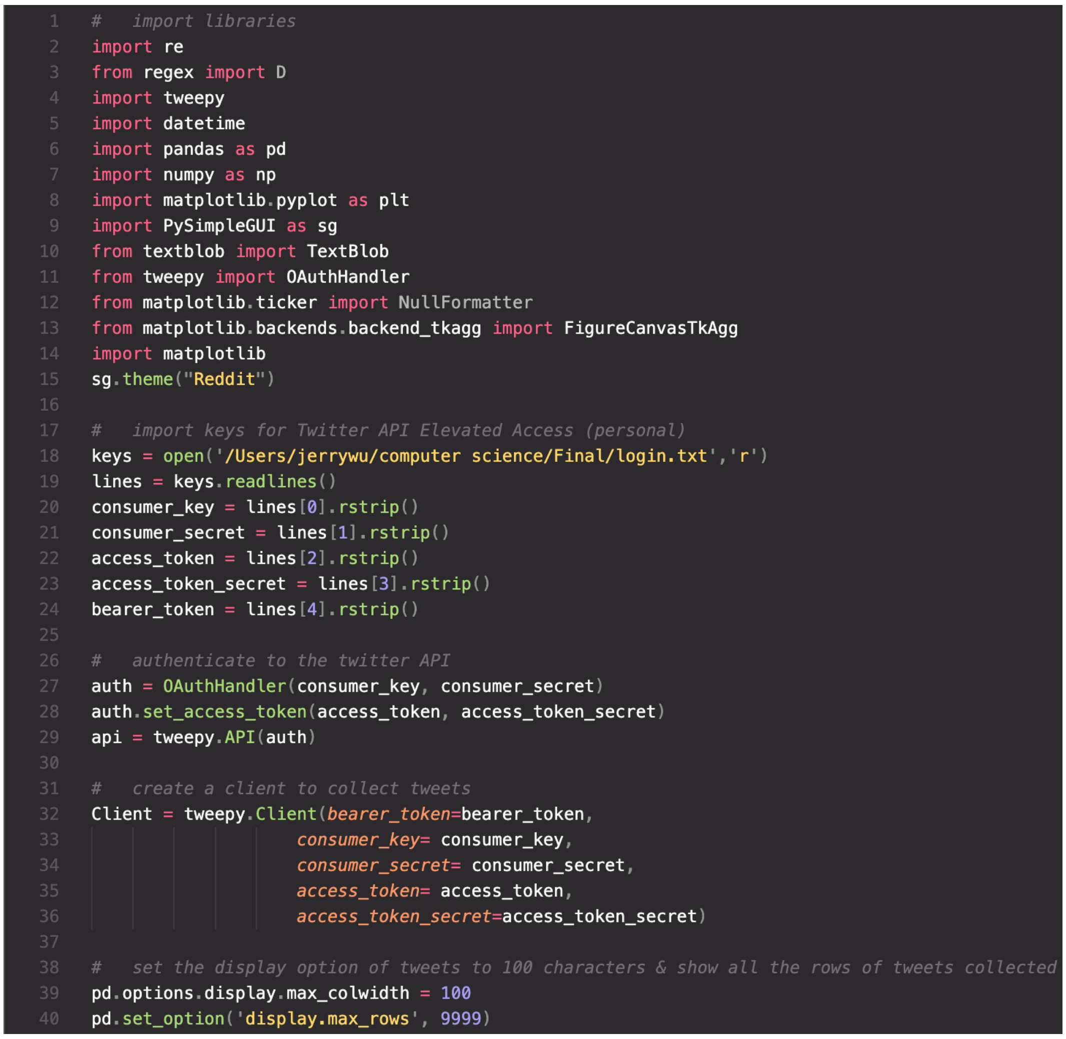
Criterion C

Techniques Used

- Additional Libraries
  - Twitter API
  - Sentiment Analysis
  - PySimpleGUI
  - Datetime
  - Matplotlib
- File I/O
- Loops
- If Statement
- Nested Loops
- Dictionary
- Arrays
- Sort Function
- Lambda function
- User-Defined Methods
- User-Defined Methods with Parameters
- User-Defined Methods with Return Values
- Class object
- Data frame

Library, Twitter API & Pandas Setting

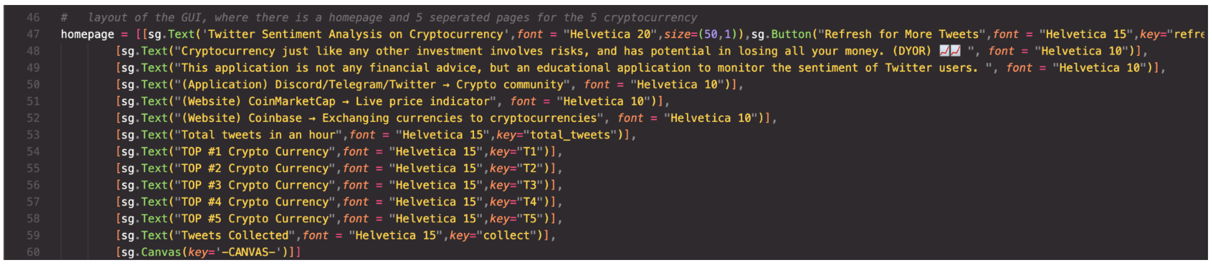


```python
#   import libraries
import re
from regex import D
import tweepy
import datetime
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import PySimpleGUI as sg
from textblob import TextBlob
from tweepy import OAuthHandler
from matplotlib.ticker import NullFormatter
from matplotlib.backends.backend_tkagg import FigureCanvasTkAgg
import matplotlib
sg.theme("Reddit")

#   import keys for Twitter API Elevated Access (personal)
keys = open('/Users/jerrywu/computer science/Final/login.txt','r')
lines = keys.readlines()
consumer_key = lines[0].rstrip()
consumer_secret = lines[1].rstrip()
access_token = lines[2].rstrip()
access_token_secret = lines[3].rstrip()
bearer_token = lines[4].rstrip()

#   authenticate to the twitter API
auth = OAuthHandler(consumer_key, consumer_secret)
auth.set_access_token(access_token, access_token_secret)
api = tweepy.API(auth)

#   create a client to collect tweets
Client = tweepy.Client(bearer_token=bearer_token,
                    consumer_key= consumer_key,
                    consumer_secret= consumer_secret,
                    access_token= access_token,
                    access_token_secret=access_token_secret)

#   set the display option of tweets to 100 characters & show all the rows of tweets collected
pd.options.display.max_colwidth = 100
pd.set_option('display.max_rows', 9999)
```

Fig 01. Import Libraries

Firstly, I installed and imported additional libraries to aid me with the program. Then I open a file in the same directory of the program to import the keys necessary for accessing the Twitter API. After reading the lines, I defined the variables for the tokens with only the value, neglecting the blank space on the right when a line is read. Then I authenticate with the Twitter API to create a client using the keys. Lastly, changing the maximum rows and length displayed in a data frame.

Homepage GUI

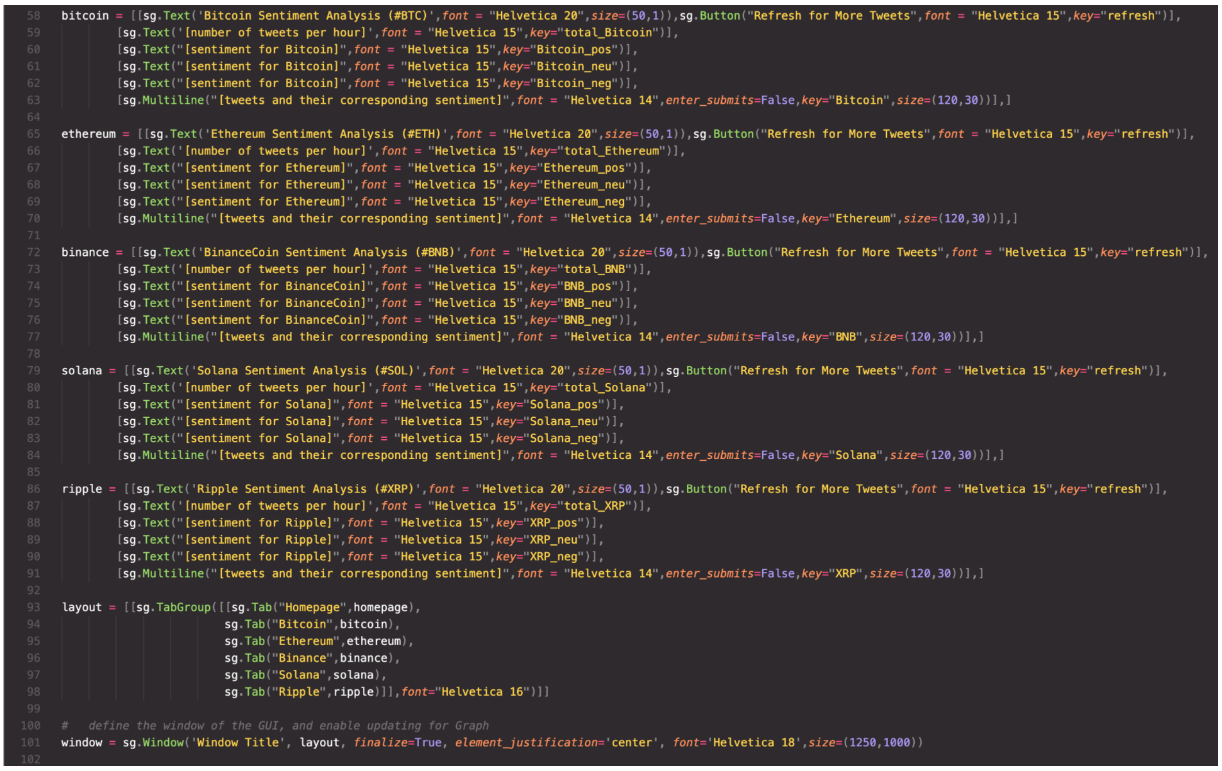


```python
#   layout of the GUI, where there is a homepage and 5 seperated pages for the 5 cryptocurrency
homepage = [[sg.Text('Twitter Sentiment Analysis on Cryptocurrency',font = "Helvetica 20",size=(50,1)),sg.Button("Refresh for More Tweets",font = "Helvetica 15",key="refre
            [sg.Text("Cryptocurrency just like any other investment involves risks, and has potential in losing all your money. (DYOR) 📈📈 ", font = "Helvetica 10")],
            [sg.Text("This application is not any financial advice, but an educational application to monitor the sentiment of Twitter users. ", font = "Helvetica 10")],
            [sg.Text("(Application) Discord/Telegram/Twitter → Crypto community", font = "Helvetica 10")],
            [sg.Text("(Website) CoinMarketCap → Live price indicator", font = "Helvetica 10")],
            [sg.Text("(Website) Coinbase → Exchanging currencies to cryptocurrencies", font = "Helvetica 10")],
            [sg.Text("Total tweets in an hour",font = "Helvetica 15",key="total_tweets")],
            [sg.Text("TOP #1 Crypto Currency",font = "Helvetica 15",key="T1")],
            [sg.Text("TOP #2 Crypto Currency",font = "Helvetica 15",key="T2")],
            [sg.Text("TOP #3 Crypto Currency",font = "Helvetica 15",key="T3")],
            [sg.Text("TOP #4 Crypto Currency",font = "Helvetica 15",key="T4")],
            [sg.Text("TOP #5 Crypto Currency",font = "Helvetica 15",key="T5")],
            [sg.Text("Tweets Collected",font = "Helvetica 15",key="collect")],
            [sg.Canvas(key='-CANVAS-')]]
```

Fig 02. Homepage Graphic User Interface

On this homepage, it will display some information about this program and cryptocurrencies, with additional tools. Then, it will display in the following line the ranking of the cryptocurrency, with a graph on the bottom showing the sentiments.

Creating Tab for the GUI

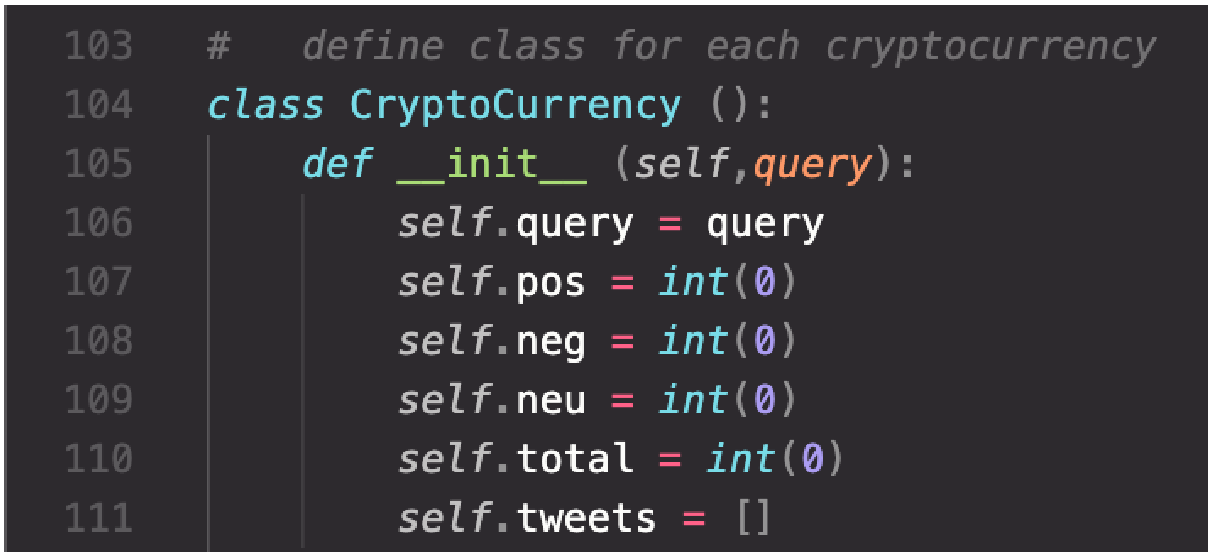


```python
bitcoin = [[sg.Text('Bitcoin Sentiment Analysis (#BTC)',font = "Helvetica 20",size=(50,1)),sg.Button("Refresh for More Tweets",font = "Helvetica 15",key="refresh")],
            [sg.Text('[number of tweets per hour]',font = "Helvetica 15",key="total_Bitcoin")],
            [sg.Text("[sentiment for Bitcoin]",font = "Helvetica 15",key="Bitcoin_pos")],
            [sg.Text("[sentiment for Bitcoin]",font = "Helvetica 15",key="Bitcoin_neu")],
            [sg.Text("[sentiment for Bitcoin]",font = "Helvetica 15",key="Bitcoin_neg")],
            [sg.Multiline("[tweets and their corresponding sentiment]",font = "Helvetica 14",enter_submits=False,key="Bitcoin",size=(120,30))],]

ethereum = [[sg.Text('Ethereum Sentiment Analysis (#ETH)',font = "Helvetica 20",size=(50,1)),sg.Button("Refresh for More Tweets",font = "Helvetica 15",key="refresh")],
            [sg.Text('[number of tweets per hour]',font = "Helvetica 15",key="total_Ethereum")],
            [sg.Text("[sentiment for Ethereum]",font = "Helvetica 15",key="Ethereum_pos")],
            [sg.Text("[sentiment for Ethereum]",font = "Helvetica 15",key="Ethereum_neu")],
            [sg.Text("[sentiment for Ethereum]",font = "Helvetica 15",key="Ethereum_neg")],
            [sg.Multiline("[tweets and their corresponding sentiment]",font = "Helvetica 14",enter_submits=False,key="Ethereum",size=(120,30))],]

binance = [[sg.Text('BinanceCoin Sentiment Analysis (#BNB)',font = "Helvetica 20",size=(50,1)),sg.Button("Refresh for More Tweets",font = "Helvetica 15",key="refresh")],
            [sg.Text('[number of tweets per hour]',font = "Helvetica 15",key="total_BNB")],
            [sg.Text("[sentiment for BinanceCoin]",font = "Helvetica 15",key="BNB_pos")],
            [sg.Text("[sentiment for BinanceCoin]",font = "Helvetica 15",key="BNB_neu")],
            [sg.Text("[sentiment for BinanceCoin]",font = "Helvetica 15",key="BNB_neg")],
            [sg.Multiline("[tweets and their corresponding sentiment]",font = "Helvetica 14",enter_submits=False,key="BNB",size=(120,30))],]

solana = [[sg.Text('Solana Sentiment Analysis (#SOL)',font = "Helvetica 20",size=(50,1)),sg.Button("Refresh for More Tweets",font = "Helvetica 15",key="refresh")],
            [sg.Text('[number of tweets per hour]',font = "Helvetica 15",key="total_Solana")],
            [sg.Text("[sentiment for Solana]",font = "Helvetica 15",key="Solana_pos")],
            [sg.Text("[sentiment for Solana]",font = "Helvetica 15",key="Solana_neu")],
            [sg.Text("[sentiment for Solana]",font = "Helvetica 15",key="Solana_neg")],
            [sg.Multiline("[tweets and their corresponding sentiment]",font = "Helvetica 14",enter_submits=False,key="Solana",size=(120,30))],]

ripple = [[sg.Text('Ripple Sentiment Analysis (#XRP)',font = "Helvetica 20",size=(50,1)),sg.Button("Refresh for More Tweets",font = "Helvetica 15",key="refresh")],
            [sg.Text('[number of tweets per hour]',font = "Helvetica 15",key="total_XRP")],
            [sg.Text("[sentiment for Ripple]",font = "Helvetica 15",key="XRP_pos")],
            [sg.Text("[sentiment for Ripple]",font = "Helvetica 15",key="XRP_neu")],
            [sg.Text("[sentiment for Ripple]",font = "Helvetica 15",key="XRP_neg")],
            [sg.Multiline("[tweets and their corresponding sentiment]",font = "Helvetica 14",enter_submits=False,key="XRP",size=(120,30))],]

layout = [[sg.TabGroup([[sg.Tab("Homepage",homepage),
                        sg.Tab("Bitcoin",bitcoin),
                        sg.Tab("Ethereum",ethereum),
                        sg.Tab("Binance",binance),
                        sg.Tab("Solana",solana),
                        sg.Tab("Ripple",ripple)]],font="Helvetica 16")]]

#   define the window of the GUI, and enable updating for Graph
window = sg.Window('Window Title', layout, finalize=True, element_justification='center', font='Helvetica 18',size=(1250,1000))
```

Fig 03. Tab of the five cryptocurrencies

In these layouts, each cryptocurrency has its own tab with a key to show the total tweets of that specific cryptocurrency, the number of sentiments, and all tweets with their corresponding sentiment. Moving on, we define the GUI window in the format of the layout variable.

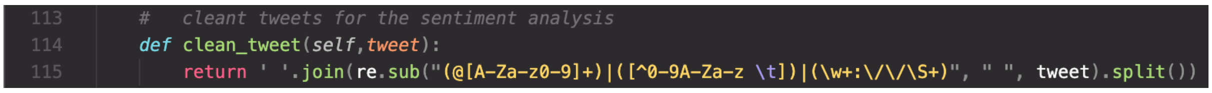
Cryptocurrency Class Object, Attributes & Function



```python
#   define class for each cryptocurrency
class CryptoCurrency ():
    def __init__ (self,query):
        self.query = query
        self.pos = int(0)
        self.neg = int(0)
        self.neu = int(0)
        self.total = int(0)
        self.tweets = []
```

Fig 04. Attributes of Class

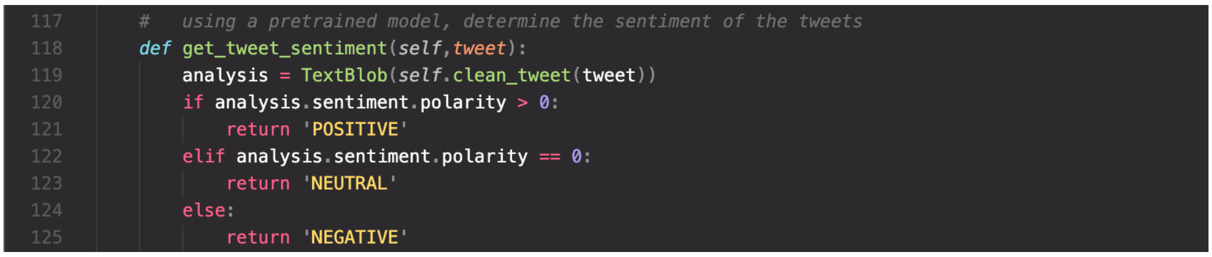
Here a class object is defined to be CryptoCurrency, and with the initial attributes of the query (The keyword to search), the number of positive, negative, and neutral tweets, total tweets posted in an hour, and a two-dimensional array of tweets for later tweet retrieval.



```python
#   cleant tweets for the sentiment analysis
def clean_tweet(self,tweet):
    return ' '.join(re.sub("(@[A-Za-z0-9]+)|([^0-9A-Za-z \t])|(\w+:\/\/\S+)", " ", tweet).split())
```

Fig 05. Clean Tweets

This function cleans the tweets for sentiment analysis. The function removes any link in the tweet, or special characters by using the regex statement[1], then the function returns the cleaned tweet for sentiment analysis.

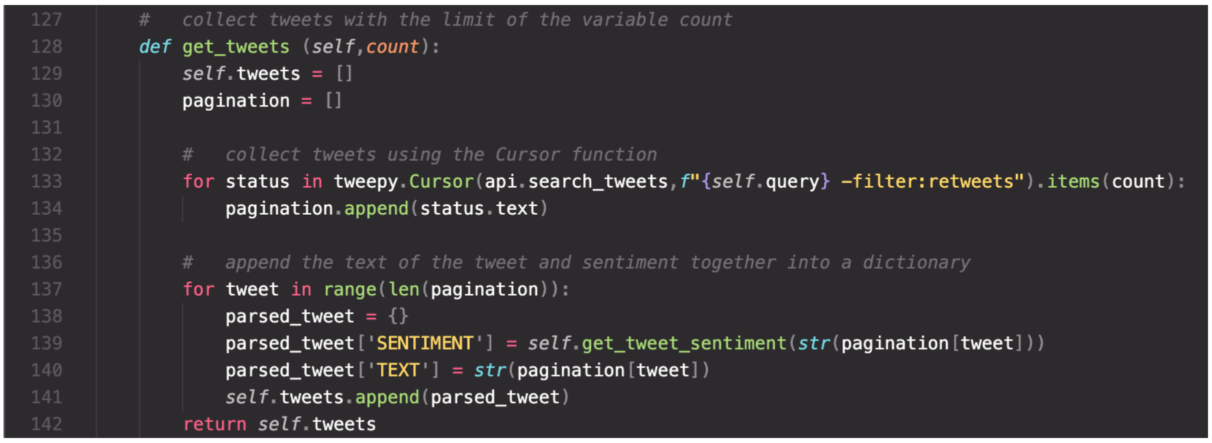


```python
#   using a pretrained model, determine the sentiment of the tweets
def get_tweet_sentiment(self,tweet):
    analysis = TextBlob(self.clean_tweet(tweet))
    if analysis.sentiment.polarity > 0:
        return 'POSITIVE'
    elif analysis.sentiment.polarity == 0:
        return 'NEUTRAL'
    else:
        return 'NEGATIVE'
```

Fig 06. Sentiment Analysis of Tweets

The function takes in a cleaned tweet and analyzes it using the external library TextBlob. The attribute of the analysis has an attribute sentiment, which is how we determine whether the sentiment is positive, neutral, or negative. In the end, returning the sentiment to the corresponding tweet.

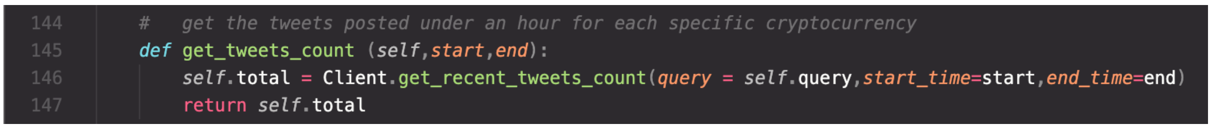
[1] "Regexr: Learn, Build, & Test Regex". 2022. *Regexr*. https://regexr.com/.



```python
    #   collect tweets with the limit of the variable count
    def get_tweets (self,count):
        self.tweets = []
        pagination = []

        #   collect tweets using the Cursor function
        for status in tweepy.Cursor(api.search_tweets,f"{self.query} -filter:retweets").items(count):
            pagination.append(status.text)

        #   append the text of the tweet and sentiment together into a dictionary
        for tweet in range(len(pagination)):
            parsed_tweet = {}
            parsed_tweet['SENTIMENT'] = self.get_tweet_sentiment(str(pagination[tweet]))
            parsed_tweet['TEXT'] = str(pagination[tweet])
            self.tweets.append(parsed_tweet)
        return self.tweets
```

Fig 07. Tweet Retrieval

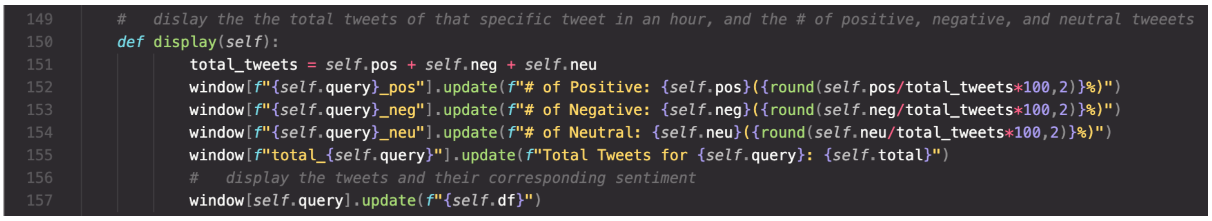
This function takes "count" as the parameter and using the Client and API variable created previously, the first for loop will repeat several times until the number of counts is reached. Since for each request, the limit of tweets collected is 100, hence collecting 2000 tweets would require the loop to run 20 times. Then for each of the tweet collected, the text is appended in a dictionary, where the tweet and corresponding sentiment is stored under the value of the keys, SENTIMENT, TEXT.



```python
    #   get the tweets posted under an hour for each specific cryptocurrency
    def get_tweets_count (self,start,end):
        self.total = Client.get_recent_tweets_count(query = self.query,start_time=start,end_time=end)
        return self.total
```

Fig 08. Total Amount of Tweets

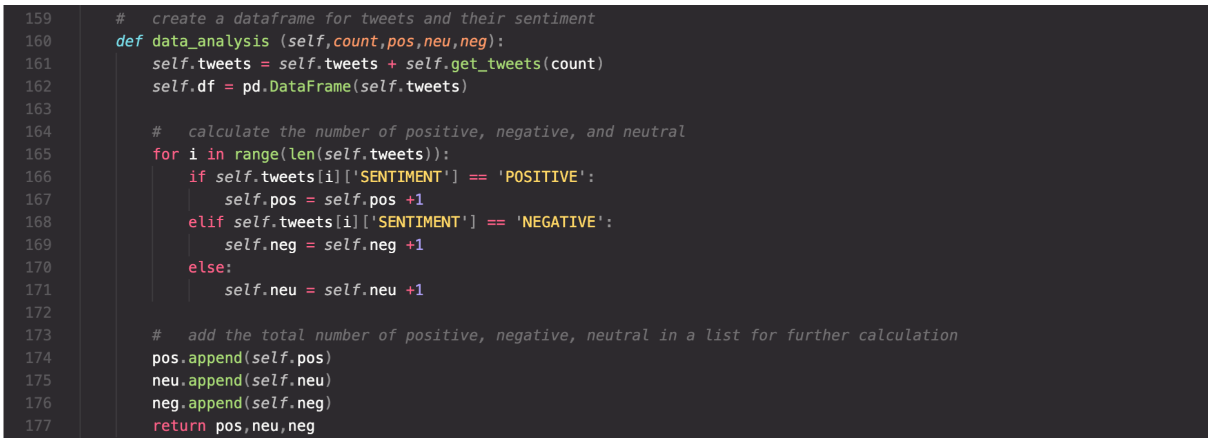
The function takes the start time and ends time as parameters, and through a function of the object Client, the function will return the total amount of tweets posted in an hour about that certain cryptocurrency.



```python
    #   dislay the the total tweets of that specific tweet in an hour, and the # of positive, negative, and neutral tweeets
    def display(self):
            total_tweets = self.pos + self.neg + self.neu
            window[f"{self.query}_pos"].update(f"# of Positive: {self.pos}({round(self.pos/total_tweets*100,2)}%)")
            window[f"{self.query}_neg"].update(f"# of Negative: {self.neg}({round(self.neg/total_tweets*100,2)}%)")
            window[f"{self.query}_neu"].update(f"# of Neutral: {self.neu}({round(self.neu/total_tweets*100,2)}%)")
            window[f"total_{self.query}"].update(f"Total Tweets for {self.query}: {self.total}")
            #   display the tweets and their corresponding sentiment
            window[self.query].update(f"{self.df}")
```

Fig 09. Display information in GUI

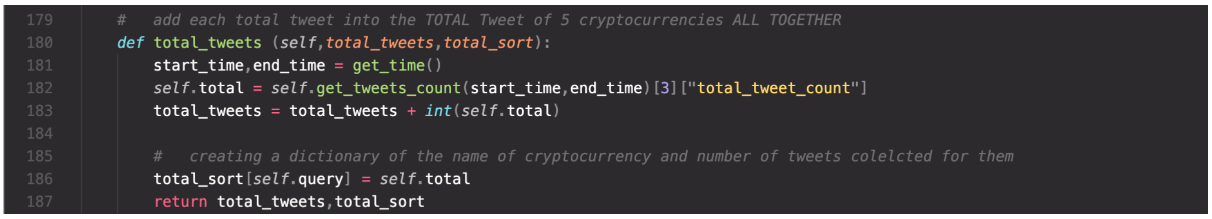
This function computes the total tweets collected by the Twitter API and displays the sentiment and its percentage of sentiment. Then the function displays the total tweets posted in an hour. Lastly, it displays the data frame created by the dictionary of tweets under the main function (Essentially the tweet and the corresponding sentiment).



```python
    #   create a dataframe for tweets and their sentiment
    def data_analysis (self,count,pos,neu,neg):
        self.tweets = self.tweets + self.get_tweets(count)
        self.df = pd.DataFrame(self.tweets)

        #   calculate the number of positive, negative, and neutral
        for i in range(len(self.tweets)):
            if self.tweets[i]['SENTIMENT'] == 'POSITIVE':
                self.pos = self.pos +1
            elif self.tweets[i]['SENTIMENT'] == 'NEGATIVE':
                self.neg = self.neg +1
            else:
                self.neu = self.neu +1

        #   add the total number of positive, negative, neutral in a list for further calculation
        pos.append(self.pos)
        neu.append(self.neu)
        neg.append(self.neg)
        return pos,neu,neg
```

Fig 10. Summing Sentiments

The data analysis function calls the function to get tweets, and store the tweets to create a data frame using the Pandas library. Then it counts the number of positive, negative, and neutral tweets to return the value in an array.

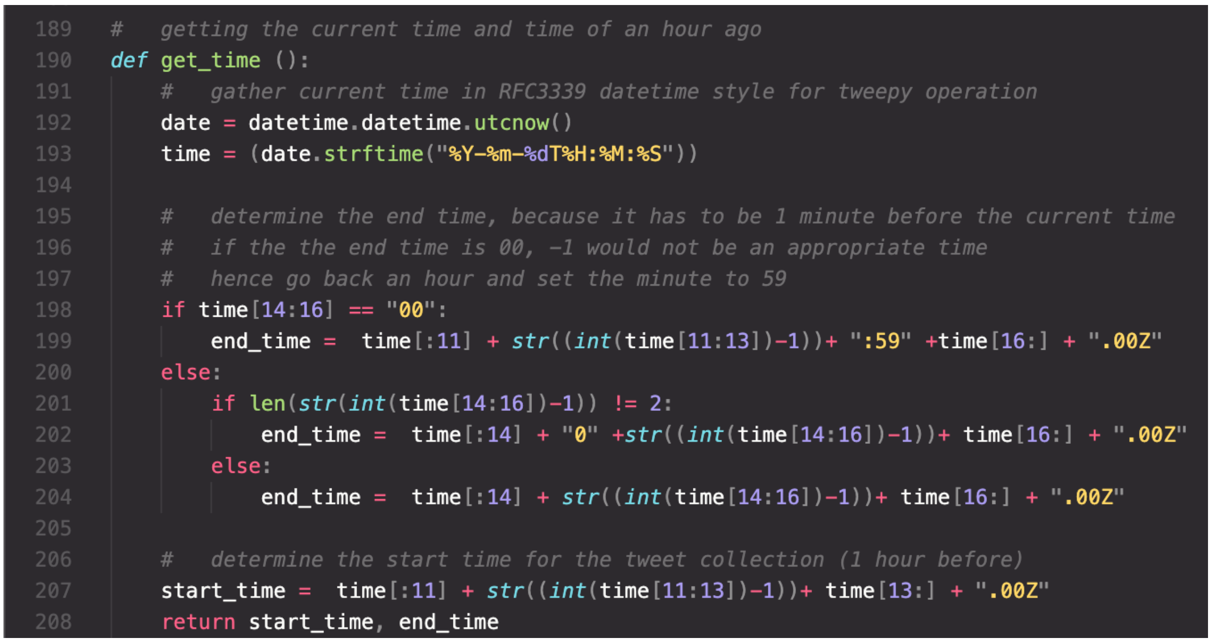


```python
    #   add each total tweet into the TOTAL Tweet of 5 cryptocurrencies ALL TOGETHER
    def total_tweets (self,total_tweets,total_sort):
        start_time,end_time = get_time()
        self.total = self.get_tweets_count(start_time,end_time)[3]["total_tweet_count"]
        total_tweets = total_tweets + int(self.total)

        #   creating a dictionary of the name of cryptocurrency and number of tweets colelcted for them
        total_sort[self.query] = self.total
        return total_tweets,total_sort
```

Fig 11. Total Tweets for All five Cryptocurrencies

This function collects the start time and ends time in order to call the function that computes the total amount of tweets posted in an hour. The function then stores the integer value using the index and calling the key of the dictionary. Then for the five cryptocurrencies combined, the function stores it in a separate variable that gets passed in as a parameter and returned as a value. The function also creates a dictionary where the name of cryptocurrency is the key, and the number of tweets collected is the value of the dictionary, this dictionary is then returned as well.

Function - Get Time



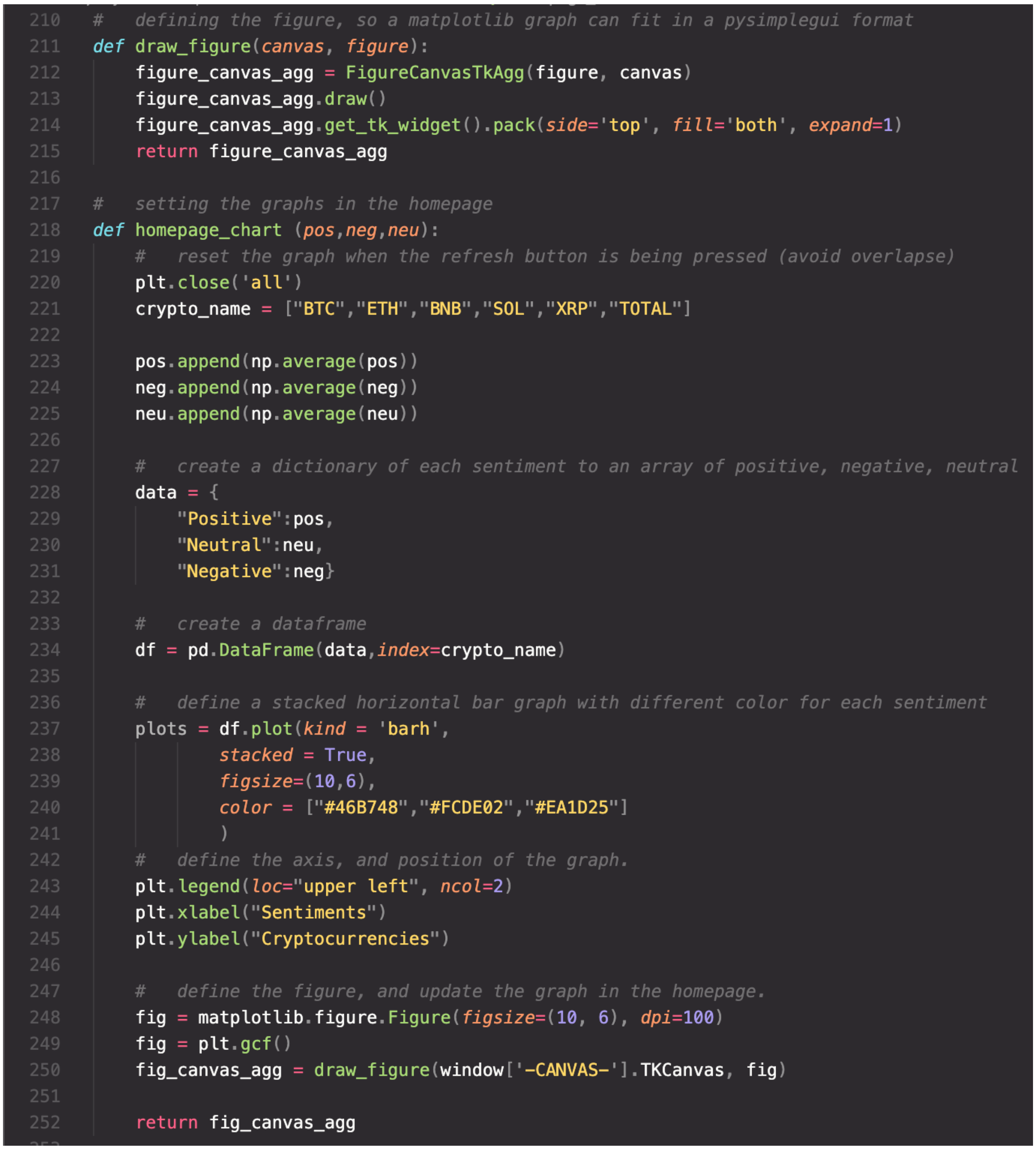
```python
#   getting the current time and time of an hour ago
def get_time ():
    #   gather current time in RFC3339 datetime style for tweepy operation
    date = datetime.datetime.utcnow()
    time = (date.strftime("%Y-%m-%dT%H:%M:%S"))

    #   determine the end time, because it has to be 1 minute before the current time
    #   if the the end time is 00, -1 would not be an appropriate time
    #   hence go back an hour and set the minute to 59
    if time[14:16] == "00":
        end_time =  time[:11] + str((int(time[11:13])-1))+ ":59" +time[16:] + ".00Z"
    else:
        if len(str(int(time[14:16])-1)) != 2:
            end_time =  time[:14] + "0" +str((int(time[14:16])-1))+ time[16:] + ".00Z"
        else:
            end_time =  time[:14] + str((int(time[14:16])-1))+ time[16:] + ".00Z"

    #   determine the start time for the tweet collection (1 hour before)
    start_time =  time[:11] + str((int(time[11:13])-1))+ time[13:] + ".00Z"
    return start_time, end_time
```

Fig 12. Start Time and End Time Calculation

The time is collected in the format of RFC3339[2] DateTime style for the purpose of Twitter API operation. Since this function is used to discover the number of tweets posted in an hour, then the start time should be an hour before the time the program is executed. The end time has to be a minute before the current time. By replacing characters of indexes, converting numbers from string to integer, and back to a string, I was able to manually change the characters of minutes and hours. There was an exception when the minute is equal to 0, and subtracting by one would not give us an appropriate time, hence the if statement ensures the end time is valid.

[2] "Understanding About RFC 3339 For Datetime Formatting In Software Engineering". 2019. *Medium*. https://medium.com/easyread/understanding-about-rfc-3339-for-datetime-formatting-in-software-engineering-940aa5d5f68a.

Function - Plot Graph

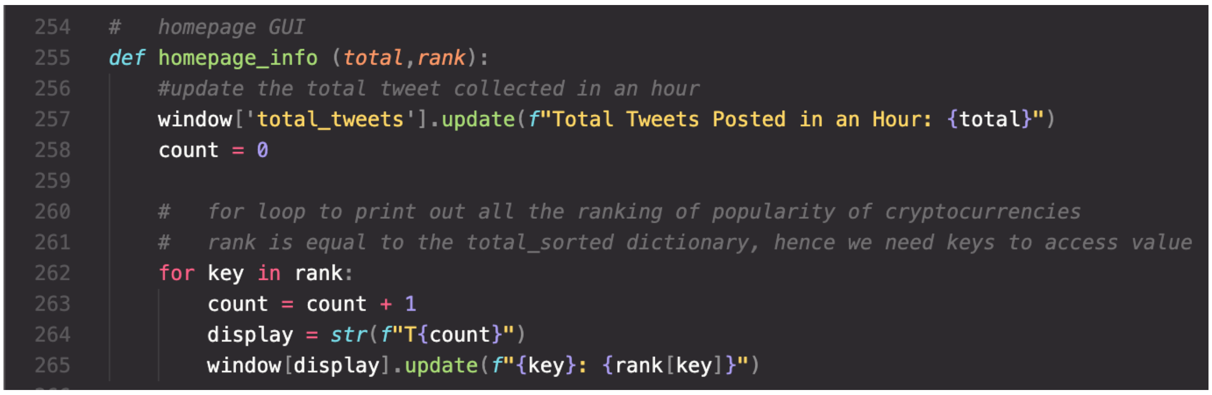


```python
#   defining the figure, so a matplotlib graph can fit in a pysimplegui format
def draw_figure(canvas, figure):
    figure_canvas_agg = FigureCanvasTkAgg(figure, canvas)
    figure_canvas_agg.draw()
    figure_canvas_agg.get_tk_widget().pack(side='top', fill='both', expand=1)
    return figure_canvas_agg

#   setting the graphs in the homepage
def homepage_chart (pos,neg,neu):
    #   reset the graph when the refresh button is being pressed (avoid overlapse)
    plt.close('all')
    crypto_name = ["BTC","ETH","BNB","SOL","XRP","TOTAL"]

    pos.append(np.average(pos))
    neg.append(np.average(neg))
    neu.append(np.average(neu))

    #   create a dictionary of each sentiment to an array of positive, negative, neutral
    data = {
        "Positive":pos,
        "Neutral":neu,
        "Negative":neg}

    #   create a dataframe
    df = pd.DataFrame(data,index=crypto_name)

    #   define a stacked horizontal bar graph with different color for each sentiment
    plots = df.plot(kind = 'barh',
            stacked = True,
            figsize=(10,6),
            color = ["#46B748","#FCDE02","#EA1D25"]
            )
    #   define the axis, and position of the graph.
    plt.legend(loc="upper left", ncol=2)
    plt.xlabel("Sentiments")
    plt.ylabel("Cryptocurrencies")

    #   define the figure, and update the graph in the homepage.
    fig = matplotlib.figure.Figure(figsize=(10, 6), dpi=100)
    fig = plt.gcf()
    fig_canvas_agg = draw_figure(window['-CANVAS-'].TKCanvas, fig)

    return fig_canvas_agg
```

Fig 13. Plotting Graph

These two functions are used to graph a stacked horizontal bar graph, by taking the parameter of sentiments, it creates a dictionary of each sentiment which then is translated into a data frame. With the matplotlib library, I defined the type of graph and the labels of the graph. In order to display the graph, I had to create a canvas object according to the PySimpleGUI Documented demo program.

Function - Display information on the Homepage

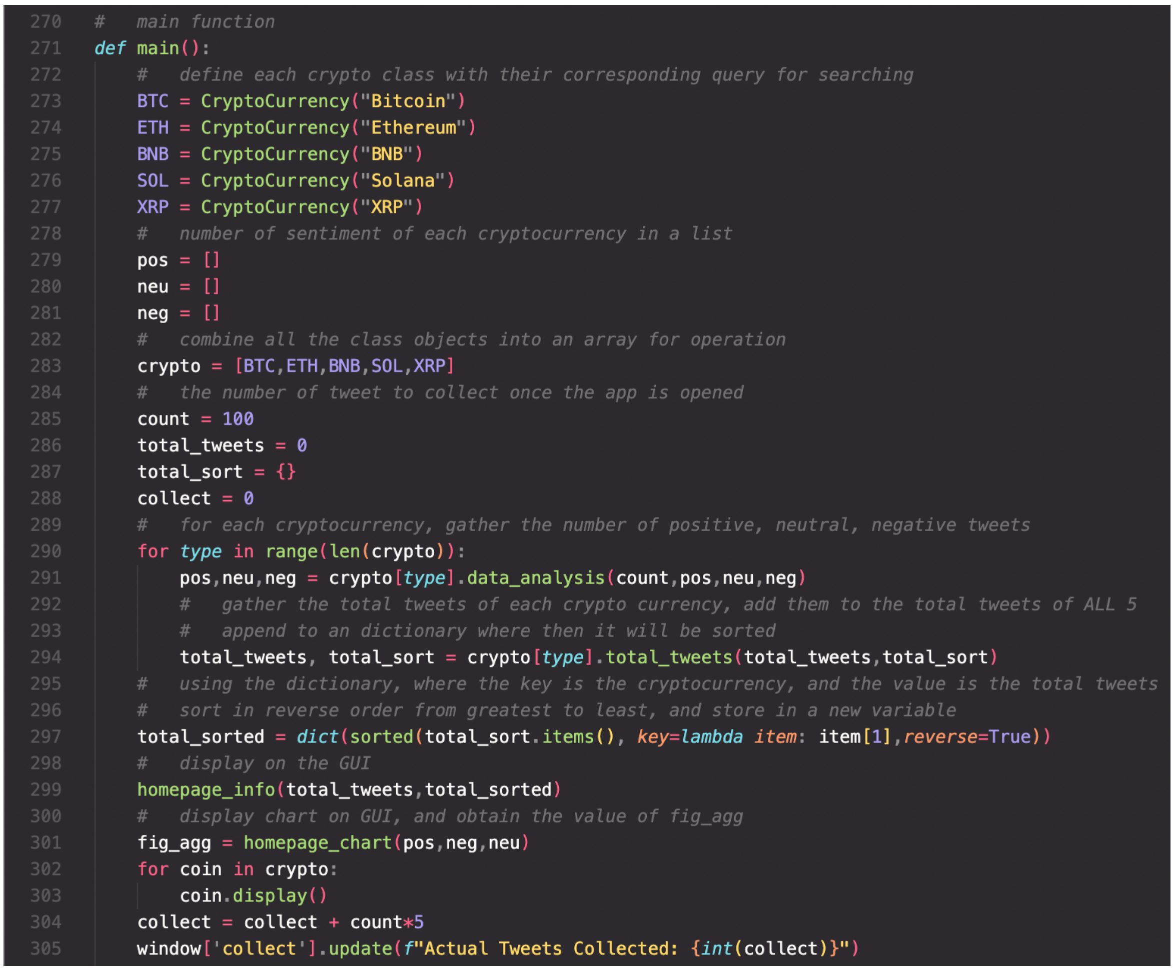


```python
#   homepage GUI
def homepage_info (total,rank):
    #update the total tweet collected in an hour
    window['total_tweets'].update(f"Total Tweets Posted in an Hour: {total}")
    count = 0

    #   for loop to print out all the ranking of popularity of cryptocurrencies
    #   rank is equal to the total_sorted dictionary, hence we need keys to access value
    for key in rank:
        count = count + 1
        display = str(f"T{count}")
        window[display].update(f"{key}: {rank[key]}")
```

Fig 14. Display Homepage Information

Firstly, the function takes the dictionary when the popularity of tweets and the name is sorted. Secondly, the function updates the total number of tweets posted about cryptocurrency. Lastly, the loop repeats 5 times, and for each time the window corresponding to the ranking is updated into displaying the cryptocurrency, and the number of tweets collected for that cryptocurrency from greatest to smallest.

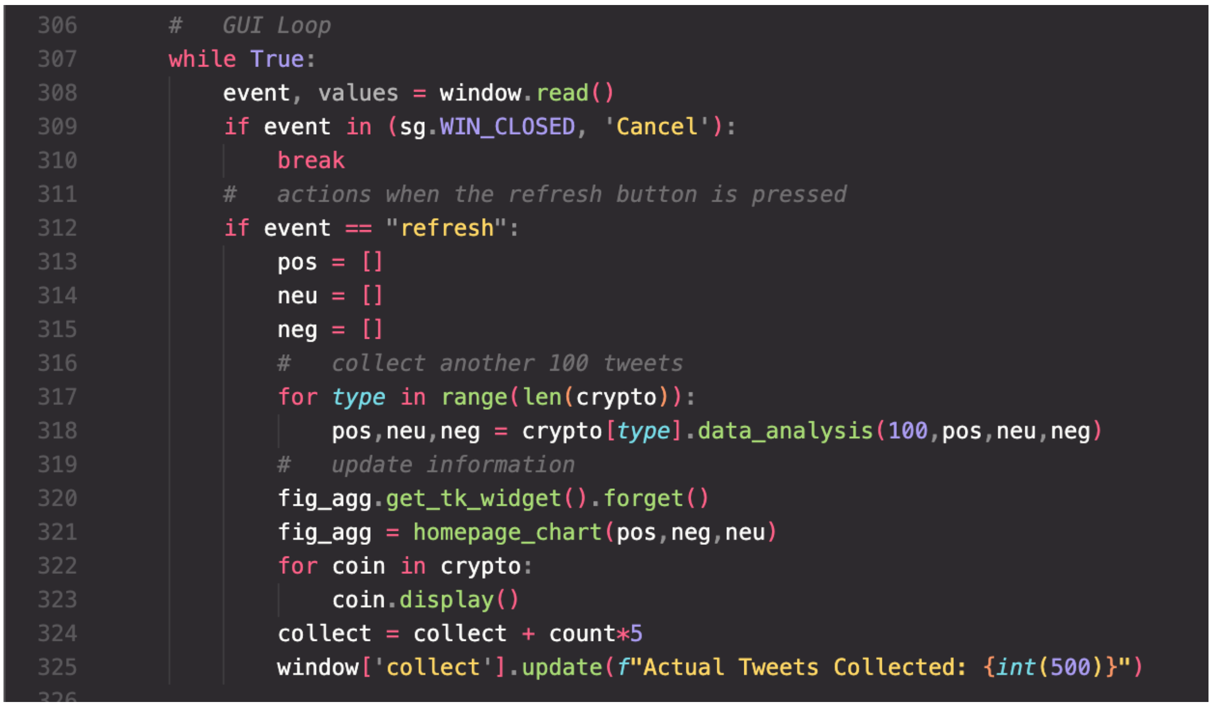
Main function



```python
#   main function
def main():
    #   define each crypto class with their corresponding query for searching
    BTC = CryptoCurrency("Bitcoin")
    ETH = CryptoCurrency("Ethereum")
    BNB = CryptoCurrency("BNB")
    SOL = CryptoCurrency("Solana")
    XRP = CryptoCurrency("XRP")
    #   number of sentiment of each cryptocurrency in a list
    pos = []
    neu = []
    neg = []
    #   combine all the class objects into an array for operation
    crypto = [BTC,ETH,BNB,SOL,XRP]
    #   the number of tweet to collect once the app is opened
    count = 100
    total_tweets = 0
    total_sort = {}
    collect = 0
    #   for each cryptocurrency, gather the number of positive, neutral, negative tweets
    for type in range(len(crypto)):
        pos,neu,neg = crypto[type].data_analysis(count,pos,neu,neg)
        #   gather the total tweets of each crypto currency, add them to the total tweets of ALL 5
        #   append to an dictionary where then it will be sorted
        total_tweets, total_sort = crypto[type].total_tweets(total_tweets,total_sort)
    #   using the dictionary, where the key is the cryptocurrency, and the value is the total tweets
    #   sort in reverse order from greatest to least, and store in a new variable
    total_sorted = dict(sorted(total_sort.items(), key=lambda item: item[1],reverse=True))
    #   display on the GUI
    homepage_info(total_tweets,total_sorted)
    #   display chart on GUI, and obtain the value of fig_agg
    fig_agg = homepage_chart(pos,neg,neu)
    for coin in crypto:
        coin.display()
    collect = collect + count*5
    window['collect'].update(f"Actual Tweets Collected: {int(collect)}")
```

Fig 15. Main function

The main function contains the GUI loop but also brings the computed values together to display operations. Firstly, I assigned the five cryptocurrencies in the class CryptoCurrency and created lists for the sentiment of each cryptocurrency. To add on, I created a list to include all the five class objects. Secondly, in the for loop, each cryptocurrency will collect the sentiments, and the total tweets collected. Thirdly, the main function sorts the dictionary where the keys and values are the names of cryptocurrencies with the number of tweets posted in an hour, this is done by using the sorted function and an inline function, lambda, running 5 times to sort the function into a dictionary from greatest to least. Then this information is displayed in the function mentioned previously.



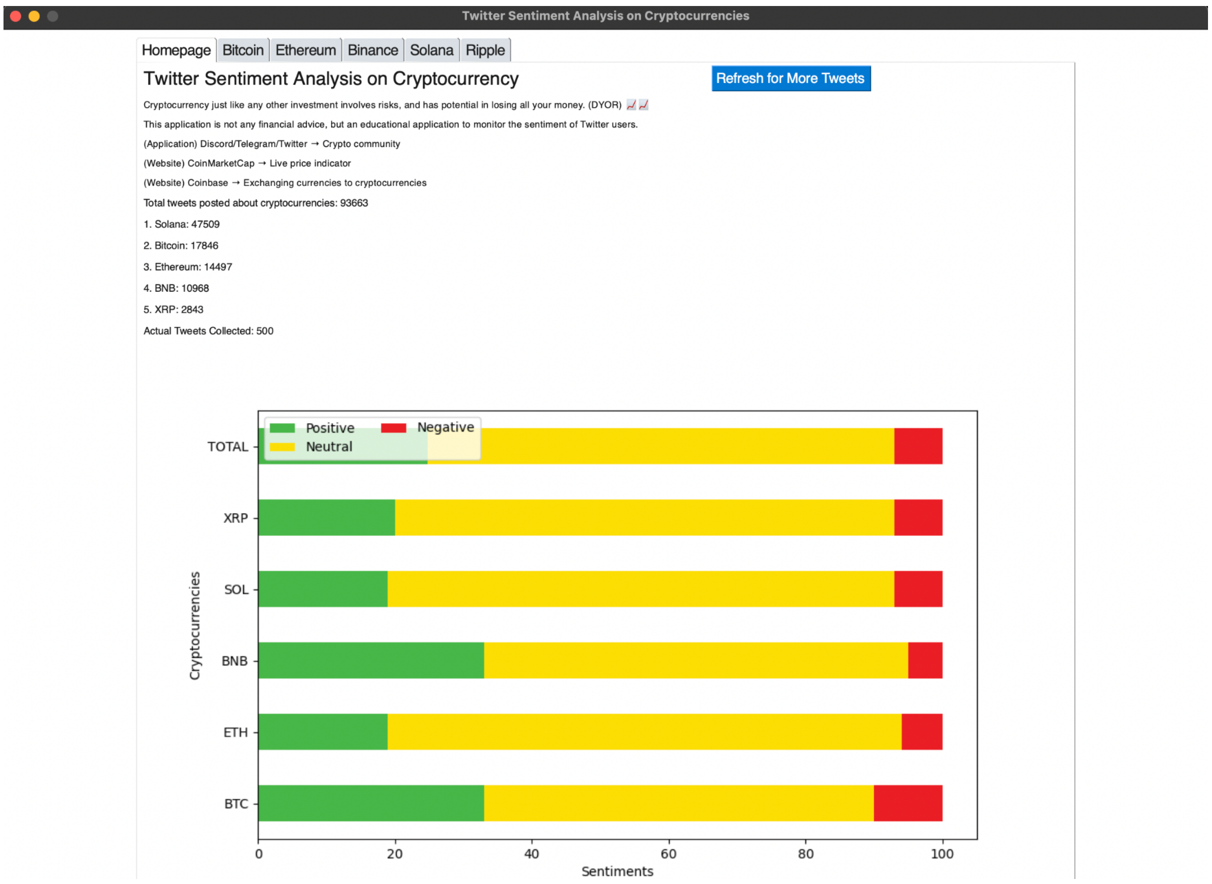
```
    #   GUI Loop
    while True:
        event, values = window.read()
        if event in (sg.WIN_CLOSED, 'Cancel'):
            break
        #   actions when the refresh button is pressed
        if event == "refresh":
            pos = []
            neu = []
            neg = []
            #   collect another 100 tweets
            for type in range(len(crypto)):
                pos,neu,neg = crypto[type].data_analysis(100,pos,neu,neg)
            #   update information
            fig_agg.get_tk_widget().forget()
            fig_agg = homepage_chart(pos,neg,neu)
            for coin in crypto:
                coin.display()
            collect = collect + count*5
            window['collect'].update(f"Actual Tweets Collected: {int(500)}")
```

Fig 16. Graphic User Interface Loop

In this graphic user interface loop, the while loops run indefinitely. When the refresh button is pressed, the count for sentiment is reset, and for each cryptocurrency, the program collects another 100 tweets and updates the graph based on the new amount of sentiment.

<u>**Word Count: 1110**</u>

Final Product



Twitter Sentiment Analysis on Cryptocurrency

Cryptocurrency just like any other investment involves risks, and has potential in losing all your money. (DYOR) 📈📈

This application is not any financial advice, but an educational application to monitor the sentiment of Twitter users.

(Application) Discord/Telegram/Twitter → Crypto community

(Website) CoinMarketCap → Live price indicator

(Website) Coinbase → Exchanging currencies to cryptocurrencies

Total tweets posted about cryptocurrencies: 93663

1. Solana: 47509

2. Bitcoin: 17846

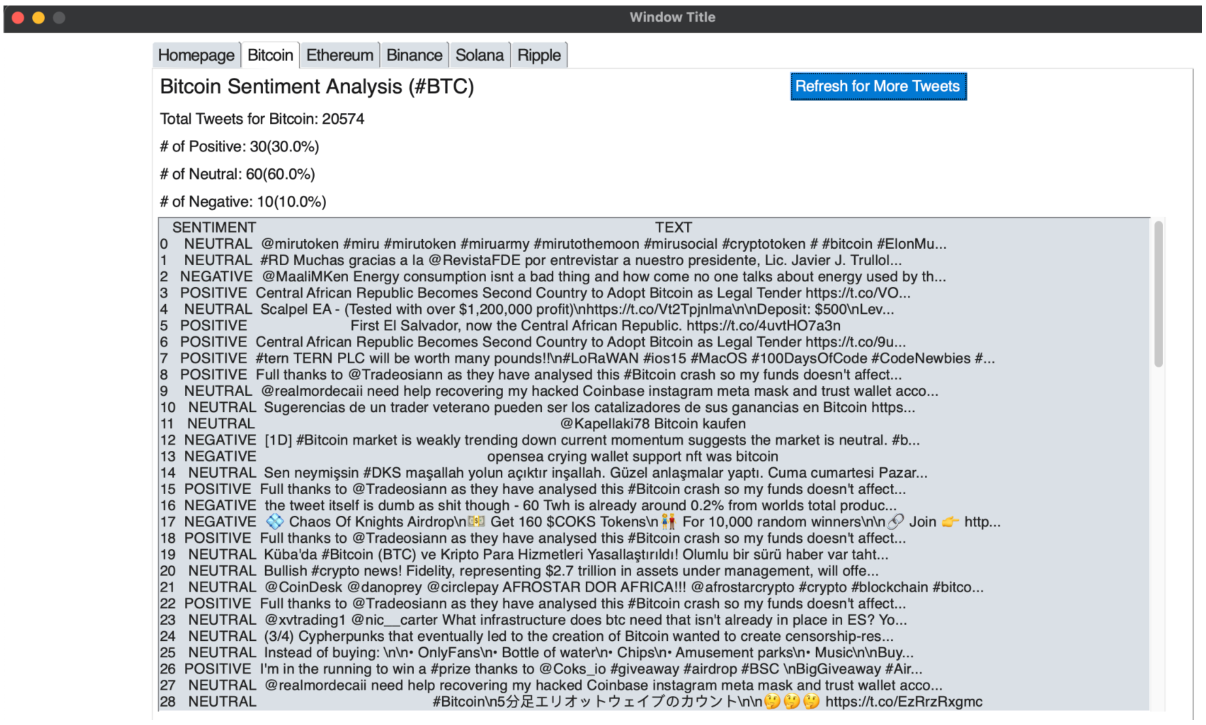
3. Ethereum: 14497

4. BNB: 10968

5. XRP: 2843

Actual Tweets Collected: 500

Fig 17. Homepage



Bitcoin Sentiment Analysis (#BTC)

Total Tweets for Bitcoin: 20574

# of Positive: 30(30.0%)

# of Neutral: 60(60.0%)

# of Negative: 10(10.0%)

```
   SENTIMENT                                              TEXT
0    NEUTRAL  @mirutoken #miru #mirutoken #miruarmy #mirutothemoon #mirusocial #cryptotoken # #bitcoin #ElonMu...
1    NEUTRAL  #RD Muchas gracias a la @RevistaFDE por entrevistar a nuestro presidente, Lic. Javier J. Trullol...
2   NEGATIVE  @MaaliMKen Energy consumption isnt a bad thing and how come no one talks about energy used by th...
3   POSITIVE  Central African Republic Becomes Second Country to Adopt Bitcoin as Legal Tender https://t.co/VO...
4    NEUTRAL  Scalpel EA - (Tested with over $1,200,000 profit)\nhttps://t.co/Vt2Tpjnlma\n\nDeposit: $500\nLev...
5   POSITIVE                    First El Salvador, now the Central African Republic. https://t.co/4uvtHO7a3n
6   POSITIVE  Central African Republic Becomes Second Country to Adopt Bitcoin as Legal Tender https://t.co/9u...
7   POSITIVE  #tern TERN PLC will be worth many pounds!!\n#LoRaWAN #ios15 #MacOS #100DaysOfCode #CodeNewbies #...
8   POSITIVE  Full thanks to @Tradeosiann as they have analysed this #Bitcoin crash so my funds doesn't affect...
9    NEUTRAL  @realmordecaii need help recovering my hacked Coinbase instagram meta mask and trust wallet acco...
10   NEUTRAL  Sugerencias de un trader veterano pueden ser los catalizadores de sus ganancias en Bitcoin https...
11   NEUTRAL                                                          @Kapellaki78 Bitcoin kaufen
12  NEGATIVE  [1D] #Bitcoin market is weakly trending down current momentum suggests the market is neutral. #b...
13  NEGATIVE                                       opensea crying wallet support nft was bitcoin
14   NEUTRAL  Sen neymişsin #DKS maşallah yolun açıktır inşallah. Güzel anlaşmalar yaptı. Cuma cumartesi Pazar...
15  POSITIVE  Full thanks to @Tradeosiann as they have analysed this #Bitcoin crash so my funds doesn't affect...
16  NEGATIVE  the tweet itself is dumb as shit though - 60 Twh is already around 0.2% from worlds total produc...
17  NEGATIVE  💠 Chaos Of Knights Airdrop\n💵 Get 160 $COKS Tokens\n👫 For 10,000 random winners\n\n🔗 Join 👉 http...
18  POSITIVE  Full thanks to @Tradeosiann as they have analysed this #Bitcoin crash so my funds doesn't affect...
19   NEUTRAL  Küba'da #Bitcoin (BTC) ve Kripto Para Hizmetleri Yasallaştırıldı! Olumlu bir sürü haber var taht...
20   NEUTRAL  Bullish #crypto news! Fidelity, representing $2.7 trillion in assets under management, will offe...
21   NEUTRAL  @CoinDesk @danoprey @circlepay AFROSTAR DOR AFRICA!!! @afrostarcrypto #crypto #blockchain #bitco...
22  POSITIVE  Full thanks to @Tradeosiann as they have analysed this #Bitcoin crash so my funds doesn't affect...
23   NEUTRAL  @xvtrading1 @nic__carter What infrastructure does btc need that isn't already in place in ES? Yo...
24   NEUTRAL  (3/4) Cypherpunks that eventually led to the creation of Bitcoin wanted to create censorship-res...
25   NEUTRAL  Instead of buying: \n\n• OnlyFans\n• Bottle of water\n• Chips\n• Amusement parks\n• Music\n\nBuy...
26  POSITIVE  I'm in the running to win a #prize thanks to @Coks_io #giveaway #airdrop #BSC \nBigGiveaway #Air...
27   NEUTRAL  @realmordecaii need help recovering my hacked Coinbase instagram meta mask and trust wallet acco...
28   NEUTRAL                          #Bitcoin\n5分足エリオットウェイブのカウント\n\n🤔🤔🤔 https://t.co/EzRrzRxgmc
```

Fig 18. Tab for Bitcoin

Bibliography

"Tweepy Documentation — Tweepy 4.8.0 Documentation". 2022. *Docs.Tweepy.Org*. https://docs.tweepy.org/en/stable/.

"Tutorials — Matplotlib 3.5.1 Documentation". 2022. *Matplotlib.Org*. https://matplotlib.org/stable/tutorials/index.html.

"Pysimplegui". 2022. *Pysimplegui.Readthedocs.Io*. https://pysimplegui.readthedocs.io/en/latest/.

"Twitter Sentiment Analysis Using Python - Geeksforgeeks". 2017. Geeksforgeeks. https://www.geeksforgeeks.org/twitter-sentiment-analysis-using-python/.

"Python Lambda". 2022. W3schools.Com. https://www.w3schools.com/python/python_lambda.asp.

"Python Regex (With Examples)". 2022. Programiz.Com. https://www.programiz.com/python-programming/regex.

"Twitter_Sentiment_Analysis/Twittersentimentalanalize.Py At Master Nagasanjay/Twitter_Sentiment_Analysis". 2022. Github. https://github.com/nagasanjay/twitter_sentiment_analysis/blob/master/TwitterSentimentalAnalize.py.

"Regexr: Learn, Build, & Test Regex". 2022. *Regexr*. https://regexr.com/.

"Understanding About RFC 3339 For Datetime Formatting In Software Engineering". 2019. *Medium*. https://medium.com/easyread/understanding-about-rfc-3339-for-datetime-formatting-in-software-engineering-940aa5d5f68a.