Module 7: Assignment

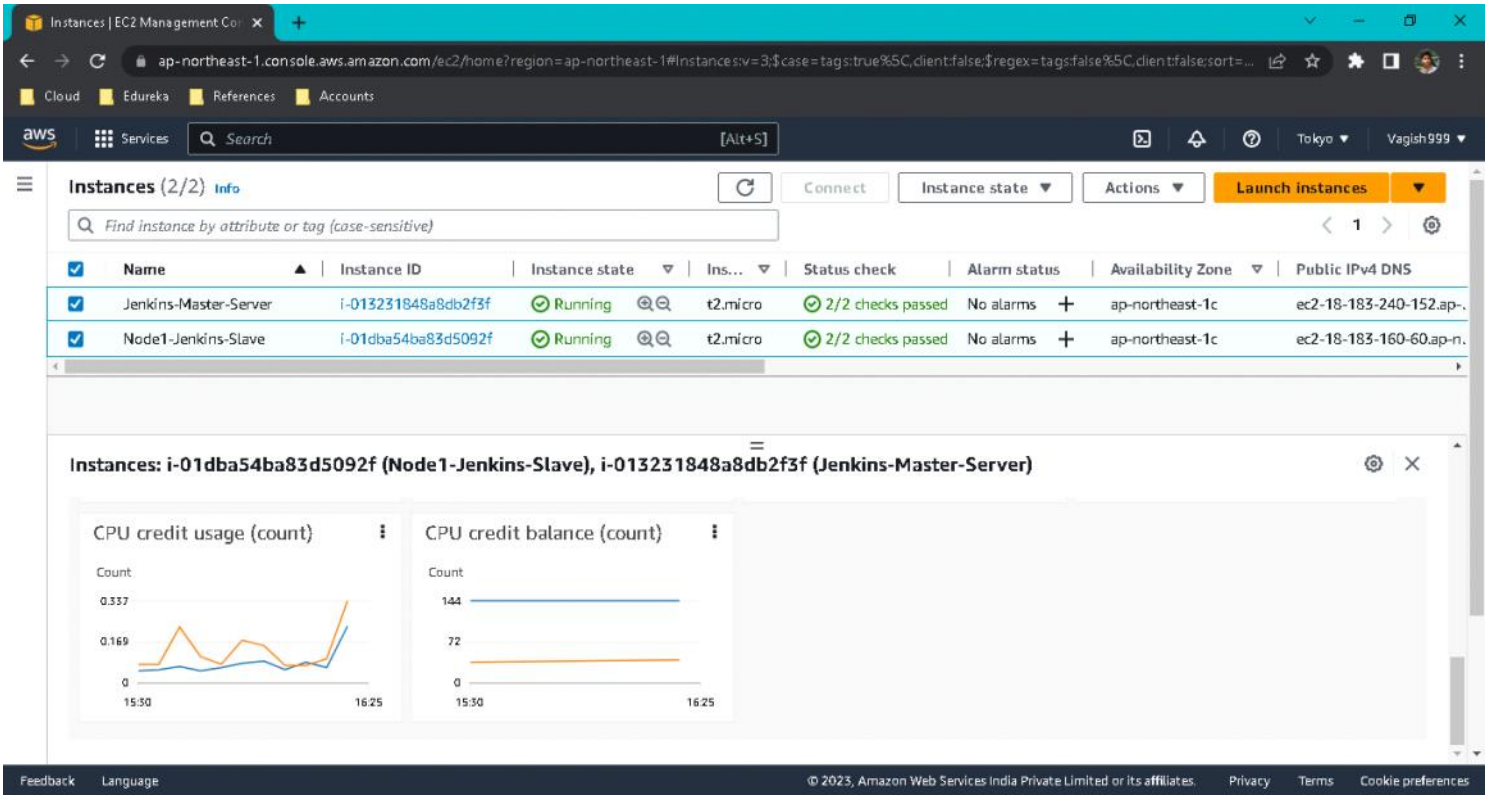
Docker is a self-contained unit of software that can be delivered on any server having its own isolated process and has its own file system. We would want to further explore Docker before we choose it for the deployment of our microservices. Basically, we would like to check the feasibility of the deployment and ease of use to make an informed decision. So perform the below-mentioned tasks and share your findings.

1. Install Docker on a linux based machine (preferable Ubuntu-20).
2. Check Docker version
3. Launch Docker Hello World example
4. List docker images

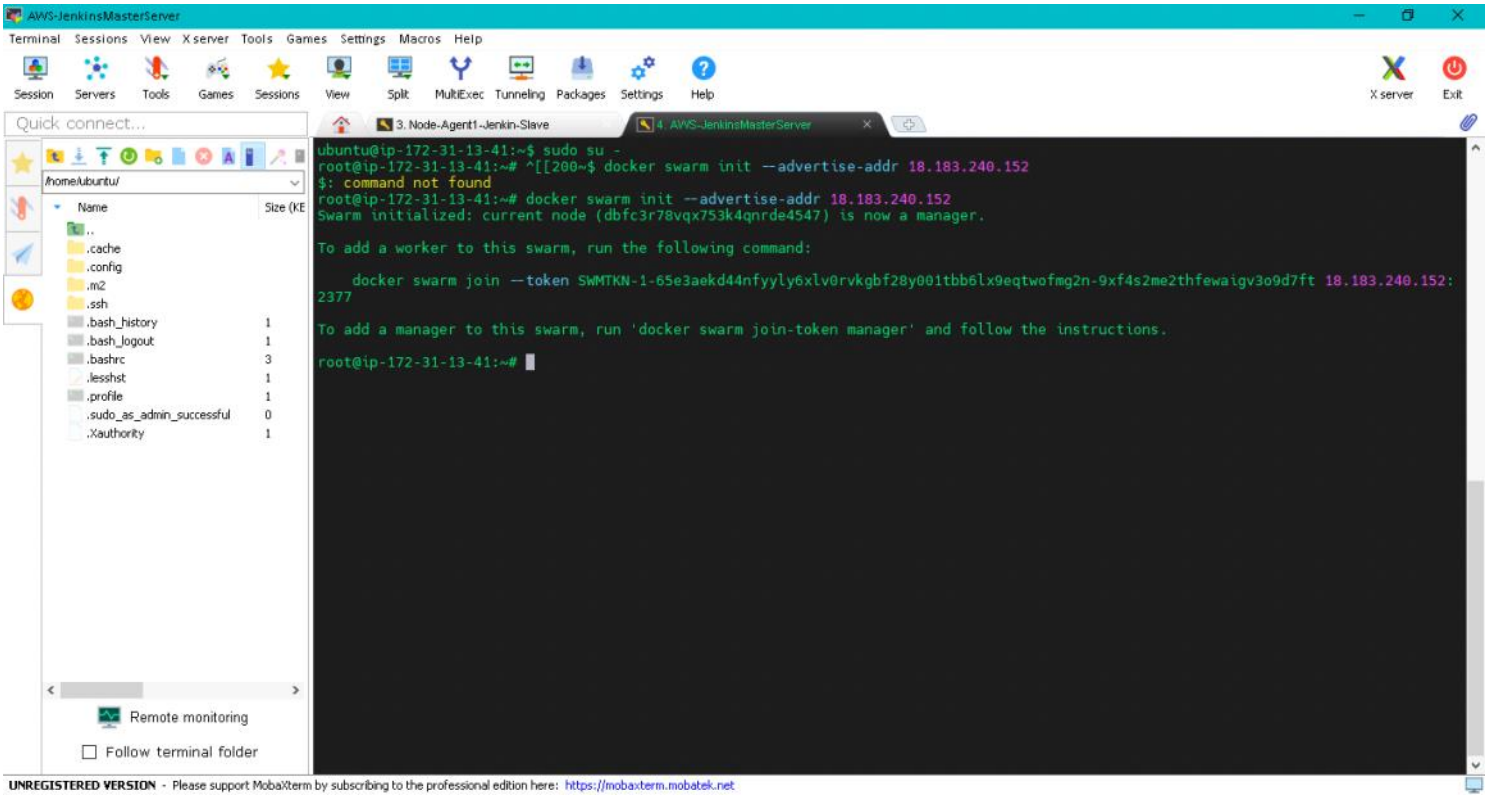
Hint: Use a Linux machine for installing Docker.

Solution:
-> Used AWS to get two VM instances used as Manager and Worked Node for Swarm cluster.
-> Used GIT checkout to get Edureka Instructor Shared code within both the Nodes.

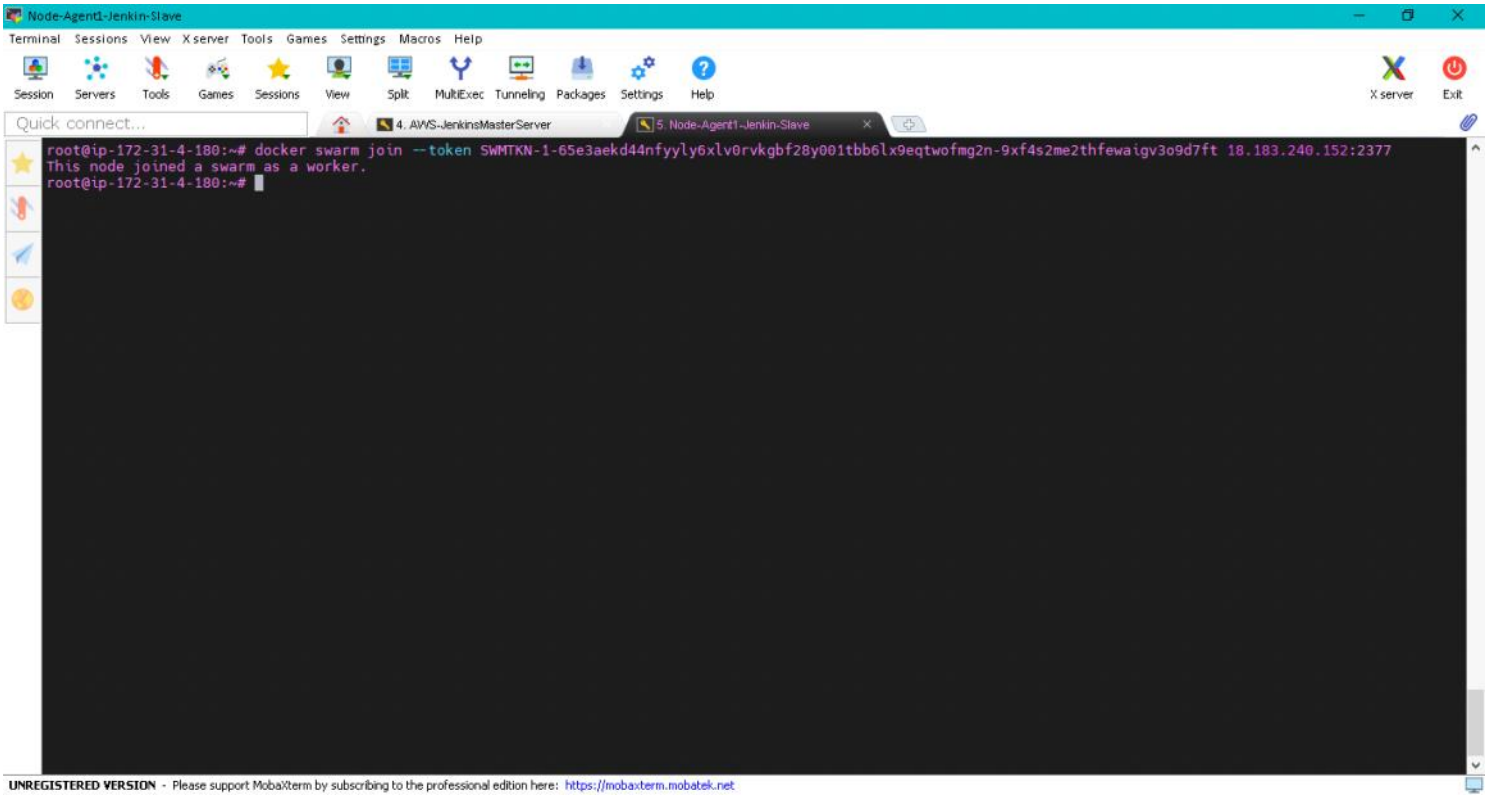
1. AWS Nodes Created : Master and Slave



2. Initiating Docker Swarm in Manager Node



3. Joining Worker Node in Docker Swarm Cluster



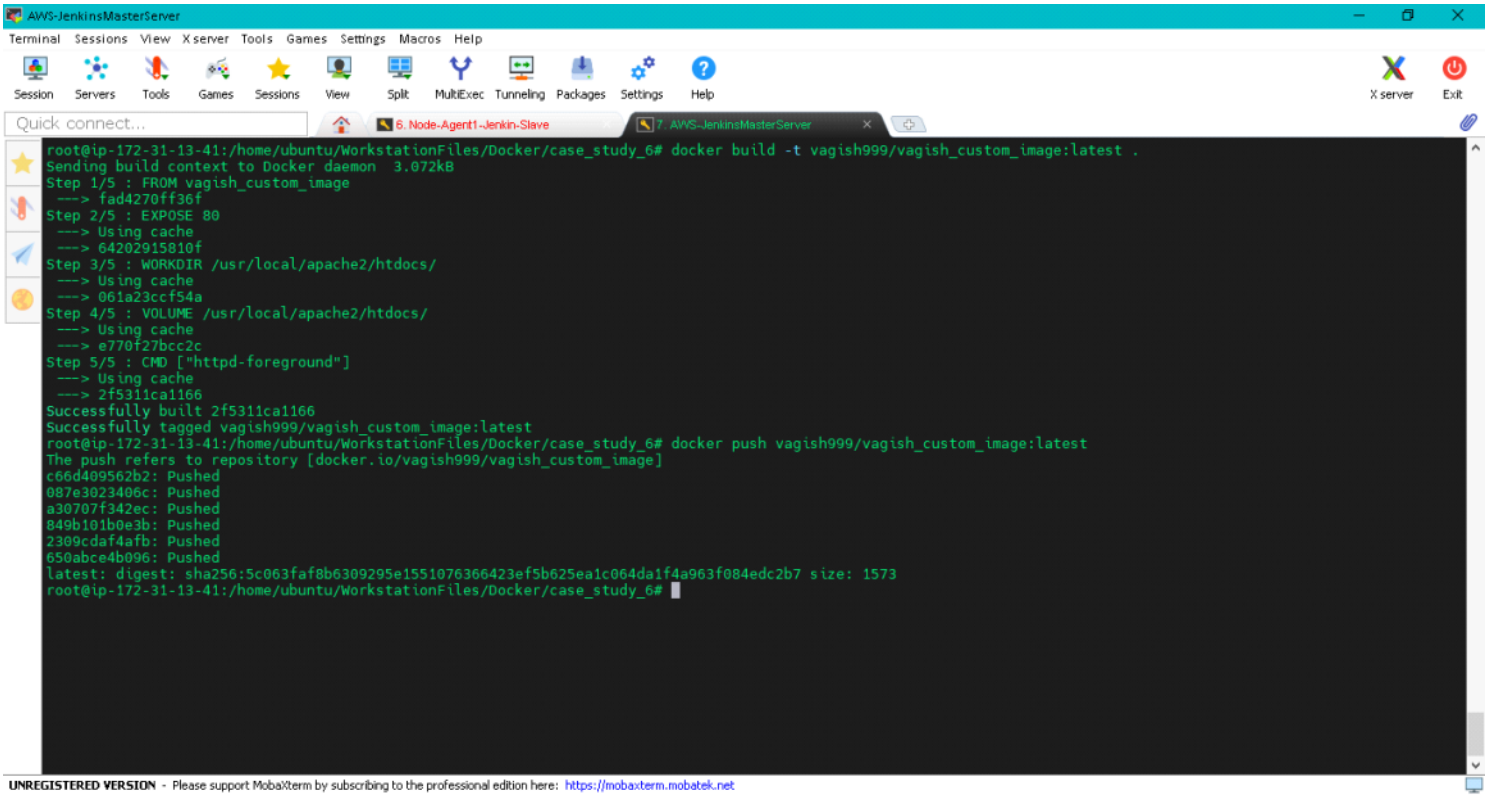
4. Both the nodes are linked now.

```
root@ip-172-31-13-41:~# docker node
demote   inspect  ls       promote  ps       rm       update
root@ip-172-31-13-41:~# docker node ls
ID                            HOSTNAME           STATUS    AVAILABILITY   MANAGER STATUS   ENGINE VERSION
zloe2e3vhhofu9ybnhvid484s     ip-172-31-4-180    Ready     Active                          20.10.21
dbfc3r78vqx753k4qnrde4547 *   ip-172-31-13-41    Ready     Active         Leader           20.10.21
root@ip-172-31-13-41:~#
```

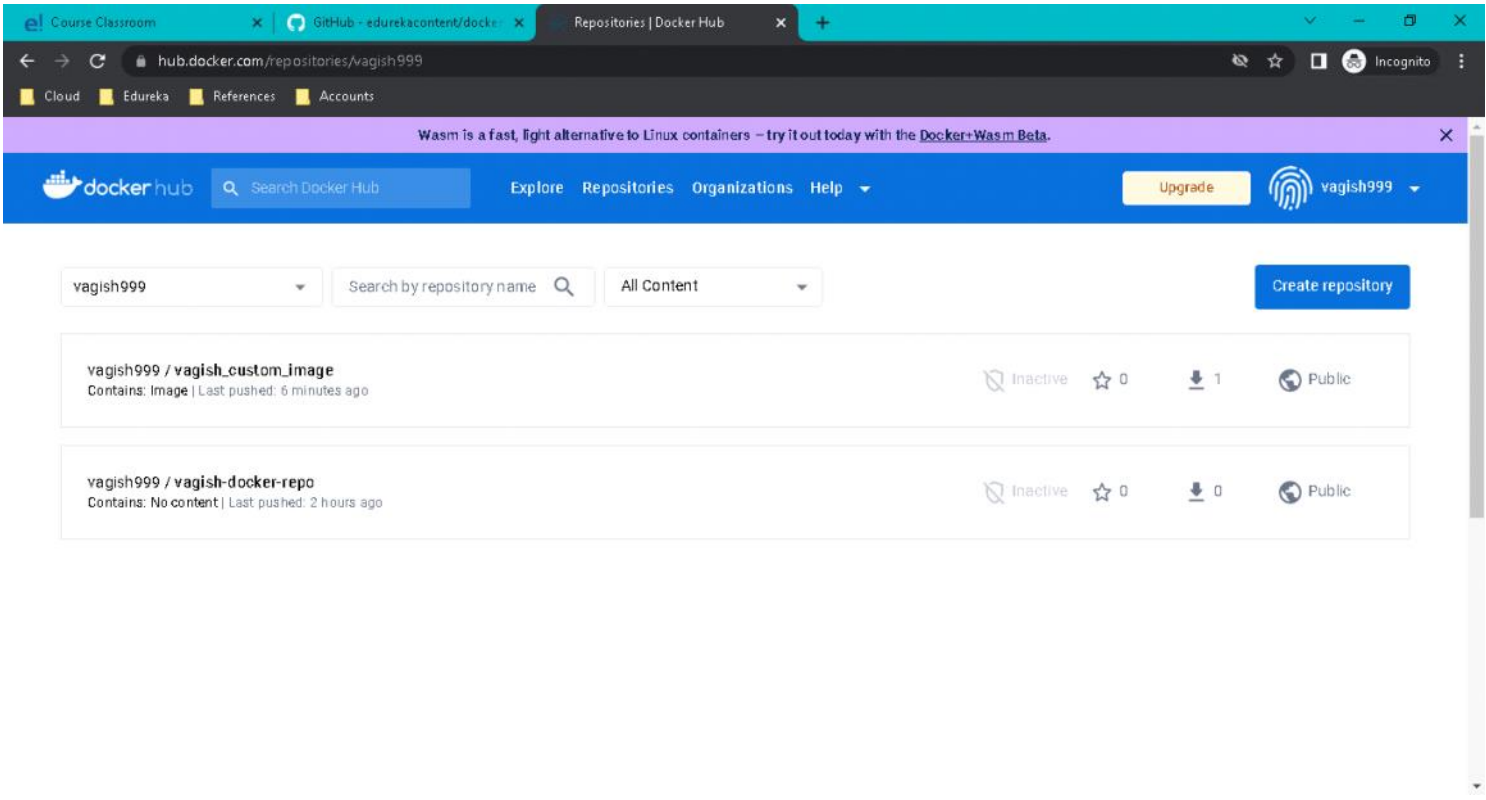
5. Creating Dockerfile to initiate server for website using custom Docker image and external Volume exposing Port 80

```
root@ip-172-31-13-41:/home/ubuntu/WorkstationFiles/Docker/case_study_6# cat Dockerfile
FROM vagish_custom_image

# Expose port 80 for HTTP traffic
EXPOSE 80

#COPY /home/ubuntu/WorkstationFiles/Docker/sampleVol01/. /usr/local/apache2/htdocs/

# Set the working directory to the Apache document root
WORKDIR /usr/local/apache2/htdocs/

# Define a volume for the website data
VOLUME /usr/local/apache2/htdocs/

# Start Apache in the foreground
CMD ["httpd-foreground"]
root@ip-172-31-13-41:/home/ubuntu/WorkstationFiles/Docker/case_study_6#
```

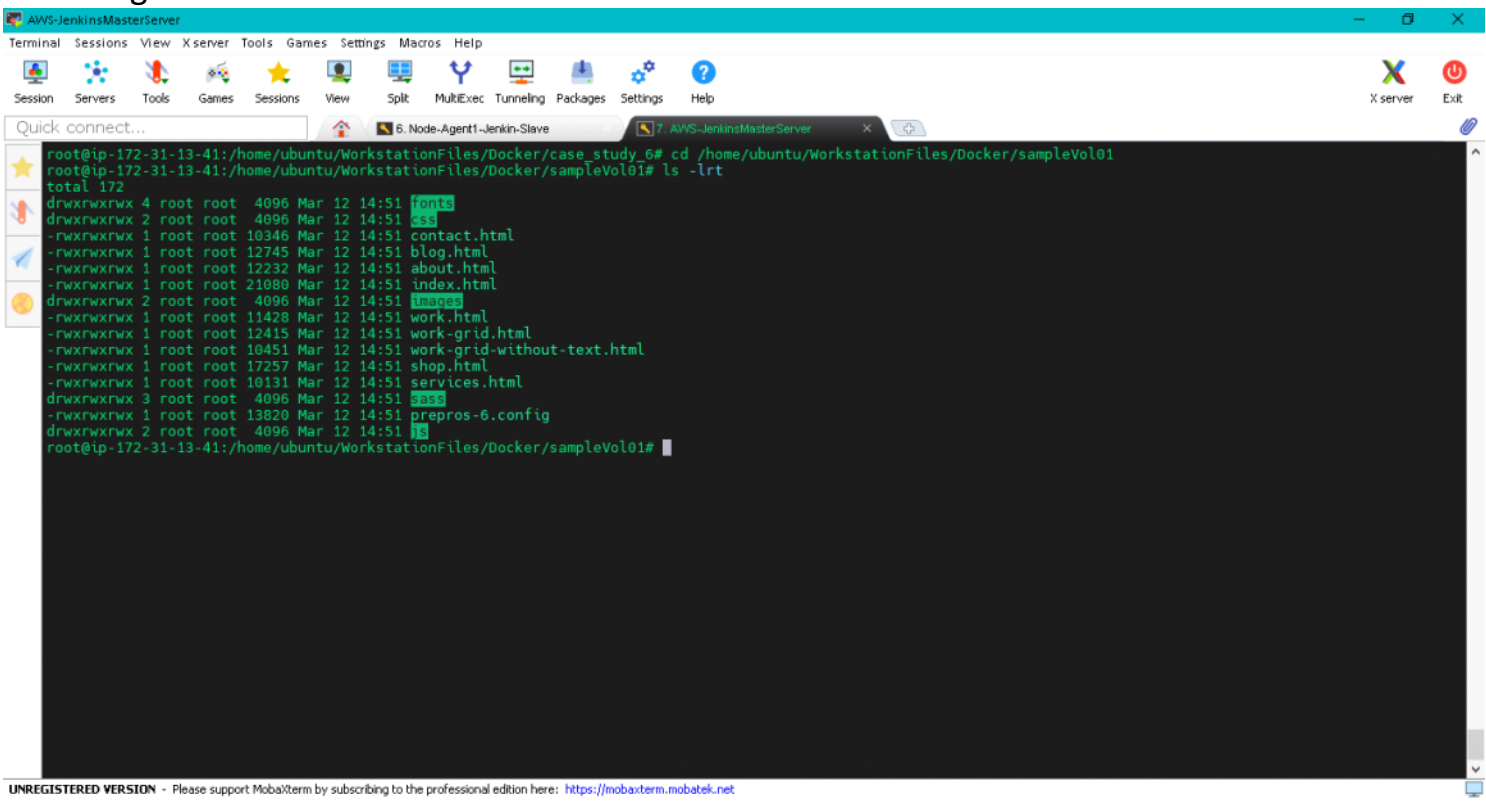
6. Pushing Custom Docker image from Manager Node into Docker Hub



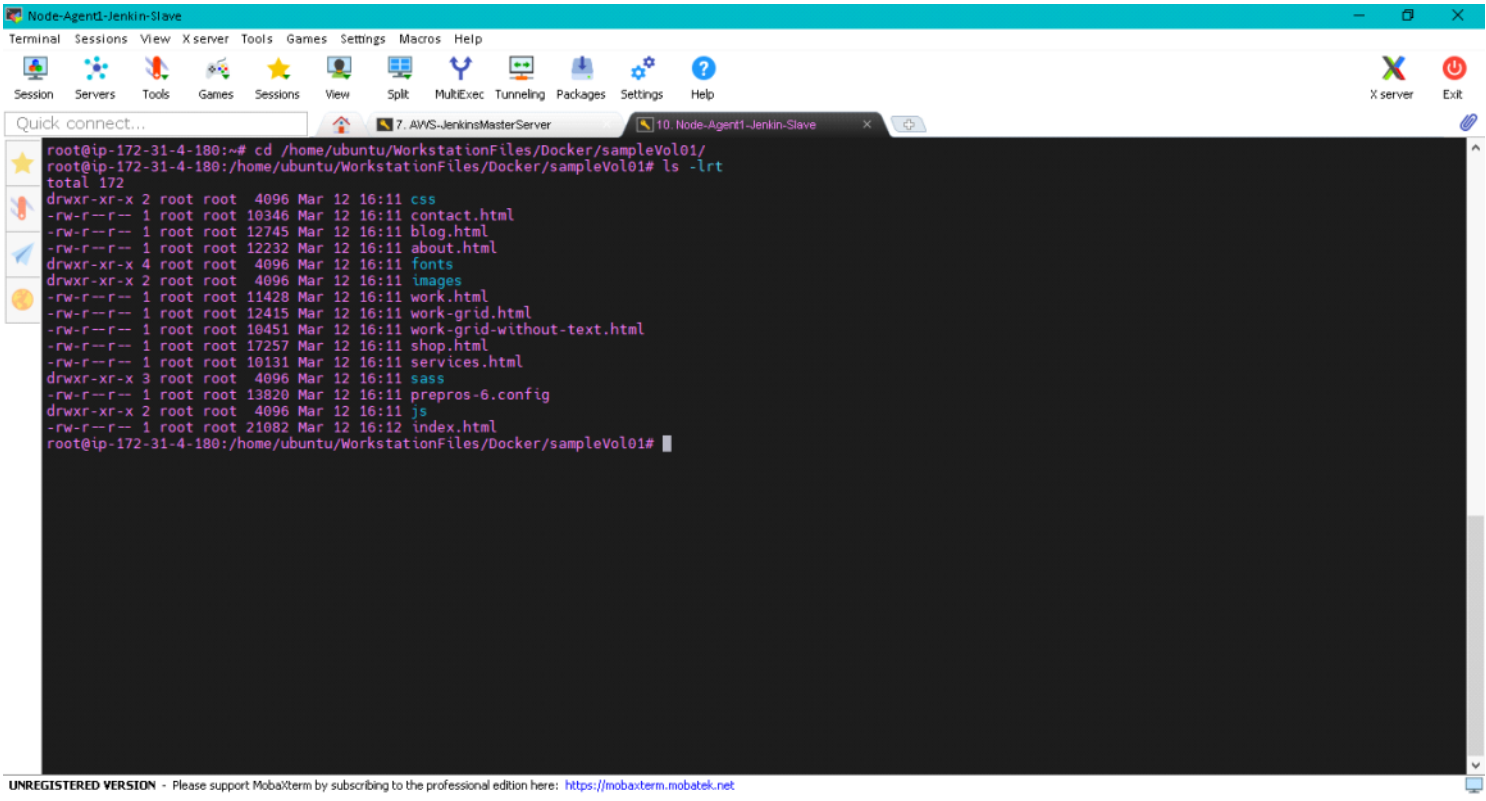
7. Custom docker image pushed into Docker hub



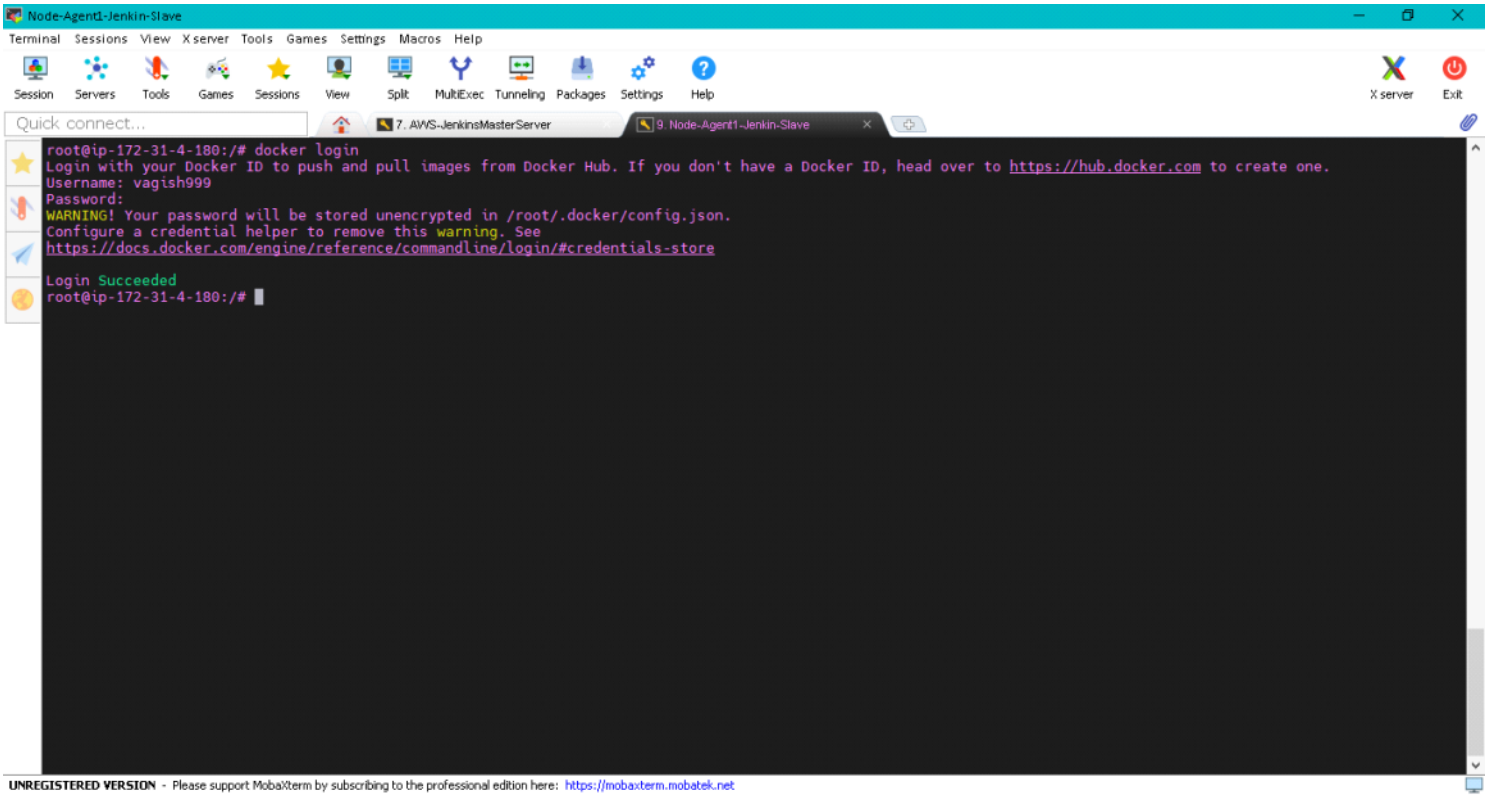
8. Checked out shared website data into Manager node from GIT.



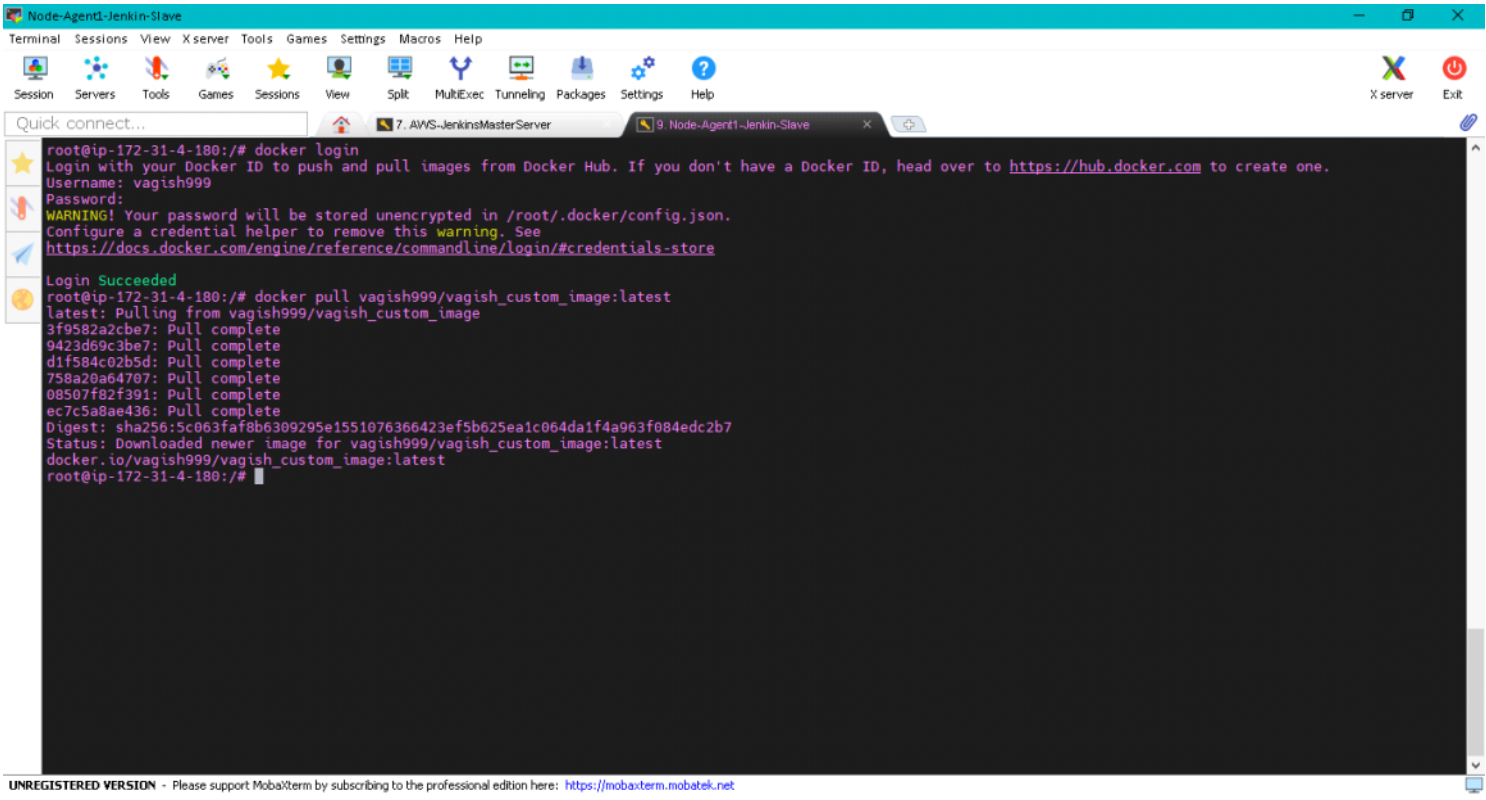
9. Checked out shared website data into Worker Node from GIT.



10. Login into Docker Hub from Worker Node



11. Pulling Custom image pushed from Manager node in Worker Node from Docker Hub



12. Creating docker-compose.yml to deploy

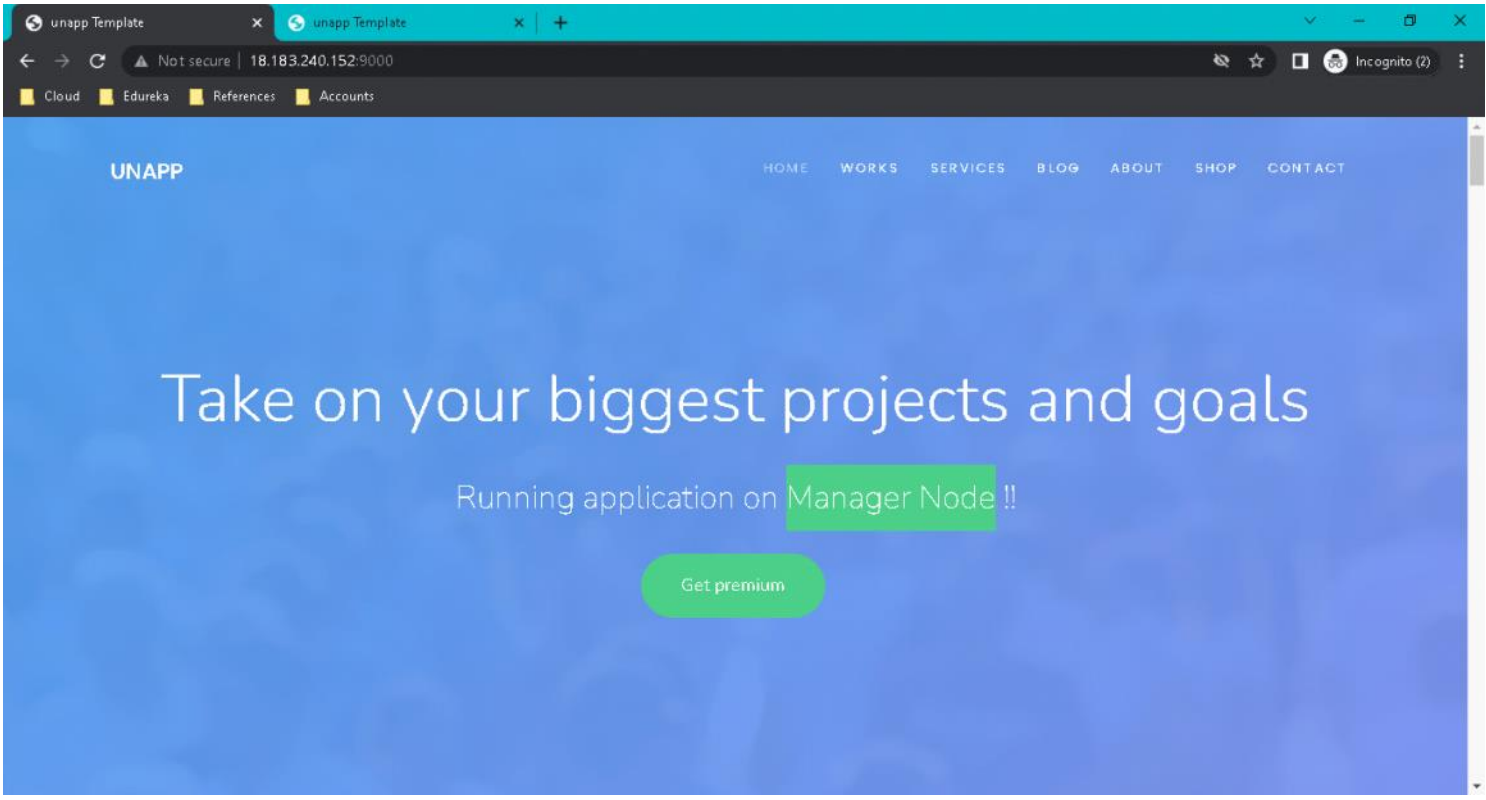
& run website using Custom Docker image in Docker Swarm Cluster (i.e. Manager & Worker node both)
Note: Volume is mapped from external folder.

```
root@ip-172-31-13-41:/home/ubuntu/WorkstationFiles/Docker/case_study_6# cat docker-compose.yml
version: '3.9'

services:
  my-service:
    image: vagish999/vagish_custom_image
    deploy:
      replicas: 5
      resources:
        limits:
          cpus: '0.5'
          memory: 256M
      restart_policy:
        condition: on-failure
    ports:
      - "9000:80"
    volumes:
      - "/home/ubuntu/WorkstationFiles/Docker/sampleVol01:/usr/local/apache2/htdocs"
root@ip-172-31-13-41:/home/ubuntu/WorkstationFiles/Docker/case_study_6#
```

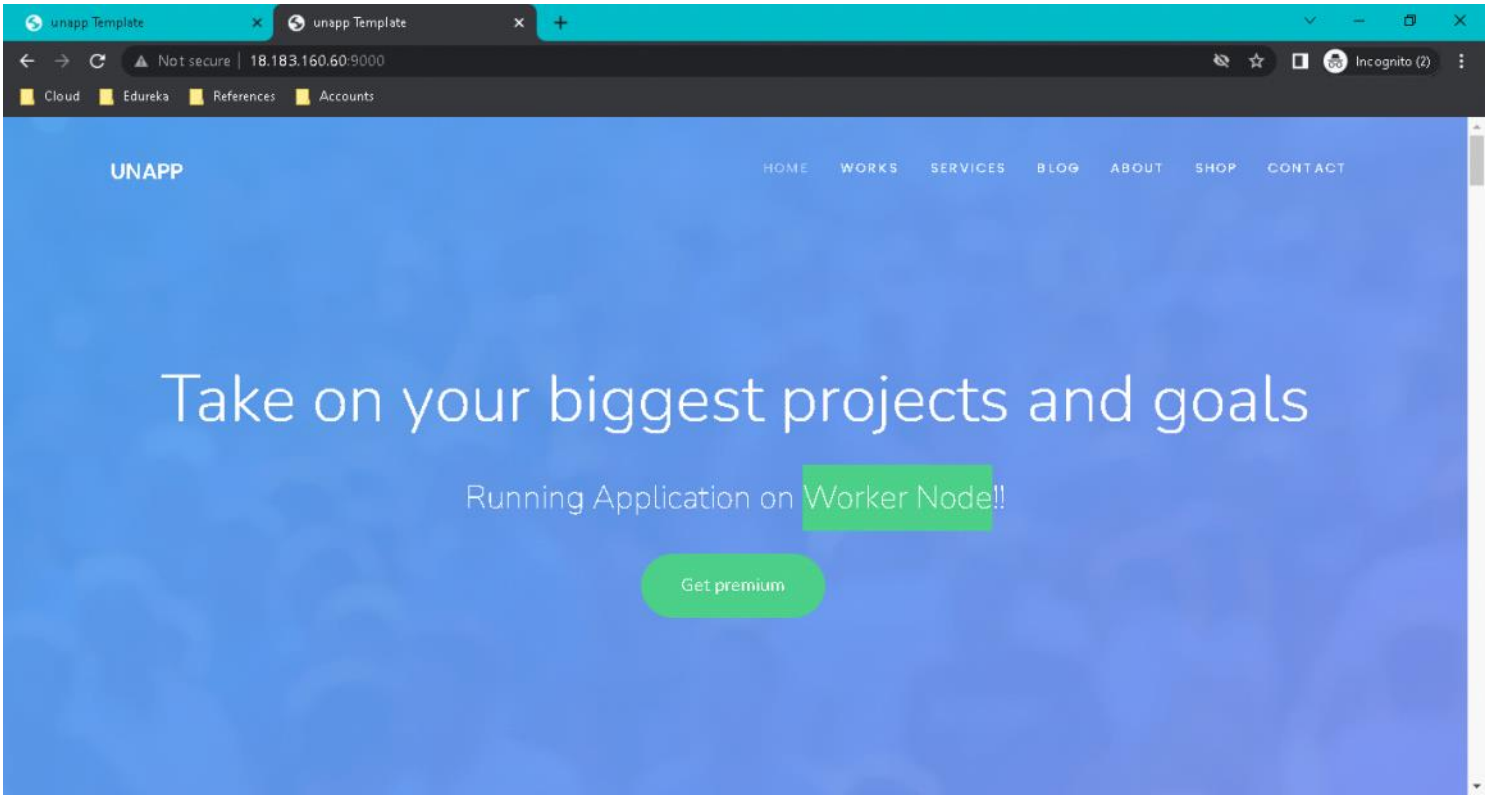
13. Deploying docker-compose into Docker Swarm cluster

```
root@ip-172-31-13-41:/home/ubuntu/WorkstationFiles/Docker/case_study_6# docker stack deploy --compose-file docker-compose.yml my-service
Creating service my-service_my-service
root@ip-172-31-13-41:/home/ubuntu/WorkstationFiles/Docker/case_study_6# docker service ls
ID             NAME                 MODE         REPLICAS   IMAGE                                      PORTS
xe7ld35dl5sz   my-service_my-service   replicated   5/5        vagish999/vagish_custom_image:latest   *:9000->80/tcp
root@ip-172-31-13-41:/home/ubuntu/WorkstationFiles/Docker/case_study_6# docker service ps my-service_my-service
ID             NAME                     IMAGE                                      NODE             DESIRED STATE   CURRENT STATE           ERROR     PORTS
xz8semwy6lf4   my-service_my-service.1   vagish999/vagish_custom_image:latest   ip-172-31-4-180   Running         Running 11 seconds ago
0qyqx8p85byg   my-service_my-service.2   vagish999/vagish_custom_image:latest   ip-172-31-13-41   Running         Running 12 seconds ago
df6cqqvtzeel   my-service_my-service.3   vagish999/vagish_custom_image:latest   ip-172-31-4-180   Running         Running 11 seconds ago
txyszuub3yxu   my-service_my-service.4   vagish999/vagish_custom_image:latest   ip-172-31-13-41   Running         Running 12 seconds ago
h4vh77eke8mr   my-service_my-service.5   vagish999/vagish_custom_image:latest   ip-172-31-4-180   Running         Running 11 seconds ago
root@ip-172-31-13-41:/home/ubuntu/WorkstationFiles/Docker/case_study_6#
```

14. Manager Node Result



15. Worker node Result



Uploaded PDF with consolidated Screenshots.
Below tasks are performed:

1. Created 2 AWS VM instance to be used as manager & worker node in Docker Swarm

2. Created Docker Swarm Cluster
3. Checked out GIT code shared in case study.
4. Created Docker Image and pushed into Docker hub from Manager node
5. Custom Docker image pulled into Worker node.
6. Created docker-compose.yaml to deploy website via Docker image into Docker Swarm Cluster.
7. Accessed website page using Public Ip of Manager & Worker node separately.