**Python and OpenAI**

<https://platform.openai.com/docs/overview>

<https://realpython.com/practical-prompt-engineering/>

[small adjustment to test Github]

**Getting and Installing an OpenAI Key**

Python calls OpenAI through an API that requires an *API Key* for authentication.

(You can create API keys at a user or service account level. Service accounts are tied to a bot and should be used to provision access for production systems. Each API key can be scoped to (a) *project keys* or (b) *user keys*. Project keys provide access to a single project (preferred option). Access [*Project API keys*](https://platform.openai.com/settings/organization/api-keys) by selecting the specific project for which you wish to generate keys.

Organization IDs are found on your [Organization settings](https://platform.openai.com/settings/organization/general) page. Project IDs are found on your [General settings](https://platform.openai.com/settings/organization/projects) page.

*Steps*

1. Open an OpenAI developer account and buy some credits.

2. Click on the *API reference* link to gather some useful information:

A screenshot of a computer

Description automatically generated

3. Go to the OpenAI API keys page (link referenced above) and click on the *Create new secret key* button:

A screenshot of a computer

Description automatically generated

4. Complete the *Create new secret key* form:

A screenshot of a computer

Description automatically generated

[Note: I don’t know if I should have created a Project first, but I selected *Default project* on my first attempt.]

*MacOS Instructions for Installing the OpenAI Key*

1. On my computer, *.zshrc* and *.bash\_profile* are stored in my user home directory:

A black text with a white background

Description automatically generated

2. Use the *ls -a* command to see the hidden and unhidden files in this directory:

A screenshot of a computer code

Description automatically generated

3. My computer has both a *.zshrc* and *.bash\_profile* file, so I added the OpenAI API Key to both files. You can edit them using *nano* within the command terminal. For example, the use following command…



…to open the following window:

A screenshot of a computer

Description automatically generated

4. Add this line to each file:



Note: The API Key should be in quotation marks. Only a part of my key is showing, but there is a quotation mark at the end of the line. I put the key all one line. (It may be possible to wrap the key, but I *know* it works if you put it on one line.)

Note: The reason there are two configuration files is that there are two versions of Terminal – one called bash and the other called zsh. You can configure the Terminal shortcut to target either one. I happen to use zsh and I believe it is the newer of the two.

5. After you add “export OPENAI… [etc.]” to both the .zshrc and .bash\_profile files, re-start your terminal to activate the new configuration. (You do not need to restart the computer, only Terminal, which is what the .zshrc and .bash\_profile files configure.)

*Windows Instructions for Installing the OpenAI Key*

[to come – involves storing the key in the environment variables]

**Installing the OpenAI Package**

1. Create and activate venv (in this case, called *venv\_openai*):



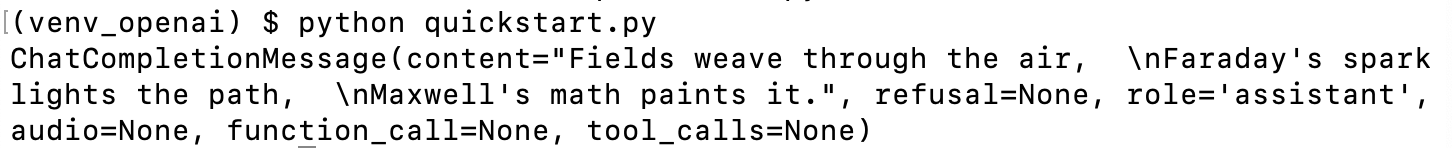
Note: The OpenAI version installed (by default) on my computer was 1.58.1

2. Create a file called quickstart.py and paste the following code into it:

A screenshot of a computer program

Description automatically generated

3. Save the file and run it in a venv as follows:



Note: Without any formatting, the message returned from OpenAI and printed will include line ending character and other metadata, but it writes a very fun, and even beautiful haiku:

Fields weave through the air,

Faraday’s spark lights the path,

Maxwell’s math paints it.

It’s not perfect, but not bad.

(continued on next page)

**Prompt Engineering**

<https://realpython.com/practical-prompt-engineering/>

The following are notes from a Real Python tutorial titled ***Prompt Engineering: A Practical Example***

1. Create a directory in a venv folder to hold several files that will be downloaded in the next step. (On one computer, I named the folder prompt\_eng\_materials.)

2. Download the project files from this location: <https://realpython.com/bonus/prompt-engineering-code/>

3. Use pip to install the required dependencies:

A close-up of a code

Description automatically generated

(On Windows, the command is: **python -m pip install -r requirements.txt** )

4. Run **pip list**, if you wish, to see the list of packages now installed:

A screenshot of a computer code

Description automatically generated

**Note: The instructions (here in these notes) assume that whether you have installed the API Key in your Terminal configuration files (on a Mac) or in the environment variables (on Windows), described at the opening of this document.**

5. Now run the app.py script with the following command:



Output:

A screenshot of a cell phone

Description automatically generated

6. In case Real Python takes down this tutorial, all of the following are the key files:

**app.py**

A screenshot of a computer program

Description automatically generated

A screenshot of a computer program

Description automatically generated

A screen shot of a computer

Description automatically generated

**Here is copyable text for app.py:**

import argparse

import os

import tomllib

from pathlib import Path

from openai import OpenAI

\_\_all\_\_ = ["get\_chat\_completion"]

# Authenticate

client = OpenAI(api\_key=os.getenv("OPENAI\_API\_KEY"))

# Load settings file

settings\_path = Path("settings.toml")

with settings\_path.open("rb") as settings\_file:

SETTINGS = tomllib.load(settings\_file)

def parse\_args() -> argparse.Namespace:

"""Parse command-line input."""

parser = argparse.ArgumentParser()

parser.add\_argument("file\_path", type=Path, help="Path to the input file")

return parser.parse\_args()

def main(args: argparse.Namespace) -> None:

file\_content = args.file\_path.read\_text("utf-8")

print(get\_chat\_completion(file\_content))

def get\_chat\_completion(content: str) -> str:

"""Send a request to the /chat/completions endpoint."""

response = client.chat.completions.create(

model=SETTINGS["general"]["model"],

messages=\_assemble\_chat\_messages(content),

temperature=SETTINGS["general"]["temperature"],

seed=12345, # Doesn't do anything for older models

)

return response.choices[0].message.content

def \_assemble\_chat\_messages(content: str) -> list[dict]:

"""Combine all messages into a well-formatted list of dicts."""

messages = [

{"role": "system", "content": SETTINGS["prompts"]["role\_prompt"]},

{"role": "user", "content": SETTINGS["prompts"]["negative\_example"]},

{

"role": "system",

"content": SETTINGS["prompts"]["negative\_reasoning"],

},

{

"role": "assistant",

"content": SETTINGS["prompts"]["negative\_output"],

},

{"role": "user", "content": SETTINGS["prompts"]["positive\_example"]},

{

"role": "system",

"content": SETTINGS["prompts"]["positive\_reasoning"],

},

{

"role": "assistant",

"content": SETTINGS["prompts"]["positive\_output"],

},

{"role": "user", "content": f">>>>>\n{content}\n<<<<<"},

{"role": "user", "content": SETTINGS["prompts"]["instruction\_prompt"]},

]

return messages

if \_\_name\_\_ == "\_\_main\_\_":

main(parse\_args())

**The settings.toml file that this script relies on has the following contents:**

[general]

chat\_models = ["gpt-3.5-turbo", "gpt-4"]

model = "gpt-3.5-turbo"

temperature = 0

[prompts]

instruction\_prompt = """

Remove personally identifiable information, only show the date,

and replace all swear words with "😤"

"""

role\_prompt = """

"""

positive\_example = """

"""

positive\_reasoning = """

"""

positive\_output = """

"""

negative\_example = """

"""

negative\_reasoning = """

"""

negative\_output = """

"""

**The chats.txt file contents are the following messages:**

[support\_tom] 2023-07-24T10:02:23+00:00 : What can I help you with?

[johndoe] 2023-07-24T10:03:15+00:00 : I CAN'T CONNECT TO MY BLASTED ACCOUNT

[support\_tom] 2023-07-24T10:03:30+00:00 : Are you sure it's not your caps lock?

[johndoe] 2023-07-24T10:04:03+00:00 : Blast! You're right!

[support\_amy] 2023-06-15T14:45:35+00:00 : Hello! How can I assist you today?

[greg\_stone] 2023-06-15T14:46:20+00:00 : I can't seem to find the download link for my purchased software.

[support\_amy] 2023-06-15T14:47:01+00:00 : No problem, Greg. Let me find that for you. Can you please provide your order number?

[greg\_stone] 2023-06-15T14:47:38+00:00 : It's 1245789. Thanks for helping me out!

[support\_louis] 2023-05-05T09:22:12+00:00 : Hi, how can I help you today?

[karen\_w] 2023-05-05T09:23:47+00:00 : MY BLASTED ORDER STILL HASN'T ARRIVED AND IT'S BEEN A WEEK!!!

[support\_louis] 2023-05-05T09:24:15+00:00 : I'm sorry to hear that, Karen. Let's look into this issue.

[support\_louis] 2023-05-05T09:25:35+00:00: Can you please provide your order number so I can check the status for you?

[karen\_w] 2023-05-05T09:26:12+00:00: Fine, it's 9876543.

[support\_louis] 2023-05-05T09:26:45+00:00: Thank you, Karen. I see there was a delay in shipping. Your order will arrive within the next 2 days.

[support\_jenny] 2023-06-18T17:35:28+00:00: Hello! How can I help you today?

[alex\_harper] 2023-06-18T17:36:05+00:00: I accidentally placed an order twice, can you help me cancel one?

[support\_jenny] 2023-06-18T17:36:25+00:00: Sure, Alex. Can you give me the order number you'd like to cancel?

[alex\_harper] 2023-06-18T17:36:55+00:00: Yes, it's 1122334. Thank you!

[support\_jenny] 2023-06-18T17:37:32+00:00: I've successfully canceled order number 1122334. You will receive a confirmation email shortly.

[support\_ben] 2023-06-29T11:51:45+00:00: Good morning, what can I assist you with today?

[lisa\_beck] 2023-06-29T11:52:20+00:00: Hi there, I received a damaged item in my order. Can you help me return it?

[support\_ben] 2023-06-29T11:52:45+00:00: I'm sorry to hear that, Lisa. Can you provide your order number and specify the damaged item?

[lisa\_beck] 2023-06-29T11:53:22+00:00: Sure, order number is 5566778 and the damaged item is a coffee mug.

[support\_rachel] 2023-05-04T08:16:37+00:00: How can I help you today?

[mike\_t] 2023-05-04T08:17:15+00:00: My coupon code isn't working at checkout. Can you help?

[support\_rachel] 2023-05-04T08:17:38+00:00: Of course, Mike. Please provide the coupon code you're trying to use.

[mike\_t] 2023-05-04T08:18:02+00:00: It's "HELLO10".

[support\_rachel] 2023-05-04T08:18:37+00:00: I've checked the code, and it seems to have expired. I apologize for the inconvenience. Here's a new code for you to use: "WELCOME15".

[support\_vincent] 2023-06-15T20:43:55+00:00: Good evening! How may I assist you?

[sara\_winters] 2023-06-15T20:44:30+00:00: Hi there, I'm having trouble logging into my account. I've tried resetting my password, but it's not working.

[support\_vincent] 2023-06-15T20:44:52+00:00: I'm sorry to hear that, Sara. Let me help you. Can you please confirm your email address?

[sara\_winters] 2023-06-15T20:45:25+00:00: Sure, it's sara.winters@email.com.

[support\_david] 2023-06-24T16:28:43+00:00: Welcome! What can I do for you today?

[jane\_d] 2023-06-24T16:29:16+00:00: Hi, I need to change my delivery address for my recent order.

[support\_david] 2023-06-24T16:29:43+00:00: Alright, Jane. Please provide your order number.

[jane\_d] 2023-06-24T16:30:11+00:00: It's 3344556. Thanks for your help!