Getting Started with the OpenAI APIs

Xavier Morera

Helping developers create epic applications

www.xaviermorera.com / www.bigdatainc.org / www.lupo.ai

What is an API?

An API, or Application Programming Interface, is a set of rules and protocols that allows one piece of software or application to interact with and access the functionality or data of another software, service, or system. It defines the methods and data formats that developers can use to request and exchange information between different software components.

APIs are essential for enabling communication and integration between different software systems, allowing them to work together to perform various tasks and functions. They serve as intermediaries that enable one piece of software to access the features or data of another, without needing to understand the internal workings of that software.

APIs are used in a wide range of applications, including web development, mobile app development, cloud computing, and more. They can be categorized into different types, including:

1. **Web APIs:** These are APIs that allow web services to communicate with each other o
internet. They are commonly used for accessing web-based resources and services, such

Regenerate

Send a message

Search ⌘ K

GET STARTED

Introduction
Quickstart
Libraries
Models
Deprecations
Tutorials
Policies

GUIDES

GPT
Chat completions API
Function calling
Completions API
Chat completions vs. Completions
Managing Tokens
FAQ
GPT best practices
Image generation

Lower values for temperature result in more consistent outputs, while higher values generate more diverse and creative results. Select a temperature value based on the desired trade-off between coherence and creativity for your specific application.

Is fine-tuning available for the latest models?

Yes, for some. Currently, you can only fine-tune `gpt-3.5-turbo` and our updated base models (`babbage-002` and `davinci-002`). See the fine-tuning guide for more details on how to use fine-tuned models.

Do you store the data that is passed into the API?

As of March 1st, 2023, we retain your API data for 30 days but no longer use your data sent via the API to improve our models. Learn more in our data usage policy. Some endpoints offer zero retention.

How can I make my application more safe?

If you want to add a moderation layer to the outputs of the Chat API, you can follow our moderation guide to prevent content that violates OpenAI's usage policies from being shown.

Should I use ChatGPT or the API?

ChatGPT offers a chat interface to the models in the OpenAI API and a range of built-in features such as integrated browsing, code execution, plugins, and more. By contrast, using OpenAI's API provides more flexibility.

The OpenAI Platform at a High Level

Purpose of the OpenAI API

Leverage cutting-edge NLP models to perform various natural language tasks, such as text generation, language translation, question-answering, sentiment analysis, and more.

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

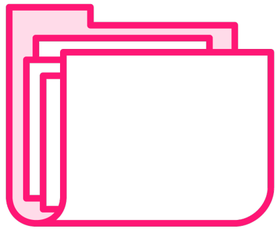
GPT best practices
Learn best practices for building with GPT models

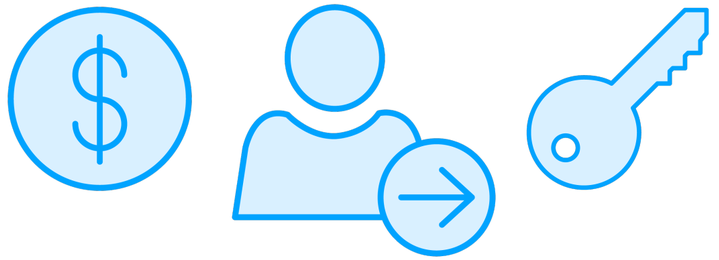
Embeddings
Learn how to search, classify, and compare text

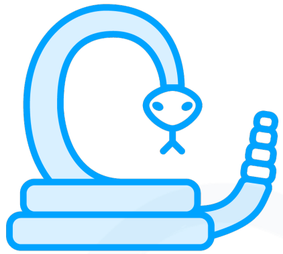
Speech to text
Learn how to turn audio into text

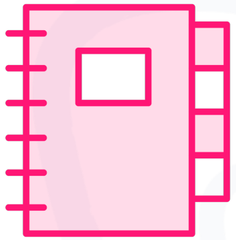
Getting Started:
The OpenAI API Starter Pack

The OpenAI API Starter Pack









The OpenAI API Starter Pack

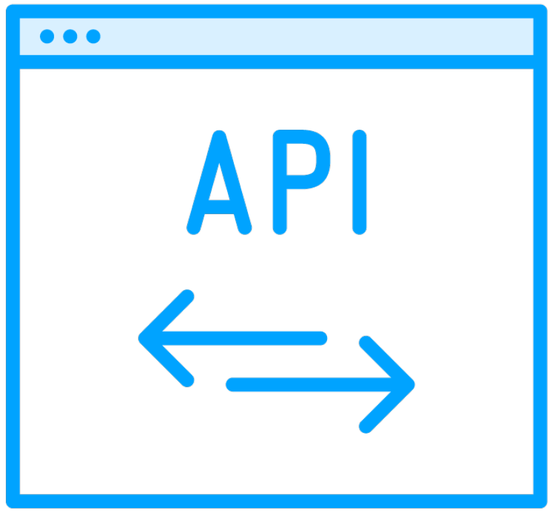
Using the OpenAI API with Python

Using the OpenAI API with Python

API

Library

Making API Requests



Make a request to a specific endpoint

- Many endpoints available
  - For different models and versions

Using the `requests` library in Python

- Setting specific parameters

Parse the `response` object to extract the model output

Search ⌘ K

GETTING STARTED

Introduction
Authentication
Making requests

ENDPOINTS

Audio
Chat
Completions
Embeddings
Fine-tuning
Files
Images
Models
Moderations

DEPRECATED

Fine-tunes
Edits

Making requests

You can paste the command below into your terminal to run your first API request. Make sure to replace `$OPENAI_API_KEY` with your secret API key.

```
curl https://api.openai.com/v1/chat/completions \
  -H "Content-Type: application/json" \
  -H "Authorization: Bearer $OPENAI_API_KEY" \
  -d '{
     "model": "gpt-3.5-turbo",
     "messages": [{"role": "user", "content": "Say this is a test!"}],
     "temperature": 0.7
   }'
```

This request queries the `gpt-3.5-turbo` model (which under the hood points to the **latest** `gpt-3.5-turbo` **model variant**) to complete the text starting with a prompt of "*Say this is a test*". You should get a response back that resembles the following:

```
{
    "id": "chatcmpl-abc123",
    "object": "chat.completion",
    "created": 1677858242,
    "model": "gpt-3.5-turbo-0613",
    "usage": {
        "prompt_tokens": 13,
```

Using the OpenAI API with Python

```
https://api.openai.com/v1/chat/completions
```

Using the OpenAI API with Python

```
https://api.openai.com/v1/chat/completions

payload = {
    "model": "gpt-3.5-turbo",
    "messages": [{"role": "user", "content": prompt}],
    "max_tokens": max_tokens
}
```

Using the OpenAI API with Python

```
https://api.openai.com/v1/chat/completions

payload = {
    "model": "gpt-3.5-turbo",
    "messages": [{"role": "user", "content": prompt}],
    "max_tokens": max_tokens
}
```

Using the OpenAI API with Python

Using the OpenAI Python Library

Search ⌘ K

GET STARTED

Introduction
Quickstart
Libraries
Python library
Node.js library
Azure OpenAI libraries
Community libraries
Models
Deprecations
Tutorials
Policies

GUIDES

GPT
GPT best practices
Image generation
Fine-tuning
Embeddings

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 service
5 openai.api_key = os.getenv("OPENAI_API_KEY")
6
7 chat_completion = openai.ChatCompletion.create(model="gpt-3.5-turbo", messages=[
```

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 "Hello world"
```

openai-python

Public

Watch 191 Fork 1.7k Star 11.4k

main 9 branches 62 tags Go to file Add file Code

logankilpatrick Update README.md (#625) 5453a19 last week 182 commits

.github/ISSUE_TEMPLATE	Create issue templates (#214)	8 months ago
examples	make SQL example pointer consistent (#273)	7 months ago
openai	Update embeddings_utils.py to set default model to text-embedding...	last week
public	Remove support for py <3.6, mypy, lots of cleanup (#19)	2 years ago
.gitignore	Add azure deployments + an example/tutorial for using Azure endpoi...	last year
LICENSE	Initial commit	3 years ago
Makefile	Create wheels (#297)	7 months ago
README.md	Update README.md (#625)	last week
chatml.md	Update chatml.md (#580)	2 months ago
pyproject.toml	Fix invalid pyproject.toml file and move to setup.cfg (#201)	9 months ago
pytest.ini	Add an option to use Azure endpoints for the /completions & /search ...	2 years ago
setup.cfg	Fixes for embedding requirements install (#227)	8 months ago
setup.py	Fix invalid pyproject.toml file and move to setup.cfg (#201)	9 months ago

README.md

About

The official Python library for the OpenAI API

pypi.org/project/openai/

python openai

Readme

MIT license

Activity

11.4k stars

191 watching

1.7k forks

Report repository

Releases 61

v0.28.1 Latest

last week

+ 60 releases

Used by 11

+ 3

Question

Why should I consider using the official OpenAI library?

(Instead of the API)

Convenient access to the API

API Requests vs. Library

API	vs.	Library
Finer control		Ease of use and convenience
More flexibility		Less code
Lightweight		Automatic tokenization
Easier to make a mistake		Officially maintained

Using the OpenAI Python Library

```
import openai

openai.api_key = key

chat_completion = openai.ChatCompletion.create(
        model="gpt-3.5-turbo",
         messages=[
                { "role": "user",
                 "content": "Hello world"
                }])
```

Using the OpenAI Python library

Chat Completion vs. Completions

Chat Completion	vs.	Completions
Interactive conversations		Direct response
Context		Simpler output
Additional capabilities		One-off output
		Legacy

Anatomy of an API Call

Anatomy of an API call

```
url = 'https://api.openai.com/v1/chat/completions'
```

url

Specifies the endpoint that will be called
    For example, chat completion
Other endpoints available
    Completion, embeddings, images...

Search ⌘ K

GETTING STARTED

Introduction
Authentication
Making requests

ENDPOINTS

Audio
Chat
The chat completion object
The chat completion chunk object
Create chat completion
Completions
Embeddings
Fine-tuning
Files
Images
Models
Moderations

Create chat completion

`POST https://api.openai.com/v1/chat/completions`

Creates a model response for the given chat conversation.

Request body

`model` string Required

ID of the model to use. See the **model endpoint compatibility** table for details on which models work with the Chat API.

`messages` array Required

A list of messages comprising the conversation so far. **Example Python code**.

+ Show properties

`functions` array Optional

A list of functions the model may generate JSON inputs for.

+ Show properties

`function_call` string or object Optional

Controls how the model calls functions. "none" means the model will not call a function and instead generates a message. "auto" means the model can pick between generating a message or calling a function. Specifying a particular function via `{"name": "my_function"}` forces the model to

NO STREAMING | STREAMING | FUNCTION CALLING

Example request gpt-3.5-turbo curl Copy

```
curl https://api.openai.com/v1/chat/completio
  -H "Content-Type: application/json" \
  -H "Authorization: Bearer $OPENAI_API_KEY"
  -d '{
    "model": "gpt-3.5-turbo",
    "messages": [
      {
        "role": "system",
        "content": "You are a helpful assista
      },
      {
        "role": "user",
        "content": "Hello!"
      }
    ]
  }'
```

Response Copy

```
{
  "id": "chatcmpl-123",
  "object": "chat.completion",
  "created": 1677652288,
  "model": "gpt-3.5-turbo-0613",
  "choices": [{
    "index": 0,
```

```
headers = {
        'Content-Type': 'application/json',
        'Authorization': f'Bearer {key}'
    }
```

headers

Contains the API key
Used for authorization
Set the content type
Optionally, to specify organization

```
data = {
        "model": "gpt-3.5-turbo",
        "messages": [{"role": "user", "content": prompt}],
        "temperature": 0.7
    }
```

data

Parameters passed to the API

- **model**
- **messages**
- **temperature**

Search ⌘ K

GET STARTED

Introduction
Quickstart
Libraries
Models
Overview
Model updates
GPT-4
GPT-3.5
DALL·E
Whisper
Embeddings
Moderation
GPT-3
How we use your data
Endpoint compatibility
Deprecations
Tutorials
Policies

Models

Diverse set of models with different capabilities and price points

Overview

The OpenAI API is powered by a diverse set of models with different capabilities and price points. You can also make customizations to our models for your specific use case with fine-tuning.

MODELS	DESCRIPTION
GPT-4	A set of models that improve on GPT-3.5 and can understand as well as generate natural language or code
GPT-3.5	A set of models that improve on GPT-3 and can understand as well as generate natural language or code
GPT base	A set of models without instruction following that can understand as well as generate natural language or code
DALL·E	A model that can generate and edit images given a natural language prompt
Whisper	A model that can convert audio into text
Embeddings	A set of models that can convert text into a numerical form
Moderation	A fine-tuned model that can detect whether text may be sensitive or unsafe
GPT-3 Legacy	A set of models that can understand and generate natural language

Message

Unit of text within a conversation

A conversation typically consists of a series of messages

- Allow guiding the model's behavior and receive coherent and context-aware responses

Each message has

- Role
- Content

Content is the actual text of the message

- Input, questions, instructions, responses...

Role

Indicates the identity of the sender of the message

- user
  - End-user interacting with the model
- system
  - Provides instructions or context to the AI model
- assistant
  - Responses generated by the AI model

Search ⌘ K

GET STARTED

Introduction
Quickstart
Libraries
Models
Deprecations
Tutorials
Policies

GUIDES

GPT
GPT best practices
Six strategies for getting better results
Write clear instructions
Provide reference text
Split complex tasks into simpler subtasks
Give GPTs time to "think"
Use external tools
Test changes systematically

Tactic: Ask the model to adopt a persona

The system message can be used to specify the persona used by the model in its replies.

SYSTEM	When I ask for help to write something, you will reply with a document that contains at least one joke or playful comment in every paragraph.
USER	Write a thank you note to my steel bolt vendor for getting the delivery in on time and in short notice. This made it possible for us to deliver an important order.

Open in Playground ↗

Tactic: Use delimiters to clearly indicate distinct parts of the input

Delimiters like triple quotation marks, XML tags, section titles, etc. can help demarcate sections of text to be treated differently.

USER	Summarize the text delimited by triple quotes with a haiku. """insert text here"""

Open in Playground ↗

SYSTEM	You will be provided with a pair of articles (delimited with XML tags) about the same topic. First summarize the arguments of each article. Then indicate which of them

Temperature

Used to control randomness or creativity in responses generated by the model

Control output by adjusting temperature

- One of the most important settings

Values

- 0.2 or lower generates deterministic responses
- 0.5 balances randomness and creativity
- 1.0 or higher makes the answers more random and creative

```
{
  "id": "chatcmpl-84AzvmyyFymS92uoTsDPTFvLxVvMy",
  "object": "chat.completion",
  "created": 1696006279,
  "model": "gpt-3.5-turbo-0613",
  "choices": [
    {
      "index": 0,
      "message": {
        "role": "assistant",
        "content": "Why don't scientists trust atoms?\n\nBecause they make up
everything!"
      },
```


response

Model generated output

```
      "message": {
        "role": "assistant",
        "content": "Why don't scientists trust atoms?\n\nBecause they make up
everything!"
      },
      "finish_reason": "stop"
    }
  ],
  "usage": {
    "prompt_tokens": 11,
    "completion_tokens": 13,
    "total_tokens": 24
  }
}
```


response

Model generated output

- **Response in natural language**

Specifies a finish_reason

- **stop, length, null, function_call, content_filter**

Listing and Retrieving Models

"All models are NOT created equal"

Search ⌘ K

GET STARTED

Introduction
Quickstart
Libraries
Models
- Overview
- Model updates
- GPT-4
- GPT-3.5
- DALL·E
- Whisper
- Embeddings
- Moderation
- GPT-3
- How we use your data
- Endpoint compatibility

Deprecations
Tutorials
Policies

Models

Overview

The OpenAI API is powered by a diverse set of models with different capabilities and price points. You can also make customizations to our models for your specific use case with fine-tuning.

MODELS	DESCRIPTION
GPT-4	A set of models that improve on GPT-3.5 and can understand as well as generate natural language or code
GPT-3.5	A set of models that improve on GPT-3 and can understand as well as generate natural language or code
GPT base	A set of models without instruction following that can understand as well as generate natural language or code
DALL·E	A model that can generate and edit images given a natural language prompt
Whisper	A model that can convert audio into text
Embeddings	A set of models that can convert text into a numerical form
Moderation	A fine-tuned model that can detect whether text may be sensitive or unsafe
GPT-3 Legacy	A set of models that can understand and generate natural language
Deprecated	A full list of models that have been deprecated

Search ⌘ K

GET STARTED

Introduction
Quickstart
Libraries
Models
Overview
Model updates
GPT-4
GPT-3.5
DALL·E
Whisper
Embeddings
Moderation
GPT-3
How we use your data
Endpoint compatibility
Deprecations
Tutorials
Policies

GPT-3.5

GPT-3.5 models can understand and generate natural language or code. Our most capable and cost effective model in the GPT-3.5 family is `gpt-3.5-turbo` which has been optimized for chat using the Chat completions API but works well for traditional completions tasks as well.

LATEST MODEL	DESCRIPTION	MAX TOKENS	TRAINING DATA
`gpt-3.5-turbo`	Most capable GPT-3.5 model and optimized for chat at 1/10th the cost of `text-davinci-003`. Will be updated with our latest model iteration 2 weeks after it is released.	4,097 tokens	Up to Sep 2021
`gpt-3.5-turbo-16k`	Same capabilities as the standard `gpt-3.5-turbo` model but with 4 times the context.	16,385 tokens	Up to Sep 2021
`gpt-3.5-turbo-instruct`	Similar capabilities as `text-davinci-003` but compatible with legacy Completions endpoint and not Chat Completions.	4,097 tokens	Up to Sep 2021
`gpt-3.5-turbo-0613`	Snapshot of `gpt-3.5-`	4,097 tokens	Up to Sep 2021

GPT-3.5 GPT-4

ChatGPT

Give me ideas
for what to do with my kids' art

Design a database schema
for an online merch store

Explain superconductors
like I'm five years old

Plan a trip
to explore the rock formations in Cappadocia, Turkey

Send a message

Search ⌘ K

GET STARTED

Introduction
Quickstart
Libraries
Models
Overview
Model updates
GPT-4
GPT-3.5
DALL·E
Whisper
Embeddings
Moderation
GPT-3
How we use your data
Endpoint compatibility
Deprecations
Tutorials
Policies

GPT-4

GPT-4 is currently accessible to those who have made at least one successful payment through our developer platform.

GPT-4 is a large multimodal model (accepting text inputs and emitting text outputs today, with image inputs coming in the future) that can solve difficult problems with greater accuracy than any of our previous models, thanks to its broader general knowledge and advanced reasoning capabilities. Like `gpt-3.5-turbo`, GPT-4 is optimized for chat but works well for traditional completions tasks using the Chat completions API. Learn how to use GPT-4 in our GPT guide.

LATEST MODEL	DESCRIPTION	MAX TOKENS	TRAINING DATA
`gpt-4`	More capable than any GPT-3.5 model, able to do more complex tasks, and optimized for chat. Will be updated with our latest model iteration 2 weeks after it is released.	8,192 tokens	Up to Sep 2021
`gpt-4-0613`	Snapshot of `gpt-4` from June 13th 2023 with function calling data. Unlike `gpt-4`, this model will not receive updates, and will be deprecated 3 months after a new version is released.	8,192 tokens	Up to Sep 2021

Search ⌘ K

GET STARTED

Introduction
Quickstart
Libraries
Models
Deprecations
Tutorials
Policies

GUIDES

GPT
GPT best practices
Image generation
Fine-tuning
Embeddings
Speech to text
Moderation
Rate limits
Error codes

Incremental model updates

As announced in March 2023, we regularly release new versions of `gpt-4` and `gpt-3.5-turbo`.

Each model version is dated with an `-MMDD` suffix; e.g., `gpt-4-0613`. The undated model name, e.g., `gpt-4`, will typically point to the latest version (e.g. `gpt-4` points to `gpt-4-0613`). Users of undated model names will be notified by email typically 2 weeks before any change takes place.

After a new version is launched, older versions will typically be deprecated 3 months later.

Migrating to replacements

Once a model is deprecated, be sure to migrate all usage to a suitable replacement before the shutdown date. Requests to models past the shutdown date will fail.

To help measure the performance of replacement models on your tasks, we have open-sourced Evals, a Python framework for evaluating language models.

If new models perform worse on your tasks, let us know by by submitting a pull request to our Evals repo with examples of the task.

Deprecation history

All deprecations are listed below, with the most recent announcements at the top.

2023-08-22: Fine-tunes endpoint

Search ⌘ K

GET STARTED

Introduction
Quickstart
Libraries
Models
Overview
Model updates
GPT-4
GPT-3.5
DALL·E
Whisper
Embeddings
Moderation
GPT-3
How we use your data
Endpoint compatibility
Deprecations
Tutorials
Policies

For details, see our API data usage policies. To learn more about zero retention, get in touch with our sales team.

Model endpoint compatibility

ENDPOINT	LATEST MODELS
/v1/audio/transcriptions	whisper-1
/v1/audio/translations	whisper-1
/v1/chat/completions	gpt-4, gpt-4-0613, gpt-4-32k, gpt-4-32k-0613, gpt-3.5-turbo, gpt-3.5-turbo-0613, gpt-3.5-turbo-16k, gpt-3.5-turbo-16k-0613
/v1/completions (Legacy)	gpt-3.5-turbo-instruct, babbage-002, davinci-002
/v1/embeddings	text-embedding-ada-002
/v1/fine_tuning/jobs	gpt-3.5-turbo, babbage-002, davinci-002
/v1/moderations	text-moderation-stable, text-moderation-latest

This list excludes all of our DALL·E models and our deprecated models.

Was this page useful?

Listing and retrieving models

Model Pricing

Pricing

Simple and flexible. Only pay for what you use.

Contact sales Learn more

Language models

Multiple models, each with different capabilities and price points. Prices are per 1,000 tokens. You can think of tokens as pieces of words, where 1,000 tokens is about 750 words. This paragraph is 35 tokens.

GPT-4

With broad general knowledge and domain expertise, GPT-4 can follow complex instructions in natural language and solve difficult problems with accuracy.

Learn about GPT-4

Model	Input	Output
8K context	$0.03 / 1K tokens	$0.06 / 1K tokens
32K context	$0.06 / 1K tokens	$0.12 / 1K tokens

GPT-3.5 Turbo

GPT-3.5 Turbo models are capable and cost-effective.

`gpt-3.5-turbo` is the flagship model of this family and is optimized for dialog.

`gpt-3.5-turbo-instruct` is an Instruct model and only supports a 4K context window.

Learn about GPT-3.5 Turbo ↗

Model	Input	Output
4K context	$0.0015 / 1K tokens	$0.002 / 1K tokens
16K context	$0.003 / 1K tokens	$0.004 / 1K tokens

Fine-tuning models

Create your own custom models by fine-tuning our base models with your training data. Once you fine-tune a model, you'll be billed only for the tokens you use in requests to that model.

Learn about fine-tuning

Model	Training	Input usage	Output usage
babbage-002	$0.0004 / 1K tokens	$0.0016 / 1K tokens	$0.0016 / 1K tokens
davinci-002	$0.0060 / 1K tokens	$0.0120 / 1K tokens	$0.0120 / 1K tokens
GPT-3.5 Turbo	$0.0080 / 1K tokens	$0.0120 / 1K tokens	$0.0160 / 1K tokens

Embedding models

Build advanced search, clustering, topic modeling, and classification functionality with our embeddings offering.

Learn about embeddings

Model	Usage
Ada v2	$0.0001 / 1K tokens

Base models

GPT base models are not optimized for instruction-following and are less capable, but they can be effective when fine-tuned for narrow tasks.

Learn about GPT base models

Other models

Image models

Build DALL·E directly into your apps to generate and edit novel images and art. Our image models offer three tiers of resolution for flexibility.

Learn about image generation

Resolution	Price
1024×1024	$0.020 / image
512×512	$0.018 / image
256×256	$0.016 / image

Audio models

Whisper can transcribe speech into text and translate many languages into English.

Learn about Whisper

Model	Usage
Whisper	$0.006 / minute (rounded to the nearest second)

Older models

We continue to improve our models and periodically retire older, less used models.

In a nutshell...

Understand costs

Know what you are spending

Avoid surprises

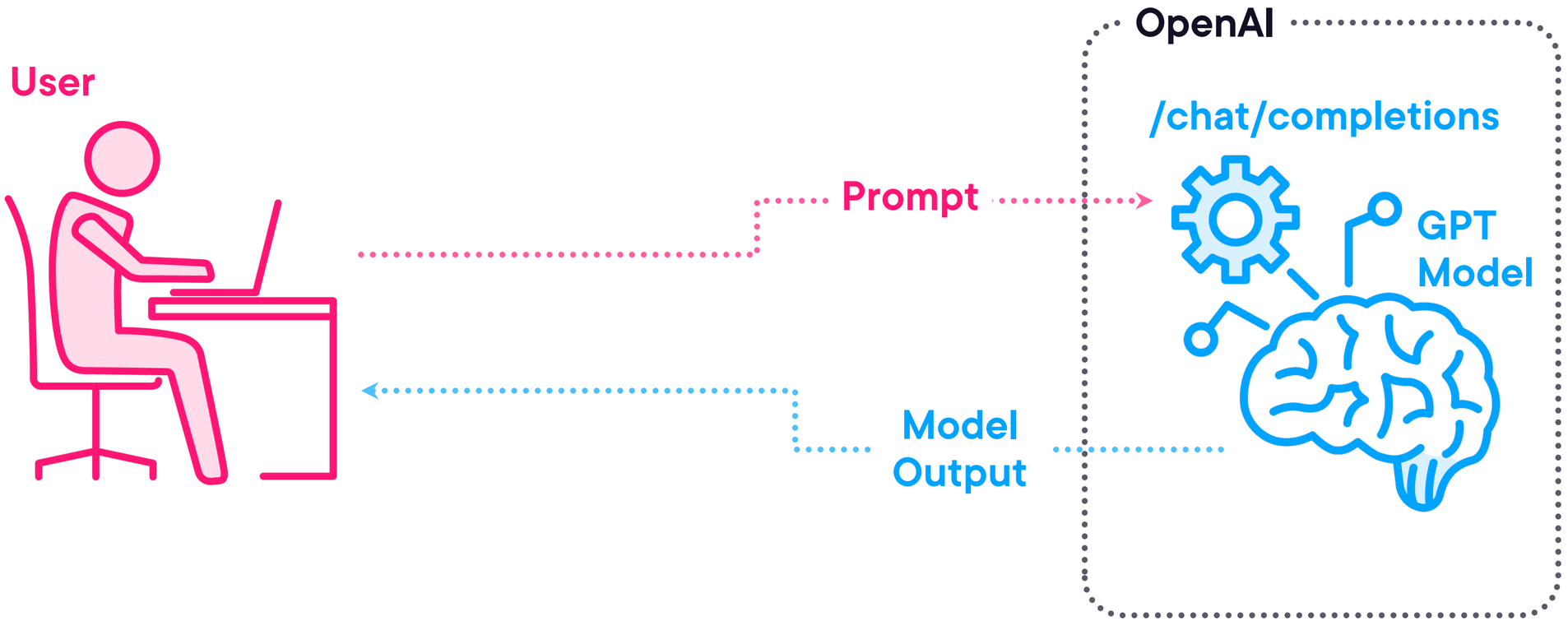
Control costs

Set a soft limit to receive an email

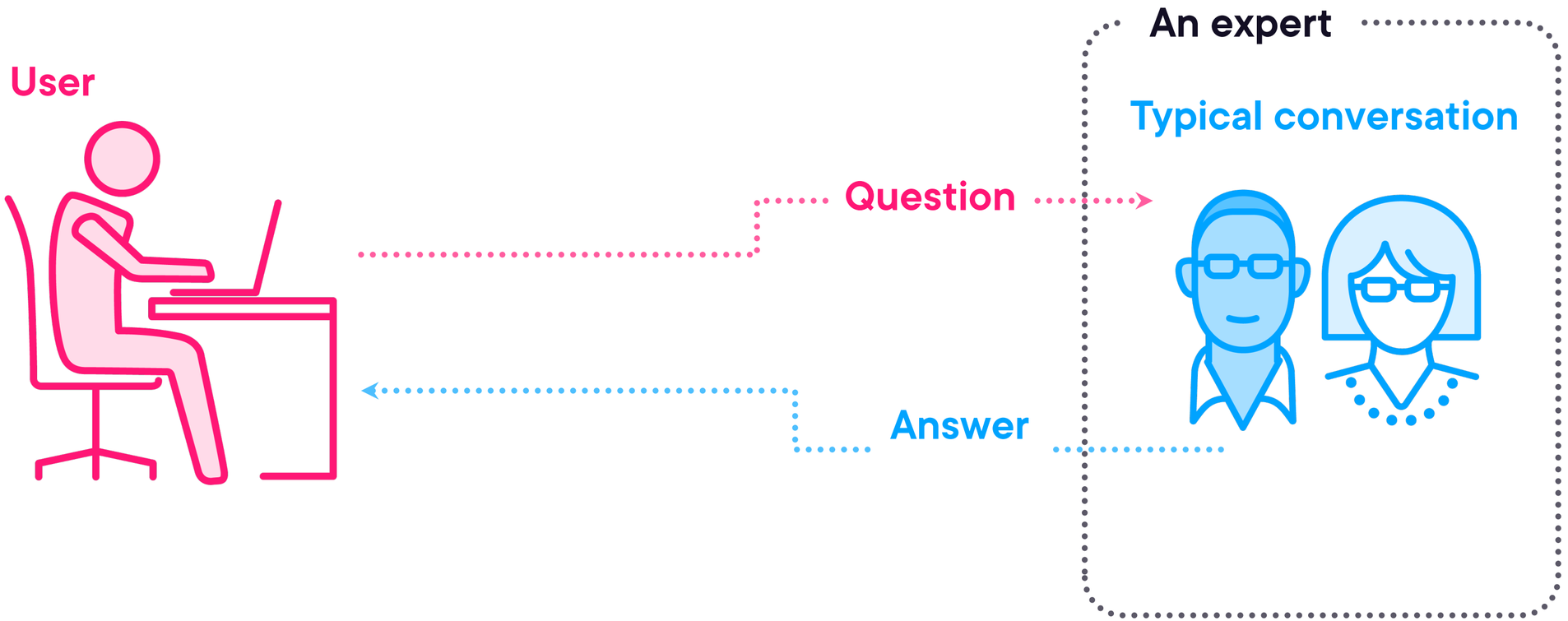
Hard limit, but it may cause disruptions to your users

Using the Chat Completions API

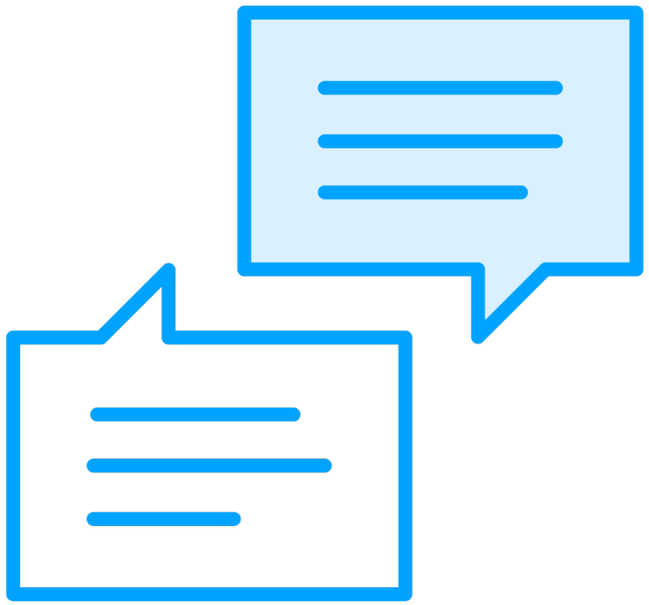
Chat Completions



Chat Completions



Chat Models



Chat models take a list of messages as input

Return a model generated message as output

Send new messages with previous output

- Continue the conversation

Makes a multi-turn conversation easy

- Can serve only for single-turn tasks

Prompting

Zero-shot prompting

Few-shot prompting

Zero-shot Prompting

Provide a prompt to the model that is not part of the training data

- A prompt with no samples

The model generates a suitable response

- Even when it is a new scenario

May require a good description to get a good response

Few-shot Prompting

Samples are provided to the model to demonstrate what's requested

Suitable alternative when it is hard to describe a request

Chat Completions Request

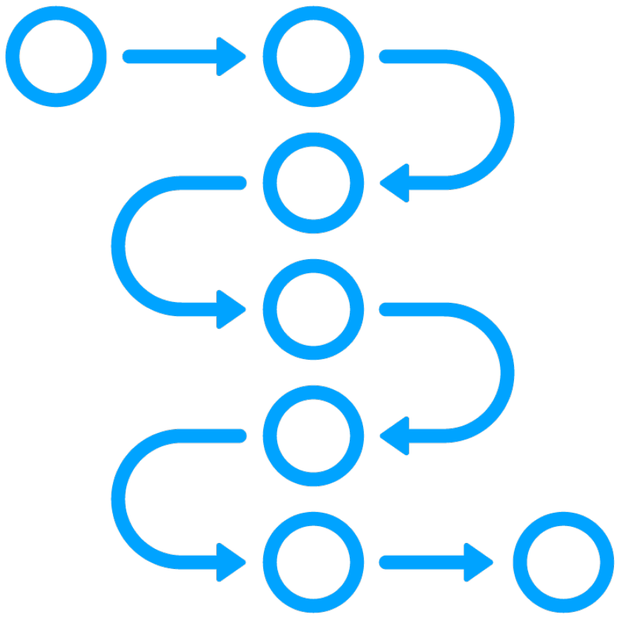
```
URL = "https://api.openai.com/v1/chat/completions"
data = {
   "model": "gpt-3.5-turbo",
   "messages": [
     {
        "role": "system",
        "content": "You are a helpful assistant."
     },
     {
        "role": "user",
        "content": "Hello!"
     }
    ],
    "temperature": 0.7
}
```

Response: Completion Object


```
{
  "id": "chatcmpl-84EByMrzWIejEIcViJDSAGR5cYDFr",
  "object": "chat.completion",
  "created": 1696018558,
  "model": "gpt-3.5-turbo-0613",
  "choices": [
    {
      "index": 0,
      "message": {
        "role": "assistant",
        "content": "Hi there! How can I assist you today?"
      },
      "finish_reason": "stop"
    }
  ],
  "usage": {
    "prompt_tokens": 19,
    "completion_tokens": 10,
    "total_tokens": 29
  }
}
```


Chat Completion

Stream Completions



Cases where responses need to be returned as soon as they are available

- Similar to when you use ChatGPT

Set `stream: true` returns tokens as soon as they are available

- Instead of waiting for the full response

Creating Images

Images API

Creating images from prompts

Edit an image based on a prompt

Create variations of an image

Creating Images

Create an image given a prompt

- Specify size
- Can request multiple images

Endpoint

- `api.openai.com/v1/images/generations`

Alternatively, also use the library

- `openai.Image.create`

Creating Images Using the API

```
URL = "https://api.openai.com/v1/images/generations"

data = {
    "prompt": "A cute baby sea otter",
    "n": 2,
    "size": "1024x1024"
}
```

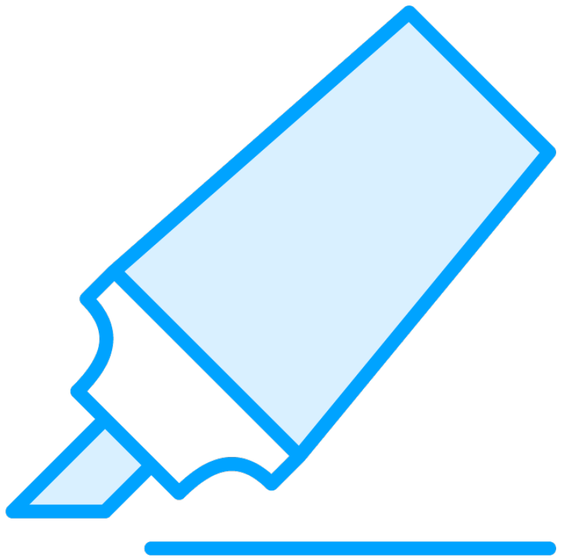
Creating Images Using the Library

```
library_response = openai.Image.create(
  prompt="A cute baby sea otter",
  n=1,
  size="1024x1024"
)
```

Creating images

Editing Images

Editing Images



Endpoint allows you to edit and extend images

- By uploading a mask

Transparent area of the mask indicates where the image should be edited

Prompt should describe the entire image

- Not the mask area

Images must be

- Square and less than 4 MB in size

Editing Images Using the API

```
URL = "https://api.openai.com/v1/images/edits"

files = {
    "image": open("./otter1.png", "rb"),
    "mask": open("./mask1.png", "rb")
}
data = {
    "prompt": "A cute baby sea otter wearing a beret",
    "n": 1,
    "size": "1024x1024"
}
```

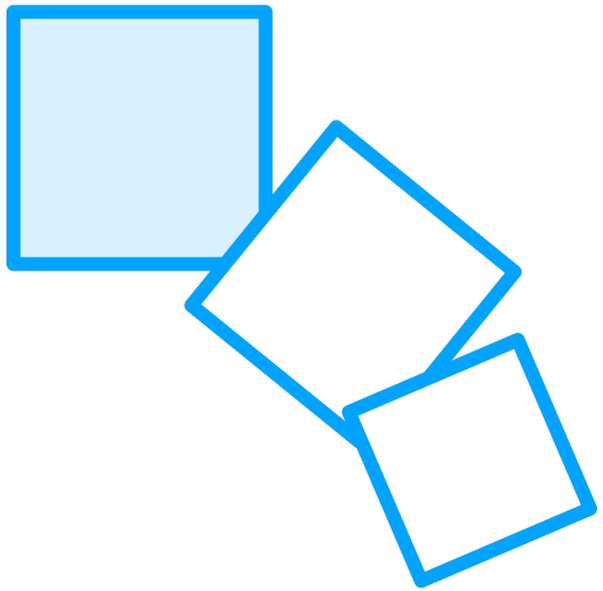
Editing Images Using the Library

```
library_response = openai.Image.create_edit(
  image=open("./otter1.png", "rb"),
  mask=open("./mask1.png", "rb"),
  prompt="A cute baby sea otter wearing a beret",
  n=1,
  size="1024x1024"
)
```

Editing images

Image Variations

Image Variations



Generate a variation of an existing image

Square image and less than 4 MB in size

Image Variations Using the API

```
URL = "https://api.openai.com/v1/images/variations"

files = {
    "image": open("./otter1.png", "rb")
}
data = {
    "n": 2,
    "size": "1024x1024"
}
```

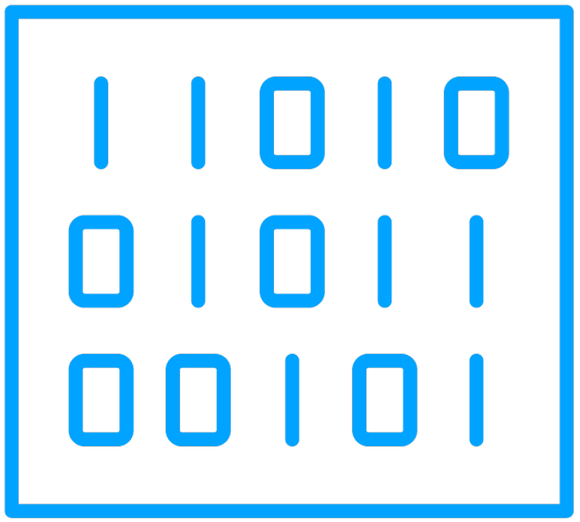
Image Variations Using the Library

```
library_response = openai.Image.create_variation(
  image=open("./otter1.png", "rb"),
  n=1,
  size="1024x1024"
)
```

Image variations

Creating Embeddings

Embeddings



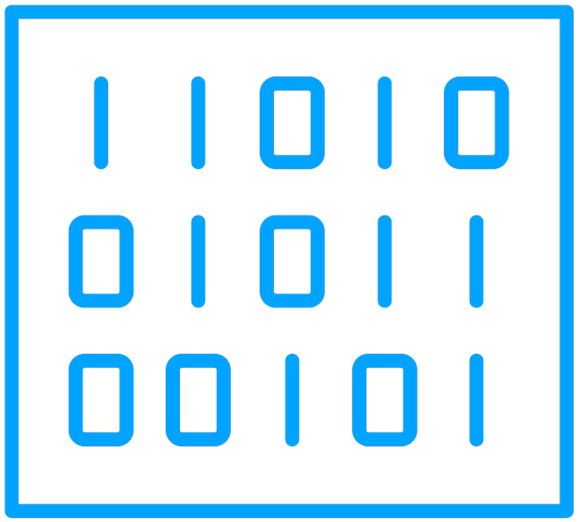
Fundamental concept in Machine Learning and Deep Learning

- Natural language processing (NLP)
- Computer vision

Representations of words, sentences, or images in a lower dimensional space

Play a crucial role in capturing and encoding underlying patterns and relationships in data

Embeddings



Vector (list) of floating-point numbers

Distance between vectors measures relatedness

- Small distance suggests high relatedness
- Large distance suggests low relatedness

Creating Embeddings

```
URL = "https://api.openai.com/v1/embeddings"

request_data = {
    "input": "The food was delicious and the waiter...",
    "model": "text-embedding-ada-002"
}
```

Creating embeddings

Transcribing Text

Transcribing Text

Possible to transcribe text

- Speech to text
- Using the Whisper model

Whisper is a general-purpose speech recognition model

- Model is called `whisper-1`

Can set temperature

Search ⌘ K

GETTING STARTED

Introduction

Authentication

Making requests

ENDPOINTS

Audio

Create transcription

Create translation

Chat

Completions

Embeddings

Fine-tuning

Files

Images

Models

Moderations

DEPRECATED

Create transcription

`POST https://api.openai.com/v1/audio/transcriptions`

Transcribes audio into the input language.

Request body

`file` file Required

The audio file object (not file name) to transcribe, in one of these formats: flac, mp3, mp4, mpeg, mpga, m4a, ogg, wav, or webm.

`model` string Required

ID of the model to use. Only `whisper-1` is currently available.

`prompt` string Optional

An optional text to guide the model's style or continue a previous audio segment. The **prompt** should match the audio language.

`response_format` string Optional Defaults to json

The format of the transcript output, in one of these options: json, text, srt, verbose_json, or vtt.

`temperature` number Optional Defaults to 0

The sampling temperature, between 0 and 1. Higher values like 0.8 will

Example request curl Copy

```
curl https://api.openai.com/v1/audio/transcrip
  -H "Authorization: Bearer $OPENAI_API_KEY" \
  -H "Content-Type: multipart/form-data" \
  -F file="@/path/to/file/audio.mp3" \
  -F model="whisper-1"
```

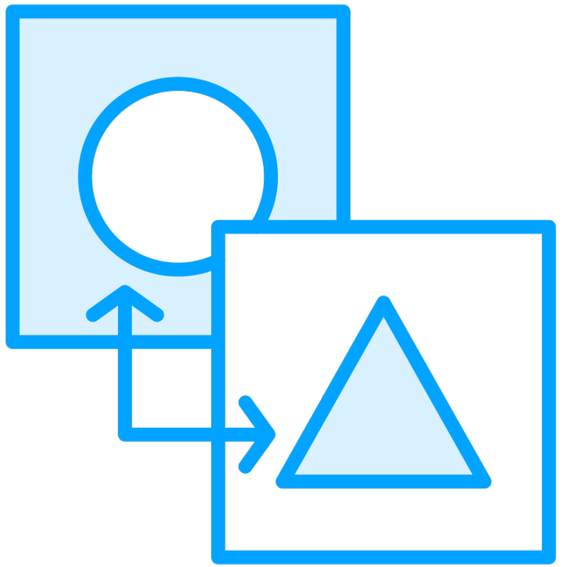
Response Copy

```
{
  "text": "Imagine the wildest idea that you'v
}
```

Transcribing text

Translating Text

Translating Text



Use Whisper to translate audio into English

Can add an optional prompt to guide the model's style

- Or continue a previous audio segment

Can set temperature

Search ⌘ K

GETTING STARTED

Introduction
Authentication
Making requests

ENDPOINTS

Audio
Create transcription
Create translation
Chat
Completions
Embeddings
Fine-tuning
Files
Images
Models
Moderations

DEPRECATED

Create translation

`POST https://api.openai.com/v1/audio/translations`

Translates audio into English.

Request body

`file` file Required

The audio file object (not file name) translate, in one of these formats: flac, mp3, mp4, mpeg, mpga, m4a, ogg, wav, or webm.

`model` string Required

ID of the model to use. Only `whisper-1` is currently available.

`prompt` string Optional

An optional text to guide the model's style or continue a previous audio segment. The **prompt** should be in English.

`response_format` string Optional Defaults to json

The format of the transcript output, in one of these options: json, text, srt, verbose_json, or vtt.

`temperature` number Optional Defaults to 0

The sampling temperature, between 0 and 1. Higher values like 0.8 will

Example request curl Copy

```
curl https://api.openai.com/v1/audio/translati
  -H "Authorization: Bearer $OPENAI_API_KEY" \
  -H "Content-Type: multipart/form-data" \
  -F file="@/path/to/file/german.m4a" \
  -F model="whisper-1"
```

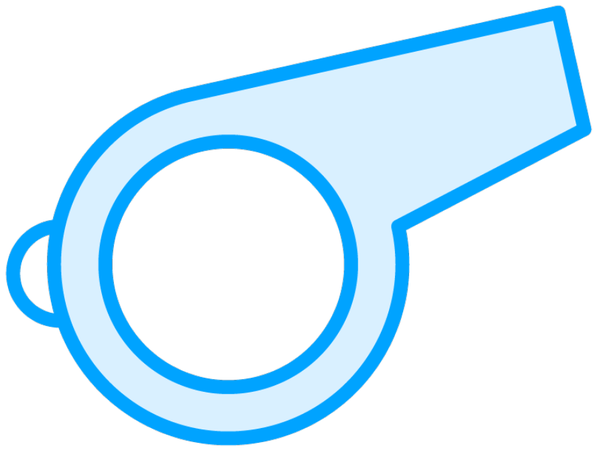
Response Copy

```
{
  "text": "Hello, my name is Wolfgang and I co
}
```

Translating text

Moderating Text

Moderating Text



API provides functionality to check if content complies with OpenAI's usage policies

- Can be used to filter content

Search ⌘ K

GET STARTED

Introduction
Quickstart
Libraries
Models
Deprecations
Tutorials
Policies

GUIDES

GPT
GPT best practices
Image generation
Fine-tuning
Embeddings
Speech to text
Moderation
Overview
Quickstart

Moderation

Overview

The moderations endpoint is a tool you can use to check whether content complies with OpenAI's usage policies. Developers can thus identify content that our usage policies prohibits and take action, for instance by filtering it.

The models classifies the following categories:

CATEGORY	DESCRIPTION
`hate`	Content that expresses, incites, or promotes hate based on race, gender, ethnicity, religion, nationality, sexual orientation, disability status, or caste. Hateful content aimed at non-protected groups (e.g., chess players) is harrassment.
`hate/threatening`	Hateful content that also includes violence or serious harm towards the targeted group based on race, gender, ethnicity, religion, nationality, sexual orientation, disability status, or caste.
`harassment`	Content that expresses, incites, or promotes harassing language towards any target.
`harassment/threatening`	Harassment content that also includes violence or serious harm towards any target.
`self-harm`	Content that promotes, encourages, or depicts acts of self-harm, such

Moderating text

Working with Files

Working with Files

Upload

List

Retrieve

Delete

Working with files