**Introduction to Containerization**

Modern applications must run consistently across diverse environments—development, testing, staging, and production. Traditional **Virtual Machines (VMs)** are resource-heavy and slow to start. In contrast, **containers** are lightweight, portable, and start almost instantly.

* **Docker** simplifies the creation and management of containers.
* **Kubernetes** orchestrates and scales containers across clusters.

**2. Docker Basics**

**Docker** is a platform for developing, shipping, and running applications in containers.

**Key Components:**

* **Docker Engine**: The core runtime that builds and runs containers.
* **Docker Images**: Read-only templates used to create containers.
* **Docker Containers**: Executable instances of Docker images.

Each container includes:

* Application code
* Libraries and dependencies
* Runtime environment

**3. What Is a Docker Image?**

A **Docker Image** is a lightweight, stand-alone, and executable software package that includes everything needed to run an application.

**Example:**

A Python image contains:

* Python interpreter
* OS-level dependencies
* Application code (if added)

Images ensure consistency across environments and can be reused to spin up multiple containers.

**4. Image Creation and Sharing**

Images are created using a **Dockerfile** or by modifying existing images.

**Workflow:**

1. **Build** an image locally using docker build.
2. **Store** it locally or push it to a registry like **Docker Hub**.
3. **Share** it with others or pull it from the registry using docker pull.

**5. Dockerfile**

A **Dockerfile** is a script containing instructions to build a Docker image.

**Example:**

FROM openjdk:17

COPY app.jar /app.jar

CMD ["java", "-jar", "/app.jar"]

**Common Commands:**

* FROM: Specifies the base image.
* COPY / ADD: Adds files to the image.
* RUN: Executes commands (e.g., install dependencies).
* CMD: Defines the default command when the container starts.

**6. Docker Hub**

**Docker Hub** is a cloud-based registry for Docker images.

* Hosts **public** and **private** repositories.
* Enables sharing and versioning of images.

**Useful Commands:**

* docker pull nginx: Downloads the official NGINX image.
* docker push myimage:v1: Uploads a custom image to Docker Hub.

**7. What Is Kubernetes?**

**Kubernetes (K8s)** is an open-source platform for **container orchestration**. It automates the deployment, scaling, and management of containerized applications.

**Key Features:**

* **Scaling**: Automatically adjusts the number of containers.
* **Self-healing**: Restarts failed containers.
* **Load balancing**: Distributes traffic across containers.
* **Service discovery**: Locates containers within the cluster.

**8. Kubernetes Concepts and Orchestration**

**Orchestration** refers to automating the lifecycle of containers—deployment, scaling, networking, and storage.

**Core Concepts:**

* **Pod**: The smallest deployable unit; can contain one or more containers.
* **Deployment**: Ensures the desired number of pods are running and updated.
* **Service**: Exposes pods to internal or external traffic.
* **Ingress**: Manages external HTTP/HTTPS traffic to services.
* **ConfigMap & Secrets**: Store configuration data and sensitive information securely.