DEVOPS Assignment (2)

Jitendra Lokesh Bhat – 4NI19IS039

1. Hypervisors and Docker : -

	Hypervisors	**Docker**
Definition	A hypervisor is a software layer that allows multiple operating systems to run on a single physical machine.	Docker is a containerization platform that allows developers to package applications and their dependencies into containers.
Virtualization method	Hypervisors use hardware virtualization to create virtual machines (VMs). Each VM runs its own operating system	Docker uses operating system level virtualization to create containers that share the host operating system's kernel.
Resource overhead	Hypervisors introduce a significant amount of overhead, as each VM requires its own operating system and resources.	Docker containers have a smaller resource overhead, as they share the host operating system and do not require their own OS.
Deployment	Hypervisors require a separate installation on the host machine. VMs can be created and configured manually or through automation	Docker can be installed on the host machine and used to create and manage containers through the command line or API.
Compatibility	Hypervisors are generally compatible	Docker containers are generally portable across

	with most operating systems, but may require specific hardware support.	different operating systems and environments, as long as Docker is installed.
Scaling	Scaling VMs can be time-consuming, as each VM requires its own resources and must be configured manually.	Docker containers are easy to scale, as they can be created and destroyed quickly and do not require manual configuration.
Isolation	Hypervisors provide strong isolation between VMs, as each VM runs its own operating system and has its own resources.	Docker provides moderate isolation between containers, as they share the host operating system and its resources.
Maintenance and updates	Maintaining and updating hypervisors and VMs can be time-consuming, as each VM requires its own operating system and patches	Docker containers are easy to update, as new containers can be created from updated images and replaced with minimal downtime.
Security	Hypervisors provide strong security, as each VM runs its own operating system and is isolated from other VMs.	Docker provides moderate security, as containers share the host operating system and its resources.

2. Comparison between Containers and Virtual machines.

Containers and virtual machines are both ways to isolate applications and their dependencies from the rest of the host system. They both have their own advantages and disadvantages, and the best choice depends on the specific needs of the application being deployed.

Here are some key differences between containers and virtual machines:

- **Isolation**: Virtual machines provide full isolation of the guest operating system from the host. This means that a virtual machine can run a completely different operating system than the host, and the guest operating system is unaware that it is running on a virtual machine. In contrast, containers share the host operating system kernel and libraries with other containers, but they provide isolation of the application and its dependencies from the rest of the host system.

- **Overhead**: Virtual machines require a full copy of the guest operating system, as well as the virtualization software, to be installed on the host. This can result in a larger resource overhead compared to containers. Containers, on the other hand, do not require a full operating system to be included in each container image, and can share the host's kernel and libraries, which makes them more lightweight than virtual machines.

- **Performance**: Because containers share the host operating system kernel and libraries, they can start up faster than virtual machines, which need to boot a full operating system. In addition, containers can be more efficient with resources such as memory and CPU, because they do not have the overhead of a full operating system. However, virtual machines can provide better isolation and resource guarantees, because they are completely separated from the host and other guest operating systems.

- **Use cases**: Virtual machines are typically used when the application requires a specific operating system or runtime environment that is different from the host system, or when the application needs to be isolated from the host and other applications. Containers are often used for applications that are designed to be portable and scalable, and that do not require a specific operating system or runtime environment.

- **Resource allocation**: In a virtual machine, resources such as CPU, memory, and storage are allocated to each individual virtual machine. This means that each virtual machine has its own set of resources, regardless of whether they are being used or not. In contrast, containers share the same host operating system and resources, which allows for more efficient resource utilization

- **Deployment**: Virtual machines require a full operating system and system libraries to be installed, which can be time-consuming and require more storage space. Containers, on the other hand, only contain the necessary code and libraries for the application, making them faster to deploy and requiring less storage space.

- **Portability**: Virtual machines are not as portable as containers due to the need to recreate the entire operating system and system libraries on each host. Containers, on the other hand, can be easily moved between hosts as long as the host operating system is compatible. This makes containers a more flexible and portable solution for deployment.