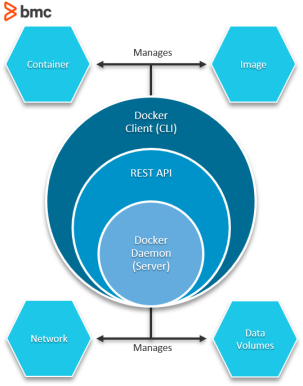
**Docker**

# What is Docker?

Docker is a Linux-based, open-source containerization platform that developers use to build, run, and package applications for deployment using containers.

Unlike [virtual machines](https://www.bmc.com/blogs/containers-vs-virtual-machines/), Docker containers offer:

* OS-level abstraction with optimum resource utilization
* Interoperability
* Efficient build and test
* Faster application execution

Fundamentally, Docker containers modularize an application’s functionality into multiple components that allow deploying, testing, or scaling them independently when needed.

Take, for instance, a Docker containerized database of an application. With such a framework, you can scale or maintain the database independently from other modules/components of the application without impacting the workloads of other critical systems.

# Components of a Docker architecture

Docker comprises the following different components within its core architecture:

* Images
* Containers
* Registries
* Docker Engine

## Containers

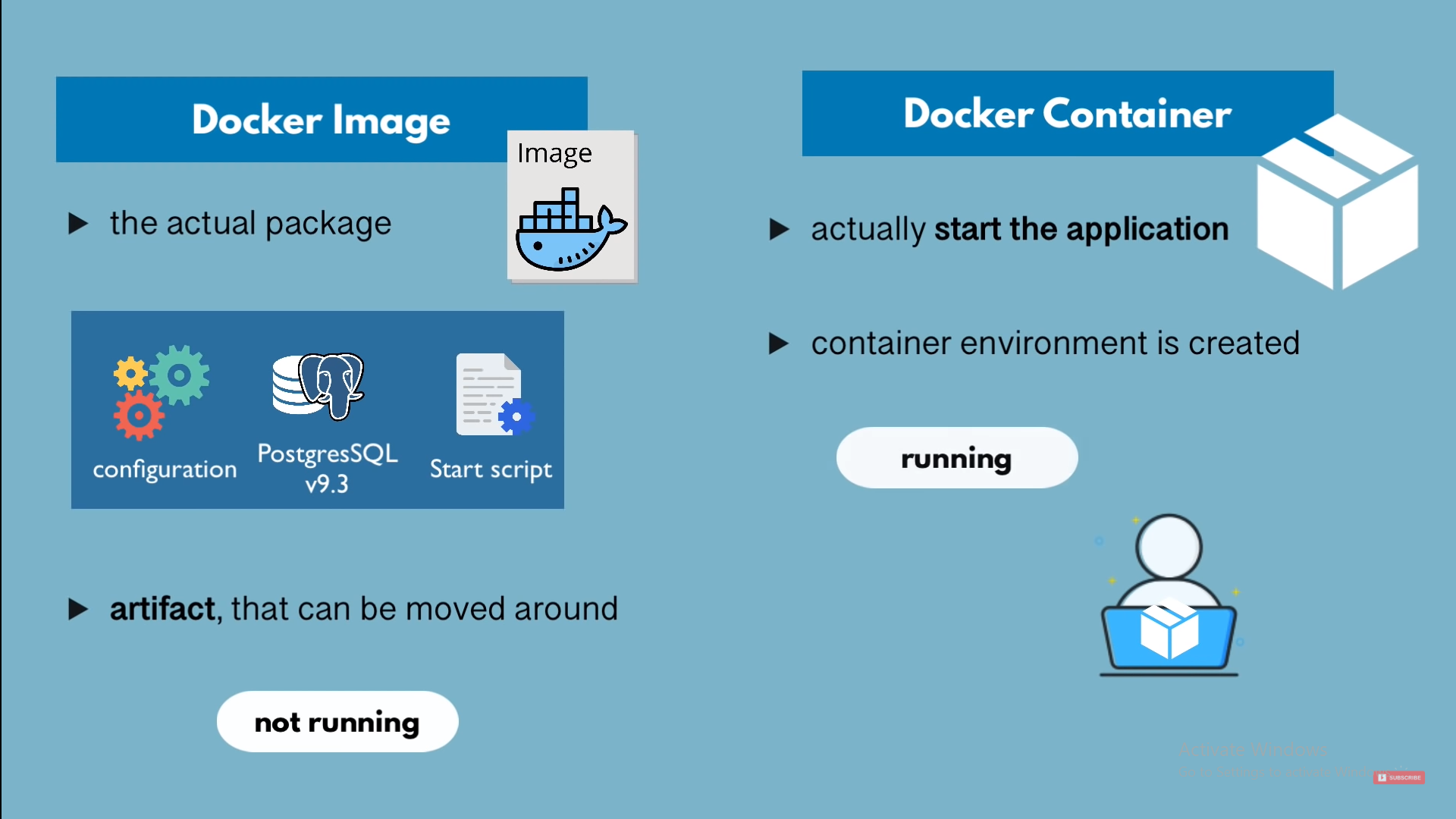
* Containers are live instances of images on which an application or its independent modules are run.In an object-oriented programming analogy, an image is a class and the container is an instance of that class. This allows operational efficiency by allowing to you to multiple containers from a single image.
* A way to package application with all necessary dependencies and configurations.Portable artifact easily shared and moved around.
* Lives in container repository (private and public like DockerHub)

## Images

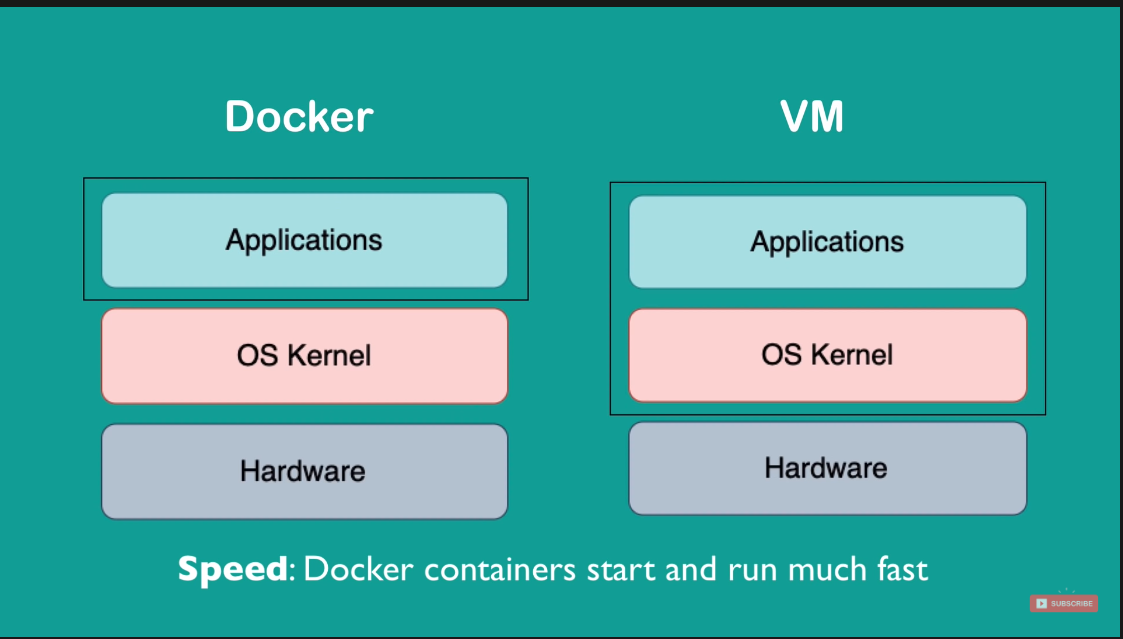
Images are like blueprints containing instructions for creating a Docker container. Images define:

* Application dependencies
* The processes that should run when the application launches

You can get images from DockerHub or create your own images by including specific instructions within a file called Dockerfile.



## Docker vs Virtual Machine(VM)



Compatibility : VM of any OS can run any OS host

## Registries

A Docker registry is like a repository of images.

The default registry is the Docker Hub**,**a public registry that stores public and official images for different languages and platforms. By default, a request for an image from Docker is searched within the Docker Hub registry.

You can also own a private registry and configure it to be the default source of images for your custom requirements.

## Docker Engine

The Docker Engine is of the core components of a Docker architecture on which the application runs. You could also consider the Docker Engine as the application that’s installed on the system that manages containers, images, and builds.

A Docker Engine uses a client-server architecture and consists of the following sub-components:

* **The** **Docker Daemon** is basically the server that runs on the host machine. It is responsible for building and managing Docker images.
* **The** **Docker Client** is a command-line interface (CLI) for sending instructions to the Docker Daemon using special Docker commands. Though a client can run on the host machine, it relies on Docker Engine’s REST API to connect remotely with the daemon.
* **A** **REST**[**API**](https://www.bmc.com/blogs/microservice-vs-api/) supports interactions between the client and the daemon.