

# 1일차 과정 (01~05)

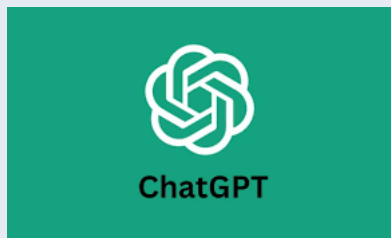


# 01. Gen AI

# 생성형 AI (Generative AI)

생성형 AI는 인공지능망을 이용하여 새로운 데이터를 생성해내는 기술로 프롬프트(Prompt)를 통해 사용자의 의도를 스스로 이해하고, 주어진 데이터로 학습, 활용하여 텍스트, 이미지, 오디오, 비디오 등 새로운 콘텐츠를 생성해내는 인공지능입니다.

ChatGPT



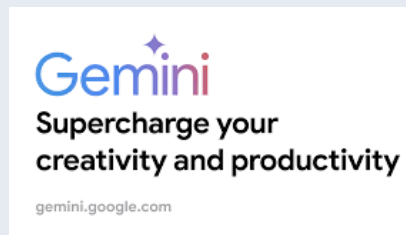
<https://chat.openai.com/>

Copilot



<https://copilot.microsoft.com/>

Gemini



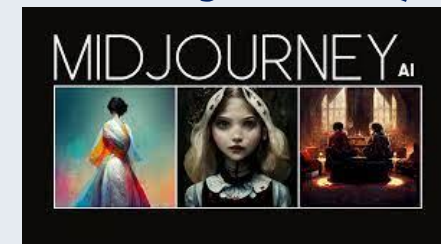
<https://gemini.google.com/>

Stable Diffusion



<https://stablediffusionweb.com/>

Midjourney



<https://www.midjourney.com/>

창의성

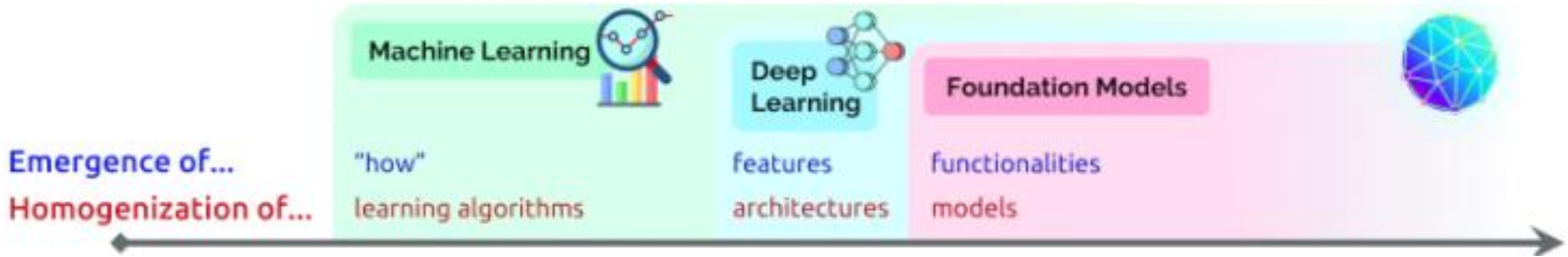
혁신성

생산성

인사이트

새로운 경험

# 인공지능(Artificial Intelligent)



**인공지능** 사람의 지적능력(추론, 인지)을 구현하고 모방하는 모든 기술

## 머신러닝

명시적인 프로그래밍 없이 학습하는 기술



선형회귀  
로지스틱회귀  
K-최근접 이웃  
결정트리  
랜덤포레스트  
서포트 벡터 머신  
클러스터링  
차원축소

## 딥러닝

인공신경망 이용해 데이터에서 패턴을 찾아내는 기술

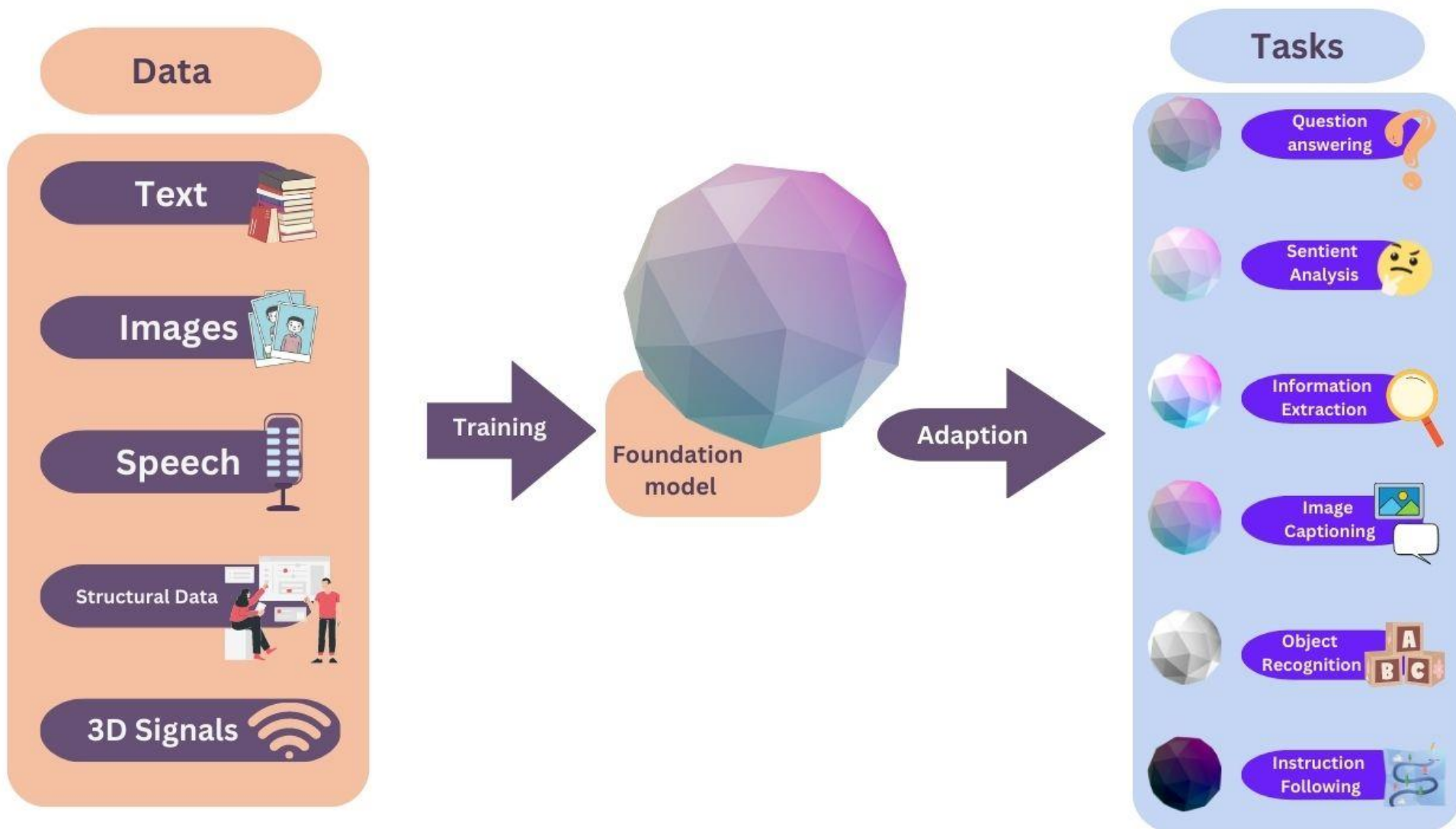


심층신경망(DNN)  
합성곱신경망(CNN)  
순환 신경망(RNN)  
생성적 적대 신경망(GAN)  
강화학습(RL)

## 파운데이션 모델

BERT  
GPT

# 파운데이션 모델 (FM, Foundation Model)



- 대용량의 폭넓은 비정형 데이터로 사전 훈련
- 복잡한 개념을 학습할 수 있는 방대한 파라미터
- 다양한 다운스트림 작업에 적용 가능
- 도메인별 데이터를 사용하여 파운데이션 모델을 사용자화

# LLM (Large Language Model)

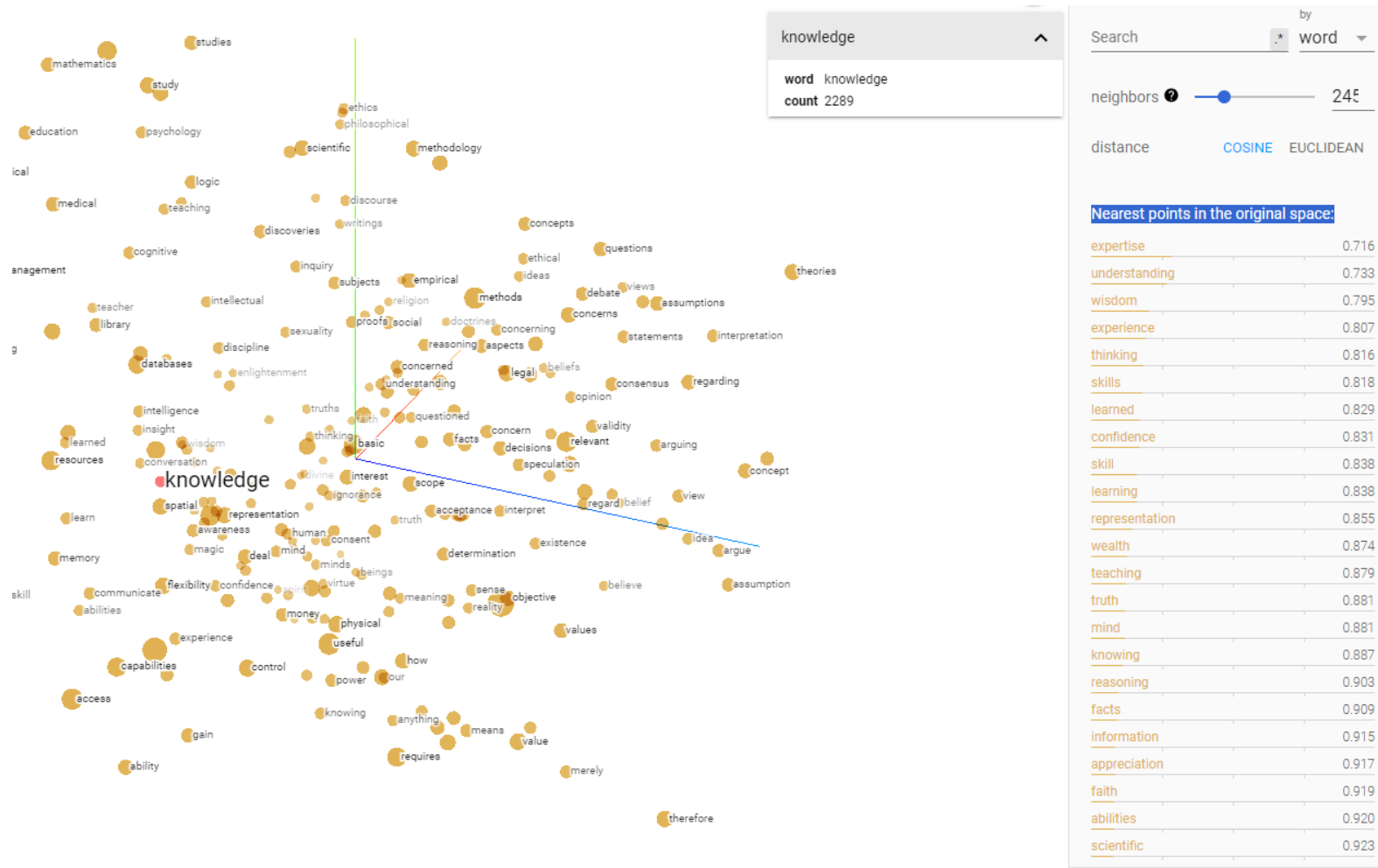
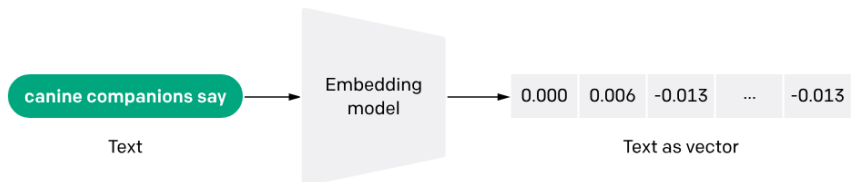
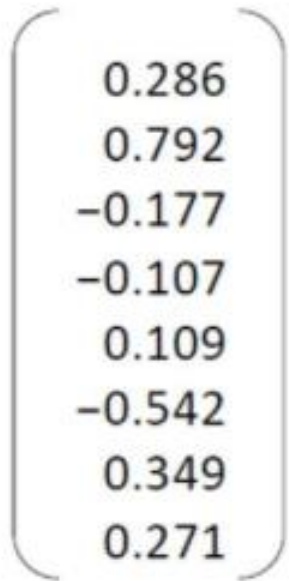
대규모 언어 모델은 텍스트와 다양한 콘텐츠를 인식하고, 요약, 번역, 예측, 생성 작업을 수행합니다





# 임베딩 (Embedding)

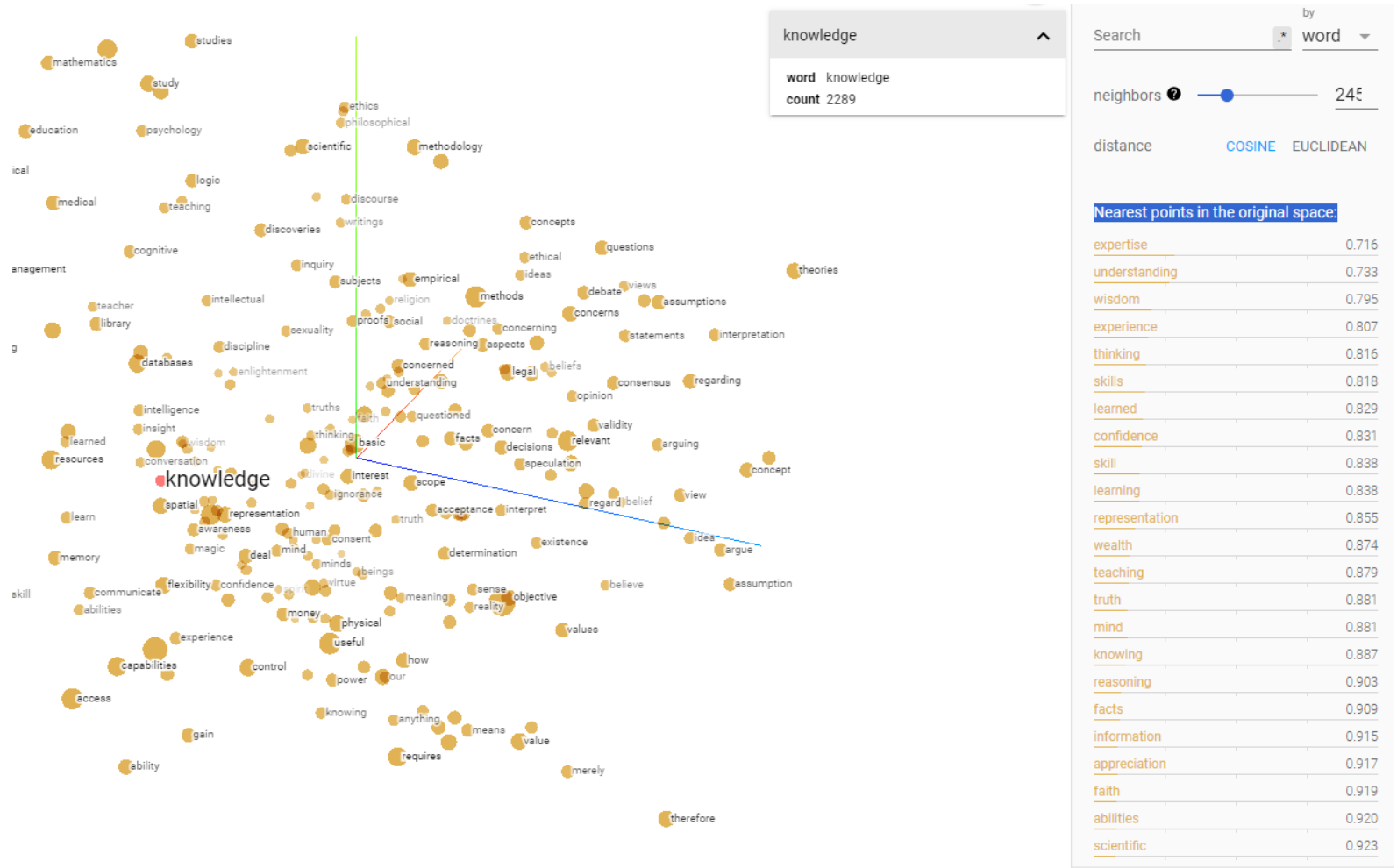
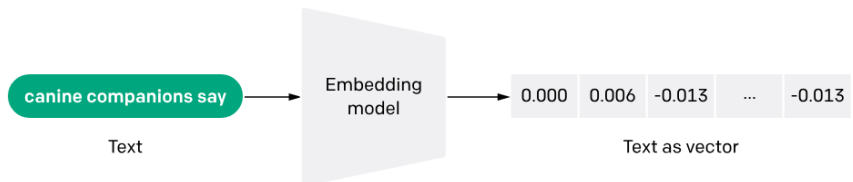
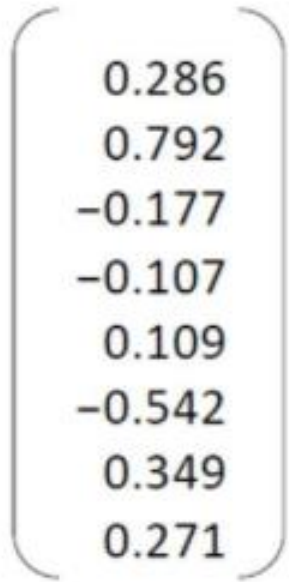
*banking* =



<https://projector.tensorflow.org/>

# 임베딩 (Embedding)

*banking* =



<https://projector.tensorflow.org/>





# Generative AI in a Nutshell

Henrik Kniberg  
Jan 2024

## Prompt Engineering/Design

**Bad prompt**  
The AI is an agent for a workshop.  
The AI is an agent for a workshop.  
The AI is an agent for a workshop.

**Good prompt**  
The AI is an agent for a workshop.  
The AI is an agent for a workshop.  
The AI is an agent for a workshop.

**Good prompt**  
The AI is an agent for a workshop.  
The AI is an agent for a workshop.  
The AI is an agent for a workshop.

## Autonomous Agents with tools



## Iterate!



## Side effect

Better communication skills overall  
Study, Practice, Learn

## Using vs Building AI-powered products



## The role of Humans

Is human role X still needed?

teacher, developer, lawyer, CEO, teacher, etc...  
Decide what to ask and how  
Evaluate results  
Compensate for AI weaknesses  
Provide context  
Legal compliance  
Data security  
etc, etc...



## The Age of AI



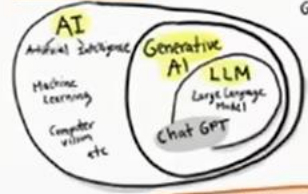
## Slow Revolutions



## Computers have gotten smarter



## Terminology



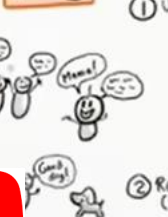
## Emergent Capabilities



## Einstein in your basement



## Training



## Biggest limitation is you

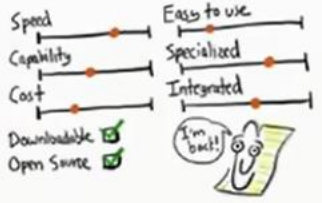
Imagination  
What can I do?  
How do I do it?

## How it works



Dogs are → animals  
Dogs are animals → that  
Dogs are animals that are known for... (bark)...

## Models, models everywhere



## Model Types

Text → Text  
Text → Image  
Image → Image  
Image → Text

Speech → Text  
Text → Audio  
Text → Video

Text → Text  
Text → Image  
Image → Image  
Image → Text

Speech → Text  
Text → Audio  
Text → Video

Text → Text  
Text → Image  
Image → Image  
Image → Text

Speech → Text  
Text → Audio  
Text → Video

Text → Text  
Text → Image  
Image → Image  
Image → Text

Speech → Text  
Text → Audio  
Text → Video

Text → Text  
Text → Image  
Image → Image  
Image → Text

Speech → Text  
Text → Audio  
Text → Video

Text → Text  
Text → Image  
Image → Image  
Image → Text

Speech → Text  
Text → Audio  
Text → Video

Text → Text  
Text → Image  
Image → Image  
Image → Text

Speech → Text  
Text → Audio  
Text → Video

Text → Text  
Text → Image  
Image → Image  
Image → Text

## Multimodal Models



## Taking AI for a walk



## 02. 개발환경

# 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.

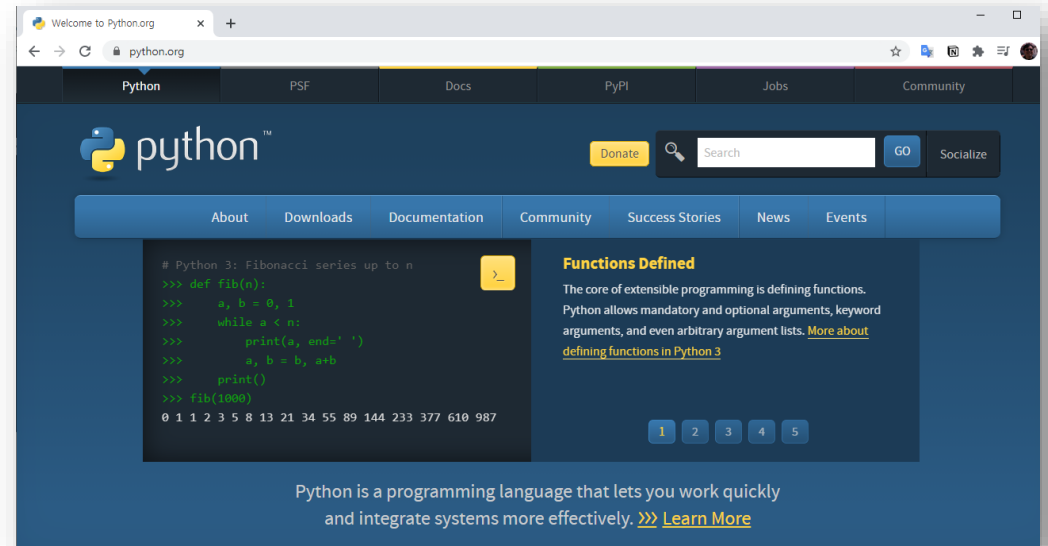
- Download [Windows embeddable package \(32-bit\)](#)
- Download [Windows embeddable package \(64-bit\)](#)
- Download [Windows embeddable package \(ARM64\)](#)
- Download [Windows installer \(32-bit\)](#)
- Download [Windows installer \(64-bit\)](#)
- Download [Windows installer \(ARM64\)](#)

<https://www.python.org/downloads/macros/>

### Stable Releases

- [Python 3.11.8 - Feb. 6, 2024](#)

- Download [macOS 64-bit universal2 installer](#)



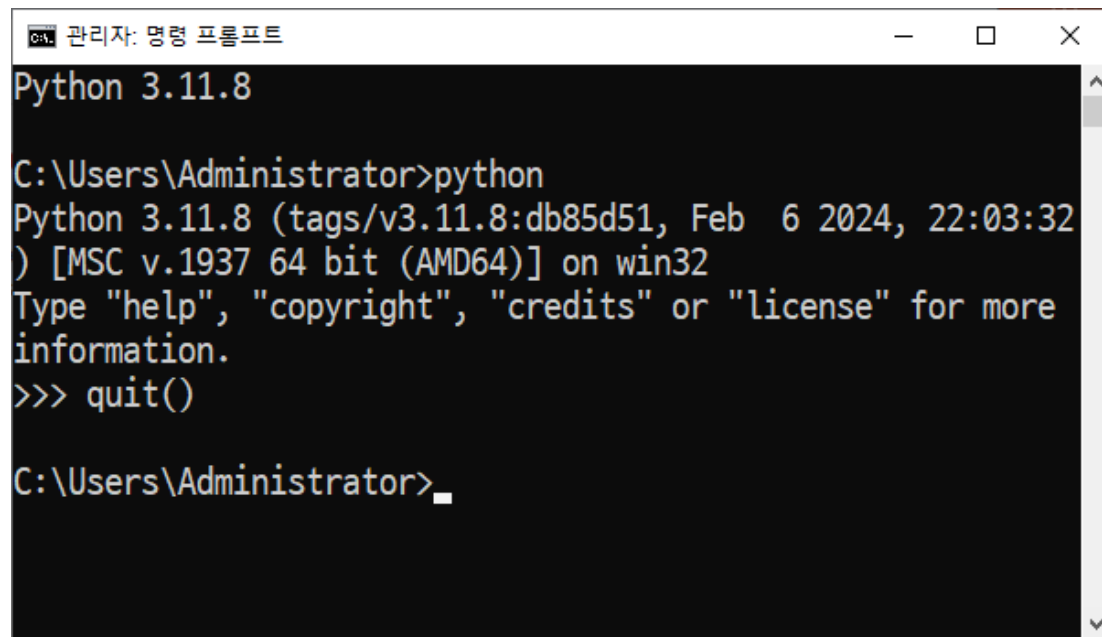
# Python 설치

## ■ 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

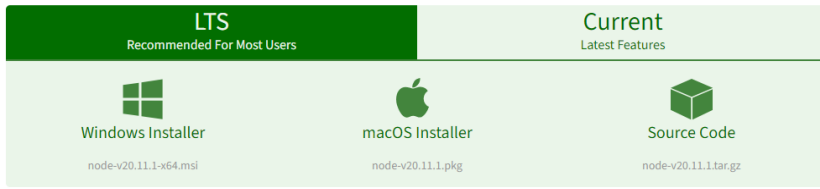
프로젝트별로 독립된 파이썬 실행 환경을 사용할 수 있는 가상 환경(Virtual Environment) 구성을 권장합니다.

- 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`

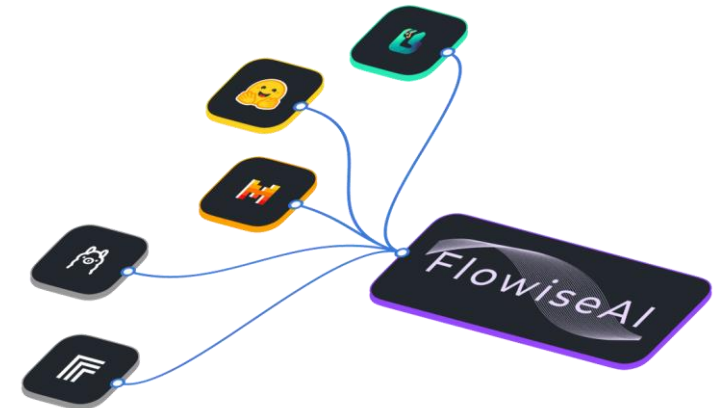
### 3. 모듈 설치

`cd Flowise`  
`yarn install`  
`yarn build`

### 4. App 실행

`yarn start`

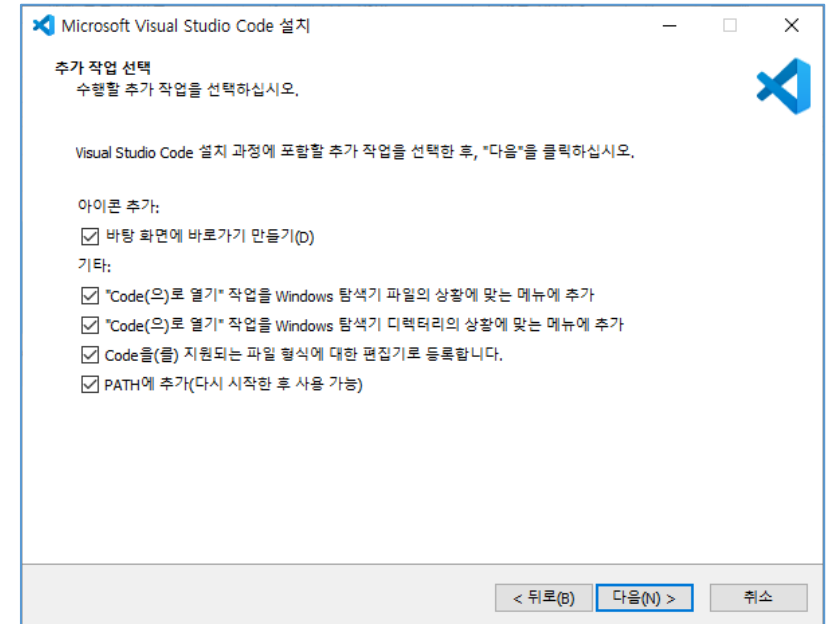
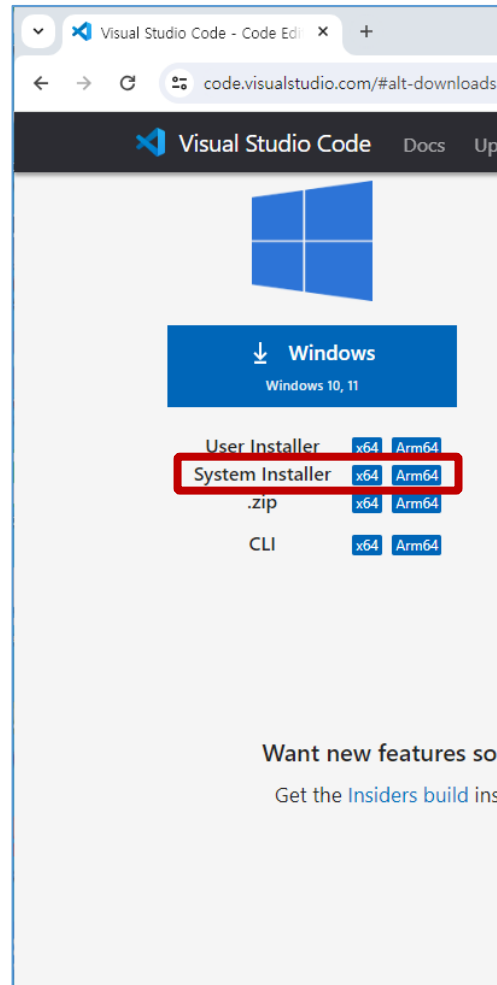
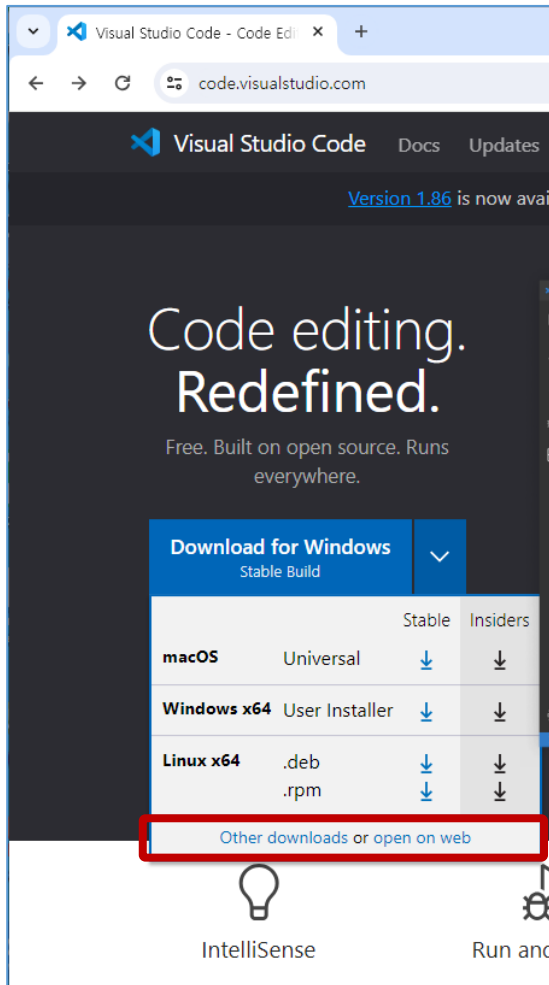
### 6. <http://localhost:3000> 접속



# VS Code 설치 – Windows

## ■ 설치 프로그램

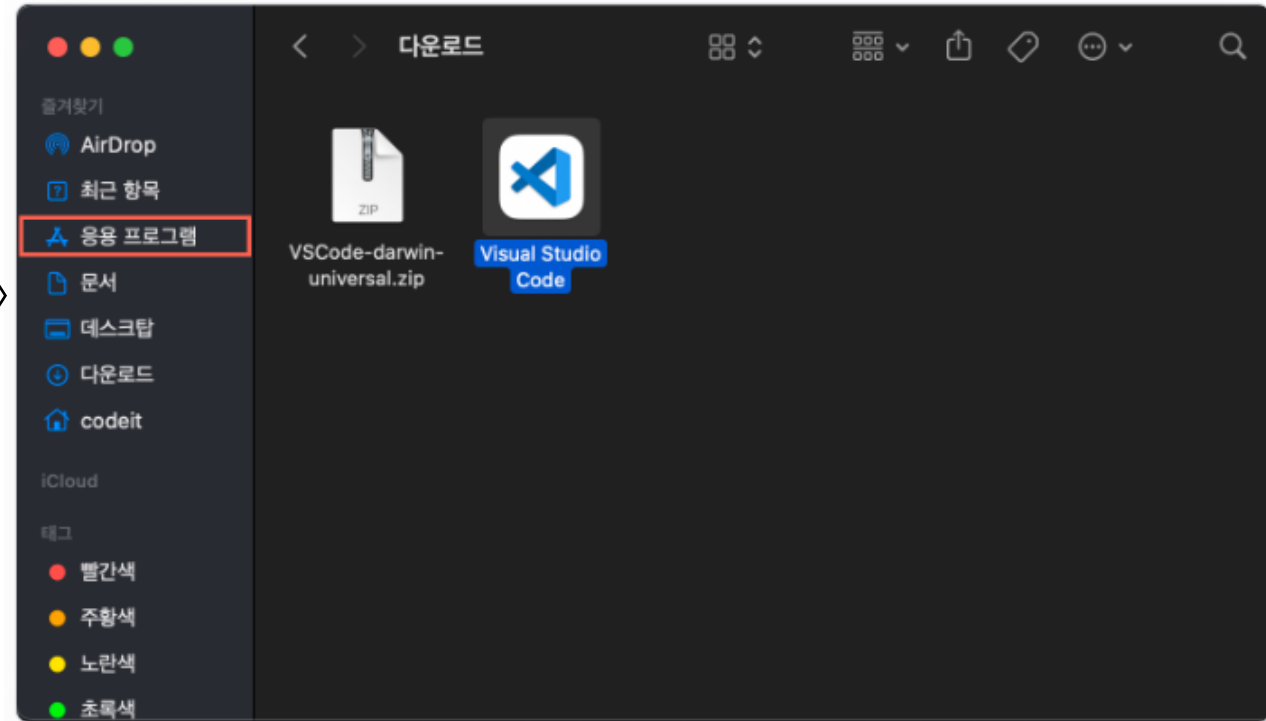
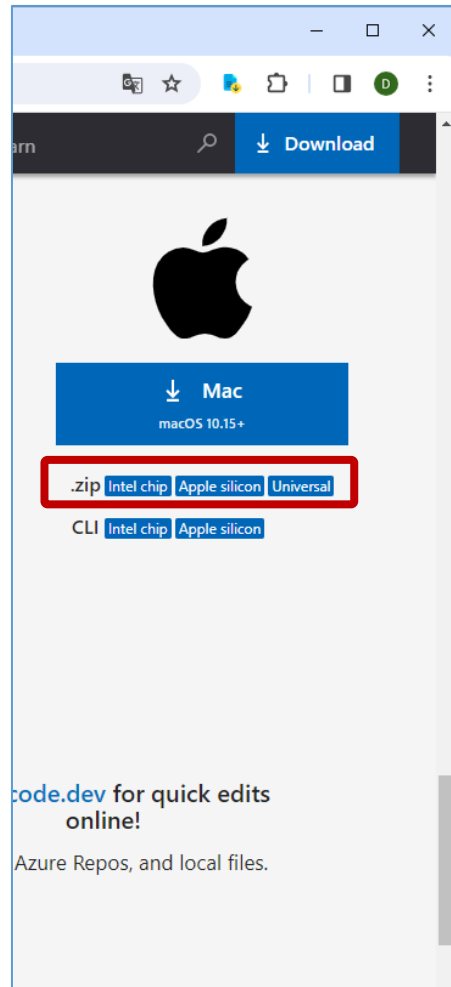
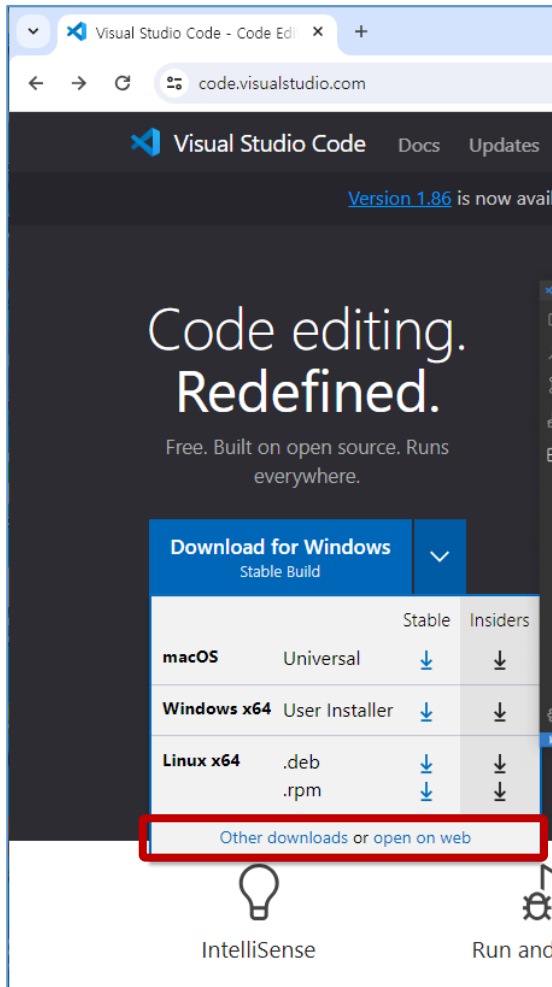
<https://code.visualstudio.com/>



# VS Code 설치 – macOS

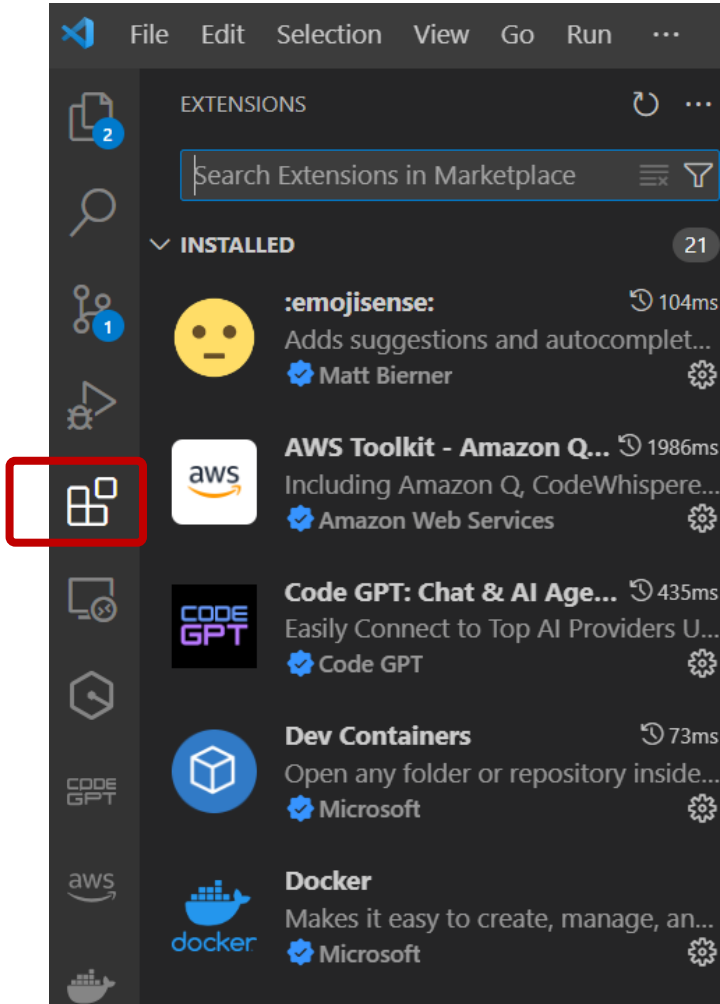
## ■ 설치 프로그램

<https://code.visualstudio.com/>



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

# VS Code Extension 설치

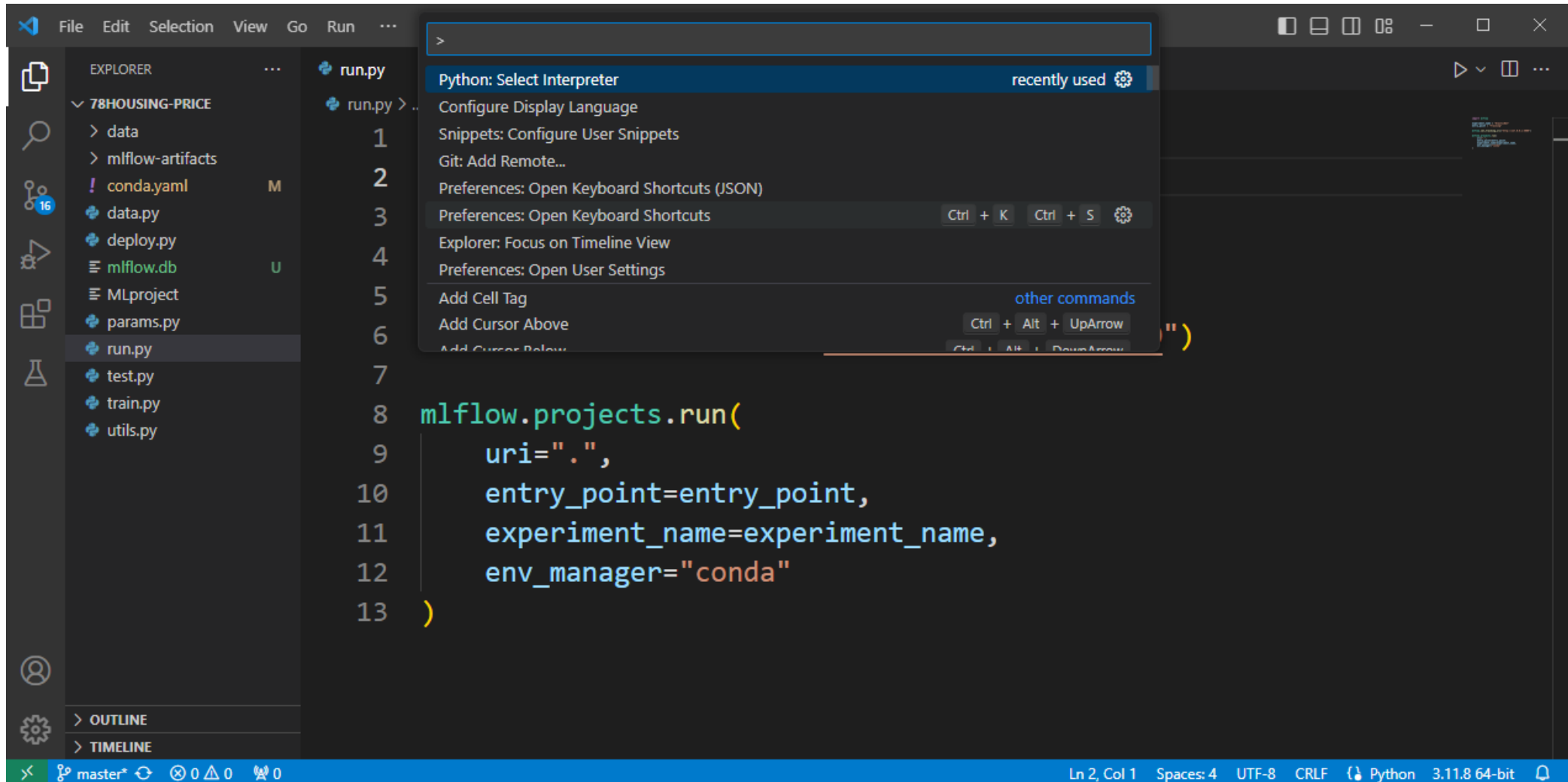


- Python : 파이썬에 대한 풍부한 지원 제공, IntelliSense(Pylance), 린팅, 디버깅, 코드 탐색 등의 기능을 제공
- Jupyter : Jupyter 노트북 지원
- Black Formatter : Python 파일에 대한 포매팅 지원 제공
- vscode-icons : Visual Studio Code용 아이콘
- TODO Highlight : 코드 내에서 TODO, FIXME 및 기타 주석을 강조 표시
- Todo Tree : TODO, FIXME와 같은 주석 태그를 빠르게 검색하고  
활동 표시줄의 트리 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



## 03. LLM API 사용



# OpenAI API 유료사용

<https://platform.openai.com/account/billing/overview>

The image shows a screenshot of the OpenAI API Billing overview page. The page is titled "Billing settings" and has tabs for "Overview", "Payment methods", "Billing history", and "Preferences". The "Overview" tab is selected, showing a "Free trial" status with "Credit remaining \$3.77". A red box highlights the "Add payment details" button, with a blue arrow pointing to a modal window. The modal is titled "Add payment details" and contains a form for adding credit card information. The form includes fields for "Card information" (with a dropdown for "카드 번호" and "MM / YY CVC"), "Name on card", "Billing address" (with fields for "Country", "Address line 1", "Address line 2", "City", "Postal code", and "State, county, province, or region"), and "Add payment details" buttons. A 5-dollar bill is overlaid on the right side of the modal. The background page also shows a sidebar with links to "Playground", "Assistants", "Fine-tuning", "API keys", "Storage", "Usage", "Settings", "Organization", "Team", "Limits", "Billing" (highlighted with a red box), "Profile", "Documentation", and "Help". At the bottom, there are four cards: "Payment methods" (Add or change payment method), "Billing history" (View past and current invoices), "Preferences" (Manage billing information), and "Usage limits" (Set monthly spend limits).

**Billing settings**

Overview Payment methods Billing history Preferences

**Free trial**

Credit remaining ⓘ

**\$3.77**

**Add payment details**

Note: This does not reflect the status of your account.

**What best describes you?**

☒ **Individual**  
I'm an individual

☐ **Company**  
I'm working on behalf of a company

**Add payment details**

Add your credit card details below. This card will be saved to your account and can be removed at any time.

**Card information**

카드 번호 MM / YY CVC

**Name on card**

**Billing address**

Country

Address line 1

Address line 2

City Postal code

State, county, province, or region

Cancel Continue

# OpenAI API Key 생성

명령 프롬프트 에서 아래 명령어 실행  
setx OPENAI\_API\_KEY "sk-kcXMU...SN5rS"

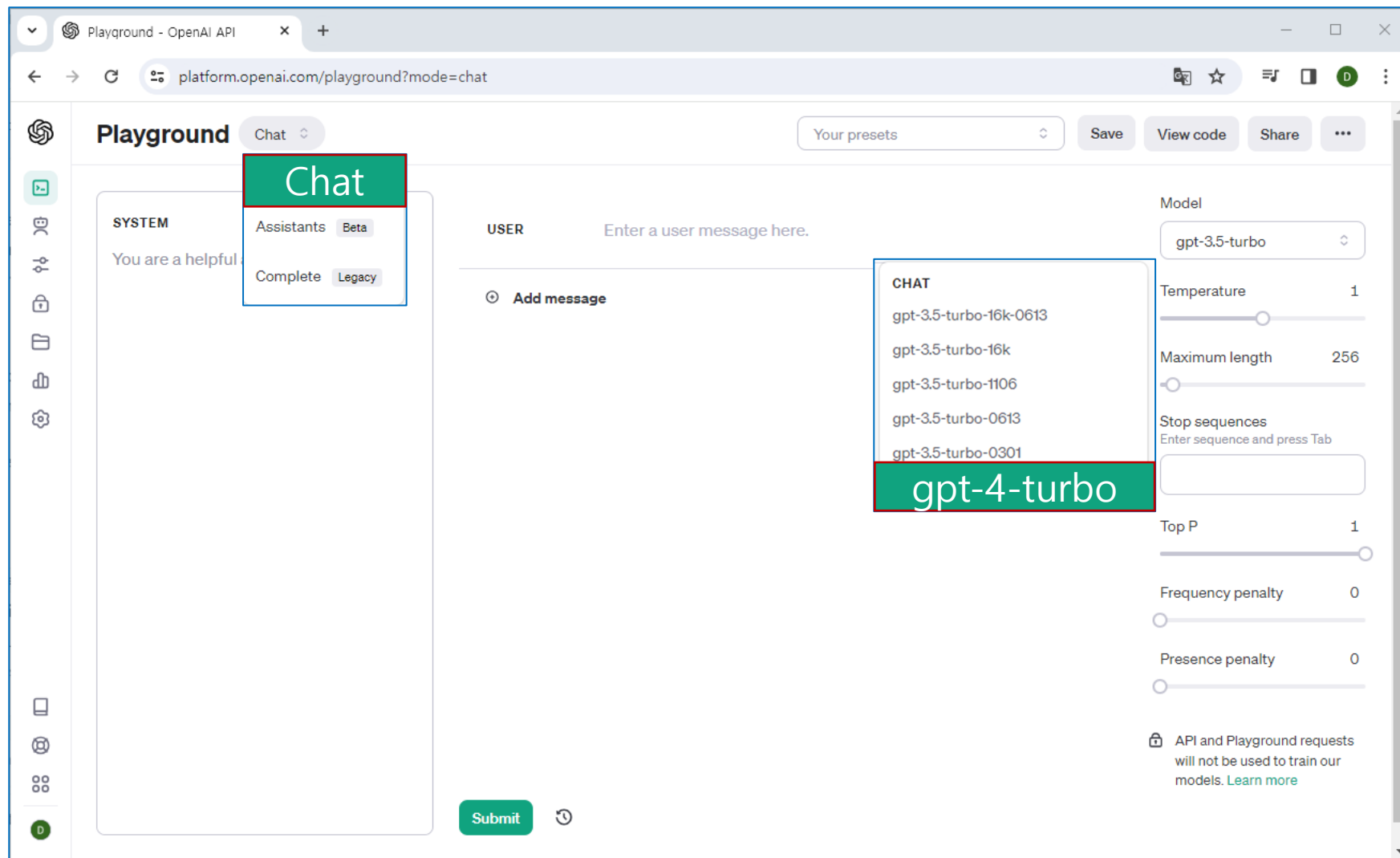
The image shows a sequence of steps to create an OpenAI API key, overlaid on a browser window showing the OpenAI platform. The steps are numbered 1 through 5:

1. Click the OpenAI logo in the top left corner of the platform.
2. Click the 'API keys' link in the left sidebar menu.
3. Click the '+ Create new secret key' button at the bottom of the 'API keys' page.
4. In the 'Create new secret key' dialog, enter a name (e.g., 'Test Key') in the 'Name' field.
5. In the 'Save your key' dialog, click the 'Copy' button to copy the generated API key.

The background browser window shows the 'API keys' page with a table listing existing keys. One key is visible with the name 'OpenAIAPIKey' and permissions set to 'All'.

# 플레이그라운드

<https://platform.openai.com/playground>



- Temperature : 값이 낮을수록 가장 높은 확률의 다음 토큰을 선택하고, 높아지면 무작위성이 높아짐
- Max Length : 모델이 생성하는 토큰 최대 길이
- Stop Sequences : 모델의 토큰 생성을 중지하는 문자열
- Top P : 값이 높으면 모델이 가능성이 낮은 단어를 포함하여 더 다양한 출력을 얻을 수 있음
- Frequency Penalty : 해당 토큰이 나타난 횟수에 비례하여 페널티 적용
- Presence Penalty : 모든 반복 토큰에 동일한 페널티 적용(2번 나타나는 토큰과 10번 나타나는 토큰 모두 동일한 페널티)

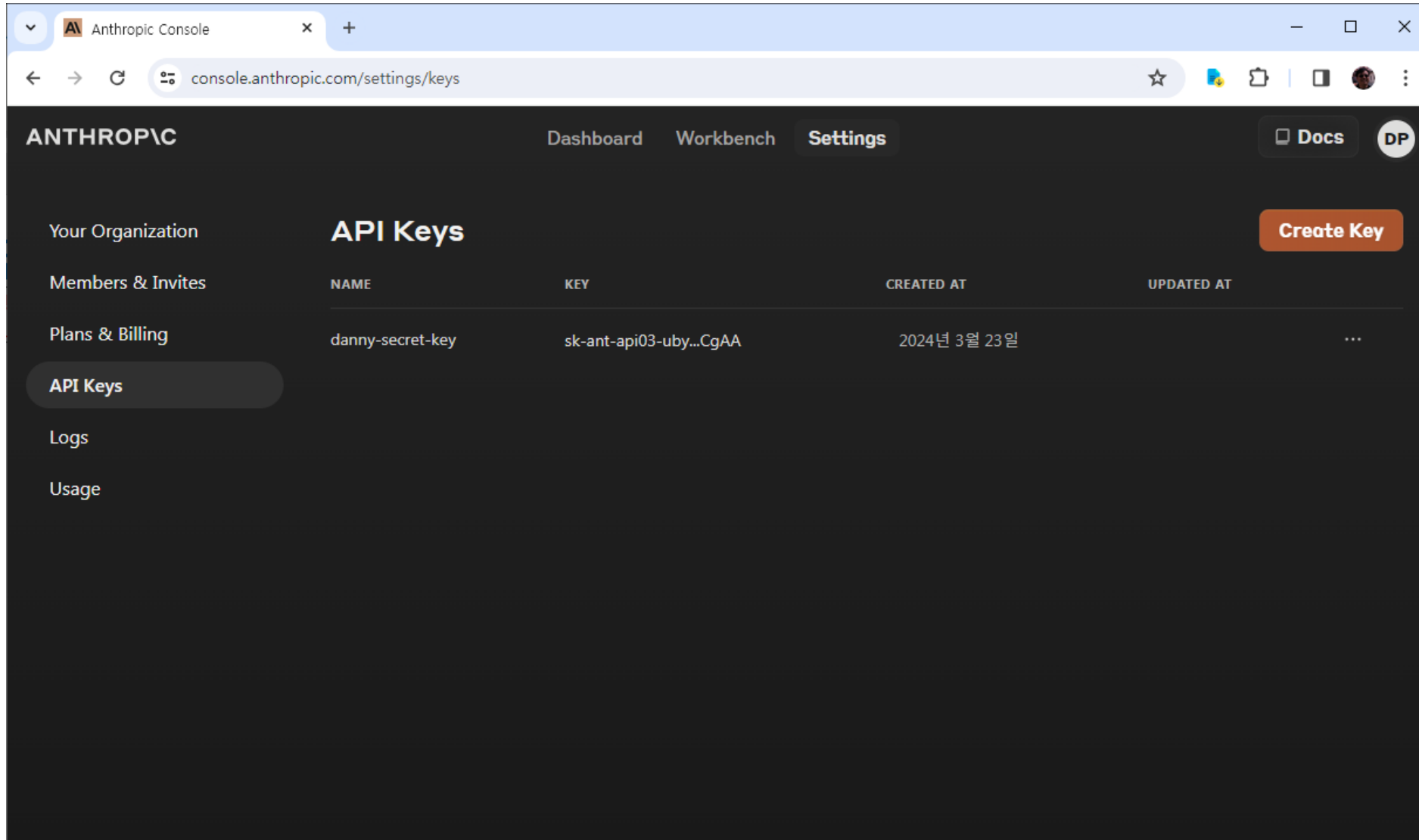
※ Temperature 와 Top\_p, 그리고 Frequency Penalty와 Presence Penalty 동시 변경은 비권장함

# 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"

The screenshot shows the Cohere dashboard's 'API Keys' section. The left sidebar contains navigation links for PLATFORM (Overview, Fine-tuning, API Keys, Connectors BETA) and SETTINGS (Team, Billing & Usage, Data retention, Profile). The main content area has a header 'cohere dashboard' and navigation links 'CORAL', 'DASHBOARD', 'PLAYGROUND', 'DOCS', 'COMMUNITY'. A message states 'You don't have any Production keys'. Below this, the 'Trial keys' section is highlighted with a red border. It includes a 'FREE' tag, a '+ New Trial key' button, and explanatory text: 'Calls made using Trial keys are free of charge. Trial keys are rate-limited, and cannot be used for commercial purposes.' A table lists the trial keys:

NAME	KEY	CREATED	CREATOR	
default	.....7tXm	Mar 3, 2024	kgpark88@gmail.com	Rename ✎ Delete 🗑

At the bottom of the trial keys section is a '+ Create Trial key' button.

# 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)

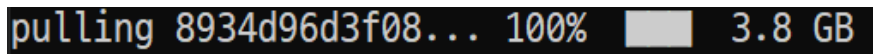


- sLLM(smaller Large Language Model , 소형 언어 모델)
  - LLM과 비교했을 때 매개변수의 수가 수십 억~수백 억개로 비교적 크기가 작은 언어모델
  - 비용절감, 보안, 특정 도메인에 활용 목적으로 사용
  - 특정 도메인 사용용도로 sLLM 을 사용하는 경우가 많아지고 있음



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

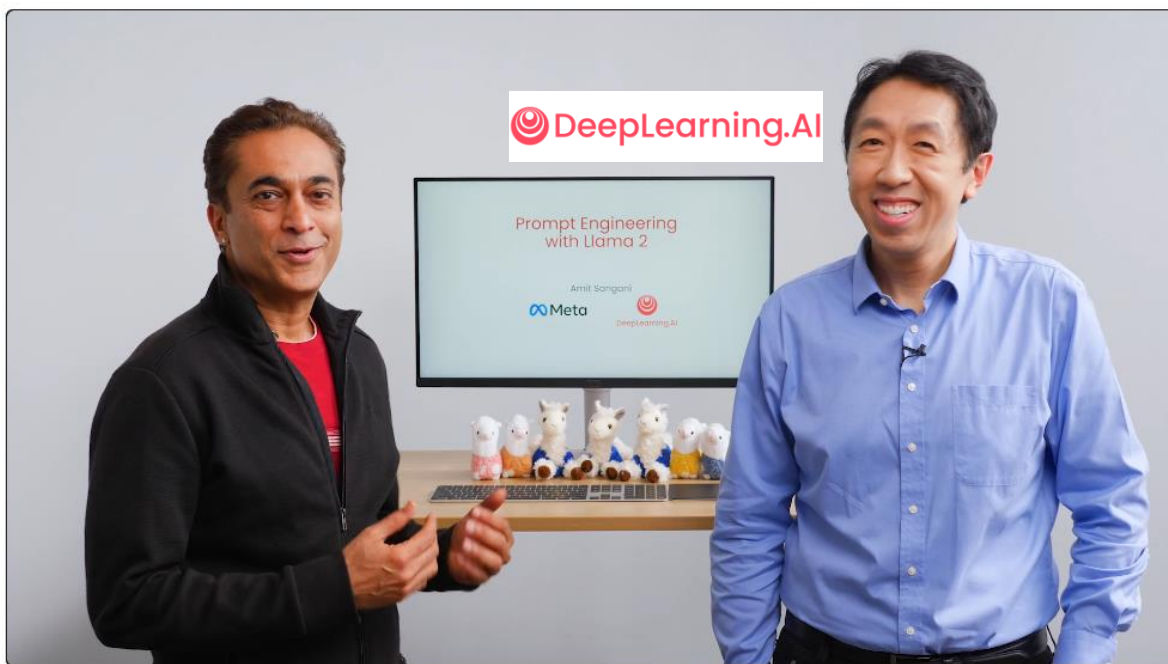


- Llama 2 (Large Language Model Meta AI)
  - 메타에서 공개한 상업적으로도 이용 가능한 오픈 소스 sLLM
  - 설치 및 실행 : `ollama run llama2` 
  - 프로그램 개발 예시

```
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

<https://www.together.ai/>

- 사이트 접속 및 회원가입
- 환경변수에  
TOGETHER\_API\_KEY 값 추가
- ./code/llama/utils.py 파일 참고 , 로컬 모델 사용 가능

명령 프롬프트 에서 아래 명령어 실행  
setx TOGETHER\_API\_KEY "995e07d0bb7f148ba7ef"

```
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]
```

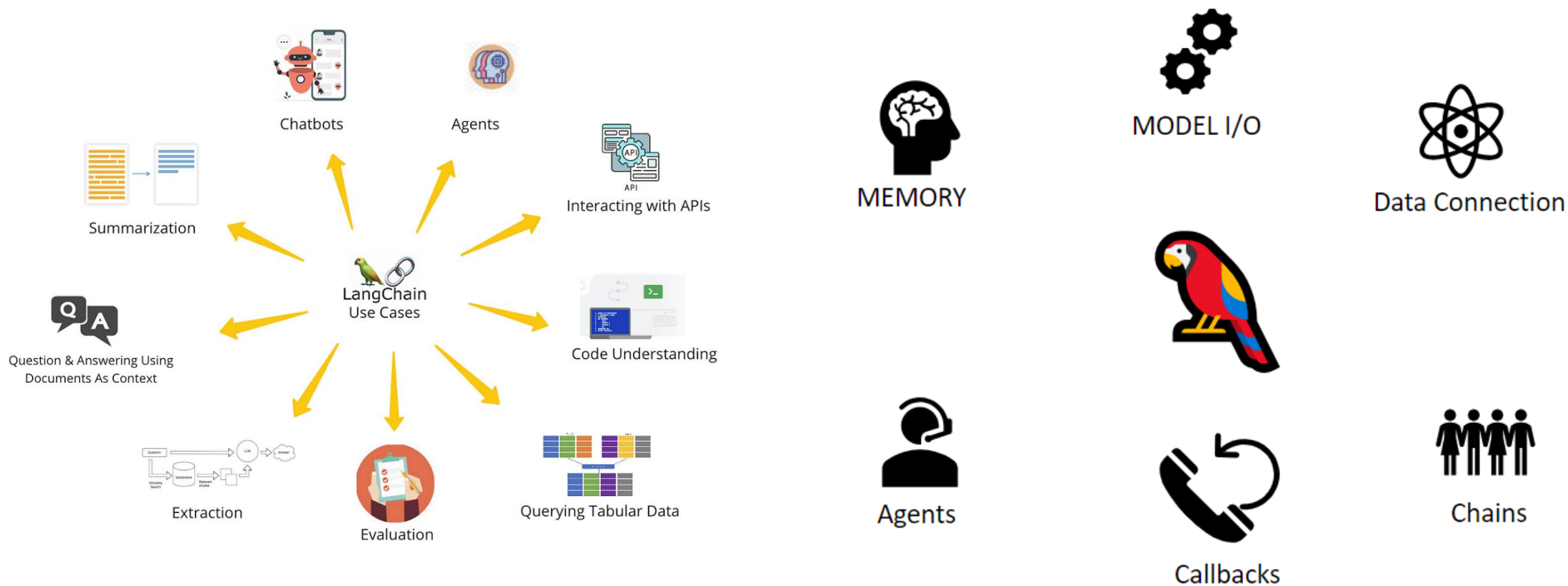
start tags

end tags

## 04. LangChain 프레임워크

# LangChain

LangChain은 언어 모델로 구동되는 애플리케이션을 개발하기 위한 프레임워크입니다.



# LangChain 퀵스타트

LangChain\_QuickStart.ipynb



```
관리자: C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.19045.4170]
(c) Microsoft Corporation. All rights reserved.

C:\genai-main\code>jupyter notebook_
```

LangChain\_QuickStart\_Cohere.ipynb



```
[3]: from langchain_openai import ChatOpenAI

llm = ChatOpenAI()
```

```
[13]: from langchain_openai import OpenAIEmbeddings

embeddings = OpenAIEmbeddings()
```

```
from langchain_community.chat_models import ChatCohere

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

```
from langchain_community.embeddings import CohereEmbeddings

embeddings = CohereEmbeddings()
```

## 05. 노코딩 SI앱 빌드



# Flowise 설치

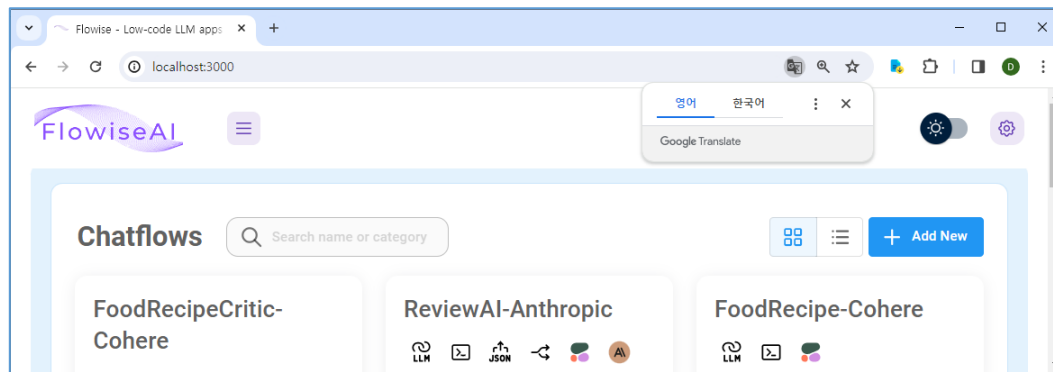
NodeJS 설치 : <https://nodejs.org/en/download>  
Flowise 설치 : npm install -g flowise

## ■ Flowise 시작 npx flowise start

```
관리자: C:\Windows\system32\cmd.exe
Microsoft Windows [Version 10.0.19045.4170]
(c) Microsoft Corporation. All rights reserved.

C:\genai-main>npx flowise start
2024-03-27 08:18:26 [INFO]: Starting Flowise...
2024-03-27 08:18:26 [INFO]: [server]: Flowise Server is listening at 3000
```

## ■ Flowise 사용 <http://localhost:3000> 접속



## ■ Flowise AI (2024) Tutorial

<https://youtube.com/playlist?list=PL4HikwTaYE0H7wBxhvQqxYcKOkZ4O3zXh&si=Z8RHj9llyijoBQDm>



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 Tutorial #6



Analysing Chatflows using LangSmith -  
FlowiseAI Tutorial #7

<https://github.com/FlowiseAI/Flowise>

# Food Recipe App

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

Flowise - Low-code LLM apps

localhost:3000/canvas/5a24d695-a2e1-4a4b-aba7-f149aa1bc0d9

## FoodRecipe

**3** Prompt Template  
Schema to represent a basic prompt for an LLM

Inputs: Langchain Hub

Template \*

당신은 숙련된 요리사입니다. 주 재료 {ingredient} 를 사용하여 맛있는 레시피를 만들어 주세요.

Format Prompt Values

```
{ 1 item
  ingredient: {{question}}
```

Select Variable

- question: User's question from chatbox
- chat\_history: Past conversation history between user and AI

**2** OpenAI  
Wrapper around OpenAI large language models

Inputs: Cache, Connect Credential \* (OpenAI), Model Name (gpt-3.5-turbo-instruct), Temperature (0.7)

Additional Parameters

Output: OpenAI

**1** LLM Chain  
Chain to run a series of LLMs against a prompt

Inputs: Language Model \*

Prompt \*

Output Parser

Input Moderation

Chain Name: chef

LLM Chain

OpenAI API

CREDENTIAL NAME \*  
OepnAI

OpenAI Api Key \*

Add

감자

감자는 다양한 방식으로 요리할 수 있기 때문에 여러 가지 레시피를 소개해 드리겠습니다.

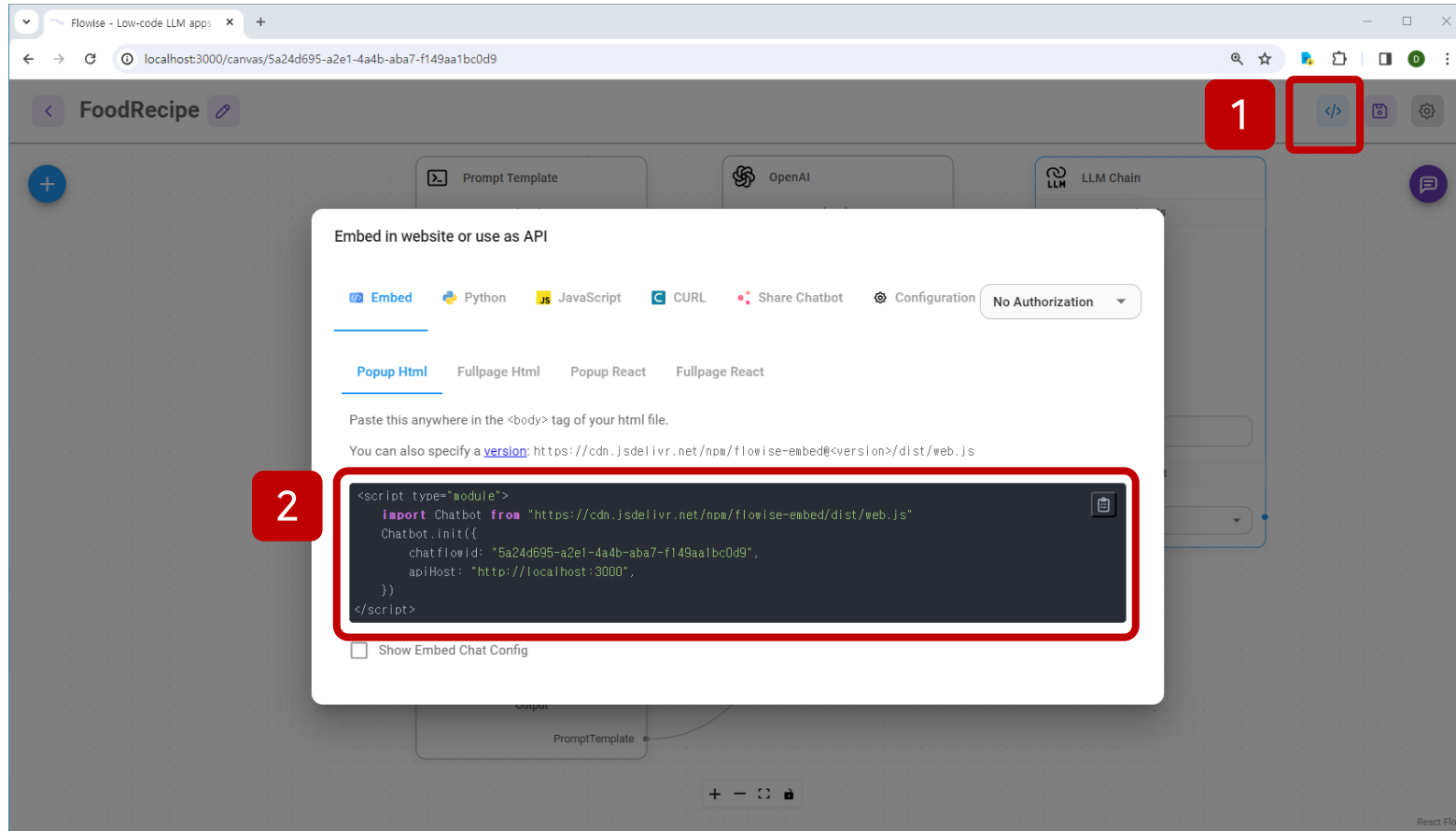
1. 감자조림

- 감자를 깨끗이 씻어 껍질을 벗긴 후 2cm 두께로 썬다.
- 양파 1/2개와 당근 1개는 비슷한 크기로 썰어 준다.
- 냄비에 참기름을 두르고 양파와 당근을 넣어 볶은 후 감자를 넣고 함께 볶는다.
- 물 1컵, 간장 2큰술, 설탕 1큰술을 넣어 감자가 잠길 정도로 물을 부어 뚜껑을 닫고 약한 불에서 10분 정도 끓인다

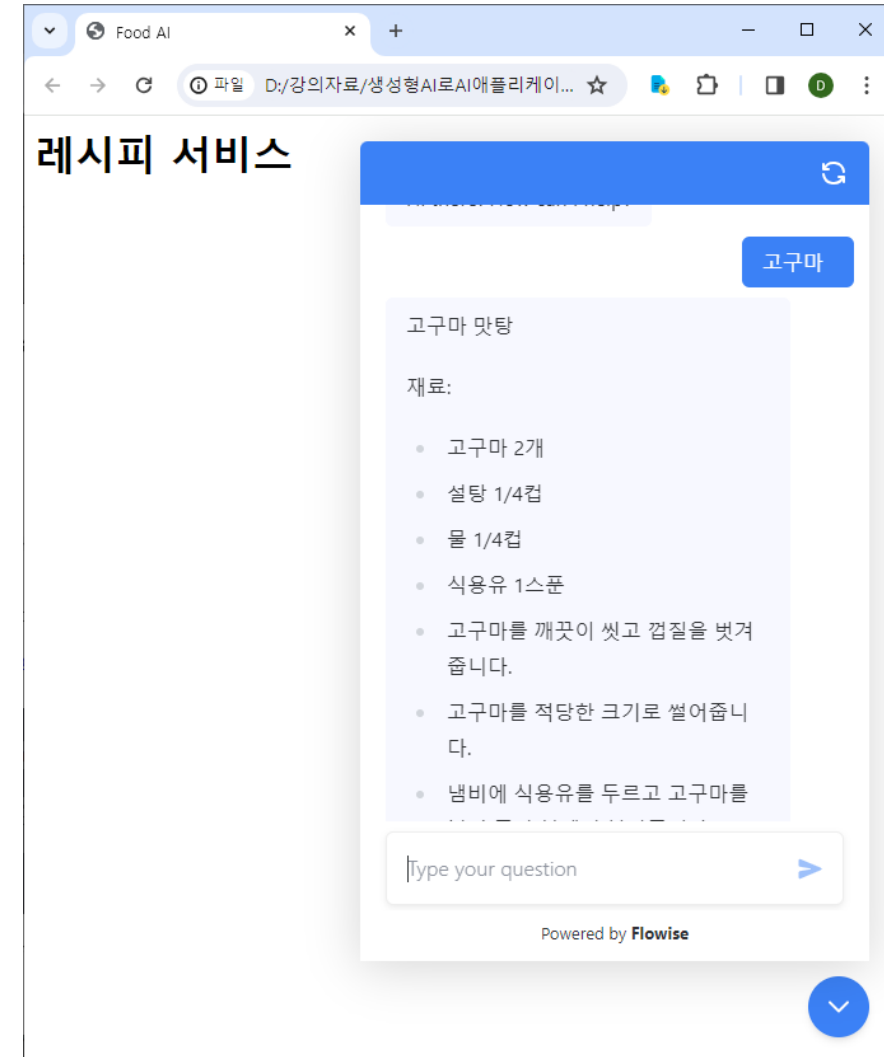
Type your question...

React Flow

# 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/](https://python.langchain.com/docs/expression_language/)

Manage Links

Additional Parameters

Output: Document

**Pinecone**

Inputs

Document

Embeddings: Pinecone API

Pinecone Index: flowise

Additional Parameters

Output: Pinecone Retriever

**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/expresio...

Type your question: What is a LCEL?



Thank you 😊