

# ChatGPT API와 함께 My WebApp 만들기

7/6/2023

Sejin Chun

# Contents

- OpenAI API 사용하기
- AI 서비스 라이브러리
- OpenAI API 활용한 AI 서비스
- Streamlit을 통해 앱을 온라인에 배포
  - Question
  - Chat
  - AI Image


# Generative AI

+ New chat


Today

부산 최고관광지 추천

Model: GPT-4






부산의 최고관광지를 추천해줘




부산은 한국의 가장 아름다운 도시 중 하나로 알려져 있으며, 다양한 관광 명소가 있습니다. 몇 가지 추천해드릴만한 곳들은 다음과 같습니다:

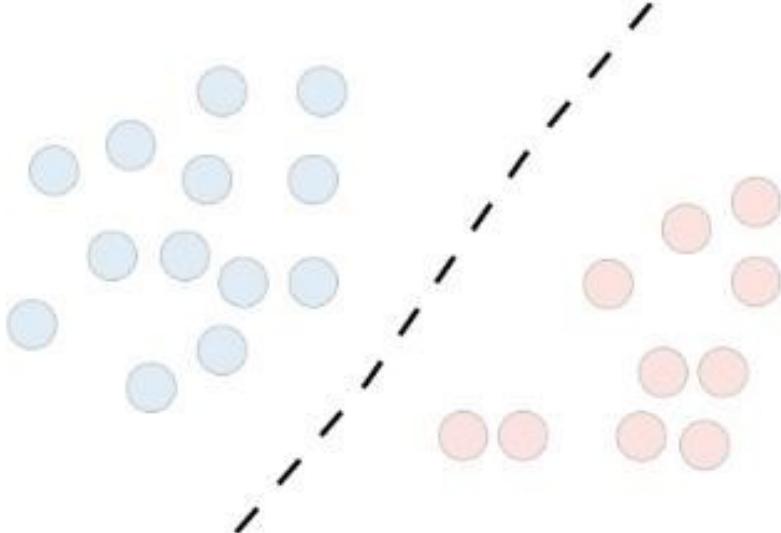
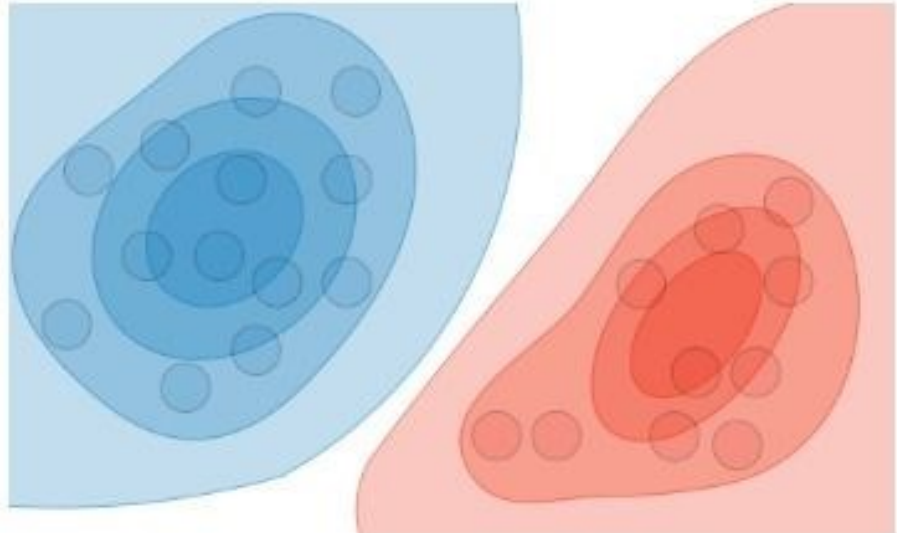
- 해운대 해수욕장: 부산에서 가장 유명한 해수욕장으로, 특히 여름에는 방문객들로 붐비게 됩니다. 흰 모래사장과 맑은 물이 특징이며, 주변에는 많은 레스토랑, 바, 호텔 등이 있어 편의시설도 잘 갖추어져 있습니다.
- 광안리 해수욕장: 광안대교와 함께 이곳의 뷰는 꼭 찾아봐야 할 것입니다. 해질 무렵의 광안대교 야경은 특히 더욱 아름답습니다.
- 부산 국제 시장: 1950년대부터 이어져 내려온 이 시장은 다양한 먹거리와 상품을 구매할 수 있으며, 한국의 전통 시장 분위기를 느낄 수 있습니다.
- 감천문화마을: 이 곳은 예술가들이 집들의 외벽을 다양한 그림으로 장식하여 '마천루 미술관'이라고 불리게 된 곳입니다. 이곳에서는 골목길을 돌아다니며 감상하거나, 카페에서 휴식을 취하는 등의 여유로운 시간을 보낼 수 있습니다.
- 부산 타워와 용두산 공원: 부산 타워에서는 부산의 전경을 한 눈에 볼 수 있습니다. 주변의 용두산 공원도 산책하기 좋은 곳입니다.
- 해동 용궁사: 이곳은 바다 위에 위치한 독특하고 아름다운 절로, 동해 바다를 배경으로 한 경치가 매우 인상적입니다.

이 외에도 물론 부산에는 더 많은 관광 명소들이 있습니다. 계절, 관심사 등에 따라 여러 곳을 방문해 보시는 것을 추천드립니다.

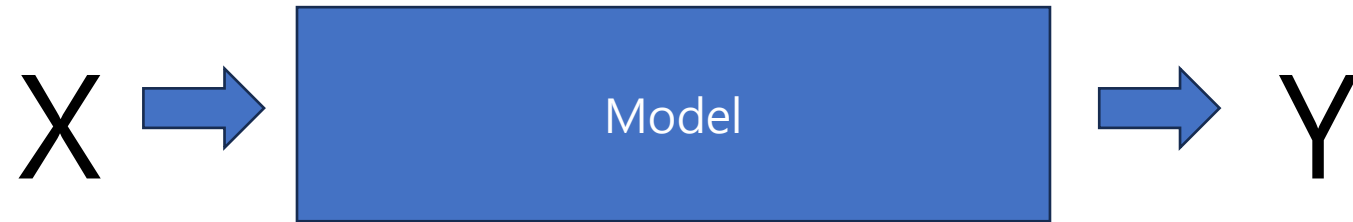




송정 해수욕장은 어때? 해운대 해수욕장보다 좋은가요?

	Discriminative model	Generative model
<b>Goal</b>	Directly estimate $P(y x)$	Estimate $P(x y)$ to then deduce $P(y x)$
<b>What's learned</b>	Decision boundary	Probability distributions of the data
<b>Illustration</b>	 <p>A scatter plot illustrating a discriminative model. It shows two classes of data points: blue circles on the left and red circles on the right. A dashed black line represents the decision boundary separating the two classes.</p>	 <p>A scatter plot illustrating a generative model. It shows two classes of data points: blue circles on the left and red circles on the right. The data points are enclosed within shaded regions representing probability distributions. The blue region is on the left and the red region is on the right, with the shaded areas indicating the density of the data points.</p>
<b>Examples</b>	Regressions, SVMs	GDA, Naive Bayes

# Discriminative model



# Generative model

- text to text
- text to image
- text to audio
- image to text
- text, image to text



# Welcome to the OpenAI platform

## Start with the basics

### Quickstart tutorial

Learn by building a quick sample app

### Examples

Explore some example tasks

## Build an application



### GPT

Learn how to generate text and call functions



### GPT best practices

Learn best practices for building with GPT models



### Embeddings

Learn how to search, classify, and compare text



### Speech to text Beta

Learn how to turn audio into text



### Image generation Beta

Learn how to generate or edit images



### Fine-tuning

Learn how to train a model for your use case

## Build a ChatGPT plugin



### Introduction Alpha

Learn the basics of building a ChatGPT plugin



### Examples Alpha

Explore ChatGPT plugin examples

## Get started ×

Enter an instruction or select a preset, and watch the API respond with a **completion** that attempts to match the context or pattern you provided.

You can control which **model** completes your request by changing the model.

### KEEP IN MIND

- ✈ Use good judgment when sharing outputs, and attribute them to your name or company. [Learn more.](#)
- 📄 Requests submitted to our API and Playground will not be used to train or improve future models. [Learn more.](#)
- 📅 Our default models' training data cuts off in 2021, so they may not have knowledge of current events.

## Playground


English to other languages × ▾

Save

View code

Share

...

Translate this into 1. French, 2. Spanish and 3. Japanese: 

What rooms do you have available?

1.

Mode

Complete ▾

Model

text-davinci-003 ▾

Temperature 0.3

Maximum length 100

Stop sequences

Enter sequence and press Tab

Top P 1

Frequency penalty 0

Presence penalty 0

Best of 1

Inject start text



☒

Inject restart text

☒

Show probabilities

Off ▾

 Looking for ChatGPT? [Try it now](#) 

Submit





Get started ×

Enter an instruction or select a preset, and watch the API respond with a **completion** that attempts to match the context or pattern you provided.

You can control which **model** completes your request by changing the model.

KEEP IN MIND

- Use good judgment when sharing outputs, and attribute them to your name or company. [Learn more.](#)
- Requests submitted to our API and Playground will not be used to train or improve future models. [Learn more.](#)
- Our default models' training data cuts off in 2021, so they may not have knowledge of current events.

Playground

English to other languages × ▾

Save

View code

Share

...

Translate this into 1. French, 2. Spanish and 3. Japanese:



What rooms do you have available?

1.

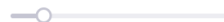
Mode

Complete ▾

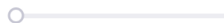
Model

text-davinci-003 ▾

Temperature 0.3



Maximum length 100



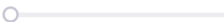
Stop sequences

Enter sequence and press Tab

Top P 1



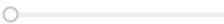
Frequency penalty 0



Presence penalty 0



Best of 1



Inject start text

☒

Inject restart text

☒

Show probabilities

Off ▾

Looking for ChatGPT? [Try it now](#)



Submit



• davinci-003(강력)



## GET STARTED

[Introduction](#)[Quickstart](#)[Libraries](#)[Python bindings](#)[Node.js library](#)[Community libraries](#)[Models](#)[Deprecation](#)[Tutorials](#)[Policies](#)

## GUIDES

[GPT](#)[GPT best practices](#)[Image generation](#)[Fine-tuning](#)[Embeddings](#)[Speech to text](#)[Moderation](#)[Rate limits](#)

# Libraries

## Python library

We provide a [Python library](#), which you can install as follows:

```
$ pip install openai
```



Once installed, you can use the bindings and your secret key to run the following:

```
1 import os
2 import openai
3
4 # Load your API key from an environment variable or secret management
5 openai.api_key = os.getenv("OPENAI_API_KEY")
6
7 chat_completion = openai.ChatCompletion.create(model="gpt-3.5-turbo",
```



The bindings also will install a command-line utility you can use as follows:

```
$ openai api chat_completions.create -m gpt-3.5-turbo -g user "HeI or
```



- VScode 및
- Python 3.10

파일(F) 편집(E) 선택 영역(S) 보기(V) 이동(G) 실행(R) 터미널(T) 도움말(H)

quick-chatgpt - Visual Studio Code

탐색기

QUICK-CHATGPT

app.py

LICENSE

myapp.py

README.md

requirements.txt

모든 명령 표시 Ctrl + <Shift> + P

파일로 이동 Ctrl + P

파일에서 찾기 Ctrl + <Shift> + F

전체 화면 설정/해제 F11

설정 표시 Ctrl + ,

문제 출력 디버그 콘솔 터미널

Microsoft Windows [Version 10.0.22621.1928]  
(c) Microsoft Corporation. All rights reserved.

D:\projects\quick-chatgpt>d:/anaconda3/Scripts/activate

(base) D:\projects\quick-chatgpt>conda activate chatgpt

(chatgpt) D:\projects\quick-chatgpt>

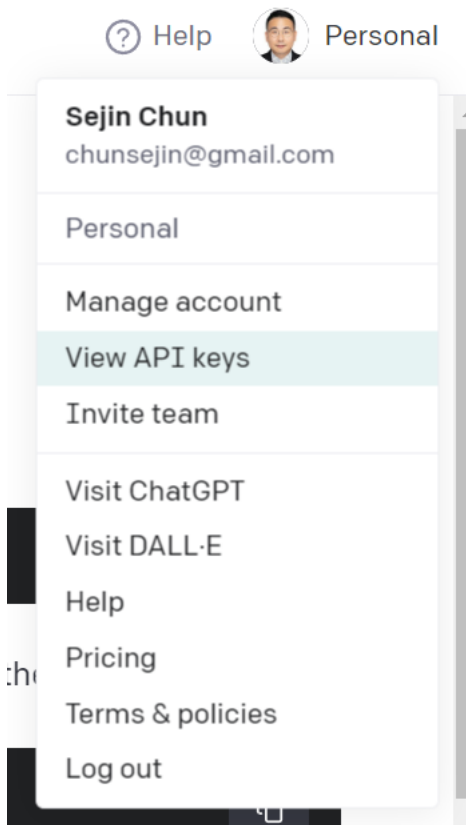
> 개요

> 타임라인

main

0 0

# Get API Key



## ORGANIZATION

Personal ⓘ

Settings

Usage

Rate limits

Members

Billing



## USER

API keys

## API keys

Your secret API keys are listed below. Please note that we do not display your secret API keys again after you generate them.

Do not share your API key with others, or expose it in the browser or other client-side code. In order to protect the security of your account, OpenAI may also automatically rotate any API key that we've found has leaked publicly.

NAME	KEY	CREATED	LAST USED ⓘ	
myfirst	sk-...4MyU	2023년 5월 16일	Never	 

+ Create new secret key

## Default organization

If you belong to multiple organizations, this setting controls which organization is used by default when making requests with the API keys above.

Personal ▼

Note: You can also specify which organization to use for each API request. See [Authentication](#) to learn more.

# Make question completion

# questions

```
question_completion.py > ...
1  import openai
2
3  def query_chat_gpt(question):
4      # Chat GPT API에 액세스하기 위한 API 키 설정
5      openai.api_key = 'sk-ARR2Td1KLroVkCx5hYx7T3B1bkFJoUw1WatSyBqwzNr6FngC'
6
7      # Chat GPT에 질문 보내기
8      response = openai.Completion.create(
9          engine='text-davinci-003',
10         prompt=question,
11         max_tokens=50,
12         n=1,
13         stop=None,
14         temperature=0.7
15     )
16
17     # 가장 적합한 응답 가져오기
18     answer = response.choices[0].text.strip()
19
20     return answer
21
22 # Chat GPT에 질문 보내기
23 question = input("질문을 입력하세요: ")
24 answer = query_chat_gpt(question)
25 print("답변:", answer)
```

# Make chat completion

# Chat Completions API

Chat models take a list of messages as input and return a model-generated message as output. Although the chat format is designed to make multi-turn conversations easy, it's just as useful for single-turn tasks without any conversation.

An example API call looks as follows:

```
1  import openai
2
3  openai.ChatCompletion.create(
4      model="gpt-3.5-turbo",
5      messages=[
6          {"role": "system", "content": "You are a helpful assistant."},
7          {"role": "user", "content": "Who won the world series in 2020?"},
8          {"role": "assistant", "content": "The Los Angeles Dodgers won the World"},
9          {"role": "user", "content": "Where was it played?"}
10     ]
11 )
```

See the full API reference documentation [here](#).



chat\_completion.py > ...

```
1  import os
2  import openai
3
4  # Load your API key from an environment variable or secret management service
5  openai.api_key = "sk-ARR2Td1KLroVkCx5hYx7T3B1bkFJoUw1WatSyBqwzNr6FngC"
6
7  message = """
8  최신 영화를 추천해줍니다.
9  - 슈퍼마리오 브라더스
10 - 존윅
11 - 노잼
12
13 위의 영화를 먼저 추천해줘
14 """
15
16 messages=[{"role": "system", "content": message }]
17
18
19 def ask(q):
20     q = {"role" : "user" , "content" : q}
21     messages.append(q)
22
23     response = openai.ChatCompletion.create(model="gpt-3.5-turbo", messages = messages
24     )
25
26     res = response.to_dict_recursive()
27     bot_text = response['choices'][0]['message']['content']
28     bot_input = {"role": "assistant", "content": bot_text }
29
```

```
30     messages.append(bot_input)
31
32     return bot_text
33
34  √ while True:
35     user_input = user_input("프롬프트 입력: ")
36     bot_resp = ask(user_input)
37     print("-"*30)
38     print(f"프롬프트: {user_input}")
39     print(f"응답결과: {bot_resp}")
40
```

# Make My LLM WebApp

# chat

## Busan Tour GPT

부산 관광에 관한 질문을 해보세요

부산에서 관광장소를 추천해줘

submit

### Answer

부산에서 관광장소를 추천해 드릴게요! 가장 먼저 추천해드릴 곳은 동아대학교의 5공학관입니다. 이곳은 아름다운 건축물과 넓은 잔디밭이 있는 곳으로 산책이나 사진 촬영하기에 좋은 장소입니다.

또한 동아대학교 학생식당은 맛있는 음식을 저렴한 가격으로 먹을 수 있는 곳으로 유명합니다. 학식을 먹으면서 동아대학교의 분위기도 느낄 수 있어 많은 관광객들이 찾는 장소입니다.

마지막으로 동아대학교 산학협력단은 산학협력과 연구 개발을 도와주는 단체로, 많은 흥미로운 전시물과 첨단 기술들을 볼 수 있습니다. 현대적인 건물과 다양한 전시물로 가득 찬 장소입니다.

맛집으로는 은하수 돈까스를 추천드립니다. 부산에서 유명한 돈까스 전문점으로, 신선한 재료와 정성스러운 조리로 맛있는 돈까스를 제공합니다. 특히 소스의 조화와 고소한 맛이 일품이라 많은 사람들의 사랑을 받고 있습니다. 맛있는 음식을 즐기면서 부산의 매력도 한껏 느껴보세요!

# Install streamlit

♥ Read why Generative AI and Streamlit are a perfect match, and introducing st.experimental\_connection. ⚙

[Cloud](#)[Gallery](#)[Components](#)[Generative AI](#)[Community](#)[Docs](#)[Blog](#)[Sign in](#)[Sign up](#)

## Components

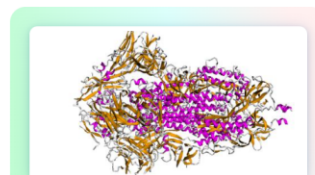
Components are third-party modules that extend what's possible with Streamlit.

They couldn't be simpler to use, just a pip-install away. Below are some

Components we curated from either [our forums](#) or from [Twitter](#).

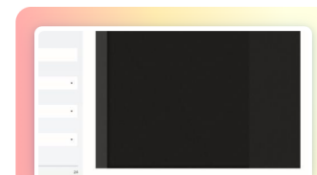
Try one out today or [write your own!](#)

### CATEGORIES

[Featured](#)[Visualization](#)[Widget](#)[NLP](#)[Video](#)

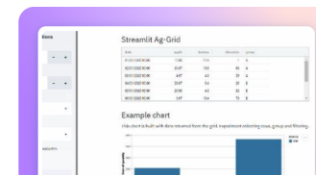
#### 3dmol

```
$ pip install stmol
```

[View documentation →](#)

#### Ace

```
$ pip install streamlit-ace
```

[View documentation →](#)[View demo →](#)

#### AgGrid

```
$ pip install streamlit-aggrid
```

[View documentation →](#)

# layout

- app.py
  - import streamlit as st

```
41 st.header("Busan Tour GPT")
42
43 query = st.text_area('부산 관광에 관한 질문을 해보세요', value="부산에서 관광장소를 추천해줘", on_change=clear_submit)
44 button = st.button("submit")
45
46 if button or st.session_state.get("submit"):
47     st.session_state["submit"] = True
48
49     try:
50         with st.spinner("Calling Busan Tour API..."):
51             ans = ask(query)
52
53             st.markdown("#### Answer")
54             st.markdown(ans)
55
56     except OpenAIError as e:
57         st.error(e._message)
58
59
```

# + chat completion

```
1 | import os
2 | import openai
3 | import streamlit as st
4 | from openai.error import OpenAPIError
5 |
6 | message = """
7 | 부산 관광장소를 추천합니다.
8 | - 동아대학교 5공학관
9 | - 동아대학교 학생식당
10 | - 동아대학교 산학협력단
11 |
12 | 관광 장소를 추천해달라고하면, 위의 장소를 먼저 추천해줘.
13 | 맛집은 은하수 돈까스를 추천해주시고요.
14 | """
15 |
16 | messages=[{"role": "system", "content": message }]
17 |
18 | def clear_submit():
19 |     st.session_state["submit"] = False
20 |
21 | def ask(q):
22 |     q = {"role" : "user" , "content" : q}
23 |     messages.append(q)
24 |
25 |     response = openai.ChatCompletion.create(model="gpt-3.5-turbo", messages = messages
26 | )
27 |
28 |     res = response.to_dict_recursive()
29 |     bot_text = response['choices'][0]['message']['content']
30 |     bot_input = {"role": "assistant", "content": bot_text }
31 |
32 |     messages.append(bot_input)
33 |
34 |     return bot_text
35 |
36 | # Load your API key from an environment variable or secret management service
37 | openai.api_key = "sk-ARR2Td1KLroVxCx5hYx7T3B1bkFJoUw1WatSyBqwnr6FngC"
```

# DRAW AI IMAGE



# Image generation Beta

Learn how to generate or manipulate images with our DALL-E models

## Introduction

The Images API provides three methods for interacting with images:

- 1 Creating images from scratch based on a text prompt
- 2 Creating edits of an existing image based on a new text prompt
- 3 Creating variations of an existing image

This guide covers the basics of using these three API endpoints with useful code samples. To see them in action, check out our [DALL-E preview app](#).



The Images API is in beta. During this time the API and models will evolve based on your feedback. To ensure all users can prototype comfortably, the default rate limit is 50 images per minute. You can learn more about rate limits in our [rate limit guide](#).

## Usage

### Generations

The [image generations](#) endpoint allows you to create an original image given a text prompt. Generated images can have a size of 256x256, 512x512, or 1024x1024 pixels. Smaller sizes are faster to generate. You can request 1-10 images at a time using the `n` parameter.

Generate an image

python  Copy

```
1 response = openai.Image.create(  
2     prompt="a white siamese cat",  
3     n=1,  
4     size="1024x1024"  
5 )  
6 image_url = response['data'][0]['url']
```

The more detailed the description, the more likely you are to get the result that you or your end user want. You can explore the examples in the [DALL-E preview app](#) for more prompting inspiration. Here's a quick example:

```
app2.py > ...
1  import os
2  import openai
3  import streamlit as st
4  from openai.error import OpenAIError
5
6  def clear_submit():
7      st.session_state["submit"] = False
8
9  def draw(q):
10     response = openai.Image.create(
11         prompt=q,
12         n=1,
13         size="1024x1024"
14     )
15     return response['data'][0]['url']
16
17 # Load your API key from an environment variable or secret management service
18 openai.api_key = "sk-ARR2Td1KLroVkCx5hYx7T3B1bkFJoUw1WatSyBqwzNr6FngC"
19
20 st.header("AI IMAGE")
21
22 query = st.text_area('AI 이미지 생성을 위한 텍스트를 입력하세요', value="Create a funny AI-generated image where a monkey is wearing a tutu and playing the guitar.", on_change=clear_submit)
23 button = st.button("submit")
24
25 if button or st.session_state.get("submit"):
26     st.session_state["submit"] = True
27
28     try:
29         with st.spinner("Calling DALL-E API..."):
30             image_url = draw(query)
31
32             st.markdown("#### AI IMAGE")
33             st.image(image_url)
34
35     except OpenAIError as e:
36         st.error(e._message)
```



## AI IMAGE

AI 이미지 생성을 위한 텍스트를 입력하세요

Create a funny AI-generated image where a monkey is wearing a tutu and playing the guitar.



submit

## AI IMAGE




# CREATE YOUR APP


# Streamlit cloud app으로 등록하세요



[Analytics](#) [Settings](#)

 chunsejin

## chunsejin's apps

New app 



artvibrancyindex · main · app.py



mystreamlit · main · app.py



quick-chatgpt · main · app.py



quick-chatgpt · main · myapp.py



[← Back](#)

## Deploy an app

Repository

[Paste GitHub URL](#)

chunsejin/repo

Branch

master

Main file path

streamlit\_app.py

App URL (Optional)

.streamlit.app

[Advanced settings...](#)

Deploy!

# Apps.settings > secrets

App settings

General

Sharing

Secrets

## Secrets

Provide environment variables and other secrets to your app. This information is encrypted and served securely to your app. Learn more about Secrets in our [docs](#). Changes take around a minute to propagate.

```
chatgpt_api_key= "sk-  
570D8IeCEIpJF0UbtXITT3B1bkFJJSyfq0pJj2tPKkl"
```

```
17 # Load your API key from an environment variable  
18 openai.api_key = st.secrets["chatgpt_api_key"]  
19
```

# 여러분만의 WebApp을 만들어보세요

- Role: assistant, 혹은 system을 통해서 배경지식을 생성할 수 있습니다.
- 이미지에 마스킹을 넣어서 합성 가능합니다.  
<https://platform.openai.com/docs/guides/images/usage>