Docker

Docker : Docker is a open source platform for developing, shipping and running the application. Docker also known as advanced OS Virtualization software platform that makes it easier to create, deploy and run application in Docker container.

Container : it is also known as engine or run time environment.

Base machine

VM software

Guest OS

Virtualization : Virtualization is use to create a virtual version of resources such as server, or application or tool.

Base machine → 16 RAM, Hard disk 1tb

Guest OS

4 GB RAM 100 GB

Using VM software we can create Virtual machine

Using Docker is can create containerization.

Docker

Earlier applications were deployed on Physical Servers

Then came virtualization technology, where we could create multiple VMs on each physical server and we could deploy applications on each VM

Now with containerization technology, we can either create containerized applications either on physical servers or on VMs

Let us see the difference between VMs and Containers ➔

Problems with VMs

1. You need to dedicate resources for the virtual machine [VM] like cpu, mem, storage, network, etc
2. VMs are resource intensive
3. VMs have their own full flagged operating systems
4. Since each VM has its own OS, therefore VMs are slow to start [kernel has to load first]
5. The size of VMs are usually in GB's
6. Since each VM has its own OS, therefore there will be licensing cost of the OS per VM
7. There is a limitation on the number of VMs that you can run simultaneously on a physical server
8. VMs are very bulky so it cant be shared easily

How containers overcome the problems associated with VMs ➔

1. You dont need to dedicate resources for the container to run
2. You also have an option to set limits on the resources that a container can utilize
3. Containers dont have a full flagged operating system
4. Containers uses the Kernel of the host operating system
5. You can start a container very fast [in few seconds]
6. You can run as many containers you want on the host machine
7. You can create a container either on a physical machine or on a virtual machine
8. A container runs as a process on the host machine
9. Containers are very light weight
10. The size of containers are usually in KBs or MBs
11. Containers provides an isolated environment for your applications to run
12. You can package everything in/inside the container that are required for the application to run

13. You can share containers with others/other teams very easily and be sure that everyone you share with gets the same container that works in the same way
14. Containers contain everything needed to run the application, so you dont have to rely on the softwares installed on the host machine

What is Docker?

Docker is an open platform for developing, shipping and running applications anywhere.

Docker provides the ability to package and run an application in an isolated environment called container.

This isolation allows you to run multiple containers simultaneously on the host machine.

What are Images [docker images]??? ➔

An image is a read-only template with the instructions to create a docker container.

Often an image is based on another image with some additional customizations or changes You can either create your own image or use images created by others.

What is a container? ➔

A container is basically a runnable instance of an image. You can even create a new image based on the current state of a running container.

What is a docker file??? ➔

Docker file automates the process of docker image creation.

A docker file contains a set of instructions/commands that the docker engine will run in order to create a docker image.

Docker Architecture ➔

Docker uses a client-server architecture.

The Docker client talks to the Docker daemon, which does the heavy lifting of building, running, and distributing your Docker containers.

Docker registries ➔

A Docker registry stores Docker images. Docker Hub is a public registry that anyone can use, and Docker looks for images on Docker Hub by default. You can even run your own private registry.

To work or do hands-on on Docker ➔

1. A Linux VM [ubuntu/centos/.....] with Internet access [MUST]

2. Docker ID [https://hub.docker.com/]

Every container has a default container command. When the default container command stops the container will have nothing to do so the container will also stop.

If you want to run a container and also login to it. You need a interactive terminal

-i => interactive

-t => terminal

How to run a container in Detached mode? ➔ -d option

-p => port mapping => -p host-port:container-port => -p 8080:80 => if you access the host on port number 8080 then you will be
redirected to container port number 80.

Virtualization and Containerization

Virtualization is a abstract version of physical machine while containerization is the abstract version of an application.

Docker Container : It is also known as instance of image or Running instance of Docker image container turn the actual application.

Docker Image : A Docker image contains everything you need to run our application.

Docker File : Docker file is a blue print of set of instruction which help to create the image.

Docker registry : Docker registry is use to publish our images. So we can pull as well as push our images in Docker Registry.

Docker hub : it is like a git hub which help to hold the open sources images.

Create the docker hub account : sign up

Link with our personal account.

Docker commands

docker --version : Shows the docker version

docker images : This command is use to find all images running in the machine

docker pull hello-world : This command is use to pull the images from Docker hub account.

docker pull busybox

docker run –it busybox

docker run -it ubuntu bash

docker pull alpine

docker run –it alpine

Case 1:

Creating custom image to display the simple message through docker image

Dockerfile

```
FROM busybox
CMD["echo","Welcome to Simple Message through Docker"]
```

Create the image using command as

docker build -t my-busybox-img . –f Dockerfile

docker run imageName/imageId

Case 2:

Creating Image to run the Java program :

```
Public class App {
    Public static void main(Stringargs[]){
    System.out.println("Welcome to Java Programs using
Docker");
    }
}
```

Dockerfile

```
FROMopenjdk:11
COPYApp.java.
RUNjavacApp.java
CMD["java","App"]
```

Create the image using command as

```
docker build -t my-java-img . -f Dockerfile
```

```
docker run imageName/imageId
```

Case 3:

Create the Spring boot application with web starter :

Create one or more than one rest api.

Once project created now using mvn command we have to create the jar file

Plz open the command in the location where pom.xml file is present.

`mvn package` : this command is use to create the jar file.

Then create the Dockerfile

```
FROM openjdk:11
COPY ./target/SpringBootWithDocker.jar .
CMD ["java",
"-jar","SpringBootWithDocker.jar"]
```

```
docker build -t spring-boot-batch8 . -f Dockerfile
```

```
docker run -d -p  8080:8080 spring-boot-batch8
```
or run on browser with link
```
docker run -d -p 8181:8080 spring-boot-batch8
```

Right side- Actual port No
Left side- Expose port No

```
docker ps
```
or these command is use to check running container
```
docker container ls
```

```
docker ps -a
```
or
```
docker contains ls -a
```
these command is use to check all container present in our machine (it may be running or stopped).

```
docker start containerId
```
start the container

`docker stop containerId`	stop the container
`docker rm containerId`	remove the container
`docker rm 3cf836d7706f –f`	This command is use to remove container forcefully
`docker rmi imageId/imageName`	these will give error
`docker rmi imageId/imageName –f`	these command use to remove images

Case 4:

Creating image for the angular project:

1) Create angular new project

ng new angular-with-docker

in template file do some changes according to your requirements.

After created the project we have to build this project.

2) Using command as : These will create dist file

ng build

3) Create Dockerfile outside of src.

FROM nginx

COPY ./dist/angulae-with-docker/ /usr/share/nginx/html

Note: /usr/share/nginx/html this part is fixed

nginx server default port number is 80

4) To create docker image

Docker build –t my-angular-img . –f Dockerfile

5) Run on command:

Docker run -d -p 80:80 my-angular-img

Docker run -d -p 81:80 my-angular-img

6) To Change Container name:

Docker run -d -p 82:80 --name my-angular-container my-angular-img

Share the images on Docker Hub :

1) Creating Tag for that image which we want to share

docker tag <imageName> <dockerhub-id>/<imageName>:version

i.e docker tag my-ang-img <dockerhub-id>/my-ang-img:2.0

2) Pushing images on Docker Hub:

docker push <dockerhub-id>/my-ang-img:2.0

if you get error as authentication then write the command as-

docker login

id :

password :