10. 커스텀 슬래시 커맨드 사용하기

커스텀 명령어 구조와 정의

/prefix : command-name [arguments]

명령어 구조 요소

prefix

명령어의 스코프를 나타냅니다. 프로젝트(project) 또는 사용자(user) 범위

- 프로젝트: 프로젝트 단위로만 사용(예) .claude/commands/create-pr.md)
- 사용자: 공통으로 사용 가능 (예) ./claude/commands/analyze-code.md)

command-name

마크다운 파일 이름에 해당하며, 명령의 정의

[arguments]

추가 매개변수를 의미, 실제 변수나 프롬프트 입력 가능

예: /project:create-pr issue-79

네임스페이싱 (Namespacing)

폴더를 정의하여 명령어를 카테고리할 수 있음. 예: /github:create-pr

마크다운 파일 기반 정의

markdown

```
---
name: "GitHub 이슈 생성"
description: "GitHub 이슈를 생성합니다."
allowed-tools: ["github"]
---

이슈를 생성합니다. 다음 정보가 필요합니다:
1. 레포지토리 이름 2. 이슈 제목 3. 이슈 내용
```

마크다운 파일은 클로드가 내용을 쉽게 파악하도록 작성됩니다.

메타데이터(Frontmatter)로 커맨드 실행 시 사용 가능한 도구와 설명을 추가가능

.AI 기반 개발에서 서브 에이전트를 활용한 토큰 컨텍스트 최적화

프로젝트 커스텀 슬래시 커맨드 (프로젝트 단위)

- 위치: .claude/commands

사용자 커스텀 슬래시 커맨드 (개인적으로 자주 사용하는 명령어 - 여러 프로젝트에 사용)

- 위치:~/.claude/commands

네임스페이스

- 프로젝트 네임스페이스 위치 :.claude/commands/(namespace〉/〈command〉.md
- 사용자 네임스페이스 위치: ~/.claude/commands/(namespace〉/(command〉.md
- 사용자 네임스페이싱 prefix:/(namespace):《command》
- 프로젝트 네임스페이싱 prefix:/(namespace):《command)

실습

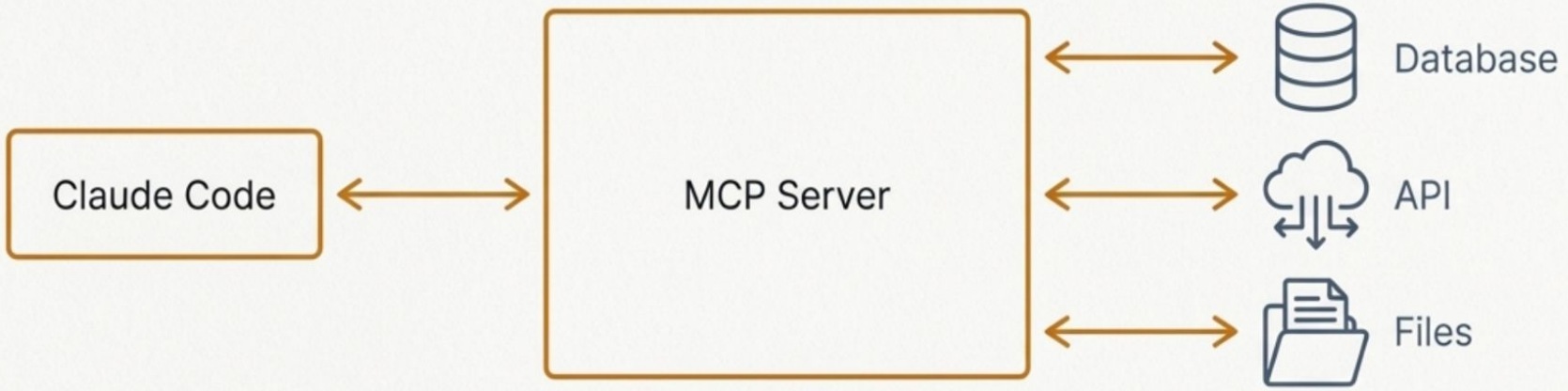
1. .claude/command/redesign.md에 저장
2. claude 창에 들어가 명령어를 진행

allowed-tools: Bash(mkdir)
description: 하나의 디자인을 기반으로 여러 가지 독창적인 디자인 콘셉트를 병렬적으로 생성합니다.
디자인 콘셉트 병렬 생성기
이 커맨드는 제공된 단일 디자인을 바탕으로 다양한 관점과 스타일을 적용하여 여러 개의 독창적인 디 자인 콘셉트를 동시에 생성합니다. 각 콘셉트는 고유한 특징과 접근 방식을 가지며, 사용자가 디자인 방향을 다각도로 검토하고 선택할 수 있도록 돕습니다.
Prompt Instruction
현재 index.html의 디자인을 5개의 새로운 디자인으로 새로 제작해줘. 각각 design_concepts 폴더에 index_1.htmL, index_2.html . 형태로 새로 파일을 만들어주고 파일별로 완전 새로운 디자인을 생성해 줘. 각 디자인은 Subagent가 담당할 수 있도록 해서 병렬로 작업해줘.

11. MCP 사용하기

MCP란?

모델 컨텍스트 프로토콜(MCP)은 LLM이 데이터베이스, API, 파일 시스템 등 외부 도구와 직접 상호작용할 수 있게 만드는 강력한 브릿지입니다.



구분	stdio 서버	SSE 서버	HTTP 서버
실행 위치	로컬 컴퓨터	원격 서버	원격 서버
통신 방향	양방향 (프로세스 통신)	단방향 (서버 → 클라이언트)	양방향 (요청/응답)
핵심 용도	로컬 도구 및 스크립트 연동	실시간 데이터 스트리밍	일반적인 원격 API 연동

MCP설정하기

로컬발식으로 MCP 추가

명령어로 추가 하기 : 문법

claude mcp add <name> <command> [args…]

.mcp.json에 직접

~~현재~~ 프로젝트의 파일 이름에 해당하며, 명령의 정의

```
{
"mpServers": {
        "context7" : {
                "type" : "stdio",
                        "conmand": "npx",
                        "args": [
                        "-y",
                        "@upstash/context7-mcp"
                        ],
                "env": 03
                }
        }
}
```

SSE MCP 추가하기

명령어로 추가 하기 : 문법

claude mcp add –transport sse <name> <command [args…]

.mcp.json에 직접

~~현재~~ 프로젝트의 파일 이름에 해당하며, 명령의 정의

```
{
"mpServers": {
        "context7" : {
                "type" : "sse",
                "url":"https://mcp.context7.com/sse"
                }
        }
}
```

HTTP MCP 추가하기

명령어로 추가 하기 : 문법

claude mcp add –transport http <name> <command> [args…]

.mcp.json에 직접

~~현재~~ 프로젝트의 파일 이름에 해당하며, 명령의 정의

```
{
"mpServers": {
        "context7" : {
                "type" : "http",
                "command": "npx",
                "args": ]
                "-y",
                "@upstash/context7-mcp"
                }]
                "env":{}
        }
    }
}
```

유용한 MCP리스트

MCP 이름	기능
Postgresql, Mongodb, Mysql	데이터베이스에 접근할 수 있습니다. 실제 데이터베이스에 어떤 데이터가 있는지 조회하고 마이그레이션 계획을 짤 때 유용합니다.
Playwright, Puppeteer	MCP로 브라우저를 조종할 수 있습니다. 엔드투엔드 테스트를 하거나 크롤링할 때 유용합니다.
context7	각종 개발 도구의 가장 최근 공식 문서를 가져옵니다.
MagicUI	21st Dev의 아름다운 UI 컴포넌트들을 적용할 수 있습니다.
GitHub	깃허브, 깃 기능을 실행할 수 있습니다.
TossPayments, evenueCat	결제 관련 기능을 구현할 수 있습니다.
Supabase	Supabase에 연결할 수 있습니다.

12. PRD와 실행 계획하기

PRD와 실행 계획

PRD (제품 요구사항 명세서)

What & Why

무엇을 만들고 왜 만들 것인지에 대한 문서입니다.(누가보더라도 이해 가능)

방향 설정
프로젝트의 방향을 명확히 설정하고, 서비스 관점을 조망

핵심 질문에 답함
어떤 사용자를 위해?
어떤 문제를 해결해 주는가?
어떤 기능과 경험을 제공해야 하는가?

AI 의도 파악
클로드와 같은 AI 에이전트가 사용자의 의도를 파악하도록 돕습니다

상호 보완

실행 계획 (Execution Plan)

How & When

구현 방법과 일정에 대한 기술적 내용입니다.

기술 명세
API 형식, 사용할 라이브러리, 에러 처리 방식 등 구체적인 기술적 결정

작업 분해
큰 문제를 작게 쪼개어 명확히 정의하고 실행 가능한 문제로 분해

의존성 파악
작업 간의 선후 관계를 정의하고, 언제 완료할지 일정을 세움

실행계획 작성법

1

작업 분해

PRD의 사용자 스토리(요구사항)를 가져와, 이를 구현하는 데 필요한 모든 기술 작업을 작게 분할합니다.

예: 프론트엔드 UI 목업, 백엔드 OAuth 콜백 개발, User 테이블에 소셜 ID 컬럼 추가 등

2

기술 명세

각 작업에 대한 구체적인 기술적 결정(API 형식, 사용할 라이브러리, 에러 처리 등)을 명시합니다.

목적: AI 에이전트에게 프롬프팅하는 데 필요한 구체적인 기술적 지침 제공

3

의존성 파악

작업 간의 선후 관계를 정의합니다. 서로 독립적인 영역이거나 의존성이 없는 태스크는 동시에 개발할 수 있도록 명시합니다.

중요성: AI 에이전트가 프로젝트 전체를 파악하지 못하는 것을 방지

4

일정 설정

각 작업이 얼마나 걸릴지 예측하고 언제 완료할지 현실적인 일정을 설정합니다.

목표: AI 에이전트가 효율적인 작업 순서를 결정하도록 가이드

분해 방법론의 핵심 이점

- AI의 컨텍스트 제약 극복: 큰 문제를 작게 쪼개어 AI가 처리할 수 있는 단위로 만듭니다.
- 병렬 처리 가능: 독립적인 작업을 동시에 처리할 수 있는 구조를 만듭니다.
- 서브 에이전트 활용: 분해된 작업을 독립적인 서브 에이전트에게 할당할 수 있습니다.AI 기반 개발에서 서브 에이전트를 활용한 토큰 컨텍스트 최적화

파일 이름 : PRD-instruction.md

문제 정의
오프라인 매장을 운영하는 소상공인 중 약 70%가 온라인 판매를 원하지만, 기존 쇼핑몰 플랫폼은 평균 30개 이상의 불필요한 기능과 10%에 달하는 높은 수수료 때문에 진입 장벽이 매우 높습니다.

타겟 사용자 및 사용 사례
이 제품을 사용할 핵심 사용자
자신만의 개성 있는 상품을 판매하는 30~40대 소상공인. SNS는 익숙하지만 코딩이나 웹사이트 제작 경험은 없는 분들.

이 기능을 사용하는 목표
매장에서 신상품이 입고되었을 때, 즉시 스마트폰으로 사진을 찍어 상품을 등록하고 싶을 때. 고객의 주문이 들어오면 앱 푸시 알림을 받고 바로 주문을 확인하고 배송 처리를 하고 싶을 때.

제안 해결책
스마트폰 앱 하나로 상품 등록부터 주문 관리, 결제까지 모든 것을 해결할 수 있는 '올인원 모바일 쇼핑몰' 솔루션을 제공합니다. 직관적인 UI를 통해 누구나 1시간 안에 자신만의 온라인 스토어를 열 수 있습니다.

목표 및 성공 지표목표: 소상공인의 온라인 커머스 진입 장벽을 낮춘다.
성공 지표: 론칭 후 6개월 내에 활성 스토어 500개 확보. / 플랫폼을 통한 연간 누적 거래액 10억 원 달성.

파일 이름 : execution_plan.md

실행 계획 제작 커맨드
너는 지금부터 복잡한 개발 작업을 아규먼트로 받아서 독립적이고 관리 가능한 태스크로 분해하고 실행 계획을 작성하는 실행계획 설계 전문가야.

실행
작업은 아래 순서대로 실행해줘.
- 태스크 분석
- 태스크 분해
- 분해된 이슈 출력
- 사용자가 확인할 수 있도록 모든 분해된 이슈를 콘솔에 출력

핵심 목적
제시된 개발 태스크를 다음과 같이 변환하기
- 적절하게 태스크 분류 (백엔드/프런트엔드/풀스택/기타)
- 서로 독립적이고 영역을 침범하지 않는 태스크
- 서로에게 의존해야 한다면 "의존성"으로 해당 태스크 등록하기
- 200K 단일 컨택스트 내에 완료할 수 있는 분량의 태스크

실행 방법
태스크 분석 프로세스
[태스크 분석 프로세스 상세히 설명]
태스크 분해 프로세스
[태스크 분해 프로세스 상세히 설명]
분해된 이슈 출력 프로세스
[분해된 이슈를 어떤 포맷을 출력해야 하는지 상세히 설명]

13. 에이전트 병렬로 실행하기

서브 에이전트 구조와 효율성 원칙

서브 에이전트는 메인 에이전트를 대신하여 작업을 수행하는 독립적인 에이전트로, 병렬 처리와 효율성을 극대화합니다.

직렬 실행

- 서브 에이전트 생성 시마다 새로운 200K 토큰 컨텍스트가 생성됩니다
- 메인 에이전트의 컨텍스트를 소비하지 않고 독립적인 작업을 수행합니다
- 작업 결과와 요약만 반환하여 토큰을 아낄 수 있습니다

"서브 에이전트는 메인 에이전트의 컨텍스트를 아끼면서 효율적인 작업이 가능합니다."

병렬 실행

- 서브 에이전트를 활용하는 가장 쉬운 방법이며 중요한 용도입니다
- 서브 에이전트를 통해 서로 독립적인 작업을 병렬로 처리합니다
- 시간을 절약하고 작업 효율을 높입니다

"한 작업이 끝날 때까지 기다리지 않고 여러 작업을 동시에 처리합니다."

독립적 작업 원칙

- 서브 에이전트는 완전히 새롭게 시작된 에이전트입니다
- 이전 작업에 대한 지식이 전혀 없습니다
- 명확하고 단순한 명령만으로 독립적으로 작동합니다

"작업이 끝난 후 결과만 메인 에이전트에게 전달합니다."AI기반 개발에서 서브 에이전트를

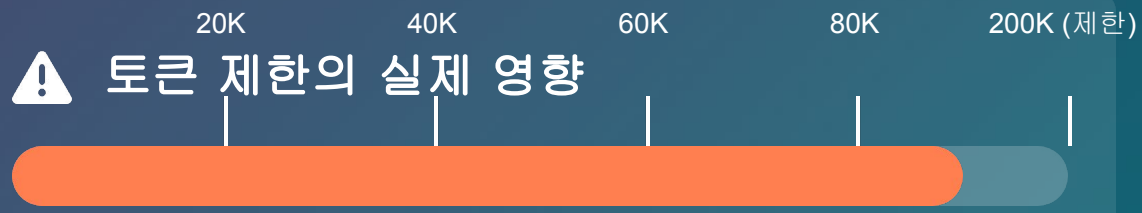
AI 에이전트의 컨텍스트 크기 제한 문제

Claude 200K 토큰 제한

클로드(Claude)의 최대 컨텍스트 사이즈는 200K 토큰입니다. 이 제한은 AI가 처리할 수 있는 정보의 양에 대한 근본적인 제약입니다.

작업 효율 저하

대화가 이 한계를 넘어가면 에이전트는 문맥을 파악하지 못하게 되어 작업 효율이 극도로 떨어집니다. AI는 처리할 수 있는 정보에 한계가 있어서 커다란 문제를 한 번에 해결하지 못합니다.



토큰 제한의 실제 영향

- 긴 대화로 인한 메모리 오버플로로 작업 실패
- 문맥을 기억하지 못해 작업의 연속성을 끊음
- 복잡한 문제를 분해하지 못해 해결하지 못함
- 해결책: 서브 에이전트를 통해 작업 분리

병렬 실행을 통한 효율성 극대화

병렬 실행은 여러 에이전트를 동시에 작동시켜 병렬로 처리하는 것을 의미하며, 시간을 절약하고 효율성을 높입니다. 서브 에이전트를 활용하면 서로 독립적인 작업을 동시에 처리할 수 있습니다.

다국어 작업

다국어 번역을 할 때 여러 언어를 독립적으로 나누어 동시에 병렬로 번역할 수 있습니다. 각 언어에 특화된 서브 에이전트에게 작업을 분배하면 시간을 크게 아낄 수 있습니다.

테스트 분량

많은 분량의 테스트를 독립적으로 나누어 동시에 병렬로 작업할 수 있습니다. 서브 에이전트는 테스트 케이스를 분배받고, 각각의 결과를 병렬로 처리합니다.

디자인 시안

여러 디자인 시안을 동시에 병렬로 작업할 수 있습니다. 서로 다른 디자인 방안을 동시에 개발하고, 결과를 비교하는 데 도움이 됩니다.

병렬 실행의 주요 이점

- 작업 시간 단축: 동시에 여러 작업을 처리함으로써 총 작업 시간 감소
- 작업 간 충돌 방지: 독립적인 작업으로 인한 충돌 없음
- 에이전트 부담 분산: 메인 에이전트의 컨텍스트 오버헤드 줄임
- 리소스 최적화: 필요한 시점에만 에이전트를 실행

서브 에이전트를 통한 컨텍스트 분리 전략

메인 에이전트

- 작업 전체 컨텍스트를 관리하지 않음
- 서브 에이전트에게 독립적인 작업 위임
- 작업 결과와 요약만 받음

메인 에이전트 컨텍스트 사용

작업 관리에 필요한 최소한의 컨텍스트만 유지

서브 에이전트 생성

- 새로운 200K 토큰의 컨텍스트 생성
- 독립적인 작업 수행
- 작업 결과와 요약만 반환

서브 에이전트 컨텍스트 사용

작업에 필요한 전체 컨텍스트 사용

토큰 절약 효과

메인 에이전트는 서브 에이전트가 작업하는 동안 발생한 모든 상세한 컨텍스트를 포함하지 않고 결과만 받기 때문에, 토큰을 크게 절약할 수 있습니다.

서브 에이전트 토큰 사용	127.6K
메인 에이전트 토큰 사용	215

토큰 절약 결과

메인 에이전트가 127K 이상의 토큰 컨텍스트를 아낄 수 있음

토큰 절약 효과 실제 사례

토큰 사용량 비교

극대적인 토큰 절약

테스트 실행 및 커버리지 개선 작업에서 서브 에이전트는 127.6K 토큰을 사용했지만, 실제 메인 에이전트는 단 215 토큰만 사용했습니다.

메인 에이전트가 컨텍스트에서 127K 이상의 토큰을 아꼈다는 것을 의미합니다!

토큰 절약 메커니즘

- 서브 에이전트에게 부분 작업을 위임하고 결과만 받음
- 상세한 로그, 린트 결과, 빌드 출력 등은 서브 에이전트에서 처리
- 메인 에이전트는 작업 결과와 요약만 받아 컨텍스트를 효율적으로 관리

실제 구현을 위한 베스트 프랙티스

컨텍스트 관리 최적화

- 서브 에이전트에게 필요한 최소한의 컨텍스트만 제공
- 작업 결과와 요약만 메인 에이전트로 반환하도록 설계
- 서브 에이전트는 독립적이고 명확한 작업만 수행하도록 지시

구현 시 주의사항

- 서브 에이전트 간의 충돌 방지를 위한 작업 분리
- Git Worktree를 사용해 독립적인 개발 환경을 설정
- 커스텀 명령어를 통해 반복적인 작업을 자동화

워크플로우 설계 가이드라인

- 작업 분해 시 의존성과 순서를 명확히 정의
- 병렬로 처리할 수 있는 작업끼리 그룹화
- 커스텀 명령어와 서브 에이전트를 조합한 복합적 워크플로우 설계

일반적인 문제점과 해결책

- 메인 에이전트의 컨텍스트 오버헤드: 서브 에이전트에게 작업 위임
- 작업 의존성 관리: 실행 계획에서 의존성을 명확히 정의
- 결과 병합 문제: 각 서브 에이전트의 결과를 적절히 조합

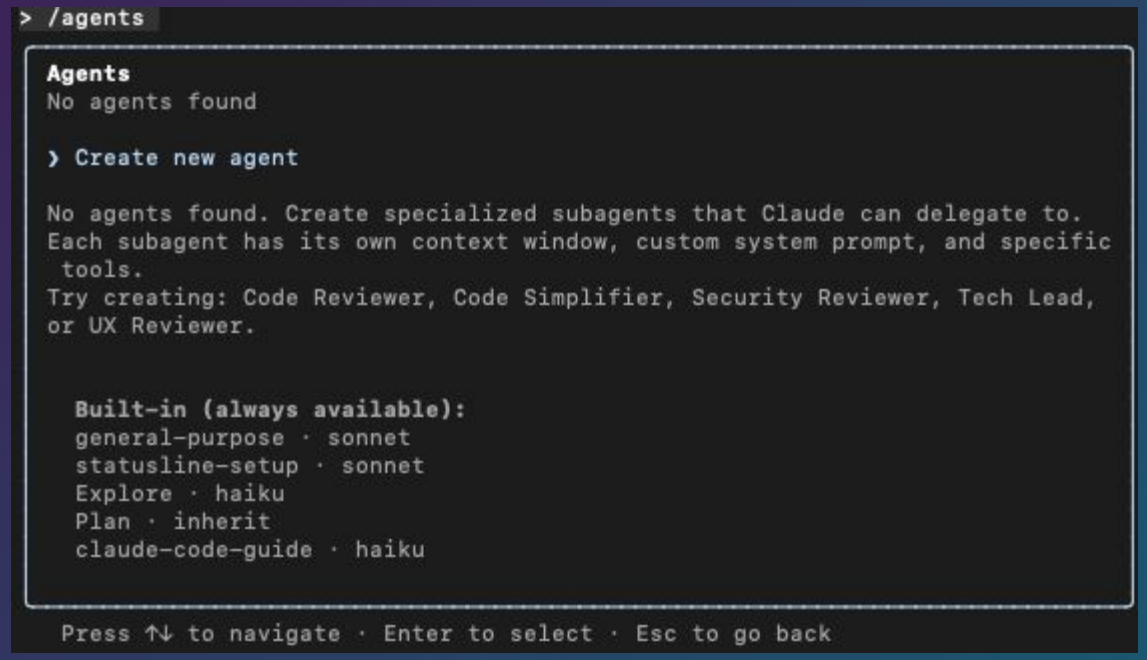
결론: 서브 에이전트를 효과적으로 사용하면 AI의 컨텍스트 제약을 극복하고 병렬 처리를 통해 작업 효율을 극대화할 수 있습니다. 실행 계획을 통해 큰 문제를 작게 쪼개고, 서브 에이전트를 통해 독립적인 작업을 처리하는 것이 토큰 컨텍스트를 최적화하는 가장 좋은 방법입니다.

커스텀 서브에이전트와 독립 에이전트

커스텀 서브에이전트: 에이전트를 정의해서 사용

커스텀 서브 에이전트 생성법

/agent 커맨트 또는 마크다운 파일을 생성하는 방식



독립 에이전트: 하나의 에이전트에서 task를 전담하는 방식으로 생성함. 여러개의 클로드 코드를 여러 터미널 창에서 동시에 실행

- 작업 환경 분리가 중요함
- git worktree를 이용하여 작업 공간 분리하기

Git Worktree를 활용한 병렬 개발 환경

독립적인 개발 환경 구성

Git Worktree를 사용하면 동일한 프로젝트에 대한 여러 개의 독립적인 작업 환경을 생성할 수 있습니다. 이를 통해 서브 에이전트는 서로 격리된 공간에서 작업을 수행할 수 있습니다.

Worktree 설정 방법

- git worktree 생성하기

  git worktree add [폴더 위치]
  git worktree add ../feature-1

- git worktree 생성하여 브랜치 이름 지정하기

  git wortree add[폴더 위치] [브랜치 이름]
  git worktree add ../feature-1 -b todo-1

- git worktree 삭제 하기

  git worktree remove [폴더위치]

- git worktree 정리하기

  git worktree prune

GitHub 이슈 연동과 자동화 워크플로우

분해된 태스크의 이슈 생성

분해된 작업 목록은 깃허브 이슈(GitHub Issue) 형태로 구체화됩니다.

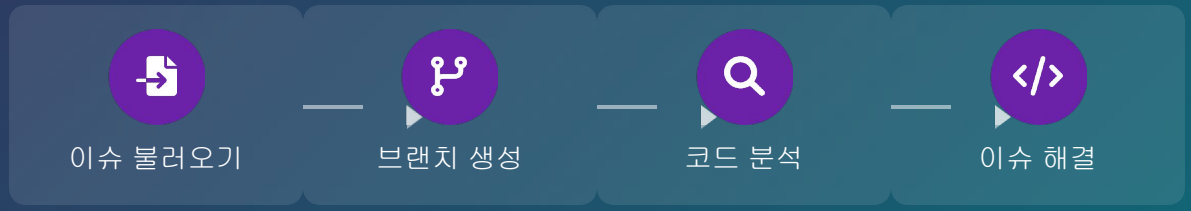
이슈는 다음과 같은 속성을 가집니다:

- 개발 영역(프런트엔드, 백엔드) 레이블
- 복잡도와 작업 유형 레이블
- 의존성 관계 정의
- 명확한 작업 지시와 기술적 세부사항

커스텀 명령어 사용

```
/decompose-issue <PRD/큰 이슈>
```

/resolve-issue 명령어를 통한 자동화



자동화 워크플로우 단계

1. 이슈를 불러오고, 코드베이스를 분석합니다
2. 해결 계획을 세우고, 서브 에이전트에게 작업을 위임합니다
3. 테스트를 작성하고, 린트와 빌드를 수행합니다
4. 검증을 통해 풀 리퀘스트를 생성합니다

토큰 컨텍스트 최적화 이점

- 작업 분해로 컨텍스트 오버헤드 감소
- 병렬 처리로 개발 시간 단축
- 자동화로 반복 작업 제거