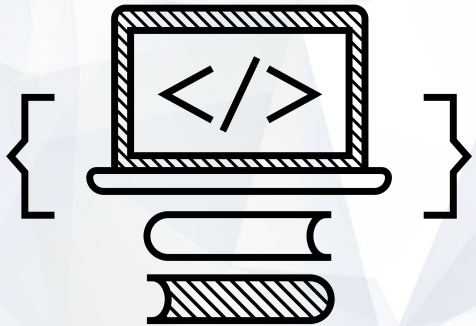
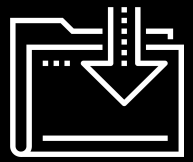


Containerization

Java Accelerator 7

Lesson 5.3




WELCOME

Learning Outcomes

By the end of this lesson, you will be able to:

Explain the differences between virtual machines and containers.

Explain the benefits of Docker.

Install and run an application in Docker.

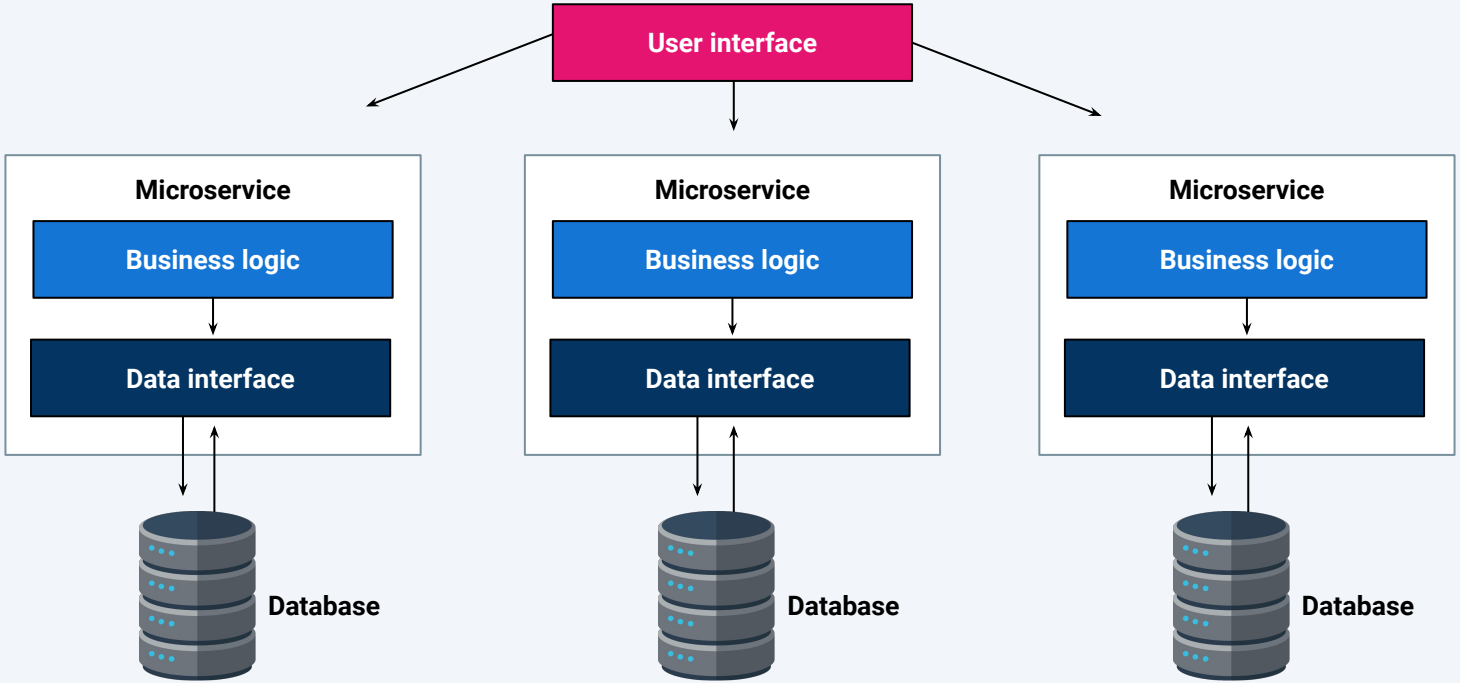
Install and run minikube in Docker.

Deploy an application to minikube.

Microservices Orchestration

Microservice architecture

With a microservices architecture, we are dividing the application into a set of independent, loosely coupled services.



Microservices Orchestration

Benefits of microservices:

Teams can work independently on parallel microservices.

Errors are isolated to specific microservices, making them easier to find.

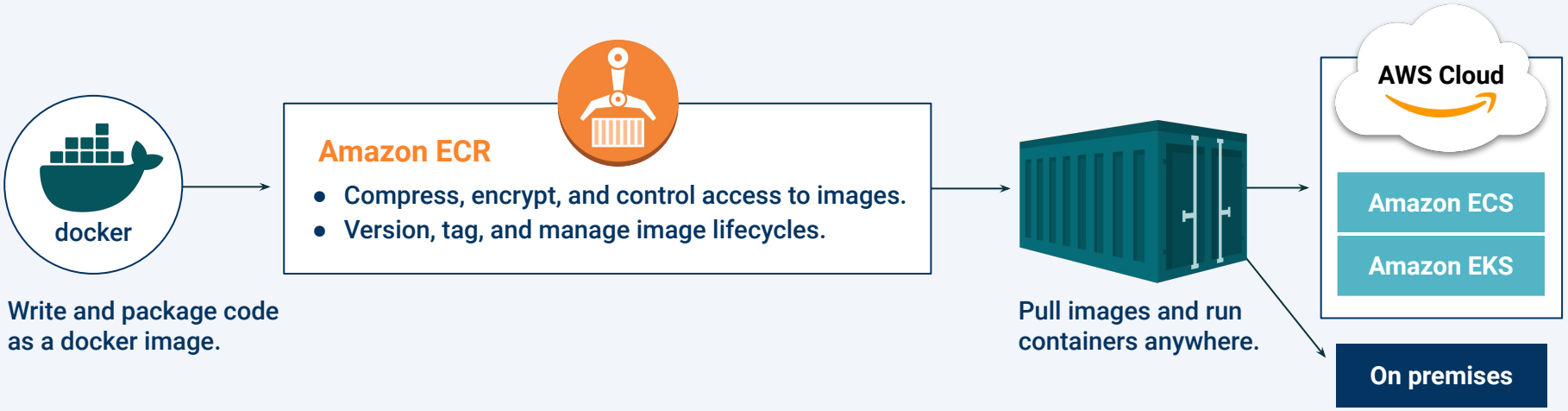
An error in one service means that other services can still function.

Individual microservices can be scaled as the need arises.

Introduction to Containers

Introduction to Containers

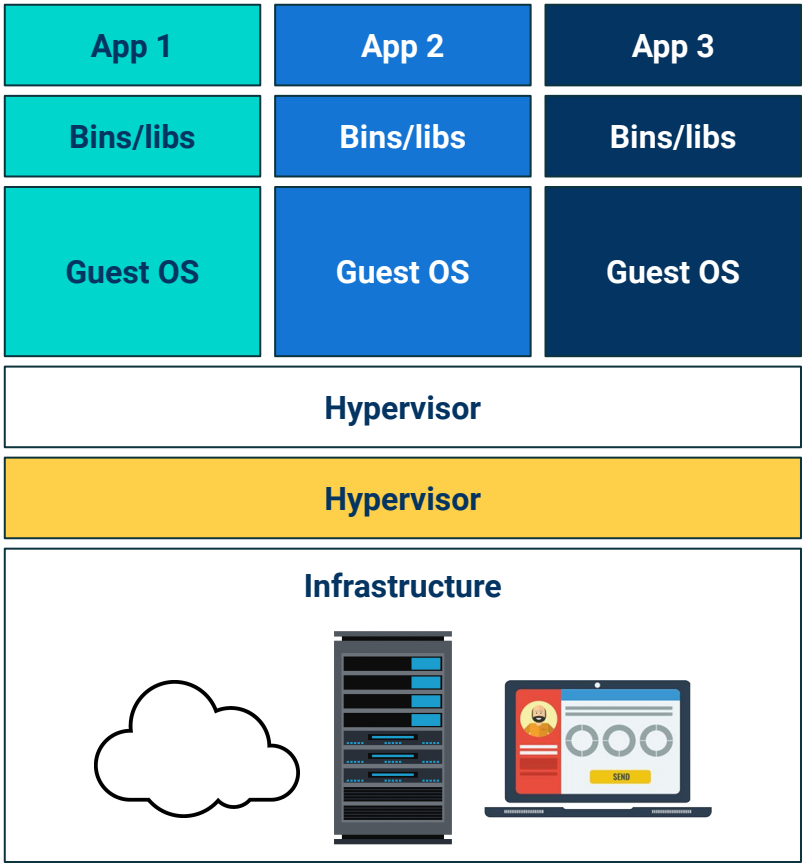
- Everything an application needs is packaged inside a container.
- This includes all of the service code, dependencies, system libraries.
- This lets us deploy a microservice by simply deploying its container.



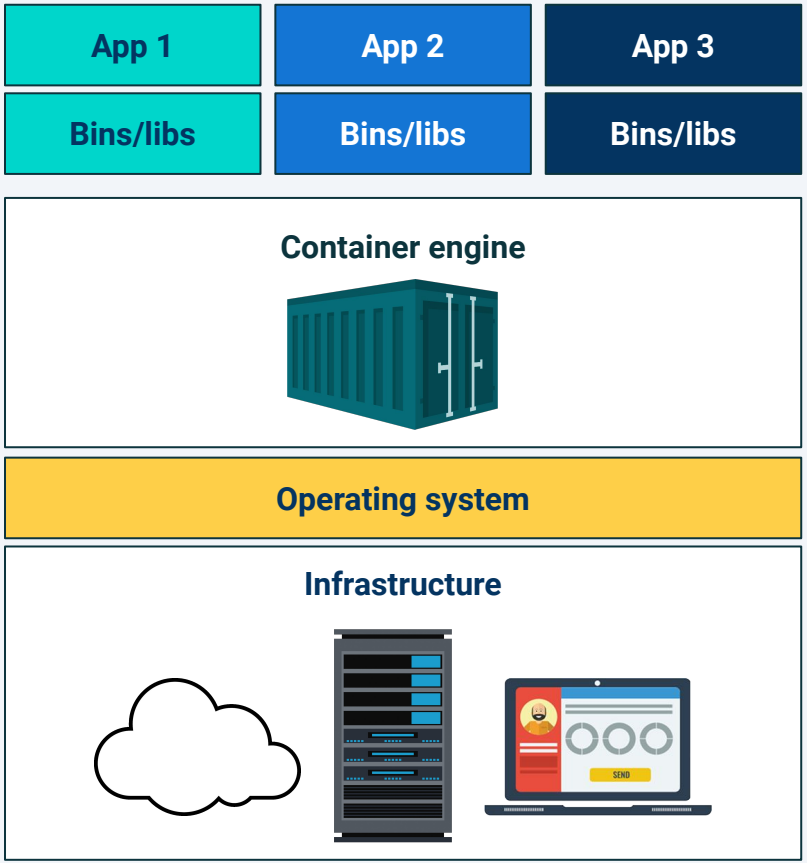
The process of managing and deploying containers is called **orchestration.**

When those containers support microservices, it is **microservices orchestration.**

Virtual Machines



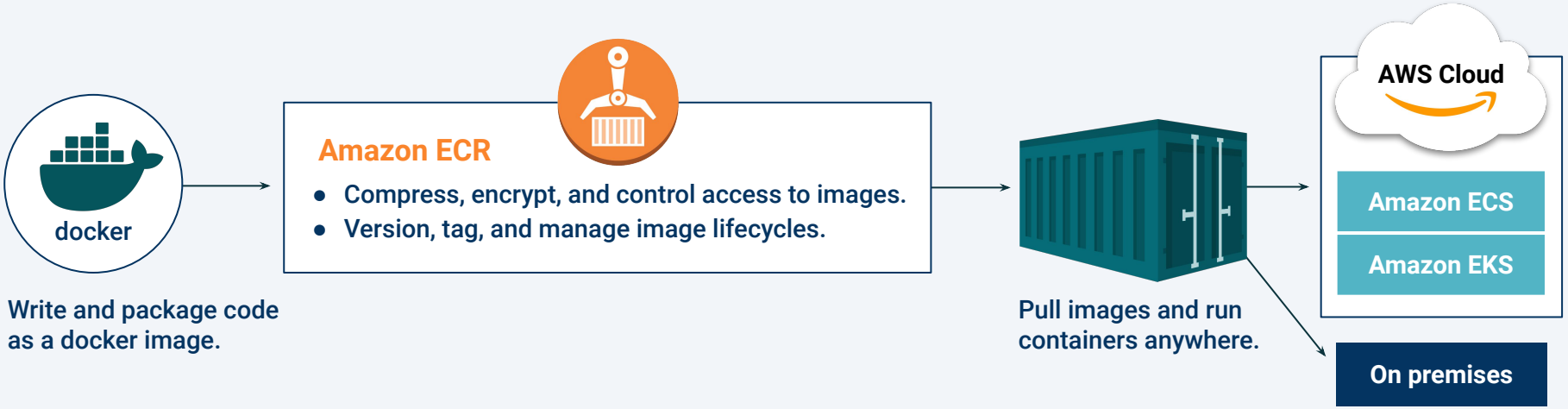
Containers



Containerization

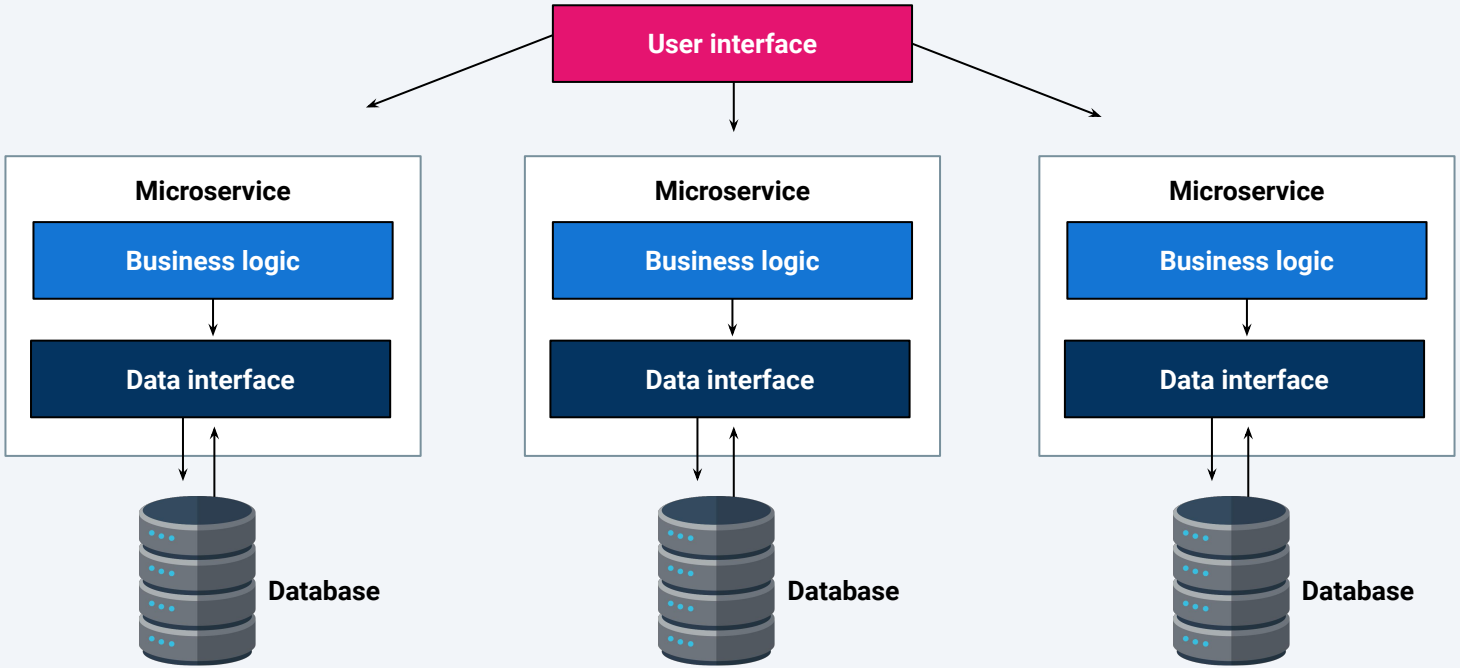
Containers provide a solution to the problem of how to get software to run reliably when moved from one computing environment to another.

This might occur when moving from a staging to a production environment or from an on-premises, physical machine in a data center to a VM in the cloud.



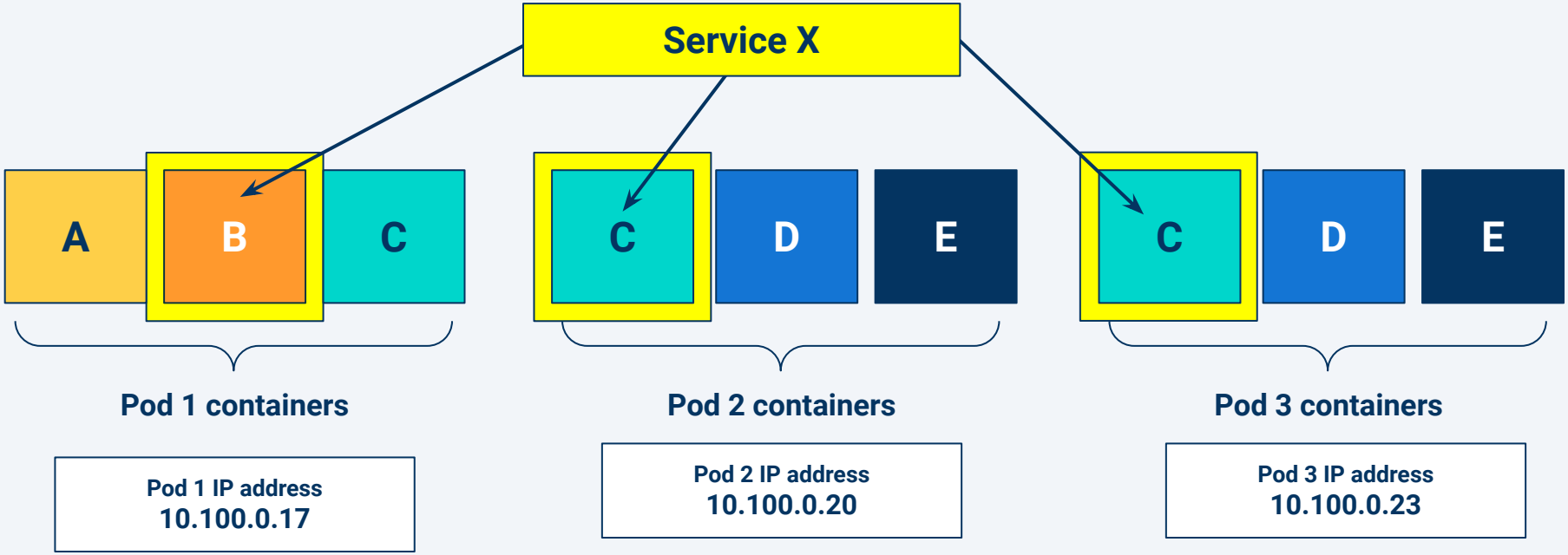
Containerization

With microservices, applications can work regardless of their software versions, dependencies, libraries, and configuration files. This is because microservices bundle these elements into a self-contained runtime environment.



Containerization

Containers work well with microservices. A container can isolate the runtime that's needed to deliver a particular microservice. The microservice can then work in different environments and with different applications.



Docker is a popular service that can package an application and its dependencies in a virtual container, which can then run on any Linux, Windows, or macOS computer. The application can thus run consistently in various environments.

Container Orchestration

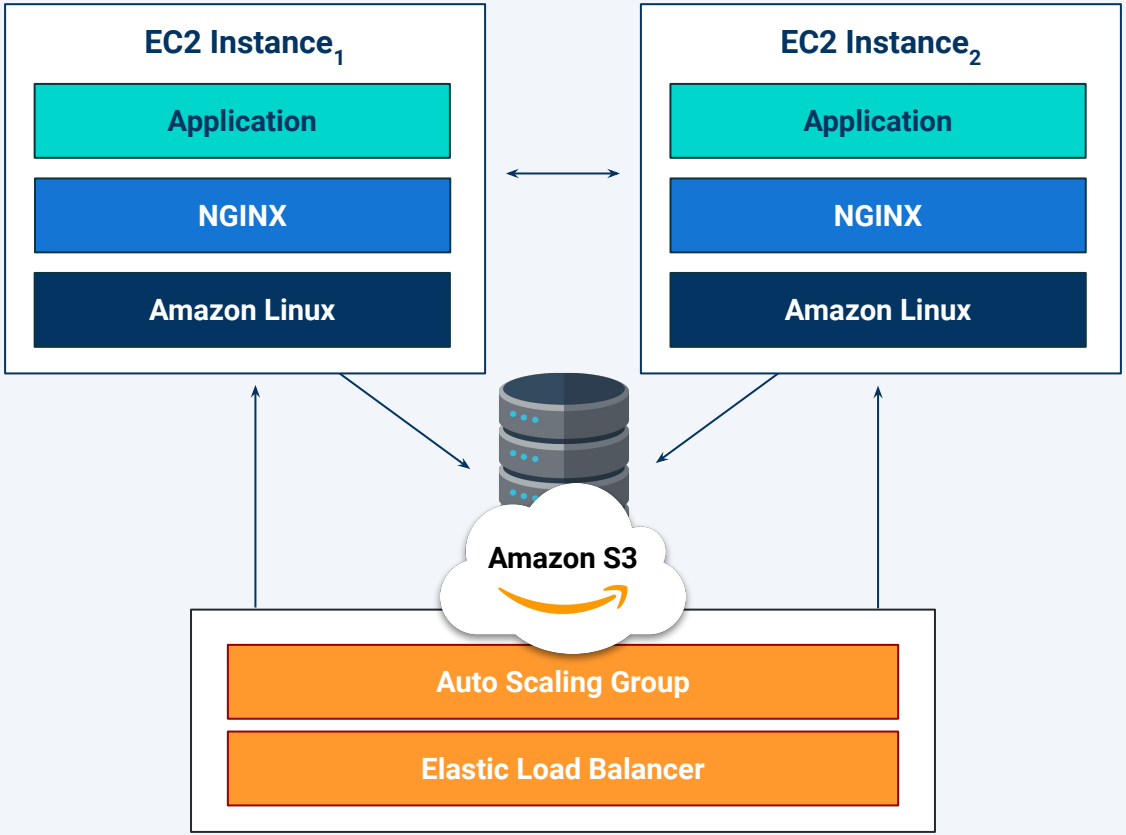
Container orchestration is the process of managing the lifecycles of containers, especially in large, dynamic environments.

This might include:

- Provisioning or deploying containers.
- Scaling them up.
- Allocating resources among them.
- Monitoring their health, or exposing services running inside a container to other applications.

Kubernetes is one of the most popular container orchestration services.

AWS Container Services



AWS Container Services

AWS offers its own container services, including:

Amazon Elastic Container Registry (Amazon ECR)	A container registry.
Amazon Elastic Container Service (Amazon ECS)	A container orchestration service.
AWS Fargate	A serverless compute engine for containers. With these services, cloud developers can work with containers in the context of other AWS services to achieve efficient and automated workflows.

Introduction to Docker

Docker Components

Client

- Docker build
- Docker pull
- Docker run

Docker_Host

Docker daemon

Containers

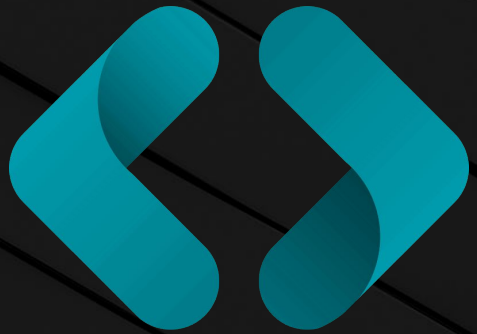
- Oracle
- NGINX
- NGINX
- Oracle

Images

- Oracle
- NGINX

Docker Registry

- Oracle
- NGINX
- Cassandra



Time to <code>