Greeshmika Korrapati - U00932594

Cloud Computing Project -2

1.1

Commands used:

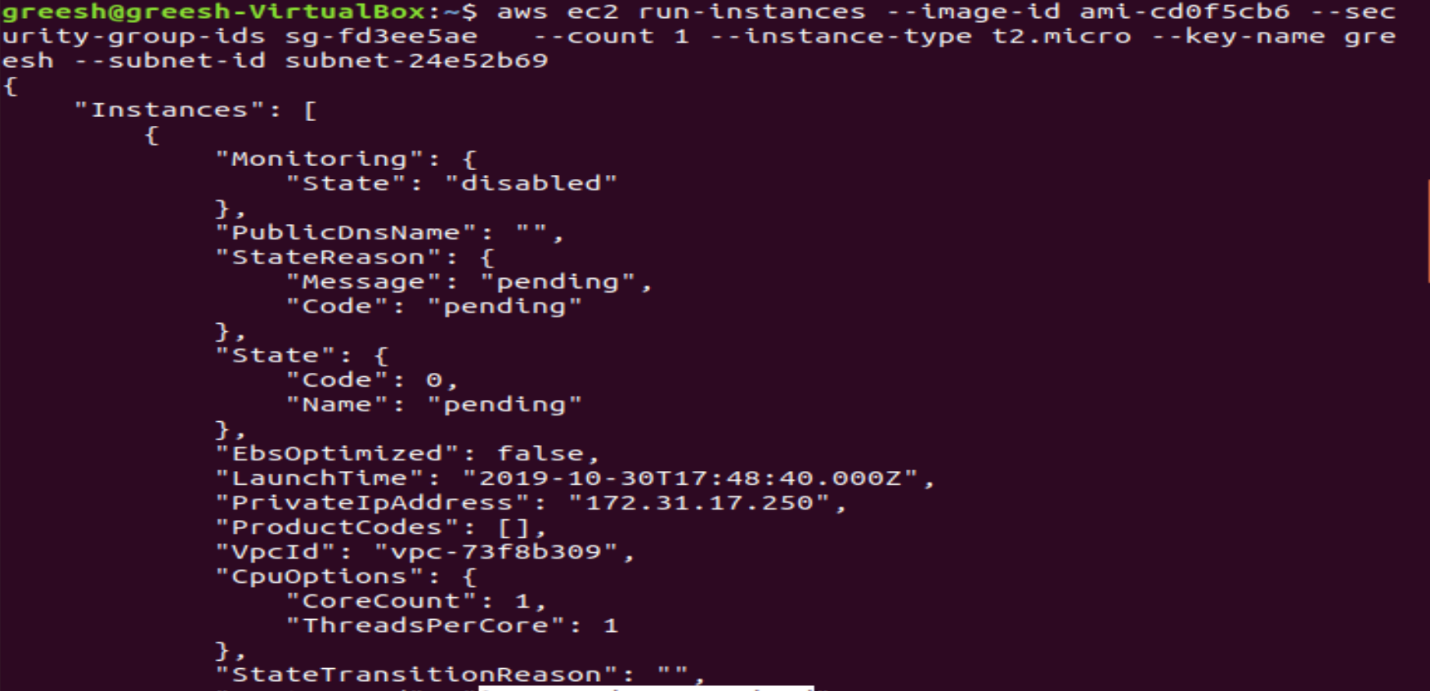
aws ec2 run-instances --image-id ami-cd0f5cb6 --security-group-ids sg-fd3ee5ae --count-1 --instance-type t2.micro --key-name greesh --subnet-id subnet-24e52b69

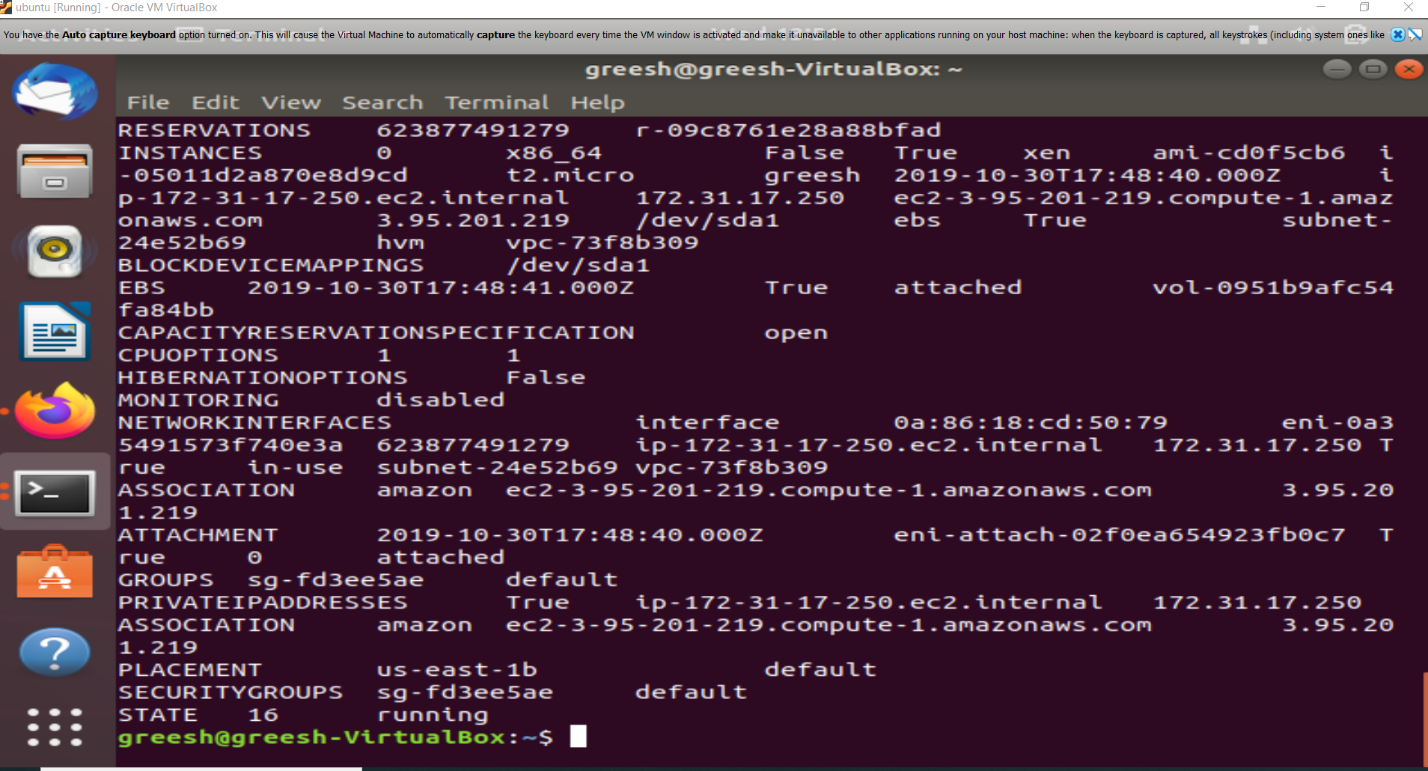
aws ec2 describe-instances

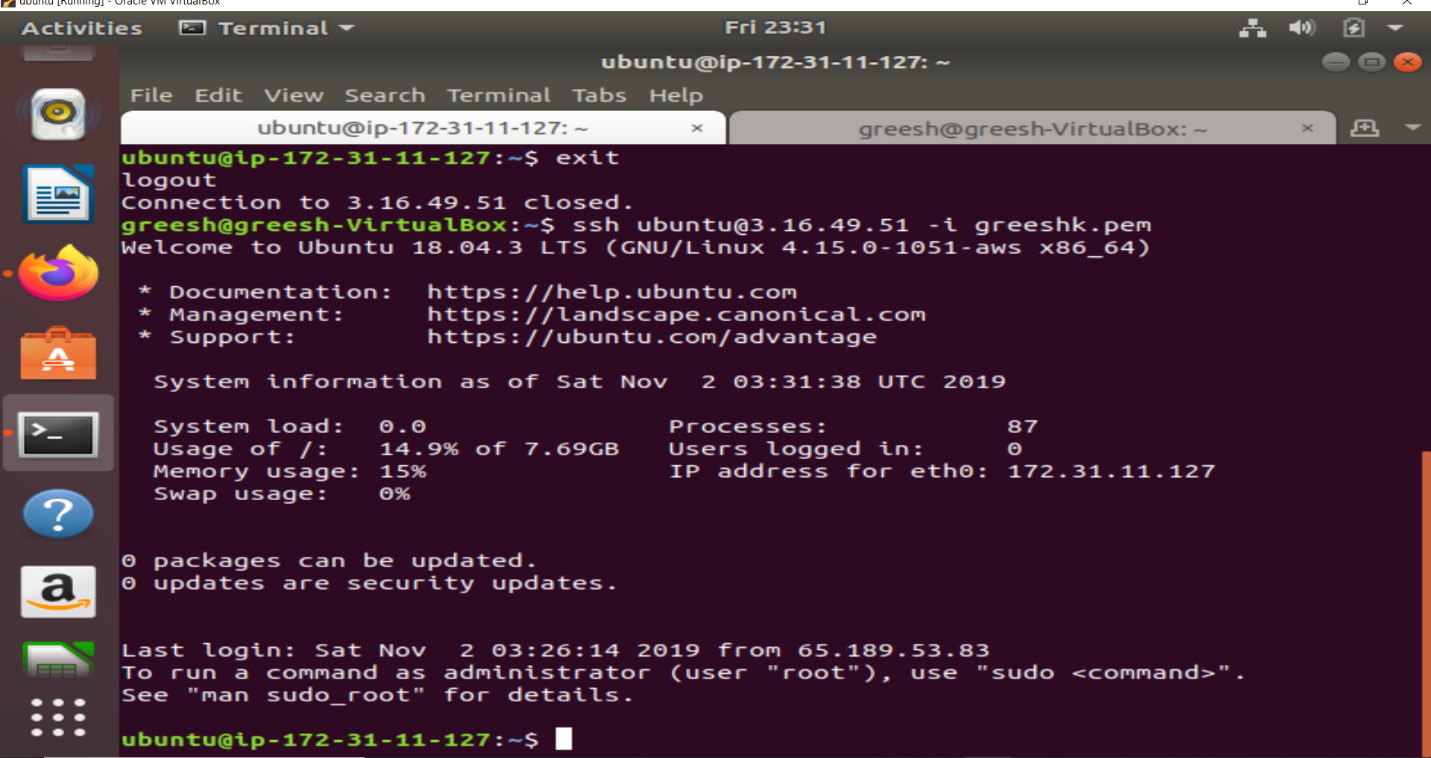
ssh [ubuntu@3.16.49.51](mailto:ubuntu@3.16.49.51) -i greeshk.pem

aws ec2 terminate-instances --instance-ids i- 05011d2a870e8d9cd

successfully created instance and connected to ssh







1.2

import boto3

import time

import sys

ec2 = boto3.resource('ec2')

client = boto3.client('ec2')

def create\_instance(num\_instances):

return ec2.create\_instances(ImageId='ami-0d5d9d301c853a04a',

InstanceType='t2.micro',

MinCount=1,

MaxCount=num\_instances)

def print\_instanceid(instanceid):

instancewait = client.get\_waiter('instance\_running')

instancecreated=[]

for id in instanceid:

instancecreated.append(id)

instancewait.wait(InstanceIds=[id])

print(instancecreated)

return

if \_\_name\_\_ == '\_\_main\_\_':

print("creating instances")

newinstance=create\_instance(3)

instanceid=(i.id for i in newinstance)

print\_instanceid(instanceid)

1.3

import boto3

import botocore

s3=boto3.resource('s3')

client=boto3.client('s3')

required\_files=client.list\_objects\_v2(Bucket = 'wsu2017fall')

#print(required\_files)

frombucket=s3.Bucket('wsu2017fall')

tobucket=s3.Bucket('greeshb')

for i in frombucket.objects.all():

keys=i.key

print("Keys in wsu2017fall: ",keys)

print("--------------------")

def reading\_bucket(buckettoread,keysin):

if (required\_files.get('Contents', None) is not None):

print("Reading files")

for i in frombucket.objects.all():

try:

filedata=i.get()['Body'].read().decode()

print(filedata)

except:

print("Directory is empty")

else:

print("Bucket is empty")

return filedata

def copying\_files():

if(required\_files.get('Contents',None)is not None):

for f in required\_files.get('Contents',None):

for filekeys in frombucket.objects.all():

sourcebucket={'Bucket':'wsu2017fall',

'Key':filekeys}

try:

s3.meta.client.copy(sourcebucket,tobucket,filekeys)

except:

pass

print("copied file successfully to {}".format(tobucket))

else:

print("Bucket is empty")

return

for i in tobucket.objects.all():

print("keys in my bucket: ",i.key)

print("--------------------")

if \_\_name\_\_ == '\_\_main\_\_':

lines\_data=reading\_bucket(frombucket,keys)

copying\_files()

1.4

FROM ubuntu

RUN apt-get update && apt-get install -y \

apt-utils \

openssh-server \

default-jre && \

python3 && \

apt-get install -y python3 python3-setuptools && \

apt-get clean && \

ln -s /usr/bin/python3 /usr/bin/python && \

apt-get install -y python-pip \

pip install py4j \

pip -V \

python -V \

python --version

RUN apt-get install -y curl

RUN echo -e "Installations done successfully" >>~/.bashrc

#RUN update-rc.d ssh defaults && ssh-keygen -f ~/.ssh/id\_rsa -t rsa -N '' && cat ~/.ssh/id\_rsa.pub >> ~/.ssh/authorized\_keys

#COPY authorized\_keys / ~/.ssh/authorized\_keys

RUN ssh-keygen -A

RUN mkdir /var/run/sshd && chmod 755 /var/run/sshd

#RUN apt-get install -y curl

RUN curl https://www-us.apache.org/dist/spark/spark-2.4.4/spark-2.4.4-bin-hadoop2.7.tgz >spark-2.4.4.tgz

RUN tar -xvf spark-2.4.4.tgz -C /usr/ && rm spark-2.4.4.tgz

COPY greesh.txt /

COPY wordcount.py /

RUN service ssh start

EXPOSE 22

Commands used:

docker build -t greeshmika/pysparkwc.latest .

docker run -it -d -p 22 greeshmika/pysparkwc:latest

sudo docker push Greeshmika/pysparkwc:latest

ssh -I “greeshkey.pem” [ubuntu@ec2-18-191-166-141.us-east-2.copute.amazonaws.com](mailto:ubuntu@ec2-18-191-166-141.us-east-2.copute.amazonaws.com)

sudo snap install docker

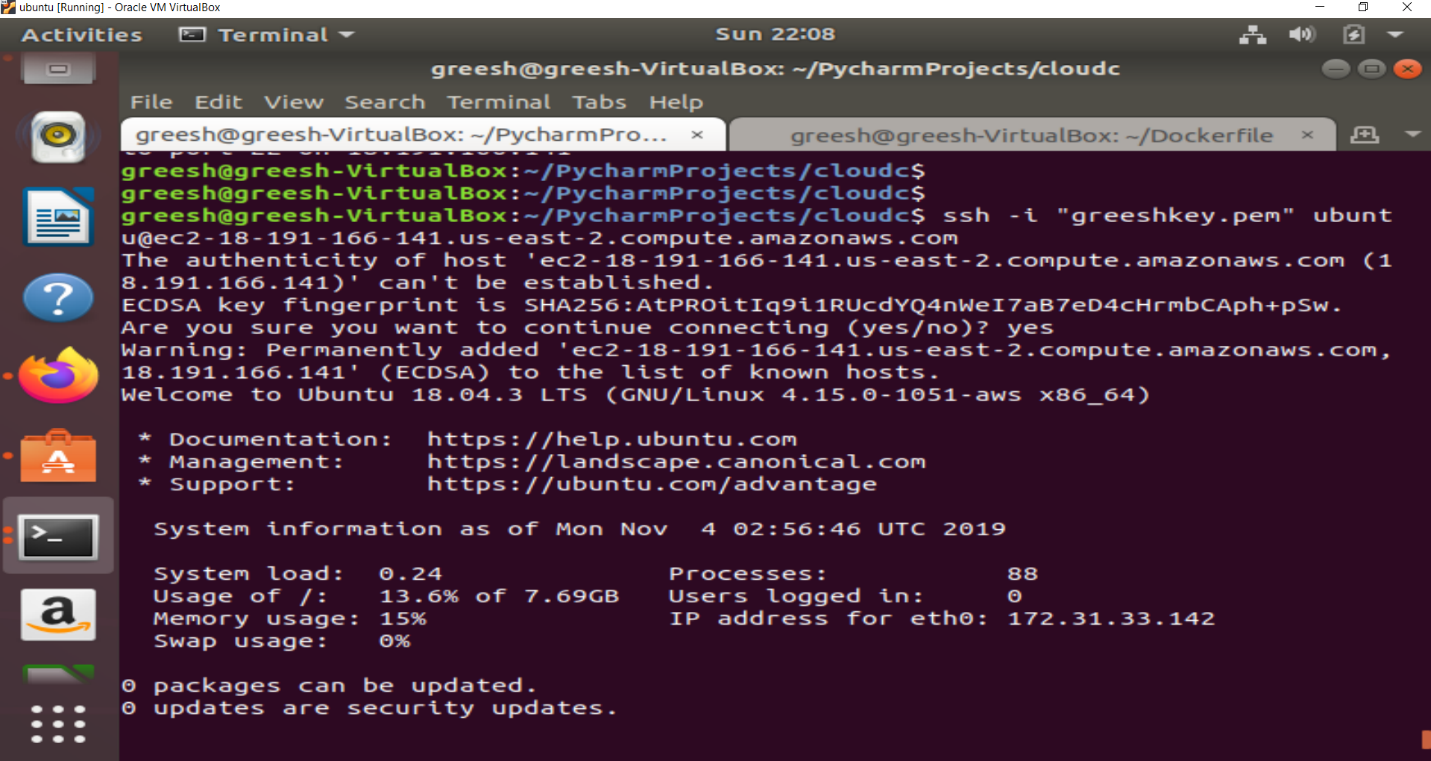
sudo chmod 666 /var/run/docker.sock

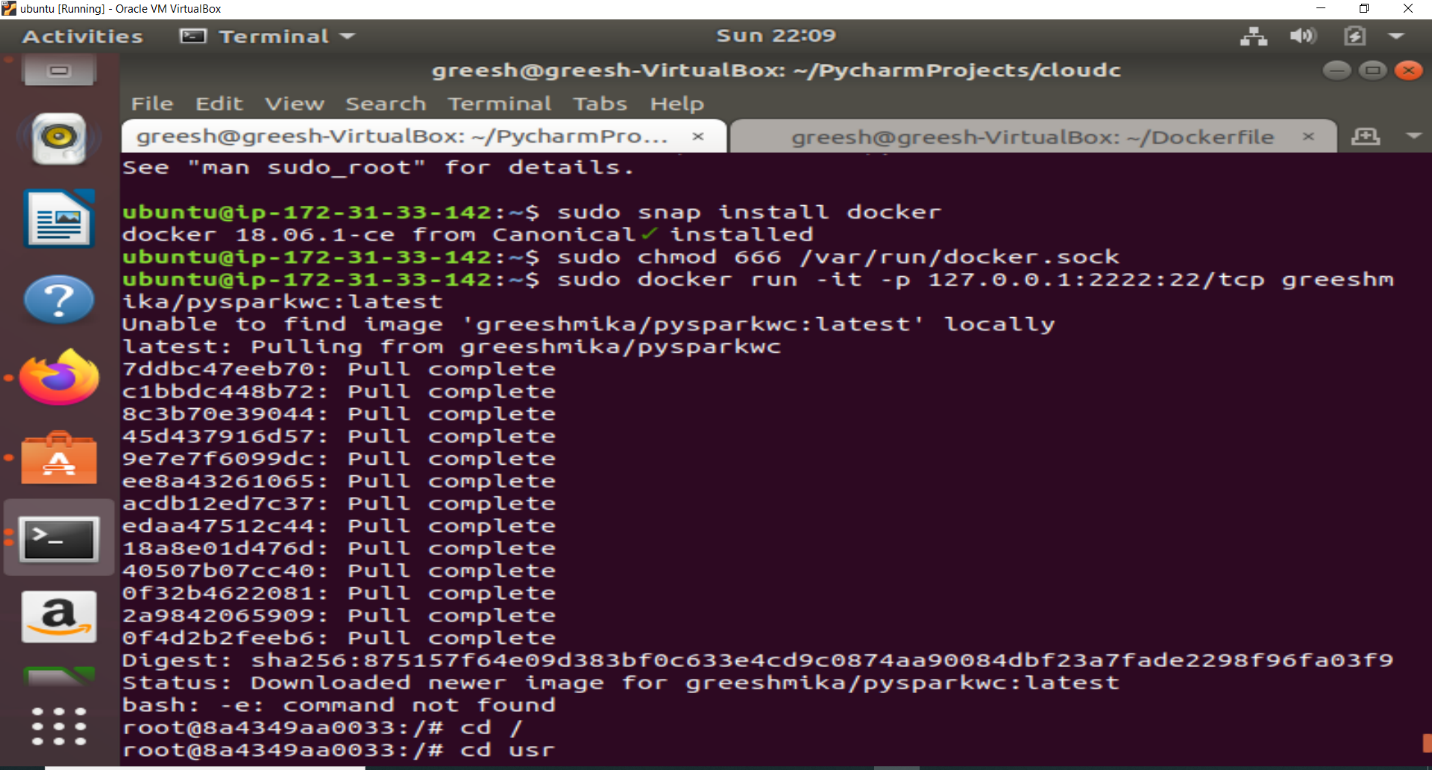
sudo docker run -it -p 127.0.0.1:2222:22/tcp greeshmika/pysparkwc:latest

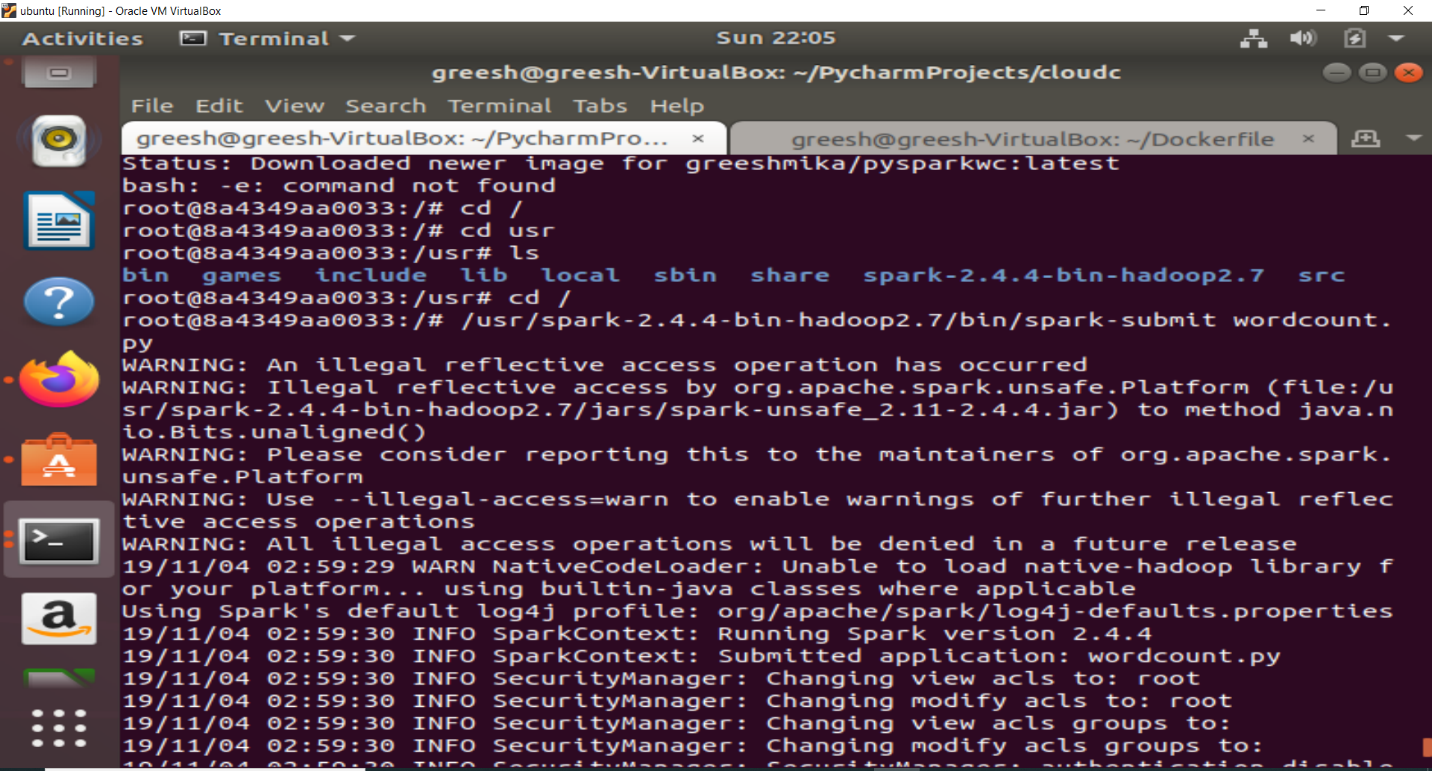
usr/spark-2.4.4-bin-hadoop-2.7/bin/spark-submit wordcount.py

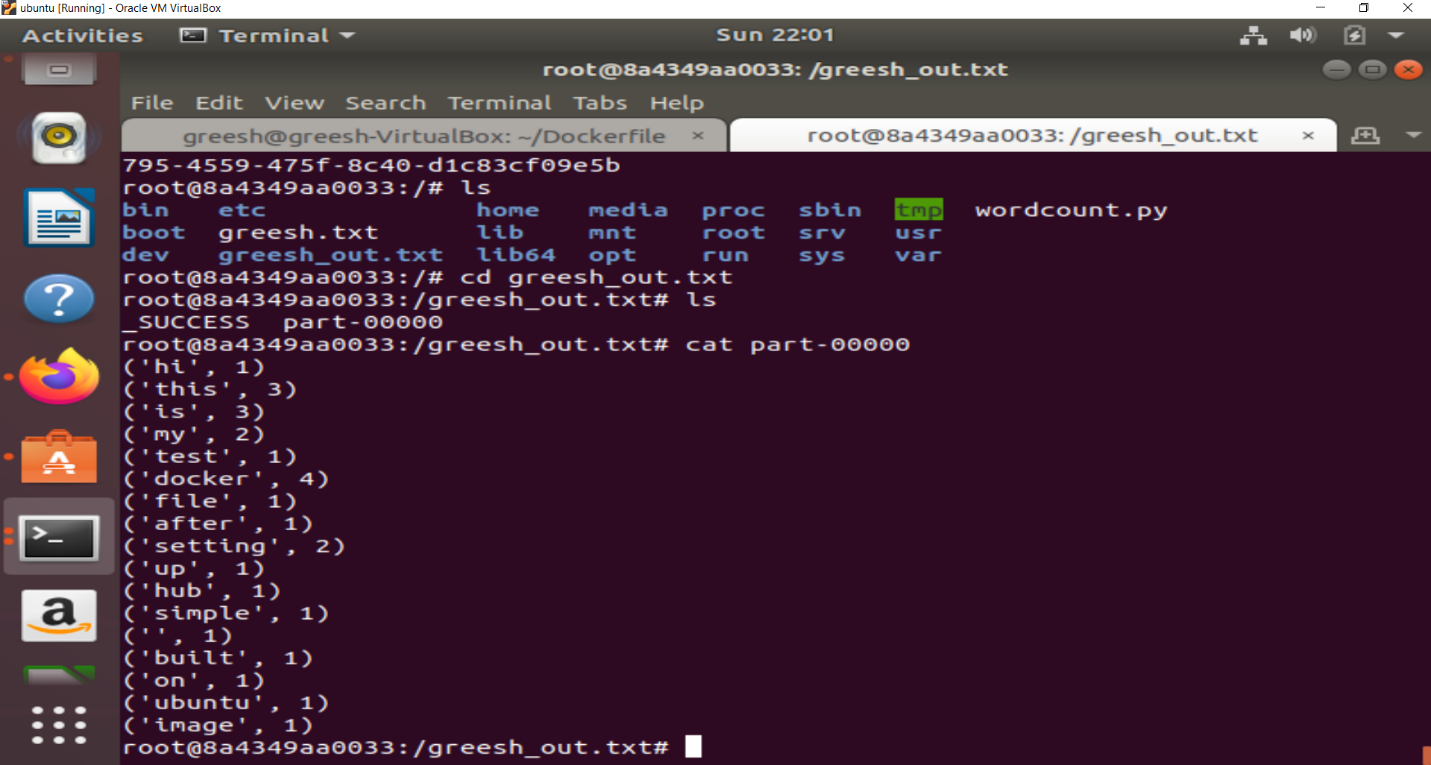
cd greesh\_out.txt

cat part-00000









Docker hub image link:

<https://cloud.docker.com/repository/docker/greeshmika/pysparkwc>

2.1

import boto3

import paramiko

import time

import socket

import sys

ec2 = boto3.resource('ec2')

client = boto3.client('ec2')

cp=paramiko.client.SSHClient()

cp.load\_system\_host\_keys()

def create\_instance(num\_instances):

return ec2.create\_instances(ImageId='ami-0d5d9d301c853a04a',

KeyName="greeshkey",

InstanceType='t2.micro',

MinCount=1,

MaxCount=num\_instances)

def print\_instanceid(instanceid):

instancewait = client.get\_waiter('instance\_running')

instancecreated = []

for id in instanceid:

instancecreated.append(id)

instancewait.wait(InstanceIds=[id])

print(instancecreated)

return

def ipaddress(instanceId):

instances\_created=client.describe\_instances(InstanceIds =[instanceId])

ip=instances\_created['Reservations'][0]['Instances'][0] ['NetworkInterfaces'][0] ['Association']['PublicIp']

return ip

def status\_monitor(newinst):

while(True):

for inst in newinstance:

pemkey=paramiko.RSAKey.from\_private\_key\_file("greeshkey.pem")

#print(pemkey)

cp.set\_missing\_host\_key\_policy(paramiko.AutoAddPolicy())

print("passed")

print(ipaddress(inst.id))

cp.connect(hostname=ipaddress(inst.id), username="ubuntu",pkey=pemkey)

print("connected to ssh")

stdin,stdout,stderr =cp.exec\_command('top -bn1 | grep Cpu')

print(inst.id ,'\t',stdout.readlines())

time.sleep(10)

return

if \_\_name\_\_ == '\_\_main\_\_':

print("creatinginstances")

newinstance = create\_instance(2)

instanceid = (i.id for i in newinstance)

print\_instanceid(instanceid)

status\_monitor(newinstance)

2.2

import boto3

import paramiko

import time

import socket

import sys

ec2 = boto3.resource('ec2')

client = boto3.client('ec2')

cp=paramiko.client.SSHClient()

cp.load\_system\_host\_keys()

errorlog=[]

def create\_instance(num\_instances):

return ec2.create\_instances(ImageId='ami-0d5d9d301c853a04a',

KeyName="greeshkey",

InstanceType='t2.micro',

MinCount=1,

MaxCount=num\_instances)

def print\_instanceid(instanceid):

instancewait = client.get\_waiter('instance\_running')

instancecreated = []

for id in instanceid:

instancecreated.append(id)

instancewait.wait(InstanceIds=[id])

print(instancecreated)

return

def ipaddress(instanceId):

instances\_created=client.describe\_instances(InstanceIds =[instanceId])

ip=instances\_created['Reservations'][0]['Instances'][0] ['NetworkInterfaces'][0] ['Association']['PublicIp']

return ip

def paramiko\_monitor(newinstance):

try:

for inst in newinstance:

pemkey=paramiko.RSAKey.from\_private\_key\_file("greeshkey.pem")

cp.set\_missing\_host\_key\_policy(paramiko.AutoAddPolicy())

print("key file is extracted")

cp.connect(hostname=ipaddress(inst.id), username="ubuntu",pkey=pemkey)

print("connected to ssh")

stdin,stdout,stderr =cp.exec\_command('sudo apt-get update && sudo apt-get install -y docker.io && sudo docker run -d -t ubuntu sh')

print("Docker Installation started")

print("Docker run command is successfull")

for instaid in stdout:

print(inst.id,'\t',ipaddress(inst.id),'\t',instaid)

except:

errorlog.append(inst.id)

for i in errorlog:

print(i)

docker\_monitor(newinstance)

return

def docker\_monitor(newinstance):

while(True):

for inst in newinstance:

pemkey = paramiko.RSAKey.from\_private\_key\_file("greeshkey.pem")

cp.connect(hostname=ipaddress(inst.id), username="ubuntu", pkey=pemkey)

print("connected to ssh")

stdin, stdout, stderr = cp.exec\_command('sudo docker ps | grep ubuntu')

for docid in stdout:

doc\_container=docid.split()[0]

stdin, stdout, stderr = cp.exec\_command('sudo docker exec {} top -bn1 | grep Cpu'.format(doc\_container))

for k in stdout:

print('{} \t {} \t {}'.format(inst.id,doc\_container,k))

time.sleep(10)

return

if \_\_name\_\_ == '\_\_main\_\_':

print("creating instances")

newinstance = create\_instance(2)

instanceid = (i.id for i in newinstance)

print\_instanceid(instanceid)

paramiko\_monitor(newinstance)