

바이브코딩으로 경험한 기회와 충격

테디노트 이경록

강연자

teddy note by Teddy(이경록)

LangChain Ambassador (South Korea)

teddy note YouTube(구독자 4.4만), Blog (연간 약 60만명)

랭체인 한국어 튜토리얼(langchain-kr) 저자

브레인크루 (주) - AI 교육 / 기업 도입 솔루션 개발

(전) 삼성전자 무선사업부

Application S/W 개발, C-Lab 사내벤처



teddy note **TeddyNote**

@teddynote · 구독자 4.1만명 · 동영상 248개
데이터 분석, 머신러닝, 딥러닝, LLM 에 대한 내용을 다룹니다.
fastcampus.co.kr/data_online_teddy 와 링크 2개

채널 맞춤설정 동영상 관리

바이브 코딩 (Vibe Coding)

바이브 코딩(Vibe Coding)

대규모 언어 모델(LLM)에 프롬프트를 입력하여 문제를 해결하는 AI 기반 프로그래밍 방식

바이브 코딩

문서 [토론](#)

위키백과, 우리 모두의 백과사전.

바이브 코딩(Vibe coding)은 [대규모 언어 모델\(LLM\)](#)에 프롬프트를 입력하여 문제를 해결하는 AI 기반 [프로그래밍](#) 방식이다.

LLM이 코드를 생성하면서 프로그래머의 역할은 직접 코드를 타이핑 하는 것이 아니라, AI와 프롬프트로 소통하며 AI가 생성한 코드 [\[1\]](#)[\[2\]](#)[\[3\]](#)를 테스트하고 수정하고, 가이드하는 방향으로 변화하고 있다.

바이브 코딩(Vibe Coding)은 지지자들에 의해 개발을 잘모르는 초보자도 [소프트웨어 엔지니어링](#)에 대한 훈련과 기술 없이 코드를 생산할 수 있게 해준다고 주장한다.[\[4\]](#)

이 용어는 2025년 2월 안드레이 카파시(Andrej Karpathy)에 의해 소개되었으며[\[5\]](#)[\[2\]](#)[\[4\]](#)[\[1\]](#) 다음 달 Merriam-Webster 사전에 "속어 및 트렌드" 명사로 등재되었다.

바이브 코딩(Vibe Coding)

2025년 2월 안드레이 카파시(Andrej Karpathy)에 의해 소개

“저는 그저 사물을 보고, 말하고, 실행하고, 복사해서 붙여넣기만 하면 대부분 작동한다.”

“실제로 코딩은 아니다”

The screenshot shows a tweet from Andrej Karpathy (@karpathy) with a blue checkmark. The tweet text is:

There's a new kind of coding I call "vibe coding", where you fully give in to the vibes, embrace exponentials, and forget that the code even exists. It's possible because the LLMs (e.g. Cursor Composer w Sonnet) are getting too good. Also I just talk to Composer with SuperWhisper so I barely even touch the keyboard. I ask for the dumbest things like "decrease the padding on the sidebar by half" because I'm too lazy to find it. I "Accept All" always, I don't read the diffs anymore. When I get error messages I just copy paste them in with no comment, usually that fixes it. The code grows beyond my usual comprehension, I'd have to really read through it for a while. Sometimes the LLMs can't fix a bug so I just work around it or ask for random changes until it goes away. It's not too bad for throwaway weekend projects, but still quite amusing. I'm building a project or webapp, but it's not really coding - I just see stuff, say stuff, run stuff, and copy paste stuff, and it mostly works.

At the bottom of the tweet card, there is a timestamp "6:17 PM · Feb 2, 2025" and a view count "4M Views". Below the tweet card, there are standard social media interaction icons: 1.2K comments, 3.9K retweets, 24K likes, 12K saves, and a share icon.

바이브 코딩(Vibe Coding)

바이브 코딩(Vibe Coding)은 지지자들에 의해 개발을 잘모르는 초보자도
소프트웨어 엔지니어링에 대한 훈련과 기술 없이 코드를 생산할 수 있게 해준다고 주장

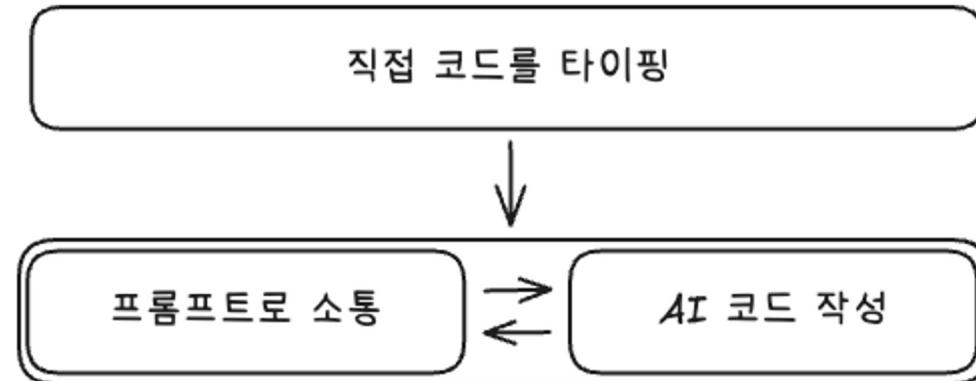
“Accept” 만 누르고 있는 게을러진 나를 발견...



바이브 코딩(Vibe Coding)

프로그래머의 역할

- [ASIS] 직접 코드를 타이핑
- [TOBE] AI가 생성한 코드를 테스트, 수정, 가이드하는 방향으로 변화



바이브 코딩에 열광하는 이유

비개발자

- 코딩을 못하는 사람도 프롬프트만으로 (프로토타입)서비스를 제작 가능

개발자

- Boiler Plate / Unit Test / Git 커밋 히스토리 / [README.md](#) 등 개발자의 생산성을 올리는데 도움을 줌
- 개발 도메인을 확장할 수 있음(Backend 개발자가 Frontend 까지 넓혀서 개발)



<비개발자 Ver>



정말 코딩 지식없이 개발이 가능할까?

코딩을 전혀 몰라도 바이브 코딩으로 개발이 가능한가?

(가능하다)

간단한 프로토타입용(MVP) 서비스 제작에는 가능(50~100명 이하)

Context Length 이하의 작은 규모의 서비스 가능

- 향후 LLM 의 Context Length 가 늘어난다면, 더 큰 규모도 가능할 가능성 있음

수 많은 사례들과 방법론이 공유되고 있고, 실제로 “동작” 하는 앱들이 많아짐

- 즉, 코딩을 모르는 바이브 코더를 위한 노하우 공유가 증가

코딩을 전혀 몰라도 바이브 코딩으로 개발이 가능한가?

(어렵다)

대규모 엔터프라이즈급의 개발은 어렵다

- 기술 부채는 결국엔 큰 리스크로 작용

Context Length 는 제한적

- 프로젝트가 커지는 데에 한계가 있음

개발 지식없이 디버깅을 하기 어렵다

- LLM 에게 오류를 맡겼다가, 프로젝트 전체가 꼬이는 것이 일쑤

빠른 속도로 AI 개발 지식이 쏟아져 나오는데, 최신 코드와 아키텍처 반영이 늦다

- 검색 MCP 도 한계가 있음. 잘못된 코드 검색

그럼에도 “바이브 개발” 을 긍정적으로 바라보는 이유

1. LLM 의 진화
2. Claude Code, Cursor AI 등 코딩 어시스턴트의 진화
3. Context Length 의 증가
4. 더 잘 하는 법! 방법론(PRD 작성, 프롬프트 작성법 등)의 진화

LLM 이 작성한 코드

LLM의 할루시네이션을 문서작업에서는 발견하기 어려울 수 있으나
만약, AI 가 작성한 코드에 “할루시네이션”이 있는 경우 바로 에러가 발생하고,
이에 대한 오류 수정을 스스로 수행하기 때문에, 비교적 안심할 수 있음

LLM 이 작성한 코드에 대한 신뢰

하지만, Syntax Error 가 아닌 **Logical Error** 가 발생하는 경우

예러는 발생하지 않지만 원하는 결과물이 나오지 않거나 제대로된 동작을 하지 않음

따라서, 테스트 주도 개발(TDD) 을 권장하고 있음

Tip 1. 테스트 주도 개발(TDD)

테스트 주도 개발(Test Driven Development)

1.  Red (실패):
 - 먼저 실패하는 테스트 코드를 작성
2.  Green (성공):
 - 테스트를 통과하는 최소한의 코드만 작성
 - 테스트를 통과시키는 것이 목표
3.  Refactor (리팩토링):
 - 코드를 개선하고 최적화
 - 테스트는 여전히 통과해야 함

테스트 주도 개발(Test Driven Development)

TDD 예시 프롬프트

<https://gist.github.com/teddylee777/3601ff6cd31fc7319e4bb94b78bc597d>

1. Test-First Writing (Red Phase)

- * **Feature Analysis**: Analyze and specify the core functionality of the LangGraph workflow
- * **Unit Test Writing**: Write failing test cases for each Node and Edge functionality
- * **Integration Test Writing**: Write test cases for the entire graph execution flow
- * **Test Case Components**:
 - * Node-level input/output validation
 - * State change verification
 - * Conditional routing tests
 - * Error handling scenarios

2. Minimal Implementation (Green Phase)

- * **Basic Structure Implementation**: Write minimal LangGraph structure that passes tests
- * **Node Implementation**: Implement only basic functionality for each node
- * **State Management**: Define state schema for StateGraph
- * **Test Pass Verification**: Ensure all written tests pass

3. Refactoring (Refactor Phase)

- * **Code Optimization**: Remove duplication, improve readability
- * **Architecture Improvement**: Minimize dependencies between nodes
- * **Performance Optimization**: Remove unnecessary LLM calls
- * **Test Maintenance**: Ensure all tests still pass after refactoring

Tip 2. 제품 요구사항 문서(PRD)

제품 요구사항 문서(PRD)

제품 개발 과정에서 만들어지는 핵심 문서

개발할 제품이나 기능에 대한 요구사항과 명세를 체계적으로 정리한 문서

주요 내용

- 목표: 왜 만드나?
- 사용자: 누가 쓰나?
- 기능: 뭘 만드나?
- 요구사항: 어떤 조건을 만족해야 하나?
- 성공지표: 어떻게 측정하나?

제품 요구사항 문서(PRD)

바이브 코더를 위한 PRD 작성 도와주는 서비스

- Cursor AI 와 연동(MCP)

체계적인 바이브코딩을 위한 AI Product Manager

아이디어를 구체화하고, MCP로 연동하고, 흐름을 한눈에 확인하세요.
일관된 작업 흐름을 유지하세요.

→ 무료로 시작하기

커뮤니티

▶ 튜토리얼 보기

출처: vooster.ai/

제품 요구사항 문서(PRD)

PRD 작성 기능 제공

✓ 완료된 답변들

프로젝트의 아이디어를 한 문장으로 설명해 주세요.
LangGraph 로 문서 기반 RAG 시스템

좋아요! 이 문제가 해결되면 사용자는 무엇이 좋아지나요?
질의 응답 방식으로 쉽게 정보 검색 가능

⚡ 질문 3

참고할 유사 서비스가 있다면, 이유와 함께 알려주세요.

없음

없다면 '없음'이라고 입력해주세요. ➔

제품 요구사항 문서(PRD)

생성된 결과 예시

제품 요구사항 문서(PRD)

1. Executive Summary

본 프로젝트는 LangGraph-Python 기반으로 PDF 문서를 자동 분석·임베딩하여 RAG(검색 결합 생성) 챗봇을 만드는 워크플로우입니다. 일반 사용자는 별도 개발 지식 없이 PDF를 업로드하고, 대화형 Q&A로 문서 내용을 즉시 탐색·이해할 수 있습니다. 핵심 가치는 정보 탐색 시간 절감, 정확한 문서 이해, 직관적 UI입니다.

2. Problem Statement

- 방대한 PDF 문서는 검색·이해가 어렵고 시간 소모가 큼
- 기존 검색은 키워드 매칭이라 문맥 이해가 부족
- 일반 사용자는 복잡한 AI 파이프라인을 구축할 수 없음
⇒ 문서 기반 질의응답을 자동화하고 대화형으로 제공하는 솔루션이 필요

3. Goals and Objectives

Primary Goal

- PDF 한 개만 업로드해도 RAG 챗봇이 즉시 생성되어 고품질 문답 제공

제품 요구사항 문서(PRD)

칸반 기능: Cursor AI 와 MCP 로 연동 가능

Backlog 7	In Progress 2	Done 3
T-003 PDF 업로드 UI 마크업 구현(웹) Objective: 웹에서 사용자가 PDF 파일을 업로드할 수 있는 마크업 구조를 작성합니다. Layout Requirements: 반응형 레이아웃, 드래그&드... 긴급도: 8 복잡도: 3 중요도: 필수	T-005 PDF 업로드 백엔드 API 개발 Objective: PDF 파일 업로드 및 저장을 위한 FastAPI 엔드포인트를 개발합니다. API Specification: POST /pdf/upload, 파일 멀티... 긴급도: 8 복잡도: 5 중요도: 필수	T-001 프로젝트 초기 스캐폴딩 및 환경 설정 Objective: 프로젝트의 초기 구조를 생성하고, Next.js/Streamlit(프론트), FastAPI(백엔드), LangGraph 파이프라인, MCP 벡터DB 연동을 위한 기... 긴급도: 9 복잡도: 3 중요도: 필수
T-004 PDF 업로드 UI 로직 구현(웹) Objective: PDF 업로드 UI의 파일 선택, 업로드 진행 표시, 오류 처리 등 인터랙션 로직을 구현합니다. State Management: 업로드 상태, 오류, 성공 이벤트... 긴급도: 8 복잡도: 4 중요도: 필수	T-002 PDF 업로드 데이터베이스 스키마 설계 및 구현 Objective: 업로드된 PDF 문서 메타데이터 및 상태 관리를 위한 데이터베이스 테이블을 설계하고 구현합니다. Schema Requirements: 파일명, 용량, 업로드 시각,... 긴급도: 8 복잡도: 4 중요도: 필수	T-006 PDF 파싱 및 임베딩 파이프라인 구현 Objective: 업로드된 PDF를 파싱, 문단 단위 임베딩 생성 및 MCP 벡터DB 저장 파이프라인을 구현합니다. Schema Requirements: 문단별 텍스트, 페이지 번... 긴급도: 9 복잡도: 7 중요도: 필수
T-007 RAG 챗 인터페이스 UI 마크업 구현(웹) Objective: 웹에서 대화형 질의응답 챗봇 인터페이스의 마크업 구조를 작성합니다. Layout Requirements: 채팅 영역, 입력창, 인용 표... 긴급도: 8 복잡도: 4 중요도: 필수	T-009 RAG 챗 백엔드 API 개발 및 LLM 연동 Objective: 챗봇의 질의응답을 처리하는 FastAPI 엔드포인트 및 LangGraph 기반 RAG 파이프라인을 구현합니다.... 긴급도: 9 복잡도: 8 중요도: 필수	

Tip 3. Git 활용

Git 이란?

코드의 변경 이력을 추적하고 관리하는 도구

**Build and ship software on a
single, collaborative platform**

Join the world's most widely adopted AI-powered developer platform.

Enter your email

Sign up for GitHub

Try GitHub Copilot

Git 이랑 친해져야 하는 이유

오픈소스 프로젝트

레퍼런스 프로젝트로 활용하기 위함

(예시)

“@<https://github.com/teddylee777/gemini-fullstack-langgraph-quickstart>

프로젝트의 아키텍쳐를 참고하여 개발하세요”

Git 이랑 친해져야 하는 이유

소스코드 관리

LLM 의 코드가 실수 할 수 있음. 하지만, 되돌아가기 어렵다는 단점이 존재

코드의 버전 관리가 중요

-o- Commits on Jun 10, 2025

Merge pull request [google-gemini#78](#) from google-gemini/fix-react-serving

 philschmid authored 2 weeks ago

update react serving part

 philschmid committed 2 weeks ago

-o- Commits on Jun 3, 2025

Merge pull request [google-gemini#4](#) from rahimnathwani/patch-1

 philschmid authored 3 weeks ago

Fix typo in prompts.py

 rahimnathwani authored 3 weeks ago

Merge pull request [google-gemini#2](#) from JayDoubleu/main

 philschmid authored 3 weeks ago

Git 이랑 친해져야 하는 이유

필수 명령어

“local” vs “remote” 개념

“commit” vs “push” 의 차이점

“stash” 의 적절한 활용

“branch” 분리 전략: main, hotfix, feature 등

챗GPT 와 물어가며 개념과 키워드는 익히도록 하자. 명령어는 LLM 이 짜줄테니!

Tip 4. UX/UI 용어 익히기

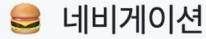
UI 컴포넌트에 대한 용어를 사용하자

LLM 이 잘 이해할 수 있는 용어로 지시하자

네모 버튼 수정해줘 (x)

링크:

<https://gist.github.com/teddylee777/81c8ce12ed21fa95b5f9f23f4b7032e5>



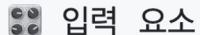
네비게이션

햄버거 메뉴 - 3줄 아이콘 (≡) 모바일 메뉴

브레드크럼 - 경로 표시 (Home > Products > Detail)

탭바 - 하단 고정 메뉴 (iOS 스타일)

사이드바 - 좌/우측 고정 메뉴



입력 요소

슬라이더 - 값 조절 바 (볼륨, 가격 범위)

토글 - ON/OFF 스위치

스테퍼 - +/- 버튼으로 숫자 조절

체크박스 - 다중 선택

라디오버튼 - 단일 선택



꼭 알고 가자!

하지만, 바이브 코딩만으로 상용 서비스가 어려운 이유

서비스 품질

- 개발 경험이 없으면 파악하거나 수정하기 어려움

보안

- 보안 취약점을 가진 코드를 생성할 수 있음

서비스 운영

- 상용 서비스는 단순히 코드만 잘 돌아가는 것이 아니라, 배포, 모니터링, 장애 대응, 데이터 관리, 사용자 지원 등 다양한 운영 요소가 필요

그럼에도 바이브 코딩을 자꾸만 해야 하는 이유

1. 진입 장벽의 극적 하락과 기술의 민주화
2. 아이디어의 빠른 실현과 MVP 제작
3. 실무적 한계와 리스크도 학습의 기회

“비개발자” - “개발자” 사이의 새로운 직군 탄생?!



AI 바이브 코딩 기반 웹프로그래머 모집

지원자격

경력	경력무관
학력	초대졸이상
스킬	API, Spring, GitHub, Gemini, MCP, Prompt, Copilot, Claude, Cursor, 바이브코딩
핵심역량	적응성, 협동심, 성장지향성, 꼼꼼함
우대	기본우대 정보처리기사

근무조건

고용형태	정규직 수습 3개월
급여	회사내규에 따름 - 면접 후 결정
지역	서울시 서초구 지도 >
시간	주5일 (월~금) 08:30~17:30
직책	팀원



<개발자 Ver>



개발자에게 위기인가 기회인가?

개발자의 입장에서 바라본 바이브 코딩

대세는 '바이브 코딩'... "1년 내 AI가 인간 대신 모든 코딩 맡을 것"

△ 임대준 기자 ◎ 입력 2025.03.16 20:10 ◎ 수정 2025.03.17 13:22 ◎ 댓글 1 ◎ 좋아요 0



AI보다 코딩 잘해?…신입 개발자의 비명, 구인공고 19% 급감 [팩플]

중앙일보 | 입력 2025.04.15 06:00

지면보기

오현우 기자

구독

인공지능(AI)의 코딩 실력이 급격히 향상되면서 저연차·저숙련 개발자들 설 자리가 좁아지고 있다. 코로나 19 팬데믹 당시 취업시장을 훨씬었던 '코딩 불패' 공식이 깨짐에 따라 신입 개발자의 취업 문턱은 급격히 올라갈 전망이다.

Cursor AI 창업자 인터뷰

“전문적인 세계에서는 아직 갈 길이 멀어요”

“코드를 실제로 보고 이해하지 않고 코딩하는 것은 실제로는 작동하지 않습니다.”



<https://www.youtube.com/watch?v=oOylEw3tPQ8>

Andrew Ng - DeepLearning.AI

LangChain Interrupt 2025

AI 지원 코딩이 깊은 기술적 이해가 필요하다고 강조

AI 시대에도 코딩을 배우는 것이 여전히 필수적이라고 주장하며, 현대 사회의 문해력에 비유

“AI가 구글의 신규 코드 중 4분의 1 이상을 생성하고 있음에도 불구하고, 코딩을 배우는 것이 여전히 중요하다고 강조”

“좋은 코드를 승인하려면 좋은 코드를 알아야 한다”



프로그래밍 패러다임 변화

비효율성의 해방

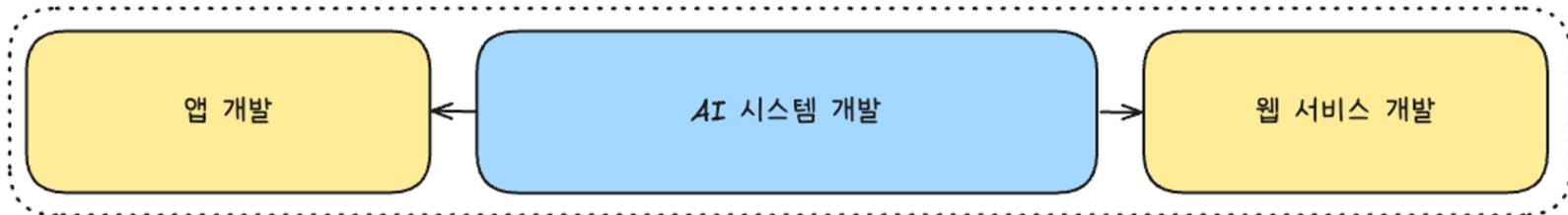
AI가 코딩의 복잡성을 해결해주고, 인간은 **더 높은 차원의 창작과 설계에 집중**할 수 있음

새로운 패러다임

개발자는 비효율적인 노동에서 벗어나 **순수한 아이디어와 비전을 다루게 될 것**

바이브코딩은 개발자 대체가 아닌 역량을 확장

한 도메인만 잘하는 개발자는 앞으로 AI에 의하여 대체될 수도 있음



개발자의 마음 가짐 변화

AI 를 도구로써 활용하자

- 게으르다고 생각 없이 모든 과정을 “Accept” 한다면, 곧 대체될 수 있음

비효율적인 일은 AI에게 위임하자

- 대신, AI 가 생성한 모든 코드를 내가 검증하고, 설명할 수 있어야 한다.



개발자 일은 줄어들었나?

개발 생산성

비효율적인 작업이 줄어든 만큼(1/3) 맡은 일이 더 많아짐(x3)

개발자에 대한 수요는 유지하고 기업은 더 높은 생산성을 기대할 수 있음

개발자의 만족도 증가

- 만족도가 낮은 작업에 대한 작업 줄어듬: 테스트 코드 작성, PR 메시지, docstring 등
- 본질에 집중한 개발(기능 개발)
- AI 코드 리뷰를 통한 품질 개선

Tip 5. 아키텍처부터 설계

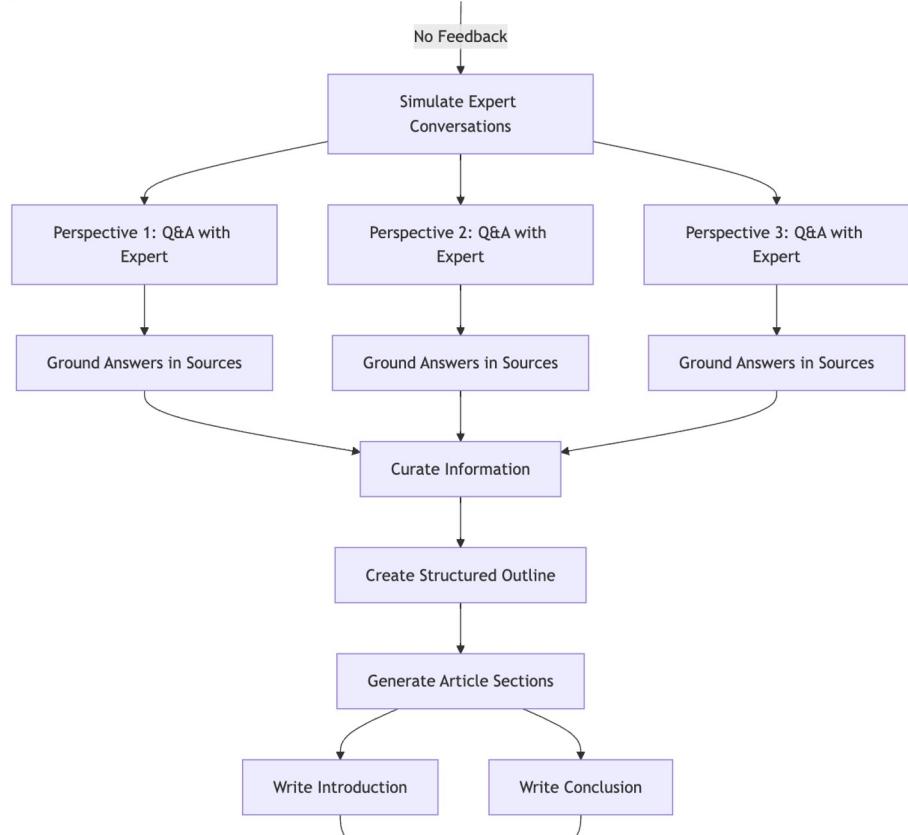
개발 아키텍처를 설계가 첫 단추

아키텍처 설계를 먼저 협의 하자.

```
src/storm_research/
├── __init__.py          # Package initialization
├── configuration.py     # Runtime configuration management
├── graph.py              # Main LangGraph graph definition and logic
├── prompts.py            # System prompts and templates
├── state.py               # State definitions (InputState, OutputState, etc.)
├── tools.py                # Tool implementations (search, etc.)
└── utils.py                 # Utility functions
```

개발 아키텍처를 설계가 첫 단추

mermaid diagram 도 좋다



Tip 6. 패키지 관리, 인프라 설정

패키지 관리

패키지 관리를 통해 협업을 위한 설정을 먼저 한다

```
## 4. Environment Setup with UV

The project uses `uv` for dependency management, configured via `pyproject.toml`:

### pyproject.toml Structure:
```toml
[project]
name = "storm-research-assistant"
version = "0.1.0"
requires-python = ">=3.11,<4.0"
dependencies = [
 "langgraph>=0.2.6",
 "langchain-openai>=0.1.22",
 "langchain-anthropic>=0.1.23",
 # ... other dependencies
]

[dependency-groups]
dev = [
 "langgraph-cli[inmem]>=0.1.71",
 "pytest>=8.3.5",
]
```
...
```



Tip 7. 완성형 프로젝트를 레퍼런스로!

LLM 은 참조를 잘한다

이미 잘 구조화된 프로젝트를 참고로 넣으면 해당 아키텍처를 잘 따라가는 편

다음의 프로젝트 구조를 참고해줘:

@/Users/teddy/Dev/github/Base-LangGraph-Project

Tip 8. Git 활용

Github Copilot 리뷰어

PR Reviewer 로 Copilot 을 지정해보자

Reviewers

| |
|---|
|  Copilot |
|  PangPangGod |

 **Pull Request Overview**

This PR introduces two new markdown templates: one for setting up a package with TDD practices (`setup-package.md`) and one for guiding pull request creation (`create-pr.md`).

- Adds instructions for initializing projects via the `uv` command in a TDD structure.
- Provides a comprehensive set of guidelines and prompts for automating pull request creation and management.

 **Reviewed Changes**

Copilot reviewed 2 out of 2 changed files in this pull request and generated 1 comment.

| File | Description |
|--|--|
| <code>commands/setup-package.md</code> | Outlines package setup commands with TDD guidance using <code>uv init</code> commands. |
| <code>commands/create-pr.md</code> | Details prompts and structured sections for creating effective pull requests. |

Github CI/CD

Claude Code / Cursor AI에서 다양한 Github Action 설정이 가능

```
yaml
name: Deploy to Vercel

on:
  push:
    branches: [ main ]

jobs:
  deploy:
    runs-on: ubuntu-latest

    steps:
      - uses: actions/checkout@v3

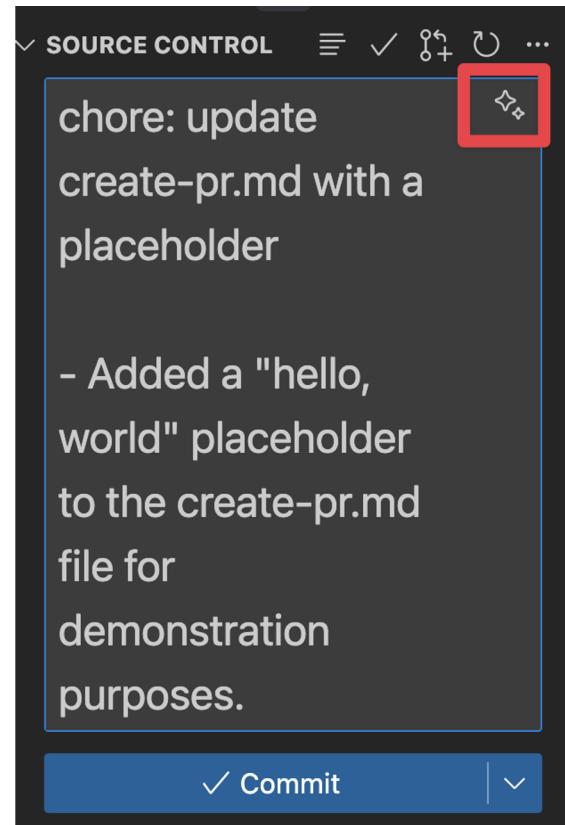
      - name: Setup Node.js
        uses: actions/setup-node@v3
        with:
          node-version: '18'
```

자잘한 Git 작업

Github Commit Message 생성

Github Conflict 해결

PR Review





[tip] Claude Code

Claude Code

CLI 기반 Vibe Coding 도구

- 설치: npm install -g @anthropic-ai/clause-code

배경

- 개발자들의 공통된 UI: “누구나 터미널은 사용하고 있음”

"Claude Code는 자체 코드의 80%를 작성했습니다."

참고자료

- <https://www.anthropic.com/engineering/clause-code-best-practices>
- <https://bagerbach.com/blog/how-i-use-clause-code>
- <https://github.com/tokenbender/agent-guides>

더 나은 결과를 얻기 위해

컨텍스트 큐레이션

- 작업 사양을 넘어선 기존 경험과 방향성 단서를 포함

아키텍처

- 시스템 컨텍스트에 잘 맞는 디자인을 만들 수 있도록 모델을 제공하고 안내하는 것은 우리의 임무
- 즉, 에이전트 도구가 실행할 수 있는 명시적인 계획을 수립하고 더 많이 관여

상세한 프롬프트(지시사항)

- 맥락에 대한 최대한 친절한 가이드를 제시

CLAUDE.md

CLAUDE.md 파일은 방대한 Context 를 파일 형태로 저장할 수 있는 “메모리” 기능

이는 번거로운 일처럼 느껴질 수 있지만 매우 유용하며 엄청난 시간을 절약

수시로 [CLAUDE.md](#) 파일을 업데이트 하는 것을 권장

frondend, backend, database 처럼 sub-folder 로 구분되어 있는 경우 각각의 하위 폴더
마다 [CLAUDE.md](#) 파일을 생성하는 것을 권장

장기기억 기능

을 눌러 기억하고 싶은 내용을 메모

CLAUDE.md에 추가하여 지침을 기억

```
# Always use descriptive variable names
```

```
# to memorize
```

모드 설정

Plan mode(shift+tab): 코딩을 시키기 전 사전의 계획을 세울때 활용

자세한 모드(CTRL+R): 자세한 로그 확인

Bash 모드(! 접두사)를 사용하여 명령을 실행: 터미널 명령어 실행

다 귀찮아 (bypass-permission)

claude --dangerously-skip-permissions

WARNING: Claude Code running in Bypass Permissions mode

In Bypass Permissions mode, Claude Code will not ask for your approval before running potentially dangerous commands.

This mode should only be used in a sandboxed container/VM that has restricted internet access and can easily be restored if damaged.

By proceeding, you accept all responsibility for actions taken while running in Bypass Permissions mode.

<https://docs.anthropic.com/s/clause-code-security>

- › 1. No, exit
- 2. Yes, I accept

동시작업

프론트엔드 + 백엔드를 동시에 사용하는 것은 훌륭한 접근

git worktree 를 사용하여 여러 에이전트와 동일한 코드베이스에서 작업

[참고] git worktree

하나의 Git 저장소에서 여러 개의 작업 디렉토리를 동시에 사용할 수 있게 해주는 기능

서브 에이전트 활용

클로드 코드에게 요청: “Use subagents”

여러 하위 에이전트를 사용하여 여러 각도에서 동시에 문제에 접근한 다음, 메인 에이전트가 메모를 비교하여 최적의 해결책을 찾도록 하는 것

UI - 이미지, 스크린샷 활용

스크린샷을 사용하세요(드래그 앤 드랍으로 OK!)

UI 문제를 디버깅하거나, 디자인을 COPY 할 때 유용

추론 프롬프트

think, think harder, ultrathink

시간은 오래 걸리지만 더 많은 생각으로 인하여 일반적으로 더 좋은 output

ultrathink: 31,999개의 토큰을 할당하는 것으로 추정

모든 것을 문서화

클로드 코드에게 생각, 현재 작업 사양, 디자인, 요구 사항 사양 등을 중간 마크다운 문서에 작성하도록 지시

- 나중에 컨텍스트 역할을 하기도 하고 스크래치 패드 역할을 하기도 함
- 세션의 길이가 길어지면 컨텍스트가 손실
- 문서를 다시 읽는 것만으로도 중요한 맥락을 되찾을 수 있음

MCP servers

context7: 실시간 문서 주입을 통해 최신 라이브러리 문서를 AI 프롬프트에 자동으로 통합

deepwiki: GitHub 저장소 문서를 Markdown으로 변환하여 제공

puppeteer: 웹 자동화 도구로, 헤드리스 브라우저 제어 기능을 제공

langchain-dev-docs: LangChain/LangGraph 최신 문서 제공

사용량 체크

npx ccusage@latest

| | | | | | | | |
|---------------|----------|--------|---------|-------------|------------|-------------|----------|
| 2025
06-22 | - opus-4 | 1,872 | 28,863 | 1,867,587 | 31,526,... | 33,424,7... | \$84.50 |
| 2025
06-23 | - opus-4 | 1,223 | 13,765 | 2,390,803 | 35,955,... | 38,361,5... | \$99.81 |
| 2025
06-25 | - opus-4 | 438 | 9,396 | 565,652 | 5,965,5... | 6,541,030 | \$20.27 |
| 2025
06-26 | - opus-4 | 61 | 383 | 16,691 | 149,290 | 166,425 | \$0.57 |
| 2025
06-28 | - opus-4 | 41 | 70 | 25,721 | 97,172 | 123,004 | \$0.63 |
| 2025
06-29 | - opus-4 | 148 | 1,218 | 252,917 | 3,201,5... | 3,455,856 | \$9.64 |
| Total | | 23,313 | 721,451 | 17,058,7... | 257,786... | 275,590,... | \$759.49 |

세션 컨텍스트 길이

이상적으로는 한 세션에서 처리할 수 있을 만큼 작은 작업을 세분화하는 것이 좋음

/clear 을 실제로 사용하지 않고 다른 세션을 시작하는 것을 권장

여러 모델을 함께 사용

믿을 수 없을 정도로 효과가 좋았던 것이 바로 “o3-pro” 와 “Claude Code” 를 동시에 실행

예시

1. o3-pro 로 PRD 문서를 작성 (혹은 PRD 에 대한 리뷰 요청)
2. 코드 작업은 Claude Opus 4 를 사용

Awesome Claude Code

<https://github.com/hesreallyhim/awesome-claude-code>

Slash-Commands

Version Control & Git

[/2-commit-fast](#) by [steadycursor](#)

Automates git commit process by selecting the first suggested message, generating structured commits with consistent formatting while skipping manual confirmation and removing Claude co-Contributorship footer

[/analyze-issue](#) by [jerseycheese](#)

Fetches GitHub issue details to create comprehensive implementation specifications, analyzing requirements and planning structured approach with clear implementation steps.

[/bug-fix](#) by [danielscholl](#)

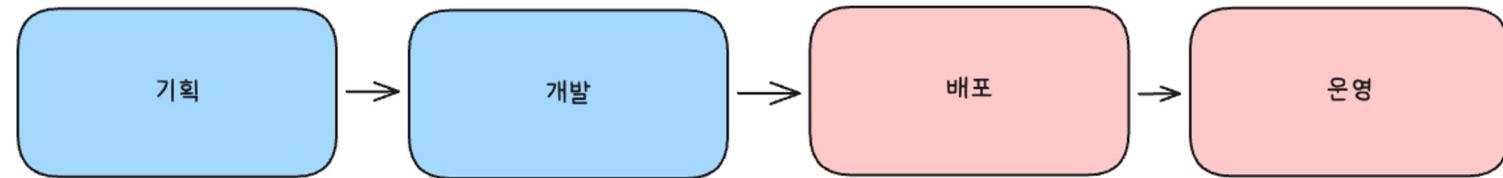
Streamlines bug fixing by creating a GitHub issue first, then a feature branch for implementing and thoroughly testing the solution before merging.



바이브 코딩이 할 수 없는 영역

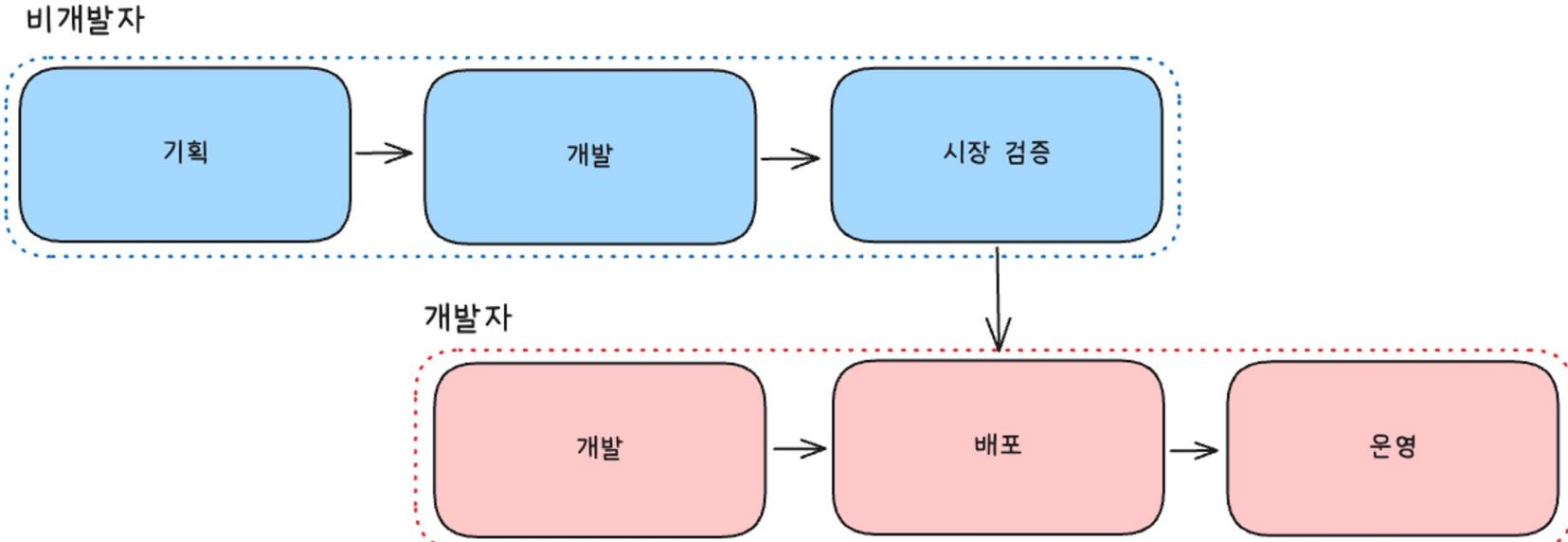
개발 프로세스

DevOps에 대한 영역은 여전히 숙제로 남아 있음



개발 프로세스

심지어, 비개발자 바이브 코더도 결국 배포 단계에는 개발자의 도움이 필요할 수 있음



취업시장에서의 개발 경쟁력

취업 시장: DevOps 엔지니어 수요 증가

연봉: 일반 개발자보다 평균 20-30% 높음

커리어: 풀스택을 넘어선 풀사이클 개발자

- 예전 개발자: "코드만 잘 짜면 돼"
- 바이브 코더: "코드 + 배포 + 모니터링 + 운영까지!"

개발자의 핵심 역량

인재 채용

- (기존) 알고리즘 기반의 코딩의 능력을 평가하는 방식
- (변경) 창의적인 접근에 대한 평가 방식 필요

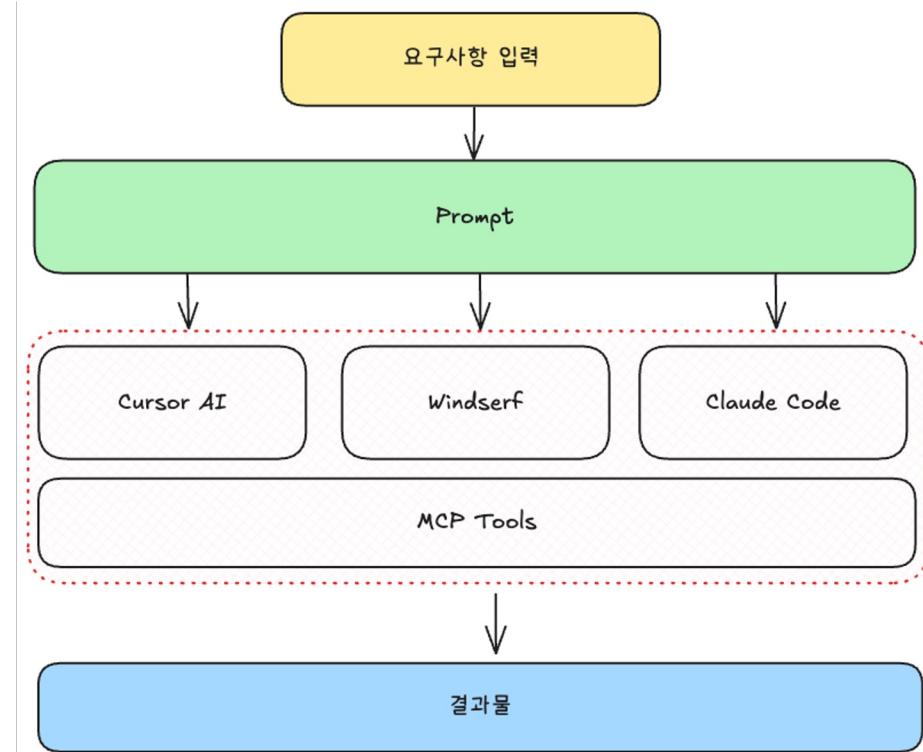
앞으로 요구되는 개발자의 핵심 역량

- 구조를 설계하고 이를 AI에게 지시할 수 있는 능력
- AI가 작성한 코드를 설명할 수 있는 능력
- 코드의 잠재적인 취약점을 찾아내고 이를 말할 수 있는 능력

개발 방식의 변화

바이브 코딩으로 서비스를 만들 때

- 요구사항 입력 (사람)
- Prompt (사람)
- 도구선택 (사람)
- MCP Tools (사람)

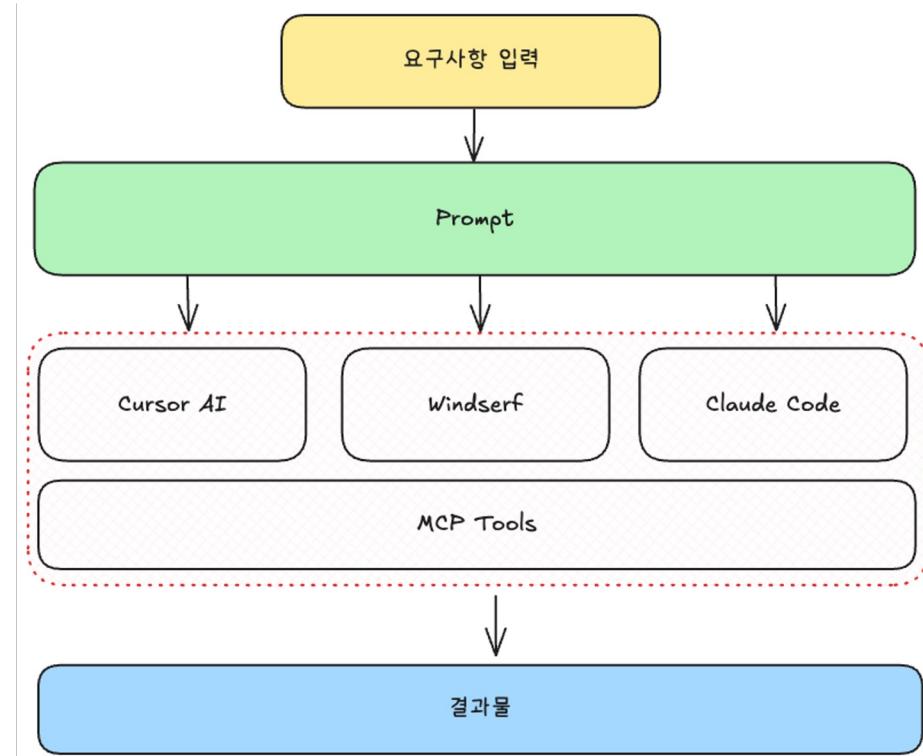


바이브 코딩으로 서비스를 만들 때

문제점

사람의 역량에 따라 결과물의 차이 발생

- 요구사항 입력 (사람)
- Prompt (사람)
- 도구선택 (사람)
- MCP Tools (사람)



바이브 코딩을 위한 사람의 역량이 중요

“다양한 AI 교육을 통해 Tool 사용법 학습을 통해 양질의 결과물을 만드려는 노력”

하지만, 빠른 AI 발전속도를 따라가기 위해서

“AI 활용 교육”이 많이 강조하는 분위기가 형성

하지만 저희는 좀 다르게 생각해 봤습니다.

AI 는 사람의 발전속도를 크게 뛰어 넘을 것으로 예상

AI 의 코딩 능력이 사람의 능력을 Golden Cross 하는 시기도 얼마 안남았다는 생각

“반자율 주행” 이 언젠가는 발전하여 “완전 자율 주행” 으로 바뀌게 되는 날을 기대하는 것처럼...

조직의 관점에서의 변화

바이브 코딩을 위한 시스템 마련

따라서, 개발자가 AI 를 활용한 코딩 생산성을 높이는 방법보다

완전 자율(Autonomous) AI Coding System 마련을 위한 준비를 시작

완전 자율 코딩 시스템

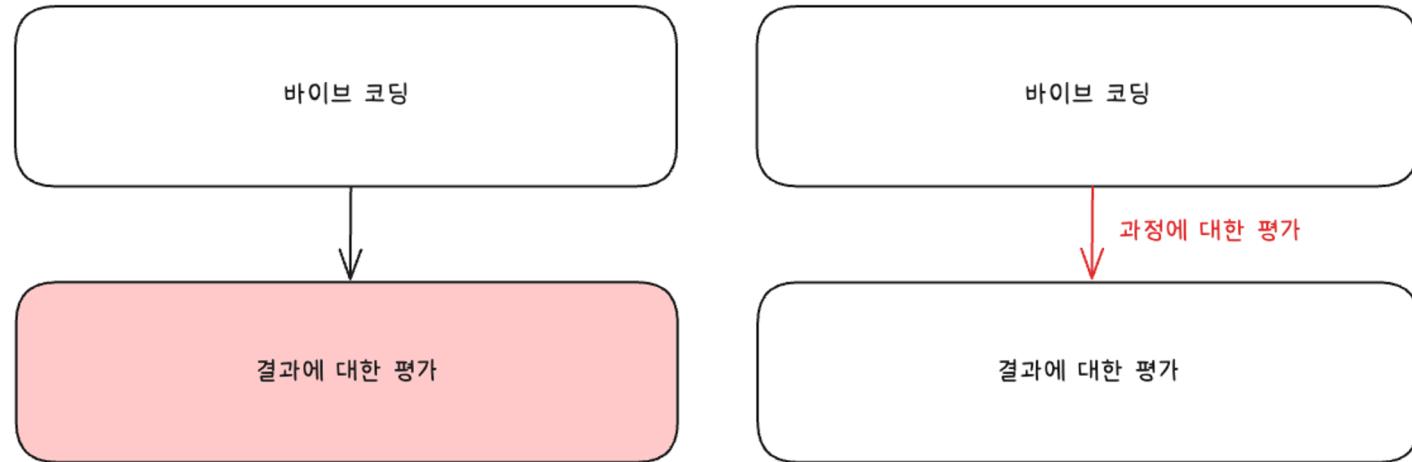
결과물 보다는 과정에 대한 세밀한 평가가 중요
(예시)

- A 프롬프트: 결과는 좋음, 더 높은 비용, 10회 Human 개입 수정
- B 프롬프트: 결과는 A 대비 85% 수준, 더 낮은 비용, 3회 Human 개입 수정

A, B 과정의 차이에 대해서 분석하고, 더 적은 수정이 있다면 이에 대한 분석 및 반영이 중요

바이브 코딩 해커톤 사례

결과 중심의 평가에서 과정 중심의 평가로 전환이 필요



바이브 코딩을 위한 MCP 개발

Agent 와 연동되는 MCP 의 연동이 매우 중요한 역할

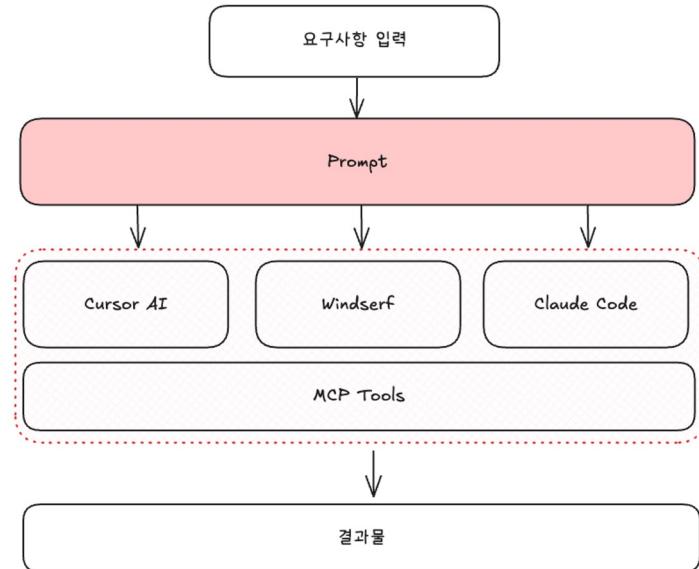
따라서, 바이브 코딩을 잘할 수 있는 MCP 도구 개발

- LangChain Dev Docs: 최신 개발 문서 조회 MCP
- LangConnect: Vector DB 조회/추가 MCP
- PRD MCP: 프로젝트 특화 PRD 작성 MCP

무엇보다 중요해진 PRD

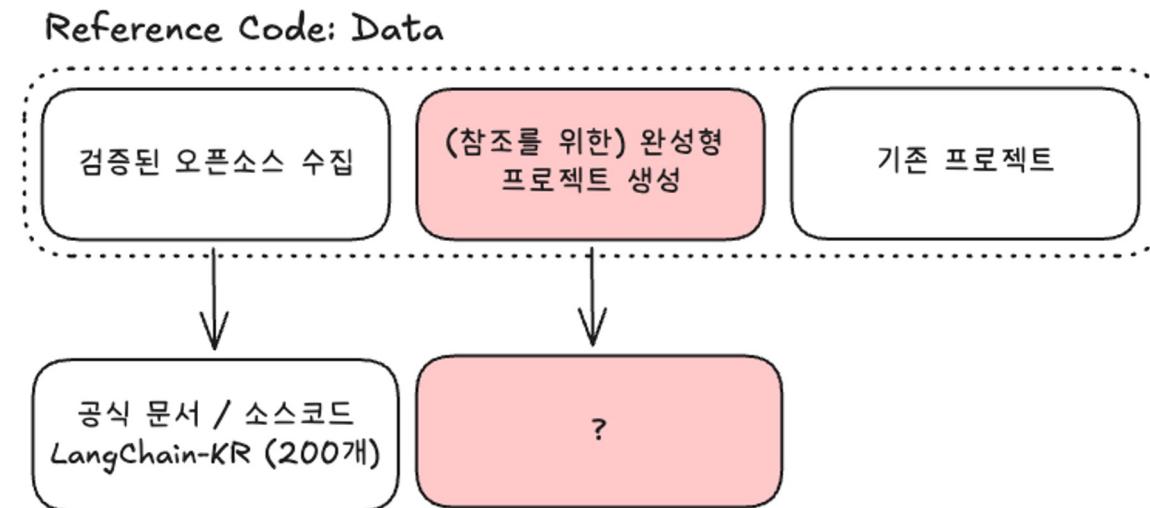
제품 요구사항 정의서 (PRD)

1. 서비스의 목표
2. 구현해야하는 기능 상세 명세
3. 기능에 대한 TODO 의 생성
4. TODO 를 순차적으로 수행해 나가는 우선순위 정의
5. 참고할 수 있는 Reference Code

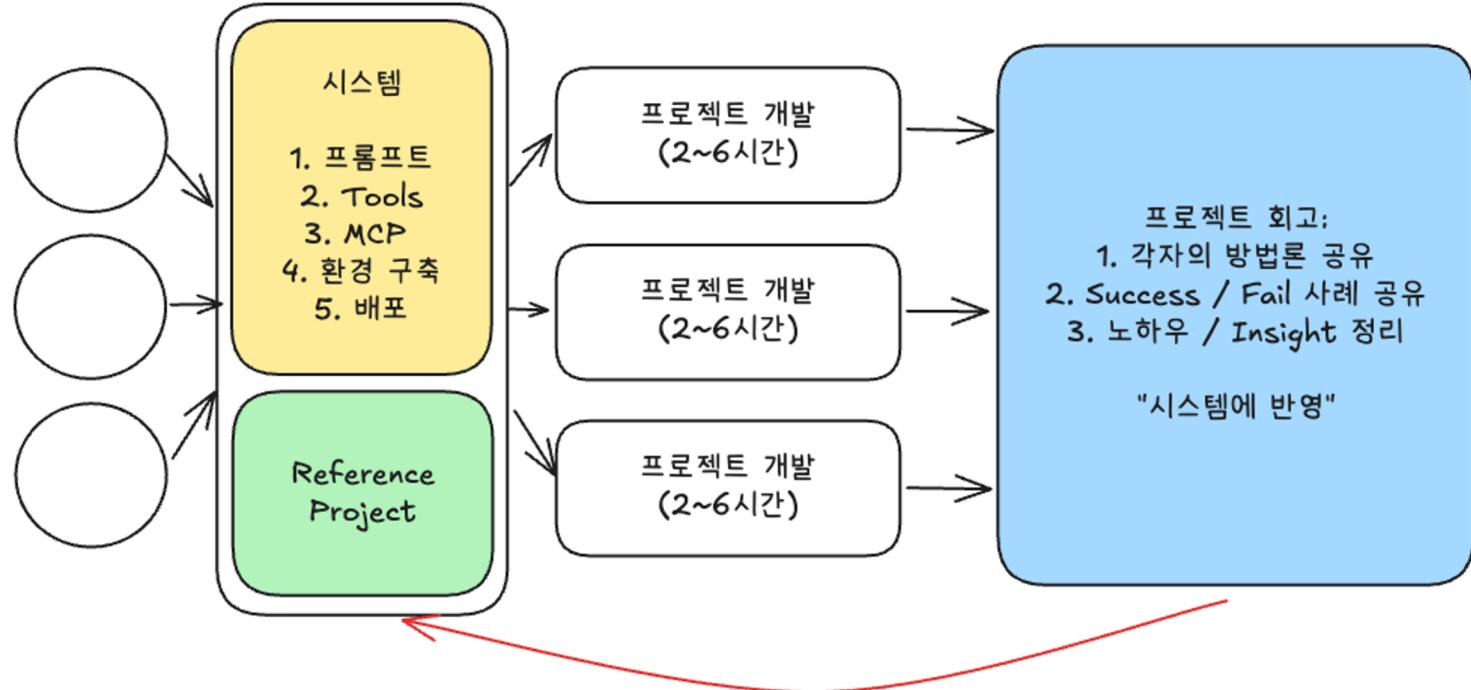


조직을 위한 바이브 코딩 시스템 마련

성공적인 바이브 코딩을 위해서는 참조 소스코드가 중요한 SEED



조직을 위한 바이브 코딩 시스템 마련



조직을 위한 바이브 코딩 시스템 마련

(강제사항) 월 5~6 개의 프로젝트 개발 (주 2~3시간 내외)

<https://github.com/orgs/teddynote-lab/repositories>

STORM Research Assistant

License MIT python 3.11+ LangGraph 0.2.6+ code style black

STORM (Synthesis of Topic Outlines through Retrieval and Multi-perspective Question Asking) - A writing system for generating grounded and organized long-form articles from scratch, with comparable breadth and depth to Wikipedia pages

Overview

STORM Research Assistant is a LangGraph-based implementation of the STORM methodology from Stanford, designed to write grounded and organized long-form articles from scratch. The system models the pre-writing stage by (1) discovering diverse perspectives for researching the given topic, (2) simulating conversations where writers with different perspectives pose questions to a topic expert grounded on trusted Internet sources, and (3) curating the collected information to create an outline before generating the final article.

YouTube Insight Extractor

Every viewer sees differently. Every perspective matters.

Transform how you extract insights from YouTube videos with context-aware AI analysis

The Core Idea

Same video. Different eyes. Infinite insights.

When a Product Manager watches a tech demo, they see market opportunities and user needs. When a UX Designer watches the same video, they notice interaction patterns and usability issues. When a Data Scientist views it, they focus on methodology and analytical approaches.

Why should AI analysis be any different?

Traditional tools give you one-size-fits-all summaries. We believe your professional context, experience level, industry background, and specific goals should shape every insight you receive.

사내 프로세스 개선툴

사내 프로세스 개선률에 바이브 코딩을 적용하면 좋은 이유

1. Production 환경에 대한 부담이 없음 (사내 이용)
2. 1인 개발에 적합
3. 사내에 공통으로 필요한 모듈 개발시 팀 전체에 크게 도움
4. 반복되거나 불필요한 업무를 획기적으로 개선

대기업 / 실리콘 밸리에서도 사내 Side Project 에 바이브코딩을 적극 적용

Claude Code 도 사내 프로젝트에서 시작하여 Product 가 된 대표적인 사례

사내 Side Project

LangConnect Client: Vector DB 모니터링 + MCP 연동 툴 (@jikime front 기여)

Main

LangConnect 클라이언트

LangConnect에 오신 것을 환영합니다.
LangChain과 PostgreSQL로 구동되는 강력한 문서 관리 및 검색 시스템입니다.

주요 기능

이 애플리케이션은 고급 검색 기능을 갖춘 문서 관리를 위한 포괄적인 인터페이스를 제공합니다:

- 컬렉션 관리**
 - 문서 컬렉션 생성 및 관리
 - 컬렉션 통계 보기
 - 컬렉션 일괄 삭제[컬렉션으로 이동](#)
- 문서 관리**
 - 여러 문서 업로드 (PDF, TXT, MD, DOCX)
 - 문서 첨부 보기 및 관리
 - 개별 첨부 또는 전체 문서 삭제[문서로 이동](#)
- 검색**
 - 시맨틱 검색: AI 기반 유사도 검색
 - 키워드 검색: 전통적인 전문 검색
 - 하이브리드 검색: 두 가지 접근법의 장점 결합
 - 고급 메타데이터 필터링[검색으로 이동](#)
- API 테스터**
 - 모든 API 엔드포인트 직접 테스트
 - API 기능 탐색
 - 통합 개발 및 디버깅[API 테스터로 이동](#)

Google의 Agent 백서에서 말하는 agent 정의에 대해서 조사해줘

Google의 Agent 백서에서 말하는 agent 정의에 대해 조사해드려겠습니다.

```
search_documents
list_collections
search_documents
search_documents
search_documents
```

요청

```
{
  'collection_id': '3f3190a3-bdaf-4105-b9a2-e12ba26164e1',
  'limit': 5,
  'query': 'cognitive architecture agent components',
  'search_type': 'hybrid'
}
```

답변

Google의 Agent 백서에서 정의하는 agent에 대해 조사한 결과를 말씀드리겠습니다.

Google의 Agent 백서에 따르면, **Agent**는 인지 아키텍처(cognitive architecture)를 통해 행동, 학습, 의사결정을 수행하는 시스템으로 정의됩니다.

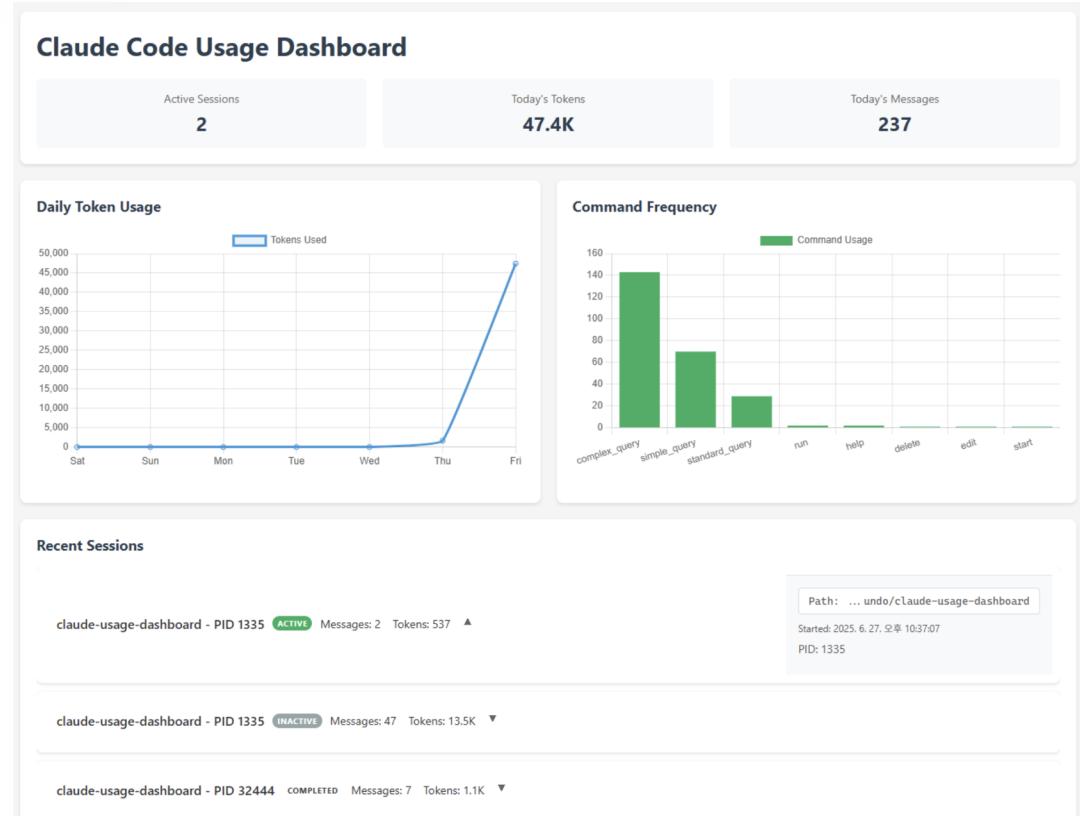
Agent의 핵심 구성 요소

Agent의 인지 아키텍처는 세 가지 필수 구성 요소로 이루어져 있습니다:

사내 Side Project

Claude Code 사용량 대시보드

- 사용량 추적 및 공유



YouTube 인사이트 추출기

해외 YouTube 기반 인사이트 추출

Persona 기반 분석

- 5년차 마케팅 전문가
- 1년차 AI Engineer

🎯 YouTube Insight Extractor

Every viewer sees differently. Every perspective matters.

Transform how you extract insights from YouTube videos with context-aware AI analysis

LangGraph Studio / Default (youtube_insight_extractor) / Graph / Chat

AI

YouTube Insight Analysis Complete

📊 Summary

Claude Code is a terminal-based, agentic coding tool that integrates seamlessly into developer workflows, enabling natural language-driven code navigation, editing, and automation (including GitHub Actions), with persistent, context-aware memory (via quad.md), rapid setup, and evolving best practices that shift developers toward conversational, review-focused coding.

🔍 Detailed Insights

🌐 Customized Insights for AI developer

🌐 Key Insights

- Claude Code is an agentic coding tool designed to work directly in the terminal, making it highly adaptable to any developer workflow, including those using VS Code, iTerm, or SSH sessions.
- Installation and setup are straightforward: requires NodeJS and a simple npm install command. Once installed, running 'claude' in the terminal launches the agent.
- Claude Code acts as a true agent, not just an autocomplete tool. It can read, edit, and navigate codebases, execute bash commands, and perform multi-step tasks based on natural language instructions.
- No special indexing or setup is required for large codebases; Claude Code works out-of-the-box with most languages and project structures.
- Integration with GitHub Actions is supported, allowing background automation of tasks like PR reviews and test writing. This can be set up with a few guided steps from the terminal.
- The Claude.md (quad.md) file system allows for persistent, context-aware instructions and preferences, both at the project and user level. This enables tailored agent behavior and shared team knowledge.
- Best practices include asking Claude to plan before coding, using extended thinking steps, and leveraging the memory system for recurring instructions.
- The workflow is evolving towards 'vibe coding'—a more conversational, agent-orchestrated approach where developers focus on reviewing and guiding code rather than writing every line themselves.
- Community feedback highlights rapid productivity gains, especially for large teams and codebases, but also notes a learning curve in learning how to interact effectively with the agent.
- Future directions include deeper integration with more IDEs, CI systems, and chat-based workflows, reducing the need to open terminals for simple tasks.

Enter message

Show next card

Scroll to bottom

API 기반 Project 스트레스 테스트

대규모 사용자 스트레스 테스트

답변 지연 시간 테스트 / 검증

에러율 정량지표 산출

Awesome Python API Stress Test

A powerful and flexible API stress testing tool built with Python, featuring async support, real-time statistics, comprehensive configuration options, and a LangGraph-based interactive mode for modular test execution.

Features

- **Interactive Mode:** Menu-driven interface powered by LangGraph for easy test selection
- **Modular Architecture:** Extensible design allowing easy addition of new test modules
- **Async Performance:** Built on `aiohttp` for high-performance concurrent requests
- **Real-time Statistics:** Live dashboard showing request metrics, response times, and success rates
- **Flexible Configuration:** Support for various HTTP methods, headers, and request bodies
- **JSON Config Files:** Load test configurations from JSON files for reproducibility
- **Rich Terminal UI:** Beautiful terminal output with progress tracking and formatted tables
- **Comprehensive Metrics:** Track response times, status codes, error rates, and throughput

Prompt Optimization

DEEVO 논문

- Debate 기반 Prompt 최적화
- 멀티 에이전트 협업 프롬프트

DEEVO - Prompt Optimizer

DEEVO (Debate-Driven Evolutionary Optimization) - AI 토론과 진화 알고리즘을 통한 프롬프트 최적화 시스템

🚀 v0.2.0 주요 업데이트

✨ 유연한 적응형 프롬프트 생성

- 하드코딩 제거: 키워드 기반 분기문 없이 모든 상황에 동적 적용
- LLM 기반 상황 분석: 도메인, 전문가 역할, 접근방식 자동 식별
- 실제 작업 프롬프트 생성: 프롬프트 엔지니어링 가이드가 아닌 실제 작업 수행 프롬프트

📋 핵심 기능

1. 유연한 상황 기반 프롬프트 생성

- 사용자 상황을 LLM이 분석하여 최적의 도메인과 전문가 역할 자동 결정
- 5가지 적응형 전략으로 다양한 관점의 프롬프트 생성
 - `domain_expert` : 도메인 전문가 관점
 - `systematic_approach` : 체계적 단계별 접근
 - `results_focused` : 결과 중심 측정 가능한 접근
 - `framework_based` : 검증된 프레임워크 기반
 - `adaptive_expert` : 유연한 맞춤형 접근

MCP Use Case

코드로 제작하는 MCP 사례(Open Source: star 178)

다양한 MCP UseCase 수록

Quick-start Auto MCP : All in one Claude Desktop and Cursor

[English](#) | [한국어](#)

Introduction

Quick-start Auto MCP is a tool that helps you easily and quickly register Anthropic's Model Context Protocol (MCP) in Claude Desktop and Cursor.

Key advantages:

- 1. Quick Setup:** Add MCP functionality to Claude Desktop and Cursor simply by running a tool and copying/pasting the generated JSON file.
- 2. Various Tools Provided:** We continuously update useful MCP tools. Stay up to date with your personalized toolkit by starring and following us. :)

LangGraph MCP Agent

실시간 도구 적용 MCP 에이전트
(Open source: star 553)



LangGraph Agents + MCP

Language English Language 한국어

GitHub langgraph-mcp-agents License MIT Python ≥3.12 Version 0.1.0

Made by [TeddyNote](#)

Add MCP Tool

Registered Tools List

- weather
- desktop-commander

Delete Delete

Apply Tool Configurations

System Information

MCP Tools Count: 21

Model: Claude 3.7 Sonnet

Reset Conversation

Agent with MCP Tools

Ask questions to the ReAct agent using MCP tools.

Read the file structure of the current folder and output it in ASCII tree format. (but exclude .venv folders)

I'll read the file structure of the current folder and output it in ASCII tree format, excluding any .venv folders. Let me do that for you.

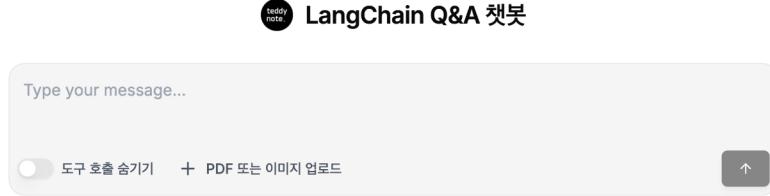
First, let's check what directories we're allowed to access: Now, let's list the current directory structure: Here's the file structure of the current folder in ASCII tree format, excluding the .venv folder:

```
.  
|-.env  
|-.env.example  
|-git  
| |-FETCH_HEAD  
| |-HEAD  
| |-config  
| |-description  
| |-hooks  
| | |-applypatch-msg.sample  
| | |-commit-msg.sample  
| | |-fsmonitor-watchman.sample  
| | |-post-update.sample
```

LangChain / LangGraph 코딩 에이전트

최신 LangChain 문서기반 코드 Agent

[링크 바로가기](#)



```
python
from langchain_teddynote import logging
logging.langsmith("Self-RAG-LangGraph")
```

2. Retriever(문서 검색기) 준비

```
python
from langchain_text_splitters import RecursiveCharacterTextSplitter
from langchain_community.document_loaders import WebBaseLoader
from langchain_community.vectorstores import Chroma
from langchain_openai import OpenAIEmbeddings

urls = [
    "https://lilianweng.github.io/posts/2023-03-15-prompt-engineering/",
    "https://lilianweng.github.io/posts/2023-03-15-agent/",

    "https://lilianweng.github.io/posts/2023-03-15-prompt-engineering/",
    "https://lilianweng.github.io/posts/2023-03-15-agent/"]
```

Type your message...

도구 호출 습기기 + PDF 또는 이미지 업로드

취소

오픈소스 프로젝트

해외 YouTube 영상 > 팟캐스트 변환

팟캐스트 목록

🎙 AI 팟캐스트 생성기

LangGraph와 다양한 LLM 모델(Azure OpenAI, OpenAI, Anthropic), OpenAI TTS를 활용한 자동 팟캐스트 제작 도구입니다. 주제를 입력하면 AI가 자동으로 리서치를 수행하고, 스크립트를 작성하며, 음성으로 변환해주는 통합 시스템입니다.

🚀 주요 기능

- 다양한 LLM 지원: Azure OpenAI, OpenAI, Anthropic 모델 선택 가능
- 자동 리서치: arXiv, PubMed, Wikipedia에서 관련 정보 자동 수집
- 스마트 스크립트 생성: AI 모델을 활용한 자연스러운 대화형 스크립트 생성
- 반복적 개선: 자동 비평 및 수정을 통한 스크립트 품질 향상
- 음성 합성: OpenAI TTS를 활용한 고품질 음성 생성 (gpt-4o-mini-tts 지원)
- 클라우드 업로드: 생성된 오디오 파일을 GoFile에 자동 업로드 (선택사항)
- 유연한 설정: LangGraph Studio를 통한 실시간 설정 조정

오픈소스 프로젝트

STORM Research



STORM Research Assistant

License MIT

python 3.11+

LangGraph 0.2.6+

code style black

STORM (Synthesis of Topic Outlines through Retrieval and Multi-perspective Question Asking) - A writing system for generating grounded and organized long-form articles from scratch, with comparable breadth and depth to Wikipedia pages

Overview

STORM Research Assistant is a LangGraph-based implementation of the STORM methodology from Stanford, designed to write grounded and organized long-form articles from scratch. The system models the pre-writing stage by (1) discovering diverse perspectives for researching the given topic, (2) simulating conversations where writers with different perspectives pose questions to a topic expert grounded on trusted Internet sources, and (3) curating the collected information to create an outline before generating the final article.

오픈소스 프로젝트

Upcoming Projects

1. RAG 검증용 데이터셋 Generator
2. 오프라인 / 온라인 데이터셋 Evaluator 파이프라인
3. 멀티 에이전트 협업 네트워크 기반 워크플로우

“글로벌 오픈소스 프로젝트”

조직을 위한 바이브 코딩 시스템 마련

결국 조직관점에서 추구하는 바는

AI 도구를 잘 활용하는 개인의 성장보다

“누구라도 수준 높은 프로젝트를 완성할 수 있는 시스템을 마련하는 데 집중”

“2025년 말에는 90% 의 코딩은 AI 가 해줄 것이다”

- Mike Krieger (Anthropic CPO, co-founder of Instagram)

Q & A

License

본 저작물의 저작권은
이경록(teddy@brain-crew.com)에게 있으며
무단으로 전재하거나 재배포하는 것을 금합니다.