**Basic Work Flow And Execution of Twitter\_Scraping\_Code\_Using\_Snscrape**

Data scraping involves pulling information out of a website and into a spreadsheet.

In our code we are going to scrape the data like date, id, url, tweet content, user, reply count, retweet count, language, source, like count from social media twitter using snscrape.

**Step 1:**   
Import all the required modules.

**import** streamlit **as** st  
**import** snscrape.modules.twitter **as** sntwitter  
**import** pandas **as** pd  
**import** datetime  
**from** pymongo **import** MongoClient

**Step 2:**

Fetch the required inputs from the user that is **keyword or Hashtag** to be searched, **date range** and **limit** of tweet count that need to be scraped.

st.title(**":blue[TWITTER SCRAPER USING SNSCRAPE]"**)  
search\_word = st.text\_input(**"\*\*:green[Enter Keyword or Hashtag to be searched: ]\*\*"**, **"python"**)  
from\_date = st.date\_input(**"\*\*:green[Select From-Date:]\*\*"**, max\_value=datetime.datetime.now())  
to\_date = st.date\_input(**"\*\*:green[Select To-Date:]\*\*"**, max\_value=datetime.datetime.now())  
limit = st.slider(**"\*\*:green[Set Tweet Scrape Limit]\*\*"**, 1, 1000, 100)  
query = search\_word + **' since:'** + str(from\_date) + **' until:'** + str(to\_date)

**Step 3:**   
Scrape the twitter data using snscrape by providing ‘limit’ and ‘query’ parameters to the twitter\_scraper function. This function will return the scraped data in the dataframe format, which can be viewed on screen with the help of streamlit command - st.dataframe().

**def** twitter\_scraper(limit\_fm, query\_fm):tweets\_list = []  
 **for** tweet **in** sntwitter.TwitterSearchScraper(query\_fm).get\_items():  
 **if** len(tweets\_list) > limit\_fm:  
 **break  
 else**:  
 tweets\_list.append(  
 [tweet.date, tweet.id, tweet.url, tweet.content, tweet.user, tweet.replyCount, tweet.retweetCount,  
 tweet.lang, tweet.source, tweet.likeCount])  
 cols = [**"date"**, **"id"**, **"url"**, **"tweet content"**, **"user"**, **"reply count"**, **"retweet count"**, **"language"**,  
 **"source"**, **"like count"**]  
 tweets\_df\_tm = pd.DataFrame(tweets\_list, columns=cols)  
 **return** tweets\_df\_tm

tweets\_df = twitter\_scraper(limit, query)  
st.subheader(**"\*\*:violet[Scraped Twitter data for '"**+search\_word+**"' from "**+str(from\_date)+**" to "**+str(to\_date)+**".]\*\*"**)  
st.dataframe(tweets\_df)

**Step 4:**   
Optionally with the help of Pymongo library we can store the data in MongoDB database by clicking the **“Upload Data”** Button, and providing required parameters to data\_upload() function.

**def** data\_upload(search\_word\_fm, tweets\_df\_fm, db\_host=**'localhost'**, db\_port=27017, user\_database=**'twitter\_db'**,  
 user\_collection=**'twitter\_search\_collection'**):  
client = MongoClient(db\_host, db\_port) db = client[user\_database] collection = db[user\_collection] keyword = search\_word\_fm + str(datetime.datetime.now())  
 tweet\_dict = tweets\_df\_fm.to\_dict(**'records'**)  
 collection.insert\_one({keyword: tweet\_dict})  
 st.success(**"Upload Successful"**)

**if** st.button(**"Upload Data"**):  
 *# Loading Data into MongoDB DataBase* data\_upload(search\_word, tweets\_df)

**Note :**

MongoDB Default Connection Setting:   
• host name : 'localhost' (function parameter - db\_host)   
• port number : 27017 (function parameter - db\_port)   
• database name : 'twitter\_db' (function parameter - user\_database)   
• collection name : 'twitter\_search\_collection'(function parameter - user\_collection)

You can set your own connection detail, database name and collection name by providing parameters to data\_upload() function or by changing the values of keyword arguments in script.

**Step 5:**   
Optionally the pandas dataframe of twitter Scraped data can be downloaded in csv or json format by clicking the **"Download data as CSV"** Button or **"Download data as JSON"** Button respectively. This function was implemented with the help of dataframe.to\_csv()and dataframe.to\_json() module of pandas along with streamlit method.

st.write(**"\*\*:green[Click to Save file in .CSV format]\*\*"**)  
st.download\_button(label=**"Download data as CSV"**, data=tweets\_df.to\_csv(), file\_name=**'data\_csv.csv'**)  
st.write(**"\*\*:green[Click to Save file in .JSON format]\*\*"**)  
st.download\_button(label=**"Download data as JSON"**, data=tweets\_df.to\_json(), file\_name=**'data\_json.json'**)