

[13주차] 바이브코딩 명세

바이브 코딩(Vibe Coding)은 AI 동반자(Cursor, Antigravity 등)와 함께 대화하듯이 코딩하는 방식입니다. 처음 이 방식에 적응하고 AI의 도움을 극대화하기 위해 준비해야 할 문서는 주로 **AI에게 프로젝트의 모든 맥락을 알려주는 문서**이며, 핵심은 **Context**와 **Current Task**를 명확히 하는 것입니다.

AI IDE는 이 문서를 '프로젝트의 성경(Bible)'처럼 활용하여, 코드 생성, 디버깅, 문서화에 있어 프로젝트의 목표와 규칙을 벗어나지 않도록 합니다.

다음은 준비해야 할 필수 문서와 그에 포함되어야 할 내용입니다.

바이브 코딩을 위한 필수 준비 문서

1. README.md (프로젝트 소개 및 실행 가이드)

프로젝트를 처음 접하는 사람(또는 새로운 팀원/AI)이 환경을 설정하고 실행하는 방법을 알려줍니다.

| 항목 | 필수 내용 | AI IDE 활용 목적 |
|---------|---|--|
| 개요 및 목적 | 프로젝트의 최종 목표 와 해결하려는 문제 를 2~3문장으로 요약합니다. | AI가 가장 먼저 읽는 문서 로, 전체적인 맥락을 파악하고 향후 모든 코드 생성의 방향을 설정합니다. |
| 기술 스택 | 사용 언어 및 버전 (예: Python 3.11), 주요 라이브러리 (예: PyTorch, Pandas, FastAPI) | AI가 해당 스택에 맞는 문법과 최신 API 를 사용하여 코드를 생성하도록 가이드합니다. |
| 설치 방법 | 환경 구축 명령어, 의존성 설치 (<code>pip install -r requirements.txt</code>) 방법 | AI가 <code>requirements.txt</code> 파일을 관리하거나, 누락된 라이브러리를 자동으로 인식하여 설치를 제안합니다. |
| 실행 방법 | 프로젝트 실행을 위한 최종 명령어 (예: <code>python src/main.py</code>) | AI가 실행 가능한 코드를 생성하고, 테스트 환경 을 구성할 때 참고합니다. |

```

1  ∨ # Board Application
2    이 어플리케이션은 사람들과 소통할 수 있는 게시판 Application 입니다.
3
4  ∨ ## Tech Stack
5    - Java 17
6    - Spring Boot 3.2.5
7    - Spring Data JPA
8    - Spring Security
9    - H2 Database
10   - Thymeleaf
11
12  ∨ ### Minor Dependencies
13    - Lombok
14    - Log4j2
15
16  ∨ ## Feature
17    - 사용자 관련 CRUD
18    - 게시글 관련 CRUD
19    - 댓글 관련 CRUD
20    - 게시글과 댓글에 대한 좋아요 기능
21    - 부적절한 게시글 또는 댓글에 대한 신고 기능

```

2. CONTEXT.md (AI를 위한 프로젝트 상세 배경)

AI가 코드를 생성할 때 반드시 참고해야 할 모든 규칙과 제약 조건을 담은 문서입니다. [README.md보다](#) 더 상세하고 개발적인 내용을 포함합니다.

| | 필수 내용 | AI IDE 활용 목적 |
|------|--|--|
| 구조 | 현재 프로젝트의 핵심 디렉토리 구조 와 각 파일의 역할 (예: <code>src/</code> <code>model.py</code> 는 모델 클래스만 정의함) | AI가 코드를 어떤 파일에 작성해야 할지, 또는 기존 파일의 어떤 부분을 수정해야 할지 정확히 결정하게 합니다. |
| 션 | 변수명 규칙 , 주석 스타일, 코드 포매팅 규칙 (예: "PEP 8 준수", "클래스명은 PascalCase", "모든 함수에 타입 힌트 사용") | AI가 팀 규칙에 맞는 일관된 코드 를 생성하도록 강제하여, 나중에 수동으로 코드를 수정할 필요성을 줄입니다. |
| 약 사항 | 특정 하드웨어 제약, 데이터 처리 방식의 요구사항 (예: "메모리 부족으로 배치 사이즈는 32를 넘길 수 없음", "특정 함수는 비동기(Async)로 구현해야 함") | AI가 비현실적인 코드 를 생성하는 것을 방지하고, 프로젝트 환경에 최적화된 솔루션 을 제안하게 합니다. |
| 세 | 사용할 데이터셋의 특징 (컬럼 이름, 데이터 타입, 결측치 처리 규칙 등) | AI가 데이터 로드, 전처리, 모델링 코드를 작성할 때 데이터 구조에 맞는 코드 를 생성하도록 보장합니다. |

CONTEXT.md 파일(또는 CLAUDE.md, GEMINI.md, llm-context.md)와 같은 유사한 이름의 파일)은 AI 비서를 위한 지속적인 계약 또는 지침 세트 역할을 하며, 프로젝트별 지침, 코딩 규칙 또는 아키텍처 원칙을 제공합니다. 이를 통해 AI는 프로젝트의 의도를 이해하고 더욱 정확하고 맞춤화된 응답을 제공할 수 있습니다.

다음은 Anthropic에서 제시한 CLAUDE.md 파일에 포함될 수 있는 내용의 예입니다.

코드

Auto (Bash) ▾



```
# CLAUDE.md 예제

# 프로젝트: 전자상거래 API

## 아키텍처 원칙
- 마이크로서비스 패턴 사용
- PostgreSQL + Redis 조합
- FastAPI + Pydantic 스택

## 코딩 규칙
- 모든 엔드포인트에 타입 힌트 사용 필수
- Pydantic 모델로 요청/응답 검증
- HTTPException으로 오류 표준화

## 테스트 요구 사항
- 모든 엔드포인트에 대한 단위 테스트 작성
- PyTest 픽스처 활용
- 90% 이상의 커버리지 유지

## 보안 고려 사항
- JWT 토큰 기반 인증
- 속도 제한 적용
- SQL 삽입 방지

이 예제는 CONTEXT.md 파일이 아키텍처 선택, 코딩 표준, 테스트 요구 사항 및 보안 고려 사항을 포함하여 프로젝트의 다양한 측면을 정의하는 방법을 보여줍니다. AI 비서는 이러한 정보를 미리 제공함으로써 모든 상호작용에서 이 맥락을 활용하여 더욱 일관되고 관련성 있는 결과를 도출할 수 있습니다.
```

3. CURRENT_TASK.md (현재 진행 중인 작업 정의)

AI에게 "지금 당장 무엇을 해야 하는지"를 명확히 지시하는 문서입니다.

| 항목 | 필수 내용 | AI IDE 활용 목적 |
|-----------------------|---|--|
| 현재 태스크 (Current Task) | 현재 해결하고자 하는 구체적인 작업을 명시합니다. (AI에게 제공하는 가장 중요한 프롬프트입니다.) | AI가 장황한 설명 없이 최소한의 작업 단계를 거쳐 문제 해결 코드를 생성하도록 집중시킵니다. |
| 요구되는 산출물 | 해당 태스크의 결과물 (예: "새로운 utils.py 파일") | AI가 작업 완료 후 어떤 파일을 확인해야 하는지 알려줍니다. |

| | | |
|----------|--|--|
| | 의 <code>load_data()</code> 함수", "오류를 수정한 train.py 파일의 15라인") | |
| 이슈 또는 버그 | 만약 버그를 수정하는 중 이라면, 오류 메시지 전문 (Traceback)과 오류가 발 생하는 코드 라인을 첨부 합니다. | AI가 오류 메시지를 기반으로 디버깅 및 수정 코드를 즉시 제안하도록 유도합니다. |

미니 프로젝트 문서 작성 팁

| 문서 작성 팁 | 이유 |
|-----------|---|
| 최신 정보 유지 | AI는 문서에 기반해 행동하므로, 문서가 최신이 아니면 엉뚱한 코드를 생성합니다. 특히 CONTEXT.md 를 업데이트하세요. |
| 간결함 유지 | 장황한 설명은 AI의 이해를 방해합니다. 필요한 정보만 핵심 키워드와 목록 형태로 명확하게 제공합니다. |
| 코드 스니펫 포함 | CONTEXT.md 에 이미 구현된 핵심 클래스나 함수 정의를 몇 줄 첨부하면, AI가 기존 코드 스타일을 더 잘 모방하게 됩니다. |