

Docker y Kubernetes Avanzado



Argentina • Chile • Colombia • México • Perú

www.netec.com | servicio@netec.com



Propiedad intelectual

Material didáctico preparado por la empresa Global K, S.A. de C.V. Registrado en Derechos de Autor.

Todos los contenidos de este Sitio (incluyendo, pero no limitado a: texto, logotipos, contenido, fotografías, audio, botones, nombres comerciales y videos) están sujetos a derechos de propiedad por las leyes de Derechos de Autor de la empresa Global K, S.A. de C.V.

Queda prohibido copiar, reproducir, distribuir, publicar, transmitir, difundir, o en cualquier modo explotar cualquier parte de este documento sin la autorización previa por escrito de Global K, S.A. de C.V. o de los titulares correspondientes.

Descripción del curso

Este curso avanzado cubre la administración y despliegue de aplicaciones en contenedores con Kubernetes y Helm en la nube. Incluye el uso de AKS y EKS para la creación de clústeres, configuración de seguridad con OAuth 2.1 y JWT, balanceo de carga y almacenamiento persistente. Además, explora componentes internos de Kubernetes, como Kube-Scheduler y Kube-Controller, y finaliza con un proyecto DevSecOps integrando herramientas de monitoreo y CI/CD (Jenkins, Prometheus, Grafana) para un entorno seguro y optimizado.





Objetivos del curso

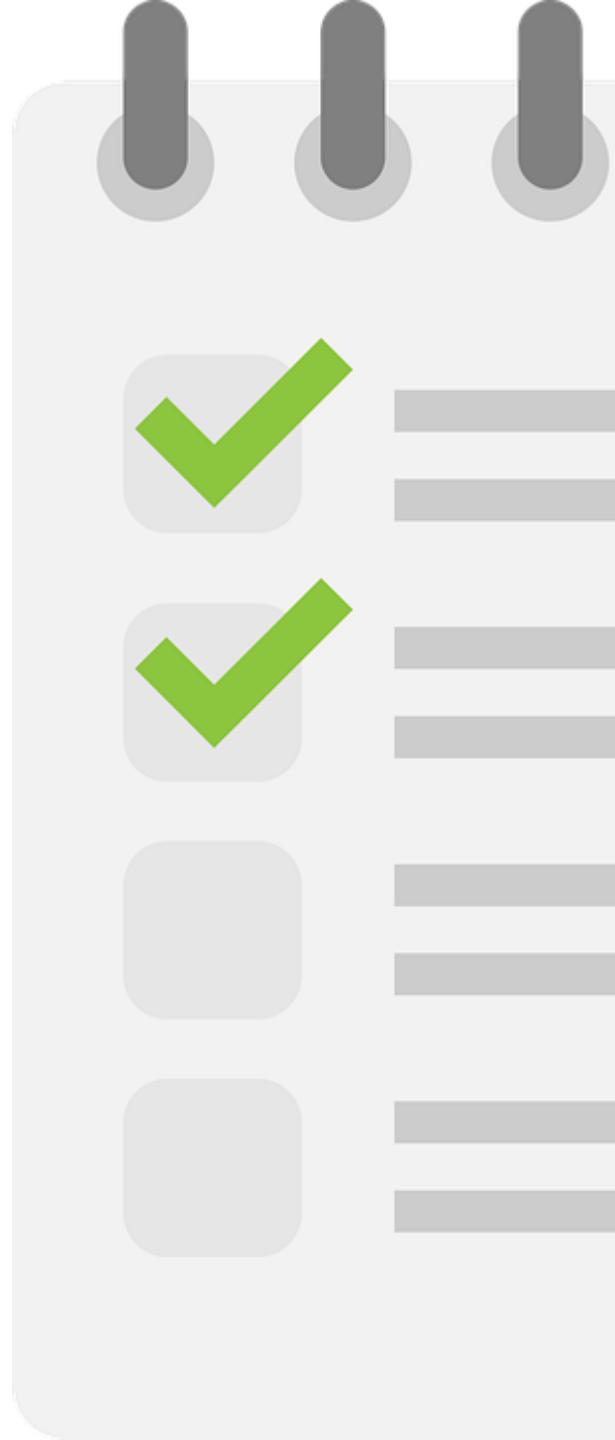
Al finalizar el curso, serás capaz de:

- Fortalecer las competencias técnicas sobre el software Docker y Kubernetes.
- Crear y usar imágenes y contenedores con Docker.
- Aprender sobre las redes de contenedores con Docker Networks y DNS Service Discovery, qué es Docker y Kubernetes y por qué utilizarlos en los microservicios.
- Centralizar las configuraciones de los microservicios con Kubernetes Config Map.
- Establecer comunicación entre microservicios.
- Habilitar balanceo de carga (LoadBalancer).
- Implementar Autenticación y Autorización con Spring Security OAuth2, JWT y Spring Authorization Server.

—

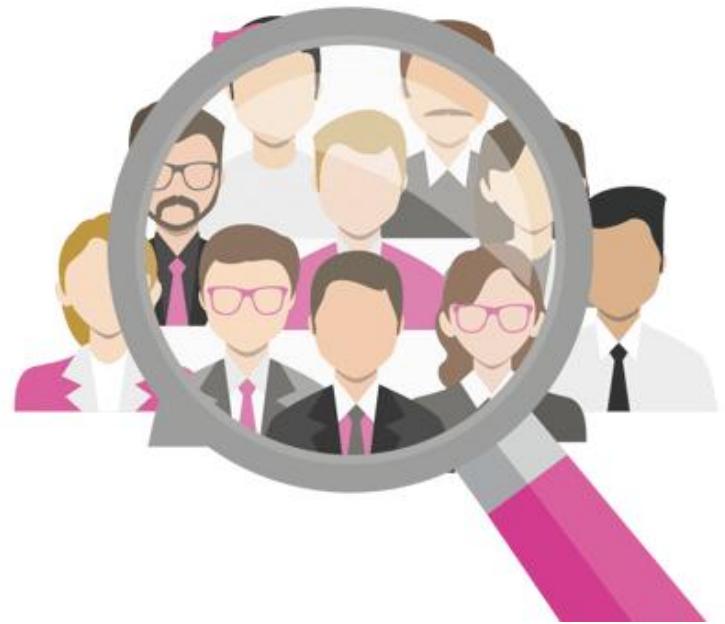
Prerrequisitos

- Conocimientos básicos de contenedores (Docker).
- Experiencia previa con servicios en la nube.
- Familiaridad con Kubernetes y sus componentes básicos (Pods, deployments, servicios).



Audiencia

- Ingenieros DevOps interesados en profundizar en despliegues con Kubernetes y seguridad.
- Desarrolladores backend y full stack que trabajan con microservicios en contenedores.
- Arquitectos de soluciones que deseen integrar DevSecOps en entornos Kubernetes.



Temario

- **Capítulo 1:** Helm Introducción
- **Capítulo 2:** Despliegue en EKS (Elastic Kubertenes Service)
- **Capítulo 3:** Kubernetes: Security JWT con OAuth 2.1
- **Capítulo 4:** Kubernetes API
- **Capítulo 5:** Explorando Pods

Presentación del grupo

¿Cuál es tu nombre?

¿Cuál es tu experiencia en desarrollo?

¿Qué tecnología/idea/software te ha impresionado?

¿Cuáles son tus expectativas respecto al curso?





Objetivos:

- Instalar Helm y aprender sus conceptos básicos.
- Desplegar un clúster AKS y gestionar contenedores.
- Configurar ECS con servicios y solucionar problemas de conexión.

Capítulo 1

Helm Introducción

1.1. Install Helm (Labs-install Helm)

Helm es un gestor de paquetes para Kubernetes que facilita la gestión, instalación y actualización de aplicaciones mediante el uso de charts. Estos charts son plantillas que organizan y describen todos los recursos necesarios para desplegar una aplicación en un clúster, optimizando el mantenimiento y escalabilidad de las mismas.

Instalación kubectl

- Nota: como prerequisito se sugiere tener instalado kubectl en la maquina antes de instalar Helm.
- Para instalar kubectl ingresar al siguiente link
<https://minikube.sigs.k8s.io/docs/start/?arch=%2Fmacos%2Farm64%2Fstable%2Fbinary+download>
- Seleccionar la información del SO.

1 Installation

Click on the buttons that describe your target platform. For other architectures, see [the release page](#) for a complete list of minikube binaries.

Operating system

Architecture

Release type

Installer type

To install the latest minikube **stable** release on **ARM64 macOS** using **binary download**:

```
curl -LO https://storage.googleapis.com/minikube/releases/latest/minikube-darwin-arm64
sudo install minikube-darwin-arm64 /usr/local/bin/minikube
```

Instalación kubectl

- Abrir una terminal para ejecutar el curl que nos proporcionó la pagina web “curl -LO https://storage.googleapis.com/minikube/releases/latest/minikube-darwin-arm64 sudo install minikube-darwin-arm64 /usr/local/bin/minikube”.

```
(base) danielflorezlopez@192 ~ % curl -LO https://storage.googleapis.com/minikube/releases/latest/minikube-darwin-arm64  
[sudo install minikube-darwin-arm64 /usr/local/bin/minikube  
 % Total    % Received % Xferd  Average Speed   Time   Time  Current  
          Dload Upload   Total Spent   Left Speed  
 100  100M  100  100M    0     0  31.7M      0  0:00:03  0:00:03 --:--:-- 31.7M]
```

Instalación kubectl

- Iniciar el minikube con el comando “minikube start”.

```
(base) danielflorezlopez@192 ~ % minikube start
😄  minikube v1.34.0 on Darwin 15.0.1 (arm64)
⭐  Automatically selected the docker driver
🔧  Using Docker Desktop driver with root privileges
👍  Starting "minikube" primary control-plane node in "minikube" cluster
🔄  Pulling base image v0.0.45 ...
💻  Downloading Kubernetes v1.31.0 preload ...
    > gcr.io/k8s-minikube/kicbase...: 441.45 MiB / 441.45 MiB  100.00% 28.13 M
    > preloaded-images-k8s-v18-v1...: 307.61 MiB / 307.61 MiB  100.00% 16.96 M
🔥  Creating docker container (CPUs=2, Memory=4000MB) ...
🌐  Preparing Kubernetes v1.31.0 on Docker 27.2.0 ...
    ■ Generating certificates and keys ...
    ■ Booting up control plane ...
    ■ Configuring RBAC rules ...
🔗  Configuring bridge CNI (Container Networking Interface) ...
🔑  Verifying Kubernetes components...
    ■ Using image gcr.io/k8s-minikube/storage-provisioner:v5
🌟  Enabled addons: storage-provisioner, default-storageclass
🚀  Done! kubectl is now configured to use "minikube" cluster and "default" namespace by default
```

Verificar instalación kubectl

- Para validar que se instaló correctamente el kubectl, ejecutar el comando “kubectl version --client”.

```
(base) danielflorezlopez@192 ~ % kubectl version --client
Client Version: v1.30.2
Kustomize Version: v5.0.4-0.20230601165947-6ce0bf390ce3
```

Verificar instalación kubectl

- Para validar que los Pod del minikube estén corriendo en el sistema, ejecutamos “kubectl get po -A”.

```
(base) danielflorezlopez@192 ~ % kubectl get po -A
NAMESPACE     NAME           READY   STATUS    RESTARTS   AGE
kube-system   coredns-6f6b679f8f-kp6r8   1/1     Running   0          9m57s
kube-system   etcd-minikube      1/1     Running   0          10m
kube-system   kube-apiserver-minikube  1/1     Running   0          10m
kube-system   kube-controller-manager-minikube  1/1     Running   0          10m
kube-system   kube-proxy-zcmvd      1/1     Running   0          9m58s
kube-system   kube-scheduler-minikube  1/1     Running   0          10m
kube-system   storage-provisioner_  1/1     Running   0          10m
```

Instalar Helm

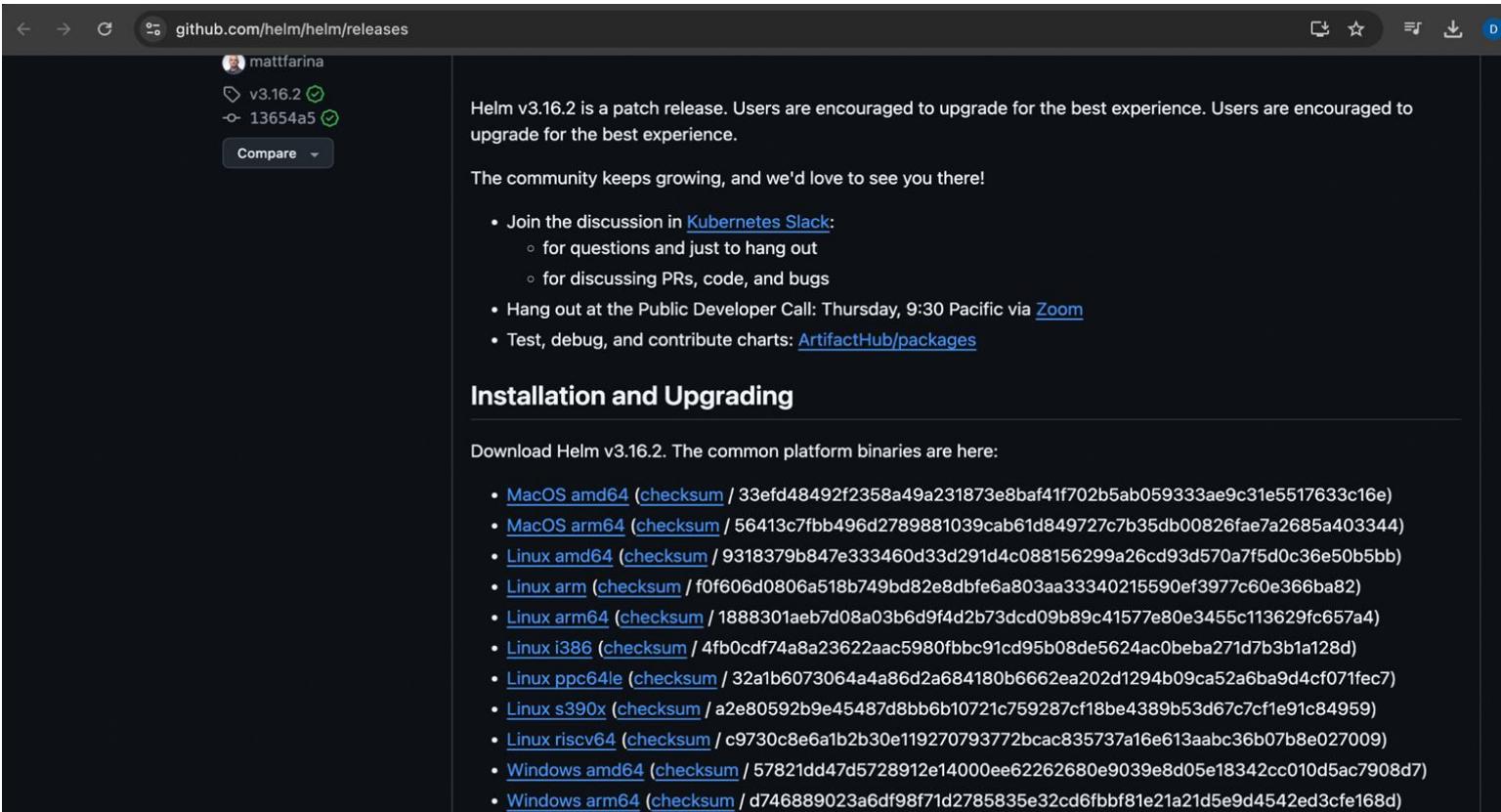
- Para instalar Helm ingresamos a la pagina web
<https://Helm.sh/es/docs/intro/install/>
- Le damos clic en “Descarga tu versión deseada”.

Cada **lanzamiento** de Helm proporciona binarios de lanzamiento para una variedad de sistemas operativos. Estas versiones binarias se pueden descargar e instalar manualmente.

1. Descarga tu **versión deseada**
2. Desempaquétala (`tar -zxvf helm-v3.0.0-linux-amd64.tar.gz`)
3. Encuentra el binario `helm` en el directorio desempaquetado, y muévelo a su destino deseado (`mv linux-amd64/helm /usr/local/bin/helm`)

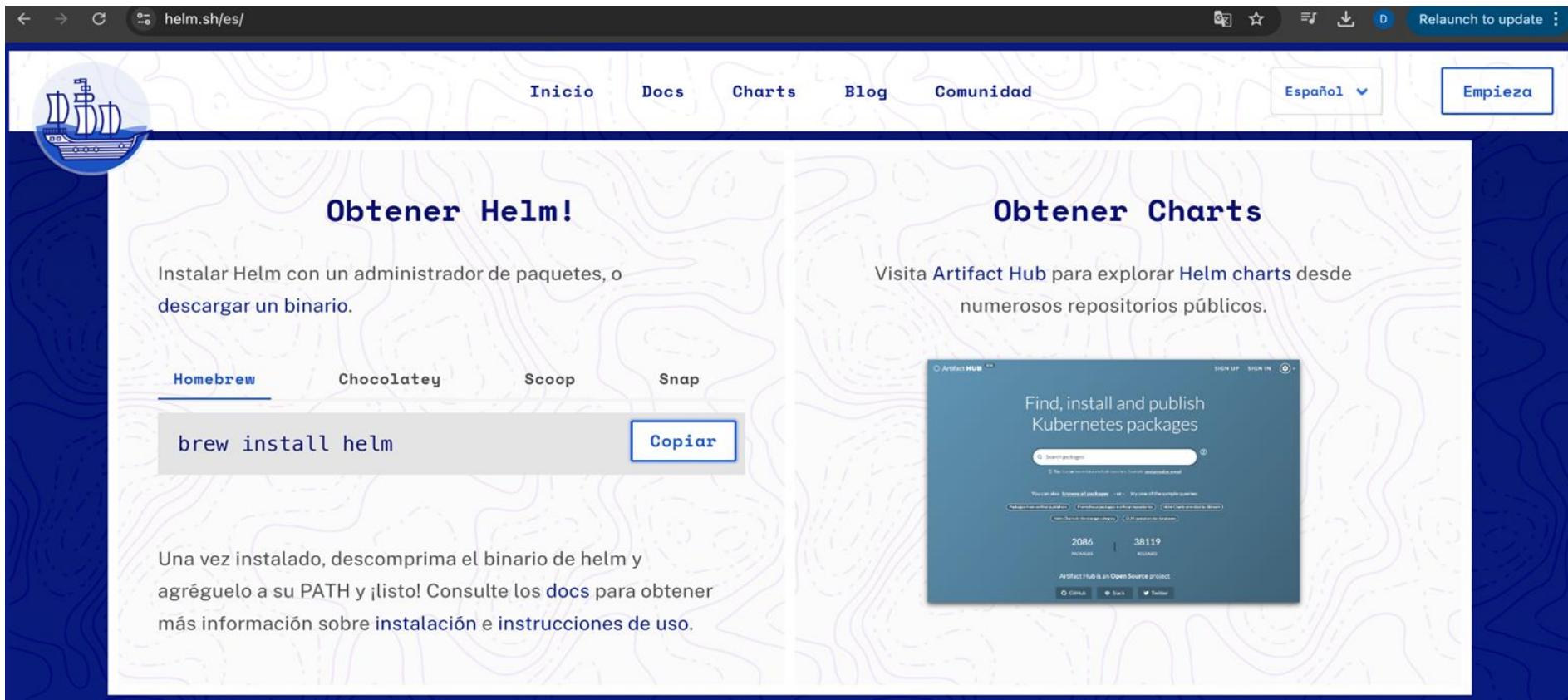
Instalar Helm

- En la ventana emergente, seleccionamos el instalador que corresponda a nuestro SO.



Instalar Helm

- Otra alternativa es instalarlo por el terminal como lo sugiere en la página oficial.



Instalar Helm

- Ejecutar el comando “brew install helm”.

```
(base) danielflorezlopez@192 ~ % brew install helm

==> Downloading https://ghcr.io/v2/homebrew/core/helm/manifests/3.16.2
#####
# 100.0%
==> Fetching helm
==> Downloading https://ghcr.io/v2/homebrew/core/helm/blobs/sha256:13ad3314a8d2eda0a9fdc822adb7117107cb1df214012245370e
#####
# 100.0%
==> Pouring helm--3.16.2.arm64_sequoia.bottle.tar.gz
==> Caveats
zsh completions have been installed to:
  /opt/homebrew/share/zsh/site-functions
==> Summary
🍺 /opt/homebrew/Cellar/helm/3.16.2: 66 files, 55.4MB
==> Running `brew cleanup helm`...
Disable this behaviour by setting HOMEBREW_NO_INSTALL_CLEANUP.
Hide these hints with HOMEBREW_NO_ENV_HINTS (see `man brew`).
(base) danielflorezlopez@192 ~ %
```

Instalar Helm

- Para validar que se instaló correctamente, ejecutamos el comando “helm version”.

```
(base) danielflorezlopez@192 ~ % helm version  
version.BuildInfo{Version:"v3.16.2", GitCommit:"13654a52f7c70a143b1dd51416d633e1071faffb", GitTreeState:"dirty", GoVersion:"go1.23.2"}  
(base) danielflorezlopez@192 ~ %
```

1.2. Helm Concepts (Labs-Helm Concepts)

- Charts: Son paquetes que nos da Helm, los cuales tienen artefactos o información que se necesita para desplegar una aplicación en Kubernetes.
- Repositories: Espacio de almacenamiento en donde se guardan los charts de Helm.
- Releases: Instancia de un Charts desplegado en un clúster de k8s.
- Template: Archivo que define los recursos que se van a generar en el Kubernetes.
- Hooks: Scripts que se ejecutan en diferentes ciclos del realse.

Agregar repositorio

- Antes de crear charts con Helm es necesario agregar un repositorio, para esto vamos a usar el comando:
- “**helm repo add bitnami https://charts.bitnami.com/bitnami**“.

```
(base) danielflorezlopez@192 ~ % helm repo add bitnami https://charts.bitnami.com/bitnami  
"bitnami" has been added to your repositories  
(base) danielflorezlopez@192 ~ %
```

Actualizar repositorio

- Es importante mantener actualizados los repositorios para que los charts estén disponibles.
- “**helm repo update**“.

```
(base) danielflorezlopez@192 ~ % helm repo update  
Hang tight while we grab the latest from your chart repositories...  
...Successfully got an update from the "bitnami" chart repository  
Update Complete. *Happy Helming!*  
(base) danielflorezlopez@192 ~ %
```

Creando un chart

Para desplegar un chart con Helm en Kubernetes se sigue la siguiente estructura:

- “Helm install <release-name> <chart>”

<release-name> es el nombre de la aplicación o release.

<chart> es el paquete preconfigurado que vamos a instalar.

Ejemplos:

Ejecutamos por la siguiente terminal el comando “Helm install my-app bitnami/nginx”.

Creando un chart

```
((base) danielflorezlopez@192 ~ % helm install my-app bitnami/nginx
NAME: my-app
LAST DEPLOYED: Fri Oct 18 23:43:35 2024
NAMESPACE: default
STATUS: deployed
REVISION: 1
TEST SUITE: None
NOTES:
CHART NAME: nginx
CHART VERSION: 18.2.3
APP VERSION: 1.27.2

** Please be patient while the chart is being deployed **
NGINX can be accessed through the following DNS name from within your cluster:

my-app-nginx.default.svc.cluster.local (port 80)

To access NGINX from outside the cluster, follow the steps below:

1. Get the NGINX URL by running these commands:

NOTE: It may take a few minutes for the LoadBalancer IP to be available.
      Watch the status with: 'kubectl get svc --namespace default -w my-app-nginx'

      export SERVICE_PORT=$(kubectl get --namespace default -o jsonpath=".spec.ports[0].port" services my-app-nginx)
      export SERVICE_IP=$(kubectl get svc --namespace default my-app-nginx -o jsonpath='{.status.loadBalancer.ingress[0].ip}')
      echo "http://${SERVICE_IP}:${SERVICE_PORT}"

WARNING: There are "resources" sections in the chart not set. Using "resourcesPreset" is not recommended for production
. For production installations, please set the following values according to your workload needs:
  - cloneStaticSiteFromGit.gitSync.resources
  - resources
+info https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/
(base) danielflorezlopez@192 ~ %
```

Listar los charts

Para listar los charts que tenemos, utilizamos el comando “helm list”.

(base) danielflorezlopez@192 ~ % helm list							
NAME	NAMESPACE	REVISION	UPDATED	STATUS	CHART	APP	VER
my-app	default	1	2024-10-18 23:43:35.079975 -0500 -05	deployed	nginx-18.2.3		1.27.2

Listar Pod

Para listar los Pods creados por Helm, usamos el comando “kubectl get pods”.

```
(base) danielflorezlopez@192 ~ % kubectl get pods
```

NAME	READY	STATUS	RESTARTS	AGE
my-app-nginx-84fd446495-tnvxh	1/1	Running	0	8m23s

Eliminar un release

Para eliminar un release, utilizamos el comando “helm uninstall <release-name>”.

```
(base) danielflorezlopez@192 ~ % helm uninstall my-app  
release "my-app" uninstalled  
(base) danielflorezlopez@192 ~ % helm list  
NAME      NAMESPACE      REVISION      UPDATED      STATUS      CHART      APP VERSION  
(base) danielflorezlopez@192 ~ %
```

1.3. Azure Kubernetes Service (AKS)

- Azure Kubernetes Service (AKS) es una plataforma administrada que implementa y gestiona los contenedores en Kubernetes.
- Permite a las aplicaciones que escalen y gestionen aplicaciones en la nube de manera eficiente, con control sobre la infraestructura, mientras Microsoft.

1.4. ¿Qué es AKS?

Azure Kubernetes Service (AKS) es un servicio de orquestación de contenedores en la nube de Azure que simplifica la implementación y administración de aplicaciones en contenedores utilizando Kubernetes.



1.5. Características principales

- **Gestión automatizada:** AKS gestiona automáticamente el clúster de Kubernetes.
- **Integración con Azure:** Integración nativa con otros servicios de Azure.
- **Escalabilidad:** Escala de manera flexible en función de la demanda.
- **Seguridad integrada:** Ofrece autenticación y control de acceso basado en roles.
- **Monitorización:** AKS incluye herramientas de monitoreo como Azure Monitor.
- **Alta disponibilidad:** Asegura disponibilidad continua de aplicaciones.
- **Facilidad de uso:** Simplifica el despliegue de Kubernetes.

Práctica 1.1 Creación servicio ECS

Objetivo:

- Crear un chart personalizable con Helm.
- Desplegar servicios con Helm.
- Crear un chart personalizable que tenga como objetivo desplegar el microservicio de correos electrónicos por medio de Helm.

Planteamiento e instrucciones:

Realiza un seguimiento de los pasos indicados en la *Guía de Laboratorios* para llevar a cabo la tarea correspondiente en el siguiente enlace: https://netec-mx.github.io/DOCK_KUB_AVA/



Tiempo para esta actividad:
30 Minutos.

Resultado esperado

Correcto deploy.

```
● (base) danielflorezlopez@192 LAB % helm install ms-email ./ms-email
NAME: ms-email
LAST DEPLOYED: Sun Nov  3 18:07:28 2024
NAMESPACE: default
STATUS: deployed
REVISION: 1
NOTES:
1. Get the application URL by running these commands:
   export NODE_PORT=$(kubectl get --namespace default -o jsonpath=".spec.ports[0].nodePort" services ms-email)
   export NODE_IP=$(kubectl get nodes --namespace default -o jsonpath=".items[0].status.addresses[0].address")
   echo http://$NODE_IP:$NODE_PORT
○ (base) danielflorezlopez@192 LAB %
```

1.6. Costos AKS

Aunque Azure AKS no tiene costo por la gestión del servicio, los gastos provienen de la infraestructura subyacente. El modelo de precios se basa en la cantidad de nodos utilizados y en los recursos asignados a estos, como CPU, memoria y almacenamiento.

Costos AKS

- **Nodos del clúster:**
 - Se cobra por los nodos utilizados para ejecutar las cargas de trabajo en contenedores.
 - Los precios varían según el tipo y tamaño de la VM utilizada, lo que permite escalar desde configuraciones básicas hasta nodos de alta capacidad.
- **Escalado automático:**
 - El escalado automático aumenta los costos a medida que se activan más nodos durante picos de demanda.
 - permite reducir los costos al desactivar nodos cuando baja la carga.

1.7. Corriendo contenedores en multi-máquinas EC2 o multi-host

AWS EC2 (Elastic Compute Cloud) es un servicio de Amazon Web Services que permite crear y gestionar instancias de servidores virtuales en la nube. Estos servidores, conocidos como instancias, proporcionan recursos de cómputo escalables según las necesidades del usuario, lo que facilita ejecutar aplicaciones sin necesidad de gestionar hardware.

Crear artefacto del proyecto

Para empaquetar el proyecto, ya sea en Maven o Gradle, debemos crear el artefacto por medio de los siguientes comandos o recurrir a la ayuda del IDE.

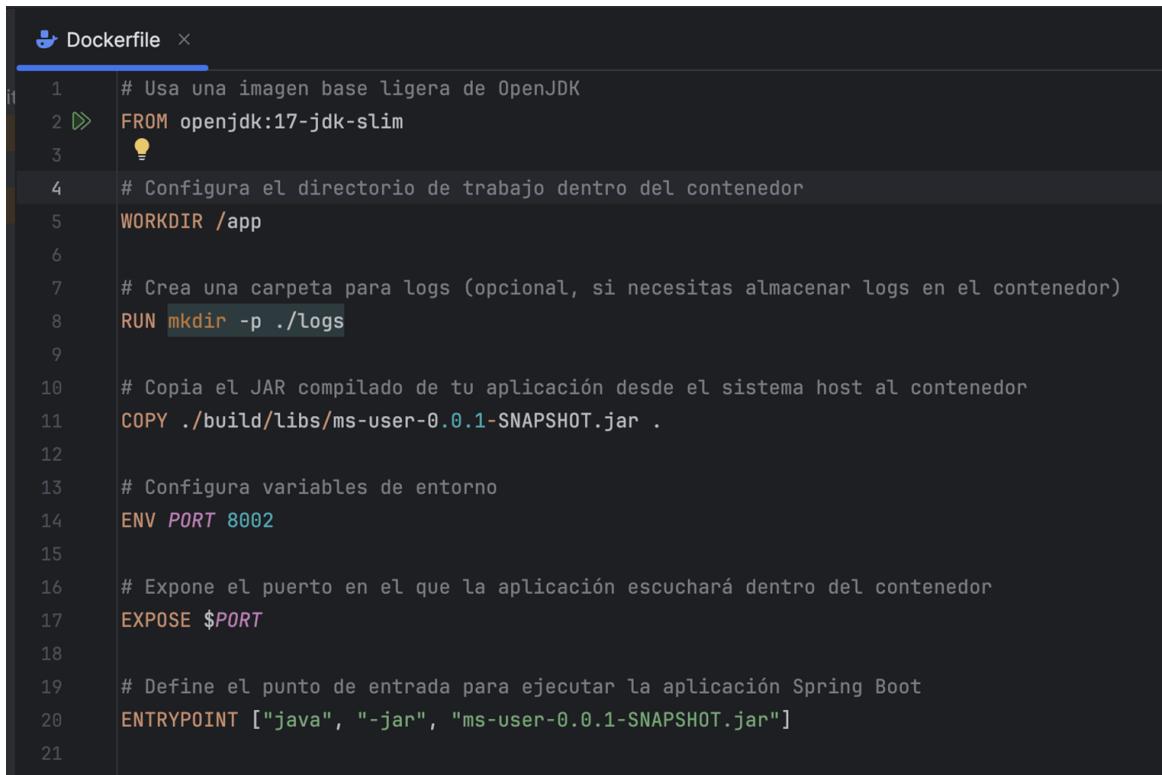
- ./gradlew clean build

The screenshot shows a dark-themed IDE interface with several panels:

- Project:** Shows the directory structure of the project, including .gradle, .idea, build, gradle, src, .env, .gitignore, Dockerfile, gradlew, gradlew.bat, README.txt, settings.gradle, External Libraries, and Scratches and Consoles.
- Dockerfile:** A file containing Docker configuration.
- .env:** An environment variable file.
- application.yml:** A configuration file with the following content:spring:
 datasource:
 url: jdbc:mysql://\${DB_HOST}:3306}/\${DB_DATABASE}:user-db
 username: \${DB_USERNAME:root}
 password: \${DB_PASSWORD:test123456}
 driver-class-name: com.mysql.cj.jdbc.Driver
 jpa:
 hibernate:
 ddl-auto: update
 properties:
 hibernate:
 dialect: org.hibernate.dialect.MySQLDialect
 springdoc:
 api-docs:
 path: /netec-doc
 show-actuator: true
 packages-to-scan: com.ms.user.controller.v2
 control:
 exception:
- Gradle:** A panel showing the available Gradle tasks for the project. The "build" task is highlighted.
- Run:** A panel showing the build task has been successful.

Crear Dockerfile

Dockerfile es un archivo que nos ayuda a contenerizar los desarrollos para ser ejecutados en un ambiente de desarrollo, pruebas o producción.



A screenshot of a code editor window titled "Dockerfile". The file contains the following Dockerfile code:

```
1 # Usa una imagen base ligera de OpenJDK
2 FROM openjdk:17-jdk-slim
3
4 # Configura el directorio de trabajo dentro del contenedor
5 WORKDIR /app
6
7 # Crea una carpeta para logs (opcional, si necesitas almacenar logs en el contenedor)
8 RUN mkdir -p ./logs
9
10 # Copia el JAR compilado de tu aplicación desde el sistema host al contenedor
11 COPY ./build/libs/ms-user-0.0.1-SNAPSHOT.jar .
12
13 # Configura variables de entorno
14 ENV PORT 8002
15
16 # Expone el puerto en el que la aplicación escuchará dentro del contenedor
17 EXPOSE $PORT
18
19 # Define el punto de entrada para ejecutar la aplicación Spring Boot
20 ENTRYPOINT ["java", "-jar", "ms-user-0.0.1-SNAPSHOT.jar"]
21
```

Creando imagen

A partir del Dockerfile podemos crear una imagen del servicio de usuarios, el cual tendrá todo el desarrollo que podrá ser desplegado como contenedores en otros sistemas o para este caso AWS.

Para esto, el comando que vamos a utilizar es:

```
docker build -t ms-user-app:latest.
```

Creando imagen

```
Terminal Local × Local (2) × + ×

(base) danielflorezlopez@192 ms-user % docker build -t ms-user-app:latest .
[+] Building 2.2s (9/9) FINISHED
    => [internal] load build definition from Dockerfile                               0.0s
    => => transferring dockerfile: 716B                                         0.0s
    => [internal] load metadata for docker.io/library/openjdk:17-jdk-slim           1.6s
    => [internal] load .dockerignore                                              0.0s
    => => transferring context: 2B                                         0.0s
    => [1/4] FROM docker.io/library/openjdk:17-jdk-slim@sha256:aaa3b3cb27e3e520b8f116863d0580c438ed55ecfa0bc126b41f68c3f62f9774  0.0s
    => [internal] load build context                                              0.4s
    => => transferring context: 53.96MB                                         0.4s
    => CACHED [2/4] WORKDIR /app                                              0.0s
    => CACHED [3/4] RUN mkdir -p ./logs                                         0.0s
    => [4/4] COPY ./build/libs/ms-user-0.0.1-SNAPSHOT.jar .                      0.1s
    => exporting to image                                                       0.1s
    => => exporting layers                                              0.1s
    => => writing image sha256:b5f643b2513ba2b699c73d03e97d2044cf95a257de5847b3f2b7398398b2dfcf  0.0s
    => => naming to docker.io/library/ms-user-app:latest                         0.0s

View build details: docker-desktop://dashboard/build/desktop-linux/desktop-linux/mcgteb5balv08r674pfh328b3

1 warning found (use docker --debug to expand):
- LegacyKeyValueFormat: "ENV key=value" should be used instead of legacy "ENV key value" format (line 14)

What's next:
  View a summary of image vulnerabilities and recommendations → docker scout quickview
(base) danielflorezlopez@192 ms-user %
```

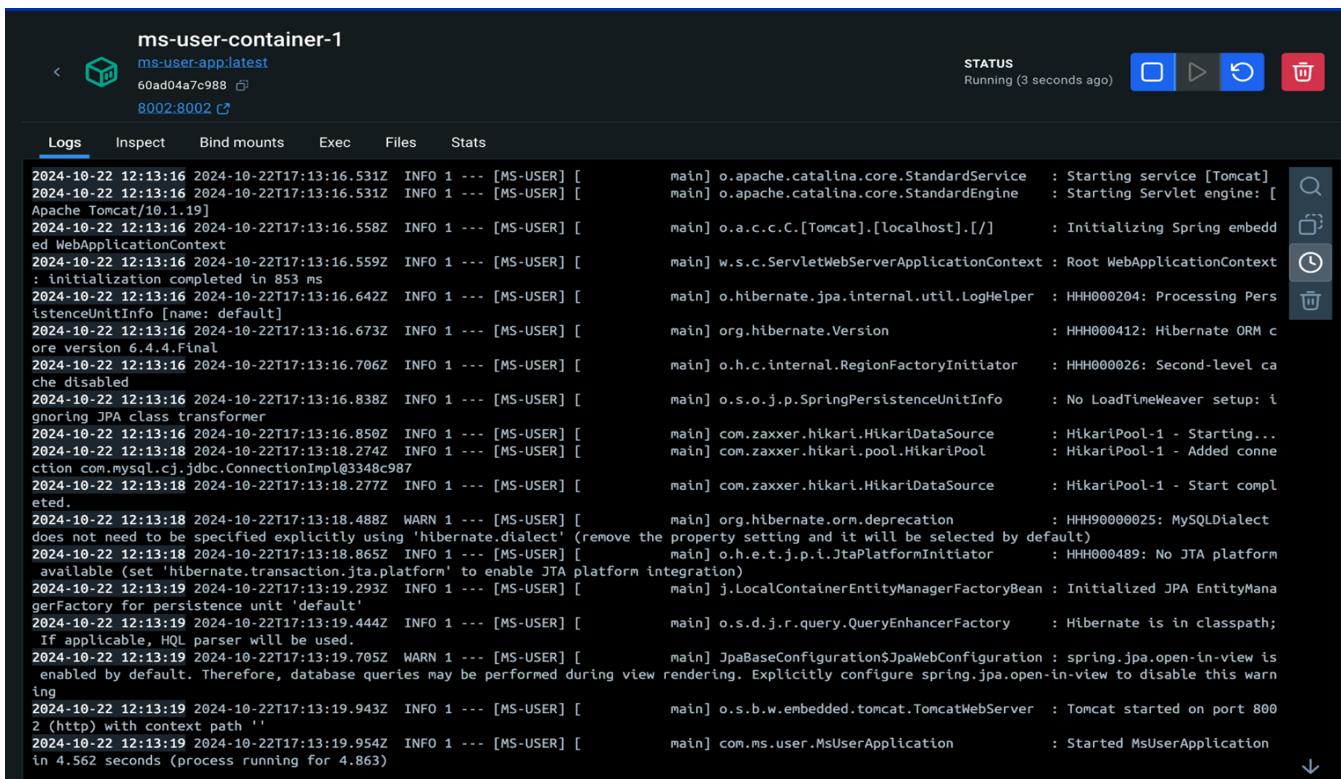
Creando contenedor

- Los contenedores se crean a partir de las imágenes que están disponibles, ya sea en un ambiente local, Docker Hub o un ECR. Para crear el contenedor vamos a realizar una variación, la cual va a ser:
- Pasar por parámetros unas variables de entorno.
- `docker run -d --env-file .env -p 8002:8002 --name ms-user-container-1 ms-user-app:latest`

```
(base) danielflorezlopez@192 ms-user % docker run -d --env-file .env -p 8002:8002 --name ms-user-container-1 ms-user-app:latest  
9fa6a522fa452ae98d44393ff69e62821bc945bb6461f08f195a935e24ff8c3f  
(base) danielflorezlopez@192 ms-user %
```

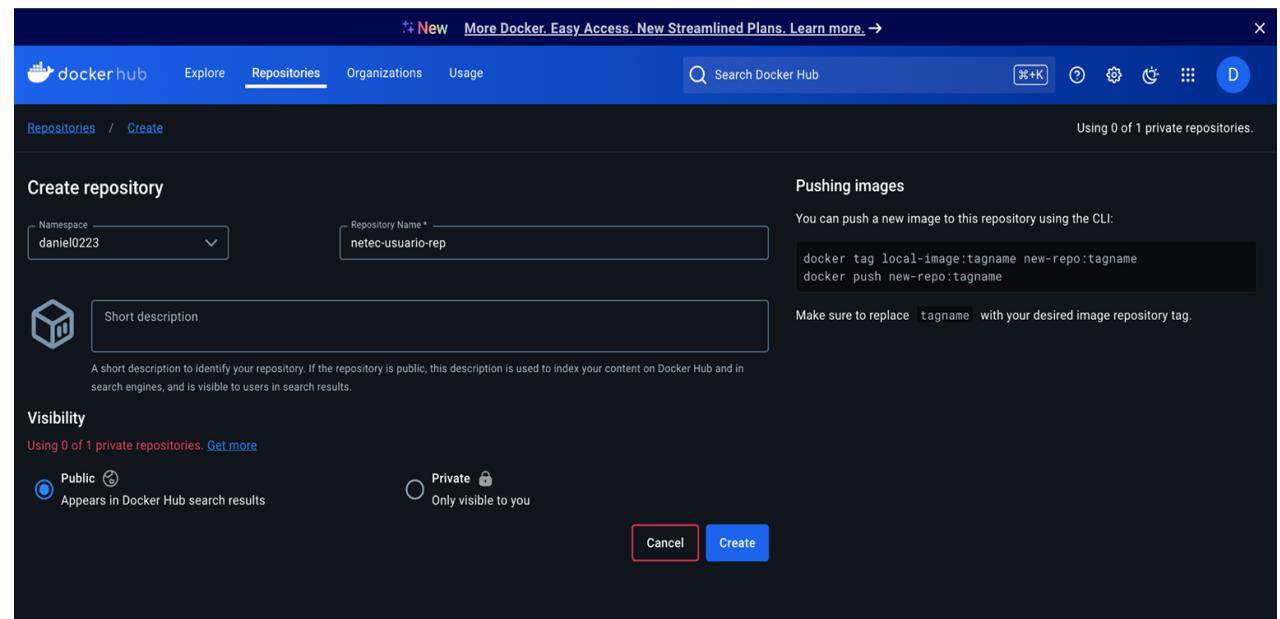
Creando contenedor

Para validar que se creó correctamente el contenedor en nuestra máquina local, vamos al desktop de Docker y validamos el estado del contenedor creado.



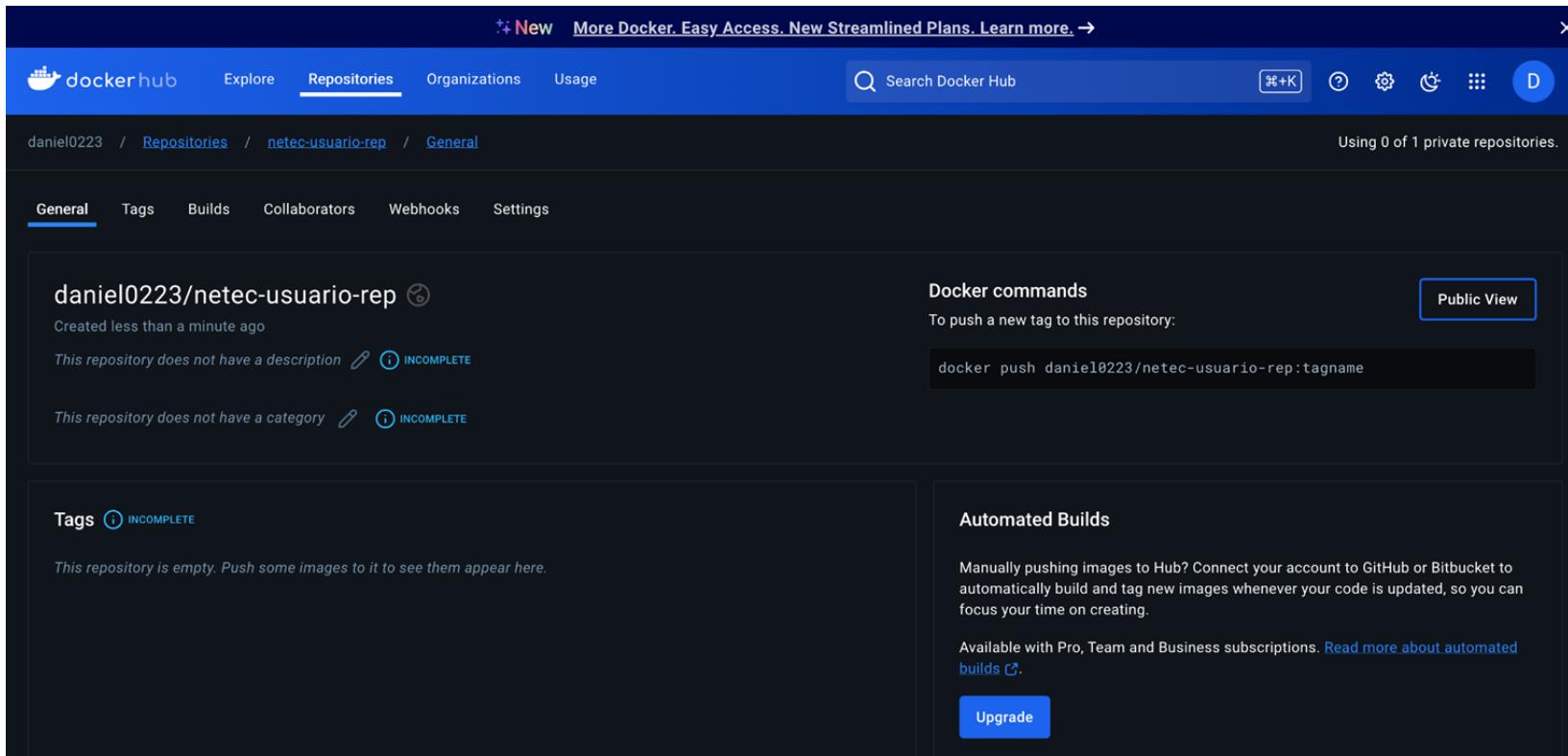
Crear repositorio en Docker Hub

- Docker Hub es una plataforma creada por Docker para publicar los contenedores de manera privada o pública.
- Para crear un repositorio en Docker Hub es necesario tener una cuenta ya creada.
- Una vez en la cuenta, ingresamos a la opción “create repository” y le damos un nombre.



Crear repositorio en Docker Hub

El repositorio se creará correctamente.



Subir imagen al repositorio

Para poder subir la imagen en el repositorio creado anteriormente, es necesario loguearnos en Docker Hub desde la consola, para esto ejecutamos el comando:

- docker login

```
(base) danielflorezlopez@192 ms-user % docker login
Log in with your Docker ID or email address to push and pull images from Docker Hub. If you don't have a Docker ID, head over to https://hub.docker.com/ to create one.
You can log in with your password or a Personal Access Token (PAT). Using a limited-scope PAT grants better security and is required for organizations using SSO. Learn more at
https://docs.docker.com/go/access-tokens/

Username: daniel0223@hotmail.es
Password:
Login Succeeded
(base) danielflorezlopez@192 ms-user %
```

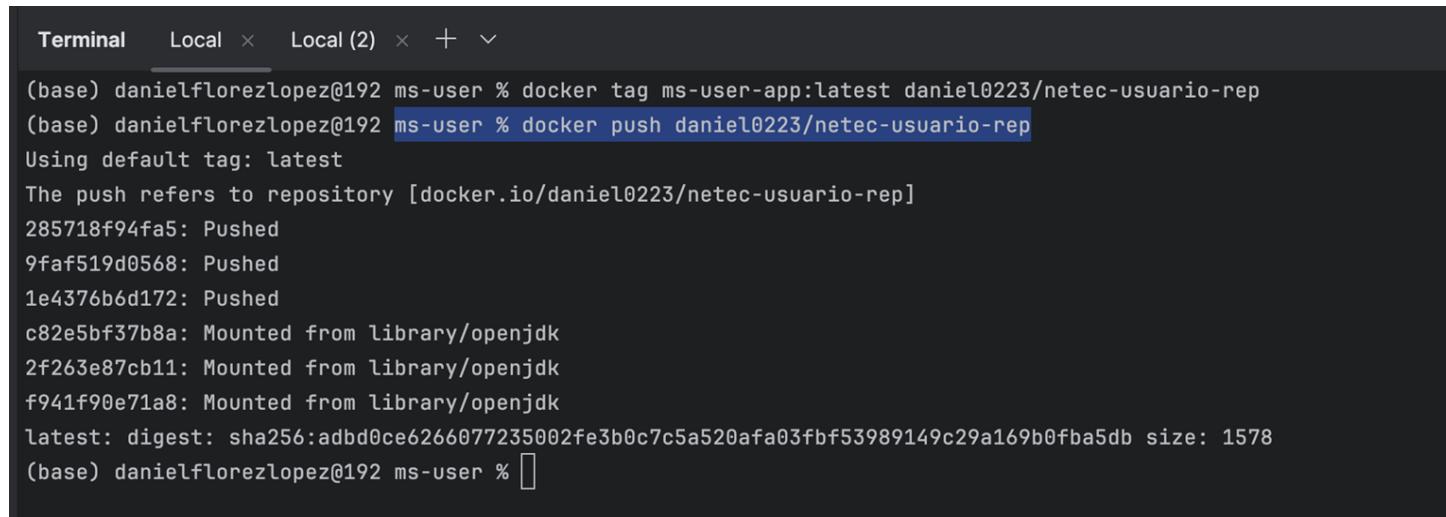
Subir imagen al repositorio

Una vez logueados satisfactoriamente, creamos un tag de nuestra imagen local para subirla a Docker Hub.

- docker tag ms-user-app:latest daniel0223/netec-usuario-rep

Luego, realizamos un push para subir la imagen.

- ms-user % docker push daniel0223/netec-usuario-rep



The screenshot shows a terminal window with two tabs: "Terminal" and "Local". The "Terminal" tab is active and displays the following command sequence:

```
(base) danielflorezlopez@192 ms-user % docker tag ms-user-app:latest daniel0223/netec-usuario-rep
(base) danielflorezlopez@192 ms-user % docker push daniel0223/netec-usuario-rep
Using default tag: latest
The push refers to repository [docker.io/daniel0223/netec-usuario-rep]
285718f94fa5: Pushed
9faf519d0568: Pushed
1e4376b6d172: Pushed
c82e5bf37b8a: Mounted from library/openjdk
2f263e87cb11: Mounted from library/openjdk
f941f90e71a8: Mounted from library/openjdk
latest: digest: sha256:abd0ce6266077235002fe3b0c7c5a520afa03fbf53989149c29a169b0fba5db size: 1578
(base) danielflorezlopez@192 ms-user %
```

Subir imagen al repositorio

The screenshot shows a Docker Hub repository page. At the top, the URL is hub.docker.com/repository/docker/daniel0223/netec-usuario-rep/general. The page title is "daniel0223 / Repositories / netec-usuario-rep / General". A banner at the top says "New More Docker. Easy Access. New Streamlined Plans. Learn more." Below the banner, there are tabs for "Explore", "Repositories" (which is selected), "Organizations", and "Usage". A search bar says "Search Docker Hub". On the right, there are icons for "Kubernetes", "Settings", "Collaborators", "Webhooks", and a "D" button.

The main content area shows the repository "daniel0223/netec-usuario-rep". It was last pushed 1 minute ago. There is a note: "This repository does not have a description" with an "INCOMPLETE" status. Another note says "This repository does not have a category" with an "INCOMPLETE" status.

Docker commands: To push a new tag to this repository, use the command: docker push daniel0223/netec-usuario-rep:tagname. A "Public View" button is available.

Tags: This repository contains 1 tag(s). The table shows one tag:

Tag	OS	Type	Pulled	Pushed
latest	⚠️	Image	a few seconds ago	a minute ago

[See all](#)

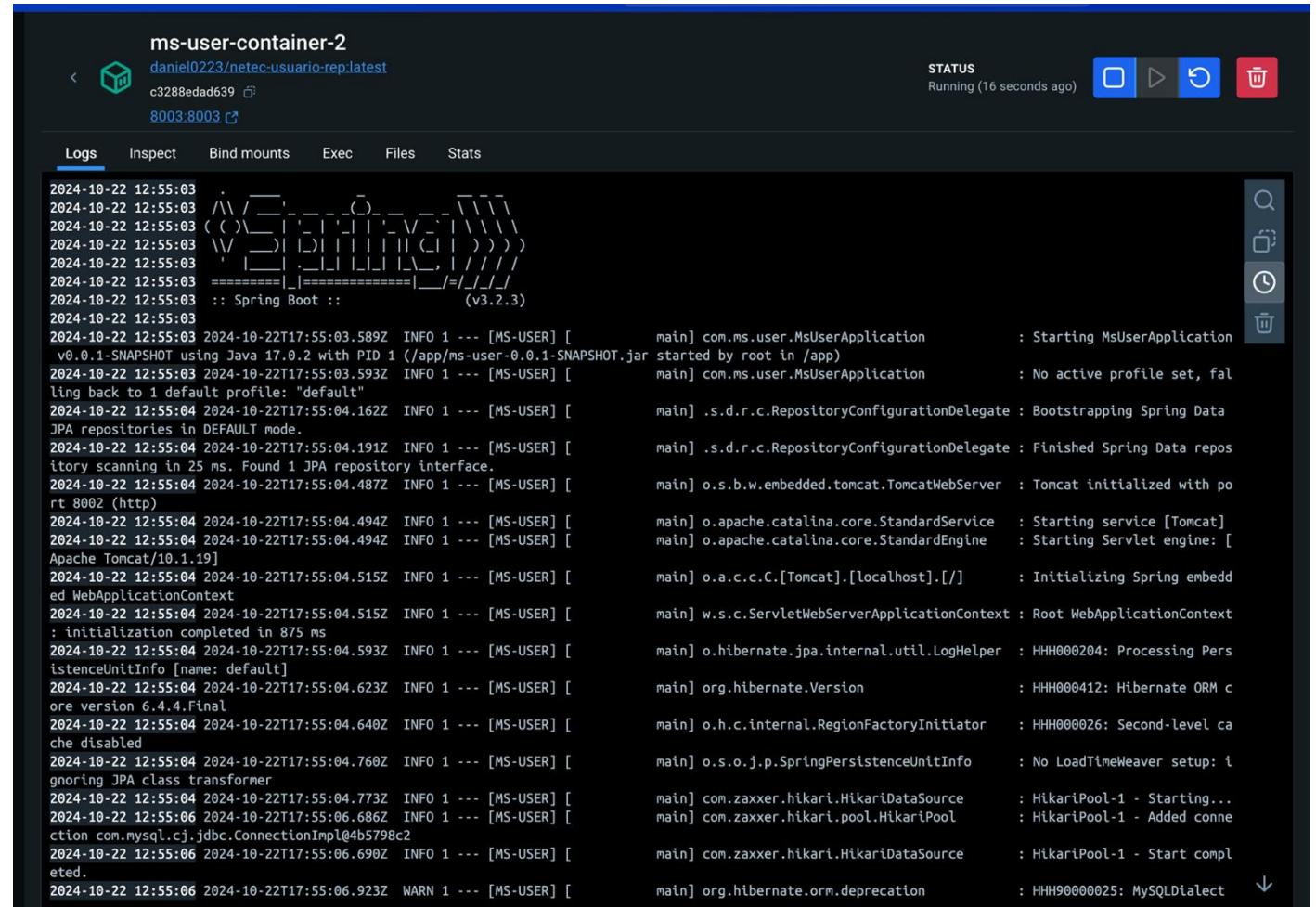
Automated Builds: Manually pushing images to Hub? Connect your account to GitHub or Bitbucket to automatically build and tag new images whenever your code is updated, so you can focus your time on creating. Available with Pro, Team and Business subscriptions. [Read more about automated builds](#).

[Upgrade](#)

Validación imagen Docker Hub

Para validar que la imagen se creó correctamente, vamos a crear el contenedor en nuestra máquina local.

- docker run -d --env-file .env -p 8003:8003 --name ms-user-container-2 daniel0223/netec-usuario-rep:latest



The screenshot shows the Docker Container Details page for 'ms-user-container-2'. The container is running and has been running for 16 seconds. The logs tab is selected, displaying the startup logs of the application. The logs show the application starting up, initializing Spring Data JPA repositories, bootstrapping Spring Data repositories, and initializing the Tomcat web server. It also shows Hibernate and MySQL dialect configuration details.

```
2024-10-22 12:55:03 2024-10-22T17:55:03.589Z INFO 1 --- [MS-USER] [main] com.ms.user.MsUserApplication : Starting MsUserApplication
2024-10-22 12:55:03 v0.0.1-SNAPSHOT using Java 17.0.2 with PID 1 (/app/ms-user-0.0.1-SNAPSHOT.jar started by root in /app)
2024-10-22 12:55:03 2024-10-22T17:55:03.593Z INFO 1 --- [MS-USER] [main] com.ms.user.MsUserApplication : No active profile set, falling back to default profile: "default"
2024-10-22 12:55:04 2024-10-22T17:55:04.162Z INFO 1 --- [MS-USER] [main] s.d.r.c.RepositoryConfigurationDelegate : Bootstrapping Spring Data JPA repositories in DEFAULT mode.
2024-10-22 12:55:04 2024-10-22T17:55:04.191Z INFO 1 --- [MS-USER] [main] s.d.r.c.RepositoryConfigurationDelegate : Finished Spring Data repository scanning in 25 ms. Found 1 JPA repository interface.
2024-10-22 12:55:04 2024-10-22T17:55:04.487Z INFO 1 --- [MS-USER] [main] o.s.w.b.w.embedded.tomcat.TomcatWebServer : Tomcat initialized with port 8002 (http)
2024-10-22 12:55:04 2024-10-22T17:55:04.494Z INFO 1 --- [MS-USER] [main] o.apache.catalina.core.StandardService : Starting service [Tomcat]
2024-10-22 12:55:04 2024-10-22T17:55:04.494Z INFO 1 --- [MS-USER] [main] o.apache.catalina.core.StandardEngine : Starting Servlet engine: []
2024-10-22 12:55:04 2024-10-22T17:55:04.515Z INFO 1 --- [MS-USER] [main] o.a.c.c.C.[Tomcat].[localhost].[] : Initializing Spring embedded WebApplicationContext
2024-10-22 12:55:04 2024-10-22T17:55:04.515Z INFO 1 --- [MS-USER] [main] w.s.c.ServletWebServerApplicationContext : Root WebApplicationContext
2024-10-22 12:55:04 2024-10-22T17:55:04.593Z INFO 1 --- [MS-USER] [main] o.hibernate.jpa.internal.util.LogHelper : HHH000204: Processing PersistenceUnitInfo [name: default]
2024-10-22 12:55:04 2024-10-22T17:55:04.623Z INFO 1 --- [MS-USER] [main] org.hibernate.Version : HHH000412: Hibernate ORM core version 6.4.4.Final
2024-10-22 12:55:04 2024-10-22T17:55:04.640Z INFO 1 --- [MS-USER] [main] o.h.c.internal.RegionFactoryInitiator : HHH000026: Second-level cache disabled
2024-10-22 12:55:04 2024-10-22T17:55:04.760Z INFO 1 --- [MS-USER] [main] o.s.o.j.p.SpringPersistenceUnitInfo : No LoadTimeWeaver setup: ignoring JPA class transformer
2024-10-22 12:55:04 2024-10-22T17:55:04.773Z INFO 1 --- [MS-USER] [main] com.zaxxer.hikari.HikariDataSource : HikariPool-1 - Starting...
2024-10-22 12:55:06 2024-10-22T17:55:06.686Z INFO 1 --- [MS-USER] [main] com.zaxxer.hikari.pool.HikariPool : HikariPool-1 - Added connection com.mysql.cj.jdbc.ConnectionImpl@4b5798c2
2024-10-22 12:55:06 2024-10-22T17:55:06.690Z INFO 1 --- [MS-USER] [main] com.zaxxer.hikari.HikariDataSource : HikariPool-1 - Start completed.
2024-10-22 12:55:06 2024-10-22T17:55:06.923Z WARN 1 --- [MS-USER] [main] org.hibernate.orm.deprecation : HHHH90000025: MySQLDialect
```

Crear instancia EC2 en AWS

Nos dirigimos a la plataforma de AWS, donde buscamos el servicio de AWS EC2.

The screenshot shows the AWS EC2 console interface. On the left, a sidebar titled "Panel de EC2" lists various EC2 management options. The main content area is divided into several sections:

- Recursos:** Displays current EC2 resources in the US East (N. Virginia) region, including 1 instance running, 0 load balancers, 0 capacity reservations, 0 elastic IP addresses, 0 auto-scaling groups, 3 security groups, 0 location groups, 0 dedicated hosts, 1 instance, 0 snapshots, 1 key pair, and 1 volume.
- Lanzar la instancia:** A button labeled "Lanzar la instancia" (Launch instance) is highlighted in orange, indicating it is the primary action available.
- Estado del servicio:** Shows the service status as "Este servicio funciona con normalidad." (The service is functioning normally).
- Zonas:** Lists three availability zones: us-east-1a (use1-az6), us-east-1b (use1-az1), and us-east-1c (use1-az2).
- Atributos de la cuenta:** Includes sections for the default VPC (vpc-06ac1a7a05e67bba0), configuration options like data protection and zones, and additional links for documentation and support.
- Información adicional:** Provides links to introductory guides, documentation, forums, prices, and contact information.

Crear instancia EC2 en AWS

Ingresamos a la opción lanzar una instancia y añadimos la siguiente información:

- Nombre : EC2-NETEC (opcional)
- AMI : UBUNTU
- Arquitectura: 64 bits(x86)
- Tipo instancia : t2.micro

The screenshot shows the AWS Launch Wizard interface. At the top, there is a search bar with the placeholder text "Busque en nuestro catálogo completo que incluye miles de imágenes de sistemas operativos y aplicaciones". Below the search bar, there are tabs for "Recientes" and "Inicio rápido", with "Inicio rápido" being the active tab. A grid of icons for different AMIs is displayed, including Amazon Linux, macOS, Ubuntu, Windows, Red Hat, and SUSE Linux. To the right of the grid, there is a search icon and a link to "Buscar más AMI". Below the grid, a section titled "Imagenes de máquina de Amazon (AMI)" shows the selected AMI: "Ubuntu Server 24.04 LTS (HVM), SSD Volume Type". This section includes the AMI ID (ami-0866a3c8686eaebea), the architecture (64 bits (x86)), and details about virtualization (hvm), ENA support, and root device type (ebs). A note indicates it is "Apto para la capa gratuita". Below this, there is a "Descripción" section with the Canonical Ubuntu 24.04 arm64 image details. Further down, there are dropdown menus for "Arquitectura" (set to 64 bits (Arm)), "ID de AMI" (set to ami-0325498274077fac5), and "Nombre de usuario" (set to ubuntu). A green button labeled "Proveedor verificado" is also present. At the bottom, there is a section for selecting the "Tipo de instancia" (t4g.nano) and a note that additional costs apply for pre-installed software.

Crear instancia EC2 en AWS

- Par de claves: vicky (crear par de claves).
- Crear un grupo de seguridad todos los checks.
- Lanzar instancia.

The screenshot shows the configuration step for a new Amazon Linux 2023 AMI instance. It includes:

- Red:** Subred: Sin preferencias, Asignar automáticamente la IP pública.
- Firewall:** Firewall (grupos de seguridad) - Create a new security group named "launch-wizard-1".
 - Crear grupo de seguridad** (selected)
 - Seleccionar un grupo de seguridad existente**
- Reglas:** Permitir el tráfico de SSH desde Anywhere (0.0.0.0/0), Permitir el tráfico de HTTPS desde Internet, and Permitir el tráfico de HTTP desde Internet.
- Nota:** A warning message states: "Las reglas con origen 0.0.0.0/0 permiten que todas las direcciones IP tengan acceso a la instancia. Le recomendamos que configure las reglas del grupo de seguridad para permitir el acceso únicamente desde direcciones IP conocidas."
- Resumen:** Número de instancias: 1, Imagen de software (AMI): Amazon Linux 2023 AMI 2023.6.2..., Tipo de servidor virtual: t2.micro, Almacenamiento (volúmenes): Volúmenes: 1 (8 GiB).
- Botones:** Cancelar, Lanzar Instancia, Código de versión preliminar.

AWS EC2

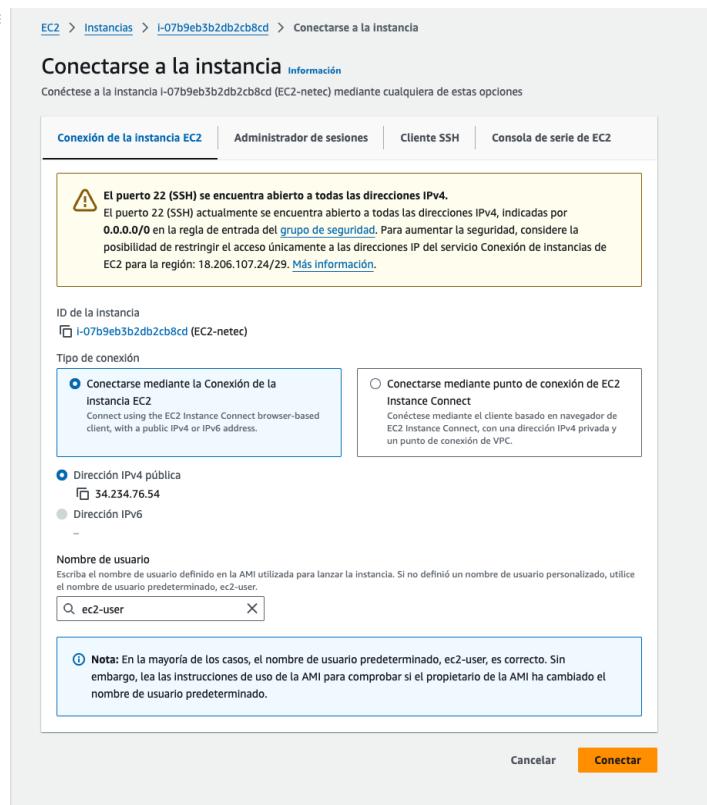
- Se crea la instancia correctamente.

The screenshot shows the AWS EC2 Instance Details page for instance i-07b9eb3b2db2cb8cd. The top navigation bar includes filters for 'EC2-netec' (selected), 'i-07b9eb3b2db2cb8cd', 'En ejecución' (Running), 't2.micro', '2/2 comprobador', 'Ver alarmas', and 'us-east-1d'. Below the header, the instance ID 'i-07b9eb3b2db2cb8cd (EC2-netec)' is displayed. The main content area has tabs for 'Detalles' (selected), 'Estado y alarmas', 'Monitoreo', 'Seguridad', 'Redes', 'Almacenamiento', and 'Etiquetas'. The 'Resumen de instancia' section contains the following details:

Detalles	Valor	Detalles	Valor	Detalles	Valor
ID de la instancia	i-07b9eb3b2db2cb8cd	Dirección IPv4 pública	34.234.76.54 dirección abierta	Direcciones IPv4 privadas	172.31.17.225
Dirección IPv6	-	Estado de la instancia	En ejecución	DNS de IPv4 pública	ec2-34-234-76-54.compute-1.amazonaws.com dirección abierta
Tipo de nombre de anfitrón	Nombre de IP: ip-172-31-17-225.ec2.internal	Nombre DNS de IP privada (solo IPv4)	ip-172-31-17-225.ec2.internal	Direcciones IP elásticas	-
Responder al nombre DNS de recurso privado	IPv4 (A)	Tipo de instancia	t2.micro	Hallazgo de AWS Compute Optimizer	Suscribirse a AWS Compute Optimizer para recibir recomendaciones.
Dirección IP asignada automáticamente	34.234.76.54 [IP pública]	ID de VPC	vpc-06ac1a7a05e67bba0	Más información	Más información
Rol de IAM	-	ID de subred	subnet-0fe4b3527e398ae2e	Nombre del grupo de Auto Scaling	-
IMDSv2	Required	ARN de instancia	arn:aws:ec2:us-east-1:754711774322:instance/i-07b9eb		

Instalar Docker en la EC2

- Para la instalación del Docker en EC2 es necesario ingresar a la instancia, para esto vamos a la opción de conectar.



The screenshot shows a terminal window on an Ubuntu 24.04.1 LTS (GNU/Linux 6.8.0-1016-aws aarch64) instance. The title bar indicates the URL is us-east-1.console.aws.amazon.com/ec2-instance-connect/ssh?region=us-east-1&connType=standard&instanceId=i-07b9eb3b2db2cb8cd. The terminal displays the following welcome message and system information:

```
Welcome to Ubuntu 24.04.1 LTS (GNU/Linux 6.8.0-1016-aws aarch64)

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

System information as of Tue Oct 22 18:26:09 UTC 2024

System load: 0.1 Temperature: -273.1 C
Usage of /: 23.0% of 6.71GB Processes: 142
Memory usage: 47% Users logged in: 0
Swap usage: 0% IPv4 address for ens5: 172.31.29.238

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

The list of available updates is more than a week old.
To check for new updates run: sudo apt update

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-172-31-29-238:~$
```

Instalar Docker en la EC2

Para la instalación vamos a la máquina oficial de Docker "<https://docs.docker.com/engine/install/ubuntu/>", seguimos los pasos o ejecutamos los siguientes comandos :

- sudo apt-get update
 - sudo apt-get install ca-certificates curl gnupg
 - sudo install -m 0755 -d /etc/apt/keyrings
 - curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /etc/apt/keyrings/docker.gpg
 - sudo chmod a+r /etc/apt/keyrings/docker.gpg
- ```
echo \
"deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.gpg]
https://download.docker.com/linux/ubuntu \
$(./etc/os-release && echo "$VERSION_CODENAME")" stable" | \
sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
```

# Instalar Docker en la EC2

- ```
sudo chmod a+r /etc/apt/keyrings/docker.gpg
echo \
"deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.gpg]
https://download.docker.com/linux/ubuntu \
$(./etc/os-release && echo "$VERSION_CODENAME")" stable" | \
sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
```
- sudo apt-get update
- sudo apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin
- sudo apt-get update
- sudo service docker start

Instalar Docker en la EC2

Para verificar la instalación, ejecutamos el comando:

- sudo docker images

```
ubuntu@ip-172-31-29-238:~$ sudo docker images
REPOSITORY      TAG          IMAGE ID      CREATED       SIZE
ubuntu@ip-172-31-29-238:~$ █
```

Creando contenedor en EC2

Una vez instalado correctamente Docker en la EC2, creamos un contenedor en la instancia de EC2 con el comando:

- docker run -d -p 8003:8003 --name ms-user-container-2 daniel0223/netec-usuario-rep:latest

```
ubuntu@ip-172-31-29-238:~$ sudo docker run -d -p 8003:8003 --name ms-user-container-2 daniel0223/netec-usuario-rep:latest
Unable to find image 'daniel0223/netec-usuario-rep:latest' locally
latest: Pulling from daniel0223/netec-usuario-rep
6d4a449ac69c: Pull complete
a59f13dc084e: Pull complete
1d5035d2d5c6: Pull complete
41a3a0f393c9: Pull complete
b29cefa58929: Pull complete
6e0ac3a6657c: Pull complete
Digest: sha256:14b4762659deb26af8d2ab6656224333ec6b6a3975b8a908eefca7780dd04e3
Status: Downloaded newer image for daniel0223/netec-usuario-rep:latest
b8f2c215d8315a071c12910a3ac273c0a350e752dd8214b0236ce6dca179f7f2
```

Validando backend

Ingresamos a la IP del la EC2 por el puerto 8083 y vemos el swagger del backend.

- <http://<ip>:8005/swagger-ui/index.html>

The screenshot shows the Swagger UI interface for the User API. The top navigation bar has a 'User' tab selected, which is described as 'API exposed for management all user'. Below the navigation, there is a list of API endpoints:

- GET /api/v2/user** get users (blue button)
- POST /api/v2/user** create user (green button)
- GET /api/v2/user/{id}** find by id user (blue button)
- DELETE /api/v2/user/{id}** delete user (red button)
- PATCH /api/v2/user/{id}** update user (green button)
- GET /api/v2/user/{id}/rankings** find by id user with rankings (blue button)
- GET /api/v2/user/test-error** (blue button)
- GET /api/v2/user/find-by-document/** Get users by type document and document (blue button)

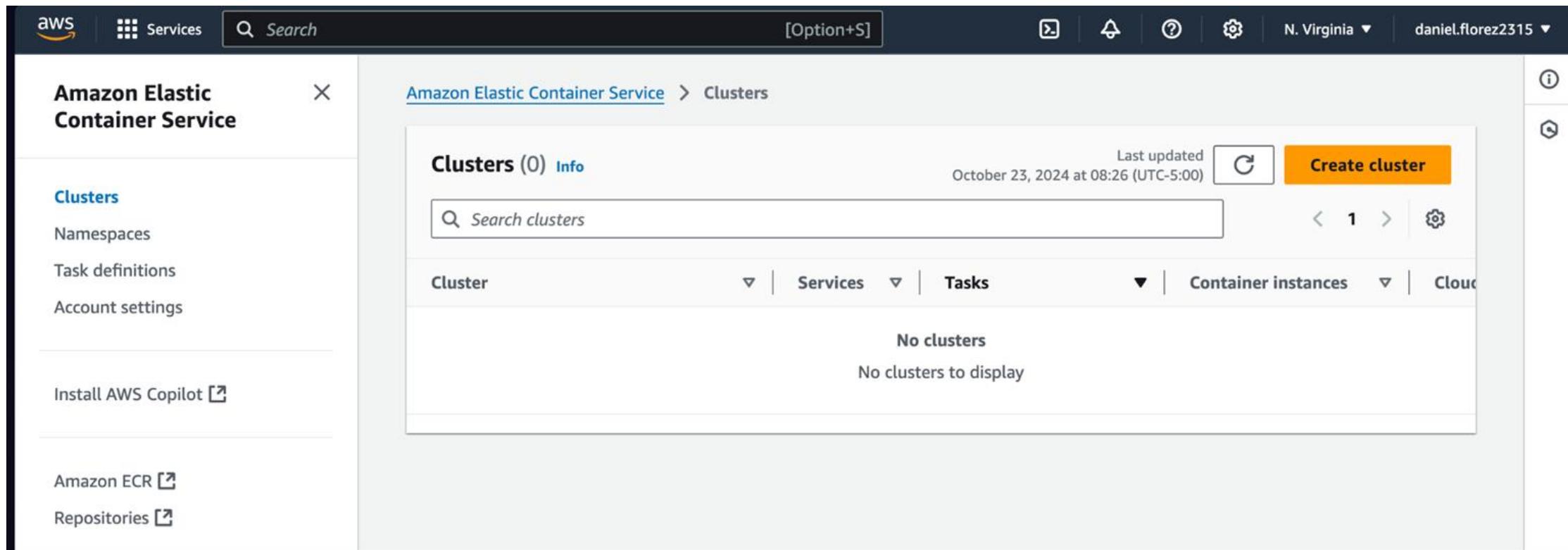
Each endpoint row has a collapse/expand arrow icon on the right side.

1.8. Creando clúster ECS y una definición de tarea

- AWS Elastic Container Service (ECS) es un servicio administrado de Amazon que permite ejecutar y escalar aplicaciones en contenedores de manera sencilla.
- Las aplicaciones se ejecutan en contenedores utilizando Docker, sin preocuparse por la administración de la infraestructura subyacente.

Creando clúster ECS

Para la creación del ECS tenemos que ir al dashboard de AWS e ingresar al servicio de ECS.



The screenshot shows the AWS Elastic Container Service (ECS) Clusters page. At the top, there's a navigation bar with the AWS logo, a search bar, and account information for "daniel.florez2315". Below the navigation bar is a sidebar for "Amazon Elastic Container Service" with options like "Clusters", "Namespaces", "Task definitions", "Account settings", "Install AWS Copilot", "Amazon ECR", and "Repositories". The main content area displays the "Clusters (0)" section. It includes a search bar labeled "Search clusters", a timestamp "Last updated October 23, 2024 at 08:26 (UTC-5:00)", and a prominent orange "Create cluster" button. Below these, there are filters for "Cluster", "Services", "Tasks", "Container instances", and "CloudWatch Metrics". A message "No clusters" and "No clusters to display" is centered on the page.

Creando clúster ECS

Accedemos a la opción de “Create cluster” e ingresamos la información:

- Clúster name: user-clúster
- Crear

The screenshot shows the 'Cluster configuration' step of the 'Create cluster' wizard. It includes fields for 'Cluster name' (set to 'user-cluster'), 'Default namespace - optional' (set to 'user-cluster'), and sections for 'Infrastructure Info' (Serverless), 'Monitoring - optional', and 'Container Insights'.

Cluster configuration

Cluster name
user-cluster
Cluster name must be 1 to 255 characters. Valid characters are a-z, A-Z, 0-9, hyphens (-), and underscores (_).

Default namespace - optional
Select the namespace to specify a group of services that make up your application. You can overwrite this value at the service level.
user-cluster

▼ Infrastructure Info Serverless
Your cluster is automatically configured for AWS Fargate (serverless) with two capacity providers. Add Amazon EC2 instances.

AWS Fargate (serverless)
Pay as you go. Use if you have tiny, batch, or burst workloads or for zero maintenance overhead. The cluster has Fargate and Fargate Spot capacity providers by default.

Amazon EC2 instances
Manual configurations. Use for large workloads with consistent resource demands.

ⓘ External instances using **ECS Anywhere** can be registered after cluster creation is complete.

▼ Monitoring - optional Info
Container Insights is turned off by default. To change the default behavior, use the CloudWatch Container Insights account setting. When you use Container Insights, there is a cost associated with it.

Use Container Insights
CloudWatch automatically collects metrics for many resources, such as CPU, memory, disk, and network. Container Insights also provides diagnostic information, such as container restart failures, that you use to isolate issues and resolve them quickly. You can also set CloudWatch alarms on metrics that Container Insights collects.

Creando clúster ECS

- Cuando se crea un clúster en ECS, automáticamente se genera una tarea en CloudFormation.

The screenshot shows the AWS Elastic Container Service (ECS) Clusters page. At the top, a green success banner displays the message: "Cluster user-cluster has been created successfully." Below the banner, the navigation bar shows "Amazon Elastic Container Service > Clusters". The main area is titled "Clusters (1) Info" and contains a search bar labeled "Search clusters". A table lists one cluster: "user-cluster". The table columns include "Cluster", "Services", "Tasks", "Container instances", "CloudWatch monitoring", and "Capacity". The "user-cluster" row shows 0 services, 0 tasks running, 0 EC2 container instances, and "Default" CloudWatch monitoring. The last updated timestamp is "October 23, 2024 at 08:46 (UTC-5:00)". On the right side of the table, there are buttons for "View cluster", "Edit", "Delete", and "Create cluster".

Creando tarea

Una vez creado el clúster, procederemos a crear una tarea.

The screenshot shows the AWS Management Console interface for the Amazon Elastic Container Service (ECS). On the left, there is a navigation sidebar with the following items:

- Amazon Elastic Container Service** (selected)
- Clusters
- Namespaces
- Task definitions** (selected)
- Account settings
- Install AWS Copilot
- Amazon ECR
- Repositories

The main content area is titled "Amazon Elastic Container Service > Task definitions". It displays the following information:

- Task definitions (0)** Info
- Last updated: October 23, 2024 at 08:48 (UTC-5:00)
- Buttons: Deploy, Create new revision, and **Create new task definition** (highlighted in orange).
- Filter by status: Active
- Table headers: Task definition and Status of last revision.
- Message: **No task definitions** and **No task definitions to display.**
- Button: **Create new task definition**.

Creando tarea

- Task definition family (nombre de la tarea): task-user.
- Infrastructure requirements: AWS Fargate - para que lo administre AWS y sea económico.
- Crearemos un ROL(siguiente diapositiva).

The screenshot shows the 'Infrastructure requirements' section of the AWS Lambda Task Definition configuration. It includes fields for Launch type (set to AWS Fargate), Operating system/Architecture (Linux/X86_64), Network mode (awsvpc), Task size (1 vCPU, 3 GB Memory), Task roles (conditional), and Task execution role (Create new role).

▼ Infrastructure requirements
Specify the infrastructure requirements for the task definition.

Launch type | [Info](#)
Selection of the launch type will change task definition parameters.
 AWS Fargate
Serverless compute for containers.

Amazon EC2 instances
Self-managed infrastructure using Amazon EC2 instances.

OS, Architecture, Network mode
Network mode is used for tasks and is dependent on the compute type selected.

Operating system/Architecture | [Info](#) **Network mode** | [Info](#)

Linux/X86_64 awsvpc

Task size | [Info](#)
Specify the amount of CPU and memory to reserve for your task.

CPU: 1 vCPU **Memory**: 3 GB

▼ Task roles - conditional

Task role | [Info](#)
A task IAM role allows containers in the task to make API requests to AWS services. You can create a task IAM role from the [IAM console](#).

-

Task execution role | [Info](#)
A task execution IAM role is used by the container agent to make AWS API requests on your behalf. If you don't already have a task execution IAM role created, we can create one for you.

Create new role

Creando role

Buscamos el Role en AWSServiceRoleForECS y creamos el role.

The screenshot shows the AWS IAM Roles page. At the top, there is a search bar with the text 'ecs' and a result count of '1 match'. Below the search bar, a table lists roles. The first role listed is 'AWSServiceRoleForECS', which is associated with the 'AWS Service: ecs (Service-Linked Role)'.

Role name	Trusted entities	Last activity
AWSServiceRoleForECS	AWS Service: ecs (Service-Linked Role)	-

Below the table, there is a section titled 'Roles Anywhere' with three options:

- Access AWS from your non AWS workloads**: Operate your non AWS workloads using the same authentication and authorization strategy that you use within AWS.
- X.509 Standard**: Use your own existing PKI infrastructure or use [AWS Certificate Manager Private Certificate Authority](#) to authenticate identities.
- Temporary credentials**: Use temporary credentials with ease and benefit from the enhanced security they provide.

Creando role

- Seleccionamos “AWS Service”
- Use case: Elastic Container ServiceTask

Trusted entity type

AWS service
Allow AWS services like EC2, Lambda, or others to perform actions in this account.

AWS account
Allow entities in other AWS accounts belonging to you or a 3rd party to perform actions in this account.

Web identity
Allows users federated by the specified external web identity provider to assume this role to perform actions in this account.

Use case
Allow an AWS service like EC2, Lambda, or others to perform actions in this account.

Service or use case
Elastic Container Service ▾

Choose a use case for the specified service.

Use case

Elastic Container Service
Allows ECS to create and manage AWS resources on your behalf.

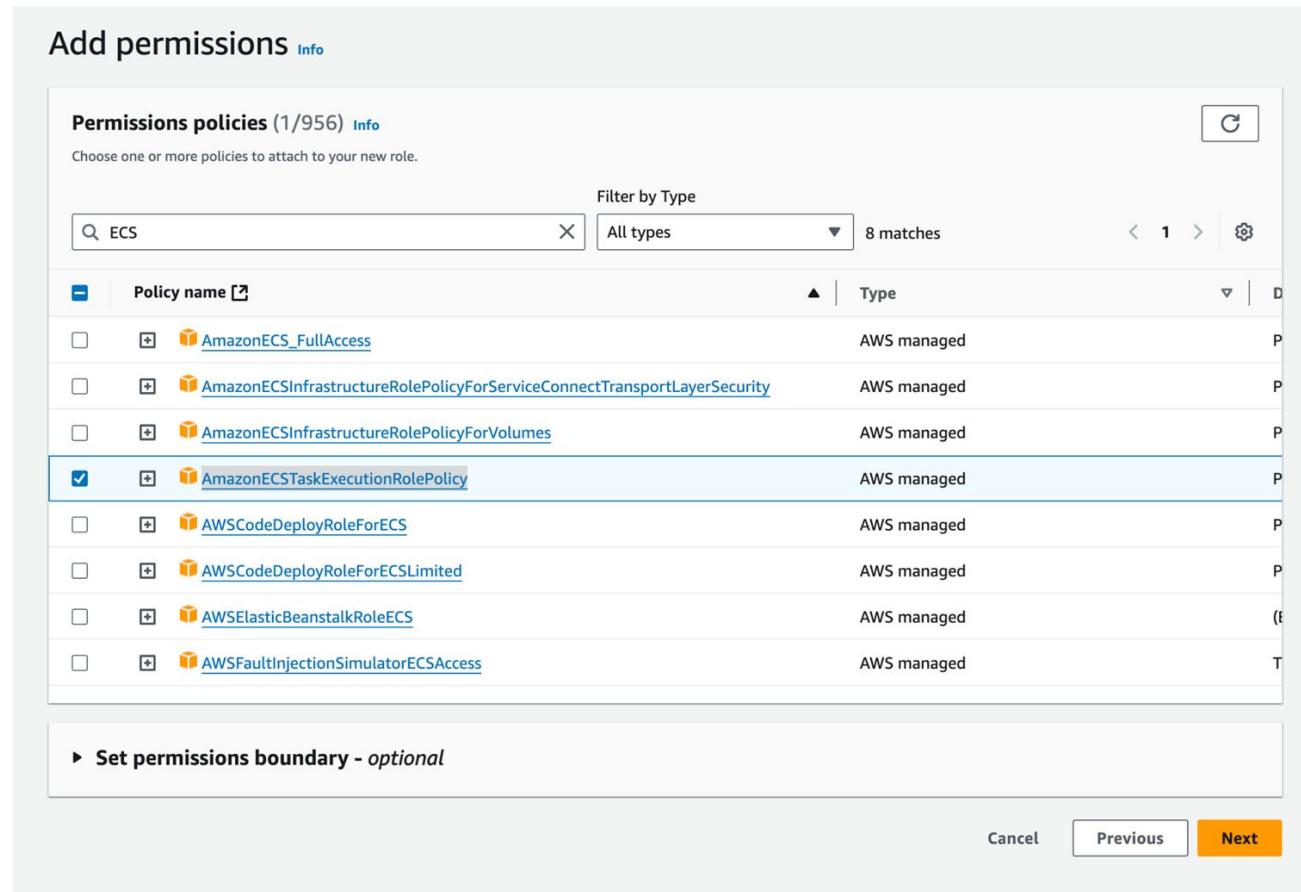
Elastic Container Service Autoscale
Allows Auto Scaling to access and update ECS services.

Elastic Container Service Task
Allows ECS tasks to call AWS services on your behalf.

EC2 Role for Elastic Container Service
Allows EC2 instances in an ECS cluster to access ECS.

Creando role

- En la política de permisos buscamos “AmazonECSTaskExecutionRolePolicy”.



Creando role

- Role name: RoleUserECSTaskExecutionRolePolicy
- Crear Role

The screenshot shows the AWS IAM Role creation process. It consists of two main sections: Step 1: Select trusted entities and Step 2: Add permissions.

Step 1: Select trusted entities

Role name: RoleUserECSTaskExecutionRolePolicy

Description: Allows ECS tasks to call AWS services on your behalf.

Trust policy:

```
1 {  
2   "Version": "2012-10-17",  
3   "Statement": [  
4     {  
5       "Sid": "",  
6       "Effect": "Allow",  
7       "Principal": {  
8         "Service": [  
9           "ecs-tasks.amazonaws.com"  
10        ]  
11      },  
12      "Action": "sts:AssumeRole"  
13    }  
14  ]  
15 }
```

Step 2: Add permissions

Permissions policy summary:

Policy name	Type	Attached as

Creando role

- Se crea el role correctamente.

The screenshot shows the AWS Identity and Access Management (IAM) service interface. On the left, there's a sidebar with 'Identity and Access Management (IAM)' at the top, followed by a search bar labeled 'Search IAM'. Below that is a 'Dashboard' section, then 'Access management' which includes 'User groups', 'Users', 'Roles' (which is currently selected and highlighted in blue), 'Policies', 'Identity providers', and 'Account settings'. At the bottom of the sidebar is another 'Access reports' section. The main content area has a green header bar with a checkmark icon and the text 'Role RoleUserECSTaskExecutionRolePolicy created.' To the right of the header are 'View role' and 'X' buttons. Below the header, the breadcrumb navigation shows 'IAM > Roles'. The main table has a header row with 'Role name' and 'Trusted entities'. There are four rows of data: 1. 'AWSServiceRoleForECS' with 'AWS Service: ecs (Service-Linked Role)'. 2. 'AWSServiceRoleForSupport' with 'AWS Service: support (Service-Linked Role)'. 3. 'AWSServiceRoleForTrustedAdvisor' with 'AWS Service: trustedadvisor (Service-Linked Role)'. 4. 'RoleUserECSTaskExecutionRolePolicy' with 'AWS Service: ecs-tasks'. Each row has a checkbox icon to its left and a 'Delete' button to its right. A large orange 'Create role' button is located on the right side of the table header. A search bar and pagination controls are also present at the bottom of the table.

Role name	Trusted entities
AWSServiceRoleForECS	AWS Service: ecs (Service-Linked Role)
AWSServiceRoleForSupport	AWS Service: support (Service-Linked Role)
AWSServiceRoleForTrustedAdvisor	AWS Service: trustedadvisor (Service-Linked Role)
RoleUserECSTaskExecutionRolePolicy	AWS Service: ecs-tasks

Creando contenedor

- Ingresamos la información:
 - Name: ms_user
 - Image URL: daniel0223/netec-usuario-rep:latest (repositorio de Docker Hub).
 - Port: 8005 (se asigna tanto interno como externo).

Container - 1 [Info](#) Essential container [Remove](#)

Container details
Specify a name, container image, and whether the container should be marked as essential. Each task definition must have at least one essential container.

Name	ms_user	Image URI	daniel0223/netec-usuario-rep:latest	Essential container	Yes
------	---------	-----------	-------------------------------------	---------------------	-----

Up to 255 letters (uppercase and lowercase), numbers, hyphens, underscores, colons, periods, forward slashes, and number signs are allowed.

Private registry [Info](#)
Store credentials in Secrets Manager, and then use the credentials to reference images in private registries.

Private registry authentication

Port mappings [Info](#)
Add port mappings to allow the container to access ports on the host to send or receive traffic. For port name, a default will be assigned if left blank.

Container port	Protocol	Port name	App protocol
8005	TCP	container-port-protocol	HTTP

[Add port mapping](#)

Read only root file system [Info](#)
When this parameter is turned on, the container is given read-only access to its root file system.

Read only

Resource allocation limits - conditional [Info](#)
Container-level CPU, GPU, and memory limits are different from task-level values. They define how much resources are allocated for the container. If container attempts to exceed the memory specified in hard limit, the container is terminated.

CPU	GPU	Memory hard limit	Memory soft limit
1 in vCPU	1	3 in GB	1 in GB

Creando variables de entorno

- Agregamos las variables de entorno, manual o por medio de un archivo.
 - Name: ms_user
 - Image URL: daniel0223/netec-usuario-rep:latest (repositorio de Docker Hub)
 - Port: 8005 (se asigna tanto interno como externo).

▼ Environment variables - optional

Environment variables | [Info](#)

Add individually
Add a key-value pair to specify an environment variable.

Key	Value type	Value	Remove
PORT	Value	PORT	Remove
DB_HOST	Value	rds-netec.chzxx6ihzzuv.us	Remove
DB_USERNAME	Value	admin	Remove
DB_PASSWORD	Value	test123456	Remove

[Add environment variable](#)

Add from file
Add environment variables in bulk by providing an environment file hosted on Amazon S3.

[Add environment file](#)

You can add 10 more environment files.

Creando variables de entorno

- Dejamos por defecto la configuración de “Use log collection” y creamos la task.

Log collection | [Info](#)

Configure your task to send container logs to a logging destination using a default configuration. See pricing information on [Amazon CloudWatch](#).

Use log collection

Key	Value type	Value	
awslogs-group	Value	/ecs/	Remove
awslogs-region	Value	us-east-1	Remove
awslogs-stream-prefix	Value	ecs	Remove
awslogs-create-group	Value	true	Remove
mode	Value	non-blocking	Remove
max-buffer-size	Value	25m	Remove

[Add log configuration option](#)

Creación task

- Dejamos por defecto la configuración de “Use log collection” y creamos la task.

The screenshot shows the AWS Elastic Container Service (ECS) Task Definitions console. A green header bar at the top displays a success message: "Task definition successfully created" and "task-user:1 has been successfully created. You can use this task definition to deploy a service or run a task." To the right of the message are "View task definition" and "X" buttons. Below the header, the breadcrumb navigation shows: Amazon Elastic Container Service > Task definitions > task-user > Revision 1 > Containers. The main title "task-user:1" is centered above a row of buttons: "Deploy ▾", "Actions ▾", and "Create new revision ▾". The "Create new revision" button is highlighted in orange. The main content area is titled "Overview" with an "Info" link. It displays the following details in four columns:

ARN	Status	Time created	App environment
arn:aws:ecs:us-east-1:43846516:4320:task-definition/task-user:1	ACTIVE	October 23, 2024 at 09:30 (UTC-5:00)	Fargate
Task role	Task execution role	Operating system/Architecture	Network mode
RoleUserECSTaskExecutionRolePolicy	ecsTaskExecutionRole	Linux/X86_64	awsvpc

1.9. Agregando servicio RDS mysql

AWS RDS (Amazon Relational Database Service) es un servicio en la nube de Amazon Web Services que facilita la configuración, operación y escalabilidad de bases de datos relacionales.

Creación RDS

Para la creación de una base de datos en RDS:

1. Nos dirigimos al dashboard de AWS.
2. Buscamos el servicio de RDS.
3. Damos clic al botón créate.

The screenshot shows the AWS Amazon RDS service dashboard. At the top, there's a navigation bar with the AWS logo, a search bar, and various icons. Below the navigation bar, the main interface has two main sections: 'Amazon RDS' on the left and 'Recursos' on the right.

Amazon RDS Panel:

- Bases de datos
- Editor de consultas
- Información sobre rendimiento
- Instantáneas de
- Exportaciones en Amazon S3
- Copias de seguridad automatizadas
- Instancias reservadas
- Proxies
- Grupos de subredes
- Grupos de parámetros
- Grupos de opciones
- Versiones de motor personalizadas
- Integraciones sin extracción, transformación y carga (ETL) [Nuevo](#)
- Eventos
- Suscripciones a eventos

Recursos:

Está usando los siguientes recursos de Amazon RDS en la región US East (N. Virginia) (usados/cuota),

Instancias de base de datos (0/40)	Grupos de parámetros (0)
Almacenamiento asignado (0 TB/100 TB)	Predeterminado (0)
Las instancias y el almacenamiento incluyen	Personalizada (0/100)
Neptune y DocumentDB.	Aumentar el límite de instancias de base de datos
Clústeres de base de datos (0/40)	Grupos de opciones (0)
Instancias reservadas (0/40)	Predeterminado (0)
Instantáneas de (0)	Personalizada (0/20)
Manual	Grupos de subredes (0/50)
Clúster de base de datos (0/100)	Plataformas compatibles VPC
Instancia de base de datos (0/100)	Red predeterminada vpc-0ba26ae453620caa
Automatizado	
Clúster de base de datos (0)	
Instancia de base de datos (0)	
Eventos recientes (0)	
Suscripciones a eventos (0/20)	

Crear base de datos:

Amazon Relational Database Service (RDS) facilita la configuración, el funcionamiento y el escalado de una base de datos relacional en la nube.

[Crear base de datos](#)

Puede utilizar una copia de seguridad de Amazon S3 para restaurar y crear una nueva base de datos Aurora MySQL y MySQL.

[Restaurar desde S3](#)

Nota: Las instancias de base de datos se iniciarán en la región US East (N. Virginia).

Creación RDS

Para la primera parte de la configuración:

- Seleccionaremos la base de datos MySQL.
- Creación estándar.

Creación estándar
Puede definir todas las opciones de configuración, incluidas las de disponibilidad, seguridad, copias de seguridad y mantenimiento.

Creación sencilla
Utilice las configuraciones recomendadas. Algunas opciones de configuración se pueden cambiar después de crear la base de datos.

Opciones del motor

Tipo de motor [Información](#)

<input type="radio"/> Aurora (MySQL Compatible) 	<input type="radio"/> Aurora (PostgreSQL Compatible) 
<input checked="" type="radio"/> MySQL 	<input type="radio"/> MariaDB 
<input type="radio"/> PostgreSQL 	<input type="radio"/> Oracle 

Creación RDS

Como plantilla, seleccionamos capa gratuita para tener más economía.

Plantillas

Elija una plantilla de ejemplo para adaptarla a su caso de uso.

- Producción**
Utilice los valores predeterminados para disfrutar de una alta disponibilidad y de un rendimiento rápido y constante.
- Desarrollo y pruebas**
Esta instancia se ha diseñado para su uso en desarrollo, fuera de un entorno de producción.
- Capa gratuita**
Utilice el nivel gratuito de RDS para desarrollar nuevas aplicaciones, probar aplicaciones existentes o adquirir experiencia práctica con Amazon RDS.
[Información](#)

Creación RDS-Configuración

En las configuraciones ingresamos esta información:

- Identificador de instancias de bases de datos: rds-netec
- Contraseña maestra: test123456

Configuración

Identificador de instancias de bases de datos [Información](#)
Escriba un nombre para la instancia de base de datos. El nombre debe ser único en relación con todas las instancias de base de datos pertenecientes a su cuenta de AWS en la región de AWS actual.

El identificador de la instancia de base de datos no distingue entre mayúsculas y minúsculas, pero se almacena con todas las letras en minúsculas (como en "miinstanciaedb"). Restricciones: de 1 a 60 caracteres alfanuméricos o guiones. El primer carácter debe ser una letra. No puede contener dos guiones consecutivos. No puede terminar con un guion.

▼ Configuración de credenciales

Nombre de usuario maestro [Información](#)
Escriba un ID de inicio de sesión para el usuario maestro de la instancia de base de datos.

1 a 16 caracteres alfanuméricos. El primer carácter debe ser una letra.

Administración de credenciales
Puede usar AWS Secrets Manager o administrar sus credenciales de usuario maestro.

Administrado en AWS Secrets Manager - más seguro
RDS genera una contraseña y la administra durante todo su ciclo de vida mediante AWS Secrets Manager.

Autoadministrado
Cree su propia contraseña o pida a RDS que cree una contraseña para que pueda administrarla.

Generar contraseña automáticamente
Amazon RDS puede generar una contraseña en su nombre, o bien puede especificar su propia contraseña.

Contraseña maestra [Información](#)

Password strength Weak

Restricciones mínimas: al menos 8 caracteres ASCII imprimibles. No puede contener ninguno de los siguientes símbolos: / " @

Confirmar la contraseña maestra [Información](#)

Creación RDS-Conectividad

- Seleccionamos la opción “Default VPC.”
- Acceso público: si
- Crear base de datos.

Conectividad [Información](#) [C](#)

Recurso de computación
Seleccione si desea configurar una conexión a un recurso de computación para esta base de datos. Al establecer una conexión, se cambiará automáticamente la configuración de conectividad para que el recurso de computación se pueda conectar a esta base de datos.

No se conecte a un recurso informático EC2
No configure una conexión a un recurso informático para esta base de datos. Puede configurar manualmente una conexión a un recurso informático más adelante.

Conectarse a un recurso informático de EC2
Configure una conexión a un recurso informático EC2 para esta base de datos.

Nube privada virtual (VPC) [Información](#)
Elija la VPC. La VPC define el entorno de red virtual para esta instancia de DB.

Default VPC (vpc-0ba26aee453620caa) [▼](#)
6 Subredes, 6 Zonas de disponibilidad

Solo se muestran las VPC con grupos de subredes de base de datos correspondientes.

Después de crear una base de datos, no puede cambiar su VPC.

Grupo de subredes de la base de datos [Información](#)
Elija el grupo de subred de DB. El grupo de subred de DB define las subredes e intervalos de IP que puede usar la instancia de DB en la VPC seleccionada.

predeterminado [▼](#)

Acceso público [Información](#)

Sí
RDS asigna una dirección IP pública a la base de datos. Las instancias de Amazon EC2 y otros recursos fuera de la VPC pueden conectarse a la base de datos. Los recursos de la VPC también pueden conectarse a la base de datos. Elija uno o varios grupos de seguridad de VPC que especifiquen qué recursos pueden conectarse a la base de datos.

No
RDS no asigna una dirección IP pública a la base de datos. Solo las instancias de Amazon EC2 y otros recursos dentro de la VPC pueden conectarse a la base de datos. Elija uno o varios grupos de seguridad de VPC que especifiquen qué recursos pueden conectarse a la base de datos.

Grupo de seguridad de VPC (firewall) [Información](#)
Elija uno o varios grupos de seguridad de VPC para permitir el acceso a su base de datos. Asegúrese de que las reglas del grupo de

Creación MySQL RDS

- Con esto se creará la base de datos con nuestras configuraciones.

The screenshot shows the AWS RDS MySQL creation wizard. The first step, "Creación de base de datos rds-netec", includes a note about potential delays and a link to credential details. The second step, "Presentación de los puntos de conexión globales", provides information on global endpoints and a link for more info. The third step, "Bases de datos (1)", shows a single database named "rds-netec" in a "Creando" state, with options for group resources, modification, services, restoration from S3, and creating a new database.

Creación de base de datos rds-netec

Es posible que el lanzamiento de la base de datos tarde unos minutos. Puede utilizar la configuración de rds-netec para simplificar la configuración de complementos de base de datos sugeridos mientras terminamos de crear su base de datos.

Ver detalles de credenciales

Presentación de los puntos de conexión globales

Cada clúster global tiene un punto de conexión global que la aplicación puede usar para enviar solicitudes a la instancia de escritura del clúster principal. Tras una operación de conmutación por error o conmutación, este punto de conexión envía automáticamente las solicitudes a la instancia de escritura del nuevo clúster principal para que no tenga que cambiar el código de la aplicación. [Más información](#)

RDS > Bases de datos

Considerar la posibilidad de crear una implementación azul-verde para minimizar el tiempo de inactividad durante las actualizaciones.

Es posible que desee considerar el uso de las implementaciones azul-verde de Amazon RDS y minimizar el tiempo de inactividad durante las actualizaciones. Una implementación azul-verde proporciona un entorno de ensayo para los cambios en las bases de datos de producción. [Guía del usuario de RDS](#) [Guía del usuario de Aurora](#)

Bases de datos (1)

Recursos del grupo Modificar Acciones Servicios Restaurar desde S3 Crear base de datos

Filtrar por bases de datos

Identificador de base de datos	Estado	Rol	Motor	Región ...	Tamaño	Recomendaciones
rds-netec	Creando	Instancia	MySQL Co...	us-east-1b	db.t4g.mi...	

1.10. Agregando servicio para exponer los contenedores

Una vez configurado el clúster y las tareas, es necesario exponer los puertos para que la aplicación sea accesible desde internet. Esto con el fin de que la aplicación sea funcional y no quede en nuestra red privada de AWS. A lo largo del curso vamos a usar un balanceador de carga para un mejor escalamiento de nuestros contenedores.

Crear servicio

- Ingresamos a la tarea de “task-user”.
- En el botón de Deploy seleccionamos Create service.

The screenshot shows the AWS Elastic Container Service (ECS) Task Definitions page. The URL in the browser is [Amazon Elastic Container Service > Task definitions > task-user](#). The main heading is "task-user (1/1) [Info](#)". To the right, it says "Last updated October 23, 2024 at 14:51 (UTC-5:00)". Below the heading is a search bar labeled "Filter task definition revisions by value". Underneath the search bar are two checkboxes: "Task definition: revision" (which is checked) and "task-user:1" (which is also checked). To the right of the checkboxes is a "Status" column showing "ACTIVE" with a green checkmark. On the far right of the row are "Deploy ▲", "Actions ▼", and a yellow "Create new revision ▼" button. A dropdown menu is open over the "Actions" button, showing options: "Create service", "Update service", and "Run task". Below the dropdown are navigation arrows and a settings gear icon.

Crear servicio

En Environment vamos a seleccionar la versión LATEST y como máquina de lanzamiento FARGATE.

The screenshot shows the 'Environment' configuration page for AWS Fargate. At the top, it says 'Existing cluster' with a dropdown menu set to 'user-cluster'. To the right are two buttons: 'Create a new cluster' and 'AWS Fargate'. Below this is a section titled 'Compute configuration (advanced)'. It contains two options: 'Capacity provider strategy' (disabled) and 'Launch type'. 'Launch type' is selected and highlighted with a blue border. A tooltip for 'Launch type' states: 'Launch tasks directly without the use of a capacity provider strategy.' Under 'Launch type', there's a dropdown menu set to 'FARGATE'. At the bottom, there's a 'Platform version' dropdown menu set to 'LATEST'.

Crear servicio

En Deployment configuration, le damos el nombre al servicio y 1 réplica.

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 application type is not applicable when launching a service.

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	Revision
task-user	1

Service name
Assign a service name that is unique for this cluster.

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.

Crear servicio

En el apartado de Networking vamos a crear un nuevo grupo de seguridad.

Security group | [Info](#)
Choose an existing security group or create a new security group.
 Use an existing security group
 Create a new security group

Security group details
Specify the configuration to use when creating the new security group.

Security group name	Security group description
sg-ecs-netec	Created in ECS Console

Security group name must be 1 to 255 characters. Valid characters are a-z, A-Z, 0-9, underscores (_), hyphens (-), colons (:), forward slashes (/), parentheses (()), hashtags (#), commas (,), at signs (@), brackets ([]), plus signs (+), equal signs (=), ampersands (&), semicolons (;), brackets ([]), exclamation points (!), dollar signs (\$), asterisks (*).

Security group description must be 1 to 255 characters. Valid characters are a-z, A-Z, 0-9, underscores (_), hyphens (-), colons (:), forward slashes (/), parentheses (()), hashtags (#), commas (,), at signs (@), brackets ([]), plus signs (+), equal signs (=), ampersands (&), semicolons (;), brackets ([]), exclamation points (!), dollar signs (\$), asterisks (*).

Inbound rules for security groups
Add one or more ingress rules for your security group.

Type	Protocol	Port range	Source	Values
Custom ... ▾	TCP	8005	Anywhere ▾	0.0.0.0/0, ::/0

Enter a valid port or port range between 0 and 65535. For example: 80 or 0-1023.

[Add rule](#)

Public IP | [Info](#)
Choose whether to auto-assign a **public** IP to the task's elastic network interface (ENI).
 Turned on

Crear servicio

Para este caso no vamos a usar un auto scaling ni un loadBalancing.

Luego crear:

The screenshot shows the 'Create service' configuration page. It includes sections for 'Load balancing - optional' and 'Service auto scaling - optional'. In the 'Load balancing' section, it says 'Configure load balancing using Amazon Elastic Load Balancing to distribute traffic evenly across the healthy tasks in your service.' Below this is a dropdown menu labeled 'Load balancer type' with 'Info' link, containing the option 'None'. In the 'Service auto scaling' section, it says 'Automatically adjust your service's desired count up and down within a specified range in response to CloudWatch alarms. You can modify your service auto scaling configuration at any time to meet the needs of your application.' Below this is a checkbox labeled 'Use service auto scaling' with the sub-instruction 'Configure service auto scaling to adjust your service's desired count'.

▼ **Load balancing - optional**
Configure load balancing using Amazon Elastic Load Balancing to distribute traffic evenly across the healthy tasks in your service.

Load balancer type | [Info](#)
Configure a load balancer to distribute incoming traffic across the tasks running in your service.
None ▾

▼ **Service auto scaling - optional**
Automatically adjust your service's desired count up and down within a specified range in response to CloudWatch alarms. You can modify your service auto scaling configuration at any time to meet the needs of your application.

Use service auto scaling
Configure service auto scaling to adjust your service's desired count

Crear servicio

Iniciar el proceso de creación de la tarea.

The screenshot shows the AWS Elastic Container Service (ECS) Task Definitions page. On the left, there's a sidebar with links for Clusters, Namespaces, Task definitions (which is selected and highlighted in blue), and Account settings. Below the sidebar are links for Install AWS Copilot and Amazon ECR. At the top right, there are buttons for View task definition, View in CloudFormation, Deploy, Actions, and Create new revision. A prominent green success message at the top states: "Task definition successfully created" and "task-user:1 has been successfully created. You can use this task definition to deploy a service or run a task." Below the message, it says "USERS deployment is in progress. It takes a few minutes." The main content area shows the details for the task definition "task-user:1". The "Overview" section includes the ARN (arn:aws:ecs:us-east-1:438465164320:task-definition/task-user:1), Status (ACTIVE), Time created (October 23, 2024 at 09:30 (UTC-5:00)), App environment (Fargate), Task role (RoleUserECSTaskExecutionRolePolicy), Task execution role (ecsTaskExecutionRole), Operating system/Architecture (Linux/X86_64), and Network mode (awsvpc).

ARN	Status	Time created	App environment
arn:aws:ecs:us-east-1:438465164320:task-definition/task-user:1	ACTIVE	October 23, 2024 at 09:30 (UTC-5:00)	Fargate
Task role	Task execution role	Operating system/Architecture	Network mode
RoleUserECSTaskExecutionRolePolicy	ecsTaskExecutionRole	Linux/X86_64	awsvpc

Validando tarea

Para ver el estado de la tarea o el contenedor que estamos ejecutando, volvemos a clúster y podemos ver el estado de la tarea y la información adicional.

The screenshot shows the AWS ECS console interface. On the left, there's a sidebar with navigation links for Clusters, Namespaces, Task definitions, Account settings, and other services like ECR and AWS Batch. The main area is titled 'user-cluster' and displays 'Cluster overview'. It shows the ARN (arn:aws:ecs:us-east-1:4384651643:cluster/user-cluster), Status (Active), CloudWatch monitoring (Default), and Registered container instances (none). Below this, under 'Services', it shows Draining (none), Active (1), Pending (1), and Running (none). Under 'Encryption', it shows Managed storage (none) and Fargate ephemeral storage. At the bottom, there are tabs for Services, Tasks (which is selected), Infrastructure, Metrics, Scheduled tasks, and Tags. The 'Tasks (1)' section shows a table with one task entry: Task ID (6f5c2c...), Last status (Pending), Desired state (Running), Health status (Unknown), Started at (none), Container instance (none), and Launch type (FARGATE).

Task	Last status	Desired state	Health status	Started at	Container instance	Launch type
6f5c2c...	Pending	Running	Unknown	-	-	FARGATE

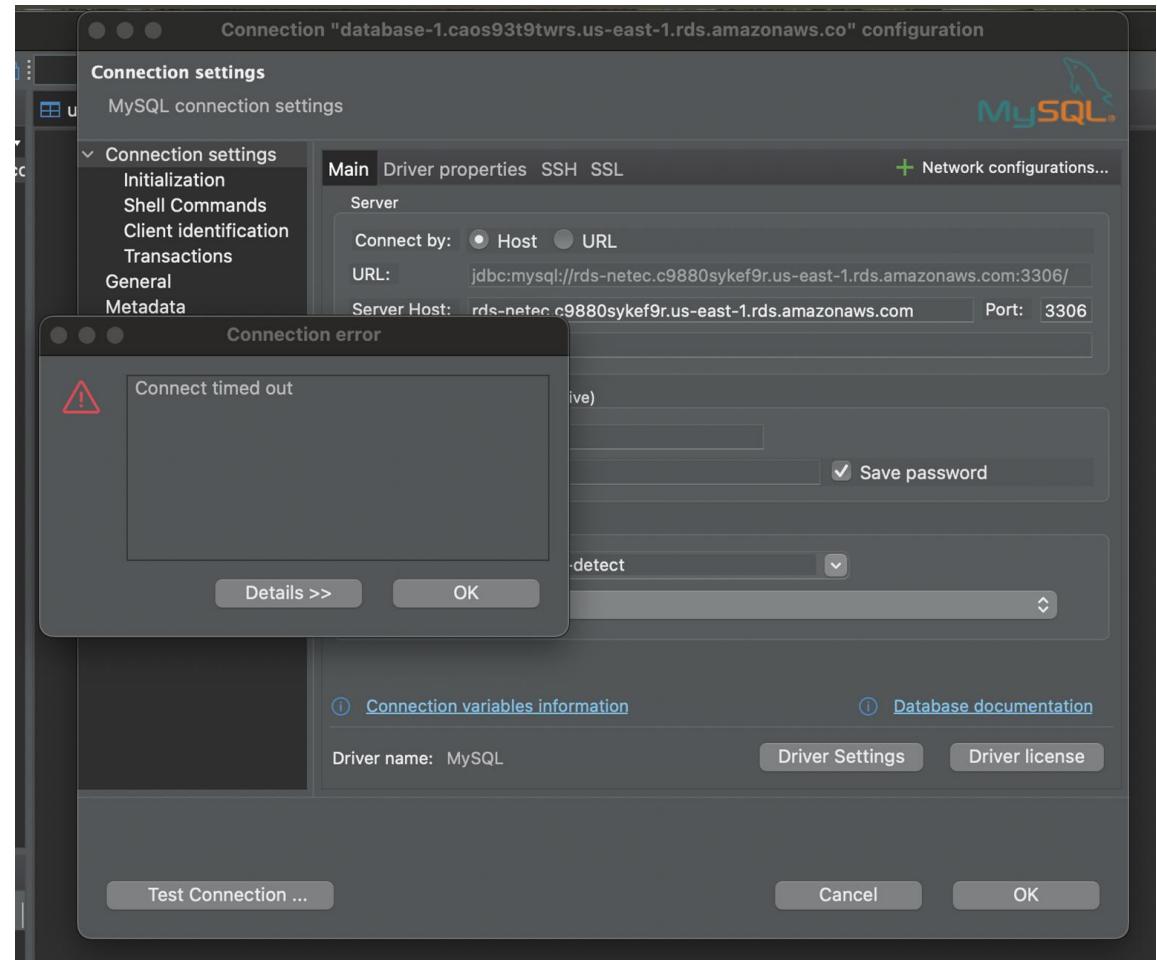
1.11. Solucionando problemas de conexión de la base de datos

No basta con crear la base de datos en RDS para tener acceso, debemos configurar unos servicios adicionales para poder conectarnos a la base de datos.

Punto de enlace y puerto	Redes	Seguridad
Punto de enlace  rds-netec.c9880sykef9r.us-east-1.rds.amazonaws.com	Zona de disponibilidad us-east-1b VPC vpc-0ba26aee453620caa Grupo de subredes default-vpc-0ba26aee453620caa Subredes subnet-0f24df3908ad10369 subnet-0107bff9b7fcfd16b subnet-028f6ff200299a106 subnet-0a14d223a6df13309 subnet-0d04c791d50a06b83 subnet-0d561047cdcc5a66b	Grupos de seguridad de la VPC default (sg-0ccd790618bf6ebb9) <input checked="" type="radio"/> Activo Accesible públicamente Sí Entidad de certificación Información rds-ca-rsa2048-g1 Fecha de la entidad de certificación May 25, 2061, 18:34 (UTC-05:00) Fecha de expiración del certificado de instancia de base de datos October 22, 2025, 15:04 (UTC-05:00)

Error de conexión a RDS

Al intentar conectarnos por medio de un gestor de base datos vamos a tener un timeOut. Esto se debe a que necesitamos configurar los S.G.



Encontrando el Security Group

Para poder configurar correctamente el Security Group, es necesario ir al apartado de “conectividad y seguridad >seguridad > clic”.

The screenshot shows the AWS RDS 'Conectividad y seguridad' (Connectivity and security) configuration page. The 'Seguridad' (Security) section displays the following details:

Punto de enlace y puerto	Redes	Seguridad
Punto de enlace rds-netec.c9880sykef9.us-east-1.rds.amazonaws.com	Zona de disponibilidad us-east-1b VPC vpc-0ba26aee453620caa Grupo de subredes default-vpc-0ba26aee453620caa Subredes subnet-0f24df3908ad10369 subnet-0107bfff9b7fcfd16b subnet-028f6ff200299a106 subnet-0a14d223a6df13309 subnet-0d04c791d50a06b83 subnet-0d561047cdcc5a66b	Grupos de seguridad de la VPC default (sg-0ccd790618bf6ebb9) Activado Accesible públicamente Sí Entidad de certificación Información rds-ca-rsa2048-g1 Fecha de la entidad de certificación May 25, 2061, 18:34 (UTC-05:00) Fecha de expiración del certificado de instancia de base de datos October 22, 2025, 15:04 (UTC-05:00)

Configurando el Security Group

Para configurar el S.G tenemos que editar las reglas de entrada.

The screenshot shows the AWS VPC Security Groups console. At the top, there's a search bar with the filter 'sg-0ccd790618bf6ebb9' applied. Below the search bar is a table with columns: Name, ID de grupo de seguridad, Nombre del grupo de seguridad, ID de la VPC, and Descripción. A single row is selected, showing 'sg-0ccd790618bf6ebb9' as the Name, 'default' as the Nombre del grupo de seguridad, 'vpc-0ba26aee453620caa' as the ID de la VPC, and 'default VPC' as the Descripción. Below the table, the text 'sg-0ccd790618bf6ebb9 - default' is displayed. At the bottom, there are tabs for 'Detalles', 'Reglas de entrada' (which is selected), 'Reglas de salida', and 'Etiquetas'. The 'Reglas de entrada' section shows a table with one rule: 'sgr-0e80be1afe83421ca' (Name), '-' (ID de la regla del gr...), 'Todo el tráfico' (Tipo), 'Todo' (Protocolo), and 'Todo' (Intervalo de puertos). There are also buttons for 'C' (Create), 'Administrar etiquetas' (Manage tags), and 'Editar reglas de entrada' (Edit rules).

Editar reglas de entrada

- En esta sección agregamos una nueva regla.
- En los S.G podemos establecer una IP específica, una serie de IPs o pública.

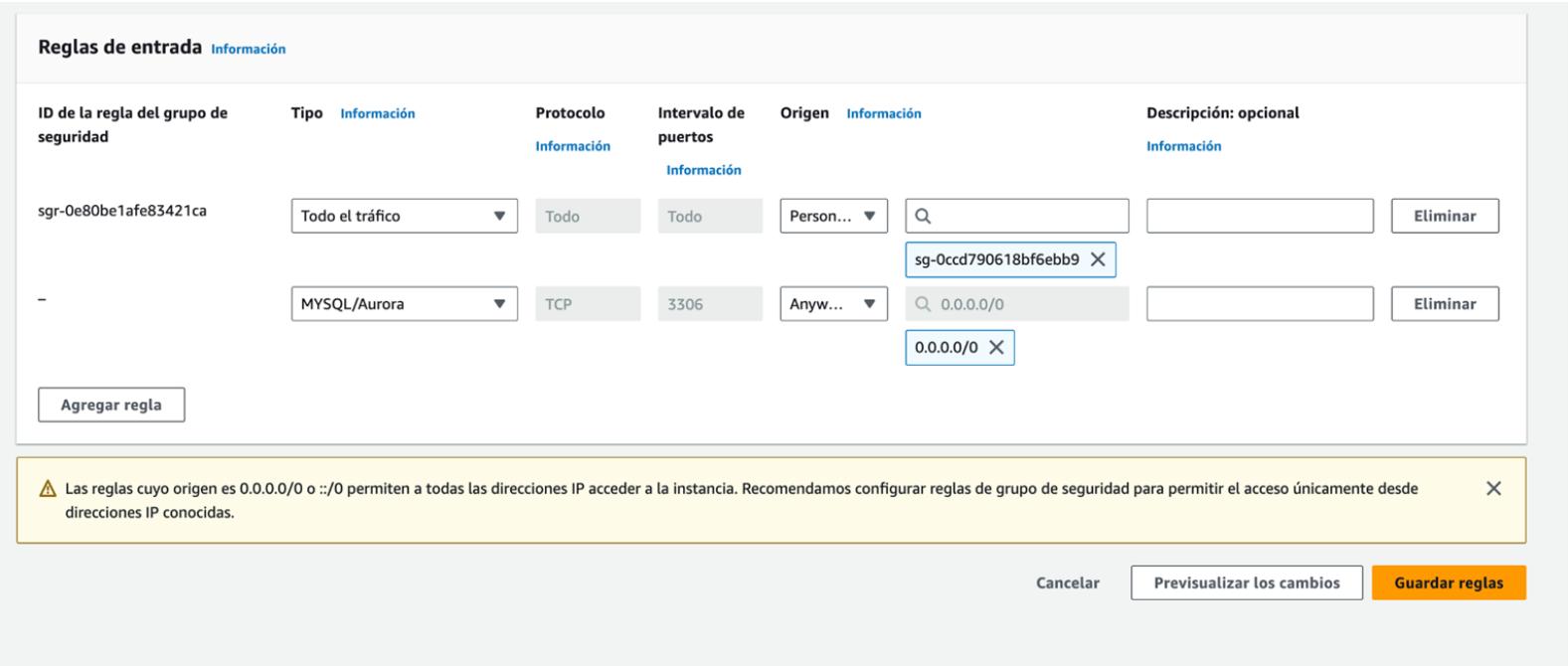
Reglas de entrada [Información](#)

ID de la regla del grupo de seguridad	Tipo	Información	Protocolo	Información	Intervalo de puertos	Origen	Información	Descripción: opcional
sgr-0e80be1afe83421ca	Todo el tráfico	▼	Todo	Todo	Person...	▼	<input type="text"/> sg-0cccd790618bf6ebb9 X	<input type="text"/> Eliminar
-	MYSQL/Aurora	▼	TCP	3306	Anyw...	▼	<input type="text"/> 0.0.0.0/0	<input type="text"/> Eliminar
							<input type="text"/> 0.0.0.0/0 X	

[Agregar regla](#)

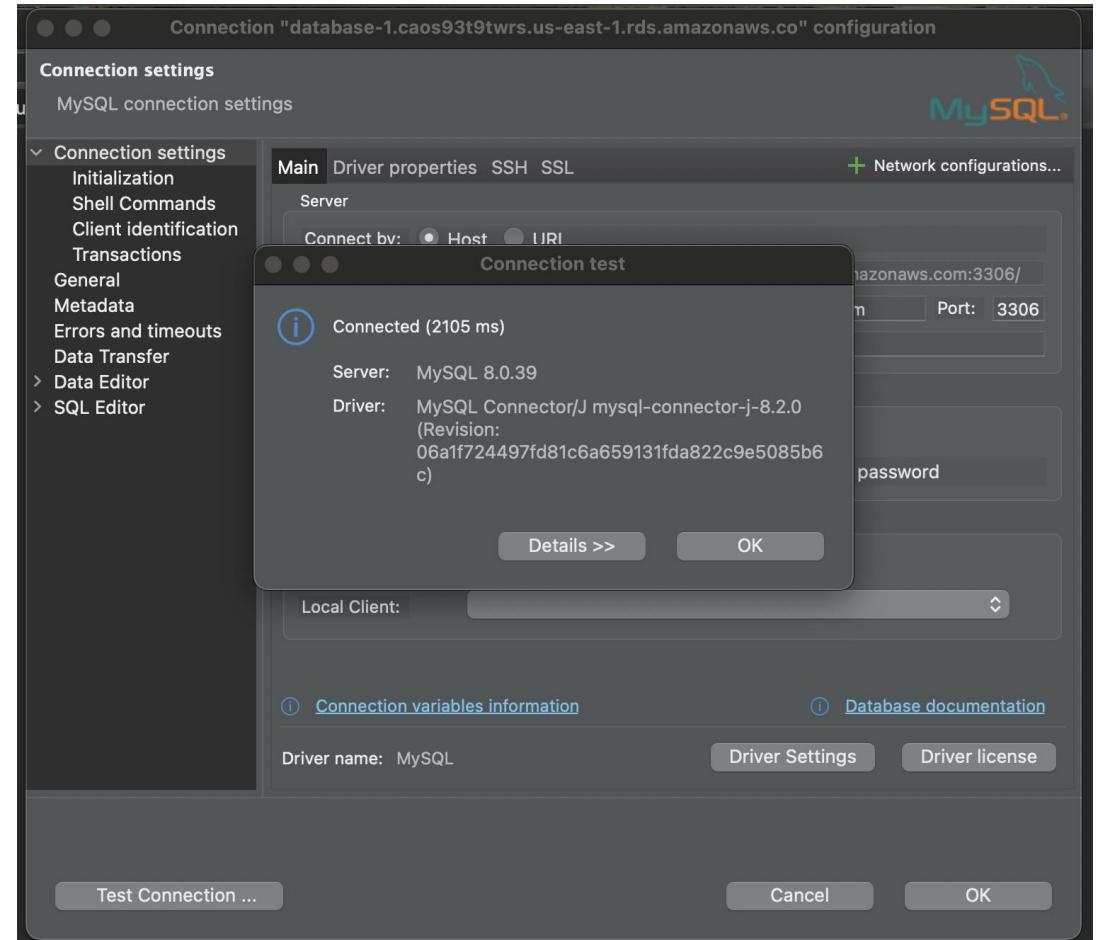
⚠️ Las reglas cuyo origen es 0.0.0.0/0 o ::/0 permiten a todas las direcciones IP acceder a la instancia. Recomendamos configurar reglas de grupo de seguridad para permitir el acceso únicamente desde direcciones IP conocidas. [X](#)

[Cancelar](#) [Previsualizar los cambios](#) [Guardar reglas](#)



Validando conexión

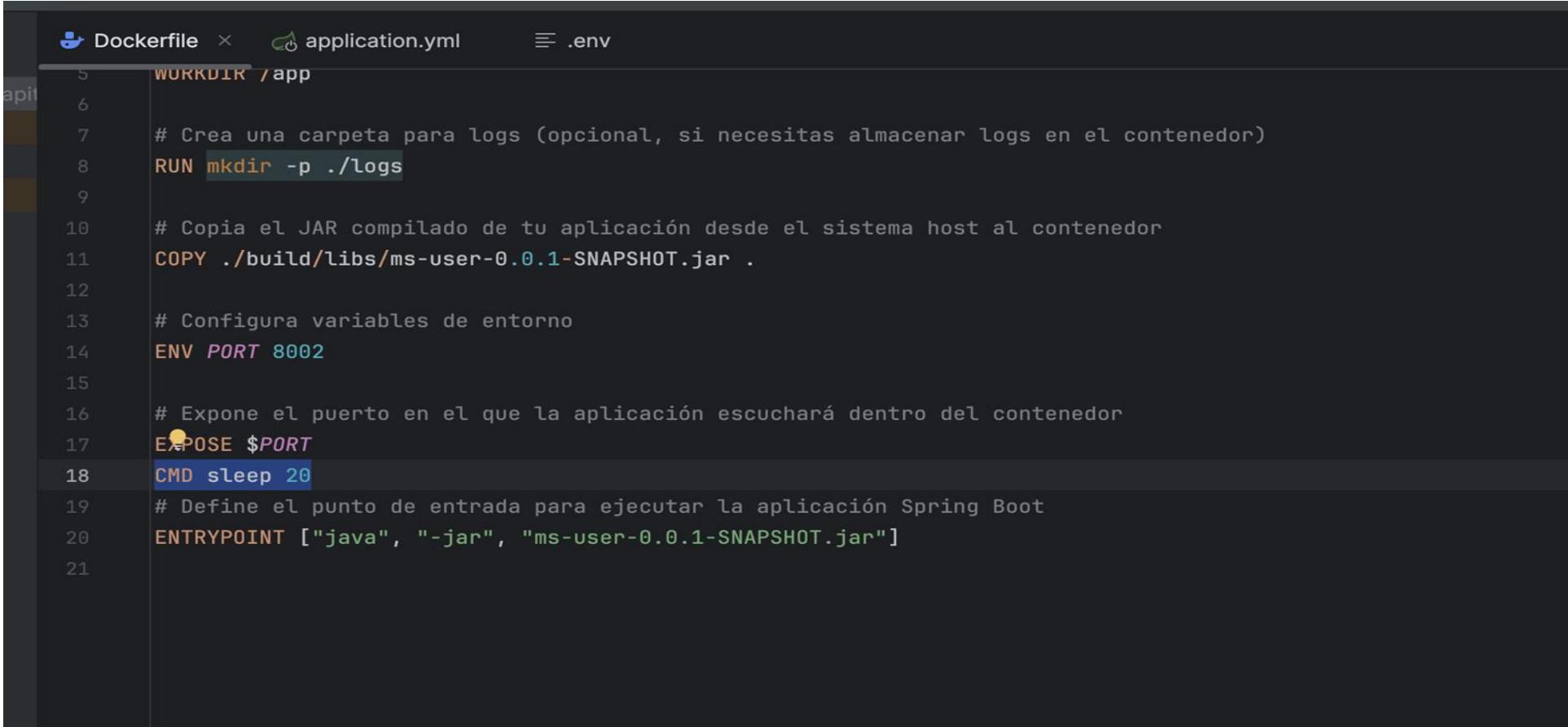
Probamos de nuevo y nos permite conectarnos con éxito.



1.12. Solución II: Problema de conexión configurando datasource

Otra forma de solucionar el error es ajustando el Dockerfile con el tiempo de espera para que los demás servicios se desplieguen correctamente, para esto agregamos: "CMD sleep 20" y recreamos la imagen.

Recreación de imagen



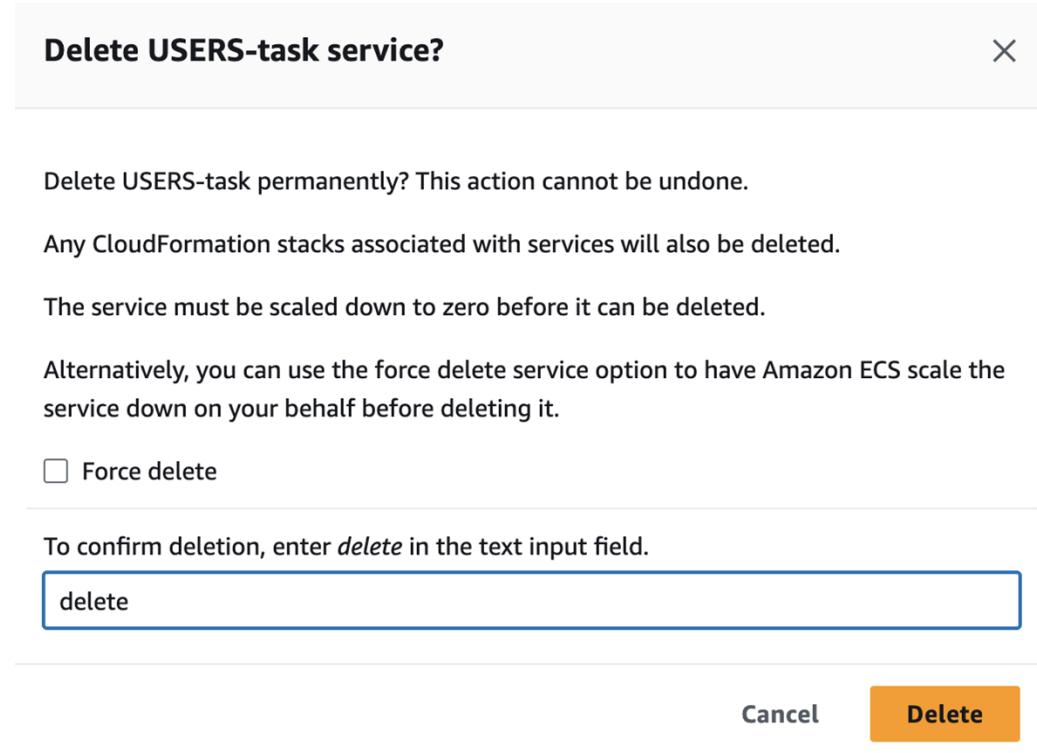
The screenshot shows a code editor with three tabs: Dockerfile, application.yml, and .env. The Dockerfile tab is active and contains the following code:

```
5 WORKDIR /app
6
7 # Crea una carpeta para logs (opcional, si necesitas almacenar logs en el contenedor)
8 RUN mkdir -p ./logs
9
10 # Copia el JAR compilado de tu aplicación desde el sistema host al contenedor
11 COPY ./build/libs/ms-user-0.0.1-SNAPSHOT.jar .
12
13 # Configura variables de entorno
14 ENV PORT 8002
15
16 # Expone el puerto en el que la aplicación escuchará dentro del contenedor
17 EXPOSE $PORT
18 CMD sleep 20
19
20 # Define el punto de entrada para ejecutar la aplicación Spring Boot
21 ENTRYPOINT ["java", "-jar", "ms-user-0.0.1-SNAPSHOT.jar"]
```

Creando un nuevo service

Primero, vamos a eliminar el service que creamos anteriormente para desplegar uno nuevo con ELB.

Ingresamos al clúster, seleccionamos delete service y lo eliminamos.



Creando un nuevo service

Nuevamente nos dirigimos a Task definition y creamos un nuevo servicio.

The screenshot shows the AWS Lambda Task definitions page. At the top, it displays "Task definitions (2) Info" and "Last updated October 23, 2024 at 17:13 (UTC-5:00)". Below this, there is a search bar labeled "Filter task definitions". A context menu is open over the first task definition, "USER-TASK-1", with options: "Deploy ▲", "Create new revision ▼", "Create new task definition ▼", "Create service", "status", "Update service", and "Run task". The "Create service" option is highlighted with a blue border. The table below lists the task definitions:

Task definition	Status of last revision
<input checked="" type="radio"/> USER-TASK-1	✓ ACTIVE
<input type="radio"/> task-user	✓ ACTIVE

Creando un nuevo service

Repetimos las configuraciones anteriores pero esta vez configuramos el ELB.

- Seleccionamos Application LoanBalancer
- Nombre del servicio: ELB-ECS
- Health check grace period: 5 (validación del estado de salud).

▼ **Load balancing - optional**
Configure load balancing using Amazon Elastic Load Balancing to distribute traffic evenly across the healthy tasks in your service.

Load balancer type [Info](#)
Configure a load balancer to distribute incoming traffic across the tasks running in your service.

Application Load Balancer

Container
The container and port to load balance the incoming traffic to

service_2 8005:8005

Host port:Container port

Application Load Balancer
Specify whether to create a new load balancer or choose an existing one.

Create a new load balancer
 Use an existing load balancer

Load balancer name
Assign a unique name for the load balancer.

Health check grace period [Info](#)
0 seconds

Listener [Info](#)
Specify the port and protocol that the load balancer will listen for connection requests on.

Create new listener
 Use an existing listener
You need to select an existing load balancer.

Port 80

Protocol HTTP

Creando un nuevo service

- Puerto de escucha: 80
- Target group name: target_usuario
- Dereistration delay: 300 (tiempo de espera).
- Se crea la tarea.

▼ **Load balancing - optional**
Configure load balancing using Amazon Elastic Load Balancing to distribute traffic evenly across the healthy tasks in your service.

Load balancer type [Info](#)
Configure a load balancer to distribute incoming traffic across the tasks running in your service.

Application Load Balancer

Container
The container and port to load balance the incoming traffic to

service_2 8005:8005

Host port:Container port

Application Load Balancer
Specify whether to create a new load balancer or choose an existing one.

Create a new load balancer
 Use an existing load balancer

Load balancer name
Assign a unique name for the load balancer.

Health check grace period [Info](#)
0 seconds

Listener [Info](#)
Specify the port and protocol that the load balancer will listen for connection requests on.

Create new listener
 Use an existing listener
You need to select an existing load balancer.

Port 80

Protocol HTTP

Creando un nuevo service

- Una vez creado el nuevo servicio, se despliega en el clúster (tarda algunos minutos en actualizarse).

The screenshot shows two stacked AWS CloudWatch Metrics dashboards.

Top Dashboard: Cluster overview

- ARN:** arn:aws:ecs:us-east-1:438465164320:cluster/user-cluster-1
- Status:** Active
- CloudWatch monitoring:** Default
- Registered container instances:** -
- Services:**
 - Draining: -
 - Active: 1
- Tasks:**
 - Pending: 0
 - Running: 1
- Encryption:**
 - Managed storage: -
 - Fargate ephemeral storage: -

Bottom Dashboard: Services (1) Info

- Services (1) Info:** USERS-task-ELB
- ARN:** arn:aws:ec...
- Status:** Active
- Type:** REPLICA
- Tasks running:** 0/1 Tasks running

Creando un nuevo service

- Validamos el estado de la tarea en el log para ver que se despliegue correctamente.

Logs (100+) Info			
You can use the filter bar below to search for and match terms, phrases, or values in your log events. Learn more about filter patterns			
Timestamp (UTC-05:00)	Message	Task	Container
October 23, 2024 at 17:28 (UTC-5:00)	2024-10-23T22:28:30.323Z INFO 1 --- [MS-USER] [main] com.ms.user.MsUserApplication : Started MsUserApplication in 15.548 seconds (process running for 16.892)	f69ab2b27a8b4d3c8 6217fc7c06f019	service_2
October 23, 2024 at 17:28 (UTC-5:00)	2024-10-23T22:28:30.246Z INFO 1 --- [MS-USER] [main] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat started on port 8002 (http) with context path ''	f69ab2b27a8b4d3c8 6217fc7c06f019	service_2
October 23, 2024 at 17:28 (UTC-5:00)	2024-10-23T22:28:28.964Z WARN 1 --- [MS-USER] [main] JpaBaseConfiguration\$JpaWebConfiguration : spring.jpa.open-in-view is enabled by default. Therefore, database queries may be performed during view rendering. Explicitly configure spring.jpa.open-in-view to disable this warning	f69ab2b27a8b4d3c8 6217fc7c06f019	service_2
October 23, 2024 at 17:28 (UTC-5:00)	2024-10-23T22:28:27.670Z INFO 1 --- [MS-USER] [main] o.s.d.j.r.query.QueryEnhancerFactory : Hibernate is in classpath; If applicable, HQL parser will be used.	f69ab2b27a8b4d3c8 6217fc7c06f019	service_2
	2024-10-23T22:28:27.042Z INFO 1 --- [MS-USER] [main]		

Creando un nuevo service

- Obtenemos la IP pública para probar.

The screenshot shows the AWS Fargate console interface. At the top, there's a header for "Fargate ephemeral storage" with an "Encryption" section (Info, Default AWS Fargate encryption) and a "Size (GiB)" of 20. Below this is a "Configuration" section with the following details:

Operating system/Architecture	Capacity provider	ENI ID	Network mode
Linux/ARM64	-	eni-0ce6ebbd9c69f718	awsvpc
CPU Memory	Launch type	Private IP	MAC address
1 vCPU 3 GB	FARGATE	44.223.49.20 open address	16:ff:e0:42:06:ab
Platform version	Container instance ID	Subnet ID	
1.4.0	-	subnet-0d7aaa7d294b9219e	
	Task definition: revision		
	USER-TASK-1:1		
	Task group		
	service:USERS-task-ELB		

A tooltip "Public IP copied" with a checkmark icon is overlaid on the public IP address "44.223.49.20". Below the configuration section, there's a "Container details for service_2" section with tabs for "Details", "Log configuration", "Restart policy", "Network bindings", "Docker labels and hosts", "Environment variables and files", and "Volumes". The "Details" tab is selected.

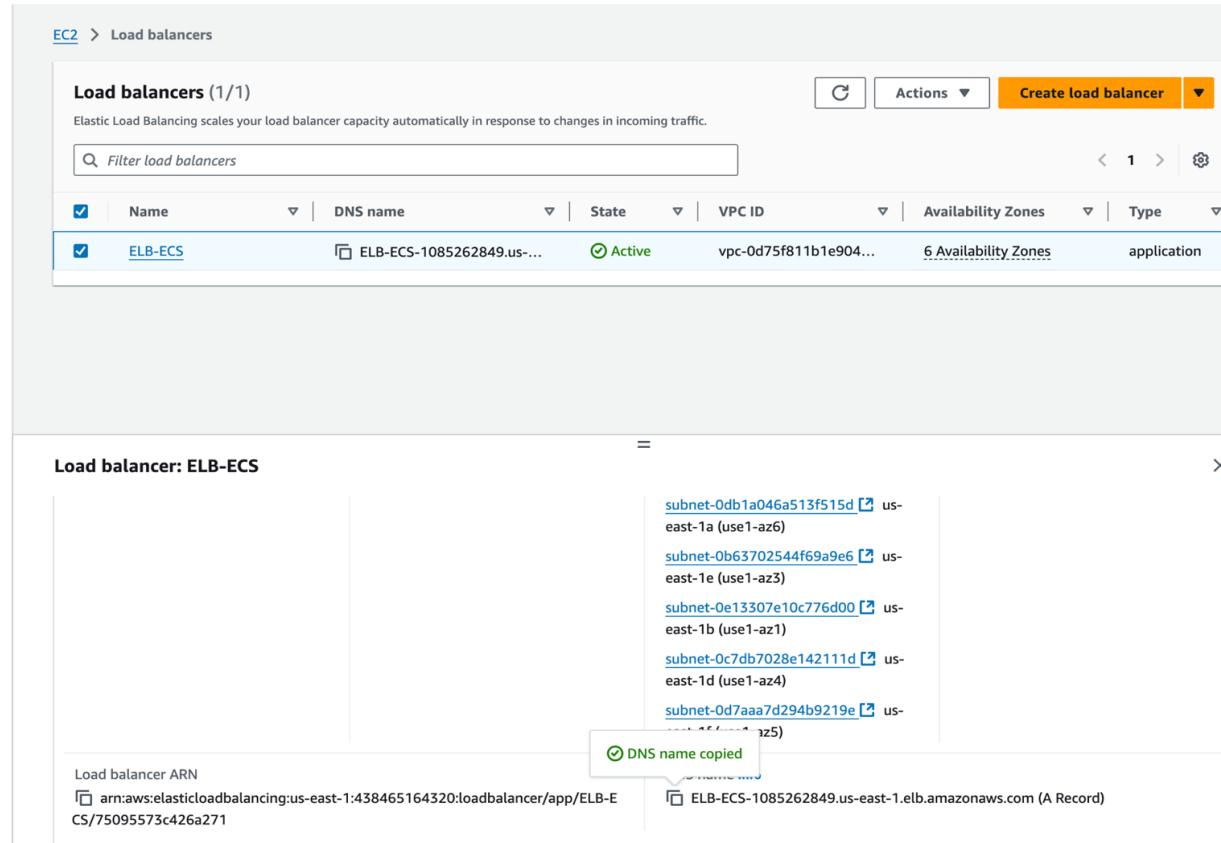
Creando un nuevo service

- Nos dirigimos al navegador para verificar.

The screenshot shows a browser window displaying the Swagger UI for a 'User' API endpoint at `/api/v2/user`. The page title is 'User API exposed for management all user'. The main section describes a `GET /api/v2/user` operation to get all users. It indicates 'No parameters' and provides a 'Try it out' button. Below this, the 'Responses' section lists four status codes: 200 (List of users, media type application/json), 401 (Unauthorized, media type application/json), 404 (endpoint not found, media type application/json), and 500 (Internal Server Error, media type application/json). At the bottom, there is a green button for a `POST /api/v2/user` operation to create a user.

Creando un nuevo service

- Para validar por medio del ELB nos dirigimos a EC2 > Load balancers.



Creando un nuevo service

- Probamos de nuevo en la web para validar que todo sea correcto.

User API exposed for management all user

GET /api/v2/user get users

This operation is for get all the users

Parameters

No parameters

Responses

Code	Description	Links
200	List of users Media type: application/json Controls Accept header.	No links
401	Unauthorized Media type: application/json	No links
404	endpoint not found Media type: application/json	No links
500	Internal Server Error Media type: application/json	No links

1.17. Probando los servicios ECS en Postman

Postman es una herramienta que se utiliza para probar los servicios de backend o las APIs.

En esta sección vamos a aprender como probar el servicio que desplegamos en AWS ECS.

Probando postman

Para obtener la IP en la cual se desplegó el backend, ingresamos a task overview y ubicamos la IP pública.

The screenshot shows the AWS Task Overview page for a task named 'USER-TASK-1'. The top navigation bar includes tabs for Configuration, Logs, Networking, Volumes (0), and Tags. The Configuration tab is selected.

Task overview

ARN arn:aws:ecs:us-east-1:4384651643 20:task/user-cluster-1/d4ad8101b50a4 6b8b7cf1f52fb85b79	Last status Running	Desired status Running	Started/Created at October 23, 2024 at 16:25 (UTC-5:00) October 23, 2024 at 16:25 (UTC-5:00)
---	------------------------	---------------------------	--

Fargate ephemeral storage

Encryption Info Default AWS Fargate encryption	Size (GiB) 20
---	------------------

Configuration

Operating system/Architecture Linux/ARM64	Capacity provider -	ENI ID eni-0ad4f8bcda997d4ab Copy	Public IP 52.207.233.169 Open address
CPU Memory 1 vCPU 3 GB	Launch type FARGATE	Network mode awsvpc	Private IP 172.31.80.34
Platform version 1.4.0	Container instance ID -	Subnet ID subnet-0c1cca33372636486 Copy	MAC address 12:5a:8f:98:8b:17
	Task definition: revision USER-TASK-1:1		

Probando postman

La aplicación cuenta con swagger por lo cual podemos ingresar a la siguiente URL

<http://52.207.233.169:8002/swagger-ui/index.html>

Nota: Si tiene dudas del puerto validar el log del servicio.

The screenshot shows the Swagger UI interface for an OpenAPI definition. The title bar indicates the URL is `http://52.207.233.169:8002/swagger-ui/index.html`. The main header says "OpenAPI definition v0 OAS 3.0" and "Supported by SMARTBEAR". Below this, there's a "Servers" dropdown set to "http://52.207.233.169:8002 - Generated server url". The main content area is titled "User API exposed for management all user". It lists several endpoints:

- GET /api/v2/user get users
- POST /api/v2/user create user
- GET /api/v2/user/{id} find by id user
- DELETE /api/v2/user/{id} delete user
- PATCH /api/v2/user/{id} update user
- GET /api/v2/user/{id}/rankings find by id user with rankings
- GET /api/v2/user/test-error
- GET /api/v2/user/find-by-document/ Get users by type document and document

Probando postman

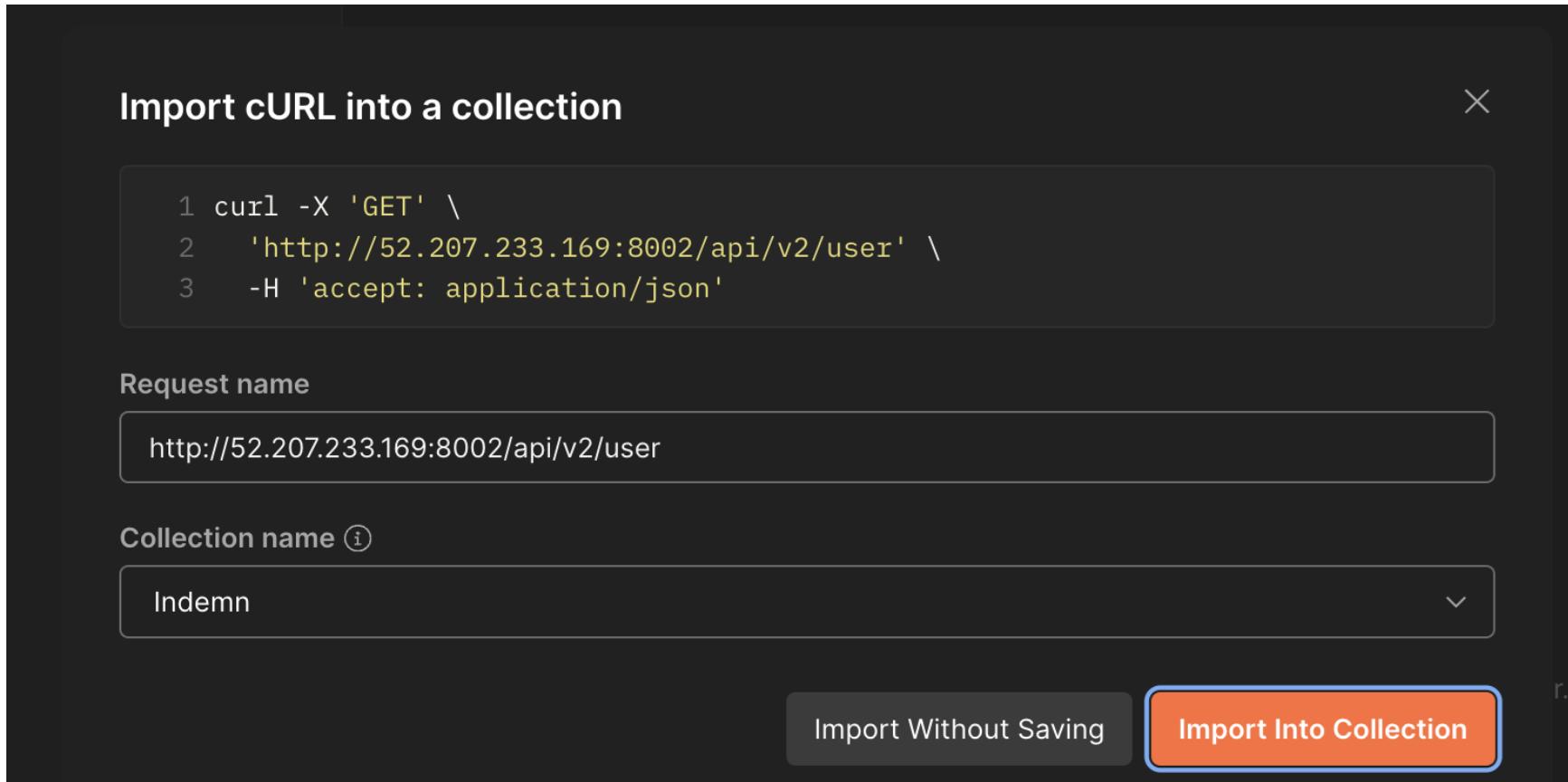
Para probar el servicio seleccionamos un servicio y probamos con try.

The screenshot shows the Postman interface for testing a 'User' API endpoint. The endpoint is `/api/v2/user` and the method is `GET`. The description is "This operation is for get all the users". Under 'Parameters', it says "No parameters". There are 'Execute' and 'Clear' buttons. In the 'Responses' section, there's a 'Curl' command and a 'Request URL' (http://52.207.233.169:8002/api/v2/user). The 'Server response' section shows a 200 status code. The 'Response body' is empty, and the 'Response headers' include:

```
connection: keep-alive
content-type: application/json
date: Wed,23 Oct 2024 21:37:08 GMT
keep-alive: timeout=60
transfer-encoding: chunked
```

Probando postman

Para probar en postman podemos tomar el cURL y lo agregamos en postman.



Probando postman

Probamos el servicio.

The screenshot shows the Postman application interface. At the top, there's a header with 'GET' and 'GET http://52.207.233.169:8'. Below the header, the URL 'http://52.207.233.169:8002/api/v2/user' is entered. To the right of the URL are 'Save' and 'Share' buttons. A large blue 'Send' button is prominently displayed. Below the URL input, tabs for 'Params', 'Auth', 'Headers (7)', 'Body', 'Scripts', and 'Settings' are visible, with 'Params' being the active tab. Under 'Params', there's a section for 'Query Params' with a table:

	Key	Value	Description	...	Bulk Edit
	Key	Value	Description	...	

At the bottom of the interface, there's a 'Body' dropdown set to 'Pretty', and a status bar showing '200 OK', '187 ms', '166 B', and a globe icon. The main body area displays the response: '1 []'.

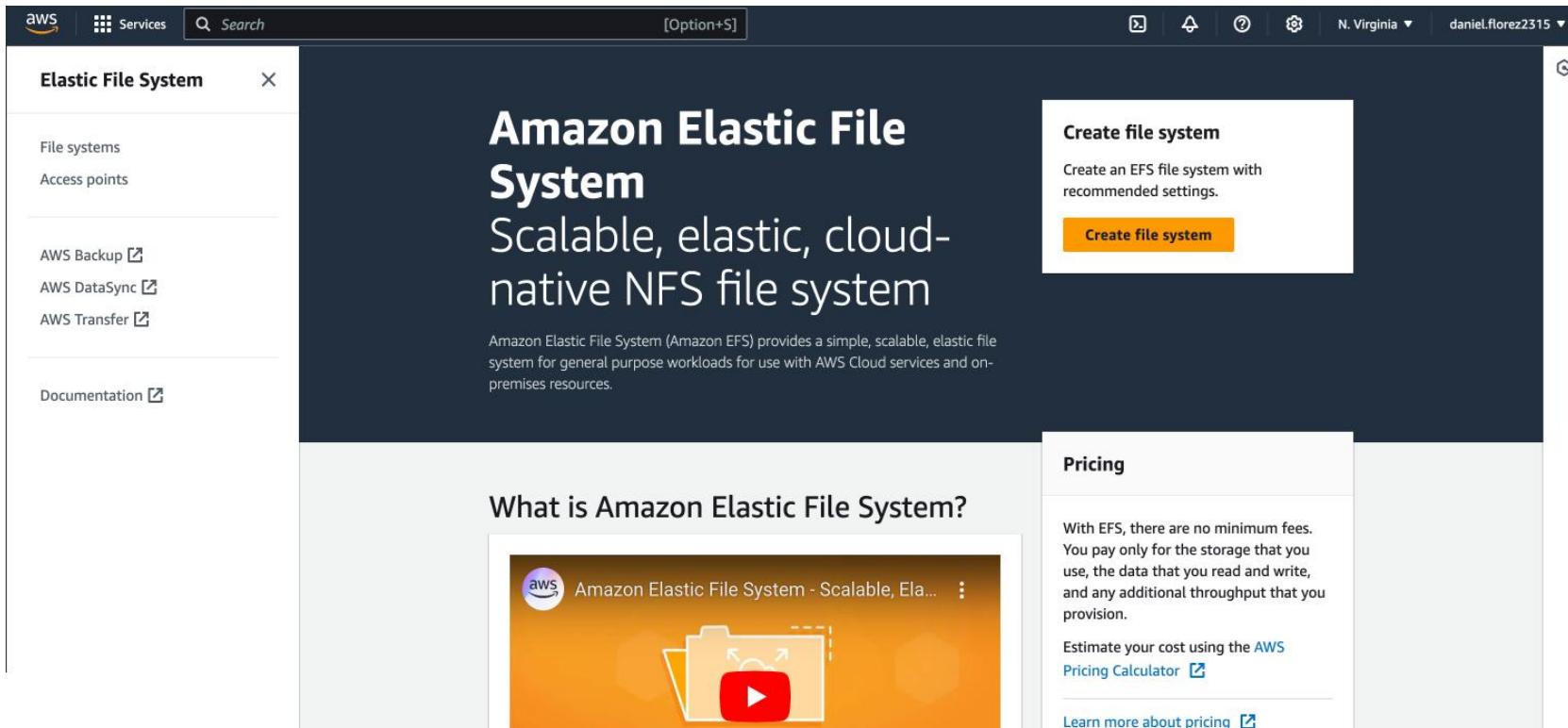
1.18. Agregando volumen EFS a contenedor

AWS EFS (Amazon Elastic File System) es un servicio de almacenamiento en la nube que proporciona un sistema de archivos elástico y escalable, diseñado para usarse con instancias de Amazon EC2.

Este servicio de EFS permite que múltiples servicios se conecten entre si.

Creando EFS

Buscamos en el Dashboard la opción de EFS.



Creando EFS

Creamos un nuevo EFS y seleccionamos customize para configurar.

Create file system X

Create an EFS file system with recommended settings. [Learn more](#)

Name - *optional*
Name your file system.

EFS-NETE

Name can include letters, numbers, and +-=_:/ symbols, up to 256 characters.

Virtual Private Cloud (VPC)
Choose the VPC where you want EC2 instances to connect to your file system.

vpc-0d75f811b1e904b1c
default

Cancel Customize Create

Creando EFS

Configuramos los backups, si es una multirregión o un regional.

General

Name - optional
Name your file system.

File system type
Choose to either store data across multiple Availability Zones or within a single Availability Zone. [Learn more](#)

Regional
Offers the highest levels of availability and durability by storing file system data across multiple Availability Zones within an AWS Region.

One Zone
Provides continuous availability to data within a single Availability Zone within an AWS Region.

Automatic backups
Automatically backup your file system data with AWS Backup using recommended settings. Additional pricing applies. [Learn more](#)

Enable automatic backups

We recommend that you create a backup policy for your file system

Lifecycle management
Automatically save money as access patterns change by moving files into the Infrequent Access (IA) or Archive storage class. [Learn more](#)

Transition into Infrequent Access (IA)	Transition into Archive	Transition into Standard
Transition files to IA based on the time since they were last accessed in Standard storage. <input type="button" value="30 day(s) since last access"/>	Transition files to Archive based on the time since they were last accessed in Standard storage. <input type="button" value="90 day(s) since last access"/>	Transition files back to Standard storage based on when they are first accessed in IA or Archive storage. <input type="button" value="None"/>

Encryption
Choose to enable encryption of your file system's data at rest. Uses the AWS KMS service key (aws/elasticfilesystem) by default. [Learn more](#)

Enable encryption of data at rest

Creando EFS

Configuramos la network.

Tener en cuenta que debemos seleccionar la network en la que está nuestro ECS.

Network

Virtual Private Cloud (VPC) [Learn more](#) Choose the VPC where you want EC2 instances to connect to your file system.

vpc-0d75f811b1e904b1c default

Mount targets

A mount target provides an NFSv4 endpoint at which you can mount an Amazon EFS file system. We recommend creating one mount target per Availability Zone. [Learn more](#)

Availability zone	Subnet ID	IP address	Security groups	Remove
us-east-1a	subnet-0db1a046a...	Automatic	Choose security gro... sg-01df16fa7312b32c3 default	X Remove
us-east-1b	subnet-0e13307e1...	Automatic	Choose security gro... sg-01df16fa7312b32c3 default	X Remove
us-east-1c	subnet-0c1cca333...	Automatic	Choose security gro... sg-01df16fa7312b32c3 default	X Remove
us-east-1d	subnet-0c7db7028...	Automatic	Choose security gro... sg-01df16fa7312b32c3 default	X Remove
us-east-1f	subnet-0d7aaa7d2...	Automatic	Choose security gro... sg-01df16fa7312b32c3 default	X Remove

Creando EFS

Por último, creamos el EFS para luego asignarlo a nuestro ECS.

The screenshot shows the Amazon EFS console interface. At the top, a green success banner displays the message: "Success! File system (fs-06913aeed3b411e4e) is available." Below the banner, the navigation bar shows "Amazon EFS > File systems". The main area is titled "File systems (1)". A "Create file system" button is visible in the top right of this section. The table below lists one file system entry:

Name	File system ID	Encrypted	Total size	Size in Standard	Size in IA	Size in Archive	Provisioned Throughput (MiB/s)
EFS-NETE	fs-06913aeed3b411e4e	Unencrypted	6.00 KiB	6.00 KiB	0 Bytes	0 Bytes	-

Agregar un EFS

Para agregar un EFS debemos ir a “EBS – Task definition” y creamos una nueva versión.

The screenshot shows the AWS Lambda Task Definitions page. At the top, it displays "Task definitions (2) Info" and "Last updated October 23, 2024 at 16:54 (UTC-5:00)". There are buttons for "Deploy", "Create new revision", and "Create new task definition". A prominent orange button labeled "Create new task definition" is highlighted. Below the buttons, there's a search bar with "Filter task definitions" and a dropdown menu set to "Active". A filter bar also shows "task-user" selected. The main table lists one task definition:

Task definition	Status of last revision
task-user	ACTIVE

Agregar un EFS

Buscamos la sección de Storage y add volumen.

The screenshot shows the 'Volumes' section of the AWS Lambda 'Add volume' configuration interface. It includes fields for Volume name, Configuration type (set to 'Configure at task definition creation'), Volume type (set to 'EFS'), File system ID (set to 'EFS-NETE (fs-06913aeed3b411e4e)'), Root directory ('/ruta/disk'), Access point ID ('None'), and an 'Advanced configurations' button.

Volumes | [Info](#)
Add one or more data volumes for your task to provide additional storage for the containers in the task. For each data volume, you must add a mount point to specify where to mount the data volume in the container.

▼ Volume - 1

Volume name | [Info](#)

Up to 255 letters (uppercase and lowercase), numbers, hyphens, and underscores are allowed.

Configuration type | [Info](#)
Choose to configure a volume in the task definition or later at deployment.
 Configure at task definition creation
You can configure bind mount, Docker, Amazon EFS, and Amazon FSx for Windows File Server volumes when creating a task definition.
 Configure at deployment
You can configure 1 Amazon EBS volume when creating or updating a service, or when running a standalone task.

Volume type | [Info](#)

Storage configurations

File system ID | [Info](#)

Root directory | [Info](#)

Directory within EFS.

Access point ID | [Info](#)

Advanced configurations

Agregar un EFS

Una vez configurada, lo creamos para que tome el cambio.

Task definition successfully created
task-user:3 has been successfully created. You can use this task definition to deploy a service or run a task.

View task definition X

Amazon Elastic Container Service > Task definitions > task-user > Revision 3 > Containers

task-user:3

Deploy ▾ Actions ▾ Create new revision ▾

Overview		Info	
ARN	arn:aws:ecs:us-east-1:43846516 4320:task-definition/task-user:3	Status	ACTIVE
Task role	RoleUserECSTaskExecutionRolePolicy	Task execution role	ecsTaskExecutionRole
Time created	October 23, 2024 at 16:59 (UTC-5:00)		
App environment	Fargate		
Operating system/Architecture	Linux/X86_64		
Network mode	awsvpc		

Validar

Para validar la correcta configuración ingresamos a la task. En volumes podemos ver el volumen que se asignó.

Containers	JSON	Task placement	Volumes (1)	Requires attributes	Tags
Volumes (1)					
Amazon EFS (1)					
Volume name	▲	Root dire...	File system ID	Access point ID	Transit encryption
efs		/ruta/disk	fs-06913aeed3b411e4e [i]	-	-

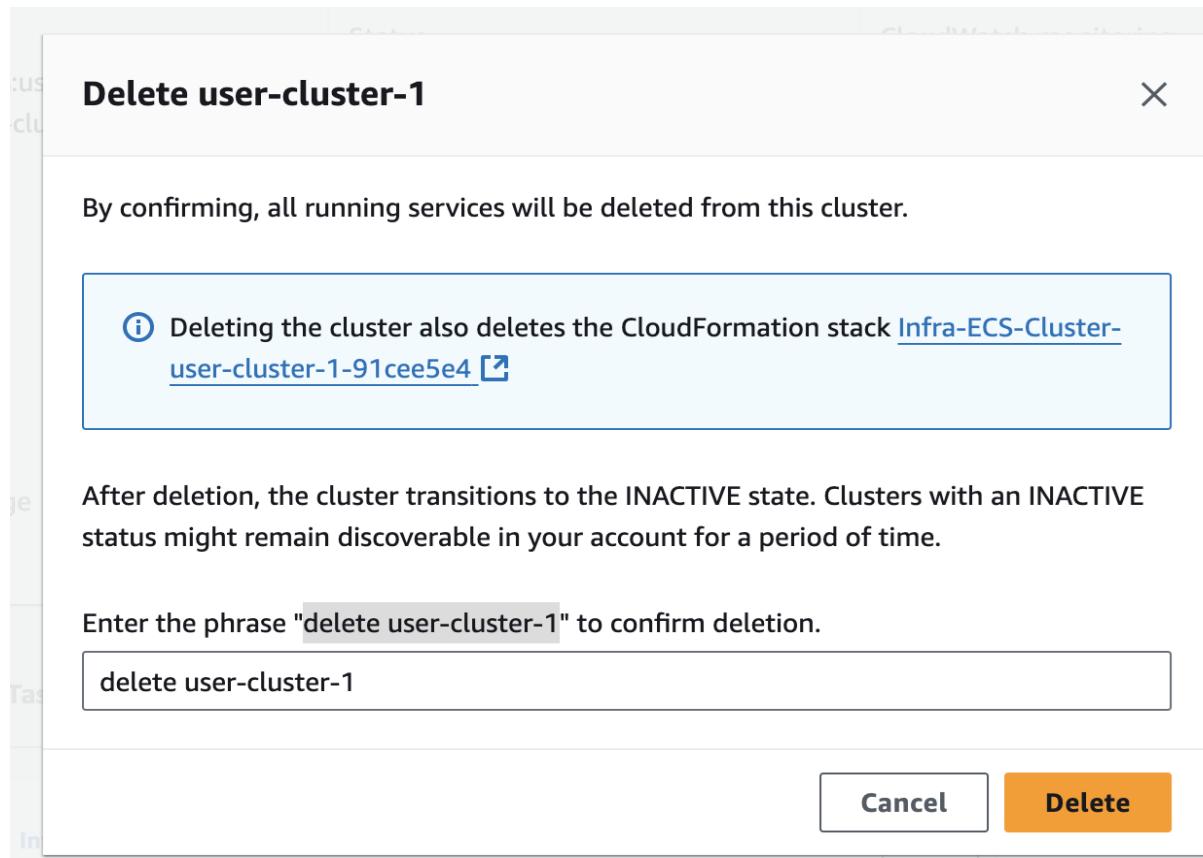
1.19. Finalizando y eliminando todos los recursos de ECS

Es importante eliminar lo que se creó en pasos previos para que no se incurran en costos.

En esta sección vamos a eliminar los servicios y configuraciones previas.

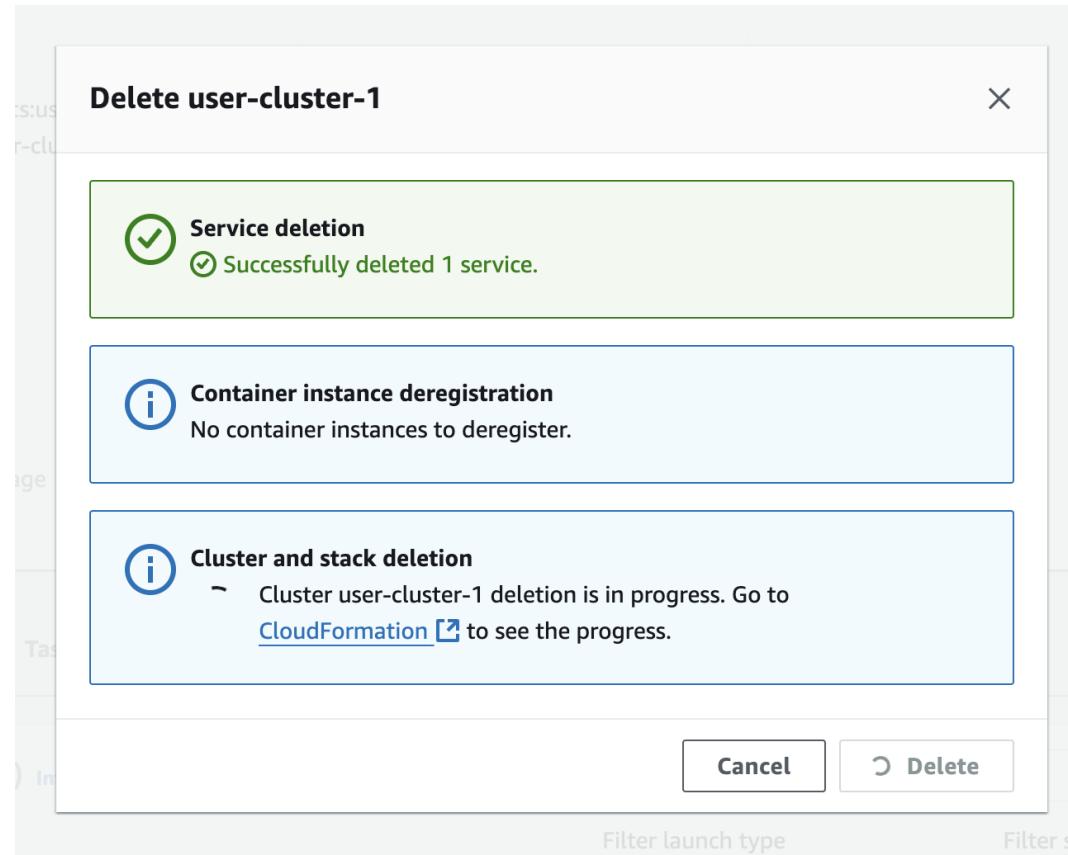
Eliminando clúster

Escribimos en la caja de texto lo que nos indica AWS.



Eliminando clúster

Lo elimina correctamente.



Eliminando clúster

Nota: Si no permite la eliminación del clúster, eliminar primero el service.

The screenshot shows the AWS Lambda interface. A green success message at the top reads "Successfully deleted USERS-task-ELB". Below it, the ECS Clusters page is displayed. The left sidebar includes links for Clusters, Namespaces, Task definitions, Account settings, Install AWS Copilot, Amazon ECR, and Repositories. The main content area shows a table header for "Clusters (0)" with columns for Cluster, Services, Tasks, Container instances, CloudWatch monitoring, and Capacity. A search bar labeled "Search clusters" is present. The message "No clusters" and "No clusters to display" is centered below the table. A "Create cluster" button is located in the top right corner of the main content area.

Eliminando ELB

Ingresamos a la tarea, en acciones hacemos clic en Deregister y automáticamente se eliminarán.

The screenshot shows the AWS Elastic Container Service (ECS) Task Definitions page. The URL is [Amazon Elastic Container Service > Task definitions > USER-TASK-1](#). The main title is **USER-TASK-1 (1/1)** with an [Info](#) link. To the right, it says "Last updated October 23, 2024 at 17:44 (UTC-5:00)". Below the title is a search bar with the placeholder [Filter task definition revisions by value](#). On the right side, there are several buttons: a "C" icon, "Deploy ▾", "Actions ▾", "Create new revision ▾", "Filter status", "Deregister", and "Delete". A dropdown menu for "Actions" is open, showing "Deregister" and "Delete". Below the search bar, there are two filter options: "Task definition: revision" (checked) and "Status". Under "Status", the entry "USER-TASK-1:1" is listed with a green checkmark and the word "ACTIVE".

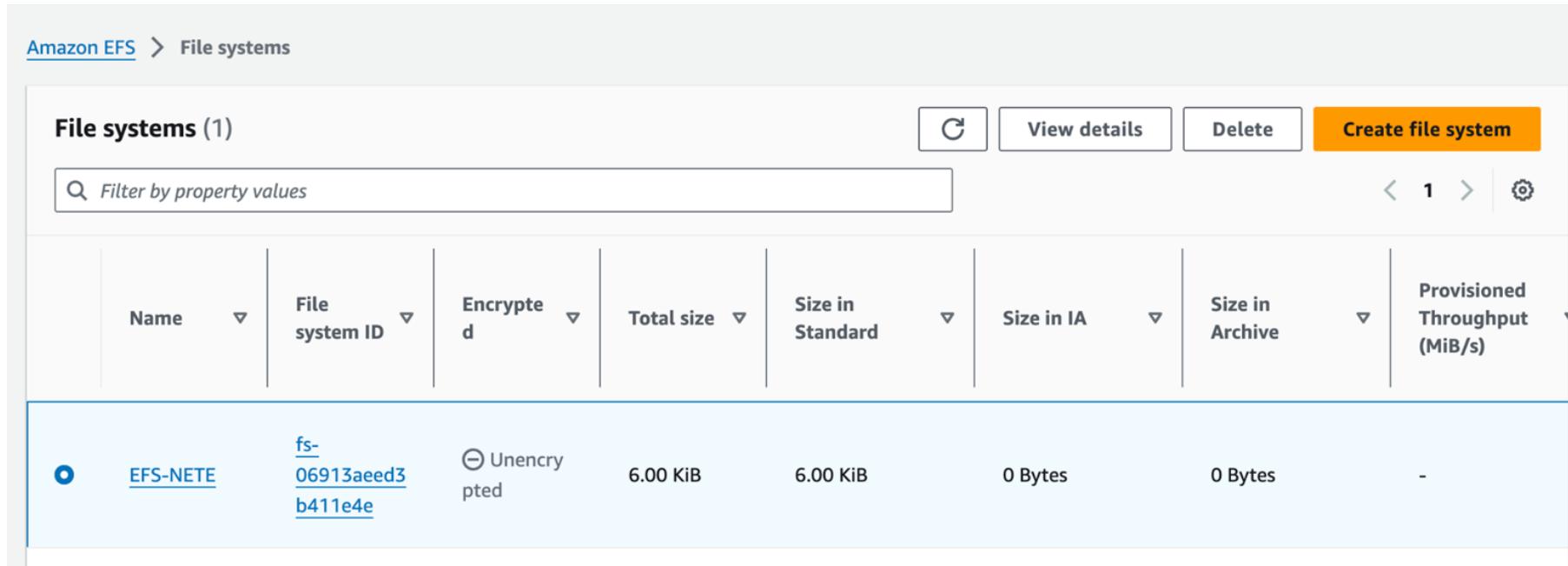
Eliminando Task

Ingresamos a EC2 > Load Balancers.
Seleccionamos el Load Balancer y lo eliminamos.

The screenshot shows the AWS EC2 Load Balancers console. In the top left, it says "EC2 > Load balancers". Below that, it displays "Load balancers (1/1)" with the message "Elastic Load Balancing scales your load balancer capacity automatically in response to changes in incoming traffic." A search bar labeled "Filter load balancers" is present. A table lists one load balancer: "ELB-ECS" with "DNS name" "ELB-ECS-1085262849.us-east-1.elb.amazonaws.com", "State" "Active", and "VPC ID" "vpc-0d75f811b1e904...". To the right of the table is an "Actions" dropdown menu with the following options: "Edit IP address type", "Edit subnets", "Manage instances", "Edit health check settings", "Manage listeners", "Edit security groups", "Edit load balancer attributes", "Manage tags", and "Delete load balancer". The "Delete load balancer" option is highlighted with a red box. The "Type" dropdown is set to "application".

Eliminando EFS

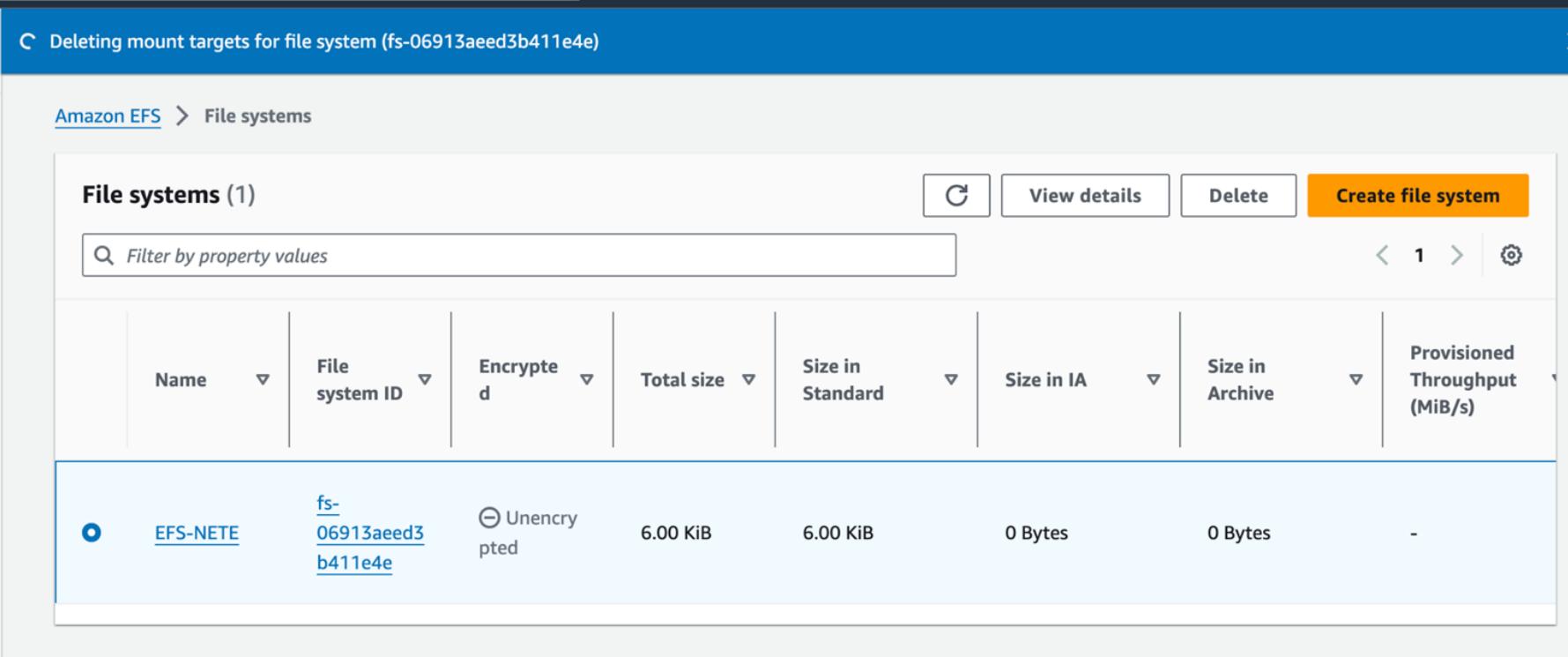
Ingresamos a EFS.
Seleccionamos el EFS y luego Delete.



The screenshot shows the 'File systems' page in the Amazon EFS console. At the top, there's a breadcrumb navigation: 'Amazon EFS > File systems'. Below it, a search bar with the placeholder 'Filter by property values' and a page number indicator '1'. The main table has columns: Name, File system ID, Encrypted, Total size, Size in Standard, Size in IA, Size in Archive, and Provisioned Throughput (MiB/s). The first row, representing the 'EFS-NETE' file system, is selected and highlighted with a blue border. The 'Delete' button in the top right corner of the table header is highlighted with an orange background.

Name	File system ID	Encrypted	Total size	Size in Standard	Size in IA	Size in Archive	Provisioned Throughput (MiB/s)
EFS-NETE	fs-06913aeed3b411e4e	Unencrypted	6.00 KiB	6.00 KiB	0 Bytes	0 Bytes	-

Eliminando EFS



The screenshot shows the Amazon EFS console interface. At the top, a blue header bar displays the message "C Deleting mount targets for file system (fs-06913aeed3b411e4e)". Below this, the navigation path "Amazon EFS > File systems" is visible. The main area is titled "File systems (1)" and contains a table with the following data:

Name	File system ID	Encrypted	Total size	Size in Standard	Size in IA	Size in Archive	Provisioned Throughput (MiB/s)
EFS-NETE	fs-06913aeed3b411e4e	Unencrypted	6.00 KiB	6.00 KiB	0 Bytes	0 Bytes	-

At the top right of the table, there are buttons for "View details", "Delete", and "Create file system". A search bar labeled "Filter by property values" is located above the table. The bottom right corner of the table has a small gear icon.

Resumen del capítulo

- Crear una imagen en Docker Hub para ser utilizado por distintos servicios de Cloud o local.
- Gestión de ECS para el despliegue de Contenedores.
- Entender la diferencia entre AKS y EKS.



Práctica 1.2 Creación de Servicio ECS

Objetivo:

- Desplegar el servicio de Emails en ECS.
- Realizar el despliegue de un servicio de backend que envía correos electrónicos.

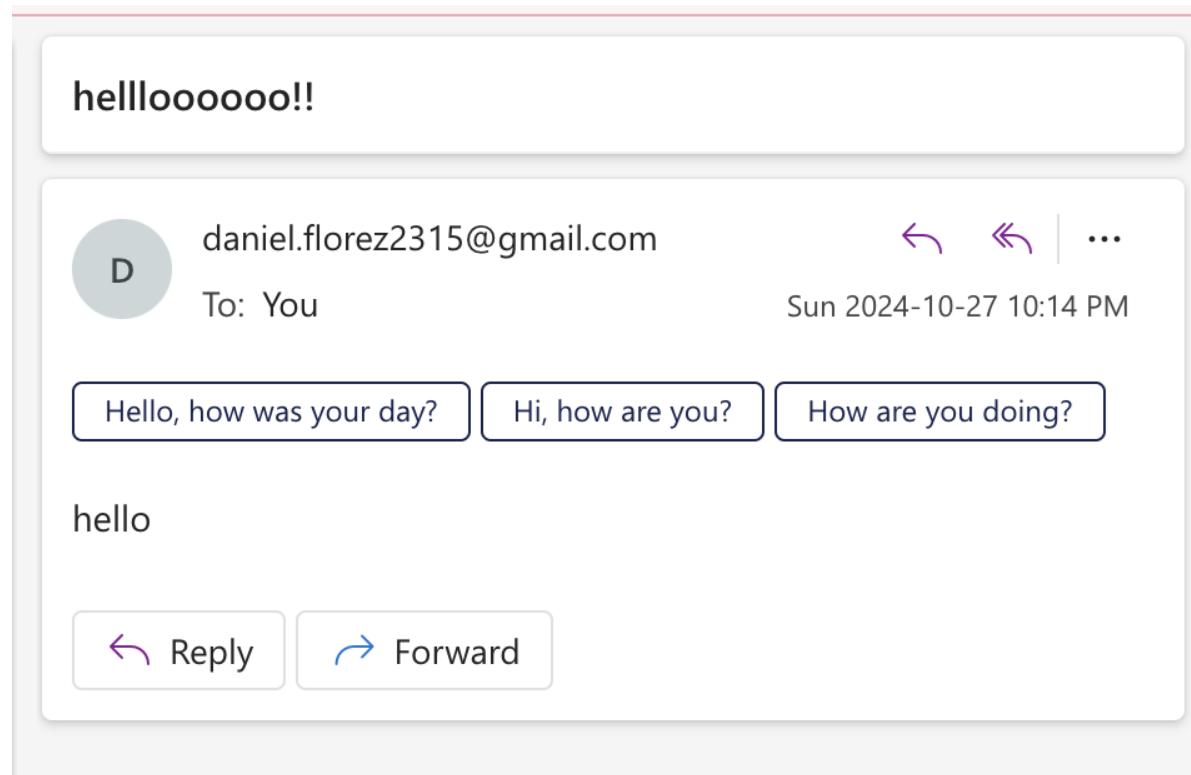
Planteamiento e instrucciones:

Realiza un seguimiento de los pasos indicados en la *Guía de Laboratorios* para llevar a cabo la tarea correspondiente en el siguiente enlace: https://netec-mx.github.io/DOCK_KUB_AVA/



Tiempo para esta actividad:
60 minutos.

Resultado esperado



Referencias Bibliográficas

- [AWS | Gestión de contenedores \(ECS\) compatible con los de Docker](#)
- [Azure Kubernetes Service \(AKS\) | Microsoft Azure](#)
- [Docker Hub | Docker Docs](#)
- [Instalar herramientas | Kubernetes](#)



Objetivos:

- Comprender que es EKS
- Desplegar microservicios con EKS
- servicios serverless fargate

Capítulo 2

Kubernetes: Despliegue en EKS (Elastic Kubertenes Service)

2.1. Aplicando todos los objetos Pods, deployments, svc en EKS y probando en Postman

EKS es el servicio de Kubernetes en el Cloud de Amazon Web Service. EKS nos permite crear una infraestructura escalable, tolerante a fallos y autogestionada por AWS.

Para este módulo vamos a ingresar a la consola de AWS para configurar el servicio de EKS.

AWS EKS

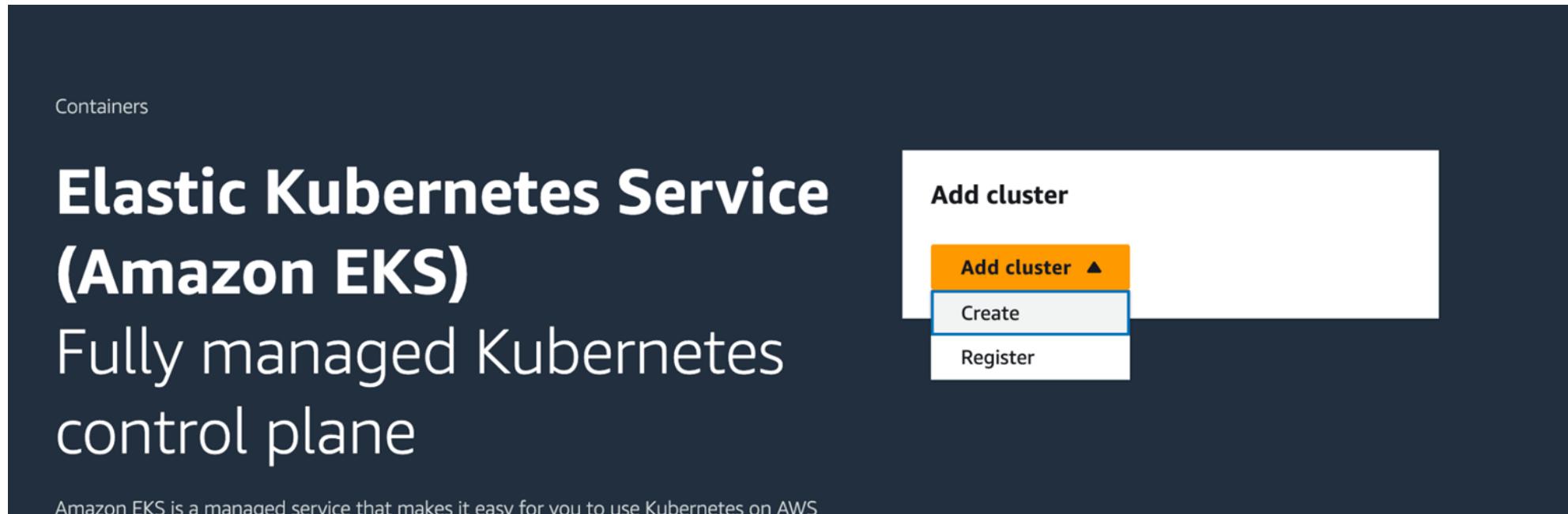
Ingresamos a la consola de AWS y buscamos el servicio de EKS.

The screenshot shows the AWS EKS service page. On the left, there is a sidebar with the title "Amazon Elastic Kubernetes Service" and a list of options: Clusters, Amazon EKS Anywhere (selected), Enterprise Subscriptions (New), Related services (Amazon ECR, AWS Batch), Console settings, Documentation (link), and Submit feedback. The main content area has a dark background with white text. It features the heading "Elastic Kubernetes Service (Amazon EKS)" and the subtext "Fully managed Kubernetes control plane". Below this, a paragraph explains that Amazon EKS is a managed service that makes it easy to use Kubernetes on AWS without needing to install and operate your own Kubernetes control plane. To the right, there is a call-to-action button labeled "Add cluster" with a dropdown arrow. At the bottom, there is a section titled "How it works" with a diagram showing a flow from three orange cubes to a central cluster of cubes, then to a laptop icon with a K8s logo, and finally to a single cube. To the right of the diagram is a "Pricing" table with four rows: EKS control plane (link to EKS pricing), Worker nodes (link to EC2 pricing), Fargate pods (link to Fargate pricing), and EKS Anywhere (link to EKS Anywhere pricing).

EKS control plane	EKS pricing
Worker nodes	EC2 pricing
Fargate pods	Fargate pricing
EKS Anywhere	EKS Anywhere pricing

Creando AWS EKS

Ingresamos a la opción Add cluster para crear nuestro clúster en EKS.



Creando AWS EKS

Le damos un nombre al clúster e ingresamos al menú de “Create a role in IAM console”.

Cluster configuration Info

Name
Enter a unique name for this cluster. This property cannot be changed after the cluster is created.

The cluster name should begin with letter or digit and can have any of the following characters: the set of Unicode letters, digits, hyphens and underscores. Maximum length of 100.

Cluster IAM role Info
Select the Cluster IAM role to allow the Kubernetes control plane to manage AWS resources on your behalf. This cannot be changed after the cluster is created. To create a new custom role, follow the instructions in the [Amazon EKS User Guide](#).

Crear ROLE para EKS

Seleccionamos AWS Service y EKS- Clutser.

IAM > Roles > Create role

Step 1
Select trusted entity

Step 2
Add permissions

Step 3
Name, review, and create

Select trusted entity Info

Trusted entity type

AWS service
Allow AWS services like EC2, Lambda, or others to perform actions in this account.

AWS account
Allow entities in other AWS accounts belonging to you or a 3rd party to perform actions in this account.

Web identity
Allows users federated by the specified external web identity provider to assume this role to perform actions in this account.

SAML 2.0 federation
Allow users federated with SAML 2.0 from a corporate directory to perform actions in this account.

Custom trust policy
Create a custom trust policy to enable others to perform actions in this account.

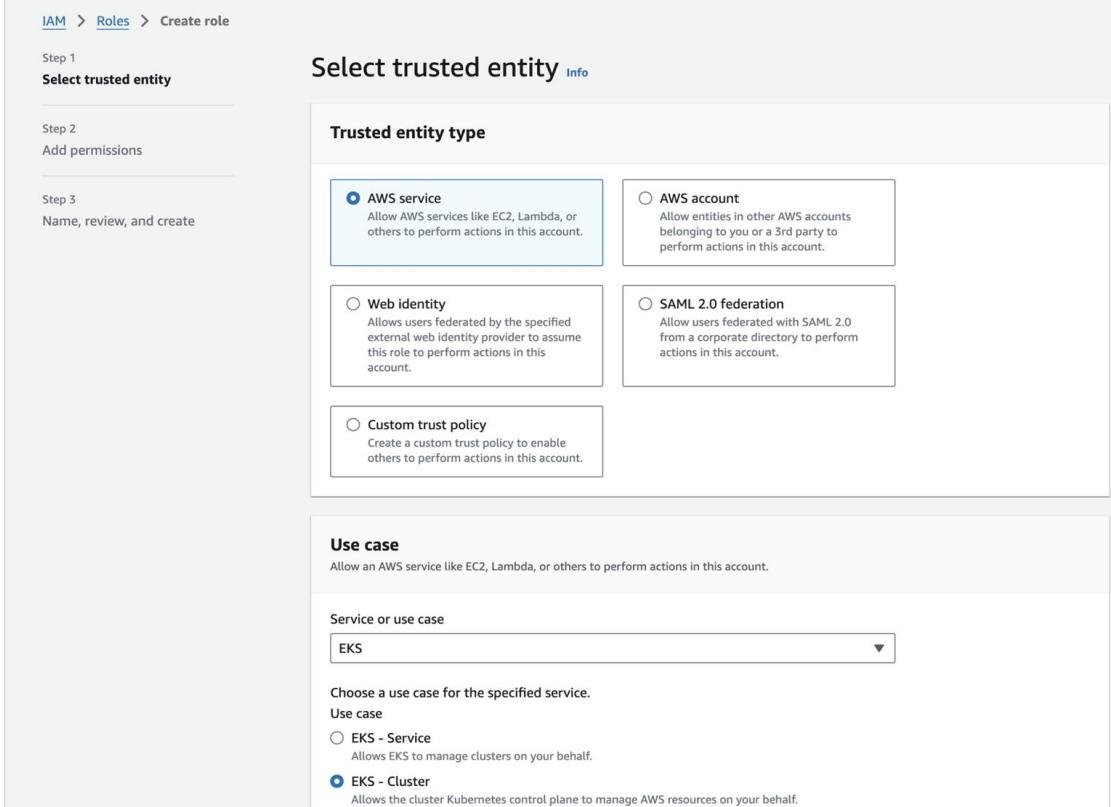
Use case
Allow an AWS service like EC2, Lambda, or others to perform actions in this account.

Service or use case
EKS

Choose a use case for the specified service.
Use case

EKS - Service
Allows EKS to manage clusters on your behalf.

EKS - Cluster
Allows the cluster Kubernetes control plane to manage AWS resources on your behalf.



Crear ROLE para EKS

Dejamos la configuración por default de AWS.

The screenshot shows the 'Add permissions' step of the IAM role creation wizard. On the left, a sidebar lists three steps: 'Select trusted entity' (highlighted in blue), 'Add permissions' (selected), and 'Name, review, and create'. The main area is titled 'Add permissions' with a 'Permissions policies (1)' section. It shows a single policy named 'AmazonEKSClusterPolicy' (type: AWS managed). Below this is an optional section for setting a 'permissions boundary'. At the bottom are 'Cancel', 'Previous', and 'Next' buttons, with 'Next' being orange and bold.

IAM > Roles > Create role

Step 1
[Select trusted entity](#)

Step 2
Add permissions

Step 3
Name, review, and create

Add permissions [Info](#)

Permissions policies (1) [Info](#)
The type of role that you selected requires the following policy.

Policy name	Type
AmazonEKSClusterPolicy	AWS managed

▶ Set permissions boundary - *optional*

Cancel Previous **Next**

Creación role

Para poder identificar el rol que estamos creando, le damos un nombre y descripción .

The screenshot shows the 'Create role' wizard in the AWS IAM console. The current step is 'Name, review, and create'. The left sidebar shows three steps: 'Select trusted entity' (selected), 'Add permissions', and 'Name, review, and create'. The main area is titled 'Role details'.

Role name: roleEksCluster

Description: example role EKS cluster

Step 1: Select trusted entities

Trust policy:

```
1 - {
2   "Version": "2012-10-17",
3   "Statement": [
4     {
5       "Effect": "Allow",
6       "Principal": {
7         "Service": [
8           "eks.amazonaws.com"
9         ]
10      },
11      "Action": "sts:AssumeRole"
12    }
13  ]
14 }
```

Agregar role

Refrescamos los roles y seleccionamos el rol que se creó en el paso anterior.

Configure cluster

Cluster configuration Info

Name
Enter a unique name for this cluster. This property cannot be changed after the cluster is created.

The cluster name should begin with letter or digit and can have any of the following characters: the set of Unicode letters, digits, hyphens and underscores. Maximum length of 100.

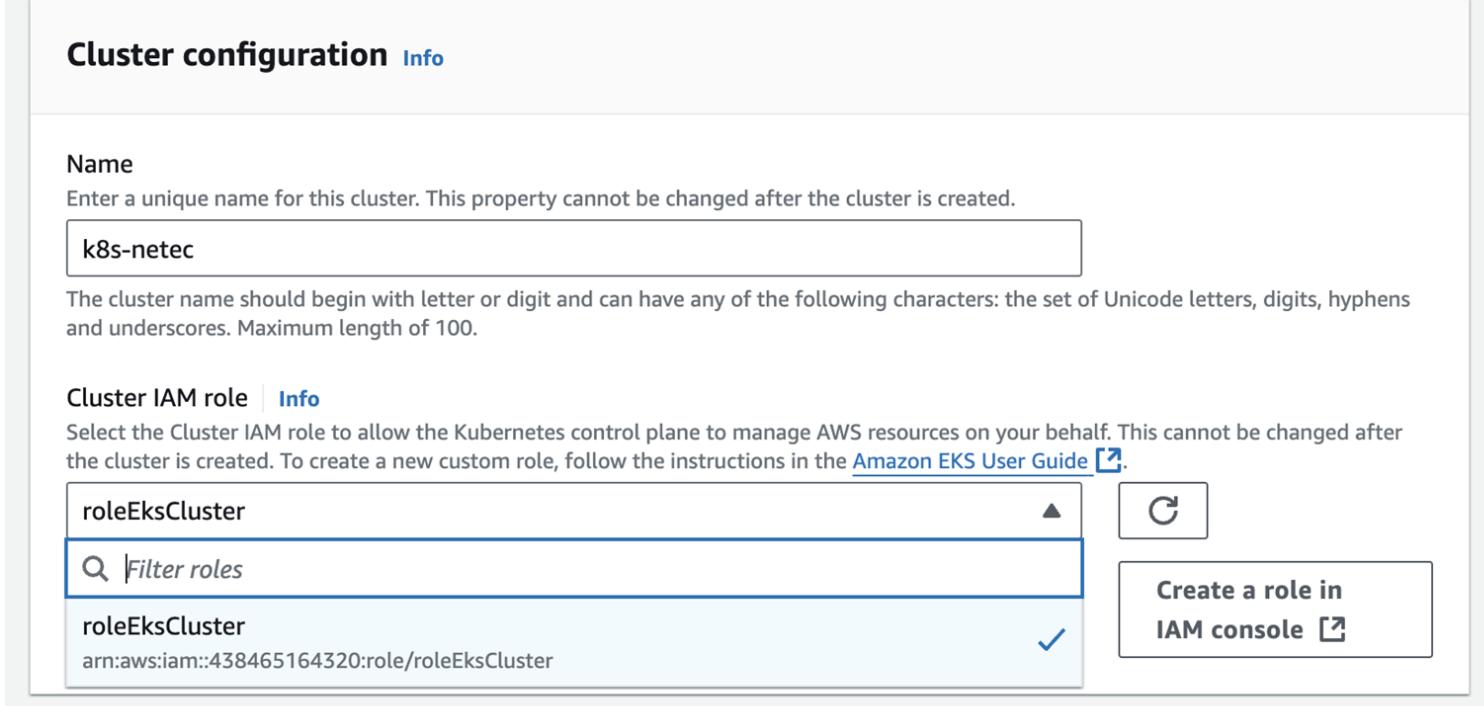
Cluster IAM role Info
Select the Cluster IAM role to allow the Kubernetes control plane to manage AWS resources on your behalf. This cannot be changed after the cluster is created. To create a new custom role, follow the instructions in the [Amazon EKS User Guide](#).

▲

Filter roles

roleEksCluster ✓

↗



Configurando EKS

Seleccionamos la versión más reciente y estándar de Kubernetes.

Kubernetes version settings

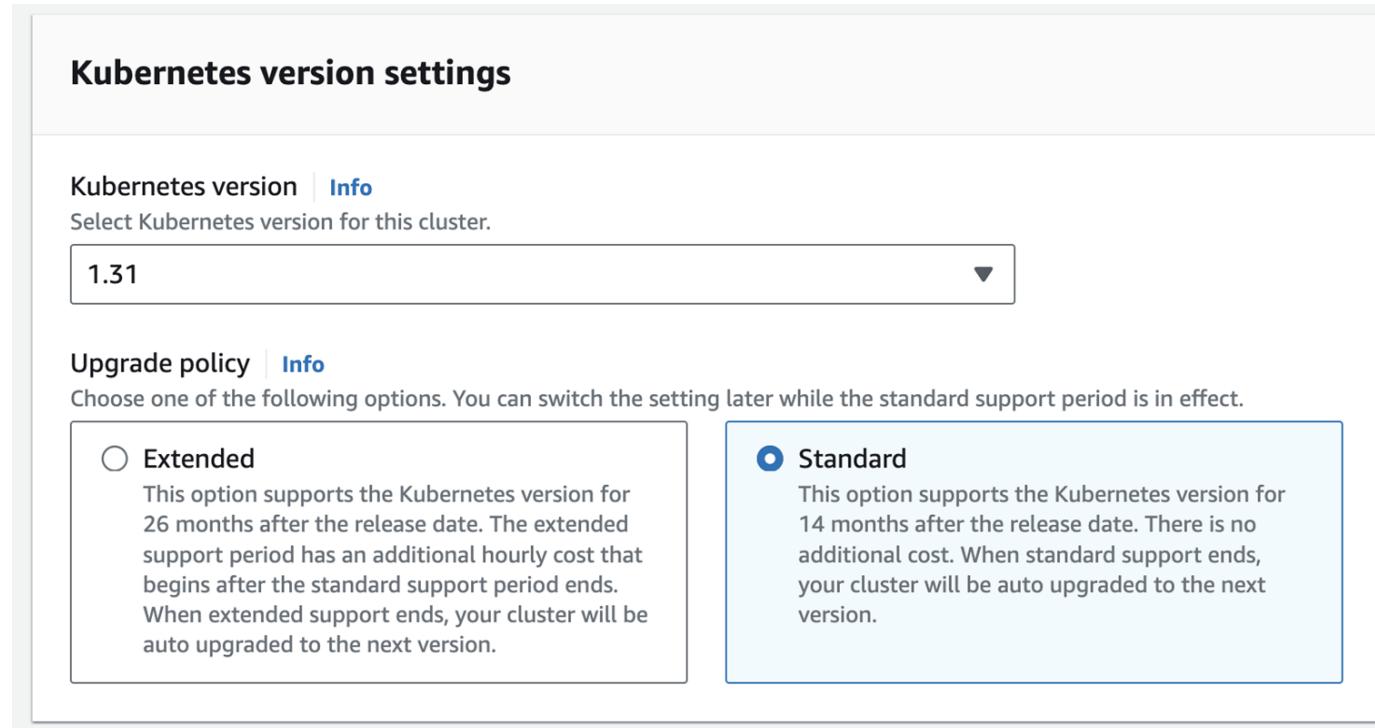
Kubernetes version | [Info](#)
Select Kubernetes version for this cluster.

1.31 ▾

Upgrade policy | [Info](#)
Choose one of the following options. You can switch the setting later while the standard support period is in effect.

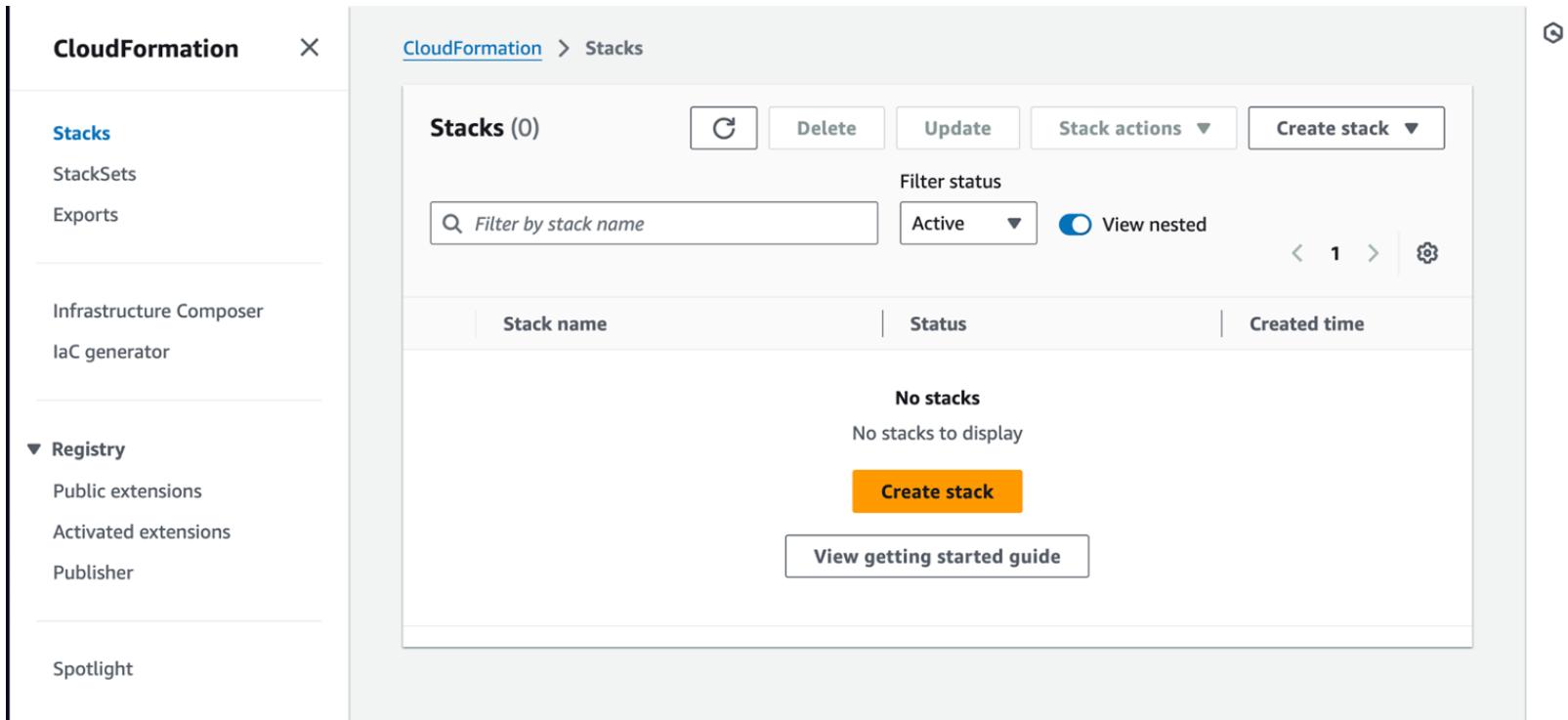
Extended
This option supports the Kubernetes version for 26 months after the release date. The extended support period has an additional hourly cost that begins after the standard support period ends.
When extended support ends, your cluster will be auto upgraded to the next version.

Standard
This option supports the Kubernetes version for 14 months after the release date. There is no additional cost. When standard support ends, your cluster will be auto upgraded to the next version.



Configurando EKS - CloudFormation

CloudFormation es un servicio que permite modelar y configurar sus recursos de infraestructura en la nube de forma automatizada.



Configurando EKS - CloudFormation

- Creamos un stack con CloudFormation para generar una VPC automáticamente.
- Ingresamos la URL <https://s3.us-west-2.amazonaws.com/amazon-eks/cloudformation/2020-10-29/amazon-eks-vpc-private-subnets.yaml> en el campo de Amazon S3 URL.

The screenshot shows the AWS CloudFormation 'Create stack' wizard. The first step, 'Create stack', is selected. It displays the 'Prerequisite - Prepare template' section. This section includes a note that you can also create a template by scanning existing resources or using the IaC generator. It provides three options for creating a template:

- Choose an existing template: Upload or choose an existing template.
- Use a sample template: Choose from our sample template library.
- Build from Infrastructure Composer: Create a template using a visual builder.

Below this, the 'Specify template' section is shown, which is currently empty. It includes a note that a template is a JSON or YAML file describing stack resources and properties. The 'Template source' section is also visible, with the 'Amazon S3 URL' option selected. This option allows providing an Amazon S3 URL to store the template. Other options include 'Upload a template file' (upload directly to the console) and 'Sync from Git' (sync from a Git repository). At the bottom, the 'Amazon S3 URL' field contains the URL: <https://s3.us-west-2.amazonaws.com/amazon-eks/cloudformation/2020-10-29/amazon-eks-vpc-private-subnets.yaml>.

Configurando EKS - CloudFormation

Le damos un nombre al stack y dejamos por default la plantilla de creación de VP seleccionada en el paso anterior.

CloudFormation > Stacks > Create stack

Step 1
[Create stack](#)

Step 2
Specify stack details

Step 3
Configure stack options

Step 4
Review and create

Specify stack details

Provide a stack name

Stack name

Stack name must be 1 to 128 characters, start with a letter, and only contain alphanumeric characters. Character count: 10/128.

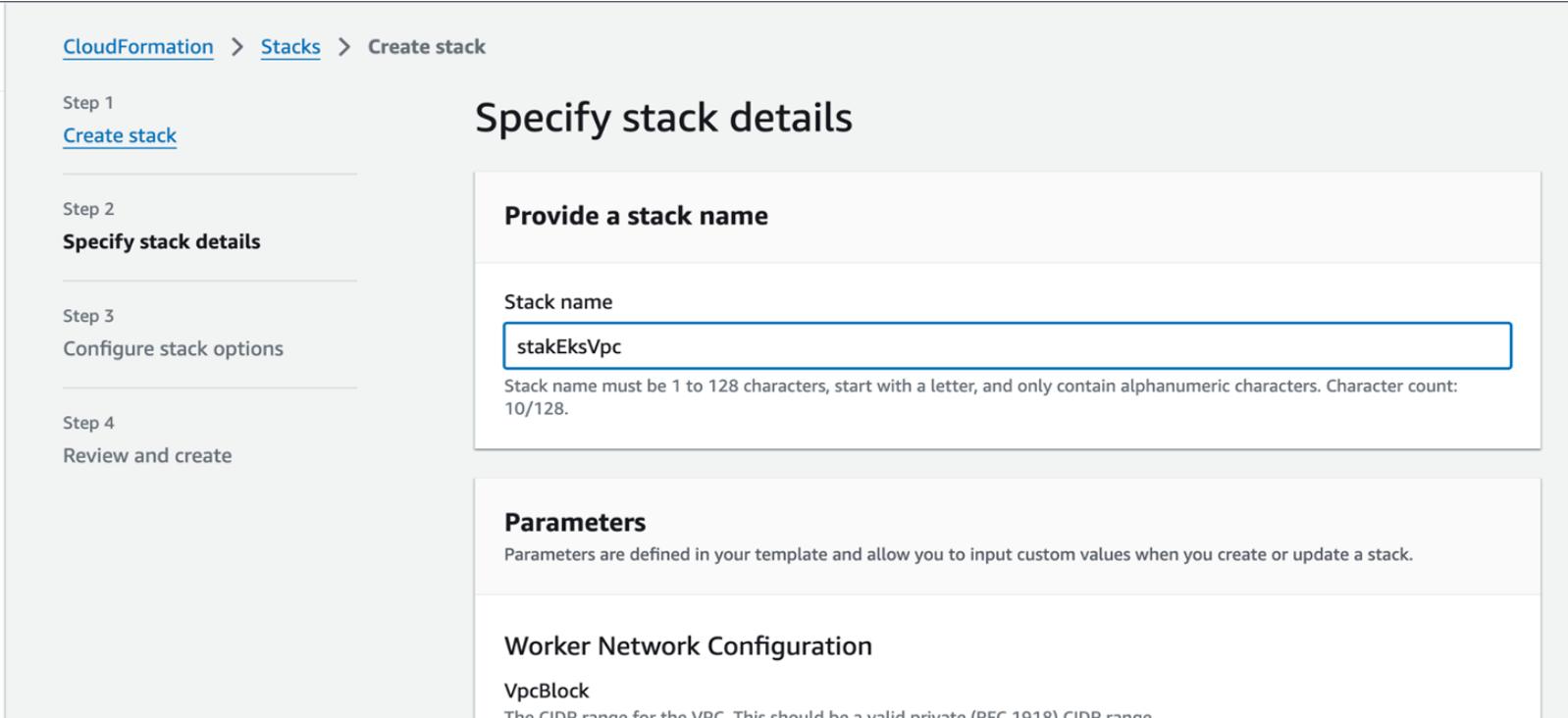
Parameters

Parameters are defined in your template and allow you to input custom values when you create or update a stack.

Worker Network Configuration

VpcBlock

The CIDR range for the VPC. This should be a valid private (RFC 1918) CIDR range.



Configurando EKS - CloudFormation

Cuando se aplica, empiezan a crearse automáticamente las redes.

The screenshot shows two views of the AWS CloudFormation console. On the left, the main 'CloudFormation' view displays a single stack named 'stakEksVpc' with a status of 'CREATE_IN_PROGRESS'. On the right, a detailed view of the same stack shows the 'Events' tab, which lists 68 events. Most of these events are for resources named 'NatGateway01' and 'NatGateway02', all of which are in 'CREATE_IN_PROGRESS' status. A few other events are listed with 'CONFIGURATION_COMPLETE' status. The table has columns for Logical ID, Status, and Detailed status.

Logical ID	Status	Detailed status
NatGateway02	CREATE_IN_PROGRESS	CONFIGURATION_COMPLETE
NatGateway01	CREATE_IN_PROGRESS	CONFIGURATION_COMPLETE
NatGateway02	CREATE_IN_PROGRESS	-
NatGateway01	CREATE_IN_PROGRESS	-
NatGateway02	CREATE_IN_PROGRESS	-
NatGateway01	CREATE_IN_PROGRESS	-
NatGateway02	CREATE_IN_PROGRESS	CREATE_COMPLETE

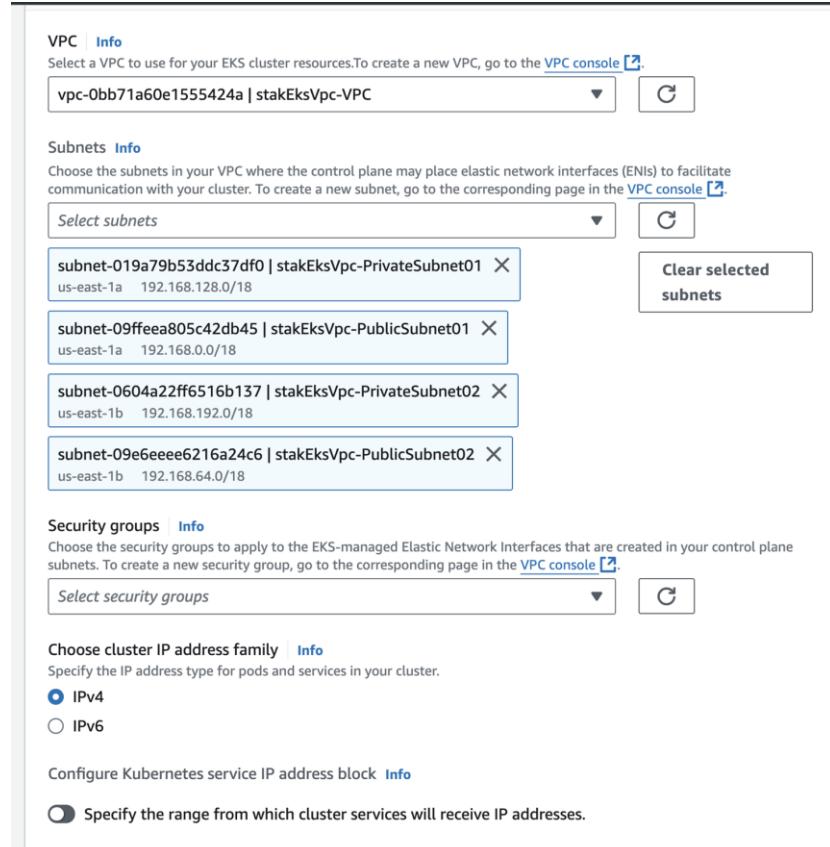
Configurando EKS-VPC

Refrescamos y seleccionamos la VPC que se creó con CloudFormation.

The screenshot shows the 'Specify networking' step of the EKS cluster creation wizard. On the left, a sidebar lists steps from 2 to 6. Step 2 is selected. The main area shows the 'Networking' configuration. It includes an 'Info' link stating that IP address family and service IP address range cannot be changed after cluster creation. A 'VPC' dropdown is set to 'vpc-0d75f811b1e904b1c | Default'. Below it is a 'Filter VPCs' input field. A list of VPCs is shown, with 'vpc-0d75f811b1e904b1c | Default' selected. To the right of the list is a note about using ENIs for VPC endpoints. A 'Clear selected subnets' button is also present. Further down, sections for 'Security groups' and 'Choose cluster IP address family' are visible, with 'IPv4' selected.

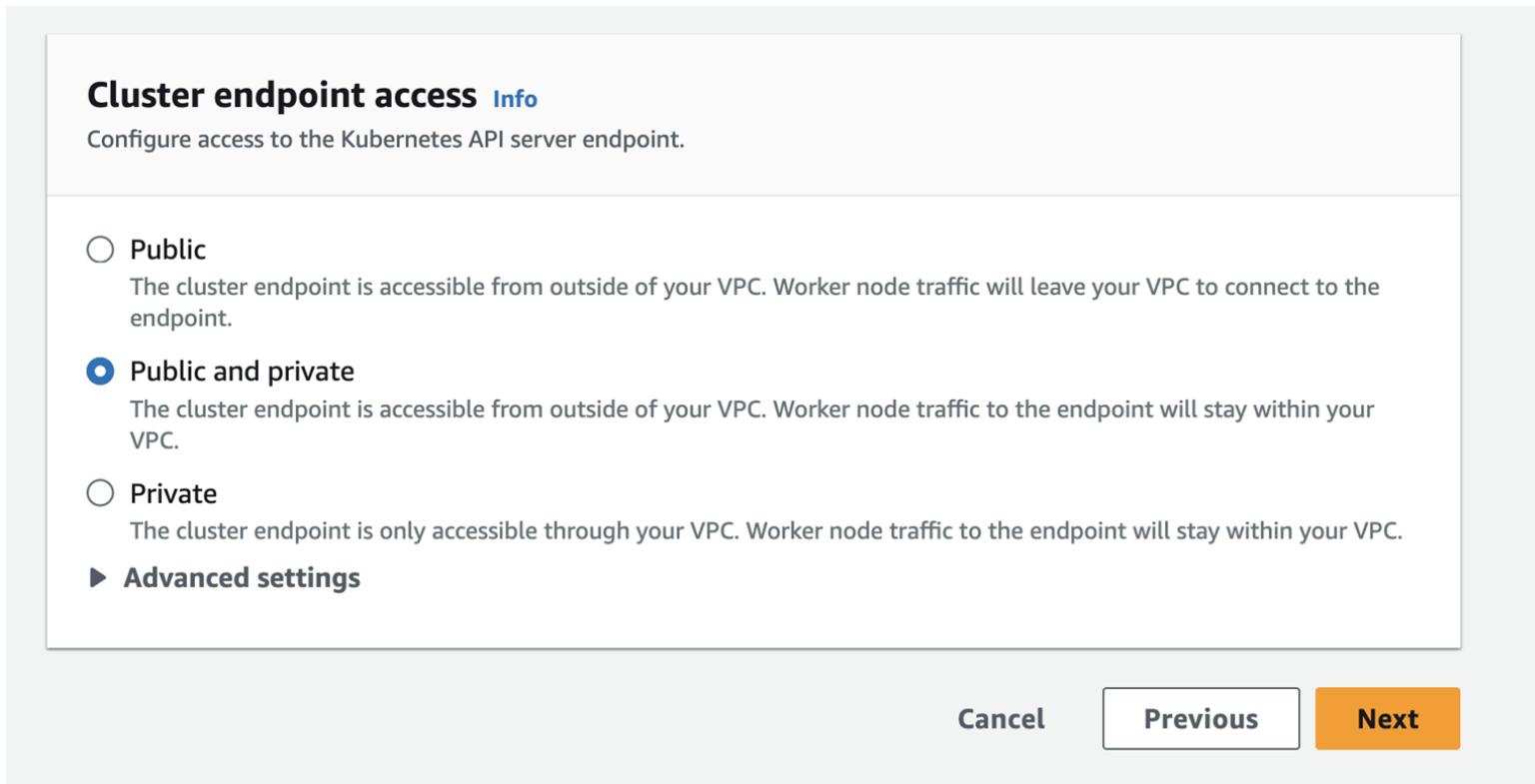
Configurando EKS-VPC

Dejamos la configuración por default de la VPC.



Configurando EKS-VPC

En Cluster endpoint access, elegimos público y privado. Después, seleccionamos next hasta crear el clúster.



Creación clúster EKS

Esperamos que se cree el clúster de EKS.

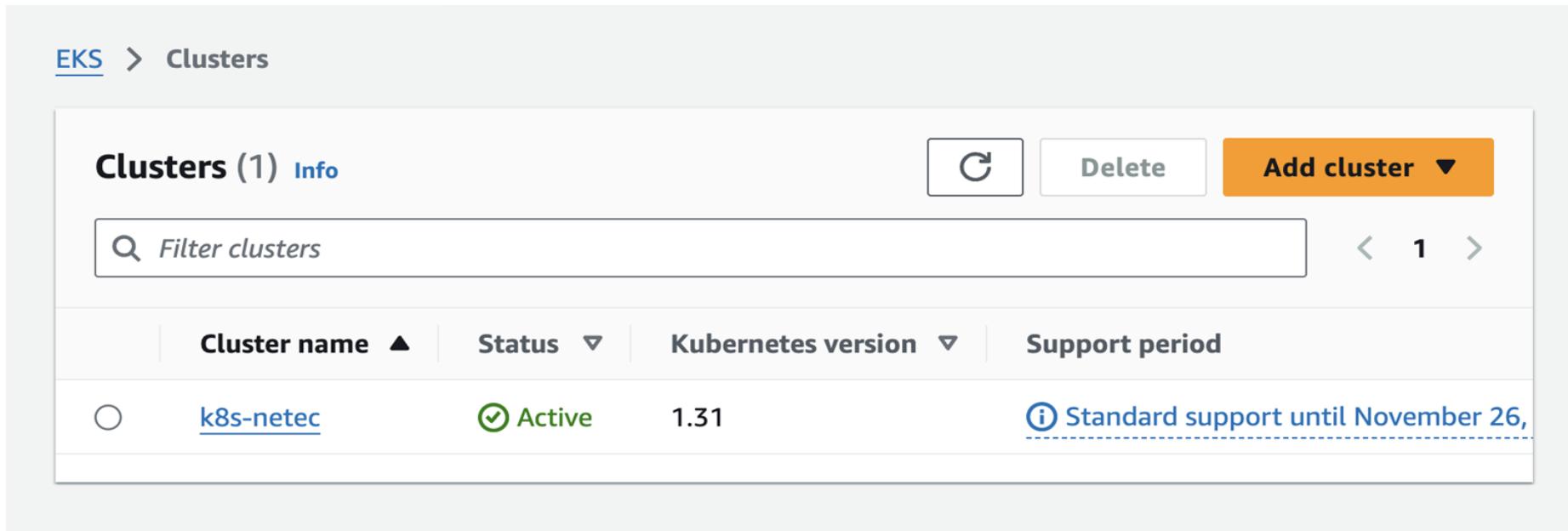
The screenshot shows the AWS EKS console interface. On the left, a sidebar lists 'Clusters', 'Amazon EKS Anywhere', 'Related services' (Amazon ECR, AWS Batch), and 'Console settings'. The main area displays a cluster named 'k8s-netec'. A blue banner at the top indicates that add-ons have been successfully added. The 'Cluster info' section shows the cluster is currently 'Creating'. The 'Details' section provides more information about the cluster's configuration and metadata.

Cluster info
Status: Creating
Kubernetes version: 1.31
Support period: Standard support until November 26, 2025
Provider: EKS

Details		
API server endpoint	OpenID Connect provider URL	Created
-	-	a few seconds ago
Certificate authority	Cluster IAM role ARN	Cluster ARN
(empty)	arn:aws:iam::43846516432:role/roleEksCluster View in IAM	arn:aws:eks:us-east-1:43846516432:cluster/k8s-netec
		Platform version: Info

Validación clúster

El status del clúster debe estar en **active** para poder avanzar con las siguientes configuraciones.

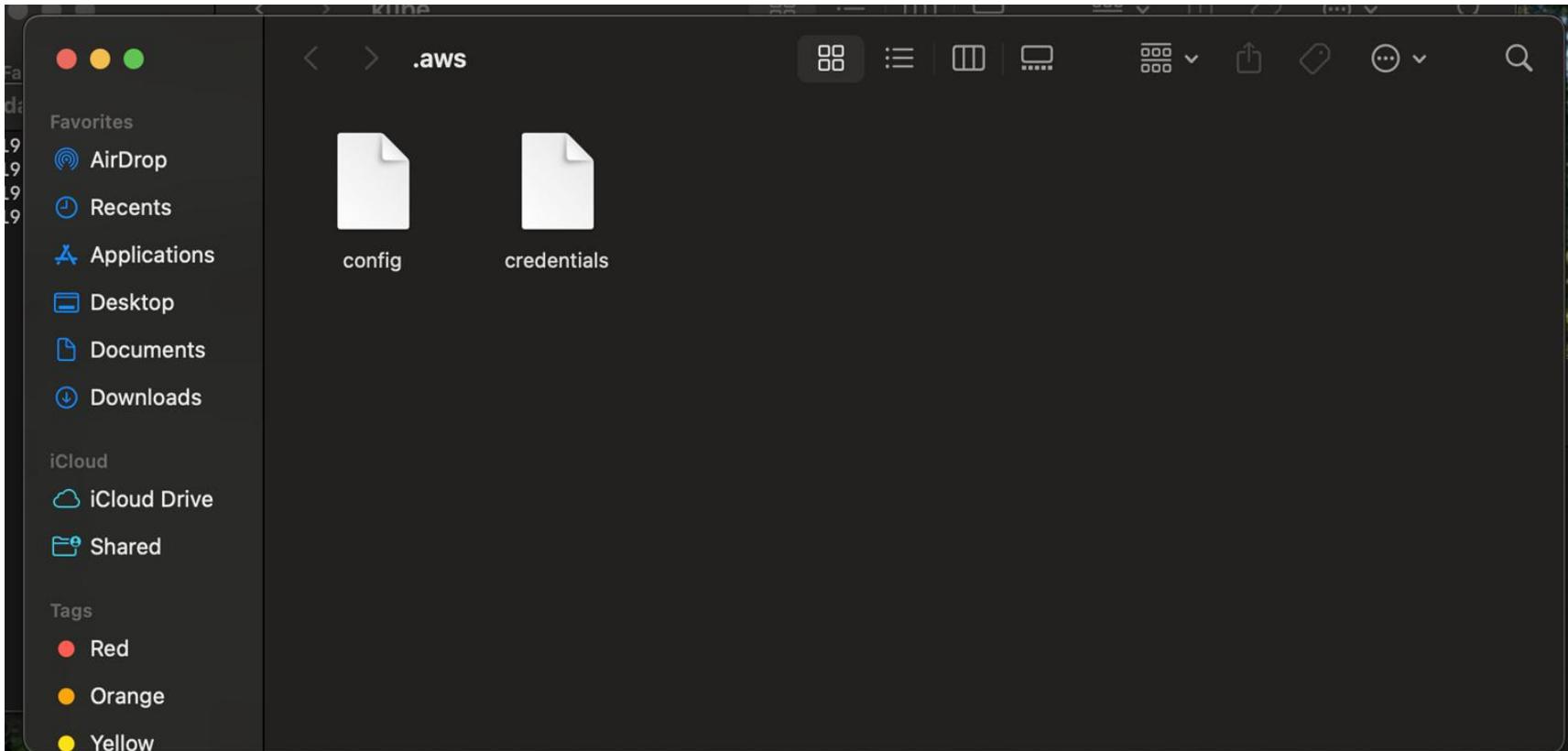


The screenshot shows the AWS EKS Clusters management interface. At the top left, it says "EKS > Clusters". Below that, there's a header with "Clusters (1) Info" and three buttons: a refresh icon, "Delete", and an orange "Add cluster" button with a dropdown arrow. There's also a search bar with the placeholder "Filter clusters" and a page navigation area with arrows and the number "1". The main table has four columns: "Cluster name" (with an upward arrow), "Status" (with a downward arrow), "Kubernetes version" (with a downward arrow), and "Support period". The first row shows a cluster named "k8s-netec" with an "Active" status (indicated by a green checkmark), version "1.31", and support ending on "November 26".

Cluster name ▲	Status ▼	Kubernetes version ▼	Support period
k8s-netec	Active	1.31	Standard support until November 26,

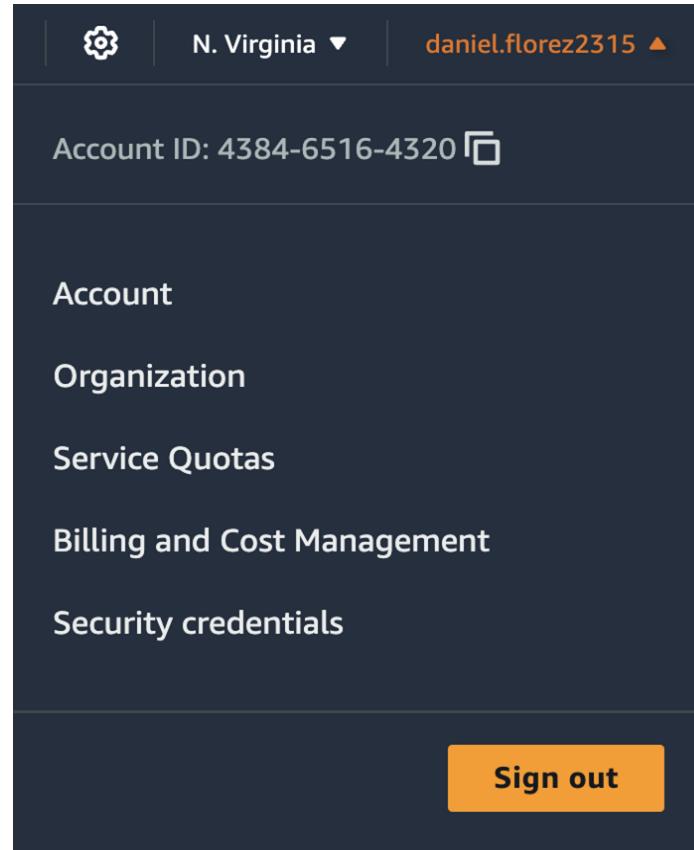
Configurando consola local

Buscamos la carpeta llamada `./aws/credentials` para configurar la conexión con AWS.



Configurando consola local

Abrimos el menú lateral derecho e ingresamos al Security credentials.



Clave de acceso

Creamos una nueva clave de acceso.

Access keys (0)

Use access keys to send programmatic calls to AWS from the AWS CLI, AWS Tools for PowerShell, AWS SDKs, or direct AWS API calls. You can have a maximum of two access keys (active or inactive) at a time. [Learn more](#)

Access key ID	Created on	Access key last used	Region last used	Service last used
No access keys				

As a best practice, avoid using long-term credentials like access keys. Instead, use tools which provide short term credentials.[Learn more](#)

[Create access key](#)

Clave de acceso

Aceptamos los términos y condiciones.

Retrieve access key [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
<input type="checkbox"/> AKIAWMFUPOQQF6S7UEVX	<input type="checkbox"/> ***** Show

Access key best practices

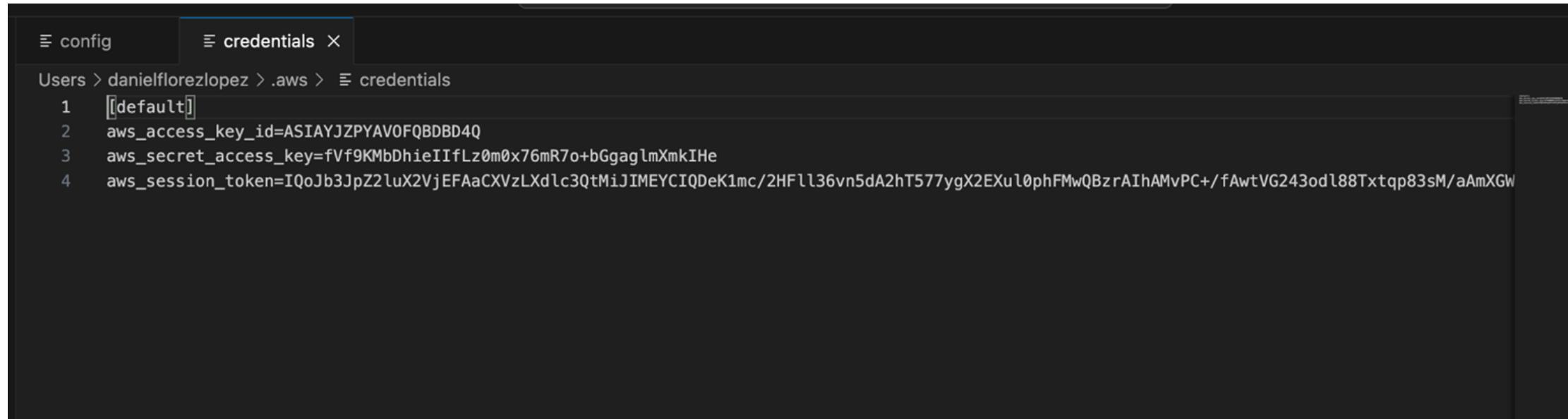
- Never store your access key in plain text, in a code repository, or in code.
- Disable or delete access key when no longer needed.
- Enable least-privilege permissions.
- Rotate access keys regularly.

For more details about managing access keys, see the [best practices for managing AWS access keys](#).

[Download .csv file](#) [Done](#)

Configurando consola local

Las agregamos al archivo credentials.



The screenshot shows a terminal window with a dark background. At the top, there are two tabs: "config" and "credentials". The "credentials" tab is active and highlighted with a blue border. Below the tabs, the path "Users > danielflorezlopez > .aws > credentials" is displayed. The main area of the terminal contains four lines of text, each starting with a number (1, 2, 3, 4) followed by an AWS credential key and its corresponding value. The "default" key is bolded.

```
1 [default]
2 aws_access_key_id=ASIAYJZPYAV0FQBDDBD4Q
3 aws_secret_access_key=fVf9KMbDhieIIfLz0m0x76mR7o+bGgaglmXmkIHe
4 aws_session_token=IQoJb3JpZ2luX2VjEFAaCXVzLXdIc3QtMiJIMEYCIQDeK1mc/2HFll36vn5dA2hT577ygX2Exul0phFMwQBzrAIhAMvPC+/fAwVG243odl88Txtqp83sM/aAmXGw
```

Configurando consola local

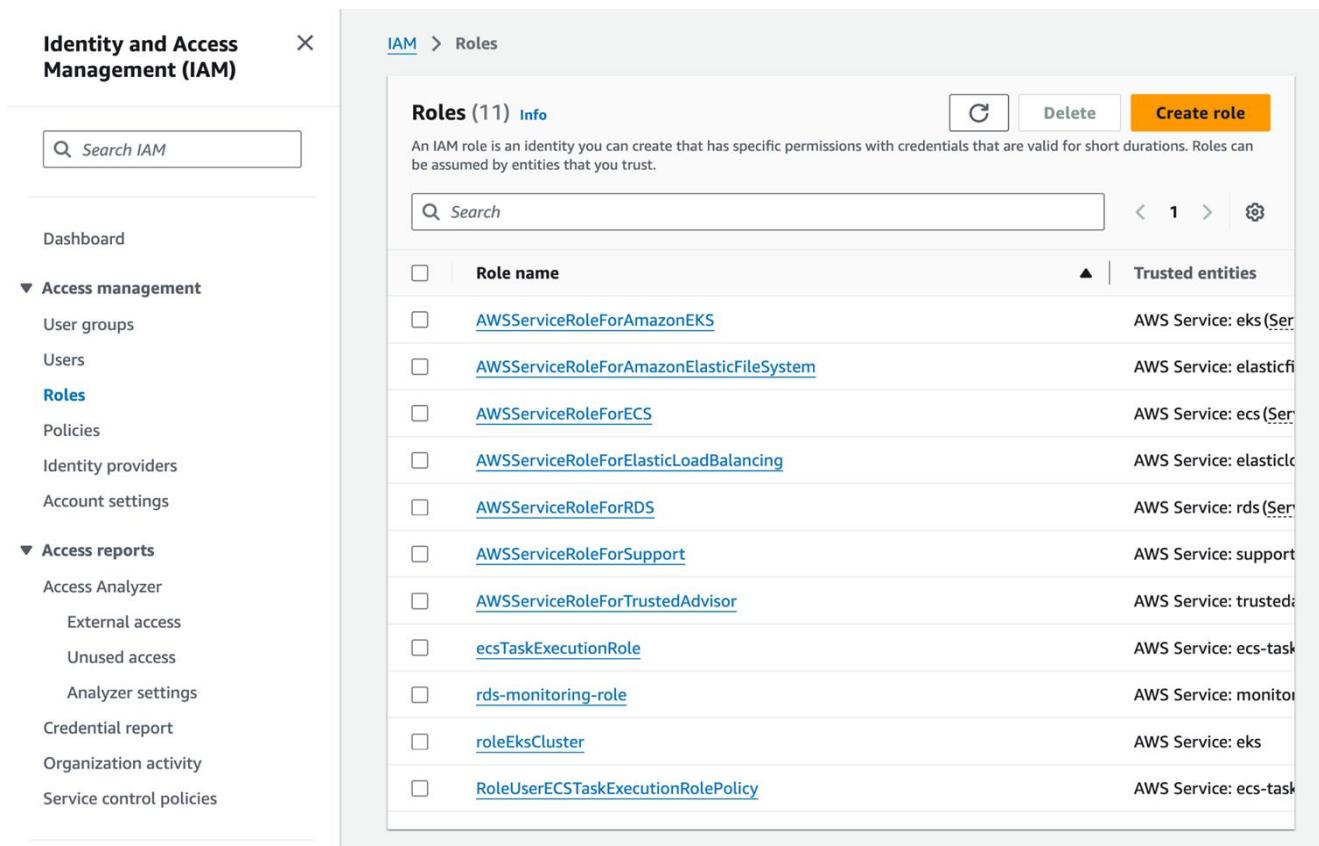
Ejecutamos el comando para poder conectarnos con el clúster desde nuestra maquina local.

- aws eks --region us-east-1 update-kubeconfig --name k8s-netec

```
(base) danielflorezlopez@192 ~ % aws eks --region us-east-1 update-kubeconfig --name k8s-netec
Added new context arn:aws:eks:us-east-1:438465164320:cluster/k8s-netec to /Users/danielflorezlopez/.kube/config
(base) danielflorezlopez@192 ~ %
```

Crear role

Ingresamos a la opción de Create rol.



The screenshot shows the AWS Identity and Access Management (IAM) service interface. On the left, there's a sidebar with navigation options like Dashboard, User groups, Users, Roles (which is selected and highlighted in blue), Policies, Identity providers, and Account settings. Below that is another section for Access reports with options like Access Analyzer, External access, Unused access, Analyzer settings, Credential report, Organization activity, and Service control policies. The main content area is titled 'Roles (11)' and contains a table with 11 rows, each representing a different IAM role. The columns in the table are 'Role name' (containing links like 'AWSServiceRoleForAmazonEKS', 'AWSServiceRoleForAmazonElasticFileSystem', etc.), 'Trusted entities' (containing links like 'AWS Service: eks (Service Role)', 'AWS Service: elasticfs (Service Role)', etc.), and a small checkbox icon. At the top right of the main content area, there are three buttons: 'Create role' (which is orange and highlighted), 'Delete', and a 'Create role' button again. A search bar and a pagination indicator ('1') are also present at the top of the table.

Role name	Trusted entities
AWSServiceRoleForAmazonEKS	AWS Service: eks (Service Role)
AWSServiceRoleForAmazonElasticFileSystem	AWS Service: elasticfs (Service Role)
AWSServiceRoleForECS	AWS Service: ecs (Service Role)
AWSServiceRoleForElasticLoadBalancing	AWS Service: elasticloadbalancing (Service Role)
AWSServiceRoleForRDS	AWS Service: rds (Service Role)
AWSServiceRoleForSupport	AWS Service: support (Service Role)
AWSServiceRoleForTrustedAdvisor	AWS Service: trustedadvisor (Service Role)
ecsTaskExecutionRole	AWS Service: ecs-taskexecutionrole (Service Role)
rds-monitoring-role	AWS Service: monitoring-role (Service Role)
roleEksCluster	AWS Service: eks (Service Role)
RoleUserECSTaskExecutionRolePolicy	AWS Service: ecs-taskexecutionrolepolicy (Service Role)

Crear role

En **Service or use case**, seleccionamos EC2 y luego siguiente.

The screenshot shows two steps of a wizard:

Trusted entity type

- AWS service**
Allow AWS services like EC2, Lambda, or others to perform actions in this account.
- AWS account**
Allow entities in other AWS accounts belonging to you or a 3rd party to perform actions in this account.
- Web identity**
Allows users federated by the specified external web identity provider to assume this role to perform actions in this account.
- SAML 2.0 federation**
Allow users federated with SAML 2.0 from a corporate directory to perform actions in this account.
- Custom trust policy**
Create a custom trust policy to enable others to perform actions in this account.

Use case

Allow an AWS service like EC2, Lambda, or others to perform actions in this account.

Service or use case

EC2

Choose a use case for the specified service.

Use case

- EC2**
Allows EC2 instances to call AWS services on your behalf.
- EC2 Role for AWS Systems Manager**
Allows EC2 instances to call AWS services like CloudWatch and Systems Manager on your behalf.

Crear role

Buscamos el role AmazonEKSWorkerNodePolicy y damos next.

Add permissions Info

Permissions policies (1/962) Info

Choose one or more policies to attach to your new role.

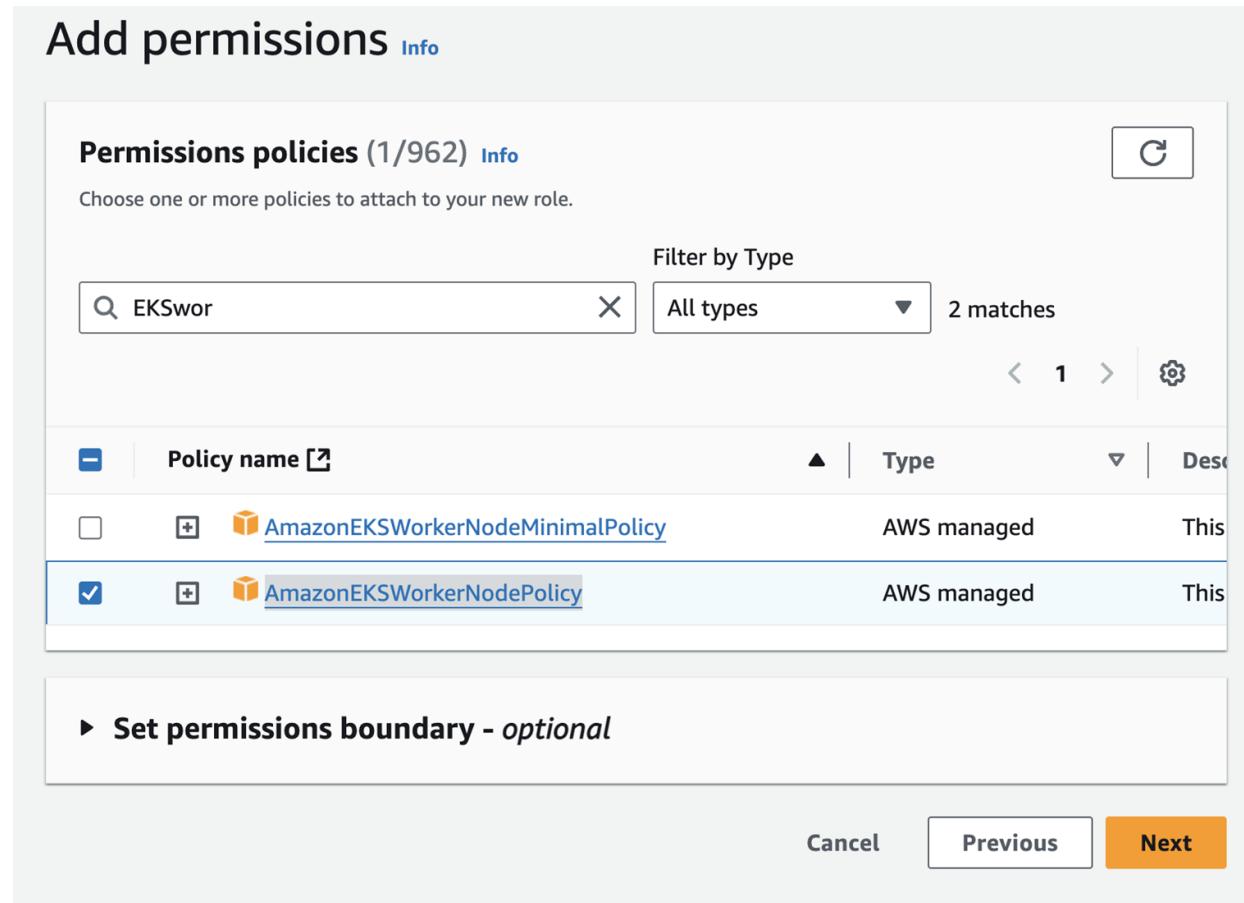
Filter by Type

EKSwor All types 2 matches

Policy name	Type	Description
<input type="checkbox"/> AmazonEKSWorkerNodeMinimalPolicy	AWS managed	This policy provides the minimum set of permissions required for an EKS worker node to interact with AWS services.
<input checked="" type="checkbox"/> AmazonEKSWorkerNodePolicy	AWS managed	This policy provides the minimum set of permissions required for an EKS worker node to interact with AWS services, including the ability to access AWS Lambda functions.

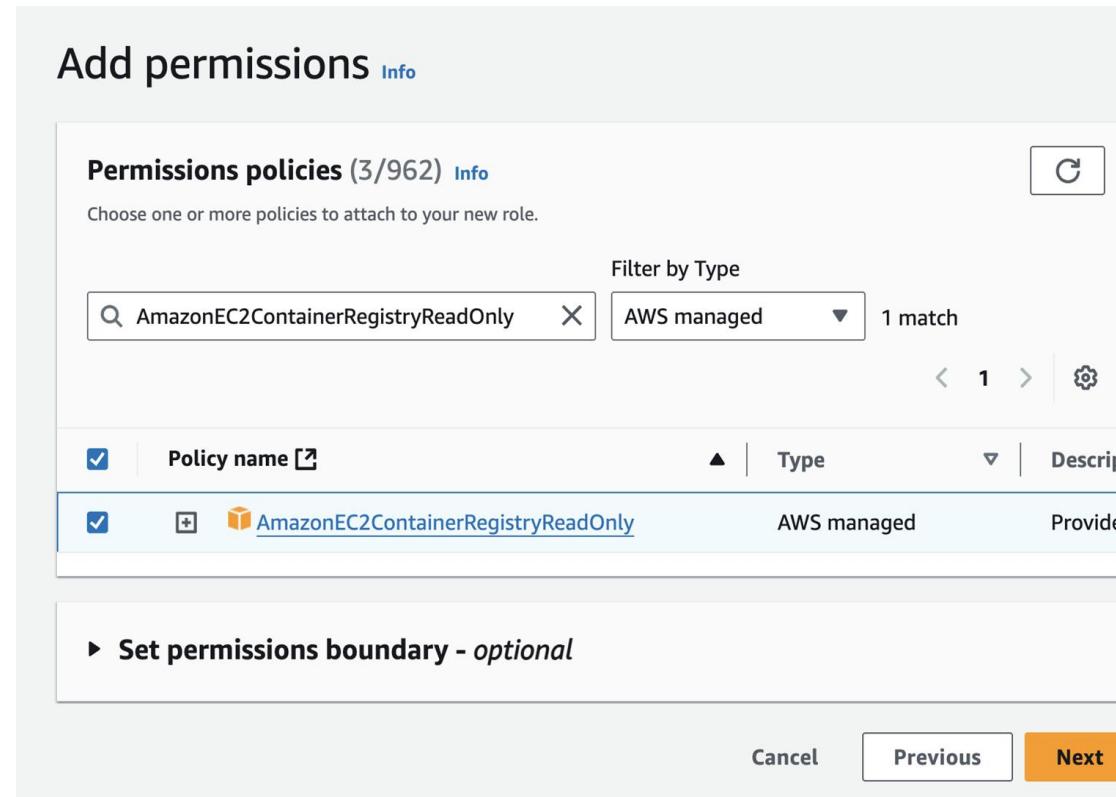
Set permissions boundary - optional

Cancel



Crear Role

Buscamos el role “AmazonEKSWorkerNodePolicy”, “AmazonEKS_CNI_Policy”, “AmazonEC2ContainerRegistryReadOnly” y siguiente



Crear role

Le damos un nombre al rol y validamos los roles seleccionados.

Name, review, and create

Role details

Role name
Enter a meaningful name to identify this role.
Maximum 64 characters. Use alphanumeric and '+,-,@-_ characters.

Description
Add a short explanation for this role.
Maximum 1000 characters. Use letters (A-Z and a-z), numbers (0-9), tabs, new lines, or any of the following characters: _+=.,@-/\[\]!#\$%^&*();'''`

Step 1: Select trusted entities

Step 2: Add permissions

Permissions policy summary

Policy name	Type	Attached as
AmazonEC2ContainerRegistryReadOnly	AWS managed	Permissions policy
AmazonEKS_CNI_Policy	AWS managed	Permissions policy
AmazonEKSWorkerNodePolicy	AWS managed	Permissions policy

Agregando workets

Ingresamos al apartado de Compute en el clúster de EKS para que el EKS pueda balancear los Pods correctamente.

The screenshot shows the AWS EKS Compute tab interface. At the top, there are tabs: Overview, Resources, Compute (which is selected), Networking, Add-ons (with a notification count of 1), Access, and Observability. Below the tabs, there are two main sections: 'Nodes (0) Info' and 'Node groups (0) Info'.
Nodes (0) Info: This section includes a search bar labeled 'Filter Nodes by property or value' and a table header with columns: Node name, Instance type, Node group, Created, and Status. A message below the table states 'No Nodes' and explains 'This cluster does not have any Nodes, or you don't have permission to view them.'
Node groups (0) Info: This section includes buttons for Edit, Delete, and Add node group. It has a table header with columns: Group name, Desired size, AMI release version, Launch template, and Status. A message below the table states 'No node groups' and explains 'This cluster does not have any node groups. Nodes that are not part of an Amazon EKS managed node group are not shown in the AWS console.' A large 'Add node group' button is centered at the bottom of this section.

Configure node group

Damos un nombre al Node group y agregamos el rol que se creó previamente.

Como consecuente, dejamos la configuración por defecto y siguiente.

Configure node group Info

A node group is a group of EC2 instances that supply compute capacity to your Amazon EKS cluster. You can add multiple node groups to your cluster.

Node group configuration

These properties cannot be changed after the node group is created.

Name
Assign a unique name for this node group.
The node group name should begin with letter or digit and can have any of the following characters: the set of Unicode letters, digits, hyphens and underscores. Maximum length of 63.

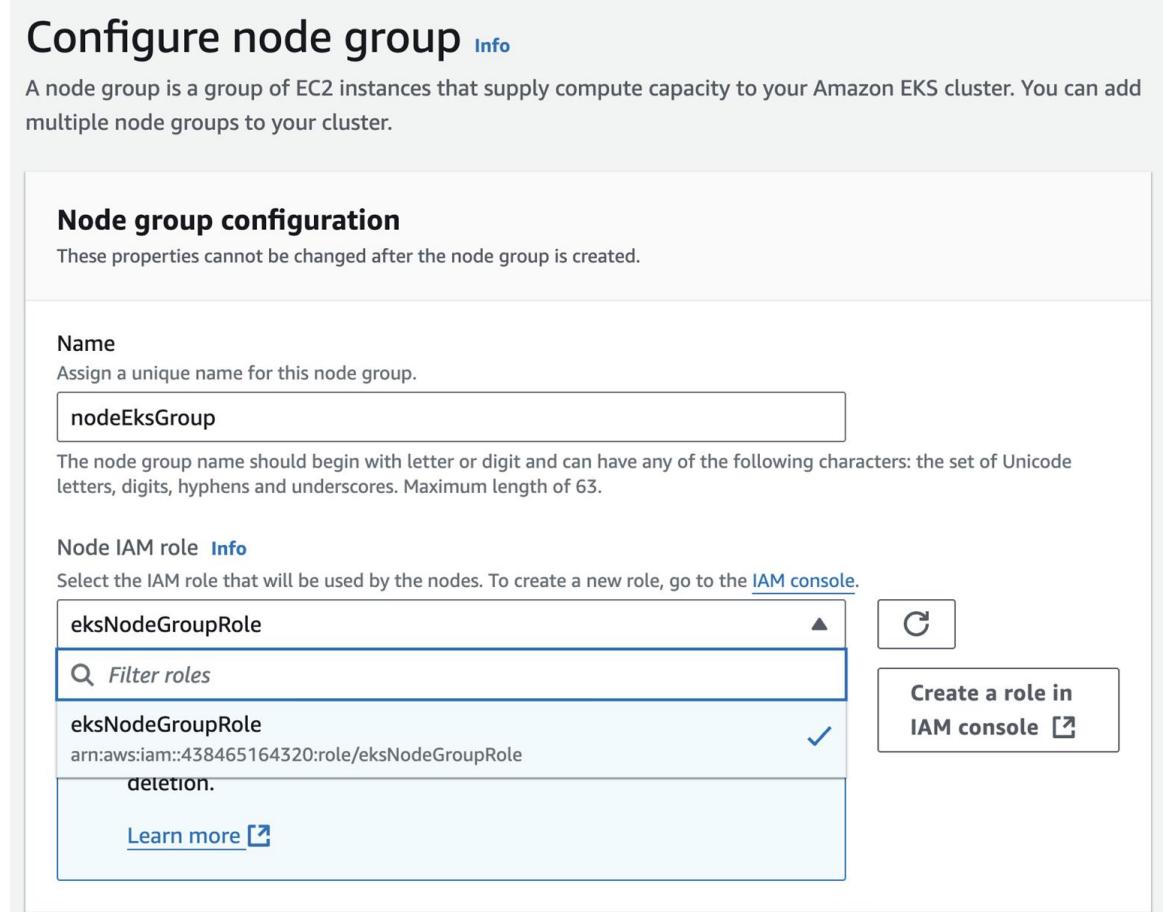
Node IAM role Info
Select the IAM role that will be used by the nodes. To create a new role, go to the [IAM console](#).

C

Create a role in IAM console

eksNodeGroupRole
arn:aws:iam::438465164320:role/eksNodeGroupRole
detention.

[Learn more](#)



Set compute and scaling configuration

Seleccionamos la configuración de la maquina de EC2 que vamos a utilizar. Tener en cuenta el tipo de AMI ya que en este se van a desplegar los Pods con nuestra solución de backend.

Node group compute configuration
These properties cannot be changed after the node group is created.

AMI type [Info](#)
Select the EKS-optimized Amazon Machine Image for nodes.
Amazon Linux 2 Arm (AL2_ARM_64) ▾

Capacity type
Select the capacity purchase option for this node group.
On-Demand ▾

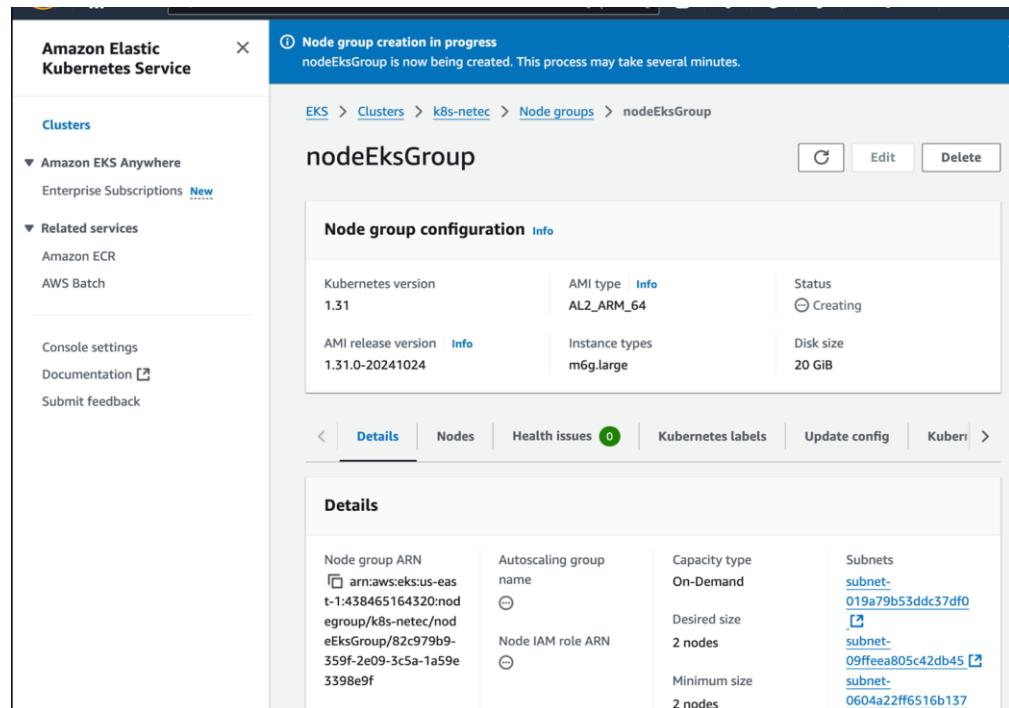
Instance types [Info](#)
Select instance types you prefer for this node group.
 X

m6g.large
vCPU: 2 vCPUs Memory: 8 GiB Network: Up to 10 Gigabit Max ENI: 3
Max IPs: 30

Disk size
Select the size of the attached EBS volume for each node.
20 GiB

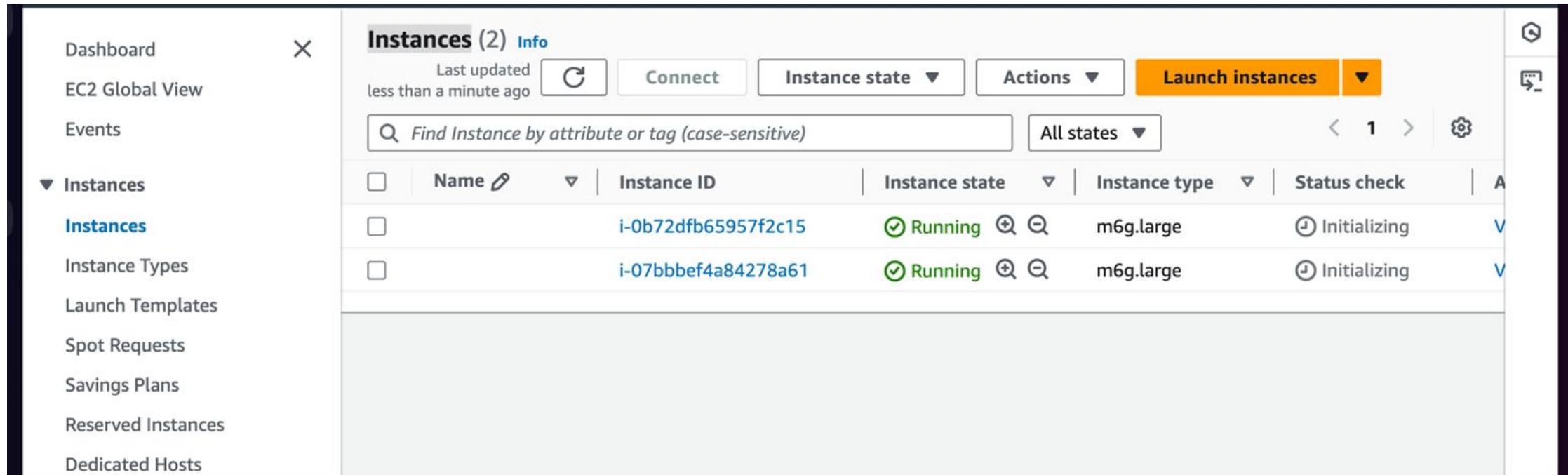
Create nodes

Dejamos por default las demás configuraciones y damos en create.



EC2 Instances

Se crean las instancias de EC2 automáticamente.



The screenshot shows the AWS EC2 Instances page with the following details:

- Dashboard**, **EC2 Global View**, **Events**, and **Instances** are listed in the sidebar.
- The main title is **Instances (2)**.
- Filter: **Last updated less than a minute ago**, **C**, **Connect**, **Instance state** (dropdown), **Actions** (dropdown), **Launch instances** (button).
- Search bar: **Find Instance by attribute or tag (case-sensitive)**.
- Instance List:

<input type="checkbox"/>	Name	Instance ID	Instance state	Instance type	Status check	A
<input type="checkbox"/>		i-0b72dfb65957f2c15	Running	m6g.large	Initializing	V
<input type="checkbox"/>		i-07bbbef4a84278a61	Running	m6g.large	Initializing	V

Nodes

Las máquinas se registran en EKS node.

Nodes (2) Info					
<input type="text"/> Filter Nodes by property or value < 1 >					
Node name	▲	Instance type	▼	Node group	▼
ip-192-168-183-56.ec2.internal		m6g.large		nodeEksGroup	Created <input type="checkbox"/> a minute ago (✓) Read y
ip-192-168-245-182.ec2.internal		m6g.large		nodeEksGroup	Created <input type="checkbox"/> a minute ago (✓) Read y

Configurando proyecto

Vamos a otorgar permisos de administrador a una cuenta de servicio específica en un clúster de Kubernetes por medio de este comando:

- `kubectl create clústerrolebinding admin --clústerrole=clúster-admin --serviceaccount=default:default`

```
(base) danielflorezlopez@192 ms-user % kubectl create clusterrolebinding admin --clusterrole=cluster-admin --serviceaccount=default:default
clusterrolebinding.rbac.authorization.k8s.io/admin created
(base) danielflorezlopez@192 ms-user %
```

Dockerizando backend

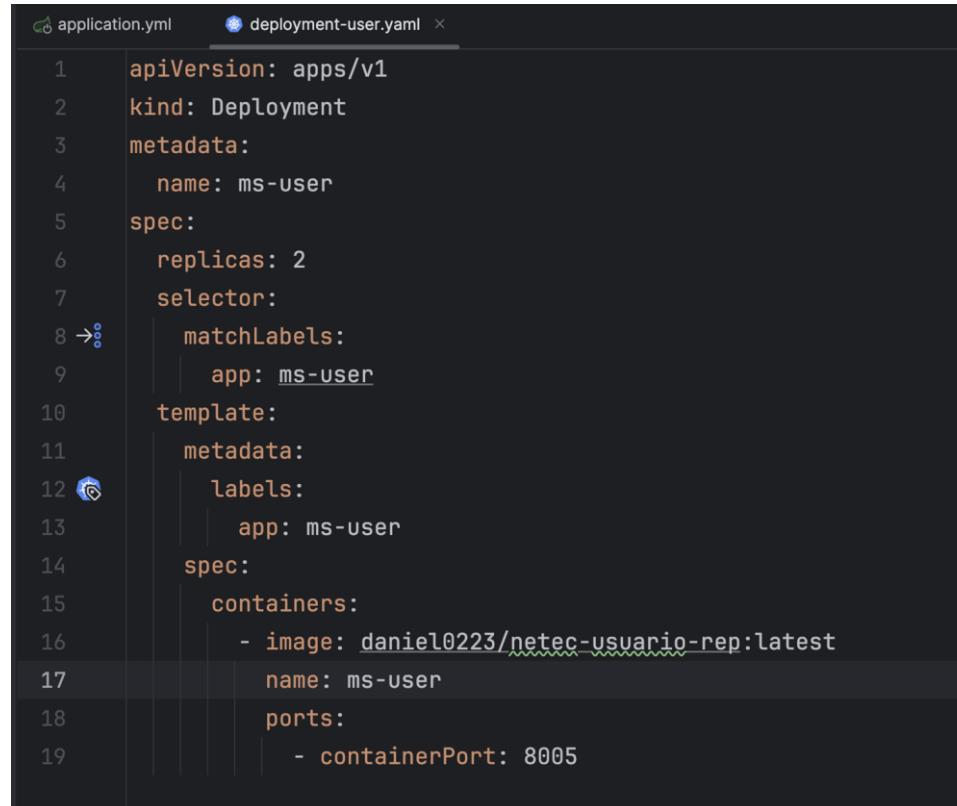
Antes de crear los Pod es necesario que nuestro servicio de backend usuarios este en Docker Hub por tal razón vamos a crear la imagen y subirla al Docker Hub.

- docker build -t netec-usuario-rep:V1 .
- docker tag netec-usuario-rep:V1 daniel0223/netec-usuario-rep
- docker push daniel0223/netec-usuario-rep

```
(base) danielflorezlopez@192 ms-user % docker tag netec-usuario-rep:V1 daniel0223/netec-usuario-rep
(base) danielflorezlopez@192 ms-user % docker push daniel0223/netec-usuario-rep
Using default tag: latest
The push refers to repository [docker.io/daniel0223/netec-usuario-rep]
ff9269fd5b8b: Pushing [=====>] 7.21MB/53.94MB
9faf519d0568: Layer already exists
1e4376b6d172: Layer already exists
c82e5bf37b8a: Layer already exists
2f263e87cb11: Layer already exists
f941f90e71a8: Layer already exists
```

Creación deployment-user.yaml

Para poder desplegar el microservicio de usuarios en EKS, es necesario tener configurado el **deployment-user.yaml**. Para esto, vamos a crear el archivo y configurarlo.



```
application.yml deployment-user.yaml
1 apiVersion: apps/v1
2 kind: Deployment
3 metadata:
4   name: ms-user
5 spec:
6   replicas: 2
7   selector:
8     matchLabels:
9       app: ms-user
10  template:
11    metadata:
12      labels:
13        app: ms-user
14    spec:
15      containers:
16        - image: daniel0223/netec-usuario-rep:latest
17          name: ms-user
18        ports:
19          - containerPort: 8005
```

DeployUser

Para realizar el despliegue del microservicio de user debemos realizar el deploy con kubectl.

- kubectl apply -f deployment-user.yaml

```
(base) danielflorezlopez@192 ms-user % kubectl apply -f deployment-user.yaml
deployment.apps/ms-user created
```

```
(base) danielflorezlopez@192 ms-user % kubectl get pods
NAME                  READY   STATUS    RESTARTS   AGE
ms-user-66d99547b5-gx9xj  1/1     Running   0          41s
ms-user-66d99547b5-hw4bp  1/1     Running   0          41s
```

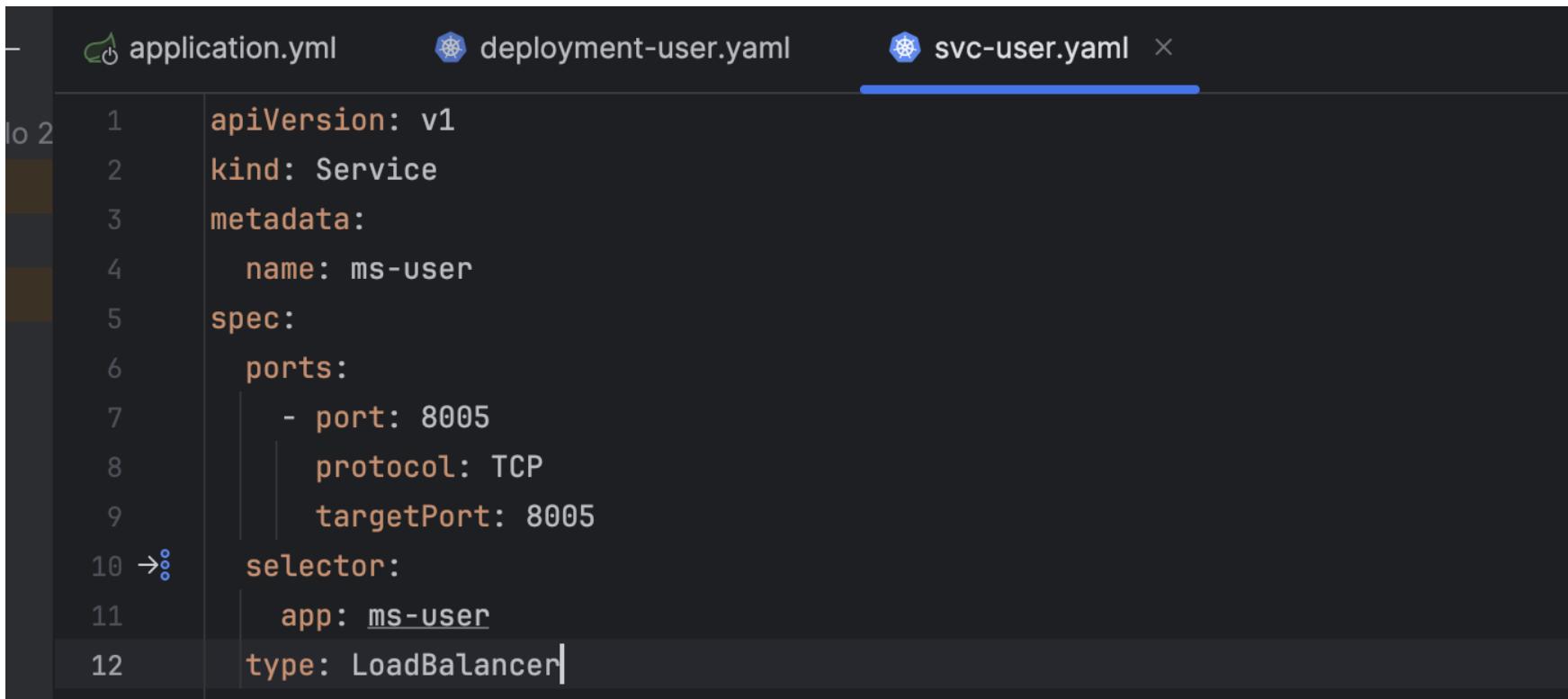
Validación despliegue

Para validar si el servicio fue correctamente desplegado listamos los Pods.

```
(base) danielflorezlopez@192 ms-user % kubectl get pods  
NAME                  READY   STATUS    RESTARTS   AGE  
ms-user-66d99547b5-gx9xj  1/1     Running   0          41s  
ms-user-66d99547b5-hw4bp  1/1     Running   0          41s
```

Creando service-user

Para poder acceder o realizar consultas al servicio es necesario crear el svc-user.yaml



```
apiVersion: v1
kind: Service
metadata:
  name: ms-user
spec:
  ports:
    - port: 8005
      protocol: TCP
      targetPort: 8005
  selector:
    app: ms-user
  type: LoadBalancer
```

Desplegando service-user

Una vez configurado procedemos a crear el service con el comando:

- `kubectl apply -f svc-user.yaml`

```
(base) danielflorezlopez@192 ms-user % kubectl apply -f svc-user.yaml  
service/ms-user created
```

```
(base) danielflorezlopez@192 ms-user % kubectl apply -f deployment-user.yaml  
deployment.apps/ms-user created  
(base) danielflorezlopez@192 ms-user % kubectl apply -f svc-user.yaml  
service/ms-user created
```

Validamos despliegue

Para obtener la ruta del load balancing usamos el comando.

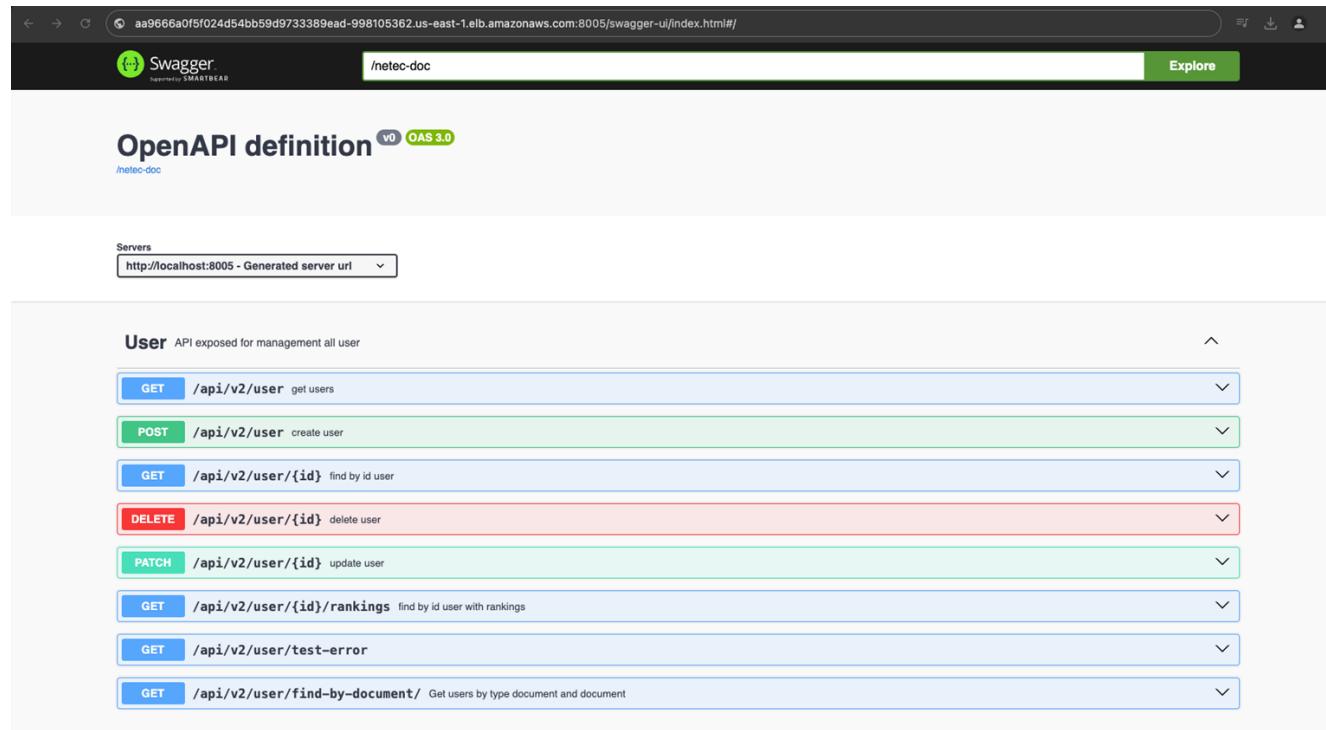
- kubectl get services

```
(base) danielflorezlopez@192 ms-user % kubectl get services

NAME      TYPE        CLUSTER-IP      EXTERNAL-IP
kubernetes  ClusterIP   10.100.0.1    <none>
ms-user    LoadBalancer  10.100.96.151 aa9666a0f5f024d54bb59d9733389ead-998105362.us-east-1.elb.amazonaws.com  8005:30388/TCP  8m
(base) danielflorezlopez@192 ms-user % 
```

Validamos despliegue

Ingresamos a la URL que se obtuvo del load balancing.



2.2. Eliminando clúster EKS

Para evitar que la facturación aumente, es necesario eliminar los servicios que se fueron levantando.

En esta sección vamos a eliminar toda la infraestructura adyacente que se desplegó.

Delete Node Group

Como primer paso, vamos a eliminar el nodeEksGroup con el botón Delete.

The screenshot shows the AWS Lambda Compute service interface. The top navigation bar includes Overview, Resources, Compute (selected), Networking, Add-ons (with 1 notification), Access, and Observability. The Compute section has tabs for Functions, Triggers, and Events. Below the tabs, there are two main sections: "Nodes (2) Info" and "Node groups (1) Info".

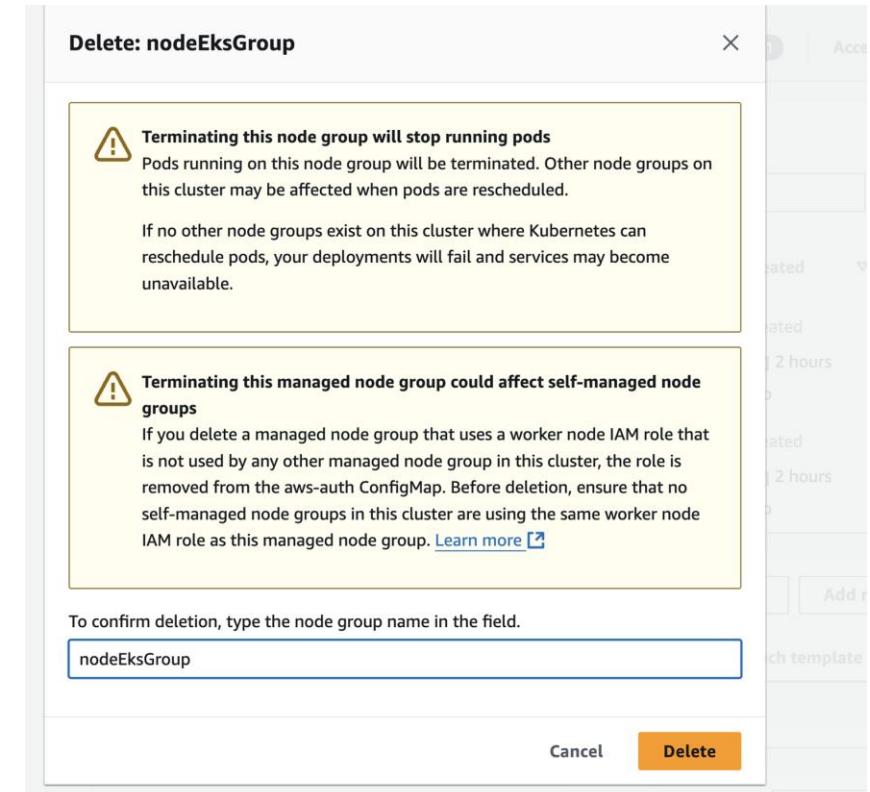
Nodes (2) Info: A table with columns: Node name, Instance type, Node group, Created, and Status. It lists two nodes:

Node name	Instance type	Node group	Created	Status
ip-192-168-183-56.ec2.internal	m6g.large	nodeEksGroup	Created 2 hours ago	Read y
ip-192-168-245-182.ec2.internal	m6g.large	nodeEksGroup	Created 2 hours ago	Read y

Node groups (1) Info: A table with columns: Group name, Desired size, AMI release version, Launch template, and Status. It lists one node group:

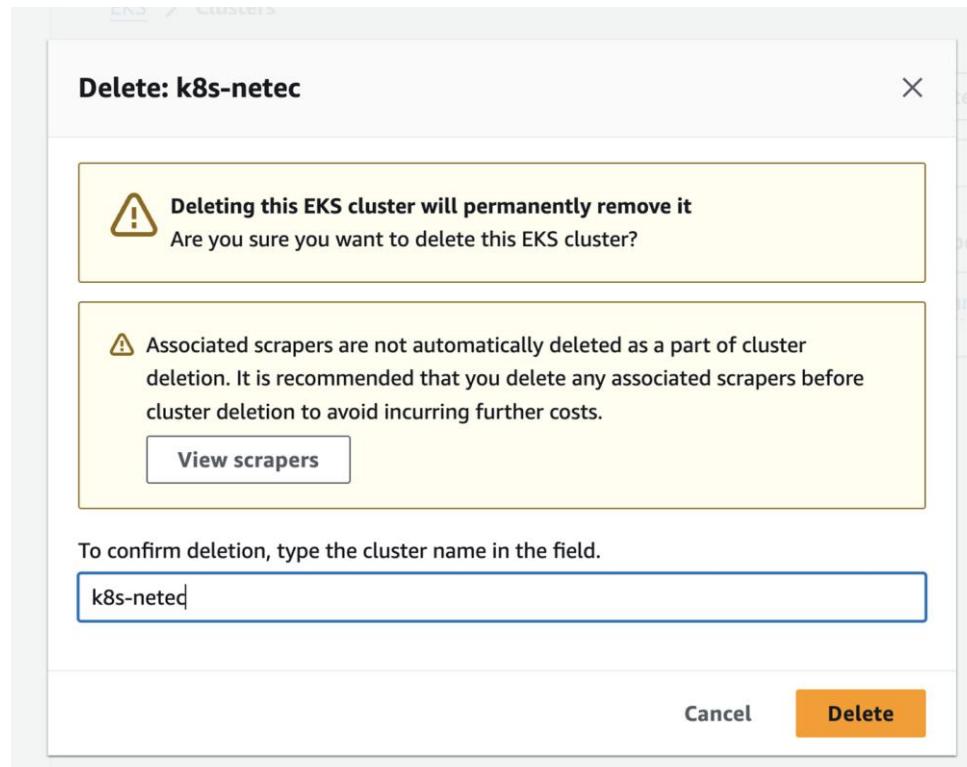
Group name	Desired size	AMI release version	Launch template	Status
nodeEksGroup	2	1.31.0-20241024	-	Active

Buttons for Edit, Delete, and Add node group are visible at the bottom of the Node groups section.



Eliminamos el clúster en EKS

Ingresamos a EKS, buscamos el clúster y seleccionamos Delete.



Clusters (1) <small>Info</small>			
<input type="text"/> Filter clusters			
Cluster name	Status	Kubernetes version	Support period
k8s-netec	Active	1.31	Standard support until November 26,

2.3. Kubernetes Ingress

Kubernetes Ingress es una tools que permite gestionar el acceso a los servicios de cada clúster. El Ingress funciona como un punto de entrada centralizado. En vez de exponer cada servicio individualmente, Ingress permite controlar el flujo de tráfico, lo que facilita la implementación de características como balanceo de carga, gestión de certificados SSL y reescritura de URL, simultáneamente.

2.4. Path-Based routing

El Path-Based routing es una forma sugerida y eficiente de dirigir las solicitudes a diferentes servicios según la ruta específica de la URL, permitiendo establecer reglas claras en Ingress que determinan a dónde se envían las solicitudes.

2.5. Host-based routing

El Host-based routing es una característica singular de Kubernetes Ingress que permite gestionar varios dominios con un solo recurso.

Al identificar el nombre del host en las solicitudes, se puede dirigir el tráfico a servicios específicos.

Resumen del capítulo

- Realizar el despliegue de un microservicio en AWS EKS.
- Comprender la diferencia entre EC2 y Fargate.
- Creación de nodos autogestionados.



Práctica 2. Despliegue en Kubernetes

Objetivo:

- Creación de manifiestos YAML para desplegar en k8s.
- Configuración proyecto backend.
- Desplegar servicio email en un minikube de Kubernetes.

Planteamiento e instrucciones:

Realiza un seguimiento de los pasos indicados en la *Guía de Laboratorios* para llevar a cabo la tarea correspondiente en el siguiente enlace: https://netec-mx.github.io/DOCK_KUB_AVA/



Tiempo para esta actividad:
45 minutos.

Resultado esperado

```
(base) danielflorezlopez@192 ms-user % kubectl get services
```

NAME	TYPE	CLUSTER-IP	EXTERNAL-IP	PORT(S)	AGE
kubernetes	ClusterIP	10.100.0.1	<none>	443/TCP	171m
ms-email	LoadBalancer	10.100.72.229	a3f3516bc1c544790a9e3184614c4ce8-133122345.us-east-1.elb.amazonaws.com	8081:30430/TCP	9s

```
(base) danielflorezlopez@192 ms-user % kubectl get pods
```

NAME	READY	STATUS	RESTARTS	AGE
ms-email-7c87f9d8fc-6rwww	1/1	Running	0	54s

Referencias Bibliográficas

- [Arquitectura de Amazon EKS - Amazon EKS](#)
- [Aplicación de escalado - AWS Application Auto Scaling - AWS](#)
- [Computación: tipos de instancias de Amazon EC2 – AWS](#)



Objetivos:

- Comprender que es Oauth2
- Seguridad en los microservicios
- Despliegue en kubectl
- Configuración de manifiesto en K8s

Capítulo 3

Kubernetes: Security JWT con OAuth 2.1

3.1. Creando microservicios

Spring Authorization Server

(OAuth 2.1)

En la arquitectura de microservicios es importante crear un microservicio dedicado 100% a la seguridad ya que en el principio básico de los microservicios cada micro debe tener una única responsabilidad.

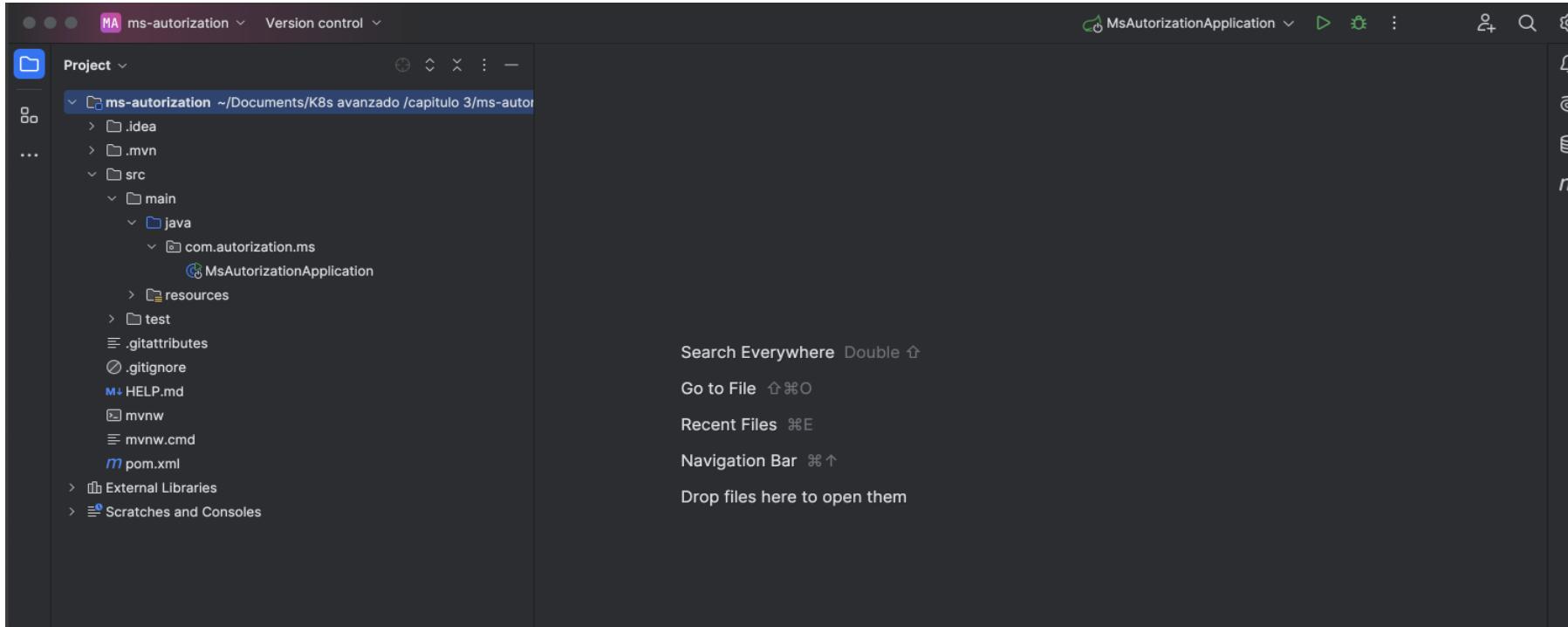
Creación microservicio Auth

Para crear el servicio de seguridad ingresamos al siguiente link y especificamos las configuraciones <https://start.spring.io/>

The screenshot shows the Spring Initializr web interface. On the left, the 'Project' section is set to Maven, Java 17, and Spring Boot 3.2.11. Project Metadata includes Group (com.autorization.ms), Artifact (ms-authorization), Name (ms-authorization), Description (ms-authorization), Package name (com.autorization.ms), and Packaging (Jar). In the center, the 'Dependencies' section has 'Spring Security' selected under the 'SECURITY' tab. This dependency is described as a highly customizable authentication and access-control framework for Spring applications. Below it, 'Spring Web' is selected under the 'WEB' tab, described as building web, including RESTful, applications using Spring MVC. It uses Apache Tomcat as the default embedded container. At the bottom, there are 'GENERATE' and 'SHARE...' buttons.

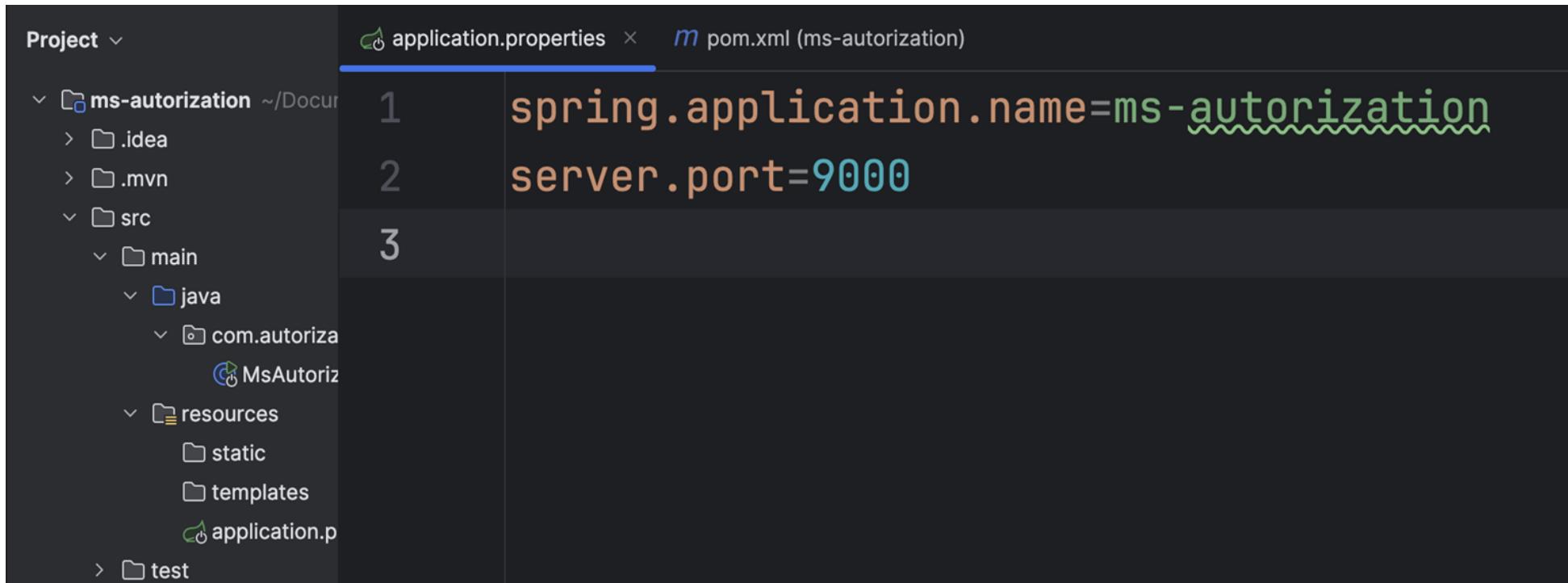
Abrir proyecto

Con ayuda de IntelliJ IDE abrimos el proyecto que hemos descomprimido y ubicado en una carpeta específica.



Configurar el application.properties

Con ayuda de IntelliJ IDE abrimos el proyecto que hemos descomprimido y ubicado en una carpeta específica.

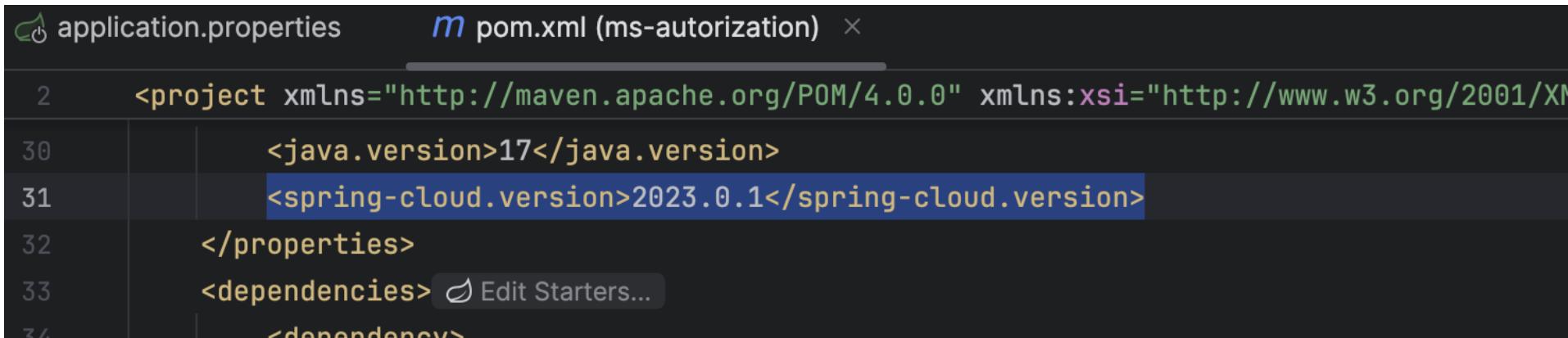


```
spring.application.name=ms-autorization
server.port=9000
```

Agregando configuración Spring Cloud

En el pom.xml , agregamos la etiqueta.

- <spring-cloud.version>2023.0.1</spring-cloud.version>



```
application.properties      m pom.xml (ms-autorization) ×

2   <project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
30  |     <java.version>17</java.version>
31  |     <spring-cloud.version>2023.0.1</spring-cloud.version>
32  |   </properties>
33  |   <dependencies> ⚡ Edit Starters...
34  |     <dependency>
```

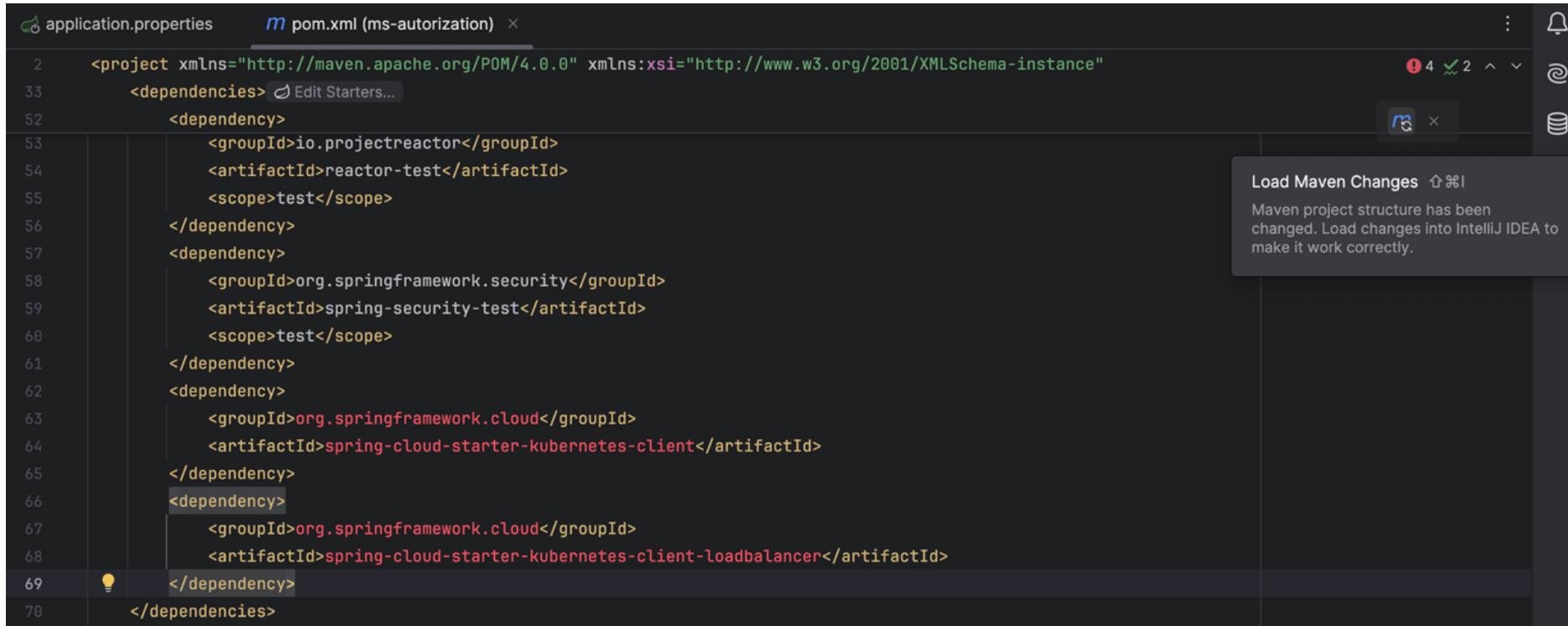
Ajustando el pom.xml

En el pom.xml , agregamos adicionalmente las dependencias de Kubernetes

```
<dependency>
    <groupId>org.springframework.cloud</groupId>
    <artifactId>spring-cloud-starter-kubernetes-client</artifactId>
</dependency>
<dependency>
    <groupId>org.springframework.cloud</groupId>
    <artifactId>spring-cloud-starter-kubernetes-client-loadbalancer</artifactId>
</dependency>
```

Ajustando el pom.xml

No olvidemos actualizar el Maven.



The screenshot shows the IntelliJ IDEA interface with two tabs open: "application.properties" and "pom.xml (ms-autorization)". The "pom.xml" tab is active, displaying XML code for a Maven project. The code includes several dependency blocks for Spring Reactor, Spring Security, and Spring Cloud Kubernetes Client. A tooltip on the right side of the interface reads: "Load Maven Changes ⌘⌥I" and "Maven project structure has been changed. Load changes into IntelliJ IDEA to make it work correctly." This indicates that there are changes in the Maven project structure that have not yet been loaded into the IDE.

```
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <dependencies>
    <dependency>
      <groupId>io.projectreactor</groupId>
      <artifactId>reactor-test</artifactId>
      <scope>test</scope>
    </dependency>
    <dependency>
      <groupId>org.springframework.security</groupId>
      <artifactId>spring-security-test</artifactId>
      <scope>test</scope>
    </dependency>
    <dependency>
      <groupId>org.springframework.cloud</groupId>
      <artifactId>spring-cloud-starter-kubernetes-client</artifactId>
    </dependency>
    <dependency>
      <groupId>org.springframework.cloud</groupId>
      <artifactId>spring-cloud-starter-kubernetes-client-loadbalancer</artifactId>
    </dependency>
  </dependencies>
```

Ajustando el pom.xml

Adicionalmente agregamos el dependencyManagement para poder descargar correctamente las dependencias de Maven.

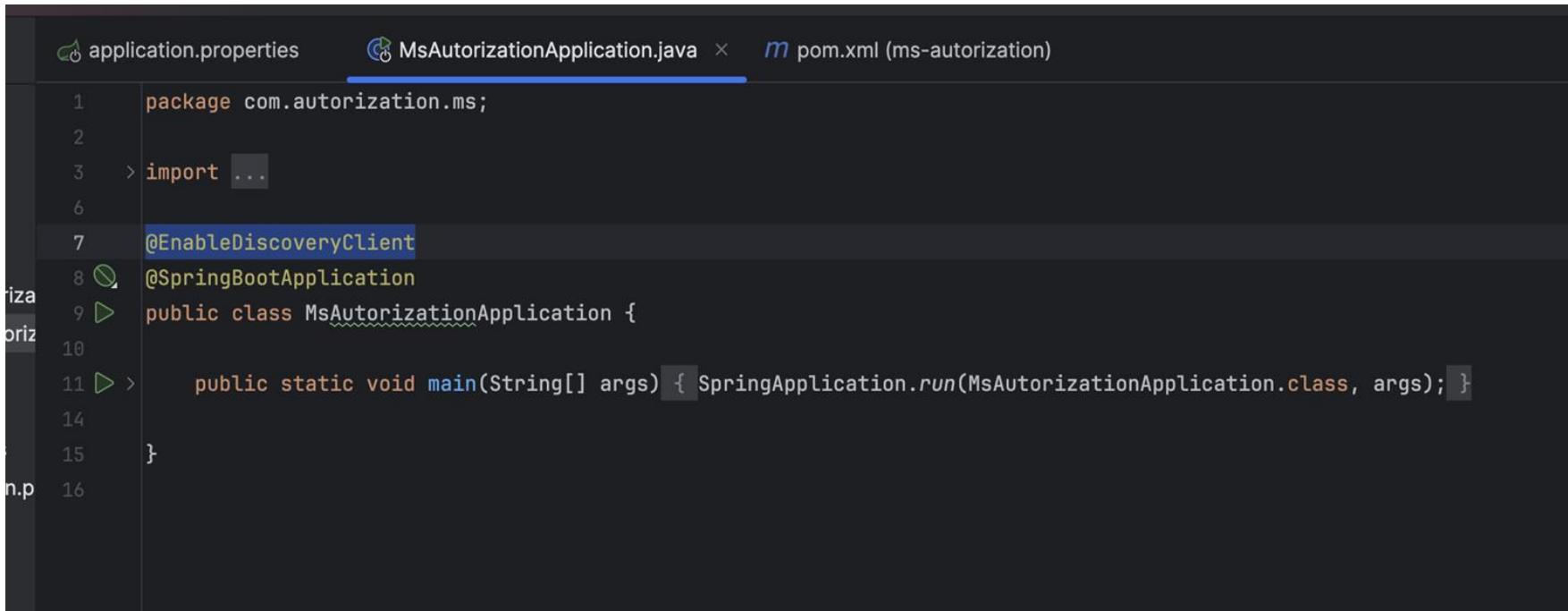
```
<dependencyManagement>
  <dependencies>
    <dependency>
      <groupId>org.springframework.cloud</groupId>
      <artifactId>spring-cloud-dependencies</artifactId>
      <version>${spring-cloud.version}</version>
      <type>pom</type>
      <scope>import</scope>
    </dependency>
  </dependencies>
</dependencyManagement>
```

Ajustando el pom.xml

```
70      </dependencies>
za    71      <dependencyManagement>
riz   72          <dependencies>
p     73              <dependency>
74      |      <groupId>org.springframework.cloud</groupId>
75          |      <artifactId>spring-cloud-dependencies</artifactId>
76          |      <version>${spring-cloud.version}</version>
77          |      <type>pom</type>
78          |      <scope>import</scope>
79      </dependency>
80      </dependencies>
81  </dependencyManagement>
82  <build>
83      <plugins>
84          <plugin>
```

Agregando el EnableDiscoveryClient

EnableDiscoveryClient: Simplifica la comunicación entre microservicios, permitiéndoles encontrar y conectarse entre sí sin configuraciones de red estáticas.



```
application.properties MsAuthorizationApplication.java pom.xml (ms-authorization)
1 package com.autorization.ms;
2
3 > import ...
6
7 @EnableDiscoveryClient
8
9 public class MsAuthorizationApplication {
10
11 >     public static void main(String[] args) { SpringApplication.run(MsAuthorizationApplication.class, args); }
14
15 }
16
```

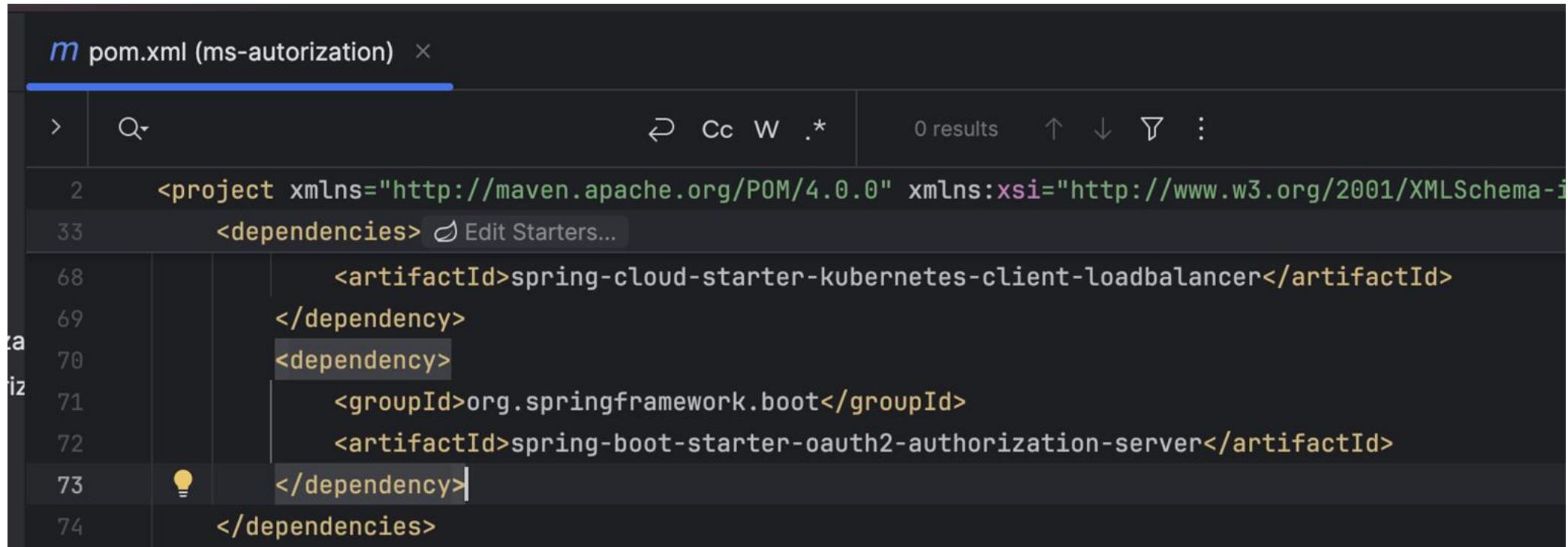
3.2. Configurando el servidor de autorización

Para implementar el oauth2 authorization es necesario añadir la dependencia en el pom.xml.

```
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-oauth2-authorization-server</artifactId>
</dependency>
```

Nota: la documentacion a seguir es <https://docs.spring.io/spring-authorization-server/reference/getting-started.html>

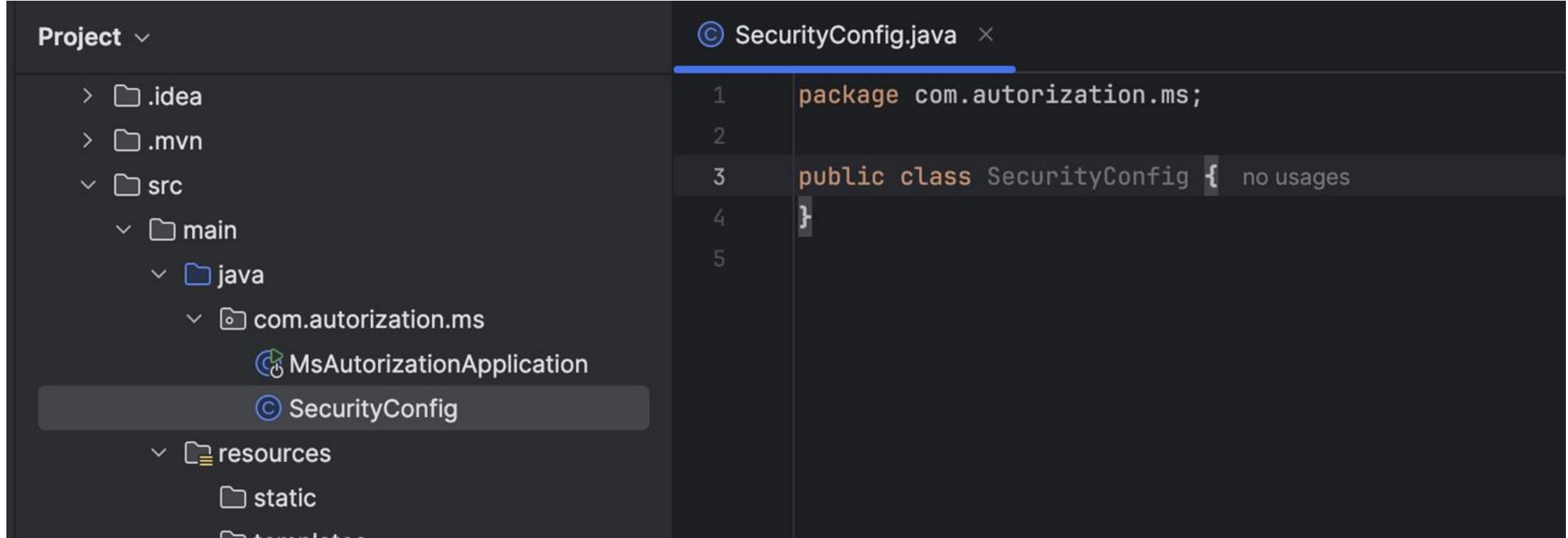
Configurando el servidor de autorización



The screenshot shows a code editor window with the file `pom.xml` open. The code is XML for a Maven project, specifically for a Spring Boot OAuth2 Authorization Server. The dependencies section includes the `spring-cloud-starter-kubernetes-client-loadbalancer` and `spring-boot-starter-oauth2-authorization-server` artifacts.

```
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
<dependencies>
    <dependency>
        <artifactId>spring-cloud-starter-kubernetes-client-loadbalancer</artifactId>
    </dependency>
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-oauth2-authorization-server</artifactId>
    </dependency>
</dependencies>
```

Creamos la clase SecurityConfig



The screenshot shows a Java development environment with a dark theme. On the left, the Project Explorer displays the following directory structure:

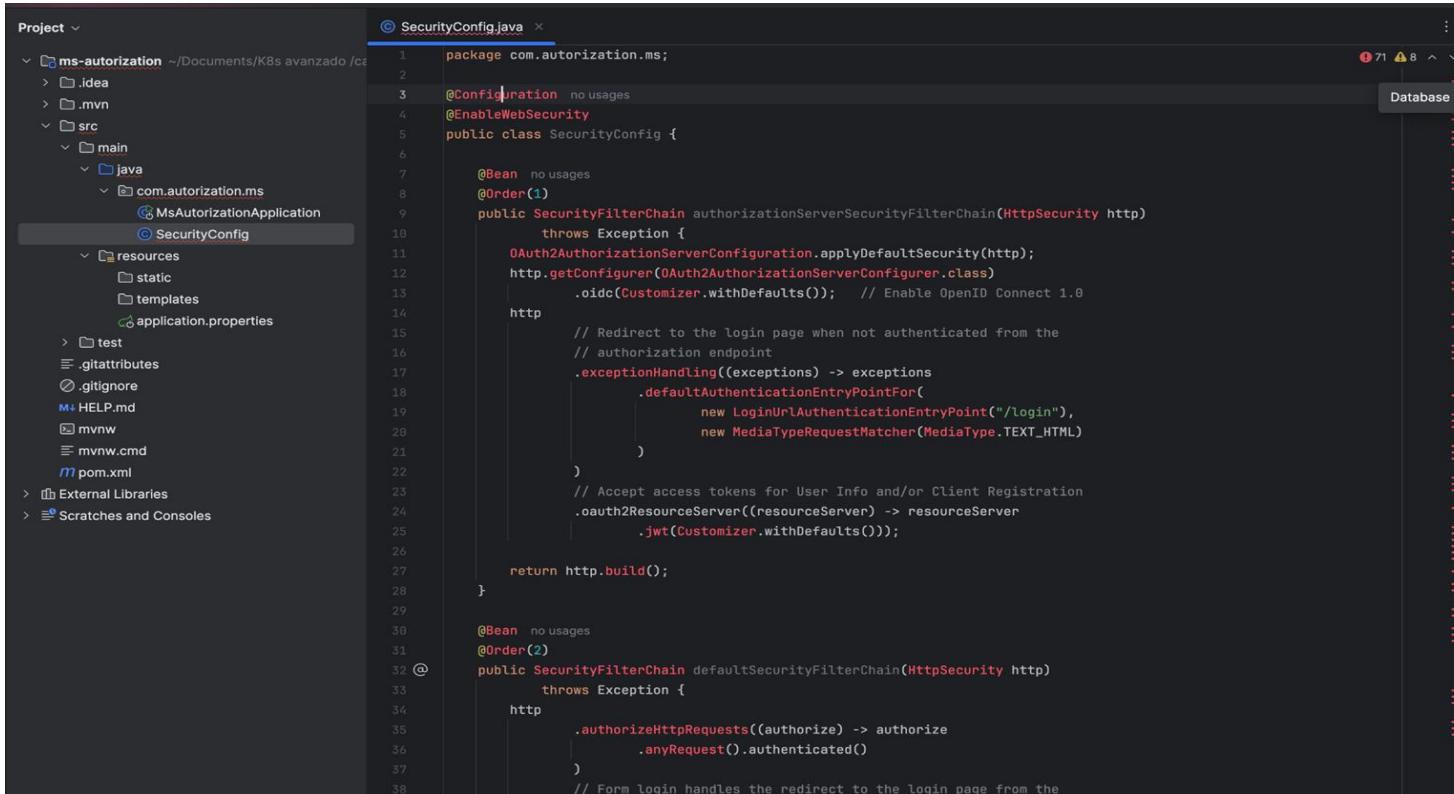
- .idea
- .mvn
- src
 - main
 - java
 - com.autorization.ms
 - MsAuthorizationApplication
 - SecurityConfig
 - resources
 - static

The 'SecurityConfig' class is selected in the project tree, highlighted with a gray background. To the right, the code editor shows the content of `SecurityConfig.java`:

```
1 package com.autorization.ms;
2
3 public class SecurityConfig { no usages
4 }
5
```

Creamos la clase SecurityConfig

Pegamos el código de la documentación de Spring Security.



The screenshot shows a Java project structure in the left panel and a code editor in the right panel. The project structure includes a main package with sub-packages ms-autorization.ms, com.autorization.ms, and SecurityConfig. The SecurityConfig.java file is open in the code editor, displaying the following Java code:

```
1 package com.autorization.ms;
2
3 @Configuration no usages
4 @EnableWebSecurity
5
6 public class SecurityConfig {
7
8     @Bean no usages
9     @Order(1)
10    public SecurityFilterChain authorizationServerSecurityFilterChain(HttpSecurity http)
11        throws Exception {
12        OAuth2AuthorizationServerConfiguration.applyDefaultSecurity(http);
13        http.getConfigurer(OAuth2AuthorizationServerConfigurer.class)
14            .oidc(Customizer.withDefaults()); // Enable OpenID Connect 1.0
15
16        http
17            // Redirect to the login page when not authenticated from the
18            // authorization endpoint
19            .exceptionHandling(exceptions) -> exceptions
20                .defaultAuthenticationEntryPointFor(
21                    new LoginUrlAuthenticationEntryPoint("/login"),
22                    new MediaTypeRequestMatcher(MediaType.TEXT_HTML)
23                )
24            )
25        // Accept access tokens for User Info and/or Client Registration
26        .oauth2ResourceServer((resourceServer) -> resourceServer
27            .jwt(Customizer.withDefaults()));
28
29
30        return http.build();
31    }
32
33    @Bean no usages
34    @Order(2)
35    public SecurityFilterChain defaultSecurityFilterChain(HttpSecurity http)
36        throws Exception {
37        http
38            .authorizeHttpRequests((authorize) -> authorize
39                .anyRequest().authenticated()
40            )
41        // Form login handles the redirect to the login page from the
42    }
}
```

Creamos la clase SecurityConfig

Agregamos las librerías correspondientes.



Netec
APRENDIZAJE EFECTIVO

Ajustando la clase SecurityConfig

- Modificamos el secret del registeredClientRepository secret por 12345.
- Modificamos los scope.

```
@Bean
public RegisteredClientRepository registeredClientRepository() {
    RegisteredClient oidcClient = RegisteredClient.withId(UUID.randomUUID().toString())
        .clientId("user-client")
        .clientSecret("{noop}12345")
        .clientAuthenticationMethod(ClientAuthenticationMethod.CLIENT_SECRET_BASIC)
        .authorizationGrantType(AuthorizationGrantType.AUTHORIZATION_CODE)
        .authorizationGrantType(AuthorizationGrantType.REFRESH_TOKEN)
        .redirectUri("http://127.0.0.1:8080/login/oauth2/code/user-client")
        .postLogoutRedirectUri("http://127.0.0.1:8080/")
        .scope("read")
        .scope("write")
        .clientSettings(ClientSettings.builder().requireAuthorizationConsent(true).build())
        .build();

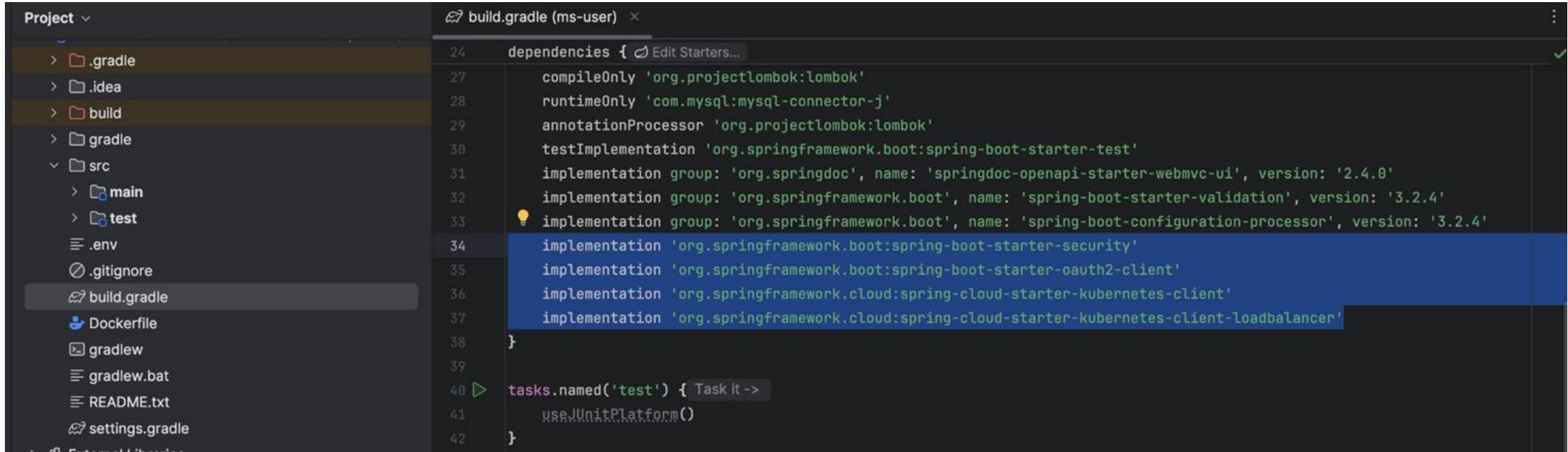
    return new InMemoryRegisteredClientRepository(oidcClient);
}
```

3.3. Configurando OAuth2 Client

En el servicio de ms-user vamos a configurar la seguridad y como primer paso vamos a agregar la dependencia de seguridad.

```
Implementation 'org.springframework.boot:spring-boot-starter-security'  
implementation 'org.springframework.boot:spring-boot-starter-oauth2-client'  
implementation 'org.springframework.cloud:spring-cloud-starter-kubernetes-client'  
implementation 'org.springframework.cloud:spring-cloud-starter-kubernetes-client-loadbalancer'
```

User build.dragle



The screenshot shows a code editor interface with a dark theme. On the left is a project navigation sidebar labeled "Project". It lists several files and folders: .gradle, .idea, build, gradle, src (with main and test subfolders), .env, .gitignore, build.gradle (which is selected and highlighted in blue), Dockerfile, gradlew, gradlew.bat, README.txt, and settings.gradle.

The main editor area has a title bar "build.gradle (ms-user)". The code itself is as follows:

```
dependencies {  
    compileOnly 'org.projectlombok:lombok'  
    runtimeOnly 'com.mysql:mysql-connector-j'  
    annotationProcessor 'org.projectlombok:lombok'  
    testImplementation 'org.springframework.boot:spring-boot-starter-test'  
    implementation group: 'org.springdoc', name: 'springdoc-openapi-starter-webmvc-ui', version: '2.4.0'  
    implementation group: 'org.springframework.boot', name: 'spring-boot-starter-validation', version: '3.2.4'  
    implementation group: 'org.springframework.boot', name: 'spring-boot-configuration-processor', version: '3.2.4'  
    implementation 'org.springframework.boot:spring-boot-starter-security'  
    implementation 'org.springframework.boot:spring-boot-starter-oauth2-client'  
    implementation 'org.springframework.cloud:spring-cloud-starter-kubernetes-client'  
    implementation 'org.springframework.cloud:spring-cloud-starter-kubernetes-client-loadbalancer'  
}  
  
tasks.named('test') { Task it ->  
    useJUnitPlatform()  
}
```

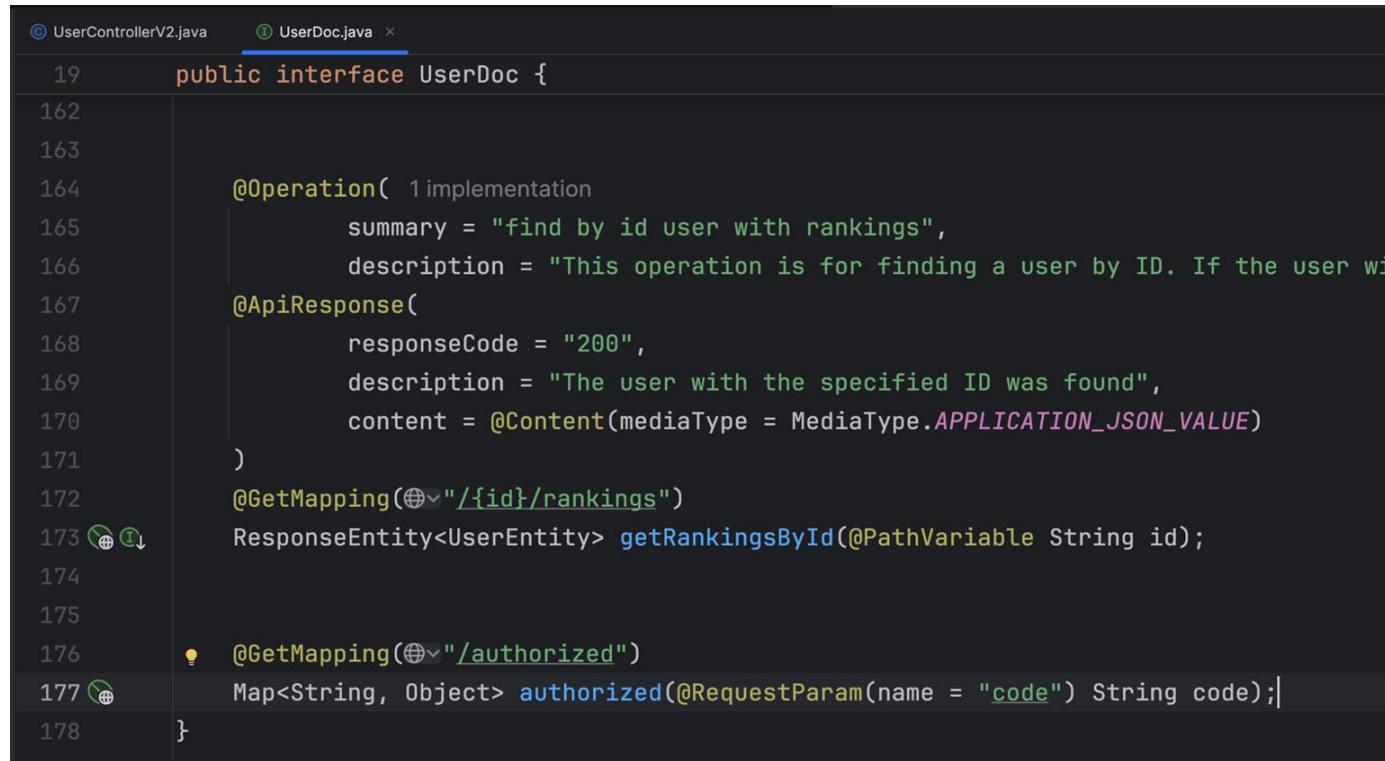
Agregando oauth2 en ms-user

Se realiza la configuración del oauth2 para establecer la seguridad en el microservicio user.

```
application.yml
1  server:
2    port: ${PORT:8001}
3  > spring: <3 keys>
17 > springdoc:
18   api-docs: <1 key>
20   show-actuator: true
21   packages-to-scan: com.ms.user.controller.v2
22 > security:
23   > oauth2:
24     > registration:
25       > ms-user:
26         provider: spring
27         client-id: ms-user
28         client-secret: 12345
29         authorization-grant-type: authorization_code
30         redirect-uri: http://127.0.0.1:8001/authorized
31         scope: read
32         client-name: ms-user
33       > provider:
34         > spring:
35           issuer-uri: http://127.0.0.1:9000
```

Creando endpoint authorized

Para implementar el endpoint authorized primero debemos agregarlo en la documentación , en la clase UserDoc.



```
 UserControllerV2.java UserDoc.java x
19  public interface UserDoc {
162
163
164      @Operation( implementation
165          summary = "find by id user with rankings",
166          description = "This operation is for finding a user by ID. If the user wi
167          @ApiResponse(
168              responseCode = "200",
169              description = "The user with the specified ID was found",
170              content = @Content(mediaType = MediaType.APPLICATION_JSON_VALUE)
171          )
172          @GetMapping("/{id}/rankings")
173      ResponseEntity<UserEntity> getRankingsById(@PathVariable String id);
174
175
176      @GetMapping("/authorized")
177      Map<String, Object> authorized(@RequestParam(name = "code") String code);
178 }
```

Creando endpoint authorized

Implementamos el método en la clase de UserControllerV2.

The screenshot shows a Java code editor with the following code:

```
@AllArgsConstructor  
@RestController  
public class UserControllerV2 implements UserDoc {  
  
    private final IUserService userService;  
  
    @Override no usages  
    public ResponseEntity<User> getUser(String id) {  
  
        @Override no usages  
        public ResponseEntity<User> authorized(String id) {  
  
            @Override no usages  
        }  
    }  
}
```

A code completion tooltip is displayed over the `authorized` method declaration, showing the following information:

- Class 'UserControllerV2' must either be declared abstract or implement abstract method 'authorized(String)' in 'UserDoc' ...
- [Implement methods](#) ↗ ↘ More actions... ↗ ↘
- com.ms.user.controller.v2
- @RestController
- public class UserControllerV2
- implements UserDoc
- ms-user.main

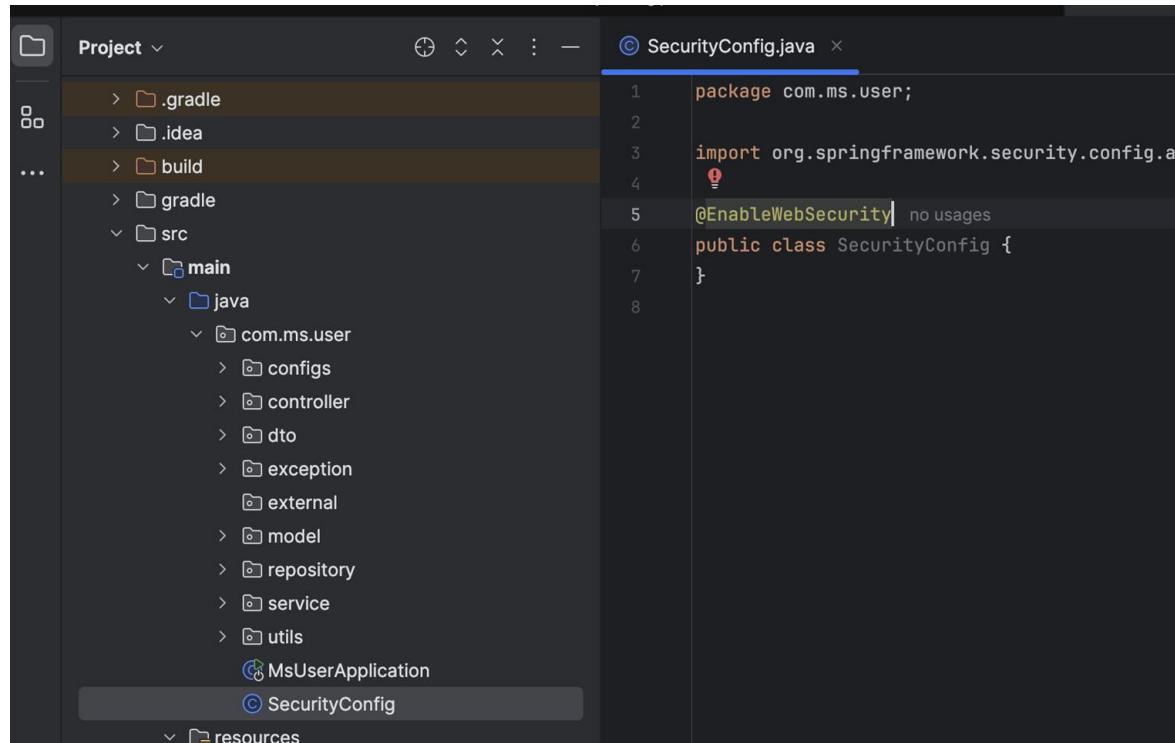
Creando endpoint authorized

Una vez implementado, vamos a retornar el “code”.

```
 UserControllerV2.java × UserDoc.java
21  public class UserControllerV2 implements UserDoc {
54      @Override no usages
55  ↗>  public ResponseEntity<?> testError(String message) { throw new M
58
59      @Override no usages
60  ↗>  public ResponseEntity getRankingsById(String id) { return this.i
63
64      @Override no usages
65  ↗>  public Map<String, Object> authorized(String code) {
66          return Collections.singletonMap("code", code);
67      }
68 }
```

Configurando SecurityWeb

A la misma altura de la clase principal, creamos la clase SecurityConfig y agregamos las etiquetas @EnableWebSecurity.



The screenshot shows a Java IDE interface with a dark theme. On the left, the Project Explorer displays the project structure:

- .gradle
- .idea
- build
- gradle
- src
 - main
 - java
 - com.ms.user
 - configs
 - controller
 - dto
 - exception
 - external
 - model
 - repository
 - service
 - utils
 - MsUserApplication
 - SecurityConfig
 - resources

Configurando SecurityWeb

Agregamos el método para filtrar las peticiones que llegan al servicio de clientes, esto con el objetivo de añadir mayor seguridad.

```
4
5     @Bean
6     SecurityFilterChain securityFilterChain(HttpSecurity http) throws Exception {
7         http
8             .authorizeHttpRequests((authorize) -> authorize
9                 .requestMatchers(ignoredPath("/authorized"), ignoredPath("/login")).permitAll()
10                .requestMatchers(HttpServletRequestMethod.GET, ignoredPath("/api/v2/user"), ignoredPath("/{id}")).hasAnyAuthority(...authorities: "SCOPE_read", "SCOPE_write")
11                .requestMatchers(HttpServletRequestMethod.POST, ignoredPath("/api/v2/user")).hasAuthority("SCOPE_write")
12                .requestMatchers(HttpServletRequestMethod.PUT, ignoredPath("/api/v2/user")).hasAuthority("SCOPE_write")
13                .requestMatchers(HttpServletRequestMethod.DELETE, ignoredPath("/api/v2/user")).hasAuthority("SCOPE_write")
14                .anyRequest().authenticated()
15            )
16            .sessionManagement((session) -> session.sessionCreationPolicy(SessionCreationPolicy.STATELESS))
17            .oauth2Login((oauth2Login) -> oauth2Login.loginPage("/oauth2/authorization/msvc-usuarios-client"))
18            .oauth2Client(withDefaults())
19            .oauth2ResourceServer((oauth2ResourceServer) -> oauth2ResourceServer.jwt(withDefaults()));
20
21         return http.build();
22     }

```

3.4. Configurando OAuth2 Resource Server

OAuth2 Resource Server permite a las aplicaciones validar y procesar los tokens de acceso emitidos por un Authorization Server. Para implementarlo agregamos una nueva dependencia:

Implementation 'org.springframework.boot:spring-boot-starter-oauth2-resource-server'

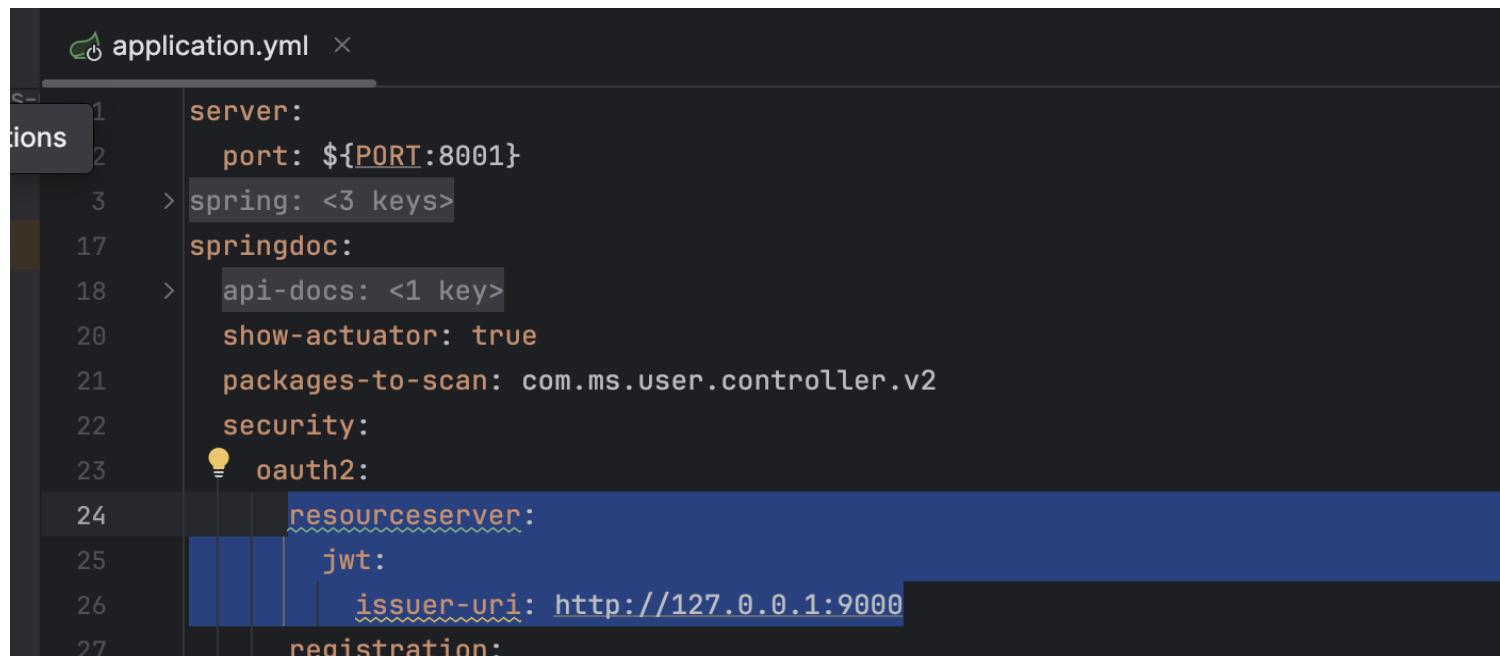
Configurando OAuth2 Resource Server

```
23
24 > dependencies { ⚙ Edit Starters...
25     implementation 'org.springframework.boot:spring-boot-starter-data-jpa'
26     implementation 'org.springframework.boot:spring-boot-starter-web'
27     compileOnly 'org.projectlombok:lombok'
28     runtimeOnly 'com.mysql:mysql-connector-j'
29     annotationProcessor 'org.projectlombok:lombok'
30     testImplementation 'org.springframework.boot:spring-boot-starter-test'
31     implementation group: 'org.springdoc', name: 'springdoc-openapi-starter-webmvc-ui', version: '2.4.0'
32     implementation group: 'org.springframework.boot', name: 'spring-boot-starter-validation', version: '3.2.4'
33     implementation group: 'org.springframework.boot', name: 'spring-boot-configuration-processor', version: '3.2.4'
34     implementation 'org.springframework.boot:spring-boot-starter-security'
35     implementation 'org.springframework.boot:spring-boot-starter-oauth2-client'
36     implementation 'org.springframework.boot:spring-boot-starter-oauth2-resource-server' ⚡
37     implementation 'org.springframework.cloud:spring-cloud-starter-kubernetes-client'
38     implementation 'org.springframework.cloud:spring-cloud-starter-kubernetes-client-loadbalancer'
39 }
```

Configurar application.yaml

Agregamos la configuración para validar token.

```
resourceserver:  
  jwt:  
    issuer-uri: http://127.0.0.1:9000
```



The screenshot shows a code editor with a dark theme. The file is named "application.yaml". The configuration is as follows:

```
server:  
  port: ${PORT:8001}  
spring:  
  <3 keys>  
springdoc:  
  <1 key>  
api-docs: true  
show-actuator: true  
packages-to-scan: com.ms.user.controller.v2  
security:  
  oauth2:  
    resourceserver:  
      jwt:  
        issuer-uri: http://127.0.0.1:9000  
    registration:
```

The "resourceserver" section is highlighted with a blue background, and the "jwt" section within it is also highlighted. The "issuer-uri" key under "jwt" is also highlighted.

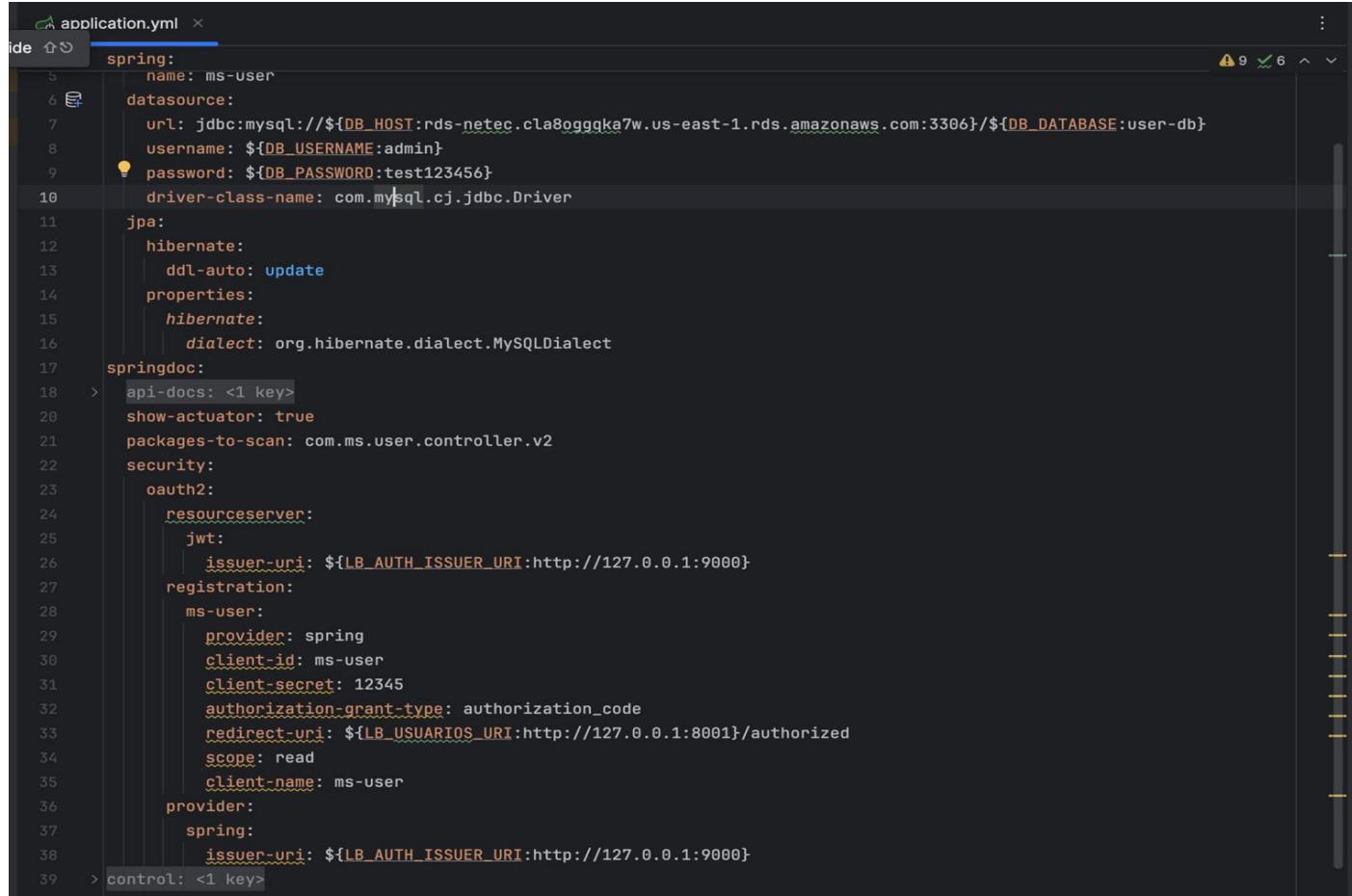
3.5. Configurando variables de entorno en msvc usuarios

Es importante crear variables de entorno en los microservicios ya que la información entre ambientes debe cambiar, como conexiones a base de datos, contraseñas y demás.

la estructura para crear variables de entorno es la siguiente

- \${NOMBRE_VARIABLE:VALOR_DEFECTO}

Configurando variables de entorno en msvc usuarios



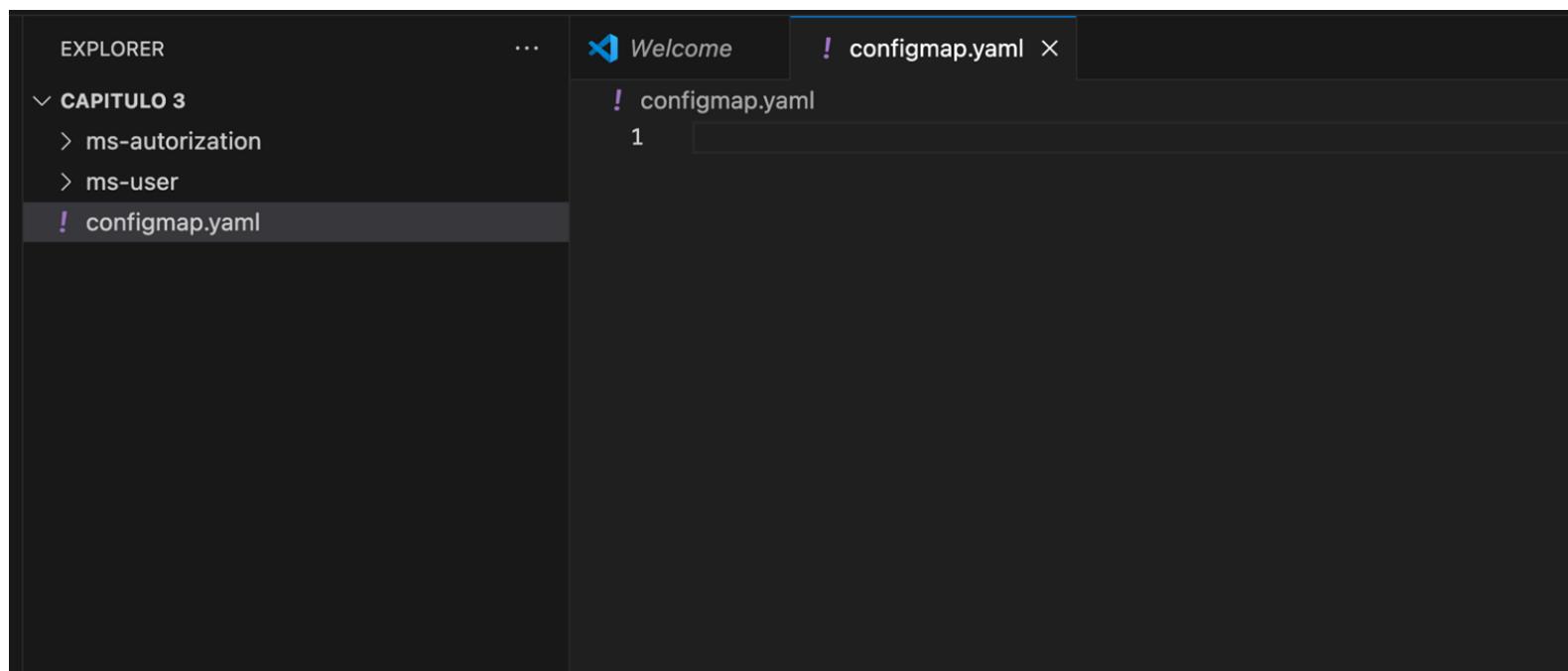
The screenshot shows a code editor window with the file 'application.yml' open. The file contains YAML configuration for a Spring Boot application, specifically for a 'ms-user' service. The configuration includes:

- spring:** Contains settings for a database connection (name: ms-user, datasource, url, username, password) and a JPA configuration (driver-class-name: com.mysql.cj.jdbc.Driver).
- springdoc:** Configures API documentation and actuator endpoints.
- security:** Configures OAuth2 security, including a resource server (jwt, issuer-uri: \${LB_AUTH_ISSUER_URI}:http://127.0.0.1:9000) and client registration for 'ms-user' (provider: spring, client_id: ms-user, client_secret: 12345, authorization_grant_type: authorization_code, redirect_uri: \${LB_USUARIOS_URI}:http://127.0.0.1:8001/authorized, scope: read, client_name: ms-user).
- control:** Configures another OAuth2 provider (provider: spring, issuer-uri: \${LB_AUTH_ISSUER_URI}:http://127.0.0.1:9000).

The code editor interface shows syntax highlighting and some code completion or validation markers.

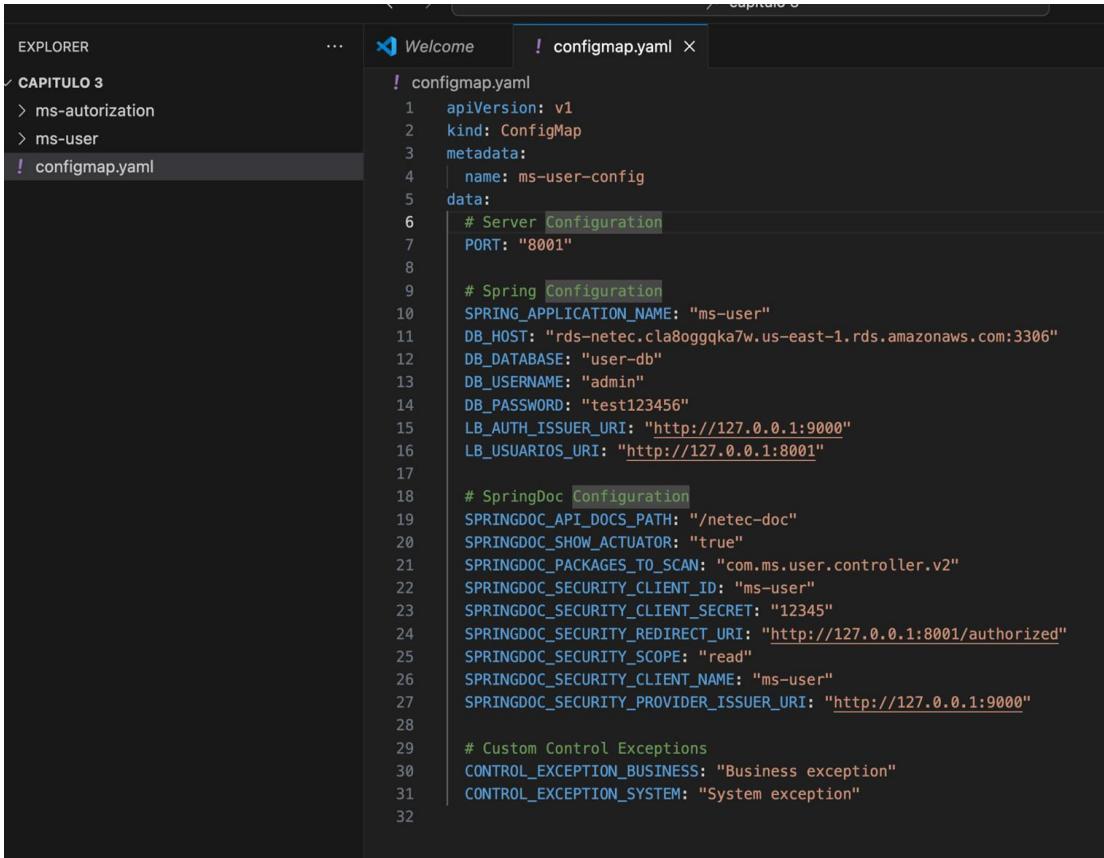
ConfigMap

ConfigMap es un archivo yaml que se utiliza en Kubernetes, permite almacenar los datos por clave-valor que son necesarios como variables de entorno.
Para poder utilizarlo se debe crear un archivo llamado configmap.yaml



ConfigMap

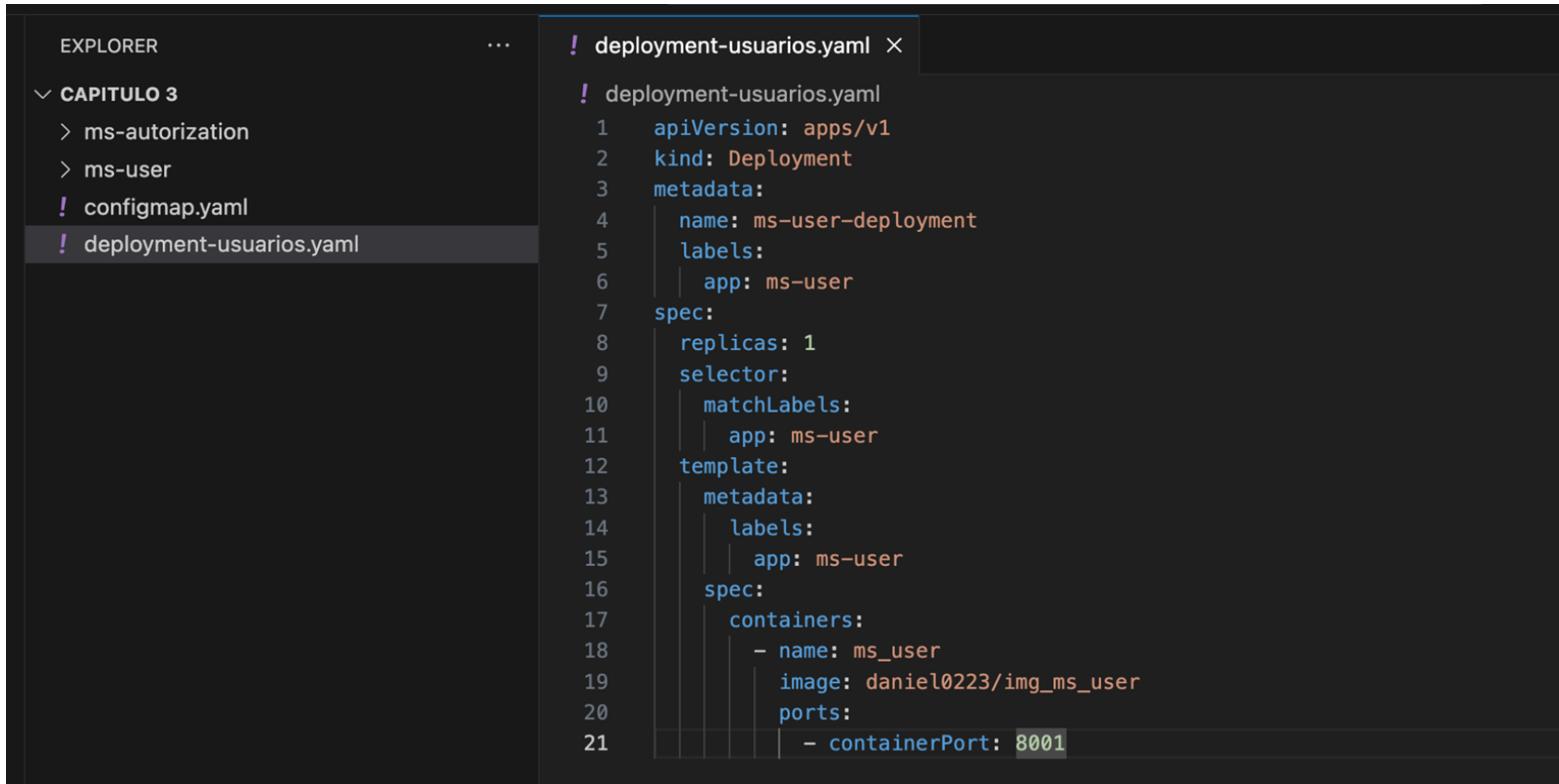
Agregamos las variables de entorno.



```
! configmap.yaml
 1  apiVersion: v1
 2  kind: ConfigMap
 3  metadata:
 4    name: ms-user-config
 5  data:
 6    # Server Configuration
 7    PORT: "8001"
 8
 9    # Spring Configuration
10   SPRING_APPLICATION_NAME: "ms-user"
11   DB_HOST: "rds-netec.cla8oggqka7w.us-east-1.rds.amazonaws.com:3306"
12   DB_DATABASE: "user-db"
13   DB_USERNAME: "admin"
14   DB_PASSWORD: "test123456"
15   LB_AUTH_ISSUER_URI: "http://127.0.0.1:9000"
16   LB_USUARIOS_URI: "http://127.0.0.1:8001"
17
18   # SpringDoc Configuration
19   SPRINGDOC_API_DOCS_PATH: "/netec-doc"
20   SPRINGDOC_SHOW_ACTUATOR: "true"
21   SPRINGDOC_PACKAGES_TO_SCAN: "com.ms.user.controller.v2"
22   SPRINGDOC_SECURITY_CLIENT_ID: "ms-user"
23   SPRINGDOC_SECURITY_CLIENT_SECRET: "12345"
24   SPRINGDOC_SECURITY_REDIRECT_URI: "http://127.0.0.1:8001/authorized"
25   SPRINGDOC_SECURITY_SCOPE: "read"
26   SPRINGDOC_SECURITY_CLIENT_NAME: "ms-user"
27   SPRINGDOC_SECURITY_PROVIDER_ISSUER_URI: "http://127.0.0.1:9000"
28
29   # Custom Control Exceptions
30   CONTROL_EXCEPTION_BUSINESS: "Business exception"
31   CONTROL_EXCEPTION_SYSTEM: "System exception"
```

deployment-user

Para mapear las variables del ConfigMap en el deploy (Pod), es necesario crear el deployment-user con la estructura básica.



```
! deployment-usuarios.yaml ×
! deployment-usuarios.yaml
1 apiVersion: apps/v1
2 kind: Deployment
3 metadata:
4   name: ms-user-deployment
5   labels:
6     app: ms-user
7 spec:
8   replicas: 1
9   selector:
10    matchLabels:
11      app: ms-user
12   template:
13     metadata:
14       labels:
15         app: ms-user
16     spec:
17       containers:
18         - name: ms_user
19           image: daniel0223/img_ms_user
20           ports:
21             - containerPort: 8001
```

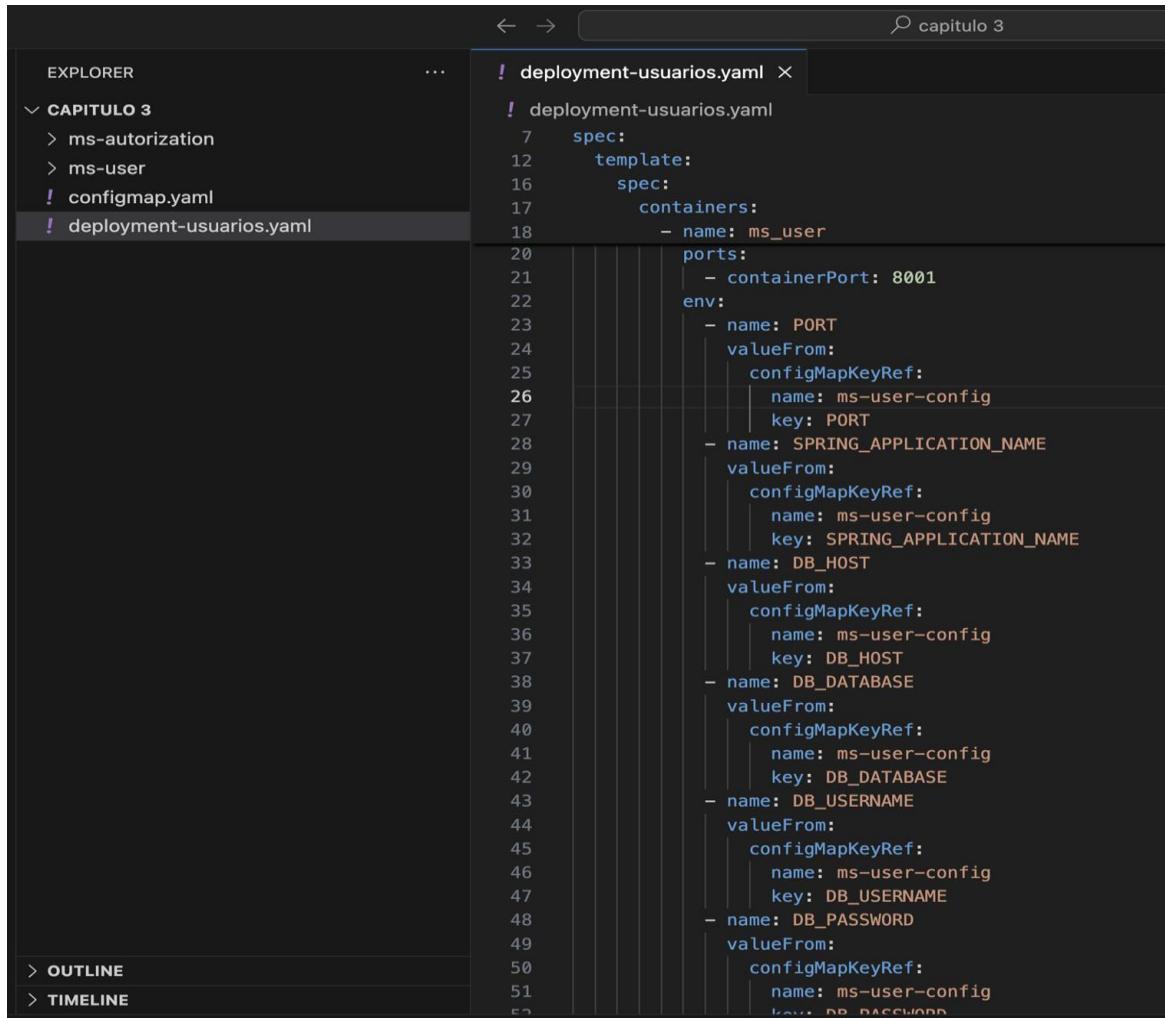
deployment-user

Agregamos una nueva sección con la siguiente estructura.

env:

- name: <NOMBRE_VARIABLE>
- valueFrom:
- configMapKeyRef:
- name: <nombre_config_map>
 - key: <nombre_llave>

deployment-user



The screenshot shows a code editor interface with a dark theme. On the left, the Explorer sidebar displays a project structure under 'CAPITULO 3' with files: ms-autorization, ms-user, configmap.yaml, and deployment-usuarios.yaml. The 'deployment-usuarios.yaml' file is currently selected and open in the main editor area. The code itself is a Kubernetes Deployment manifest named 'deployment-usuarios.yaml'. It defines a single container named 'ms_user' with port 8001. The container's environment variables are defined using ConfigMap keys from a 'ms-user-config' ConfigMap. The variables include PORT, SPRING_APPLICATION_NAME, DB_HOST, DB_DATABASE, DB_USERNAME, and DB_PASSWORD.

```
! deployment-usuarios.yaml
7   spec:
12     template:
16       spec:
17         containers:
18           - name: ms_user
20             ports:
21               - containerPort: 8001
22             env:
23               - name: PORT
24                 valueFrom:
25                   configMapKeyRef:
26                     name: ms-user-config
27                     key: PORT
28               - name: SPRING_APPLICATION_NAME
29                 valueFrom:
30                   configMapKeyRef:
31                     name: ms-user-config
32                     key: SPRING_APPLICATION_NAME
33               - name: DB_HOST
34                 valueFrom:
35                   configMapKeyRef:
36                     name: ms-user-config
37                     key: DB_HOST
38               - name: DB_DATABASE
39                 valueFrom:
40                   configMapKeyRef:
41                     name: ms-user-config
42                     key: DB_DATABASE
43               - name: DB_USERNAME
44                 valueFrom:
45                   configMapKeyRef:
46                     name: ms-user-config
47                     key: DB_USERNAME
48               - name: DB_PASSWORD
49                 valueFrom:
50                   configMapKeyRef:
51                     name: ms-user-config
52                     key: DB_PASSWORD
```

3.6. Configurando variables de entorno en MSVC Auth

Creamos las variables de entorno en nuestro microservicio Auth, en la clase `SecurityConfig` con el método `RegisteredClientRepository`.

Estas variables las obtendremos del ambiente, por tal razón, se debe agregar el atributo `Environment`.

Atributo Environment

```
41  
42     @Configuration  
43     @EnableWebSecurity  
44     public class SecurityConfig {  
45  
46         @Autowired  
47     private Environment environment;  
48 }
```

Leyendo Environment

Para leer la variable, usamos el atributo env y especificamos la variable para leer env.getProperty("NAME_VARIABLE").

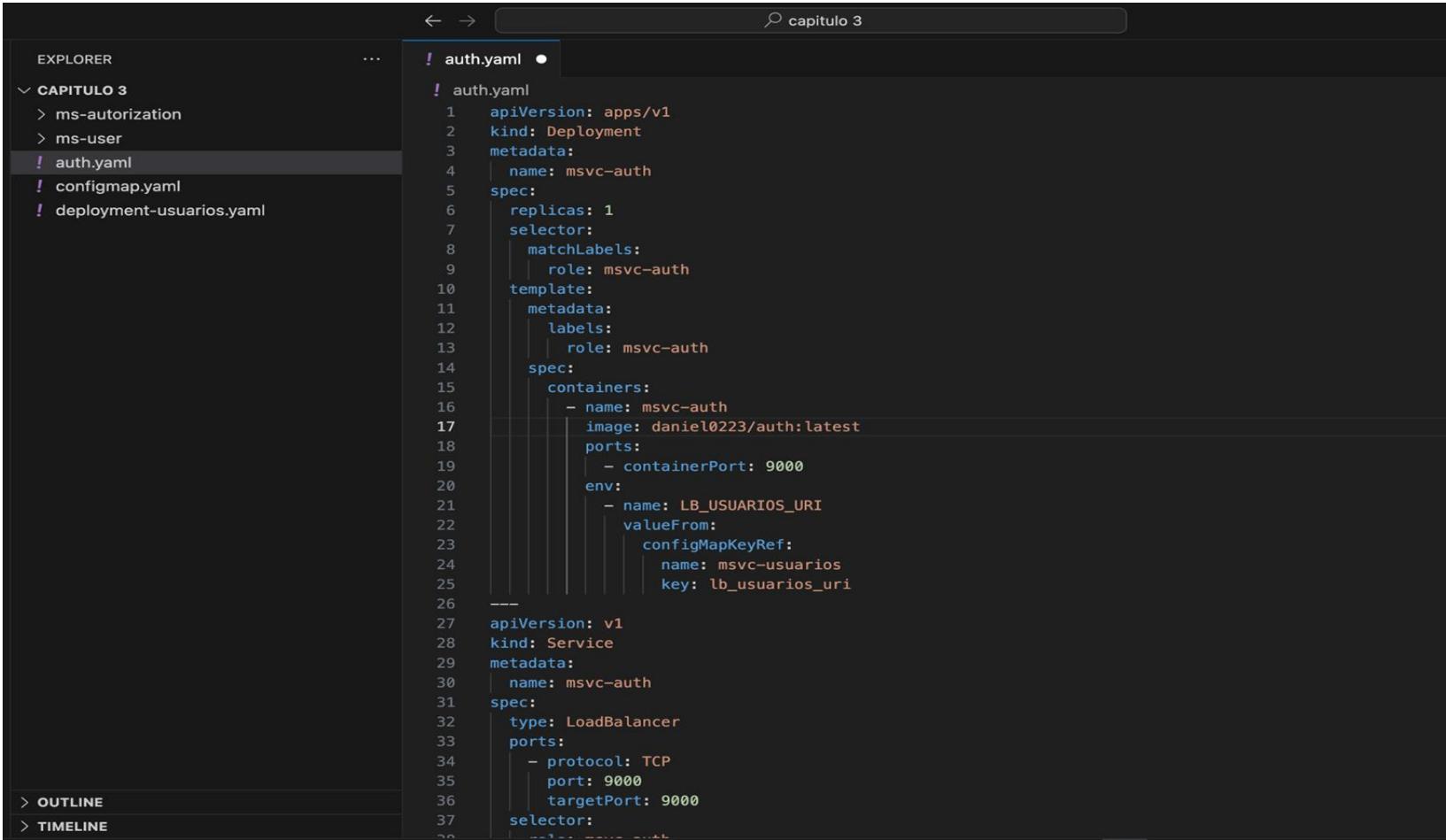
```
@Bean
public RegisteredClientRepository registeredClientRepository() {
    RegisteredClient oidcClient = RegisteredClient.withId(UUID.randomUUID().toString())
        .clientId("ms-user")
        .clientSecret("{noop}12345")
        .clientAuthenticationMethod(ClientAuthenticationMethod.CLIENT_SECRET_BASIC)
        .authorizationGrantType(AuthorizationGrantType.AUTHORIZATION_CODE)
        .authorizationGrantType(AuthorizationGrantType.REFRESH_TOKEN)
        .redirectUri(environment.getProperty("LB_USUARIOS_URI")+"/login/oauth2/code/ms-user")
        .redirectUri(environment.getProperty("LB_USUARIOS_URI")+"/authorized")
        .postLogoutRedirectUri("http://127.0.0.1:8080/")
        .scope("read")
        .scope("write")
        .clientSettings(ClientSettings.builder().requireAuthorizationConsent(true).build())
        .build();

    return new InMemoryRegisteredClientRepository(oidcClient);
}
```

3.7. Escribiendo Deployment y Service de Auth

Procedemos a crear el manifiesto de Kubernetes para el microservicio Auth con el nombre deployment-auth.yaml, este manifiesto va a tener las configuraciones necesarias para desplegar el Pod de una forma correcta en Kubernetes, tanto como su réplica, puertos, variables de entorno y demás.

Escribiendo Deployment y Service de Auth



```
! auth.yaml ●
! auth.yaml
1 apiVersion: apps/v1
2 kind: Deployment
3 metadata:
4   name: msvc-auth
5 spec:
6   replicas: 1
7   selector:
8     matchLabels:
9       role: msvc-auth
10    template:
11      metadata:
12        labels:
13          role: msvc-auth
14      spec:
15        containers:
16          - name: msvc-auth
17            image: daniel0223/auth:latest
18            ports:
19              - containerPort: 9000
20            env:
21              - name: LB_USUARIOS_URI
22                valueFrom:
23                  configMapKeyRef:
24                    name: msvc-usuarios
25                    key: lb_usuarios_uri
26
27 apiVersion: v1
28 kind: Service
29 metadata:
30   name: msvc-auth
31 spec:
32   type: LoadBalancer
33   ports:
34     - protocol: TCP
35       port: 9000
36       targetPort: 9000
37   selector:
```

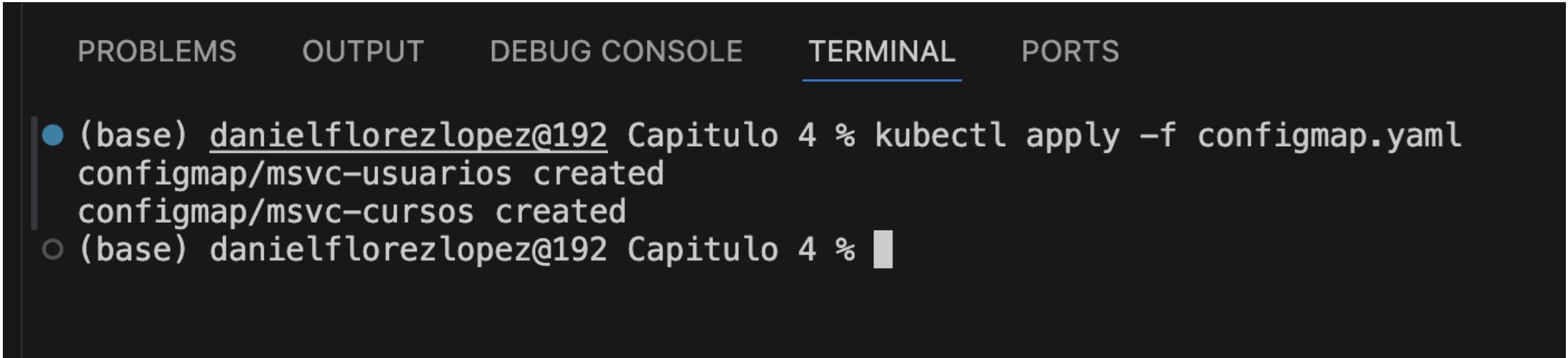
3.8. Aplicando configuraciones y cambios, probando con Postman

En esta sección vamos a empezar a realizar el despliegue de todas las configuraciones previas que realizamos en el código en Kubernetes. Para este paso nos vamos a apoyar de los yaml y comandos que hemos visto a lo largo del curso.

Desplegando ConfigMap

Para desplegar el ConfigMap usamos el siguiente comando:

- kubectl apply -f configmap.yaml



The screenshot shows a terminal window with several tabs at the top: PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL (which is underlined), and PORTS. The terminal content displays the following text:

```
● (base) danielflorezlopez@192 Capitulo 4 % kubectl apply -f configmap.yaml
configmap/msvc-usuarios created
configmap/msvc-cursos created
○ (base) danielflorezlopez@192 Capitulo 4 % █
```

Desplegando auth.yaml

Para desplegar el auth usamos el siguiente comando:

- kubectl apply -f auth.yaml

```
● (base) danielflorezlopez@192 Capitulo 4 % kubectl apply -f auth.yaml
deployment.apps/msvc-auth created
service/msvc-auth created
○ (base) danielflorezlopez@192 Capitulo 4 % █
```

Probamos servicio

El servicio responderá 401 ya que es necesario tener el token o loguearse para obtener el token.

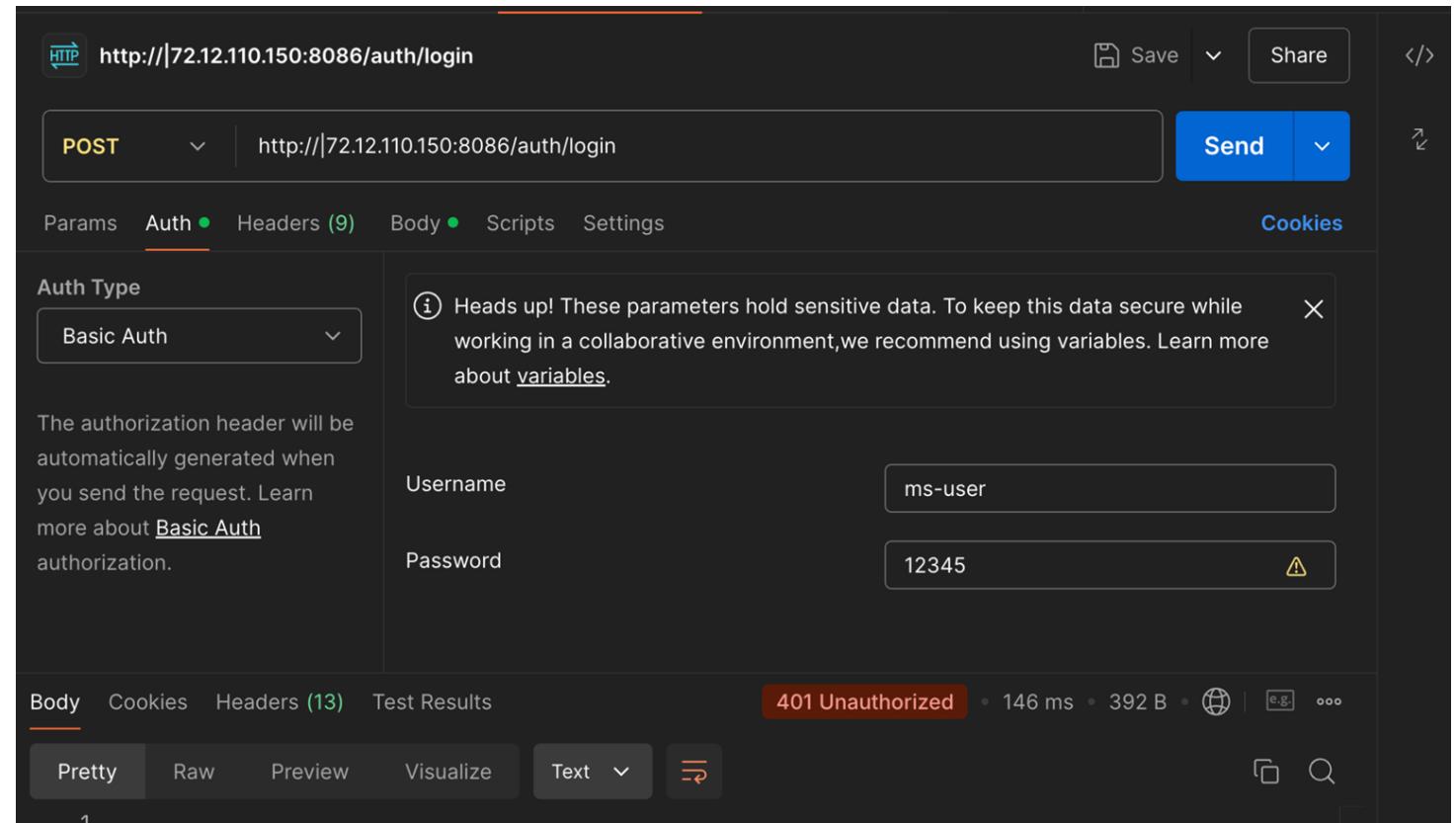
The screenshot shows a POST request to `http://72.12.110.150:8086/auth/login`. The Headers tab is selected, showing 8 headers. The Body tab is also visible. The response status is **401 Unauthorized**, with a response time of 146 ms and a body size of 392 B. The response content is a single character: `1`.

3.9. Login OAuth 2.1

Para poder realizar el login en la aplicación es necesario que se genere un token, ya que sin token no es posible realizar la autenticación correctamente. Para esto, debemos intentar loguearnos en la aplicación y así poder generar el token.

Obtener el token

Para obtener el token debemos seleccionar en POST Basic Auth e ingresar las credenciales con las que se configuró el servicio en la clase registeredClientRepository.



Obtener el token

HTTP <http://172.12.110.150:8086/auth/login>

Save Share </>

POST <http://172.12.110.150:8086/auth/login> Send ▾ ↗

Params Auth Headers (12) Body Scripts Settings Cookies

Headers 9 hidden

	Key	Value	Description	...	Bulk Edit	Presets	...
<input checked="" type="checkbox"/>	code	eyJhbGciOiJIUzUxMiJ9.eyJyb2xIljoiY...			
<input checked="" type="checkbox"/>	grant_type	authorization_code		...			
<input checked="" type="checkbox"/>	redirect_uri	http://172.12.110.150/authorization		...			
	Key	Value	Description				

Body Cookies Headers (14) Test Results 200 OK • 113 ms • 670 B • [Global](#) | [e.g.](#) •••

Pretty Raw Preview Visualize JSON ▾  

```
1 {  
2   "token": "eyJhbGciOiJIUzUxMiJ9.  
     eyJyb2xIljoiYWRtaW4iLCJjb21wYW55IjoiQXVsYU1hdHJpeIIsImlkIjoxLCJzdWIiOiJkYW5pZWwiLCJpYXQi  
     OjE3MzA2MTE5NzgsImV4cCI6MTczMDY1NTE3OH0.  
     fk4h600DyMQmMVEBkHu88PyipqrVC1xNsq7bLmImsrErwZ3CdBWzVIw1kNCRZKnyjuFfAP-CpSFv3Qma1tYRvA"
```

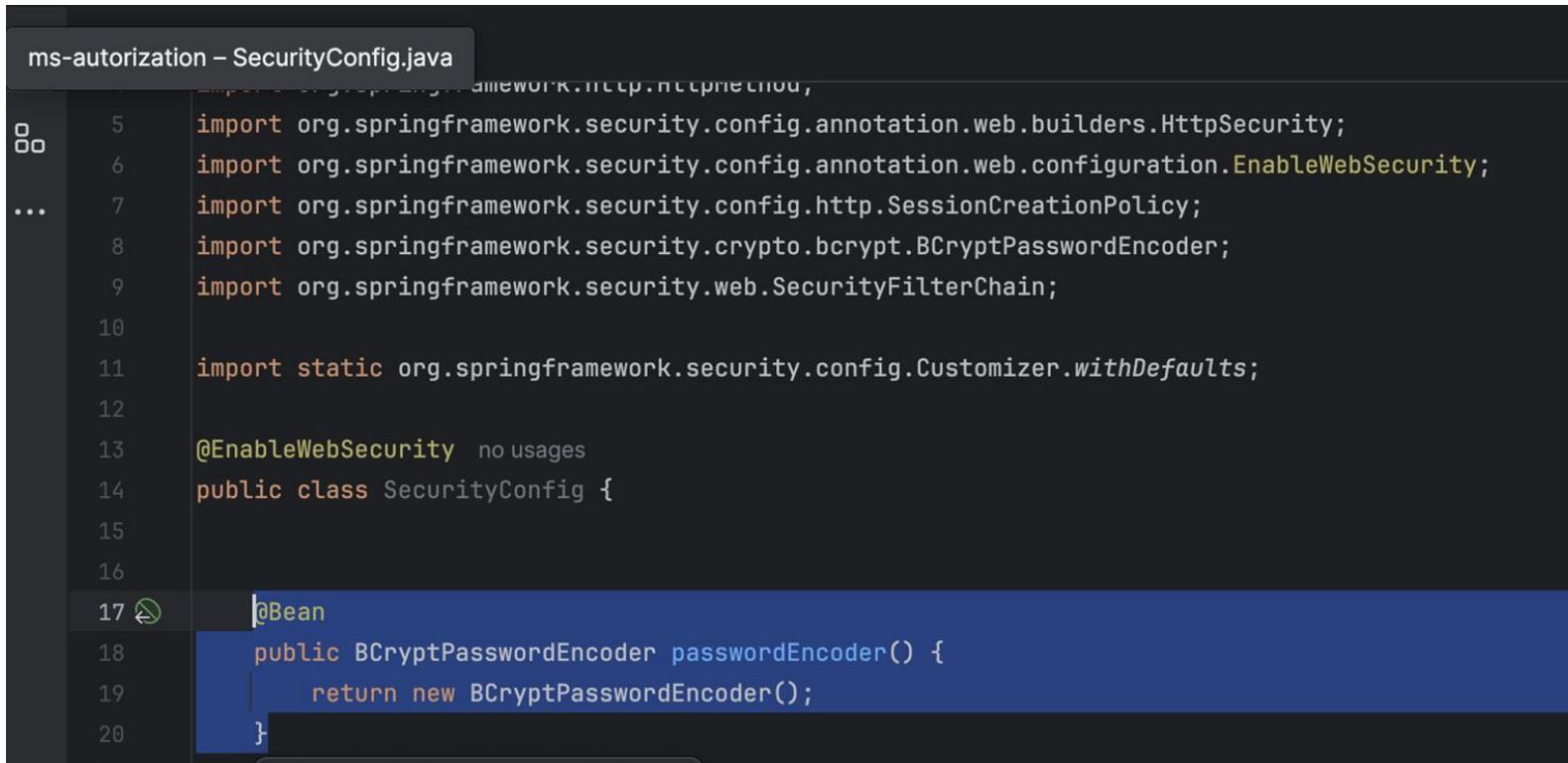
3.10. Encriptando Password BCrypt

La encriptación de contraseñas es importante para dar más seguridad y no dejarla expuesta en caso de que sea interceptada o puedan acceder a la base de datos personal no autorizado.

Para este proceso, una vez encriptada la contraseña, el sistema realizará una serie de algoritmos para validar si es correcta.

Crear passwordEncoder en ms-user

Crearemos un nuevo método que retorne el BCryptPasswordEncoder.



```
ms-autorization - SecurityConfig.java
  ...
  5 import org.springframework.security.config.annotation.web.builders.HttpSecurity;
  6 import org.springframework.security.config.annotation.web.configuration.EnableWebSecurity;
  ...
  7 import org.springframework.security.config.http.SessionCreationPolicy;
  8 import org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;
  9 import org.springframework.security.web.SecurityFilterChain;
  ...
 10
 11 import static org.springframework.security.config.Customizer.withDefaults;
 12
 13 @EnableWebSecurity no usages
 14 public class SecurityConfig {
 15
 16
 17     @Bean
 18     public BCryptPasswordEncoder passwordEncoder() {
 19         return new BCryptPasswordEncoder();
 20     }
 21 }
```

Implementar en controller passwordEncoder

```
@AllArgsConstructor  
@RestController  
public class UserControllerV2 implements UserDoc {  
  
    private final IUserService iUserService;  
  
    private final BCryptPasswordEncoder passwordEncoder
```

Implementar en controller passwordEncoder

Vamos a cifrar la contraseña cuando ingresa al servicio de create user.

```
@Override no usages
public ResponseEntity<UserEntity> create(UserDto userDto) {
    userDto.setPassword(passwordEncoder.encode(userDto.getPassword()));
    ⚡ Extract Surround // ⌂ : service.create(userDto);
}
```

3.11. Configurando BCrypt Password Encoder en el servidor de autorización

En el microservicio usuarios se realizó la encriptación de la contraseña, ahora para poder validar o desencriptar la contraseña, vamos a realizar el ajuste en el servicio de autorización.

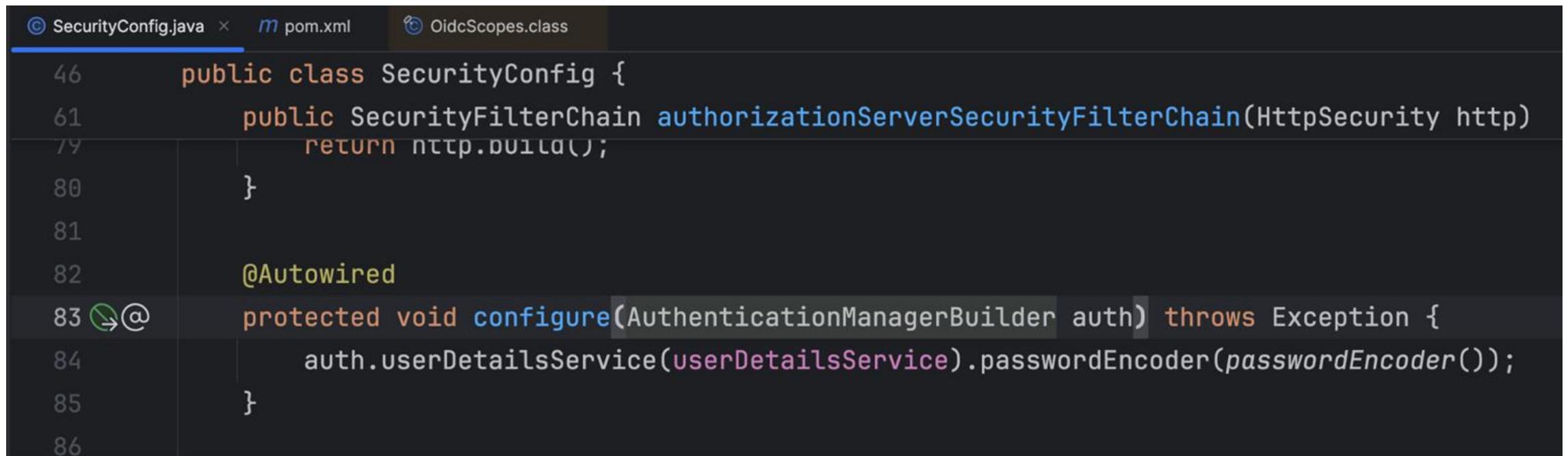
Crearemos el atributo UserDetailsService en la clase SecurityConfig.

Inyectando SecurityConfig

```
43 @Configuration
44 @EnableWebSecurity
45 public class SecurityConfig {
46
47     @Autowired
48     private Environment environment;
49
50     @Autowired
51     private UserDetailsService userDetailsService;
52
53     @Bean
54     public static BCryptPasswordEncoder passwordEncoder() {
55         return new BCryptPasswordEncoder();
56     }
57 }
```

Método configure

Crearemos el método configure para manejar la autenticación.



The screenshot shows a Java code editor with three tabs at the top: 'SecurityConfig.java' (selected), 'pom.xml', and 'OidcScopes.class'. The code in 'SecurityConfig.java' is as follows:

```
46     public class SecurityConfig {
47         public SecurityFilterChain authorizationServerSecurityFilterChain(HttpSecurity http) {
48             return http.build();
49         }
50     }
51
52     @Autowired
53     protected void configure(AuthenticationManagerBuilder auth) throws Exception {
54         auth.userDetailsService(userDetailsService).passwordEncoder(passwordEncoder());
55     }
56 }
```

3.12. Implementando método handler login (Controlador)

Lo que se busca es que por medio de esta configuración podamos loguearnos desde un web service directamente y obtener el token mucho más fácil. Para esto debemos crear un end point llamado login en el controller.

Creando endpoint

En el microservicio user crearemos primero la doc del endpoint de login.



The screenshot shows a code editor with two tabs: UserControllerV2.java and UserDoc.java. The UserDoc.java tab is active, displaying the following Java code:

```
19     public interface UserDoc {  
173     ResponseEntity<UserEntity> getRankingsById(@PathVariable String id);  
174  
175  
176     @GetMapping("/authorized") 1 implementation  
177     Map<String, Object> authorized(@RequestParam(name = "code") String code);  
178  
179     @GetMapping("/login")  
180     ResponseEntity<?> loginByDocument(@RequestParam String document);  
181  
182
```

Creando endpoint

Implementamos el método con su lógica.

```
@Override no usages
public ResponseEntity<?> loginByDocument(String document) {
    var user = iUserService.getByDocument(document);
    if (user.getBody() != null) {
        return ResponseEntity.ok(user);
    }
    return ResponseEntity.notFound().build();
}
```

Implementando WebClientConfig

En el servicio de Auth creamos la clase WebClientConfig, el cual tiene la configuración para el login.

The screenshot shows a dark-themed IDE interface with the following details:

- Title Bar:** ms-autorization – WebClientConfig.java
- Project Tree:** ms-autorization (~/Documents/K8s avanzado)
 - .idea
 - .mvn
 - src
 - main
 - java
 - com.authorization.ms
 - MsAuthorizationApplication
 - SecurityConfig
 - WebClientConfig
 - resources
 - static
 - templates
 - application.properties
 - test
- Code Editor:** WebClientConfig.java
 - Content:

```
1 package com.authorization.ms;
2
3 public class WebClientConfig { no usages
4 }
5
```

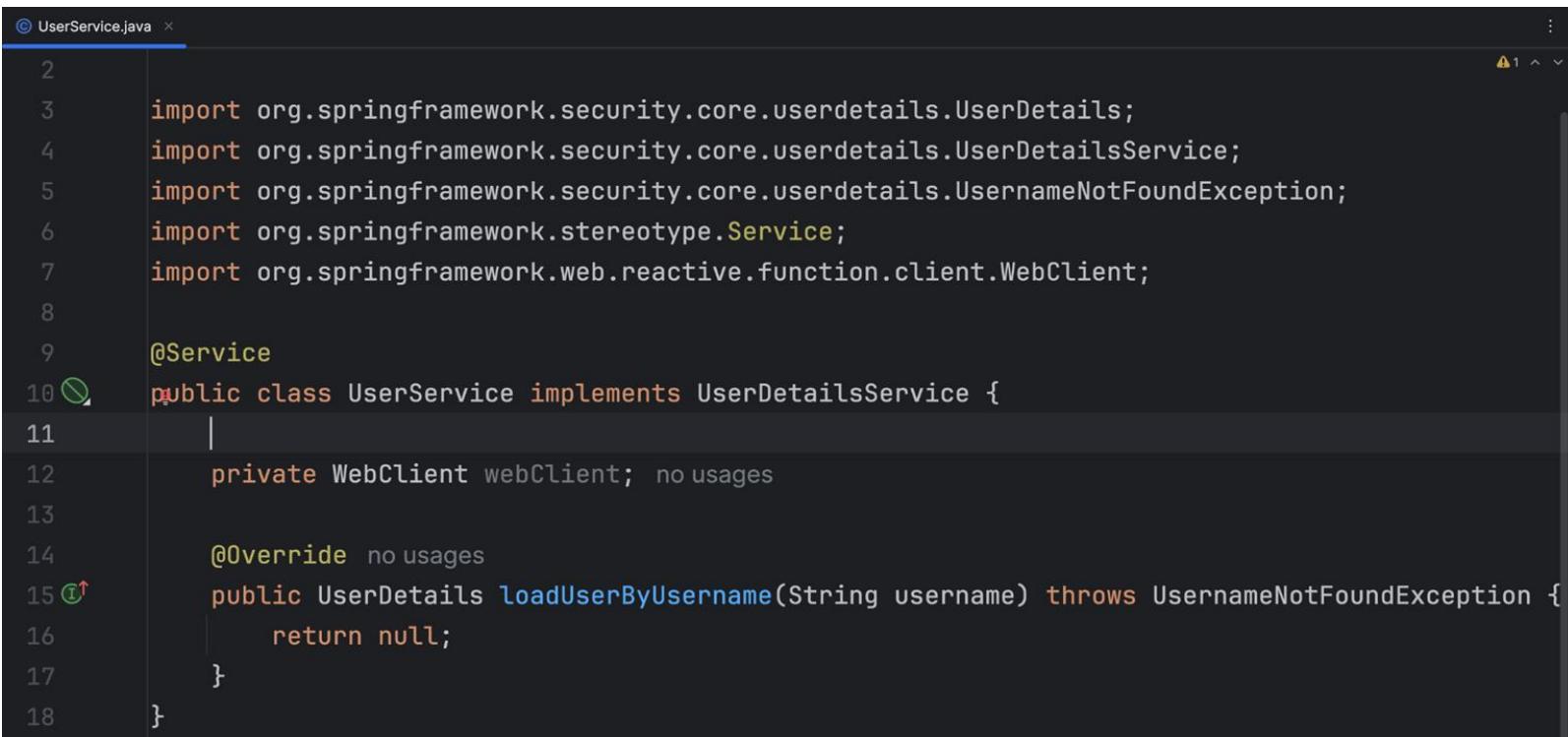
Implementando WebClientConfig

Agregamos la anotación de configuración y el bean para la creación del WebClient.

```
© WebClientConfig.java ×  
1 package com.autorization.ms;  
2  
3 import org.springframework.cloud.client.loadbalancer.LoadBalanced;  
4 import org.springframework.context.annotation.Bean;  
5 import org.springframework.context.annotation.Configuration;  
6 import org.springframework.web.reactive.function.client.WebClient;  
7  
8 @Configuration  
9 public class WebClientConfig {  
10  
11     @LoadBalanced  
12     @Bean  
13     WebClient.Builder webClient() {  
14         return WebClient.builder();  
15     }  
16 }
```

Creando UserService

La clase UserService es utilizada para validar el login en la aplicación desde el microservicio Auth.

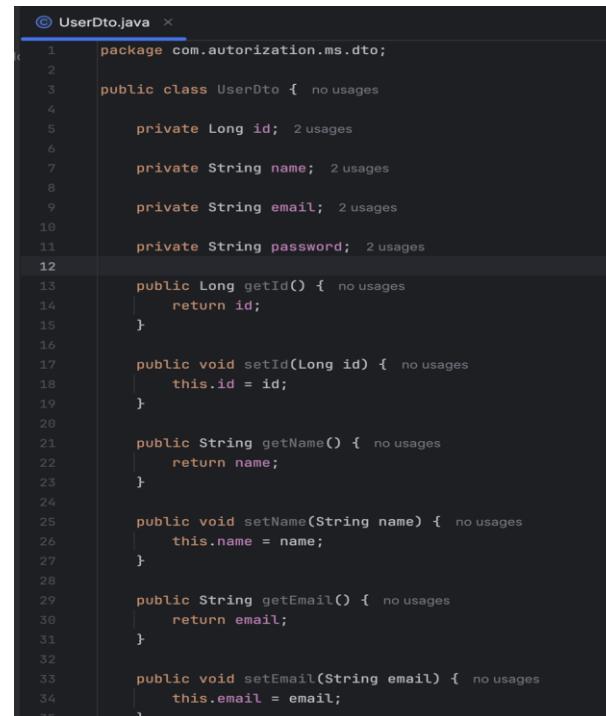


The screenshot shows a code editor window with the title bar "UserService.java". The code is written in Java and defines a service class that implements the UserDetailsService interface. The class uses a WebClient dependency to load user details by username. There is one warning icon in the status bar at the top right.

```
UserService.java
2
3 import org.springframework.security.core.userdetails.UserDetails;
4 import org.springframework.security.core.userdetails.UserDetailsService;
5 import org.springframework.security.core.userdetails.UsernameNotFoundException;
6 import org.springframework.stereotype.Service;
7 import org.springframework.web.reactive.function.client.WebClient;
8
9 @Service
10 public class UserService implements UserDetailsService {
11     |
12     private WebClient webClient; no usages
13
14     @Override no usages
15     public UserDetails loadUserByUsername(String username) throws UsernameNotFoundException {
16         return null;
17     }
18 }
```

3.13. Implementando clase UserDetailsService para login personalizado con WebClient

Primero, crearemos el DTO de cliente para poder realizar las validaciones correctamente. Para esto crearemos una nueva clase llamada UserDto dentro del package dto.



```
 1 package com.autorization.ms.dto;
 2
 3 public class UserDto { no usages
 4
 5     private Long id; 2 usages
 6
 7     private String name; 2 usages
 8
 9     private String email; 2 usages
10
11     private String password; 2 usages
12
13     public Long getId() { no usages
14         return id;
15     }
16
17     public void setId(Long id) { no usages
18         this.id = id;
19     }
20
21     public String getName() { no usages
22         return name;
23     }
24
25     public void setName(String name) { no usages
26         this.name = name;
27     }
28
29     public String getEmail() { no usages
30         return email;
31     }
32
33     public void setEmail(String email) { no usages
34         this.email = email;
35     }
}
```

Configurando UserDetailsService

Agregamos la lógica del servicio basado en el USER de Spring security.

```
@Service
public class UserService implements UserDetailsService {

    private WebClient webClient; no usages

    private WebClient.Builder client; 1 usage

    @Override no usages
    public UserDetails loadUserByUsername(String email) throws UsernameNotFoundException {
        try {

            UserDto user = client
                .build() WebClient
                .get() RequestHeadersUrlSpec<capture of ?>
                .uri( uri: "http://ms-user/login", uri -> uri.queryParam( name: "email", email).build() ) capture of ?
                .accept(MediaType.APPLICATION_JSON)
                .retrieve() ResponseSpec |
                .bodyToMono(UserDto.class) Mono<UserDto>
                .block();

            return new User(email, user.getPassword(), enabled: true, accountNonExpired: true, credentialsNonExpired: true, accountNonLocked: true,
                Collections.singleton(new SimpleGrantedAuthority( role: "ROLE_USER")));
        } catch (RuntimeException e) {
            throw new UsernameNotFoundException(e.getMessage());
        }
    }
}
```

3.14. Aplicando cambios y probando

Cada vez que se realiza un cambio en los microservicios es necesario realizar el despliegue en nuestro minikube para volver a probar.

- (base) danielflorezlopez@192 Capítulo 3 % kubectl apply -f deployment-usuarios.yaml
deployment.apps/msvc-usuarios created
- (base) danielflorezlopez@192 Capítulo 3 % kubectl apply -f auth.yaml
deployment.apps/msvc-auth unchanged
service/msvc-auth unchanged

3.15. WebClient con SpringCloud LoadBalancer

El Load Balancer es un módulo que permite realizar el equilibrio o el balanceo de las peticiones que hacen al servidor, distribuyendo el tráfico al Pod o nodo con menos carga esto para que el servicio pueda ser estable y escalable.

Para esta parte del módulo vamos a configurar el Loan Balancer del Cloud para un mejor rendimiento de la aplicación.

Ajustando servicios

Para garantizar que el servicio UserService funcione correctamente sin especificar explícitamente el puerto en la URL, es necesario configurar al cliente con balanceo de carga. Esto se logra de la siguiente manera:

- En la clase WebClientConfig, agrega la anotación @LoadBalanced al vean del WebClient.Builder. Esto permite que las llamadas a los servicios se gestionen automáticamente a través de un balanceador de carga.

```
6 import org.springframework.web.reactive.function.client.ClientResponse;
7
8 @Configuration
9 public class WebClientConfig {
10
11     @LoadBalanced
12     @Bean
13     WebClient.Builder webClient() {
14         return WebClient.builder();
15     }
16 }
17 }
```

Ajustando servicios

En el método `loadUserByUsername`, utiliza el `WebClient.Builder` configurado para realizar la solicitud a `UserService`, utilizando el nombre lógico del servicio (`http://ms-user/login`) en lugar de especificar un puerto fijo.

```
20  
21     private WebClient.Builder client; 1 usage  
22  
23     @Override no usages  
24     public UserDetails loadUserByUsername(String email) throws UsernameNotFoundException {  
25         try {  
26  
27             UserDto user = client  
28                 .build() WebClient  
29                 .get() RequestHeadersUriSpec<capture of ?>  
30                 .uri( uri: "http://ms-user/login", uri -> uri.queryParam( name: "email", email).build() ) capture of ?  
31                 .accept(MediaType.TEXT_PLAIN) Extract Surround // ⌂ :  
32                 .retrieve() Response<UserDto>  
33                 .bodyToMono(UserDto.class) Mono<UserDto>  
34                 .block();  
35  
36             return new User(email, user.getPassword(), enabled: true, accountNonExpired: true, credentialsNonExpired: true, accountNonLocked: true, authorities: Collections.singleton(new SimpleGrantedAuthority(role: "ROLE_USER")));  
37         } catch (RuntimeException e) {  
38             throw new UsernameNotFoundException(e.getMessage());  
39         }  
40     }  
41 }  
42 }
```

3.16. Propagar token JWT

Para que los servicios que estamos consumiendo soliciten el token o pueda ser validado, es necesario especificar en los servicios el `@RequestHeader` y que sea requerido.

```
@PostMapping("/example")
public void getAllInformation (@RequestHeader(value = "Authorization", required = true) String jwt) {
}
```

3.17. Login OAuth2 Cliente Http Postman

Una vez realizadas todas las configuraciones, ejecutamos la validación con postman.

Es necesario tener en cuenta que para esta prueba los microservicios deben estar desplegados en el Kubernetes.

Login OAuth2 Cliente Http Postman

The screenshot shows the Postman application interface. At the top, the URL `http://172.53.20.223/auth/create` is entered in the address bar, and the method is set to `POST`. Below the header, the `Headers` tab is selected, showing a table with one row and two columns: `Key` and `Value`. The table displays the key `Key` and the value `Value`. In the bottom section, the `Body` tab is selected, showing the response status `200 OK`, response time `166 ms`, and response size `483 B`. The response body contains the token `$2a$10$mszJpUggI2t7RsaCWLFXs.0bcRzTl5btEGpzMRPlr8kh2zNFGks0a`.

Key	Value
Key	Value

Body Cookies Headers (14) Test Results 200 OK • 166 ms • 483 B • [e.g.](#) • [...](#)

Pretty Raw Preview Visualize Text ▾

1 \$2a\$10\$mszJpUggI2t7RsaCWLFXs.0bcRzTl5btEGpzMRPlr8kh2zNFGks0a

Resumen del capítulo

- Implementación de Spring Security.
- Creación y generación de Dockerfile.
- Construcción de YAML para microservicios.
- Despliegue de microservicios en minikube.



Práctica 3. Microservicio de seguridad

Objetivo:

- Crear un Microservicio dedicado a la seguridad y permita la creación de usuarios
- Manejo de JWT.
- Crear un microservicio basado en Spring Security para la gestión de usuarios con una base de datos en la memoria.

Planteamiento e instrucciones:

Realiza un seguimiento de los pasos indicados en la *Guía de Laboratorios* para llevar a cabo la tarea correspondiente en el siguiente enlace: https://netec-mx.github.io/DOCK_KUB_AVA/



Tiempo para esta actividad:
80 minutos.

Resultado esperado

The screenshot shows the Postman application interface. At the top, it displays a POST request to `http://localhost:8084/auth/create`. The request method is set to POST, and the URL is `http://localhost:8084/auth/create`. The Body tab is selected, showing the following JSON payload:

```
1 {
2   "userName": "daniel",
3   "password": "1234",
4   "role": "admin"
5 }
```

Below the request, the response section is visible. It shows a status of 200 OK, a time of 456 ms, and a size of 492 B. The response body is displayed in Pretty format:

```
1 {
2   "id": 1,
3   "userName": "daniel",
4   "password": "$2a$10$NKpS75rk058ozHnPNm80p0uj21p6lwghZ2TIM16AcNPIe7Wkw8kAq",
5   "role": "admin"
6 }
```

Referencias Bibliográficas

- [OAuth 2.0 – Oauth](#)
- [JSON Web Tokens - jwt.io](#)
- [Spring Security](#)





Objetivos:

- Comprender la arquitectura interna que tiene Kubernetes.

Capítulo 4

Kubernetes API

4.1. Kube-Scheduler y recursos

Cuando ingresa una petición por medio del API o kubectl, Kubernetes se comunica con Scheduler con las especificaciones que solicito el usuario para el despliegue de un Pod. El Scheduler valida el nodo que cuenta con los recursos para ubicar el Pod de una forma óptima y correcta.

En dado caso que el Scheduler no logre encontrar un nodo con los recursos necesarios, pasa la tarea a modo espera hasta que un nodo escale o esté disponible.

4.2. Kube-Controller

En la arquitectura de Kubernetes el kube-controller es el encargado de verificar que los nodos estén disponibles según la configuración previa, por ejemplo:

El kube-controller monitoriza que siempre están disponibles 3 réplicas, en dado caso que un Pod se llegue a caer, el controller automáticamente levanta un nuevo Pod para que el Kubernetes cuente con los 3 nodos iniciales.

4.3. ETCD: Base de datos de Kubernetes

ETCD es una base de datos clave-valor, esta base de datos es utilizada en Kubernetes para guardar las configuraciones, estados y contenedores que se usan en los Pod.

Una ventaja que tiene k8s al usar ETCD es que si se necesita cambiar entre versiones o volver a una versión anterior lo puede hacer más fácil ya que ETCL tiene los estados anteriores de los Pod.

4.4. Kubelet

Kubelet es un servicio que se está ejecutando en cada una de las máquinas o clústers de Kubernetes, para escuchar las instrucciones del master y retornar información del máster.

Kubelet también es el encargado de informarle al master el estado de salud o recursos disponibles.

4.5. Kube-Proxy

Kube-Proxy es un servicio que se está ejecutando en cada clúster y tiene la responsabilidad de la administración de la red para gestionar correctamente las réplicas de cada Pod para que se comuniquen entre ellos. Kube-Proxy cuenta con las siguientes características:

- Balanceo de carga.
- Manejo de reglas de red.
- Compatibilidad con múltiples protocolos de red.

4.6. Container Runtime

En cada clúster esclavo de Kubernetes, es necesario tener instalado Docker esto para que el agente de Container Runtime se encargue de realizar la creación de los contenedores de Docker que van a tener los Pods para lograr una alta disponibilidad.

- Gestión de contenedores.
- Aislamiento de recursos.

4.7. Kubernetes: Administración de storage y recursos de asignación

Kubernetes gestiona el almacenamiento y asignación de recursos para aplicaciones. Permite almacenar datos de forma persistente con Persistent Volumes y configurar límites de CPU y memoria para cada contenedor, asegurando que los recursos se utilicen eficientemente y se mantenga la estabilidad y escalabilidad del sistema.

4.8. Kubernetes troubleshooting and debugging

Cuando se realiza la administración de Kubernetes, es importante conocer el estado de los Pods, configuración y demás, para esto los comandos más básicos son:

- `kubectl get pods`: Muestra todos los Pods en el clúster junto con su estado.
- `kubectl describe Pod <pod-name>`: Lista información detallada sobre un Pod específico.
- `kubectl logs <pod-name>` : Muestra los registros de un contenedor en un Pod.

Resumen del capítulo

- Comprender la arquitectura interna en Kubernetes.
- Comprender cómo Kubernetes administra y gestiona sus nodos para garantizar la alta disponibilidad.



Práctica 4. Arquitectura K8s

Objetivo:

- Realizar un diagrama de arquitectura interna de Kubernetes.
- Realizar un diagrama de los componentes de Kubernetes y entender la responsabilidad de cada módulo.

Planteamiento e instrucciones:

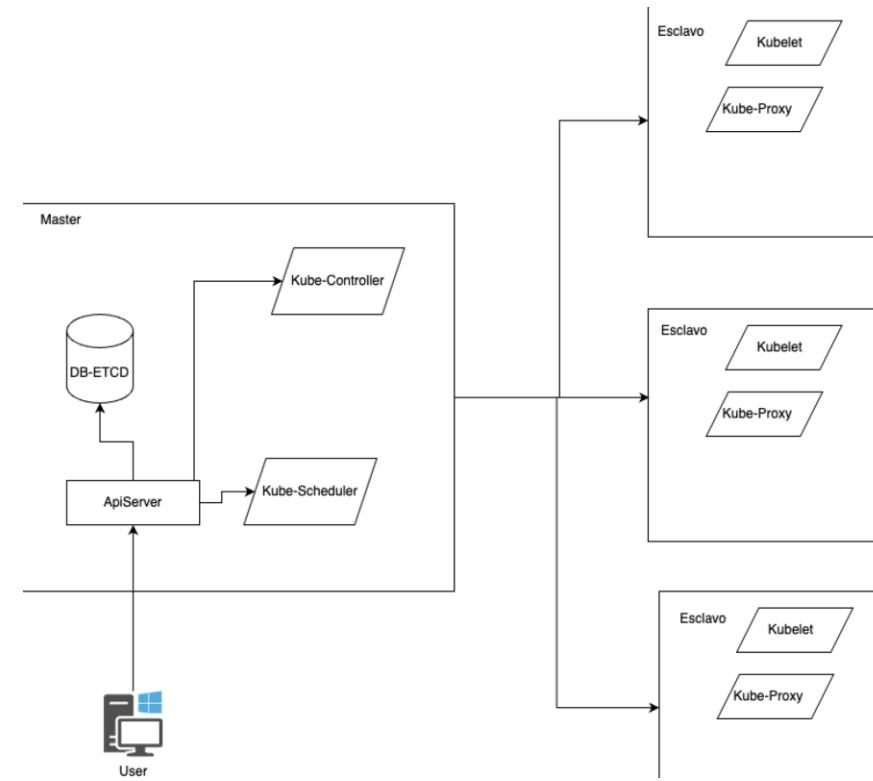
Realiza un seguimiento de los pasos indicados en la *Guía de Laboratorios* para llevar a cabo la tarea correspondiente en el siguiente enlace: https://netec-mx.github.io/DOCK_KUB_AVA/



Tiempo para esta actividad:
20 minutos.

Resultado esperado

Nota: En una arquitectura de Kubernetes que sugiere que la cantidad de maestros sean pares y de esclavos impares



Referencias Bibliográficas

- [Concepts | Kubernetes](#)
- [Concepts | Kubernetes](#)
- [Arquitectura de Kubernetes | Kubernetes](#)



Objetivos:

- Administrar correctamente Pods desde la terminal.
- Comprender las estrategias de seguridad, monitoreo, buenas prácticas en Kubernetes.

Capítulo 5

Explorando Pods

5.1. Creación y eliminación de Pods

Pod representa una instancia de una aplicación o de una tarea específica en el clúster y puede contener uno o varios contenedores que comparten el mismo espacio de red y recursos.

Los Pods se utilizan para ejecutar instancias de aplicaciones o servicios de forma aislada y gestionada.

Iniciando minikube

Para iniciar Kubernetes en la máquina local se ejecuta el comando “minikube start”.

```
(base) danielflorezlopez@192 ~ % minikube start
😊  minikube v1.34.0 on Darwin 15.0.1 (arm64)
✨  Using the docker driver based on existing profile
👍  Starting "minikube" primary control-plane node in "minikube" cluster
🚜  Pulling base image v0.0.45 ...
🔄  Restarting existing docker container for "minikube" ...
🐳  Preparing Kubernetes v1.31.0 on Docker 27.2.0 ...
🔍  Verifying Kubernetes components...
  - Using image gcr.io/k8s-minikube/storage-provisioner:v5
⭐️  Enabled addons: storage-provisioner, default-storageclass
🚀  Done! kubectl is now configured to use "minikube" cluster and "default" namespace by default
(base) danielflorezlopez@192 ~ %
```

Creación Pod

En Kubernetes el comando que se utiliza para la creación de despliegue es el comando run, con este comando podemos basarnos para la creación de un Pod en un YAML o en una imagen específica. Para crear el Pod de forma básica seguimos la siguiente estructura :

- kubectl run <namepod> --image=<name-imagen>:<version>

```
[base) danielflorezlopez@192 ~ % kubectl run podtest --image=nginx:alpine
pod/podtest created
(base) danielflorezlopez@192 ~ %
```

Verificar Pod

Para listar los Pods que se tienen activos en Kubernetes usamos el comando:

- kubectl get pods

```
[(base) danielflorezlopez@192 ~ % kubectl get pods
  NAME      READY     STATUS    RESTARTS   AGE
  podtest   1/1      Running   0          5m46s]
```

Detalle de un Pod

Es importante conocer el log de los Pods para saber detalles como errores o información general.

- `kubectl describe pod <pod>`

```
(base) danielflorezlopez@192 ~ % kubectl describe pod podtest
Name:          podtest
Namespace:     default
Priority:      0
Service Account: default
Node:          minikube/192.168.49.2
Start Time:    Sat, 02 Nov 2024 08:09:57 -0500
Labels:        run=podtest
Annotations:   <none>
Status:        Running
IP:           10.244.0.7
IPs:
  IP: 10.244.0.7
Containers:
  podtest:
    Container ID: docker://097b022ef3cc9eca2726ffffdb29ce224bc5e8b6a3608f662196648a4bf0600d2
    Image:         nginx:alpine
    Image ID:     docker-pullable://nginx@sha256:2140dad235c130ac861018a4e13a6bc8aea3a35f3a40e20c1b060d51a7efd250
    Port:         <none>
    Host Port:   <none>
    State:       Running
    Started:    Sat, 02 Nov 2024 08:10:02 -0500
    Ready:       True
    Restart Count: 0
    Environment: <none>
    Mounts:
      /var/run/secrets/kubernetes.io/serviceaccount from kube-api-access-vmqp8 (ro)
Conditions:
  Type          Status
  PodReadyToStartContainers  True
  Initialized    True
  Ready          True
  ContainersReady  True
  PodScheduled   True
Volumes:
  kube-api-access-vmqp8:
    Type:          Projected (a volume that contains injected data from multiple sources)
    TokenExpirationSeconds: 3607
    ConfigMapName:  kube-root-ca.crt
    ConfigMapOptional: <nil>
    DownwardAPI:   true
    QoS Class:    BestEffort
    Node-Selectors: <none>
    Tolerations:   node.kubernetes.io/not-ready:NoExecute op=Exists for 300s
                   node.kubernetes.io/unreachable:NoExecute op=Exists for 300s
Events:
  Type  Reason  Age  From            Message
  ----  ----   --  --              --
  Normal Scheduled 10m  default-scheduler  Successfully assigned default/podtest to minikube
  Normal Pulling  10m  kubelet        Pulling image "nginx:alpine"
  Normal Pulled   10m  kubelet        Successfully pulled image "nginx:alpine" in 4.857s (4.857s including waiting). Image size: 50841357 bytes.
  Normal Created  10m  kubelet        Created container podtest
```

Eliminar Pod

Para la eliminación de un Pod usamos el comando :

- kubectl delete pod <nombre>

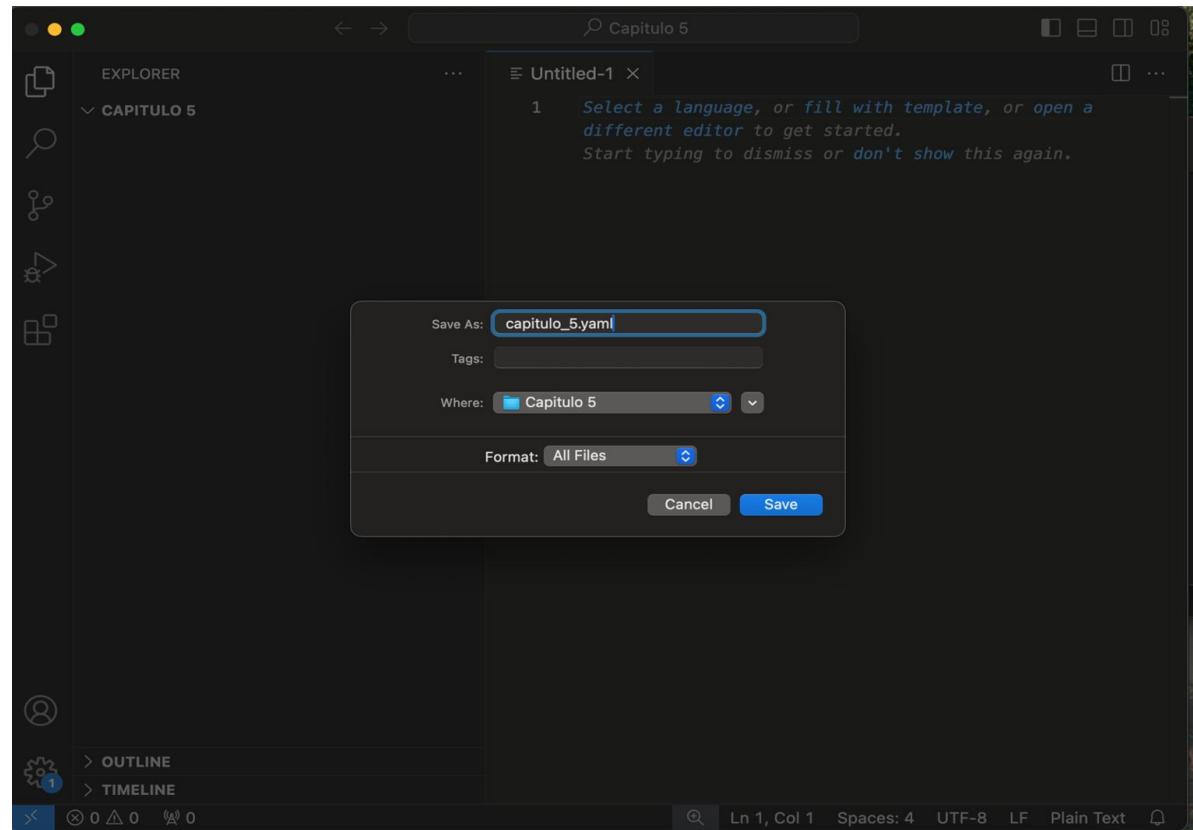
```
(base) danielflorezlopez@192 ~ % kubectl delete pod podtest  
pod "podtest" deleted
```

5.2. Manifiestos de Kubernetes

- Un manifiesto en Kubernetes es un archivo YAML en el cual se definen los recursos que queremos actualizar o crear en Kubernetes.
- Se sugiere usar los manifiestos porque podemos crear múltiples recursos de una forma más organizada y agrupada o administrarlo de una forma mas sencilla.

Creación del manifiesto

Para crear un manifiesto de Kubernetes solo basta con crear un archivo con la extensión .yaml.



Conceptos de un manifiesto

El YAML o manifiesto está compuesta por las siguientes secciones:

- **apiVersion:** Especifica la versión de la API de Kubernetes que se está usando para el recurso.
- **kind:** Define el tipo de recurso que se está creando, como Pod, Service, Deployment, entre otros.
- **metadata:** Información adicional sobre el recurso, como su nombre y labels.
 - Los labels ayudan a clasificar y organizar el recurso.
 - Las anotaciones son para datos complementarios de los usuarios o información adicional.

Conceptos de un manifiesto

- **spec**: Es la sección donde se especifican las configuraciones detalladas del recurso, como la configuración de los contenedores dentro de un Pod.
- **containers**: Lista de contenedores que se ejecutarán dentro del recurso.
- **name**: Nombre del recurso o contenedor, utilizado para identificarlo de forma única dentro del namespace en el clúster.
- **image**: Especifica la imagen del contenedor que se usará para este recurso.
- **ports**: Define los puertos que serán expuestos por los contenedores para la comunicación externa o interna dentro del clúster.

Creación del manifiesto

Una vez creado el archivo .yaml, procedemos con la creación de su contenido.

```
! capítulo_5.yaml ●  
! capítulo_5.yaml  
5   metadata:  
6     name: manifiesto-k8s  # Nombre del Pod, único dentro del espacio de nombres (namespace)  
7     labels:      # Etiquetas que ayudan a clasificar y seleccionar el recurso  
8       app: capítulo-5  
9  
10    # Definición de las especificaciones del Pod  
11    spec:  
12      containers:    # Lista de contenedores que se ejecutarán en este Pod  
13        - name: contenedor-1 # Nombre del contenedor en el Pod  
14          image: nginx           # Imagen de contenedor que se utilizará (en este caso, nginx)  
15
```

Creación del nuevo Pod

Para crear el nuevo Pod con ayuda del manifiesto debemos ubicarnos en la ruta en la cual está el manifiesto y ejecutar el siguiente comando:

- `kubectl apply -f <nombre_manifiesto.yaml>`

```
● (base) danielflorezlopez@192 Capitulo 5 % kubectl apply -f capitulo_5.yaml
pod/mi-pod created
○ (base) danielflorezlopez@192 Capitulo 5 % clear
● (base) danielflorezlopez@192 Capitulo 5 % kubectl get pods
  NAME      READY   STATUS    RESTARTS   AGE
  mi-pod    1/1     Running   0          37s
○ (base) danielflorezlopez@192 Capitulo 5 % █
```

Eliminar por Pod con manifiesto

Para eliminar el Pod con ayuda del manifiesto debemos ubicarnos en la ruta en la cual está el manifiesto y ejecutar el siguiente comando:

- `kubectl apply -f <nombre_manifiesto.yaml>`

```
● (base) danielflorezlopez@192 Capitulo 5 % kubectl delete -f capitulo_5.yaml
pod "mi-pod" deleted
● (base) danielflorezlopez@192 Capitulo 5 % kubectl get pods
No resources found in default namespace.
○ (base) danielflorezlopez@192 Capitulo 5 %
```

5.3. Pods con más de un contenedor

Una gran ventaja que tiene el manifiesto es la creación de una configuración completa y organizada para administrar múltiples recursos. Con ayuda de este manifiesto podemos crear N contenedores solo adicionando un par de líneas adicionales.

Solo hace falta agregar en el spect un nuevo contenedor en la sección containers:

```
containers
- name: <nombre>
  image: <nombre_imagen>
- name: <nombre>
  image: <nombre_imagen>
```

Ejemplo multiple containers

```
! capítulo_5.yaml ×  
! capítulo_5.yaml  
7  
8   spec:  
9     containers:  
10    - name: contenedor-1  
11      image: nginx  
12      ports:  
13        - containerPort: 80  
14  
15    - name: contenedor-2  
16      image: mysql:latest  
17      env:  
18        - name: MYSQL_ROOT_PASSWORD  
19          value: "rootpassword"  
20      ports:  
21        - containerPort: 3306
```

Ejemplo multiple containers

Para crear los nuevos containers en Kubernetes volvemos a utilizar el comando visto en la sección anterior.

```
(base) danielflorezlopez@192 Capitulo 5 % kubectl apply -f capitulo_5.yaml
pod/manifiesto-k8s created
(base) danielflorezlopez@192 Capitulo 5 % kubectl get pod
NAME           READY   STATUS    RESTARTS   AGE
manifiesto-k8s  2/2     Running   0          5s
(base) danielflorezlopez@192 Capitulo 5 % █
```

5.4. End-To-End DEVSECOPS Kubernetes Project

Kubernetes es una aplicación que ayuda a desplegar las aplicaciones en un entorno de alto rendimiento, sin embargo, podemos realizar una integración completa y automática desde el inicio del desarrollo hasta el despliegue automático en la infraestructura de Kubernetes pasando por buenas prácticas de seguridad de DevSecOps.

Ciclo vida DevSecOps en k8s

- **Planificación:** Se define el objetivo. Requerimientos tanto de negocio como de seguridad para las normativas a aplicar.
- **Desarrollo:** Construcción de reglas de seguridad para el pipeline e integración de herramientas como SonarQube o Checkmarx.
- **Implementación GIT:** Implementación de reglas en GIT para el manejo de ramas o GitOps para el despliegue automático.
- **Escaneo de imágenes:** Por medio de la herramienta Trivy o Clair se realiza un análisis a la imagen de Docker para buscar vulnerabilidades.
- **Despliegue:** Se realiza la puesta en producción para las aplicaciones una vez pasen los filtros de seguridad.

5.5. ¿Qué es DevSecOps?

Es un marco de trabajo que ha venido siendo adoptado por varias compañías con el fin de integrar la seguridad en cada fase del desarrollo del software y el día a día de las operaciones.

Lo que se busca con DevSecOps es no dejar como última etapa del desarrollo la seguridad, si no que desde el inicio se lleven buenas prácticas.

5.6. Seguridad del entorno y de los datos

Los datos son el componente más importante de una compañía, ya que estos datos son el corazón de la organización, por esta razón es importante identificar cuales son los datos sensibles y aplicar un cifrado.

Los datos tienen dos estados:

- **Reposo:** Un dato en reposo es cuando está en la base de datos, un almacén como S3 , EFS o EBS.
- **Tránsito:** cuando los datos están siendo transportados de un punto a otro se dice que es un dato en tránsito.

Cifrado de datos en reposo

AWS ofrece estrategias para proteger nuestros datos en reposo para mayor seguridad, una de estas es:

- KMS(AWS Key Management Service): Permite la creación y gestión de claves de cifrado.
- SSE (S3 server.side Encryption).
 - SSE-S3 : Cifra automáticamente AWS os objetos almacenados en S3.
 - SSE-KMS: Utiliza KMS para la administración de las KEY en S3.
 - SSE-C: El cliente Gestiona su propia llave.

Cifrado de datos en tránsito - AWS

- TLS/SSL: Certificado que cifra los datos que se encuentran en tránsito entre el cliente y AWS.
- AWS VPN: Permite establecer una conexión segura y cifrada desde una red local hasta AWS.
- AWS Direct Connect: Ofrece una conexión directa desde las instalaciones del cliente hasta las zonas de disponibilidad de AWS sin ser expuesta por internet.

5.7. Seguridad del proceso CI/CD

- **Control de acceso y autenticación:** Crear usuarios con el mínimo privilegio en la herramienta y no usar una cuenta genérica o admin.
- **Análisis de código estático:** Implementar una herramienta que permite encontrar vulnerabilidades o reglas del código como cobertura de pruebas unitarias, código duplicado o métodos sin usar.
- **Pruebas de seguridad dinámica:** Realizar un monitoreo constante en búsqueda de vulnerabilidades en el código, imágenes, infraestructura o pruebas automáticas.

5.8. Tools (GitLab, GitHub, AWS, Jenkins)

En el desarrollo de software es importante integrar herramientas para la automatización del ciclo de vida como, el almacenamiento de código, CI/CD, integración con el Cloud, entre otros.



GitLab - GitHub

Plataforma de gestión de repositorio de código fuente que permite trabajar con varios integrantes por medio de ramas e integraciones de código. Estos repositorios cuentan con las siguientes características:

- Revisión de código: Crea un proceso de revisión y aprobación de código por parte de los integrantes del equipo.
- Ramas: Permite la creación de copias del código a partir de una main para un mejor control y trabajo individual o separación de ambientes como producción, desarrollo o QA.

Amazon Web Service

AWS es un servicio mundial en la nube con variedad de servicios para crear infraestructura IaaS PaaS ,SaaS, FaaS. Sus características principales son :

- Global a un clic: Permite desplegar tus aplicaciones en cualquier parte del mundo.
- Seguridad: Ofrece un sin fin de servicios para administrar tus datos de forma segura como KMS , TLS, IAM.
- Escalabilidad: Permite que tus servicios escalen automáticamente o a tu necesidad.
- Pagar por lo que usas: Su modelo de facturación es que solo se factura cuando tienes un servicio iniciado o el almacenamiento que estés usando.

Jenkins

Es un software o herramienta que por medio de pipelines permite automatizar el ciclo de vida del software en CI/CD. Permite agilizar el marco de trabajo de DevOps.

- Automatización: Creación de Scripts, procesos o pipeline para aplicar procesos de CI/CD.
- Distribución de cargas: Puede trabajar con múltiples clústeres para la gestión de tareas de despliegue.
- Plugins: Integración con múltiples plugin para realizar configuraciones o conexiones con distintos servicios como AWS , Azure entre otros.

5.9. Prometheus, Node Exporter, Grafana, Sonarqube, OWASP, Kubernetes

A lo largo del proceso de CI/CD es importante tener un monitoreo constante en cada fase para tener certeza del estado de salud de la aplicación, alta disponibilidad, validaciones de reglas de buenas prácticas de código y poder rastrear los logs en caso de una falla o errores del proceso.

Prometheus

Prometheus es la plataforma que se utiliza para el monitoreo, registro de logs, creación de reglas o recopilación de métricas de una aplicación en tiempo real. Es ideal cuando manejamos una arquitectura de microservicios.

Características

- **PromQL:** Por medio de un lenguaje básico permite crear reglas para el filtro de métricas y logs.
- **Alertas:** Permite crear alertas personalizadas sobre el estado del servidor o registro de errores.

Grafana

Plataforma que permite visualizar y monitorear una aplicación por medio de gráficos, tableros personalizados y alertas en una interfaz intuitiva y fácil de usar.

Características:

- Personalización: Permite crear tableros o gráficos con base a la necesidad del negocio.
- Alertas: Permite la creación de alertas en tiempo real.
- Panel de control: Creación de paneles de control personalizados.

SonarQube

Herramienta que permite realizar un análisis al código estático para validar la calidad, seguridad, buenas prácticas, cobertura de código, encontrar problemas de código duplicado y sugerencias de mejoras.

Características

- Análisis calidad de código: Por medio de reglas evalúa el estado del código permitiendo que el producto final sea confiable.
- Multilenguaje: Tiene compatibilidad de análisis con mas de 30 lenguajes de programación diferentes.
- Integracion CI: Puede ser implementado en el proceso de CI para evaluar el código estático.

5.10. SDLC: Desarrollo seguro de aplicaciones

SDLC (El Ciclo de Vida de Desarrollo de Software), se encarga de aplicar buenas prácticas de seguridad en cada etapa del desarrollo de software hasta la puesta a producción, esto con el objetivo de mitigar las vulnerabilidades de seguridad.



5.11. Performance y Troubleshooting

Una aplicación correcta de Performance y Troubleshooting es importante para garantizar que el producto o aplicación funcione correctamente y de manera eficiente, o sin interrupciones, ya que es importante tener una alta disponibilidad ante los usuarios finales y lograr un buen desempeño de la misma.

Definición Performance y Troubleshooting

- **Performance:** Se enfoca en el monitoreo y optimización de los recursos que se encuentra usando un clúster tanto en memoria, CPU, red y capacidad, esto para que el sistema funcione correctamente y responda en tiempos óptimos.
- **Monitoreo en tiempo real:** Se sugiere usar herramientas como Grafana , Cloud Watch en AWS para ver en tiempo real el estado de los servicios o los clústers
- **Troubleshooting:** Se encarga de identificar, diagnosticar y resolver problemas de rendimiento o de la aplicación
- **Identificar errores:** Revisión de logs para encontrar fallas.
- **Analisis:** Encontrar la causa raíz del error para ser corregido y ajustado.

5.12. Incluir herramientas de monitoreo tanto de los contenedores como del clúster

Supervisar clústeres y contenedores es esencial para mantener el rendimiento y la estabilidad de aplicaciones en entornos basados en contenedores, como Kubernetes. Aquí se describen algunas de las herramientas más comúnmente empleadas para esta tarea:

- Prometheus
- Grafana
- ELK
- cAdvisor

Resumen del capítulo

- Comprender la importancia de usar DevSecOps a lo largo del proceso de software para garantizar la seguridad.
- Configuración de un manifiesto y gestión de los Pods.
- Herramientas para dar monitoreo a la aplicación y a la infraestructura.



Práctica 5. Múltiples contenedores

Objetivo:

- Realizar el despliegue de dos bases de datos en un solo manifiesto.
- Crear un manifiesto en Kubernetes que permita desplegar múltiples servicios.

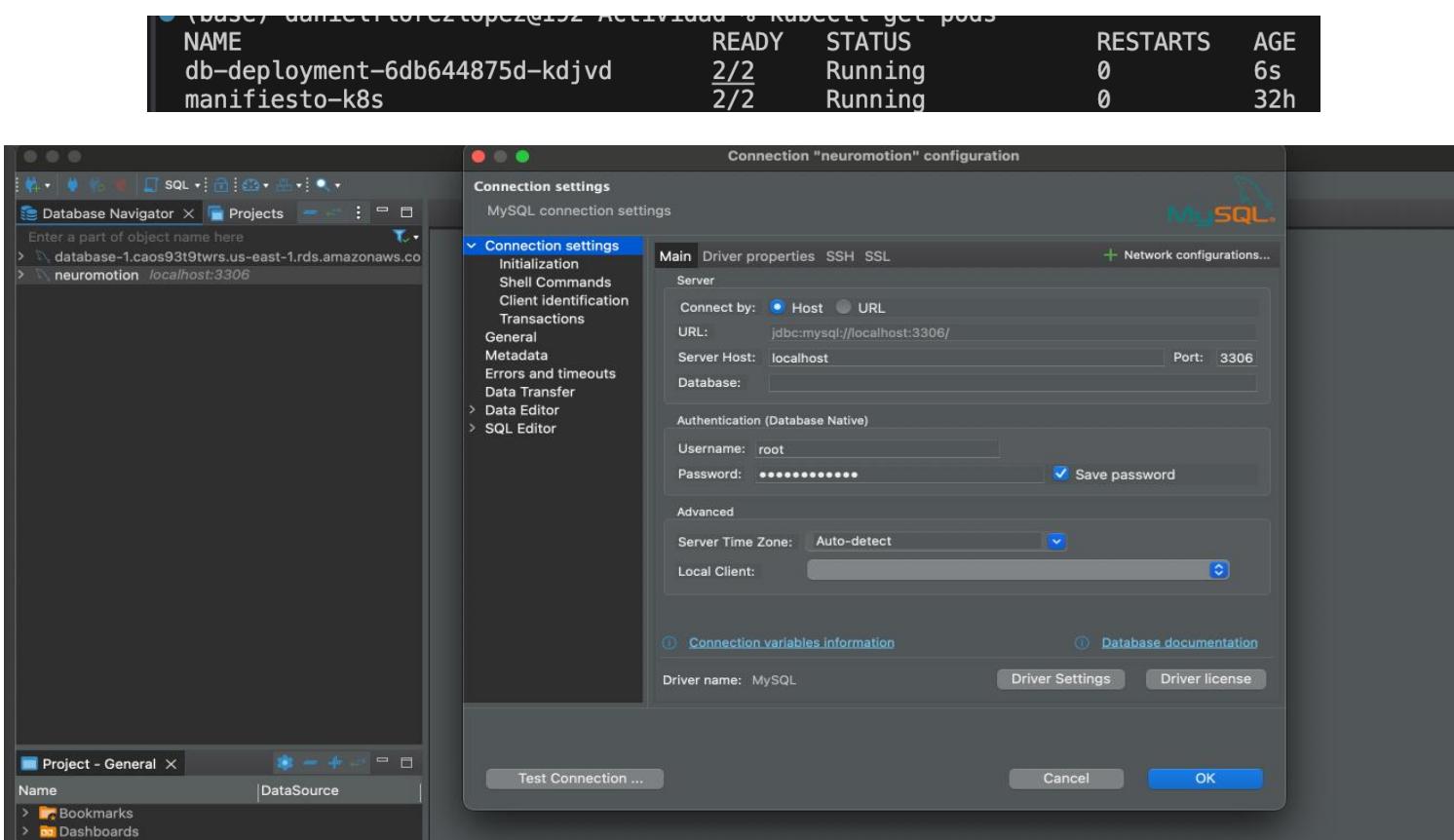
Planteamiento e instrucciones:

Realiza un seguimiento de los pasos indicados en la Guía de Laboratorios para llevar a cabo la tarea correspondiente en el siguiente enlace: https://netec-mx.github.io/DOCK_KUB_AVA/



Tiempo para esta actividad:
45 minutos.

Resultado esperado



Referencias Bibliográficas

- ¿Qué es DevSecOps?
<https://aws.amazon.com/es/what-is/devsecops/>
- Kubernetes Monitoring
<https://grafana.com/docs/grafana-cloud/monitor-infrastructure/kubernetes-monitoring/>
- METRIC AND LABEL NAMING
<https://prometheus.io/docs/practices/naming/>

Glosario

- **CI/CD:** Ciclo de Integración Continua y Entrega Continua, metodologías para automatizar pruebas y despliegue de aplicaciones.
- **Elastic Kubernetes Service (EKS):** Servicio administrado de Kubernetes en AWS para desplegar aplicaciones de contenedores en la nube.
- **Helm:** Herramienta de gestión de paquetes para Kubernetes que simplifica el despliegue y administración de aplicaciones en contenedores.
- **Pods:** Unidad básica de ejecución en Kubernetes, que puede contener uno o más contenedores.
- **OAuth 2.1 y JWT:** Protocolos de autorización y autenticación utilizados para asegurar la comunicación entre servicios mediante tokens de acceso.

¡Gracias!

¿Dudas o comentarios?

