**Lessons learned from building this project**

1.

Text

Description automatically generated

* (R)!!!! Note how the **if \_\_name\_\_ == “\_\_main\_\_”** is used!!

When I run this script in m terminal, by doing python codealong.py, it will run this script, but it will start from whatever you put within **if \_\_name\_\_ == “\_\_main\_\_”,** and in this case we call on the main() function, and so it runs main()!

2. What ‘echo’ means

* In a lot of the installations or when running commands from the internet, very often they use ‘echo’, like this for example:

echo 'eval "$(/opt/homebrew/bin/brew shellenv)"' >> /

PTO

**(R)!!! In computing, ‘echo’ is a command that outputs the strings that are passed to it as arguments.**

And this makes so much sense, because if you look at the example above, and thinking about other examples where I’ve seen echo, it is used to pass a string as argument, because without echo, you are basically writing a string into say your terminal, and you WILL OF COURSE NOT get the desired output!

3. Xcode is the tool developers use to build apps for the Apple ecosystem- MacOS, iOS, and all thigns Apple.

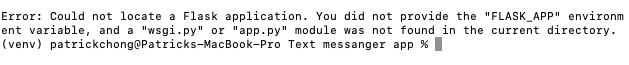
4. I tried ignoring Miguel’s set up for the microblog and build a similar flask app without things like the app folder and without doing FLASK\_APP = …

But when I write a Python script like this for example:

Text

Description automatically generated

* And in particular, when we have line 3, which is **app = Flask(\_\_name\_\_)**
* If you try to simply do python3 + name\_of\_python\_file
* You get the following error:



* So you MUST set an environment variable (something like **FLASK\_APP = microblog.py**), and we also need to have a **app.py module** in the CWD.

**PTO**

(R)!!!!!

An alternative to doing the above, is to set

**if \_\_name\_\_ == “\_\_main\_\_” :**

**app.run(debug=True)**

Then you won’t need the app folder, and can simply run the app.py file directly. You don’t even have to call it app.py it can be whatever.py, but then of course, you need to change the code directly above this to whatever.run(debug=True) as well.

**$PATH issue**

4.5 Before running the above, I accidentally ran FLASK\_APP=start.py in my terminal, but it is supposed to be “export FLASK\_APP=start.py”, and my **environment variable** **“$PATH” got messed up** somehow- it could be that I did something else as well.

But the **issue now is that none of the commands are working**, like ‘pip’ or ‘flask’, etc. It is not even working in my global directory!

Meaning that when I run “pip list” or just “pip” or when I run “flask”, I get the error:

**Zsh: command not found: pip or flask**

When looking up this problem, people say that it is because $PATH has been set to something wrong, and I need to fix this.

(R)!!!

$PATH is **a environment variable that is file location-related**. When one types a command to run, the system looks for it in the directories specified by PATH in the order specified. You can view the directories specified by typing echo $PATH in the terminal.

Very good source: <https://astrobiomike.github.io/unix/modifying_your_path>

5. (R)!!! V IMP

When you are running your Python script, and when you **render\_template()** to open a HTML file like this:

Text

Description automatically generated

* (R)!!!! when we get to render\_template, Flask looks for a **‘templates’ folder** by default.
* So you’ll want to save the ‘index.html’ file in the templates folder

So the structure of the files and folders are:

Top level directory

|--template/

|--index.html

|--app.py

N.B. I also tried to change the name of ‘app’ to something else, and then in the screenshot above to change the ‘app.run(debug=True)’ to something else but it doesn’t work! – it needs to be called app- because I think Flask looks for ‘app’.

(R)!!!! Worth going over the Flask documentation- <https://flask.palletsprojects.com/en/2.1.x/quickstart/>

Notice in the above we DO NOT need to set “export FLASK\_APP = …” and instead we can directly run “python3 app.py” in our terminal. This is because Flask includes a shortcut, in that if you name your file app.py or wsgi.py, you DO NOT need to set “export FLASK\_APP=…”, however if you name your pytohon file anything else, then you will need to set this **environment variable** in your terminal before executing “flask run”!!!!

With this in mind, it makes perfect sense why you cannot change the name app.py and the app in “app.run(debug=True)” to something else. Because you would have to set an environment variable (export FLASK\_APP = … ).

6. You CANNOT execute a python script on your computer/terminal so that when a user clicks on say a ‘submit’ button on your website it executes this. Because this is very dangerous!

What you need to do is create a server (Flask server for example), so that when the user clicks on the ‘submit’ button, it will make a call to the server, and that will execute the python script that is stored there!

(R)!!!

Everything I have been doing up to this point has been doing exactly this! – i.e. using a server. A server is basically a website. Notice whenever I execute my python code in my terminal, I get a website and type it into the web browser.

This website is the SERVER!!!!

So I need to link my ‘Submit’ button in my html code to this website, which is easy! hehe

7. Another big challenge was to take the input from the html website, because users would go onto the html website, and type in their text there and click the ‘submit’ button, but then in my Python script, I need to be able to get that message to send it to the recipient!!!

So the challenge is how do I take the message once the user has ‘submitted’ it and then include it in my Python script: Flask WTF-forms + SQLAlchemy

In theory one way I can think of doing it is to treat the inputted message on the html website as a ‘form’, and then store it in a database using SQLalchemy. Then I can retrieve it in my python code!