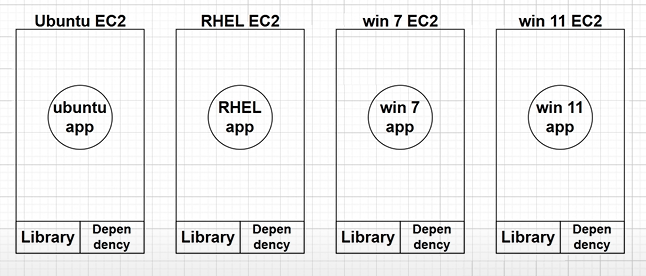
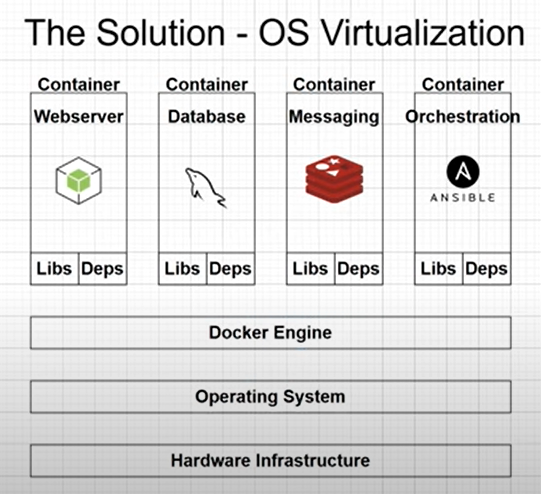
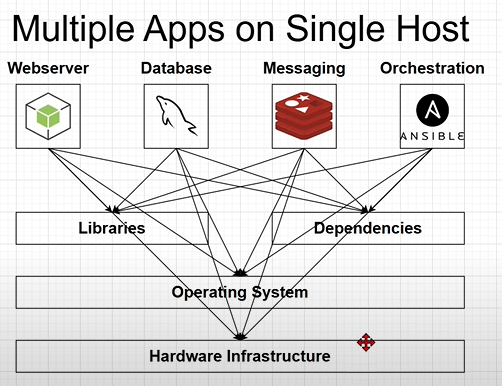
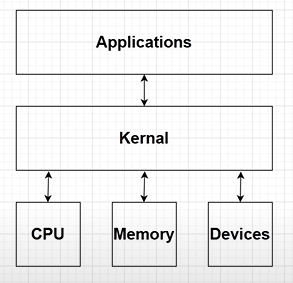
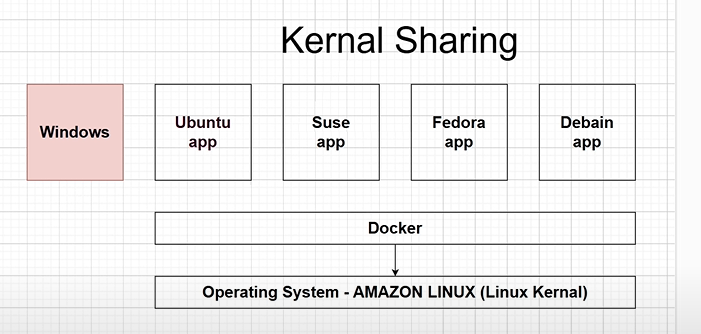
**PART-1 --> Master Docker from scratch for Beginners**

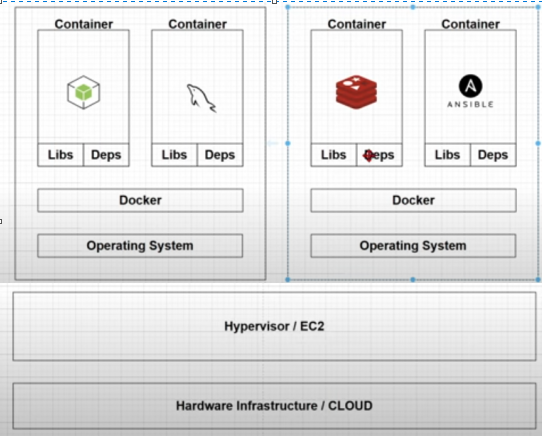
* Dual OS --> HyperVisor --> Splitting OS
* The 20% to 30% consumes storage and memory by default in order to run OS
* Cost --> 4 ec2-instance 400$
* In one OS, to install and run multiple applications, we need dependencies and libraries
* Dependencies clashes
* Libraries clashes
* Hardware clashes

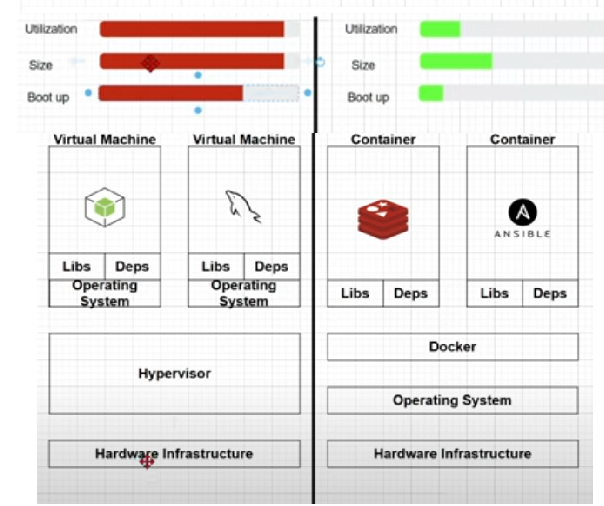


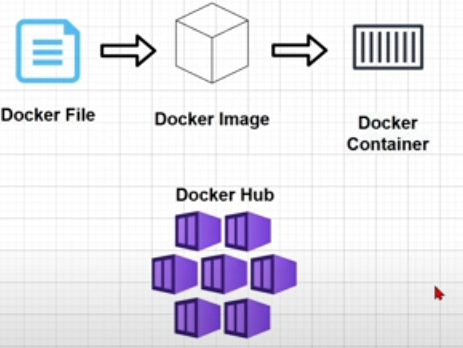








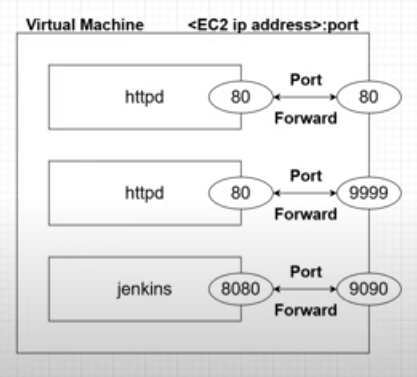




* Hub.docker.com --> search in goggle official link

**Ec2-ubuntu**

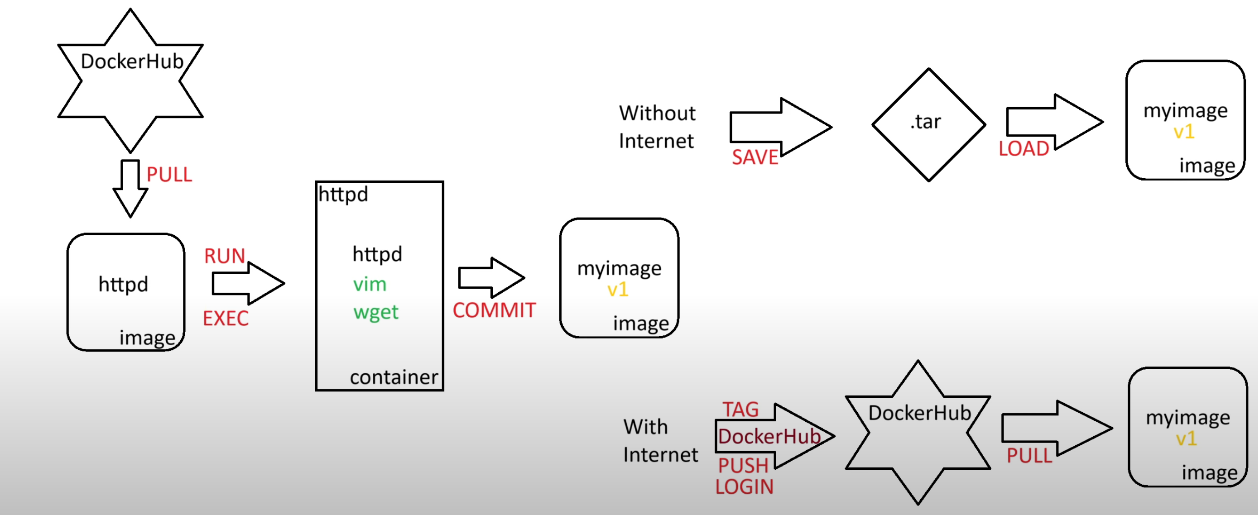
* **in security groups --> TCP 0-65535 --> to expose port number to internet**
* **sudo su - root**
* **apt install docker.io**
* **apt-get update**
* **docker**
* **docker images --> display images**
* **docker pull httpd**
* **docker ps --> display containers and status running only --> ps means process status**
* **docker ps -a --> display containers and all status like running, stoped, deleted…etc --> -a means all**
* **docker run -itd -p “9123:80” httpd --> -p means publish/port, 9123 means host port, 80 means container port and httpd means image.**
* **docker run -itd --name “container name” -p “3213:80” httpd**
* **docker pull nginx**
* **docker run -itd --name “container name” -p “9213:80” nginx**
* **docker run -itd --name “my jenkins” -p “8888:8080” jenkins/jenkins**



* docker stop container\_id --> to stop container
* docker rm container\_id --> to remove container
* docker rm -f container\_id --> to remove container forcefully without stopping
* docker rmi image\_name --> to remove image --> rm means remove container and rmi means remove image --> not able to delete if any containers using image
* cat /etc/os-release --> to see which os family this server belongs to.
* docker exec -it container\_name /bin/bash --> to go inside the container--> exec means execute
  + cat /etc/os-release --> inside container we can see which os family
  + cd htdocs/
  + cat index.html
  + apt install vim --> inside container
  + apt-get update --> if unable to install update
  + exit --> logout container
* docker top container\_name --> to display the running processes of a container
* top --> all processes --> linux task manager
* docker stats --> ctrl c --> exit --> to display cpu, memory, storage …etc
* docker inspect container\_id --> to display container information
* docker inspect image\_id --> to display image information --> inspect means information

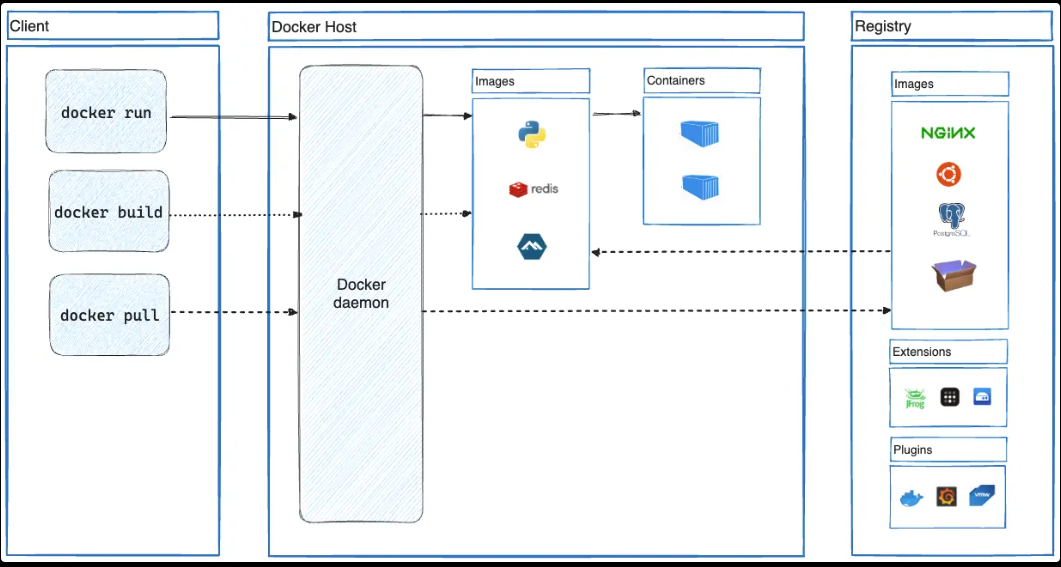
**PART-2 --> Master Adv Docker CLI, Docker Hub, Volumes & DockerFile**

* Docker playground --> goggle --> play with docker --> to practice docker commands
* apt-get update
* apt install docker.io
* docker images
* docker pull httpd
* docker run -itd -p “80:80” httpd
* docker ps -a
* docker exec -it container\_name /bin/bash
  + apt install vim --> inside container
  + apt-get update --> if unable to install update
  + vim --> it is a software
  + apt install wget
  + wget --> it is a software
* docker commit container\_id image\_name:version1 --> converting container as a image.
* Two ways to share images to team members
  + Without internet
  + With internet
* Without internet --> it extracts to a single tar file inside multipule files
  + docker save -o ./myimagebackup.tar myimage:v1 --> -o means output
  + ls -lh --> list of files and folder with size
  + scp --> linux to linux file transfer
  + docker rm -f container1 container2
  + docker rmi image1 image2
  + docker load -i myimagebackup.tar
  + docker images
* With internet --> push to dockerhub
  + docker tag myimage:v1 iammithran/newimage:v1
  + docker images
  + docker push iammithran/newimage:v1
  + docker login --> use link to give access
  + docker pull iammithran/newimage:v1

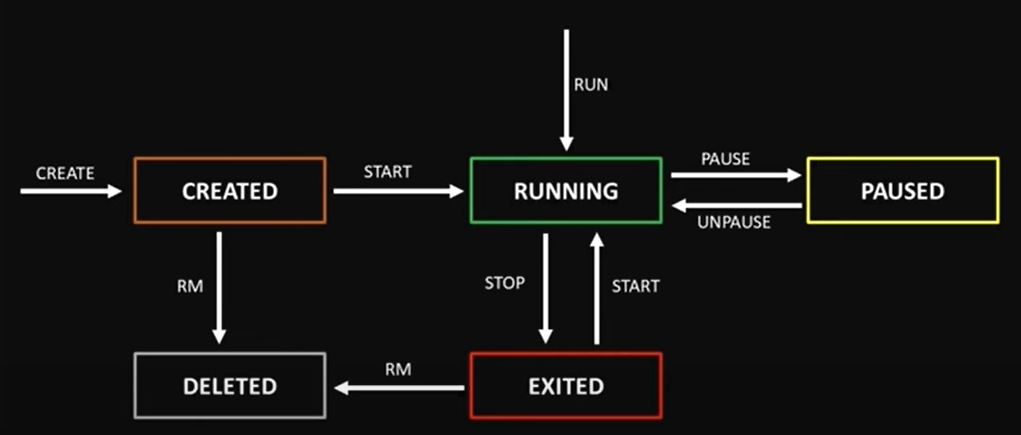




**Docker architecture:-**

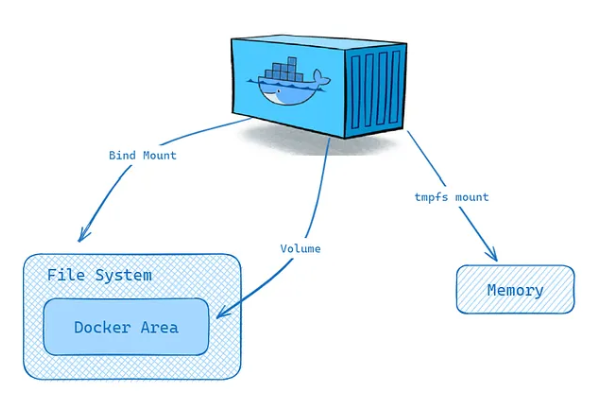


**Docker life cycle --> container life cycle**



* docker run --> shortcut -->run is combination of create and start
* Dlw pause and stop
  + Pause means it will hold the cpu, memory --> fast restart.
  + Stop means freeup the cpu, memory --> it takes some time to restart.
* docker run -itd -p “80:80” iammithran/newimage:v1
* docker exec -it container\_id /bin/bash

**Docker Volume**



**VOLUMES**

* EBS --> Elastic Block Store

**Bind Mount**

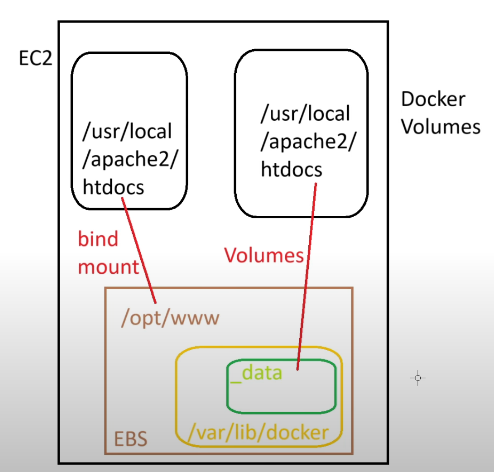
* cd /opt/ --> extra addons --> extra additional packages
* docker run -itd -p “9000:80” -v “/opt/www:/usr/local/apache2/htdocs” iammitran/newimage:v1 --> -v means volume --> ebs binding to a container --> synchronized

**Volume**

* cd /var/lib/docker/ volumes --> docker installed path in local machine
* docker volume ls
* docker volume create volume\_name
* docker run -itd -p “9000:80” --mount source=my-vol,target=/usr/local/apache2/htdocs iammitran/newimage:v1

**Tmpfs mount**

* It is used for container performance and security purposes.
* If the container is deleted, the files are also deleted.



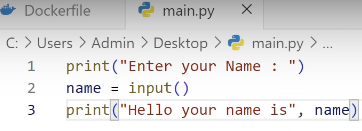
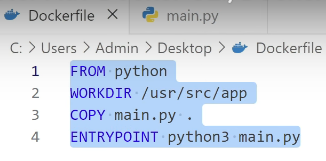
**Docker File**

* vi dockerfile
  + FROM httpd
  + RUN apt-get update && apt install vim -y
* docker build -t myimage:v1 /root/ --> -t means tag --> /root/ means docker file path
* vi dockerfile
  + FROM myimage:v1
  + RUN apt install wget -y
* docker build -t myimage:v2 -->To create customized images.
* docker run -itd -p “99:80” http

**-d detached mode (run in background) --> generated logs**

* docker logs container\_name

**-i interactive mode or input**

* docker build -t mypythonimage:v1 . --> . current working directory
* docker run -i mypythonimage:v1 --> -i means to interact with the container

**-t terminl**

* docker run -it mypythonimage:v1 /bin/bash --> bourne again shell
* docker run -it mypythonimage:v1 sh --> old terminal
* docker run -i mypythonimage:v1 python3 --> python terminal

**EC2-linux**

* sudo su - root
* yum update
* yum install docker
* systemctl start docker
* systemctl status docker
* docker images
* docker ps -a
* docker pull httpd
* docker exec -it container\_name /bin/bash
  + apt-get install -y vim
  + apt-get update
  + apt install nano
  + apt-get remove -y vim
* docker login -u susmitha789257
  + dckr\_pat\_akAO9V3\_E1a9ggZ4D11BWnuvr\_I --> access token