Amazon Web Services
(container)

Agenda

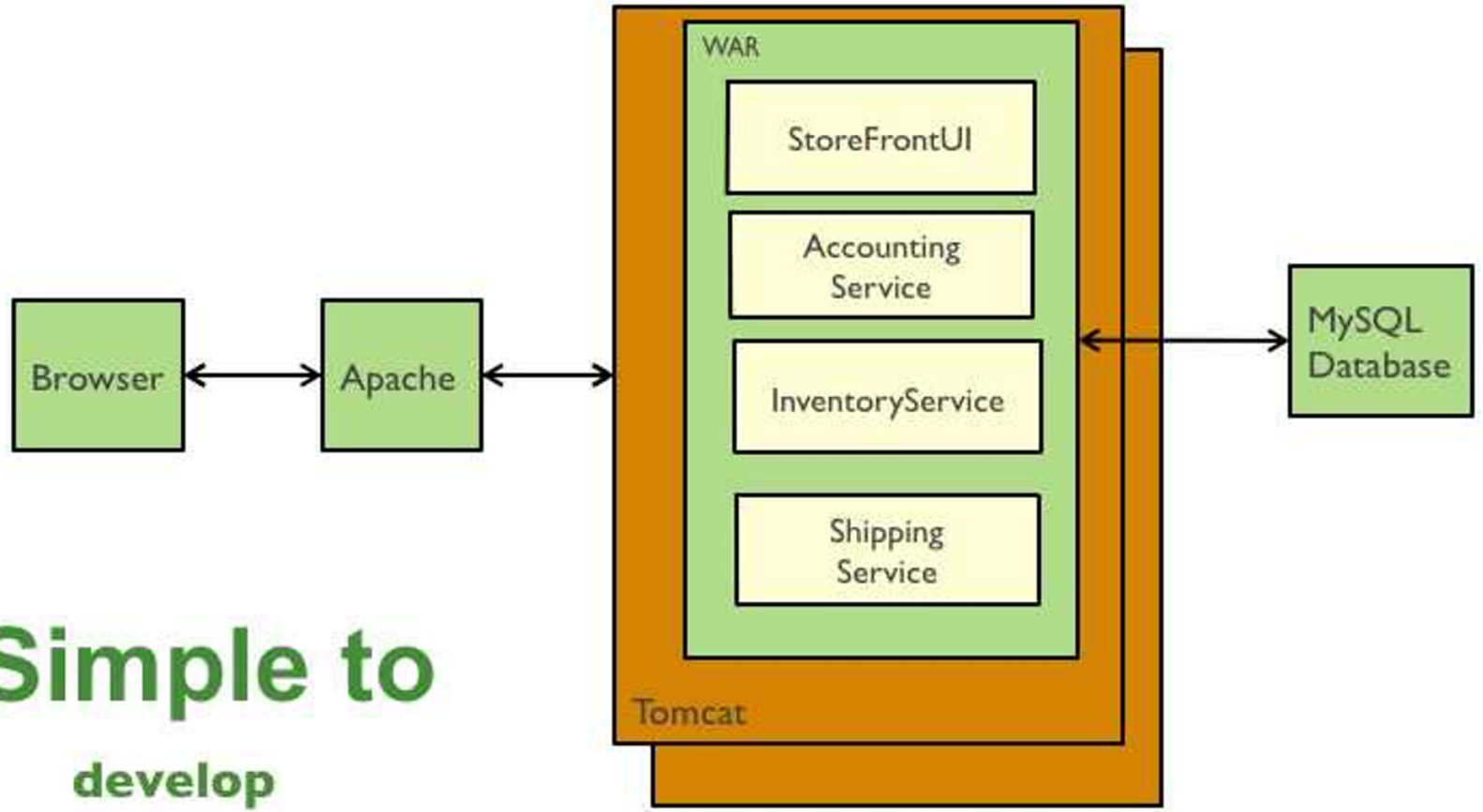
- Introduction
- What is Monolithic & microservices?
- Its Advantages & disadvantages
- Container
- Docker

What is monolethic ?

Monolethic = Mono – Single, Lethic – stone

- Monolethic application depend & build with a single-tiered software application in which the user interface & data access code are combined into a single program from a single platform.
- Everything reside in a same box

Traditional web application architecture

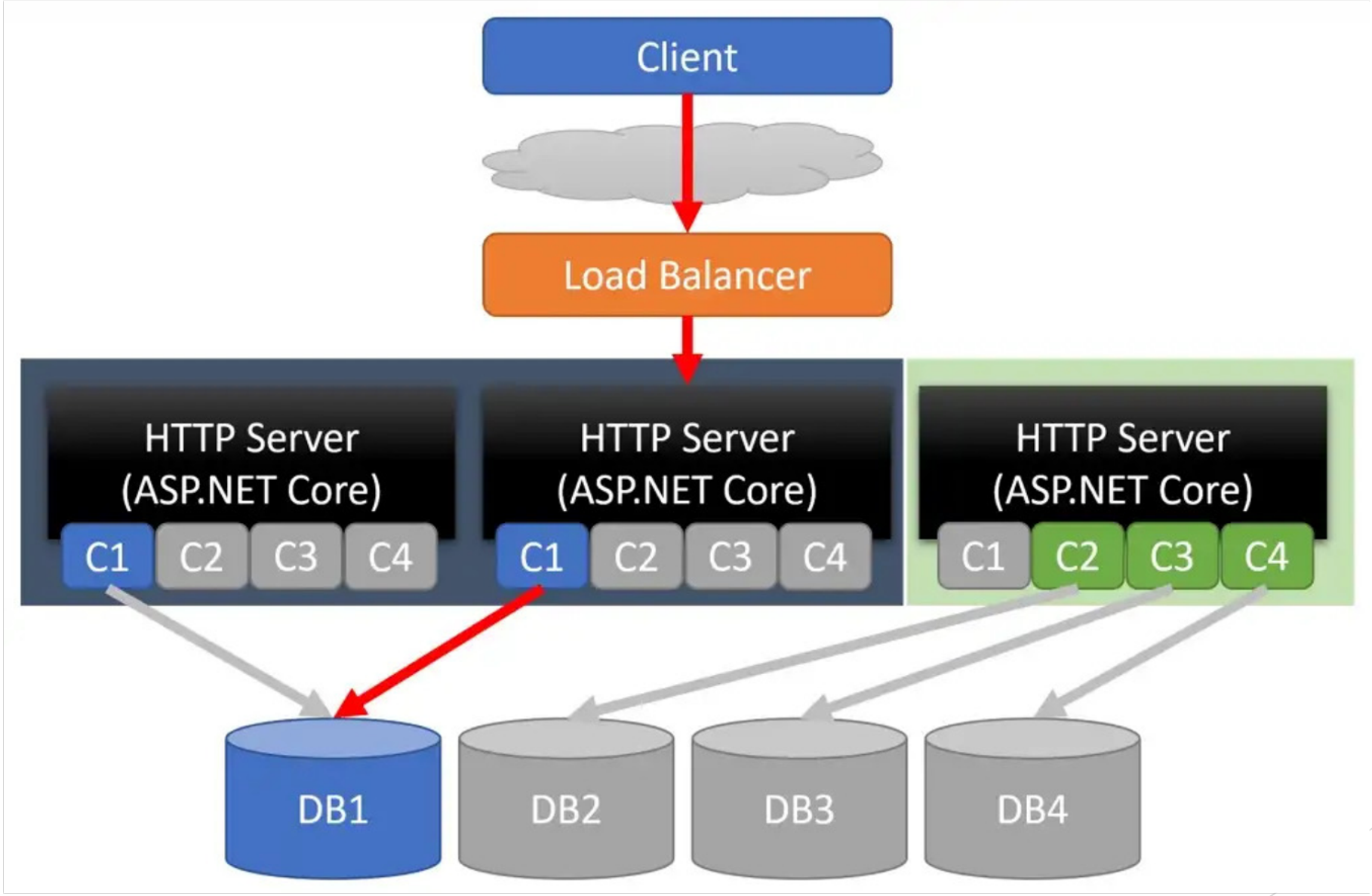


Simple to

develop
test
deploy
scale

Advantages & Disadvantages

- Advantages
  - Easy to deploy
  - Low complexity
- Disadvantages
  - Hard to scale
  - Slow performance
  - Single point of failure
  - Slow continuous development
  - Rigid

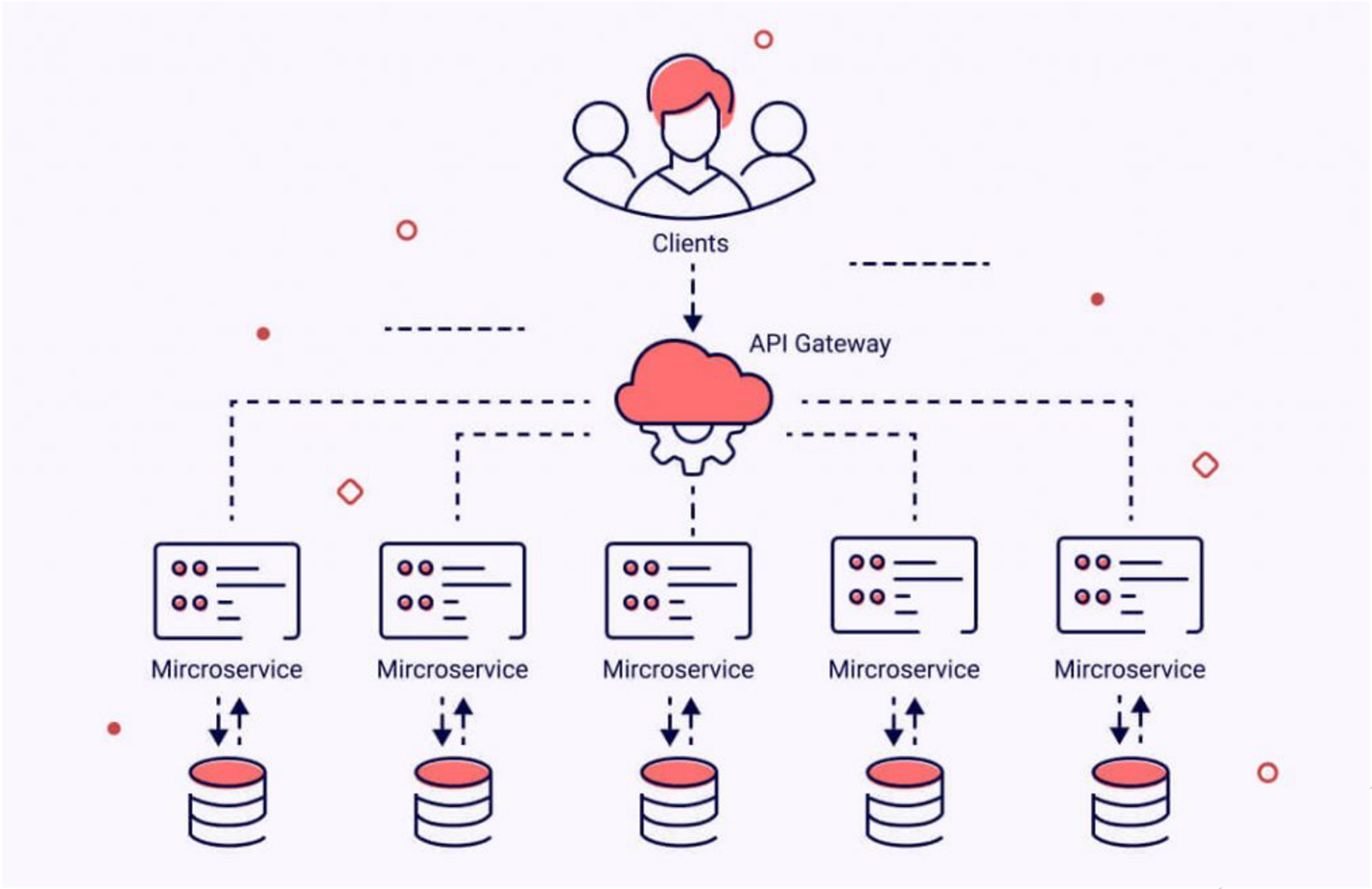

Client
Load Balancer
HTTP Server
(ASP.NET Core)
C1
C2
C3
C4
HTTP Server
(ASP.NET Core)
C1
C2
C3
C4
HTTP Server
(ASP.NET Core)
C1
C2
C3
C4
DB1
DB2
DB3
DB4

Microservices

- Microservices are an architectural and organizational approach to software development where software is composed of small independent services that communicate over well-defined APIs.

Advantages & disadvantages

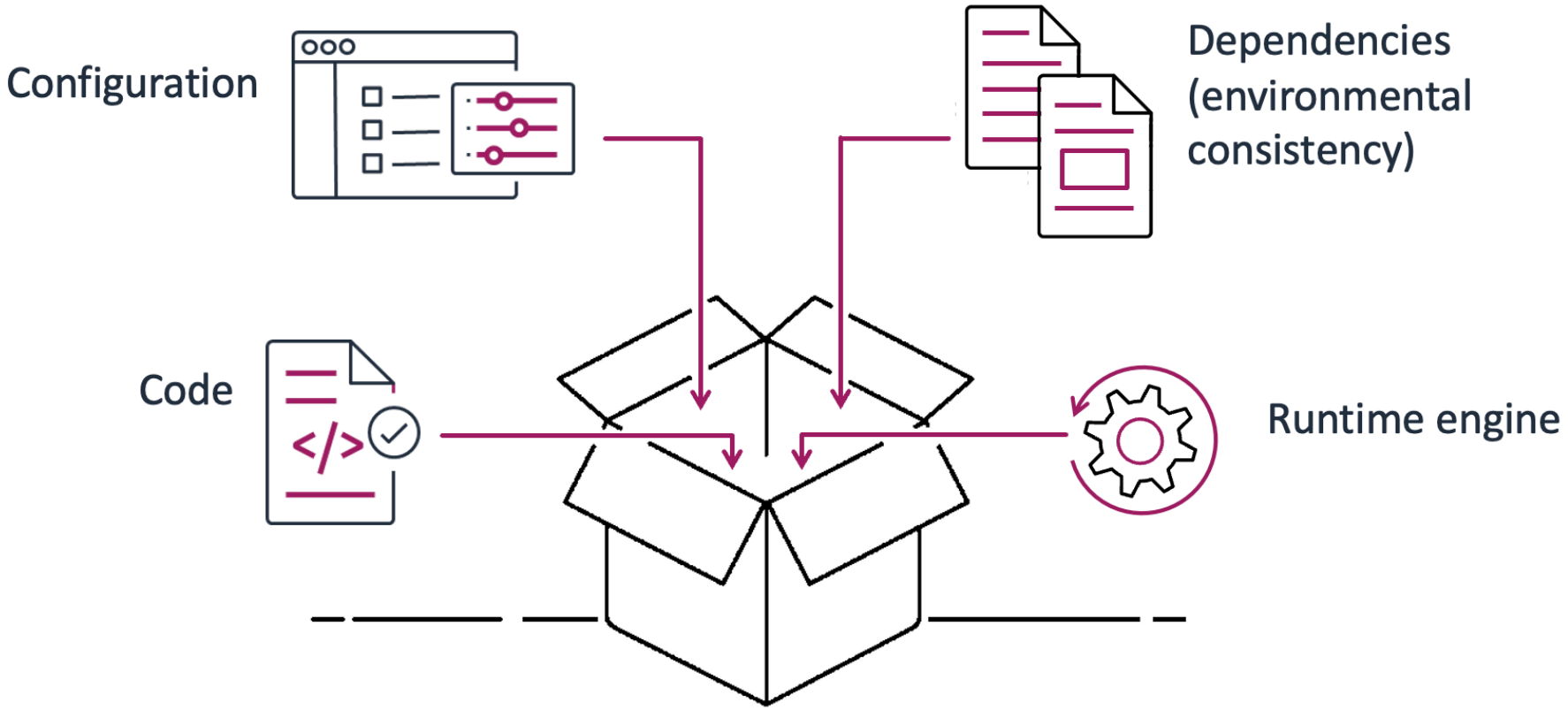
- Advantages
  - Loosely couped
  - Independent development
  - Mixed technology stack
  - Scaling
  - flexible
- Disadvantages
  - High complexity (manage mix stack)
  - consistency
  - Automation
  - debugging


Clients
API Gateway
Mircroservice
Mircroservice
Mircroservice
Mircroservice
Mircroservice

Container (ECS – Elastic Container Service)

- Containers **provide a standard way to package your application's code, configurations, and dependencies into a single object**. Containers share an operating system installed on the server and run as resource-isolated processes, ensuring quick, reliable, and consistent deployments, regardless of environment.

Containers


Configuration
Dependencies
(environmental
consistency)
Code
Runtime engine

Advantages

- Advantages
  - Faster then VM
  - Less memory intensive
  - Easy deployment
  - Less bugs
  - Highly scalable

What is Docker?

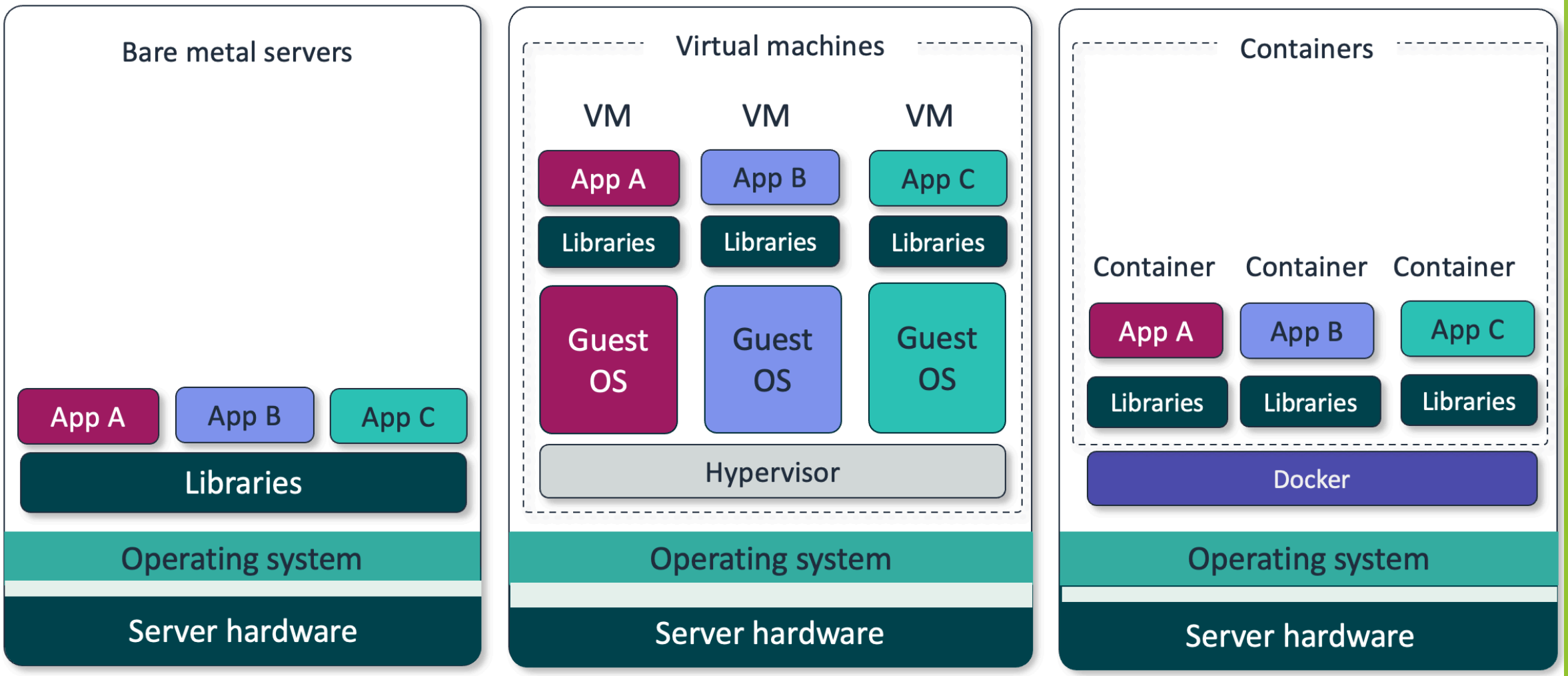
- Docker is a software development platform to deploy apps
- Apps are packaged in containers that can be run on any OS
- Apps run the same, regardless of where they're run
  - Any machine
  - No compatibility issues
  - Predictable behavior
  - Less work
  - Easier to maintain and deploy
  - Works with any language, any OS, any technology

Use cases: microservices architecture, lift-and-shift apps from on-premises to the AWS Cloud

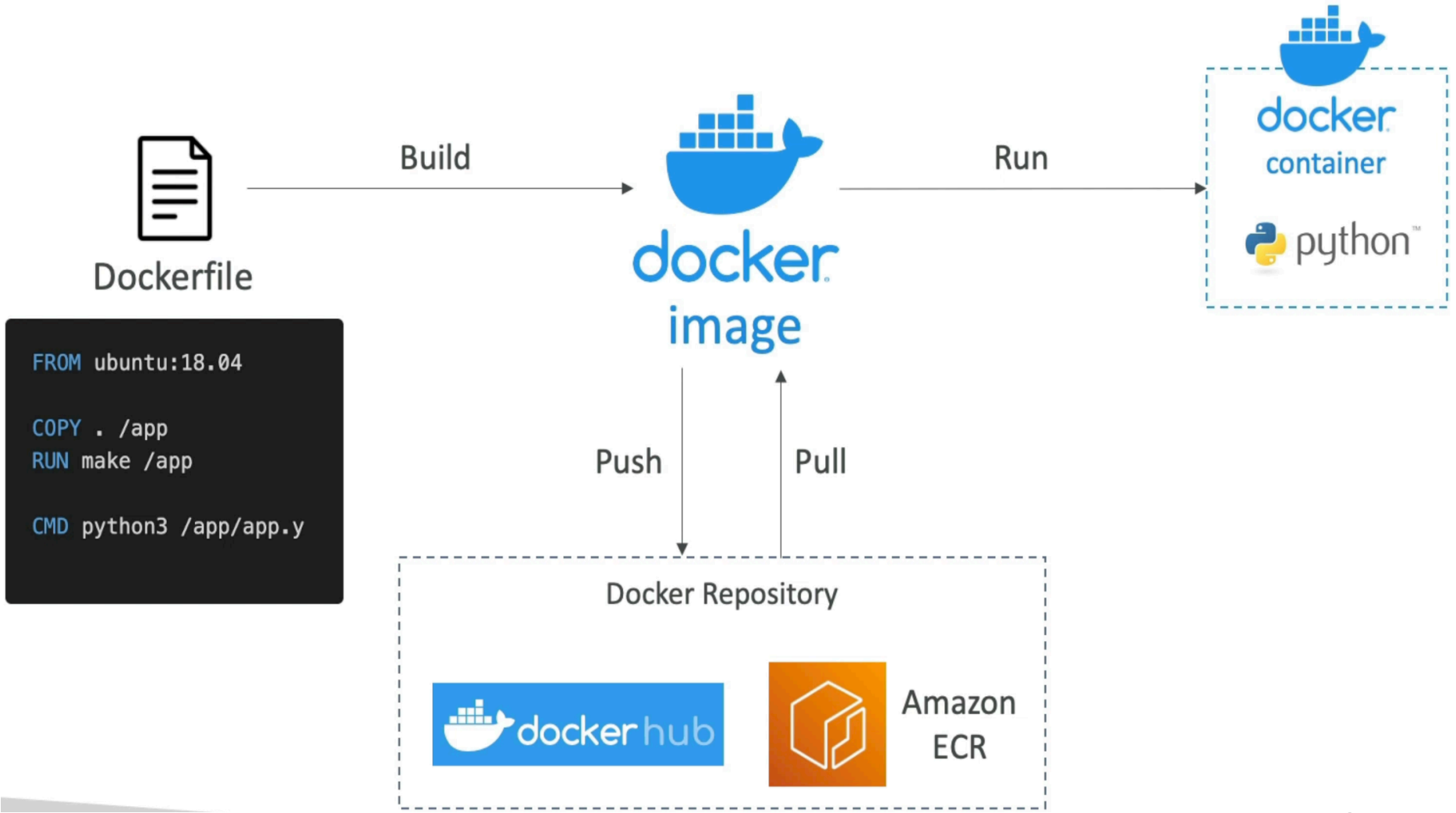
Where are Docker images stored?

- Docker images are stored in Docker Repositories
- Docker Hub (https://hub.docker.com)
  - Public repository
  - Find base images for many technologies or OS (eg. Ubuntu, MySQL etc)
- Amazon ECR (Amazon Elastic container Registry)
  - Private repository
  - Public repository (Amazon ECR Public Gallery https://gallery.ecr.aws)

Levels of abstraction and virtualization



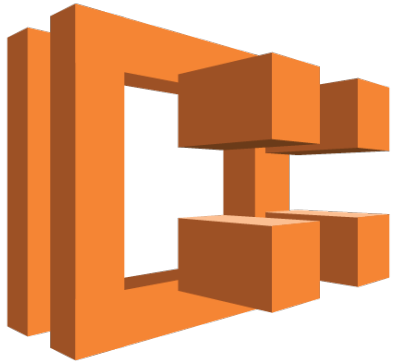
How Docker will work?



Why ECS & EKS

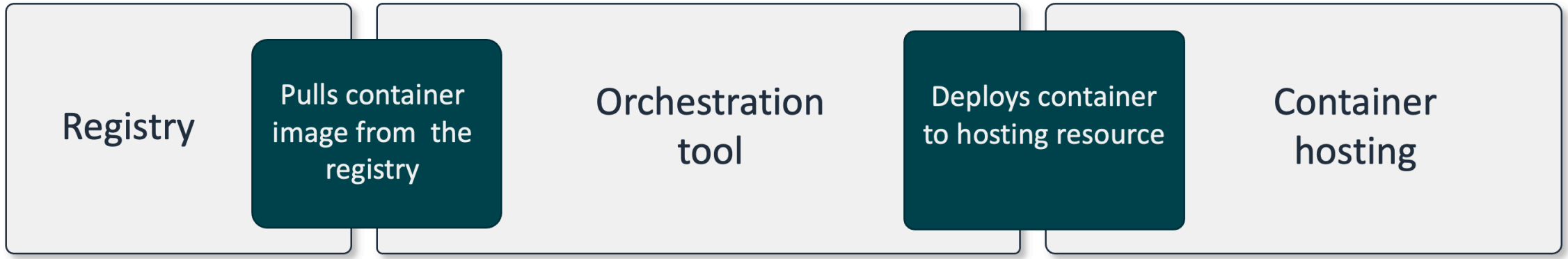
- Auto healing recreates unhealthy virtual machines (VMs) using the original instance template. Auto scaling performs automatic scaling based on the measured load

Amazon **EKS**



AWS ECS

Running containers on AWS



Amazon Elastic Container Registry (Amazon ECR)

Amazon Elastic Kubernetes Service (Amazon EKS)

Amazon Elastic Container Service (Amazon ECS)

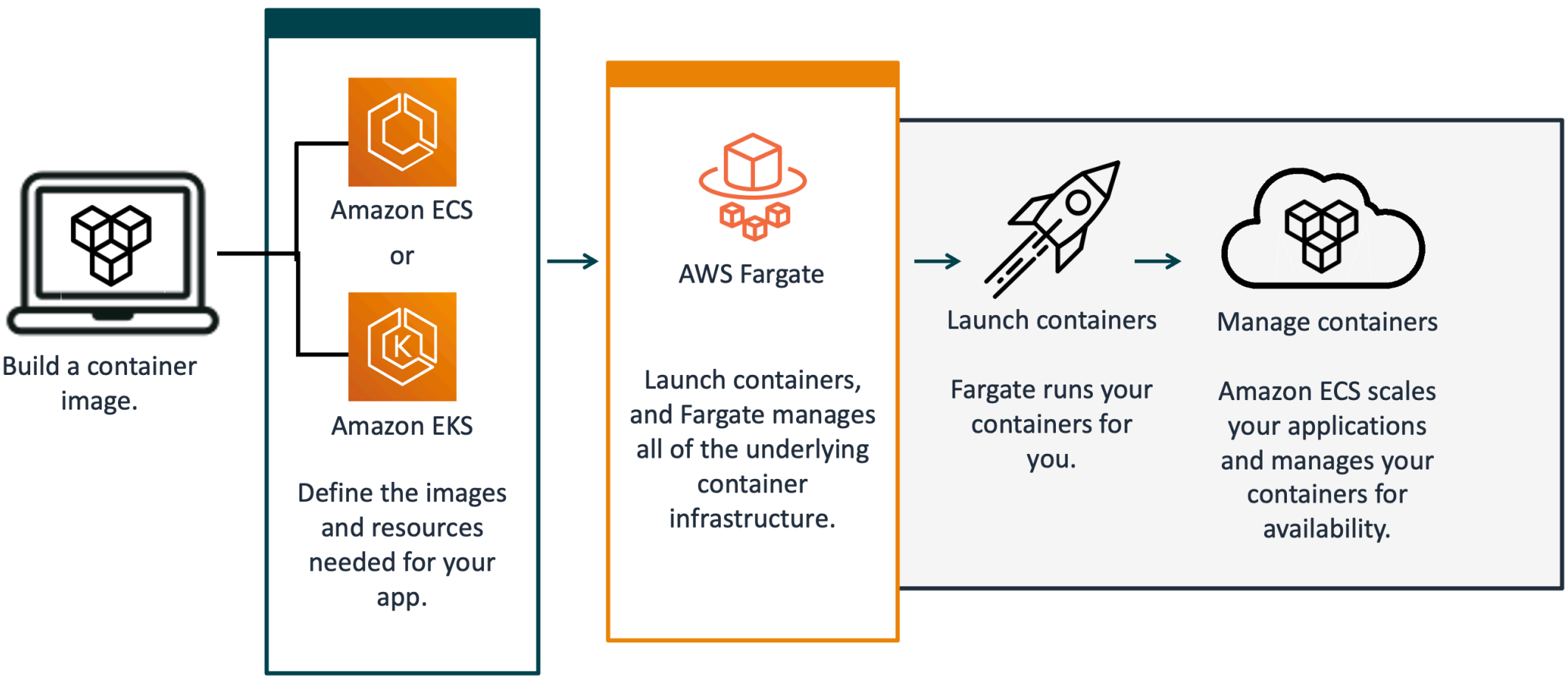
AWS Fargate

Amazon EC2

AWS Fargate

- AWS Fargate is a technology for Amazon ECS and Amazon EKS that you can use to run containers without having to manage servers or clusters. With Fargate, you no longer have to provision, configure, and scale clusters of VMs to run containers. This removes the need to choose server types, decide when to scale your clusters, or optimize cluster packing.
- Fargate eliminates the need for you to interact with or think about servers or clusters. With Fargate, you can focus on designing and building your applications instead of managing the infrastructure that runs them.

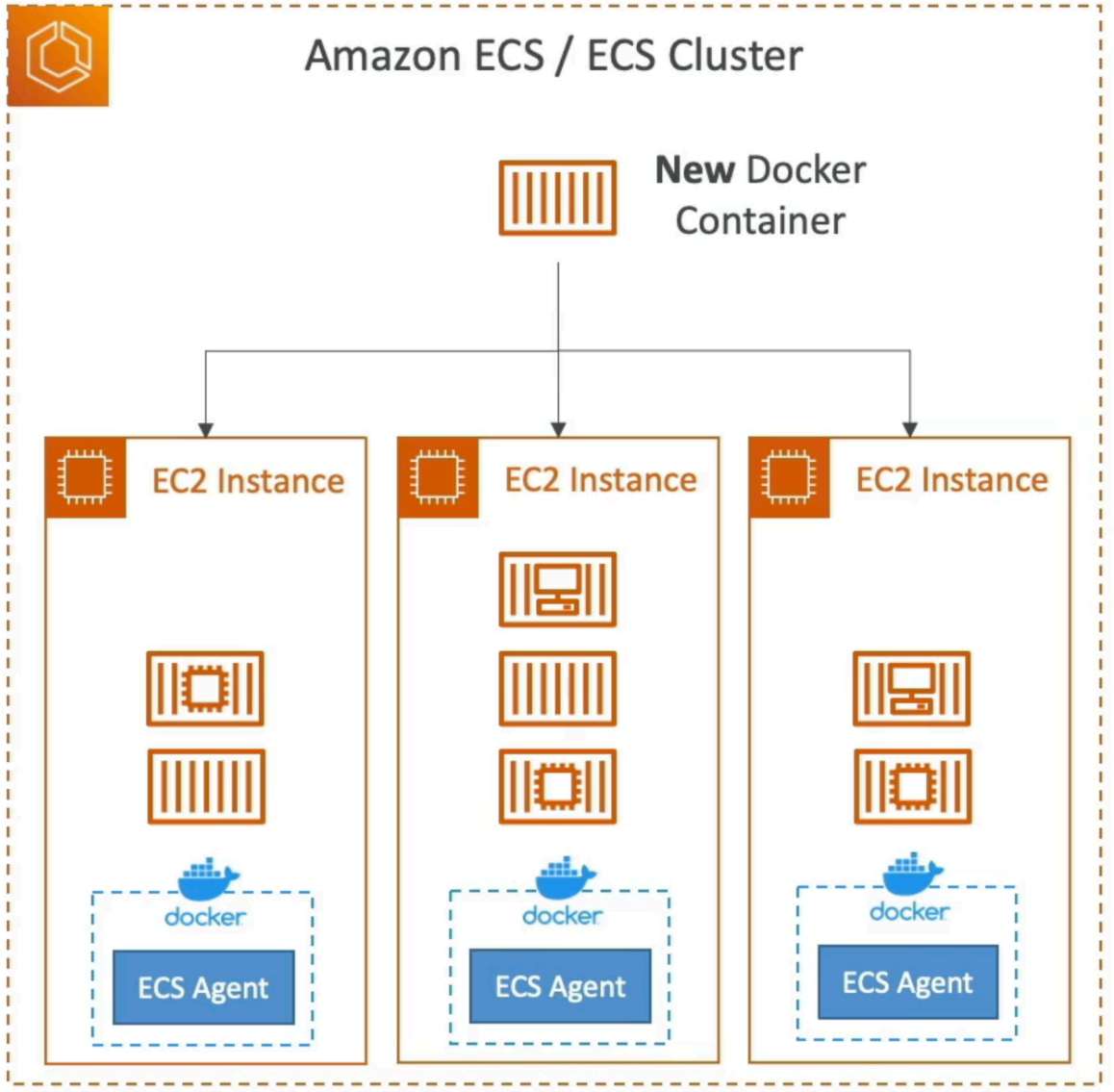
AWS Fargate


Build a container image.
Amazon ECS
or
Amazon EKS
Define the images and resources needed for your app.
AWS Fargate
Launch containers, and Fargate manages all of the underlying container infrastructure.
Launch containers
Fargate runs your containers for you.
Manage containers
Amazon ECS scales your applications and manages your containers for availability.

Amazon ECS – EC2 Launch Type

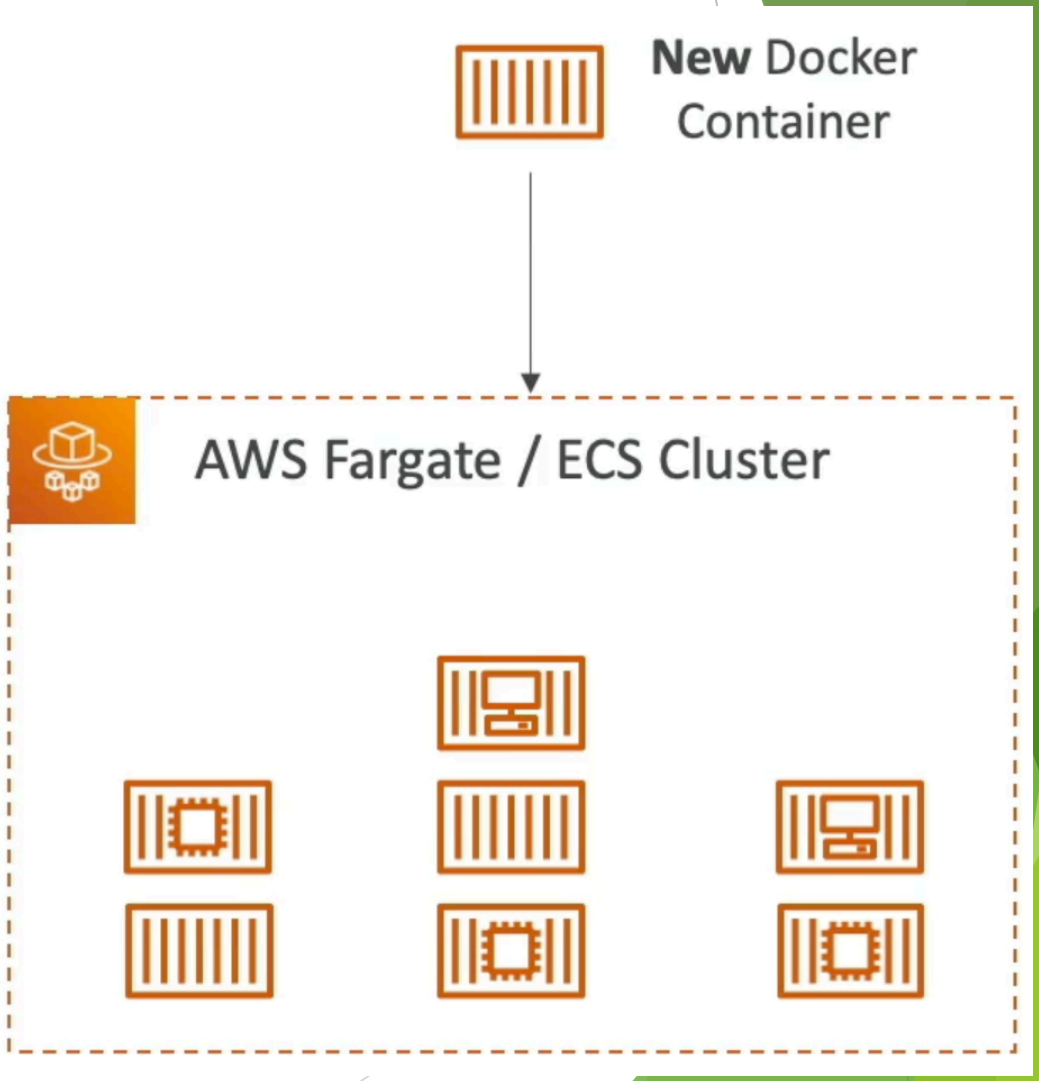
- ECS = Elastic Container Service
- Launch Docker containers on AWS = Launch ECS Tasks on ECS Cluster

- EC2 Launch type: you must provision & maintain the infrastructure (the EC2 instances)
- Each EC2 instance must run the ECS agent to register in the ECS Cluster
- AWS Takes care of starting / stopping containers



Amazon ECS – Fargate Launch Type

- Launch Docker containers on AWS
- You do not provision the infrastructure (no EC2 instances to manage)
- Its all serverless!
- You just create task definitions
- AWS just runs ECS tasks for you based on the CPU / RAM you need
- To scale, just increase the number of tasks. Simple – no more EC2 instances



Choosing AWS container services

Least effort → Most effort

Choose your orchestration service.	 Amazon ECS	 Amazon EKS
Choose your hosting type.	 AWS Fargate	 Amazon EC2