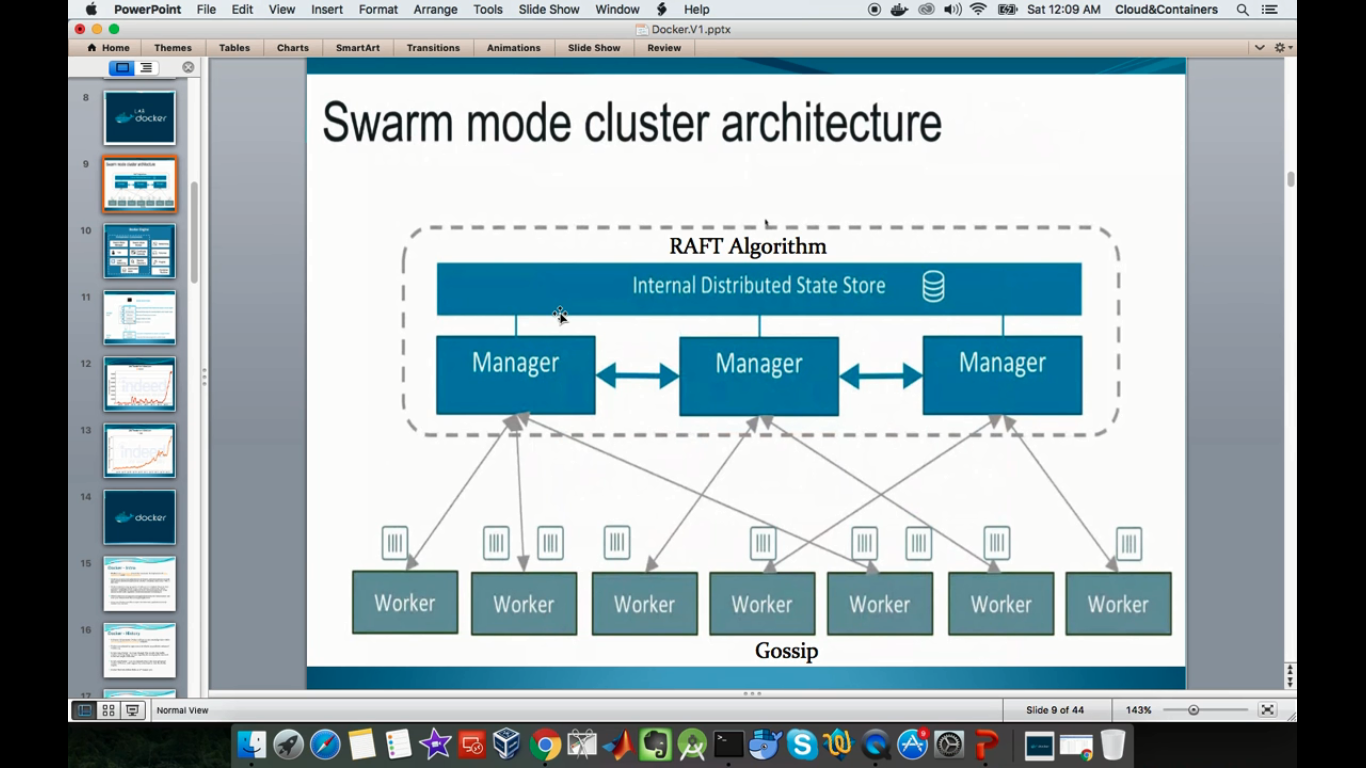
**Docker Swarm**

**Docker Swarm:**

Docker Swarm is a clustering and scheduling tool forDocker containers. With Swarm, IT administrators and developers can establish and manage a cluster of Docker nodes as a single virtual system.

Docker Swarm works on RAFT Algorithm find below diagram to understand functionality of docker Swarm



**Important Docker-swarm-Features:**

* The Clustermanagement is integrated with Docker Engine
* Decentralized Design
* Declarative Service Model
* Scaling
* Desired State Reconsiliation
* Secure by Default
* Automatic Load Balancing
* Rolling Updates

**Practical Session:**

Step1: We first we need give naming conventions to the nodes

# hostname master1

# vi /etc/hostname

master1

# hostname master2

# vi /etc/hostname

master2

# hostname master3

# vi /etc/hostname

master3

# hostname node1

# vi /etc/hostname

node1

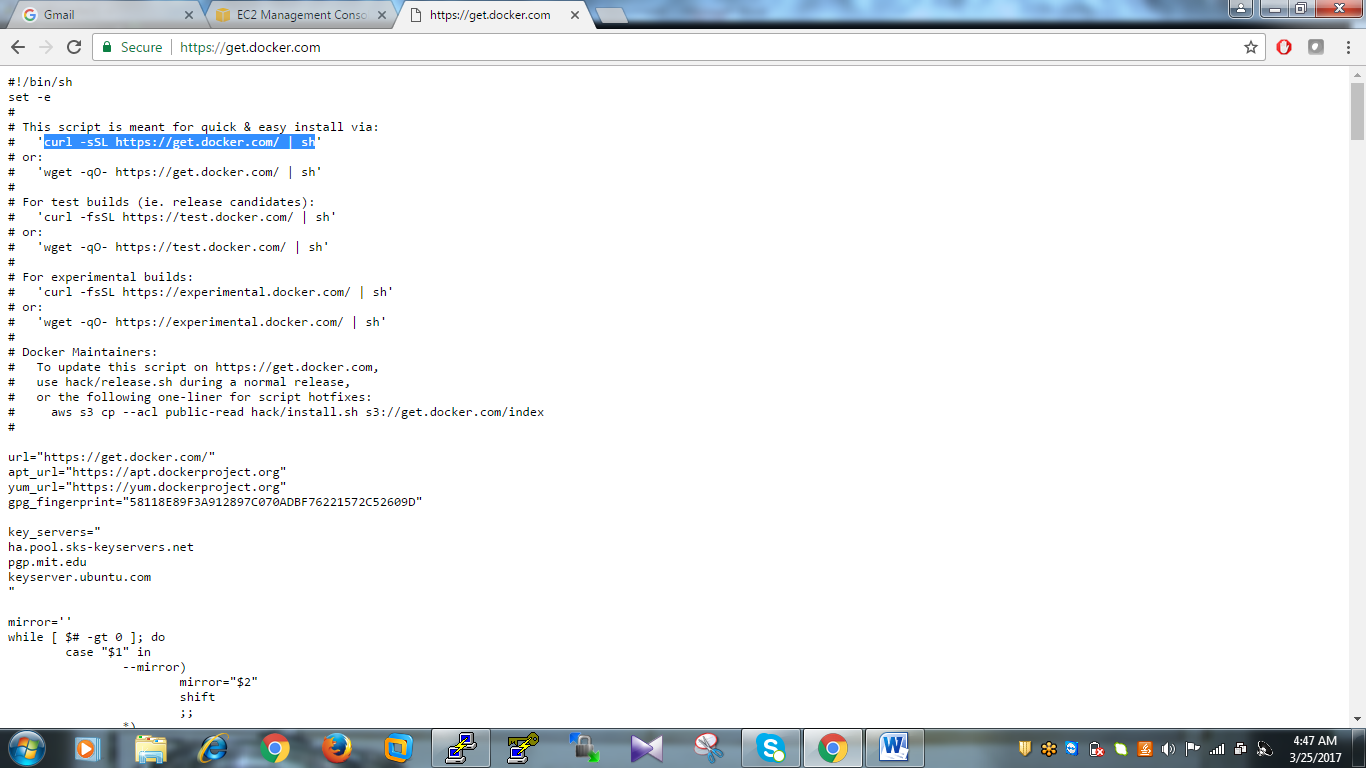
# hostname node2

# vi /etc/hostname

node2

Step2: Then copy the following URL from get.docker.com

# curl -sSL https://get.docker.com/ | sh

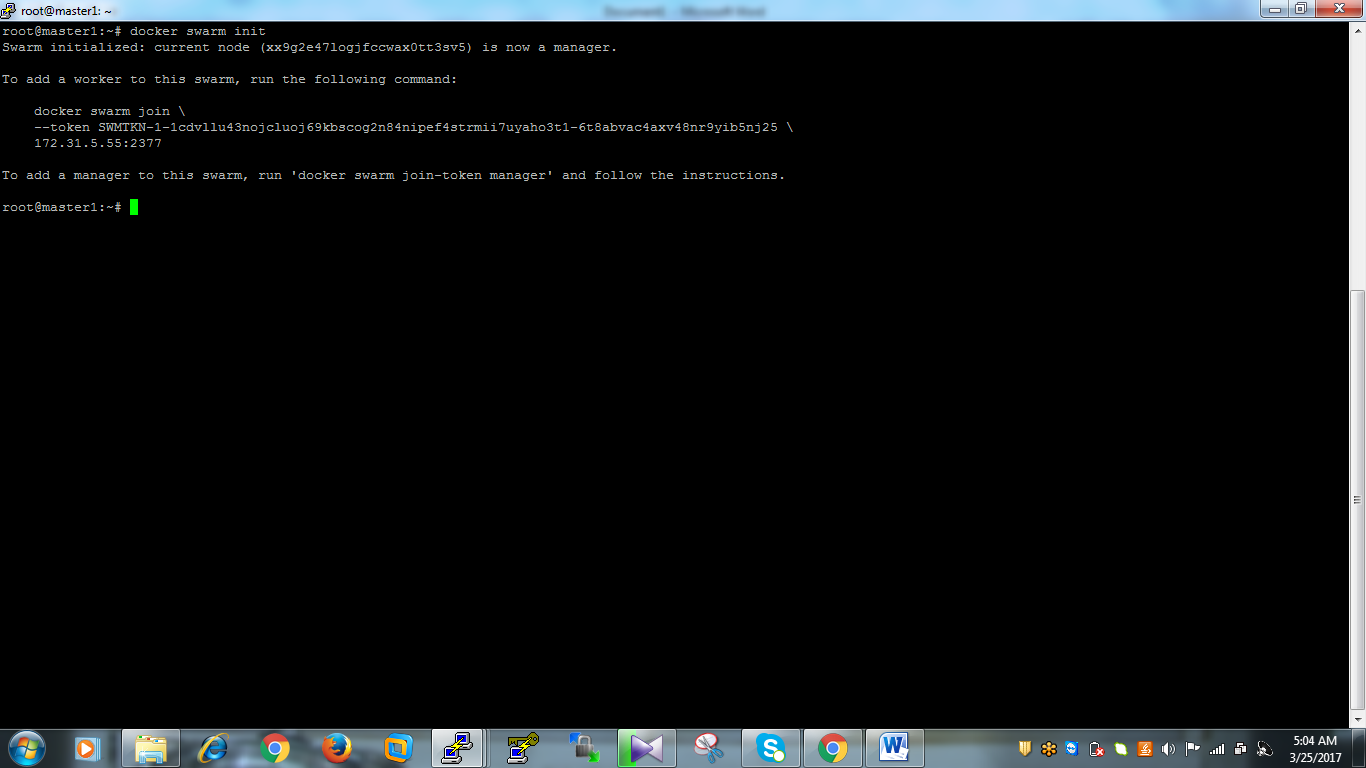


Step3: Like same you need to install docker in every machine using above script

Step4 : After completion of install of docker in each and every node then type the following command in one of master node preferably master1

# docker swarm init

Then you will get like following

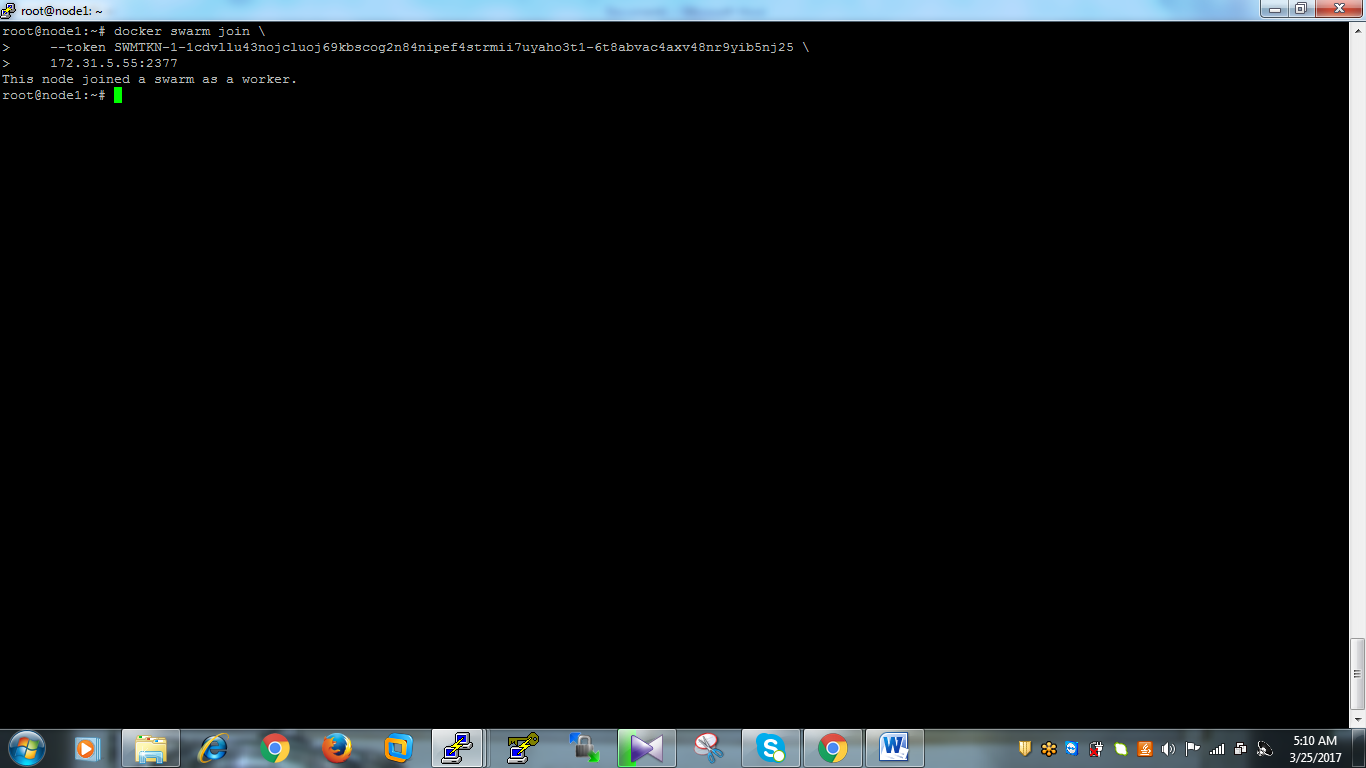


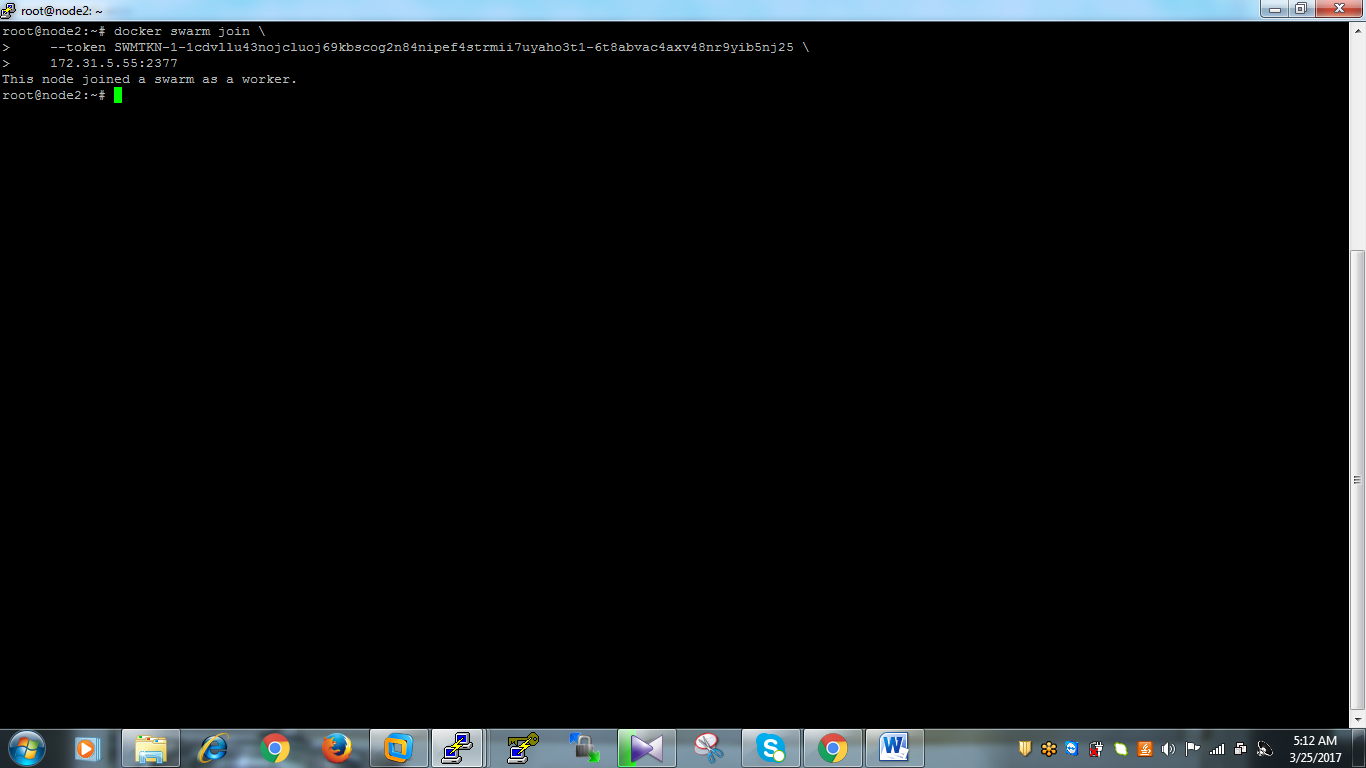
Step5 : Then you need to copy the like following code and execute the node machines to add workers to the manager like following.

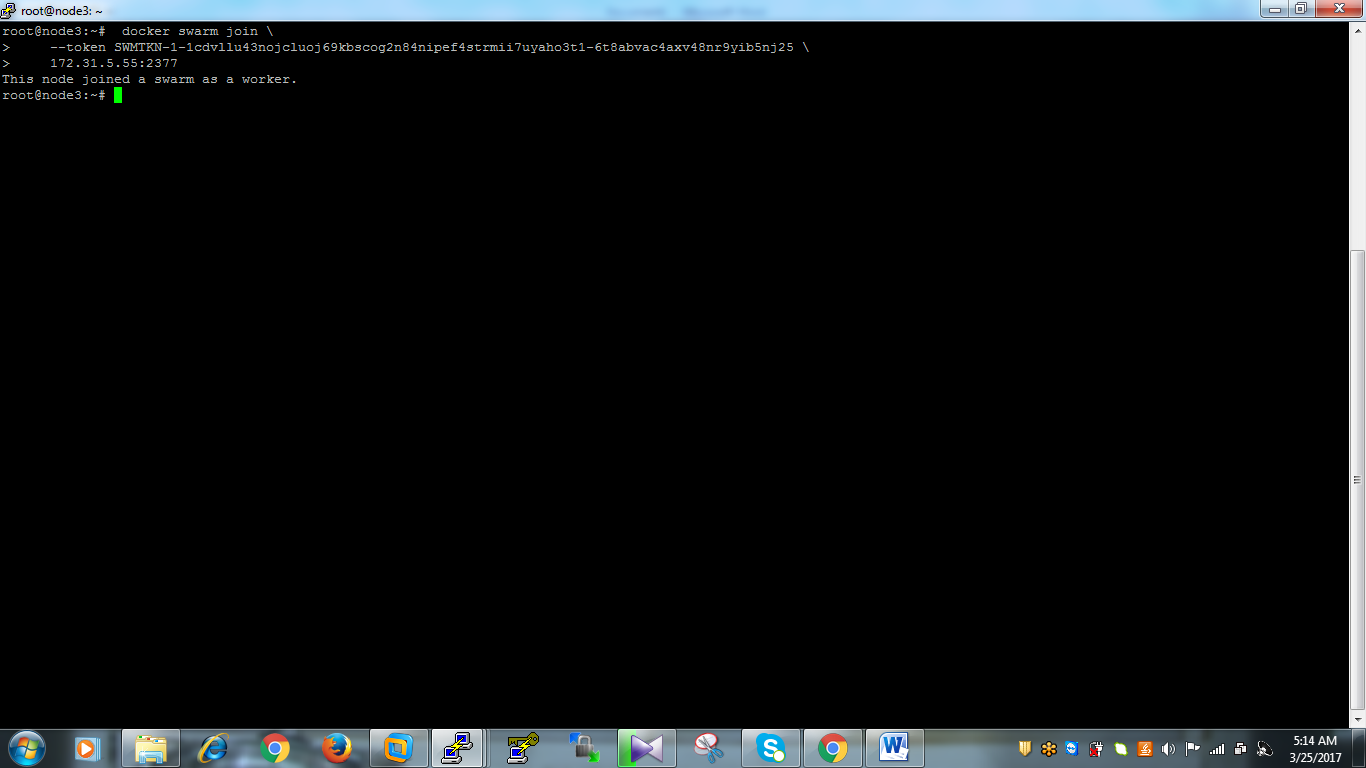
Ex : docker swarm join \

--token SWMTKN-1-1cdvllu43nojcluoj69kbscog2n84nipef4strmii7uyaho3t1-6t8abvac4axv48nr9yib5nj25 \

172.31.5.55:2377







Step6: After that you need to execute the following in master1 and the copy the link and execute it on master2 to add master2 to the managers list

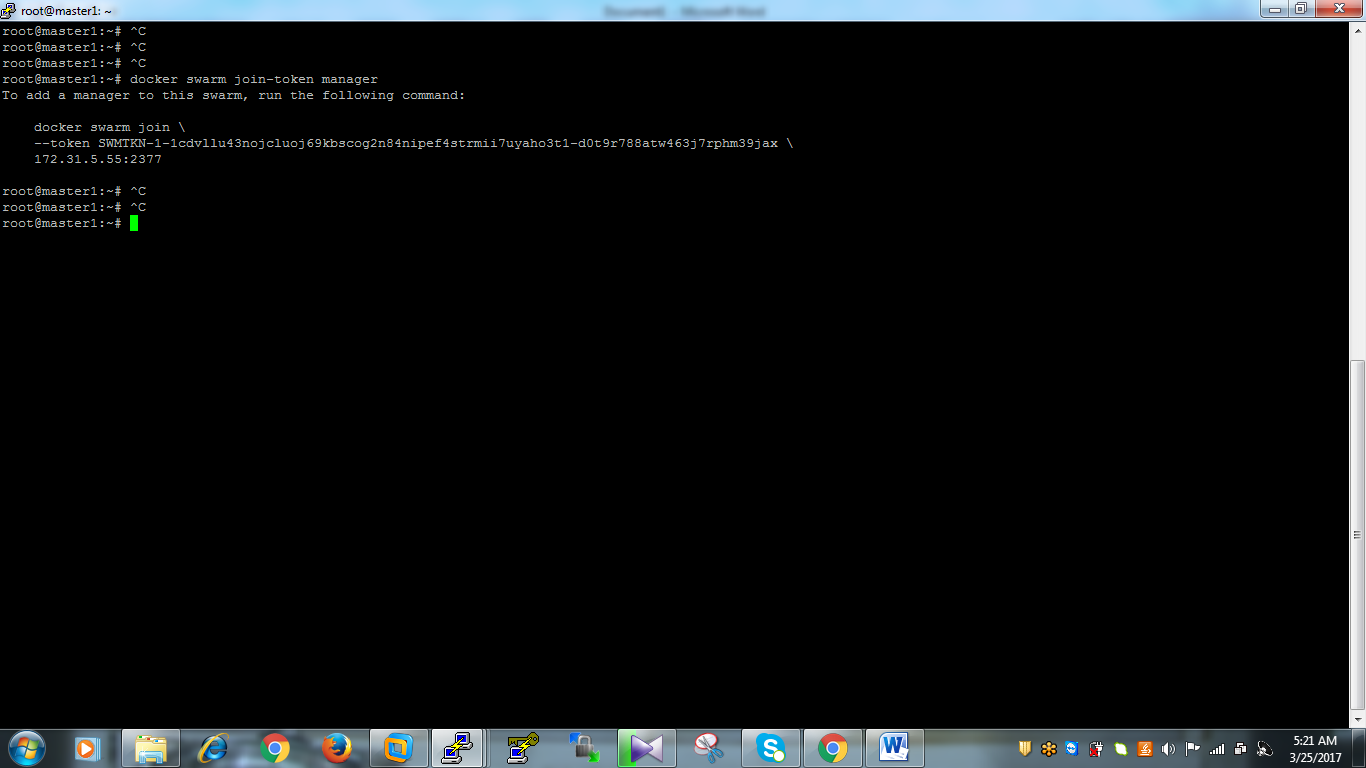
# docker swarm join-token manager

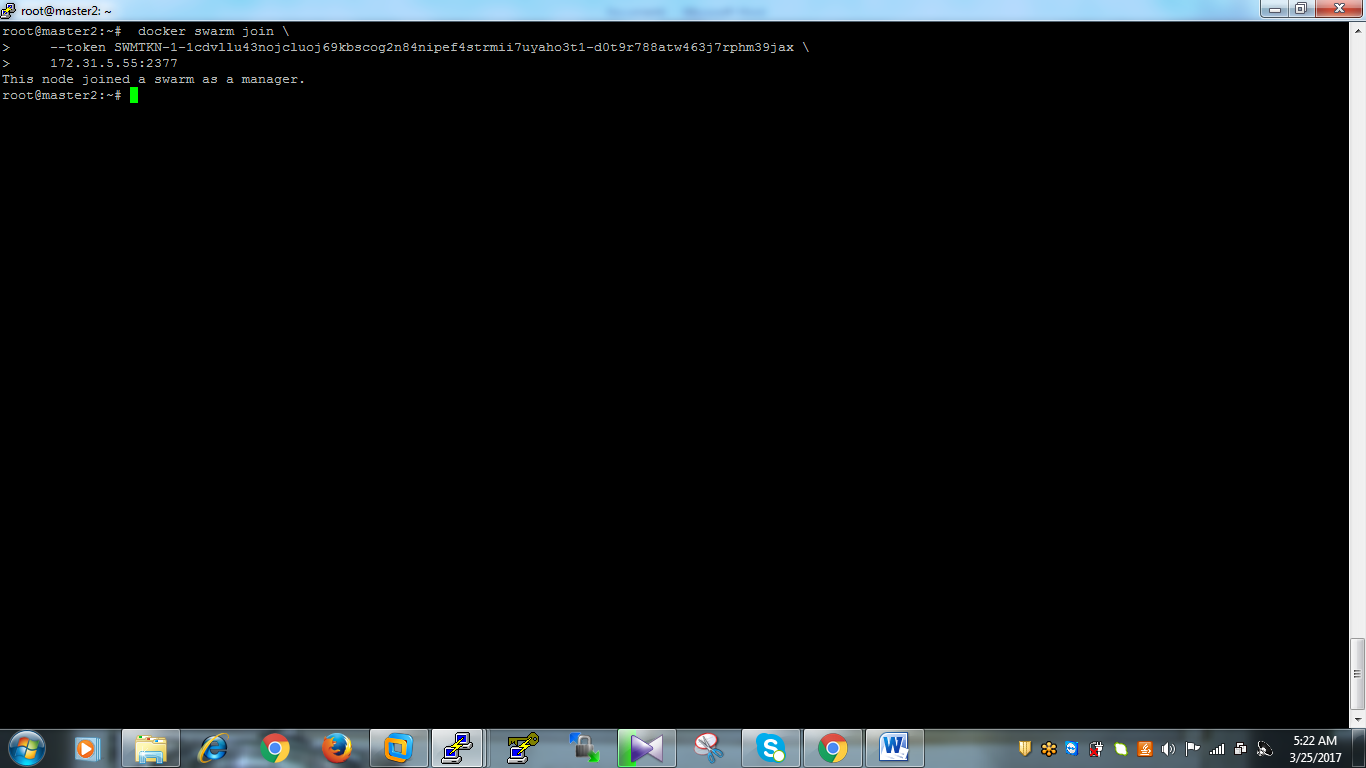
Ex : docker swarm join \

--token SWMTKN-1-1cdvllu43nojcluoj69kbscog2n84nipef4strmii7uyaho3t1-d0t9r788atw463j7rphm39jax \

172.31.5.55:2377

You need to copy the above link and execute it on master2 to add master2 to managers list

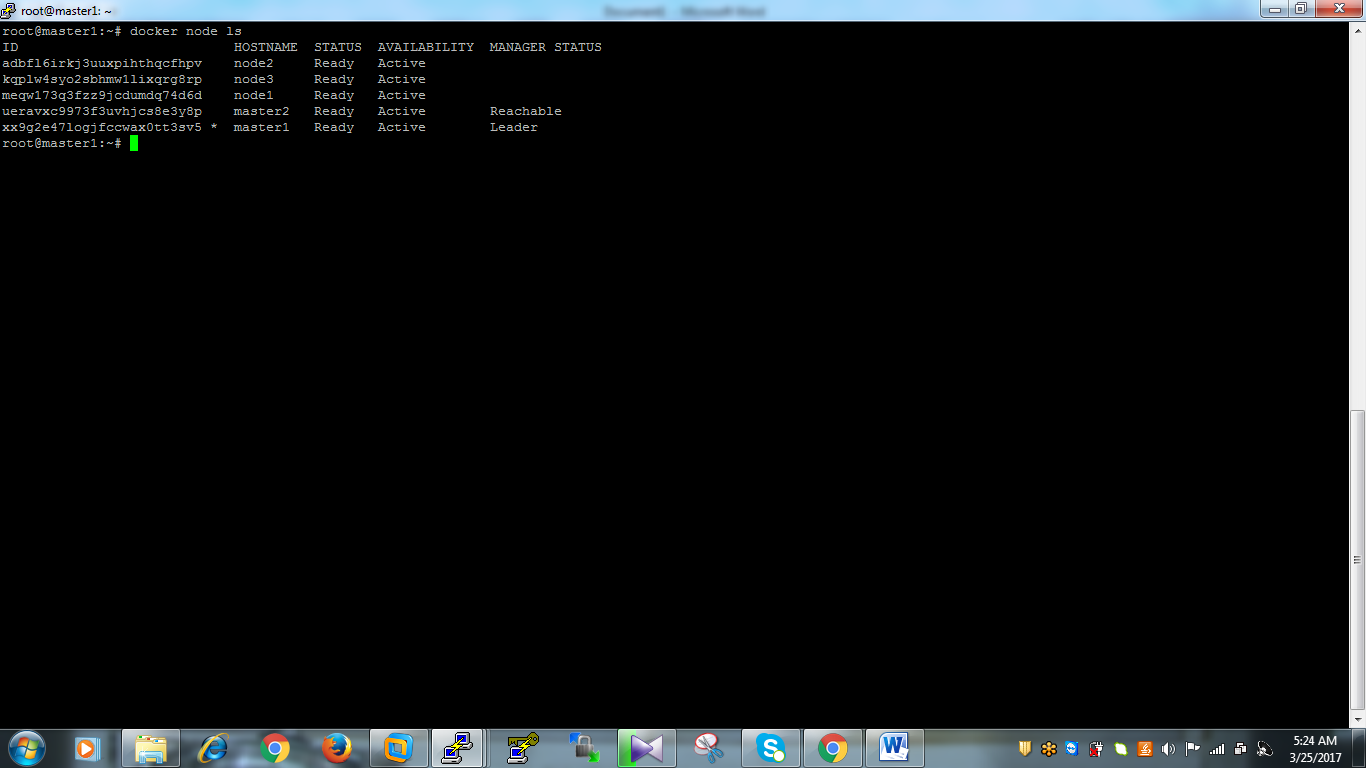




Then master2 is added to the managers list

Step7: The type the following command on both masters nodes to findout nodes list

# docker node ls

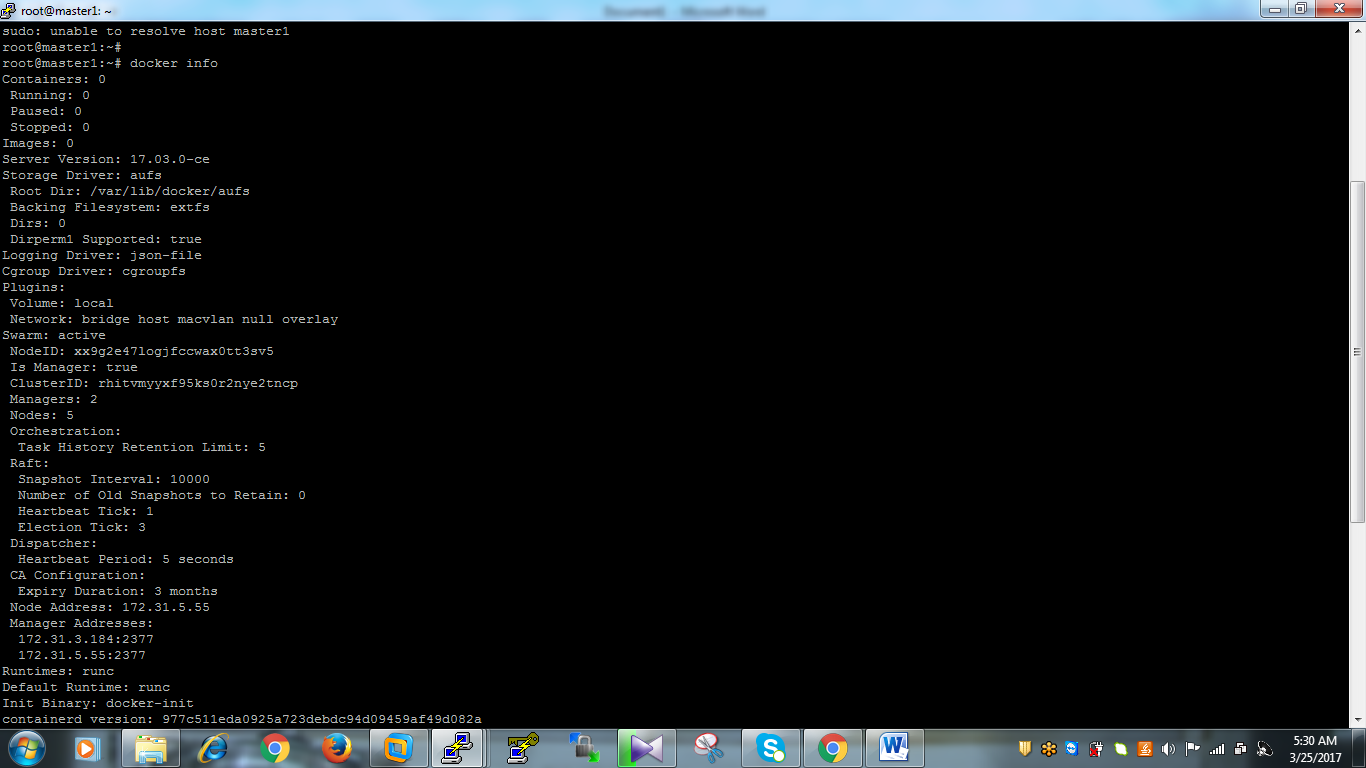


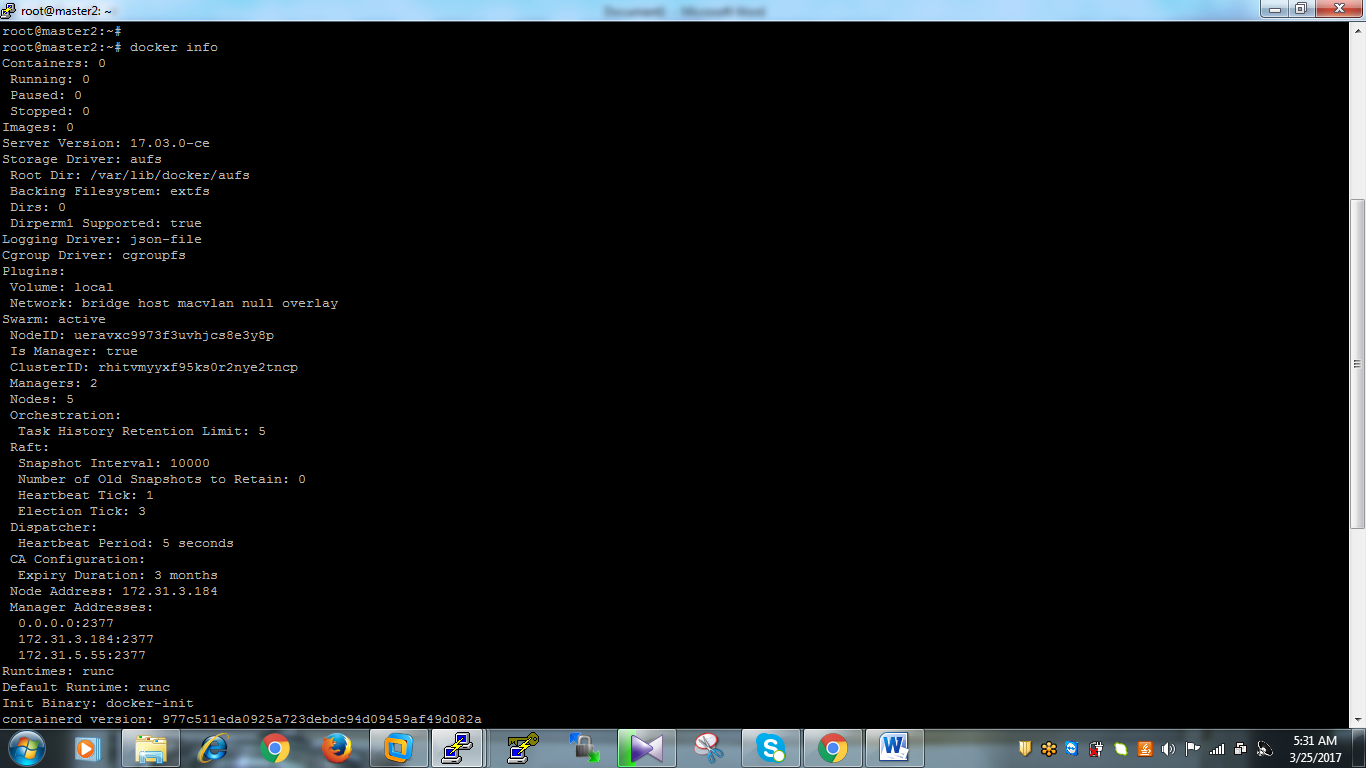


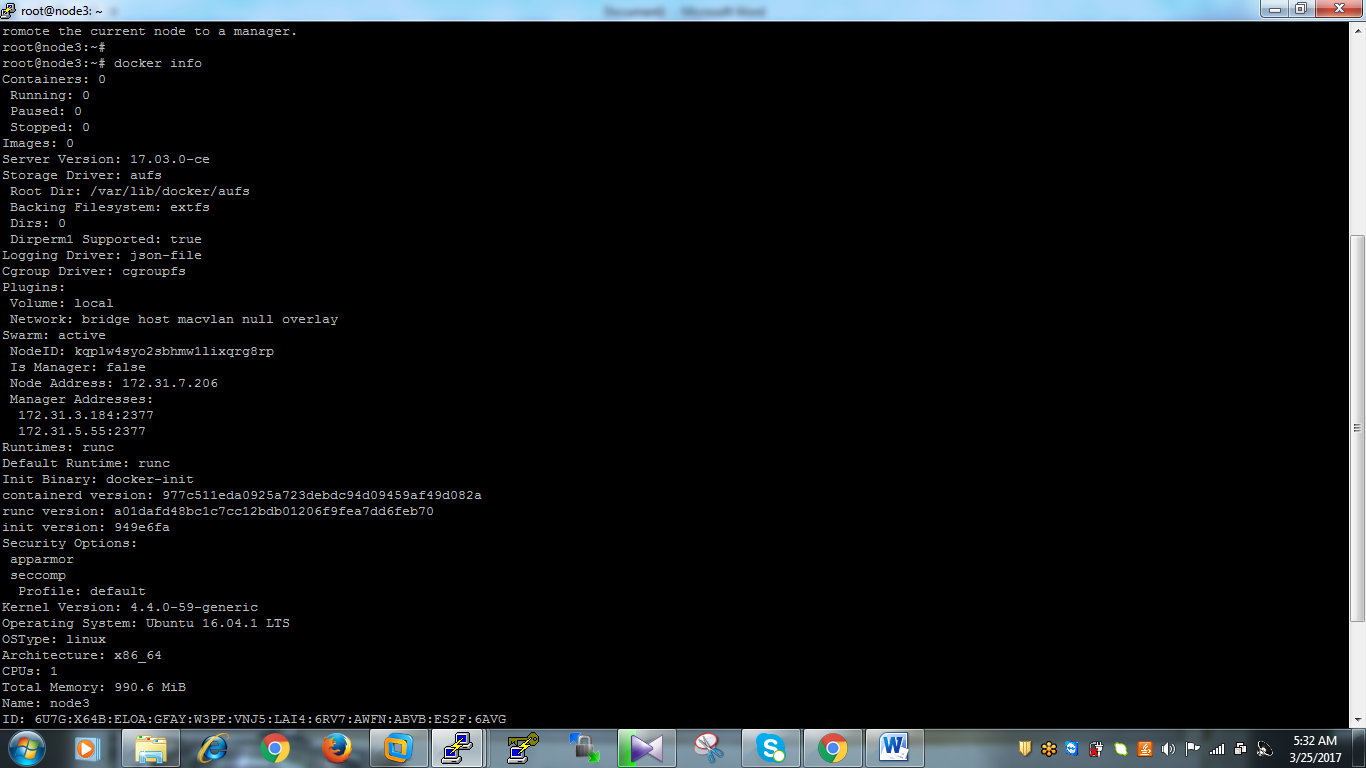
If you execute above in nodes rather than masters you will get error as output becoz nodes are not a masters. In above atleast one should be the Leader.

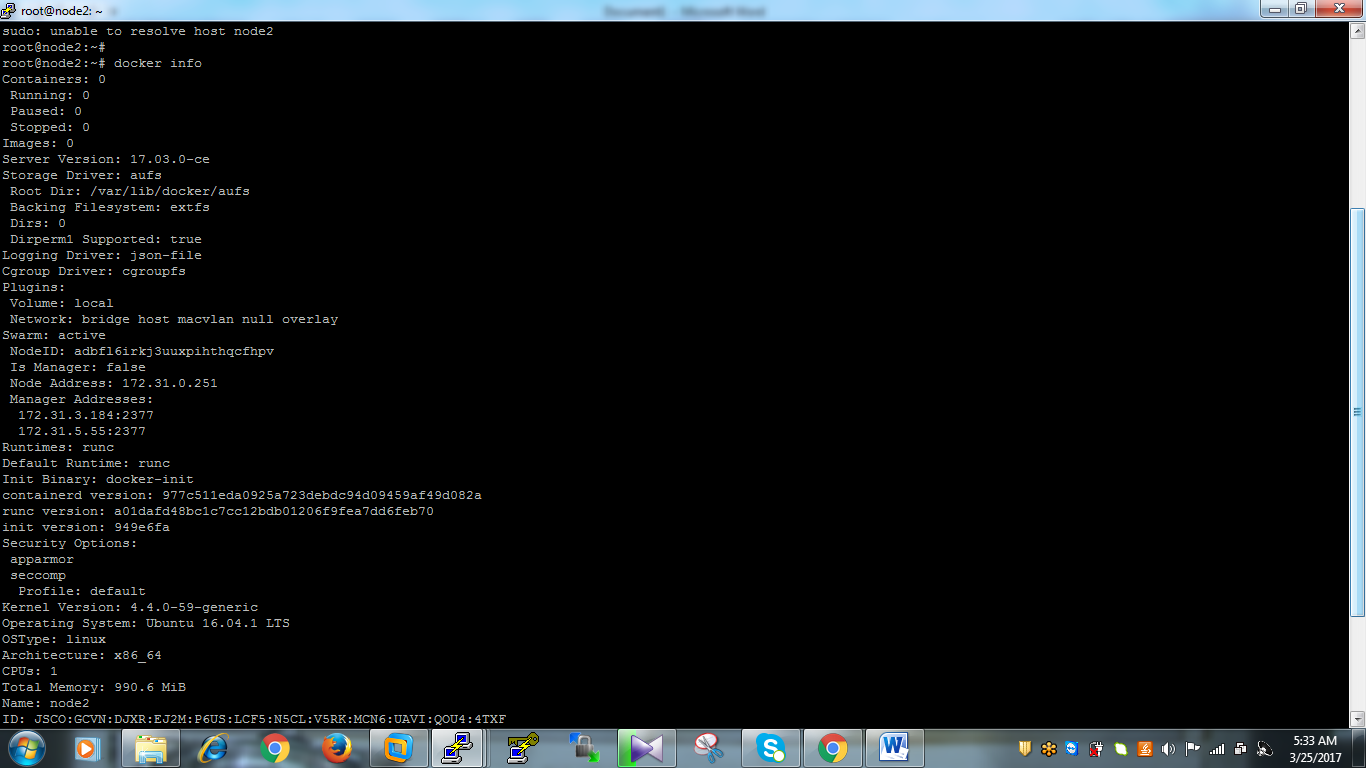
Step8: Then execute the following command in each and every machine to findout nodes information wether it is master or node

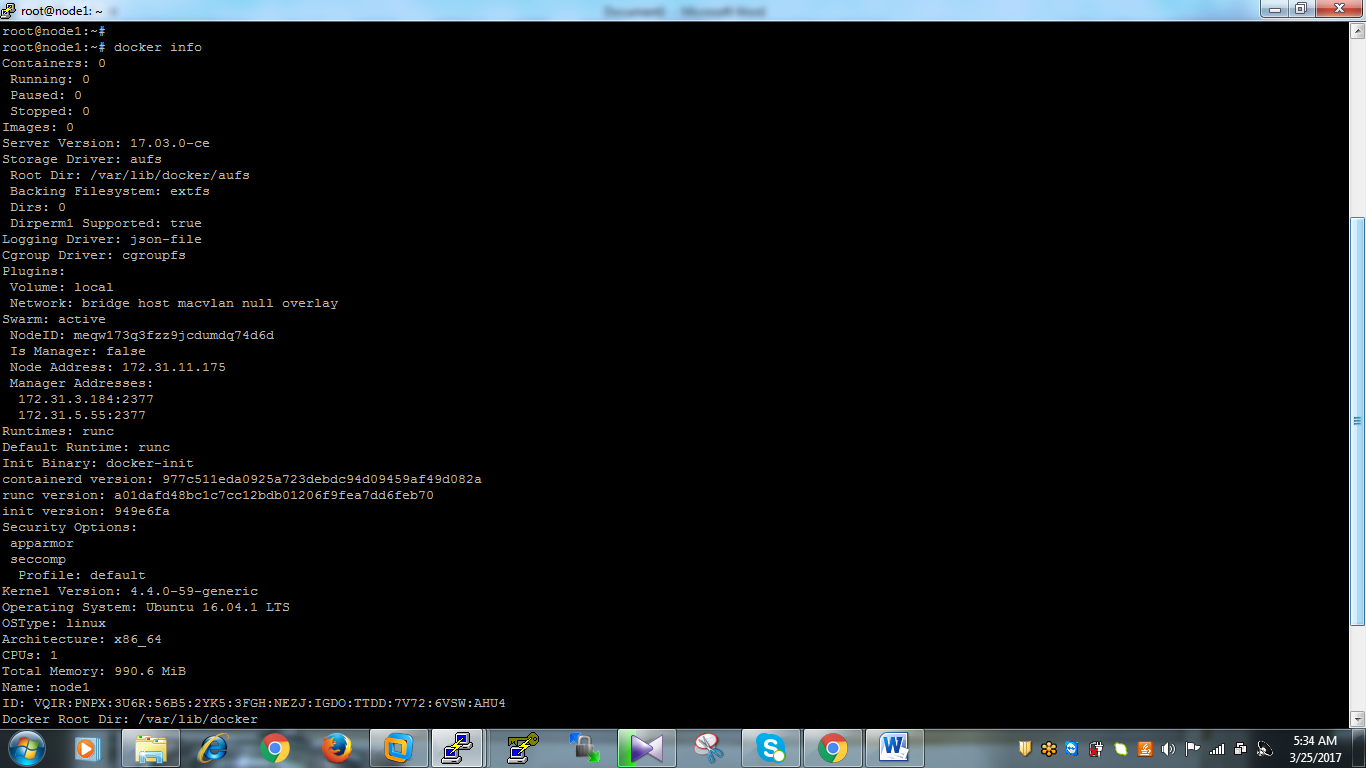
# docker info











Step9: Then find following commands carefully

If you want to add nodes in managers list use following command

# docker node promate node-name

If you want to remove remove nodes in managers list use following command

# docker node demote node-name

If you want remove nodes from swarm managers use following

# docker node rm node-name

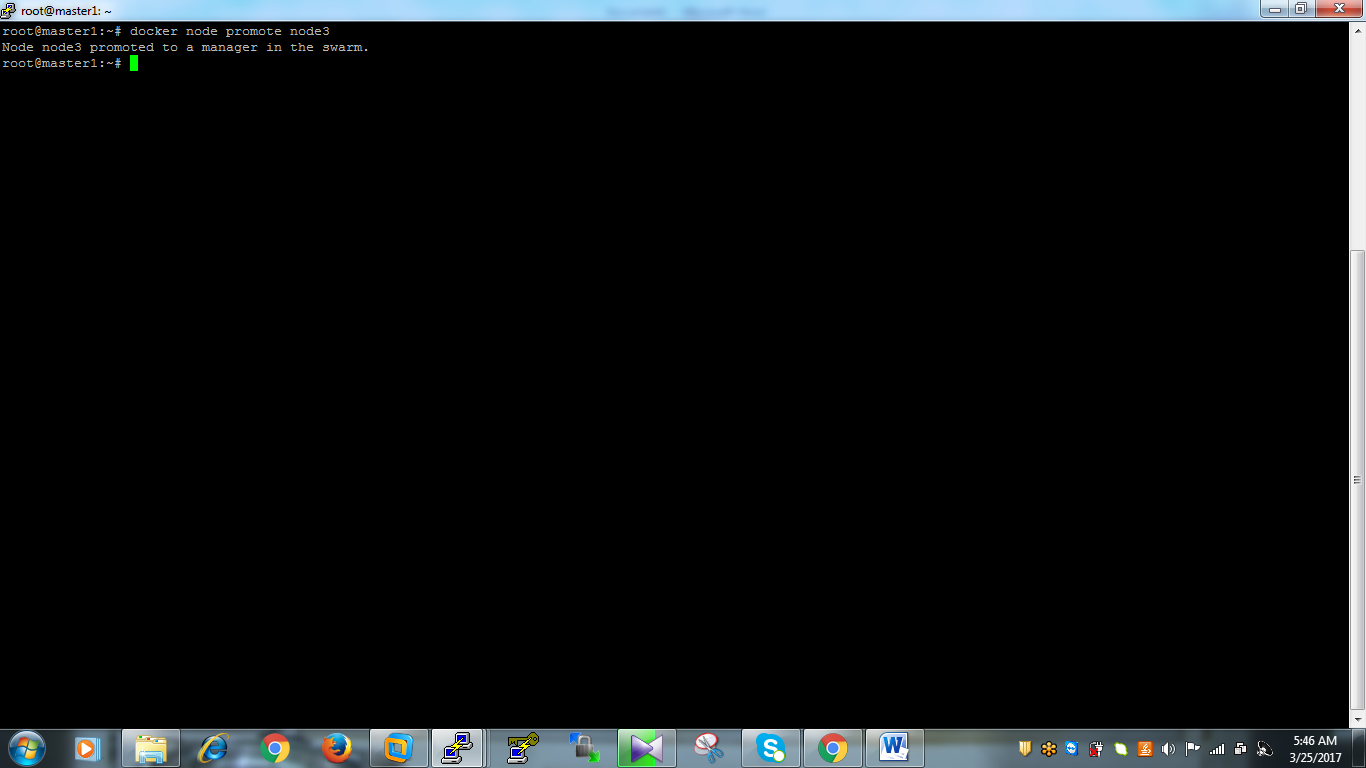
Note: The above commands are should execute on master Leader node only.

Step10: Here I’m trying to add node3 to the swarm managers list using following command.

# docker node promote node3

If you want to find node information as JSON use following

# docker node inspect self



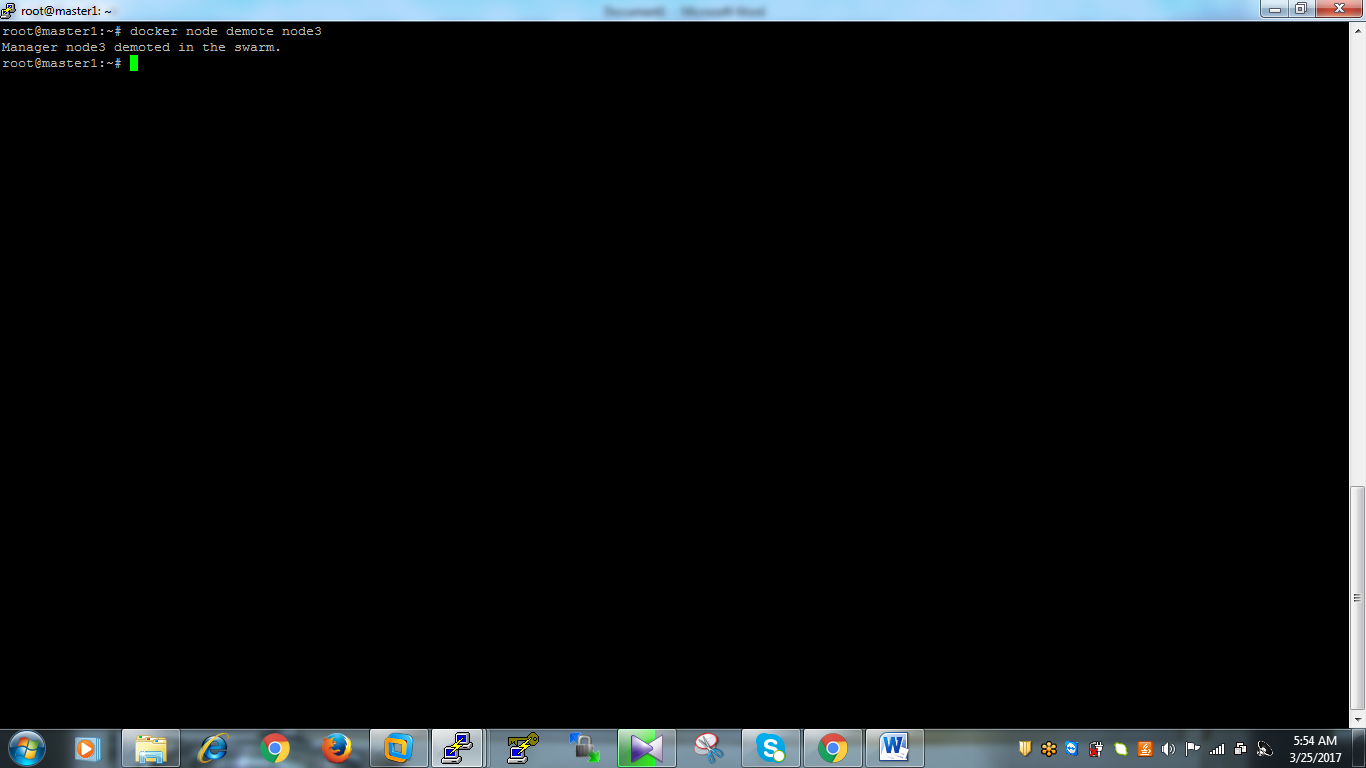
Step11: Then check the nodes list using following command in each and every node

# docker info



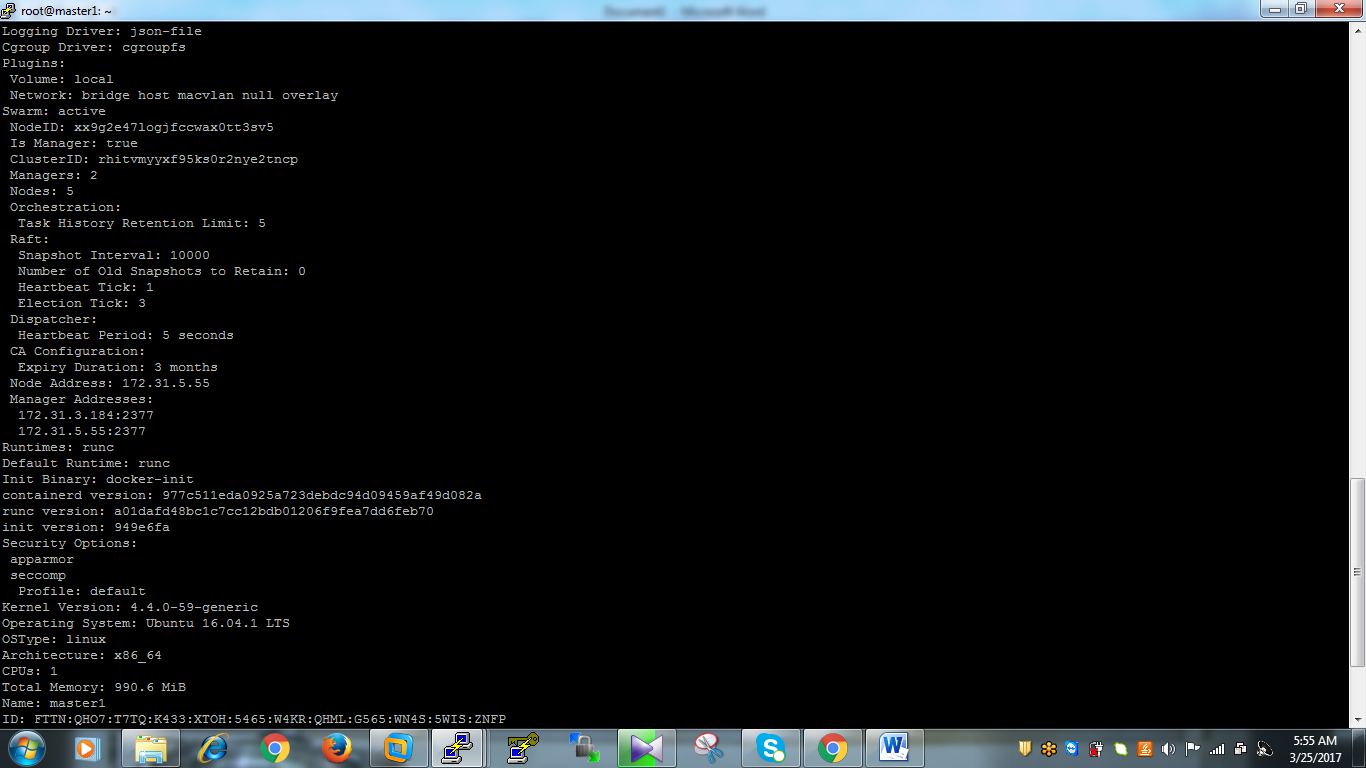
Step12: If you want to remove nodes3 from managers list you can use following command on masters leader node

# docker node demote node3



Then you can find again number of managers and nodes list using following command on each and every node

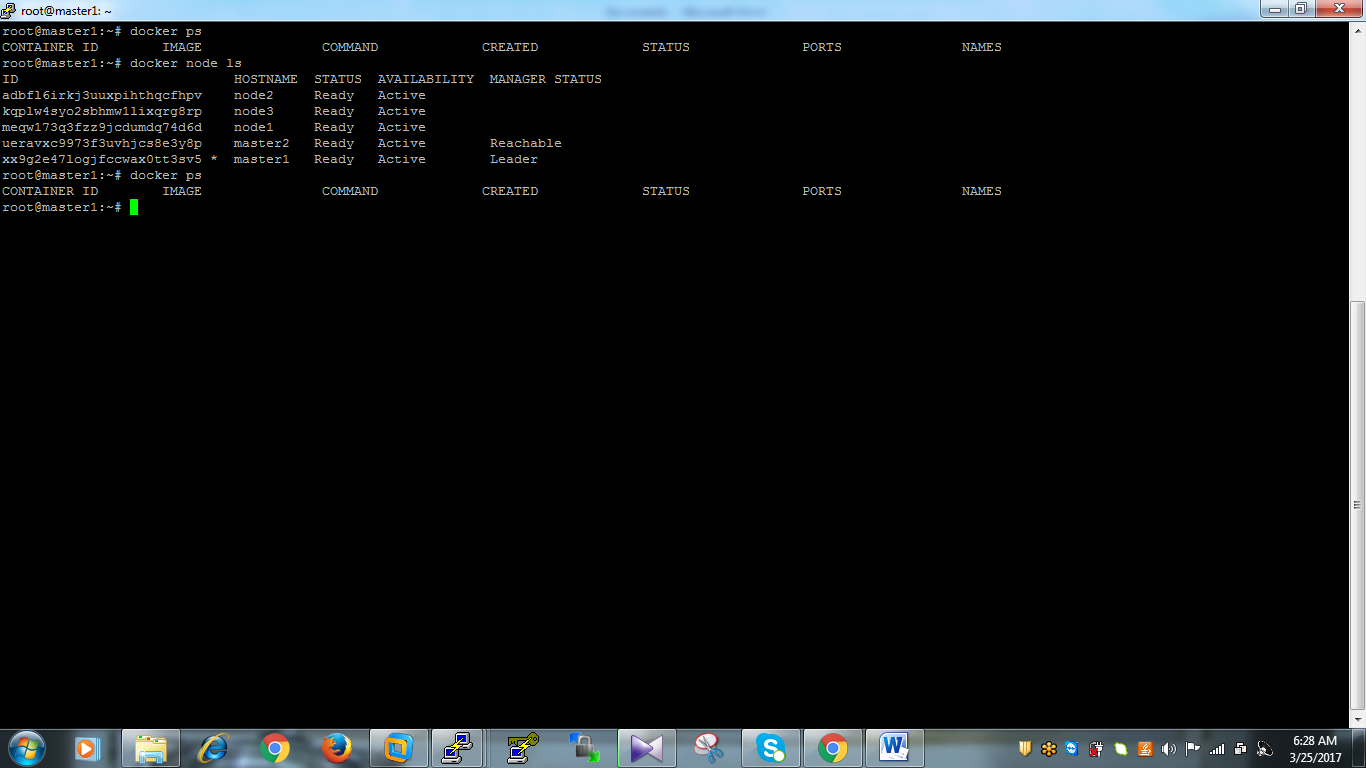
# docker info



Step13: Then you need to run services for running the containers in docker orchestration using following command

* In docker as of now we can see pull the images and create the images using Docker file and then run the containers on the images.
* But here we can pull and run the conatiners at a time
* If you shutdown the master1 node this is a Leader automatically master2 should be the Leader, If down the master2 also all containers will be in stopped state.
* As per RAFT Algarithem atleast two masters should be run -1/2 & -1/2+1
* Atleast two masters should be run
* Now we can see practically

Step14: If you find below snapshot master1 is the Leader and you don’t have any images



Step15: Then we are going to download the image and run the container containers at a time in all the nodes using the following sing command.

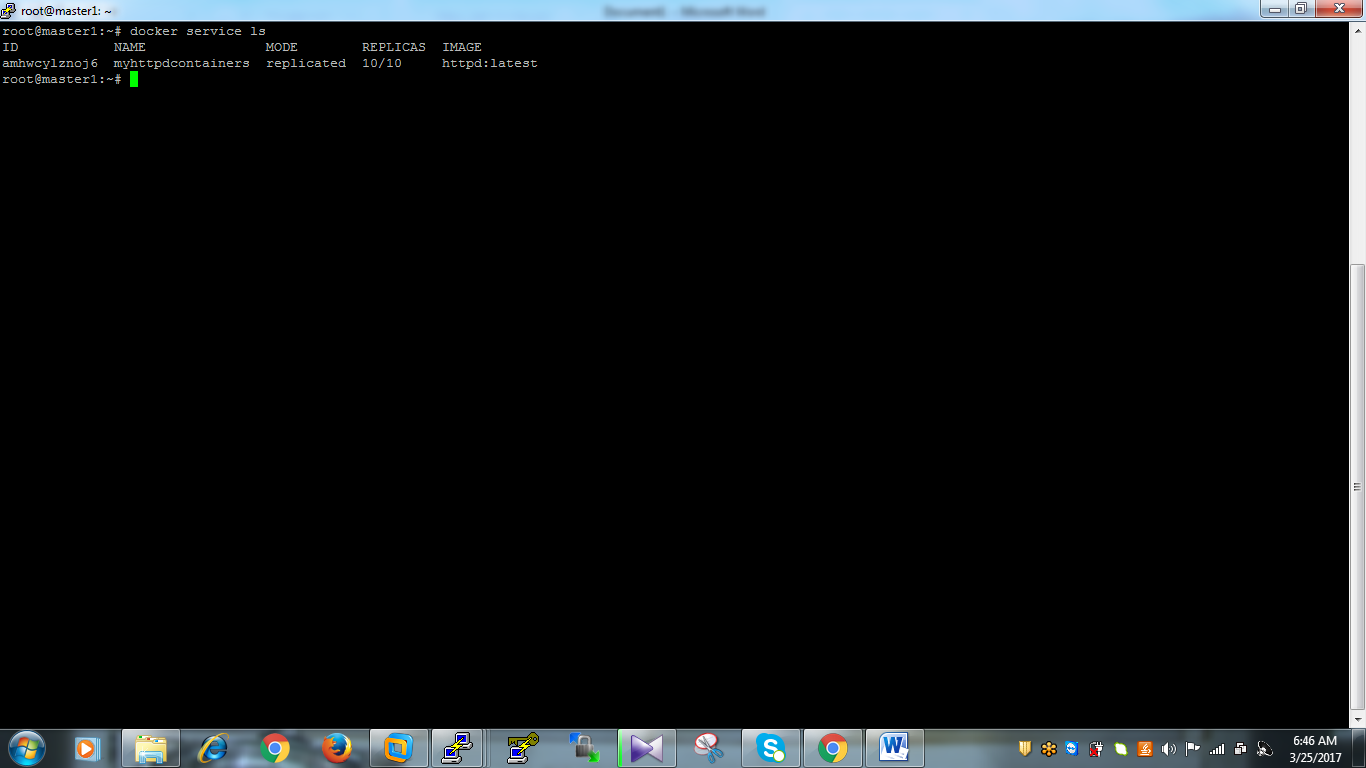
# docker service create --name myhttpdcontainers -- replicas 10 httpd

* Here services in the sense we are creating container services for httpd
* --name name it gives the name of container service this is our custom one
* --replicas are how many containers we want run the nodes simultaneously
* Httpd is we are downloading and running the containers on that image in all the nodes
* Here below you can observe we have downloaded the image and created containers and running containers on every node atleast two containers should run means we have given replicas 10 means we have 5 nodes it will distribute 10 replicas 10/2 two means runs every has two containers you can check on every node. You can see two nodes.



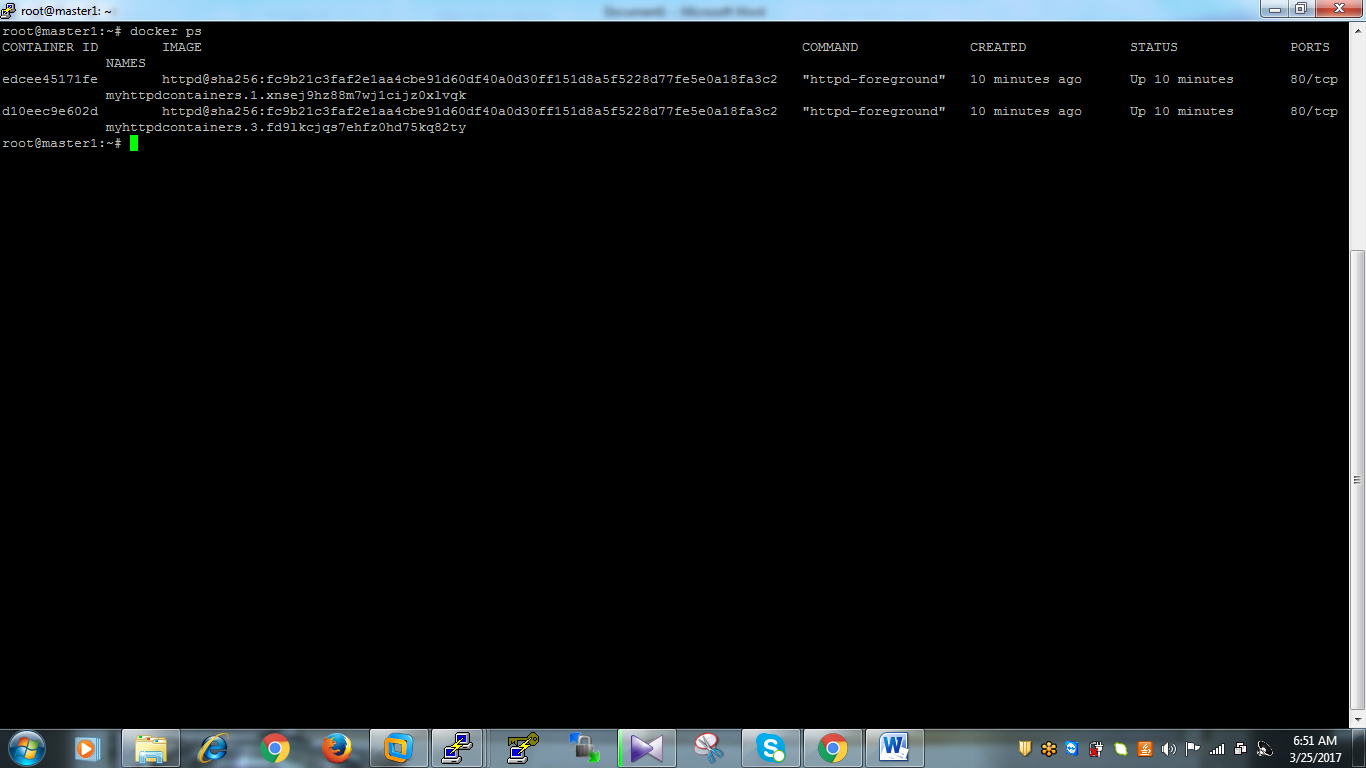
Step16: you can see number of services are running in you managers nodes using following command

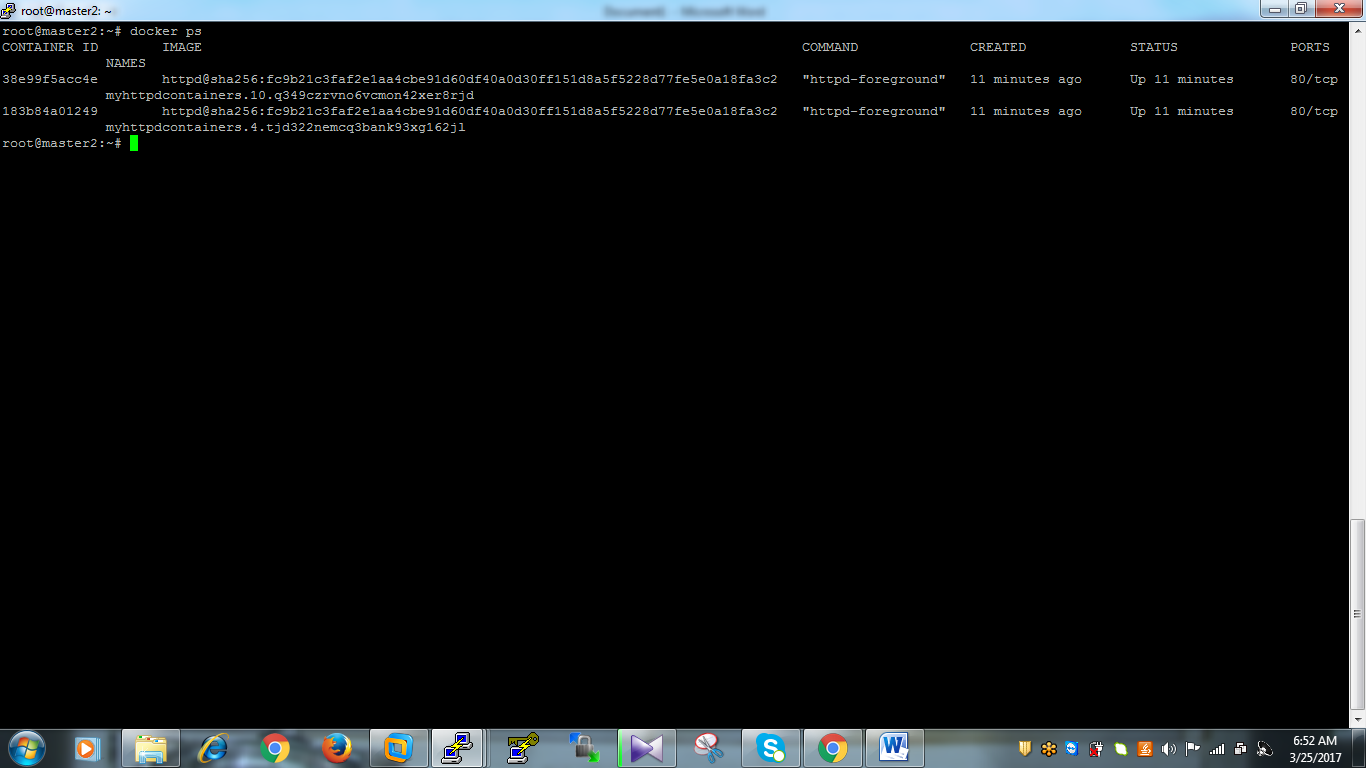
# docker service ls

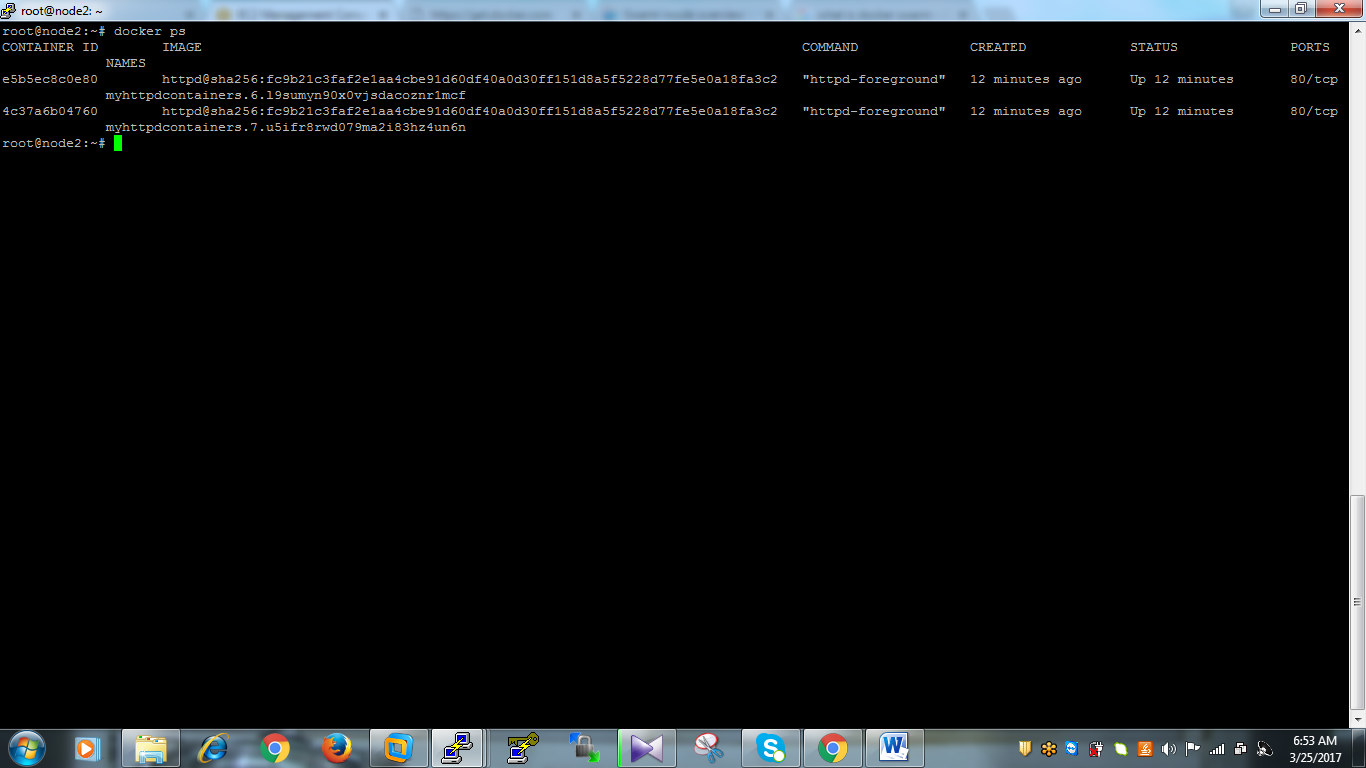


Step16: Then you can find number of containers are running on your each and every node by using following command.

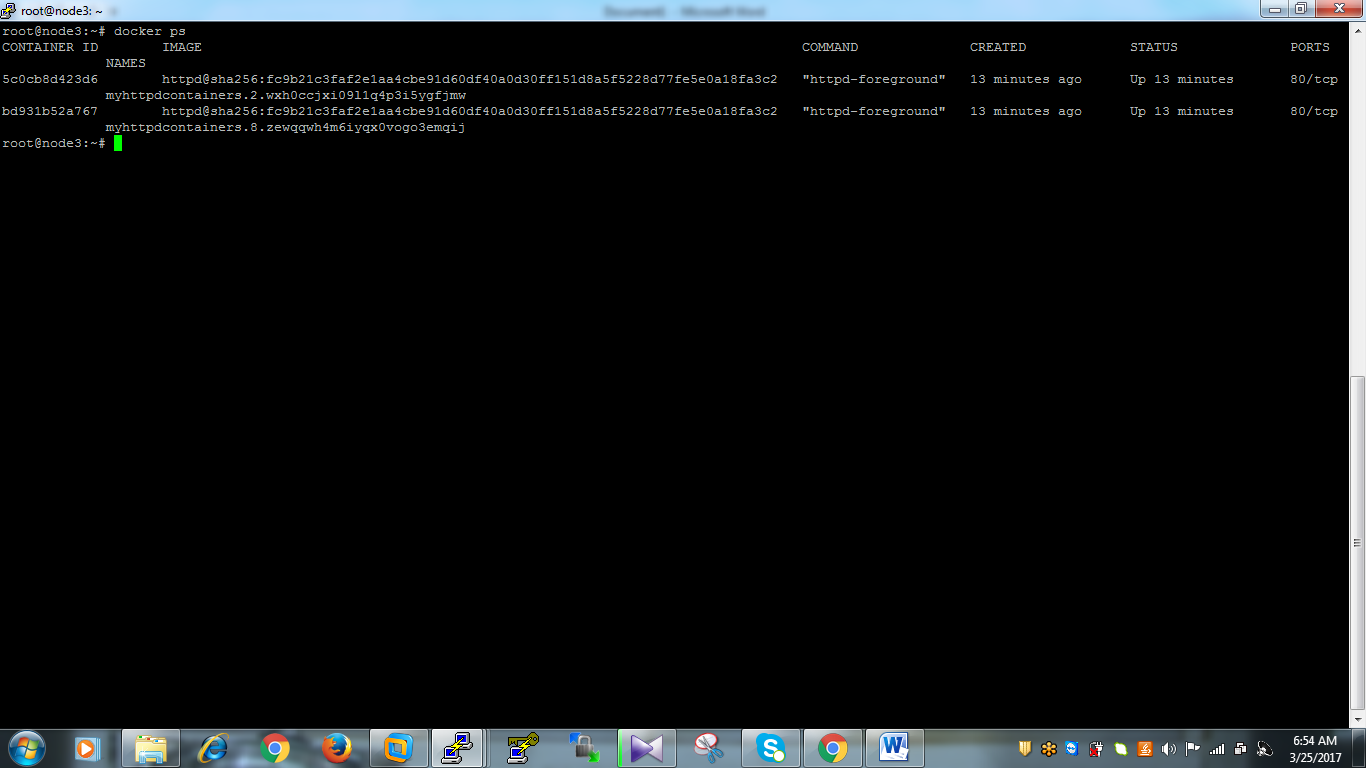
# docker ps





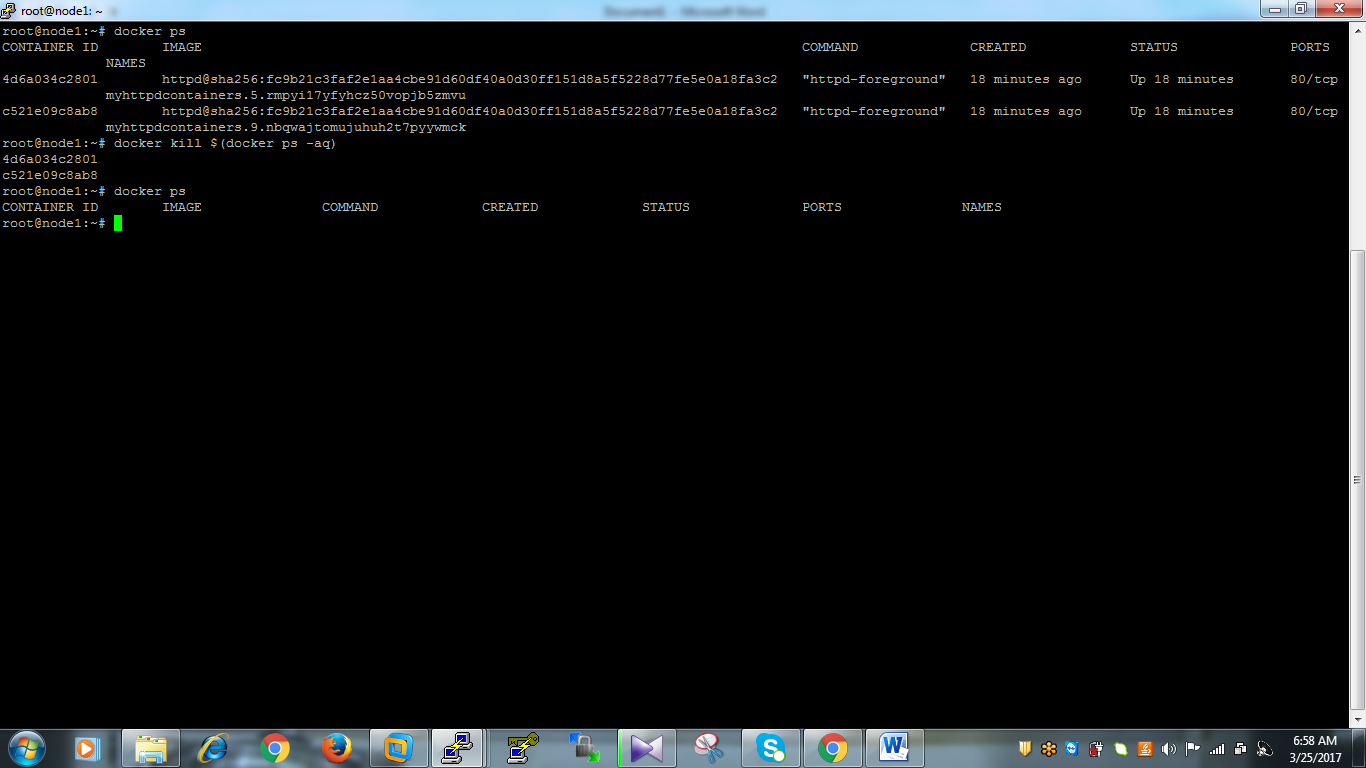






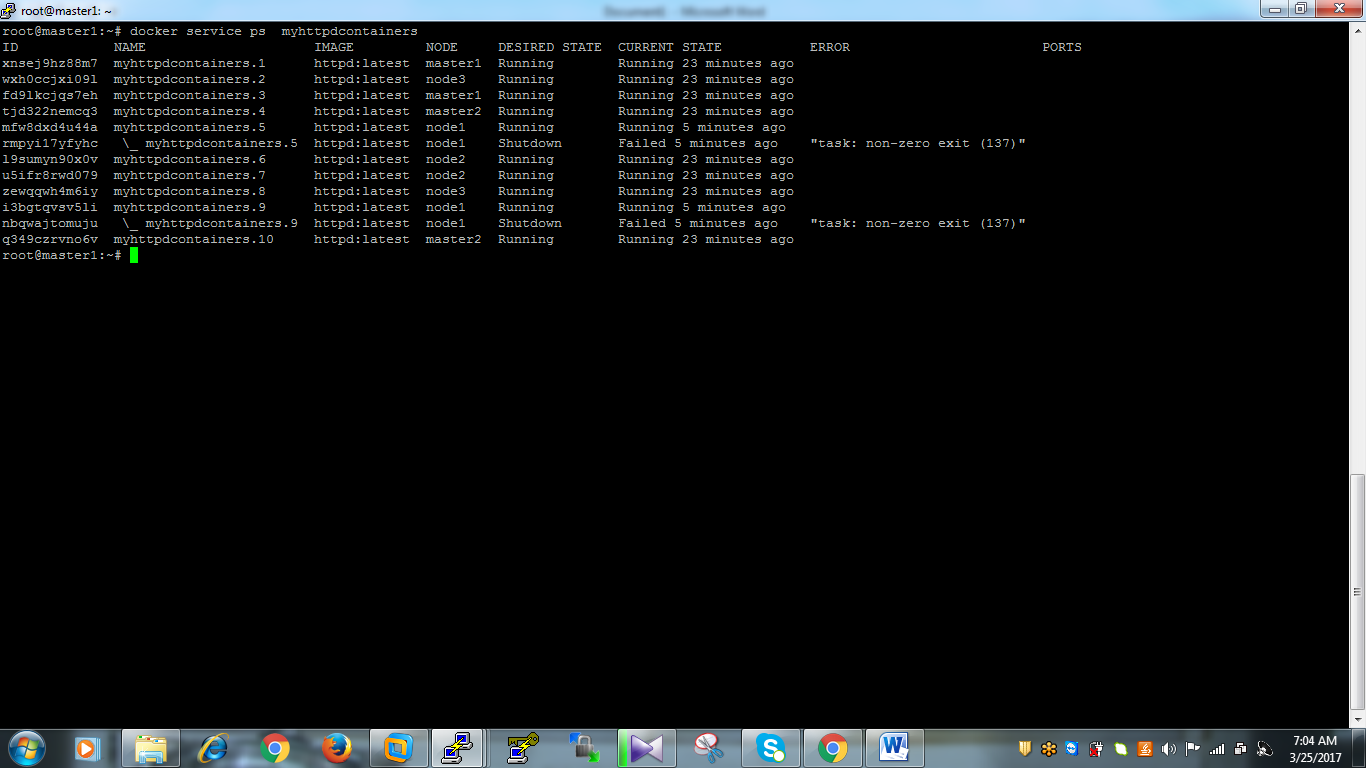
Step17: Then I’m going to bring down the containers in node1 then we can see automatically stopped containers should run automatically again. Use following command in node1 to bring down the containers

# docker kill $(docker ps -aq)



Then immediately go to master1 node or master2 node and check the running containers list using following command

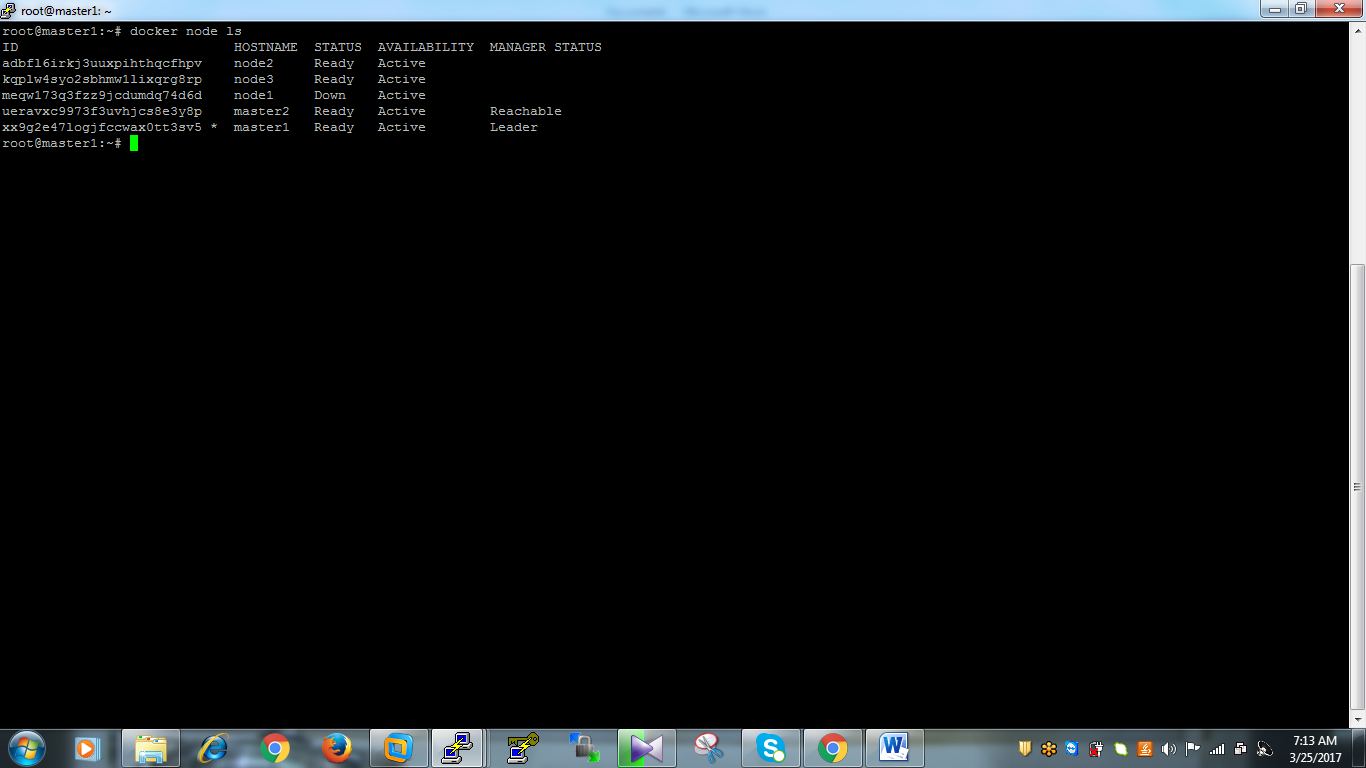
# docker service ps myhttpdcontainers



Above snapshot you can find each all containers are running the stopped containers are automatically up.

Step18: Here I’m bringing down node1 using following command

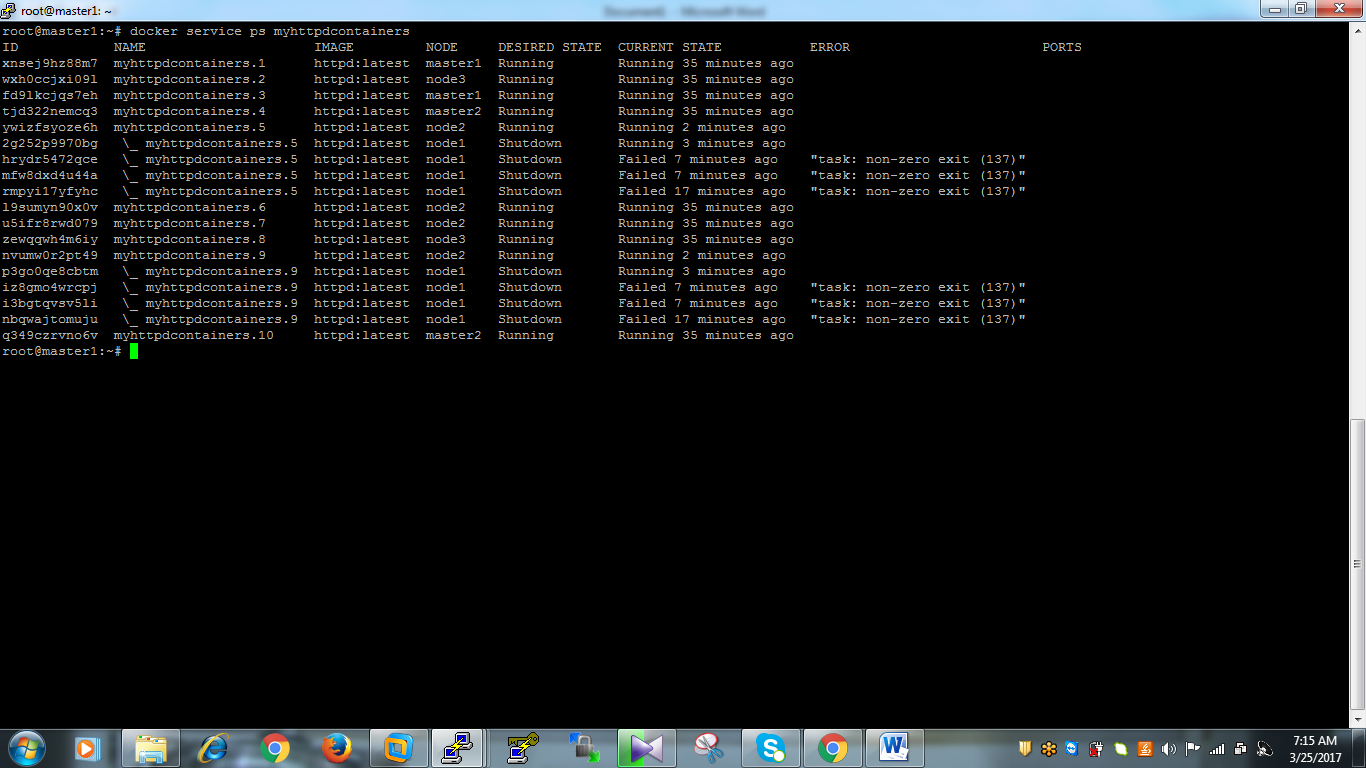
# shutdown –h now



* If you observe above image node Availability is active but status is down becoz we brought down the node1

Step19: If you observe following image we have brought down node1 based on our replicas 10 containers should be run distributed each and every node on two containers, but we stopped node1 it has running two containers but if you observe following image it’s from master1 it is running 10 containers eventhough our node1 is in down state. If you want to check all the running containers of httpd you can use following command

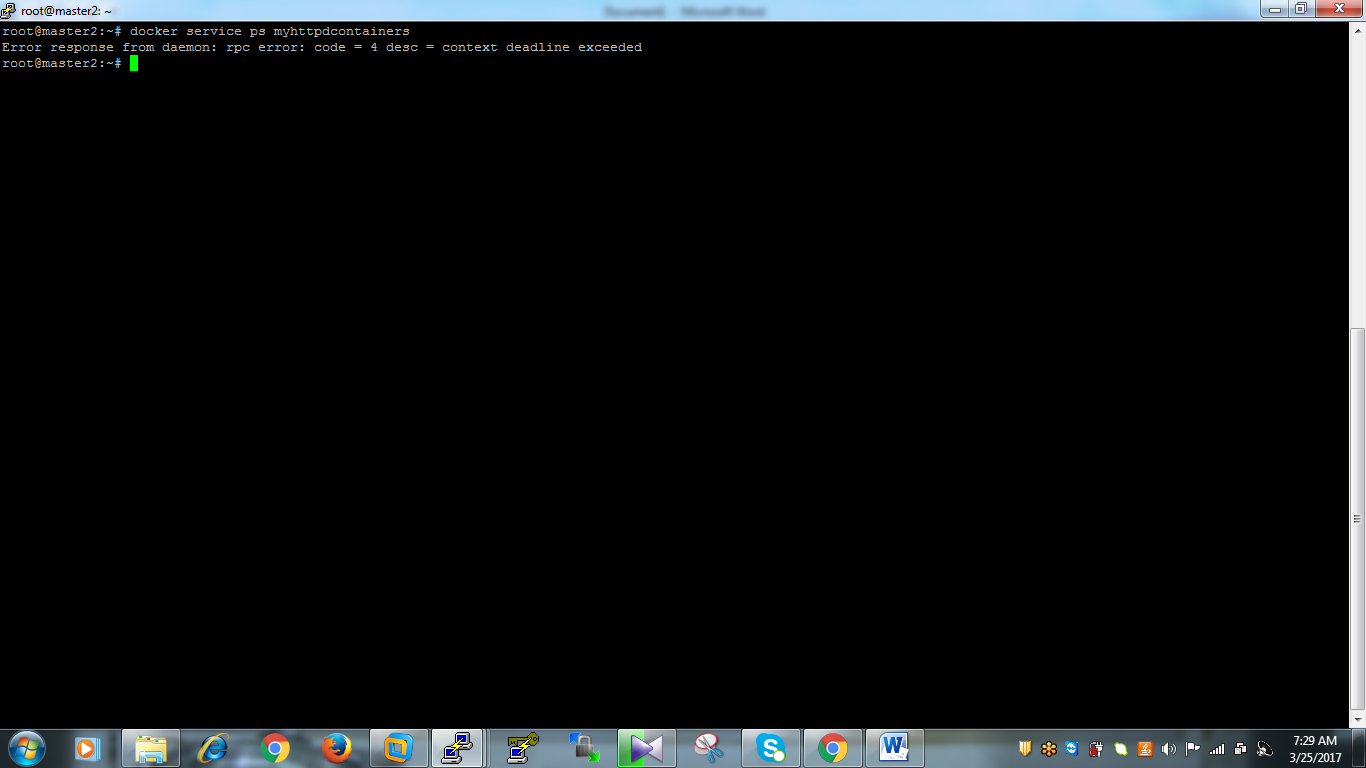
# docker service ps myhttpdcontainers



Step20: Here I’m bringing down master1 node , lets see what will happen

# shutdown –h now

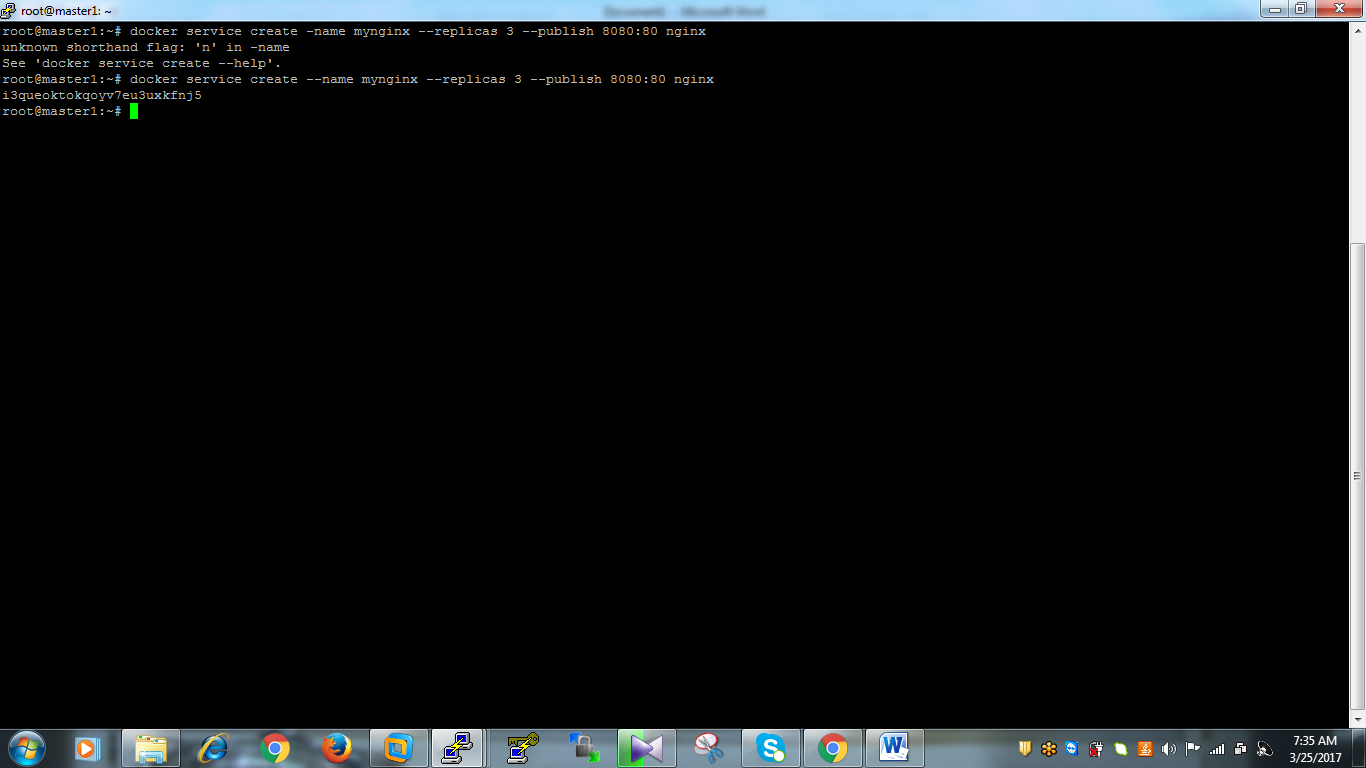
Step21: If you observe following images you can’t manage docker orchestration becoz there should be atleast two managers are in up



Step22: Now we are going create nginx publish localhost 8080 by using following command

# docker service create –name nginx –replicas 3 –publish 8080:80 nginx

If you observe following image

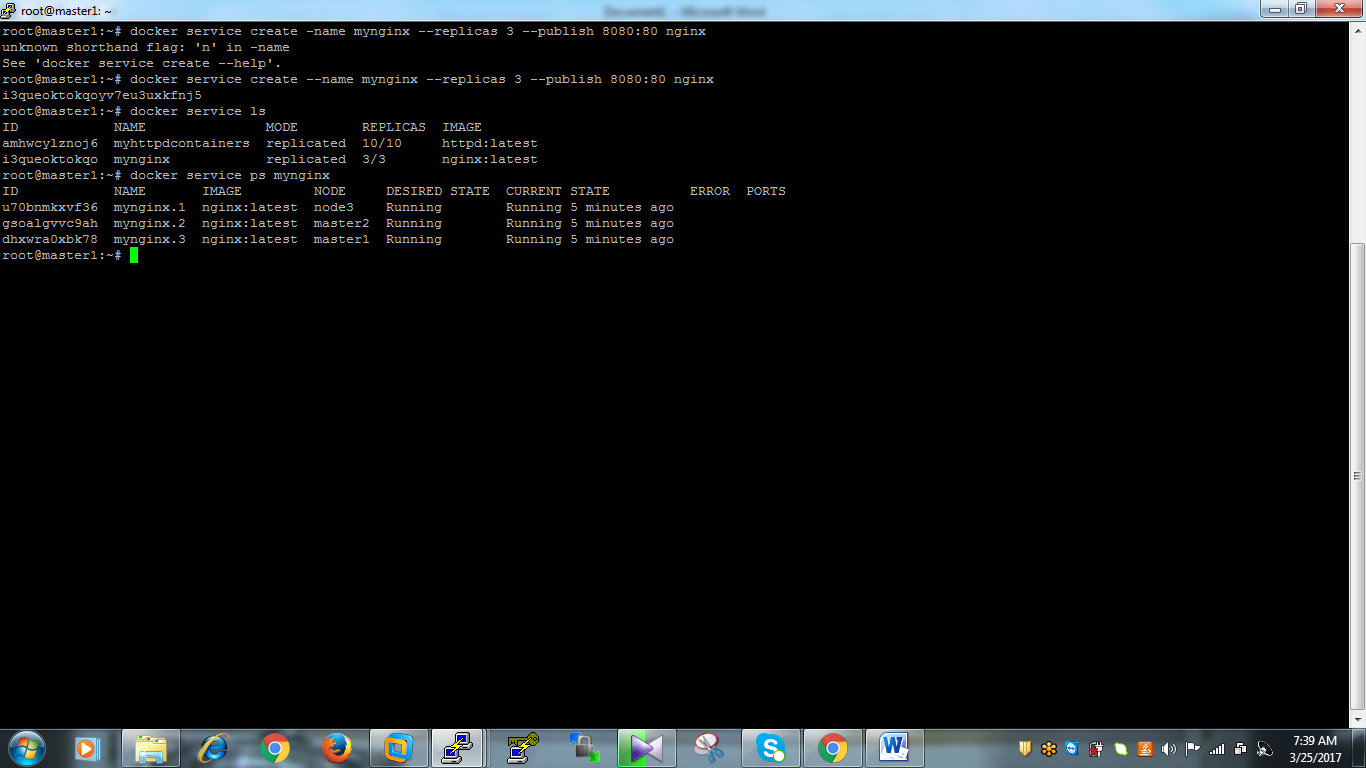


Then check number services are running using following command

# docker service ls



# docker service ps mynginx



If you check your nginx localhostin nodes side you will find using following command

# curl localhost:8080