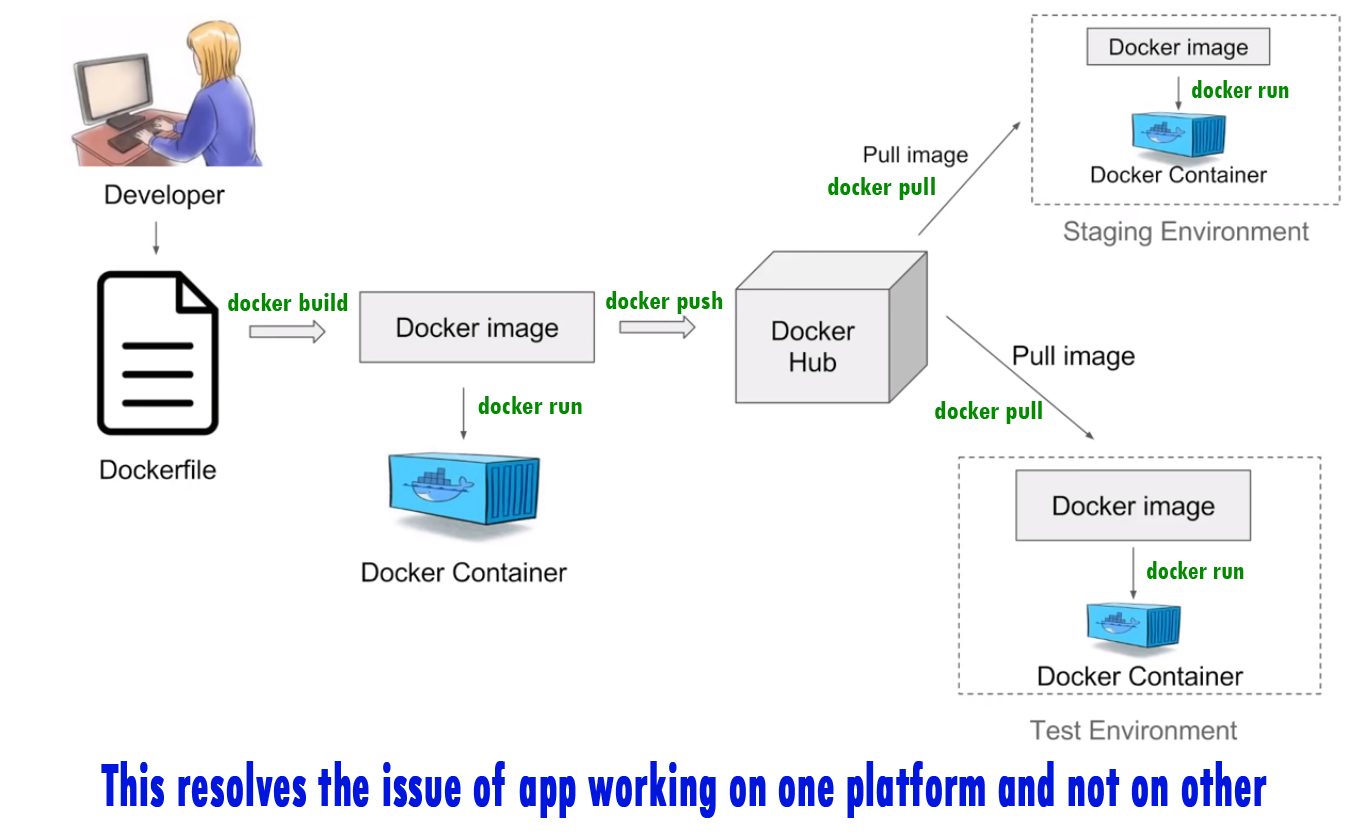
**Docker**



What is Docker?

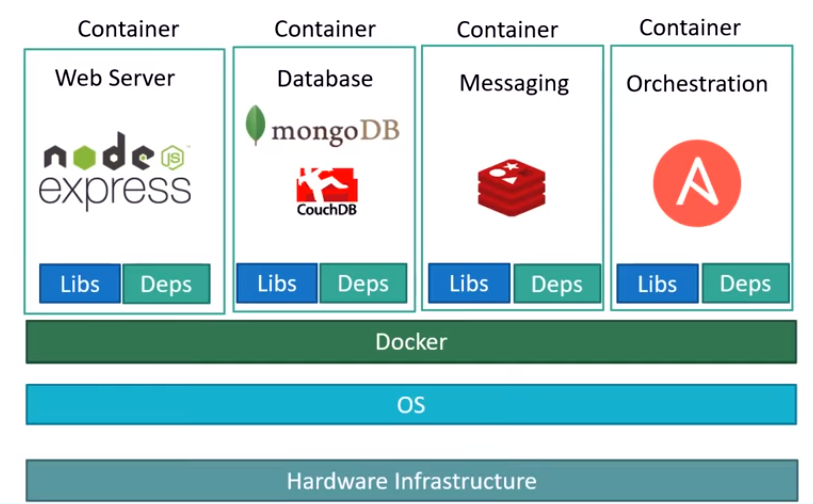
Docker is a container platform of software. This is a tool that designed to make it easier to deploy and run applications by using containers. The container allows a developer to package up an application with its dependencies, and environment. Finally the Docker ships the containers to all the possible platforms in a standard way.

For example:

A developer will package up an application with its components into a box called container and the docker will take care of shipping these containers to all the possible platforms (linux, mac, windows).

So, this resolves the issue of the application working on one platform and not on the other.

Docker Container:



Docker container is a runtime/running instance of a Docker image. Containers are isolated and have own environments, a set of processes/services, network interfaces, mounts etc. and share the same OS kernel.

Docker Image:

Docker images are templates used to create Docker containers. A container is a running instance of an image. A single image can also be used to create multiple containers. The images are stored in the docker hub/registry as well as local machine.

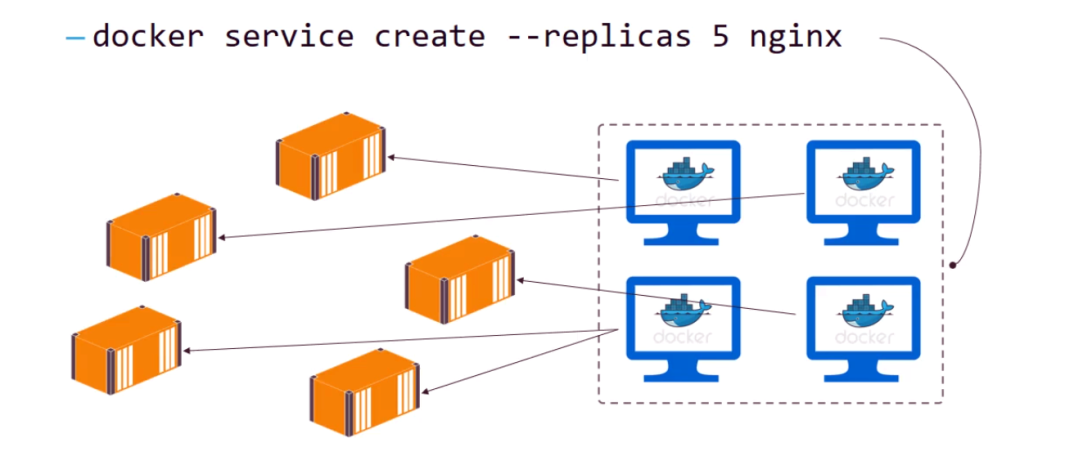
Docker file:

A text file with instructions to build a Docker image.

What is Cluster?

A cluster is a group of services and other resources that acts like a single system, and enables high availability, load balancing as well as parallel processing.

Docker Service:



Docker run will start a single/standalone instance/container of a single image in a single machine/node. So the image will run through one container.

Whereas, a service defines one or more instances/containers of a single image and deployed on one or more machines/nodes for high availability, load balancing and parallel processing. So the image will run or response fast. With the Docker service, you can easily manage a group of containers of the same image. You can scale them up/down or update via a container cluster/orchestration tool (Docker Swarm, Kubernetes). These containers could be deployed to different nodes, and all the containers will run in a distributed environment system. So, Services are actually just containers in production of multiple nodes (acts as single node using a cluster tool).

**docker service create** is used to create instances (called tasks) of that service running in a cluster (called Swarm, Kubernetes) of computers (called nodes). Those tasks are containers of course, but not standalone containers. So the containers are manageable.

For example:

**docker service create --name SERVICE\_NAME --replicas 3 IMAGE:TAG**

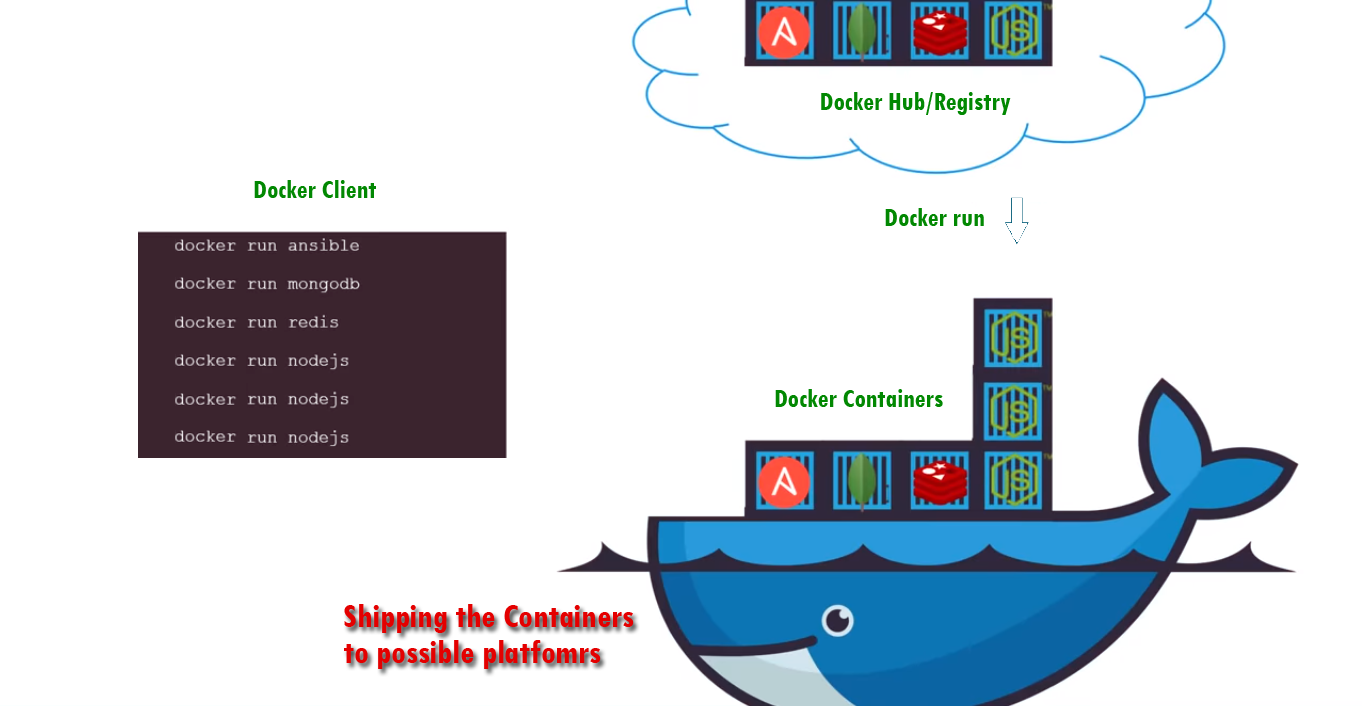
creates 3 tasks of the SERVICE\_NAME service, which is based on the IMAGE:TAG image.

Ref: <https://docs.docker.com/engine/swarm/how-swarm-mode-works/services/>

For a larger application, we often need a microservice system. Normally, services might include an HTTP server, a database server, or any other type of executable program that you wish to run in a distributed environment. In this case, Docker service is multiple hosts solution whereas Docker run is single host solution.

Advantages:

1. The port where the cluster (Swarm) will make the service available outside the cluster
2. An overlay network for the service to connect to other services in the cluster
3. CPU and memory limits and reservations
4. The number of replicas/tasks of the image to run in the cluster

Docker run:

docker run hello-world

It will run the hello-world image if not found locally it will download from docker hub and run it.

docker pull image

This will image from docker hub and run it