Práctica 4: Orquestación de Contenedores en AWS ECS

Introducción

En esta práctica, se ha implementado un servicio en **AWS Elastic Container Service (ECS)** utilizando **AWS Fargate** como infraestructura sin servidor (*serverless*). Se documenta el proceso de creación del clúster, la definición de tareas, el balanceador de carga y la creación del servicio.

Además, se ha experimentado un error durante la creación del clúster debido a un problema interno de AWS (**Error Code: ServerException, Status Code: 500**). La solución ha sido evaluar en el stack el problema y relanzar manualmente el proceso.

---

1. Creación del Clúster

Pasos realizados:

1. Acceder a **AWS ECS** y seleccionar **"Create Cluster"**.
2. Configurar:
    - **Nombre**: `alba-digi-cluster`
    - **Infraestructura**: `AWS Fargate (serverless)`
    - **Tags**: Key: `Name`, Value: `alba-digi-cluster`
3. Crear el clúster.
4. Debido al error **500 ServerException**, AWS CloudFormation reintentó la creación del clúster.
5. Una vez completada la creación, se verificó el estado en **CloudFormation > Events**, confirmando el evento **CREATE_COMPLETE** para `ECSCluster`.
6. Se capturó la pantalla del evento exitoso como evidencia (adjunta a este documento).

---

2. Creación de la Definición de Tarea (Task Definition)

1. Acceder a **AWS ECS** y seleccionar **"Task Definitions"**.
2. Crear una nueva definición de tarea y seleccionar **"Fargate"**.
3. Configurar:
    - **Nombre**: `alba-digi-task`
    - **Task Role**: `LabRole`
    - **Network Mode**: `awsvpc`
    - **Sistema Operativo**: `Linux`
    - **Task Execution Role**: `LabRole`

    - **CPU**: `1 vCPU`
    - **Memoria**: `3GB`
4. Agregar un contenedor:
    - **Nombre**: `alba-digi-api-rest`
    - **Imagen**: `public.ecr.aws/n0h7v2z5/alba-digi-api-rest:latest`
    - **Memoria (Soft Limit)**: `2GB`
    - **Puerto**: `5050` (TCP, HTTP)
5. Crear la definición de tarea y capturar una captura de pantalla.

---

3. Creación del Balanceador de Carga

1. Acceder a **AWS EC2** y seleccionar **"Load Balancers"**.
2. Crear un nuevo balanceador de carga **Application Load Balancer**.
3. Configurar:
    - **Nombre**: `alba-digi-load-balancer`
    - **Scheme**: `Internet-facing`
    - **IP Address Type**: `IPv4`
    - **Network Mapping**: Todas las zonas disponibles
    - **Security Groups**: Dejar el predeterminado
4. Crear un **Target Group**:
    - **Nombre**: `alba-digi-target-group`
    - **Protocolo**: `HTTP`, **Puerto**: `5050`
    - **Health Check**: `HTTP`, Path: `/alive`
5. Asociar el *Target Group* al balanceador y completar la configuración.
6. Capturar la pantalla del balanceador de carga creado.

---

4. Creación del Servicio

1. Acceder a **AWS ECS** y seleccionar el clúster `alba-digi-cluster`.
2. Crear un servicio y configurar:
    - **Launch Type**: `Fargate`
    - **Task Definition**: `alba-digi-task (latest)`
    - **Nombre del Servicio**: `alba-digi-service`
    - **Number of tasks**: `2`
    - **Load Balancer Type**: `Application Load Balancer`
    - **Health Check Grace Period**: `30`
    - **Container to Load Balance**: `alba-digi-api-rest:5050:5050`
3. Completar la configuración y capturar una imagen del servicio creado.

---

5. Prueba del Servicio

1. Copiar el **DNS Name** del balanceador de carga desde **EC2 > Load Balancers**.

En una terminal, ejecutar:
curl -v http://{DNS_NAME}/alive

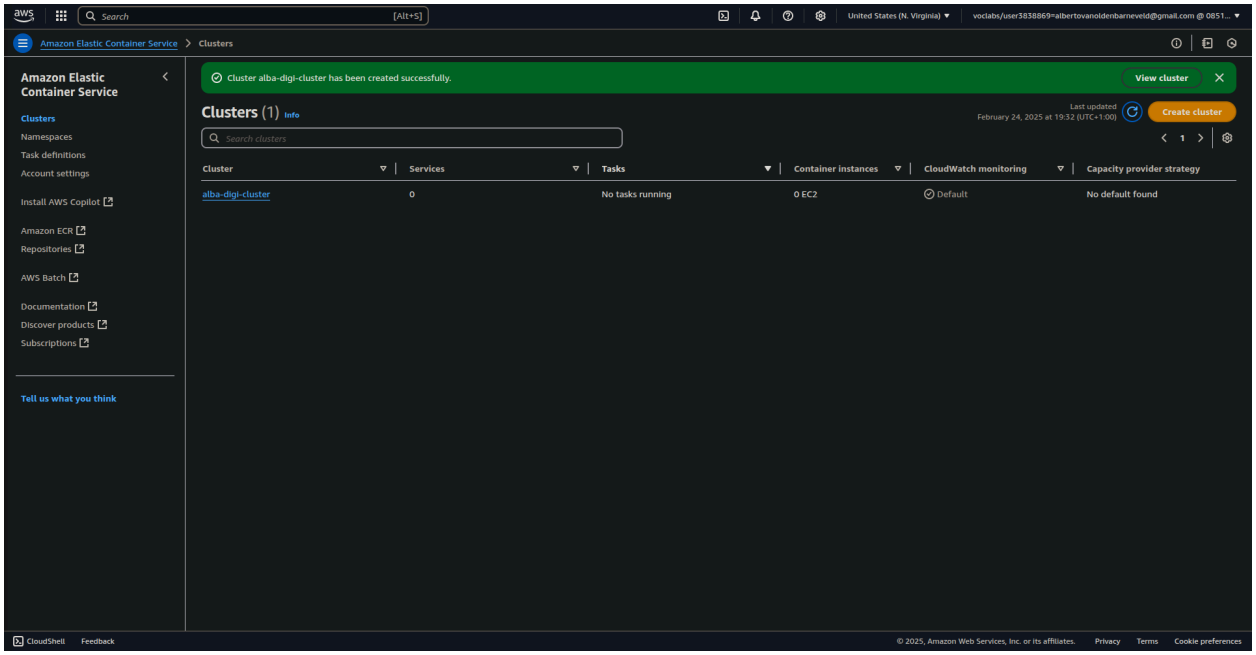
2. (Reemplazar `{DNS_NAME}` con la URL copiada).
3. Si la petición falla, ajustar las **Inbound Rules** del Security Group para permitir conexiones HTTP.
4. Volver a ejecutar `curl` y capturar la pantalla de la respuesta exitosa.

Conclusión

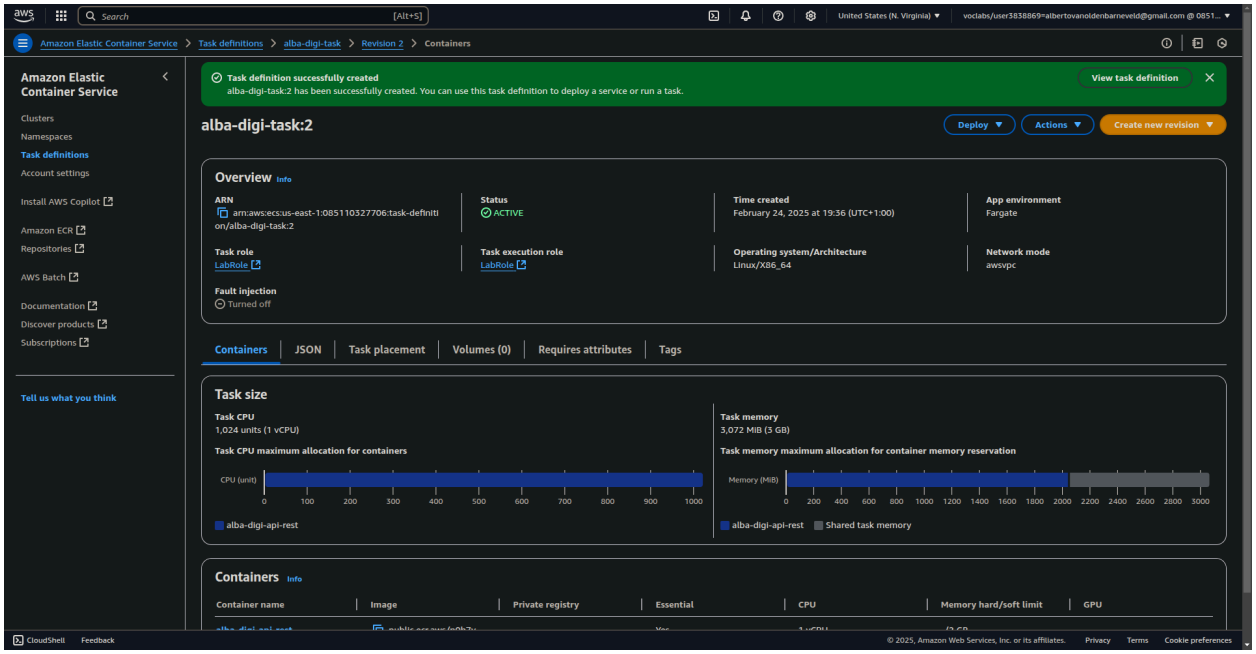
Se ha desplegado un servicio en **AWS ECS con Fargate**, configurando un **clúster**, una **definición de tarea**, un **balanceador de carga**, y un **servicio** con dos tareas. A pesar del error inicial en la creación del clúster, la solución fue esperar a que AWS CloudFormation reintentara el proceso.

Las capturas de pantalla tomadas a lo largo del proceso validan la correcta ejecución de cada etapa.

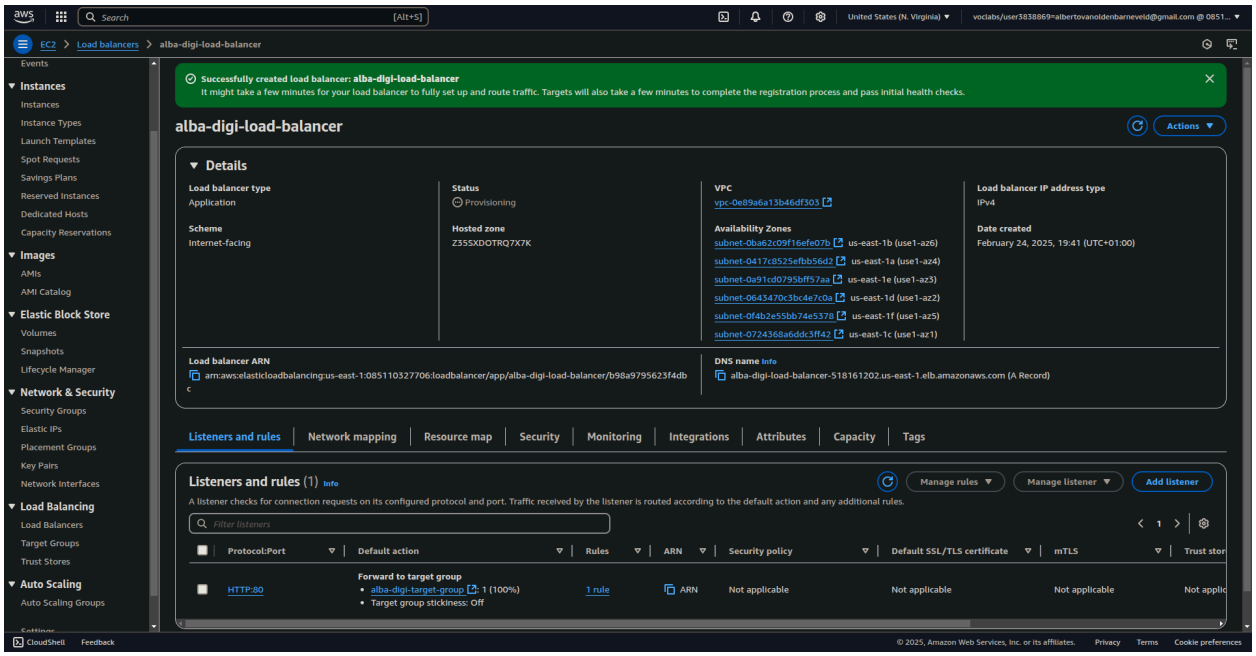
Captura de pantalla 1:



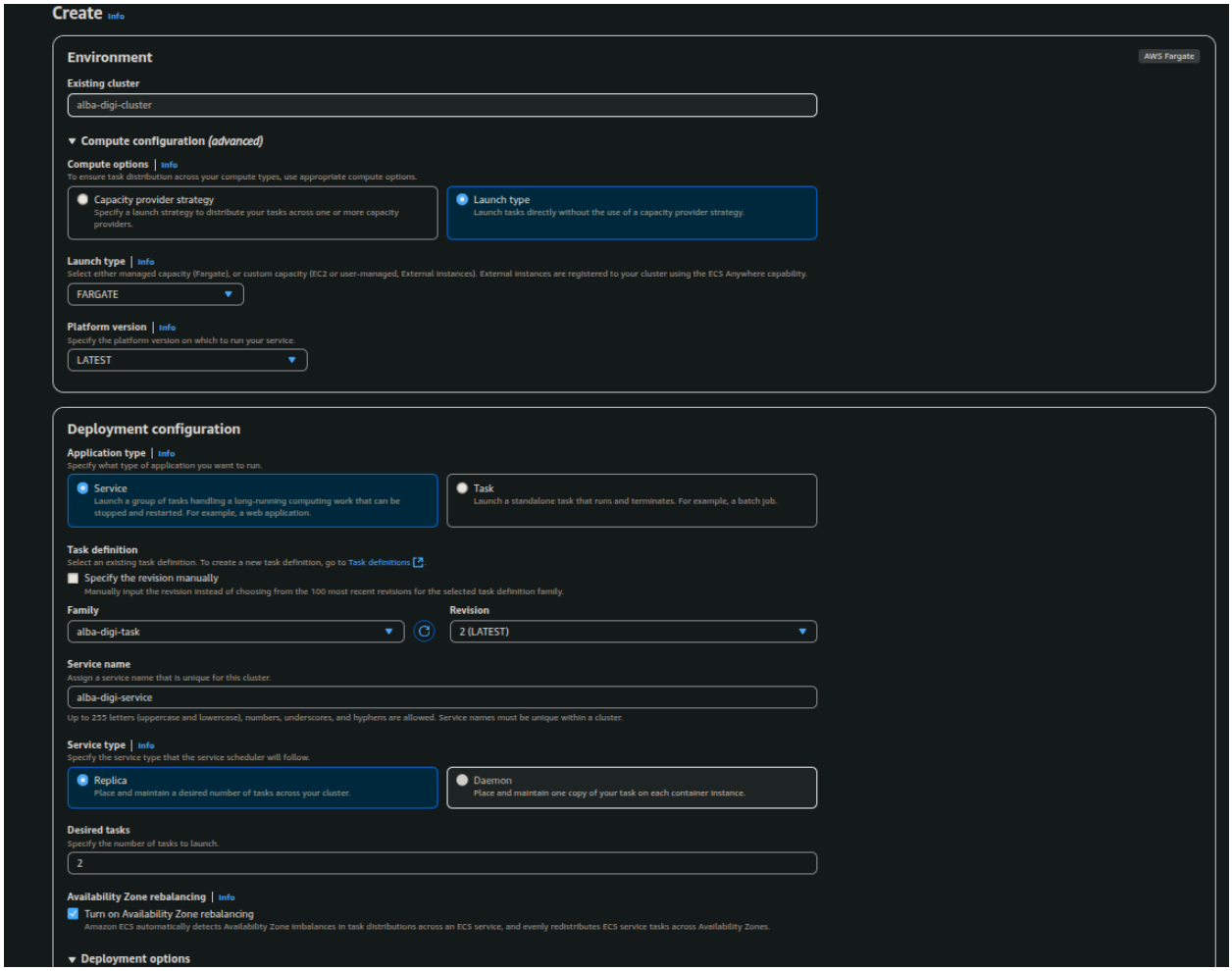
Captura de pantalla 2:

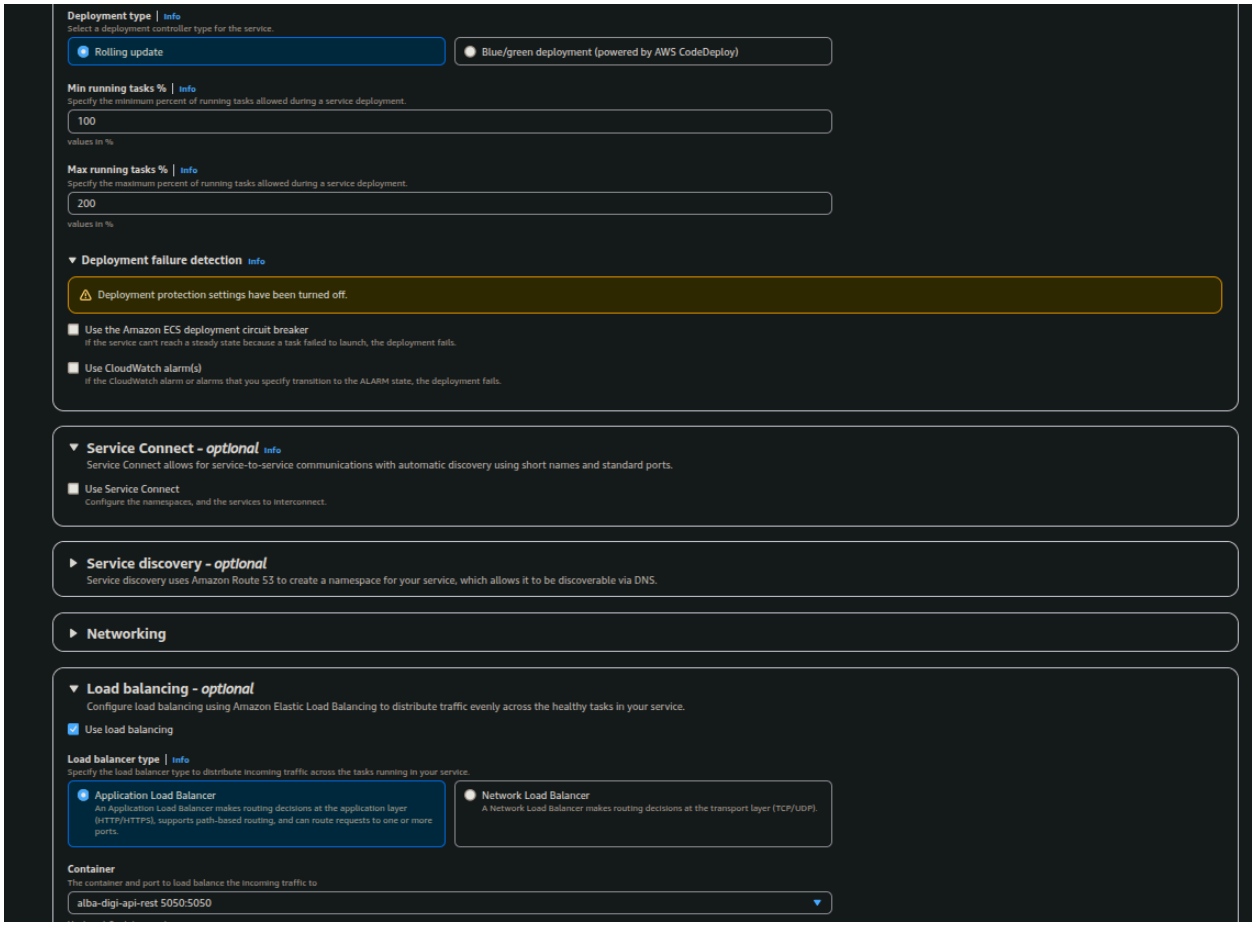


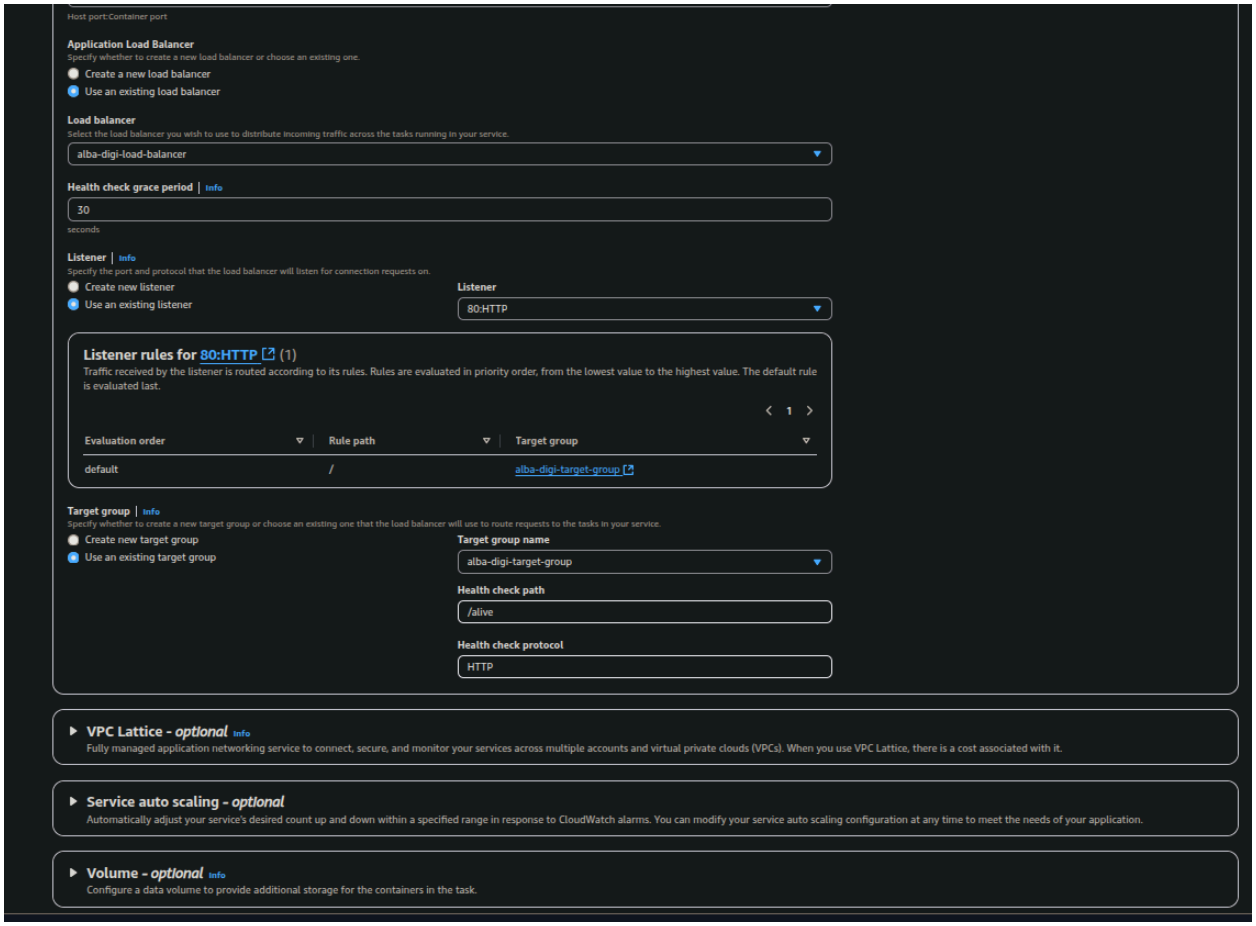
Captura de pantalla 3:



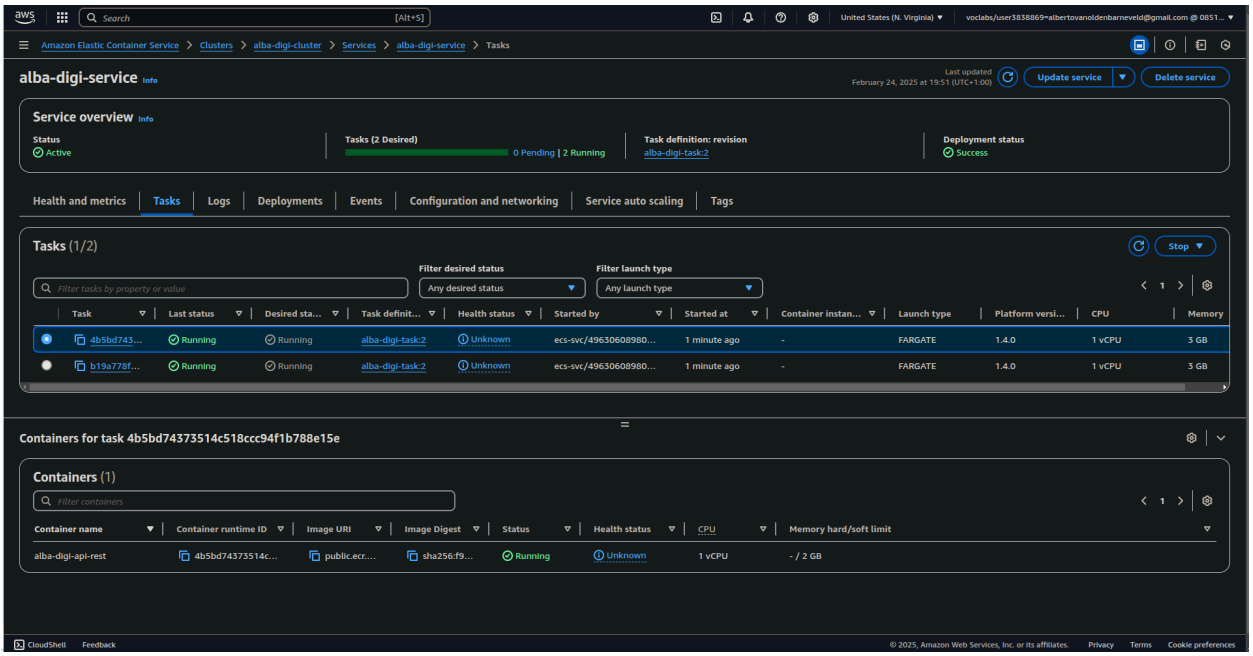
Capturas de pantalla 4:


Create Info
Environment
AWS Fargate
Existing cluster
alba-digi-cluster
Compute configuration (advanced)
Compute options | Info
To ensure task distribution across your compute types, use appropriate compute options.
Capacity provider strategy
Specify a launch strategy to distribute your tasks across one or more capacity providers.
Launch type
Launch tasks directly without the use of a capacity provider strategy.
Launch type | Info
Select either managed capacity (Fargate), or custom capacity (EC2 or user-managed, External instances). External instances are registered to your cluster using the ECS Anywhere capability.
FARGATE
Platform version | Info
Specify the platform version on which to run your service.
LATEST
Deployment configuration
Application type | Info
Specify what type of application you want to run.
Service
Launch a group of tasks handling a long-running computing work that can be stopped and restarted. For example, a web application.
Task
Launch a standalone task that runs and terminates. For example, a batch job.
Task definition
Select an existing task definition. To create a new task definition, go to Task definitions
Specify the revision manually
Manually input the revision instead of choosing from the 100 most recent revisions for the selected task definition family.
Family
alba-digi-task
Revision
2 (LATEST)
Service name
Assign a service name that is unique for this cluster.
alba-digi-service
Up to 255 letters (uppercase and lowercase), numbers, underscores, and hyphens are allowed. Service names must be unique within a cluster.
Service type | Info
Specify the service type that the service scheduler will follow.
Replica
Place and maintain a desired number of tasks across your cluster.
Daemon
Place and maintain one copy of your task on each container instance.
Desired tasks
Specify the number of tasks to launch.
2
Availability Zone rebalancing | Info
Turn on Availability Zone rebalancing
Amazon ECS automatically detects Availability Zone imbalances in task distributions across an ECS service, and evenly redistributes ECS service tasks across Availability Zones.
Deployment options

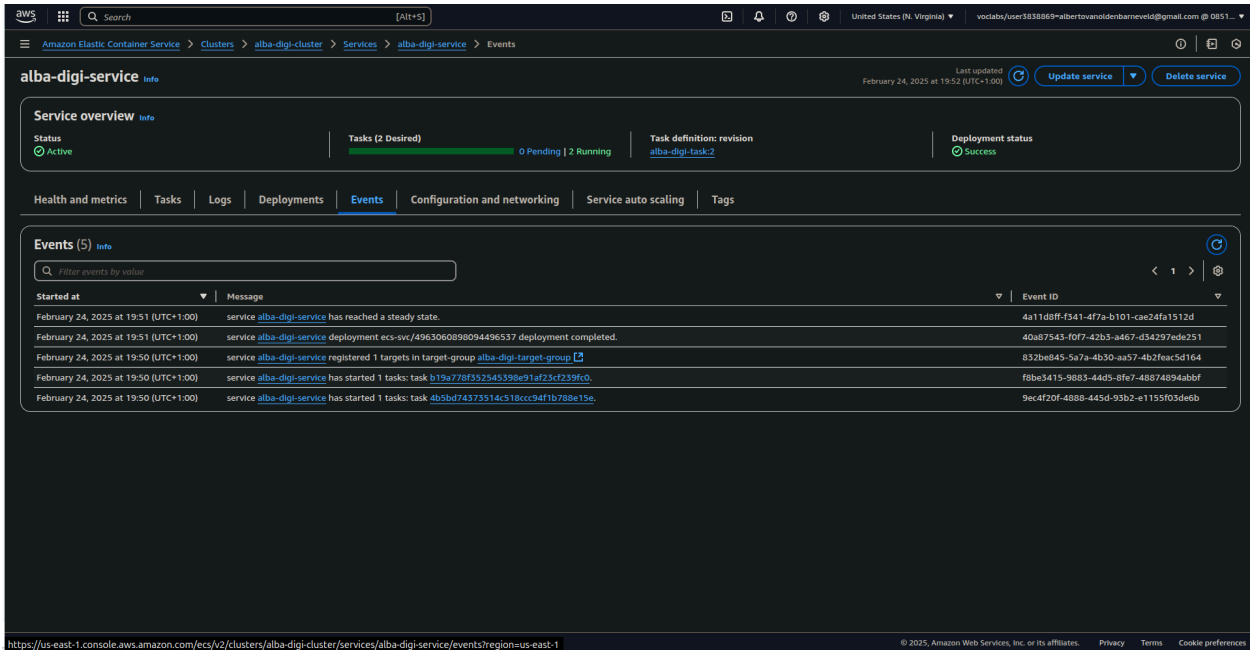

Deployment type | Info
Select a deployment controller type for the service.
Rolling update
Blue/green deployment (powered by AWS CodeDeploy)
Min running tasks % | Info
Specify the minimum percent of running tasks allowed during a service deployment.
100
values in %
Max running tasks % | Info
Specify the maximum percent of running tasks allowed during a service deployment.
200
values in %
Deployment failure detection Info
Deployment protection settings have been turned off.
Use the Amazon ECS deployment circuit breaker
If the service can't reach a steady state because a task failed to launch, the deployment fails.
Use CloudWatch alarm(s)
If the CloudWatch alarm or alarms that you specify transition to the ALARM state, the deployment fails.
Service Connect - optional Info
Service Connect allows for service-to-service communications with automatic discovery using short names and standard ports.
Use Service Connect
Configure the namespaces, and the services to interconnect.
Service discovery - optional
Service discovery uses Amazon Route 53 to create a namespace for your service, which allows it to be discoverable via DNS.
Networking
Load balancing - optional
Configure load balancing using Amazon Elastic Load Balancing to distribute traffic evenly across the healthy tasks in your service.
Use load balancing
Load balancer type | Info
Specify the load balancer type to distribute incoming traffic across the tasks running in your service.
Application Load Balancer
An Application Load Balancer makes routing decisions at the application layer (HTTP/HTTPS), supports path-based routing, and can route requests to one or more ports.
Network Load Balancer
A Network Load Balancer makes routing decisions at the transport layer (TCP/UDP).
Container
The container and port to load balance the incoming traffic to
alba-digi-api-rest 5050:5050

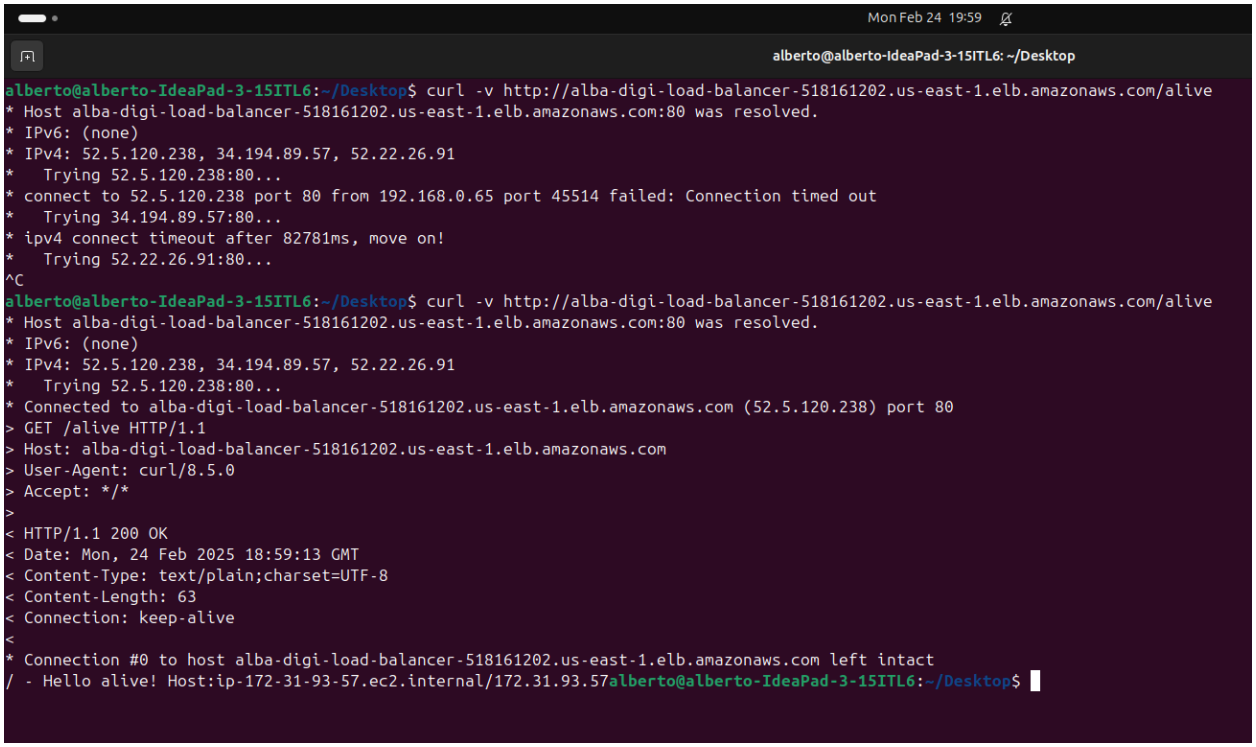


Capturas de pantalla 5:
Tasks del servicio



Eventos del servicio:


Search [Alt+S]
United States (N. Virginia)
Amazon Elastic Container Service > Clusters > alba-digi-cluster > Services > alba-digi-service > Events
alba-digi-service Info
Last updated February 24, 2025 at 19:52 (UTC+1:00)
Update service
Delete service
Service overview Info
Status
Active
Tasks (2 Desired)
0 Pending | 2 Running
Task definition: revision
alba-digi-task:2
Deployment status
Success
Health and metrics
Tasks
Logs
Deployments
Events
Configuration and networking
Service auto scaling
Tags
Events (5) Info
Filter events by value
Started at
Message
Event ID
February 24, 2025 at 19:51 (UTC+1:00)
service alba-digi-service has reached a steady state.
4a11d8ff-f341-4f7a-b101-cae24fa1512d
February 24, 2025 at 19:51 (UTC+1:00)
service alba-digi-service deployment ecs-svc/4963606898094496537 deployment completed.
40a87543-f0f7-42b3-a467-d34297ede251
February 24, 2025 at 19:50 (UTC+1:00)
service alba-digi-service registered 1 targets in target-group alba-digi-target-group
832be845-5a7a-4b30-aa57-4b2feac5d164
February 24, 2025 at 19:50 (UTC+1:00)
service alba-digi-service has started 1 tasks: task b19a778f355245598e91af25cf239fc0.
f8be3415-9883-44d5-8fe7-48874894abbf
February 24, 2025 at 19:50 (UTC+1:00)
service alba-digi-service has started 1 tasks: task 4b5bd74373514c518ccc94f1b788e15e.
9ec4f20f-4888-445d-93b2-e1155f03de6b
https://us-east-1.console.aws.amazon.com/ecs/v2/clusters/alba-digi-cluster/services/alba-digi-service/events?region=us-east-1
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences


Mon Feb 24 19:59
alberto@alberto-IdeaPad-3-15ITL6: ~/Desktop
alberto@alberto-IdeaPad-3-15ITL6:~/Desktop$ curl -v http://alba-digi-load-balancer-518161202.us-east-1.elb.amazonaws.com/alive
* Host alba-digi-load-balancer-518161202.us-east-1.elb.amazonaws.com:80 was resolved.
* IPv6: (none)
* IPv4: 52.5.120.238, 34.194.89.57, 52.22.26.91
*   Trying 52.5.120.238:80...
* connect to 52.5.120.238 port 80 from 192.168.0.65 port 45514 failed: Connection timed out
*   Trying 34.194.89.57:80...
* ipv4 connect timeout after 82781ms, move on!
*   Trying 52.22.26.91:80...
^C
alberto@alberto-IdeaPad-3-15ITL6:~/Desktop$ curl -v http://alba-digi-load-balancer-518161202.us-east-1.elb.amazonaws.com/alive
* Host alba-digi-load-balancer-518161202.us-east-1.elb.amazonaws.com:80 was resolved.
* IPv6: (none)
* IPv4: 52.5.120.238, 34.194.89.57, 52.22.26.91
*   Trying 52.5.120.238:80...
* Connected to alba-digi-load-balancer-518161202.us-east-1.elb.amazonaws.com (52.5.120.238) port 80
> GET /alive HTTP/1.1
> Host: alba-digi-load-balancer-518161202.us-east-1.elb.amazonaws.com
> User-Agent: curl/8.5.0
> Accept: */*
>
< HTTP/1.1 200 OK
< Date: Mon, 24 Feb 2025 18:59:13 GMT
< Content-Type: text/plain;charset=UTF-8
< Content-Length: 63
< Connection: keep-alive
<
* Connection #0 to host alba-digi-load-balancer-518161202.us-east-1.elb.amazonaws.com left intact
/ - Hello alive! Host:ip-172-31-93-57.ec2.internal/172.31.93.57alberto@alberto-IdeaPad-3-15ITL6:~/Desktop$