Basic Web Dev

1. Install programs through terminal or online
    a. Python
    b. Pip
    c. Virtualenv
    d. Ngrok
2. Create requirements.txt for your virtualenv which has to be reloaded each time the virtualenv is reactivated. The requirements folder holds programs or libraries that the virtualenv will use, for our purposes the requirements.txt will hold. (Just type in the three lines below into a text file)
    a. Flask==0.10.1
    b. Twilio
    c. requests
3. To activate virtual environment
    a. Create a virtualenv file with terminal command: virtualenv twilioenv(**twilioenv can be any name, this is something you will need to remember though so make it relevant.**)
    b. Activate virtual env file with terminal command: source twilioenv/bin/activate
    c. Twilioenv should now show up on the left side of your terminal line in paranthesis
    d. To deactivate later, just type in deactivate
4. Now use your requirements.txt file to load the libraries you want
    a. terminal command: pip install -r requirements.txt
5. Create a simple program and run on terminal: python cats.py

    a.
6. Open a new tab in terminal to activate ngrok
    a. Find what local port your program is running on for mine it was 5000
    b. Terminal command: ./ngrok 5000
    c. This makes it so that the local host can be accessed by anyone on any computer
    d. Note: each time you change your code, you need to reload the local host

Resources:

- http://flask.pocoo.org/docs/0.10/quickstart/#quickstart
- https://github.com/Giphy/GiphyAPI/blob/master/README.md

- https://www.twilio.com/docs/api/rest/sending-messages
- http://rest.elkstein.org/2008/02/more-complex-rest-requests.html

Part 2

Make all results from the data array appear on the webpage. This can be done by looping through the array and adding each element to a single string and finally returning the whole string.
*Note:
 is an html code that allows a new line to be created.

Part 3

Make the search results dynamic and not static. This finally uses the query variable written in the code, this allows the user to search up particular subjects for gifs by adding ?Body=<something> after the /cats path. To code this simply add in the query variable in the sample url.

To create a unique domain

ngrok usually creates a random string as your server URL for example: http://7347d96a.ngrok.com

To create your own unique url that ngrok will forward to each time, type this into the console **./ngrok -subdomain Kevin 5000**

where Kevin is just what you name it to be. You might need to sign up for this feature to work, but its purpose is for convenience only.

Part 4

Now we will actually implement to Twilio API into the code first import a twilio library by typing into your code “from twilio.rest import TwilioRestClient”. Then sign up for a free twilio account and they will give you a custom number. After receiving your number, click on the numbers tab at the top

of the webpage and click on the number they gave you. Scroll all the way to the bottom where there is a tab for Messaging. In the request URL, type in your ngrok URL. Now click on accounts and find your account_sid and auth_token, we will be using these later on.

NOTE*: your account_sid and auth_token are unique to your account and is what allows you to send requests to the server, if others know these numbers and you have a paid account, they can continuously charge your server.

Now go to Twilio Docs and you will see this section

This is the code we will be adding into our program. Twilio has programmed their libraries so that is has become relatively simple to get and request things from their server. Your type in your account_sid and auth_token as strings in those variables. The body variable within messages is the URL data you want to send and from our code it would be URLlist[0]. **You're getting data from your Twilio number so input that into there and the to is the server (Not sure about this yet have to reconfirm)**. The media_url will not work unless you have a paid Twilio account and if you include it in your code, an error will occur.

Normally the apps.route function does not allow any new methods except for its default one. So in your @apps.route function include this into the code

@app.route("/cats", methods = ["GET", "POST"])

This allows the get and post method in your code.If this was done correctly, you should now be able to text your twilio number with some word and it will reply with a URL to a gif about the word you sent.

Side Note

Usually you do not want to include your account_sid and your auth_token in your actual code especially if you are posting the code to a public git account. However, there is a way to allow you to store these variables within you OS so that you can call upon them in your code when you want to use

them. To do so you need to type in “import os” into your code so that it can use the OS libraries. Now create a new file and name it any file but the extension has to be “.sh”, I named mine SetVariable.sh, this means that this is bash code (not sure what that is). In your new file type these in

and type in your account_sid after Twiliosid and auth_token after Twiliotoken.

Now in your terminal type in ./{file name} and you will see that your computer has denied your permission, usually typing in sudo before it helps clear that permission but it is important to know that the reason why permission was denied was because bash code can be malicious and do things to your operating system. **The “chmod” command allows new code to be used to change the os, so type this command into your terminal to give your file access to your computer OS**

chmod +x {file name}

and then retype

./{file name}

(need confirmation on this way of doing it)

You can also type in (**This needs to be done each time you activate the virtual env**)

source {file name}

to store the variables and then

printenv

to see whether or not your variables have been stored, they should be listed with the variable name you gave them in your file.

Now to include these into your code

```
app=Flask(__name__)

@app.route("/cats", methods = ["GET", "POST"]) #allows the ge
def cats():
    account_sid = os.environ["Twiliosid"]
    auth_token = os.environ["Twiliotoken"]
    query = request.values.get("Body", None) #None is to chec
```

where Twiliosid and Twiliotoken are the variables I assigned to them in the bash file.

Completed code for mobile Twilio app to receive gifs