**9. Deploy a EMR Cluster for doing Big data analytics using Spark**

**HARDWARE REQUIREMENTS**: Core I5 Processor, 4 GB RAM, 40GB HDD

**SOFTWARE REQUIREMENTS**: Amazon AWS, Amazon EMR, Amazon S3, GIT, Gradle 6.x, Java 1.8

**Description:** Analyzing Big Data on Apache Spark Cluster setup using Amazon EMR.

**Amazon EMR** cluster provides a managed Hadoop framework that makes it easy, fast and cost-effective to process vast amounts of data dynamically scalable on Amazon EC2 instances.

It is possible to run popular distributed frameworks such as Apache Spark and HBase on Amazon EMR and interact with data in other AWS data stores such as Amazon S3 and Amazon DynamoDB.

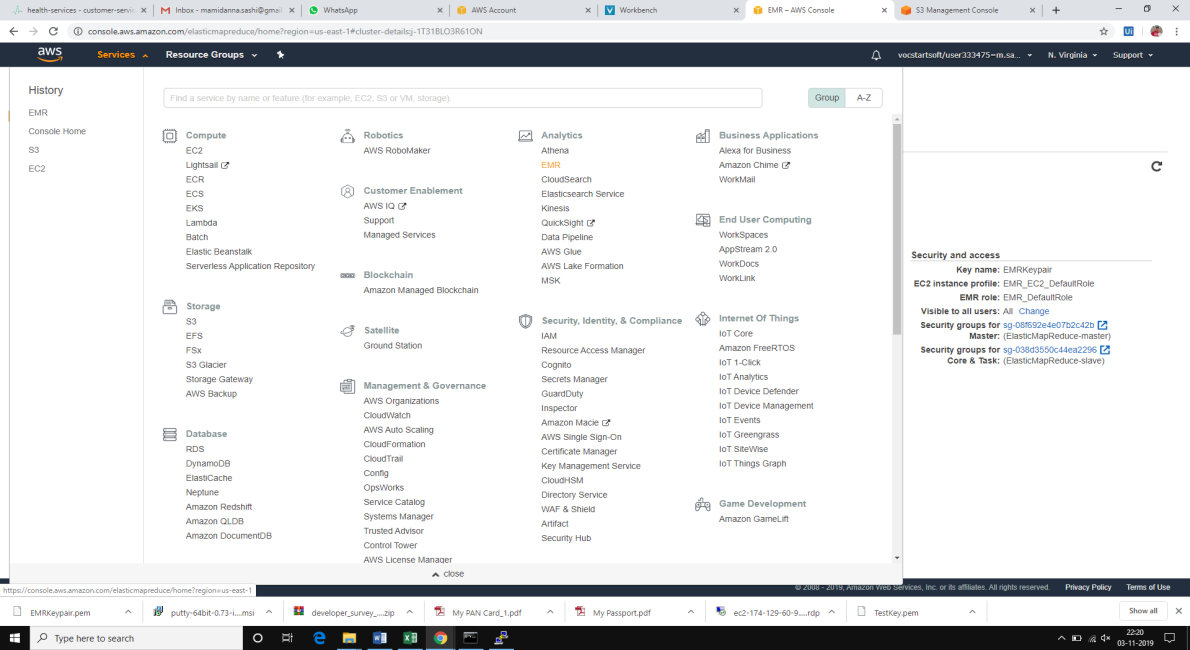
In this experiment we will run out Spark application on top of Hadoop cluster by processing the input data source into Amazon S3.

**Amazon S3** is a distributed storage system & AWS’s equivalent to HDFS.

By saving our input data source into S3, each spark node deployed on the EMR cluster can read the input data source from S3.

**Source Code:**

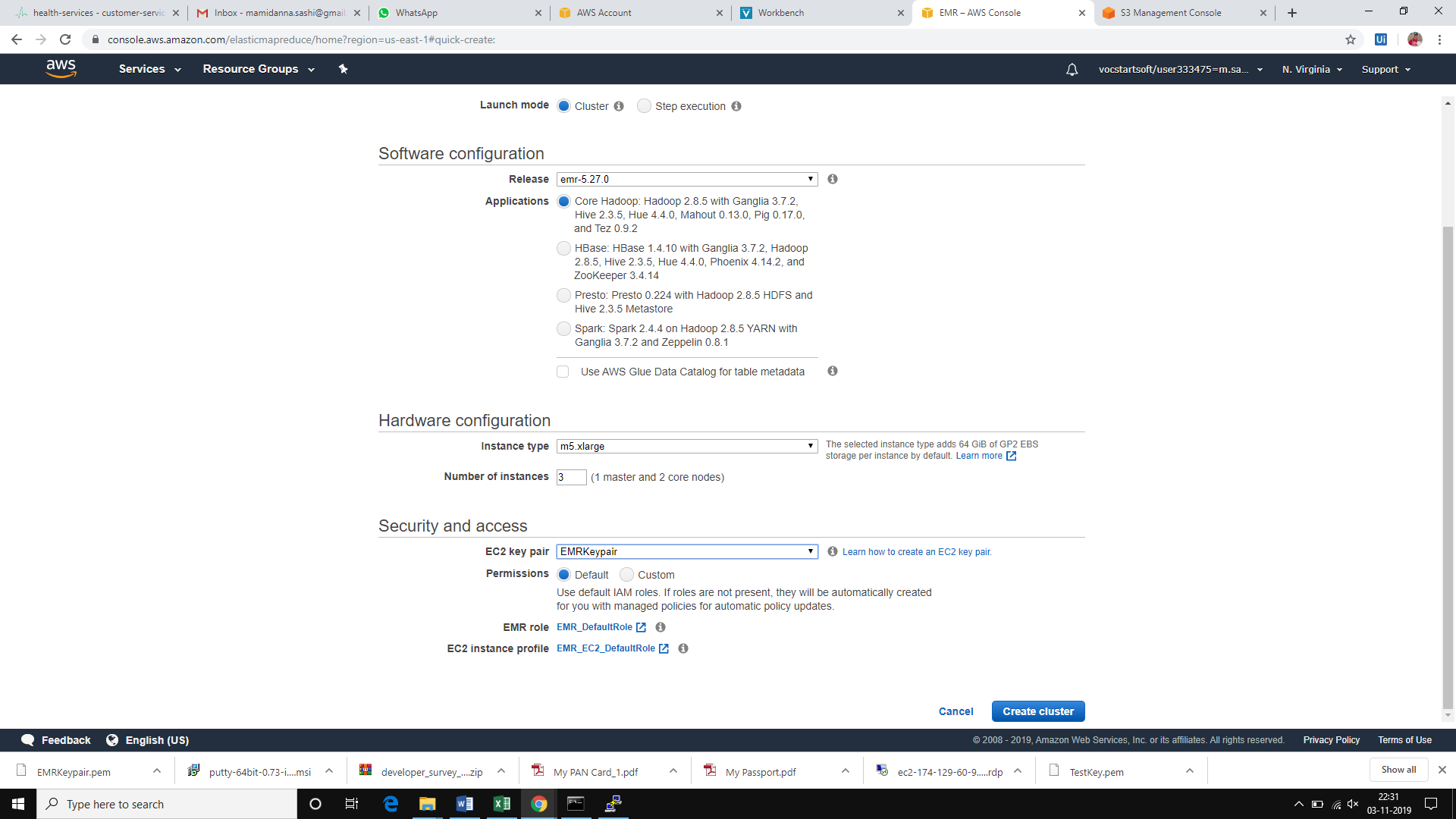
1. Login to Amazon Educate account and navigate to the AWS dashboard.
2. Click on Amazon EMR service



1. Click on create an EMR Cluster.
2. Give a name and select the type of applications that run on the cluster.
3. Select “Spark: Spark 2.4.4 on Hadoop 2.8.5 YARN with Ganglia 3.7.2 and Zeppelin 0.8.1”
4. Select the hardware configuration of the instance to be m5.xlarge and create one master and a slave node for the experiment purpose
5. Create a EC2 key pair, if not there already, else choose the keypair and continue.
6. To create a EC2 keypair, head back to EC2 dashboard and select Key pairs, to create and download the EC2 keypair.

EC2 Keypair is necessary to login to the Master node, for submitting big data jobs onto the Spark environment.

1. Leave the rest of the security permissions at default options and click on create cluster



1. The cluster creation will take around 10-15 minutes and in the process will create one master and slave node AMI’s.
2. Meanwhile, navigate to the website stackoverflow.com/research to download a sample dataset for processing data. Download the 2016 Full Data Set
3. Create a S3 storage bucket for uploading the “2016StackOverflowSurveyResponses.csv”. Ensure that the filename doesn’t have spaces.
4. Download the GIT repository “<https://github.com/jleetutorial/sparkTutorial>”
5. Open the source code from sparkSQL file “StackOverFlowSurvey.java”
6. When we run the Apache Spark application on the Amazon EMR cluster, Spark cluster will know how many nodes are available and where is the Master machine. So the cluster will provide a sensible setting at runtime.
7. The source code has already mentioned the settings for Master node, which will override the runtime configuration. But since we are running the application on a Local mode, Spark application will only run on one machine.
8. Remove the “Master option” when creating the Spark Session from the source code.

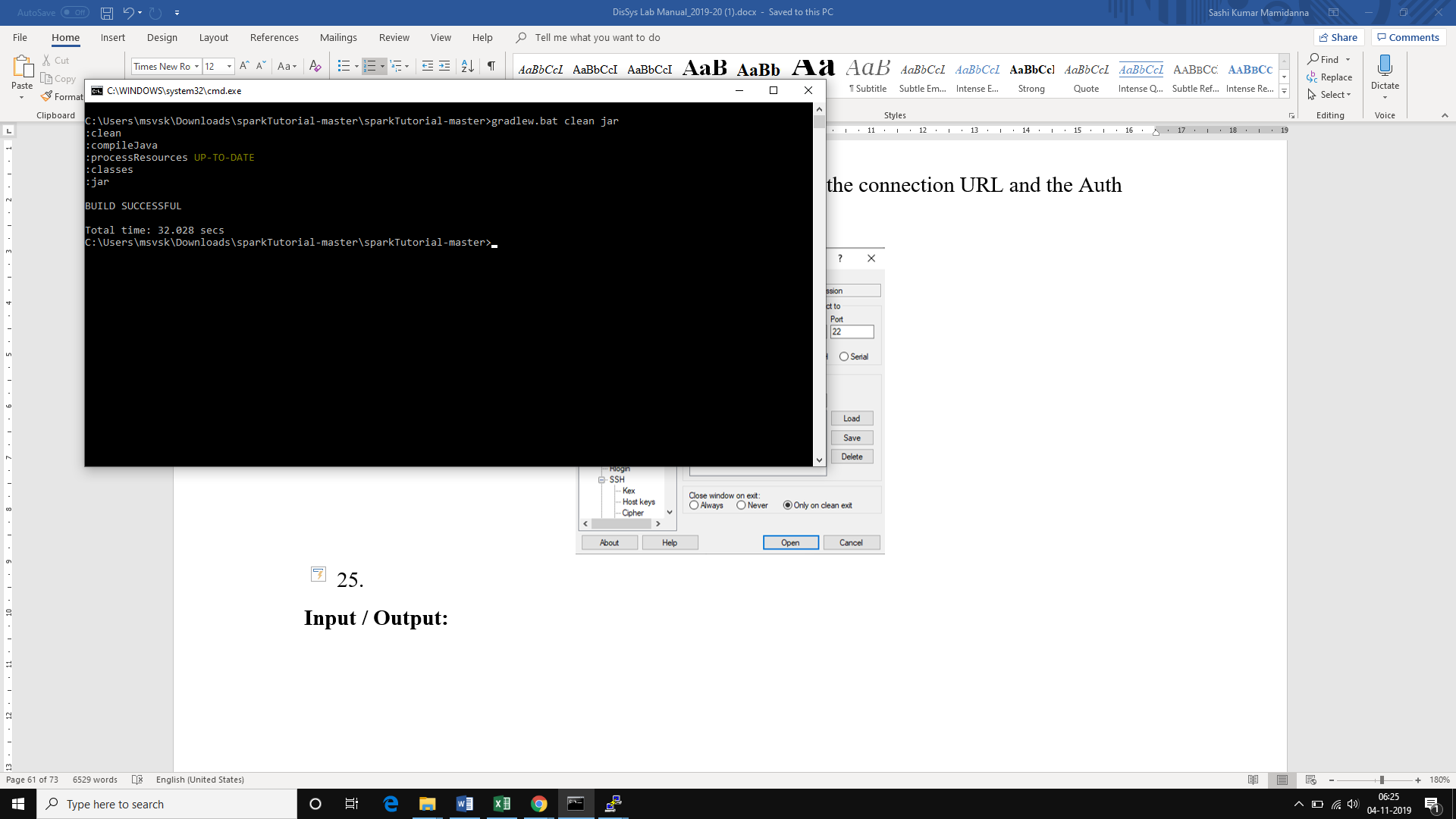
SparkSession session = SparkSession.builder().appName("StackOverFlowSurvey").getOrCreate();

1. Update the name of the S3 bucket in the source code by using the name of the path as “s3n://bucket\_name/input\_file.csv”

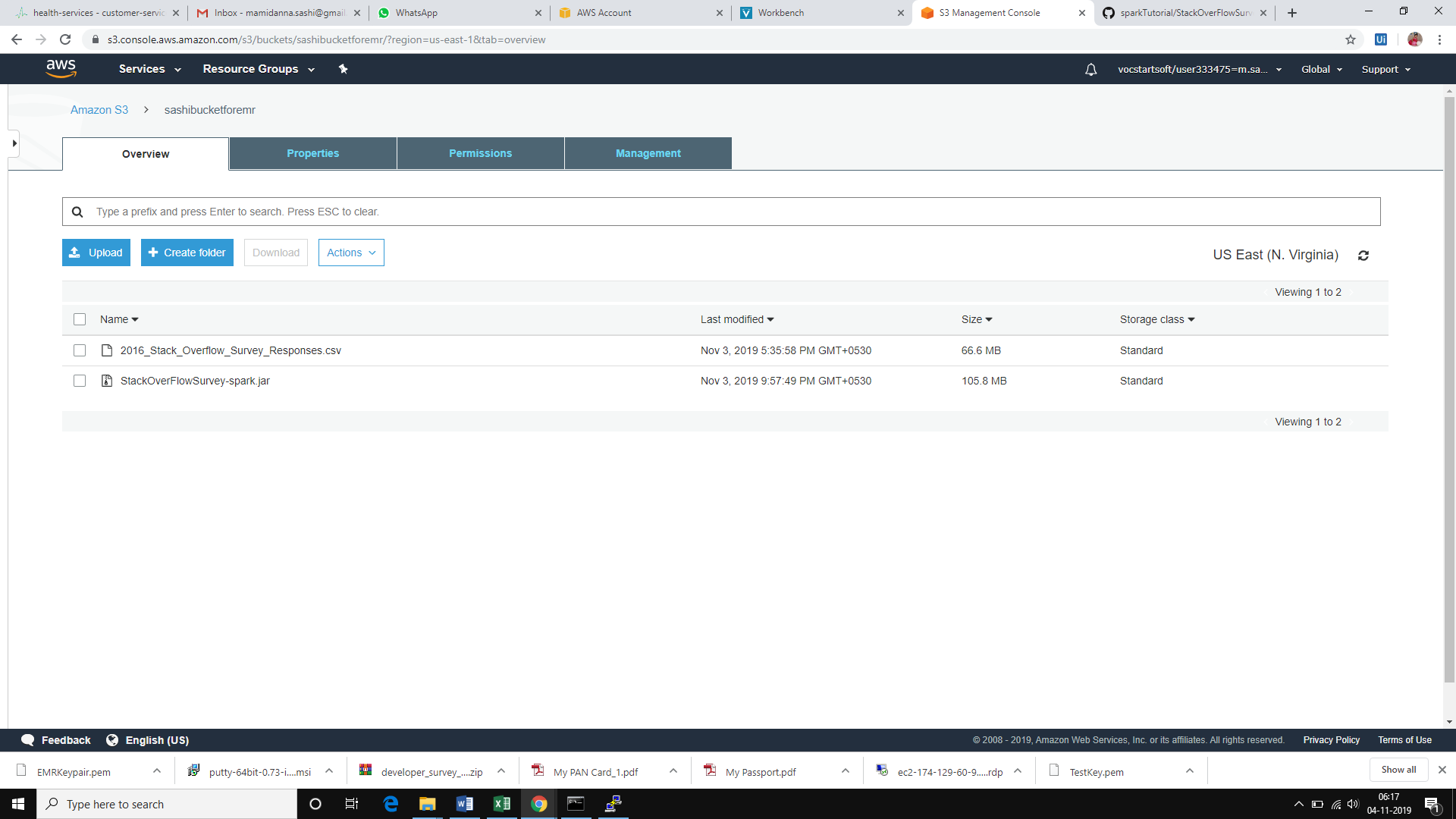
Dataset<Row> responses = dataFrameReader.option("header","true").csv("*s3n://sashibucketforemr/2016\_Stack\_Overflow\_Survey\_Responses.csv*");

1. Save the file and create the jar file out of the source code using Gradle Build batch file.
2. Open command prompt and navigate to the project folder “sparkTutorial-Master”
3. Type the below command to create a jar file in the ./build/libs folder

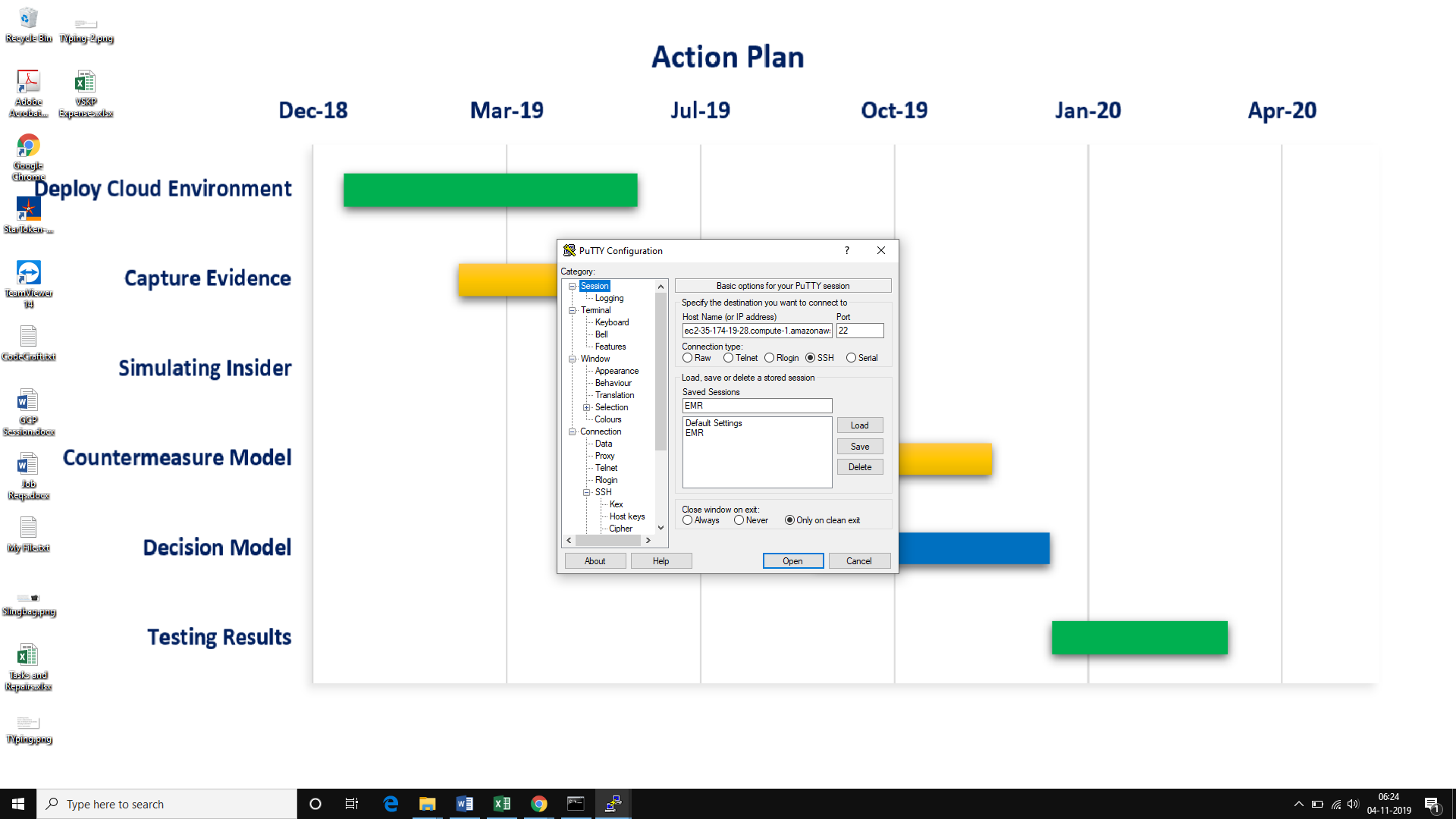
gradlew.bat clear jar



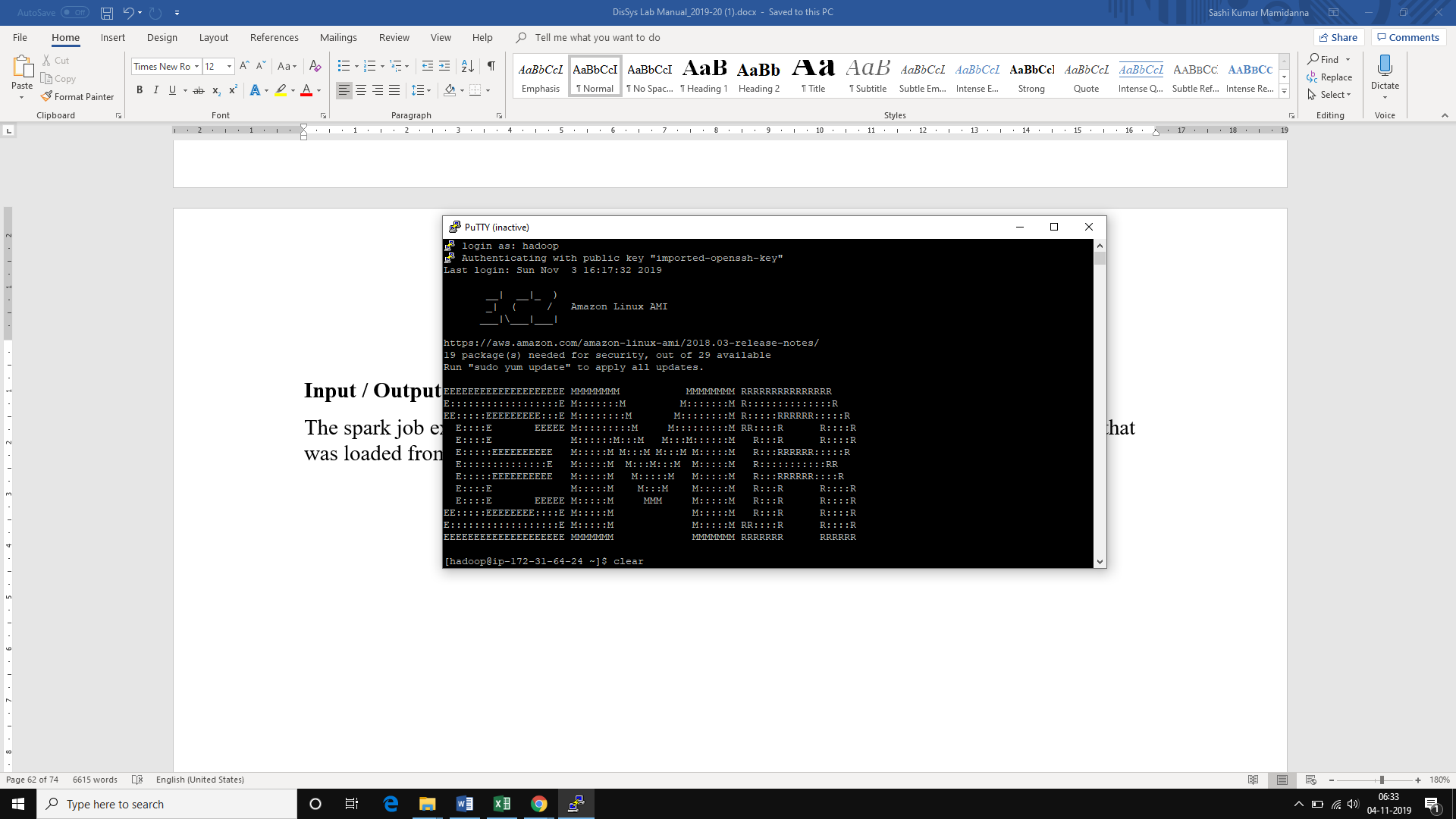
1. Now upload this jar file into the S3 bucket that was created to upload the input CSV.



1. Login to the Spark Cluster using SSH. Use PuttyGen to convert the .pew keypair into putty compatible key (.ppk)
2. Create a session to the Spark Cluster on putty using the connection URL and the Auth key on the session

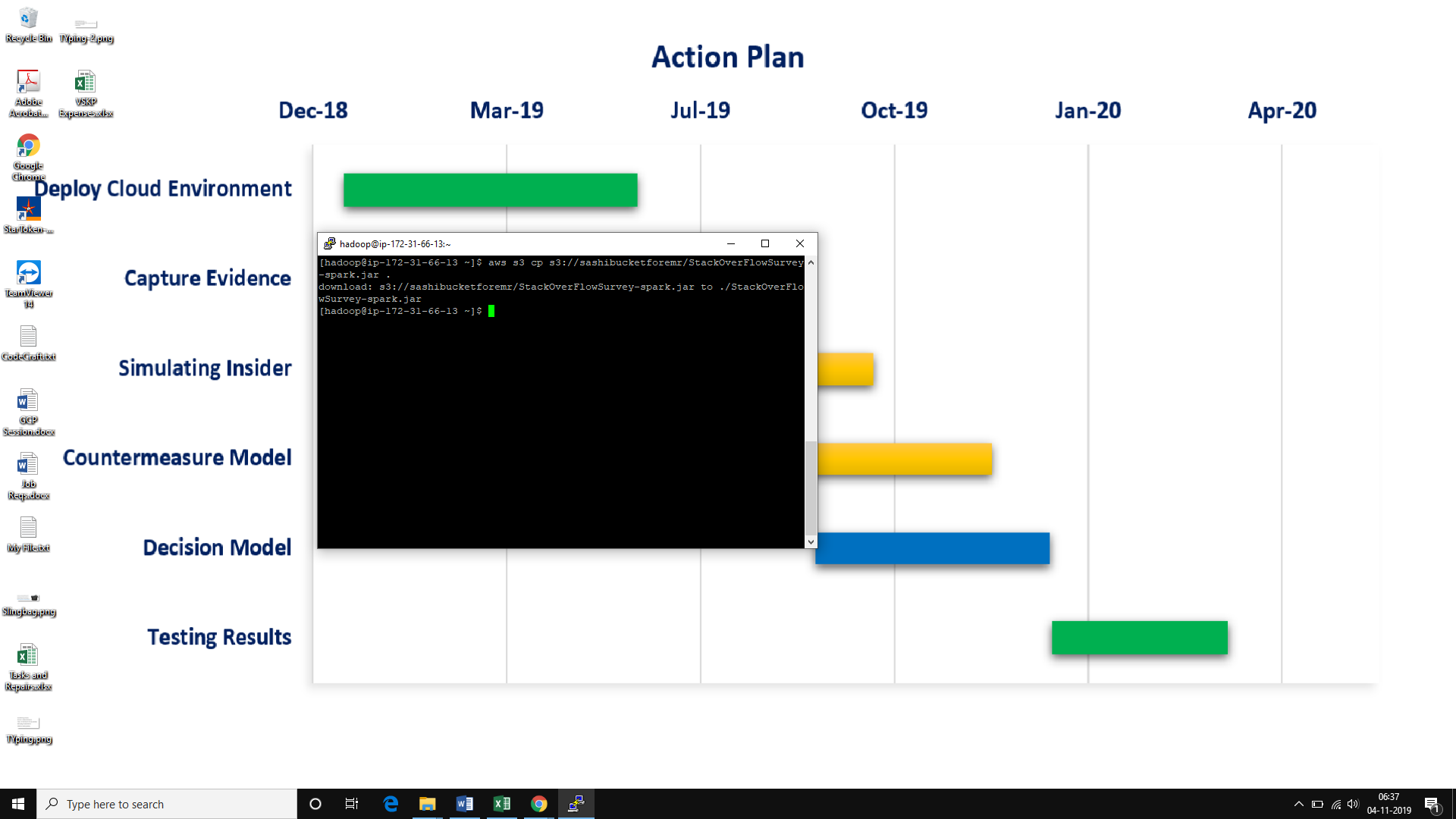


1. Login as the default user “Hadoop”, and you have now entered the EMR cluster console.



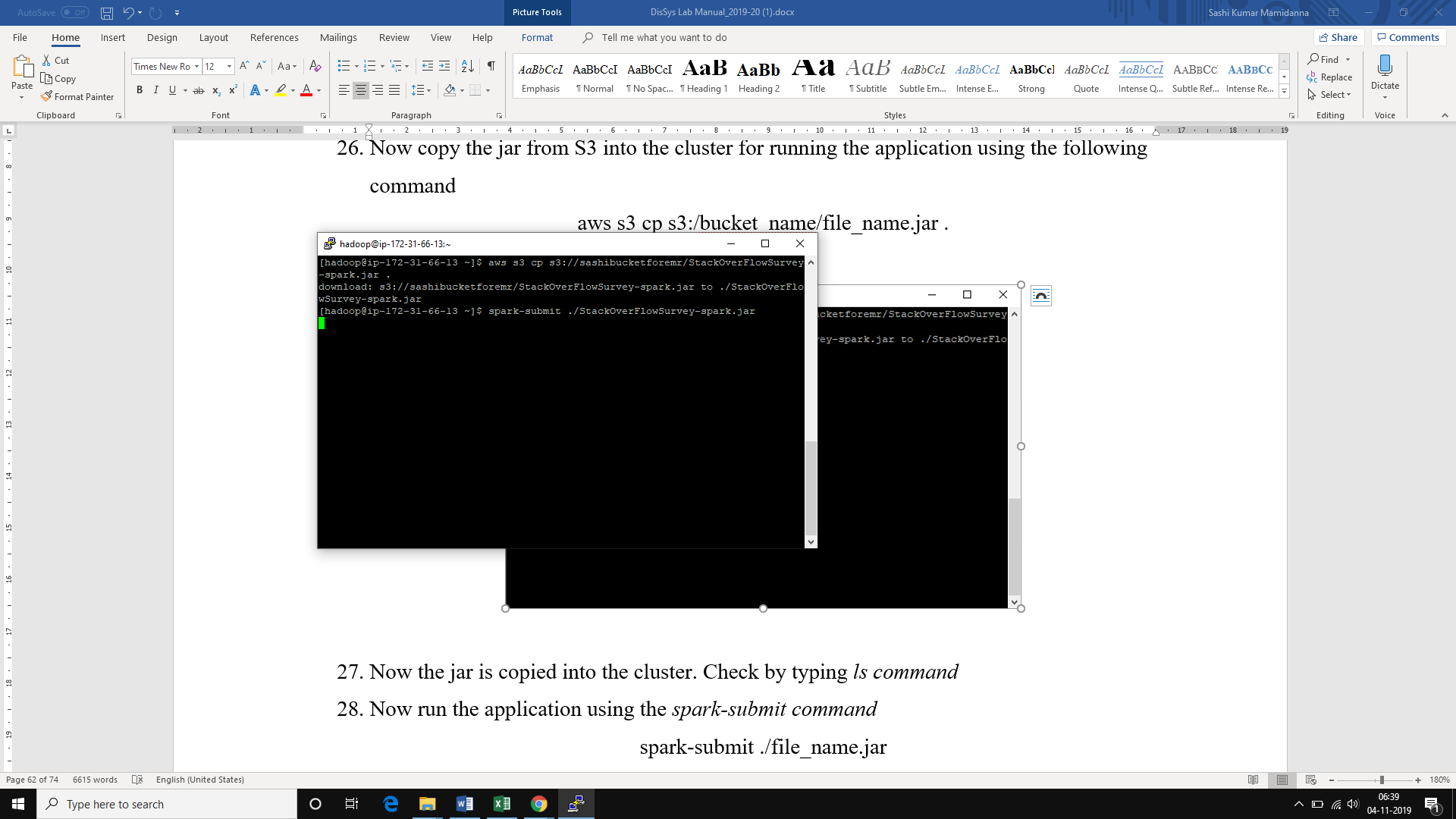
1. Now copy the jar from S3 into the cluster for running the application using the following command

aws s3 cp s3:/bucket\_name/file\_name.jar .



1. Now the jar is copied into the cluster. Check by typing *ls command*
2. Now run the application using the *spark-submit command*

spark-submit ./file\_name.jar



**Input / Output:**

The spark job executes on the EMR cluster and generates all the reports from the input file that was loaded from Amazon S3

