UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Facultat d'Informàtica de Barcelona

FIB

AWS Fargate

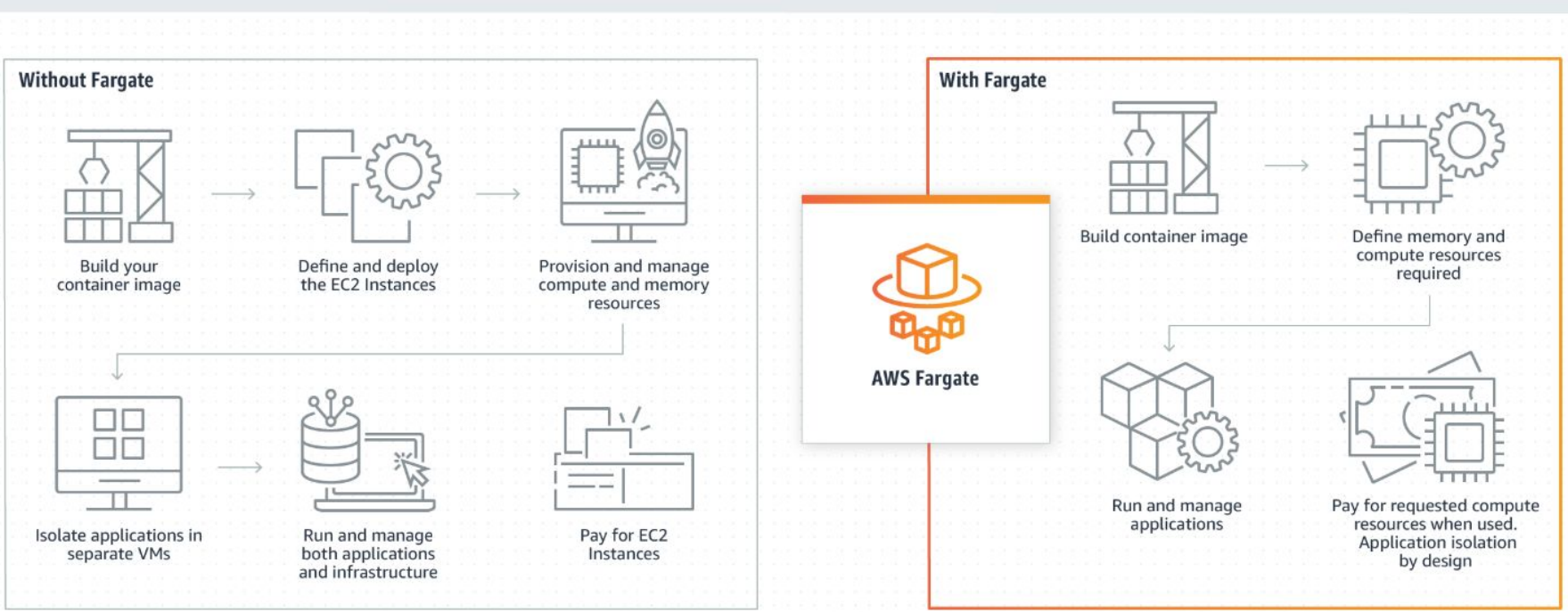
Team Members: Leonardo Egidati, Baptist Krugler, Matteo Salari, Xavier Lopez
Course: Cloud Computing & Big Data Analytics
Date: 12th May 2025

What is AWS Fargate?

A serverless compute engine for containers on AWS

- **No servers to manage** – forget EC2 nodes or Kubernetes worker sets.
- **Works with ECS & EKS** – choose AWS's native orchestrator (ECS) or managed Kubernetes (EKS).
- **Declarative resources** – you specify vCPU, memory, IAM role, and networking; Fargate provisions the rest.
- **Per-task micro-VM isolation** – every task / pod runs in its own lightweight VM for strong security boundaries.
- **Automatic scaling & patching** – availability, OS updates, and cluster capacity are handled by AWS.
- **Pay for what you use** – per-second billing on CPU + RAM, no charge when containers aren't running.

You build the container, AWS runs it.

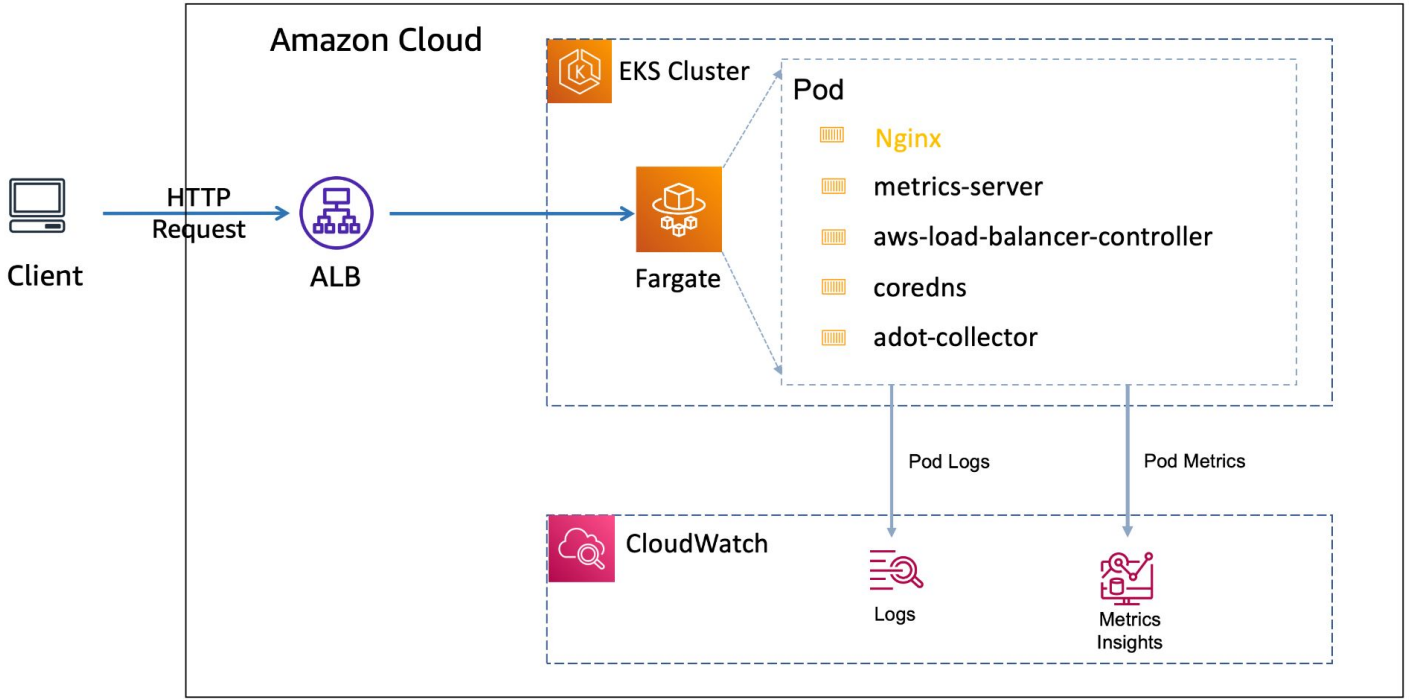

Without Fargate
Build your container image
Define and deploy the EC2 Instances
Provision and manage compute and memory resources
Isolate applications in separate VMs
Run and manage both applications and infrastructure
Pay for EC2 Instances
With Fargate
AWS Fargate
Build container image
Define memory and compute resources required
Run and manage applications
Pay for requested compute resources when used. Application isolation by design

When to use it

Ideal scenarios

- **Spiky or unpredictable traffic**
  Auto-scales in seconds without pre-provisioning capacity (e.g., flash-sale website, event-driven APIs).
- **Microservices & SaaS**
  Dozens of small, independent services where managing EC2 fleets would be overhead.
- **Batch & data pipelines**
  Short-lived ETL, log processing, or event-driven jobs—pay only while containers run.
- **Secure multi-tenant apps**
  Each tenant's task runs in its own micro-VM, reducing "noisy-neighbor" risk.
- **CI/CD preview environments**
  Spin up disposable review stacks on every pull request, tear them down after tests.
- **Regulated workloads needing host abstraction**
  Isolation handled by AWS, simplifying compliance checks (PCI-DSS, HIPAA, etc.).

Basic architecture



Quick comparison

AWS Fargate

Serverless: no EC2 hosts to provision, patch, or scale.

Granular billing: pay per second for the vCPU + RAM your tasks use.

Automatic scaling: tasks spin up or down with load, even to zero.

Strong isolation: each task/pod runs inside its own Firecracker micro-VM.

Low ops overhead: you declare image, CPU, memory, IAM role, and run.

EC2 / Self-managed Kubernetes

Full host control: pick instance types, install agents, tweak kernel.

Reserved/Spot savings: cheaper for steady 24 × 7 workloads when hosts stay busy.

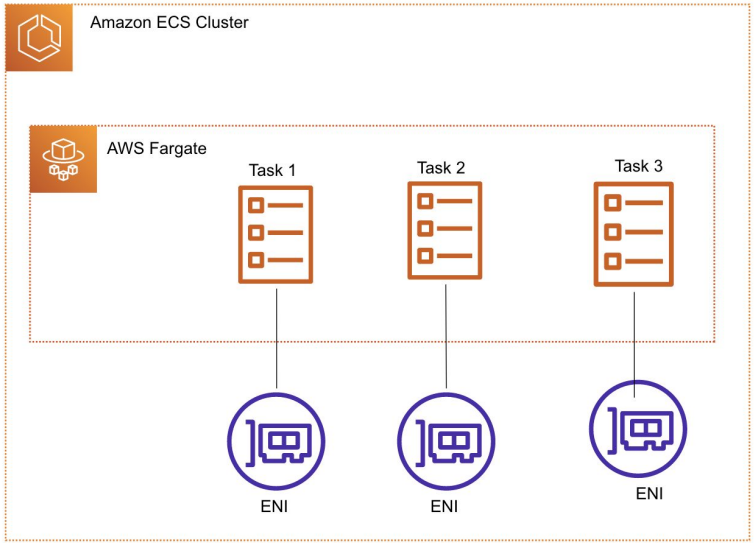
Manual capacity management: you size, scale, and patch the cluster.

Manual security management: you need to configure isolation and security manually.

Higher ops burden: upgrades, security fixes, autoscaling logic are on you.

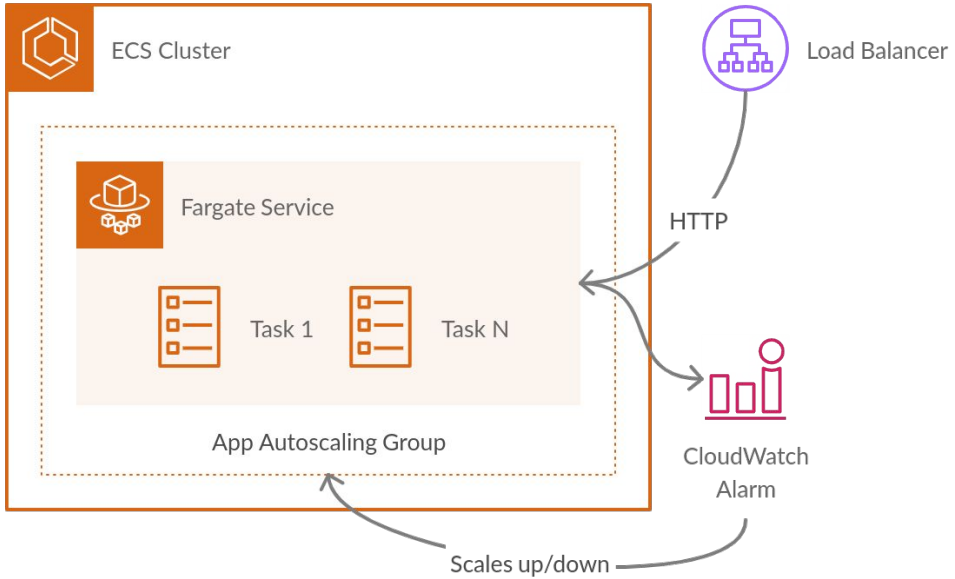
Part 1: Single task on Fargate

Goal: Understand the basic components of AWS Fargate in ECS. Define an ECS task, run a task in a cluster and understand the benefits of running a service instead of a task.



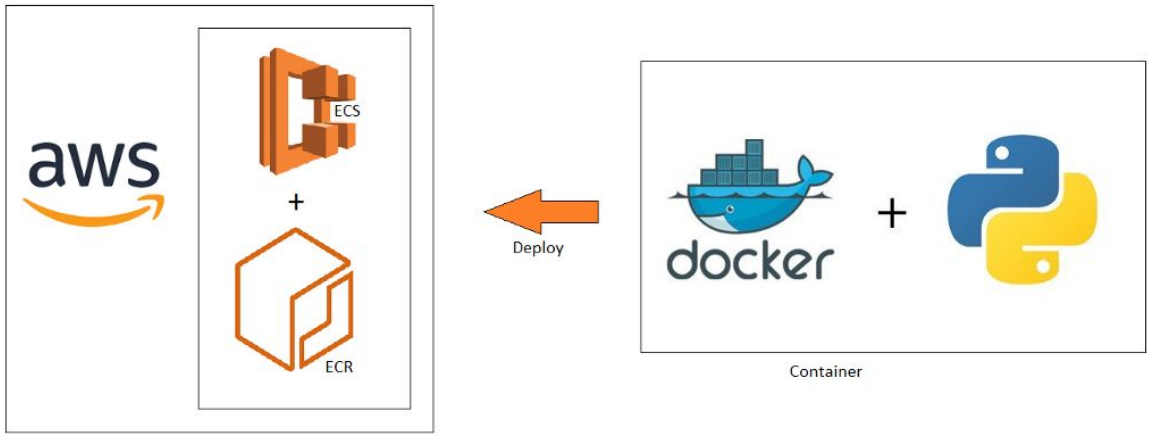
Part 2: Fargate+ALB+Auto-Scaling

Goal: Deploy a containerized web application using AWS Fargate and ECS, expose it via an Application Load Balancer (ALB), configure automatic scaling based on CPU usage, and safely clean up all resources.



Part 3: Custom image pipeline

Goal: Running a custom image in AWS ECS with Fargate. Upload a custom image into AWS ECR and defining a new task with or image. Understand how a cluster can have different services, one with each custom image.



Key takeaways & QA

Key points to remember

- **Serverless for containers** – Fargate removes host management; you focus on code.
- **Elastic by design** – tasks scale up, down, or to zero automatically.
- **Pay-only-when-running** – per-second billing on vCPU + RAM, no idle cost.
- **Strong isolation** – each task/pod gets its own micro-VM for security.
- **Best fit** – bursty microservices, short-lived jobs, rapid CI/CD previews.
- **Watch-outs** – higher cost for 24 × 7 loads, no GPU, limited host tweaks.
- **Decision rule** – choose Fargate when agility beats deep infrastructure control.