How to Set Up Your Trading Bot

**Intro**

Your trading bot can be run either locally (on your own computer) or in the could (on the Heroku server). In both cases, we needed to install the following on our computer and/or create associated account.

* 1: Anaconda (a Python installation)
* 2: Visual Studio Code (a code editor) w/ Python add-on
* 3: TDA Developer Account
* 4: TDA Token Reset

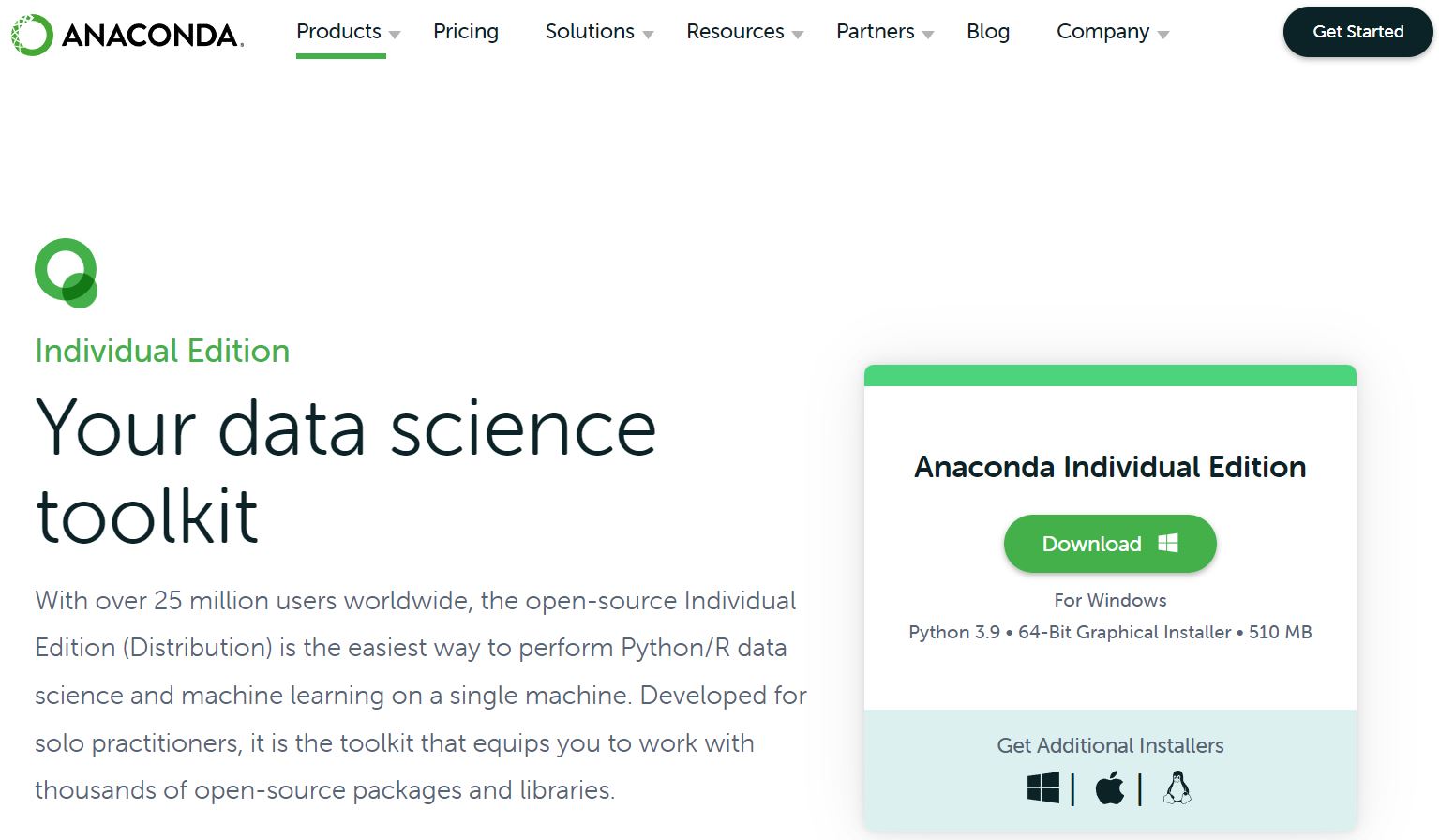
**1: Anaconda**

Why do you need Anaconda?

It installs Python on your computer, as well as a number of needed Python add-ons

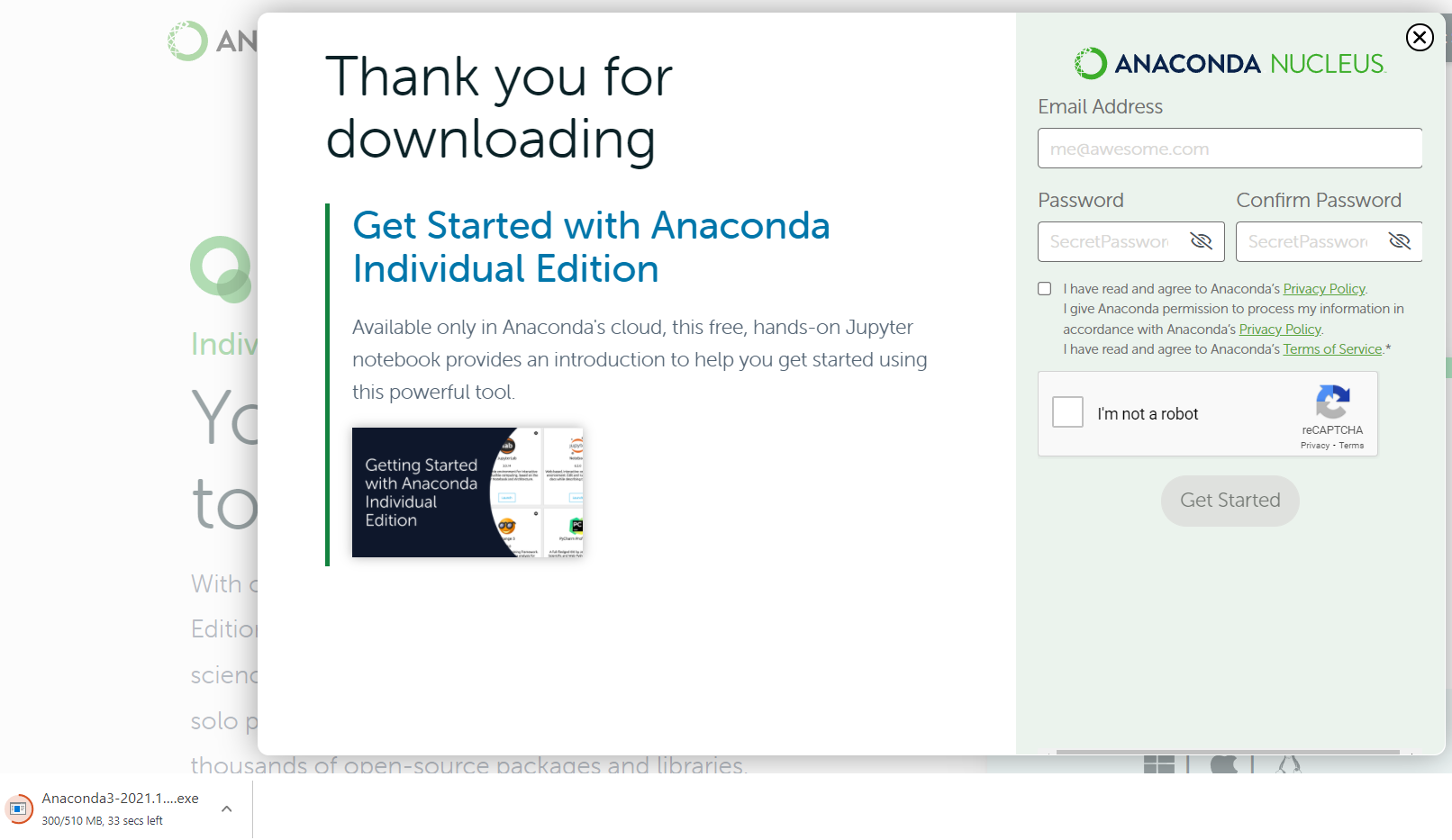
Link: <https://www.anaconda.com/products/individual>

Tutorial: <https://www.youtube.com/watch?v=V6ZAv7hBH6Y>



This link takes you to the Windows installation by default. For Apple or Linux, click on the icons in the bottom right under “Get Additional Installers”. After doing that, click on “Download”

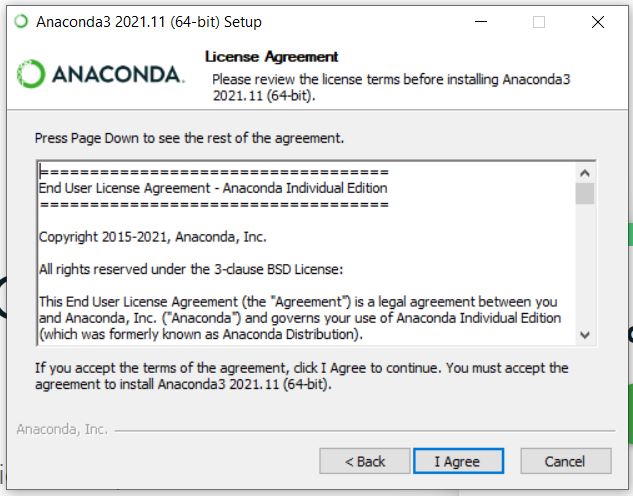
The next window will prompt you to save the installation setup file somewhere. Save it anywhere you would like.



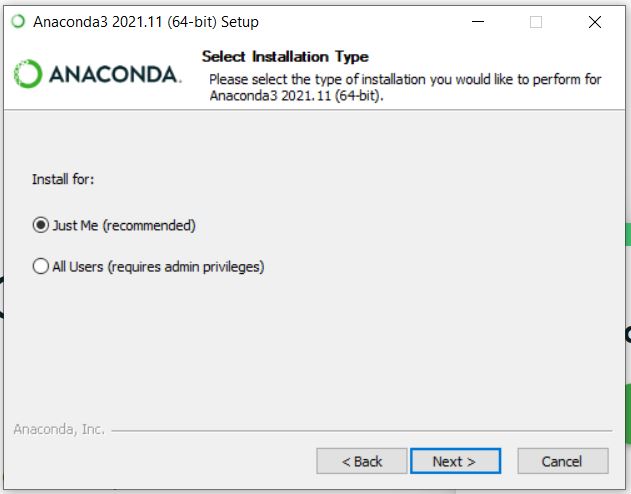
Anaconda would like you to create an account with them, but this is not actually required. Notice on the bottom left corner of this screenshot you can see the Anaconda .exe downloading enough though I haven’t filled out this form. Feel free to click the “X” on the top right corner to close the form. When the .exe finishes downloading, double click on it.



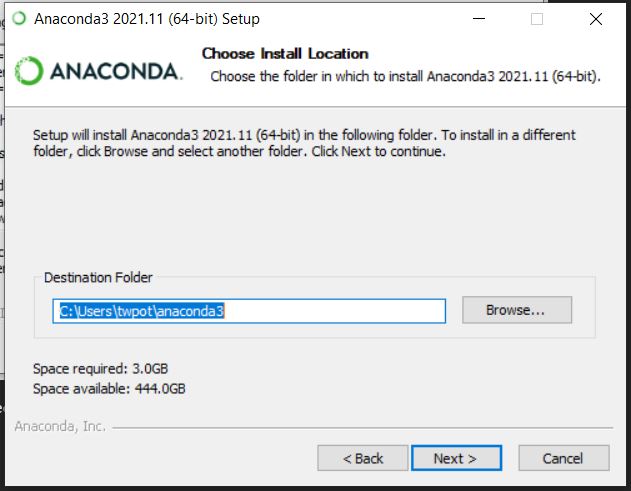
Click “Next >”



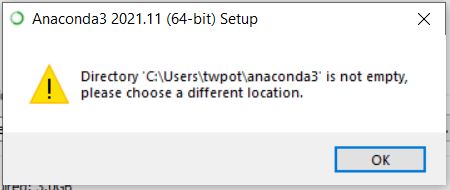
Click “I agree”



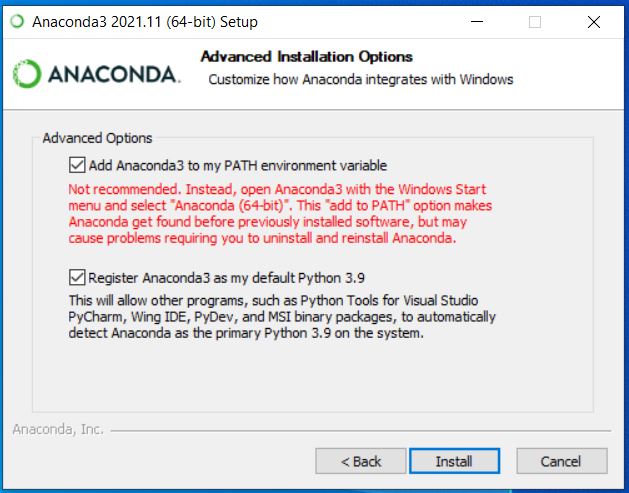
Select “Just me (recommended)” and click “Next >”



Go ahead and use the default destination folder and click “Next >”



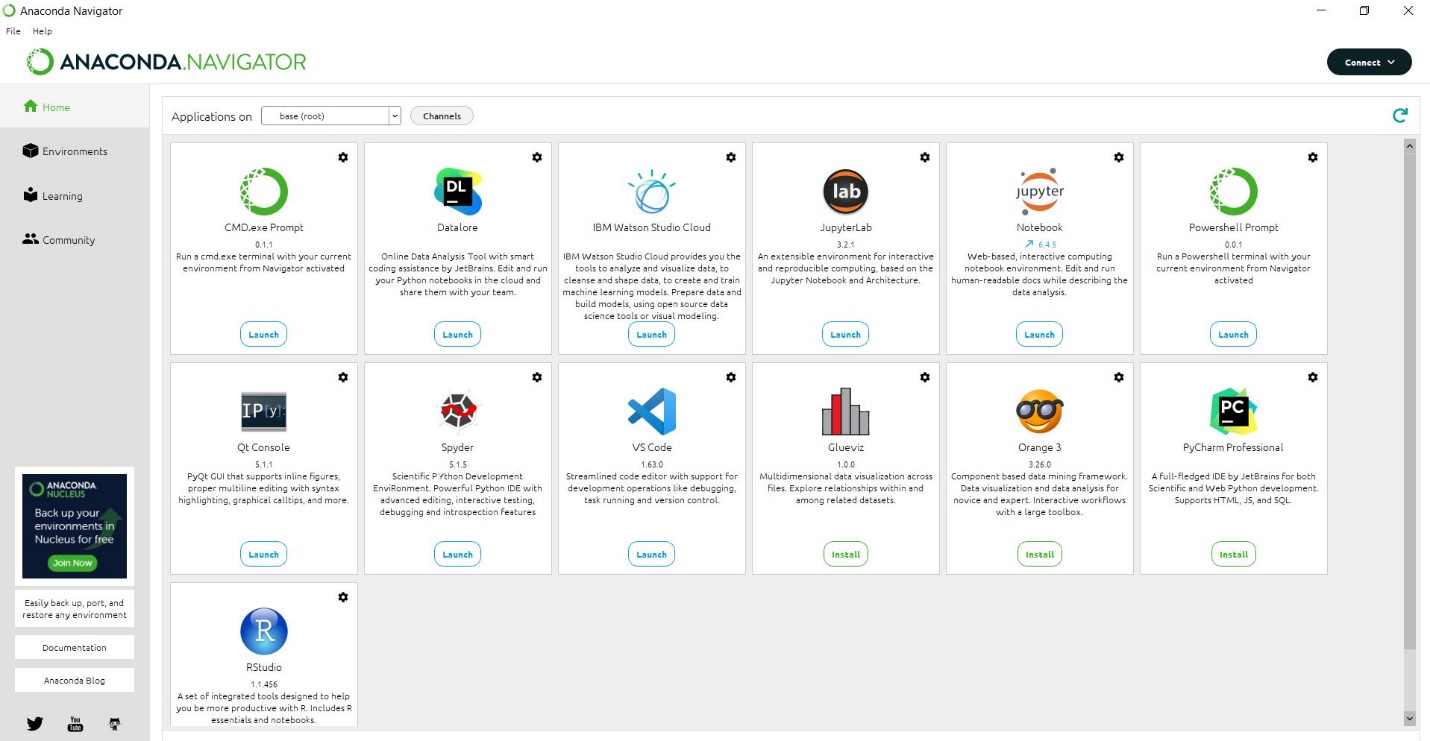
You probably won’t get this error, but if you do, it means you already have Anaconda installed and don’t need to proceed with the rest of the steps in this setup guide. If you didn’t get the error, please continue.



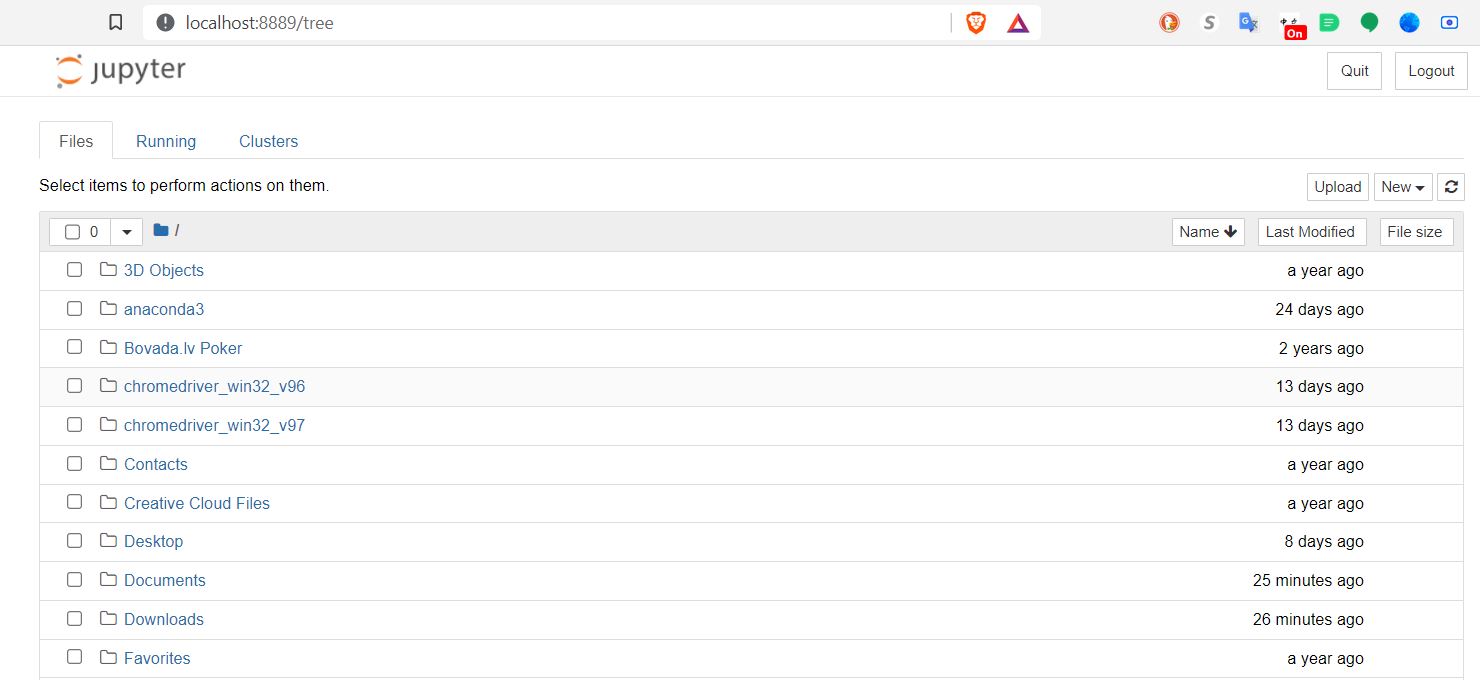
Under “Advanced options”, check both boxes. Yes, I am telling you to check the first box in red that says “Not recommended”. If you don’t check the first box, the set-up will require additional steps. Click “Install”



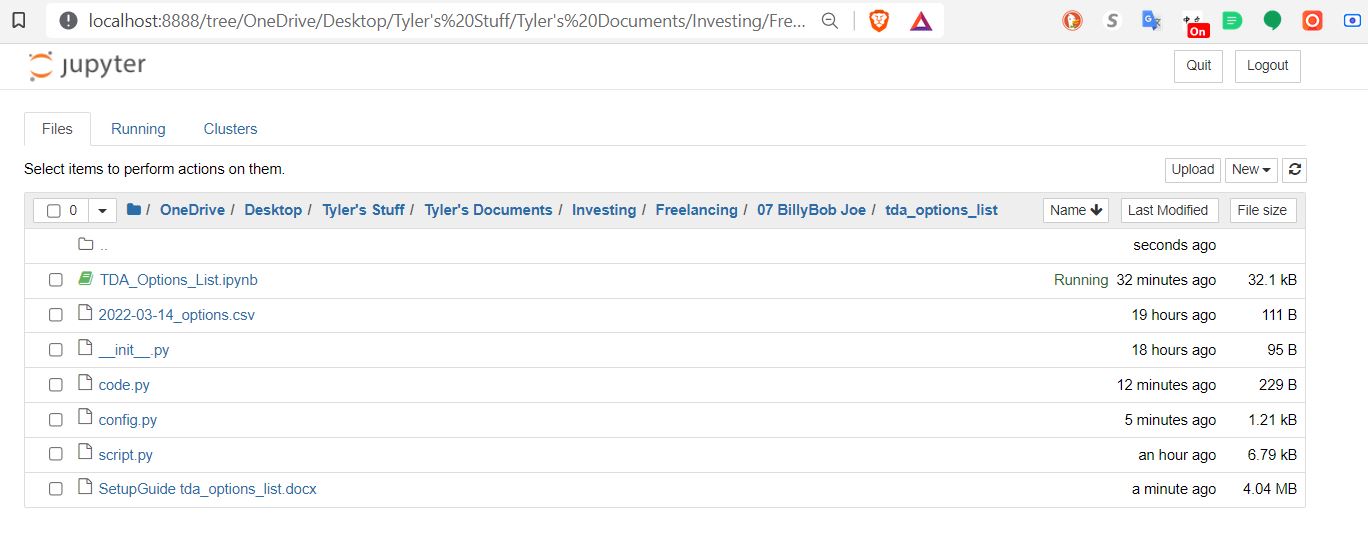
Eventually you will arrive at the above page. Click “Finish” and try to open Anaconda. You should see something similar the below page. You probably won’t see an icon for “VS Code” because we haven’t installed that yet. For now, let’s try to open a .ipynb file. Click “Launch” under “Jupyter Notebook” (top right icon, move 1 left).



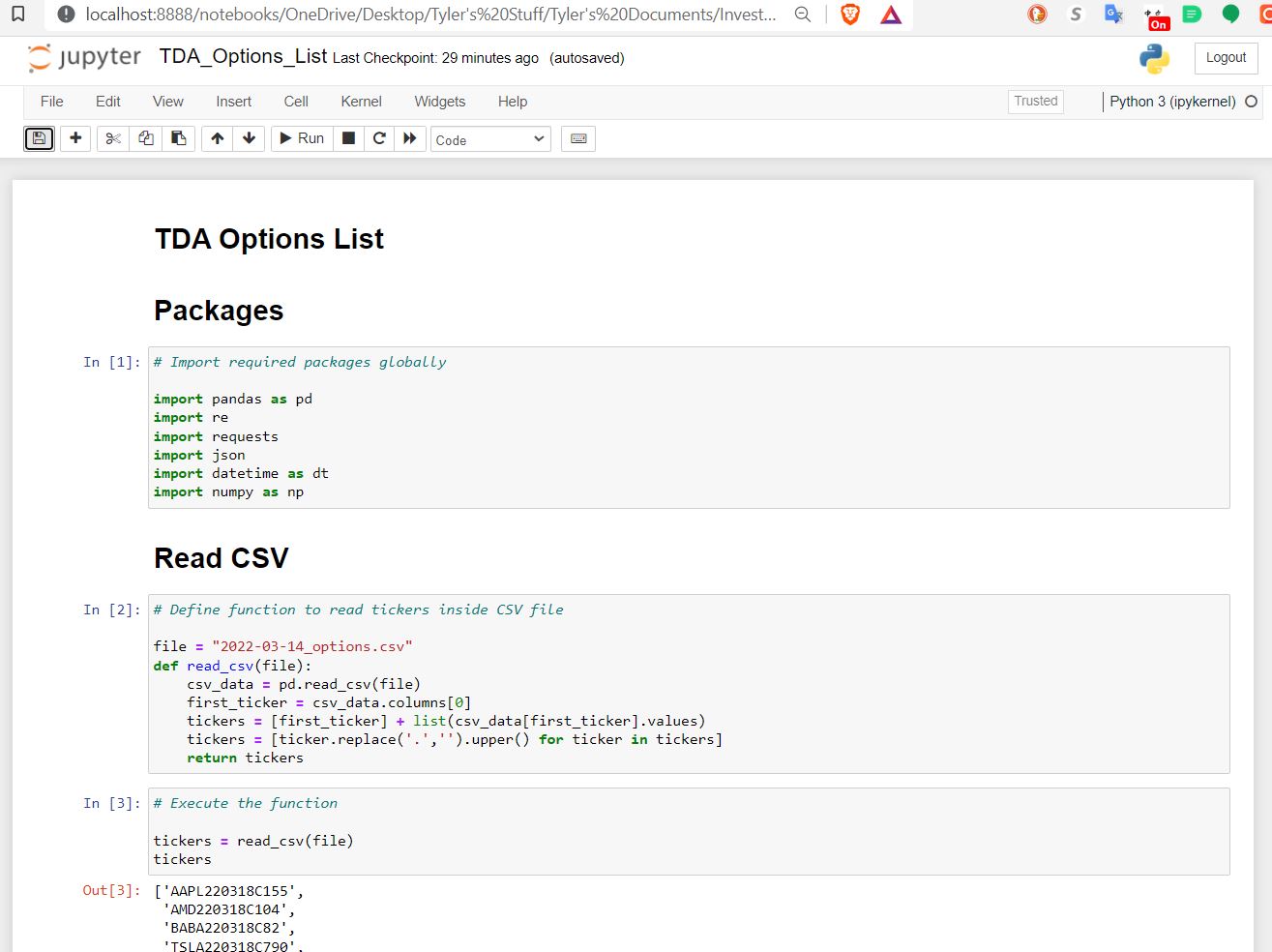
You should end up on a page like this. You will need to navigate through your computer files to find the location of your .ipynb file.



After you find the .ipynb “TDA\_Options\_List”, double click on it to open it



You should land on a page like the one below:

****

One of the main differences between an .ipynb file and .py file is that .ipynb are much more readable than .py files, as they allow for code to be separate into blocks with headings, and can also support visual graphs/charts. I use Jupyter notebooks to develop my scripts and then move the code to Visual Studio Code at the end.

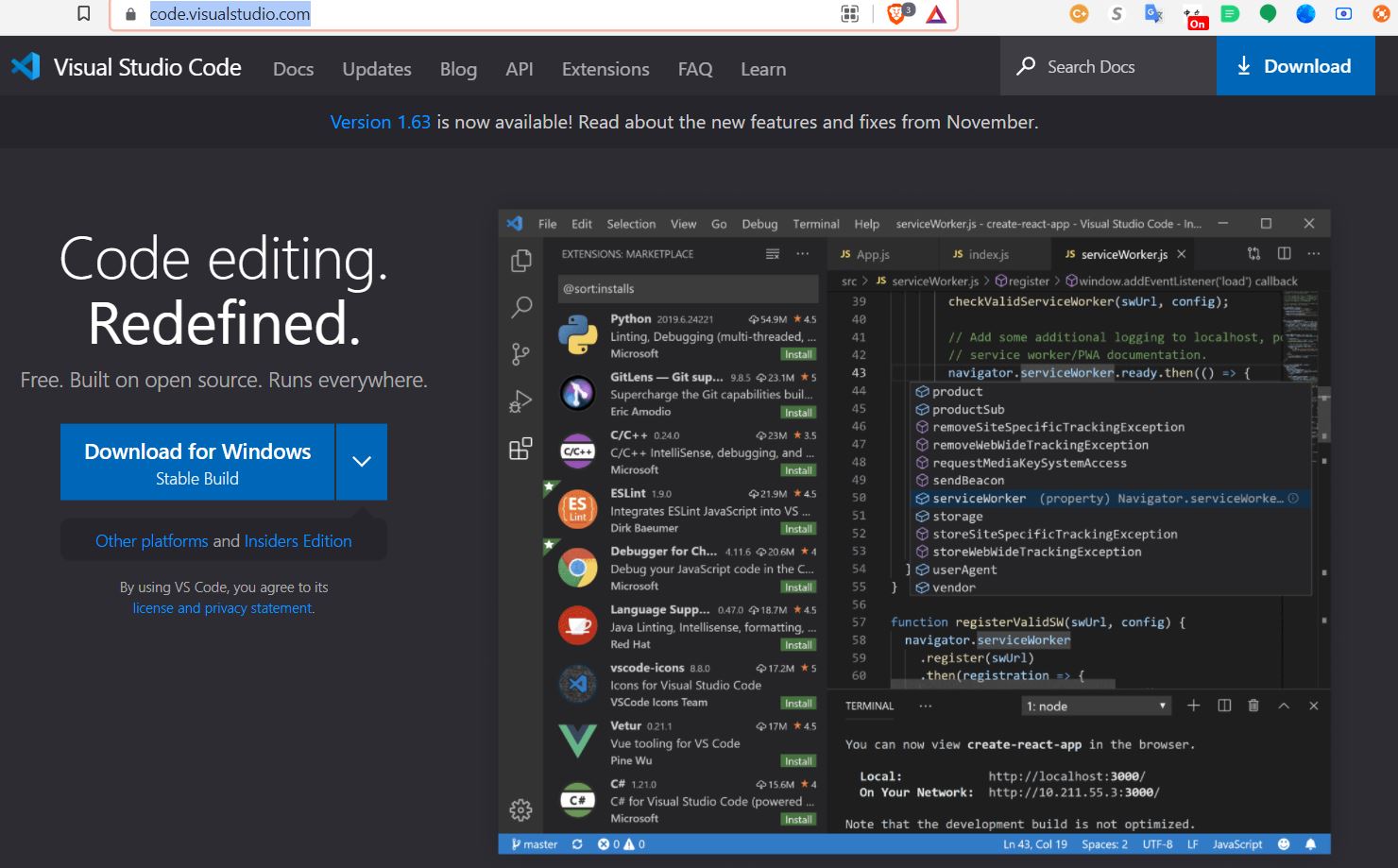
**2: Visual Studio Code**

Why do you need Visual Studio Code?

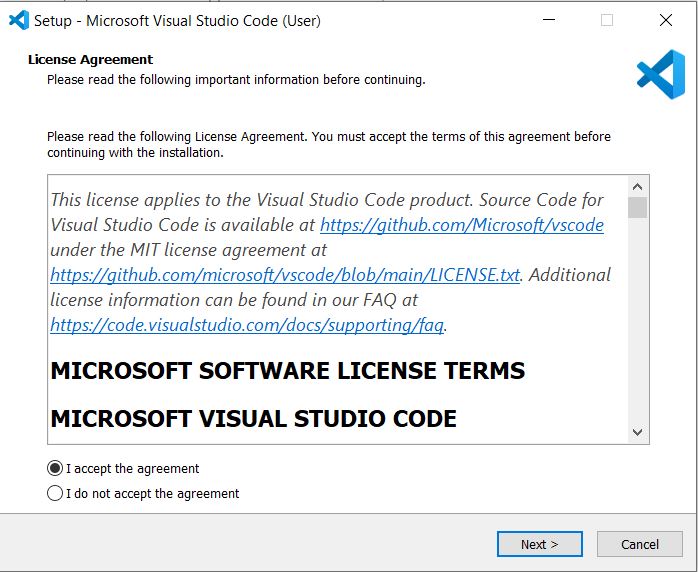
Visual Studio Code is a code editor. This is where you will open up the deliverables I sent to you and launch them into the cloud.

Link: <https://code.visualstudio.com/>

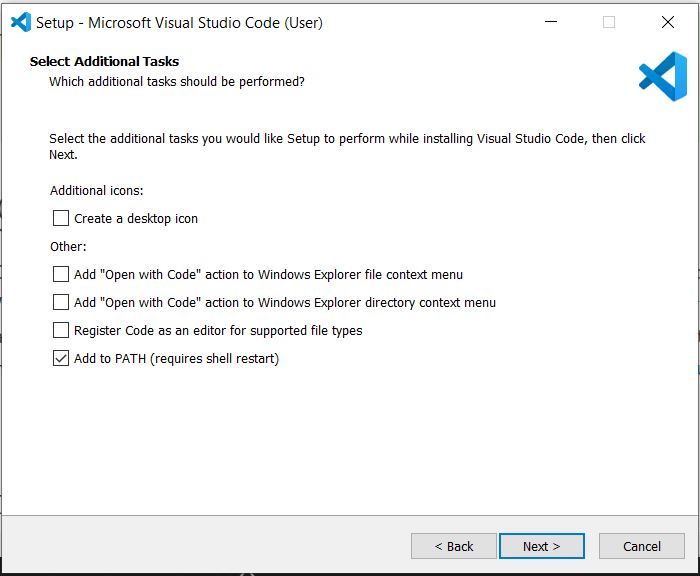
Tutorial: <https://www.youtube.com/watch?v=veJvQ88ULOM>



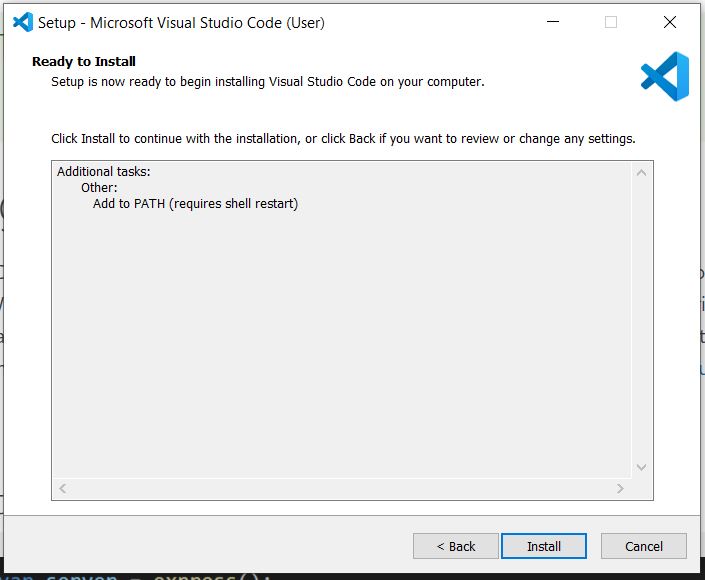
Go to the Visual Studio Code website, and click on “Download for Windows”. After the .exe finishes downloading, open it up.



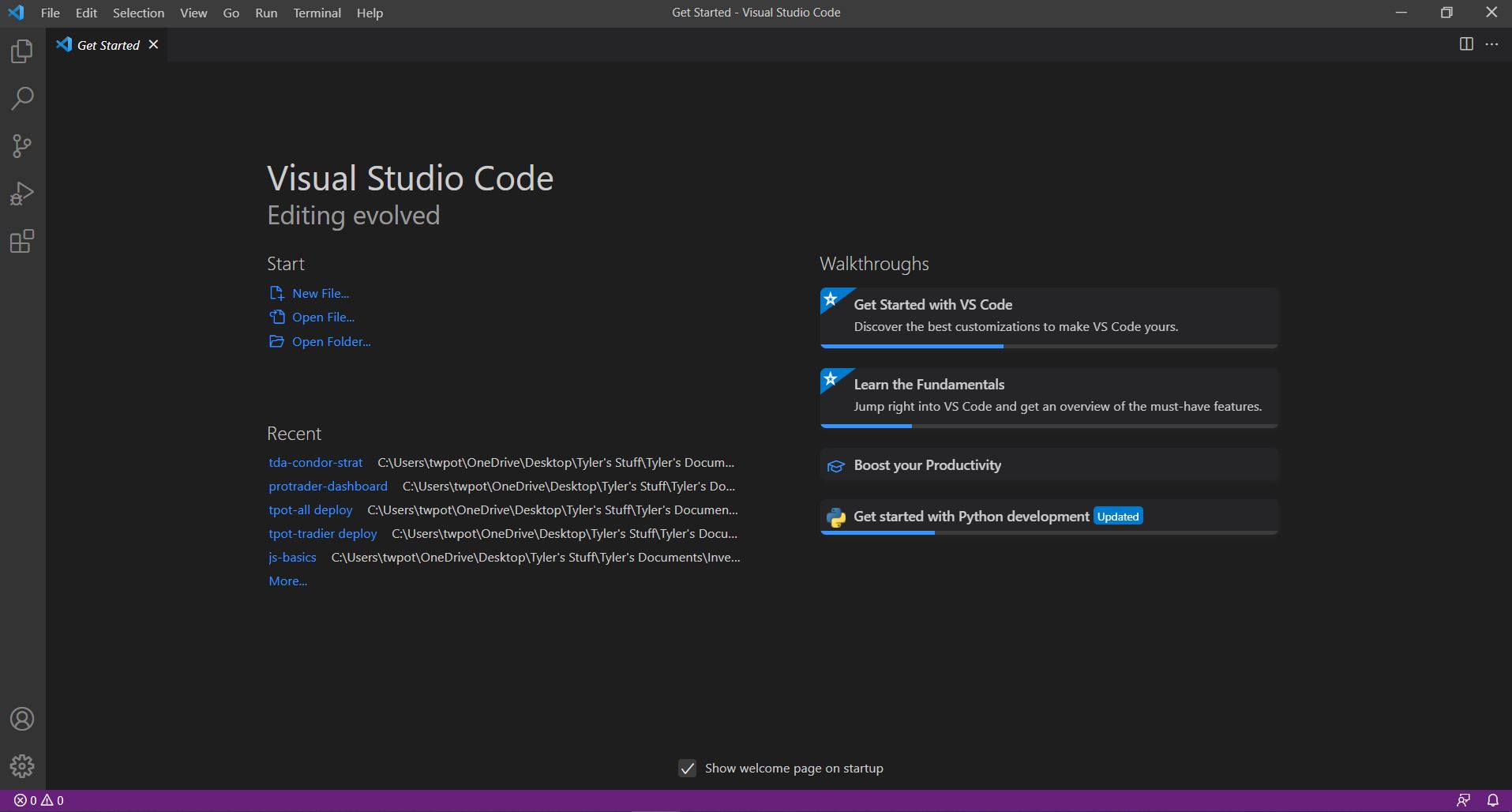
Select “I accept the agreement” and then click “Next >”



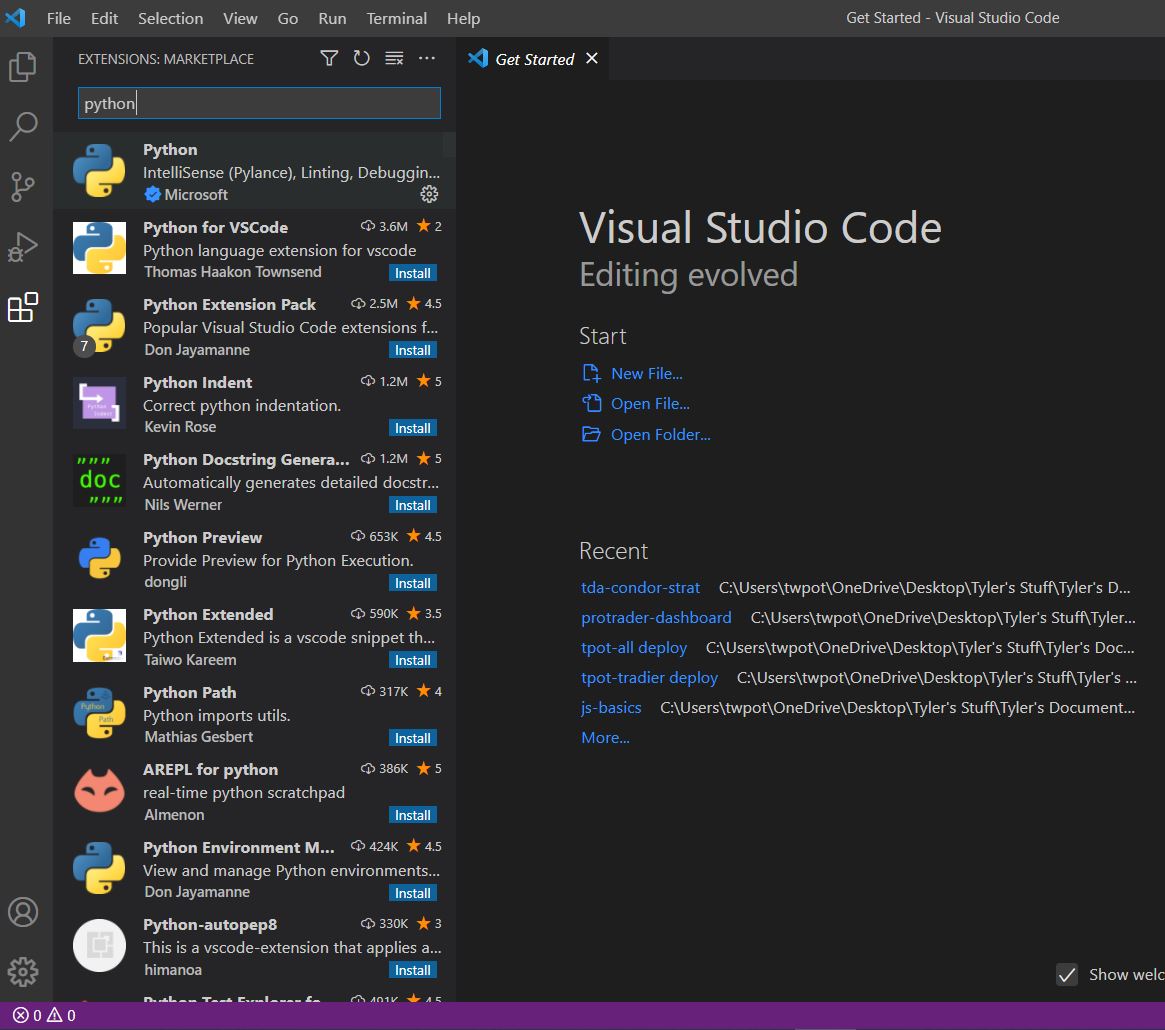
Make no changes here. Click “Next >”



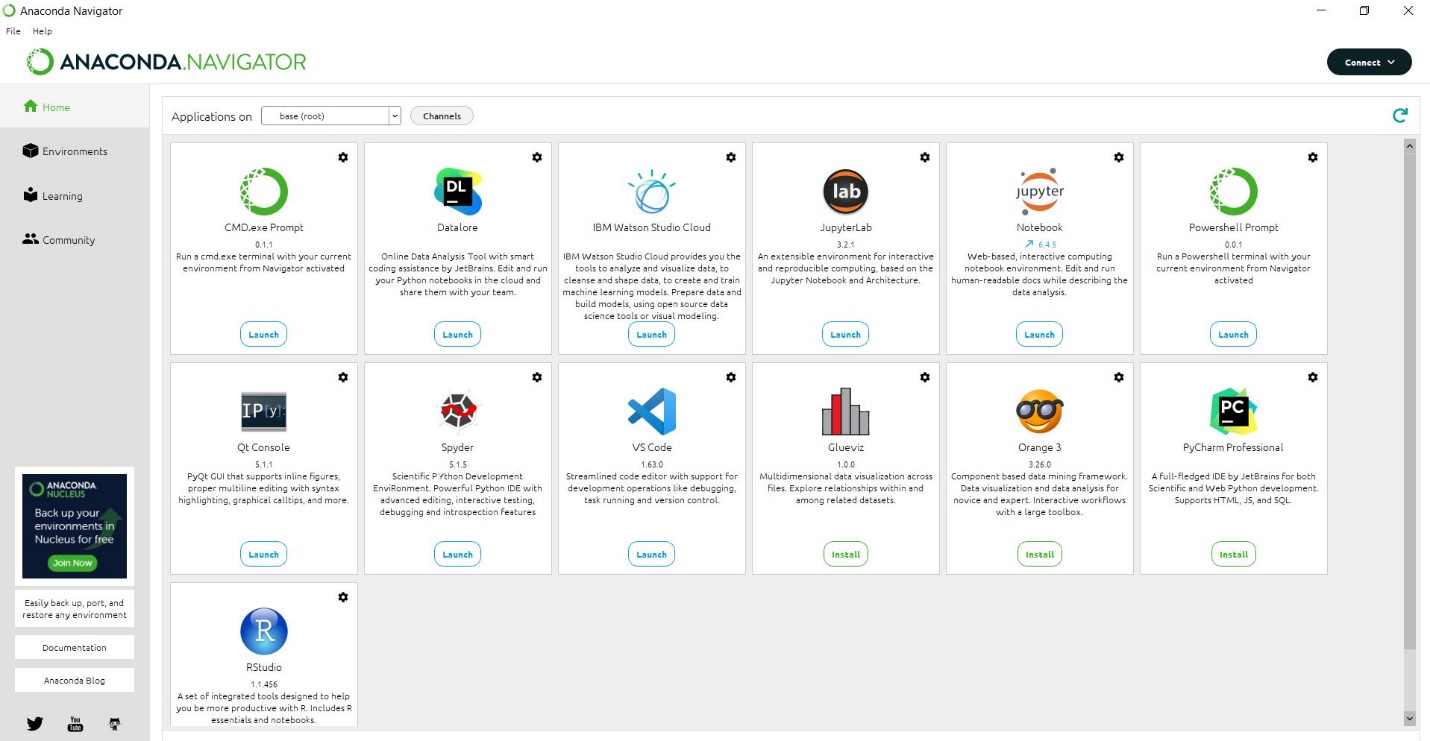
Finally, click “Install”. Once done, open up it up. You should come to the page below.



Next, we need to add the Python add-on to Visual Studio Code. Click on the “extensions” icon, which the is the 5th icon down on the top left navigation sidebar. When you hover over it, it should say “Extensions”. Then, type in “Python”.



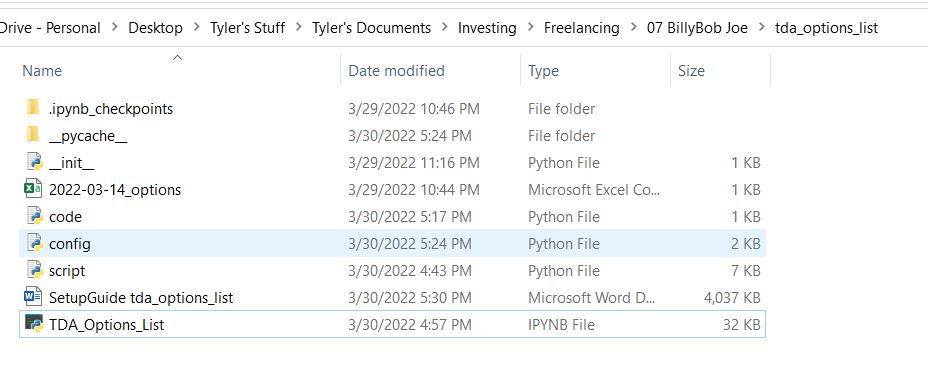
Select the first result, click install on the bottom right. After doing this, completely close out of Visual Studio, as well as Anaconda if that is still running. Restart Anaconda and launch Visual Studio Code from within Anaconda by clicking “Launch” in the below screen. An icon for “VS Code” should now be there.



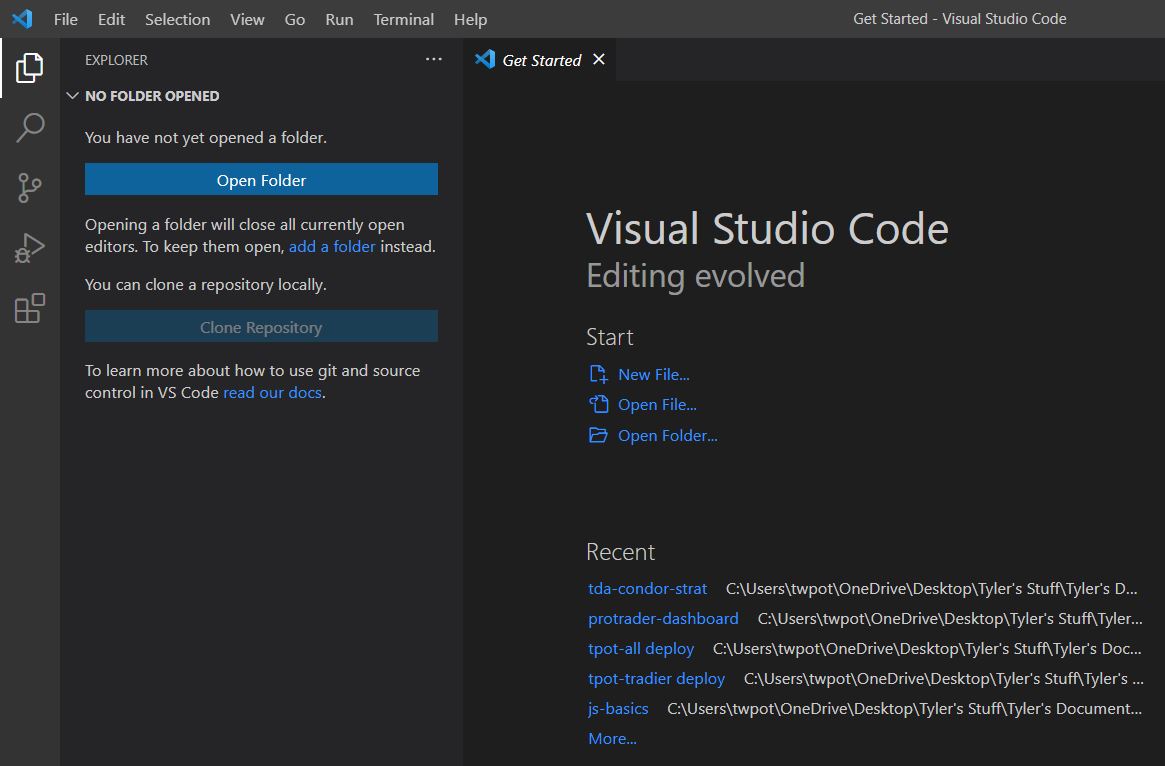
We are about to open up the deliverables I sent to you, but before we do that, we need to make sure the file structure is correct. In any place, on your hard drive, create the following structure:

|  |  |
| --- | --- |
| First level | Second level |
| tda\_options\_list (folder) |  |
|  | \_\_init\_\_ (.py) |
|  | 2022-03-14\_options (.csv) |
|  | code (.py) |
|  | config (.py) |
|  | script (.py) |
|  | SetupGuide tda\_options\_list (.docx) |
|  | TDA\_Options\_list (.ipynb) |

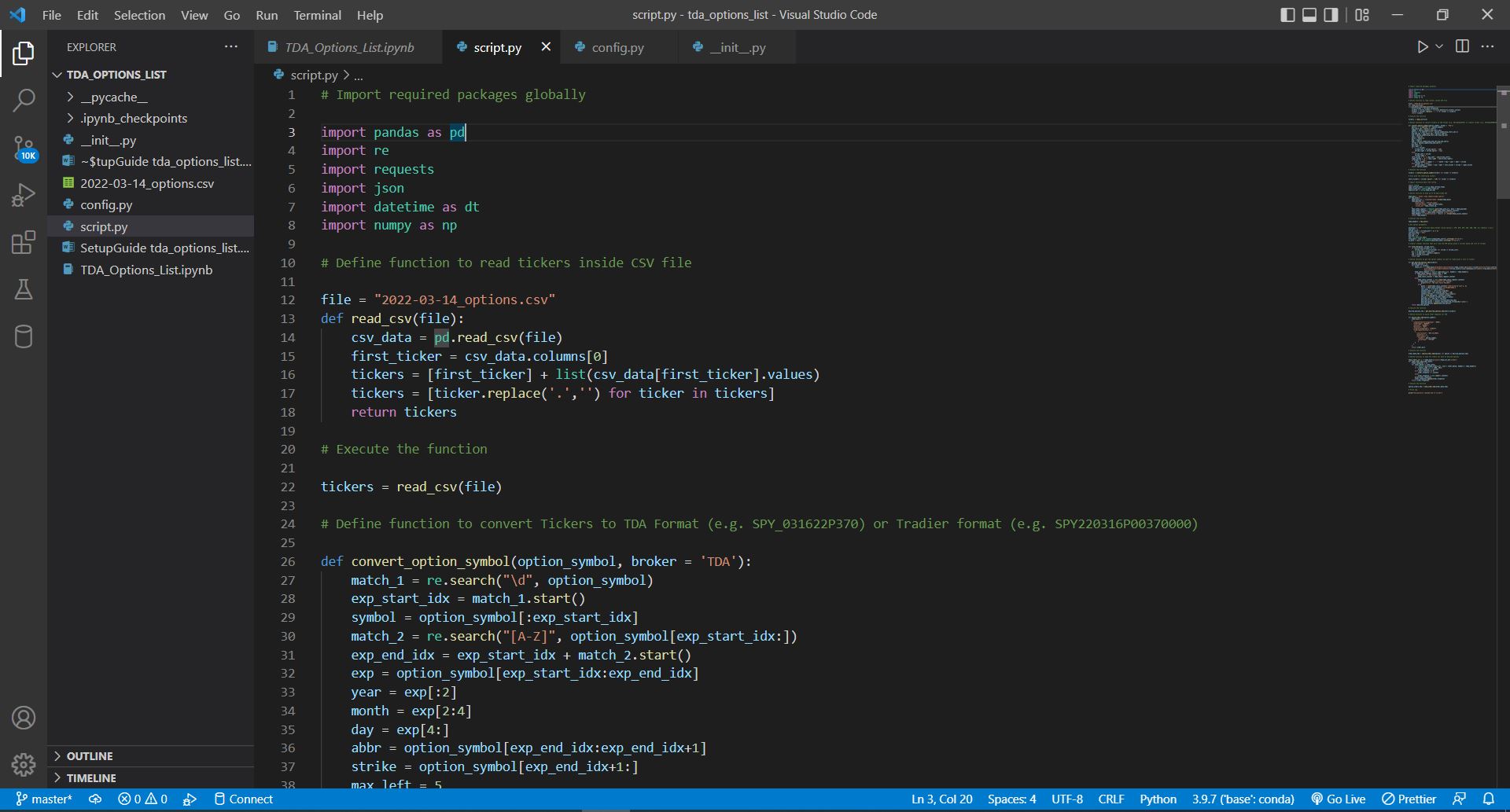
Here is a screenshot of the second level. Don’t worry about the \_\_pycache\_\_ or .ipynb\_checkpoints folders. Those will create themselves.



Next, in Visual Studio Code, open the entire folder. Click on the “Explorer” icon (top left sidebar 1st icon on top), and then “Open Folder”. Find the folder you just created and select “Open folder”



Now we can see all the deliverables:



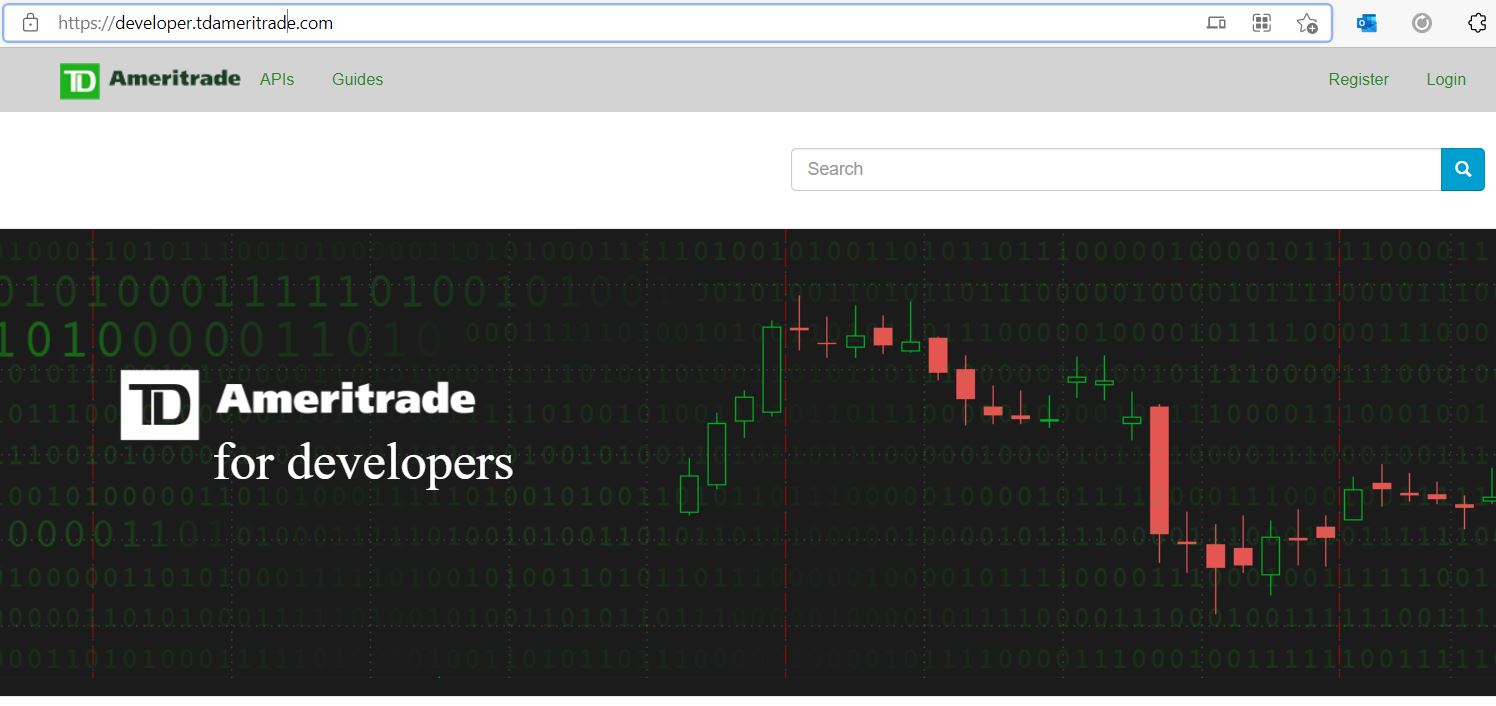
We are going to want to run this file, but right now it won’t work because first we need to do some other things.

**3: TDA Developer Account**

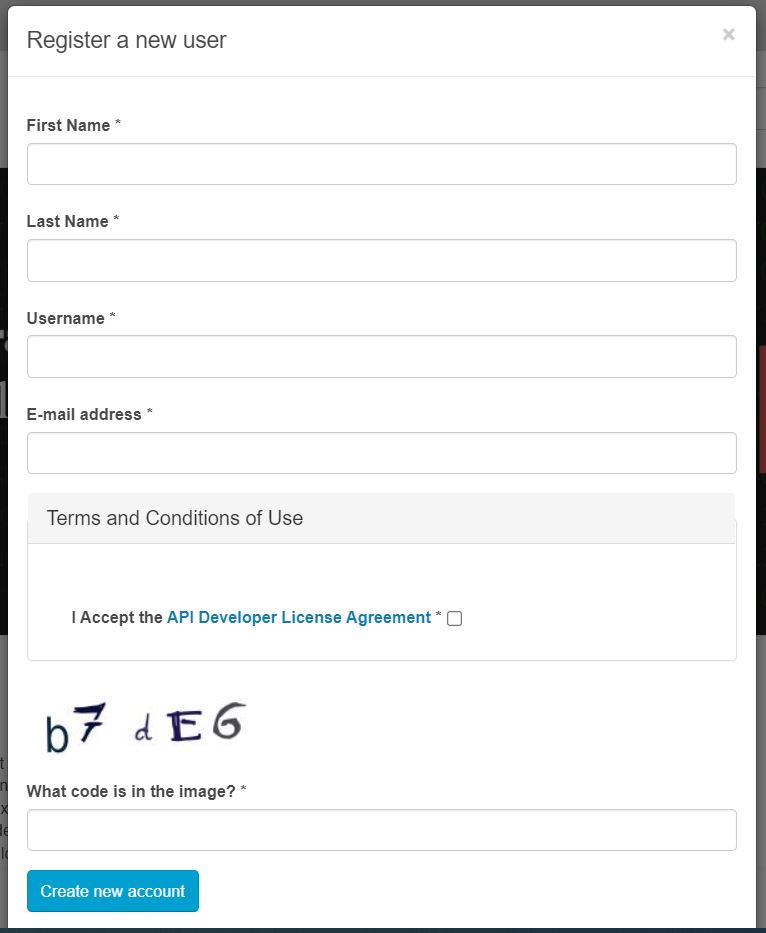
TDA requires that we setup a developer account in order to use their API, so let’s do that.

Link: <https://developer.tdameritrade.com>

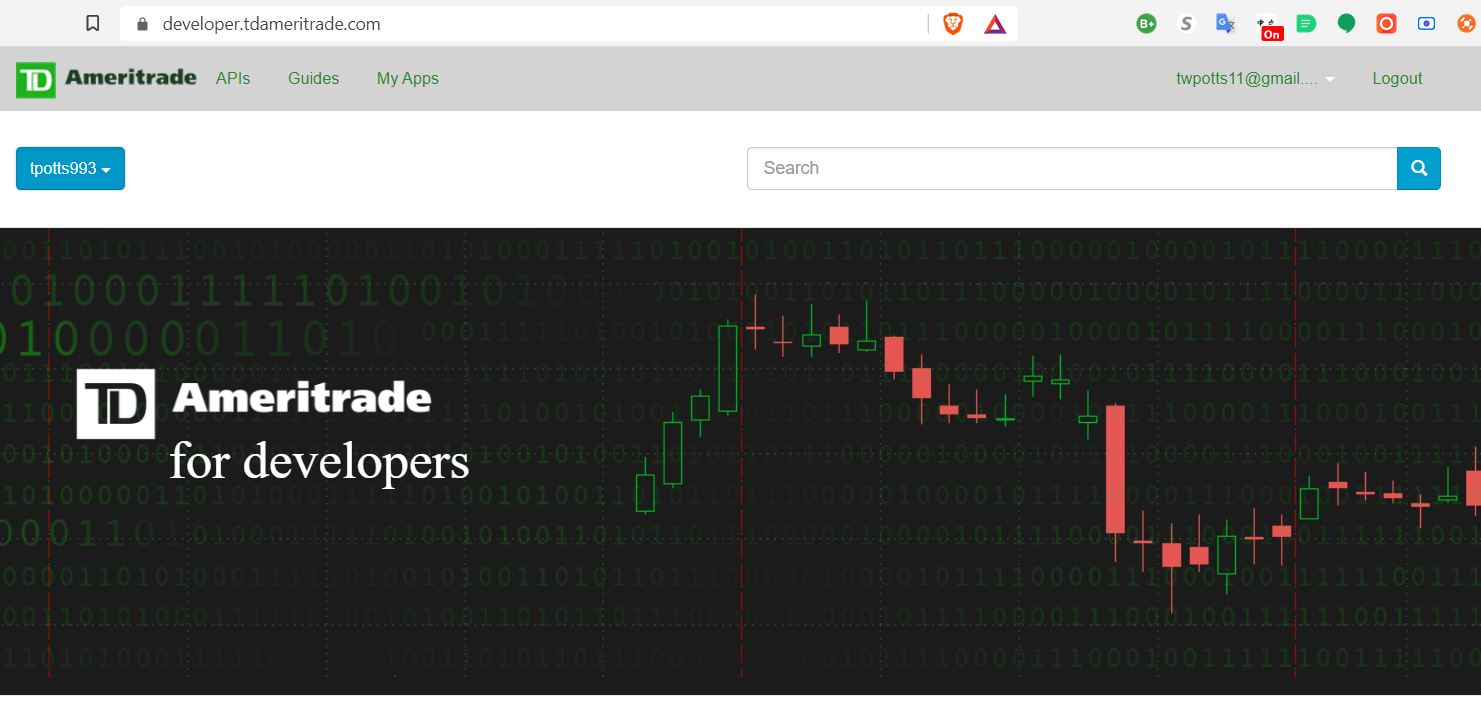
Tutorial: <https://www.youtube.com/watch?v=mLjtVVqp5iw>



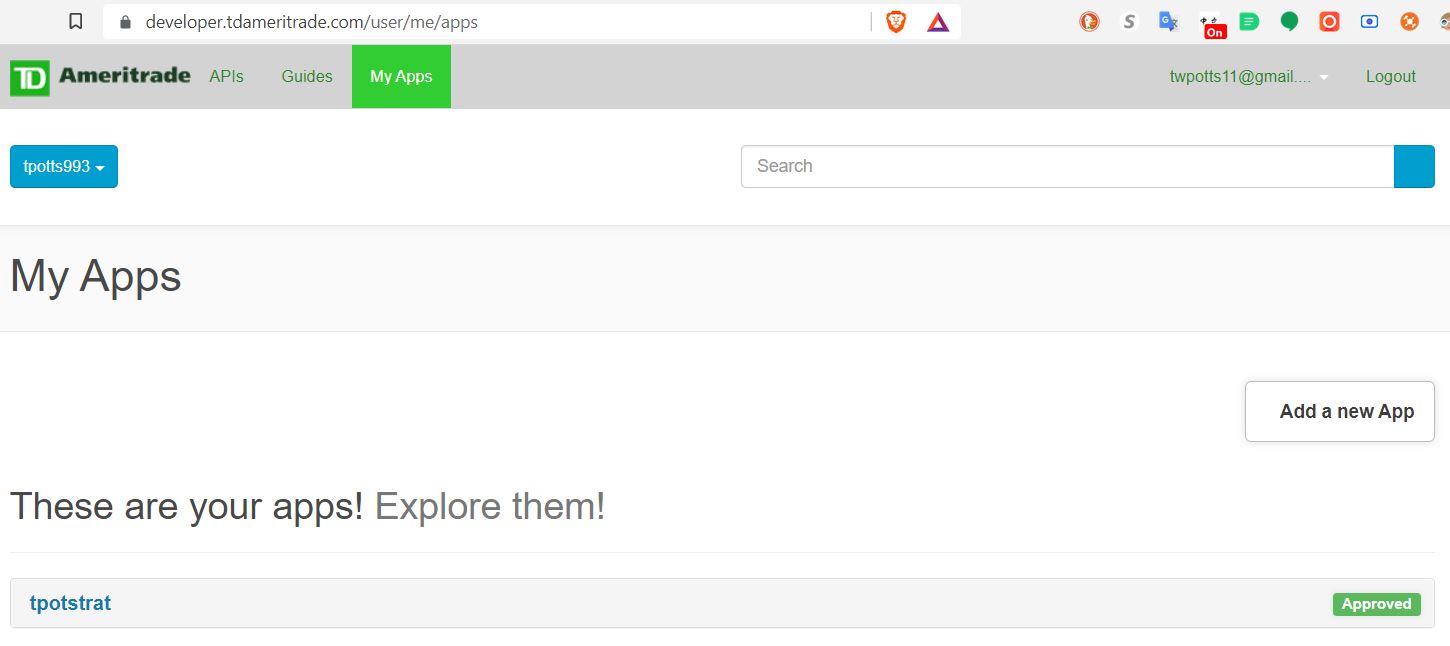
Click on “Register” in the top right corner.



Fill in your information. Once you finish do that, go back to the home page and this time you should be able to see a tab for “My Apps”. Click on that.

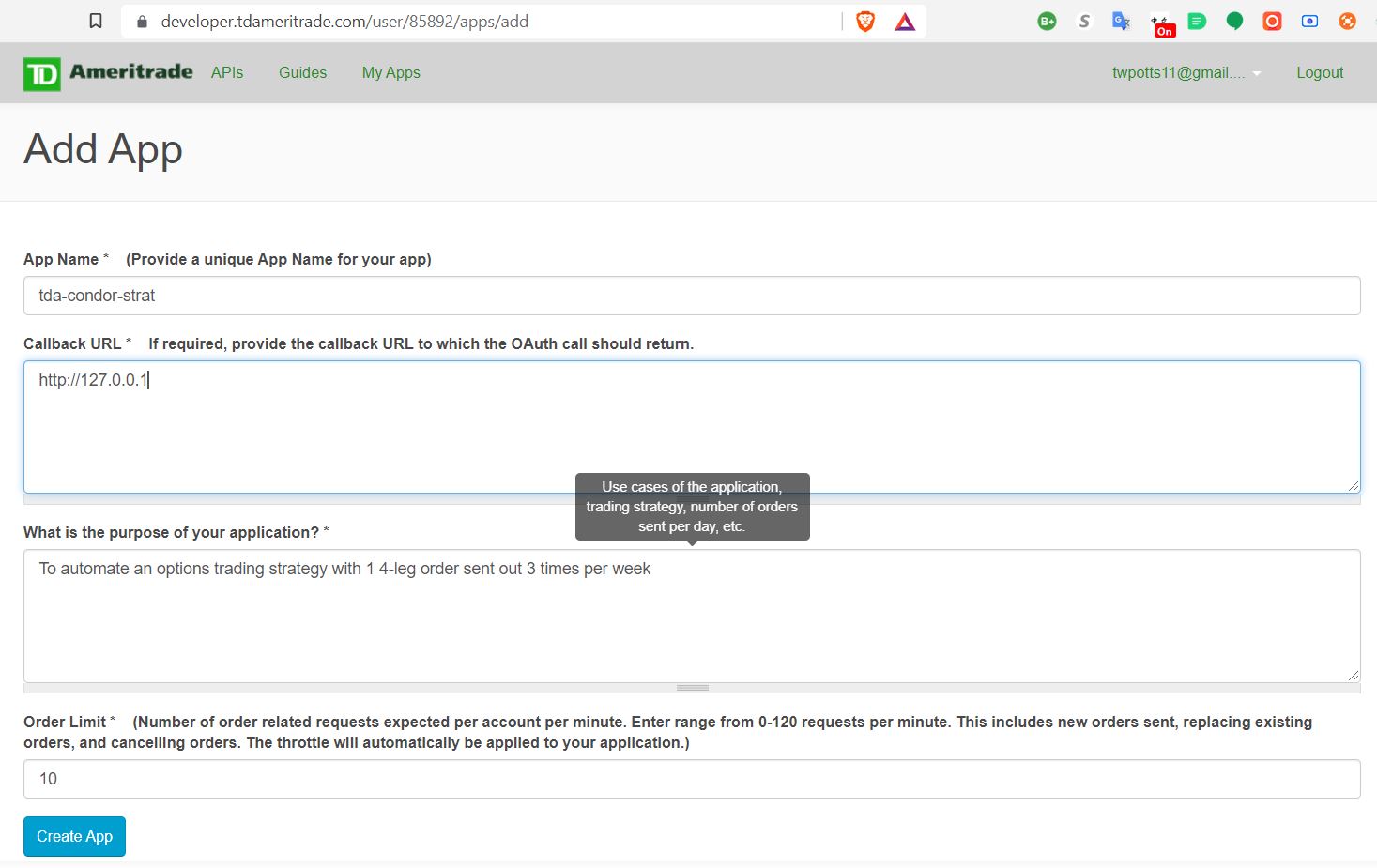


Next, click on “Add a new app” on the right side. There is a bug on this website that may cause that button to disappear. If that happens to you, refresh or try a new browser, and try to click on it as fast as possible.

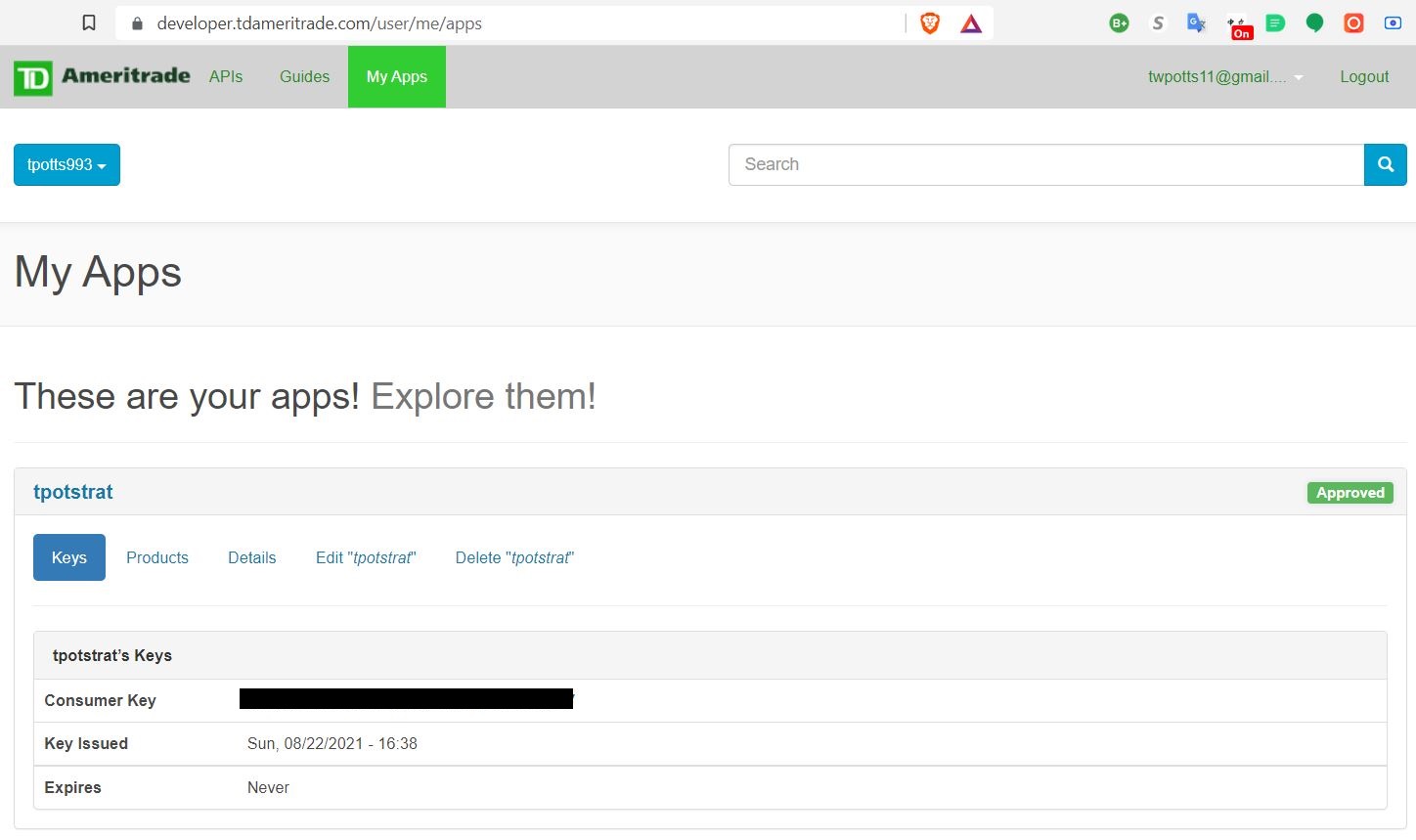


Fill in the details to look something like this

|  |  |
| --- | --- |
| Name | tda-options-list (your name) |
| Callback URL | <http://127.0.0.1> |
| Purpose | To send orders based on tickers in a CSV file |
| Order limit | 10 |



Click “Create app” at the bottom. Next, let’s get the API by going back to “My Apps” and clicking on your newly created app, then the “Keys” tab.



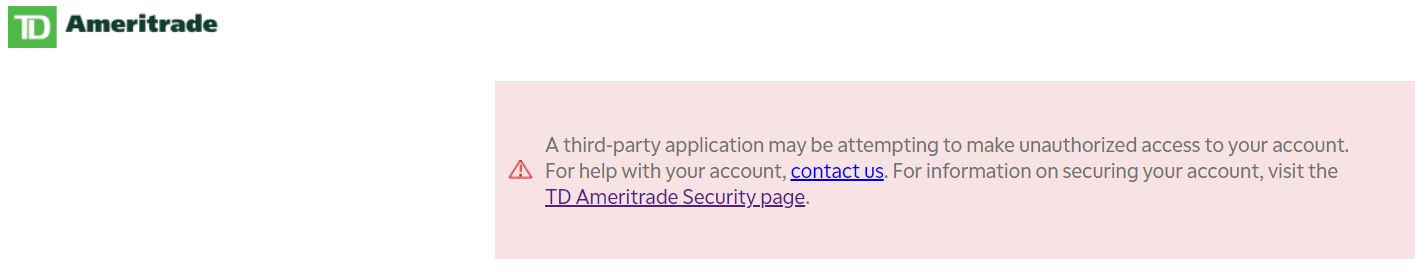
Copy down this “Consumer Key” (where the blacked out box is), and don’t share it with anyone else. Congrats, you created an app and got a key, but now you need use this key to get a token.

**4: TDA Token Reset**

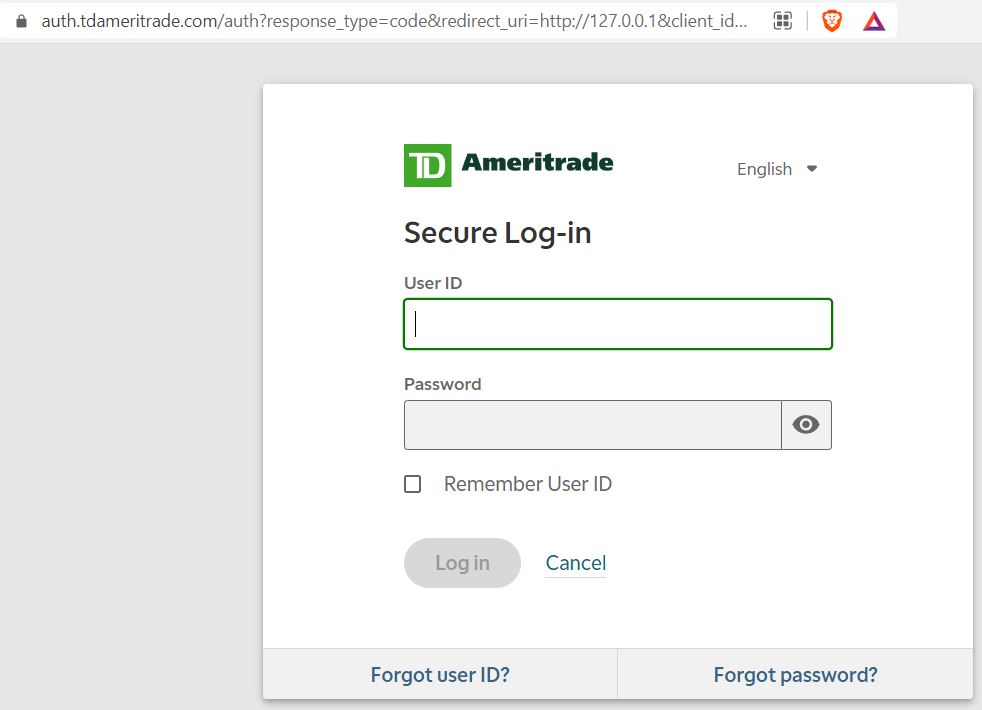
We must use a token in order to authenticate with the TDA API, and TDA has decided that this token shall expire after 90 days (other brokers use non-expiring tokens). If your token is expired, your bot will not work. Therefore, you must follow these steps manually every 90 days.

Link: [https://auth.tdameritrade.com/auth?response\_type=code&redirect\_uri=http://127.0.0.1&client\_id={YOUR\_CONSUMER\_KEY}%40AMER.OAUTHAP](https://auth.tdameritrade.com/auth?response_type=code&redirect_uri=http://127.0.0.1&client_id=%7bYOUR_CONSUMER_KEY%7d%40AMER.OAUTHAP)

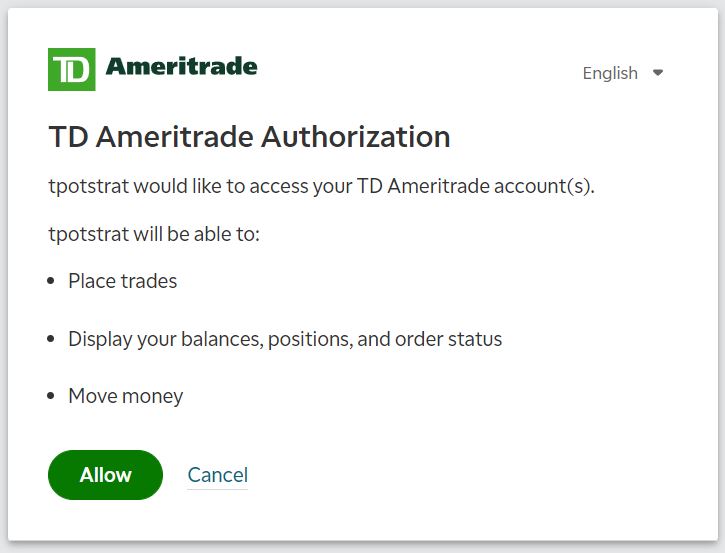
I have seen some attempts out on the web to automate this process, but a lot of these no longer work and usually generate the following erorr. Unfortunately for those of us who automation, TDA seems to fight against our goals of 100% automation.



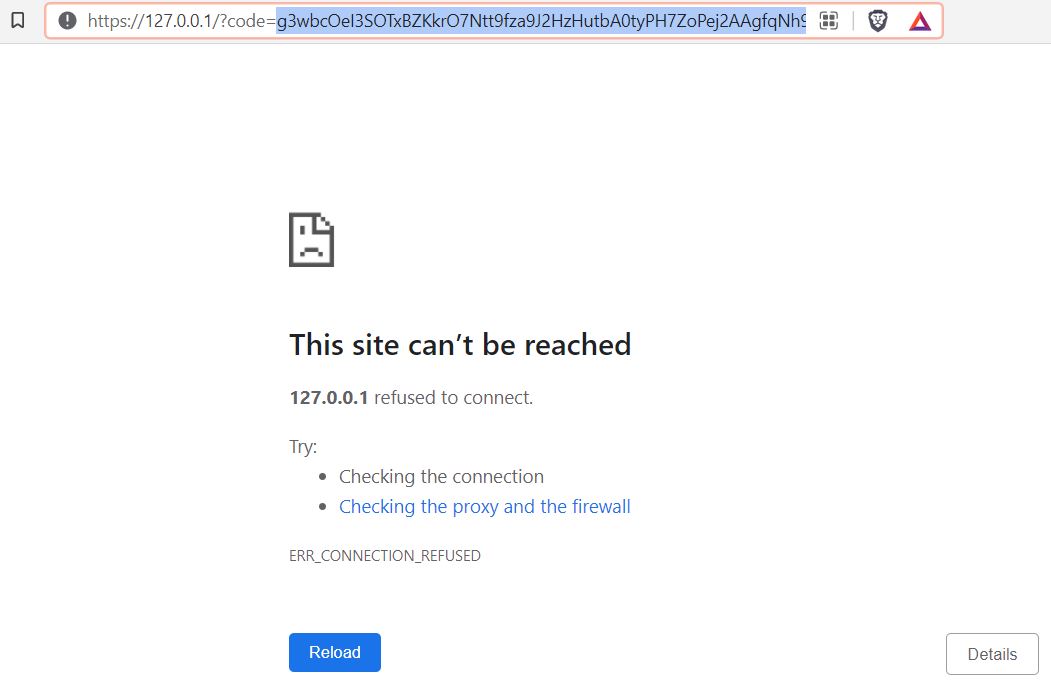
Substitute in the consumer key you just copied down into the 1st link above (auth.tdameritrade …). Enter that URL into a browser. It should take you to the following page:



Enter in your normal TD Ameritrade login information.



Click “Allow” in order to enable your app to make automated trades on your behalf. The next page will say “This site can’t be reached”, but this is NOT an error. We aren’t interested in the content; rather, we want to copy the section of the URL up top that follows after “code=” (highlighted in blue). Copy down that entire segment.



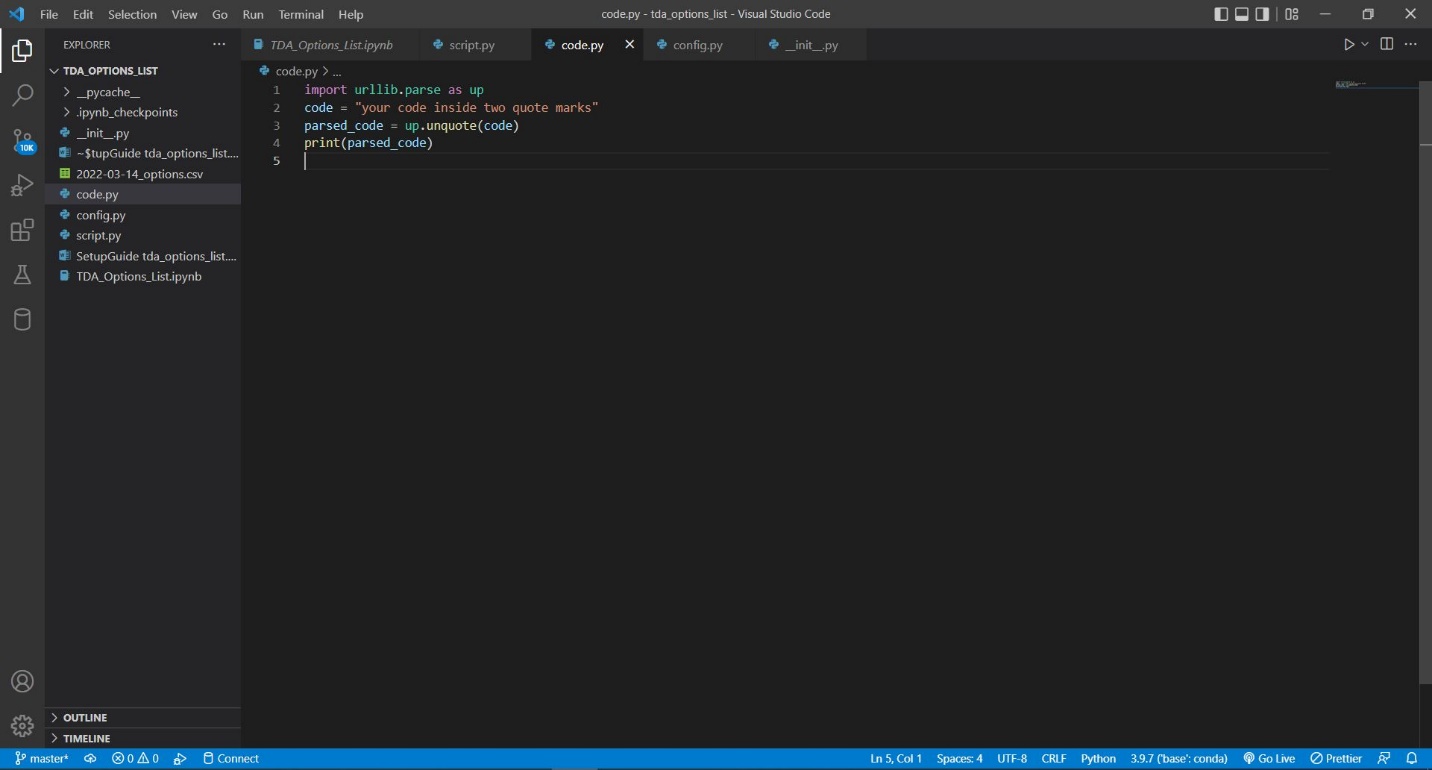
Next, we need to parse this code. Go back to Visual Studio Code, open up the project folder, and click on “code.py”. You should see the following code:

import urllib.parse as up

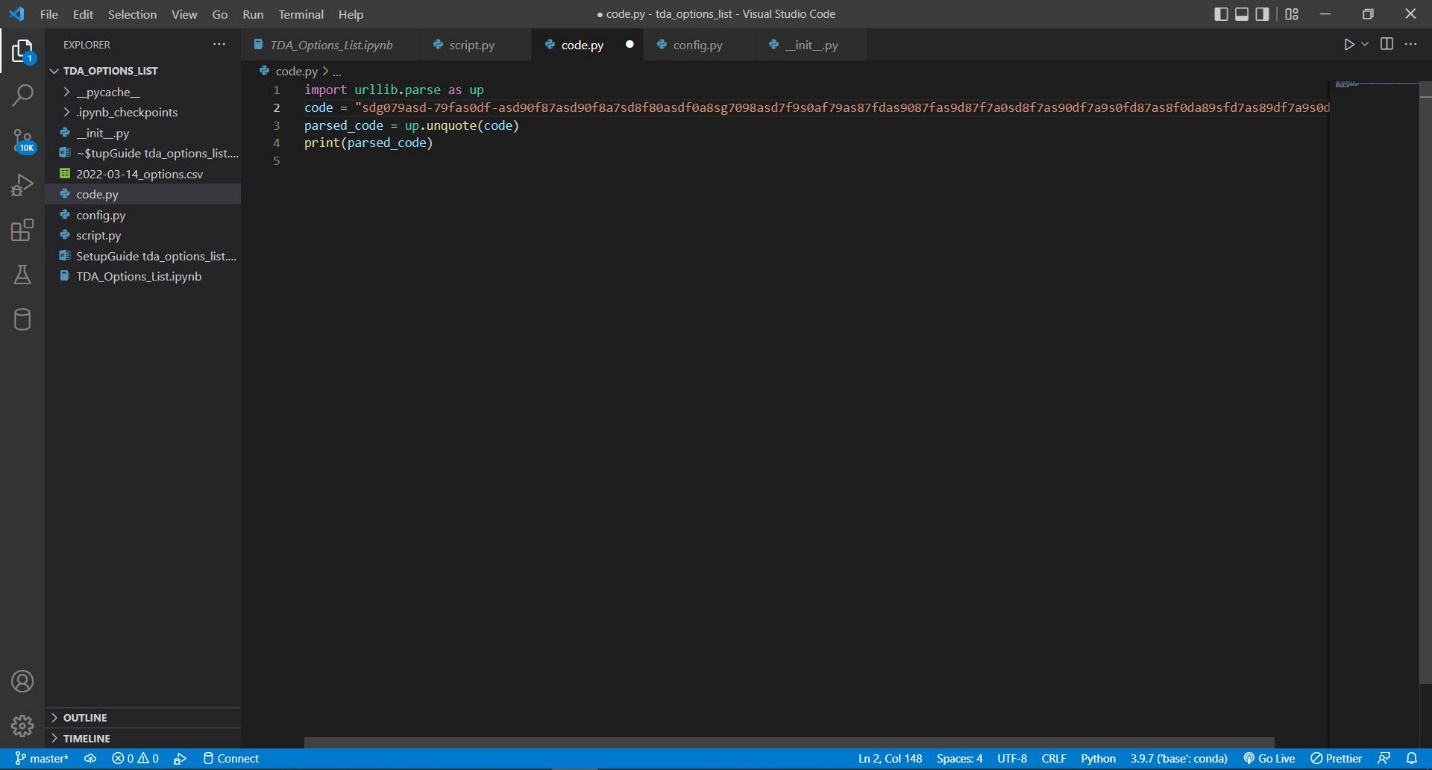
code = ‘{your code inside two quote marks}’

parsed\_code = up.unquote(code)

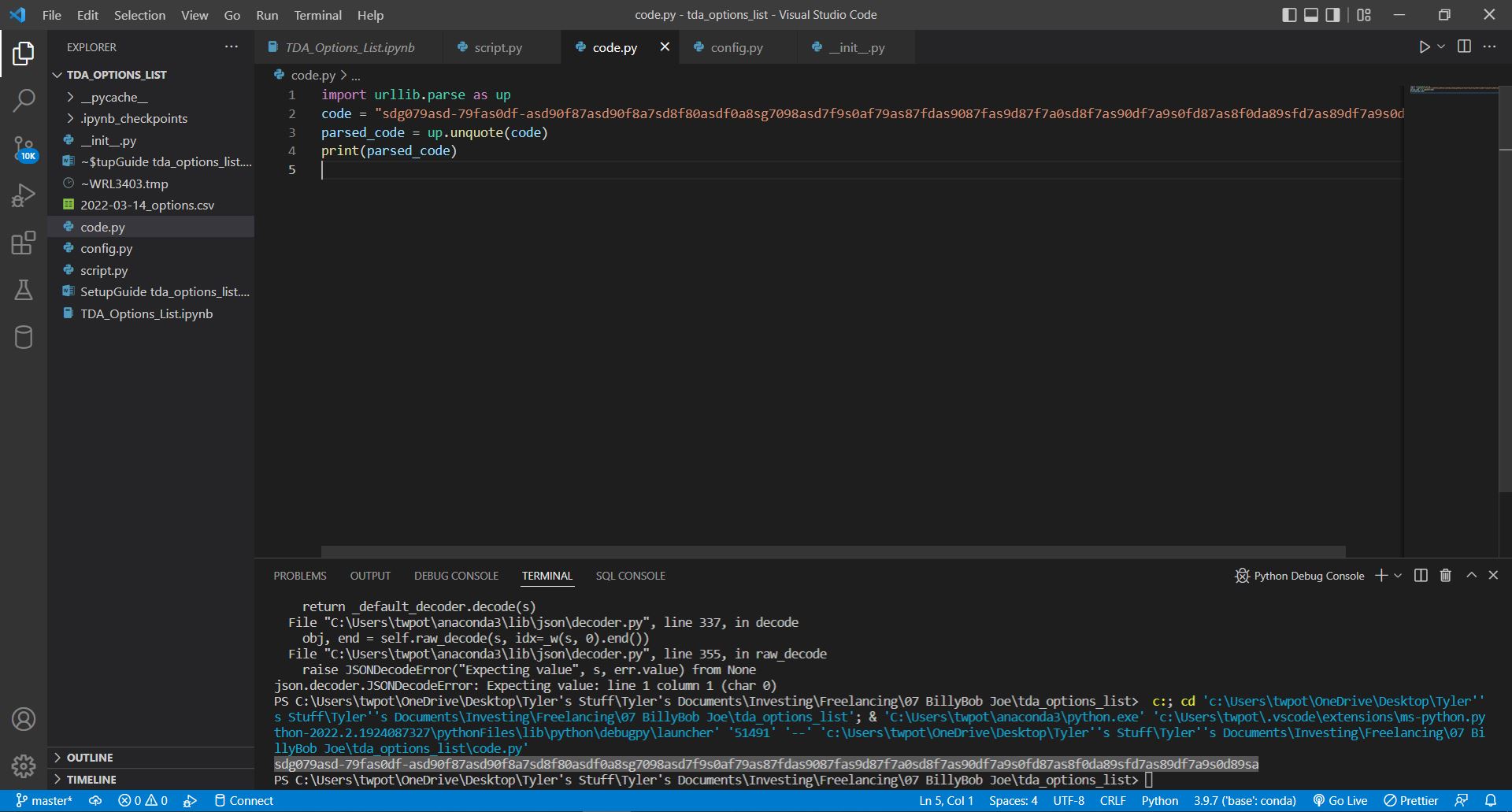
print(parsed\_code)



On line 2, where it says “code”, paste your new code inside the quotation marks. Your screen should now look something like this.



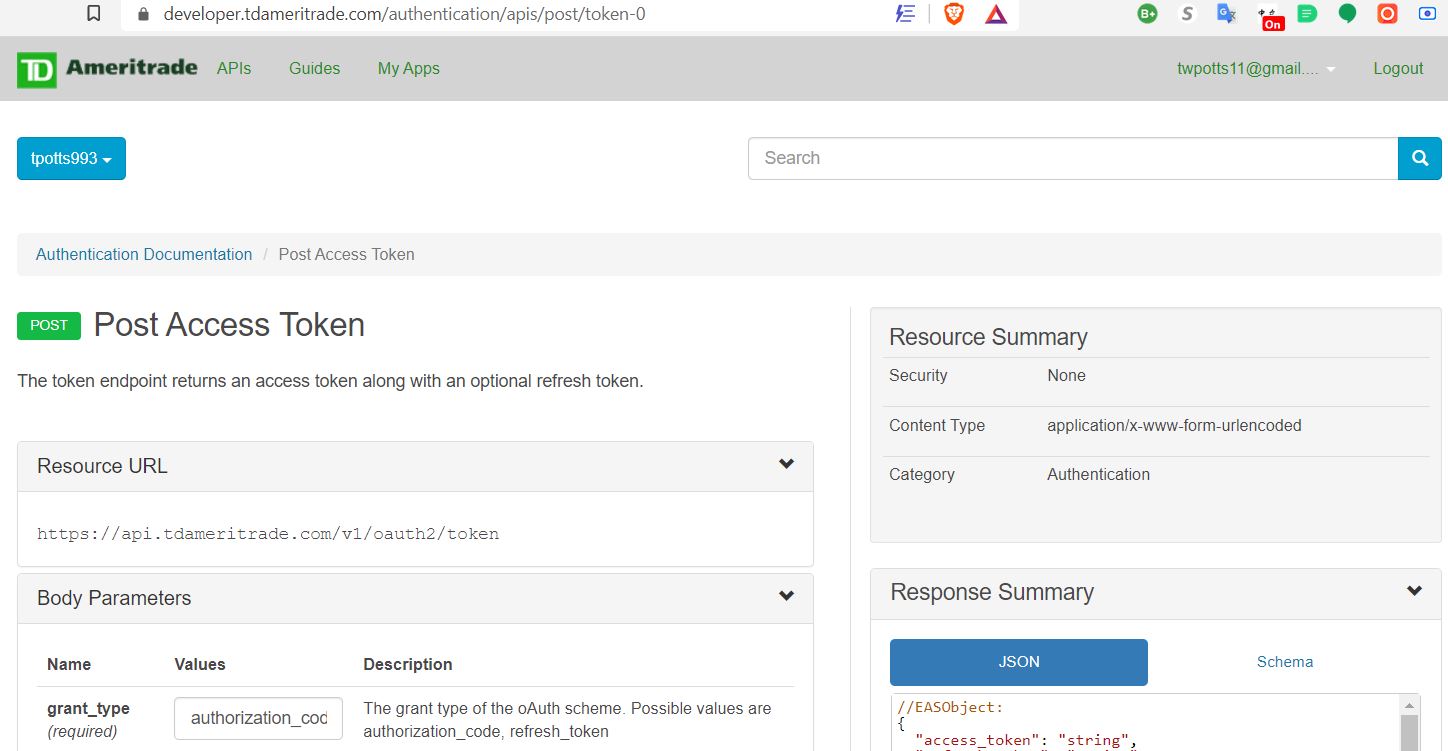
Now we need to run this file. To do that, on the top nav menu, click on “Run” then “Run Without Debugging”, or you can also use keyboard shortcut CTRL+F5



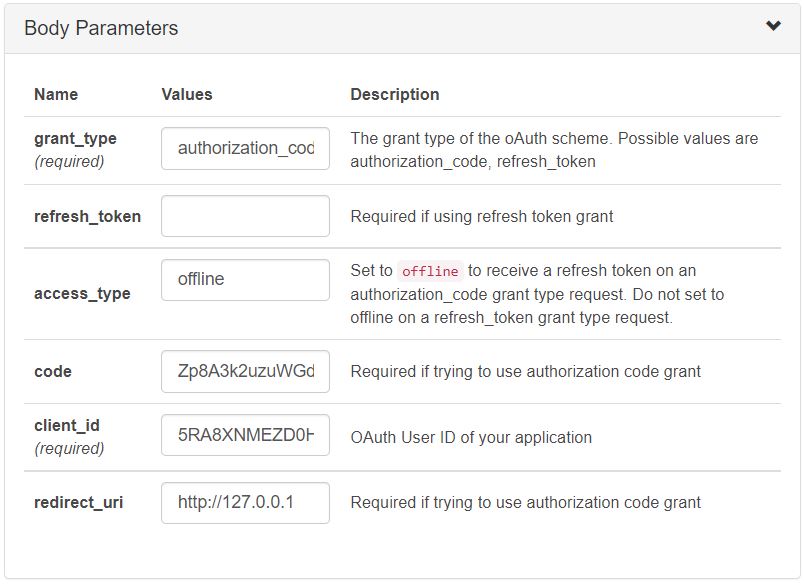
In the “Terminal” that appeared on the bottom, copy the new code that you see highlighted in the image above. The new code will look very similar to the old code, except it no longer contains “%” signs as well as a few other changes.

Next, go to the following website. It should look like below.

<https://developer.tdameritrade.com/authentication/apis/post/token-0>

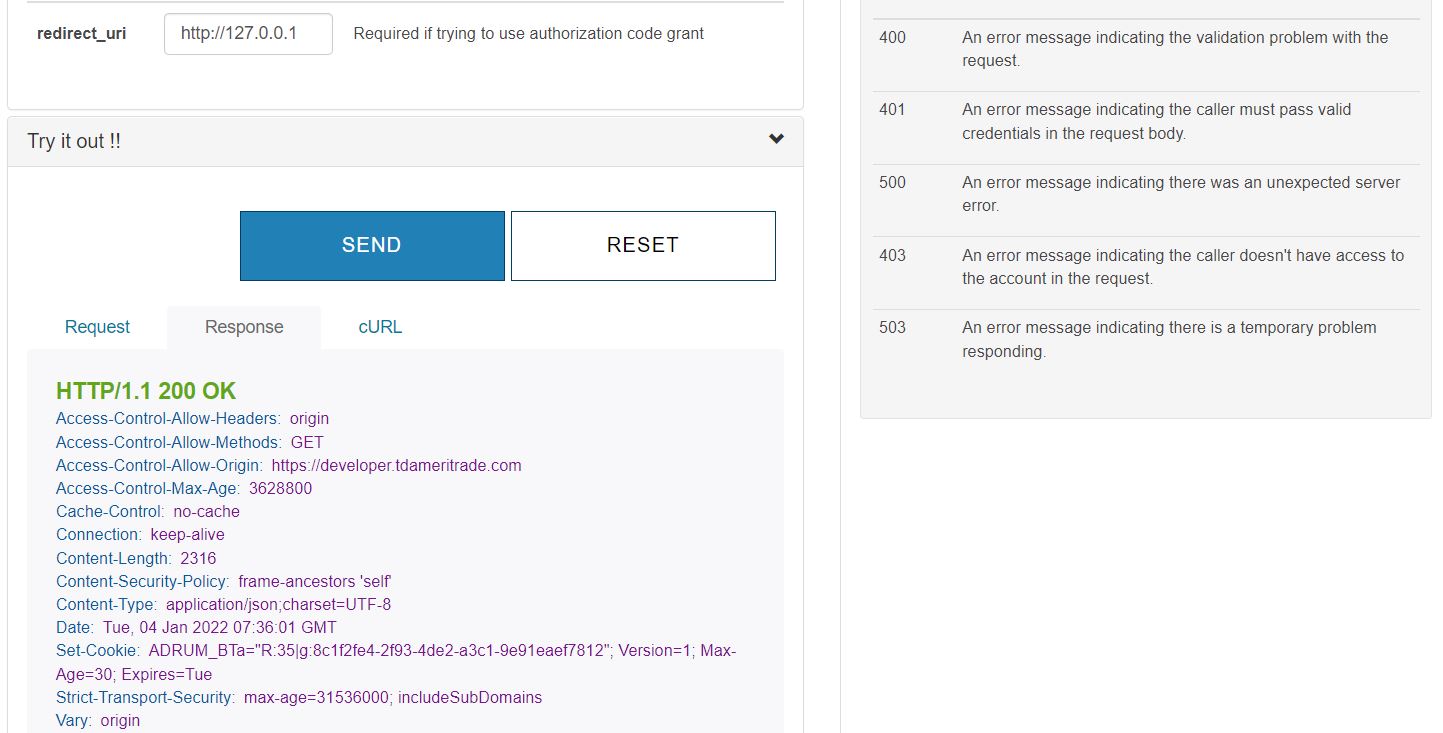


Fill out the “Body Parameters” section like this:

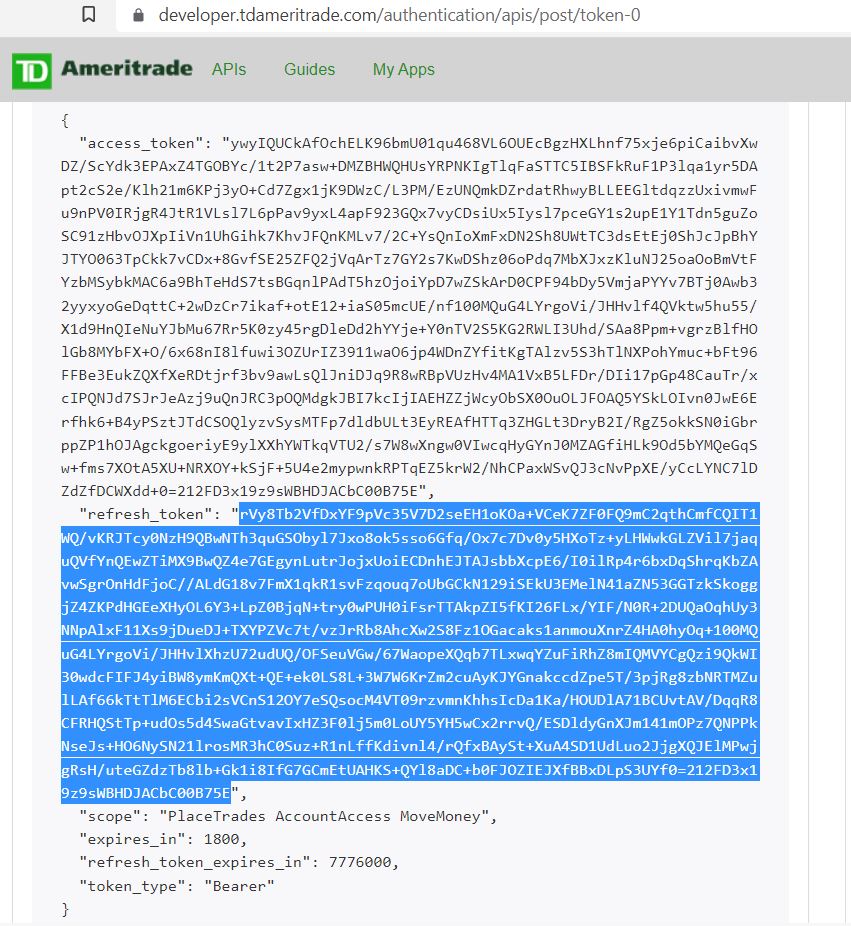


|  |  |
| --- | --- |
| grant\_type | authorization\_code |
| refresh\_token |  |
| access\_type | offline |
| code | {the new parsed code we just generated} |
| client\_id | {your Consumer Key} |
| redirect\_uri | http://127.0.0.1 |

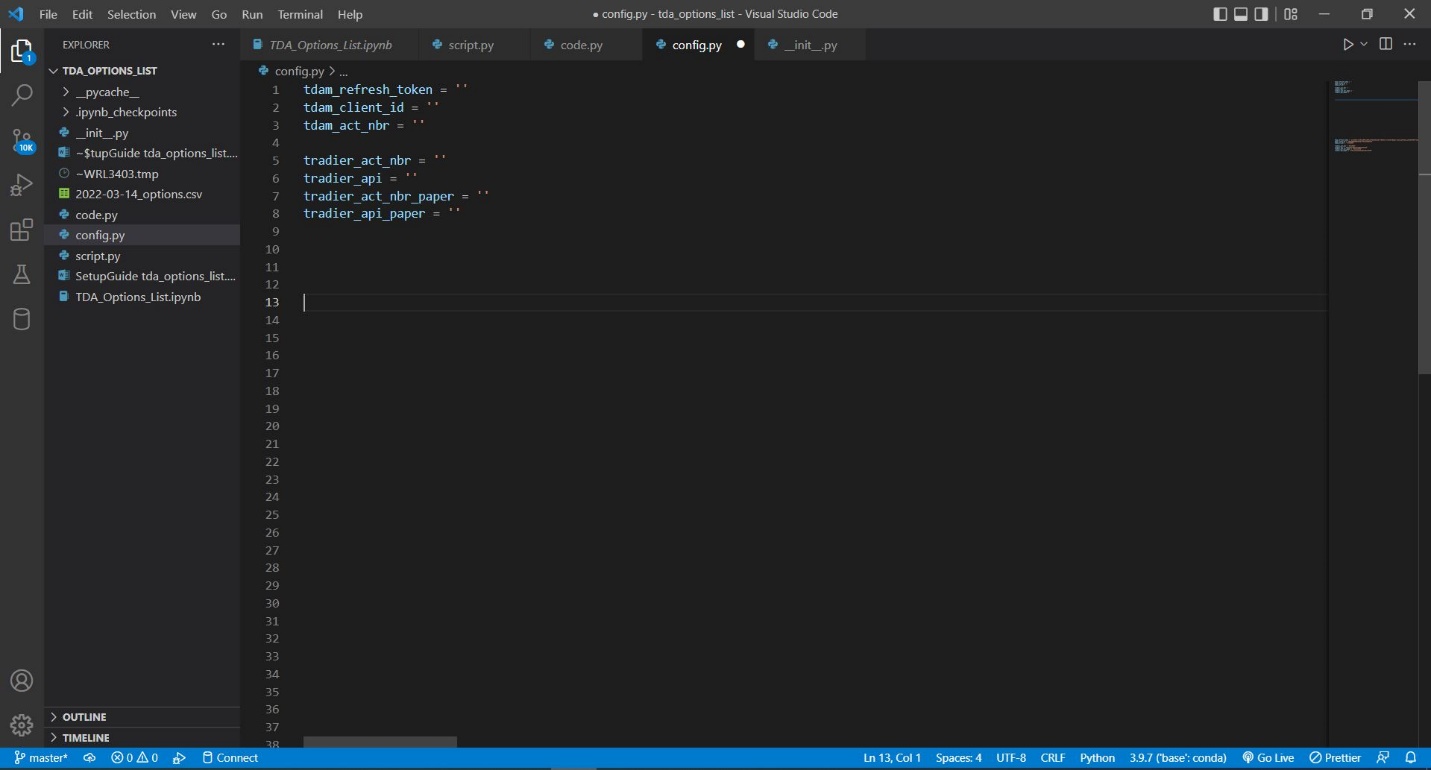
At the bottom, click “SEND”. If you get a 200 status code back, then congratulations, it worked!



If you did not get a 200 status, try to do everything again, but faster. Sometimes your “code” will timeout if not used within about 10 minutes. Also make sure that secure login page (the one that says “This site can’t be reached”) is still open.

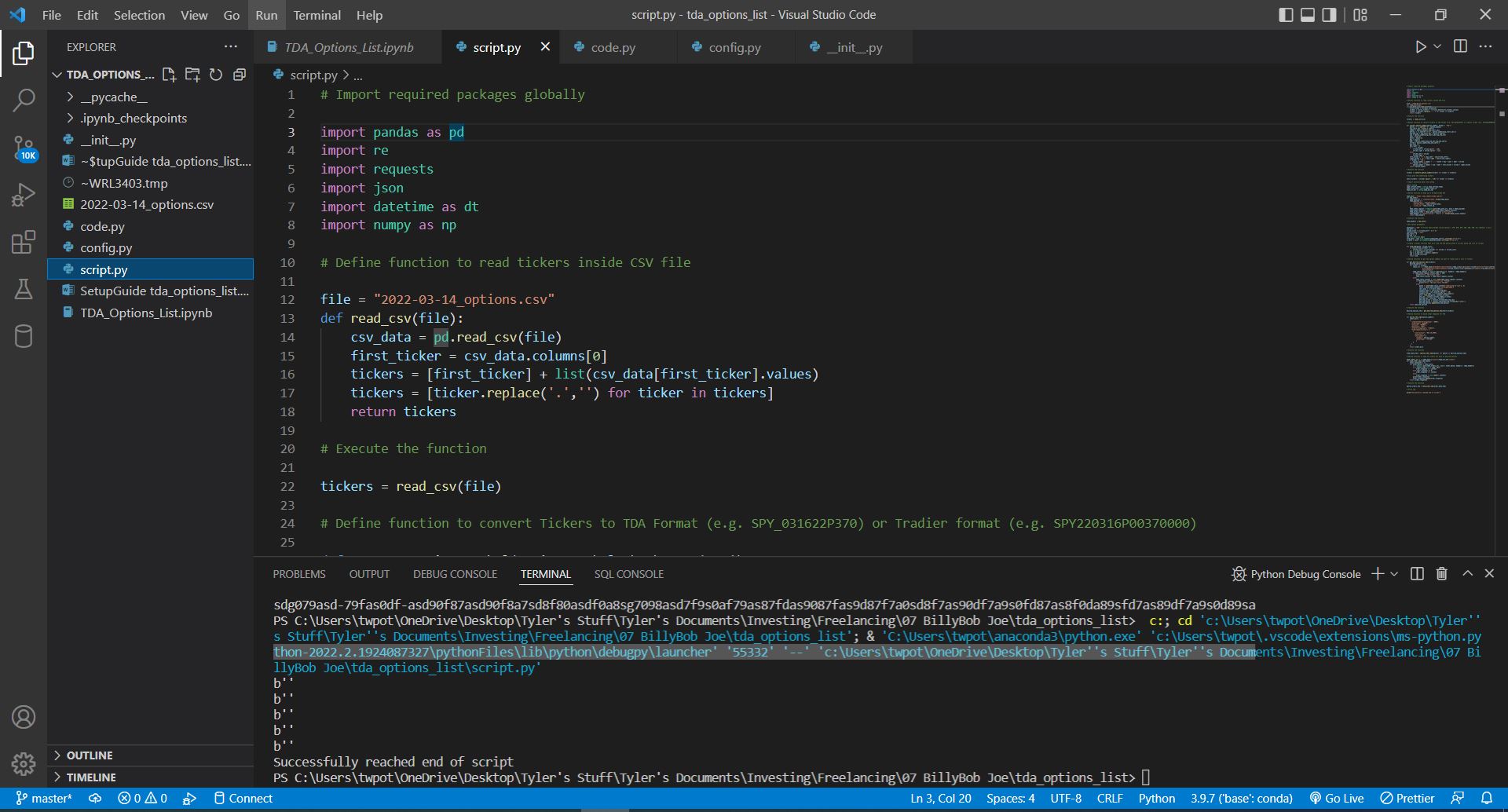


Copy down the refresh token (highlighted in blue). Next, open up Visual Studio Code and go to your config.py file.



Paste the refresh token into the highlighted area of the config file (tdam\_refresh\_token = ). Also fill in the rest of the config variables with appropriate TDA and Tradier info. Make sure to save the file by clicking on File > Save or CTRL + S.

Now you should be able to run your script. Go back to script, click Run > Run Without Debugging (CTRL + F5. You should be able to see “Successfully reached end of script” printed in the bottom terminal



Congrats, you are done. If you get errors, we will need to troubleshoot.