

Automated Setup & Installation Guide for Hadoop Single Node Cluster Environment (Pseudo Distributed mode) using light-weight script with Spark/Cassandra/MongoDB

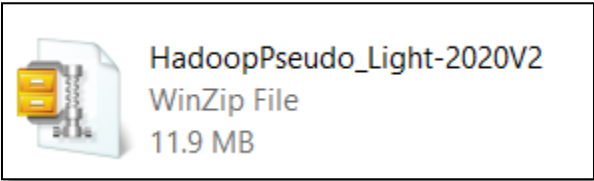
Version :- 2020V2

**Developed & Tested
by
RAJU CHAL
LKM , Accenture- ATCI**

Context :-

We will be using automated script for installation & configurations of “**Hadoop/Spark Single Node Cluster**” on Laptop /Desktop using light-weight script shared with you .

Script:-



File Name :- HadoopPseudo_Light-2020V2.zip

Contents of script :-

Hadoop-vagrant-lightweight > HadoopPseudo_Light-2020V2					
Name	Status	Date modified	Type	Size	
.vagrant	✓	02-May-20 03:33 PM	File folder		
dataset	✓	12-Jul-20 11:32 AM	File folder		
shared_folder	✓	18-May-17 11:41 AM	File folder		
bootstrap-mn	✓	28-Jul-20 09:23 AM	Shell Script	14 KB	
cmd-list	✓	07-Apr-20 02:42 PM	Text Document	1 KB	
core-site	✓	24-Mar-20 04:56 PM	XML Document	2 KB	
hbase-env	✓	25-Apr-20 05:45 PM	Shell Script	8 KB	
hbase-site	✓	14-Mar-20 07:53 PM	XML Document	3 KB	
hdfs-site	✓	24-Mar-20 04:56 PM	XML Document	2 KB	
hive-config	✓	10-Jun-18 03:46 PM	Shell Script	2 KB	
hive-env	✓	02-May-20 03:26 PM	Shell Script	3 KB	
hive-site	✓	30-Jul-18 02:59 PM	XML Document	3 KB	
hosts	✓	03-Apr-20 08:29 AM	File	1 KB	
mapred-site	✓	09-Apr-16 01:42 AM	XML Document	1 KB	
masters	✓	31-May-19 01:06 PM	File	1 KB	
my.cnf	✓	15-Apr-20 10:07 AM	CNF File	4 KB	
Readme	✓	09-Apr-16 05:55 PM	Text Document	3 KB	
regionserver	✓	14-Mar-20 06:38 PM	File	1 KB	
Required-Software	✓	06-Apr-20 05:29 PM	Text Document	1 KB	
setup	✓	07-Apr-20 02:59 PM	Windows Comma...	1 KB	
slaves	✓	03-Apr-20 08:31 AM	File	1 KB	
spark-defaults.conf	✓	18-May-17 10:35 AM	CONF File	2 KB	
spark-env	✓	18-May-17 10:34 AM	Shell Script	4 KB	
Vagrantfile	✓	07-Apr-20 04:01 PM	File	1 KB	
yarn-site	✓	31-May-19 12:36 PM	XML Document	2 KB	

Software with version to be installed

<u>Software</u>	<u>Version</u>
Hadoop	2.9.2
Spark	2.4.5
Sbt	1.2.0
Hive	2.3.7
Pig	0.16.0
Cassandra	3.0.20
MongoDB	4.0.9
Sqoop	1.4.7
HBase	1.6.0
Kafka	2.4.1
Scala	2.12.2
JDK	8u131
MySQL	5.7
Python	3.6

Download & Install the pre-requisite software

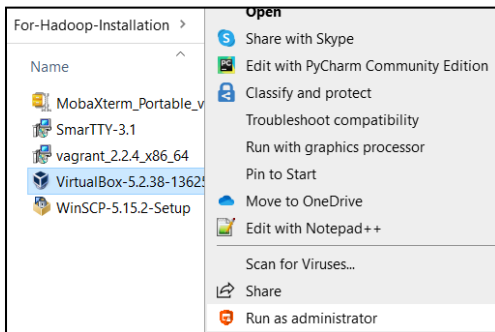
Pre-requisite:-

- During entire installation procedure your Laptop/Desktop should be connected with Internet.
- Minimum RAM required:- 8 GB

1) Download and Install Oracle Virtual Box

<https://download.virtualbox.org/virtualbox/5.2.38/VirtualBox-5.2.38-136252-Win.exe>

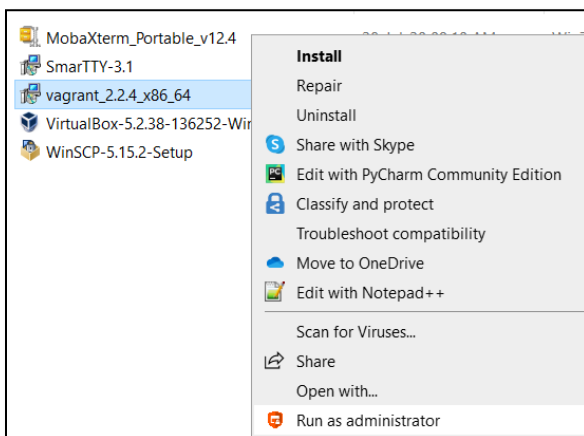
Right click on downloaded software → click on “Run as administrator”



2) Download and Install Vagrant version 2.2.4

https://releases.hashicorp.com/vagrant/2.2.4/vagrant_2.2.4_x86_64.msi

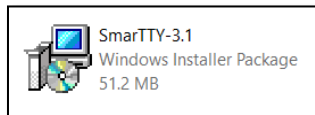
Right click on downloaded software → click on “Run as administrator”



After installation “RESTART” the system

3) Download SmarTTY

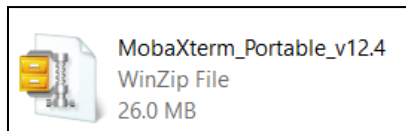
<http://sysprogs.com/getfile/409/SmarTTY-3.1.msi>



OR

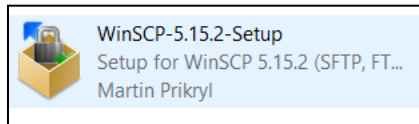
Download MobaXTerm

https://download.mobatek.net/2012020021813110/MobaXterm_Portable_v20.1.zip



4) Download WinSCP

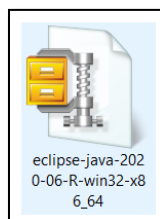
<https://winscp.net/eng/download.php>



5) Eclipse Download (OPTIONAL)

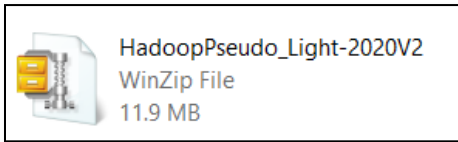
https://ftp.yz.yamagata-u.ac.jp/pub/eclipse//technology/epp/downloads/release/2020-06/R/eclipse-java-2020-06-R-win32-x86_64.zip

unzip and run it

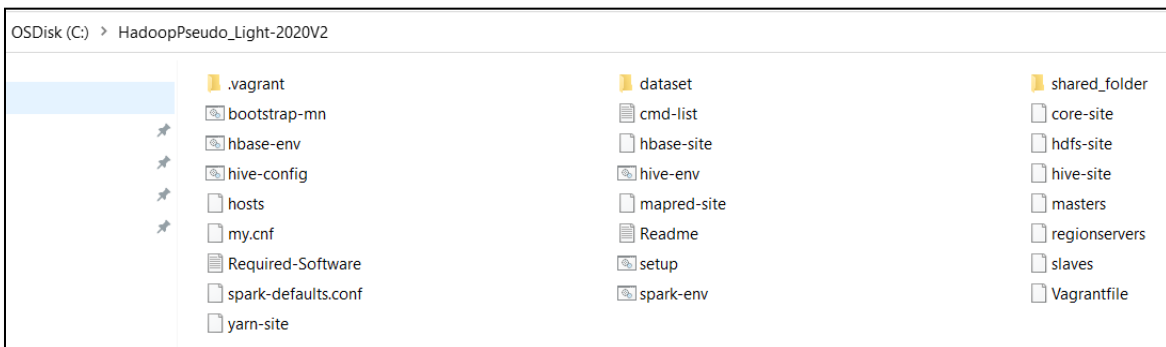
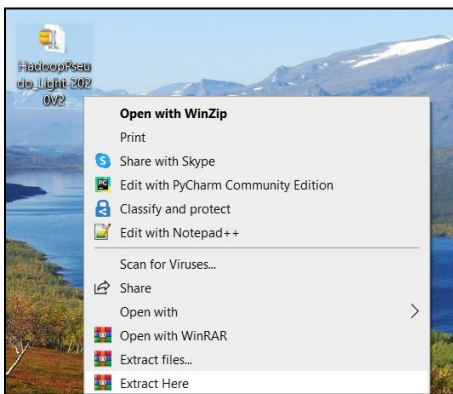


Installation Process

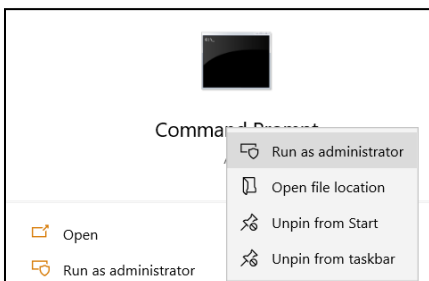
1. Download the shared zip file - **HadoopPseudo_Light-2020V2.zip**



2. Unzip it → Right click on the ZIP file → Click on “Extract Here” → copy the extracted root folder to C-Drive



3. Open **command prompt** of Windows in **Administrator** mode



4. Change the directory to the extracted folder **HadoopPseudo_Light-2020V2** → run “**setup.cmd**” command

```
C:\Users\raju.chal>cd c:\
```

```
c:\>cd HadoopPseudo_Light-2020V2
```

```
c:\HadoopPseudo_Light-2020V2>setup.cmd
```

```
Administrator: C:\Windows\system32\cmd.exe
Microsoft Windows [Version 10.0.18363.959]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\raju.chal>cd c:\

c:\>cd HadoopPseudo_Light-2020V2

c:\HadoopPseudo_Light-2020V2>setup.cmd
```

Wait till you get back the Command Prompt [**c:\HadoopPseudo_Light-2020V2>**]

Depending on the bandwidth total installation may take 45 mins to 1 hr time

```
c:\HadoopPseudo_Light-2020V2>setup.cmd

c:\HadoopPseudo_Light-2020V2>vagrant box add ubuntu/trusty64 --insecure
==> vagrant: A new version of Vagrant is available: 2.2.9 (installed version: 2.2.4)!
==> vagrant: To upgrade visit: https://www.vagrantup.com/downloads.html

==> box: Loading metadata for box 'ubuntu/trusty64'
    box: URL: https://vagrantcloud.com/ubuntu/trusty64
==> box: Adding box 'ubuntu/trusty64' (v20190514.0.0) for provider: virtualbox
The box you're attempting to add already exists. Remove it before
adding it again or add it with the '--force' flag.

Name: ubuntu/trusty64
Provider: virtualbox
Version: 20190514.0.0

c:\HadoopPseudo_Light-2020V2>vagrant up
```

```
c:\HadoopPseudo_Light-2020V2>vagrant up
Bringing machine 'Master' up with 'virtualbox' provider...
==> Master: Importing base box 'ubuntu/trusty64'...
==> Master: Matching MAC address for NAT networking...
==> Master: Checking if box 'ubuntu/trusty64' version '20190514.0.0' is up to date...
==> Master: Setting the name of the VM: HadoopPseudo_Light-2020V2_Master_1595913072027_30213
==> Master: Clearing any previously set forwarded ports...
==> Master: Clearing any previously set network interfaces...
==> Master: Preparing network interfaces based on configuration...
    Master: Adapter 1: nat
    Master: Adapter 2: hostonly
==> Master: Forwarding ports...
    Master: 22 (guest) => 2222 (host) (adapter 1)
==> Master: Running 'pre-boot' VM customizations...
==> Master: Booting VM...
```

```
Master: 20/07/28 05:23:29 INFO util.ShutdownHookManager: Shutdown hook called
Master: 20/07/28 05:23:29 INFO util.ShutdownHookManager: Deleting directory /tmp/spark-14b689e6-c89b-44fd-964c-3381d1ab81c1
Master: 20/07/28 05:23:29 INFO util.ShutdownHookManager: Deleting directory /tmp/spark-bff40e0f-0fec-43df-88f7-c47f00367759
Master: Your environment is ready

c:\HadoopPseudo_Light-2020V2>
```

5. After getting back the Command Prompt type "**vmagrant ssh**" to login to Linux Box

C:\HadoopPseudo_Light-2020v2>vmagrant ssh

```
c:\HadoopPseudo_Light-2020V2>vmagrant ssh
Welcome to Ubuntu 14.04.6 LTS (GNU/Linux 3.13.0-170-generic x86_64)

* Documentation:  https://help.ubuntu.com/

System information as of Tue Jul 28 05:22:44 UTC 2020

System load:  1.07               Processes:    99
Usage of /:   11.3% of 39.34GB    Users logged in: 0
Memory usage: 35%               IP address for eth0: 10.0.2.15
Swap usage:   0%                 IP address for eth1: 192.168.56.70

Graph this data and manage this system at:
https://landscape.canonical.com/

New release '16.04.6 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

vmagrant@master:~$ ls bigdata/
```

vmagrant@master:~\$ jps

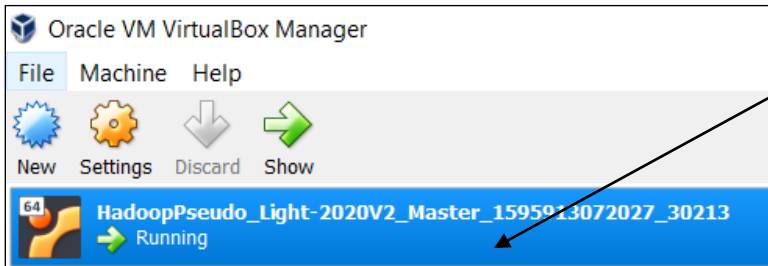
```
11538 Jps
9716 DataNode
9942 SecondaryNameNode
10520 Master
9528 NameNode
10107 ResourceManager
10446 NodeManager
10750 Worker
```

vmagrant@master:~\$

```
vmagrant@master:~$ exit
logout
Connection to 127.0.0.1 closed.

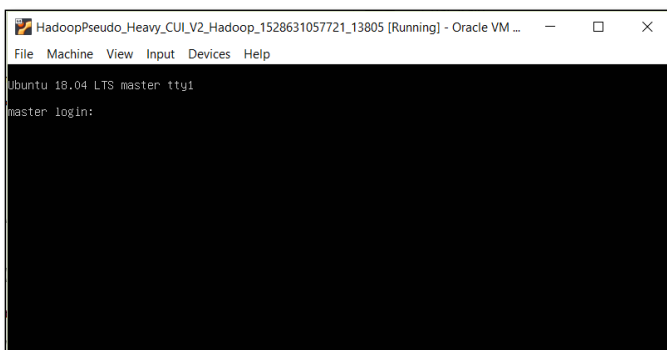
c:\HadoopPseudo_Light-2020V2>
```


6. Open the **Oracle VirtualBox** that you have already installed, you will observe one Linux machine is running as shown below

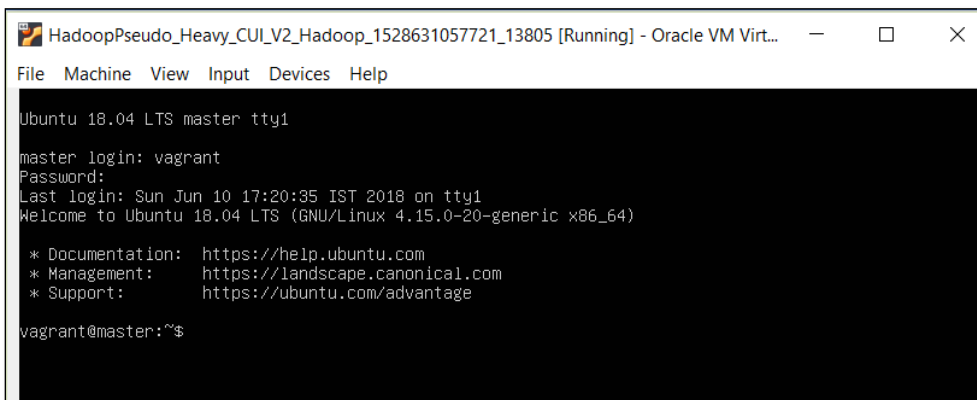


Note :- If it is not able to start, then - > You need to enable Virtualization on your laptop/desktop to create a virtualized environment on your desktop. The steps for the same depend on your laptop/desktop model. You should take help from Tech Support

6. Select the Linux box and click on the **Show** button in the toolbar, you will be getting the following screen



Login name :- vagrant
Password :- vagrant

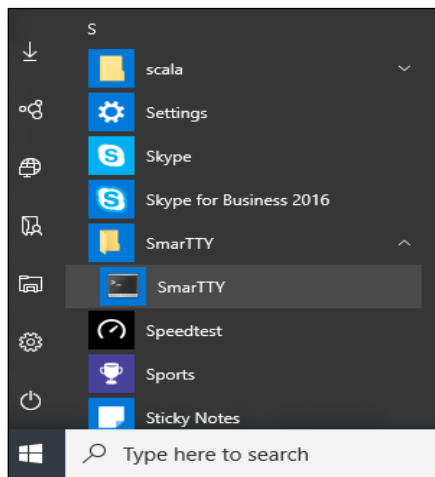


Connecting SmartTTY with the Linux Node

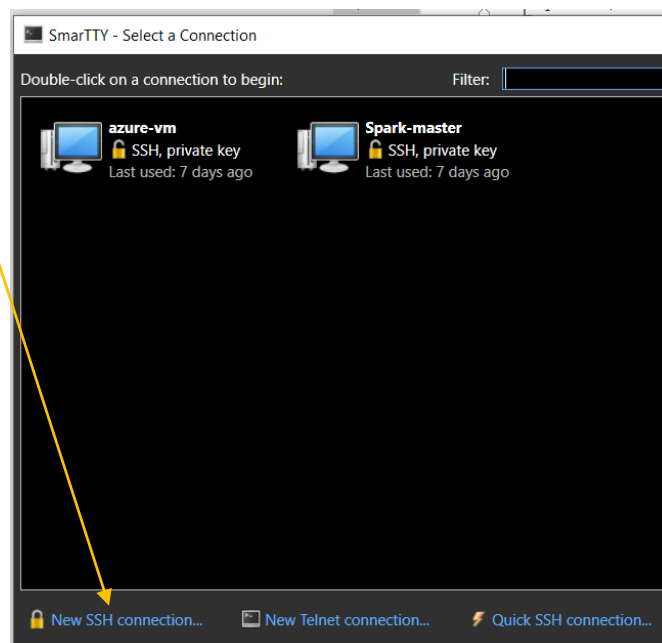
1. Install SmartTTY.

- a. SmartTTY is a free multi-tabbed SSH client that supports copying files and directories with SCP on-the-fly and editing files in-place.

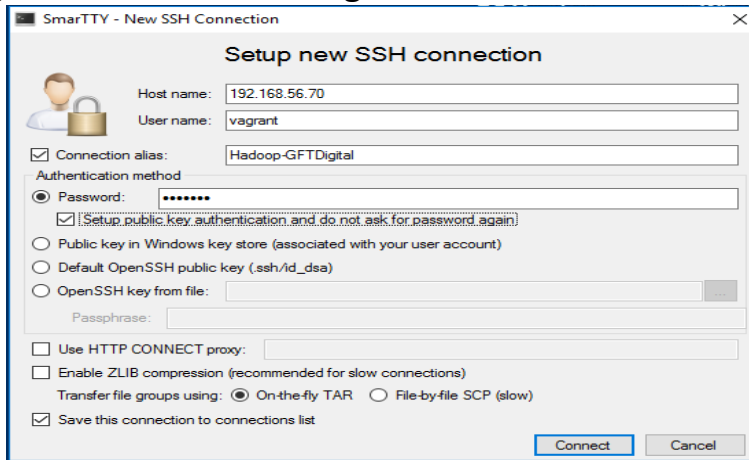
2. To Connect SmartTTY with Hadoop Node , click on SmartTTY menu ,



3. Click on “New SSH Connection “



4. Fill the dialog box with the following information as shown below

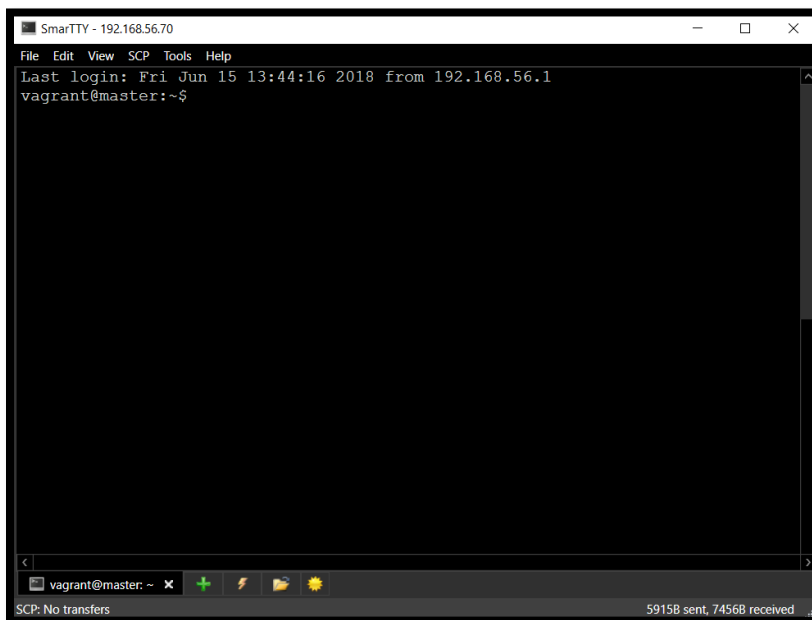


Host Name :- 192.168.56.70

User Name :- vagrant

Password :- vagrant

Click on “Connect”



You can open Multiple TAB connected with the Linux Node.

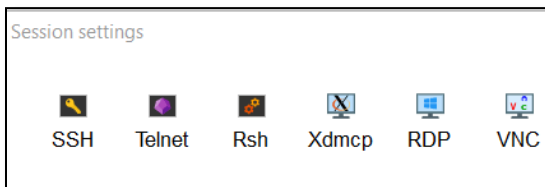
Now your Hadoop/Spark environment is ready .

Connecting MobaXTerm with the Linux Node

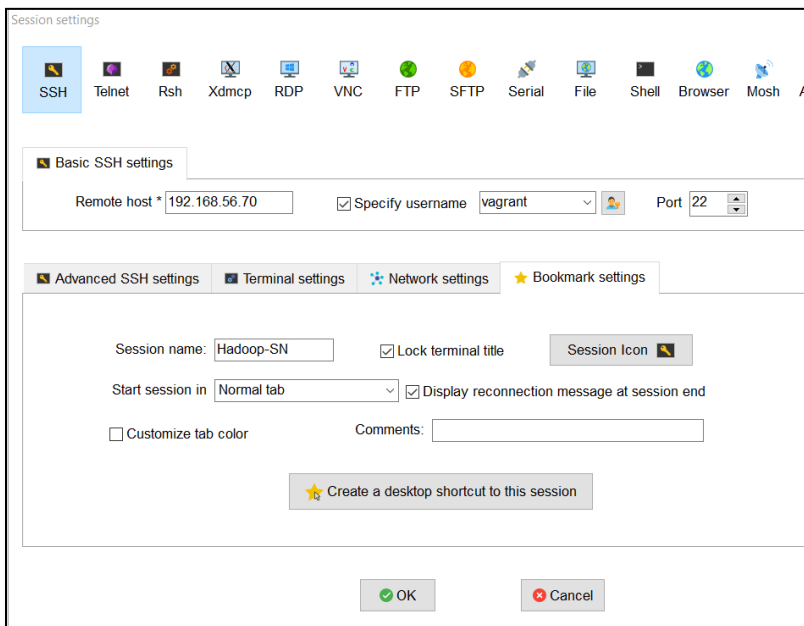
1. Open **MobaXTerm**



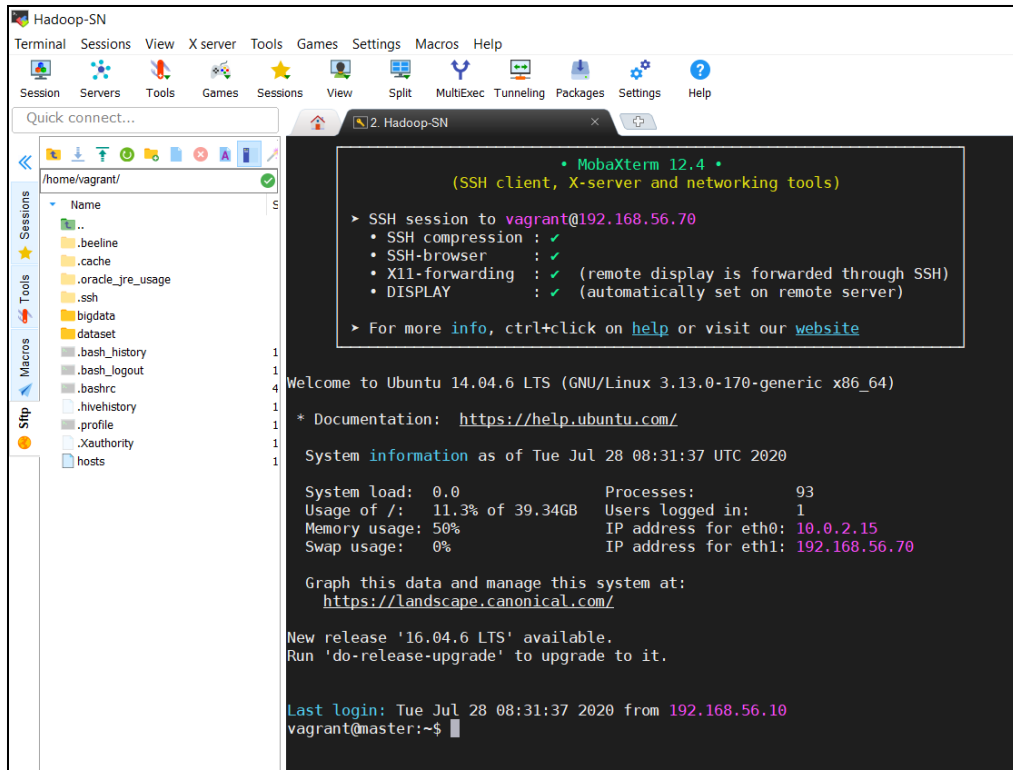
Tool Bar → Click on “SSH” button → Click on “SSH” button



2. Fill the dialog box with the following information as shown below



Click on “OK”



Now your Hadoop/Spark environment is ready .

Check Hive Service

vagrant@master:~\$ hive

```
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/home/vagrant/bigdata/hive/lib/log4j-slf4j-impl-2.6.2.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/home/vagrant/bigdata/hadoop/share/hadoop/common/lib/slf4j-log4j12-1.7.25.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.apache.logging.slf4j.Log4jLoggerFactory]

Logging initialized using configuration in jar:file:/home/vagrant/bigdata/hive/lib/hive-common-2.3.7.jar!/hive-log4j2.properties Async: true
Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider using a different execution engine (i.e. spark, tez) or using Hive 1.X releases.
hive> show databases;
OK
default
Time taken: 5.43 seconds, Fetched: 1 row(s)
hive> _
```

Check Pig Service

vagrant@master:~\$ pig

```
hadoop file system at: hdfs://master:9000
2020-07-28 08:50:02,940 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instead, use fs.defaultFS
2020-07-28 08:50:02,981 [main] INFO org.apache.pig.PigServer - Pig Script ID for the session: PIG-default-9c7fa0ffc80b-42a9-8e9e-b79daf92c07d
2020-07-28 08:50:02,981 [main] WARN org.apache.pig.PigServer - ATS is disabled since yarn.timeline-service.enabled set to false
grunt> _
```

Check Spark Service

vagrant@master:~\$ spark-shell --master spark://master:7077

```
vagrant@master:~$ spark-shell --master spark://master:7077
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
Spark context Web UI available at http://master:4040
Spark context available as 'sc' (master = spark://master:7077, app id = app-20200728085743-0001).
Spark session available as 'spark'.
Welcome to

  ____
 /  _ \
/_/_/ \_/_/  version 2.4.5

Using Scala version 2.11.12 (Java HotSpot(TM) 64-Bit Server VM, Java 1.8.0_131)
Type in expressions to have them evaluated.
Type :help for more information.

scala> _
```


Shutdown the Node

If you want to shutdown your node completely ,

please type the following command in the **\$** prompt (Either in Putty or in the Linux node directly).

```
$ sudo init 0
```

Your node will be shutdown.

Next time when you want to start it ,

- you have to open it from the **Oracle Virtual Box**.
- Select the node from the Oracle Virtual Box, click on the “**Start**” button .
- After the node has been started in the Virtual Box, connect it from windows using **Putty** .

Start the services again

For Hadoop (Mandatory)

```
$ start-dfs.sh  
$ start-yarn.sh
```

For Hadoop (Optional)

```
$ mr-jobhistory-daemon.sh start historyserver
```

For Spark (Mandatory)

```
$ start-master.sh  
$ start-slaves.sh
```

Check the services :-

```
$ jps
```

Check HBase Services

To start the service

```
$ start-hbase.sh
```

```
agrant@master:~$ jps
```

```
4720 HRegionServer
1633 NodeManager
1333 SecondaryNameNode
1141 DataNode
4791 Jps
2825 ApplicationHistoryServer
4521 HQuorumPeer
1516 ResourceManager
4575 HMaster
1023 NameNode
```

Web interface

<http://192.168.56.70:16010>

<http://192.168.56.70:16030>

```
agrant@master:~$ hbase shell
```

```
HBase Shell; enter 'help<RETURN>' for list of supported commands.
Type "exit<RETURN>" to leave the HBase Shell
Version 1.2.6, rUnknown, Mon May 29 02:25:32 CDT 2017

hbase(main):001:0> █
```

Test HBase

```
hbase(main):001:0> create 'test', 'cf'
```

0 row(s) in 6.2640 seconds

=> Hbase::Table - test

```
hbase(main):002:0> list
```

TABLE
test

1 row(s) in 0.4770 seconds

=> ["test"]

```
hbase(main):003:0> put 'test', 'row1', 'cf:a', 'value1'
```

0 row(s) in 1.6720 seconds

```
hbase(main):004:0> put 'test', 'row2', 'cf:b', 'value2'
```

0 row(s) in 0.0560 seconds

```
hbase(main):005:0> put 'test', 'row3', 'cf:c', 'value3'
```

0 row(s) in 0.2260 seconds

```
hbase(main):006:0> scan 'test'
```

ROW	COLUMN+CELL
row1	column=cf:a, timestamp=1529056467058, value=value1
row2	column=cf:b, timestamp=1529056476408, value=value2
row3	column=cf:c, timestamp=1529056484435, value=value3

3 row(s) in 0.0790 seconds

To Stop the service

```
$ stop-hbase.sh
```

Check MySQL Services

```
vagrant@master:~$ mysql -u root -p
```

Enter password: **root**

```
mysql> show databases;
```

```
+-----+
| Database          |
+-----+
| information_schema |
| metastore_db      |
| mysql              |
| performance_schema |
+-----+
4 rows in set (0.24 sec)
```

Check Cassandra Services

Start Cassandra in the foreground by invoking

```
$ bin/cassandra -f
```

from the command line.

Press “Control-C” to stop Cassandra.

Start Cassandra in the background by invoking

```
$ bin/cassandra
```

from the command line.

To Stop Cassandra running in Background

Invoke

```
kill pid
```

or

```
pkill -f CassandraDaemon
```

to stop Cassandra, where **pid** is the Cassandra process id,

which you can find for example by invoking `pgrep -f CassandraDaemon`.

Verify that Cassandra is running

by invoking

```
bin/nodetool status
```

from the command line.

Configuration files are located in the **conf** sub-directory.

Due to this, it is necessary to either start Cassandra with root privileges or change **conf/cassandra.yaml**

CQLSH

cqlsh is a command line shell for interacting with Cassandra through CQL. It is shipped with every Cassandra package, and can be found in the **bin/** directory alongside the **cassandra** executable. It connects to the single node specified on the command line.

For example:

```
$ bin/cqlsh localhost
```

Connected to Test Cluster at localhost:9042.

[cqlsh 5.0.1 | Cassandra 3.8 | CQL spec 3.4.2 | Native protocol v4]

Use HELP for help.

```
cqlsh> SELECT cluster_name, listen_address FROM system.local;
```

```
cluster_name | listen_address
```

```
-----+-----
```

```
Test Cluster | 127.0.0.1
```

(1 rows)

```
cqlsh>
```

Check MongoDB Services

Start MongoDB server

```
$ mongod
```

```
2018-06-15T15:28:41.663+0530 I COMMAND [initandlisten] setting featureCompatibilityVersion to 3.6
2018-06-15T15:28:41.685+0530 I STORAGE [initandlisten] createCollection: local.startup_log with generated UUID: ee022a43-f237-4c10-bb71-d0094eb5c8ea
2018-06-15T15:28:41.699+0530 I FTDC [initandlisten] Initializing full-time diagnostic data capture with directory '/data/db/diagnostic.data'
2018-06-15T15:28:41.700+0530 I NETWORK [initandlisten] waiting for connections on port 27017
```

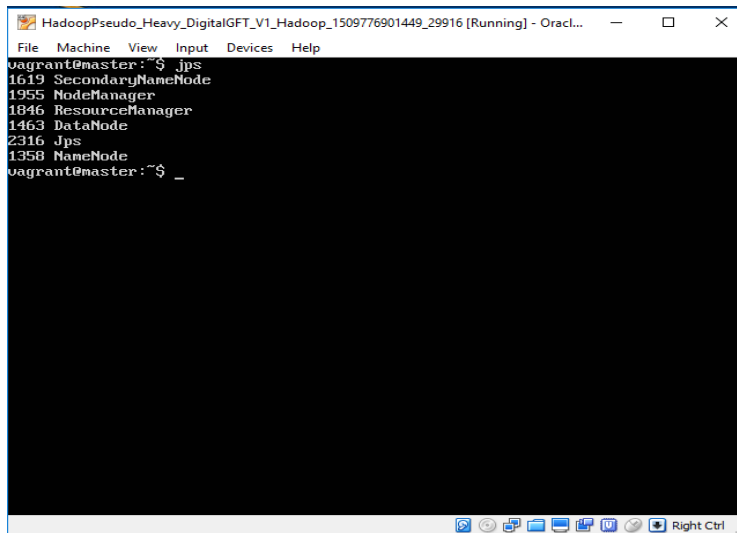
Start Mongo Shell in another TAB

```
$ mongo
```

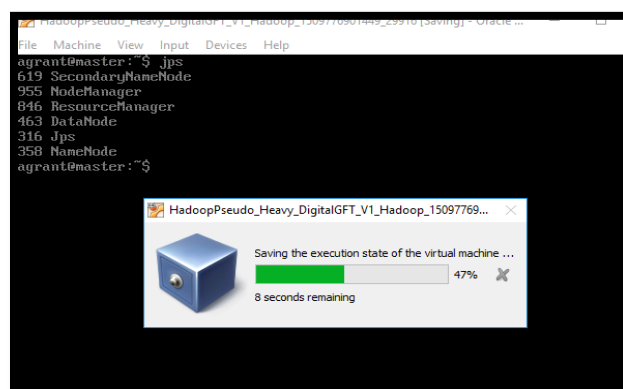
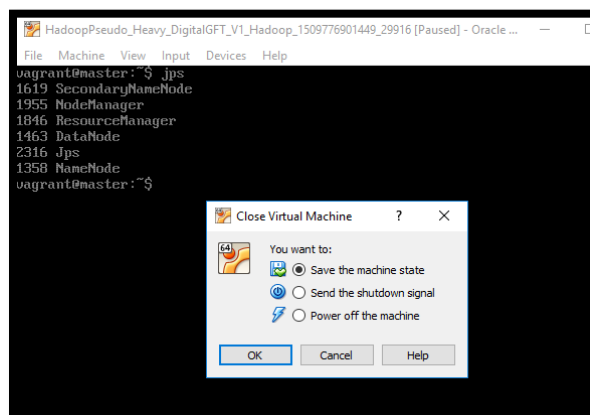
```
>
```

Suspend the Linux Node from Virtual Box

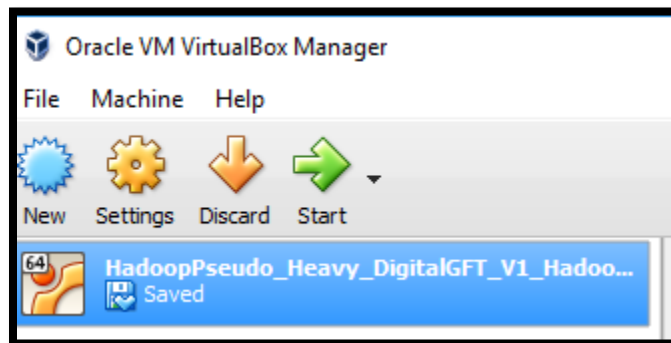
1. Click on the “close” button of the Linux Window opened in Virtual Box



2. It will open another dialog box asking about the operations of your choice , click on the choice “Save the machine state ” →Click on “OK”

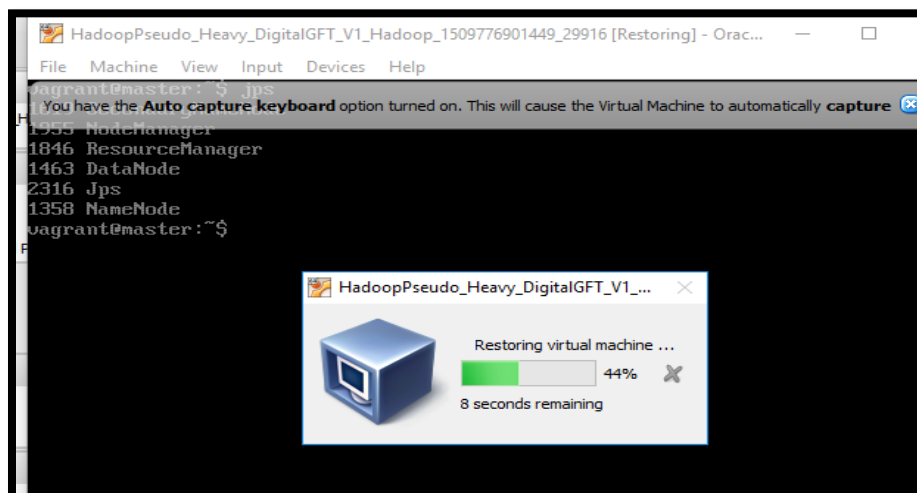
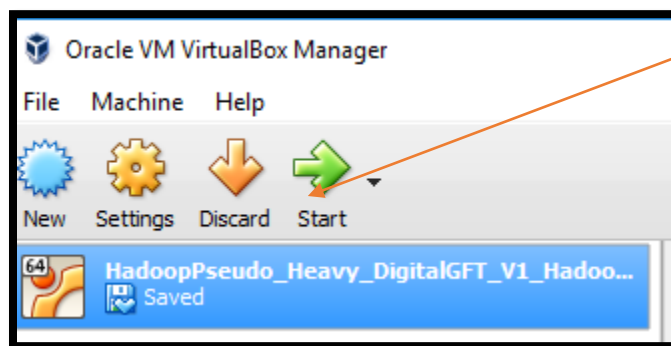


3. In the Virtual Box window the Linux node will be shown in “SAVED” mode .



To start the Linux node from “saved” state

Select the Linux Node in the Virtual Box window (shown in “**saved**” mode) → click on “**Start**” button



Check the “Hadoop Services” using “**jps**” command; if the services are not running , start the services using the following commands.

```

HadoopPseudo_Heavy_DigitalGFT_V1_Hadoop
File Machine View Input Devices Help
vagrant@master:~$ jps
1619 SecondaryNameNode
1955 NodeManager
1846 ResourceManager
1463 DataNode
2344 Jps
1358 NameNode
  
```

```

$ start-dfs.sh
$ start-yarn.sh
  
```

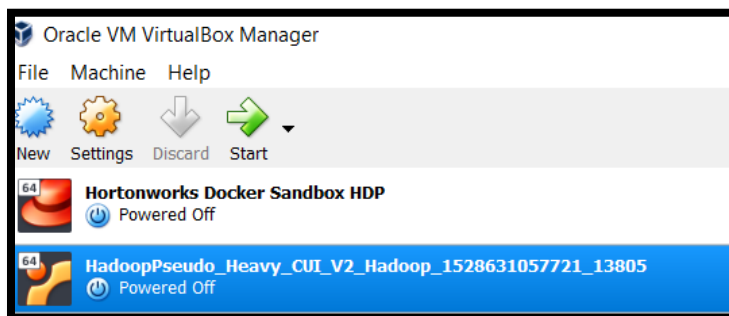
Shutdown the Node

To shutdown the Hadoop Node completely

Type the following command in the **\$** prompt (Either in Putty or in the Linux node directly).

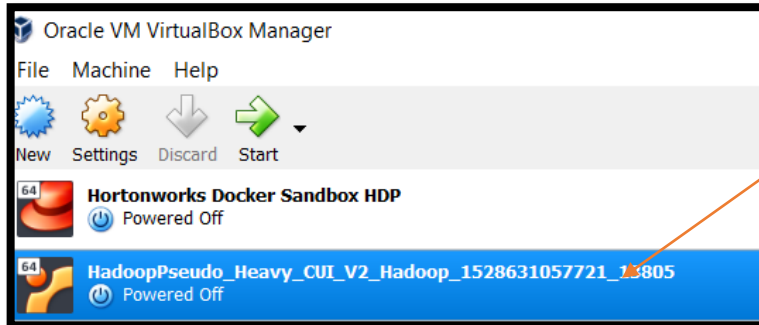
```
$ sudo init 0
```

The node will be shutdown and it will shown as “**Powered Off**” state in the Virtual Box Window.



To Start the Hadoop Node from “Powered Off” state

- Open the **Oracle Virtual Box**.
- Select the node from the Oracle Virtual Box, click on the “**Start**” button .
- After the node has been started in the Virtual Box, connect it from windows using **Putty** or **SmarTTY**.



Start the services using the following commands: -

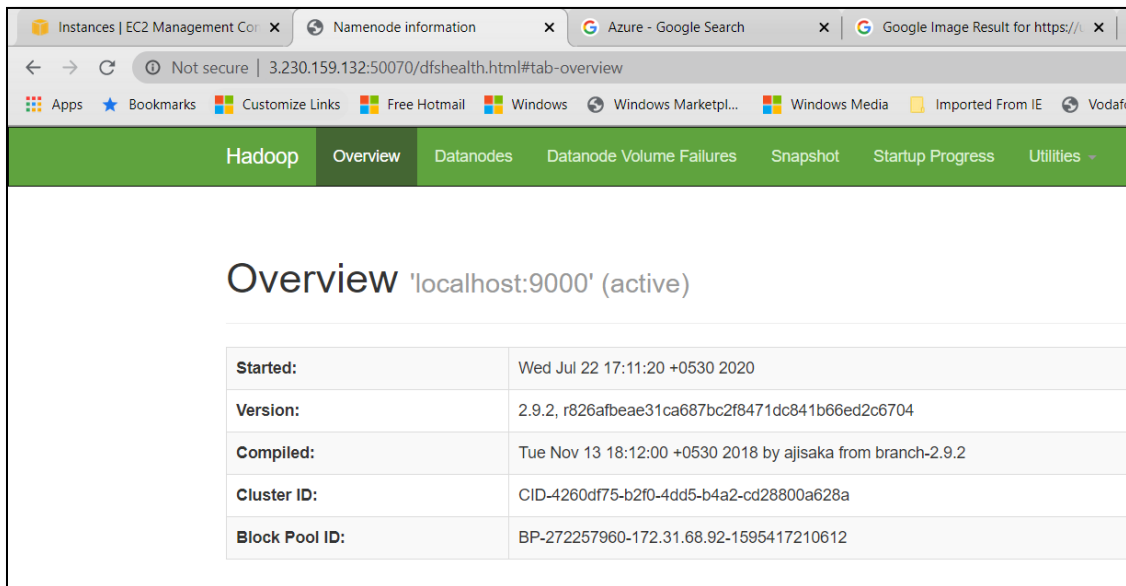
```
$ start-dfs.sh  
$ start-yarn.sh
```

Check the services :-

```
$ jps
```


Check Namenode web interface

<http://192.168.56.70:50070>

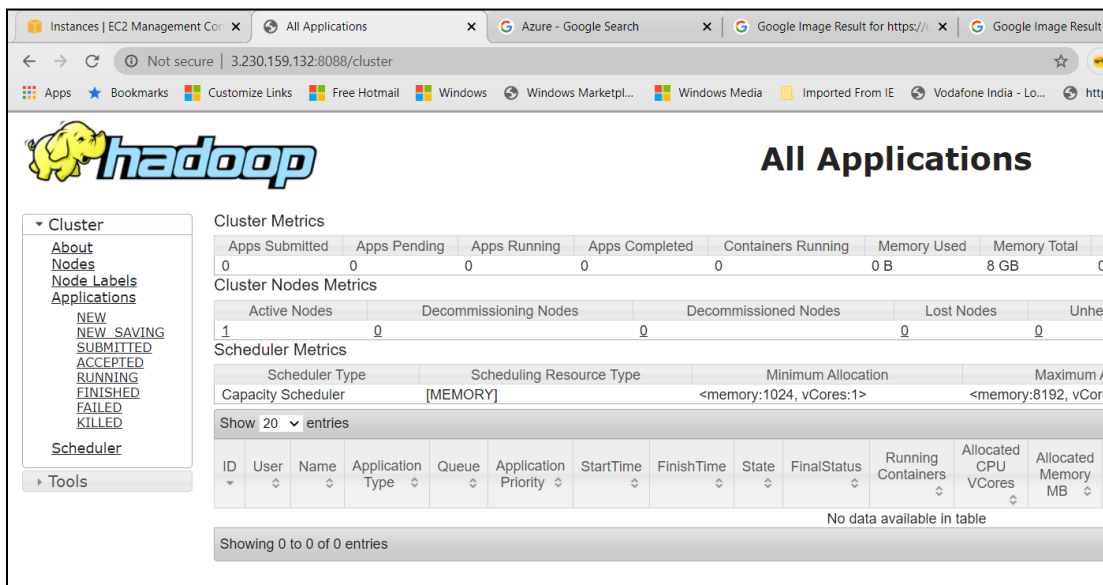


The screenshot shows the Hadoop Namenode web interface. The browser tabs include 'Instances | EC2 Management Console', 'Namenode information', 'Azure - Google Search', and 'Google Image Result for https://'. The address bar shows 'Not secure | 3.230.159.132:50070/dfshealth.html#tab-overview'. The navigation bar has tabs for 'Hadoop', 'Overview', 'Datanodes', 'Datanode Volume Failures', 'Snapshot', 'Startup Progress', and 'Utilities'. The main content area is titled 'Overview 'localhost:9000' (active)' and contains a table with the following information:

Started:	Wed Jul 22 17:11:20 +0530 2020
Version:	2.9.2, r826afbeae31ca687bc2f8471dc841b66ed2c6704
Compiled:	Tue Nov 13 18:12:00 +0530 2018 by ajsaka from branch-2.9.2
Cluster ID:	CID-4260df75-b2f0-4dd5-b4a2-cd28800a628a
Block Pool ID:	BP-272257960-172.31.68.92-1595417210612

Check Resource Manager web interface

<http://192.168.56.70:8088>



The screenshot shows the Hadoop Resource Manager web interface. The browser tabs include 'Instances | EC2 Management Console', 'All Applications', 'Azure - Google Search', and 'Google Image Result for https://'. The address bar shows 'Not secure | 3.230.159.132:8088/cluster'. The navigation bar has tabs for 'Cluster', 'About Nodes', 'Node Labels', 'Applications', 'NEW', 'NEW SAVING', 'SUBMITTED', 'ACCEPTED', 'RUNNING', 'FINISHED', 'FAILED', 'KILLED', 'Scheduler', and 'Tools'. The main content area is titled 'All Applications' and contains several tables:

Cluster Metrics

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Memory Used	Memory Total
0	0	0	0	0	0 B	8 GB

Cluster Nodes Metrics

Active Nodes	Decommissioning Nodes	Decommissioned Nodes	Lost Nodes	Unhealthy Nodes
1	0	0	0	0

Scheduler Metrics

Scheduler Type	Scheduling Resource Type	Minimum Allocation	Maximum Allocation
Capacity Scheduler	[MEMORY]	<memory:1024, vCores:1>	<memory:8192, vCores:1>

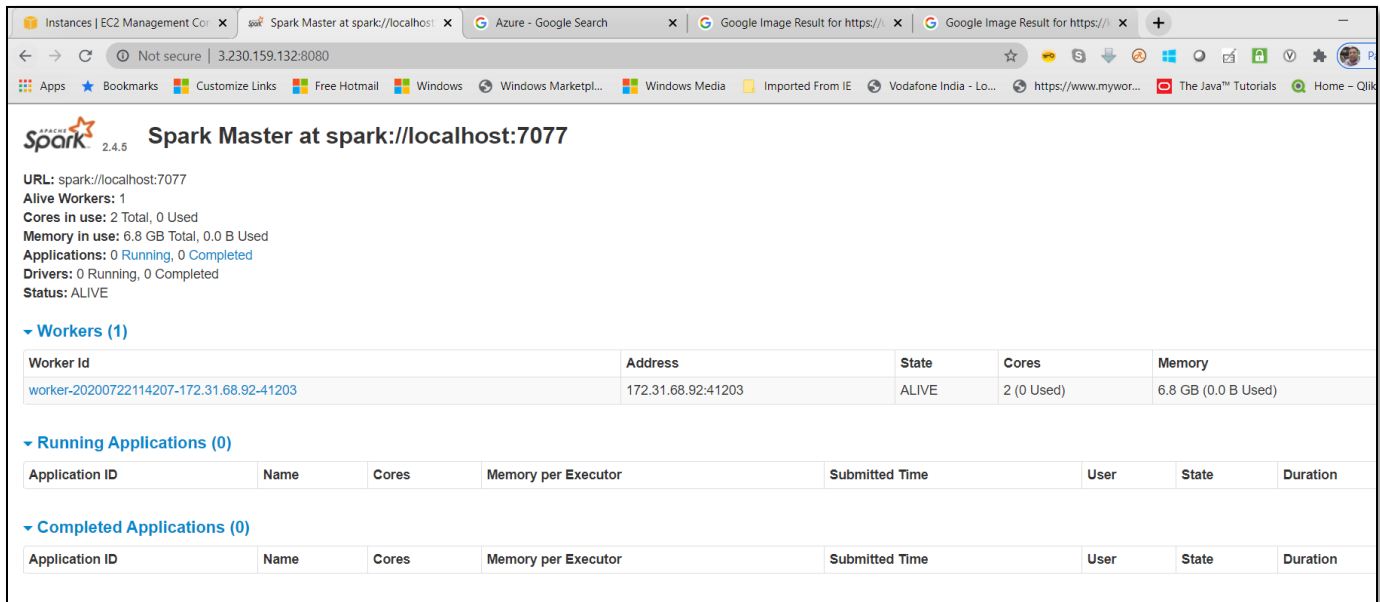
Showing 20 entries

ID	User	Name	Application Type	Queue	Application Priority	StartTime	FinishTime	State	FinalStatus	Running Containers	Allocated CPU V-Cores	Allocated Memory MB
No data available in table												

Showing 0 to 0 of 0 entries

Check Spark Master web interface

[http:// 192.168.56.70:8080](http://192.168.56.70:8080)



Spark Master at spark://localhost:7077

URL: spark://localhost:7077
 Alive Workers: 1
 Cores in use: 2 Total, 0 Used
 Memory in use: 6.8 GB Total, 0.0 B Used
 Applications: 0 Running, 0 Completed
 Drivers: 0 Running, 0 Completed
 Status: ALIVE

▼ Workers (1)

Worker Id	Address	State	Cores	Memory
worker-20200722114207-172.31.68.92-41203	172.31.68.92:41203	ALIVE	2 (0 Used)	6.8 GB (0.0 B Used)

▼ Running Applications (0)

Application ID	Name	Cores	Memory per Executor	Submitted Time	User	State	Duration
----------------	------	-------	---------------------	----------------	------	-------	----------

▼ Completed Applications (0)

Application ID	Name	Cores	Memory per Executor	Submitted Time	User	State	Duration
----------------	------	-------	---------------------	----------------	------	-------	----------

Happy Clustering