ChatGPT project : Customer Support System

Khoi Duong
Prof. Chang
CS589
1/31/2024

Step 1.1 to 1.2

Please link to the previous document on this project at:

Week 2 HW 1 - CS589 - Khoi Duong - 19610

For this presentation, we will focus on Step 1.3 (Web-base Solution - Node.js webserver)

- Step 1.3: Web-based Solution (Node.js webserver)

  Enhance the result of Step 1.1 to allow users to ask questions about the website using a browser. - Javascript (Node.js) based

  - Step 1.3.1: Study how to use Javascript (Node.js) to create a web-based interface to ChatGPT
    - Note:
      - The Javascript (Node.js) program should be run on Ubuntu
  - Step 1.3.2: Integrate the Javascript (Node.js) code created in Step 1.1 and Step 1.3.1 to a create a web-based interface to let the users ask ChatGPT questions about the website using a browser.
    - Hint
      - Node.js cals Python code

Step 1.3.1: Study how to use Javascript (Node.js) to create a web-based interface to ChatGPT

Take reference from
https://hc.labnet.sfbu.edu/~henry/sfbu/course/machine_learning/chatgpt/slide/quickstart.html#Quickstart%20-%20Node.js

---

A. Install version 19.x node.js on Windows' Ubuntu
B. Download the ChatGPT sample code by cloning this repository.
1. Creating the working directory

```
mkdir quickstart_node; cd quickstart_node
```

2. Download the code to the working directory

git clone https://github.com/openai/openai-quickstart-node.git

C. Add your API key

```
cd openai-quickstart-node
cp .env.example .env
vi .env
```

Step 1.3.1

```
PS D:\VS CODE\Python\CS589> git clone https://github.com/openai/openai-quickstart-node.git

Cloning into 'openai-quickstart-node'...
remote: Enumerating objects: 120, done.
remote: Counting objects: 100% (66/66), done.
remote: Compressing objects: 100% (30/30), done.
remote: Total 120 (delta 50), reused 36 (delta 36), pack-reused 54Receiving objects:  82%
Receiving objects: 100% (120/120), 93.50 KiB | 1.42 MiB/s, done.

Resolving deltas: 100% (52/52), done.
PS D:\VS CODE\Python\CS589>
```

Step 1.3.1

```
PS D:\VS CODE\Python\CS589> cd .\openai-quickstart-node\
PS D:\VS CODE\Python\CS589\openai-quickstart-node> cp .env.example .env
PS D:\VS CODE\Python\CS589\openai-quickstart-node> vi .env
```

Then, run ‘npm install’ and ‘npm run dev’ to start the program.

```
PS D:\VS CODE\Python\CS589\openai-quickstart-node> npm install

added 26 packages, and audited 27 packages in 31s

3 packages are looking for funding
  run `npm fund` for details

5 moderate severity vulnerabilities

To address issues that do not require attention, run:
  npm audit fix

To address all issues (including breaking changes), run:
  npm audit fix --force

Run `npm audit` for details.
PS D:\VS CODE\Python\CS589\openai-quickstart-node>
```

Step 1.3.1

We can see the result at http://localhost:3000

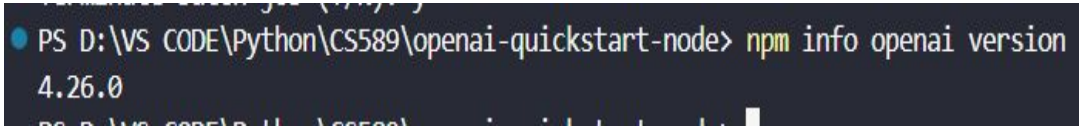
Step 1.3.1

We have the errors below:

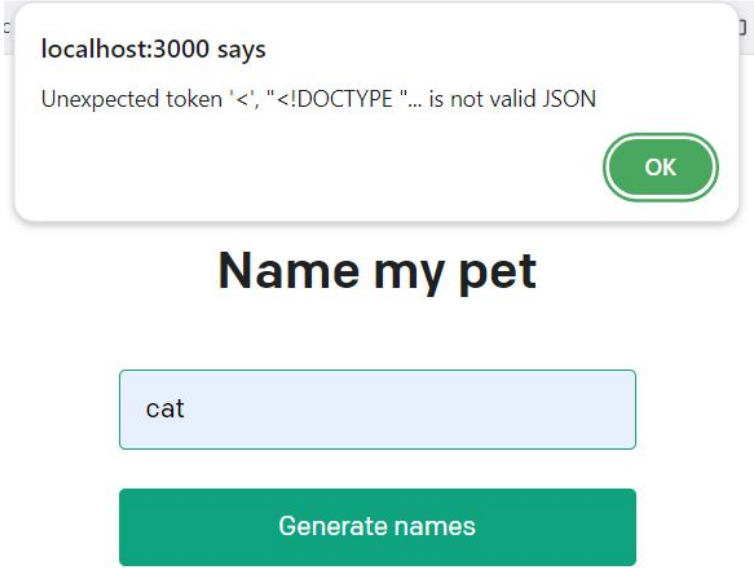
```
 ✓ Ready in 4.4s
Browserslist: caniuse-lite is outdated. Please run:
  npx browserslist@latest --update-db
  Why you should do it regularly: https://github.com/browserslist/browserslist#browsers-da
ta-updating
 o Compiling / ...
 ✓ Compiled / in 4.7s (199 modules)
 ⚠ Fast Refresh had to perform a full reload. Read more: https://nextjs.org/docs/messages/
fast-refresh-reload
 ✓ Compiled /api/generate in 185ms (65 modules)
 × pages\api\generate.js (3:22) @ eval
 × TypeError: openai__WEBPACK_IMPORTED_MODULE_0__.Configuration is not a constructor
    at eval (webpack-internal:///(api)/./pages/api/generate.js:10:23)
  1 | import { Configuration, OpenAIApi } from "openai";
  2 |
> 3 | const configuration = new Configuration({
    |                       ^
  4 |   apiKey: process.env.OPENAI_API_KEY,
  5 | });
  6 | const openai = new OpenAIApi(configuration);

Terminate batch job (Y/N)? y
```

Step 1.3.1

After taking some reference from [OpenAI API error: "Module 'openai' has no exported member 'Configuration'" - Stack Overflow](), it appears that the OpenAI NodeJS SDK is v4, but the code we use is in v3 version.

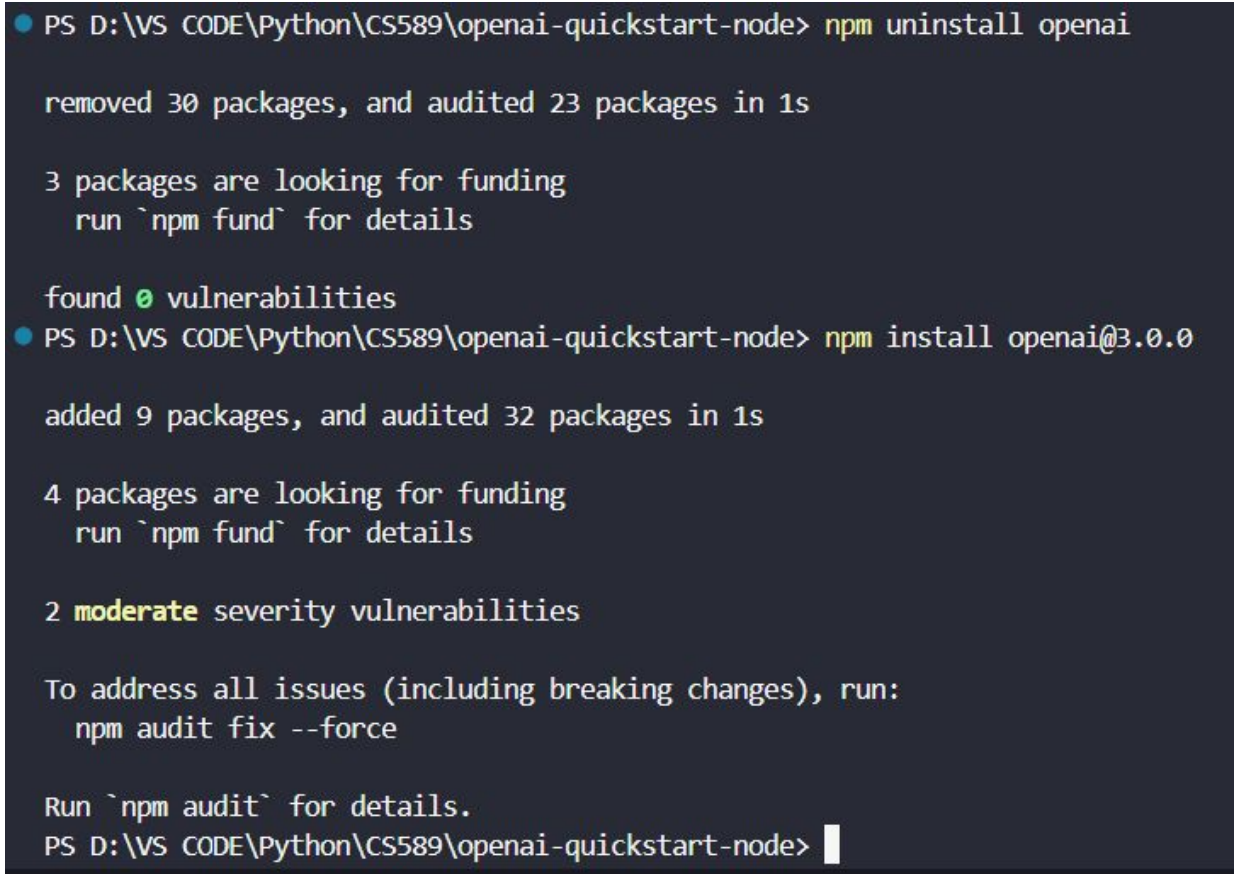


After checking, it is true that we have v4 version.



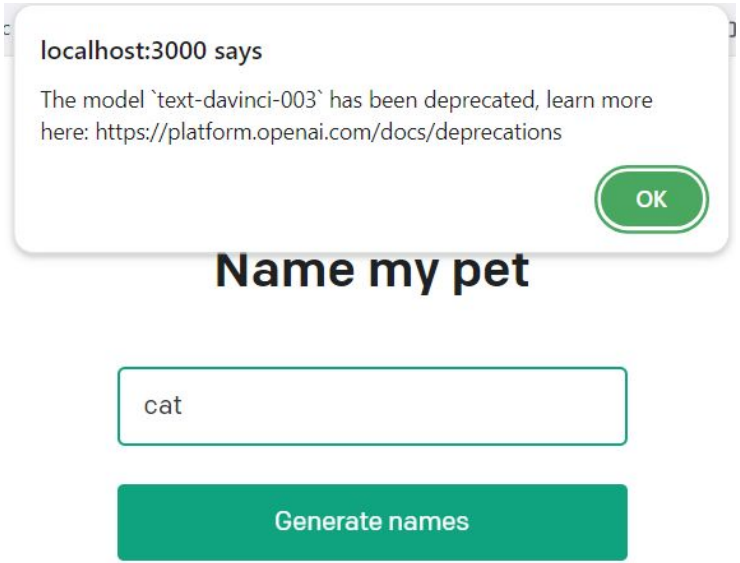
Step 1.3.1

The link provided above also gives us a solution to the problem. However, I suggest one easy and fast way to fix this problem. Instead of migrating the code which may yield more errors, we can simply install the v3 of OpenAI NodeJS SDK.



Step 1.3.1

Try again and we get another familiar problems:




```
PS D:\VS CODE\Python\CS589\openai-quickstart-node> npm run dev

> openai-quickstart-node@0.1.0 dev
> next dev

  ▲ Next.js 13.5.6
  - Local:        http://localhost:3000
  - Environments: .env

 ✓ Ready in 4.7s
 ✓ Compiled / in 2.1s (197 modules)
 ✓ Compiled /api/generate in 203ms (63 modules)
404 {
  error: {
    message: 'The model `text-davinci-003` has been deprecated, learn more here: https://p
latform.openai.com/docs/deprecations',
    type: 'invalid_request_error',
    param: null,
    code: 'model_not_found'
  }
}
```

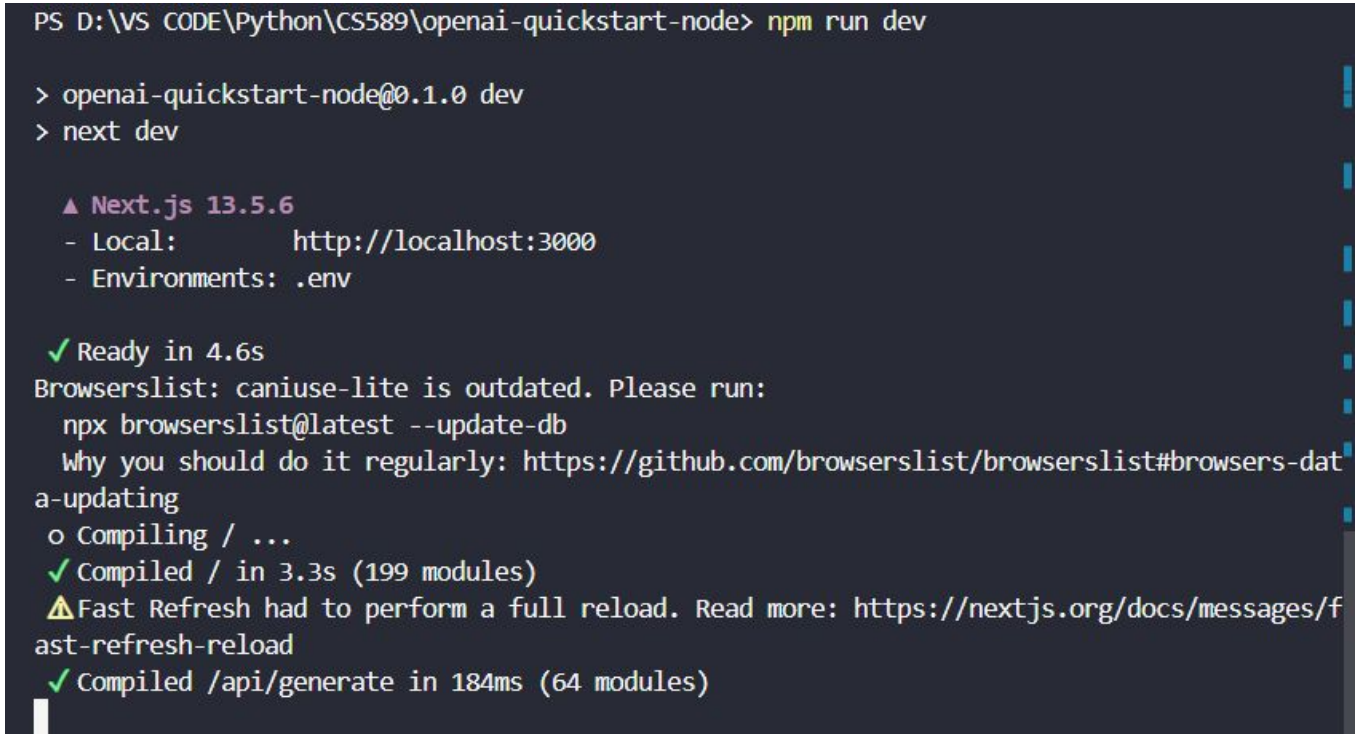
Step 1.3.1

We just have to change \openai-quickstart-node\pages\api\generate.js (line 30) into “gpt-3.5-turbo-instruct” (since the model “text-davinci-003” has been deprecated)

```
28    try {
29      const completion = await openai.createCompletion({
30        model: "gpt-3.5-turbo-instruct",
31        prompt: generatePrompt(animal),
32        temperature: 0.6,
33      });
34      res.status(200).json({ result: completion.data.choices[0].text });
35    } catch(error) {
36      // Consider adjusting the error handling logic for your use case
```

Step 1.3.1

Run again and we have successful result:



Name my pet

hamster

Generate names

Super Scurry, Mighty Rodent, Hamtaro the Hero

Name my pet

cat

Generate names

Purrfect Avenger, Clawed Crusader, Super Whiskers

Name my pet

dog

Generate names

Super Pup, Canine Crusader, Mighty Mutt

Name my pet

horse

Generate names

Galloping Guardian, Mighty Mustang, Super Stallion

Step 1.3.2: Integrate the Javascript (Node.js) to create a web-based interface

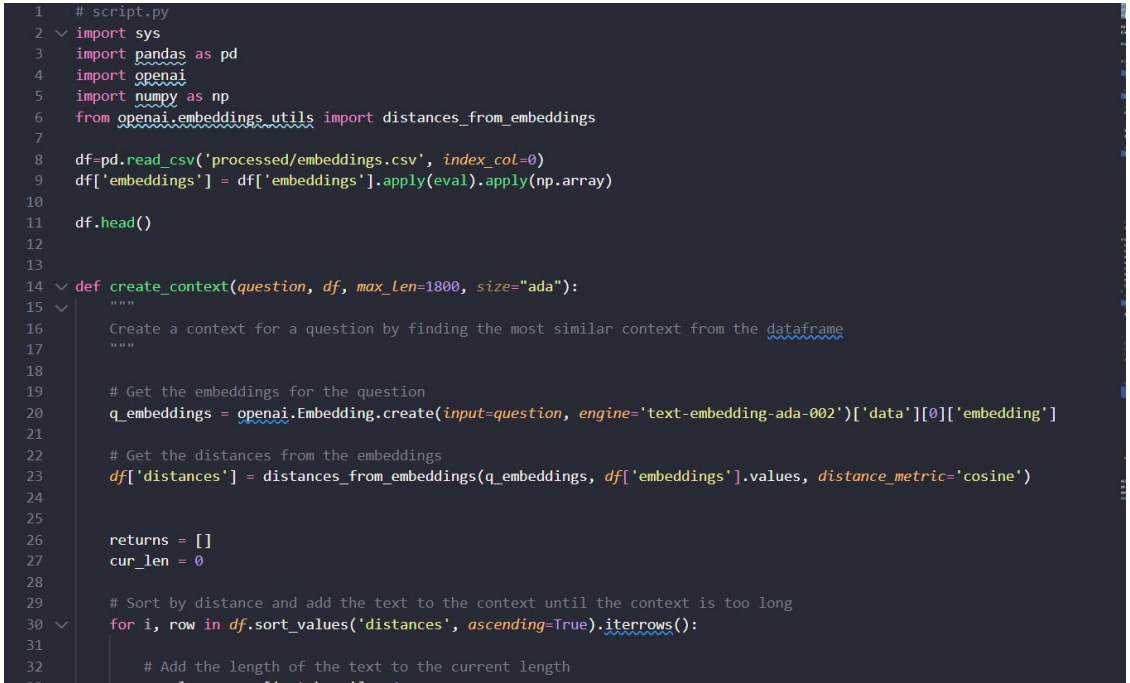
We will apply the same packet of code from step 1.3.1 and the Python program from our step 1.2 to create a web-based interface using Node.js

Then, we start to modify a little on the package.json to add our scripts to the Python program (which run crawldata.py and embeddata.py later)

```
"scripts": {
  "dev": "next dev",
  "build": "next build",
  "start": "next start",
  "lint": "next lint",
  "install": "pip install -r requirements.txt",
  "crawl": "python3 crawldata.py",
  "embedding": "python3 embeddata.py"
},
```

Step 1.3.2

Create ‘script.py’ as a Python script which will be called later by Nodejs (full code in GitHub file)



```python
# script.py
import sys
import pandas as pd
import openai
import numpy as np
from openai.embeddings_utils import distances_from_embeddings

df=pd.read_csv('processed/embeddings.csv', index_col=0)
df['embeddings'] = df['embeddings'].apply(eval).apply(np.array)

df.head()


def create_context(question, df, max_len=1800, size="ada"):
    """
    Create a context for a question by finding the most similar context from the dataframe
    """

    # Get the embeddings for the question
    q_embeddings = openai.Embedding.create(input=question, engine='text-embedding-ada-002')['data'][0]['embedding']

    # Get the distances from the embeddings
    df['distances'] = distances_from_embeddings(q_embeddings, df['embeddings'].values, distance_metric='cosine')


    returns = []
    cur_len = 0

    # Sort by distance and add the text to the context until the context is too long
    for i, row in df.sort_values('distances', ascending=True).iterrows():

        # Add the length of the text to the current length
```

Step 1.3.2

Create ‘pages/api/getResult.js’ (full code in GitHub file)

```
// pages/api/getResult.js
import { exec } from 'child_process';

// Set your OpenAI API key as an environment variable
const OPENAI_API_KEY = process.env.OPENAI_API_KEY;
if (!OPENAI_API_KEY) {
  console.error('OPENAI_API_KEY is not set. Please set your API key.');
  process.exit(1);
}

// // run embedText.py
// const { spawn } = require('child_process');

// const pythonProcess = spawn('python3', ['embedText.py']);
// pythonProcess.on('close', (code) => {
//   if (code === 0) {
//     console.log('Python Script Execution Successful');
//   } else {
//     console.error(`Python Script Execution Failed with Code ${code}`);
//   }
// });

export default (req, res) => {

  if (req.method === 'GET') {
    const question = req.query.question || ''; // Default to 0 if parameter is not provided

    // Execute the Python script with the provided parameter
    exec(`python3 script.py ${question}`, (error, stdout, stderr) => {
      if (error) {
        console.error(`Error executing Python script: ${error}`);
        res.status(500).json({ result: 'Internal Server Error' });
```

Step 1.3.2

Then, we follow these steps:

````
Make a copy of the example environment variables file:

```bash
$ cp .env.example .env
```

Add your [API key](https://beta.openai.com/account/api-keys) to the newly created `.env` file.
````

Step 1.3.2

Then, we install required packages with ‘npm run install’

```
PS D:\VS CODE\Python\CS589\CusService_Nodejs> npm run install

> openai-quickstart-node@0.1.0 install
> pip install -r requirements.txt

Collecting autopep8==1.6.0 (from -r requirements.txt (line 1))
  Downloading autopep8-1.6.0-py2.py3-none-any.whl (45 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 45.3/45.3 kB 1.1 MB/s eta 0:00:00
Collecting click==8.0.3 (from -r requirements.txt (line 2))
  Downloading click-8.0.3-py3-none-any.whl (97 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 97.5/97.5 kB 2.7 MB/s eta 0:00:00
Collecting et-xmlfile==1.1.0 (from -r requirements.txt (line 3))
  Downloading et_xmlfile-1.1.0-py3-none-any.whl (4.7 kB)
Collecting Flask==2.0.2 (from -r requirements.txt (line 4))
  Downloading Flask-2.0.2-py3-none-any.whl (95 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 95.2/95.2 kB 2.7 MB/s eta 0:00:00
Collecting itsdangerous==2.0.1 (from -r requirements.txt (line 5))
  Downloading itsdangerous-2.0.1-py3-none-any.whl (18 kB)
Collecting Jinja2==3.0.2 (from -r requirements.txt (line 6))
  Downloading Jinja2-3.0.2-py3-none-any.whl (133 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 133.8/133.8 kB 4.0 MB/s eta 0:00:00
Collecting MarkupSafe==2.0.1 (from -r requirements.txt (line 7))
  Downloading MarkupSafe-2.0.1-cp38-cp38-win_amd64.whl (14 kB)
Collecting openpyxl==3.0.9 (from -r requirements.txt (line 8))
  Downloading openpyxl-3.0.9-py2.py3-none-any.whl (242 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 242.2/242.2 kB 3.7 MB/s eta 0:00:00
Collecting pandas-stubs==1.2.0.35 (from -r requirements.txt (line 9))
  Downloading pandas_stubs-1.2.0.35-py3-none-any.whl (159 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 159.3/159.3 kB 4.8 MB/s eta 0:00:00
```

Step 1.3.2

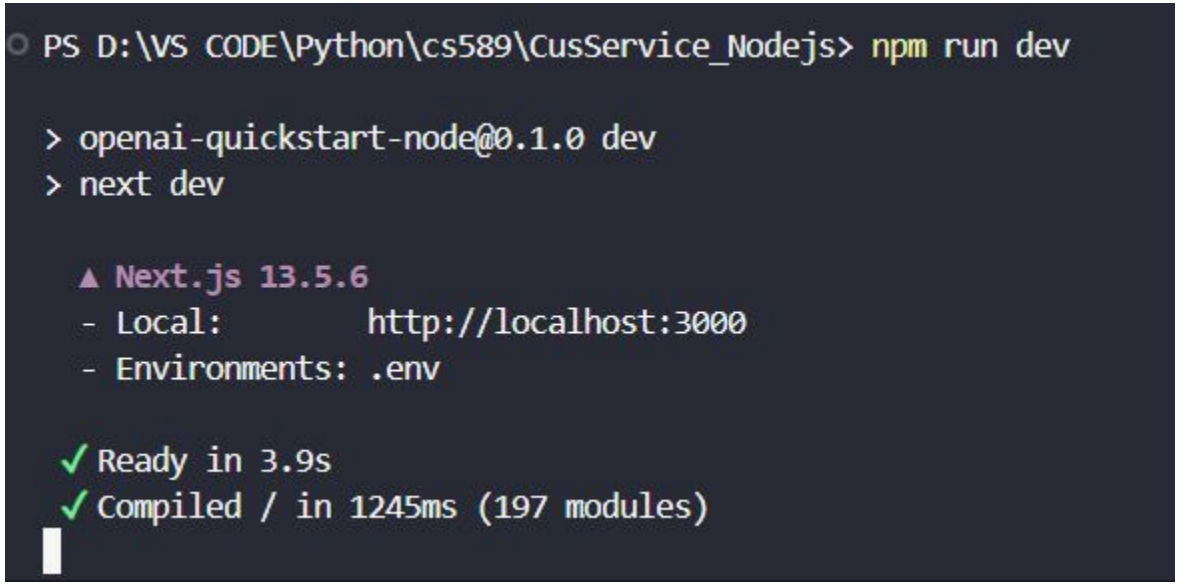
Then, run ‘npm run crawl’ to run ‘crawldata.py’ and use ‘npm run embedding’ to run ‘embeddata.py’.

```
PS D:\VS CODE\Python\cs589\CusService_Nodejs> npm run embedding

> openai-quickstart-node@0.1.0 embedding
> python3 embeddata.py
```

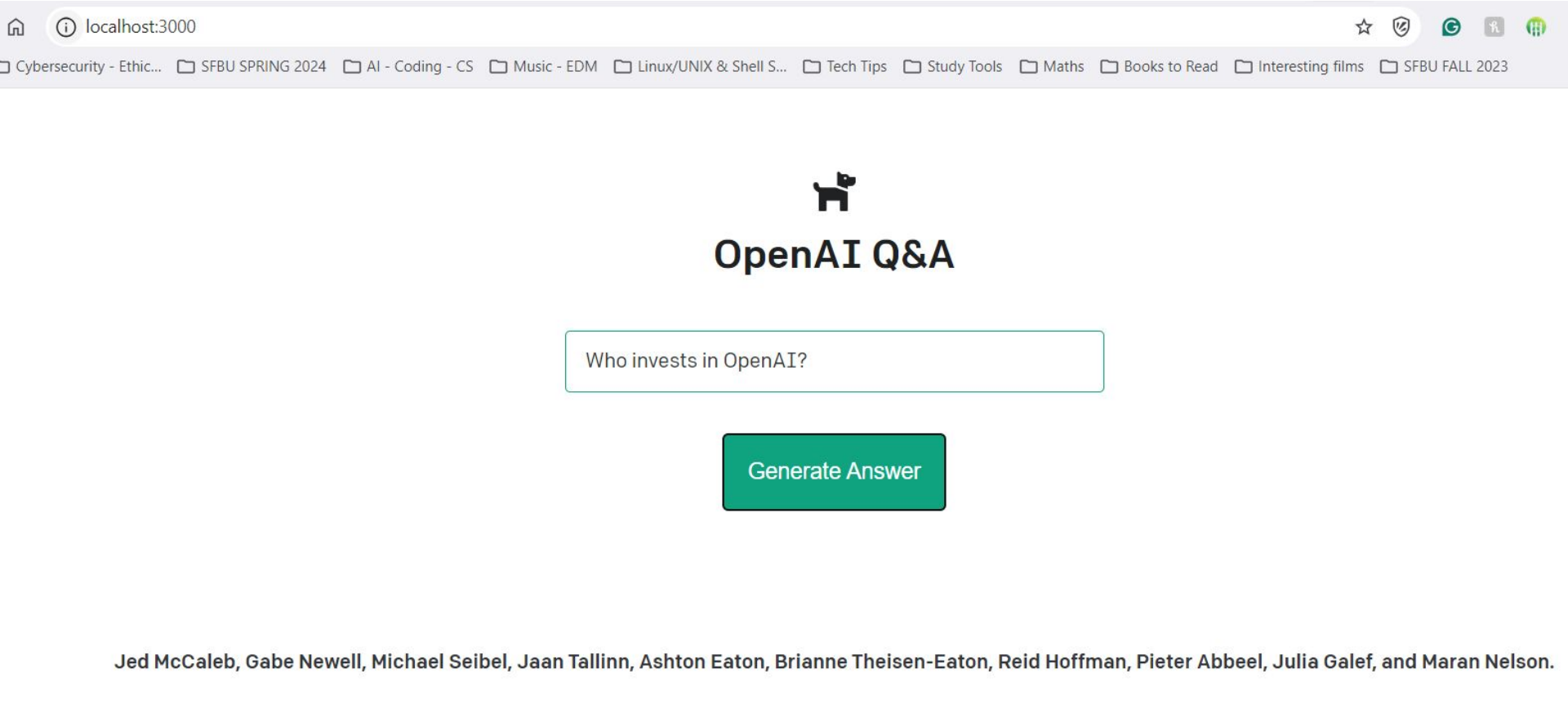
```
PS D:\VS CODE\Python\cs589\CusService_Nodejs> npm run crawl

> openai-quickstart-node@0.1.0 crawl
> python3 crawldata.py

https://openai.com/
https://openai.com/research/dall-e-3-system-card
https://openai.com/research?models=dall-e-3
Unable to parse page https://openai.com/research?models=dall-e-3
https://openai.com/research/forecasting-misuse
https://openai.com/research/forecasting-misuse#content
https://openai.com/research/dall-e-2-pre-training-mitigations
https://openai.com/dall-e-2
https://openai.com/blog/dall-e-introducing-outpainting
https://openai.com/blog?authors=openai
Unable to parse page https://openai.com/blog?authors=openai
https://openai.com/blog/data-partnerships
https://openai.com/customer-stories/government-of-iceland
https://openai.com/customer-stories/government-of-iceland#content
https://openai.com/customer-stories?topics=language
Unable to parse page https://openai.com/customer-stories?topics=langua
ge
https://openai.com/customer-stories/digital-green
https://openai.com/customer-stories/digital-green#content
https://openai.com/customer-stories/be-my-eyes
https://openai.com/customer-stories/be-my-eyes#content
https://openai.com/customer-stories/morgan-stanley
https://openai.com/customer-stories/morgan-stanley#content
https://openai.com/customer-stories#content
https://openai.com/customer-stories/ironclad
https://openai.com/customer-stories/ironclad#content
https://openai.com/customer-stories/summer-health
https://openai.com/customer-stories/summer-health#content
https://openai.com/customer-stories/waymark
```

Step 1.3.2

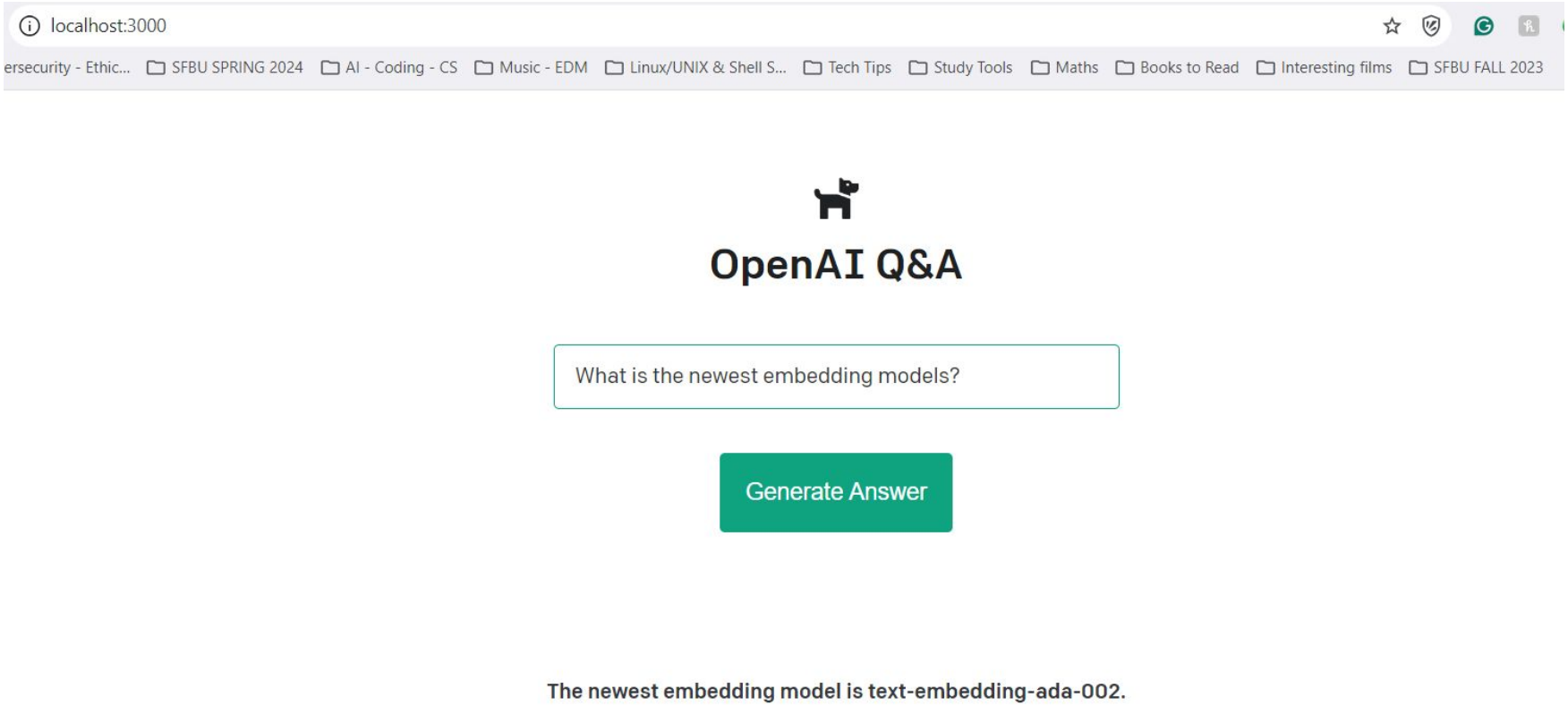
Then, run ‘npm dev run’ to run the program.



Step 1.3.2


localhost:3000
Cybersecurity - Ethic...
SFBU SPRING 2024
AI - Coding - CS
Music - EDM
Linux/UNIX & Shell S...
Tech Tips
Study Tools
Maths
Books to Read
Interesting films
SFBU FALL 2023
OpenAI Q&A
Who invests in OpenAI?
Generate Answer
Jed McCaleb, Gabe Newell, Michael Seibel, Jaan Tallinn, Ashton Eaton, Brianne Theisen-Eaton, Reid Hoffman, Pieter Abbeel, Julia Galef, and Maran Nelson.

Step 1.3.2


localhost:3000
ersecurity - Ethic...
SFBU SPRING 2024
AI - Coding - CS
Music - EDM
Linux/UNIX & Shell S...
Tech Tips
Study Tools
Maths
Books to Read
Interesting films
SFBU FALL 2023
OpenAI Q&A
What is the newest embedding models?
Generate Answer
The newest embedding model is text-embedding-ada-002.

Source code & reference:

Error codes - OpenAI API

Production best practices - OpenAI API

Embeddings - OpenAI API

GitHub from OpenAI: Node.js example app from the OpenAI API quickstart tutorial

GitHub: https://github.com/MynameisKoi/CS589/tree/main/CusService_Nodejs