**Docker**

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

Day 1

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

Virtualization

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

Here we have baremetal(H/W) on top of which we install the hostOS

and on the hostOS we install an application called hypervisor

On the hypervisor we can install different OS's as guest OS's

and on the guest OS we can install which ever application we want

These application have to pass through so many layers to access the

h/w resources

Containarization

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

Here we have a baremetal on top of which we have host OS and

on the host OS we install an application called Docker Engine

On the Docker engine we can run any application that we want

directly/

Docker is used for performing "process isolation" ie it remove the

dependency that an application has on an underlying OS and it allows

it to be run directly on the Docker engine

Due to this reason the licensing of the OS's need not be purchased/

Also these containers are extremly light weight and they consume

very less amount of H/W resources

Creating any application can be done in a matter of seconds

Docker can be used at all the stages of S/w development,testing and Delivery

BUILD---->SHIP----->RUN

Installing docker on Windows

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

1 Open https://docs.docker.com/docker-for-windows/install/

2 Download docker for windows--->Install it

3 To execute docker commands

Open Power shell and run the docker commands

Note: Docker can be installed only on Windows 10 Prof 64 bit version

and Windows 2016 Server edition

Note: Once docker is installed on Windows it activates an application

called Windows Hypervisor and once this is activated it will not

allow us to run any other virtualization s/w

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

Day 2

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

Installing Docker on Linux

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

1 Create an ubuntu 22 AWS instance

Method 1:

2 Open http://get.docker.com

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

sh get-docker.sh

or

Method : 2

sudo apt-get update -y

sudo apt-get install docker.io -y

Note: The first command downloads the shell script which can

install docker and the second command executes the shell script

Images and Containers

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

A docker image is a combination of the bin/libs that are necessary for

a s/we application to run.All the docker images are available at

http://hub.docker.com

A running instance of an image is called as a container.Any number of

containers can be created from one docker image

Docker Host: The server where docker is docker is isntalled and where

we execute the docker commands is called as docker host

Docker client: This is a backgorund process which is responsible for

accepting the docker commands and it will pass those commands to

a the docker deamon

Docker deamon is another background process which accepts the commands from

the docker client and route them to work on images,containers or registry

Docker Registry: This is the location where docker images can be preserved

This is of 2 type public and private

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

Important Docker Commands

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

Working on docker images

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

1 To download a docker image

docker pull image\_name

2 To upload a docker image

docker push image\_name

3 To see the list of docker images on the docker host

docker image ls

(or)

docker images

4 To search for a docker image

docker search image\_name

5 To delete a docker image

docker rmi image\_name/image\_id

6 To get detailed info about an image

docker image inspect image\_name/image\_id

7 To create a docker image from a container

docker commit container\_name/container\_id new\_image\_name

8 To create an image from a docker file

docker build -t new\_image\_name .

9 To save an image as a tar file

docker image save image\_name

10 To get an image back from a tar file

docker image load tarfile\_name

11 To delete all the image

docker image prune

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

Working on Docker container

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

12 To see the list of running docker containers

docker container ls

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

docker ps -a

14 To start a stopped container

docker start container\_name/container\_id

15 To stop a container

docker stop container\_name/container\_id

16 To restart a container

docker restart container\_name/container\_id

To restart after 20 seconds

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

17 To delete a stopped container

docker rm container\_name/container\_id

18 To delete a running container

docker rm -f container\_name/container\_id

19 To stop all container

docker stop $(docker ps -aq)

20 To delete all stopped containers

docker rm $(docker ps -aq)

21 To delete all container(running and stopped)

docker rm -f $(docker ps -aq)

22 To see the logsw generated by a container

docker logs container\_name/container\_id

23 To see the port opened by a container

docker port container\_name/container\_id

24 To get detailed info about a container

docker inspect container\_name/container\_id

25 To execute a command or a process in a container from outside

docker exec -it container\_name/container\_id command

Eg: To run the bash shell in a container

docker exec -it container\_name/container\_id bash

26 To come out of a container without exit so that the container run in the background

ctrl+p,ctrl+q

27 To go back into such a container

docker attach container\_name/container\_id

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

Day 3

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

28 To create a new docker container

docker run image\_name

Run commnads options

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

--name: This is used to give a name to a container

-d : Used to run the container in detached mode as a background process

-e : Used to pass environment variables to a container

--restart: This will attempt to restart the container if it stops

-rm: Used to delete a container on exit

-v : Used to attach volumes to a container

--volumes-from: USed for sharing volumes between multiple containers

-p : USed for port mapping it will link the container port(internal port)

with the host port(external port)

Eg: -p 8080:80 Here 8080 is host port and 80 is container port

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

mapped with a host port that will be greater than 30000

--link: Used to link multilple containers to create a mult container

architecture

-it : Used to open interactive terminal in the container

-m : Used to specify an upper limit on the amout of memory that a container

can use

-c : Used to specify the percentage of CPU that the container can use

--network: USed to start a container on a specific network

Working on docker networks

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

29 To see the list of all docker networks

docker network ls

30 To create a network

docker network create --driver network\_type network\_name

31 To get detailed info about a network

docker network inspect network\_name/network\_id

33 To attach a network to a running contianer

docker network connect network\_name/network\_id container\_name/container\_id

34 To remove a running container from a network

docker network disconnect network\_name/network\_id container\_name/container\_id

35 To delete a network

docker network rm network\_name/network\_id

Working on docker volumes

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

36 To see the list of docker volumes

docker volume ls

37 To create a docker volume

docker volume create volume\_name

38 To delete a docker volume

docker volume rm volume\_name

39 To get detailed info about a volume

docker volume inspect volume\_name

40 To delete all unused images,volumes and networks

docker system prune -a

UseCase

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

Create an nginx container in detached mode and also perform port mapping

docker run --name webserver -d -p 8888:80 nginx

To access the nginx from the browser

public\_ip\_dockerhost:8888

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

UseCase

Create a httpd container and perform automatic port mapping

docker run --name appserver -d -P httpd

To check ports used by the htptd container

docker port appserver

To access the httpd from a browser

public\_ip\_of\_dockerhost:port\_no\_from\_previouscommand

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

UseCase

Create jenkins as a container

docker run --name myjenkins -d -p 8080:8080 jenkins

To access jenkins from a browser

public\_ip\_of\_dockerhost:8080

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

Day 4

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

UseCase 1

Start centos as a container and also open interactive

terminal in it

docker run --name c1 -it centos

To come out of the centos container back to the host machine

exit

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

UseCase 2

Start an ubuntu container and open interactive terminal in it

docker run --name u1 -it ubuntu

To come out of the ubuntu container with out exit

ctrl+p,ctrl+q

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

UseCase 3

Start a postgres database as a container

docker run --name mypostgres -d -e POSTGRES\_PASSWORD=tiru postgres

To check if the postgres container is running

docker container ls

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

UseCase 4

Start mysql:5 as a container and login as root user

Create some sql tables in the database container

1 Create a mysql container

docker run --name mydb -d -e MYSQL\_ROOT\_PASSWORD=tiru mysql:5

2 To check if the mysql container is running

docker container ls

3 To open interactive bash shell in the container

docker exec -it mydb bash

4 To login as root user

mysql -u root -p

Password: tiru

5 To see the list of databases

show databases;

6 To move into any of these databases

use db\_name;

Eg: use sys;

7 To create emp and dept tables here

Open https://justinsomnia.org/2009/04/the-emp-and-dept-tables-for-mysql/

Copy the code for creation of emp and dept tables and paste in the

container

8 To see the data in the tables in the container

select \* from emp;

select \* from dept;

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

Creating Multi container architecture

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

This can be done in 3 ways

1 --link option (This is docker run command option and it is depricated)

2 docker-compose

3 Docker Networking

Using --link option

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

This is outdated way of creating a multi container architecture

and it will be removed in the upcoming versions of docker

UseCase 1

Create 2 busybox container and link them.Check if they are pinging

1 Start a busybox container

docker run --name c1 -it busybox

2 Come out of busybox container without exit

ctrl+p,ctrl+q

3 Create another busybox container and link with the first c1 container

docker run --name c2 -it --link c1:my\_c1 busybox

4 Check if in c2 container we can ping to c1 container

ping c1 (It will ping)

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

Day 5

=========================================================================-

UseCase-1

Create a mysql container and link it with a wordpress container

A development environment required by a wordpress developer

should get created

1 Create a mysql:5 version

docker run --name mydb -d -e MYSQL\_ROOT\_PASSWORD=tiru mysql:5

2 Create a wordpress container and link with the mysql container

docker run --name mywordpress -d -p 5555:80 --link mydb:mysql wordpress

3 Check if the wordpress and mysql containers are running

docker container ls

4 See if the wordpress container is linked with mysql container

docker inspect mywordpress

Search for "Links" section

5 To access the wordpress application from a browser

public\_ip\_of\_dockerhost:5555

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

UseCase-2

Create a dev environment where mysql container is linked with ghost

container

1 Create a mysql container

docker run --name mydb -d -e MYSQL\_ROOT\_PASSWORD=tiru mysql:5

2 Create a ghost container

docker run --name ghost -d -P --link mydb:mysql ghost

3 To see the port used by ghost container

docker port ghost

4 To access the ghost application from browser

public\_ip\_dockerhost:port\_from\_step3

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

UseCase 3

Create CI-CD environment where a jenkins container is linked

with 2 tomcat containers one for qaserver and other for

prodserver

1 Create a jenkins container

docker run --name myjenkins -d -p 5050:8080 jenkins

2 To access jenkins from browser

public\_ip\_of\_dockerhost:5050

3 Start a tomcat container and name it QAserver

docker run --name qaserver -d -p 6060:8080 --link myjenkins:jenkins tomcat

4 Start another tomcat container and name it prodserver

docker run --name prodserver -d -p 7070:8080 --link myjenkins:jenkins tomcat

5 Check if all 3 containers are running

docker container ls

UseCase 4

===========

Create LAMP architecture using docker containers

1 Create a mysql container

docker run --name mydb -d -e MYSQL\_ROOT\_PASSWORD=intellqiit mysql

2 Create an apache(httpd) container and link with mysql container

docker run --name apache -d -p 9090:80 --link mydb:mysql httpd

3 Create a php container and link with mysql and apache containers

docker run --name php -d --link mydb:mysql --link apache:httpd php:7.2-apache

4 Check if all three containers are running

docker container ls

5 To check if php container is linked with mysql and apache containers

docker inspect php

Go to "Links" section