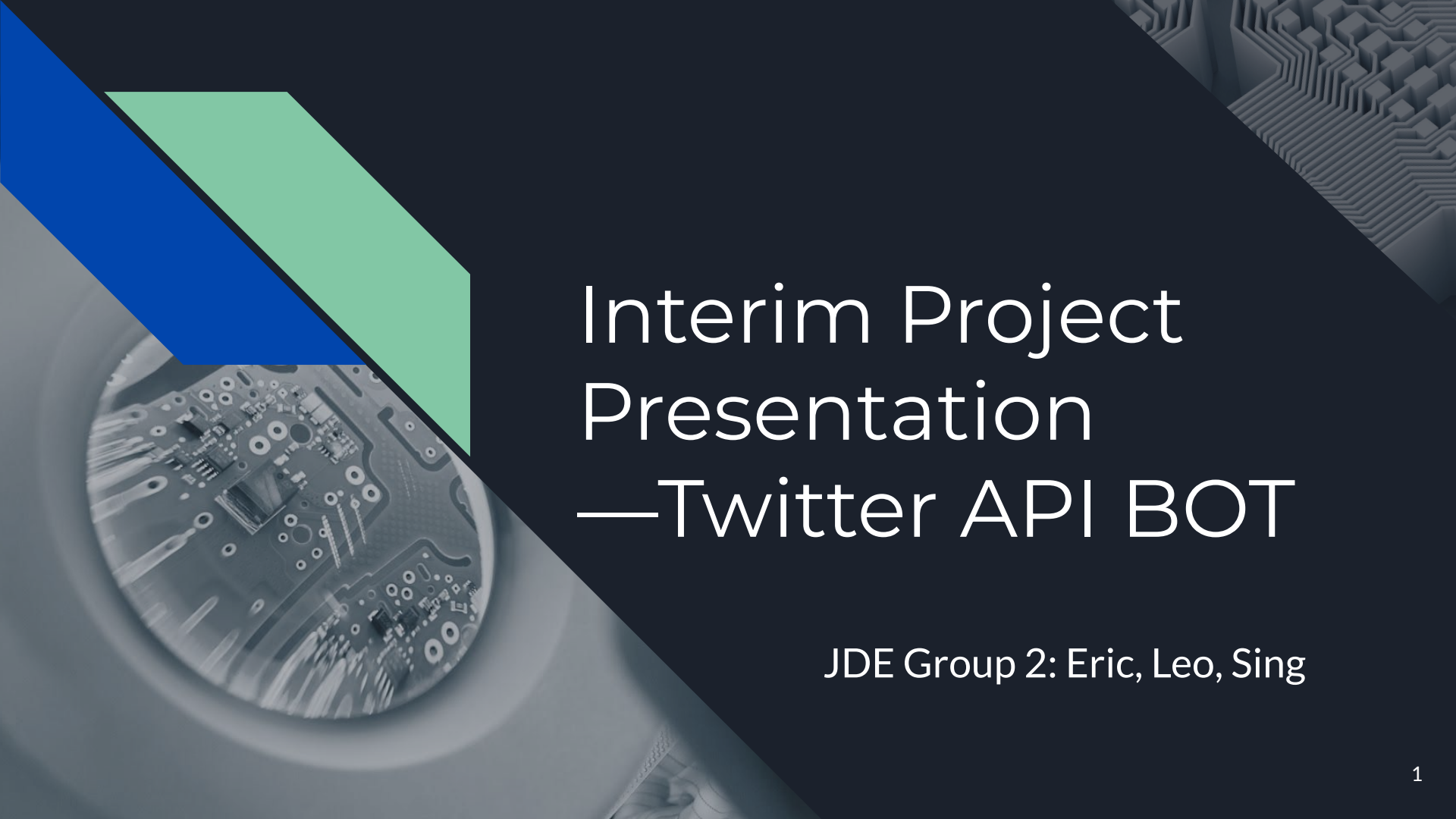


Interim Project Presentation —Twitter API BOT

JDE Group 2: Eric, Leo, Sing

Table of Contents

Introduction

- Why do we need to build the data crawler and database?
- How do we build the data crawler and database?
- What benefit does the data crawler and database bring?

Introduction

Twitter is an online social media site that allows users to send and read short (i.e. 280 characters) messages called "tweets" in real time. The Twitter database contains a wealth of information about the popularity ratings of politicians, celebrities, and other public figures, as well as their social media activity. This Twitter database can be queried for insight into public affairs, especially Coronavirus and Vaccination.

Problems (Requirements)

01 The crawler can collect the user's profile information from Twitter user- Joe Biden (@JoeBiden).

02 The crawler can collect the user's social network information from the Twitter user- Joe Biden (@JoeBiden).

03 The crawler can collect the tweets using the following two keywords: [Coronavirus, Vaccination].

Solution

We would use Python to extract and transform the data. Then, the data will be loaded in Postgre database and the searcher can get the data by utilizing PgAdmin interface so they can gain basic insights from it without technical knowledge.

Product

Advantages

- *Our BOT can not only find the twitter information of Joe Biden, but also other users. (Required basic Python knowledge)*
- *The retrieval date and time is also recorded automatically into the database. This can allow the searcher to know whether the data retrieved is updated or not.*
- *Still updating*

Assumptions

- Users would have a working knowledge of Python and PostgreSQL.

- Users install all necessary libraries.

Library

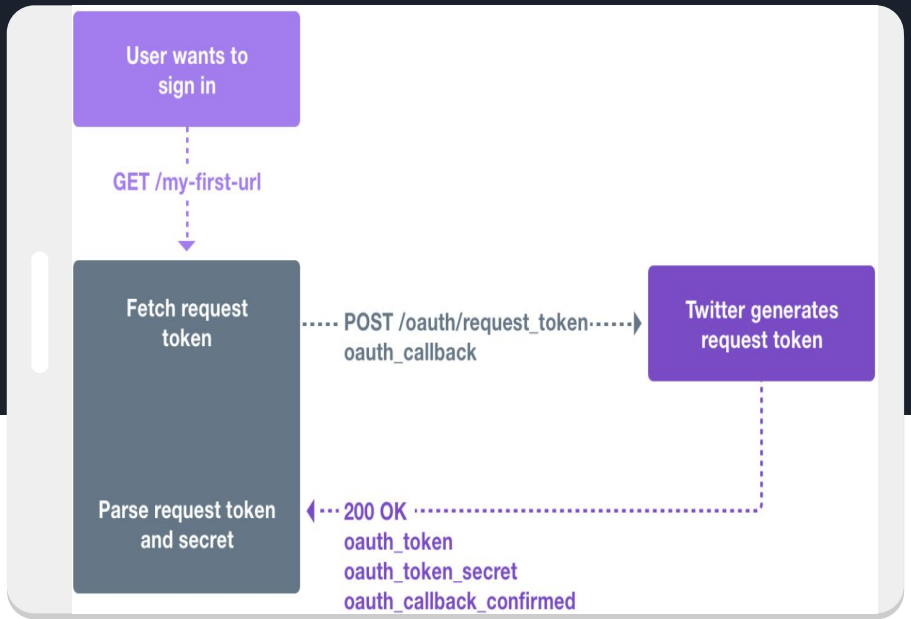
```
import psycopg2
import tweepy
import csv
import pandas as pd
import datetime
```

Consideration in Library

	Tweepy	Twint
Preparation	Make an elevated account on Twitter API, access authentication coding	Pip install
Functionality	Users info, tweets, interact with Twitter	tweets
Legality	Formal	Not really
Limit	Depends on access level. For elevated users. 2,000,000 tweets per month	Unlimited
Performance	Faster	Slower

Open Authentication (OAuth)

An open standard for authentication that is adopted by Twitter to provide access to the protected information.

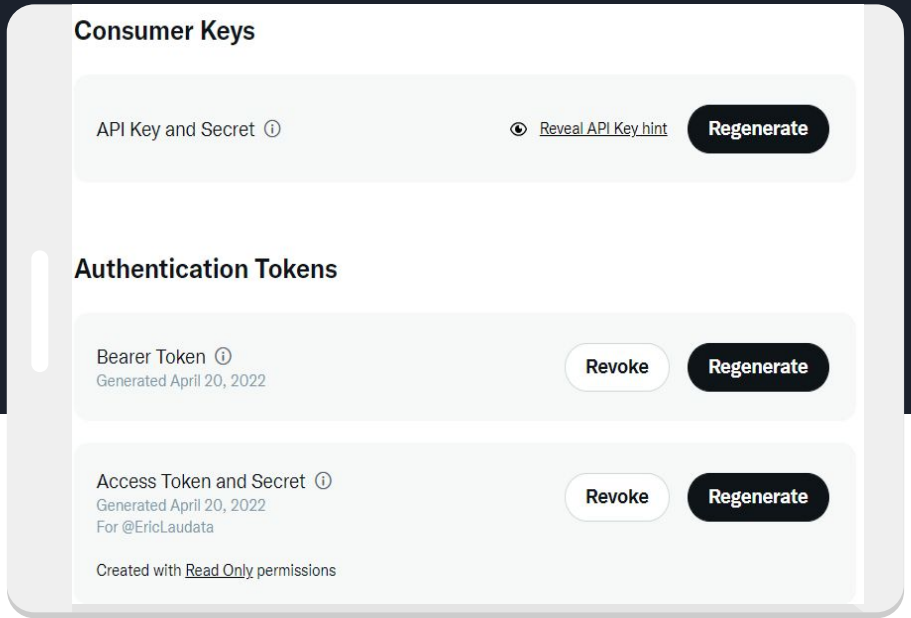


OAuth provides a safer alternative to traditional authentication approaches using a three-way handshake.

Here is the reference for more details about OAuth:https://developer.twitter.com/en/docs/authentication/oauth-2-0

Extraction of API authentication and login

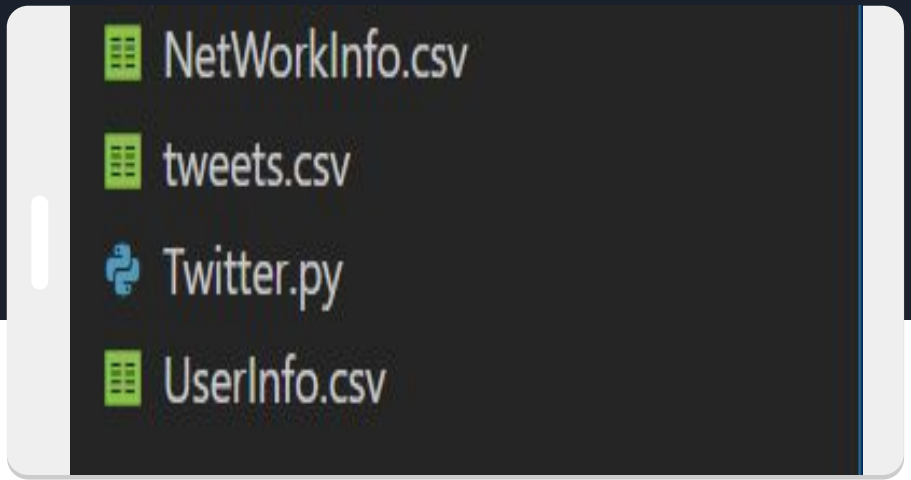
```
API_KEY = '' # User own key
API_SECRET = '' # User secret key
BEARER_TOKEN = '' # User own bearer token
ACCESS_TOKEN = '' # User own token
ACCESS_TOKEN_SECRET = '' # User own secret token
auth = tweepy.OAuthHandler(API_KEY,API_SECRET)
auth.set_access_token(ACCESS_TOKEN,ACCESS_TOKEN_S
ECRET)
api = tweepy.API(auth)
search_key ="UserInfo"
```



User may find their keys and token at https://developer.twitter.com/en/portal/projects/

Comma-separated values (CSV)

```
#Generate and open a new CSV file
csvFile =
open(search_key+".csv","a+",newline="",
encoding="utf-8")
csvWriter = csv.writer(csvFile)
#[...]
#Write data to CSV file
    csvWriter.writerow(tweets)
```



- Flat file
- Simple
- Delimiter is a comma (others is acceptable)

User Information API

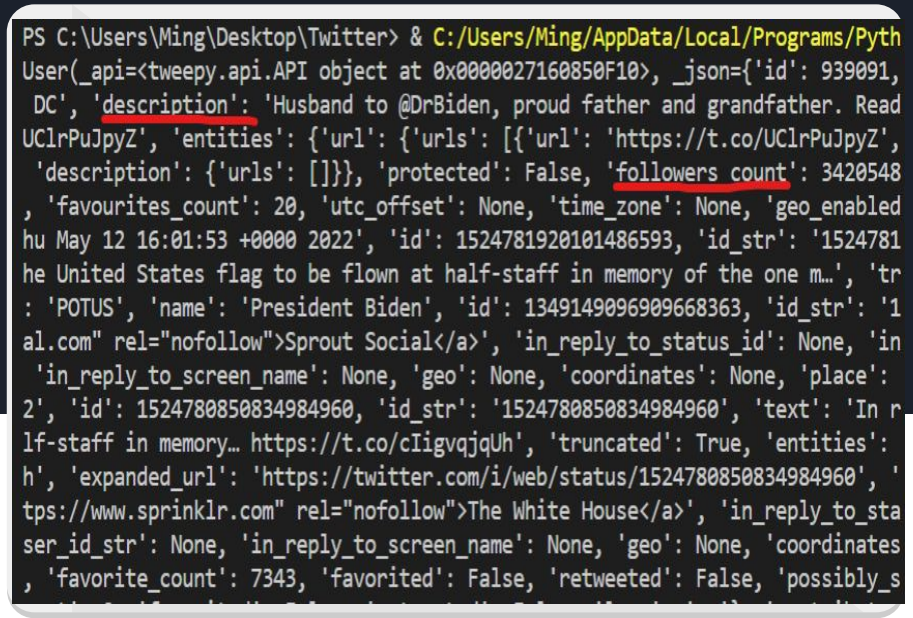
```
statuses =
api.lookup_users(user_id=[User_ID])
for tweet in statuses:

 tweets = [
      tweet.id,
      Tweet.name,
      Tweet.created_at,
      Tweet.location,
      Tweet.description]
```

```
Status(_api=<tweepy.api.API object at 0x0000015B00090F10>, _json={'created_at
88', 'text': 'If the Court overturns Roe, it will fall on our nation's electe
 True, 'entities': {'hashtags': [], 'symbols': [], 'user_mentions': [], 'urls
1603759901708288', 'display_url': 'twitter.com/i/web/status/1…', 'indices': [
', 'in_reply_to_status_id': None, 'in_reply_to_status_id_str': None, 'in_repl
': {'id': 939091, 'id_str': '939091', 'name': 'Joe Biden', 'screen_name': 'Jo
 grandfather. Ready to build back better for all Americans. Official account
//t.co/UClrPuJpyZ', 'expanded_url': 'http://joebiden.com', 'display_url': 'jo
wers_count': 34205604, 'friends_count': 48, 'listed_count': 39068, 'created_a
': None, 'geo_enabled': False, 'verified': True, 'statuses_count': 8202, 'lan
alse, 'profile_background_color': '565959', 'profile_background_image_url': '
tps://abs.twimg.com/images/themes/theme1/bg.png', 'profile_background_tile':
_normal.jpg', 'profile_image_url_https': 'https://pbs.twimg.com/profile_image
file_banners/939091/1626295479', 'profile_link_color': '233F94', 'profile_sid
'323232', 'profile_use_background_image': True, 'has_extended_profile': False
est_sent': False, 'notifications': False, 'translator_type': 'none', 'withhel
 'is_quote_status': False, 'retweet_count': 7430, 'favorite_count': 51386, 'f
```

Collecting the selected user's profile information from Twitter user(e.g. Joe Biden).

Network Information API

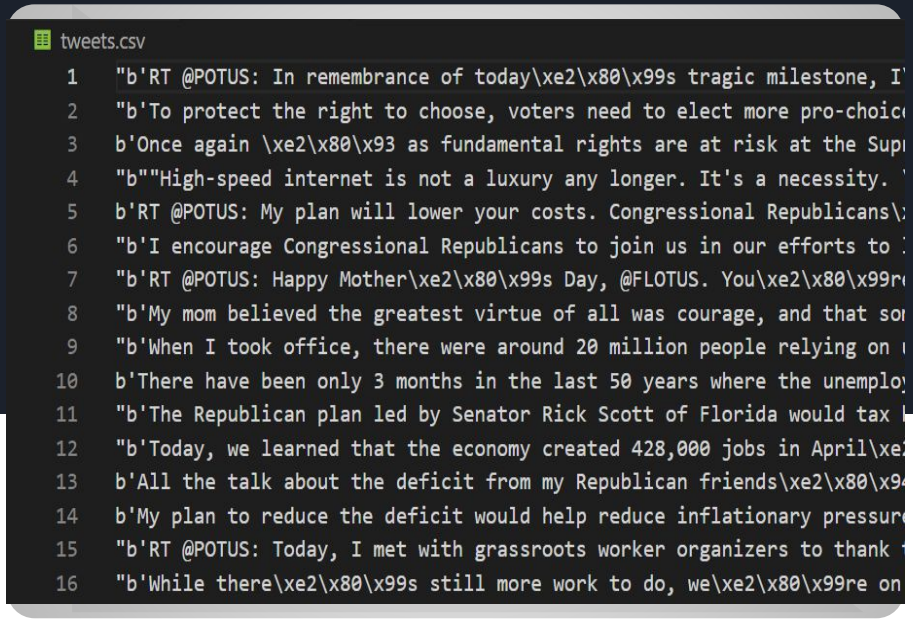
```
statuses =
api.lookup_users(user_id=[User_ID])
for tweet in statuses:

tweets = [
tweet.id,
tweet.followers_count,
tweet.friends_count]
    tweets.append(str
(datetime.datetime.today()))
```



Collecting the selected user's social network information from the Twitter(e.g. Joe Biden).

Tweets API

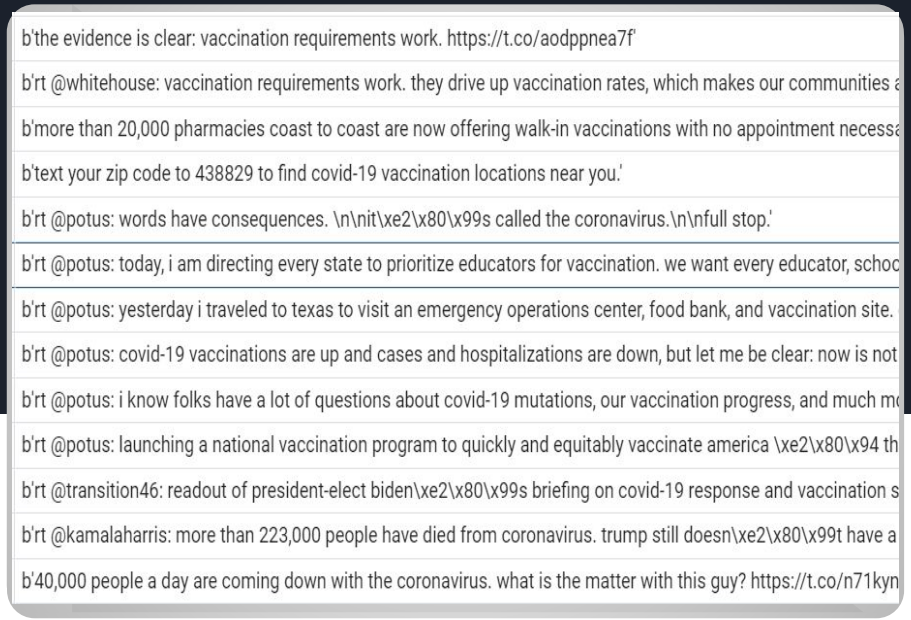
```
for tweet in

tweepy.Cursor(api.user_timeline,
screen_name=UserName,
count=200).items():

tweets = [
tweet.text.encode("utf-8"),
tweet.created_at,
tweet.user.id]
```



Collecting the selected user's tweets using keyword(e.g. Joe Biden).

Convert to lower letters

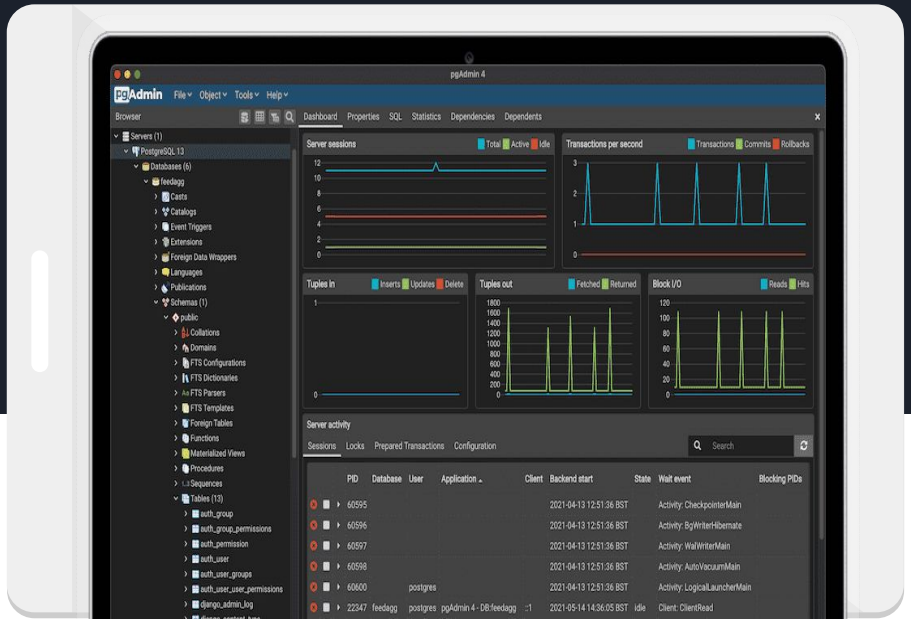
```
#Read Covid-19.CSV as PD
df = pd.read_csv('tweets.csv')
#Add title
df.columns =['text','time','tweetid']
#Convert all words in TEXT into lowercase for
subsequent SQL search
df['text'] = df['text'].str.lower()
```



lowercase all texts.

Connect to PostgreSQL

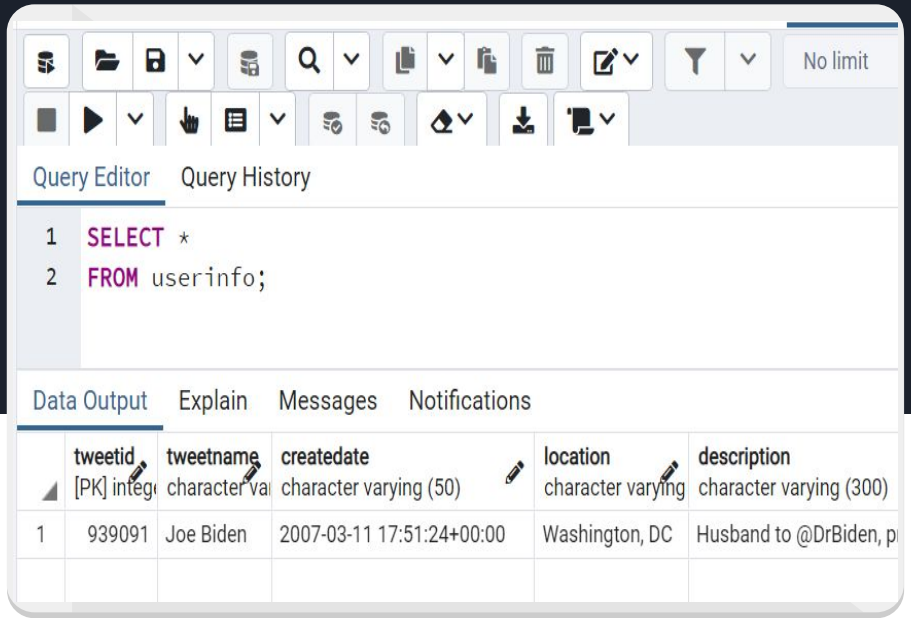
```
try:
    conn = psycopg2.connect(
        host = hostname,
        dbname = database,
        user = username,
        password = pwd,
        port = port_id
    )
    cur = conn.cursor()
```



PostgreSQL is a powerful, open source object-relational database system with over 30 years of active development that has earned it a strong reputation for reliability, feature robustness, and performance.

User Information Table

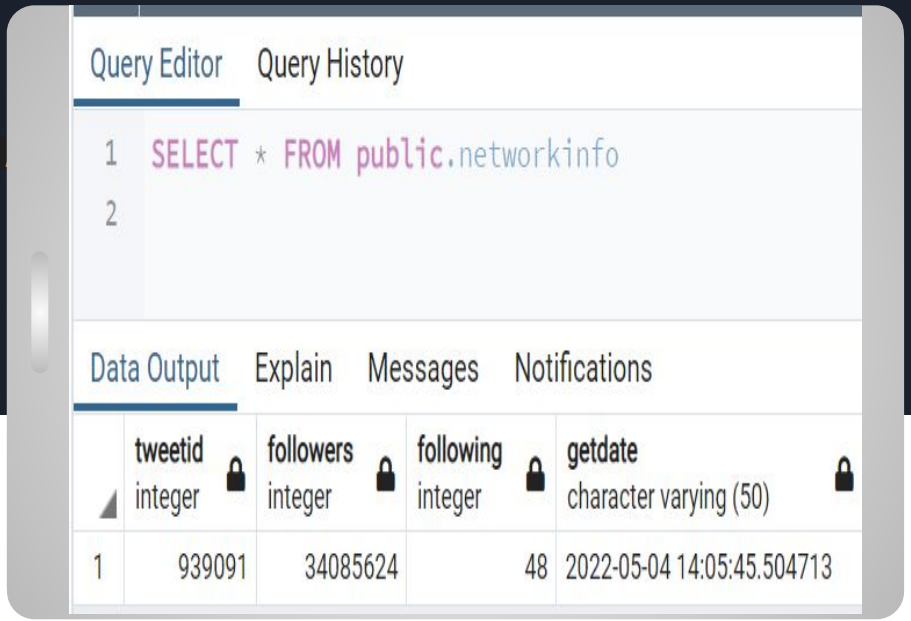
```
#Create Table(userinfo)
    create_script = '''CREATE TABLE IF NOT
EXISTS userinfo(
        tweetid int NOT NULL PRIMARY KEY,
        tweetname varchar(50),
        createdate varchar(50),
        location varchar(100),
        description varchar(300)
        )'''
    cur.execute(create_script)
```



Storing the selected user's information from the Twitter(e.g. Joe Biden).

Network Information Table

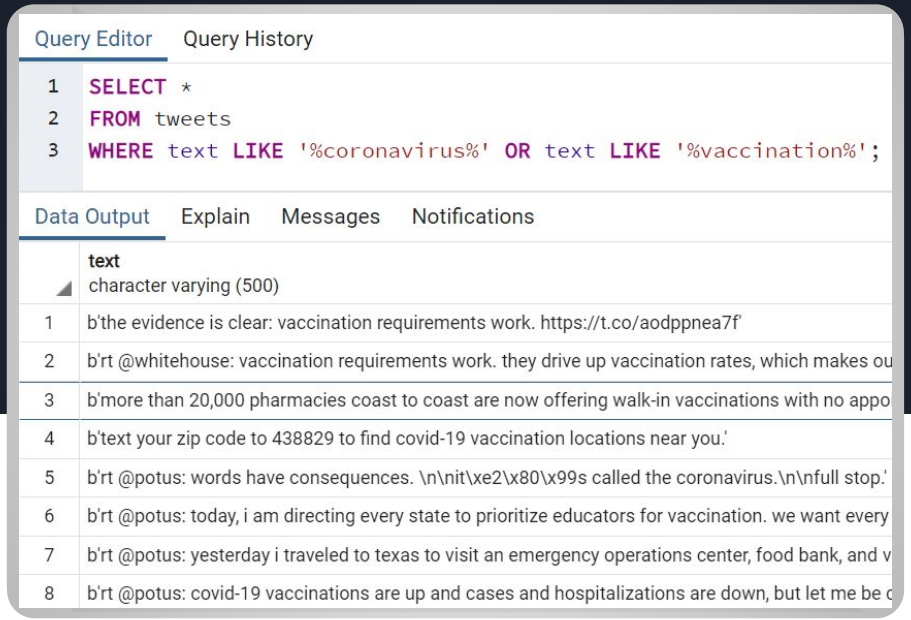
```
cur.execute('DROP TABLE IF EXISTS networkinfo)
create_script = '''CREATE TABLE IF NOT EXISTS
networkinfo
(
        tweetid int REFERENCES userinfo(tweetid),
        followers int,
        following int,
        getdate varchar(50)
        )'''
    cur.execute(create_script)
```



Storing the selected user's social network information from the Twitter(e.g. Joe Biden).

Tweets Table

```
# Create Table --- (tweets)
    cur.execute('''
        CREATE TABLE IF NOT EXISTS tweets (
            text varchar(500),
            time varchar,
            tweetid int REFERENCES
userinfo(tweetid))
                ''')
```



Query Editor | Query History

```
1 SELECT *
2 FROM tweets
3 WHERE text LIKE '%coronavirus%' OR text LIKE '%vaccination%';
```

Data Output | Explain | Messages | Notifications

	text character varying (500)
1	b'the evidence is clear: vaccination requirements work. https://t.co/aodppnea7f'
2	b'rt @whitehouse: vaccination requirements work. they drive up vaccination rates, which makes ou
3	b'more than 20,000 pharmacies coast to coast are now offering walk-in vaccinations with no appo
4	b'text your zip code to 438829 to find covid-19 vaccination locations near you.'
5	b'rt @potus: words have consequences. \n\nit\xe2\x80\x99s called the coronavirus.\n\nfull stop.'
6	b'rt @potus: today, i am directing every state to prioritize educators for vaccination. we want every
7	b'rt @potus: yesterday i traveled to texas to visit an emergency operations center, food bank, and v
8	b'rt @potus: covid-19 vaccinations are up and cases and hospitalizations are down, but let me be c

Storing the selected user's tweets using keyword(e.g. Joe Biden).

Input Data To PostgreSQL

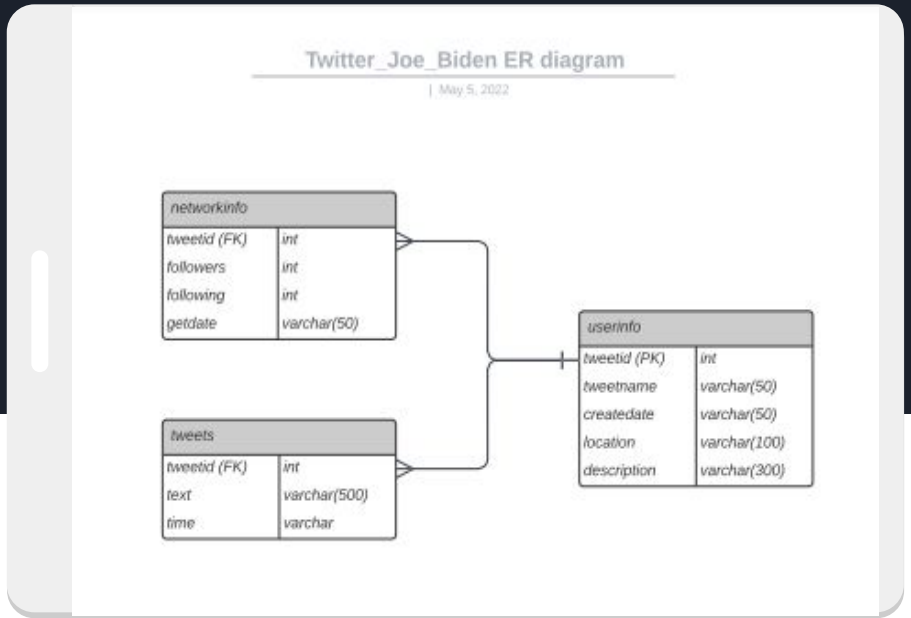
```
#Login PostgreSQL info
hostname = 'localhost'
database = 'demo'
username = 'postgres'
pwd = '123456'
port_id = 5432
conn = None
cur = None
#Conn to SQL
try:
    conn = psycopg2.connect(
        host = hostname,
        dbname = database,
        user = username,
        password = pwd,
        port = port_id)

    cur = conn.cursor()
```

```
#Del Table ----(userinfo)
    cur.execute('DROP TABLE IF
EXISTS userinfo')

#Create Table (userinfo)
    create_script = '''CREATE
TABLE IF NOT EXISTS userinfo(
        tweetid int NOT NULL
PRIMARY KEY,
        tweetname varchar(50),
        createdate varchar(50),
        location varchar(100),
        description
varchar(300)
        )'''
    cur.execute(create_script)
```

```
#Open CSV(UserInfo)
    my_file = open('UserInfo.csv')
    print('file opened in memory')

#Open CSV Upload to DB
    SQL_STATEMENT = """
    COPY userinfo FROM STDIN WITH
    CSV
    DELIMITER AS ','
    """

cur.copy_expert(sql=SQL_STATEMENT,file=my_file)
    print('file copied to db')

    conn.commit()
    print('import UserInfo to db completed')
    cur.close()
    conn.close()
except Exception as error:
    print(error)
```

Entity–relationship model

Postgre database included Three tables, connected with the tweetid.



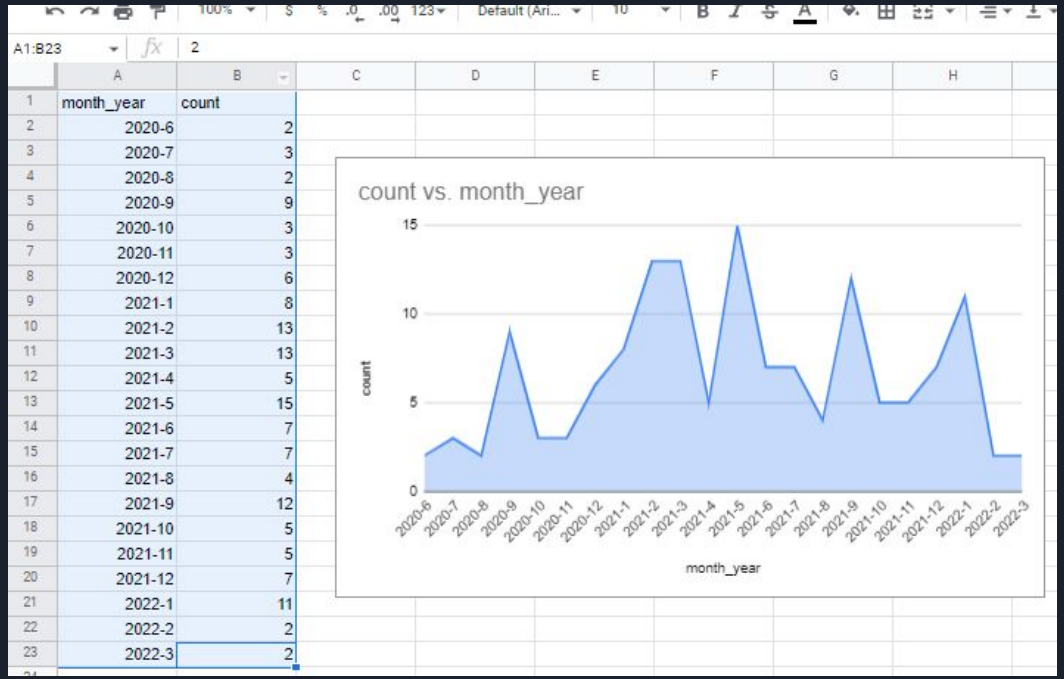
A Strong (Identifying) Relationship between userinfo, networkinfo and tweets.

Business rules

- Each user may have more than one network information. Each networkinfo only belongs to one user. Therefore, the relationship between userinfo and networkinfo is 1:M.
- Each user may have more than one tweets. Each tweets only belongs to one user. Therefore, the relationship between tweets and order is 1:M.

Application

Search keywords ‘vaccin’ or ‘corona’



	A	B
1	month_year	count
2	2020-6	2
3	2020-7	3
4	2020-8	2
5	2020-9	9
6	2020-10	3
7	2020-11	3
8	2020-12	6
9	2021-1	8
10	2021-2	13
11	2021-3	13
12	2021-4	5
13	2021-5	15
14	2021-6	7
15	2021-7	7
16	2021-8	4
17	2021-9	12
18	2021-10	5
19	2021-11	5
20	2021-12	7
21	2022-1	11
22	2022-2	2
23	2022-3	2

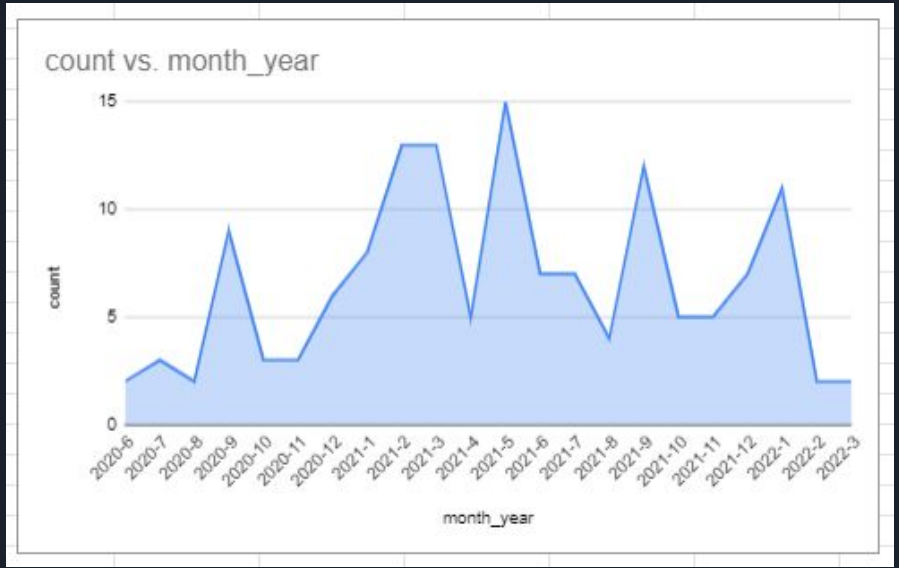

```
SELECT concat(EXTRACT(year FROM cast(time as timestamp)),'-',
EXTRACT(month FROM cast(time as timestamp))) as month_year,
count(*)
FROM covid_19
WHERE text LIKE '%corona%'
OR text LIKE '%vaccin%'
GROUP BY month_year
ORDER BY month_year;
```

	month_year text	count bigint
1	2020-10	3
2	2020-11	3
3	2020-12	6
4	2020-6	2
5	2020-7	3
6	2020-8	2
7	2020-9	9
8	2021-1	8
9	2021-10	5
10	2021-11	5
11	2021-12	7
12	2021-2	13
13	2021-3	13
14	2021-4	5

```
SELECT concat(EXTRACT(year FROM cast(time as timestamp)),'
EXTRACT(month FROM cast(time as timestamp))) as month_year
count(*)
FROM covid_19
WHERE text LIKE '%coronavirus%'
OR text LIKE '%vaccination%'
GROUP BY month_year
ORDER BY month_year;
```

	month_year text	count bigint
1	2020-10	3
2	2020-6	1
3	2020-7	2
4	2020-8	2
5	2020-9	2
6	2021-1	1
7	2021-10	2
8	2021-2	4
9	2021-3	2
10	2021-5	2

Analysis



- 1st peak = 2020/11 United States president election day
- Rising = 2020/12/11 First Covid-19 Vaccines approved
- Trough = 2021/7-8 Tokyo Olympics 2021
- last peak = 2021/11/3 WHO announced as Global Pandemic
- After 2022/2, fewer, as the Ukraine Russia Crisis

Trend:

Around 5 months a peak will occur

First 0 tweet in 2022/4, continue in 2022/5

Difficulties

- Design
- Implementation
- Time

Project timeline

Brainstorming/ Researching

A list of ideas generated, consisting of conceptual sketches, function and key components

Phase 1

A summary of open-source libraries comparison linked to project

Phase 2

The database schema, table structure, and entity-relationship model

The source code of crawler

Phase 3

A user guide about how to use crawler

Twitter data crawler prototype

Presentation deck

Final checking/ Testing

Testing the crawler prototype

Presentation

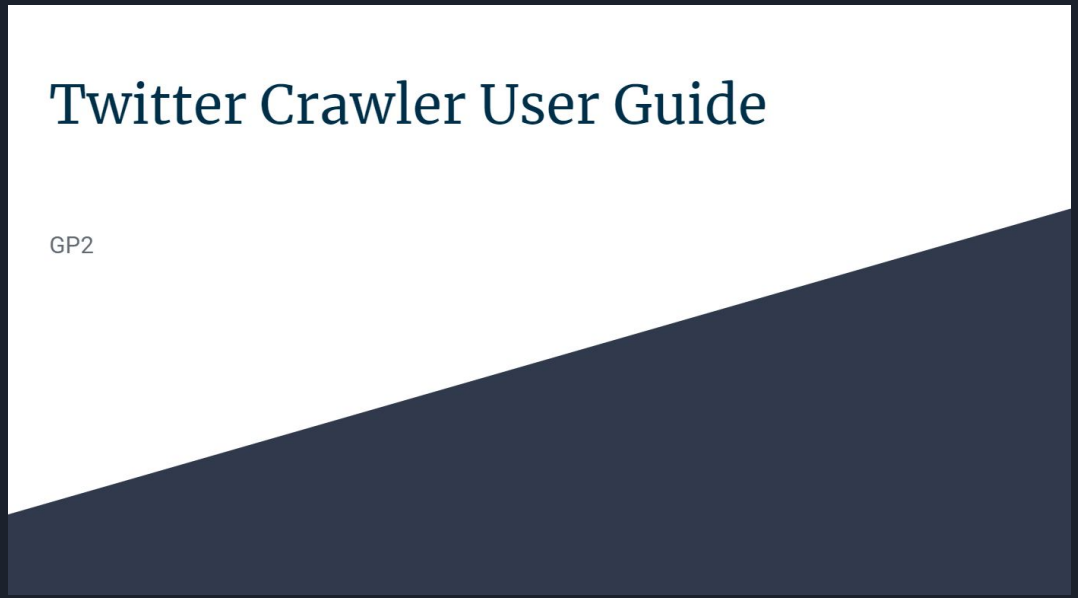
Contact us

Eric:
panlau520@gmail.com

Leo:
shadow02041@gmail.com

Sing:
lokaising9@gmail.com

Appendix 1: User Guide



Q&A Section

Thank you!