In [1]:

```
import pandas as pd
import csv
```

In [2]:

```
df=pd.read_csv("../data/CometLanding.csv")
```

In [3]:

```
df
```

Out[3]:

	id_str	from_user	text	created_at	time	geo_coordina
0	5.409304e+17	amika0078788556	RT @VersaTechnology: Congratulations @Philae20...	Fri Dec 05 18:07:14 +0000 2014	05/12/2014 18:07	N
1	5.409300e+17	ChrisDMarshall	CometWatch 2 December Â» Rosetta - ESA's comet...	Fri Dec 05 18:05:46 +0000 2014	05/12/2014 18:05	N
2	5.409300e+17	MHuuskoL	RT @EUCouncil: After the #CometLanding - Astro...	Fri Dec 05 18:05:36 +0000 2014	05/12/2014 18:05	N
3	5.409293e+17	SaraGomezAranci	RT @EUCouncil: After the #CometLanding - Astro...	Fri Dec 05 18:03:00 +0000 2014	05/12/2014 18:03	N
4	5.409292e+17	CBCDay6	RT @shaunmajumder: Feels good to be the @CBCDa...	Fri Dec 05 18:02:32 +0000 2014	05/12/2014 18:02	N
...	...	...	...	...	...	
77314	5.324601e+17	ABForScience	This means that the actual landing will be ar...	Wed Nov 12 09:09:26 +0000 2014	12/11/2014 09:09	N
77315	5.324601e+17	atieyK	RT @ObservingSpace: Weâ€™ve been waiting 10 ye...	Wed Nov 12 09:09:26 +0000 2014	12/11/2014 09:09	N
77316	5.324601e+17	j0nny5	RT @maxplanckpress: Accomazzo (flight director...	Wed Nov 12 09:09:26 +0000 2014	12/11/2014 09:09	N
77317	5.324601e+17	nsentse	7 hours of waiting #CometLanding	Wed Nov 12 09:09:26 +0000 2014	12/11/2014 09:09	N
77318	5.324601e+17	grery92	RT @dsdanyds: TopTrendIT: TT ITALIA 09:59\n1.#...	Wed Nov 12 09:09:26 +0000 2014	12/11/2014 09:09	N

77319 rows × 17 columns

In [4]:

```
len(df)
```

Out[4]:

```
77319
```

In [5]:

```
df.dtypes
```

Out[5]:

```
id_str                      float64
from_user                    object
text                         object
created_at                   object
time                         object
geo_coordinates              object
user_lang                    object
in_reply_to_user_id_str     float64
in_reply_to_screen_name      object
from_user_id_str            float64
in_reply_to_status_id_str   float64
source                       object
profile_image_url            object
user_followers_count        float64
user_friends_count          float64
status_url                   object
entities_str                 object
dtype: object
```

In [6]:

```
df.sort_values(by='time', ascending=True)
```

Out[6]:

	id_str	from_user	text	created_at	time	geo_coordinate
1685	5.392087e+17	jorgerojas00	RT @UniversoCnocido: En una sola imagen todos ...	Mon Dec 01 00:05:43 +0000 2014	01/12/2014 00:05	Na
1684	5.392089e+17	jorgerojas00	RT @UniversoCnocido: .@Philae2014 ya estÃ¡ seg...	Mon Dec 01 00:06:29 +0000 2014	01/12/2014 00:06	Na
1683	5.392089e+17	x5_dc	RT @ScienceNews: Good night, Philae. We may se...	Mon Dec 01 00:06:40 +0000 2014	01/12/2014 00:06	Na
1682	5.392089e+17	x5_dc	RT @SNStudents: This photo shows the primary l...	Mon Dec 01 00:06:44 +0000 2014	01/12/2014 00:06	Na
1681	5.392091e+17	x5_dc	RT @ScienceNews: Philae lander sent in a surpr...	Mon Dec 01 00:07:12 +0000 2014	01/12/2014 00:07	Na
...	...	...	...	...	...	
1689	5.391996e+17	LukeGolds	Â¦ 693 Â¦ Philae touches down on the surface ...	Sun Nov 30 23:29:35 +0000 2014	30/11/2014 23:29	Na
1688	5.391996e+17	LukeGolds	â™¦ 584 #cometlanding â™¦ comet landing â™...	Sun Nov 30 23:29:37 +0000 2014	30/11/2014 23:29	Na
1687	5.392013e+17	diuk37	RT @davidshukmanbbc: Interesting new take on #...	Sun Nov 30 23:36:25 +0000 2014	30/11/2014 23:36	Na
1686	5.392047e+17	ben_economics	RT @rjmlaird: My @ESA_Rosetta t-shirts have ar...	Sun Nov 30 23:49:50 +0000 2014	30/11/2014 23:49	Na
19364	NaN	NaN	NaN	NaN	NaN	Na

77319 rows × 17 columns

In [7]:

```
df['time']=pd.to_datetime(df['time'], dayfirst=True)
```

In [8]:

```
day=df.groupby(df['time'].dt.date)
```

In [9]:

```
day.size()
```

Out[9]:

```
time
2014-11-12    73253
2014-11-26      400
2014-11-27      497
2014-11-28      711
2014-11-29      428
2014-11-30      343
2014-12-01      603
2014-12-02      475
2014-12-03      312
2014-12-04      205
2014-12-05       91
dtype: int64
```

In [10]:

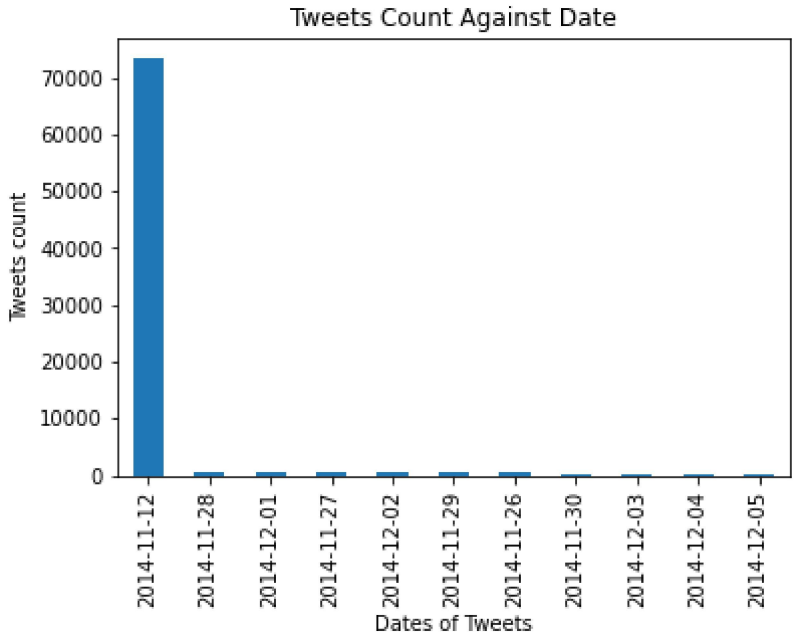
```
import matplotlib.pyplot as plt
%matplotlib inline
```

Graphs to show tweet activity over time.

First, a bar chart:

In [11]:

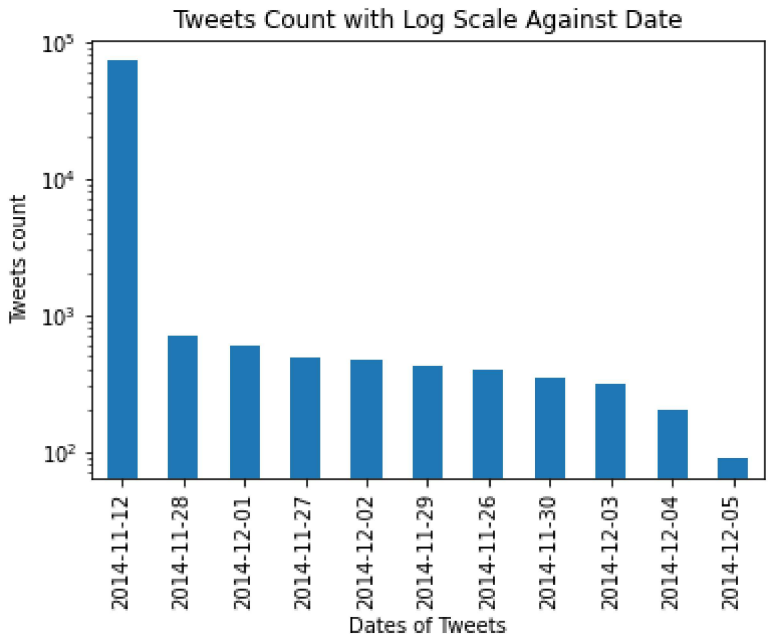
```
df['time'].dt.date.value_counts().plot(kind="bar").set(xlabel='Dates of Tweets', ylabel='Tw
plt.show()
```



Let's get a better look with a log scale:

In [12]:

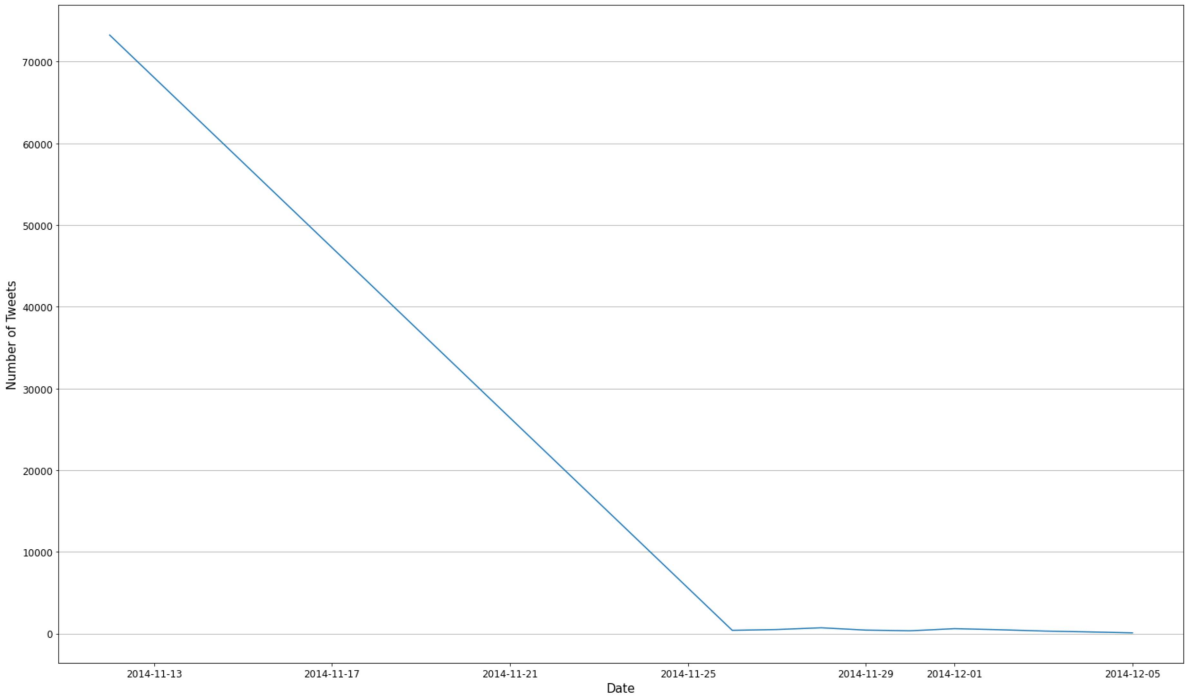
```
df['time'].dt.date.value_counts().plot(kind="bar").set(yscale='log', xlabel='Dates of Tweet
plt.show()
```



Now a line graph:

In [13]:

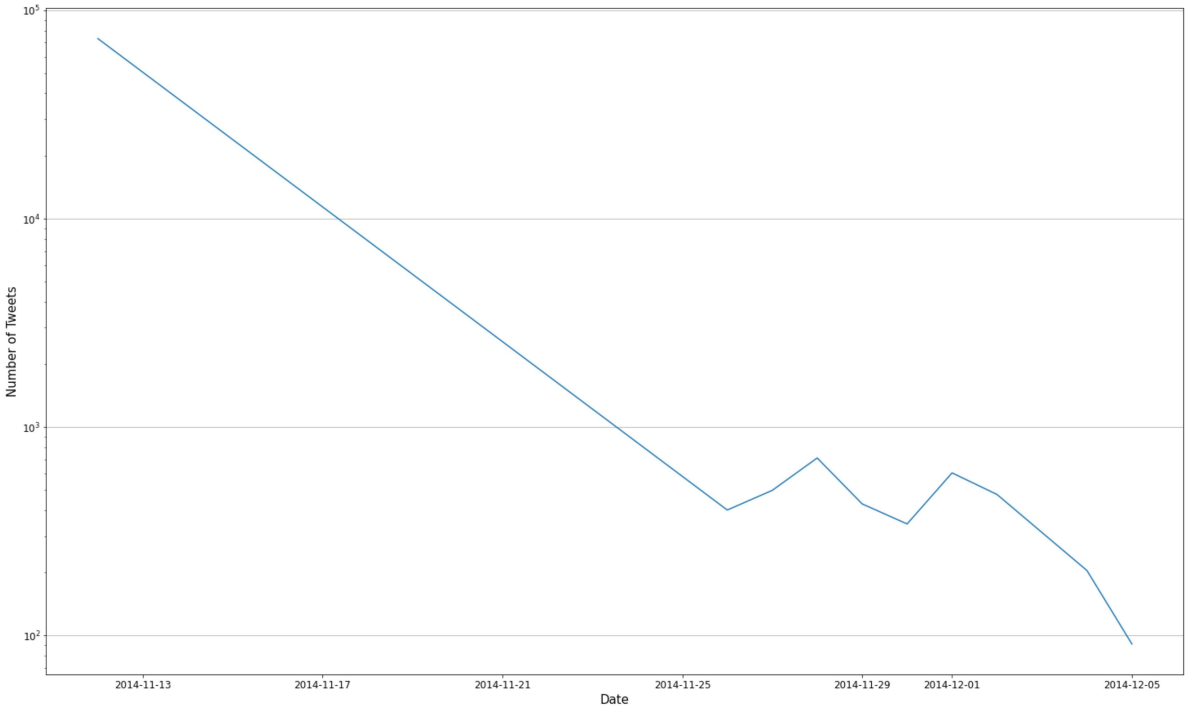
```
df.sort_values(by='time', ascending=True)
df['time']=pd.to_datetime(df['time'], dayfirst=True)
plt.xticks(fontsize=12)
plt.yticks(fontsize=12)
plt.xlabel("Date", fontsize=15)
plt.ylabel("Number of Tweets", fontsize=15)
df['time'].dt.date.value_counts().plot(figsize=(25,15))
plt.grid(axis = 'y')
plt.show()
```



Once again, let's get a better look by using a log scale on the y-axis

In [14]:

```
df['time'].dt.date.value_counts().plot(figsize=(25,15), logy=True)
plt.xticks(fontsize=12)
plt.yticks(fontsize=12)
plt.xlabel("Date", fontsize=15)
plt.ylabel("Number of Tweets", fontsize=15)
plt.grid(axis = 'y')
plt.show()
```



Check number of different (unique) users

In [15]:

```
# https://stackoverflow.com/questions/36106490/how-to-get-unique-values-from-multiple-colum
# By Yaakov Bressler (Last accessed date: 08-04-2022)
users=df.groupby(df['from_user_id_str']).agg(['unique'])
unique_users = len(users.index)
```

Number of replies

In [16]:

```
df_replies=df[['in_reply_to_user_id_str', 'in_reply_to_screen_name']]
df_replies=df_replies.dropna(subset=['in_reply_to_user_id_str', 'in_reply_to_screen_name'],
```

In [17]:

```
len(df_replies.index)
```

Out[17]:

```
1725
```

Lets have a look at what languages the tweets are in:

In [18]:

```
#Only display languages that appear frequently, lets say more than 400 times.
df2 = df.groupby('user_lang').filter(lambda x : len(x)<400)
lst = df2['user_lang'].tolist()
lst = list(dict.fromkeys(lst))
lst
for ln in lst:
    df['user_lang'].replace(ln, 'other', inplace=True)
```

In [19]:

```
lang=df.groupby(df['user_lang'])
```

In [20]:

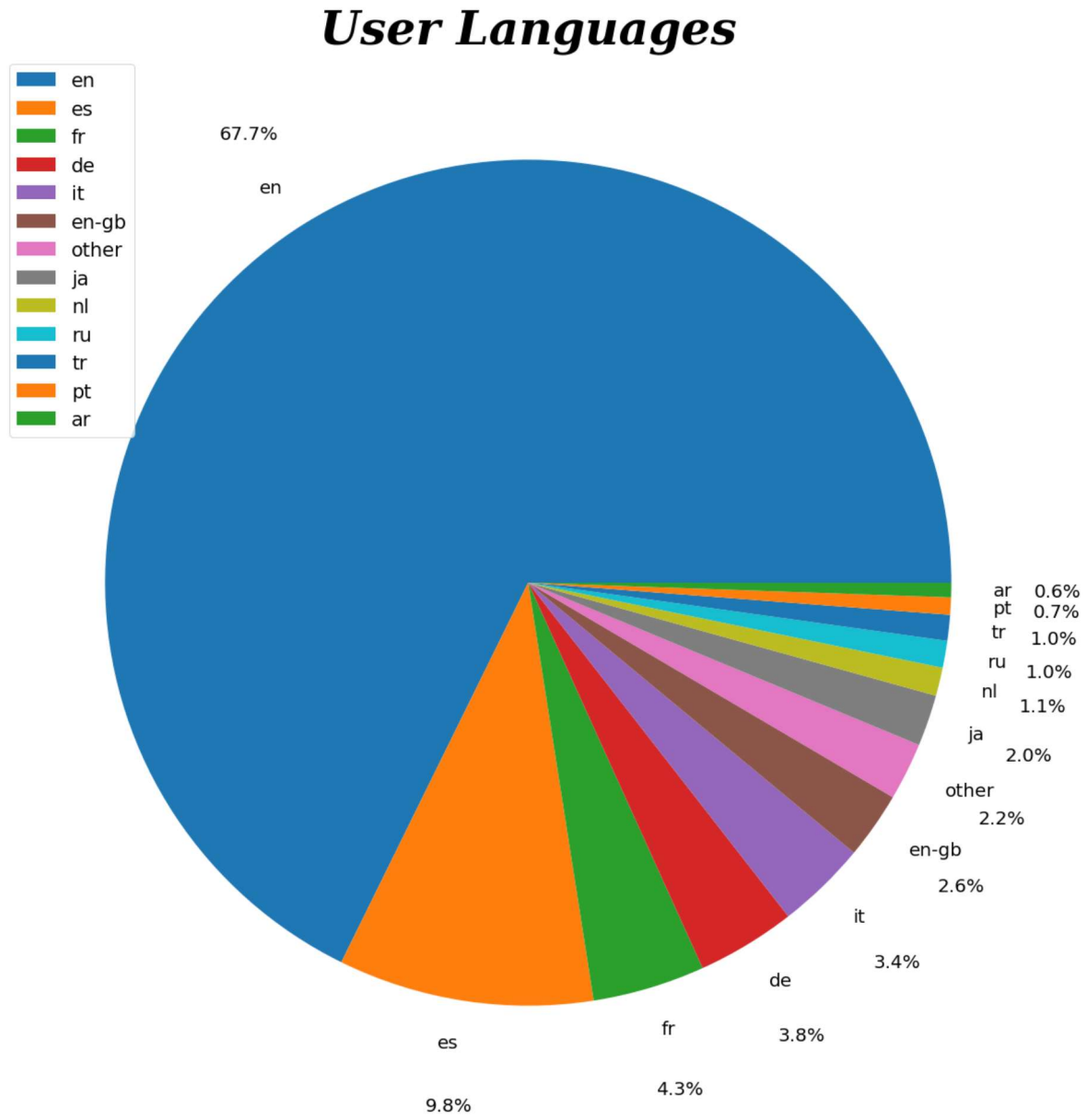
```
lang.size()
```

Out[20]:

```
user_lang
ar          428
de         2917
en        52359
en-gb      1972
es         7544
fr         3315
it         2665
ja         1514
nl          838
other      1703
pt          508
ru          794
tr          761
dtype: int64
```

In [21]:

```
csfont = {'fontname':'Serif'}
df['user_lang'].value_counts().plot(kind="pie", figsize=(21,21), fontsize=20, autopct='%1.1
plt.xlabel("")
plt.ylabel("")
plt.title(label="User Languages", fontsize=50, fontstyle="italic", **csfont, fontweight="bo
plt.legend(loc='upper left',fontsize=21)
plt.show()
```



We can also determine the frequency of hashtags

In [22]:

```
import json
import re
df = pd.read_csv("../data/CometLandingRefined.csv")['entities_str']
df.reset_index()
```

Out[22]:

	index	entities_str
0	0	{"hashtags":[{"text":"Philae","indices":[49,56...
1	1	{"hashtags":[{"text":"CometLanding","indices":...
2	2	{"hashtags":[{"text":"CometLanding","indices":...
3	3	{"hashtags":[{"text":"CometLanding","indices":...
4	4	{"hashtags":[{"text":"MiniMansbridge","indices...
...	...	...
77047	77047	{"hashtags":[{"text":"CometLanding","indices":...
77048	77048	{"hashtags":[{"text":"cometlanding","indices":...
77049	77049	{"hashtags":[{"text":"CometLanding","indices":...
77050	77050	{"hashtags":[{"text":"CometLanding","indices":...
77051	77051	{"hashtags":[{"text":"GUERRIERO","indices":[44...

77052 rows × 2 columns

In [23]:

```
lst = []
for index in range(df.shape[0]):
    results = re.findall(r"{\"text\":\".+\",", str(df.iloc[index]))
    for i in results:
        if(i.startswith('{\"text\":\"') and i.endswith("\",")):
            ents=i.split(",")
            ents=filter(lambda x: "text" in str(x), ents)
            for j in ents:
                lst.append(j)
```

In [24]:

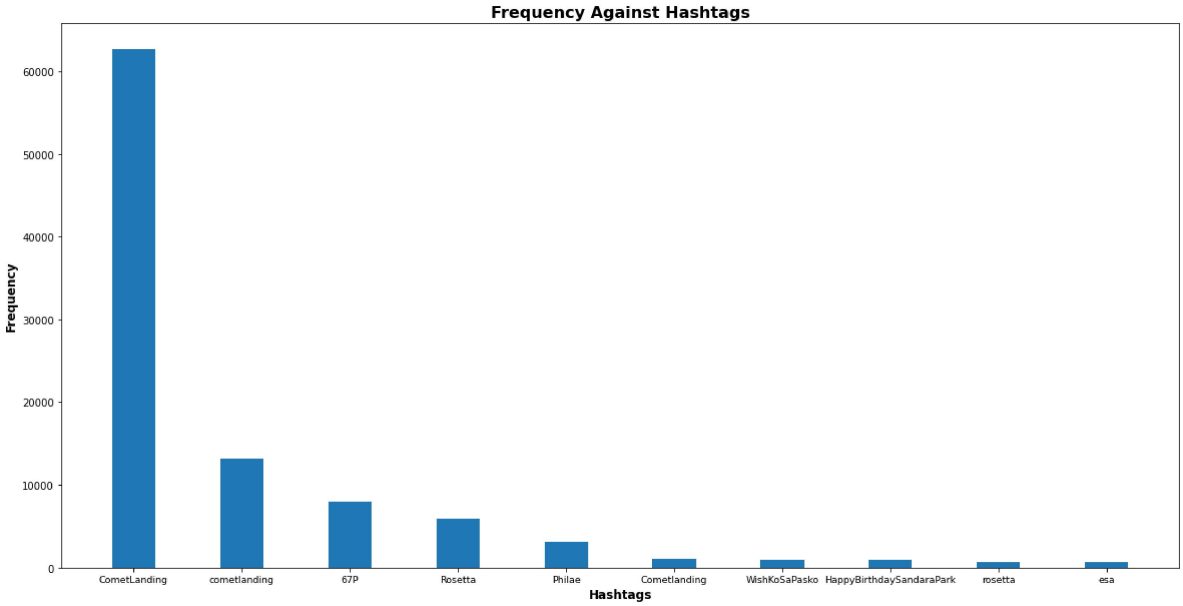
```
cleanlst = []
lst=filter(lambda x: "{\"text\":" in str(x), lst)
for i in lst:
    lst2 = i.split(":")
    cleanlst.append(lst2[1].replace("\"", ""))
print('Total number of hashtags: ' + str(len(cleanlst)))
```

```
Total number of hashtags: 115113
```

iterate down the rows take final column code that extract and take top 10

In [25]:

```
from collections import Counter
from itertools import chain
hashtag_counts=Counter(cleanlst)
plt.figure(figsize=(20,10))
hashtag_plot=plt.bar(*zip(*hashtag_counts.most_common()[:10]), width=0.4) # Getting the top
plt.xticks(fontsize=9.5)
plt.xlabel("Hashtags", fontsize=12, fontweight="bold")
plt.ylabel("Frequency", fontsize=12, fontweight="bold")
plt.title("Frequency Against Hashtags", fontsize=16, fontweight="bold")
plt.show()
```



Just for fun - lets create a wordcloud for hashtag content

In [26]:

```
from wordcloud import WordCloud
# removing duplicates from cleanlst
unique_cleanlst = list(dict.fromkeys(cleanlst))
text_for_word_cloud = ' '.join(unique_cleanlst)
word_cloud = WordCloud(collocations=False, background_color='white').generate(text_for_word
plt.imshow(word_cloud, interpolation='bilinear')
plt.axis("off")
plt.show()
```

In [27]:

```
df = pd.read_csv("../data/CometLanding.csv")
size_all = len(df)
#Remove all retweets
def retweet(inp):
    if(str(inp).startswith("RT @")):
        return True
    else:
        return False
df=df[df['text'].map(retweet) == False]
num_retweets = size_all - len(df)
```

In [28]:

```
num_retweets
```

Out[28]:

```
59966
```

In [29]:

```
df_replies=df[['in_reply_to_user_id_str', 'in_reply_to_screen_name']]
df_replies=df_replies.dropna(subset=['in_reply_to_user_id_str', 'in_reply_to_screen_name'],
replies = len(df_replies.index)
replies
```

Out[29]:

```
1695
```

In [30]:

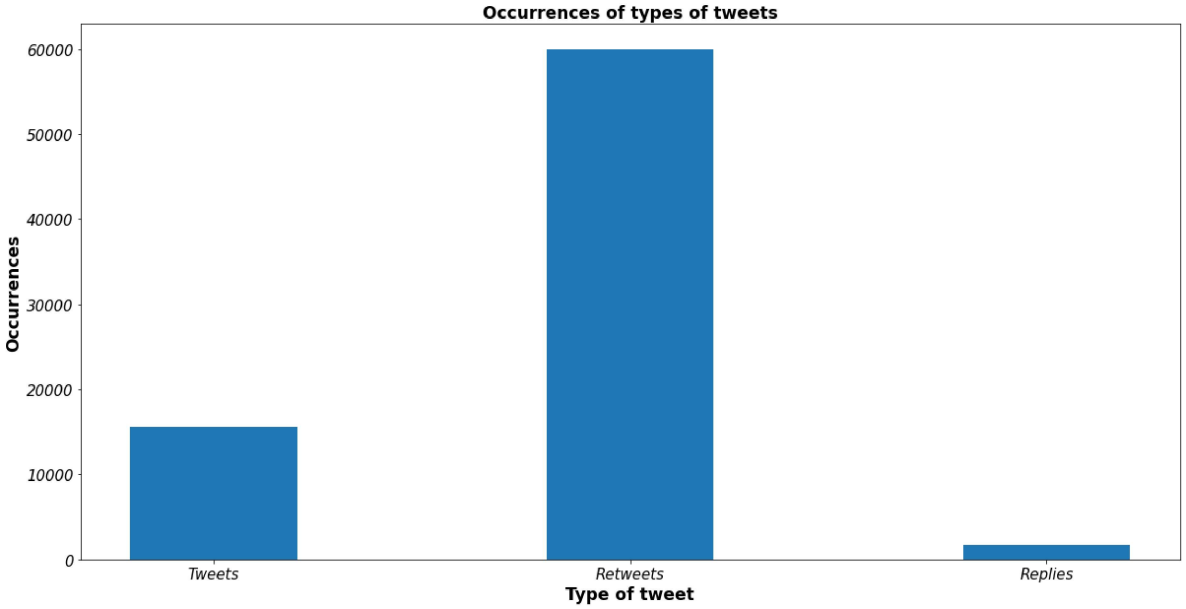
```
standard_tweets = len(df) - replies
standard_tweets
```

Out[30]:

```
15658
```

In [31]:

```
data = {'Tweets':standard_tweets, 'Retweets':num_retweets, 'Replies':replies}
types = list(data.keys())
values = list(data.values())
plt.figure(figsize=(20,10))
plt.bar(types, values, width=0.4)
plt.xticks(fontsize=15, fontstyle='italic')
plt.yticks(fontsize=15, fontstyle='italic')
plt.xlabel('Type of tweet', fontsize=17, fontweight='bold')
plt.ylabel('Occurrences', fontsize=17, fontweight='bold')
plt.title('Occurrences of types of tweets', fontsize=17, fontweight='bold')
plt.show()
```



In [32]:

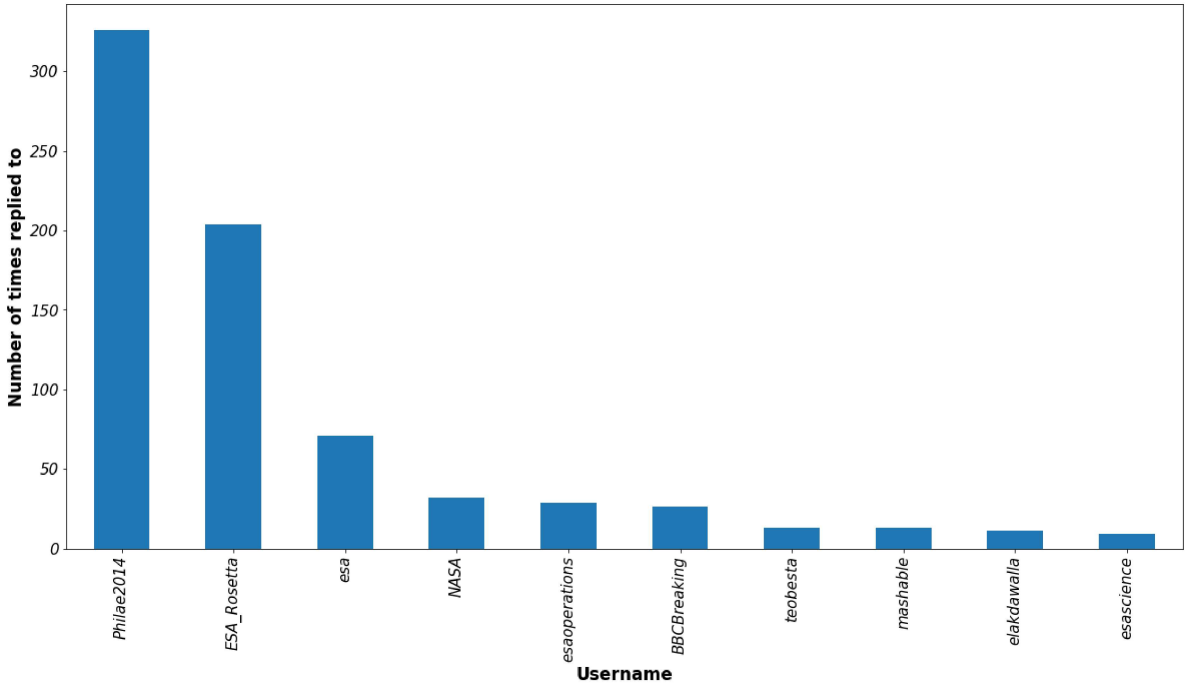
```
print("Average number of tweets per user: " + str(standard_tweets/unique_users))
print("Average number of retweets per user: " + str(num_retweets/unique_users))
print("Average number of replies per user: " + str(replies/unique_users))

print("\nAverage number of tweets of any kind per user: " + str(size_all/unique_users))
```

```
Average number of tweets per user: 0.3118129679783336
Average number of retweets per user: 1.1941612235144177
Average number of replies per user: 0.033754181934044926

Average number of tweets of any kind per user: 1.5397283734267961
```

Top 10 most replied to users:

In [33]:

```
repl = df['in_reply_to_screen_name'].value_counts()[:10]

plt.figure(figsize=(20,10))
plt.xticks(fontsize=15, fontstyle='italic')
plt.yticks(fontsize=15, fontstyle='italic')
plt.xlabel('Username', fontsize=17, fontweight='bold')
plt.ylabel('Number of times replied to', fontsize=17, fontweight='bold')
repl.plot(kind='bar')
plt.show()
```



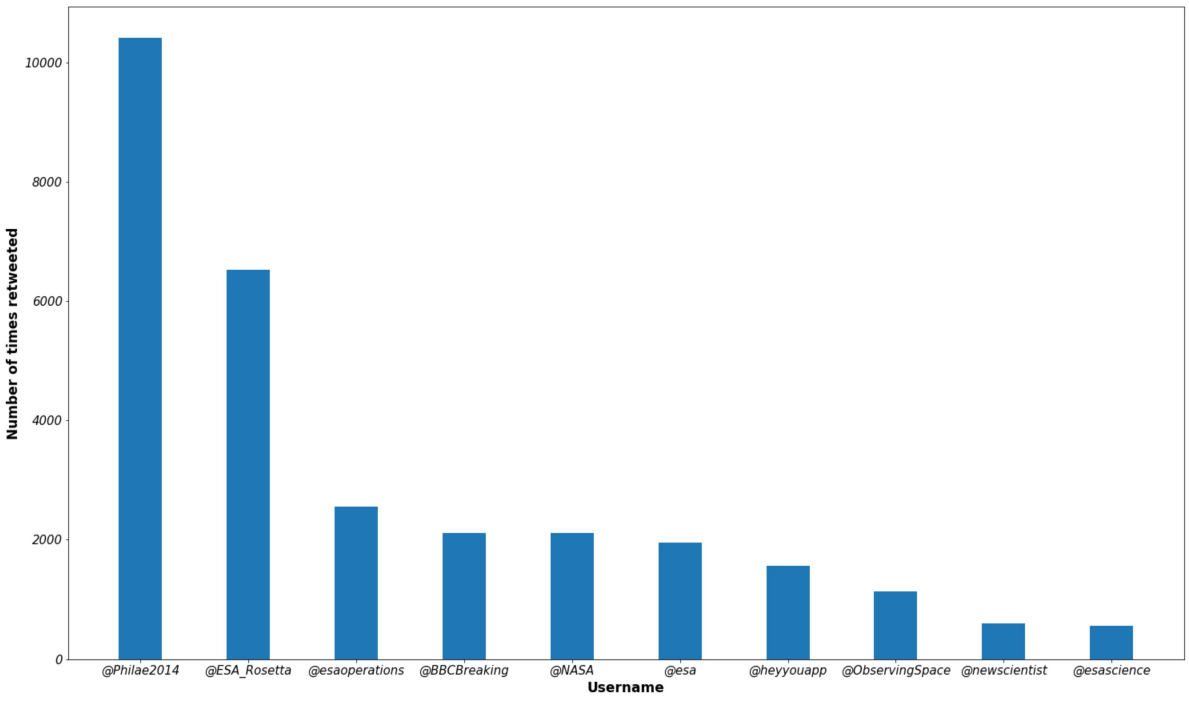
Now let's find out which users were retweeted the most (Top 10):

In [34]:

```
#Only keep retweets
df = pd.read_csv("../data/CometLanding.csv")
size_all = len(df)
#Remove all retweets
def retweet(inp):
    if(str(inp).startswith("RT @")):
        return True
    else:
        return False
df=df[df['text'].map(retweet) == True]
num_retweets = len(df)
```

In [35]:

```
usrs = []
usrsclean = []
for i in range(num_retweets):
    txt_vals = str(df.loc[:, 'text'].values[i]).split(" ")
    usrs.append(txt_vals[1])
for strng in usrs:
    usrsclean.append(strng[:-1])
usrsclean

rt_counts=Counter(usrsclean)
plt.figure(figsize=(25,15))
plt.xticks(fontsize=15, fontstyle='italic')
plt.yticks(fontsize=15, fontstyle='italic')
plt.xlabel('Username', fontsize=17, fontweight='bold')
plt.ylabel('Number of times retweeted', fontsize=17, fontweight='bold')
plt.bar(*zip(*rt_counts.most_common()[:10]), width=0.4)
plt.show()
```



Applications Used for Sending Tweets (Top 5):

In [36]:

```
df = pd.read_csv("../data/CometLanding.csv")
srces = []
for i in range(len(df)):
    html = str(df.loc[:, 'source'].values[i])
    relst = re.findall(">.+<", html)
    for i in relst:
        srces.append(i[1:-1])
```

In [37]:

```
src_counts=Counter(srces)
plt.figure(figsize=(25,15))
plt.xticks(fontsize=15, fontstyle='italic')
plt.yticks(fontsize=15, fontstyle='italic')
plt.xlabel('Application used', fontsize=17, fontweight='bold')
plt.ylabel('Number of tweets', fontsize=17, fontweight='bold')
plt.bar(*zip(*src_counts.most_common()[:5]), width=0.4)
plt.show()
```