Part1 In-Depth Study:

**Steps:**

1. EC2 m4.large instance should be instantiated (Elastic search >2gb RAM)
2. Security Group configuration:

**Edit inbound rules**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Type** | **Protocol** | **Port range** | **Source** | **Description - optional** |
| HTTP | TCP | 80 | 0.0.0.0/0 | - |
| SSH | TCP | 22 | 0.0.0.0/0 | - |
| Custom TCP | TCP | 5601 | 0.0.0.0/0 | - |
| HTTPS | TCP | 443 | 0.0.0.0/0 | - |

1. wget -qO - https://artifacts.elastic.co/GPG-KEY-elasticsearch | sudo apt-key add -
2. echo "deb https://artifacts.elastic.co/packages/7.x/apt stable main" | sudo tee -a /etc/apt/sources.list.d/elastic-7.x.list
3. echo "deb https://artifacts.elastic.co/packages/oss-7.x/apt stable main" | sudo tee -a /etc/apt/sources.list.d/elastic-7.x.list
4. sudo apt-get update && sudo apt-get install elasticsearch
5. Open the Elasticsearch configuration file at: */etc/elasticsearch/elasticsearch.yml*, and apply the following configurations:

network.host: "localhost"

http.port: 9200

cluster.initial\_master\_nodes: ["<PrivateIP"]

1. Start the Elasticsearch service:
2. sudo service elasticsearch start
3. Verify the installation by cURLing:

sudo curl http://localhost:9200

1. sudo apt-get install default-jre
2. java -version
3. sudo apt-get install logstash

Part2 In-Depth study:

Have Spark installed locally. Also learn how to connect from Jupyter notebook with AWS EMR. Take any example to submit a Spark job from inside Jupyter notebook locally and in AWS.

For local:

1. Wordcount program:

import sys

from pyspark import SparkContext, SparkConf

if \_\_name\_\_ == "\_\_main\_\_":

# create Spark context with necessary configuration

sc = SparkContext("local","PySpark Word Count Example")

# read data from text file and split each line into words

words = sc.textFile("it732\_test.txt").flatMap(lambda line: line.split(" "))

# count the occurrence of each word

wordCounts = words.map(lambda word: (word, 1)).reduceByKey(lambda a,b:a +b)

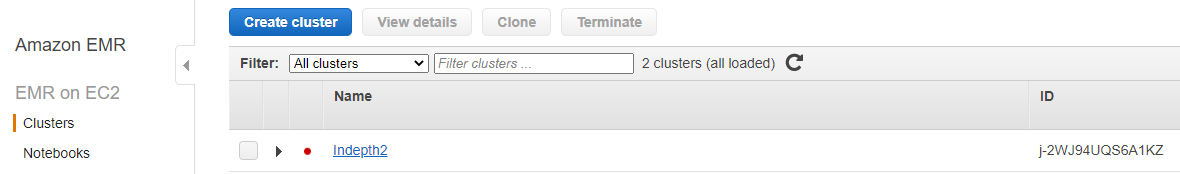
#it gives the word count of an article on coronavirus

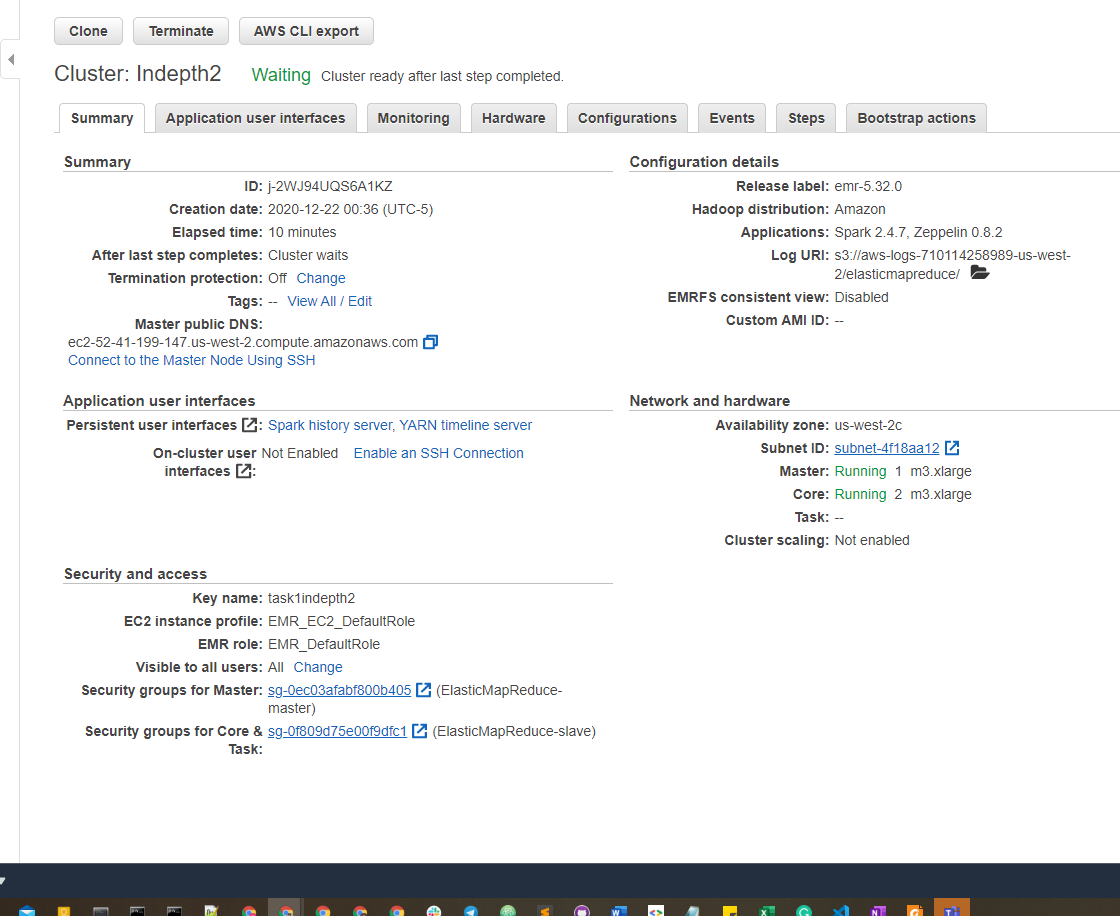
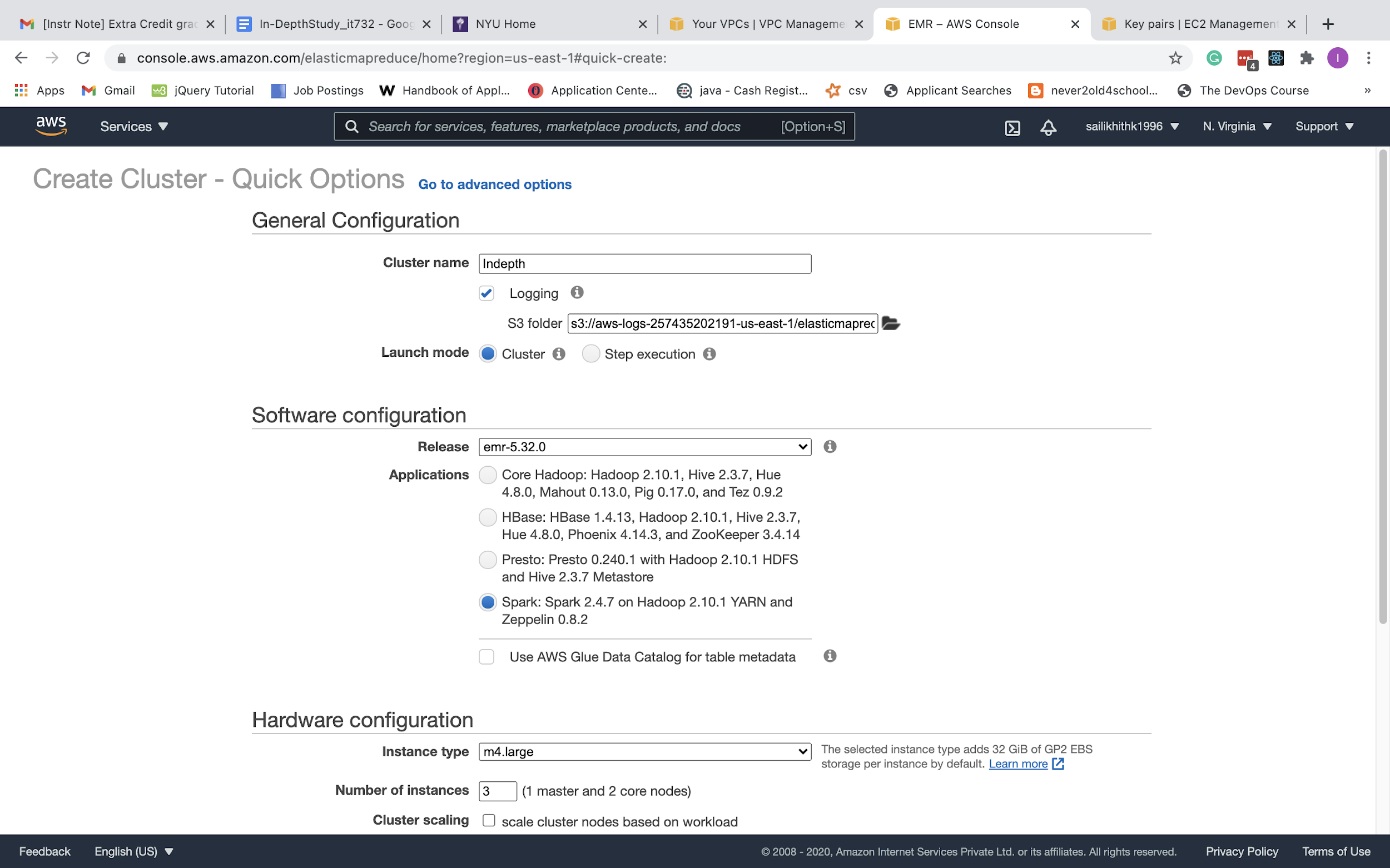
for x in wordCounts.collect():

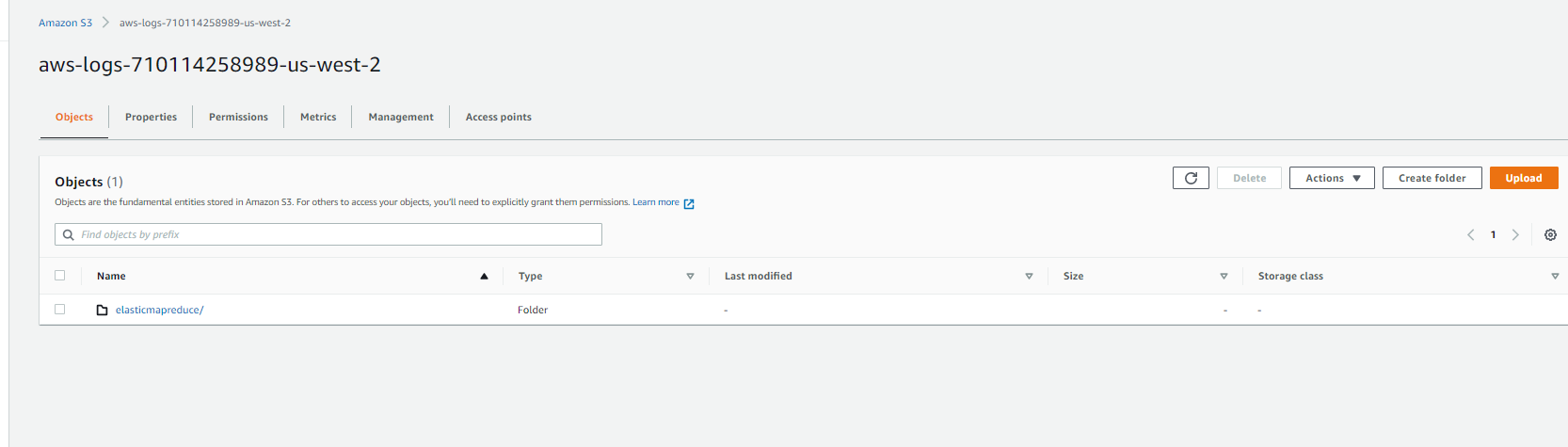
print(x)

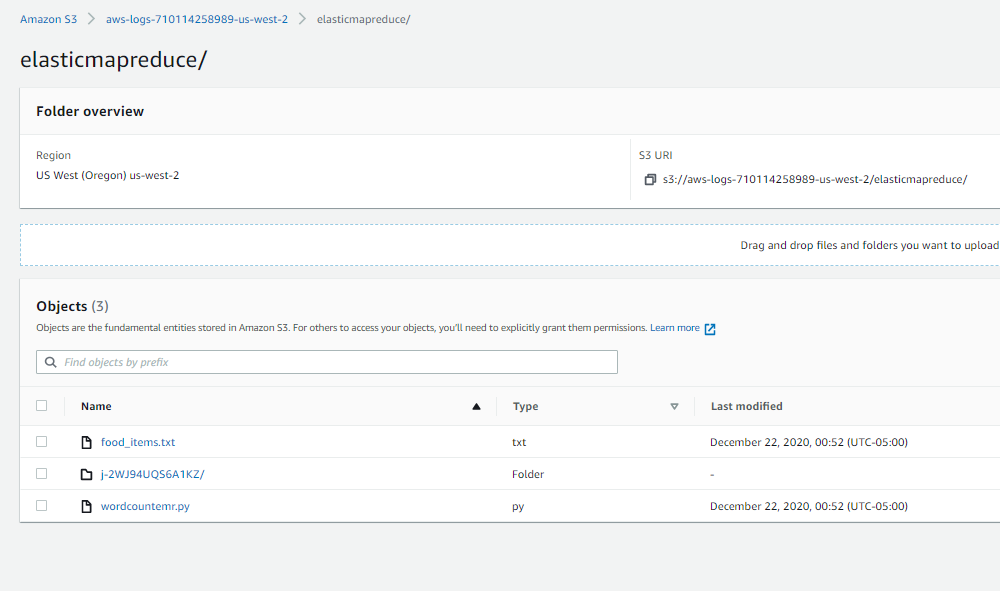
1. Run this in jupyter notebook

AWS EMR:

1. Make sure a default VPC is running in the background.
2. Go to AWS EMR. The cluster name is ‘indepth2’. S3 bucket will be created by the cluster. In software configurations, for applications choose the spark option.

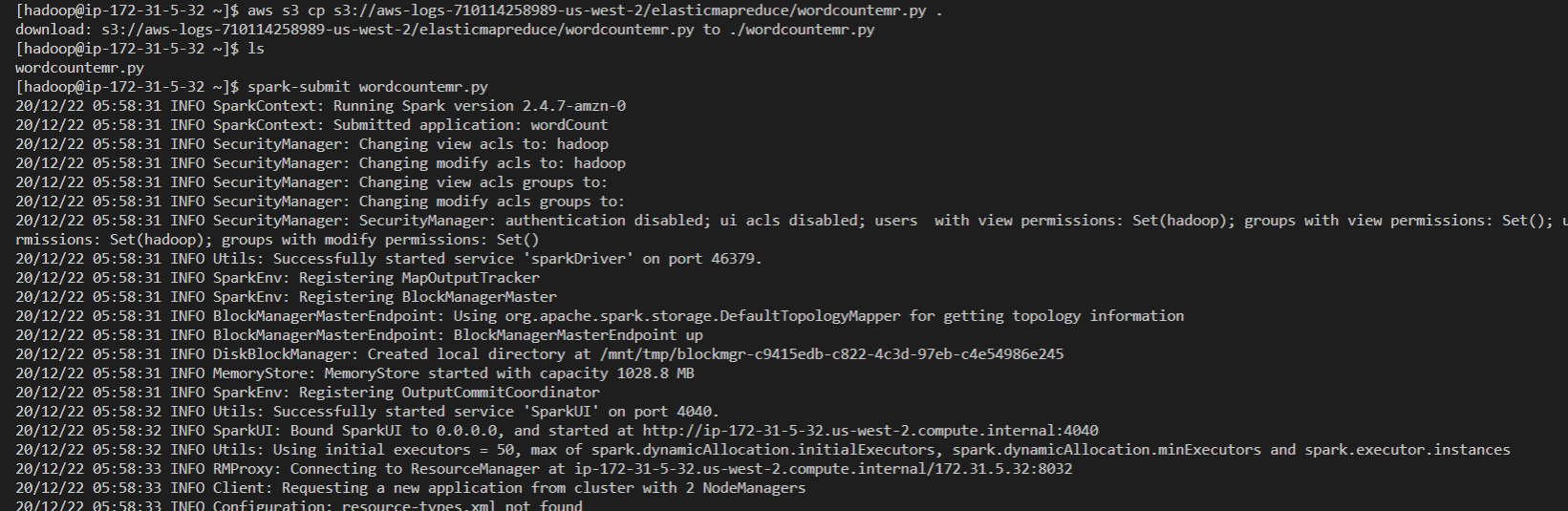


1. For security and access select an existing ec2 key pair.
2. It takes about 15 minutes for the cluster to set up.
3. Go to security groups and choose the security group for the master node. Go to edit inbound rules and add ‘SSH’ and source ‘anywhere’.
4. After that click on save rules and wait for your cluster to deploy.
5. This creates a S3 bucket. 
6. Upload text file and python file to S3 bucket.

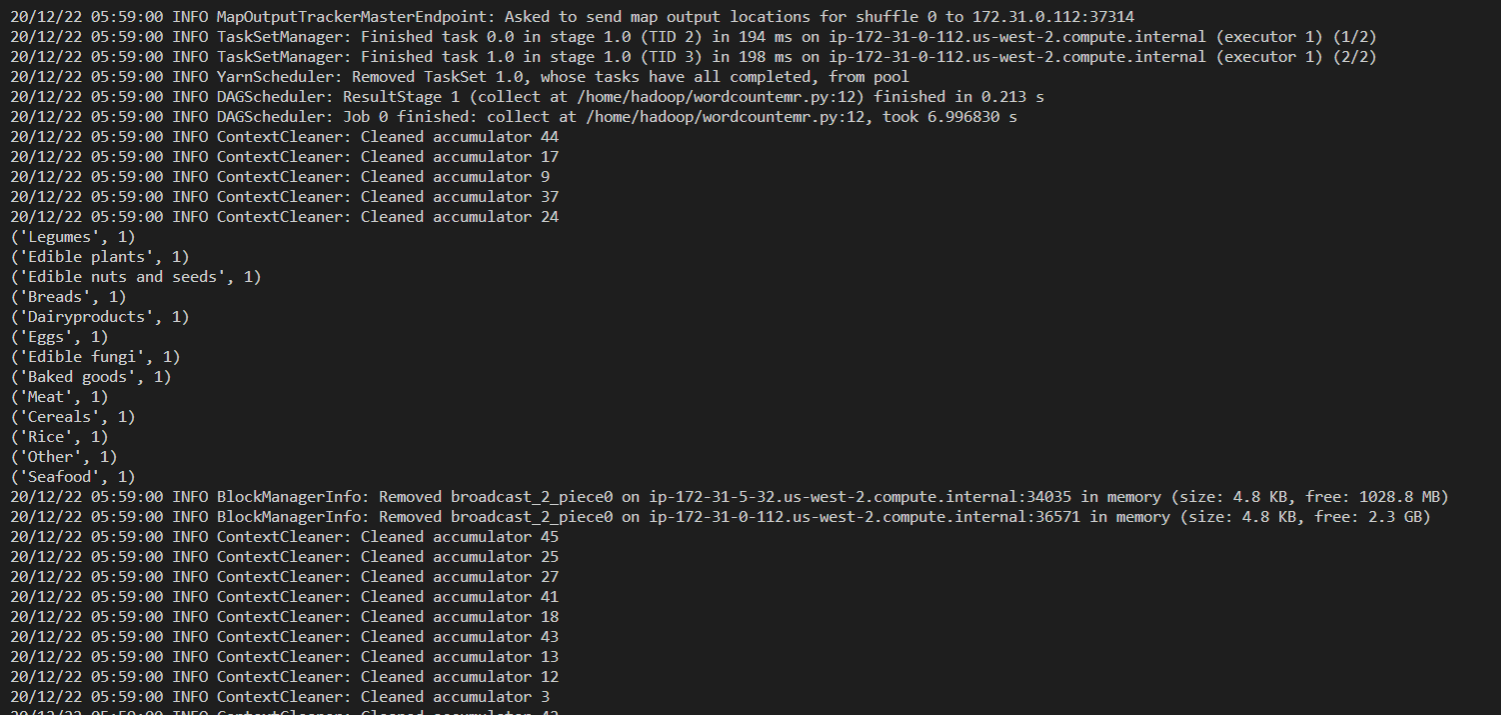


1. SSH from local machine to you hadoop ec2 instance(masternode) to use EMR, you will see the EMR logo
2. Copy python file from s3 to EMR:

**aws s3 cp s3://aws-logs-710114258989-us-west-2/elasticmapreduce/wordcountemr.py**



1. When you run a spark-submit your script starts to run and outputs are displayed



1. The code for the above python script is:

**import sys**

**from pyspark import SparkContext, SparkConf**

**if \_\_name\_\_ == "\_\_main\_\_":**

**# create Spark context with necessary configuration**

**sc = SparkContext(appName='wordCount')**

**# read data from text file and split each line into words**

**words = sc.textFile("s3://aws-logs-710114258989-us-west-2/elasticmapreduce/food\_items.txt").flatMap(lambda line: line.split(","))**

**# count the occurrence of each word**

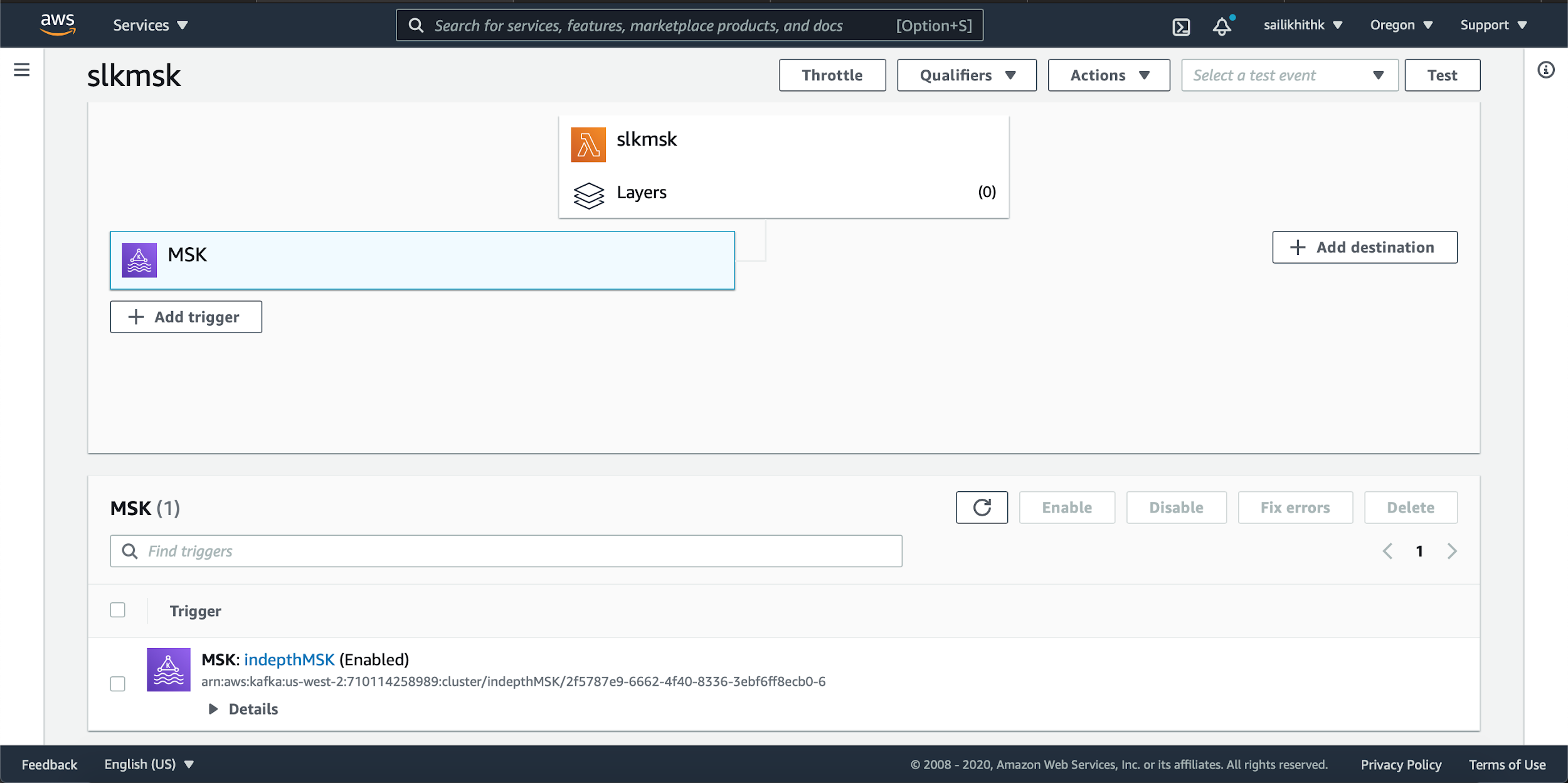
**wordCounts = words.map(lambda word: (word, 1)).reduceByKey(lambda a,b:a +b)**

**for x in wordCounts.collect():**

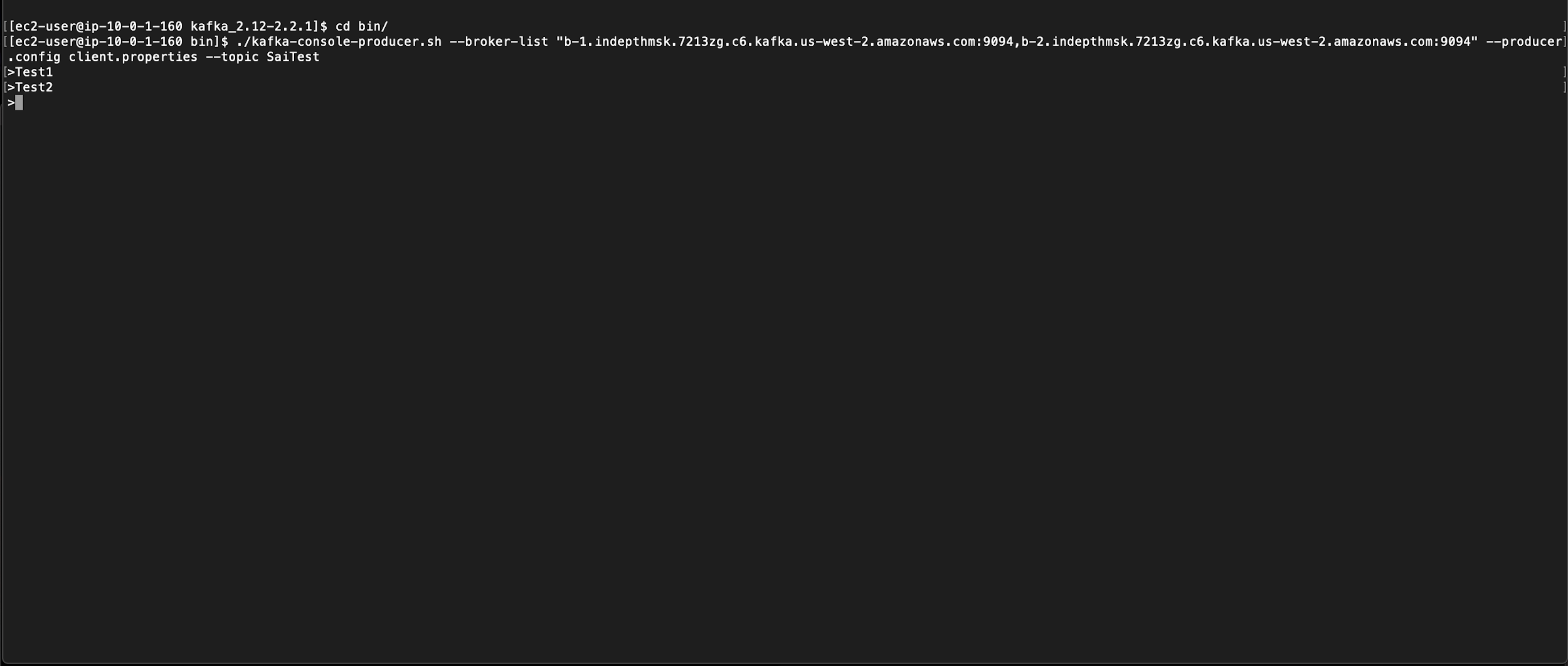
**print(x)**

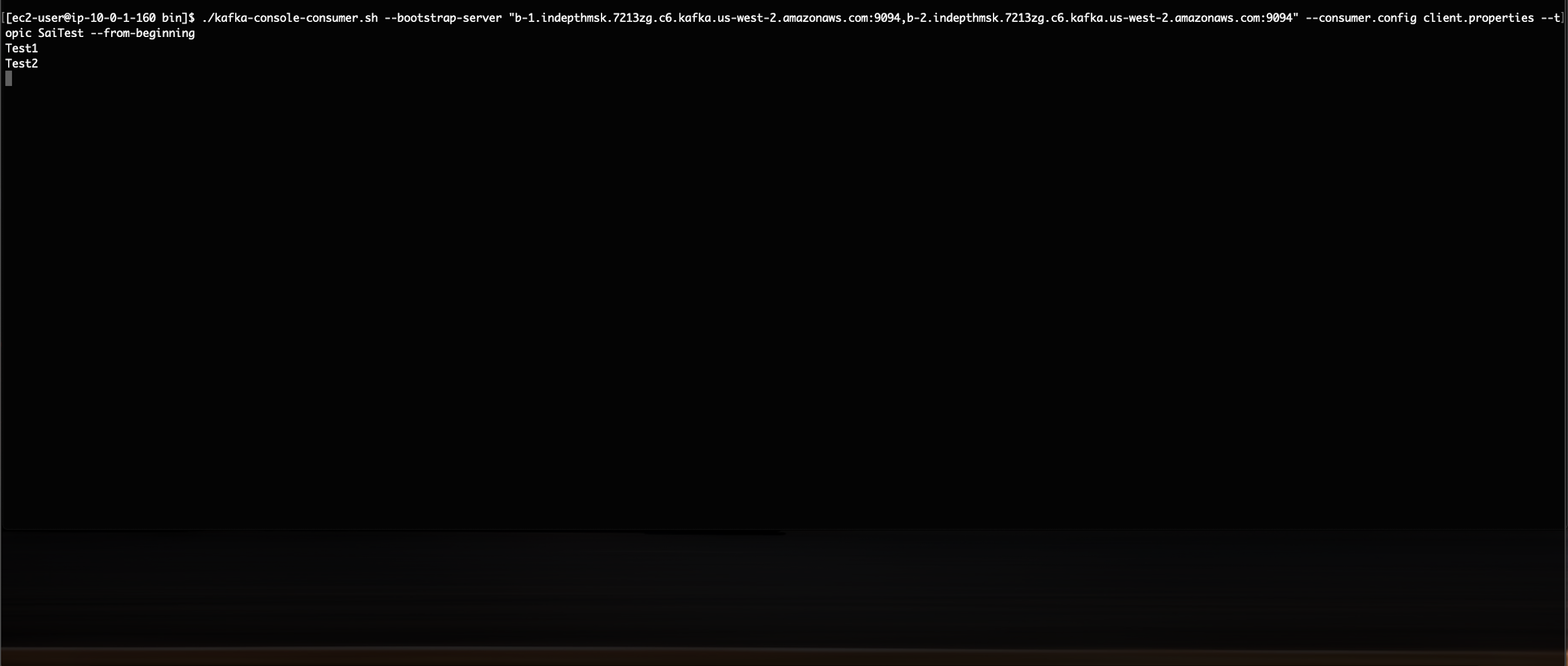
**Once I’m done with these, I started assignment 2.**

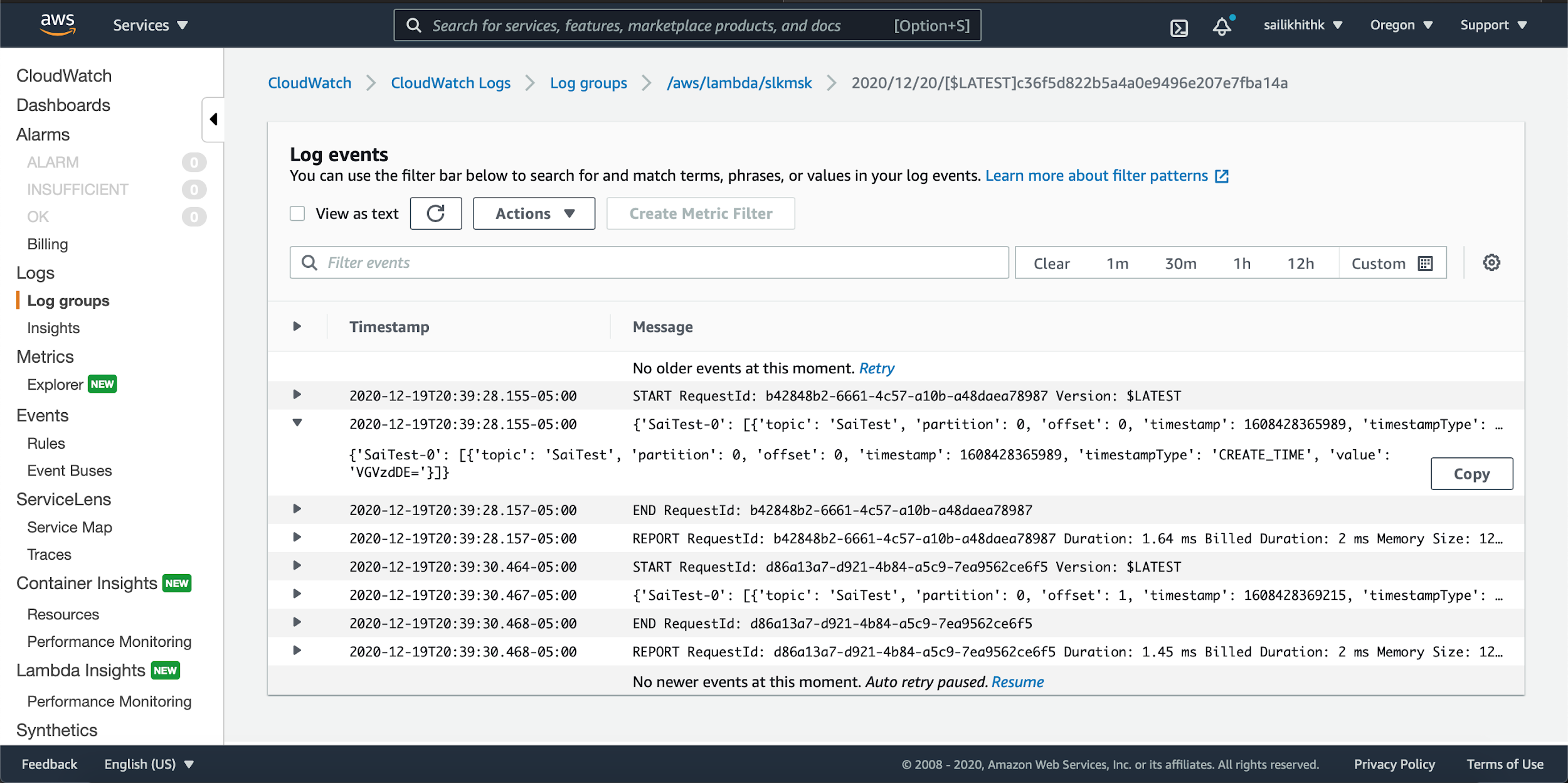
This is the configuration of the lambda for this task which is having Kafka stream as the trigger.



1. **Created the producer**

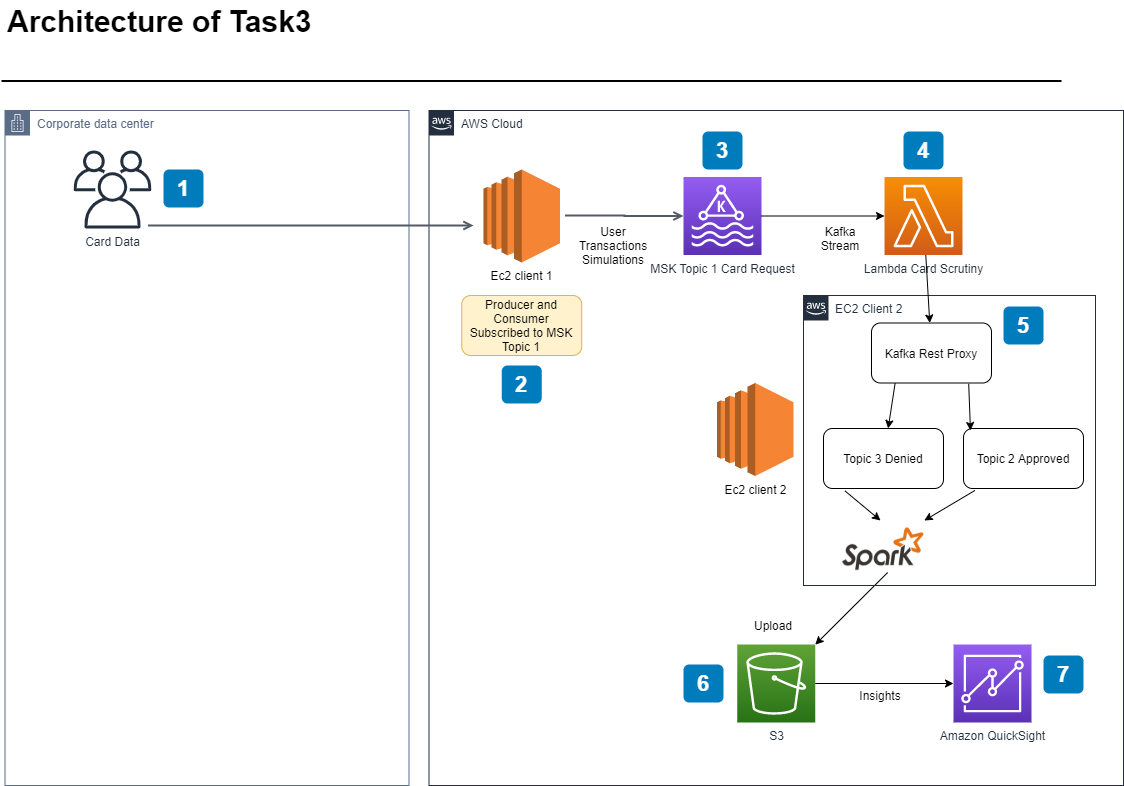
****

1. **The topics created from SaiTest are appearing here. Test1, Test2.**
2. These will be appearing in consumer.
3. 
4. We can cross check the event in cloudwatch logs as follows:



Task 3 In-Depth study:

**Architecture:**

****

### **Demo:** <https://stream.nyu.edu/media/slkrrcitTask3/1_1bh2sh2v>