

DOCKER SWARM REPLICATION

Steps to create replicas

Step 1:

1. First create three instances with allowing all traffic to that instance
2. Install the docker and start the docker service in every instance
3. In manager server generate the token by the command

```
docker swarm init
```

4. Copy the token as shown in below figure

```
[root@ip-172-31-3-112 ~]# docker swarm init
Swarm initialized: current node (na3a1ka7abcswn6b3neu7c1ke) is now a manager.

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

    docker swarm join --token SWMTKN-1-3mr8vk61q2zrrahvxz5lh8c9685hp7nm7evlzs3kschccuky8t-a92vrc5hka2b4h9e4jeivmh9n 172.31.3.112:2377

To add a manager to this swarm, run 'docker swarm join-token manager' and follow the instructions.

[root@ip-172-31-3-112 ~]#
```

5. Paste the copied key in second server and third server it will create the worker node in second and third servers as shown in below figure

```
Complete!
[root@ip-172-31-1-252 ~]# systemctl start docker
[root@ip-172-31-1-252 ~]# docker swarm join --token SWMTKN-1-3mr8vk61q2zrrahvxz5lh8c9685hp7nm7evlzs3kschccuky8t-a92vrc5hka2b4h9e4jeivmh9n 172.31.3.112:2377
This node joined a swarm as a worker.
[root@ip-172-31-1-252 ~]#
```

6. We have to create the service in manager node with replicas by the command is

```
docker service create --name Netflix --replicas 3 --publish 8080:80 httpd
```

It will create the service and the manager can distribute the service equally to the workers nodes

7. We create three services from manager server one service is created in manager node and second service is created in first worker node and third service is created in second worker node as shown in below figure

```
[root@ip-172-31-3-112 ~]# docker swarm init
Swarm initialized: current node (na3a1ka7abcswn6b3neu7c1ke) is now a manager.

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

    docker swarm join --token SWMTKN-1-3mr8vk61q2zrrahvxz5lh8c9685hp7nm7evlzs3kschccuky8t-a92vrc5hka2b4h9e4jeivmh9n 172.31.3.112:2377

To add a manager to this swarm, run 'docker swarm join-token manager' and follow the instructions.

[root@ip-172-31-3-112 ~]# docker service create --name netflix --replicas 3 --publish 8080:80 httpd
b4sunyxernhtqkawai9xt38uk
overall progress: 3 out of 3 tasks
1/3: running [=====>]
2/3: running [=====>]
3/3: running [=====>]
verify: Service converged
[root@ip-172-31-3-112 ~]# docker ps
CONTAINER ID   IMAGE      COMMAND                  CREATED        STATUS        PORTS          NAMES
40f742ee2dfe   httpd:latest  "httpd-foreground"      26 seconds ago Up 24 seconds  80/tcp        netflix.3.vib6bgo2oee1h0sigb9mqns0
[root@ip-172-31-3-112 ~]#
```

Step 2:

1. Up to now we get the image from docker hub but know we can create the image
2. By using Dockerfile we can create the image as shown in below figure

```
FROM ubuntu
RUN apt update -y
RUN apt install apache2 -y
COPY index.html /var/www/html
CMD ["/usr/sbin/apachectl", "-D", "FOREGROUND"]
```

3. Type html page for login and copy the html code from w3 school

4. Create index.html file in manager node with **vi index.html** and paste the copied html code in editor and save it
5. Build the image by using command is

```
docker build -t image .
```

It will build the image as shown in below figure

```
[root@ip-172-31-3-112 ~]# docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS          NAMES
b3ffc431dc6a   httpd:latest   "httpd-foreground"      5 minutes ago Up 5 minutes   80/tcp         hotstar.2.4quuo8hmgte8dh0w103ko3end
40f742ee2dfe   httpd:latest   "httpd-foreground"      29 minutes ago Up 29 minutes   80/tcp         netflix.3.vib6bgo2oee1h0s igb9mqns0

[root@ip-172-31-3-112 ~]# vi index.html
[root@ip-172-31-3-112 ~]# vi Dockerfile
[root@ip-172-31-3-112 ~]# docker build -t image .
[+] Building 26.2s (9/9) FINISHED                                docker:default
=> [internal] load build definition from Dockerfile                0.0s
=> => transferring dockerfile: 235B                                0.0s
=> [internal] load metadata for docker.io/library/ubuntu:latest   2.1s
=> [internal] load .dockerignore                                  0.0s
=> => transferring context: 2B                                       0.0s
=> [1/4] FROM docker.io/library/ubuntu:latest@sha256:99c35190e22d294cdace2783ac55effc69d32896daaa265f0bbdbcd4fbc3e5 2.4s
=> => resolve docker.io/library/ubuntu:latest@sha256:99c35190e22d294cdace2783ac55effc69d32896daaa265f0bbdbcd4fbc3e5 0.0s
=> => sha256:99c35190e22d294cdace2783ac55effc69d32896daaa265f0bbdbcd4fbc3e5 6.69kB / 6.69kB 0.0s
=> => sha256:5d070ad5f7fe63623cbb99b4fc0fd997f5591303d4b03ccce50f403957d0ddc4 424B / 424B 0.0s
=> => sha256:59ab366372d56772eb54e426183435e6b0642152cb449ec7ab52473af8ca6e3f 2.30kB / 2.30kB 0.0s
=> => sha256:ff65ddf9395be21bfe1f320b7705e539ee44c1053034f801b1a3cbbf2d0f4056 29.75MB / 29.75MB 0.5s
=> => extracting sha256:ff65ddf9395be21bfe1f320b7705e539ee44c1053034f801b1a3cbbf2d0f4056 1.7s
=> [internal] load build context                                0.0s
=> => transferring context: 3.97kB                                0.0s
=> [2/4] RUN apt update -y                                     6.2s
=> [3/4] RUN apt install apache2 -y                           13.5s
=> [4/4] COPY index.html /var/www/html                        0.1s
=> exporting to image                                         1.8s
=> => exporting layers                                             1.8s
=> => writing image sha256:a57de728733a401e847e8dbb1087b11e5771a4fd859427916e08c3578e883b 0.0s
=> => naming to docker.io/library/image                          0.0s
[root@ip-172-31-3-112 ~]#
```

6. To create the service with created image by the command is

```
Docker service create --name Gpay --replicas 3 --publish 84:80 image
```

As shown in below figure

```
[root@ip-172-31-3-112 ~]# docker service create --name Gpay --replicas 3 --publish 84:80 image
image:latest could not be accessed on a registry to record
its digest. Each node will access image:latest independently,
possibly leading to different nodes running different
versions of the image.

ti3x5i6rgbxqy4c3lem7xo1aw
overall progress: 3 out of 3 tasks
1/3: running [=====>]
2/3: running [=====>]
3/3: running [=====>]
verify: Service converged
```

Step 3:

1. Go to docker hub and create the repository in docker hub

2. In manager node type **docker login** command it will ask credentials of docker hub
3. Provide user name and password of docker hub as shown in below figure

```
[root@ip-172-31-3-112 ~]# docker login
Log in with your Docker ID or email address to push and pull images from Docker Hub. If you don't have a Docker ID, go to hub.docker.com/ to create one.
You can log in with your password or a Personal Access Token (PAT). Using a limited-scope PAT grants access only to the repositories you create and to the repositories you have write access to. Organizations using SSO. Learn more at https://docs.docker.com/go/access-tokens/

Username: abeed123
Password:
WARNING! Your password will be stored unencrypted in /root/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
[root@ip-172-31-3-112 ~]# docker tag image abeed123/docker-swarm-repo
[root@ip-172-31-3-112 ~]# docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
abeed123/docker-swarm-repo	latest	a57de728733a	47 minutes ago	227MB
image	latest	a57de728733a	47 minutes ago	227MB
httpd	<none>	1bcf11fa154f	3 months ago	148MB

```
[root@ip-172-31-3-112 ~]#
```

4. Check the images by using command **docker images** and copy the recently created image
5. Type **docker push abeed123/docker-swarm-repo** it will push to docker hub
6. Now create the service by command

docker service create --name paytm --replicas 10 --publish 90:80 image

It will create now effectively

7. If u stop the container wontedly then that container will start immediately in the server