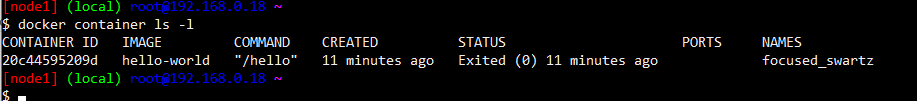
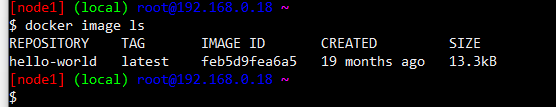
1. Run hello-world docker container and observe the container status.

Command ” docker container run hello-world”



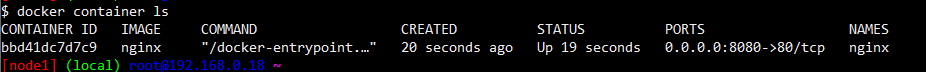
1. Check the docker images and also write down the size of hello-world image.

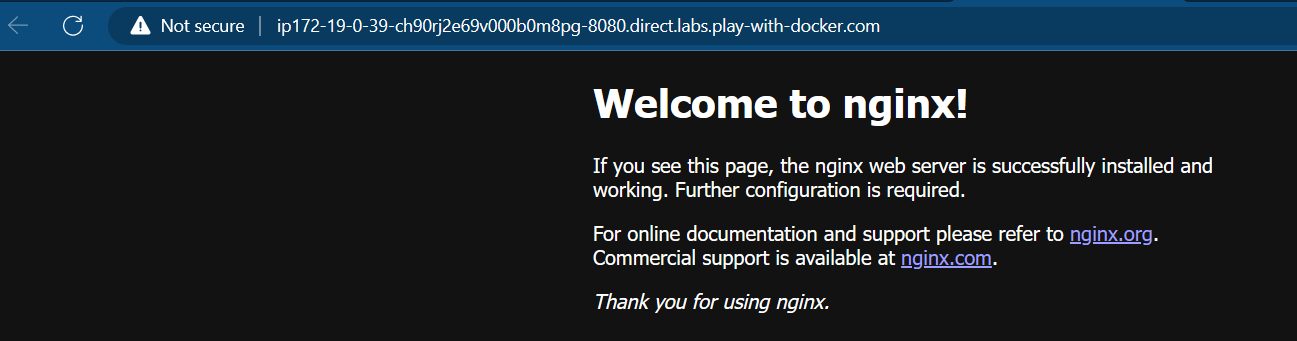
Command “ docker image ls “



1. Run the nginx container with name as nginx1 and expose it on 8080 port on docker host.

Command “ docker container run –-name nginx1 –d –p 8080:80 nginx “



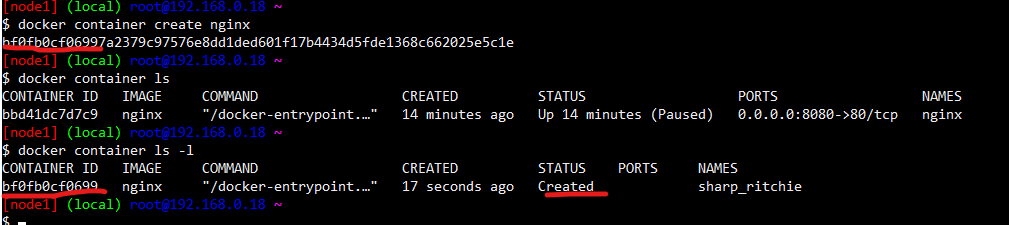


4. Explain docker container lifecycle

**Docker container lifecycle**

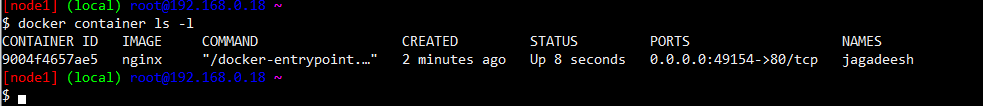
* Docker lifecycle states
  + Created

Command “ docker container create –name jagadeesh nginx “



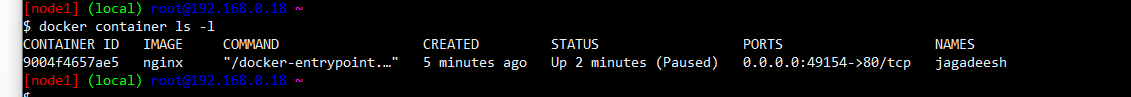
* + Running

Command “ docker container start jagadeesh “



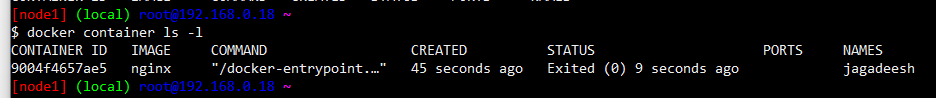
* + Paused

Command “ docker container pause jagadeesh “



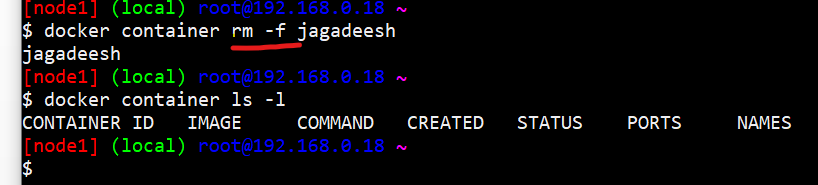
* + Stopped

Command “ docker container stop jagadeesh “



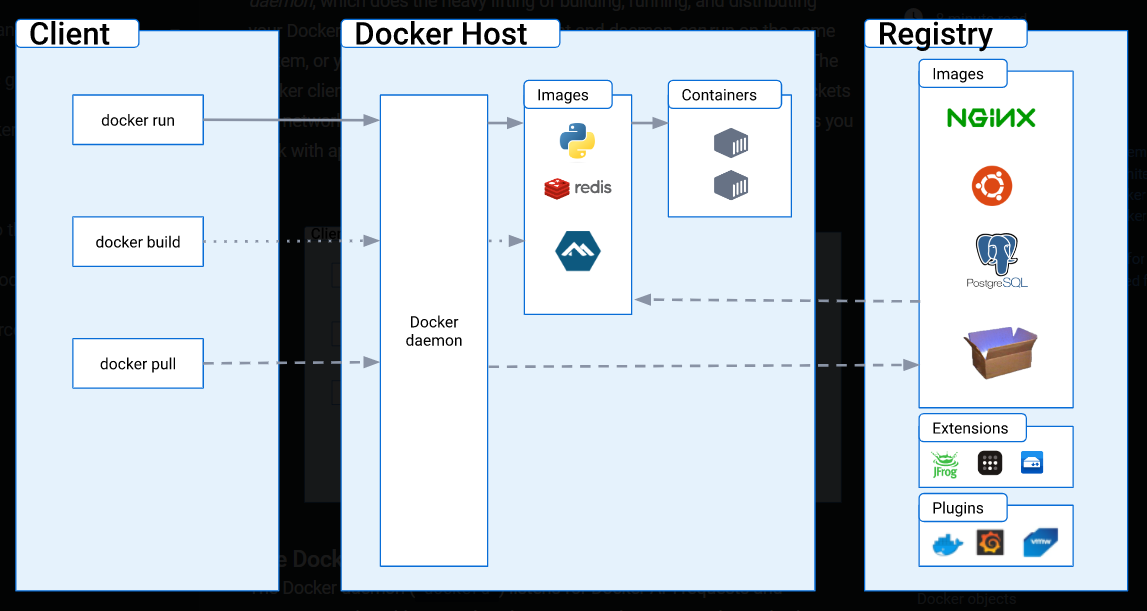
* + Deleted

Command “ docker container rm –f jagadeesh “



5. Explain what happens when you run the docker container.

* docker client will forward the request to docker daemon
* docker daemon will check if the image exists locally. if yes creates the container by using image
* if the image doesnot exist, then docker daemon tries to download the image from docker registry connected. The default docker registry is docker hub.
* Downloading image into local repo from registy is called as pull.
* Once the image is pulled the container is created.



6. Explain the Docker Architecture

Docker uses a client-server architecture. The Docker client talks to the Docker daemon, which does the heavy lifting of building, running, and distributing your Docker containers. The Docker client and daemon can run on the same system, or you can connect a Docker client to a remote Docker daemon. The Docker client and daemon communicate using a REST API, over UNIX sockets or a network interface. Another Docker client is Docker Compose, that lets you work with applications consisting of a set of containers.