

Claude Code 실전 활용

팀별 AI 에이전트 설계 가이드

개발팀, 영업팀, 회사 전체를 위한 맞춤형 AI 자동화 전략

대상
전사 임직원

발표 시간
45-50분

슬라이드
45장

목차

02

오늘 다룰 내용

1. 왜 팀별 활용이 중요한가?

2. 개발팀 활용법 & PR 리뷰 에이전트

3. 영업팀 활용법 & 경쟁사 분석 에이전트

4. 디자인팀 활용법 & Figma 연동

5. 회사 전체 활용법 & 리서치 에이전트

6. 에이전트 설계 방법론

7. 도입 가이드 & 마무리

핵심 메시지

Claude Code는 개발팀만의 도구가 아닙니다. 영업, 경영지원, HR 등 모든 팀이 각자의 반복 업무를 자동화할 수 있습니다.

Situation - 현재 상황

03

AI 코딩 도구, 개발팀만 쓰고 있나요?

92%

개발자가 AI 코딩 도구 사용 (2025)

?%

영업팀의 AI 도구 활용률

?%

경영지원팀의 AI 활용률

AI의 진짜 가치는 "전사적 활용"에서 나온다

질문: 우리 회사는 Claude Code를 어디까지 쓰고 있나요?

Complication - 문제점

04

팀별 업무 특성이 다르다

팀	주요 업무	반복 작업
개발팀	코드 작성 / 리뷰	PR 리뷰, 버그 분석, 문서화
영업팀	고객 대응 / 제안	경쟁사 조사, 제안서, 고객사 분석
경영지원	행정 / 리서치	지원사업 검토, 보고서, 회의록

핵심

각 팀에 맞는 AI 활용법이 필요하다

Question - 핵심 질문

05

어떻게 팀별로 최적화할 것인가?

개발팀: PR 리뷰를 자동화할 수 있을까?

영업팀: 경쟁사 동향을 실시간 추적?

경영지원: 지원사업 공고를 자동 요약?

답

**Skills + MCP + Agent 조합
으로**

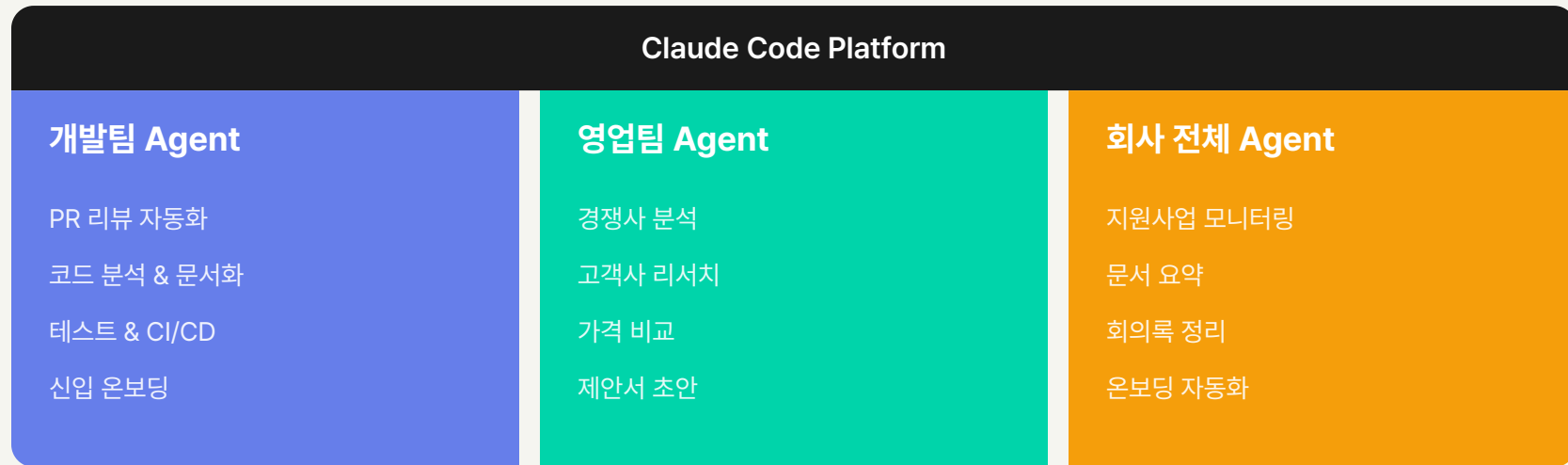
팀 맞춤형 AI 자동화가 가능합니다

연간 \$850,000 절감 사례 (Anthropic 기업 고객 평균)

Answer - 해결책

06

팀별 맞춤형 에이전트



각 팀의 반복 업무를 Skills + MCP로 자동화

PART 2

개발팀 활용법

PR 리뷰부터 CI/CD 자동화까지

PR 처리 시간

30%↓

온보딩 기간

80%↓

버그 분석 시간

67%↓

개발팀의 Pain Points

08

반복되는 업무들

PR 리뷰에 하루 2-3시간 소요

신입 온보딩에 시니어 시간 낭비

버그 분석을 위한 로그 파싱

코드베이스 문서화 미비

결과

개발 속도 저하 기술 부채 누적

실제 개발 시간보다 부수 작업에 더 많은 시간 소요

개발팀 생산성의 가장 큰 적은 "반복 작업"

활용 사례 1: PR 리뷰 자동화

09

30% 빠른 PR 처리

PR 리뷰 요청

> "이 PR 리뷰해줘.
보안 이슈와 성능 문제 중심으로"

Claude Code 실행:

1. 변경된 파일 분석
2. 코드 스타일 검사
3. 보안 취약점 탐지
4. 성능 이슈 식별
5. 우선순위별 피드백 제공

결과 예시

- 🔴 Critical: SQL Injection 위험
- 🟡 Warning: N+1 쿼리 발견
- 🟢 Suggestion: 변수명 개선 권장

Pilot 프로그램 결과

PR 처리 시간 30% 단축
코드 리뷰 품질 향상
팀 협업 효율 증가

활용 사례 2: 신입 온보딩

몇 주 → 1-2일로 단축

신입 개발자의 첫날

> "이 프로젝트 아키텍처 설명해줘"

> "인증 로직은 어디서 처리해?"

> "테스트 실행 방법 알려줘"

Claude Code:

- 전체 코드베이스 분석
- 아키텍처 다이어그램 생성
- 주요 패턴 및 컨벤션 설명

Before

코드베이스 파악 2-3주

After

코드베이스 파악 1-2일

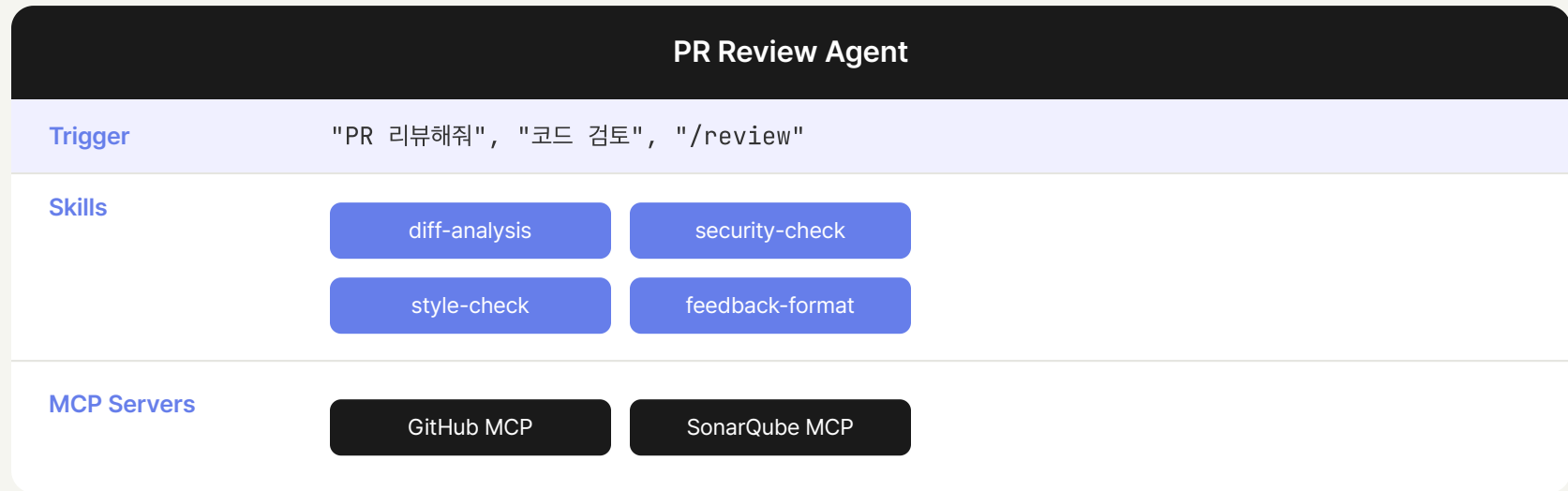
시니어 개발자 시간 절약

신입이 Claude Code에게 질문 → 시니어는 리뷰에 집중

PR 리뷰 에이전트 설계

11

에이전트 구조



Skills로 검토 로직 정의 + MCP로 GitHub/CI 도구 연결 = 완전 자동화된 PR 리뷰

PR 리뷰 에이전트 - SKILL.md

12

에이전트 정의 파일

```
# .claude/agents/pr-reviewer.md
```

```
---
```

```
name: pr-reviewer
```

```
description: 팀 표준 기반 PR 리뷰
```

```
allowed-tools: Read, Grep, Glob, Bash(git:*)
```

```
---
```

```
# PR 리뷰 프로세스
```

```
1. git diff로 변경 파일 목록 확인
```

```
2. 각 파일별 변경 내용 분석
```

```
3. 우선순위별 피드백 분류:
```

```
● Critical: 머지 전 필수 수정
```

```
● Warning: 수정 권장
```

```
● Suggestion: 개선 고려
```

```
4. CLAUDE.md 규칙 기반 검토
```

개발팀 활용 - ROI

13

숫자로 보는 효과

지표	Before	After	개선
PR 처리 시간	4시간	2.8시간	30%↓
온보딩 기간	2주	2-3일	80%↓
버그 분석 시간	15분	5분	67%↓
솔루션 설계	1주	3일	45%↓

출처: Anthropic 고객 사례, Salesforce 엔지니어링 팀, Altana

PART 3

영업팀 활용법

경쟁사 분석부터 고객 맞춤 제안까지

경쟁사 조사

87%↓

고객사 리서치

81%↓

고객 대응 시간

50%↑

영업팀의 Pain Points

15

반복되는 업무들

경쟁사 동향 파악에 매일 1-2시간

고객사 정보 수집에 제안서당 3-4시간

회의록 정리 및 후속 조치 정리

제안서/발표자료 작성 시간 과다

결과

실제 고객 대응 시간 부족

리서치에 시간을 뺏겨 정작 고객과의 대화에 집중하지 못함

리서치 자동화 = 고객 대응 시간 확보

활용 사례 1: 경쟁사 뉴스 모니터링

실시간 경쟁사 동향 파악

매일 아침 경쟁사 뉴스 요약

> "지난 24시간 A사, B사, C사
관련 뉴스 요약해줘.
우리 제품 관련 내용 중심으로"

Claude Code + Web Search:

1. 경쟁사별 뉴스 검색
2. 핵심 내용 추출
3. 우리 비즈니스 영향도 분석
4. Slack 채널에 요약 공유

Before

매일 2시간 직접 검색

After

매일 15분 리포트 확인

자동화 설정 후 매일 아침 Slack에 경쟁사 동향 요약 수신

활용 사례 2: 고객사 리서치

17

제안 전 고객사 심층 분석

미팅 전 고객사 조사

> "다음 주 미팅 예정인 X사 분석해줘.
최근 투자 동향, 기술 스택,
주요 과제 중심으로"

Claude Code 실행:

- 공개 IR 자료 분석
- 기술 블로그/GitHub 분석
- 채용 공고로 기술 스택 파악
- 맞춤 제안 포인트 도출

분석 결과 예시

X사 기업 분석

시리즈 B 투자 유치 (100억)

기술 스택: React, Node.js

주요 과제: 레거시 마이그레이션

최근 채용: SRE 3명

제안 포인트

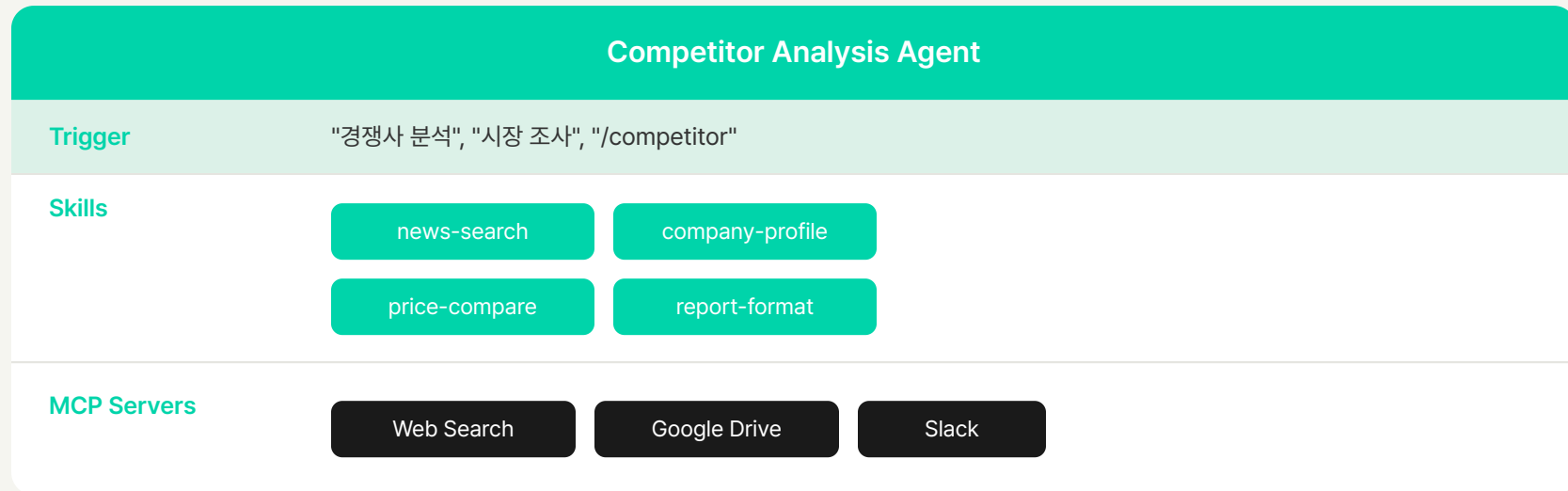
마이그레이션 자동화 솔루션 중심 제안 권장

4시간 리서치 → 45분으로 단축 (81% 절감)

경쟁사 분석 에이전트 설계

18

에이전트 구조



Web Search + Google Drive + Slack = 자동 리서치 → 저장 → 팀 공유까지 원스톱

영업팀 활용 - ROI

19

숫자로 보는 효과

지표	Before	After	개선
경쟁사 조사	2시간/일	15분/일	87%↓
고객사 리서치	4시간	45분	81%↓
제안서 작성	8시간	3시간	62%↓
고객 대응 시간	4시간/일	6시간/일	50%↑

리서치 시간 절감 → 고객 대응 시간 50% 증가

PART 4

디자인팀 활용법

Figma 연동부터 디자인 시스템 자동화까지

디자인-코드 핸드오프

80%↓

광고 배리에이션

100x

디자인 QA 시간

90%↓

디자인팀의 반복 업무들

AI로 자동화할 수 있는 영역

디자인-개발 핸드오프

스펙 문서 작성, 토큰 값 전달, 수정 사항 커뮤니케이션에 하루 2-3시간

디자인 시스템 QA

코드와 디자인 토큰 불일치, 컴포넌트 스펙 검증에 많은 시간 소요

광고/배너 배리어이션

동일 디자인에 문구/이미지만 바꾸는 반복 작업, 건당 30분 소요

프로토타입-코드 전환

Figma 프로토타입을 실제 코드로 구현하는 과정에서 디테일 손실

결과

디자인-개발 사이클 지연, 디자인 품질 저하, 크리에이티브 업무에 집중할 시간 부족

활용 사례 1: Figma MCP 연동

Design-to-Code 자동화

```
# Figma 디자인을 코드로 변환
```

```
> "이 Figma 프레임을 React  
컴포넌트로 변환해줘.  
디자인 시스템 토큰 사용해서"
```

Claude Code + Figma MCP:

1. Figma 파일 구조 분석
2. 디자인 토큰 자동 매핑
3. 컴포넌트 코드 생성
4. Storybook 문서 자동 생성

Before

디자이너-개발자
왕복 커뮤니케이션 5회+

After

정확한 스펙의 코드
자동 생성 → 즉시 리뷰

핸드오프 시간 80% 절감 + 디자인-코드 불일치 0%

활용 사례 2: 디자인 시스템 QA

토큰 불일치 자동 감지

디자인 시스템 검증 자동화

> "Figma 디자인 토큰과 코드의
CSS 변수 일치 여부 검사해줘"

Claude Code 실행:

1. Figma에서 디자인 토큰 추출
2. 코드베이스 CSS 변수 스캔
3. 불일치 항목 리스트업
4. 수정 제안 or 자동 수정

Figma "Check Designs" AI와 연동

자동 QA 결과 예시

color-primary: #667eea

Code: #6366f1 (불일치)

spacing-md: 16px

Code: 14px (불일치)

border-radius: 12px

Code: 12px (일치)

2개 불일치 → 자동 수정 제안

디자인 시스템 일관성 유지 + QA 시간 90% 절감

활용 사례 3: 광고 배리에이션

Anthropic 마케팅팀 실제 사례

광고 배리에이션 자동 생성

> "이 광고 템플릿으로
100개 배리에이션 만들어줘.
헤드라인/설명 조합: ads.csv"

Claude Code + Figma Plugin:

1. 템플릿 프레임 식별
2. CSV에서 텍스트 조합 로드
3. 100개 프레임 자동 생성

→ 배치당 0.5초 (기존 30분)

Before

100개 × 30분 = 50시간

After

100개 × 0.5초 = 50초

"몇 시간의 복불이 0.5초로" - Anthropic Growth

마케팅/퍼포먼스팀 필수 도구 → A/B 테스트 속도 100배 향상

디자인 에이전트 설계

Figma MCP + Design System 자동화

Design Workflow Agent |

Trigger: "디자인 코드로", |
"토큰 검증" |

Skills: |

└─ figma-to-code (변환) |
└─ token-validate (검증) |
└─ batch-generate (배리레이션) |
└─ storybook-doc (문서화) |

MCP Servers: |

└─ Figma Dev Mode MCP |
└─ Browser MCP (검증) |

Figma MCP 설정

```
// .claude/.mcp.json  
"figma": {  
  "url": "http://127.0.0.1:3845/sse"  
}
```

Figma Desktop 앱 필요

VS Code, Cursor, Claude Desktop 지원

디자인팀 도입 체크리스트

- ☐ Figma Desktop 설치 및 MCP 활성화
- ☐ 디자인 토큰 정의 (variables)
- ☐ CLAUDE.md에 컴포넌트 규칙 명시

PART 5

회사 전체 활용법

지원사업 분석부터 문서 요약까지

지원사업 검토

90%↓

문서 요약

92%↓

연간 절감

\$850K

활용 사례 1: 지원사업 모니터링

21

맞춤 지원사업 알림

정부 지원사업 자동 모니터링

> "우리 회사가 신청 가능한
R&D 지원사업 찾아서 요약해줘."

업종: IT 서비스

규모: 중소기업

매출: 50억 이하"

Claude Code 실행:

1. 지원사업 사이트 검색
2. 자격 요건 매칭
3. 핵심 정보 추출 (마감일, 지원금)
4. 우선순위 정렬 보고서 생성

Before

매주 5시간
사이트별 직접 검색

After

매주 30분
맞춤 보고서 확인

90% 시간 절감 + 놓치는 지원사업 0%

활용 사례 2: 문서 요약

22

500K 컨텍스트의 힘

긴 문서 분석 및 요약

> "이 100페이지 보고서 분석해줘.
핵심 인사이트 5개와
우리 사업 관련 시사점 정리해줘"

Claude Code (Enterprise):

→ 500K 토큰 = 수백 페이지 문서
→ 전체 맥락 이해 후 요약

500K 컨텍스트 윈도우

수백 개 영업 트랜스크립트
수십 개 100+ 페이지 문서
중간 규모 코드베이스

한 번에 전체 문서 분석 가능 → 맥락 손실 없음

Before: 100페이지 문서 요약

2시간 읽고 정리

After: Claude Code

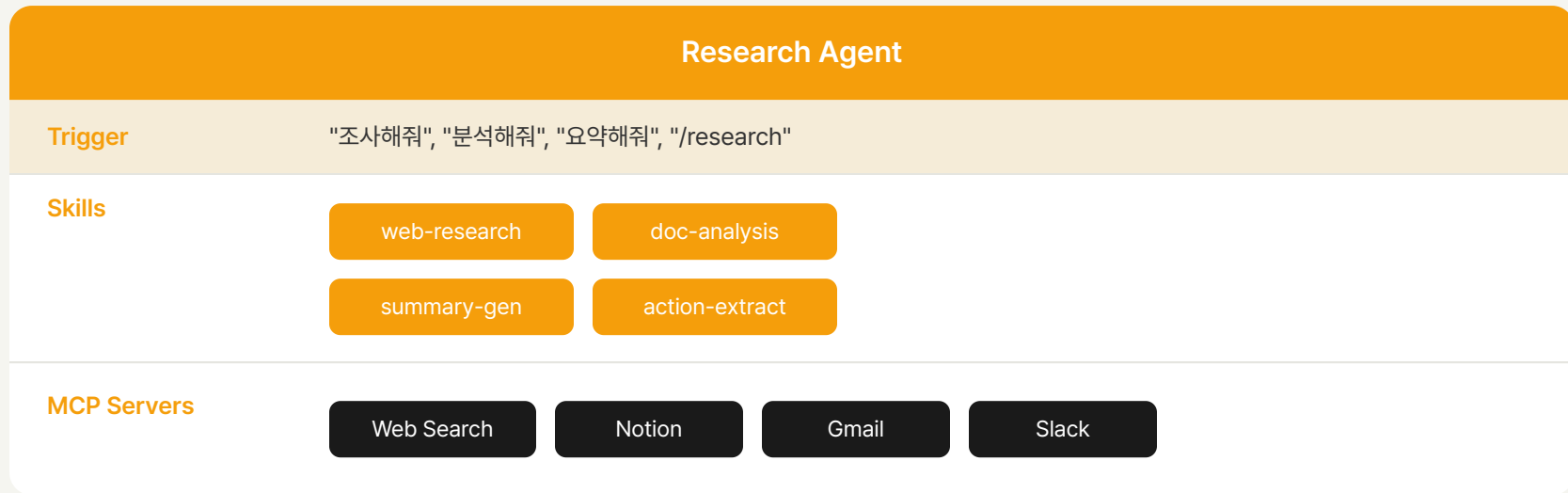
10분 분석 완료 (92%↓)

Enterprise 플랜: 500K 컨텍스트 윈도우 제공

리서치 에이전트 설계

23

범용 리서치 에이전트 구조



검색 → 분석 → 요약 → 액션아이템 추출 → Notion/Slack 공유까지 자동화

회사 전체 활용 - ROI

24

숫자로 보는 효과

지표	Before	After	개선
지원사업 검토	5시간/주	30분/주	90%↓
문서 요약	2시간/건	10분/건	92%↓
온보딩 준비	4시간	30분	87%↓
회의록 정리	30분/건	5분/건	83%↓

Anthropic 기업 고객 평균

연간 절감 효과: \$850,000

PART 6

에이전트 설계 방법론

팀 맞춤형 AI 자동화 구축하기

1. IDENTIFY

반복 업무 식별

2. DESIGN

에이전트 설계

3. DEPLOY

배포 & 개선

Step 1: IDENTIFY

26

반복 업무 식별 매트릭스



자동화 우선순위

- A: 최우선 자동화**
빈도 높음 + 자동화 쉬움
PR 리뷰, 경쟁사 모니터링
- B: 반자동화**
빈도 높음 + 판단 필요
제안서 초안, 보고서 작성
- C: 도구 지원**
빈도 낮음 + 자동화 가능
코드 리팩토링, 마이그레이션

A 영역부터 시작해서 점진적으로 확장

Step 2: DESIGN

27

SKILL.md 작성 템플릿

```
# .claude/skills/[skill-name]/SKILL.md
```

```
---
```

```
name: [스킬 이름]
```

```
description: [한 줄 설명]
```

```
allowed-tools: [허용 도구 목록]
```

```
---
```

```
# 트리거 키워드
```

```
"[키워드1]", "[키워드2]"
```

```
# 입력
```

```
[필수 입력 1], [선택 입력 2]
```

```
# 프로세스
```

```
1. [단계 1] 2. [단계 2] 3. [단계 3]
```

```
# 출력 형식
```

```
[출력 포맷: 마크다운, JSON, 리포트 등]
```

MCP 서버 선택 가이드

28

용도별 추천 MCP 서버

용도	MCP 서버	활용 팀
코드 관리	GitHub, GitLab	개발팀
문서 관리	Notion, Google Drive	전체
커뮤니케이션	Slack, Gmail	전체
프로젝트 관리	Jira, Linear	개발/PM
웹 정보	Web Search, Context7	영업/전체

전체 MCP 서버 목록: mcp.cat

Step 3: DEPLOY

29

에이전트 테스트 체크리스트

- 트리거 키워드 정상 동작

- 입력값 검증 로직 확인

- 출력 형식 일관성 확인

- 에러 처리 시나리오 테스트

- 권한 설정 적절성 검토

- 비용(토큰 사용량) 측정

- 팀원 피드백 수렴

- Git 커밋 및 팀 공유

작게 시작 → 피드백 수집 → 점진적 개선 → 전사 확대

PART 6

마무리

핵심 테이크어웨이 & 다음 단계

핵심 테이크어웨이

31

오늘 기억할 5가지

1 팀 맞춤형 - 개발/영업/경영지원 각각 다른 활용법

2 에이전트 = Skills + MCP - 조합으로 무한 확장

3 작게 시작 - 한 가지 반복 업무부터 자동화

4 ROI 측정 - 시간 절감 → 비용 절감으로 환산

5 점진적 확장 - 성공 사례 기반으로 전사 확대

다음 단계

내일부터 할 일

이번 주

1. 팀별 반복 업무 3개 식별
2. 가장 ROI 높은 1개 선정
3. 첫 번째 에이전트 초안 작성

이번 달

4. 에이전트 테스트 및 개선
5. 팀 내 공유 및 피드백
6. 성과 측정 (시간 절감)

Quick Start - 오늘 바로 해볼 것

1. 내 업무 중 가장 귀찮은 반복 작업 1개 떠올리기
2. Claude Code에 "이 작업 자동화할 수 있어?"라고 물어보기

리소스

33

더 알아보기

공식 문서

code.claude.com/docs

Best Practices

anthropic.com/engineering

MCP 서버 목록

mcpcat.io

GitHub 커뮤니티

github.com/anthropics/claude-code

팀 내부 리소스

Slack: #claude-code-help

Notion: /AI 활용 가이드

담당자: [담당자 이름]

Office Hour: 매주 금요일 3PM

질문이 있으시면 언제든지 Slack 채널로 문의해주세요!

Q&A

질문 & 답변

특정 팀 활용법
관련 질문

에이전트 설계
관련 질문

도입 계획
관련 질문

감사합니다!