DAY -1

DOCKER

Docker is a software platform that allows you to build, test, and deploy applications quickly. Docker packages software into standardized units called containers that have everything the software needs to run including libraries, system tools, code, and runtime. Using Docker, you can quickly deploy and scale applications into any environment and know your code will run.

Using Docker lets you ship code faster, standardize application operations, seamlessly move code, and save money by improving resource utilization. With Docker, you get a single object that can reliably run anywhere. Docker's simple and straightforward syntax gives you full control. Wide adoption means there is a robust ecosystem of tools and off-the-shelf applications that are ready to use with Docker.

You can use Docker containers as a core building block creating modern applications and platforms. Docker makes it easy to build and run distributed microservices architectures, deploy your code with standardized continuous integration and delivery pipelines, build highly-scalable data processing systems, and create fully-managed platforms for your developers.

What is Docker?

Docker is popular virtualization software that helps its users in developing, deploying, monitoring, and running applications in a Docker Container with all their dependencies.

Docker containers include all dependencies (frameworks, libraries, etc.) to run an application in an efficient and bug-free manner.

Docker Containers have the following benefits:

- Light-weight
- Applications run in isolation
- Occupies less space
- Easily portable and highly secure
- Short boot-up time

What is a Virtual Machine?

A virtual machine (VM) is a computing environment or software that aids developers to access an operating system via a physical machine.

Docker vs. Virtual Machine

Differences	Docker	Virtual Machine
Operating system	Docker is a container-based model where containers are software packages used for	It is not a container-based model; they use user space along with the kernel space of an OS

	executing an application on any operating system In Docker, the containers share the host OS kernel Here, multiple workloads can run on a single OS	It does not share the host kernel Each workload needs a complete OS or hypervisor
Performance	Docker containers result in high-performance as they use the same operating system with no additional software (like hypervisor) Docker containers can start up quickly and result in less boot-up time	Since VM uses a separate OS; it causes more resources to be used Virtual machines don't start quickly and lead to poor performance
Portability	With docker containers, users can create an application and store it into a container image. Then, he/she can run it across any host environment Docker container is smaller than VMs, because of which the process of transferring files on the host's filesystem is easier	It has known portability issues. VMs don't have a central hub and it requires more memory space to store data While transferring files, VMs should have a copy of the OS and its dependencies because of which image size is increased and becomes a tedious process to share data

Speed	The application in Docker containers starts with no delay since the OS is already up and running These containers were basically designed to save time in the deployment process of an application	It takes a much longer time than it takes for a container to run applications To deploy a single application, Virtual Machines need to start the entire OS, which would cause a full boot process