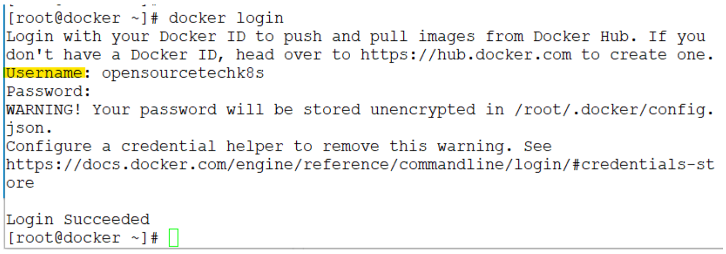
Docker LAB

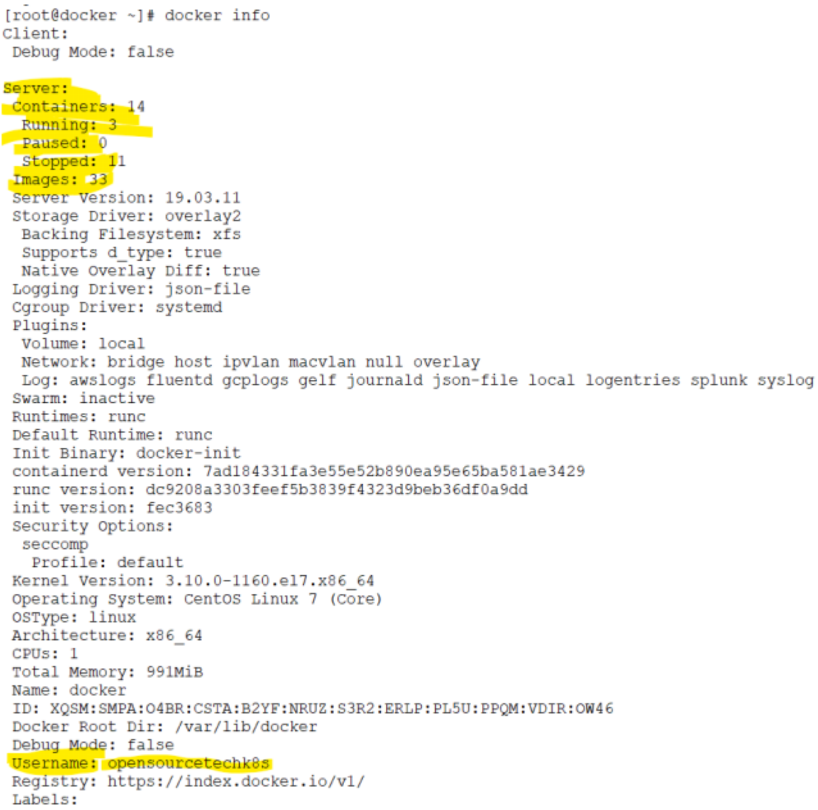
1) **Create docker id and login**

- **Create a id on docker HUB**
- **Login with that id on your local docker instance**



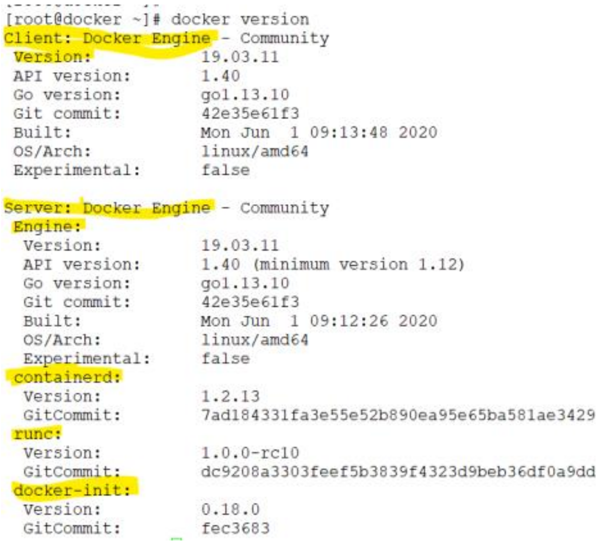
```
[root@docker ~]# docker login
Login with your Docker ID to push and pull images from Docker Hub. If you
don't have a Docker ID, head over to https://hub.docker.com to create one.
Username: opensourcetechk8s
Password:
WARNING! Your password will be stored unencrypted in /root/.docker/config.
json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-st
ore

Login Succeeded
[root@docker ~]#
```

- **Verify you login and state of docker resources (container, Images, storage)**



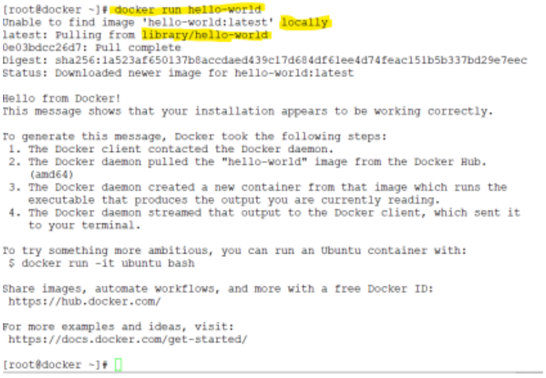
```
[root@docker ~]# docker info
Client:
 Debug Mode: false

Server:
 Containers: 14
  Running: 3
  Paused: 0
  Stopped: 11
 Images: 33
 Server Version: 19.03.11
 Storage Driver: overlay2
  Backing Filesystem: xfs
  Supports d_type: true
  Native Overlay Diff: true
 Logging Driver: json-file
 Cgroup Driver: systemd
 Plugins:
  Volume: local
  Network: bridge host ipvlan macvlan null overlay
  Log: awslogs fluentd gcplogs gelf journald json-file local logentries splunk syslog
 Swarm: inactive
 Runtimes: runc
 Default Runtime: runc
 Init Binary: docker-init
 containerd version: 7ad184331fa3e55e52b890ea95e65ba581ae3429
 runc version: dc9208a3303feef5b3839f4323d9beb36df0a9dd
 init version: fec3683
 Security Options:
  seccomp
   Profile: default
 Kernel Version: 3.10.0-1160.el7.x86_64
 Operating System: CentOS Linux 7 (Core)
 OSType: linux
 Architecture: x86_64
 CPUs: 1
 Total Memory: 991MiB
 Name: docker
 ID: XQSM:SMPA:O4BR:CSTA:B2YF:NRUZ:S3R2:ERLP:PL5U:PPQM:VDIR:OW46
 Docker Root Dir: /var/lib/docker
 Debug Mode: false
 Username: opensourcetechk8s
 Registry: https://index.docker.io/v1/
 Labels:
```

- **Check docker version and list of components used by Docker**
  - These are the key features docker provides in comparison with LXC



```
[root@docker ~]# docker version
Client: Docker Engine - Community
 Version:           19.03.11
 API version:       1.40
 Go version:        go1.13.10
 Git commit:        42e35e61f3
 Built:             Mon Jun  1 09:13:48 2020
 OS/Arch:           linux/amd64
 Experimental:      false

Server: Docker Engine - Community
 Engine:
  Version:          19.03.11
  API version:      1.40 (minimum version 1.12)
  Go version:       go1.13.10
  Git commit:       42e35e61f3
  Built:            Mon Jun  1 09:12:26 2020
  OS/Arch:          linux/amd64
  Experimental:     false
 containerd:
  Version:          1.2.13
  GitCommit:        7ad184331fa3e55e52b890ea95e65ba581ae3429
 runc:
  Version:          1.0.0-rc10
  GitCommit:        dc9208a3303feef5b3839f4323d9beb36df0a9dd
 docker-init:
  Version:          0.18.0
  GitCommit:        fec3683
```

- **Run a command to test your docker installation and check the output:**
  - Always docker tries to get image locally, if it doesn't exist than docker will try to pull from docker HUB



```
[root@docker ~]# docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
0e03bdcc26d7: Pull complete
Digest: sha256:1a523af650137b8accdaed439c17d684df61ee4d74feac151b5b337bd29e7eec
Status: Downloaded newer image for hello-world:latest

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
 $ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
 https://hub.docker.com/

For more examples and ideas, visit:
 https://docs.docker.com/get-started/

[root@docker ~]# 
```

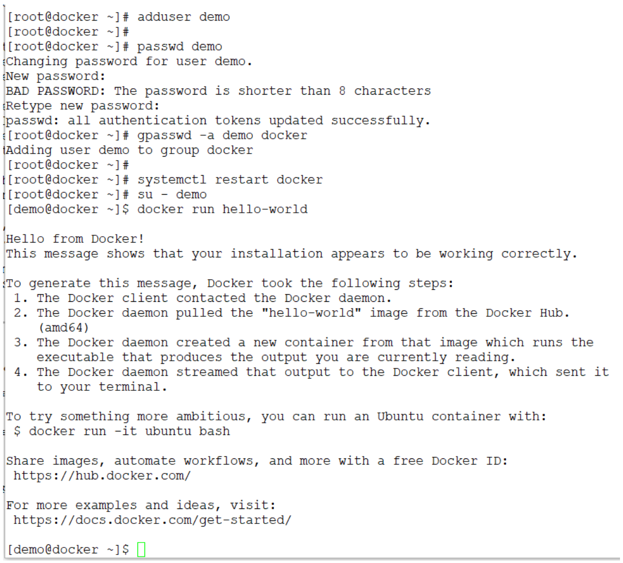
2) Run docker with non-root user

NOTE: The docker daemon binds to a Unix socket instead of a TCP port. By default that Unix socket is owned by the user root and other users can only access it using sudo.

The docker daemon always runs as the root user.

If you don't want to use sudo when you use the docker command, create a Unix group called docker and add users to it. When the docker daemon starts, it makes the ownership of the Unix socket read/writable by the docker group.

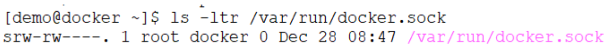
Warning: The docker group grants privileges equivalent to the root user. For details on how this impacts security in your system, see *Docker Daemon Attack Surface*.



```
[root@docker ~]# adduser demo
[root@docker ~]#
[root@docker ~]# passwd demo
Changing password for user demo.
New password:
BAD PASSWORD: The password is shorter than 8 characters
Retype new password:
passwd: all authentication tokens updated successfully.
[root@docker ~]# gpasswd -a demo docker
Adding user demo to group docker
[root@docker ~]#
[root@docker ~]# systemctl restart docker
[root@docker ~]# su - demo
[demo@docker ~]$ docker run hello-world

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
 $ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
 https://hub.docker.com/

For more examples and ideas, visit:
 https://docs.docker.com/get-started/

[demo@docker ~]$ 
```

- Check the ownership of the docker socket file

  # ls –ltr /var/run/docker.sock



```
[demo@docker ~]$ ls -ltr /var/run/docker.sock
srw-rw----. 1 root docker 0 Dec 28 08:47 /var/run/docker.sock
```

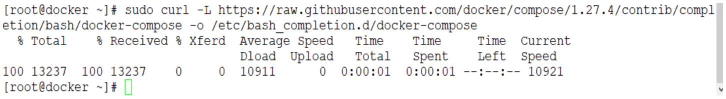
3) Docker command auto completion:

LINUX

1) On a current Linux OS (in a non-minimal installation), bash completion should be available.

2) Place the completion script in /etc/bash_completion.d/.

   # yum -y install bash-completion
   # sudo curl -L https://raw.githubusercontent.com/docker/compose/1.27.4/contrib/completion/bash/docker-compose -o /etc/bash_completion.d/docker-compose



```
[root@docker ~]# sudo curl -L https://raw.githubusercontent.com/docker/compose/1.27.4/contrib/compl
etion/bash/docker-compose -o /etc/bash_completion.d/docker-compose
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100 13237  100 13237    0     0  10911      0  0:00:01  0:00:01 --:--:-- 10921
[root@docker ~]# 
```

3) Logout and login again (reload bash)

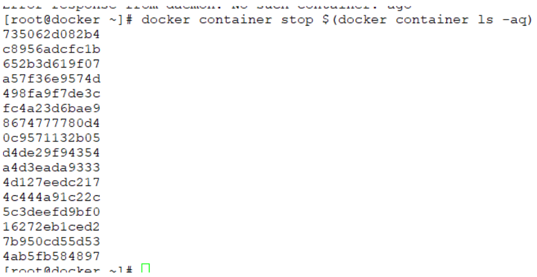
4) Run your first container and access services from outside

- **List all containers**

  # docker container ps
  # docker container ps -a

- **Stop all running containers**

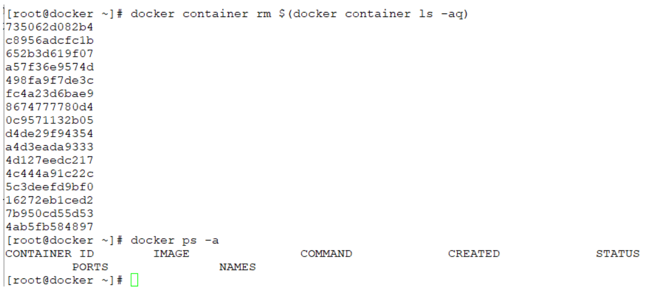
  # docker container stop $(docker container ls -aq)



```
Error response from daemon: No such container: ago
[root@docker ~]# docker container stop $(docker container ls -aq)
735062d082b4
c8956adcfc1b
652b3d619f07
a57f36e9574d
498fa9f7de3c
fc4a23d6bae9
8674777780d4
0c9571132b05
d4de29f94354
a4d3eada9333
4d127eedc217
4c444a91c22c
5c3deefd9bf0
16272eb1ced2
7b950cd55d53
4ab5fb584897
[root@docker ~]#
```

- **Delete all stopped containers**

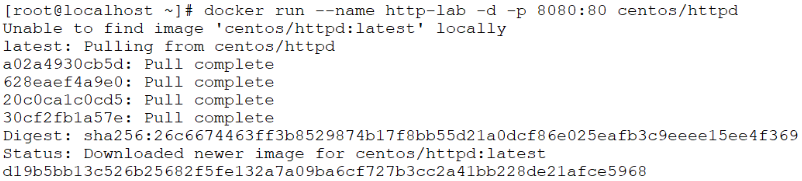
  # docker container rm $(docker container ls -aq)



```
[root@docker ~]# docker container rm $(docker container ls -aq)
735062d082b4
c8956adcfc1b
652b3d619f07
a57f36e9574d
498fa9f7de3c
fc4a23d6bae9
8674777780d4
0c9571132b05
d4de29f94354
a4d3eada9333
4d127eedc217
4c444a91c22c
5c3deefd9bf0
16272eb1ced2
7b950cd55d53
4ab5fb584897
[root@docker ~]# docker ps -a
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS
         PORTS               NAMES
[root@docker ~]#
```

- **Create a test container and run following commands**

  # docker run --name http-lab -d -p 8080:80 centos/httpd



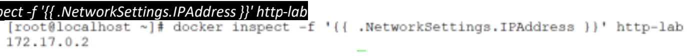
```
[root@localhost ~]# docker run --name http-lab -d -p 8080:80 centos/httpd
Unable to find image 'centos/httpd:latest' locally
latest: Pulling from centos/httpd
a02a4930cb5d: Pull complete
628eaef4a9e0: Pull complete
20c0ca1c0cd5: Pull complete
30cf2fb1a57e: Pull complete
Digest: sha256:26c6674463ff3b8529874b17f8bb55d21a0dcf86e025eafb3c9eeee15ee4f369
Status: Downloaded newer image for centos/httpd:latest
d19b5bb13c526b25682f5fe132a7a09ba6cf727b3cc2a41bb228de21afce5968
```

- ◆ Browse your container service using host ip and port
  http://192.168.1.74:8080/

5) Customize your web page running in container (web server)

- **Pull the container ip address and check the http webpage**

  # docker inspect -f '{{ .NetworkSettings.IPAddress }}' http-lab



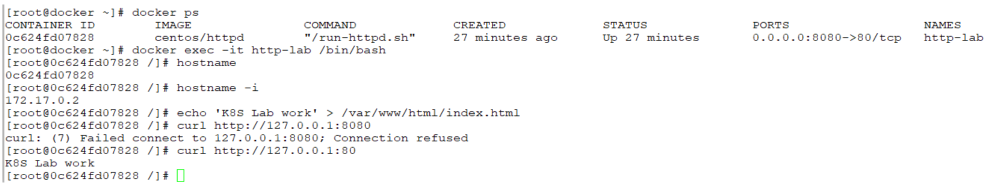
```
[root@localhost ~]# docker inspect -f '{{ .NetworkSettings.IPAddress }}' http-lab
172.17.0.2
```

- ○ Browse container service using containerip:containerport

  # curl http://172.17.0.2

- **Login in the container and modify your web page**

# docker exec -it http-lab /bin/bash
# hostname -i
# echo "k8s Lab Demo" > /var/www/html/index.html
# curl http://172.17.0.2



```
[root@docker ~]# docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS                  NAMES
0c624fd07828        centos/httpd        "/run-httpd.sh"     27 minutes ago      Up 27 minutes       0.0.0.0:8080->80/tcp   http-lab
[root@docker ~]# docker exec -it http-lab /bin/bash
[root@0c624fd07828 /]# hostname
0c624fd07828
[root@0c624fd07828 /]# hostname -i
172.17.0.2
[root@0c624fd07828 /]# echo 'K8S Lab work' > /var/www/html/index.html
[root@0c624fd07828 /]# curl http://127.0.0.1:8080
curl: (7) Failed connect to 127.0.0.1:8080; Connection refused
[root@0c624fd07828 /]# curl http://127.0.0.1:80
K8S Lab work
[root@0c624fd07828 /]#
```

- ◆ Browse your container service using host ip and port
  http://192.168.1.74:8080/

6) Play with docker images

- **Create image from running container** *(completed in above step)*

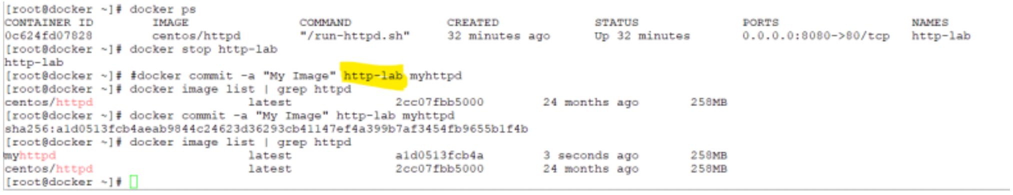
  - ◆ Browse your container service using host ip and port
    http://192.168.1.74:8080/

  - ◆ **Log out and stop the container (recommended)**

    # docker ps -a
    # # docker stop d19b5bb13c52

- **List and modify your image**

  # docker image ls
  # docker commit -a "my Image" http-lab myhttpd
  - -a for author
  - <name of the container> <name of the image [new]>



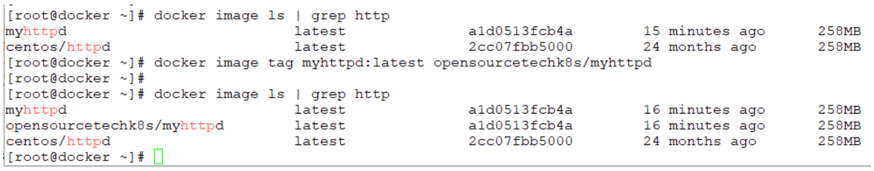
```
[root@docker ~]# docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS                  NAMES
0c624fd07828        centos/httpd        "/run-httpd.sh"     32 minutes ago      Up 32 minutes       0.0.0.0:8080->80/tcp   http-lab
[root@docker ~]# docker stop http-lab
http-lab
[root@docker ~]# #docker commit -a "My Image" http-lab myhttpd
[root@docker ~]# docker image list | grep httpd
centos/httpd                  latest              2cc07fbb5000        24 months ago       258MB
[root@docker ~]# docker commit -a "My Image" http-lab myhttpd
sha256:a1d0513fcb4aeab9844c24623d36293cb41147ef4a399b7af3454fb9655b1f4b
[root@docker ~]# docker image list | grep httpd
myhttpd                       latest              a1d0513fcb4a        3 seconds ago       258MB
centos/httpd                  latest              2cc07fbb5000        24 months ago       258MB
[root@docker ~]# 
```

- **Push image to docker**
  - Pre-requisite
    - Login on the docker hub
    - Create a repo on docker hub
      - Go to repositories -> create -> give repo name as "**myhttpd**" -> select public

□ **Tag your image with repository path is must**

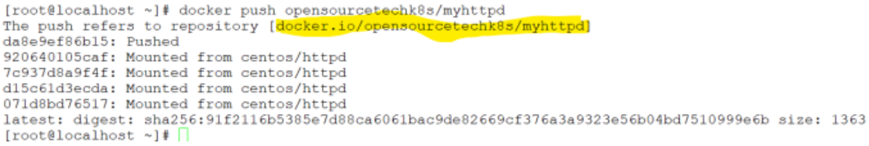
Syntax to tag image:
docker image tag SOURCE_IMAGE[:TAG] TARGET_IMAGE[:TAG]
- TARGET_IMAGE[:TAG] = **dockerid/repositoryname**

# docker image tag myhttpd opensourcetechk8s/myhttpd



```
[root@docker ~]# docker image ls | grep http
myhttpd                       latest              a1d0513fcb4a        15 minutes ago      258MB
centos/httpd                  latest              2cc07fbb5000        24 months ago       258MB
[root@docker ~]# docker image tag myhttpd:latest opensourcetechk8s/myhttpd
[root@docker ~]#
[root@docker ~]# docker image ls | grep http
myhttpd                       latest              a1d0513fcb4a        16 minutes ago      258MB
opensourcetechk8s/myhttpd     latest              a1d0513fcb4a        16 minutes ago      258MB
centos/httpd                  latest              2cc07fbb5000        24 months ago       258MB
[root@docker ~]# 
```

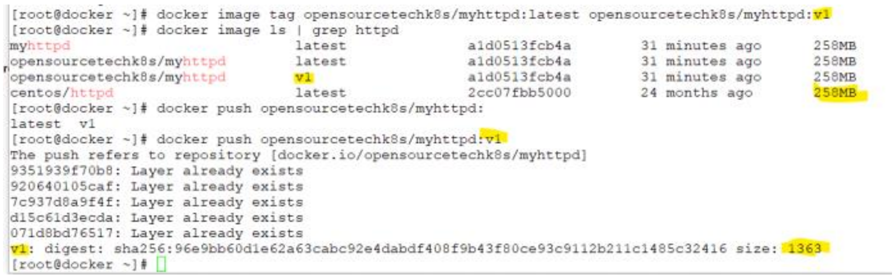
□ **Push image in your repo on docker hub**



```
[root@localhost ~]# docker push opensourcetechk8s/myhttpd
The push refers to repository [docker.io/opensourcetechk8s/myhttpd]
da8e9ef86b15: Pushed
920640105caf: Mounted from centos/httpd
7c937d8a9f4f: Mounted from centos/httpd
d15c61d3ecda: Mounted from centos/httpd
071d8bd76517: Mounted from centos/httpd
latest: digest: sha256:91f2116b5385e7d88ca6061bac9de82669cf376a3a9323e56b04bd7510999e6b size: 1363
[root@localhost ~]# 
```

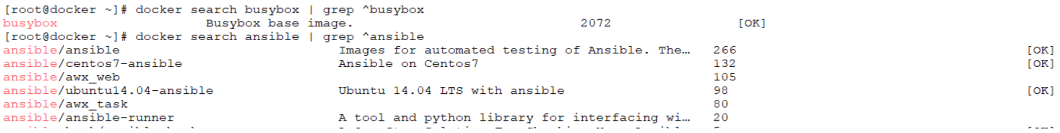
- **Tag your image for a new version of code**

docker image tag opensourcetechk8s/myhttpd:latest opensourcetechk8s/myhttpd:v1



```
[root@docker ~]# docker image tag opensourcetechk8s/myhttpd:latest opensourcetechk8s/myhttpd:v1
[root@docker ~]# docker image ls | grep httpd
myhttpd                       latest              a1d0513fcb4a        31 minutes ago      258MB
opensourcetechk8s/myhttpd     latest              a1d0513fcb4a        31 minutes ago      258MB
opensourcetechk8s/myhttpd     v1                  a1d0513fcb4a        31 minutes ago      258MB
centos/httpd                  latest              2cc07fbb5000        24 months ago       258MB
[root@docker ~]# docker push opensourcetechk8s/myhttpd:
latest  v1
[root@docker ~]# docker push opensourcetechk8s/myhttpd:v1
The push refers to repository [docker.io/opensourcetechk8s/myhttpd]
9351939f70b8: Layer already exists
920640105caf: Layer already exists
7c937d8a9f4f: Layer already exists
d15c61d3ecda: Layer already exists
071d8bd76517: Layer already exists
v1: digest: sha256:96e9bb60d1e62a63cabc92e4dabdf408f9b43f80ce93c9112b211c1485c32416 size: 1363
[root@docker ~]# 
```

- **How to search images -** search images on docker hub
  https://docs.docker.com/engine/reference/commandline/search/

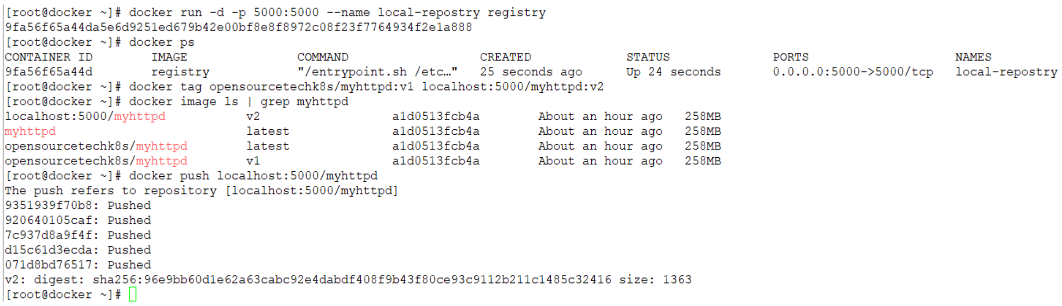


```
[root@docker ~]# docker search busybox | grep ^busybox
busybox                    Busybox base image.                                 2072                [OK]
[root@docker ~]# docker search ansible | grep ^ansible
ansible/ansible                                   Images for automated testing of Ansible. The…   266                                   [OK]
ansible/centos7-ansible                           Ansible on Centos7                              132                                   [OK]
ansible/awx_web                                                                                   105
ansible/ubuntu14.04-ansible                       Ubuntu 14.04 LTS with ansible                   98                                    [OK]
ansible/awx_task                                                                                  80
ansible/ansible-runner                            A tool and python library for interfacing wi…   20
```

Local Registry:

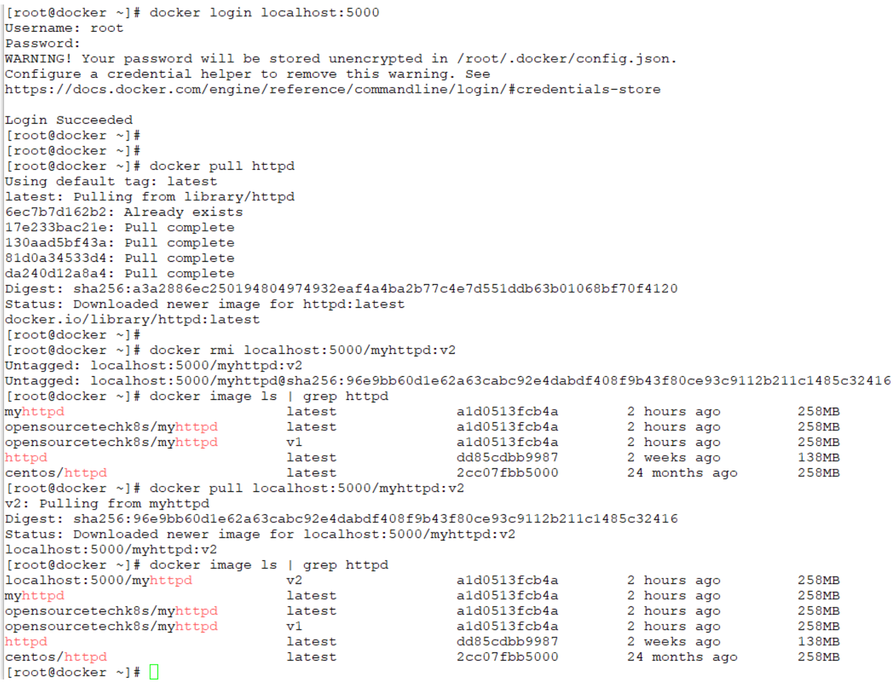
The registry image is configured to start on port 5000 in the container, so we will expose the host port also as 5000. Use be low command to download registry image from Docker HUB to Docker Host and launch a container.

# docker run -d -p <*host port:container port*> --name <*Container Name*> <*Image Name*>
- ◇ Host port = 5000
- ◇ Container port = 5000
- ◇ Container Name = local_registry
- ◇ Image Name = registry (not available locally so downloading from Docker HUB)



```
[root@docker ~]# docker run -d -p 5000:5000 --name local-repostry registry
9fa56f65a44da5e6d9251ed679b42e00bf8e8f8972c08f23f7764934f2e1a888
[root@docker ~]# docker ps
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS              PORTS                    NAMES
9fa56f65a44d        registry            "/entrypoint.sh /etc…"   25 seconds ago      Up 24 seconds       0.0.0.0:5000->5000/tcp   local-repostry
[root@docker ~]# docker tag opensourcetechk8s/myhttpd:v1 localhost:5000/myhttpd:v2
[root@docker ~]# docker image ls | grep myhttpd
localhost:5000/myhttpd           v2                  a1d0513fcb4a        About an hour ago   258MB
myhttpd                          latest              a1d0513fcb4a        About an hour ago   258MB
opensourcetechk8s/myhttpd        latest              a1d0513fcb4a        About an hour ago   258MB
opensourcetechk8s/myhttpd        v1                  a1d0513fcb4a        About an hour ago   258MB
[root@docker ~]# docker push localhost:5000/myhttpd
The push refers to repository [localhost:5000/myhttpd]
9351939f70b8: Pushed
920640105caf: Pushed
7c937d8a9f4f: Pushed
d15c61d3ecda: Pushed
071d8bd76517: Pushed
v2: digest: sha256:96e9bb60d1e62a63cabc92e4dabdf408f9b43f80ce93c9112b211c1485c32416 size: 1363
[root@docker ~]#
```

Browse registry with your docker host ip address : http://192.168.1.79:5000/v2/_catalog



```
[root@docker ~]# docker login localhost:5000
Username: root
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@docker ~]#
[root@docker ~]#
[root@docker ~]# docker pull httpd
Using default tag: latest
latest: Pulling from library/httpd
6ec7b7d162b2: Already exists
17e233bac21e: Pull complete
130aad5bf43a: Pull complete
81d0a34533d4: Pull complete
da240d12a8a4: Pull complete
Digest: sha256:a3a2886ec250194804974932eaf4a4ba2b77c4e7d551ddb63b01068bf70f4120
Status: Downloaded newer image for httpd:latest
docker.io/library/httpd:latest
[root@docker ~]#
[root@docker ~]# docker rmi localhost:5000/myhttpd:v2
Untagged: localhost:5000/myhttpd:v2
Untagged: localhost:5000/myhttpd@sha256:96e9bb60d1e62a63cabc92e4dabdf408f9b43f80ce93c9112b211c1485c32416
[root@docker ~]# docker image ls | grep httpd
myhttpd                          latest              a1d0513fcb4a        2 hours ago         258MB
opensourcetechk8s/myhttpd        latest              a1d0513fcb4a        2 hours ago         258MB
opensourcetechk8s/myhttpd        v1                  a1d0513fcb4a        2 hours ago         258MB
httpd                            latest              dd85cdbb9987        2 weeks ago         138MB
centos/httpd                     latest              2cc07fbb5000        24 months ago       258MB
[root@docker ~]# docker pull localhost:5000/myhttpd:v2
v2: Pulling from myhttpd
Digest: sha256:96e9bb60d1e62a63cabc92e4dabdf408f9b43f80ce93c9112b211c1485c32416
Status: Downloaded newer image for localhost:5000/myhttpd:v2
localhost:5000/myhttpd:v2
[root@docker ~]# docker image ls | grep httpd
localhost:5000/myhttpd           v2                  a1d0513fcb4a        2 hours ago         258MB
myhttpd                          latest              a1d0513fcb4a        2 hours ago         258MB
opensourcetechk8s/myhttpd        latest              a1d0513fcb4a        2 hours ago         258MB
opensourcetechk8s/myhttpd        v1                  a1d0513fcb4a        2 hours ago         258MB
httpd                            latest              dd85cdbb9987        2 weeks ago         138MB
centos/httpd                     latest              2cc07fbb5000        24 months ago       258MB
[root@docker ~]#
```

7) Persistent Disk

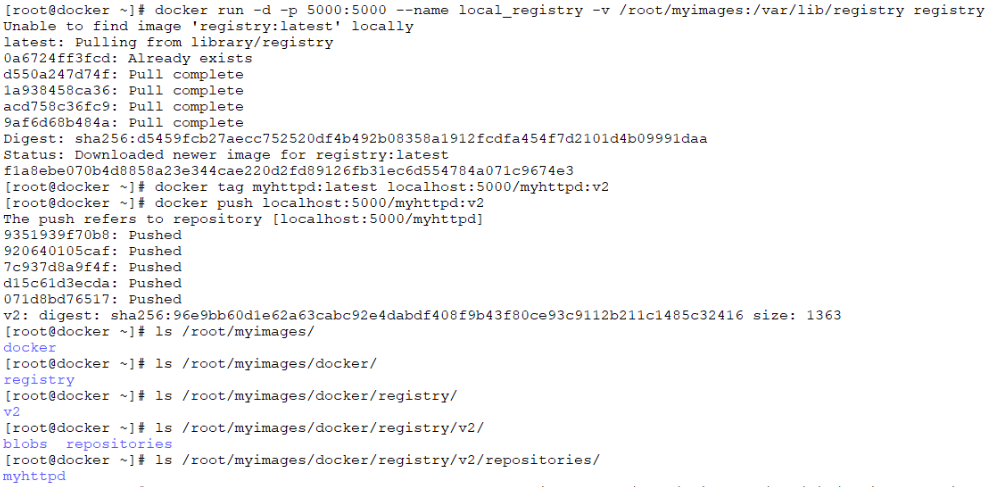
- For docker registry
  - Run private registry with persistent storage:

    # docker run -d -p 5000:5000 --name local_registry -v /root/myimage:/var/lib/registry registry

  - Tag image to push to local repository

    # docker image tag opensourcetechk8s/myhttpd:v1 localhost:5000/myhttpd:v2

    # docker push localhost:5000/myhttpd



```
[root@docker ~]# docker run -d -p 5000:5000 --name local_registry -v /root/myimages:/var/lib/registry registry
Unable to find image 'registry:latest' locally
latest: Pulling from library/registry
0a6724ff3fcd: Already exists
d550a247d74f: Pull complete
1a938458ca36: Pull complete
acd758c36fc9: Pull complete
9af6d68b484a: Pull complete
Digest: sha256:d5459fcb27aecc752520df4b492b08358a1912fcdfa454f7d2101d4b09991daa
Status: Downloaded newer image for registry:latest
f1a8ebe070b4d8858a23e344cae220d2fd89126fb31ec6d554784a071c9674e3
[root@docker ~]# docker tag myhttpd:latest localhost:5000/myhttpd:v2
[root@docker ~]# docker push localhost:5000/myhttpd:v2
The push refers to repository [localhost:5000/myhttpd]
9351939f70b8: Pushed
920640105caf: Pushed
7c937d8a9f4f: Pushed
d15c61d3ecda: Pushed
071d8bd76517: Pushed
v2: digest: sha256:96e9bb60d1e62a63cabc92e4dabdf408f9b43f80ce93c9112b211c1485c32416 size: 1363
[root@docker ~]# ls /root/myimages/
docker
[root@docker ~]# ls /root/myimages/docker/
registry
[root@docker ~]# ls /root/myimages/docker/registry/
v2
[root@docker ~]# ls /root/myimages/docker/registry/v2/
blobs  repositories
[root@docker ~]# ls /root/myimages/docker/registry/v2/repositories/
myhttpd
```

8) Networking

- Bridge
- Network