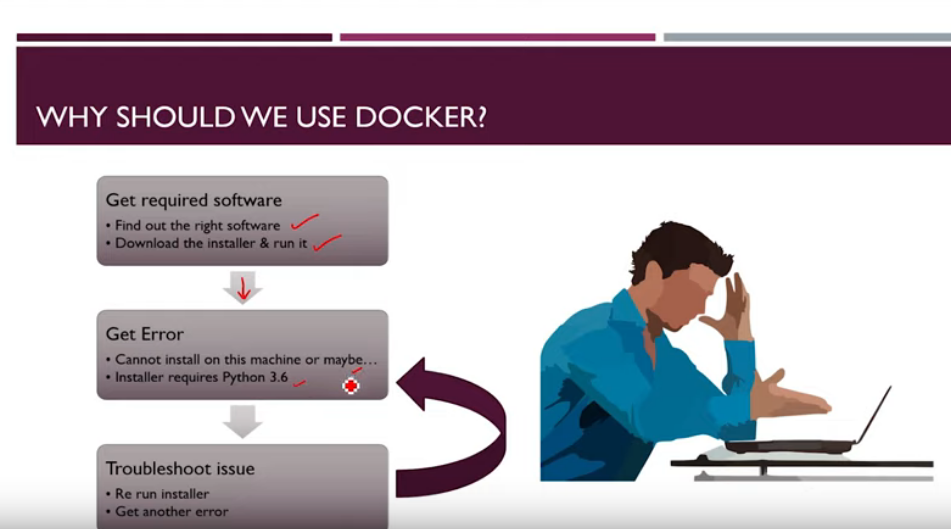
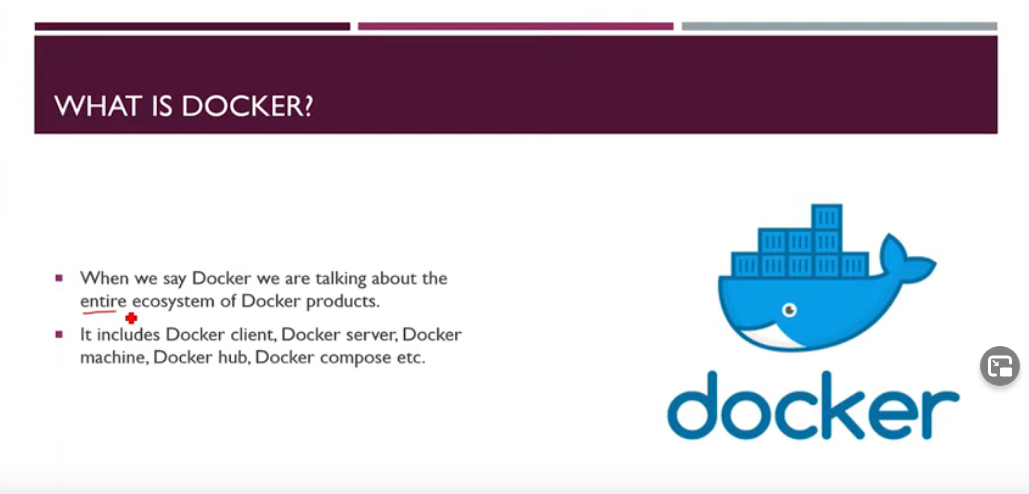
***Docker KeyNotes***

**1) Why Docker ?**



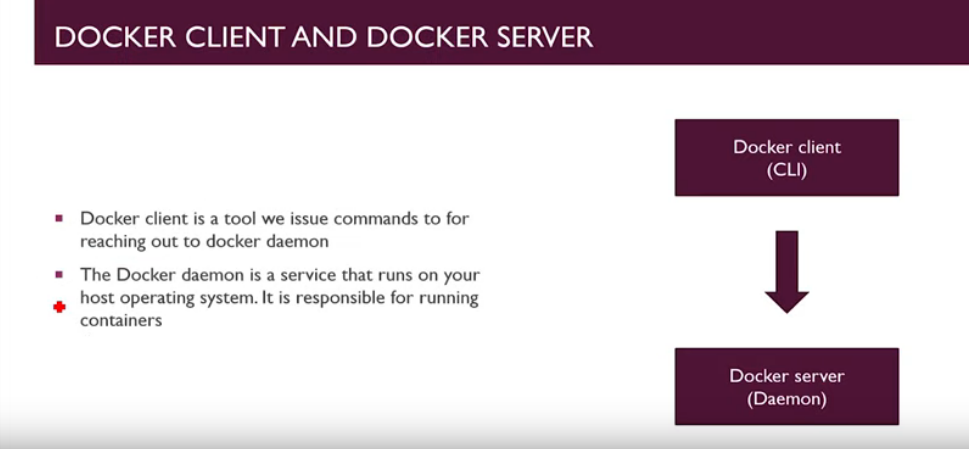
**2) What is Docker ?**

**- Docker is a set of platform as a service products that use OS-level virtualization to deliver software in packages called containers.**

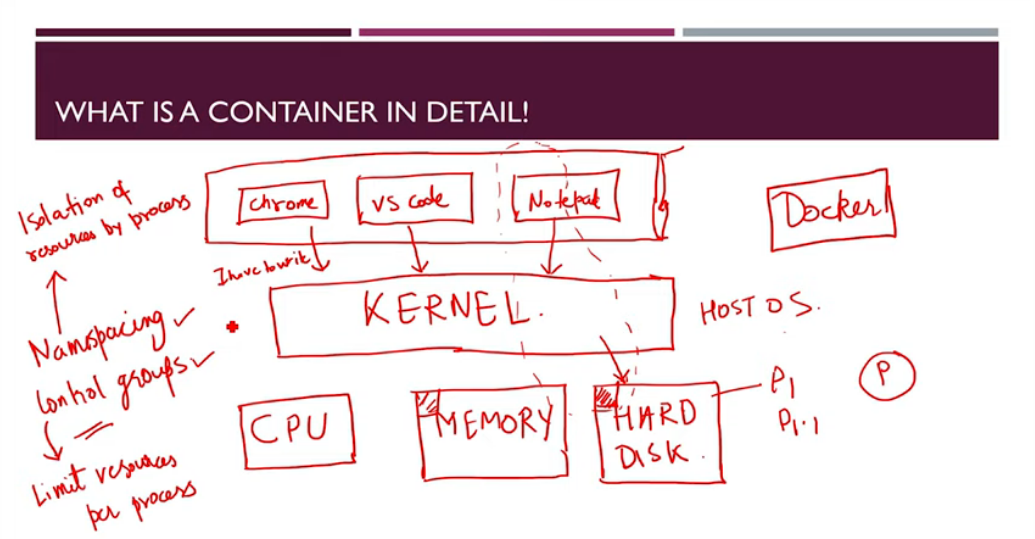
**-** [**Docker**](https://github.com/docker/docker)**is a tool designed to make it easier to create, deploy, and run applications by using containers**

**- Docker is an open platform for developing, shipping, and running applications. Docker enables you to separate your applications from your infrastructure so you can deliver software quickly. With Docker, you can manage your infrastructure in the same ways you manage your applications.**

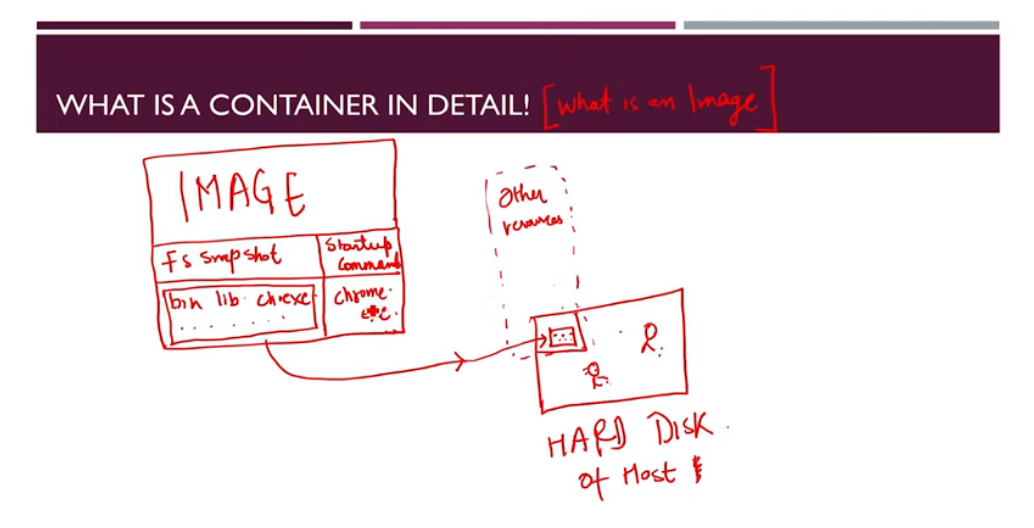
**3) Docker Client(CLI) & Server(Daemon)**

****

**4) Container**

****

**5 ) Image**



**# Some Basics Commands For Docker**

**i) Installation in Ubuntu**

**Using Docker Engine**

\* sudo apt-get update

\* sudo apt install docker.io

**or**

**Using Conventional Scripts**

\* curl -fsSL https://get.docker.com -o get-docker.sh

\* sudo sh get-docker.sh

**ii) Getting Started**

\* docker run hello-world

It will check the image file locally , if not available then it will go to docker hub and then download it to local and also keep record in cache , so that if next time we will need , it will directly redirected from our cache.

When we execute “docker run” command , default command run and then exit.

docker run = docker create + docker start

\* docker create hello-world

It will create a container and result a hash string, but it is not started yet.

\* docker start -a <hash strng resulted after creation>

“-a” = attach to container and show the result

\*docker ps

It shows the running container

\* docker logs

It will show the logs of that container( what is running inside )

**iii) Stop Docker**

\*docker stop <conyainer id>

Stop that specific container

\* docker kill <container id>

It will forcely dtop that container.

It give a sick kill messaged the process to close. It means stop write away.

**iv) Running another command in a container**

\* docker exec -i -t <container id> <command>

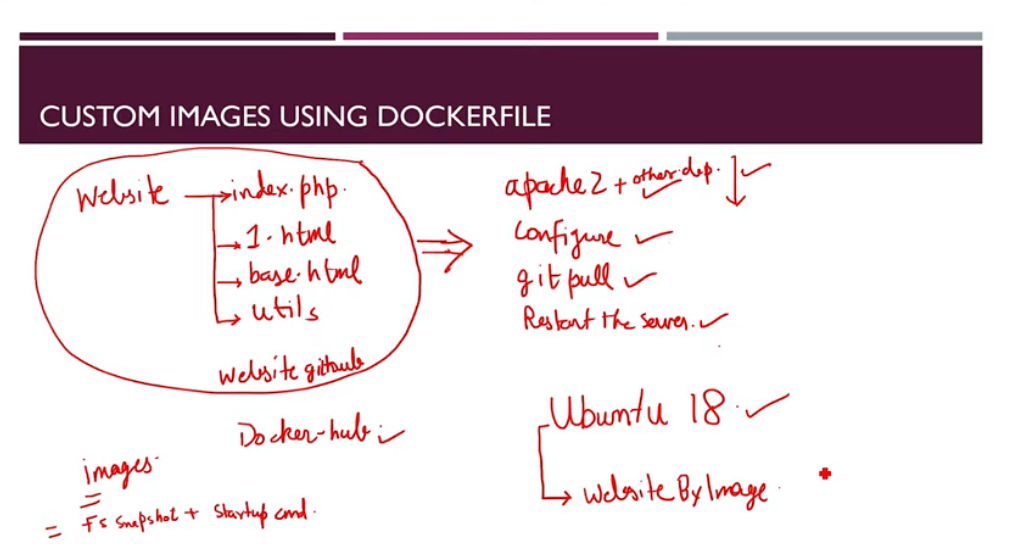
e.g- docker exec -i -t c&5fdvhfd sh

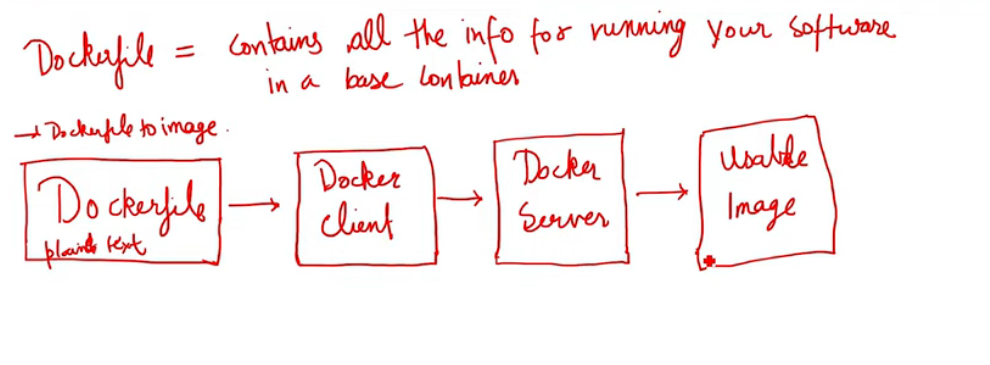
“-i” : is a flag , which runs inside or stdin

“-t”: pretty formating

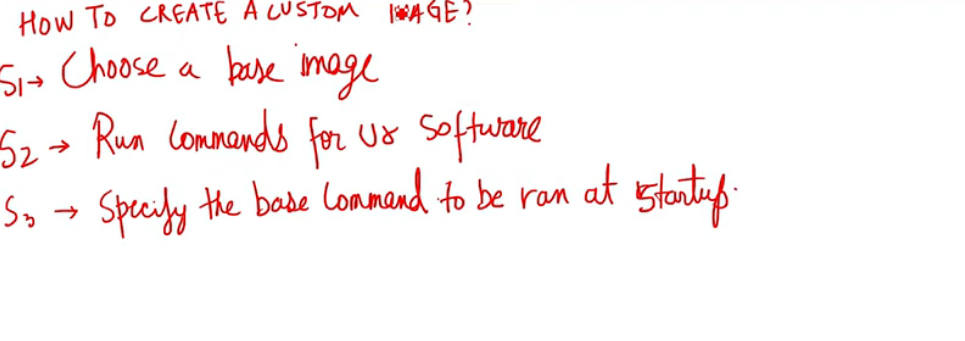
“sh” : is a command processor , like a terminal which start the terminal inside that container

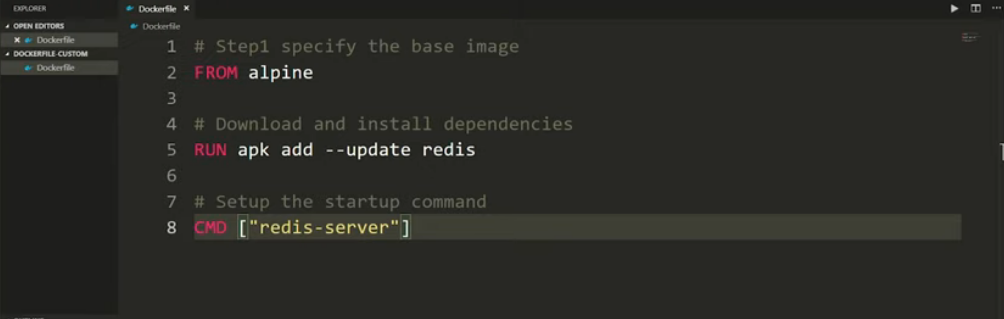
**6) Custom Operations**

****



**7 ) How to Create a Custom Image**

****

**e.g –** create a file named Dockerfile

after creation of Dockerfile , we need to run some command as follows :

\* docker build .

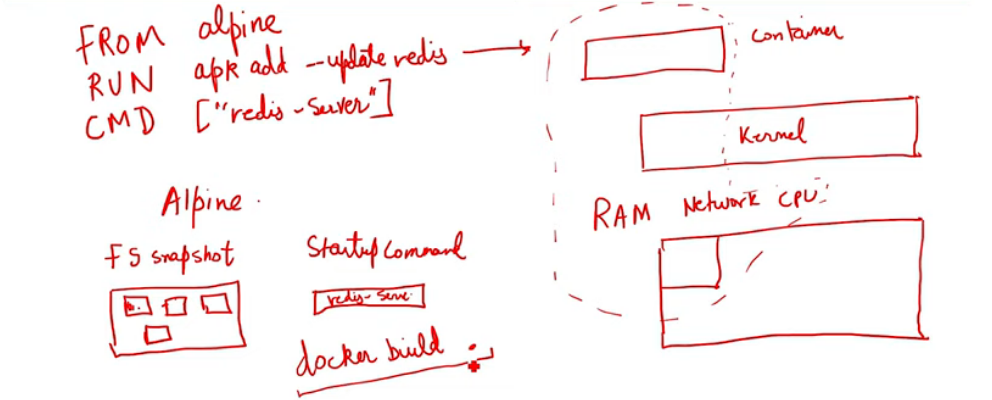
It will return some container id

Now , we can run our custom docker as

\* docker run <our container id>

e.g – here container is “redis-server”

explaination of example Dockerfile:



# Tagging Our Custom Images:

\* docker build -t aaditya/redis:latest .

“aaditya”: docker username

“redis” : image name

“latest” : version

after execution , it will tag our custom image

and we can call our image file using that name.

# Starting our custom Image:

\* docker run aaditya/redis

It will start our custom image

#Image Commit

\* docker commit c ‘CMD “redis-server”’ <container id>