DOCKER

Virtualization

===================

This is the process of running multiple OS's parallelly on

a single pice of h/w.

Here we have h/w(bare metal) on top of which we have host os

and on the host os we install an application called as hypervisor

On the hypervisor we can run any no of OS's as guest OS

The disadvantage of this approach is these application running on the

guest OS have to pass through n number of lavers to access the H/W

resources.

Containarization

======================

Here we have bare metal on top of whcih we install the host Os

and on the hsot OS we install an application called as Docker Engine

On the docker engine we can run any application in the form of containers

Docker is a technology for creating thse containers

Docker achive what is commonly called as "process isolation"

ie all the applications(processes) have some dependency on a specific

OS.This dependency is removed by docker and we can run them on any

OS as containers if we have Docker engine installed

These containers pass through less no of layers to access the h/w resources

also organizations need not spend money on purchasing licenses of different

OS's to maintian various applications

Docker can be used at the the stages of S/W development life cycle

Build---->Ship--->Run

===========================================================================

Day 2-3

===========================================================================

Setup of Docker on Windows

==============================

1 Download docker desktop from

https://www.docker.com/products/docker-desktop

2 Install it

3 Once docker is installed we can use Power shell

to run the docker commands

===============================================================================

Create an ubuntu linux machine using vagrant

==================================================

1 Download oracle virtual box from

https://www.virtualbox.org/wiki/Downloads

2 Install it

3 Download and install vagrant

https://www.vagrantup.com/downloads

4 Download the vagrant file and copy it into an empty folder

5 Open cmd prompt

6 Change directory to the folder where the vagrantfile is copied

cd path\_of\_folder

7 vagrant up

8 USername and password is:vagrant

========================================================================

Using AWS

================

1 Login in AWS account

2 Create ane new Ubuntu 20 instance

3 To connect to this ubuntu instance use gitbash

https://git-scm.com/downloads

=====================================================================\

Installing docker on Linux

================================

1 Open get.docker.com

2 Copy and paste the below 2 commands

curl -fsSL https://get.docker.com -o get-docker.sh

sh get-docker.sh

=================================================================

Images and Containers

===========================

A Docker image is a combination of bin/libs that are necessary

for a s/w application to work.Initially all the s/w's of docker

are available in the form of docker images

A running instance of an image is called as a container

==========================================================

Docker Host: The server where docker is installed is called

docker host

Docker client: This is CLI of docker which accepts the docker commands

from the users and passes to a background process called docker deamon

Docker deamon: This accepts the commands comming from docker client

and routes them to work on docker images or contaienr or the registry

Docker registry: This is the location where docker images are stored

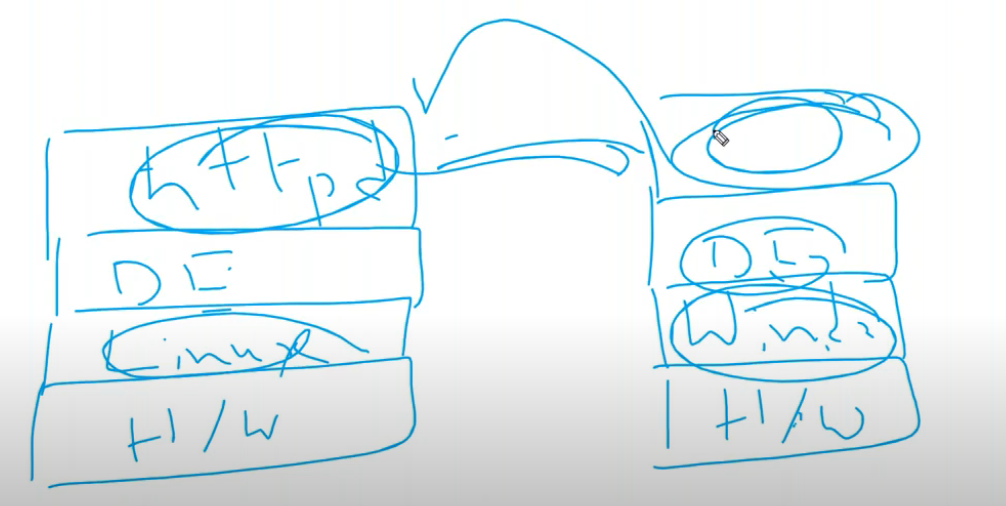
This if of 2 type

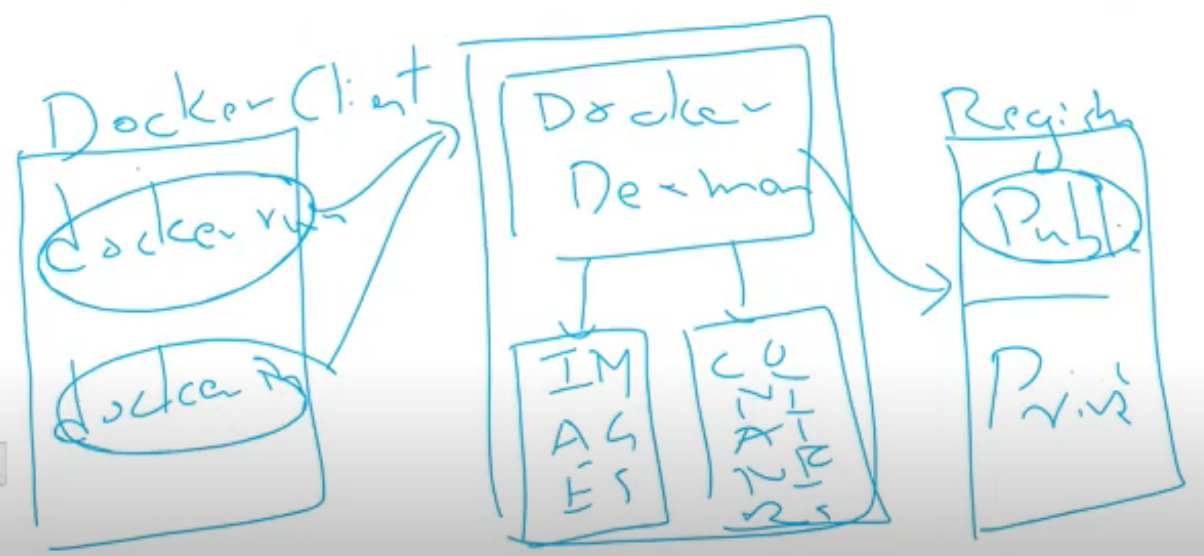
1 Public (hub.docker.com)

2 Private: This is set up on one of our internal servers

========================================================

Images that are designed to work on Linux will work on Windows BUT images that are designed to work on Windows itself will not work on Linux





Day 4

============================================================================

Important docker commands

===============================

Working on docker images

============================

1 To download a docker image

docker pull image\_name

2 To upload a docker image to docker registry

docker push image\_name

3 To search for a docker image on the registry

docker search image\_name

4 To see the list of images downloaded

docker image ls

(or)

docker images

5 To delete a docker image

docker rmi image\_name

6 To delete all docker images

docker system prune -af

7 To create a dcoker image from a customsied container

docker commit container\_name/container\_id image\_name

8 To create an image from a docker file

docker build -t image\_name .

9 To get history for a docker image

docker image history image\_name

10 To get detailed info about a docker image

docker image inspect image\_name

Working on docker containers

=================================

11 To see the list of running containers

docker container ls

12 To see the list of all containers (running and stopped)

docker ps -a

13 To stop a running container

docker stop container\_name/container\_id

14 To start a stopped container

docker start container\_name/container\_id

15 To restart a container

docker restart container\_name/container\_id

To restart after 20 seconds

docker restart -t 20 container\_name/container\_id

16 To delete a stopped container

docker rm container\_name/container\_id

17 To delete a running container

docker rm -f container\_name/container\_id

18 To stop all running containers

docker stop $(docker ps -aq)

19 To delete all stopped containers

docker rm $(docker ps -aq)

20 To delete all containers (running and stopped)

docker rm -f $(docker ps -aq)

21 To get detailed info about a container

docker inspect container\_name/container\_id

22 To see the logs used by a container

docker logs container\_name/container\_id

23 To create a container

docker run image\_name

Run command options

-------------------

--name: Used to assign a name for the container

-it : Used to open interactive terminal in the container

-d : Used to run the containers in detached modes as a background process

-e : Used to pass environment variables to containers

-v : Used to attach a external fodler or device as a volume

--volumes-from : Used for sharing volumes between containers

-rm : USed to delete a container on exit

-p : Used for port mapping it will link the container port with

the host port

Eg: -p 8080:80 here 80 is the container port also called as internal

port and 8080 is host port also called as external port

-P: Used for automatic port mapping ie it will link the container port

with a host port that is greater than 30000

--link : This is used to create alink between multiple containers

to setup the microservices architecture.

--network: Used for creating a network of containers

-c: Used to specify the amount of maximum cpu that can be assigned

to a containers

-m : Used to sepcify the maximim amount of memory that a contaienr

can use