

Diplomatura en DevOps
Mundos E

Edición 2403

mundosE

**“Informe Desarrollo Práctico Integrador
(PIN FINAL)”**

Tutor:

- Guazzardo, Marcelo

Grupo 1:

- Cabral, Damian Esteban
- Huataquispe Poma, Arnold
- Gonzalez, Claudio
- Rico, Cristian
- Ferreira, Alexander

Proyecto de automatización de Infraestructura en AWS, creación y monitoreo de un Clúster EKS.

1. Introducción

Este proyecto, denominado PIN FINAL, ha sido diseñado para desplegar y gestionar un clúster de Kubernetes en AWS utilizando Elastic Kubernetes Service (EKS). Su propósito es integrar diferentes herramientas vistas durante la diplomatura de DevOps en MundosE. Utilizamos AWS para crear recursos como EC2, EKS, Load Balancer, etc... y herramientas de monitoreo para crear una infraestructura optimizada con enfoque DevOps.

El proyecto se compone de los siguientes aspectos principales:

- Aprovisionamiento de infraestructura: Se crea una instancia EC2 en AWS que funciona como máquina de administración, con herramientas esenciales instaladas en el User Data, como AWS CLI, kubectl, eksctl, Docker, necesarias para interactuar con los recursos de AWS, es como un Bastion.
- Creación y configuración de un clúster Kubernetes (EKS): Se utiliza eksctl para desplegar un clúster de Kubernetes administrado con tres nodos.
- Gestión de accesos y permisos: Se configura IAM y el configmap/aws-auth para administrar usuarios y accesos al clúster.
- Monitoreo y visualización de métricas: Se despliega Prometheus para recolectar métricas del clúster o sus pods, y Grafana para visualizarlas a través de dashboards personalizables.

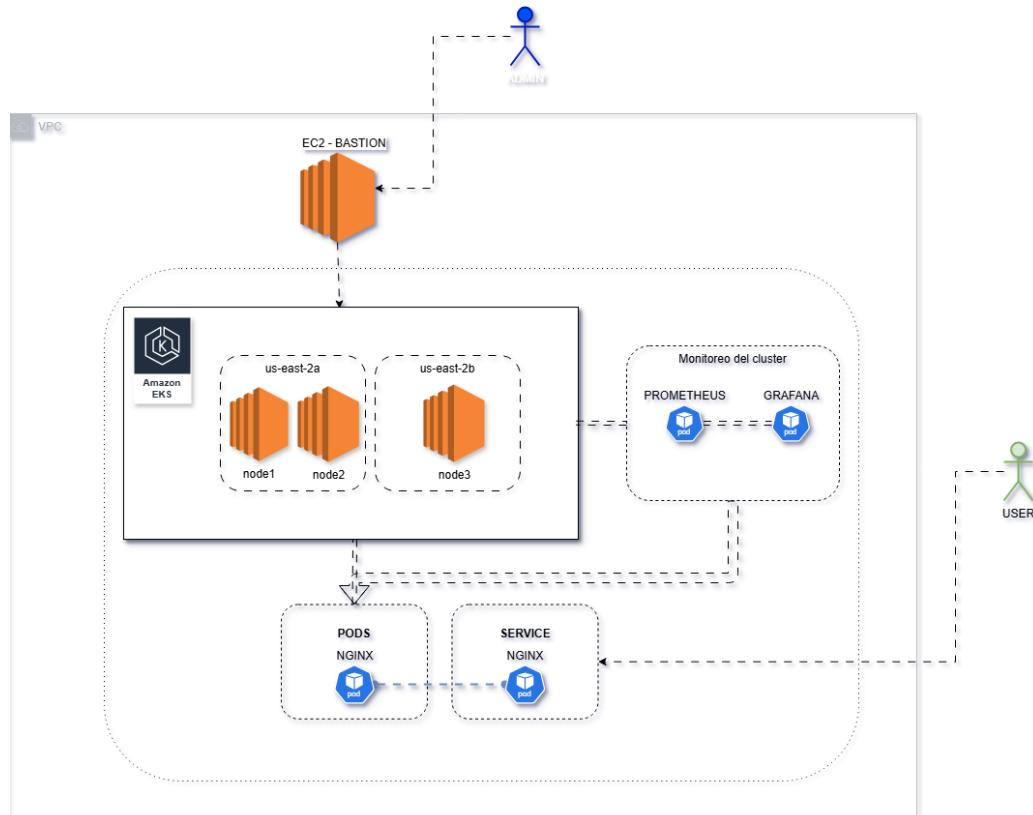
En términos generales, el proyecto automatiza la creación y gestión de un entorno Kubernetes en AWS, integrando monitoreo y logging centralizado con herramientas modernas de DevOps.

2. Objetivos

El proyecto tiene como objetivo general del proyecto es aprender y poner en práctica diversas tecnologías y herramientas a través de un laboratorio. Para ello, se implementa una infraestructura en AWS, de manera automatizada y eficiente. Para ello, se desplegará un clúster EKS con eksctl, configurando los nodos de trabajo y los permisos de acceso adecuados. Además, se utilizarán scripts para instalar herramientas clave como AWS CLI, Docker, Helm y Terraform en una instancia EC2, facilitando la gestión de la infraestructura.

Para recolectar las métricas de los PODs, se habilitará un sistema centralizado con Prometheus y Grafana para la recopilación y visualización de métricas. Se realizarán pruebas para verificar la correcta operación del clúster. Finalmente, se establecerá un proceso de limpieza automatizado para eliminar de forma segura los recursos cuando ya no sean necesarios, asegurando una administración eficiente y escalable del entorno.

3. Topología general



Topología general del proyecto DevOps. Representa cómo los diferentes componentes interactúan entre sí:

- **EC2 Instance:** Es la máquina base donde se instalan herramientas necesarias como AWS CLI, kubectl, Docker, Helm y Terraform.
- **AWS CLI:** Se usa para interactuar con AWS y configurar el clúster EKS.
- **EKS Cluster:** Contiene los nodos del clúster donde se ejecutan los workloads.
- **Worker Nodes:** Son los nodos donde se despliegan los contenedores de Kubernetes.
- **Prometheus:** Monitorea el rendimiento del clúster y sus aplicaciones (Nginx).
- **Grafana:** Visualiza los datos de Prometheus mediante dashboards.

4. Descripción del Proyecto

Este proyecto tiene como objetivo la creación y configuración de un clúster Kubernetes en AWS EKS, proporcionando una solución integral para la gestión de contenedores con una infraestructura robusta de monitoreo y logging. Mediante la combinación de Prometheus y Grafana, se garantiza la observabilidad del sistema, facilitando la detección y solución de problemas.

La implementación se desarrolla en seis fases estructuradas, asegurando un despliegue organizado y eficiente, desde la preparación de la infraestructura hasta la posible eliminación de recursos.

4.1 FASE 0: prerequisitos

(Paso 1) Se creara un usuario en AWS

ID : xlexferr02aws
Ur : DevOps_2403_G1
Pass: *****

Specify user details

User details

User name: DevOps_2403_G1

Provide user access to the AWS Management Console - optional

If you're providing console access to a person, it's a best practice [to manage their access in IAM Identity Center](#).

Are you providing console access to a person?

Specify a user in Identity Center - Recommended

We recommend that you use Identity Center to provide console access to a person. With Identity Center, you can centrally manage user access to their AWS accounts and cloud applications.

I want to create an IAM user

We recommend that you create IAM users only if you need to enable programmatic access through access keys, service-specific credentials for AWS CodeCommit or Amazon Kinesis, or a backup credential for emergency account access.

Console password

Autogenerated password

You can view the password after you create the user.

Custom password

Enter a custom password for the user.

* Must be at least 8 characters long.
* Must include at least three of the following mix of character types: uppercase letters (A-Z), lowercase letters (a-z), numbers (0-9), and symbols (! @ # \$ % ^ & * () _ + - (hyphen) = { } { }) ^

Show password

Users must create a new password at next sign-in - Recommended

Users automatically get the [IAMUserChangePassword](#) policy to allow them to change their own password.

If you are creating programmatic access through access keys or service-specific credentials for AWS CodeCommit or Amazon Kinesis, you can generate them after you create this IAM user.

[Learn more](#)

(Paso 2) Se creara un par de claves SSH en la consola de AWS, para acceder a la EC2

Name: pin2403ec2g1

EC2 > Key pairs > Create key pair

Create key pair Info

Key pair
A key pair, consisting of a private key and a public key, is a set of security credentials that you use to prove your identity when connecting to an instance.

Name
pin2403ec2g1
The name can include up to 255 ASCII characters. It can't include leading or trailing spaces.

Key pair type Info
 RSA ED25519

Private key file format
 .pem For use with OpenSSH
 .pk For use with PuTTY

Tags - optional
No tags associated with the resource.
[Add new tag](#)
You can add up to 50 more tags.

Se crea Access Key – IAM – ur: DevOps_2403_G1

Access key best practices & alternatives Info

Avoid using long-term credentials like access keys to improve your security. Consider the following use cases and alternatives.

Use case
 Command Line Interface (CLI)
You plan to use this access key to enable the AWS CLI to access your AWS account.

tag: AK_pin2403iam_g1

Set description tag - optional Info

The description for this access key will be attached to this user as a tag and shown alongside the access key.

Description tag value
Describe the purpose of this access key and where it will be used. A good description will help you rotate this access key confidently
AK_pin2403iam_g1
Maximum 256 characters. Allowed characters are letters, numbers, spaces representable in UTF-8, and: _ . : / = + - @

Access key: ****

Secret access key: ****

Retrieve access keys Info

Access key
If you lose or forget your secret access key, you cannot retrieve it. Instead, create a new access key and make the old key inactive.

Access key	Secret access key
AKIA23WHUEI	IVKcO8ei0O9KhwcuL

[Hide](#)

4.2 FASE 1: Preparación de la infraestructura

(Paso 1) Se crea una instancia EC2 en AWS con Ubuntu 22.04.

Instancia name: Admin-K8s-Instance-G1

The screenshot shows the 'Launch an instance' wizard. In the 'Name and tags' section, the instance name is set to 'Admin-K8s-Instance-G1'. Below it, the 'Application and OS Images (Amazon Machine Image)' section is expanded, showing various AMI options. The 'Ubuntu' option is selected, showing its details: 'Ubuntu Server 24.04 LTS (HVM), SSD Volume Type' with AMI ID 'ami-04b4f1a9cf54c11d0'. A note indicates it's 'Free tier eligible'. On the right, there's a link to 'Browse more AMIs'.

Tipo: t2.medium (2 vCPU, 4GB RAM)

The screenshot shows the 'Launch an instance' wizard. In the 'Description' section, it details the selected AMI: 'Ubuntu Server 24.04 LTS (HVM), SSD Volume Type' with AMI ID 'ami-04b4f1a9cf54c11d0'. It also notes 'Free tier eligible'. In the 'Architecture' section, the choice is '64-bit (x86)'. The 'Username' is set to 'ubuntu' and is marked as a 'Verified provider'. In the 'Instance type' section, 't2.medium' is selected, showing its details: 'Family: t2 - 2 vCPU - 4 GiB Memory - Current generation: true'. It lists pricing for On-Demand and On-Demand SUSE base pricing. A note at the bottom states 'Additional costs apply for AMIs with pre-installed software'.

Key pair

You can use a key pair to securely connect to your instance. Ensure that you have access to the selected key pair before you launch the instance.

Key pair name - required

pin2403ec2g1

Create new key pair

Network settings

Network

vpc-0877038e4fc1e1c05 | JAFC-default

Subnet

No preference (Default subnet in any availability zone)

Auto-assign public IP

Enable

Additional charges apply when outside of free tier allowance

Firewall (security groups)

A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

Create security group **Select existing security group**

We'll create a new security group called 'launch-wizard-2' with the following rules:

Allow SSH traffic from
Helps you connect to your instance Anywhere
0.0.0.0/0

Allow HTTPS traffic from the internet
To set up an endpoint, for example when creating a web server

Allow HTTP traffic from the internet
To set up an endpoint, for example when creating a web server

Storage

Storage (volumes)

EBS Volumes

Volume 1 (AMI Root) (Custom)

Storage type | Info EBS

Device name - required | Info /dev/sda1

Snapshot | Info snap-00cdccb5239896f89

Size (GiB) | Info 25

Volume type | Info gp3

IOPS | Info 3000

Delete on termination | Info Yes

Encrypted | Info Not encrypted

KMS key | Info Select

KMS keys are only applicable when encryption is set on this volume.

Throughput | Info 125

User data

User data - optional | [Info](#)
 Upload a file with your user data or enter it in the field.

[Choose file](#)

```
#!/bin/bash
echo "Instalando AWS CLI"
sudo apt update -y && sudo apt upgrade -y
curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"
sudo apt install -y unzip
unzip awscliv2.zip
sudo ./aws/install

# Instalar kubectl
echo "Instalando kubectl..."
curl -fsSL -o /usr/local/bin/kubectl https://dl.k8s.io/release/v1.29.0/bin/linux/amd64/kubectl
chmod +x /usr/local/bin/kubectl

# Instalar eksctl
echo "Instalando eksctl..."
curl --silent --location
"https://github.com/weaveworks/eksctl/releases/latest/download/eksctl_Linux_amd64.tar.gz" | tar zx
-C /tmp
sudo mv /tmp/eksctl /usr/local/bin

# Instalar Docker
echo "Instalando Docker..."
sudo apt install -y docker.io
sudo usermod -aG docker $USER
sudo systemctl enable docker
sudo systemctl start docker

# Instalar Docker Compose
echo "Instalando Docker Compose..."
curl -fsSL -o /usr/local/bin/docker-compose
https://github.com/docker/compose/releases/latest/download/docker-compose-linux-x86_64
chmod +x /usr/local/bin/docker-compose

echo "Instalando Helm..."
curl -fsSL https://raw.githubusercontent.com/helm/helm/main/scripts/get-helm-3 | bash

# Instalar Terraform
echo "Instalando Terraform..."
sudo apt install -y gnupg software-properties-common
wget -O https://apt.releases.hashicorp.com/gpg | sudo gpg --dearmor -o
/usr/share/keyrings/hashicorp-archive-keyring.gpg
echo "deb [signed-by=/usr/share/keyrings/hashicorp-archive-keyring.gpg]
https://apt.releases.hashicorp.com ${{lsb_release -cs}} main" | sudo tee
/etc/apt/sources.list.d/hashicorp.list
sudo apt update && sudo apt install -y terraform
```

▼ Summary

Number of instances | [Info](#)

1

Software Image (AMI)

Canonical, Ubuntu, 24.04, amd64 noble image
 ami-04b4f1a9cf54c11d0

Virtual server type (instance type)

t2.medium

Firewall (security group)

New security group

Storage (volumes)

1 volume(s) - 25 GiB

ⓘ Free tier: In your first year of opening an AWS account, you get 750 hours per month of t2.micro instance usage (or t3.micro where t2.micro isn't available) when used with free tier AMIs, 750 hours per month of public IPv4 address usage, 30 GiB of EBS storage, 2 million I/Os, 1 GB of snapshots, and 100 GB of bandwidth to the internet.

[Cancel](#) [Launch instance](#) [Preview code](#)

ⓘ Success
 Successfully initiated launch of instance (i-09f34eba96632dbed)

(Paso 2) Se instalan herramientas necesarias en la instancia EC2, como AWS CLI, kubectl, eksctl, Docker, Helm y Terraform (ec2_user_data.sh).

(Paso 3) Se prueban credenciales AWS en la instancia EC2 para poder operar con AWS EKS.

Esta IP, cambia cada vez que se apaga la máquina, para que quede fija la misma IP, deberíamos utilizar una ELASTIC IP

IP Public: 18.117.106.180

ssh -i pin2403ec2g1.pem ubuntu@18.117.106.180

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS ESP-IDF SERIAL MONITOR AZURE POSTMAN CONSOLE
PS D:\Estudio\DevOps\Proyecto integrador\Proyecto final\Llaves User> ssh -i pin2403ec2g1.pem ubuntu@54.227.189.188
The authenticity of host '54.227.189.188' (54.227.189.188) can't be established.
ED25519 key fingerprint is SHA256:Hc3ZDF/cxDiWrTtx1BJkwADNeBpBQK1isetkNUuzj8.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '54.227.189.188' (ED25519) to the list of known hosts.
Welcome to Ubuntu 24.04.2 LTS (GNU/Linux 6.8.0-1021-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/pro

System information as of Thu Feb 27 20:58:13 UTC 2025

System load:  0.0          Processes:           117
Usage of /:   14.6% of 23.17GB   Users logged in:     0
Memory usage: 11%
Swap usage:   0%
IPv4 address for enX0: 172.31.31.254

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

1 additional security update can be applied with ESM Apps.
Learn more about enabling ESM Apps service at https://ubuntu.com/esm

*** System restart required ***

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

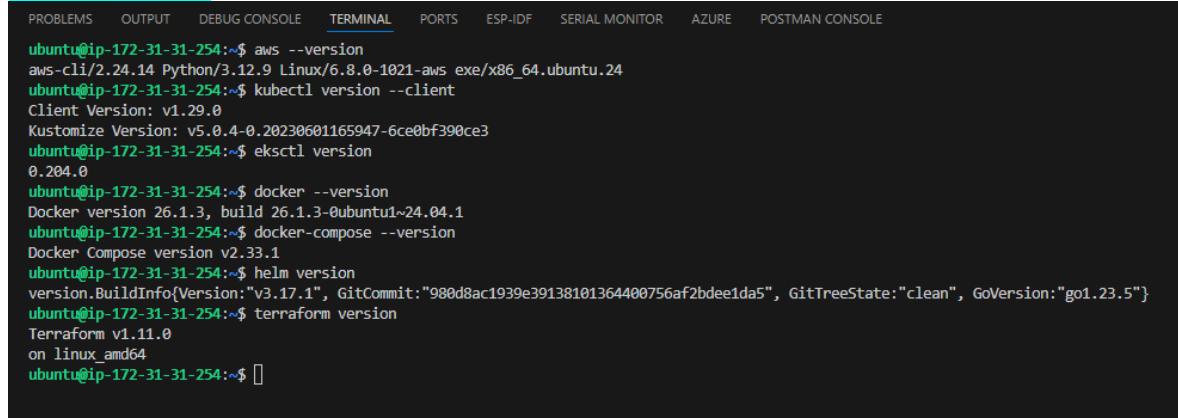
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@ip-172-31-31-254:~$ 

```

Validación de los servicios instalados

```
aws --version
kubectl version --client
eksctl version
docker --version
docker-compose --version
helm version
terraform version
```



```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS ESP-IDF SERIAL MONITOR AZURE POSTMAN CONSOLE
ubuntu@ip-172-31-2-56:~$ aws --version
aws-cli/2.24.14 Python/3.12.9 Linux/6.8.0-1021-aws exe/x86_64.ubuntu.24
ubuntu@ip-172-31-2-56:~$ kubectl version --client
Client Version: v1.29.0
Kustomize Version: v5.0.4-0.20230601165947-6ce0bf390ce3
ubuntu@ip-172-31-2-56:~$ eksctl version
0.204.0
ubuntu@ip-172-31-2-56:~$ docker --version
Docker version 26.1.3, build 26.1.3-0ubuntu1~24.04.1
ubuntu@ip-172-31-2-56:~$ docker-compose --version
Docker Compose version v2.33.1
ubuntu@ip-172-31-2-56:~$ helm version
version.BuildInfo{Version:"v3.17.1", GitCommit:"980d8ac1939e39138101364400756af2bdee1da5", GitTreeState:"clean", GoVersion:"go1.23.5"}
ubuntu@ip-172-31-2-56:~$ terraform version
Terraform v1.11.0
on linux_amd64
ubuntu@ip-172-31-2-56:~$
```

(Paso 4) Desde la EC2, se configuran las credenciales para AWS

```
aws configure
aws sts get-caller-identity
```



```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS ESP-IDF SERIAL MONITOR AZURE POSTMAN CONSOLE
ubuntu@ip-172-31-2-56:~$ aws configure
AWS Access Key ID [None]: AKIA
AWS Secret Access Key [None]:
Default region name [None]: us-east-2
Default output format [None]: json
ubuntu@ip-172-31-2-56:~$ aws sts get-caller-identity
{
    "UserId": "AIDA23MHLIEGO2E6NP74R6",
    "Account": "746669220253",
    "Arn": "arn:aws:iam::746669220253:user/DevOps_2403_G1"
}
ubuntu@ip-172-31-2-56:~$
```

4.3 FASE 2: Creación del clúster Kubernetes en AWS EKS

(Paso 4) Se crea un clúster Kubernetes en AWS EKS con 3 nodos (create-cluster.sh).

Se han cambiado parámetros importantes en create-cluster.sh como

- CLUSTER_NAME=mundosE-EKS-cluster-G1 (Cambio el nombre del clúster)
- --name mundosE-EKS-cluster-G1 \(Reemplazamos el nombre del Cluster)
- --region us-east-2 \(Reemplazamos la región)
- ##ssh-access \(No se requiere acceso por SSH a los nodos, porque estos los administra AWS)
- ##ssh-public-key jenkins \(grupo de nodos ng-default, facilita la administración y escalabilidad.)
- --nodegroup-name ng-default \(Trabajamos con 2 solo zona de disponibilidad)
- --zones us-east-2^a \(Habilitación de OIDC Despues de la Creación del Clúster)
- eksctl utils associate-iam-oidc-provider --region=\$AWS_REGION --cluster=\$CLUSTER_NAME --approve \(Actualización de Add-ons)

`eksctl` → Es la herramienta específica CLI, que se utiliza para crear y administrar clústeres **EKS** en AWS.

Este comando iniciará la creación del clúster en AWS con los parámetros actualizados.

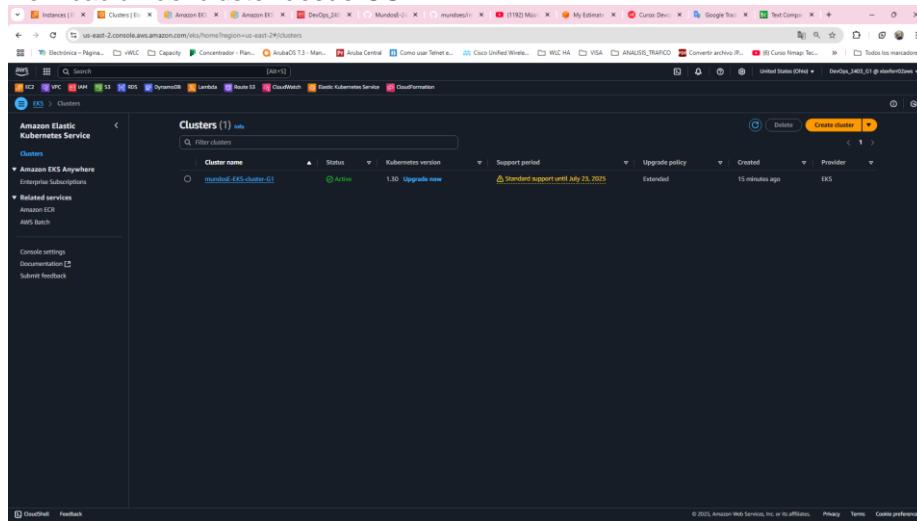
```
bash create-cluster.sh
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS 1 SERIAL MONITOR

● ubuntu@ip-172-31-2-56:~/MundosE-2403-PIN_FINAL-G1/01_eksctl$ ls -l
total 8
-rw-rw-r-- 1 ubuntu ubuntu 179 Feb 28 14:42 configmap.sh
-rw-rw-r-- 1 ubuntu ubuntu 786 Feb 28 22:03 create-cluster.sh
● ubuntu@ip-172-31-2-56:~/MundosE-2403-PIN_FINAL-G1/01_eksctl$ bash create-cluster.sh
Credenciales testeadas, proceder con la creacion de cluster.
```

Verificación del clúster desde CLI

```
eksctl get cluster --name mundosE-EKS-cluster-G1 --region us-east-2
```

Verificación del clúster desde GUI



PODS

Resource types

- Workloads
- PodTemplates
- Pods
- Replicasets
- Deployments
- Statefulsets
- Daemonsets
- Jobs
- CronJobs
- PriorityClasses
- HorizontalPodAutoscalers

Workloads: Pods [10]

Pod is the smallest and simplest Kubernetes object. A Pod represents a set of running containers on your cluster. [Learn more](#)

All namespaces View details

Name	Age
aws-node-8tg0	Created 3 minutes ago
aws-node-j4tf	Created 3 minutes ago
aws-node-l6fz	Created 3 minutes ago
coredns-854574ff-4g4d	Created 8 minutes ago
coredns-854574ff-8b97	Created 8 minutes ago
kube-proxy-hwgl	Created 5 minutes ago
kube-proxy-4hmg	Created 5 minutes ago
kube-proxy-4tgr	Created 5 minutes ago
metrics-server-4cd0d7bd-2n49	Created 8 minutes ago
metrics-server-4cd0d7bd-ccwf	Created 8 minutes ago

Compute

Overview Resources Compute Networking Add-ons Access Observability Update history Tags

Nodes (3) info

Filter nodes by property or value

Node name	Instance type	Compute	Managed by	Created	Status
ip-192-168-18-107.us-east-2.compute.internal	t2.small	Node group	ng-default	Created 6 minutes ago	Ready
ip-192-168-35-98.us-east-2.compute.internal	t2.small	Node group	ng-default	Created 6 minutes ago	Ready
ip-192-168-54-166.us-east-2.compute.internal	t2.small	Node group	ng-default	Created 6 minutes ago	Ready

Node groups (1) info

Node groups implement basic compute scaling through EC2 Auto Scaling groups.

Group name	Desired size	AMIS release version	Launch template	Status
ng-default	5	1.30.9-20210224	eksctl-mundosE-EKS-cluster-G1-nodegroup-ng (1)	Active

Fargate profiles (0) info

No Fargate profiles

This cluster does not have any Fargate profiles.

Networking

mundosE-EKS-cluster-G1

Your cluster's Kubernetes version (1.30) will reach the end of standard support on July 23, 2025. On that date, your cluster will enter the extended support period with additional fees. For more information, see the [upgrade page](#). [Upgrade now](#)

Cluster info

Status: Active	Kubernetes version: 1.30	Support period: Standard support until July 23, 2025	Provider: EKS
----------------	--------------------------	--	---------------

Networking

VPC: Info vpc-0d450e72b009e57	Subnets: Info subnet-0ff527518c03a70f subnet-0ff527518c03a70f subnet-0a6463732c99f subnet-0a6463732c99f subnet-07da5194d57647db	Cluster security group: Info sg-0f55a6e01cc07f2	API server endpoint access: Info Public Public access source allowlist: 0.0.0.0/0 (open to all traffic)
----------------------------------	--	--	---

Add-ons

Add-on	Category	Status	Version	EKS Pod Identity	IAM role for service account (IRSA)
CoreDNS	Networking	Degraded	v1.11.1-eksbuild.0	-	Not set
Amazon VPC CNI	Networking	Upgrading	v1.19.0-eksbuild.1	-	Not set
kube-proxy	Networking	Active	v1.30.6-eksbuild.5	-	Not set
Metrics Server	Monitoring	Upgrading	v0.4.0-eksbuild.0	-	Not set

(Paso 5) Se edita el ConfigMap de Kubernetes para gestionar permisos de acceso (configmap.sh).

Este paso permite que usuarios o roles específicos tengan permisos administrativos dentro del clúster EKS.

- Validación que el clúster está activo y que `kubectl` está configurado correctamente.
`kubectl get nodes`
- Editar el ConfigMap manualmente
`bash 01_eksctl/configmap.sh`
- Se sustituye `arn:aws:iam::746669220253:user/DevOps_2403_G1` con el ARN correcto de tu usuario IAM.
- Para obtener el ARN del usuario IAM, se ejecuta:
`aws sts get-caller-identity --query "Arn" --output text`
- Después de guardar ConfigMap, se aplica configuración con:
`kubectl get configmap -n kube-system aws-auth -o yaml`

Se edita con vi -> ESC :wq

```

$ kubectl edit -n kube-system configmap/aws-auth
 1 #!/bin/bash
 2
 3 kubectl edit -n kube-system configmap/aws-auth
 4
 5 # mapRoles: [
 6 #   - groups:
 7 #     - system:masters
 8 #       username: system:iam:arn:aws:iam::xxxxxxxxxxxx:root
 9 #       userRole: xxx
10
11 # Please edit the object below. Lines beginning with a '#' will be ignored,
12 # and an empty file will abort the edit. If an error occurs while saving this file will be
13 # responded with the relevant failures.
14
15 apiVersion: v1
16 data:
17   mapRoles:
18     - groups:
19       - system:bootstrappers
20       - system:nodes
21       - system:masters
22         roles:
23           - system:aws:iam:arn:aws:iam::746669220253:role/eksctl-mundosEKS-cluster-G1-node-NodeInstanceRole-ZE3WdscrC2H
24             username: system:node:{[EC2PrivateDNSName]}
25
26   mapUsers:
27     - users:
28       - username: arn:aws:iam::746669220253:user/DevOps_2403_G1
29         username: DevOps_2403_G1
30       - groups:
31         - system:masters
32
33 kind: ConfigMap
34 metadata:
35   creationTimestamp: "2025-03-01T01:38:44Z"
36   name: aws-auth
37   namespace: kube-system
38   resourceVersion: "1342"
39   uid: 8701a6c2-d078-411c-b8d4-83ca0496953f

```

```

$ kubectl get configmap -n kube-system aws-auth -o yaml
 1 #!/bin/bash
 2
 3 kubectl edit -n kube-system configmap/aws-auth
 4
 5 # mapRoles: [
 6 #   - groups:
 7 #     - system:masters
 8 #       username: system:iam:arn:aws:iam::xxxxxxxxxxxx:root
 9 #       userRole: xxx
10
11 # Please edit the object below. Lines beginning with a '#' will be ignored,
12 # and an empty file will abort the edit. If an error occurs while saving this file will be
13 # responded with the relevant failures.
14
15 apiVersion: v1
16 data:
17   mapRoles:
18     - groups:
19       - system:bootstrappers
20       - system:nodes
21       - system:masters
22         roles:
23           - system:aws:iam:arn:aws:iam::746669220253:role/eksctl-mundosEKS-cluster-G1-node-NodeInstanceRole-ZE3WdscrC2H
24             username: system:node:{[EC2PrivateDNSName]}
25
26   mapUsers:
27     - users:
28       - username: arn:aws:iam::746669220253:user/DevOps_2403_G1
29         username: DevOps_2403_G1
30       - groups:
31         - system:masters
32
33 kind: ConfigMap
34 metadata:
35   creationTimestamp: "2025-03-01T01:38:44Z"
36   name: aws-auth
37   namespace: kube-system
38   resourceVersion: "7254"
39   uid: 8701a6c2-d078-411c-b8d4-83ca0496953f

```

Cambios realizados y explicación

- Se mantiene mapRoles: (permite que los nodos EC2 se unan al clúster).
- Se agregó mapUsers: (permite que el usuario IAM DevOps_2403_G1 tenga acceso administrativo al clúster).

4.4 FASE 4: Implementación del monitoreo con Prometheus y Grafana

Prometheus almacena métricas históricas en un volumen persistente para evitar pérdida de datos si el pod se reinicia.

Entonces los volúmenes gp2 es una clase de almacenamiento en AWS EBS (Elastic Block Store) utilizada comúnmente para volúmenes persistentes en Kubernetes.

Antes que nada debemos verificar si en mi región tenemos disponible almacenamiento gp2

kubectl get storageclass

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS 1    SERIAL MONITOR

● ubuntu@ip-172-31-2-56:~/MundosE-2403-PIN_FINAL-G1$ kubectl get storageclass
  NAME      PROVISIONER      RECLAIMPOLICY      VOLUMEBINDINGMODE      ALLOWVOLUMEEXPANSION      AGE
  gp2       kubernetes.io/aws-ebs   Delete        WaitForFirstConsumer   false                    72m
○ ubuntu@ip-172-31-2-56:~/MundosE-2403-PIN_FINAL-G1$ 
```

(Paso 12) Se despliega Prometheus en Kubernetes para recolectar métricas (prometheus-deploy.sh).

bash prometheus-deploy.sh

The screenshot shows a terminal session on an Ubuntu host (SSH: 10.117.166.12) with several tabs open. The current tab displays the execution of a shell script to deploy Prometheus and Grafana using Helm.

```
$ create-cluster.sh M $ grafana-deploy.sh $ prometheus-deploy.sh
MundoE-2403-PIN-FINAL-G1> 04_monitoreo > prometheus-deploy.sh
$ #!/bin/bash
#
# Verificar si los repositorios de Helm están configurados
if ! helm repo list | grep -q 'prometheus-community'; then
    echo "Agregando repositorio de Prometheus..."
    helm repo add prometheus-community https://prometheus-community.github.io/helm-charts
fi

ubuntu@172-31-2-56:/~$ ls -l
total 12
-rw-r--r-- 1 ubuntu ubuntu 315 Feb 28 14:44 grafana-deploy.sh
-rw-r--r-- 1 ubuntu ubuntu 221 Feb 28 14:44 grafana.yam
-rw-r--r-- 1 ubuntu ubuntu 1054 Mar  1 03:11 prometheus-deploy.sh
ubuntu@172-31-2-56:/~$ bash prometheus-deploy.sh
Error: no repositories to show!
  * $Regrado repository of Prometheus...
  * Prometheus has been added to your repositories
  * $Regrado repository of Grafana...
  * "grafana" has been added to your repositories
  * Using --upgrade we grab latest from your chart repositories...
  * Successfully got an update from the "grafana" chart repository
  * Successfully got an update from the "prometheus-community" chart repository
  * Update Complete, 🎉 Happy Helm-ing!
  * Helm charts have been created
  * NAME: prometheus
  * LAST DEPLOYED: Sat Mar  1 03:12:07 2025
  * NAMESPACE: prometheus
  * STATUS: deployed
  * REVISION: 1
  * TEST SUITE: None
  * NOTES:
The Prometheus server can be accessed via port 9090 on the following DNS name from within your cluster:
prometheus-server.prometheus.svc.cluster.local

Get the Prometheus server URL by running these commands in the same shell:
export POD_NAME=$(kubectl get pods --namespace prometheus -l app.kubernetes.io/name=prometheus,app.kubernetes.io/instance=prometheus" -o jsonpath=".items[0].metadata.name")
kubectl --namespace prometheus port-forward $POD_NAME 9090

The Prometheus alertmanager can be accessed via port 9093 on the following DNS name from within your cluster:
prometheus-alertmanager.prometheus.svc.cluster.local

Get the Alertmanager URL by running these commands in the same shell:
export POD_NAME=$(kubectl get pods --namespace prometheus -l app.kubernetes.io/name=alertmanager,app.kubernetes.io/instance=prometheus" -o jsonpath=".items[0].metadata.name")
kubectl --namespace prometheus port-forward $POD_NAME 9093
#####
##### WARNING: Pod Security Policy has been disabled by default since #####
##### It deprecated after k8s 1.25, use #####
#####   (Index .Values "prometheus-node-exporter" "rbac" #####
#####     "pipedLineWith" (Index .Values #####
#####       "prometheus-node-exporter" "rbac" "pipedLineWith") #####
#####     "in case you still need it. #####
#####
```

comando para verificar los volúmenes persistentes

```
kubectl get pvc -n prometheus
```

comando para ver el estado de los pods:

```
kubectl get pods -n prometheus
```

comando para verificar el tipo de servicio de Prometheus:

El servicio prometheus-server tiene TYPE: ClusterIP, lo cual significa que solo puede ser accedido dentro del clúster.

kubectl get svc -n prometheus

```
File Edit Selection View Go Run Terminal Help ↵ → ○ ubuntu [SSH: 18.117.166.12]
EXPLORER ... $ create-cluster.sh M $ grafana-deploy.sh M $ prometheus-deploy.sh M $ delete-prometheus.sh X
MundoE2403-PIN_FINAL-G1/05_Delete > $ delete-prometheus
1 #!/bin/bash
2
3 # Confirmación antes de proceder
4 echo "⚠️ ATENCIÓN: Este script eliminará completamente Prometheus y todos sus recursos."
5 read -p "¿Estás seguro de que deseas continuar? (yes/no): " CONFIRM
6 if [[ "$CONFIRM" != "yes" ]]; then
7   echo "Cancelando eliminación."
8   exit 1
9 fi
10
11 # Desinstalar Prometheus
12 echo "🔴 Eliminando Prometheus..."
13 helm uninstall prometheus --namespace prometheus
14
15 # Eliminar el namespace de Prometheus
16 echo "🔴 Eliminando namespace 'prometheus'..."
17
18 M PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS SERIAL MONITOR
19
20 ⓘ ubuntu@172-31-2-56:~/MundoE2403-PIN_FINAL-G1/04_monitors$ kubectl get svc -n prometheus
21 NAME          STATUS    VOLUME   CAPACITY   ACCESS MODES  STORAGECLASS   VOLUMEATTIBUTES CLASS   AGE
22
23 prometheus-server   Ready     pvc-c3e54f78-495b-4cc2-bfba-1ce2cf2de0ce  8Gi   RWD   gp2        cuse>           11m
24 prometheus-alertmanager   Ready     pvc-630a4043-720d-40f3-891c-85df18841c1b  1Gi   RWD   gp2        cuse>           11m
25
26 ⓘ ubuntu@172-31-2-56:~/MundoE2403-PIN_FINAL-G1/04_monitors$ kubectl get pods -n prometheus
27 NAME                               READY   STATUS    RESTARTS   AGE
28
29 prometheus-alertmanager-0   1/1     Running   0          11m
30 prometheus-kube-state-metrics-4554695c64-avvg2   1/1     Running   0          11m
31 prometheus-prometheus-node-exporter-gmr2v   1/1     Running   0          11m
32 prometheus-prometheus-node-exporter-lqd4   1/1     Running   0          11m
33 prometheus-prometheus-node-exporter-tdf2f   1/1     Running   0          11m
34 prometheus-prometheus-pushgateway-65cbff6f-8fp2   2/2     Running   0          11m
35
36 ⓘ ubuntu@172-31-2-56:~/MundoE2403-PIN_FINAL-G1/04_monitors$ kubectl get pods -n prometheus
37 NAME                               READY   STATUS    RESTARTS   AGE
38
39 prometheus-server-6c5b5959cd-dwqld   1/1     Running   0          11m
40
41 ⓘ ubuntu@172-31-2-56:~/MundoE2403-PIN_FINAL-G1/04_monitors$ kubectl port-forward -n prometheus deploy/prometheus-server 8080:9090 --address 0.0.0.0
Forwarding from 0.0.0.0:8080 → 3056
42
43 ⓘ curl http://172-31-2-56:3056/metrics
44
45 NAME          TYPE    CLUSTER-IP      EXTERNAL-IP   PORT(S)   AGE
46
47 prometheus-alertmanager   ClusterIP  10.100.112.21   <none>     9093/TCP   20m
48 prometheus-alertmanager-headless   ClusterIP  None         <none>     9093/TCP   20m
49 prometheus-kube-state-metrics   ClusterIP  10.100.210.125  <none>     8080/TCP   20m
50 prometheus-prometheus-node-exporter   ClusterIP  10.100.145.115  <none>     9100/TCP   20m
51 prometheus-prometheus-pushgateway   ClusterIP  10.100.213.9   <none>     9091/TCP   20m
52
53 ⓘ curl http://172-31-2-56:3056/metrics
54
```

(Paso 13) Se despliega Grafana para visualizar métricas (grafana-deploy.sh).

```
Muestra todos los recursos en el namespace grafana  
kubectl get all -n grafana
```

```
Lista los PersistentVolumeClaims (PVC) en el namespace grafana  
kubectl get pvc -n grafana
```

```
File Edit Selection View Go Run Terminal Help < - > ubuntu [SSH: 18.117.166.12]
EXPLORER
URUNBU [SSH: 18.117.166.12] ...
$ create-cluster.sh M $ grafana-deploy.sh M x $ delete-grafana.sh U I grafon.yaml I prometheus-deploy.sh M $ delete-prometheus.sh U
MundoE2403-PIN-FINAL-G1>04_monitoreo> $ grafana-deploy.sh
#!/bin/bash
...
# Verificar si el repositorio de Grafana está agregado
if ! helm repo list | grep -q grafana; then
    echo "Agregando repositorio de Grafana..."
    helm repo add grafana https://grafana.github.io/helm-charts
fi

# Actualizar repositorio
helm repo update

# Crear el namespace para Grafana si no existe
kubectl get namespace grafana &>/dev/null || kubectl create namespace grafana

# Desplegar Grafana en EKS
helm install grafana grafana/grafana \
--namespace grafana \
--set persistentVolumeClassName="gp2" \
--set persistence.enabled=true \
--set adminPassword="contraseña" \
--set service.type=ClusterIP

# Verificar la instalación
kubectl get all -n grafana
25

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS SERIAL MONITOR
ubuntu@ip-172-31-2-56:~/MundoE2403-PIN-FINAL-G1>04_monitoreo> $ kubectl get all -n grafana
NAME READY STATUS RESTARTS AGE
pod/grafana-5459c4587c-jn5dx 1/1 Running 0 2m55s

NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE
service/grafana ClusterIP 10.108.78.46 cronle 2m55s

NAME READY UP-TO-DATE AVAILABLE AGE
deployment.apps/grafana 1/1 1 1 2m55s

NAME DESIRED CURRENT READY AGE
replicaset.apps/grafana-5459c4587c 1 1 1 2m55s
ubuntu@ip-172-31-2-56:~/MundoE2403-PIN-FINAL-G1>04_monitoreo> $ kubectl get pvc -n grafana
NAME STATUS VOLUME CAPACITY ACCESS MODES STORAGECLASS VOLUMEATTACHMENTCLASS AGE
grafana-pvc Bound pvc-98c9133a-8aee-4f86-8aef-2e501d58e6 10Gi RWO gp2 2m55s
ubuntu@ip-172-31-2-56:~/MundoE2403-PIN-FINAL-G1>04_monitoreo>
```

Verificación de los pods en Grafana:
`kubectl get pods -n grafana`

Verificación del servicio (svc) de Grafana:
`kubectl get svc -n qrafana`

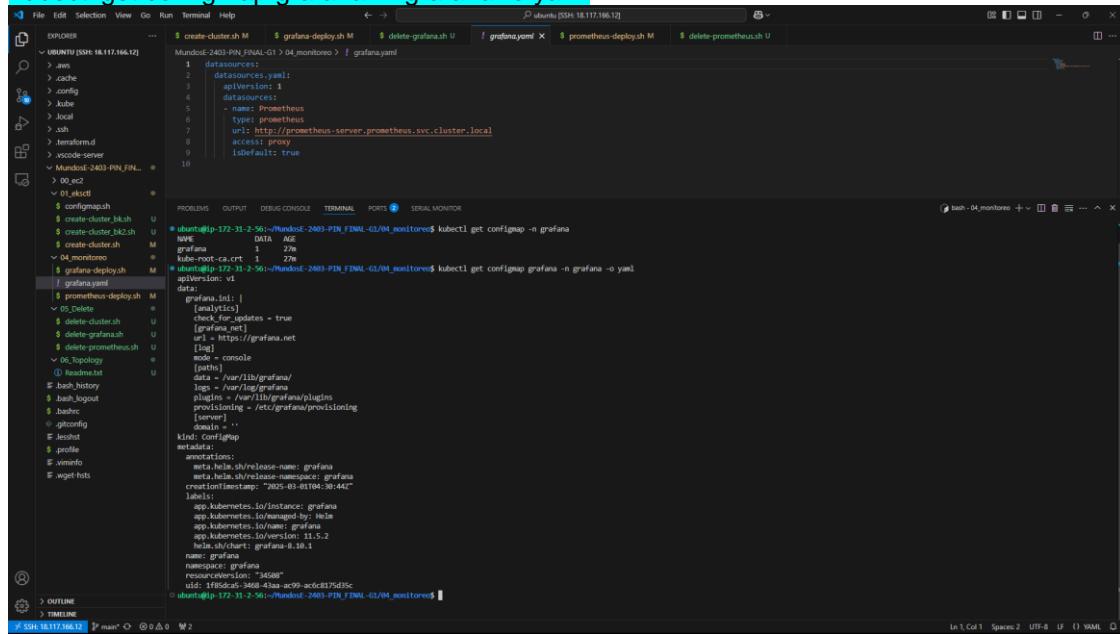
Grafana tiene su configuración almacenada en un ConfigMap. Grafana configurado correctamente dentro del clúster.

```
kubectl get configmap -n grafana
```

(Paso 14) Se configura Grafana para obtener datos desde Prometheus (grafana.yaml).

kubectl get configmap -n grafana

kubectl get configmap grafana -n grafana -o yaml



```
ubuntu@ip-172-31-2-56:~/Mundod-2403-PIN_FINAL-G1/04_monitores> kubectl get configmap -n grafana
ubuntu@ip-172-31-2-56:~/Mundod-2403-PIN_FINAL-G1/04_monitores> kubectl get configmap grafana -n grafana -o yaml
```

```
apiVersion: v1
kind: ConfigMap
metadata:
  creationTimestamp: "2025-03-01T08:36:44Z"
  name: grafana
  namespace: grafana
  resourceVersion: "43498"
  uid: f16d6c5-3468-43a4-ac99-ac6c875d35c
data:
  grafana.yaml: |-
    datasources:
    - name: Prometheus
      type: prometheus
      url: http://prometheus-server.prometheus.svc.cluster.local
      access: proxy
      isDefault: true
```

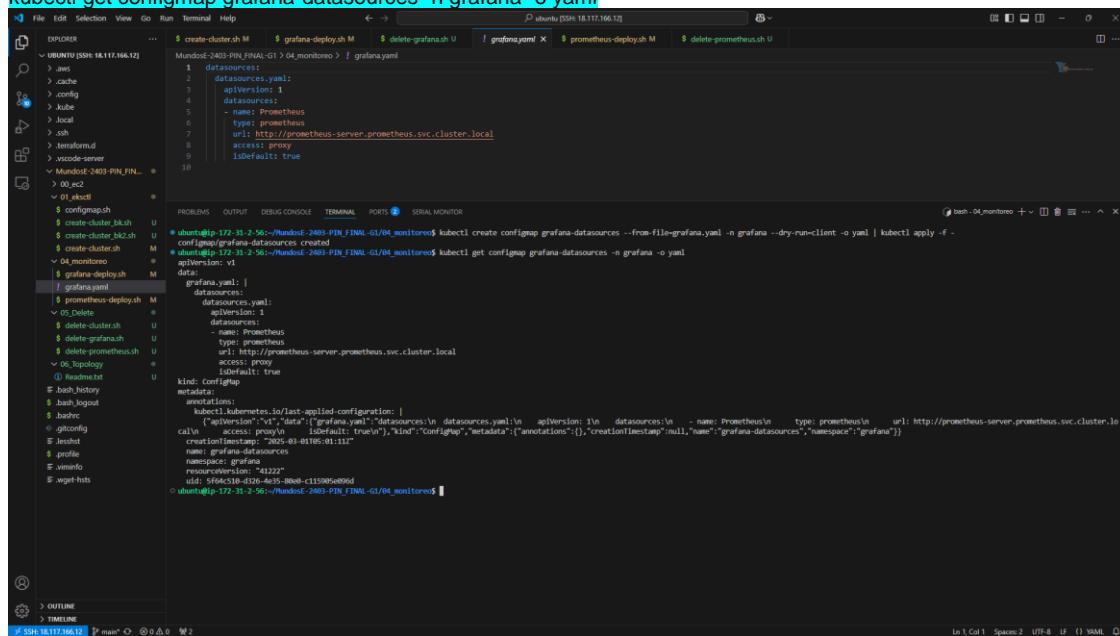
Aplicar la configuración

Ejecuta el siguiente comando para cargar la configuración en Grafana:

kubectl create configmap grafana-datasources --from-file=grafana.yaml -n grafana --dry-run=client -o yaml | kubectl apply -f -

Verifica que la configuración se aplicó correctamente:

kubectl get configmap grafana-datasources -n grafana -o yaml



```
ubuntu@ip-172-31-2-56:~/Mundod-2403-PIN_FINAL-G1/04_monitores> kubectl create configmap grafana-datasources --from-file=grafana.yaml -n grafana --dry-run=client -o yaml | kubectl apply -f -
configmap/grafana-datasources created
ubuntu@ip-172-31-2-56:~/Mundod-2403-PIN_FINAL-G1/04_monitores> kubectl get configmap grafana-datasources -n grafana -o yaml
```

```
apiVersion: v1
kind: ConfigMap
metadata:
  creationTimestamp: "2025-03-01T08:36:44Z"
  name: grafana-datasources
  namespace: grafana
  resourceVersion: "43498"
  uid: f16d6c5-3468-43a4-ac99-ac6c875d35c
data:
  grafana.yaml: |-
    datasources:
    - name: Prometheus
      type: prometheus
      url: http://prometheus-server.prometheus.svc.cluster.local
      access: proxy
      isDefault: true
```

4.5 FASE 5: Instalación de Classic Load Balancer

(Paso 15) Se despliega el servicio de Load Balancer para acceder a los recursos de manera externa.

(Paso 16) validar acceso a los recursos.

```
kubectl get svc -n grafana
```

```
File Edit Selection View Go Run Terminal Help ↵ → ○ ubuntu [SSH: 18.117.166.12]
```

```
EXPLORER
```

```
URUNBIT [SSH: 18.117.166.12]
```

```
> aws  
> .cache  
> config  
> kube  
> jocal  
> .ssh  
> terraform.d  
> vscode-server  
> Mundoos2403-PIN_FINAL-G1... ●  
> 00_ec2  
  01_eksctl  
  $ configmap.sh  
  $ create-cluster_b1sh U  
  $ create-cluster_b12sh U  
  $ create-cluster.sh U  
  ✓ 04_monitrore U  
  $ deploy_load_balancer.sh U  
  $ grafana-deploy.sh M  
  $ grafana.yaml  
  $ prometheus-deploy.sh M  
  ✓ 05_Delete U  
  $ delete-cluster.sh U  
  $ delete-grafana.sh U  
  $ delete-prometheus.sh U  
  ✓ 06_headme.txt  
  $ bash_history  
  $ bash_logout  
  $ bashrc  
  $ gitconfig  
  $ lesshtd  
  $ profile  
  $ viminfo  
  $ wget hsts
```

```
# create-cluster.sh M $ grafana-deploy.sh M X $ delete-grafana.sh U $ grafana.yaml $ deploy_load_balancer.sh U $ prometheus-deploy.sh M $ delete-prometheus.sh U
```

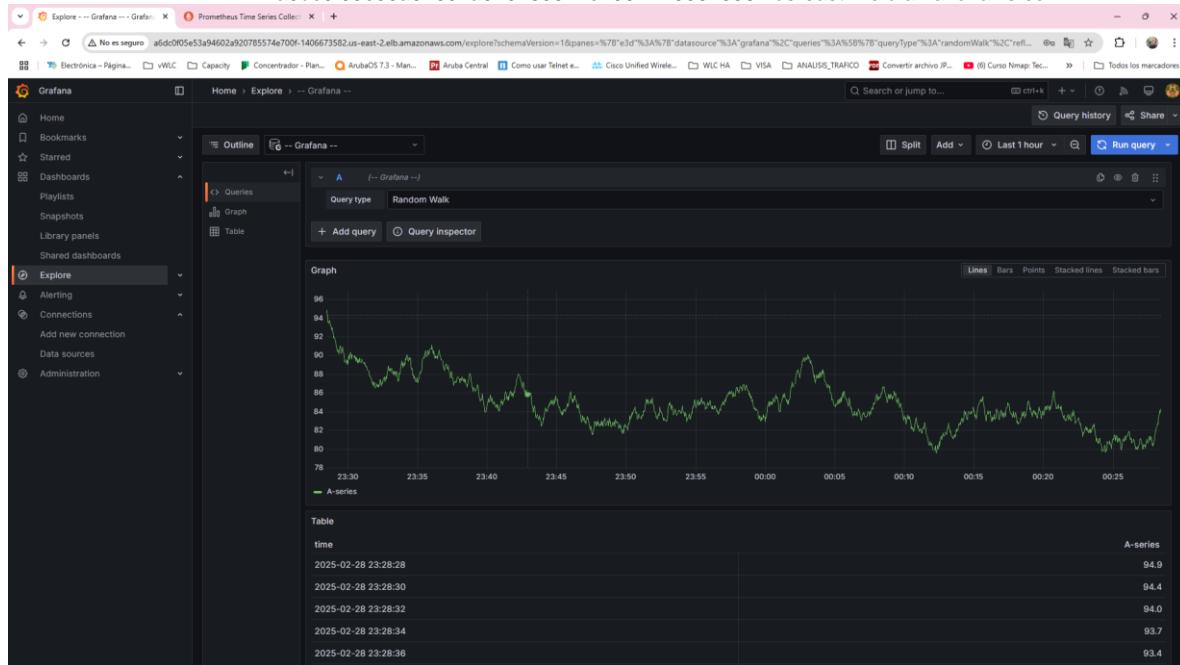
```
ubuntu@ip-172-31-2-56:~/Mundoos2403-PIN_FINAL-G1/04_monitrore$ kubectl get svc -n grafana
```

```
NAME      TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)          AGE
grafana   LoadBalancer 10.100.78.46   <none>        80:30625/TCP   4d
grafana   ClusterIP   10.100.112.121  <none>        80:30625/TCP   4d
prometheus-alertmanager-headless   ClusterIP   10.100.112.121  <none>        9093/TCP,9464/TCP  4d
prometheus-alertmanager-headless   ClusterIP   None         <none>        9093/TCP,9464/TCP  4d
prometheus-kube-state-metrics    ClusterIP   10.100.219.175  <none>        8000/TCP,9090/TCP  4d
prometheus-metric-exporter       ClusterIP   10.100.145.115  <none>        9090/TCP,9100/TCP  4d
prometheus-prometheus-pushtagateway ClusterIP   10.100.213.155  <none>        9091/TCP           4d
prometheus-server                LoadBalancer 10.100.172.214  aedea0de7031f4347a4cc4ebab43254-1427972924.us-east-2.elb.amazonaws.com  80:38128/TCP  4d
```

```
ubuntu@ip-172-31-2-56:~/Mundoos2403-PIN_FINAL-G1/04_monitrore$
```

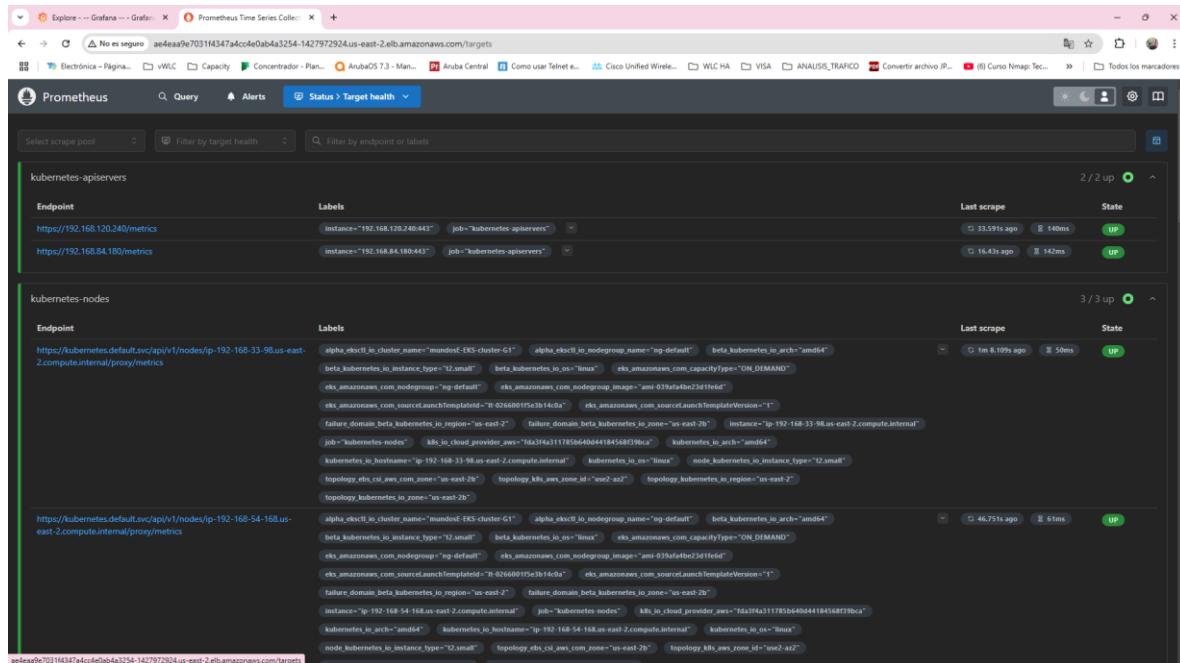
Grafana

- TYPE: LoadBalancer → Grafana ahora está expuesto públicamente.
- EXTERNAL-IP: a6dc0f05e53a94602a920785574e700f-1406673582.us-east-2.elb.amazonaws.com



Prometheus

- TYPE: LoadBalancer → Prometheus también está expuesto públicamente.
- EXTERNAL-IP: ae4eaa9e7031f4347a4cc4e0ab4a3254-1427972924.us-east-2.elb.amazonaws.com



(Paso 17) Se despliega un Nginx en Kubernetes, para que el prometheus, pueda recolectar las métricas y con grafana podamos observarlas a través de un dashboard, también se accede externamente.

```

$ deploy.nginx.sh
MundoE-2403-PIN_FINAL-G1 > 04_monitoreo > $ deploy.nginx.sh
...
15 # Despliegando Nginx en Kubernetes
16 echo "# Despliegando Nginx..."
17 kubectl apply -n nginx -f - <<EOF
18 apiVersion: apps/v1
19 kind: Deployment
20 metadata:
21   name: nginx-deployment
22 spec:
23   replicas: 1
24   selector:
25     matchLabels:
26       app: nginx
27   template:
28     metadata:
29       labels:
30         app: nginx
31     spec:
32       containers:
33         name: nginx
34           image: nginx:latest
35       ports:
36         - containerPort: 80
37   EOF
38
39 # Exponer Nginx externamente con un LoadBalancer
40
41 delete.nginx_monitor... U
42 delete.nginx... U
43 delete-cluster... U
44 delete-grafana... U
45 delete-prometheus... U
46 README.txt U
47 .gitignore U
48 .keep_logout U
49 .lesshist U
50 .lesshosts U
51 .profile U
52 .venvinfo U
53 wget-hists U
54
55 > OUTLINE
56 > TIMELINE
57
58 18.117.166.12 2 main* O ② 0 0 0 ④ 2

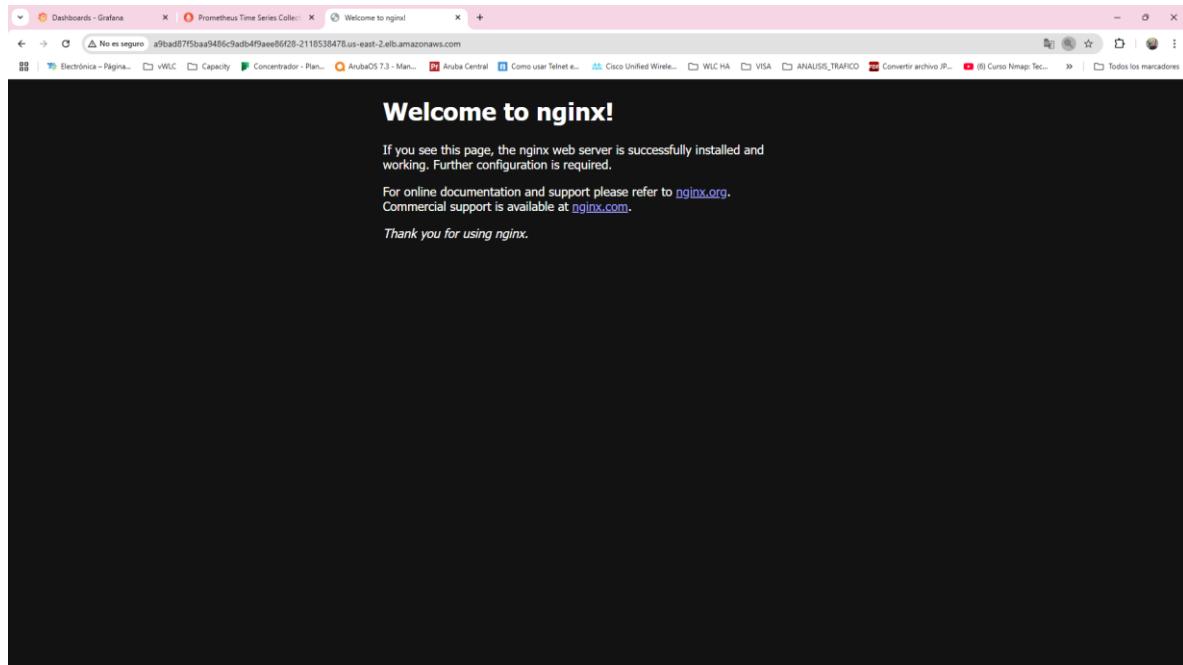
```

kubectl get svc -n nginx

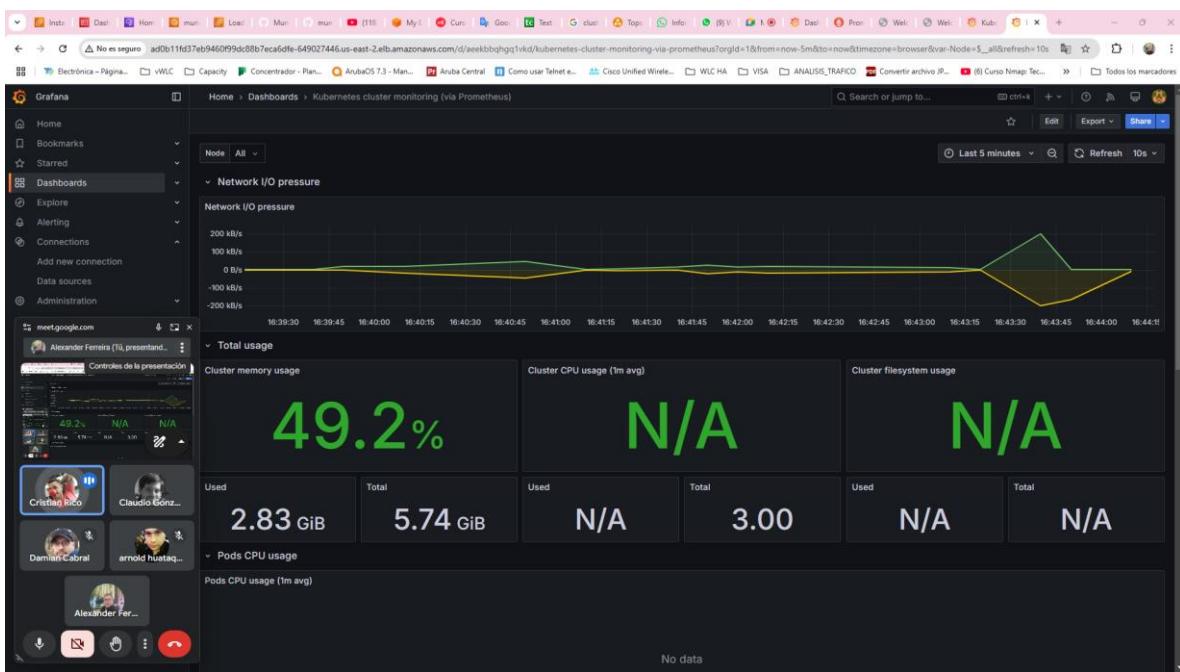
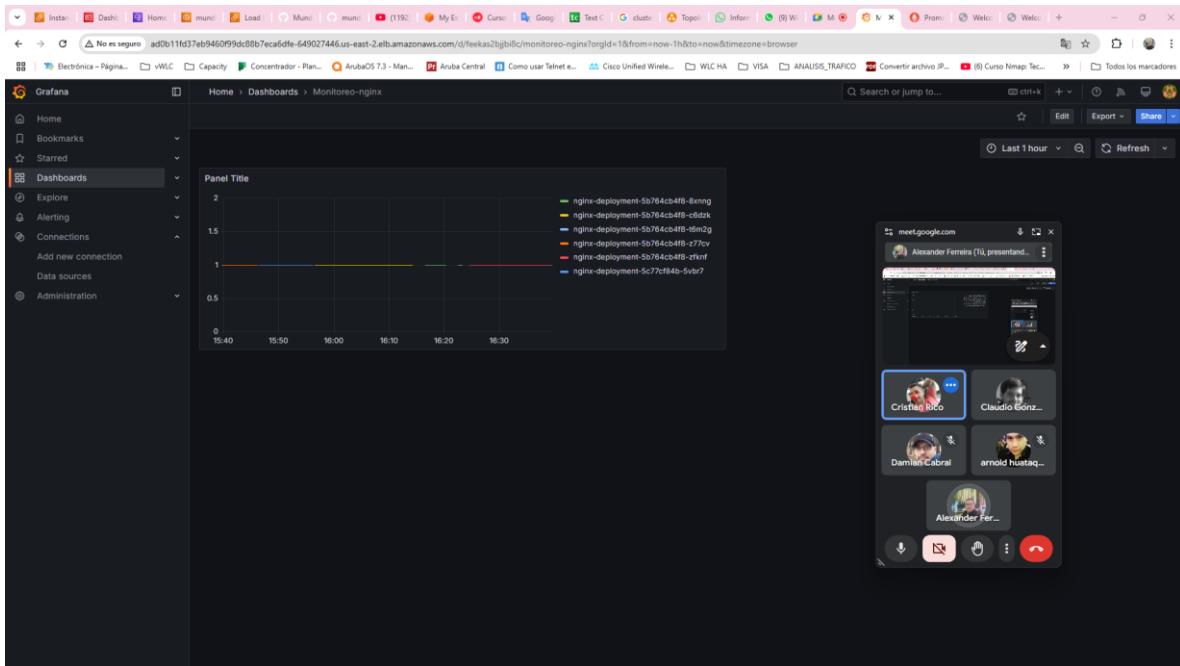
```

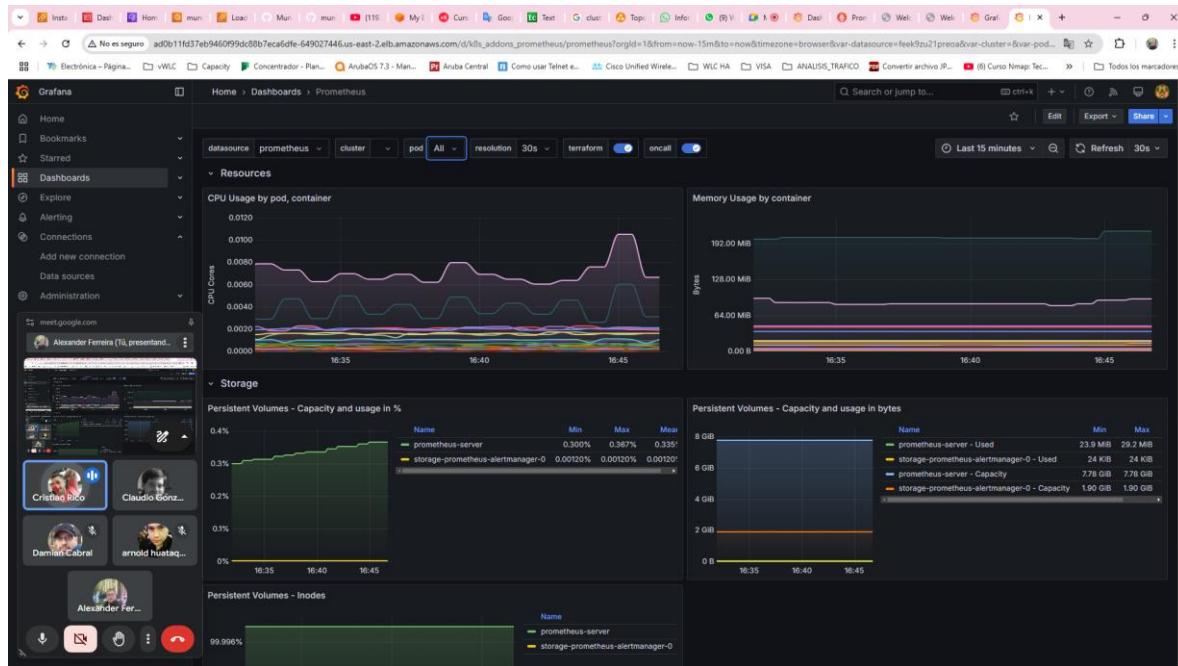
● ubuntu@ip-172-31-2-56:~/MundoE-2403-PIN_FINAL-G1/04_monitoreo$ kubectl get svc -n nginx
NAME          TYPE      CLUSTER-IP    EXTERNAL-IP        PORT(S)          AGE
nginx-service LoadBalancer 10.100.225.158 a9bad87f5baa9486c9adb4f9aee86f28-2118538478.us-east-2.elb.amazonaws.com 80:31548/TCP 88s
● ubuntu@ip-172-31-2-56:~/MundoE-2403-PIN_FINAL-G1/04_monitoreo$ 

```



Configuración grafana

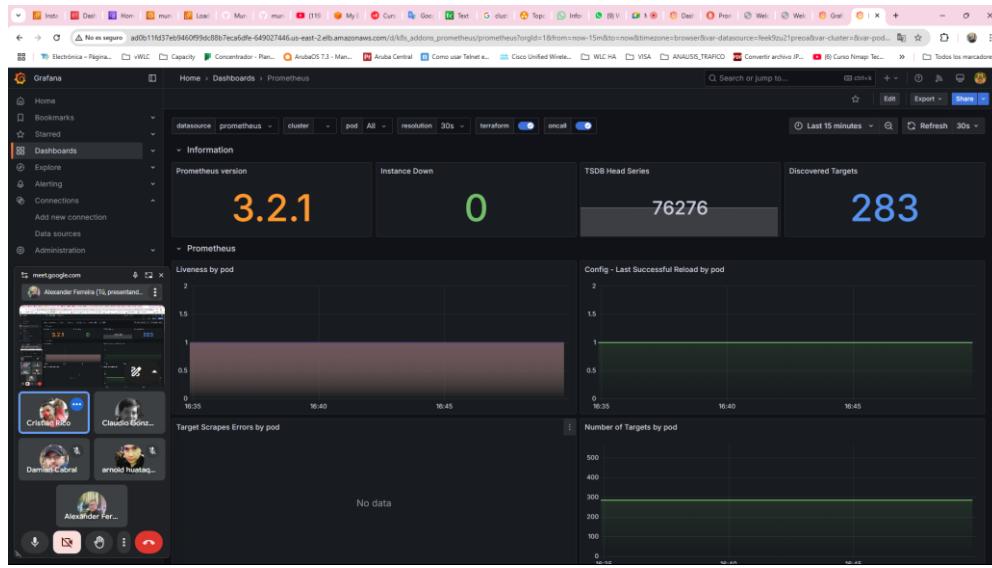




Bajamos POD de grafana

A screenshot of a developer's workspace. On the left, there's a file explorer showing a directory structure for 'MundoE-2403-PIN_FINAL-G1' containing files like 'aws', 'cache', 'config', 'kube', 'local', 'ssh', 'terraform', and 'grafana-deploy.sh'. A terminal tab titled 'grafanayaml' is open, displaying YAML configuration for a Prometheus datasource. Another terminal tab shows a list of pods using 'kubectl get pod'. In the center, there's a video conference interface from 'meet.google.com' with participants including Alexander Ferrira, Cristian Rico, Claudio Gonz... (partially visible), Damian Cobral, and Arnold Hosteq... (partially visible). The bottom of the screen features a toolbar with icons for problems, output, debug console, terminal, ports, and serial monitor. The status bar at the bottom shows the path 'SSH 3.145.77.24' and the command 'ls main'.

La configuración de guarda en el volumen persistente



4.6 FASE 6: Eliminación de recursos (Opcional)

(Paso 17) Se eliminan los recursos creados si es necesario

Vamos a borrar los recursos en este orden:

Eliminar Load Balancers

Bash delete_load_balancer.sh

Eliminar volúmenes persistentes

Bash delete_volumen_pc.sh

Eliminar el clúster de EKS

bash delete-cluster.sh

Verificar si AWS dejó recursos huérfanos

```
aws elb describe-load-balancers --query "LoadBalancerDescriptions[*].LoadBalancerName"
aws ec2 describe-volumes --query "Volumes[*].VolumeId"
```

Eliminación del Laod balancer

The screenshot shows the AWS EC2 Load Balancers console. The left sidebar navigation includes: Dashboard, EC2 Global View, Events, Instances (Instances, Instance Types, Launch Templates, Spot Requests, Savings Plans, Reserved Instances, Dedicated Hosts, Capacity Reservations), Images, Elastic Block Store (Volumes, Snapshots, Lifecycle Manager), Network & Security (Security Groups, Elastic IPs, Placement Groups, Key Pairs, Network Interfaces), Load Balancing (Load Balancers). The main content area is titled 'Load balancers' and displays a message: 'Elastic Load Balancing scales your load balancer capacity automatically in response to changes in incoming traffic.' Below this is a search bar labeled 'Filter load balancers' and a table header with columns: Name, DNS name, State, VPC ID, Availability Zones, Type, Date created. A message at the bottom states: 'You don't have any load balancers in us-east-2'. A blue button labeled 'Create load balancer' is visible.

Eliminación del Cluster

The screenshot shows the AWS EKS Clusters console. The left sidebar navigation includes: Amazon Elastic Kubernetes Service (Clusters), Amazon EKS Anywhere (Enterprise Subscriptions), Related services (Amazon ECR, AWS Batch), and links for Console settings, Documentation, and Submit feedback. The main content area is titled 'Clusters (1) Info' and shows a table with one item: 'mundos-E-KS-cluster-G1'. The table columns are: Cluster name, Status, Kubernetes version, Support period, Upgrade policy, Created, and Provider. The status is 'Deleting', Kubernetes version is '1.31', support period is 'Standard support until November 26, 2025', upgrade policy is 'Extended', created was '6 hours ago', and provider is 'EKS'. A blue button labeled 'Delete' is visible at the top right of the table.

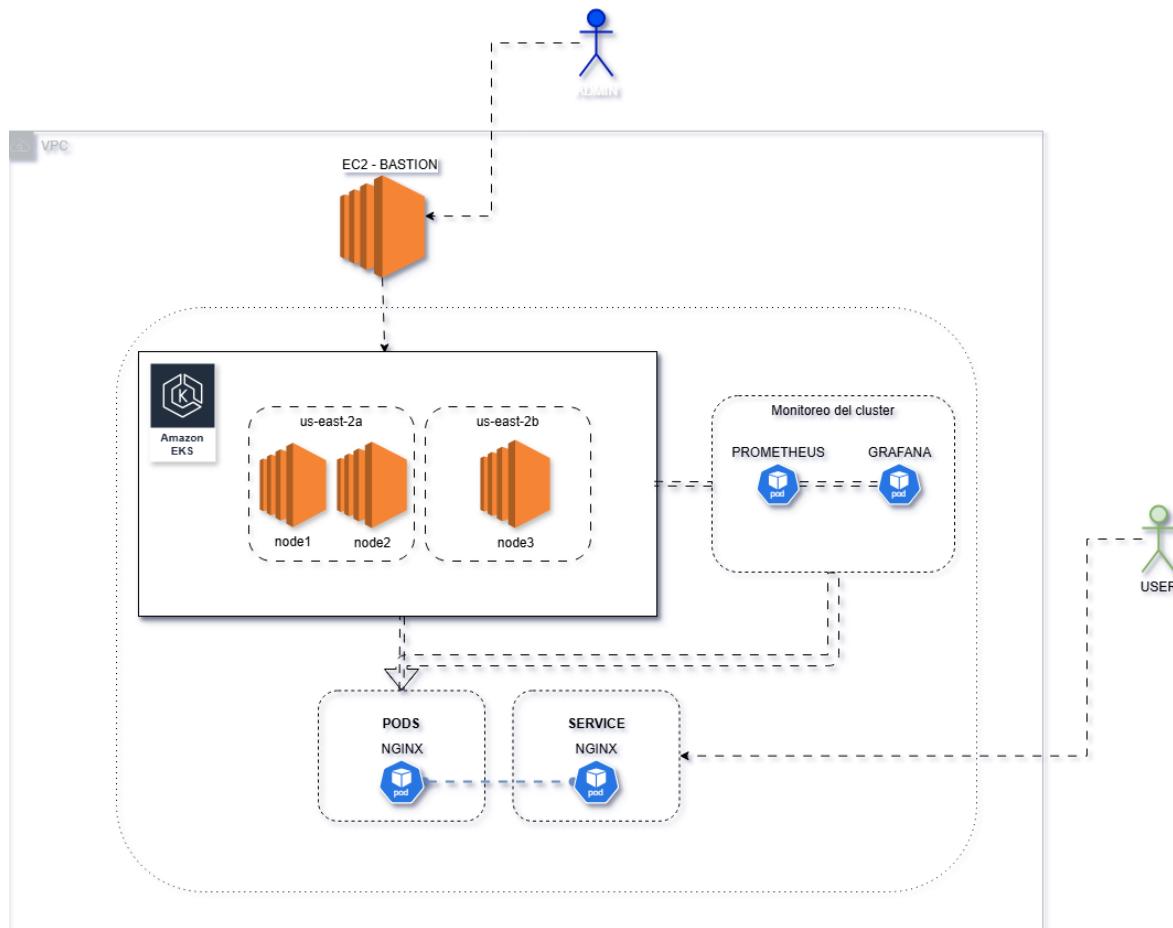
5. Conclusiones

Se logró automatizar completamente la infraestructura en AWS con eksctl, kubectl y helm, optimizando despliegues y reduciendo errores manuales. La configuración correcta de permisos, volúmenes y puertos fue clave para garantizar el funcionamiento estable de los servicios.

El monitoreo en tiempo real con Prometheus y Grafana permitió la observabilidad del clúster, implementando Exporters y ServiceMonitors para recolectar métricas críticas. Se utilizaron Load Balancers para exponer servicios como Nginx, asegurando accesibilidad y escalabilidad.

Se resolvieron desafíos en almacenamiento, compatibilidad de versiones y permisos mediante validaciones de storageClass y actualización de addons.

Se implementó una estrategia ordenada para la eliminación de recursos, evitando bloqueos en AWS. La modularidad del proyecto permite escalar y agregar nuevos servicios sin afectar la estabilidad. En conclusión, esta infraestructura automatizada y monitoreada con Kubernetes y AWS demuestra buenas prácticas de DevOps, garantizando eficiencia, seguridad y escalabilidad.



https://github.com/XLEXFERR02/MundosE-2403-PIN_FINAL-G1.git