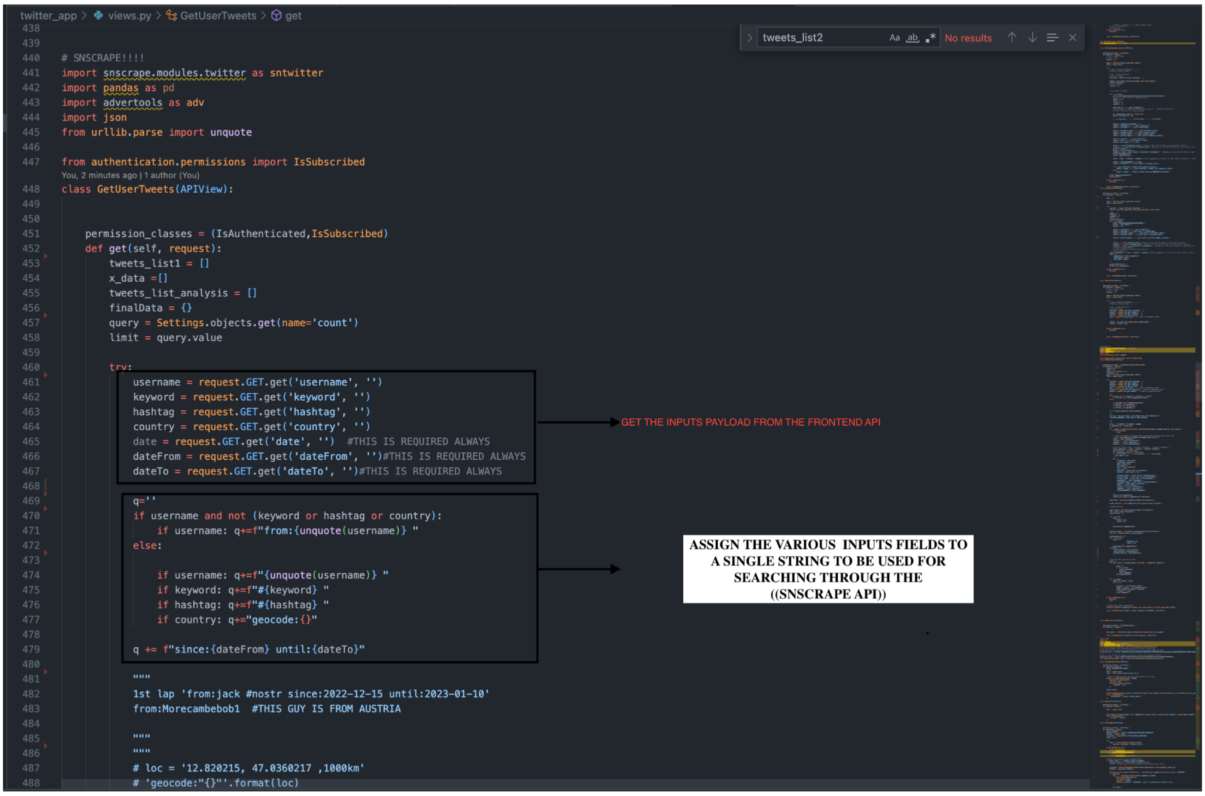
- This is the class being used to get the twitter data GetUserTweets
- HERE IS A VISUAL REPRESENTATION OF THE WAY ITS FETCHING DATA AND GETTING TO RETURN IT FOR THE FRONT END
- URL :: /api/twit/user-analysis-specific
-

1)

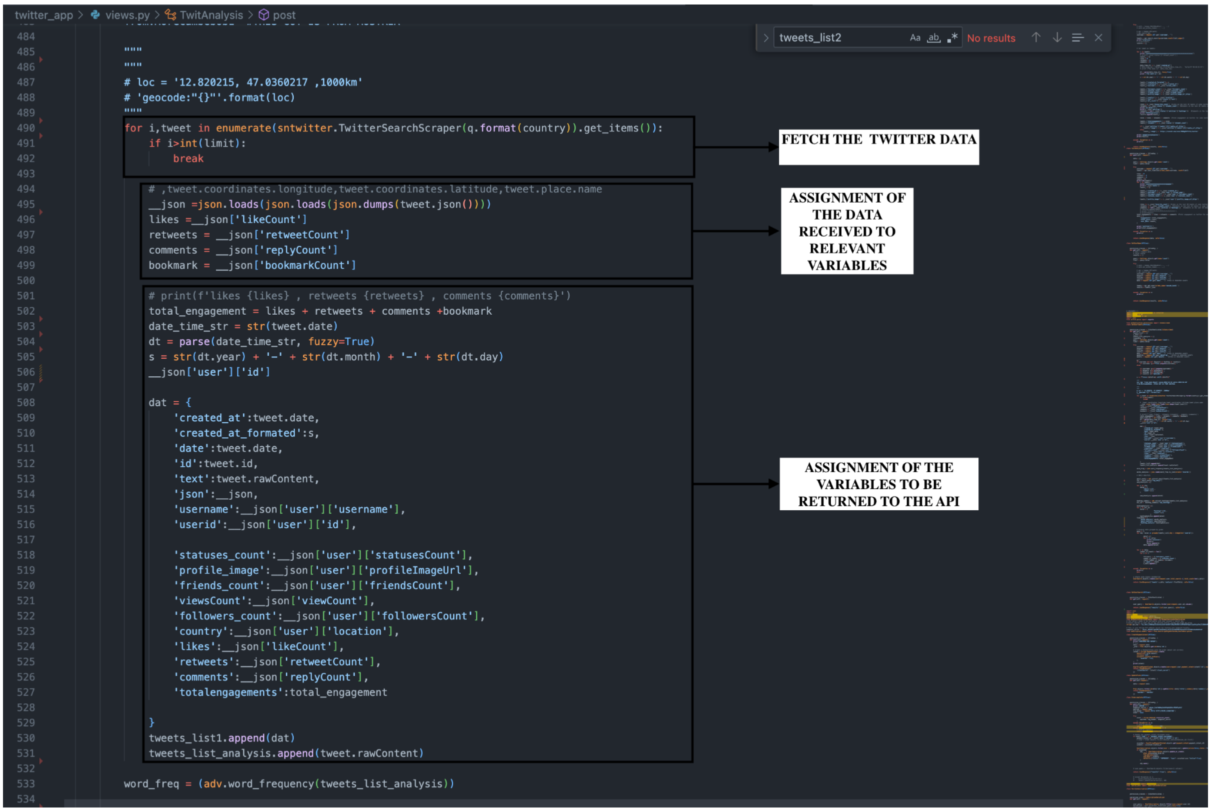

twitter_app > views.py > GetUserTweets > get
tweets_list2
Aa
ab
.*
No results
You, 2 minutes ago | 1 author (You)

```python
# SNSCRAPE!!!!
import snscrape.modules.twitter as sntwitter
import pandas as pd
import advertools as adv
import json
from urllib.parse import unquote

from authentication.permissions import IsSubscribed
class GetUserTweets(APIView):


    permission_classes = (IsAuthenticated,IsSubscribed)
    def get(self, request):
        tweets_list1 = []
        x_data =[]
        tweets_list_analysis = []
        finalData = {}
        query = Settings.objects.get(name='count')
        limit = query.value

        try:
            username = request.GET.get('username', '')
            keyword = request.GET.get('keyword', '')
            hashtag = request.GET.get('hashtag', '')
            country = request.GET.get('country', '')
            date = request.GET.get('date', '')  #THIS IS REQUIRED ALWAYS
            dateFrom = request.GET.get('dateFrom', '')#THIS IS REQUIRED ALWAYS
            dateTo = request.GET.get('dateTo', '')#THIS IS REQUIRED ALWAYS

            q=''
            if username and not (keyword or hashtag or country):
                if username: q+=f"from:{unquote(username)} "
            else:

                if username: q+=f"{unquote(username)} "
                if keyword: q+=f"#{keyword} "
                if hashtag: q+=f"#{hashtag} "
                if country: q+="geocode:{}"

            q += f"since:{dateFrom} until:{dateTo}"

            """
            1st lap 'from:jack #nostr since:2022-12-15 until:2023-01-10'
            from:Morecambebob1  #THIS GUY IS FROM AUSTRIA

            """
            """
            # loc = '12.820215, 47.0360217 ,1000km'
            # 'geocode:"{}"'.format(loc)
```

GET THE INPUTS PAYLOAD FROM THE FRONTEND API

ASSIGN THE VARIOUS INPUTS FIELDS TO A SINGLE STRING TO BE USED FOR SEARCHING THROUGH THE ((SNSCRAPE API))

2)



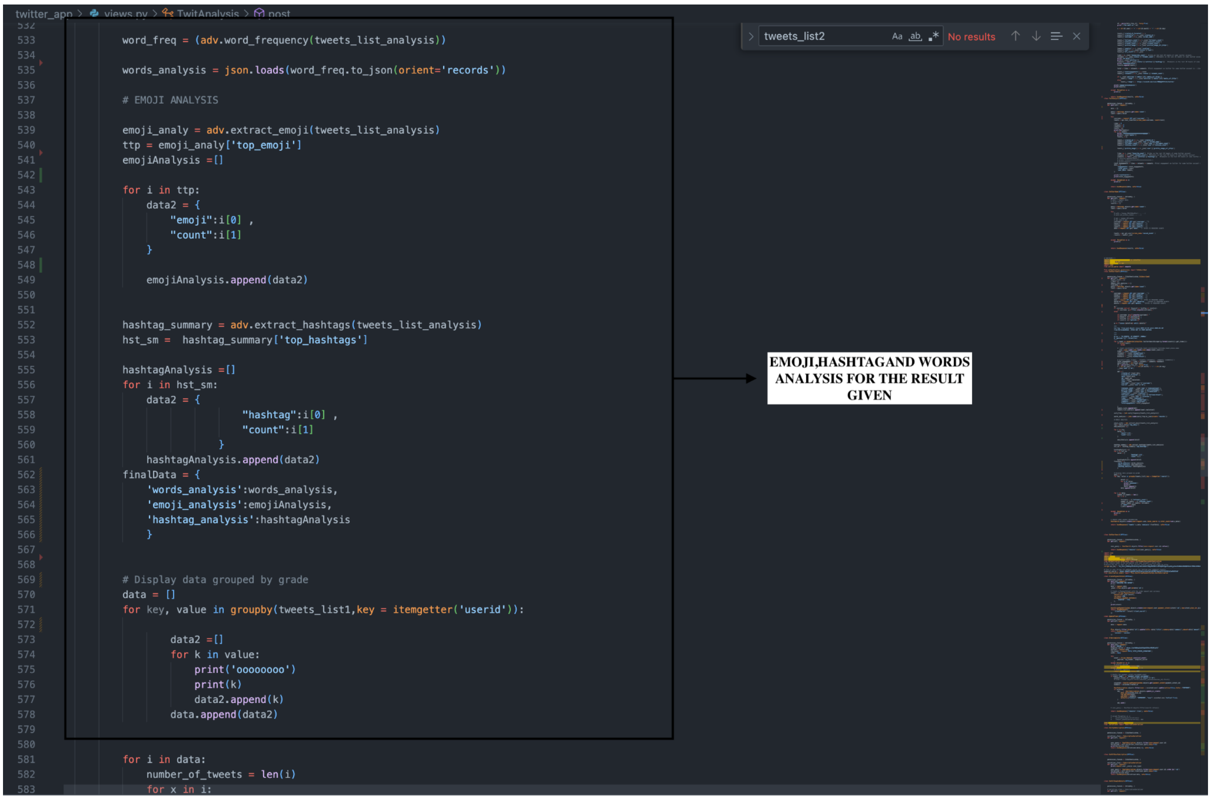
```python
        """
        """
        # loc = '12.820215, 47.0360217 ,1000km'
        # 'geocode:"{}"'.format(loc)
        """
        for i,tweet in enumerate(sntwitter.TwitterSearchScraper(q.format(country)).get_items()):
            if i>int(limit):
                break

            # ,tweet.coordinates.longitude,tweet.coordinates.latitude,tweet.place.name
            __json =json.loads(json.loads(json.dumps(tweet.json())))
            likes =__json['likeCount']
            retweets = __json['retweetCount']
            comments = __json['replyCount']
            bookmark = __json['bookmarkCount']

            # print(f'likes {likes} , retweets {retweets} , comments {comments}')
            total_engagement = likes + retweets + comments +bookmark
            date_time_str = str(tweet.date)
            dt = parse(date_time_str, fuzzy=True)
            s = str(dt.year) + '-' + str(dt.month) + '-' + str(dt.day)
            __json['user']['id']

            dat = {
                'created_at':tweet.date,
                'created_at_formated':s,
                'date':tweet.date,
                'id':tweet.id,
                'text':tweet.rawContent,
                'json':__json,
                'username':__json['user']['username'],
                'userid':__json['user']['id'],

                'statuses_count':__json['user']['statusesCount'],
                'profile_image':__json['user']['profileImageUrl'],
                'friends_count':__json['user']['friendsCount'],
                'viewsCount':__json['viewCount'],
                'followers_count':__json['user']['followersCount'],
                'country':__json['user']['location'],
                'likes':__json['likeCount'],
                'retweets':__json['retweetCount'],
                'comments':__json['replyCount'],
                'totalengagements':total_engagement

            }
            tweets_list1.append(dat)
            tweets_list_analysis.append(tweet.rawContent)

        word_freq = (adv.word_frequency(tweets_list_analysis))
```

FETCH THE TWITTER DATA

ASSIGNMENT OF THE DATA RECEIVED TO RELEVANT VARIABLES

ASSIGNMENT OF THE VARIABLES TO BE RETURNED TO THE API

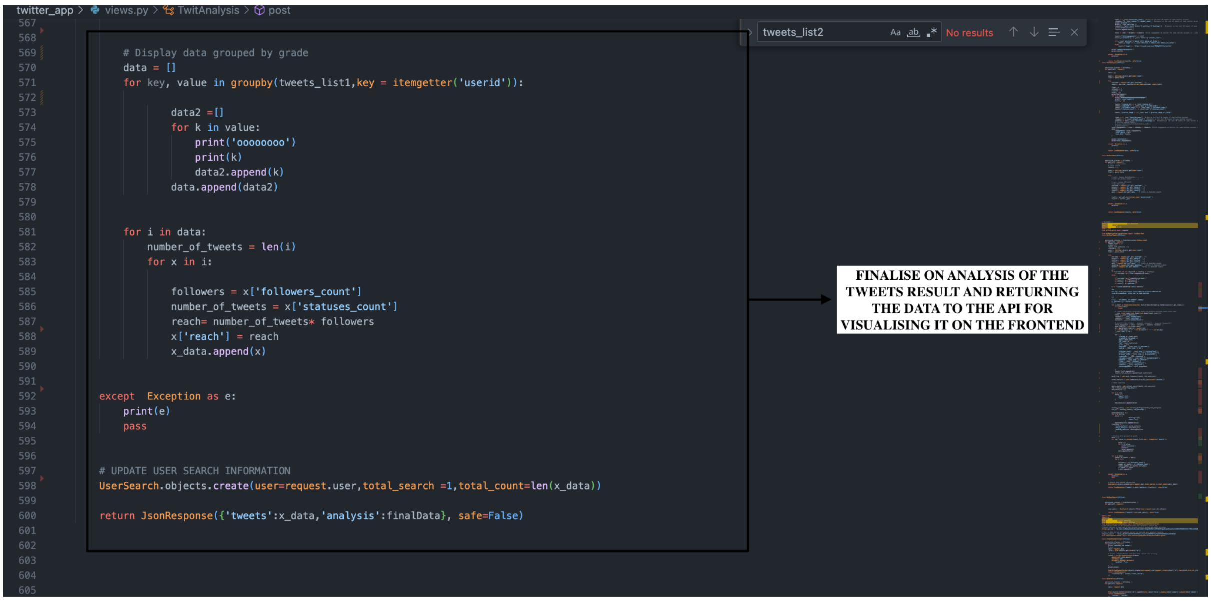
3)



```python
        word_freq = (adv.word_frequency(tweets_list_analysis))

        words_analysis = json.loads(word_freq.to_json(orient='records'))

        # EMOJI ANALYSIS

        emoji_analy = adv.extract_emoji(tweets_list_analysis)
        ttp = emoji_analy['top_emoji']
        emojiAnalysis =[]

        for i in ttp:
            data2 = {
                "emoji":i[0] ,
                "count":i[1]
            }

            emojiAnalysis.append(data2)


        hashtag_summary = adv.extract_hashtags(tweets_list_analysis)
        hst_sm =  hashtag_summary['top_hashtags']

        hashtagAnalysis =[]
        for i in hst_sm:
            data2 = {
                        "hashtag":i[0] ,
                        "count":i[1]
                    }
            hashtagAnalysis.append(data2)
        finalData = {
            'words_analysis':words_analysis,
            'emoji_analysis':emojiAnalysis,
            'hashtag_analysis':hashtagAnalysis
            }


        # Display data grouped by grade
        data = []
        for key, value in groupby(tweets_list1,key = itemgetter('userid')):

                data2 =[]
                for k in value:
                    print('oooooooo')
                    print(k)
                    data2.append(k)
                data.append(data2)


        for i in data:
            number_of_tweets = len(i)
            for x in i:
```

EMOJI,HASHTAGAND WORDS ANALYSIS FOR THE RESULT GIVEN

4)



```python
    # Display data grouped by grade
    data = []
    for key, value in groupby(tweets_list1,key = itemgetter('userid')):

            data2 =[]
            for k in value:
                print('oooooooo')
                print(k)
                data2.append(k)
            data.append(data2)


    for i in data:
        number_of_tweets = len(i)
        for x in i:

            followers = x['followers_count']
            number_of_tweets = x['statuses_count']
            reach= number_of_tweets* followers
            x['reach'] = reach
            x_data.append(x)


except  Exception as e:
    print(e)
    pass


# UPDATE USER SEARCH INFORMATION
UserSearch.objects.create(user=request.user,total_search =1,total_count=len(x_data))

return JsonResponse({'tweets':x_data,'analysis':finalData}, safe=False)
```

FINALISE ON ANALYSIS OF THE TWEETS RESULT AND RETURNING THE DATA TO THE API FOR VISUALISING IT ON THE FRONTEND