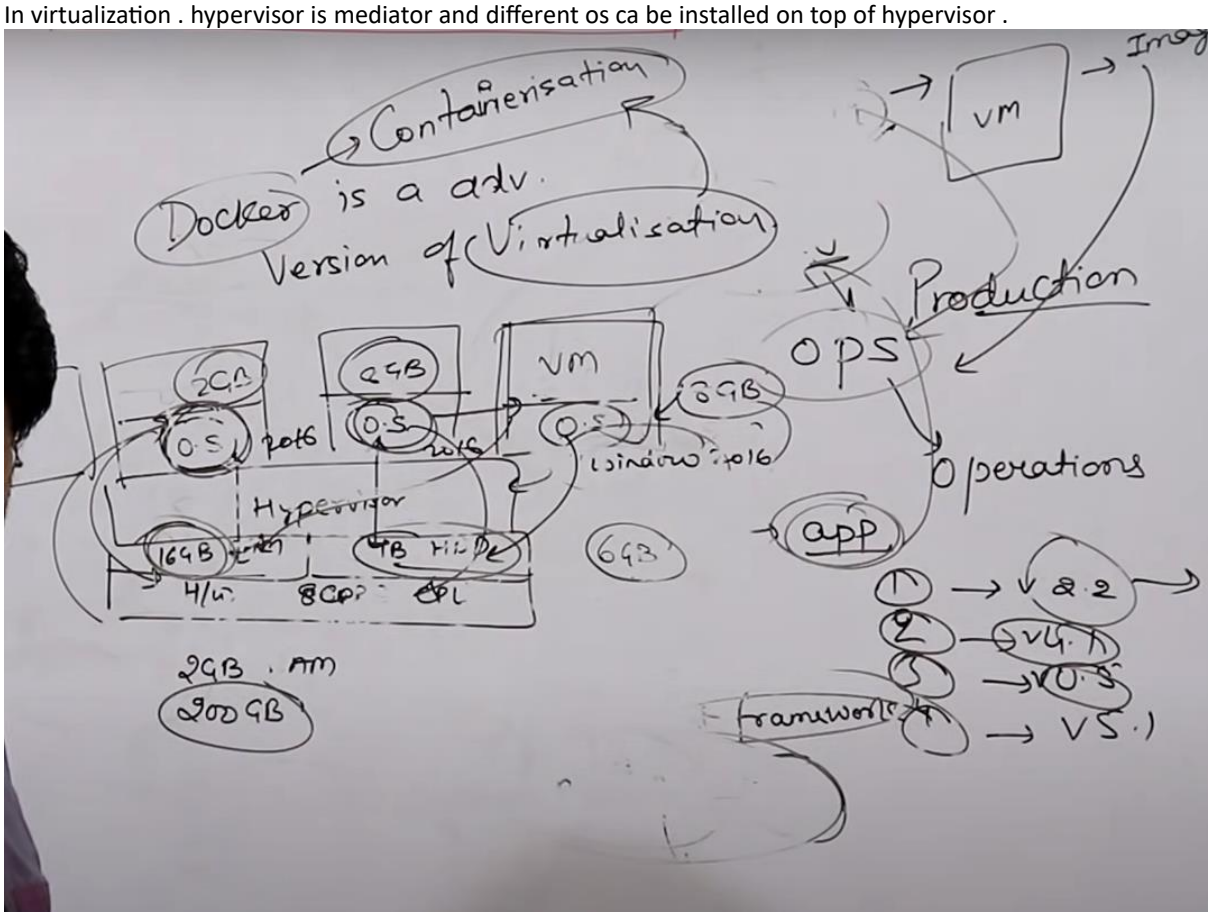
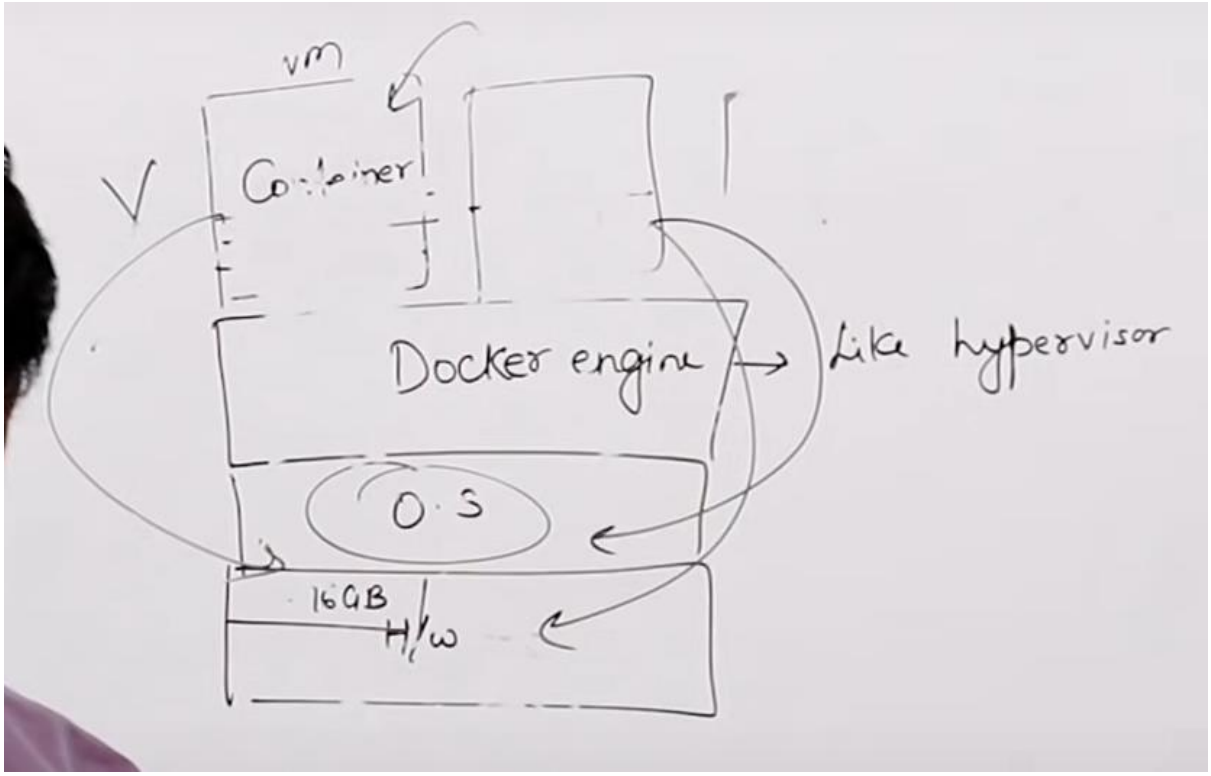
Container is like a virtual machine , Docker is a tool which creates this vm i.e. container . Docker is known as docker engine . We will create image and share it between teams to use . Docker is advance version of virtualization . Docker does containerization .

In virtualization . hypervisor is mediator and different os ca be installed on top of hypervisor .



Hypervisor use storage of host machine .
Storage will be shred between all os in virtualization.

For docker we will install docker engine on host os . Docker engine will create containers . container don’t have any self os . Container uses host system os .



Lec-24 → What is Docker Architecture & Container

→ Docker is an open-source Centralised Platform designed to Create, deploy and run applications.

→ Docker uses Container on the host O.S to run applications. It allows applications to use the same linux Kernel as a System on the host Computer, rather than Creating a Whole Virtual O.S.

→ We Can install docker on any O.S but Docker engine runs natively on Linux distribution.

→ Docker written in 'go' language.

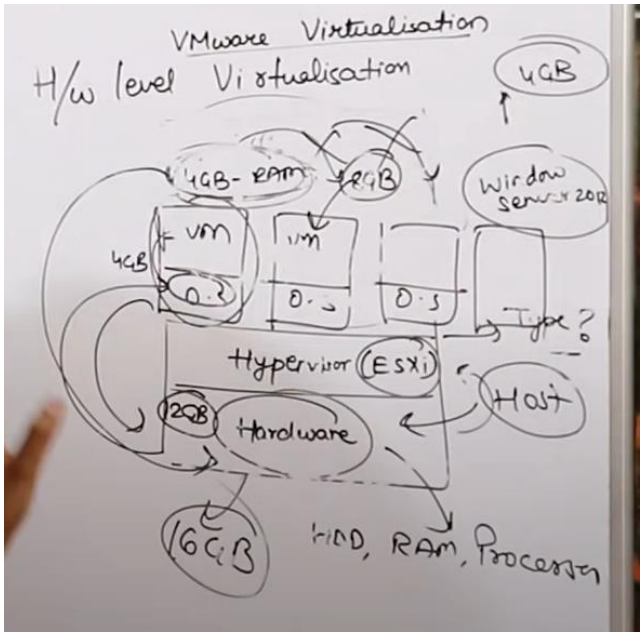
→ Docker is a tool that performs O.S Level Virtualization, also Known as Containerization.

→ Before Docker, many users faces the problem that a particular Code is running in the developer's System but not in the User's System.

→ Docker was first Release in March 2013. It is developed by Solomon Hykes and Sebastian Pahl.

→ Docker is a Set of Platform as a Service that uses O.S Level Virtualization. Whereas VMware uses Hardware level Virtualisation.

Try to remember and if you remember then follow



Container have very light weight os .

Advantages of Docker

→ No pre-allocation of RAM

→ CI Efficiency → Docker enables you to build a Container image and use that Same image across every Step of the deployment process.

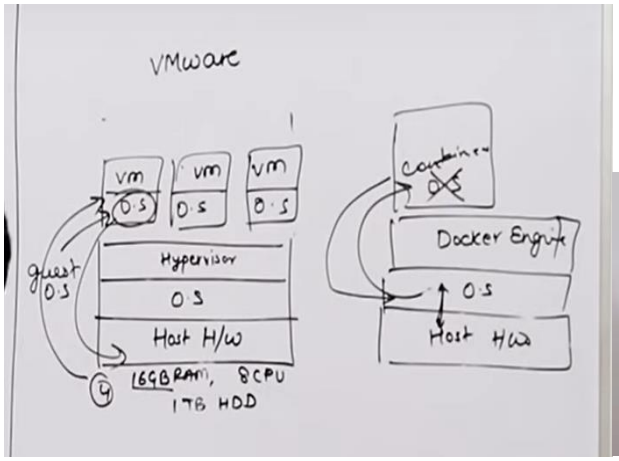
→ Less Cost

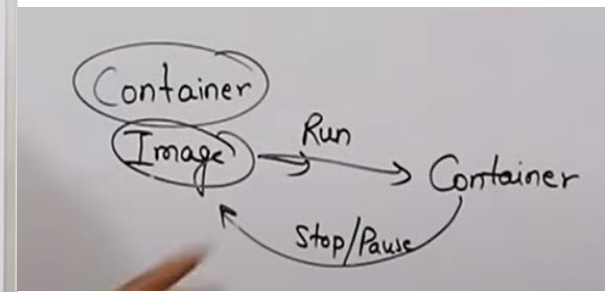
→ It is light in Weight

→ It Can run on physical H/W / Virtual H/W or on Cloud.

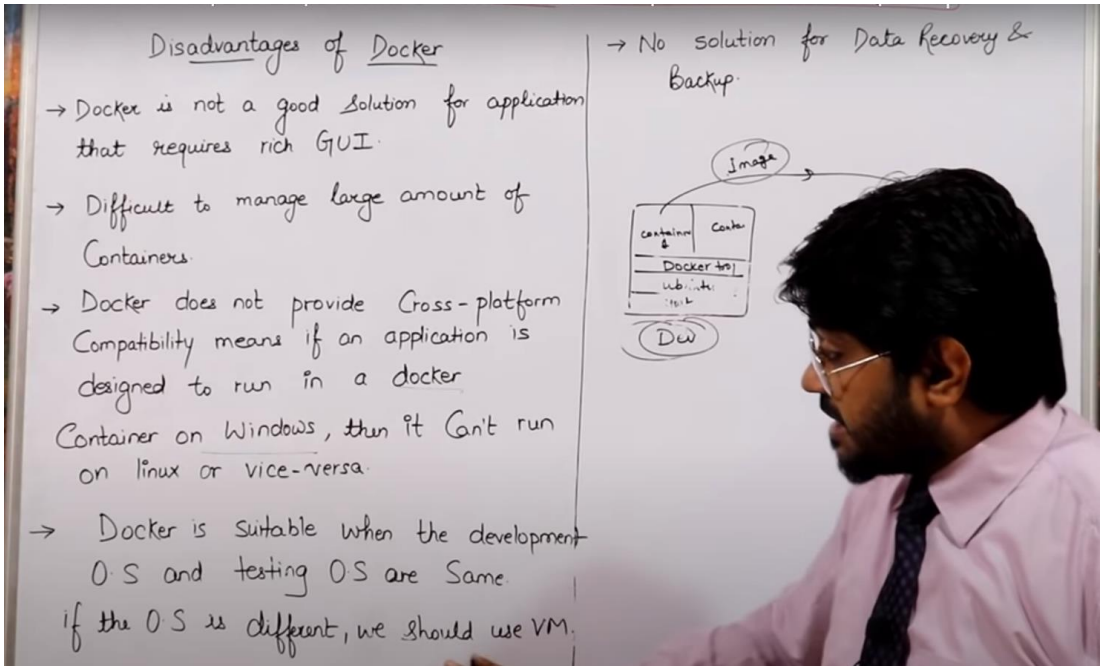
→ You Can re-use the image.

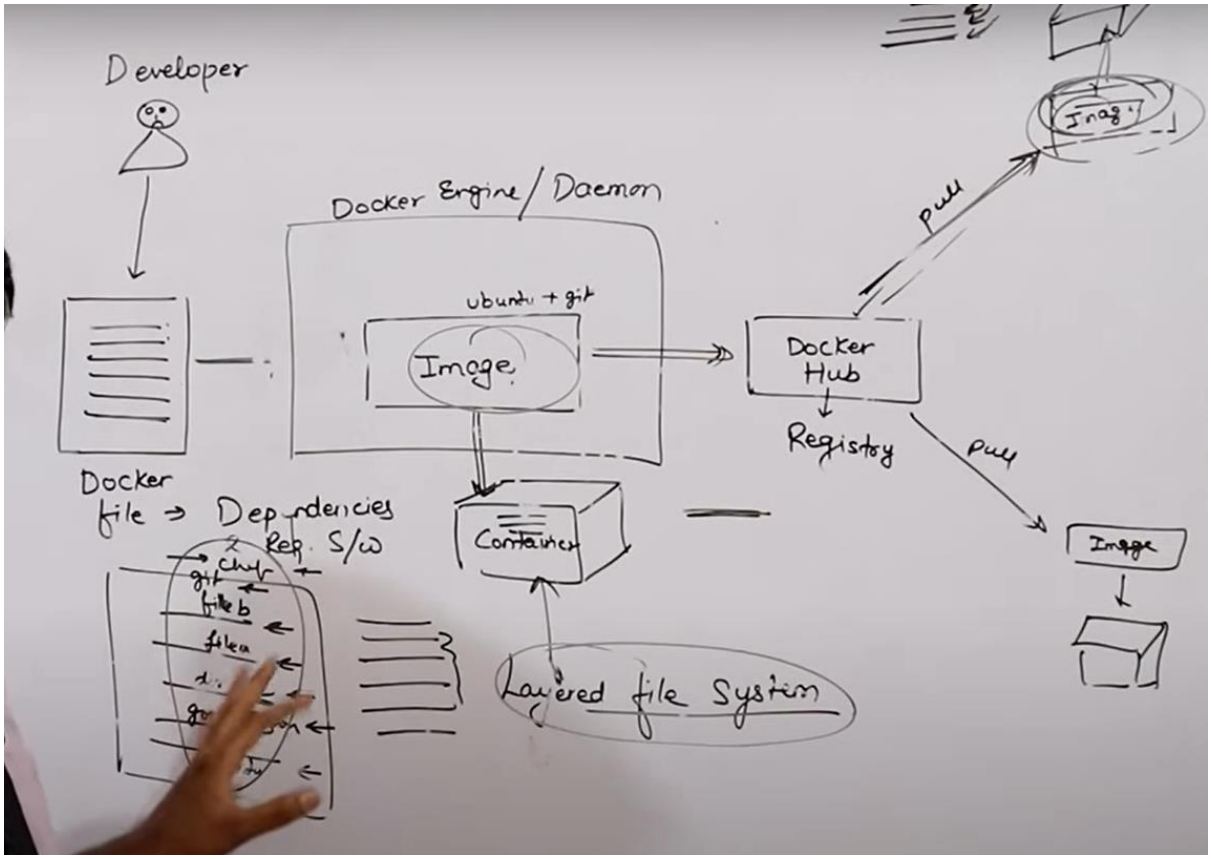
→ It took Very less time to Create Container

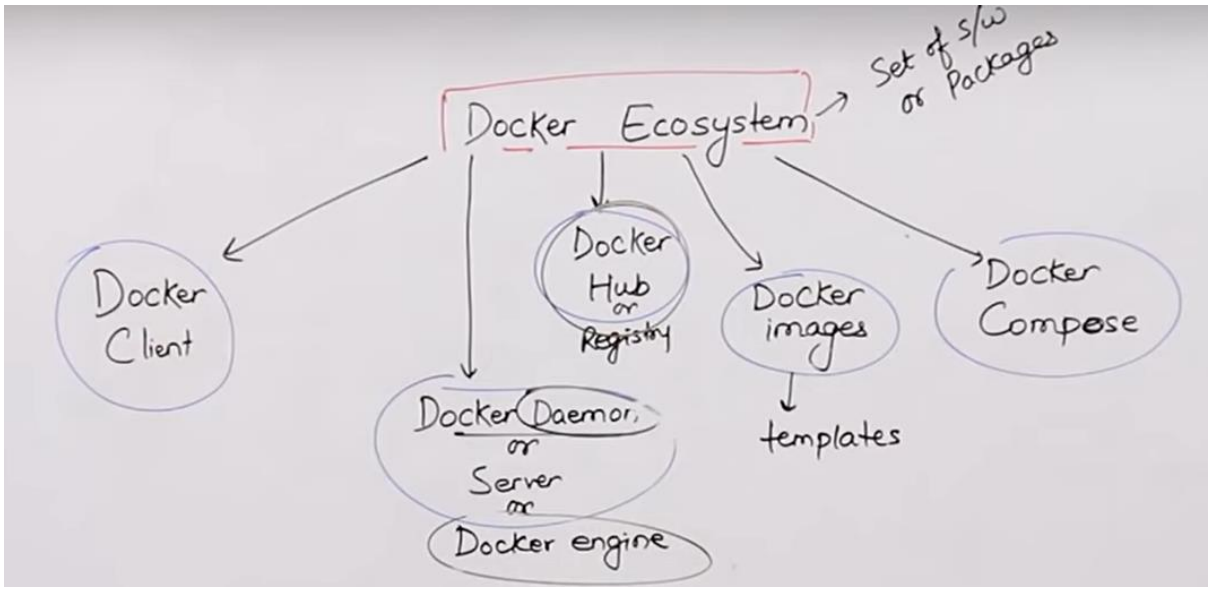




We can make changes in container but not in image .




Developer
Docker Engine / Daemon
ubuntu + git
Image
Docker Hub
Registry
pull
pull
Image
Docker file → Dependencies
Rep. S/w
Container
Layered file System
Image


Docker Ecosystem
Set of s/w or Packages
Docker Client
Docker Hub or Registry
Docker images
templates
Docker Compose
Docker Daemon or Server or Docker engine

Components of Docker

Docker Daemon

→ Docker daemon runs on the Host O.S.
→ It is responsible for running Containers to manages docker services.

→ Docker Daemon Can Communicate with Other daemons.

Docker Client

Docker users Can interact with docker daemon through a Client.

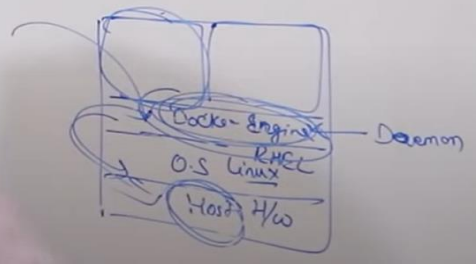
→ Docker Client uses Commands and Rest API to Communicate with the docker daemon

→ when a Client runs any Server Command on the docker Client terminal, the Client terminal Sends these docker Commands to the docker daemon.

→ It is possible for docker Client to Communicate with more than one daemon.

Docker Host

Docker Host is used to provide an environment to execute and run applications. It Contains the docker daemon, images, Containers, networks and storages.



Docker Hub/Registry

Docker registry manages and stores the docker images.
private

There are two types of registries in the docker

① Public Registry → Public registry is also Called as docker hub.

② Private Registry → It is used to share images within the enterprise.

Docker images

→ Docker images are the read only binary templates used to Create docker Containers.

or

Single file with all dependencies and Configuration required to run a program.
Containers

Ways to Create an Images

① Take image from docker hub
② Create image from docker file
③ Create image from existing docker Containers.

Docker Container

→ Container hold the entire packages that is needed to run the application.

or

In other words, we Can Say that, the image is a template and the Container is a Copy of that template.

→ Container is like a Virtual Machine.

→ Images becomes Container when they run on docker engine.

Lec. 26 → Basic Commands in Docker

To see all images present in your local machine

```
J# docker images
```

To find out images in docker hub

```
J# docker search jenkins
```

To download image from dockerhub to local machine

```
J# docker pull jenkins
```

To give name to Container

→ docker run -it --name bhupinder ubuntu /bin/bash

interactive mode

terminal

To Check, Service is start or not

→ Service docker status

To Start Container

→ docker start bhupinder

To go inside container

→ docker attach bhupinder

To See all Containers

→ docker ps -a

To See only running Containers

→ docker ps ← Process status

To Stop Container

→ docker stop bhupinder

To delete Container

→ docker rm bhupinder

Try to remember and if you remember then follow

```
root@host01:~$ docker run -it ubuntu /bin/bash
root@f61d7c304c4a:/# ls
bin  boot  dev  etc  home  lib  lib32  lib64  libx32  media  mnt  opt  proc  root
root@f61d7c304c4a:/# cat /etc/os-release
PRETTY_NAME="Ubuntu 22.04.2 LTS"
NAME="Ubuntu"
VERSION_ID="22.04"
VERSION="22.04.2 LTS (Jammy Jellyfish)"
VERSION_CODENAME=jammy
ID=ubuntu
ID_LIKE=debian
HOME_URL="https://www.ubuntu.com/"
SUPPORT_URL="https://help.ubuntu.com/"
BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
UBUNTU_CODENAME=jammy
root@f61d7c304c4a:/# exit
exit
root@host01:~$
```

```
root@host01:~$ docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
root@host01:~$ docker ps -a
CONTAINER ID   IMAGE     COMMAND                  CREATED              STATUS                          PORTS     NAMES
f61d7c304c4a   ubuntu    "/bin/bash"              About a minute ago   Exited (0) About a minute ago             zealous_bassi
73edff410a48   ubuntu    "/bin/bash --name sa…"   2 minutes ago        Exited (2) 2 minutes ago                  fervent_sinoussi
33e94efe2414   ubuntu    "/bin/bash --name=sa…"   2 minutes ago        Exited (2) 2 minutes ago                  recursing_cartwright
root@host01:~$ docker images
REPOSITORY   TAG       IMAGE ID       CREATED         SIZE
ubuntu       latest    08d22c0ceb15   6 weeks ago     77.8MB
centos       latest    5d0da3dc9764   19 months ago   231MB
root@host01:~$
```

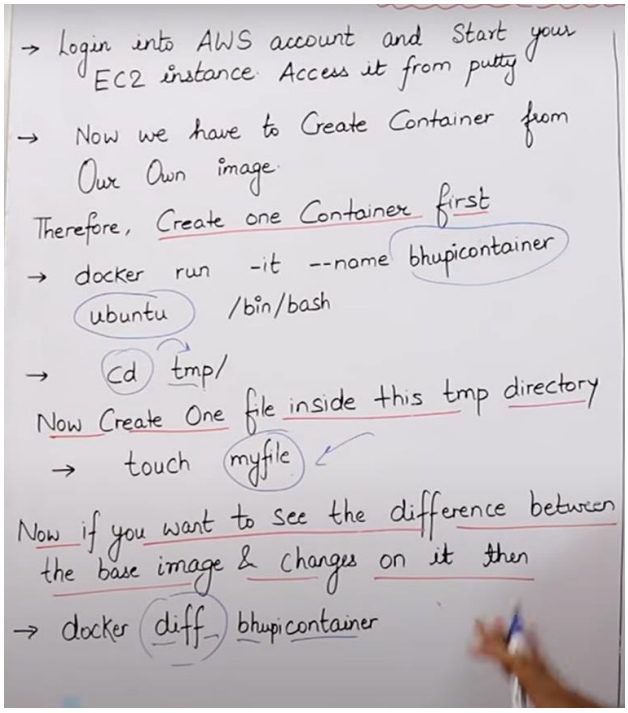
```
root@host01:~$ docker run -it centos /bin/bash
[root@d7877349f76b /]# cat /etc/os-release
NAME="CentOS Linux"
VERSION="8"
ID="centos"
ID_LIKE="rhel fedora"
VERSION_ID="8"
PLATFORM_ID="platform:el8"
PRETTY_NAME="CentOS Linux 8"
ANSI_COLOR="0;31"
CPE_NAME="cpe:/o:centos:centos:8"
HOME_URL="https://centos.org/"
BUG_REPORT_URL="https://bugs.centos.org/"
CENTOS_MANTISBT_PROJECT="CentOS-8"
CENTOS_MANTISBT_PROJECT_VERSION="8"
[root@d7877349f76b /]#
```

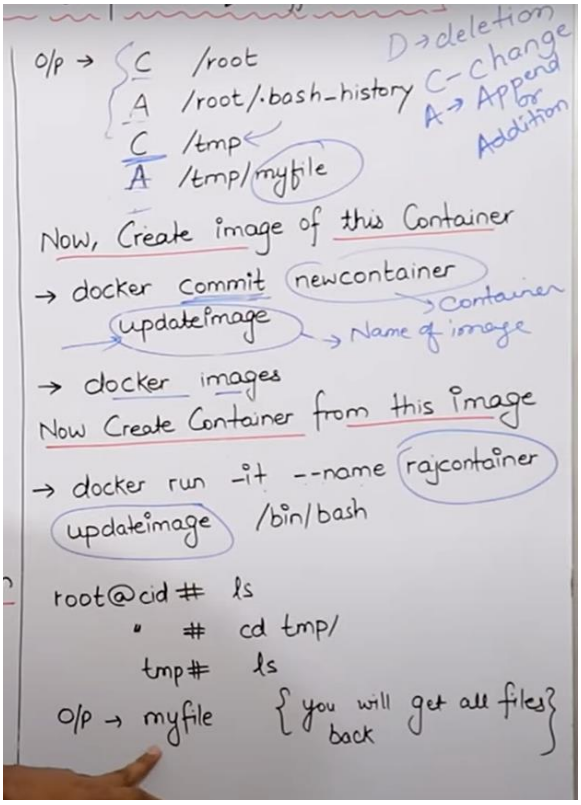
```
root@host01:~$ docker ps -a
CONTAINER ID   IMAGE     COMMAND                  CREATED          STATUS                      PORTS     NAMES
d7877349f76b   centos    "/bin/bash"              33 seconds ago   Exited (0) 3 seconds ago            musing_albattani
4314e73ce6da   centos    "/bon/bash"              39 seconds ago   Created                             determined_wiles
f61d7c304c4a   ubuntu    "/bin/bash"              3 minutes ago    Exited (0) 3 minutes ago            zealous_bassi
73edff410a48   ubuntu    "/bin/bash --name sa…"   4 minutes ago    Exited (2) 4 minutes ago            fervent_sinoussi
33e94efe2414   ubuntu    "/bin/bash --name=sa…"   4 minutes ago    Exited (2) 4 minutes ago            recursing_cartwright
root@host01:~$
```

```
root@host01:~$ docker search jenkins
NAME                          DESCRIPTION                                     STARS     OFFICIAL   AUTOMATED
jenkins                       DEPRECATED; use "jenkins/jenkins:lts" instead   5617      [OK]
jenkins/jenkins               The leading open source automation server       3473
jenkins/jnlp-slave            a Jenkins agent which can connect to Jenkins…   156                  [OK]
jenkins/inbound-agent                                                         96
jenkins/ssh-agent             Docker image for Jenkins agents connected ov…   36
jenkins/slave                 base image for a Jenkins Agent, which includ…   49                   [OK]
jenkins/jnlp-agent-maven      A JNLP-based agent with Maven 3 built in        8
bitnami/jenkins               Bitnami Docker Image for Jenkins                61                   [OK]
jenkins/ssh-slave             A Jenkins slave using SSH to establish conne…   39                   [OK]
jenkins/agent                                                                 52
jenkins/jnlp-agent-ruby                                                       1
jenkins/jnlp-agent-docker                                                     8
jenkins/jnlp-agent-node                                                       1
jenkins/jnlp-agent-python     A JNLP-based agent with Python built in         3
```

```
root@host01:~$ docker pull jenkins/jenkins
Using default tag: latest
latest: Pulling from jenkins/jenkins
b0248cf3e63c: Pull complete
33b6b181dd75: Pull complete
a0ae619c7e04: Pull complete
628a3df031c5: Pull complete
0c9cd1e5e117: Pull complete
4a90cdec99a2: Pull complete
a8bb08b626f0: Pull complete
0564e42b8d2d: Pull complete
fd62b7502c2d: Pull complete
f1461e578f49: Pull complete
9cde15aa1b91: Pull complete
f055553ffae1: Pull complete
bcf1ce5799b0: Pull complete
Digest: sha256:7560cc798140cdcdef5b75ca069c28b2a
```

docker run -it –name sapna ubuntu /bin/bash






```
root@host01:~$ service docker status
● docker.service - Docker Application Container Engine
     Loaded: loaded (/lib/systemd/system/docker.service; enabled; vendor preset: enabled)
     Active: active (running) since Fri 2023-04-21 12:10:56 UTC; 38s ago
TriggeredBy: ● docker.socket
       Docs: https://docs.docker.com
   Main PID: 839 (dockerd)
      Tasks: 8
     Memory: 95.2M
     CGroup: /system.slice/docker.service
             └─839 /usr/bin/dockerd

Apr 21 12:10:56 host01 dockerd[839]: time="2023-04-21T12:10:56.087178715Z" level=debug msg=
```

```
root@host01:~$ docker run --name sapna -it ubuntu /bin/bash
Unable to find image 'ubuntu:latest' locally
latest: Pulling from library/ubuntu
2ab09b027e7f: Pull complete
Digest: sha256:67211c14fa74f070d27cc59d69a7fa9aeff8e28ea118ef3babc295a0428a6d21
Status: Downloaded newer image for ubuntu:latest
root@111ab2a6bd52:/# ls
bin  boot  dev  etc  home  lib  lib32  lib64  libx32  media  mnt  opt  proc  roo
root@111ab2a6bd52:/# cd tmp/
root@111ab2a6bd52:/tmp# ls
root@111ab2a6bd52:/tmp# touch newfile
root@111ab2a6bd52:/tmp# ls
newfile
root@111ab2a6bd52:/tmp# exit
exit
root@host01:~$
```

```
exit
root@host01:~$ docker diff sapna
C /root
A /root/.bash_history
C /tmp
A /tmp/newfile
root@host01:~$
```

```
A /tmp/newfile
root@host01:~$ docker commit sapna updatedimage
sha256:9c6a37efde7b2608f781a9bc853c5050503ef8bacc45e8ebbb15cd8d23bb74b9
root@host01:~$ docker images
REPOSITORY      TAG       IMAGE ID       CREATED          SIZE
updatedimage    latest    9c6a37efde7b   29 seconds ago   77.8MB
ubuntu          latest    08d22c0ceb15   6 weeks ago      77.8MB
root@host01:~$ docker run -it --name newc undatedimage /bin/bash
Unable to find image 'undatedimage:latest' locally
docker: Error response from daemon: pull access denied for undatedimage,
is denied.
See 'docker run --help'.
root@host01:~$ docker run -it --name new updatedimage:latest /bin/bash
root@16b14e1cbdef:/# ls
bin  boot  dev  etc  home  lib  lib32  lib64  libx32  media  mnt  opt  pr
root@16b14e1cbdef:/# cd tmp/
root@16b14e1cbdef:/tmp# ls
newfile
root@16b14e1cbdef:/tmp#
```

Dockerfile

→ Dockerfile is basically a text file. It Contains Some set of instruction.

→ Automation of Docker image Creation

Docker Components

FROM → for base image. This Command must be on top of the dockerfile.

RUN → To execute Commands, it will Create a layer in image.

MAINTAINER → Author/ Owner/ Description

COPY → Copy files from local system (docker VM) We need to provide Source, destination (We Can't download file from internet and any remote repo)

ADD → Similar to COPY but, it provides a feature to download files from internet, also we extract file at docker image side.

EXPOSE → To expose ports such as port 8080 for tomcat, port 80 for nginx etc.

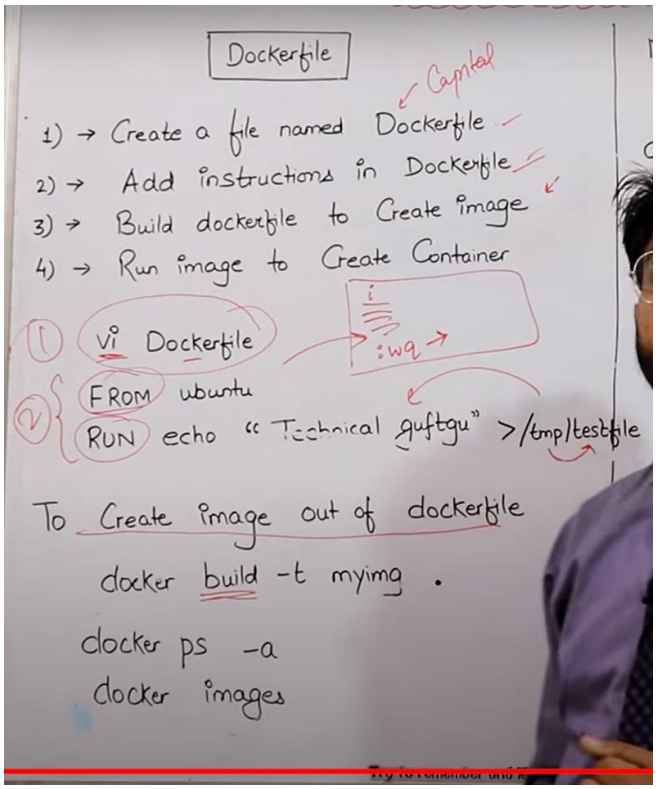
WORKDIR → To set working directory for a Container.

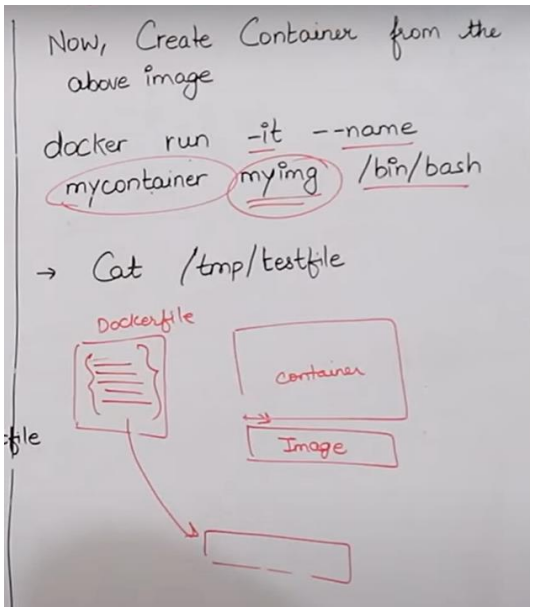
CMD → Execute Commands but during Container Creation.

ENTRYPOINT → Similar to CMD, but has higher priority over CMD, first commands will be executed by ENTRYPOINT only.

ENV → Environment Variables

ARG : ARG is only available during the build of a Docker image (RUN etc.), not after the image is created and containers are started from it (ENTRYPOINT, CMD). Basically ARG command inside a Docker file is to define the name of a parameter and its default value.






```
root@host01:~$ service docker status
● docker.service - Docker Application Container Engine
     Loaded: loaded (/lib/systemd/system/docker.service; enabled; vendor preset: enabled)
     Active: active (running) since Fri 2023-04-21 13:16:24 UTC; 34s ago
TriggeredBy: ● docker.socket
       Docs: https://docs.docker.com
   Main PID: 822 (dockerd)
      Tasks: 8
     Memory: 96.9M
     CGroup: /system.slice/docker.service
             └─822 /usr/bin/dockerd

Apr 21 13:16:24 host01 dockerd[822]: time="2023-04-21T13:16:24.390054958Z" level=debug msg
```

Dockerfile ×

Dockerfile

```
1  FROM ubuntu
2  RUN echo "Test Doekcfile" > /tmp/test
```

```
root@host01:~/workspace$ docker build -t test .
[+] Building 155.8s (6/6) FINISHED
 => [internal] load .dockerignore
 => => transferring context: 2B
 => [internal] load build definition from Dockerfile
 => => transferring dockerfile: 86B
 => [internal] load metadata for docker.io/library/ubuntu:latest
 => [1/2] FROM docker.io/library/ubuntu@sha256:67211c14fa74f070d27cc59
 => => resolve docker.io/library/ubuntu@sha256:67211c14fa74f070d27cc59
 => => sha256:67211c14fa74f070d27cc59d69a7fa9aeff8e28ea118ef3babc295a0
 => => sha256:7a57c69fe1e9d5b97c5fe649849e79f2cfc3bf11d10bbd5218b4eb61
 => => sha256:08d22c0ceb150ddeb2237c5fa3129c0183f3cc6f5eeb2e7aa4016da3
 => => sha256:2ab09b027e7f3a0c2e8bb1944ac46de38cebab7145f0bd6effebfe54
 => => extracting sha256:2ab09b027e7f3a0c2e8bb1944ac46de38cebab7145f0b
 => [2/2] RUN echo "Test Doekcfile" > /tmp/test
 => exporting to image
 => => exporting layers
 => => writing image sha256:eee0026690a97c57f2798df682d77ededdf29096dc
 => => naming to docker.io/library/test
root@host01:~/workspace$ docker images
REPOSITORY   TAG       IMAGE ID       CREATED          SIZE
test         latest    eee0026690a9   32 seconds ago   77.8MB
root@host01:~/workspace$ docker run -it --name sapna test:latest /bin/bash
root@08ddbdfa92bf:/# ls
bin  boot  dev  etc  home  lib  lib32  lib64  libx32  media  mnt  opt  proc  r
root@08ddbdfa92bf:/# cd tmp/
root@08ddbdfa92bf:/tmp# ls
test
root@08ddbdfa92bf:/tmp# cd test
bash: cd: test: Not a directory
root@08ddbdfa92bf:/tmp# ;s
bash: syntax error near unexpected token `;'
root@08ddbdfa92bf:/tmp# cat test
Test Doekcfile
root@08ddbdfa92bf:/tmp#
```

Dockerfile ×

Dockerfile

```
1  FROM ubuntu
2  WORKDIR /tmp
3  RUN echo "Test Dockerfile" > /tmp/test
4  ENV myname sapna
5  COPY testfile1 /tmp
6  ADD test.tar.gz /tmp
```

```
root@host01:~/workspace$ touch testfile1
root@host01:~/workspace$ ls
Dockerfile  testfile1
root@host01:~/workspace$ touch test
root@host01:~/workspace$ tar -cvf test.tar test
test
root@host01:~/workspace$ ls
Dockerfile  test  testfile1  test.tar
root@host01:~/workspace$ gzip test.tar
root@host01:~/workspace$ rm -rf test
root@host01:~/workspace$ ls
Dockerfile  testfile1  test.tar.gz
root@host01:~/workspace$
```

```
Dockerfile  testfile1  test.tar.gz
root@host01:~/workspace$ docker build -t newimage .
[+] Building 31.7s (10/10) FINISHED
 => [internal] load build definition from Dockerfile
 => => transferring dockerfile: 158B
 => [internal] load .dockerignore
 => => transferring context: 2B
 => [internal] load metadata for docker.io/library/ubuntu:lates
 => CACHED [1/5] FROM docker.io/library/ubuntu@sha256:67211c14
 => [internal] load build context
 => => transferring context: 183B
 => [2/5] WORKDIR /tmp
 => [3/5] RUN echo "Test Dockerfile" > /tmp/test
 => [4/5] COPY testfile1 /tmp
 => [5/5] ADD test.tar.gz /tmp
 => exporting to image
 => => exporting layers
 => => writing image sha256:b1bb5dd2d048b9a72a31dd62a260945404
 => => naming to docker.io/library/newimage
```

```
root@host01:~/workspace$ docker rm 1d7c2
1d7c2
root@host01:~/workspace$ docker rmi newimage:latest
Untagged: newimage:latest
Deleted: sha256:b1bb5dd2d048b9a72a31dd62a26094540472c8cb9778b6c20b9ce2d48eab8f47
root@host01:~/workspace$ docker build -t newimage .
[+] Building 5.1s (2/3)
 => [internal] load .dockerignore
 => => transferring context: 2B
 => [internal] load build definition from Dockerfile
 => => transferring dockerfile: 162B
 => [internal] load metadata for docker.io/library/ubuntu:latest
 => => naming to docker.io/library/newimage
root@host01:~/workspace$ docker run -it --name sapna1 newimage:latest /bin/bash
root@bc21da496365:/tmp# ls
test  testfile  testfile1
root@bc21da496365:/tmp# cat test
root@bc21da496365:/tmp# cat testfile
Test Dockerfile
root@bc21da496365:/tmp#
```