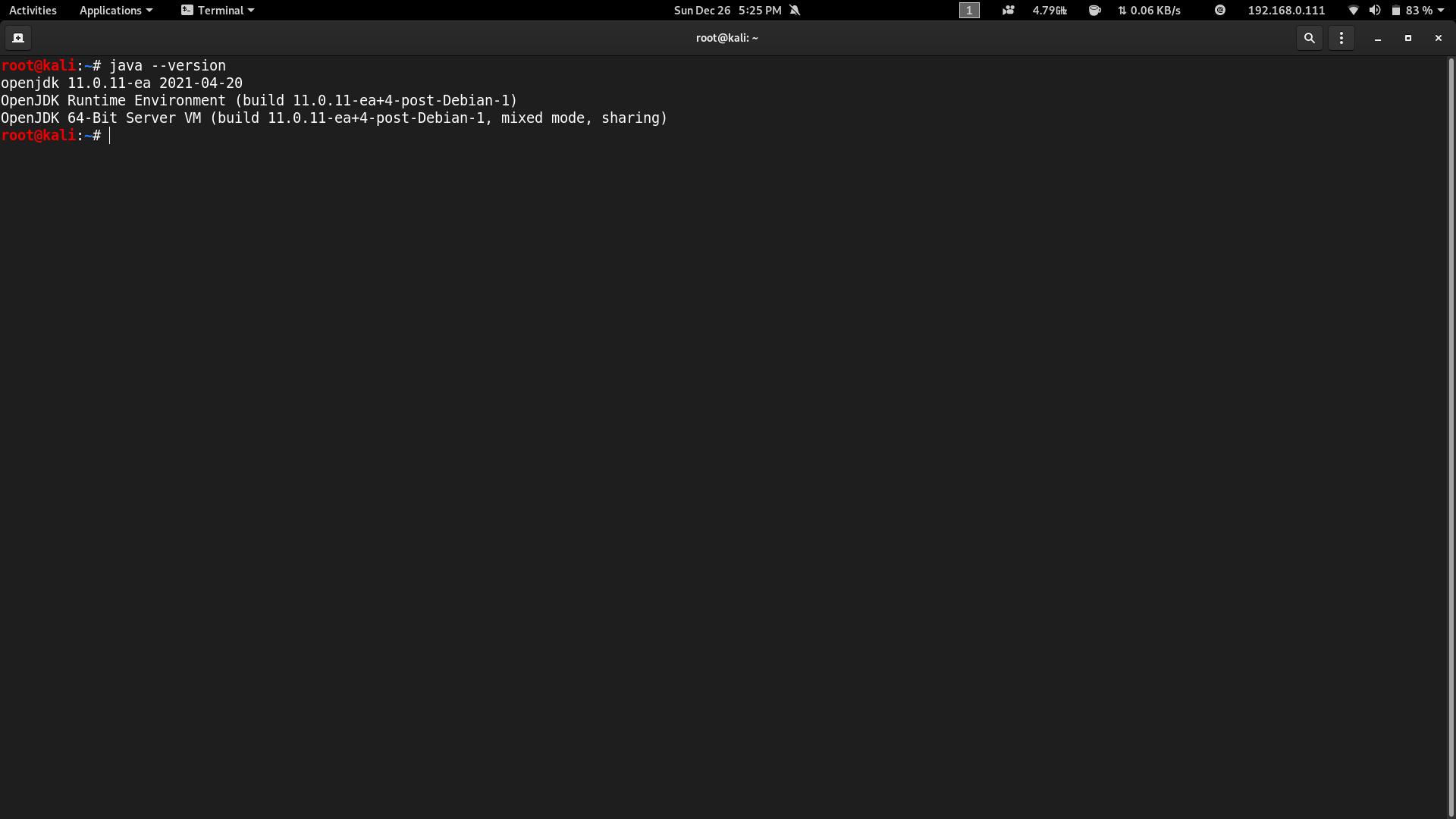
**SPARK – INSTALLATION**

The following steps show how to install Apache Spark.

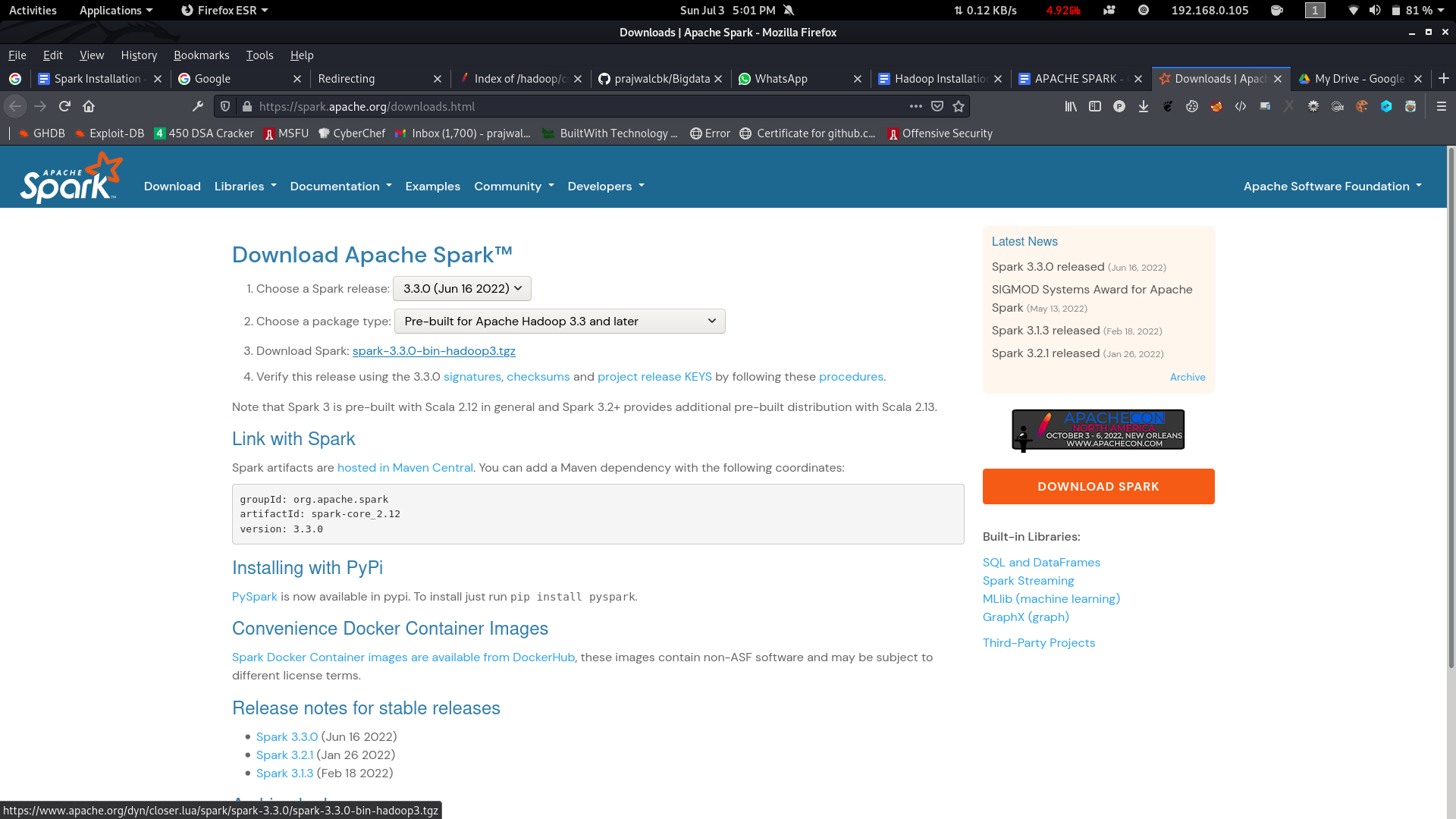
**Step 1: Verifying Java Installation**

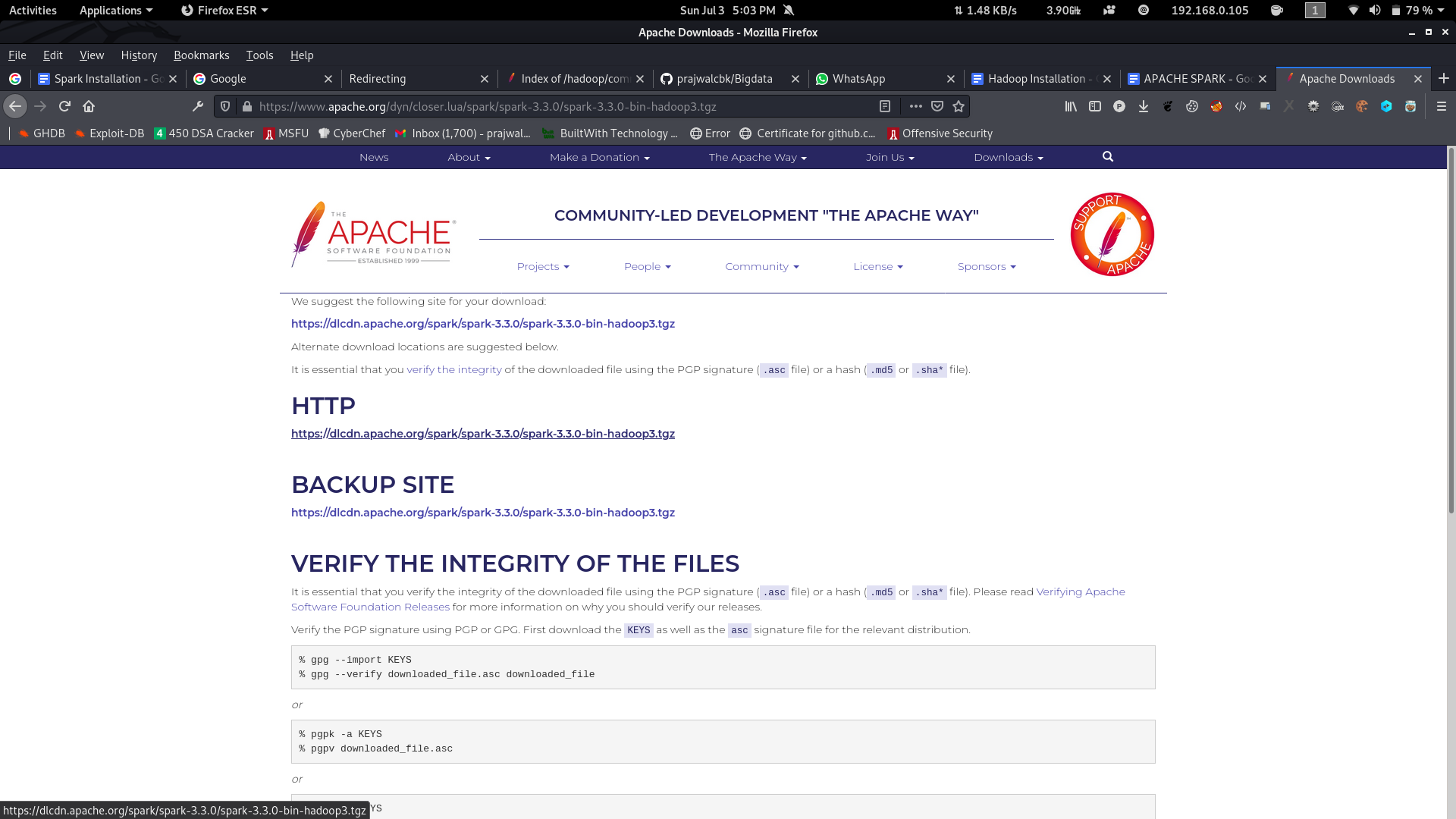
If Java is already installed on your system, you get to see the following response or some other versions.

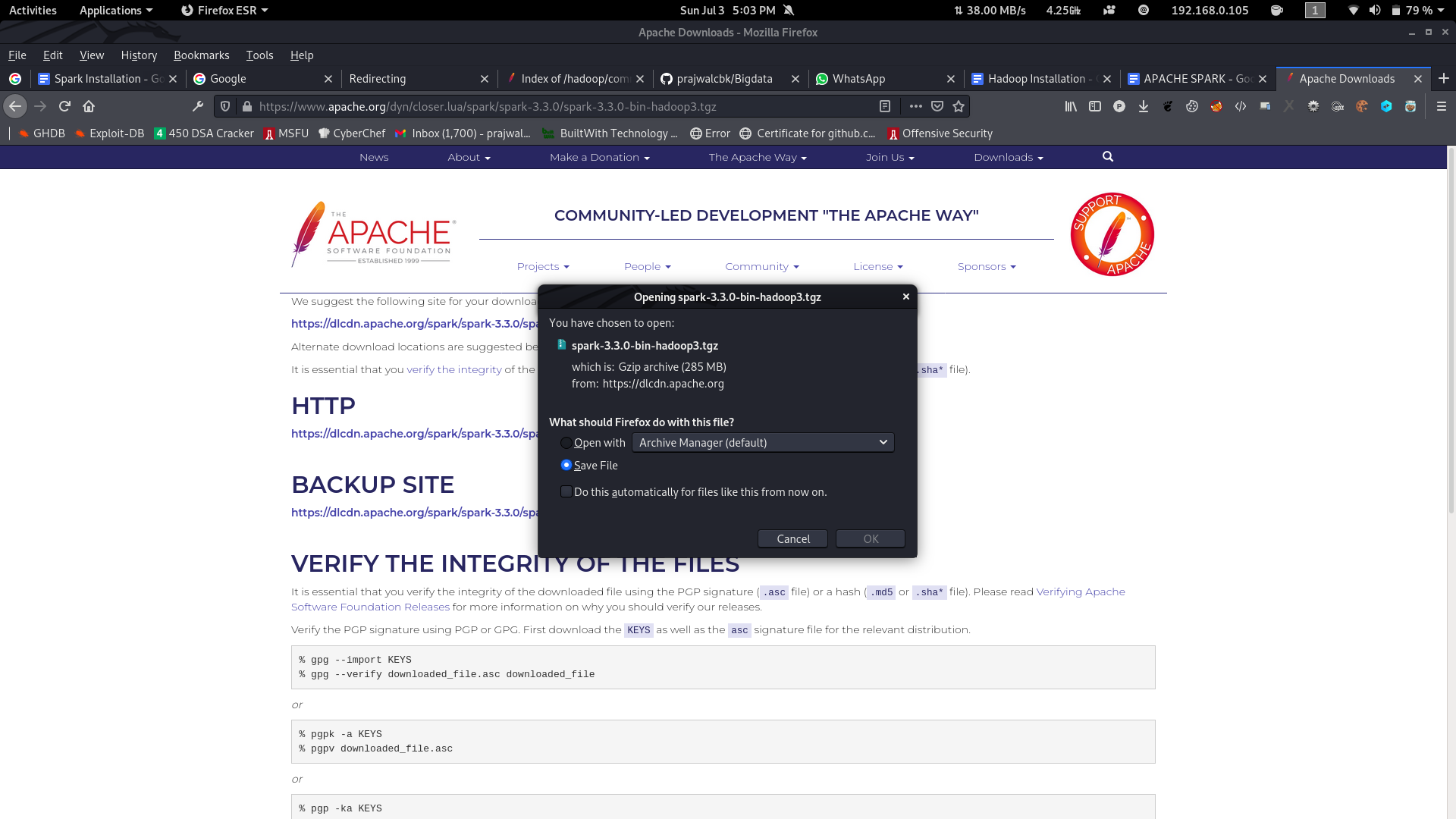
****

**Step 2: Downloading Apache Spark**

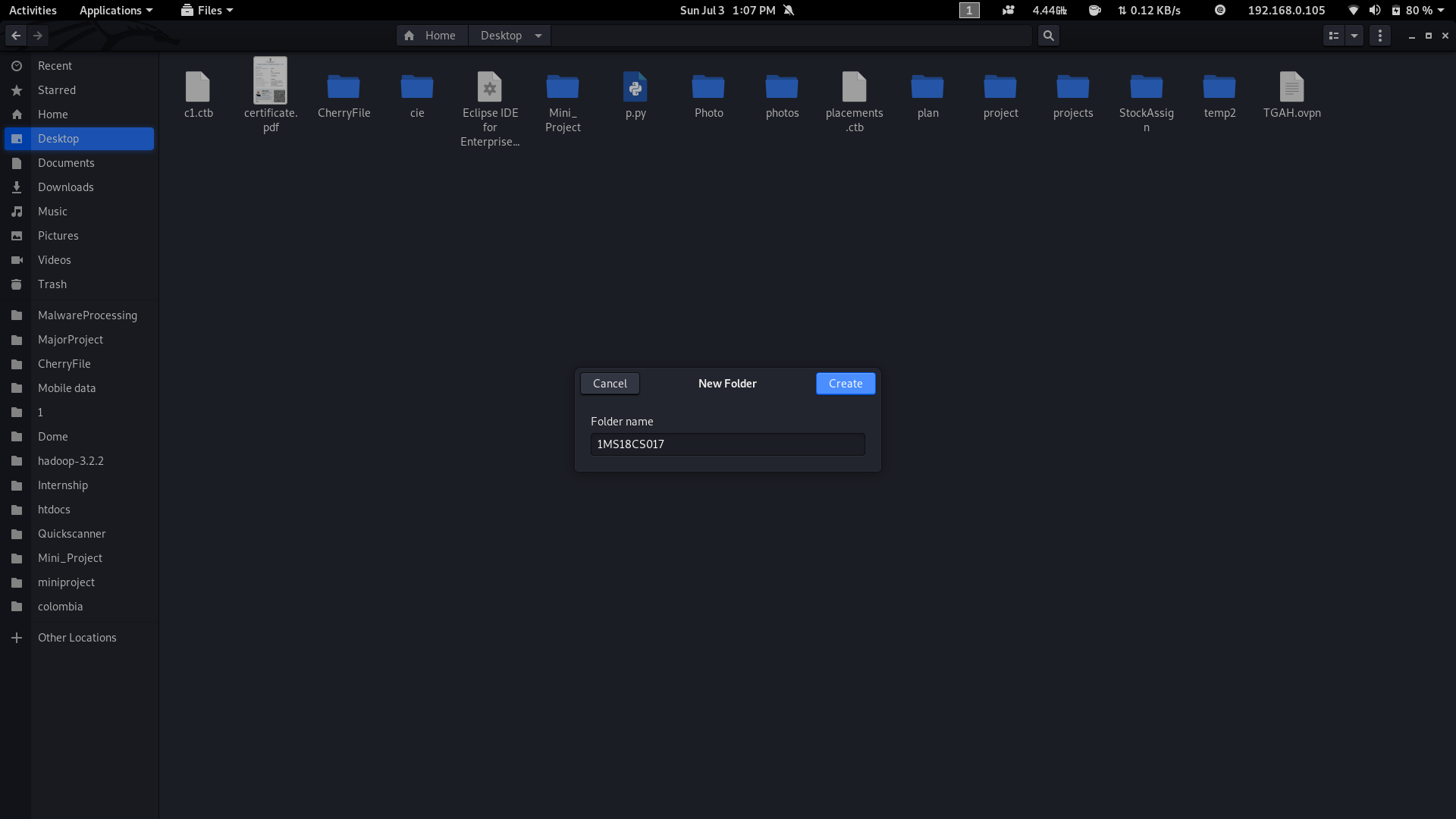
Download the latest version of Spark by visiting the following link Download Spark <https://spark.apache.org/downloads.html> . Select the latest version in Spark release and select pre-built for Apache Hadoop 3.3 and later . Click on the Download Spark link . It will navigate to one more page , and use HTTP to download the file . After downloading it, you will find the Spark tar file in the download folder.



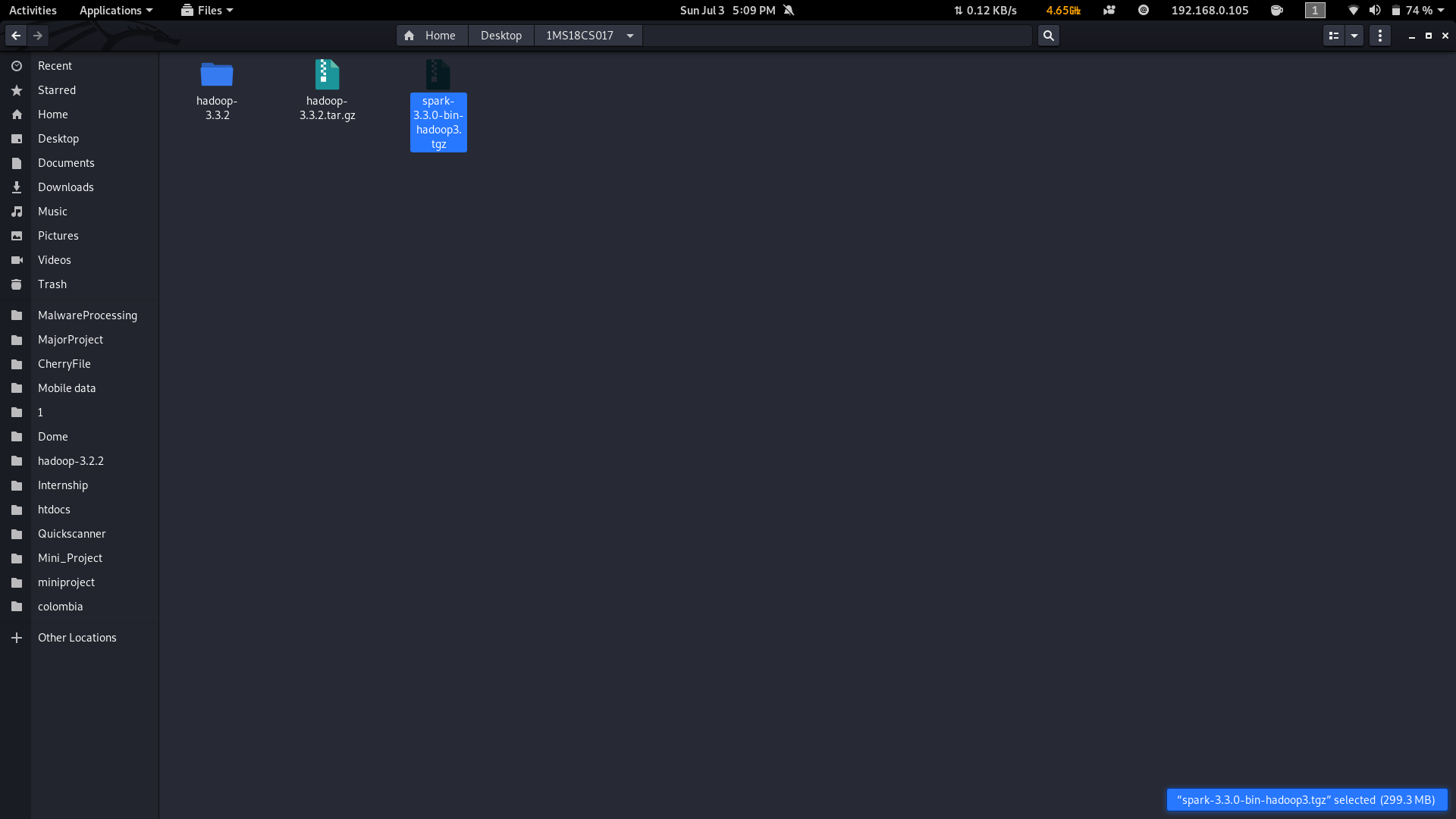




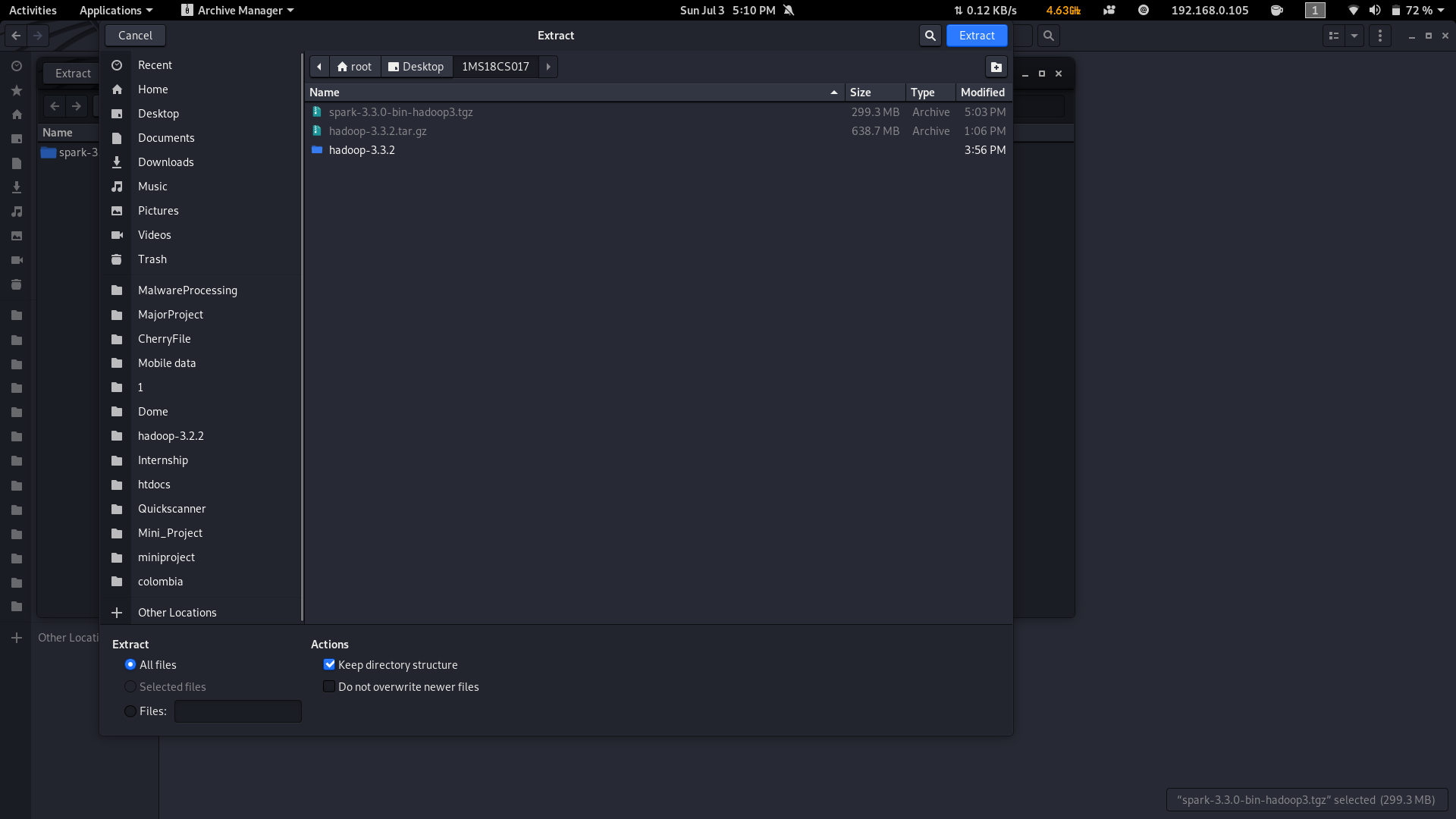
**Step 3: Create a new Folder inside Desktop , name the Folder as your USN <1ms18cs017>.**



**Step 4 . Move the Downloaded Spark File to USN <1ms18cs017> Folder.**

****

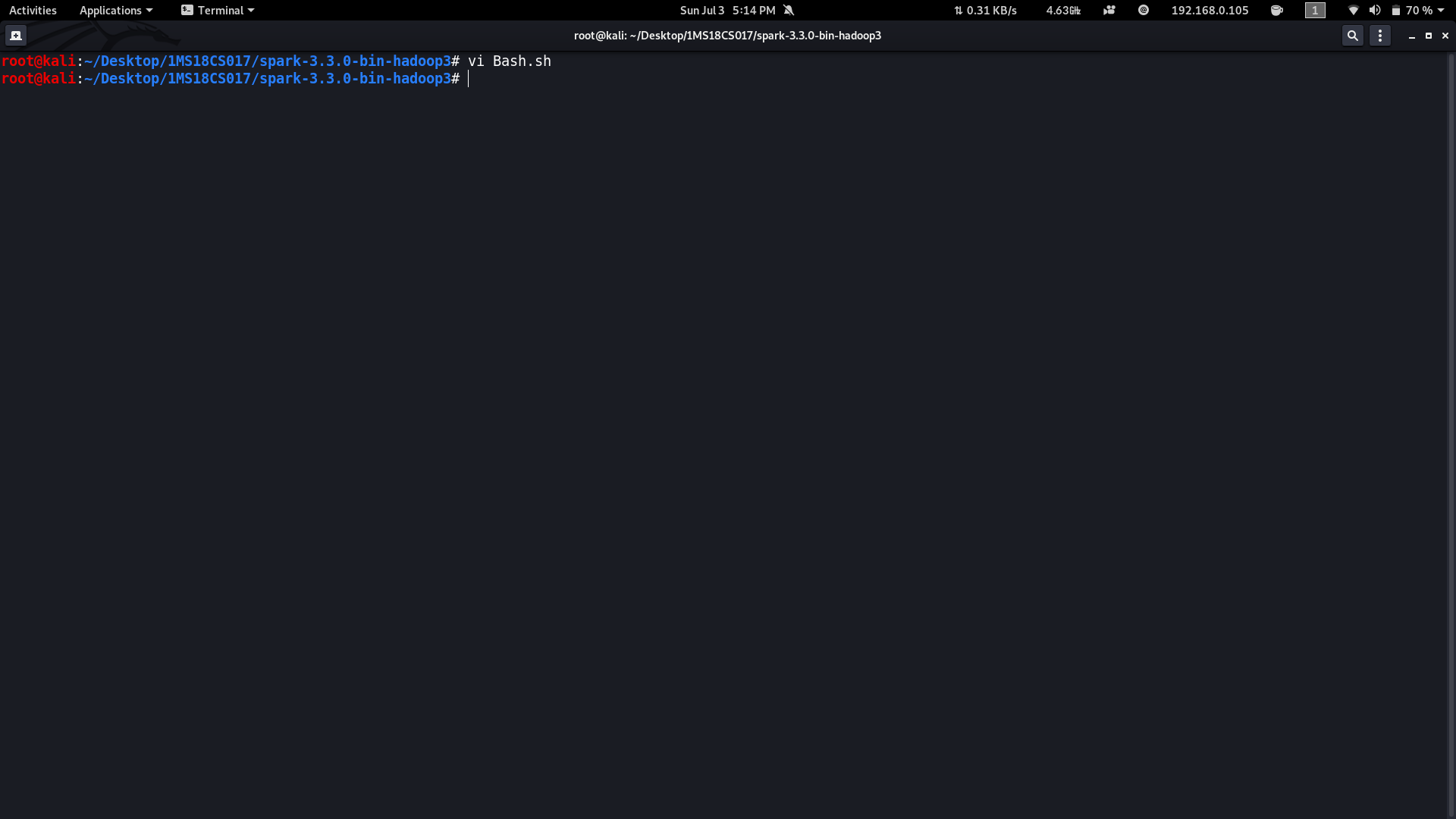
**Step 5. Right Click on that File and Extract inside the USN <1ms18cs017> Folder.**

****

**Step 6:. Open Terminal**

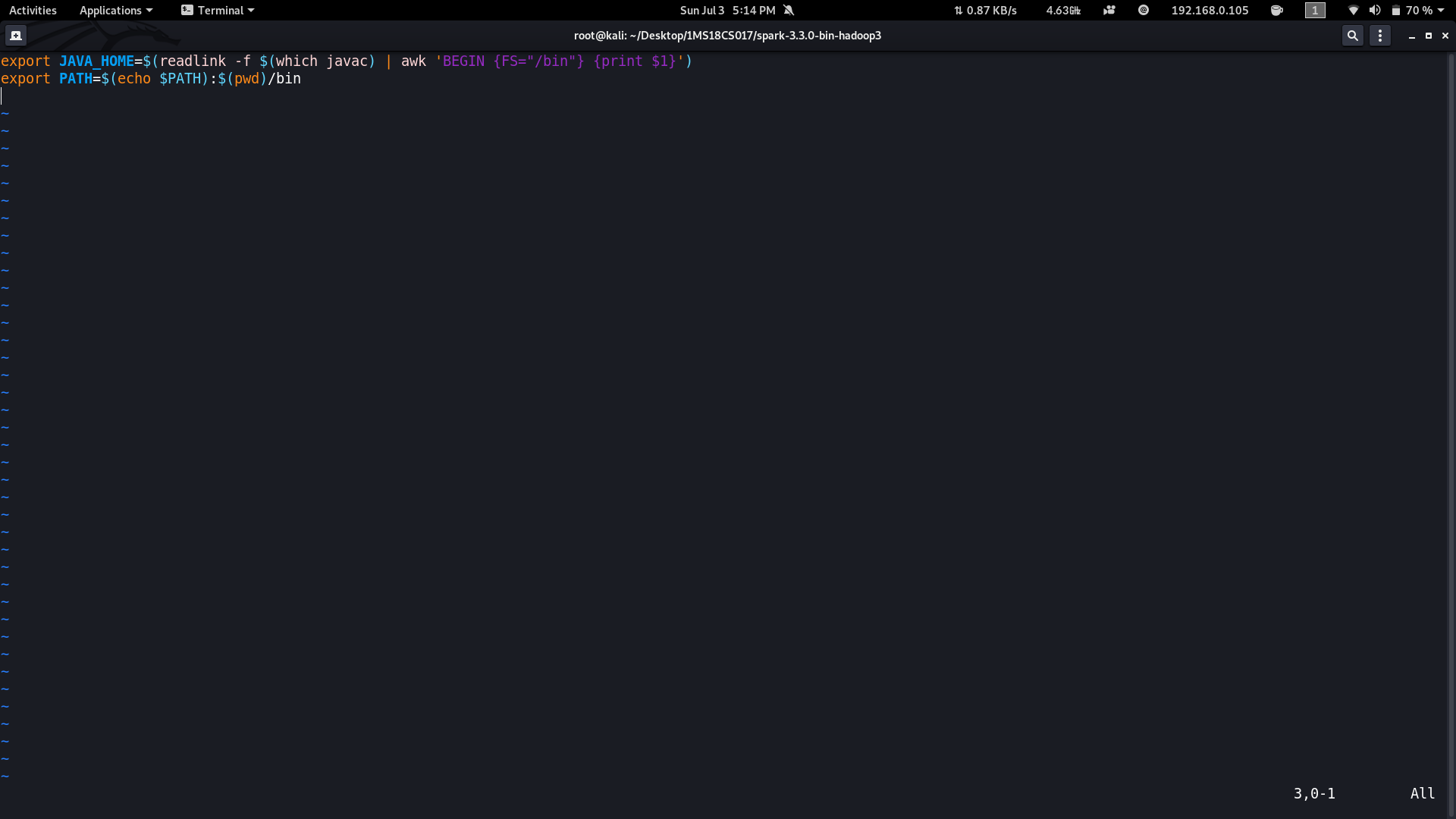
**Navigate to Extracted Hadoop Folder cd ~/Desktop/<1ms18cs017>/spark-3.3.0-bin-hadoop3**

**7. Create a New File named Bash.sh**



**8. Copy the Below code and Paste inside Bash.sh and save that File.**

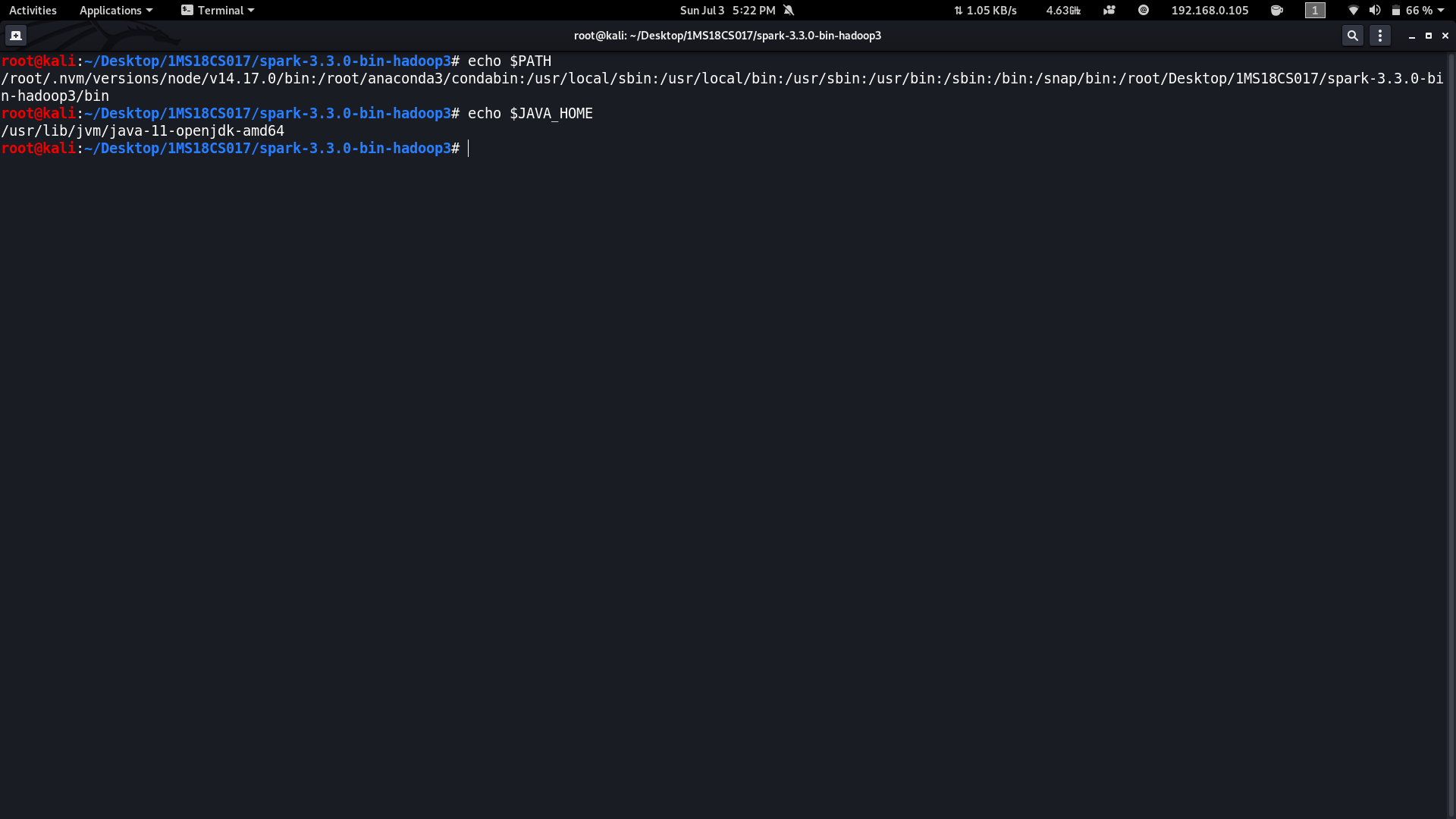
| export JAVA\_HOME=$(readlink -f $(which javac) | awk 'BEGIN {FS="/bin"} {print $1}')  if ! command -v spark-shell --version &> /dev/null  then  export PATH=$(echo $PATH):$(pwd)/bin  fi |
| --- |

****

**9. Execute the bash.sh File using following command source Bash.sh.**

**NOTE: Make source before compiling or running spark compile this file.**

**10. Verify JAVA\_HOME variable to be set to Java Path and PATH variable has your USN Spark Folder.If any previous PATH set to Spark Folder remove that inside .bashrc file.**

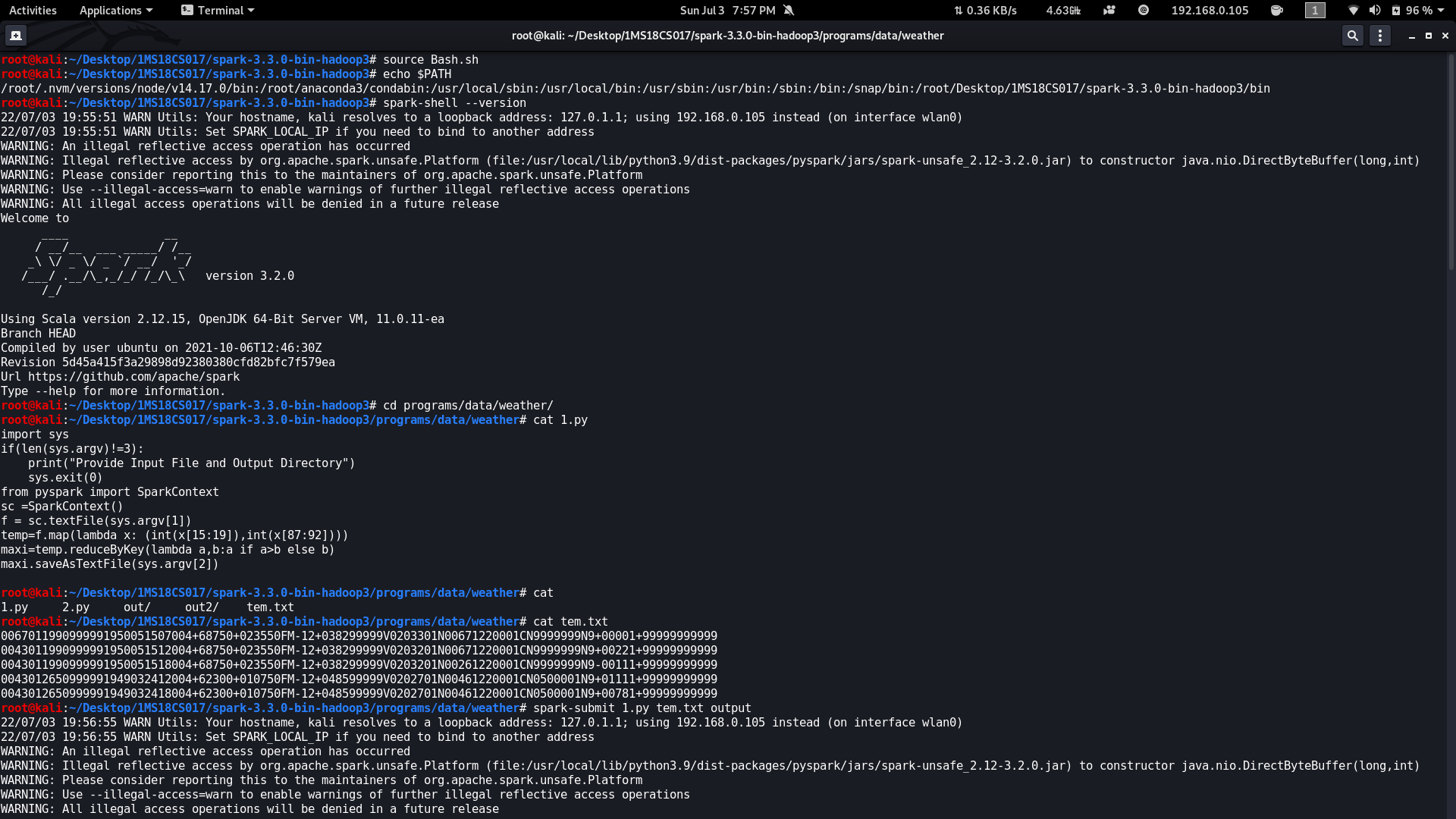
****

**11. Verify Hadoop is Installed or not by executing spark-shell --version command.if command gives Information about Hadoop command then Hadoop is Successfully Installed.**



**Execute all spark python files with spark-submit<python\_filename>.py <inputFile> <outputfolder>**

**Write a spark to analyze the given weather report data and to generate a report with cities having maximum temperature for a particular year**



import sys

if(len(sys.argv)!=3):

print("Provide Input File and Output Directory")

sys.exit(0)

from pyspark import SparkContext

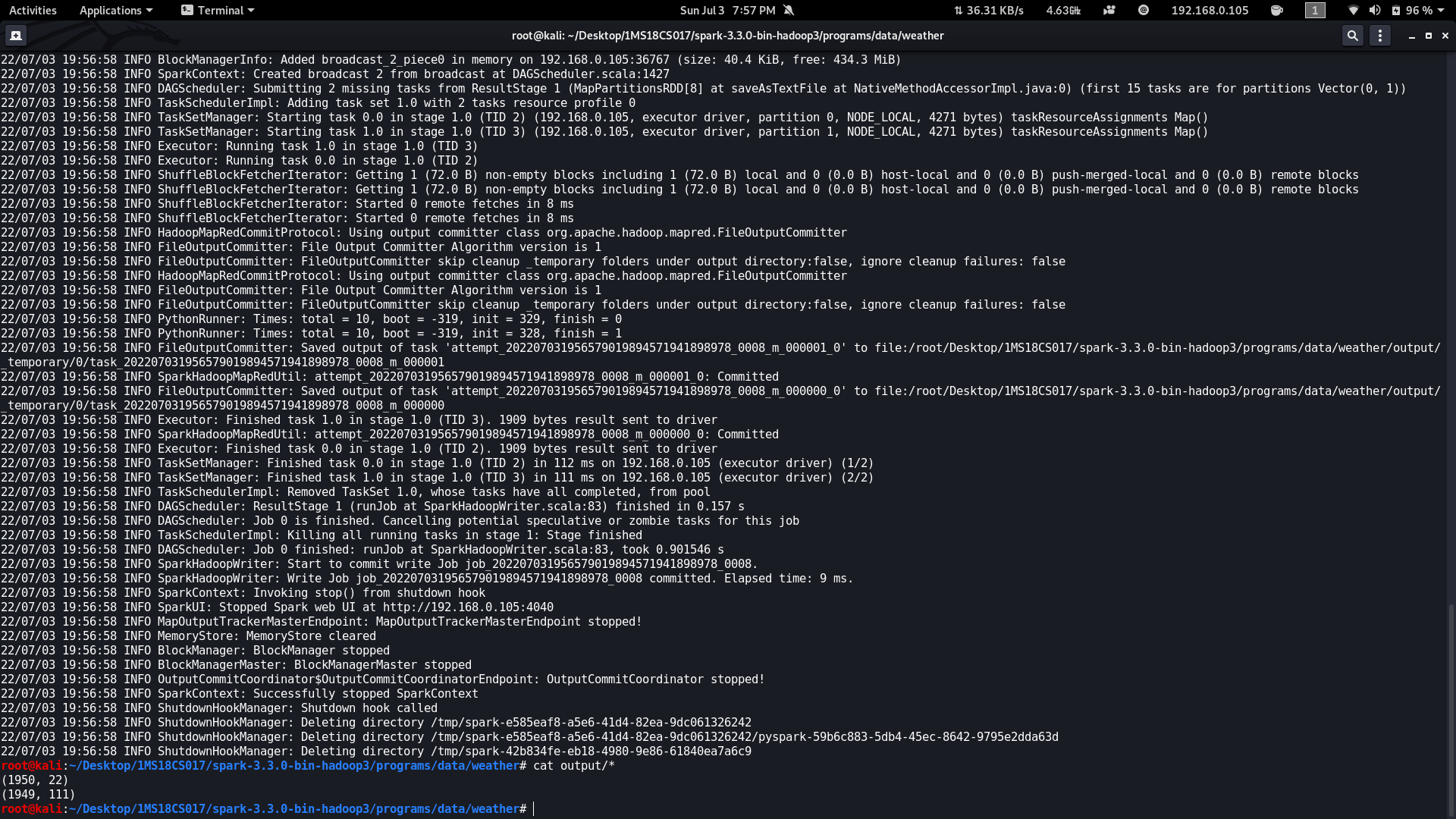
sc =SparkContext()

f = sc.textFile(sys.argv[1])

temp=f.map(lambda x: (int(x[15:19]),int(x[87:92])))

maxi=temp.reduceByKey(lambda a,b:a if a>b else b)

maxi.saveAsTextFile(sys.argv[2])



**Write a spark to analyze the given weather report data and to generate a report with cities having minimum temperature for a particular year**

import sys

if(len(sys.argv)!=3):

print("Provide Input File and Output Directory")

sys.exit(0)

from pyspark import SparkContext

sc =SparkContext()

f = sc.textFile(sys.argv[1])

temp=f.map(lambda x: (int(x[15:19]),int(x[87:92])))

mini=temp.reduceByKey(lambda a,b:a if a<b else b)

mini.saveAsTextFile(sys.argv[2])

**Write a spark program to analyze the given Earthquake data and generate statistics with region and magnitud**e

import sys

if(len(sys.argv)!=3):

print("Provide Input File and Output Directory")

sys.exit(0)

from pyspark import SparkContext

sc =SparkContext()

f = sc.textFile(sys.argv[1])

temp=f.map(lambda x: (x.split(',')[11],float(x.split(',')[8])))

maxi=temp.reduceByKey(lambda a,b:a if a>b else b)

maxi.saveAsTextFile(sys.argv[2])

**Write a spark program to analyze the given Earthquake data and generate statistics with region and depth**

import sys

if(len(sys.argv)!=3):

print("Provide Input File and Output Directory")

sys.exit(0)from pyspark import SparkContext

sc =SparkContext()

f = sc.textFile(sys.argv[1])

temp=f.map(lambda x: (x.split(',')[11],float(x.split(',')[9])))

maxi=temp.reduceByKey(lambda a,b:a if a>b else b)

maxi.saveAsTextFile(sys.argv[2])

**Write a spark program to analyze the given Earthquake data and generate statistics with region and latitude**

import sys

if(len(sys.argv)!=3):

print("Provide Input File and Output Directory")

sys.exit(0)

from pyspark import SparkContext

sc =SparkContext()

f = sc.textFile(sys.argv[1])

temp=f.map(lambda x: (x.split(',')[11],float(x.split(',')[6])))

maxi=temp.reduceByKey(lambda a,b:a if a>b else b)

maxi.saveAsTextFile(sys.argv[2])

**Write a spark program to analyze the given Earthquake data and generate statistics with region and longitude**

import sys

if(len(sys.argv)!=3):

print("Provide Input File and Output Directory")

sys.exit(0)

from pyspark import SparkContext

sc =SparkContext()

f = sc.textFile(sys.argv[1])

temp=f.map(lambda x: (x.split(',')[11],float(x.split(',')[7])))

maxi=temp.reduceByKey(lambda a,b:a if a>b else b)

maxi.saveAsTextFile(sys.argv[2])

**Write a spark program to analyze the given Insurance data and generate a statistics report with the construction building name and the count of building.**

import sys

if(len(sys.argv)!=3):

print("Provide Input File and Output Directory")

sys.exit(0)

from pyspark import SparkContext

sc =SparkContext()

f = sc.textFile(sys.argv[1])

temp=f.map(lambda x: (x.split(',')[16],1))

data=temp.countByKey()

dd=sc.parallelize(data.items())

dd.saveAsTextFile(sys.argv[2])

**Write a spark program to analyze the given Insurance data and generate a statistics report with the county name and its frequency.**

import sys

if(len(sys.argv)!=3):

print("Provide Input File and Output Directory")

sys.exit(0)

from pyspark import SparkContext

sc =SparkContext()

f = sc.textFile(sys.argv[1])

temp=f.map(lambda x: (x.split(',')[2],1))

data=temp.countByKey()

dd=sc.parallelize(data.items())

dd.saveAsTextFile(sys.argv[2])

**Write a map-reduce program to analyze the given employee record data and generate a statistics report with the total Sales for female and male employees**

import sys

if(len(sys.argv)!=3):

print("Provide Input File and Output Directory")

sys.exit(0)

from pyspark import SparkContext

sc =SparkContext()

f = sc.textFile(sys.argv[1])

temp=f.map(lambda x: (x.split('\t')[3],float(x.split('\t')[8])))

total=temp.reduceByKey(lambda a,b : a+b)

total.saveAsTextFile(sys.argv[2])

**Write a map-reduce program to analyze the given sales records over a period of time and generate data about the country’s total sales, and the total number of the products**

import sys

if(len(sys.argv)!=3):

print("Provide Input File and Output Directory")

sys.exit(0)

from pyspark import SparkContext

sc =SparkContext()

f = sc.textFile(sys.argv[1])

temp=f.map(lambda x: (x.split(',')[7],1))

data=temp.countByKey()

dd=sc.parallelize(data.items())

dd.saveAsTextFile(sys.argv[2])

**Write a map-reduce program to analyze the given sales records over a period of time and generate data about the country’s total sales and the frequency of the payment mode.**

import sys

if(len(sys.argv)!=3):

print("Provide Input File and Output Directory")

sys.exit(0)

from pyspark import SparkContext

sc =SparkContext()

f = sc.textFile(sys.argv[1])

temp=f.map(lambda x: (x.split(',')[3],1))

data=temp.countByKey()

dd=sc.parallelize(data.items())

dd.saveAsTextFile(sys.argv[2])