Docker

→ Docker hub is a public repository for Docker

→ **Command**

↦ ① Docker pull python
This will be like a kind of installation

↦ ② docker run python
This will install & run together

↦ ③ docker Ps → check / list out container that are running.

→ **Docker Image Vs Docker Container**

① Image is actual package with configuration + postgreSQL + startscript

② Artifact that can be moved around.

③ Not running

Container:
When we pull image on local repo & start it, Now application starts & that Create the Container environment

Running.

* We can run multiple Version of application on same System.

Docker Vs Virtual Machine.

1. They are both Virtualization tools
2. Docker Virtualizes application layer of the OS
3. VM has application layer & its own OS.

What will get installed along with docker

① Docker Engine : necessary to run docker container
② Docker CLI (command Line client) :
③ Docker Compose
④ Docker Content Trust
⑤ Kubernetes
⑥ Credential helper

docker tutorial Command.

docker run -d -p 80:80 docker/getting-started

Container has port 5000 binded to it. This makes it possible to talk to the application running inside a container

Command

docker images : check all the existing image on docker.

Once I get docker image, I need it running so that my application can connect to It., we will have to Create a container of the image, that will make it possible to connect to the redis application
↳ image.

Command

① docker run redis

new tab Once docker running

② docker ps

③ Ctrl+c → to stop redis

④ docker run -d redis
docker run in a detached mode

⑤ To restart a container → we would need the container ID

↳ docker stop Container ID → 80136...
This will stop docker Container

↳ docker start ID

⑥ docker ps -a
Show all the container that a running or not running

④ docker run redis:4.0

pull docker with particular Version

* Both Version will have same port to listen

* Container is just a VM running on the host.

Port binding

To solve the above problem of same port. We can bind the port.

Command

① docker run -p 6000:6379

-p = port of host

6000 = port to which i need to bind

6379 = port that will be binded.

② docker run -p 6001:6379

debugging

① docker logs ID or docker logs name

Gives the Entire log of that Container.

② Provide Name to Container

docker run -d -p 6001:6379 --name redis-older redis:4.0

③ docker exec -it id/container /bin/bash

it → incorrect terminal.

Here we can navigate through different type of file/folder in a container.

↳ exit → to exit the terminal.

↳ we can do it with name as well

↳ docker exec -it name /bin/bash

Run Vs Start

1] docker run will take an image with specific or latest version.

Eg: docker run redis:4.0

2] docker start we work with images.

Eg: docker run -p -d --name // run

when we stop it & restart we can just say

docker start containerid. // start

Docker N/W

Docker creates its own isolated docker n/w where the container runs.

Command

① docker network ls

② docker network create mongo-network

default port for mongo dB is 27017.

Compose → .yaml file

This is a file where the docker Com

→ file structure: docker Compose

① Version
② services :
  ⓐ container name
  ⓑ image
  ⓒ Ports
  ⓓ env variable

→ indentation is important

Start docker Container using docker Compose file.

→ Command

① docker-compose file.yaml up -d
↓
Start all container in file.yaml

② docker network ls

* When you restart a container, everything you configured in that container is gone [data loss]

③ To stop Containers.

ⓐ docker stop id

OR

ⓑ docker-compose -f file.yaml down

Stop & remove container in addition to this it also removes n/w

Deploy

To deploy application should be packaged into its own docker Container.

Docker file

It is a blueprint for creating docker images.

⇒ Syntax

- ~~from~~ image: (~~from~~ node)
- env variable
- RUN linux command

// director is created inside Container & not on localhost.

- COPY linux command

// COPY executes on the host machine

- CMD → executes entry point.

// This will alway be the final command.

Difference b/n RUN & CMD

CMD is a entry point & can have multiple Run command.

Command

→ docker build -t my-app:version .
(. → current directory.)

→ docker rmi id → delete the container.

→ docker rmi id → delete the image.

bash | sh → Try them interchangably.

AWS ECR

Step1 : Docker Registory
Create a private repository for docker

Image Naming in Docker registries

registoryDomain/imageName:tag

Command

① docker tag img rename-img

tag will rename a given img

② docker push img name

Docker Volume

3 types of Docker Volume

a docker run -v → Host Volume

⑤

② Create Volume just by refering container directory.

For each container a folder is generated which gets mounted automatically → Anonomous folder / Volume

③ Named Volumes:

Reference Volumes just by name.

Mostly used & prefered is named Volume.

Docker Volume location

Mac = /var/lib/docker/volumes

Windows : C:\ProgramData\docker\Volumes.

Docker for mac creates a Linux Virtual Machine & Store all the Docker data here.

End Screen Session

Ctrl + a + K , Y

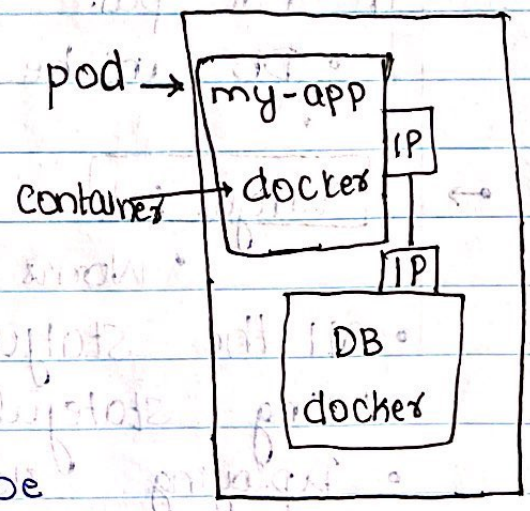
Kubernetes

This is a open source container - Orchestration tool

Node & pod

→ Node : a simple Server, physical or Virtual Machine

→ Pod : • Smallest Unit of Kubernetes
• pod is a abstraction over containers.
• pod runs one application container inside of it usually, you can run multiple container if you want.
• Each pod gets its own IP address, ie internal Ip & not public IP address, & each pod communicate with One another or dB using this ip address.
• pod dies easily.
• new Ip is assigned each time pod is created.



Service

• a static Ip that can be attached to each pod
• Life cycle of Service & pod is not connected. Even if pod dies, Service & Ip will stay

Architecture

① Each node has multiple pods
② Worker Nodes / nodes do the actual work.
③ 3 process needs to run on every node
   ⓐ Container runtime
   ⓑ Kublet interact with both the container & node, Kublet starts the pod with a container inside.
   · Communicates via Services
   ⓒ Kube proxy

Master process :

Master node/process - manage all the process
· 4 process run on Every master Server.
① API Server.
② Scheduler
③ Controller Manager.
④ etcd → a key value store of a cluster state

minicube

· One node cluster, where the master process & working process, both run on One node. &
· Creates a Virtual box on local host

Kubectl

command line tool for Kubernetes (K8s) cluster.