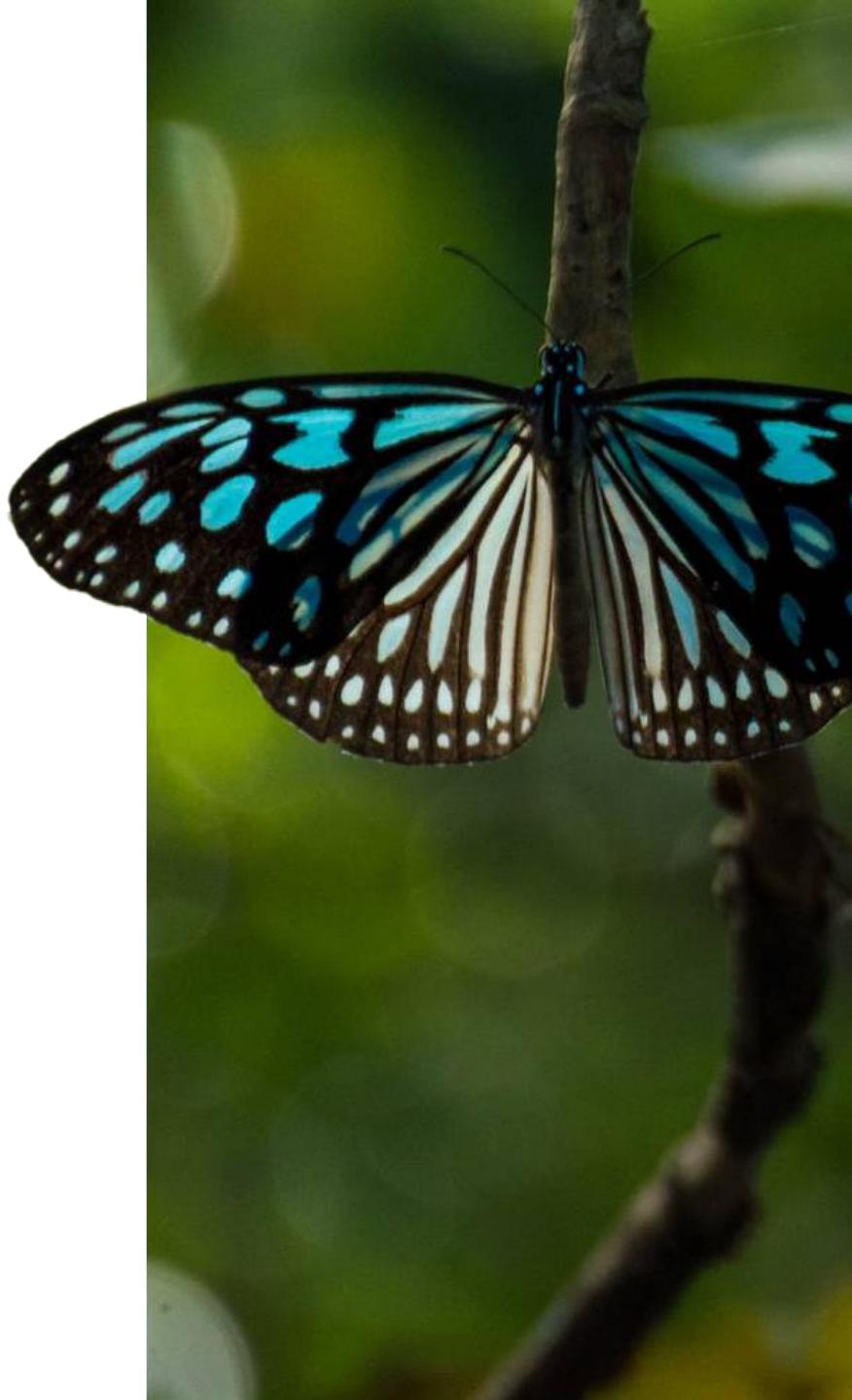


EVERYTHING YOU NEED TO KNOW TO TAKE THE FREE COURSE, CHATGPT PROMPT ENGINEERING FOR DEVELOPERS

VERSION 1.0, JULY 2023

A great way to learn about ChatGPT and Large Language Models – and you don't have to be a developer to take the course!

Mike Boucher



training description

This presentation will provide a recipe for turning ChatGPT **unknown unknowns** into **known unknowns** into **knowns** by leveraging the free 1-hour online course from DeepLearning.AI called, ChatGPT Prompt Engineering for Developers.

After this presentation, you'll have the knowledge and a roadmap to begin your ChatGPT learning journey.

Everything you need to know to take the free course,
ChatGPT Prompt Engineering for developers





outline

01

Introduction

02

Accounts
Setup

03

(Optional)
Computer
Setup

04

(Optional)
Using
Jupyter
Notebook

Presentation in a Nutshell

- Intro to ChatGPT with a brief overview of the course
- Create a ChatGPT account
- Sign up for the course at Deeplearning.AI

The rest is optional

- Install Anaconda
- Install 3 libraries in the Anaconda Python 3.10 instance
 - python-dotenv
 - openai
 - Redlines
- Get and install an OpenAI key on your PC
- Doing simple operations in Jupyter Notebook



lesson 01

Introduction to ChatGPT and the course, ChatGPT Prompt Engineering for Developers



What is ChatGPT?

- ChatGPT is a large language model (LLM) that can engage in human-like conversations
- ChatGPT is trained on a massive text dataset, including books, articles, and websites
- There are 2 ways to interact with ChatGPT
 - Web Interface (with a browser like doing a Google search)
 - API Interface (with a software program)

Prompts and Responses

- Prompt – the query that is input to ChatGPT
- Response – the reply from ChatGPT
- Prompt engineering is the process of creating and reviewing high-quality prompts to guide language models.



In 50 words or less, as a provider or nurse seeing a patient for chronic care, what would I document in a chronic care visit?



In a chronic care visit, as a provider or nurse, you would document the patient's chief complaint, medical history, physical examination findings, current medications, treatment plan, education/counseling provided, patient response/compliance, and follow-up recommendations.



In 10 words or less, as a patient what should I expect in a chronic care visit with my doctor?



Assessment, treatment plan, education, and follow-up recommendations for chronic condition.

ChatGPT Web Interface

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

The screenshot shows the ChatGPT web interface. On the left, there's a sidebar with a 'New chat' button and a list of previous conversations: 'Chronic Care Document', 'Carequality & CalDEx: Connect', 'Automated SOW Generation', 'Certification Maintenance Req', 'NextGen supports Blue Butt', 'NextGen User Guide Request', and 'AeroGlide UltraSlim Toothbrus'. The main area has a large input field at the bottom with placeholder text 'Send a message...'. Above it, a red box highlights a user query and its response. The user query is: 'In 50 words or less, as a provider or nurse seeing a patient for chronic care, what would I document in a chronic care visit?'. The AI response is: 'In a chronic care visit, as a provider or nurse, you would document the patient's chief complaint, medical history, physical examination findings, current medications, treatment plan, education/counseling provided, patient response/compliance, and follow-up recommendations.' At the very bottom, there's a 'Regenerate response' button and a footer note: 'Free Research Preview. ChatGPT may produce inaccurate information about people, places, or facts. ChatGPT May 24 Version'.



ChatGPT API Interface

jupyter 5 Transforming Last Checkpoint: 05/16/2023 (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 (ipykernel)

In [1]:

```
import openai
import os

from dotenv import load_dotenv, find_dotenv
_ = load_dotenv(find_dotenv()) # read local .env file

openai.api_key = os.getenv('OPENAI_API_KEY')
```

In [2]:

```
def get_completion(prompt, model="gpt-3.5-turbo", temperature=0):
    messages = [{"role": "user", "content": prompt}]
    response = openai.ChatCompletion.create(
        model=model,
        messages=messages,
        temperature=temperature,
    )
    return response.choices[0].message["content"]
```

In [3]:

```
prompt = f"""
Translate the following English text to Spanish: \
```\nHi. I would like to order a blender\n```\n"""
response = get_completion(prompt)
print(response)
```

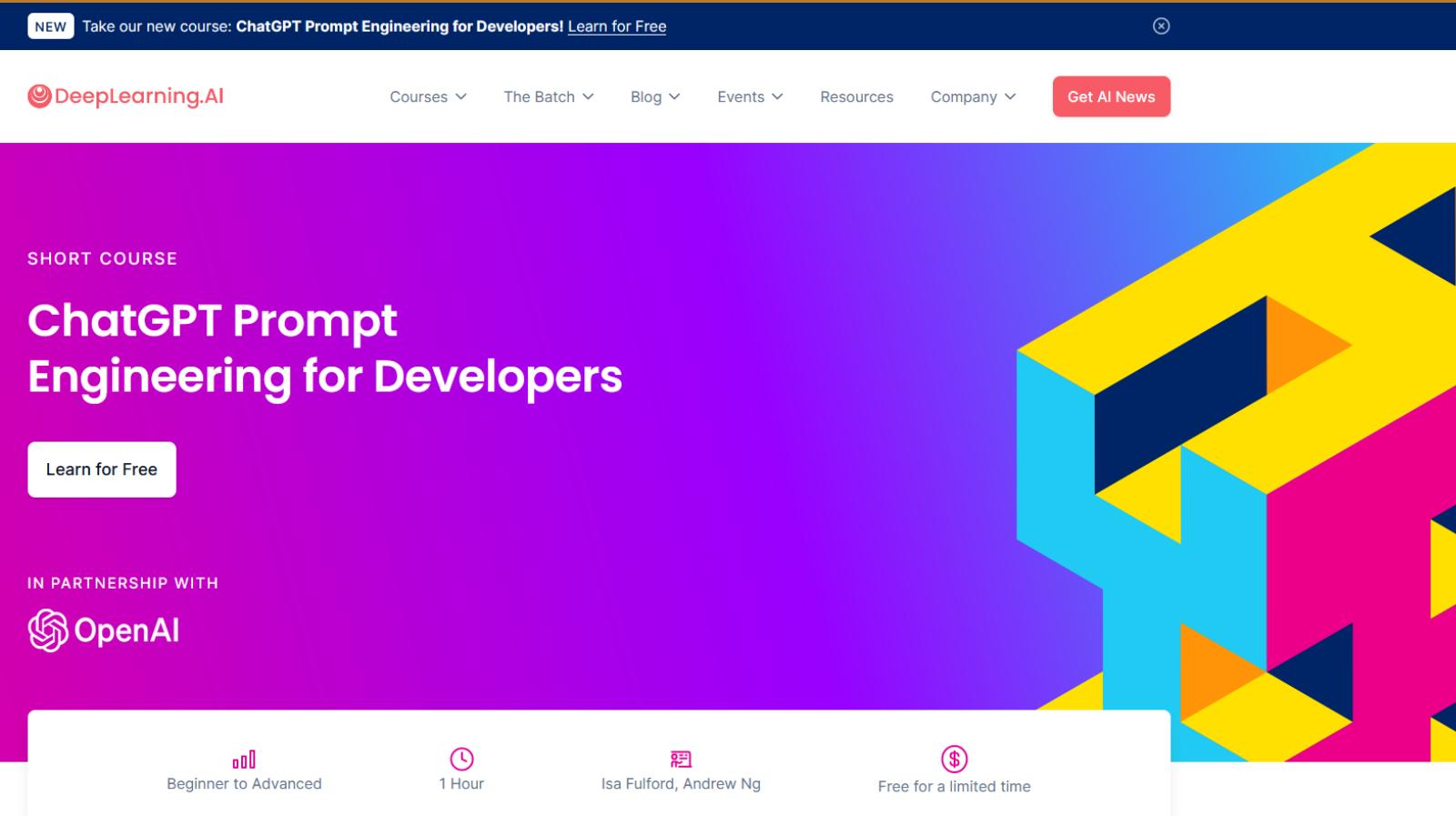
**Prompt**

**Response**



# ChatGPT Prompt Engineering for Developers

<https://www.deeplearning.ai/short-courses/chatgpt-prompt-engineering-for-developers/>



Everything you need to know to take the free course,  
ChatGPT Prompt Engineering for developers

1-hour course

- Explains what ChatGPT is and different ways to use it
- Demonstrates the concepts with simple code
- Provides code examples so you can follow along and experiment on your own

# Course Format

The screenshot shows the course interface. On the left, a sidebar lists course sections: Introduction, Guidelines, Iterative (highlighted in pink), Summarizing, Inferring, Transforming, Expanding, Chatbot, Conclusion, Course Feedback, and Community. The main area has a title "Iterative Prompt Development" and a subtitle "Generate a marketing product description from a product fact sheet". It features a video player with two hosts, Isa Fulford and Andrew Ng, and a code editor window below it. The code editor contains Python code for generating a marketing product description from a fact sheet using the OpenAI API.

## How to Use the Course

- Listen to Isa Fulford and Andrew Ng
- Watch
- Do

Everything you need to know to take the free course,  
ChatGPT Prompt Engineering for developers

This screenshot shows a different part of the course. The sidebar includes sections like Introduction, Guidelines, Iterative (highlighted in pink), Summarizing, Inferring, Transforming, Expanding, Chatbot, Conclusion, Course Feedback, and Community. The main area has a title "Iterative Prompt Development" and a subtitle "Generate a marketing product description from a product fact sheet". It includes a video player and a code editor with Python code for generating a marketing product description. A large circular diagram on the right titled "Iterative Prompt Development" illustrates the process: Idea -> Prompt -> Implementation (code/data) -> Experimental result -> Error Analysis -> Idea, forming a continuous loop.

This screenshot shows another section of the course. The sidebar lists sections: Introduction, Guidelines, Iterative (highlighted in pink), Summarizing, Inferring, Transforming, Expanding, Chatbot, Conclusion, Course Feedback, and Community. The main area has a title "Iterative Prompt Development" and a subtitle "Generate a marketing product description from a product fact sheet". It features a video player and a code editor with Python code for generating a marketing product description. The code editor shows a series of code snippets and their corresponding outputs.

# ChatGPT Concepts in the Course

- **Iterating your prompts** - how to refine your prompts to get what you want
- **Summarizing** - using ChatGPT to summarize information
- **Inferring** - using ChatGPT to infer the emotion of a written product review
- **Transforming** - using ChatGPT to translate text to a different language or to a certain writing style or to correct grammar
- **Expanding** - Using ChatGPT to take a short piece of text and generate a longer piece of text such as an email response or an essay
- **Building an AI Chatbot** - building a chatbot to take restaurant orders

# Running Examples in the Course

- The course embeds a tool called Jupyter Notebook which allows you to view, edit, and run the (Python) code examples from the course
- Jupyter Notebook is used by many data scientists to develop machine learning models using the Python programming language with specialized machine learning libraries and tools
  - E.g., [Applied Machine Learning in Healthcare](#)

The screenshot shows the Udemy course page for "Applied Machine Learning in Healthcare". The course title is prominently displayed on the left. To the right, there's a circular graphic with various healthcare-related topics like Breast Cancer Detection, Diabetes Onset Prediction, DNA Classification, Heart Disease Prediction, and Autism Screening. Below the graphic is a video player showing a play button and a timestamp of 0:00 / 1:21. To the right of the video player is a sidebar titled "Course content" with sections: Section 1: Introduction, Section 2: Breast Cancer Detection, Section 3: Diabetes Onset Detection, Section 4: DNA Classification - The Dataset, Section 5: Heart Disease Prediction with Neural Networks, and Section 6: Autism Screening. Each section has a thumbnail and a duration.

Everything you need to know to take the free course,  
ChatGPT Prompt Engineering for developers

The screenshot shows a course page from DeepLearning.AI titled "ChatGPT Prompt Engineering for Developers". The main content area features a Jupyter Notebook titled "jupyter I6-transforming". The notebook contains two code snippets. The first snippet is for setting up an OpenAI API key:import openai  
import os  
  
from dotenv import load\_dotenv, find\_dotenv  
\_ = load\_dotenv(find\_dotenv()) # read local .env file  
openai.api\_key = os.getenv('OPENAI\_API\_KEY')The second snippet is a function for generating completion using GPT-3.5-turbo:def get\_completion(prompt, model="gpt-3.5-turbo", temperature=0.7):  
 messages = [{"role": "user", "content": prompt}]  
 response = openai.ChatCompletion.create(  
 model=model,  
 messages=messages,  
 temperature=temperature,  
 )  
 return response.choices[0].message["content"]Below the notebook, there's a video player showing a play button and a timestamp of 0:00 / 12:31. Logos for OpenAI and DeepLearning.AI are visible at the bottom.

# Tips for Using Jupyter Notebook

- Always start in the top (first) cell
- Use the **Run** control to execute code in a cell
- If the cell calls to ChatGPT, allow a few seconds for a response to show up
- If you don't get a response after a minute or 2, try rerunning the cell or start from the top again
- If the Jupyter notebook window is empty, refresh your browser
- Use the left button of your mouse to select or go back to an earlier cell
- Edit the code in a cell and try running that



jupyter i6-transforming

```
from dotenv import load_dotenv, find_dotenv
_ = load_dotenv(find_dotenv()) # read local .env file
openai.api_key = os.getenv('OPENAI_API_KEY')

In [2]: def get_completion(prompt, model="gpt-3.5-turbo", temperature=0):
 messages = [{"role": "user", "content": prompt}]
 response = openai.ChatCompletion.create(
 model=model,
 messages=messages,
 temperature=temperature,
)
 return response.choices[0].message["content"]
```

**Translation**

ChatGPT is trained with sources in many languages. This gives the model the ability to do translation. Here are some examples of how to use this capability.

```
In [3]: prompt = f"""
Translate the following English text to Spanish: \
``Hi, I would like to order a blender``.
"""
response = get_completion(prompt)
print(response)

Hola, me gustaría ordenar una licuadora.
```

jupyter i6-transforming

```
from dotenv import load_dotenv, find_dotenv
_ = load_dotenv(find_dotenv()) # read local .env file
openai.api_key = os.getenv('OPENAI_API_KEY')

In [2]: def get_completion(prompt, model="gpt-3.5-turbo", temperature=0):
 messages = [{"role": "user", "content": prompt}]
 response = openai.ChatCompletion.create(
 model=model,
 messages=messages,
 temperature=temperature,
)
 return response.choices[0].message["content"]
```

**Translation**

ChatGPT is trained with sources in many languages. This gives the model the ability to do translation. Here are some examples of how to use this capability.

```
In [4]: prompt = f"""
Translate the following English text to Spanish: \
``Hi, I would like to order a banana``.
"""
response = get_completion(prompt)
print(response)

Hola, me gustaría pedir un plátano.
```

jupyter i6-transforming

File Edit View Insert Cell Kernel Help Not Trusted Python 3 (ipykernel)

**Transforming**

In this notebook, we will explore how to use Large Language Models for text transformation tasks such as language translation, spelling and grammar checking, tone adjustment, and format conversion.

**Setup**

```
In [1]: import openai
import os

from dotenv import load_dotenv, find_dotenv
_ = load_dotenv(find_dotenv()) # read local .env file

openai.api_key = os.getenv('OPENAI_API_KEY')
```

```
In [2]: def get_completion(prompt, model="gpt-3.5-turbo", temperature=0):
 messages = [{"role": "user", "content": prompt}]
 response = openai.ChatCompletion.create(
 model=model,
 messages=messages,
 temperature=temperature,
)
 return response.choices[0].message["content"]
```

**Translation**

ChatGPT is trained with sources in many languages. This gives the model the ability to do translation. Here are some examples of how to use this capability.



# lesson 1 summary

- ChatGPT is a large language model (LLM) that can engage in human-like conversations using a Prompt & Response mechanism
- You can interact with ChatGPT via a web browser (like a Google search) or via an API with a software program
- *ChatGPT Prompt Engineering for Developers* is a 1-hour course from a company called DeepLearning.AI that uses simple code examples, running in Jupyter Notebook, that utilize the ChatGPT API to teach the concepts for using ChatGPT
  - Jupyter Notebook allows you to run and modify the code from the course

# lesson 02

Creating Your Accounts with  
OpenAI and DeepLearning.AI

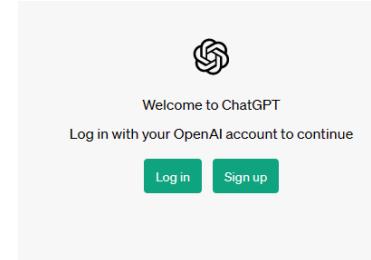


# Creating your OpenAI Account

1. Go to <https://chat.OpenAi.com>
2. Sign up

I had problems creating my account. I kept getting the error, "Signup is currently unavailable, please try again later."

The way I got it to work was to complete the signup on my phone, with Firefox in incognito mode. After that I was able to login normally with Firefox on my PC.



## Create your account

Please note that phone verification is required for signup. Your number will only be used to verify your identity for security purposes.

[Edit](#)

[Resend link](#)

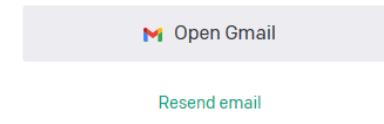
Your password must contain:  
✓ At least 8 characters

[Continue](#)

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

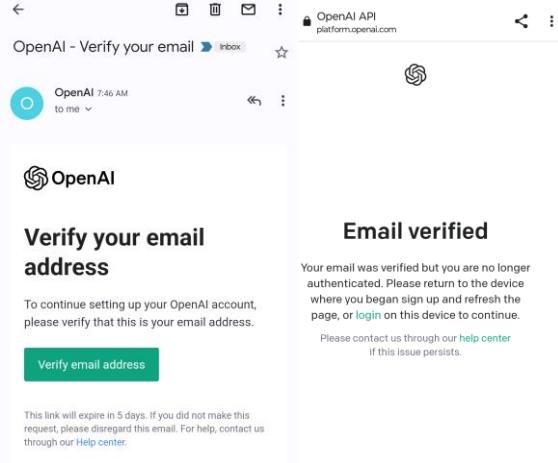
## Verify your email

We sent an email to ██████████@gmail.com. Click the link inside to get started.



[Open Gmail](#)

[Resend email](#)



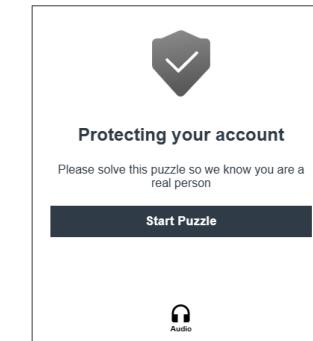
**Tell us about you**

[Continue](#)

By clicking "Continue", you agree to our [Terms](#) and acknowledge our [Privacy policy](#)

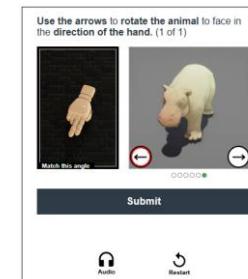
## Verify your phone number

[Send code](#)



Protecting your account  
Please solve this puzzle so we know you are a real person

[Start Puzzle](#)



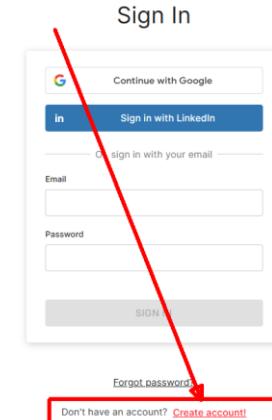
Use the arrows to rotate the animal to face in the direction of the hand. (1 of 1)  
Match this angle  
Submit  
Audio  
Restart

## SMS Verification

# Course Signup

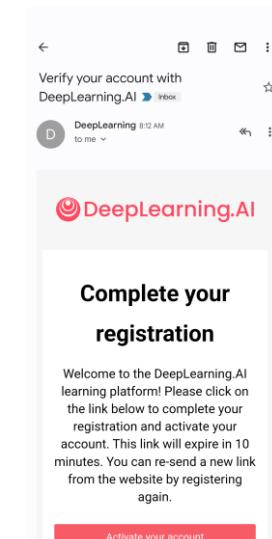
1. Go to [ChatGPT Prompt Engineering for Developers - DeepLearning.AI](#)
2. Select 'Learn for Free' and follow the prompts

<https://www.deeplearning.ai/short-courses/chatgpt-prompt-engineering-for-developers/>



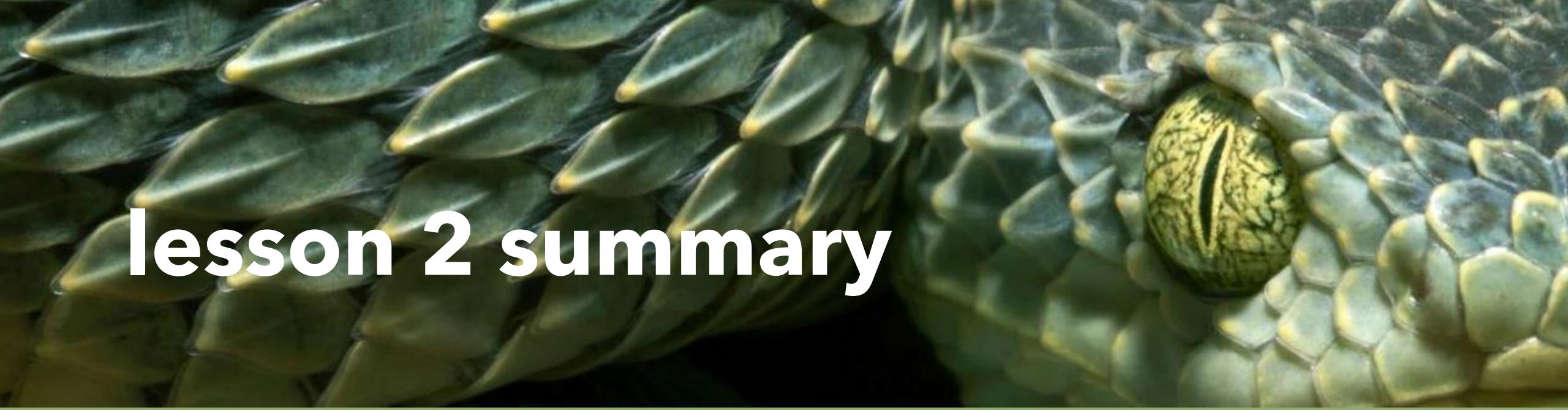
## Create Your Account

A screenshot of a 'Create Your Account' form. It has a 'Email Address' field containing 'm [REDACTED]@gmail.com', a red 'SIGN UP' button, and a note below stating: 'By clicking "SIGN UP", you agree to our [terms of service](#), [privacy policy](#) and [cookie policy](#)'.



A screenshot of a course platform interface. The top bar shows 'DLAI - Learning Platform Beta learn.deeplearning.ai'. The main content area is titled 'One Step Away' and says: 'Please fill in your full name and set a password to complete registration!'. It has fields for 'Name' (containing 'Michael Boucher') and 'Password' (containing '\*\*\*\*\*'). There is a checkbox for 'Subscribe to receive AI news, events and course updates from DeepLearning.AI!' and a red 'SUBMIT' button. On the right, a sidebar lists course sections: 'Introduction', 'Guidelines', 'Iterative', 'Summarizing', 'Inferring', 'Transforming', 'Expanding', 'Chatbot', 'Conclusion', 'Course Feedback', and 'Community'.

Everything you need to know to take the free course,  
ChatGPT Prompt Engineering for developers



# lesson 2 summary

- At this point you've successfully created a ChatGPT account & have signed up for the course, *ChatGPT Prompt Engineering for Developers*
- You are now ready to take the course and play with the course's code examples using the Jupyter Notebook that is running inside the course.
- OPTIONAL: If you want to run and experiment with the code samples outside the course, then you'll need to install a few things on your computer
- *In the next lesson we're going to get your computer set up so that you can run all the examples and follow along with the course using Jupyter Notebook separate from the course. This will enable you to save and rerun the course code on your own at any time without being logged into the course.*

# lesson 03

Setting up Your computer to  
follow along with the course



# Lesson 3

## topics

Install  
Anaconda/Jupyter  
Notebook

Install 3 libraries

- python-dotenv
- openai
- Redlines

Setup  
your  
OpenAI  
key

# Why do I have to install stuff on my computer?

- The course runs Python code examples using a tool called Jupyter Notebook
  - Jupyter Notebook is a handy way to write programs in small steps
  - Jupyter Notebook (and Python) are used a lot by Data Scientists
  - Anaconda is the tool you install to get Jupyter Notebook
- The course uses a few Python libraries that don't come with a stock Python install
  - We're going to install the libraries ahead of time, so you don't get errors when you are follow along in the course
- To connect your program to ChatGPT you need an API key that's unique to you, so you'll get the key from OpenAPI and install it on your computer

# topic 1 of 3

## Installing Anaconda / Jupyter Notebook

### Steps

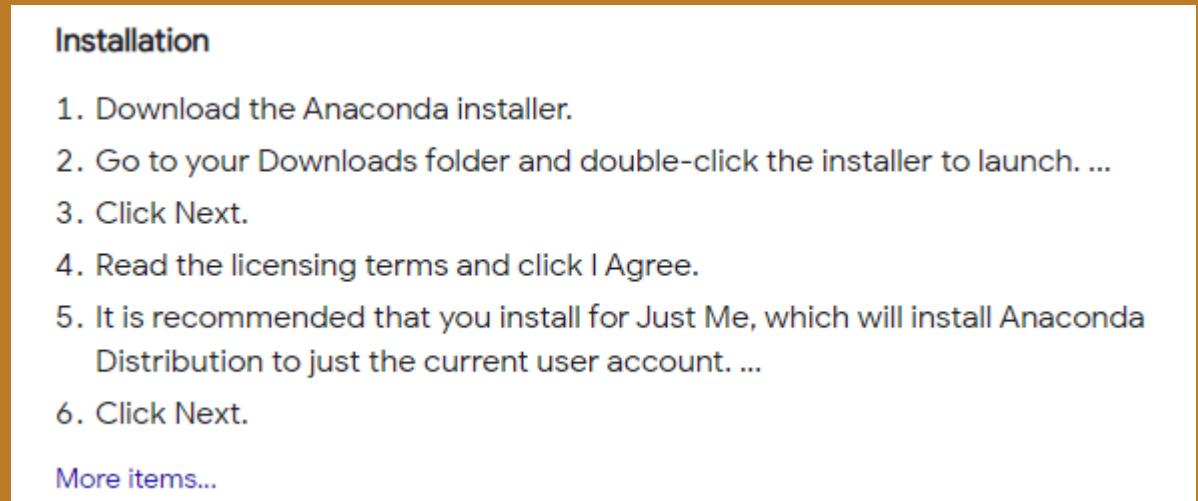
1. Install Anaconda
2. Launch Anaconda
3. Create a working project folder
4. Launch Jupyter Notebook
5. Create your 1<sup>st</sup> Jupyter Notebook

# Progress

1. Install Anaconda / Jupyter Notebook
2. Install 3 Python libraries
  - python-dotenv
  - openai
  - Redlines
3. Setup your OpenAI key

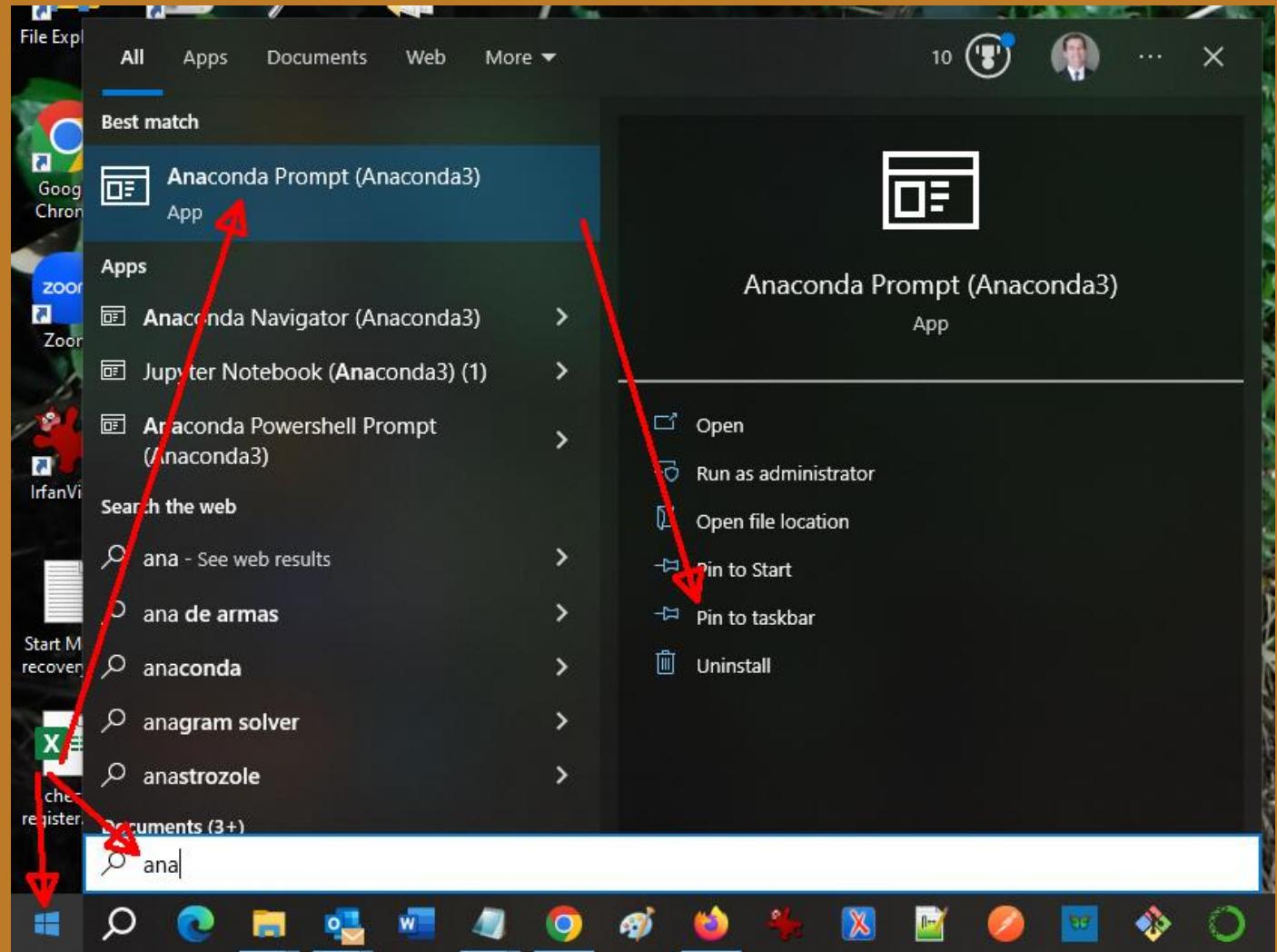
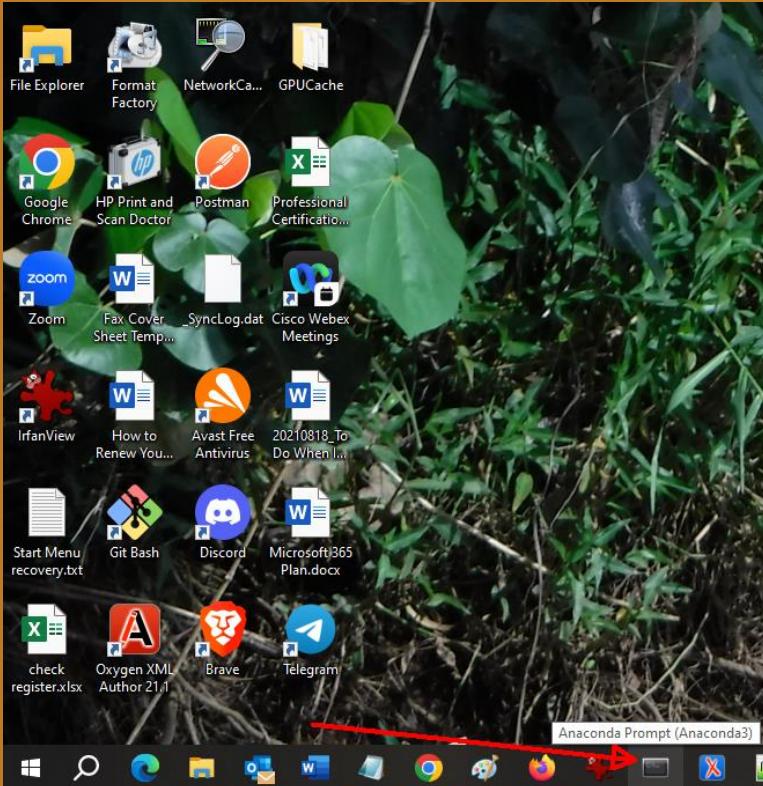
# Install Anaconda / Jupyter Notebook

- Follow the instructions from here:  
<https://docs.anaconda.com/free/anaconda/install/windows/>
- Accept the defaults
  - Install for **Just Me** (not for all users)
  - Do not add Anaconda to your PATH environment variable
- When done, create a shortcut



# Pin Anaconda to the Task Bar

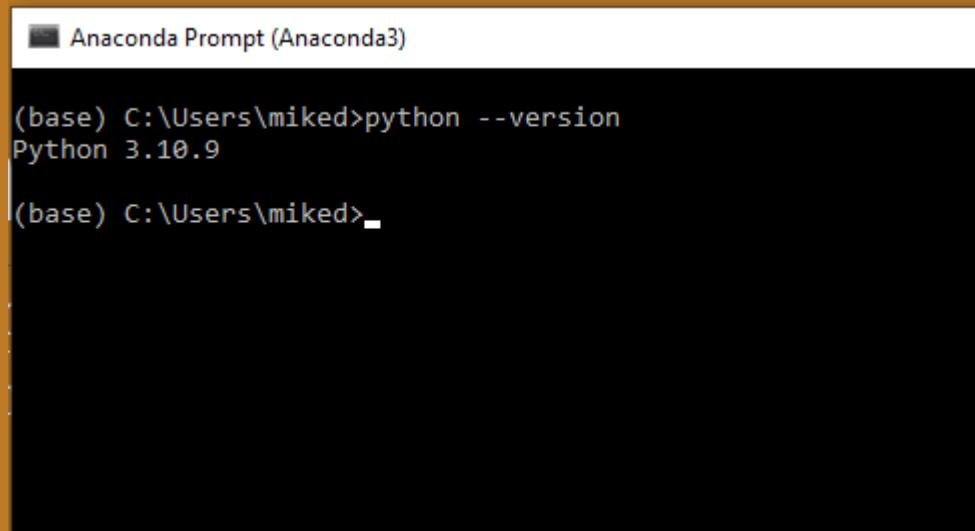
1. Windows Start
  2. “ana”
  3. Pin to taskbar



# Verify Python Version

Open an Anaconda prompt from the task bar and then run the following command to verify that you're running Python 3.10 (or later)

*python --version*



The screenshot shows a Windows taskbar at the bottom with several pinned icons. Above it is a dark-themed Anaconda Prompt window titled "Anaconda Prompt (Anaconda3)". The window contains the following text:  
(base) C:\Users\miked>python --version  
Python 3.10.9  
(base) C:\Users\miked>\_

# Create a Project folder, then open Jupyter Notebook

Open an Anaconda prompt from the task bar and then run the following commands:

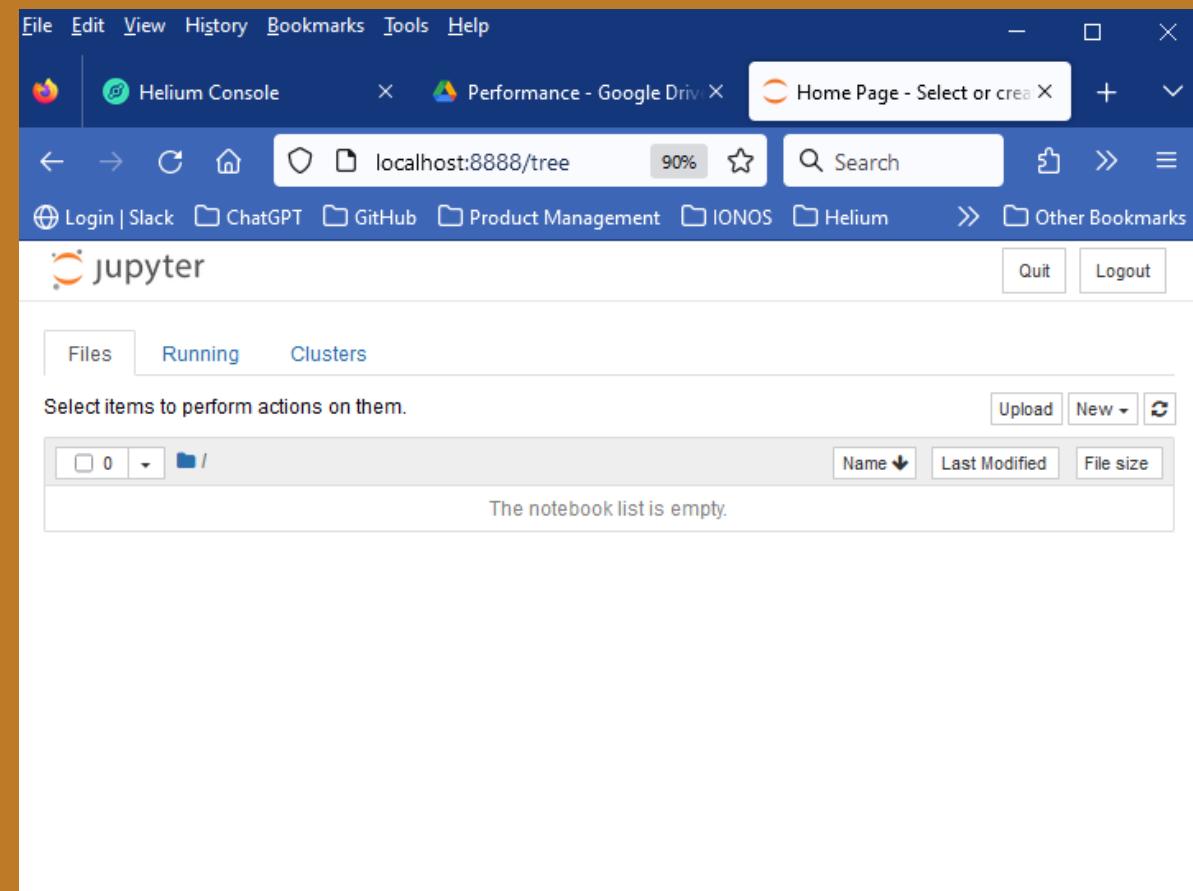
1. *dir*
2. *mkdir myprojects*
3. *cd myprojects*
4. *mkdir openai*
5. *cd openai*
6. *jupyter notebook*

```
Anaconda Prompt (Anaconda3) - jupyter notebook

(base) C:\Users\miked>mkdir myprojects
(base) C:\Users\miked>cd myprojects
(base) C:\Users\miked\myprojects>mkdir openai
(base) C:\Users\miked\myprojects>cd openai

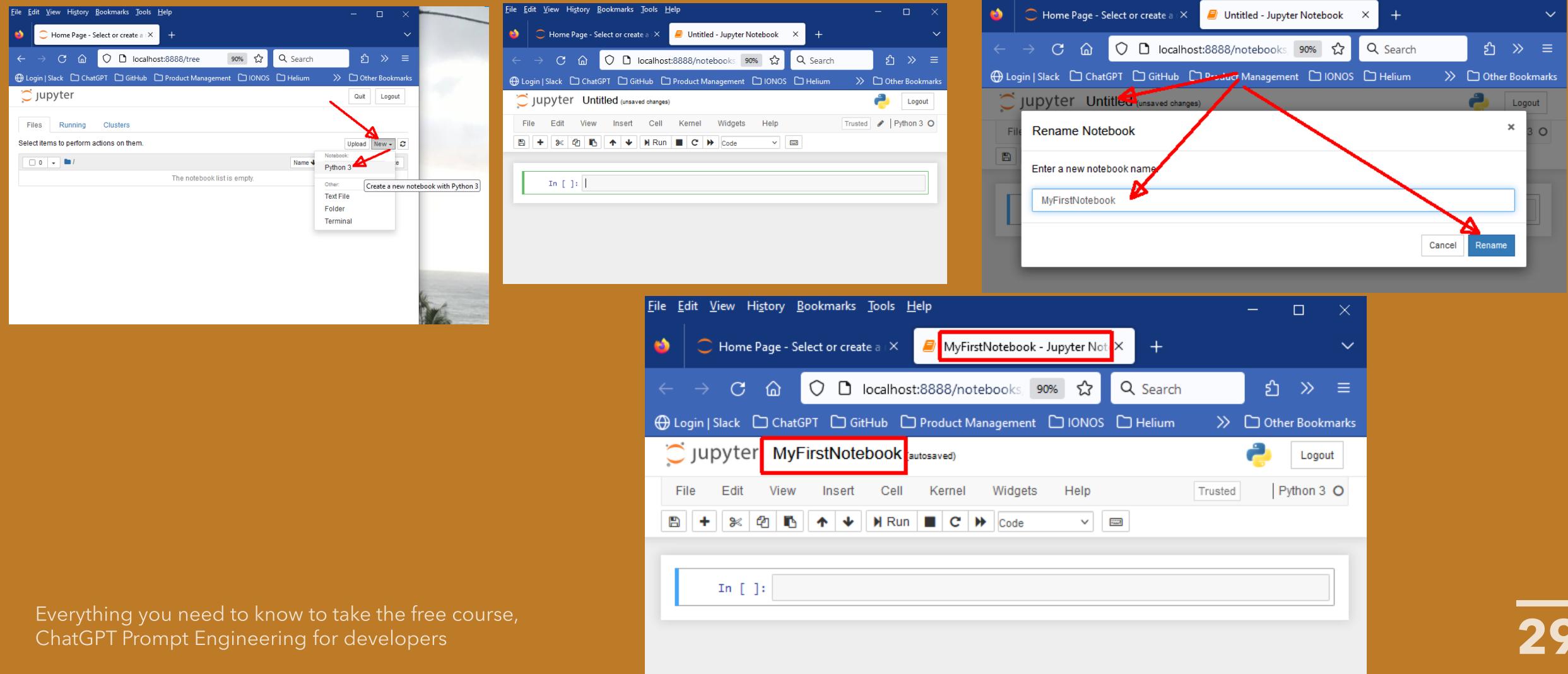
(base) C:\Users\miked\myprojects\openai>jupyter notebook
[I 08:06:11.692 NotebookApp] JupyterLab extension loaded from C:\Users\miked\Anaconda3\lib\site-packages\jupyterlab
[I 08:06:11.692 NotebookApp] JupyterLab application directory is C:\Users\miked\Anaconda3\share\jupyter\lab
[I 08:06:11.694 NotebookApp] Serving notebooks from local directory: C:\Users\miked\myprojects\openai
[I 08:06:11.694 NotebookApp] The Jupyter Notebook is running at:
[I 08:06:11.694 NotebookApp] http://localhost:8888/?token=ce990b16f3cf65178807e69e326bac982b990a37eb4014e
[I 08:06:11.695 NotebookApp] or http://127.0.0.1:8888/?token=ce990b16f3cf65178807e69e326bac982b990a37eb4014e
[I 08:06:11.695 NotebookApp] Use Control-C to stop this server and shut down all kernels (twice to skip confirmation).
[C 08:06:11.864 NotebookApp]

To access the notebook, open this file in a browser:
 file:///C:/Users/miked/AppData/Roaming/jupyter/runtime/nbserver-7820-open.html
Or copy and paste one of these URLs:
 http://localhost:8888/?token=ce990b16f3cf65178807e69e326bac982b990a37eb4014e
 or http://127.0.0.1:8888/?token=ce990b16f3cf65178807e69e326bac982b990a37eb4014e
```



# Create a new Jupyter Notebook

New > Python3 to create the notebook, then rename the notebook



# Congrats

You're now running Anaconda and Jupyter Notebook on your PC and have created your first jupyter notebook!

Everything you need to know to take the free course,  
ChatGPT Prompt Engineering for developers



# Progress

1. Install Anaconda / Jupyter Notebook

2. Install 3 Python libraries

- python-dotenv
- openai
- Redlines

3. Setup your OpenAI key

# topic 3 of 3

Installing 3 Python libraries

## Steps

1. Install the Python library, python-dotenv
2. Install the Python library, openai
3. Install the Python library, Redlines



# Installing python-dotenv

Trying to run the first bit of code from the course, I got a dotenv error.

```
In [1]: import openai
import os

from dotenv import load_dotenv, find_dotenv
_ = load_dotenv(find_dotenv())

openai.api_key = os.getenv('OPENAI_API_KEY')

ModuleNotFoundError Traceback (most recent call last)
<ipython-input-1-0863633b1d8c> in <module>
 2 import os
 3
----> 4 from dotenv import load_dotenv, find_dotenv
 5 _ = load_dotenv(find_dotenv())
 6

ModuleNotFoundError: No module named 'dotenv'
```

Solution: from the Anaconda prompt install the library, python-dotenv with the following command:

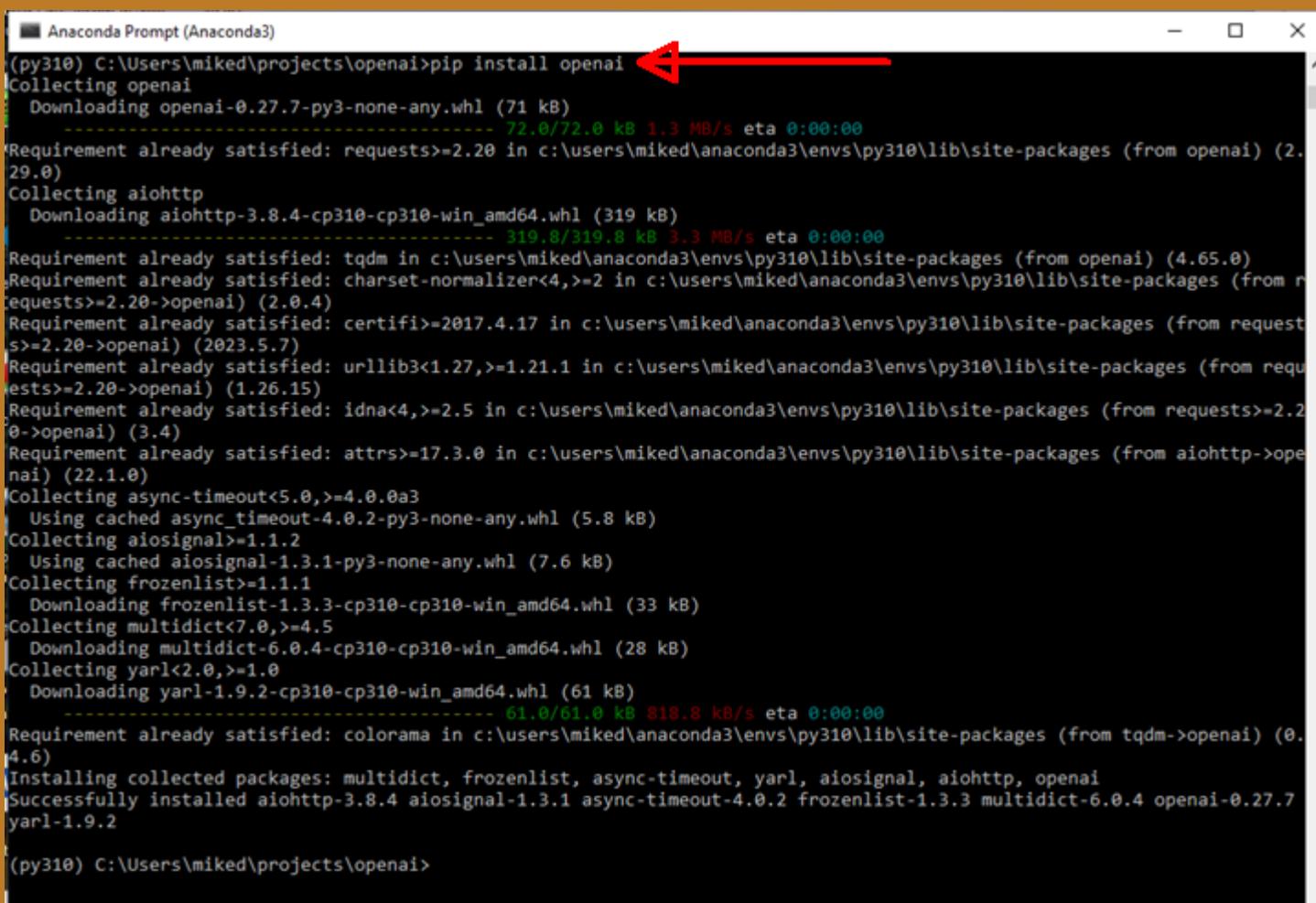
```
python -m pip install python-dotenv
```

```
[I 07:50:19.476 NotebookApp] Kernel restarted: ef0b658d-6290-4def-b916-d2cc4ae94ac9
[I 07:50:20.518 NotebookApp] Restoring connection for ef0b658d-6290-4def-b916-d2cc4ae94ac9:0b5c6674472d485ccb0f2be266033
[base] C:\Users\miked\projects\openai>python -m pip install python-dotenv
Collecting python-dotenv
 Downloading https://files.pythonhosted.org/packages/64/62/f19d1e9023aacb47241de3ab5a5d5fedf32c78a71a9e365bb2153378c141
/python_dotenv-0.21.1-py3-none-any.whl
Installing collected packages: python-dotenv
Successfully installed python-dotenv-0.21.1
(base) C:\Users\miked\projects\openai>
```

# Installing openai

From the Anaconda prompt install the library, openai with the following command:

**pip install openai**



```
Anaconda Prompt (Anaconda3)
(py310) C:\Users\miked\projects\openai>pip install openai
Collecting openai
 Downloading openai-0.27.7-py3-none-any.whl (71 kB)
 72.0/72.0 kB 1.3 MB/s eta 0:00:00
Requirement already satisfied: requests>=2.20 in c:\users\miked\anaconda3\envs\py310\lib\site-packages (from openai) (2.29.0)
Collecting aiohttp
 Downloading aiohttp-3.8.4-cp310-cp310-win_amd64.whl (319 kB)
 319.8/319.8 kB 3.3 MB/s eta 0:00:00
Requirement already satisfied: tqdm in c:\users\miked\anaconda3\envs\py310\lib\site-packages (from openai) (4.65.0)
Requirement already satisfied: charset-normalizer<4,>=2 in c:\users\miked\anaconda3\envs\py310\lib\site-packages (from requests>=2.20->openai) (2.0.4)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\miked\anaconda3\envs\py310\lib\site-packages (from requests>=2.20->openai) (2023.5.7)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in c:\users\miked\anaconda3\envs\py310\lib\site-packages (from requests>=2.20->openai) (1.26.15)
Requirement already satisfied: idna<4,>=2.5 in c:\users\miked\anaconda3\envs\py310\lib\site-packages (from requests>=2.20->openai) (3.4)
Requirement already satisfied: attrs>=17.3.0 in c:\users\miked\anaconda3\envs\py310\lib\site-packages (from aiohttp>openai) (22.1.0)
Collecting async-timeout<5.0,>=4.0.0a3
 Using cached async_timeout-4.0.2-py3-none-any.whl (5.8 kB)
Collecting aiosignal>=1.1.2
 Using cached aiосignal-1.3.1-py3-none-any.whl (7.6 kB)
Collecting frozenlist>1.1.1
 Downloading frozenlist-1.3.3-cp310-cp310-win_amd64.whl (33 kB)
Collecting multidict<7.0,>=4.5
 Downloading multidict-6.0.4-cp310-cp310-win_amd64.whl (28 kB)
Collecting yarl<2.0,>=1.0
 Downloading yarl-1.9.2-cp310-cp310-win_amd64.whl (61 kB)
 61.0/61.0 kB 818.8 kB/s eta 0:00:00
Requirement already satisfied: colorama in c:\users\miked\anaconda3\envs\py310\lib\site-packages (from tqdm>openai) (0.4.6)
Installing collected packages: multidict, frozenlist, async-timeout, yarl, aiосignal, aiohttp, openai
Successfully installed aiohttp-3.8.4 aiосignal-1.3.1 async-timeout-4.0.2 frozenlist-1.3.3 multidict-6.0.4 openai-0.27.7 yarl-1.9.2

(py310) C:\Users\miked\projects\openai>
```

# Installing Redlines

The course chapter on Transforming requires the Redlines python package. Here's the error you get when you try to use Redlines without having it installed.

```
In [27]: from redlines import Redlines

diff = Redlines(text,response)
display(Markdown(diff.output_markdown))

ModuleNotFoundError Traceback (most recent call last)
<ipython-input-27-4c4c6ae221b6> in <module>
----> 1 from redlines import Redlines
 2
 3 diff = Redlines(text,response)
 4 display(Markdown(diff.output_markdown))

ModuleNotFoundError: No module named 'redlines'
```

Solution: from the Anaconda prompt install the library, Redlines, with the following command:

```
pip install redlines
```

```
C:\Users\miked\projects\openai>pip install redlines
Collecting redlines
 Downloading redlines-0.3.0-py3-none-any.whl (5.5 kB)
Installing collected packages: redlines
Successfully installed redlines-0.3.0

[notice] A new release of pip available: 22.3 -> 23.1.2
[notice] To update, run: python.exe -m pip install --upgrade pip
C:\Users\miked\projects\openai>
```

Everything you need to know to take the free course,  
ChatGPT Prompt Engineering for developers

# Progress

1. Install Anaconda / Jupyter Notebook
2. Install 3 Python libraries
  - python-dotenv
  - openai
  - Redlines
3. Setup your OpenAI key

# topic 3 of 3

Getting and Installing your  
OpenAI Key

## Steps

1. Create a Key in your OpenAI account
2. Put the Key into an environment variable on your PC

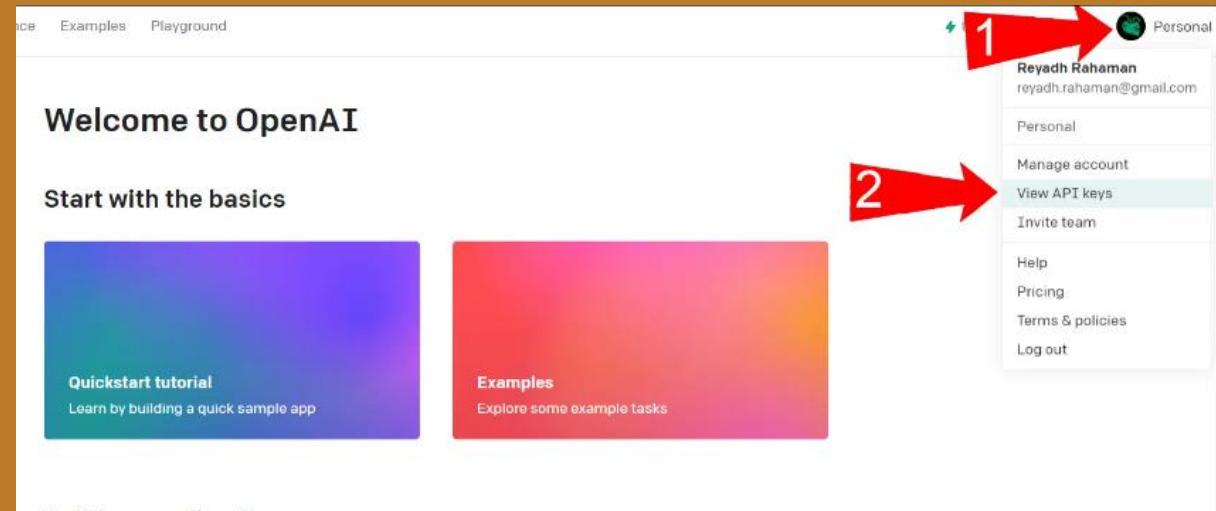


# Getting Your OpenAI Key

Based on instructions from here:

<https://www.howtogeek.com/885918/how-to-get-an-openai-api-key/>

1. Go to OpenAI's Platform website at <https://platform.openai.com> and sign in with your OpenAI account.
2. Click your profile icon at the top-right corner of the page and select "View API Keys"
3. Click "Create New Secret Key" to generate a new API key
4. Name the key "My First openai Key"
5. Copy the key to Notepad or any other handy text editor

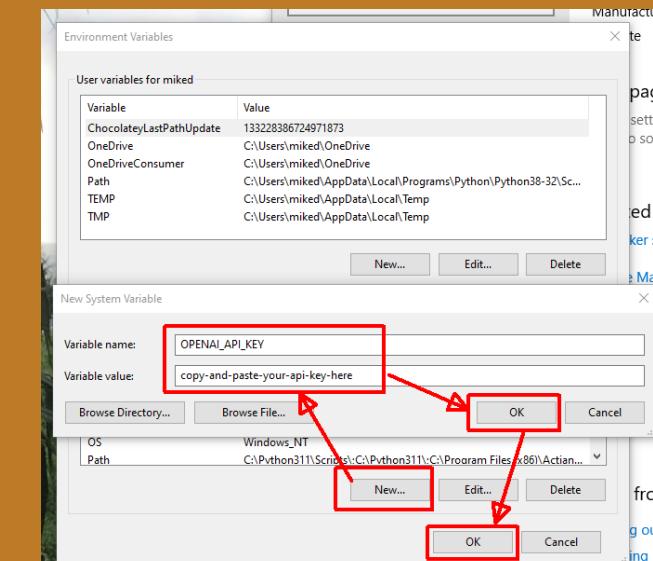
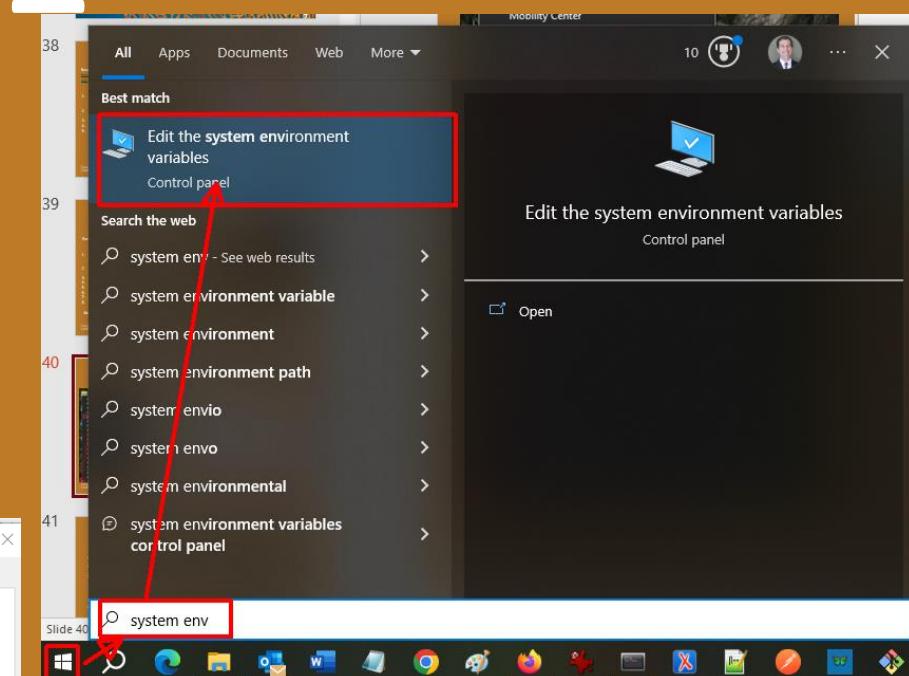
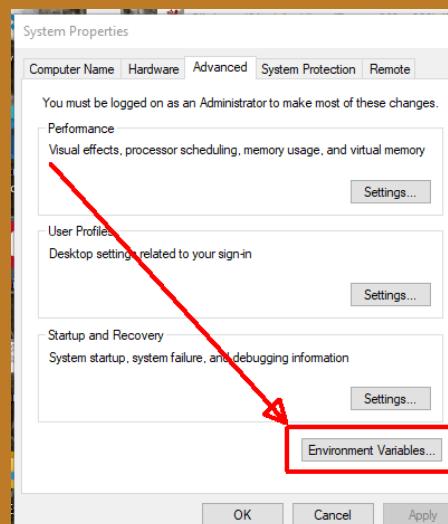
A screenshot of the 'API keys' section within the OpenAI Platform settings. The heading 'API keys' is at the top. Below it, a message says 'Your secret API keys are listed below. Please note that we do not display your secret API keys again after you generate them.' Another message below that says 'Do not share your API key with others, or expose it in the browser or other client-side code. In order to protect the security of your account, OpenAI may also automatically rotate any API key that we've found has leaked publicly.' A red arrow points to the '+ Create new secret key' button. Below this, there is a 'Default organization' section with a dropdown menu set to 'Personal'. A note at the bottom states 'Note: You can also specify which organization to use for each API request. See [Authentication](#) to learn more.' The page number '38' is located in the bottom right corner.

# Creating an OPENAI\_API\_KEY environment variable

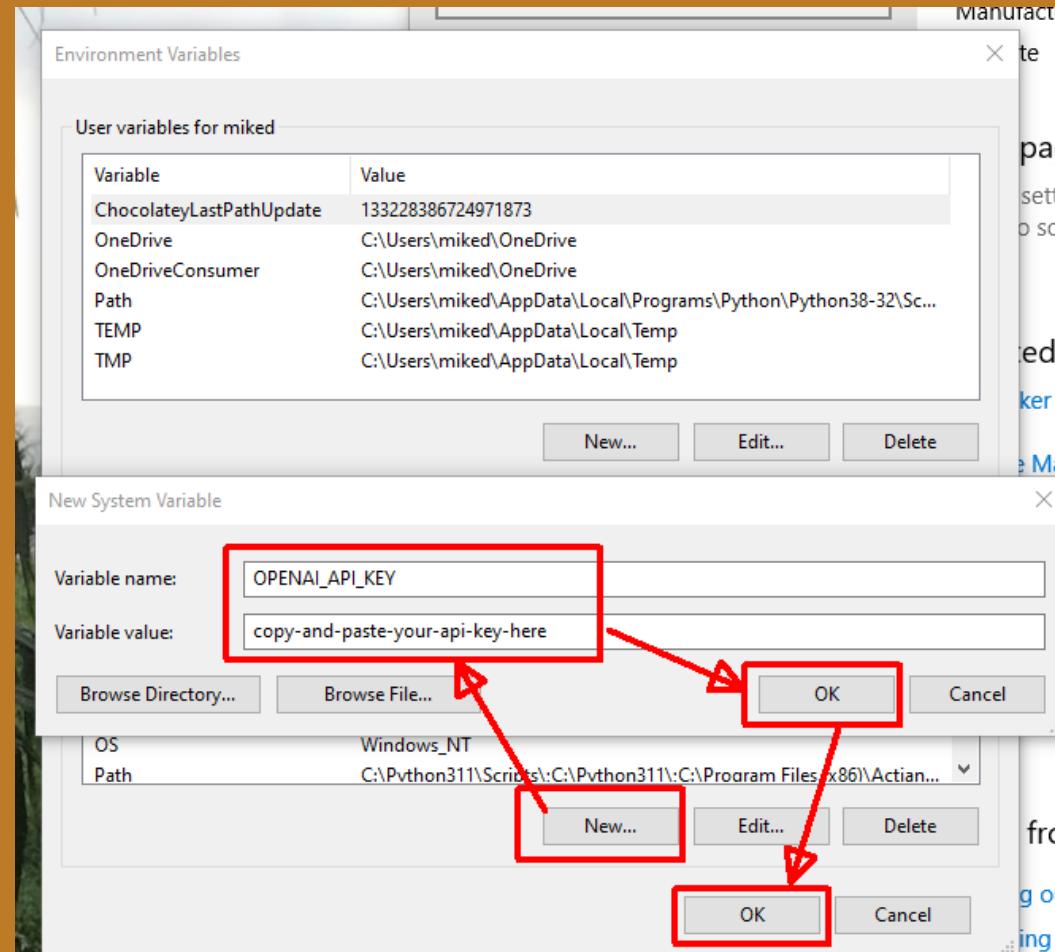
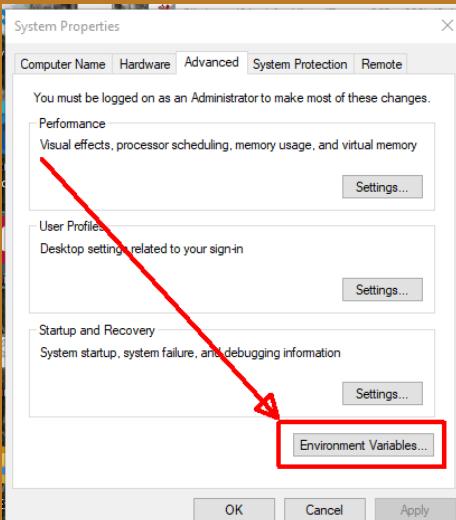
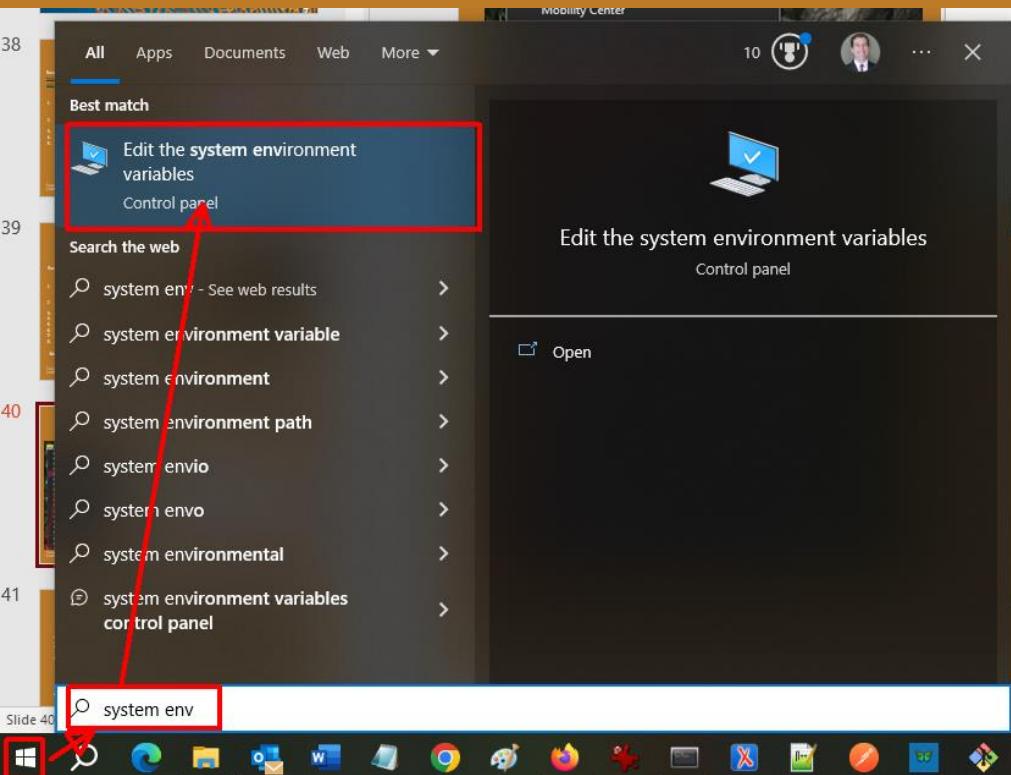
Based on instructions from [here](#):

1. On the Windows taskbar, right-click the Windows icon type 'System env' then select 'Edit the system environment variables'
  1. If you don't see the option to edit environment variables, search for
2. On the Advanced tab, click **Environment Variables**.
3. Under System Variables, click **New** to create a new environment variable.
4. Set **Variable Name** to OPENAI\_API\_KEY
5. Set Variable Value to the key you got from your OpenAI account
6. Select **OK**
7. Select **OK**

See bigger screenshots on the next slide

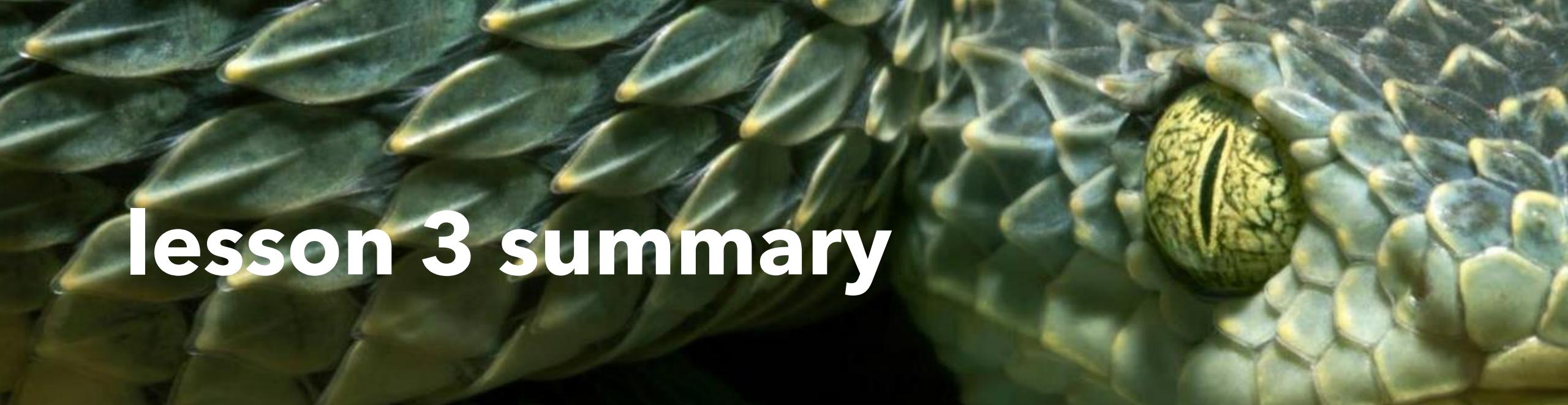


# Bigger Screenshots



# Progress

1. Install Anaconda / Jupyter Notebook
2. Install Python 3.10 in Anaconda
3. Install 3 Python libraries
  - python-dotenv
  - openai
  - Redlines
4. Setup your OpenAI key



# lesson 3 summary

- You've now set up the following tools on your computer
  - Installed Anaconda /Jupyter Notebook
  - Installed some extra Python libraries the course uses
  - Installed an OpenAI key so you programs can make calls to ChatGPT
- Your computer should be set up to run all the course examples
- *In the next lesson we're going to run through the basics of using Jupyter Notebook so you'll be comfortable following along with the course*

# Congrats

You've installed the extra Python libraries needed to run all the course examples

AND

You've installed an OpenAI Key on your PC so your programs can call ChatGPT

Everything you need to know to take the free course,  
ChatGPT Prompt Engineering for developers



# lesson 04

Using Jupyter Notebook



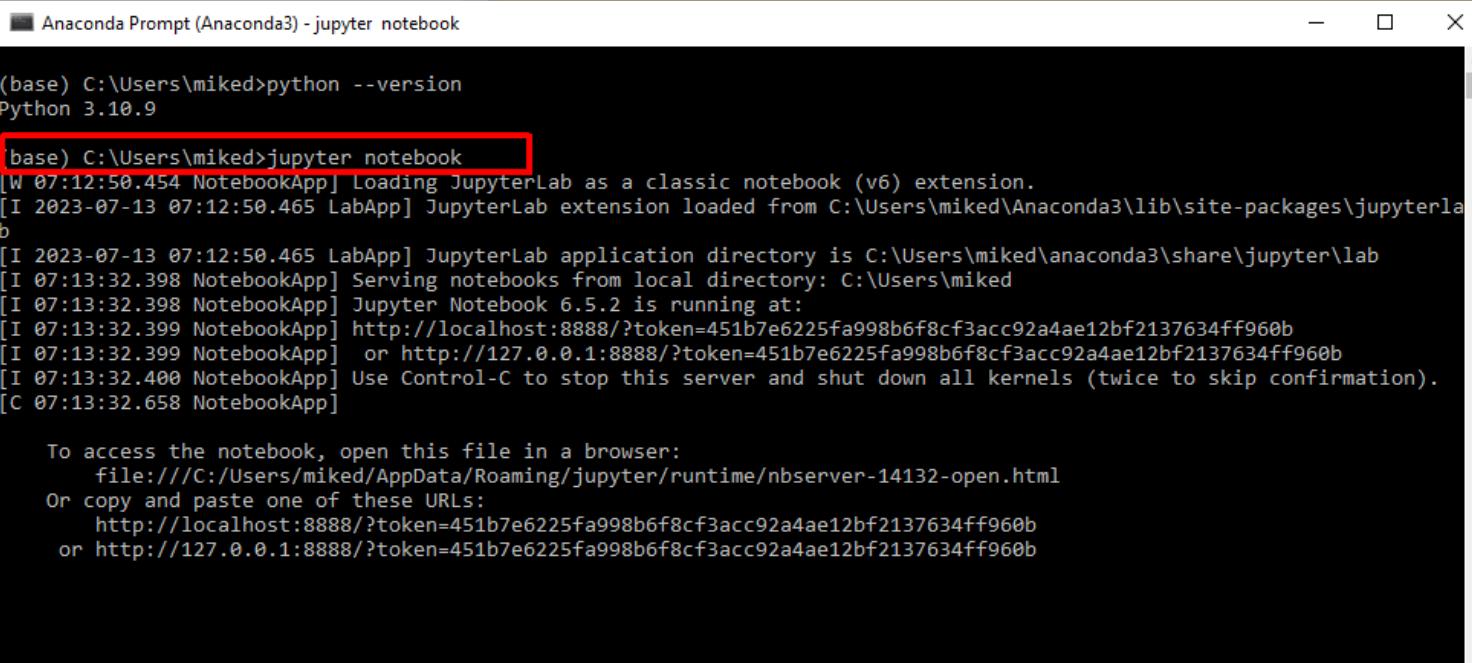
# Using Jupyter Notebook

1. Starting Jupyter Notebook from an Anaconda prompt
2. Creating a new notebook
3. Renaming a notebook
4. Saving a notebook
5. Closing a notebook
6. Opening an existing notebook
7. Adding code to a notebook
8. Deleting a notebook cell
9. Running code in a notebook

# Starting Jupyter Notebook

Open an Anaconda prompt, then run the following command:

*jupyter notebook*

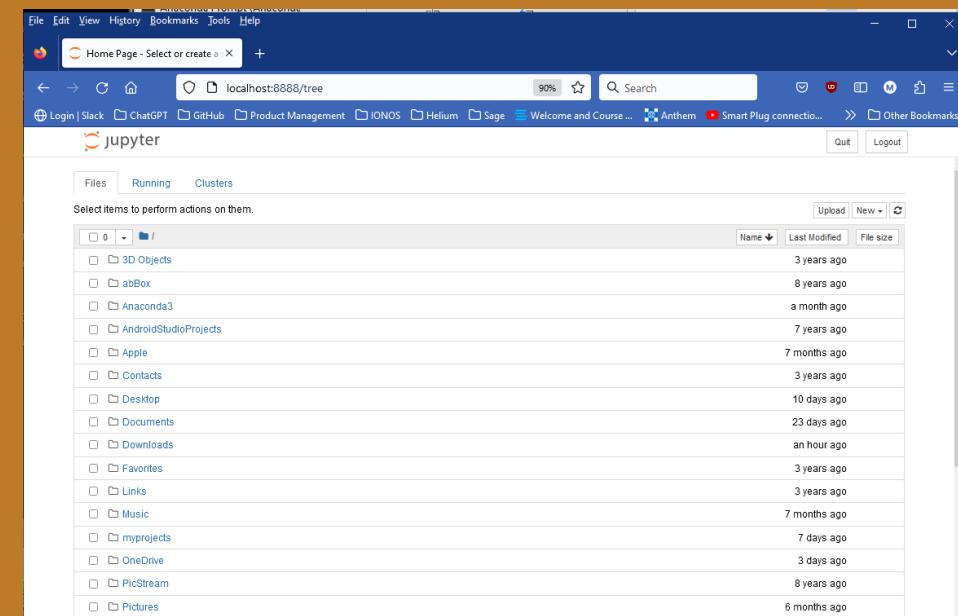


```
Anaconda Prompt (Anaconda3) - jupyter notebook

(base) C:\Users\miked>python --version
Python 3.10.9

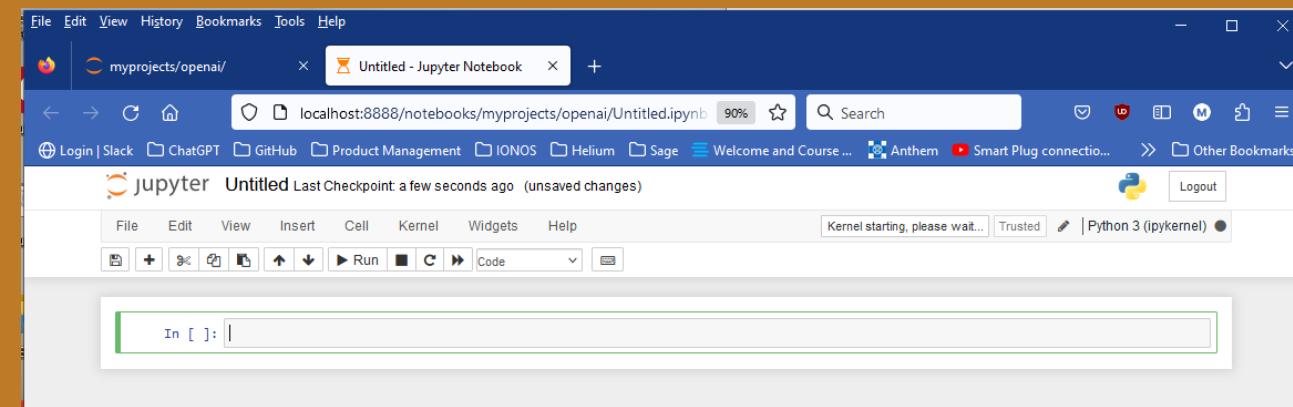
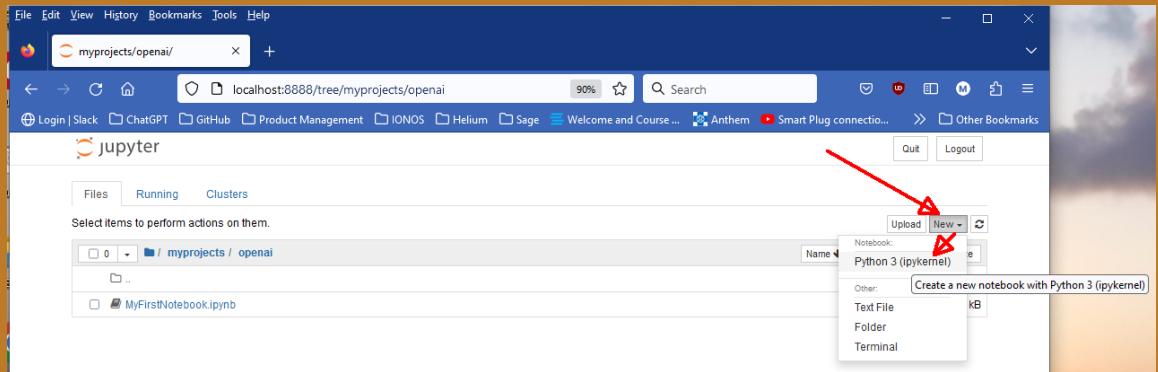
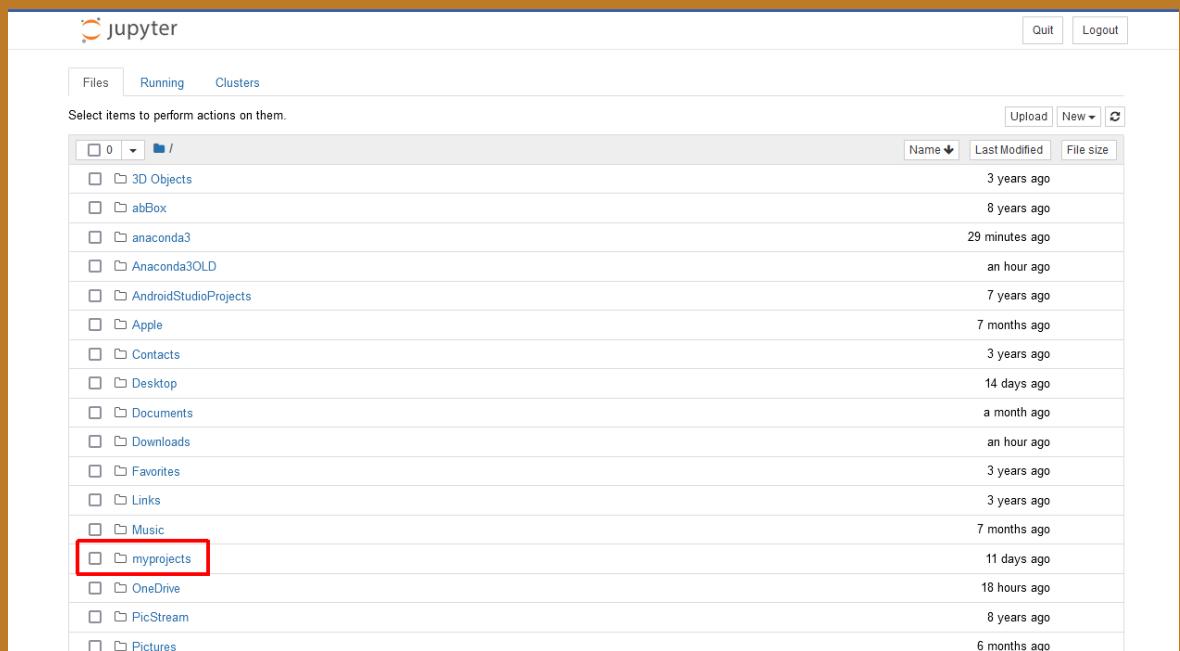
(base) C:\Users\miked>jupyter notebook
[W 07:12:50.454 NotebookApp] Loading JupyterLab as a classic notebook (v6) extension.
[I 2023-07-13 07:12:50.465 LabApp] JupyterLab extension loaded from C:\Users\miked\Anaconda3\lib\site-packages\jupyterlab
[I 2023-07-13 07:12:50.465 LabApp] JupyterLab application directory is C:\Users\miked\anaconda3\share\jupyter\lab
[I 07:13:32.398 NotebookApp] Serving notebooks from local directory: C:\Users\miked
[I 07:13:32.398 NotebookApp] Jupyter Notebook 6.5.2 is running at:
[I 07:13:32.399 NotebookApp] http://localhost:8888/?token=451b7e6225fa998b6f8cf3acc92a4ae12bf2137634ff960b
[I 07:13:32.399 NotebookApp] or http://127.0.0.1:8888/?token=451b7e6225fa998b6f8cf3acc92a4ae12bf2137634ff960b
[I 07:13:32.400 NotebookApp] Use Control-C to stop this server and shut down all kernels (twice to skip confirmation).
[C 07:13:32.658 NotebookApp]

To access the notebook, open this file in a browser:
file:///C:/Users/miked/AppData/Roaming/jupyter/runtime/nbserver-14132-open.html
Or copy and paste one of these URLs:
http://localhost:8888/?token=451b7e6225fa998b6f8cf3acc92a4ae12bf2137634ff960b
or http://127.0.0.1:8888/?token=451b7e6225fa998b6f8cf3acc92a4ae12bf2137634ff960b
```



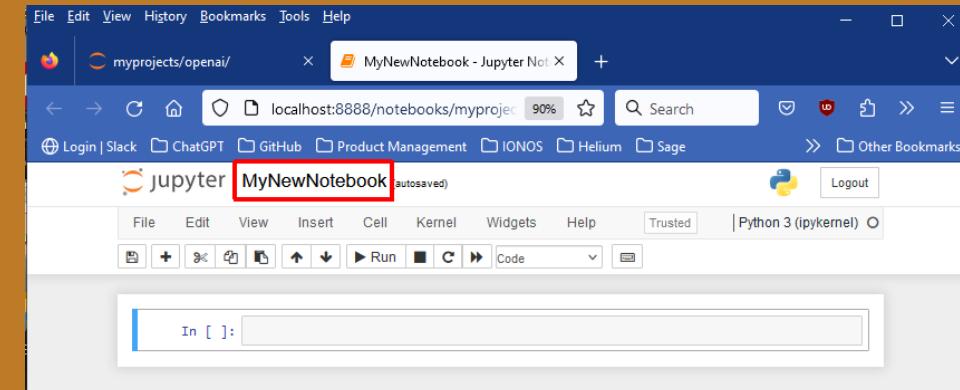
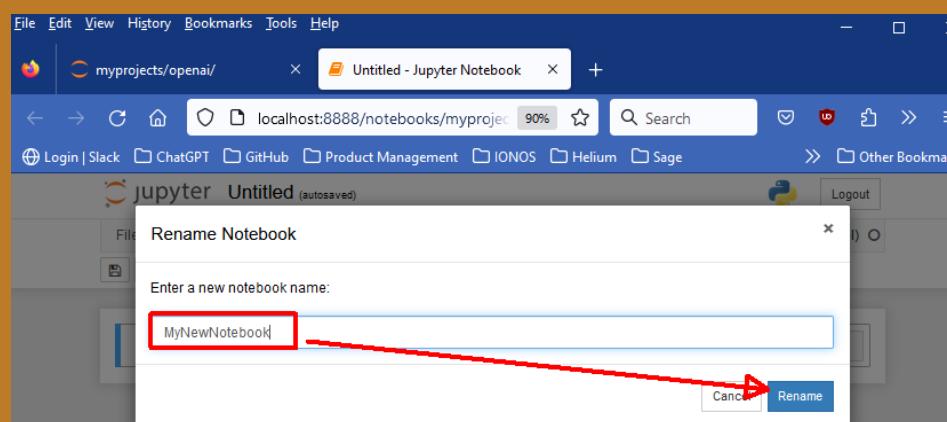
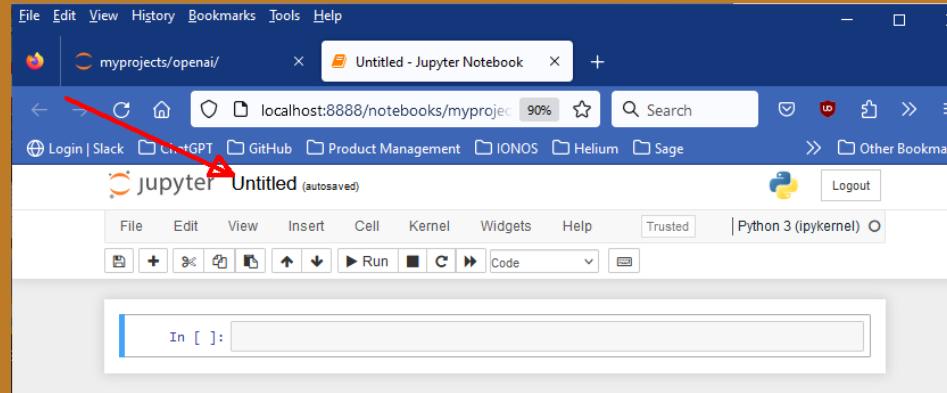
# Creating a new Notebook

1. Open Jupyter Notebook
2. Navigate to the folder where you want to create the notebook
  - Myprojects > openai
3. New > Python 3



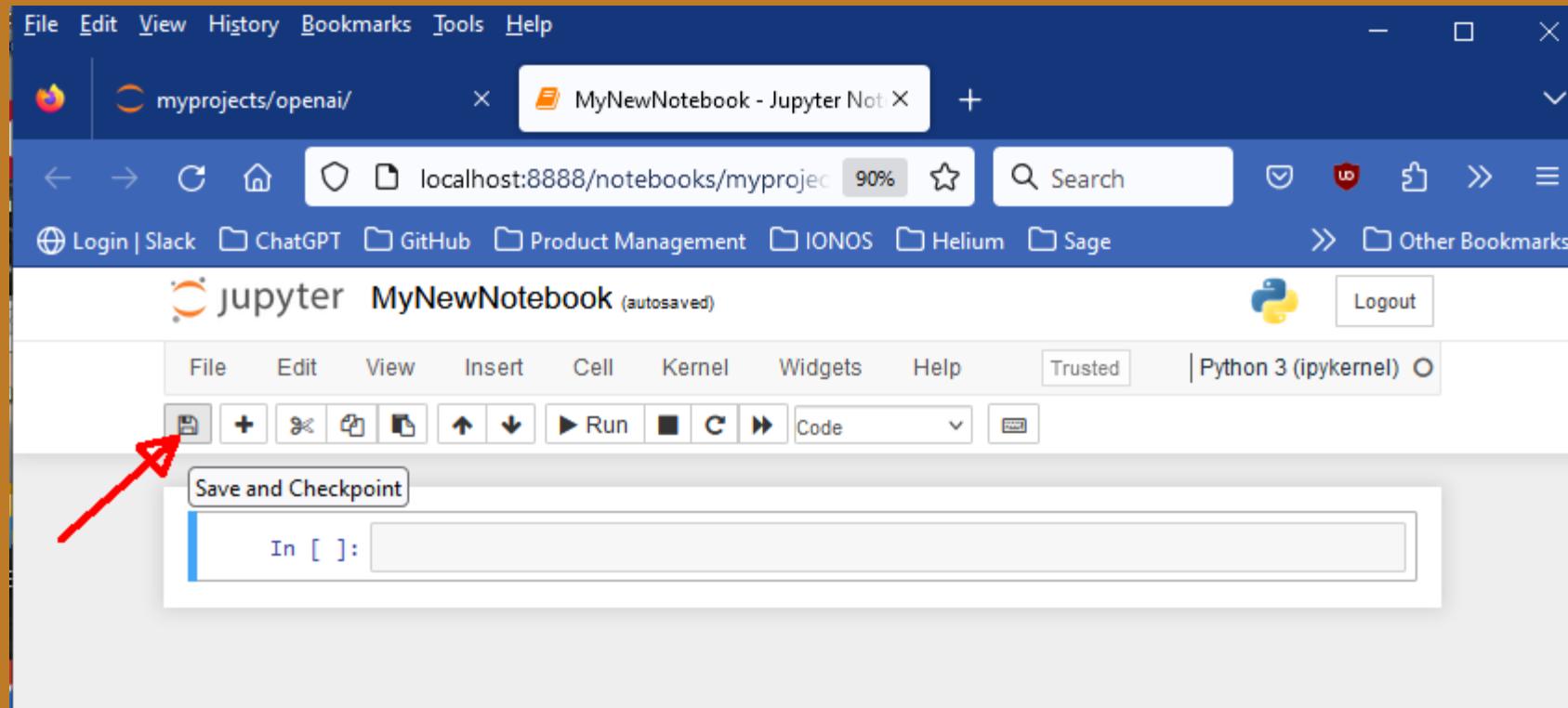
# Renaming a Notebook

1. Click on the title to bring up the Rename Notebook dialog box
2. Enter a new notebook name, *1 Guidelines for Prompting*, then select Rename



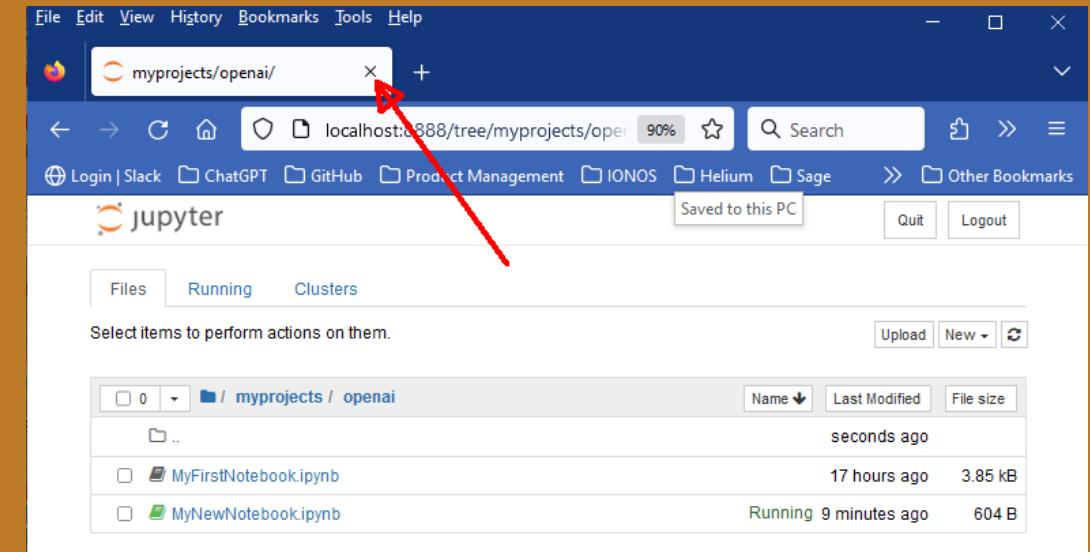
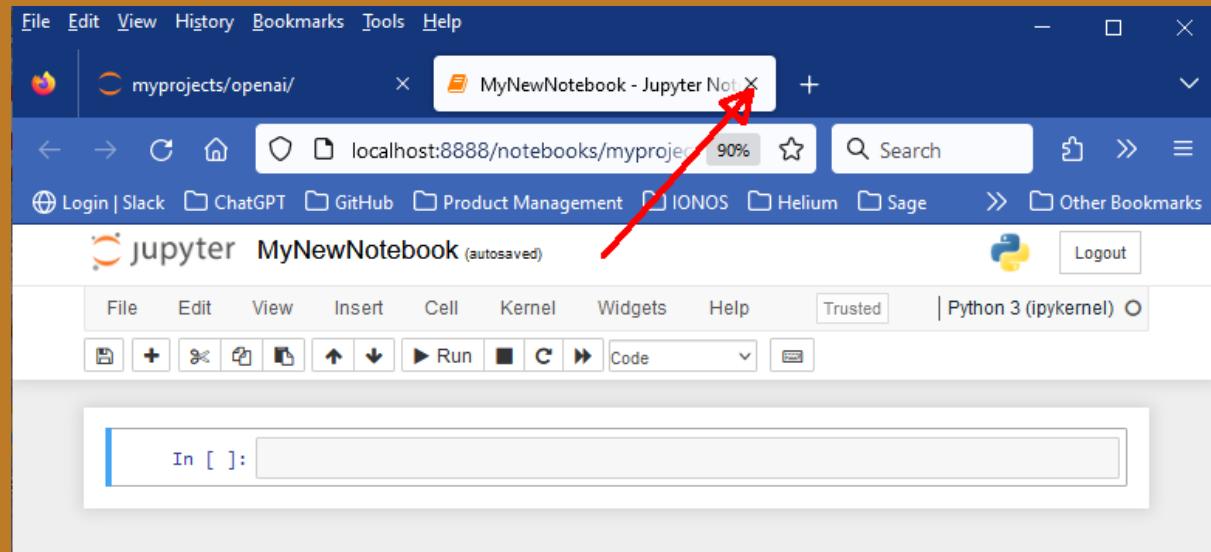
# Saving a Notebook

Use the  icon to save your changes (Jupyter Notebook periodically auto-saves too)



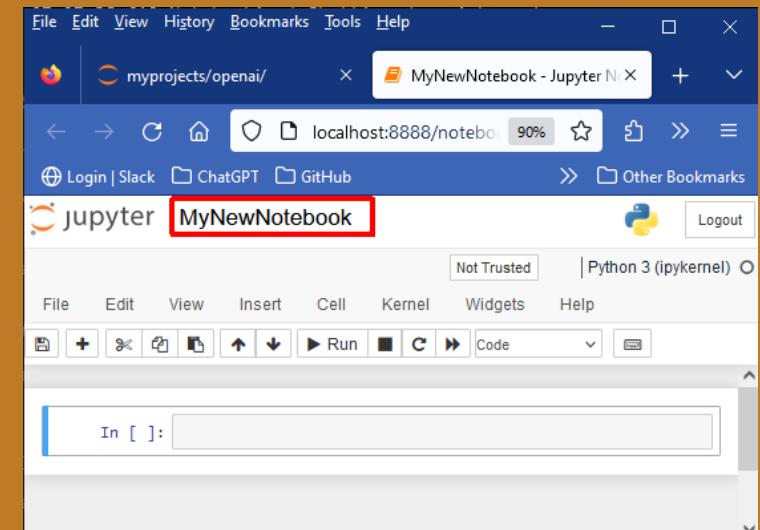
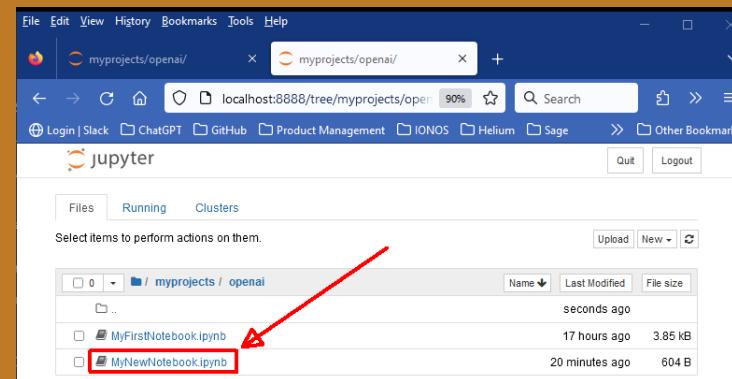
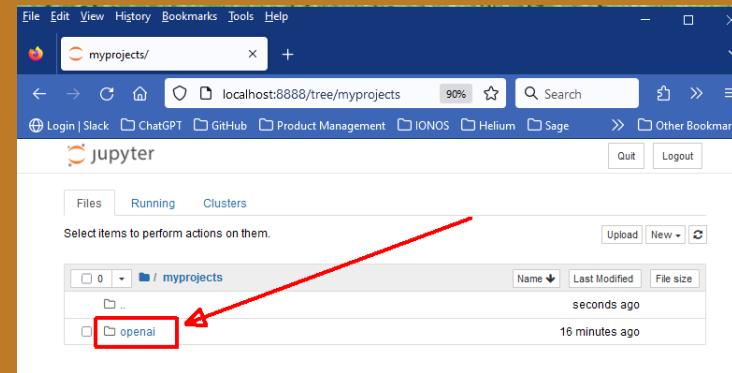
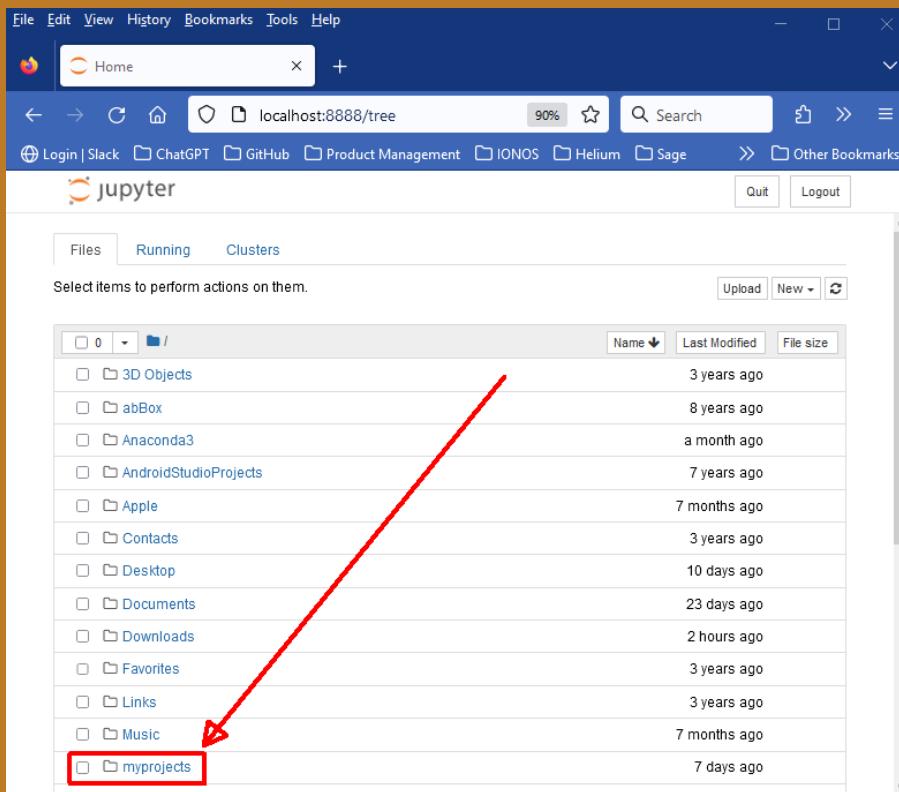
# Closing a Notebook

1. Close the browser window running the Notebook
2. (optional) Close the browser window showing the notebook listing
3. (optional) Close the Anaconda prompt window



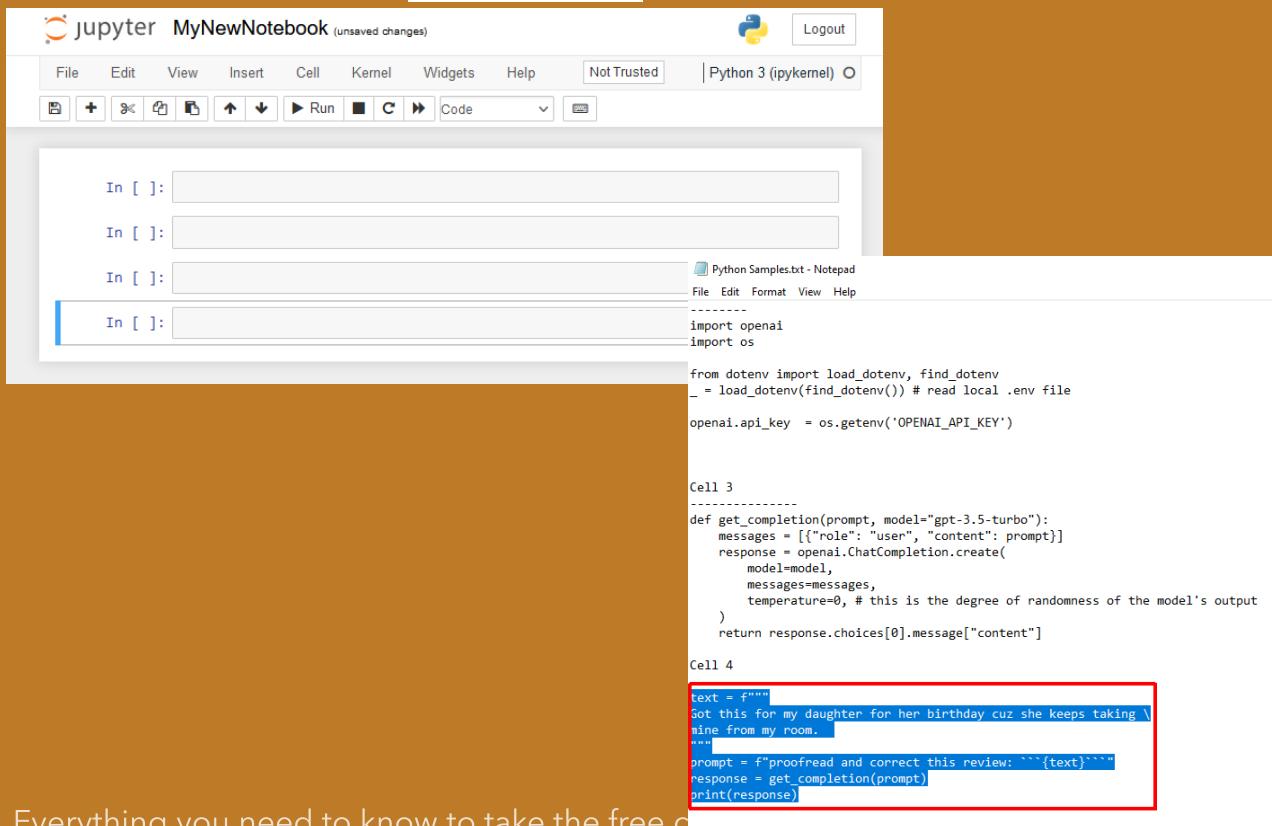
# Opening an existing Notebook

1. Open Jupyter Notebook (start Anaconda, then run `jupyter notebook`)
2. Navigate to the folder containing the notebook
3. Double-click the notebook



# Adding Code to a Notebook

1. Use the  icon to add additional cells
2. Write or paste some code into the cells
3. Use the  Run icon to run/step thru 1 cell at a time (start from the 1<sup>st</sup> cell)



In [ ]:

In [ ]:

In [ ]:

In [ ]:

Python Samples.txt - Notepad

```

import openai
import os

from dotenv import load_dotenv, find_dotenv
_ = load_dotenv(find_dotenv()) # read local .env file

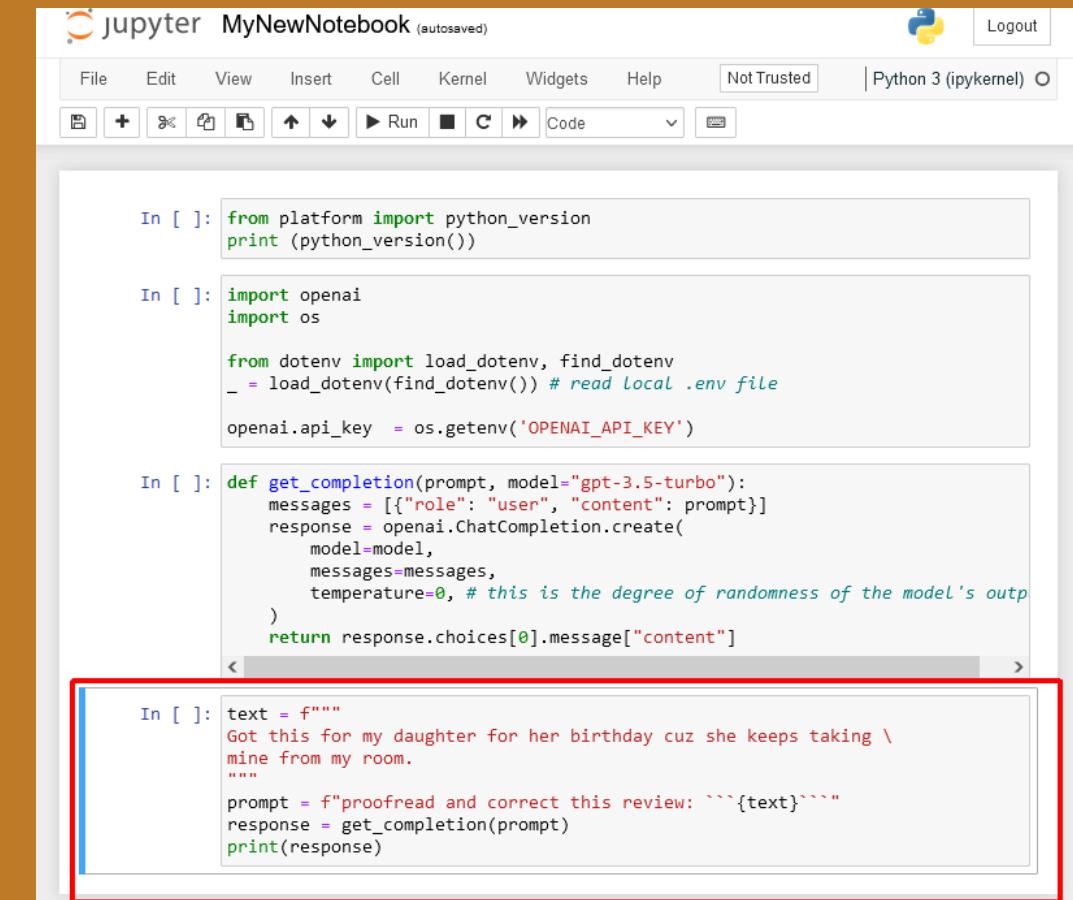
openai.api_key = os.getenv('OPENAI_API_KEY')

Cell 3

def get_completion(prompt, model="gpt-3.5-turbo"):
 messages = [{"role": "user", "content": prompt}]
 response = openai.ChatCompletion.create(
 model=model,
 messages=messages,
 temperature=0, # this is the degree of randomness of the model's output
)
 return response.choices[0].message["content"]

Cell 4

text = f"""
Got this for my daughter for her birthday cuz she keeps taking \nmine from my room.
"""
prompt = f"proofread and correct this review: ``{text}``"
response = get_completion(prompt)
print(response)
```



In [ ]:

```

from platform import python_version
print(python_version())

import openai
import os

from dotenv import load_dotenv, find_dotenv
_ = load_dotenv(find_dotenv()) # read local .env file

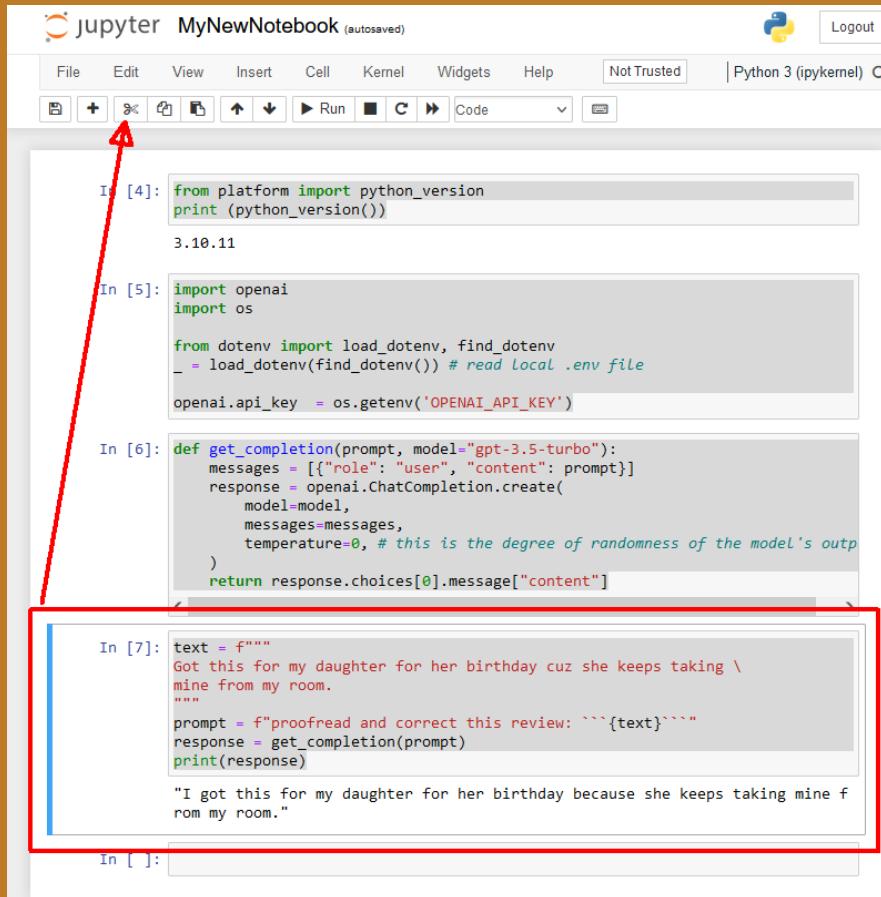
openai.api_key = os.getenv('OPENAI_API_KEY')

def get_completion(prompt, model="gpt-3.5-turbo"):
 messages = [{"role": "user", "content": prompt}]
 response = openai.ChatCompletion.create(
 model=model,
 messages=messages,
 temperature=0, # this is the degree of randomness of the model's output
)
 return response.choices[0].message["content"]

text = f"""
Got this for my daughter for her birthday cuz she keeps taking \nmine from my room.
"""
prompt = f"proofread and correct this review: ``{text}``"
response = get_completion(prompt)
print(response)
```

# Deleting a Notebook Cell

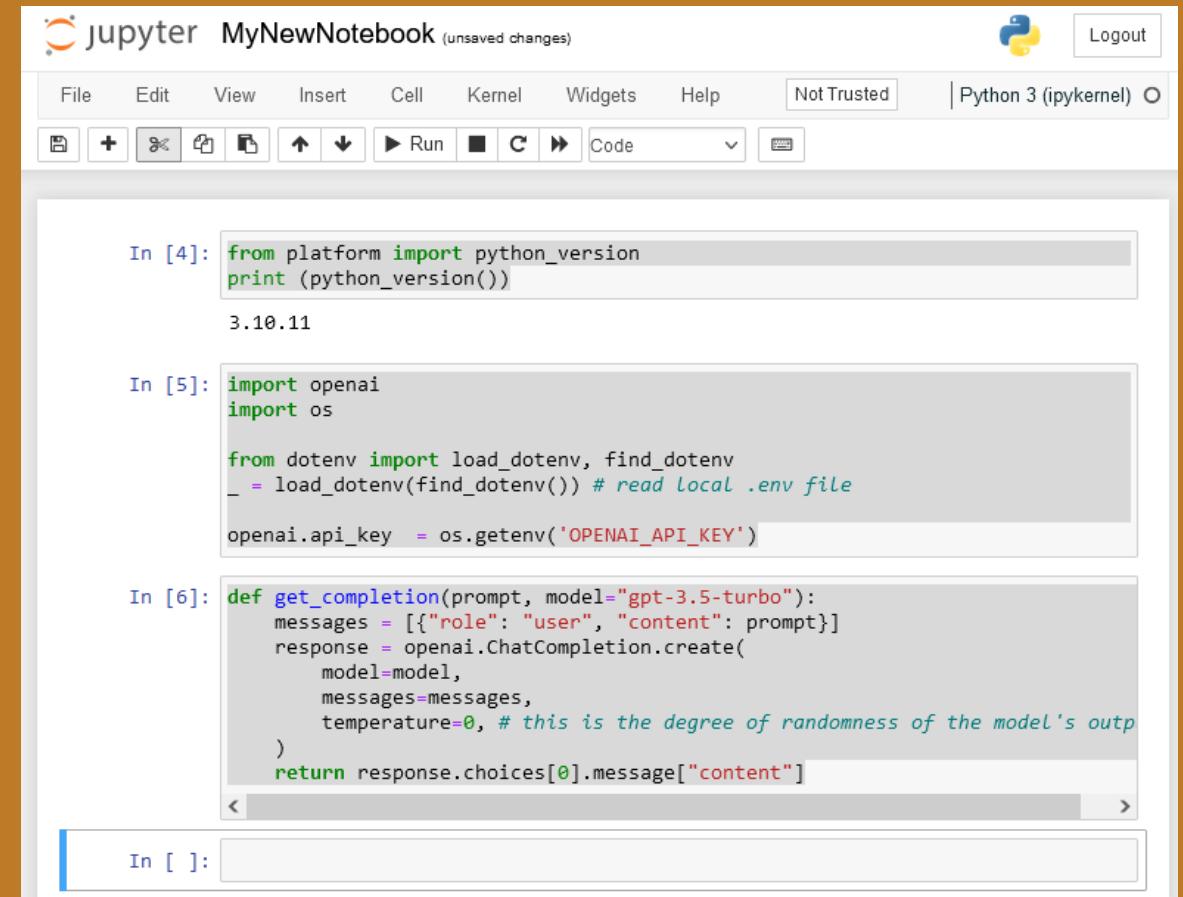
Select the cell to be deleted, then use the  icon



A screenshot of a Jupyter Notebook interface titled "MyNewNotebook". The notebook has several cells:

- In [4]: `from platform import python_version  
print(python_version())`  
3.10.11
- In [5]: `import openai  
import os  
  
from dotenv import load_dotenv, find_dotenv  
_ = load_dotenv(find_dotenv()) # read local .env file  
  
openai.api_key = os.getenv('OPENAI_API_KEY')`
- In [6]: `def get_completion(prompt, model="gpt-3.5-turbo"):  
 messages = [{"role": "user", "content": prompt}]  
 response = openai.ChatCompletion.create(  
 model=model,  
 messages=messages,  
 temperature=0, # this is the degree of randomness of the model's output  
 )  
 return response.choices[0].message["content"]`
- In [7]: `text = f"""\n Got this for my daughter for her birthday cuz she keeps taking mine from my room.\n """  
prompt = f"proofread and correct this review: ```{text}```"  
response = get_completion(prompt)  
print(response)  
  
"I got this for my daughter for her birthday because she keeps taking mine from my room."`
- In [ ]:

The cell In [7] is highlighted with a red rectangle and a blue vertical bar on its left, indicating it is selected. A red arrow points from the top-left towards the delete icon in the toolbar.

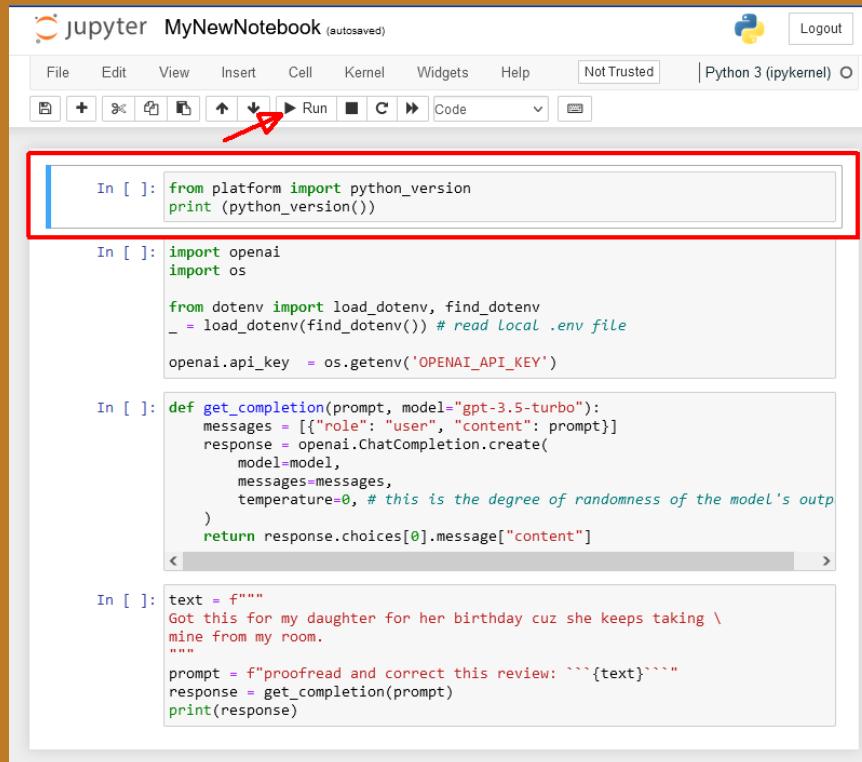


A screenshot of the same Jupyter Notebook interface after the cell In [7] has been deleted. The notebook now contains the following cells:

- In [4]: `from platform import python_version  
print(python_version())`  
3.10.11
- In [5]: `import openai  
import os  
  
from dotenv import load_dotenv, find_dotenv  
_ = load_dotenv(find_dotenv()) # read local .env file  
  
openai.api_key = os.getenv('OPENAI_API_KEY')`
- In [6]: `def get_completion(prompt, model="gpt-3.5-turbo"):  
 messages = [{"role": "user", "content": prompt}]  
 response = openai.ChatCompletion.create(  
 model=model,  
 messages=messages,  
 temperature=0, # this is the degree of randomness of the model's output  
 )  
 return response.choices[0].message["content"]`
- In [ ]:

# Running Code in a Notebook

1. Select the 1<sup>st</sup> cell
2. Use the  Run icon to run/step thru 1 cell at a time
3. Cells that have output will show it under the cell

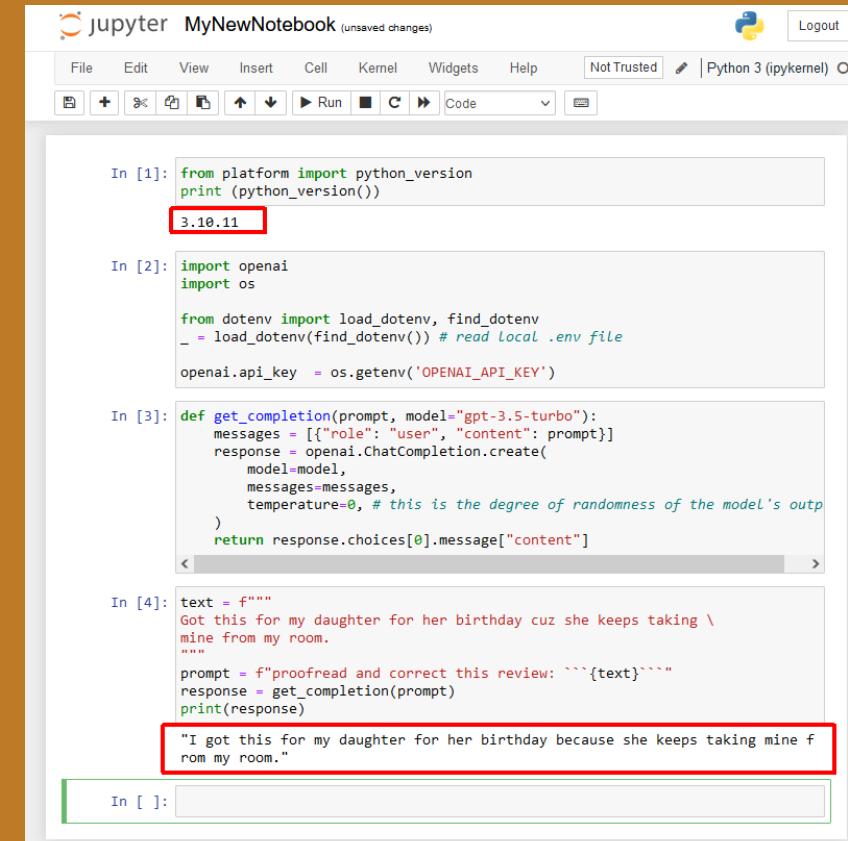


In [1]: `from platform import python_version  
print(python_version())`

In [2]: `import openai  
import os  
  
from dotenv import load_dotenv, find_dotenv  
_ = load_dotenv(find_dotenv()) # read local .env file  
  
openai.api_key = os.getenv('OPENAI_API_KEY')`

In [3]: `def get_completion(prompt, model="gpt-3.5-turbo"):  
 messages = [{"role": "user", "content": prompt}]  
 response = openai.ChatCompletion.create(  
 model=model,  
 messages=messages,  
 temperature=0, # this is the degree of randomness of the model's output  
 )  
 return response.choices[0].message["content"]`

In [4]: `text = f"""\nGot this for my daughter for her birthday cuz she keeps taking \\  
mine from my room.\n"""\n\nprompt = f"proofread and correct this review: ```{text}```"  
response = get_completion(prompt)  
print(response)`



In [1]: `from platform import python_version  
print(python_version())`  
3.10.11

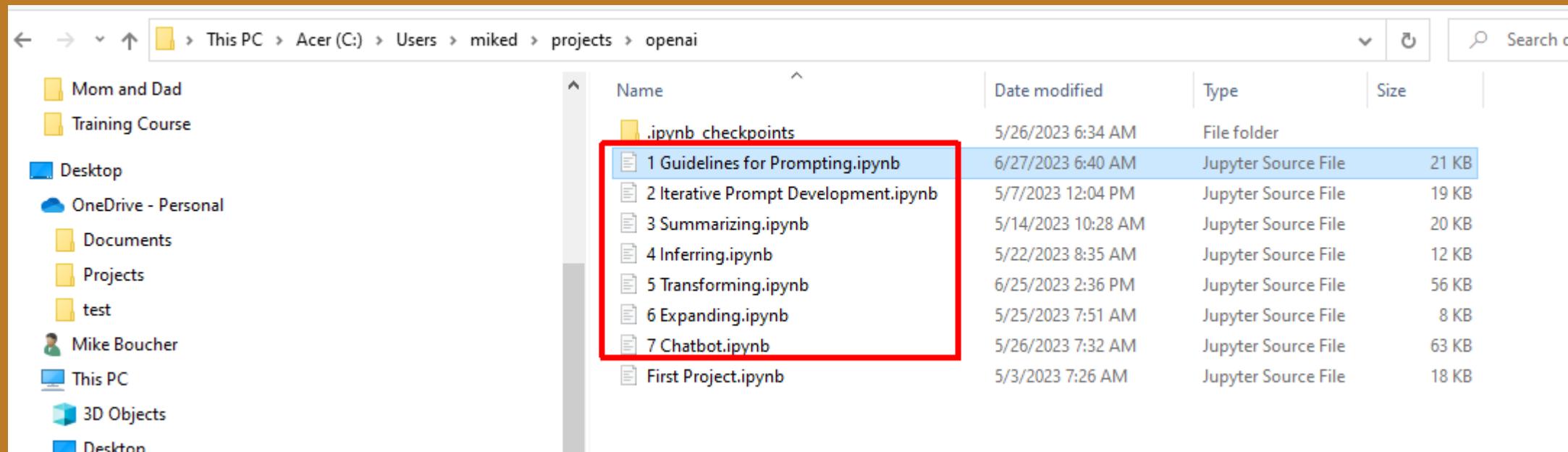
In [2]: `import openai  
import os  
  
from dotenv import load_dotenv, find_dotenv  
_ = load_dotenv(find_dotenv()) # read local .env file  
  
openai.api_key = os.getenv('OPENAI_API_KEY')`

In [3]: `def get_completion(prompt, model="gpt-3.5-turbo"):  
 messages = [{"role": "user", "content": prompt}]  
 response = openai.ChatCompletion.create(  
 model=model,  
 messages=messages,  
 temperature=0, # this is the degree of randomness of the model's output  
 )  
 return response.choices[0].message["content"]`

In [4]: `text = f"""\nGot this for my daughter for her birthday cuz she keeps taking \\  
mine from my room.\n"""\n\nprompt = f"proofread and correct this review: ```{text}```"  
response = get_completion(prompt)  
print(response)`  
"I got this for my daughter for her birthday because she keeps taking mine f  
rom my room."

# Suggestion - Create Notebooks for the Class

Using the knowledge gained in this section, you may want to create a Notebook for each chapter of the class.



# Occasional Timeout Error

- When you're running an example from the course that makes multiple calls to the openai API, sometimes you'll get a timeout error.
- This is because the free account allows 3 API calls per minute. If you try more than that, you'll get an error.
- When this occurs, just move on (or sign up for a paying account) or introduce a time delay into the code.
- If you're running the Jupyter Notebook from within the course you will not get this error.

In [26]:

```
for issue in user_messages:
 prompt = f"Tell me what language this is: '{issue}'"
 lang = get_completion(prompt)
 print(f"Original message ({lang}): {issue}")

 prompt = f"""
 Translate the following text to English \
 and Korean: '{issue}'"""
 response = get_completion(prompt)
 print(response, "\n")
 time.sleep(20)

File ~/anaconda3/lib/site-packages/openai/api_requestor.py:763, in APIRequestor._interpret_response_line(self, rbody, rcode, rheaders, stream)
 761 stream_error = stream and "error" in resp.data
 762 if stream_error or not 200 <= rcode < 300:
--> 763 raise self.handle_error_response(
 764 rbody, rcode, resp.data, rheaders, stream_error=stream_error
 765)
 766 return resp
```

RateLimitError: Rate limit reached for default-gpt-3.5-turbo in organization org-frdB6Ua4LlRzGLIULT6X1QKc on requests per min. Limit: 3 / min. Please try again in 20s. Contact us through our help center at help.openai.com if you continue to have issues. Please add a payment method to your account to increase your rate limit. Visit <https://platform.openai.com/account/billing> to add a payment method.

(above) Hitting the Rate Limit Error

Everything you need to know to take the free course,  
ChatGPT Prompt Engineering for developers

In [28]:

```
import time
for issue in user_messages:
 prompt = f"Tell me what language this is: '{issue}'"
 lang = get_completion(prompt)
 print(f"Original message ({lang}): {issue}")

 time.sleep(20)

 prompt = f"""
 Translate the following text to English \
 and Korean: '{issue}'"""
 response = get_completion(prompt)
 print(response, "\n")
 time.sleep(20)
```

Original message (The language is French.): La performance du système est plus lente que d'habitude.  
The performance of the system is slower than usual.

시스템의 성능이 평소보다 느립니다.

Original message (The language is Spanish.): Mi monitor tiene pixeles que no se iluminan.  
English: "My monitor has pixels that do not light up."

Korean: "내 모니터에는 밝아지지 않는 픽셀이 있습니다."

Original message (The language is Italian.): Il mio mouse non funziona  
English: "My mouse is not working."  
Korean: "내 마우스가 작동하지 않습니다."

Original message (The language is Polish.): Mój klawisz Ctrl jest zepsuty  
English: "My Ctrl key is broken"  
Korean: "내 Ctrl 키가 고장 났어요"

Original message (The language is Chinese.): 我的屏幕在闪烁  
English: My screen is flickering.  
Korean: 내 화면이 깜빡거립니다.

(above) Using 20 second time delays to avoid the Rate Limit Error

# Congrats

Your PC is set up and you know everything you need to know to successfully follow along with every example from the course, ChatGPT Prompt Engineering for developers

Everything you need to know to take the free course,  
ChatGPT Prompt Engineering for developers





# Recap

Set up OpenAI and DeepLearning.AI accounts

Installed Anaconda/Jupyter Notebook and supporting Python libraries

Set up an OpenAI key on your PC

Learned how to use Jupyter Notebook

# Next Steps

1. Create a new Jupyter Notebook called, *1 Guidelines for Prompting*
2. Start the course
3. Use your newly created notebook, *1 Guidelines for Prompting*, to run the examples from the first chapter of the course

# references

Reference	Link
ChatGPT signup/login	<a href="https://chat.openai.com/auth/login">https://chat.openai.com/auth/login</a>
Course - ChatGPT Prompt Engineering for Developers	<a href="https://www.deeplearning.ai/short-courses/chatgpt-prompt-engineering-for-developers/">https://www.deeplearning.ai/short-courses/chatgpt-prompt-engineering-for-developers/</a>
Installing Anaconda on Windows	<a href="https://docs.anaconda.com/free/anaconda/install/windows/">https://docs.anaconda.com/free/anaconda/install/windows/</a>
Installing Latest Version of Python in Anaconda	<a href="https://conda.io/projects/conda/en/latest/user-guide/tasks/manage-python.html">https://conda.io/projects/conda/en/latest/user-guide/tasks/manage-python.html</a>
How to Generate an OpenAI key and install it on your PC	<a href="https://www.howtogeek.com/885918/how-to-get-an-openai-api-key/">https://www.howtogeek.com/885918/how-to-get-an-openai-api-key/</a>

# The End

Contact: mikeboucher@yahoo.com

Everything you need to know to take the free course,  
ChatGPT Prompt Engineering for developers

# Appendix Slide - Installing Python 3.10 in Anaconda

If you're running an older version of Anaconda on your PC already, you may not have Python 3.10 installed. In that case, you'll need to install Python 3.10 in Anaconda. Alternatively, uninstall Anaconda, then reinstall the latest version (that's what I did).

The following slides explain how to do that.

## Steps

1. Install Python 3.10 in Anaconda
2. Run Python 3.10 in Anaconda
3. From a Jupyter Notebook, write and run your 1<sup>st</sup> program to check the python version

# Python 3.10

- Anaconda installs Python version 3.7 by default
- Anaconda allows you to run different versions of Python
- One of the examples in the course uses a Python library that requires Python version 3.10 or higher
- So let's install Python version 3.10 in Anaconda

We'll follow the instructions from here: <https://conda.io/projects/conda/en/latest/user-guide/tasks/manage-python.html>

- Open an anaconda prompt
- Check your Python version with: `python --version`
- Run the command: `conda create -n py310 python=3.10 anaconda`
- Then when that's done, activate py310 with the following command: `conda activate py310`
- Verify you're running Python 3.10 with: `python --version`

## Installing a different version of Python

To install a different version of Python without overwriting the current version, create a new environment and install the second Python version into it:

1. Create the new environment:

- To create the new environment for Python 3.9, in your terminal window or an Anaconda Prompt, run:

```
conda create -n py39 python=3.9 anaconda
```

### • Note

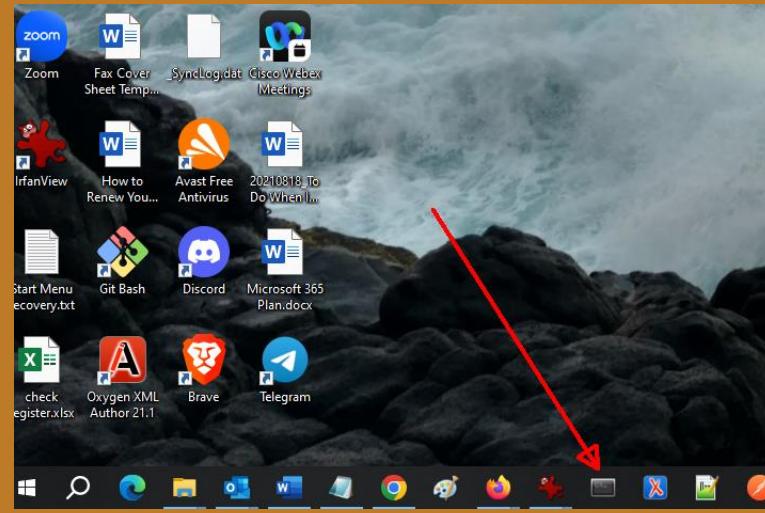
Replace `py39` with the name of the environment you want to create. `anaconda` is the metapackage that includes all of the Python packages comprising the Anaconda distribution. `python=3.9` is the package and version you want to install in this new environment. This could be any package, such as `numpy=1.19`, or multiple packages.

2. Activate the new environment.

3. Verify that the new environment is your current environment.
4. To verify that the current environment uses the new Python version, in your terminal window or an Anaconda Prompt, run:

```
python --version
```

# Installing Python 3.10



```
Anaconda Prompt (Anaconda3) - jupyter notebook

(base) C:\Users\miked>python --version
Python 3.7.4
```

1. Open an anaconda prompt
2. Check your Python version with the command, `python --version`
3. Run the command, `conda create -n py310 python=3.10 anaconda`

Everything you need to know to take the free course,  
ChatGPT Prompt Engineering for developers

```
Select Administrator: Anaconda Prompt (Anaconda3) - conda create -n py310 python=3.10 anaconda

(base) C:\WINDOWS\system32>conda create -n py310 python=3.10 anaconda
Collecting package metadata (current_repotdata.json): done
Solving environment: done

==> WARNING: A newer version of conda exists. <=
 current version: 4.8.1
 latest version: 23.3.1

Please update conda by running

$ conda update -n base -c defaults conda

Package Plan

environment location: C:\Users\miked\Anaconda3\envs\py310

added / updated specs:
- anaconda
- python=3.10

The following packages will be downloaded:

 package build

 _anaconda_depends-2023.03 py310_0 67 KB
 alabaster-0.7.12 pyhd3eb1b0_0 16 KB
 anaconda-custom py310_1 12 KB
 anyio-3.5.0 py310haa95532_0 169 KB
 appdirs-1.4.4 pyhd3eb1b0_0 12 KB
 argon2-cffi-21.3.0 pyhd3eb1b0_0 15 KB
 argon2-cffi-bindings-21.2.0 py310h2bfff1b_0 36 KB
 arrow-1.2.3 py310haa95532_1 159 KB
 astroid-2.14.2 py310haa95532_0 400 KB
 astropy-5.1 py310h9128911_0 6.4 MB
 asttokens-2.0.5 py310h9128911_0 20 KB
 atomicwrites-1.4.0 py310h9128911_0 11 KB
 attrs-22.1.0 py310haa95532_0 85 KB
 automat-20.2.0 py310h9128911_0 31 KB
 autopep8-1.6.0 py310haa95532_0 43 KB
 babel-2.11.0 py310h9128911_0 6.8 MB
 backcall-0.2.0 py310h9128911_0 13 KB
 bcrypt-3.2.0 py310h9128911_0 37 KB
 beautifulsoup4-4.12.2 py310haa95532_0 212 KB
 binaryornot-0.4.4 py310h9128911_0 351 KB
 black-23.3.0 py310haa95532_0 273 KB
 bleach-4.1.0 py310h9128911_0 123 KB
 blosc-1.21.3 h6c2663c_0 86 KB
 bokeh-2.4.3 py310haa95532_0 7.6 MB
 bottleneck-1.3.5 py310h9128911_0 106 KB
 brotli-1.0.9 h2bbff1b_7 18 KB
 brotli-bin-1.0.9 h2bbff1b_7 19 KB
 brotlipy-0.7.0 py310h9128911_0 335 KB
 ca-certificates-2023.01.10 haa95532_0 121 KB
 certifi-2023.5.7 py310haa95532_0 153 KB
 cffi-1.15.1 py310h9128911_0 239 KB
 cfitsio-3.470 h2bbff1b_7 518 KB
 chardet-4.0.0 py310haa95532_0 220 KB
```

# Installing Python 3.10

```
■ Select Administrator: Anaconda Prompt (Anaconda3) - conda create -n py310 python=3.10 anaconda
tabulate pkgs/main/win-64::tabulate-0.8.10-py310haa95532_0
tbb pkgs/main/win-64::tbb-2021.8.0-h59b6b97_0
tbb4py pkgs/main/win-64::tbb4py-2021.8.0-py310hs9b6b97_0
tblib pkgs/main/noarch::tblib-1.7.0-pyhd3eb1b0_0
tenacity pkgs/main/win-64::tenacity-8.2.2-py310haa95532_0
terminado pkgs/main/win-64::terminado-0.17.1-py310haa95532_0
text-unicodecode pkgs/main/noarch::text-unicodecode-1.3-pyhd3eb1b0_0
textdistance pkgs/main/noarch::textdistance-4.2.1-pyhd3eb1b0_0
threadpoolctl pkgs/main/noarch::threadpoolctl-2.2.0-pyhd6d9192_0
three-merge pkgs/main/noarch::three-merge-0.1.1-pyhd3eb1b0_0
tifffile pkgs/main/noarch::tifffile-2021.7.2-pyhd3eb1b0_2
tinyccss2 pkgs/main/win-64::tinyccss2-1.2.1-py310haa95532_0
tk pkgs/main/win-64::tk-8.6.12-h2bbff1b_0
tldextract pkgs/main/noarch::tldextract-3.2.0-pyhd3eb1b0_0
tokenizers pkgs/main/win-64::tokenizers-0.11.4-py310he518icf_1
toml pkgs/main/noarch::toml-0.10.2-pyhd3eb1b0_0
tomli pkgs/main/win-64::tomli-2.0.1-py310haa95532_0
tomlkit pkgs/main/win-64::tomlkit-0.11.1-py310haa95532_0
toolz pkgs/main/win-64::toolz-0.12.0-py310haa95532_0
tornado pkgs/main/win-64::tornado-6.2-py310hb2bbff1b_0
tqdm pkgs/main/win-64::tqdm-4.65.0-py310h999e9c_0
traitlets pkgs/main/win-64::traitlets-5.7.1-py310haa95532_0
transformers pkgs/main/win-64::transformers-4.24.0-py310haa95532_0
twisted pkgs/main/win-64::twisted-22.10.0-py310hb2bbff1b_0
twisted-icpsupport pkgs/main/win-64::twisted-icpsupport-1.0.2-py310hb2bbff1b_0
typing-extensions pkgs/main/win-64::typing-extensions-4.5.0-py310haa95532_0
typing_extensions pkgs/main/win-64::typing_extensions-4.5.0-py310haa95532_0
tzdata pkgs/main/noarch::tzdata-2023c-h04d1681_0
ujson pkgs/main/win-64::ujson-5.4.0-py310hd7b12b_0
unidecode pkgs/main/noarch::unidecode-1.2.0-pyhd3eb1b0_0
urllib3 pkgs/main/win-64::urllib3-1.26.15-py310haa95532_0
vc pkgs/main/win-64::vc-14.2-h21ff451_1
vs2015_runtime pkgs/main/win-64::vs2015_runtime-14.27.29016-h5e58377_2
w3lib pkgs/main/noarch::w3lib-1.21.0-pyhd3eb1b0_0
watchdog pkgs/main/win-64::watchdog-2.1.6-py310haa95532_0
wcwidth pkgs/main/noarch::wcwidth-0.2.5-pyhd3eb1b0_0
webencodings pkgs/main/win-64::webencodings-0.5.1-py310haa95532_1
websocket-client pkgs/main/win-64::websocket-client-0.58.0-py310haa95532_4
werkzeug pkgs/main/win-64::werkzeug-2.2.3-py310haa95532_0
whatthepatch pkgs/main/win-64::whatthepatch-1.0.2-py310haa95532_0
wheel pkgs/main/win-64::wheel-0.38.4-py310haa95532_0
widgetsnbextension pkgs/main/win-64::widgetsnbextension-4.0.5-py310haa95532_0
win_inet_pton pkgs/main/win-64::win_inet_pton-1.1.5-py310haa95532_0
winpty pkgs/main/win-64::winpty-0.4.3-4
wrapt pkgs/main/win-64::wrapt-1.14.1-py310hb2bbff1b_0
xarray pkgs/main/win-64::xarray-2022.11.0-py310haa95532_0
xlwings pkgs/main/win-64::xlwings-0.29.1-py310haa95532_0
xz pkgs/main/win-64::xz-5.4.-h8cc25b3_0
yaml pkgs/main/win-64::yaml-2.2.5-he774522_0
yapf pkgs/main/noarch::yapf-0.31.0-pyhd3eb1b0_0
zeromq pkgs/main/win-64::zeromq-4.3.4-hd77b12b_0
zfp pkgs/main/win-64::zfp-0.5.5-hd77b12b_6
zict pkgs/main/win-64::zict-2.2.0-py310haa95532_0
zipp pkgs/main/win-64::zipp-3.11.0-py310haa95532_0
zlib pkgs/main/win-64::zlib-1.2.13-h8cc25b3_0
zope pkgs/main/win-64::zope-1.0-py310haa95532_1
zope.interface pkgs/main/win-64::zope.interface-5.4.0-py310hb2bbff1b_0
zstd pkgs/main/win-64::zstd-1.5.5-hd43e919_0
```

Enter 'y' to proceed with the install

Everything you need to know to take the free course,  
ChatGPT Prompt Engineering for developers

```
■ Select Administrator: Anaconda Prompt (Anaconda3) - conda create -n py310 python=3.10 anaconda
menuinst-1.4.19 94 KB #####
zict-2.2.0 50 KB #####
pynetworkx-1.5.0 1.2 MB #####
arrow-1.2.3 159 KB #####
soup sleeve-2.4 70 KB #####
xlwings-0.29.1 1.2 MB #####
pysocks-1.7.1 28 KB #####
three-merge-0.1.1 10 KB #####
pyrsistent-0.18.0 87 KB #####
pyqtwebengine-5.15.7 123 KB #####
bottleneck-1.3.5 106 KB #####
mkl-service-2.4.0 48 KB #####
qt-main-5.15.2 59.4 MB #####
pyzmq-25.0.2 406 KB #####
itsdangerous-2.0.1 18 KB #####
libprotobuf-1.0.9 30 KB #####
libxslt-1.1.37 448 KB #####
glib-2.69.1 1.8 MB #####
Sniffio-1.2.0 15 KB #####
zfp-0.5.5 139 KB #####
importlib_metadata-6 8 KB #####
urllib3-1.26.15 195 KB #####
gstreamer-1.18.5 1.7 MB #####
curl-7.88.1 147 KB #####
binaryornot-0.4.4 351 KB #####
mkl-2021.4.0 114.9 MB #####
idna-3.4 97 KB #####
python-snappy-0.6.1 34 KB #####
libxml2-2.10.3 2.9 MB #####
tomli-2.0.1 25 KB #####
libffi-3.4.4 113 KB #####
psutil-5.9.0 353 KB #####
watchdog-2.1.6 113 KB #####
jmespath-0.10.0 22 KB #####
pylint-2.16.2 767 KB #####
qtconsole-5.4.2 207 KB #####
pyyaml-6.0 153 KB #####
toml-0.10.2 20 KB #####
et_xmlfile-1.1.0 16 KB #####
incremental-21.3.0 17 KB #####
regex-2022.7.9 308 KB #####
pandas-1.5.3 10.6 MB #####
networkx-2.8.4 2.6 MB #####
json5-0.9.6 21 KB #####
openjpeg-2.4.0 219 KB #####
prometheus_client-0. 90 KB #####
jupyter_client-8.1.0 197 KB #####
setuptools-66.0.0 1.2 MB #####
pyerna-2.0.0 360 KB #####
service_identity-18. 13 KB #####
plotly-5.9.0 4.1 MB #####
rope-1.7.0 50 KB #####
flask-2.2.2 443 KB #####
intervaltree-3.1.0 25 KB #####
giflib-5.2.1 88 KB #####
pandocfilters-1.5.0 11 KB #####
Preparing transaction: done | 100%
Verifying transaction: done | 100%
Executing transaction: | - | 100%
```

If it gets stuck at Executing transaction, do a CTRL-C

# Installing Python 3.10 - Last Step

```
Administrator: Anaconda Prompt (Anaconda3)
y qtconsole-5.4.2 207 KB #####
pyyaml-6.0 153 KB #####
toml-0.10.2 20 KB #####
et_xmlfile-1.1.0 10 KB #####
incremental-21.3.0 17 KB #####
regex-2022.7.9 308 KB #####
pandas-1.5.3 10.6 MB #####
networkx-2.8.4 2.6 MB #####
jsons-0.9.6 21 KB #####
openjpeg-2.4.0 219 KB #####
prometheus_client-0. 90 KB #####
jupyter_client-8.1.0 197 KB #####
setuptools-66.0.0 1.2 MB #####
ipyerfa-2.0.0 360 KB #####
service_identity-18. 13 KB #####
plotly-5.9.0 4.1 MB #####
rtree-1.0.1 50 KB #####
rope-1.7.0 443 KB #####
flask-2.2.2 167 KB #####
intervaltree-3.1.0 25 KB #####
giflib-5.2.1 88 KB #####
pandocfilters-1.5.0 11 KB #####
Preparing transaction: done
Verifying transaction: done
Executing transaction: b'\n\n Windows 64-bit packages of scikit-learn can be accelerated using scikit-learn-intelex.\n More details are available here: https://intel.github.io/scikit-learn-intelex\n For example:\n $ conda\n a install scikit-learn-intelex\n $ python -m sklearnex my_application.py\n\n'/ DEBUG menuinst_win32: _init_(199): Menu: name: 'Anaconda${PY_VER} ${PLATFORM}', prefix: 'C:\Users\miked\Anaconda3\envs\py310', env_name: 'py310', mode: 'user', used_mode: 'user' DEBUG menuinst_win32:create(323): Shortcut cmd is C:\Users\miked\Anaconda3\python.exe, args are ['C:\Users\miked\Anaconda3\cwp.py', 'C:\Users\miked\Anaconda3\envs\py310', 'C:\Users\miked\Anaconda3\envs\py310\python.exe', 'C:\Users\miked\Anaconda3\envs\py310\Scripts\jupyter-notebook-script.py', '"%USERPROFILE%"'] DEBUG menuinst_win32:_init_(199): Menu: name: 'Anaconda${PY_VER} ${PLATFORM}', prefix: 'C:\Users\miked\Anaconda3\envs\py310', env_name: 'py310', mode: 'user', used_mode: 'user' DEBUG menuinst_win32:create(323): Shortcut cmd is C:\Users\miked\Anaconda3\pythonw.exe, args are ['C:\Users\miked\Anaconda3\cwp.py', 'C:\Users\miked\Anaconda3\envs\py310', 'C:\Users\miked\Anaconda3\envs\py310\pythonw.exe', 'C:\Users\miked\Anaconda3\envs\py310\Scripts\spyder-script.py'] DEBUG menuinst_win32:create(323): Shortcut cmd is C:\Users\miked\Anaconda3\python.exe, args are ['C:\Users\miked\Anaconda3\cwp.py', 'C:\Users\miked\Anaconda3\envs\py310', 'C:\Users\miked\Anaconda3\envs\py310\python.exe', 'C:\Users\miked\Anaconda3\envs\py310\Scripts\spyder-script.py', '--reset'] done
#
To activate this environment, use
#
$ conda activate py310
#
To deactivate an active environment, use
#
$ conda deactivate
(base) C:\WINDOWS\system32>conda activate py310
(py310) C:\WINDOWS\system32>python --version
Python 3.10.11
(py310) C:\WINDOWS\system32>
```

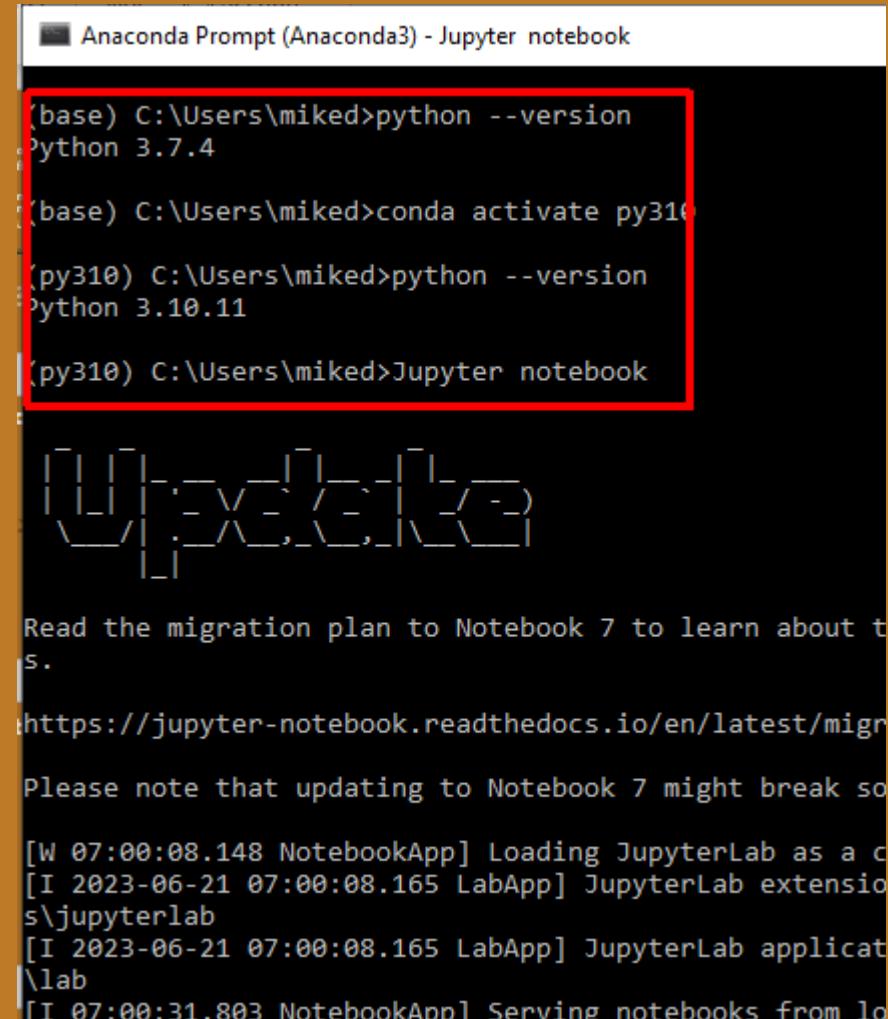
- After the install completes, activate py310 with the following command: *conda activate py310*
- Verify you're running Python 3.10 with: *python --version*

# Launching Jupyter Notebook on Python 3.10

From the Anaconda prompt:

1. `python --version` // Check the Python version that is running
2. `conda activate py310` // Activating Python 3.10
3. `python --version` // Verify that you're running Python 3.10
4. `Jupyter notebook` // open Jupyter Notebook

Wait for a minute Jupyter Notebook to open in your default browser



The screenshot shows a terminal window titled "Anaconda Prompt (Anaconda3) - Jupyter notebook". It displays the following command history:

```
(base) C:\Users\miked>python --version
Python 3.7.4

(base) C:\Users\miked>conda activate py310
(py310) C:\Users\miked>python --version
Python 3.10.11

(py310) C:\Users\miked>Jupyter notebook
```

The last three lines of the command history are highlighted with a red box.

Below the terminal, there is a decorative graphic of a spiral staircase made of brackets and parentheses.

Read the migration plan to Notebook 7 to learn about the changes.  
<https://jupyter-notebook.readthedocs.io/en/latest/migration.html>

Please note that updating to Notebook 7 might break some extensions.

```
[W 07:00:08.148 NotebookApp] Loading JupyterLab as a custom extension
[I 2023-06-21 07:00:08.165 LabApp] JupyterLab extension loaded at /jupyterlab
[I 2023-06-21 07:00:08.165 LabApp] JupyterLab application initialized
[I 07:00:31.803 NotebookApp] Serving notebooks from local directory
```

# Checking Python Version in Jupyter Notebook

The image consists of three screenshots of the Jupyter Notebook interface, each with a red arrow pointing to a specific action or result:

- Screenshot 1:** Shows the Jupyter file browser with the 'myprojects' folder highlighted by a red box.
- Screenshot 2:** Shows the Jupyter file browser with the 'myprojects/opencai' folder path highlighted by a red box. A red arrow points from this screenshot to the 'Run' button in the notebook toolbar.
- Screenshot 3:** Shows the Jupyter Notebook interface with a code cell containing the command `from platform import python_version  
print(python_version())`. This command is highlighted by a red box. The output of the cell shows the Python version as 3.10.11. The 'Run' button is also highlighted by a red arrow.

# Congrats

You're now running Python 3.10 and have written and run your first python program in Jupyter Notebook!

Feel free to add Python and Jupyter Notebook as skills on your LinkedIn profile

Everything you need to know to take the free course,  
ChatGPT Prompt Engineering for developers

