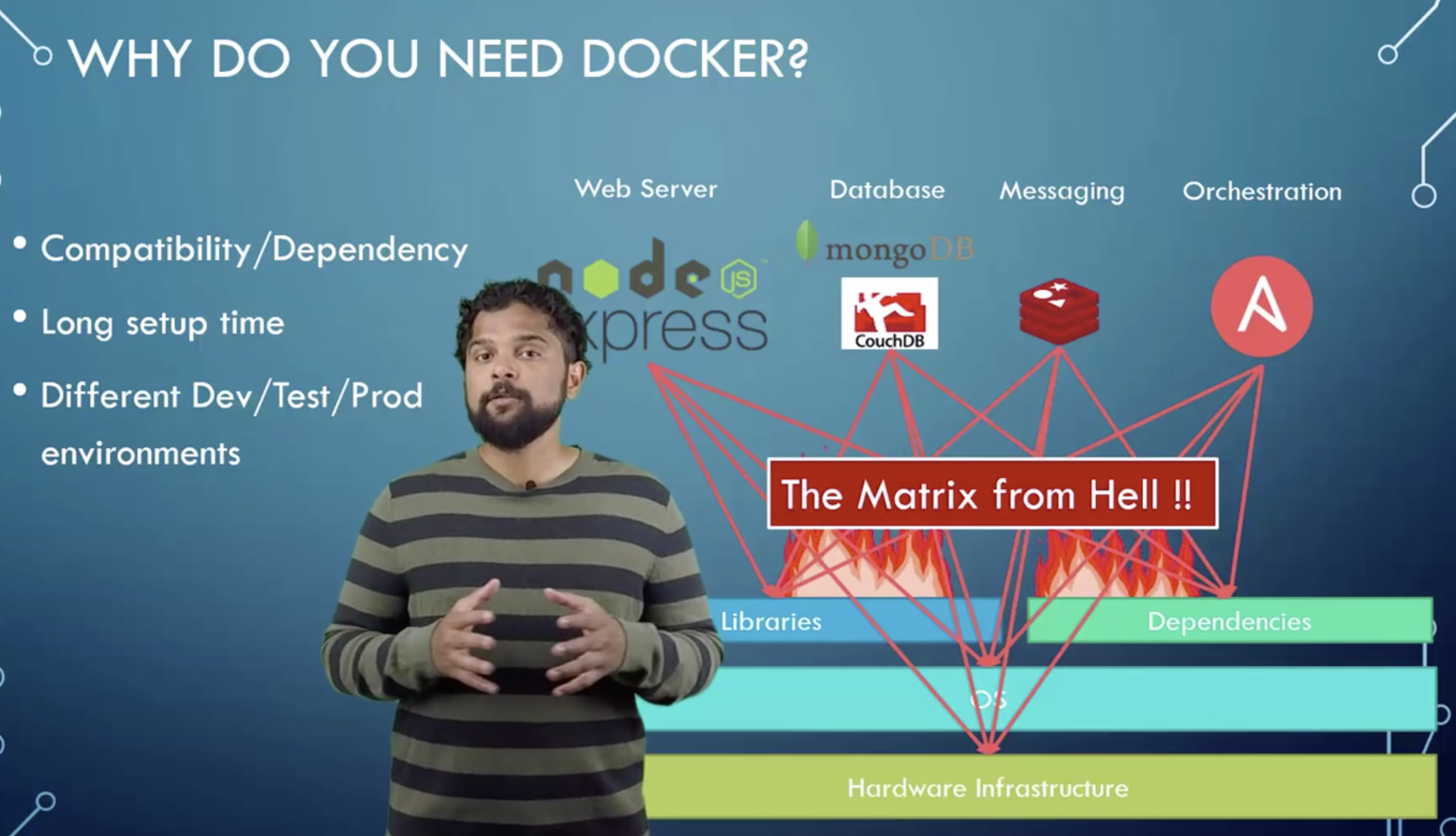
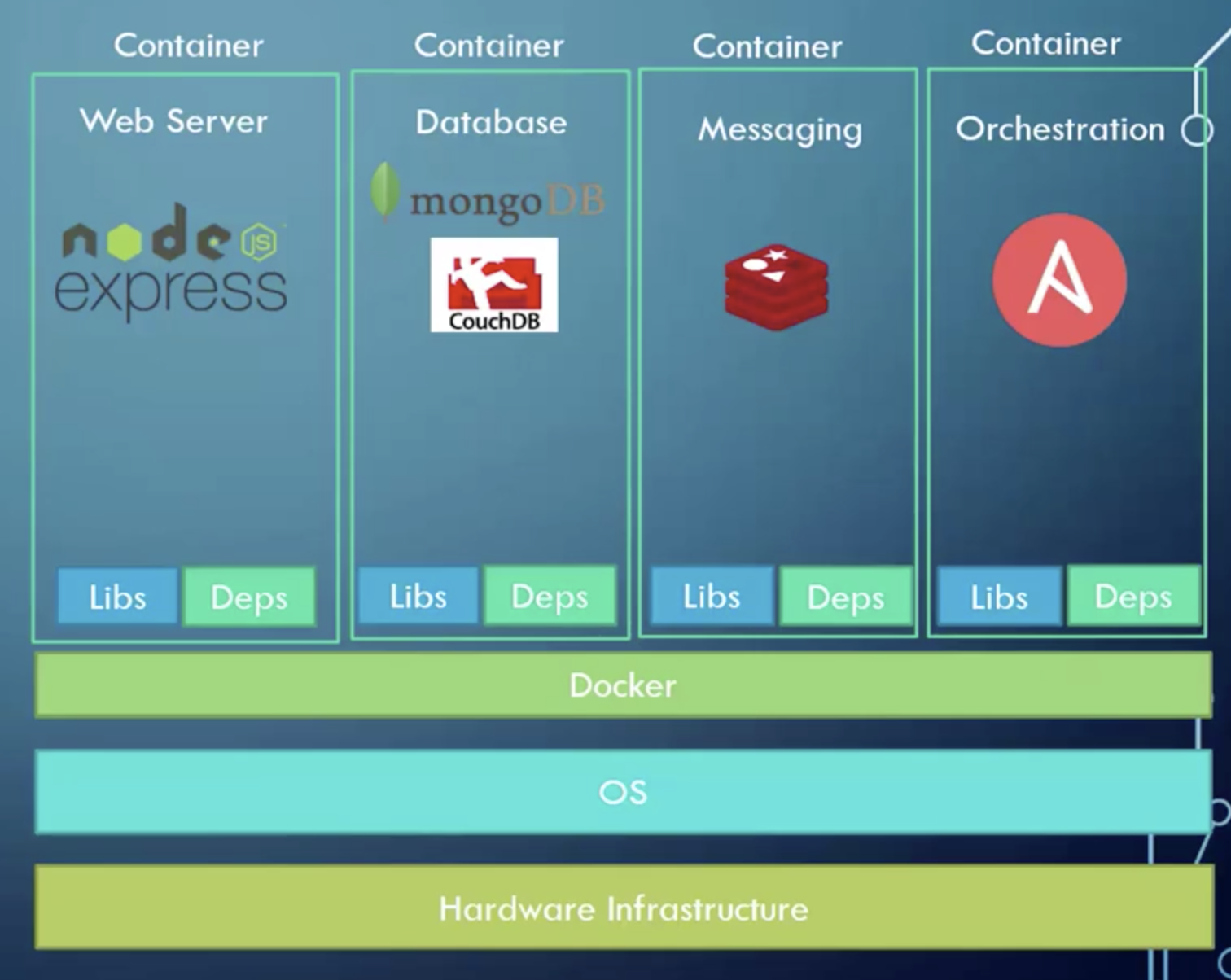
**Introduction**

* Why do we need Docker?
  + Building am end-to-end stack can be sometimes tricky, as we need to make sure all components **are compatible with the OS.**
  + Compatibility between services and dependencies.
  + Upgrading newer versions
  + New developers onboard find it difficult to set up a new environment.

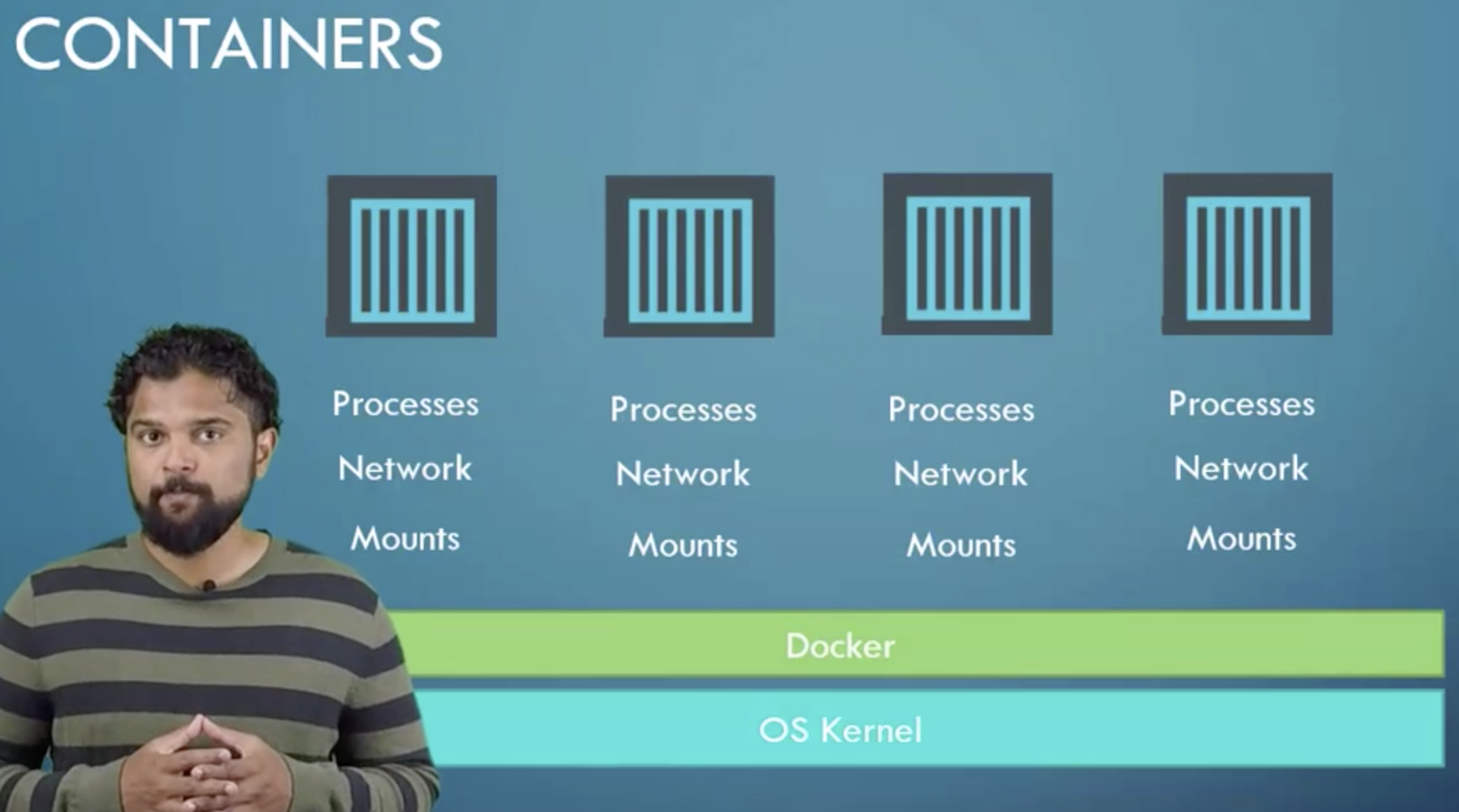


* With Docker, we can run each component in a separate container with its own libraries and dependencies, all in the same VM and OS, but with separate environment or containers.
* Need to set up a Docker configuration once for all developer and use a run command to set up the environment.



**What are Containers?**

* Containers are completely isolated environments with their own processes, mounts and networking interfaces, just like VM’s, except they all share the same OS Kernel.

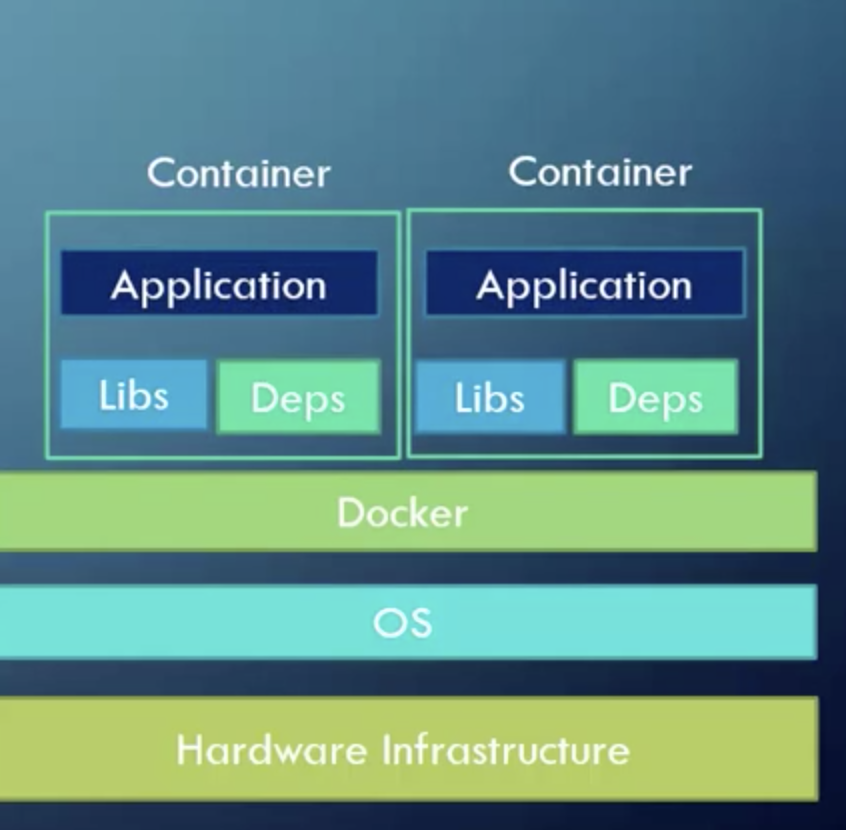


**Sharing the Kernel?**

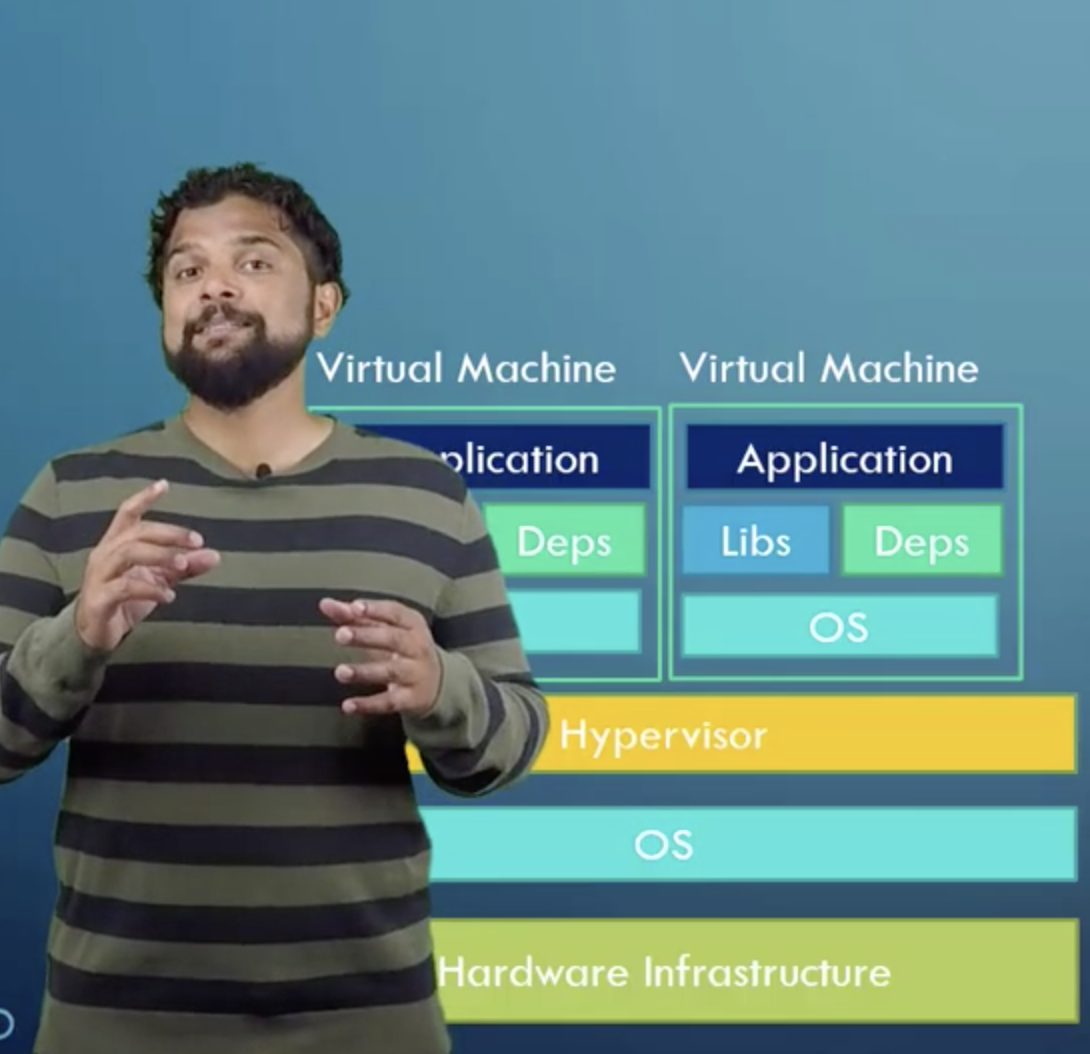
* You won’t be able to run a Windows based container on a Docker host with Linux OS on it. For that you need to run Docker with a Windows server.
  + Docker is not a VM!! It is not meant to run multiple OS and Kernels in the same Hardware.
  + The main purpose of Docker is to **containerize applications**, to **ship them** and to **run them**.

**Differences between VM’s and Containers**

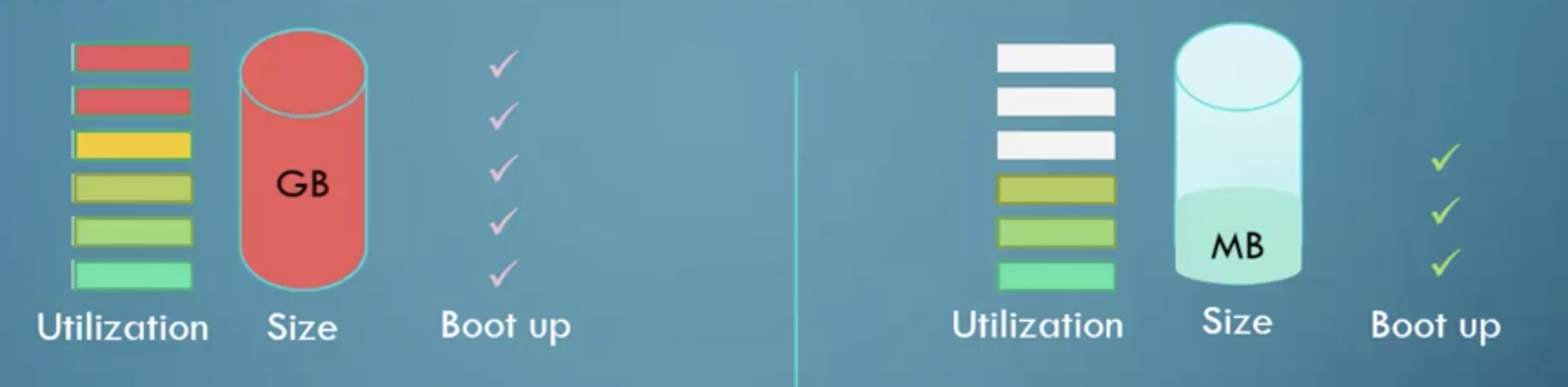
* With Docker, we have the underlying hardware, the OS and containers that will be managed by Docker.



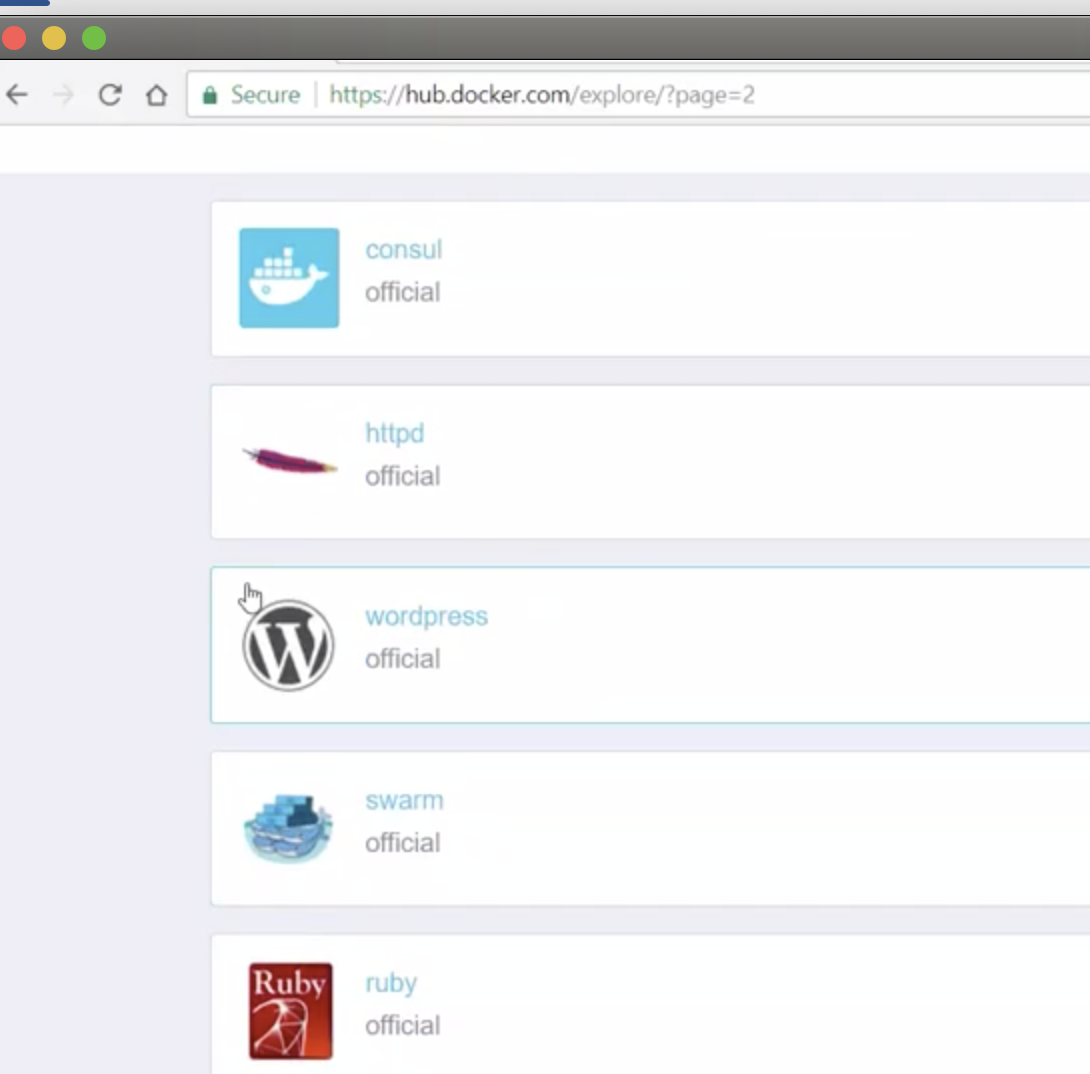
* In the case of VM’s, every VM has its own OS



* Another difference is that VM’s are heavy in utilizing machine’s resources, takes longer to boot up and uses higher disk space, while **Docker is lightweight** and uses small amount of resources.



* We can find ready-made Docker configs with OS’s pre-installed and configured in the Docker Hub or docker store.
  + You can create your own image and push it to the Hub.



**Example of commands:**

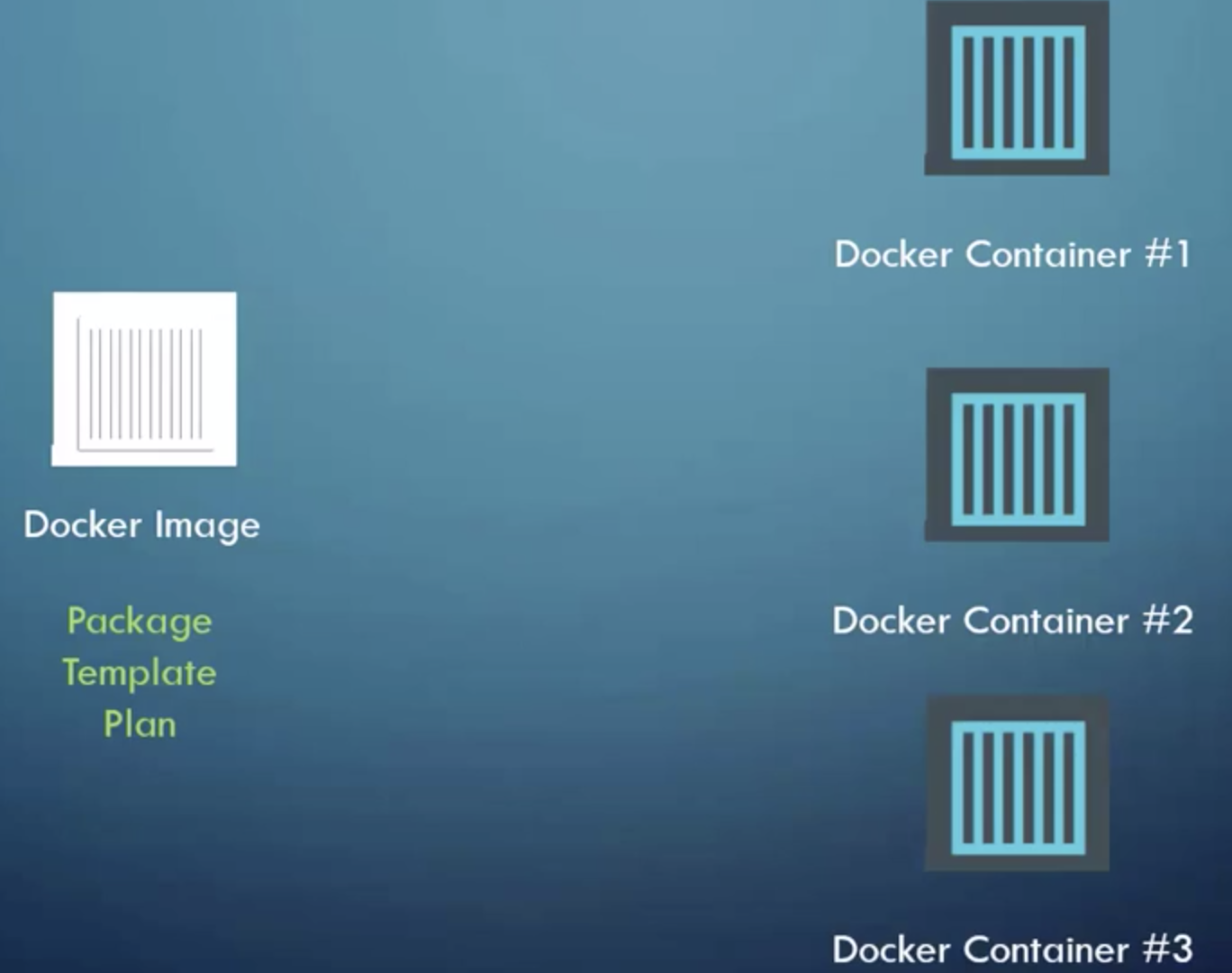
* + **Docker run <image\_name>**
    - *Docker run ansible*
  + **We can run multiples instances:**

*Docker run nodejs*

*Docker run nodejs*

**Container Vs Image**

* + An Image is a package or a Template. It is used to **create one or more Containers.**



* + **Containers are running instances of images**. They are isolated and have their own environment and set of processes.

**Examples of commands**

$ docker run hello-world

Hello from Docker!

This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:

1. The Docker client contacted the Docker daemon.

2. The Docker daemon pulled the "hello-world" image from the Docker Hub.

(amd64)

3. The Docker daemon created a new container from that image which runs the

executable that produces the output you are currently reading.

4. The Docker daemon streamed that output to the Docker client, which sent it

to your terminal.

To try something more ambitious, you can run an Ubuntu container with:

**$ docker run -it ubuntu bash**

Share images, automate workflows, and more with a free Docker ID:

https://hub.docker.com/

For more examples and ideas, visit:

<https://docs.docker.com/get-started/>

$ docker run -it ubuntu bash

root@fbf03bcaed9a:/# cat /etc/\*release\*

DISTRIB\_ID=Ubuntu

DISTRIB\_RELEASE=18.04

DISTRIB\_CODENAME=bionic

DISTRIB\_DESCRIPTION="Ubuntu 18.04.2 LTS"

NAME="Ubuntu"

VERSION="18.04.2 LTS (Bionic Beaver)"

ID=ubuntu

ID\_LIKE=debian

PRETTY\_NAME="Ubuntu 18.04.2 LTS"

VERSION\_ID="18.04"

HOME\_URL="https://www.ubuntu.com/"

SUPPORT\_URL="https://help.ubuntu.com/"

BUG\_REPORT\_URL="https://bugs.launchpad.net/ubuntu/"

PRIVACY\_POLICY\_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"

VERSION\_CODENAME=bionic

UBUNTU\_CODENAME=bionic

* **Exit from container**

root@fbf03bcaed9a:/# exit