CS5540 Summer 2022 Project Phase 1 - Report

Github Link: https://github.com/ShriRamyaA/Big_Data_Management

Team 2:
Jack Zhang, Salam Othman, Shri Ramya Ashok, Ye Wang

→ Section 1: Lessons Learnt

- In this project we learnt how to link our twitter development account to production and how to authenticate it in order to pull data from a live stream.
- Not all the twitter developer account requests get approved. The approval strategy is very random. One of the two requests by the same person got rejected and another accepted.
- We also learnt how to make specific commands to filter out selected information.
- Since the data that we pulled was semi structured, we learnt that we needed to parse that data to later be able to locate and extract the urls and the hashtags.
- In Spark, we treated each line in the file as a string and an element of Spark RDD. Then, we moved to splitting the strings into words for the count.
- Configuration of the XML file is of utmost importance after installing Hadoop before running the WordCount example.
- Sometimes when the 'start-all.sh' command is run, namenode or datanode might not have started successfully. Always use the 'jps' command to check that all the Hadoop daemons are running successfully. Hadoop daemons: NameNode, DataNode, SecondaryNameNNode, NodeManager, ResourceManager.
- We learnt that most of the times when hadoop related commands are not run successfully, it was because of permission issues.
- Introduced to 'HDFS' file system commands.
- Hadoop installation was fairly easy. It was the configuration of xml files and creation of directories that took effort and time.

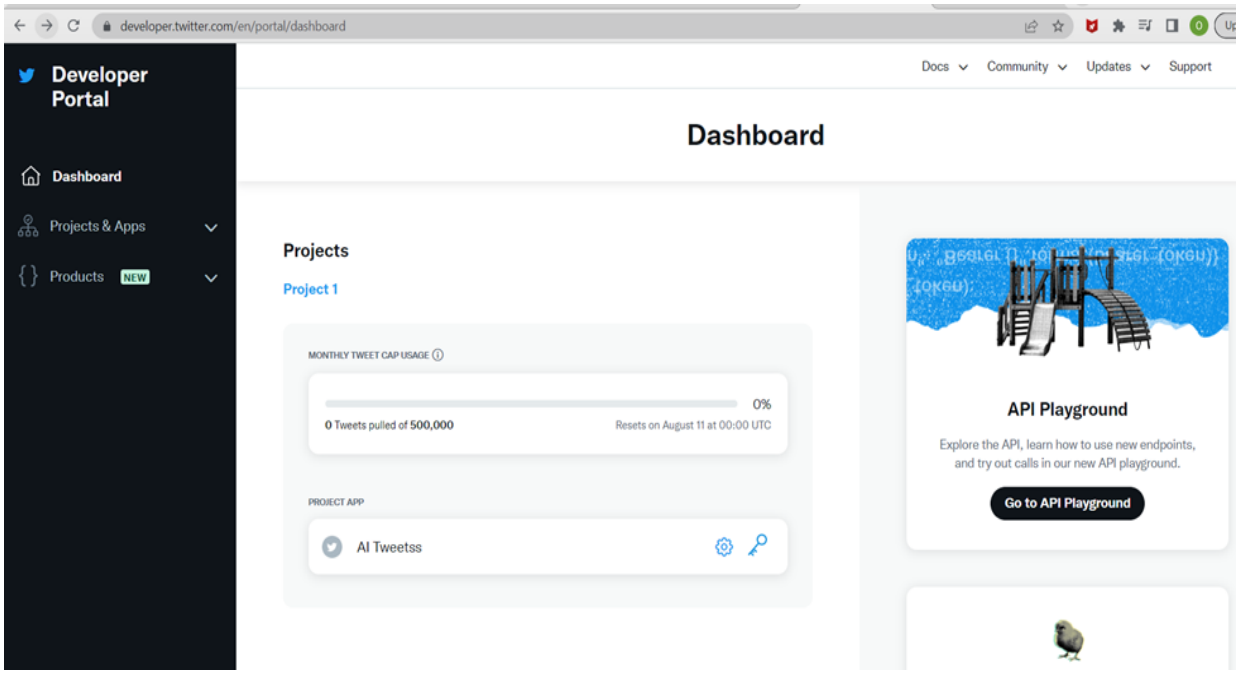
→ Section 2: Accomplishments

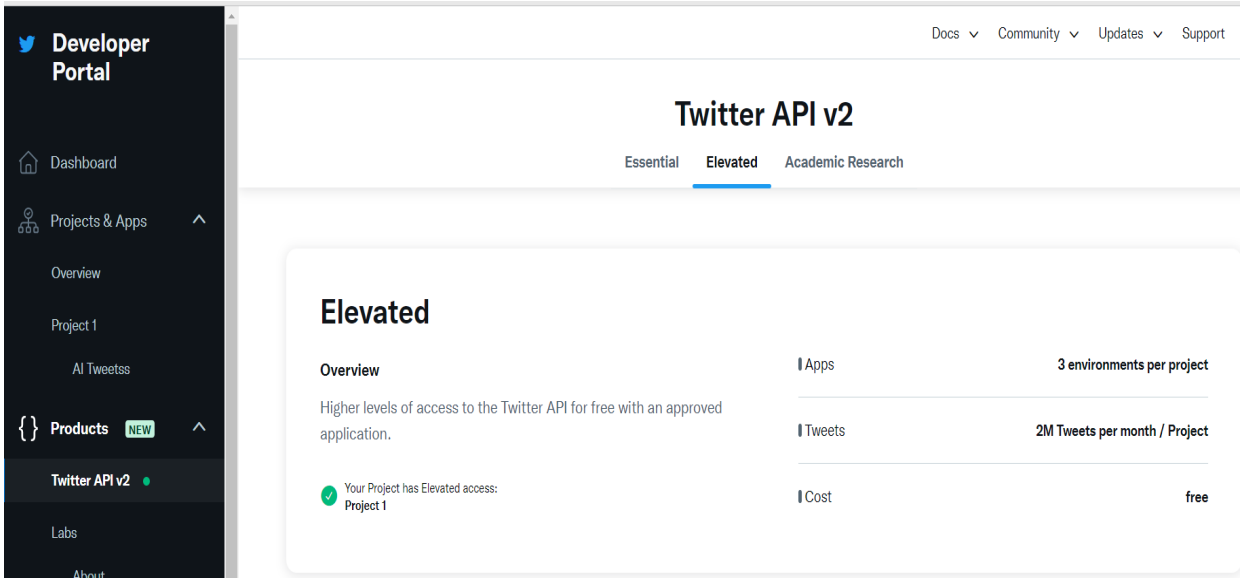
- Approved for a developer account with twitter.
- Linked a social media platform to a database.
- Processed a semi- structured data.
- Learned to work with some Spark related functions.
- Able to successfully install hadoop in the VM.
- Able to run the MapReduce function in Hadoop with both hashtag and url input and get the wordcount output.
- Browsed hdfs file system directories using hdfs commands.

→ Section 3: Detailed explanation on the work done (with screenshots)

1. Collect Tweets using Twitter's Streaming APIs

→ First, we created an app on Twitter and signed up for an Elevated account to earn access to the development environment to extract tweets.





→ Next, we used the Twitter Streaming API to collect tweets. In Jupyter Notebook, we imported 'Tweepy' and we created values to feed the API key and key secret as well as the access token and access token secret from our Twitter account.

In [1]:
```
#source: https://radimrehurek.com/gensim/models/ldamodel.html
import pandas as pd
import os
# Check the working directory
os.getcwd()
```

```
C:\Users\wanye\Anaconda3\lib\site-packages\numpy\_distributor_init.py:30: UserWarning: loaded more than 1 DLL from .libs:
C:\Users\wanye\Anaconda3\lib\site-packages\numpy\.libs\libopenblas.WCDJNK7YVMPZQ2ME2ZZHJJRJ3JIKNDB7.gfortran-win_amd64.dll
C:\Users\wanye\Anaconda3\lib\site-packages\numpy\.libs\libopenblas.XWYDX2IKJW2NMTWSFYNGFUWKQU3LYTCZ.gfortran-win_amd64.dll
  warnings.warn("loaded more than 1 DLL from .libs:"
```

Out[1]:
```
'C:\\Users\\wanye'
```

In [2]:
```
os.chdir('C:\\Users\\wanye\\Documents\\5540_project')
```

In [17]:
```
!pip install tweepy
```

```
Requirement already satisfied: tweepy in c:\users\wanye\anaconda3\lib\site-packages (4.10.0)
Requirement already satisfied: oauthlib<4,>=3.2.0 in c:\users\wanye\anaconda3\lib\site-packages (from tweepy) (3.2.0)
Requirement already satisfied: requests-oauthlib<2,>=1.2.0 in c:\users\wanye\anaconda3\lib\site-packages (from tweepy) (1.3.0)
Requirement already satisfied: requests<3,>=2.27.0 in c:\users\wanye\anaconda3\lib\site-packages (from tweepy) (2.27.1)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in c:\users\wanye\anaconda3\lib\site-packages (from requests<3,>=2.27.0->t
weepy) (1.26.8)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\wanye\anaconda3\lib\site-packages (from requests<3,>=2.27.0->twee
py) (2021.10.8)
Requirement already satisfied: charset-normalizer~=2.0.0 in c:\users\wanye\anaconda3\lib\site-packages (from requests<3,>=2.27.
0->tweepy) (2.0.4)
Requirement already satisfied: idna<4,>=2.5 in c:\users\wanye\anaconda3\lib\site-packages (from requests<3,>=2.27.0->tweepy)
(3.3)
```

→ Using tweepy, we authenticated the API key and key secret. We also used the stream function on all the keys and tokens to pull the current stream of tweets. We decided to pull the tweets related to “gas price” only, so we specified that as a keyword. We pulled 10,000 tweets related to “gas price” using a for loop and appended those tweets to the tweet_cursor list that we created.

In [3]:
```
import tweepy
from tweepy import Stream
#from tweepy.streaming import StreamListener
```

In [ ]:
```
consumer_key='MyKey'
consumer_secret='MyScrete'
access_token='MyToken'
access_token_secret='MyTokenScrete'
```

In [5]:
```
auth = tweepy.AppAuthHandler(consumer_key, consumer_secret)
api = tweepy.API(auth)
```

In [6]:
```
myStream=tweepy.Stream (consumer_key, consumer_secret, access_token, access_token_secret)
```

Cursor Approach to Iteractively Download Tweets

In [7]:
```
keywords='gas price'
limit=10000
tweets_cursor=[]
for tweet in tweepy.Cursor(api.search_tweets, q=keywords, count=100, tweet_mode='extended').items(limit):
    tweets_cursor.append(tweet)
```

2. Extract all hashtags and URLs in the tweets

→ In the tweets_cursor list, the tweets were collected in 'json' format. To organize the data, we created a for loop to write one tweet on a separate line. Later, for each line, we saved the content into the "data" list organized per columns for the tweet text, time of the tweet, the hashtags, the url, the source of the tweet and the user that inserted the tweet. Finally, we saved the data list to a csv file we named 'tweets_10K" using pandas.

Extract URLs, hastags, created_at, source, and user information

```
for tweet in tweets_cursor:
            with open("results.jsonl", "w+" ) as f:
                # Here we are writing 1 Tweet object JSON per line
                f.write(json.dumps(tweet) + "\n")
```

In [9]:
```
columns=['tweet','created_at','hashtags','urls','source','user']
data=[]

for tweet in tweets_cursor:
    data.append([tweet.full_text,tweet.created_at,
                 tweet.entities['hashtags'],tweet.entities['urls'],
                 tweet.source, tweet.user
                ])

df=pd.DataFrame(data, columns=columns)
df.head()
```

Out[9]:

	tweet	created_at	hashtags	urls	source	user
0	RT @JoeBiden: The price of oil is down about 2...	2022-07-15 20:42:37+00:00	[]	[]	Twitter for iPad	User(_api=<tweepy.api.API object at 0x000001B0...
1	@Jacob_Rees_Mogg I believe the price for gas i...	2022-07-15 20:42:33+00:00	[]	[]	Twitter Web App	User(_api=<tweepy.api.API object at 0x000001B0...
2	RT @Sbh08Mae: @RepKimSchrier knows gas & o...	2022-07-15 20:42:31+00:00	[{'text': 'WA08', 'indices': [107, 112]}]	[]	Twitter Web App	User(_api=<tweepy.api.API object at 0x000001B0...
3	RT @JoeBiden: The price of oil is down about 2...	2022-07-15 20:42:31+00:00	[]	[]	Twitter for iPad	User(_api=<tweepy.api.API object at 0x000001B0...
4	@RussianEmbassy @BBCRadio4 @mfa_russia @BBCWor...	2022-07-15 20:42:27+00:00	[]	[]	Twitter Web App	User(_api=<tweepy.api.API object at 0x000001B0...

In [10]:
```
df.to_csv('tweets_10k.csv')
```

→ To Extract the URLs, we looped into the url column of our dataset and we extracted the urls and saved them into a new urls list that we created. We saved that list in a csv format under "urls.txt".

 # Extract URLs

In [52]:
```
urls_record=df['urls']
urls=[]
for record in urls_record:
    if record!=[]:
        for item in record:
            print(item['url'])
            urls.append(item['url'])
```

```
https://t.co/QZ46Vc3AKl
https://t.co/wFvIahkwFX
https://t.co/BeIiuLfvab
https://t.co/iHVUaYWfAF
https://t.co/1XmEj1aIjY
https://t.co/3VXOjla12N
https://t.co/5QYA6SEJ6H
https://t.co/T8wwkghzy1
https://t.co/I1hCdFXVt1
https://t.co/3VXOjla12N
https://t.co/Q7Rr2Epxak
https://t.co/5QYA6SEJ6H
https://t.co/CQlg4Qs3P6
https://t.co/ajAHBu1Kae
https://t.co/5QYA6SEJ6H
https://t.co/5QYA6SEJ6H
https://t.co/3VXOjla12N
https://t.co/5QYA6SEJ6H
https://t.co/5QYA6SEJ6H
```

In [53]:
```
df_urls = pd.DataFrame(urls)
df_urls.to_csv('urls.txt', header=None, index=False)
```

→ Similarly, for the hashtags, we looped into our dataframe, extracted the hashtags and saved them in csv file name “hashtags.txt”

Extract Hashtags

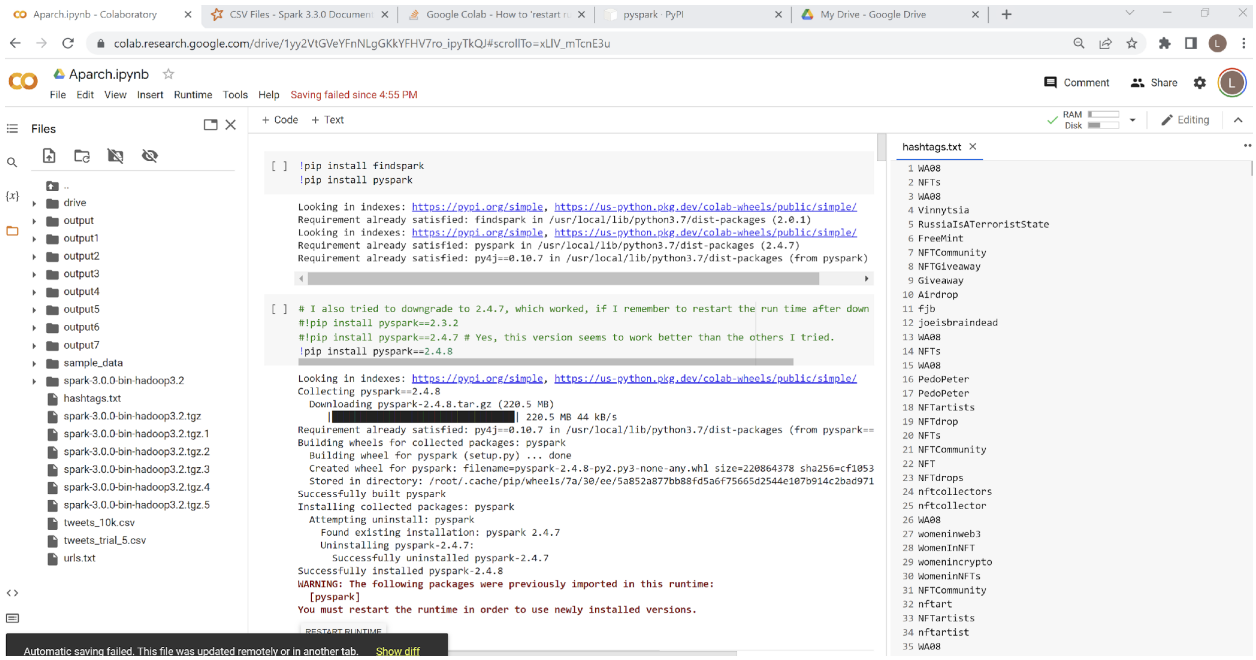
In [58]:
```
hashtag_records=df['hashtags']
hashtags=[]
for record in hashtag_records:
    if record!=[]:
        for item in record:
            print(item['text'])
            hashtags.append(item['text'])
```

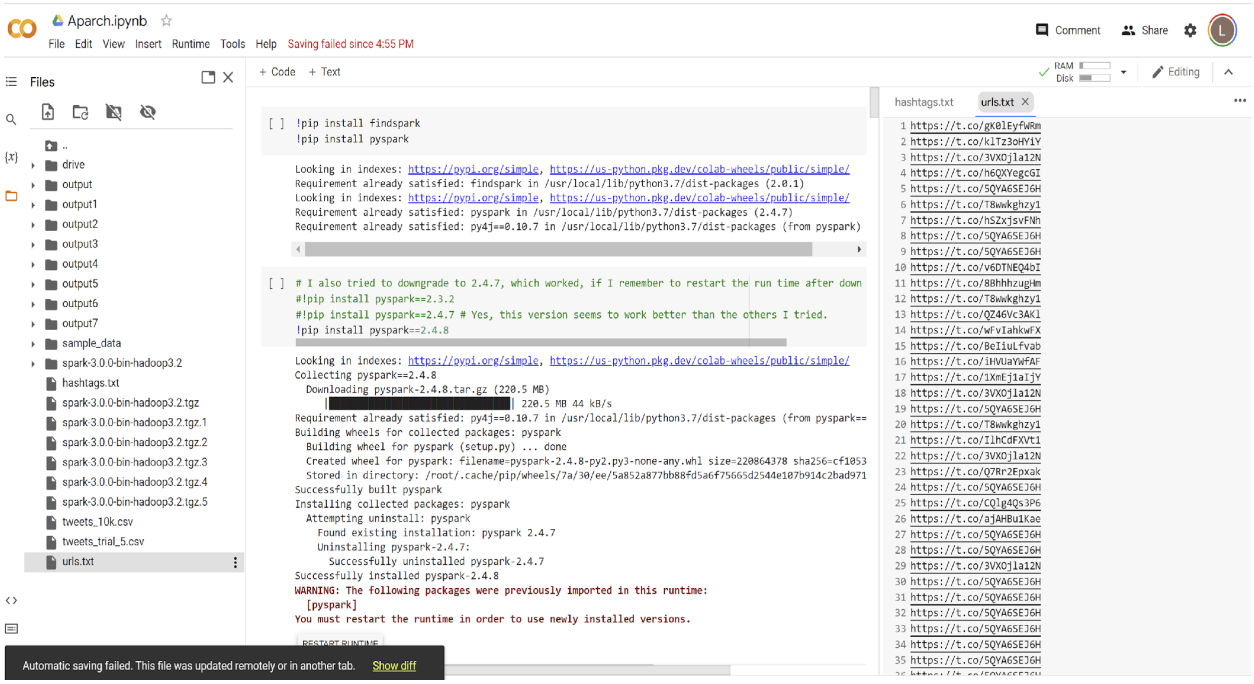
```
WA08
NFTs
WA08
Vinnytsia
RussiaIsATerroristState
FreeMint
NFTCommunity
NFTGiveaway
Giveaway
Airdrop
fjb
joeisbraindead
WA08
NFTs
WA08
PedoPeter
PedoPeter
NFTartists
NFTdrop
```

In [59]:
```
df_hashtags = pd.DataFrame(hashtags)
df_hashtags.to_csv('hashtags.txt', header=None, index=False)
```

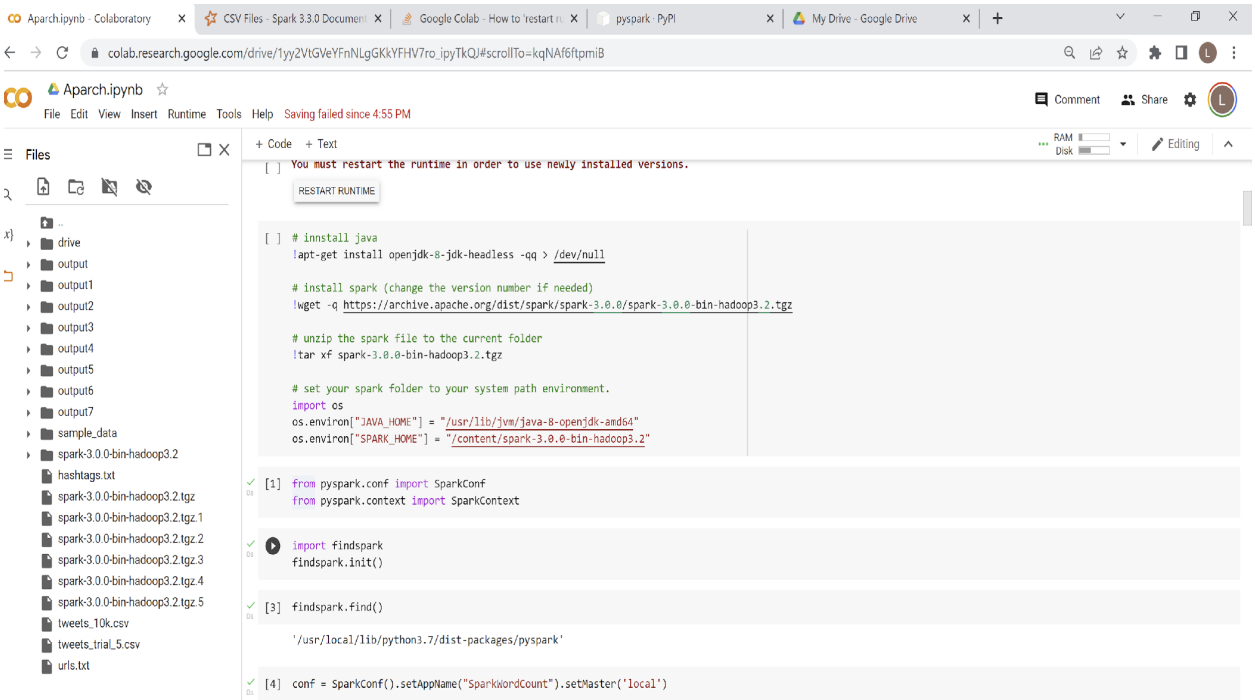
3. WordCount example in Spark

→ In Google colab, we made sure that ‘findspark’ and ‘pyspark’ are installed. We transferred the hashtags.txt and urls.txt files to Colab Google files and we made sure they are reading correctly.

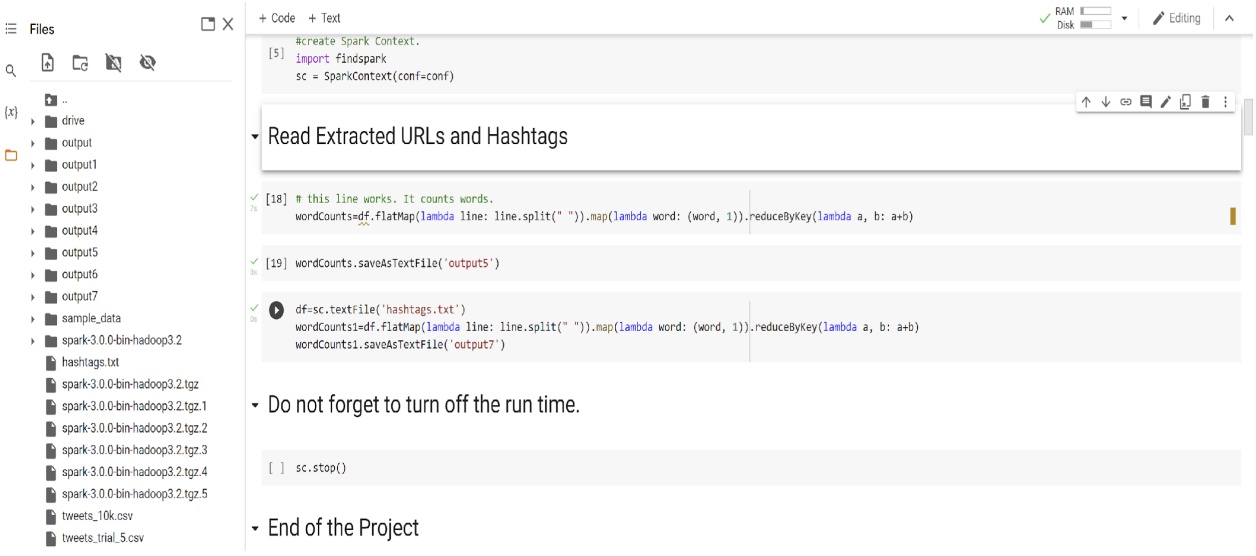


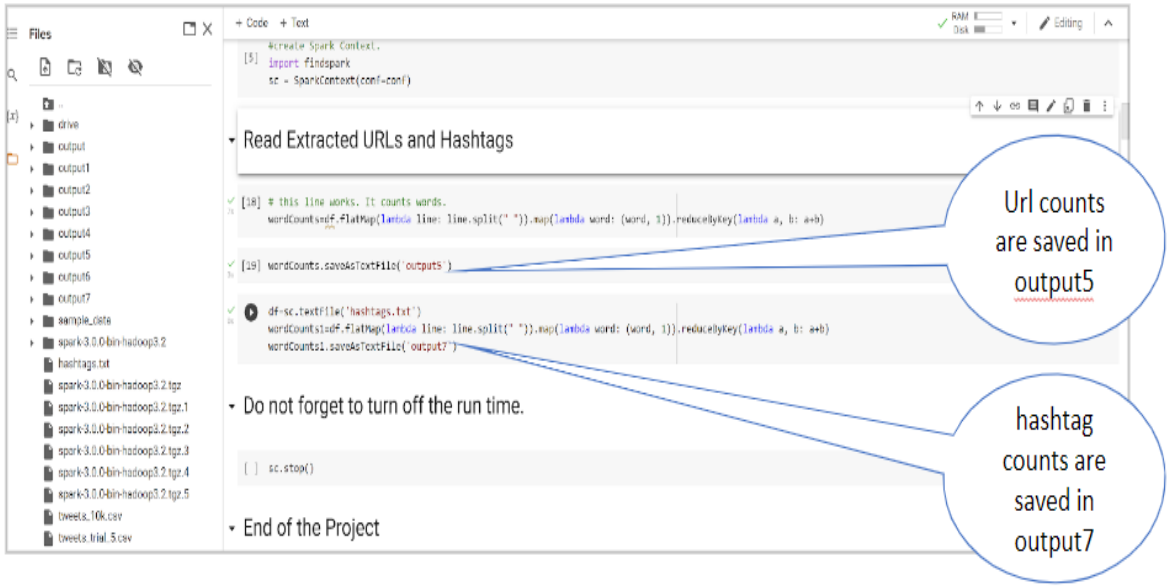


→Next, we configured the spark folder to our system path environment

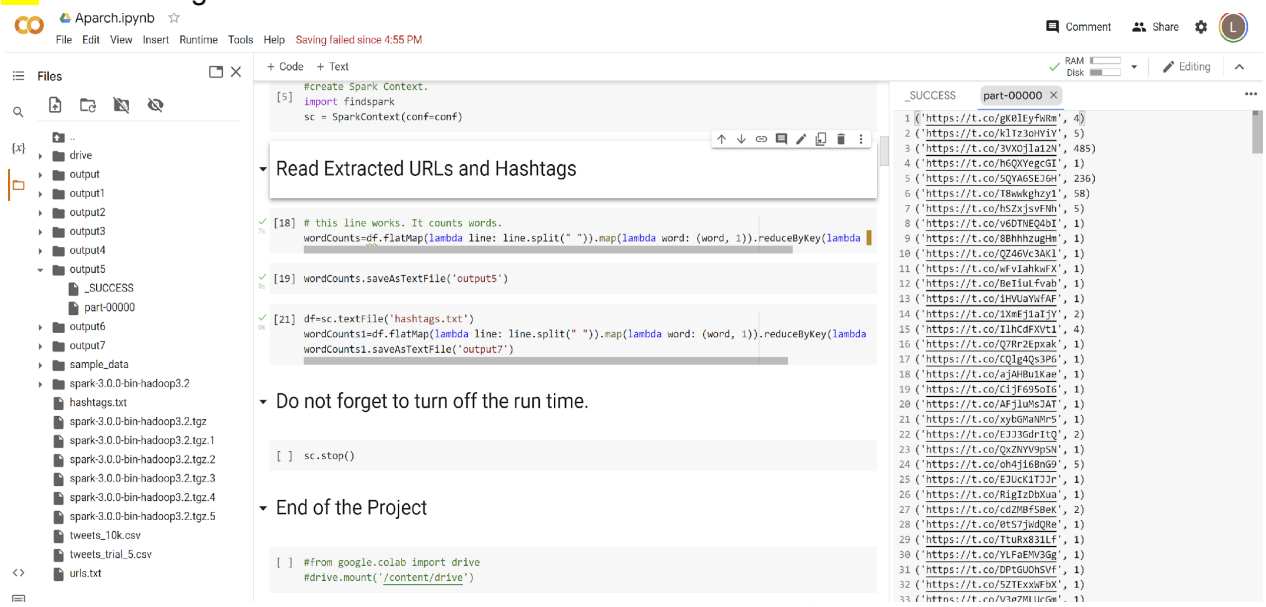


→Count the words in spark using Pyspark and save the count

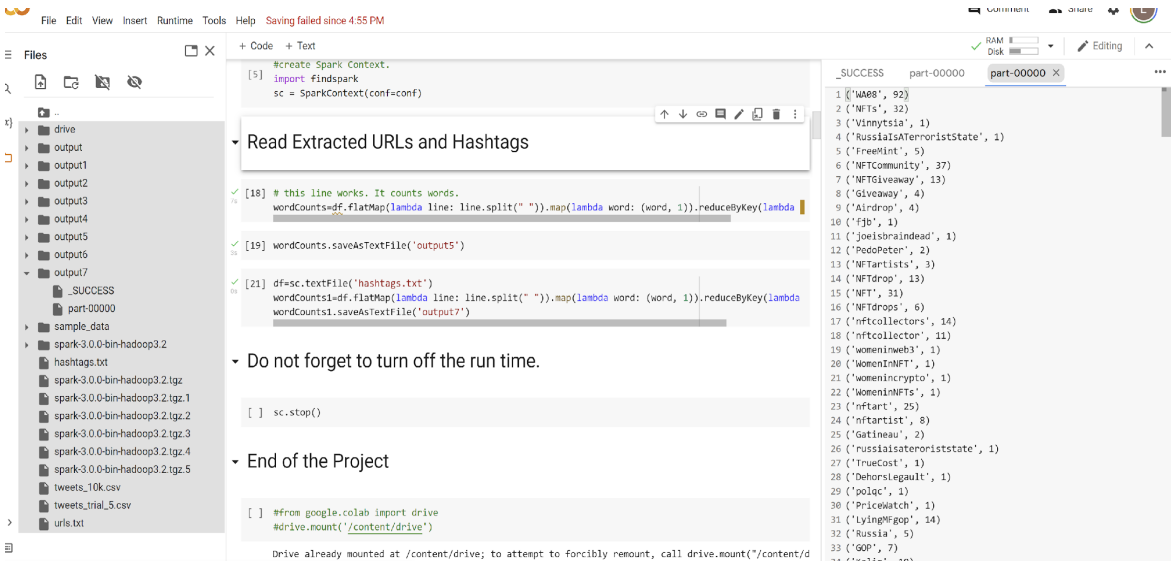




→Show the log file for the URLs



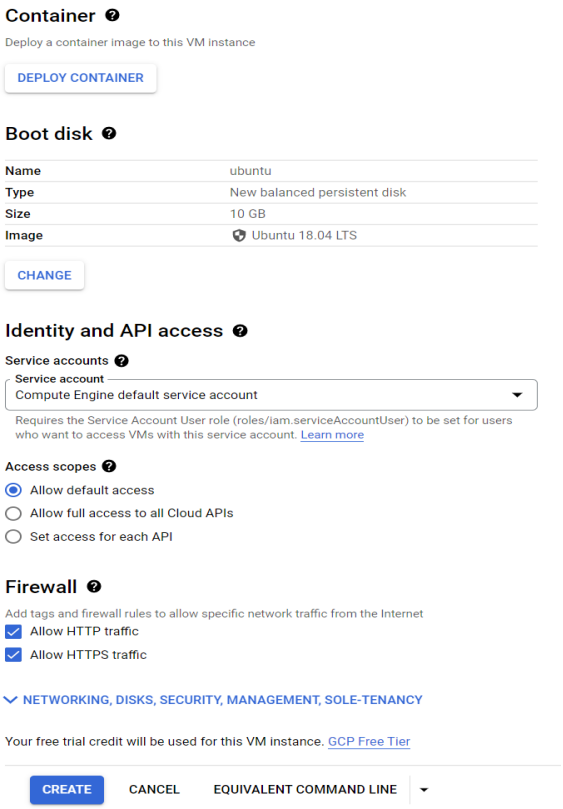
→Show the log file for the hashtags



4. WordCount example in Hadoop

→ We are using a Virtual Machine instance in GCP(Google Cloud Platform) to install Hadoop and Run the WordCount program.

→ Create VM Instance. Ubuntu is selected as the operating system.



→ Install Java.

Command:

sudo apt-get install default-jdk

```
sachg@ubuntu:~$ sudo update-alternatives --config java
There is only one alternative in link group java (providing /usr/bin/java): /usr/lib/jvm/java-11-openjdk-amd64/bin/java
Nothing to configure.
sachg@ubuntu:~$ sudo update-alternatives --config javac
There is only one alternative in link group javac (providing /usr/bin/javac): /usr/lib/jvm/java-11-openjdk-amd64/bin/javac
Nothing to configure.
sachg@ubuntu:~$ java -version
openjdk version "11.0.15" 2022-04-19
OpenJDK Runtime Environment (build 11.0.15+10-Ubuntu-0ubuntu0.18.04.1)
OpenJDK 64-Bit Server VM (build 11.0.15+10-Ubuntu-0ubuntu0.18.04.1, mixed mode, sharing)
sachg@ubuntu:~$
```

→ Add dedicated hadoop user. This allows all the installation to be insulated from the rest of the environment.

Command:

sudo adduser –ingroup hadoop hduser

sudo adduser hduser sudo

```
sachg@ubuntu:~$ sudo adduser --ingroup hadoop hduser
Adding user `hduser' ...
Adding new user `hduser' (1002) with group `hadoop' ...
Creating home directory `/home/hduser' ...
Copying files from `/etc/skel' ...
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
Changing the user information for hduser
Enter the new value, or press ENTER for the default
        Full Name []: Shri Ramya Ashok
        Room Number []:
        Work Phone []:
        Home Phone []:
        Other []:
Is the information correct? [Y/n] y
sachg@ubuntu:~$ sudo adduser hduser sudo
Adding user `hduser' to group `sudo' ...
Adding user hduser to group sudo
Done.
sachg@ubuntu:~$ groups hduser
hduser : hadoop sudo
sachg@ubuntu:~$
```

→ Install SSH. The hadoop control scripts rely on SSH to peform cluster-wide operations. For example, there is a script for stopping and starting all the daemons in the clusters. We have to generate an SSH key for the hduser user.

Command:

sudo apt-get install ssh

su hduser

ssh-keygen -t rsa -P ""

$HOME/.ssh/authorized_keys

ssh localhost

```
sachg@ubuntu:~$ sudo apt-get install ssh
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following package was automatically installed and is no longer required:
  libnuma1
Use 'sudo apt autoremove' to remove it.
The following NEW packages will be installed:
  ssh
0 upgraded, 1 newly installed, 0 to remove and 7 not upgraded.
Need to get 5192 B of archives.
After this operation, 108 kB of additional disk space will be used.
Get:1 http://us-central1.gce.archive.ubuntu.com/ubuntu bionic-updates/main amd64 ssh
Fetched 5192 B in 0s (42.0 kB/s)
Selecting previously unselected package ssh.
(Reading database ... 69441 files and directories currently installed.)
Preparing to unpack .../ssh_1%3a7.6p1-4ubuntu0.7_all.deb ...
Unpacking ssh (1:7.6p1-4ubuntu0.7) ...
Setting up ssh (1:7.6p1-4ubuntu0.7) ...
sachg@ubuntu:~$ which ssh
/usr/bin/ssh
sachg@ubuntu:~$ which sshd
/usr/sbin/sshd
sachg@ubuntu:~$ su hduser
Password:
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

hduser@ubuntu:/home/sachg$ ssh-keygen -t rsa -P ""
Generating public/private rsa key pair.
Enter file in which to save the key (/home/hduser/.ssh/id_rsa):
Created directory '/home/hduser/.ssh'.
Your identification has been saved in /home/hduser/.ssh/id_rsa.
Your public key has been saved in /home/hduser/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:z/c2Nb1dVykI6kHjaOeazINvBlK0YJcBtwzfa5ToZXg hduser@ubuntu
The key's randomart image is:
+---[RSA 2048]----+
| o.oo            |
| o=+= .o .       |
|. +*.E+ o . .   .|
|  .o=o.=   . . ..|
|  ...o+ S     . o|
| . ..  o o     .=|
|  . = o   o .  .*|
|   . O     . .o..|
|    +..      ... |
+----[SHA256]-----+
hduser@ubuntu:/home/sachg$ cat $HOME/.ssh/id_rsa.pub >> $HOME/.ssh/authorized_keys
hduser@ubuntu:/home/sachg$ ssh localhost
```

→ Install Hadoop

Command:

sudo mkdir -p /usr/local/hadoo

wget

http://mirrors.sonic.net/apache/hadoop/common/hadoop-3.3.3/hadoop-3.3.3.tar.g

z

```
hduser@ubuntu:~$ sudo mkdir -p /usr/local/hadoop
[sudo] password for hduser:
hduser@ubuntu:~$ wget http://mirrors.sonic.net/apache/hadoop/common/hadoop-3.3.3/hadoop-3.3.3.tar.gz
--2022-07-16 00:39:31--  http://mirrors.sonic.net/apache/hadoop/common/hadoop-3.3.3/hadoop-3.3.3.tar.gz
Resolving mirrors.sonic.net (mirrors.sonic.net)... 157.131.224.201, 2001:5a8:601:4003:157:131:224:201
Connecting to mirrors.sonic.net (mirrors.sonic.net)|157.131.224.201|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 645040598 (615M) [application/x-gzip]
Saving to: 'hadoop-3.3.3.tar.gz'

hadoop-3.3.3.tar.gz                                    100%[=====================================

2022-07-16 00:39:42 (59.7 MB/s) - 'hadoop-3.3.3.tar.gz' saved [645040598/645040598]

hduser@ubuntu:~$
```

→ Set read/write permission

Command:

sudo chown -R hduser:hadoop /usr/local/hadoop

```
hduser@ubuntu:~$ sudo mv * /usr/local/hadoop/
hduser@ubuntu:~$ sudo chown -R hduser:hadoop /usr/local/hadoop
```

→ Setup configuration files

→ Add this to bashrc file at the end of the file:

#HADOOP VARIABLES START export

JAVA_HOME=/usr/lib/jvm/java-11-openjdk-amd64 export

HADOOP_INSTALL=/usr/local/hadoop export

PATH=$PATH:$HADOOP_INSTALL/bin export

PATH=$PATH:$HADOOP_INSTALL/sbin export

HADOOP_MAPRED_HOME=$HADOOP_INSTALL export

HADOOP_COMMON_HOME=$HADOOP_INSTALL export

HADOOP_HDFS_HOME=$HADOOP_INSTALL export

YARN_HOME=$HADOOP_INSTALL export

HADOOP_COMMON_LIB_NATIVE_DIR=$HADOOP_INSTALL/lib/native export

HADOOP_OPTS="-Djava.library.path=$HADOOP_INSTALL/lib" export

HADOOP_HOME_WARN_SUPPRESS=1 export

HADOOP_ROOT_LOGGER="WARN,DRFA" #HADOOP VARIABLES END

→ Add JAVA_HOME to hadoop-env.sh file

export JAVA_HOME==/usr/lib/jvm/java-11-openjdk-amd64

→ Create a tmp directory and change the permissions

Command:

sudo mkdir -p /app/hadoop/tmp

sudo chown hduser:hadoop /app/hadoop/tmp

→ Add the following to core-site.xml file within <configuration></configuration> tag

<property>

<name>hadoop.tmp.dir </name>

<value>/app/hadoop/tmp</value>

<description>A base for other temporary directories.</description>

</property>

<property>

<name>fs.deafult.name</name>

<value>hdfs://localhost:54310</value>

<description>>The name of the default file system. A URI whose scheme and authority determine the FileSystem implementation. The uri's scheme determines the config property (fs.SCHEME.impl) naming the FileSystem implementation class. The uri's authority is used to determine the host, port, etc. for a filesystem.</description>

</property>

```
hduser@ubuntu:~$ sudo chown -R hduser:hadoop /usr/local/hadoop
hduser@ubuntu:~$ vim ~/.bashrc
hduser@ubuntu:~$ source !/.bashrc
-bash: !/.bashrc: event not found
hduser@ubuntu:~$ source ~/.bashrc
hduser@ubuntu:~$ vi /usr/local/hadoop/etc/hadoop/hadoop-env.sh
hduser@ubuntu:~$ vi /usr/local/hadoop/etc/hadoop/hadoop-env.sh
hduser@ubuntu:~$ vim /usr/local/hadoop/etc/hadoop/hadoop-env.sh
hduser@ubuntu:~$ sudo vim /usr/local/hadoop/etc/hadoop/hadoop-env.sh
hduser@ubuntu:~$ sudo vi /usr/local/hadoop/etc/hadoop/hadoop-env.sh
hduser@ubuntu:~$ sudo vi /usr/local/hadoop/etc/hadoop/hadoop-env.sh
hduser@ubuntu:~$ ls
hduser@ubuntu:~$ cd /usr/local/hadoop
hduser@ubuntu:/usr/local/hadoop$ ls
hadoop-3.3.3  hadoop-3.3.3.tar.gz
hduser@ubuntu:/usr/local/hadoop$ cd hadoop-3.3.3
hduser@ubuntu:/usr/local/hadoop/hadoop-3.3.3$ ls
LICENSE-binary  LICENSE.txt  NOTICE-binary  NOTICE.txt  README.txt  bin  etc  include  lib  libexec
hduser@ubuntu:/usr/local/hadoop/hadoop-3.3.3$ cd ..
hduser@ubuntu:/usr/local/hadoop$ cd etc
-bash: cd: etc: No such file or directory
hduser@ubuntu:/usr/local/hadoop$ cd hadoop-3.3.3
hduser@ubuntu:/usr/local/hadoop/hadoop-3.3.3$ mv * /usr/local/hadoop/
hduser@ubuntu:/usr/local/hadoop/hadoop-3.3.3$ cd ..
hduser@ubuntu:/usr/local/hadoop$ ls
LICENSE-binary  LICENSE.txt  NOTICE-binary  NOTICE.txt  README.txt  bin  etc  hadoop-3.3.3  hadoop-
hduser@ubuntu:/usr/local/hadoop$ cd ..
hduser@ubuntu:/usr/local$ cd ..
hduser@ubuntu:/usr$ cd ..
hduser@ubuntu:/$ sudo vi /usr/local/hadoop/etc/hadoop/hadoop-env.sh
hduser@ubuntu:/$ sudo mkdir -p /app/hadoop/tmp
hduser@ubuntu:/$ sudo chown hduser:hadoop /app/hadoop/tmp
hduser@ubuntu:/$ vi /usr/local/hadoop/etc/hadoop/core-site.xml
```

→ Add this to the mapred-site.xml file within <configuration></configuration> tag

<property>
<name>mapred.job.tracker</name>
<value>localhost:54311</value>
<description>The host and port that the MapReduce job tracker runs at. If "local", then jobs are run in-process as a single map and reduce task
</description>
</property>

→ Create namenode, datanode directories and change the permissions.

Command:

sudo mkdir -p /usr/local/hadoop_store/hdfs/namenode
sudo mkdir -p /usr/local/hadoop_store/hdfs/datanode
sudo chown -R hduser:hadoop /usr/local/hadoop_store

→ Add this to hdfs-site.xml file within <configuration></configuration> tag

<property>
<name>dfs.replication</name>
<value> 1 </value>
<description>Default block replication. The actual number of replications can be specified when the file is created. The default is used if replication is not specified in create time.</description>
</property>
<property>
<name>dfs.namenode.name.dir</name>
<value>file:/usr/local/hadoop_store/hdfs/namenode</value>
</property>
<property>
<name>dfs.datanode.data.dir</name>
<value>file:/usr/local/hadoop_store/hdfs/datanode</value>
</property>

→ hadoop namenode -format command should be executed once before we start using Hadoop

Command:

hadoop namenode –format

```
hduser@ubuntu:/$ vim /usr/local/hadoop/etc/hadoop/mapred-site.xml
hduser@ubuntu:/$ sudo mkdir -p /usr/local/hadoop_store/hdfs/namenode
hduser@ubuntu:/$ sudo mkdir -p /usr/local/hadoop_store/hdfs/datanode
hduser@ubuntu:/$ sudo chown -R hduser:hadoop /usr/local/hadoop_store
hduser@ubuntu:/$ vim /usr/local/hadoop/etc/hadoop/hdfs-site.xml
hduser@ubuntu:/$ hadoop namenode -format
WARNING: Use of this script to execute namenode is deprecated.
WARNING: Attempting to execute replacement "hdfs namenode" instead.

WARNING: /usr/local/hadoop/logs does not exist. Creating.
```

→ Start all processes. For checking the running process in our Hadoop cluster we use JPS command.JPS stands for Java Virtual Machine Process Status Tool.

Command:

Start-all.sh

jps

```
hduser@ubuntu:/$ start-all.sh
WARNING: Attempting to start all Apache Hadoop daemons as hduser in 10 seconds.
WARNING: This is not a recommended production deployment configuration.
WARNING: Use CTRL-C to abort.
Starting namenodes on [localhost]
Starting datanodes
Starting secondary namenodes [ubuntu]
ubuntu: Warning: Permanently added 'ubuntu,10.128.0.3' (ECDSA) to the list of known hosts.
Starting resourcemanager
Starting nodemanagers
hduser@ubuntu:/$ jps
9268 NodeManager
8567 DataNode
8392 NameNode
8825 SecondaryNameNode
9578 Jps
9099 ResourceManager
hduser@ubuntu:/$
```

→ WordCount execution

→ Create WordCount.java and compile it

WordCount.java code: Refer WordCount.java file

```
hduser@ubuntu:/$ cd /home/hduser
hduser@ubuntu:~$ vim WordCount.java
hduser@ubuntu:~$ vim WordCount.java
hduser@ubuntu:~$ hadoop com.sun.tools.javac.Main WordCount.java
```

→ Create jar

Command:

jar cf wc.jar WordCount*.class

```
hduser@ubuntu:~$ jar cf wc.jar WordCount*.class
```

→ Move the input file (hashtags.txt) to input dir

```
hduser@ubuntu:~$ hdfs dfs -mkdir -p input_dir
hduser@ubuntu:~$ vim hashtags.txt
hduser@ubuntu:~$ hdfs dfs -put hashtags.txt input dir
hduser@ubuntu:~$ hadoop fs -ls input_dir
Found 1 items
-rw-r--r--   1 hduser supergroup      11276 2022-07-16 01:09 input_dir/hashtags.txt
```

→ Run the application(input is hashtags.txt)

```
hduser@ubuntu:~$ hadoop jar wc.jar WordCount input_dir output
```

→ Log and Output files

```
hduser@ubuntu:~$ cd /usr/local/hadoop
hduser@ubuntu:/usr/local/hadoop$ ls
LICENSE-binary  LICENSE.txt  NOTICE-binary  NOTICE.txt  README.txt  bin  etc  hadoop-3.3.3  hadoop-3.3.3.tar.gz
include  lib  libexec  licenses-binary  logs  sbin  share
hduser@ubuntu:/usr/local/hadoop$ cd logs
hduser@ubuntu:/usr/local/hadoop/logs$ ls
SecurityAuth-hduser.audit          hadoop-hduser-datanode-ubuntu.out  hadoop-hduser-namenode-ubuntu.out     hadoo
p-hduser-nodemanager-ubuntu.out      hadoop-hduser-resourcemanager-ubuntu.out    hadoop-hduser-secondarynamenode-
ubuntu.out  userlogs
hadoop-hduser-datanode-ubuntu.log  hadoop-hduser-namenode-ubuntu.log  hadoop-hduser-nodemanager-ubuntu.log  hadoo
p-hduser-resourcemanager-ubuntu.log  hadoop-hduser-secondarynamenode-ubuntu.log  hadoop.log
hduser@ubuntu:/usr/local/hadoop/logs$ hdfs dfs -ls output
Found 2 items
-rw-r--r--   1 hduser supergroup          0 2022-07-16 01:11 output/_SUCCESS
-rw-r--r--   1 hduser supergroup       4461 2022-07-16 01:11 output/part-r-00000
```

→ Screenshot of output file (when input is hashtags file)

```
Dunmore      3
EFT    1
ETF    1
ETH    28
ETHGasPrice 2
EU     20
Election2022        1
Energy       1
EnergyCrisis        2
EnergyPurchasePlatform   1
Escape       1
Ethereum     36
Europe       1
FREEMINT     32
FTM    1
FTMGasPrice 1
FYP    2
Fantom       1
FarronCousins       1
Fed    1
Flagstaff    1
Ford   1
FranklinsClub       1
FreeMint     5
FreePress    1
Fresh 2
GOP    7
GOPAreWorthlessFascists 1
GOPLiesAboutEverything   1
GOPpies      1
GREED 1
GREEDFLATION        1
Gas    8
GasCrisis    1
GasPrice     5
GasPrices    2
GasSupply    8
Gatineau     2
Giveaway     4
GlobalRecession     1
GoodMorningCMSir    1
LyingMFgop   14
MAGA   2
MASS   1
MATIC 8
MONEY 1
MedHat       3
MikePapantonio     1
MobiizeForMidterms        13
MohammedBinSalman 1
MorningJoe   2
Morningjoe   4
NFT    31
NFTAirdrop   1
NFTArt       3
NFTCollectibles    2
NFTCollection      21
NFTCollectors      2
NFTCommmunity      4
NFTCommunity       37
NFTGiveaway 13
NFTGiveaways       14
NFTJapan     2
NFTMarketplace     2
NFTMinting   1
NFTProject   3
NFTProjects 1
NFTart       1
NFTartist    3
NFTartists   3
NFTartwork   1
NFTcollectibles    1
NFTcollections     4
NFTdrop      13
NFTdrops     6
NFTs   32
NFTsales     2
NFTshill     7
NFT宣伝枠     2
NY11DeservesBetter        1
NaturalGas   2
NetZero      1
```

→ Urls

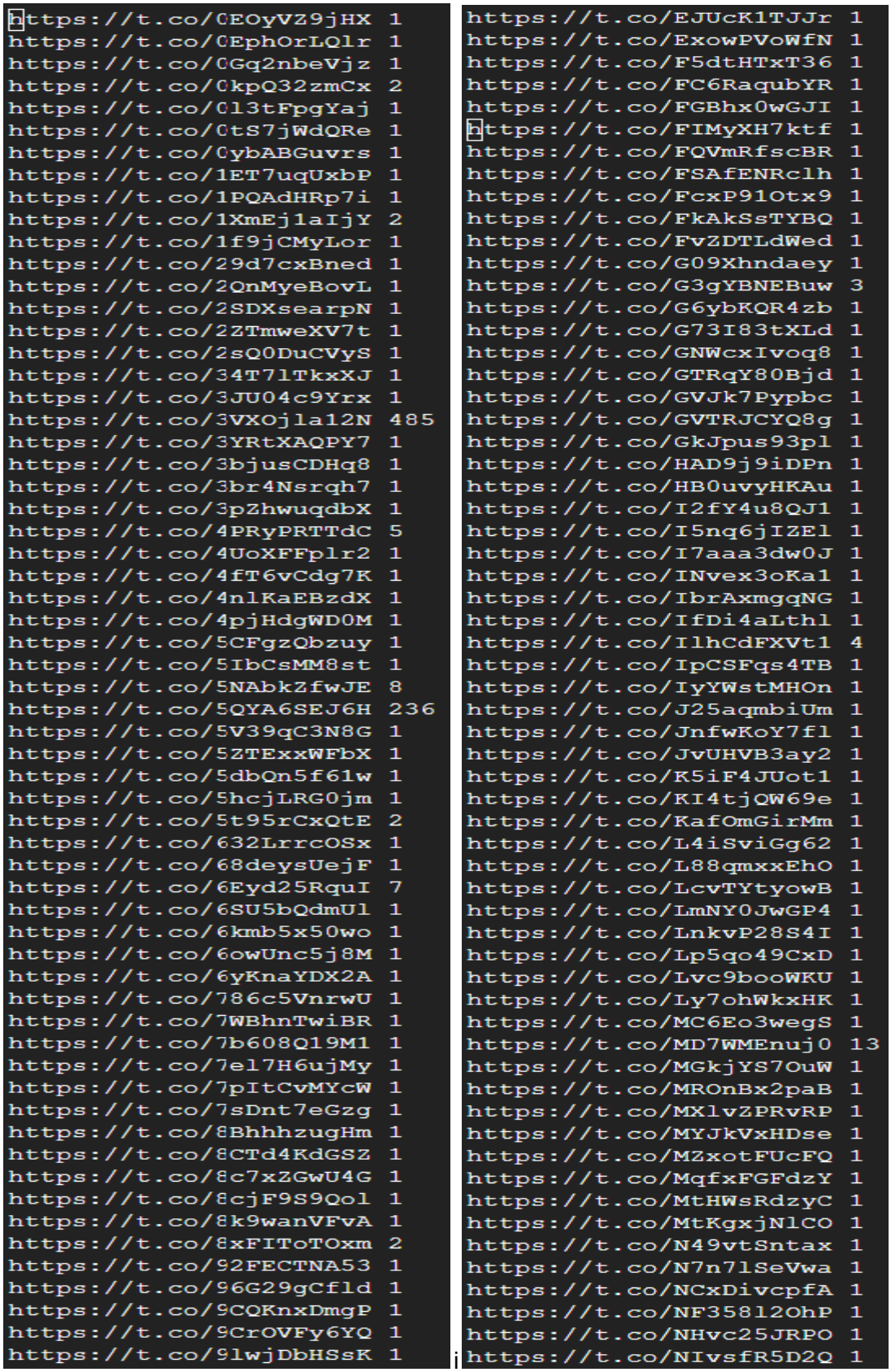
Run the application(input is ursl.txt)

```
hduser@ubuntu:~$ hdfs dfs -put urls.txt input_dir
hduser@ubuntu:~$ hadoop fs -ls input_dir
Found 1 items
-rw-r--r--   1 hduser supergroup      35150 2022-07-17 14:35 input_dir/urls.txt
hduser@ubuntu:~$ hadoop jar /home/hduser/wc.jar WordCount input_dir output
hduser@ubuntu:~$ hadoop fs -ls output
Found 2 items
-rw-r--r--   1 hduser supergroup          0 2022-07-17 14:36 output/_SUCCESS
-rw-r--r--   1 hduser supergroup      11397 2022-07-17 14:36 output/part-r-00000
```

→ Downloading the output file

```
hduser@ubuntu:/$ hadoop fs -copyToLocal output /usr/
hduser@ubuntu:/$ ls usr
bin  games  include  lib  local  output  sbin  share  src
```

→ Screenshot of output file (when input is urls file)



```
https://t.co/CEOyVZ9jHX 1
https://t.co/CEphOrLQlr 1
https://t.co/CGq2nbeVjz 1
https://t.co/CkpQ32zmCx 2
https://t.co/Cl3tFpgYaj 1
https://t.co/CtS7jWdQRe 1
https://t.co/CybABGuvrs 1
https://t.co/1ET7uqUxbP 1
https://t.co/1PQAdHRp7i 1
https://t.co/1XmEj1aIjY 2
https://t.co/1f9jCMyLor 1
https://t.co/29d7cxBned 1
https://t.co/2QnMyeBovL 1
https://t.co/2SDXsearpN 1
https://t.co/2ZTmweXV7t 1
https://t.co/2sQ0DuCVyS 1
https://t.co/34T7lTkxXJ 1
https://t.co/3JU04c9Yrx 1
https://t.co/3VXOjla12N 485
https://t.co/3YRtXAQPY7 1
https://t.co/3bjusCDHq8 1
https://t.co/3br4Nsrqh7 1
https://t.co/3pZhwuqdbX 1
https://t.co/4PRyPRTTdC 5
https://t.co/4UoXFFplr2 1
https://t.co/4fT6vCdg7K 1
https://t.co/4nlKaEBzdX 1
https://t.co/4pjHdgWD0M 1
https://t.co/5CFgzQbzuy 1
https://t.co/5IbCsMM8st 1
https://t.co/5NAbkZfwJE 8
https://t.co/5QYA6SEJ6H 236
https://t.co/5V39qC3N8G 1
https://t.co/5ZTExxWFbX 1
https://t.co/5dbQn5f61w 1
https://t.co/5hcjLRG0jm 1
https://t.co/5t95rCxQtE 2
https://t.co/632LrrcOSx 1
https://t.co/68deysUejF 1
https://t.co/6Eyd25RquI 7
https://t.co/6SU5bQdmUl 1
https://t.co/6kmb5x50wo 1
https://t.co/6owUnc5j8M 1
https://t.co/6yKnaYDX2A 1
https://t.co/786c5VnrwU 1
https://t.co/7WBhnTwiBR 1
https://t.co/7b608Q19M1 1
https://t.co/7el7H6ujMy 1
https://t.co/7pItCvMYcW 1
https://t.co/7sDnt7eGzg 1
https://t.co/8BhhhzugHm 1
https://t.co/8CTd4KdGSZ 1
https://t.co/8c7xZGwU4G 1
https://t.co/8cjF9S9Qol 1
https://t.co/8k9wanVFvA 1
https://t.co/8xFIToTOxm 2
https://t.co/92FECTNA53 1
https://t.co/96G29gCfld 1
https://t.co/9CQKnxDmgP 1
https://t.co/9CrOVFy6YQ 1
https://t.co/9lwjDbHSsK 1
https://t.co/EJUcK1TJJr 1
https://t.co/ExowPVoWfN 1
https://t.co/F5dtHTxT36 1
https://t.co/FC6RaqubYR 1
https://t.co/FGBhx0wGJI 1
https://t.co/FIMyXH7ktf 1
https://t.co/FQVmRfscBR 1
https://t.co/FSAfENRclh 1
https://t.co/FcxP91Otx9 1
https://t.co/FkAkSsTYBQ 1
https://t.co/FvZDTLdWed 1
https://t.co/G09Xhndaey 1
https://t.co/G3gYBNEBuw 3
https://t.co/G6ybKQR4zb 1
https://t.co/G73I83tXLd 1
https://t.co/GNWcxIvoq8 1
https://t.co/GTRqY80Bjd 1
https://t.co/GVJk7Pypbc 1
https://t.co/GVTRJCYQ8g 1
https://t.co/GkJpus93pl 1
https://t.co/HAD9j9iDPn 1
https://t.co/HB0uvyHKAu 1
https://t.co/I2fY4u8QJ1 1
https://t.co/I5nq6jIZEl 1
https://t.co/I7aaa3dw0J 1
https://t.co/INvex3oKa1 1
https://t.co/IbrAxmgqNG 1
https://t.co/IfDi4aLthl 1
https://t.co/IlhCdFXVt1 4
https://t.co/IpCSFqs4TB 1
https://t.co/IyYWstMHOn 1
https://t.co/J25aqmbiUm 1
https://t.co/JnfwKoY7fl 1
https://t.co/JvUHVB3ay2 1
https://t.co/K5iF4JUot1 1
https://t.co/KI4tjQW69e 1
https://t.co/KafOmGirMm 1
https://t.co/L4iSviGg62 1
https://t.co/L88qmxxEhO 1
https://t.co/LcvTYtyowB 1
https://t.co/LmNY0JwGP4 1
https://t.co/LnkvP28S4I 1
https://t.co/Lp5qo49CxD 1
https://t.co/Lvc9booWKU 1
https://t.co/Ly7ohWkxHK 1
https://t.co/MC6Eo3wegS 1
https://t.co/MD7WMEnuj0 13
https://t.co/MGkjYS7OuW 1
https://t.co/MROnBx2paB 1
https://t.co/MXlvZPRvRP 1
https://t.co/MYJkVxHDse 1
https://t.co/MZxotFUcFQ 1
https://t.co/MqfxFGFdzY 1
https://t.co/MtHWsRdzyC 1
https://t.co/MtKgxjNlCO 1
https://t.co/N49vtSntax 1
https://t.co/N7n7lSeVwa 1
https://t.co/NCxDivcpfA 1
https://t.co/NF35812OhP 1
https://t.co/NHvc25JRPO 1
https://t.co/NIvsfR5D2Q 1
```

→ List of log files
/usr/local/hadoop/logs/SecurityAuth-hduser.audit
/usr/local/hadoop/logs/hadoop-hduser-datanode-ubuntu.out
/usr/local/hadoop/logs/hadoop-hduser-namenode-ubuntu.out
/usr/local/hadoop/logs/hadoop-hduser-nodemanager-ubuntu.out
/usr/local/hadoop/logs/hadoop-hduser-resourcemanager-ubuntu.out
/usr/local/hadoop/logs/hadoop-hduser-secondarynamenode-ubuntu.out
/usr/local/hadoop/logs/hadoop-hduser-datanode-ubuntu.log
/usr/local/hadoop/logs/hadoop-hduser-namenode-ubuntu.log
/usr/local/hadoop/logs/hadoop-hduser-nodemanager-ubuntu.log
/usr/local/hadoop/logs/hadoop-hduser-resourcemanager-ubuntu.log
/usr/local/hadoop/logs/hadoop-hduser-secondarynamenode-ubuntu.log
/usr/local/hadoop/logs/hadoop.log

→ Refer github link https://github.com/ShriRamyaA/Big_Data_Management for input, output and log files.
→ Input files:
Phase 1 → inputs

→ Spark Files:
Phase 1 → Spark

→ Hadoop WordCount java file, output and log files:
Phase 1 → Hadoop