DOCKER

**Docker** is defined as the platform for containerizing the applications to isolate it from each other in order to ensure high availability and more efficiency irrespective of the environments such as **Development, Testing or Production**.

Docker is a tool that allows developers, sys-admins etc. to easily deploy their applications in a sandbox (called containers) to run on the host operating system i.e. Linux.

With DOCKER, you can treat containers like extremely lightweight, modular virtual machines. And you get flexibility with those containers—you can create, deploy, copy, and move them from environment to environment, which helps optimize your apps for the cloud.

Docker Engine is a client-server application with these major components:

* A server which is a type of long-running program called a daemon process (the dockerd command).
* A REST API which specifies interfaces that programs can use to talk to the daemon and instruct it what to do.
* A command-line interface (CLI) client (the docker command).
* The CLI uses the Docker REST API to control or interact with the Docker daemon through scripting or direct CLI commands

Docker uses a client-server architecture.

The Docker client and daemon can run on the same system, or you can connect a Docker client to a remote Docker daemon. The Docker client and daemon communicate using a REST API, over UNIX sockets, or a network interface.

*he core components that compose Docker:*

* + - *The Docker client and server also called the Docker Engine.*
    - [*Docker Images*](https://lms.clarusway.com/mod/lesson/view.php?id=2159)
    - *Registries*
    - [*Docker Containers*](https://lms.clarusway.com/mod/lesson/view.php?id=2158)
* [**Docker images**](https://lms.clarusway.com/mod/lesson/view.php?id=2159)
* Images are the building blocks of the Docker world. Containers are launched from images. [Docker images](https://lms.clarusway.com/mod/lesson/view.php?id=2159" \o "Docker Images) are light-weight, portable, reproduce-able, and declarative.

Because of **high resource sharing**, we can build [Docker containers](https://lms.clarusway.com/mod/lesson/view.php?id=2158" \o "Docker Containers) with a **small footprint** that makes application **distribution easier** and it makes container **startup times faster**.

What are the components of Docker Architecture

* **Docker Client:** This performs Docker build pull and run operations to establish communication with the Docker Host. The Docker command uses Docker API to call the queries to be run.
* **Docker Host:** This component contains Docker Daemon, Containers and its images. The images will be the kind of metadata for the applications which are containerized in the containers. The Docker Daemon establishes a connection with Registry.
* **Registry:**This component will be storing the [Docker images](https://lms.clarusway.com/mod/lesson/view.php?id=2159" \o "Docker Images). The public registries are Docker Hub and Docker Cloud which can be used by anyone.

What is the advantage of Docker over hypervisors?  
A: Docker is light weight and more efficient in terms of resource uses because it uses the host underlying kernel rather than creating its own hypervisor.

Q: What is Docker Container  
A: A Docker Container is a form of encapsulation to the application which holds all the dependencies which share the kernel with other containers in the duration of running the isolated processes on the host operating system. A Docker container can be created by creating a Docker image. These [Docker images](https://lms.clarusway.com/mod/lesson/view.php?id=2159" \o "Docker Images) can be run after that using Docker commands. Docker containers are the instances of the [Docker images](https://lms.clarusway.com/mod/lesson/view.php?id=2159" \o "Docker Images) at the runtime. [Docker images](https://lms.clarusway.com/mod/lesson/view.php?id=2159" \o "Docker Images) can be stored in any public hosts or private hosts like Docker hub. Docker Image is a set of files which can be run in an isolated process.