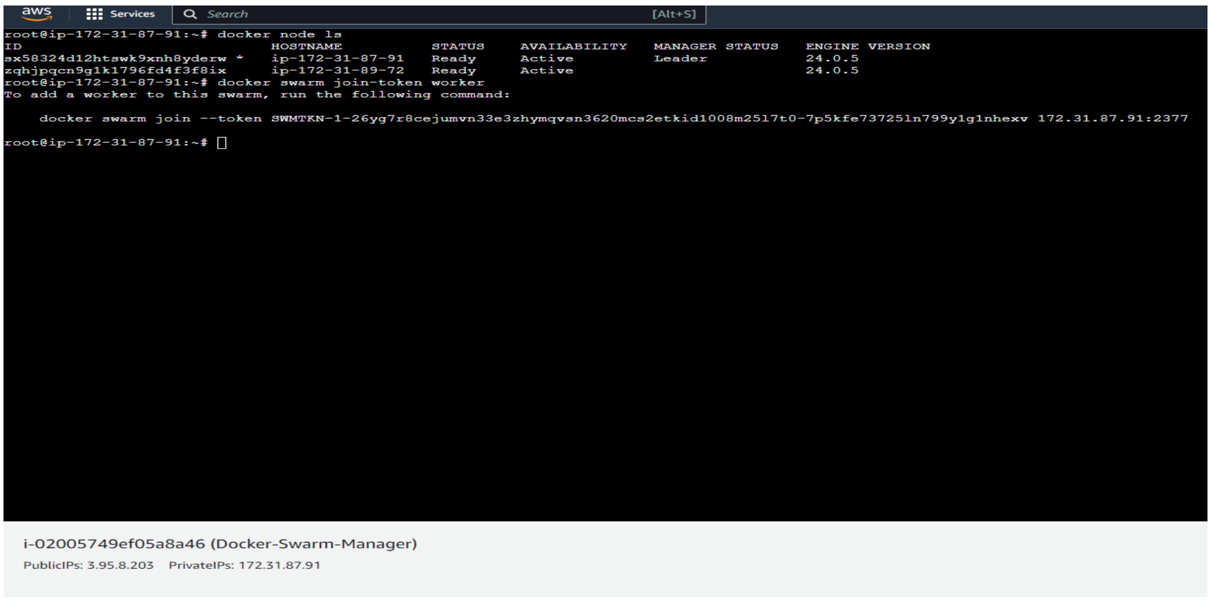
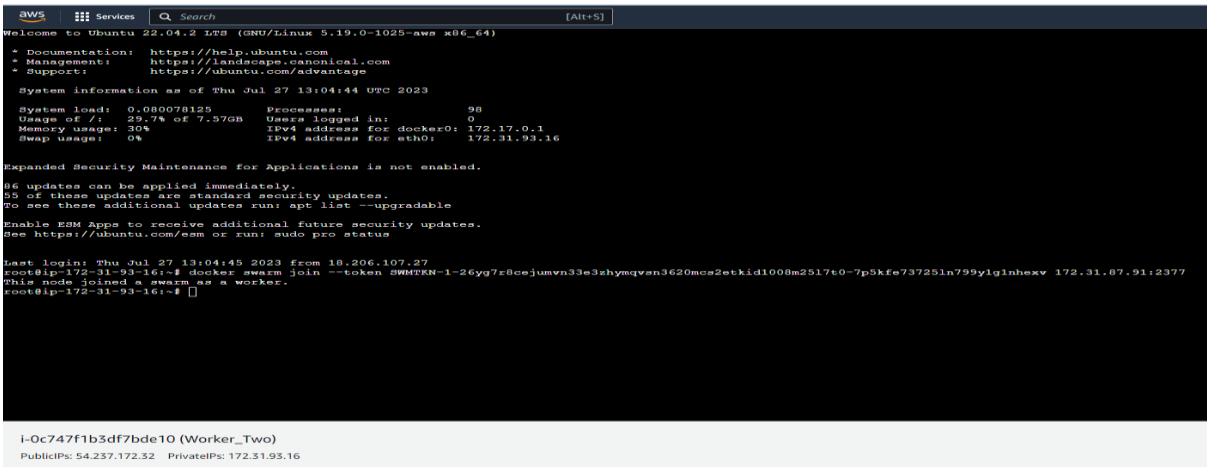
We use command docker swarm join-token worker to get worker command token so that we can copy paste the command in worker 2 system.



```
root@ip-172-31-87-91:~# docker node ls
ID                            HOSTNAME          STATUS    AVAILABILITY   MANAGER STATUS   ENGINE VERSION
sx58324d12htswk9xnh8yderw *   ip-172-31-87-91   Ready     Active         Leader           24.0.5
zqhjpqcn9g1k1796fd4f3f8ix     ip-172-31-89-72   Ready     Active                          24.0.5
root@ip-172-31-87-91:~# docker swarm join-token worker
To add a worker to this swarm, run the following command:

    docker swarm join --token SWMTKN-1-26yg7r8cejumvn33e3zhymqvsn3620mcs2etkid1008m2517t0-7p5kfe737251n799y1g1nhexv 172.31.87.91:2377

root@ip-172-31-87-91:~# 
```

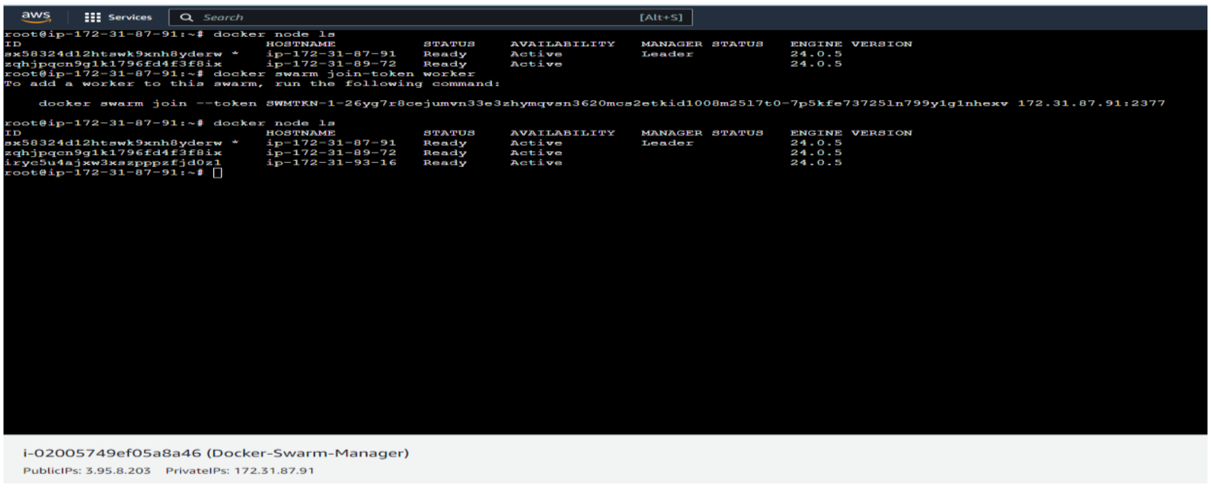
i-02005749ef05a8a46 (Docker-Swarm-Manager)
PublicIPs: 3.95.8.203   PrivateIPs: 172.31.87.91

As we can see that user joined as a node worker which shown in below screenshot. If we run the command docker node ls in manager user system we can user with worker and manager status.



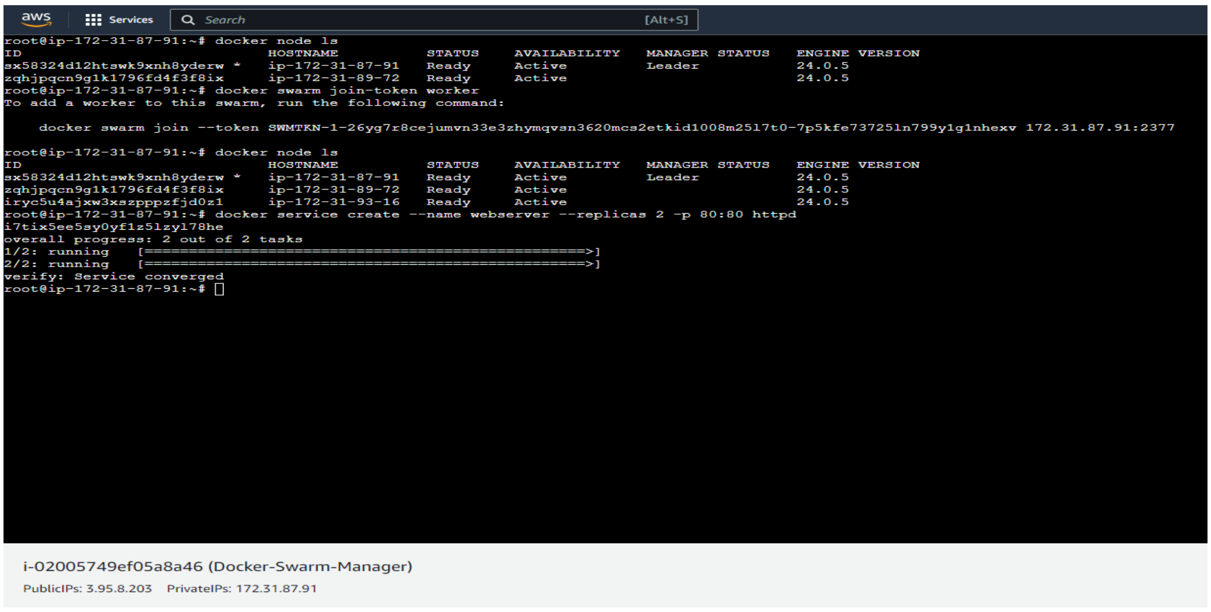
```
Welcome to Ubuntu 22.04.2 LTS (GNU/Linux 5.19.0-1025-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  System information as of Thu Jul 27 13:04:44 UTC 2023

  System load:  0.080078125       Processes:                98
  Usage of /:   29.7% of 7.57GB   Users logged in:          0
  Memory usage: 30%               IPv4 address for docker0: 172.17.0.1
  Swap usage:   0%                IPv4 address for eth0:    172.31.93.16


Expanded Security Maintenance for Applications is not enabled.

86 updates can be applied immediately.
55 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status


Last login: Thu Jul 27 13:04:45 2023 from 18.206.107.27
root@ip-172-31-93-16:~# docker swarm join --token SWMTKN-1-26yg7r8cejumvn33e3zhymqvsn3620mcs2etkid1008m2517t0-7p5kfe737251n799y1g1nhexv 172.31.87.91:2377
This node joined a swarm as a worker.
root@ip-172-31-93-16:~# 
```

i-0c747f1b3df7bde10 (Worker_Two)
PublicIPs: 54.237.172.32   PrivateIPs: 172.31.93.16



```
root@ip-172-31-87-91:~# docker node ls
ID                            HOSTNAME           STATUS    AVAILABILITY   MANAGER STATUS   ENGINE VERSION
sx58324d12htswk9xnh8yderw *   ip-172-31-87-91    Ready     Active         Leader           24.0.5
zqhjpqcn9g1k1796fd4f3f8ix     ip-172-31-89-72    Ready     Active                          24.0.5
root@ip-172-31-87-91:~# docker swarm join-token worker
To add a worker to this swarm, run the following command:

    docker swarm join --token SWMTKN-1-26yg7r8cejumvn33e3zhymqvsn3620mcs2etkid1008m2517t0-7p5kfe737251n799y1g1nhexv 172.31.87.91:2377

root@ip-172-31-87-91:~# docker node ls
ID                            HOSTNAME           STATUS    AVAILABILITY   MANAGER STATUS   ENGINE VERSION
sx58324d12htswk9xnh8yderw *   ip-172-31-87-91    Ready     Active         Leader           24.0.5
zqhjpqcn9g1k1796fd4f3f8ix     ip-172-31-89-72    Ready     Active                          24.0.5
iryc5u4ajxw3xszpppzfjd0z1     ip-172-31-93-16    Ready     Active                          24.0.5
root@ip-172-31-87-91:~# 
```

i-02005749ef05a8a46 (Docker-Swarm-Manager)
PublicIPs: 3.95.8.203    PrivateIPs: 172.31.87.91



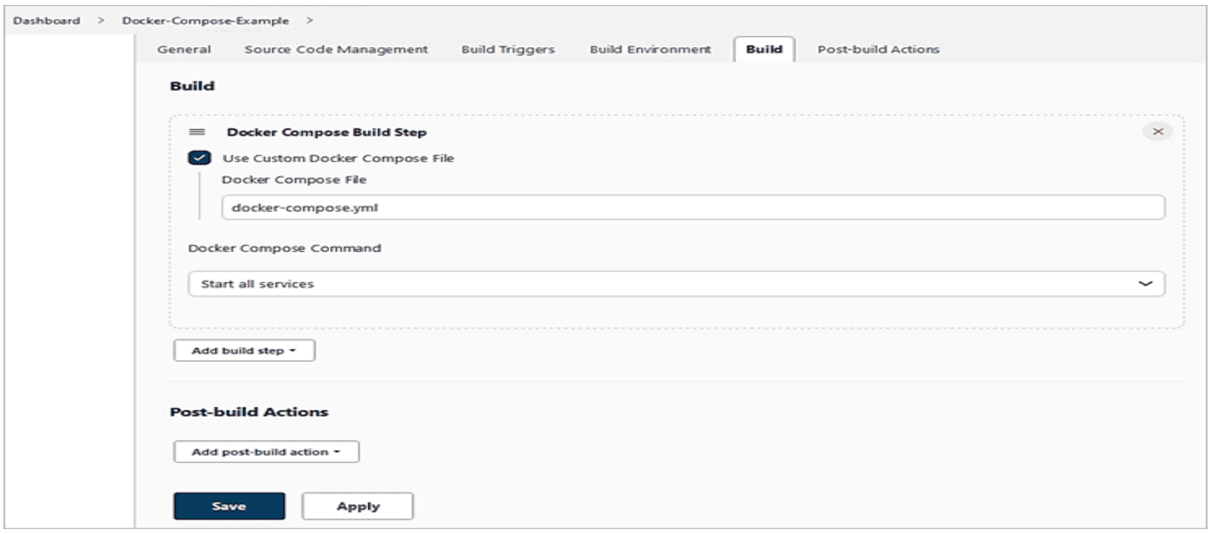
```
root@ip-172-31-87-91:~# docker node ls
ID                            HOSTNAME           STATUS    AVAILABILITY   MANAGER STATUS   ENGINE VERSION
sx58324d12htswk9xnh8yderw *   ip-172-31-87-91    Ready     Active         Leader           24.0.5
zqhjpqcn9g1k1796fd4f3f8ix     ip-172-31-89-72    Ready     Active                          24.0.5
root@ip-172-31-87-91:~# docker swarm join-token worker
To add a worker to this swarm, run the following command:

    docker swarm join --token SWMTKN-1-26yg7r8cejumvn33e3zhymqvsn3620mcs2etkid1008m2517t0-7p5kfe737251n799y1g1nhexv 172.31.87.91:2377

root@ip-172-31-87-91:~# docker node ls
ID                            HOSTNAME           STATUS    AVAILABILITY   MANAGER STATUS   ENGINE VERSION
sx58324d12htswk9xnh8yderw *   ip-172-31-87-91    Ready     Active         Leader           24.0.5
zqhjpqcn9g1k1796fd4f3f8ix     ip-172-31-89-72    Ready     Active                          24.0.5
iryc5u4ajxw3xszpppzfjd0z1     ip-172-31-93-16    Ready     Active                          24.0.5
root@ip-172-31-87-91:~# docker service create --name webserver --replicas 2 -p 80:80 httpd
i7tix5ee5sy0yf1z5lzyl78he
overall progress: 2 out of 2 tasks
1/2: running   [==================================================>]
2/2: running   [==================================================>]
verify: Service converged
root@ip-172-31-87-91:~# 
```

i-02005749ef05a8a46 (Docker-Swarm-Manager)
PublicIPs: 3.95.8.203    PrivateIPs: 172.31.87.91

## Jenkins Job Using Docker Compose Plugin

Let’s look at adding a Build step in Jenkins to call the docker-compose.yml file and start all the services defined in the file as part of the Docker Compose Command. The docker-compose.yml file is present in the root of the GitHub repository.

The rest of the freestyle job configuration for SCM definition remains the same as in the previous section.


Dashboard > Docker-Compose-Example >

General | Source Code Management | Build Triggers | Build Environment | Build | Post-build Actions

**Build**

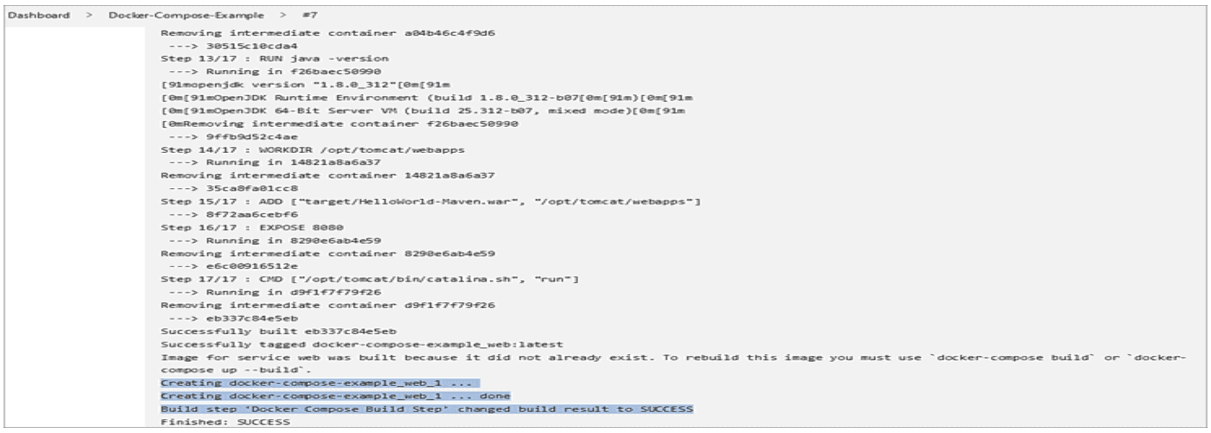
Docker Compose Build Step

[✓] Use Custom Docker Compose File

Docker Compose File: docker-compose.yml

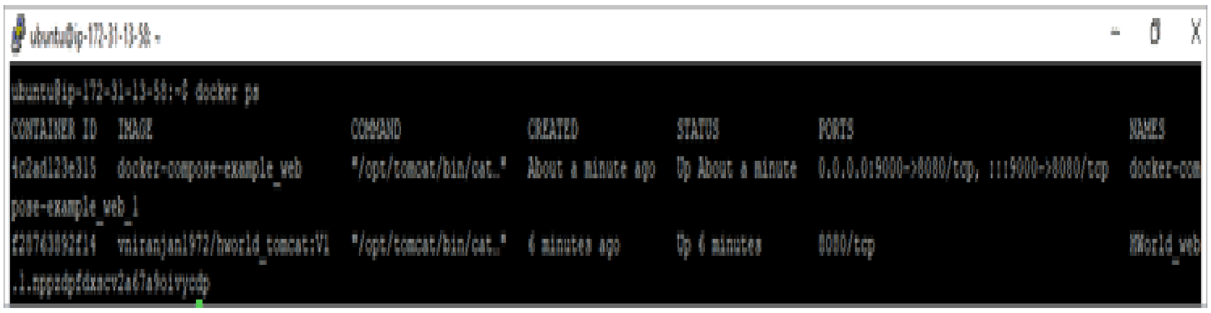
Docker Compose Command: Start all services

Click on Save and trigger a build. Once done, look at the console output and the services running in the docker host.


Dashboard > Docker-Compose-Example > #7

```
Removing intermediate container a04b46c4f9d6
 ---> 30515c10cda4
Step 13/17 : RUN java -version
 ---> Running in f26baec50990
[91mopenjdk version "1.8.0_312"[0m[91m
[0m[91mOpenJDK Runtime Environment (build 1.8.0_312-b07[0m[91m)[0m[91m
[0m[91mOpenJDK 64-Bit Server VM (build 25.312-b07, mixed mode)[0m[91m
[0mRemoving intermediate container f26baec50990
 ---> 9ffb9d52c4ae
Step 14/17 : WORKDIR /opt/tomcat/webapps
 ---> Running in 14821a8a6a37
Removing intermediate container 14821a8a6a37
 ---> 35ca8fa01cc8
Step 15/17 : ADD ["target/HelloWorld-Maven.war", "/opt/tomcat/webapps"]
 ---> 8f72aa6cebf6
Step 16/17 : EXPOSE 8080
 ---> Running in 8290e6ab4e59
Removing intermediate container 8290e6ab4e59
 ---> e6c00916512e
Step 17/17 : CMD ["/opt/tomcat/bin/catalina.sh", "run"]
 ---> Running in d9f1f7f79f26
Removing intermediate container d9f1f7f79f26
 ---> eb337c84e5eb
Successfully built eb337c84e5eb
Successfully tagged docker-compose-example_web:latest
Image for service web was built because it did not already exist. To rebuild this image you must use `docker-compose build` or `docker-
compose up --build`.
Creating docker-compose-example_web_1 ...
Creating docker-compose-example_web_1 ... done
Build step 'Docker Compose Build Step' changed build result to SUCCESS
Finished: SUCCESS
```

Run ‘docker ps’ command to look at services as defined in docker-compose.yml file.


ubuntu@ip-172-31-13-58: ~

```
ubuntu@ip-172-31-13-58:~$ docker ps
CONTAINER ID   IMAGE                          COMMAND                  CREATED              STATUS              PORTS                                       NAMES
4c2ad123e315   docker-compose-example_web     "/opt/tomcat/bin/cat…"   About a minute ago   Up About a minute   0.0.0.0:9000->8080/tcp, :::9000->8080/tcp   docker-com
pose-example_web_1
f20743092f14   vniranjan1972/hworld_tomcat:V1 "/opt/tomcat/bin/cat…"   6 minutes ago        Up 6 minutes        8080/tcp                                    HWorld_web
.1.nppzdpfdxacv2a67a9oivycdp
```

The service contains tasks that need to be executed on Manager and Worker nodes. In this section, we will look at how using Jenkins as a Docker Stack can be used to deploy multiple services that are primarily containers across different machines. The services run as part of the stack can also be configured across multiple replicas.

Docker stack makes use of a YAML file to deploy multiple services. In this example, I am using the below docker-compose.yml file and a 3 node cluster.

```
version: ‘3.8’
services:
web:
image: vniranjan1972/hworld_tomcat:V1(Sample User Name)
ports:
– 9000:8080
```

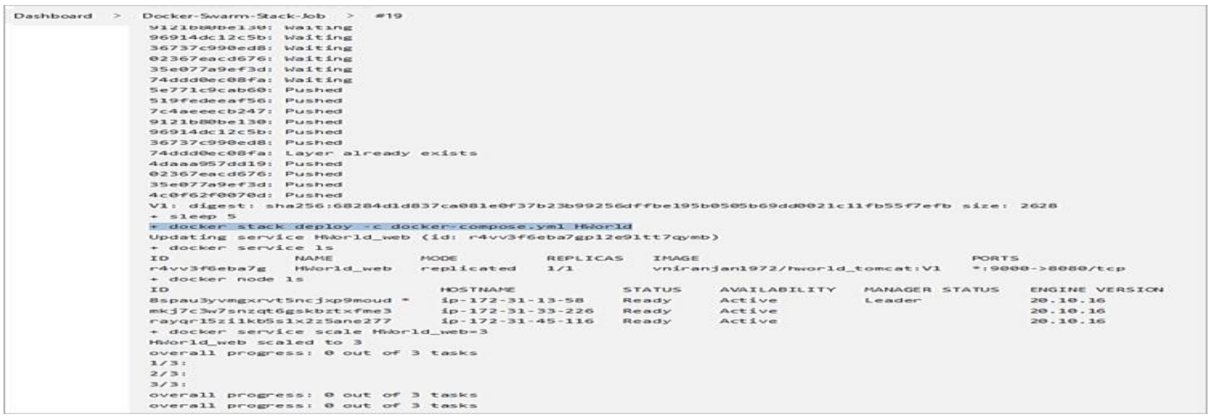
In the YAML file, I am using a built image since the build command is not supported by docker stack deploy. The Dockerfile is the same as in section 2.2.

In Jenkins, use a Execute shell Build step and add the below commands:

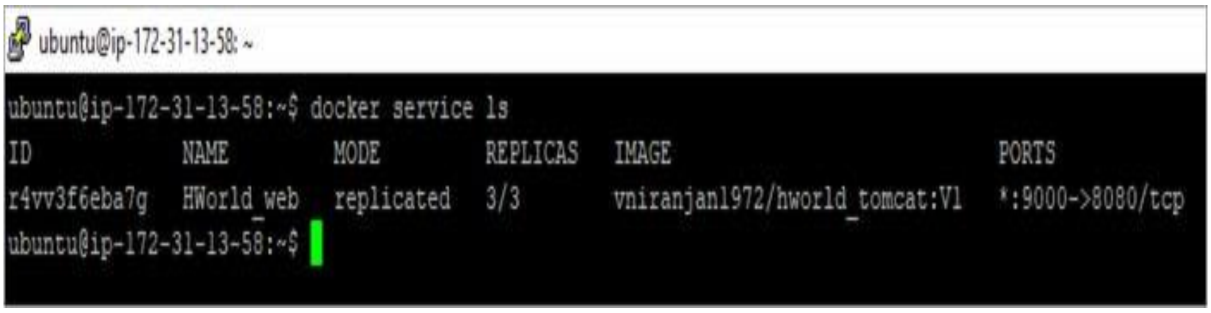
```
docker login -u “vniranjan1972” -p “<Docker Hub Token>”
docker build -t vniranjan1972/hworld_tomcat:V1 .
docker push vniranjan1972/hworld_tomcat:V1
sleep 5
docker stack deploy -c docker-compose.yml HWorld
docker service ls
docker node ls
docker service scale HWorld_web=3
docker service ps HWorld_web
```

Save the Job and trigger a build. The job is run on the manager node.

Console Output



```
9121b80be130: Waiting
96914dc12c5b: Waiting
36737c990ed8: Waiting
02367eacd676: Waiting
35e077a9ef3d: Waiting
74ddd0ec08fa: Waiting
5e771c9cab60: Pushed
519fedeeaf56: Pushed
7c4aeeecb247: Pushed
9121b80be130: Pushed
96914dc12c5b: Pushed
36737c990ed8: Pushed
74ddd0ec08fa: Layer already exists
4daaa957dd19: Pushed
02367eacd676: Pushed
35e077a9ef3d: Pushed
4c0f62f0070d: Pushed
V1: digest: sha256:68284d1d837ca081e0f37b23b99256dffbe195b0505b69dd0021c11fb55f7efb size: 2628
+ sleep 5
+ docker stack deploy -c docker-compose.yml HWorld
Updating service HWorld_web (id: r4vv3f6eba7gp12e91tt7qymb)
+ docker service ls
ID             NAME         MODE         REPLICAS   IMAGE                             PORTS
r4vv3f6eba7g   HWorld_web   replicated   1/1        vniranjan1972/hworld_tomcat:V1    *:9000->8080/tcp
+ docker node ls
ID                            HOSTNAME            STATUS    AVAILABILITY   MANAGER STATUS   ENGINE VERSION
8spau3yvmgxrvt5ncjxp9moud *   ip-172-31-13-58     Ready     Active         Leader           20.10.16
mkj7c3w7snzqt6gskbztxfme3     ip-172-31-33-226    Ready     Active                          20.10.16
rayqr15zi1kb5s1x2z5ane277     ip-172-31-45-116    Ready     Active                          20.10.16
+ docker service scale HWorld_web=3
HWorld_web scaled to 3
overall progress: 0 out of 3 tasks
1/3:
2/3:
3/3:
overall progress: 0 out of 3 tasks
overall progress: 0 out of 3 tasks
```



```
ubuntu@ip-172-31-13-58:~$ docker service ls
ID             NAME         MODE         REPLICAS   IMAGE                            PORTS
r4vv3f6eba7g   HWorld_web   replicated   3/3        vniranjan1972/hworld_tomcat:V1   *:9000->8080/tcp
ubuntu@ip-172-31-13-58:~$
```

As you can see, the service is deployed across the nodes in the cluster.