

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

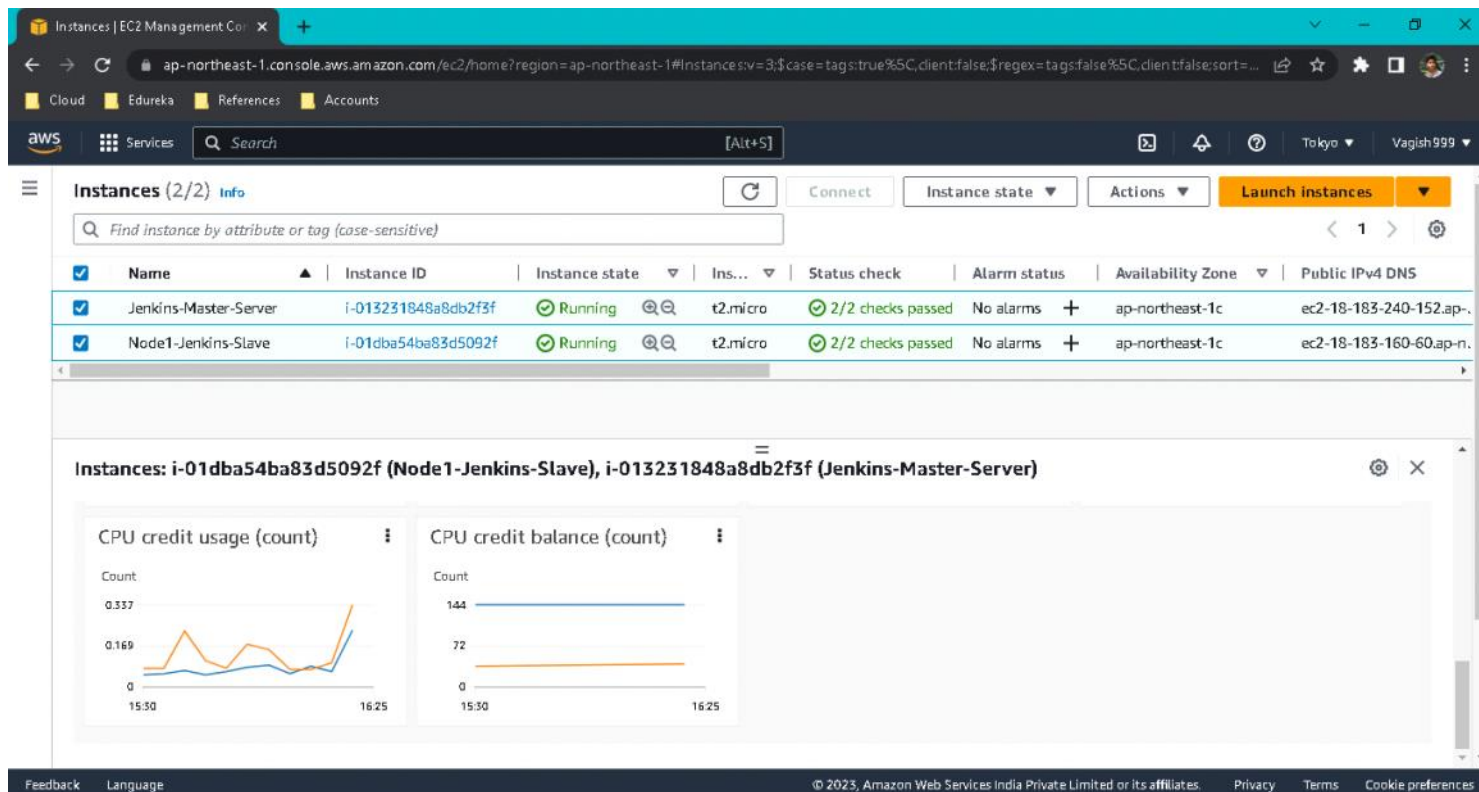
Hint: Use a Linux machine for installing Docker.

Solution:

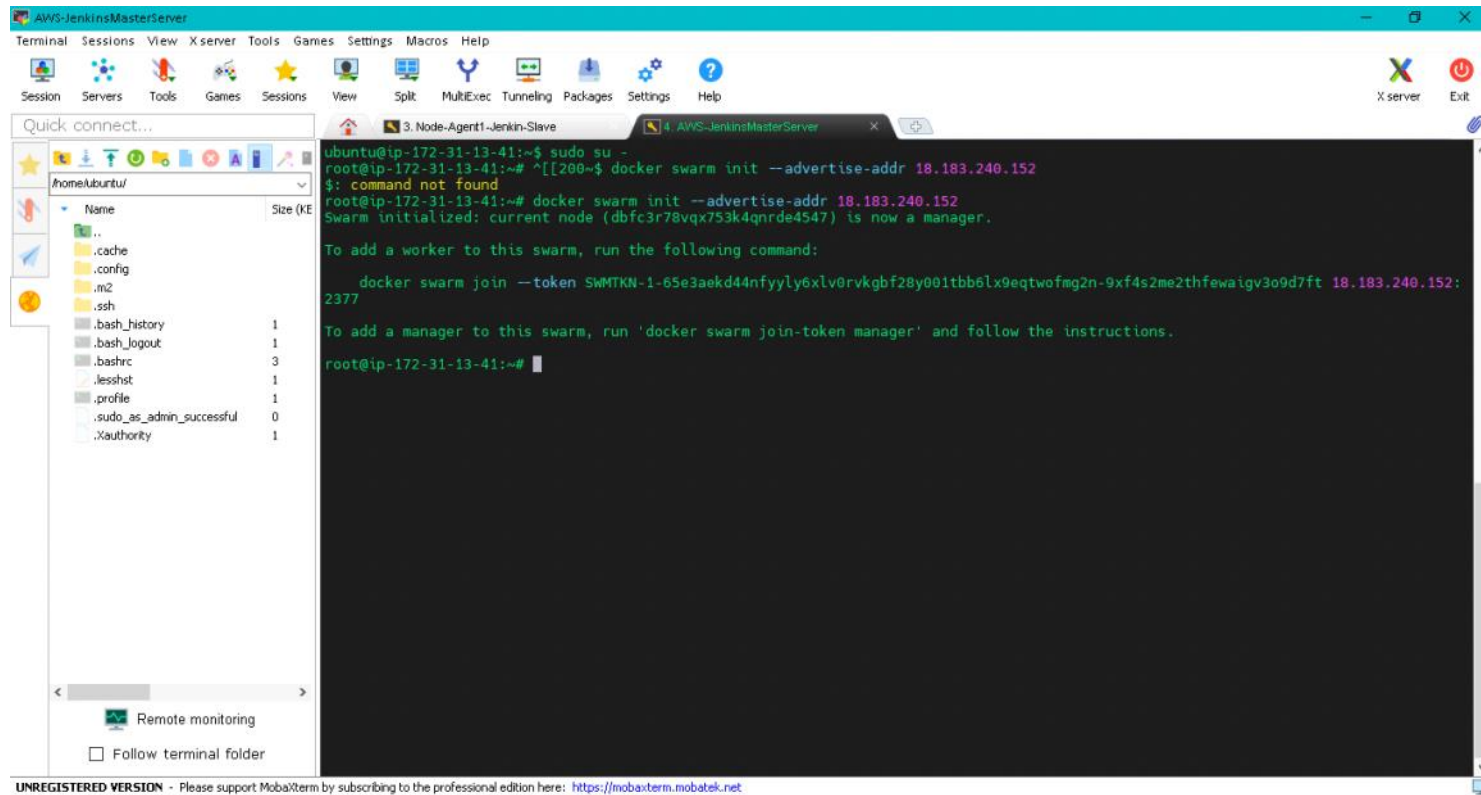
-> Used AWS to get two VM instances used as Manager and Worked Node for Swarm cluster.

-> 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



The screenshot shows a MobaXterm window titled 'AWS-JenkinsMasterServer'. On the left, a 'Quick connect...' sidebar shows a file tree for 'home/ubuntu/'. The main terminal pane shows the following commands and output:

```
ubuntu@ip-172-31-13-41:~$ sudo su -
root@ip-172-31-13-41:~# ^[[200~$ docker swarm init --advertise-addr 18.183.240.152
$: command not found
root@ip-172-31-13-41:~# docker swarm init --advertise-addr 18.183.240.152
Swarm initialized: current node (dbfc3r78vqx753k4qnrde4547) is now a manager.

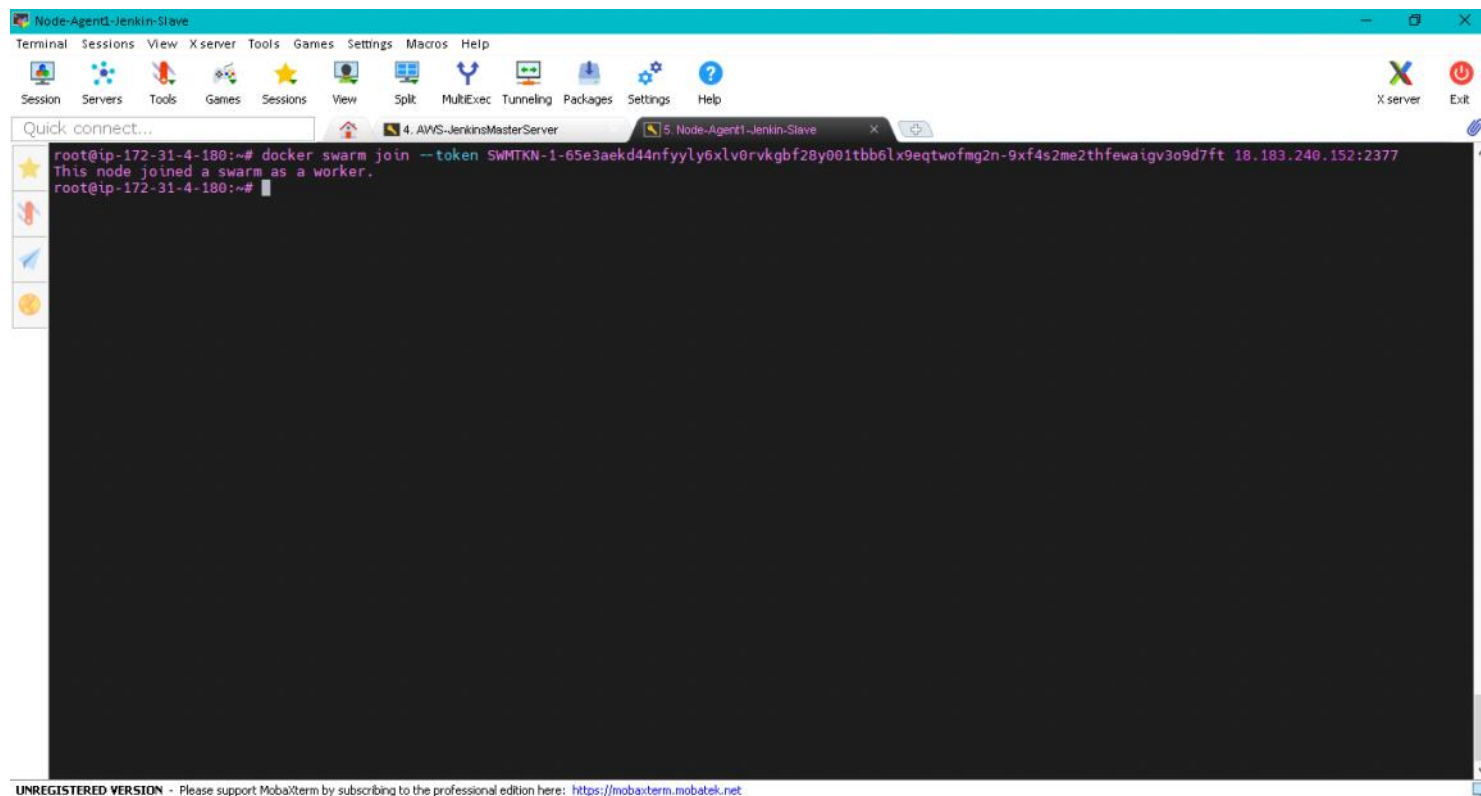
To add a worker to this swarm, run the following command:

    docker swarm join --token SWMTKN-1-65e3aekd44nfyly6xlv0rvkgbf28y001tbb6lx9eqtwofmg2n-9xf4s2me2thfewaigv3o9d7ft 18.183.240.152:2377

To add a manager to this swarm, run 'docker swarm join-token manager' and follow the instructions.
root@ip-172-31-13-41:~#
```

At the bottom of the window, a message reads: 'UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>'.

3. Joining Worker Node in Docker Swarm Cluster

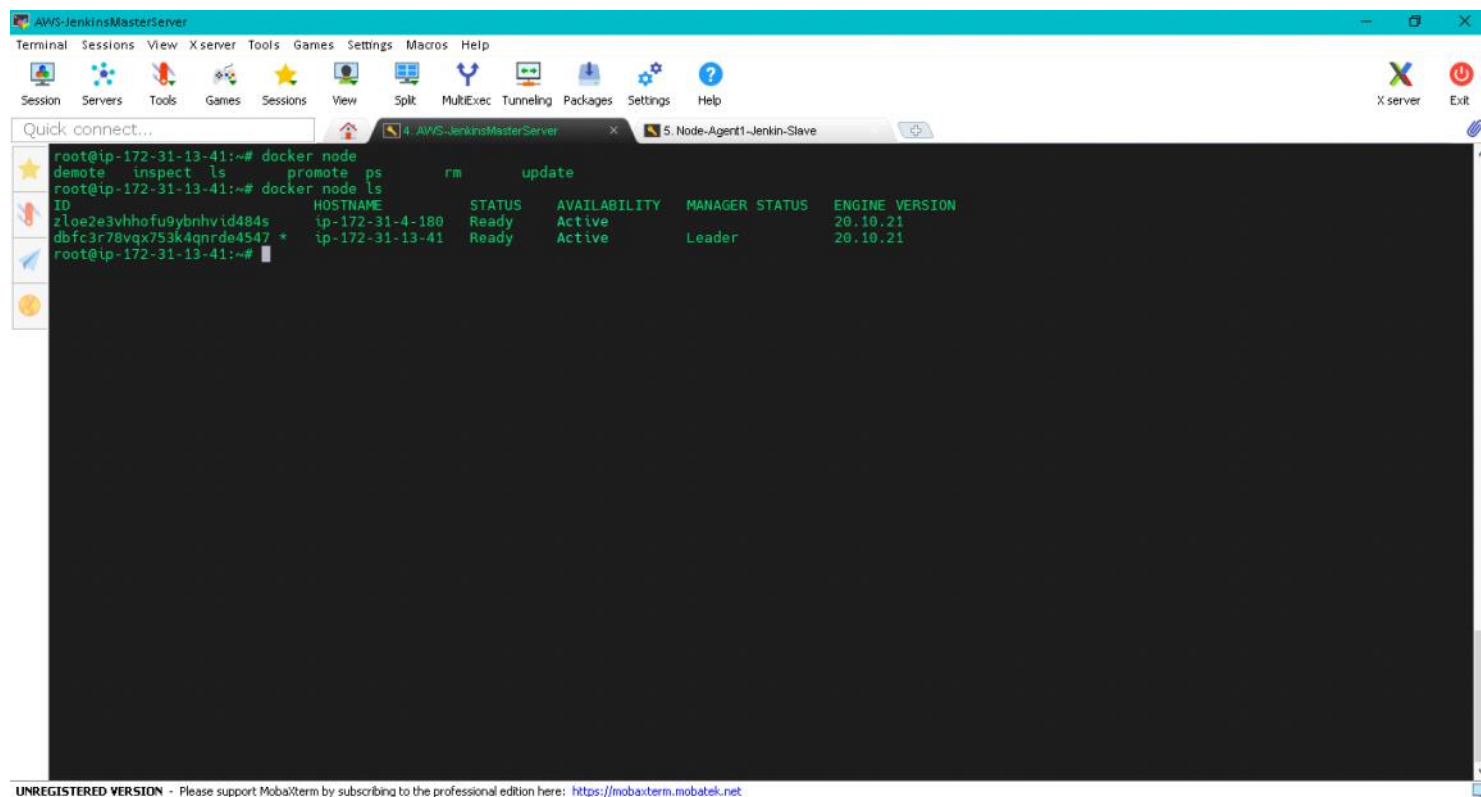


The screenshot shows a MobaXterm window titled 'Node-Agent1-Jenkin-Slave'. The main terminal pane shows the following commands and output:

```
root@ip-172-31-4-180:~# docker swarm join --token SWMTKN-1-65e3aekd44nfyly6xlv0rvkgbf28y001tbb6lx9eqtwofmg2n-9xf4s2me2thfewaigv3o9d7ft 18.183.240.152:2377
This node joined a swarm as a worker.
root@ip-172-31-4-180:~#
```

At the bottom of the window, a message reads: 'UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>'.

4. Both the nodes are linked now.

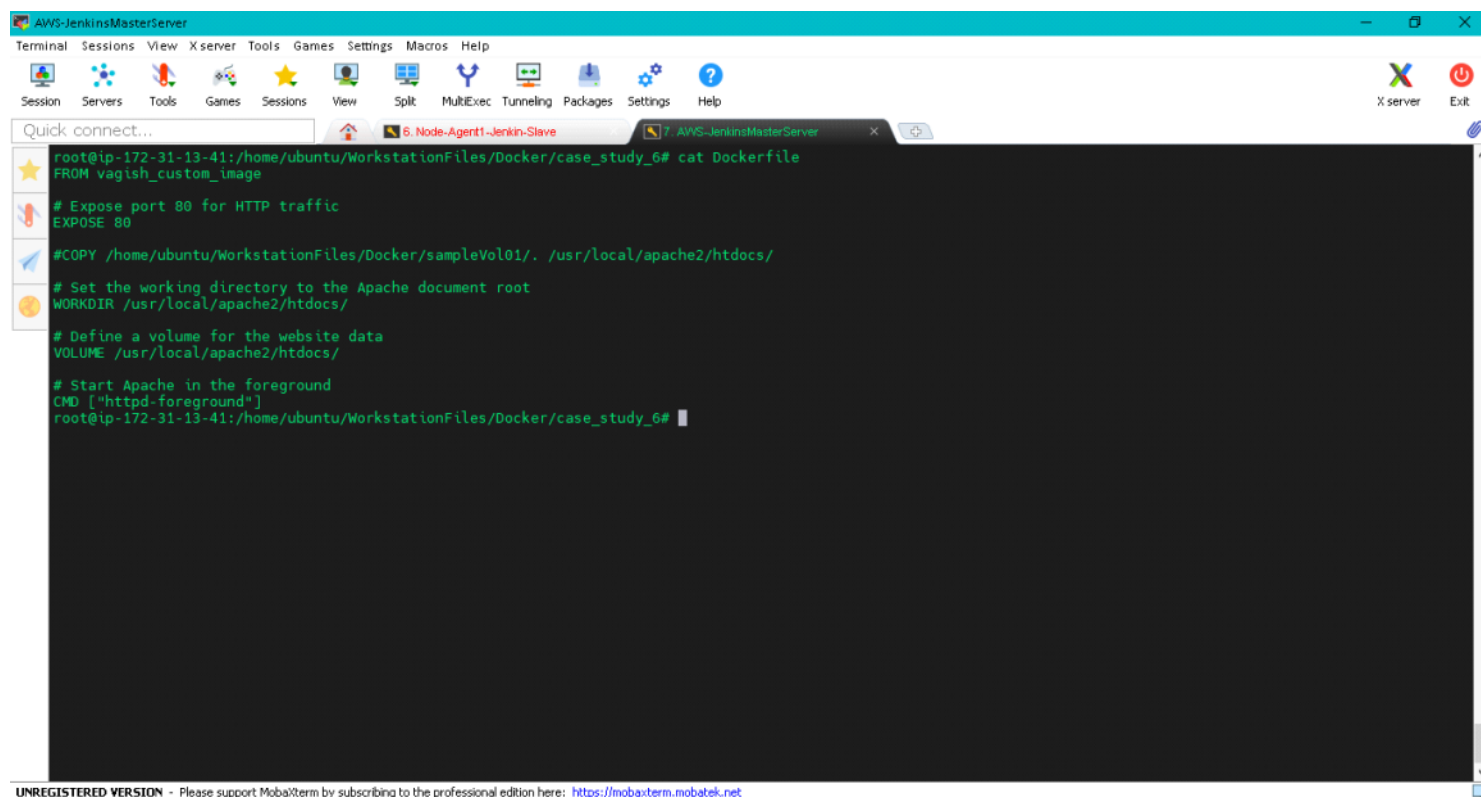


The screenshot shows a MobaXterm terminal window with the title bar 'AWS-JenkinsMasterServer'. The terminal displays the output of the command 'docker node ls', which lists two nodes: 'zloe2e3vhhofu9ybnhvid484s' and 'dbfc3r78vqx753k4qnrde4547'. The output is formatted as a table with columns: ID, HOSTNAME, STATUS, AVAILABILITY, MANAGER STATUS, and ENGINE VERSION.

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

Below the terminal window, a footer message reads: 'UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>'.

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

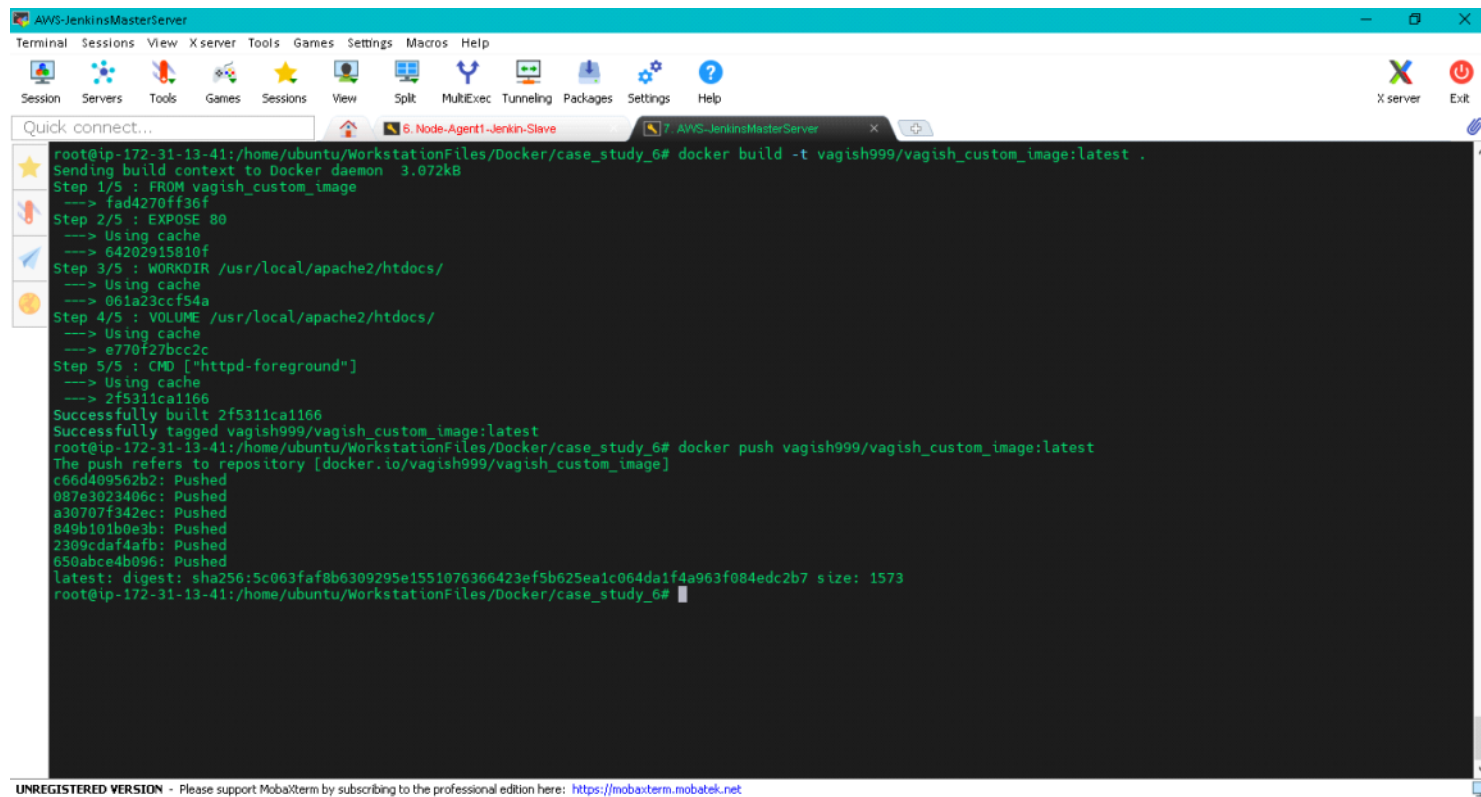


The screenshot shows a MobaXterm terminal window with the title bar 'AWS-JenkinsMasterServer'. The terminal displays the contents of a Dockerfile, which includes instructions for exposing port 80, copying files, setting the working directory, defining a volume, and starting Apache in the foreground.

```
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#
```

Below the terminal window, a footer message reads: 'UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>'.

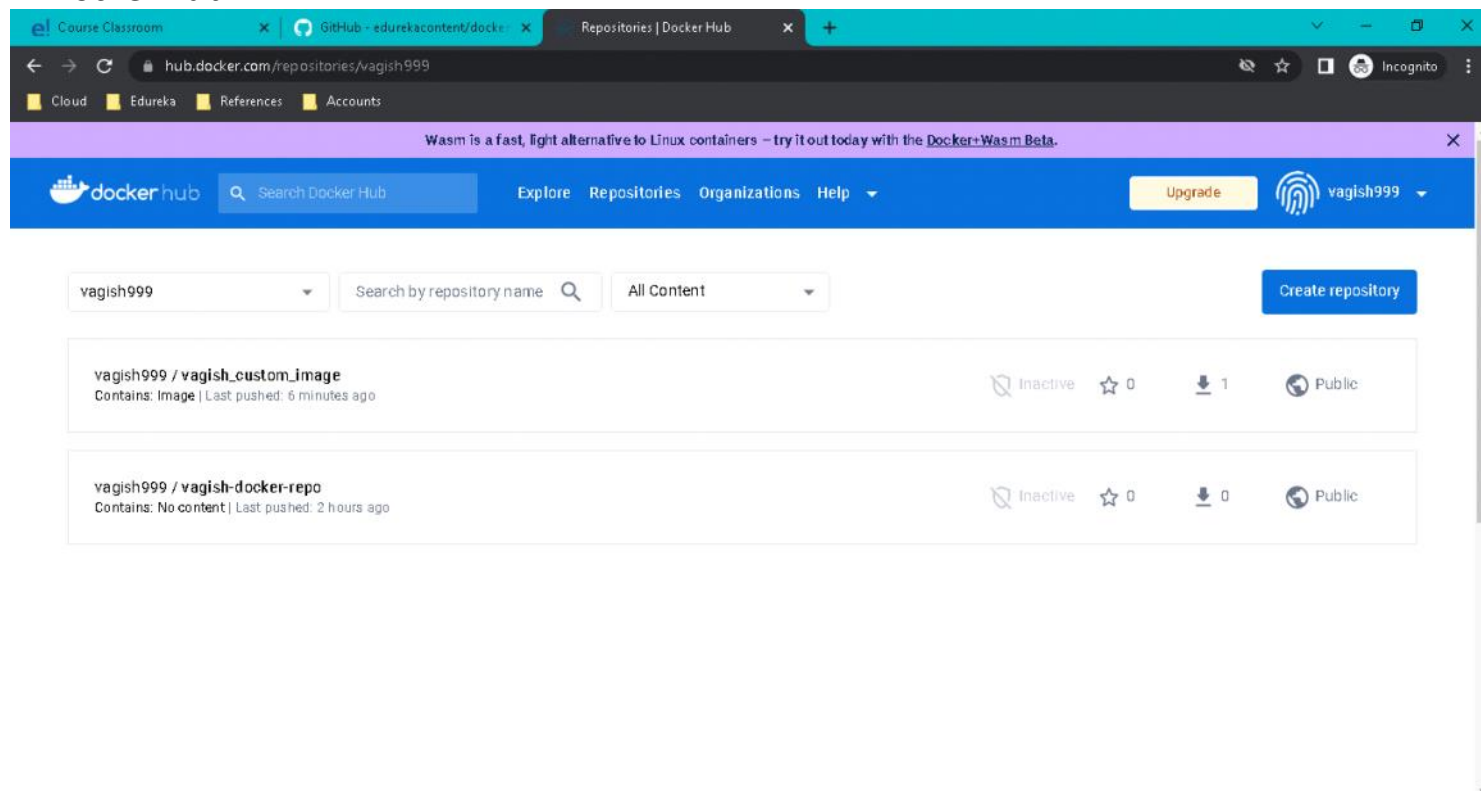
6. Pushing Custom Docker image from Manager Node into Docker Hub



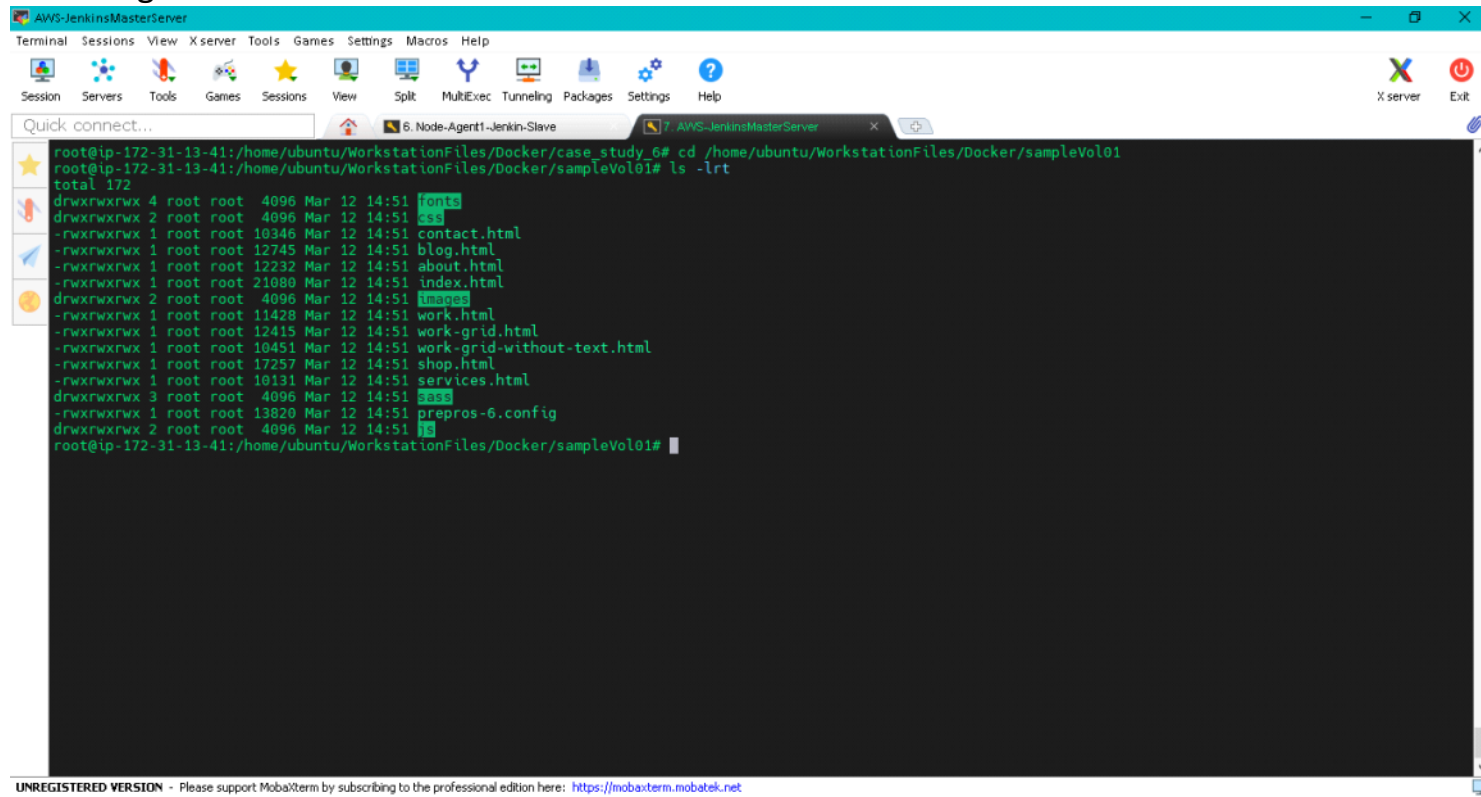
```
root@ip-172-31-13-41:/home/ubuntu/WorkstationFiles/Docker/case_study_6# docker build -t vagish999/vagish_custom_image:latest .
Sending build context to Docker daemon 3.072kB
Step 1/5 : FROM vagish_custom_image
--> fad4270ff36f
Step 2/5 : EXPOSE 80
--> Using cache
--> 64202915810f
Step 3/5 : WORKDIR /usr/local/apache2/htdocs/
--> Using cache
--> 061a23ccf54a
Step 4/5 : VOLUME /usr/local/apache2/htdocs/
--> Using cache
--> e770f27bcc2c
Step 5/5 : CMD ["httpd-foreground"]
--> Using cache
--> 2f5311ca1166
Successfully built 2f5311ca1166
Successfully tagged vagish999/vagish_custom_image:latest
root@ip-172-31-13-41:/home/ubuntu/WorkstationFiles/Docker/case_study_6# docker push vagish999/vagish_custom_image:latest
The push refers to repository [docker.io/vagish999/vagish_custom_image]
c66d409562b2: Pushed
087e3023406c: Pushed
a30707f342ec: Pushed
849b101b0e3b: Pushed
2309cda4afab: Pushed
650abce4b096: Pushed
latest: digest: sha256:5c063faf0b6309295e1551076366423ef5b625ea1c064da1f4a963f084edc2b7 size: 1573
root@ip-172-31-13-41:/home/ubuntu/WorkstationFiles/Docker/case_study_6#
```

UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>

7. Custom docker image pushed into Docker hub



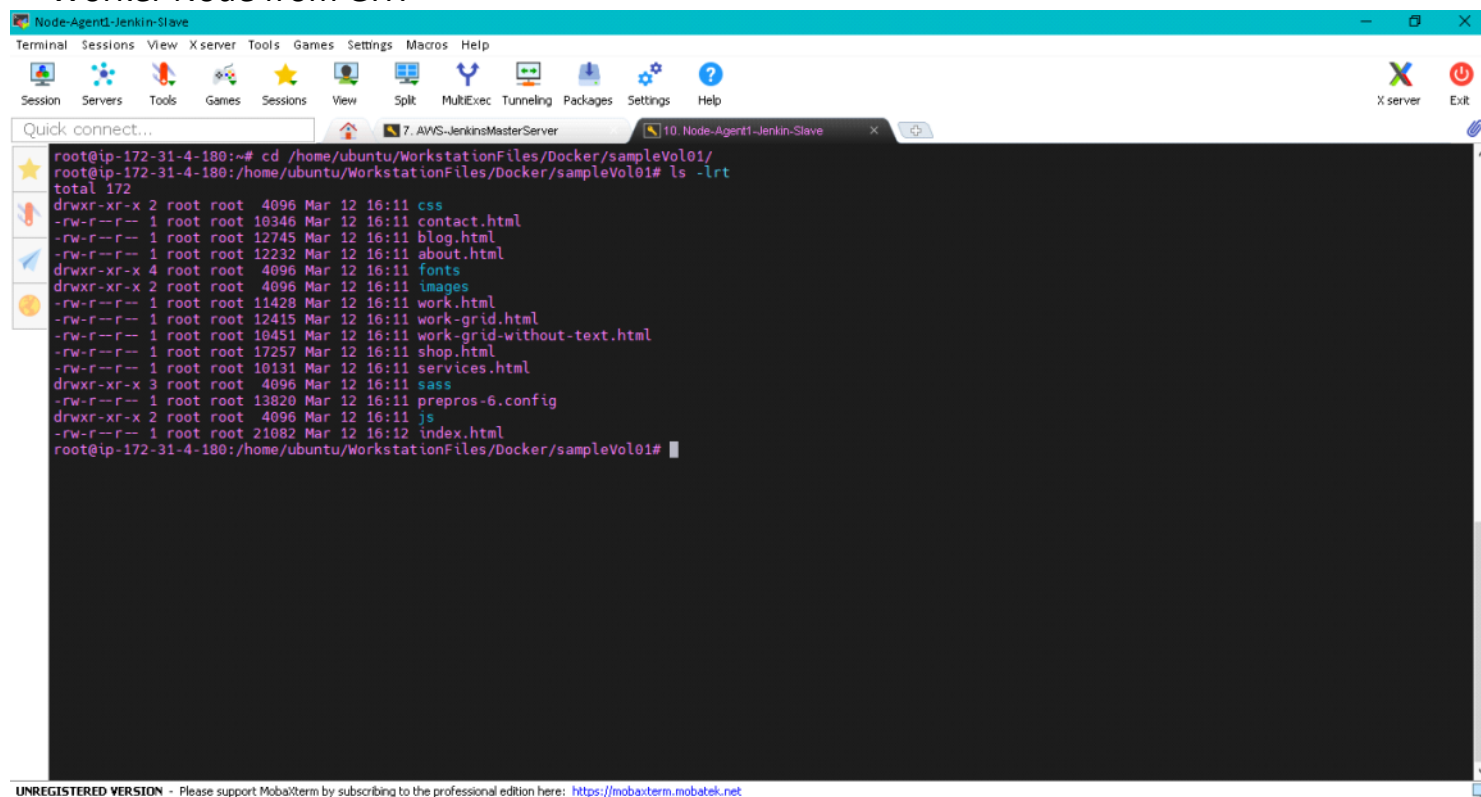
8. Checked out shared website data into Manager node from GIT.



The screenshot shows a MobaXterm window titled 'AWS-JenkinsMasterServer'. The terminal displays the following commands and output:

```
root@ip-172-31-13-41:/home/ubuntu/WorkstationFiles/Docker/case_study_6# cd /home/ubuntu/WorkstationFiles/Docker/sampleVol01
root@ip-172-31-13-41:/home/ubuntu/WorkstationFiles/Docker/sampleVol01# ls -lrt
total 172
drwxrwxrwx 4 root root 4096 Mar 12 14:51 fonts
drwxrwxrwx 2 root root 4096 Mar 12 14:51 css
-rwxrwxrwx 1 root root 10346 Mar 12 14:51 contact.html
-rwxrwxrwx 1 root root 12745 Mar 12 14:51 blog.html
-rwxrwxrwx 1 root root 12232 Mar 12 14:51 about.html
-rwxrwxrwx 1 root root 21080 Mar 12 14:51 index.html
drwxrwxrwx 2 root root 4096 Mar 12 14:51 images
-rwxrwxrwx 1 root root 11428 Mar 12 14:51 work.html
-rwxrwxrwx 1 root root 12415 Mar 12 14:51 work-grid.html
-rwxrwxrwx 1 root root 10451 Mar 12 14:51 work-grid-without-text.html
-rwxrwxrwx 1 root root 17257 Mar 12 14:51 shop.html
-rwxrwxrwx 1 root root 10131 Mar 12 14:51 services.html
drwxrwxrwx 3 root root 4096 Mar 12 14:51 sass
-rwxrwxrwx 1 root root 13820 Mar 12 14:51 prepros-6.config
drwxrwxrwx 2 root root 4096 Mar 12 14:51 js
root@ip-172-31-13-41:/home/ubuntu/WorkstationFiles/Docker/sampleVol01#
```

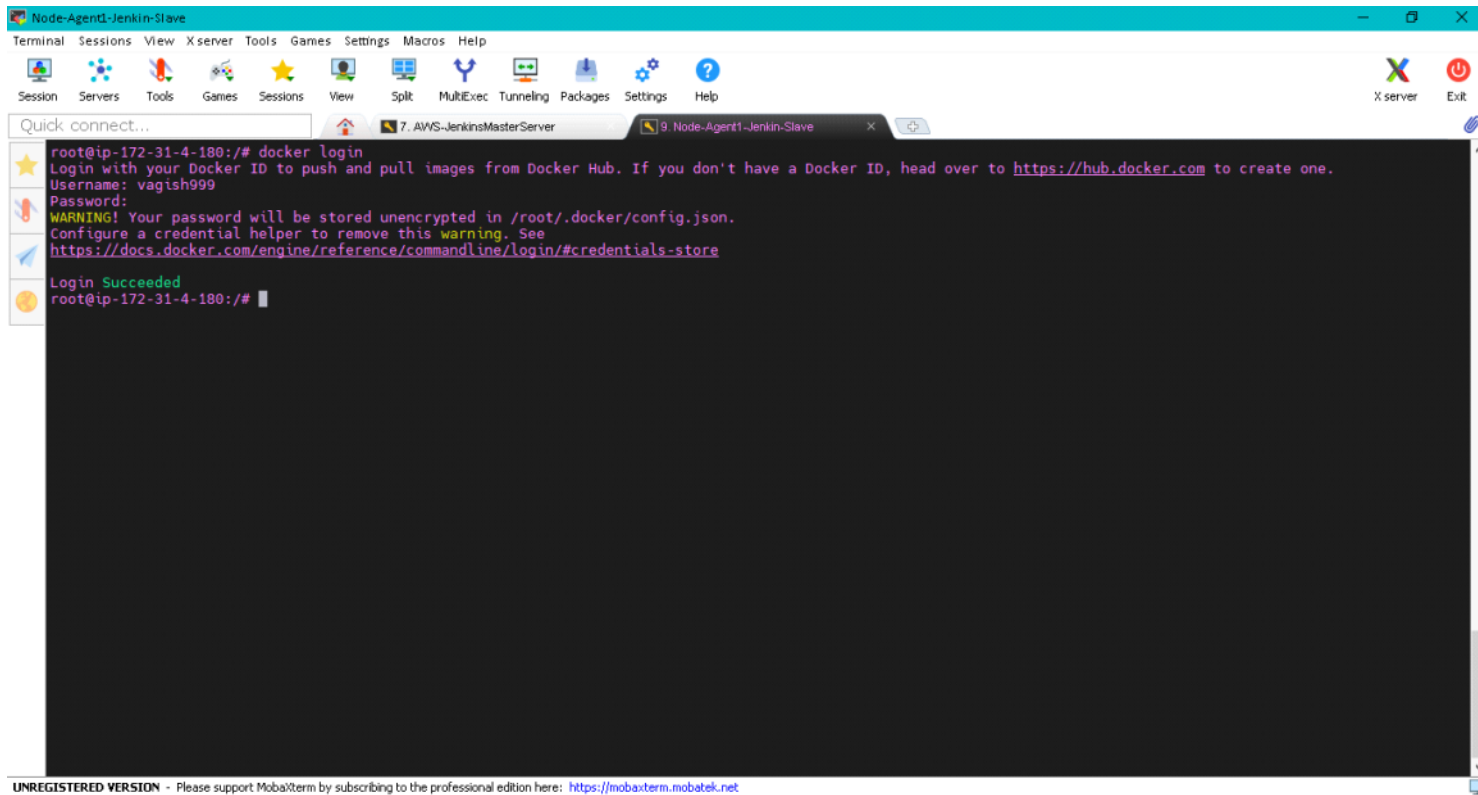
9. Checked out shared website data into Worker Node from GIT.



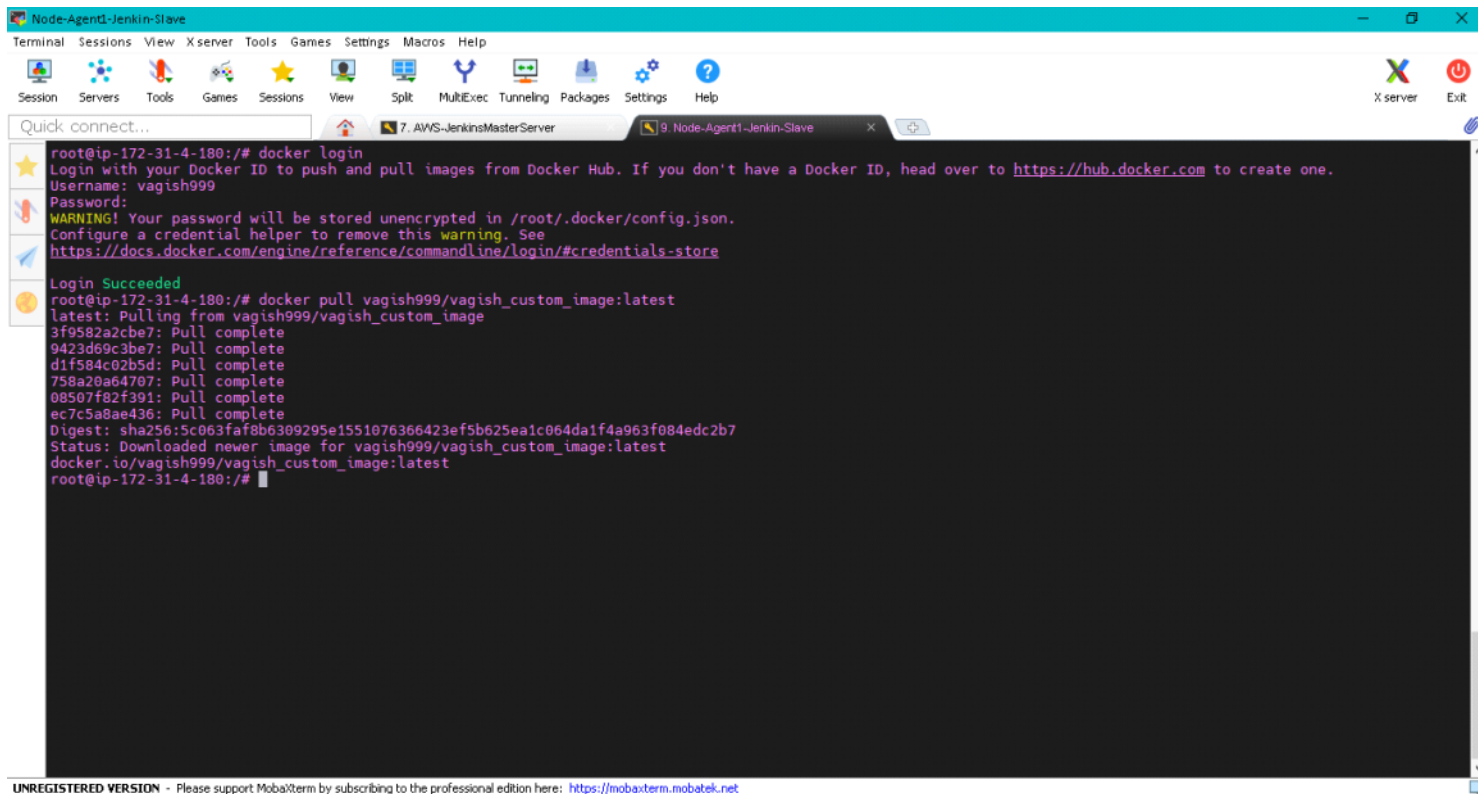
The screenshot shows a MobaXterm window titled 'Node-Agent1-Jenkins-Slave'. The terminal displays the following commands and output:

```
root@ip-172-31-4-180:~# cd /home/ubuntu/WorkstationFiles/Docker/sampleVol01/
root@ip-172-31-4-180:/home/ubuntu/WorkstationFiles/Docker/sampleVol01# ls -lrt
total 172
drwxr-xr-x 2 root root 4096 Mar 12 16:11 css
-rw-r--r-- 1 root root 10346 Mar 12 16:11 contact.html
-rw-r--r-- 1 root root 12745 Mar 12 16:11 blog.html
-rw-r--r-- 1 root root 12232 Mar 12 16:11 about.html
drwxr-xr-x 4 root root 4096 Mar 12 16:11 fonts
drwxr-xr-x 2 root root 4096 Mar 12 16:11 images
-rw-r--r-- 1 root root 11428 Mar 12 16:11 work.html
-rw-r--r-- 1 root root 12415 Mar 12 16:11 work-grid.html
-rw-r--r-- 1 root root 10451 Mar 12 16:11 work-grid-without-text.html
-rw-r--r-- 1 root root 17257 Mar 12 16:11 shop.html
-rw-r--r-- 1 root root 10131 Mar 12 16:11 services.html
drwxr-xr-x 3 root root 4096 Mar 12 16:11 sass
-rw-r--r-- 1 root root 13820 Mar 12 16:11 prepros-6.config
drwxr-xr-x 2 root root 4096 Mar 12 16:11 js
-rw-r--r-- 1 root root 21082 Mar 12 16:12 index.html
root@ip-172-31-4-180:/home/ubuntu/WorkstationFiles/Docker/sampleVol01#
```

10. Login into Docker Hub from Worker Node

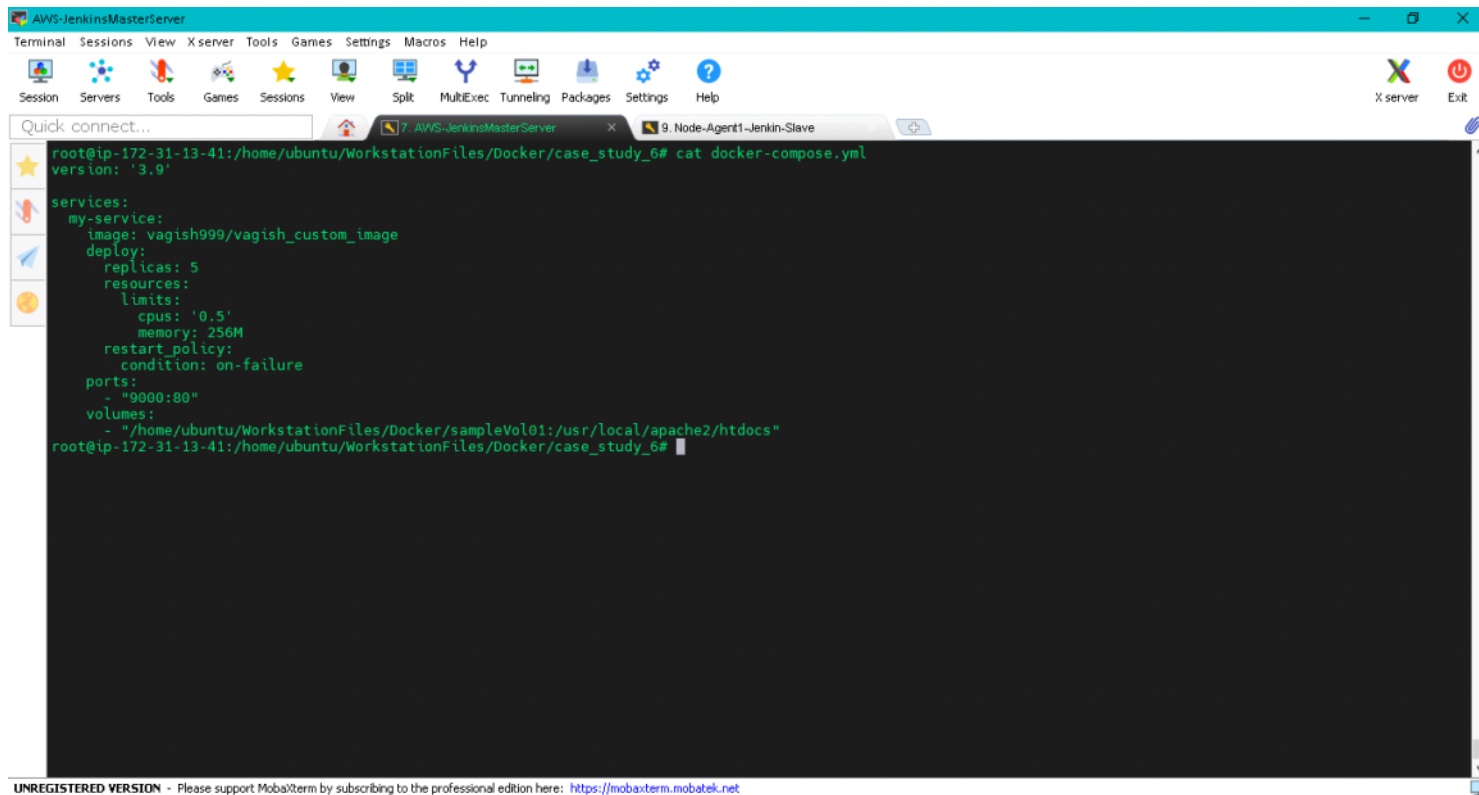


11. Pulling Custom image pushed from Manager node in Worker Node from Docker Hub



12. Creating docker-compose.yml to deploy

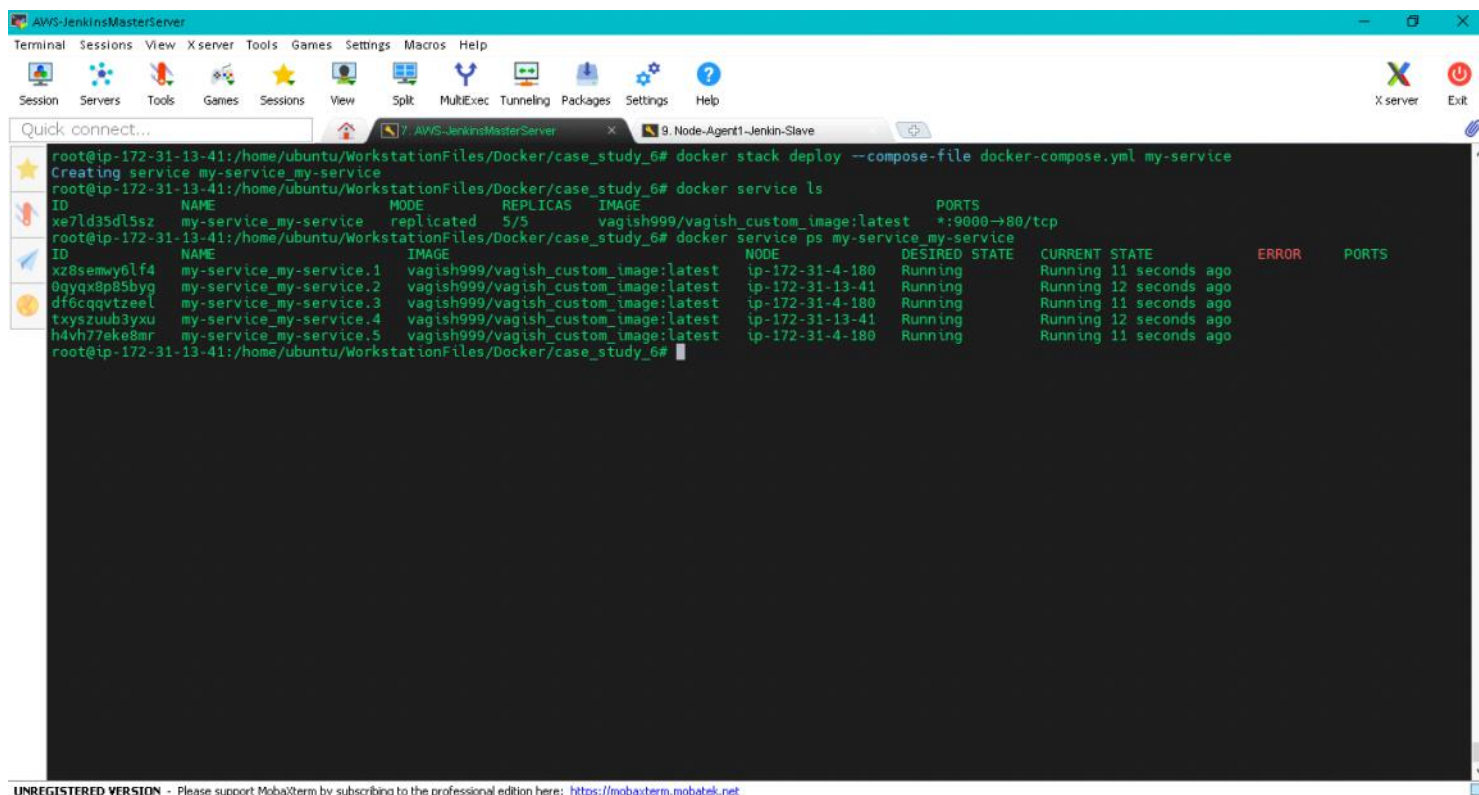
& run website using Custom Docker image in Docker Swarm Cluster (i.e. Manager & Worker node both)
Note: Volume is mapped from external folder.



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

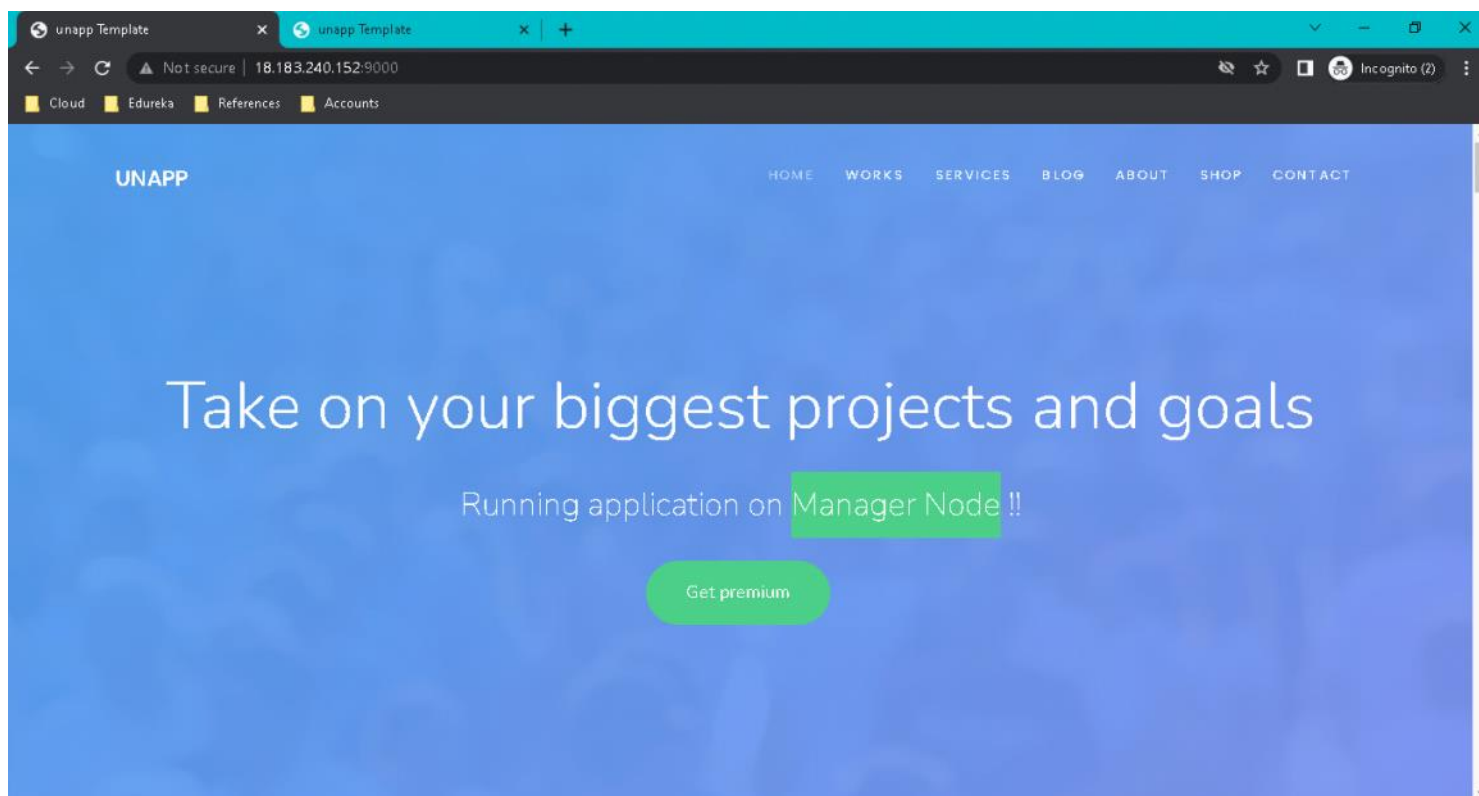
services:
  my-service:
    image: vagish999/vagish_custom_image
    deploy:
      replicas: 5
      resources:
        limits:
          cpus: '0.5'
          memory: 256M
      restart_policy:
        condition: on-failure
    ports:
      - "9000:80"
    volumes:
      - "/home/ubuntu/WorkstationFiles/Docker/sampleVol01:/usr/local/apache2/htdocs"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

13. Deploying docker-compose into Docker Swarm cluster

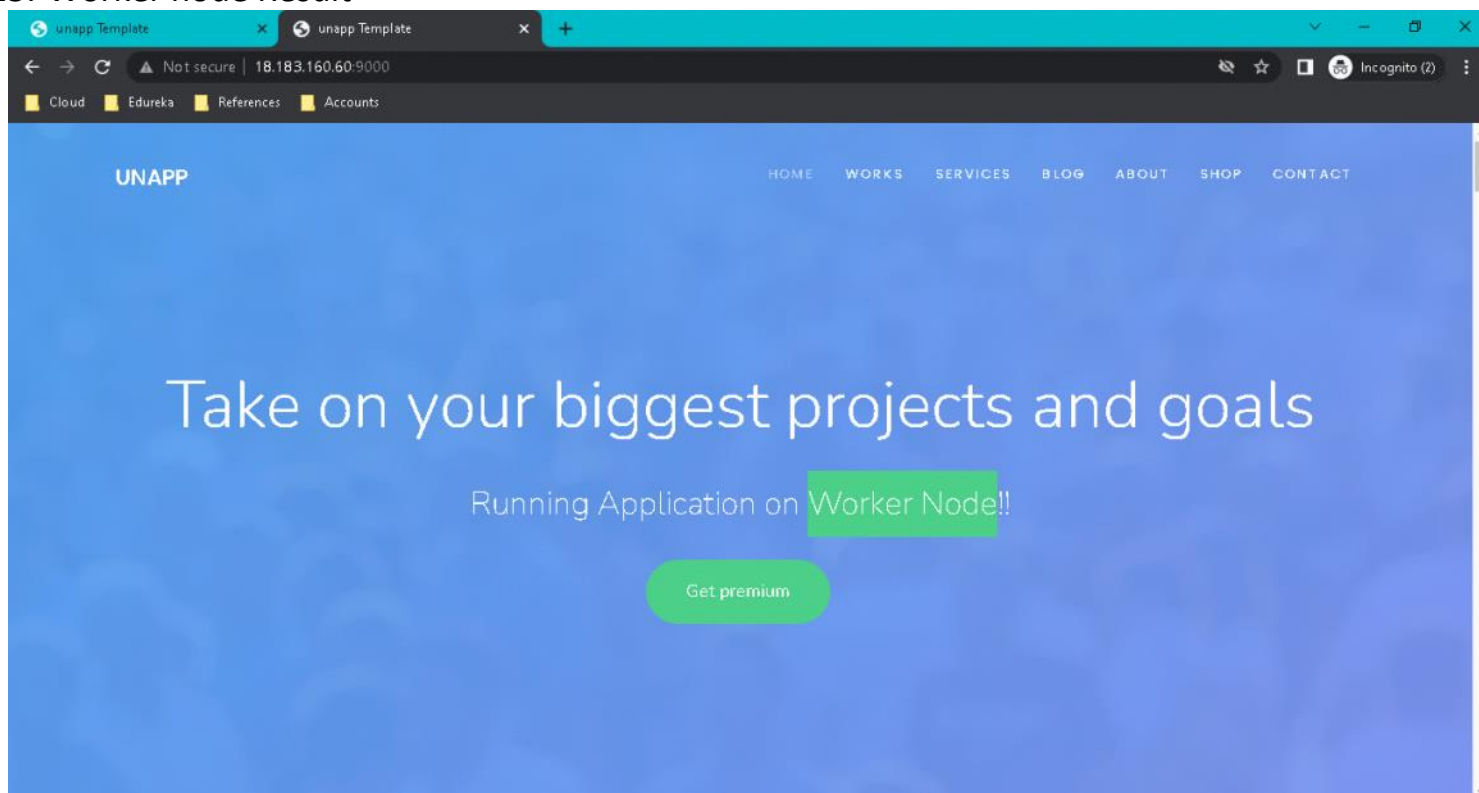


```
root@ip-172-31-13-41:/home/ubuntu/WorkstationFiles/Docker/case_study_6# docker stack deploy --compose-file docker-compose.yml my-service
Creating service my-service_my-service
root@ip-172-31-13-41:/home/ubuntu/WorkstationFiles/Docker/case_study_6# docker service ls
ID            NAME                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
xz8semmy6lf4  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
df6cqvztzeel  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
h4vh77ke8mr   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.