

DOC TÉCNICO DE IMPLEMENTACIÓN Y GESTIÓN DE CI/CD E INFRAESTRUCTURA PARA EL APLICATIVO WEB “FOOTPASSION”



Promotor: Alejandro Iglesias Carpintero

Cliente: SportsMedia

Proveedor cloud: AWS

IES DE TEIS - 2023

ÍNDICE

NOTAS INICIALES.....	3
SU APLICACIÓN.....	4
INTRO.....	4
BACKEND.....	4
BASE DE DATOS.....	6
DESARROLLO DE LA INFRAESTRUCTURA.....	7
CREACIÓN DE LA IMAGEN DOCKER A USAR.....	8
INSTALACIÓN DE LA CLI DE AMAZON EN UBUNTU 22.04.....	12
HOST BASTIÓN.....	13
BASE DE DATOS.....	14
ALMACENAMIENTO PARA LOS CONTENEDORES.....	20
CLÚSTER DE CONTENEDORES DE APLICACIÓN.....	25
IMAGEN AL REPOSITORIO.....	25
CREACIÓN DEL CLÚSTER Y DEFINICIÓN DE TAREAS.....	27
SISTEMA DE CI-CD.....	38
REPOSITORIO DE GIT.....	38
SUBIDA DEL CÓDIGO A GIT.....	38
GITHUB ACTIONS.....	41
PLAN DE CONTROL DE MODIFICACIONES Y MANTENIMIENTO.....	49
WEBGRAFÍA.....	50

NOTAS INICIALES

La cuenta a utilizar será la mía personal, que no utilizaba anteriormente ningún recurso. Desde este momento se hará como si la cuenta raíz fuera la empresa y el usuario de IAM (Identity Access Management) que se usará habitualmente será alex, que sería el administrador. A partir de ahí el hecho de que la cuenta sea mía no tiene relevancia en las aplicaciones prácticas de este trabajo.

Todo este proyecto supone una casuística ficticia con empresas ficticias. El único fin de ello es crear un contexto propio para el entendimiento de cómo éste proyecto se aplicaría en el mundo real.

Por ello también la creación de las máquinas y entornos solo sirven de ejemplo, pues la operatividad a lo largo del tiempo se hará con todos los servicios mencionados pero dentro del alcance de la [capa gratuita](#). Ejemplo:

- La base de datos inicialmente se crea con un despliegue multi-AZ y 8 núcleos de procesador con 200 GB. Dado que esto me supondría un coste personal enorme se ha cambiado por otra dentro de la capa gratuita

A efectos prácticos la funcionalidad final es la misma y la previsión de costes se haría con el entorno supuesto de producción.

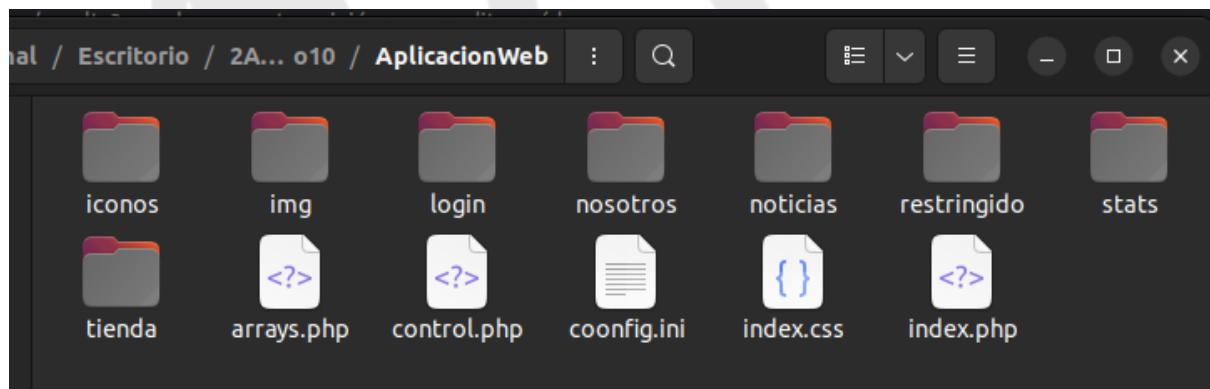
SU APLICACIÓN

INTRO

Se trata de una interfaz muy completa que contempla el manejo de noticias y estadísticas, así como un portal web.

Los datos más relevantes para nuestra infraestructura es que la aplicación funciona con php 7.2, mysql 8.0, y los entornos locales han sido probados y testeados sobre apache, por lo que estas son las especificaciones técnicas que se tendrán que contemplar.

BACKEND



En la foto superior encontramos una visión general de la estructuración de los archivos de la aplicación, en el que destacan los siguientes archivos y carpetas:

- arrays.php : Contiene las consultas que se realizan a la base de datos
- control.php : Todo el control de modificación de datos e interacción de la base de datos se realizan desde este fichero
- config.ini : Contiene las credenciales con las que se acceden a la base de datos.
- index.css : Conjunto de todo el código css utilizado por las diferentes páginas.
- iconos : Una carpeta con los favicon que utiliza la aplicación para mostrar en el navegador u cuando se añade a favoritos.
- img : La carpeta donde se montará el volumen que contiene las fotos que irán variando a lo largo del tiempo.

Se muestra a continuación la estructura de directorios y archivos de la aplicación sin contar con el volumen de imágenes y la carpeta de iconos:

```
ale@aleubu:~/Escritorio/trabaj/aplicacionprueba (copia)$ tree
.
├── arrays.php
├── control.php
├── config.ini
├── index.css
├── index.php
└── login
    ├── login.php
    ├── olvido.php
    ├── registrado.php
    ├── registrarse.php
    └── style.css
├── nosotros
    ├── nosotros.php
    └── term-y-con.php
├── noticias
    ├── noticiasext.php
    └── noticias.php
└── restringido
    ├── admnliga
    │   ├── equiposins.php
    │   ├── equipos.php
    │   ├── golesins.php
    │   ├── infopartido.php
    │   ├── jugadoresins.php
    │   ├── jugadores.php
    │   ├── partidosins.php
    │   ├── partidos.php
    │   └── tarjetasins.php
    ├── introducir
    │   ├── misnoticias.php
    │   ├── misproductos.php
    │   ├── noticiains.php
    │   └── productosins.php
    ├── resindex.php
    ├── sinpermiso.php
    └── term-y-con.php
├── stats
    ├── clasificacion.php
    ├── infopartido.php
    ├── jugadores.php
    ├── partidos.php
    ├── rankings.php
    ├── stats.php
    └── tarjetas.php
└── tienda
    ├── carrito.php
    ├── producto.php
    └── tienda.php
```

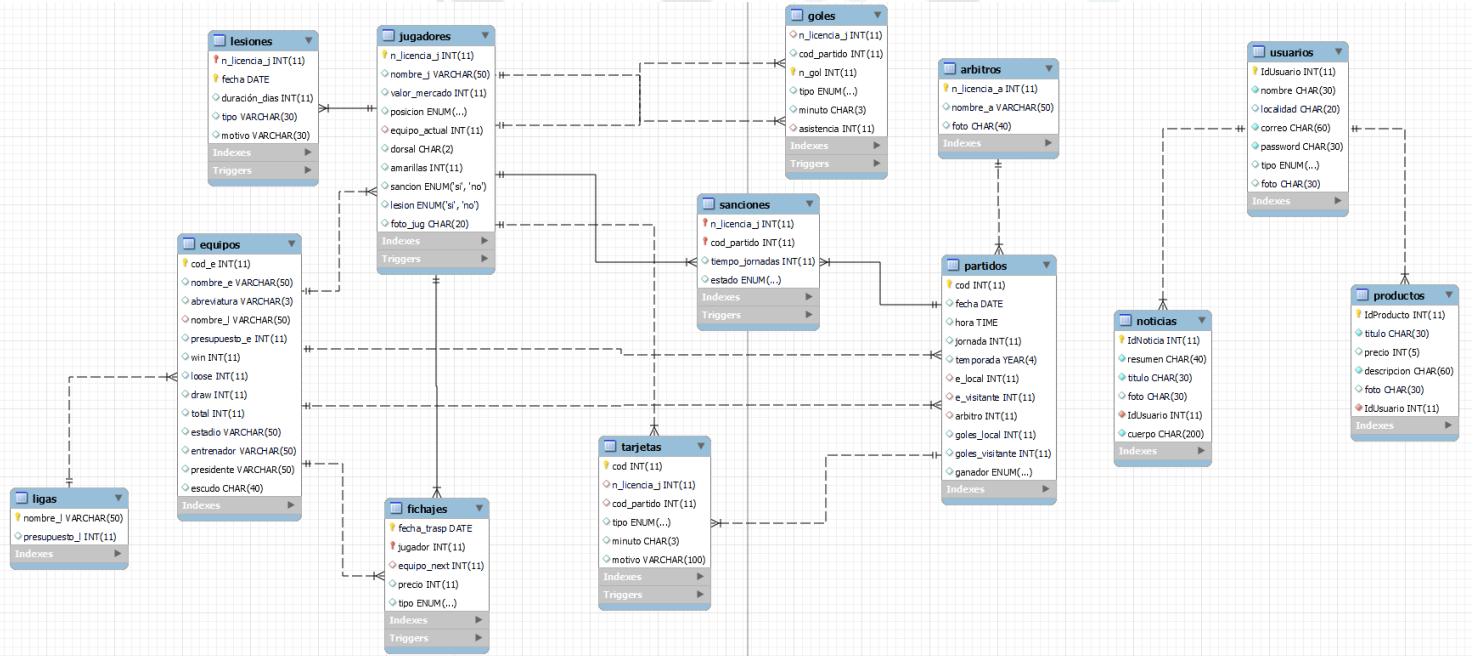
- En la raíz se encuentran, como se comentó antes, los archivos principales.
- En la carpeta login encontramos las diferentes páginas relacionadas con el inicio de sesión y muestra de info del usuario, la página de olvido.php no es operativa.
- En nosotros encontramos un par de páginas estáticas con información
 - Dentro de noticias.php se muestran las noticias y se aplica el filtro, para así acceder a noticiasext.php cuando se accede a “leer más”.
- En restringido encontramos todas las páginas a las que los usuarios autenticados como “randoms” no pueden acceder
 - Por una parte encontramos la parte destinada a gestión de la liga, con los elementos para insertar, modificar y eliminar los partidos, goles, asistencias, tarjetas, etc ...
 - Por otra parte la carpeta introducir aglomera páginas para buscar y acceder a la configuración de noticias y productos.
 - Además, en “restringido” encontramos una página a la que redirige si no se tienen privilegios adecuados y un menú para acceder a las diferentes páginas vistas anteriormente.
- Dentro de stats encontramos todas las diferentes páginas de visualización pública de datos y estadísticas de la liga, así como resultados, jugadores y otra info.
- Por último, en la tienda encontramos el buscador de productos, la visión ampliada de los mismos, y el carrito donde se guarda la info en arrays dentro de la variable \$_SESSION.

BASE DE DATOS

Es una base de datos relacional en mysql con muchas tablas diferentes, todas relacionadas entre ellas, separando la parte de estadísticas de la de usuarios, tienda y noticias. Destacan las siguientes anotaciones para tener en cuenta:

- La aplicación y control de estadísticas y flujo de datos funciona con una serie de triggers y procedimientos que necesitan ser controlados por ciertos usuarios. Dichos permisos son especiales y se tendrá que configurar el servidor para poder otorgarlos al ejecutar el script sql.
- Todos los campos que hacen referencias a imágenes se usan como almacenamiento de las rutas de la imagen del elemento, por lo que dichas rutas serán siempre relativas partiendo desde la raíz del servidor web, es decir, /var/www/html.
 - Por lo tanto, las rutas conformarán estructuras del tipo “./img/...” y los ficheros ya tendrán las modificaciones pertinentes para poder acceder desde los diferentes niveles de directorios

Aquí se deja el modelo relacional completo:



DESARROLLO DE LA INFRAESTRUCTURA

Para esta parte crítica del proyecto, lo que se hará será seguir los pasos lógicos de creación de los recursos de la infraestructura, y así podremos ir probando las diferentes partes de la infraestructura.

De cara a la creación de la estructura, hay elementos que se desplegarán a través de la CLI y otras desde la consola de AWS.



La situación inicial es que la cuenta está vacía, con 1 VPC y 3 subredes (1 por cada AZ de la región de España). Estas son las que utilizaremos a lo largo de este proyecto.

Además, dado a que esta región ha sido desplegada tras el 2019, no está habilitada por defecto en las cuentas de AWS, por lo que tendrá que hacerse manualmente para poder traspasar los recursos necesarios como permisos de IAM a esa región.

Para ello entramos en el menú de administrar regiones de aws y activamos la nueva región. Tras unos minutos, ya podemos operar dentro de ésta.

The screenshot shows the AWS IAM Session Tokens compatibility settings. A modal dialog is open, titled "Cambiar la compatibilidad de la región". The dialog contains a warning message: "Los tokens de sesión válidos en todas las regiones de AWS son de gran tamaño. Si almacena tokens de sesión, los de mayor tamaño podrían afectar sus sistemas." Below the message, it says "Tokens de sesión del punto de conexión global (https://sts.amazonaws.com) válidos en:" followed by two radio button options: "Regiones de AWS habilitadas de forma predeterminada" (unchecked) and "Todas las regiones de AWS" (checked). At the bottom are "Cancelar" and "Guardar los cambios" buttons.

Región	Estado	Acción
África (Ciudad del Cabo)	Desactivada	Activar
Asia Pacifico (Hong Kong)	Desactivada	Activar
Asia Pacifico (Hyderabad)	Desactivada	Activar
Asia Pacifico (Yakarta)	Desactivada	Activar
Asia Pacifico (Melbourne)	Desactivada	Activar
Europa (Zúrich)	Desactivada	Activar
Europa (Milán)	Desactivada	Activar
Oriente Medio (EAU)	Desactivada	Activar
Oriente Medio (Bárein)	Desactivada	Activar
Europa (España)	Activando	
Asia Pacifico (Tokio)	Habilitada de forma predeterminada	
Asia Pacifico (Seúl)	Habilitada de forma predeterminada	
Asia Pacifico (Osaka)	Habilitada de forma predeterminada	
Asia Pacifico (Mumbai)	Habilitada de forma predeterminada	
Asia Pacifico (Singapur)	Habilitada de forma predeterminada	
Asia Pacifico (Sidney)	Habilitada de forma predeterminada	

Por otro lado, habrá que hacer otro pequeño cambio en la configuración de la cuenta, ya que los tokens de inicio de sesión sólo estaban habilitados para las regiones por defecto.

Así, convertimos nuestros tokens en globales, es decir, para poder usarse por los usuarios de IAM en todas las regiones.

CREACIÓN DE LA IMAGEN DOCKER A USAR

Lo que se tendrá que hacer es crear en docker una nueva imagen a partir una del repositorio oficial:

La imagen que se usará será la de [php](#), usando su versión 8.2.0 en apache, esta imagen usa el Apache httpd en debian junto con PHP (como mod_php) y usa el framework mpm_prefork por defecto.

Como añadido, se le instalará el complemento externo de mysqli, con utilidades necesarias para el funcionamiento de la aplicación, además, se le incluye un comando para otorgar permisos necesarios al usuario www-data (El que usa el daemon de apache2), para que pueda borrar archivos del volumen y añadir nuevos.

```
FROM php:8.2.0-apache
ARG DEBIAN_FRONTEND=noninteractive
RUN docker-php-ext-install mysqli
RUN a2enmod rewrite
CMD chown -R www-data:www-data /var/www/html && service apache2
start && tail -f /dev/null
```

Para comprobar su buen funcionamiento, crearemos un entorno simple a través de docker compose, dónde tendremos una base de datos mysql con el puerto 3306 expuesto en el host para interactuar más adelante con el workbench. En cambio este tendrá una ip estática para que se pueda conectar la aplicación de php con él.

Con respecto al backend, exponemos el puerto 80 y creamos un volumen en la carpeta con la aplicación web.

Por último definimos en un archivo yaml el docker-compose con las especificaciones anteriormente mencionadas y lo creamos. Añadimos además una especificación en networks con la red que se creará para este test.

```

1 version: '3.3'
2 services:
3   db:
4     image: mysql:5.7
5     restart: always
6     environment:
7       MYSQL_PASSWORD: 'abc123.'
8       # Password for root access
9       MYSQL_ROOT_PASSWORD: 'abc123.'
10    ports:
11      # <Port exposed> : < MySQL Port running inside container>
12      - '3306:3306'
13    expose:
14      # Opens port 3306 on the container
15      - '3306'
16    networks:
17      prueba:
18        ipv4_address: 172.18.0.10
19
20 web:
21   image: prueba
22   restart: always
23   ports:
24     - '80:80'
25   expose:
26     - '80'
27   volumes:
28     - /home/ale/Escritorio/trabaj/aplicacionprueba:/var/www/html
29     # Where our data will be persisted
30   networks:
31     prueba:
32       ipv4_address: 172.18.0.11
33
34
35 networks:
36   prueba:
37     ipam:
38       config:
39         - subnet: 172.18.0.0/24
40

```

En el fichero con la información de la configuración de conexión de la aplicación asignamos la ip de ese rango asignada a la base de datos (al llevarla a producción se sustituirá por el punto de enlace de la base de datos de la nube):

```

4
5 [root]
6 Server = "172.18.0.10"
7 User = "root"
8 Password = "abc123."
9 Database = "grupo10"
0
1 [admin]
2 Server = "172.18.0.10"
3 User = "admin"
4 Password = "adminliga123."
5 Database = "grupo10"

```

Por último lo ejecutamos:

```

ale@aleubu:~/Escritorio/trabaj/prueba$ docker-compose up -d
[+] Running 3/3
  :: Network prueba_prueba    Created                               0.0s
  :: Container prueba-web-1   Started                               0.4s
  :: Container prueba-db-1   Started                               0.3s
ale@aleubu:~/Escritorio/trabaj/prueba$ 

```

Para tener la base de datos de la aplicación lo que haremos es conectarnos con el cliente con el usuario root a localhost y ejecutamos el script de la base de datos.

#	id	nombre_j	valor_mercado	posicion	equipo_actua_dorsal	amarillas	sancion	lesion	foto_jug
1	Iago Aspas Juncal	5000000	DC	1	10	1	no	no	NULL
2	Hugo Mallo	50000	LD	1	2	1	no	no	NULL
3	Agustín Marchesin	50000	POR	1	13	1	no	no	NULL
4	Iván Villar	50000	POR	1	1	0	no	no	NULL
5	Oscar Minguez	50000	DFC	1	3	0	no	no	NULL
6	Unai Núñez	50000	DFC	1	4	0	no	no	NULL
7	Javi Galán	50000	LI	1	17	1	no	no	NULL
8	Joseph Aldo	50000	DFC	1	15	1	no	no	NULL
9	Kevin Vázquez	50000	LD	1	20	0	no	no	NULL
10	Oscar Rodríguez	50000	MCO	1	5	0	no	no	NULL
11	Denis Suárez	50000	MCO	1	6	0	no	no	NULL
12	Fran Beltrán	50000	MC	1	8	0	no	no	NULL
13	Franco Cervi	50000	EI	1	11	0	no	no	NULL
14	Renato Tapia	50000	MCD	1	14	0	no	no	NULL
15	Miguel Baeza	50000	MC	1	16	0	no	no	NULL
16	Williot Swedberg	50000	MC	1	19	0	no	no	NULL
17	Augusto Solarí	50000	ED	1	21	0	no	no	NULL
18	Luca de la Torre	50000	ED	1	23	0	no	no	NULL
19	Carles Pérez	50000	ED	1	7	0	no	no	NULL

Ya solo falta hacer la petición a localhost y vemos que aporta los resultados provenientes de la base de datos y fotos del punto de montaje que creamos.

The screenshot shows a web browser window for 'FootPassion' at 'localhost/tienda/tienda.php'. The header features the 'FOOTPASSION' logo on a green grass background, a user icon, and navigation links for 'Noticias', 'Estadísticas', 'Tienda', and 'Restringido'. Below the header is a search bar with a 'Buscar' button, and filters for 'Camisetas', 'Merch', 'Botas', and 'No ordenar'. Two jerseys are displayed: one with a blacked-out face and another with 'DE BRUYNE' and the number '17'. The main section is titled 'JUGADORES DE LA LIGA' and includes a search bar for 'Nombre' and dropdown filters for 'Dorsal', 'Cualquiera', and 'Cualquiera'. A player card for 'Iago Aspas Juncal' from 'Real Club Celta de Vigo' (Dorsal 10, Posición: DC) is shown with a portrait photo.

Qué cambiaría:

- El docker compose solo es para las pruebas, no se va a usar en ningún momento de la producción.
- El volumen consistirá en uno que se montará más adelante dentro de los contenedores con un servicio de AWS.
- La base de datos será accesible gracias a políticas de grupos de seguridad, pero en la configuración de conexión sql se cambiaría por el nombre DNS de la red interna de la nube.

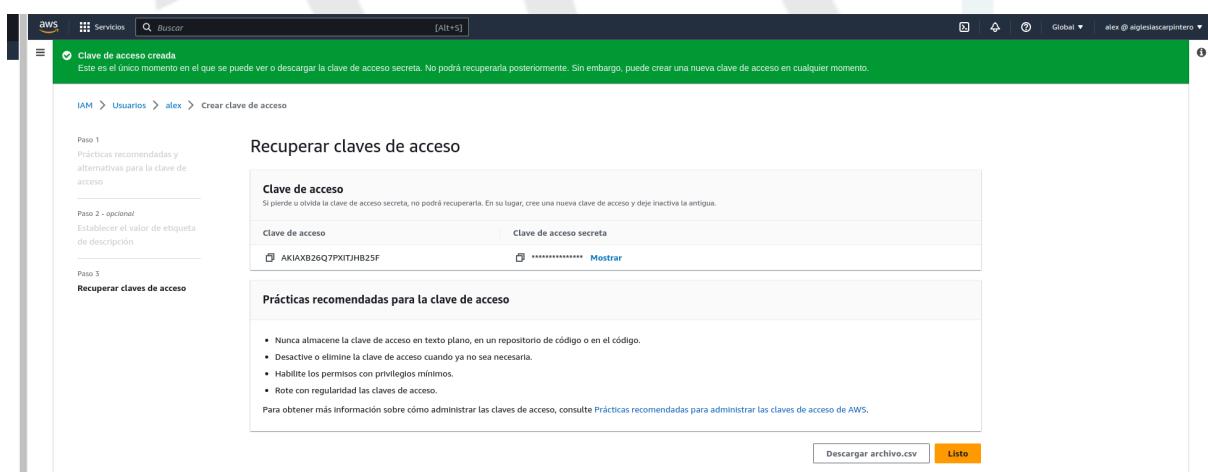
INSTALACIÓN DE LA CLI DE AMAZON EN UBUNTU 22.04

Solamente tendremos que descargar el zip con el ejecutable, descomprimir, y ejecutar.

```
curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"
unzip awscliv2.zip
sudo ./aws/install
```

```
ale@aleubu:~$ sudo ./aws/install
[sudo] contraseña para ale:
You can now run: /usr/local/bin/aws --version
ale@aleubu:~$ aws --version
aws-cli/2.11.13 Python/3.11.3 Linux/5.19.0-38-generic exe/x86_64/ubuntu.22 prompt/off
ale@aleubu:~$ ls
```

Ahora crearemos un par de claves de acceso para poder acceder, y las descargamos en un CSV.



Ahora solo falta indicar los parámetros que usará la CLI para autenticarse y poder ejecutar los comandos. Se especifican las claves de acceso, la región en la que se trabaja por defecto (España), y el formato de salida, que será json para que nos sirva más adelante.

```
ale@aleubu:~$ aws configure
AWS Access Key ID [None]: AKIAXB26Q7PXITJHB25F
AWS Secret Access Key [None]: *****9WE41Kkr
Default region name [None]: eu-south-2
Default output format [None]: json
ale@aleubu:~$ aws ec2 describe-instances
{
    "Reservations": []
}
ale@aleubu:~$
```

HOST BASTIÓN

En este documento ya se hizo y hará referencia múltiples veces a un host bastión por el que se accede a la base de datos y al volumen EFS, por lo tanto, vale la pena resaltar algunos pequeños aspectos en el despliegue de la máquina.

- Se trata de una máquina virtual muy simple y pequeña (2 vCPU y 4GB RAM)
- Lleva instalado como sistema operativo Amazon Linux, un OS ligero de alta compatibilidad con los sistemas de AWS, que nos es más que suficiente para la utilidad que le queremos dar
- Se despliega en la AZ eu-south-2c, dato que será importante más adelante.
- Al ser desplegado en una subred pública, cada vez que se inicie se le dará una dirección ip pública con su respectivo nombre DNS, además de tener asociado un nombre DNS privado fijo.

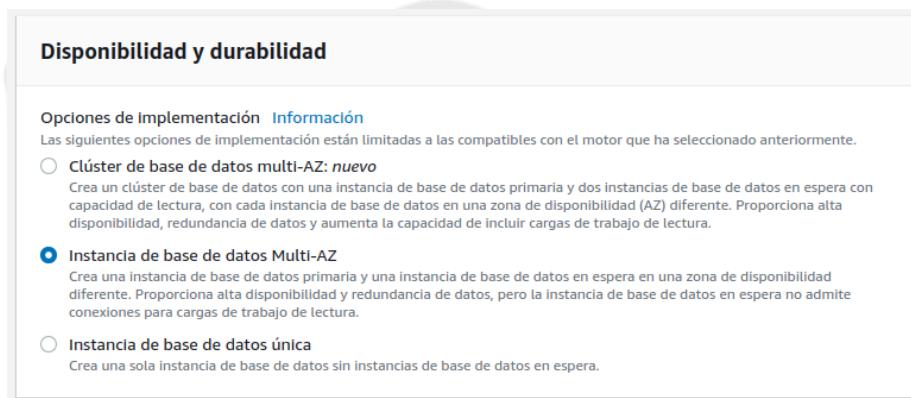
▼ Detalles de la instancia Información		
Plataforma	ID de AMI	Monitoreo desactivado
Linux/UNIX (inferior)	ami-0a2ac5650064429f0	Protección de terminación desactivado
Detalles de la plataforma	Nombre de AMI	Ubicación de AMI
Linux/UNIX	al2023-ami-2023.0.20230329.0-kernel-6.1-x86_64	amazon/al2023-ami-2023.0.20230329.0-kernel-6.1-x86_64
Detener la protección desactivado	Hora de lanzamiento	Key pair assigned at launch
	Mon May 01 2023 13:00:50 GMT+0200 (hora de verano de Europa central) (21 days)	clavessh-tunel-rds
Ciclo de vida normal	Índice de lanzamiento de AMI	ID de kernel
Motivo de transición de estado User initiated (2023-05-01 12:47:09 GMT)	0	-
Mensaje de transición de estado Client.UserInitiatedShutdown: User initiated shutdown	Especificación de crédito unlimited	ID de disco RAM
Propietario	Operación de uso	Current instance boot mode
484992678894	RunInstances	uefi
Permitir etiquetas en los metadatos de la instancia desactivado	Modo de arranque	Responder a RBN de DNS de nombre de host IPv4
	uefi-preferred	Habilitado
Utilizar RBN como nombre de host del SO invitado desactivado		

BASE DE DATOS

Crearemos en primer lugar un grupo de seguridad para el acceso a la base de datos

Vamos a comenzar creando la base de datos que queríamos: Parámetros configurados:

- MySQL 8.0.32
- Plantilla: Producción
- Disponibilidad:



- Identificador-Clúster database-1
- Nombre master user: administrador
- Passwd master user: abc123..
- Tipo de instancia:db.m5.xlarge
 - 4 vCPUs
 - 16 GiB RAM
 - 4750 Mbps
 - 150 GB de almacén de instancias
- Almacenamiento: 400 GiB y 3000 IOPS (Input/Output Operations Per Second)
- No queremos que sea accesible de forma pública

El resto de parámetros podemos dejarlos por defecto

```
aws rds create-db-instance \
--db-instance-identifier
db-footpassion \
--allocated-storage 400 \
--db-instance-class db.m5.xlarge \
--engine mysql \
--master-username administrador \
--master-user-password abc123.. \
--no-publicly-accessible \
--multi-az \
--engine-version 8.0.32 \
--deletion-protection
--db-parameter-group-name
editparameters
```

Tras un rato ya se nos ha creado la base de datos creada en la vpc por defecto que es España y en una AZ aleatoria de entre las 3 posibles. Si ahora simulamos una caída de la base de datos, la réplica de la otra zona de disponibilidad se levantaría..

```
ale@aleubu: ~ aws rds create-db-instance \
--db-instance-identifier db-footpassion \
--allocated-storage 400 \
--db-instance-class db.m5.xlarge \
--engine mysql \
--master-username admin \
--master-user-password abc123.. \
--no-publicly-accessible \
--multi-az \
--engine-version 8.0.32 \
--deletion-protection
{
    "DBInstance": {
        "DBInstanceIdentifier": "db-footpassion",
        "DBInstanceClass": "db.m5.xlarge",
        "Engine": "mysql",
        "DBInstanceStatus": "creating",
        "MasterUsername": "admin",
        "AllocatedStorage": 400,
        "PreferredBackupWindow": "06:17-06:47",
        "BackupRetentionPeriod": 1,
        "DBSecurityGroups": [],
        "VpcSecurityGroups": [
            {
                "VpcSecurityGroupId": "sg-00fd0eede28450261",
                "Status": "active"
            }
        ],
        "DBParameterGroups": [
            {
                "DBParameterGroupName": "default.mysql8.0",
                "ParameterApplyStatus": "in-sync"
            }
        ],
        "DBClusterIdentifier": null
    }
}
```

Bases de datos										
<input checked="" type="checkbox"/> Recursos del grupo <input type="button" value="C"/> <input type="button" value="Modificar"/> <input type="button" value="Acciones"/> Restaurar desde S3 <input type="button" value="Crear base de datos"/> < 1 > 										
	Identificador de base de datos	Rol	Motor	Región y AZ	Tamaño	Estado	CPU	Actividad actual	Mantenimiento	
●	db-footpassion	Instancia	MySQL Community	eu-south-2b	db.m5.xlarge	Disponible	0.84%	0 Conexiones		

Ahora bien, para securizar el acceso, en lugar de exponer la base de datos vamos a colocar una muy pequeña instancia que usaremos de bastión para entrar desde el exterior, hacer un túnel ssh y acceder a la base de datos.

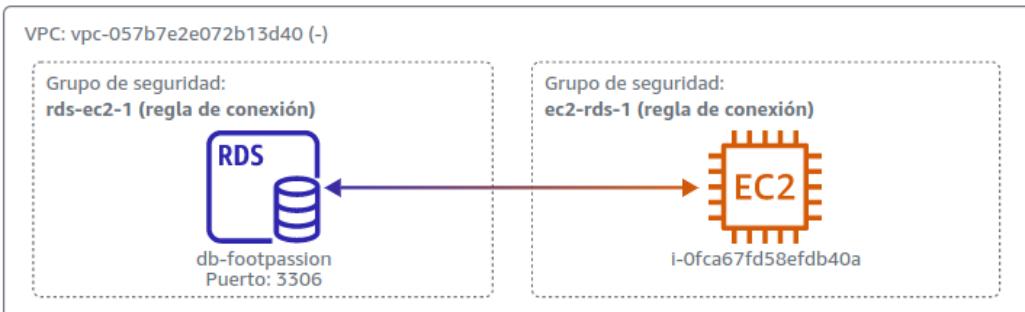
Instancias (1) Información										
<input type="button" value="C"/> <input type="button" value="Conectar"/> Estado de la instancia <input type="button" value="A"/> <input type="button" value="Acciones"/> Lanzar instancias < 1 > 										
<input type="checkbox"/> Buscar Instancia por atributo o etiqueta (case-sensitive)										
Name	ID de la instancia	Estado de la i...	Tipo de inst...	Comprobación ...	Estado de la ...	Zona de dispon...	DNS de IPv4 pública	Dirección IP...	IP elástica	
tunel-db	i-0fc6a67fd58befdb40a	En ejecución	t5.micro	2/2 comprobacion	Sin alarmas	eu-south-2c	ec2-18-101-27-237.eu...	18.101.27.237	-	

Para poder conectarse de 1 a otra, aprovechamos 1 opción que nos ofrece aws de configurarnos

Resumen de conexión [Información](#)

Está configurando una conexión entre la base de datos de RDS [db-footpassion](#) y la instancia de EC2 [i-0fca67fd58efdb40a](#).

Para configurar una conexión entre la base de datos y la instancia de EC2, el grupo de seguridad de VPC `rds-ec2-1` se agrega a la base de datos y el grupo de seguridad de VPC `ec2-rds-1` se agrega a la instancia de EC2.



Lo único que falta es crear el túnel que nos permita pasar por la instancia, y para ello se tendrá que ejecutar este comando, teniendo en el directorio la clave .pem que se le proporcionará al cliente;

```
ssh -i "clavessh-tunel-rds.pem" -L  
3333:db-footpassion.chd3axdvsmme.eu-south-2.rds.amazonaws.com:3306  
ec2-user@18.101.27.237 -N -f
```

Lo que hace este comando es, escrito con palabras:

“Que todo lo que vaya hacia el puerto 3333 de localhost lo mande al puerto 3306 de la url especificada, y eso me lo haces a través de la ip 18.101.27.237, a la cual te conectarás usando el usuario ec2-user y con la clave clavessh-tunel-rds.pem”

Los parámetros son para que se ejecute en 2º Plano sin que se vea nada en la terminal y el cliente pueda trabajar correctamente.

Ahora estableceremos la conexión y revisaremos que hay un puerto escuchando, además probaremos la conexión establecida para ver si se puede llegar hasta el final:

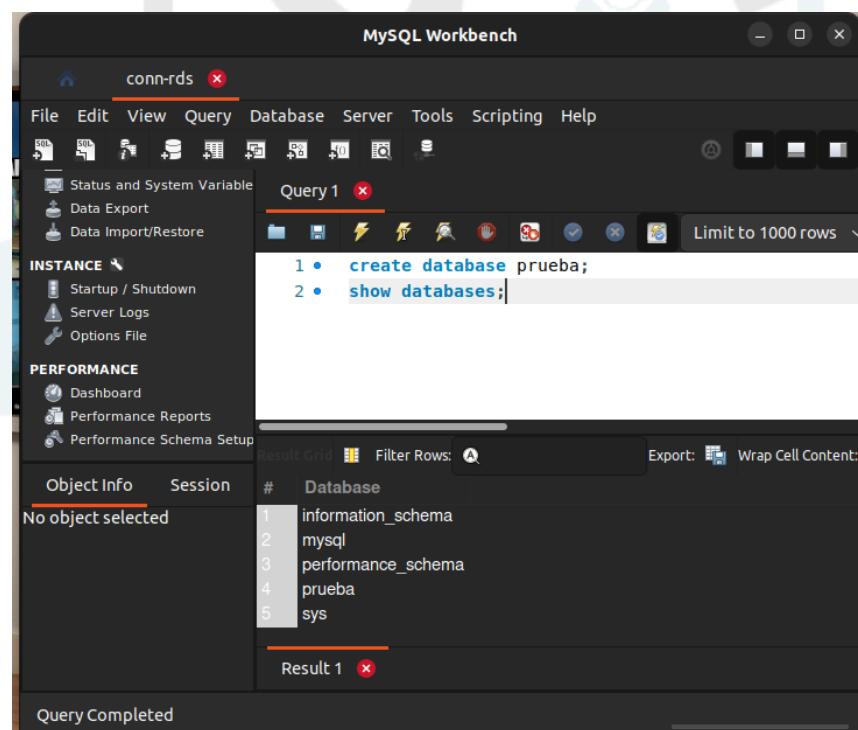
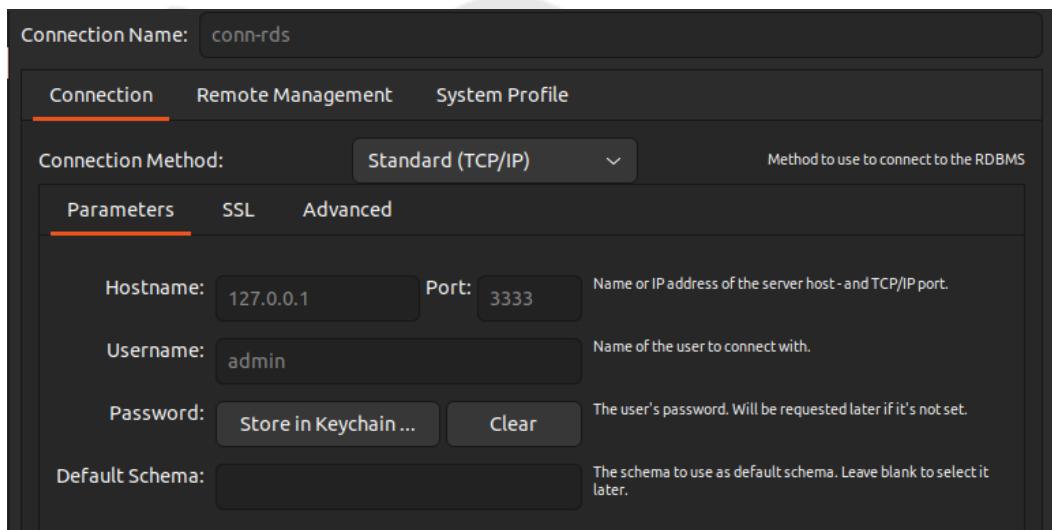


```

ale@aleubu:~/Escritorio$ ssh -i "clavessh-tunel-rds.pem" -L 3333:db-footpassion.chd3axdvsdde.eu-south-2.rds.amazonaws.com:3306 ec2-user@18.101.27.237 -N -f
ale@aleubu:~/Escritorio$ lsof -i4 -P | grep -i "listen" | grep 3333
ssh     8004 ale    5u  IPv4  98169      0t0  TCP localhost:3333 (LISTEN)
ale@aleubu:~/Escritorio$ nc -zv 127.0.0.1 3333
Connection to 127.0.0.1 3333 port [tcp/*] succeeded!
ale@aleubu:~/Escritorio$ 

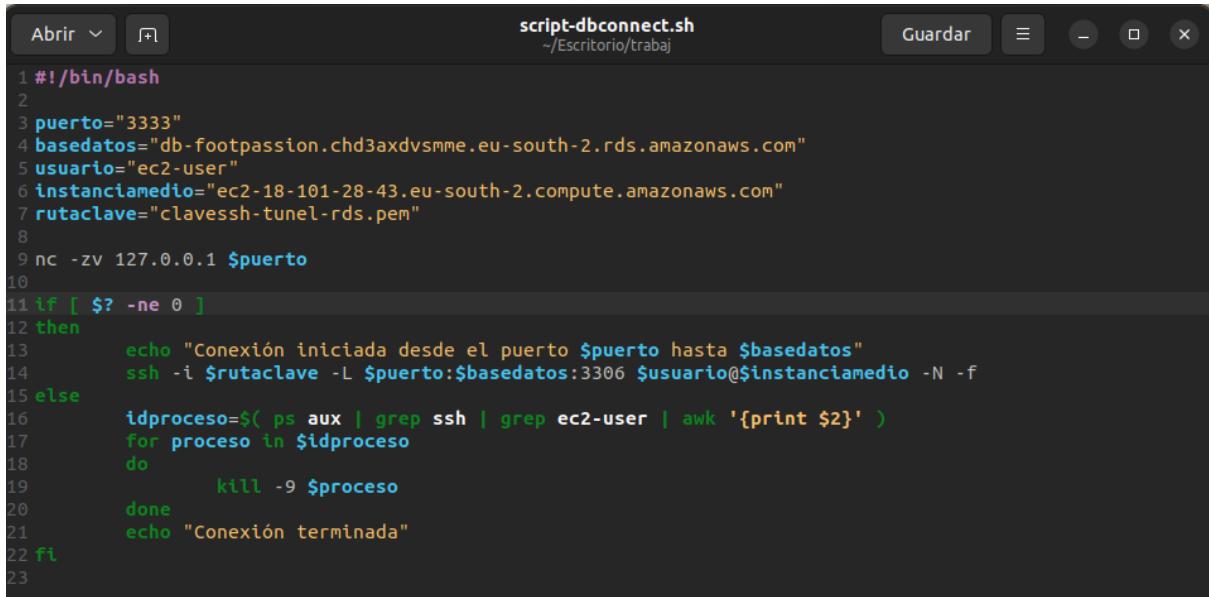
```

Ahora toca acceder con mysql workbench entrando por localhost al puerto en el que escucha, y probamos que de verdad está realizando bien la conexión:



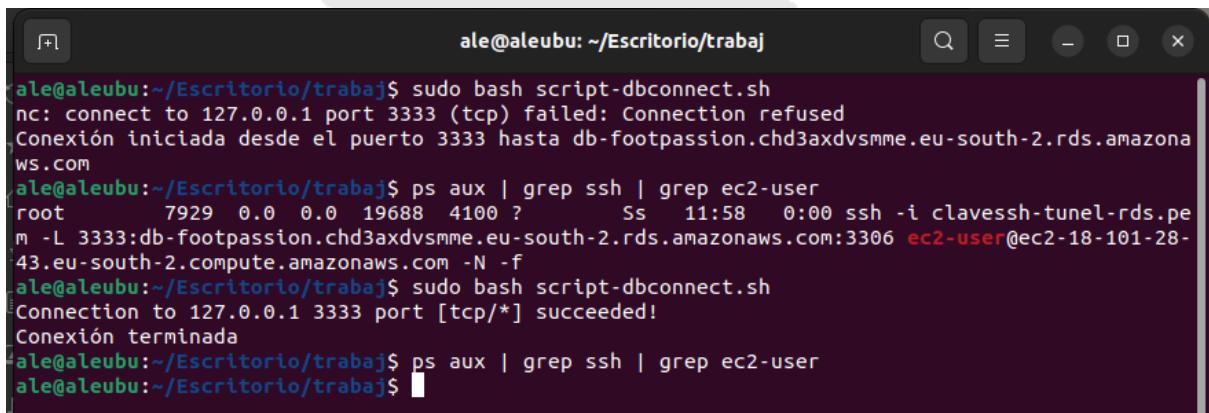
Por último y como cortesía de la casa, le proporcionamos al cliente un script para que abra el túnel si está cerrado y que lo cierre si está abierto.

Primero revisa si el puerto especificado de localhost está escuchando. En caso afirmativo busca los procesos relacionados con ssh y el usuario específico con el que nos conectamos para matar dichos procesos. Si por el contrario está cerrado, ejecuta el comando explicado anteriormente en segundo plano para abrir la conexión, para lo que pedirá contraseña, claro está.



```
script-dbconnect.sh
~/Escritorio/trabaj
Guardar
Abrir ▾
script-dbconnect.sh
~/Escritorio/trabaj
Guardar
- □ ×
1 #!/bin/bash
2
3 puerto="3333"
4 basedatos="db-footpassion.chd3axdvsme.eu-south-2.rds.amazonaws.com"
5 usuario="ec2-user"
6 instanciamedio="ec2-18-101-28-43.eu-south-2.compute.amazonaws.com"
7 rutaclave="clavessh-tunel-rds.pem"
8
9 nc -zv 127.0.0.1 $puerto
10
11 if [ $? -ne 0 ]
12 then
13     echo "Conexión iniciada desde el puerto $puerto hasta $basedatos"
14     ssh -i $rutaclave -L $puerto:$basedatos:3306 $usuario@$instanciamedio -N -f
15 else
16     idproceso=$( ps aux | grep ssh | grep ec2-user | awk '{print $2}' )
17     for proceso in $idproceso
18     do
19         kill -9 $proceso
20     done
21     echo "Conexión terminada"
22 fi
23
```

Pasamos los parámetros de acceso primero, y ejecutamos el comando para probar la conexión por el puerto de escucha. En caso de que la salida sea de error (distinta a 0), entonces mataremos todos los procesos que tengan relación con esa conexión ssh, y para filtrar usaremos el nombre de usuario que se ha utilizado (ec2-user).



```
ale@aleubu:~/Escritorio/trabaj$ sudo bash script-dbconnect.sh
nc: connect to 127.0.0.1 port 3333 (tcp) failed: Connection refused
Conexión iniciada desde el puerto 3333 hasta db-footpassion.chd3axdvsme.eu-south-2.rds.amazonaws.com
ale@aleubu:~/Escritorio/trabaj$ ps aux | grep ssh | grep ec2-user
root      7929  0.0  0.0 19688  4100 ?        Ss   11:58  0:00 ssh -i clavessh-tunel-rds.pem -L 3333:db-footpassion.chd3axdvsme.eu-south-2.rds.amazonaws.com:3306 ec2-user@ec2-18-101-28-43.eu-south-2.compute.amazonaws.com -N -f
ale@aleubu:~/Escritorio/trabaj$ sudo bash script-dbconnect.sh
Connection to 127.0.0.1 3333 port [tcp/*] succeeded!
Conexión terminada
ale@aleubu:~/Escritorio/trabaj$ ps aux | grep ssh | grep ec2-user
ale@aleubu:~/Escritorio/trabaj$
```

Para tener nuestra base de datos completamente operativa, le pedimos al cliente que pruebe el script de inicio de la base de datos desde su cliente para poder probar su operatividad, pero se encuentra con problemas de que no puede realizar operaciones que requieran de permisos SUPER.

Para ello estableceremos la variable log_bin_trust_function_creators a 1. Esto es posible creando un grupo de parámetros en la consola de aws y metiendo la base de datos en dicho grupo, reiniciándola.

Finalmente ya puede ejecutar su script y tener todo listo para la puesta en producción.

ALMACENAMIENTO PARA LOS CONTENEDORES

Crearemos ahora el volumen de EFS para poder adjuntar a las instancias que se vayan creando por las tareas. Éste consiste en un volumen de tan solo 6KiB, pero no es un problema puesto que EFS escala automáticamente y solamente se paga por lo que se utiliza, conforme se suben archivos el volumen crecerá.

El problema de los contenedores reside en que el almacenamiento de la capa superior no es persistente, así que se debe montar un volumen. Una opción para ello podría ser el volumen de docker, pero solo es compatible con EBS, el cual solo se puede adjuntar a 1 instancia a la vez

Para usar volúmenes del sistema de archivos de Amazon EFS para sus contenedores, se debe especificar las configuraciones de volumen y punto de montaje en su definición de tarea. Esto se configurará más adelante con la creación de tareas pero por ahora creamos el volumen:

```
aws efs create-file-system \
--encrypted \
--creation-token FileSystemForWalkthrough1 \
--tags Key=Name,Value=volfotos \
--region eu-south-2
```



```
ale@aleubu:~$ aws efs create-file-system \
--encrypted \
--creation-token FileSystemForWalkthrough1 \
--tags Key=Name,Value=volfotos \
--region eu-south-2
{
    "OwnerId": "484992678894",
    "CreationToken": "FileSystemForWalkthrough1",
    "FileSystemId": "fs-095ef927774d7b6c2",
    "FileSystemArn": "arn:aws:elasticfilesystem:eu-south-2:484992678894:file-system/fs-095ef927774d7b6c2",
    "CreationTime": "2023-04-18T22:54:38+02:00",
    "LifeCycleState": "creating",
    "Name": "volfotos",
    "NumberOfMountTargets": 0,
    "SizeInBytes": {
        "Value": 0,
        "ValueInIA": 0,
        "ValueInStandard": 0
    },
    "PerformanceMode": "generalPurpose",
    "Encrypted": true,
    "KmsKeyId": "arn:aws:kms:eu-south-2:484992678894:key/8d425855-ef49-4602-a6e4-cc8080df2bf3",
    "ThroughputMode": "bursting",
    "Tags": [
        {
            "Key": "Name",
            "Value": "volfotos"
        }
    ],
    ....skipping...
```

Sistemas de archivos (1)								
C Ver detalles Eliminar Crear un sistema de archivos								
Nombre	ID del sistema de archivos	Cifrado	Tamaño total	Tamaño en EFS estándar	Tamaño en EFS de acceso poco frecuente	Rendimiento aprovisionado (MiB/s)	Estado del sistema de archivos	Hora de creación
volfotos	fs-095ef927774d7b6c2	Cifrado	6.00 KIB	6.00 KIB	0 bytes	-	Disponible	Tue, 18 Apr 2023 20:54:38 GMT

Sin embargo, para que posteriormente podamos tener acceso desde los contenedores a este sistema de archivos debemos permitir en el grupo de seguridad asociado al EFS que permita las conexiones entrantes al puerto 2049.

EC2 > Grupos de seguridad > Crear grupo de seguridad

Crear grupo de seguridad Información

Un grupo de seguridad actúa como un firewall virtual para que la instancia controle el tráfico de entrada y salida. Para crear un nuevo grupo de seguridad, complete los campos siguientes.

Detalles básicos

Nombre del grupo de seguridad Información
EFS
El nombre no se puede editar después de su creación.

Descripción Información
Permitir puerto 2049

VPC Información
vpc-057b7e2e072b13d40

Reglas de entrada Información

Tipo <small>Información</small>	Protocolo <small>Información</small>	Intervalo de puertos <small>Información</small>	Origen <small>Información</small>	Descripción: opcional <small>Información</small>
TCP personalizado	TCP	2049	Anywhere-Int.	0.0.0.0/0 X

[Agregar regla](#)

Acto seguido, crearemos los diferentes puntos de montaje (1 en cada AZ), en los que se van a ir asociando los diferentes contenedores, cada uno al de la AZ en dónde se despliegue. A estos puntos de montaje le asociamos el grupo de seguridad que acabamos de crear.

Zona de disponibilidad

Virtual Private Cloud (VPC)
Elija la VPC en la que desea que las instancias EC2 se conecten a su sistema de archivos.
vpc-057b7e2e072b13d40
predeterminado

Destinos de montaje

Un destino de montaje proporciona un punto de enlace NFSv4 en el que puede montar un sistema de archivos de Amazon EFS. Le recomendamos que cree un destino de montaje por zona de disponibilidad. [Más información](#)

Zona de disponibilidad	ID de la subred	Dirección IP	Grupos de seguridad
eu-south-2a	subnet-02d5bcc5b9474bca	Automático	Elegir grupos de seguridad sg-091140f2e5a6030c7 X EFS
eu-south-2b	subnet-0f92359bc314b6585	Automático	Elegir grupos de seguridad sg-091140f2e5a6030c7 X EFS
eu-south-2c	subnet-0c28bc8e106a30c8b	Automático	Elegir grupos de seguridad sg-091140f2e5a6030c7 X EFS

Solo falta crear una forma de acceder al EFS, y para ello lo que se hará desde el host bastión, al que podemos acceder por ssh, montar el sistema y añadir los archivos.

```
ale@aleubu:~/Escritorio/trabajos$ scp -i clavessh-tunel-rds.pem fotos.zip ec2-user@ec2-18-100-62-133.eu-south-2.compute.amazonaws.com:/home/ec2-user/fotos.zip
fotos.zip                                         100%   27MB   1.7MB/s   00:16
ale@aleubu:~/Escritorio/trabajos$
```

Nos conectamos, actualizamos los repositorios y comprobamos que estén instalados los nfs-utils.

```
A newer release of "Amazon Linux" is available.  
Version 2023.0.20230419:  
Run "/usr/bin/dnf check-release-update" for full release and version update info  
 ,      #_  
~\_\_ #####_          Amazon Linux 2023  
~~ \_\#####\_  
~~   \###|  
~~     \#/ ____  https://aws.amazon.com/linux/amazon-linux-2023  
~~       V~' '-'>  
~~~           /  
~~. _ . _ /  
~~. / _ /  
~~. /m /'  
Last login: Mon Apr 24 11:15:31 2023 from 90.166.134.107  
[ec2-user@ip-172-31-14-212 ~]$ sudo yum -y update
```

```
[ec2-user@ip-172-31-14-212 ~]$ sudo yum -y install nfs-utils
Last metadata expiration check: 0:47:00 ago on Mon Apr 24 11:12:12 2023.
Package nfs-utils-1:2.5.4-2.rc3.amzn2023.0.3.x86_64 is already installed.
Dependencies resolved.
Nothing to do.
Complete!
[ec2-user@ip-172-31-14-212 ~]$
```

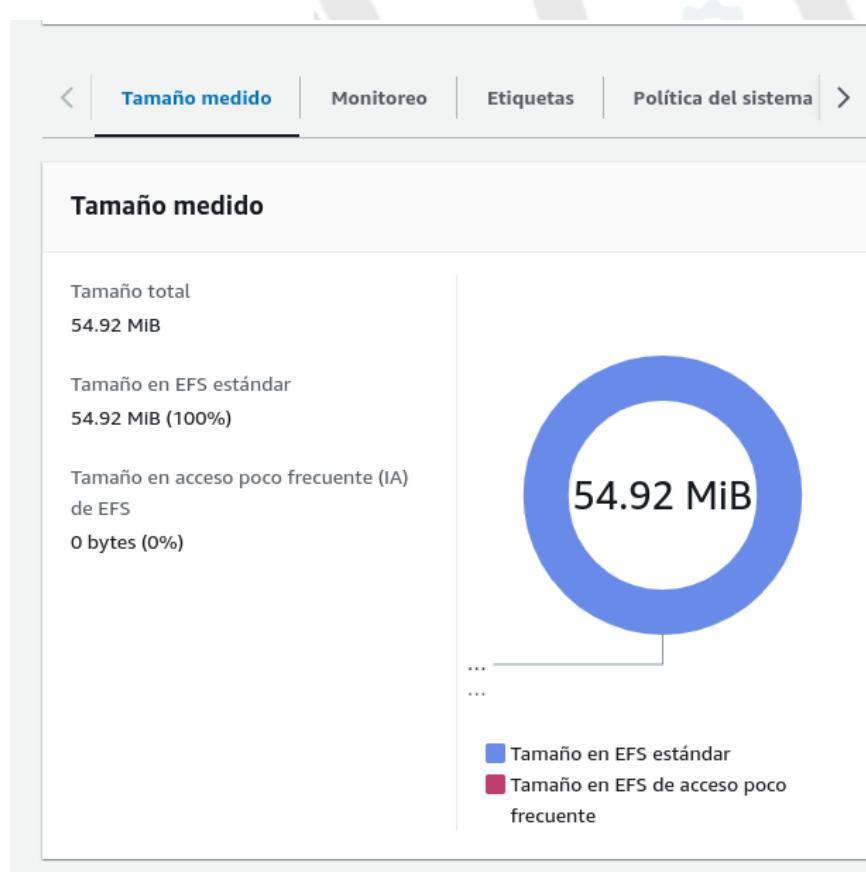
Creamos ahora la carpeta donde montaremos el EFS, lo montamos usando el destino de montaje de la misma zona de disponibilidad que el EC2, es decir, la c.

```
[ec2-user@ip-172-31-14-212 ~]$ mkdir montaje-efs  
[ec2-user@ip-172-31-14-212 ~]$ sudo mount -t nfs -o nfsvers=4.1,rsize=1048576,ws  
ize=1048576,hard,timeo=600,retrans=2,noresvport 172.31.6.251:/ montaje-efs  
[ec2-user@ip-172-31-14-212 ~]$ ls
```

Descomprimimos el zip que mandamos y comprobamos que el tamaño del EFS ha subido de 6 KiB a mucho más

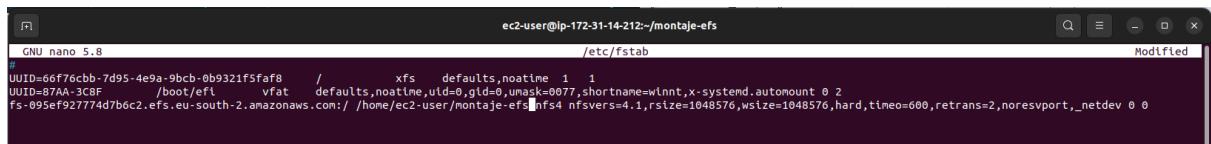
```
[ec2-user@ip-172-31-14-212 montaje-efs]$ sudo unzip fotos.zip
Archive: fotos.zip
  creating: arbitros/
  creating: escudos/
  inflating: escudos/10.gif
  inflating: escudos/11.gif
  inflating: escudos/12.gif
  inflating: escudos/13.gif
  inflating: escudos/14.gif
  inflating: escudos/15.gif
  inflating: escudos/16.gif
  inflating: escudos/17.gif
  inflating: escudos/18.gif
```

```
[ec2-user@ip-172-31-14-212 montaje-efs]$ ls
FONDO.png      carrito.png      fotos.zip        lupa.png      productos
LogoProyecto.png cesped.jpg      gigachad.jpg    mike.png     roja.jpg
amarilla.png    cesped_natural.jpg jugadores      noticias    roja.png
arbitros        editar.png      laliga.jpg     papelera.png user.png
aspas.jpg       escudos        letrasheader.png pelota.jpg  usuarios
[ec2-user@ip-172-31-14-212 montaje-efs]$
```



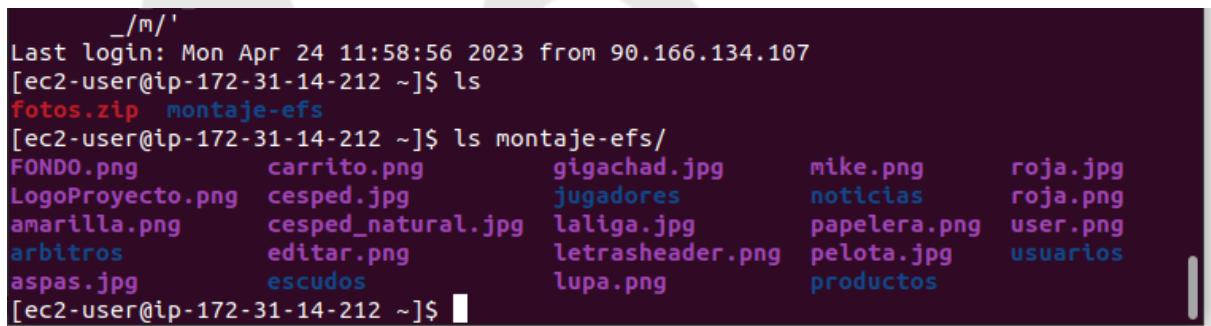
Sin embargo, este volumen no es persistente, por lo que tras 1 reinicio se perderá y no se montará. Para ello añadimos 1 línea en el archivo de /etc/fstab:

```
file_system_id.efs.aws-region.amazonaws.com:/           mount_point      nfs4
nfsvers=4.1,rsize=1048576,wsize=1048576,hard,timeo=600,retrans=2,noresvport,_netd
ev 0 0
```



```
ec2-user@ip-172-31-14-212:~/montaje-efs
GNU nano 5.8
/etc/fstab
#UUID=66f76cbb-7d95-4e9a-9bc9-0b9321f5faf8   /          xfs    defaults,noatime 1 1
UUID=87AA-3C8F        /boot/efi    vfat   defaults,noatime,uid=0,gid=0,umask=0077,shortname=wlnnt,x-systemd.autounmount 0 2
fs-695ef92774d7b6c2.efs.eu-south-2.amazonaws.com:/ /home/ec2-user/montaje-efs nfs4 nfsvers=4.1,rsize=1048576,wsize=1048576,hard,timeo=600,retrans=2,noresvport,_netdev 0 0
```

Reiniciamos y comprobamos que los archivos continúan en su sitio.



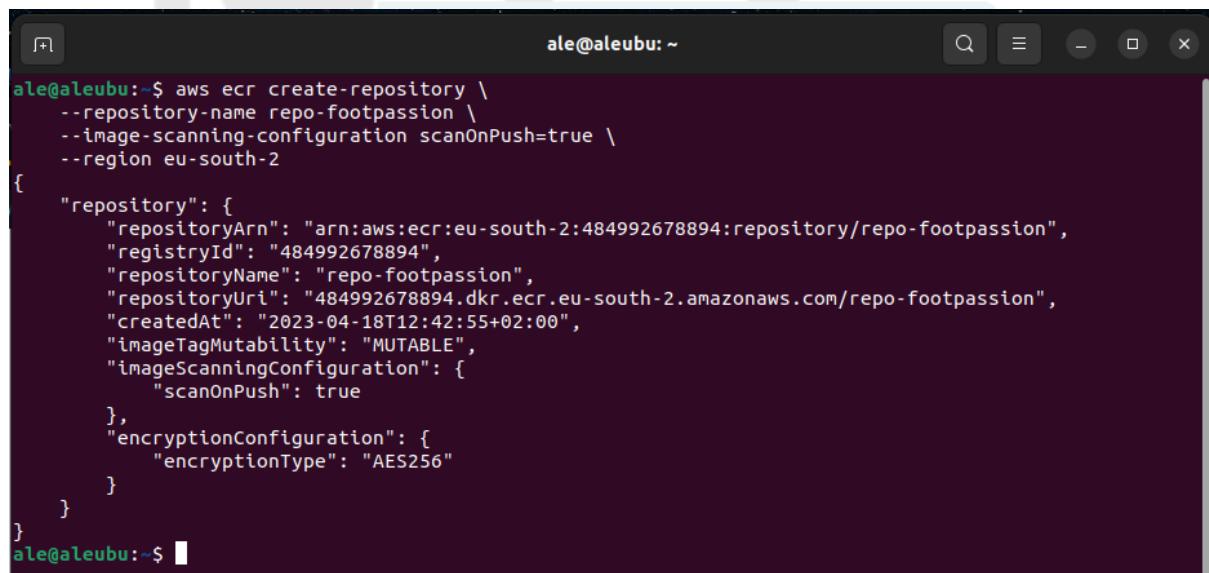
```
_/m/
Last login: Mon Apr 24 11:58:56 2023 from 90.166.134.107
[ec2-user@ip-172-31-14-212 ~]$ ls
fotos.zip  montaje-efs
[ec2-user@ip-172-31-14-212 ~]$ ls montaje-efs/
FONDO.png      carrito.png      gigachad.jpg      mike.png      roja.jpg
LogoProyecto.png  cesped.jpg      jugadores      noticias      roja.png
amarilla.png    cesped_natural.jpg  laliga.jpg      papelera.png  user.png
arbitros        editar.png      letrasheader.png pelota.jpg    usuarios
aspas.jpg       escudos        lupa.png       productos
[ec2-user@ip-172-31-14-212 ~]$
```

CLÚSTER DE CONTENEDORES DE APLICACIÓN

IMAGEN AL REPOSITORIO

Lo primero que debemos hacer es pasar la imagen que utilizaremos a un repositorio, que inicialmente será una muy simple basada en la testeada pero solo con un archivo que contiene la función phpinfo(). Al ser un servicio regional, especificaremos que se creará en eu-south-2:

```
aws ecr create-repository \
--repository-name repo-footpassion \
--image-scanning-configuration scanOnPush=true \
--region eu-south-2
```



```
ale@aleubu:~$ aws ecr create-repository \
--repository-name repo-footpassion \
--image-scanning-configuration scanOnPush=true \
--region eu-south-2

{
  "repository": {
    "repositoryArn": "arn:aws:ecr:eu-south-2:484992678894:repository/repo-footpassion",
    "registryId": "484992678894",
    "repositoryName": "repo-footpassion",
    "repositoryUri": "484992678894.dkr.ecr.eu-south-2.amazonaws.com/repo-footpassion",
    "createdAt": "2023-04-18T12:42:55+02:00",
    "imageTagMutability": "MUTABLE",
    "imageScanningConfiguration": {
      "scanOnPush": true
    },
    "encryptionConfiguration": {
      "encryptionType": "AES256"
    }
  }
}
ale@aleubu:~$
```



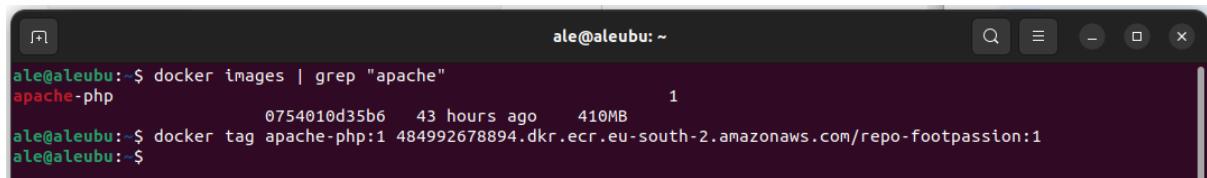
Repositorios privados (1)						
Nombre del repositorio	URI	Creado en	Inmutabilidad de etiqueta	Análisis durante el envío	Tipo de cifrado	Caché de extracción
repo-footpassion	484992678894.dkr.ecr.eu-south-2.amazonaws.com/repo-footpassion	18 de abril de 2023, 12:42:55 (UTC+02)	Desactivado	Habilitado	AES-256	Inactivo

Ahora bien, para poder subir la imagen al repositorio en primer lugar hacemos docker tag a la imagen en cuestión con ese formato:

id_cuenta.dkr.ecr.region.amazonaws.com/nombre_repo

Este proceso se hace para crear una referencia a dicha imagen en local con la información del repositorio para que a continuación al realizar el push lo mande al lugar correcto. Esto ahora quedaría de la siguiente manera:

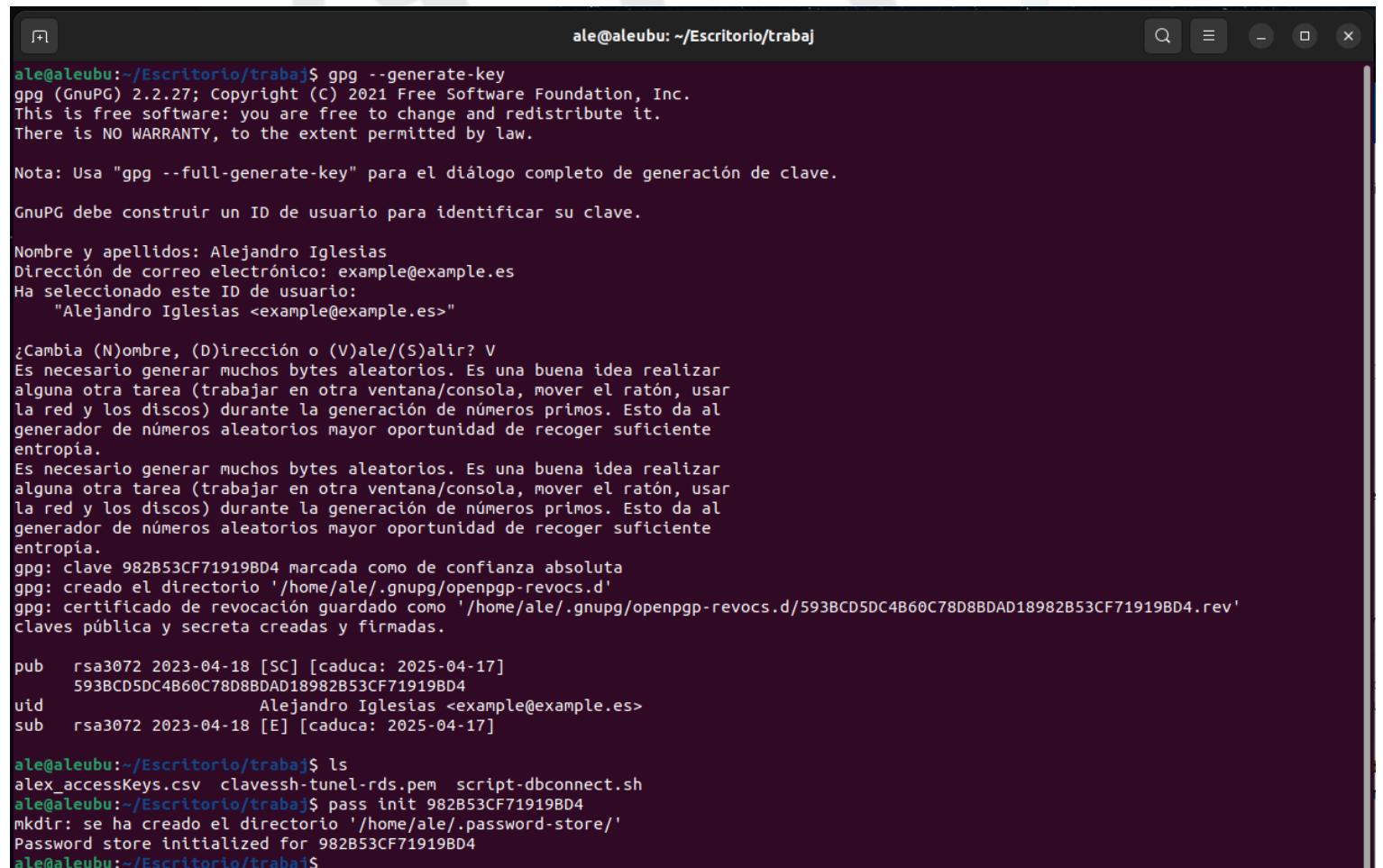
```
docker tag apache-php:1  
484992678894.dkr.ecr.eu-south-2.amazonaws.com/repo-footpassion:1
```



```
ale@aleubu:~$ docker images | grep "apache"  
apache-php 0754010d35b6 43 hours ago 410MB 1  
ale@aleubu:~$ docker tag apache-php:1 484992678894.dkr.ecr.eu-south-2.amazonaws.com/repo-footpassion:1  
ale@aleubu:~$
```

Ya tenemos la imagen con el tag, pero antes de enviarla al repositorio debemos configurar docker con las credenciales necesarias en docker login para poder subir elementos al repositorio.

Para ello, se empieza inicializando pass, que es lo que utiliza docker para almacenar las credenciales. Para ello generamos una clave e inicializamos pass con dicha clave creada.

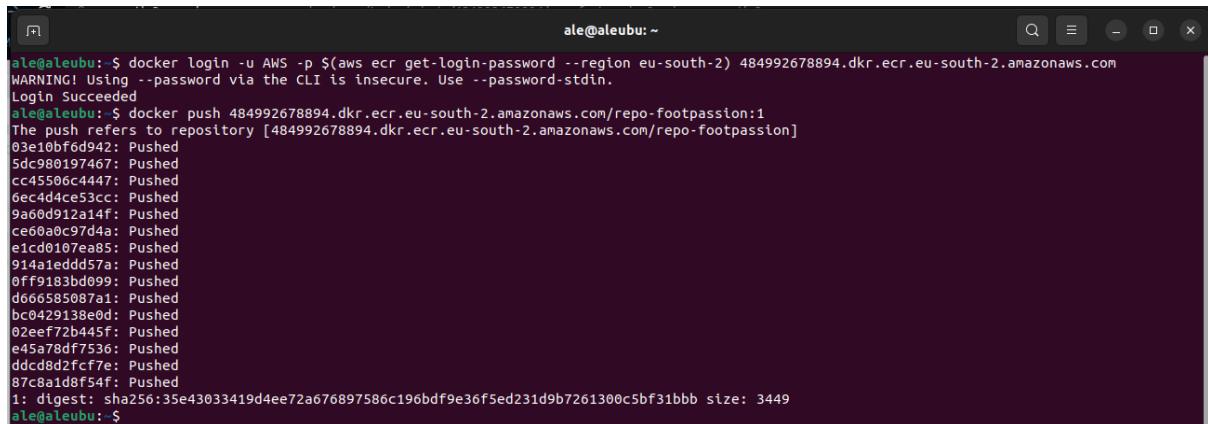


```
ale@aleubu:~/Escritorio/trabaj$ gpg --generate-key  
gpg (GnuPG) 2.2.27; Copyright (C) 2021 Free Software Foundation, Inc.  
This is free software: you are free to change and redistribute it.  
There is NO WARRANTY, to the extent permitted by law.  
  
Nota: Usa "gpg --full-generate-key" para el diálogo completo de generación de clave.  
  
GnuPG debe construir un ID de usuario para identificar su clave.  
  
Nombre y apellidos: Alejandro Iglesias  
Dirección de correo electrónico: example@example.es  
Ha seleccionado este ID de usuario:  
"Alejandro Iglesias <example@example.es>"  
  
¿Cambia (N)ombre, (D)irección o (V)ale/(S)alir? V  
Es necesario generar muchos bytes aleatorios. Es una buena idea realizar  
alguna otra tarea (trabajar en otra ventana/consola, mover el ratón, usar  
la red y los discos) durante la generación de números primos. Esto da al  
generador de números aleatorios mayor oportunidad de recoger suficiente  
entropía.  
Es necesario generar muchos bytes aleatorios. Es una buena idea realizar  
alguna otra tarea (trabajar en otra ventana/consola, mover el ratón, usar  
la red y los discos) durante la generación de números primos. Esto da al  
generador de números aleatorios mayor oportunidad de recoger suficiente  
entropía.  
gpg: clave 982B53CF71919BD4 marcada como de confianza absoluta  
gpg: creado el directorio '/home/ale/.gnupg/openpgp-revocs.d'  
gpg: certificado de revocación guardado como '/home/ale/.gnupg/openpgp-revocs.d/593BCD5DC4B60C78D8BDAD18982B53CF71919BD4.rev'  
claves pública y secreta creadas y firmadas.  
  
pub rsa3072 2023-04-18 [SC] [caduca: 2025-04-17]  
593BCD5DC4B60C78D8BDAD18982B53CF71919BD4  
uid Alejandro Iglesias <example@example.es>  
sub rsa3072 2023-04-18 [E] [caduca: 2025-04-17]  
  
ale@aleubu:~/Escritorio/trabaj$ ls  
alex.accessKeys.csv clavessh-tunel-rds.pem script-dbconnect.sh  
ale@aleubu:~/Escritorio/trabaj$ pass init 982B53CF71919BD4  
mkdir: se ha creado el directorio '/home/ale/.password-store/'  
Password store initialized for 982B53CF71919BD4  
ale@aleubu:~/Escritorio/trabaj$
```

Ahora ya podremos recoger las credenciales, guardarlas en docker login y poder identificarnos en la cli para hacer el push:

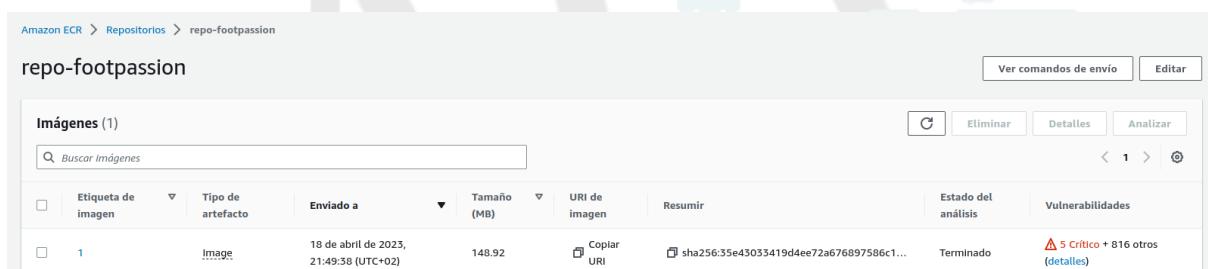
```
sudo docker login -u AWS -p $(aws ecr get-login-password --region eu-south-2)  
484992678894.dkr.ecr.eu-south-2.amazonaws.com
```

```
docker push 484992678894.dkr.ecr.eu-south-2.amazonaws.com/repo-footpassion:1
```



```
ale@aleubu:~$ docker login -u AWS -p $(aws ecr get-login-password --region eu-south-2) 484992678894.dkr.ecr.eu-south-2.amazonaws.com  
WARNING! Using --password via the CLI is insecure. Use --password-stdin.  
Login Succeeded  
ale@aleubu: $ docker push 484992678894.dkr.ecr.eu-south-2.amazonaws.com/repo-footpassion:1  
The push refers to repository [484992678894.dkr.ecr.eu-south-2.amazonaws.com/repo-footpassion]  
03e10bffd942: Pushed  
5dc980197467: Pushed  
cc45506c4447: Pushed  
6ec4d4ce53cc: Pushed  
9a00d912a14f: Pushed  
ce60a0c97d4a: Pushed  
e1cd0107ea85: Pushed  
914a1edd57a: Pushed  
0ff9183bd099: Pushed  
d666585087a1: Pushed  
bc0429138e0d: Pushed  
02eef72b445f: Pushed  
e45a78df7536: Pushed  
ddcd8d2fcf7e: Pushed  
87c8a1d8f54f: Pushed  
1: digest: sha256:35e43033419d4ee72a676897586c196bdf9e36f5ed231d9b7261300c5bf31bbb size: 3449  
ale@aleubu: $
```

Ya tenemos nuestra imagen para usar en el clúster.



Imágenes (1)						
Etiqueta de imagen		Tipo de artefacto	Enviado a	Tamaño (MB)	URI de imagen	Estado del análisis
<input type="checkbox"/>	1	image	18 de abril de 2023, 21:49:38 (UTC+02)	148.92	Copiar URI sha256:35e43033419d4ee72a676897586c1...	Terminado

CREACIÓN DEL CLÚSTER Y DEFINICIÓN DE TAREAS

En resumen, en esta parte crearemos el clúster con todos los sistemas que permiten la automatización del despliegue.

En primer lugar se crea el clúster, el cual por sí solo no hace nada, pero hay que ir especificando lo que va a contener. Este cluster está dentro de nuestra VPC regional, actuando simultáneamente en las 3 subredes creadas.

El clúster cluster-footpassion se ha creado correctamente.

Amazon Elastic Container Service > Clústeres

Clústeres (1) Información

Buscar en clústeres

Clúster	Servicios	Tareas	Instancias de contenedor registradas	Supervisión de CloudWatch	Estrategia de proveedor de capacidad
cluster-footpassion	0	No hay tareas en ejecución.	0	Valor predeterminado	No se ha encontrado ningún valor predeterminado

< 1 > ⌂

En segundo lugar se crean las definiciones de tareas, dónde cada tarea es un elemento diferente que se ejecutará dentro de una instancia de EC2 diferente. A partir de esa definición se crea un servicio, pues necesitamos una orquestación de tarea prolongada.

Para ello se comienza colocando nombre, uri de imagen y el puerto que se va a mapear. Este mapeo es doble, es decir, que mapea el puerto 80 del contenedor con el 80 de la máquina que se ejecute.

Amazon Elastic Container Service > Crear una nueva definición de tarea

Paso 1 Configurar la definición de tarea y los contenedores

Paso 2 Configurar el entorno, el almacenamiento, el monitoreo y las etiquetas

Paso 3 Revisar y crear

Configurar la definición de tarea y los contenedores

Configuración de definición de tareas

Familia de definición de tareas [Información](#)
Especifique un nombre de familia de definición de tarea único.
despliegue_web
Se permiten hasta 255 letras (mayúsculas y minúsculas), números, guiones y guiones bajos.

Contenedor: 1 [Información](#)

Contenedor esencial Eliminar

Detalles del contenedor
Especifique un nombre, una imagen de contenedor y si el contenedor debe marcarse como esencial. Cada definición de tarea debe tener al menos un contenedor esencial.

Nombre	URI de imagen	Contenedor esencial
aplicación	484992678894.dkr.ecr.eu-south-2.amazonaws.com/rep	Sí

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

Mapeos de puertos [Información](#)
Agregue mapeos de puertos para permitir que el contenedor obtenga acceso a los puertos del host para enviar o recibir tráfico. Cualquier cambio en la configuración de mapeos de puertos afecta a la configuración de conexión del servicio asociado.

Puerto del contenedor	Protocolo	Eliminar
80	TCP	

Primero se configura el entorno de la máquina, con los recursos que va a usar y los roles a usar, que serán los que aws de forma predeterminada tienen para poder acceder a servicios comunes. En este caso, cada tarea consta de 2 CPU y 4GB de memoria, todo asignado al contenedor de la aplicación, pues es el único que estará en la tarea.

Estos son recursos bastante bajos, pero se ha de tener en cuenta que se van a ejecutar muchas tareas dependiendo de la carga de trabajo.

▼ Entorno
Especifique los requisitos de infraestructura para la definición de tarea.

Tamaño de la tarea | [Información](#)
Especifique la cantidad de CPU y memoria que reservar para su tarea.

CPU	Memoria
2 vCPU	4 GB

▼ Tamaño del contenedor - opcional [Información](#)

Para cada contenedor asociado a la tarea, especifique la CPU y la memoria en el nivel del contenedor.

Contenedor	CPU (en vCPU)	Memoria (en GB)	GPU	
aplicacion	2	4	0	Eliminar

[Agregar](#)

▼ Roles de tarea- condicionales

Rol de tarea | [Información](#)
Un rol de IAM de tarea permite a los contenedores de la tarea realizar solicitudes de API a los servicios de AWS. Puede crear un rol de IAM de tarea desde la [consola de IAM](#).

ecsTaskExecutionRole

Rol de ejecución de tareas | [Información](#)
El agente de contenedor utiliza un rol de IAM de ejecución de tareas para realizar solicitudes a la API de AWS en su nombre. Si aún no tiene un rol de IAM de ejecución de tareas creado, podemos crear uno por usted.

ecsTaskExecutionRole

En cuanto se crea se ha de revisar que en ambos sitios de roles se esté configurado el ecsTaskExecutionRole, para que tenga permisos para poder acceder al registro de imágenes ECR.

En el apartado de volumen se debe configurar como vamos a montar el EFS que creamos anteriormente, usando todo el volumen para montarlo en la carpeta donde están definidas las imágenes según el código de la aplicación (/var/www/html/img/).

▼ Volumen: 1 Eliminar

Agregar y configurar almacenamiento
Agregue uno o varios volúmenes de datos a su tarea para proporcionar almacenamiento adicional para los contenedores de la tarea. Para cada volumen de datos, debe agregar un punto de montaje para especificar dónde montar el volumen de datos en el contenedor.

Tipo de volumen | [Información](#)
 ▾

Configuraciones de almacenamiento

Nombre del volumen Información <input type="text" value="fotos"/>	ID del sistema de archivos Información Crear nuevo en la consola de Amazon EFS  <input type="text" value="fs-095ef927774d7b6c2"/> ▾ 
Directorio raíz Información Directorio en EFS. <input type="text" value="/"/>	ID de punto de acceso Información Crear nuevo en la consola de Amazon EFS  <input type="text" value="Ninguno"/> ▾ 

Configuraciones avanzadas

Puntos de montaje de contenedor [Información](#)

Para cada volumen de datos asociado a la tarea, agregue un punto de montaje de contenedor para determinar dónde se monta el volumen de datos.

Contenedor <input type="text" value="aplicacion"/> ▾	Volumen de origen <input type="text" value="fotos"/> ▾	Ruta del contenedor <input type="text" value="/var/www/html/"/>	Solo lectura <input type="text" value="No"/> ▾ Eliminar
--	--	---	--

[Añadir punto de montaje](#)

Una vez creada la configuración de tarea, para poder aplicarla y ejecutarla se debe ejecutar en forma de tarea o servicio, como se explicó anteriormente. En nuestro caso definiremos un servicio. ¿Por qué?, pues porque un servicio es una tarea pero que se prolonga en el tiempo y tiene un estado definido que se tiene que mantener (a diferencia de las tareas simples que comienzan y acaban).

Para ello en primer lugar crearemos el Grupo de seguridad: permitir tráfico web de entrada desde cualquier sitio.

Detalles básicos

Nombre del grupo de seguridad [Información](#)
ECS
El nombre no se puede editar después de su creación.

Descripción [Información](#)
Seguridad de ECS

VPC [Información](#)
Q vpc-057b7e2e072b13d40

Reglas de entrada [Información](#)

Tipo	Información	Protocolo	Información	Intervalo de puertos	Información	Origen	Información	Descripción: opcional	Información
HTTP	▼	TCP		80		Anywh...	▼	<input type="text"/>	X
HTTPS	▼	TCP		443		Anywh...	▼	<input type="text"/>	X

[Agregar regla](#)

Una vez creado este ya podemos

cluster-footpassion

[C](#) [Actualizar el clúster](#) [Eliminar clúster](#)

Información general sobre el clúster

ARN cluster-footpassion	Estado Activo	Supervisión de CloudWatch Valor predeterminado	Instancias de contenedor registradas -
Servicios		Tareas	
Vaciando	-	Pendiente	Ejecutando

[Servicios](#) [Tareas](#) [Infraestructura](#) [Métricas](#) [Tareas programadas](#) [Etiquetas](#)

Servicios (0) [Información](#)

Nombre del servicio	Estado	ARN	Tipo de ser...	Implementaciones y tareas	Última impl...	Definición ...	Revisión
No hay servicios No hay servicios que mostrar.							

[Crear](#)

Para ello se especifica en primer lugar la definición de tarea que vamos a utilizar, con la versión de la misma.

Configuración de implementación

Tipo de aplicación | [Información](#)
Especifique el tipo de aplicación que desea ejecutar.

Servicio
Lance un grupo de tareas que gestionen un trabajo informático de ejecución prolongada que se pueda detener y reiniciar. Por ejemplo, una aplicación web.

Tarea
Lance una tarea independiente que se ejecute y finalice. Por ejemplo, un trabajo por lotes.

Definición de tarea
Seleccione una definición de tarea existente. Para crear una nueva definición de tarea, vaya a [Definiciones de tareas](#).

Especificar la revisión manualmente
Ingrese manualmente la revisión en lugar de elegir entre las 100 revisiones más recientes para la familia de definición de tareas seleccionada.

Familia	Revisión
despliegue_web	1 (MÁS RECIENTE)

Nombre del servicio
Asigne un nombre único a este servicio.

Tipo de servicio | [Información](#)
Especifique el tipo de servicio que seguirá el programador de servicio.

Réplica
Coloque y mantenga un número deseado de tareas en su clúster.

Daemon
Coloque y mantenga una copia de la tarea en cada instancia del contenedor.

A continuación establecemos el servicio en las 3 subredes disponibles y le asignamos el grupo de seguridad creado anteriormente

▼ **Redes**

VPC | Información
Elija la nube virtual privada que desea utilizar.

vpc-057b7e2e072b13d40
valor predeterminado

Subredes
Elija las subredes dentro de la VPC que el programador de tareas debe tener en cuenta para su ubicación.

Elegir subredes

subnet-02d5bcc5b9474bca X eu-south-2a subnet-0f92359bc314b6585 X eu-south-2b

subnet-0c28bc8e106a30c8b X eu-south-2c

Grupo de seguridad | Información
Elija un grupo de seguridad existente o cree uno nuevo.

Utilizar un grupo de seguridad existente
 Crear un nuevo grupo de seguridad

Nombre del grupo de seguridad
Elija un grupo de seguridad existente.

sg-0e09e3eab33f6e2cb X ECS

IP pública | Información
Elija si desea asignar automáticamente una IP pública a la interfaz de red elástica (ENI) de la tarea.

Activado

Configuramos el balanceo de carga entre los contenedores de la definición ed tarea que creamos antes, usando el balanceador de aplicación (se puede escoger el de nivel de red que es más complejo y caro), pero este último en nuestro caso no aportaría ninguna ventaja operativa).

Se usa como grupo de destino uno nuevo que se crea automáticamente y que registrará los recursos que se vayan utilizando en el servicio.

▼ Balanceo de carga - *opcional*

Tipo de balanceador de carga | [Información](#)
Configure un balanceador de carga para distribuir el tráfico entrante entre las tareas que se ejecutan en el servicio.

Balanceador de carga de aplicaciones

Balanceador de carga de aplicaciones
Especifique si desea crear un nuevo balanceador de carga o elegir uno existente.

Crear un nuevo balanceador de carga

Usar un balanceador de carga existente

Nombre del balanceador de carga
Asigne un nombre único al balanceador de carga.

Servicio-Web-Footpassion

Elegir el contenedor para平衡ear la carga

aplicacion 80:80

Agente de escucha | [Información](#)
Especifique el puerto y el protocolo en los que el balanceador de carga escuchará las solicitudes de conexión.

Crear nuevo agente de escucha

Utilizar un agente de escucha existente
Debe seleccionar un equilibrador de carga existente.

Puerto	Protocolo
<input type="text" value="80"/>	<input type="button" value="HTTP"/>

Grupo de destino | [Información](#)
Especifique si desea crear un nuevo grupo de destino o elegir uno existente que el equilibrador de carga utilizará para dirigir las solicitudes a las tareas del servicio.

Crear nuevo grupo de destino

Utilizar un grupo de destino existente
Debe seleccionar un equilibrador de carga existente.

Nombre del grupo de destino	Protocolo
<input type="text" value="Web-footpassion"/>	<input type="button" value="HTTP"/>

Ruta de comprobación de estado | [Información](#)

Protocolo de comprobación de estado

Periodo de gracia de comprobación de estado | [Información](#)

segundos

Por último pero no menos importante se crea el escalado automático de nuestro servicio, una manera de que se vayan levantando contenedores conforme es necesario.

Para ello definimos unos mínimos y máximos de elementos levantados, así como la estrategia de control del escalado, que en este caso será el uso de CPU, es decir, que al superar el 70 % de CPU, se creará un nuevo contenedor dentro del grupo, entrando en el balanceador de carga, etc...

▼ Escalado automático de servicios - *opcional*

Ajuste automáticamente el recuento deseado del servicio hacia arriba y hacia abajo dentro de un rango especificado en respuesta a las alarmas de CloudWatch. Puede modificar la configuración del escalado automático de servicios en cualquier momento para satisfacer las necesidades de la aplicación.

Utilizar el escalado automático del servicio
Configurar el escalado automático de servicios para ajustar el recuento deseado del servicio

Cantidad mínima de tareas
El límite inferior al que el escalado automático de servicios puede ajustar el recuento deseado del servicio.

Cantidad máxima de tareas
El límite superior al que el escalado automático de servicios puede ajustar el recuento deseado del servicio.

Tipo de política de escalado | [Información](#)
Cree una política de seguimiento de destino o de escalado por pasos.

Seguimiento de destino
Aumente o reduzca el número de tareas que el servicio ejecuta en función de un valor de destino para una métrica específica.

Escalado por pasos
Utilice la consola clásica para aumentar o disminuir la cantidad de tareas que ejecuta el servicio en función de un conjunto de ajustes de escala, conocidos como ajustes de paso, que varían en función del tamaño de la interrupción de alarma.

Nombre de política

Métrica de servicio de ECS

Valor de destino

Periodo de recuperación de escalado horizontal

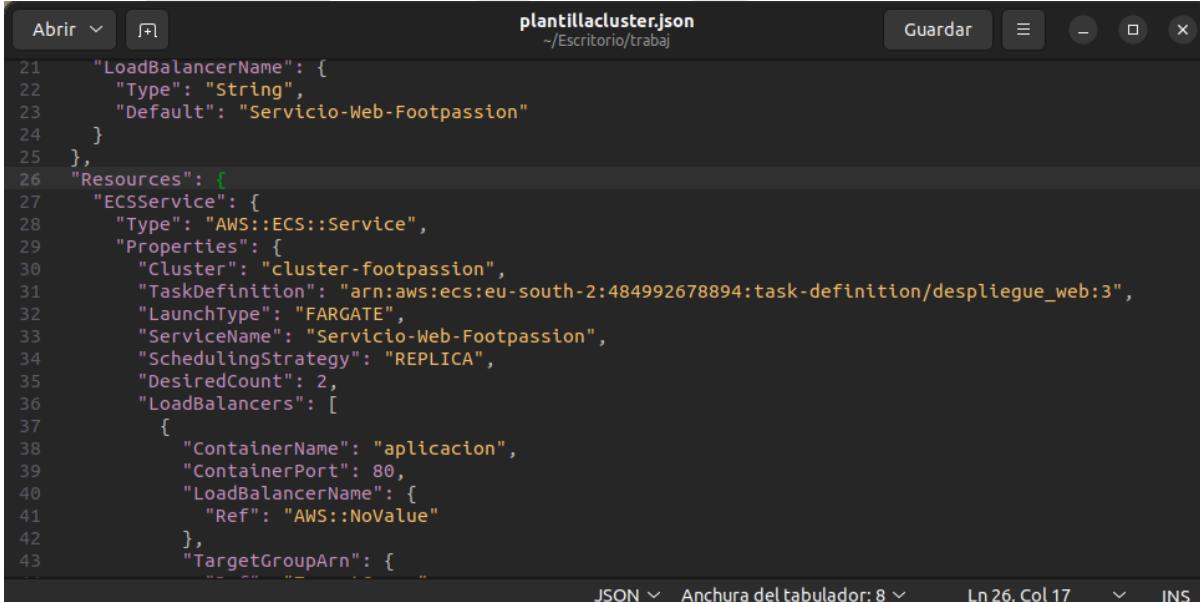
Periodo de recuperación de desescalado horizontal

Desactivar la acción de desescalar horizontalmente

A la hora del despliegue, sin nosotros solicitarlo, se está utilizando el servicio de Cloud Formation (Infrastructure as Code de AWS) para desplegar de 1 sola vez muchos recursos diferentes:

- Balanceador de carga
- Grupos de seguridad
- Servicio de ECS
- Montaje de archivos
- Grupos de destino
-

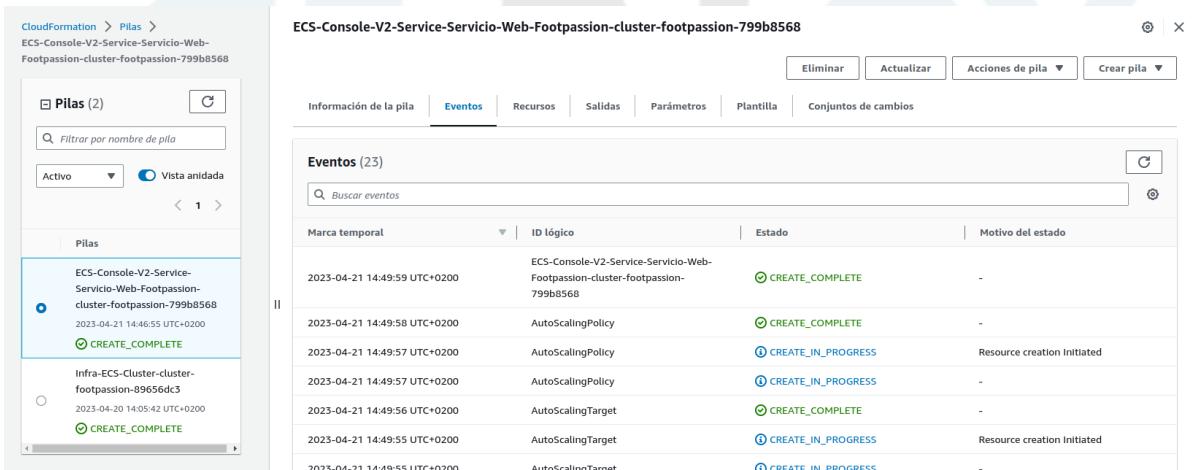
Dentro de la etapa de desarrollo, se usó la plantilla de Cloud Formation para guardarla y así poder eliminar el servicio cuando terminen las pruebas (así no gasta recursos), pudiendo levantar de nuevo la infraestructura importando el archivo simplemente.



```

  "LoadBalancerName": {
    "Type": "String",
    "Default": "Servicio-Web-Footpassion"
  },
  "Resources": {
    "ECSService": {
      "Type": "AWS::ECS::Service",
      "Properties": {
        "Cluster": "cluster-footpassion",
        "TaskDefinition": "arn:aws:ecs:eu-south-2:484992678894:task-definition/despliegue_web:3",
        "LaunchType": "FARGATE",
        "ServiceName": "Servicio-Web-Footpassion",
        "SchedulingStrategy": "REPLICAS",
        "DesiredCount": 2,
        "LoadBalancers": [
          {
            "ContainerName": "aplicacion",
            "ContainerPort": 80,
            "LoadBalancerName": {
              "Ref": "AWS::NoValue"
            },
            "TargetGroupArn": {
              "Fn::GetAtt": [
                "LoadBalancerName"
              ]
            }
          }
        ]
      }
    }
  }
}

```



The screenshot shows the AWS CloudFormation console interface. On the left, there's a sidebar with navigation links like 'CloudFormation > Pilas > ECS-Console-V2-Service-Servicio-Web-Footpassion-cluster-footpassion-799b8568'. The main area displays a table of events for the stack 'ECS-Console-V2-Service-Servicio-Web-Footpassion-cluster-footpassion-799b8568'. The table has columns for 'Marca temporal' (Timestamp), 'ID lógico' (Logical ID), 'Estado' (State), and 'Motivo del estado' (Reason). The events listed are all in 'CREATE_COMPLETE' state, indicating successful creation of resources. Some events have a note: 'Resource creation initiated'.

Marca temporal	ID lógico	Estado	Motivo del estado
2023-04-21 14:49:59 UTC+0200	ECS-Console-V2-Service-Servicio-Web-Footpassion-cluster-footpassion-799b8568	CREATE_COMPLETE	
2023-04-21 14:49:58 UTC+0200	AutoScalingPolicy	CREATE_COMPLETE	
2023-04-21 14:49:57 UTC+0200	AutoScalingPolicy	CREATE_IN_PROGRESS	Resource creation initiated
2023-04-21 14:49:57 UTC+0200	AutoScalingPolicy	CREATE_IN_PROGRESS	
2023-04-21 14:49:56 UTC+0200	AutoScalingTarget	CREATE_COMPLETE	
2023-04-21 14:49:55 UTC+0200	AutoScalingTarget	CREATE_IN_PROGRESS	Resource creation initiated
2023-04-21 14:49:55 UTC+0200	AutoScalingTarget	CREATE_IN_PROGRESS	

Servicio-Web-Footpassion se ha implementado correctamente.

Amazon Elastic Container Service > Clústeres > cluster-footpassion > Servicios > Servicio-Web-Footpassion > Estado

Servicio-Web-Footpassion Información

C Actualizar servicio Eliminar el servicio

Estado y métricas | Registros | Configuración y tareas | Implementaciones y eventos | Redes | Etiquetas

Estado información

ARN	Estado	Tareas	Estado actual de las implementaciones
cluster-footpassion/Servicio-Web-Footpassion	Activó	0 pendiente/s, 2 en ejecución / 2 deseado(s)	2 completado(s)
▼ Estado del equilibrador de carga			
Nombre del balanceador de carga	Destinos totales	Destinos en buen estado	Destinos en mal estado
Servicio-Web-Footpassion	2	2	0

Aquí observamos el resultado final de la implementación completa, por ahora no muy efectiva pues no tiene la aplicación dentro. Pero ese es el siguiente paso.

No es seguro | servicio-web-footpassion:1230042215.eu-south-2.elb.amazonaws.com

PHP Version 7.2.34

System Linux ip-172-31-33-252.eu-south-2.compute.internal 5.10.173-154.642.amzn2.x86_64 #1 SMP Wed Mar 15 00:26:42 UTC 2023 x86_64

Build Date Dec 11 2020 10:50:00

Configure Command ./configure --build=x86_64-linux-gnu --with-config-file-path=/etc/php --with-config-file-scan-dir=/etc/php/conf.d --enable-option-checking=fatal --with-mysqli --with-pdo-mysql --enable-fpm --mbstring --enable-mbregex --with-password-argon2 --with-sodium=shared --with-pdo-sqlite=tsrmi --with-sqlite3=tsrmi --with-curl --with-bz2 --with-zip --with-libxml=ibxml86_64-linux-gnu --with-apxs2 --disable-rpath --with-xpm-dir=/usr/include/X11 --with-xpm=shared

Server API Apache 2.0 Handler

Virtual Directory Support disabled

Configuration File (php.ini) Path /etc/focalstack/php

Loaded Configuration File (none)

Scan this dir for additional .ini files /etc/focalstack/php/conf.d

Additional .ini files parsed /etc/focalstack/php/conf.d/docker-php-ext-sodium.ini

PHP API 20170718

PHP Extension 20170718

Zend Extension 320170718

Zend Extension Build APIS20170718.NTS

PHP Extension Build APIS20170718.NTS

Debug Build no

Thread Safety disabled

Zend Signal Handling enabled

Zend Memory Manager enabled

Zend Multibyte Support provided by mbstring

IPv6 Support enabled

DTrace Support disabled

Registered PHP Streams https, https, compress,zlib, php, file, glob, data, http, ftp, phar

Registered Stream Socket Transports tcp, udp, unix, udg, ssl, lls, tlsv1.0, tlsv1.1, tlsv1.2

Registered Stream Filters zlib*, convert.iconv*, string.rot13, string.toupper, string.tolower, string.strip_tags, convert*, consumed, dechunk

This program makes use of the Zend Scripting Language Engine: Zend Engine v3.2.0. Copyright (c) 1998-2018 Zend Technologies

zendengine

▼ Distribution of targets by Availability Zone (AZ)

Select values in this table to see corresponding filters applied to the Registered targets table below.

Last fetched seconds ago						
Zone	Total targets	Healthy	Unhealthy	Unused	Initial	Draining
eu-south-2a	1	1	0	0	0	0
eu-south-2b	1	1	0	0	0	0

Targets | Monitoring | Health checks | Attributes | Tags

Registered targets (2)

C Deregister Register targets

Filter resources by property or value

IP address	Port	Zone	Health status	Health status details
172.31.38.204	80	eu-south-2a	healthy	
172.31.18.85	80	eu-south-2b	healthy	

SISTEMA DE CI-CD

El llamado “Continuous Integration, continuous distribution”, se trata de una manera automatizada de poder desplegar el código de nuevas versiones a las aplicaciones de producción, así como la ejecución de un pipeline para cualquier tipo de implementaciones que requieran continuidad y automatización.

El proceso que se usará en esta sección para la automatización del código y el despliegue es el siguiente:

- Creación de repositorio privado en Git
- Subida de la aplicación a Git
- GitHub Actions

REPOSITORIO DE GIT

Se crea en primer lugar un repositorio privado al que se darán credenciales a los desarrolladores para poder crear nuevas versiones.



SUBIDA DEL CÓDIGO A GIT

En primer lugar creamos unos tokens de acceso personales para los miembros de desarrollo.

A screenshot of the GitHub developer settings page under "Personal access tokens (classic)". It shows a sidebar with options like GitHub Apps, OAuth Apps, Personal access tokens (selected), Fine-grained tokens (Beta), and Tokens (classic). The main area has a "Generate new token" button and instructions for generating tokens for scripts or testing. It also explains that personal access tokens function like OAuth access tokens.

GitHub Apps
OAuth Apps
Personal access tokens Beta
Fine-grained tokens
Tokens (classic)

New personal access token (classic)

Personal access tokens (classic) function like ordinary OAuth access tokens. They can be used instead of a password for Git over HTTPS, or can be used to [authenticate to the API over Basic Authentication](#).

Note

Acceso a la aplicación de footpassion.

What's this token for?

Expiration *

No expiration The token will never expire!

GitHub strongly recommends that you set an expiration date for your token to help keep your information secure.
[Learn more](#)

Select scopes

Scopes define the access for personal tokens. [Read more about OAuth scopes](#).

<input checked="" type="checkbox"/> repo	Full control of private repositories
<input type="checkbox"/> repo:status	Access commit status
<input type="checkbox"/> repo_deployment	Access deployment status
<input type="checkbox"/> public_repo	Access public repositories
<input type="checkbox"/> repo:invite	Access repository invitations
<input type="checkbox"/> security_events	Read and write security events

GitHub Apps
OAuth Apps
Personal access tokens Beta
Fine-grained tokens
Tokens (classic)

Personal access tokens (classic)

Generate new token ▾

Revoke all

Tokens you have generated that can be used to access the [GitHub API](#).

Make sure to copy your personal access token now. You won't be able to see it again!

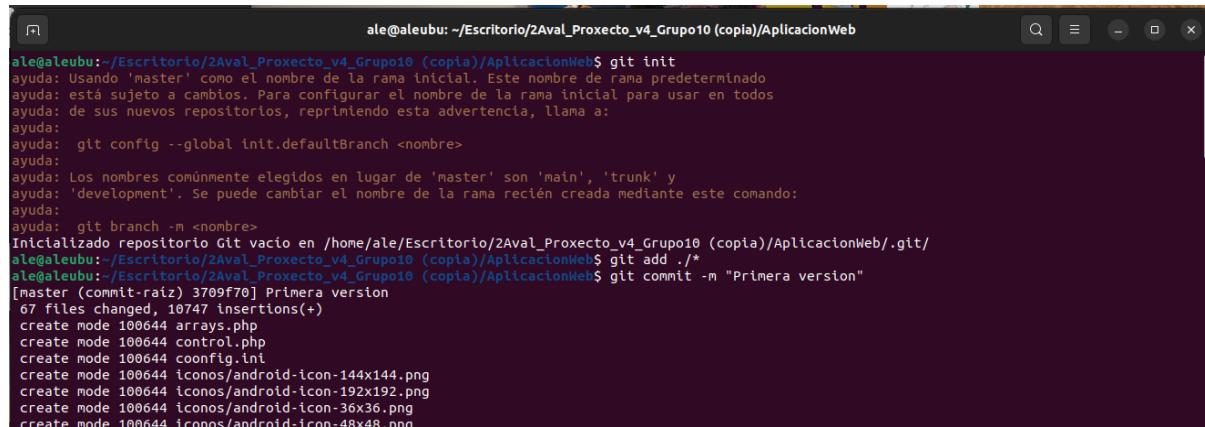
045zd 

Delete

Personal access tokens (classic) function like ordinary OAuth access tokens. They can be used instead of a password for Git over HTTPS, or can be used to [authenticate to the API over Basic Authentication](#).

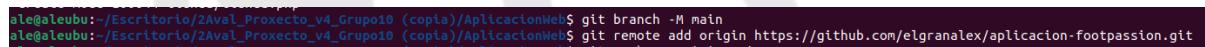
Y dichos tokens son los que se usan para autenticarse a la hora de iniciar un repositorio git en local y hacer el git push al repositorio de GitHub

En primer lugar inicializamos git en el directorio de trabajo donde tenemos el código, añadimos todos los archivos de la carpeta y hacemos commit de los cambios.



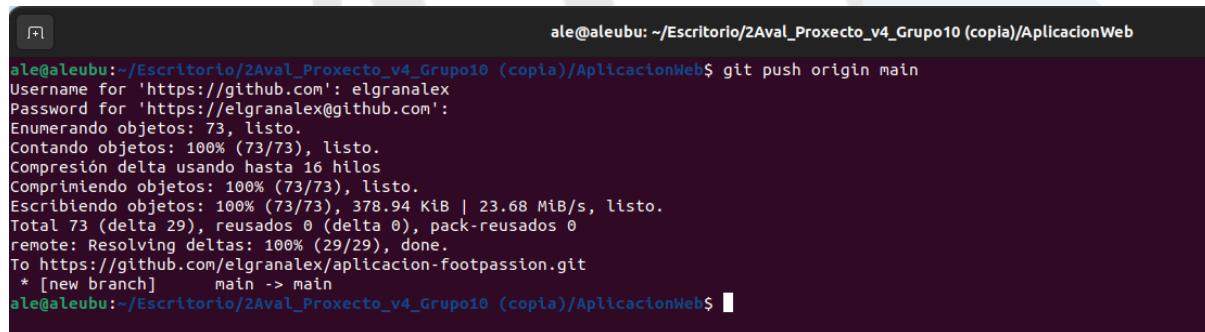
```
ale@aleubu:~/Escritorio/2Aval_Proyecto_v4 Grupo10 (copia)/AplicacionWeb$ git init
ayuda: Usando 'master' como el nombre de la rama inicial. Este nombre de rama predeterminado
ayuda: está sujeto a cambios. Para configurar el nombre de la rama inicial para usar en todos
ayuda: de sus nuevos repositorios, reprimiendo esta advertencia, llama a:
ayuda: git config --global init.defaultBranch <nombre>
ayuda:
ayuda: Los nombres comúnmente elegidos en lugar de 'master' son 'main', 'trunk' y
ayuda: 'development'. Se puede cambiar el nombre de la rama recién creada mediante este comando:
ayuda:
ayuda: git branch -m <nombre>
Inicializado repositorio Git vacío en /home/ale/Escritorio/2Aval_Proyecto_v4 Grupo10 (copia)/AplicacionWeb/.git/
ale@aleubu:~/Escritorio/2Aval_Proyecto_v4 Grupo10 (copia)/AplicacionWeb$ git add ./*
ale@aleubu:~/Escritorio/2Aval_Proyecto_v4 Grupo10 (copia)/AplicacionWeb$ git commit -m "Primera version"
[master (commit-raíz) 3709f70] Primera version
 67 files changed, 10747 insertions(+)
 create mode 100644 arrays.php
 create mode 100644 control.php
 create mode 100644 config.ini
 create mode 100644 iconos/android-icon-144x144.png
 create mode 100644 iconos/android-icon-192x192.png
 create mode 100644 iconos/android-icon-36x36.png
 create mode 100644 iconos/android-icon-48x48.png
```

Nos colocamos en la rama principal de nuestra repo de Git local y añadimos el origin de nuestro repositorio privado de github.



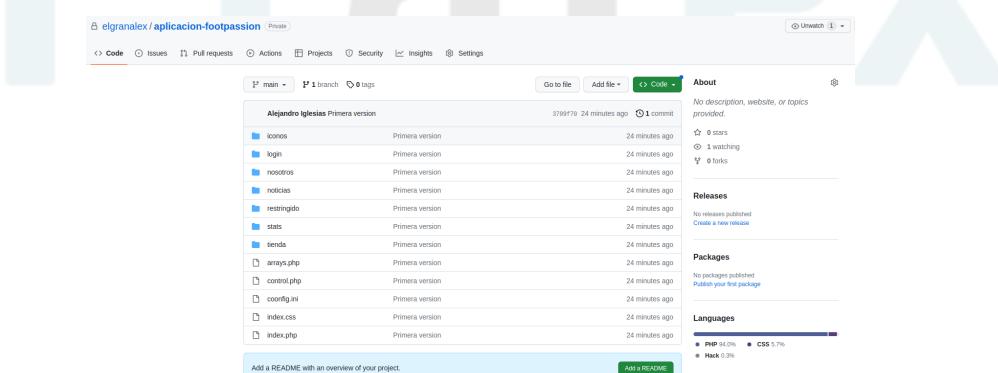
```
ale@aleubu:~/Escritorio/2Aval_Proyecto_v4 Grupo10 (copia)/AplicacionWeb$ git branch -M main
ale@aleubu:~/Escritorio/2Aval_Proyecto_v4 Grupo10 (copia)/AplicacionWeb$ git remote add origin https://github.com/elgranalex/aplicacion-footpassion.git
```

Hacemos push de los cambios al repositorio privado usando el token que conseguimos anteriormente como contraseña:

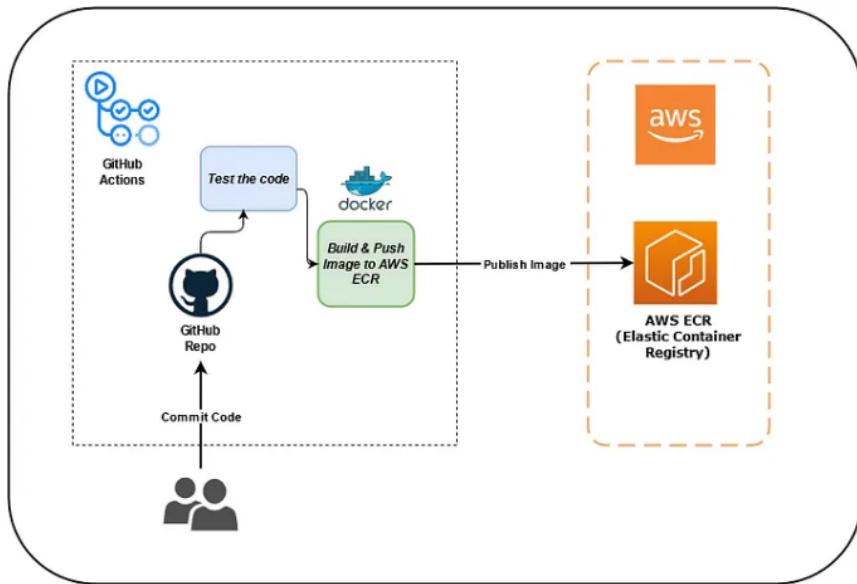


```
ale@aleubu:~/Escritorio/2Aval_Proyecto_v4 Grupo10 (copia)/AplicacionWeb$ git push origin main
Username for 'https://github.com': elgranalex
Password for 'https://elgranalex@github.com':
Enumerando objetos: 73, listo.
Contando objetos: 100% (73/73), listo.
Compresión delta usando hasta 16 hilos
Comprimiendo objetos: 100% (73/73), listo.
Escribiendo objetos: 100% (73/73), 378.94 KiB | 23.68 MiB/s, listo.
Total 73 (delta 29), reusados 0 (delta 0), pack-reusados 0
remote: Resolving deltas: 100% (29/29), done.
To https://github.com/elgranalex/aplicacion-footpassion.git
 * [new branch]      main -> main
ale@aleubu:~/Escritorio/2Aval_Proyecto_v4 Grupo10 (copia)/AplicacionWeb$
```

Ya lo tenemos visible.



GITHUB ACTIONS



GitHub Actions es una plataforma de integración continua y entrega continua (CI/CD) que le permite automatizar un pipeline de compilación, prueba e implementación. Se pueden crear flujos de trabajo que construyen y prueban cada solicitud de extracción en el repositorio, o implementar solicitudes de extracción combinadas en producción.

Esto se consigue gracias a un trigger específico que hace que salte automáticamente las acciones determinadas en una máquina virtual de github nueva y provisionada (runner), donde se ejecutarán todos los elementos especificados y se eliminará al acabar.

Ahora bien, como pequeño inciso, remarcar que la imagen usada en local será ligeramente modificada para facilitar el despliegue. Para ello simplemente quitamos del dockerfile las líneas que daban permisos:

A screenshot of a GitHub commit interface. The commit is titled 'elgranalex Update dockerfile ✓'. Below the title, there are tabs for 'Code' (selected), 'Blame', and statistics ('9 lines (6 loc) · 166 Bytes'). The code editor shows a Dockerfile with the following content:

```
1 FROM php:8.2.0-apache
2 ARG DEBIAN_FRONTEND=noninteractive
3 RUN docker-php-ext-install mysqli
4 COPY ./ /var/www/html/
5 RUN mkdir /var/www/html/img
6 RUN a2enmod rewrite
```

Para ello, se debe crear un archivo de tipo YML dentro del directorio .github/workflows. Una vez en ese archivo, se van a ir especificando los diferentes componentes y partes que se ejecutarán. Las partes del archivo son las siguientes:

1. Se le crea un nombre al action y se determina cual es el trigger que desencadenará la acción, en este caso el trigger será un push hacia la rama principal del repositorio.

```
name: Deploy to Amazon ECS

on:
  push:
    branches: [ "main" ]
```

2. En siguiente lugar establecemos las diferentes variables de entorno que se usarán en el archivo:
 - a. La región
 - b. El repositorio de ECR
 - c. El nombre del clúster
 - d. El nombre del servicio
 - e. El archivo en el que recogeremos por un comando la definición de tarea
 - f. El nombre del contenedor de la task definition sobre el que se ejecutarán los cambios
3. Definición de los permisos y del job (conjunto de pasos a seguir, se ejecutan en la máquina virtual “runner” de ubuntu)

```
permissions:
  contents: read

jobs:
  deploy:
    name: Deploy
    runs-on: ubuntu-latest
    environment: production
```

4. Steps: (Dentro de los diferentes steps, dentro del bloque de `uses`, se mencionarán unas acciones ya predefinidas por github donde nosotros simplemente añadimos los diferentes argumentos necesarios para que se ejecute

- a. Step1: Añadir una acción que añade el repositorio en la máquina virtual para poder ejecutar scripts o hacer referencia a archivos

```
steps:  
- name: Checkout  
  uses: actions/checkout@v3
```

- b. Step2: Configurar las credenciales de aws para poder acceder a los servicios posteriores.

```
- name: Configure AWS credentials  
  uses: aws-actions/configure-aws-credentials@v1  
  with:  
    aws-access-key-id: ${{ secrets.AWS_ACCESS_KEY_ID }}  
    aws-secret-access-key: ${{ secrets.AWS_SECRET_ACCESS_KEY }}  
    aws-region: ${{ env.AWS_REGION }}
```

Ahora bien, dónde se hace referencia a “secrets. (...)” se hace referencia a los secretos y variables del repositorio de github, y para poder configurarlo debemos ir a ajustes/Secrets and variables.

The screenshot shows the GitHub repository settings page for 'elgranalex / aplicacion.footpassion'. The 'Actions secrets and variables' section is highlighted. It contains a table with two columns: 'Secrets' and 'Variables'. The 'Secrets' tab is selected, showing a message: 'There are no secrets for this repository.' The 'Variables' tab is also present. On the right side of the table, there is a green button labeled 'New repository secret'.

Y creamos 2 repositorios diferentes, 1 para el AWS_ACCESS_KEY_ID y otro para el AWS_SECRET_ACCESS_KEY. Este clave -> clave secreta es la manera de conectarse a AWS para acceder a través de la CLI.

Actions secrets and variables

Secrets and variables allow you to manage reusable configuration data. Secrets are **encrypted** and are used for sensitive data. [Learn more about encrypted secrets](#). Variables are shown as plain text and are used for **non-sensitive** data. [Learn more about variables](#).

Anyone with collaborator access to this repository can use these secrets and variables for actions. They are not passed to workflows that are triggered by a pull request from a fork.

The screenshot shows the GitHub Actions interface for managing secrets. At the top, there are tabs for 'Secrets' (selected), 'Variables', and a green button for 'New repository secret'. Below the tabs, two secrets are listed: 'AWS_ACCESS_KEY_ID' and 'AWS_SECRET_ACCESS_KEY', both updated 'now'. Each secret entry has edit and delete icons.

c. Step3: Loguearnos en AWS

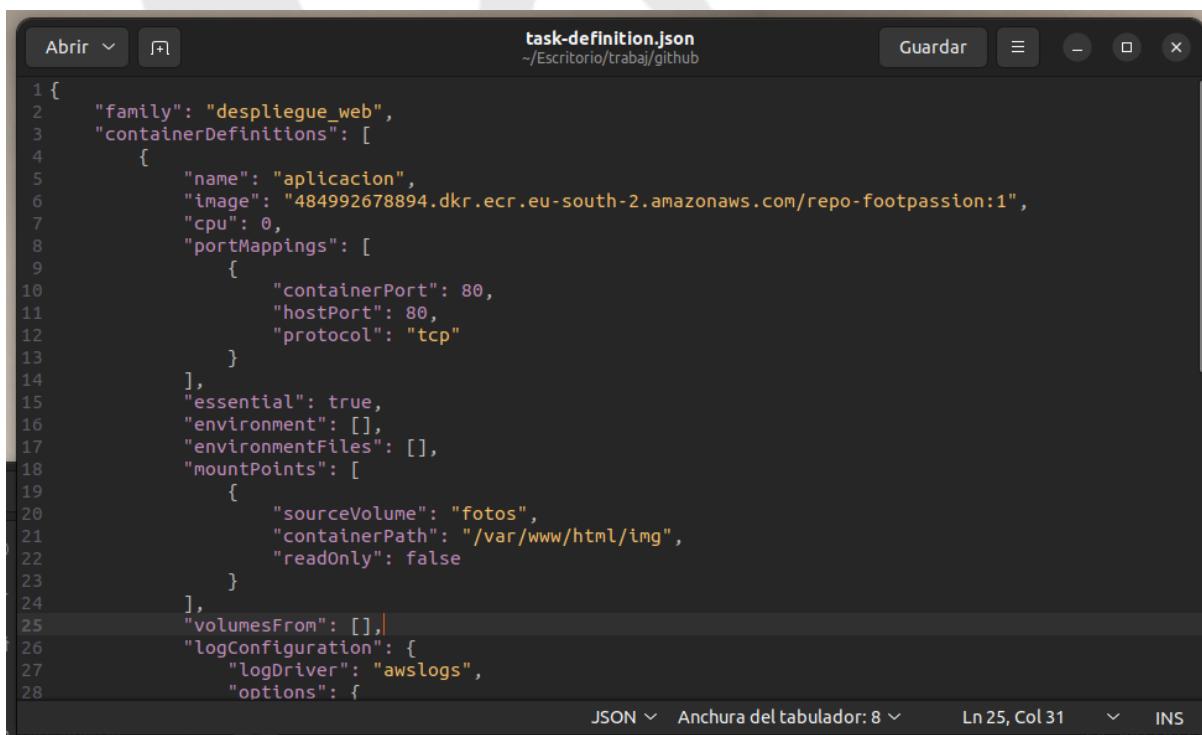
```
- name: Login to Amazon ECR
  id: login-ecr
  uses: aws-actions/amazon-ecr-login@v1
```

d. Step4: Creamos la imagen que más adelante mandaremos a nuestro repositorio. Para ello obviamente debemos incluir en la raíz del repositorio un dockerfile que especifique las condiciones de este docker build (Se creará más adelante).

```
- name: Build, tag, and push image to Amazon ECR
  id: build-image
  env:
    ECR_REGISTRY: ${{ steps.login-ecr.outputs.registry }}
    IMAGE_TAG: ${{ github.sha }}
  run:
    # Build a docker container and
    # push it to ECR so that it can
    # be deployed to ECS.
    docker build -t $ECR_REGISTRY/$ECR_REPOSITORY:$IMAGE_TAG .
    docker push $ECR_REGISTRY/$ECR_REPOSITORY:$IMAGE_TAG
    echo "image=$ECR_REGISTRY/$ECR_REPOSITORY:$IMAGE_TAG" >> $GITHUB_OUTPUT
```

e. Step6: Aquí es donde añadimos la nueva imagen en el archivo. Para ello utilizamos el archivo json task definition que tendremos que subir al repositorio. Simplemente se descarga la plantilla json de lo que creamos en la consola de aws para que se puedan modificar ciertos cambios y desplegarlos.

```
- name: Fill in the new image ID in the Amazon ECS task definition
  id: task-def
  uses: aws-actions/amazon-ecs-render-task-definition@v1
  with:
    task-definition: ${{ env.ECS_TASK_DEFINITION }}
    container-name: ${{ env.CONTAINER_NAME }}
    image: ${{ steps.build-image.outputs.image }}
```



```
task-definition.json
~/Escritorio/trabaj/github

1 {
2   "family": "despliegue_web",
3   "containerDefinitions": [
4     {
5       "name": "aplicacion",
6       "image": "484992678894.dkr.ecr.eu-south-2.amazonaws.com/repo-footpassion:1",
7       "cpu": 0,
8       "portMappings": [
9         {
10           "containerPort": 80,
11           "hostPort": 80,
12           "protocol": "tcp"
13         }
14       ],
15       "essential": true,
16       "environment": [],
17       "environmentFiles": [],
18       "mountPoints": [
19         {
20           "sourceVolume": "fotos",
21           "containerPath": "/var/www/html/img",
22           "readOnly": false
23         }
24       ],
25       "volumesFrom": [],
26       "logConfiguration": {
27         "logDriver": "awslogs",
28         "options": {}}
```

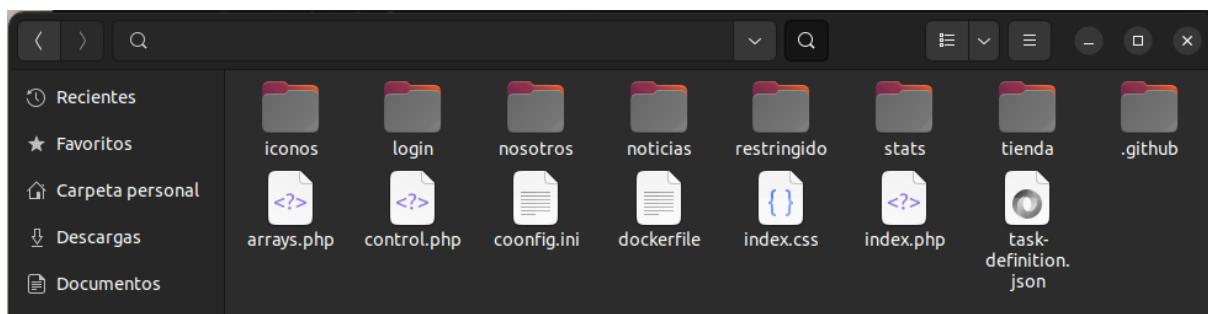
f. Step7: Por último, desplegamos la definición de tarea nueva para que se incluya en el servicio y podamos comenzar a desplegar el nuevo contenedor.

```
- name: Deploy Amazon ECS task definition
  uses: aws-actions/amazon-ecs-deploy-task-definition@v1
  with:
    task-definition: ${{ steps.task-def.outputs.task-definition }}
    service: ${{ env.ECS_SERVICE }}
    cluster: ${{ env.ECS_CLUSTER }}
    wait-for-service-stability: true
```

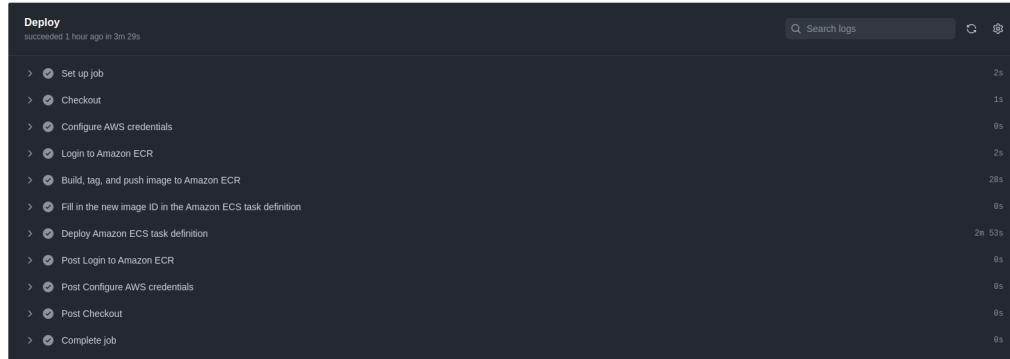
A continuación debemos crear el archivo de dockerfile que determinarán las acciones a realizar para crear la nueva imagen en la máquina virtual de github. Esta ya será la imagen que se pondrá para producción.

Simplemente cogeremos la imagen que modificamos anteriormente.

Finalmente así es como quedaría el directorio que se clonará a la rama main para realizar el primer action. A excepción del dockerfile y el directorio .github, todo lo demás son componentes de la aplicación, por lo que posteriormente los desarrolladores podrán realizar y modificar sus propias ramas, para luego hacer el merge sobre la rama principal, que cambie los archivos afectados y automáticamente se despliegue el workflow.



En el momento en el que ya lo subimos al repositorio el workflow comienza a ejecutar sus jobs.



Vemos en la foto anterior como el pipeline completo se ha ejecutado inmediatamente tras 3 min de creación y despliegue.

Consecuencia De los anteriores jobs:

- Subida de nueva imagen a nuestro repositorio.

Imágenes (2)								Ver comandos de envío	Editar
	Etiqueta de imagen	Tipo de artefacto	Envío a	Tamaño (MB)	URI de imagen	Resumir	Estado del análisis	Vulnerabilidades	
<input type="checkbox"/>	923063e7fdade488a1df828075fd51586987e5ff	Image	01 de mayo de 2023, 12:45:04 (UTC+02)	166.87	Copiar URI	sha256:2d512a186049b655f6ebae3dd4926...	Terminado	⚠ 1 Crítico + 311 otros	(detalles)
<input type="checkbox"/>	1	Image	20 de abril de 2023, 13:58:58 (UTC+02)	148.92	Copiar URI	sha256:9ca599d558d640ec7e928e9576b1b...	Terminado	⚠ 5 Crítico + 820 otros	(detalles)

- Creación de una nueva revisión de la definición de tarea para desplegar en el servicio del clúster

despliegue_web:7				Implementar	Acciones	Crear una nueva revisión
Información general Información						
ARN arn:aws:ecs:eu-south-2:484992678894:task-definition/despliegue_web:7	Estado ACTIVE		Hora de creación 1/5/2023, 10:45:05 UTC		Entorno de la aplicación FARGATE	
Rol de tarea ecsTaskExecutionRole	Rol de ejecución de tareas ecsTaskExecutionRole		Sistema operativo/arquitectura Linux/X86_64		Modo de red awsvpc	
Contenedores JSON Almacenamiento Etiquetas						
Tamaño de la tarea						
CPU de tareas 1 vCPU			Memoria de tareas 2 GB			
Contenedores Información						
Nombre del contenedor	Imagen	Private registry	Esencial	CPU	Memoria	GPU
aplicacion	484992678894.dkr.ecr.e...	-	true	0	-	-

- Despliegue de la nueva definición de tarea sobre el servicio activo.

Servicio-Web-Footpassion [Información](#)

Estado y métricas | Registros | **Configuración y tareas** | Implementaciones y eventos | Redes | Etiquetas

Configuración del servicio [Información](#)

ARN del servicio: cluster-footpassion/Servicio-Web-Footpassion

Definición de tarea: revisión despliegue_web:7

Tipo de lanzamiento: FARGATE

Tipo de servicio: REPLICAS

Creado por: arn:aws:iam::484992678894:user/allex

Balanceador de carga:

Contenedores para la tarea 472e14b09bfe4770b8691533fbea9063

Nombre ...	ID de tie...	URI de i...	Resum...	Estado	Estado	CPU	Límite i...
aplicacion	472e14...	484992678...	sha256:2d...	Running	Desconocido	0	- / -

- Así, ya podemos revisar nuestra aplicación desplegada en la nube, dispuesta a realizar cualquier nuevo cambio.

Amazon ECS | EC2 Management Console | Bases de datos - Consola | FootPassion | aplicación-footpassion/c | + | Actualizar

FOOTPASSION

Noticias | Estadísticas | Tienda | Restringido

CLASIFICACIÓN

Posición	Nombre	P.Ganados	P.Perdidos	P.Empatados	Puntos
1	Real Madrid	1	0	0	3
2	Atlético de Madrid	1	0	0	3
3	Villarreal Club de Fútbol	1	0	0	3
4	Club Atlético Osasuna	1	0	0	3
5	Valencia CF	1	0	0	3
6	Real ...	1	0	0	3

PLAN DE CONTROL DE MODIFICACIONES Y MANTENIMIENTO

Ya se ha desplegado toda la infraestructura de producción, desarrollo y control, lo que sigue ahora es una buena organización interna de los desarrolladores para trabajar conforme a una metodología compacta que solo permita modificaciones en la rama principal en caso de que las modificaciones hayan sido comprobadas.

A partir de este momento, cada vez que se realicen modificaciones en la rama principal del código, se activará el pipeline de CI/CD y se desplegarán una serie de acciones ya explicadas para terminar con el fin último, el despliegue automatizado y controlado de modificaciones del código en producción sin tocar la infraestructura utilizada en la nube. Aquí tenemos ejemplos de cambios realizados en local directamente a la rama principal:

- En primer lugar clonamos el repositorio en una carpeta local, que ya tendrá git inicializado con el origin.

```
ale@aleubu:~/Escritorio/trabaj$ git clone https://github.com/elgranalex/aplicacion-footpassion.git github
Clonando en 'github'...
Username for 'https://github.com': elgranalex
Password for 'https://elgranalex@github.com':
remote: Enumerating objects: 289, done.
remote: Counting objects: 100% (132/132), done.
remote: Compressing objects: 100% (98/98), done.
remote: Total 289 (delta 57), reused 57 (delta 15), pack-reused 157
Recibiendo objetos: 100% (289/289), 470.62 KiB | 11.00 KiB/s, listo.
Resolviendo deltas: 100% (136/136), listo.
```

- Ahora simplemente se añaden los elementos que queremos cambiar, se hace commit y se realiza el push el remoto.

```
ale@aleubu:~/Escritorio/trabaj/github$ git add .
ale@aleubu:~/Escritorio/trabaj/github$ git commit -m "cambio"
[main ff9f052] cambio
 3 files changed, 5 insertions(+), 3 deletions(-)
ale@aleubu:~/Escritorio/trabaj/github$ git push origin main
Username for 'https://github.com': elgranalex
Password for 'https://elgranalex@github.com':
Enumerando objetos: 11, listo.
Contando objetos: 100% (11/11), listo.
Comprimiendo objetos: 100% (6/6), listo.
Escribiendo objetos: 100% (6/6), 605 bytes | 605.00 KiB/s, listo.
Total 6 (delta 4), reusados 0 (delta 0), pack-reusados 0
remote: Resolving deltas: 100% (4/4), completed with 4 local objects.
To https://github.com/elgranalex/aplicacion-footpassion.git
 5995e35..ff9f052 main -> main
ale@aleubu:~/Escritorio/trabaj/github$ 
```

Los desarrolladores realizarán lo mismo pero cada uno trabajando en su rama y creando una pull request para cambios sobre producción.

WEBGRAFÍA

https://docs.aws.amazon.com/es_es/AmazonECS/latest/developerguide/application_architecture.html

<https://www.qualoom.es/blog/arquitectura-wordpress-aws/>

<https://es.linkedin.com/pulse/infraestructura-como-c%C3%B3digo-devops-jos%C3%A9-mar%C3%A1-vigueras-bernal>

<https://galvarado.com.mx/post/tutorial-infraestructura-como-c%C3%B3digo-con-terraform/>

<https://pilotcoresystems.com/insights/aws-fargate-vs-ecs-which-one-is-best-for-my-workload/>

<https://www.youtube.com/watch?v=Fq5Va7Hb4Cg>

<https://repost.aws/es/knowledge-center/efs-mount-on-ecs-container-or-task>

<https://www.youtube.com/watch?v=ahPaTmSRUa4>

<https://dev.to/aws-builders/aws-importexport-infrastructure-as-code-cloudformation-terraform-12jn>

<https://aws.amazon.com/es/rds/features/multi-az/>

<https://aws.amazon.com/es/rds/mysql/>

<https://www.youtube.com/watch?v=LeGMPfi43rM>

<https://repost.aws/es/knowledge-center/rds-connect-ec2-bastion-host>

<https://linux.how2shout.com/3-ways-to-install-mysql-workbench-on-ubuntu-22-04-lts-linux/>

https://docs.aws.amazon.com/es_es/efs/latest/ug/wt1-create-efs-resources.html

<https://repost.aws/knowledge-center/efs-mount-on-ecs-container-or-task>

<https://www.youtube.com/watch?v=tcV8utwxXro>

<https://www.youtube.com/watch?v=Y06WP9MbYrY&t=2s>

https://docs.aws.amazon.com/es_es/AmazonECS/latest/developerguide/tutorial-efs-volumes.html

<https://www.youtube.com/watch?v=JANKxp728dg>

<https://www.youtube.com/watch?v=mdFOohfheJc>

<https://www.youtube.com/watch?v=sIhm4YOMK6Q>

<https://registry.terraform.io/providers/hashicorp/aws/latest/docs>

<https://online.visual-paradigm.com/es/diagrams/features/aws-architecture-diagram-tool/>

<https://www.ibm.com/es-es/cloud/learn/containers#:~:text=Los%20contenedores%20son%20unidades%20ejecutables.tradicional%20o%20en%20el%20cloud.>

<https://www.youtube.com/watch?v=gjRoNFopFig>

<https://openwebinars.net/blog/contenedores-de-software-que-son-y-que-ventajas-ofrecen/>

https://keepcoding.io/blog/que-es-dockerfile/#Caracteristicas_de_Dockerfile

<https://docs.docker.com/engine/reference/builder/>

<https://www.javiergarzas.com/2015/07/que-es-docker-sencillo.html>

<https://www.hpe.com/es/es/what-is/containers.html#:~:text=Los%20contenedores%20son%20tecnolog%C3%ADAsque,sistema%20operativo%20en%20cualquier%20contexto.>

<https://guiadev.com/top-5-alternativas-a-docker/>

<https://www.netapp.com/blog/containers-vs-vms/>

<https://www.atlassian.com/es/microservices/cloud-computing/containers-vs-vms>

<https://www.redhat.com/es/topics/containers/what-is-docker>

<https://www.ibm.com/es-es/topics/docker>

<https://luisiblogdeinformatica.com/crear-dockerfile/>

<https://www.hostinger.es/tutoriales/como-crear-contenedor-docker>

<https://openwebinars.net/blog/que-es-dockerfile/>

<https://keepcoding.io/blog/como-usar-el-comando-docker-run/>

<https://pandorafms.com/blog/es/docker-run/>

https://dev.to/prox_sea/docker-curso-practico-con-ejemplos-2fkd

<https://github.com/harbur/docker-workshop/tree/master/doc/02-docker-compose>

<https://colaboratorio.net/davidochobits/sysadmin/2018/despliegue-de-aplicaciones-con-docker-compose/>

<https://coffeebytes.dev/docker-compose-tutorial-con-comandos-en-gnu-linux/>

<https://www.returngis.net/2020/12/como-funcionan-las-redes-en-docker/>

<https://azure.microsoft.com/es-es/solutions/kubernetes-on-azure/get-started/>

<https://www.campusmvp.es/recursos/post/docker-vs-kubernetes-en-que-se-diferencian.aspx>

<https://cloud.google.com/kubernetes-engine/docs/tutorials/hello-app?hl=es-419>

<https://www.xataka.com/otros/docker-a-kubernetes-entendiendo-que-contenedores-que-mayores-revoluciones-industria-desarrollo>

<https://azure.microsoft.com/es-es/solutions/kubernetes-on-azure/deployment-strategy/>

<https://www.youtube.com/watch?v=DCoBcpOA7W4>

<https://keepcoding.io/blog/que-es-kubelet#:~:text=La%20herramienta%20de%20Kubelet%20se,un%20recurso%20de%20pod%20determinado.>

<https://kubernetes.io/docs/concepts/>

<https://kinsta.com/es/blog/instalar-docker-ubuntu/#instalar-docker-engine-en-ubuntu>

<https://bluu.ee/docs/snippets/docker-desktop-para-ubuntu-22-04/>

<https://docs.docker.com/desktop/install/ubuntu/>

<https://www.docker.com/blog/how-kubernetes-works-under-the-hood-with-docker-desktop/>

<https://geekflare.com/es/kubectl-examples/>

<https://kubernetes.io/docs/reference/kubectl/cheatsheet/>

<https://kubernetes.io/es/docs/concepts/overview/working-with-objects/kubernetes-objects/>

https://www.juniper.net/documentation/en_US/day-one-books/topics/concept/yaml-file-for-kubernetes.html

<https://www.ramoncarrasco.es/es/content/es/kb/176/fichero-de-configuracion-yaml-de-kubernetes>

<https://geekflare.com/es/kubernetes-monitoring-tools/>

<https://www.youtube.com/watch?v=q-ZicDSb3Cc>

<https://devopscube.com/setup-prometheus-monitoring-on-kubernetes/>

https://www.youtube.com/watch?v=yvUQMdgBz_c

<https://geekflare.com/es/terraform-for-beginners/>

<https://developer.hashicorp.com/terraform/docs>

<https://terraform-infraestructura.readthedocs.io/es/latest/comandos/>

<https://juancadh.medium.com/conectar-carpetas-local-con-repositorio-de-github-8d983798998e>

<https://wdflat.com/product/whisper/>



Rialex-Sportmedia © 2023 by Alejandro Iglesias Carpintero
is licensed under Attribution-NonCommercial-ShareAlike 4.0
International. To view a copy of this license, visit
<http://creativecommons.org/licenses/by-nc-sa/4.0/>