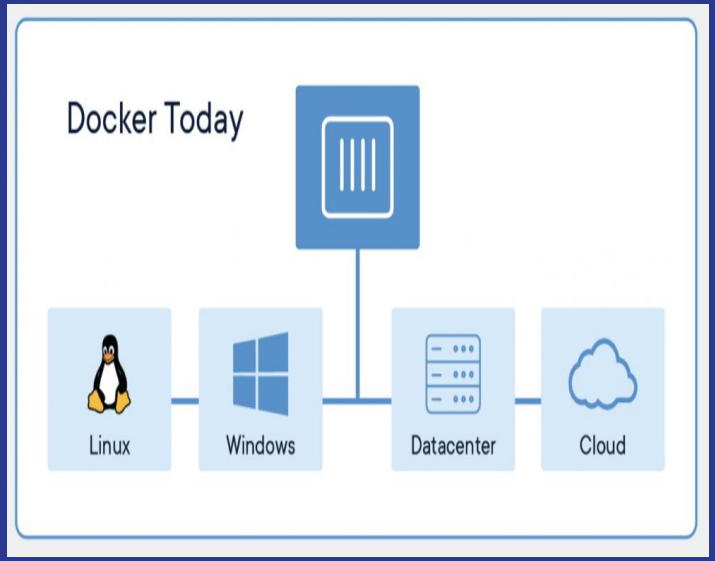


Docker Containers

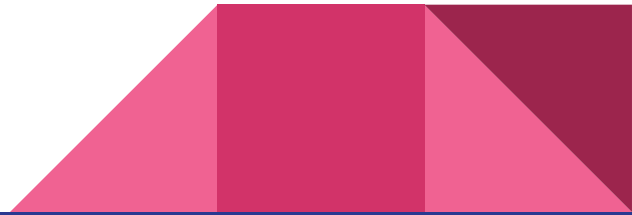
Created & Presented By: Janvi

What is Docker ?

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.

Docker provides tooling and a platform to manage the lifecycle of your containers:

- Develop your application and its supporting components using containers.
- The container becomes the unit for distributing and testing your application.
- When you’re ready, deploy your application into your production environment, as a container or an orchestrated service. This works the same whether your production environment is a local data center, a cloud provider, or a hybrid of the two.



Docker Container

Docker provides the ability to package and run an application in a loosely isolated environment called a container. The isolation and security allows you to run many containers simultaneously on a given host.

A container is a standard unit of software that packages up code and all its dependencies so the application runs quickly and reliably from one computing environment to another. A Docker container image is a lightweight, standalone, executable package of software that includes everything needed to run an application: code, runtime, system tools, system libraries and settings.

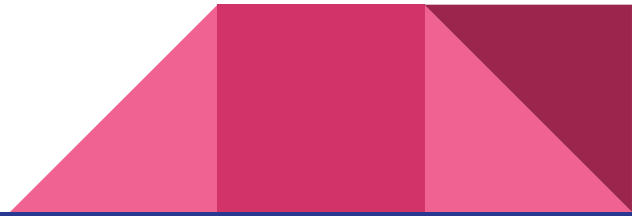
Container images become containers at runtime and in the case of Docker containers – images become containers when they run on Docker Engine.

Containers isolate software from its environment and ensure that it works uniformly despite differences for instance between development and staging.

Benefits Of Docker Container :

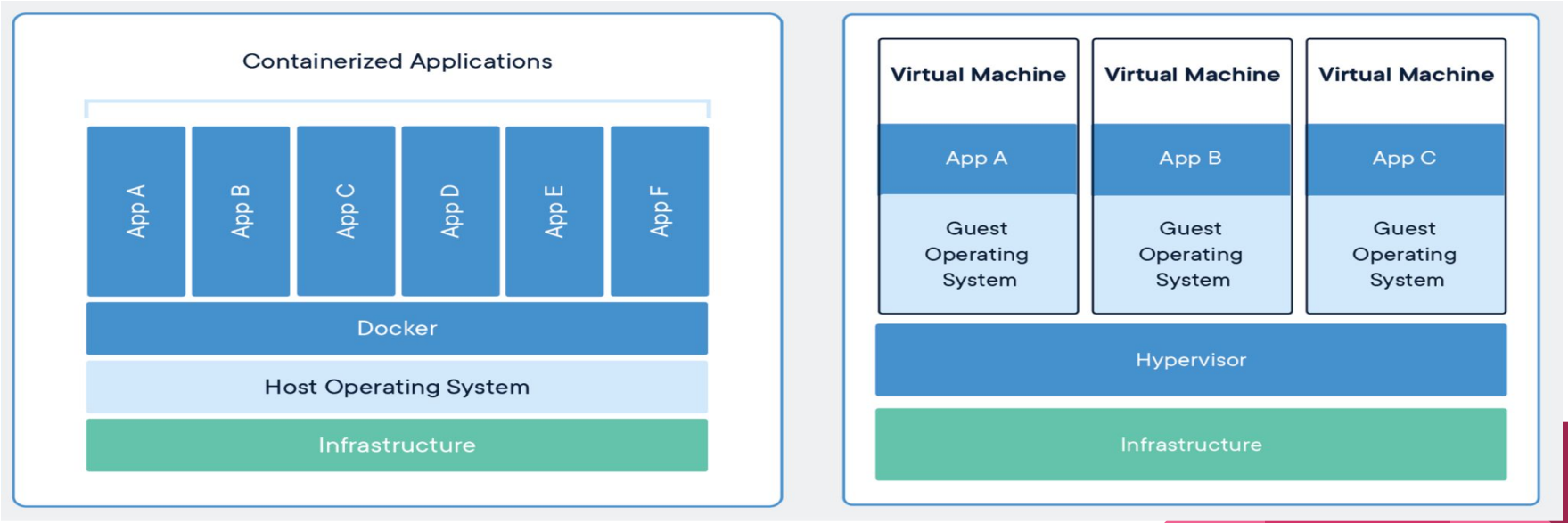
The benefits of Docker in building and deploying applications are many:

- Caching a cluster of containers
- Flexible resource sharing
- Scalability - many containers can be placed in a single host
- Running your service on hardware that is much cheaper than standard servers
- Fast deployment, ease of creating new instances, and faster migrations.
- Ease of moving and maintaining your applications
- Better security, less access needed to work with the code running inside containers, and fewer software dependencies
- Ease in debugging.



Docker Container v/s Virtual Machines

Docker container can be thought of as a virtual machines but in reality its not. Containers and virtual machines have similar resource isolation and allocation benefits, but function differently because containers virtualize the operating system instead of hardware. Containers are more portable and efficient.



Docker Container v/s Virtual Machines

Containers:

Containers are an abstraction at the app layer that packages code and dependencies together. Multiple containers can run on the same machine and share the OS kernel with other containers, each running as isolated processes in user space. Containers take up less space than VMs (container images are typically tens of MBs in size), can handle more applications and require fewer VMs and Operating systems.

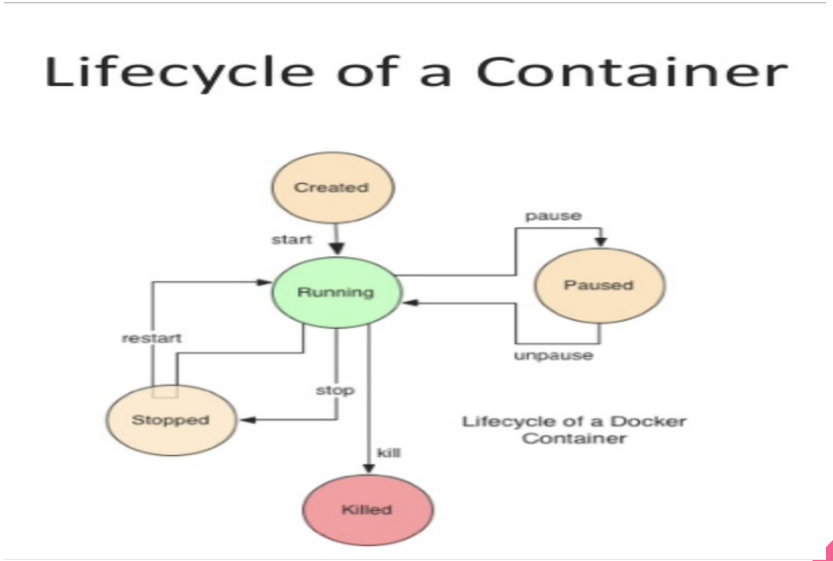
Virtual Machines:

Virtual machines (VMs) are an abstraction of physical hardware turning one server into many servers. The hypervisor allows multiple VMs to run on a single machine. Each VM includes a full copy of an operating system, the application, necessary binaries and libraries – taking up tens of GBs. VMs can also be slow to boot.

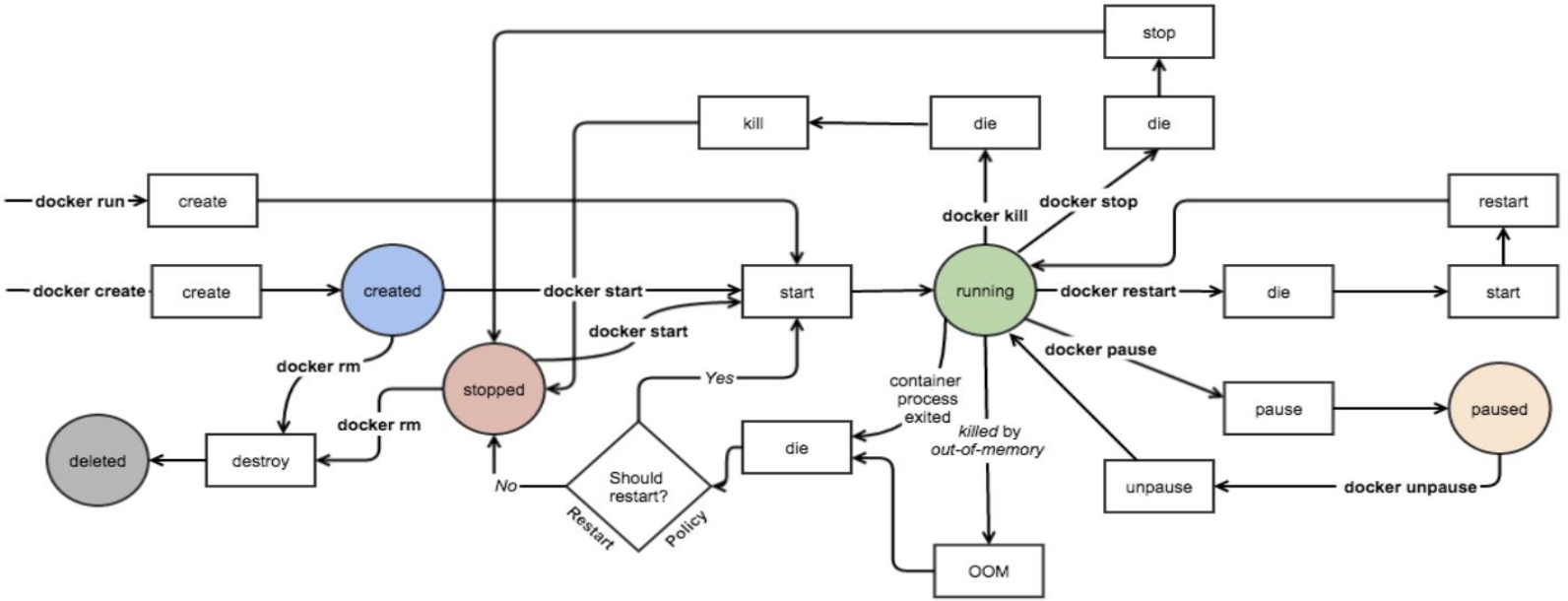
Containers and VMs used together provide a great deal of flexibility in deploying and managing app

Docker containers that run on Docker Engine:

- **Standard:** Docker created the industry standard for containers, so they could be portable anywhere
- **Lightweight:** Containers share the machine's OS system kernel and therefore do not require an OS per application, driving higher server efficiencies and reducing server and licensing costs
- **Secure:** Applications are safer in containers and Docker provides the strongest default isolation capabilities in the industry



Docker Lifecycle



Docker Container commands

Create container $ docker create –name ubuntu-cont ubuntu

Run docker container

- $ docker run -itd ubuntu
- $ docker run -itd –name ubuntu-cont ubuntu

Pause container $ docker pause <container-id/name>

Unpause container $ docker unpause <container-id/name>

Start container $ docker start <container-id/name>

Stop container $ docker stop <container-id/name>

Restart container $ docker restart <container-id/name>

Kill container $ docker kill <container-id/name>

Destroy container $ docker rm <container-id/name>