조업의 설비 예방 점검 외주 사례와 규모"처럼 씁니다.
2. 공개 자료로 답이 나올 만한가.
3. 출처 기준이 적혀 있는가.
4. 너무 많지 않은가. 연습에서는 서너 개가 적당합니다.

딥리서치를 시작시키기 전에 Claude가 먼저 조사 계획을 내놓게 하고, 사람이 읽고 고친 뒤에 시작시킵니다. 그냥 두면 엉뚱한 것을 조사합니다. 출처는 공공기관, 공시, 주요 언론을 우선하고, 개인 블로그와 커뮤니티 글은 근거로 쓰지 않게 지시합니다.

결과는 질문별 요약, 핵심 숫자 표(숫자, 단위, 기준 시점, 출처, 발행일), 출처 목록, 확인하지 못한 것의 순서로 받습니다. 출처를 못 찾은 내용은 [출처 없음]으로 표시하게 하고, 그 줄은 제안서에 쓰지 않습니다. 이 결과도 같은 이름의 html로 함께 받으면 읽기 편합니다.

4. 초안과 사람 검토

초안은 계획서의 목차와 조사 결과 두 파일을 재료로 씁니다. 쓰기 전에 확인할 것이 있으면 먼저 묻게 합니다. 조사 결과에 없는 사실은 지어내지 말고 [확인 필요]로 남기게 하고, 숫자에는 출처를 괄호로 붙이게 합니다. 초안도 md와 html로 받습니다.

검토는 AI에게 맡기지 않습니다. 사람이 html을 읽으며 고칠 곳마다 쪽과 문단 위치를 적어 한 줄씩 코멘트를 달고, 그것을 Claude에게 넘겨 새 버전 파일로 저장하게 합니다. 무엇을 왜 고쳤는지는 HISTORY.md에 한 줄씩 남기게 합니다.

코멘트는 네 마디로 시작하면 짧게 쓸 수 있습니다.

마디	쓰는 때	코멘트 예(가상의 예)
딥리서치로 확인	사실이나 숫자가 의심될 때	3쪽 시장 규모, 출처가 오래됐으니 다시 확인
정합성	앞뒤 수치나 논리가 어긋날 때	5쪽 비용 가정이 앞쪽 표와 안 맞음
톤다운	어조가 셀 때	6쪽 둘째 문단, 단정 표현을 낮춰 주세요
보수적으로	전망이 낙관적일 때	7쪽 매출 전망을 보수적으로 다시 잡아 주세요

19장의 자가검증은 AI가 스스로 검증 질문을 만들게 하는 방법입니다. 이 장의 검토는 사람이 직접 읽고 코멘트로 짚는 방법이라는 점이 다릅니다. 둘은 겹쳐 써도 됩니다.

---

5. 원본 하나, 목적별 사양, 양식 규칙

흐름은 이렇습니다. 재료(자료 폴더, 조사 결과, 요청서)에서 원본 md를 만들고, 원본을 아트보드로 펼쳐 보고, 확정되면 docx, pptx, PDF로 뽑습니다. 내용은 원본에서만 고칩니다. 뽑은 파일은 언제든 다시 뽑을 수 있는 사본입니다. 뽑은 파일을 손으로 고치면 워드와 슬라이드의 숫자가 따로 놀고, 어느 것이 최신인지 헷갈리고, AI가 매번 docx를 다시 해석해야 합니다. md는 제목, 표, 목록을 기호로 적은 글 파일이어서 AI가 가장 정확히 읽습니다.

같은 원본에서 문서를 바꾸는 것은 내용이 아니라 사양입니다. 사양은 목적, 읽는 사람, 분량, 형식입니다.

문서	읽는 사람	분량과 형식	중심
제안서 본문	평가하는 사람	길게, 워드 문서	요구 항목 순서와 근거
결재용 기획서	결재권자	한 장, 워드 문서	결론 먼저, 요청 사항
발표 자료	발표를 듣는 사람	짧게, 슬라이드	장마다 한 메시지
공유 메일	팀원	몇 줄	결정, 일정, 할 일

원본은 그대로 두고 사양만 새로 주면 새 문서가 나옵니다. 남는 것은 문서 한 벌이 아니라 다시 쓸 수 있는 원본과 규칙입니다.

양식은 말보다 파일로 넘깁니다. 회사 양식 파일과 예전 문서를 주고, 규칙을 뽑아 양식규칙.md로 남기게 합니다. 담을 것은 다음과 같습니다.

- 쪽, 여백, 글꼴, 제목 단계와 번호 체계
- 표 모양, 강조 방식, 숫자와 단위 표기
- 문체: 문장 끝 형태, 기호 단계, 출처 다는 법
- 규칙마다 어느 파일 어디서 봤는지

워드 서식을 수치로 직접 줄 수도 있습니다. 본문 글자 크기와 줄 간격, 문단 사이 간격, 큰 제목과 작은 제목의 크기, 표 머리행 색, 여백, 쪽번호 위치, 이모지 금지 같은 항목입니다. 서식을 정하지 않으면 기본 글꼴과 여백이 쓰이고, 제목 크기가 들쭉날쭉해지고, 이모지가 깨질 수 있습니다. 회사 양식 docx가 있다면 열어 보고 그대로 맞추게 하는 편이 가장 확실합니다.

---

6. 아트보드로 보고 번호로 고치기

아트보드는 원본을 최종 쪽 모양으로 펼쳐 놓은 확인용 html입니다. 이 책에서는 도구 이름에 의존하지 않고 쪽 모양으로 펼친 html 확인 화면이라고 부르겠습니다. 양식규칙대로 쪽마다 펼치고 판 번호를 붙이면 "3번이 이상하다"처럼 짚을 수 있습니다. 완성된 docx를 열어 고치는 것보다, 이 화면을 보며 번호로 짚어 고치는 편이 빠릅니다.

무엇을 어디서 고치는지는 아래 표로 정리합니다.

고칠 것	고치는 곳	비고
내용, 숫자, 문장	원본 md	고친 뒤 아트보드를 다시 펼칩니다
배치, 줄바꿈, 표 모양	아트보드	늘 그렇게 할 것이면 양식규칙.md에도 반영합니다
글자 한두 개	마지막 docx	손으로 고친 뒤에는 다시 뽑지 않습니다

"아트보드 다시 펼쳐 줘" 한마디로 되도록 CLAUDE.md에 방법을 적어 둡니다. 다시 펼칠 때는 내가 고친 쪽 나누기와 표 배치를 그대로 두고 바뀐 내용만 반영하게 합니다. 본문의 근거 번호는 양식규칙대로 (참고 1)처럼 보이게 하고, 실제로 쓴 근거만 붙임 자료로 모으게 합니다. 확정되면 그때 형식을 골라 docx, pptx, PDF를 뽑습니다. 양식 docx를 쓰는 경우 쓴 절만 채우고, 안 쓴 절의 작성 안내 줄과 표지의 안내 문구는 지웁니다. 본문에 내부 근거 번호가 남지 않았는지도 확인합니다.

발표 자료도 같은 순서입니다. 문서에서 내용과 흐름을 이미 확인했으므로 스토리라인을 다시 짜지 않고 곧바로 아트보드로 만듭니다. 회사 슬라이드 양식의 색, 글꼴, 표지를 맞추게 하고, 아트보드에서 다 고친 뒤에 pptx로 뽑습니다. 슬라이드 번호와 화면 캡처로 한 장씩 고칩니다.

---

7. 눈을 뜨게 하고, 마지막은 사람이

AI에게 눈이 없는 것이 아니라 감고 있는 것에 가깝습니다. 파일이 만들어졌다는 것과 제대로 보인다는 것은 다릅니다. 만든 파일을 직접 열어 다음을 봅니다.

- 파일이 열리는가, 글꼴이 깨지지 않았는가
- 제목 번호와 목차의 쪽번호가 맞는가
- 표가 쪽 경계에서 잘리지 않았는가

이상한 곳은 화면을 캡처해 위치와 함께 Claude에게 넘깁니다. 캡처 단축키는 Windows에서 Win+Shift+S, macOS에서 Shift+Command+4입니다. 사용하는 환경에서 이미지 입력이 되는지는 공식 문서로 먼저 확인합니다. 표가 잘렸다면 "3쪽 표가 쪽 경계에서 잘렸어, 표 앞에서 쪽을 나눠 줘"처럼 위치를 적어 요청하면 됩니다.

마감은 사람이 합니다. 슬라이드는 장마다 글자가 상자 밖으로 나갔는지 보고, 빈 장과 중복된 장을 지웁니다. AI가 만든 슬라이드에는 이런 장이 남아 있을 수 있습니다. PDF로 내보낸 뒤에는 PDF를 직접 열어 쪽수, 글꼴 깨짐, 잘림을 확인합니다. "다 됐습니다"라는 말은 믿지 않습니다.

> **핵심** 만들었다는 보고가 아니라 내가 직접 열어 본 화면이 완성의 기준입니다. 마지막 손질은 사람이 합니다.

이 장에서 가져갈 마디를 정리하면 다음과 같습니다.

마디	뜻
하네스	규칙, 진행, 이유를 폴더에 파일로 둡니다
계획 먼저	본문보다 계획서를 먼저 받습니다
되묻게 하기	객관식으로 한 번에 하나씩 묻게 합니다
원본 하나	내용은 원본 md에서만 고칩니다
양식은 파일째	말로 설명하지 않고 양식 파일을 줍니다

마디	뜻
아트보드 먼저	뽑기 전에 쪽 모양으로 보고 번호로 고칩니다
마지막은 사람	직접 열어 확인하고 마무리합니다

실습

실습 1. 하네스 세 파일 만들기

빈 폴더를 하나 만들고, 아래 프롬프트로 세 파일을 만듭니다. 만든 뒤 세션을 닫고 새 세션에서 "이어서 하자"만 입력합니다. Claude가 현재 상태를 먼저 말하면 성공입니다.

프롬프트

> 이 폴더에서 제안서 한 건을 끝까지 만들 거야. 다른 세션에서 열어도 바로 이어서 할 수 있게 하네스 파일 세 개를 먼저 만들어 줘.
> - CLAUDE.md: 작업 규칙. 맨 위에 '작업을 시작하면 PROGRESS.md와 HISTORY.md를 먼저 읽고, 지금 상태와 다음 할 일을 한 줄로 요약한 뒤 시작한다.'를 쓰고, 그 아래에 산출물은 산출물 폴더에, 자료는 자료 폴더에 / 모르면 지어내지 말고 [확인 필요]로 / 파일명은 이름_날짜_v번호 를 적어 줘.
> - PROGRESS.md: 목표, 상태, 결정, 다음, 막힌 것
> - HISTORY.md: 결정한 것과 그 이유를 한 줄씩
> 만들고 나서 각 파일을 한 줄씩 설명해 줘.

실습 2. 되묻기와 계획서

가상의 주제를 하나 정합니다. 예를 들어 "중소 제조업체의 설비 점검 구독 서비스 제안"입니다. 먼저 되묻기를 시작합니다.

프롬프트

> 이제 제안서 기획을 할 거야. 주제는 [주제]야. 바로 계획을 쓰지 말고, 계획에 꼭 필요한 것을 나에게 먼저 물어봐.
> - 계획이 충분히 구체화될 때까지 계속 물어봐
> - 객관식으로 물어봐. 보기에 없으면 직접 쓸게
> - 한 번에 하나씩. 내가 '모르겠다'고 하면 네가 안을 제시해 줘
> - 질문이 끝나면 그때 계획을 써 줘

질문에 끝까지 답한 뒤, 계획서를 파일로 받습니다.

프롬프트

> 지금까지 답한 것으로 계획서를 구체적으로 써 줘. 산출물/제안서_계획서.md 로 저장해 줘.
> - 칸: 제안 개요, 읽는 사람과 결정 사항, 목차안, 조사할 것(질문 목록), 출처 기준, 일정
> - 조사할 것은 딥리서치에 그대로 넘길 수 있게 질문 형태로 써 줘
> - 모르는 칸은 [확인 필요]로 남겨 줘
> 그다음 같은 내용을 한눈에 보는 html로도 만들어 줘. 로컬에서 더블클릭하면 열리는 파일 하나로, 인터넷 연결 없이 열려야 해. 배경은 밝게, 글자는 진하게, 인쇄해도 잉크가 많이 들지 않게 해 줘. 산출물/제안서_계획서.html 로 저장해 줘.

조사할 것 목록을 네 가지 점검으로 읽고 서너 개로 줄입니다. 목록의 각 항목을 내 자료, 웹 검색, 딥리서치 중 어디로 보낼지도 표시해 봅니다.

실습 3. 양식 규칙과 아트보드

양식 파일과 예전 문서가 있다면 아래 프롬프트로 양식 규칙을 뽑습니다. 없으면 가상의 양식 설명을 직접 적어 줍니다.

프롬프트

> 자료/ 폴더의 양식 파일과 예전 문서를 열어 보고, 이 문서의 규칙을 산출물/양식규칙.md 로 정리해 줘.
> - 쪽, 여백, 글꼴, 제목 단계와 번호 체계
> - 표 모양, 강조 방식, 숫자와 단위 표기
> - 문체: 문장 끝 형태, 기호 단계, 출처 다는 법
> - 규칙마다 어느 파일 어디서 봤는지 적어 줘

짧은 원본 md 하나를 준비해 아트보드로 펼칩니다.

프롬프트

> 산출물/제안서.md 를 아트보드 html 하나로 펼쳐 줘. 산출물/아트보드.html 로 저장해 줘.
> - 양식규칙.md 대로 A4 세로 쪽 모양으로, 쪽마다 오른쪽 위에 판 번호를 붙여 줘
> - 로컬에서 더블클릭하면 열리는 파일 하나로, 밝은 배경과 진한 글자로 해 줘
> - '아트보드 다시 펼쳐 줘' 한마디면 되게 CLAUDE.md 에 방법을 적어 줘
> - 다시 펼 때는 내가 고친 쪽 나누기와 표 배치는 그대로 두고 바뀐 내용만 반영해 줘

판 번호와 화면 캡처로 두 곳을 고친 뒤 docx로 뽑고, 만든 docx와 PDF를 직접 열어 쪽수, 글꼴 깨짐, 표 잘림을 확인합니다.

선택 과제: 잘 통한 과정을 메타 프롬프트 파일로 저장합니다. 주제 자리는 [ ]로 비우고 되묻는 질문 목록도 안에 넣습니다. 저장한 뒤 다른 주제로 한 번 더 돌려 봅니다.

정리

- 규칙, 진행, 이유를 폴더의 파일로 남기면 세션이 바뀌어도 이어 갈 수 있고, 확인은 "이어서 하자" 한마디로 합니다.
- 본문보다 계획이 먼저입니다. Claude가 되묻게 하고, 계획서는 md와 html로 받습니다.
- 조사는 세 갈래로 나누고, 내 자료부터 읽습니다. 출처 없는 줄은 제안서에 쓰지 않습니다.
- 원본은 하나이고 문서는 목적대로 뽑습니다. 양식은 파일째 넘기고, 뽑기 전에 아트보드로 보고 번호로 고칩니다.
- 마지막은 사람이 직접 열어 확인합니다. 다음 17장에서는 같은 방법으로 끝난 일을 보고하는 결과보고서를 다룹니다.

17장

결과보고서: 기록과 맞춰 보는 문서

결과보고서는 조사해서 쓰는 문서가 아니라 기록과 맞춰 보는 문서입니다. 이 장은 끝난 일의 기록을 모으고, 계획과 나란히 놓고, 부풀리지 않고 쓰고, 숫자를 원 기록까지 따라가 확인하는 흐름을 다룹니다. 16장의 작업 폴더 규칙과 단일 원본 개념을 전제로 합니다. 예시는 모두 가상의 예이며, 숫자는 설명을 위해 지어낸 것입니다.

한 줄 위임이 위험한 이유

"결과보고서 써 줘"라고 한 줄로 맡기면 세 가지 일이 생깁니다.

- 그럴듯한 성과 문장이 먼저 나옵니다.
- 목표에 못 미친 항목이 뒤로 숨거나 순화됩니다.
- 빈칸의 원인을 AI가 그럴듯하게 짐작해서 채웁니다.

결과보고서에서 AI는 말을 부풀리고, 모르는 원인을 짐작으로 메웁니다. 일지에도 없는 이유가 "참가자 일정 문제"처럼 자연스러운 문장으로 들어가면 읽는 사람은 사실로 받아들입니다. 그래서 세 원칙을 먼저 세웁니다.

> **핵심** 재료는 기록입니다. 기준은 계획입니다. 원인은 사람이 답합니다.

기준이 계획이라는 말은 "무엇을 했나"가 아니라 "계획한 것 중 무엇이 됐나"를 쓴다는 뜻입니다. 한 일을 나열하면 성과처럼 읽히지만, 계획 옆에 놓으면 달성과 미달이 드러납니다.

새 폴더 세팅

끝난 사업마다 새 폴더를 열고, 16장에서 만든 규칙 파일(CLAUDE.md)을 복사해 결과보고서용으로 고칩니다. 바꿀 것은 세 가지입니다.

- 확인 표시를 `[담당자 확인 필요]` 하나로 통일합니다.
- 숫자는 원 기록에서 가져오고 출처 번호를 답니다.
- 기록에 없는 원인, 조치, 성과는 쓰지 않고 `[담당자 확인 필요]`로 남깁니다.

진행 기록(PROGRESS.md)과 결정 이유 기록(HISTORY.md)은 새로 만듭니다.

프롬프트

> 이 폴더에서 끝난 사업의 결과보고서를 만들 거야. 지난 폴더 [ ]의 CLAUDE.md를 복사해서 결과보고서용으로 고쳐줘.
> - 첫 줄, 파일 두는 곳, 파일명 규칙은 그대로
> - 제안서 전용 규칙은 지워줘
> - 확인 표시는 [담당자 확인 필요] 하나로 통일해줘
> - 추가 규칙: 숫자는 원 기록에서 가져오고 출처를 단다 / 기록에 없는 원인, 조치, 성과는 쓰지 말고 [담당자 확인 필요]로 남긴다
> - PROGRESS.md와 HISTORY.md는 새로 만들어줘

기록 모으기

재료 다섯 종

보고서의 재료는 다섯 종류의 기록입니다.

종류	예	쓰임
처음 계획	계획서, 품의서, 목표가 적힌 메일	보고의 기준
실적 숫자	집계 파일, 내려받은 데이터, 정산 내역	계획과 비교할 값
경과 기록	운영 일지, 회의록, 일정이 바뀐 메일	추진 경과와 원인의 단서
반응	설문, 의견, 문의	결과의 근거
증빙	결과물, 명단의 인원수	한 일의 확인

처음 계획이 없으면 보고서가 서지 않습니다. 계획서가 없던 일이라면 시작할 때 무엇을 바랐는지 세 줄로 적어 함께 넣습니다.

넣기 전에 지울 것

Claude가 읽은 파일의 내용은 서버로 전송되고, 한 번 보낸 내용은 회수할 수 없습니다. 기록 폴더에 넣기 전에 네 가지를 지운 사본을 만듭니다. 8장의 파일 첨부 주의와 14장의 개인정보 지우기와 같은 원칙입니다.

지울 것	방법
고객 정보	이름, 연락처 대신 집계한 숫자만
직원 실명	'담당자 A'처럼 역할로 대체
미공개 실적, 내부 금액	조직 규정이 허용하는지 먼저 확인
사진 속 얼굴, 화면	가린 사본만

무엇을 넣어도 되는지는 업종과 조직마다 다릅니다. 법적으로 단정하지 않고, 조직의 규정이 우선합니다.

기록목록부터

보고서를 쓰기 전에 무엇이 있고 무엇이 빠졌는지 표로 봅니다. 열은 번호, 파일명, 종류, 다루는 기간, 주요 숫자입니다. 번호는 파일명 앞 숫자와 맞춰 `R-01` 형식으로 고정하고, 한번 붙인 번호는 바꾸지 않습니다. 이후 모든 출처 표시가 이 번호를 가리키기 때문입니다.

표 아래에는 '빠진 것'을 적습니다. 기록이 비는 기간, 계획에는 있는데 실적이 없는 항목이 들어갑니다. 이 칸이 첫 번째 할 일 목록입니다.

프롬프트

> 자료 폴더의 파일을 전부 훑어보고 기록목록.md를 만들어줘. 보고서는 아직 쓰지 마.
> - 열: 번호 / 파일명 / 종류(계획, 실적, 경과, 반응, 증빙, 양식) / 다루는 기간 / 주요 숫자
> - 번호는 파일명 앞 숫자와 맞춰줘(01_ → R-01). 한번 붙인 번호는 바꾸지 마
> - 표 아래에 '빠진 것': 기록이 비는 기간, 계획에는 있는데 실적이 없는 항목
> - 개인정보로 보이는 것이 있으면 옮겨 적지 말고 그 파일 이름만 알려줘

추진 경과표

메일, 회의록, 일지에 흩어진 일을 날짜 한 줄에 하나씩 세웁니다. 열은 날짜, 한 일, 정한 것, 근거(R-번호)입니다. 계획과 달라진 일정은 '계획 → 실제'로 나란히 적습니다. 기록에 날짜가 없으면 짐작하지 않고 `[담당자 확인 필요]`로 남깁니다.

이 표는 보고서의 추진 경과 절이 되고, 나중에 '이미 한 조치'를 쓸 때 근거가 됩니다.

계획 대비 실적: 보고서의 심장

계획 한 칸과 실적 한 칸을 나란히 놓으면 보고서의 절반이 끝납니다. 실적표의 열은 아홉 가지입니다.

- 지표, 단위, 기준(이상 또는 이내)
- 계획, 실적, 차이, 판정
- 실적의 출처, 차이의 원인

판정은 달성과 미달을 같은 무게로 씁니다. 미달에는 완곡한 말을 붙이지 않습니다.

실적표는 수식으로

비율과 차이를 AI가 계산해 값만 적게 하지 않고, 엑셀 수식으로 남깁니다. 원자료는 다른 시트에 옮기고, 실적, 차이, 판정은 그 시트를 가리키는 수식으로 만듭니다. 값만 적힌 표는 어디서 나온 숫자인지 따라갈 수 없지만, 수식은 14장의 '합계 맞춰 보기'처럼 누구나 다시 확인할 수 있습니다. 비율은 계획 문서에 적힌 분모로 계산합니다.

규칙은 세 가지입니다.

- 실적을 기록에서 못 찾으면 '기록 없음'이라고 적습니다.
- 값이 비면 직접 계산하지 말고 멈춰서 알리게 합니다.
- 차이의 원인 칸은 `[담당자 확인 필요]`로 비워 둡니다.

엑셀 수식을 유지하는 방식이나 양식 파일을 채우는 범위는 사용하는 환경과 도구에 따라 다르므로, 처음에는 작은 파일로 결과를 확인합니다.

예산은 항목별로

총액이 계획 이내여도 항목마다 남은 돈에는 이유가 있어야 합니다. 열은 항목, 계획, 집행, 잔액, 집행률, 잔액의 사유, 출처입니다. 잔액, 집행률, 합계는 수식으로 만들고, 이미 있는 시트의 수식 칸은 건드리지 않게 합니다. 잔액의 사유는 회의록, 메일, 일지에 있는 것만 쓰고, 없으면 `[담당자 확인 필요]`로 남깁니다.

프롬프트

> 처음 계획(R-01)의 성과지표로 계획 대비 실적표를 만들어 산출물 폴더에 계획대비실적.xlsx로 저장해줘.
> - 지표 이름과 측정 방법은 R-01 그대로. 비율은 R-01에 적힌 분모로
> - 열: 지표 / 단위 / 기준(이상, 이내) / 계획 / 실적 / 차이 / 판정 / 실적의 출처 / 차이의 원인
> - 숫자 칸에는 숫자만. 원자료는 다른 시트에 옮기고 실적, 차이, 판정은 그 시트를 가리키는 수식으로
> - 실적을 기록에서 못 찾으면 '기록 없음'
> - 차이의 원인 칸은 [담당자 확인 필요]로 채워 둬. 네가 짐작해서 채우지 마
> - 예산 시트도 추가해줘: 잔액, 집행률, 합계는 수식으로, 잔액의 사유는 기록에 있는 것만

원인은 묻게 하고 내가 답한다

왜 목표에 못 미쳤는지는 AI가 알 수 없습니다. `[담당자 확인 필요]` 칸을 차이가 큰 것부터 하나씩 나에게 묻게 합니다. 일지, 회의록, 메일, 설문의 서술 응답에 단서가 있으면 먼저 보여 주고 맞는지 묻게 합니다. 질문은 객관식으로, 한 번에 하나씩 하고, 보기에 없으면 직접 씁니다.

답이 들어온 칸에는 출처 표기를 네 가지로 구분해 붙입니다.

표기	뜻
`[R-번호]`	기록에 적혀 있음
`[R-번호·담당자 추정]`	기록을 쓴 사람의 짐작
`[담당자 답변]`	지금 내가 답한 것
`[원인 확인 중]`	내가 모른다고 한 경우

AI가 보기를 제안할 수는 있지만 안을 지어내 칸에 넣지는 않습니다. 이 표기 덕분에 읽는 사람은 어느 원인이 기록이고 어느 것이 짐작인지 구분할 수 있습니다.

프롬프트

> 계획대비실적.xlsx의 [담당자 확인 필요] 칸을 차이가 큰 것부터 하나씩 나에게 물어봐.
> - 일지, 회의록, 메일, 설문 서술 응답에 단서가 있으면 먼저 보여주고 맞는지 물어봐
> - 객관식으로, 한 번에 하나씩. 보기에 없으면 직접 쓸게
> - 기록에 있으면 [R-번호], 기록을 쓴 사람의 짐작이면 [R-번호·담당자 추정], 내 답이면 [담당자 답변]
> - 내가 모른다고 하면 [원인 확인 중]. 네가 안을 지어내지 마
> - 답할 때마다 바로 그 칸에 적고, 수식 칸은 건드리지 마

부풀리지 않고 쓰기

성과 세 층

성과는 세 층으로 나눕니다.

- 산출: 한 것(워크숍 몇 회 운영, 수료 몇 명)
- 결과: 달라진 것(사후 설문 응답자 중 몇 퍼센트가 업무에 적용)
- 효과: 조직에 남은 것(측정하지 않았다면 쓰지 않습니다)

AI는 '한 것'을 '남은 것'처럼 씁니다. 층을 나누고 위층일수록 더 단단한 근거를 요구하면 이 흐름이 막힙니다.

부풀린 문장과 기록대로 쓴 문장

형용사 자리에 숫자를 넣고, '덕분에'는 근거가 있을 때만 씁니다. 다음 표의 숫자는 모두 가상입니다.

부풀린 문장	기록대로 쓴 문장	바꾼 이유
만족도가 매우 높았음	4점 만점에 3.1점, 목표 3.5점 미달(응답 40명)	형용사를 숫자로
워크숍 덕분에 업무 효율이 크게 향상됨	응답자 40명 중 22명이 업무에 적용, 효율은 미측정	인과는 근거가 있을 때만
목표를 대부분 달성함	지표 4개 중 1개 달성, 3개 미달	개수로
일부 일정 조정	2회차를 3주 뒤로 연기(강사 일정 변경, R-05)	무엇이 언제에서 언제로, 왜

부풀린 문장은 틀린 문장이 아니라 확인할 수 없는 문장입니다. 읽는 사람이 근거를 따라갈 수 없다는 점이 문제입니다.

성과 절은 숫자부터

성과 절은 지표마다 계획, 실적, 차이, 판정 순서로 씁니다. 미달 지표도 달성 지표와 같은 무게로 씁니다. 숫자마다 출처 번호를 붙이고, 엑셀 값을 그대로 옮기며 다시 계산하거나 반올림하지 않습니다. 비율에는 분모를 밝힙니다. 엑셀 값이 비어 있으면 직접 계산하지 말고 멈춰서 알리게 합니다. 쓰지 않을 표현은 미리 정합니다: 매우, 크게, 대부분, 기여, 덕분에, 일부 조정.

절 끝에는 '측정의 한계'를 적습니다. 설문 응답률, 측정 시점, 빠진 구간이 들어갑니다. 한계가 적힌 보고서가 숫자를 더 믿게 합니다.

프롬프트

> 계획대비실적.xlsx와 추진경과표.md로 결과보고서의 '추진 성과' 절만 써서 산출물 폴더의 보고서_본문.md에 붙여줘.
> - 지표마다 계획 → 실적 → 차이 → 판정 순서로. 미달 지표도 같은 무게로. 원인은 문제점 절에서 쓸 거야
> - 숫자마다 [R-02]처럼 출처를 붙이고, xlsx의 값을 똑같이. 비율은 분모를 밝혀
> - xlsx 값이 비어 있으면 직접 계산하지 말고 멈춰서 알려줘
> - 매우, 크게, 대부분, 기여, 덕분에, 일부 조정은 쓰지 마. 형용사 대신 숫자로
> - 성과는 산출, 결과, 효과 세 층으로 나누고, 측정 안 한 효과는 쓰지 마
> - 절 끝에 '측정의 한계'(응답률, 측정 시점, 빠진 구간)를 적어줘
> - 다 쓰면 PROGRESS.md에 '추진 성과 절 완료'라고 적어줘

문제점은 네 칸

잘 안 된 것은 네 칸으로 씁니다: 사실, 원인, 이미 한 조치, 다음에 바꿀 것.

- 원인은 실적표의 원인 칸에 적힌 것만 씁니다. 확인 중이면 다음에 바꿀 것도 `[담당자 확인 필요]`로 둡니다.
- 이미 한 조치는 경과표와 기록에 있는 것만, 출처와 함께 씁니다.
- 다음에 바꿀 것은 원인 하나에 한 줄이며, 원인과 상관없는 일반론은 쓰지 않습니다.
- 순화하지 않고 숫자 그대로 씁니다. "다소 미흡"이 아니라 "목표보다 0.4점 낮음"입니다.

> **핵심** 문제를 숨기지 않은 보고서가 더 믿을 만합니다.

프롬프트

> 미달한 지표마다 '문제점과 개선' 절을 써서 보고서_본문.md에 붙여줘.
> - 네 칸 순서로: 사실 / 원인 / 이미 한 조치 / 다음에 바꿀 것
> - 원인은 계획대비실적.xlsx의 원인 칸에 적힌 것만. '확인 중'이면 다음에 바꿀 것도 [담당자 확인 필요]
> - 이미 한 조치는 추진경과표의 '정한 것'과 메일, 회의록, 일지에 있는 것만 R-번호와 함께
> - '다음에 바꿀 것'은 원인 하나에 한 줄. 원인과 상관없는 일반론은 쓰지 마
> - 순화하지 마. '일부', '다소'로 줄여 쓰지 말고 숫자 그대로

첫 장 요약과 표준 목차

결재권자는 첫 장만 읽습니다. 요약은 본문을 줄인 것이 아니라 판단에 필요한 것만 뽑은 것이며, 다섯 줄로 씁니다.

1. 결론
2. 핵심 숫자
3. 잘 안 된 것
4. 바꿀 것
5. 요청

요약은 맨 마지막에 씁니다. 본문보다 먼저 쓴 요약은 본문과 어긋납니다.

표준 목차는 요약, 사업 개요, 추진 경과, 추진 성과, 문제점과 개선, 향후 계획과 건의, 붙임(실적표, 예산 시트, 출처 기록 목록)입니다. 개요는 처음 계획에서, 경과는 경과표에서 가져옵니다. 건의는 후보를 사람이 고릅니다. 설문의 서술 응답과 미적용 사유를 건수로 모아(빈칸도 따로 세어) 보여 주게 하고, 어느 것을 건의로 올릴지 정하는 것은 사람입니다. 본문을 완성할 때 이미 쓴 절의 내용은 바꾸지 않게 지시합니다.

양식에 붓기

조직의 보고서 양식(docx)이 있으면 확정된 본문 md를 양식에 다시 붓습니다. 원본은 md이고 docx는 사본이라는 16장의 원칙 그대로입니다.

- 양식의 글꼴, 제목 스타일, 바닥글은 그대로 둡니다. 맨 위 정보표는 채우되 작성자는 `[담당자 확인 필요]`로 둡니다.
- '(작성 안내)' 줄은 지우고 내용으로 바꿉니다. 안내 줄의 회색과 기울임은 물려받지 않게 합니다.
- 실적표는 엑셀 값 그대로 옮깁니다. 값이 비어 있으면 멈추고 알리게 합니다.
- 본문의 출처 번호는 붙임의 '출처 기록 목록'으로 옮깁니다.
- 다 채우면 PDF로 뽑아 쪽수와 넘친 곳을 확인합니다.

마지막 점검은 두 가지입니다. 안내 문구나 회색 기울임이 남았는지, 그리고 실적표 숫자가 엑셀과 같은지(반올림이 한 번 더 들어가지 않았는지) 봅니다.

거꾸로 따라가기와 새 세션 검토

숫자 셋을 내 손으로

Claude에게 시키지 않고 직접 합니다.

1. 본문에서 인원, 금액, 비율을 하나씩 고릅니다.
2. 출처 번호를 기록목록에서 찾아 파일 이름과 열까지 따라갑니다.

3. 원 파일을 열어 그 열의 합계, 평균, 건수를 직접 내 같은 값이 나오는지 봅니다.
4. 하나라도 다르면 멈추고, 나머지 숫자도 다시 봅니다.

세 개가 맞으면 나머지를 믿을 근거가 생깁니다. 19장의 자가검증은 같은 AI가 스스로 검증 질문을 만드는 방식입니다. 여기서는 사람이 원 기록까지 직접 따라가고, AI는 별도 세션에서 독자 역할만 맡습니다. 쓴 쪽이 검증하면 놓치는 것을 두 가지 방식이 서로 보완합니다.

새 세션에서 결재권자의 눈으로

쓰던 세션은 사정을 다 알고 있어서 처음 읽는 사람이 걸리는 곳을 놓칩니다. PROGRESS.md를 갱신하고 세션을 닫은 뒤, 같은 폴더에서 새 세션을 열어 검토만 시킵니다. 문서는 고치지 못하게 하고 결과를 검토의견.md로 받습니다.

프롬프트

> 산출물 폴더의 결과보고서 docx를 처음 받은 결재권자라고 생각하고 검토해줘. 문서는 고치지 마.
> - 첫 장만 읽고 무엇을 결정할 수 있는지, 못 한다면 무엇이 빠졌는지
> - 예상 질문 다섯 개
> - 본문 숫자가 계획대비실적.xlsx, 추진경과표.md와 다른 곳
> - 부풀린 표현: 매우, 크게, 대부분, 기여, 덕분에, 일부 조정
> - [담당자 확인 필요]가 아직 남아 있는 곳
> - 고객 정보, 실명, 미공개 수치가 남아 있는 곳
> - 결과는 산출물 폴더에 검토의견.md로 저장해줘

반영도 새 세션에서 합니다. 검토의견과 내가 단 코멘트를 보고서 본문 md에 먼저 반영하고, 직전 문서와 같은 서식으로 양식에 다시 부어 새 버전(v번호 +1)으로 저장합니다. docx만 따로 고치지 않습니다. 무엇을 왜 고쳤는지는 HISTORY.md에 한 줄씩 남깁니다.

코멘트는 짚는 말을 정해 두면 빠릅니다. 결과보고서판 네 마디는 다음과 같습니다.

말	쓰는 때
기록으로 확인	인원, 금액이 기록과 다른 것 같을 때
정합성	앞뒤 숫자가 어긋날 때
숫자로	형용사가 남아 있을 때
순화하지 마	미달이 '일부', '다소'로 흐려졌을 때

다음 보고는 기록만 갈아 넣기

한 번 잘 돌린 과정은 메타 프롬프트 파일로 저장합니다. 순서는 기록목록, 경과표, 계획 대비 실적, 원인 묻기, 절 단위 작성, 요약, 양식, 검토입니다. 규칙은 전부 넣습니다: 지운 사본인지 먼저 묻기, 출처 번호, 원인은 묻기, 쓰지 않을 표현, 미달도 같은 무게. 사업 이름과 보고 기간 자리는 `[ ]`로 비워 둡니다. 16장의 제안서 기획 메타 프롬프트 옆에 나란히 둡니다. 하나는 앞으로 할 일을 설득하는 틀이고, 다른 하나는 끝난 일을 보고하는 틀입니다.

프롬프트

> 이 폴더의 CLAUDE.md, PROGRESS.md, HISTORY.md와 산출물 폴더를 읽고, 여기서 한 결과보고서 과정을 다음 보고서에도 쓸 수 있는 프롬프트로 만들어줘.
> - 순서: 기록목록 → 추진경과표 → 계획대비실적 → 원인 묻기 → 절 단위 작성 → 요약 → 양식 → 검토
> - 규칙을 다 넣어줘: 지운 사본인지 먼저 묻기, 출처 번호, 원인은 묻기, 쓰지 않을 표현, 미달도 같은 무게
> - 현재 모델에 맞는 프롬프트 작성 방식을 적용하되 자율성을 과도하게 제한하지 마
> - 사업 이름과 보고 기간만 바꿔 넣게 [ ] 자리를 비워 둬
> - 산출물/메타프롬프트_결과보고서.md로 저장해줘

다음 보고의 순환은 네 걸음입니다. 지운 사본을 넣고, "이어서 하자"로 시작하고, 숫자 셋을 따라가고, 결재를 올립니다. 틀과 규칙은 폴더에 남아 있어 바뀌는 것은 자료 폴더뿐입니다. 두 번째 보고서부터는 쓰는 일이 아니라 확인하는 일이 됩니다.

실습

실습 1. 기록목록 만들기

끝난 일 하나를 고릅니다(없으면 가상의 사업을 정합니다). 다섯 종 기록이 어디에 있는지 찾고, 넣기 전에 지울 것 네 가지로 사본을 만듭니다. 그런 다음 앞의 기록목록 프롬프트로 기록목록.md와 '빠진 것'을 만듭니다.

실습 2. 계획 대비 실적표와 원인 묻기

가상의 사업을 정해 지표 3~5개와 예산 3~4개 항목을 만듭니다. 계획 대비 실적 프롬프트로 엑셀을 만들고, 비율 칸이 원자료 시트를 가리키는 수식인지 직접 클릭해 확인합니다. 이어서 원인 묻기 프롬프트를 실행해 `[담당자 확인 필요]` 칸을 하나씩 채웁니다.

실습 3. 성과 절과 검토

성과 절 프롬프트로 한 절을 쓰게 한 뒤, 금지 표현이 남았는지 문서에서 직접 검색합니다. 본문 숫자 셋을 원 기록까지 따라가 합계를 직접 내 비교합니다(Claude는 쓰지 않습니다). 마지막으로 PROGRESS.md를 갱신하고 세션을 닫은 뒤, 새 세션에서 검토 프롬프트를 실행해 검토의견.md를 받습니다.

정리

- 재료는 기록입니다. 결과보고서는 조사가 아니라 기록과 맞춰 보는 문서입니다.
- 넣기 전에 고객 정보, 실명, 미공개 수치, 사진 속 얼굴을 지웁니다. 규정이 우선입니다.
- 기준은 계획입니다. 계획 대비 실적을 엑셀 수식으로 남기고 미달도 같은 무게로 씁니다.
- 원인은 사람이 답합니다. 기록, 추정, 내 답, 확인 중을 표기로 구분하고, 형용사 대신 숫자를 씁니다.
- 숫자 셋을 원 기록까지 직접 따라가고, 새 세션에서 결재권자의 눈으로 검토시킵니다.

18장

간단한 웹앱으로 업무 자동화

반복 업무 중에는 엑셀과 메신저로는 한계가 오는 일이 있습니다. 요청을 받고, 담당자를 정하고, 상태를 추적하는 일이 대표적입니다. 이 장에서는 한 장짜리 요구사항 문서(PRD)에서 시작해 데모를 만들고, 구글 시트에 데이터를 저장하고, 웹에 올려 운영하는 과정을 다룹니다.

언제 웹앱이 필요한가

웹앱을 만들기 전에 먼저 시트나 폼으로 충분하지 않은지 따져 봅니다. 다음 중 둘 이상이 해당할 때 웹앱을 고려합니다.

- 여러 사람이 같은 데이터를 동시에 입력하고, 각자 보는 화면이 달라야 한다.
- 입력할 때마다 상태가 바뀌고(접수, 처리 중, 완료), 그 상태를 한눈에 봐야 한다.
- 입력 규칙이 있어서 잘못된 값을 막아야 한다.
- 시트를 직접 열어 고치면 실수가 자주 난다.

입문 과정 3회차에서 반복 업무를 고르고 AI에게 맡기는 법을, 4회차에서 결과물을 웹에 올리고 링크로 공유하는 법을 익혔습니다. 이 장은 두 회차를 이어서, 저장과 운영까지 갖춘 작은 앱으로 넓히는 단계입니다. 회차별 실습 프롬프트는 액티전스 자료실에서 볼 수 있습니다.

가상의 예: 사내 요청 접수 앱

가상의 소규모 회사에서 총무 담당자 한 명이 사무용품 구매, 회의실 예약, 장비 수리 요청을 메신저와 구두로 받고 있다고 가정합니다. 요청이 누락되고, 누가 처리 중인지 모르고, 월말에 정리가 어렵습니다. 이 문제를 해결하는 요청 접수 앱의 요구사항은 아래와 같습니다.

- 직원은 요청 유형, 제목, 내용, 희망 날짜를 입력해 요청을 등록한다.
- 총무 담당자는 요청 목록을 보고 상태(접수, 처리 중, 완료, 보류)를 바꾸고 메모를 남긴다.
- 직원은 자기가 낸 요청의 상태를 볼 수 있다.
- 월말에 유형별 건수를 볼 수 있다.

기능이 많지 않은 이 정도 규모가 첫 웹앱으로 알맞습니다.

1단계. 한 장짜리 PRD 쓰기

PRD(Product Requirements Document)는 무엇을 만들지 적은 문서입니다. AI에게 앱을 만들게 할 때 PRD는 "빈칸을 AI가 추측하지 않게 하는 문서"입니다. 길게 쓸 필요는 없고, 다음 항목이 한 장에 들어가면 됩니다.

항목	내용
문제	지금 무엇이 불편한가, 누가 불편한가
사용자	어떤 역할이 있고 각자 무엇을 하는가
성공 기준	만든 뒤 무엇이 달라지면 성공인가
하지 않을 것	이번에 만들지 않을 기능
제약	지켜야 할 규칙(개인정보, 비용, 사용 기기)
화면과 기능	화면별로 무엇을 보여 주고 무엇을 누르는가
데이터	저장할 항목과 상태 값

특히 "하지 않을 것"이 중요합니다. 로그인, 결제, 알림 문자 같은 기능은 처음에는 빼겠다고 적어 두면 AI가 범위를 넓히지 않습니다. 기능마다 "누르면 이렇게 된다"는 동작과, 완성됐다고 판단할 수 있는 확인 기준을 한 줄씩 붙입니다. 이 방식은 입문 과정 2회차에서 연습한 "버튼을 누르면 무엇이 일어나는지까지 말한다"와 같은 원리입니다.

PRD 작성을 도와주는 PRD 스킬이 있습니다(부록 B). 아이템 한 줄을 주면 필요한 항목을 추론해 채우고, 꼭 필요한 것만 객관식으로 물어 PRD 문서를 만들어 줍니다. 스킬을 쓰지 않더라도 실습의 프롬프트로 같은 효과를 낼 수 있습니다.

2단계. 데모부터 만들기

처음부터 실제 데이터와 로그인까지 붙이지 않고, 가짜 데이터로 돌아가는 데모를 먼저 만듭니다. 데모에서 확인할 것은 화면 흐름과 사용성입니다. 요청 등록, 목록 보기, 상태 변경, 월말 집계가 한 바퀴 돌아가는지 눌러 보며 확인합니다.

데모 단계가 있으면 좋은 이유는 세 가지입니다.

- 실제 개인정보를 쓰지 않고 시험할 수 있습니다.
- 동료에게 보여 주고 의견을 받기 쉽습니다. 말로 설명하는 것보다 눌러 보게 하는 것이 훨씬 빠릅니다.
- 로그인과 저장은 나중에 붙이면 되므로 첫 결과가 빨리 나옵니다.

원샷 프로토타이핑

액티전스 라이브러리에는 PRD 한 장으로 빈 폴더에서 배포까지 이어 가는 카드와 스킬이 있습니다. 라이브러리의 "원클릭 풀스택" 카드와 같은 내용을 스킬로 만든 것이 원샷 프로토타이핑 스킬입니다(부록 A, 부록 B). 카드의 흐름을 요약하면 다음과 같습니다.

1. 로그인 없이 모든 기능을 눌러 볼 수 있는 데모 빌드를 만든다.
2. 기능 초안이 통과하면 화면 디자인을 다듬는다.
3. 보안을 점검하고 강화한다.
4. 웹에 배포하고 실제 주소에서 동작을 확인한다.

저장소는 기본적으로 구글 시트이고, 이를 위한 구글 클라우드 설정(서비스 계정, 키, 시트 공유)까지 이 흐름에 포함되어 있습니다. 이 스킬은 "프로토타이핑"이라는 단어와 PRD가 모두 있을 때만 작동하도록 만들어져 있습니다. 카드는 사람의 확인 없이 끝까지 진행하는 성격이 강하므로, 처음에는 짧은 PRD로 시험하고 결과를 직접 확인하는 것이 좋습니다. 카드와 스킬의 전문은 프롬프트 라이브러리와 스킬 페이지에 있습니다.

풀스택 기능 구현

이미 만든 앱에 기능을 하나 더 붙일 때는 풀스택 기능 구현 스킬이 맞습니다. 데이터 저장, 서버 처리, 화면, 테스트를 기능 하나 단위로 함께 만듭니다. 예를 들어 요청 접수 앱에 "첨부 사진 올리기"를 추가하는 식입니다. 이 스킬은 "풀스택"과 "구현"이라는 단어가 모두 들어 있을 때 작동합니다. 새 앱을 처음 만들 때는 원샷 프로토타이핑을, 있는 앱에 기능을 더할 때는 풀스택 기능 구현을 씁니다.

여러 세션과 서브에이전트로 일을 나눠 맡기는 방법은 10장에서, 스킬을 설치하는 방법은 11장에서 다룹니다.

3단계. 구글 시트에 데이터 저장하기

데모가 마음에 들면 실제 저장소를 붙입니다. 첫 앱의 저장소로는 구글 시트가 적당합니다. 이미 아는 도구이고, 데이터를 사람이 직접 열어 볼 수 있으며, 별도의 데이터베이스를 배우지 않아도 됩니다. 동시에 많은 사용자가 쓰거나 데이터가 아주 많아지면 한계가 있으므로, 그때는 전문 데이터베이스로 옮기는 것을 AI와 함께 검토합니다.

데이터 3단계 사다리

데이터를 어디까지 앱에 연결할지는 세 단계로 나눠 판단합니다.

단계	방식	할 수 있는 일
1단	데이터가 화면 안에만 잠깐 보입니다.	새로고침하면 사라집니다. 데모에 알맞습니다.
2단	시트를 읽기 전용으로 게시하고, 고칠 때는 시트에서 직접 고칩니다.	화면은 시트를 비춥니다. 화면에서 쓸 수는 없습니다.
3단	시트와 클라우드 서비스 계정을 정식으로 연결합니다.	화면에서 직접 입력하고 수정합니다.

대부분의 업무 도구는 2단으로 충분합니다. 시트를 공개할 수 없거나 화면에서 데이터를 써야 할 때만 3단으로 올립니다. 어느 단계든 시트 한 장이 원본(단일 원본, SSOT, 7장)이고 화면은 그것을 비출 뿐입니다. 2단에는 한계가 두 가지 있습니다. 읽기만 되고, 게시 주소를 아는 사람은 누구나 볼 수 있습니다. 민감한 정보가 든 시트는 2단으로 게시하지 않습니다.

3단 연결의 원리와 설정

3단에서는 앱이 구글 시트에 직접 로그인하지 않습니다. 앱 전용의 "서비스 계정"이라는 로봇 계정을 만들고 그 계정에 시트를 공유합니다. 앱은 서비스 계정의 열쇠(키)로 시트를 읽고 씁니다. 설정은 목적 기준으로 네 가지입니다. 메뉴 이름은 자주 바뀌므로 공식 문서와 Claude의 안내를 대조합니다.

1. 클라우드 프로젝트를 만듭니다.
2. 구글 시트를 프로그램에서 쓸 수 있도록 해당 API(프로그램용 창구)를 켭니다.
3. 서비스 계정을 만들고 키를 받습니다.
4. 그 서비스 계정의 이메일 주소에 시트를 공유합니다.

막히면 지금 보는 화면을 설명하고 Claude에게 다음에 할 일을 묻습니다. 계정 생성과 키 발급은 사람이 직접 합니다.

시트 설계

- 시트 한 장에 한 종류의 데이터만 둡니다. 요청 접수 앱이라면 "요청" 시트 하나로 시작합니다.
- 첫 줄에 열 이름을 둡니다: 접수번호, 접수시각, 요청자, 유형, 제목, 내용, 희망일, 상태, 담당자 메모, 처리일.
- 상태는 정해진 값만 쓰도록 드롭다운으로 제한합니다.
- 행을 지우지 말고 상태로 관리합니다. 기록이 남아야 월말 집계와 이력 확인이 됩니다.
- 앱이 쓰는 시트와 사람이 보는 보고용 시트를 분리합니다. 사람이 앱 시트를 직접 고치다 열 순서를 바꾸면 앱이 멈춥니다.

비밀 값 다루기

서비스 계정 키, 시트 주소, 발송 서비스의 API 키는 모두 비밀 값입니다. 다음 규칙을 지킵니다.

- **채팅창에 붙여 넣지 않습니다.** 대화 기록은 다른 곳에 남을 수 있습니다. 키가 필요하다면 "키 값은 환경 변수 파일에 직접 넣을 테니, 어떤 이름으로 읽도록 코드를 짜 달라"고 요청합니다.
- **코드 파일에 쓰지 않습니다.** 값은 `.env` 같은 환경 변수 파일에 두고, 코드는 이름으로만 읽습니다. 특히 화면(브라우저)에서 실행되는 코드에 키를 넣으면 브라우저의 개발자 도구로 누구나 꺼내 볼 수 있으므로, 키는 서버 쪽 코드에서만 씁니다.
- **저장소에 올리지 않습니다.** `.env` 파일은 Git이 무시하도록 `.gitignore`에 적습니다. 이미 올렸다면 키를 폐기하고 새로 발급받습니다. 파일에서 지우는 것만으로는 이력에 남기 때문입니다.
- **배포 서비스의 환경 변수 설정에 넣습니다.** 배포된 앱이 읽는 값은 배포 서비스의 설정 화면에 입력합니다.
- **권한을 최소로 줍니다.** 서비스 계정에는 해당 시트 하나만 공유하고, 필요한 권한만 줍니다.
- **키가 노출됐다고 의심되면 즉시 폐기합니다.**

비밀번호·키를 입력하는 단계도 사람이 직접 합니다. Claude에게 시킬 일은 순서를 안내받고, 코드가 환경 변수를 올바르게 읽는지 확인하는 것입니다.

4단계. 웹에 배포하기

파일에서 주소로

앱을 만들고 나면 파일 상태로 둘지, 주소를 붙일지가 갈립니다.

파일로 공유	주소로 공유
메일이나 메신저로 보내야 합니다.	링크만 보내면 됩니다.
사람마다 다른 버전을 갖게 됩니다.	고치면 모두가 동시에 최신을 봅니다.
컴퓨터가 고장 나면 사라집니다.	변경 이력이 남습니다.

파일로 끝난 도구는 만든 사람만 쓰고 조직에 남지 않습니다. 주소가 붙는 순간 내 도구가 우리 도구가 됩니다. 보통 코드는 저장소 서비스에 올려 원본과 이력을 보관하고, 호스팅 서비스가 주소를 붙이며 저장소가 바뀌면 자동으로 다시 배포합니다. 한 번 연결해 두면 "고쳐 줘" 한마디로 주소의 내용이 최신이 됩니다. 이 흐름은 20장에서 이어서 다룹니다.

주소에는 나만 볼 수 있는 것(localhost), 임시 주소, 정식 도메인의 구분이 있습니다. 6장의 "주소의 네 가지"에서 정리했습니다.

배포는 만든 앱을 인터넷의 서버에 올려 주소로 열 수 있게 하는 일입니다. 입문 과정 4회차에서 정적 파일을 올려 본 경험이 있다면, 이제는 서버 처리가 있는 앱을 올립니다. Vercel 같은 호스팅 서비스가 흔히 쓰이며, 구체적인 올리는 방법은 서비스 공식 문서(Vercel 문서)의 안내를 따릅니다. 서비스의 메뉴와 명령은 바뀌므로 AI가 알려 주는 메뉴 이름을 그대로 믿지 말고 공식 문서와 대조합니다.

배포 전 점검 목록입니다.

- 비밀 값이 코드나 저장소에 없는지 확인했다.
- 환경 변수를 배포 서비스에 입력했다.
- 데모 데이터가 아니라 실제 시트에 연결되는지 확인했다.
- 누가 이 주소로 접근할 수 있는지 정했다. 사내용이라면 주소만 알면 누구나 열 수 있는 상태가 되지 않도록 접근 제한 방법을 정했다.
- 이전 버전으로 되돌리는 방법을 알고 있다.

배포 후에는 로그인하지 않은 시크릿 창에서도 열어 보고, 내 기기가 아닌 다른 기기에서 열어 보고, 요청 하나를 실제로 등록한 뒤 시트에 들어갔는지 확인합니다. 이 확인 과정은 20장에서 더 자세히 다룹니다.

5단계. 운영하기

앱은 만든 날부터 운영이 시작됩니다. 작은 앱이라도 다음은 정해 둡니다.

- **담당자**: 문제가 생기면 누가 고치는가.

- **백업**: 시트의 복사본을 주기적으로 만들어 둔다.
- **권한 정리**: 퇴사자나 담당이 바뀐 사람의 접근을 정기적으로 정리한다.
- **개인정보**: 시트에 담긴 개인정보의 보관 기간을 정하고 지난 기록을 지운다.
- **변경 기록**: 앱을 고칠 때마다 무엇을 왜 바꿨는지 짧게 남긴다. 세션 인계 문서(부록 B의 Session-Handoff 스킬)가 도움이 됩니다.
- **의견 수집**: 사용자가 불편한 점을 적을 곳을 하나 둔다.

문제가 생기면 입문 과정 4회차에서 연습한 질문 틀처럼 오류 메시지와 상황을 그대로 AI에게 보여 주고 원인을 좁힙니다. 버그가 어렵다면 디버깅 스킬을 씁니다(부록 B). 앱이 중요한 업무를 맡게 되면 계획을 더 깊게 세우고 자가검증을 거치는 방식으로 넘어갑니다(19장).

실습

실습 1. 요청 접수 앱 PRD 쓰기

아래 프롬프트로 한 장짜리 PRD를 만듭니다. AI가 질문을 먼저 하게 해서 빈칸을 줄입니다.

프롬프트

> 나는 [직무]이고, 우리 팀의 [요청 접수 업무]를 웹앱으로 만들고 싶어. 아직 코드를 만들지 말고 먼저 한 장짜리 PRD를 써 줘.
>
> [업무 설명]
> - 지금은 [메신저/구두/메일]로 요청을 받고 있어
> - 요청 유형은 [유형 목록]
> - 처리하는 사람은 [몇 명, 역할]
> - 불편한 점은 [누락, 진행 상황 모름, 월말 정리 어려움 등]
>
> PRD 작성 전에 꼭 필요한 질문만 객관식 위주로 한 번에 하나씩 물어봐 줘. 답이 모이면 아래 항목으로 PRD를 써 줘.
> 1. 문제 2. 사용자와 역할 3. 성공 기준(측정 가능한 문장으로) 4. 하지 않을 것(최소 3개, 로그인·결제·알림 문자 포함 여부 판단) 5. 제약(개인정보, 사용 기기) 6. 화면별 기능(각 기능마다 누르면 무슨 일이 일어나는지와 확인 기준 한 줄) 7. 저장할 데이터 항목과 상태 값
>
> 내가 말하지 않은 사실은 지어내지 말고 "가정:"이라고 표시해 줘. 처음 버전은 로그인 없이 가짜 데이터로 동작하는 데모를 기준으로 써 줘.

실습 2. 데모 빌드 요청하기

PRD가 나오면 파일로 저장한 뒤 Claude Code 세션을 엽니다(9장).

프롬프트

> 작업 폴더의 PRD.md를 읽고 이 PRD의 데모를 만들어 줘.
> - 로그인 없이 모든 기능을 눌러 볼 수 있어야 해.
> - 데이터는 코드 안의 가짜 데이터로 시작하되, 나중에 구글 시트로 바꿀 수 있게 데이터를 읽고 쓰는 부분을 한 파일로 분리해 줘.
> - PRD의 "하지 않을 것"은 만들지 마.
> - 비밀 값은 코드에 쓰지 말고 환경 변수 이름만 정해 둬. 값은 내가 직접 넣을 거야.
> - 끝나면 PRD의 확인 기준을 하나씩 어떻게 눌러 보면 되는지 단계별로 알려 줘.
> 먼저 만들 파일 목록과 계획을 보여 주고, 내가 확인하면 시작해 줘.

실습 3. 비밀 값 점검하기

프롬프트

> 이 프로젝트에 비밀 값이 코드나 저장소에 들어 있는지 점검해 줘. API 키, 서비스 계정 키, 비밀번호, 개인 이메일 주소가 파일에 직접 적혀 있는지 찾고, .env 파일이 .gitignore에 포함되어 있는지 확인해 줘. 발견한 것은 파일 이름과 위치만 알려 주고 값 자체는 출력하지 마.

선택 과제. 시트를 읽기 전용으로 게시해 화면에 비추기 (2단)

가상의 요청 목록 시트(개인정보가 없는 연습용 데이터)를 만들고 읽기 전용으로 게시합니다. 게시 방법은 구글 공식 안내를 따르고, 게시한 주소는 연습이 끝나면 게시를 중단합니다.

프롬프트

> 구글 시트를 읽기 전용으로 게시한 주소가 있어. 이 데이터를 읽어서 표로 보여 주는 화면만 만들어 줘. 쓰기 기능은 만들지 마. 시트 주소는 코드에 직접 쓰지 말고 환경 변수 이름으로 읽게 해 줘. 이 방식(2단)의 한계 두 가지도 마지막에 정리해 줘.

정리

- 웹앱은 여러 사람이 같은 데이터를 쓰고 상태를 추적해야 할 때 필요합니다. 시트나 폼으로 충분하면 거기서 멈춥니다.

- PRD는 문제, 사용자, 성공 기준, 하지 않을 것, 제약, 기능, 데이터를 한 장에 담고, "하지 않을 것"이 범위를 지켜 줍니다.
- 데모를 먼저 만들어 눌러 본 뒤 구글 시트 저장을 붙이고, 읽기 전용 게시(2단)로 충분한지 먼저 따져 봅니다. 새 앱은 원샷 프로토타이핑, 기존 앱의 기능 추가는 풀스택 기능 구현이 맞습니다.
- 서비스 계정 키와 API 키는 채팅에 붙여 넣지 않고, 코드에 쓰지 않고, 환경 변수로만 다루며, 새는 즉시 폐기합니다.
- 배포는 파일이 아니라 주소로 공유하는 일이며, 공식 문서를 따르고, 올린 뒤 다른 기기에서 실제 등록을 해 보며, 운영 담당자·백업·권한 정리·개인정보 보관 기간을 정합니다.

4부

품질과 운영

19장 · 계획의 깊이와 자가검증

20장 · 세션에서 배포까지

19장

계획의 깊이와 자가검증

AI에게 일을 맡길 때 품질은 두 곳에서 갈립니다. 하나는 시작 전에 계획을 얼마나 깊게 세우느냐이고, 다른 하나는 결과를 내놓기 전에 얼마나 다시 확인하느냐입니다. 이 장은 작업 크기에 맞춰 계획의 깊이를 고르는 법, 출력 직전에 검증하는 법, 그럴듯하지만 알맹이 없는 결과를 줄이는 법, 그리고 최종 판단은 사람이 한다는 원칙을 다룹니다.

계획에도 적정 깊이가 있습니다

모든 작업에 같은 무게의 계획을 붙이면 비효율이 생깁니다. 오타 하나 고치는 일에 설계 문서를 쓰면 흐름이 끊기고, 여러 파일을 건드리는 구조 변경을 계획 없이 시작하면 나중에 되돌릴 일이 쌓입니다. 작업 크기에 따라 세 단계로 나누면 판단이 쉬워집니다.

단계	대상	방식
No plan	오타 수정, 문구 교체, 간단한 정리	한두 문장으로 지시하고 바로 실행
Local plan	기능 하나 추가, 한 모듈 안의 중간 규모 수정	계획 모드로 계획을 먼저 보고 승인 후 실행
Two-pass	구조 변경, 이전 작업, 기존 동작이 바뀌는 큰 변경	계획을 세운 뒤 한 번 더 검토하고 실행

영향을 받는 파일의 범위가 대략의 기준입니다. 한두 문장으로 설명되고 어떤 파일이 바뀌는지 분명하면 No plan, 몇 개 파일이 바뀌지만 한 모듈 안에서 끝나면 Local plan, 여러 모듈에 걸치거나 되돌리기 어렵다면 Two-pass입니다. 파일 수보다 되돌리기 어려운 정도가 더 중요한 신호입니다. 파일은 하나뿐이어도 결제나 개인정보처럼 실수의 비용이 큰 영역이면 한 단계 위로 올립니다.

어떻게 고르는가

경계가 애매하면 한 단계 위로 올립니다. 계획이 과해서 잃는 것은 몇 분의 시간이지만, 계획이 부족해서 잃는 것은 작업 결과 자체일 수 있기 때문입니다. 다음 세 질문으로 빠르게 판단합니다.

1. 이 일을 두 문장으로 설명할 수 있는가. 설명할 수 없으면 계획이 필요합니다.
2. 틀렸을 때 한 번에 되돌릴 수 있는가. 되돌릴 수 없으면 한 단계 위로 올립니다.
3. 내가 결과를 눈으로 확인할 수 있는가. 확인 수단이 없으면 계획 단계에서 확인 방법부터 정합니다.

Local plan의 실제 모습

Claude Code에서는 계획 모드가 이 역할을 합니다. 계획 모드에서는 Claude가 파일을 읽고 계획을 제안하되, 사람이 승인하기 전에는 파일을 수정하지 않습니다. 계획 모드를 켜는 방법과 화면 표시는 버전에 따라 달라질 수 있으므로(2026년 11월 기준) 공식 문서에서 현재 방법을 확인합니다.

계획이 올라오면 다음 네 가지를 읽습니다.

- 건드리는 파일 목록이 예상과 맞는가
- 건드리지 말라고 한 부분이 목록에 들어 있지 않은가
- 확인 방법(테스트, 실행, 화면 점검)이 계획에 들어 있는가
- 중간에 막히면 어디서 멈추는가

네 가지 중 하나라도 비어 있으면 승인하지 않고 보완을 요청합니다. 계획을 읽는 일은 짧지만, 이 몇 분이 실행 뒤의 수정 시간을 크게 줄입니다.

Two-pass: 계획을 한 번 더 본다

Two-pass는 계획을 두 번 보는 방식입니다. 첫 번째 패스에서 Claude가 계획을 세우고, 두 번째 패스에서 다른 관점으로 그 계획을 검토합니다. 두 번째 검토는 같은 세션에서 "이 계획의 빠진 점과 위험을 찾아 달라"고 요청해도 되고, 새 세션을 열어 계획 문서만 넘겨 검토시켜도 됩니다. 새 세션이 낫습니다. 앞선 대화의 맥락이 없는 쪽이 계획의 허점을 더 잘 찾기 때문입니다. 새 세션과 서브에이전트를 활용하는 방법은 10장에서 다룹니다.

> **핵심** 계획의 깊이는 작업 크기가 아니라 틀렸을 때의 비용에 맞춥니다. 애매하면 한 단계 위로 올립니다.

시작 전 합의서를 기준으로 삼습니다

9장에서 다룬 합의서(헌법)는 계획과 검증이 만나는 지점입니다. 목표, 제약, 우선순위 세 가지가 적혀 있으면 계획을 검토할 때 "이 계획이 제약을 깨지 않는가"를 기준 삼아 볼 수 있고, 검증할 때도 같은 제약을 항목별로 다시 확인할 수 있습니다. 합의서가 없으면 검토 기준이 매번 기억에 의존합니다. 작업이 길어질수록 처음 정한 제약이 흐려지므로, 합의서는 계획의 출발점이자 검증의 채점표로 씁니다.

출력 직전에 한 번 더 확인시킵니다

AI가 그럴듯하지만 틀린 답을 내놓는 흔한 이유는, 내놓은 답을 스스로 다시 확인한 적이 없기 때문입니다. 사람도 쓴 글을 다시 읽지 않으면 오탈자를 놓칩니다. 자가검증은 AI에게 이 재확인을 의무로 붙이는 방법입니다. 4단계로 구성합니다.

1단계. 핵심 주장 세 개를 고릅니다

초안의 모든 문장을 검증할 필요는 없습니다. 틀리면 결과 전체가 무너지는 사실만 세 개 추립니다. 숫자, 날짜, 법적 근거, 외부 서비스의 동작 방식, 코드가 실제로 존재한다고 주장한 함수나 파일이 대표적입니다.

2단계. 독립적인 검증 질문을 만듭니다

각 주장을 초안과 떼어 놓고 "이것이 정말 맞는가"를 묻는 별도 질문으로 바꿉니다. 분리가 핵심입니다. 초안을 옆에 둔 채 검증하면 초안의 표현에 끌려가 그대로 맞다고 답하기 쉽습니다. 초안을 보지 않은 상태에서 질문만 받아 답하게 하면 어긋남이 드러납니다.

3단계. 검증 답변을 초안과 비교합니다

검증 답변이 초안과 다르면 초안을 고칩니다. 어긋나지 않았더라도 근거를 확인할 수 없는 주장은 "확인되지 않음"으로 표시하게 합니다. 모르는 것을 모른다고 적게 하는 것이 이 단계의 목적입니다.

4단계. 제약 체크리스트로 마무리합니다

마지막으로 합의서의 제약을 항목별로 다시 확인합니다. 예를 들어 "고객 실명은 쓰지 않는다", "기존 파일은 수정하지 않는다", "결과는 표로 낸다" 같은 항목이 모두 지켜졌는지 하나씩 예, 아니오로 답하게 합니다. 아니오가 있으면 고친 뒤 다시 내놓게 합니다.

이 4단계는 검증 질문을 따로 만들어 답하게 하는 연구(Chain-of-Verification)와 스스로 비판하고 고치는 연구(Self-Refine)의 발상을 업무용으로 줄인 것입니다. 이름을 외울 필요는 없고, "주장을 고르고, 따로 묻고, 비교하고, 제약을 확인한다"는 순서만 기억하면 됩니다.

한계를 알고 쓴다

자가검증이 만능은 아닙니다. 같은 모델에게 "이게 맞아?"라고 되묻는 검증에는 특히 한계가 있습니다. 1장의 가짜 판례 사례에서 챗봇은 자기가 만든 판례를 진짜라고 확인해 주었습니다. 독립된 질문, 별도의 세션, 공식 자료, 사람의 확인 가운데 하나는 반드시 거쳐야 합니다.

같은 모델이 같은 지식으로 검증하므로 모델이 애초에 모르는 사실은 걸러지지 않습니다. 이럴 때는 두 가지를 덧붙입니다. 첫째, 최신 정보나 수치는 공식 출처를 직접 확인하게 합니다(웹 검색이나 문서 조회가 가능한 환경이라면 출처를 함께 달게 합니다). 둘째, 중요한 결과는 사람이 원본 자료와 대조합니다. 자가검증은 사람의 검토 시간을 줄이는 도구이지 검토를 대신하는 도구가 아닙니다.

'AI 같은 결과'를 줄이는 법

내용은 맞는데 어딘가 밋밋하고, 어디서 본 듯한 글이 나오는 경우가 있습니다. 대개 지시가 막연해서 가장 평균적인 답이 나온 것입니다. 다음 다섯 가지를 지시에 넣으면 많이 달라집니다.

1. 읽는 사람을 정합니다. "신입 직원", "처음 문의하는 고객"처럼 구체적일수록 문장이 구체적이 됩니다.
2. 실제 자료를 붙입니다. 우리 조직의 실제 문서, 지난 답변, 말투 예시가 있으면 평균적인 문장에서 벗어납니다.
3. 쓰지 말아야 할 표현을 적습니다. "획기적", "최고의" 같은 과장 표현이나 반복되는 맺음말 습관을 금지 목록으로 줍니다.
4. 분량과 구조를 제한합니다. "세 문단 이내", "표 하나와 요약 두 줄"처럼 한계를 두면 군더더기가 줄어듭니다.
5. 첫 결과를 그대로 쓰지 않습니다. "가장 일반적인 문장 세 곳을 찾아 구체적인 문장으로 바꿔 달라"고 한 번 더 요청합니다.

디자인이나 문서 양식에도 같은 원리가 적용됩니다. 피해야 할 패턴을 목록으로 미리 합의해 두면 결과가 매번 같은 틀로 수렴하는 것을 막을 수 있습니다.

테스트와 점검은 AI가 아닌 도구에게 맡깁니다

코드를 다루는 작업에서 가장 믿을 만한 검증은 AI의 말이 아니라 실제로 실행한 결과입니다. 테스트, 문법 검사(lint), 타입 검사, 빌드가 여기에 해당합니다. 이 명령들은 같은 입력에 같은 결과를 냅니다. 이 성질을 재현성이라고 하며, 자가검증의 한계를 보완해 줍니다.

작업을 맡길 때 다음 규칙을 함께 줍니다.

- 실제로 실행한 명령의 결과만 보고합니다. 실행하지 않은 테스트를 통과했다고 쓰지 않습니다.
- 수정마다 테스트가 먼저 실패하고 수정 후에 통과하도록 합니다. 버그를 고칠 때는 같은 버그가 다시 생기지 않도록 그 상황을 재현하는 테스트를 남기게 합니다.

- 변경 전에 되돌릴 수 있는 지점(커밋)을 만듭니다. 20장에서 다룹니다.

코드가 아닌 작업에도 같은 정신이 통합니다. 데이터 분석이라면 합계가 원본 합계와 맞는지, 문서라면 인용한 수치가 원문과 같은지를 기계적으로 대조하는 단계를 둡니다.

품질을 한 번에 끌어올리는 순서

액티전스 프롬프트 카드 중 품질 파이프라인 카드는 테스트 보강, 코드 리뷰, 정리를 한 흐름으로 묶었습니다. 순서에는 이유가 있습니다. 테스트로 안전망이 없는 코드를 고치면 무엇이 깨졌는지 알 수 없으므로 테스트를 먼저 만듭니다. 리뷰하지 않은 코드를 정리하면 문제를 안은 채 모양만 다듬게 되므로 리뷰가 정리보다 앞섭니다. 즉 덮어 두지 않은 것은 리팩터링하지 않고, 검토하지 않은 것은 정리하지 않습니다. 카드의 자세한 내용은 부록 A에 있습니다.

AI 슬롭과 워크슬롭

AI로 대량 생산한 그럴듯하지만 알맹이 없는 결과물을 AI 슬롭(slop)이라고 부릅니다. 업무 문서에서는 워크슬롭(workslop)이라고 합니다. 이때 일이 사라진 것이 아니라 받는 사람에게 넘어간 것입니다. 만들기가 쉬워지면 이해하고 검토하는 일이 새로운 병목이 됩니다.

진짜 손해는 시간이 아니라 신뢰입니다. "이 사람이 보낸 건 한 번 더 확인해야 한다"는 인식이 생기면 그 사람의 모든 산출물이 느려집니다. 워크슬롭의 피해를 수치로 조사한 연구가 있다고 알려져 있으나, 이 책은 원출처를 확인하기 전에는 수치를 인용하지 않습니다.

보내기 전에 세 가지를 스스로 묻습니다.

1. 내가 내용을 이해했는가
2. 근거를 직접 확인했는가
3. 받는 사람이 할 일이 줄어드는가

세 질문 중 하나라도 아니오라면 보내지 않고 손봅니다.

최종 판단은 사람이 합니다

AI는 초안을 만들고 점검을 돕지만, 발송, 게시, 결제, 배포 같은 되돌리기 어려운 결정은 사람이 합니다. 검토할 때는 다음 질문을 던집니다.

- 이 결과가 틀렸을 때 누가 어떤 피해를 입는가
- 내가 직접 확인한 것과 AI의 말을 믿은 것이 구분되어 있는가
- 근거가 필요한 주장에 출처가 있는가

검토 시간은 위험에 비례해 씁니다. 내부 메모의 오타는 훑어보면 되고, 고객에게 나가는 안내문이나 법적 판단이 걸린 문서는 원본과 대조하면서 읽습니다. 법률, 세무, 의료처럼 전문 자격이 필요한 영역은 AI의 결과를 참고 자료로만 쓰고 해당 전문가에게 확인합니다.

초안, 검수, 발송과 기록

1장의 세 단계(AI가 초안, 사람이 검수, 발송과 기록)에서 이 장이 맡는 것은 2단계입니다. 3단계의 기록도 같은 흐름에 속합니다. 무엇을 검수하고 어디를 고쳤는지가 남아야 다음 일에서 같은 실수가 반복되지 않습니다. 기록을 조직의 자산으로 쌓는 방법은 22장에서 다룹니다.

사람이 직접 코멘트로 짚는 검토

AI에게 검토를 통째로 맡기지 않고, 사람이 읽으면서 고칠 곳에 쪽과 문단 위치를 적어 한 줄씩 코멘트로 넘기는 방법이 있습니다. 반영된 결과는 새 버전 파일로 저장하게 하고, 무엇을 왜 고쳤는지는 기록 파일에 한 줄씩 남기게 합니다. 고칠 곳을 짚을 때는 네 마디로 충분합니다.

마디	제안서와 기획서	결과보고서
첫째	확인(사실과 숫자가 의심될 때 다시 조사)	기록으로 확인
둘째	정합성(앞뒤 수치와 논리가 어긋날 때)	정합성
셋째	톤다운(어조가 너무 셀 때)	숫자로(형용사 자리에 숫자)
넷째	보수적으로(전망이 낙관적일 때)	순화하지 마(흐려진 미달을 그대로)

코멘트 예(가상의 예)입니다.

프롬프트

> 3쪽 둘째 문단: 근거가 오래됐어. 최신 자료로 다시 확인해줘.
> 5쪽 표: 앞 문단의 숫자와 정합성이 안 맞아.

검토가 쓰던 세션의 영향을 받지 않게 하는 방법도 있습니다. 쓰던 세션은 사정을 다 알아서 처음 읽는 사람이 걸리는 곳을 놓칩니다. 진행 기록을 갱신하고 세션을 닫은 뒤, 같은 폴더에서 새 세션을 열어 독자의 눈으로 검토만 시킵니다. 이때 "문서는 고치지 마"를 명시합니다. 숫자는 내 손으로 셋을 골라 원 기록까지

따라가 직접 합계를 내 봅니다. 셋이 맞으면 나머지를 믿을 근거가 생깁니다. 제안서의 흐름은 16장에서, 결과보고서의 흐름은 17장에서 자세히 다룹니다.

실습

실습 1. 자가검증 꼬리말 붙이기

평소 쓰는 요청 하나를 골라, 요청 끝에 아래 꼬리말을 그대로 붙여 보냅니다. 결과에서 검증 질문 답변과 수정된 부분이 실제로 나오는지 확인합니다.

프롬프트

> [출력 전 자가검증 — 아래 순서를 지킨 뒤 최종본만 보여 주세요]
> 1. 초안에서 틀리면 안 되는 핵심 주장 3개를 고릅니다.
> 2. 각 주장을 초안을 보지 않은 상태로 "이것이 사실인가"를 묻는 독립 질문으로 바꿔 스스로 답합니다.
> 3. 답이 초안과 다르거나 근거를 확인할 수 없으면 초안을 고치고, 확인할 수 없는 내용은 "확인되지 않음"이라고 표시합니다.
> 4. 이 요청에서 내가 정한 제약을 항목별로 나열하고, 각각 지켰는지 예/아니오로 확인합니다. 아니오가 있으면 고친 뒤 내보냅니다.
> 마지막에 "검증 요약" 3줄(고른 주장, 바뀐 내용, 확인되지 않은 내용)을 붙입니다.

실습 2. 계획의 깊이 판단 연습

지금 하려는 업무 세 가지를 떠올리고, 각각 No plan, Local plan, Two-pass 중 어디에 해당하는지 AI와 함께 판단해 봅니다.

프롬프트

> 아래 작업 3가지를 각각 No plan / Local plan / Two-pass 중 하나로 분류해 주세요.
> 판단 기준은 (1) 두 문장으로 설명 가능한가, (2) 틀렸을 때 되돌리기 쉬운가, (3) 결과를 내가 눈으로 확인할 수 있는가입니다.
> 애매하면 한 단계 위로 분류하고 이유를 한 줄로 적어 주세요.
> 각 작업마다 시작 전에 합의해야 할 제약을 2개씩 제안해 주세요.
>
> 작업 1: (내용)
> 작업 2: (내용)
> 작업 3: (내용)

실습 3. 밋밋한 결과 다듬기

최근에 AI가 만든 글이나 답변 하나를 붙여 넣고 다음 요청을 보냅니다.

프롬프트

> 아래 글에서 어디서나 쓸 수 있는 일반적인 문장 3곳을 찾아 이유를 설명하고, 읽는 사람이 [읽는 사람 설명]이라는 전제로 구체적인 문장으로 바꿔 주세요.
> 과장 표현(획기적, 최고의 등)과 반복되는 맺음말은 쓰지 않습니다.
> 바꾼 문장은 원문과 나란히 표로 보여 주세요.
>
> (여기에 글 붙여 넣기)

선택 과제. 보내기 전 질문 세 개

최근에 만든 결과물 하나에 "내가 이해했는가, 근거를 확인했는가, 받는 사람이 할 일이 줄어드는가" 세 질문을 적용해 봅니다. 아니오가 나온 항목을 한 줄씩 적고 고쳐서 다시 확인합니다.

정리

- 계획의 깊이는 작업 크기와 틀렸을 때의 비용에 맞춥니다. 애매하면 한 단계 위로 올립니다.
- 자가검증은 핵심 주장 고르기, 독립 질문, 비교, 제약 체크리스트의 4단계입니다.
- 자가검증은 사람의 검토를 줄여 줄 뿐 대신하지 않습니다. 최신 정보와 수치는 원본으로 대조합니다.
- 밋밋한 결과는 읽는 사람, 실제 자료, 금지 표현, 분량 제한을 지시에 넣어 줄입니다.
- 같은 모델에게 되묻는 검증은 한계가 있으므로 독립된 질문, 별도 세션, 공식 자료, 사람의 확인을 거칩니다. 워크슬롭은 일을 받는 사람에게 넘기는 것이니 보내기 전에 세 질문으로 점검합니다.
- 코드 검증은 AI의 말이 아니라 실제로 실행한 테스트와 빌드 결과를 기준으로 하고, 최종 판단은 사람이 합니다.

20장

세션에서 배포까지

AI와 만든 결과물은 대화창 안에 머물면 아직 일이 끝난 것이 아닙니다. 이 장은 준비, 개발, 검증, 배포의 네 단계를 한 흐름으로 묶고, 그 바탕이 되는 버전 관리(git), 배포와 되돌리기, 비밀 값 관리, 세션 사이의 인계를 설명합니다.

네 단계 흐름

액티전스 라이브러리는 작업을 준비, 개발, 검증, 배포의 네 단계로 정리합니다. 외울 필요는 없고, 지금 하는 일이 어느 단계인지 알면 필요한 도구가 정해집니다.

단계	하는 일	남기는 것
준비	합의서 작성, 목표와 제약 정리, 새 세션 시작 또는 이전 세션 이어받기	합의서, 작업 범위
개발	기능 추가, 품질 개선, 디버깅	커밋, 진행 기록
검증	자가검증, 리뷰, 테스트, 실제 화면 확인	검증 결과
배포	패키징, 공개, 공개 후 확인	배포 주소, 되돌릴 대상

각 단계마다 해당 프롬프트 카드가 있고, 단계 사이에 넘기는 정보의 형식도 정해 둘 수 있습니다. 카드 목록은 부록 A에 있습니다.

도구 네 가지의 역할

만든 결과물이 파일로만 남으면 만든 사람만 쓰고 조직에는 남지 않습니다. 파일에서 주소로, 주소에서 데이터로 옮겨 가는 데는 네 가지 도구가 각자의 역할을 맡습니다.

도구	역할	없으면
Claude Code	만드는 사람	만들 수 없습니다
GitHub	설계도 보관소와 변경 이력	되돌릴 수 없습니다
호스팅 서비스(예: Vercel)	주소를 주고 자동으로 배포	나만 쓰게 됩니다
구글 시트와 클라우드 설정	데이터 장부	데이터를 저장하지 못합니다

순서는 만든다, 올린다, 주소를 받는다, 데이터를 관리한다입니다. 한 칸씩 올라갑니다. 파일과 주소의 차이는 18장에서 비교합니다. GitHub에 올리면 호스팅 서비스가 변경을 감지해 다시 배포하는 관계는 한 번만 연결해 두면 됩니다. 그 뒤로는 고쳐 달라는 한마디로 주소의 내용이 최신이 됩니다. 연동 방법과 무료 사용 범위는 각 서비스의 공식 문서에서 확인합니다(2026년 11월 기준).

준비

준비 단계의 산출물은 합의서입니다. 9장에서 만든 합의서가 이미 있다면 이번 작업에 맞게 목표와 제약을 갱신하고, 없으면 짧게라도 새로 씁니다. 새 프로젝트라면 어떤 도구를 쓰고 어디에 배포할지도 이 단계에서 정합니다. 이전 작업을 이어받는다면 마지막 인계 문서를 먼저 읽게 합니다(아래 "세션 사이의 인계" 참조).

개발

개발 단계에서는 19장에서 정한 계획의 깊이로 작업합니다. 작업 단위는 작게 쪼갭니다. 한 번에 많이 바꾸면 무엇이 문제를 일으켰는지 찾기 어렵습니다. 작은 단위가 끝날 때마다 커밋을 남기는 습관이 이 단계의 핵심입니다.

검증

검증은 AI의 말이 아니라 실행 결과로 합니다. 테스트와 빌드를 돌리고, 화면이 있는 결과물이라면 실제로 열어 눈으로 봅니다. 핵심 주장 확인과 제약 체크리스트는 19장의 방법을 그대로 씁니다. 검증에서 실패하면 개발 단계로 돌아가는 것이 정상입니다.

배포

배포는 결과물을 다른 사람이 쓸 수 있는 곳에 올리는 일입니다. 올리기 전 점검, 올린 뒤 확인, 문제가 생겼을 때 되돌릴 방법을 한 세트로 준비합니다.

---

git: 변경의 안전장치

git은 파일의 변경 이력을 기록하는 버전 관리 도구입니다. AI가 파일을 여러 개 수정하는 작업에서는 사실상 필수입니다. 잘못 고쳐졌을 때 이전 상태로 돌아갈 방법이 있어야 마음 놓고 맡길 수 있기 때문입니다. 터미널 명령에 익숙하지 않아도, 아래 네 가지 개념만 알면 됩니다.

- **커밋(commit)**: 지금 상태를 이름표와 함께 저장하는 것. 게임의 저장 지점과 같습니다.
- **이력(history)**: 커밋이 쌓인 목록. 언제 무엇이 바뀌었는지 볼 수 있습니다.

- **되돌리기(revert, restore)**: 이전 커밋의 상태로 돌아가는 것.
- **브랜치(branch)**: 본 흐름을 건드리지 않고 시험 삼아 작업하는 갈래.

기본 흐름

명령

```
git status
git add .
git commit -m "문의 응대 안내문 템플릿 추가"
git log --oneline
```

`git status`는 지금 바뀐 파일을 보여 주고, `git add .`는 바뀐 파일을 저장 대상으로 올리고, `git commit`은 그 상태를 이름표와 함께 저장합니다. `git log --oneline`은 지금까지의 커밋을 한 줄씩 보여 줍니다. 위 명령은 읽기만 해도 이해할 수 있도록 단순한 것만 골랐습니다.

되돌리기

커밋이 있으면 되돌릴 수 있습니다. 가장 안전한 방법은 문제 커밋의 변경을 거꾸로 적용하는 새 커밋을 만드는 것입니다.

명령

```
git revert <되돌릴 커밋 번호>
```

이 방식은 이력을 지우지 않고 되돌린 사실까지 기록으로 남깁니다. 이력을 지우거나 강제로 덮어쓰는 명령(`git reset --hard`, `git push --force` 등)은 작업 내용을 잃을 수 있으므로, AI에게 맡길 때도 사람의 확인 없이는 실행하지 않도록 합의서에 적어 둡니다.

AI에게 커밋을 맡길 때의 규칙

- 변경하기 전에 먼저 커밋해서 되돌릴 지점을 만듭니다.
- 커밋 하나에는 한 가지 목적만 담습니다. 기능 추가와 정리를 섞지 않습니다.
- 커밋 메시지는 무엇을 왜 바꿨는지 한 줄로 씁니다.
- 커밋과 원격 저장소로의 업로드(push)는 요청했을 때만 하게 합니다.

git을 더 깊이 배우고 싶다면 무료로 공개된 Pro Git 한국어판을 검색해 보세요. 처음에는 시작하기와 git의 기초 두 장만 읽으면 충분합니다.

비밀 값을 지키는 법

API 키, 비밀번호, 서비스 연결 주소처럼 외부에 알려지면 안 되는 값을 비밀 값이라고 합니다. AI와 함께 코드를 만들 때 가장 흔하고 피해가 큰 실수가 이 값을 코드나 대화에 그대로 적는 것입니다.

원칙

1. 비밀 값은 코드 파일에 쓰지 않고 환경 변수에 둡니다. 로컬에서는 보통 `.env` 같은 파일에 모읍니다.
2. 그 파일은 git이 추적하지 않도록 `.gitignore`에 등록합니다. 등록하지 않으면 커밋과 함께 이력에 남습니다.
3. 대화창에 비밀 값을 붙여 넣지 않습니다. 값이 필요하면 "환경 변수 이름만 사용하고 값은 내가 직접 넣겠다"고 지시합니다.
4. 비밀 값이 들어가는 예시 파일은 값을 비운 형태(`.env.example`)로 만들어 어떤 변수가 필요한지만 알립니다.
5. 배포 서비스에는 그 서비스의 환경 변수 설정 화면에 사람이 직접 입력합니다.

공개 저장소의 위험

공개로 설정된 저장소에 비밀 값이 한 번이라도 올라가면, 나중에 파일에서 지워도 이력에 남습니다. 이미 올라갔다면 파일을 고치는 것으로는 부족하고, 해당 키를 발급처에서 폐기하고 새로 발급해야 합니다. 이것이 가장 확실한 대응이며, 이력을 정리하는 작업은 그 뒤의 일입니다. 저장소를 공개로 바꾸기 전에는 이력 전체에 비밀 값이 없는지 점검하게 하고, 점검 결과는 사람이 확인합니다.

AI에게 지시하는 방법

프롬프트

> 비밀 값 규칙: API 키와 비밀번호는 코드와 대화에 쓰지 않고 환경 변수 이름으로만 참조합니다. .env는 .gitignore에 있는지 확인하고, 커밋 전에 변경 목록에 비밀 값이 포함되지 않았는지 점검합니다.

이 문장을 합의서의 제약에 한 줄로 넣어 두면 세션마다 반복해서 말할 필요가 없습니다.

배포와 되돌리기

배포 대상은 작업에 따라 다릅니다. 웹 서비스라면 호스팅 서비스, 도구라면 사내 공유 위치, 문서라면 공유 폴더일 수 있습니다. 어떤 대상이든 다음 세 가지는 같습니다.

올리기 전

- 테스트와 빌드가 통과했는가
- 비밀 값이 코드나 이력에 들어가지 않았는가
- 지금 운영 중인 버전이 무엇인지 기록했는가. 이것이 되돌릴 대상입니다.
- 배포 후 확인할 항목(주소 접속, 핵심 기능 하나, 오류 로그)을 정했는가

올린 직후

배포가 성공했다는 메시지는 확인의 끝이 아닙니다. 공개 주소에 실제로 접속해 핵심 기능을 하나 이상 눌러 보고, 오류가 없는지 확인합니다. 로그인하지 않은 시크릿 창에서 열면 공개 상태에서 무엇이 보이는지 그대로 알 수 있습니다. 호스팅 서비스마다 배포 이력과 이전 배포로 되돌리는 기능을 제공하는 경우가 많으므로, 사용하는 서비스에서 되돌리는 방법을 배포하기 전에 한 번 확인해 둡니다.

문제가 생겼을 때

1. 먼저 되돌립니다. 원인 분석은 되돌린 뒤에 합니다.
2. 이전 버전으로 복구된 것을 공개 주소에서 확인합니다.
3. 무엇이 문제였는지 기록하고, 같은 문제가 반복되지 않도록 검증 항목에 추가합니다.

배포는 되돌리기 어려운 결정에 속하므로 19장에서 말한 대로 최종 확인은 사람이 합니다. AI에게 배포를 맡기는 경우에도 "배포 직전에 점검 결과를 보고하고 내 승인을 받은 뒤에 실행한다"는 조건을 둡니다.

쓰고 난 뒤의 정리

공용 컴퓨터이거나 더 쓰지 않을 도구는 퇴장 절차를 밟습니다. 앱과 배포 서비스에서 로그아웃하고, 필요하면 프로그램을 제거하고, 작업 폴더와 설정 폴더에 남은 파일을 정리합니다. 폴더를 지우기 전에 복사해 두어야 하며, 이미 올린 주소는 앱을 지워도 그대로 열린다는 점을 기억합니다. 처음 만드는 사람을 위한 순서는 6장에 있습니다.

세션 사이의 인계

긴 작업은 한 세션에서 끝나지 않습니다. 대화가 길어지면 앞의 내용이 흐려지고 응답이 느려지므로, 중간에 새 세션으로 옮기는 것이 정상입니다. 이때 필요한 것이 인계 문서입니다. 인계 문서가 없으면 새 세션은 처음부터 다시 파악해야 하고, 그 시간이 낭비됩니다.

인계 문서에 담을 것

- **변경한 것**: 이번 세션에서 바꾼 파일과 내용
- **유지할 것**: 바꾸면 안 되는 결정과 제약
- **하지 않은 것**: 이번에 일부러 손대지 않은 범위
- **다음에 할 일**: 새 세션이 가장 먼저 해야 하는 일과 확인 명령

변경, 유지, 범위 밖의 세 축으로 적으면 새 세션이 무엇을 건드려도 되는지 바로 알 수 있습니다. 진행 상황은 별도의 진행 기록 파일에 한 줄씩 남기고, 세션이 끊겨도 그 파일을 읽으면 이어갈 수 있게 합니다.

진행 기록과 이유 기록을 나누기

인계 문서와 별개로, 폴더에 두 파일을 따로 남기는 운영이 효과적입니다. PROGRESS.md에는 지금 어디까지 했고 다음 시작점이 무엇인지를 적고, HISTORY.md에는 무엇을 왜 그렇게 정했는지를 한 줄씩 적습니다. 앞의 것은 이어서 할 때 읽고, 뒤의 것은 되돌아가야 할 때 근거로 읽습니다. 둘을 섞으면 진행 상황은 길어지고 결정의 이유는 묻힙니다.

세션을 닫기 전에는 PROGRESS.md를 먼저 갱신합니다. 이어서 할 세션은 그 파일을 읽고 현재 상태부터 말하게 합니다. 쓰던 세션은 사정을 다 알아서 처음 읽는 사람이 걸리는 곳을 놓치기 쉽습니다. 그래서 일이 끝나면 쓰던 세션을 닫고, 같은 폴더에서 새 세션을 열어 검토만 시킵니다. 이때 문서를 고치지 말고 의견만 파일로 남기게 하면, 반영은 그 의견을 보고 사람이 지시합니다. 검증 방법은 19장에서 더 다룹니다.

언제 새 세션으로 넘어가는가

한 세션이 길어졌다는 신호는 다음과 같습니다.

- 앞에서 정한 제약을 어기기 시작한다
- 같은 설명을 반복해서 요구한다
- 응답이 눈에 띄게 느려진다

이런 신호가 보이면 인계 문서를 작성하고 새 세션을 엽니다. 작업 단위 하나가 끝난 지점에서 넘어가는 것이 가장 깔끔합니다. 자동 요약 기능이 있는 환경이라도 중요한 결정은 문서로 따로 남기는 편이 안전합니다.

품질 파이프라인 카드

액티전스 프롬프트 카드 중 품질 파이프라인 카드는 검증 단계를 한 번에 돌리는 카드입니다. 테스트 보강, 코드 리뷰, 정리의 세 단계로 이루어지며, 앞 단계가 끝나야 뒤 단계로 갑니다.

- 시작 전에 작업 대상의 종류(웹 앱, 도구, 라이브러리 등)와 쓸 수 있는 검사 명령을 코드에서 직접 찾아 한 줄로 기록합니다. 정의되어 있지 않은 명령은 실행하지 않습니다.
- 해당하지 않는 검사는 사유와 함께 제외합니다. 예를 들어 화면이 없는 도구에는 화면 비교 검사를 적용하지 않습니다.
- 테스트가 없는 코드는 먼저 테스트로 감싸고, 리뷰를 마친 뒤에만 정리합니다.

새 기능을 만든 뒤 안정화하거나 오래된 코드를 다듬을 때 쓰는 카드입니다. 사용법은 카드를 입력창에 붙여 넣고 그 아래에 "결제 모듈 안정화"처럼 작업 대상을 한 줄 적는 것입니다(가상의 예). 전체 카드는 부록 A에서 찾을 수 있고, 라이브러리 화면에서 복사할 수도 있습니다(https://actigence.ai).

실습

실습 1. 배포 전 점검 요청

작업 중인 결과물이 있다면 배포나 공유 직전에 아래 요청을 보냅니다. AI의 점검 결과를 그대로 믿지 말고, 보고된 항목 중 두세 개는 직접 확인합니다.

프롬프트

> 배포 전 점검을 해 주세요. 실행하지 않은 것을 통과했다고 쓰지 마시고, 실제로 실행한 명령과 결과만 보고해 주세요.
> 1. 이 프로젝트에서 정의된 테스트·검사·빌드 명령을 찾아 실행하고 결과를 표로 보고합니다. 정의되지 않은 항목은 "없음"으로 적습니다.
> 2. 코드와 변경 목록에 API 키, 비밀번호, 개인정보가 들어 있는지 검색하고, .env 계열 파일이 .gitignore에 있는지 확인합니다.
> 3. 지금 운영 중인 버전이 무엇인지(되돌릴 대상)와 되돌리는 방법을 한 줄로 적습니다.
> 4. 배포 후 확인할 항목 3가지를 제안합니다.
> 5. 위 결과를 근거로 "배포 가능 / 보류"를 판단하되, 실제 배포는 하지 말고 제 승인을 기다립니다.

실습 2. 인계 문서 작성

오늘 한 작업을 마무리하며 새 세션으로 넘길 문서를 만듭니다.

프롬프트

> 이번 세션의 작업을 다음 세션이 처음부터 다시 파악하지 않고 이어받을 수 있도록 인계 문서를 만들어 주세요.
> 형식은 다음 네 섹션입니다.
> - 변경한 것: 바꾼 파일과 내용
> - 유지할 것: 바꾸면 안 되는 결정과 제약
> - 하지 않은 것: 일부러 손대지 않은 범위
> - 다음에 할 일: 가장 먼저 할 일과 확인 방법
> 시행착오 과정은 빼고 확정된 결과와 그 근거만 적어 주세요. 확인되지 않은 내용은 "미확인"으로 표시합니다.

PROGRESS.md와 HISTORY.md를 나누는 버전으로도 해 봅니다. 아래 프롬프트는 같은 작업을 두 파일로 남깁니다.

프롬프트

> 이번 세션의 작업을 두 파일로 남겨 주세요.
> - PROGRESS.md: 목표, 상태, 결정, 다음, 막힌 것 다섯 칸으로 지금 어디까지 했고 다음 시작점이 무엇인지만 적습니다.
> - HISTORY.md: 이번에 정한 것과 그 이유를 한 줄씩 적습니다.
> 확인되지 않은 내용은 "미확인"으로 표시하고, 파일을 만든 뒤 각 파일을 한 줄씩 설명해 주세요.

새 세션을 열고 "이어서 하자"라고만 입력해, Claude가 PROGRESS.md를 읽고 현재 상태를 먼저 말하는지 확인합니다.

실습 3. 되돌리기 연습

연습용 폴더에서 파일 하나를 만들고 커밋한 뒤, 일부러 내용을 망가뜨리고 이전 상태로 되돌려 봅니다. 중요한 파일이 있는 폴더에서는 하지 않습니다.

프롬프트

> 연습용 빈 폴더에서 git 저장소를 만들고, 메모 파일 하나를 만들어 커밋해 주세요. 그다음 파일 내용을 일부러 바꾸고 커밋한 뒤, git revert로 직전 변경을 되돌려 주세요. 각 단계에서 실행한 명령과 git log --oneline 결과를 보여 주고, 무슨 일이 일어났는지 비전문가 눈높이로 설명해 주세요.

정리

- 작업은 준비, 개발, 검증, 배포의 네 단계로 보고, 각 단계의 산출물을 남깁니다. 만들기, 보관, 주소, 데이터는 Claude Code, GitHub, 호스팅 서비스, 구글 시트가 나눠 맡습니다.

- git의 커밋은 되돌릴 지점입니다. 변경 전에 커밋하고, 한 커밋에는 한 목적만 담습니다.
- 비밀 값은 코드와 대화에 쓰지 않고 환경 변수에 두며, 공개 저장소에 올라갔다면 키를 폐기하고 새로 발급합니다.
- 배포는 점검, 공개 주소 확인, 되돌릴 대상 기록을 한 세트로 하고, 문제가 생기면 원인 분석보다 되돌리기가 먼저입니다.
- 세션 사이에는 변경, 유지, 범위 밖, 다음 할 일을 담은 인계 문서로 이어가고, 진행 기록과 이유 기록은 파일을 나눠 남깁니다.

5부

조직에 심기

21장 · 실행형 AX 컨설팅

22장 · 사내 AX 담당자와 첫 3개월

21장

실행형 AX 컨설팅

AX(AI Transformation, 인공지능 전환)는 조직의 일하는 방식을 AI에 맞게 바꾸는 일입니다. 이 장은 액티전스가 제안하는 실행형 AX 컨설팅이 무엇이고, 교육이나 자문과 어떻게 다르며, 어떤 구조로 진행되고, 언제 끝나는지를 설명합니다. 컨설팅을 받을 조직뿐 아니라 사내에서 AX를 직접 추진하려는 사람에게도 일의 구조를 이해하는 데 쓸 수 있습니다.

이 장에 나오는 AX, 워크플로우, 단일 원본(SSOT), 챔피언의 뜻은 7장에서 정의했습니다. AI 활용과 AX의 비교 표, 직원이 자라는 네 단계도 7장에 있으므로 먼저 읽고 오면 이 장의 설명이 더 쉽게 이어집니다.

실행형 AX 컨설팅이란

한 줄로 정의하면 이렇습니다. 진단만 하고 보고서를 남기는 것이 아니라, 필요한 도구를 직접 만들어 전달하고 그것을 운영할 사람까지 세우는 컨설팅입니다.

AI 도입이 실패하는 흔한 경로가 있습니다. 교육을 받고 돌아왔지만 업무에 적용할 시간이 없고, 자문을 받아 계획서를 얻었지만 실행할 사람이 없는 경우입니다. 실행형은 이 간극을 줄이는 데 초점을 둡니다. 컨설턴트가 사무실에서 담당자 옆에 앉아 도구를 함께 만들고, 마지막에는 담당자가 혼자 돌릴 수 있는 상태로 넘깁니다.

자문형과 실행형

구분	자문형	실행형
산출물	진단 보고서, 전략 제언	작동하는 도구와 운영 담당자
실행 주체	고객사	컨설턴트가 직접 수행
끝났을 때	무엇을 해야 하는지 안다	이미 돌아가고 있다
실패하는 지점	실행되지 않으면 결과가 없다	범위를 좁게 잡지 않으면 미완성이 된다

자문형은 방향을 제시하는 데 강점이 있고, 실행형은 만들어 넘기는 데 강점이 있습니다. 어느 쪽이 낫다기보다 조직의 상황에 맞는 쪽이 있습니다. 방향 자체가 불분명하다면 자문이 먼저일 수 있고, 하고

싶은 일은 분명한데 손이 부족하다면 실행형이 맞습니다.

교육, 자문, 실행형 비교

구분	교육	자문형	실행형 AX
하는 일	방법을 가르친다	방향을 제시한다	직접 만들고 넘긴다
주체	직원이 배워서	고객사가 실행	컨설턴트와 담당자가 함께
다루는 것	도구를 쓰는 역량	전략과 구조	업무 구조와 실제 도구
끝났을 때	할 줄 아는 사람	계획서	돌아가는 도구와 운영할 사람

교육이 끝나면 배움이 남고, 자문이 끝나면 계획이 남고, 실행형 AX가 끝나면 작동하는 시스템과 그것을 돌릴 사람이 남습니다. 이 책의 1부부터 4부까지는 교육에 해당하는 내용이고, 5부는 그 배움을 조직에 정착시키는 방법입니다.

대행과의 차이

대행은 시킨 일을 하고 계속 맡습니다. 실행형은 무엇을 만들지 찾아내는 일부터 시작하고, 일정 기간 뒤에 손을 뗍니다. 그래서 앞부분의 진단과 설계, 뒷부분의 이관과 자립이 서비스의 절반을 차지합니다.
만들기만 하는 것이 아니라 넘기는 것까지가 일입니다.

누가 무엇을 하는가

역할은 세 갈래로 나뉩니다.

- **컨설턴트가 직접 하는 일**: 워크플로우 분석, AX 포인트 도출, 솔루션 설계와 개발, 사용 절차서 작성
- **함께 하는 일**: 우선순위 결정, 개발 범위 확정, 중간 결과 검토, 완성 기준 합의
- **고객사가 하는 일**: 업무 설명과 자료 제공, 담당자 지정, 방문일 동석, 실제 사용과 운영

AX 포인트는 AI나 자동화로 개선할 여지가 큰 업무 지점을 뜻합니다.

DIY, DFY, DWY

AI 활용을 조직에 들이는 방식은 세 가지로 구분할 수 있습니다.

- **DIY(Do It Yourself)**: 배워서 직접 한다. 교육과 특강이 여기에 해당합니다.
- **DFY(Done For You)**: 대신 해 준다. 진단과 개발이 여기에 해당합니다.

- **DWY(Done With You)**: 함께 하며 넘긴다. 이관과 교육이 여기에 해당합니다.

실행형 AX는 DFY로 시작해 DWY를 거쳐 DIY로 끝나도록 설계합니다. 처음에는 컨설턴트가 만들고, 중간에는 함께 만들고, 끝에는 담당자가 혼자 합니다.

AX가 바꾸는 네 층위

AX는 도구 하나를 도입하는 일이 아니라 일하는 방식의 층을 바꾸는 일입니다. 네 층위로 나누면 무엇을 진단하고 무엇으로 바꾸는지 분명해집니다.

층위	진단 대상	전환 후
업무 절차	사람의 기억과 구두 전달에 의존하는 규칙	문서화된 표준 절차
반복 작업	매번 백지에서 시작하는 글쓰기와 정리	저장된 프롬프트와 자동화로 실행
의사결정	감과 경험에 의존	데이터 기반 정기 보고
지식 자산	담당자가 나가면 사라짐	조직에 쌓이는 문서와 도구

네 번째 층위가 특히 중요합니다. 담당자의 머릿속에만 있던 노하우가 문서와 도구로 남으면 사람이 바뀌어도 조직에 남습니다.

진행 구조: 12회 반복 사이클

> **핵심** 아래 구조는 제안서 기준 예시 모델입니다. 기간과 방문 횟수, 방문 시간은 조직의 규모와 상황에 맞춰 조정할 수 있습니다.

예시 모델은 3개월 동안 주 1회, 한 번에 6시간, 총 12회 안팎(약 72시간)을 고객사 사무실에서 진행합니다. 사내 AX 담당자 1인이 필수이고, 종료 조건은 기간 만료가 아니라 인수인계 완료입니다. 종료 이후의 유지보수와 자문은 선택입니다. 착수는 1시간 상담으로 시작합니다. 이 상담에서 현재 상황을 듣고 진행할지, 어떤 방식으로 진행할지를 함께 정합니다. 비용 등 계약 조건은 상담에서 별도로 안내하며 이 책에서는 다루지 않습니다.

선형이 아니라 반복

흔히 1개월 진단, 2개월 개발, 3개월 이관의 직선 단계를 떠올리지만, 이 구조는 12회를 돌리는 반복 사이클입니다. 이유는 두 가지입니다. 첫째, 사전 자료가 충분하면 첫 방문 1시간 미팅만으로도 AX 포인트의 상당 부분이 나옵니다. 둘째, 진짜 문제는 초안을 만들어 봐야 드러납니다. 진단을 몇 주 동안 끝낸 뒤에 개발을 시작하면 늦습니다.

방문마다 반복되는 5단계

매 방문은 같은 다섯 단계로 진행합니다.

1. **지난주 점검**: 지난 방문에서 넘긴 과제가 어디까지 진행됐는지 확인합니다.
2. **진단과 파악**: 새로 드러난 문제와 막힌 지점을 현장에서 살핍니다.
3. **개발**: 우선순위가 높은 항목의 도구와 문서를 만듭니다.
4. **초안 확인**: 만든 초안을 담당자가 직접 써 보고 수정할 점을 말합니다.
5. **다음 주 과제 인계**: 방문일 사이에 담당자가 혼자 해 볼 과제를 정해 넘깁니다.

방문 사이의 6일

이 구조에서 방문 사이의 6일은 쉬는 시간이 아닙니다. 담당자가 혼자 과제를 실행하고 막히는 시간입니다. 막힌 지점은 다음 방문의 교육 주제가 됩니다. 혼자 해 보고, 막히고, 배우고, 다시 해 보는 과정을 12번 반복하면 담당자의 실력이 자랍니다. 컨설턴트가 옆에서 모든 것을 해 주면 방문이 끝난 뒤 바로 멈춥니다.

사무실에 상주하는 이유

어떤 문의가 하루에 몇 번 들어오는지, 어떤 문서를 몇 번씩 다시 쓰는지는 옆에서 봐야 알 수 있습니다. 인터뷰에서는 "늘 하던 일"이라 말하지 않는 부분이 많습니다. 한 화면을 같이 보고, 손은 담당자가 움직입니다. 컨설턴트가 키보드를 잡는 시간이 길어질수록 담당자의 습득이 줄어들기 때문입니다.

시기별 비중 변화

시기	비중이 큰 활동	주도
초반(1~4회)	진단	컨설턴트가 초안 제작
중반(5~8회)	개발	컨설턴트와 담당자 병행
후반(9~12회)	이관	담당자 주도, 컨설턴트는 검토와 예외 처리

초반에는 컨설턴트가 먼저 만들어 보여 주고, 후반에는 담당자가 만들고 컨설턴트가 검토합니다. 이관은 마지막 달에 몰아서 하지 않고 매주 조금씩 합니다.

착수 준비와 세 가지 활동

착수 전에 준비할 네 가지는 22장에서 자세히 다룹니다. 요약하면 담당자의 좌석과 워크스테이션, 사내 AX 담당자 선정, AI 계정을 조직 명의로 생성해 결제하는 것, 사전 자료(문제 목록과 워크플로우)입니다.

진행 중의 활동은 세 가지로 묶입니다.

A. 진단과 파악

"AI로 무엇을 할 수 있는가"가 아니라 "지금 어디가 막혀 있는가"에서 출발합니다. 현장 관찰과 인터뷰, 업무 단계 분해, 반복과 중복과 병목 식별, AX 포인트 우선순위 정하기가 포함됩니다. 산출물은 업무 흐름도, AX 포인트 목록, 우선순위표입니다.

B. 개발

실제 데이터, 문서, 말투를 반영해 도구를 만듭니다. 매 회차마다 초안을 시연하고 수정하며, 예외 상황을 처리하는 방법도 설계합니다. 산출물은 작동하는 솔루션 세트와 사용 절차서입니다.

C. 이관과 교육

담당자가 직접 조작하고, 수정하고 확장하는 방법을 배우고, 계정, 문서, 자산이 사내로 넘어왔는지 확인합니다. 산출물은 운영 절차서, 주의사항, 인계 확인서입니다.

산출물의 유형

- 프롬프트 자산: 상담 스크립트 생성기, 콘텐츠 템플릿
- 문서 체계: 운영 매뉴얼, 표준 작업 절차(SOP, Standard Operating Procedure), 온보딩 가이드
- 자동화 흐름: 월간 보고 루틴, 공지 일괄 생성
- 콘텐츠 생성 체계: 이미지와 영상 제작 절차
- 운영 절차서: 사용 절차, 주간 체크리스트
- 운영 인력: 사내 AX 담당자

마지막 항목이 빠지면 안 됩니다. 도구는 돌릴 사람이 없으면 멈춥니다.

일반화한 예를 들면 회원 응대용 상황별 메시지 생성기, 규정에 근거한 문의 답변 초안 생성기(발송 연동 포함), 월간 운영 보고 자동화, 교육 과정 운영 문서 자동 생성 같은 것들입니다(가상의 예). 어떤 업종에서든 반복되는 글쓰기, 정리, 보고가 대상입니다.

완료 기준

완료는 기간이 아니라 상태로 판단합니다. 아래 세 가지가 모두 충족되어야 합니다.

1. 담당자가 혼자 각 솔루션을 실행할 수 있다.
2. 새 요구에 맞춰 수정할 수 있다.
3. 계정, 문서, 프롬프트 자산이 사내에 모두 남아 있다.

세 번째가 자주 놓치는 항목입니다. 컨설턴트의 개인 계정에서 만든 도구는 컨설턴트가 떠나면 함께 사라집니다. 처음부터 조직 명의 계정에서 작업해야 하는 이유입니다.

일하는 원칙

작업 원칙

- **진단 우선**: 문제를 특정하기 전에는 도구를 만들지 않습니다.
- **현장 관찰**: 말로 설명된 업무와 실제 업무는 다를 수 있으므로 직접 봅니다.
- **직접 실행, 즉시 착수**: 설명서만 쓰지 않고 바로 만들어 봅니다.
- **주 단위 검토, 매주 이관**: 마지막에 몰아서 넘기지 않습니다.
- **동석 개발**: 담당자와 한 화면을 보며 만듭니다.
- **자산 잔존**: 끝났을 때 모든 자산이 조직에 남습니다.
- **사람이 최종 판단**: AI는 초안을 만들고, 발송과 결정은 사람이 합니다.

데이터와 보안 원칙

- 최소 수집: 필요한 정보만 받습니다.
- 익명화: 이름, 연락처, 주민등록번호, 결제 정보는 제거한 상태에서 작업합니다.
- 조직 명의 계정 사용

- 사람의 검수 절차를 도구 안에 내장합니다. 예를 들어 답변 초안은 사람이 확인한 뒤에만 발송되도록 만듭니다.
- 종료 시 자료를 반환하거나 파기합니다.

개인정보와 관련된 구체적인 의무는 해당 법령과 조직의 내부 규정을 따라야 하며, 필요하면 법무 담당자나 전문가와 확인합니다.

진행이 어려운 경우

처음부터 미리 말해 두는 편이 서로에게 낫습니다. 다음과 같은 상황이면 이 방식은 맞지 않을 수 있습니다.

- 담당자가 지정되지 않았다.
- 담당자가 1년 안에 이탈할 가능성이 높다.
- 담당자에게 인수인계 책임을 기대하기 어렵다.
- 담당자에게 의사결정 권한이 없다.
- 방문일에 담당자가 동석할 수 없다.
- 방문일 사이의 6일 동안 작업할 시간이 없다.
- 사전 자료를 제공할 수 없다.
- 업무 절차 자체가 정의되어 있지 않다.
- 개발 범위가 계속 바뀐다.
- 현장 접근이 제한된다.

이런 경우에는 1시간 상담에서 그대로 말하고, 절차 정리나 교육이 먼저 필요하다면 그 방향을 제안합니다. 모든 조직에 실행형이 맞는 것은 아니며, 맞지 않다는 판단도 상담의 결과입니다.

실습

실습 1. 우리 조직의 AX 포인트 후보 찾기

아래 프롬프트에 업무 설명을 붙여 넣고 AX 포인트 후보를 뽑아 봅니다. 고객 이름과 연락처 같은 개인정보는 빼고 작성합니다. 결과는 확정이 아니라 후보 목록이므로, 목록에서 "AI로 안 될 것 같다"는 이유로 미리 빼지 않습니다.

프롬프트

> 아래는 우리 조직의 한 부서가 하는 업무 설명입니다. 개인정보는 제거했습니다.
> 이 업무에서 AX 포인트(AI나 자동화로 개선할 여지가 큰 지점) 후보를 찾아 주세요.
>
> 다음 다섯 가지 신호를 기준으로 찾습니다.
> 1. 같은 내용을 두 번 이상 입력하는 일
> 2. 사람마다 결과가 달라지는 일
> 3. 찾는 데 시간이 걸리는 자료
> 4. 하고 나면 또 해야 하는 일
> 5. "왜 이렇게 하지?" 싶은 일
>
> 각 후보에는 아래 세 질문도 적용해 주세요. 셋 다 "예"이면 1순위 후보입니다.
> (1) 이 흐름에서 가장 오래 걸리는 단계인가
> (2) 그 단계가 매번 같은 방식으로 반복되는가
> (3) 그 단계의 입력과 출력이 정해져 있는가
>
> 결과는 표로 정리해 주세요. 열은 [후보 / 해당 신호 번호 / 세 질문 결과 / 개선 방향 / 불편도(상중하) / 확인할 질문]입니다.
> "AI로 어려울 것 같다"는 이유로 후보를 빼지 말고, 어려운 이유는 확인할 질문 열에 적어 주세요.
> 마지막에 세 질문 결과, 불편도, 소요 시간을 기준으로 우선순위 3개를 제안하고, 제안의 근거는 제가 준 설명에서만 가져와 주세요. 설명에 없는 내용은 추측하지 말고 질문으로 남겨 주세요.
>
> [업무 설명 붙여 넣기]

실습 2. 담당자 후보 점검

7장의 직원 성장 4단계 중 3단계(업무를 개선하는)와 4단계(도구를 직접 만들어 쓰는)에 가까운 사람이 후보로 떠오르는지 함께 봅니다. 사내에서 AX를 추진할 담당자를 떠올리고, 22장의 선정 기준에 비추어 점검합니다.

프롬프트

> 우리 조직에서 AX 담당자 후보를 점검하려 합니다. 후보의 이름은 쓰지 않고 역할만 적겠습니다.
> 다음 여섯 항목을 우선순위 순서대로 각각 상, 중, 하로 평가하는 표를 만들어 주세요.
> 1. 경영진과의 신뢰관계
> 2. 1년 이상 근속 가능성
> 3. 퇴사 시 인수인계에 대한 책임감
> 4. 의사결정 권한
> 5. 업무 이해도
> 6. 투입 가능한 시간(방문일 동석과 방문일 사이 작업)
> 각 항목에서 제가 확인해야 할 질문을 하나씩 같이 적어 주세요.
> 후보 설명: [후보의 역할과 근속 기간, 맡고 있는 일을 간단히]

정리

- 실행형 AX 컨설팅은 도구를 직접 만들어 전달하고, 그것을 운영할 사람까지 세우는 방식입니다. 끝났을 때 작동하는 도구와 담당자가 남습니다.
- DFY로 시작해 DWY를 거쳐 DIY로 끝나도록 설계하고, 이관은 마지막에 몰아서 하지 않고 매주 합니다.
- 제안서 기준 예시 모델은 3개월 동안 주 1회 6시간입니다. 매 방문은 점검, 진단, 개발, 초안 확인, 과제 인계의 다섯 단계를 반복하고, 방문 사이의 6일은 담당자가 직접 해 보는 시간입니다.
- 완료는 담당자가 혼자 실행하고, 수정하고, 자산이 사내에 남아 있을 때입니다. 종료 조건은 기간 만료가 아닌 인수인계 완료입니다.
- 담당자가 없거나 방문일에 동석할 수 없는 등 진행이 어려운 경우에는 처음에 그대로 말하고, 절차 정리나 교육을 먼저 권합니다.

22장

사내 AX 담당자와 첫 3개월

AX를 조직에 심는 일은 결국 사람이 합니다. 도구와 문서는 만들 수 있지만, 그것을 매주 돌리고 고치고 다음 사람에게 넘기는 일은 사내 담당자의 몫입니다. 이 장은 착수 전에 준비할 네 가지, 담당자를 고르는 기준, 문제 목록과 워크플로우를 쓰는 법, 데이터와 보안 원칙, 인수인계 점검표, 그리고 끝난 뒤의 선택지를 설명합니다. 외부 컨설팅을 받는 경우에도, 사내에서 직접 시작하는 경우에도 같은 순서로 쓸 수 있습니다.

착수 전에 준비할 네 가지

21장에서 소개한 실행형 AX는 네 가지가 준비되어 있을 때 첫 방문부터 속도가 납니다.

1. **AX 담당자의 좌석과 워크스테이션**: 작업이 끝난 뒤에도 상시 쓰는 자리여야 합니다.
2. **사내 AX 담당자 선정**: 성패를 가르는 가장 중요한 항목입니다.
3. **AI 계정의 조직 명의 생성과 결제**: 종료 뒤에 계정과 작업 이력이 사내에 남도록 처음부터 조직 명의로 만듭니다. 개인 계정에서 시작해 나중에 옮기려 하면 이력이 따라오지 않을 수 있습니다.
4. **사전 자료**: 문제 목록과 워크플로우.

요금제의 종류와 한도, 가격은 자주 바뀌므로 이 책에서는 다루지 않습니다. 계정을 만들 때는 공식 요금제 페이지(https://claude.com/pricing)에서 현재 조건을 확인하고(2026년 11월 기준), 선택 기준은 2장을 참고합니다.

담당자 자리와 장비

장비는 사양이 높을수록 좋은 것이 아니라 어디에 투자해야 효과가 큰지 아는 것이 중요합니다. 아래는 2026년 11월 기준 권장이며, 제품과 사양은 계속 바뀌므로 구매 전에 현재 시점의 기준을 다시 확인합니다.

투자 우선순위는 다음 순서를 권합니다.

1. 저장장치(NVMe 방식 SSD): 파일 탐색과 작업 폴더 처리 속도에 직접 영향을 줍니다.

2. 화면 면적: 듀얼 모니터 이상. 대화창, 결과물, 참고 문서를 동시에 띄워 놓는 일이 많습니다.
3. 네트워크 안정성: 유선 연결을 권합니다.
4. 메모리: 여유 있는 용량. 브라우저 탭, 편집기, 도구를 동시에 열어 둡니다.
5. CPU: 최근 세대의 일반 사양이면 충분합니다.
6. 그래픽카드: 우선순위가 가장 낮습니다.

흔한 오해

- **외장 그래픽카드가 필요하다는 오해**: Claude 같은 서비스의 추론은 서비스 쪽 서버에서 일어납니다. 내 컴퓨터에서 모델을 직접 돌리는 경우가 아니라면 그래픽카드의 영향은 작습니다.
- **인터넷은 빠르기만 하면 된다는 오해**: 속도보다 끊기지 않는 안정성이 중요합니다. 연결이 끊기면 진행 중이던 작업이 중단되거나 사라질 수 있습니다.
- **장비만 좋으면 된다는 오해**: 장비가 좋아도 계정의 사용 한도가 낮으면 작업이 멈춥니다. 여러 세션을 동시에 돌리는 작업(10장)은 사용량이 많아지므로 요금제 선택과 함께 준비합니다.

챔피언이라는 역할

챔피언은 직책이 아니라 역할입니다. 조직 안에서 변화를 실제로 굴리는 사람이며, 다음 세 가지를 갖춥니다.

- 업무를 매우 빠르고 쉽게 처리합니다.
- 더 빠르고 쉽게 처리하는 시스템을 구축하고 개선합니다.
- 자동화 프로세스를 구축해 본인 외 다른 직원도 쓸 수 있게 합니다.

회사에 이런 사람이 한 명만 있어도 회사가 바뀔 수 있습니다. 이 책에서 말하는 AX 담당자는 이 챔피언 역할을 맡은 사람입니다. AX를 이루는 세 조각은 워크플로우(무엇을 바꾸는가), SSOT(무엇을 근거로 하는가), 챔피언(누가 바꾸는가)이며, 앞의 둘은 7장에서 다뤘습니다.

담당자를 고르는 기준

사내 AX 담당자의 선정은 컨설팅의 성패를 가르는 가장 큰 변수입니다. 흔히 AI를 잘 아는 사람을 앞세우려 하지만, 우선순위는 다음과 같습니다.

1. **경영진과의 신뢰관계**: 현장의 문제를 솔직하게 말하고 결정을 이끌어 낼 수 있어야 합니다.

2. **최소 1년 이상 근속의 확실성**: 오래 남을 사람이어야 지식이 조직에 남습니다.
3. **퇴사 시 인수인계에 대한 책임감**: 떠나더라도 다음 사람에게 넘길 사람이어야 합니다.
4. **의사결정 권한**: 업무 방식을 바꾸는 결정을 내리거나 직접 올릴 수 있어야 합니다.
5. **업무 이해도**: 어떤 일이 어떻게 흘러가는지 알아야 합니다.
6. **투입 가능한 시간**: 방문일에 동석하고, 방문일 사이에 실제 작업을 할 수 있어야 합니다.

AI나 IT에 대한 이해도는 우선순위가 가장 낮습니다. AI 기술은 배우면 되지만 신뢰와 책임감은 가르칠 수 없기 때문입니다. "AI를 잘 아는 사람보다 오래 남을 사람이 낫다"는 원칙으로 정리할 수 있습니다.

> **핵심** 담당자를 고를 때는 AI 실력보다 신뢰, 근속, 인수인계 책임이 먼저입니다.

담당자의 현재 위치 확인

담당자 후보가 직원 성장 4단계 중 어디에 있는지 한 줄로 확인합니다. 1단계는 AI를 배우는 직원, 2단계는 AI를 업무에 활용하는 직원, 3단계는 AI로 업무를 개선하는 직원, 4단계는 AI 업무 도구를 직접 만들어 쓰는 직원입니다. 챔피언 역할에는 3단계 이상이 필요하고, 후보가 그 아래라면 어떤 과제로 올릴지를 함께 정합니다.

담당자를 지원하는 법

담당자 혼자서는 지속하기 어렵습니다. 담당 업무에서 일정 시간을 빼 주고, 방문일과 작업일을 일정에 반영하고, 경영진이 그 일의 중요성을 공개적으로 밝혀야 합니다. AX가 담당자의 부수적인 일이 되면 방문일 사이의 6일 동안 작업이 이루어지지 않고, 그러면 사이클이 돌지 않습니다.

사전 자료 쓰는 법

사전 자료는 두 가지입니다. 문제 목록과 워크플로우입니다.

문제 목록 쓰기

전 부서, 전 업무에서 불편한 점을 많이 모을수록 좋습니다. 형식은 자유입니다. 메모 앱이든 표든 문장이든 상관없습니다. 순서만 불편도와 소요 시간이 큰 것부터 놓으면 됩니다. 아래 다섯 가지 신호로 찾으면 쉽습니다.

- 같은 내용을 두 번 이상 입력하는 일
- 사람마다 결과가 달라지는 일

- 찾는 데 시간이 걸리는 자료
- 하고 나면 또 해야 하는 일
- "이거 왜 이렇게 하지?" 싶은 일

목록이 3개면 3개만 쓰고, 30개면 30개를 모두 씁니다. 항목이 많으면 우선순위가 높은 것부터 동시에 진행합니다.

문제 하나를 쓸 때는 다음 여섯 가지를 적으면 충분합니다.

항목	예시(가상의 예)
어떤 일인가	방문 문의에 대한 답변 작성
얼마나 자주	하루에 여러 번
얼마나 걸리나	한 번에 10분 안팎
누가 하나	접수 담당자
지금 방식	지난 답변을 찾아 복사해 고쳐 씀
불편한 점	담당자마다 답이 다름

워크플로우와 정보 원본 쓰기

워크플로우는 업무가 흘러가는 순서를 말합니다. 여기서 핵심은 정보 원본(SOT, Source of Truth)입니다. 정보가 처음 기록되는 지점이고, 이후 단계는 이 원본을 가져다 쓰는 것이 이상적입니다.

정보 원본은 하나로 정하고, 나머지 파일은 참고용으로 표시합니다. 원본이 없는 증상과 만드는 3단계는 7장에 있습니다. 단계는 원본을 정하고, 나머지에 참고 표시를 하고, 바뀌면 원본만 고치는 것입니다.

예를 들어 종이 신청서를 받아 엑셀에 옮기고, 결제 프로그램에 다시 입력하고, 월간 보고서에 또 옮겨 적는다면 같은 정보를 네 번 입력하는 것입니다(가상의 예). 이렇게 같은 정보가 반복 입력되는 지점이 곧 자동화 지점입니다. 업무 순서를 화살표로 이어 그리고, 각 단계에서 어떤 정보가 어디에 입력되는지 표시하면 됩니다.

미리 걸러내지 않습니다

가장 중요한 원칙은 쓰는 사람이 미리 판단하지 않는 것입니다. "늘 이렇게 해 왔으니 당연하다"고 여기는 일과 "AI로는 안 될 것"이라며 미리 뺀 일에 가장 큰 개선 여지가 있는 경우가 많습니다. 목록에 없는 문제는 존재하지 않는 문제가 됩니다. 되는지 안 되는지는 쓰는 사람이 아니라 함께 살펴보며 판단합니다.

첫 3개월을 보내는 방법

예시 모델(21장)의 12회 사이클을 담당자의 입장에서 보면 흐름은 이렇습니다. 방문일에는 컨설턴트와 함께 한 화면을 보며 문제를 파악하고 초안을 확인합니다. 방문 사이의 6일에는 받은 과제를 혼자 해 봅니다. 막힌 지점은 메모해 두었다가 다음 방문에서 묻습니다.

담당자가 지킬 습관

- **막힌 지점을 숨기지 않습니다.** 막힌 곳이 곧 다음 교육 주제입니다.
- **과제는 완벽하지 않아도 해 봅니다.** 해 보지 않으면 어디서 막히는지 모릅니다.
- **만든 것은 기록합니다.** 어떤 프롬프트를, 어떤 자료로, 어떻게 썼는지 남기면 자산이 됩니다.
- **한 번에 하나씩 넘깁니다.** 새로운 도구를 만들 때마다 사용자에게 한 가지만 가르칩니다.
- **AI의 결과는 사람이 검토합니다.** 19장의 자가검증을 쓰되, 외부로 나가는 결과는 사람이 마지막으로 확인합니다.

데이터와 보안 원칙

AX에서는 업무 자료가 AI 서비스로 전달됩니다. 아래 원칙을 처음부터 정해 둡니다.

1. **최소 수집**: 업무에 꼭 필요한 정보만 받습니다.
2. **익명화**: 이름, 연락처, 주민등록번호, 결제 정보를 제거한 뒤 작업합니다. 예시로 쓸 때도 마찬가지입니다.
3. **조직 명의 계정 사용**: 개인 계정에 업무 자료를 올리지 않습니다.
4. **검수 절차 내장**: 고객에게 나가는 답변 초안은 사람이 확인한 뒤에 발송되도록 도구에 단계를 넣습니다.
5. **종료 시 자료 반환과 파기**: 외부 컨설턴트가 받은 자료는 종료 때 돌려받거나 파기합니다.
6. **비밀 값 관리**: API 키와 비밀번호는 코드나 대화에 쓰지 않습니다(20장).

개인정보 보호에 관한 법적 의무는 해당 법령과 조직의 내부 정책이 우선합니다. 어떤 자료를 AI 서비스에 올려도 되는지는 서비스의 개인정보 처리방침(https://www.anthropic.com/legal/privacy)과 조직이 가입한 상품의 약관을 함께 확인합니다. 상품 종류에 따라 적용되는 문서가 다를 수 있습니다(2026년 11월 기준). 필요하면 법무 담당자나 개인정보 보호 담당자와 상의합니다.

인수인계 점검표

종료 조건은 기간이 아니라 인수인계 완료입니다. 다음 항목을 하나씩 확인하며 마무리합니다.

담당자의 역량

- 각 솔루션을 담당자 혼자 처음부터 끝까지 실행할 수 있다.
- 새 요구가 생겼을 때 프롬프트나 문서를 수정할 수 있다.
- 결과가 이상할 때 어디를 확인해야 하는지 안다.

자산의 위치

- AI 계정이 조직 명의이고, 관리자 권한이 사내에 있다.
- 프롬프트, 문서, 템플릿이 사내 저장소에 정리되어 있다.
- 비밀 값과 연동 설정의 위치가 문서로 남아 있다.
- 컨설턴트의 개인 계정이나 개인 저장소에 남은 것이 없다.

운영 문서

- 사용 절차서와 주간 점검표가 있다.
- 주의사항과 예외 처리 방법이 적혀 있다.
- 인계 확인서에 담당자가 확인한 항목이 정리되어 있다.

담당자의 후임

담당자가 바뀌어도 이어질 수 있도록 후임이 문서만 보고 시작할 수 있는지 점검합니다. 이를 위해 담당자 본인이 쓴 인수인계 문서를 한 번 다른 사람에게 읽혀 봅니다.

끝난 뒤의 선택지

종료 후에는 네 가지 선택지가 있고, 모두 선택 사항이며 별도 운영입니다.

- **유지보수**: 운영 중 생긴 문제와 환경 변화에 대응하는 수정
- **확장 개발**: 새로 찾은 AX 포인트를 추가로 만드는 개발
- **운영 자문**: 정기적으로 점검하고 방향을 조언
- **담당자 교체 대응**: 담당자가 바뀔 때 새 담당자에게 넘기는 교육

아무것도 선택하지 않아도 3개월의 결과물은 조직에 남아야 합니다. 그것이 완료 기준입니다. 비용이나 계약 조건은 이 책에서 다루지 않고, 상담에서 확인합니다.

실습

실습 1. 문제 목록 작성

우리 팀에서 반복되는 불편을 정리해 봅니다. 아래 프롬프트에 생각나는 대로 적은 메모를 붙여 넣으면 AI가 표 형식으로 정리해 줍니다. 개인정보는 빼고, 걸러내지 말고 모두 적습니다.

프롬프트

> 아래는 우리 팀의 업무 불편에 대한 메모입니다. 생각나는 대로 적은 것이고 정리되어 있지 않습니다.
> 개인정보는 제거했습니다.
> 이 메모를 문제 목록 표로 정리해 주세요.
> 열은 [번호 / 어떤 일인가 / 얼마나 자주 / 한 번에 얼마나 걸리나 / 누가 하나 / 지금 방식 / 불편한 점 / 해당 신호]입니다.
> 해당 신호는 다음 다섯 가지 중에서 고릅니다: 1 같은 내용을 두 번 이상 입력, 2 사람마다 결과가 다름, 3 찾는 데 시간이 걸리는 자료, 4 하고 나면 또 해야 하는 일, 5 "왜 이렇게 하지?" 싶은 일.
> 규칙은 다음과 같습니다.
> - 항목을 합치거나 빼지 않습니다. AI로 안 될 것 같아도 목록에서 제외하지 않습니다.
> - 메모에 없는 정보(빈도, 소요 시간 등)는 추측하지 말고 "미확인"으로 적습니다.
> - 표 아래에 "미확인 항목에 대해 제가 확인해야 할 질문"을 모아 적습니다.
> - 마지막에 같은 정보를 여러 번 입력하는 것으로 보이는 지점이 있으면 별도로 표시합니다.
>
> [메모 붙여 넣기]

실습 2. 워크플로우 그려 보기

문제 하나를 골라 그 일이 흘러가는 순서와 정보 입력 지점을 정리합니다.

정리한 뒤에는 같은 일을 세 가지 상태로 나란히 그려 비교합니다. 지금 방식, AI 활용(한 단계만 AI로 바꿈), AX(단계가 사라지고 순서가 바뀜)입니다. 걸리는 시간은 실측이 아니면 "미확인"으로 두고, 대략의 차이만 적습니다. 상태의 정의는 7장을 참고합니다.

프롬프트

> 아래는 우리 팀의 한 업무가 진행되는 순서입니다.
> 각 단계를 번호로 정리하고, 단계마다 (1) 누가 하는지 (2) 어떤 정보가 어디에 입력되는지 (3) 걸리는 시간(미확인이면 미확인)을 표로 만들어 주세요.
> 같은 정보가 두 번 이상 입력되는 지점은 굵게 표시하고, 정보가 처음 기록되는 지점(원본)이 어디인지 제안해 주세요.
> 순서나 시간을 추측해서 채우지 말고, 설명에서 불분명한 부분은 질문으로 남겨 주세요.
>
> [업무 진행 순서 설명]

정리

- 착수 전에 담당자의 자리와 장비, 담당자 선정, 조직 명의 AI 계정, 사전 자료(문제 목록과 워크플로우)의 네 가지를 준비합니다.
- 장비는 저장장치, 화면 면적, 네트워크 안정성 순으로 투자합니다. 외장 그래픽카드는 필요하지 않다는 것이 2026년 11월 기준 권장입니다.
- 담당자는 신뢰, 근속, 인수인계 책임, 권한, 업무 이해, 투입 시간 순으로 고르고, AI 실력은 가장 뒤입니다.
- 문제 목록은 미리 걸러내지 않고 모두 씁니다. 같은 정보가 반복 입력되는 지점이 곧 자동화 지점입니다.
- 종료는 담당자의 실행과 수정 능력, 사내에 남은 자산, 운영 문서를 점검표로 확인한 인수인계 완료입니다. 이후 유지보수, 확장, 자문, 교체 대응은 선택입니다.

부록

부록 A · 프롬프트 카드 12종

부록 B · 스킬 14종

부록 C · 용어 사전

부록 D · 체크리스트

부록 A

프롬프트 카드 12종

이 부록은 액티전스 프롬프트 라이브러리의 카드 12종을 한 장씩 정리한 안내서입니다. 각 카드를 언제 쓰는지, 무엇을 준비해야 하는지, 무엇이 나오는지를 적었습니다. 카드 본문은 길고 자주 고쳐지므로 이 책에 싣지 않습니다. 복사해서 쓸 최신본은 actigence.ai/vibe-coding-prompts에서 받습니다.

카드 쓰는 법

카드는 Claude Code에 붙여 넣는 긴 지시문입니다. 본문은 영어로 쓰여 있습니다. 지시를 정확하게 전달하기 위한 선택이며, 읽는 사람이 이해해야 할 내용은 이 부록과 카드 설명란에 한국어로 있습니다. 결과 보고는 카드가 한국어로 하도록 지시하고 있어, 실행하면 한국어 보고를 받습니다.

사용 순서는 모든 카드에서 같습니다.

1. 작업 폴더에서 Claude Code를 엽니다(9장).
2. 카드 본문 전체를 복사해 첫 메시지에 붙여 넣습니다.
3. 카드 바로 아래에 이번 요청을 한두 문장으로 덧붙입니다. 카드의 "사용 예"가 그 문장의 견본입니다.
4. 카드가 요구하는 첨부물이 있으면 함께 둡니다. 예를 들어 PRD(제품 요구사항 문서) 파일이 필요한 카드가 있습니다.
5. 결과를 받으면 19장의 검수 절차로 확인합니다.

주의할 점이 세 가지 있습니다.

- 카드는 통째로 붙여 넣습니다. 일부만 잘라 쓰면 안전장치와 검증 단계가 빠집니다.
- 카드마다 자율 실행의 범위가 다릅니다. 배포까지 확인 없이 진행하는 카드가 있으니, 실행 전에 카드 설명과 이 부록의 "주의"를 읽습니다. 배포 단계가 있는 카드는 연결된 호스팅 계정에 실제로 올라간다는 뜻입니다(20장).
- 카드 안의 모델 이름과 세부 절차는 시간이 지나며 바뀝니다. 이 부록은 카드의 목적과 입출력만 다루므로, 세부 변경과 관계없이 유효합니다.

카드는 크게 네 묶음입니다. 새 프로젝트를 시작하는 카드, 기능을 만드는 카드, 품질을 점검하고 고치는 카드, 디자인과 시각화 카드입니다. 아래 순서는 이 묶음을 따릅니다.

프로젝트 시작

원클릭 풀스택 (setup-fullstack-prototype)

- **언제 쓰는가** — 빈 폴더에서 시작해 배포된 시제품까지 한 세션에 끝내고 싶을 때. 화면 전체를 로그인 없이 눌러볼 수 있는 데모 빌드가 목표입니다.
- **준비물** — PRD 파일 한 개. PRD가 없으면 카드는 한 번 멈추고 요청합니다. PRD는 부록 B의 `prd-vibecoding` 스킬로 만들 수 있습니다. 데이터 저장소는 구글 시트가 기본이므로 구글 계정과 클라우드 프로젝트 사용 권한이 필요합니다.
- **산출물** — 배포된 데모, 데이터를 담는 시트, 디자인 기준 문서, 단계별 통과 기록. 카드는 네 개의 관문(데모 기능 초안, 디자인 고도화, 보안 강화, 배포와 검증)을 차례로 통과합니다.
- **주의** — 확인 없이 배포까지 진행합니다. 처음에는 연습용 계정과 폴더에서 돌립니다. 계정과 키가 만들어지므로 결과물에 어떤 키가 남았는지 마지막에 점검합니다(20장).

자동 적용 (setup-seo-aeo-geo)

- **언제 쓰는가** — 이미 운영 중인 Next.js 사이트의 검색 노출을 한 번에 손볼 때. 전통 검색 최적화(SEO), AI 답변 엔진 노출(AEO), 생성형 엔진 노출(GEO)을 함께 다룹니다. 카드는 Next.js 사이트를 기준으로 쓰여 있습니다.
- **준비물** — 깃(Git) 저장소로 관리되는 사이트 코드, 변경 사항이 없는 깨끗한 작업 상태. 더러운 상태면 카드가 중단 보고를 냅니다.
- **산출물** — 별도 브랜치에 담긴 변경, 풀 리퀘스트(변경 제안서), 적용 전후 비교 기록. 기존 파일은 직접 고치지 않고 제안 사본으로 냅니다.
- **주의** — 색인 제거 신호(noindex, 다른 주소로의 canonical, 콘텐츠 차단 등)는 증거를 확인한 뒤 경로 단위로만 적용하도록 되어 있습니다. 결과가 나오면 검색 노출이 줄지 않았는지 반드시 직접 확인합니다.

개발 작업

기능 추가: 참고 분석 (dev-merged-3)

- **언제 쓰는가** — 새 기능을 만들기 전에 비슷한 서비스나 오픈소스를 살펴 가져올 것과 하지 않을 것을 추리고 싶을 때.

- **준비물** — 분석할 대상의 주소나 로컬 경로. 카드는 사용자가 준 출처만 분석하고 임의로 찾지 않습니다.
- **산출물** — 채택할 패턴, 버릴 패턴, 라이선스 점검 결과, 우리 프로젝트 구조에 맞게 옮긴 설계.
- **주의** — 코드를 그대로 복사하지 않고 구조와 패턴만 가져오도록 되어 있습니다. 라이선스가 맞지 않으면 그 출처는 제외하고 직접 구현안을 냅니다. 라이선스 판단은 최종적으로 사람이 확인합니다.

기능 추가: 풀스택 (dev-merged-4)

- **언제 쓰는가** — 기능 하나를 데이터 저장, 서버 처리, 화면, 테스트까지 한 번에 넣을 때.
- **준비물** — 기능 설명 한두 문단, 이미 있는 저장소. 새 프로젝트가 아니라 기존 프로젝트용입니다.
- **산출물** — 모든 계층의 변경, 되돌릴 수 있는 데이터 변경 파일, 입력 검증, 핵심 경로를 끝까지 따라가는 테스트 1건 이상.
- **주의** — 데이터 구조를 바꾸는 작업은 되돌리는 절차까지 실제로 실행해 확인하도록 되어 있습니다. 운영 데이터에 닿는 변경은 사전에 백업 여부를 확인합니다. 기능이 여러 개면 하나씩 따로 요청합니다.

기능 추가: 화면 (dev-6)

- **언제 쓰는가** — 화면과 컴포넌트, 상태 연결만 만들 때. 백엔드는 가짜 데이터로 두어도 됩니다.
- **준비물** — 만들 화면 설명, 기존 디자인 기준(있다면). 기준이 없으면 카드가 기존 코드에서 추론하고 가정을 기록합니다.
- **산출물** — 로딩, 빈 상태, 오류 세 가지 상태를 갖춘 화면, 접근성 점검, 화면 캡처를 근거로 한 독립 디자인 검토 결과.
- **주의** — 캡처로 확인된 것만 완료로 칩니다. 코드만 고치고 화면을 보지 않았다면 완료가 아닙니다.

품질과 디버깅

디버깅 (dev-merged-5)

- **언제 쓰는가** — 버그를 만났을 때. 가끔만 발생하거나 재현이 어려운 경우, 데이터가 이상한 경우도 포함합니다.
- **준비물** — 증상, 발생 조건, 오류 메시지나 화면 캡처. 가능하면 처음 발생한 시점.
- **산출물** — 근거로 확인된 원인, 수정, 수정 전에는 실패하고 후에는 통과하는 회귀 테스트, 재현 스크립트.

- **주의** — 원인이 증거로 확인되기 전에는 코드를 고치지 않는 것이 핵심 규칙입니다. 데이터가 틀린 경우 영향 범위를 실제 조회로 센 뒤 고치며, 수정 전에 되돌릴 지점을 만듭니다.

품질 파이프라인 (dev-quality-pipeline)

- **언제 쓰는가** — 기능을 만든 뒤 안정화하거나 오래된 코드를 정리할 때. 테스트 보강, 코드 리뷰, 정리를 이 순서로 진행합니다.
- **준비물** — 대상 범위(변경분, 모듈, 저장소 전체 중 하나). 테스트와 검사 명령은 카드가 저장소에서 찾습니다.
- **산출물** — 보강된 테스트, 리뷰 결과, 정리된 코드, 각 단계의 실행 결과.
- **주의** — 테스트 없는 코드는 정리하지 않고, 리뷰 전의 코드는 다듬지 않는다는 순서 규칙이 있습니다. 새 기능 추가는 이 카드의 일이 아닙니다.

추적 설계 (dev-tracking-vercel-ga4)

- **언제 쓰는가** — 사이트 운영을 시작하거나, 전환(가입, 신청, 결제)을 측정하지 못하는 기존 사이트에 추적을 붙일 때.
- **준비물** — 사이트 코드, 측정할 행동 목록, 분석 도구 계정. 호스팅이 Vercel이고 분석 도구가 GA4인 구성이 가장 자세히 정의되어 있고, 다른 구성은 같은 기능을 다른 방식으로 구현합니다.
- **산출물** — 이벤트 설계, 구현 코드, 중복 집계 점검, 데이터 품질 점검 방법.
- **주의** — 동의 관리와 개인정보 규정(한국 개인정보 보호법 등)이 설계에 포함됩니다. 어떤 정보를 수집하는지는 반드시 사람이 검토하고, 방침 문서와 일치시킵니다(13장, 부록 D).

디자인과 시각화

디자인 변경 (dev-13)

- **언제 쓰는가** — 기능 구현이 끝난 뒤 화면 하나나 컴포넌트 하나의 모양을 다듬을 때. 큰 구조 변경 없이 손질하는 용도입니다.
- **준비물** — 대상 화면, 기존 디자인 토큰(색과 글자 규칙을 모아둔 정의). 범위를 밝히지 않으면 카드가 가장 좁은 범위로 가정합니다.
- **산출물** — 범위 안의 변경, 변경 전후 캡처, 범위 밖에서 발견한 문제의 후속 목록.
- **주의** — 범위 밖 파일은 눈에 띄는 결함이 있어도 고치지 않고 목록에만 올립니다. 전면 개편은 이 카드가 아니라 아래 재설계 카드입니다.

디자인 감사와 개선 (dev-14)

- **언제 쓰는가** — 프로젝트 전체의 사용성을 점검하고 싶을 때. 검사 담당 다섯 갈래(현황, 일관성, 접근성, 상호작용, 흐름)가 동시에 점검합니다.
- **준비물** — 실행 가능한 프로젝트. 어떤 항목을 고칠지는 감사 보고서를 본 뒤 사용자가 정합니다.
- **산출물** — 종합 감사 보고서, 우선순위 선택지, 선택한 항목만 실행한 변경.
- **주의** — 우선순위 결정은 사람의 몫입니다. 카드는 감사 결과를 임의로 전부 고치지 않습니다. 이미 결정이 있다면 그 출처를 밝히고 진행합니다.

디자인 전면 재설계 (dev-design-overhaul)

- **언제 쓰는가** — Claude Code로 만든 프로젝트의 디자인 전체를 하나의 토큰 체계로 다시 짤 때. 기능과 데이터 흐름은 그대로 두고 레이아웃과 화면 구성까지 바꿉니다.
- **준비물** — 프로젝트, 그리고 원하는 분위기·참고 자료·제약이 있으면 지정 칸에 적습니다. 지정하면 최우선으로 반영됩니다.
- **산출물** — 방향 선언과 기각한 대안, 통합 토큰, 화면별 적용 결과, 기능 회귀 점검, 렌더 캡처.
- **주의** — 기존 기능이 그대로 동작하는지 변경마다 확인하도록 되어 있습니다. 시작 전에 저장소를 커밋해 두고, 결과가 마음에 들지 않으면 되돌릴 수 있게 합니다(9장).

발표자료 HTML (dev-slides-html)

- **언제 쓰는가** — 기획안이나 보고서 문서 하나로 발표 자료를 만들 때. 결과는 파일 하나의 HTML입니다.
- **준비물** — 원문 문서(.md 등). 이미지 생성 도구(12장)가 연결되어 있으면 장면 이미지를 생성해 넣습니다.
- **산출물** — 슬라이드 구성표, 생성 이미지, 도식, 한국어 슬라이드 전체가 담긴 단일 HTML.
- **주의** — 생성 이미지 안에 글자가 들어가지 않도록 장면을 설계하고, 삽입 전에 눈으로 확인하는 절차가 있습니다. 이미지 생성에 실패하면 빈칸을 채우지 않고 어느 슬라이드인지 보고합니다. 발표 전에 숫자와 문장을 원문과 대조합니다.

카드 고르는 법

상황별로 가장 먼저 볼 카드는 아래와 같습니다.

상황	카드
빈 폴더에서 시제품까지	원클릭 풀스택
운영 중인 사이트의 검색 노출	자동 적용
만들기 전에 참고 사례 정리	기능 추가: 참고 분석
저장소에 기능 하나 추가	기능 추가: 풀스택
화면만 추가	기능 추가: 화면
버그	디버깅
기능 완성 후 안정화	품질 파이프라인
전환 측정	추적 설계
화면 하나 손질 / 전체 점검 / 전면 개편	디자인 변경 / 디자인 감사와 개선 / 디자인 전면 재설계
문서를 발표자료로	발표자료 HTML

카드 목록과 설명은 actigence.ai/vibe-coding-prompts에서 계속 갱신됩니다. 카드의 개념 배경은 actigence.ai/manual의 사용설명서와 부록 C의 용어 사전을 참고합니다.

이 책 본문의 복사용 프롬프트

카드와 별개로, 각 장의 본문과 실습에는 바로 복사해 쓰는 짧은 프롬프트가 있습니다. 상황별로 찾을 곳을 모았습니다. 대괄호 안은 본인 내용으로 바꿉니다.

상황	위치	요지
되묻게 하기, 떠넘기기 대응 3단 재촉	3장	객관식으로 한 번에 하나씩 묻게 하고, 방법 안내만 하면 직접 하라고 세 번 밀어붙입니다
한 줄 요청, 요청문 만들기, 모아서 고치기	5장	번호 붙인 질문으로 요청문을 받고, 한 줄 결과와 비교하고, 고칠 점을 한 번에 보냅니다
PRD 다섯 칸, 열 줄 요청문, 배포	6장	작업 폴더에서 PRD를 만들고 인포그래픽으로 검수한 뒤 요청문으로 줄여 파일을 만들고 주소를 붙입니다
병목 찾기 3질문	7장	가장 오래 걸리는 단계, 매번 같은 방식인지, 입력과 출력이 정해져 있는지 묻습니다

상황	위치	요지
규모 큰 검토 맡기기	10장	문서 여러 개의 충돌 조항을 나눠서 전수 대조시키고 재검증까지 요청합니다
하네스 세팅, 계획서, 조사 의뢰	16장	규칙·진행·이유 파일을 만들고, 계획을 파일로 받아 확정한 뒤 조사를 맡깁니다
양식 규칙 추출, 아트보드	16장	양식 파일에서 규칙을 뽑고, 쪽 모양으로 펼친 확인 화면에서 번호로 고칩니다
기록목록, 계획 대비 실적표	17장	자료를 훑어 번호를 붙이고, 실적표를 엑셀 수식으로 만듭니다
원인 묻기, 성과 절, 문제점 절	17장	빈 원인은 사람에게 묻고, 숫자부터 쓰며, 미달은 순화하지 않습니다
새 세션 검토	17장	처음 받은 결재권자의 눈으로 문서를 고치지 않고 검토만 시킵니다

부록 B

스킬 14종

이 부록은 액티전스 스킬 카탈로그의 14종을 정리한 안내서입니다. 스킬마다 하는 일, 호출하는 말, 넣어야 할 입력, 나오는 결과, 주의할 점을 적었습니다. 스킬의 개념과 만드는 법은 11장에서 다룹니다. 파일은 actigence.ai/skills에서 내려받습니다.

설치와 사용의 기본

스킬은 `SKILL.md` 라는 파일에 이름과 설명, 지시문을 담은 작업 묶음입니다. Claude는 설명에 적힌 조건과 사용자의 말이 맞으면 해당 스킬을 불러 씁니다.

- **내려받기** — actigence.ai/skills에서 스킬 파일을 받습니다. 파일 하나로 된 스킬은 `.md`, 참고 문서가 딸린 스킬은 `.zip` 입니다. 이 부록의 14종 중 `oneshot-prototyping`, `fullstack-feature`, `debug-rootcause`, `seo-aeo-geo` 가 zip입니다.
- **Claude Code에서** — `스킬이름/SKILL.md` 형태로 둡니다. 모든 프로젝트에서 쓰려면 `~/.claude/skills/` 아래에, 한 프로젝트에서만 쓰려면 프로젝트 폴더의 `.claude/skills/` 아래에 둡니다. zip 스킬은 압축을 풀어 같은 위치에 넣습니다(2026년 11월 기준).
- **Claude 앱과 웹에서** — 개인 요금제에서는 설정의 스킬 메뉴(Customize > Skills)에서 zip 파일로 올립니다. 코드 실행 기능이 켜져 있어야 합니다. 팀·기업 요금제는 조직 관리자의 설정에 따릅니다(2026년 11월 기준). 메뉴 이름은 바뀔 수 있으므로 공식 도움말을 따릅니다.
- **공식 문서** — Claude Code 스킬 문서, Claude 도움말: 스킬 사용하기(2026년 11월 기준).

사용할 때의 공통 주의점은 다음과 같습니다.

- 스킬은 지시문이지만 파일을 만들고 명령을 실행하는 작업으로 이어질 수 있습니다. 신뢰할 수 있는 출처의 스킬만 설치하고, 함께 들어 있는 스크립트와 참고 파일을 열어 본 뒤 씁니다(11장).
- 호출 조건이 엄격한 스킬이 있습니다. 조건에 필요한 단어가 없으면 스킬이 발동하지 않거나 중단을 알립니다. 아래 "트리거"를 그대로 쓰는 것이 안전합니다.
- 스킬 결과도 검수 대상입니다. 숫자, 인용, 법령 내용은 사람이 확인합니다(19장).
- 스킬 파일은 갱신됩니다. 설치본이 오래되었다면 사이트의 최신본으로 바꿉니다.

프롬프트 작성

GMP — 메타 프롬프트 프로토콜 (gmp)

- **하는 일** — 프롬프트나 스킬 파일을 표준 구조로 설계하거나 개선합니다. 구조는 XML 태그 계층, 처음과 끝에 같은 핵심 규칙을 두는 이중 고정, 우선순위 표시, 자가검증입니다.
- **트리거** — "프롬프트 만들어줘", "프롬프트 작성해줘", "이걸 프롬프트로", "GMP 적용", "메타 프롬프트", "스킬 만들어줘", "SKILL.md 만들어줘".
- **입력** — 만들려는 작업 설명, 또는 고치려는 기존 프롬프트.
- **결과** — 구조를 갖춘 프롬프트 또는 스킬 `.md` 파일.
- **주의** — 프롬프트를 만들어 달라는 요청에만 쓰는 스킬입니다. "보고서 써줘", "코드 짜줘"처럼 작업 자체를 시키는 말에는 맞지 않습니다. 메타 프롬프팅의 원리는 3장을 봅니다.

문서 생성

PRD-Vibecoding — 바이브코딩 PRD 작성 (prd-vibecoding)

- **하는 일** — 한 줄짜리 아이디어에서 제품 요구사항 문서를 합성합니다. 사용자 선호가 결정적인 항목만 객관식으로 묻고, 데이터 모델, 화면, 권한 표는 자동으로 설계합니다.
- **트리거** — "PRD 만들어줘", "PRD 작성해줘", "바이브코딩 PRD", "프로토타이핑 사양서", "클로드 코드용 PRD".
- **입력** — 아이템을 설명하는 한 줄에서 몇 줄. 질문에는 객관식으로 답합니다.
- **결과** — 단일 `PRD.md`. 13개 절과 자율 실행 설정 절을 포함하며, 부록 A의 원클릭 풀스택 카드와 `oneshot-prototyping` 스킬이 그대로 읽는 형식입니다.
- **주의** — 사용자만 아는 사실(대상 고객, 제약 조건)은 지어내지 않고 질문으로 확인합니다. 결과를 읽고 빠진 요구가 없는지 점검한 뒤 개발에 넘깁니다(4장).

Session-Handoff — 세션 인계 문서 (session-handoff)

- **하는 일** — 현재 세션의 확정 산출물, 결정과 근거, 제약, 범위(변경할 것, 유지할 것, 하지 않을 것)를 뽑아 다음 세션이 이어받을 `.md` 문서를 만듭니다.
- **트리거** — "핸드오프 문서 만들어줘", "세션 핸드오프", "다음 세션으로 넘길 문서", "이 작업 다른 대화에서 이어가".

- **입력** — 현재 세션의 대화와 작업 상태. 별도 첨부는 필요하지 않습니다.
- **결과** — 확정된 내용은 원문 그대로 보존하고, 시행착오와 폐기된 경로는 걸러낸 인계 문서.
- **주의** — 단순 요약 요청에는 맞지 않습니다. 이어받을 의도가 있을 때 씁니다. 컨텍스트가 가득 차기 전에 만들어야 내용이 온전합니다(10장, 20장).

Gov-Startup-Bizplan — 사업계획서 작성 (gov-startup-bizplan)

- **하는 일** — 정부지원 창업사업용 사업계획서 `.docx`를 두 번의 패스로 작성합니다. 먼저 사실을 확인하고, 그 뒤에 본문을 씁니다.
- **트리거** — "사업계획서 작성", "정부지원사업", "TIPS 사업계획서", "초기창업패키지", "예비창업패키지".
- **입력** — PRD, 팀 정보, 공고문이나 제출 양식.
- **결과** — 양식에 맞춘 `.docx`. 모든 수치에 기관명과 연도의 출처를 병기하고, 현재 패스에서 쓰지 않는 장은 원본 그대로 둡니다.
- **주의** — 수치는 웹 검색으로 확인한 출처가 있어야 하며, 출처가 없는 수치는 쓰지 않는다는 규칙이 있습니다. 그래도 제출 전에 공고문 요건과 출처를 직접 대조합니다. 제출 책임은 작성자에게 있습니다.

HWPX — 한글 문서 처리 (hwpx)

- **하는 일** — 한글(Hancom Office)의 `.hwpx` 파일을 생성, 읽기, 편집합니다. 내부 XML을 직접 다루는 방식입니다.
- **트리거** — ".hwpx 파일", "한글 문서", "Hancom", "OWPML".
- **입력** — 참고용 `.hwpx` 파일(있으면 구조를 최대한 그대로 복원), 바꿀 텍스트와 데이터.
- **결과** — 표준 형식의 `.hwpx` 파일. 참고 문서가 있으면 서식과 쪽수가 어긋나는지 검사합니다.
- **주의** — 실행 환경에 Python과 XML 처리 라이브러리가 필요합니다. 쪽수와 서식은 한글 프로그램에서 열어 눈으로 확인합니다.

검증과 평가

FactCheck — 7축 팩트체크 (factcheck)

- **하는 일** — 보고서, 사업계획서, 조사 보고서 같은 근거 기반 문서를 일곱 축(사실, 논리, 가정, 수치, 맥락, 법규, 표현)으로 보수적으로 점검합니다. 의심스러우면 표시합니다.
- **트리거** — "팩트체크해줘", "검증해줘", "사실 확인", "이거 맞는지 확인", "제출해도 되는지".

- **입력** — 검증할 문서. 딥리서치 직후의 결과물도 대상입니다(15장).
- **결과** — 대화에는 간결한 요약, 별도 파일에는 수정 흔적 없이 정리된 개선본.
- **주의** — 출처 없는 수치는 자동으로 표시합니다. 핵심 사실은 웹 검색으로 교차 확인하는 방식이므로, 검색이 닿지 않는 내부 자료는 "미확인"으로 남습니다. 미확인 항목은 사람이 확인합니다.

Paper Quality Enhancer — 학술 논문 고도화 (paper-quality-enhancer)

- **하는 일** — 논문 원고를 여덟 축(논증, 문헌, 방법론, 인용, 기여, 명료성, 재현성, 윤리)으로 분석하고, 적대적 심사를 모의해 투고 수준을 끌어올립니다.
- **트리거** — "논문 개선", "논문 고도화", "논문 다듬어줘", "투고 전 검토", "피어리뷰 시뮬레이션", "Reviewer 2".
- **입력** — 원고 파일, 투고하려는 학술지나 학회, 분야.
- **결과** — 개선된 원고와 심사 보고서, 투고 등급 매칭.
- **주의** — 단순 교정, 번역, 요약에는 맞지 않습니다. 문헌 인용은 환각 방지 절차가 있어도 실제 존재 여부를 직접 확인합니다. 연구 윤리와 저작권 책임은 저자에게 있습니다.

시각화

Infographic — 문서 도식화 (infographic)

- **하는 일** — 사업계획서, 보고서, 제안서의 핵심을 15~20개 절의 인포그래픽 HTML로 바꿉니다.
- **트리거** — "인포그래픽 만들어줘", "도식화해줘", "시각화", "문서에 넣을 그림".
- **입력** — 원문 문서. 원문의 목차 순서가 그대로 도식의 순서가 됩니다.
- **결과** — 흰 배경, 검은 글자, 한 가지 포인트 색, 가로형 레이아웃의 HTML.
- **주의** — 수치는 원문 그대로 쓰고 반올림이나 추정을 하지 않습니다. 개인정보는 마스킹합니다. 결과의 숫자를 원문과 한 번 대조합니다.

Gov-Doc-Infographic — 정부지원 도식화 (gov-doc-infographic)

- **하는 일** — 정부지원사업 제출용 사업계획서나 연구개발계획서를 심사위원 친화 형식으로 도식화합니다.
- **트리거** — "사업계획서 인포그래픽", "사업계획서 도식화", "연구개발계획서 도식화", "계획서에 넣을 그림".
- **입력** — 제출용 계획서 원문.

- **결과** — 전체 인포그래픽 HTML 하나와, 문서 초반 개요 표의 칸에 넣는 소형 인포그래픽 HTML 하나.
- **주의** — 마케팅 이미지나 SNS용 그림, 발표 슬라이드에는 맞지 않습니다. 일반 문서는 위의 Infographic을 씁니다. 제출 양식의 분량과 형식 요건은 공고문으로 직접 확인합니다.

Matjip-Scout — 근본 맛집 탐색 (matjip-scout)

- **하는 일** — 지역과 상황(접대, 데이트, 혼밥, 회식, 출장)을 받아 오래되고 검증된 식당만 골라 리포트로 냅니다.
- **트리거** — "맛집 추천해줘", "맛집 찾아줘", "근본 맛집", "출장 맛집", "회식 장소".
- **입력** — 지역, 상황, 인원이나 선호가 있으면 함께.
- **결과** — 추천 목록, 기준 미달로 제외한 후보와 사유, 탐색 방법과 신뢰도 안내가 담긴 `.md`.
- **주의** — 심층 연구(Deep Research) 모드에서만 제대로 동작하며, 일반 모드에서는 모드 전환을 안내하고 멈춥니다. 영업 여부, 휴무일, 메뉴는 방문 전에 식당에 직접 확인합니다. 일상 업무와는 거리가 있으나, 조사형 스킬의 예시로 실습에 쓰기 좋습니다(15장).

개발

원샷 프로토타이핑 (oneshot-prototyping)

- **하는 일** — PRD 하나로 빈 폴더에서 배포까지 한 세션에 진행합니다. 부록 A의 원클릭 풀스택 카드와 같은 절차를 스킬로 묶은 것입니다.
- **트리거** — "프로토타이핑"이라는 단어가 있고, 요청에 PRD가 첨부되었거나 작업 폴더에 PRD 파일이 있을 때 두 조건이 함께 맞아야 발동합니다. 예: "이 PRD로 프로토타이핑 해줘". "프로토타입 만들어줘"만으로는 발동하지 않습니다.
- **입력** — PRD 파일.
- **결과** — 배포된 데모, 구글 시트 저장소, 디자인 기준, 단계별 통과 기록.
- **주의** — 배포와 계정 준비까지 자동으로 진행합니다. 연습 환경에서 먼저 돌립니다(20장). 필요한 연결(디자인 도구 등)이 없으면 같은 기준의 로컬 경로로 진행합니다.

풀스택 기능 구현 (fullstack-feature)

- **하는 일** — 기존 저장소에 기능 하나를 그 기능이 닿는 모든 계층에 넣습니다. 설계, 구현, 검증, QA(품질 확인) 네 역할이 붙습니다.

- **트리거** — "풀스택"과 "구현" 두 단어가 함께 있을 때만 발동합니다. 예: "즐겨찾기 기능 풀스택으로 구현해줘".
- **입력** — 기능 설명이나 요구 문서, 이미 있는 저장소.
- **결과** — 전 계층 변경, 테스트, 관문별 결과표.
- **주의** — 기능 하나 단위입니다. 여러 개면 나누어 요청합니다. 새 프로젝트 시작이나 버그 수정에는 맞지 않습니다. 부록 A의 "기능 추가: 풀스택" 카드와 같은 계열입니다.

디버깅 (debug-rootcause)

- **하는 일** — 증상에서 근거로 확인된 원인까지 추적해 고치고 회귀 테스트로 마감합니다. 가설 세 개를 동시에 세워 반증하며, 재현 시나리오를 잡기 전에는 코드를 고치지 않습니다.
- **트리거** — 메시지에 "디버깅"이라는 단어가 있을 때만 발동합니다. 예: "로그인 빈 화면 디버깅 해줘". "버그 고쳐줘", "에러 나는데"만으로는 발동하지 않습니다.
- **입력** — 증상 설명, 오류 로그, 화면 캡처.
- **결과** — 원인 규명, 수정, 회귀 테스트.
- **주의** — 같은 문제를 다시 잡는 테스트까지 남기는 것이 완료 조건입니다. 운영 데이터가 얽힌 경우 수정 전 백업 여부를 확인합니다(20장).

SEO · AEO · GEO 적용 (seo-aeo-geo)

- **하는 일** — 운영 중인 사이트에 검색 노출 최적화 세 가지를 별도 브랜치에서 적용하고 풀 리퀘스트로 마감합니다.
- **트리거** — "SEO"라는 단어가 있을 때만 발동합니다. 예: "이 사이트에 SEO 적용해줘". "검색 노출 올려줘" 같은 표현만으로는 발동하지 않습니다.
- **입력** — 깃으로 관리되는 사이트 코드, 깨끗한 작업 상태.
- **결과** — 브랜치와 풀 리퀘스트(본문이 최종 보고서), 문서 세트, 생성만 된 데이터베이스 변경 파일.
- **주의** — 새 프로젝트를 만드는 작업에는 맞지 않습니다. 기존 파일은 직접 고치지 않고 제안 사본으로 내며, 적용 전후의 크롤 도달성과 성능 지표를 비교합니다. 데이터베이스 변경은 생성만 하고 적용은 사람이 합니다.

스킬 고르는 법

하려는 일	스킬
프롬프트, 스킬 파일 만들기	gmp
아이디어를 개발 사양서로	prd-vibecoding
작업을 다음 세션에 넘기기	session-handoff
제출 전 문서 검증	factcheck
문서를 그림으로	infographic / gov-doc-infographic
한글 파일 다루기	hwpx
PRD로 시제품 배포까지	oneshot-prototyping
기존 저장소에 기능 하나	fullstack-feature
버그 원인 추적	debug-rootcause
운영 사이트 검색 노출	seo-aeo-geo

최신 스킬 목록과 설명은 actigence.ai/skills에서 확인합니다.

부록 C

용어 사전

이 책과 액티전스 도구에서 자주 마주치는 용어를 정리했습니다. 한글 용어는 가나다순, 영문 용어는 알파벳순입니다. 괄호 안은 용어를 다룬 장입니다. 다시 찾아볼 때 쓰는 용도이며, 더 많은 용어는 actigence.ai/glossary에 있습니다.

한글 용어

- **AX와 DX** — DX(Digital Transformation)는 종이에서 디지털로, AX는 디지털에서 AI로 일하는 방식을 바꾸는 일입니다. 파일을 줄여 데이터로 관리하는 쪽으로 나아갑니다(7장).
- **계획 깊이** — 작업 크기에 맞춰 계획에 들이는 무게를 정하는 기준입니다. No plan(바로 실행), Local plan(계획 모드로 계획을 보고 승인), Two-pass(계획을 세운 뒤 한 번 더 검토) 순으로 무거워집니다(9장, 19장).
- **계획 대비 실적표** — 지표마다 계획, 실적, 차이, 판정, 출처, 원인을 나란히 둔 표입니다. 결과보고서의 중심이며 비율은 수식으로 남깁니다(17장).
- **계획 모드** — 실행에 들어가기 전에 계획만 세우고 사용자의 검토를 받는 Claude Code의 작업 방식입니다(9장, 19장).
- **과적합** — 배운 예시에 지나치게 맞춰져 새 상황에서 엉뚱하게 답하는 현상입니다(1장).
- **단일 원본(SSOT, Single Source of Truth)** — 어떤 주제의 정보가 한 곳에만 있는 상태입니다. 나머지 문서는 이 원본에서 만든 사본으로 다룹니다(7장, 16장, 18장).
- **대규모 검토 모드** — 서브에이전트를 자동으로 나눠 지휘해 많은 문서를 교차 검토하는 방식입니다. 시간과 사용량이 크게 들고, 결과의 최종 확인은 사람이 합니다. 제공 여부와 이름은 제품에 따라 다릅니다(10장).
- **데모** — 가짜 데이터로 화면 흐름과 사용성을 먼저 눌러 보게 만든 시제품입니다. 실제 데이터와 로그인은 이후에 붙입니다(18장).
- **도메인** — 사람이 쓰는 주소 이름(예: ○○○.com)입니다. DNS가 이름을 컴퓨터가 쓰는 번호(IP 주소)로 바꿔 찾아갑니다(6장).
- **되돌리기(롤백)** — 배포나 변경이 잘못되었을 때 직전의 정상 상태로 돌아가는 일입니다. 되돌릴 대상을 미리 기록해 두어야 빠르게 할 수 있습니다(20장).

- **딥리서치** — 여러 출처를 오래 찾아 읽고 종합해 출처가 달린 보고서를 만드는 조사 방식입니다. 출처가 있어도 열어서 확인하는 일은 사람의 몫입니다(15장).
- **마크다운(md)** — 제목과 목록, 표를 기호 몇 개로 적는 일반 텍스트 형식입니다. AI가 읽고 쓰기 쉬워 원본 문서에 알맞습니다(7장).
- **멀티 세션** — 여러 작업을 각각 다른 세션에서 나눠 진행하는 방식입니다. 세션끼리는 기억을 공유하지 않으므로 인계 문서로 넘깁니다(10장).
- **메타 프롬프트(재사용 프롬프트 파일)** — 잘 통한 작업 과정을 주제만 바꿔 다시 쓸 수 있게 저장한 프롬프트입니다. 주제 자리는 비워 둡니다(3장, 16장, 17장).
- **메타 프롬프팅** — 프롬프트를 직접 쓰는 대신 AI에게 좋은 프롬프트를 설계하게 하는 방식입니다(3장).
- **멱등성** — 같은 요청을 여러 번 보내도 결과가 한 번 보낸 것과 같은 성질입니다. 문자 발송처럼 중복이 문제 되는 자동화에서 필요합니다(부록 D).
- **모델** — 응답을 만들어 내는 AI 본체입니다. Claude에는 성능과 속도가 다른 여러 모델이 있고, 쓸 수 있는 모델은 요금제와 시점에 따라 다릅니다(2장).
- **바이브코딩** — 코드를 한 줄씩 직접 쓰는 대신 의도를 말로 전하고, 나온 결과를 보며 다듬어 가는 개발 방식입니다(4장).
- **배포** — 만든 결과물을 다른 사람이 쓸 수 있는 곳에 올리는 일입니다. 올리기 전 점검, 올린 뒤 확인, 되돌릴 방법을 한 세트로 준비합니다(20장).
- **백엔드** — 프로그램이 어떻게 움직이는지를 정하는 규칙의 층입니다. 화면(프론트엔드), 정보(데이터베이스)와 함께 세 층을 이룹니다(6장).
- **변환 사다리** — 원본 md에서 html, docx, pdf나 pptx로 내려가며 문서를 만드는 순서입니다. 고칠 때는 위쪽 원본에서 고치고 다시 내려보냅니다(7장).
- **브랜치** — 깃에서 기존 작업 흐름과 분리해 시험 삼아 작업하는 갈래입니다(20장).
- **사고 강도(effort)** — 모델이 답하기 전에 얼마나 오래 생각할지 정하는 설정입니다. 높일수록 사용량이 늘어납니다. 설정 방법은 공식 문서로 확인합니다(2장, 10장).
- **서브에이전트** — 메인 작업에서 특정 일을 맡겨 별도의 대화 공간에서 처리하게 하는 보조 에이전트입니다(10장).
- **서버와 클라이언트** — 요청하는 쪽(브라우저)이 클라이언트, 응답하는 쪽(늘 켜져 있는 컴퓨터)이 서버입니다(6장).
- **세션** — Claude와 시작부터 끝까지 이어지는 한 번의 작업 대화입니다. 세션이 끝나면 기억은 이어지지 않으므로 중요한 것은 파일에 남깁니다(9장, 10장).

- **스킬** — 이름, 설명, 지시문을 담은 `SKILL.md` 파일로 만든 작업 묶음입니다. 조건이 맞으면 Claude가 불러 씁니다(11장, 부록 B).
- **아첨(sycophancy)** — 모델이 사용자가 듣고 싶어 할 답 쪽으로 치우치는 경향입니다. 틀린 주장에도 동의할 수 있으므로 동의와 공감을 구하는 용도로는 신뢰하지 않습니다(1장).
- **아트보드** — 원본 문서를 최종 쪽 모양으로 펼쳐 놓은 확인용 화면입니다. 쪽 번호로 짚어 고친 뒤 확정된 모양대로 문서를 만듭니다(16장).
- **아티팩트** — Claude가 대화 옆에 만들어 보여 주는 문서, 코드, 웹페이지 같은 결과물입니다(8장).
- **양식 규칙 파일** — 회사 양식과 지난 문서에서 쪽, 글꼴, 제목 단계, 문체 규칙을 뽑아 적어 둔 파일입니다. 말로 설명하는 대신 파일로 넘깁니다(16장).
- **에이전트** — 목표를 받아 파일을 열고 만들고 실행하는 여러 단계를 스스로 이어서 수행하는 AI입니다(1장).
- **완료 기준** — 작업이 끝났다고 인정하는 조건을 확인 가능한 문장으로 적은 것입니다. "잘 동작한다"가 아니라 "표의 합계가 원본과 일치한다"처럼 씁니다(9장).
- **온도** — 답을 고를 때 흔한 말만 고를지 드문 말도 고를지 정하는 설정입니다. 낮으면 비슷한 답이 되풀이되고, 높으면 새롭지만 지어낼 위험이 커집니다(1장).
- **워크슬롭(workslop, AI 슬롭)** — AI로 쉽게 찍어 낸 그럴듯하지만 알맹이 없는 결과물입니다. 만든 사람의 수고가 받는 사람의 검토 부담으로 넘어갑니다(19장).
- **워크트리** — 깃 저장소 하나를 여러 폴더에 펼쳐 서로 다른 브랜치에서 동시에 작업하게 하는 기능입니다. 세션을 병행할 때 파일 충돌을 줄입니다(10장).
- **워크플로우** — 하나의 결과물을 내기까지 일이 흘러가는 순서입니다. AX는 이 순서의 칸을 줄이거나 바꾸는 일입니다(7장).
- **인계 문서(핸드오프)** — 세션 사이 또는 사람 사이에 작업을 넘기는 문서입니다. 확정된 것, 결정의 이유, 남은 일을 담습니다(10장, 20장, 22장).
- **자가검증** — AI가 자기 결과를 다시 점검하는 단계입니다. 핵심 주장 고르기, 독립 질문, 비교, 제약 체크리스트의 4단계가 대표적입니다(19장).
- **저장소(리포지토리)** — 프로젝트의 파일과 변경 이력을 담는 공간입니다(20장).
- **정보 원본(SOT, Source of Truth)** — 정보가 처음 기록되는 지점입니다. 값이 어긋나면 이 원본을 따릅니다(22장).
- **챔피언** — 직책이 아니라 역할입니다. 조직 안에서 AI 활용을 앞서 해내고, 다른 직원도 쓸 수 있게 흐름을 만들어 굴리는 사람입니다(7장, 22장).
- **커넥터** — Claude를 메일, 문서 같은 외부 서비스에 연결하는 통로입니다(8장, 11장).

- **커밋** — 깃에서 현재까지의 변경을 하나의 기록으로 저장하는 일입니다. 커밋해 두면 그 시점으로 돌아갈 수 있습니다(20장).
- **컨텍스트** — AI가 한 번의 작업에서 참고하는 모든 내용입니다. 대화, 첨부 파일, 지침이 포함됩니다(8장, 10장).
- **컨텍스트 엔지니어링** — 이 책에서는 AI에게 어떤 자료를 얼마나 주어 작업하게 할지 설계하는 일을 가리킵니다. 정의는 자료마다 조금씩 다릅니다(1장).
- **컨텍스트 요약(compact)** — 긴 대화를 요약해 컨텍스트를 줄이는 기능입니다. 요약 과정에서 세부가 빠질 수 있으므로 중요한 결정은 문서로 남깁니다(10장, 20장).
- **클라이언트** — 서버에 요청을 보내는 쪽입니다. "서버와 클라이언트" 항목을 봅니다(6장).
- **토큰** — AI가 글을 처리하는 단위입니다. 단어보다 작은 조각인 경우가 많습니다(1장).
- **포트** — 한 컴퓨터 안의 문 번호입니다. `localhost:3000`의 3000이 포트입니다(6장).
- **프로젝트** — Claude 앱에서 관련 대화, 파일, 지침을 한곳에 묶어 두는 작업 공간입니다(8장).
- **프론트엔드** — 눈에 보이는 화면의 층입니다(6장).
- **프롬프트** — AI에게 주는 입력 문장이나 지시문입니다(3장).
- **프롬프트 주입** — 문서나 웹페이지 안에 숨겨진 지시로 AI의 행동을 바꾸려는 공격입니다. 외부에서 가져온 내용은 지시가 아니라 자료로 다룹니다(11장).
- **하네스** — 이 책에서는 규칙, 진행, 이유를 파일로 남겨 다음 세션과 다음 사람이 이어받게 하는 작업 환경을 가리킵니다(9장, 16장).
- **하네스 엔지니어링** — 이 책에서는 AI가 긴 시간 큰 작업을 하도록 작업 환경과 점검 장치를 갖추는 일을 가리킵니다. 정의는 유동적입니다(1장).
- **합의서** — 작업 전에 목표, 규칙, 금지 사항, 완료 기준을 적어 두는 `CLAUDE.md` 파일을 이 책에서 부르는 이름입니다. Claude Code가 세션을 시작할 때 읽지만 강제 규칙은 아니므로, 반드시 막아야 하는 일은 권한 설정으로 지킵니다(9장).
- **호스팅** — 서버를 빌려 늘 켜 두고, 올린 결과물을 주소로 열 수 있게 하는 서비스입니다(6장, 18장).
- **확인 필요 표기([확인 필요], [담당자 확인 필요])** — AI가 지어내지 않고 빈칸으로 남기는 표시입니다. 사람이 확인해 채웁니다(16장, 17장).
- **환각(할루시네이션)** — AI가 사실이 아닌 내용을 그럴듯하게 만들어 내는 현상입니다. 출처 요구와 대조 검증으로 줄입니다(1장, 15장).
- **환경 변수** — 비밀 키나 주소처럼 코드 밖에 두고 실행 환경에서 넣어 주는 설정값입니다. 코드나 저장소에 직접 적지 않습니다(13장, 18장, 20장).

영문 용어

- **AI(Artificial Intelligence)** — 사람의 지적 작업 일부를 수행하도록 만든 소프트웨어를 통틀어 부르는 말입니다(1장).
- **API(Application Programming Interface)** — 프로그램끼리 요청과 응답을 주고받는 약속된 창구입니다. 문자 발송 서비스나 Claude를 코드에서 부를 때 씁니다(13장).
- **AX(AI Transformation)** — 조직의 일하는 방식을 AI에 맞게 바꾸는 일입니다(7장, 21장).
- **CLAUDE.md** — 프로젝트 폴더에 두는 지침 파일입니다. 이 책에서는 합의서라고 부릅니다(9장).
- **CLI/GUI** — CLI(Command Line Interface)는 글로 명령해 조작하는 방식, GUI(Graphical User Interface)는 버튼과 화면으로 조작하는 방식입니다. Claude Code는 CLI에 쉬운 말투를 얹은 도구입니다(9장).
- **CRM(Customer Relationship Management)** — 문의, 고객 정보, 응대 이력을 관리하는 체계입니다. 문의 응대 자동화(13장)의 바탕입니다.
- **DNS(Domain Name System)** — 도메인 이름을 컴퓨터가 쓰는 번호로 바꿔 주는 전화번호부 같은 체계입니다(6장).
- **DX(Digital Transformation)** — 종이와 아날로그 업무를 디지털로 옮기는 일입니다. AX의 앞 단계로 설명합니다(7장).
- **Git** — 파일의 변경 이력을 기록하고 되돌릴 수 있게 하는 버전 관리 도구입니다(20장).
- **GMP(메타 프롬프트 프로토콜)** — 프롬프트를 구조화해 설계하는 액티전스의 프로토콜입니다. 부록 B의 `gmp` 스킬이 이를 적용합니다(3장, 11장).
- **IP 주소** — 컴퓨터가 쓰는 번호입니다(6장).
- **LLM(Large Language Model)** — 대량의 글로 학습한 거대 언어 모델입니다. Claude도 여기에 속합니다(1장).
- **localhost** — 이 컴퓨터 자신을 가리키는 이름입니다. 미리보기 주소로 쓰이며 이 컴퓨터에서만 열립니다(6장).
- **MCP(Model Context Protocol)** — AI 응용 프로그램을 외부 도구와 데이터에 연결하는 오픈 표준입니다. Anthropic이 처음 공개했습니다(11장).
- **md** — 마크다운 파일의 확장자입니다. 문서의 원본 형식으로 씁니다(7장).
- **PRD(Product Requirements Document)** — 무엇을 왜 누구를 위해 만드는지 적은 제품 요구사항 문서입니다. 바이브코딩의 출발점입니다(4장, 18장).
- **RLS(Row Level Security)** — 데이터베이스에서 행 단위로 접근을 제한하는 보안 기능입니다(6장).

- **SKILL.md** — 스킬 한 개를 정의하는 파일입니다. 맨 위에 이름과 설명, 아래에 지시문을 적습니다(11장).
- **SSOT(Single Source of Truth)** — 정보의 원본이 하나뿐인 상태입니다(7장).
- **Two-pass** — 계획을 세운 뒤 한 번 더 별도로 검토하고 실행하는 방식입니다. 여러 영역에 걸치거나 되돌리기 어려운 작업에 씁니다(9장, 19장).

부록 D

체크리스트

실무에서 바로 인쇄해 쓰도록 만든 점검표 열세 가지입니다. 항목 끝의 괄호는 근거를 설명한 장입니다. 해당하지 않는 항목은 건너뛰되, 건너뛴 이유를 한 줄 적어 둡니다.

AI 도입 착수 전

- [ ] AI로 줄이려는 업무가 무엇인지 한 문장으로 적었다. (1장, 21장)
- [ ] 그 업무의 현재 소요 시간과 오류 유형을 직접 재 보았다. 추정치로 대신하지 않았다. (21장)
- [ ] AI에 넘겨도 되는 정보와 넘기면 안 되는 정보를 구분했다. (1장, 이 부록의 보안·개인정보 항목)
- [ ] 결과를 최종 확인하고 책임질 사람을 정했다. (19장, 22장)
- [ ] 가장 작은 첫 업무 하나를 골랐다. 여러 업무를 동시에 시작하지 않는다. (21장)
- [ ] 쓸 도구와 요금제를 정했다. 한도와 가격은 공식 요금제 페이지에서 확인했다. (2장)
- [ ] 첫 3개월의 목표와 점검 시점을 정했다. (22장)
- [ ] 실패했을 때 예전 방식으로 돌아갈 방법을 확보했다. (20장)

요금제 점검

- [ ] 개인용인지 팀용인지, 쓰는 사람 수를 확인했다. (2장)
- [ ] 필요한 기능(프로젝트, Claude Code, 조사 기능 등)이 선택한 요금제에서 제공되는지 공식 안내로 확인했다. (2장)
- [ ] 사용량 한도에 걸렸을 때의 대안(작업 시간 조정, 모델 변경, 요금제 변경)을 정해 두었다. (2장)
- [ ] 자료가 학습에 쓰이는지, 보관 기간이 어떤지 요금제별 정책을 읽었다. (2장)
- [ ] 결제 담당자와 결제 수단을 정했다. 개인 카드로 시작했다면 이전 계획을 세웠다. (2장)
- [ ] 한 달 사용 후 실제 사용량과 요금제가 맞는지 다시 점검할 날짜를 잡았다. (2장)

프롬프트 작성

- [ ] 목적을 한 문장으로 썼다. 무엇을 얻고 싶은지 분명하다. (3장)
- [ ] 이 결과를 누가, 어디에 쓰는지 배경을 알렸다. (3장)
- [ ] 필요한 자료를 첨부하거나 붙였다. AI가 알 수 없는 내용을 기억에 맡기지 않았다. (3장, 8장)
- [ ] 원하는 형식(분량, 표 여부, 말투)을 구체적으로 지정했다. (3장)
- [ ] 하지 말아야 할 것(지어내기, 범위 밖 수정 등)을 명시했다. (3장, 9장)
- [ ] 모르면 모른다고 답하게 했고, 근거를 함께 달게 했다. (3장, 19장)
- [ ] 좋은 결과의 견본이 있으면 한두 개 첨부했다. (3장)
- [ ] 한 번에 너무 많은 일을 시키지 않고 단계를 나누었다. (3장, 4장)
- [ ] 잘 된 프롬프트는 프로젝트 지침이나 스킬로 저장해 재사용한다. (8장, 11장)

처음 세팅

- [ ] 대화창이 아니라 데스크탑 앱에서 폴더를 연결해 쓴다. (8장)
- [ ] 결과가 중요한 일에는 상위 등급 모델을 골랐다. 모델 이름은 문서에 못 박지 않았다. (1장, 2장)
- [ ] 중요한 일에는 사고 강도를 높였고, 높일수록 사용량이 더 든다는 점을 알고 있다. (2장)
- [ ] 작업 전용 폴더를 정했고, 원본은 백업해 두었다. (8장)
- [ ] 첨부 전에 고객 정보, 직원 실명, 미공개 수치, 사진 속 얼굴과 화면 네 가지를 지웠다. (8장)

단일 원본 점검

- [ ] 증상 확인: 최종_최종 같은 이름의 파일이 있는가. (7장)
- [ ] 증상 확인: 같은 숫자를 두 사람이 다르게 말하는가. (7장)
- [ ] 증상 확인: 어느 것이 최신인지 물어봐야 아는가. (7장)
- [ ] 증상 확인: 규정이 메일, 공유 폴더, 개인 컴퓨터에 흩어져 있는가. (7장)
- [ ] 위 증상 중 하나라도 있다면 만든다. 정한다: 이 주제의 원본은 이 파일 하나다. (7장)
- [ ] 표시한다: 나머지는 참고 폴더나 보관 폴더로 옮겼다. (7장)
- [ ] 갱신한다: 바뀌면 원본만 고친다. (7장)

- [ ] 병목 3질문: 가장 오래 걸리는 단계는 어디인가, 매번 같은 방식으로 반복되는가, 입력과 출력이 정해져 있는가. 셋 다 예이면 AI로 바꿀 1순위다. (7장)

문서 제출 전

- [ ] 원본 md 파일이 하나이고, 내용은 그 원본에서만 고쳤다. (16장)
- [ ] 양식 규칙 파일이 있다. (16장)
- [ ] 쪽 모양으로 펼친 확인 화면(아트보드)으로 배치를 확인했다. (16장)
- [ ] PDF를 직접 열어 쪽수, 글꼴, 표 잘림을 봤다. (16장)
- [ ] 남은 [확인 필요] 표시가 없다. (16장)
- [ ] 마지막 손질은 사람이 했다. (16장)

결과보고서 숫자 점검

- [ ] 본문 숫자 셋을 원 기록까지 직접 따라가 대조했다. (17장)
- [ ] 부풀린 표현의 금지어 목록을 검색했다. (17장)
- [ ] 개인정보를 지운 사본을 썼다. (17장)
- [ ] 원인 칸에 AI의 짐작이 없고, 사람이 답했거나 기록에 있는 것만 적었다. (17장)
- [ ] 요약은 본문을 다 쓴 뒤 맨 마지막에 썼다. (17장)

결과물 검수

- [ ] 보내기 전 세 질문(내가 이해했는가, 근거를 확인했는가, 받는 사람이 할 일이 줄어드는가)에 모두 예라고 답했다. (19장)
- [ ] 숫자, 날짜, 고유명사를 원자료와 하나씩 대조했다. (19장)
- [ ] 인용과 출처가 실제로 존재하는지 열어서 확인했다. (15장, 19장)
- [ ] AI가 "확인했다"고 말한 것을 내가 실행해서 다시 확인했다. (19장)
- [ ] 요청하지 않은 변경이 섞여 있지 않은지 변경 목록을 훑었다. (9장)
- [ ] 분석 결과는 원본 데이터의 합계나 건수와 맞는지 대조했다. (14장)
- [ ] 완료 기준을 먼저 적어 두었고, 그 기준으로 통과 여부를 판단했다. (9장, 19장)

- [ ] 다른 세션이나 다른 사람이 같은 자료를 독립적으로 검토했다. (10장, 19장)
- [ ] 잘못된 점은 프롬프트나 합의서에 반영해 같은 오류가 반복되지 않게 했다. (9장)

배포 전

- [ ] 변경분을 커밋했고, 직전 정상 상태로 돌아갈 방법을 적어 두었다. (9장, 20장)
- [ ] 테스트와 검사 명령을 실제로 실행했고, 결과 원문을 확인했다. (19장, 20장)
- [ ] 미리보기 배포에서 핵심 경로를 직접 눌러 보았다. (20장)
- [ ] 휴대폰 화면과 데스크톱 화면 모두에서 확인했다. (18장, 20장)
- [ ] 비밀 키가 코드와 커밋 이력에 들어가지 않았다. (20장)
- [ ] 운영 환경에 필요한 환경 변수가 모두 등록되었다. (20장)
- [ ] 배포 후 확인할 주소와 정상 기준(응답, 화면, 데이터)을 미리 정했다. (20장)
- [ ] 배포 후 문제가 생기면 누가 어떤 순서로 롤백하는지 정해 두었다. (20장)
- [ ] 오류가 났을 때 알려 줄 방법(알림, 로그 확인)이 준비되었다. (20장)

작은 웹 화면 공개 전

- [ ] 화면에 가상의 연습 내용만 있고 실명, 연락처, 건강 정보, 남의 사진과 글이 없다. (5장, 6장)
- [ ] 계산(개수, 겹침)이 손으로 세어도 맞다. (6장)
- [ ] 값을 비워 두거나 아주 긴 글자, 25:00 같은 이상한 값을 넣어도 화면이 깨지지 않는다. (6장)
- [ ] 브라우저 창을 좁혀 휴대폰 화면을 흉내 내 보았고, 글자가 잘리지 않는다. (6장)
- [ ] 한국어 표기, 오타, 날짜 모양이 맞다. (6장)
- [ ] 허락 창에 나온 명령을 읽었고, 삭제와 내려받기, 작업 폴더 밖의 파일, 모르는 설치 명령은 허용하지 않았다. (6장, 9장)
- [ ] 올린 주소를 시크릿 창과 내 휴대폰에서 열어 보았다. (6장, 18장, 20장)
- [ ] 로그아웃과 폴더 정리 같은 퇴장 절차를 알고 있다. (6장, 20장)

보안·개인정보

- [ ] 개인정보(이름, 연락처, 주소 등)를 AI에 넣어야 하는지 먼저 따졌고, 필요 없으면 가렸다. (1장, 13장)

- [ ] 수집하는 정보의 항목, 목적, 보관 기간을 정리했고 방침 문서와 일치한다. (13장, 18장)
- [ ] 수집 근거를 확인했고, 동의를 받는 경우 동의 기록이 남는다. (13장)
- [ ] 비밀 키와 비밀번호는 환경 변수나 비밀 저장소에만 있다. (20장)
- [ ] 연결한 커넥터와 도구의 권한 범위가 필요한 만큼으로 제한되어 있다. (11장)
- [ ] 외부 문서와 웹페이지의 내용을 지시가 아닌 자료로 다루도록 합의서에 적었다. (9장, 11장)
- [ ] 자동 실행 권한에서 삭제, 외부 발송, 결제 같은 되돌리기 어려운 동작은 사람이 승인한다. (9장)
- [ ] 퇴사자와 담당 변경 시 계정, 키, 접근 권한을 회수하는 절차가 있다. (22장)
- [ ] 법령 관련 내용은 공식 법령 문서로 직접 확인했다. 이 부록은 법률 자문이 아니다. (13장)

문자 발송 자동화 운영

- [ ] 발송 서비스의 공식 문서로 사전 등록이 필요한 항목(발신번호 등)을 확인했다. (13장)
- [ ] 문의 답변과 광고를 구분했고, 광고성 문자는 수신 동의를 받은 대상에게만 보낸다. 법령상 요건은 공식 문서로 확인했다. (13장)
- [ ] 연습용 번호로 시험 발송을 먼저 했고, 실제 고객에게는 그 뒤에 켰다. (13장)
- [ ] 같은 문의에 두 번 보내지 않도록 발송 여부를 기록하고 확인한다(멱등성). (13장)
- [ ] 문의 내용을 AI가 분류하는 경우, 자동 발송 대상과 사람이 보는 대상을 나누었다. (13장, 19장)
- [ ] 발송 실패와 재시도의 기록이 남고 확인할 사람이 정해져 있다. (13장)
- [ ] 수신 거부 요청을 받았을 때 처리하는 절차가 있다. (13장)
- [ ] 문구는 담당자가 최종 승인한 것만 쓰고, 변경 이력이 남는다. (13장)
- [ ] 발송 서비스 키는 환경 변수에만 있고, 사용량을 정기적으로 확인한다. (13장, 20장)

사내 담당자 인계

- [ ] 현재 쓰는 도구, 계정, 담당자, 비용 지불 주체를 한 장에 정리했다. (22장)
- [ ] 합의서와 프롬프트, 스킬 파일의 위치와 최신본을 밝혔다. (9장, 11장, 22장)
- [ ] 진행 중인 작업은 핸드오프 문서로 상태와 남은 일을 적었다. (10장, 22장)
- [ ] 자동화마다 무엇이 어디서 실행되는지, 멈추는 방법이 무엇인지 적었다. (13장, 18장)
- [ ] 알려진 문제와 해보고 폐기한 방법을 기록했다. (22장)
- [ ] 후임자가 혼자서 한 건을 끝까지 해보도록 시험했다. (22장)

- [ ] 접근 권한과 비밀 키를 후임자에게 옮기고 이전 담당자 권한을 회수했다. (22장)
- [ ] 정기 점검 일정(월별 사용량, 결과 품질, 규정 변경)을 정했다. (22장)

판권

제목	바이브코딩, 클로드 실무 토탈 솔루션
부제	개념부터 도구, 업무 자동화, 조직 안착까지
지은이	액티전스(ACTIGENCE)
버전	v1.0.0 (2026-11-07 14:17 KST 기준)
최신본	https://actigence.ai/library/downloads
문의	cio@actigence.ai

개정 이력

1. **v1.0.0** 2026-11-07 14:17 KST — 첫 발행

제품의 화면·요금제·기능은 수시로 바뀝니다. 본문에 "(2026년 11월 기준)"으로 표시한 내용은 공식 문서에서 최신 상태를 확인합니다.