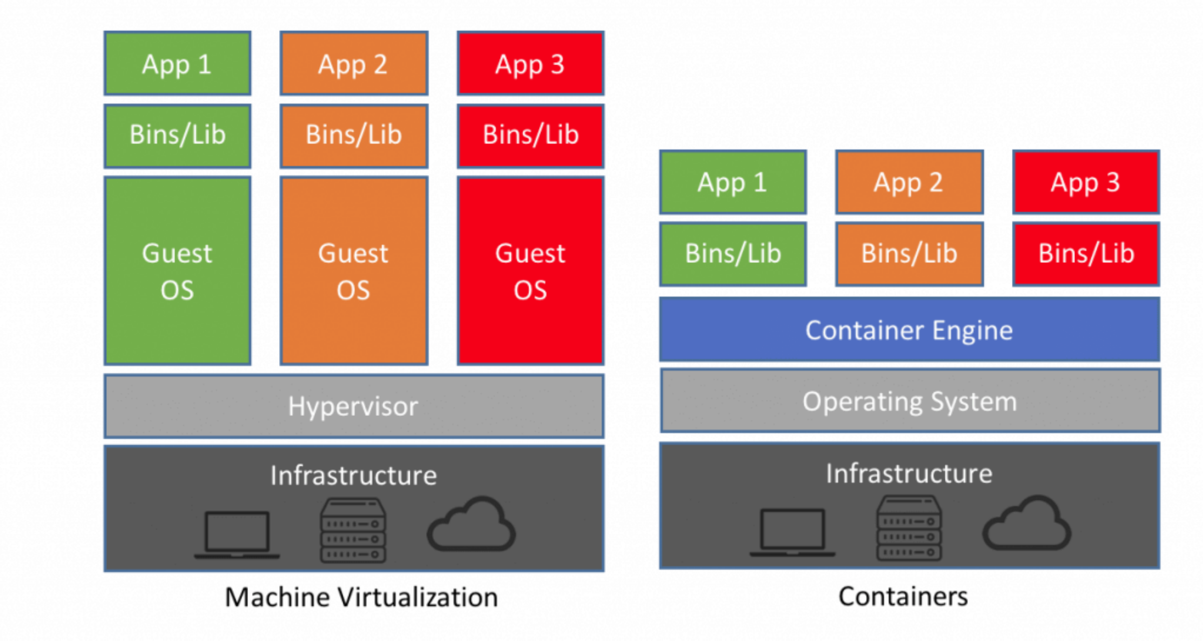
What is Virtualization?

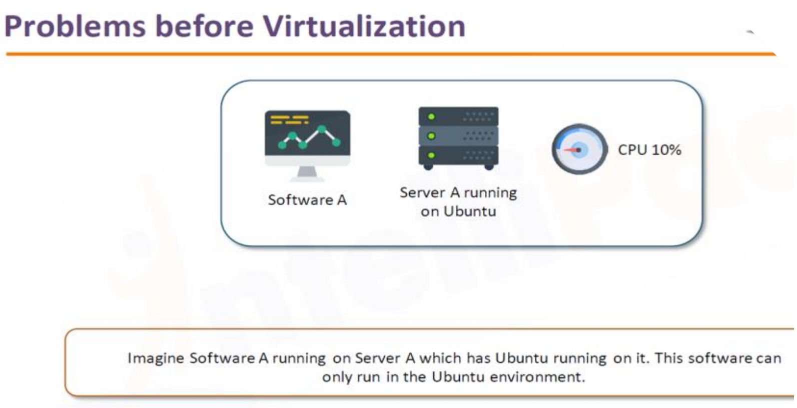
Virtualization is the process of running multiple virtual systems or resources on top of a single physical machine. These resources could be a storage device, network or even an operating system.



Earlier, the process for deploying a service was slow and painful. First, the developers were writing code; then the operations team would deploy it on bare metal machines, where they had to look out for library versions, patches, and language compilers for the code to work. If there were some bugs or errors, the process would start all over again, the developers would fix it, and then again the operational team was there to deploy.

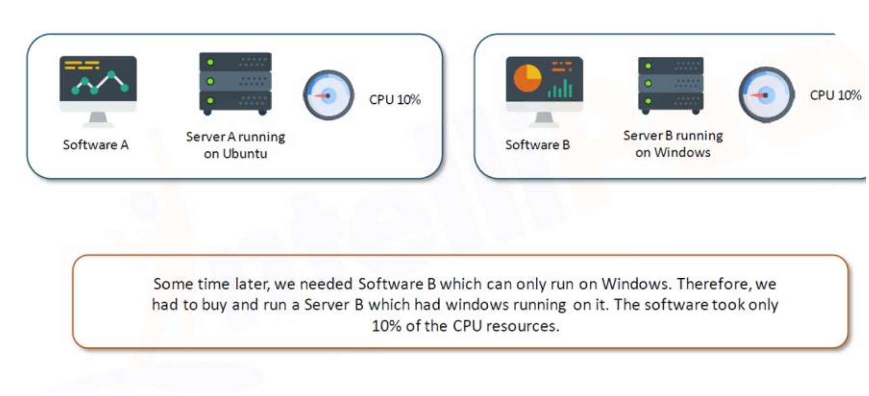
There was an improvement with the creation of Hypervisors. Hypervisors have multiple Virtual machines or VMs on the same host, which may be running or turned off. VMs decreased the waiting time for deploying code and bug fixing in a big manner, but the real game changer was Docker containers.

Problems before Virtualization

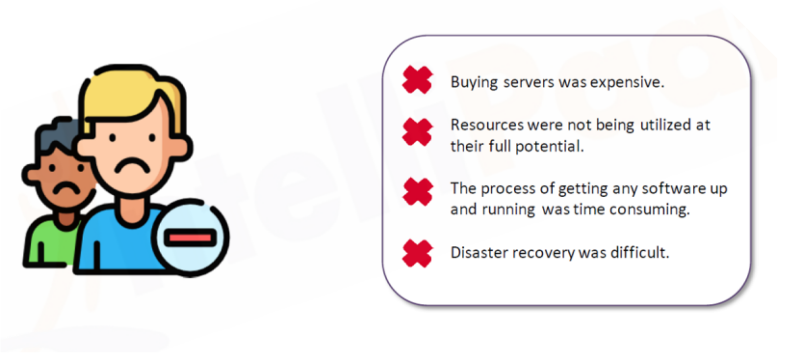


Problems before Virtualization

Imagine Software A running on Server A which has Ubuntu running on it. This software can only run in the Ubuntu environment.

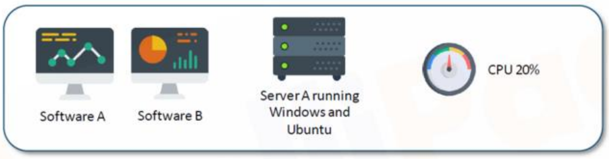


Some time later, we needed Software B which can only run on Windows. Therefore, we had to buy and run a Server B which had windows running on it. The software took only 10% of the CPU resources.



- ✖ Buying servers was expensive.
- ✖ Resources were not being utilized at their full potential.
- ✖ The process of getting any software up and running was time consuming.
- ✖ Disaster recovery was difficult.

After Virtualization



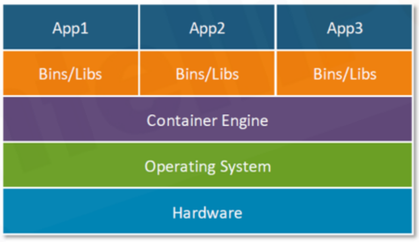
Windows and Ubuntu OS now are running on the same server in parallel using the Virtualization technology. This accounts for better CPU utilization and cost savings!

Advantages of Virtualization

- ✓ It results in reduced spending.
- ✓ Resources are utilized more efficiently.
- ✓ Process of getting software up and running is shorter.
- ✓ Easier backup and disaster recovery is available.

What is Containerization?

Application **containerization** is an OS-level virtualization method used to deploy and run distributed applications without launching an entire virtual machine (VM) for each app.



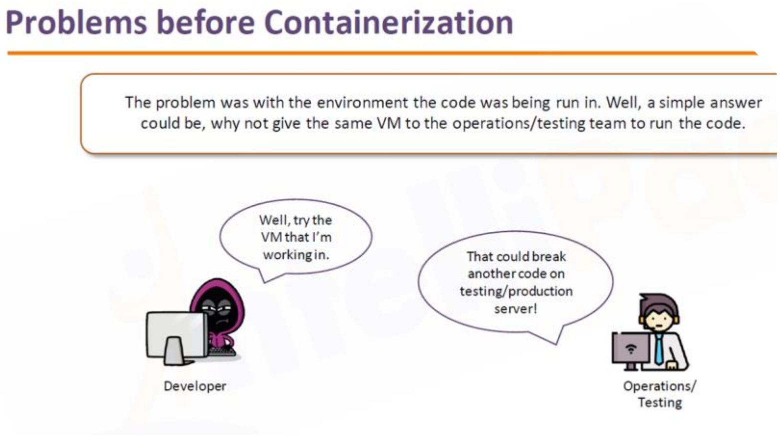
App1 | App2 | App3

Bins/Libs | Bins/Libs | Bins/Libs

Container Engine

Operating System

Hardware

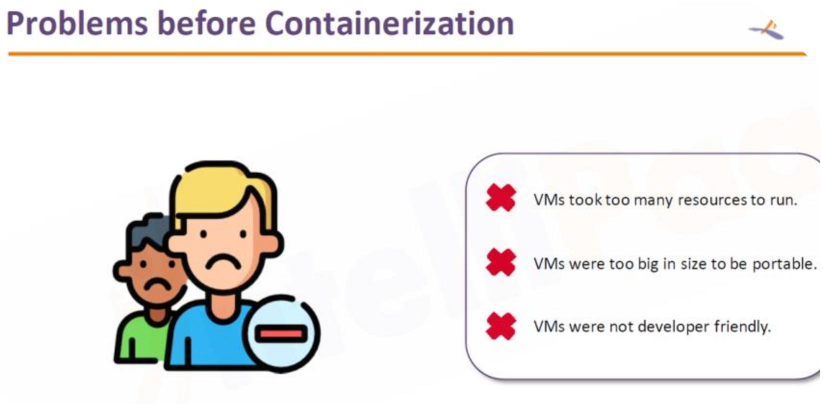


Problems before Containerization

The problem was with the environment the code was being run in. Well, a simple answer could be, why not give the same VM to the operations/testing team to run the code.

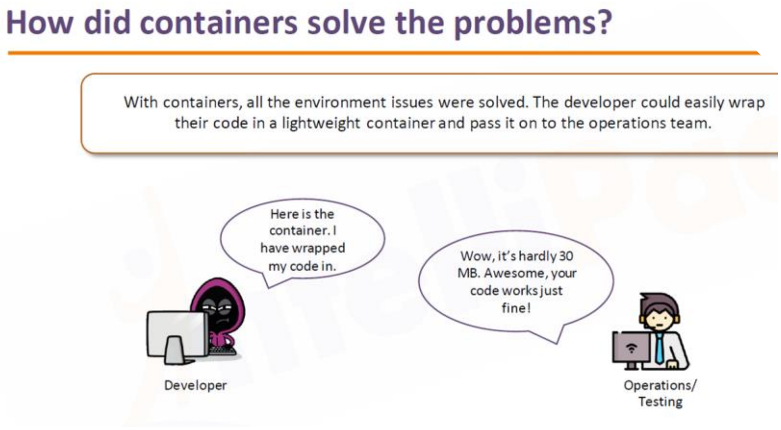
Well, try the VM that I'm working in.

That could break another code on testing/production server!



Problems before Containerization

- ✖ VMs took too many resources to run.
- ✖ VMs were too big in size to be portable.
- ✖ VMs were not developer friendly.

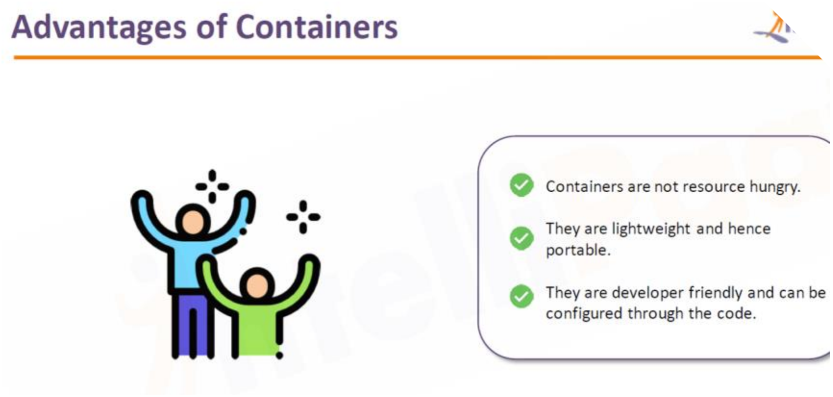


How did containers solve the problems?

With containers, all the environment issues were solved. The developer could easily wrap their code in a lightweight container and pass it on to the operations team.

Here is the container. I have wrapped my code in.

Wow, it's hardly 30 MB. Awesome, your code works just fine!

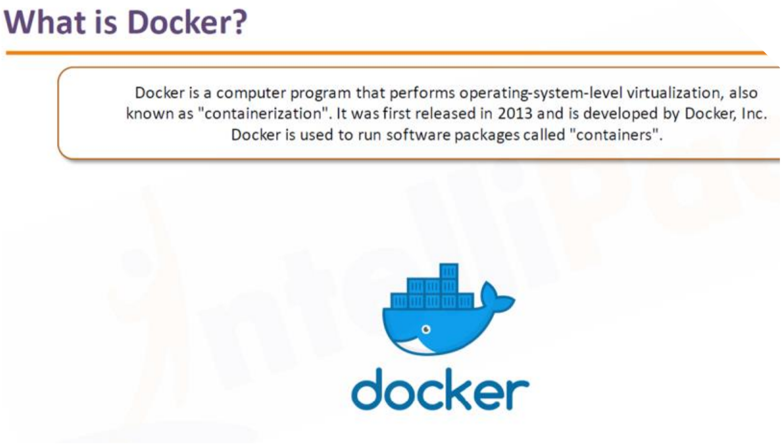


Advantages of Containers

- ✓ Containers are not resource hungry.
- ✓ They are lightweight and hence portable.
- ✓ They are developer friendly and can be configured through the code.

Containerization Tools

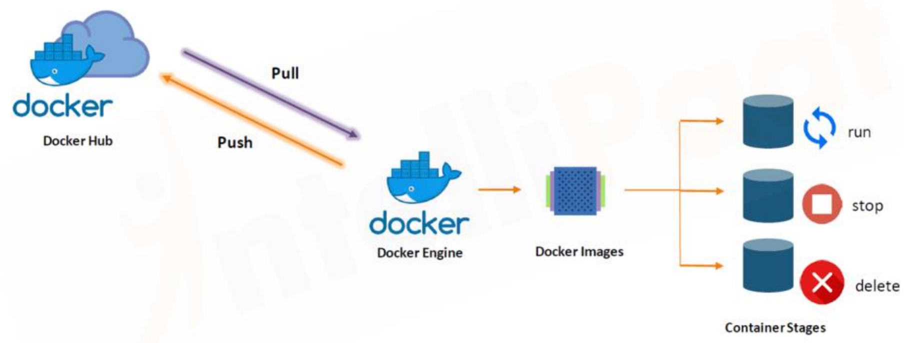
MESOS
rkt
docker



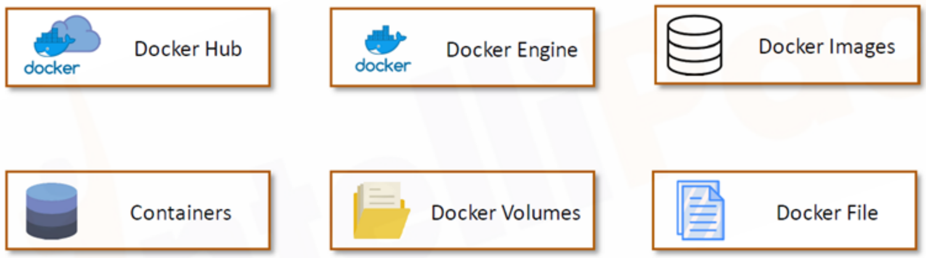
What is Docker?

Docker is a computer program that performs operating-system-level virtualization, also known as "containerization". It was first released in 2013 and is developed by Docker, Inc. Docker is used to run software packages called "containers".

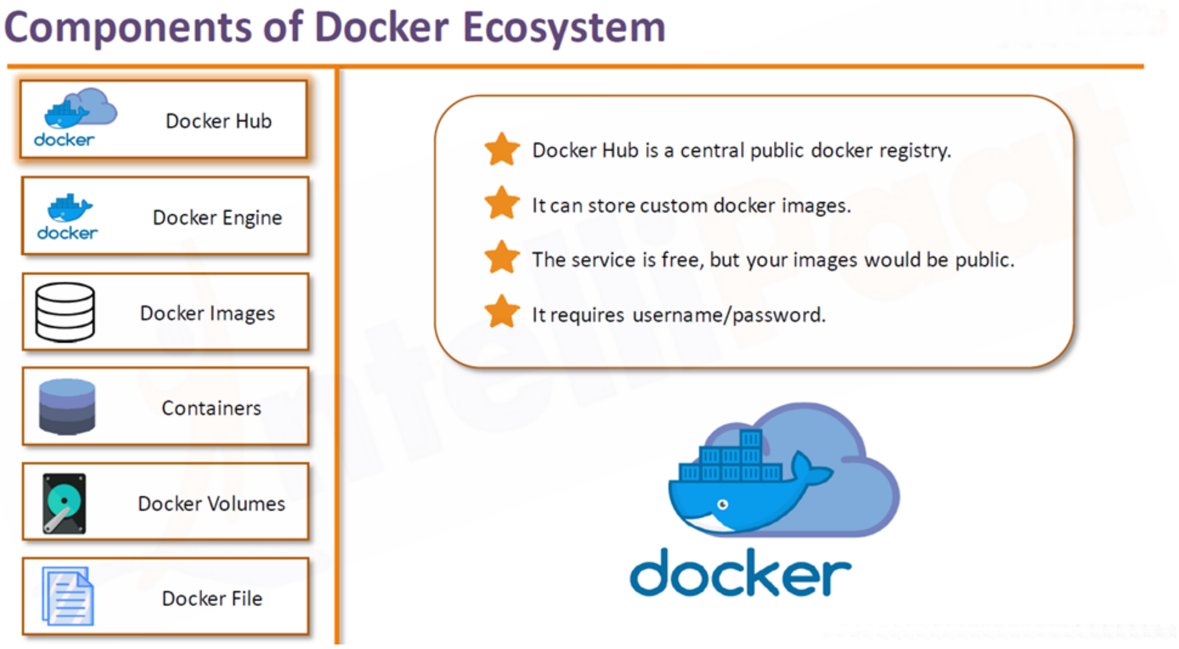
Docker Container Life Cycle



Components of Docker Ecosystem



- Docker Hub
- Docker Engine
- Docker Images
- Containers
- Docker Volumes
- Docker File



Components of Docker Ecosystem

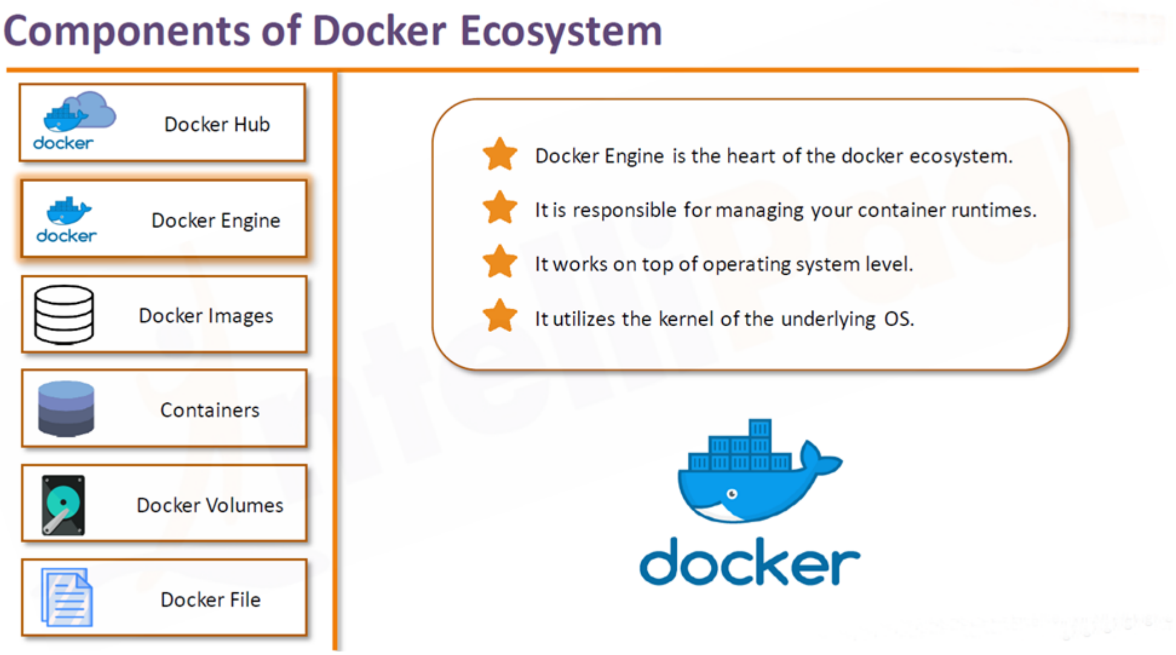
- Docker Hub
- Docker Engine
- Docker Images
- Containers
- Docker Volumes
- Docker File

★ Docker Hub is a central public docker registry.

★ It can store custom docker images.

★ The service is free, but your images would be public.

★ It requires username/password.



Components of Docker Ecosystem

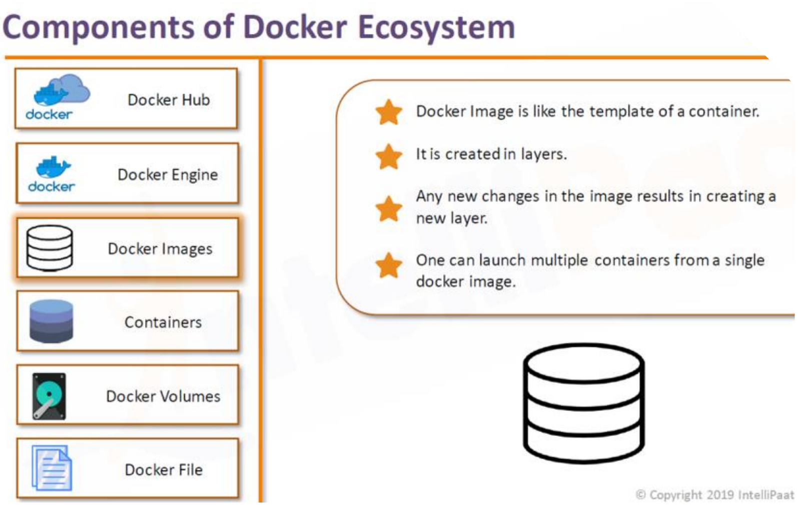
- Docker Hub
- Docker Engine
- Docker Images
- Containers
- Docker Volumes
- Docker File

★ Docker Engine is the heart of the docker ecosystem.

★ It is responsible for managing your container runtimes.

★ It works on top of operating system level.

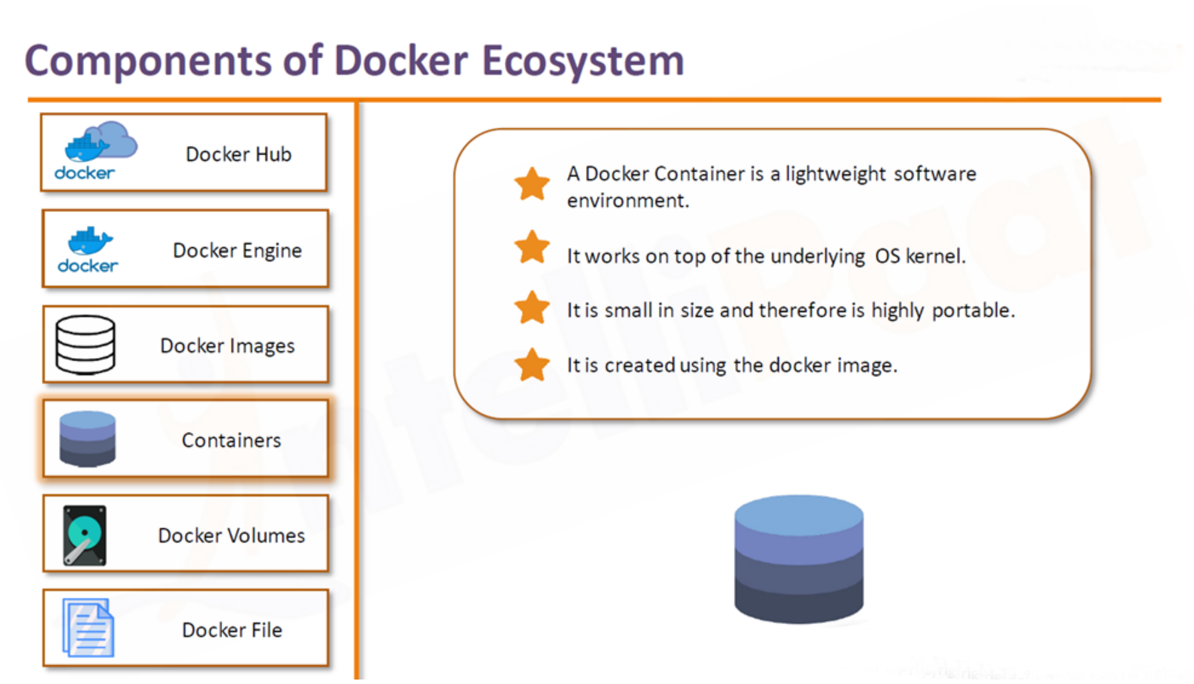
★ It utilizes the kernel of the underlying OS.



Components of Docker Ecosystem

- Docker Hub
- Docker Engine
- Docker Images
- Containers
- Docker Volumes
- Docker File

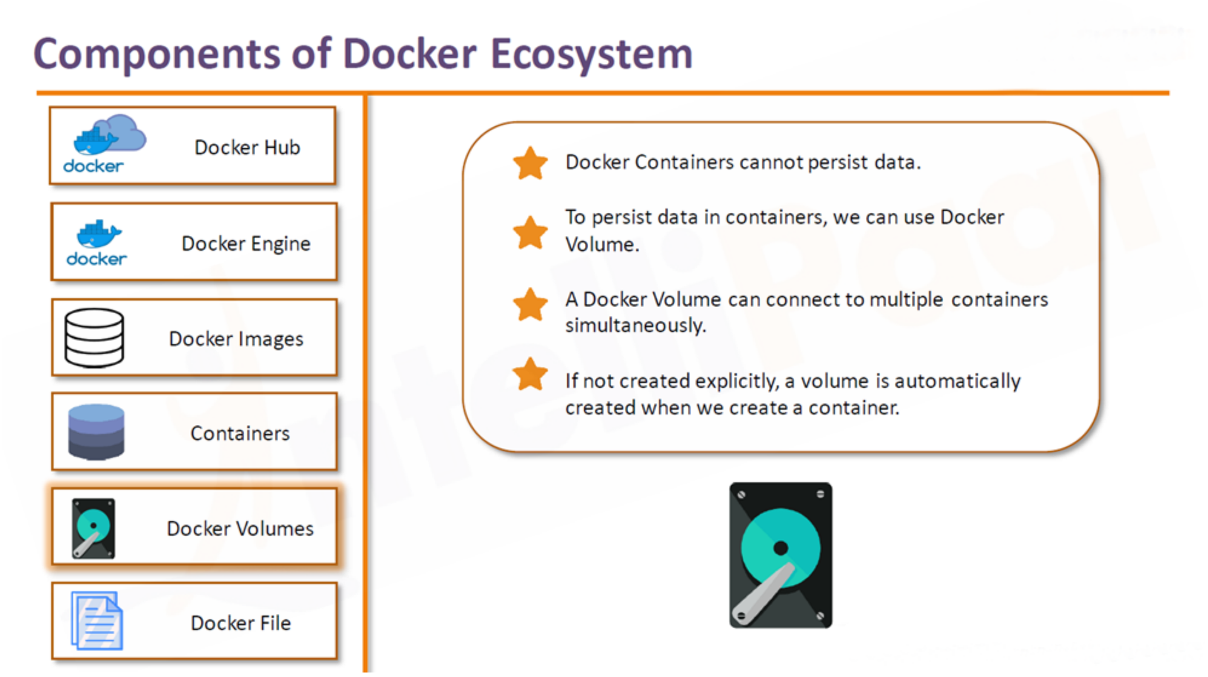
- Docker Image is like the template of a container.
- It is created in layers.
- Any new changes in the image results in creating a new layer.
- One can launch multiple containers from a single docker image.



Components of Docker Ecosystem

- Docker Hub
- Docker Engine
- Docker Images
- Containers
- Docker Volumes
- Docker File

- A Docker Container is a lightweight software environment.
- It works on top of the underlying OS kernel.
- It is small in size and therefore is highly portable.
- It is created using the docker image.

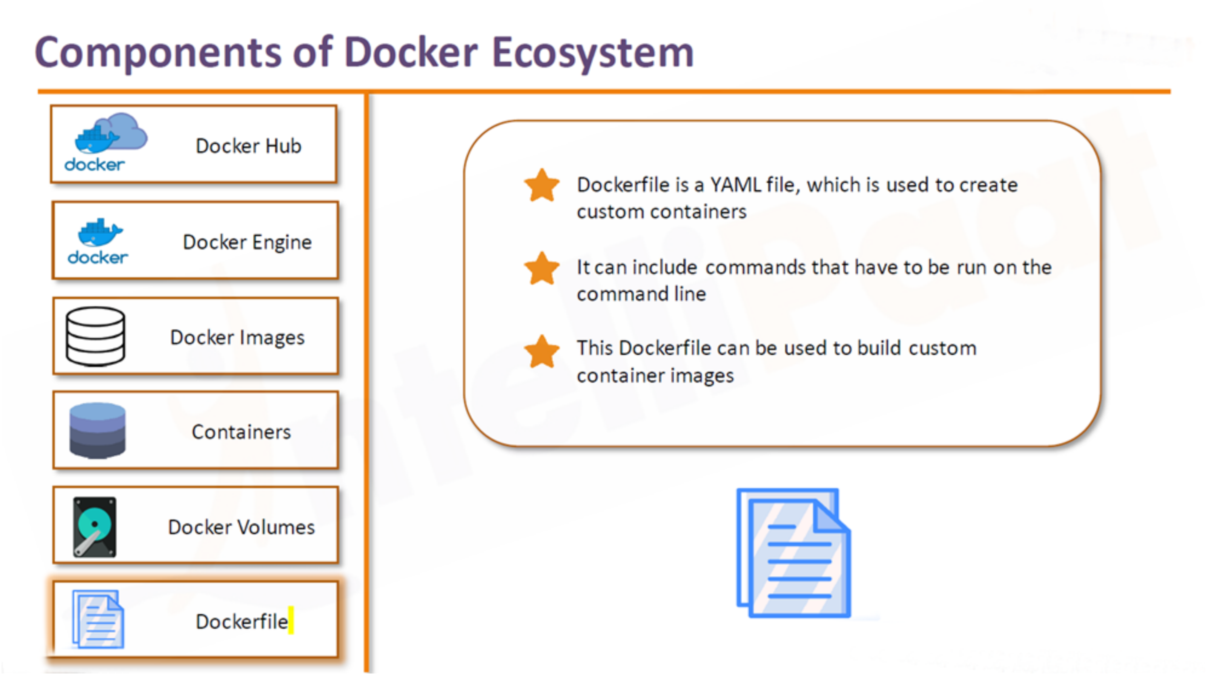


Components of Docker Ecosystem

- Docker Hub
- Docker Engine
- Docker Images
- Containers
- Docker Volumes
- Docker File

docker
docker

- Docker Containers cannot persist data.
- To persist data in containers, we can use Docker Volume.
- A Docker Volume can connect to multiple containers simultaneously.
- If not created explicitly, a volume is automatically created when we create a container.



Components of Docker Ecosystem

- Docker Hub
- Docker Engine
- Docker Images
- Containers
- Docker Volumes
- Dockerfile

docker
docker

- Dockerfile is a YAML file, which is used to create custom containers
- It can include commands that have to be run on the command line
- This Dockerfile can be used to build custom container images