

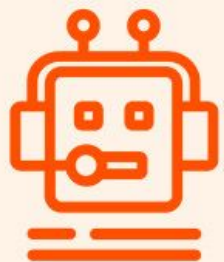
생성 AI로 smart 하게 일하는 방법

(feat. python)
신세계 DT AI/ML담당 정유선



What is AI?

What is AI? ANI vs. AGI vs. ASI



Artificial narrow intelligence (ANI)

Designed to perform specific tasks



Artificial general intelligence (AGI)

Can behave in a human-like way across all tasks



Artificial super intelligence (ASI)

Smarter than humans—the stuff of sci-fi

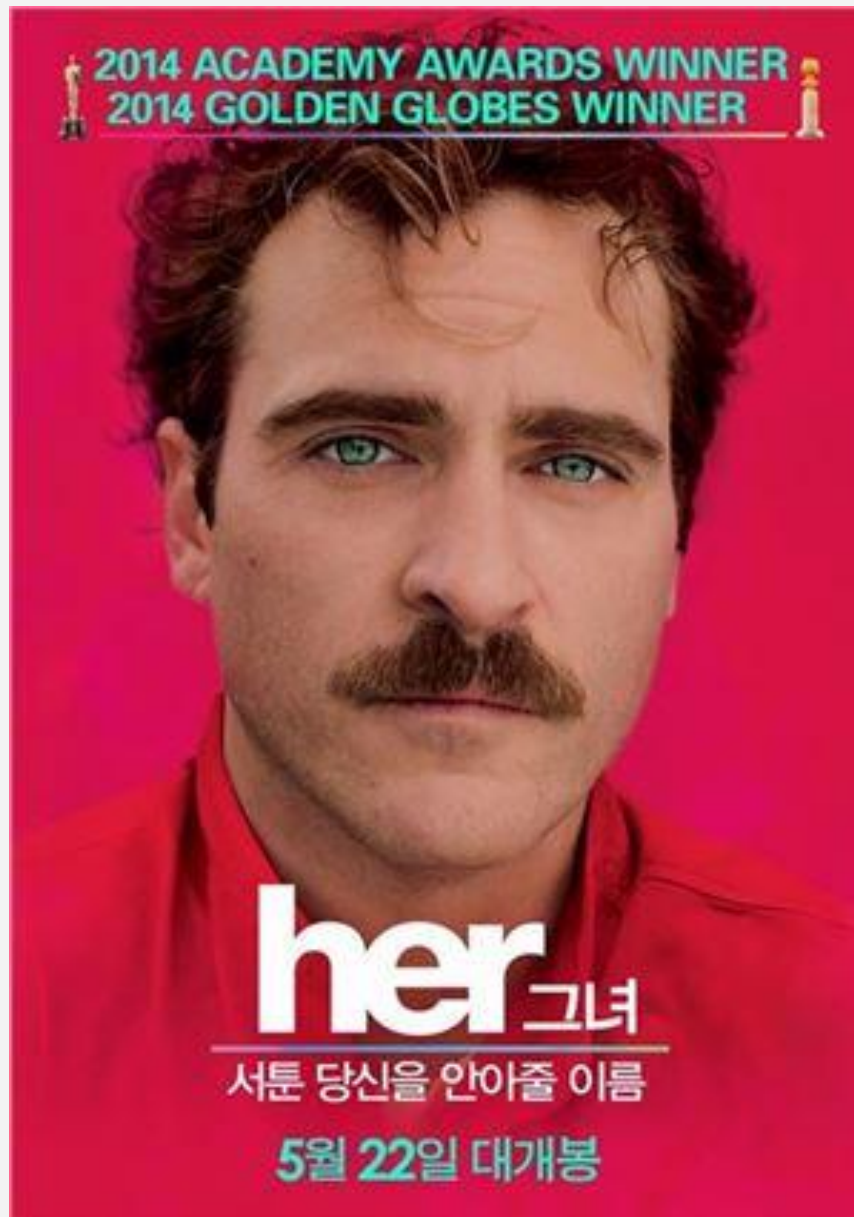
—zapier

AGI(Artificial General Intelligence) 범용 인공지능

인간 수준의 지능을 가진 인공지능을 말하며, 인간과 같이 추론, 이해, 학습, 감정, 특정 상황에 적응하는 능력 등이 있는 인공지능

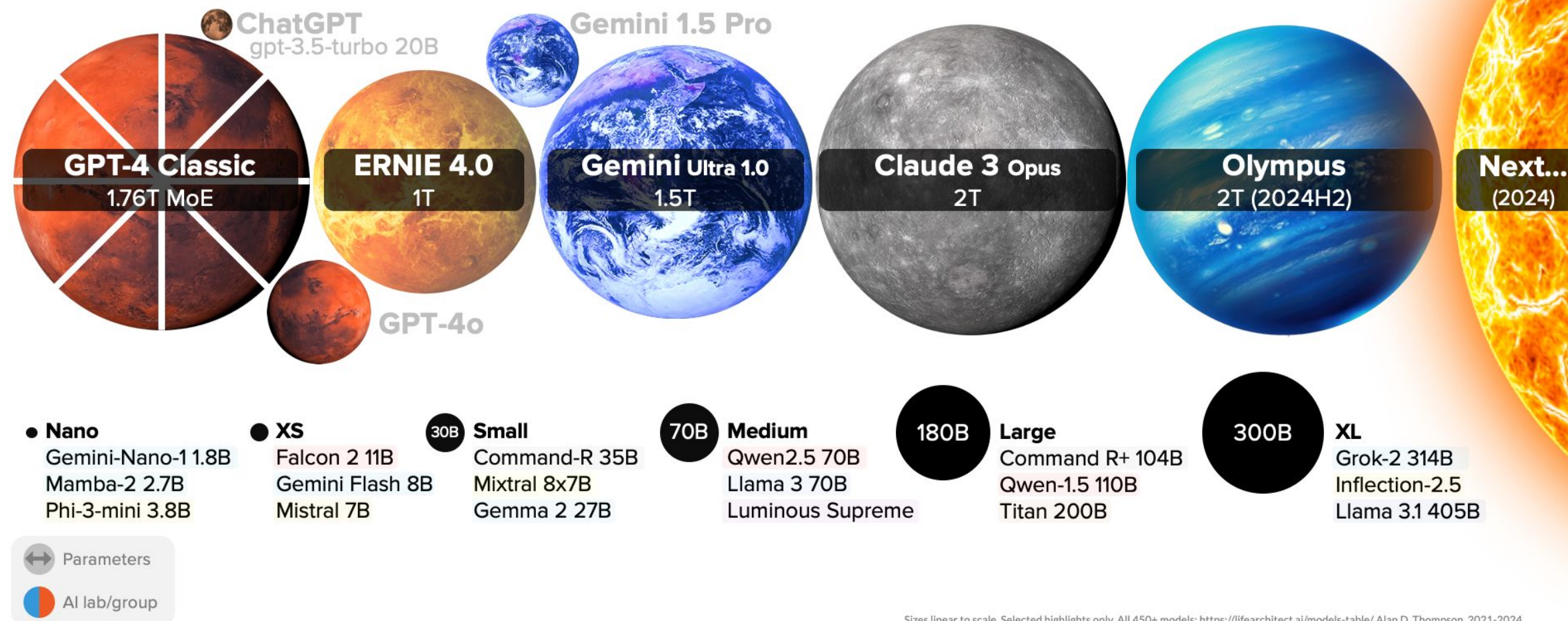
현존하는 인공지능은 특정화 된(specific task)에 특화된 Narrow AI이며, OpenAI의 GPT-4 수준이 AGI의 초기 형태로 언급, GPT-4는 일부 표준화된 테스트에서 인간보다 더 뛰어난 수행 능력을 보여줌

영화 속 AGI vs ASI



LLM 어디까지 커질래?

LARGE LANGUAGE MODEL HIGHLIGHTS (OCT/2024)



LifeArchitect.ai/models & 450+ more models at LifeArchitect.ai/models-table

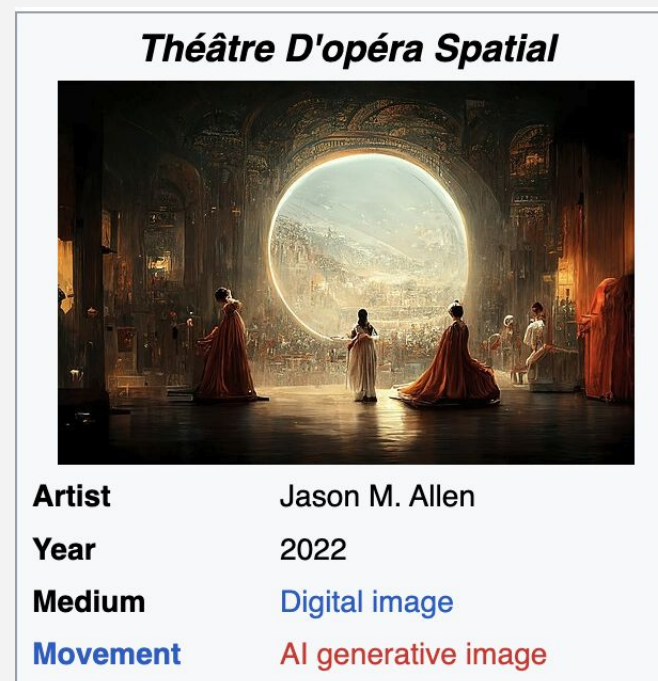
LLM (Large Language Model)

A large language model (LLM) is a type of machine learning model that can perform a variety of natural language processing (NLP) tasks, including generating and classifying text, answering questions in a conversational manner and translating text from one language to another.

The label “large” refers to the number of values (parameters) the model can change autonomously as it learns. Some of the most successful LLMs have hundreds of billions of parameters.

생성 AI란

생성형 인공지능(generative artificial intelligence) 또는 **생성형 AI(generative AI)**는 프롬프트에 대응하여 텍스트, 이미지, 기타 미디어를 생성할 수 있는 일종의 인공지능(AI) 시스템이다. 생성형 AI는 입력 트레이닝 데이터의 패턴과 구조를 학습한 다음 유사 특징이 있는 새로운 데이터를 만들어낸다. <https://ko.wikipedia.org/wiki>



AI가 그린 그림, 예술일까?

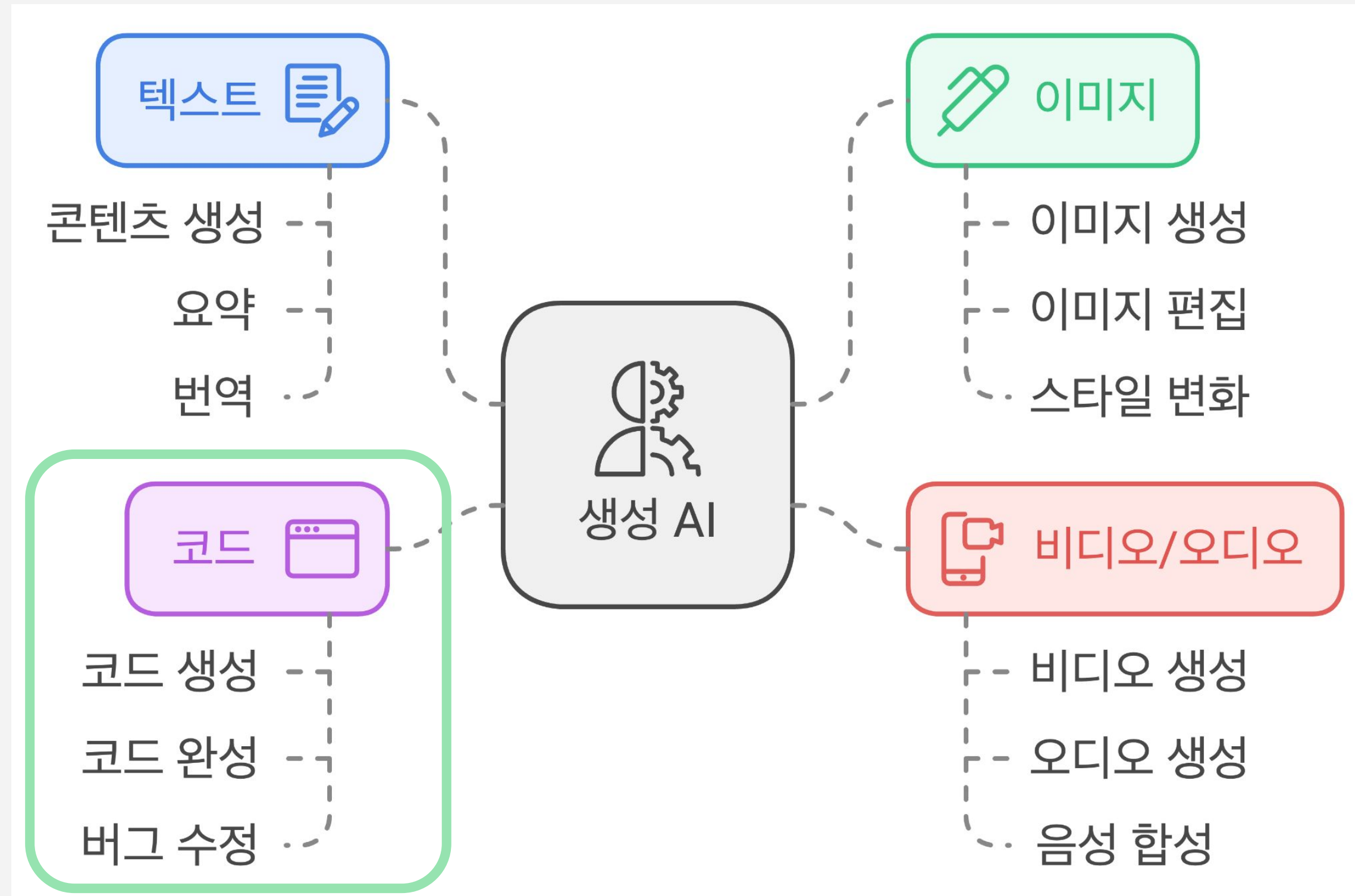
김재윤 기자 | 입력 2022.11.02 12:05 | 수정 2022.11.02 20:54 | 댓글 1



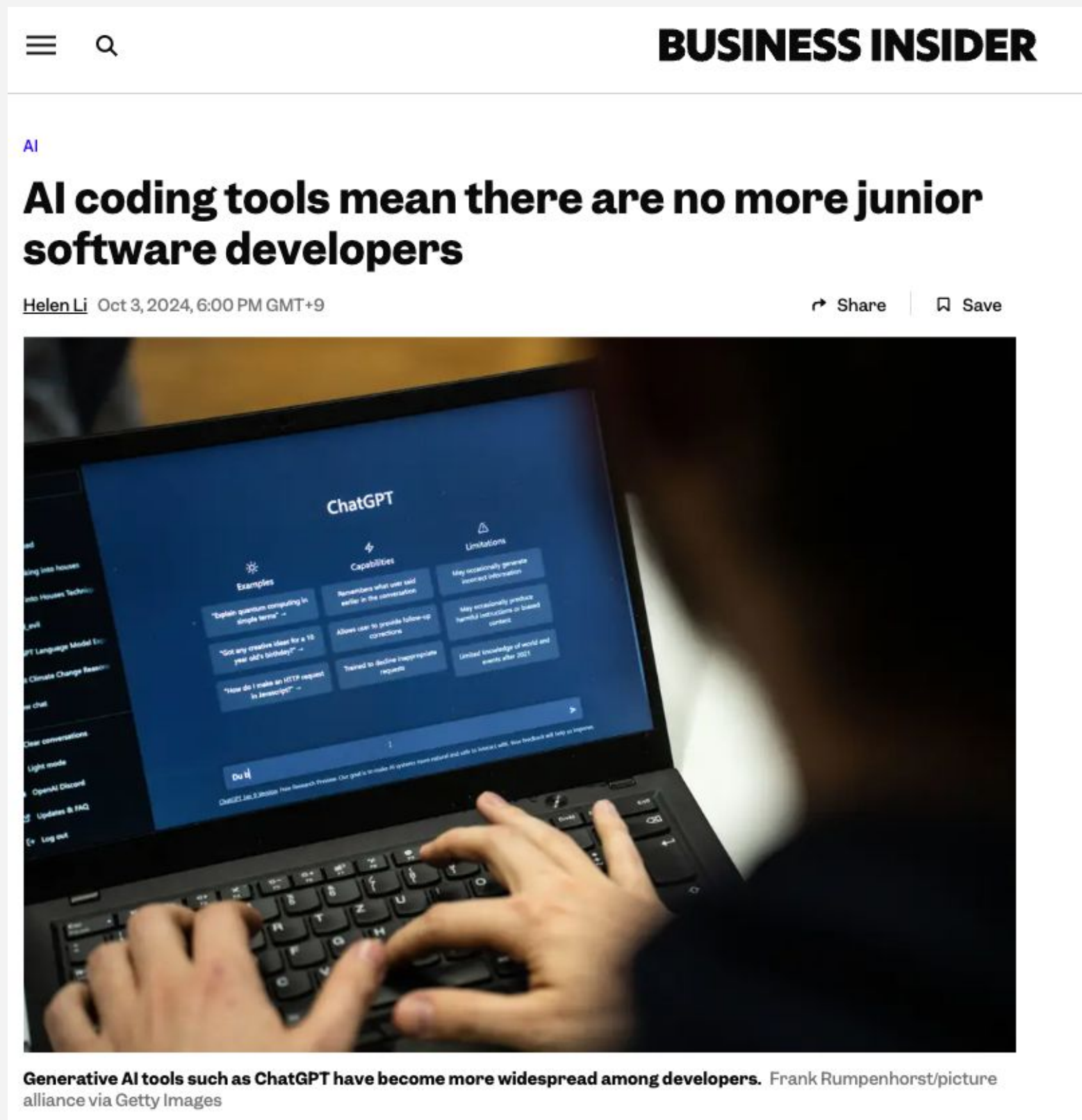
'AI 그림' 미술대회 수상 논란
-이성복(미술학)·조창오(철학) 교수
-"AI는 훌륭한 도구, 수상 문제 없어"
-"인간의 창의적 발상에 긍정적"

<https://channelpnu.pusan.ac.kr/news/articleView.html?idxno=31920>

생성 AI의 능력과 활용 가능 데이터



생성 AI와 주니어 개발자의 코드 생산성 향상 관련



8월에 발표된 GitHub 설문 조사에 따르면 미국, 브라질, 독일, 인도의 2,000명의 응답자 중 97% 이상이 직장에서 AI 코딩 도구를 사용했다고 답했습니다. Microsoft, Accenture, 익명의 Fortune 100 전자 제조 회사의 데이터를 분석한 연구에 따르면 **생성 AI 코드 제안 도구는 소프트웨어 개발자 생산성을 26%까지 높일 수도 있습니다.**

개발자들은 AI 코딩 지원을 도입하면 일자리를 잃애는 것이 아니라 소프트웨어 엔지니어링 분야가 발전하는 데 도움이 될 것이라고 말합니다. 이는 교사들이 도입에 처음 반대했음에도 불구하고 계산기가 수학 컴퓨팅을 가속화한 것과 같습니다.

<https://www.businessinsider.com/ai-coding-tools-junior-software-developers-microsoft-github-copilot-chatgpt-2024-10>

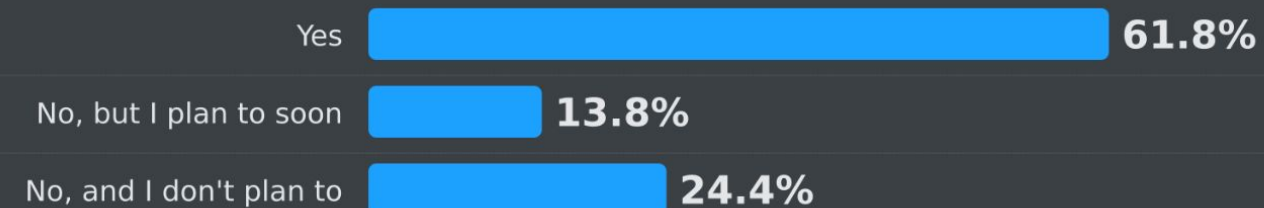
AI 툴의 활용

76% of all respondents are using or are planning to use AI tools in their development process this year, an increase from last year (70%). Many more developers are currently using AI tools this year, too (62% vs. 44%).

72% of all respondents are favorable or very favorable of AI tools for development. This is lower than last year's favorability of 77%; a decline in favorability could be due to disappointing results from usage.

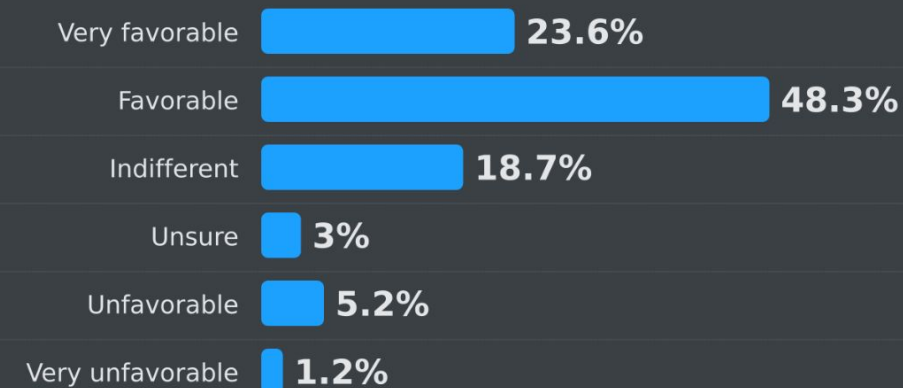
Sentiment and usage / All Respondents

AI tools in the development process

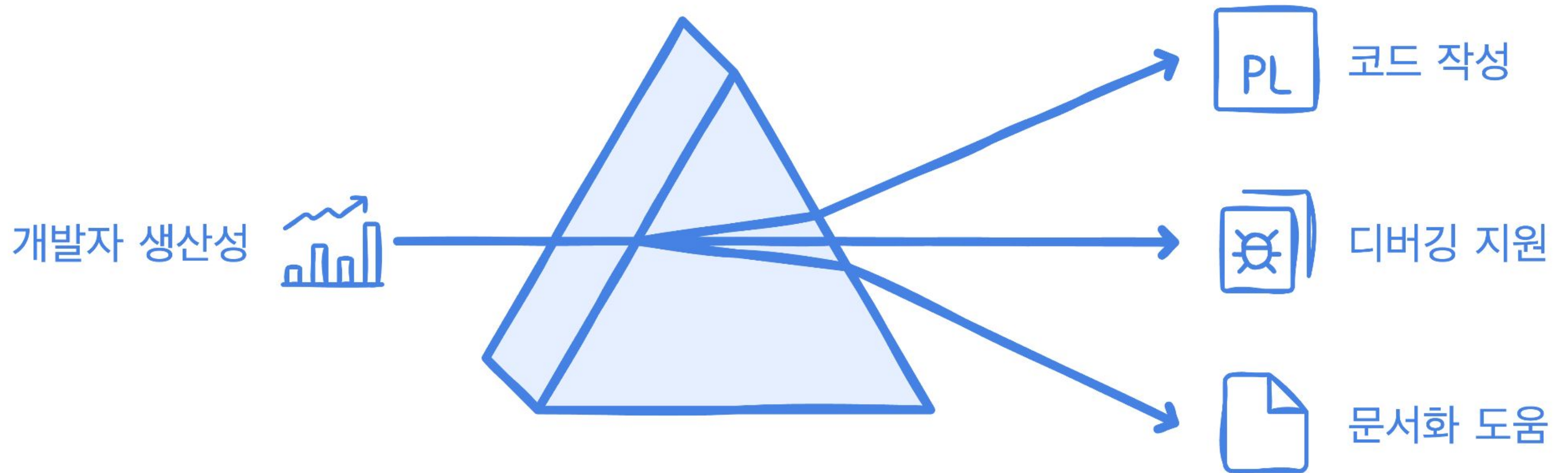


Sentiment and usage / All Respondents

AI tool sentiment

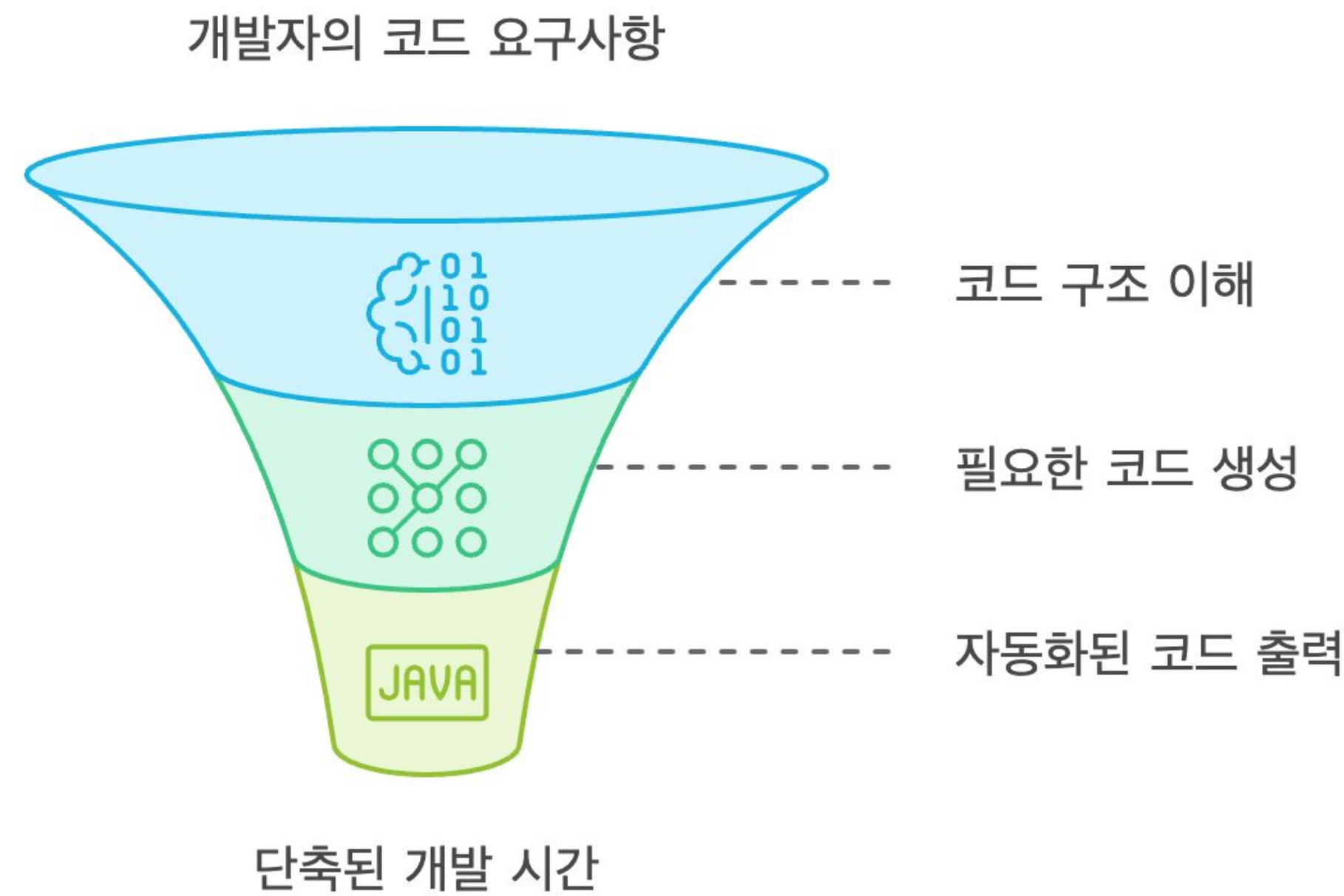


AI 코드 에디터의 역할



AI 코드 에디터의 활용

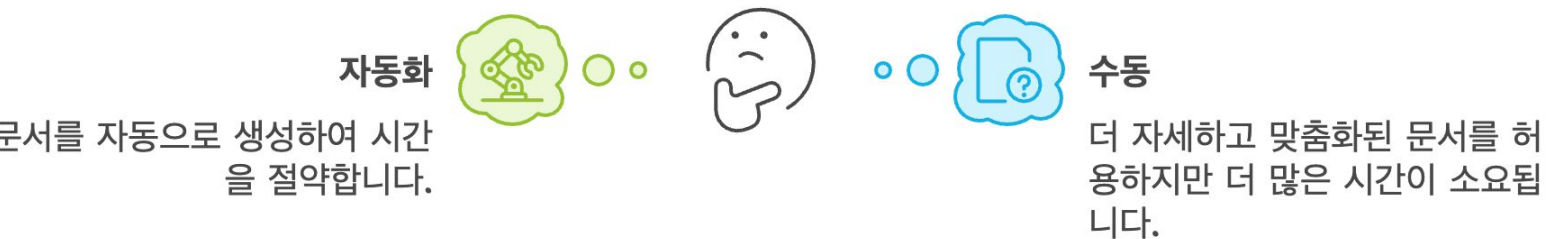
AI 기반 코드 생성 프로세스



AI-강화 디버깅 효율성



개발자는 문서화를 자동화해야 할까요?

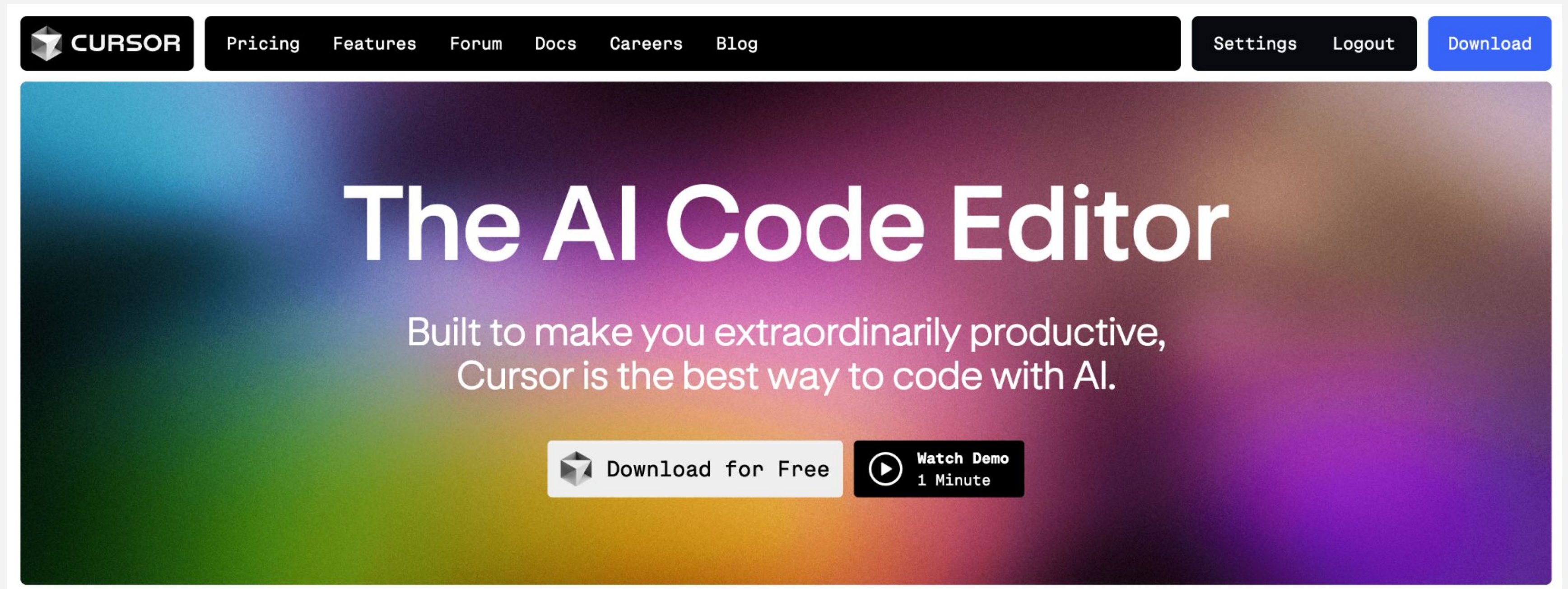


AI 강화 코드 품질 및 협업



생성 AI 코드 에디터

AI를 활용한 코드 에디터 (vscode기반) <https://www.cursor.com/>



생성 AI 코드 에디터 활용하기

AI 기반 코드 개발 및 향상

프로젝트 골격 생성



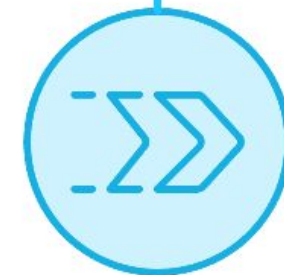
API 사양 작성



테스트 결과 생성



코드 리팩토링



기능 개발



테스트 코드 추가

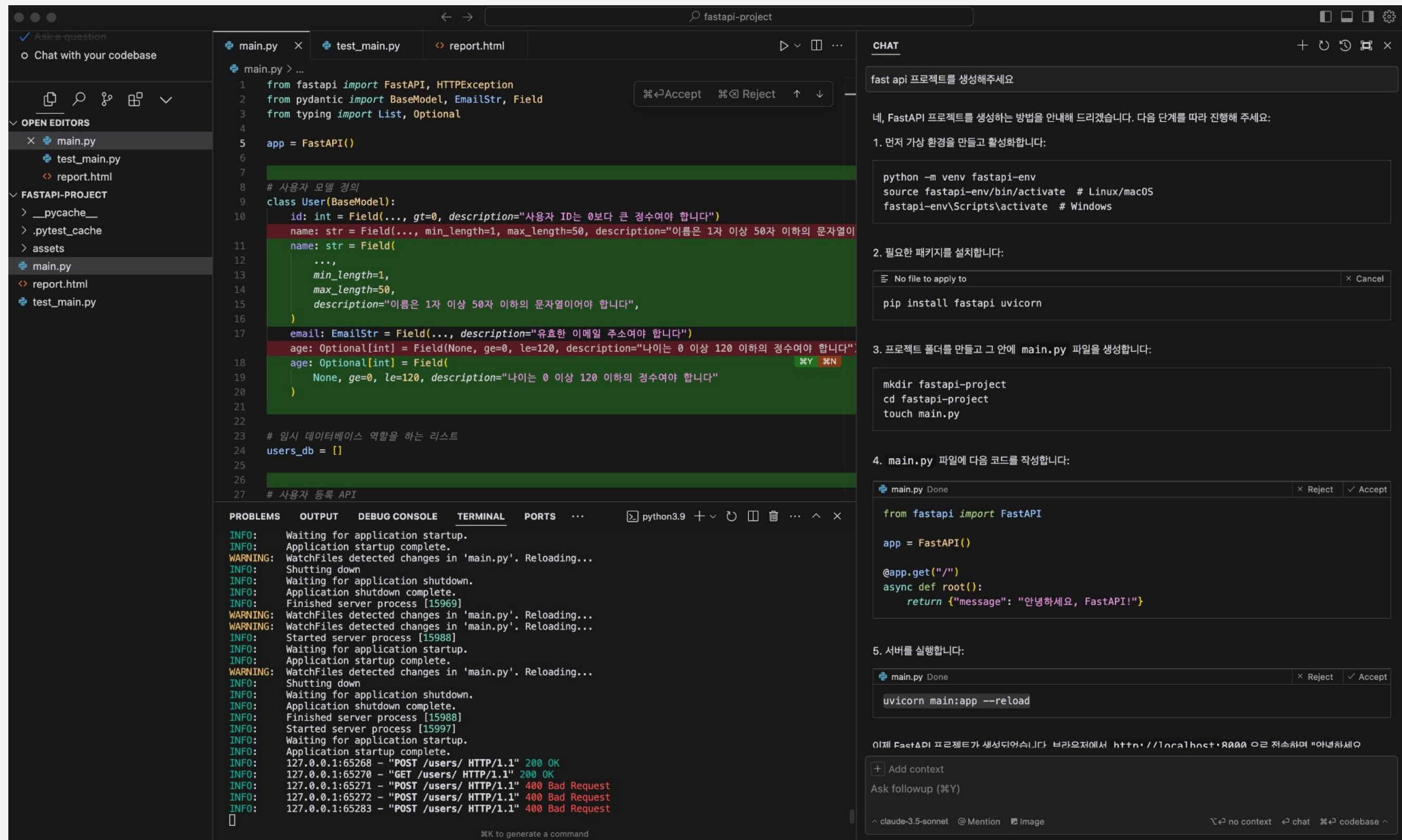


코드 디버깅



코드 리뷰 수행

AI 코드 에디터- 프로젝트 골격 생성



AI 코드 에디터 - 기능 추가

The screenshot displays an AI code editor interface with a dark theme. The main editor window shows a Python file named `main.py` with the following code:

```
1 from fastapi import FastAPI, HTTPException
2 from pydantic import BaseModel, EmailStr, Field
3 from typing import List, Optional
4
5 app = FastAPI()
6
7
8 # 사용자 모델 정의
9 class User(BaseModel):
10     id: int = Field(..., gt=0, description="사용자 ID는 0보다 큰 정수여야 합니다")
11     name: str = Field(..., min_length=1, max_length=50, description="이름은 1자 이상 50자 이하의 문자열이어야 합니다")
12     email: EmailStr = Field(..., description="유효한 이메일 주소여야 합니다")
13     age: Optional[int] = Field(None, ge=0, le=120, description="나이는 0 이상 120 이하의 정수여야 합니다")
14
15 # 임시 데이터베이스 역할을 하는 리스트
16 users_db = []
17
18 # 사용자 등록 API
19 @app.post("/users/", response_model=User)
20 async def create_user(user: User):
21     # ID 중복 확인
22     if any(existing_user.id == user.id for existing_user in users_db):
23         raise HTTPException(status_code=400, detail="이미 존재하는 사용자 ID입니다")
24     users_db.append(user)
25     return user
26
27 # 모든 사용자 조회 API
28 @app.get("/users/", response_model=List[User])
29 async def read_users():
30     return users_db
31
32 # 특정 사용자 조회 API
33 @app.get("/users/{user_id}", response_model=User)
34 async def read_user(user_id: int):
35     user = next((user for user in users_db if user.id == user_id), None)
36     if user is None:
37         raise HTTPException(status_code=404, detail="사용자를 찾을 수 없습니다")
38     return user
39
40 # 사용자 정보 수정 API
41 @app.put("/users/{user_id}", response_model=User)
```

The chat window on the right shows a conversation with an AI assistant. The user asks: "사용자 정보를 등록하고 수정하고 삭제하고 조회하는 API를 만듭니다". The assistant responds: "네, 사용자 정보를 관리하는 API를 만들어 보겠습니다. 사용자 등록, 수정, 삭제, 조회 기능을 구현하겠습니다. 먼저 `main.py` 파일을 다음과 같이 수정해주세요:".

The chat window also shows a code snippet for the `main.py` file, which is a simplified version of the code in the main editor window.

The bottom of the editor shows a terminal window with the following output:

```
INFO: 127.0.0.1:65270 - "GET /users/ HTTP/1.1" 200 OK
INFO: 127.0.0.1:65271 - "POST /users/ HTTP/1.1" 400 Bad Request
INFO: 127.0.0.1:65272 - "POST /users/ HTTP/1.1" 400 Bad Request
INFO: 127.0.0.1:65283 - "POST /users/ HTTP/1.1" 400 Bad Request
```

AI 코드 에디터 - API Spec 만들기

```
main.py > update_user
77 |         return templates.TemplateResponse(
78 |             "index.html",
79 |             {"request": request, "users": users_db, "message": "사용자가 등록되었습니다"},
80 |         )
81 |
82 |
83 | # 사용자 등록 API
84 | @app.post("/users/", response_model=User)
85 | @app.post("/users/", response_model=User, summary="사용자 등록 API", description="새 사용자를 등록하는 API 엔드포인트입니다.")
86 | async def create_user(user: User):
87 |     # ID 중복 확인
88 |     if any(existing_user.id == user.id for existing_user in users_db):
89 |         raise HTTPException(status_code=400, detail="이미 존재하는 사용자 ID입니다")
90 |     users_db.append(user)
91 |     return user
92 |
93 | # 모든 사용자 조회 API
94 | @app.get("/users/", response_model=List[User])
95 | @app.get("/users/", response_model=List[User], summary="모든 사용자 조회", description="등록된 모든 사용자의 정보를 조회합니다.")
96 | async def read_users():
97 |     return users_db
98 |
99 | # 특정 사용자 조회 API
100 | @app.get("/users/{user_id}", response_model=User)
101 | @app.get("/users/{user_id}", response_model=User, summary="특정 사용자 조회", description="지정된 ID의 사용자 정보를 조회합니다.")
102 | async def read_user(user_id: int):
103 |     user = next((user for user in users_db if user.id == user_id), None)
104 |     if user is None:
105 |         raise HTTPException(status_code=404, detail="사용자 ID를 찾을 수 없습니다")
106 |     return user
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

Requirement already satisfied: typing-extensions>=4.0 in ./venv (base) jerry ~/workspace/fastapi-project

uvicorn main:app --reload

INFO: Will watch for changes in these directories: ['/Use

INFO: Uvicorn running on http://127.0.0.1:8000 (Press CTR

INFO: Started reloader process [36379] using StatReload

INFO: Started server process [36409]

INFO: Waiting for application startup.

INFO: Application startup complete.

127.0.0.1:57563 - "GET /users/create HTTP/1.1" 200

127.0.0.1:57563 - "GET /static/styles.css HTTP/1.1"

127.0.0.1:57566 - "GET /static/chat_interface.js HT

127.0.0.1:57572 - "GET / HTTP/1.1" 200 OK

127.0.0.1:57572 - "GET /users/create HTTP/1.1" 200

127.0.0.1:57583 - "POST /users/create HTTP/1.1" 200

127.0.0.1:57585 - "GET / HTTP/1.1" 200 OK

127.0.0.1:57668 - "GET /docs HTTP/1.1" 200 OK

127.0.0.1:57668 - "GET /openapi.json HTTP/1.1" 200

사용자 관리 API 1.0.0 OAS 3.1

/openapi.json

사용자 정보를 관리하는 FastAPI 기반 애플리케이션입니다.

default

- GET / 메인 페이지
- GET /users/create 사용자 등록 폼
- POST /users/create 사용자 등록 처리
- GET /users/ 모든 사용자 조회
- POST /users/ 사용자 등록 API
- GET /users/{user_id} 특정 사용자 조회

지정된 ID의 사용자 정보를 조회합니다.

Parameters

Cancel

swagger 스펙에 설명 자동으로 추가해줘

네, Swagger 스펙에 설명을 자동으로 추가하기 위해 FastAPI의 기능을 활용하겠습니다. 각 엔드포인트와 모델에 대한 설명을 추가하겠습니다.

이 변경사항들은 Swagger UI에 자동으로 반영될 것입니다. 주요 변경 사항은 다음과 같습니다.

- FastAPI 인스턴스 생성 시 title, description, version 매개변수를 추가하여 API 전체에 대한 정보를 제공했습니다.

- User 모델에 Config 클래스를 추가하여 예시 데이터를 제공했습니다.

- 각 API 엔드포인트에 summary와 description 매개변수를 추가하여 해당 엔드포인트의 기능을 설명했습니다.

이렇게 하면 Swagger UI에서 각 엔드포인트와 모델에 대한 자세한 설명을 볼 수 있게 되어, API 사용자들이 더 쉽게 이해하고 사용할 수 있을 것입니다.

Completed Reapply all Save all Reject all Accept all

+ main.py x main.py (1-112) x

Edit, refactor, or add code (/ for commands)

Esc to close

claude-3.5-sonnet Submit

AI 코드 에디터 - 테스트 코드 추가

The screenshot displays an AI code editor interface with three main panels:

- Left Panel (WELCOME):** Shows a progress bar at 60% done and a list of tasks: Finish onboarding, Accept an autocomplete, Prompt an edit, Ask a question, and Chat with your codebase.
- Center Panel (test_main.py):** Displays the following Python code:

```
1 from fastapi.testclient import TestClient
2 from main import app
3
4 client = TestClient(app)
5
6 def test_create_user():
7     response = client.post(
8         "/users/",
9         json={"id": 1, "name": "홍길동", "email": "hong@example.com", "age": 30}
10    )
11    assert response.status_code == 200
12    assert response.json() == {
13        "id": 1,
14        "name": "홍길동",
15        "email": "hong@example.com",
16        "age": 30
17    }
18
19 def test_read_users():
20     response = client.get("/users/")
21     assert response.status_code == 200
22     assert len(response.json()) > 0
23
24 def test_read_user():
25     response = client.get("/users/1")
26     assert response.status_code == 200
27     assert response.json() == {
28         "id": 1,
29         "name": "홍길동",
30         "email": "hong@example.com",
31         "age": 30
32     }
33
34 def test_update_user():
35     response = client.put(
36         "/users/1",
37         json={"id": 1, "name": "홍길동", "email": "newhong@example.com", "age": 31}
38     )
39     assert response.status_code == 200
40     assert response.json() == {
41         "id": 1,
42         "name": "홍길동",
43         "email": "newhong@example.com",
44         "age": 31
45     }
46
```
- Right Panel (CHAT):** Contains instructions and generated test code.

1. 최근 변경사항 검토:

 - 코드 저장소의 커밋 히스토리를 확인하세요.
 - 최근 추가된 기능이나 수정된 부분을 파악하세요.

2. 새로운 테스트 케이스 작성:

 - 각 새로운 기능에 대해 적절한 테스트를 추가하세요.
 - 변경된 로직에 대해 기존 테스트를 수정하거나 새 테스트를 작성하세요.

3. 예시 테스트 코드:

새로운 사용자 등록 기능이 추가되었다고 가정해 봅시다. 다음과 같은 테스트를 추가할 수 있습니다:

```
import pytest
from app.models import User
from app.services import register_user

def test_user_registration_success():
    # 새 사용자 등록 테스트
    user_data = {
        "username": "newuser",
        "email": "newuser@example.com",
        "password": "securepassword123"
    }
    new_user = register_user(user_data)

    assert isinstance(new_user, User)
    assert new_user.username == "newuser"
    assert new_user.email == "newuser@example.com"

def test_user_registration_duplicate_username():
    # 중복된 사용자명으로 등록 시도 테스트
    existing_user_data = {
        "username": "existinguser",
        "email": "existing@example.com",
        "password": "password123"
    }
    register_user(existing_user_data)

    duplicate_user_data = {
        "username": "existinguser",
        "email": "new@example.com",
        "password": "newpassword123"
    }
```

AI 코드 에디터 - 테스트 결과 생성

report.html

Report generated on 23-Sep-2024 at 11:02:49 by [pytest-html](#) v4.1.1

Environment

Python	3.9.1
Platform	macOS-14.0-arm64-arm-64bit
Packages	• pytest: 7.4.2 • pluggy: 1.0.0
Plugins	• html: 4.1.1 • metadata: 3.1.1 • anyio: 3.7.1 • hydra-core: 1.2.0
JAVA_HOME	/Library/Java/JavaVirtualMachines/zulu-15.jdk/Contents/Home

Summary

6 tests took 14 ms.

(Un)check the boxes to filter the results.

☒ 0 Failed, ☒ 6 Passed, ☒ 0 Skipped, ☒ 0 Expected failures, ☒ 0 Unexpected passes, ☒ 0 Errors, ☒ 0 Reruns

[Show all details](#) / [Hide all details](#)

Result 	Test	Duration	Links
Passed	test_main.py::test_create_user	6 ms	
Passed	test_main.py::test_read_users	2 ms	
Passed	test_main.py::test_read_user	1 ms	
Passed	test_main.py::test_update_user	2 ms	
Passed	test_main.py::test_delete_user	2 ms	
Passed	test_main.py::test_read_deleted_user	2 ms	

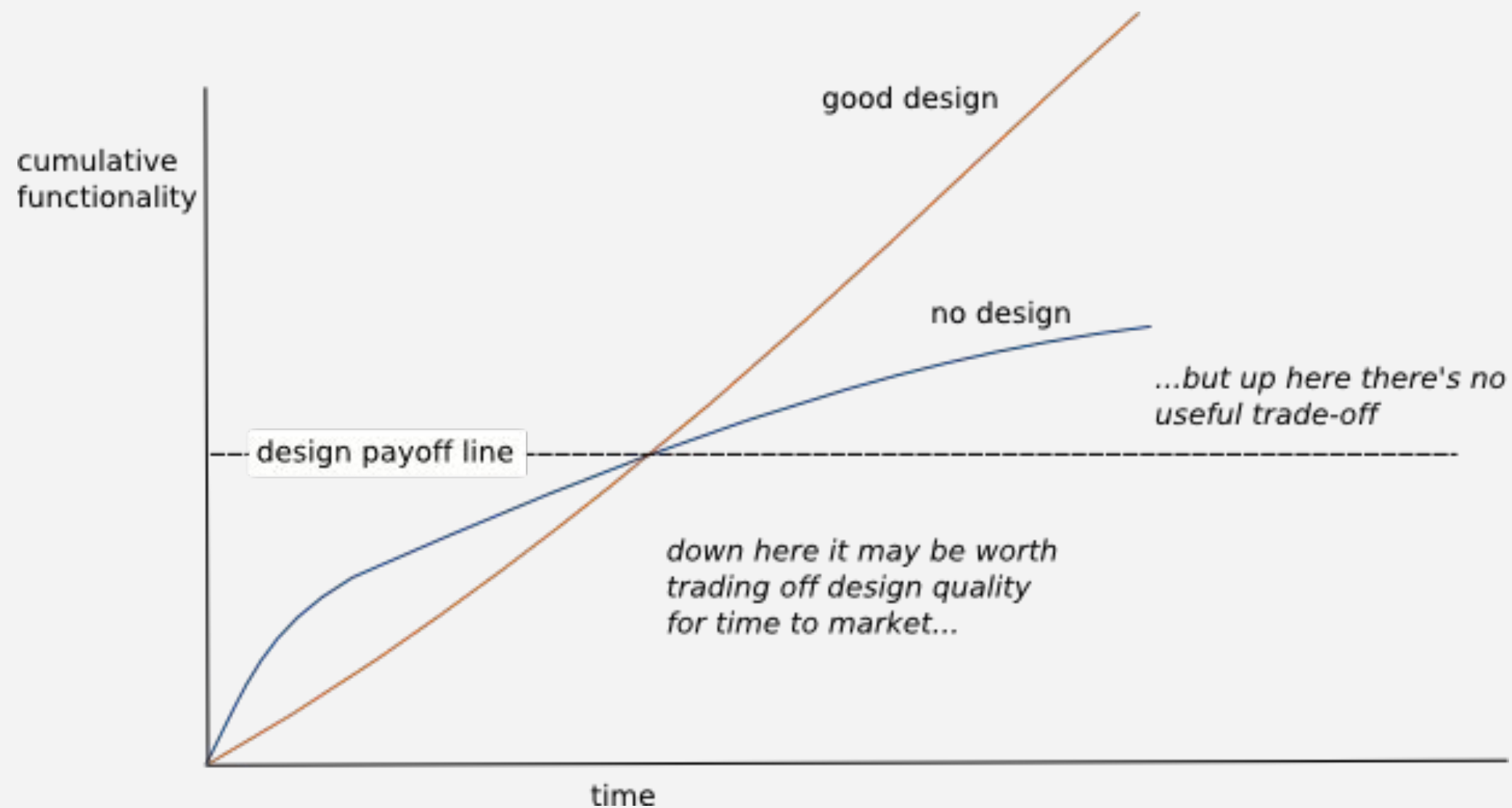
AI 코드 에디터 - Debug with AI

The screenshot displays an AI code editor interface with three main panels:

- Code Editor:** Shows a Python file named `test_main.py` with the following code:

```
1 from fastapi.testclient import TestClient
2 from main import app
3
4 client = TestClient(app)
5
6 def test_create_user():
7     response = client.post(
8         "/users/",
9         json={"id": 1, "name": "홍길동", "email": "hong@example.com", "age": 30}
10    )
11    assert response.status_code == 200
12    assert response.json() == {
13        "id": 1,
14        "name": "홍길동",
15        "email": "hong@example.com",
16        "age": 30
17    }
18
19 def test_read_users():
20     response = client.get("/users/")
21     assert response.status_code == 200
22     assert len(response.json()) > 0
23
24 def test_read_user():
25     response = client.get("/users/1")
26     assert response.status_code == 200
27     assert response.json() == {
28         "id": 1,
29         "name": "홍길동",
30         "email": "hong@example.com",
31         "age": 30
32     }
```
- CHAT:** Contains a message from the user: "Please help me debug this code. Only debug the latest error." The AI responds: "이 오류를 디버그하는 데 도움을 드리겠습니다. 가장 최근의 오류에 초점을 맞추겠습니다. 오류 메시지를 보면 다음과 같은 문제가 있습니다: E ModuleNotFoundError: No module named 'fastapi'". The AI then provides a list of steps to resolve the issue: 1. FastAPI가 설치되어 있는지 확인하세요. 2. 가상 환경을 사용 중이라면, 가상 환경이 활성화되어 있고 FastAPI가 해당 환경에 설치되어 있는지 확인하세요. 3. 설치 후에도 문제가 지속된다면, Python 인터프리터가 올바른 환경을 가리키고 있는지 확인하세요. 4. `pip list` 명령어를 실행하여 FastAPI가 설치된 패키지 목록에 있는지 확인하세요.
- Terminal:** Shows the output of the command `pip install fastapi`. The output indicates that FastAPI is already installed: "Requirement already satisfied: fastapi in /Users/jerryjung/.pyenv/versions/3.9.1/lib/python3.9/site-packages (from fastapi)".

AI 코드 에디터 - 코드 리팩토링

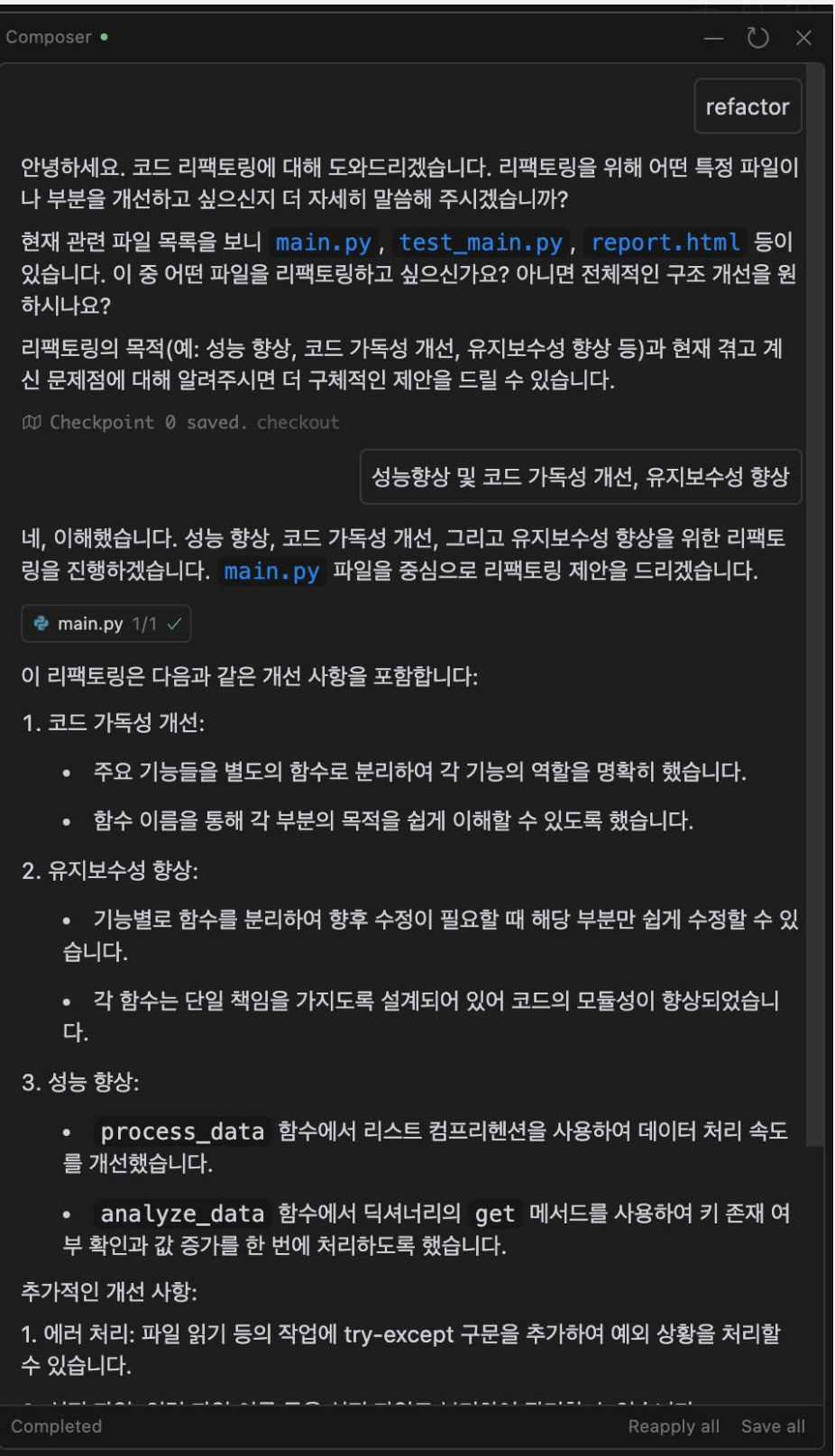
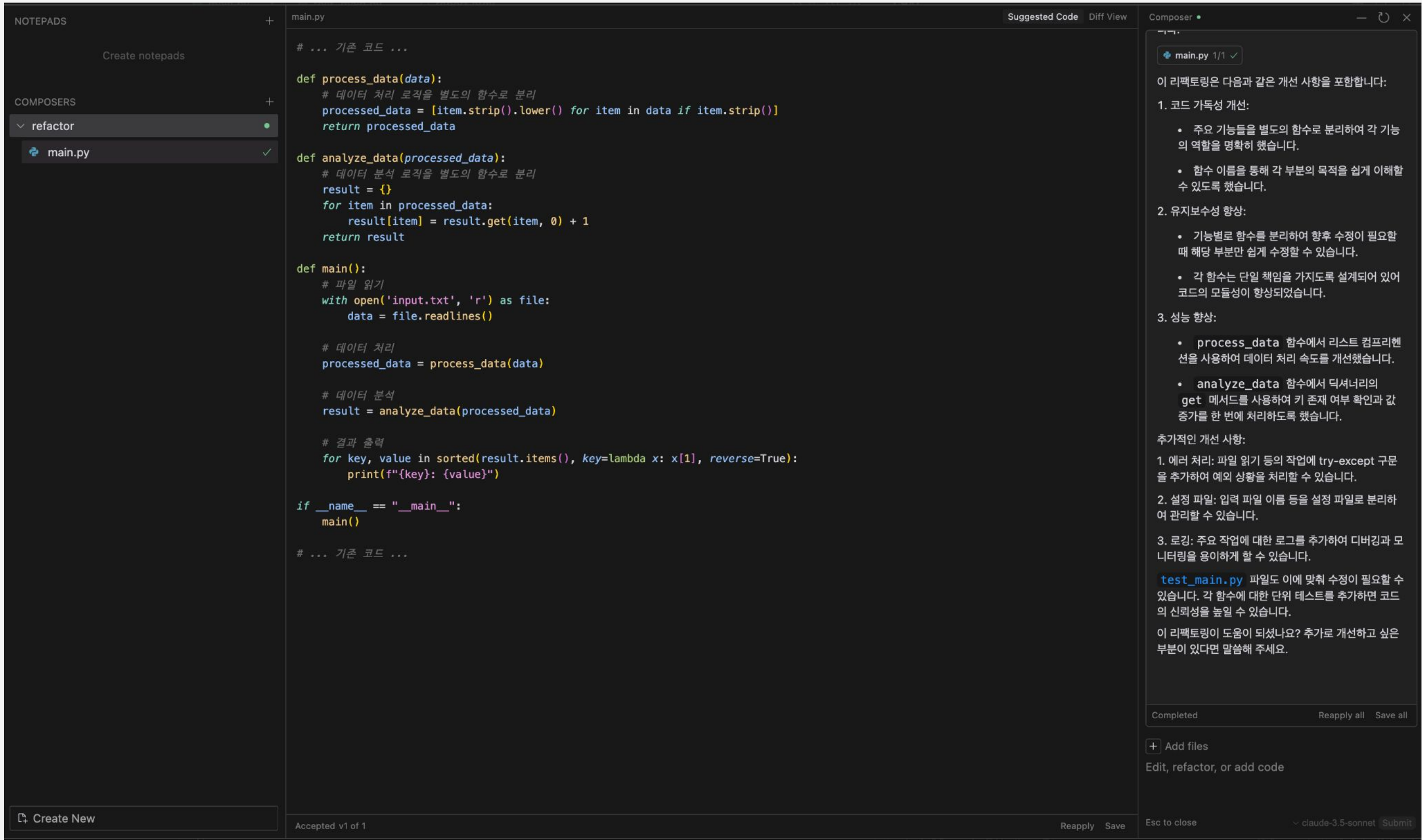


지속적 코드 개선의 중요성

가상 그래프 플롯은 두 가지 상상의 고정관념적 프로젝트에 대한 기능(누적) 대 시간을 제공합니다. 하나는 좋은 디자인이고 다른 하나는 디자인이 없습니다. 디자인을 하지 않는 프로젝트는 사전 디자인이든 애자일 기법이든 디자인 활동에 노력을 기울이지 않습니다. 이러한 활동에 노력을 기울이지 않기 때문에 이 프로젝트는 처음에 기능을 더 빨리 생성합니다. 무디자인의 문제점은 디자인에 노력을 기울이지 않으면 코드 기반이 악화되고 수정하기 어려워져 생산성이 낮아진다는 것입니다. 생산성은 라인의 기울기입니다. 좋은 디자인은 생산성을 더 일정하게 유지하므로 어느 시점(디자인 보상 라인)에서 무디자인 프로젝트의 누적 기능을 추월하고 계속해서 더 나은 성과를 낼 것입니다.

<https://martinfowler.com/bliki/DesignStaminaHypothesis.html>

AI 코드 에디터 - 코드 리팩토링



AI 코드 에디터- UI만들기

사용자 관리

새 사용자 등록

사용자 목록

ID	이름	이메일	나이
13445	아	eee@test.com	13

Checkpoint 1 saved. checkout

ui도 만들어주세요

네, 사용자 인터페이스(UI)를 만들어 드리겠습니다. FastAPI와 함께 사용할 수 있는 간단한 HTML 템플릿을 만들어 보겠습니다. 이를 위해 FastAPI의 템플릿 렌더링 기능을 사용하겠습니다.

먼저 `main.py` 파일을 수정하고, 그 다음 HTML 템플릿을 만들겠습니다.

`main.py` 2/2 ✓

이제 HTML 템플릿을 만들어 보겠습니다. `templates` 디렉토리를 만들고 그 안에 다음 파일들을 생성해주세요:

`index.html` 1/1 ✓

`create_user.html` 1/1 ✓

마지막으로, 간단한 CSS 파일을 만들어 스타일을 추가하겠습니다. `static` 디렉토리를 만들고 그 안에 다음 파일을 생성해주세요:

이제 FastAPI 애플리케이션을 실행하면 간단한 웹 인터페이스를 통해 사용자를 관리할 수 있습니다. 메인 페이지에서는 사용자 목록을 볼 수 있고, "새 사용자 등록" 버튼을 통해 새로운 사용자를 추가할 수 있습니다.

이 UI는 기본적인 기능만 구현되어 있습니다. 사용자 정보 수정이나 삭제 기능을 추가하려면 추가적인 페이지와 폼을 만들어야 합니다. 필요하시다면 이러한 기능도 추가로 구현해 드릴 수 있습니다.

Completed

Reapply all Save all

AI 코드 에디터- UI변경

새 사용자 등록

사용자 ID를 입력해주세요 (0보다 큰 정수):

33443

이름을 입력해주세요 (1-50자):

김말똥

이메일 주소를 입력해주세요:

md@test.com

나이를 입력해주세요 (0-120, 선택사항):

여기에 입력하세요...

제출

돌아가기

Composer

main.py ✓

index.html ✓

create_user.html ✓

styles.css ✓

JS

...

+

↺

⌂

×

네, 입력창을 대화형 인터페이스로 변경하겠습니다. 이를 위해 JavaScript를 사용하여 단계별로 사용자 정보를 입력받는 방식으로 변경하겠습니다. create_user.html 파일을 수정하고, 새로운 JavaScript 파일을 추가하겠습니다.

먼저, create_user.html 파일을 다음과 같이 수정합니다:

create_user.html 2/2 ✓

그리고 static 폴더에 새로운 JavaScript 파일 chat_interface.js 를 생성하고 다음 내용을 추가합니다:

JS chat_interface.js 1/1 ✓

마지막으로, styles.css 파일에 다음 스타일을 추가합니다:

이렇게 변경하면 사용자 등록 과정이 대화형 인터페이스로 변경됩니다. 사용자는 챗봇과 대화하는 것처럼 단계별로 정보를 입력하게 되며, 최종적으로 입력한 정보를 확인하고 제출할 수 있습니다.

main.py 파일의 create_user 함수는 그대로 유지해도 됩니다. JavaScript에서 폼 데이터를 생성하여 제출하기 때문입니다.

이 변경사항을 적용하면 사용자 등록 과정이 더욱 대화형이고 사용자 친화적으로 변경될 것입니다.

gpt-4o

cursor-small

claude-3.5-sonnet

gpt-4o-mini

o1-mini

o1-preview

⌘ / toggle model

Completed

+ Add files

Edit, refactor, or add code (/ for commands)

Esc to close

Save all

Submit

AI 코드 에디터 - 잘 사용하는 법

“2027년까지 개발자 70%가 사용한다는데...” AI 코딩 도구의 함정

좋은 것 알지만...주의해야 할 3가지

그러나 깃클리어는 AI 지원 프로그래밍의 사용이 증가함에 따라 "이탈", "이동", "복사/붙여넣기" 코드의 양이 크게 증가한다는 사실도 발견했다.

'이탈'은 리포지토리에 푸시됐다가 2주 이내에 되돌리거나 제거 또는 업데이트되는 코드다. 2023년 이전에는 개발자가 모든 코드를 직접 작성하는 경우는 상대적으로 드물었고, 코드 이탈은 3~4%에 불과했다. 하지만 코파일럿이 베타 버전으로 제공된 첫해, 즉 챗 GPT가 출시된 해에는 전체 코드 이탈이 9%나 급증했다.

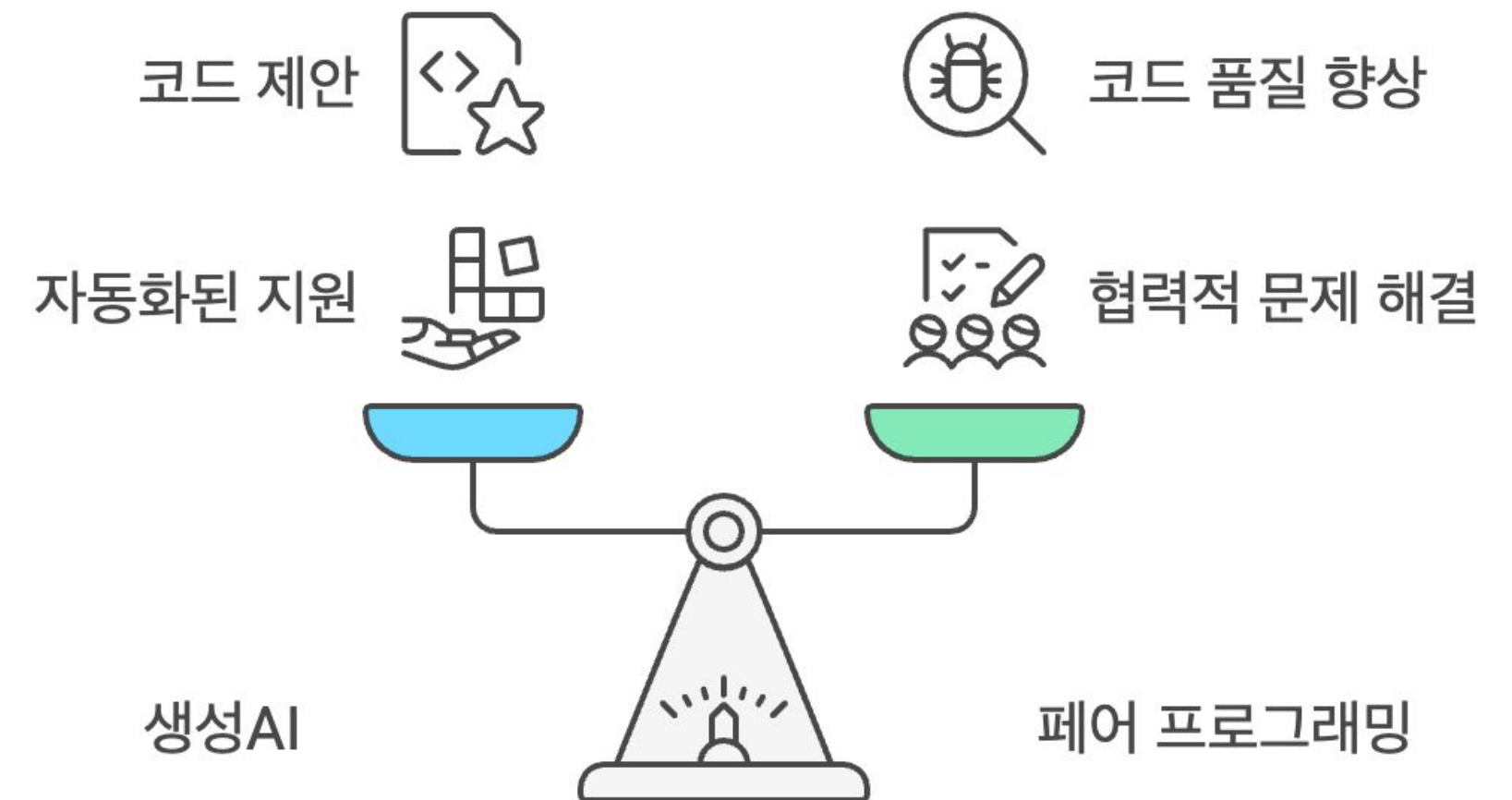
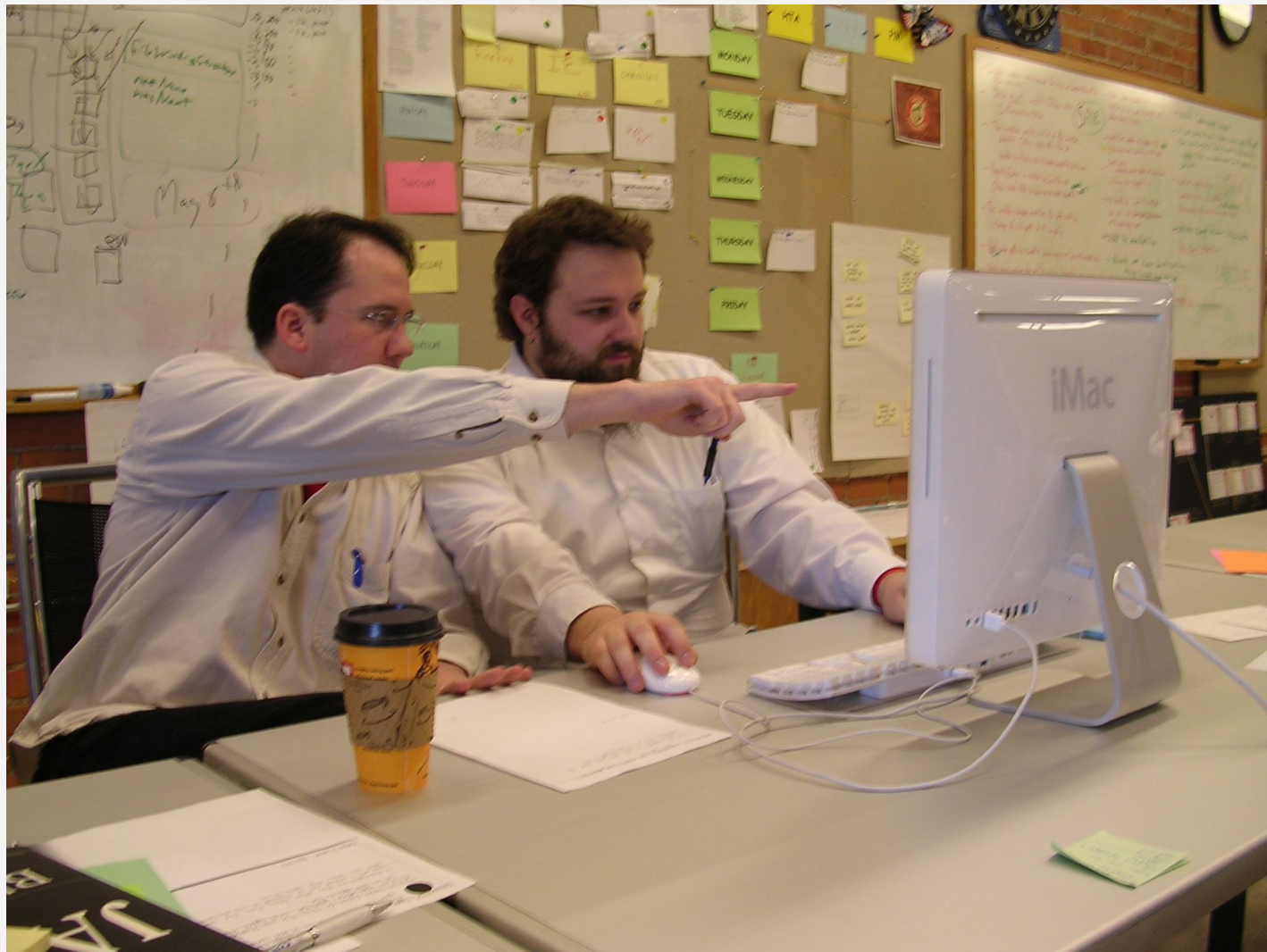
2022년부터 2023년까지 AI 어시스턴트 도구의 증가는 리포지토리에 푸시되는 '실수 코드'와 밀접한 상관관계가 있었다. 코드 생성에 사용되는 코파일럿 보급률은 2021년에 0%, 2022년에 5~10%, 2023년에 30%로 나타났다고 깃클리어는 밝혔다. 또한 "현재 패턴이 2024년까지 계속된다면 2주 이내에 전체 코드 변경 사항의 7% 이상이 되돌려질 것이다. 2021년의 두 배에 달하는 비율"이라고 설명했다.

코드 복사/붙여넣기만큼 장기적인 코드 유지 관리에 치명적인 재앙은 없을 것이다. 재사용된 코드에는 이전의 실수, 보안 취약점 또는 기타 문제가 포함될 수 있다.

AI 코드 에디터 - 잘 사용하는 법

동료 프로그래밍 또는 **쌍 프로그래밍**은 애자일 소프트웨어 개발 중 하나로 하나의 컴퓨터에서 두 사람의 프로그래머가 작업하는 방법이다. 코드를 작성하는 사람이 진행자(driver)가 되고 다른 한 사람이 관찰자(observer, navigator)가 되어 코드 검토(code review)를 하며 프로그래밍을 작성한다. 두 프로그래머는 수시로 역할을 바꾼다. 코드 검토를 하는 동안에 관찰자는 전략적 방향을 제시하거나 고려하여 개선하기 위한 아이디어와 해결 가능성이 있는 문제를 제시한다. 진행자는 작업을 완료하는 기술적 측면에 모든 관심을 집중시키고 관찰자는 안전망과 가이드의 역할을 담당하게 된다.

<https://ko.wikipedia.org/wiki>



소프트웨어 개발에서 AI 지원과 페어프로그래밍

Thank You



정유선

신세계 DT AI/ML 담당

jerryjung@apache.org