Docker

1. docker -> it allows us to create a container
2. container:-windows app container required windows host
   linux app container required linux host
3. docket help us to run and application in isolated environment
4. Parts of Docker

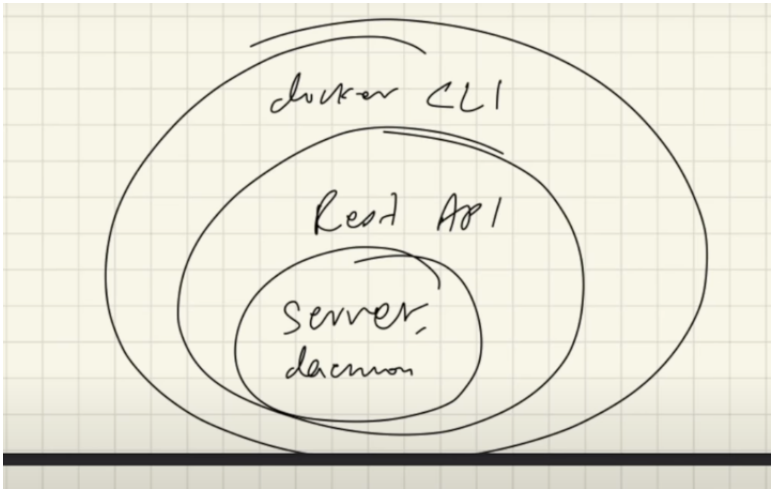
   ▼ 1. Docker Runtime

   Runtime → Docker engine → orchestration
   Runtime → allows us to start and stop container

   1. Low leve Runtime (runc) → it works with os and start and stops the containers
   2. ContainerD → its role is to manage runc, container interaction with the network
      pulling the images

   ▼ 2.Docker Engine

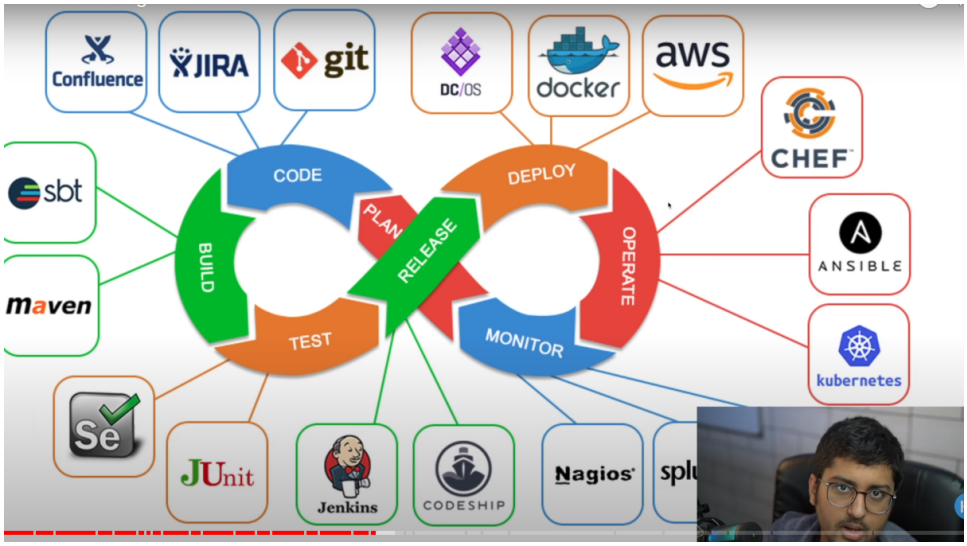
   we use it to interact with docker { Docker Daemon }
   it creates client server architecture



▼ 3.**orchestration**

update the version , remove the systems , scale the systems
toh ye sab automatically hona chahiye

5. see once we have to show ship the running application it is difficuilt but we can share instruction to someone with the help of file called docker image
6. DockerFile —> Image —> Container
   and image can be shared to the other people
7. Different Container Runtimes:
   a. container d
   b. lxc
   c. runc
   d. cri-o
   e. rkt
8. Which container should we use ➖
   Open Container Initiative
9.



💡 1 . Interactive enviroment : dont exit out of the container

2. If we attach a bash to the end of the commnadn of the docker file then both the terminals gets connected

3. read for the docker networking from the different sources

Docker Commands

▼ 1.Docker Run Hello-world

this is basic command to run for the docker file at first file is not present there so it can no track it so first it downloads from the docker hub and then ready to server and after installing it if we again hit the command then it doesn't say that the file is missing
hello-world is image name

▼ $ docker images

it is used to check all the current running images

▼ $ docker ps

it gives the list of the current running containers

▼ $ docker container ls

it will show all the containers

▼ $ docker stop id

to stop the container with given id

▼ $docker ps -a

it will show all the containers that has been stopped

▼ $ docker rm <id>

to remove the docker containers

▼ $ docker inspect <name>

it gives all the information regarding the container

- $ docker container prune -f

  it deletes all the container
  prune means delete all the containers
  -f means dont ask anyting just do it

- $ docker run alpine ping www.civo.com

  it will download the alpine and will ping to the civo.com

- $ docker -d run alpine ping www.civo.com

  it will return id of conatiner and will ping it in the background

- $ docker logs —since 5s <container id>

- $ docker rmi alpine -f

  it will remove all the images from the docker

- $ docker run -d nginx

  it will run the container niginx with default portal

- $ docker run -d -p 8080:80 nginx

  it is like forbidding the port …
  it means when we are going to port 8080 it will get the containers from 80

- $ if we want

  to share the container for the images inthe docker to the friend for that we can use commit messages

- $ docker commit -m “added names file.txt “ 062a4370

- $ docker images -q :- it gives ids of the images

- $ docker rmi (docker images -q)

▼ $ docker images -q —no-trunc
this returns the hash values of the images and it gets started with the image id letters

▼ Docker Layers :

there are certain custom things which run called as layers
ex. ubntu image is the collection of 4 files that it self made up of images
some files are directly imported as they are alredy downloaded while the previous image was dowloaded if comong images are found they are not going to dowload them it makes it very fast

Create Our Own Images :- Docker files