

Use ChatGPT to build a web-based system that can answer questions about a website

Special Topics: Generative AI-Driven Intelligent Apps
Development

Ademilton Marcelo da Cruz Nunes (19679)

Links

Github:

[https://github.com/ademiltonnunes/Machine-Learning/tree/main/ChatGPT%20/Use%20ChatGPT%20to%20create%20customer%20support%20website%20\(data%20source:%20web%20pages\)](https://github.com/ademiltonnunes/Machine-Learning/tree/main/ChatGPT%20/Use%20ChatGPT%20to%20create%20customer%20support%20website%20(data%20source:%20web%20pages))

Table of Content

- Introduction
- Tutorial Analysis
- Tutorial Analysis - Setting up a web crawler
- Tutorial Analysis - Building embedding indexes
- Tutorial Analysis - Building a question-answer system with embedding indexes
- Jupyter Notebook
- Python on Ubuntu
- Python Flask - Quick-Start
- Python Flask
- Conclusion

Introduction

This project aims to implement a customer support system using ChatGPT to build a web-based system that can answer questions about a website. We are using as a basis the tutorial available on the open website: [OpenAI document - Website Q&A with Embeddings](#).

In order to increase learning, I will be documenting the development process in a few steps, which will be the development of the system on Jupyter Notebook, Python on Ubuntu and Web Service based and Python Flask.

Tutorial Analysis

The tutorial [Website Q&A with Embeddings](#) to build an web-based system that can answer questions about your website is separated into three sections:

- Setting up a web crawler
- Building embedding indexes
- Building a question-answer system with embedding indexes

Tutorial Analysis - Setting up a web crawler

This process acquires data in text format through the process of crawling on a website.

This project is using the website sfbu.edu.

This crawling process takes texts from the homepage and will visit all links and pages on the website. From all captured data, it is indexed and placed in the form of embeddings.

Tutorial Analysis - Building embedding indexes

With the crawling result data, this is tokenized. The token process also standardizes the size of the inputs for the embedding process, breaking words into the same size.

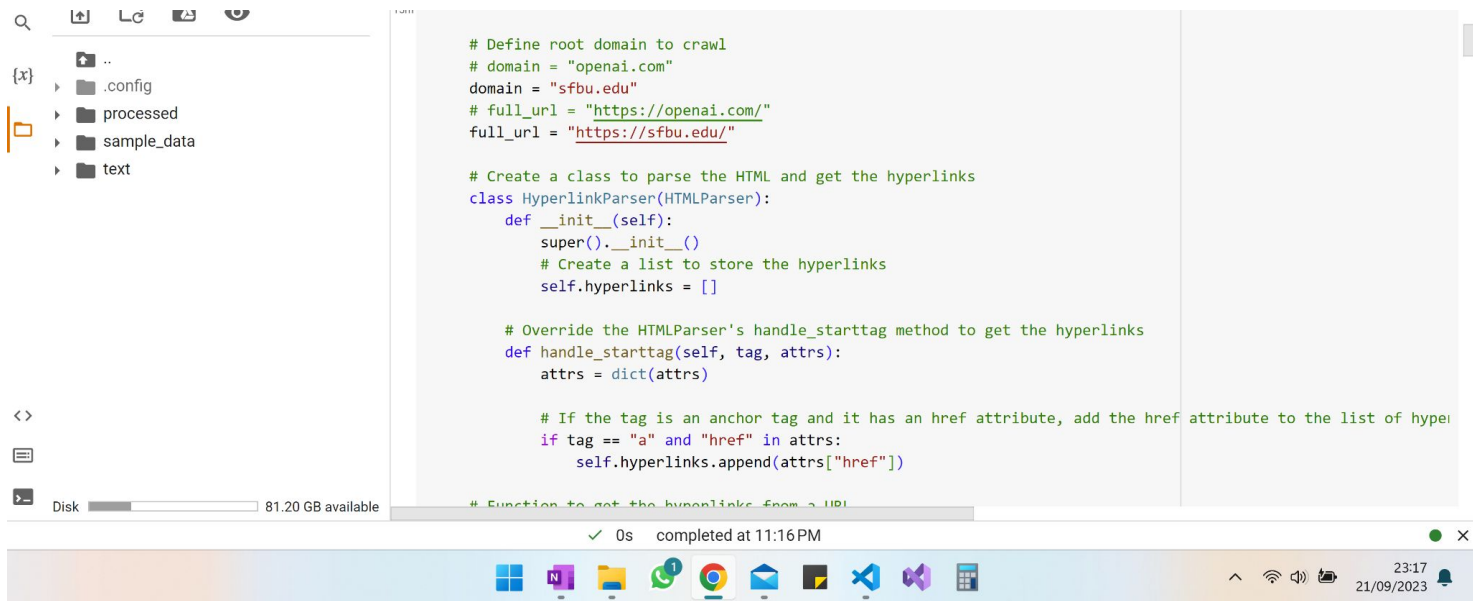
All data is embedding into indexes so that ChatGPT can use this data to answer questions

Tutorial Analysis - Building a question-answer system with embedding indexes

When embeddings are ready, the system is ready to receive simple questions and then give answers. The system answers users' by comparing existing embeds indexes and retrieving the answer. The gpt-3.5-turbo-instruct model is the model that will be used in this project. It generates a natural sounding response based on the retrieved text.

Jupyter Notebook

The image below shows the crawling process where I used the sfbu.edu website



```
# Define root domain to crawl
# domain = "openai.com"
domain = "sfbu.edu"
# full_url = "https://openai.com/"
full_url = "https://sfbu.edu/"

# Create a class to parse the HTML and get the hyperlinks
class HyperlinkParser(HTMLParser):
    def __init__(self):
        super().__init__()
        # Create a list to store the hyperlinks
        self.hyperlinks = []

    # Override the HTMLParser's handle_starttag method to get the hyperlinks
    def handle_starttag(self, tag, attrs):
        attrs = dict(attrs)

        # If the tag is an anchor tag and it has an href attribute, add the href attribute to the list of hyperlinks
        if tag == "a" and "href" in attrs:
            self.hyperlinks.append(attrs["href"])

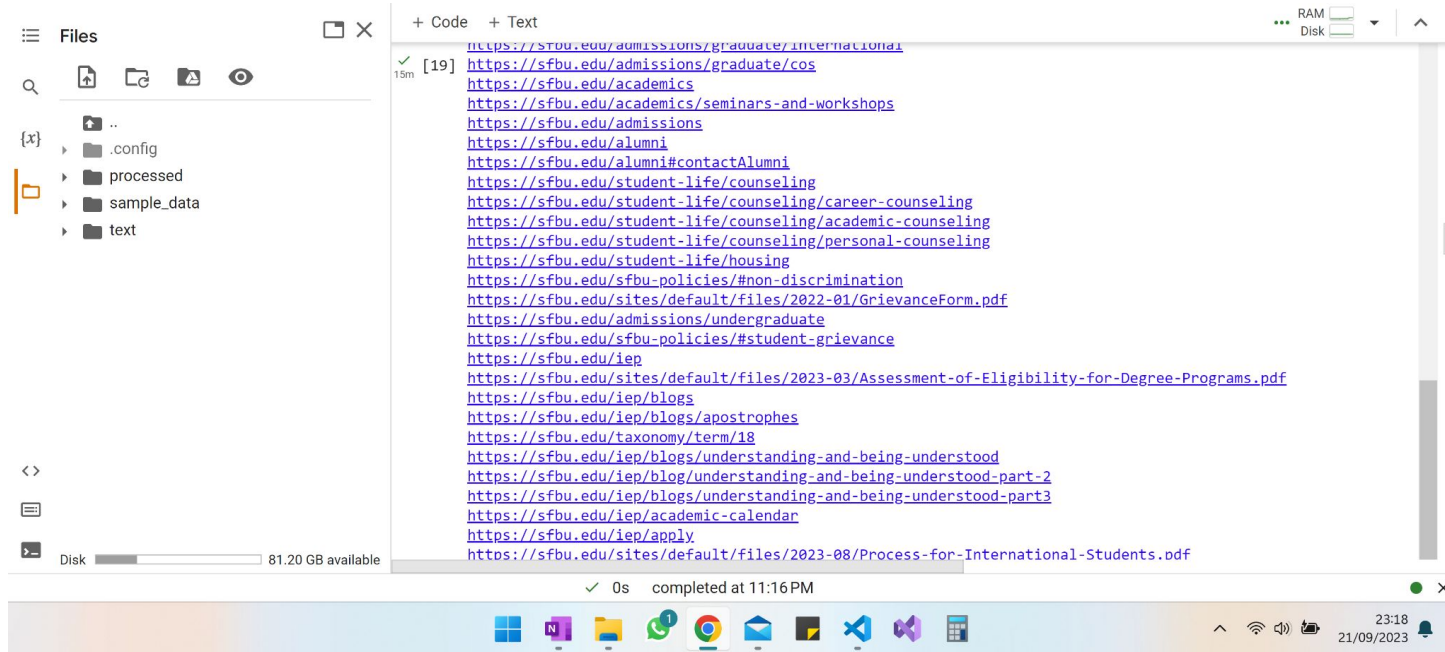
# Function to get the hyperlinks from a URL
```

0s completed at 11:16 PM

23:17 21/09/2023

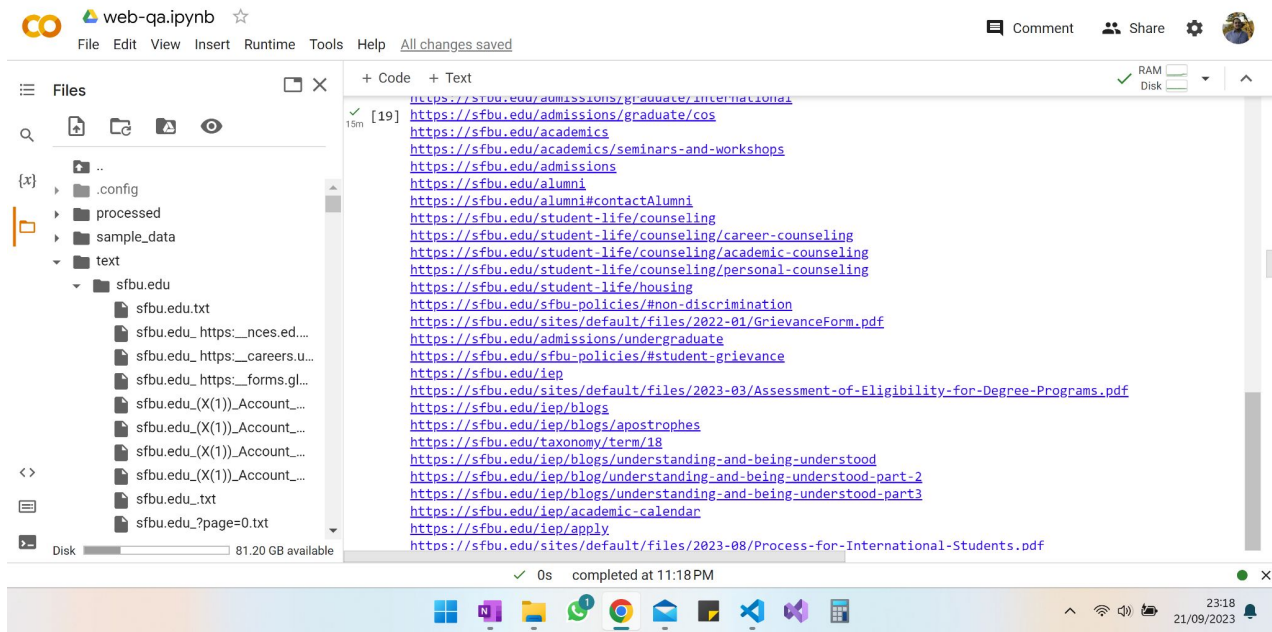
Jupyter Notebook

During the crawling process, other links and urls on the website are visited in addition to the homepage



Jupyter Notebook

After visiting links, the texts are saved in txt files



The screenshot displays a Jupyter Notebook environment. The top bar shows the notebook name 'web-qa.ipynb' and a status 'All changes saved'. The left sidebar contains a 'Files' panel with a tree view of the file system. The main area shows a code cell with 19 lines of text, each a URL. The bottom status bar indicates the notebook is 'completed at 11:18 PM'.

Files

- ..
- .config
- processed
- sample_data
- text
 - sfbu.edu
 - sfbu.edu.txt
 - sfbu.edu_https__nces.ed...
 - sfbu.edu_https__careers.u...
 - sfbu.edu_https__forms.gl...
 - sfbu.edu_(X(1))_Account...
 - sfbu.edu_(X(1))_Account...
 - sfbu.edu_(X(1))_Account...
 - sfbu.edu_(X(1))_Account...
 - sfbu.edu_(X(1))_Account...
 - sfbu.edu.txt
 - sfbu.edu_?page=0.txt

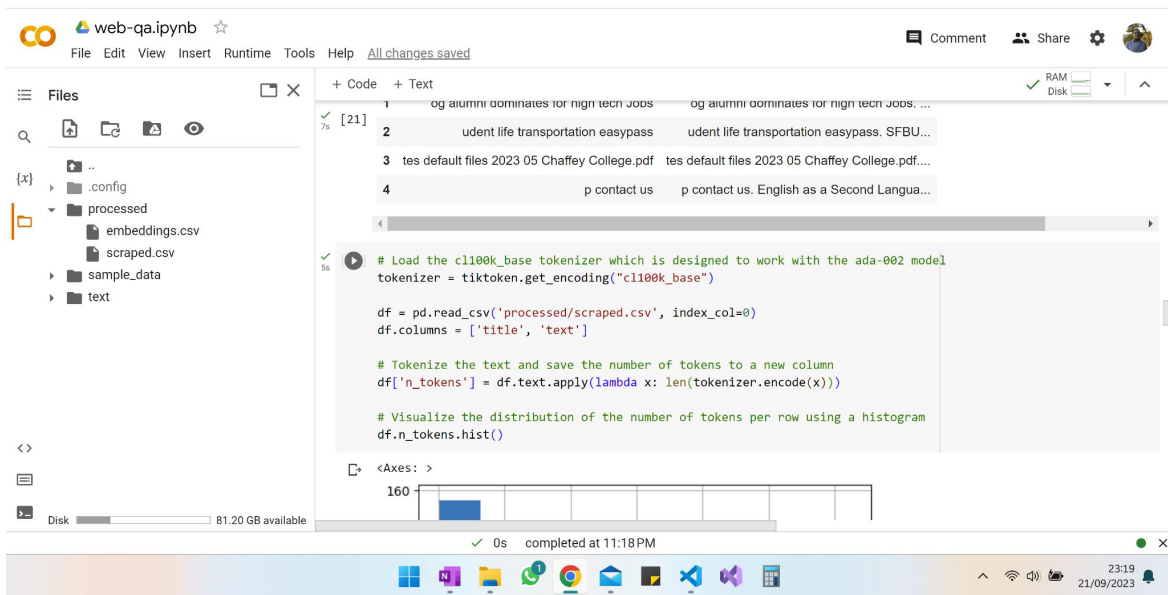
Code

```
[19] https://sfbu.edu/admissions/graduate/international  
https://sfbu.edu/admissions/graduate/cos  
https://sfbu.edu/academics  
https://sfbu.edu/academics/seminars-and-workshops  
https://sfbu.edu/admissions  
https://sfbu.edu/alumni  
https://sfbu.edu/alumni#contactAlumni  
https://sfbu.edu/student-life/counseling  
https://sfbu.edu/student-life/counseling/career-counseling  
https://sfbu.edu/student-life/counseling/academic-counseling  
https://sfbu.edu/student-life/counseling/personal-counseling  
https://sfbu.edu/student-life/housing  
https://sfbu.edu/sfbu-policies/#non-discrimination  
https://sfbu.edu/sites/default/files/2022-01/GrievanceForm.pdf  
https://sfbu.edu/admissions/undergraduate  
https://sfbu.edu/sfbu-policies/#student-grievance  
https://sfbu.edu/iep  
https://sfbu.edu/sites/default/files/2023-03/Assessment-of-Eligibility-for-Degree-Programs.pdf  
https://sfbu.edu/iep/blogs  
https://sfbu.edu/iep/blogs/apostrophes  
https://sfbu.edu/taxonomy/term/18  
https://sfbu.edu/iep/blogs/understanding-and-being-understood  
https://sfbu.edu/iep/blog/understanding-and-being-understood-part-2  
https://sfbu.edu/iep/blogs/understanding-and-being-understood-part3  
https://sfbu.edu/iep/academic-calendar  
https://sfbu.edu/iep/apply  
https://sfbu.edu/sites/default/files/2023-08/Process-for-International-Students.pdf
```

0s completed at 11:18 PM

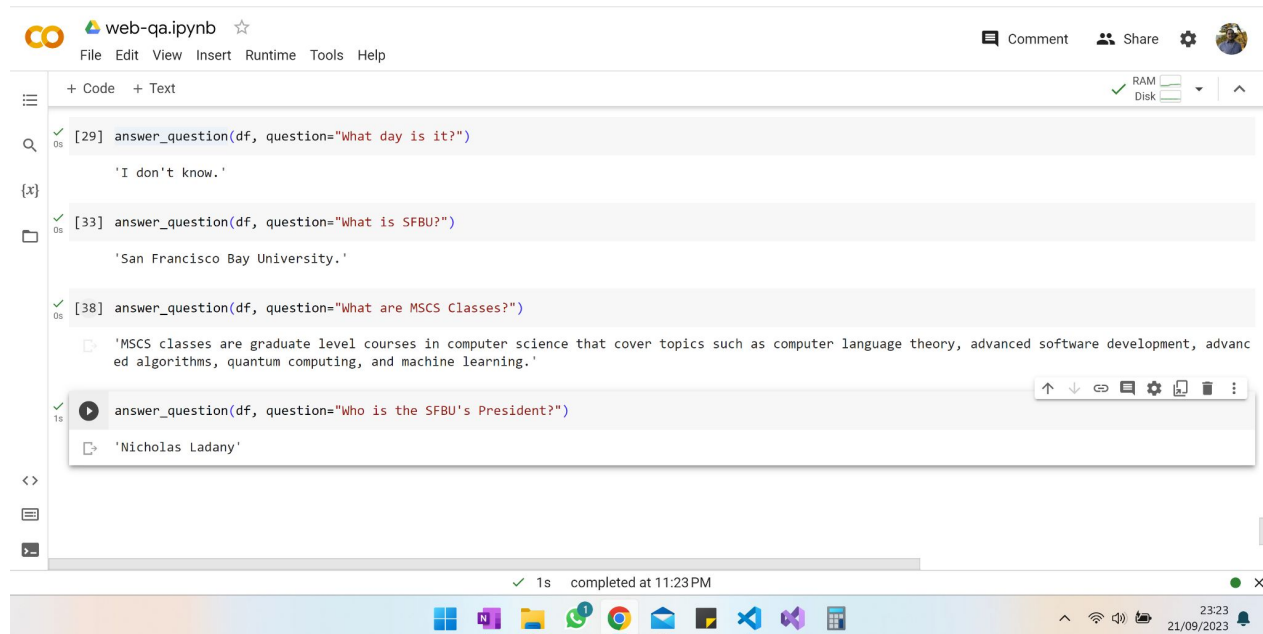
Jupyter Notebook

After this, the tokenization process is carried out on embedding indexes



Jupyter Notebook

After the crawling and embeddings process, the system is ready to answer questions related to the website used. If the system does not know the answer, it says: "I don't know"



```
web-qa.ipynb
File Edit View Insert Runtime Tools Help

+ Code + Text
RAM
Disk

[29] answer_question(df, question="What day is it?")
'I don't know.'

[33] answer_question(df, question="What is SFBU?")
'San Francisco Bay University.'

[38] answer_question(df, question="What are MSCS Classes?")
'MSCS classes are graduate level courses in computer science that cover topics such as computer language theory, advanced software development, advanced algorithms, quantum computing, and machine learning.'

answer_question(df, question="Who is the SFBU's President?")
'Nicholas Ladany'

1s completed at 11:23 PM
23:23
21/09/2023
```

Python on Ubuntu

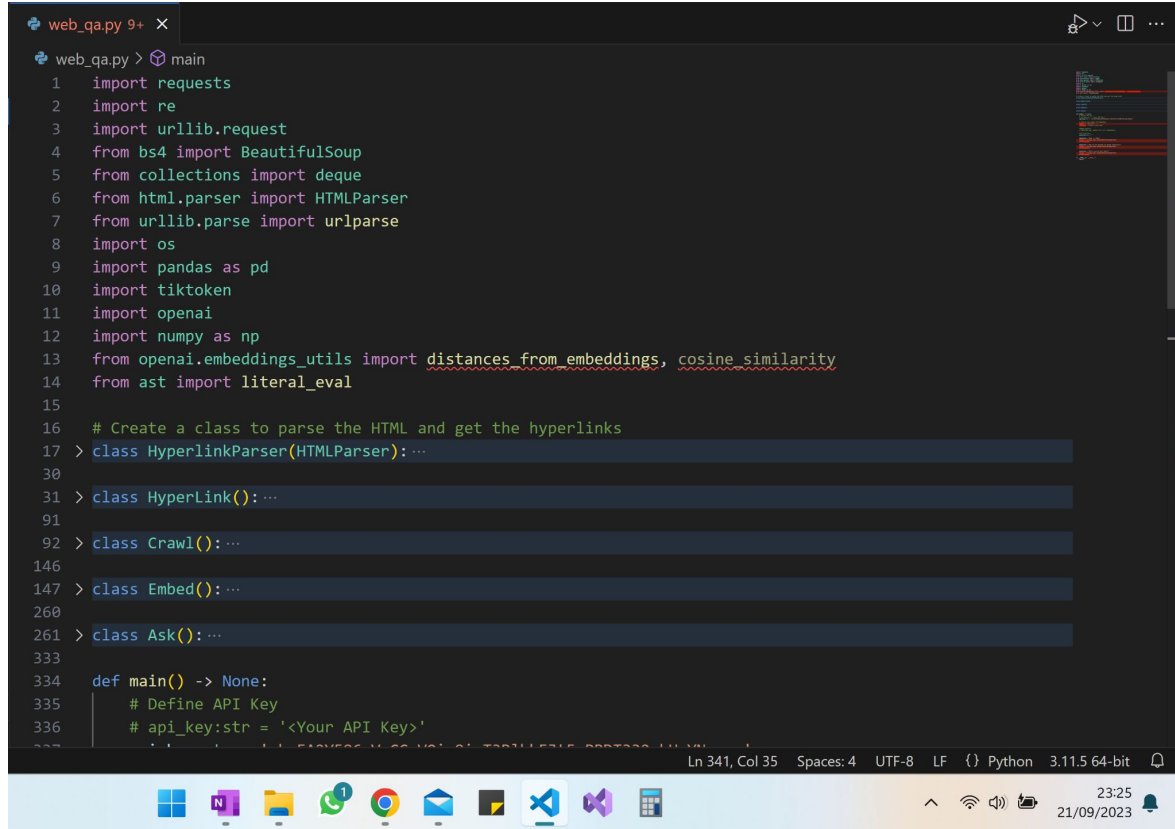
The Jupyter notebook content was exported in Python code. For this to work it was necessary to install some modules in Python. These modules are:

- `pip install beautifulsoup4`
- `pip install tiktoken`
- `pip install openai`
- `pip install pandas`
- `pip install matplotlib`
- `pip install plotly`
- `pip install scipy`
- `pip install scikit-learn`

Python on Ubuntu

With the Python code, this was refactored to a more object-oriented pattern. In this format, the code will be more adapted to future changes. Classes were created with their independent methods, as you can see in the next image:

Python on Ubuntu



```
web_qa.py x
web_qa.py > main
1  import requests
2  import re
3  import urllib.request
4  from bs4 import BeautifulSoup
5  from collections import deque
6  from html.parser import HTMLParser
7  from urllib.parse import urlparse
8  import os
9  import pandas as pd
10 import tiktoken
11 import openai
12 import numpy as np
13 from openai.embeddings_utils import distances_from_embeddings, cosine_similarity
14 from ast import literal_eval
15
16 # Create a class to parse the HTML and get the hyperlinks
17 > class HyperlinkParser(HTMLParser): ...
30
31 > class HyperLink(): ...
91
92 > class Crawl(): ...
146
147 > class Embed(): ...
260
261 > class Ask(): ...
333
334 def main() -> None:
335     # Define API Key
336     # api_key:str = '<Your API Key>'
337     # ...
```

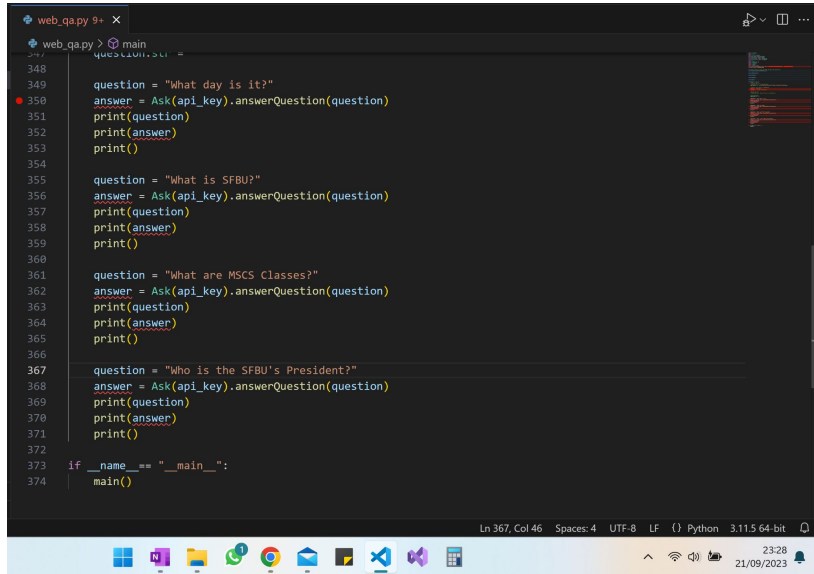
Ln 341, Col 35 Spaces: 4 UTF-8 LF {} Python 3.11.5 64-bit

23:25
21/09/2023

Python on Ubuntu

An initial test was carried out in the console, asking the same questions asked in the Jupyter notebook, so that answers and behavior could be compared.

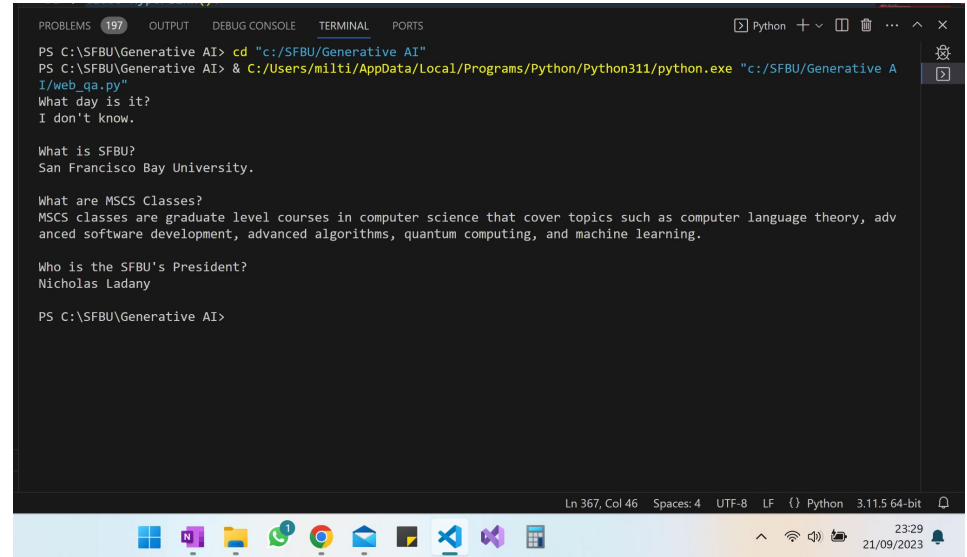
Python on Ubuntu



A screenshot of a code editor window titled 'web_qa.py 9+ X' showing a Python script. The script defines a function 'main' that interacts with an 'Ask' API to answer questions. The questions and their corresponding answers are: 'What day is it?' (I don't know.), 'What is SFBU?' (San Francisco Bay University.), 'What are MSCS Classes?' (MSCS classes are graduate level courses in computer science that cover topics such as computer language theory, advanced software development, advanced algorithms, quantum computing, and machine learning.), and 'Who is the SFBU's President?' (Nicholas Ladany). The script includes a standard Python entry point check at the bottom.

```
348
349
350 question = "What day is it?"
351 answer = Ask(api_key).answerQuestion(question)
352 print(question)
353 print(answer)
354 print()
355
356 question = "What is SFBU?"
357 answer = Ask(api_key).answerQuestion(question)
358 print(question)
359 print(answer)
360 print()
361
362 question = "What are MSCS Classes?"
363 answer = Ask(api_key).answerQuestion(question)
364 print(question)
365 print(answer)
366 print()
367
368 question = "Who is the SFBU's President?"
369 answer = Ask(api_key).answerQuestion(question)
370 print(question)
371 print(answer)
372 print()
373
374 if __name__ == "__main__":
375     main()
```

The status bar at the bottom indicates 'Ln 367, Col 46', 'Spaces: 4', 'UTF-8', 'LF', 'Python', and '3.11.5 64-bit'.



A screenshot of a terminal window titled 'Python' showing the execution of the Python script. The terminal displays the output of the script, which matches the answers provided in the code: 'What day is it?' (I don't know.), 'What is SFBU?' (San Francisco Bay University.), 'What are MSCS Classes?' (MSCS classes are graduate level courses in computer science that cover topics such as computer language theory, advanced software development, advanced algorithms, quantum computing, and machine learning.), and 'Who is the SFBU's President?' (Nicholas Ladany). The terminal also shows the command prompt 'PS C:\SFBU\Generative AI>' and the command to run the script: 'cd "c:\SFBU\Generative AI" I/web_qa.py'.

```
PS C:\SFBU\Generative AI> cd "c:\SFBU\Generative AI"
PS C:\SFBU\Generative AI> C:\Users\milti\AppData\Local\Programs\Python\Python311\python.exe "c:\SFBU\Generative A
I/web_qa.py"
What day is it?
I don't know.

What is SFBU?
San Francisco Bay University.

What are MSCS Classes?
MSCS classes are graduate level courses in computer science that cover topics such as computer language theory, adv
anced software development, advanced algorithms, quantum computing, and machine learning.

Who is the SFBU's President?
Nicholas Ladany

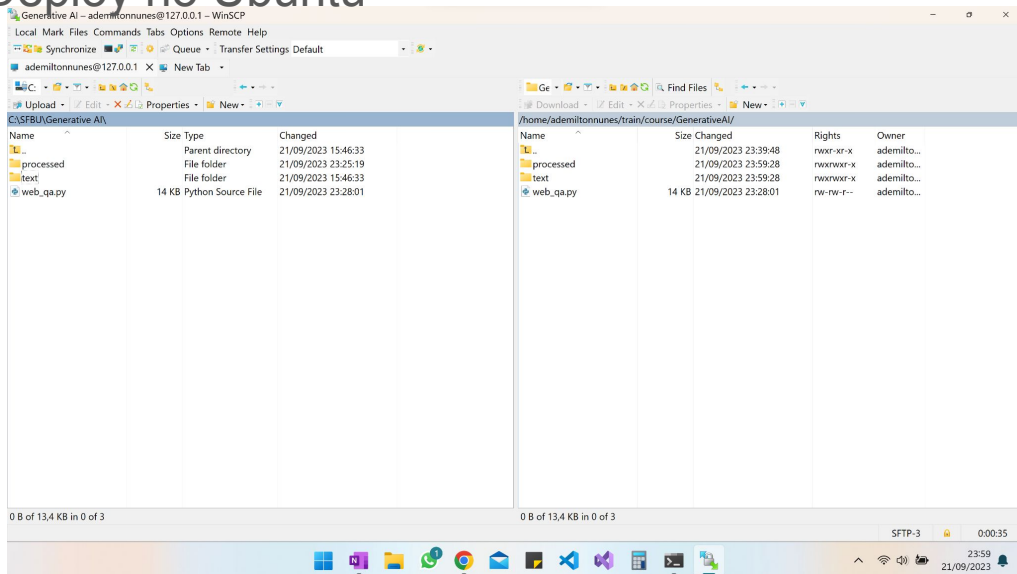
PS C:\SFBU\Generative AI>
```

The status bar at the bottom indicates 'Ln 367, Col 46', 'Spaces: 4', 'UTF-8', 'LF', 'Python', and '3.11.5 64-bit'.

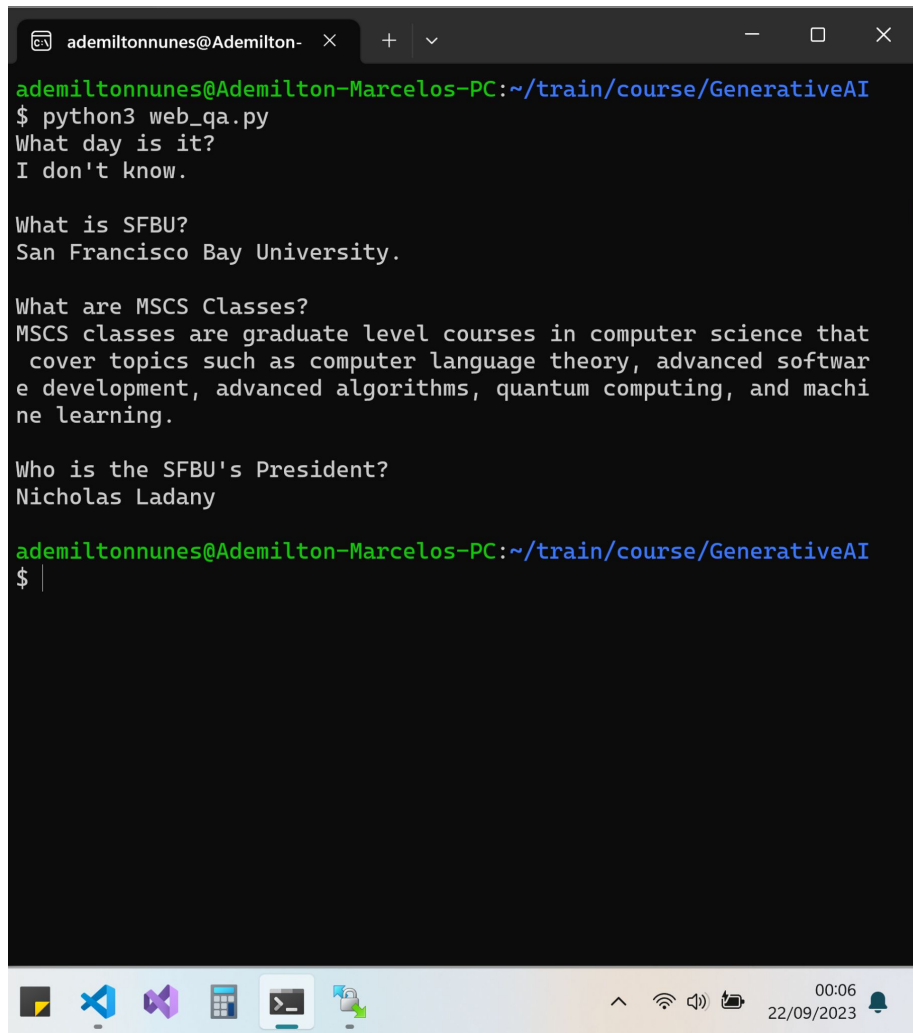
Python on Ubuntu

Running the project on an Ubuntu server, the same questions were asked to carry out the test.

Deploy no Ubuntu



Python on Ubuntu



```
ademiltonnunes@Ademilton-Marcelos-PC:~/train/course/GenerativeAI
$ python3 web_qa.py
What day is it?
I don't know.

What is SFBU?
San Francisco Bay University.

What are MSCS Classes?
MSCS classes are graduate level courses in computer science that
cover topics such as computer language theory, advanced software
development, advanced algorithms, quantum computing, and machine
learning.

Who is the SFBU's President?
Nicholas Ladany

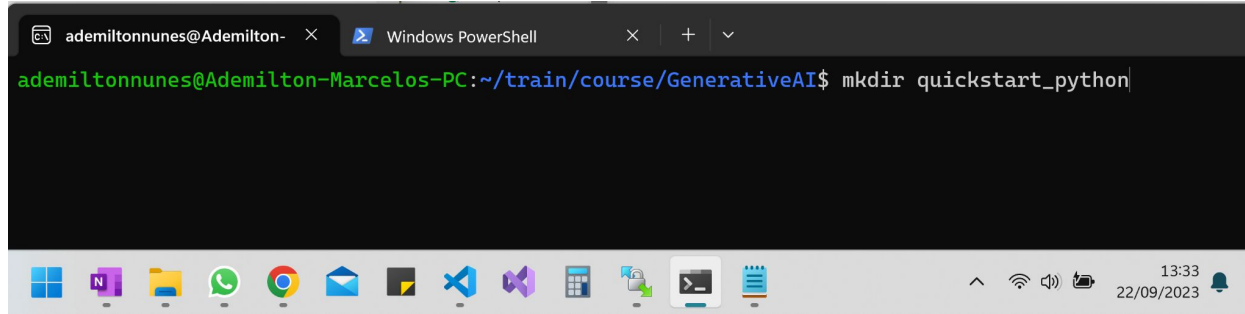
ademiltonnunes@Ademilton-Marcelos-PC:~/train/course/GenerativeAI
$ |
```

Python Flask - Quick-Start

To turn this project into a web-based system that can answer questions about your website, first I downloaded an [openai-quickstart Python Flask tutorial](#), this is a example of Python Flask using Chat GPT to suggest pet's names.

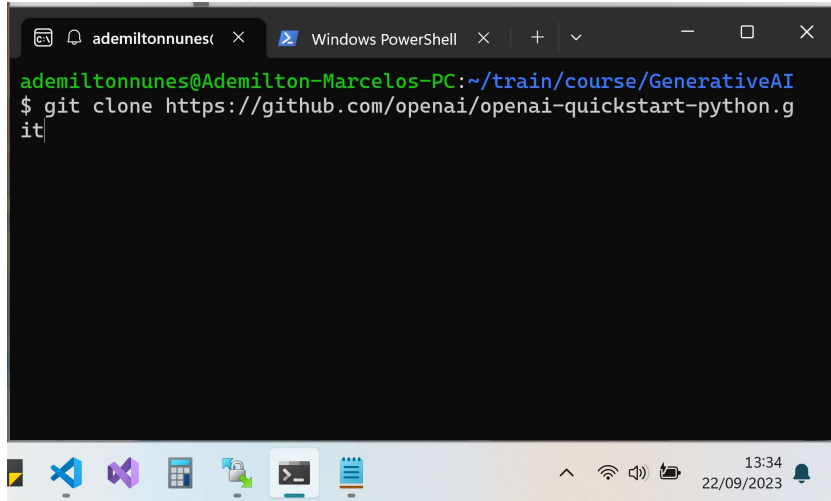
For the Python Flask quickstart project to be possible, it is necessary to download the repository and create a development environment.

Python Flask - Quick-Start



```
ademiltonnunes@Ademilton-PC:~/train/course/GenerativeAI$ mkdir quickstart_python
```

This screenshot shows a Windows PowerShell terminal window. The title bar indicates the user is 'ademiltonnunes' on a machine named 'Ademilton-PC'. The terminal shows the command 'mkdir quickstart_python' being entered at the prompt. The taskbar at the bottom displays various application icons and the system clock showing 13:33 on 22/09/2023.



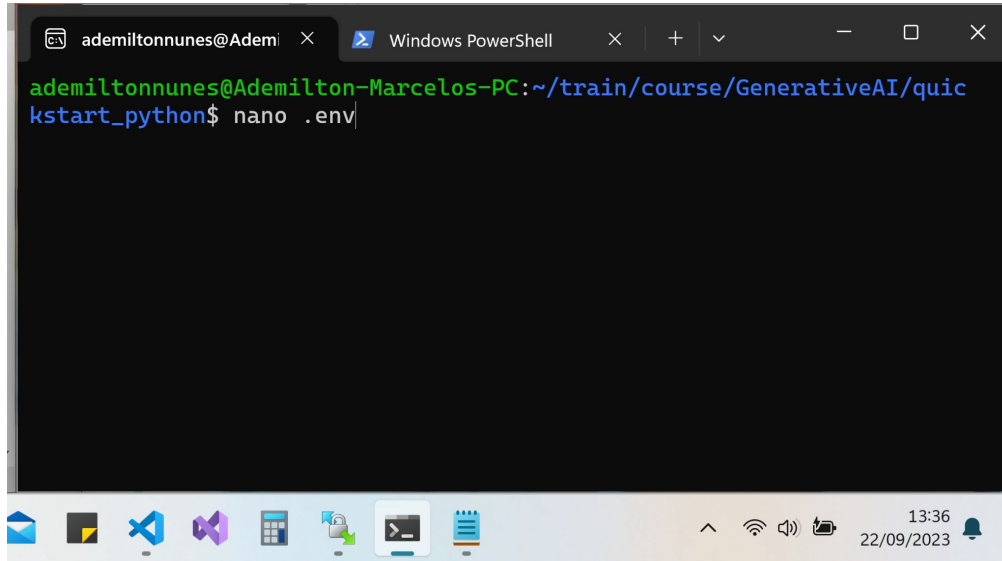
```
ademiltonnunes@Ademilton-Marcelos-PC:~/train/course/GenerativeAI$ git clone https://github.com/openai/openai-quickstart-python.git
```

This screenshot shows a Windows PowerShell terminal window. The title bar indicates the user is 'ademiltonnunes' on a machine named 'Ademilton-Marcelos-PC'. The terminal shows the command 'git clone https://github.com/openai/openai-quickstart-python.git' being entered at the prompt. The taskbar at the bottom displays various application icons and the system clock showing 13:34 on 22/09/2023.

Downloading the project

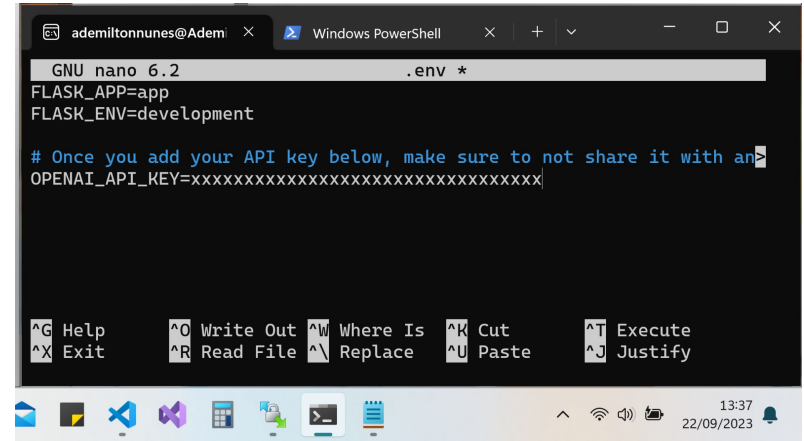
Python Flask - Quick-Start

Configuring API keys



A screenshot of a Windows PowerShell terminal window. The title bar shows 'ademiltonnunes@Ademi' and 'Windows PowerShell'. The terminal content shows the user 'ademiltonnunes@Ademilton-Marcelos-PC' in the directory '~/train/course/GenerativeAI/quic' running the command 'kstart_python\$ nano .env'. The Windows taskbar is visible at the bottom with various application icons and a system clock showing 13:36 on 22/09/2023.

```
ademiltonnunes@Ademilton-Marcelos-PC:~/train/course/GenerativeAI/quic
kstart_python$ nano .env
```



A screenshot of a nano text editor window. The title bar shows 'ademiltonnunes@Ademi' and 'Windows PowerShell'. The editor is editing a file named '.env'. The content of the file includes 'FLASK_APP=app', 'FLASK_ENV=development', and a comment about adding an API key followed by a line of placeholder text 'OPENAI_API_KEY=xx'. The nano editor's help menu is visible at the bottom, showing shortcuts for Help, Exit, Write Out, Read File, Where Is, Replace, Cut, Paste, Execute, and Justify. The Windows taskbar is visible at the bottom with various application icons and a system clock showing 13:37 on 22/09/2023.

```
GNU nano 6.2 .env *
FLASK_APP=app
FLASK_ENV=development

# Once you add your API key below, make sure to not share it with an
OPENAI_API_KEY=xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
```

Python Flask

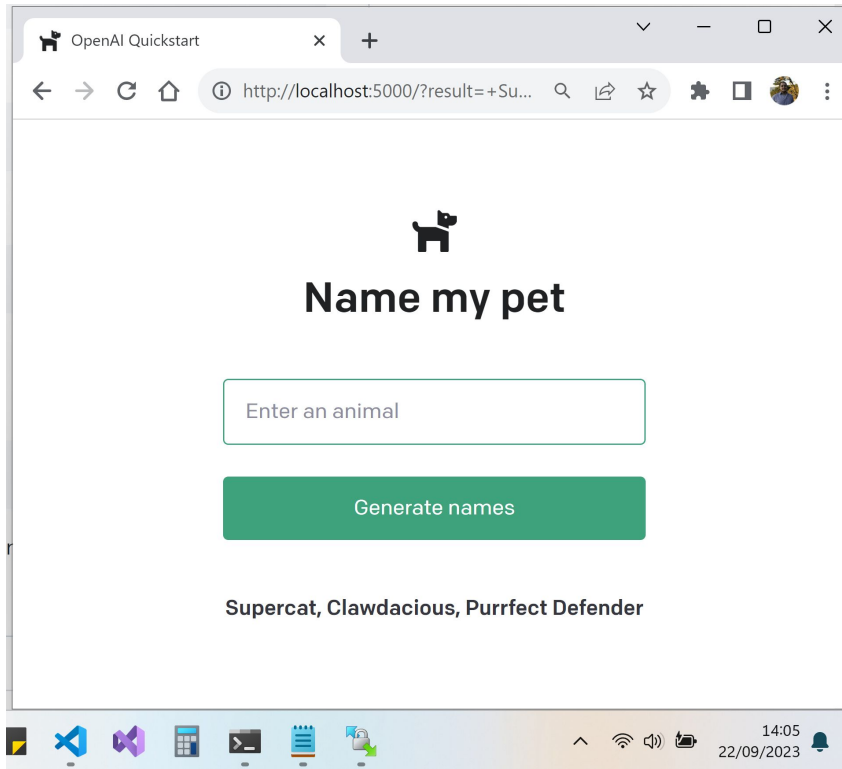
Creating and activating the development environment.

```
ademiltonnunes@Ademilton-Marcelos-PC:~/train/course/GenerativeAI/quickstart_python$ sudo apt install python3.10-venv
[sudo] password for ademiltonnunes:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  python3-pip-whl python3-setuptools-whl
The following NEW packages will be installed:
  python3-pip-whl python3-setuptools-whl python3.10-venv
0 upgraded, 3 newly installed, 0 to remove and 61 not upgraded.
Need to get 2473 kB of archives.
After this operation, 2882 kB of additional disk space will be used.
Do you want to continue? [Y/n]
```

```
ademiltonnunes@Ademilton-Marcelos-PC:~/train/course/GenerativeAI/quickstart_python$ python3 -m venv venv
ademiltonnunes@Ademilton-Marcelos-PC:~/train/course/GenerativeAI/quickstart_python$ . venv/bin/activate
(venv) ademiltonnunes@Ademilton-Marcelos-PC:~/train/course/GenerativeAI/quickstart_python$
```


Python Flask - Quick-Start

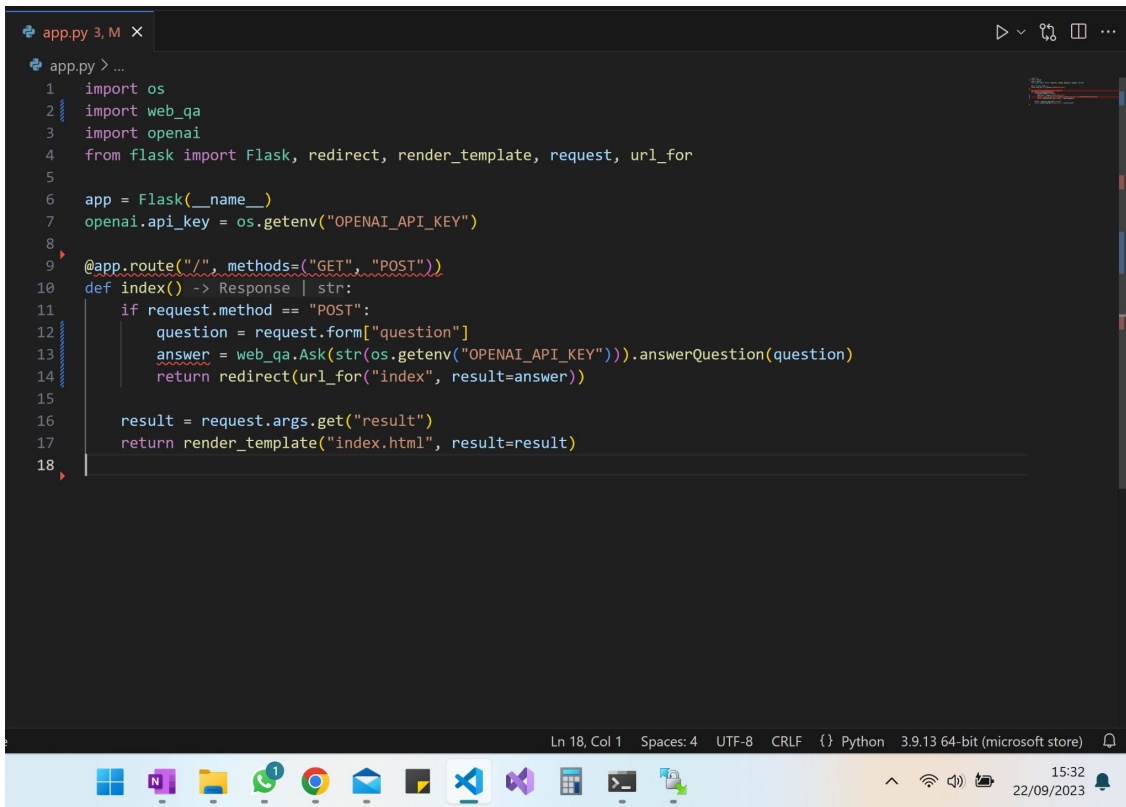
Executing the quick-start project. This project suggests pet names according to the type of pet you have, for example: cats and dogs



Python Flask

Taking the quick-start project as a reference, I integrate it with our projeto of answering questions on Python.

Because I refactored the project into a more object-oriented format, this integration was very simple. It was only necessary to import the class that receives and responds to questions and the system in Python was ready to work to answer questions.



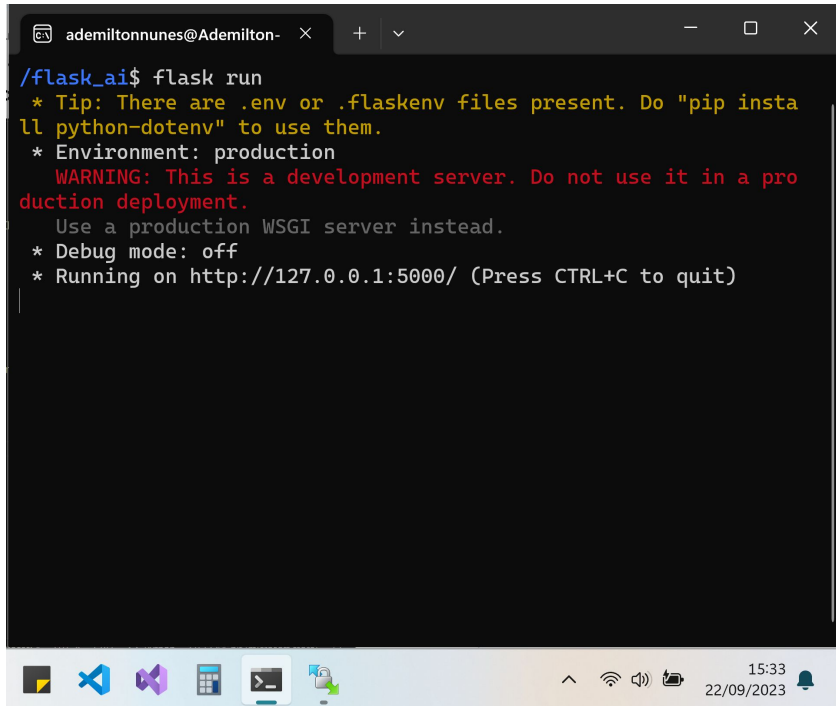
```
app.py 3, M X
app.py > ...
1 import os
2 import web_qa
3 import openai
4 from flask import Flask, redirect, render_template, request, url_for
5
6 app = Flask(__name__)
7 openai.api_key = os.getenv("OPENAI_API_KEY")
8
9 @app.route("/", methods=("GET", "POST"))
10 def index() -> Response | str:
11     if request.method == "POST":
12         question = request.form["question"]
13         answer = web_qa.Ask(str(os.getenv("OPENAI_API_KEY"))).answerQuestion(question)
14         return redirect(url_for("index", result=answer))
15
16     result = request.args.get("result")
17     return render_template("index.html", result=result)
18
```

Ln 18, Col 1 Spaces: 4 UTF-8 CRLF {} Python 3.9.13 64-bit (microsoft store)

15:32 22/09/2023

Python Flask

The Python Flask project also uses Ubuntu. Ubuntu is being used as a server.

A terminal window with a dark background and light-colored text. The window title bar shows 'ademiltonnunes@Ademilton-'. The terminal output shows the command '/flask_ai\$ flask run' followed by several informational messages from Flask, including a tip about .env files, the environment set to 'production', a warning about using a development server, and the server running on 'http://127.0.0.1:5000/'. The bottom of the image shows a Linux desktop environment with various application icons and a system tray with the date '22/09/2023' and time '15:33'.

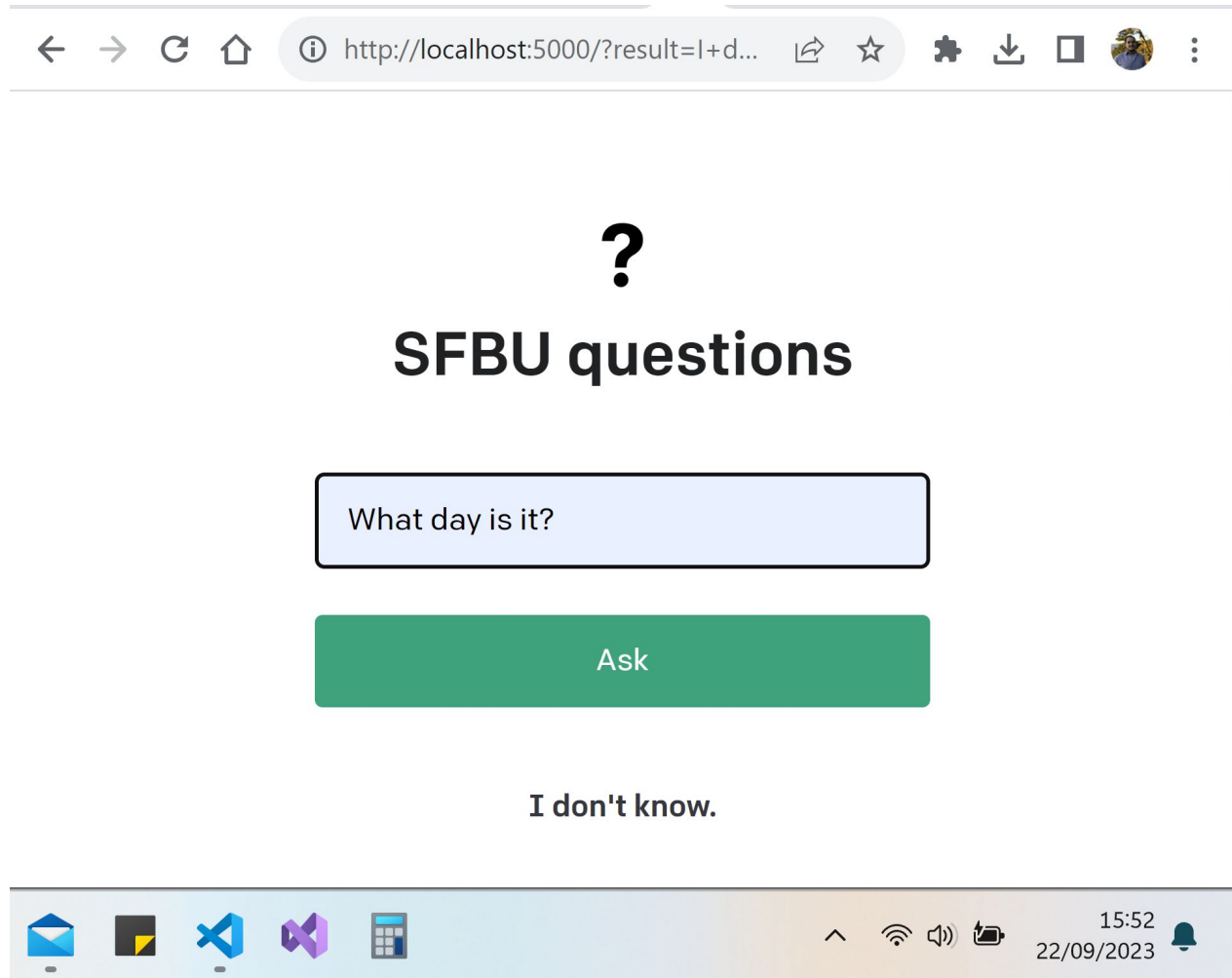
```
ademiltonnunes@Ademilton- x + v - □ ×  
/flask_ai$ flask run  
* Tip: There are .env or .flaskenv files present. Do "pip install python-dotenv" to use them.  
* Environment: production  
  WARNING: This is a development server. Do not use it in a production deployment.  
  Use a production WSGI server instead.  
* Debug mode: off  
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

Starting ubuntu to be a server

Python Flask

With the Python Flask project, I submitted the same questions to compare with previous results. The same answers were gotten.

Python Flask



Python Flask

? SFBU questions

What are MSCS Classes?

Ask

MSCS classes are graduate level courses in computer science that cover topics such as computer language theory, advanced software development, advanced algorithms, quantum computing, and machine learning.



15:52
22/09/2023



Python Flask

?

SFBU questions

Who is the SFBU's President?

Ask

Nicholas Ladany



15:53

22/09/2023



Python Flask

?

SFBU questions

What is SFBU?

Ask

San Francisco Bay University.



15:51
22/09/2023



Conclusion

The tutorial [OpenAI document - Website Q&A with Embeddings](#) in which implement a customer support system using ChatGPT to build a web-based system that can answer questions about a website was the starting point for being able to develop this in web-based. It offered success at all stages of the process and can be easily adapted and integrated. In this project, we demonstrate the process in Jupyter Notebook, Python being on Ubuntu and Web-based in Python Flask. The entire website process can be found in the github repository added in this document