Python 설치

■ 파이썬 다운로드

https://www.python.org/downloads/windows/

Stable Releases

•Python 3.11.8 - Feb. 6, 2024

•Note that Python 3.11.8 *cannot* be used on Windows 7 or earlier.

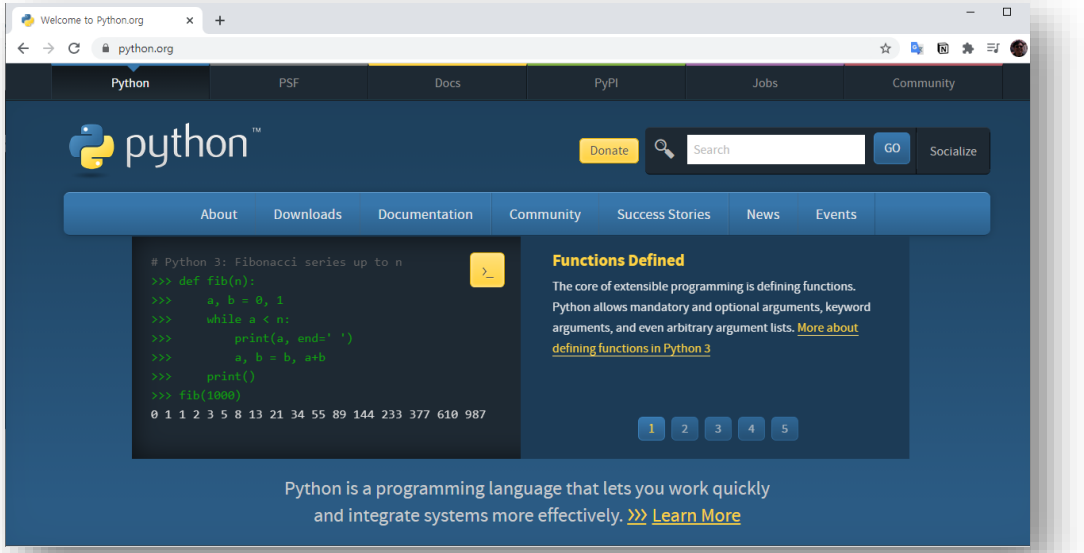
- Download Windows embeddable package (32-bit)
- Download Windows embeddable package (64-bit)
- Download Windows embeddable package (ARM64)
- Download Windows installer (32 -bit)
- Download Windows installer (64-bit)
- Download Windows installer (ARM64)

https://www.python.org/downloads/macos/

Stable Releases

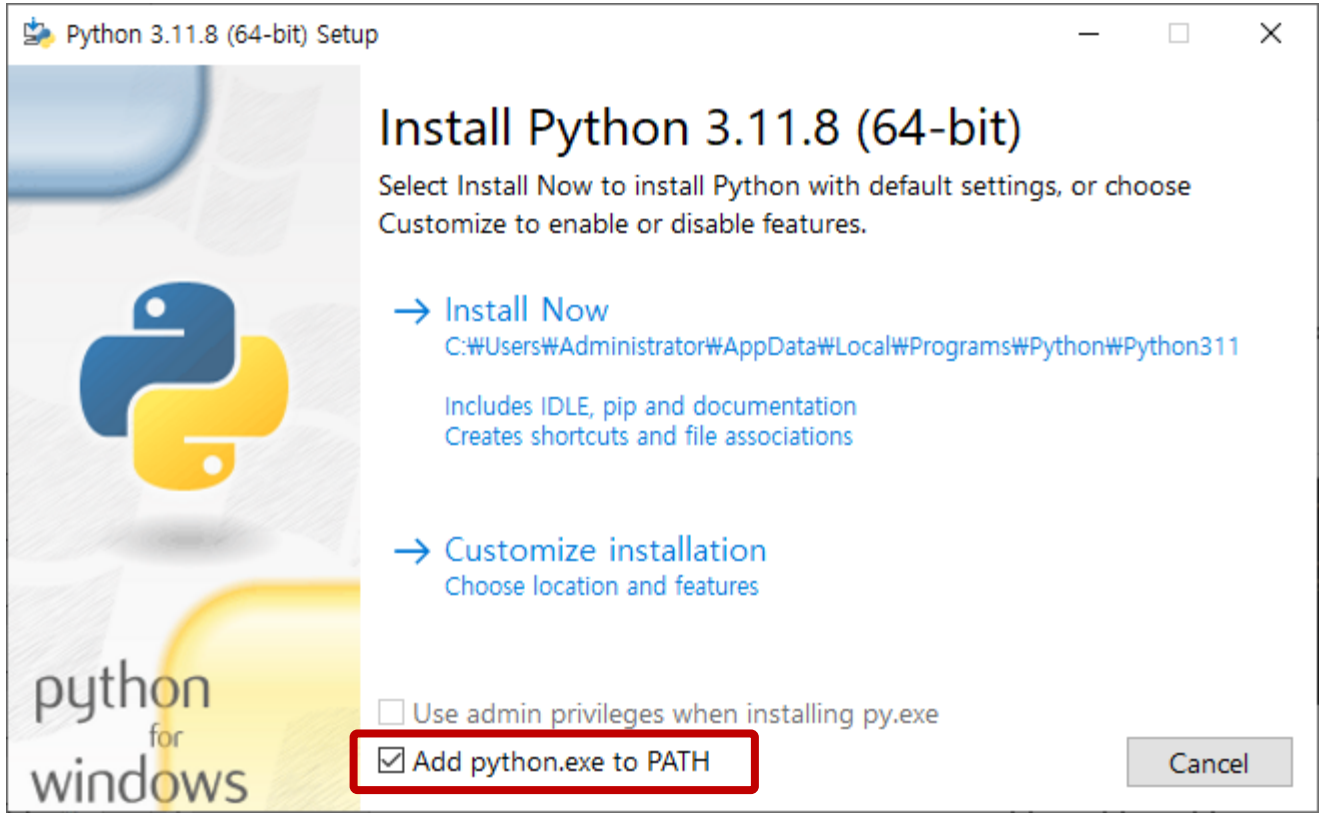
•Python 3.11.8 - Feb. 6, 2024

- Download macOS 64-bit universal2 installer



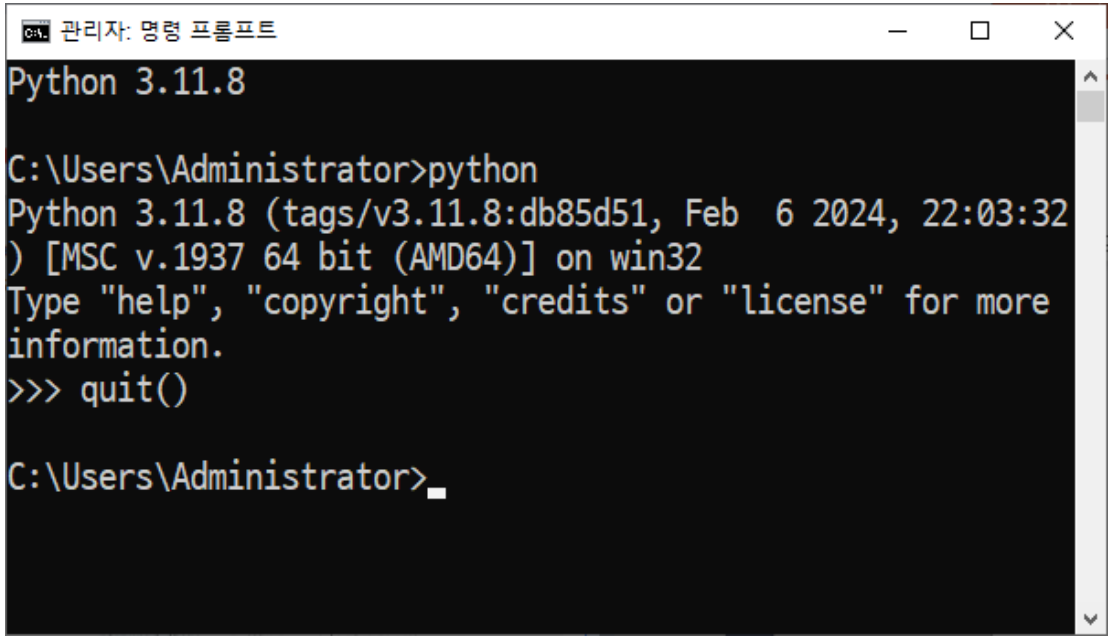
Python 설치

■ 파이썬 설치



■ 파이썬 실행

- 버전 확인 : python --version
- 실행 : python
- 종료 : quit()



Python 가상환경 설치 – Windows

프로젝트별로 독립된 파이썬 실행 환경을 사용할 수 있는 가상 환경(Virtual Environment) 구성을 권장합니다.

- 가상환경 생성 : `python -m venv py311`
- 가상환경 실행 : `py311\Scripts\activate.bat`
- 파이썬 패키지 설치 : `pip install jupyterlab notebook openai`
  - Jupyter Lab 실행 : `jupyter lab`
  - Jupyter Notebook 실행 : `jupyter notebook`
- 패키지 목록파일 만들기
  `pip freeze > requirements.txt`
- 패키지 목록파일로 패키지 설치 하는 방법
  `pip install -r requirements.txt`
- 파이썬 패키지 삭제 : `pip uninstall 패키지명`

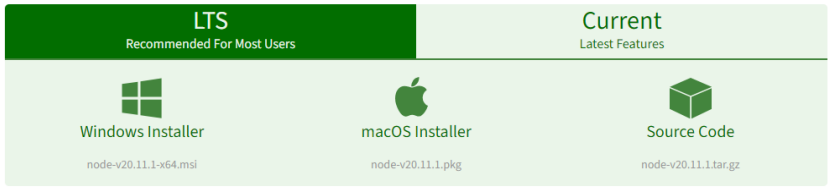
Python 가상환경 설치 – macOS/Linux

프로젝트별로 독립된 파이썬 실행 환경을 사용할 수 있는 가상 환경(Virtual Environment) 구성을 권장합니다.

- 가상환경 생성 : `python3 -m venv py311`
- 가상환경 실행 : `source py311/bin/activate`
- 파이썬 패키지 설치 : `pip3 install jupyterlab notebook openai`
  - Jupyter Lab 실행 : `jupyter lab`
  - Jupyter Notebook 실행 : `jupyter notebook`
- 패키지 목록파일 만들기
  ```
  pip3 freeze > requirements.txt
  ```
- 패키지 목록파일로 패키지 설치 하는 방법
  ```
  pip3 install -r requirements.txt
  ```
- 파이썬 패키지 삭제 : `pip3 uninstall jupyterlab`

Flowise 설치

NodeJS 설치 : https://nodejs.org/en/download



Flowise 설치
npm install -g flowise

Flowise 시작
npx flowise start

Flowise 사용
http://localhost:3000 접속

Developers (선택사항)

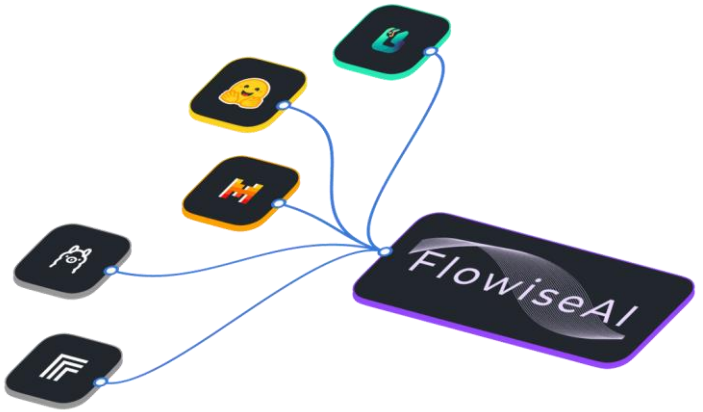
1. Yarn 설치
npm i -g yarn

2. Repository 복제
git clone https://github.com/FlowiseAI/Flowise.git

3. 모듈 설치
cd Flowise
yarn install
yarn build

4. App 실행
yarn start

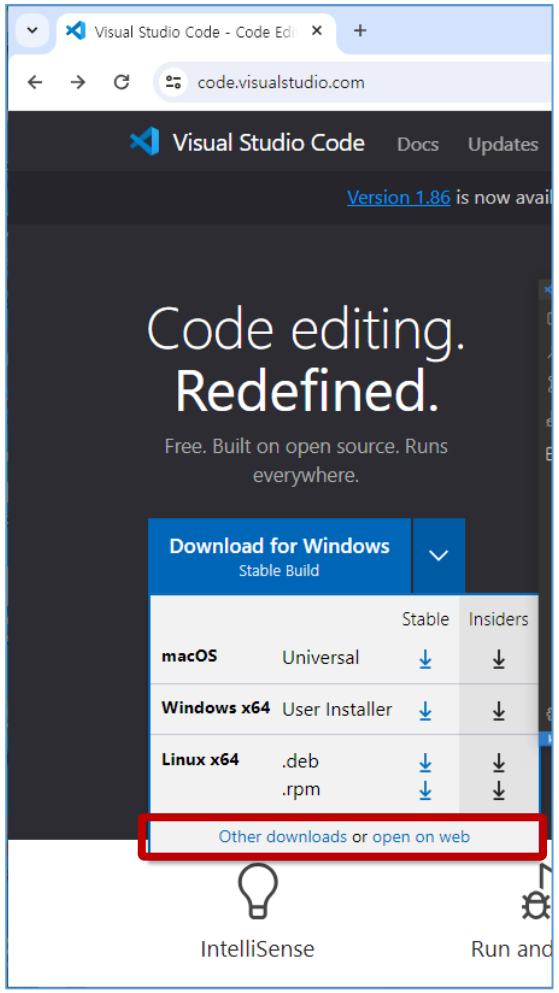
6. http://localhost:3000 접속

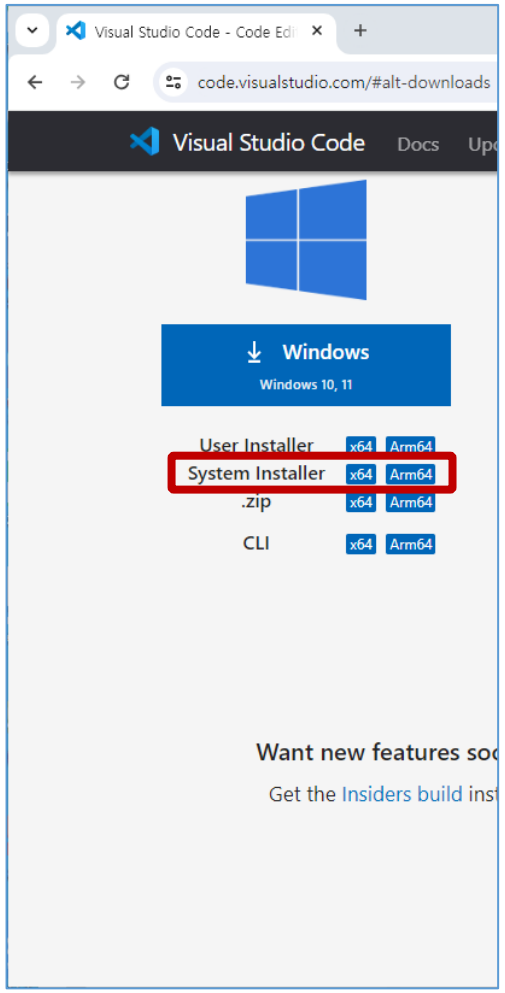


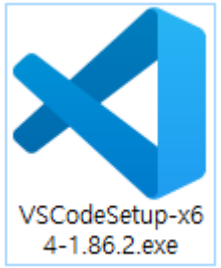
https://github.com/FlowiseAI/Flowise

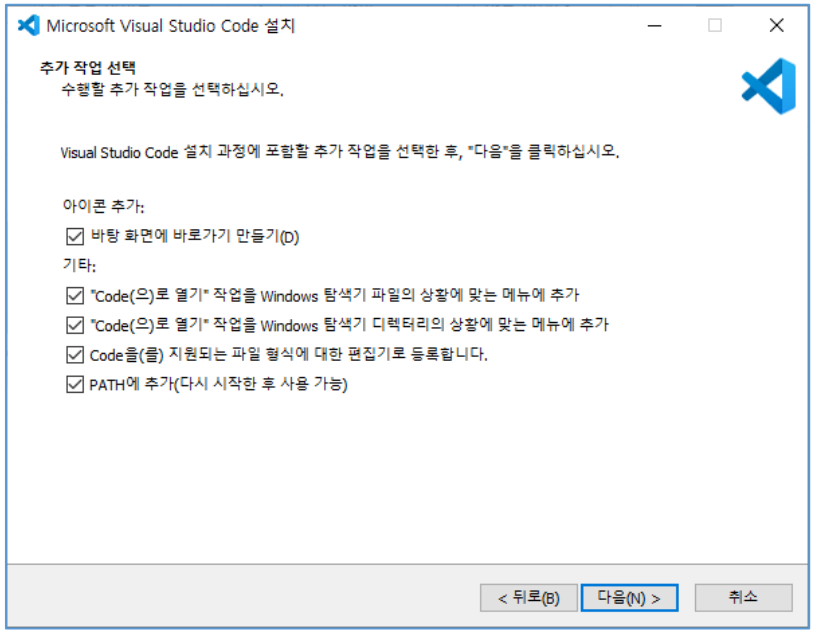
VS Code 설치 – Windows

■ 설치 프로그램 https://code.visualstudio.com/



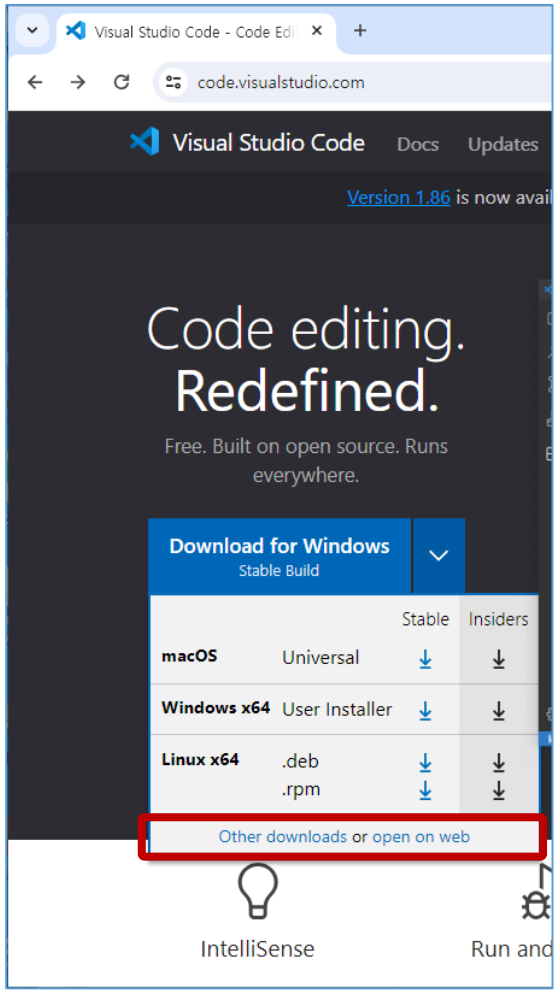


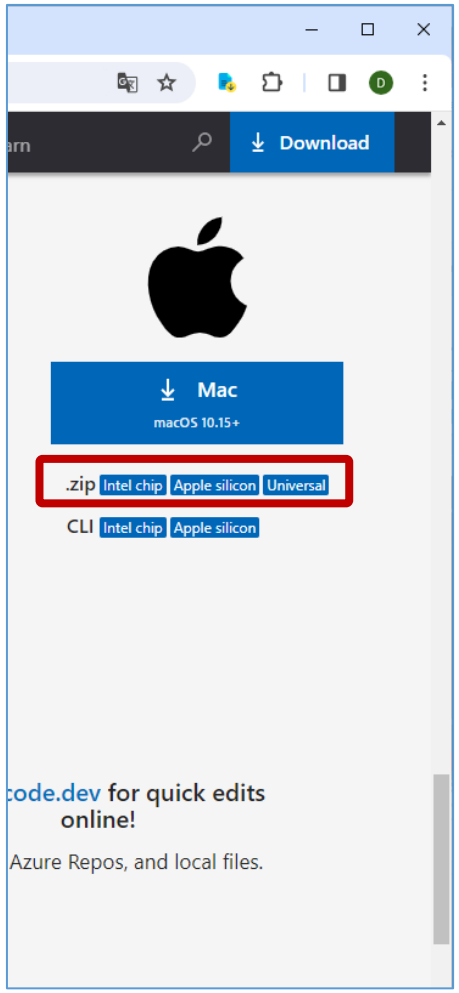




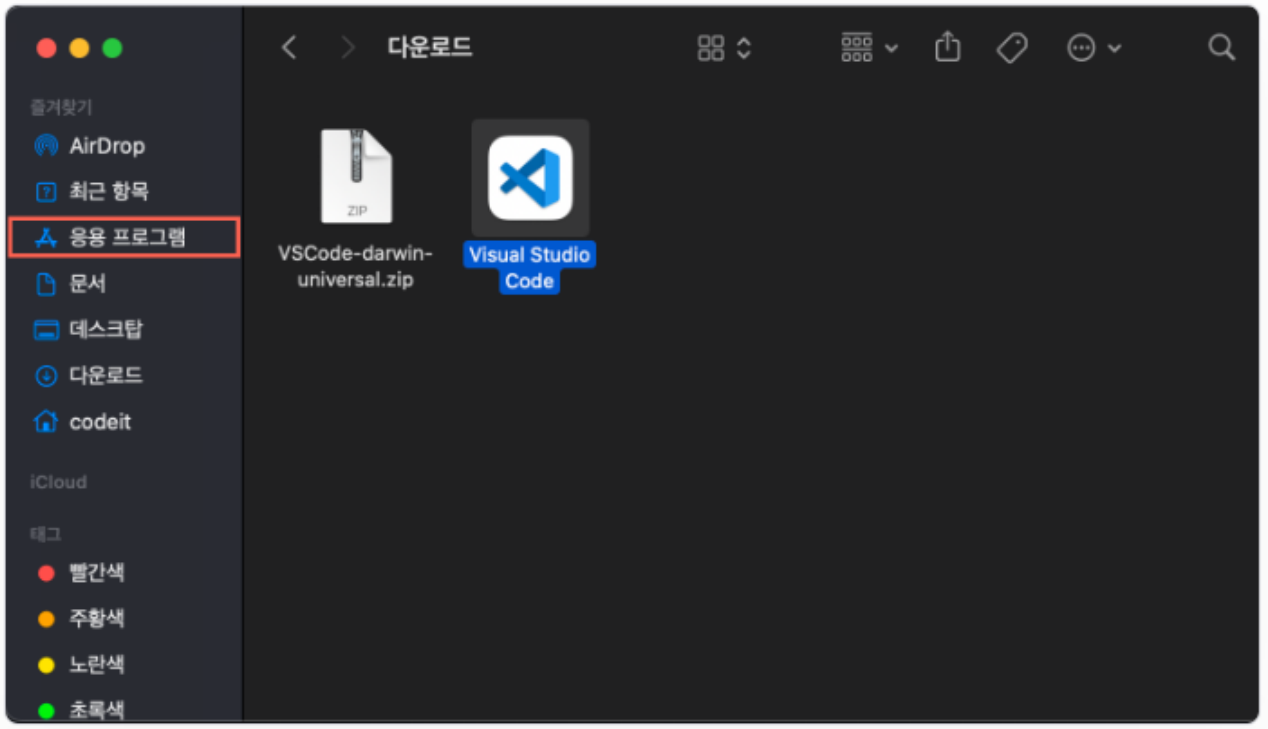
VS Code 설치 – macOS

■ 설치 프로그램 https://code.visualstudio.com/

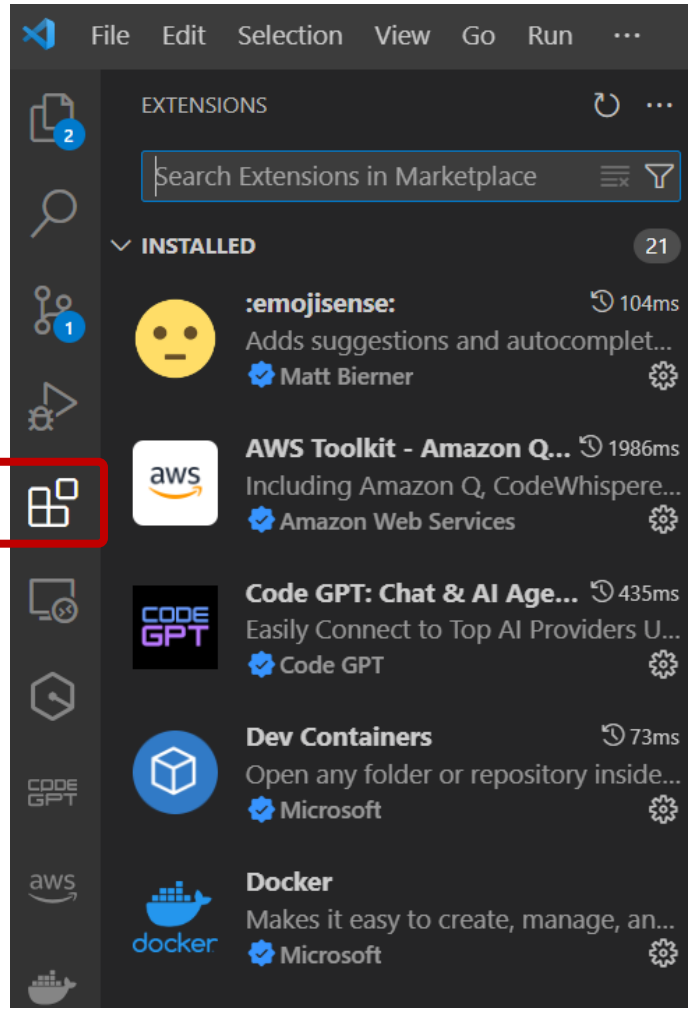




Visual Studio Code를 응용 프로그램(Applications) 폴더로 옮겨 주세요.



VS Code Extension 설치

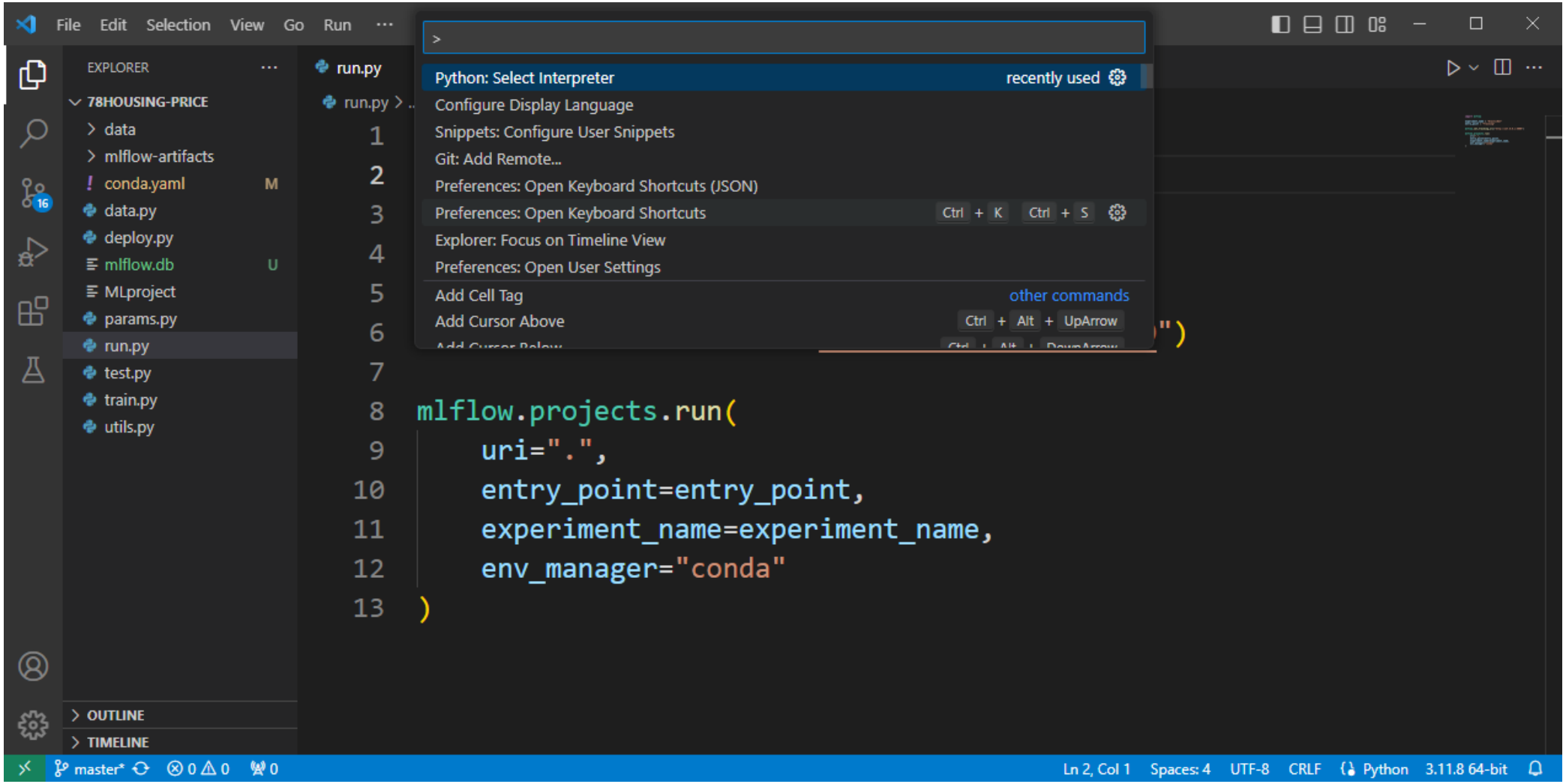


- **Python** : 파이썬에 대한 풍부한 지원 제공, IntelliSense(Pylance), 린팅, 디버깅, 코드 탐색 등의 기능을 제공
- **Jupyter** : Jupyter 노트북 지원
- **Black Formatter** : Python 파일에 대한 포맷팅 지원 제공
- **vscode-icons** : Visual Studio Code용 아이콘
- **TODO Highlight** : 코드 내에서 TODO, FIXME 및 기타 주석을 강조 표시
- **Todo Tree** : TODO, FIXME와 같은 주석 태그를 빠르게 검색하고 활동 표시줄의 트리 보기에 표시
- **Path Intellisense** : 파일 이름 자동 완성
- **Live Preview** : 웹페이지 미리 보기
- **REST Client** : REST 클라이언트

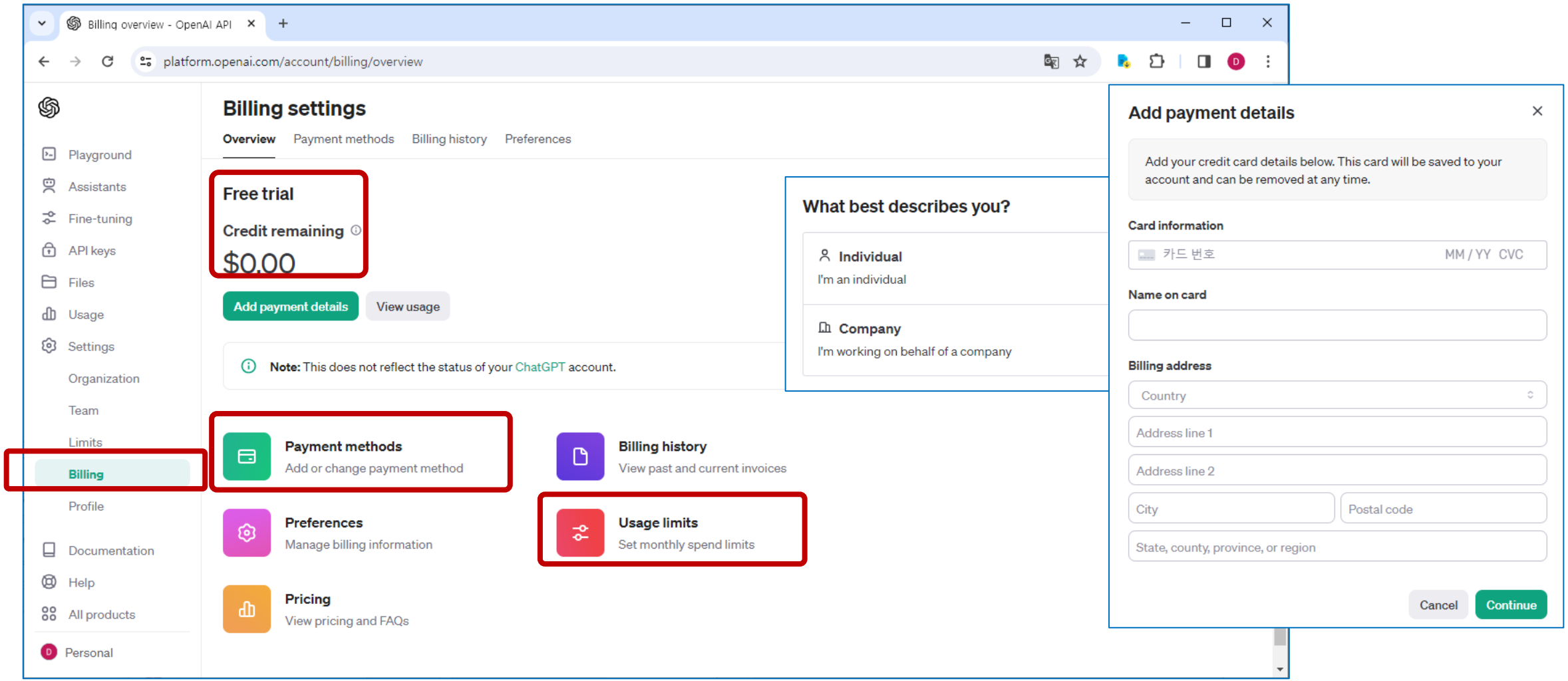
VS Code : Python 선택

- **Windows : Ctrl + Shift + P**
  Python : Select Interpreter

- **macOS : ⌘ + ⇧ + P** ⇧⇧
  Python : Select Interpreter
  Shell Command: Install 'code' command in PATH

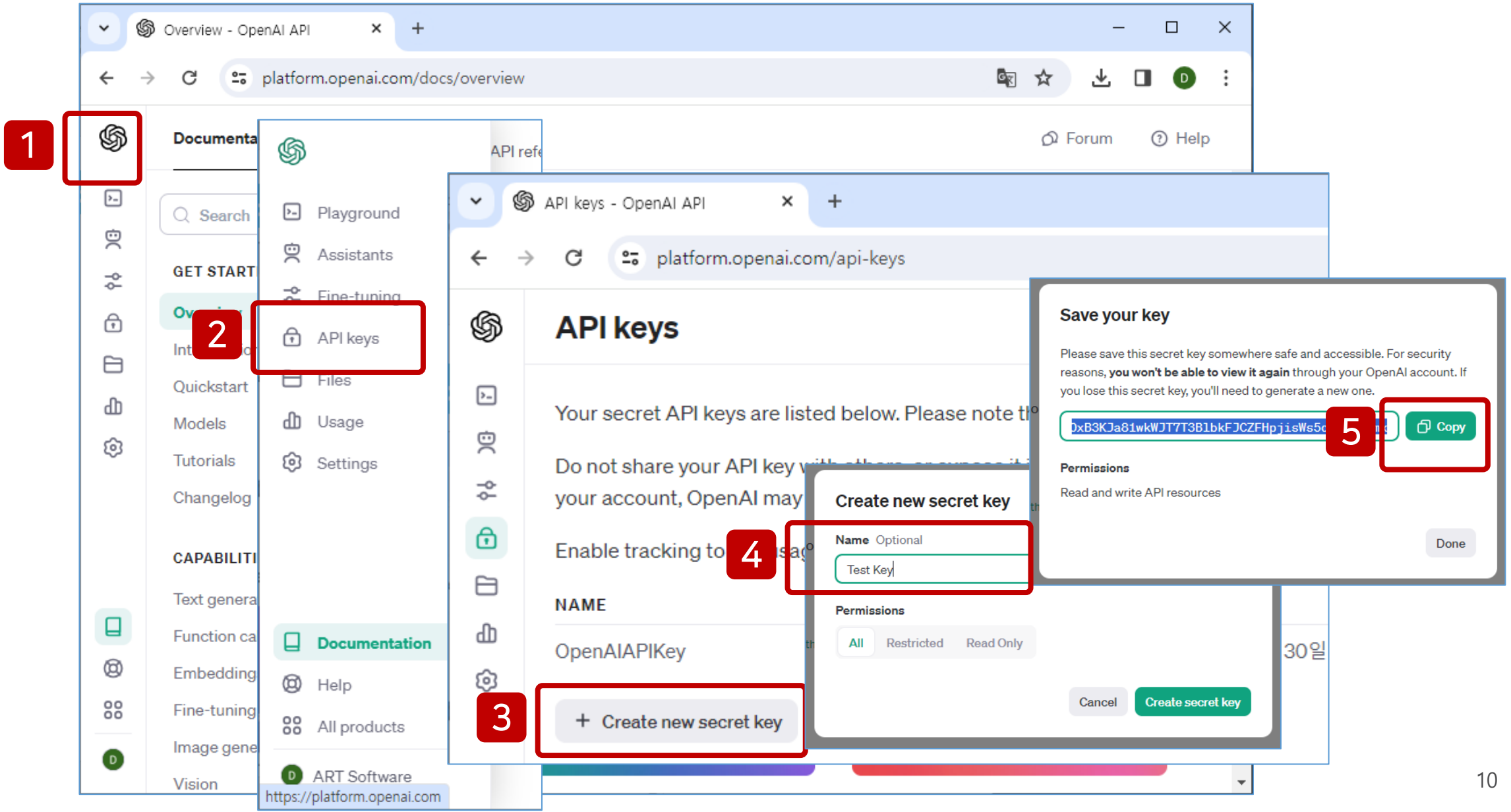


OpenAI API 유료사용 https://platform.openai.com/account/billing/overview



OpenAI API Key 생성

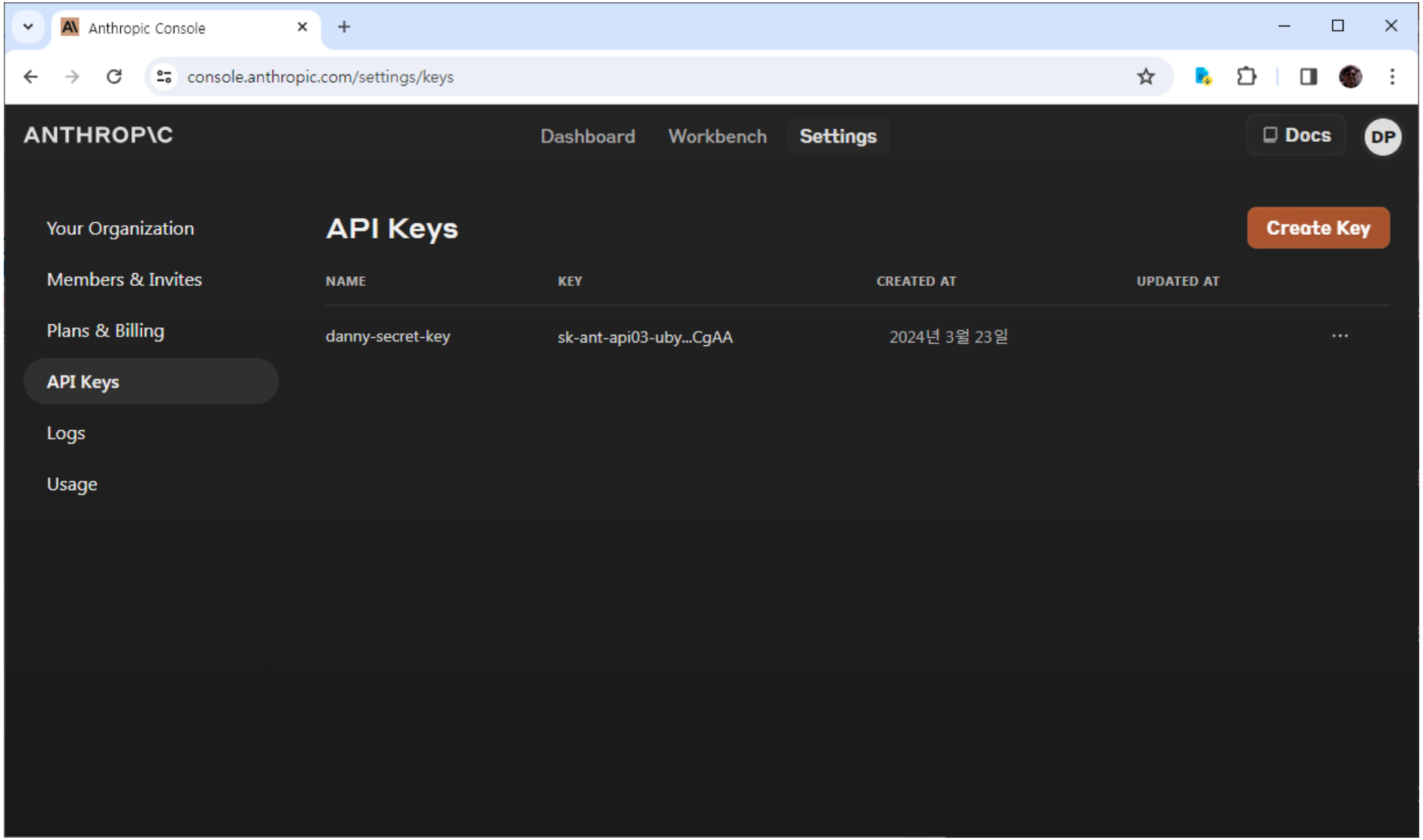
명령 프롬프트 에서 아래 명령어 실행
setx OPENAI_API_KEY "sk-kcXMU...SN5rS"



Anthropic API 무료사용

https://console.anthropic.com/settings/keys

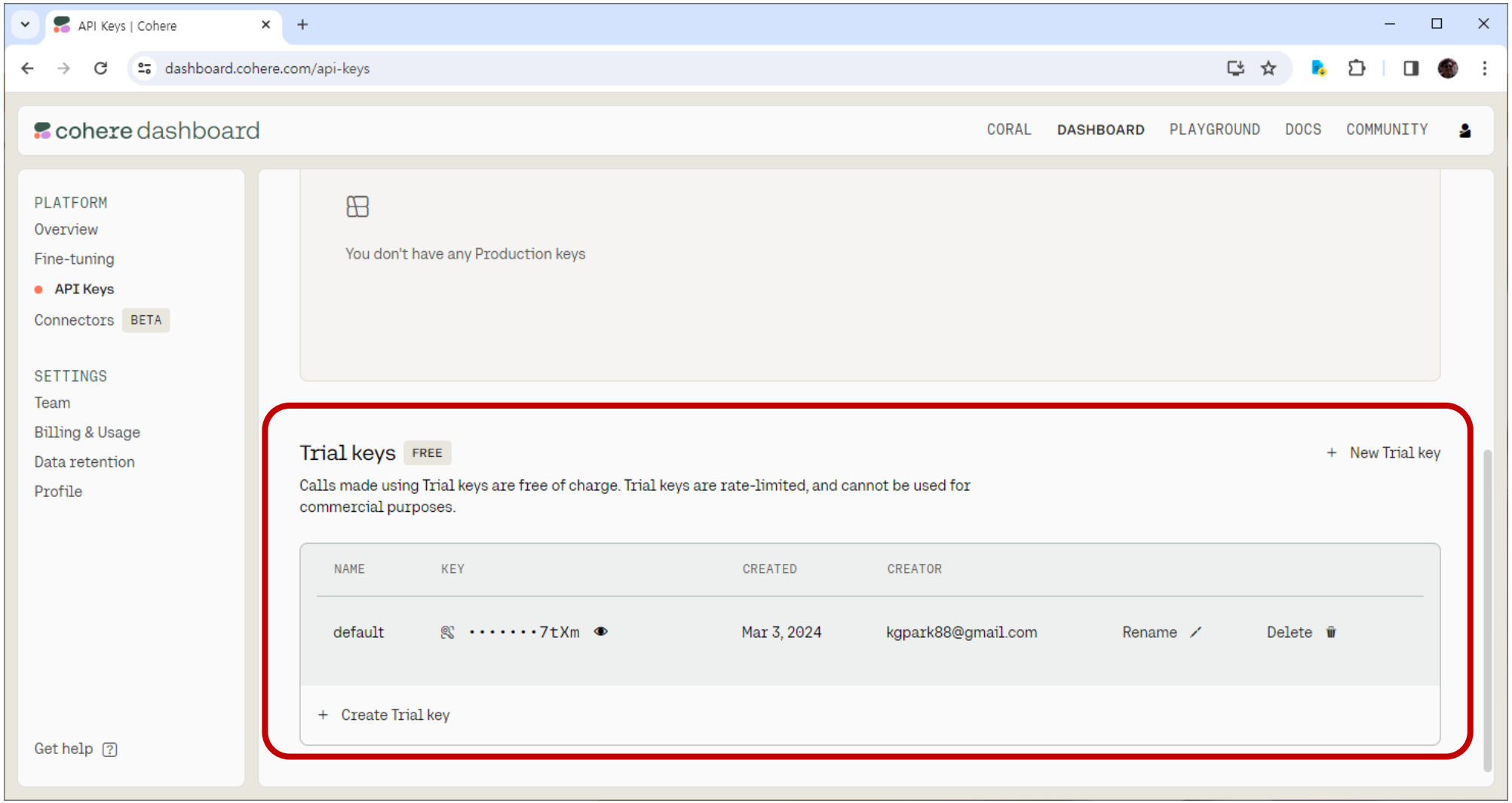
명령 프롬프트 에서 아래 명령어 실행
setx ANTHROPIC_API_KEY "sk-ant-api.....ulCgAA"



Cohere API 무료사용

https://dashboard.cohere.com/api-keys

명령 프롬프트 에서 아래 명령어 실행
setx COHERE_API_KEY "Axt...ZX7tXm"



OpenAI API 실습

openai_api.ipynb

```
MODEL = "gpt-3.5-turbo"
response = client.chat.completions.create(
    model=MODEL,
    messages=[
        {
            "role": "system",
            "content": "당신은 창의적인 감각으로 복잡한 프로그래밍 개념을 설명하는 데 능숙한 시인입니다.",
        },
        {"role": "user", "content": "생성형AI로 AI솔루션을 개발하는 것을 아름답게 표현하는 시를 작성해 주세요."},
    ],
    temperature=0,
)
```

[7] Python

Llama 2 (sLLM)

▪ sLLM(smaller Large Language Model , 소형 언어 모델)

- LLM과 비교했을 때 매개변수의 수가 수십 억~수백 억개로 비교적 크기가 작은 언어모델
- 비용절감, 보안, 특정 도메인에 활용 목적으로 사용
- 특정 도메인 사용용도로 sLLM 을 사용하는 경우가 많아지고 있음

▪ Ollama 설치

- 로컬 환경에서 다양한 언어 모델을 실행할 수 있게 지원하는 오픈소스
- 모델 종류 : https://ollama.com/library
- 설치 파일 다운로드 : https://ollama.ai/

▪ Llama 2 (Large Language Model Meta AI)

- 메타에서 공개한 상업적으로도 이용 가능한 오픈 소스 sLLM
- 설치 및 실행 : ollama run llama2

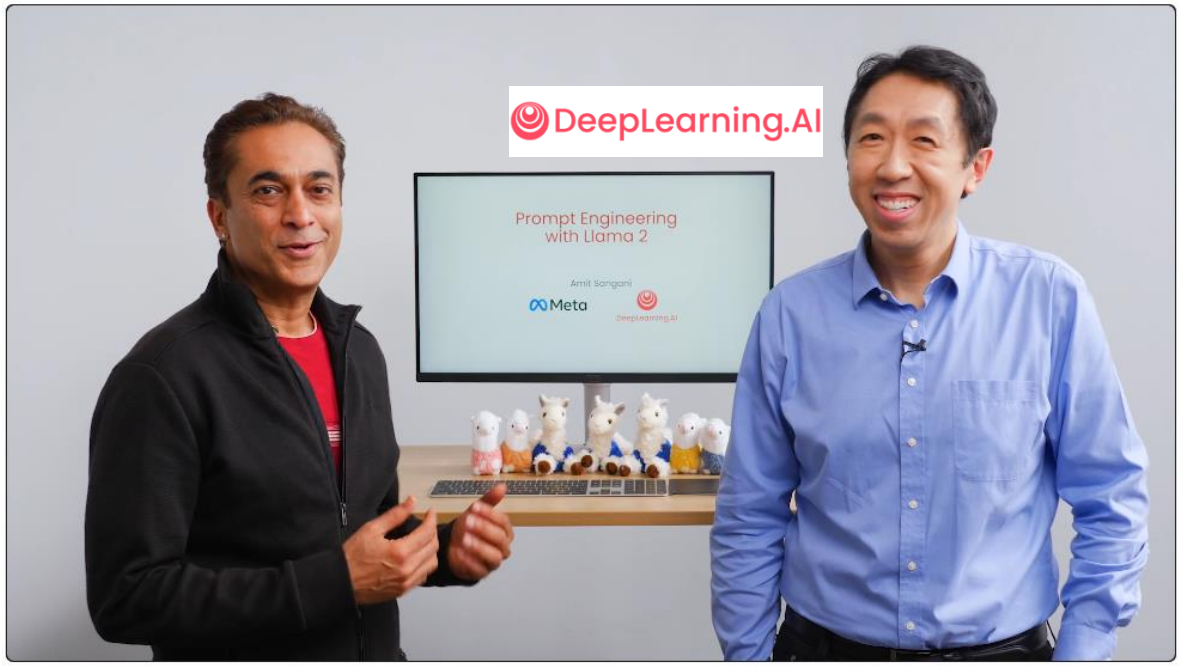
```
pulling 8934d96d3f08... 100% ▕▕ 3.8 GB
```

- 프로그램 개발 예시

```
from langchain_community.llms import Ollama
llm = Ollama(model="llama2")
llm.invoke("Hello")
```

Llama 2 (sLLM)

https://learn.deeplearning.ai/courses/prompt-engineering-with-llama-2/




```
"[INST] Help me write a birthday card… [/INST]"
```

instruction tags

start tags

```
prompt_chat = f"""
    <s>[INST]{user prompt 1}[/INST]
    Assistant: {model response 1}</s>
    <s>[INST]{user prompt 2}[/INST]
    Assistant: {model response 2}</s>
    …
    <s>[INST]{user prompt 3}[/INST]
```

end tags

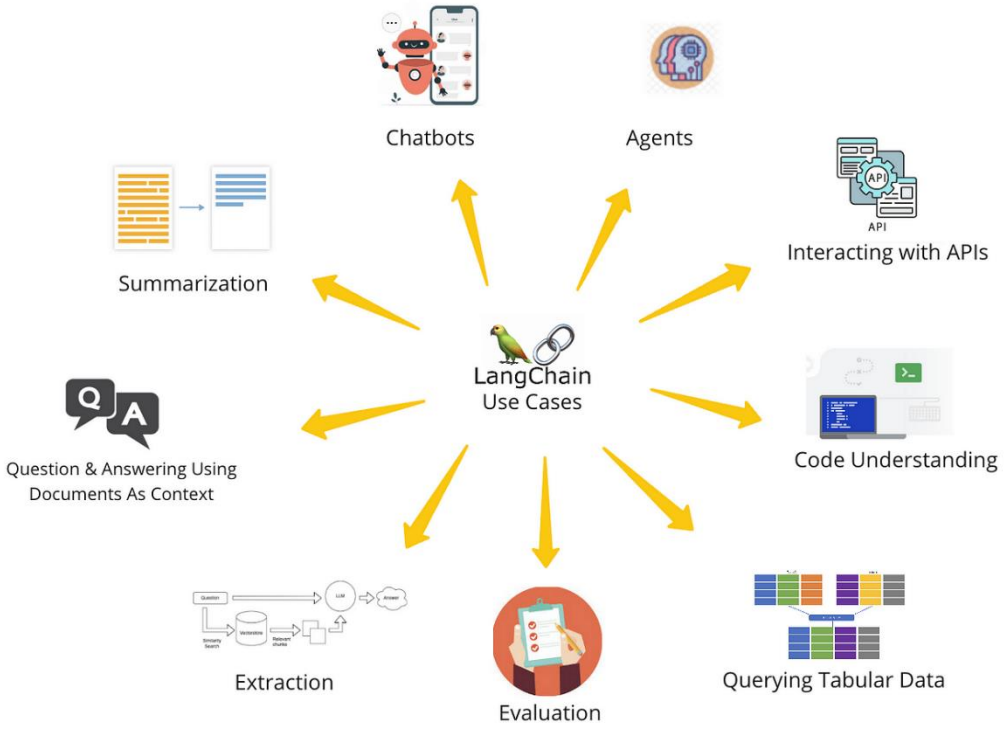
https://www.together.ai/

- 사이트 접속 및 회원가입
- 환경변수에 TOGETHER_API_KEY 값 추가
- ./code/llama/utils.py 파일 참고

명령 프롬프트 에서 아래 명령어 실행
setx TOGETHER_API_KEY "995e07d0bb7f148ba7ef"

LangChain

LangChain은 언어 모델로 구동되는 애플리케이션을 개발하기 위한 프레임워크입니다.



MEMORY

MODEL I/O

Data Connection

Agents

Callbacks

Chains

LangChain 퀵스타트

LangChain_QuickStart.ipynb

```python
from langchain_openai import ChatOpenAI

llm = ChatOpenAI()
```

Python

```python
from langchain_openai import OpenAIEmbeddings

embeddings = OpenAIEmbeddings()
```

Python

LangChain_QuickStart_Cohere .ipynb

```python
from langchain_community.chat_models import ChatCohere

# llm = ChatCohere(cohere_api_key="...")
llm = ChatCohere()
```

Python

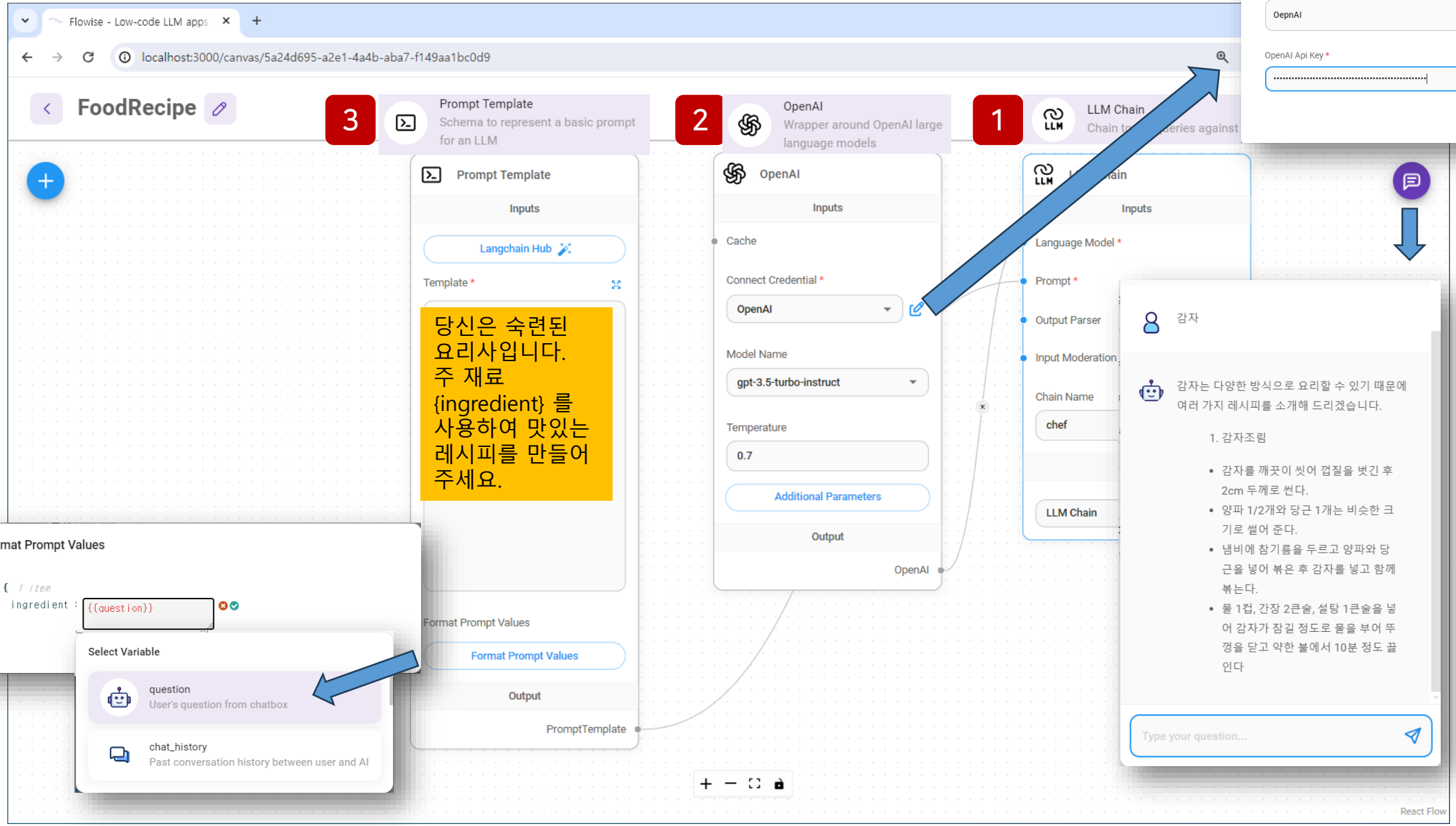
```python
from langchain_community.embeddings import CohereEmbeddings

embeddings = CohereEmbeddings()
```

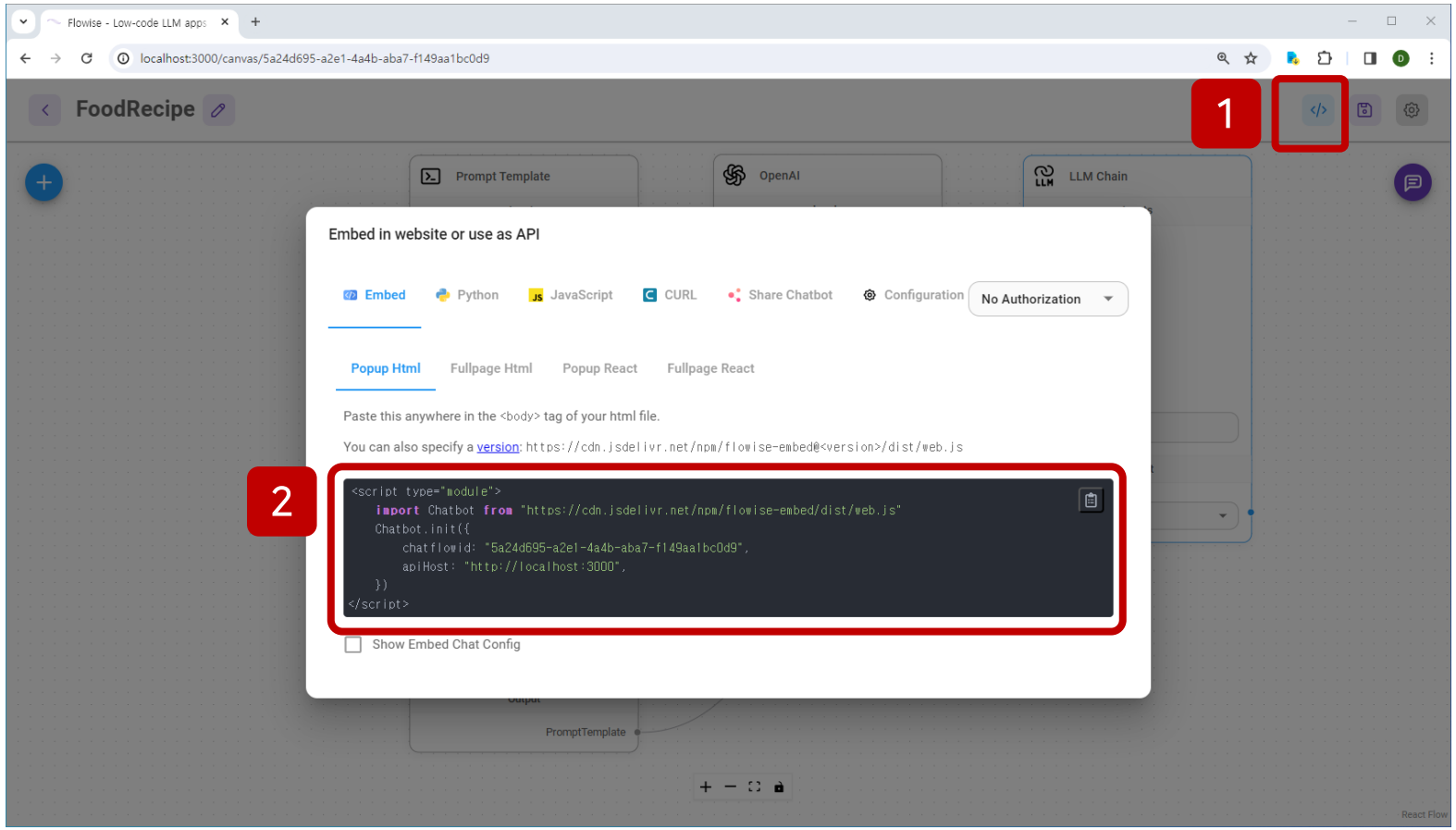
Python

Food Recipe App

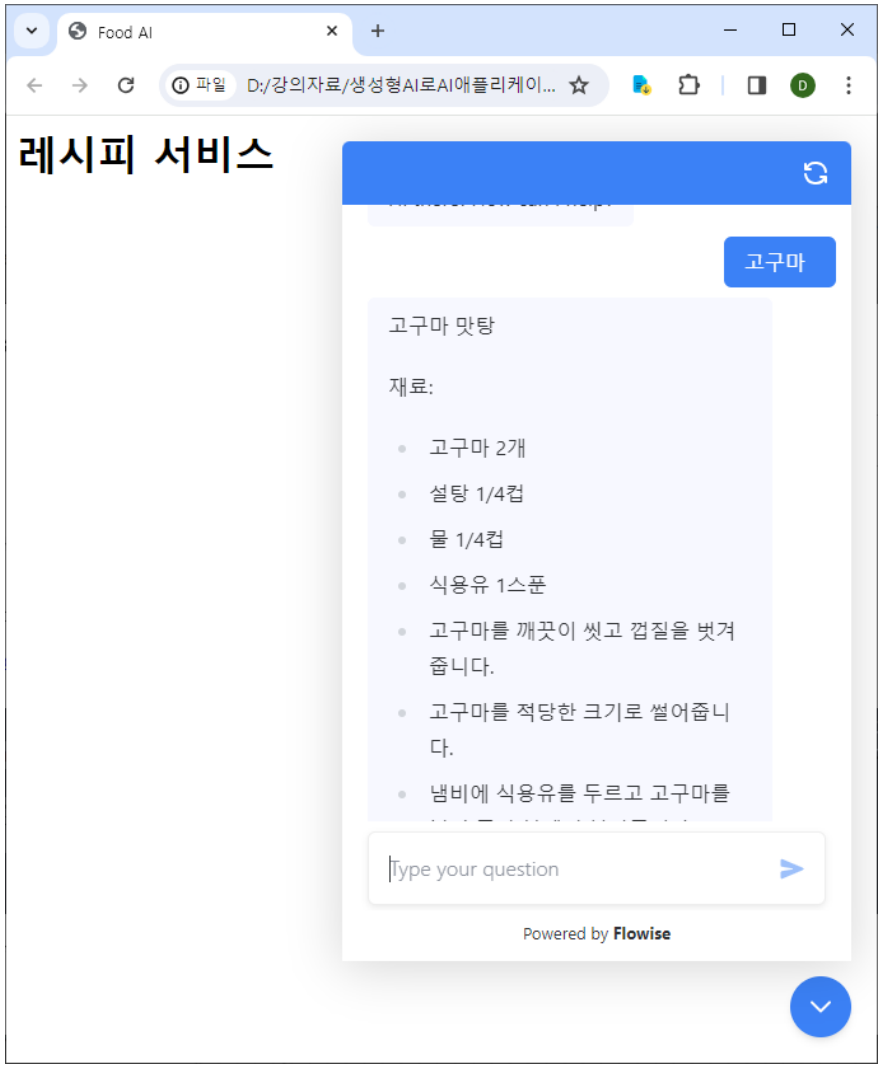
https://platform.openai.com/api-keys



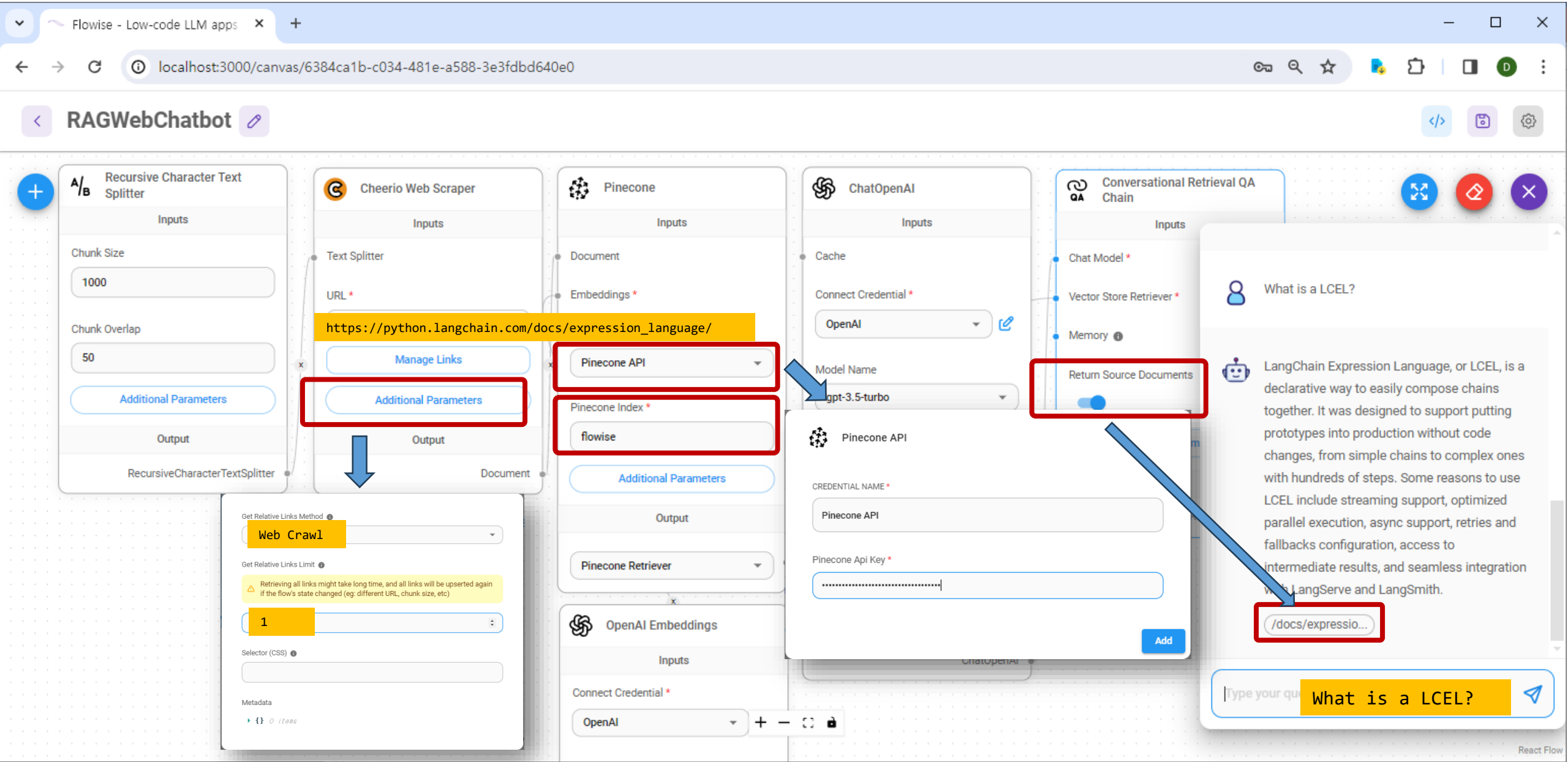
Food Recipe App



index.html



RAG Chatbot


Flowise - Low-code LLM apps
localhost:3000/canvas/6384ca1b-c034-481e-a588-3e3fdbd640e0
RAGWebChatbot
Recursive Character Text Splitter
Inputs
Chunk Size
1000
Chunk Overlap
50
Additional Parameters
Output
RecursiveCharacterTextSplitter
Cheerio Web Scraper
Inputs
Text Splitter
URL *
https://python.langchain.com/docs/expression_language/
Manage Links
Additional Parameters
Output
Document
Get Relative Links Method
Web Crawl
Get Relative Links Limit
Retrieving all links might take long time, and all links will be upserted again if the flow's state changed (eg: different URL, chunk size, etc)
1
Selector (CSS)
Metadata
Pinecone
Inputs
Document
Embeddings *
Pinecone API
Pinecone Index *
flowise
Additional Parameters
Output
Pinecone Retriever
OpenAI Embeddings
Inputs
Connect Credential *
OpenAI
ChatOpenAI
Inputs
Cache
Connect Credential *
OpenAI
Model Name
gpt-3.5-turbo
Pinecone API
CREDENTIAL NAME *
Pinecone API
Pinecone Api Key *
Add
Conversational Retrieval QA Chain
Inputs
Chat Model *
Vector Store Retriever *
Memory
Return Source Documents
What is a LCEL?
LangChain Expression Language, or LCEL, is a declarative way to easily compose chains together. It was designed to support putting prototypes into production without code changes, from simple chains to complex ones with hundreds of steps. Some reasons to use LCEL include streaming support, optimized parallel execution, async support, retries and fallbacks configuration, access to intermediate results, and seamless integration with LangServe and LangSmith.
/docs/expressio...
What is a LCEL?
React Flow

Flowise AI (2024) Tutorial

https://youtube.com/playlist?list=PL4HikwTaYE0H7wBxhvQqxYcKOkZ4O3zXh&si=Z8RHj9IIyijoBQDm

Build AI Apps WITHOUT Coding: Flowise Tutorial #1

Creating Chatflows & LLM Chains - FlowiseAI Tutorial #2

Combining Multiple Chains (Prompt Chaining) - FlowiseAI Tutorial #3

Output Parsers & IfElse Function - FlowiseAI Tutorial #4

Building Chatbots with Long-Term Memory - FlowiseAI Tutorial #5

Chatting With Your Own Data! Chat, Predict, & Analyze - FlowiseAI Tutoria...

Analysing Chatflows using LangSmith - FlowiseAI Tutorial #7