
GenAI HW3

以API快速搭建自己的應用

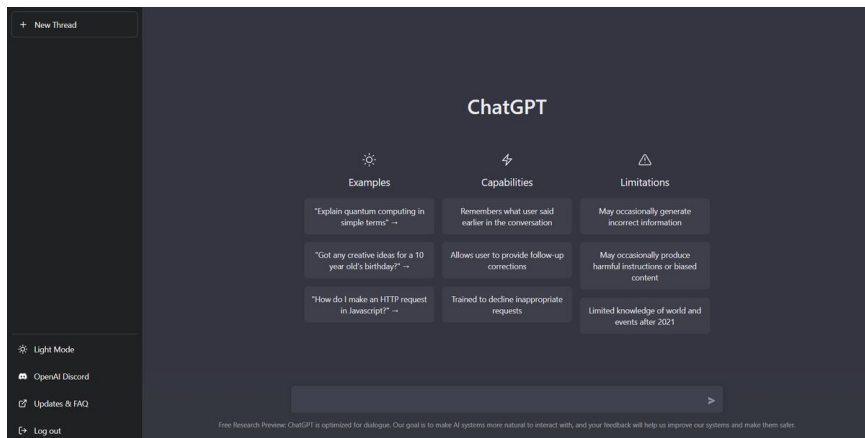
TA: 尤展鴻, 陳守仁, 呂睿超

ntu-gen-ai-2024-spring-ta@googlegroups.com

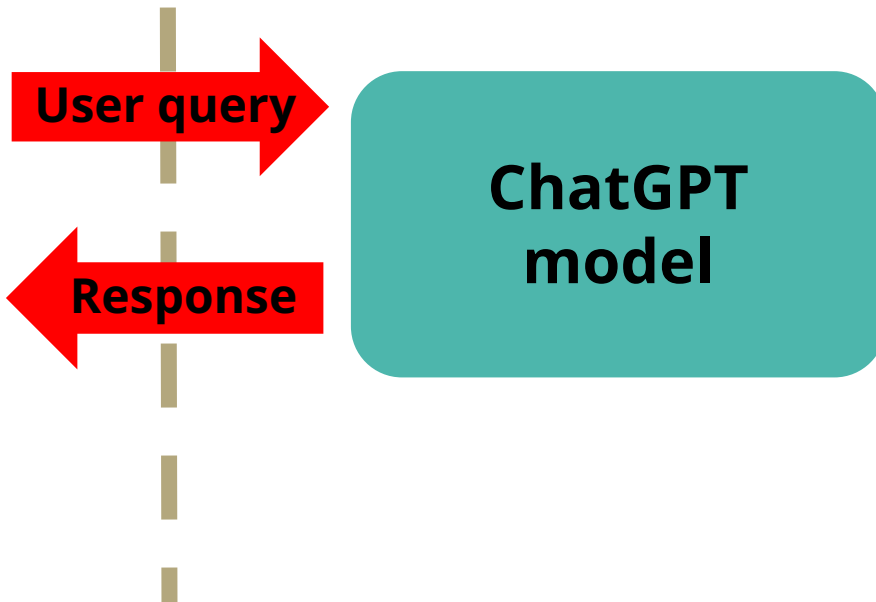
Deadline: 2024/03/21 23:59:59 (UTC+8)

Using ChatGPT Model by Browser

In the HW2, we use ChatGPT model through the browser interface



Browser



Creating Your Own Application Using ChatGPT

For example, you want to create an App (應用程式) that adds appropriate emojis to any given input, and you want to power it by ChatGPT

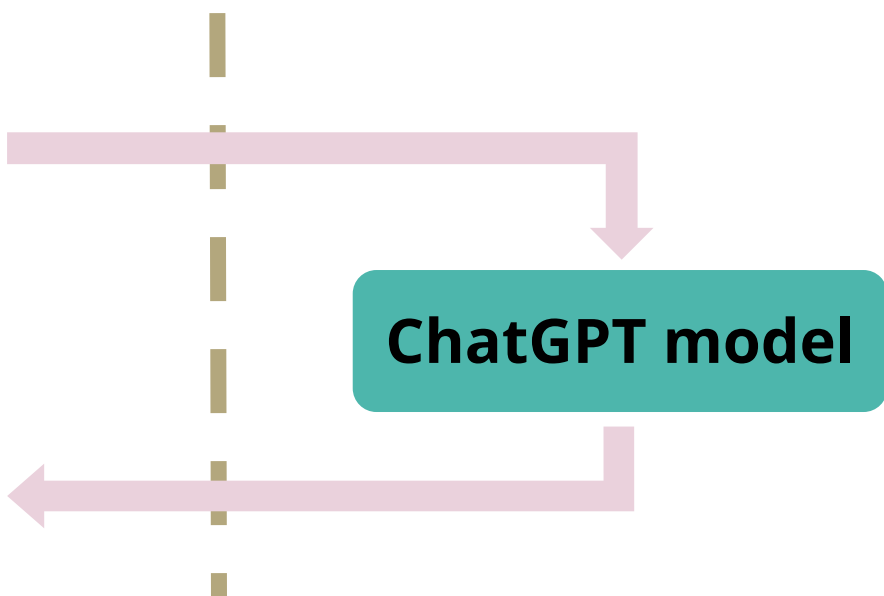
Emoji Adder App

User input:

Human evaluation is indispensable and inevitable for assessing the quality of texts generated by machine learning models or written by humans.

Paragraph with emoji:

Human evaluation 🧑 is indispensable and inevitable 🚫 for assessing the quality of texts 📄 generated by machine learning models 🤖 or written by humans 📖.



ChatGPT model

Creating Your Own Application Using ChatGPT

While you can manually copy the user input to the browser and copy the response from ChatGPT to the user end, this is too slow

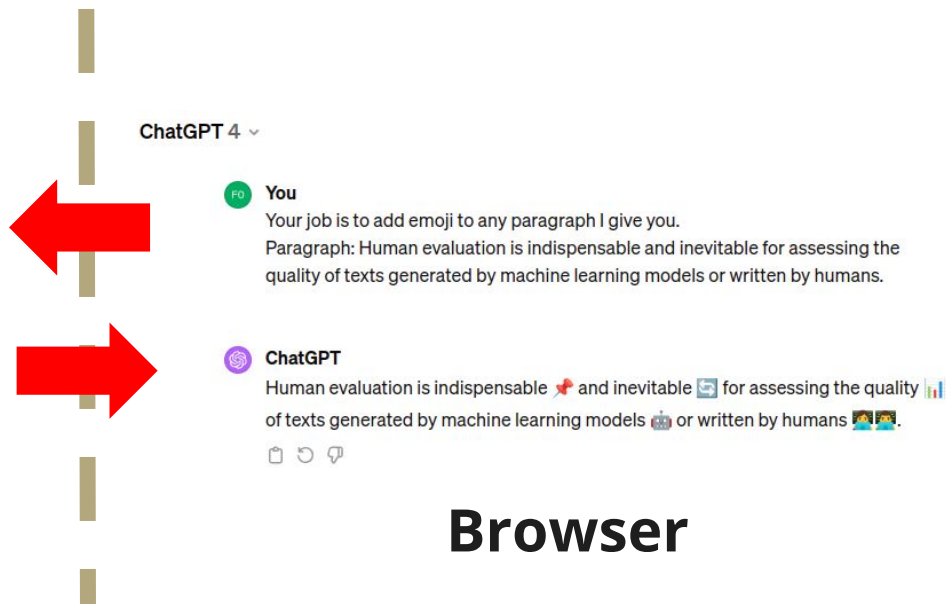
Emoji Adder App

User input:

Human evaluation is indispensable and inevitable for assessing the quality of texts generated by machine learning models or written by humans.

Paragraph with emoji:

Human evaluation 🧑 is indispensable and inevitable 🚫 for assessing the quality of texts 📄 generated by machine learning models 🤖 or written by humans 📖.



Creating Your Own Application Using ChatGPT API

The ChatGPT API (Application Programming Interface) enable programmatic (程式化的) interaction with the ChatGPT model

Emoji Adder App

User input:

Human evaluation is indispensable and inevitable for assessing the quality of texts generated by machine learning models or written by humans.

Paragraph with emoji:

Human evaluation 🧑 is indispensable and inevitable 🚫 for assessing the quality of texts 📄 generated by machine learning models 🤖 or written by humans 📖.

```
import openai
openai.api_key = 'XXXXXXXXX'

response = openai.Completion.create(
    engine="davinci",
    prompt="Add emoji to the following paragraph: <Paragraph>",
    max_tokens=60
)

print(response.choices[0].text.strip())
```

ChatGPT model

Response:

Creating Your Own Application Using ChatGPT API

You tell ChatGPT what you want it to do in a programmatic way

Emoji Adder App

User input:

Human evaluation is indispensable and inevitable for assessing the quality of texts generated by machine learning models or written by humans.

Paragraph with emoji:

Human evaluation 🧑 is indispensable and inevitable 🚫 for assessing the quality of texts 📝 generated by machine learning models 🤖 or written by humans 📖.

```
import openai
openai.api_key = 'XXXXXXXXX'

response = openai.Completion.create(
    engine="davinci",
    prompt="Add emoji to the following paragraph: <Paragraph>",
    max_tokens=60
)

print(response.choices[0].text.strip())
```

ChatGPT model

Response:

Creating Your Own Application Using ChatGPT API

ChatGPT return the response by following you what you ask it to do

Emoji Adder App

User input:

Human evaluation is indispensable and inevitable for assessing the quality of texts generated by machine learning models or written by humans.

Paragraph with emoji:

Human evaluation 🧑 is indispensable and inevitable 🚫 for assessing the quality of texts 📝 generated by machine learning models 🤖 or written by humans 📖.

```
import openai
openai.api_key = 'XXXXXXXXX'

response = openai.Completion.create(
    engine="davinci",
    prompt="Add emoji to the following paragraph: <Paragraph>",
    max_tokens=60
)

print(response.choices[0].text.strip())
```

ChatGPT model

Response:

What You Will Learn in This Homework

You will learn how to build some simple applications using ChatGPT with its API

1. You will learn how to call the ChatGPT API (and other APIs)
2. You will learn how to design the prompts to achieve the functionality you want

```
1. → import openai
      openai.api_key = 'XXXXXXXXXX'

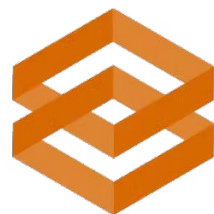
2. → response = openai.Completion.create(
      ... engine="davinci",
      prompt="Add emoji to the following paragraph: <Paragraph>",
      ... max_tokens=60
      )

      print(response.choices[0].text.strip())
```

What You Will Learn in This Homework

You will learn how to build some simple **applications** using ChatGPT with its API

- Need to write some codes → Google Colab
- Need some place to host the **APP** → **Gradio**



gradio

Overview in Today's TA Lecture

1. Step 1: Learn how to use colab for coding and Gradio for demo
2. Step 2: Learn what API you can use and how to obtain the API keys (similar to password)
3. Step 3: Create an application on Colab and Gradio using the API

Warning of Codings

No worries, the TAs have already written the codes.

Outline

- **Colab and Gradio Tutorial:** For coding and demo
- **Available APIs:** APIs you can use to build your own APP
- **Formal Task Introduction:** The APPs you need to create in HW3
- **Grading by Evaluation Assistants on DaVinci**
- **Submission and Grading on NTU COOL**
- **Important Dates**
- **Regulations**

Links

- [NTU Cool HW3 Forum](#)
- [MediaTek DaVinci](#)
- [OpenAI ChatGPT](#)
- [Google Gemini](#)
- [Colab link to this homework](#)

Colab and Gradio Tutorial

Introduction



What is Colab?

Colab, or "Colaboratory", allows you to write and execute Python in your browser, with

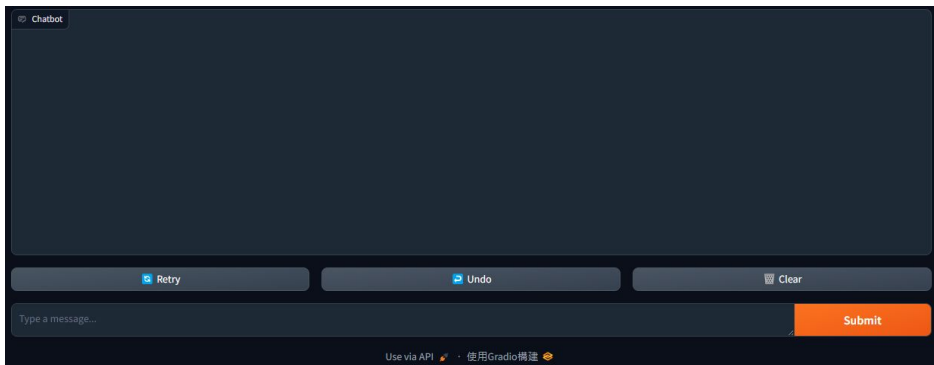
- **Zero configuration required**
- **Free access to GPUs**
- **Easy sharing**

Introduction



What is Gradio?

Gradio is a tool that allows you to demo any application with a friendly website so that anyone can use it, anywhere.



We will use the following google colab notebook for tutorial demonstration

<https://colab.research.google.com/drive/1N7WTdVseHg4pNQpDMENo2aHOfsCS2-If?usp=sharing>

Demo for Colab and Gradio Tutorial

Colab and Gradio Tutorial ☆

檔案 編輯 檢視畫面 插入 執行階段 工具 說明 系統不會儲存變更

RAM 記憶體 81.40 GB 可用

Colab AI

檔案

- {x}
- sample_data

程式碼 + 文字

複製到雲端硬碟

Create a copy in your drive

- Gradio Tutorial
 - Install Packages
 - [] 4.1 個隱藏的儲藏格
 - Import and Setup
 - [] 4.1 個隱藏的儲藏格
 - Run your code!
 - [] 4.1 個隱藏的儲藏格

You can change the name here

The screenshot shows the Google Colab interface. At the top, the notebook title is '[Colab and Gradio Tutorial] 的副本', which is highlighted with a red box. Below the title bar, the left sidebar contains a file explorer with a folder icon highlighted by a red box and labeled 'File location'. The main workspace shows a notebook with sections: 'Gradio Tutorial', 'Install Packages' (highlighted with a red box), 'Import and Setup', and 'Run your code!'. A text box on the right says 'Click this to unfold the blocks' with an arrow pointing to the 'Install Packages' section. Another text box on the right says 'If it looks like these: Hit it again.' with arrows pointing to the 'Connect' and 'Reconnect' buttons. In the top right corner, the 'RAM' dropdown menu is highlighted with a red box. The bottom status bar shows '已連線至 [Python 3 Google Compute Engine 後端]'.

File location

Click this to unfold the blocks

If it looks like these: Hit it again.

RAM

已連線至 [Python 3 Google Compute Engine 後端]

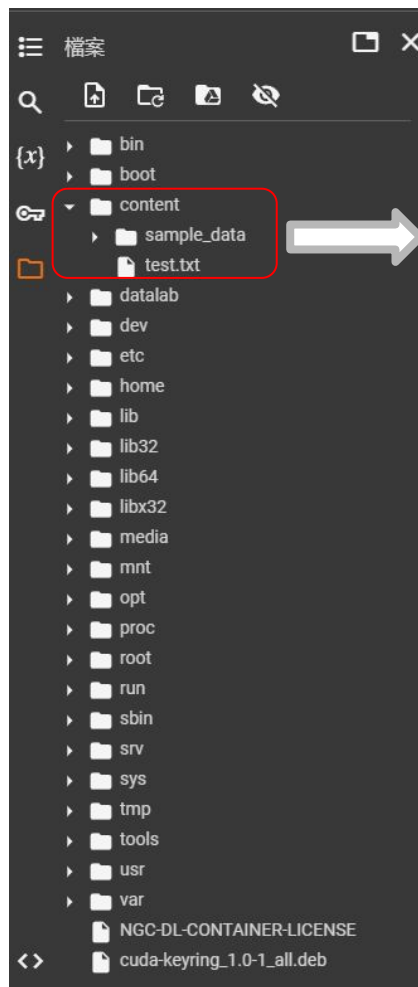


Upload your file

Download your file



If you accidentally
click this

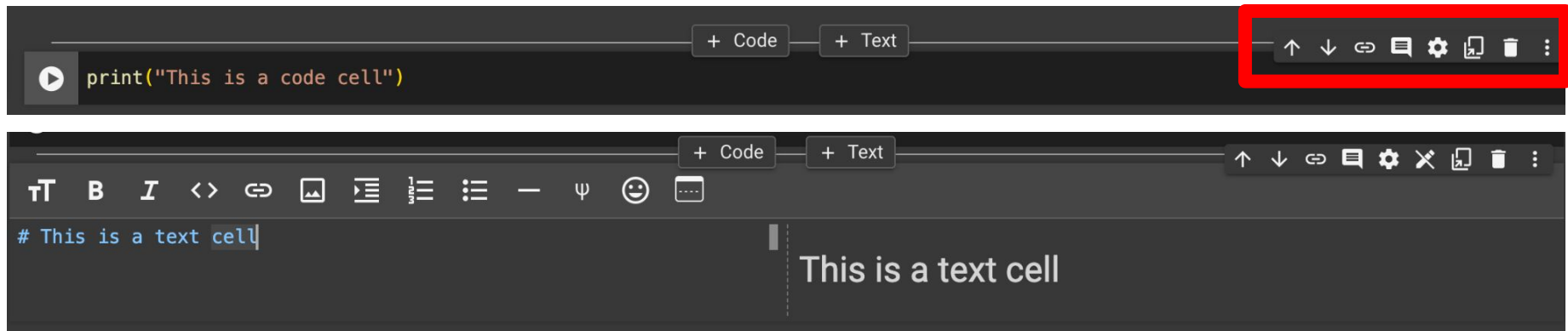


You can find
your files here

Cells

Create a new cell by clicking on `+ Code` or `+ Text`

These options allow you to moving your cell up/down or copy and deleting it



Run Cell

Run Cell by clicking on the 

 Install Packages



```
# Install Gradio and Google Gemini Package
!pip3 install gradio
!pip install -q -U google-generativeai
```

Code Unexecuted



```
!pip install gradio
```

Code Executing



```
!pip install gradio
```

Code Executed

```
!pip install gradio
```

```
Collecting websockets<12
```

Error Occurred

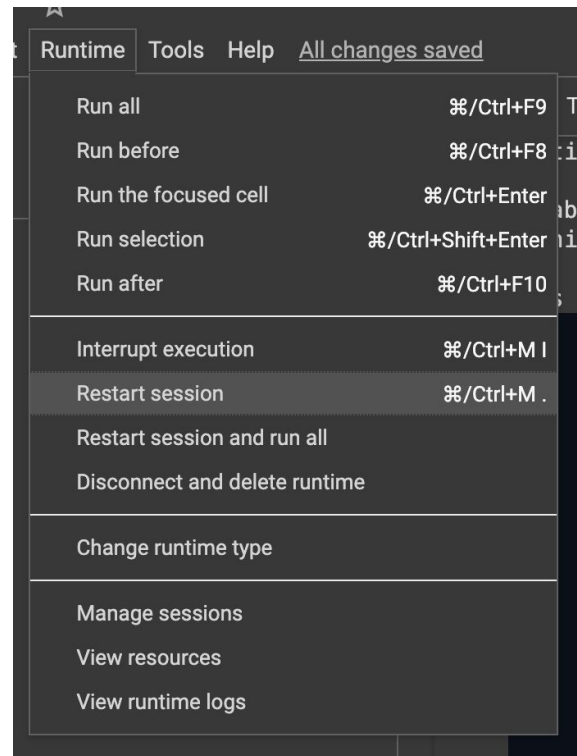
 

```
print("This is a code cell")
```

Problems you may encounter

Colab will **automatically disconnect** if idle timeout (90 min., sometimes varying) or when your screen goes black.

If you “Disconnect and delete runtime” or “Restart Session”, your files will be gone.



Run these three blocks from top to down

The screenshot shows a Google Colab notebook titled "[Colab and Gradio Tutorial] 的副本". The notebook contains several code blocks. Three blocks are highlighted with red rectangles, indicating they should be run from top to bottom:

- Block 1: A text cell containing "This is a text cell".
- Block 2: A code cell containing `[2] !pip install gradio`.
- Block 3: A code cell containing `[5] import gradio as gr`, `from typing import List, Tuple`, and `import numpy as np`.

Below these highlighted blocks, there is another code cell containing a function definition for a chatbot interface:

```
[ ] def reset() -> List:
    return []

def interact_roleplay(chatbot: List[Tuple[str, str]], user_input: str) -> List[Tuple[str, str]]:
    responses = ["You are right", "HaHa", "I don't know"]
    response = np.random.choice(responses, 1)[0]
    chatbot.append((user_input, response))

    return chatbot
```

The left sidebar shows a file explorer with a folder named "sample_data". The bottom status bar indicates "81.27 GB 可用".

Click this link will lead you to a browser version gradio

Gradio Tutorial

Chatbot

You can also use gradio inside colab

reset_button.click(reset, outputs=[chatbot])

```
demo.launch(debug=True)
```

Setting queueing in Gradio (you can turn this off by setting 'share=False' in 'launch()' explicitly).

Colab notebook detected. This cell will run indefinitely so that you can see errors and logs. To turn off, set debug=False in launch().

Running on public URL: <https://08f5d855001ccfb5a9.gradio.live>

Gradio Tutorial

Chatbot

You can also use gradio inside colab

執行中 (已持續 42 秒) <cell line: 23> launch() > block_thread()

Gradio Tutorial

Chatbot

Good morning

I don't know

Dialogue

Input

Good morning



Input box

Send

Reset

Click Send to enter
your input

Click Reset to reset
the dialogue

```
# function to reset the conversation
def reset() -> List:
    return []

# function to generate the output,
# it will randomly generate one response from ["You are right", "HaHa", "I don't know"]
def interact(chatbot: List[Tuple[str, str]], user_input: str) -> List[Tuple[str, str]]:
    responses = ["You are right", "HaHa", "I don't know"]
    response = np.random.choice(responses, 1)[0]
    chatbot.append((user_input, response))

    return chatbot

# gradio body
with gr.Blocks() as demo:
    gr.Markdown(f"# Gradio Tutorial")
    chatbot = gr.Chatbot()
    input_textbox = gr.Textbox(label="Input", value = "")

    with gr.Row():
        sent_button = gr.Button(value="Send")
        reset_button = gr.Button(value="Reset")

    sent_button.click(interact, inputs=[chatbot, input_textbox], outputs=[chatbot])
    reset_button.click(reset, outputs=[chatbot])

demo.launch(debug = True)
```

Gradio Tutorial

Chatbot

Input

Reset

```
# function to reset the conversation
def reset() -> List:
    return []

# function to generate the output,
# it will randomly generate one response from ["You are right", "HaHa", "I don't know"]
def interact(chatbot: List[Tuple[str, str]], user_input: str) -> List[Tuple[str, str]]:
    responses = ["You are right", "HaHa", "I don't know"]
    response = np.random.choice(responses, 1)[0]
    chatbot.append((user_input, response))

    return chatbot

# gradio body
with gr.Blocks() as demo:

    gr.Markdown(f"# Gradio Tutorial")

    chatbot = gr.Chatbot()

    input_textbox = gr.Textbox(label="Input", value = "")

    with gr.Row():

        sent_button = gr.Button(value="Send")

        reset_button = gr.Button(value="Reset")

    sent_button.click(interact, inputs=[chatbot, input_textbox], outputs=[chatbot])

    reset_button.click(reset, outputs=[chatbot])

demo.launch(debug = True)
```

Gradio Tutorial

Chatbot

Input

haha

Send

Reset

When it's spinning, the gradio is connected to the colab.
When it stops, the gradio will no longer work.

```
# function to reset the conversation
def reset() -> List:
    return []

# function to generate the output,
# it will randomly generate one response from ["You are right", "HaHa", "I don't know"]
def interact(chatbot: List[Tuple[str, str]], user_input: str) -> List[Tuple[str, str]]:
    responses = ["You are right", "HaHa", "I don't know"]
    response = np.random.choice(responses, 1)[0]
    chatbot.append((user_input, response))

    return chatbot

# gradio body
with gr.Blocks() as demo:

    gr.Markdown(f"# Gradio Tutorial")

    chatbot = gr.Chatbot()

    input_textbox = gr.Textbox(label="Input", value = "")

    with gr.Row():

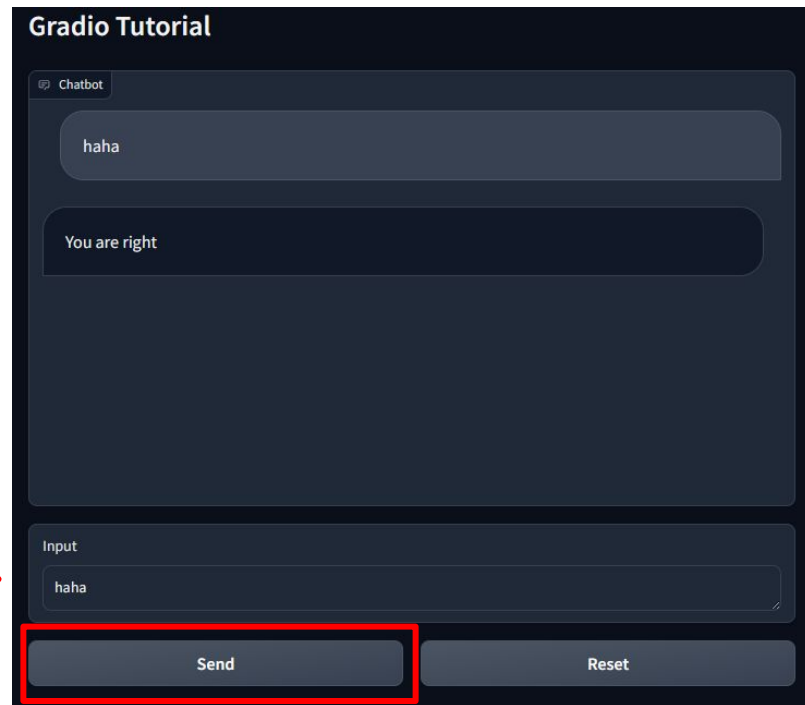
        sent_button = gr.Button(value="Send")

        reset_button = gr.Button(value="Reset")

    sent_button.click(interact, inputs=[chatbot, input_textbox], outputs=[chatbot])

    reset_button.click(reset, outputs=[chatbot])

demo.launch(debug = True)
```



Available APIs

Available APIs

You only have to **choose one API** to use.

Google Gemini API

- Free to use

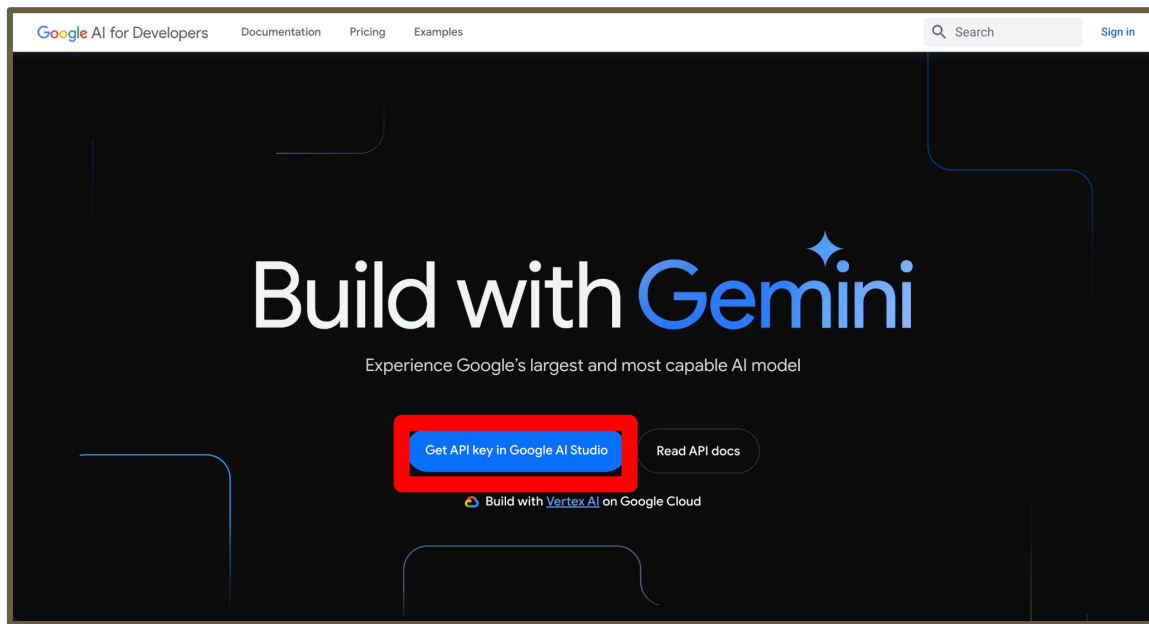
ChatGPT (OpenAI API)

- Might not be free

Available APIs: Google Gemini

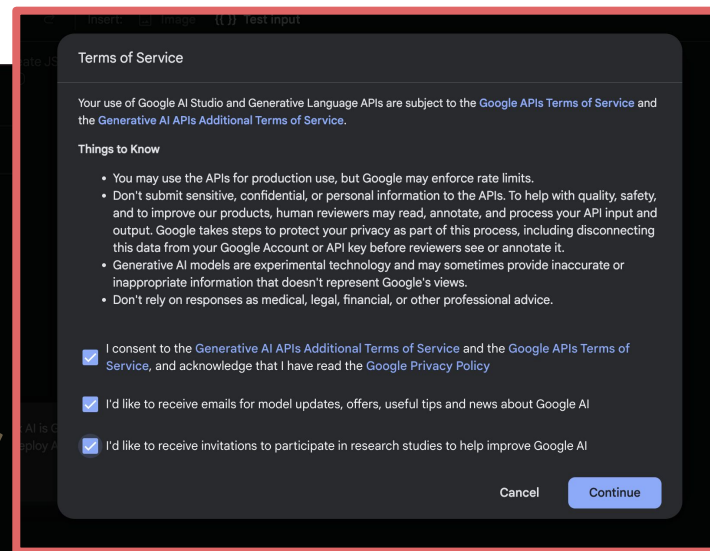
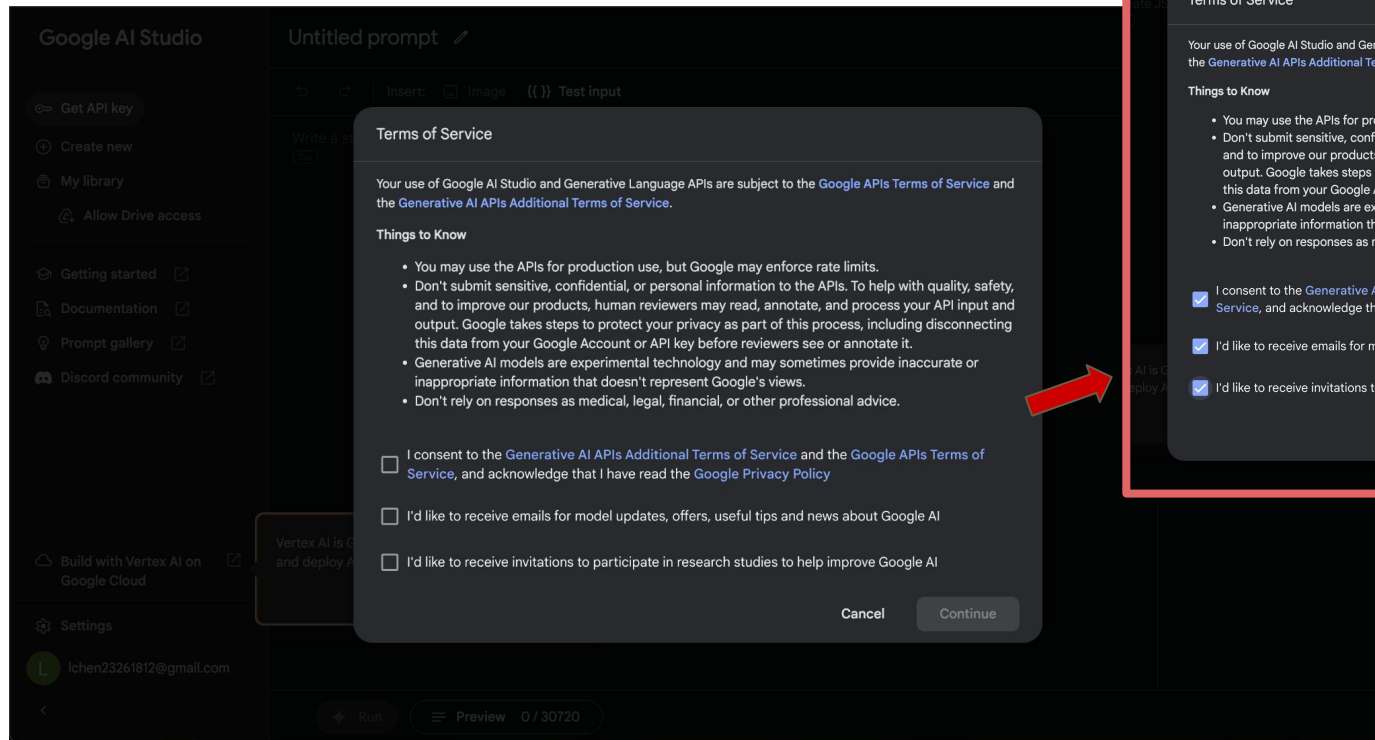
Google Gemini API

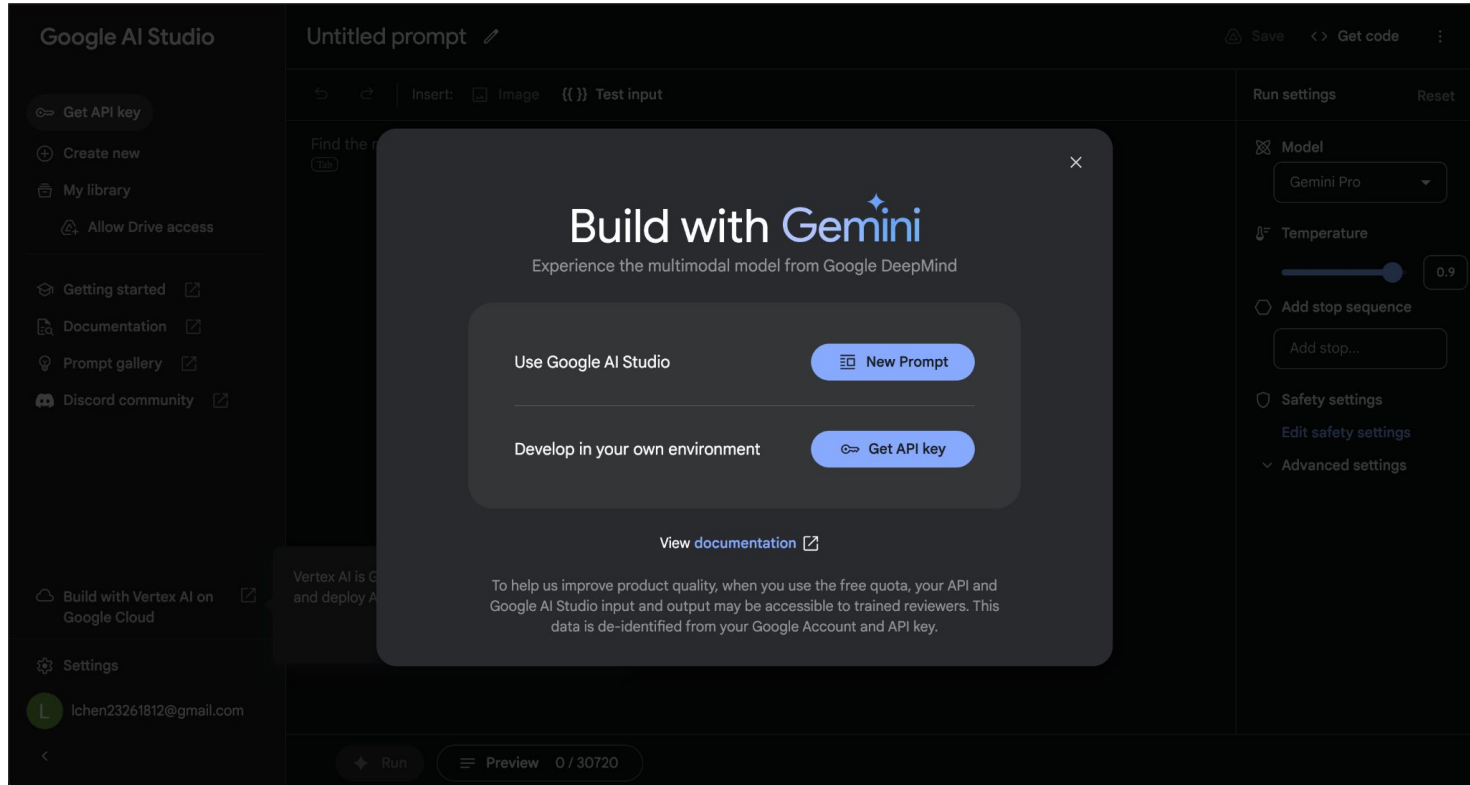
<https://ai.google.dev/>



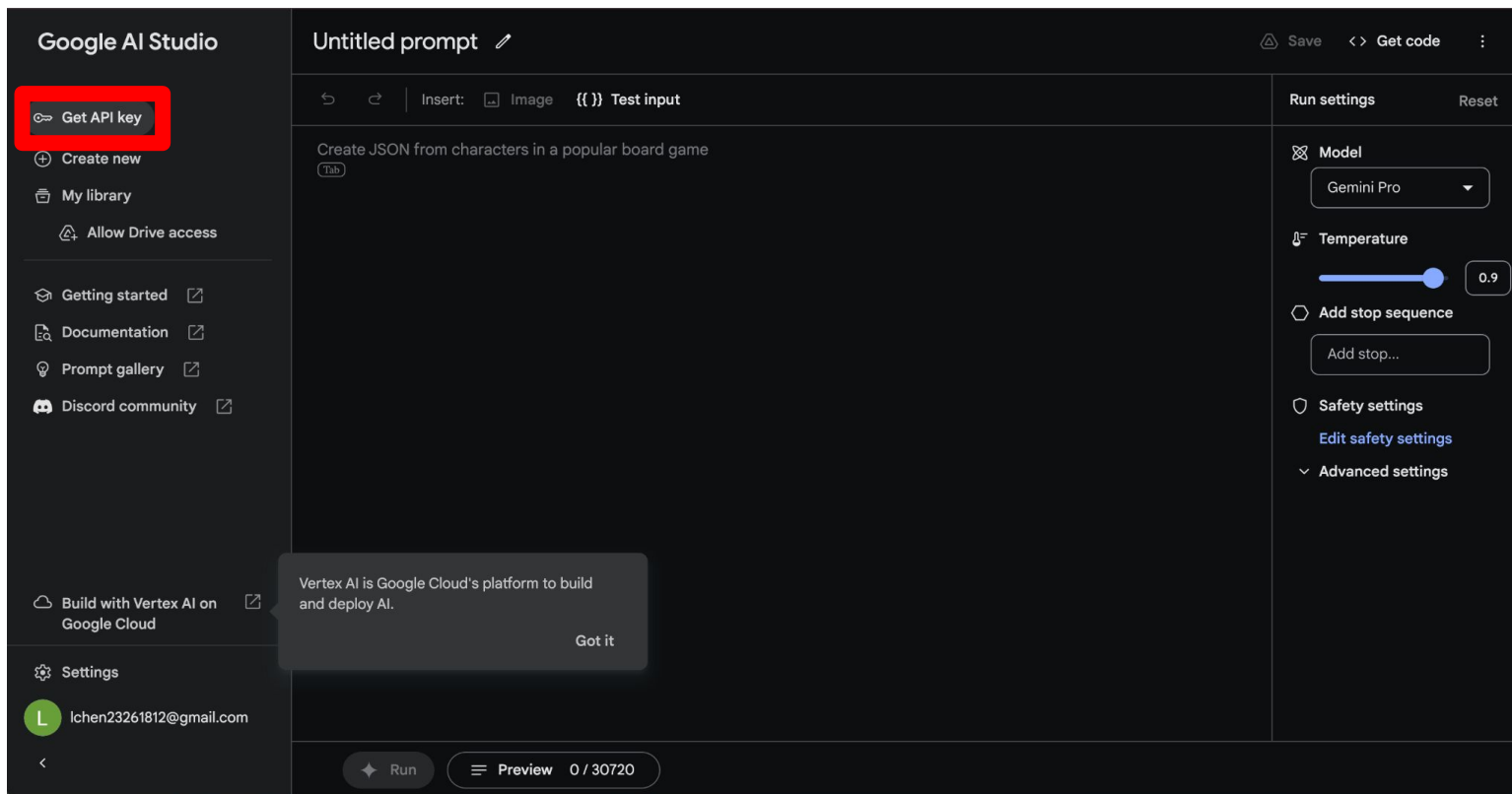
Demo for Gemini API key

Step 1: Accept Terms of Service





Step 2: Get API Key



The screenshot displays the Google AI Studio web interface. On the left sidebar, the 'Get API key' button is highlighted with a red rectangle. Below it are options for 'Create new', 'My library', and 'Allow Drive access'. Further down are links for 'Getting started', 'Documentation', 'Prompt gallery', and 'Discord community'. At the bottom of the sidebar, there is a 'Build with Vertex AI on Google Cloud' link and a 'Settings' option. The main area is titled 'Untitled prompt' and contains a text input field with the prompt 'Create JSON from characters in a popular board game'. To the right of the main area is the 'Run settings' panel, which includes a 'Model' dropdown set to 'Gemini Pro', a 'Temperature' slider set to 0.9, an 'Add stop sequence' button, and 'Safety settings' and 'Advanced settings' sections. A 'Got it' button is visible in a tooltip at the bottom of the main area. The bottom of the interface features a 'Run' button and a 'Preview' button with a token count of '0 / 30720'.

Step 3: Create API Key

Google AI Studio

Get API key

API keys

Google AI Studio creates a new Google Cloud project for each new API key. You also can create an API key in an existing Google Cloud project. All projects are subject to the [Google Cloud Platform Terms of Service](#).

Note: The Gemini API is currently in public preview. Production applications are not supported yet.

[Create API key in new project](#) or [Create API key in existing project](#)

Getting started [↗](#)

Documentation [↗](#)

Prompt gallery [↗](#)

Discord community [↗](#)

Build with Vertex AI on Google Cloud [↗](#)

Settings

L lchen23261812@gmail.com

Final: Copy

API keys

Google AI Studio creates a new Google Cloud project for each new API key. You also can create an API key in an existing Google Cloud project. All projects are subject to the [Google Cloud Platform Terms of Service](#).

Note: The Gemini API is currently in public preview. Production applications are not supported yet.

[Create API key in new project](#) or [Create API key in existing project](#)

API key generated

Use your API keys securely. Do not share them or embed them in code the public can view.

AlzaSyAoGw2nt7sg_MGEPhsDjBiDA9Ow0GIWBtE

Copy

Action

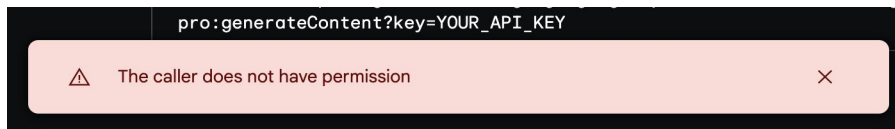
Quickly test the API by running a cURL command

```
curl \
  -H 'Content-Type: application/json' \
  -d '{"contents":[{"parts":[{"text":"Write a story about a magic backpack"}]}]' \
  -X POST https://generativelanguage.googleapis.com/v1beta/models/gemini-pro:generateContent?key=YOUR_API_KEY
```

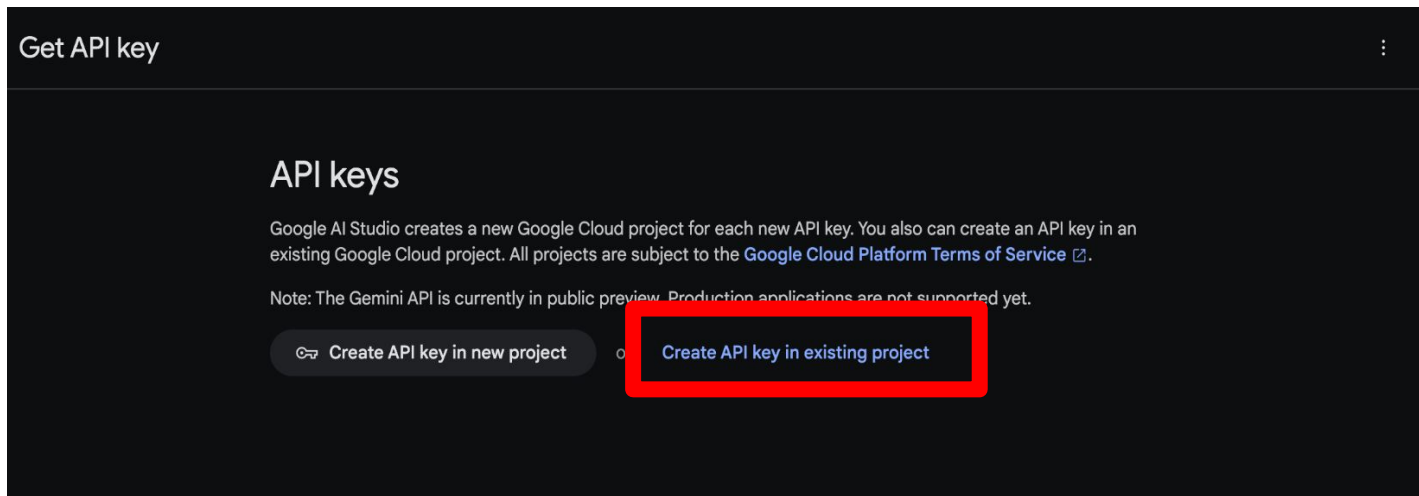
[API quickstart guide](#)

Problems you may encounter

IF

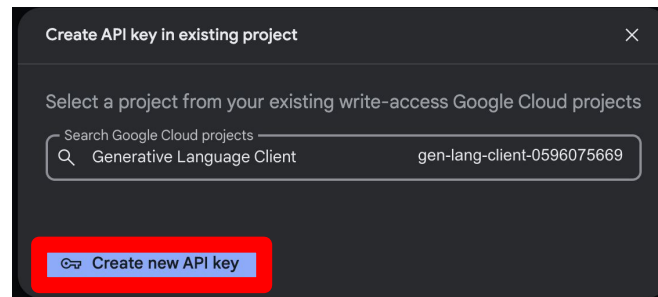
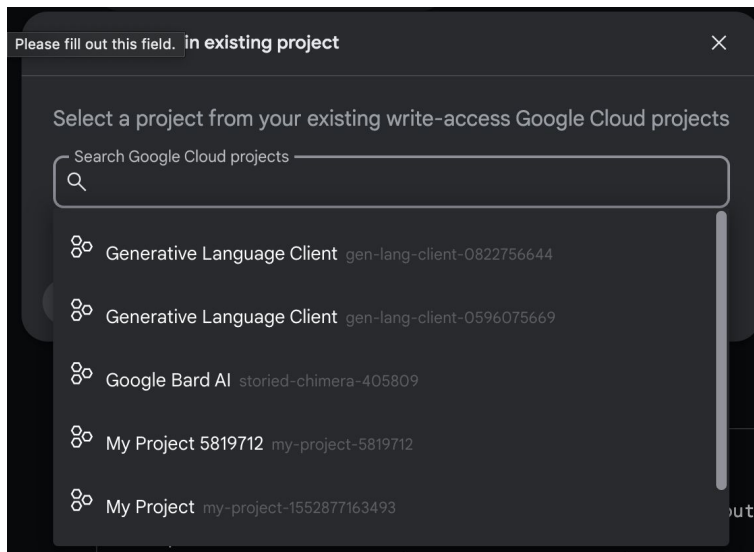


THEN



Problems you may encounter

THEN



Issue #2

Do not use *G-Suite accounts*, use your own account

The screenshot shows a web page from Google AI for Developers. On the left is a sidebar with a 'Filter' button and a list of categories: Python, Embeddings, Debugging, and Other guides. The 'Access Google AI Studio with Google Workspace' link under the Debugging section is highlighted. The main content area has a breadcrumb 'Google AI for Developers > Products', a 'Was this helpful?' section with a 'Send feedback' button, and a title 'Access Google AI Studio with your Workspace account'. Below the title is a table of contents with links to 'On this page', 'Prerequisites', 'Turn on Early Access apps', and 'Troubleshooting'. The 'Prerequisites' section is expanded, showing a paragraph about account permissions and a link to 'Learn more'. At the bottom, it starts with 'As an admin, you can control who uses Early Access apps:'.

Filter

Python
Structured data extraction

Embeddings
Overview
Anomaly detection with embeddings
Clustering with Embeddings
Document Q&A
Text classifier with embeddings

Debugging
Access Google AI Studio with Google Workspace
Troubleshooting guide
Google AI Studio FAQs
Request increase in quota
Release notes

Other guides
Firebase extensions

Google AI for Developers > Products

Was this helpful?

[Send feedback](#)

Access Google AI Studio with your Workspace account

On this page
Prerequisites
Turn on Early Access apps
Troubleshooting

This page explains how to enable Google AI Studio for Google Workspace users in your organization. If you're having trouble accessing Google AI Studio with your Workspace account, see [Troubleshooting](#).

Prerequisites

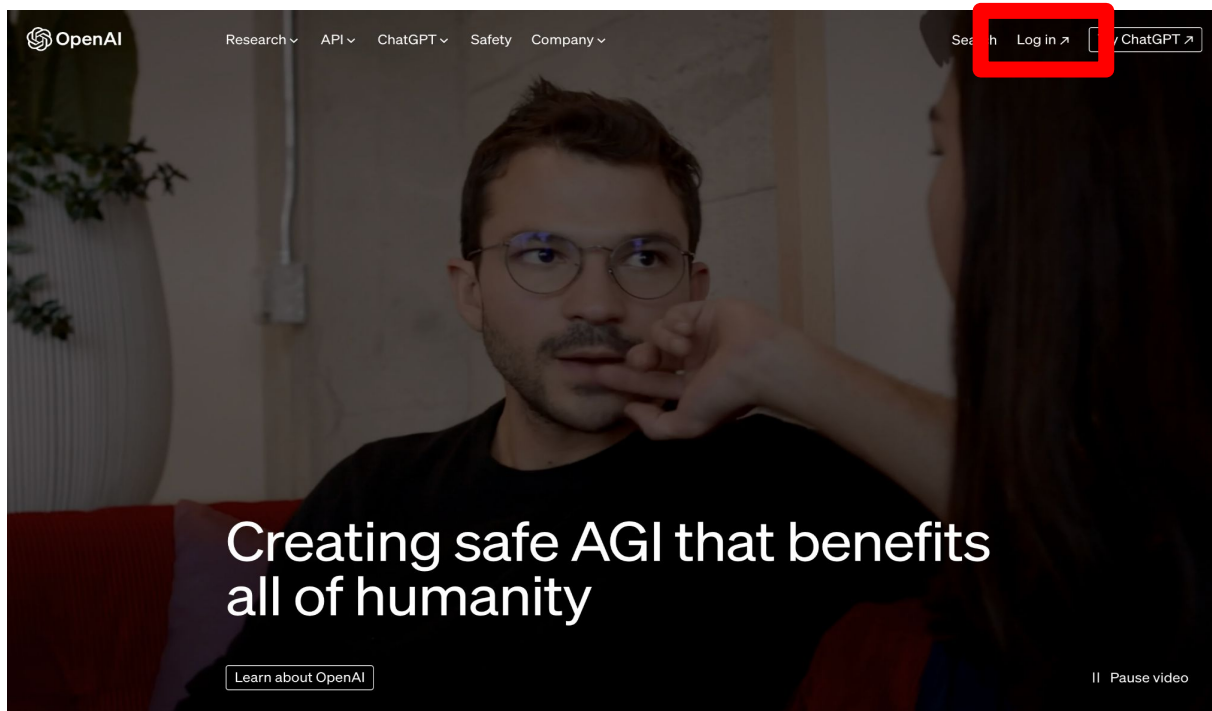
Your current account might not have permission to do these steps. To continue, make sure you're signed in to an *admin account*. [Learn more](#)

As an admin, you can control who uses Early Access apps:

Available APIs: ChatGPT API

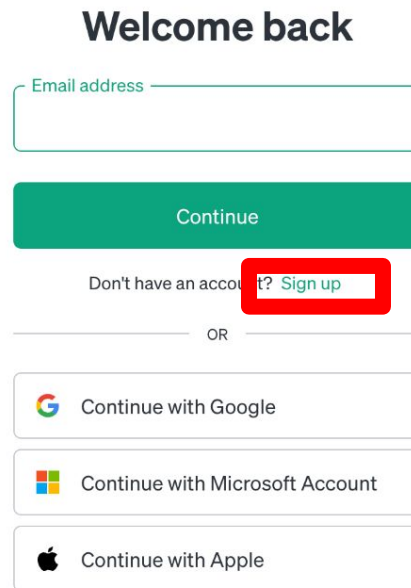
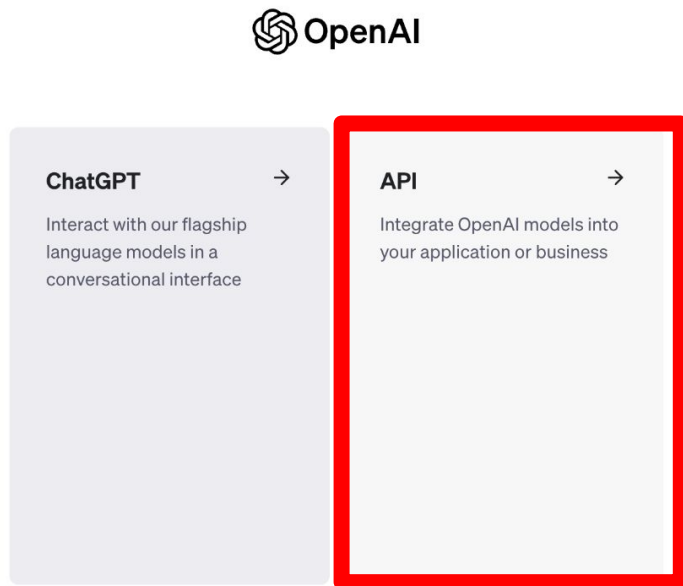
Step 0 - Go to OpenAI Website

<https://openai.com/>



Demo for ChatGPT API key

Step 1 - Login/Create Account



Step 2 - Create Account(Cont.)

Create your account

fanandy1607@gmail.com

Edit

Password



Your password must contain:

✓ At least 12 characters

Continue

Already have an account? [Log in](#)

Tell us about you

Full name

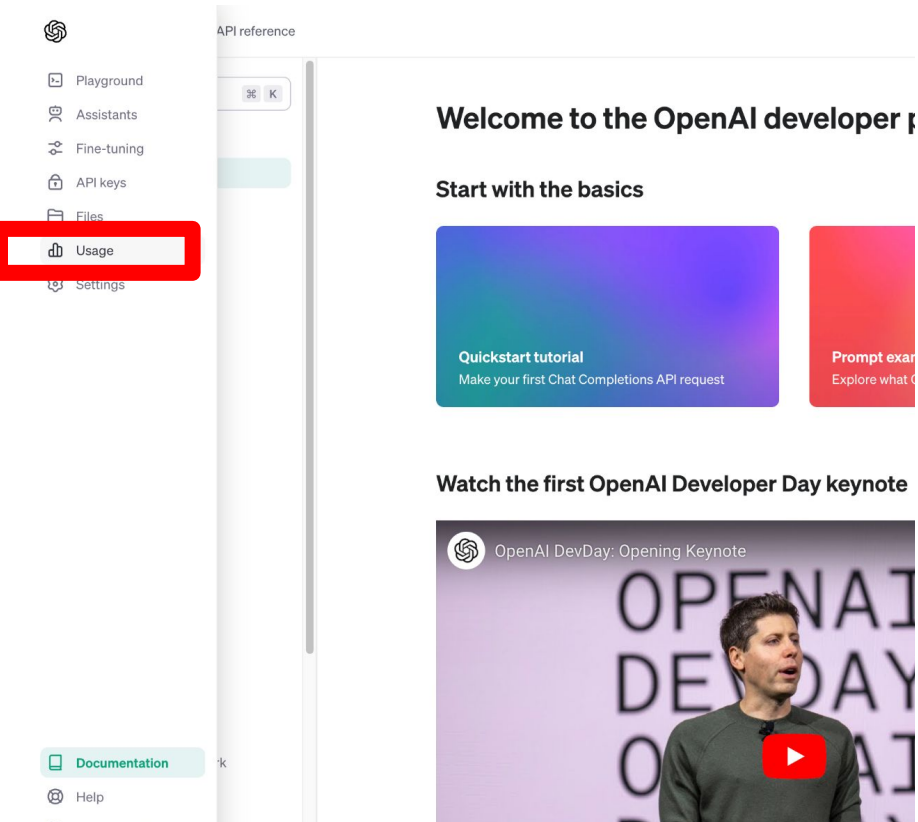
Organization name (optional)

Birthday

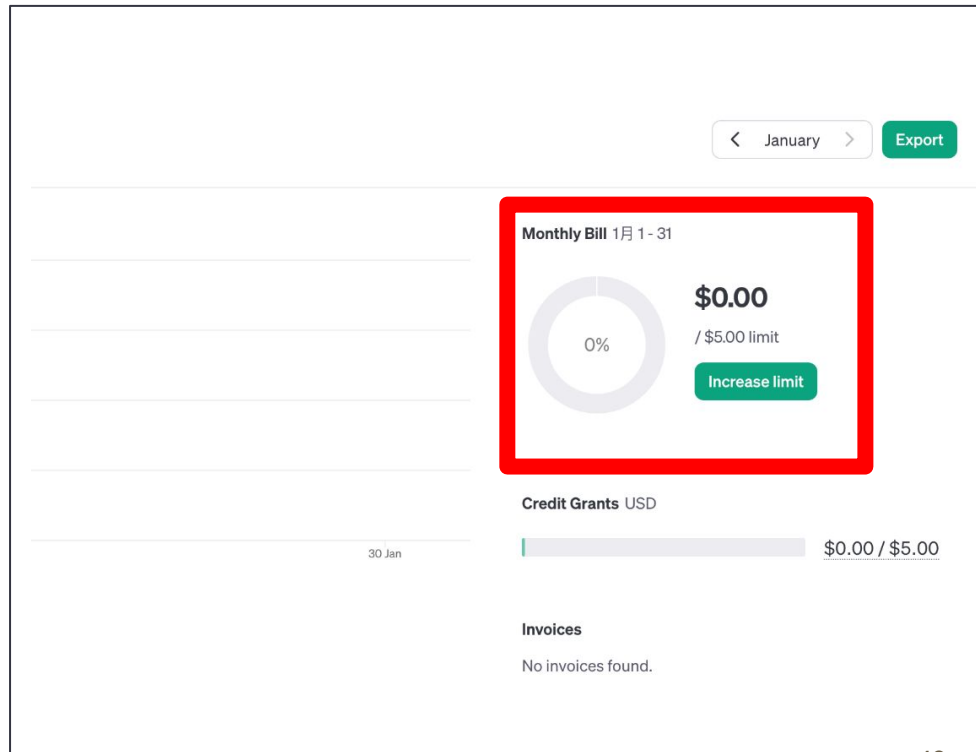
By clicking "Agree", you agree to our [Terms](#) and have read our [Privacy Policy](#)

Agree

Step 3 - Check Usage



The screenshot shows the OpenAI developer portal dashboard. On the left, a sidebar contains navigation links: Playground, Assistants, Fine-tuning, API keys, Files, Usage (highlighted with a red box), and Settings. The main content area is titled 'Welcome to the OpenAI developer portal' and includes a 'Start with the basics' section with links to a 'Quickstart tutorial' and a 'Prompt examples' guide. Below this is a video player for the 'OpenAI DevDay: Opening Keynote'.



The screenshot shows the OpenAI usage page. At the top, there is a navigation bar with a date selector set to 'January' and an 'Export' button. The main content area is divided into two columns. The left column contains a table with headers for 'Usage', 'Model', and 'Created At'. The right column contains a 'Monthly Bill' section for the period '1月 1 - 31'. This section displays a donut chart showing '0%' usage, a balance of '\$0.00' out of a '\$5.00 limit', and an 'Increase limit' button. Below the bill, there is a 'Credit Grants' section showing a bar chart for 'USD' with a value of '\$0.00 / \$5.00'. At the bottom, an 'Invoices' section states 'No invoices found.'

Step 4-1 - Phone Verification

- OpenAI API could probably require phone verification
- Follow the verification instructions



Playground

Assistants

Fine-tuning

API keys

Files

Usage

Settings

Documentation

Help

All products

Fanandy

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 disable any API key that we've found has leaked publicly.

Enable tracking to see usage per API key on the [Usage page](#).

Verify your phone number to create an API key

Start verification

You currently do not have any API keys

Create one using the button below to get started

+ Create new secret key

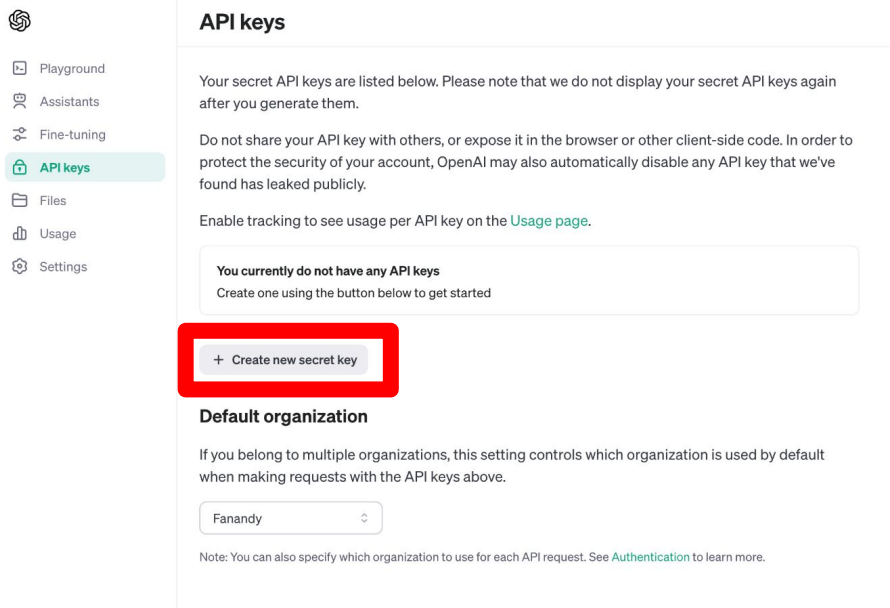
Default organization

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

Fanandy

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

Step 4-2 Create API key



The screenshot shows the OpenAI API keys management page. On the left is a sidebar with icons for Playground, Assistants, Fine-tuning, API keys (highlighted), Files, Usage, and Settings. The main content area is titled "API keys". It contains a warning message: "Your secret API keys are listed below. Please note that we do not display your secret API keys again after you generate them." followed by a security notice: "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 disable any API key that we've found has leaked publicly." Below this is a link to the "Usage page" to enable tracking. A box states "You currently do not have any API keys" with a subtext "Create one using the button below to get started". A button labeled "+ Create new secret key" is highlighted with a red rectangle. At the bottom, the "Default organization" section shows "Fanandy" selected from a dropdown menu, with a note about specifying organizations per request.

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 disable any API key that we've found has leaked publicly.

Enable tracking to see usage per API key on the [Usage page](#).

You currently do not have any API keys
Create one using the button below to get started

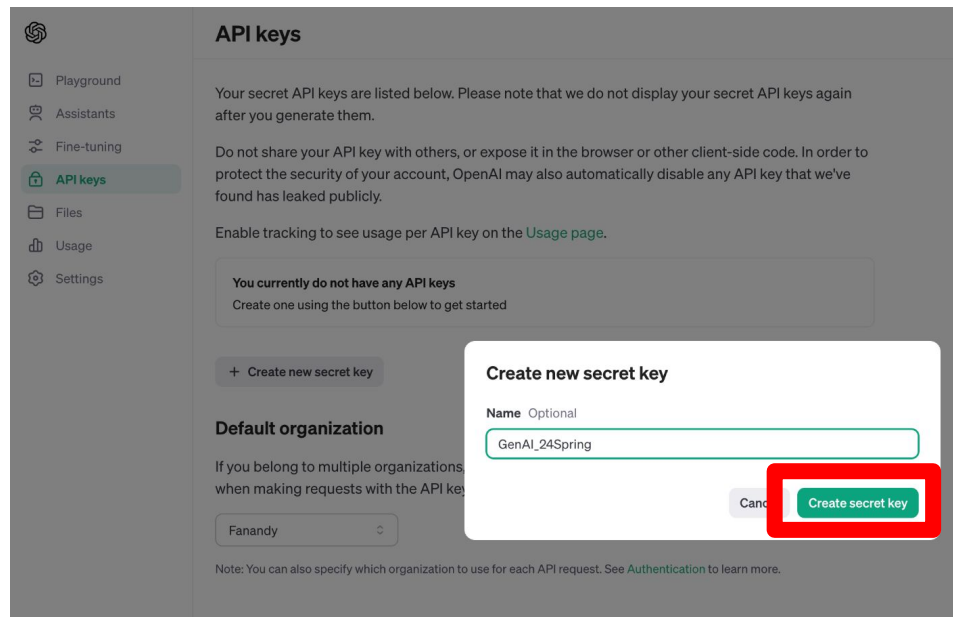
+ Create new secret key

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.

Fanandy

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



This screenshot shows the same OpenAI API keys page as the previous one, but with a modal dialog open for creating a new secret key. The modal is titled "Create new secret key" and has a "Name" field with the value "GenAI_24Spring". At the bottom right of the modal, a button labeled "Create secret key" is highlighted with a red rectangle. The background page is dimmed, showing the same sidebar and API keys section.

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 disable any API key that we've found has leaked publicly.

Enable tracking to see usage per API key on the [Usage page](#).

You currently do not have any API keys
Create one using the button below to get started

+ Create new secret key

Create new secret key

Name Optional
GenAI_24Spring

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.

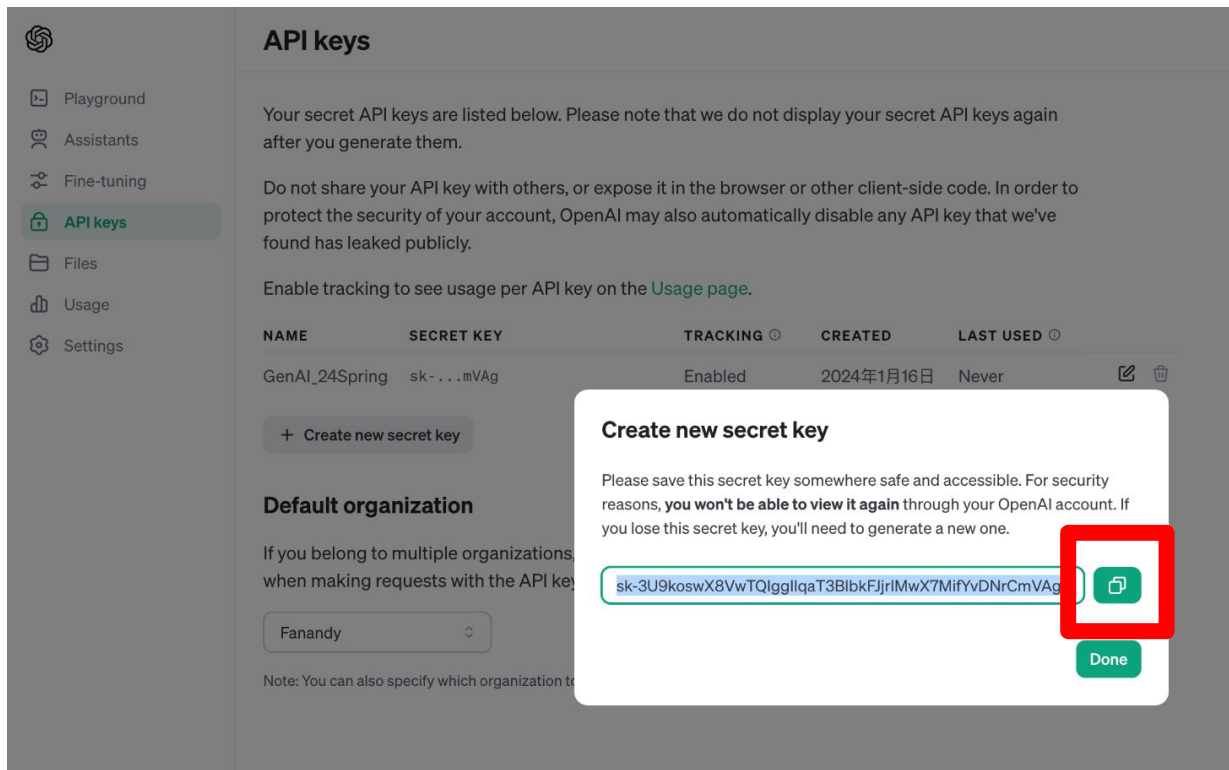
Fanandy

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

Create secret key

Step 4-3 Copy and take good care of your API key !!!

- Please remember to **COPY** your API key, **REMEMBER** it and **KEEP IT SAFE!**
- This is the **only chance** you will see the whole key



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 disable any API key that we've found has leaked publicly.

Enable tracking to see usage per API key on the [Usage page](#).

NAME	SECRET KEY	TRACKING	CREATED	LAST USED
GenAI_24Spring	sk-...mVAg	Enabled	2024年1月16日	Never

[+ Create new secret key](#)

Default organization

If you belong to multiple organizations, when making requests with the API key, you can specify which organization to use.

Fanandy

Note: You can also specify which organization to use in the API key's settings.

Create new secret key

Please save this secret key somewhere safe and accessible. For security reasons, **you won't be able to view it again** through your OpenAI account. If you lose this secret key, you'll need to generate a new one.

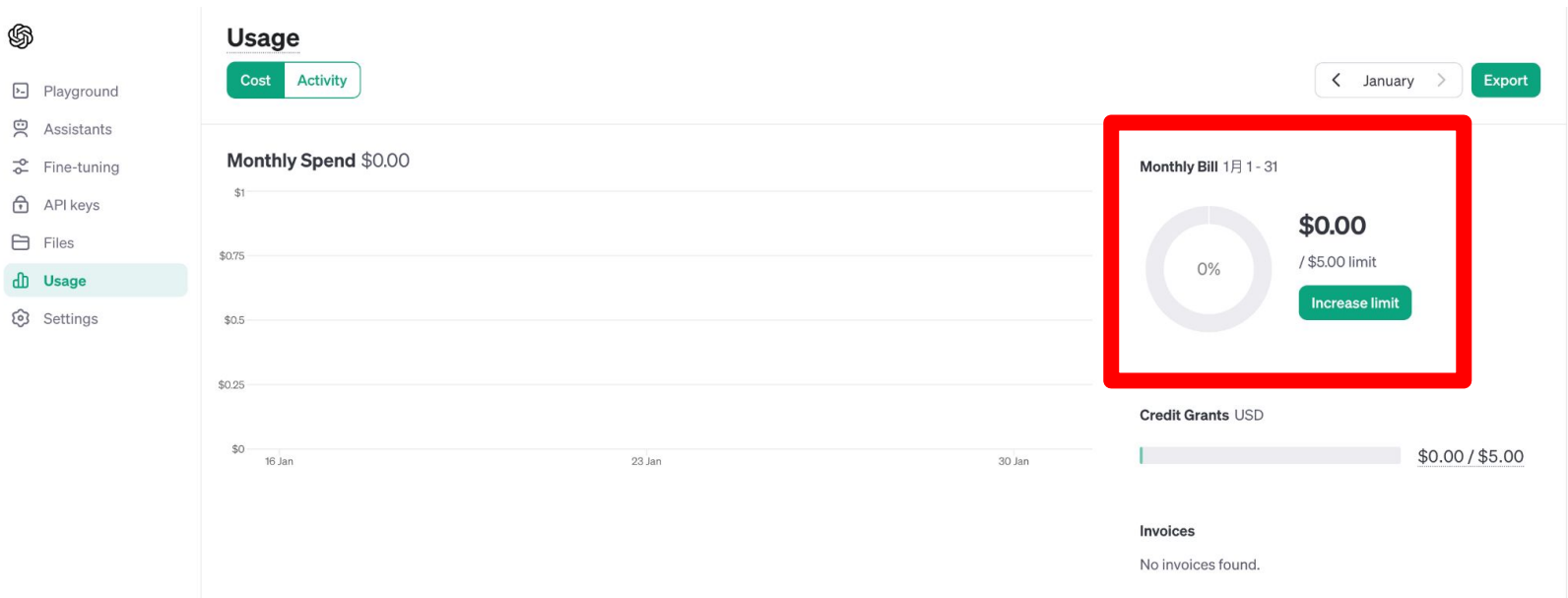
sk-3U9koswX8VwTQlgllqaT3BlbkFJrIMwX7MifYvDNrCmVAg...

[Copy](#)

[Done](#)

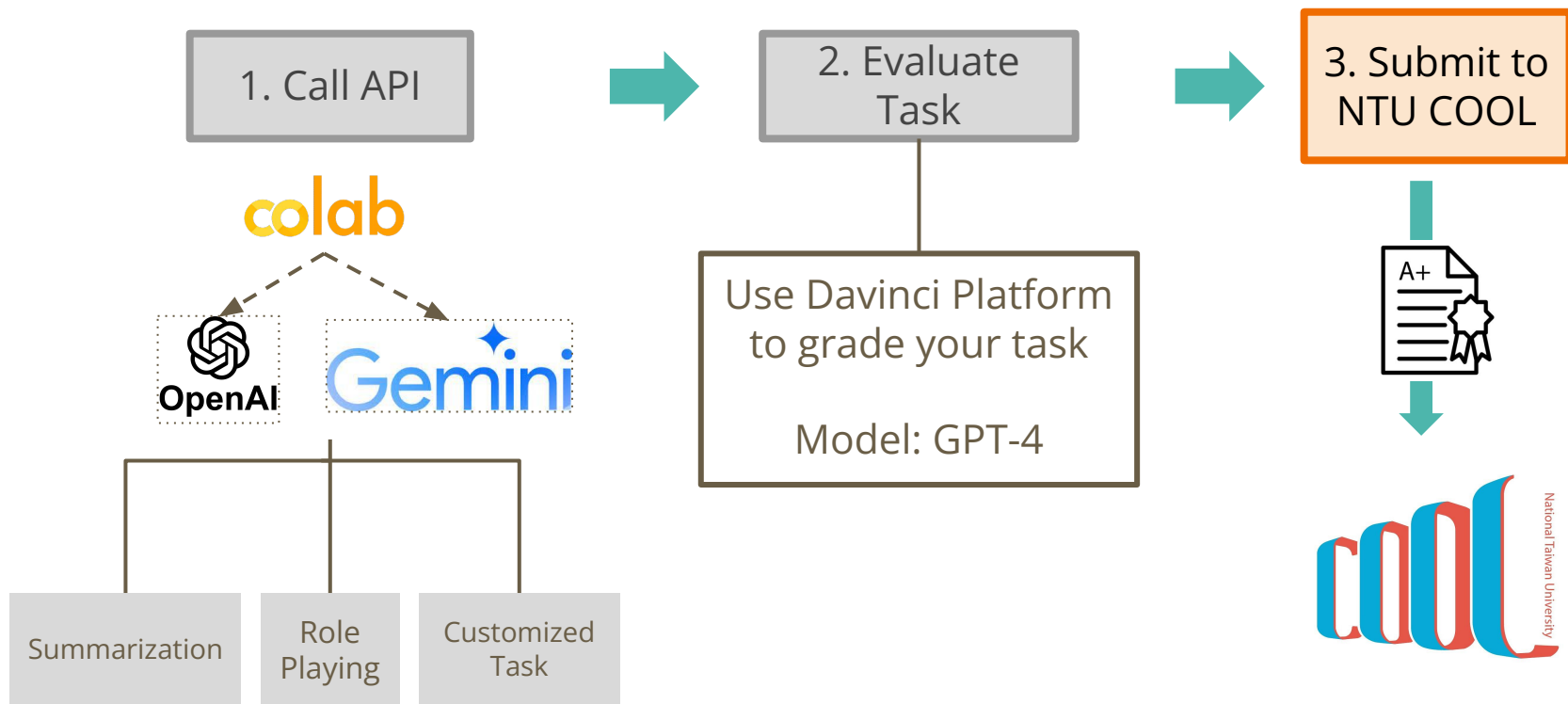
OpenAI API - Free Usage

- Free usage - If you are a new account to OpenAI : Free \$5.00 API usage.
- Otherwise, OpenAI API **costs money!!!**



Task Introduction

Workflow



Task

Objective:

Understand how to build your own Language Model Application by **calling API** and **feeding prompts**.

Task:

1. Summarization (3 points)
2. Role Playing (3 points)
3. Customized Task (4 points)

Colab link to this homework:

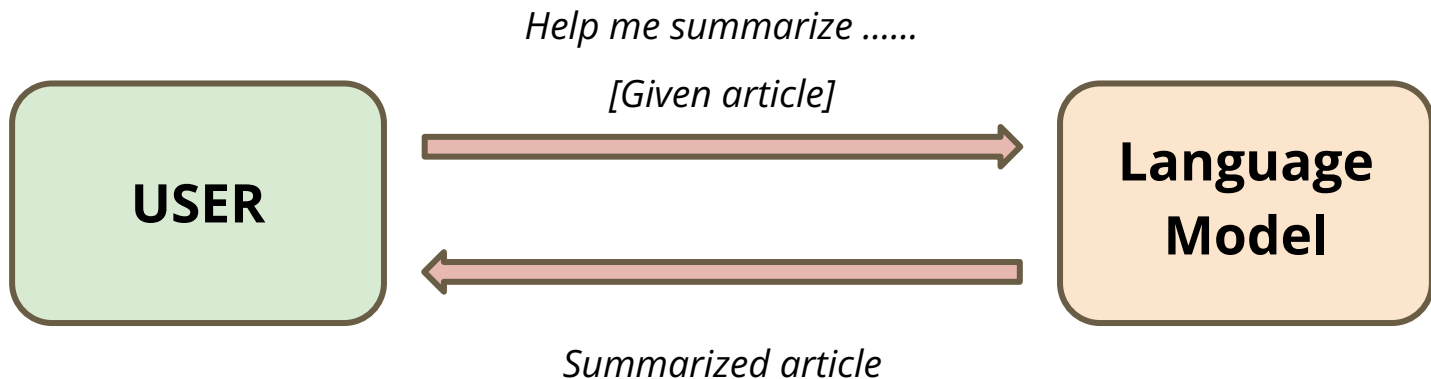
https://colab.research.google.com/drive/15jh4v_TBPsTylBhi0Fz46gEkjvhzGaBR?usp=sharing

Task Introduction:

Task 1: Summarization

Task 1: Summarization

1. We want you to create a bot that can perform article summarization.
2. Design a prompt that enables the language model to summarize the article.





How can I help you today?

Plan a tour
for architectural photography in Barcelona

Suggest fun activities
for a family visiting San Francisco

Brainstorm incentives
for a customer loyalty program in a small bookstore

Come up with concepts
for a retro-style arcade game

Please summarize the following article for me.

With house prices soaring, it's not easy finding somewhere to live. And this community has thrown in the towel. Meet Seattle's rolling neighborhood of RVs, where each unassuming vehicle is a capsule home. The unusual format has been captured in a series of photographs by visual journalist Anna Erickson. Meet Bud Dodson, 57, and welcome to his home: An RV in Seattle's SoDo where he watches over the parking lot in exchange for a spot. No place like home: John Warden, 52, has turned his \$200 vehicle into his home after his apartment burned down years ago. There are around 30 drivers that float in and out of this parking lot in the SoDo (South of Downtown) area



made the RV his home and taken good care of it,' Anna described. 'It was more functional [than John's] and a cleaner space with a bed, kitchen and bathroom.' Whether organized or eclectic, however, each one is home. 'None of them seem to want to move on,' Anna said. 'It's not perfect but they seem pretty content. Move in, move out: Some have agreements to stay, but others have to keep driving around to find a spot. John works as a panhandler at Safeco Fields stadium, where he used to work as a cook. He is content with his life in between the usual confines of society. Personal: To many this may just seem like a parking lot but for these men it is a very personal space. 'Bud is very grateful, he said the parking lot owner is just such a nice guy to let him live like this.' She came across them when she stopped to ask a seemingly homeless man for directions. 'We got talking,' she said, 'and he mentioned that he lived nearby in an RV. I went round to look and there was a whole bunch of them.' Curious, she spent about two months returning to the spot, meeting with the community and building their trust. 'These RVs are their homes so it's a very personal thing,' she explained.

Summarization



ChatGPT

The article explores the community of RV-dwellers in Seattle's SoDo area, where approximately 30 people live in their vehicles. Visual journalist Anna Erickson captured their lifestyle through photographs. The residents, such as Bud Dodson and John Worden, have made their RVs into homes after facing housing challenges, like apartment fires. Dodson maintains the parking lot in exchange for a semi-permanent spot, while Worden works as a panhandler at Safeco Field. Despite the challenges, residents find contentment in their unique homes, whether organized or eclectic. Erickson spent two months building trust with the community to document their personal spaces.



Is this conversation helpful so far?

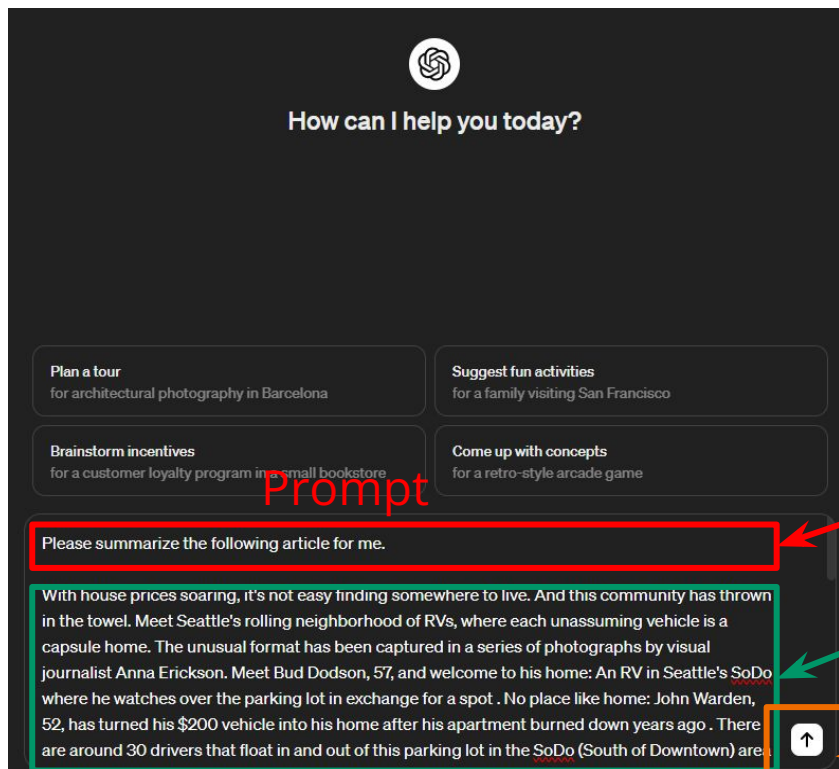
Message ChatGPT...



Article

Send

Browser



Article

Send

60

API Call

```
def interact_summarization(prompt: str, article: str, temp = 1.0) -> List[Tuple[str, str]]:  
    """  
    * Arguments  
    ..... - prompt: the prompt that we use in this section  
    ..... - article: the article to be summarized  
    ..... - temp: the temperature parameter of this model.  
    Temperature is used to control the output of the chatbot.  
    ..... The higher the temperature is, the more creative response you will get.  
    """  
    input = f"{prompt}\n{article}"  
    response = client.chat.completions.create(  
        model="gpt-3.5-turbo",  
        messages = [{'role': 'user', 'content': input}],  
        temperature = temp,  
        max_tokens=200,  
    )  
    return [(input, response.choices[0].message.content)]
```

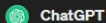
Prompt Article

Send

60

made the RV his home and taken good care of it,' Anna described. 'It was more functional [than John's] and a cleaner space with a bed, kitchen and bathroom.' Whether organized or eclectic, however, each one is home. 'None of them seem to want to move on,' Anna said. 'It's not perfect but they seem pretty content. Move in, move out: Some have agreements to stay, but others have to keep driving around to find a spot. John works as a panhandler at Safeco Fields stadium, where he used to work as a cook. He is content with his life in between the usual confines of society. Personal: To many this may just seem like a parking lot but for these men it is a very personal space. 'Bud is very grateful, he said the parking lot owner is just such a nice guy to let him live like this.' She came across them when she stopped to ask a seemingly homeless man for directions. 'We got talking,' she said, 'and he mentioned that he lived nearby in an RV. I went round to look and there was a whole bunch of them.' Curious, she spent about two months returning to the spot, meeting with the community and building their trust. 'These RVs are their homes so it's a very personal thing,' she explained.

Summarization



ChatGPT

The article explores the community of RV-dwellers in Seattle's SoDo area, where approximately 30 people live in their vehicles. Visual journalist Anna Erickson captured their lifestyle through photographs. The residents, such as Bud Dodson and John Worden, have made their RVs into homes after facing housing challenges, like apartment fires. Dodson maintains the parking lot in exchange for a semi-permanent spot, while Worden works as a panhandler at Safeco Field. Despite the challenges, residents find contentment in their unique homes, whether organized or eclectic. Erickson spent two months building trust with the community to document their personal spaces.



Is this conversation helpful so far?



Message ChatGPT...



Summarization

```
# function to call the model to generate
def interact_summarization(prompt: str, article: str, temp = 1.0) -> List[Tuple[str, str]]:
    """
    * Arguments

    - prompt: the prompt that we use in this section

    - article: the article to be summarized

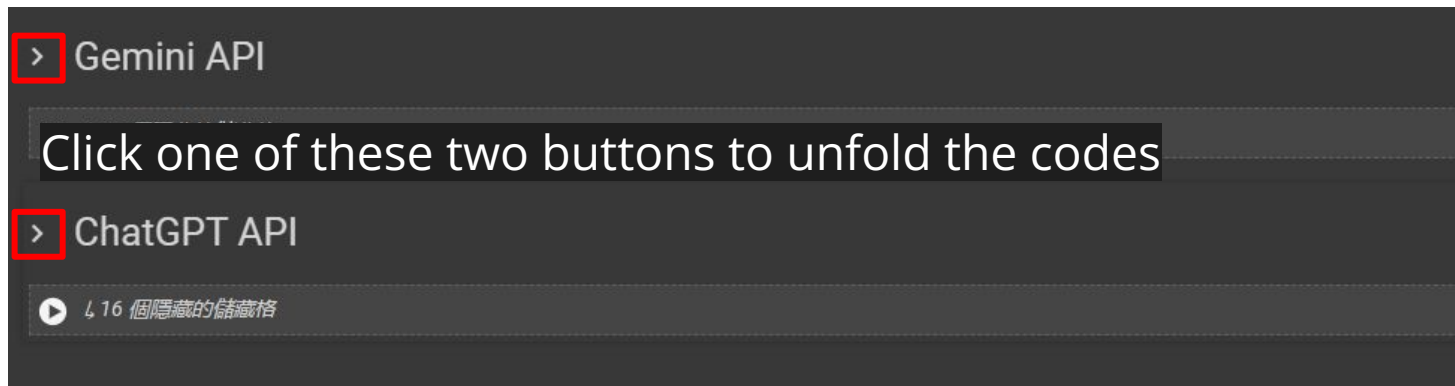
    - temp: the temperature parameter of this model. Temperature is used to control
            The higher the temperature is, the more creative response you will
            get.

    """
    input = f"{prompt}\n{article}"
    response = client.chat.completions.create(
        model="gpt-3.5-turbo",
        messages = [{'role': 'user', 'content': input}],
        temperature = temp,
        max_tokens=200,
    )

    return [(input, response.choices[0].message.content)]
```

Step 0

Choose one of the API you want to use



We will use **ChatGPT API** in the following slides and later demonstrations.

Step 1

1. Run this cell to install the packages

```
▼ Install Packages

Install all the necessary packages, it may take some time.

▶ # install required packages
!pip install openai
!pip install gradio
```

2. Enter your ChatGPT API key here and run this cell

```
▼ Import and Setup

Remember to fill in your OpenAI API in this block, tutorial to get the api is in our homework3 slides. Please make double sure not to share your api with anyone else.

[] # import the packages
import openai
import gradio as gr
import json
from typing import List, Dict, Tuple

## TODO: Fill in your OpenAI api in the "" part
OPENAI_API_KEY = ""
client = openai.OpenAI(api_key=OPENAI_API_KEY)

# Check if you have set your ChatGPT API successfully
# You should see "Set ChatGPT API successfully!!" if nothing goes wrong.
try:
    response = client.chat.completions.create(
        model="gpt-3.5-turbo",
        messages=[{"role": "user", "content": "test"}],
        max_tokens=1,
    )
    print("Set ChatGPT API successfully!!")
except:
    print("There seems to be something wrong with your ChatGPT API. Please follow our demonstration in the slide to get a correct one.")
```

Fill in your ChatGPT API here 

Remember to run these two cells whenever you reconnect to colab.

Step 2

Under this text cell Part1: Summarization

▼ Part 1: Summarization

In this task, you are asked to prompt your chatbot into a **summarizer**. Its job is when the user inputs an article, it can summarize the article for the user.

You need to:

1. Come up a prompt for summarization and fill it in `prompt_for_summarization`.
2. Hit the run button . (The run button will turn into this state  when successfully executed.) It will pop up an interface that looks like this: (It may look a little bit different if you use dark mode.)

Setting queue=True in a Colab notebook requires sharing enabled. Setting 'share=True' (you can turn this off by setting 'share=False' in 'launch()') explicitly).

Fill in your prompt for summarization in "FILL IN YOUR PROMPT" and **hit the run button**.

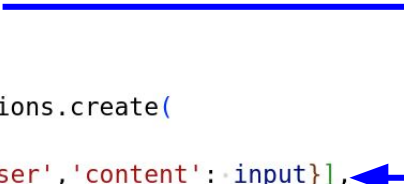
```
## TODO: Input the prompt in the prompt_for_summarization = "FILL IN YOUR PROMPT"  
  
# function to reset the conversation  
def reset() -> List:  
    return []
```

Example: "Please summarize this article for me."

Step 2

The prompt you just filled in will be used in the API call to instruct ChatGPT what it should do

```
def interact_summarization(prompt: str, article: str, temp = 1.0) -> List[Tuple[str, str]]:
    """
    * Arguments
    ..... - prompt: the prompt that we use in this section
    ..... - article: the article to be summarized
    ..... - temp: the temperature parameter of this model.
               Temperature is used to control the output of the chatbot.
    ..... | ..... The higher the temperature is, the more creative
               response you will get.
    .....
    .... input = f"{prompt}\n{article}"
    .... response = client.chat.completions.create(
    ..... | .... model="gpt-3.5-turbo",
    ..... | .... messages = [{'role': 'user', 'content': input}],
    ..... | .... temperature = temp,
    ..... | .... max_tokens=200,
    .... )
    .... return [(input, response.choices[0].message.content)]
```



Step 3

There will be a gradio interface like this pop up.

Part1: Summarization

Fill in any article you like and let the chatbot summarize it for you!!

Chatbot

Article

With house prices soaring, it's not easy finding somewhere to live. And this community has thrown in the towel. Meet Seattle's rolling neighborhood of RVs, where each unassuming vehicle is a capsule home. The unusual format has been captured in a series of photographs by visual journalist Anna Erickson. Meet Bud Dodson, 57, and welcome to his home: An RV in Seattle's SoDo where he watches over the parking lot in exchange for a spot. No place like home: John Warden, 52, has turned his \$200 vehicle into his home after his apartment burned down years ago. There are around 30 drivers that float in and out of this parking lot in the SoDo (South of downtown) area of the city in Washington State. One might not notice them in the mornings as hundreds of workers in the nearby factories, such as Starbucks, park up and rush into work. But on the weekends, as the rabble flocks back to their beds, this unique group remains. John Warden, 52, has been living in his vehicle for years since his apartment burned down and he was left homeless. He told Anna his car cost \$200, and doesn't drive very well. But for a home, it's just about enough. Though plan on the outside, it is a Pandora's Box inside, Anna tells DailyMail.com. 'It was scattered with trinkets that he had been collecting over the years,' she explained, 'and a pile of beer cans that he was saving to turn in for money.' For work, he panhandles while helping people find parking spaces at Safeco Field stadium, where he used to be a cook. People come and go for work in the factories nearby, but on the weekend it is just the RV-dwellers that area left. Daily life: Here Bud can be seen preparing himself a barbecue on the gravel outside his capsule home, one of about 30 in the community. Eclectic: While Bud's RV is organized and functional, John's is full of trinkets and belongings dating back years. Alongside him - most of the time - is Bud Dodson, 57. While some are forced to move about regularly, Dodson, a maintenance man, looks after the parking lot in exchange for a semi-permanent spot. His home has its own unique stamp on it. 'He had really made the RV his home and taken good care of it,' Anna described. 'It was more functional [than John's] and a cleaner space with a bed, kitchen and bathroom.' Whether organized or eclectic, however, each one is home. 'None of them seem to want to move on,' Anna said. 'It's not perfect but they seem pretty content. Move in, move out: Some have agreements to stay, but others have to keep driving around to find a spot. John works as a panhandler at Safeco Fields stadium, where he used to work as a cook. He is content with his life in between the usual confines of society. Personal: To many this may just seem like a parking lot but for these men it is a very personal space. 'Bud is very grateful, he said the parking lot owner is just such a nice guy to let him live like this.' She came across them when she stopped to ask a seemingly homeless man for directions. 'We got talking,' she said, 'and he mentioned that he lived nearby in an RV. I went round to look and there was a whole bunch of them.' Curious, she spent about two months returning to the spot, meeting with the community and building their trust. 'These RVs are their homes so it's a very personal thing,' she explained.

Temperature

Here you can enter whatever article you want to summarize

Step 3

The article will also be a part of the input to ChatGPT

```
def interact_summarization(prompt: str, article: str, temp = 1.0) -> List[Tuple[str, str]]:
    """
    * Arguments

    - prompt: the prompt that we use in this section

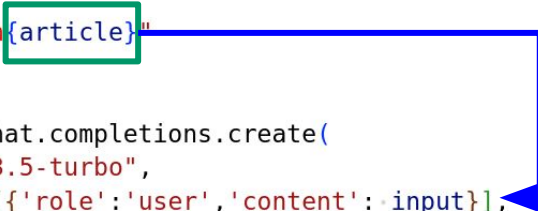
    - article: the article to be summarized

    - temp: the temperature parameter of this model.
      Temperature is used to control the output of the chatbot.
      The higher the temperature is, the more creative response you will get.

    """
    input = f"{prompt}\n{article}"

    response = client.chat.completions.create(
        model="gpt-3.5-turbo",
        messages=[{'role': 'user', 'content': input}],
        temperature=temp,
        max_tokens=200,
    )

    return [(input, response.choices[0].message.content)]
```



Step 4

Article

With house prices soaring, it's not easy finding somewhere to live. And this community has thrown in the towel. Meet Seattle's rolling neighborhood of RVs, where each unassuming vehicle is a capsule home. The unusual format has been captured in a series of photographs by visual journalist Anna Erickson. Meet Bud Dodson, 57, and welcome to his home: An RV in Seattle's SoDo where he watches over the parking lot in exchange for a spot. No place like home: John Warden, 52, has turned his \$200 vehicle into his home after his apartment burned down years ago. There are around 30 drivers that float in and out of this parking lot in the SoDo (South of Downtown) area of the city in Washington State. One might not notice them in the mornings as hundreds of workers in the nearby factories, such as Starbucks, park up and rush into work. But on the weekends, as the rabble flocks back to their beds, this unique group remains. John Worden, 52, has been living in his vehicle for years since his apartment burned down and he was left homeless. He told Anna his car cost \$200, and doesn't drive very well. But for a home, it's just about enough. Though plan on the outside, it is a Pandora's Box inside, Anna tells DailyMail.com. 'It was scattered with trinkets that he had been collecting over the years,' she explained, 'and a pile of beer cans that he was saving to turn in for money.' For work, he panhandles while helping people find parking spaces at Safeco Field stadium, where he used to be a cook. People come and go for work in the factories nearby, but on the weekend it is just the RV-dwellers that area left. Daily life: Here Bud can be seen preparing himself a barbecue on the gravel outside his capsule home, one of about 30 in the community. Eclectic: While Bud's RV is organized and functional, John's is full of trinkets and belongings dating back years. Alongside him - most of the time - is Bud Dodson, 57. While some are forced to move about regularly, Dodson, a maintenance man, looks after the parking lot in exchange for a semi-permanent spot. His home has its own unique stamp on it. 'He had really made the RV his home and taken good care of it,' Anna described. 'It was more functional [than John's] and a cleaner space with a bed, kitchen and bathroom.' Whether organized or eclectic, however, each one is home. 'None of them seem to want to move on,' Anna said. 'It's not perfect but they seem pretty content. Move in, move out: Some have agreements to stay, but others have to keep driving around to find a spot. John works as a panhandler at Safeco Fields stadium, where he used to work as a cook. He is content with his life in between the usual confines of society. Personal: To many this may just seem like a parking lot but for these men it is a very personal space. 'Bud is very grateful, he said the parking lot owner is just such a nice guy to let him live like this.' She came across them when she stopped to ask a seemingly homeless man for directions. 'We got talking,' she said, 'and he mentioned that he lived nearby in an RV. I went round to look and there was a whole bunch of them.' Curious, she spent about two months returning to the spot, meeting with the community and building their trust. 'These RVs are their homes so it's a very personal thing,' she explained.

Temperature

Temperature is used to control the output of the chatbot. The higher the temperature is, the more creative response you will get.

Temperature 1

Send **Reset**

Hit "Send" to get the response.

Export

You can use this slider to adjust the temperature here. The higher the temperature, the more creative response you will get.

Step 5

The response will display in gradio like this

Part1: Summarization

Fill in any article you like and let the chatbot summarize it for you!!

Chatbot

dating back years . Alongside him - most of the time - is Bud Dodson, 57. While some are forced to move about regularly, Dodson, a maintenance man, looks after the parking lot in exchange for a semi-permanent spot. His home has its own unique stamp on it. 'He had really made the RV his home and taken good care of it,' Anna described. 'It was more functional [than John's] and a cleaner space with a bed, kitchen and bathroom.' Whether organized or eclectic, however, each one is home. 'None of them seem to want to move on,' Anna said. 'It's not perfect but they seem pretty content. Move in, move out: Some have agreements to stay, but others have to keep driving around to find a spot. John works as a panhandler at Safeco Fields stadium, where he used to work as a cook. He is content with his life in between the usual confines of society. Personal: To many this may just seem like a parking lot but for these men it is a very personal space. 'Bud is very grateful, he said the parking lot owner is just such a nice guy to let him live like this.' She came across them when she stopped to ask a seemingly homeless man for directions. 'We got talking,' she said, 'and he mentioned that he lived nearby in an RV. I went round to look and there was a whole bunch of them.' Curious, she spent about two months returning to the spot, meeting with the community and building their trust. 'These RVs are their homes so it's a very personal thing,' she explained.

Capturing the essence of Seattle's unique rolling neighborhood of RVs, Photographer Anna Erickson delves into the lives of the residents who have made these unassuming vehicles their homes. Meet Bud Dodson, 57, a maintenance man who looks after the parking lot in exchange for a spot for his RV. While across from him, John Worden, 52, has turned his \$200 vehicle into his home after his apartment burned down. In this community of around 30 residents in Seattle's SoDo area, each RV has its own story and personal touch. Take a closer look at how these individuals have found contentment and comfort within the confines of their unconventional homes.

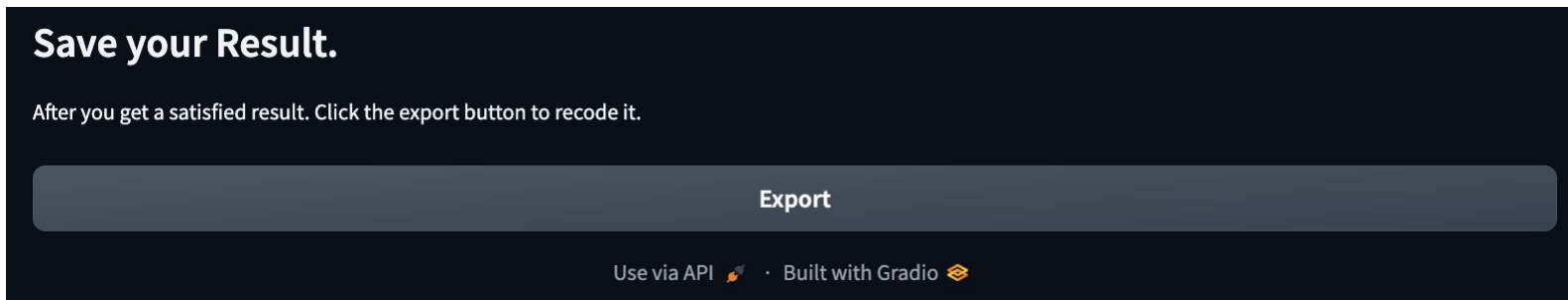
Article

With house prices soaring, it's not easy finding somewhere to live. And this community has thrown in the towel. Meet Seattle's rolling neighborhood of RVs, where each unassuming vehicle is a capsule home. The unusual format has been captured in a series of photographs by visual journalist Anna Erickson. Meet Bud Dodson, 57, and welcome to his home: An RV in Seattle's SoDo where he watches over the parking lot in exchange for a spot. No place like home: John Warden, 52, has turned his \$200 vehicle into his home after his apartment burned down years ago. There are around 30 drivers that float in and out of this parking lot in the SoDo (South of Downtown) area of the city in Washington State. One might not notice them in the

Save your results

For each tasks, after you get a satisfied result. Make sure to hit the **“Export”** button.

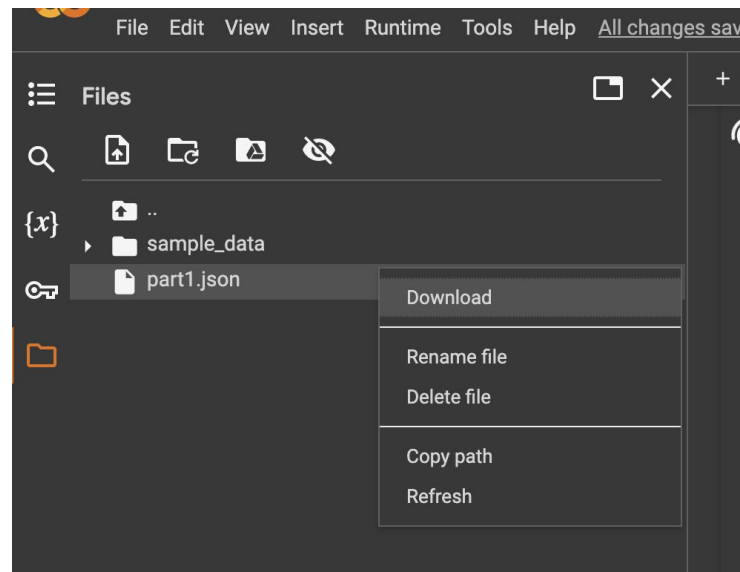
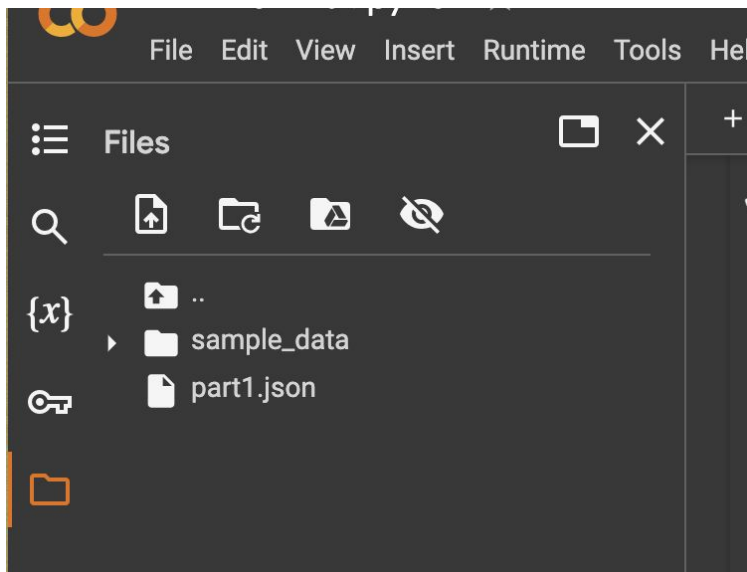
If you hit the “Export” button again, the old file will be replaced.



Save your results

After that, you will see the file appeared in the file list.

Make sure to download it to your own computer before it disappear.



After you save your results ...

Go to “Check and print your result” for the corresponding part

```
▶ with open("part1.json", "r") as f:
    context = json.load(f)

chatbot = context['chatbot']
article = context['article']
summarization = chatbot[0][-1]

with gr.Blocks() as demo:
    gr.Markdown("# Part1: Summarization\nFill in any article you like and let the chatbot summarize it for you!")
    chatbot = gr.Chatbot(height=800, value = context['chatbot'])
    article_textbox = gr.Textbox(label="Article", interactive = False, value = context['article'])
    with gr.Column():
        gr.Markdown("# Copy this part to the grading system.")
        gr.Textbox(label="article", value = article, show_copy_button = True)
        gr.Textbox(label="summarization", value = summarization, show_copy_button = True)
```

RUN IT

Copy this part to the grading system.

article

With house prices soaring, it's not easy finding somewhere to live. And this community has thrown in the towel. Meet Seattle's rolling neighborhood of RVs, where each unassuming vehicle is a capsule home. The unusual format has been captured in a series of photographs by visual journalist Anna Erickson. Meet Bud Dodson, 57, and welcome to his home: An RV in Seattle's SoDo where he watches over the parking lot in exchange for a spot. No place like home: John Warden, 52, has turned his \$200 vehicle into his home after his apartment burned down years ago. There are around 30 drivers that float in and out of this parking lot in the SoDo (South of Downtown) area of the city in Washington State. One might not notice them in the mornings as hundreds of workers in the nearby factories, such as Starbucks, park up and rush into work. But on the weekends, as the rabble flocks back to their beds, this unique group remains. John Worden, 52, has been living in his vehicle for years since his apartment burned down and he was left homeless. He told Anna his car cost \$200, and doesn't drive very well.

summarization

****Summary****

In Seattle, a community of individuals has found solace and shelter in a collection of RVs parked in a SoDo (South of Downtown) parking lot. These vehicles have become their homes, each with its own unique character and story.

****Individuals Featured****

*****Bud Dodson, 57:**** A maintenance man who watches over the parking lot in exchange for a semi-permanent spot in his RV.

*****John Worden, 52:**** A former cook who has lived in his \$200 RV since his apartment burned down years ago.

****Community Dynamics****

Copy
Paste



NTU GenAI 2024 Spring HW3 Auto Grading System

Auto Grading System for NTU GenAI 2024 Spring HW3.

Summarization Grader

This chatbot would use the given article and the correspond summarization to grade your score.

article

Enter a value for article...

summarization

Enter a value for summarization...

Submit

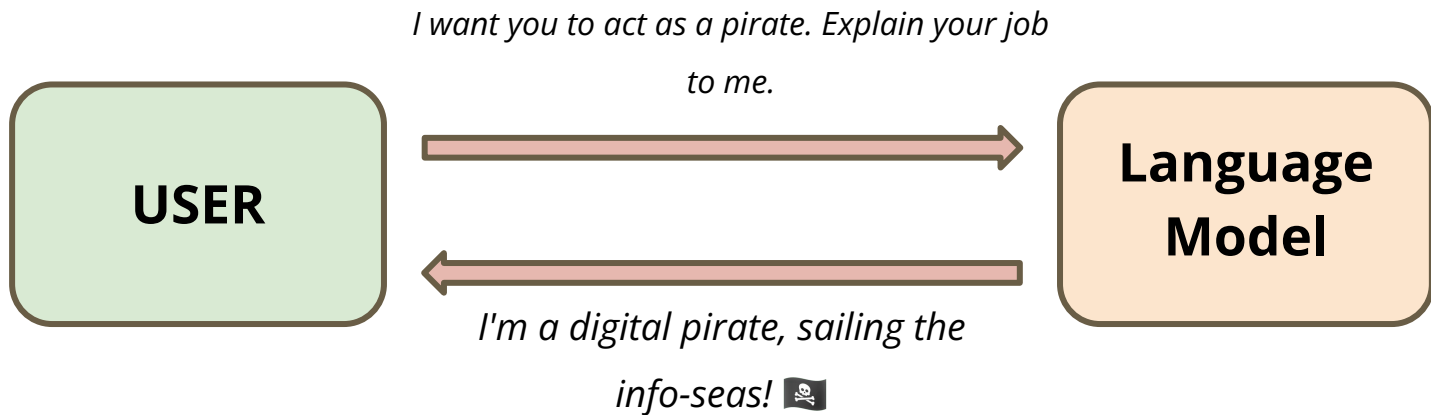
Demo for Task 1

Task Introduction:

Task 2: Role Playing

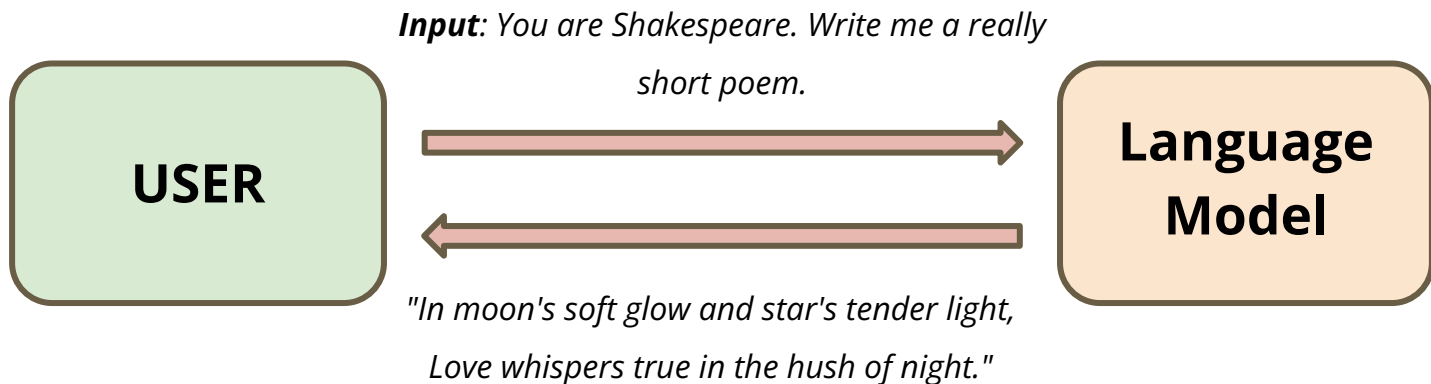
Task 2: Role Playing

- We want you to design a bot service that you can **play a role play game** with the LM.
- Engage in a **multi-turn conversation** with the LM
- **Prompt:** Tell the chatbot to play whatever character.
- **Later inputs:** Interact with the chatbot.

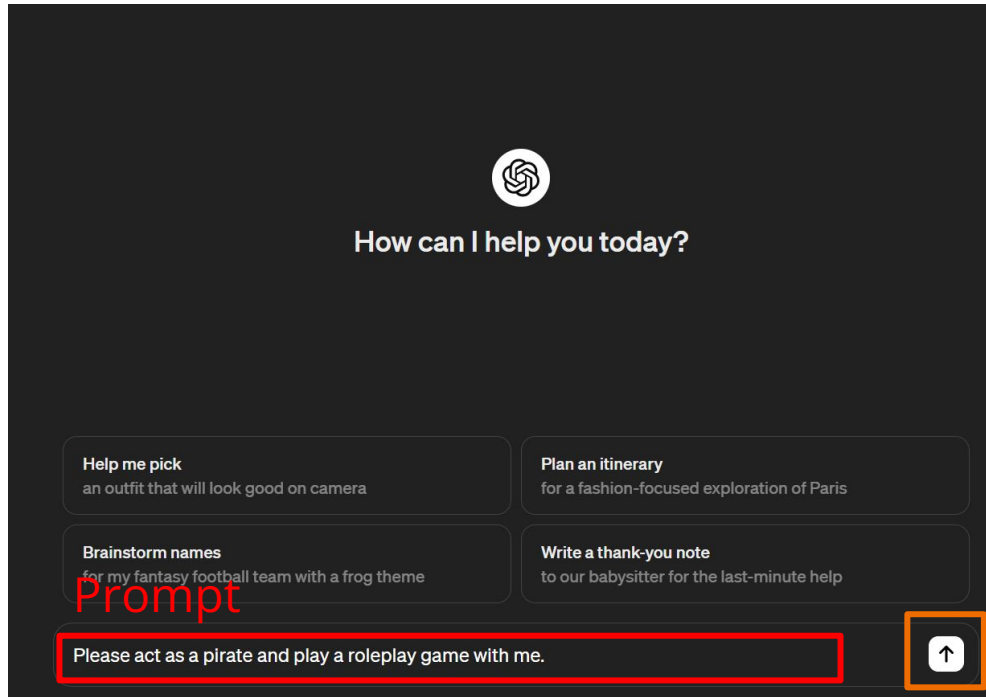


Rules: Role Play

1. Communicate with the LM for **2 rounds (excluding the prompt)**
 - a. User → LM, LM Response (這樣算一次)

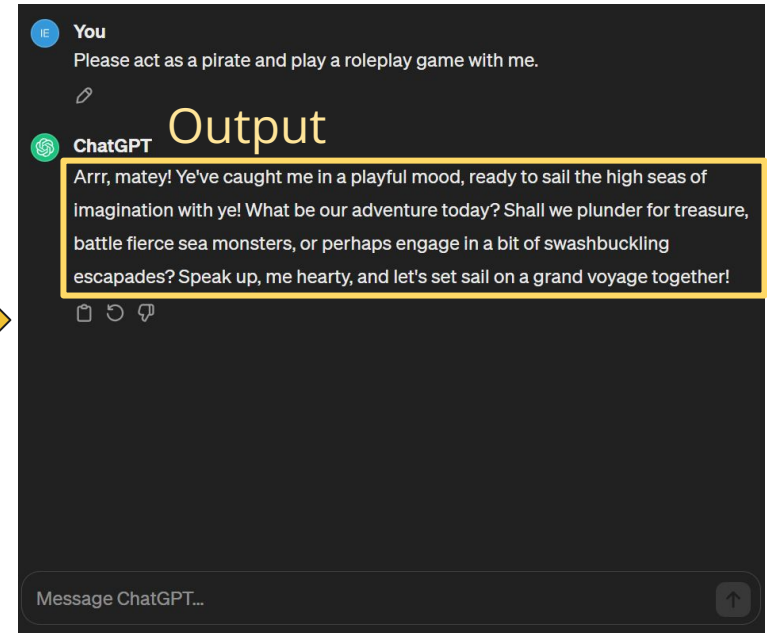
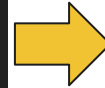


Note: Observe the multi-turn conversation (history of conversation recorded)



Prompt

Send



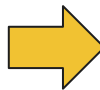
Output

You
Please act as a pirate and play a roleplay game with me.

ChatGPT
Arrr, matey! Ye've caught me in a playful mood, ready to sail the high seas of imagination with ye! What be our adventure today? Shall we plunder for treasure, battle fierce sea monsters, or perhaps engage in a bit of swashbuckling escapades? Speak up, me hearty, and let's set sail on a grand voyage together!

History

New Input
Let's search for the treasures!!



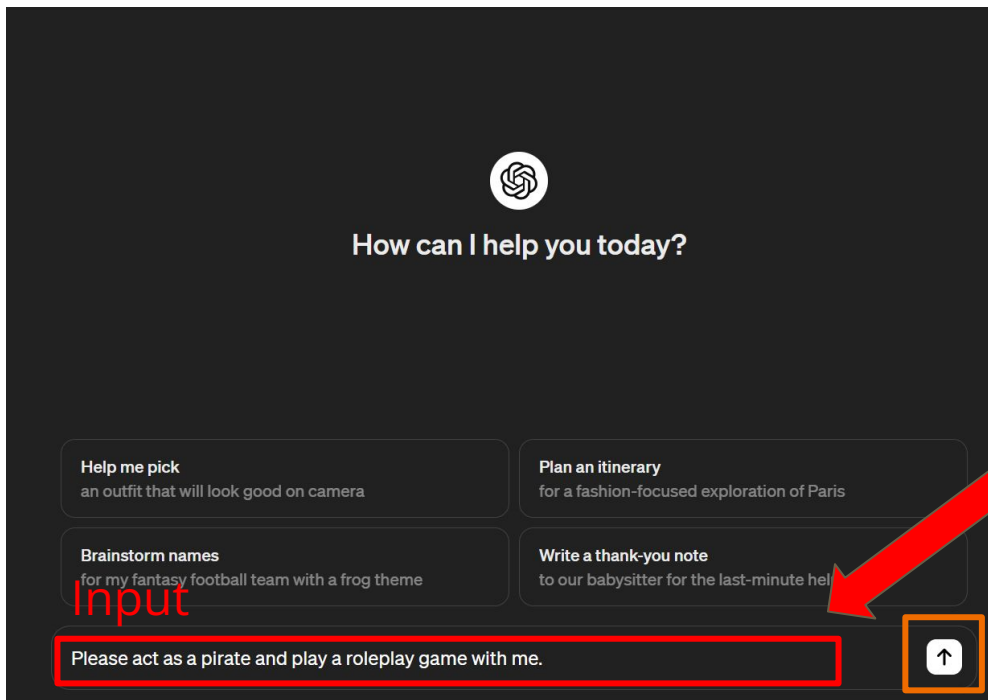
ChatGPT
Arrr, matey! Ye've caught me in a playful mood, ready to sail the high seas of imagination with ye! What be our adventure today? Shall we plunder for treasure, battle fierce sea monsters, or perhaps engage in a bit of swashbuckling escapades? Speak up, me hearty, and let's set sail on a grand voyage together!

You
Let's search for the treasures!!

ChatGPT
Arrr, a fine choice indeed, me hearty! Hoist the sails and prepare the crew, for we're off to seek the legendary treasure buried on a distant island! But beware, matey, for the journey ahead be fraught with peril and mystery.

As we navigate through treacherous waters and brave the storms that rage

Message ChatGPT...



Input

Send

```
def interact_roleplay(chatbot: List[Tuple[str, str]],
user_input: str, temp=1.0) -> List[Tuple[str, str]]:
    """
    ...* Arguments
    ...- user_input: the user input of each round of
        conversation
    ...- temp: the temperature parameter of this model.
        Temperature is used to control the output of the chatbot.
        The higher the temperature is, the more creative
        response you will get.
    ...
    """
    try:
        messages = []
        for input_text, response_text in chatbot:
            messages.append({'role': 'user', 'content':
                input_text})
            messages.append({'role': 'assistant', 'content':
                response_text})
        messages.append({'role': 'user', 'content':
            user_input})

        response = client.chat.completions.create(
            model="gpt-3.5-turbo",
            messages = messages,
            temperature = temp,
            max_tokens=200,
        )

        chatbot.append((user_input, response.choices[0].
            message.content))

    except Exception as e:
        print(f"Error occurred: {e}")
        chatbot.append((user_input, f"Sorry, an error
            occurred: {e}"))
    return chatbot
```

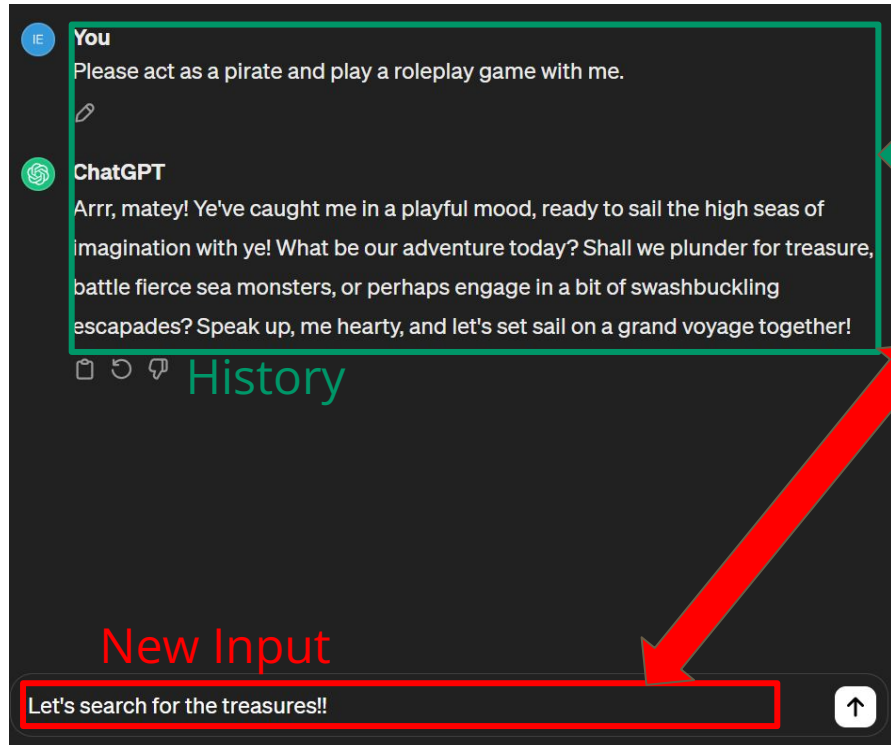
Input

Send

History

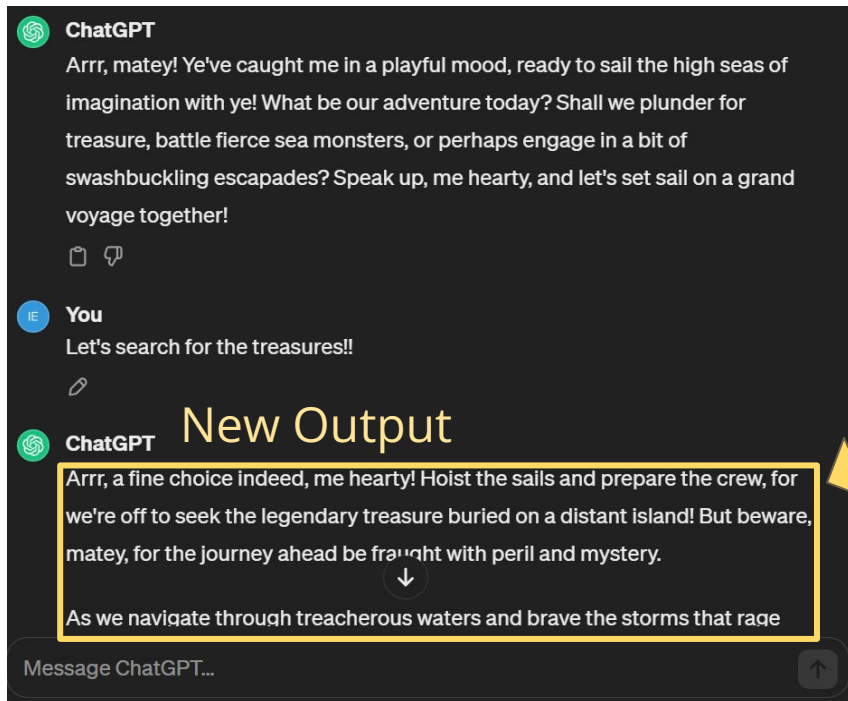
user: The input you typed

assistant: The output the bot gener



```
messages = []  
for input_text, response_text in chatbot:  
    messages.append({'role': 'user', 'content':  
        input_text})  
    messages.append({'role': 'assistant', 'content':  
        response_text})  
  
messages.append({'role': 'user', 'content':  
    user_input})  
  
response = client.chat.completions.create(  
    model="gpt-3.5-turbo",  
    messages = messages,  
    temperature = temp,  
    max_tokens=200,  
)  
  
chatbot.append((user_input, response.choices[0].  
    message.content))
```

When using API, you have to record the history manually.



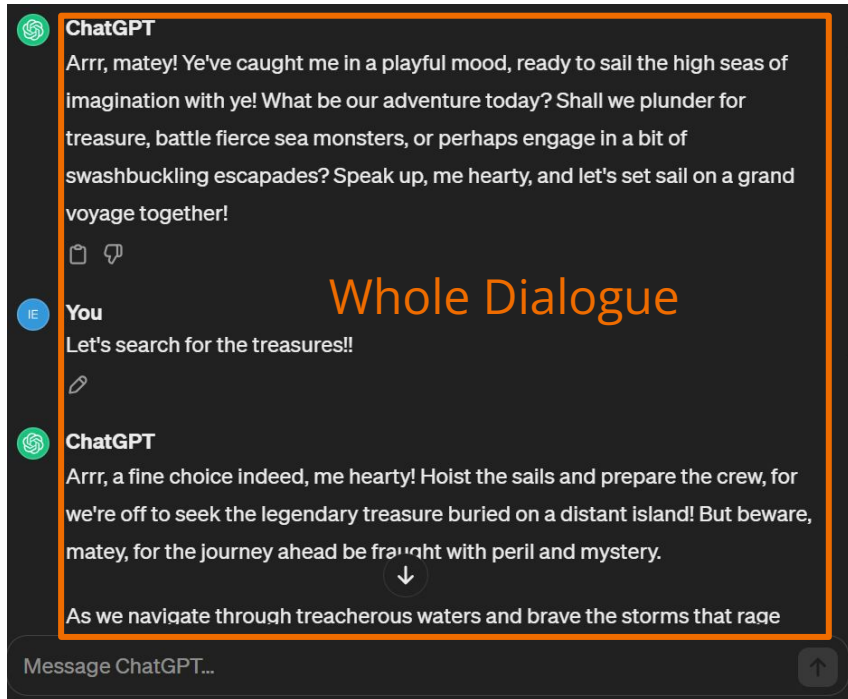
```
messages = []
for input_text, response_text in chatbot:
    messages.append({'role': 'user', 'content': input_text})
    messages.append({'role': 'assistant', 'content': response_text})

messages.append({'role': 'user', 'content': u' New Output'})

response = client.chat.completions.create(
    model="gpt-3.5-turbo",
    messages = messages,
    temperature = temp,
    max_tokens=200,
)

chatbot.append((user_input, response.choices[0].message.content))
```

When using API, you have to record the history manually.



```

messages = []
for input_text, response_text in chatbot:
    messages.append({'role': 'user', 'content':
        input_text})
    messages.append({'role': 'assistant', 'content':
        response_text})

messages.append({'role': 'user', 'content':
    user_input})

response = client.chat.completions.create(
    model="gpt-3.5-turbo",
    messages = messages,
    temperature = temp,
    max_tokens=200,
)

chatbot.append((user_input, response.choices[0].
    message.content))

```

Whole Dialogue

When using API, you have to record the history manually.

Note

```
# TODO: Fill in the below two lines: character_for_chatbot and  
prompt_for_roleplay  
# The first one is the character you want your chatbot to play  
# The second one is the prompt to make the chatbot be a  
certain character  
character_for_chatbot = "FILL IN YOUR CHARACTER"  
prompt_for_roleplay = "FILL IN YOUR PROMPT"
```

Role: Fill in the role you want and it will be in the red block below

Input: Actual Input for conversation.

Which character do you want your chatbot to play?

Input

Demo for Task 2

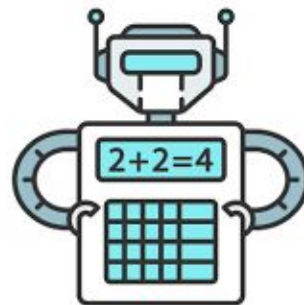
Task Introduction:

Task 3: Customize Task

Task 3: Customize Task

- We want you to create a customized service bot.
- You can think of any tasks that you want the Bot to do. For example,
 - A bot that can solve simple math problems
 - A bot that always output the antonym of the user's input word
 - Any creative tasks that you can think of

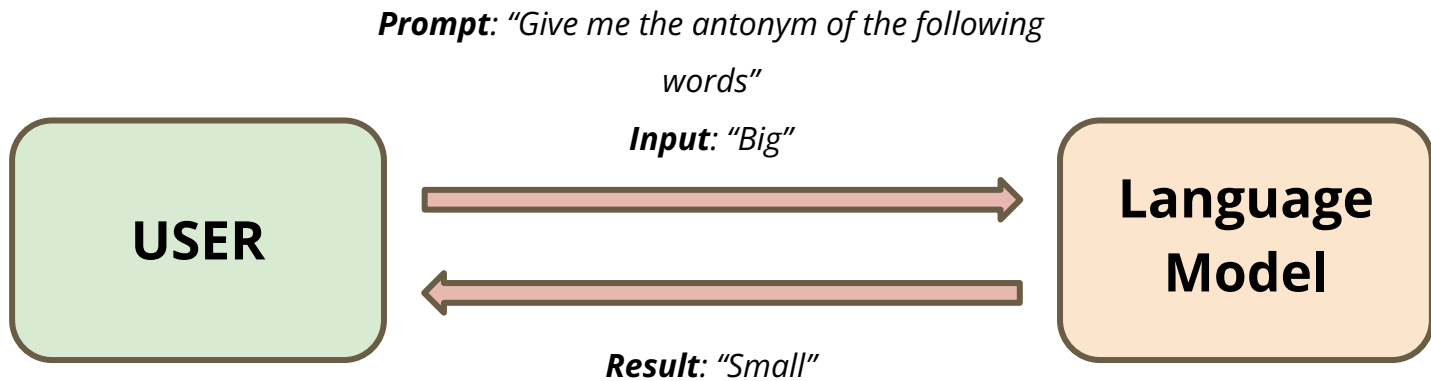
Be Creative!

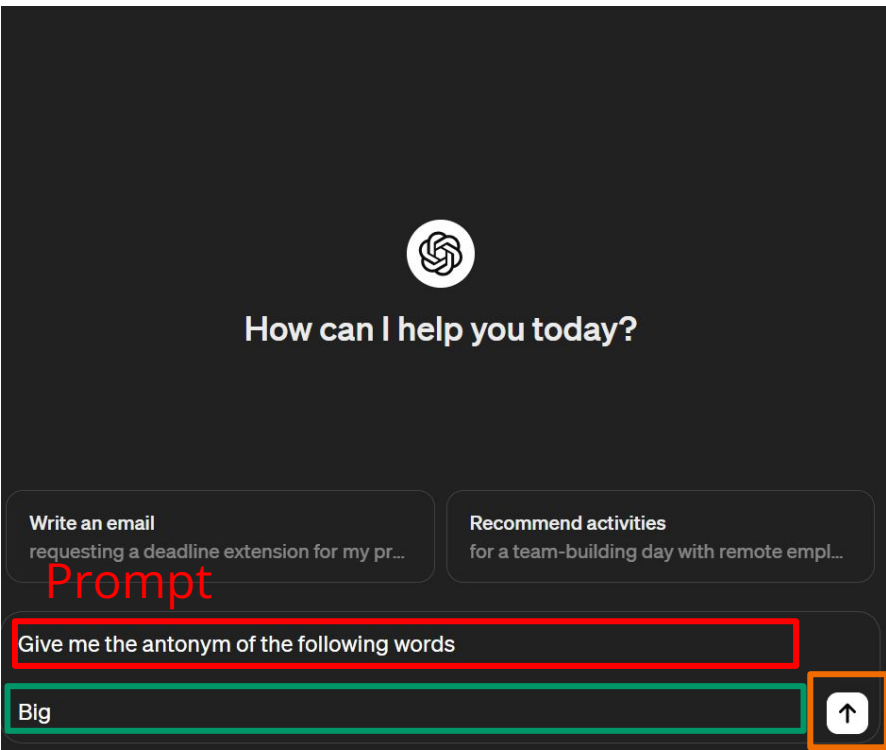


Calculation

Task 3: Customize Task

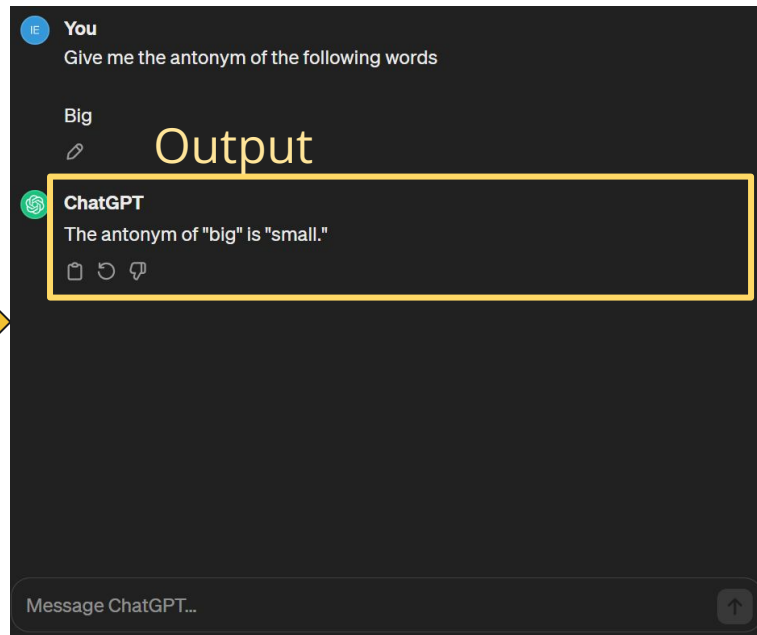
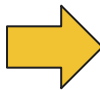
1. Enter a **prompt** for LLM to perform a certain “Task” (Do not use tasks that will produce too long outputs, specifically 1k tokens).
2. Enter the **input** of “Task”
3. Single or multiple rounds are both ok, but do not exceed 3 rounds.
4. Export the conversation log





Input

Send





How can I help you today?

Write an email
requesting a deadline extension for my pr...

Recommend activities
for a team-building day with remote empl...

Prompt

Give me the antonym of the following words

Big

Input



Send

```
def interact_customize(chatbot: List[Tuple[str, str]], prompt: str, user_input: str, temperature = 1.0) -> List[Tuple[str, str]]:
    """
    ...*. Arguments
    .....-- chatbot: the model itself, the conversation is stored in list of tuples
    .....-- prompt: the prompt for your designated task
    .....-- user_input: the user input of each round of conversation
    .....-- temp: the temperature parameter of this model. Temperature is used to control the output of the chatbot. The higher the temperature is, the more creative response you will get.
    ..."""
    try:
        messages = []
        messages.append({'role': 'user', 'content': prompt})
        for input_text, response_text in chatbot:
            messages.append({'role': 'user', 'content': input_text})
            messages.append({'role': 'assistant', 'content': response_text})
        messages.append({'role': 'user', 'content': user_input})

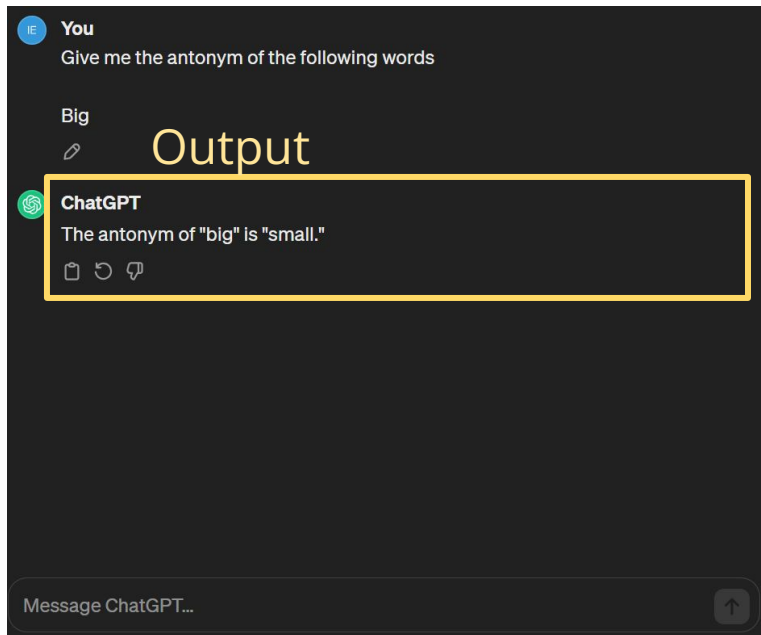
        response = client.chat.completions.create(
            model="gpt-3.5-turbo",
            messages = messages,
            temperature = temperature,
            max_tokens=200,
        )

        chatbot.append((user_input, response.choices[0].message.content))
```

Prompt

Input

Send



```
def interact_customize(chatbot: List[Tuple[str, str]], prompt: str, user_input: str, temperature = 1.0) -> List[Tuple[str, str]]:
    """
    * Arguments

    - chatbot: the model itself, the conversation is stored in list of tuples

    - prompt: the prompt for your designated task

    - user_input: the user input of each round of conversation

    - temp: the temperature parameter of this model. Temperature is used to control the output of the chatbot. The higher the temperature is, the more creative response you will get.

    """
    try:
        messages = []
        messages.append({'role': 'user', 'content': prompt})
        for input_text, response_text in chatbot:
            messages.append({'role': 'user', 'content': input_text})
            messages.append({'role': 'assistant', 'content': response_text})

        messages.append({'role': 'user', 'content': user_input})

        response = client.chat.completions.create(
            model="gpt-3.5-turbo",
            messages = messages,
            temperature = temperature,
            max_tokens=200,
        )

        chatbot.append((user_input, response.choices[0].message.content))
```

Both single-turn and multi-turn is acceptable.

Note

```
# TODO: Fill in the below two lines: chatbot_task and  
chatbot_task  
# The first is for you to tell the user that the chatbot can  
perform certain task  
# The second one is the prompt that make the chatbot able to  
do certain task  
chatbot_task = "FILL_IN_THE_TASK"  
prompt_for_task = "FILL_IN_THE_PROMPT"
```

**Fill in the description of your task and it will be in the red block below
(e.g. Give antonym)**

Actual Input for conversation.

Describe what you want your chatbot to do.

FILL_IN_THE_TASK

Input

Demo for Task 3

How to Delete Your API Keys

OpenAI API

<https://platform.openai.com/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 disable any API key that we've found has leaked publicly.

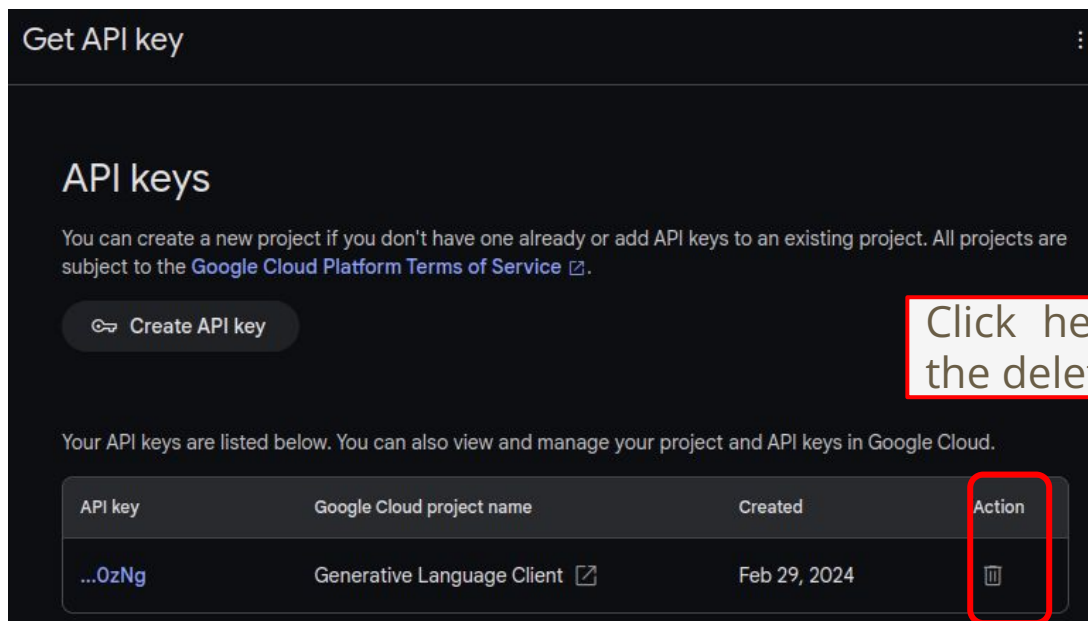
Enable tracking to see usage per API key on the [Usage page](#).

Click here to remove
the delete API keys

NAME	SECRET KEY	TRACKING ⓘ	CREATED	LAST USED ⓘ	PERMISSIONS	
new	sk-...B3QI	Enabled	Jan 23, 2024	Never	All	 
key1	sk-...jYke	Enabled	Feb 10, 2024	Feb 11, 2024	All	 
key2	sk-...8DRP	Enabled	Feb 10, 2024	Feb 11, 2024	All	 
arxiv	sk-...FeFz	Enabled	Feb 27, 2024	Mar 7, 2024	All	 
battle	sk-...n1CI	Enabled	Feb 29, 2024	Mar 2, 2024	All	 

Gemini API

<https://aistudio.google.com/app/apikey>



Get API key

API keys

You can create a new project if you don't have one already or add API keys to an existing project. All projects are subject to the [Google Cloud Platform Terms of Service](#).

Create API key

Your API keys are listed below. You can also view and manage your project and API keys in Google Cloud.

API key	Google Cloud project name	Created	Action
...OzNg	Generative Language Client ↗	Feb 29, 2024	

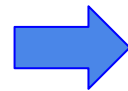
Click here to remove the delete API keys

Issues You Might Encounter

ModuleNotFoundError : No module named 'xxx'

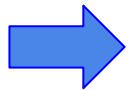
- If you encounter this error

```
-----  
ModuleNotFoundError                                Traceback (most recent call last)  
<ipython-input-1-775a2fd69b31> in <cell line: 4>()  
    2 import google.generativeai as genai  
    3 from typing import List, Tuple  
----> 4 import gradio as gr  
      5 import json  
      6  
ModuleNotFoundError: No module named 'gradio'
```



- You haven't installed the packages yet

Solution: Execute corresponding code block (e.g. *!pip3 install xxx*)



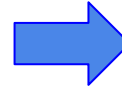
```
# Install required packages  
!pip3 install gradio  
!pip install -q -U google-generativeai
```

NameError : name 'List' is not defined

- If you encounter this error

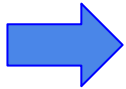
```
NameError                                Traceback (most recent call last)
<ipython-input-1-7d9430f1bb29> in <cell line: 5>()
      3
      4 # function to clear the conversation
----> 5 def reset() -> List:
      6     return []
      7

NameError: name 'List' is not defined
```



- You haven't import the packages

Solution: Execute this code block



▽ Import and Setup

Remember to fill in your Gemini API in this block, the tutorial to get the API is in our homework 3 slides.
Please make sure not to share your API with anyone else.

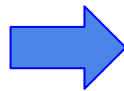


```
# Import packages
import google.generativeai as genai
from typing import List, Tuple
```

FileNotFoundError : [Errno 2] No such file or directory

- If you encounter this error

```
-----  
FileNotFoundError                                Traceback (most recent call last)  
<ipython-input-2-f5b78b6c4587> in <cell line: 2>()  
    1 # load the conversation log json file  
----> 2 with open("part1.json", "r") as f:  
      3     context = json.load(f)  
      4  
      5 chatbot = context['chatbot']  
  
FileNotFoundError: [Errno 2] No such file or directory: 'part1.json'
```

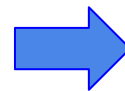
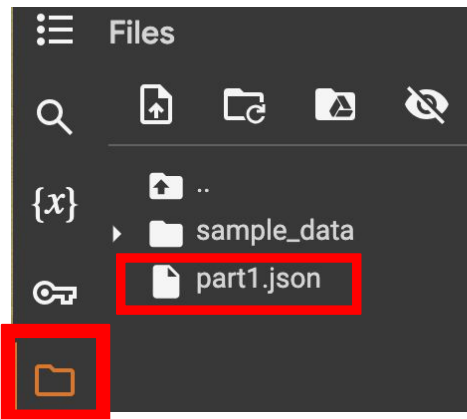


- You might forgot to export the json file or upload one.

Solution:

Please check if there exists the required json file

1.



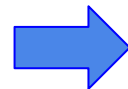
If not, refer to corresponding pages such as [p67](#)

Gradio UI not showing : block running but not working

- If you encounter this error



```
reset_button = gr.Button(value="Reset")  
  
with gr.Column():  
    gr.Markdown("# Save your Result.\n After you get a satisfied result. Click Export")  
    export_button = gr.Button(value="Export")  
    sent_button.click(interact_roleplay, inputs=[chatbot, input_textbox, temperature_slider], outputs=[chatbot])  
    reset_button.click(reset, outputs=[chatbot])  
    export_button.click(export_roleplay, inputs=[chatbot, description_textbox])  
  
demo.launch(debug = True)
```



- You might forgot to stop other code blocks (especially those Gradio UI blocks)

Solution:

Check and stop other running code blocks



```
demo.launch(debug = True)  
... Setting queue=True in a Colab notebook requires sharing enabled. Setting `share`  
Colab notebook detected. This cell will run indefinitely so that you can see errors.  
Running on public URL: https://c398c508428cbe3d06.gradio.live  
This share link expires in 72 hours. For free permanent hosting and GPU upgrades
```

Wrong API key

- If you see this output (The block will finish execution without error occurring)

```
# You should see "Set Gemini API sucessfully!!" if nothing goes wrong.
try:
    model.generate_content(
        "test",
    )
    print("Set Gemini API sucessfully!!")
except:
    print("There seems to be something wrong with your Gemini API. Please follow our demonstration in the slide to get a correct one.")
```

There seems to be something wrong with your Gemini API. Please follow our demonstration in the slide to get a correct one.

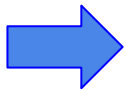
Solution:

Refer to [p33-p42](#) (Gemini)

[p43-p52](#) (OpenAI)

to double check if you retrieve the correct API key.

Then fill in the api key as the description (refer to [p62](#))



Prompting Issue

Problem: Several prompts will encounter such error.

Tell me how to stay healthy

Sorry, an error occurred: The `response.text` quick accessor only works for simple (single-`Part`) text responses. This response is not simple text. Use the `result.parts` accessor or the full `result.candidates[index].content.parts` lookup instead.

Solution: Change your prompts

Evaluation Assistants on DaVinci for Grading

Grading

Model Version: **GPT-4-Turbo-8k**

Assistant: GenAI Homework3 Assistant

- 3 Prompts
 - Summarization Grader
 - Roleplay Grader
 - Customized Task Grader

Copy your result to the Grading Assistant in DaVinci Platform.

Link of Davinci Platform: <https://dvcbot.net/>

DaVinci Install Assistant

+ Chat

Search

No Data.

No Data.

1. Press "+ Chat"

- Plugins
- Prompts
- DVCx Library
- Assistants

Model
GPT-4-Turbo-8k

Plugins
Select plugins used in chat

Conversation Style
☒ Precise ☐ Neutral ☐ Creative

2. Choose "Assistant"

+ Assistant

Search

No Data.

- DaVinci 達哥
- Automatic judging system
- Translation Pro
- Doc Writing Expert
- Visual Design Master - ...

Suggestion

Type a message or type "/" to select a prompt...

DaVinci

+ Chat

Search

No Data.

More Actions 99%

← | →

Assistant File Prompt

Plugins Prompts DVCx Library Assistants

1. Search "GenAI Homework3 Assistant"

Q GenAI Homework3 Assistant

All 1 assistants

Sort by A - Z

GenAI Homework3 Assistant
IE71325@GMAIL.COM

Others ★ 0 (0) ↓ 3

Auto Grading System for NTU GenAI 2024 Spring HW3.

2. Click "GenAI Homework3 Assistant"

Assistant File Prompt

+ Assistant

Search

DaVinci 達哥

Automatic judging system

Translation Pro

Doc Writing Expert

Visual Design Master - ...

Type a message or type "/" to select a prompt...

+ Chat

Q Search

No Data.

More Actions

99%



Plugins

Prompts

DVCx Library

Assistants

[← Back](#)

Assistant / GenAI Homework3 Assistant



GenAI Homework3 Assistant

IE71325@GMAIL.COM

Others

★ 0 (0)



3



Install

1. Install[About](#)[Review](#)

Plugins(0)



{...} Prompts(3)



Files(0)

Description

Auto Grading System for NTU GenAI 2024 Spring HW3.

Authorization

Suggestion



Type a message or type "/" to select a prompt...



Assistant

File

Prompt

+ Assistant

Q Search



DaVinci 達哥



Automatic judging system



Translation Pro



Doc Writing Expert



Visual Design Master - ...

DaVinci Grading

DaVinci

1. Create a new chat for each task

+ Chat

Search

No Data.

Plugins

Prompts

DVCx Library

Assistants

Model

GPT-4-Turbo-8k

Plugins

Select plugins used in chat

Conversation Style

☒ Precise

☐ Neutral

☐ Creative

Assistant

File

Prompt

+ Assistant

Search

DaVinci 達哥

GenAI Homework3 Assi...

Automatic judging system

Translation Pro

Doc Writing Expert

Visual Design Master - ...

2. Click to see Assistant list

3. Choose (GenAI Homework3 Assistant)

⚠ Ensure to open a **NEW** chat every time you re-evaluate the result

4. Choose your Grader/Task

Prompts

{...} Summarization Grader

{...} Roleplay Grader

{...} Customized Task Grader



GenAI Homework3 Assistant

Auto Grading System for NTU GenAI 2024 Spring HW3.

Prompts

Summarization Grader

This chatbot would use the given article and the correspond summarization to grade your score.

article

Enter a value for article

summarization

Enter a value for summarization

Submit

**Fill in these textbox
with results from
Colab**

Suggestion

DaVinci

+ Chat

Search

Seattle's Rolling Neiq...

Assistant : GenAI Homework3 Assistant | Model : GPT-4-Turbo-8k |

claims of partial occupation. These developments have sparked concerns regarding Ukraine's military tactics and the sustainability of their defense amid a munitions crisis, prompting a critical evaluation of command performance by Col. Gen. Oleksandr Syrskyi, the newly appointed Ukrainian military commander. Syrskyi's criticism and adjustments in command highlight the ongoing challenges faced by Ukraine in maintaining its defensive positions against a resurgent Russian offensive, exacerbated by limitations in Western support regarding ammunition and personnel.

Grade: 2 points

Feedback: The summary is clear and captures the main ideas of the article, including the recent Russian advances, the strategic retreat of Ukrainian forces from Avdiivka, and the challenges faced by Ukraine's military in terms of munitions and command. It also mentions the new Ukrainian military commander, Col. Gen. Oleksandr Syrskyi, and his response to the situation on the front lines.

However, the summary does not fully convey the depth of the "near-existential debate" for frontline troops mentioned in the original article, nor does it detail the specific criticisms made by Syrskyi regarding the performance of his officers. Additionally, the summary could have included the context of Syrskyi's appointment following the replacement of the previous commander, which is relevant to understanding the current state of the Ukrainian military leadership.

To improve, the summary could benefit from incorporating these nuances and providing a more detailed account of the internal challenges within the Ukrainian military, as well as the implications of the reported shortcomings in Western aid. This would offer a more comprehensive overview of the complexities of the situation and the pressures facing Ukraine's defense strategy.

Regenerate Response

Suggestion

More Actions 99%

Type a message or type "/" to select a prompt...

Assistant

File

Prompt

+ Assistant

Search

DaVinci 達哥

GenAI Homework3 Assi...

Automatic judging system

Translation Pro

Doc Writing Expert

Visual Design Master - ...

Press this to download as .json file

Assistant gives feedback and a final score for your essay

Demo for Evaluation Assistants on DaVinci for Grading

Daily Usage Limit

There is a **daily usage quota limitation** on the DaVinci platform, based on your quota, you can run

- Summarization: ~5 times/day, **or (not and)**
- Role Play: ~10 times/day, **or (not and)**
- Customized: ~15 times/day

Note: This depends on the amount of words you input

... More Actions 99% 

You can also check the **bottom left** of Davinci Platform remaining usages

BE CAREFUL

Avoid Harmful Prompts and Tasks

- Harmful, offensive, or disrespectful prompts or task will most likely receive a low score.

Evaluation Prompts for 旁聽生

For those who wants to judge their score by themselves, we provide the full prompts for evaluation list here:

- [Summarization](#)
- [Roleplay](#)
- [Customized](#)

To use the prompt, replace the contents inside {{ }}.

Eg. Replace “{{article}}” by your article in summarization task.

Submission

Submission (NTU COOL)

- NTU COOL
 - Turn in the json file you **exported from the DaVinci Platform**
 - **Rename it** as the following format, lower case

<student_ID>_part1_hw3.json

<student_ID>_part2_hw3.json

<student_ID>_part3_hw3.json

Submission (NTU COOL)

- **Do not submit code**
- Json File must include score for each task given by grading assistant, like this: Grade: 3 points
- No restrictions on upload quota but only the most recent submission will be kept
- **No late submission is allowed**
- **Please make sure that you submit .json files otherwise you will get 0 points.**
- **BE CAREFUL NOT TO SUBMIT THE JSON FILE PRODUCE FROM COLAB OTHERWISE YOU WILL GET 0 POINT**

 We will compare the files you submit with our backend data records to ensure they match. Therefore, please **do NOT** manually edit your JSON file.

Important Dates

Important Dates

- Deadline for Submission (NTU Cool)

2024/03/21 23:59:59 (UTC+8)

- Grading Release Date

2024/04/04 23:59:59 (UTC+8)

Regulations

Regulations

- Plagiarism in any form is prohibited.
- **Do NOT** share your prompts and evaluation results (JSON files) with others.
- **Do NOT** submit the JSON files that are not obtained using your Davinci account
- **Do NOT** attempt to manually edit your JSON files.
- 第一次違反以上規定, **該作業0分, 學期總成績再乘以 0.9**
- 第二次違反以上規定, **學期成績 F**
- Prof. Lee & the TAs preserve the rights to change the rules & grades.

If You Have Any Questions

- NTU Cool HW3 作業討論區
 - 如果同學的問題不涉及作業答案或隱私, 請 **一律使用** NTU Cool 討論區。在發問前請使用 NTU Cool 討論區的搜尋功能找找看你的問題是不是有人問過了
 - 助教們會優先回答 NTU Cool討論區上的問題
 - 如果寄信問的問題已經在 NTU Cool 討論區回答過了, 助教只會回信跟你 說這題已經在 NTU Cool 回答了, 助教不會在信件中回答你
 - 如果是要問程式出錯的問題的話, 請將錯誤訊息連同在哪裡發生的一起截圖傳給我們
- Email: ntu-gen-ai-2024-spring-ta@googlegroups.com
 - Title should start with [\[GenAI 2024 Spring HW3\]](#)
 - Email with the wrong title will be moved to trash automatically
- TA Hours
 - Time: **3/8 (16:30 - 17:20), 3/15 (13:30-14:20, 16:30 - 17:20)**
 - Location: **綜合大講堂**