Docker

Docker is an open-source platform for developing, shipping and running application. Docker is a containerization platform that allows developers to package application and their dependencies into lightweight, portable containers.

-->Docker is a cline server architecture

-->developer write the docker file.

-->developers and operation teem also write the docker file.

Why use docker

**Portability:** runs consistently across different environments (development, testing, production, cloud).

**Lightweight:** uses fewer system resources compared to virtual machines.

**Fast deployment:** containers start instantly.

**Scalability:** easily scales application using orchestration tools like Kubernetes.

How dockers works:

**Docker file:** a secret defining a how to build a container.

* A docker file is a text file containing instructions that tell docker how to build a container image.

**Docker image:** a snapshot application and dependence.

* A **Docker Image** is a lightweight, standalone, and executable package that includes everything needed to run an application—code, runtime, libraries, environment variables**,** and dependencies. It acts as a blueprint for creating Docker containers.

**Docker container:** a running instant of docker image.

* Docker container is running instance of image.

A Docker container is a lightweight, standalone, and executable unit that packages an application along with its dependencies, libraries, and configurations. Containers run from Dockerimages and provide a consistent environment across different systems.

**Docker hub:** a cloud-based registry for sharing document image.

* Docker is repository service and it is a cloud based services and where people push there docker containers image and also pull the image docker container image from docker hub any

time via the internet.

Mainly DevOps team uses the Docker Hub. It is an open-source tool and freely available for all operating systems. It is like storage where we store the images and pull the images when it is required.

Docker uses:

Software developers write code abstract the separate build package stored an image in central use light weight mission runs docker uses.

Docker engine:

The software that hosts the containers is named docker engine.

Docker Engine is the core, open-source containerization technology that allows you to package, ship, and run applications in containers, providing a consistent environment across different infrastructures.

* Docker image will be stored in containers
* Along with the containers image will be stored in docker hub.
* Docker image will be pulled into multiple environments &

Using one image one container will be created.

* For n no of image ---- n of containers will be there docker hub.
* Used to take the back up of the both images & containers.

How to install the docker ------

1.create a docker hub account

2.create 2 repo in docker hub

3. create 1 instance

4. Sudo -su

5.ifconfig -a

6. yum install -y docker

7. docker --version

8. ifconfig -a

9. service docker start

10. default Docker Root Dir: use-🡪 cd /var/lib/docker

11. docker images or docker image ls

12. docker PS

13. docker PS -a --> to see all containers

14.to pull the image

1.pulling a docker image from online (official website).

2.by writing the docker file

15. I am pulling the docker image from online

--> to pull an image called -nginx (web server)

--> command-- docker pull nginx-- to pull the image from online

--> docker images -- to check images pulled or not

--> docker inspect image -- to know complete information

**20-05-2025**

Steps to create docker images and pull and push how to use commands.

Open docker hub account and create 2Repositories

Names: dev, qa

Open AWS account and create one instance name is docker then connect to the git bash use this

* ssh -i "docker.pem" [ec2-user@ec2-13-49-72-26.eu-north-1.compute.amazonaws.com](mailto:ec2-user@ec2-13-49-72-26.eu-north-1.compute.amazonaws.com).

then connect to the git bash.

change the root use command 🡪Sudo su

change Docker Root Directory: command use

🡪 cd /var/lib/docker

🡪this command shows the images -- Docker images

🡪this command used to list running containers. Its use to check the containers states which one is active and there details.

* Docker ps
* Docker ps -a this command show all containers details

1. This command will download the latest official Nginx image from Docker Hub.

Use-- docker pull nginx

2.to know complete information of image

🡪 docker inspect image

3. this command used to where it as pull or no

🡪Docker images

And show the all-local images

5.push the images in docker hub

🡪 Docker tag Nginx: latest saibaji/dev/sai

Docker push saibaji/dev:sai

6.create container to push images

🡪 docker run -itd –name Umesh -p 40:80 53a18edff809

whether its containers created and running chick it then shows containers

🡪Docker ps

7.then check it is running or no

<http://13.49.74.50:40/>

8. stop the running container

🡪 Docker stop sai

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

2. ubuntu

1.pull the ubuntu in docker hub

🡪docker pull ubuntu

2.the show pull or no

🡪Docker images

3.Push the images in docker hub

🡪 docker tag ubuntu: latest saibaji/dev:ubu

🡪 docker push saibaji/dev:ubu

4.Crate container to push image

🡪docker run -itd –name my-ubuntu -p 8080:80 ubuntu bash

Install Apache server

🡪apt update && apt install -y apache2

5.To start the service

🡪service apache2 start

6. If any containers are running stop them by passing container name

🡪docker stop <myubuntu>

7. Check whether container iscreated and running

🡪docker ps

8. To check the image getting or no

<http://13.49.72.26:8080>

3.httpd

1. pull the ubuntu in docker hub

🡪docker pull httpd

2. the show pull or no

🡪Docker images

3. Push the images in docker hub

🡪 docker tag httpd: latest saibaji/dev:httpd

Docker push saibaji/dev:httpd

4. Crate container to push image

🡪docker run -itd –name my-httpd -p 80:80 httpd

5. Check whether container is created and running

🡪docker ps

6. To check the image getting or no

<http://13.49.72.26:80>

7. to stop the running container

🡪docker stop httpd

4.tomcat

1. pull the tomcat in docker hub

🡪docker pull tomcat:8.0.52

2.the show pull or no

🡪Docker images

3. Push the images in docker hub

🡪 docker tag tomcat:8.0.52 saibaji/dev:tom

Docker push saibaji/dev:tm

4. Crate container to push image

🡪docker run -itd –name tomcat-server2 -p 8080:80

b4b762737ed4

5. Check whether container is created and running

🡪docker ps

6. To check the image getting or no

<http://13.49.72.26:8080>

7. to stop the running container

🡪docker stop tomcat-server2

5.python

1.pull the python in docker hub

🡪docker pull python

2.The show pull or no

🡪Docker images

3.Push the images in docker hub

🡪 docker tag python:latest saibaji/dev:pyt

Docker push saibaji/dev:pyt

4. Crate container to push image

🡪docker run -itd –name python -p 50:50 python

5.Check whether container is created and running

🡪docker ps

7.to run the python command in docker by using

🡪docker exec -it python python --exit

8.to stop the running container

🡪docker stop python

6.mysql

1. To pull the image using command

Docker pull mysql

2. Whether image pulled or not

Docker images

3 .To push the images in docker hub account

Docker tag mysql:latest saibaji/dev:sql

Docker push saibaji/dev:sql

4. To create container to push image

docker run –name MySQL -e

MYSQL\_ROOT\_PASSWORD=Saibaji@12 -d mysql

docker exec -it MySQL MySQL -u root -p

-enter password

5. Check whether container is created and running

🡪docker ps

6. to stop the running container

🡪docker stop MySQL

7.php

1.To pull the image in php

Docker pull php

2. wither image is pull or not check it

Docker images

3.to push the images in docker hub account

docker tag php:apache saibaji/dev:my-php

docker push saibaji/dev:my-php

4.to create container to push image

docker run -d -p 8080:80 --name my-php-app

$(pwd):/var/www/html php:apache

5. containers created and running then show

Docker ps

6. then check it is running or no

<http://16.171.237.43:8080/>

7.stop the running container

🡪 Docker stop my-php-app

8. Jenkins

1. to pull the Jenkins image in

Docker pull Jenkins/Jenkins

2. whether image pull or not

Docker images

3.to push the image sin docker hub account

docker tag Jenkins/Jenkins: latest saibaji/dev:jen

docker push saibaji/dev:jen

4.to create container to push image

docker run -d --name Jenkins-server -p 8080:8080 -p

50000:50000 Jenkins/Jenkins

5. is used to view the logs of a running or stopped Jenkins

Container

docker logs Jenkins-server

6.whether container created or not

Docker ps

7.check image getting or not

<http://16.171.237.43:8080/>

8.then stop running container

Docker stop Jenkins-server