**Docker**

Docker is a centralized platform for packaging, deploying, and running applications. Before Docker, many users face the problem that a particular code is running in the developer's system but not in the user's system. So, the main reason to develop docker is to help developers to develop applications easily, ship them into containers, and can be deployed anywhere.

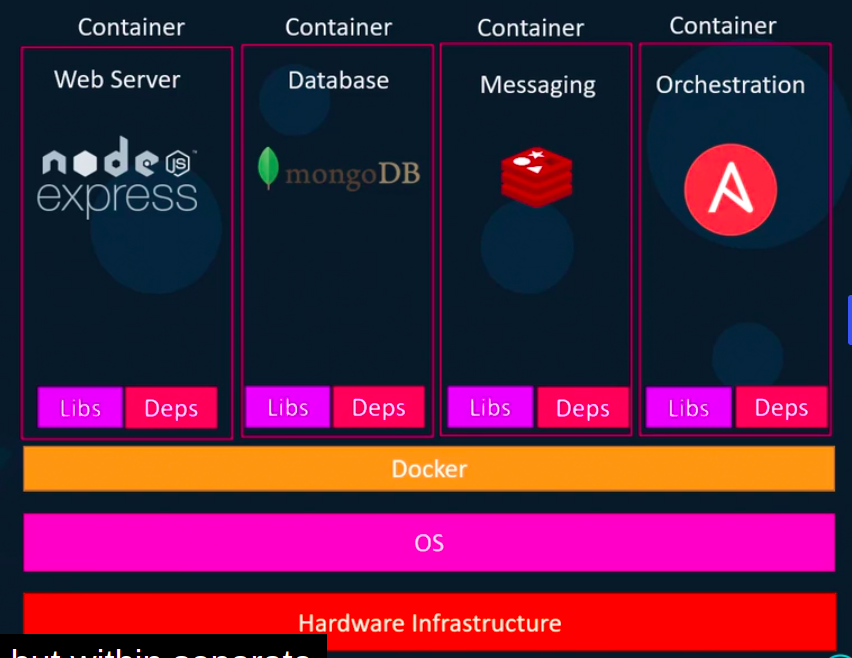
Docker is an open-source centralized platform designed to create, deploy, and run applications. Docker uses container on the host's operating system to run applications. It allows applications to use the same Linux kernel as a system on the host computer, rather than creating a whole virtual operating system.

**Maintainability.**

### Docker Containers

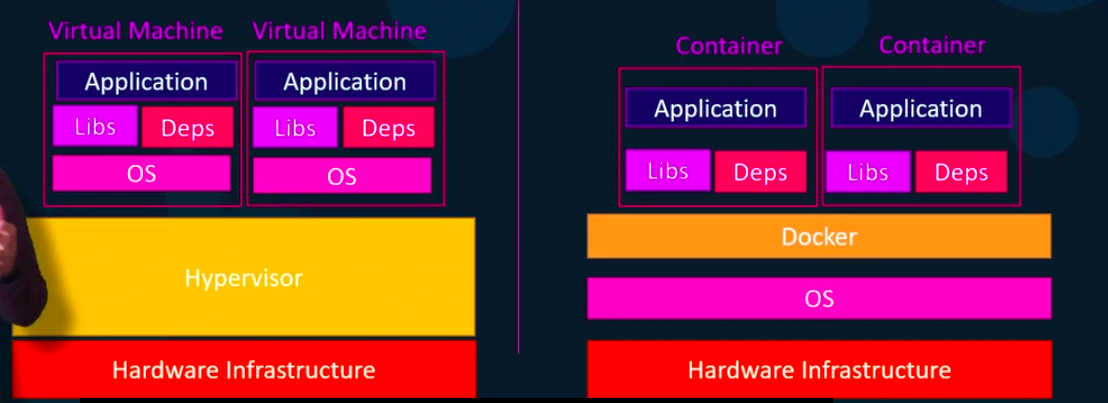
Docker containers are the lightweight alternatives of the virtual machine. It allows developers to package up the application with all its libraries and dependencies, and ship it as a single package. The advantage of using a docker container is that you don't need to allocate any RAM and disk space for the applications. It automatically generates storage and space according to the application requirement.

(or)

container are complete **isolated environment , they can have there own processes or services there own network interfaces**.except they all share the same OS kernel.Docker utilizes LXC containers. **Containers are portable.**

### Containers Vs. Virtual Machine

|  |  |
| --- | --- |
| Containers | Virtual Machine |
| Integration in a container is faster and cheap. | Integration in virtual is slow and costly. |
| No wastage of memory. | Wastage of memory. |
| It uses the same kernel, but different distribution. | It uses multiple independent operating systems. |



**Docker images:**

**image is a package or a template just like template it is used to create a one or more containers.**

Containers are running instances of images that are isolated and have there own environments and set of processes.

Docker images are the read-only binary templates used to create Docker Containers. It uses a private container registry to share container images within the enterprise and also uses public container registry to share container images within the whole world. Metadata is also used by docket images to describe the container's abilities.

Or

Images are used to store and ship applications. An image can be used on its own to build a container or customized to add additional elements to extend the current configuration.

**Docker commands:**

docker run nginx=> run the instance nginx of application.

Docker ps => print all running containers.

Docker ps -a => shows all previously and now running container.

Docker stop ContainerName. => to stop container.

Docker rm CotainerName => to remove the container.

Docker images => list of images.

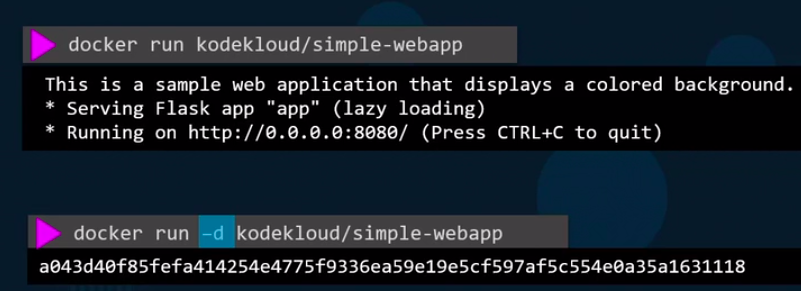
Docker rmi nginx (IMAGE\_NAME)=> to remove the image (first remove all containers.)

docker pull nginx => pull and store in locally

docker run container sleep 5 => sleep for 5 sec.

Docker exec distract\_mcclintock cat /*etc*/hosts => executes

Run – Attach and detach :



docker attach IDNUM(a04f9)

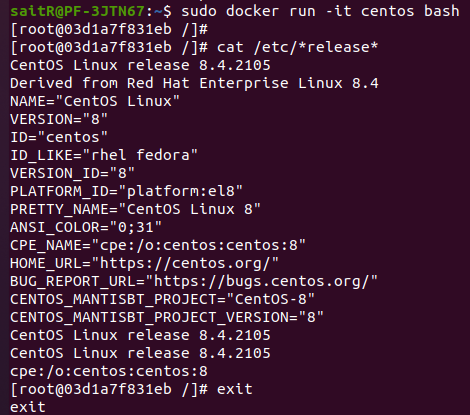
docker attach

DEMO:

sudo docker run -it centos bash

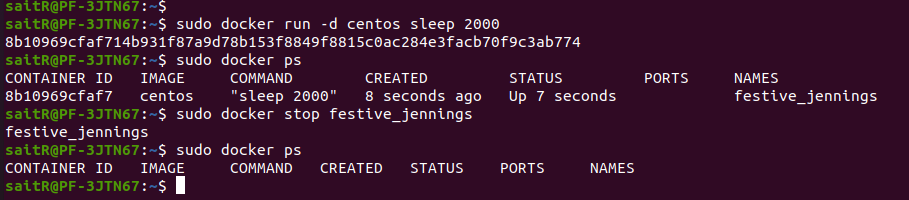
cat /*etc/*\*release\*

exit



**docker sleep :**



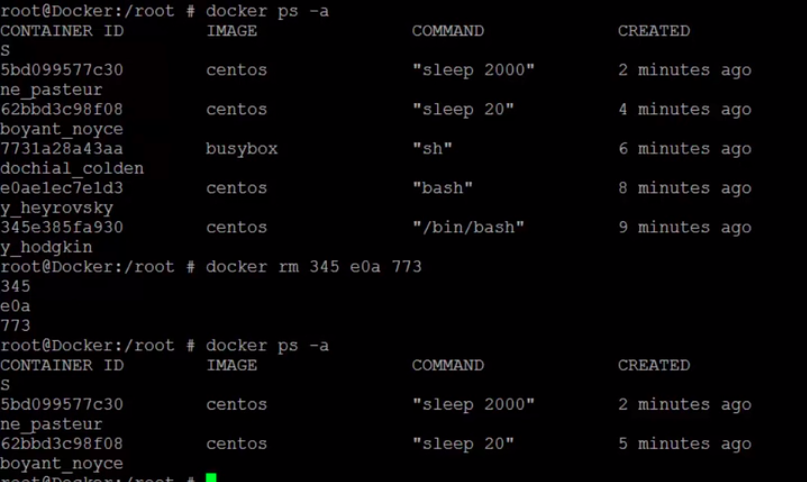


**remove container:**

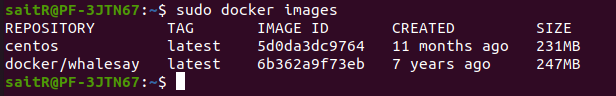
remove by container Name:

remove by container ID:

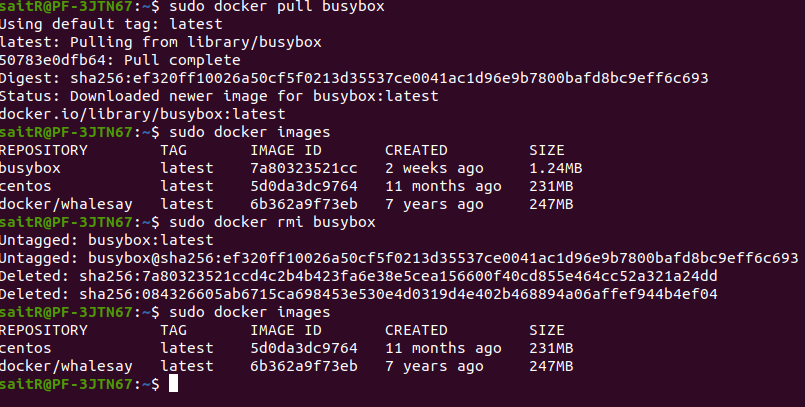




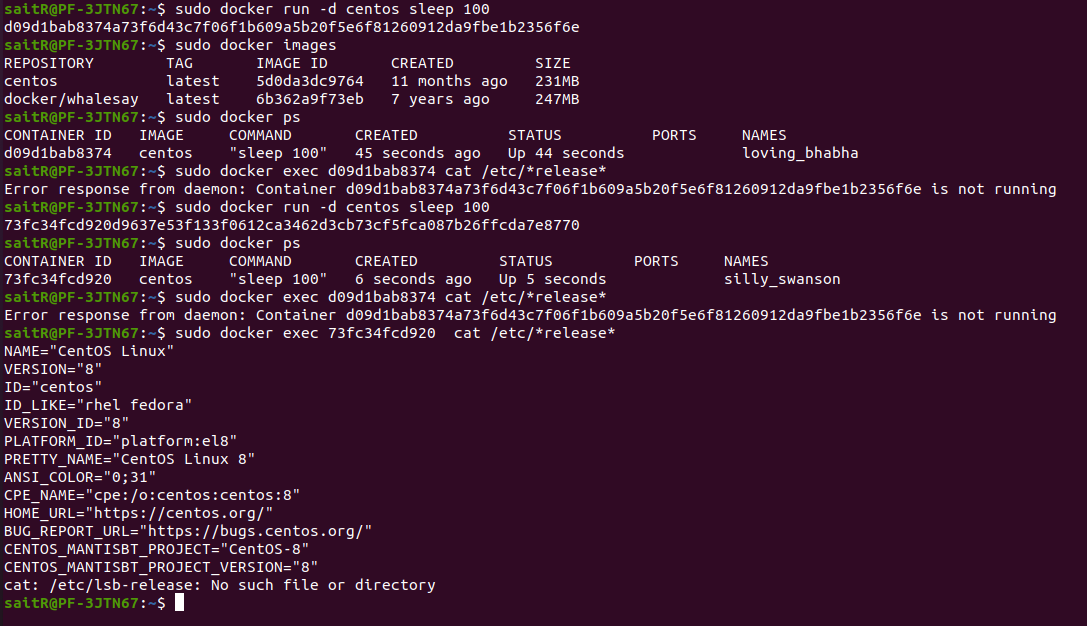
**Docker Images:**



**remove docker images:**



**docker Exec: exec command execute on a running container.**

****

**To delete all containers and images:**

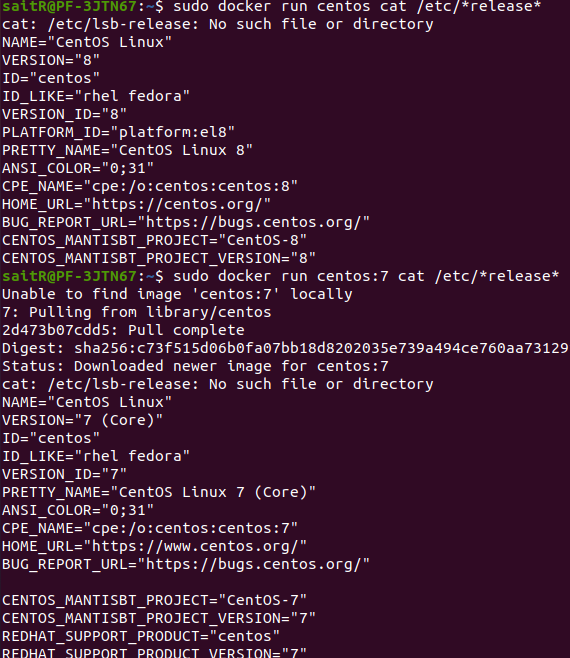
docker rmi <IMAGE:TAG>

Stop and delete all the containers being used by images.  
Then run the command to delete all the available images: docker rmi $(docker images -aq)

**run tag:**

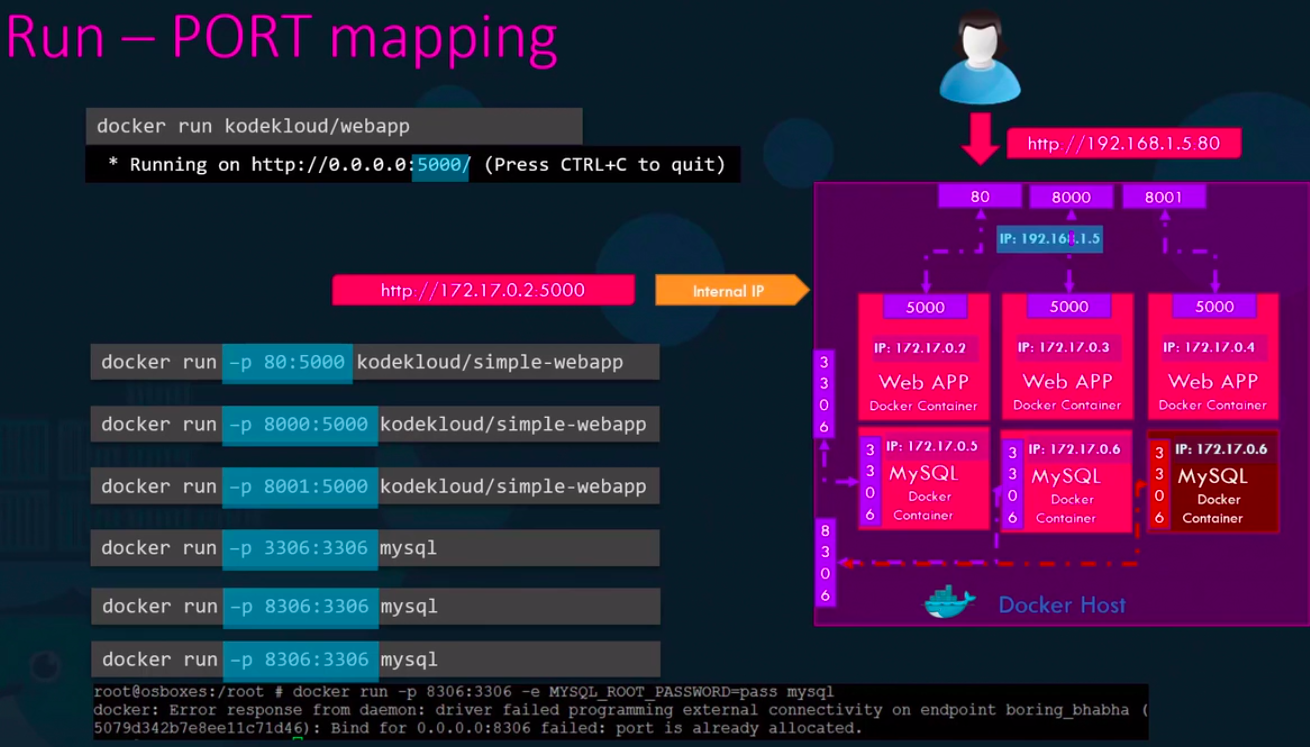
docker run redis => pulls the latest version of the redis.

Docker run redis:4.0.4 => pulls the version mentioned

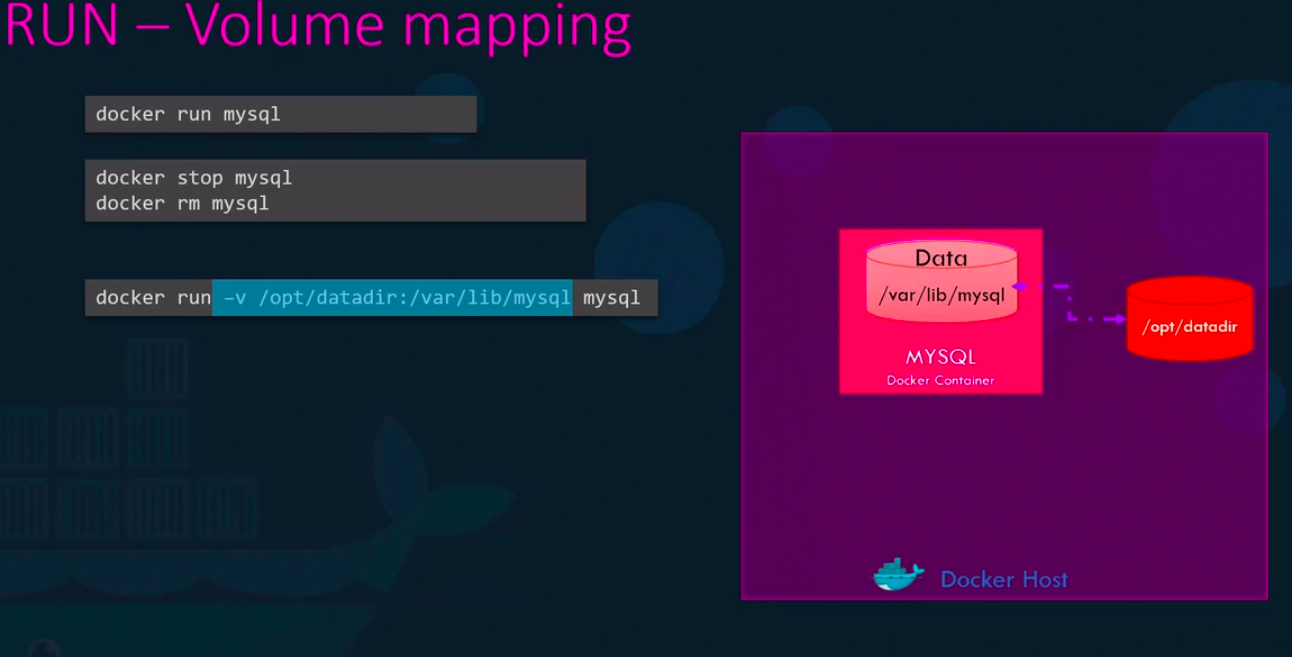


Run – STDIN:

**Run -PORT mapping:**



Run Volume Mapping:



**Inspect Container:**

docker inspect Container\_Name:

**Container Logs:**

docker logs container\_name

**Docker Images:**

**$ docker build -t webapp-color => create a docker image.**

**commands:**

apt-get update

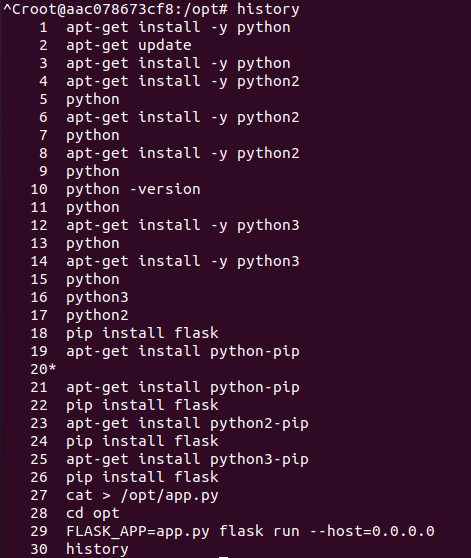
apt-get install -y python3

apt-get install python3-pip

pip install flask

create a copy application code to */opt/app.py*

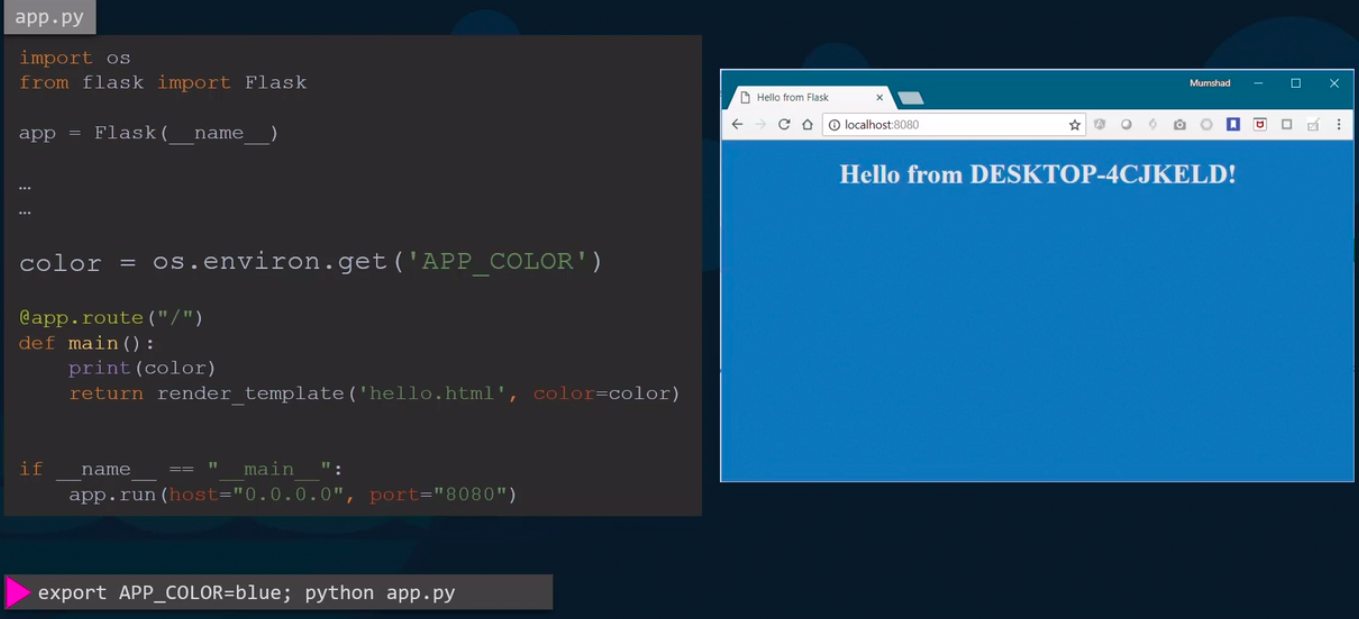
FLASK\_APP=/opt/app.py flask run --host=0.0.0.0

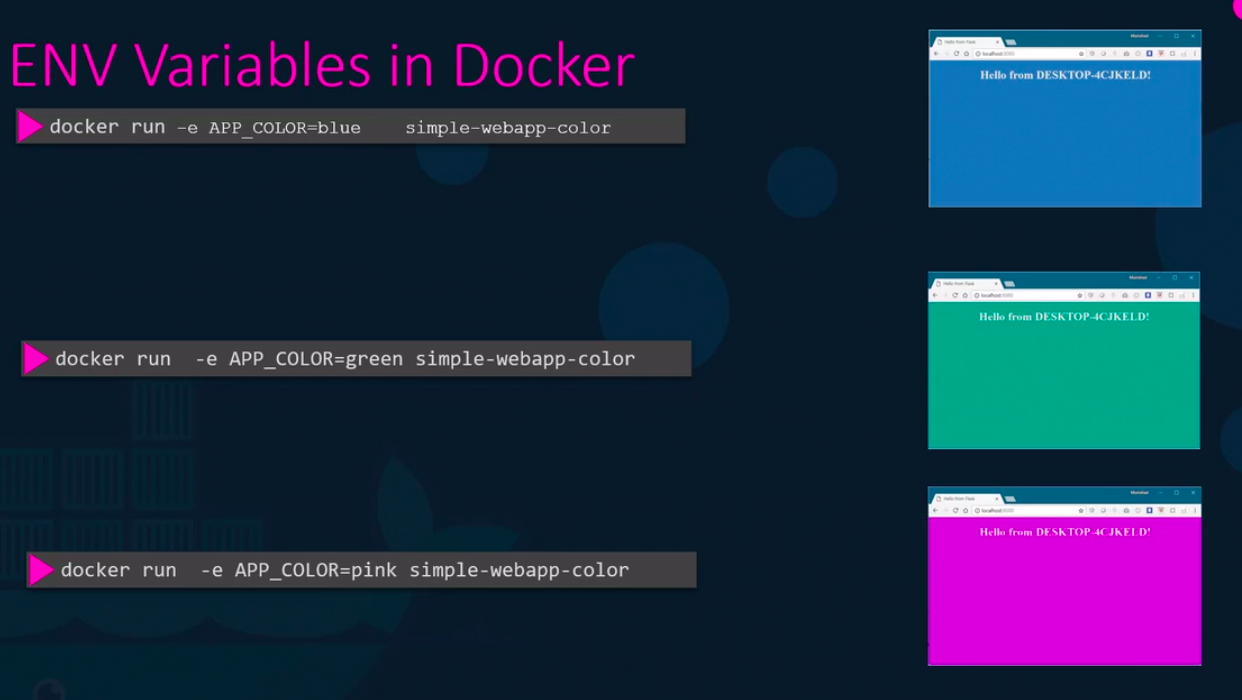


**Environment variables:**

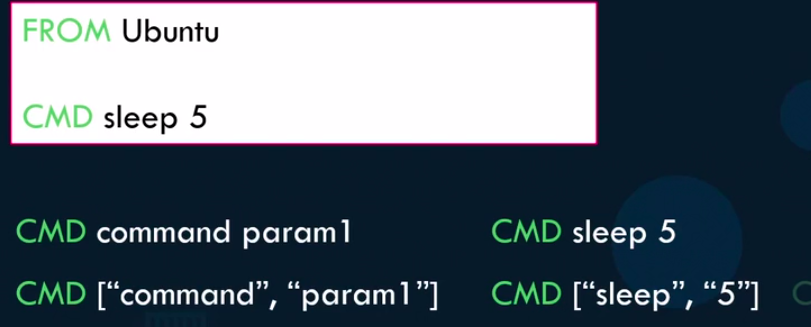
in the dockerfile we can set the env variables,

color =os.environ.get(‘App-COLOR’);

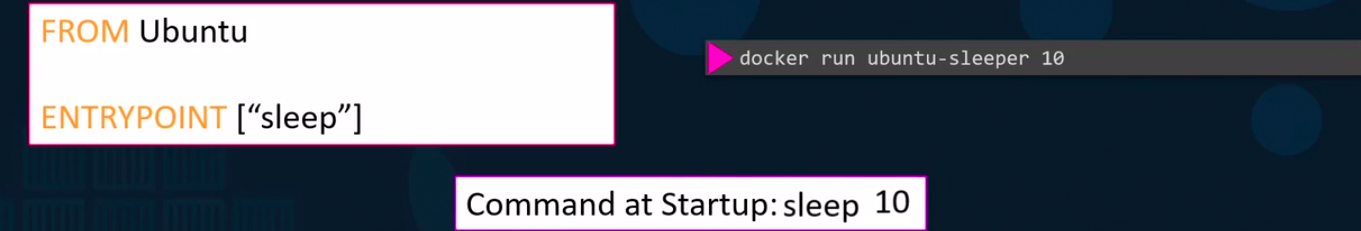








docker run ubuntu sleep 10



FROM Ubuntu

ENTRYPOINT [“sleep”]

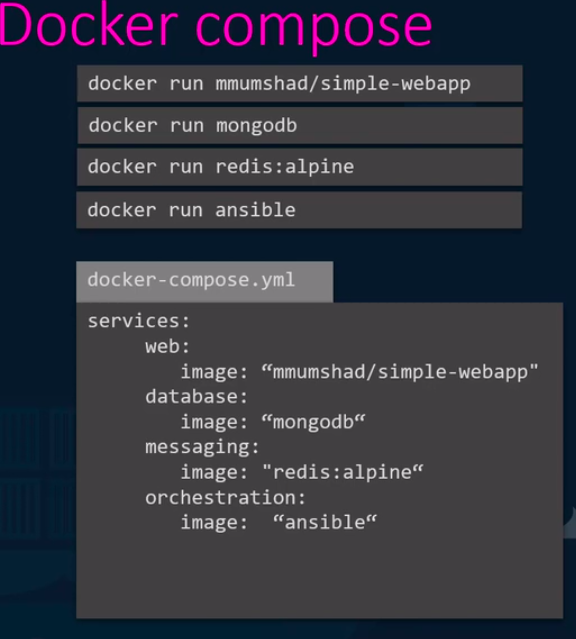
CMD[“5”]

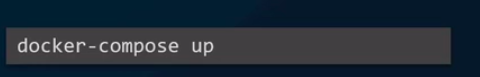
docker run ubuntu-container 15

docker run –entrypoint sleep2.0 ubuntu-container 10

**Docker compose :**

Compose is a tool for defining and running multi-container Docker applications. With Compose, you use a YAML file to configure your application’s services. Then, with a single command, you create and start all the services from your configuration.





docker-compose up => to bring up the application stack.

