**Principles of Big Data Management (CS5540)**

**Phase 1 Report**

**Team :**

Vamsi Draksharam (16291789)

Sai Charan Kottapalli (16247878)

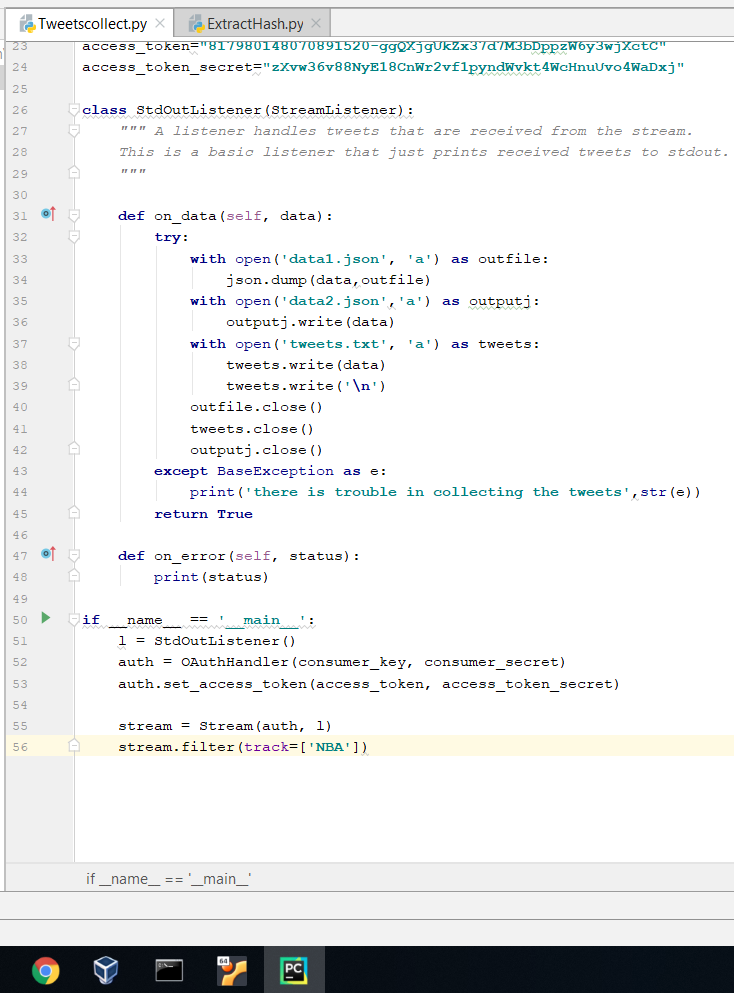
Divya Reddy Bandari (16281700)

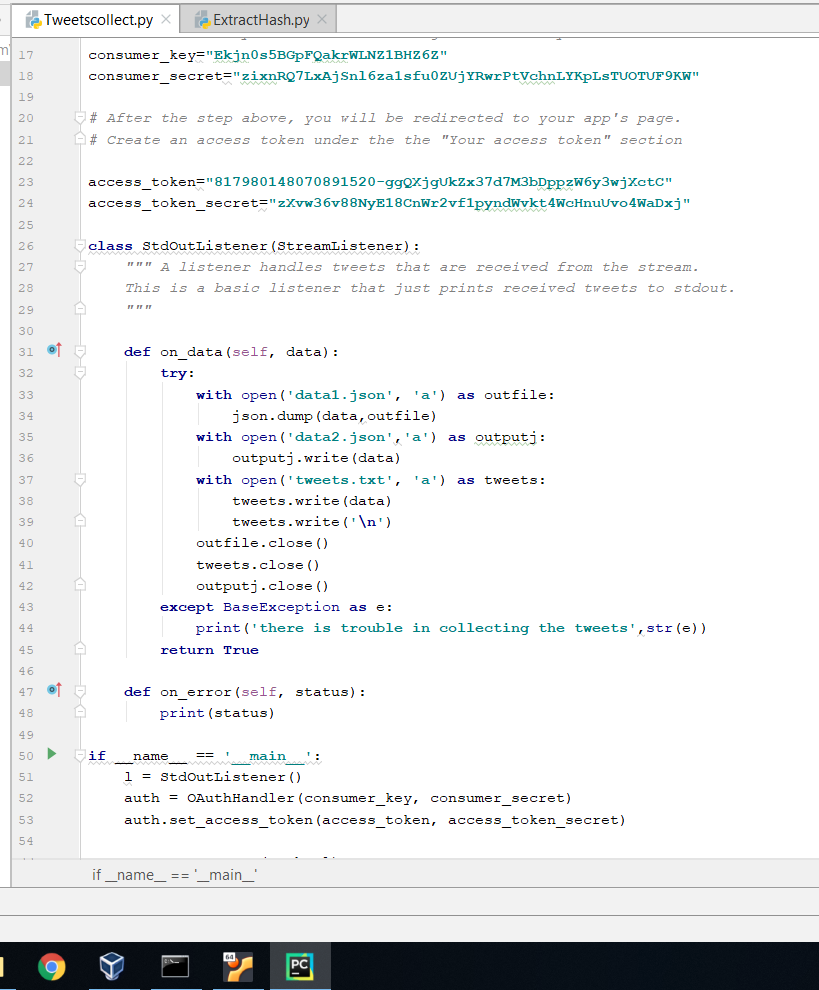
**Task 1**

Collection of the tweets using Twitter’s streaming API

Import tweepy

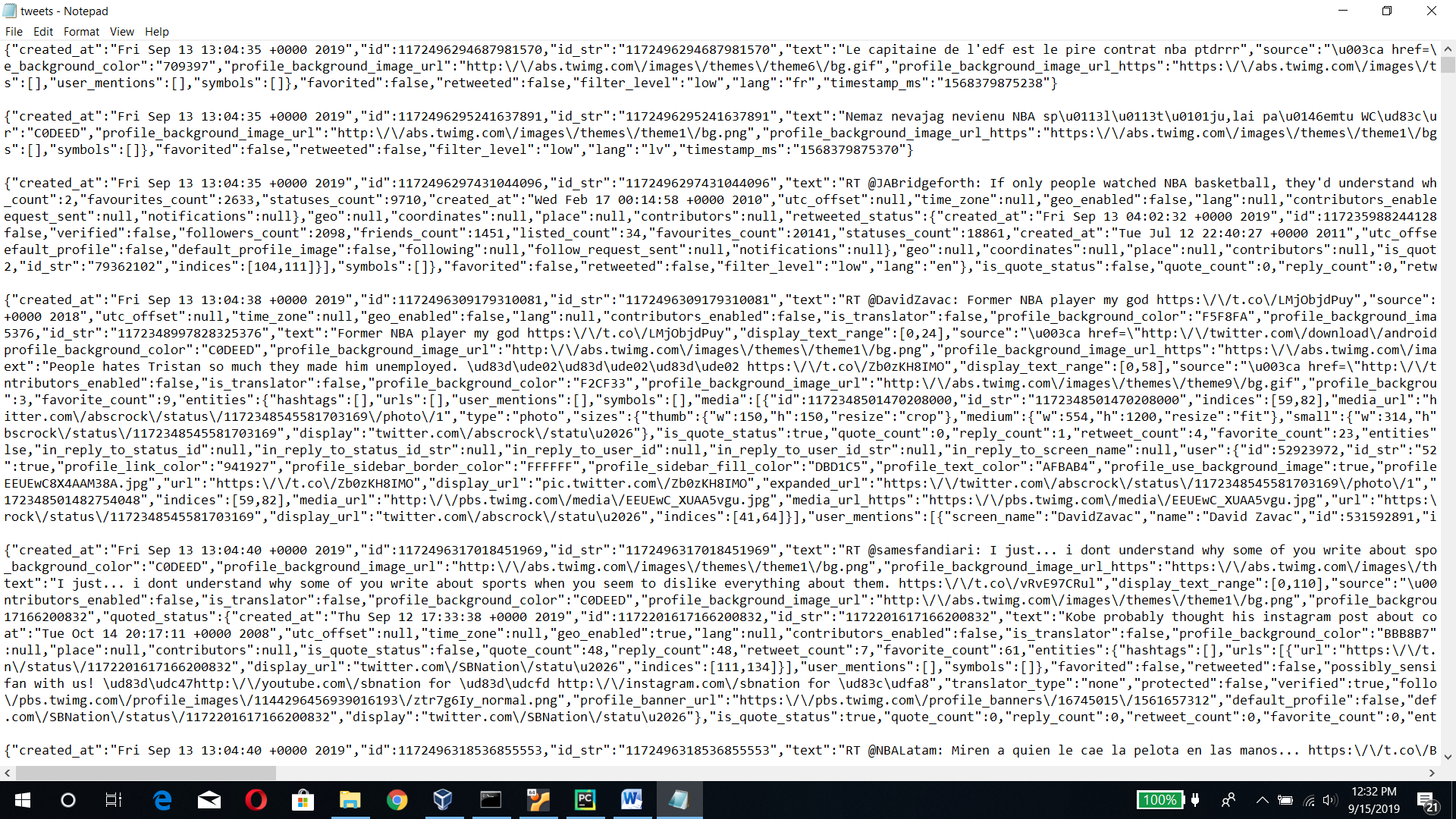
**Python code for collection of tweets :**





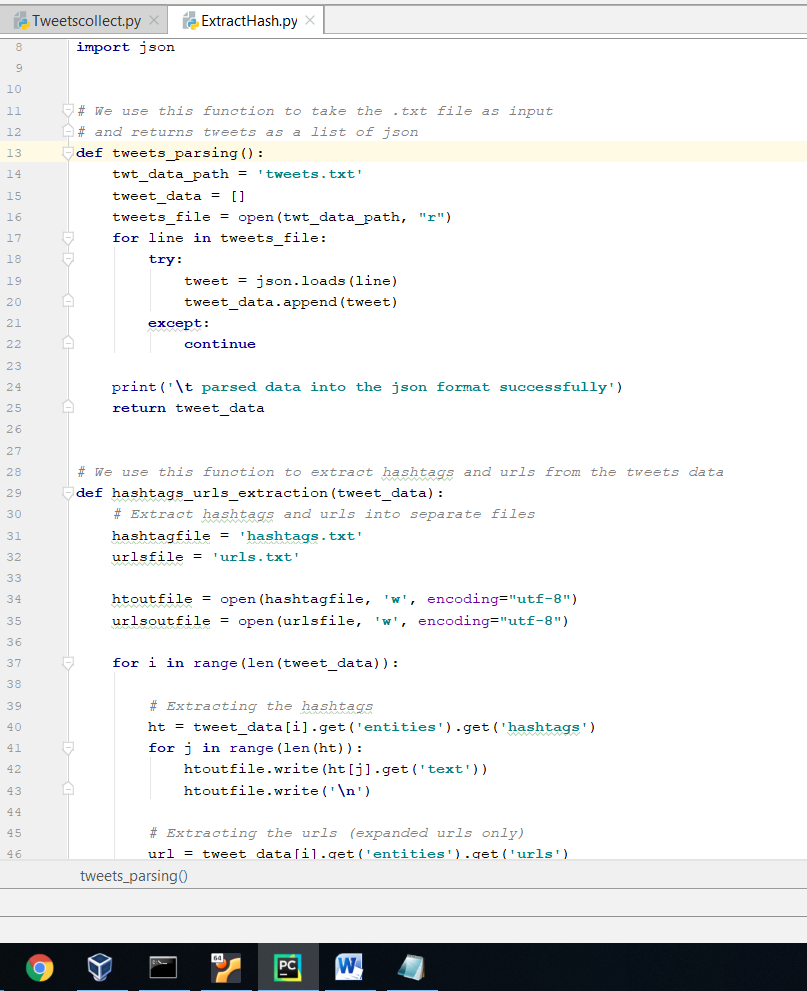
The python code specified above collects all the tweets into a txt file by using Twitter’s streaming API “tweepy”. Once the key is generated after requesting through Twitter’s developer account, we developed a code in Python to collect tweets by filtering the data with a keyword “NBA”

The output file generated is around **300,410 KB**



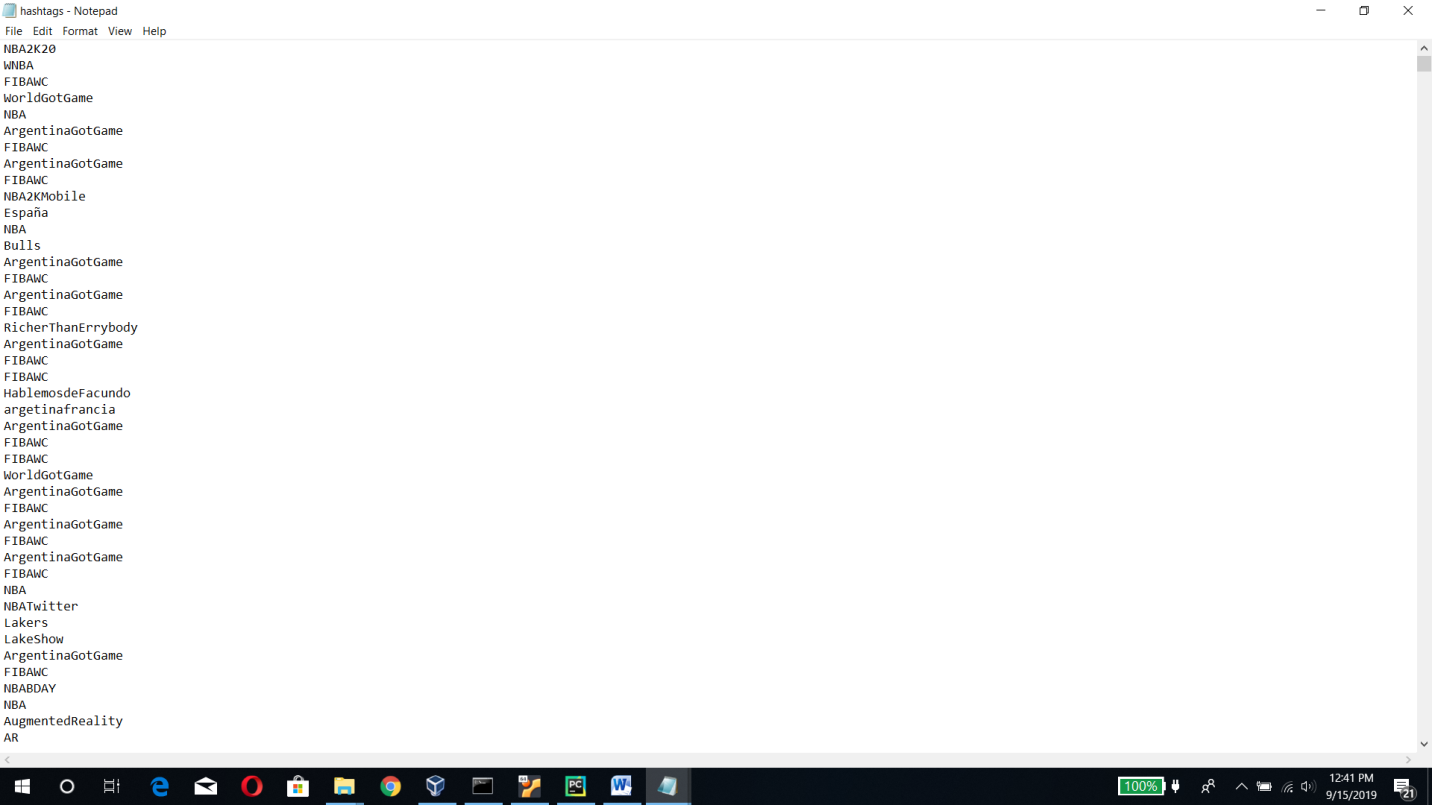
**Task 2**

**Extraction of Hashtags and URLs:**

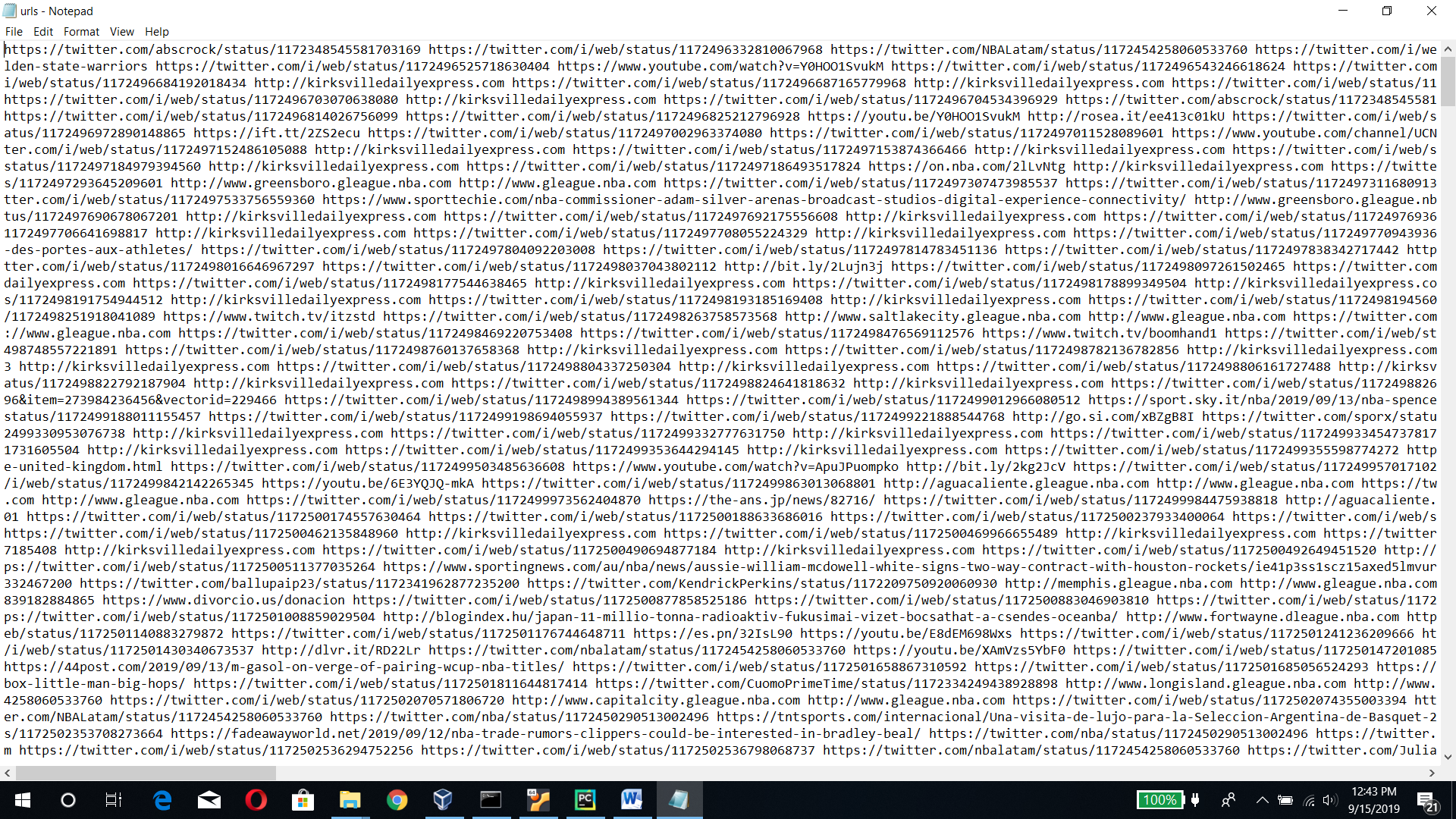
We use the above program to parse the tweets into json and extract all the hashtags and urls. Finally we write them to 2 txt files named “hashtags.txt” and “urls.txt”.

**Output :**

hashtags.txt



urls.txt:



**Task 3**

**WordCount using Apache Hadoop:**

We dealt with various commands on the Ubuntu terminal to work on the WordCount using Hadoop 2.9.2.

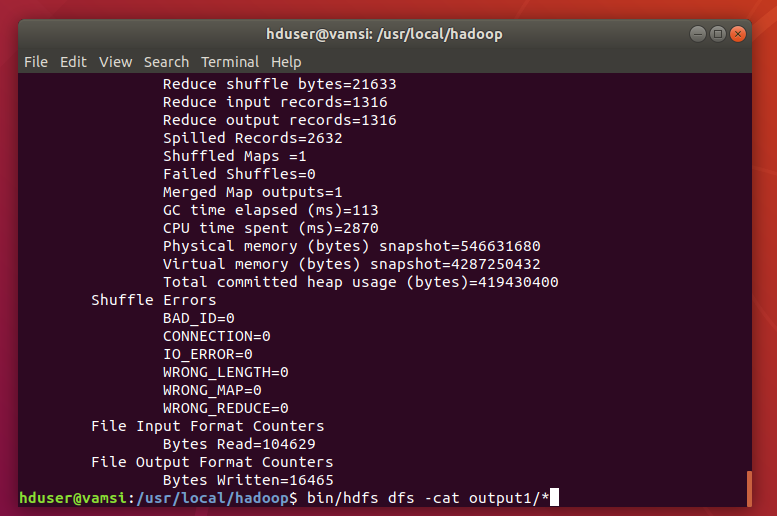
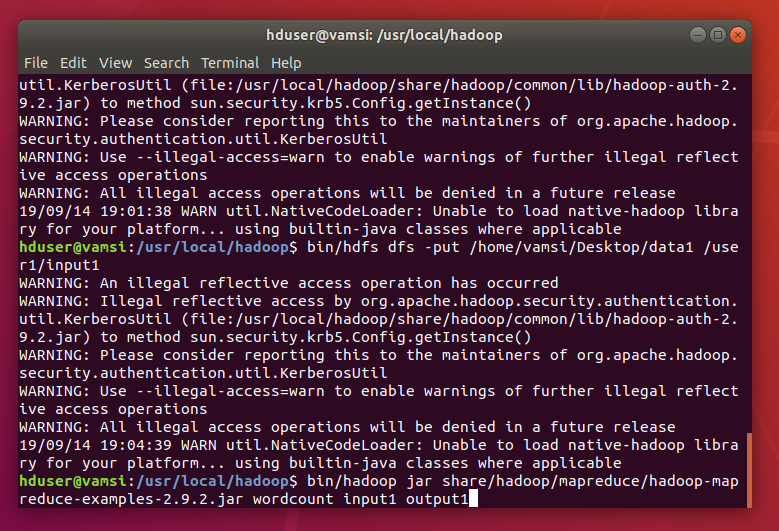
Once the Hadoop is successfully installed, start all the nodes including datanode, namenode etc. Start Yarn.

Type “jps” on the terminal and make sure that the Hadoop is running.

We have created a folder called “input1” for hashtags in the distributed filesystem and moved the text file hashtags.txt to it using command “bin/hadoop jar share/hadoop/mapreduce/hadoop-mapreduce-examples-2.9.2.jar wordcount user1/input1 output1”.

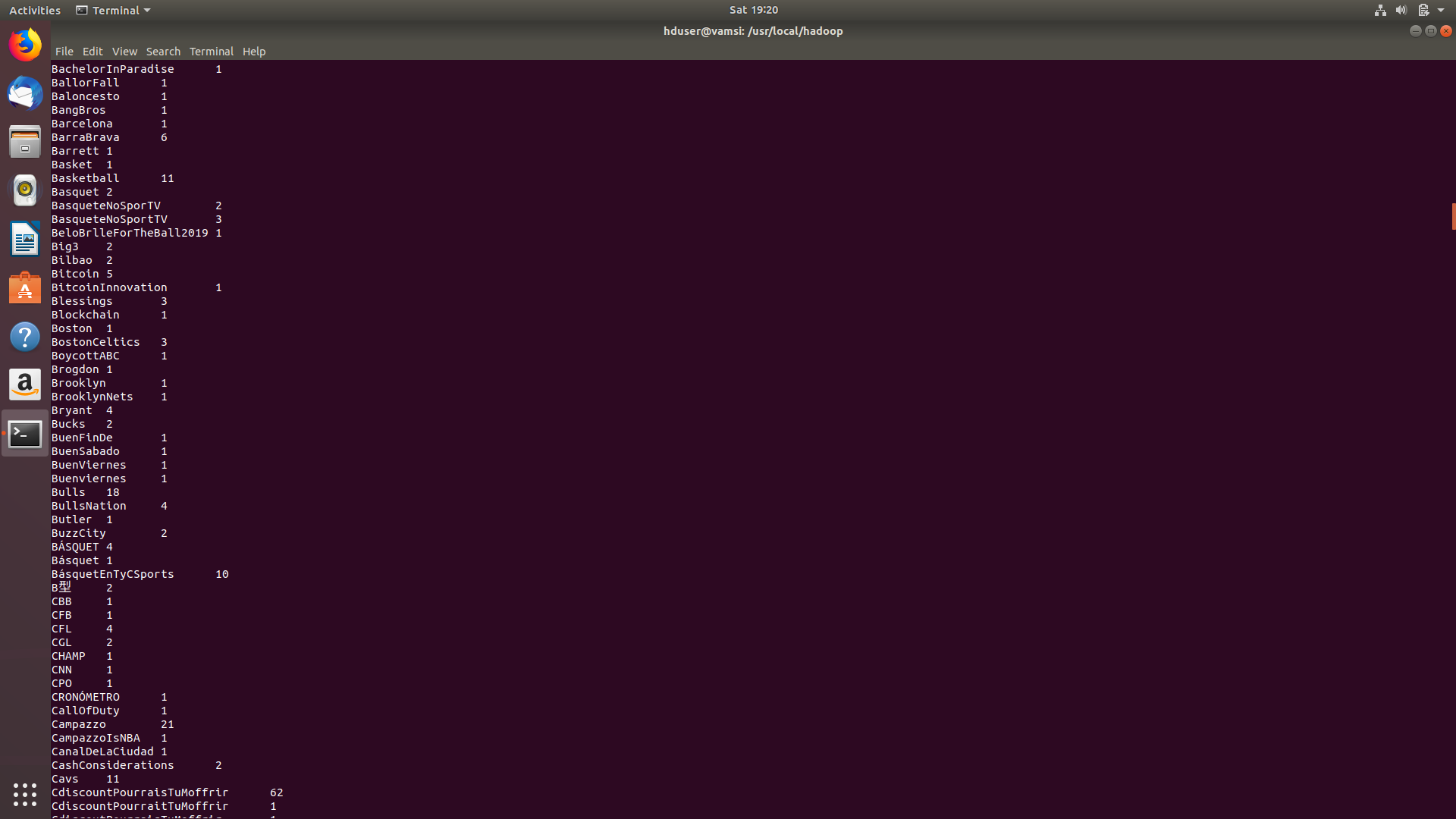
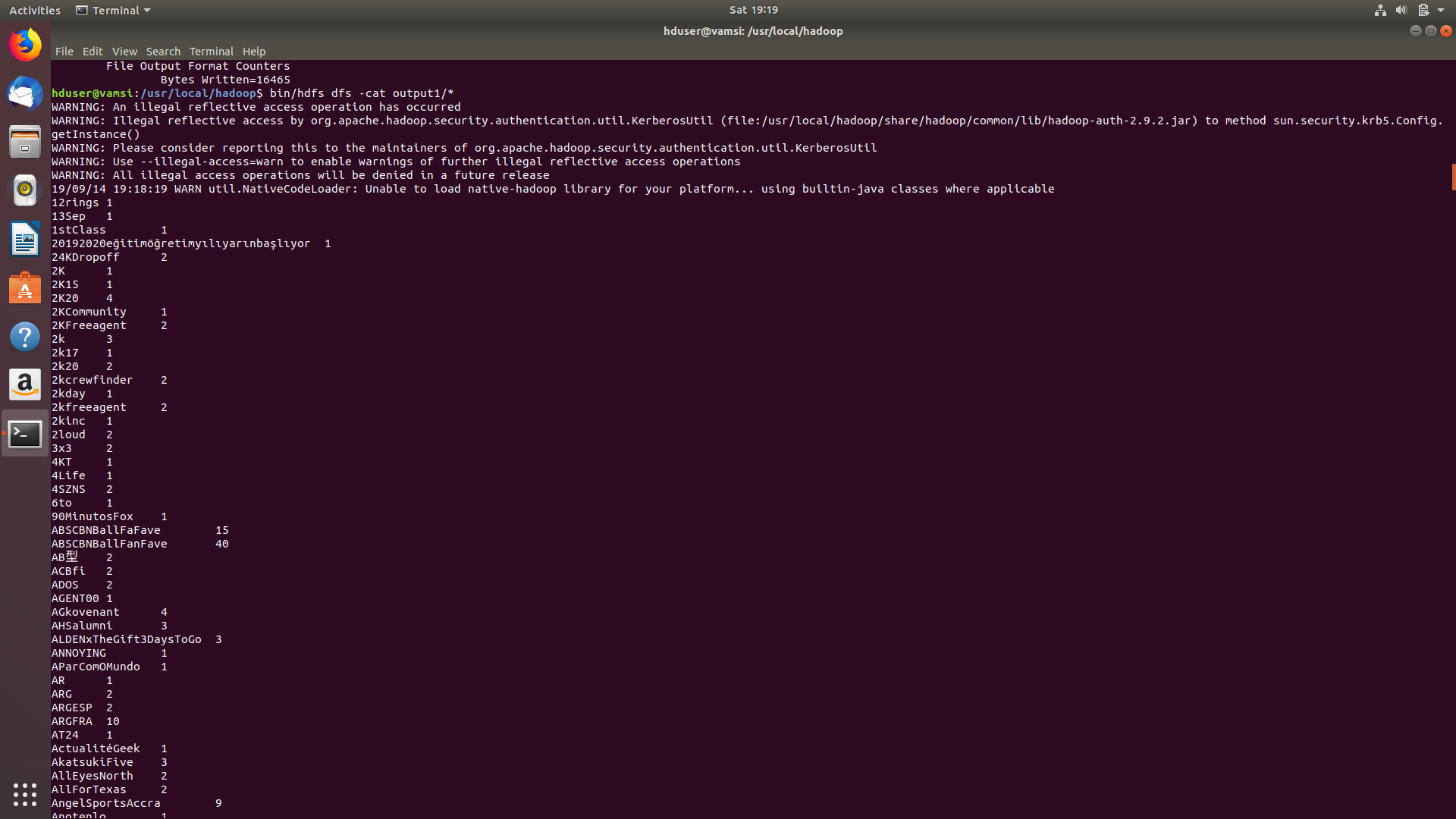
At first the mapping is done, followed by reduction. This way, the WordCount program processes the input file ie “hashtags.txt” which is present in “input1” and generates the output and log files in output1.

In the similar manner we have implemented the WordCount program on the urls file “urls.txt” by using “input2” and “output2” as input and output folders.

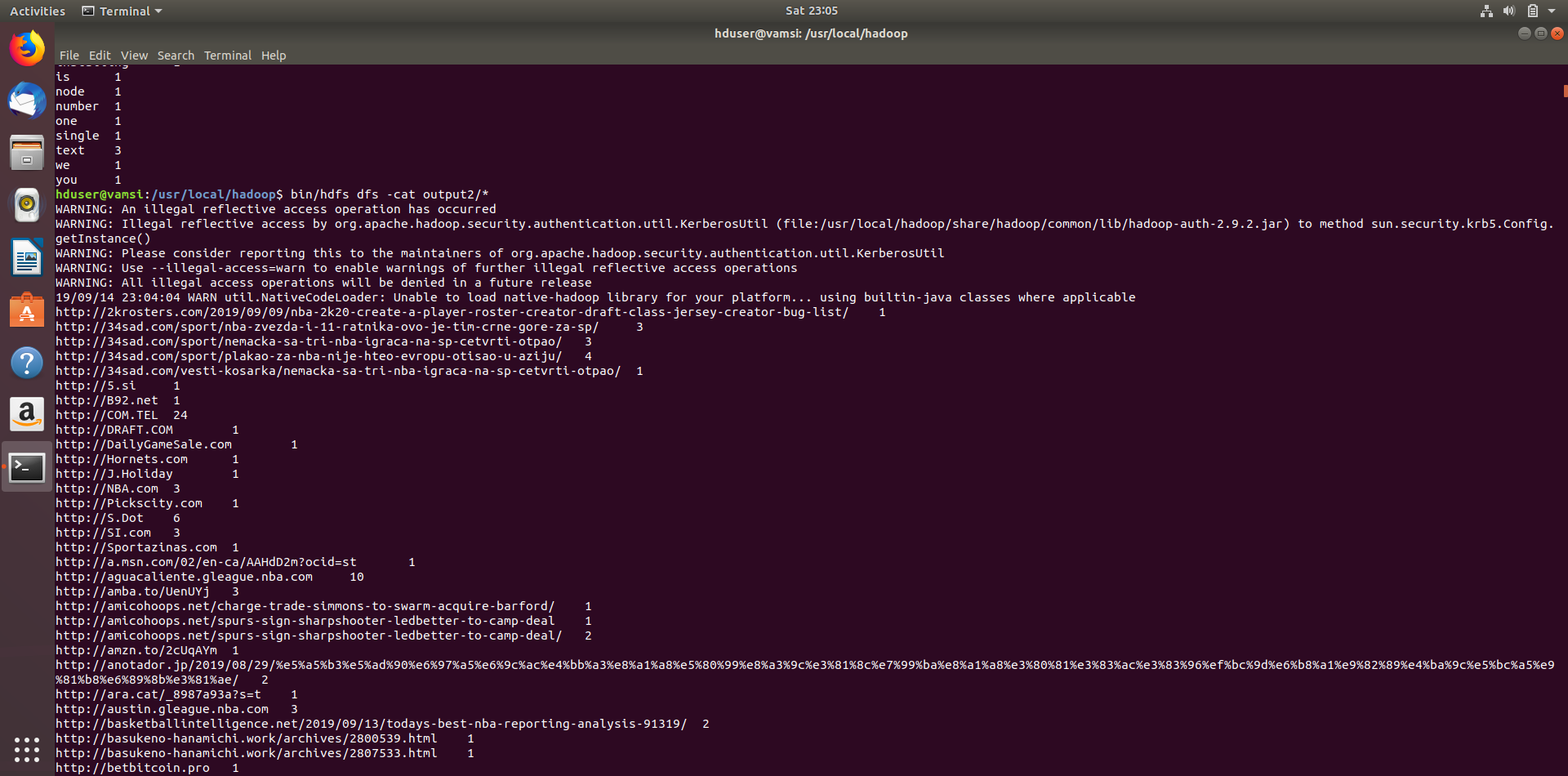


**Outputs:**

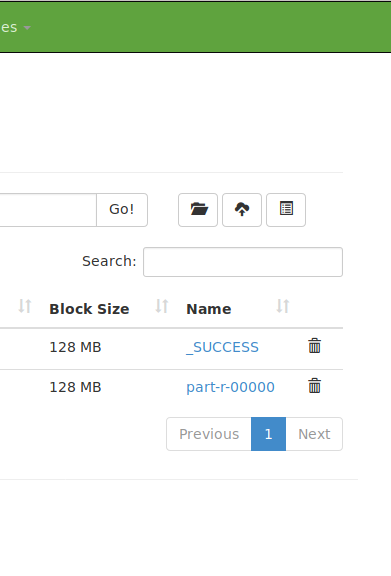
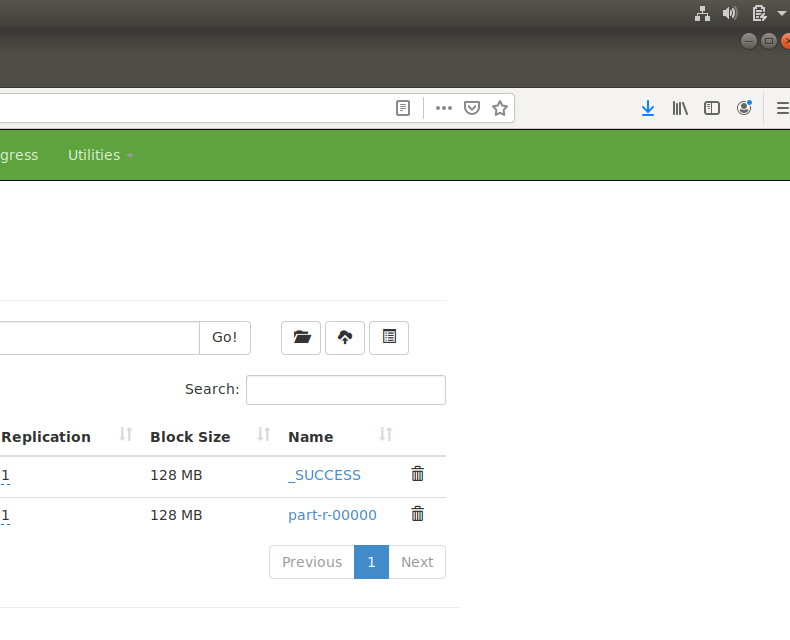
Hashtags:



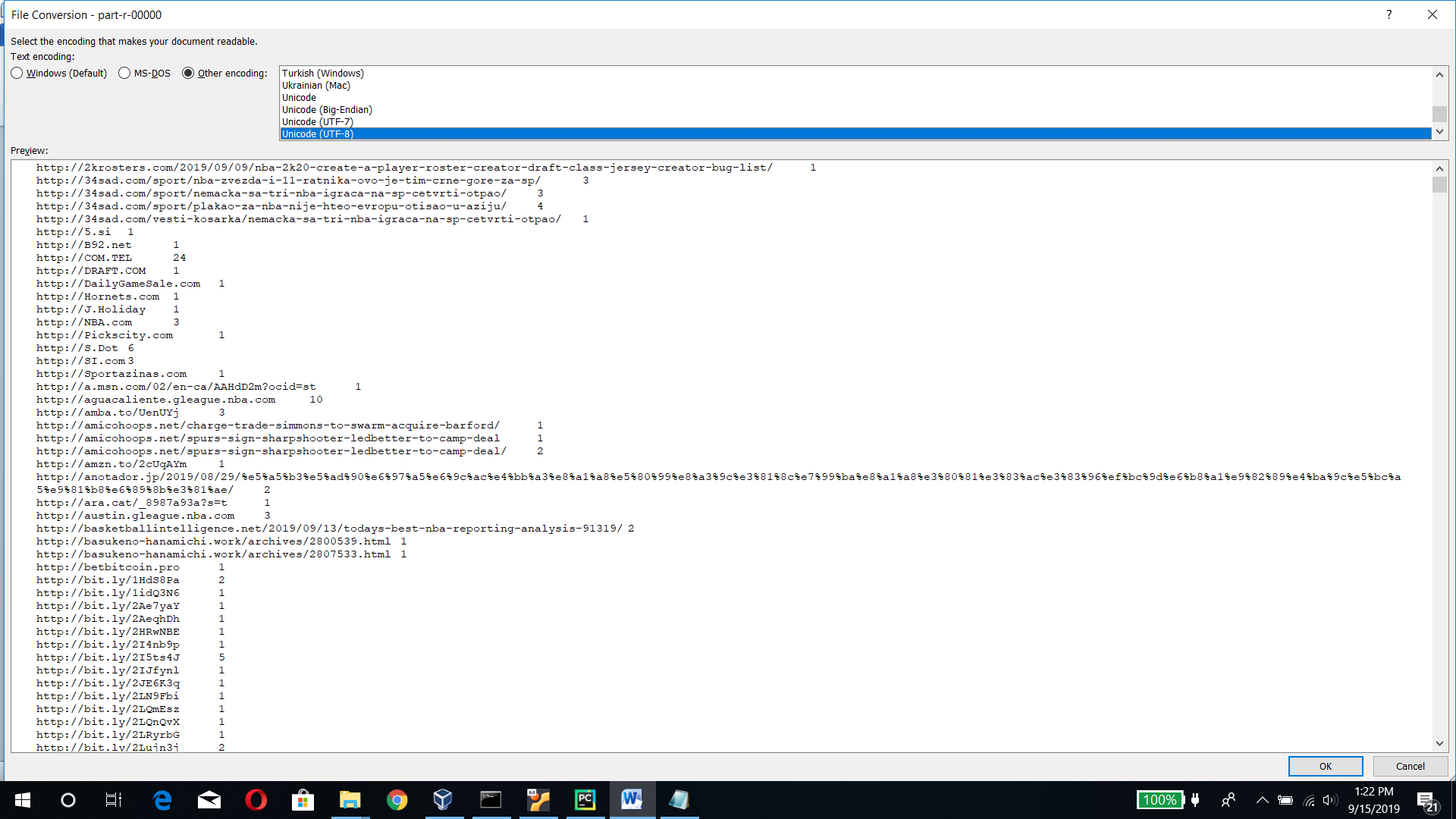
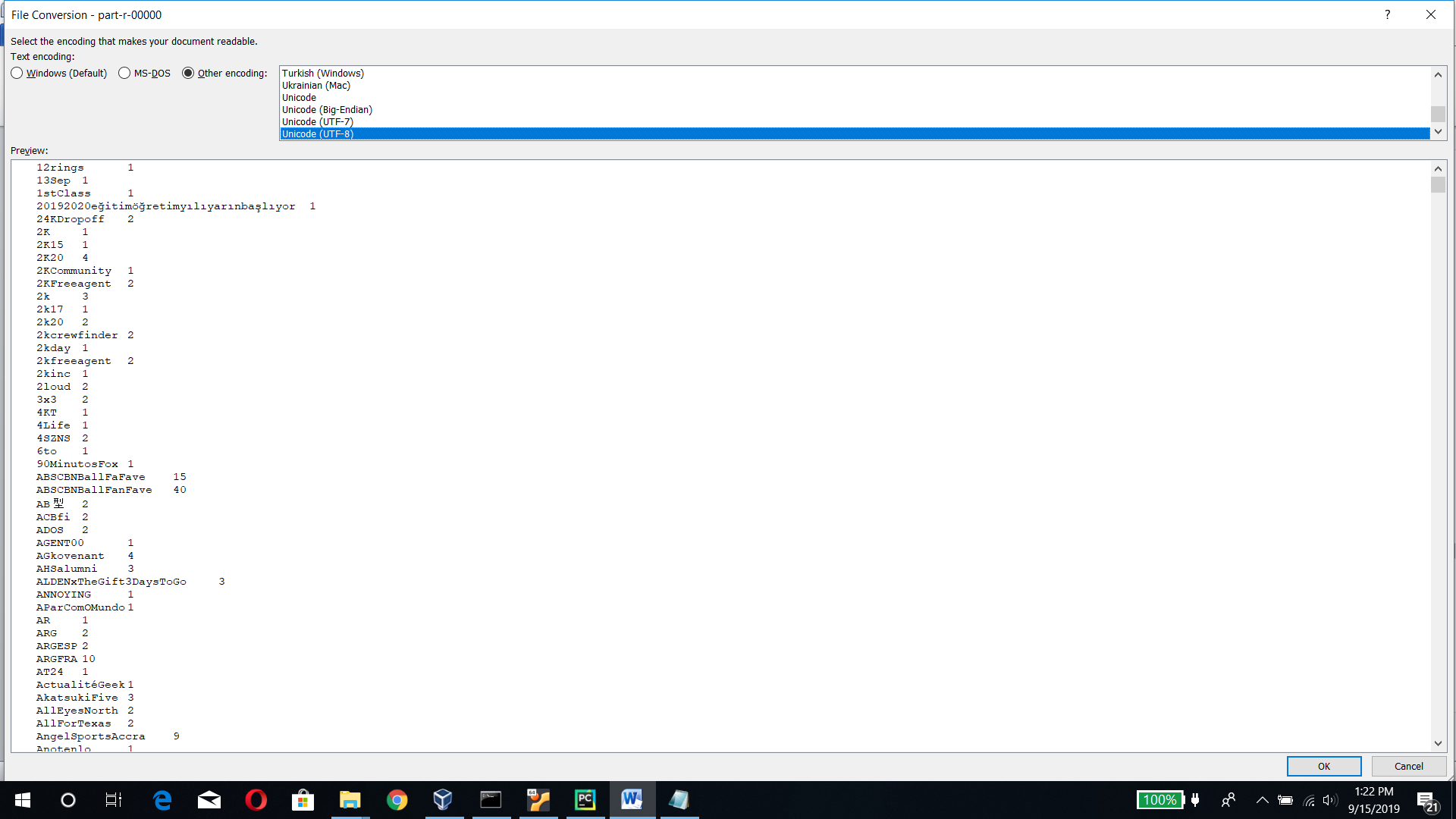
URLs:



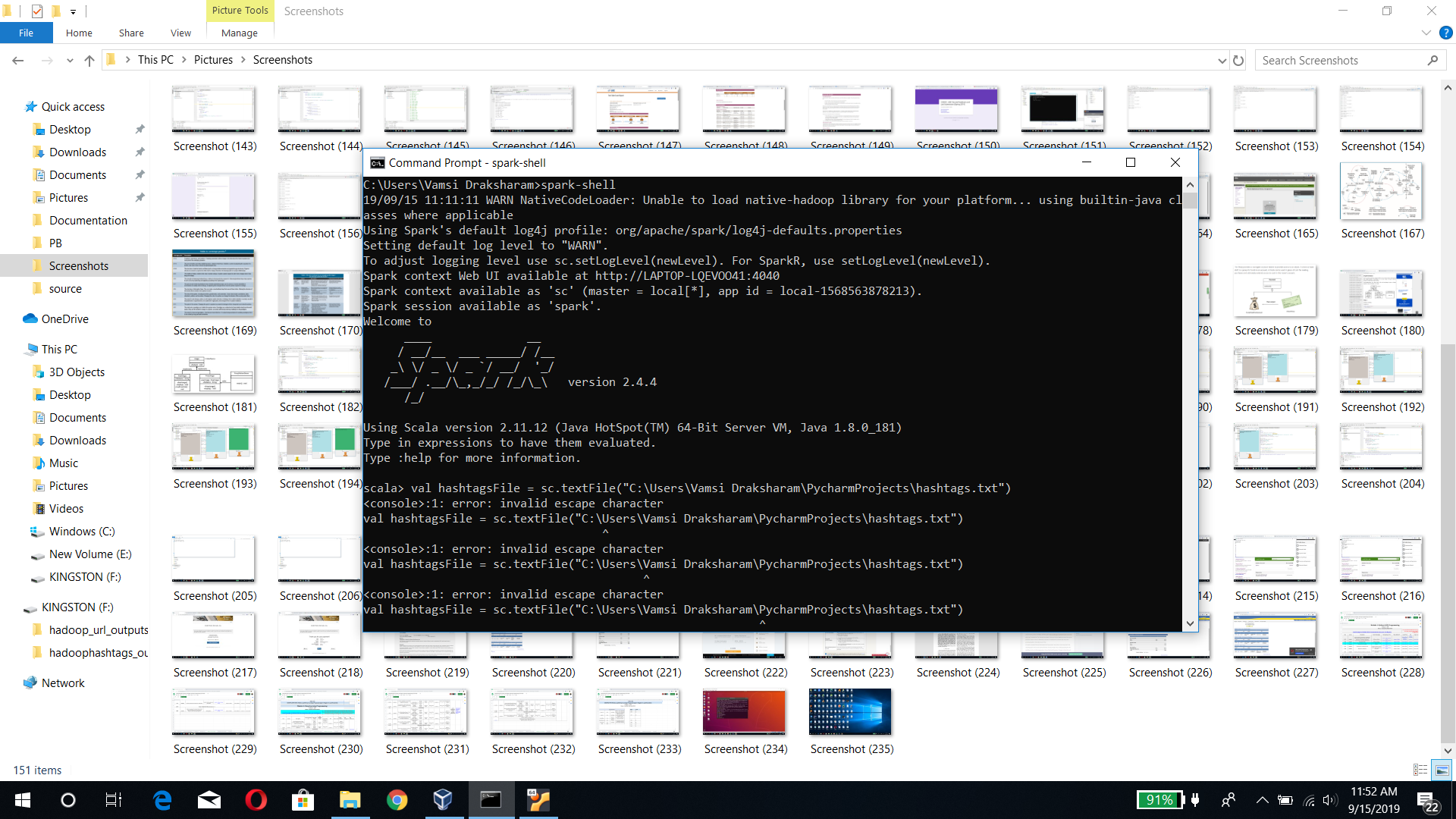
**Log files generated for hashtags and urls:**



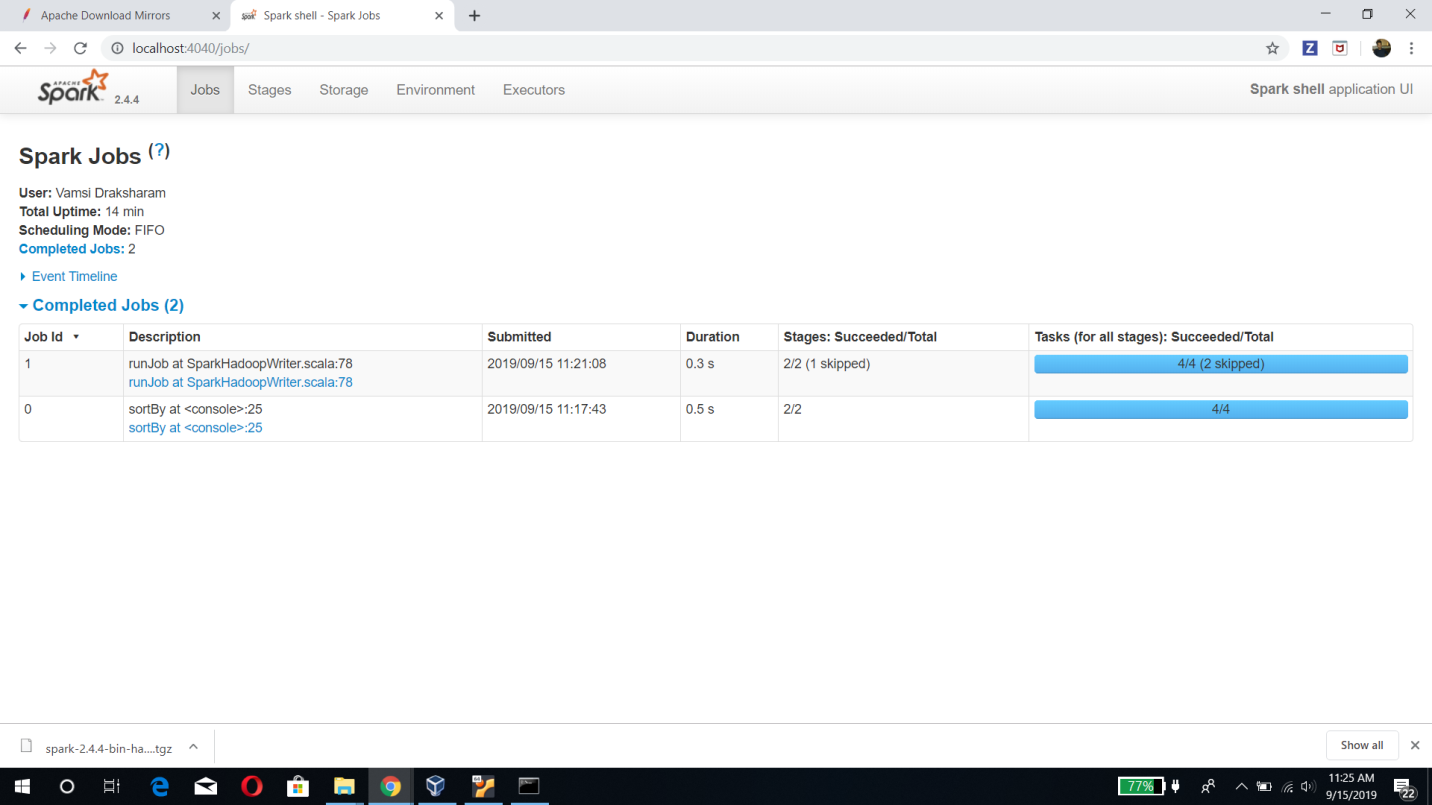
**Output files showing hastags, urls and their counts:**



**WordCount Program using Apache Spark:**



We have developed a code on scala, where the rdd is created by using text filefunction of spark. We generated words from each line by taking space as a delimiter, using flatMap function of spark.The we appended “1”as a value, with the word as a key. We add the value by key and generate words at the output, whose sum is saved to the text file. This implementation wasdone using reduceByKey function.



**Output files for hashtags and urls(Spark):**

