What are microservices?

Microservices (or microservices architecture) are a cloud native architectural approach in which a single application is composed of many loosely coupled and independently deployable smaller components, or services.

These services typically have their own technology stack, inclusive of the database and data management model; communicate with one another over a combination of REST APIs, event streaming, and message brokers; and are organized by business capability, with the line separating services often referred to as a bounded context.

While much of the discussion about microservices has revolved around architectural definitions and characteristics, their value can be more commonly understood through fairly simple business and organizational benefits:

Code can be updated more easily - new features or functionality can be added without touching the entire application
Teams can use different stacks and different programming languages for different components.
Components can be scaled independently of one another, reducing the waste and cost associated with having to scale entire applications because a single feature might be facing too much load.
Microservices might also be understood by what they are not. The two comparisons drawn most frequently with microservices architecture are monolithic architecture and service-oriented architecture (SOA).

The difference between microservices and monolithic architecture is that microservices compose a single application from many smaller, loosely coupled services as opposed to the monolithic approach of a large, tightly coupled application.

We can run these microservices on Physical Machines | Virtual Machines | Containers. It is always recommended to host the containers on Virtual Environment. Containers virtualize at the O/S Level where as Virtual Machines virtualize at the hard ware level. So, If we host our containers on Virtual environment we get the best of both worlds like virtualizing at the hardware level and O/S Level.

What is container orchestration?

Container orchestration automates the deployment, management, scaling, and networking of containers. Enterprises that need to deploy and manage hundreds or thousands of Linux® containers and hosts can benefit from container orchestration.

Container orchestration can be used in any environment where you use containers. It can help you to deploy the same application across different environments without needing to

redesign it. And microservices in containers make it easier to orchestrate services, including storage, networking, and security.

Containers give your microservice-based apps an ideal application deployment unit and self-contained execution environment. They make it possible to run multiple parts of an app independently in microservices, on the same hardware, with much greater control over individual pieces and life cycles.

Managing the lifecycle of containers with orchestration also supports DevOps teams who integrate it into CI/CD workflows. Along with application programming interfaces (APIs) and DevOps teams, containerized microservices are the foundation for cloud-native applications.

What is container orchestration used for?

Use container orchestration to automate and manage tasks such as:

- Provisioning and deployment
- Configuration and scheduling
- Resource allocation
- Container availability
- Scaling or removing containers based on balancing workloads across your infrastructure
- Load balancing and traffic routing
- Monitoring container health
- Configuring applications based on the container in which they will run
- Keeping interactions between containers secure

Container orchestration tools :

Container orchestration tools provide a framework for managing containers and microservices architecture at scale. There are many container orchestration tools that can be used for container lifecycle management. Some popular options are Kubernetes, Docker Swarm, and Apache Mesos.

Kubernetes is an open source container orchestration tool that was originally developed and designed by engineers at Google. Google donated the Kubernetes project to the newly formed Cloud Native Computing Foundation in 2015.