

Creating the Container by using Service

1. Create the Image.
2. Tag the Image.
3. Push the Image to Docker Hub.
4. Create the Node.
5. Create the service.

1.Create Image: To create images we should have a Docker file. {vi Docker file. <" D must be in Caps">}

Example: inside the docker file.

```
root@ip-172-31-26-39:~  
FROM ubuntu  
RUN apt update -y  
RUN apt install nginx -y  
COPY index.html /var/www/html  
EXPOSE 80  
CMD ["nginx", "-g", "daemon off;"]  
~  
~  
~
```

Save and exit.

Now create the file as shown above index.html.

Now create the image {docker build -t image1 space (.}
(Here is image1: image name you to create.)

Before creating the tag first, we have to create the repository in Docker Hub.

2.Tag the Image: To create the tag we must use command called {`docker tag image1 username/repository name`}.

(Here repository name: name of the repository in docker hub).

```

rw-r--r--. 1 root root 127 Nov 20 15:42 Dockerfile
-rwxr-xr-x. 2 root root 6 Nov 20 08:03 docker
rw-r--r--. 1 root root 1493 Nov 20 13:56 index.html
root@ip-172-31-26-39 ~]# vi Dockerfile
root@ip-172-31-26-39 ~]# vi Dockerfile
root@ip-172-31-26-39 ~]# docker build -t task1 .
+ Building 0.9s (9/9) FINISHED
docker:default
=> [internal] load build definition from Dockerfile
0.0s
=> transferring dockerfile: 229B
0.0s
=> [internal] load metadata for docker.io/library/ubuntu:latest
0.8s
=> [internal] load .dockerignore
0.0s
=> transferring context: 2B
0.0s
=> [1/4] FROM docker.io/library/ubuntu:latest@sha256:278628f08d4979fb9af9ead44277dbc9c92c2465922310916ad0c46ec9999295
0.0s
=> [internal] load build context
0.0s
=> transferring context: 92B
0.0s
=> CACHED [2/4] RUN apt update -y
0.0s
=> CACHED [3/4] RUN apt install nginx -y
0.0s
=> CACHED [4/4] COPY index.html /var/www/html
0.0s
=> exporting to image
0.0s
=> exporting layers
0.0s
=> writing image sha256:1db77e2b4e4a2383b756c7ae4cf470d3a00abd6517d7a3650de5303c123ad99a
0.0s

```

Here you will get the tag of the image in the form of `username/repository name`.

Before pushing you must log the docker hub {docker login}

Username: your docker username.

Password: your docker password

3.Push the Image to Docker Hub:

By using the tag we are pushing the image to Docker hub.

Docker push username/repository name {docker push username/repository name}

```
[root@ip-172-31-26-39 ~]# docker push praveenkumar8/task
Using default tag: latest
The push refers to repository [docker.io/praveenkumar8/task]
17ee86e79dc7: Mounted from praveenkumar8/docker
9cf52937e2b4: Mounted from praveenkumar8/docker
6bee6eed2d93: Mounted from praveenkumar8/docker
27123a71e85e: Mounted from praveenkumar8/docker
latest: digest: sha256:7a281ec2169bf0769d6a6120b0e923405ed655b63e7790b674f9f44694b834cb size: 1159
[root@ip-172-31-26-39 ~]#
```

4. Create the Node: To create Node we have to use the command called `{docker swarm init}`.

```
[root@ip-172-31-26-39 ~]# docker swarm init
Swarm initialized: current node (lxpuagagsdn8xlyxc9ltucjm3) is now a manager.

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

    docker swarm join --token SWMTKN-1-4bga20o2ynkuo7s9a4fifx2qn7ewfj33jwr05obxa2il79ku5e-bqesfiu4kqhvk4stt0rn6j3in 172.31.26.39:2377

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

[root@ip-172-31-26-39 ~]#
```

Now the Node is created.

```
[root@ip-172-31-26-39 ~]# docker node ls
```

ID	HOSTNAME	STATUS	AVAILABILITY	MANAGER STATUS	ENGINE VERSION
xpuagagsdn8xlyxc9ltucjm3 *	ip-172-31-26-39.ap-southeast-1.compute.internal	Ready	Active	Leader	25.0.6

```
[root@ip-172-31-26-39 ~]#
```

5.Create the service: By using the Node the Service will be created.

To create the service {**docker service create --name (container name) --publish 8000:80 httpd**}

```
[root@ip-172-31-26-39 ~]# docker service create --name book-my-show --publish 8000:80 httpd
pp15216q4lh2yuvbn02lcx2os
overall progress: 1 out of 1 tasks
1/1: running [=====>]
verify: Service converged
[root@ip-172-31-26-39 ~]# docker service ls
```

ID	NAME	MODE	REPLICAS	IMAGE	PORTS
pp15216q4lh2	book-my-show	replicated	1/1	httpd:latest	*:8000->80/tcp

```
[root@ip-172-31-26-39 ~]#
```

Along with the service Container also creates as shown below.

```
[root@ip-172-31-26-39 ~]# docker ps
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
c6cd9c70019f	httpd:latest	"httpd-foreground"	11 minutes ago	Up 11 minutes	80/tcp	book-my-show.1.9cglvf98oy81c6t92cieu7pg9
43a1b53a49b0	nginx	"/docker-entrypoint..."	9 hours ago	Up 8 hours	0.0.0.0:8001->80/tcp, :::8001->80/tcp	my_webapp2
78a3f09908ec	nginx	"/docker-entrypoint..."	9 hours ago	Up 8 hours	0.0.0.0:8000->80/tcp, :::8000->80/tcp	my_webapp1

```
[root@ip-172-31-26-39 ~]#
```

