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)

Table of Content

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.

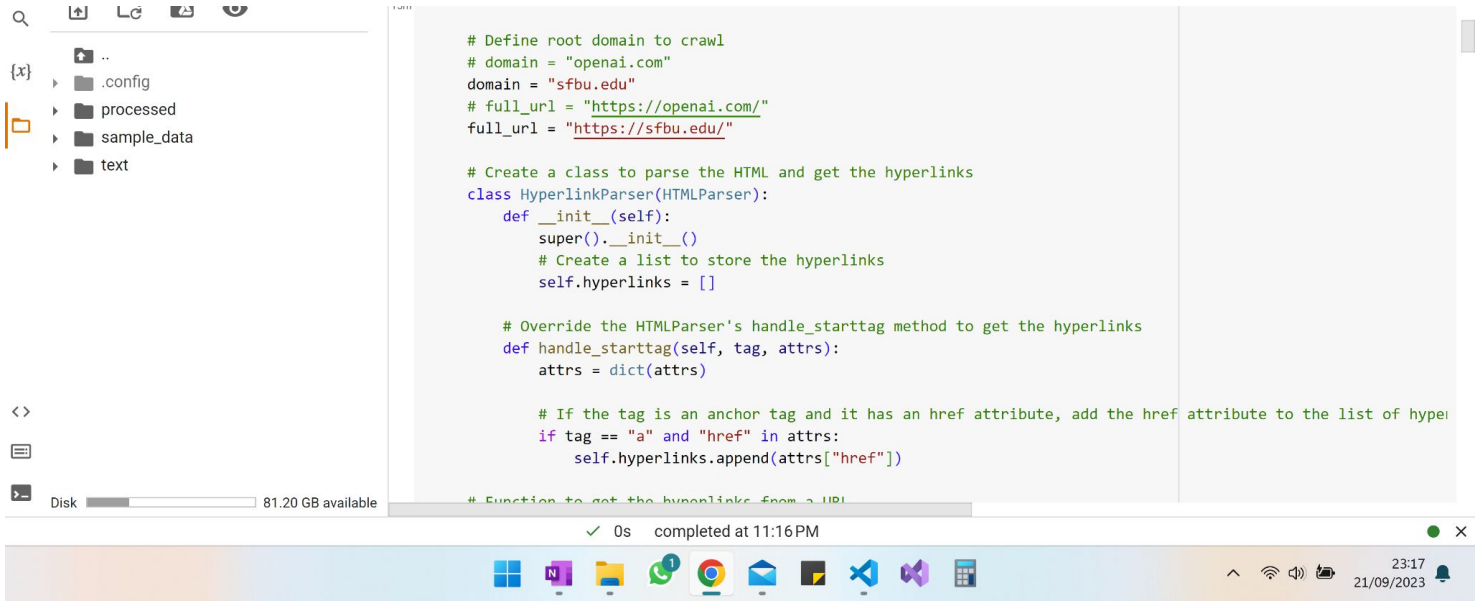
All data is embedding into indexes so that ChatGPT can use this data to answer questions

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.

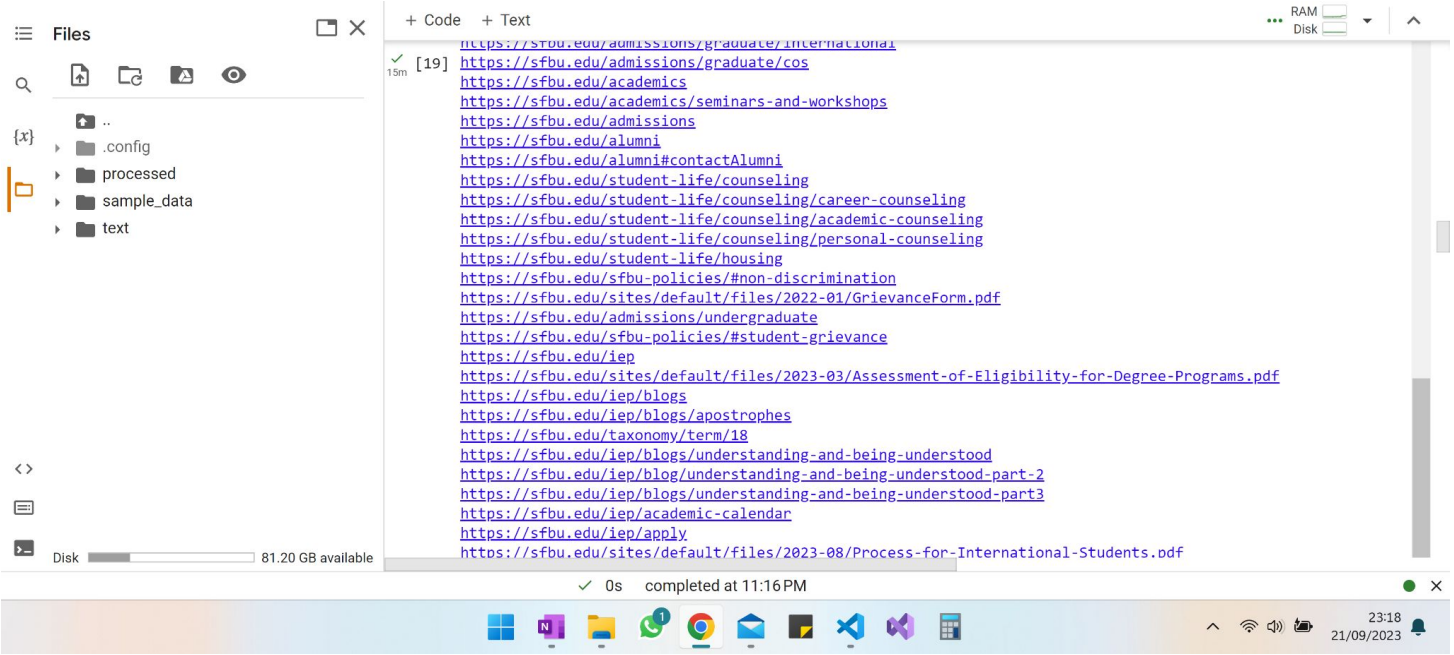
Jupyter Notebook

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



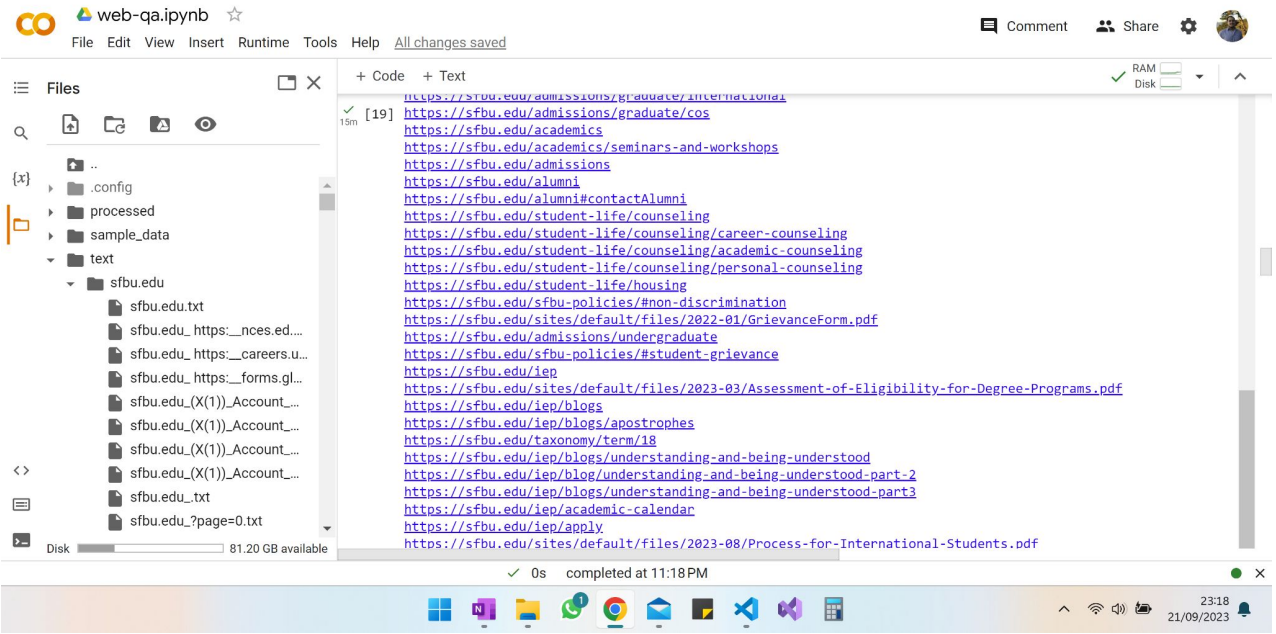
Jupyter Notebook

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



Jupyter Notebook

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

web-qa.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

Files

..
.config
processed
embeddings.csv
scraped.csv
sample_data
text

Disk 81.20 GB available

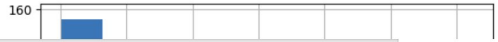
+ Code + Text

[21]

1	og alumni dominates for high tech Jobs	og alumni dominates for high tech Jobs. ...
2	udent life transportation easypass	udent life transportation easypass. SFBU...
3	tes default files 2023 05 Chaffey College.pdf	tes default files 2023 05 Chaffey College.pdf....
4	p contact us	p contact us. English as a Second Langua...

```
# Load the cl100k_base tokenizer which is designed to work with the ada-002 model
tokenizer = tiktoken.get_encoding("cl100k_base")

df = pd.read_csv('processed/scraped.csv', index_col=0)
df.columns = ['title', 'text']

# Tokenize the text and save the number of tokens to a new column
df['n_tokens'] = df.text.apply(lambda x: len(tokenizer.encode(x)))

# Visualize the distribution of the number of tokens per row using a histogram
df.n_tokens.hist()
```

```
<Axes: >
```

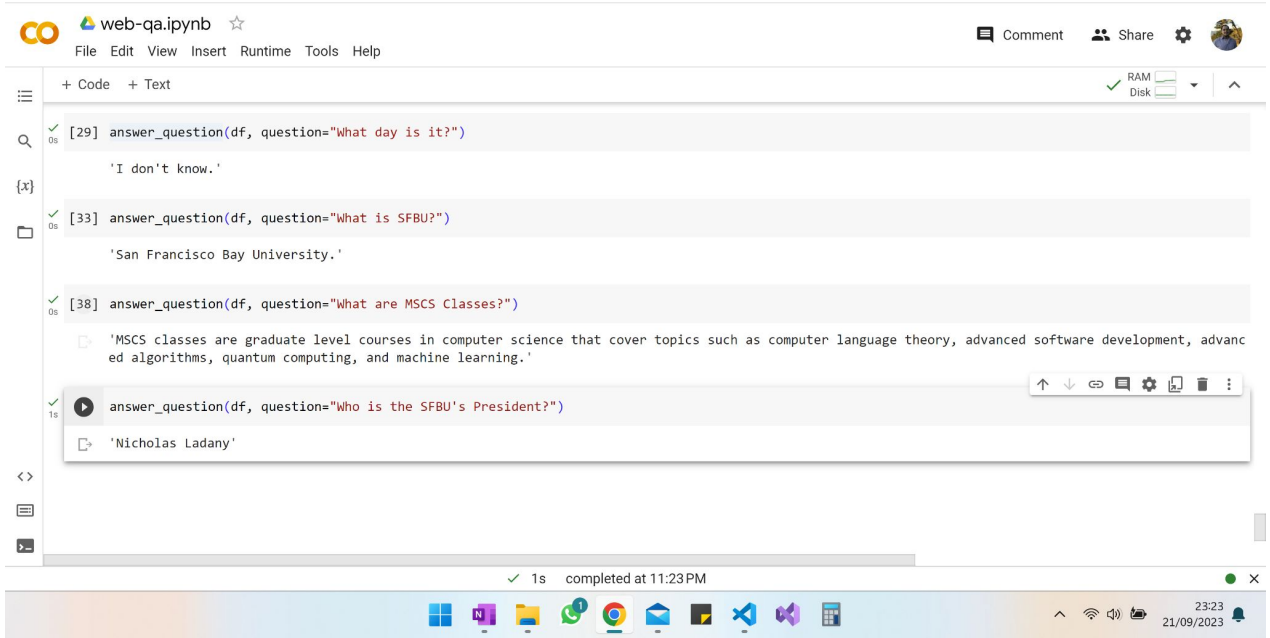


0s completed at 11:18 PM

23:19
21/09/2023

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"



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 9+

web_qa.py > main

```python
import requests
import re
import urllib.request
from bs4 import BeautifulSoup
from collections import deque
from html.parser import HTMLParser
from urllib.parse import urlparse
import os
import pandas as pd
import tiktoken
import openai
import numpy as np
from openai.embeddings_utils import distances_from_embeddings, cosine_similarity
from ast import literal_eval

# Create a class to parse the HTML and get the hyperlinks
class HyperlinkParser(HTMLParser):···

class HyperLink():···

class Crawl():···

class Embed():···

class Ask():···

def main() -> None:
    # Define API Key
    # api_key:str = '<Your API Key>'
```

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

web_qa.py 9+

web_qa.py > main

```python
    question = "What day is it?"
    answer = Ask(api_key).answerQuestion(question)
    print(question)
    print(answer)
    print()

    question = "What is SFBU?"
    answer = Ask(api_key).answerQuestion(question)
    print(question)
    print(answer)
    print()

    question = "What are MSCS Classes?"
    answer = Ask(api_key).answerQuestion(question)
    print(question)
    print(answer)
    print()

    question = "Who is the SFBU's President?"
    answer = Ask(api_key).answerQuestion(question)
    print(question)
    print(answer)
    print()

if __name__== "__main__":
    main()
```

Ln 367, Col 46 Spaces: 4 UTF-8 LF {} Python 3.11.5 64-bit

23:28 21/09/2023

PROBLEMS 197 OUTPUT DEBUG CONSOLE TERMINAL PORTS Python

```
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>
```

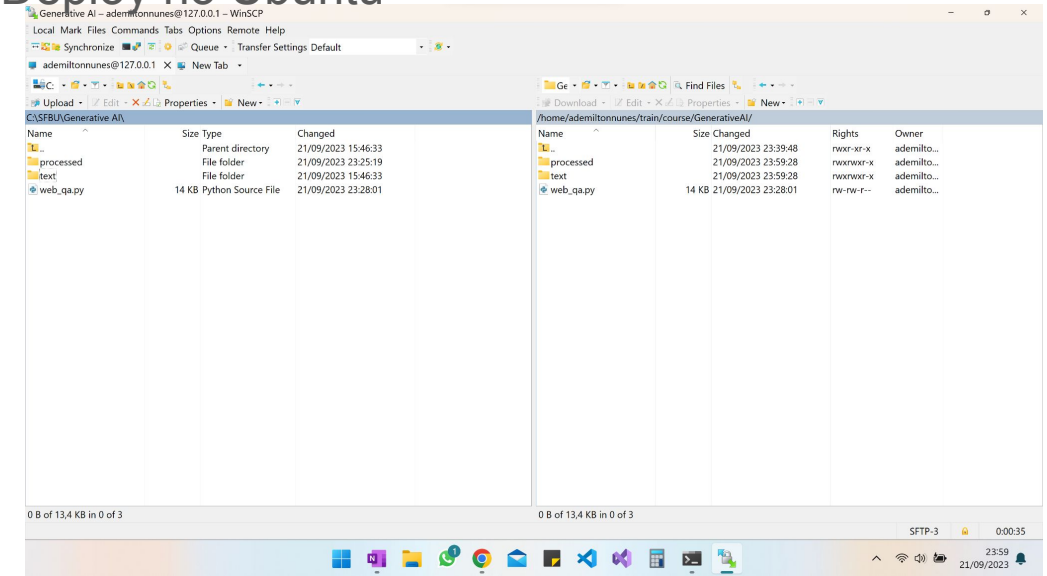
Ln 367, Col 46 Spaces: 4 UTF-8 LF {} Python 3.11.5 64-bit

23:29 21/09/2023

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 softwar
e development, advanced algorithms, quantum computing, and machi
ne learning.

Who is the SFBU's President?
Nicholas Ladany

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

00:06
22/09/2023

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-Marcelos-PC:~/train/course/GenerativeAI$ mkdir quickstart_python
```

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

Downloading the project

Python Flask - Quick-Start

Configuring API keys

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

13:36
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=xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

^G Help      ^O Write Out ^W Where Is  ^K Cut       ^T Execute
^X Exit      ^R Read File ^\ Replace   ^U Paste     ^J Justify
```

13:37
22/09/2023

Python Flask

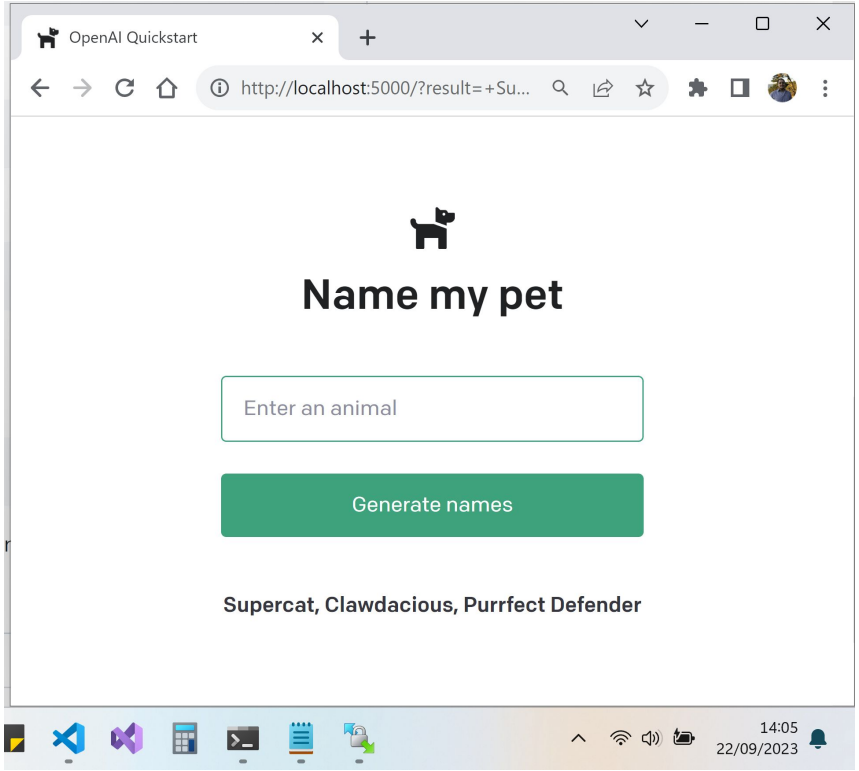
Creating and activating the development environment.

```
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. Iit 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

app.py > ...

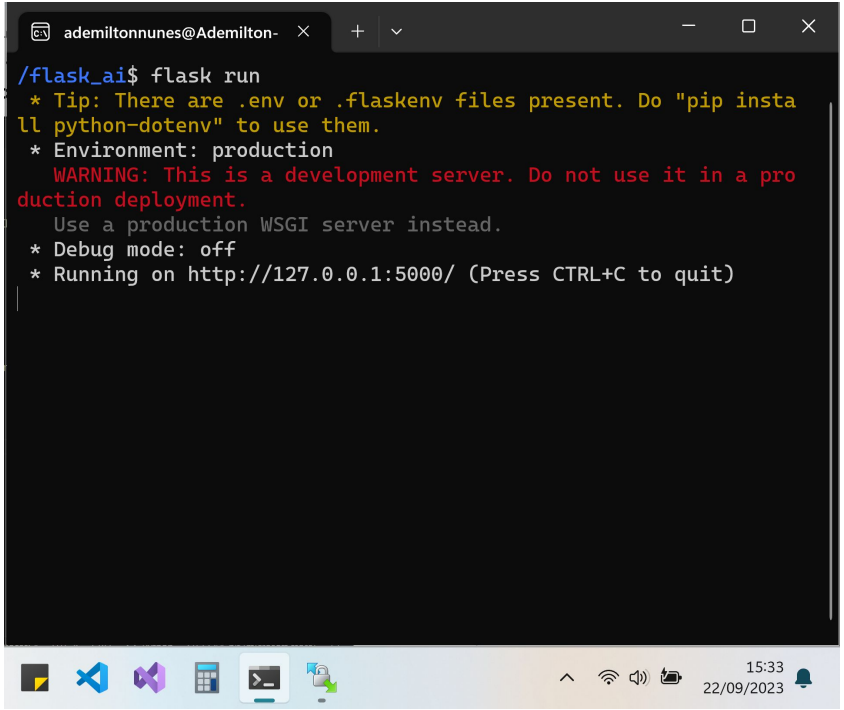
```python
import os
import web_qa
import openai
from flask import Flask, redirect, render_template, request, url_for

app = Flask(__name__)
openai.api_key = os.getenv("OPENAI_API_KEY")

@app.route("/", methods=("GET", "POST"))
def index() -> Response | str:
    if request.method == "POST":
        question = request.form["question"]
        answer = web_qa.Ask(str(os.getenv("OPENAI_API_KEY"))).answerQuestion(question)
        return redirect(url_for("index", result=answer))

    result = request.args.get("result")
    return render_template("index.html", result=result)
```

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.

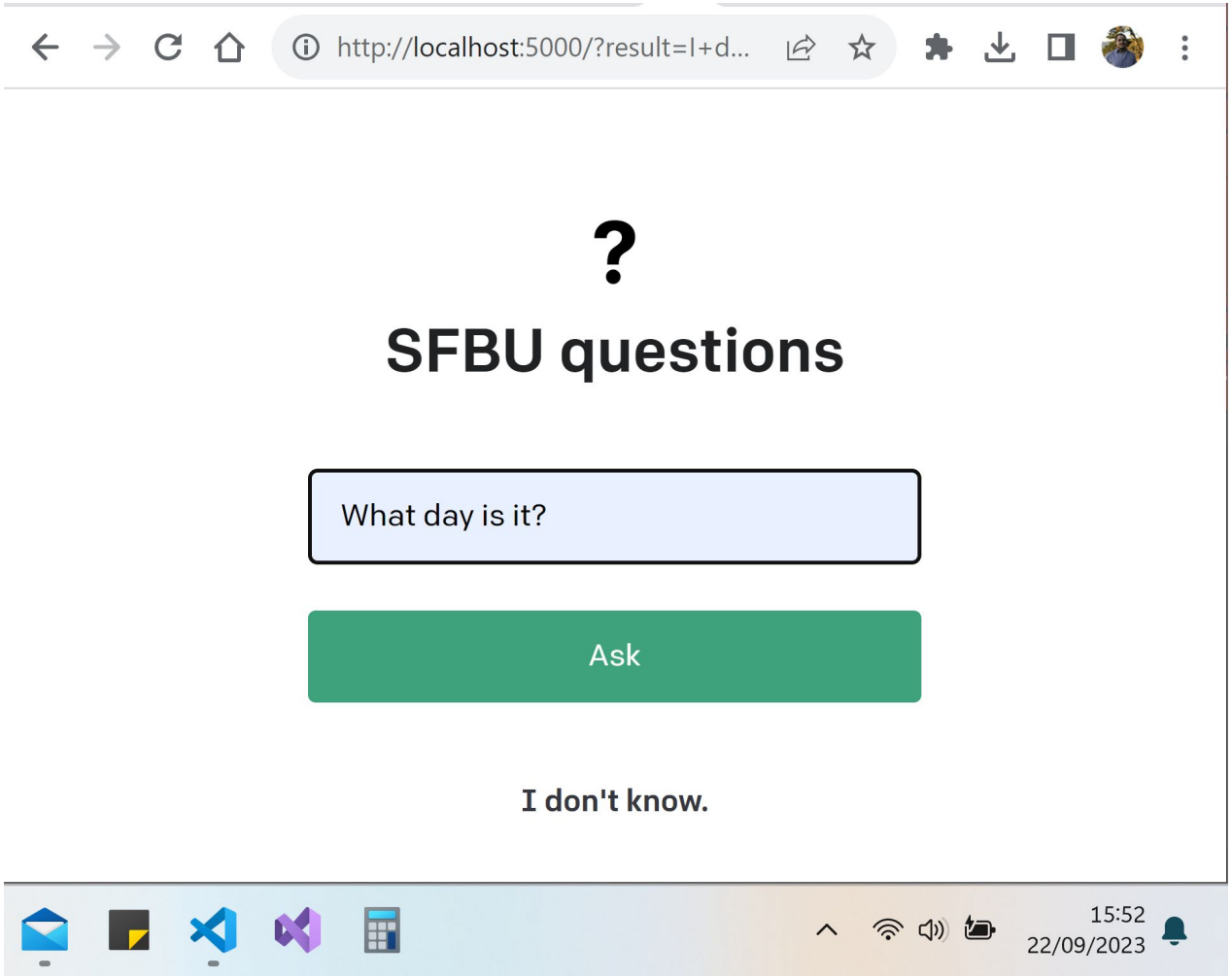


Starting ubuntu to be a server

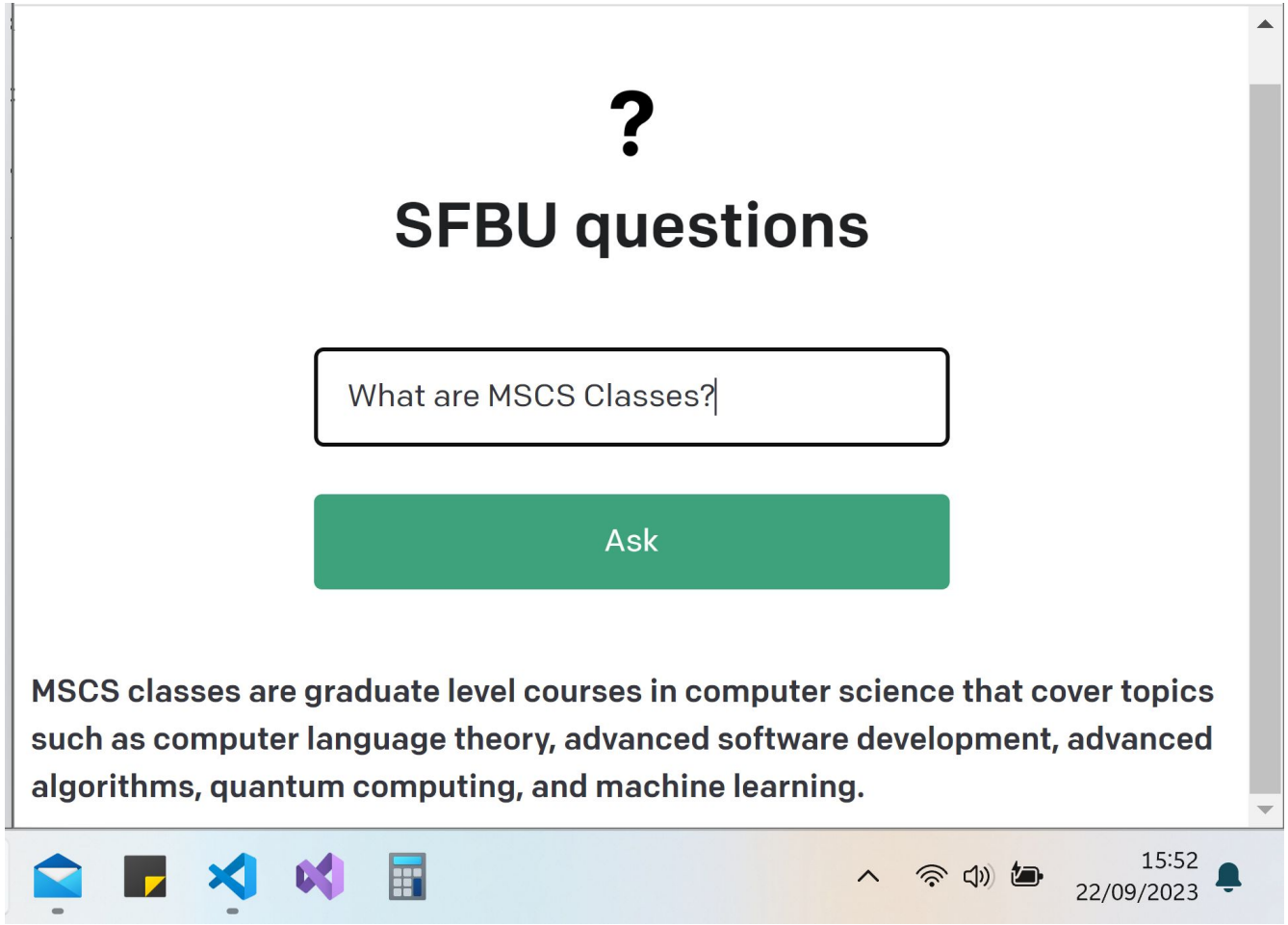
Python Flask

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

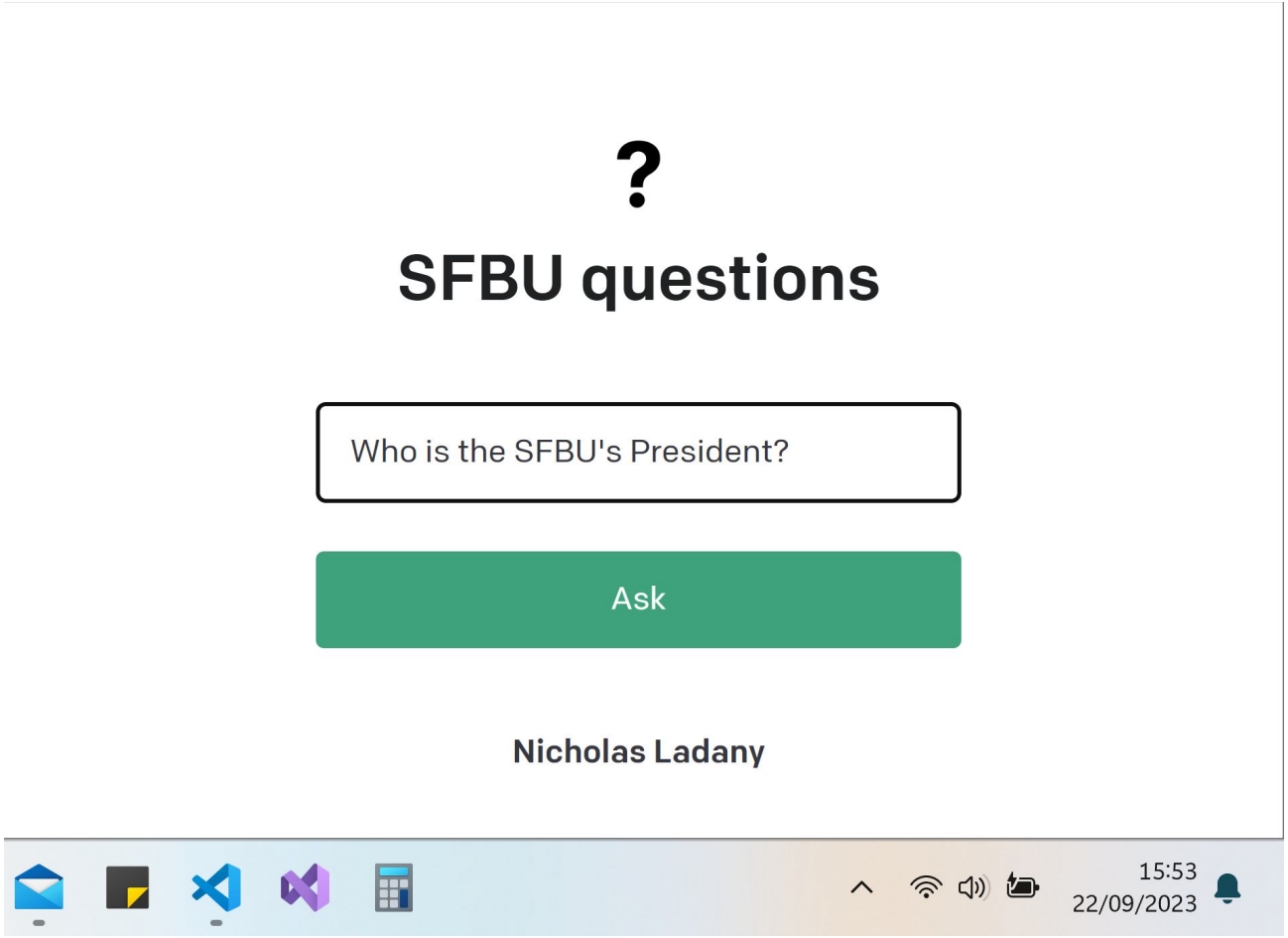
Python Flask



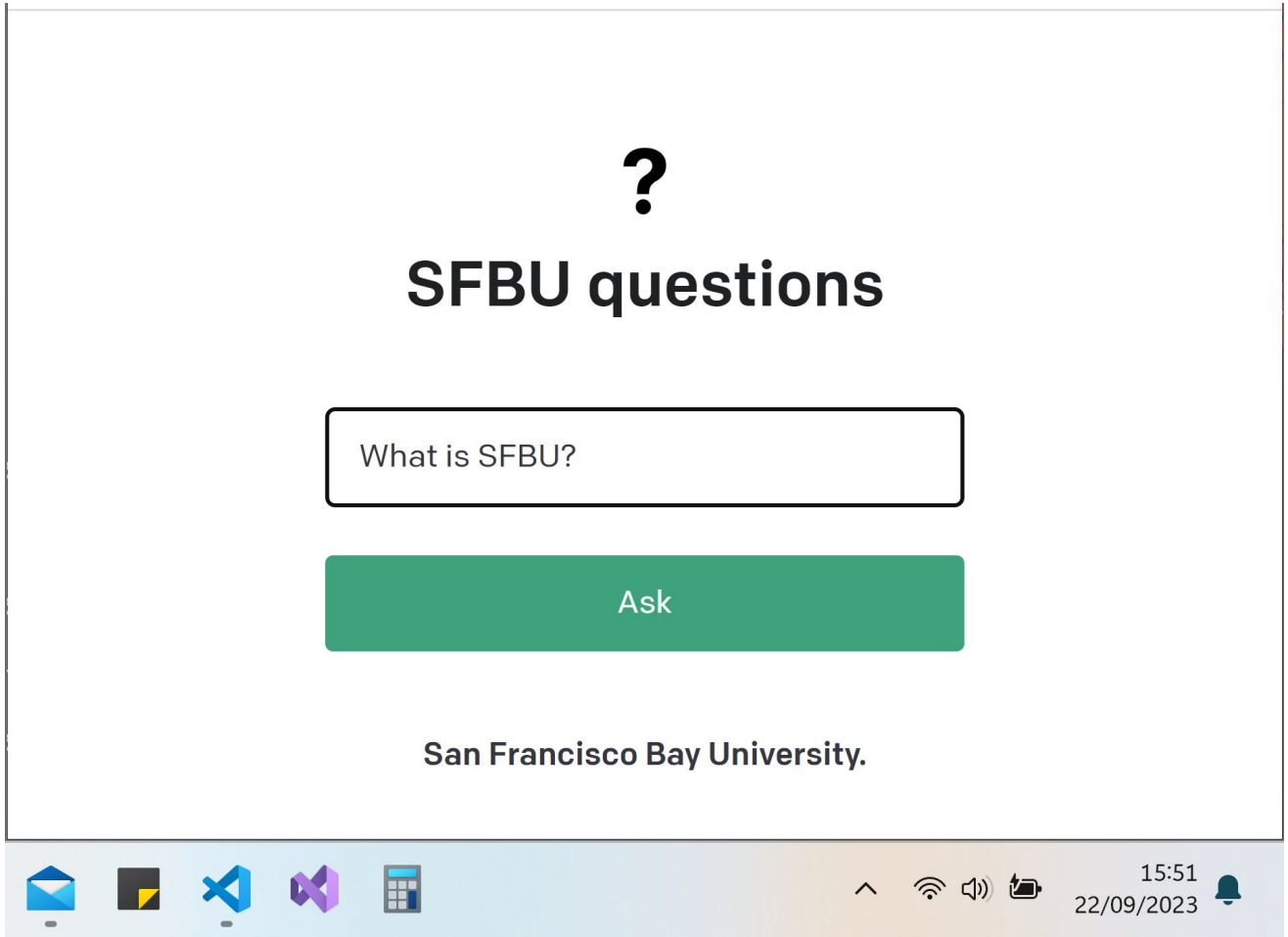
Python Flask



Python Flask



Python Flask



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