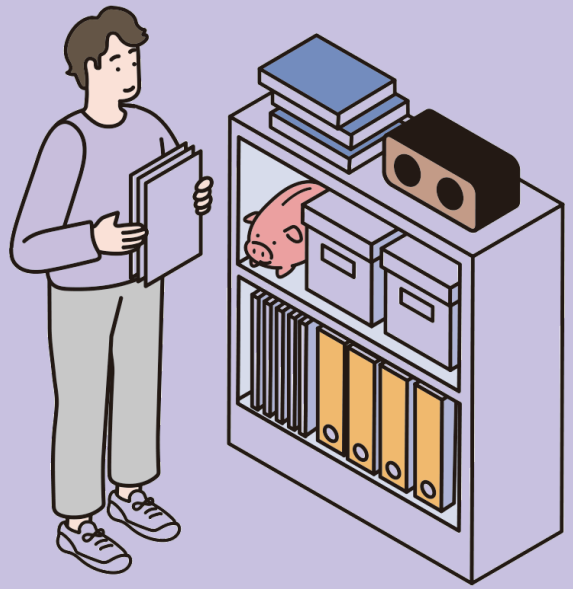


강의용 슬라이드 PART 1-4 + 부록

요즘 교사를 위한 웹앱 만들기

with 바이브 코딩

교실의 불편을 발견하는 눈에서 시작해, 배포까지 마친 하나의 웹 서비스로 끝나는 여덟 개의 장.

이상선 · 김진관 · 이대형 · 윤신영 · 김상섭 지음
한빛미디어 · 2026

OPENING

“이게 왜 이렇게 불편하지?”

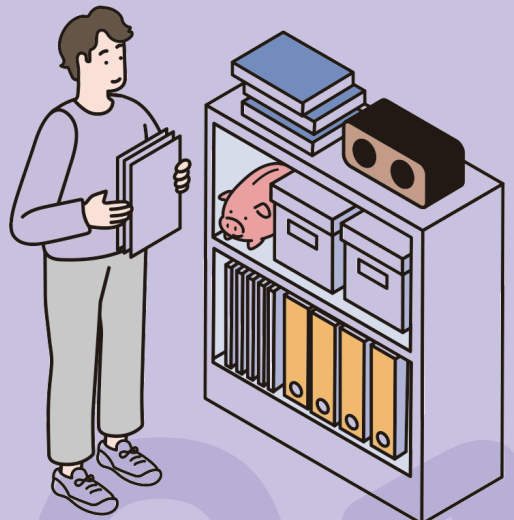
세상의 모든 위대한 서비스는 결국 누군가가 아주 사소한 불편함을 해결하고 싶다는 간절한 마음에서 시작되었습니다. 그리고 교실은, 그런 불편함이 가장 많이 쌓여 있는 공간입니다.

오늘의 여정

앱에 기관을 하나씩 달아 갑니다

네 개의 파트가 각각 앱에 없던 능력을 하나씩 더합니다.

01	**아이디어를 서비스로 전환하는 첫걸음** 교실의 불편을 PRD로 · 브라우저에서 도는 첫 앱 · Lovable로 첫 배포	화면
02	**앱에 지식을 더하다** API와 JSON · 나이스 급식 조회 · 카메라로 보고 손글씨를 읽는 앱	눈과 귀
03	**앱에 클라우드 기능을 심다** 데이터베이스와 BaaS · Supabase To-Do 앱 · RLS로 잠그기	기억
04	**로컬 환경에서 진짜 개발자처럼 일하기** 내 컴퓨터를 개발 서버로 · Git으로 되돌리기 · Vercel 자동 배포	작업실
부록	**그 밖의 인공지능 도구들** Cursor · Claude Code — 소개와 설치	확장



Part 01

아이디어를 서비스로 전환하는 첫걸음

Preview 내 불편함이 앱이 되기까지

01 모든 아이디어는 교실에서 시작된다

02 브라우저에서 동작하는 나의 첫 프로그램

03 AI와 함께 만드는 첫 웹 서비스, Lovable과 웹앱의 개념

PART 01 · 1장

200명의 서술형 답안지는 16시간짜리 노동이다

16 시간 이상

답안지 한 장에 **3~5분**, 200명이면 **1,000분**. 같은 반 객관식은 OMR 판독기가 **1분**이면 끝내는 일이다.

중학교 2학년 사회 · 총 7개 반 · 학생 200명+

여기서 무너진다

100번째 답안지쯤이면 피드백은 '논리적임', '잘 썼음' 같은 짧은 문구만 남는다.



6월의 어느 오후, 12년 차 혜심 선생님의 교무실. 점수 부여와 맞춤형 피드백과 과세특 기초 자료 정리가 한꺼번에 밀려 있다.

PART 01 · 1장

기술은 발전했는데 평가만 과거에 머물러 있다

교실의 디지털 역설 — 도구는 늘었지만, 읽고 판단하는 일은 그대로 남았다.

기성 AI 채점 보조 도구

키워드가 들어 있는지 센다

- 정해진 키워드의 포함 여부만 확인한다
- 그 문장이 왜 그렇게 쓰였는지는 보지 않는다

"결국 사람이 일일이 읽어봐야 하는구나."

VS

교사가 읽고 싶은 것

이 아이가 무엇을 생각했는가

- **사회적 공감**이 담겼는가
- **창의적 대안**을 내놓았는가
- 지난 과제보다 나아졌는가

혜심 선생님이 바란 것

"사진만 찍으면 텍스트로 변환되고, 제가 정한 기준에 맞춰 초안 피드백을 만들어주는 도구만 있다면 말이죠."

체념을 정의로 바꾸면, 불편함은 입력·판단·출력 세 문제가 된다



세 칸으로 쪼개는 순간 어디를 기계에 맡길지가 보인다. 이 구조는 잠시 뒤 만날 모든 앱의 뼈대, 입력 · 처리 · 출력과 같다.

PART 01 · 1장

한숨이 앱의 이름이고,
한숨의 이유가 핵심 기능이다

받아 적을 것

“에휴, 이걸 또
언제 다 하나”

이 한숨을 포착하면 앱의 이름이 나온다. 한숨이 나온 이유를 한 줄 더 적으면, 그게 첫 화면에 들어갈 기능이 된다.

순서를 뒤집지 말 것

바이브 코딩 자체가 목적이 되어서는 안 된다. 선생님의 삶을 더 낫게 하는 수단이어야 한다.

매일 되풀이되는 질문

- “오늘 급식 뭐예요?”라는 반복되는 질문
- 발표자를 뽑을 때 생기는 공정성 논란
- 흩어지는 상담 기록

매번 새로 하는 일

- 학기마다 다시 하는 청소 구역 배정
- 우리 반에 맞는 타이머가 없는 아쉬움

PART 01 · 1장

좋은 문제는 만능이 아니라 칼날처럼 구체적이다

나쁜 문제

"행정 업무를 자동화하고 싶다."

범위가 너무 넓고 모호하다. 어디서부터 손대야 할지 사람도 AI도 알 수 없다.

오른쪽의 네 가지 가운데 하나도 들어 있지 않다.

VS

좋은 문제

"매주 월요일 1교시에 수거하는 독후감 활동지에서, 학생들이 공통적으로 언급한 키워드 3가지만 뽑아내고 싶다."

대상 시간 행위 결과

이미 배우셨다

분석 · 설계 · 개발 · 실행 · 평가. **ADDIE 모형**을 아는 교사는 앱 개발의 절차를 이미 배운 사람이다. 마지막 평가 단계의 실제 모습은 "선생님, 이 버튼이 안 눌려요!"다.

PART 01 · 1장

“나는 [사용자]로서, [목표]를 위해, [기능]이 필요하다”

사용자	목표	필요한 기능
교사	채점 고충을 덜기	답안지 사진을 찍어 텍스트로 바꾸는 기능
교사	개별 맞춤형 성장을 확인하기	텍스트에서 핵심 키워드를 추출해 그래프로 보여주는 기능

✅ 기능 1: OCR 인식 및 텍스트화 (구현안)

1. 기능 정의 (개발 관점)

- 입력: 답안지 이미지 (jpg, png, pdf)
- 처리: OCR 엔진으로 텍스트 추출
- 출력: 구조화된 텍스트 (문장 단위 or 번호별 답안)

문장 하나를 받은 AI는 곧바로 입력 · 처리 · 출력으로 되받아 적는다.

AI는 독심술사가 아니다

세계 최고의 개발자이지만, 우리 교실의 맥락은 모르는 신입사원이기도 하다. 선생님의 언어를 시스템의 언어로 옮기는 **번역가**가 필요하고, 그 자리가 이 한 문장이다.

대괄호 세 칸을 채우는 일이 곧 **바이브를 맞추는** 일이다. 추상적인 구름 위의 생각을 구체적인 현실의 문장으로 끌어내리는 과정이다.

PART 01 · 1장

MVP는 아이언맨 슈트가 아니라 바퀴 달린 의자다

슈퍼 앱의 함정

다 넣으려다 아무것도 못 만든다

앱을 구상하다 보면 기능이 저절로 붙는다.

- 로그인 기능도 넣고 싶고
- 알람도 울리게 하고 싶고
- 레벨업 시스템까지 붙이고 싶다

여기서 멈춘다

기능이 하나 늘 때마다 완성은 한 주씩 멀어진다. 그러다 결국 아무것도 손에 남지 않는다.

MVP · 최소 기능 제품

핵심 기능만 동작하는 최소한의 제품. 앱이 작동하는 데 꼭 필요한 기능부터 먼저 완성한다.

단순히 작은 제품이 아니라 **검증 가능한** 최소 제품이라는 점이 핵심이다.

혜심쌤 채점 앱의 MVP

"사진을 찍으면 글자가 인식된다"

한 학기 분량 제본 교재가 아니라 한 장짜리 활동지부터.

PART 01 · 1장

PRD 한 장이 고민의 80%를 미리 없앤다

1페이지 PRD	혜심쌤의 매직 피드백 비서
제품 이름	혜심쌤의 매직 피드백 비서
배경과 목적	200명의 서술형 답안지 채점 시간을 줄이고 질 높은 피드백을 제공하기 위함
핵심 기능	답안지 사진 OCR 인식 · 채점 기준 업로드와 AI 자동 채점 · 피드백 result PDF 리포트 생성
사용자 환경	교무실 데스크톱이나 태블릿으로, 채점 기간에 집중적으로 사용
성공의 정의	채점 시간이 기존 대비 50% 이상 단축되었는가

"안녕, 나는 이런 PRD를 가지고 앱을 만들려고 해.
먼저 첫 번째 기능을 구현해 줄래?"

AI의 응답

"좋아요. PRD 기준으로 기능 1, 답안지 OCR 인식 및 텍스트화를 실제로 구현 가능한 형태로 설계하고 코드까지 바로 쓸 수 있게 만들어 드릴게요."

다섯 칸을 채운 문서를 통째로 붙여넣는 것으로 대화가 시작된다. 고통이 솔직하게 담긴 문서가 가장 훌륭한 PRD다.

PART 01 · 2장

복사 → 메모장 → .html로 저장 → 더블클릭, 네 번이면 앱이 태어난다

1

복사

AI가 내놓은 코드 블록 오른쪽 위의 복사 아이콘을 누른다. 읽을 필요는 없다.

2

메모장

메모장을 열고 붙여넣는다. 다른 프로그램은 설치하지 않는다.

3

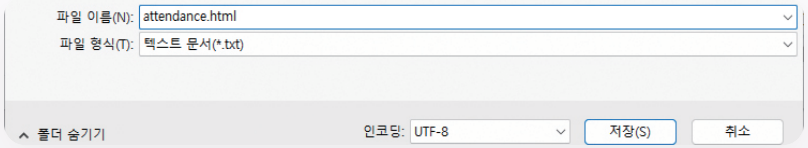
.html로 저장

파일 이름을 attendance.html로 적고 저장을 누른다.

4

더블클릭

저장한 파일을 두 번 누르면 브라우저가 열리며 앱이 뜬다.



여기서 제일 많이 막힌다

파일 형식이 '텍스트 문서(*.txt)'로 되어 있어도 괜찮다. 이름 끝이 반드시 **.html**이어야 하고, 인코딩은 **UTF-8**로 둔다.

PART 01 · 2장

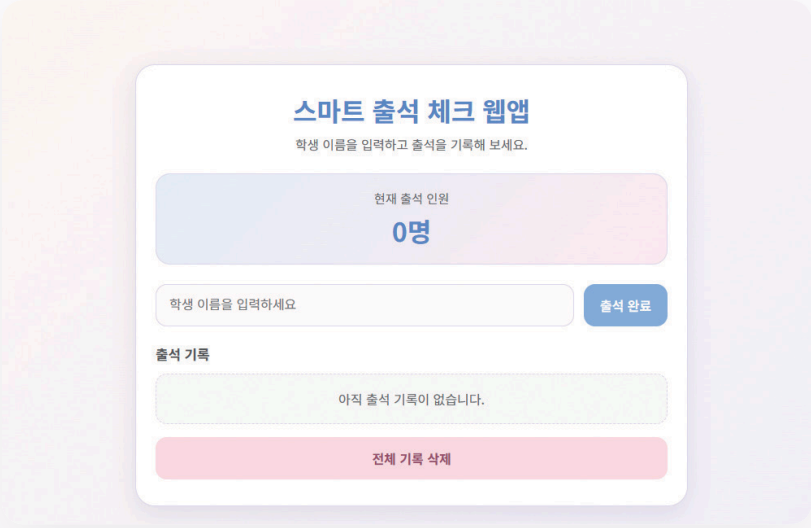
코드 한 줄 치지 않았는데 작동하는 도구가 생겼다

여러분은 지금까지 단 한 줄의 코드도 직접 타이핑하지 않았다. 그런데 불편함을 해결하는 **실제로 작동하는 도구**가 세상에 태어났다.

이름을 넣고 버튼을 누르면 시각이 기록되고 인원수가 올라간다. 그게 전부다. 방금 만든 것은 아직 '숨만 쉬는' MVP다. 그런데 프롬프트에 적은 다섯 줄이 화면의 다섯 부분으로 그대로 옮겨 왔다. 기능은 지금부터 말로 붙여 나가면 된다.

오늘 가져갈 감각

내가 상상하고 설명하면, 그것이 기술이 된다.



메모장으로 저장한 attendance.html을 더블클릭한 결과. 파스텔톤 화면, 실시간 인원 카운트, 출석 기록 영역이 프롬프트에 적은 그대로 나왔다.

PART 01 · 2장

HTML은 설계도, CSS는 인테리어, JavaScript는 점원이다

HTML · 구조

건축 설계도

기둥을 세우고 테이블과 창문의 자리를 정한다. 브라우저는 이 뼈대를 DOM이라는 좌석 배치도로 읽는다.

```
<h1>제목</h1>
```

제목

HTML만 있을 때 — 검은 글자 하나

CSS · 디자인

인테리어

벽지 색과 조명 밝기를 정한다. 화면이 커지고 작아져도 배치가 알아서 따라 바뀐다.

```
background-color: yellow;
```

제목

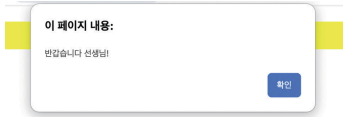
CSS를 얹으면 — 노란 상자에 파란 글자

JavaScript · 기능

점원

손님이 벨을 누르면 다가온다. 눌렀을 때 무엇이 일어날지, 그 반응을 담당한다.

```
onclick="alert('반갑습니다!')"
```



JavaScript를 달면 — 누르니 창이 뜬다

PART 01 · 2장

로컬 스토리지는 그 기기 안에서만 열리는 작은 창고다

새로고침해도 명단이 남는 이유이자, 옆자리 컴퓨터에서는 비어 있는 이유다.

LOCAL STORAGE

브라우저가 가진 작은 창고

브라우저가 내주는 작은 메모장이다. 완벽히 독립되고 고립된 저장소라서, 창을 닫았다 다시 열어도 적어 둔 내용이 그대로 남아 있다.

> “아까 만든 출석 체크 앱에 ‘로컬 스토리지 저장 기능’을 추가해 줘. 새로고침해도 출석한 학생 명단이 그대로 남아있게 해줘.”

다만, 그 기기 안에서만

- 교실 컴퓨터에 적은 기록은 집에서 열리지 않는다
- 같은 기기라도 브라우저를 바꾸면 빈 창고다
- 동학년 선생님과 함께 볼 방법이 없다

그래서 다음이 필요하다

여럿이 같이 보려면 창고를 내 컴퓨터 밖에, 클라우드에 두어야 한다.

BRIDGE · 2장

내 컴퓨터에서만 열리는 파일은 아직 서비스가 아니다

attendance.html은 내 바탕화면에서만 열립니다. 메신저로 보내도 상대의 기기에서는 빈 창고가 열립니다. 주소를 가진 서비스가 되려면, 창고를 내 컴퓨터 밖에 두어야 합니다.

PART 01 · 3장

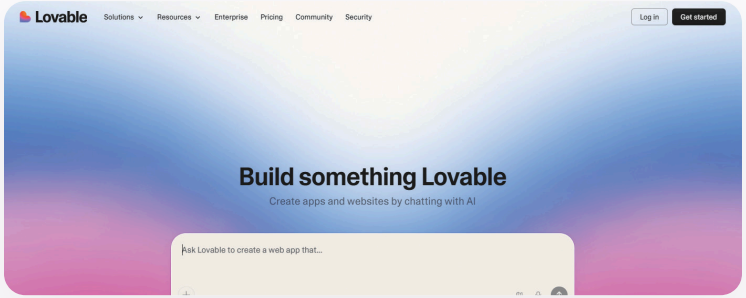
Lovable은 대화로 주문받는 풀스택 앱 빌더다

lovable.dev

요술 램프의 지니

프로그래밍 언어를 몰라도 된다. 카카오톡으로 동료 선생님과 대화하듯 채팅창에 원하는 기능을 적으면 화면과 서버와 데이터베이스가 한꺼번에 만들어진다. 원래는 **GPT 엔지니어**라는 오픈소스 프로젝트에서 출발했다.

> "우리 반 학생들의 청소 당번을 무작위로 정해주는 화면을 만들어 줘."



첫 화면은 입력창 하나가 전부다. 영어로 보이더라도 한글로 적으면 그대로 알아듣는다.

Lovable 클라우드

서버와 데이터베이스까지 Lovable 안에서 해결하는 원스탑 기능이다. 예전처럼 API 키를 밖에서 복사해 오는 과정이 대부분 사라진다.

PART 01 · 3장

화면만 주문하지 말고, 데이터 구조까지 함께 주문하라

1

데이터베이스 구조

무엇을 남길지 먼저 적는다. 작성자 이름, 글 내용, 수정용 비밀번호.

2

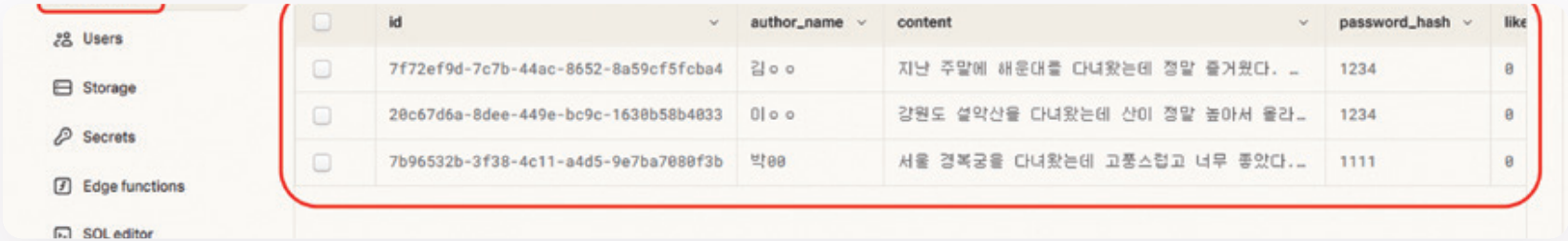
화면 구성 · UI

타임라인 형태, 오른쪽 위 추가 버튼, 이름과 내용과 비밀번호를 받는 모달 폼.

3

기능 · Logic

비밀번호가 맞을 때만 수정·삭제. 최신 글이 위로, 열 개씩 페이지네이션.



주문한 대로 만들어진 posts 테이블. 첫 블록에 적어 둔 항목이 그대로 컬럼이 되었다 — author_name · content · password_hash.

PART 01 · 3장

Publish 버튼 하나가 내 파일을 개업한 가게로 바꾼다

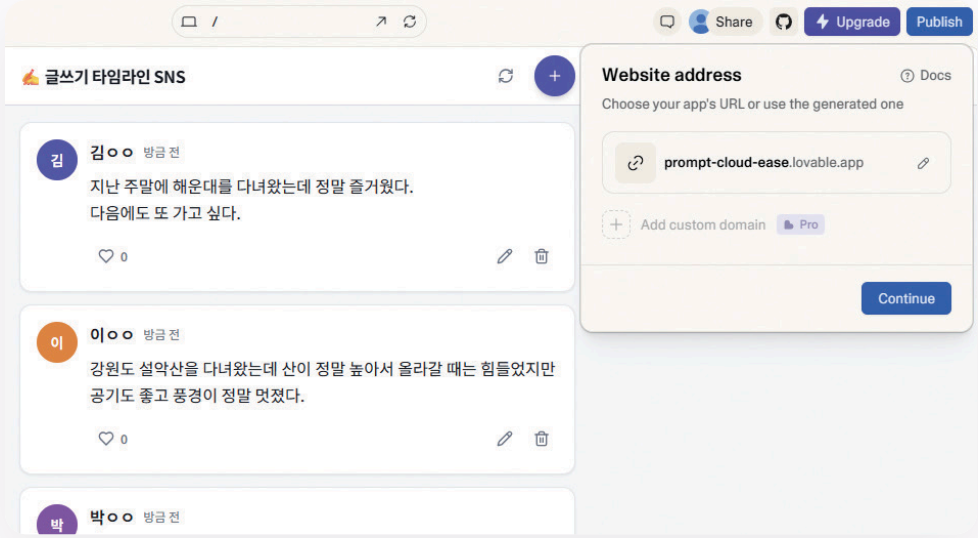
배포 · DEPLOY / PUBLISH

내 컴퓨터에서 연습하던 주방을, 번화가에 가게를 내고 간판을 다는 일이다. 누구나 주소만 있으면 들어올 수 있게 된다.

```
프로젝트명.lovable.app
```

주소가 생긴 뒤

학생은 집에서 스마트폰으로 글을 남기고, 교사는 퇴근길 지하철에서 그 글을 본다.



[Publish]를 누르면 주소가 제안된다. [Continue]까지 두 번이면 끝난다.

PART 01 · 3장

웹앱은 홀·주방·냉장고·주문서로 이루어진 식당이다

프론트엔드 · 클라이언트 사이드

손님이 앉는 홀

눈으로 보고 손으로 누르는 모든 영역. 메뉴판과 테이블과 점원이 여기 있다. 브라우저가 이 홀을 지어 올리는 시공팀이다.

백엔드 · 서버 사이드

보이지 않는 주방

넘어온 주문을 실제로 처리해 결과를 돌려준다. 평균을 내고 등급을 매기는 규칙, 곧 앱의 두뇌가 여기서 돈다.

데이터베이스

대형 냉장고

육류 칸과 채소 칸을 엄격히 나눠 둔 창고. 누가 언제 무슨 글을 남겼는지가 칸별로 보관된다.

API

주고받는 주문서

홀과 주방은 서로 들어가지 않는다. 약속된 양식의 주문서로만 대화한다. 그 양식이 API다.

RECAP · PART 01

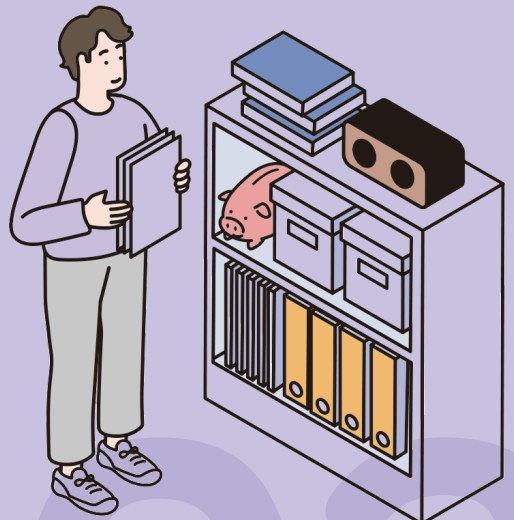
여기까지 답할 수 있습니까

1. **내 교실의 불편 하나를 "나는 [사용자]로서, [목표]를 위해, [기능]이 필요하다"로 말할 수 있습니까?**
   좋은 문제는 칼날처럼 구체적입니다. 대상 · 시간 · 행위 · 결과가 다 들어 있어야 합니다.

2. **그 앱에서 기능을 하나만 남긴다면 무엇을 남기겠습니까?**
   아이언맨 슈트가 아니라 바퀴 달린 의자입니다. 나머지는 버리는 게 아니라 다음 주에 붙이는 것입니다.

3. **HTML과 CSS와 JavaScript가 각각 무슨 일을 하는지 한 줄로 설명할 수 있습니까?**
   설계도 · 인테리어 · 점원입니다. 이름을 지목해서 고쳐 달라고 해야 AI의 대답이 정확해집니다.

4. **내 파일이 서비스가 되려면 무엇이 더 필요한지 말할 수 있습니까?**
   주소 하나, 그리고 내 기기 밖에 있는 창고입니다. 화면만 주문하면 화면만 남습니다.



Part 02

앱에 지식을 더하다

Preview 고립된 섬에서, 연결된 대륙으로

04 서비스는 혼자가 아니야, API 연동하기

05 눈과 귀, 생각을 가진 AI 웹앱 만들기

PART 02 · 4장

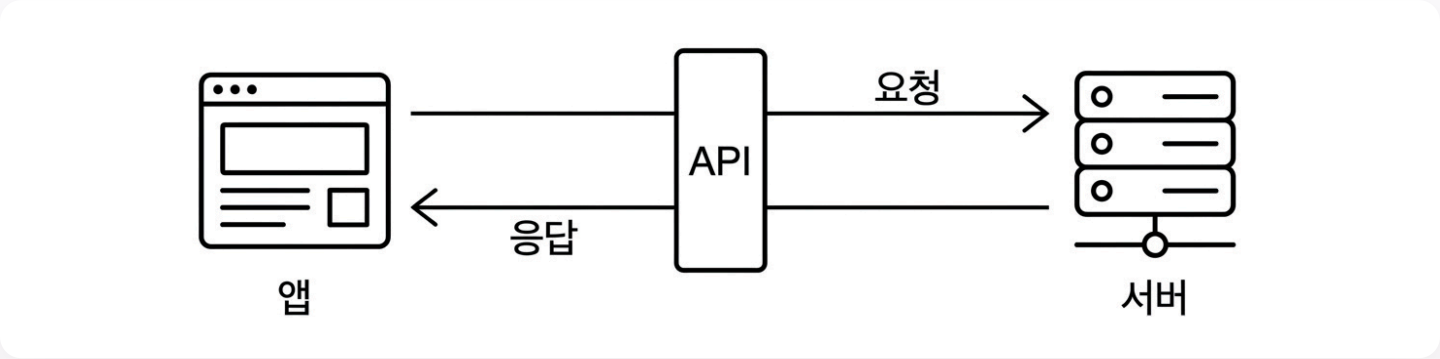
우리가 만든 앱은 아직 고립된 섬이다

우리 반 안에서만, 내가 입력한 것만 압니다. 디자인이 깔끔하고 버튼이 잘 작동해도, 정작 앱 내부를 채울 살아 있는 실시간 정보가 없습니다.

"선생님, 내일 현장체험 학습 가는데 날씨는 어떨까요?"

PART 02 · 4장

API는 사람이 아니라 기계들끼리 쓰는 대화 창구다



앱이 약속된 주소로 요청을 보내면, 서버가 응답을 돌려준다. 그 사이에 선 것이 API다.

사람 ↔ 기계 · UI

버튼과 입력창

사람이 컴퓨터와 상호작용하도록 돕는 시각적 장치.

기계 ↔ 기계 · API

텍스트로 여는 창구

인간이 배제된 상태에서 두 소프트웨어가 데이터를 교환하는 인터페이스.

PART 02 · 4장

서버는 전체를 열지 않고, 허락한 것만 건넨다

API 통신은 철저하게 '사전에 약속된 주소(URL)'와 '안전한 규칙' 위에서 이루어진다.

흔한 오해

창고 문을 통째로 연다

데이터를 준다고 하면 남의 서버 안을 마음대로 들여다보는 장면을 떠올린다. 그래서 개방이 곧 위험이라고 생각한다.

VS

실제

창구로 정해진 것만 건넨다

서버는 자신의 전체 데이터베이스를 외부에 직접 노출하지 않는다. 오직 외부로 공유하기로 허락한 특정한 데이터만 선별해 돌려보낸다.

요청은 이렇게 생겼다

"서울초등학교의 오늘 중식 메뉴 데이터를 보내달라"고 정확한 양식에 맞춰 질문을 던지는 방식이다.

PART 02 · 4장

API가 돌려주는 건 표가 아니라 이름표 붙은 텍스트 덩어리다

JSON · 제이슨

“이름표” : “내용”

API를 통해 우리 앱으로 전달되는 텍스트 데이터의 표준 양식이다. 핵심 규칙은 단 하나, **이름표(Key)와 내용(Value)이 한 쌍을 이룬다**는 사실이다. 이름표를 큰따옴표로 감싸고 콜론 뒤에 실제 데이터를 놓는다.

사전을 넘기듯

구조를 알면 원하는 정보의 위치를 손가락으로 짚어 낼 수 있다.

```
{
  "학교": "서울초등학교",
  "날짜": "2024-05-05",
  "메뉴": "비빔밥, 된장국, 우유"
}
```

기계들의 통로를 거쳐온 데이터는 컴퓨터가 가장 빠르게 처리할 수 있는 ‘날것의 텍스트 덩어리’ 형태를 띤다.

다음 장에서 쓸 문장

이름표를 읽을 줄 알면 AI에게 어느 Key를 꺼내라고 지시할 수 있다.

PART 02 · 4장

DDISH_NM이 요리명이라는 걸 알면 프롬프트가 달라진다

```
{
  "MMEAL_SC_NM": "중식",
  "DDISH_NM": "비빔밥(1.5.)",
  "CAL_INFO": "650.5 Kcal"
}
```

나이스(NEIS) 급식 API가 실제로 돌려주는 응답의 일부다.

요리명 뒤의 숫자

알레르기 유발 식재료 번호다. 학생의 안전과 직결되는 정보이므로 읽기 쉬운 형태로 다듬어 제공한다.

이름표(Key)	무슨 뜻인가
MMEAL_SC_NM	식사 구분 — 중식
DDISH_NM	요리명 — 비빔밥
CAL_INFO	열량 — 650.5 Kcal

응답받은 JSON 중에서 '메뉴'라는 Key 값만 추출해서 화면에 띄워줘.

PART 02 · 4장

국가가 이미 지어 둔 데이터 생태계를 부품처럼 가져다 쓴다

오픈 API · OPEN API

누구나 쓰라고 열어 둔 데이터 통로

국가 기관이나 IT 기업이 자신들의 방대한 데이터베이스를 누구나 공공의 목적으로 활용하도록 개방해 둔 통로다.

교실의 계산

매일 아침 조리실 식단표를 베껴 앱에 수기로 입력할 이유가 사라진다.



나이스 교육정보 개방 포털

`open.neis.go.kr`

전국 시·도교육청이 관리하는 유치원부터 고등학교까지의 핵심 교육 데이터.



공공데이터 포털

`data.go.kr`

날씨, 미세먼지, 학교 정보, 교통, 교육 통계 등 국가 인프라 데이터.

PART 02 · 4장

프론트엔드는 홀, 백엔드는 주방, API는 종업원과 주문서다

프론트엔드

식당의 홀

사용자 눈에 보이는 앱의 화면과 버튼.

백엔드

식당의 주방

데이터베이스 저장 및 복잡한 정보 처리.

API

종업원과 주문서

프론트엔드와 백엔드 사이의 데이터 전달 통로.

왜 나누는가 ① 철저한 보안 유지

데이터를 깊숙한 주방에 숨겨 두고, 오직 허락된 창구를 통해서만 꺼내 오게 만든다.

왜 나누는가 ② 유연한 확장성

튼튼한 주방 하나만 잘 만들어 두면, 어느 화면에서 주문서를 날려도 같은 요리가 나온다.

PART 02 · 4장

인증 키는 서버로 들어가는 디지털 교직원증이다



open.neis.go.kr · 인증키 신청 양식

1 로그인

구글·카카오톡·네이버 등 소셜 계정으로 들어간다.

2 [활용가이드] - [인증키 신청]

활용 목적란에 '학교 급식 조회 앱 개발 학습'이라고 적는다.

3 즉시 발급, 그리고 보관

화면에 뜬 문자열을 복사해 메모장에 붙여 둔다.

두 가지 제한

인증 키는 **최대 15개**까지 발급받을 수 있으며, **타인에게 양도할 수 없다.**

PART 02 · 4장

프롬프트 한 문단이 급식 조회 앱이 된다

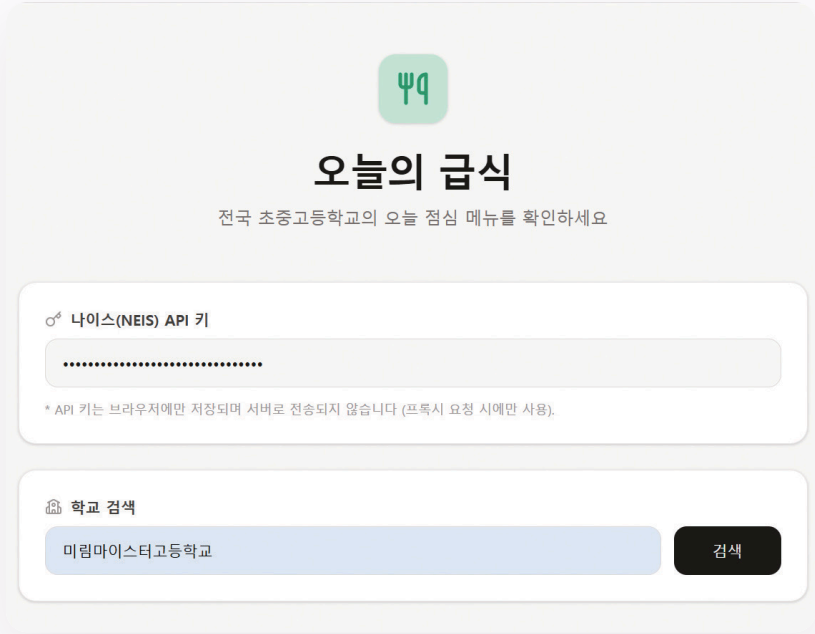
aistudio.google.com · [Build]

```
전국 초중고등학교의 오늘 급식 메뉴를 보여주는 웹앱을 만들어 줘. 화면 상단에 '나이스 API 키 입력칸'을 만들고, 그 아래에 '학교 이름 검색창'을 만들어 줘. 학교를 검색하면 나이스(NEIS) 학교기본정보 API와 급식식단정보 API를 연결해서 오늘 메뉴(중식)를 예쁜 카드 모양으로 띄워줘.
```

기관 이름과 API 이름을 직접 불러 준 것이 전부다. 코드는 한 줄도 쓰지 않았다.

실습 순서를 지킬 것

키를 위 칸에 붙여넣고, 학교 이름을 넣고, 검색을 누른다. 키 없이 검색하면 아무것도 나오지 않는다.



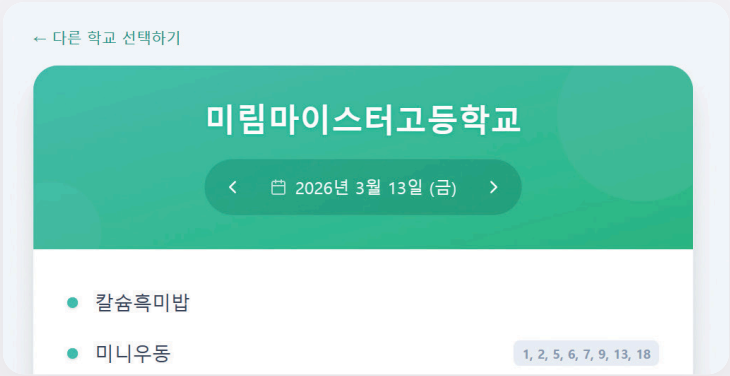
미리보기에 API 키 입력칸과 검색창이 실시간으로 만들어진다.

PART 02 · 4장

완성이 아니라 개선이 바이브 코딩의 본체다

세부 기능을 완성해 나가는 과정이 '코드를 작성하는 일'에서 '문제를 정확히 설명하는 일'로 바뀐다.

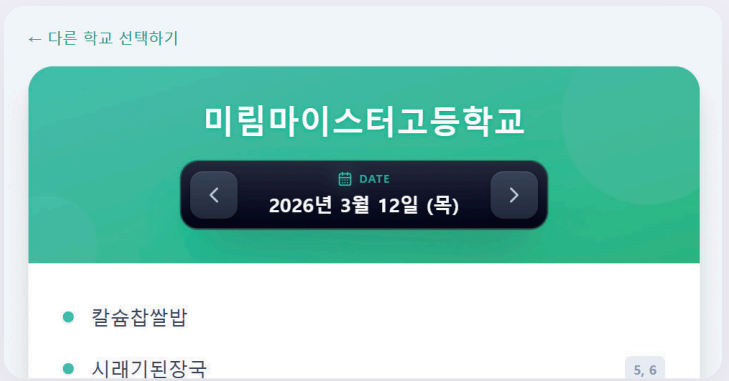
처음 나온 앱



메뉴마다 알레르기 번호를 붙이고, 날짜 이동 화살표까지 달았다. 다만 날짜 바가 눈에 들어오지 않는다.

개선

세 번 고친 앱



Edit tool로 날짜 바만 집어 다시 주문했다. "버튼을 눌러 조작하는 UI라는 인상이 분명하게 보이도록 해줘."

RECAP · 4장

프롬프트에 세 가지만 넣으면 API 코드는 AI가 쓴다

1. **API 주소(엔드포인트)**
   데이터를 주고받을 정확한 목적지 주소를 알려줍니다.

2. **API 인증 키**
   서버의 문을 열 수 있는 디지털 출입증을 함께 제공합니다.

3. **요청 목적**
   단순히 확인만 할 것인지, 새로운 정보를 기록할 것인지 일상적인 언어로 명확히 설명합니다.

메서드	식당에서라면
GET 조회	“우리 테이블 주문 내역 좀 보여 주세요.”
POST 생성	“새로운 메뉴를 주문할게요.”

```
API 명세서를 참고하여 학생 명단 API 주소와
발급받은 API 키를 사용해서, 새로운 학생
데이터를 POST 방식으로 추가하는 기능을 만들어
주세요.
```

PART 02 · 5장

글자를 모르는 아이는 챗봇에게 질문할 수 없다

실제 교실은 학생들의 표정, 칠판의 판서, 오가는 대화처럼 글자보다 훨씬 풍부한 형태의 정보로 채워져 있습니다.

5장을 여는 질문

키보드 입력이 서툰 유치원생이나 초등학교 저학년 학생들에게 기존의 텍스트 기반 AI 챗봇은 사용하기 어려운 도구입니다. 글자를 알아야만 질문을 던질 수 있기 때문입니다.

PART 02 · 5장

앱에 눈과 두뇌와 입을 하나씩 달아 준다

만들 앱의 이름은 **찰칵! 영어 선생님(Snap & Learn English)**. 교실 물건을 비추면 영어 단어를 소리 내어 알려 준다.

입력 · INPUT

눈

카메라

학생이 궁금한 사물을 화면에 담고 '찰칵' 버튼을 누른다. 기기의 웹캠을 쓴다.

처리 · PROCESS

두뇌

AI 비전

제미나이의 이미지 분석 기능이 사진 속 중심 사물 하나를 판별한다.

출력 · OUTPUT

입

음성 합성

찾아낸 영어 이름을 화면에 띄우고 원어민 발음으로 읽어 준다.

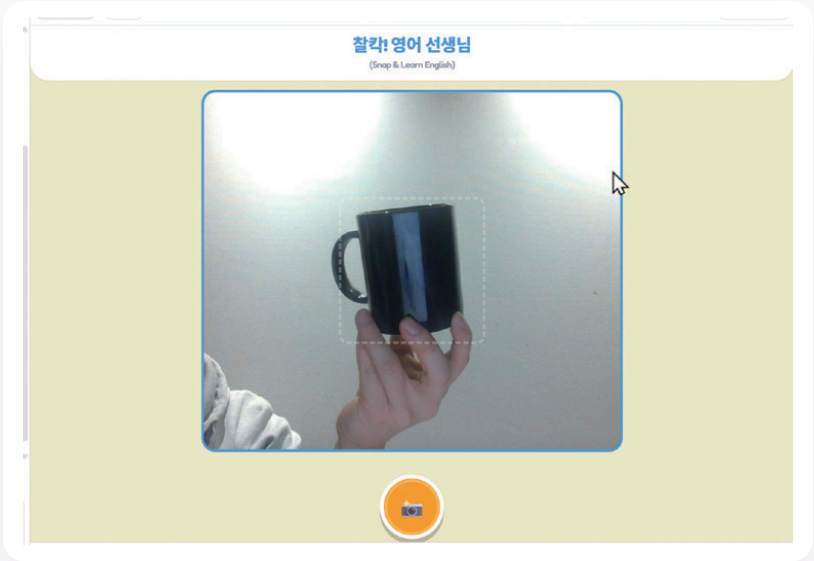
멀티모달

선생님이 학생의 글을 읽고 목소리를 듣고 표정을 보며 아이를 종합적으로 이해하듯, 인공지능도 글자와 사진과 소리를 한 번에 보고 듣는다.

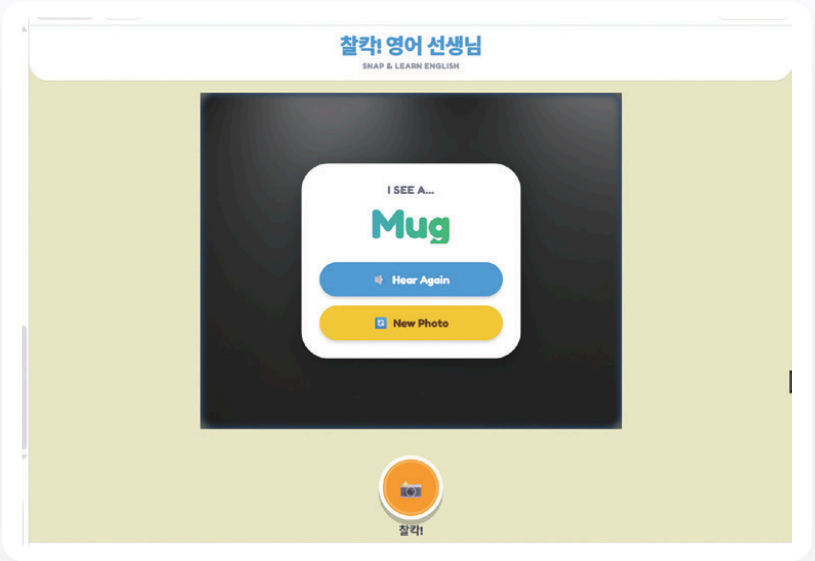
PART 02 · 5장

사물을 비추면 ‘Mug’가 뜨고 소리가 난다

[Build] → [Allow Camera access] → [찰칵]. 세 번 누르면 실행까지 끝난다.



웹캠 화면 가운데에 물건을 두고 아래 ‘찰칵’ 버튼을 누른다.



단어가 뜨는 동시에 원어민 발음이 흘러나온다. [Hear Again]을 누르면 다시 읽어 준다.

PART 02 · 5장

지시문 한 줄만 바꾸면 전혀 다른 앱이 된다

뼈대는 그대로다. 카메라로 찍고, 분석하고, 소리로 읽어 준다. 바뀐 것은 **무엇을 찾을 것인가** 한 줄뿐이다.

확장 ① 두근두근 우리 반 발표자 뽑기

```
사진 속에 있는 학생들 중에서 무작위로 딱 한 명을
선택해 줘. 그리고 그 학생의 인상착의를 한국어로
묘사해 줘.
```

**"회색 후드티를 입은
오늘의 주인공 당첨!"**

한 줄

확장 ② How do you feel?

```
화면 속 인물의 표정을 분석해 줘. 그 표정에 가장 잘
어울리는 영어 감정 단어(Happy, Sad, Angry, Surprised
등)를 딱 하나만 알려줘.
```

"YOU LOOK... Happy"

PART 02 · 5장

키를 만들려면 프로젝트부터 있어야 한다

1

Get API Key

aistudio.google.com 첫 화면 좌측 하단 아이콘.

2

[API 키 만들기]

우측 상단 버튼. 여기서 막히는 분이 많다.

3

[+프로젝트 만들기]

이름을 'Gemini Project'로 적고 만든다.

4

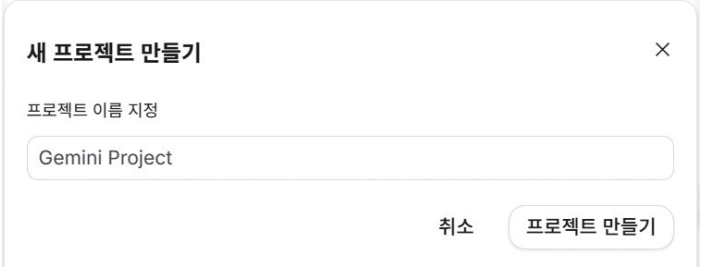
[키 만들기]

영문과 숫자가 섞인 긴 문자열이 뜬다. 복사해 둔다.

왜 버튼이 눌리지 않는가

처음 접속한 상태에서는 인증 키를 담아 둘 **프로젝트**가 아직 없어서 키 발급 버튼이 활성화되지 않는다. 프로젝트명은 아직 영문만 지원한다.

여기서 받는 키는 앞서 발급받은 **나이스 인증 키와 다른 키**다. 둘을 섞어 쓰면 아무것도 동작하지 않는다.

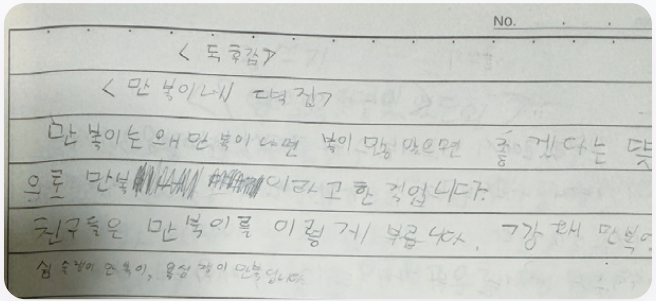


PART 02 · 5장

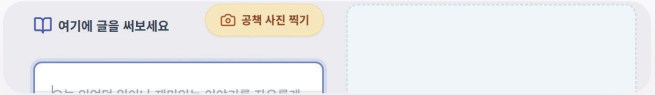
삐뚤빼뚤한 손글씨도 AI는 읽는다

비전이 사진 속에 **무엇이 있는지** 본다면, OCR은 사진 속 **이미지 형태의 글자**를 디지털 텍스트로 한 글자씩 옮겨 적는다.

① 공책 사진 찍기



② 입력창 위에 버튼이 생긴다



프롬프트 한 문단으로 [공책 사진 찍기] 버튼이 붙는다.

③ 텍스트로 자동 입력 → 선생님의 피드백 편지



잘한 점 1가지 · 고칠 점 1가지 · 수정된 문장 추천, 세 가지 형식으로 돌아온다.

PART 02 · 5장

시스템 프롬프트는 AI에게 주는 배역 설정서다

동일한 학생의 글이라도 이 설정에 따라 인공지능의 반응이 크게 달라진다.

배역 A

```
너는 초등학교 4학년 학생들을 가르치는 다정한
선생님이야.
```

학생의 감정에 공감하며 격려하는 피드백을 준다. 칭찬할 점을 먼저 찾고, 고칠 점은 한 가지만 부드럽게 짚는다.

같은 글

배역 B

```
너는 엄격한 맞춤법 선생님이야.
```

오탈자나 띄어쓰기 오류를 정확하게 짚어내는 데 집중한다. 같은 글에서 전혀 다른 목록이 나온다.

메타 프롬프팅

배역 설정서가 투박하면 다듬는 일까지 AI에게 맡긴다. 역할·임무·실행 절차·제약 조건에 맞춰 다시 써 달라고 요청하는 방식이다.

RECAP · PART 02

프로토타입과 서비스 사이에는 보안·과금·개인정보가 있다

보안

소스 보기로 키가 보인다

브라우저에서 작동하는 프론트엔드 앱은 '소스 보기'만으로 코드 안의 API 인증 키가 노출될 위험이 있습니다.

과금

첫 호출부터 비용이 붙는다

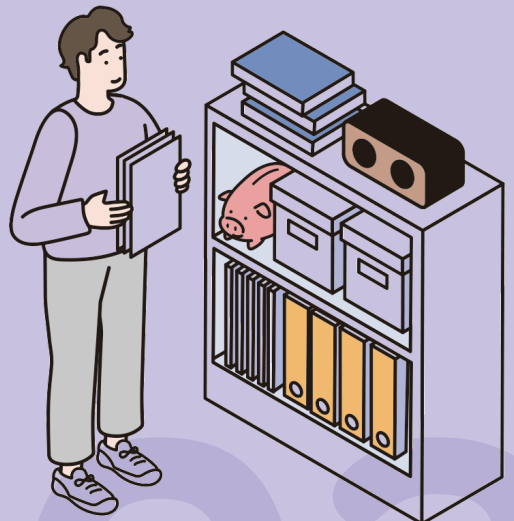
2026년 4월부터 Pro 모델의 무료 등급 제공이 전면 종료되어 첫 호출부터 비용이 발생합니다.

개인정보

입력이 학습에 쓰일 수 있다

무료 등급은 요청 데이터가 제품 개선에 사용될 수 있습니다. 얼굴 사진·실명·전화번호는 넣지 않도록 지도해야 합니다.

SaaS · 소프트웨어	PaaS · 플랫폼	BaaS · 백엔드
완성된 요리를 식당에서 사 먹기	세팅된 상업용 주방 빌리기	다듬어진 채소와 만능 소스로 내 요리 완성하기
Google Classroom, Zoom	**Google App Engine, Heroku**	**Firebase, Supabase**



Part 03

앱에 클라우드 기능을 심다

Preview 클라우드로 나의 앱에 날개 달기

06 왜 데이터베이스가 필요할까요?

PART 03 · 6장

새로고침 한 번에 사라지는 앱은 아직 서비스가 아니다

앱을 종료하거나 새로고침하면 지금까지 입력한 데이터가 모두 사라집니다. 모래사장에 글씨를 쓰는 것과 같습니다. 작성할 때는 분명히 보이지만, 파도가 한 번 지나가면 흔적 없이 사라집니다.

PART 03 · 6장

데이터베이스는 교무실 캐비닛이고, 그 속은 엑셀과 똑같이 생겼다

비유 · 교무실 캐비닛

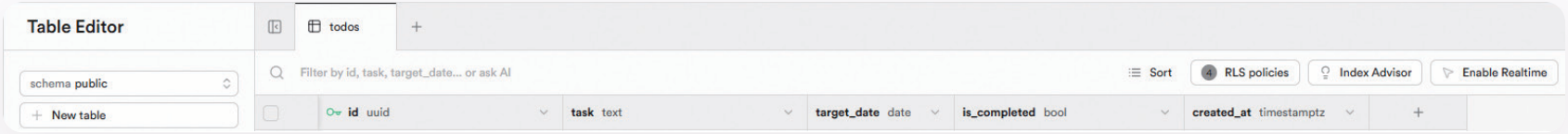
정해진 규칙에 따라 보관된다

생활기록부, 성적표, 출석부가 규칙대로 꽂혀 있어 필요할 때 원하는 정보를 빠르게 찾을 수 있다. 데이터베이스는 **사라지는 임시 기억이 아니라, 오랫동안 보존할 수 있는 영구적인 저장 공간**이다.

구조 · 엑셀과 같다

파일이 DB, 시트가 테이블

행(Row)에는 각각의 데이터가 저장되고, 열(Column)에는 데이터의 종류가 정리된다. 계산기 같은 일회성 도구는 필요 없지만, **기록을 기억해야 하는 순간부터** 이 표가 필요해진다.



Supabase Table Editor의 열 구조. 열(Column)이 데이터의 종류를 정한다 — task는 글자, target_date는 날짜, is_completed는 참·거짓.

PART 03 · 6장

복잡해 보여도 데이터베이스는 네 가지 주문만 외운다

컴퓨터 과학에서는 네 동작의 앞 글자를 따서 CRUD(크루드)라고 부른다.

C

Create 생성

새로운 정보를 서랍장에 추가한다.

신규 전학생의 정보를 명부에 새로 등록하기

SQL · INSERT

R

Read 읽기

이미 저장된 정보를 화면에 불러온다.

3학년 2반 학생들의 이름만 명부에서 쭉 뽑아오기

SQL · SELECT

U

Update 수정

변경된 내용을 찾아 새 값으로 바꾼다.

번호가 바뀐 학생의 연락처 정보만 지우고 새로 적기

SQL · UPDATE

D

Delete 삭제

더 이상 필요 없는 정보를 안전하게 파기한다.

졸업생 또는 전학 간 학생의 데이터를 삭제하기

SQL · DELETE

이 CRUD의 원리만 이해하면, 상상하는 어떤 교육용 앱 기능이든 데이터베이스로 구현할 수 있다.

PART 03 · 6장

백엔드는 이제 짓는 것이 아니라 빌려 쓰는 것이다

과거의 백엔드

서버부터 직접 지었다

- 물리적인 서버 컴퓨터를 구매하거나 임대한다
- 리눅스 명령어가 가득한 검은 화면과 씨름한다
- 시스템을 직접 설치하고 유지보수한다

지금

BaaS · Backend as a Service

완비된 공유 주방을 빌린다

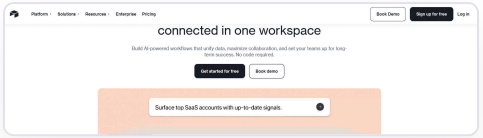
- 회원가입, 데이터베이스, 이미지 저장소 등 복잡한 뒷단(Backend) 기능을 클릭 몇 번으로 가져다 쓴다
- 냉장고를 사고 가스 배관을 설치할 필요가 없다
- 복잡한 서버 구축 없이 실제 서비스를 만든다

책의 비유

식당을 열고 싶은 사장님은 주방(BaaS)을 빌려 곧바로 맛있는 요리(프론트엔드 화면)를 만드는 데만 집중하면 된다.

PART 03 · 6장

Airtable · Firebase · Supabase, 셋의 차이는 성능이 아니라 용도다

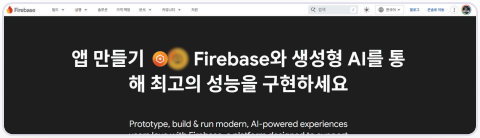


에어테이블 Airtable

자유자재로 변신하는 마법의 엑셀

직관적인 데이터 입력, 달력 및 칸반 보드 등 훌륭한 시각화 기능 제공, 코딩 없는 자동화 연동

이런 선생님께 — 코딩 없이 빠르고 직관적으로 간단한 학급 관리 도구를 만들고 싶은 선생님

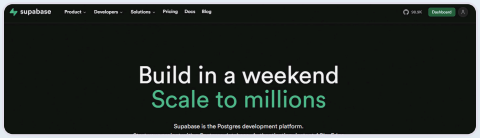


파이어베이스 Firebase

구글이 준비한 백엔드 선물 세트

화면 실시간 동기화, 회원가입과 사진 보관함 등 앱 개발에 필요한 필수 기능 통합 제공

이런 선생님께 — 우리 반 익명 채팅방이나 실시간 투표 앱처럼 역동적인 기능이 필요한 선생님



슈파베이스 Supabase

철저하고 똑똑한 만능 교무부장

엑셀과 유사한 직관적 구조, 강력한 관계형 데이터 연결, 완벽한 보안 통제 및 확장성

이런 선생님께 — 데이터를 체계적으로 연결하고, 장기적으로 완성도 높은 맞춤형 웹 앱을 기획하시는 선생님

PART 03 · 6장

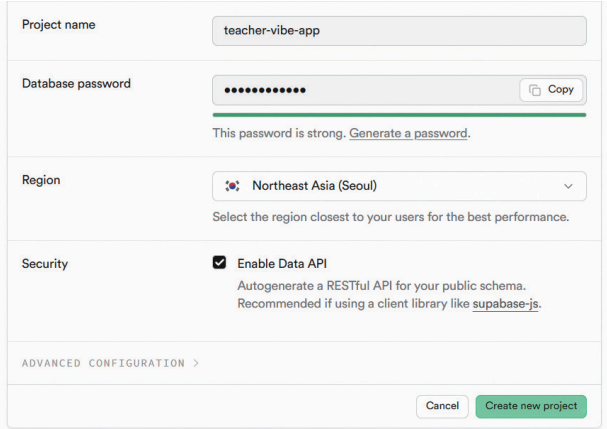
데이터 본부부터 세운다 — 마스터 키와 Seoul을 잊지 말 것

1. supabase.com에 접속해 로그인하고 우측 상단 **[Start your project]**를 클릭한다.
2. 프로젝트 목록에서 **[+ New Project]**를 클릭한다.
3. 이름을 **teacher-vibe-app**으로 짓고 비밀번호와 리전을 정한 뒤 **[Create new project]**를 클릭한다.

여기서 놓치면 되돌리기 어렵다

Database password는 단순한 로그인 비밀번호가 아니라 데이터베이스의 핵심(PostgreSQL)에 접근하는 '마스터 키'다. 메모장이나 비밀번호 관리자에 꼭 적어 둔다.

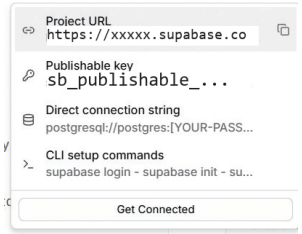
Region은 Seoul. 한국에서 접속할 때 데이터를 주고받는 속도가 가장 빠르다.



Create a new project — Project name, Database password, Region: Northeast Asia (Seoul). 무료 요금제로 프로젝트 2개까지 운영할 수 있다.

PART 03 · 6장

프론트엔드가 백엔드로 들어가려면 출입증 두 장이 필요하다



'Project Settings' → [API Keys] → [Create new API keys], 그다음 'Project overview'에서 [copy].

1

Project URL

```
https://xxxxx.supabase.co
```

우리 데이터베이스의 정확한 웹상 '위치(주소)'를 의미한다.

2

Publishable Key

```
sb_publishable... / eyJhbG...
```

아주 긴 문자열. 이것이 바로 출입증 역할을 하는 '키(Key)'다.

공개 키가 맞다

'anon public' 키는 말 그대로 공개(public) 키다. 프론트엔드 코드에 직접 넣어도 되도록 설계되었다.

단, 조건이 붙는다

RLS를 설정하지 않으면 누구든 프로젝트에 저장된 데이터를 조회·수정·삭제할 수 있다. **반드시 RLS(Row Level Security)와 함께 써야** 안전하다.

PART 03 · 6장

프롬프트 한 장에 화면·테이블·CRUD를 한꺼번에 시킨다

```
당신은 친절한 '풀스택 웹 개발자'입니다.
"날짜별 투두(To-Do) 리스트"를 관리할 수 있는
일정 관리 웹앱을 제작해 주세요.
1. Supabase 설정 정보
- Project URL: https://xxxxx.supabase.co
- Publishable key: sb_publishable..
2. 프론트엔드 요구사항(화면 구성)
- 할 일 내용(Text), 목표 날짜(Date Picker)
- '추가하기' 버튼, 날짜별로 그룹화한 카드 목록
3. 백엔드 기능 요구사항(데이터 처리)
- 테이블 생성 쿼리를 setup.sql로 작성해 주세요.
```



구글 AI Studio 입력창에 붙여넣고 [Build]를 클릭한다.

AI가 알아서 하는 세 가지

1. **Supabase와 연결하기**
   메모장에 적어 둔 출입증(Key)을 코드에 넣어 본부의 문을 열어 준다.

2. **테이블 설계도 만들기**
   데이터를 담을 그릇(테이블)을 만드는 명령어(SQL 쿼리)를 대신 써 준다.

3. **CRUD 자동 연결**
   추가·읽기·수정·삭제를 버튼에 연결해 준다.

쿼리(Query)는 데이터베이스에 내리는 '명령어'다.

PART 03 · 6장

AI가 써 준 설계도를 붙여넣고 Run — 테이블로 가는 길은 두 갈래다

추천 · SQL 코드로 한 번에

SQL Editor에 붙여넣고 [Run]

- 좌측 **[SQL Editor]** 클릭
- setup.sql을 붙여넣고 **[Run]** → 'Success!'
- 'Table Editor'에서 todos 테이블 확인

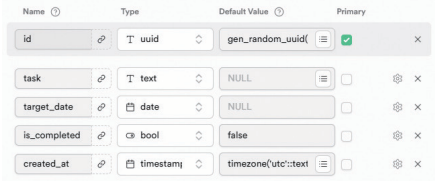
```
create table if not exists public.todos (
  id uuid default gen_random_uuid() primary key,
  task text not null,
  target_date date not null,
  is_completed boolean default false not null
);
```

또는

대안 · 마우스 클릭으로

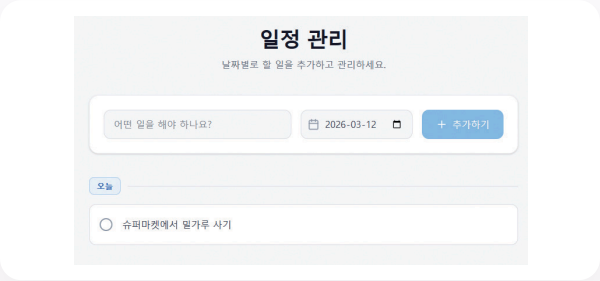
Table Editor에서 직접 만들기

- 'Table Editor' → **[+ New Table]**, Name은 todos
- **[Add column]**으로 task(text) · target_date(date) · is_completed(bool, false) 추가
- **[save]** 클릭. id와 created_at은 건드리지 않는다



PART 03 · 6장

클라우드에 남는 순간, 프로그램은 서비스가 된다

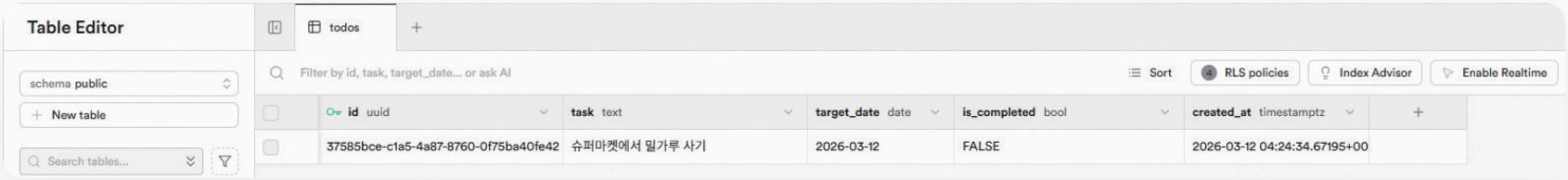


① 구글 AI Studio 미리보기에서 '슈퍼마켓에서 밀가루 사기'를 입력하고 [+ 추가하기].

여기가 분기점이다

새로고침을 해도, 다른 컴퓨터에서 접속해도 이 데이터는 안전하게 보관된다.

버튼을 눌렀는데 화면만 그대로라면 '동기화' 로직이 빠진 것이다. **"저장 직후 자동으로 목록을 다시 불러와 최신 데이터가 보이도록 해줘"**라고 한 줄만 더 요청하면 된다.



② 같은 문장이 Supabase의 todos 테이블에 행 하나로 들어와 있다. target_date 2026-03-12, is_completed FALSE.

PART 03 · 6장

지금 이 앱은 문이 열린 금고다

주소만 알면 누구나 들어와서 남의 할 일을 보고, 심지어 지울 수도 있습니다. 화면 코드에서 비밀번호를 검사하는 건 답이 아닙니다. 브라우저로 보낸 코드는 개발자 도구로 훤히 들여다보입니다.

DON'T 01

지금은 구글 AI Studio 안에서만 테스트합니다

DON'T 02

배포는 7장의 로컬 환경·환경변수·RLS를 적용한 뒤에

DON'T 03

테스트가 끝나면 todos 테이블의 행을 Delete로 지웁니다

RECAP · PART 03

RLS는 금고실 문이 아니라 상자마다 채우는 자물쇠다

문만 통과하면 남의 상자까지 열리던 방식을, 상자마다 주인을 확인하는 방식으로 바꾼 것입니다.

```
-- 처음 setup.sql : 누구나
create policy "Allow public read access"
  on public.todos for select
  using (true);

-- RLS 강화 : 인증된 사용자만
create policy "Users can view their own todos"
  on public.todos for select
  to authenticated
  using (true);
```

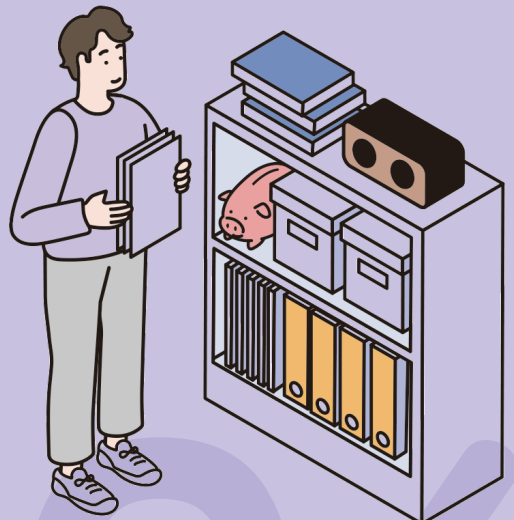
옛 정책은 drop policy if exists로 먼저 지웁니다. 'Authentication' → [Users] → [Add user]에서 admin@planner.vibe 계정을 등록하면, 앱 실행 시 로그인 모달이 먼저 뜹니다.

여기까지 답할 수 있습니까 — PART 03

1. **앱이 데이터를 기억하게 하려면 무엇이 필요합니까?**
   클라우드 데이터베이스입니다. 그 동작은 CRUD 네 가지가 전부입니다.

2. **백엔드를 직접 짓지 않는 방식을 뭐라 부릅니까?**
   BaaS입니다. Supabase는 그중 오픈소스 관계형 데이터베이스입니다.

3. **키만 넣은 앱을 그대로 배포해도 됩니까?**
   안 됩니다. 행마다 자물쇠를 채운 뒤에야 남에게 건넬 수 있습니다.



Part 04

로컬 환경에서 진짜 개발자처럼 일하기

Preview 공유 오피스를 떠나, 나만의 개발 스튜디오로

07 내 컴퓨터를 React 개발 서버로 만들기

08 개발부터 배포까지, 내 손으로 완성하기

PART 04 · OPENING

**“인터넷이 끊기면 어떡하지? 서비스가 유료로 바뀌면?
내가 만든 게 진짜 내 것이 맞나?”**

편리한 공유 오피스에는 내 이름표가 없다

도구가 다 갖춰져 있어 누구나 쉽게 시작할 수 있지만, 작업 공간과 규칙은 서비스가 정합니다. 이제 작업실을 내 컴퓨터로 옮깁니다.

PART 04 · 7장

로컬로 내려와야 하는 이유는 딱 세 가지다

로컬 개발 환경은 인터넷이나 특정 서비스에 의존하지 않고 프로젝트를 자유롭게 관리하는 자리다.

REASON 01

라이브러리 자유

전 세계 개발자들이 만든 수백만 개의 오픈소스 라이브러리를 자유롭게 설치하고 활용할 수 있다.

REASON 02

인터넷 독립

인터넷이 끊겨도, 서비스가 점검 중이어도, 해외 출장 중 비행기 안에서도 코드를 작성하고 실행할 수 있다.

REASON 03

버전 관리

Git이라는 버전 관리 도구로 코드의 모든 변경 이력을 저장한다. 게임의 세이브 포인트와 같다.

한 번 넘으면 되돌아가지 않는다

처음엔 낯설다. 그런데 한 번 손에 익으면 다시는 공유 오피스로 돌아가고 싶지 않다.

PART 04 · 7장

IDE는 코딩을 위한 최신식 주방이다

칼, 도마, 오븐, 레시피 북이 한곳에 있어야 요리사가 요리에만 집중할 수 있다.

1장에서 쓴 도구

메모장

- 글자를 저장하는 일만 한다
- 코드를 실행하려면 창을 바꿔 브라우저로 가야 한다
- 오타를 알려 주지 않는다. 안 열리고 나서야 안다

VS

지금부터 쓸 도구

IDE · 통합 개발 환경

- 코드 작성, 수정, 실행을 한 화면에서 처리한다
- 파일 트리, 편집 창, 터미널, AI가 한 창에 모여 있다
- Antigravity · Cursor · Windsurf는 모두 VS Code 계보라, 하나를 익히면 나머지도 통한다

그래서 이 파트는 도구를 하나로 고정한다

실습은 Antigravity IDE 하나로만 진행한다. Cursor와 Claude Code는 부록에서 다시 만난다.

PART 04 · 7장

Antigravity는 코드를 제안하지 않고 직접 쓴다

단순한 AI 코드 도우미를 넘어, 여러 AI 에이전트를 동시에 운용하는 Agent-First IDE다.

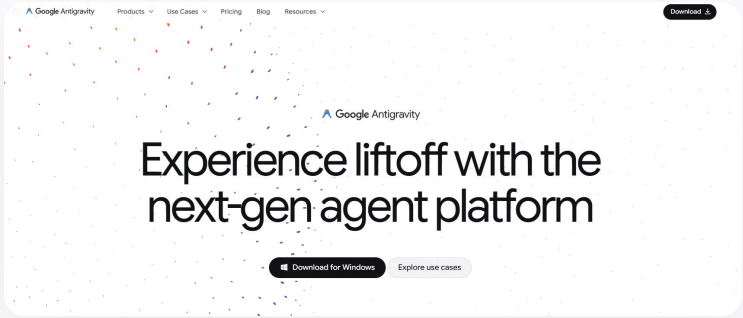
AGENT · 에이전트

AI 주니어 개발자

에이전트가 지시를 이해하고, **파일을 직접 만들고, 코드를 작성하고, 오류까지 스스로 고친다.** 자동완성이 아니라 옆자리에서 일하는 동료에 가깝다.

이름의 뜻

'반중력(Anti-Gravity)'처럼, 개발의 무거운 부담을 날려버리겠다는 의지가 담겨 있다.



antigravity.google — 제품 페이지에서 바로 내려받는다.

출시	2025년 11월 18일, Google 공식 출시
2.0 개편	2026년 5월 Google I/O — 에이전트 운용 독립 앱과 코드 편집 IDE로 분리. 이 실습 기준은 Antigravity IDE
요금	개인 사용자는 무료로 시작

PART 04 · 7장

‘Add to PATH’를 놓치면 터미널에서 IDE를 못 연다

STEP 01

사전 점검

Google 계정 · 인터넷 · 운영체제 (Mac은 칩 종류까지) · 저장 공간 500MB 이상

STEP 02

내려받기

antigravity.google/product/antigravity-ide에서 [Antigravity IDE] 탭 → 내 OS 파일

STEP 03

설치

동의 체크 → 설치 경로 지정 → 추가 작업 선택에서 **‘Add to PATH’ 체크 확인** → [설치]

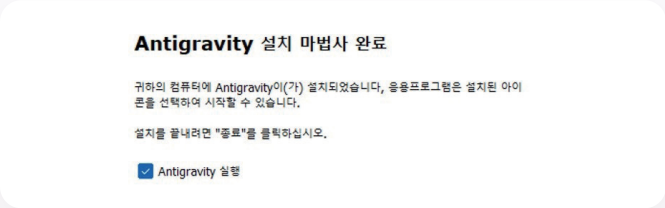
STEP 04

첫 실행

[Continue with Google] 로그인 → Ctrl+Shift+P → Configure Display Language → [한국어(ko)] → Restart

여기서만 되돌릴 수 없다

이 옵션을 체크하지 않으면 나중에 터미널에서 Antigravity IDE를 실행할 수 없다. Mac은 아이콘을 Applications 폴더로 드래그하고, ‘개발자를 확인할 수 없습니다’ 경고가 뜨면 [개인 정보 보호 및 보안]에서 [허용]을 누른다.

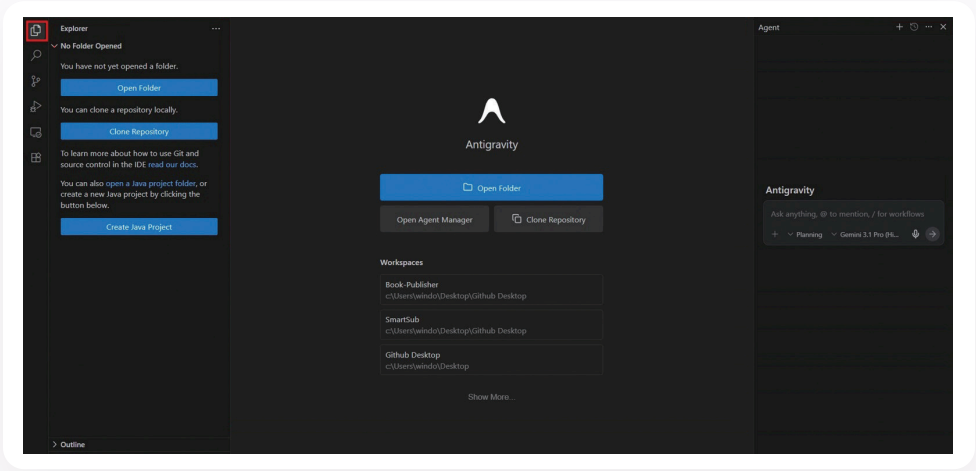


설치 마지막 화면 — [Antigravity 실행] 체크

PART 04 · 7장

실제로 쓰는 영역은 딱 두 곳이다

처음 열면 파일 트리 대신 우측 패널이 먼저 보일 수 있다. '이게 뭐지?' 하고 당황하지 않아도 된다.



Antigravity IDE 최초 화면. 폴더를 열기 전에도 우측 에이전트 패널은 이미 열려 있다.

1 Editor View — 좌측과 가운데

화면 좌측에는 파일 트리, 가운데에는 코드 편집 영역이 놓인다. 어떤 파일이 있고 지금 무엇을 고치는 중인지를 보는 자리다.

2 에이전트 패널 — 우측

여기에 작업 지시를 입력하면 AI가 코드를 자동으로 작성한다. **이 우측 패널이 Antigravity IDE의 핵심이다.**

단축키 하나만 외운다

패널이 안 보이면 Ctrl + L(Mac은 Cmd + L). 여러 AI 직원에게 업무를 나눠 주는 관제탑이라고 생각하면 된다.

PART 04 · 7장

첫 프로젝트는 문장 하나로 시작한다

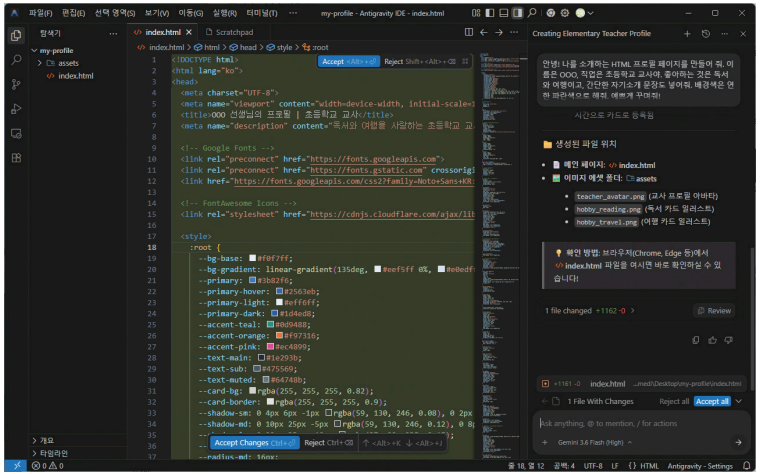
바탕화면에 빈 폴더 하나와 빈 파일 하나. 나머지는 에이전트에게 말로 시킨다.

```
안녕! 나를 소개하는 HTML 프로필 페이지를 만들어 줘. 이름은
[여기에 본인 이름 입력], 직업은 초등학교 교사야. 좋아하는
것은 독서와 여행이고, 간단한 자기소개 문장도 넣어줘.
배경색은 연한 파란색으로 해줘. 예쁘게 꾸며줘!
```

1. 바탕화면에 **my-profile** 폴더를 만들고 [폴더 열기...]로 연다
2. 탐색기 빈 곳을 우클릭해 [새 파일...] → **index.html**
3. Ctrl + L로 에이전트 패널에 위 문장을 붙여넣는다
4. [Accept all] → Ctrl + S → 파일을 더블 클릭

폴더 이름 규칙

영어 소문자로, 띄어쓰기 없이. 한글 폴더 이름은 나중에 오류를 일으킨다.



문장 하나에 코드 1,162줄이 생겼다. 우측 [Accept all]을 누르기 전까지는 아무것도 확정되지 않는다.

PART 04 · 7장

localhost는 아파트 이름이고, 포트는 호수다

주소(동)가 같더라도 호수(호)가 다르면 다른 집이다.

`http://localhost:5173`

localhost — 아파트 이름. 인터넷이 아니라 내 컴퓨터 안에서만 돌아가는 개발 공간이다.

5173 — 호수. 내 컴퓨터에서 실행 중인 서비스 번호다. localhost:3000은 "내 컴퓨터의 3000번 호에 있는 서비스에 연결해 줘."라는 뜻이다.

127.0.0.1은 같은 집이다

모든 컴퓨터가 자기 자신을 가리킬 때 쓰는 루프백 주소다.
127.0.0.1:3000과 localhost:3000은 같은 서버를 가리킨다.

포트	사용하는 서비스
3000	React 개발 서버
8080	다른 웹 서버
5173	Vite 개발 서버
5432	PostgreSQL 데이터베이스

호수가 겹치면

"포트 3000이 이미 사용 중입니다(Port 3000 is already in use)"가 뜬다. 기존에 실행 중인 프로그램을 종료하거나, 개발 서버를 다른 포트(3001번)로 실행하면 된다.

PART 04 · 7장

로컬은 비밀 주방, 배포는 정식 오픈이다

유명 레스토랑에는 손님이 들어올 수 없는 비밀 주방이 있다.

LOCAL · 로컬

비밀 주방

내가 만든 앱을 테스트하고 수정하는 개발 공간으로, 기본적으로 내 컴퓨터에서만 접속할 수 있다. 셰프들은 이곳에서 새 메뉴를 만들고, 실패를 반복해도 부담 없이 실험한다.

비밀 주방에서 요리를 망쳐도 손님이 모른다. 로컬에서 코드를 잘못 써도 전 세계 사용자에게는 아무 영향이 없다.

VS

REMOTE · 리모트(웹 서버)

정식 오픈한 레스토랑

개발이 완료된 서비스를 웹 서버에 올리면, URL만 알면 누구나 접속하여 사용할 수 있는 상태가 된다. 정식으로 문을 연 레스토랑에는 누구나 방문해 같은 메뉴를 즐긴다.

공개되는 순간 실수도 함께 공개된다. 그래서 주방에서 충분히 굽고 나서 문을 연다.

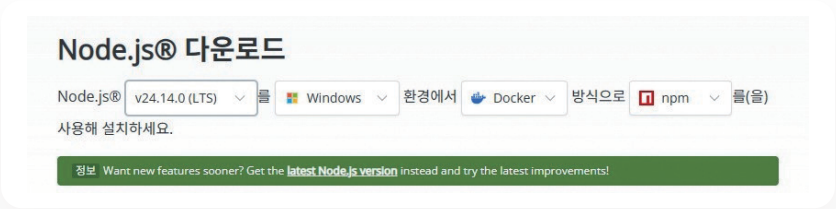
7장에서 비밀 주방을 차리고, 8장에서 정식 오픈까지 간다. 순서를 바꾸면 손님 앞에서 요리를 망친다.

PART 04 · 7장

Node.js는 공구함, npm은 앱 마켓, Vite는 모델하우스다

NODE.JS · 공구함

React 프로젝트를 만들고, 패키지를 설치하고, 개발 서버를 돌리는 모든 작업이 Node.js 위에서 이루어진다. 공구함이 없으면 나머지 도구를 꺼낼 수 없다.



nodejs.org에서 상단의 LTS 버전을 고른다. LTS는 Long Term Support의 약자로, 안정성이 검증된 버전이다.

NPM · 앱 마켓

필요한 부품을 받아 오는 곳

'Node Package Manager'의 줄임말. 새 스마트폰을 사면 앱을 따로 설치하듯, 프로젝트도 **npm install**로 필요한 패키지를 받는다.

VITE · 모델하우스

기둥과 벽이 이미 서 있는 집

아무것도 없는 빈 땅에서 시작하지 않는다. 예전에 쓰던 create-react-app은 설치도 서버 실행도 느렸고, Vite가 그 문제를 획기적으로 해결했다.

PART 04 · 7장

명령어 네 줄이면 내 컴퓨터가 개발 서버가 된다

`npm create vite@latest`

프로젝트 틀을 만든다. Project name my-first-app · framework React · variant JavaScript

`cd my-first-app`

cd는 Change Directory, 즉 '방 바꾸기'다. 방금 만든 폴더 안으로 들어간다

`npm install`

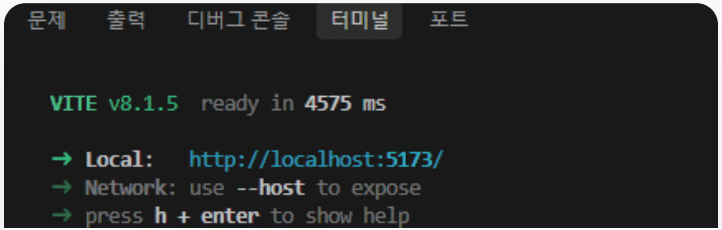
필요한 패키지를 받는다. 인터넷 속도에 따라 30초에서 1~2분. node_modules 폴더가 생긴다

`npm run dev`

자동차도 시동을 걸어야 움직인다. package.json에 미리 적어 둔 개발 서버 실행 단축 명령이다

```
VITE ready

→ Local: http://localhost:5173/
→ Network: use --host to expose
→ press h + enter to show help
```



실제 IDE 터미널. 찍히는 버전과 소요 시간은 설치 시점에 따라 달라진다.

PART 04 · 7장

저장하는 순간 화면이 바뀐다

Hot Reload — 코드를 수정하고 저장하는 순간, 브라우저가 자동으로 새로고침되어 변경 내용을 즉시 반영한다.

1. 브라우저(localhost:5173)와 IDE 창을 나란히 배치한다
2. 좌측 탐색기에서 **src 폴더 → App.jsx**
3. Ctrl + F로 **Vite + React**를 검색해 노란 강조를 찾는다
4. 텍스트를 바꾸고 **Ctrl + S**. 새로고침은 누르지 않는다

왜 이런 게 생겼나

예전에는 수정할 때마다 저장하고, 브라우저로 전환해서 새로고침을 눌렀다. 이 과정이 하루에 수백 번 반복되다 보니 개발자들이 아예 자동화해 버린 것이다.

BEFORE · Vite가 만들어 준 기본 화면



AFTER · 저장만 눌렀다



App.jsx 19행의 글자만 바꿨다. 나머지 화면은 그대로다.

RECAP · 7장

오류는 실패가 아니라 예정된 순서다

처음 환경을 세팅할 때는 예상치 못한 오류가 나오는 게 오히려 자연스러운 일입니다.

1. **포트가 이미 사용 중이라고 나옵니다 — `Error: listen EADDRINUSE: address already in use :::5173`**
   터미널에서 Ctrl + C로 실행 중인 서버를 종료하거나, 다른 터미널 창에서 실행 중인 npm run dev가 있는지 확인합니다.

2. **node -v를 입력했더니 버전이 너무 낮습니다(v14 이하)**
   React와 Vite의 최신 버전들은 Node.js v16 이상을 요구합니다. nodejs.org에서 최신 LTS를 덮어쓰기로 다시 설치합니다.

3. **npm install 중에 오류가 납니다**
   대부분 인터넷 연결 문제이거나 npm 캐시 문제입니다. `npm cache clean --force` 뒤에 다시 `npm install`.

4. **브라우저에서 localhost:5173이 안 열립니다**
   npm run dev가 실행 중인지 확인합니다. 백신·방화벽이 포트를 막을 수 있고, 주소창에 'http://'를 붙였는지도 봅니다.

그래도 안 되면 — 최후 수단

에이전트 패널(Ctrl + L)에 오류 메시지를 그대로 붙여넣고 "이 오류가 왜 나는지 설명하고 해결 방법을 알려줘."라고 물어봅니다.

PART 04 · 8장

‘진짜최종_이게맞음.hwp’는 버전 관리가 아니다

```
assignment_final.hwp
assignment_final2.hwp
assignment_final_진짜최종.hwp
assignment_final_진짜최종2.hwp
assignment_final_진짜진짜최종_이게맞음.hwp
```

어제까지 잘 작동하던 앱이 오늘 수정하다가 망가졌을 때, Git이 있으면 ‘어제 버전으로 돌아가기’ 클릭 한 번이면 끝납니다.

PART 04 · 8장

Git은 일기장, GitHub는 금고, Desktop은 열쇠꾸러미다

GIT · 내 컴퓨터

일기장

내 컴퓨터에 설치되는 프로그램.
타임머신처럼 코드의 역사를 기록해 "지금 이 상태를 저장해", "3일 전 상태로 돌아가" 같은 명령을 처리한다.

GITHUB · 인터넷 클라우드

금고 겸 박물관

Git으로 관리하는 코드를 인터넷에 올려두는 공간. 구글 드라이브가 일반 파일을 저장하듯, GitHub는 코드와 코드의 변경 이력을 저장한다.

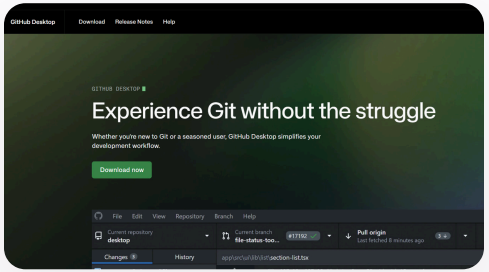
GITHUB DESKTOP · 클릭 몇 번

열쇠꾸러미 앱

원래 Git은 터미널에서 명령어를 입력해야 한다. 그 조작법을 리모컨 버튼처럼 바꿔주는 앱이다. 명령어를 외울 필요는 없다.

폴더에 블랙박스가 달린다

프로젝트 폴더를 Git으로 관리하기 시작하면 보이지 않는 **.git** 폴더가 생긴다. 이 폴더가 모든 변경 이력을 기록한다. 그리고 GitHub 없이는 자동 배포가 없다. Git을 배우는 것은 곧 배포를 위한 준비다.



PART 04 · 8장

커밋은 보스전 직전의 세이브다

프로그램이 정상적으로 동작하는 중요한 시점마다 "지금 이 상태를 기록한다"고 눌러 둔다.

STEP 01

고치고 저장

Antigravity IDE에서 코드를 수정하고 Ctrl + S

STEP 02

변경 확인

[Changes] 탭에 바뀐 파일이 자동으로 뜬다

STEP 03

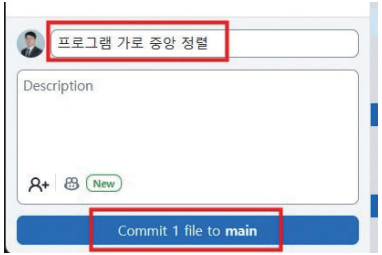
Summary 한 줄

좌측 아래 칸에 무엇을 바꿨는지 적는다

STEP 04

Commit to main

누르는 순간 세이브 포인트가 하나 생긴다



Summary 칸에 한 줄 적고 [Commit 1 file to main]. [Changes] 탭에서 초록은 추가, 빨강은 삭제다.

"수정함"보다 **"로그인 버튼 색상 파란색으로 변경"**이 훨씬 유용하다.

언제 커밋하나

하나의 기능을 완성했을 때, 잘 작동하는 걸 확인했을 때, 큰 수정을 시작하기 전에 보험으로, 오늘 작업을 마무리할 때. 기준을 잡기 어렵다면 1-2시간마다 한 번이라고 생각한다.

PART 04 · 8장

일부러 망가뜨려 봐야 안전망을 믿게 된다

커밋을 두 개 이상 남긴 뒤, App.jsx를 일부러 망가뜨리고 저장한다.

DISCARD CHANGES

아직 커밋하지 않았을 때

[Changes] 탭에서 파일을 우클릭해 [Discard Changes]. 새 커밋이 생기지 않으며, 마지막 커밋 상태로 즉시 돌아간다. 빠르고 간단하다.

VS

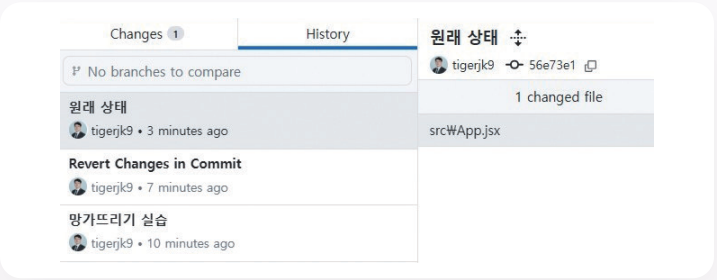
REVERT

이미 커밋한 뒤

[History] 탭에서 대상 커밋을 우클릭해 [Revert Changes in Commit]. **역사를 지우는 게 아니라 '되돌렸다'는 새로운 기록을 남긴다.**

기호 세 줄이 튀어나오면

Git이 현재 코드와 되돌리려는 코드 중 어느 쪽을 쓸지 정하지 못해 선택을 맡긴 것이다. '<<<<<<<'부터 '=======' 까지는 현재 코드, '=======' 부터 '>>>>>>>' 까지는 되돌리려는 코드다. 쓸 쪽만 남기고 기호 줄은 지운다.



PART 04 · 8장

한 줄 고치고 Push했을 뿐인데 세계가 바뀐다

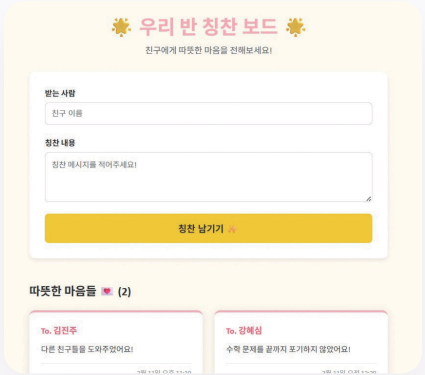
CI/CD — Continuous Integration / Continuous Deployment. 코드를 GitHub에 올리면 새 버전이 저절로 배포된다.

1. **App.jsx 제목을 "전 세계 배포 성공!"으로 고치고 저장**
2. **GitHub Desktop — Summary "앱 제목 수정"**
   [Commit to main] → [Push origin]
3. **Vercel이 변경을 스스로 감지해 다시 빌드한다**
   여기서 사람이 하는 일은 없다. 약 1~2분
4. **북마크한 주소를 새로고침하면 제목이 바뀐다**
   PC뿐 아니라 스마트폰에서도 같은 주소로 열린다

처음 한 번만 설정한다

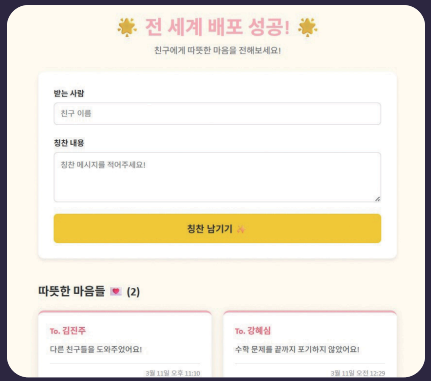
vercel.com에서 [Continue with GitHub] → [Add New] → [Project] → 저장소 Import → 환경변수 두 개 등록 → [Deploy].

BEFORE



배포 직후의 앱

AFTER · Push만 했다



전 세계 어디서든, 같은 주소

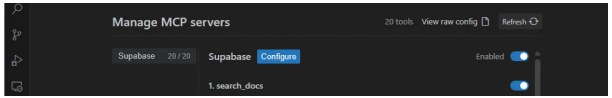
RECAP · PART 04

MCP는 AI를 말하는 존재에서 행동하는 존재로 바꾼 표준이다

MCP · MODEL CONTEXT PROTOCOL

2024년 11월, Anthropic이 처음 공개한 오픈 프로토콜입니다. AI가 파일 시스템, 데이터베이스, GitHub 같은 외부 도구와 표준 방식으로 연결되어 작업할 수 있도록 만든 통신 규약입니다.

TV도, 에어컨도, 셋톱박스도 하나로 제어하는 **만능 리모컨**과 같습니다. 예전에는 AI가 설계도를 그리면 사람이 벽돌을 쌓았지만, 지금은 AI가 벽돌까지 쌓습니다.



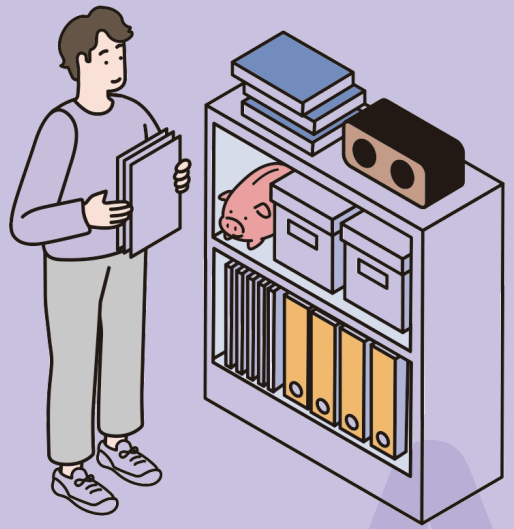
Supabase를 MCP로 연결하자 AI가 도구 20개를 손에 쥐었습니다.

여기까지 답할 수 있습니까 — PART 04

1. **내 컴퓨터에서 개발 서버를 띄우고, 그 주소가 왜 localhost:5173인지 설명할 수 있습니까?**
   아파트 이름과 호수 · 네 줄의 명령어 · 저장하면 바뀌는 화면

2. **커밋과 푸시의 차이를 말하고, 망가진 코드를 되돌릴 수 있습니까?**
   일기장과 금고 · Summary 한 줄 · Discard와 Revert의 갈림길

3. **한 줄 고치고 Push했을 뿐인데 왜 사이트가 저절로 바뀌는지 설명할 수 있습니까?**
   GitHub 없이는 자동 배포가 없다 · CI/CD



부록

그 밖의 인공지능 도구들

Preview 도구는 계속 바뀐다

A 내 컴퓨터 안의 AI 개발 작업실, Cursor

B 터미널에서 일하는 CLI 기반 도구, Claude Code

도구는 계속 바뀐다 — 그래서 부록은 소개와 설치만 다룬다

Antigravity IDE 하나를 끝까지 써 보셨다면, 별도의 상세 설명 없이도 다른 도구를 쓸 준비가 되신 겁니다. 그래서 부록은 가장 많이 쓰이는 **Cursor**와 **Claude Code**를 소개와 설치 방법 위주로만 다룹니다.

도구는 끊임없이 업데이트되며 기능과 화면이 바뀝니다. 항상 공식 웹페이지에서 최신 변경 사항을 확인하시면 됩니다.

부록 A · CURSOR

Cursor는 VS Code를 통째로 포크했다

전 세계 개발자들이 가장 많이 사용하는 편집기 VS Code를 기반으로 개발된 **AI 중심 통합 개발도구(IDE)**다. 프로젝트를 통째로 이해하고 스스로 계획을 세우는 AI 개발 에이전트가 중심에 있다.

1 에디터 중심의 작업실

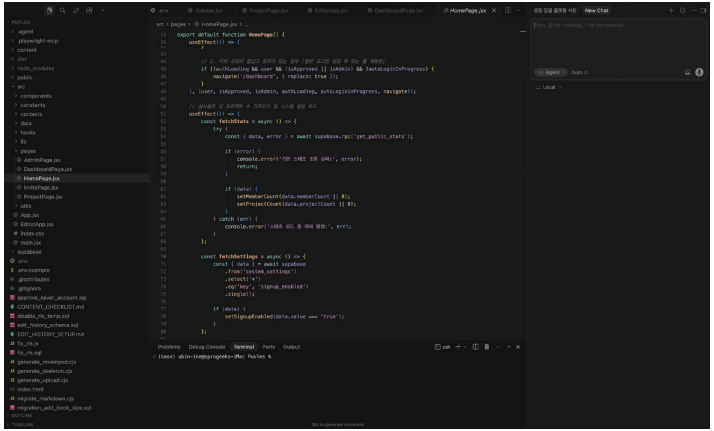
Claude Code가 "명령줄에서 만나는 유능한 비서"라면, Cursor는 "모든 도구가 갖춰진 AI 전용 설계 사무소"다.

2 완벽한 프로젝트 이해

파일 하나가 아니라 전체 폴더의 구조와 파일 간 관계를 스스로 파악한다. "로그인 로직이 어디 있지?"에 답할 수 있다.

3 기존 환경 그대로

VS Code를 포크(Fork)해 만들었다. 쓰던 확장 프로그램·테마·단축키를 **100% 그대로** 가져온다.



IDE 모드로 전환한 Cursor — 좌측 탐색기, 중앙 코드 편집기, 우측 AI 채팅창, 하단 터미널이 한 화면에 있다

부록 A · CURSOR

Cursor의 무기는 다섯 가지다

2026년 3월 기준, Cursor는 단순한 자동완성 도구를 넘어 **독립적인 에이전트 환경**으로 진화했다.

무기 01

독자적인 AI Composer 작업 환경 제공

AI가 여러 파일을 동시에 수정하고, 복잡한 작업을 단계적으로 수행하도록 지원한다.

약 4배

일반 AI 모델보다 코딩 작업이 빠르다. 대부분의 대화와 코드 작성을 **30초 이내**에 처리한다.

무기 02

Built-in Browser

크롬 창을 따로 띄울 필요가 없다. 에러가 나면 에러 화면과 콘솔 로그를 보고 스스로 코드를 고친다.

무기 03

Multi-Agent 병렬 작업

한 문제에 **최대 8개** 에이전트가 병렬 투입된다. Git 워크트리로 충돌을 막는다.

무기 04

플러그인 & 자동화

2026년 초 업데이트. 푸시할 때마다 보안 취약점을 검사하는 Agentic Security Review가 대표적이다.

무기 05

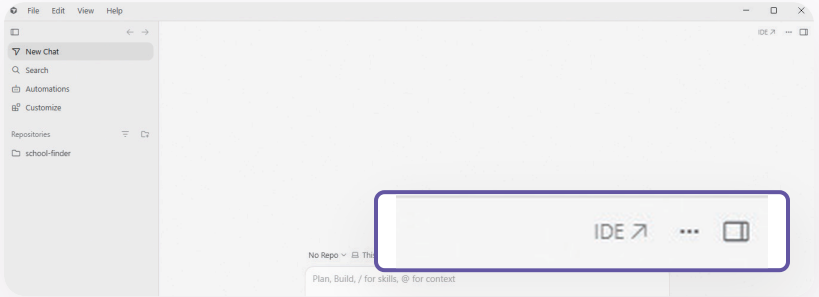
Rules(규칙) 시스템

.cursorrules 파일에 “항상 한국어로 설명해” 같은 규칙을 적어 두면 늘 적용된다.

부록 A · CURSOR 설치

설치는 여섯 단계 —
진짜 관문은 4단계 [IDE] 버튼이다

1. **cursor.com** 접속 → [Download] 클릭
2. 설치 파일 실행 → 로그인·회원 가입 화면
3. 가입한 방식으로 로그인 (예제에서는 구글 계정)
4. **우측 상단 [IDE] 버튼을 눌러 IDE 모드로 전환**
5. 보안 팝업에서 [Yes, I trust the authors]
6. [File] → [Open Folder...]로 작업 폴더 열기



기본값은 에이전트 모드다. 우측 상단 구석의 [IDE]를 눌러야 실습 화면으로 바뀐다 — 상자는 그 부분을 3배로 키운 것이다

사전 준비

Windows 10/11 · macOS · Linux. 무료 사용 가능, 본격적인 이용은 월 $20 Pro 플랜.

부록 A · 실전 수칙

“회원가입 만들어줘”는 실패한다 — 구체적으로 말하고 Diff를 읽어라

이렇게 말하면 헤맨다

“회원가입 만들어줘.”

어디에 둘지, 무엇을 입력받을지, 어떤 모양일지가 전부 빠졌다. AI는 남은 빈칸을 제 마음대로 채운다.

VS

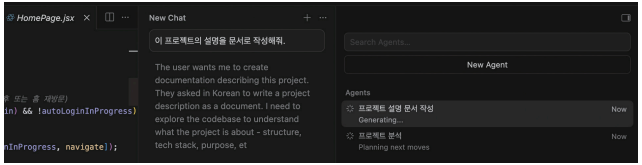
이렇게 말하면 된다

“우측 위에 이메일과 비밀번호로 로그인할 수 있는 화면을 만들고, 버튼 색상은 파란색으로 해줘.”

위치·입력 항목·색까지 지정했다. 고쳐 달라고 다시 말할 일이 줄어든다.

ACCEPT 전에

AI가 코드를 쓰면 수정 전후 비교(Diff)가 화면에 나타난다. 의도한 대로 바뀌었는지 확인한 뒤 [Accept]를 누르는 습관을 들이자. 자동 수락 기능은 권한을 사용자에게 확인받도록 설정해 둔다.



실제 요청 화면 — 프롬프트 한 줄에 두 에이전트가 동시에 붙어 일한다

부록 B · CLAUDE CODE

Claude Code는 IDE를 버리고 터미널로 갔다

가장 큰 특징은 IDE 안에서 쓰는 방식이 아니라, 터미널에서 직접 실행하는 **CLI(Command Line Interface)** 기반 도구라는 점이다.

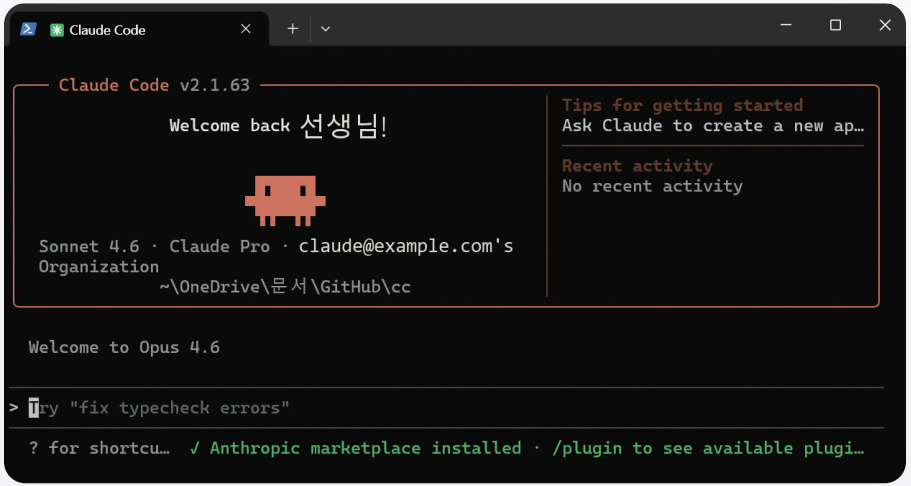
명령줄에서 만나는 유능한 비서

마우스로 버튼을 클릭하는 대신 키보드로 명령어와 자연어를 입력한다. 검은 화면의 첫인상은 어색해도, 파일을 만들고 코드를 고치고 테스트를 실행하는 방식은 Cursor나 Antigravity IDE와 크게 다르지 않다.

준비물

Claude 계정 — Pro·Max·Team·Enterprise 구독
Windows 10/11 PC (macOS도 지원)

2026년 3월 기준, 모바일·웹에서 PC 세션에 접속하는 리모트 컨트롤은 Max 플랜 전용이다. Pro 플랜도 기본 기능은 정상 사용할 수 있다.

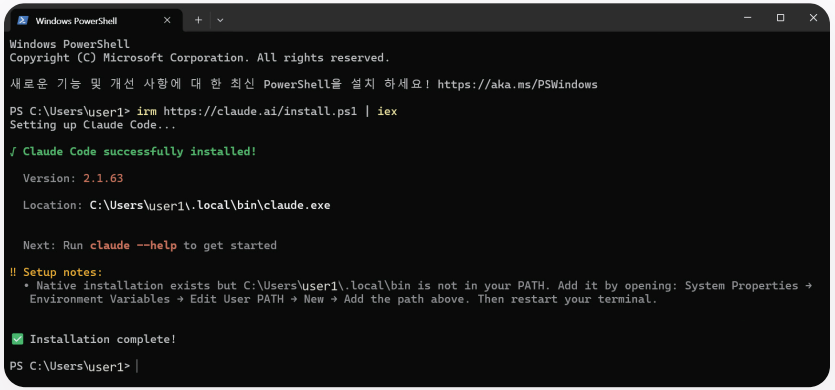


설정을 마치면 나타나는 대시보드 — 하단 입력창에 원하는 작업을 자연어로 적으면 된다

설치는 PowerShell 한 줄이다

```
# PowerShell을 관리자 권한으로 실행한 뒤 붙여넣고 Enter
irm https://claude.ai/install.ps1 ¦ iex
```

```
# 현재 폴더가 곧 작업 공간이다
cd C:\Users\사용자이름\Documents\GitHub
md cc
cd cc
claude
```



Version: 2.1.63 · 설치 경로와 Installation complete 메시지가 뜨면 끝이다.

빨간 오류가 떠도

설치는 이미 끝났다. 실행 파일 경로가 Windows PATH에 등록되지 않았을 뿐이다. PowerShell에 두 줄만 입력하면 바로 복구된다.

여기까지 오면 남는 건 프롬프트 한 줄이다. 도구는 또 바뀐다 — 그때마다 공식 웹페이지를 먼저 열면 된다.

SUMMARY

바이브 코딩의 실력은 코드를 쓰는 손이 아니라, 문제를 정확히 설명하는 입에서 나옵니다.

PART 01

한숨을 사용자 스토리와 PRD로 바꾼 사람만이 쓸 만한 앱을 얻었다

PART 02

API 주소·인증 키·요청 목적을 말해 준 사람만이 급식 데이터를 받았다

PART 03

테이블 구조와 권한까지 지시한 사람만이 사라지지 않는 앱을 가졌다

PART 04

무엇이 잘못됐는지 설명할 줄 아는 사람만이 오류에서 빠져나왔다

내일 할 한 가지

내일 아침, 가장 먼저 나온 한숨 하나를 받아 적으세요.

오늘 아침 "이게 왜 이렇게 불편하지?"로 시작했습니다.

그 한 줄이 여러분의 첫 PRD 첫 문장입니다. 오늘 배운 것 중에 지금 당장 필요한 건 그것 하나뿐입니다.

1. 한숨을 **사용자 스토리 한 문장**으로 바꾼다
   "나는 [사용자]로서, [목표]를 위해, [기능]이 필요하다"
2. 기능을 **세 개까지만** 남긴다 (MVP)
3. 그대로 AI에게 붙여넣고 **첫 화면**을 받는다

실습 프롬프트 · 완성본 소스 · 배포 데모

`github.com/lifeofpi-ux/vibecode-for-teacher`