Quang Khai Le

EE 104

Lab 8

Youtube link:

OpenAI File Q&A: <https://youtu.be/x3KK3H7dx_0>

OpenAI Web Crawl Q&A: <https://youtu.be/JTK6sVOBaN8>

Simple Traffic Controller Part 2, Hardware: <https://youtu.be/NNaq_yKlJrs>

Github link: <https://github.com/blade199916/Lab-8>

Introduction:

You will find all the instructions you need in this readme.txt file to run my apps, program and prompt correctly to try two methods of OpenAI chatbot with the provided Q&A directory files. In addition, the traffic controller circuit second part is enhanced with the state machine logic to follow compared to the previous lab

Requirements:

Before starting the lab, make sure you have the Python Spider package ready up to version 3.6 or later already installed. You will adhere to the directions in these files, which you can obtain from Canvas, for the OpenAI portion:

1. openai-quickstart.txt

2. FileQ&A\_Instructions.txt

3. WebCrawlQ&A.txt

4. You should also download the web-qa.py example Python source code.

You will write your own software to use with the KRIA interface.

Part 1: OpenAI File Q&A

You can upload up to 75 text files or Word documents using this program's ability to build a local host. After being uploaded, the application will take the data it needs from these files and utilize it to provide answers. This application will provide you the skills to swiftly access and use the information in your files, whether you're trying to learn more about the subject matter or get a greater understanding of it. With this update, the program's advantages will be emphasized and made clearer.

b. Running the program

#Step 1:Set up OpenAI API and Pinecone API keys and index name. Retrieve from these sites:

https://platform.openai.com/docs/api-reference/introduction

https://www.pinecone.io/

For Index, select "FROM COLLECTION" and pick NEW

index\_name="ee104"

Dimensions=1536, Metric=Cosine, Pod Type= S1

then click the button "CREATE INDEX"

#Step 2:Run the Google Colab from the first Jupiter cell to the second-last one.

You will see a pop-up window with the title "Creating index openai-trec"

click on "OPEN COLLAB" in another tab or another window.

It will look like this: https://colab.research.google.com/github/pinecone-io/examples/blob/master/integrations/openai/semantic\_search\_openai.ipynb

#Step 3: Download the source code from https://github.com/openai/openai-cookbook.

You will need Node,js and nam for the Next.js client.

Run this command:

pip install openai

npm install openai

DOWNLOAD THE SOURCE CODE

# Go to: https://github.com/openai/openai-cookbook

# Create a ".env" file using NotePad++ or any text editor that allow you to save

# in this format .env

# and type this line below, then save the file as "all type" with file name ".env"

# env stands for environment

# OPEN\_API\_KEY = "YOUR\_API\_KEY"

# YOUR OPENAI ORG KEY = "YOUR\_OPENAI\_ORG\_KEY"

# PINECONE\_API\_KEY="YOUR\_PINECONE\_API\_KEY"

# PINECONE\_ENVIRONMENT="Your\_environment"

Step 4 run into Power Shell as administrator by two different folder at the same time, one for Server, and one for Client

#Step 4: Run in Server directory

1. Change directory to your Server directory.

2. Fill out the config.yaml file with your Pinecone API key, index name and environment.

3. In your PowerShell window, run this command:

pip install openai

npm install openai

Then run following commands one by one

python install virtualenv

python -m venv venv

.\venv\Scripts\activate

pip install -r .\requirements.txt

Finally run, python app.py

# You should now see the followings:

\* Serving Flask app 'app'

\* Debug mode: on

INFO:werkzeug:←[31m←[1mWARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.←[0m

\* Running on http://127.0.0.1:8080

INFO:werkzeug:←[33mPress CTRL+C to quit←[0m

INFO:werkzeug: \* Restarting with stat

WARNING:werkzeug: \* Debugger is active!

INFO:werkzeug: \* Debugger PIN: 576-986-598

\_ Run on Client directory

1. In Powershell, change directory to Client

pip install openai

npm install openai

2. Type the command to run the Next.js client:

Npm run dev

You should see something similar to the followings:

> file-q-and-a@0.1.0 dev

> next dev

ready - started server on 0.0.0.0:3000, url: http://localhost:3000

event - compiled client and server successfully in 2.7s (166 modules)

wait - compiling...

event - compiled successfully in 124 ms (133 modules)

Attention: Next.js now collects completely anonymous telemetry regarding usage.

This information is used to shape Next.js' roadmap and prioritize features.

You can learn more, including how to opt-out if you'd not like to participate in this anonymous program, by visiting the following URL:

https://nextjs.org/telemetry

wait - compiling / (client and server)...

event - compiled client and server successfully in 1962 ms (757 modules)

#Step 6: Open http://localhost:3000 with your browser to see the app.

Upload your Q and A file to the site

Search for a content within that file.

Part 2: OpenAI Web Crawl Q&A

This application is intended to help you create a question-and-answer tool regarding San Jose State University. It will get information from the SJSU.edu website and respond to any questions you may have about it.

Step 1: Running the software Create an API key using this website's instructions: https://platform.openai.com/docs/api-reference/introduction

Visit https://platform.openai.com/account/api-keys to obtain an API key.

Run this command on Powershell, pip install openai

# Create a folder. For this project, let's call it C:\web-crawl-q-and-a

# using the Linux command "md" (make directory)

PS C:\>md web-crawl-q-and-a

# Use the "cd" (change directory) to go inside the directory:

PS C:\> cd .\web-crawl-q-and-a\

PS C:\web-crawl-q-and-a>

# Create a ".env" file using NotePad++ or any text editor that allow you to save

# in this format .env

# and type this line below, then save the file as "all type" with file name ".env"

# env stands for environment

OPEN\_API\_KEY = "YOUR\_API\_KEY"

#Step2: DOWNLOAD THE CODE:

Clone the full code for this tutorial on GitHub and put it in a local directory

on the computer. For example: put the files in C:\web-crawl-q-and-a

https://github.com/openai/openai-cookbook/tree/main/apps/web-crawl-q-and-a

#Step 3: PREPARE THE ENVIRONMENT:

PS C:\> cd .\web-crawl-q-and-a\

PS C:\web-crawl-q-and-a> python install virtualenv (or c:\python310\Python install virtualenv if you have to use the absolute path)

PS C:\web-crawl-q-and-a> python -m venv env

PS C:\web-crawl-q-and-a> ls

Directory: C:\web-crawl-q-and-a

Mode LastWriteTime Length Name

---- ------------- ------ ----

d----- 3/3/2023 8:28 AM env

-a---- 3/2/2023 5:56 PM 1342 requirements.txt

-a---- 3/2/2023 5:56 PM 81844 web-qa.ipynb

-a---- 3/2/2023 5:56 PM 13302 web-qa.py

#Notice that the new directory "env" has been created.

PS C:\web-crawl-q-and-a> ls env

Directory: C:\web-crawl-q-and-a\venv

Mode LastWriteTime Length Name

---- ------------- ------ ----

d----- 3/3/2023 8:26 AM Include

d----- 3/3/2023 8:26 AM Lib

d----- 3/3/2023 8:26 AM Scripts

-a---- 3/3/2023 8:26 AM 77 pyvenv.cfg

PS C:\web-crawl-q-and-a> .\env\Scripts\activate

(env) PS C:\web-crawl-q-and-a>

Go to powershell window and run,

PS C:\OpenAI\web-crawl-q-and-a> Get-ExecutionPolicy -List

PS C:\OpenAI\web-crawl-q-and-a> Set-ExecutionPolicy -ExecutionPolicy Unrestricted -Scope CurrentUser

Execution Policy Change

The execution policy helps protect you from scripts that you do not trust. Changing the execution policy might expose

you to the security risks described in the about\_Execution\_Policies help topic at

https:/go.microsoft.com/fwlink/?LinkID=135170. Do you want to change the execution policy?

[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is "N"): Y

PS C:\OpenAI\web-crawl-q-and-a> Get-ExecutionPolicy -List

Scope ExecutionPolicy

----- ---------------

MachinePolicy Undefined

UserPolicy Undefined

Process Undefined

CurrentUser Unrestricted

LocalMachine Unrestricted

PS C:\OpenAI\web-crawl-q-and-a> .\env\Scripts\activate

#### You can learn more about Powershell policy here:

#### https://learn.microsoft.com/en-us/powershell/module/microsoft.powershell.core/about/about\_execution\_policies?view=powershell-7.3#use-group-policy-to-manage-execution-policy

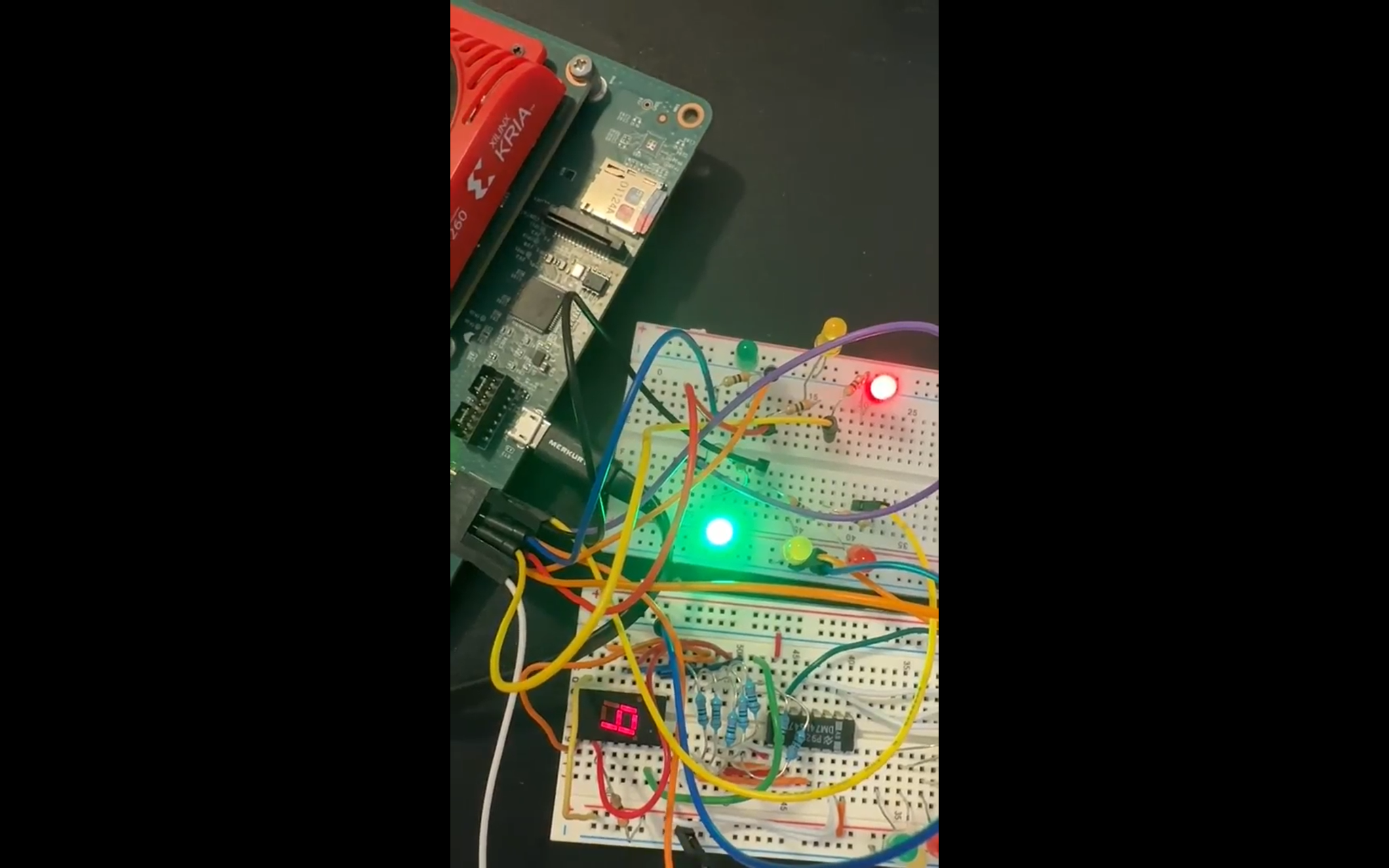
#The command below will take some time to finish all installations

(env) PS C:\web-crawl-q-and-a> pip install -r requirements.txt

Part 3: Simple Traffic Controller Part 2, Hardware

Simple Traffic Controller: In Part 2, you will construct a software Finite State Machine to replace the Finite State Machine you used in Part 1 (which was done with Flip Flops).

Since East Santa Clara Street is given priority, it always has a green signal and pedestrian walkways are permitted, barring requests for pedestrian crossings from 7th Street or vehicles entering from 7th Street.



To develop your hardware controller component, you will get the following hardware:

For traffic signals, see LED.

7-Segment for the 7th Street pedestrian countdown signal

Push-button to take input from pedestrian

7-Segment to 74LS47 BCD For use as a clock, decoder 74LS193 Binary Up/Down Counter with Clear 74LS90 Decade Counter

Resistors LED, 7-Segment, and other open-collector signal connections

Capacitors

You'll provide your own:

Breadboard Your prior courses' breadboard will be used again.

Following the logic of the state machine, we applied it into the 7 segment with a counter and LEDs circuit to simulate the traffic light controller.

1st step: Follow the code in the Jupiter to set and write the pin number corresponding to the logical design.

2nd step: Construct the circuit like the previous lab including 6 LEDs divided into two symmetry lights simulating the traffic light.

3rd step: If everything is right and is correct, when you click each step of the code, the LEDs change simultaneously as described in the videos. The delay is unavoidable for both LEDs lit up sequence plus the 7 segment with counter. In addition, the 7 segment with counter should change corresponding to the changing of the LEDs as a result.

So far, that is it for my individual and group project for lab 8. I hope you can follow and get it work your own