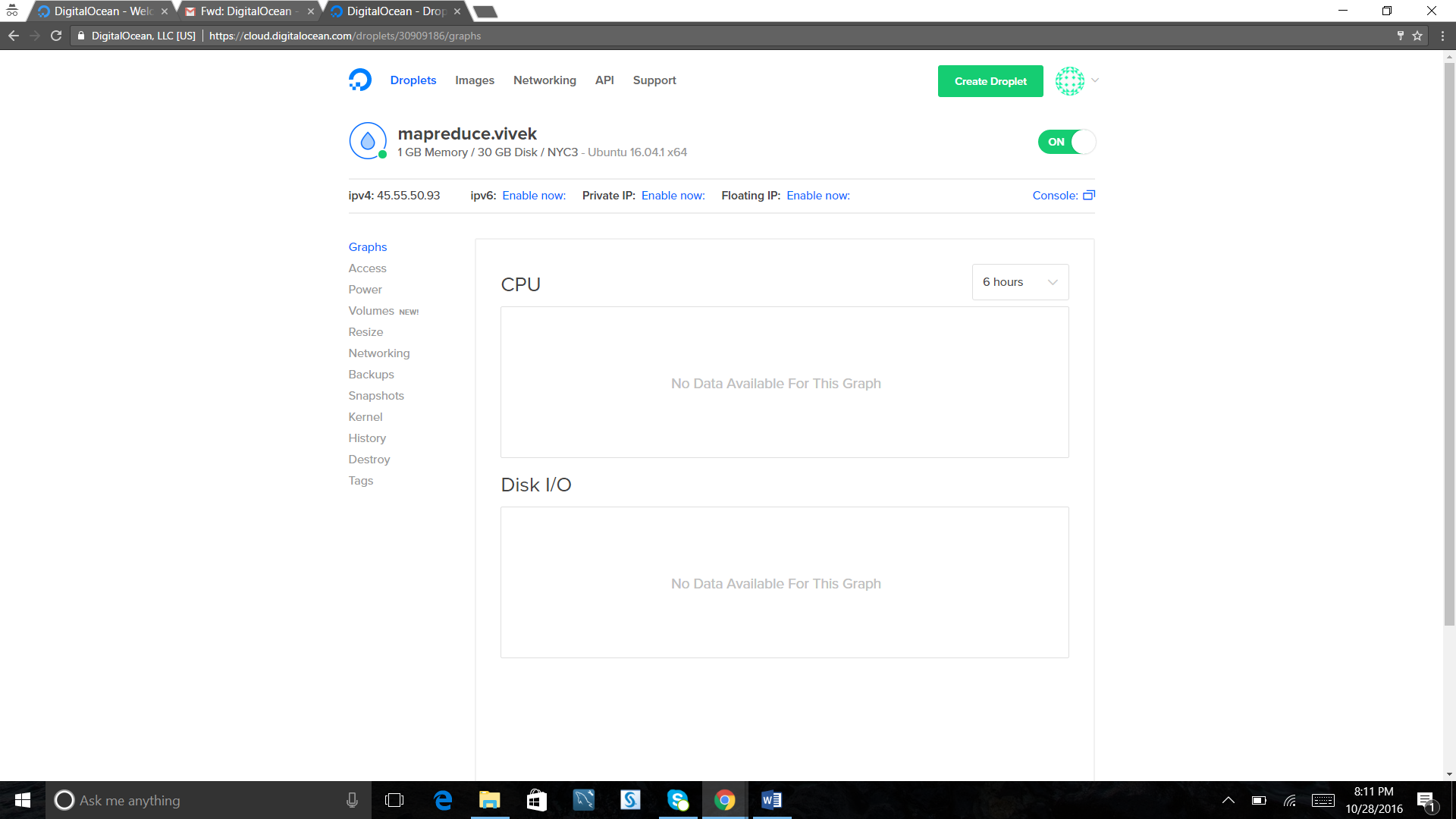
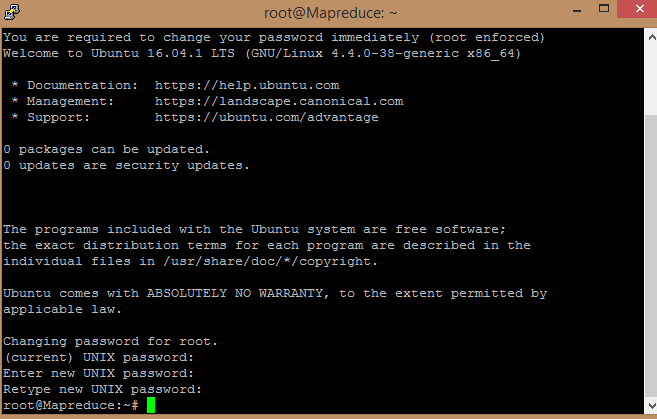
# **CLOUD DATA STORAGE Vivek Ravi, 800934913**

MAP REDUCE

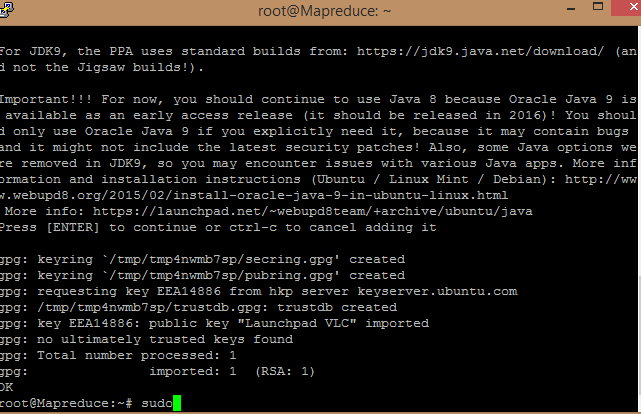
Necessary screenshots for creating account in digital ocean

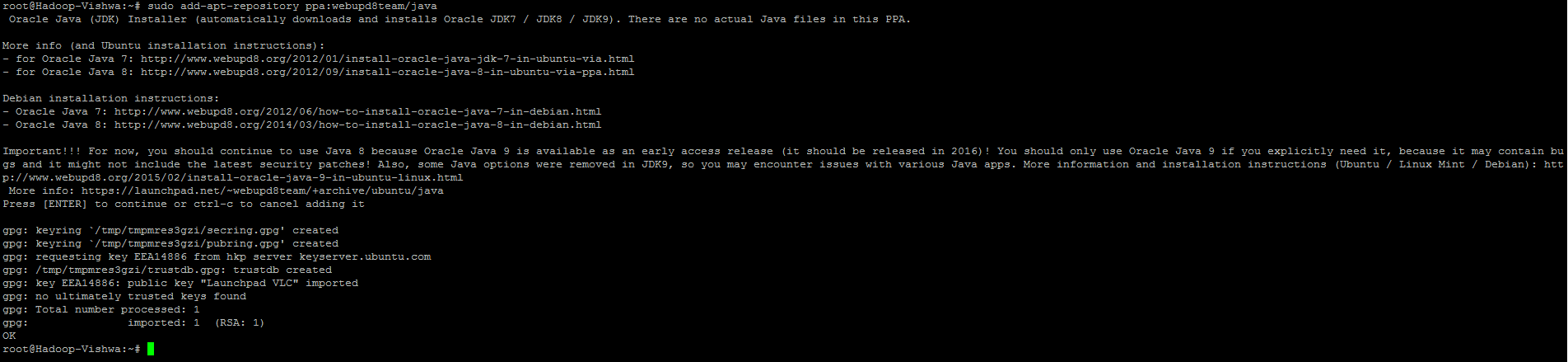


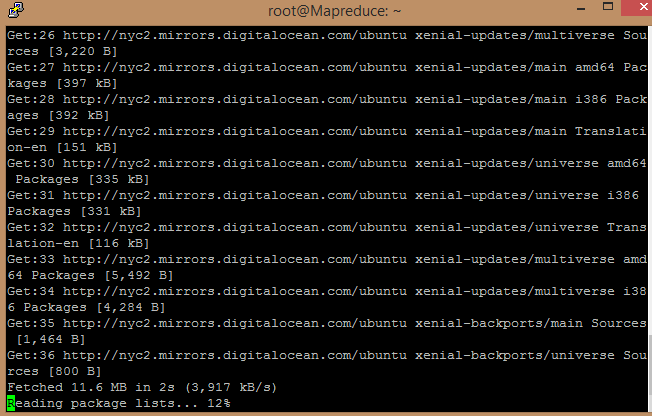


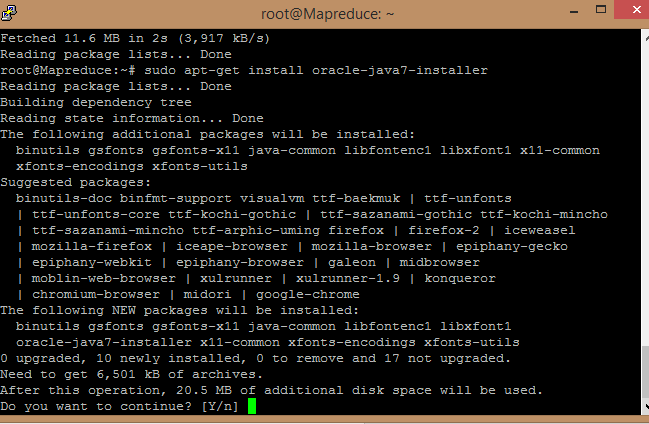
**Install Hadoop**:

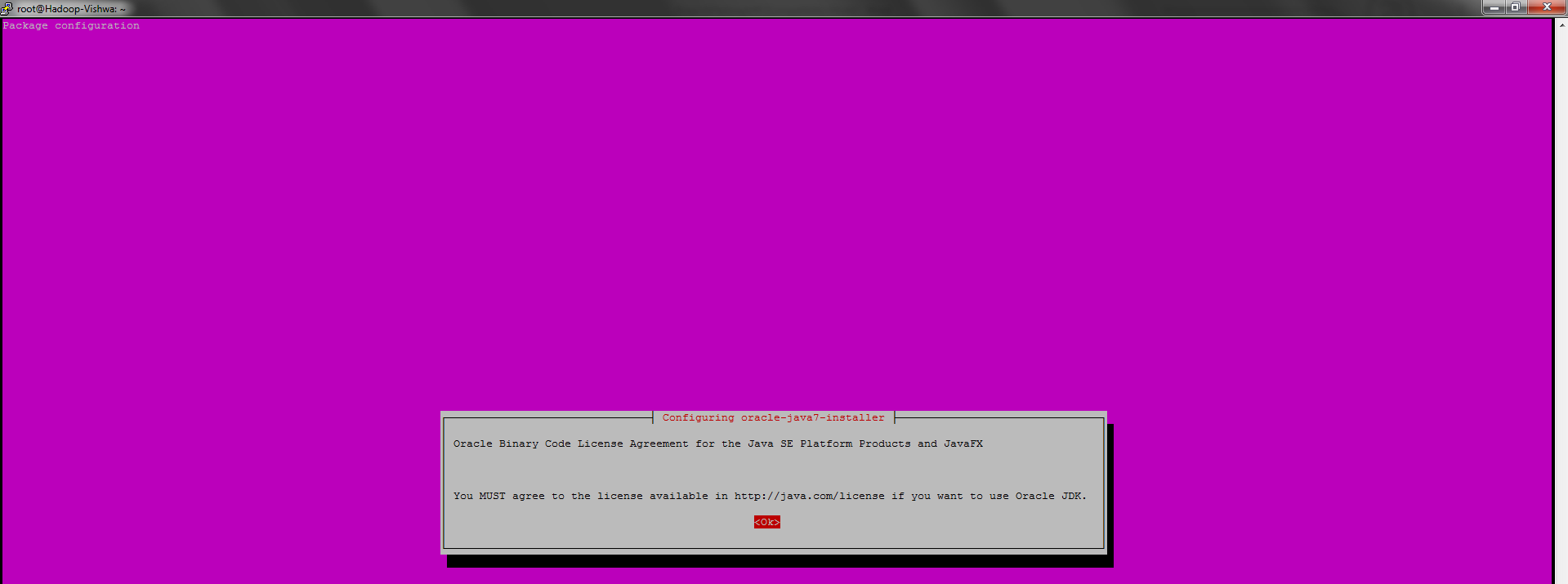
**sudo add-apt-repository ppa:webupd8team/java**

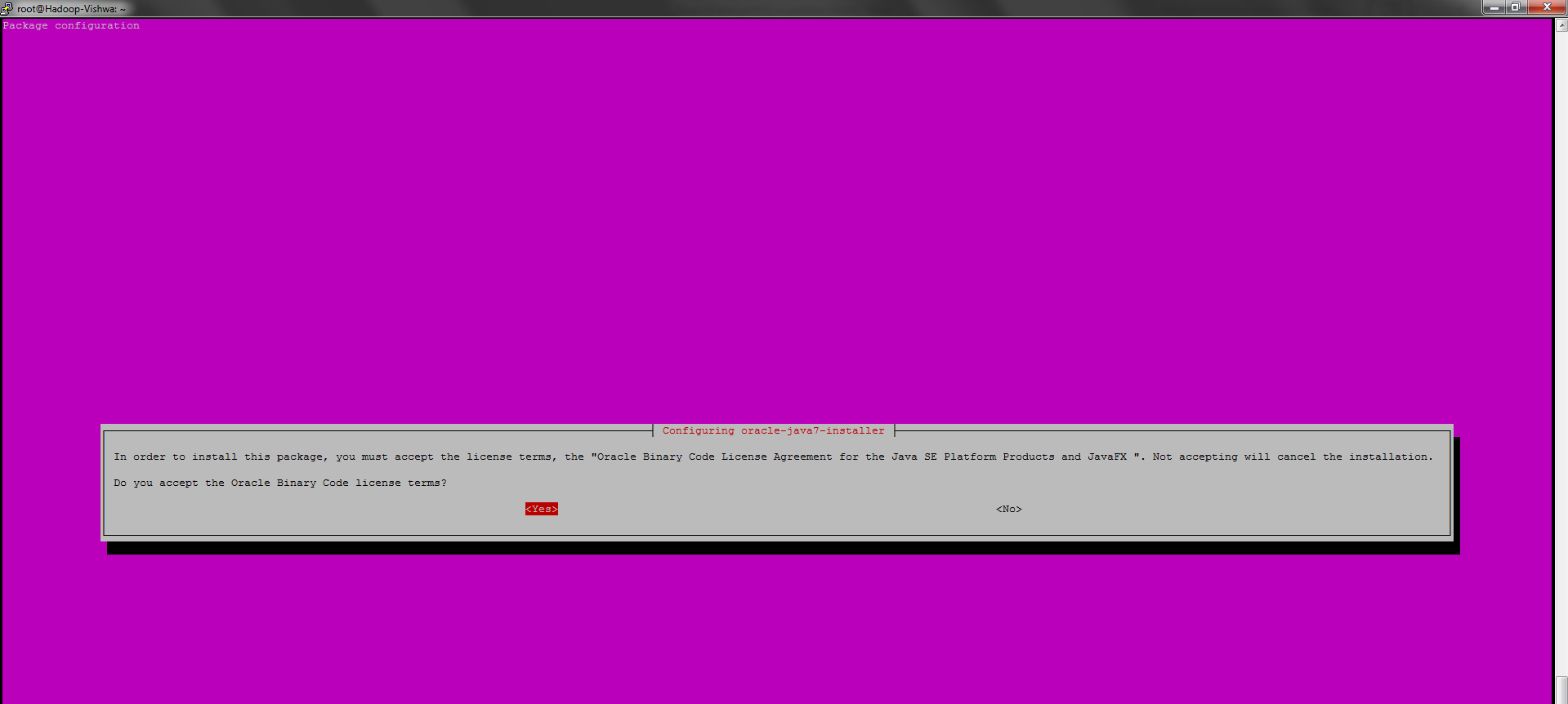


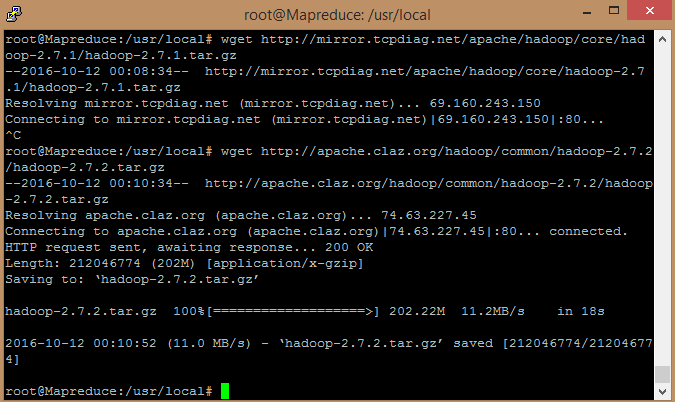
  
**sudo apt-get update**



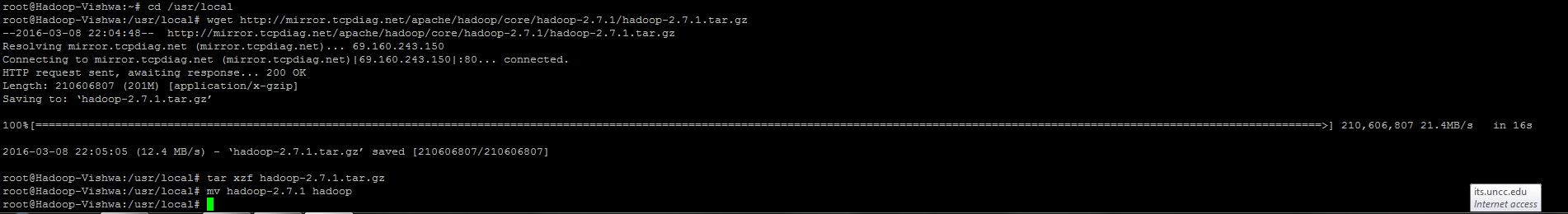






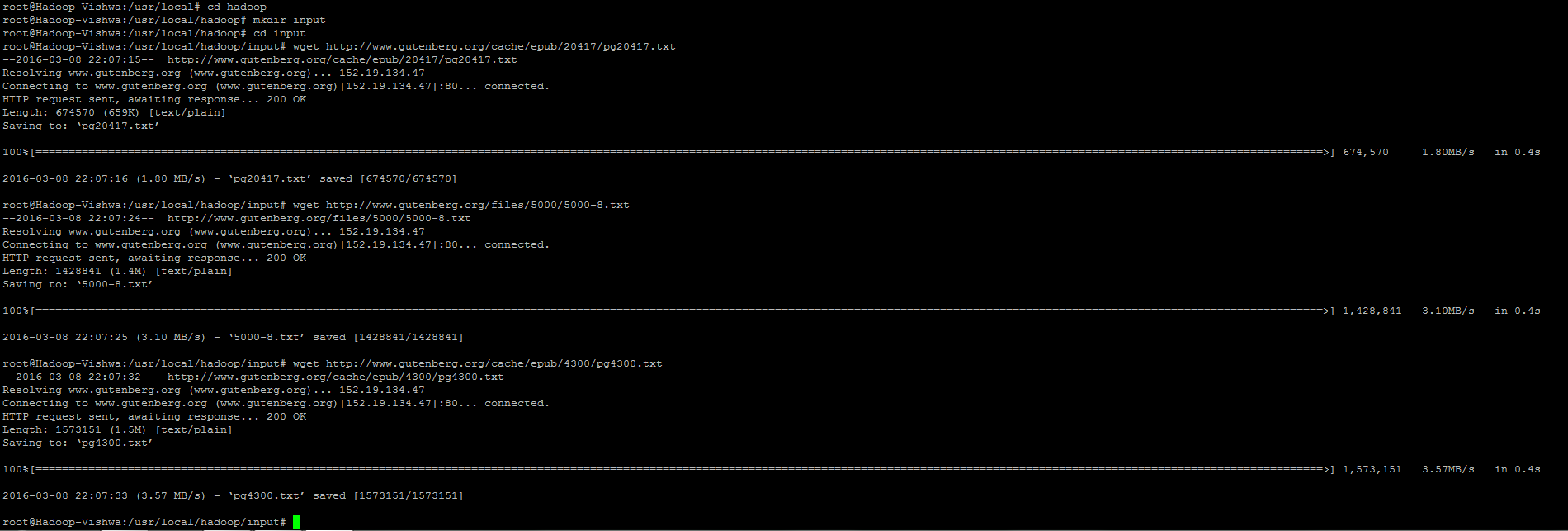


**cd /usr/local  
wget** [**http://mirror.tcpdiag.net/apache/hadoop/core/hadoop-2.7.1/hadoop-2.7.1.tar.gz**](http://mirror.tcpdiag.net/apache/hadoop/core/hadoop-2.7.1/hadoop-2.7.1.tar.gz) **tar xzf hadoop-2.7.1.tar.gz  
mv hadoop-2.7.1 hadoop**

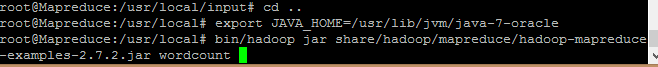


**Standalone HADOOP Operation**:

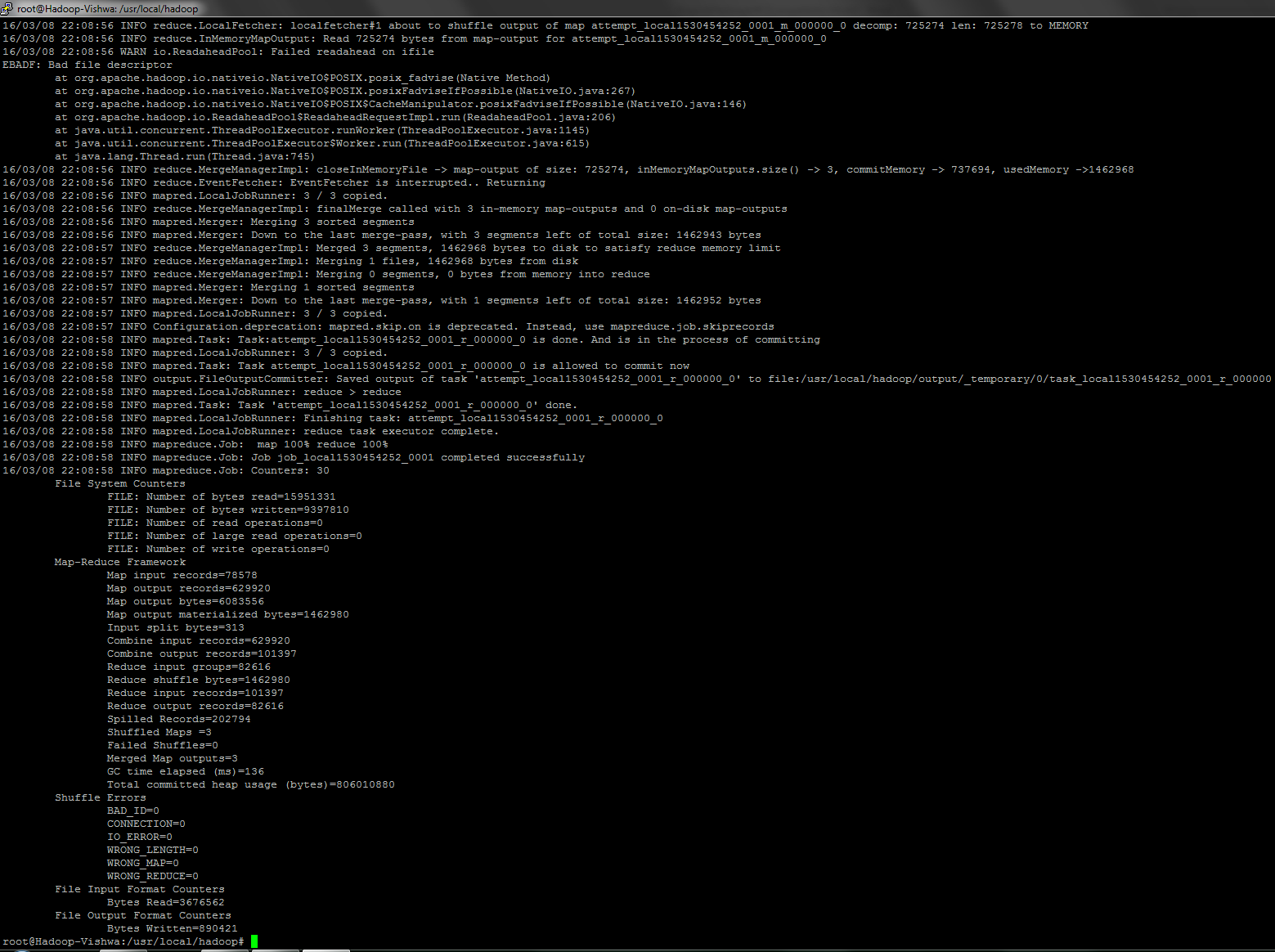
**mkdir input  
cd input**  
**wget** [**http://www.gutenberg.org/cache/epub/20417/pg20417.txt**](http://www.gutenberg.org/cache/epub/20417/pg20417.txt) **wget** [**http://www.gutenberg.org/files/5000/5000-8.txt**](http://www.gutenberg.org/files/5000/5000-8.txt) **wget** [**http://www.gutenberg.org/cache/epub/4300/pg4300.txt**](http://www.gutenberg.org/cache/epub/4300/pg4300.txt)



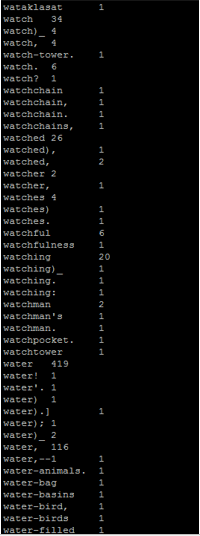
**cd ..  
export JAVA\_HOME=/usr/lib/jvm/java-7-oracle**

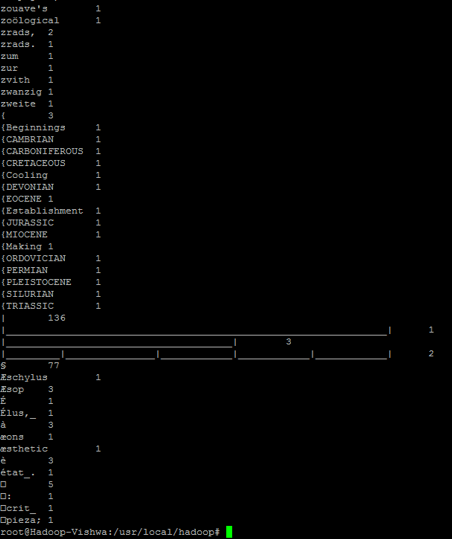


**bin/hadoop jar share/hadoop/mapreduce/hadoop-mapreduce-examples-2.7.1.jar wordcount input output**



**cat output/\***





**Pseudo-Distributed Operation:**  
**vi etc/hadoop/core-site.xml to add  
….  
<configuration>**

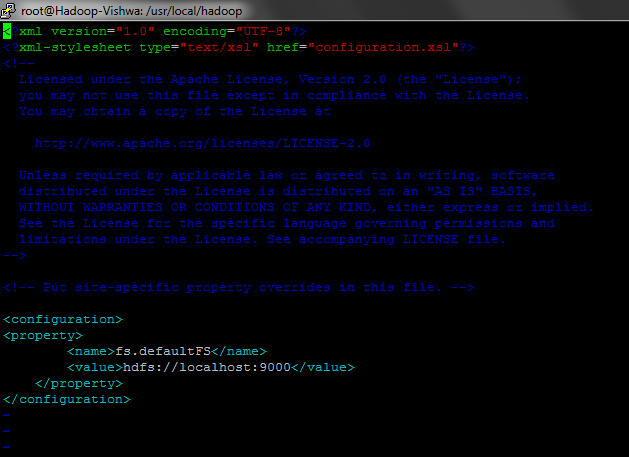
**<property>**

**<name>fs.defaultFS</name>**

**<value>hdfs://localhost:9000</value>**

**</property>**

**</configuration>**



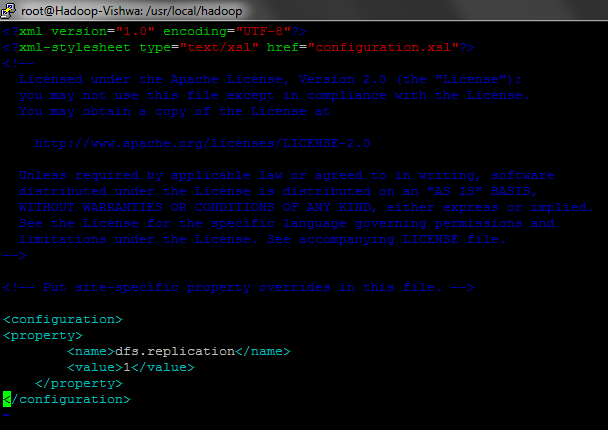
vi etc/hadoop/hdfs-site.xml:  
…  
**<configuration>**

**<property>**

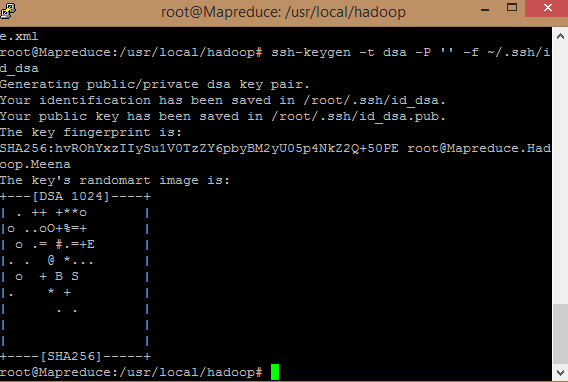
**<name>dfs.replication</name>**

**<value>1</value>**

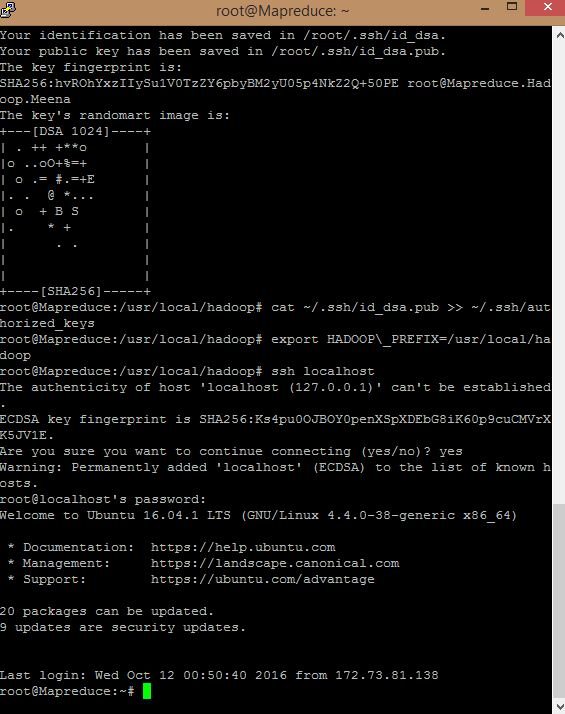
**</property>**

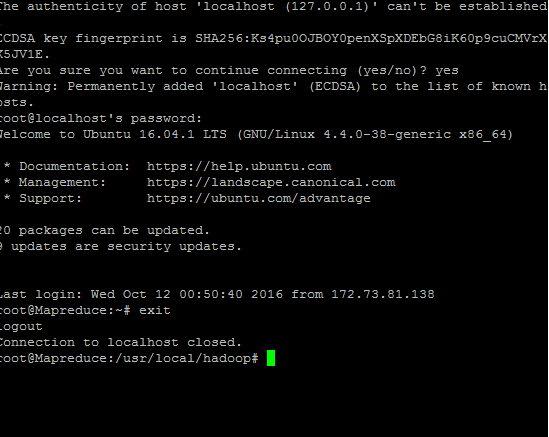
**</configuration>**  
…  


**ssh-keygen -t dsa -P '' -f ~/.ssh/id\_dsa**



**cat ~/.ssh/id\_dsa.pub >> ~/.ssh/authorized\_keys  
export HADOOP\\_PREFIX=/usr/local/hadoop  
ssh localhost  
exit**





**add the following to end of $HOME/.bashrc  
…………..  
# Set Hadoop-related environment variables  
export HADOOP\_HOME=/usr/local/hadoop  
# Set JAVA\_HOME (we will also configure JAVA\_HOME directly for Hadoop later on)  
export JAVA\_HOME=/usr/lib/jvm/java-7-oracle  
# Some convenient aliases and functions for running Hadoop-related commands  
unalias fs &> /dev/null  
alias fs="hadoop fs"  
unalias hls &> /dev/null  
alias hls="fs -ls"  
# If you have LZO compression enabled in your Hadoop cluster and  
# compress job outputs with LZOP (not covered in this tutorial):  
# Conveniently inspect an LZOP compressed file from the command  
# line; run via:  
#  
# $ lzohead /hdfs/path/to/lzop/compressed/file.lzo  
#  
# Requires installed 'lzop' command.  
#  
lzohead () {  
 hadoop fs -cat $1 | lzop -dc | head -1000 | less  
}  
# Add Hadoop**

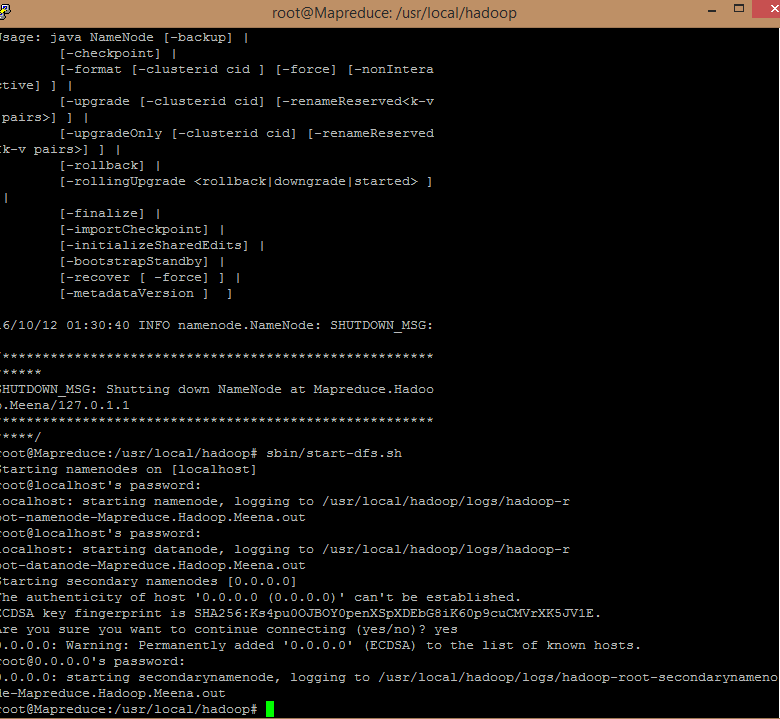


vi /usr/local/hadoop/etc/hadoop/hadoop-env.sh to **revise**   
export JAVA\_HOME=/usr/lib/jvm/java-7-oracle  


**bin/hdfs namenode –format**

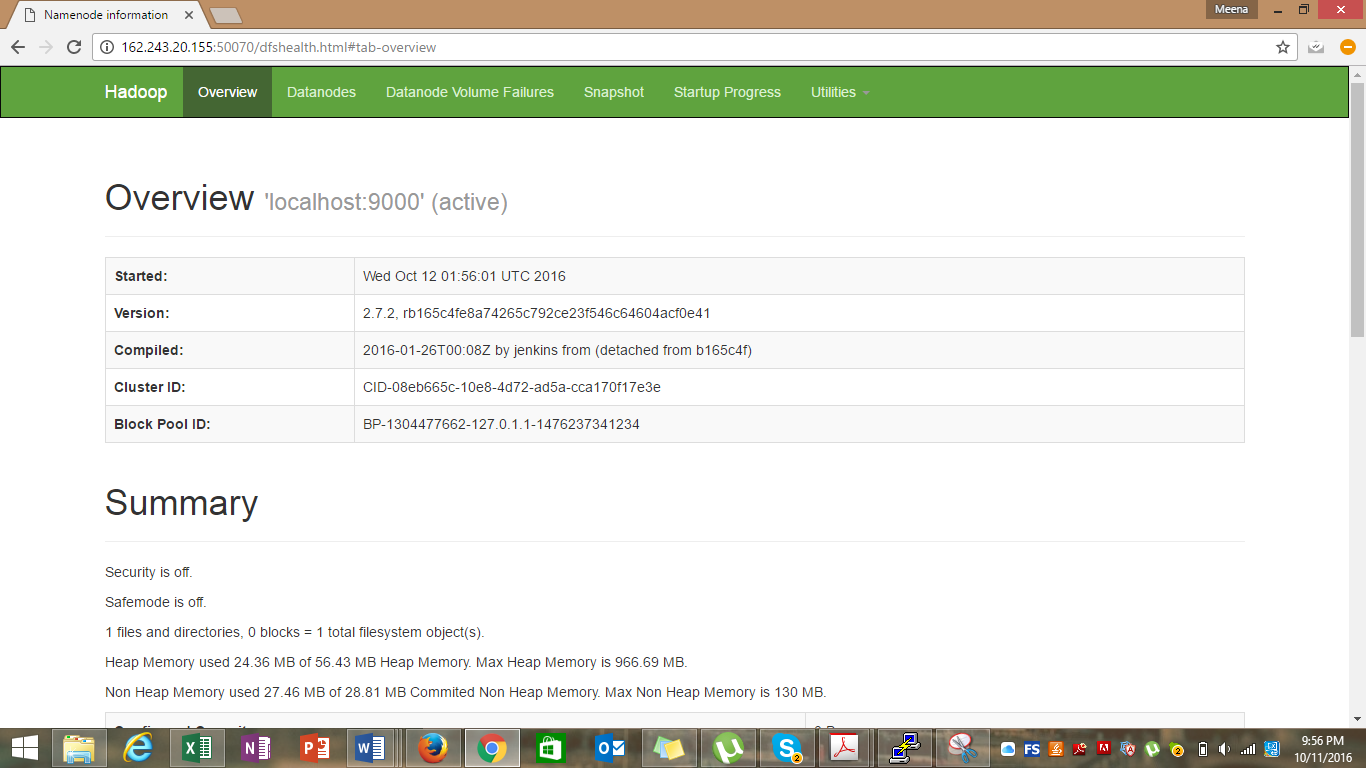


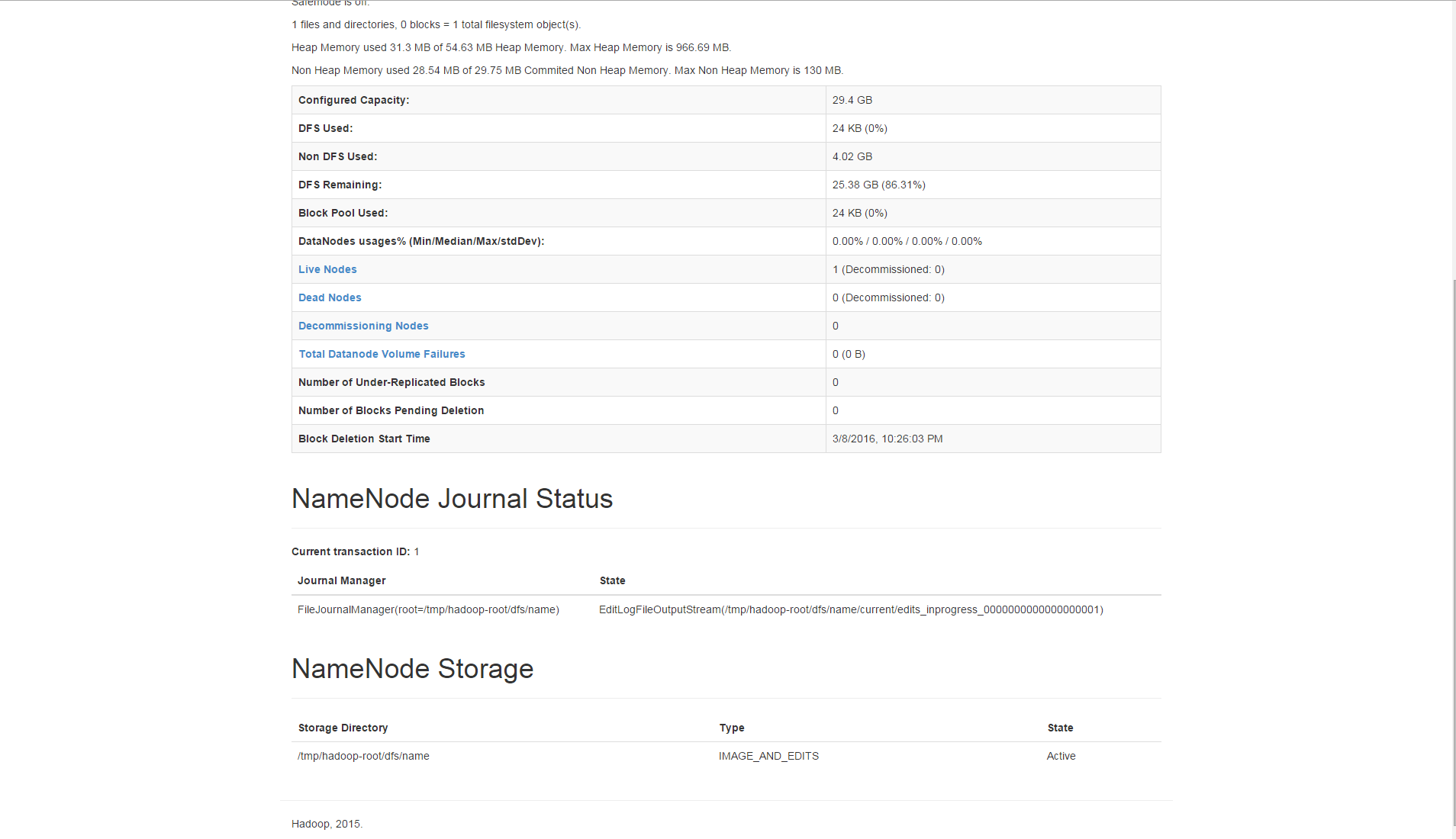
**sbin/start-dfs.sh**



**check your server is running by open**  
<http://yourIPaddress:50070>

http:// 45.55.50.93:50070/

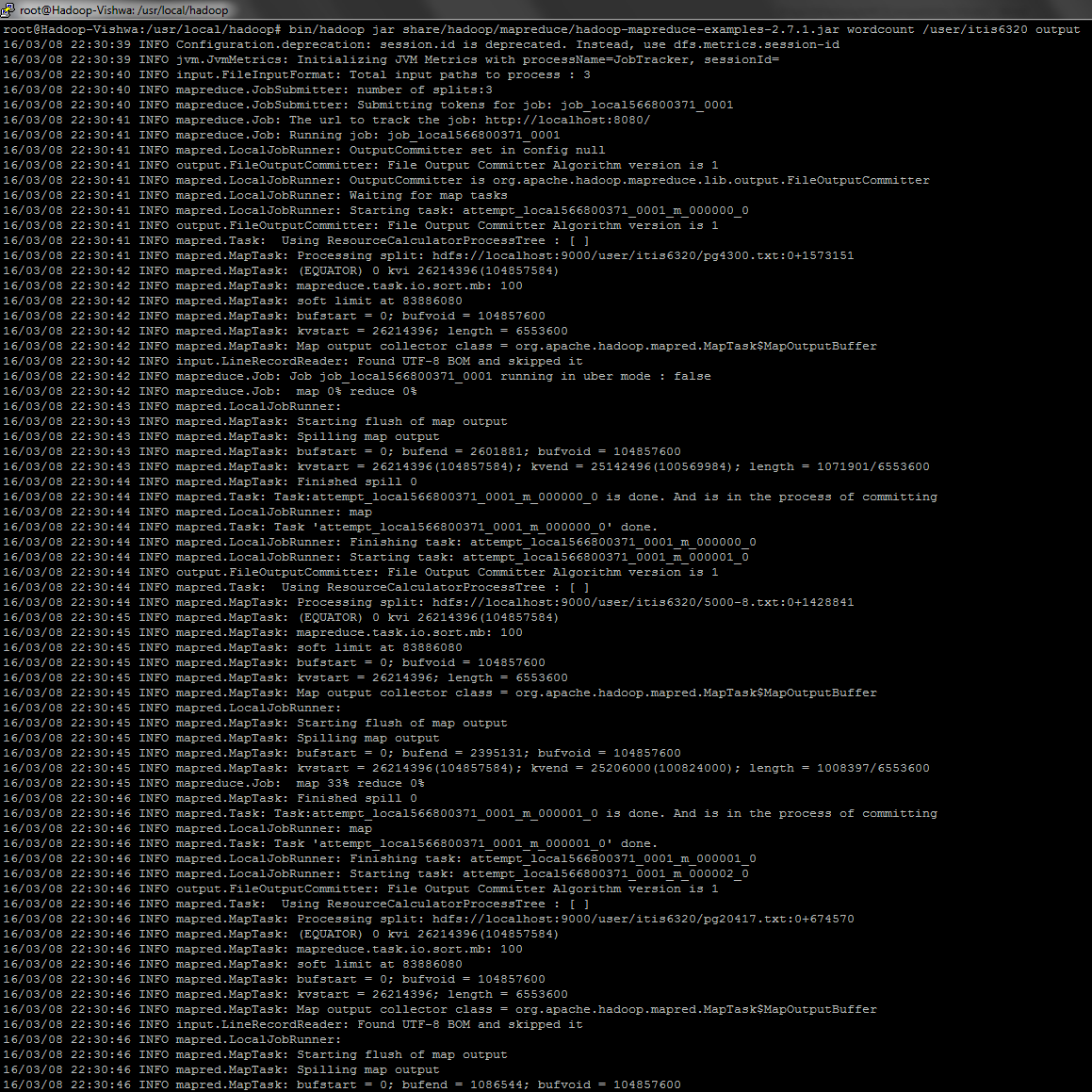


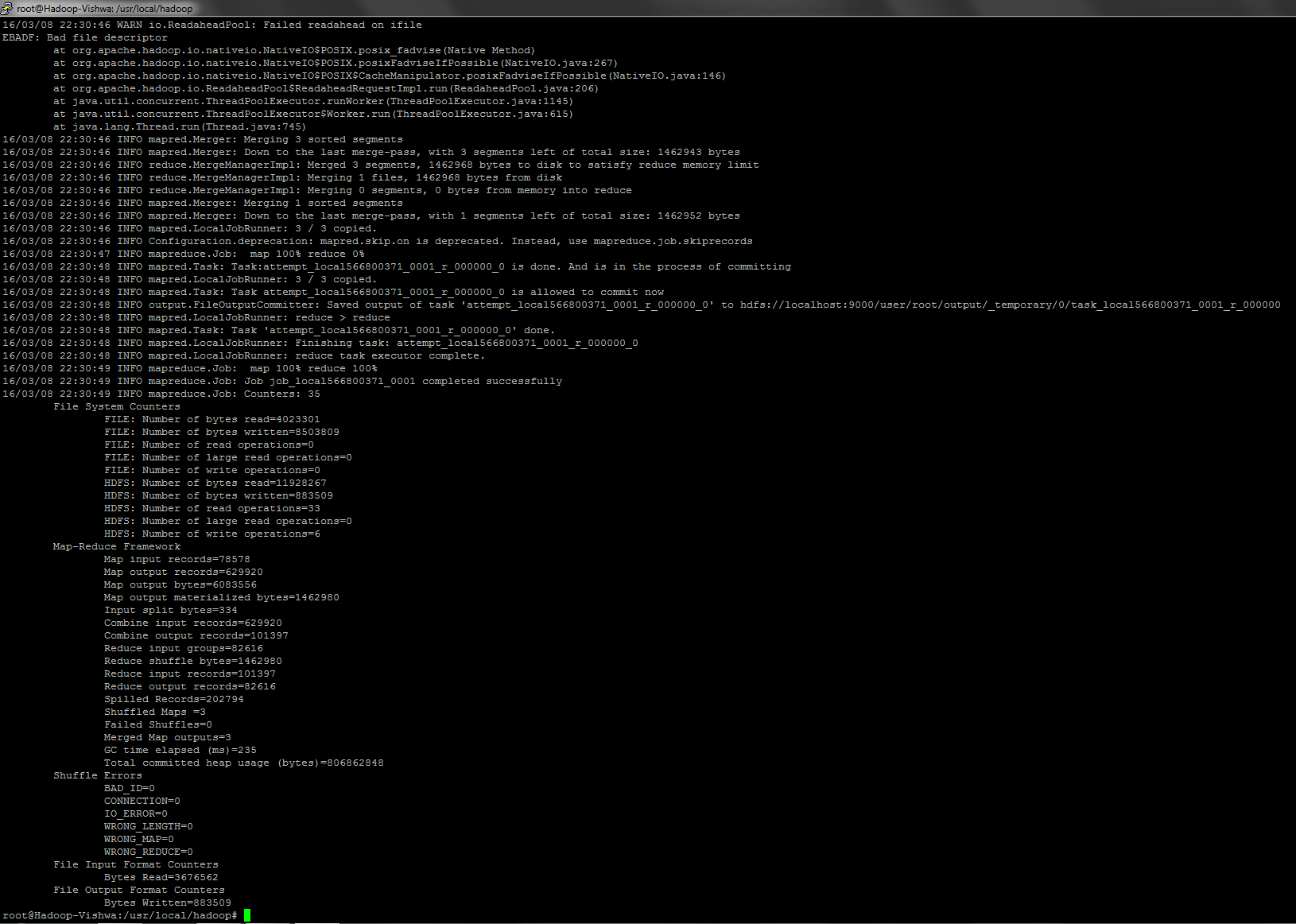


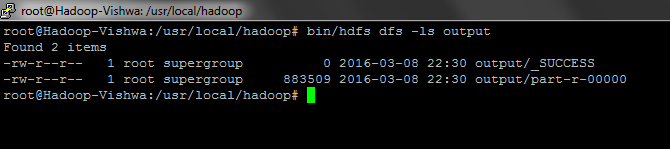
**bin/hdfs dfs -mkdir /user  
bin/hdfs dfs -mkdir /user/itis6320  
bin/hdfs dfs -put input/\* /user/itis6320  
bin/hdfs dfs -ls /user/itis6320**



**bin/hadoop jar share/hadoop/mapreduce/hadoop-mapreduce-examples-2.7.1.jar wordcount /user/itis6320 output**



  
**bin/hdfs dfs -ls output**

  
**bin/hdfs dfs -rm output/\*  
sbin/stop-dfs.sh**

**mv etc/hadoop/mapred-site.xml.template etc/hadoop/mapred-site.xml  
vi etc/hadoop/mapred-site.xml  
…  
<configuration>**

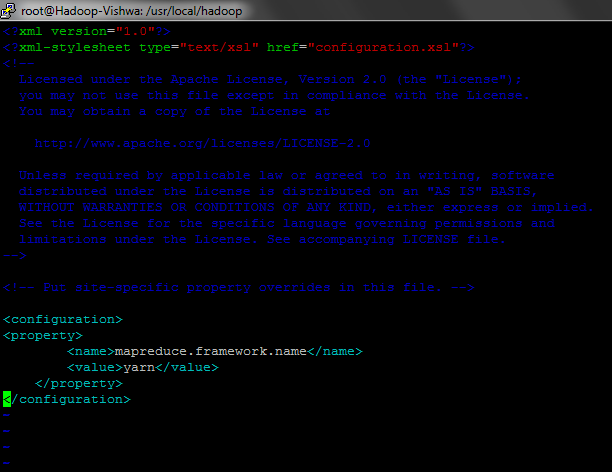
**<property>**

**<name>mapreduce.framework.name</name>**

**<value>yarn</value>**

**</property>**

**</configuration>**  
…

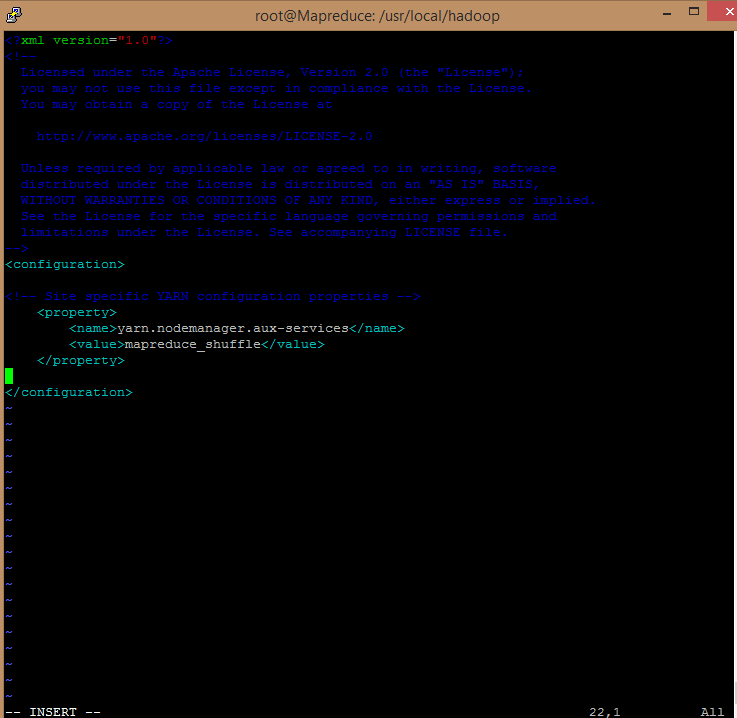
  
**vi etc/hadoop/yarn-site.xml  
…  
<configuration>**

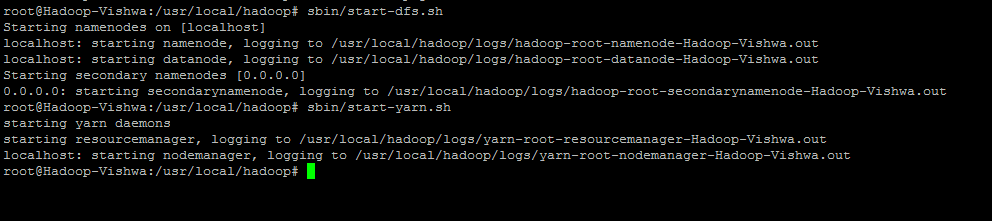
**<property>**

**<name>yarn.nodemanager.aux-services</name>**

**<value>mapreduce\_shuffle</value>**

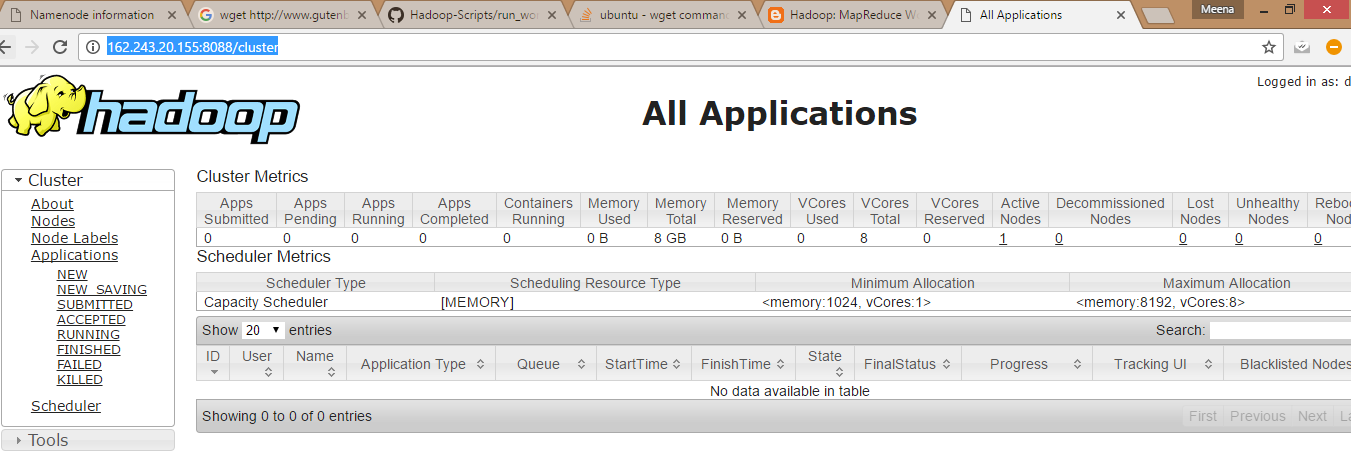
**</property>**

**</configuration>**  
…  


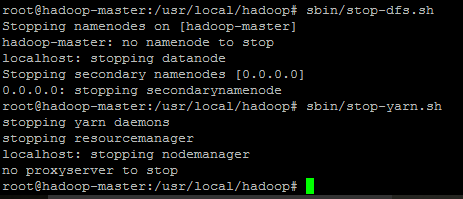
**sbin/start-dfs.sh  
sbin/start-yarn.sh**

<http://yourIPaddress:8088/>

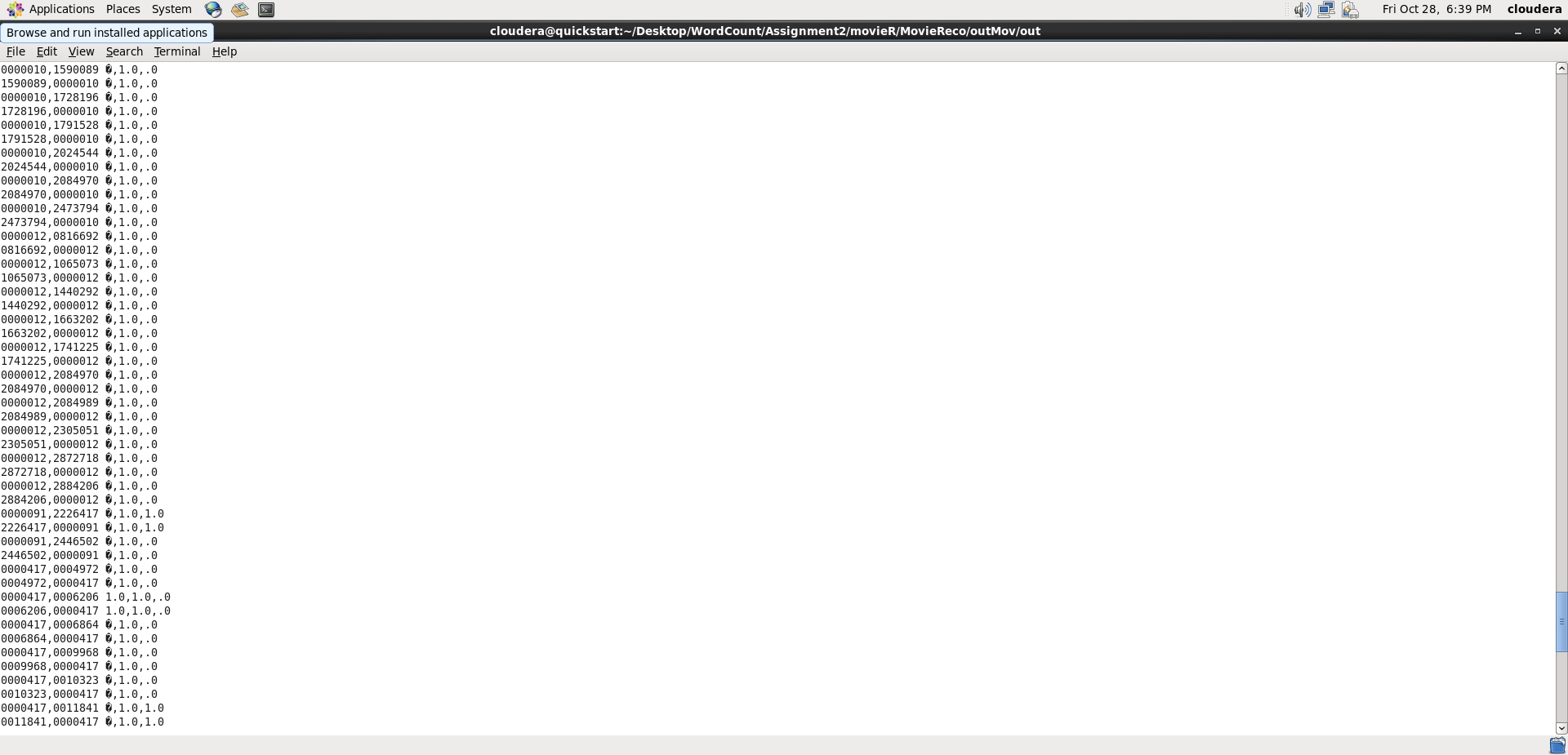
[http:// 45.55.50.93:8088/](http://162.243.20.155:8088/)



**sbin/stop-dfs.sh  
sbin/stop-yarn.sh**



**Map Reduce output**



**PART 2:**

Movie Similarities

For this part, we'll process a large corpus of movie ratings for the purpose of providing recommendations. When you're done, your program will help you decide what to watch on Netflix tonight. For each pair of movies in the data set, you will compute their [statistical correlation](http://en.wikipedia.org/wiki/Correlation_and_dependence) and [cosine similarity](http://en.wikipedia.org/wiki/Cosine_similarity) (see [this blog](http://blog.echen.me/2012/02/09/movie-recommendations-and-more-via-mapreduce-and-scalding/) for a discussion of these and other potential similarity metrics). Since this isn't a statistics class, the calculation of similarity metrics for Python and Java will be provided, but you need to provide them with the correct inputs.

For this section of the assignment, we have two input data sets: a small set ([details](https://github.com/sidooms/MovieTweetings)) for testing on your local machine or on digitalOcean and a large set ([details](http://grouplens.org/datasets/movielens/)) for running on Amazon's cloud.

The small set data can be obtained by:

wget <https://github.com/sidooms/MovieTweetings/blob/master/latest/ratings.dat>   
wget <https://github.com/sidooms/MovieTweetings/blob/master/latest/movies.dat>   
  
**Code and Output:**

The following java code implements the Mapper and reducer functions.

* In the first part of the code, the mapper function calculates the movie ratings for a user and the reducer function combine users based on movie ratings.
* In the second part of the code, the mapper function calculates the rating for a movie based on user ratings and reducer function combines the movie ratings for a specific movie.
* The final part of the code, summarize and display the movie ids, correlation, cosine similarities, jaccard coefficient.



Output:

Sample output which contains the similarity metrics.

Movie id Movie Id Correlation Cosine Jaccard

0000417, 1840309 -.7, 1.0, .0

1840309, 0000417 -.7, 1.0, .0

0000417, 1853728 .7, 1.0, .0

1853728, 0000417 .7, 1.0, .0

0000417, 1853739 .0, 1.0, .0

1853739, 0000417 .0, 1.0, .0

0000417, 1860213 -.7, .8, .0

1860213, 0000417 -.7, .8, .0

0000417, 1860353 0, 1.0, .0

1860353, 0000417 0, 1.0, .0

0000417, 1865505 1.0, 1.0, .0

1865505, 0000417 1.0, 1.0, .0