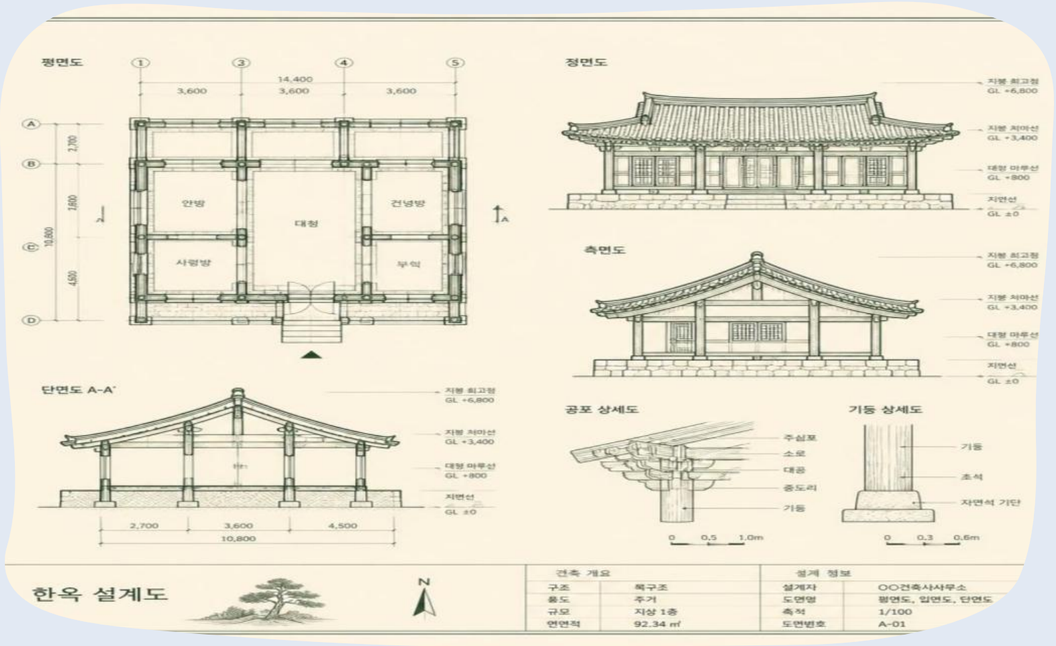
교사를 위한 바이브코딩: AI와 함께 하는 교실 도구 개발 실전 가이드

AI와 코딩이 낯선 선생님들을 위한 실전 매뉴얼

ART THERAPY INSTRUCTOR

목차

PRD란 무엇인가?



개념

제품 요구사항 명세서: 만들려는 프로그램의 기능과 동작을 명시한 문서

교실 비유

수업 지도안처럼, PRD가 구체적이면 프로젝트가 성공합니다

비주얼 가이드

건물의 정교한 청사진(설계도) 도면처럼, PRD는 프로그램 개발의 모든 세부사항을 담은 설계 문서입니다

비개발자가 폭망하는 근본 원인

망하는 대화

- '우리 반 애들용 타이머 앱 만들어줘'
- AI는 마술사가 아닙니다
- 대충 만들면 코드가 꼬입니다
- 요구가 불명확하면 프로젝트 실패

흥하는 대화

- '대상: 초3, 기본 5분 카운트다운,
  1분 남으면 배경 빨강·경고음'
- 구체적 지시
- 성공적인 결과물

교사용 마이크로 PRD 템플릿

1. 도구 명칭 및 목적

IT 용어를 빼고 교무실용으로 핵심 5가지만 작성하세요.
예: 급식 당번 추첨기 — 이름과 목적을 한 문장으로

2. 사용자(Target)

이 도구를 실제로 쓸 사람을 구체적으로 적습니다. 예: 초4 학생 및 담임 — 사용자에 따라 인터페이스가 달라집니다

3. 핵심 기능

프로그램이 반드시 할 동작을 나열하세요.
예: 버튼 눌러 랜덤 추첨, 중복 제외

4. 예외 규칙

특수 상황·제외 조건을 적으세요.
예: 전학 간 5번 학생은 제외 — 오류 방지 필수

5. 디자인 콘셉트

화면 색상·분위기·느낌을 설명하세요.
예: 파스텔 노란색, 유치원 느낌 — UX 향상

PRD 작성 1단계: 핵심 가치 및 타깃 정의

질문: 누가 사용하는가?

- 이 도구를 교실에서 누가 쓰며, 왜 필요한가?

- 교사가 혼자 교단 모니터에서 컨트롤하는 도구인지 확인

- 학생들이 스마트기기로 직접 접속해서 조작하는 도구인지 확인

- 사용자 유형에 따라 인터페이스 크기와 버튼 배열이 결정됩니다

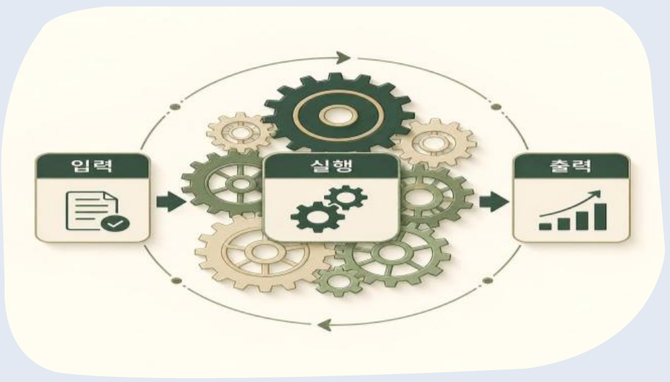
인터페이스 설계 결정

- 교사용: 대형 모니터 화면에 최적화된 큰 버튼과 명확한 레이아웃

- 학생용: 소형 태블릿이나 스마트폰 화면에 맞는 터치 친화적 UI

- 타깃 사용자를 명확히 해야 성공적인 도구 개발 가능

PRD 작성 2단계: 기능적 요구사항(FR)



FR의 개념과 작성 원칙

- FR(Functional Requirements):
  프로그램이 '반드시 실행해야 하는 행동'
- 동사 중심으로 쪼개서 지시
- 명확한 행동 단위로 정의

동사 중심 기능 분해

- 학생 이름을 입력하는 칸이 있어야 한다
- '추첨 시작' 버튼을 누르면 이름이
  무작위로 굴러가다 멈춰야 한다
- 각 기능을 구체적 동작으로 표현

입력-실행-출력 단계별 명세

- 당첨된 이름은 '오늘의 당번' 목록에 누적 기록
- 입력 → 처리 → 출력 흐름 설계
- 톱니바퀴처럼 맞물리는 논리 구조 완성

PRD 작성 3단계: 비기능적 요구사항(NFR)

성능과 속도 조건

- 느린 교실 와이파이에서도 화면은 2초 이내 로딩
- 응답 속도 최적화로 대기 시간 최소화

학교 환경 호환성

- 다양한 기기(모니터·태블릿·폰) 대응
- 학교 네트워크·브라우저 호환성 확보

데이터 안정성

- 새로고침 남발에도 데이터 초기화 방지
- 예기치 않은 조작에도 데이터 손실 방지

PRD 작성 4단계: 사용자 여정 추적

사용자 여정 지도의 개념

- 접속부터 목적 달성까지의 모든 사용 흐름을 상상하세요
- 각 단계에서 보고, 클릭하고 기대하는 행동을 명확히 정의
- 발자국 모양 타임라인으로 시각화하세요

교실 도구 여정 설계

- 실제 교실 사용 시나리오를 반영하세요
- 교사·학생 상호작용을 고려해 경로를 설계
- 예: [접속]→[반 설정]→[추첨]→[당첨 확인]→[단톡 공유]

접속 및 설정

- 웹사이트 접속
- 반 설정 또는 학생 정보 입력
- 초기 화면 로딩 및 인터페이스 확인
- 사용자 권한 설정

실행 및 확인

- 핵심 기능 버튼 클릭
- 결과 화면 표시
- 데이터 저장 및 기록
- 재실행 가능 여부 확인

공유 및 마무리

- 결과 복사 또는 캡처
- 단톡방 공유 링크 생성
- 다음 수업을 위한 초기화
- 종료 및 로그아웃

PRD 작성 5단계: 예외 처리 명세

잘못된 입력 방어 로직

- 프로그램이 망가지는 순간은
사용자가 '잘못된 행동'을 했을 때입니다
- AI에게 미리 방어벽을 세우라고
지시하세요
- 예외 상황을 사전에 예측하고
대응 방안 마련

- 학생 번호에 숫자가 아닌 한글 이름을
넣었을 때
- 아무것도 선택하지 않고 버튼을
눌렀을 때
- 필수 입력 항목이 비어있을 때

교실 필수 예외 처리

- '숫자만 입력해 주세요'라는 알림창 띄우기
- 경고음 발생으로 사용자 주의 환기
- 입력 형식 가이드 메시지 표시

- 빈 값 입력 시 진행 차단
- 중복 데이터 입력 방지
- 범위를 벗어난 값 입력 제한
- 방패가 에러 미사일을 막아내듯 안정성 확보

완성된 PRD를 AI에게 주입하기

안티그래비티 대화창 활용

- 안티그래비티 2.0 중앙 대화창을 엽니다
- 작성한 마이크로 PRD 내용을 복사하여 붙여넣기
- 한글로 명확하게 지시 전달

마법의 접착 프롬프트

- '내가 작성한 제품 요구사항 명세서(PRD)를 바탕으로 앱 개발을 시작해줘'
- '코드를 먼저 짜지 말고, 요구사항을 완벽히 이해했는지 확인 답변부터 보내줘'
- AI와의 소통 시작점

요구사항 이해 확인

- AI가 PRD를 정확히 이해했는지 확인
- 빠진 부분이나 애매한 부분 재확인
- 개발 시작 전 상호 합의 완료

명령어 활용: /grill-me (역질문 공격)

/grill-me 기능

- 결정하기 어렵거나 PRD에 빈틈이 있을 때 사용
- AI가 반대로 날카로운 질문을 던지게 하는 명령어

기획서 빈틈 검증

- '기획서에 빠진 부분이 있을 거야' 형태의 요청
- '/grill-me로 검증할 질문 3개만 던져줘' 등으로 완성도 향상

아이디어 정교화

- 미처 생각하지 못한 예외 상황 발견
- 기능 간 충돌 가능성 사전 점검
- 더 견고한 명세서 완성

실전 /grill-me 대화 엿보기

AI의 역질문 예시

- '선생님, 발표자 추첨기 기획서 잘 보았습니다. 그런데 만약 당첨된 학생이 화장실에 가고 없다면 다시 뽑기 기능이 필요하지 않을까요?'
- '번호가 저장되는 데이터는 브라우저를 닫아도 유지되어야 할까요?'
- AI가 PRD의 빈틈을 찾아 날카롭게 질문을 던집니다.
- 교사는 미처 생각하지 못했던 예외 상황을 발견하게 됩니다.

교사의 답변으로 명세 완성

- '아! 다시 뽑기 버튼 필수야. 데이터는 유지 안 돼도 상관없어.'
- 교사의 명확한 답변으로 요구사항이 완벽하게 보완됩니다.
- 탁구 랠리처럼 질문과 답변을 주고받으며 기획서가 단단해집니다.
- 이 과정을 거치면 개발 중 혼란이 사라지고 완성도가 높아집니다.

점진적 빌드업(Slice) 개발법

1단계: 기본 기능 먼저 구현

- 숫자가 무작위로 뜨는 기본 추첨 기능만 먼저 완성합니다.
- 잘 작동하는지 확인하고 테스트합니다.
- 한 번에 욕심내면 AI도 체하고 코드도 꼬입니다.

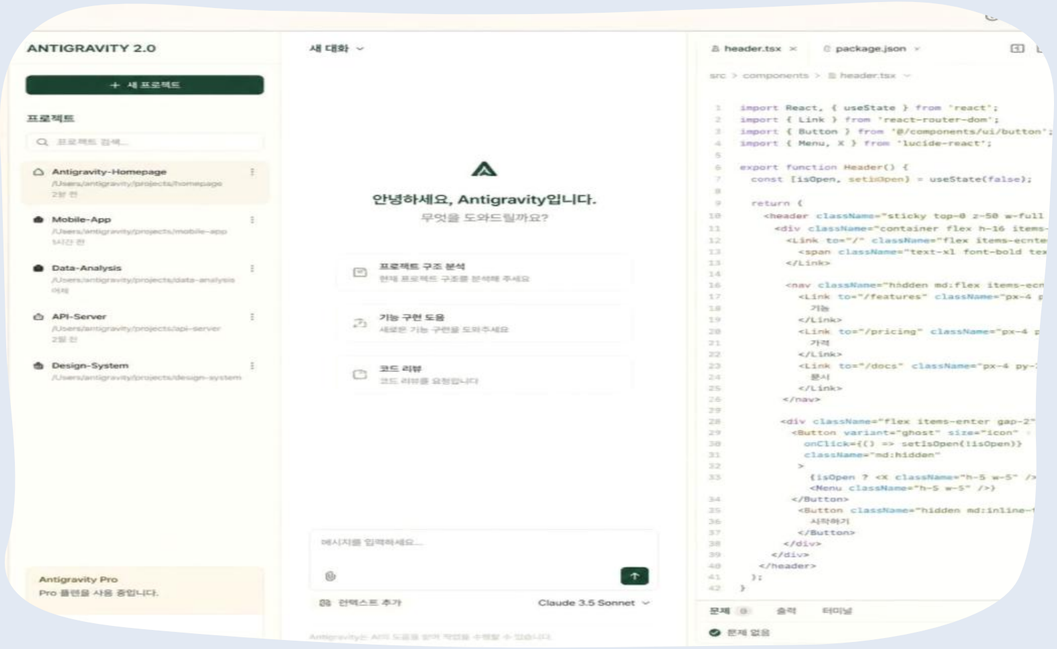
2단계: 효과음 추가

- 기본 기능이 완벽하면 '여기에 효과음 추가해줘'라고 요청합니다.
- 단계별로 기능을 쌓아 올립니다.
- 각 단계마다 확인하며 안정성을 확보합니다.

3단계: 디자인 세련되게 변경

- 모든 기능이 완벽하면 '배경을 초록색 칠판 느낌으로 세련되게 바꿔줘'라고 디자인을 요청합니다.
- 슬라이스 개발법으로 완성도를 높입니다.

안티그래비티 2.0 화면 구조



삼분할 화면의 역할

- 좌·중·우 3단으로 나뉜 대시보드 구조입니다
- 각 영역의 역할이 명확하며 혼동을 줄입니다

오른쪽 창은 코드 보관소

- AI 작업 결과를 투명하게 보여주는 공간입니다
- 순수 소스코드 파일이 쌓이며 미리보기 창이 아닙니다

마우스 조작 금지 영역

- 우측 창의 영어 텍스트는 내부용입니다
- 신경 쓰지 말고 중앙 대화창에 한글로 지시하세요

좌측: 프로젝트 목록 서랍장

과거 작업 내역 보관

- 내가 그동안 생성하고 작업해 온 프로젝트 폴더들의 목록 창입니다.
- 교무실 서랍장처럼 체계적으로 정리되어 있습니다.
- 과거 프로젝트를 안전하게 보관합니다.

주제별 대화방 분리

- 교실 타이머, 자리 배치 등 주제별로 분리됩니다
- 각 프로젝트마다 독립 대화방이 생성됩니다
- 작업 내역이 섞이지 않고 깔끔히 관리됩니다

언제든 꺼내어 수정 가능

- 과거 작업 내역을 언제든 다시 꺼내어 수정할 수 있습니다.
- 필요할 때마다 서랍을 열듯 프로젝트를 불러옵니다.
- 지속적인 개선과 업데이트가 가능합니다.

중앙: AI 비서와의 대화창

한글로 명령 내리는 공간

- 교사가 지휘관처럼 명령하는 공간입니다.
- 한글로 편하게 소통합니다.

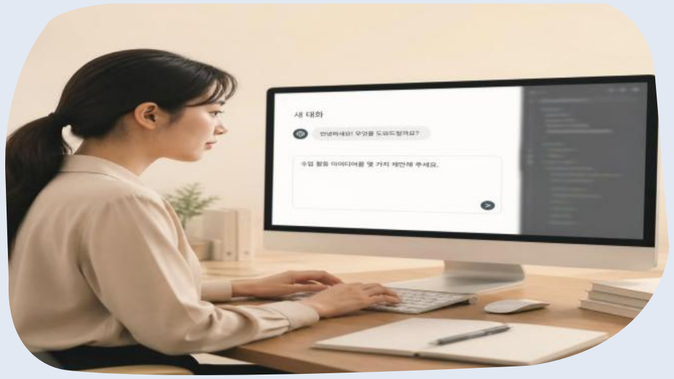
AI 사고 과정 실시간 출력

- AI의 사고 과정이 문장으로 실시간 출력됩니다.
- 작업 과정을 투명하게 확인할 수 있습니다.

말풍선 속 빛나는 전구

- 아이디어가 떠오르는 순간을 시각화합니다.
- 활발한 대화형 협업 공간입니다.

우측: 아티팩트 공간의 진실



소스코드 파일 보관함

- AI가 만든 소스코드 파일을 차곡차곡 보관합니다.
- 코드 문서가 체계적으로 정리되어 있습니다.

AI의 비밀 일기장

- 우측 창의 영어는 AI의 비밀 일기장입니다.
- 선생님이 건드릴 필요 없는 기술적 내용입니다.

한글 대화창에만 집중

- 눈길도 주지 마세요!
- 선생님은 오직 중앙 대화창에 한글로만 지시하시면 됩니다.
- AI가 알아서 코드를 작성하고 관리합니다.

ART THERAPY INSTRUCTOR

로컬, IDE, 터미널 용어 이해

로컬(Local)

- 내 컴퓨터 내부의 독립된 작업 공간입니다.
- 외부 해킹 걱정 없는 안전한 실험실입니다.

IDE (통합 개발 환경)

- 코딩·디버깅·빌드를 한 곳에서 처리하는 환경입니다.
- 모든 도구가 정리된 통합 개발 환경입니다.

터미널(Terminal)

- 키보드로 명령하는 텍스트 기반 창입니다.
- 긴급 명령을 직접 전달하는 도구입니다.

ART THERAPY INSTRUCTOR

MCP와 반응형 디자인

MCP (Model Context Protocol)

- AI 모델이 내 로컬 컴퓨터의 실제 파일과 도구를 안전하게 읽고 쓸 수 있도록 연결하는 최신 범용 프로토콜 표준

- 교실 비유: AI 비서에게 교무실 열쇠 꾸러미를 건네주어, 비서가 직접 서랍을 열고 필요한 학생 명렬표 파일을 꺼내와 작업할 수 있게 허가해 주는 공식 통로

- 샌드박스(우물)를 벗어나 외부 도구들과 유기적으로 결합하는 혁신적 기술

반응형 디자인 (Responsive UI)

- 접속하는 기기의 화면 크기에 맞춰 레이아웃과 글씨 크기가 자동으로 조절되는 유연한 화면 설계 기술

- 학교 현실: 교사가 스마트폰으로 볼 때, 학생들이 교실 태블릿으로 볼 때, TV 모니터로 띄울 때 화면이 깨지지 않고 찰떡같이 맞춰지는 필수 기술

- 하나의 웹사이트가 모든 기기에서 완벽하게 작동

ART THERAPY INSTRUCTOR

디버깅: 버그라는 벌레 잡기

디버깅(Debugging)의 개념

- 에러의 원인을 찾아 수정하는 과정입니다.
- 코드 오류에 당황하지 마세요.

교사의 대처법

- 에러 메시지를 복사해서 대화창에 던지기
- '이 벌레 잡아줘'라고 지시하면 됨
- 침착하게 에러 내용 전달이 핵심

AI 활용 디버깅

- AI에게 에러 메시지를 그대로 복사하여 전달
- AI가 자동으로 원인 분석 및 해결책 제시
- 빠르고 정확한 문제 해결 가능

ART THERAPY INSTRUCTOR

웹 브라우저 개발자 도구 (F12)

F12 키 활용법

- 버튼 무반응 시 F12 누르기
- 오른쪽에 개발자 도구 창 열림

콘솔(Console) 탭

- 'Console' 탭 클릭
- 붉은 글씨로 에러 원인과 위치 확인

에러 메시지 복사

- 붉은 에러 메시지 통째 복사
- AI 창에 붙여넣기하여 디버그

복구 포인트와 Git 연동

복구 포인트의 개념

- 특정 시점의 코드 상태를 스냅샷으로 저장
- 문제 발생 시 과거 상태로 복구 가능한 유일한 보루
- 안전한 백업 시스템

타임머신 기록 장치

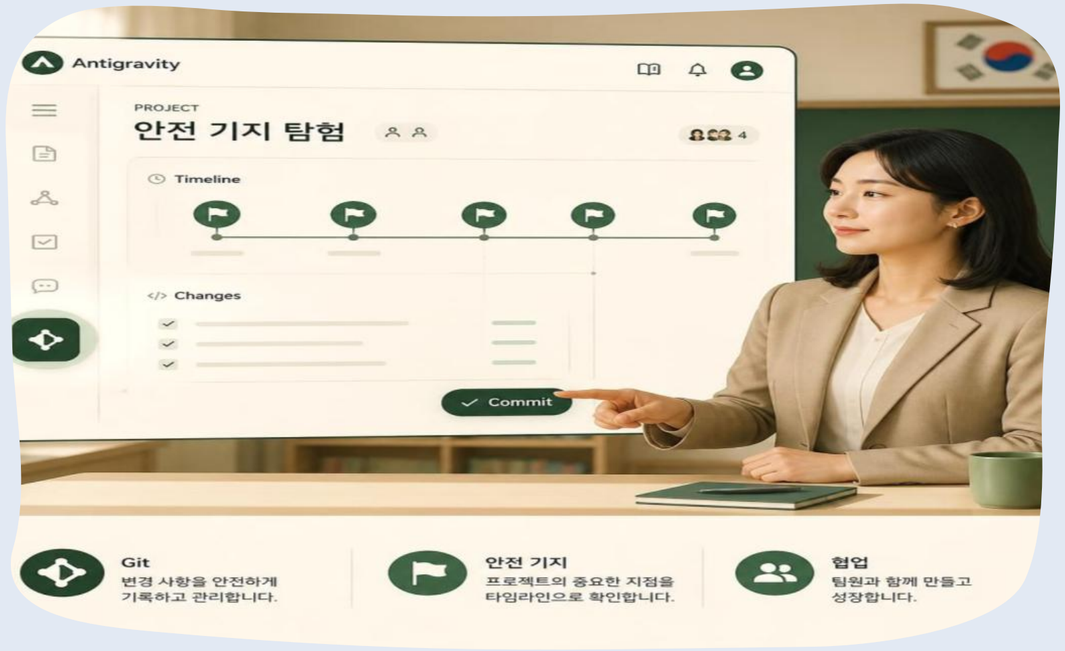
- 망가진 프로그램을 한순간에 복구
- 과거 성공 시점으로 되돌아가기
- 시간여행 같은 복구 기능

게임 세이브 비유

- 보스 전 '세이브' 누르는 것과 동일
- 안전 지점 확보 후 도전
- 실패해도 다시 시작 가능

ART THERAPY INSTRUCTOR

안티그래비티 실전 복구 매뉴얼



Git 아이콘 찾기

- 좌측 하단 Git 아이콘 클릭
- 기능 완성 시마다 실행해 습관화

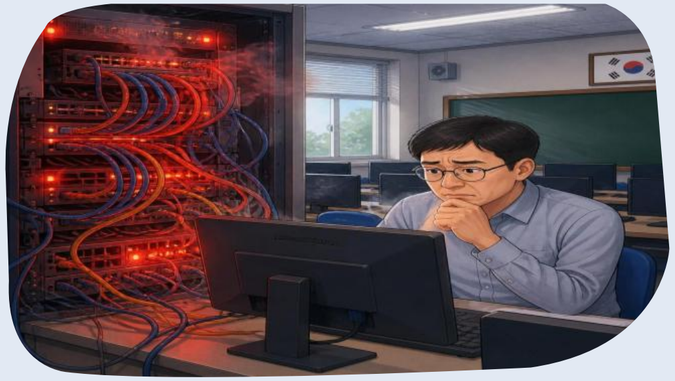
Commit 저장하기

- '기본 타이머 완성'이라고 메모 적기
- [Commit(저장)] 버튼 누르기
- 성공 시점 기록 완료

과거로 되돌리기

- 문제 생기면 과거 성공 시점 선택
- 1초 만에 되돌아가 안전 기지 복구

학교 현장 교육용 앱 보안



외부인 접근 차단 필요

- 우리 반 학생들이 직접 사용하는 도구입니다.
- 외부인이 주소로 접근할 수 있으며
- 화면 조작·데이터 변조 위험 존재

네트워크 부하 방지

- 학교 네트워크망에 과도한 부하를 줄 수 있습니다.
- 동시 접속자 수 제한 설정이 필요합니다.
- 서버 다운 시 수업 진행에 차질이 생깁니다.

민감 정보 노출 대형 사고 예방

- 학생 이름, 성적 등 개인정보가 노출될 수 있습니다.
- 보안이 허술하면 대형 사고로 이어집니다.
- 배포 전 철저한 보안 점검이 필수입니다.

ART THERAPY INSTRUCTOR

환경변수(.env): 금고 열쇠 분리

API 비밀 키 격리 기술

- API 키를 코드에 직접 쓰지 않습니다.
- 별도 .env 파일에 보관합니다.

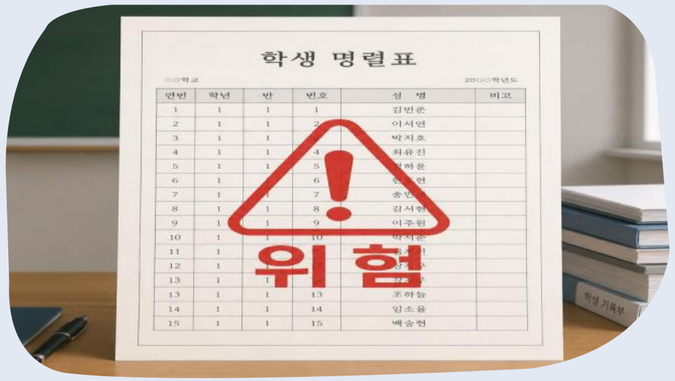
소스코드와 인증 정보 분리

- 인증 정보를 코드와 분리합니다.
- .env는 깃업로드에서 제외합니다.

교장실 금고 속 황금 열쇠

- 정답표를 공개하지 않습니다.
- 교장실 금고에 보관합니다.

학생 개인정보 하드코딩 절대 금지



코드 내부 직접 입력의 위험성

- 하드코딩: 코드에 학생 이름, 성적을 직접 입력하는 행위입니다.
- 소스코드가 외부에 노출되면 개인정보가 유출됩니다.
- 절대 금지해야 하는 위험한 방식입니다.

CSV 파일 업로드 방식 권장

- 웹사이트 화면에서 교사가 직접 파일을 업로드합니다.
- 메모리에 잠시 띄우고 작업 후 삭제합니다.
- 영구 저장하지 않아 안전합니다.

LLM 서버 학습 데이터 수집 방어

- AI 서버가 학생 데이터를 학습하지 못하게 차단합니다.
- 개인정보 보호법 준수가 필수입니다.
- 방어 기제 설정으로 안전하게 보호합니다.

ART THERAPY INSTRUCTOR

AI 보안 크리틱 최종 점검

배포 전 보안 검사 프롬프트

- 배포 직전에 반드시 실행합니다.
- 중앙 창에 보안 검사 프롬프트 입력.

개인정보 유출 위험 리뷰

- 학생 개인정보 노출 여부를 확인합니다.
- 하드코딩된 개인정보를 검토합니다.

API 키 노출 문제 철저 검증

- 코드에 API 키가 노출됐는지 확인합니다.
- .env로 안전 분리되었는지 점검합니다.

배포(Deployment)와 URL 원리

로컬에서 공용 서버로 이사

- 내 컴퓨터 로컬 작업실에서 만든 프로그램을 이동합니다.
- 전 세계 사람들이 접속할 수 있는 공용 인터넷 서버로 옮깁니다.
- 이 과정을 배포(Deployment)라고 합니다.

고유 인터넷 주소(URL) 발급

- 배포가 성공하면 고유한 웹 주소가 생성됩니다.
- 예: https://my-school-tool.vercel.app
- 이 주소만 있으면 누구나 접속할 수 있습니다.

전 세계 어디서나 접속 가능

- 스마트폰·태블릿·PC 어디서나 접속됩니다.
- 주소만 공유하면 즉시 사용할 수 있습니다.

기존 배포 방식의 한계

Netlify/Surge 수동 업로드

- 완성된 HTML을 매번 수동 업로드합니다.
- 드래그나 터미널로 올리며 시간이 걸립니다.

매번 재빌드 및 재업로드 번거로움

- 한 글자 수정해도 처음부터 다시 빌드합니다.
- 재업로드 반복으로 효율이 낮습니다.

무거운 책 보따리 직접 운반

- 매번 무거운 파일을 직접 옮깁니다.
- 자동화된 시스템이 필요합니다.
- 용도에 따라 수동 업로드가 필요할 수 있습니다.

ART THERAPY INSTRUCTOR

목차

Vercel 플랫폼 이해

- 초고속 자동 배포 플랫폼
- 실시간 웹사이트 최신화
- 우렁각시 같은 자동화

3대 인프라 연동

- 내 컴퓨터 작업 환경
- GitHub 클라우드 창고
- Vercel 자동 인쇄소

GitHub 계정 연결

- 안티그래비티 설정
- GitHub 권한 승인
- 비밀 통로 개설 완료

코드 창고 업로드

- 한글 명령어로 업로드
- AI 자동 터미널 수행
- 클라우드 안전 복사

Vercel 배포 실습

- GitHub 로그인 연동
- Import 버튼 클릭
- Deploy로 배포 완료

응급 상황 대처법

- 무한 루프 강제 종료
- 대화 리셋 방법
- API 환경변수 설정

Vercel 이란?

전 세계 개발자가 사랑하는 클라우드

- 최첨단 스마트 자동 배포 서비스
- 코드 수정 시 즉시 감지
- 별도 설정 없이 자동 작동

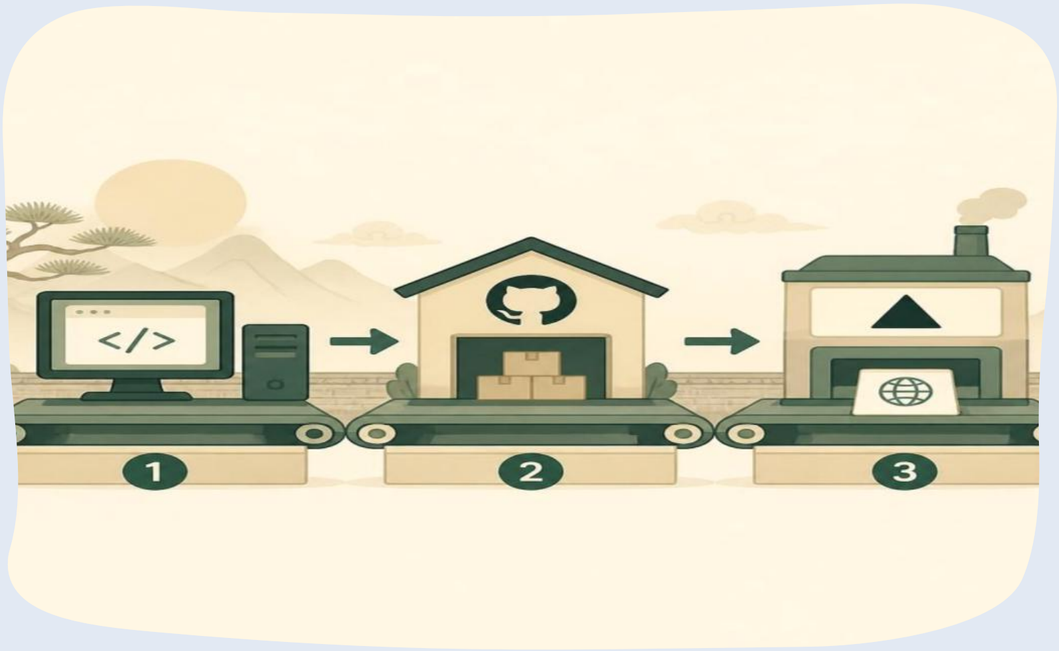
3초 만에 실시간 자동 배포

- 저장소에 코드 업로드 시
- Vercel이 실시간 감시
- 인터넷 주소 즉시 갱신

우렁각시 같은 자동화 시스템

- 선생님은 코드만 수정
- 나머지는 자동 처리
- 손댈 필요 없는 자동화

3대 인프라 연동 파이프라인



내 컴퓨터

- 안티그래비티 로컬 환경
- 한글로 코딩 지시
- 작업이 시작되는 현장

GitHub 창고

- 원격 저장소 클라우드
- 코드 안전 보관
- 인터넷 백업 창고

Vercel 인쇄소

- 자동 배포 서버
- 창고 실시간 감시
- 주소 자동 인쇄 시스템

실전 1단계: GitHub 계정 연결

GitHub 연동 클릭

- 안티그래비티 2.0 설정 메뉴
- 'GitHub 연동' 버튼 클릭
- 인터넷 창 자동 실행

권한 승인 완료

- 깃허브 계정 로그인
- 'Authorize' 파란 버튼
- 권한 승인 완료

비밀 통로 개설 완료

- 컴퓨터 작업실과 연결
- 인터넷 백업 창고 연동
- 보안 통로 개설 성공

ART THERAPY INSTRUCTOR

실전 2단계: 코드 업로드

한글 명령어 입력

- 안티그래비티 대화창 실행
- '이 프로젝트를 깃허브에 새 레포지토리로 업로드해줘'
- 한글로 편하게 명령

AI 자동 수행

- AI가 명령어 자동 해석
- 터미널 명령어 자동 수행
- 복잡한 Git 명령 불필요
- 모든 과정 자동화

클라우드 업로드 완료

- 소스코드 전체 복사
- GitHub 원격 저장소 생성
- 인터넷 클라우드 업로드
- 실시간 진행 상황 표시

안전한 백업 완료

- 코드 안전하게 보관
- 언제든 복구 가능
- 팀 협업 준비 완료
- 다음 단계 준비 완료

실전 3단계: Vercel 로그인 및 연동

GitHub 계정으로 1초 회원가입

- Vercel 홈페이지 접속
- 'Continue with GitHub' 버튼 클릭
- 즉시 로그인 및 회원가입 완료

프로젝트 Import 버튼 클릭

- 대시보드에서 깃허브 창고 확인
- 프로젝트 옆 [Import] 클릭 후 완료

Deploy 버튼으로 배포 완료

- [Deploy] 버튼 한 번만 클릭
- 자동 배포 시작
- 전 세계 접속 가능한 주소 생성 완료

ART THERAPY INSTRUCTOR

초비상: Vercel 배포 후 앱 먹통 현상

단골 에러 상황

- 배포 완료, 주소 생성됨
- 버튼 클릭 시 작동 안 함
- 흰 화면만 나타남
- 당황하지 마세요!

먹통의 원인

- .env 파일 미업로드
- 원격 저장소에 올리지 않음
- Vercel이 API 키를 모름
- 서비스 중단 발생

보안상 누락된 열쇠

- 제미나이 API 키 정보 없음
- 교장실 금고 열쇠 분실 상태
- 엔진에 연료가 없는 것과 동일
- 선생님 잘못 아님

실전 4단계: Vercel에 API Key 환경변수 설정

Settings 탭 이동

- Vercel 프로젝트 페이지 상단
- [Settings] 탭 클릭
- 설정 화면 진입

Environment Variables

- 좌측 메뉴 확인
- [Environment Variables] 클릭
- 환경변수 관리 화면 열기

Key 값 입력

- Key 칸에 입력
- GEMINI_API_KEY 정확히 기입
- 대소문자 구분 주의

Value 값 저장

- Value 칸에 내 API 키 붙여넣기
- [Save] 버튼 클릭
- 엔진에 연료 공급 완료!

ART THERAPY INSTRUCTOR

자동 배포의 마법 체험하기

한글로 코드 수정 지시

- 안티그래비티 대화창에 한글로 명령
- '타이머 글씨 크기 2배로 키워줘'
- AI가 즉시 코드 수정 시작

AI가 자동으로 GitHub 업로드

- AI가 수정된 코드 자동 저장
- GitHub 원격 저장소에 자동 업로드
- 백업 창고에 최신 버전 보관 완료

Vercel이 3초 만에 주소 갱신

- Vercel이 GitHub 변경 감지
- 전 세계 인터넷 주소 실시간 갱신
- 새로고침만 누르면 변경사항 확인!

응급 상황 대처법 1: 무한 루프 및 먹통

증상 확인

- 웹페이지가 뱅글뱅글 회전
- 브라우저 완전히 굳어버림
- AI가 코드를 잘못 작성
- 무한 루프 발생

대처 방법

- 터미널 창 클릭
- 명령 프롬프트 창 선택
- 키보드 [Ctrl + C] 준비
- 여러 번 강하게 누르기

즉시 강제 종료

- 폭주하던 로컬 서버 중단
- 구동 즉시 강제 종료
- 안전하게 멈춤
- 비상정지 완료

ART THERAPY INSTRUCTOR

응급 상황 대처법 2: 대화 리셋 및 기억 상실

증상: AI 혼란 및 무한 반복

- AI가 혼란에 빠진 상태
- 에러 메시지를 줘도 소용없음
- 똑같은 잘못된 코드만 반복
- 대화가 늪에 빠짐

대처법 1: 새 대화방 생성

- 좌측 메뉴에서 새 대화 시작
- 깨끗한 상태로 다시 시작
- 이전 대화 내용 완전 초기화
- 미련 없이 리셋

대처법 2: 맥락 무시 요청

- 프롬프트에 명확히 지시
- '지금까지 대화 맥락 전부 무시'
- 우측 아티팩트 코드만 확인 요청
- 새로운 마음으로 대기

최종 코드만 기억하도록 지시

- 최종 저장된 코드 파일 구조만 인식
- 자잘한 대화 흔적 지우기
- 칠판 지우개로 깨끗하게 정리
- 다음 지시 대기 상태 전환

ART THERAPY INSTRUCTOR

교사용 바이브코딩 핵심 치트키

기획 단계

- 화내지 말고 촘촘한 교사용 마이크로 PRD 양식을 먼저 작성
- 기능 요구사항을 구체적으로 정리

개발 단계

- 욕심내지 말고 기능 1개 성공할 때마다 Git Commit 저장
- 타임머신처럼 안전한 복원 지점 확보

마무리 단계

- 배포 전 에이전트 보안 검증 필수
- Vercel 설정 창에 API 환경변수 반드시 등록

ART THERAPY INSTRUCTOR

AI와 함께 교육의 한계를 넘어서다

교육 상상력 실현

- 코딩 습득이 목표가 아닙니다
- 교육 철학을 도구로 즉시 실현
- 교육적 상상력 구체화

기술 장벽 제거

- 아무런 기술적 장벽 없이 개발 가능
- 내 교실에 필요한 도구를 직접 제작
- AI가 모든 기술적 어려움 해결

창의적 교실 구현

- AI와 Vercel이라는 최고의 비서 활용
- 더 편리하고 창의적인 청학고 교실
- 학생 맞춤형 교육 도구 제작

대한민국의 미래 교육을 향하여

교육 철학 실현

- 선생님의 교육 철학을
실제 도구로 구현
- 개인별 교육 가치관 반영 가능

학생 참여 확대

- 학생과 함께 만드는 미래 교육 현장
- 맞춤형 학습 도구 공동 개발

편리한 교실 환경

- 더 편리하고 창의적인 수업 환경 조성
- 반복 업무 자동화로 교육에 집중

지속 가능한 혁신

- 바이브코딩으로 지속적 교실 혁신
- AI 시대 교육 리더십 확보

감사합니다

선생님들의 창의적인 교실 혁신을 응원합니다.
AI와 함께 더 나은 교육의 미래를 만들어 가세요!