

Tarea3 - Docker

Adrián García de la Cera

Enlace repositorio GitHub:

<https://github.com/fuejomusta/dockerAdri-n.git>

Enlace vídeo presentación:

<https://www.loom.com/share/31ab362060de4f6cbaf8e188eb7d6d5?sid=1b899e87-8821-4900-b2a8-a2c2dc6f8b3b>

Tarea3 - Docker

Pasos previos

Ejercicio 1 - Contenedores en red y Docker Desktop

Enunciado

1.Creación de red bridge redej1

2. Crea un contenedor con una imagen de [mariadb](#)

Definir credenciales

3. Crear un contenedor con [Adminer](#)

4. Conectar contenedores a redej1

Configurar redes

Conexión a interfaz grafica

Ejecución del script SQL

5.Instalación y comprobaciones con [Disk usage](#)

Borrado de volúmenes, contenedores y la red

Ejercicio 2 - Docker Compose

Enunciado

Desarrollo

Creación archivo

Inicio de servicio

Pruebas con filebrowser

Finalización del servicio

Ejercicio 3 - imagen con Dockerfile - Aplicación web

Enunciado

Desarrollo

Creación de archivos

Creación de la imagen

Comprobaciones

Subida de imagenes

Borrado y descarga de imagen

Pasos previos

Lo primero que vamos a realizar es crear el repositorio GitHub, con la rama principal llamada "main".

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Required fields are marked with an asterisk (*).

Owner * Repository name *

 fuejomusta / dockerAdrián

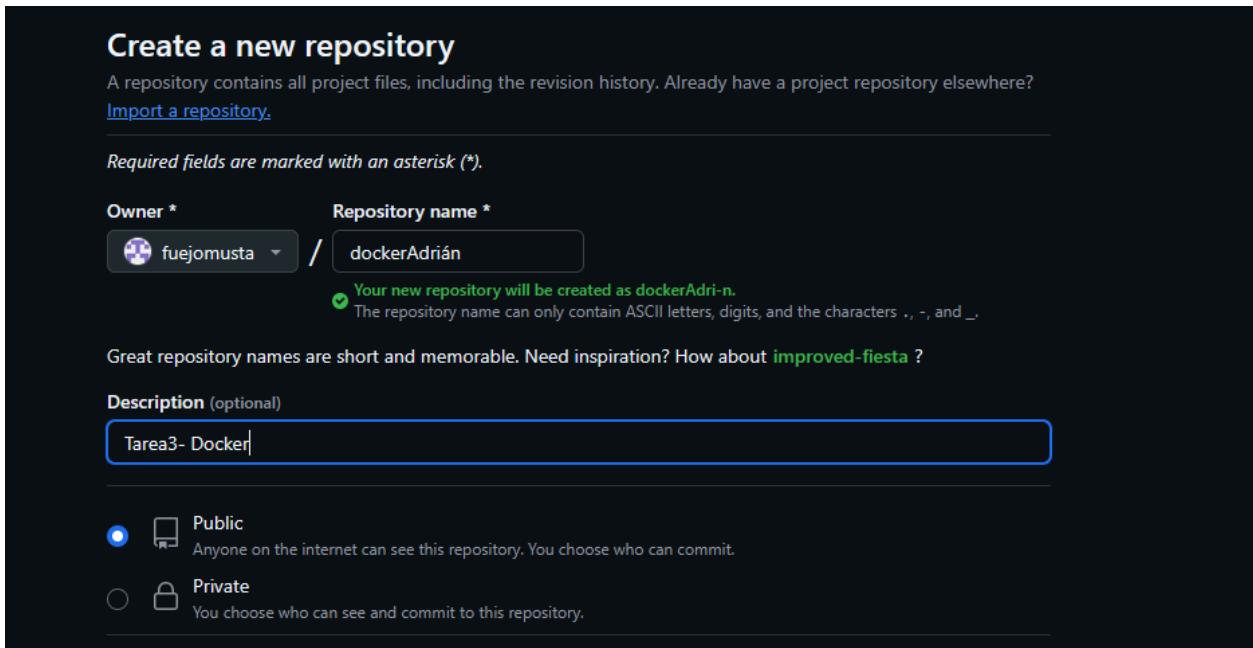
Your new repository will be created as dockerAdri-n.
The repository name can only contain ASCII letters, digits, and the characters ., -, and _.

Great repository names are short and memorable. Need inspiration? How about [improved-fiesta](#) ?

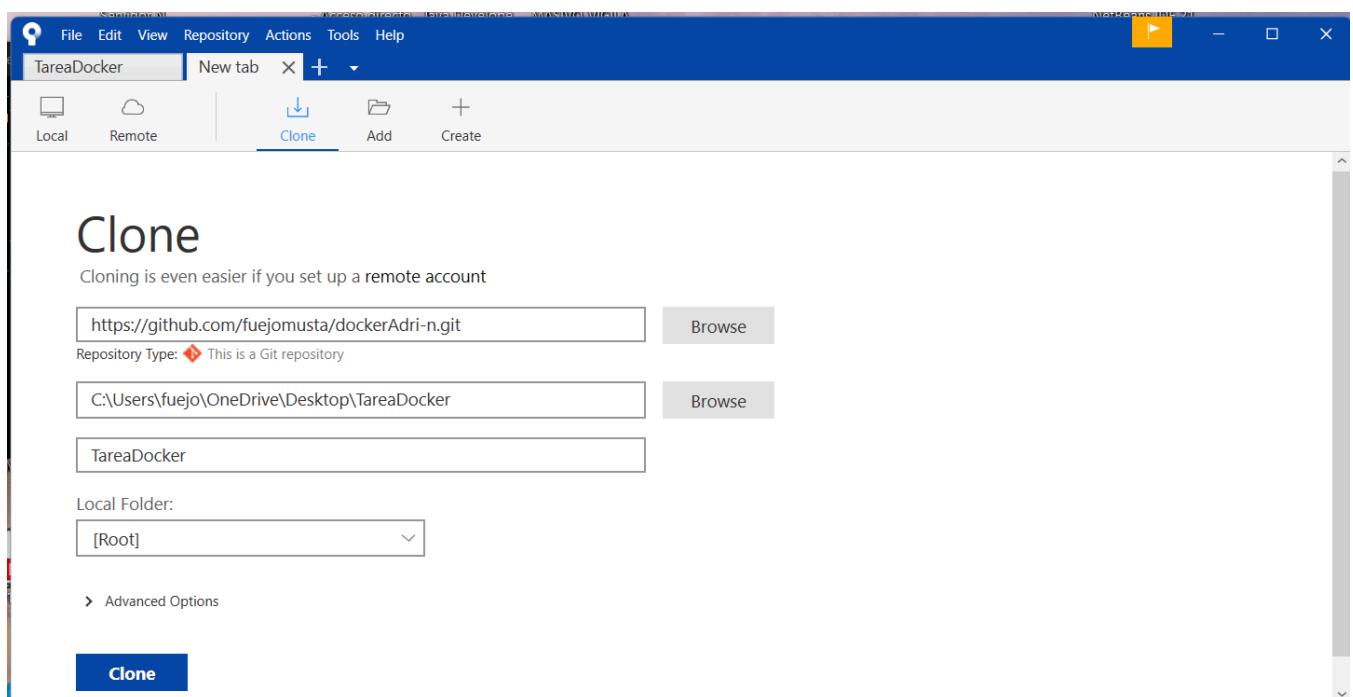
Description (optional)
Tarea3- Docker

Public
Anyone on the internet can see this repository. You choose who can commit.

Private
You choose who can see and commit to this repository.



Una vez creado el repositorio, lo clonamos en soucetree para poder trabajar con él en local.



File Edit View Repository Actions Tools Help

TareaDocker New tab X +

Local Remote Clone Add Create

Clone

Cloning is even easier if you set up a remote account

Repository Type:  This is a Git repository

https://github.com/fuejomusta/dockerAdri-n.git Browse

C:\Users\fuejo\OneDrive\Desktop\TareaDocker Browse

TareaDocker

Local Folder:

[Root]

Advanced Options

Clone

Con el repositorio ya creada y clonado en local, procedemos a crear las ramas, una por ejercicio:

```
git branch ejercicio-1
git branch ejercicio-2
git branch ejercicio-3
```

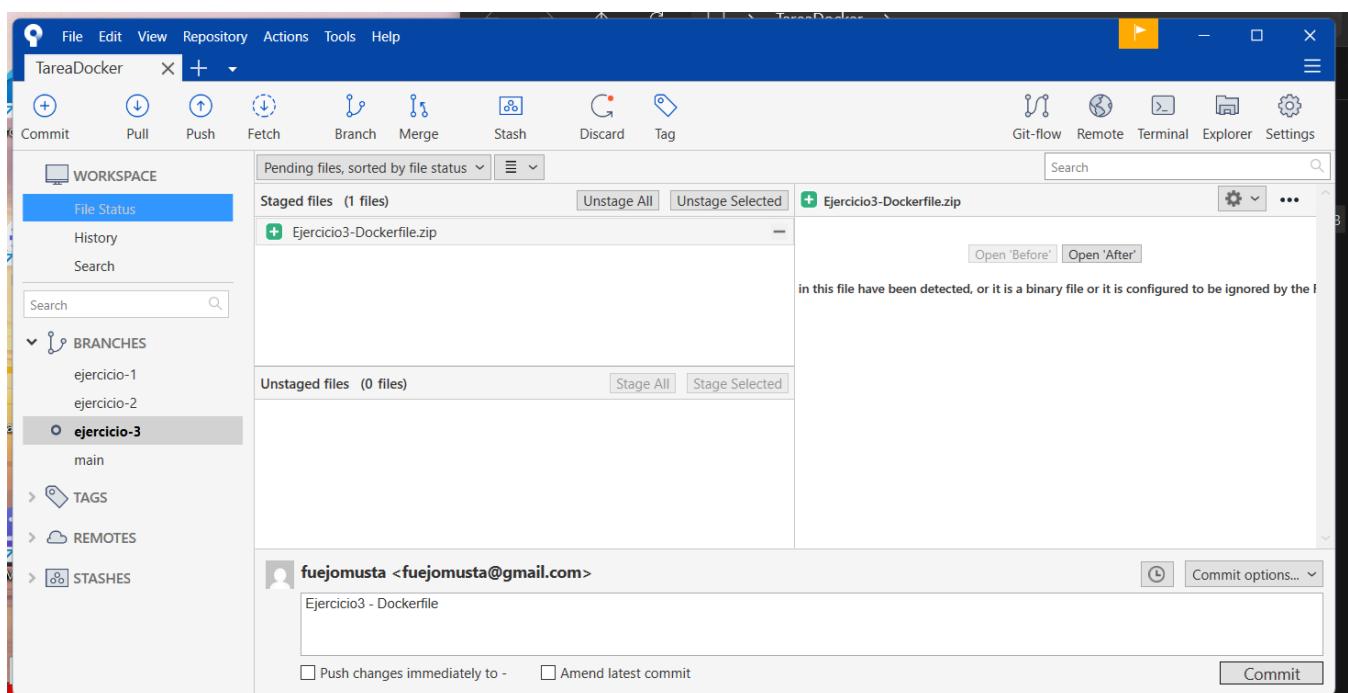
```
fuejo@LAPTOP-67TJQE96 MINGW64 ~/OneDrive/Desktop/TareaDocker (main)
$ git branch -a
  ejercicio-1
* main

fuejo@LAPTOP-67TJQE96 MINGW64 ~/OneDrive/Desktop/TareaDocker (main)
$ git branch ejercicio-2
* main

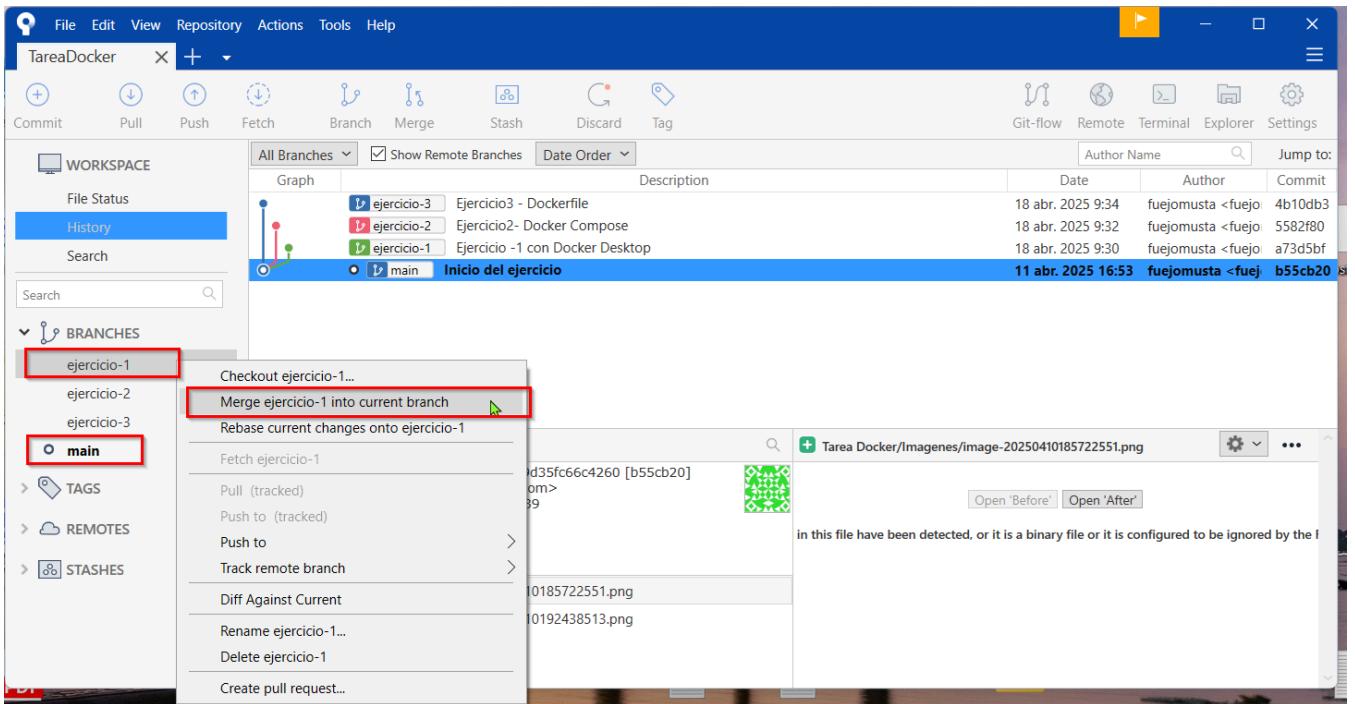
fuejo@LAPTOP-67TJQE96 MINGW64 ~/OneDrive/Desktop/TareaDocker (main)
$ git branch ejercicio-3
* main

fuejo@LAPTOP-67TJQE96 MINGW64 ~/OneDrive/Desktop/TareaDocker (main)
$ git branch -a
  ejercicio-1
  ejercicio-2
  ejercicio-3
* main
```

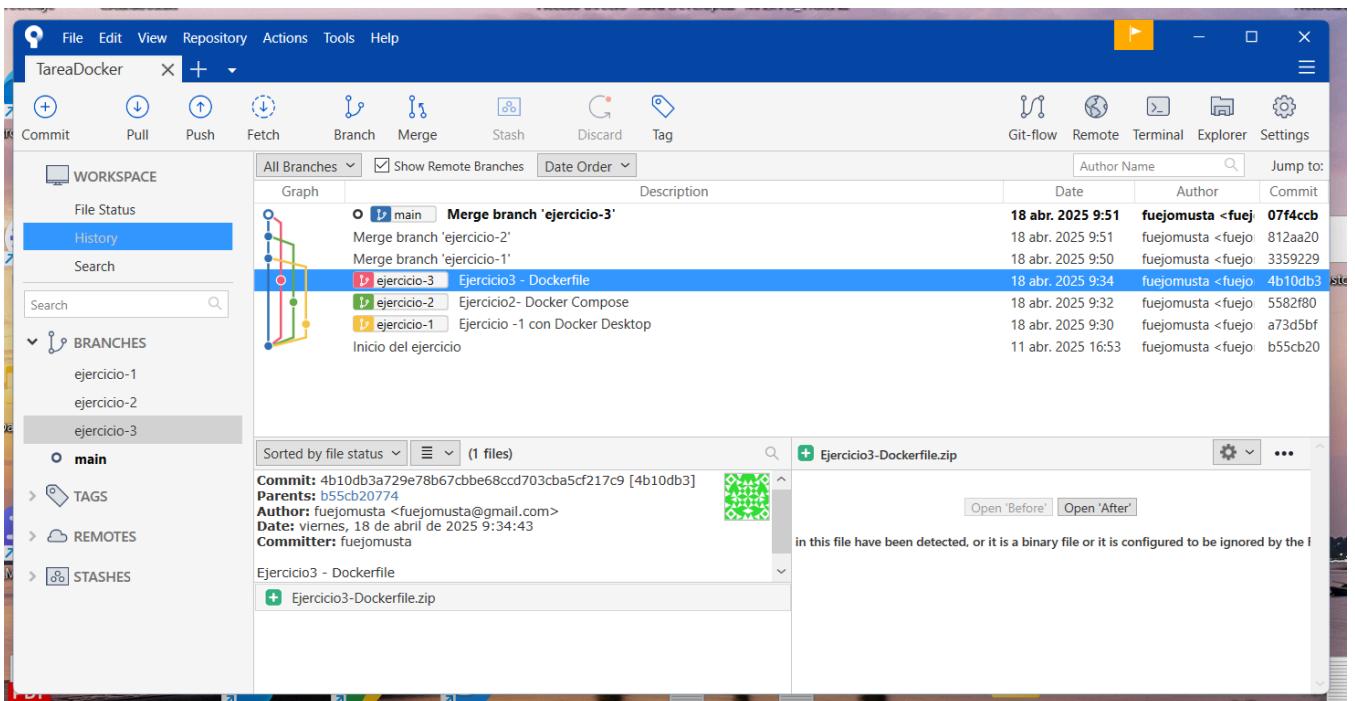
Con las ramas ya creadas, procedemos a guardar en cada una de ellas el ejercicio correspondiente, realizando el guardado y el commit.



Para fusionar las ramas desde `sourcetree` seleccionamos la rama principal `main`, hacemos clic con botón derecho sobre la rama a fusionar y seleccionamos `Merge ejercicio-*`.



Como vemos en la siguiente imagen de `Sourcetree` todas las ramas se han fusionado con la rama principal `main`



Ejercicio 1 - Contenedores en red y Docker Desktop

Adrián García de la Cera

Tarea3 - Docker

Pasos previos

Ejercicio 1 - Contenedores en red y Docker Desktop

Enunciado

- 1.Creación de red bridge `redej1`
2. Crea un contenedor con una imagen de `mariadb`
 - Definir credenciales
3. Crear un contenedor con `Adminer`
4. Conectar contenedores a `redej1`
 - Configurar redes
 - Conexión a interfaz gráfica
 - Ejecución del script SQL
- 5.Instalación y comprobaciones con `Disk usage`
 - Borrado de volúmenes, contenedores y la red

Ejercicio 2 - Docker Compose

- Enunciado
- Desarrollo
- Creación archivo
 - Inicio de servicio
 - Pruebas con filebrowser
 - Finalización del servicio

Ejercicio 3 - imagen con Dockerfile - Aplicación web

- Enunciado
- Desarrollo
- Creación de archivos
 - Creación de la imagen
 - Comprobaciones
 - Subida de imágenes
- Borrado y descarga de imagen

Enunciado

Este ejercicio se resolverá con Docker Desktop, si se soluciona con comandos la nota será 0

Si necesitas usar comandos, usa la Terminal integrada en Desktop

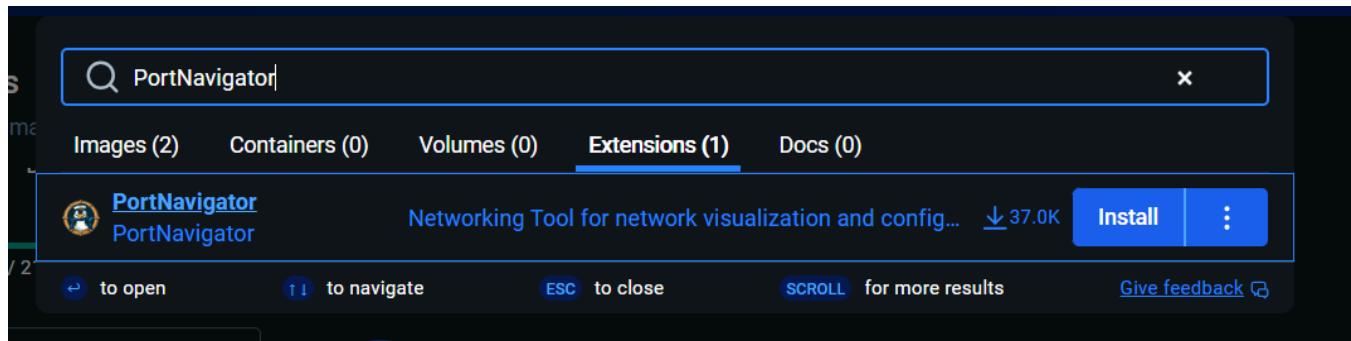
SUGERENCIA: Utiliza la extensión PortNavigator para gestionar las redes en Docker Desktop

- 1.Crea una red bridge `redej1`
- 2.Crea un contenedor con una imagen de `mariadb` que estará en la red `redej1`.Este contenedor se ejecutará en segundo plano, y será accesible a través del puerto 3306.
 - Definir una contraseña para el usuario `root`, y un usuario con tu nombre de pila y con contraseña. La BD por defecto será `DAW`.
 - Genera un script SQL que cree una tabla `módulos` con algunos registros con los nombres de los módulos que estás estudiando.
- 3.Crear un contenedor con `Adminer` o con `phpMyAdmin` que se pueda conectar al contenedor de la BD.
- 4.Desde la interfaz gráfica elegida, conéctate a la BD con tu usuario personal, ejecuta el script con los datos de los módulos y muestra la BD y la tabla creados.
- 5.Elige entre estas dos opciones:

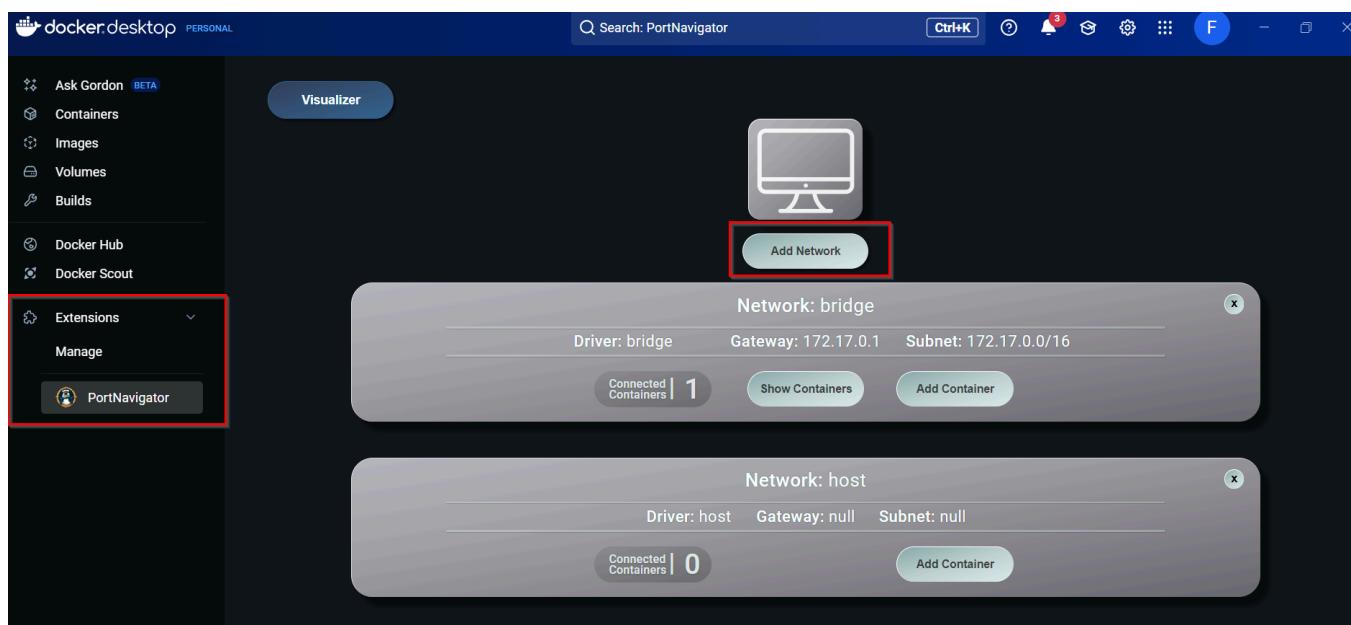
- Instala la extensión `Disk usage`, muestra el espacio ocupado, borra algo...
- Instala la extensión `Resource usage` y muestra su salida cuando estén activos los contenedores.

1.Creación de red bridge redej1

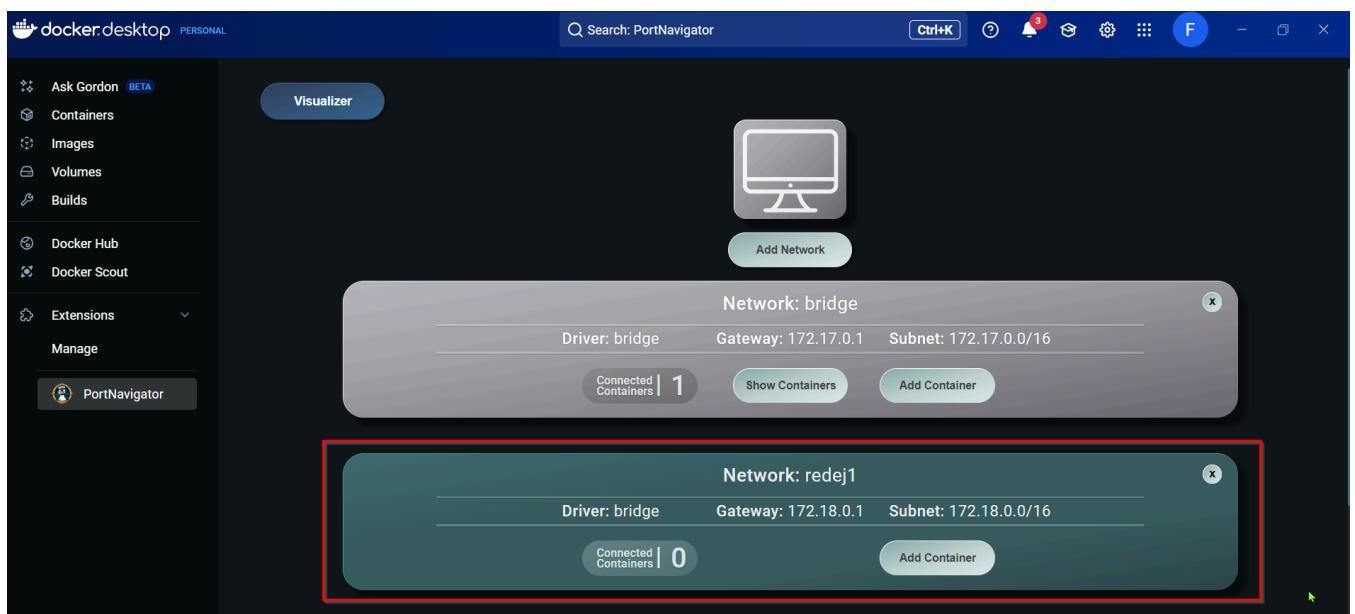
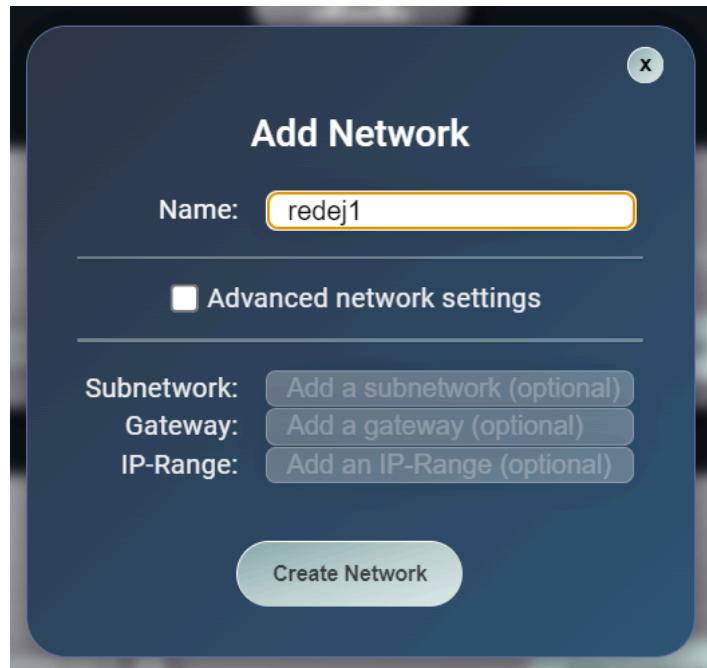
Para crear la red usaremos como aconseja el título la extensión `PortNavigator`, para ello lo primero que debemos realizar es instalarla, en la parte central superior buscamos la extensión y pulsamos sobre el botón `Install`.



Una vez instalada, nos aparecerá en la parte izquierda, en la sección de extensiones, pulsamos sobre ella y en el botón central "Add Network"

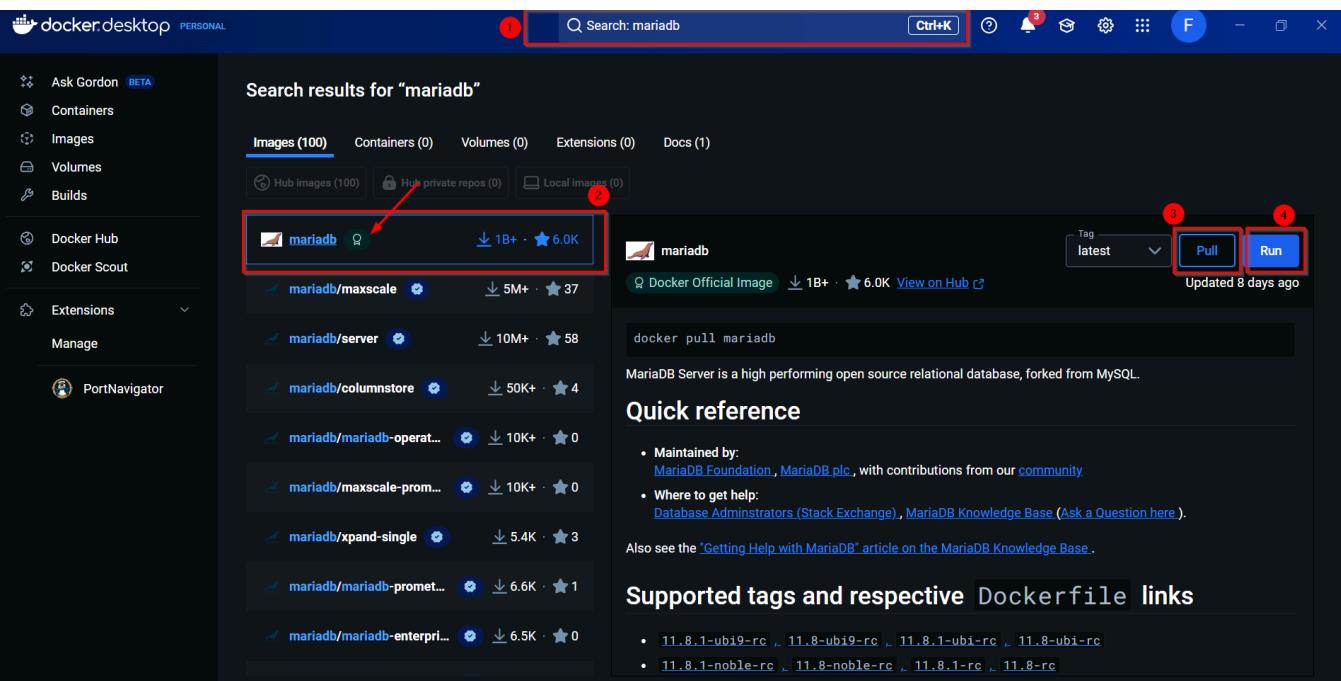


La creamos con nombre `redej1` como solicita el enunciado:

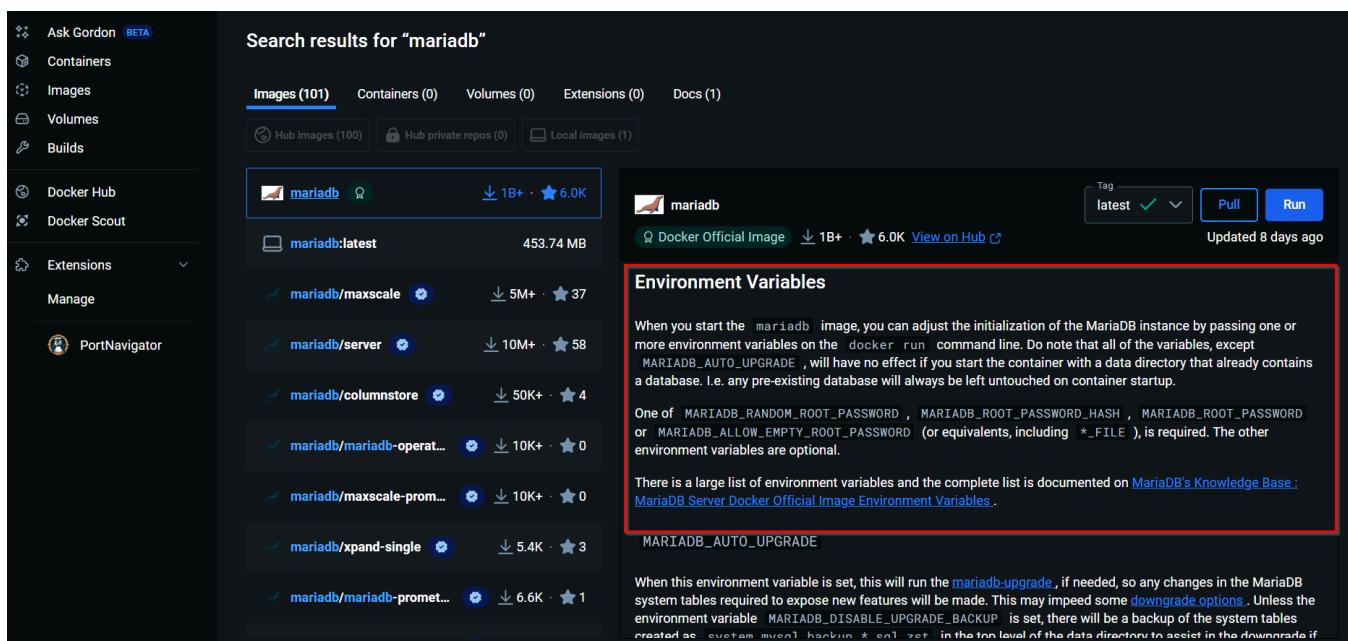


2. Crea un contenedor con una imagen de mariadb

Para crear el contenedor, el primer paso es usar el buscador de Docker Desktop para buscar la imagen oficial de `mariadb` (Descargaremos la imagen oficial, que se indica con un símbolo de una medalla junto al nombre), la descargamos con el botón `Pull` y creamos el contenedor con el botón `Run`.



Para crear el contenedor con los datos requeridos en el enunciado, accedemos a la documentación de la imagen la cual podemos revisar o desde la web de Docker Hub o debajo de los propios botones de **Pull** y **Run**.

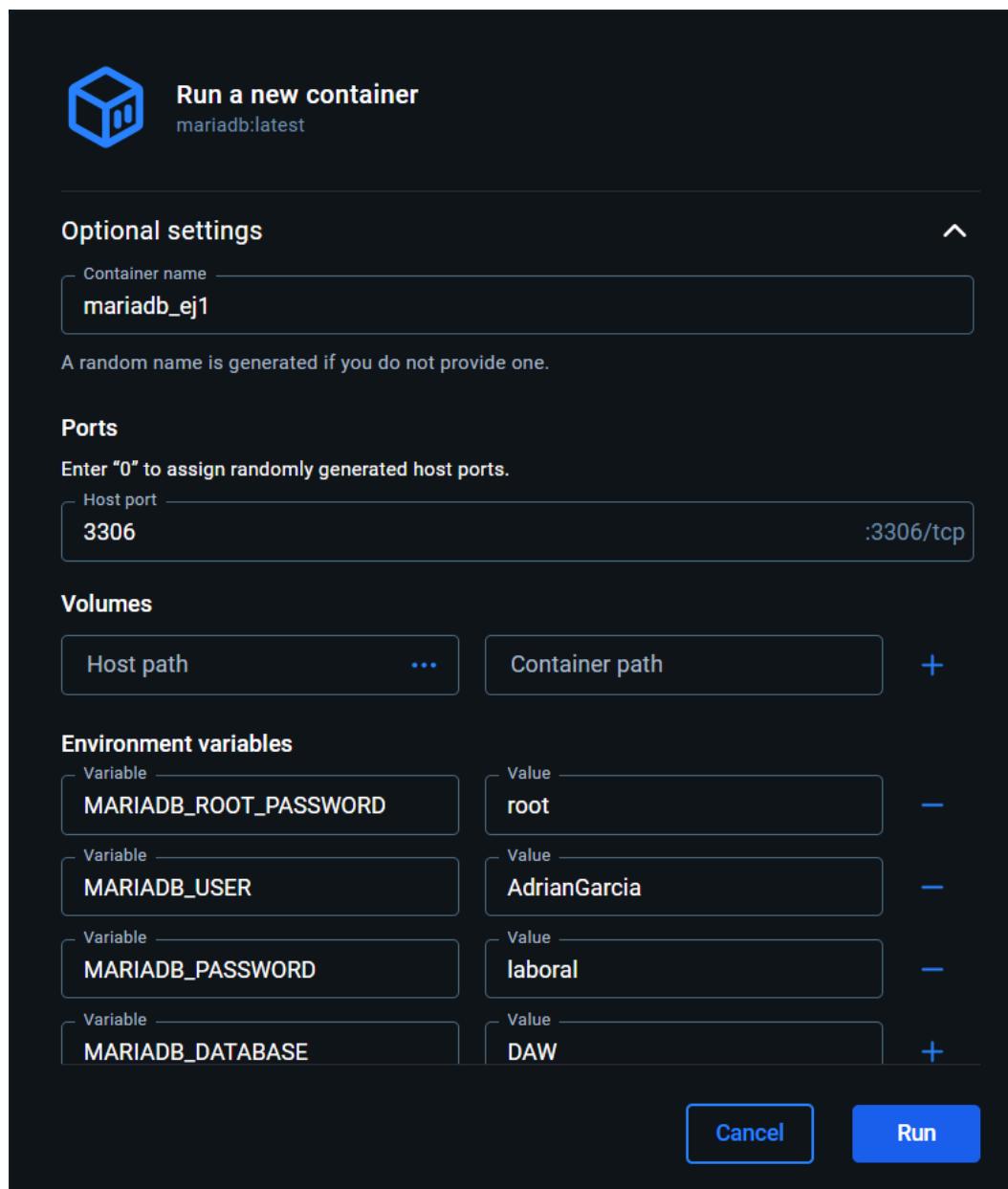


Definir credenciales

Al pulsar el botón **Run** desplegamos **optional settings** y procedemos a definir las opciones:

- Definir la contraseña para el usuario root: Revisando la documentación, observamos que la variable se llama `MARIADB_ROOT_PASSWORD`, le estableceremos de contraseña `root`.
- Definir usuario con mi nombre y contraseña: En la documentación encontramos un enlace ([MariaDB's Knowledge Base : MariaDB Server Docker Official Image Environment Variables](#)) a la web oficial de `mariadb` en esta observamos el nombre de las variables para asignar valores que deseamos.
 - Nombre de usuario: La variable de entorno según documentación es `MARIADB_USER`, a la cual le daremos como valor mi nombre y apellido, `AdrianGarcia`.

- Contraseña: La variable de entorno se llama `MARIADB_PASSWORD`, a la que le daremos como valor `laboral`.
- Nombre de la base de datos `DAW`, para ello le asignaremos a la variable de entorno `MARIADB_DATABASE` el valor `DAW`.



Una vez finalizada la configuración, pulsamos sobre `Run` y comienza a descargarse y configurarse.

The screenshot shows the Docker Desktop interface. On the left, there's a sidebar with various options like Ask Gordon, Containers, Images, Volumes, Builds, Docker Hub, Docker Scout, Extensions, Manage, and PortNavigator. The 'Containers' option is selected. In the main area, it shows a list of containers: 'mariadb_ej1' (running, status: 'Running (2 seconds ago)'), 'bfad0f356fde' (running), and 'mariadb:latest' (running). Below the list, there are tabs for Logs, Inspect, Bind mounts, Exec, Files, and Stats. The Logs tab is active, displaying the terminal output of the MariaDB container. The output shows the startup logs of MariaDB 11.7.2, including the MySQL configuration file path and the log sequence number.

```

2025-04-17 17:29:09 0 [Note] Starting MariaDB 11.7.2-MariaDB-ubuntu2404 source revision 80867a69feaeb5df3abb1bfa7d4e713ccbf027 server_id 1
K/cBYSpI Hv7Vnkhv3HQbeVPtg: as process 1
2025-04-17 17:29:09 0 [Note] InnoDB: Compressed tables use zlib 1.3
2025-04-17 17:29:09 0 [Note] InnoDB: Number of transaction pools: 1
2025-04-17 17:29:09 0 [Note] InnoDB: Using crc32 + pclmulqdq instructions
2025-04-17 17:29:09 0 [Note] mariadb: 0. TMPFILE is not supported on /tmp (disabling future attempts)
2025-04-17 17:29:09 0 [Note] InnoDB: Using luring
2025-04-17 17:29:09 0 [Note] InnoDB: Initializing buffer pool, total size = 128.000MiB, chunk size = 2.000MiB
2025-04-17 17:29:09 0 [Note] InnoDB: Completed initialization of buffer pool
2025-04-17 17:29:09 0 [Note] InnoDB: File system buffers for log disabled (block size=4096 bytes)
2025-04-17 17:29:09 0 [Note] InnoDB: End of log at LSN=47629
2025-04-17 17:29:09 0 [Note] InnoDB: Opened 3 undo tablespaces
2025-04-17 17:29:09 0 [Note] InnoDB: 128 rollback segments in 3 undo tablespaces are active.
2025-04-17 17:29:09 0 [Note] InnoDB: Setting file '/ibtmp1' size to 12.000MiB. Physically writing the file full; Please wait ...
2025-04-17 17:29:09 0 [Note] InnoDB: File '/ibtmp1' size is now 12.000MiB.
2025-04-17 17:29:09 0 [Note] InnoDB: log sequence number 47629; transaction id 14
2025-04-17 17:29:09 0 [Note] InnoDB: Loading buffer pool(s) from /var/lib/mysql/ib_buffer_pool
2025-04-17 17:29:09 0 [Note] Plugin 'FEEDBACK' is disabled.
2025-04-17 17:29:09 0 [Note] Plugin 'wsrep-provider' is disabled.
2025-04-17 17:29:09 0 [Note] InnoDB: Buffer pool(s) load completed at 250417 17:29:09
2025-04-17 17:29:10 0 [Note] Server socket created on IP: '0.0.0.0'.
2025-04-17 17:29:10 0 [Note] Server socket created on IP: '::'.
2025-04-17 17:29:10 0 [Note] mariadb: Event Scheduler: Loaded 0 events
2025-04-17 17:29:10 0 [Note] mariadb: ready for connections.
Version: '11.7.2-MariaDB-ubuntu2404' socket: '/run/mysqld/mysqld.sock' port: 3306 mariadb.org binary distribution

```

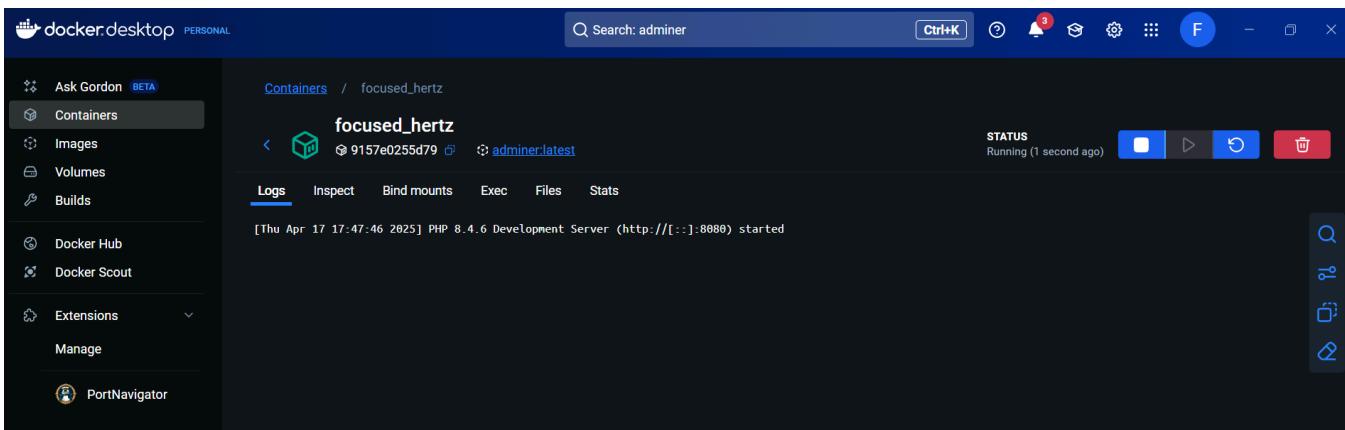
This screenshot shows the Docker Desktop interface with the 'Containers' tab selected. It displays three running containers: 'magical_volhard', 'magical_almeida', and 'mariadb_ej1'. The 'mariadb_ej1' container is highlighted with a red border. The interface includes a search bar, CPU usage metrics (0.04% / 1200%), memory usage (315.44MB / 3.67GB), and a 'Show charts' button. Below the table, there are sections for 'Walkthroughs' and 'View more in the Learning center'.

	Name	Container ID	Image	Port(s)	CPU (%)	Last started	Actions
<input type="checkbox"/>	magical_volhard	ebdfb0281209	docker	8080:80	0%	1 hour ago	D ⋮ W
<input type="checkbox"/>	magical_almeida	93b0d587e871	fuejomusta/ejercicio3:v1	8080:80	0.01%	1 hour ago	D ⋮ W
<input type="checkbox"/>	mariadb_ej1	bfad0f356fde	mariadb:latest	3306:3306	0.03%	3 minutes ago	D ⋮ W

3. Crear un contenedor con Adminer

Al igual que se hizo anteriormente con mariadb en el buscador buscamos `Adminer`, lo descargamos con el botón `Pull` y creamos el contenedor con `Run`.

This screenshot shows the Docker Hub search results for 'adminer'. The search bar at the top contains 'adminer'. Below the search bar, there are tabs for 'Images (50)', 'Containers (0)', 'Volumes (0)', 'Extensions (0)', and 'Docs (0)'. The 'Images (50)' tab is selected. Under the search results, there's a card for the 'adminer' image, which has 100M+ stars and 943 reviews. It includes a 'Pull' button and a 'Run' button. At the bottom of the page, there are links for 'Hub images (50)', 'Hub private repos (0)', and 'Local images (0)'.

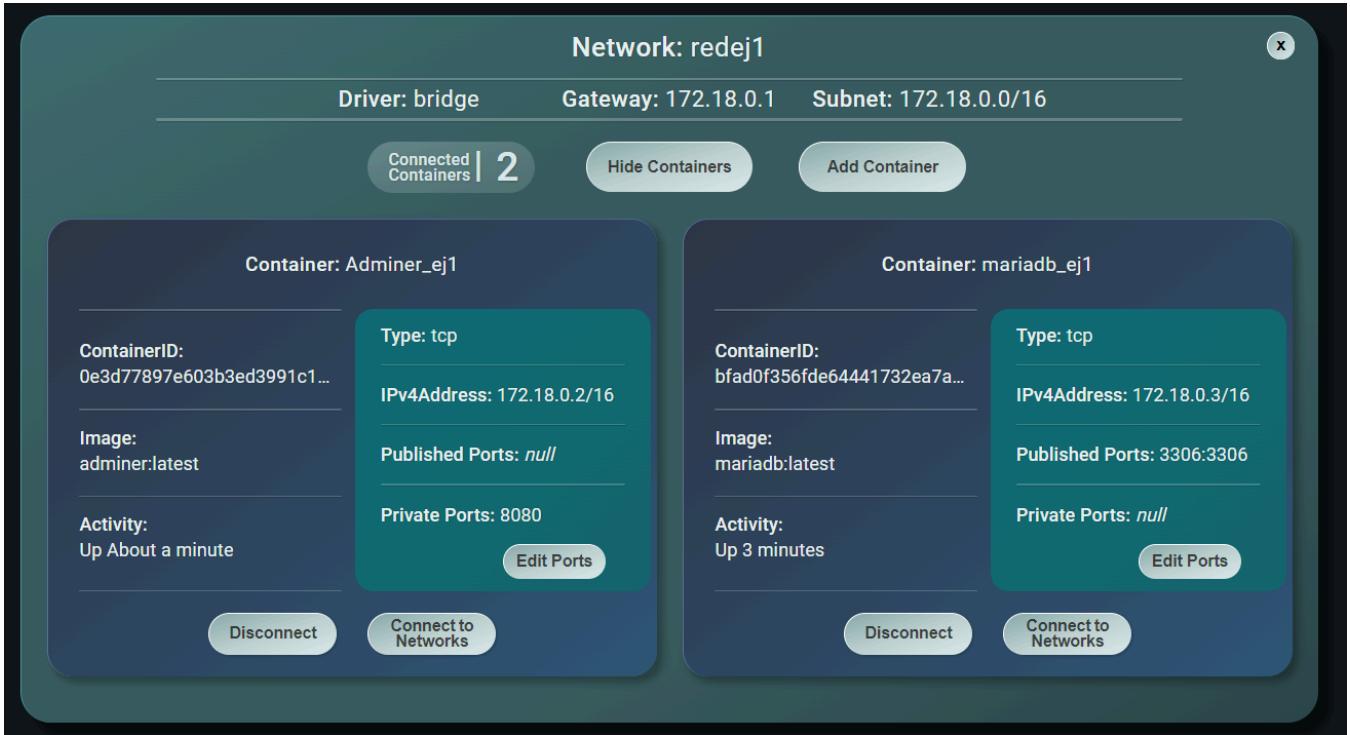


4. Conectar contenedores a redej1

Configurar redes

Como podemos ver en la siguiente imagen, actualmente tenemos ambos contenedores conectados a la red `bridge` por defecto, debemos desconectarlos y conectarlos a la red creada `redej1`





Una vez conectados comprobamos su ip con `ifconfig` y realizamos un ping para comprobar la conexión:

```

/var/www/html $ ifconfig
eth0      Link encap:Ethernet HWaddr A6:7D:07:62:55:D5
          inet addr:172.18.0.2  Bcast:172.18.255.255  Mask:255.255.0.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:15 errors:0 dropped:0 overruns:0 frame:0
          TX packets:3 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:1298 (1.2 kB)  TX bytes:126 (126.0 B)

lo       Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

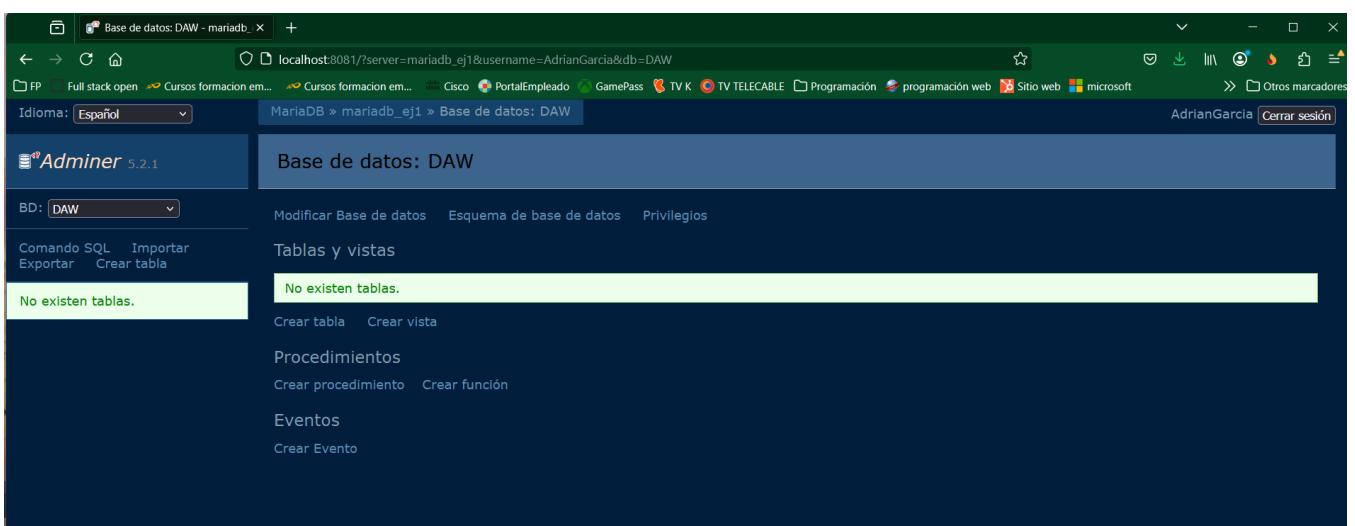
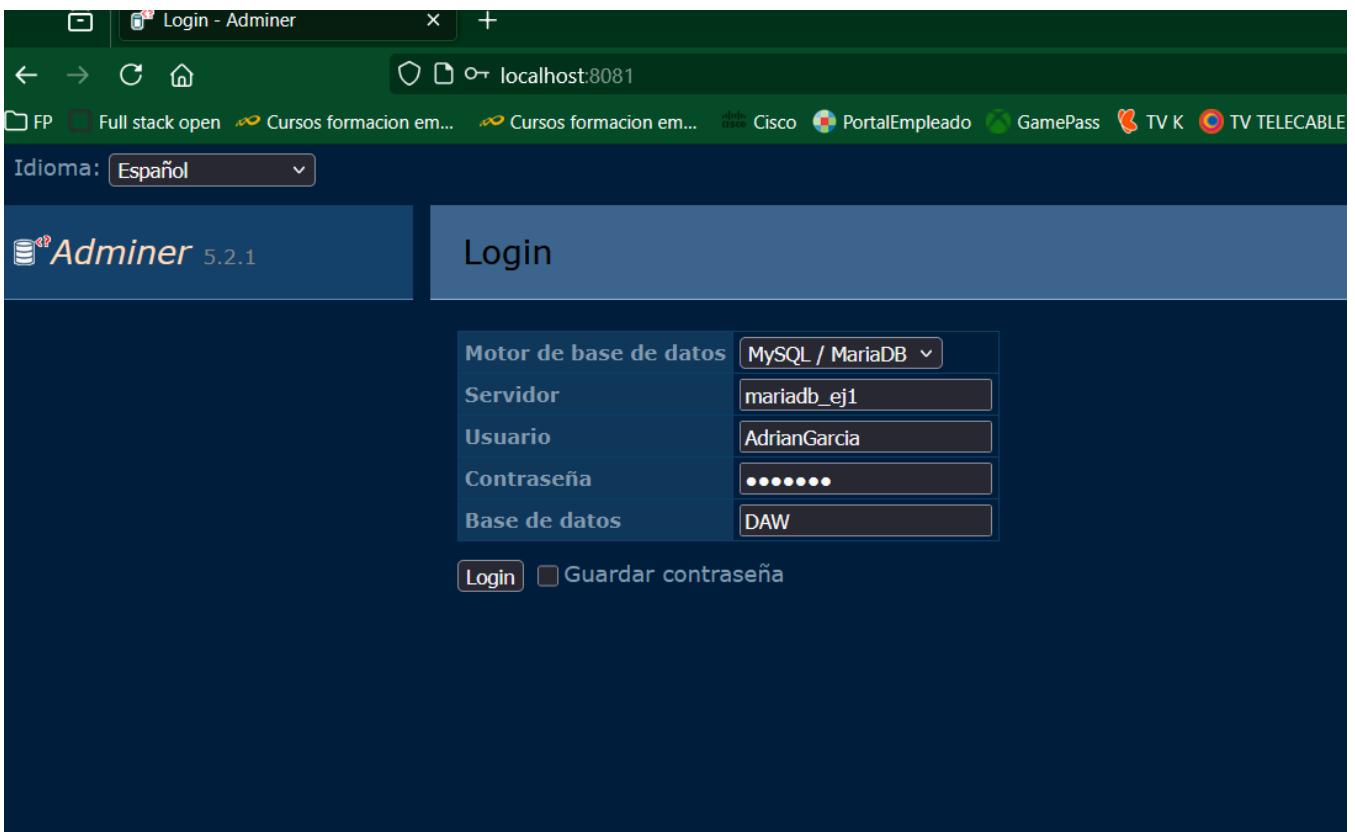
/var/www/html $ ping 172.18.0.2
PING 172.18.0.2 (172.18.0.2): 56 data bytes
64 bytes from 172.18.0.2: seq=0 ttl=42 time=0.156 ms
64 bytes from 172.18.0.2: seq=1 ttl=42 time=0.138 ms
64 bytes from 172.18.0.2: seq=2 ttl=42 time=0.108 ms
64 bytes from 172.18.0.2: seq=3 ttl=42 time=0.113 ms
64 bytes from 172.18.0.2: seq=4 ttl=42 time=0.164 ms
64 bytes from 172.18.0.2: seq=5 ttl=42 time=0.107 ms
64 bytes from 172.18.0.2: seq=6 ttl=42 time=0.108 ms
^C

```

Conexión a interfaz grafica

Para conectarnos a través del navegador indicamos:

`http://localhost:8081/`



Ejecución del script SQL

Una vez dentro seleccionaremos Comando SQL y ejecutaremos el script:

```
CREATE TABLE modulos (
    id INT AUTO_INCREMENT PRIMARY KEY,
    nombre VARCHAR(75) NOT NULL
);

INSERT INTO modulos (nombre) VALUES
('cliente'),
('interfaces'),
('despliegue'),
('ingles'),
('programación');
```

The screenshot shows the Adminer web interface for MySQL. The left sidebar shows the database 'DAW' selected. The main area is titled 'Comando SQL' and contains the following SQL code:

```

CREATE TABLE modulos (
    id INT AUTO_INCREMENT PRIMARY KEY,
    nombre VARCHAR(255) NOT NULL
);

INSERT INTO modulos (nombre) VALUES
('cliente'),
('interfaces'),
('despliegue'),
('ingles'),
('programación');

CREATE TABLE modulos (
    id INT AUTO_INCREMENT PRIMARY KEY,
    nombre VARCHAR(255) NOT NULL
);

INSERT INTO modulos (nombre) VALUES
('cliente'),
('interfaces'),
('despliegue'),
('ingles'),
('programación');

```

Below the code, two green status bars indicate the results: 'Consulta ejecutada, 0 registros afectados.' and 'Consulta ejecutada, 5 registros afectados.'

Con el script ya ejecutado, podemos comprobar en registros que realmente se han añadido:

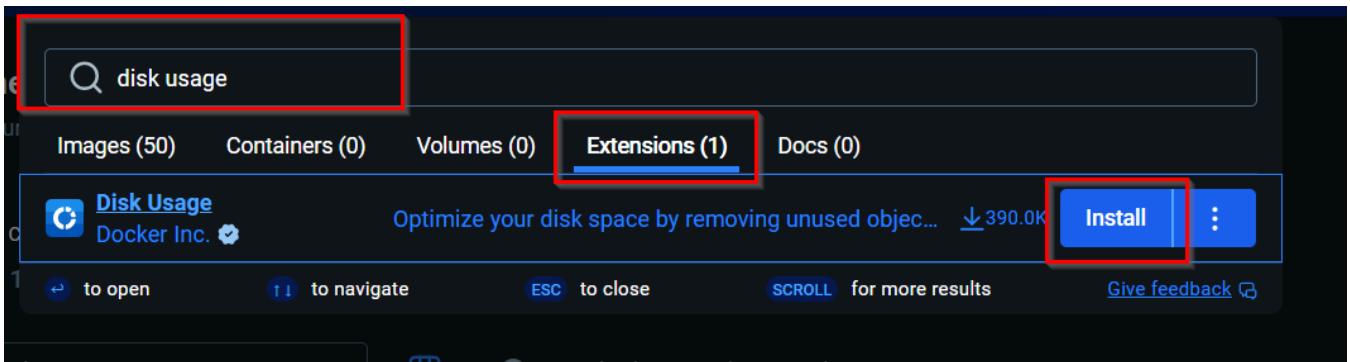
The screenshot shows the Adminer web interface displaying the results of the executed SQL query. The left sidebar shows the database 'DAW' selected. The main area is titled 'Mostrar: modulos' and contains the following table:

	id	nombre
Modificar	1	cliente
Modificar	2	interfaces
Modificar	3	despliegue
Modificar	4	ingles
Modificar	5	programación

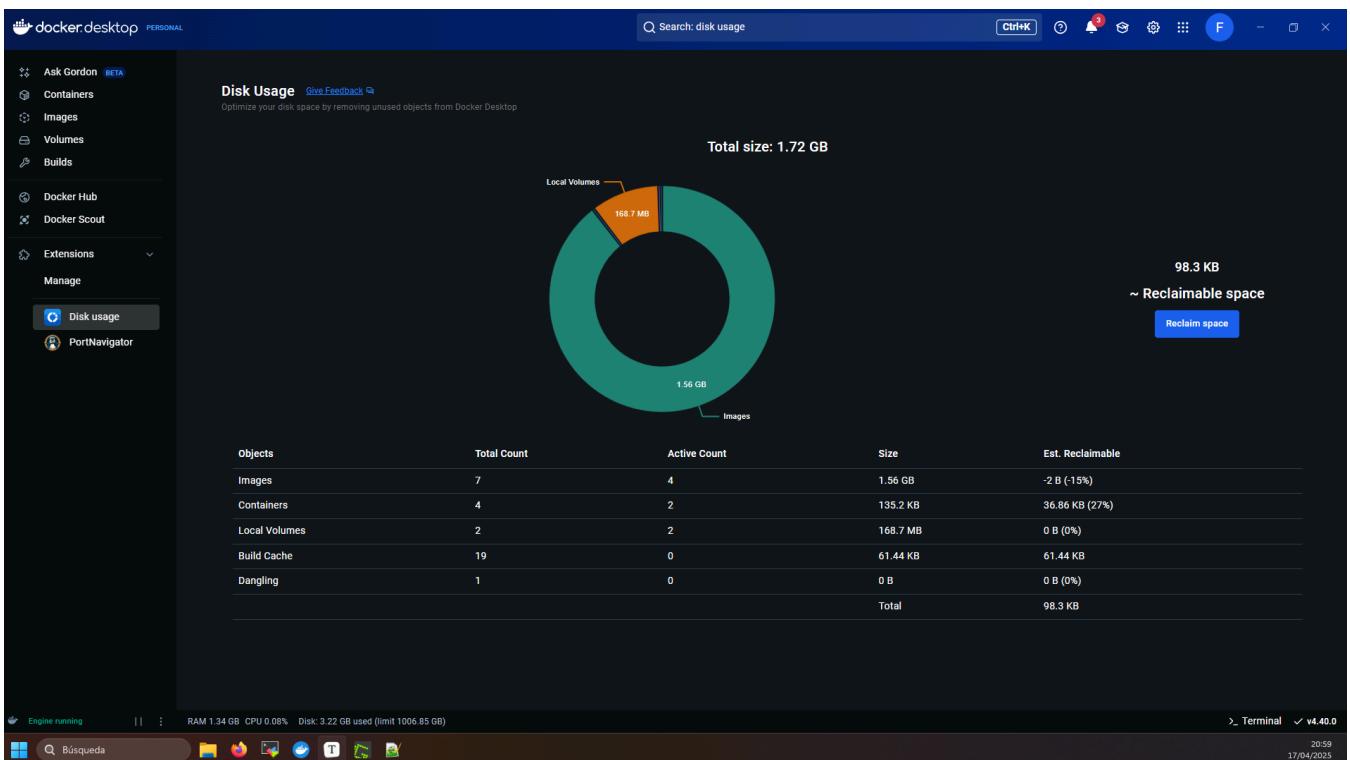
Below the table, there are buttons for 'Resultado completo' (5 registros), 'Modify' (Guardar), 'Selected (0)', and 'Exportar (5)'.

5. Instalación y comprobaciones con Disk Usage

Lo primero que debemos hacer al igual que con `mariadb` y `adminer` es buscarlo en el buscador, nos vamos a la pestaña `Extensions` y pulsamos `Install`.

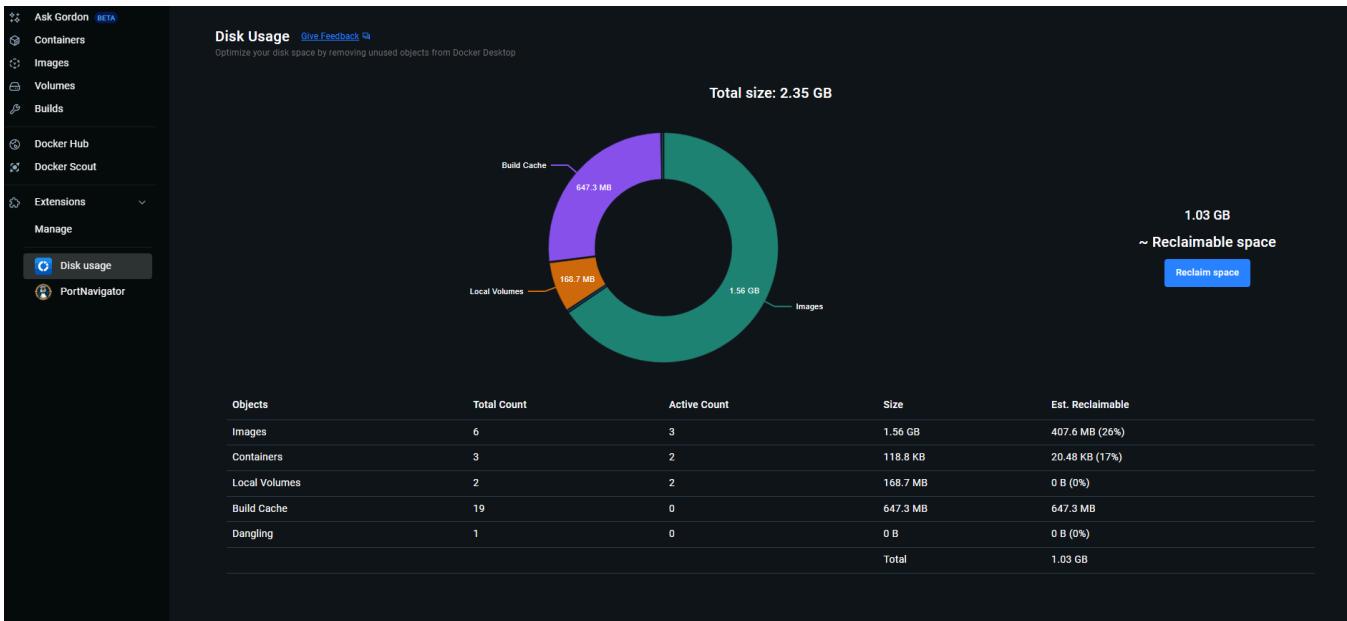


Una vez instalado, en el menú de la izquierda nos vamos a Extensions y seleccionamos Disk usage, nos mostrará un grafico con el espacio utilizado por las imágenes y los volúmenes y un listado de los mismos:

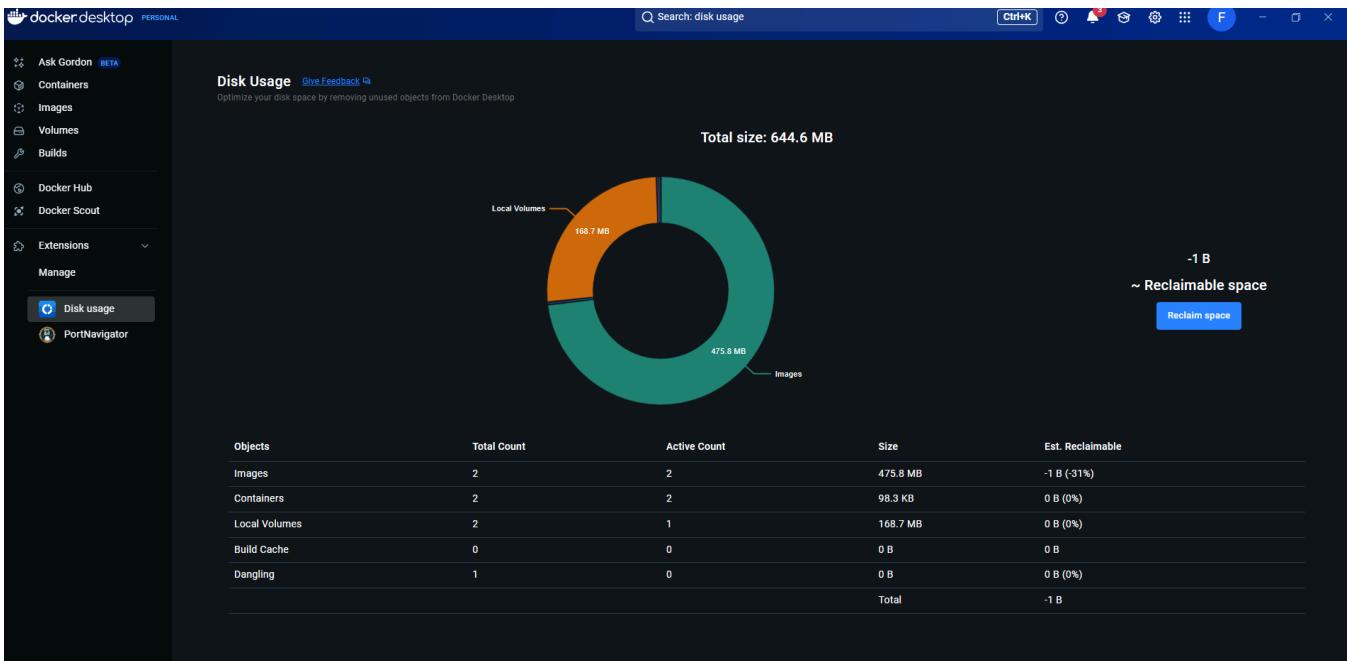


Borramos algún contenedor y alguna imagen

Name	Container ID	Image	Port(s)	CPU (%)	Last started	Actions
magical.volhard	ebdfb0281209	docker	8080:80	0%	3 hours ago	
magical.almeida	93b0d587e871	fuejomusta/ejercicio3:v1	8080:80	0%	3 hours ago	
mariadb_ej1	bfad0f356fde	mariadb:latest	3306:3306	0.02%	33 minutes ago	
Adminer_ej1	ee941df42a9d	adminer:latest	8081:8080	0%	21 minutes ago	



Vemos que tenemos mucho almacenamiento usado en cache, pero usando la herramienta `Reclaim space` (botón azul a la derecha de la imagen) la cual nos indica que tenemos 1.03GB para liberar, lo liberamos:



Borrado de volúmenes, contenedores y la red

Para eliminar los contenedores, en el menú contenedores usamos el botón `delete` con forma de cubo de basura:

Containers [Give feedback](#)

View all your running containers and applications. [Learn more](#)

Container CPU usage [?](#)
0.03% / 1200% (12 CPUs available)

Container memory usage [?](#)
294.85MB / 3.67GB

Show charts

Search Only show running containers

<input type="checkbox"/>	Name	Container ID	Image	Port(s)	CPU (%)	Last started	Actions
<input type="checkbox"/>	mariadb_ej1	bfad0f356fde	mariadb:latest	3306:3306	0.02%	41 minutes ago	
<input type="checkbox"/>	Adminer_ej1	ee941df42a9d	adminer:latest	8081:8080	0.01%	29 minutes ago	

docker:desktop PERSONAL

Ask Gordon [BETA](#)

- Containers** [Give feedback](#)
- Images
- Volumes
- Builds
- Docker Hub
- Docker Scout
- Extensions [Manage](#)
- Disk usage
- PortNavigator

Containers [Give feedback](#)

View all your running containers and applications. [Learn more](#)

Your running containers show up here

A container is an isolated environment for your code

What is a container? 5 mins How do I run a container? 6 mins

[View more in the Learning center](#)

Para borrar la red, nos dirigimos a `Extensions`, y en `PortNavigator` nos mostrará todas las redes, y en la red `redej1` pulsamos en la `x` de arriba a la derecha para borrarla.

Network: redej1

Driver: bridge Gateway: 172.18.0.1 Subnet: 172.18.0.0/16

Connected Containers | 0 Add Container

Ejercicio 2 - Docker Compose

Adrián García de la Cera

Tarea3 - Docker

Pasos previos

[Ejercicio 1 - Contenedores en red y Docker Desktop](#)

Enunciado

- 1.Creación de red bridge redej1
2. Crea un contenedor con una imagen de [mariadb](#)
 - Definir credenciales
3. Crear un contenedor con [Adminer](#)
4. Conectar contenedores a redej1
 - Configurar redes
 - Conexión a interfaz gráfica
 - Ejecución del script SQL
- 5.Instalación y comprobaciones con [Disk usage](#)
 - Borrado de volúmenes, contenedores y la red

Ejercicio 2 - Docker Compose

- Enunciado
- Desarrollo
- Creación archivo
 - Inicio de servicio
 - Pruebas con filebrowser
 - Finalización del servicio

Ejercicio 3 - imagen con Dockerfile - Aplicación web

- Enunciado
- Desarrollo
- Creación de archivos
 - Creación de la imagen
 - Comprobaciones
 - Subida de imágenes
- Borrado y descarga de imagen

Enunciado

Explorar la imagen de la aplicación FileBrowser en este repositorio en GitHub: <https://hub.docker.com/r/hurlenko/filebrowser>

Escribir un fichero compose.yaml para desplegarla. Los datos se pueden guardar utilizando volúmenes o utilizando bind-mount.

Entregar, al menos, las siguientes capturas de pantalla y los comandos y/o operaciones con Docker Desktop empleados para resolver el ejercicio:

- Captura de pantalla y documento donde se vea el fichero docker-compose.yaml que has creado.
- Captura de pantalla donde se vean los volúmenes/carpetas donde se han almacenado los datos.
- Captura de pantalla donde se vea la aplicación funcionando, sube algún fichero, cambia el lenguaje a español...
- Explicar brevemente cómo funciona esta aplicación y qué hace.

Desarrollo

El desarrollo del ejercicio se va a realizar en una maquina virtual y lo dividiremos en partes, en la primera crearemos el fichero compose.yaml, y explicaremos sus partes, en la segunda iniciaremos el servicio accediendo a la web y mostrando su funcionamiento realizando cambio de idioma, etc y por ultimo eliminaremos el contenedor.

Para entrar en contexto, filebrowser es un explorador de archivos web que como su web indica:

"filebrowser proporciona una interfaz de gestión de archivos dentro de un directorio especificado y se puede utilizar para cargar, eliminar, previsualizar, renombrar y editar sus archivos. Permite la creación de múltiples usuarios y cada usuario puede tener su propio directorio. Se puede utilizar como una aplicación independiente o como middleware."

Creación archivo

Lo primero que vamos a hacer es crear el directorio donde alojar el archivo compose.yaml y el propio archivo con los siguientes comandos:

Creamos el directorio, accedemos al mismo y creamos el archivo compose.yaml:

```
mkdir ejercicioDocker2  
cd ejercicioDocker2  
touch compose.yaml
```

```
docker@ej...:~$ mkdir ejercicioDoker2  
docker@ej...:~$ cd ejercicioDoker2  
bash: cd: ejercicioDoker2: No existe el archivo o el directorio  
docker@ej...:~$ cd ejercicioDoker2  
docker@ej...:~/ejercicioDoker2$ touch compose.yaml  
docker@ej...:~/ejercicioDoker2$ ls  
compose.yaml  
docker@ej...:~/ejercicioDoker2$ █
```

Una vez creado el archivo accedemos al mismo

```
nano compose.yaml
```

Una vez dentro completamos con el siguiente codigo:

```
version: '3.8'  
  
services:  
  filebrowser:  
    image: hurlenko/filebrowser  
    container_name: filebrowser  
    restart: unless-stopped  
    ports:  
      - "8080:8080"  
    volumes:  
      - ./filebrowser/config:/config  
      - ./filebrowser/data:/srv  
    environment:
```

```
- FB_BASEURL=/filebrowser
```

El archivo compose.yaml se divide en varias partes:

1. La versión en este caso 3.8
2. Se definen los servicios (contenedores) que se van a desplegar en este caso filebrowser.
3. Se especifica la imagen del contenedor a utilizar.
4. Asignamos un nombre al contenedor.
5. Definimos la política de reinicio, para que se reinicie automáticamente si se detiene excepto si el usuario lo detiene manualmente.
6. Mapeamos el puerto 8080:8080 para acceder a la interfaz.
7. Montamos los volúmenes de configuración y de archivos, para que cuando se elimine o reinicie el servicio no se pierda la información.
8. Configuramos la URL base la aplicación.

```
GNU nano 6.2          compose.yaml
version: '3.8'          1
services:
  filebrowser:
    image: hurlenko/filebrowser
    container_name: filebrowser
    restart: unless-stopped
  ports:
    - "8080:8080"
  volumes:
    - ./filebrowser/config:/config
    - ./filebrowser/data:/srv
  environment:
    - FB_BASEURL=/filebrowser
```

[14 líneas leídas]

^G Ayuda ^O Guardar ^W Buscar ^K Cortar ^T Ejecutar ^C Ubicación

Una vez configurado el archivo compose.yaml y guardado ejecutamos el servicio en segundo plano.

Inicio de servicio

```
docker compose up -d
```

Esto descargará la imagen, creará la red, creará y arrancara el contendor y montará los volúmenes.

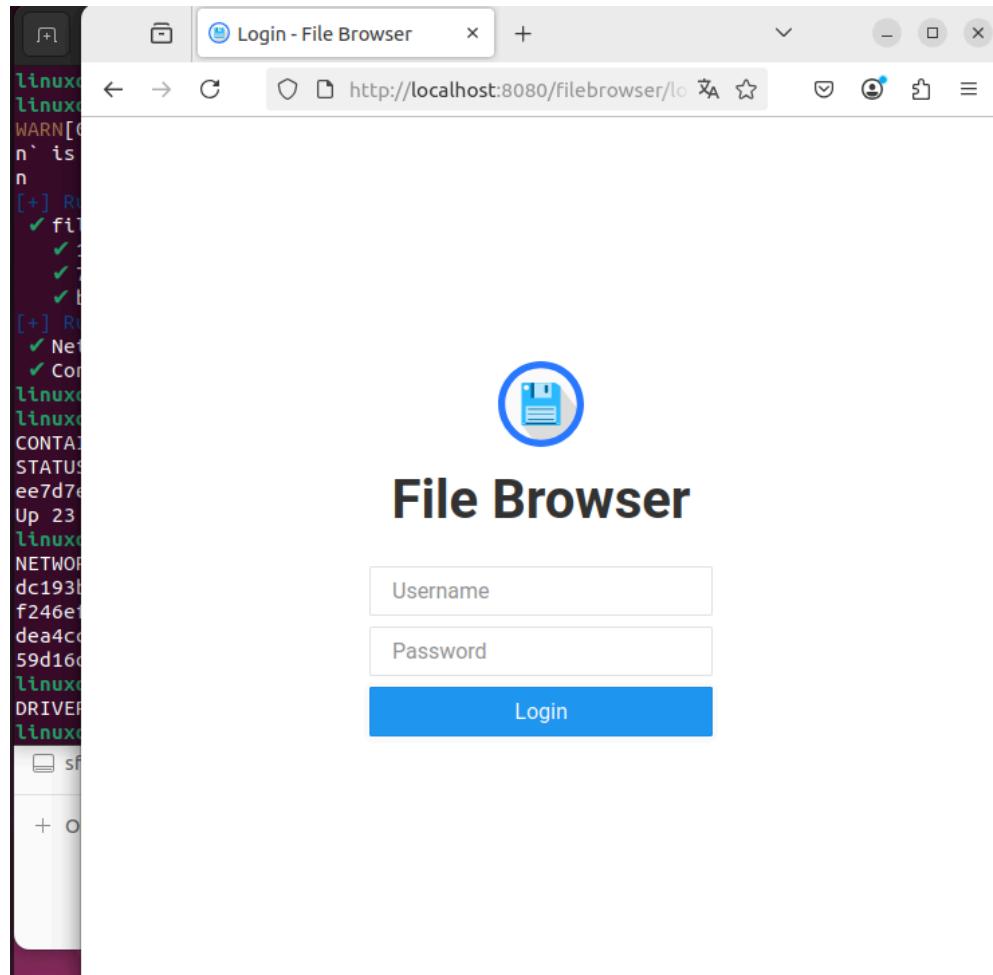
```

linuxdocker@linuxdocker-VirtualBox:~/ejercicioDoker2$ docker compose up -d
WARN[0001] /home/linuxdocker/ejercicioDoker2/compose.yaml: the attribute `versio
n` is obsolete, it will be ignored, please remove it to avoid potential confusio
n
[+] Running 4/4
  ✓ filebrowser Pulled
    ✓ 1f3e46996e29 Pull complete          5.3s
    ✓ 7b276281f0bb Pull complete          1.6s
    ✓ bbe6e0b5af75 Pull complete          2.7s
[+] Running 2/2
  ✓ Network ejerciciodoker2_default  C...
  ✓ Container filebrowser           Started          2.8s
linuxdocker@linuxdocker-VirtualBox:~/ejercicioDoker2$ nano compose.yaml
linuxdocker@linuxdocker-VirtualBox:~/ejercicioDoker2$ docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED        NAMES
STATUS         PORTS          NAMES
ee7d7ef49fdb  hurlenko/filebrowser   "/filebrowser --root..."   23 minutes ago
Up 23 minutes  0.0.0.0:8080->8080/tcp, [::]:8080->8080/tcp   filebrowser
linuxdocker@linuxdocker-VirtualBox:~/ejercicioDoker2$ docker network ls
NETWORK ID     NAME          DRIVER      SCOPE
dc193b91cc76   bridge        bridge      local
f246ef2b60be   ejerciciodoker2_default  bridge      local
dea4cc626aa3   host          host       local
59d16d7cdb9d   none          null       local

```

Ya creado el archivo compose.yaml y montado comprobamos si accedemos correctamente al servicio.

<http://localhost:8080>

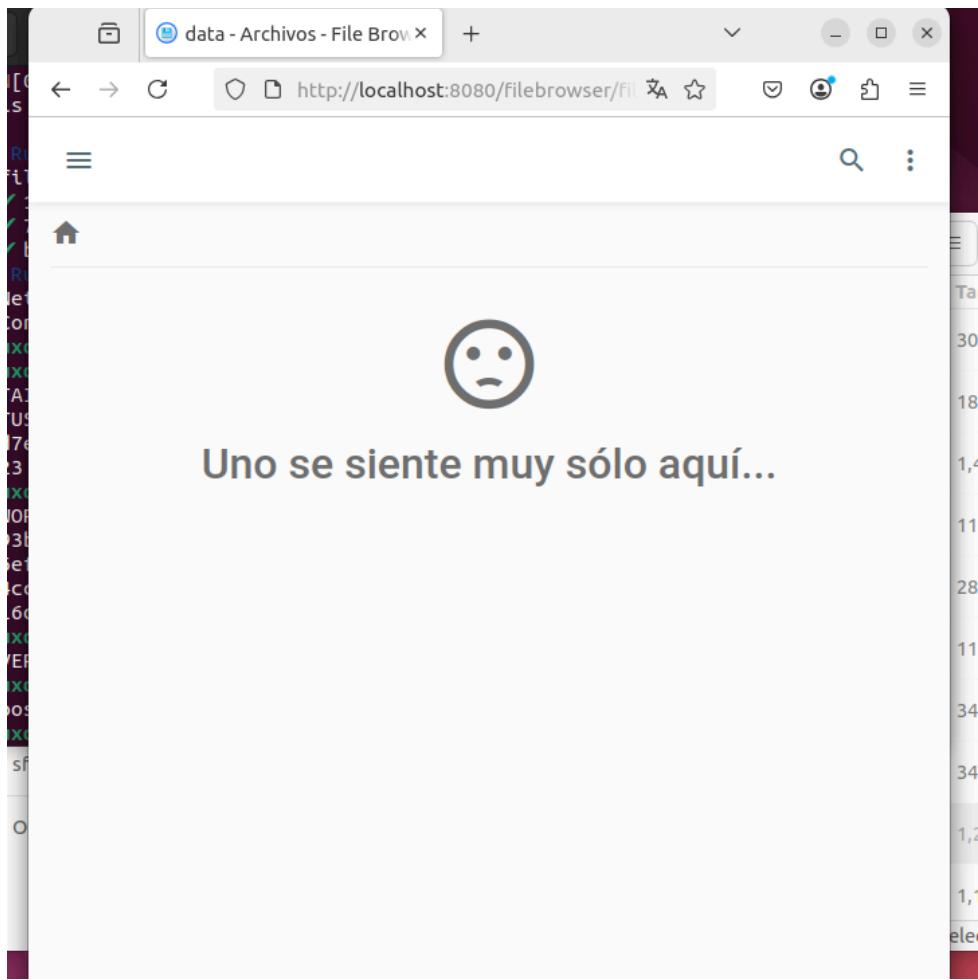


Pruebas con filebrowser

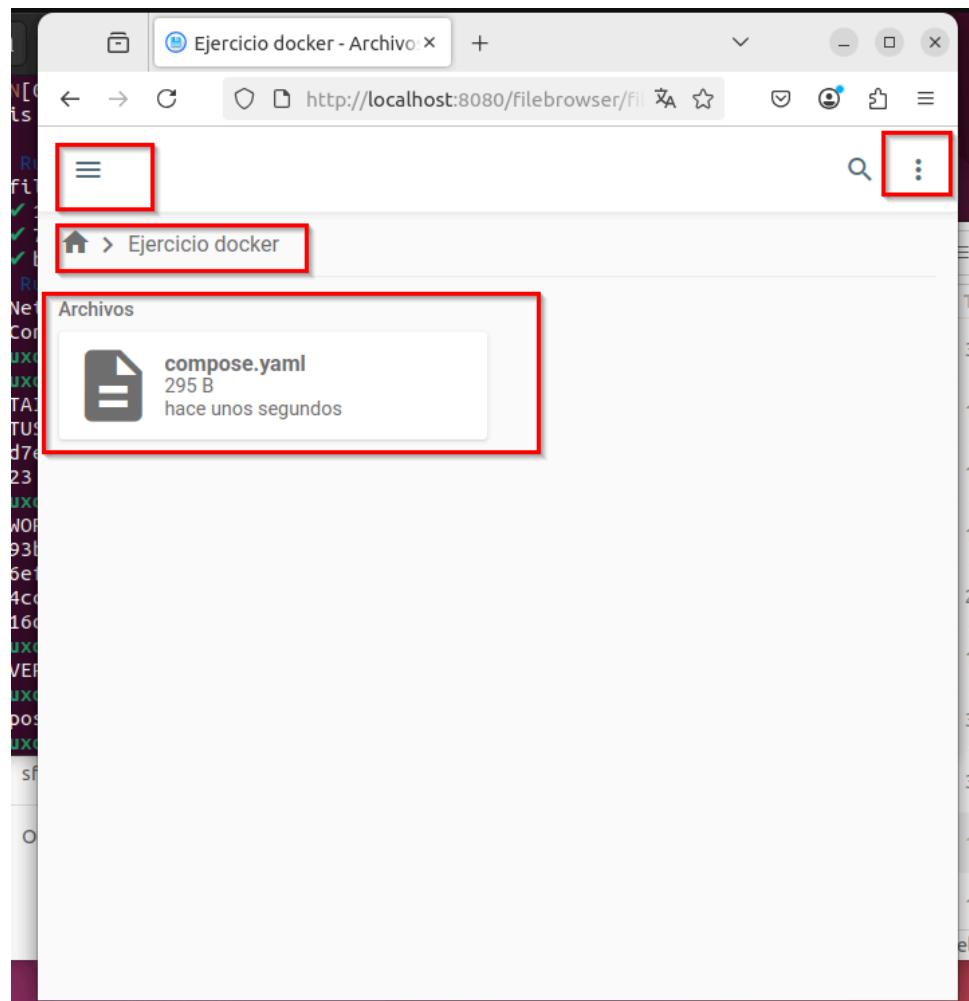
Nos logamos con las credenciales que se indican en el enunciado (user: admin password: admin) y procedemos a cambiar el idioma en Settings

The screenshot shows a web browser window titled "Profile Settings - File Browser". The URL in the address bar is "http://localhost:8080/filebrowser/settings". A dropdown menu is open, listing various languages. The option "Español" is highlighted with a grey background. Other options include עברית, Magyar, العربية, Català, Deutsch, Ελληνικά, English, Français, Icelandic, Italiano, 日本語, 한국어, Dutch (Belgium), and English. Below the dropdown is a button labeled "UPDATE". At the bottom of the page, there is a "Change Password" section with a field labeled "Your new password".

Como vemos a continuación, de momento no tenemos ningún archivo subido a filebrowser:



Vamos a crear dentro de filebrowser una carpeta llamada "Ejercicio docker" , para ello, en la esquina superior derecha hacemos clic en las 3 barras para que nos despliegue el menú y seleccionamos nueva carpeta, lo que nos abrirá un cuadro para indicar el nombre de la misma, una vez creada directamente accede a la misma, ahora en la parte superior derecha damo a los 3 puntos y seleccionamos subir, nos abrirá el gestor de archivos de linux y seleccionamos el fichero a subir en nuestro caso hemos seleccionado el propio archivo compose.yaml.



Finalización del servicio

Para finalizar vamos a detener el servicio y eliminar usamos el comando

```
docker compose down
```

En la siguiente imagen vemos como tras detener el servicio, como se han usado volúmenes a persistido la información:

```
linuxdocker@linuxdocker-VirtualBox:~/ejercitodocker2$ cd ejercicioDoker2/
linuxdocker@linuxdocker-VirtualBox:~/ejercitodoker2$ tree
.
└── compose.yaml
    └── filebrowser
        ├── config
        └── filebrowser.db
            └── data
                └── filebrowser.db

3 directories, 2 files
```

Ejercicio 3 - imagen con Dockerfile - Aplicación web

Adrián García de la Cera

Tarea3 - Docker

Pasos previos

Ejercicio 1 - Contenedores en red y Docker Desktop

Enunciado

1.Creación de red bridge redej1

2. Crea un contenedor con una imagen de `mariadb`

Definir credenciales

3. Crear un contenedor con `Adminer`

4. Conectar contenedores a redej1

Configurar redes

Conexión a interfaz grafica

Ejecución del script SQL

5.Instalación y comprobaciones con `disk usage`

Borrado de volúmenes, contenedores y la red

Ejercicio 2 - Docker Compose

Enunciado

Desarrollo

Creación archivo

Inicio de servicio

Pruebas con filebrowser

Finalización del servicio

Ejercicio 3 - imagen con Dockerfile - Aplicación web

Enunciado

Desarrollo

Creación de archivos

Creación de la imagen

Comprobaciones

Subida de imágenes

Borrado y descarga de imagen

Enunciado

Necesitamos un fichero Dockerfile que automatice las siguientes operaciones para crear una imagen que contenga un servidor con un sitio web y un script php. Características de la imagen:

- Usa un contenedor que ejecute una instancia de la imagen `php:7.4-apache`, que se llame ejercicio3 y que sea accesible desde un navegador en el puerto 8000.
- Coloca en el directorio raíz del servicio web (`/var/www/html`) un "sitio web" donde figure tu nombre - el sitio deberá tener al menos un archivo `index.html` sencillo y un archivo `.css`.
- Coloca en ese mismo directorio raíz el siguiente script php , llámalo `fecha.php`.

```

<?php
setlocale(LC_TIME, "es_ES.UTF-8");
$mes_actual = strftime("%B");
$fecha_actual = date("d/m/Y");
$hora_actual = date("H:i:s");
echo "<h1>Información</h1>";
echo "<p>Hoy es $fecha_actual</p>";
echo "<p>El mes es: <strong>$mes_actual</strong></p>";
echo "<p>Hora: $hora_actual</p>";
?>

```

- Ver la salida del script `fecha.php` y de la página `index.html` en el navegador.

Una vez creada la imagen, súbelo a tu cuenta de Docker Hub

- Borra la imagen de tu Docker local
- Baja ('pull') de tu cuenta la imagen que acabas de subir
- Muestra las imágenes que tienes
- Ejecuta un contenedor usando esa imagen

Deberás entregar, al menos, las siguientes capturas de pantalla, los comandos empleados y/o operaciones con Docker Desktop para resolver cada apartado:

- creación inicial del contenedor - documenta los pasos hasta el borrado del mismo.
- bloque de código con el Dockerfile.
- creación de la nueva imagen.
- subida de la imagen a tu cuenta de Docker Hub.
- operación de 'pull' de la imagen de Docker Hub
- creación de un nuevo contenedor con esa imagen y su ejecución.
Cambia el puerto del contenedor, por ejemplo, `-p 1234:80`.
- el acceso al navegador con la página html y con el script `php`.

Desarrollo

Creación de archivos

Lo primero que vamos a realizar para la ejecución del ejercicio es crear los ficheros `index.html`, `estilos.css` y `fecha.php` en la carpeta web.

```
index.html estilos.css fecha.php Dockerfile
1 <!DOCTYPE html>
2 <html lang="es">
3   <head>
4     <meta charset="UTF-8">
5     <meta name="viewport" content="width=device-width, initial-scale=1.0">
6     <title>Ejercicio3.Dockerfile de Adrián García</title>
7     <link rel="stylesheet" href="estilos.css">
8   </head>
9   <body>
10    <h1>Ejercicio3.Dockerfile</h1>
11    <h2>Asignatura Despliegue de aplicaciones web</h2>
12    <p>Mi nombre es Adrián García de la Cera</p>
13    <p>Este es el ejercicio 3 de Dockerfile del modulo de despliegue del FP de Desarrollo de aplicaciones Web impartido en el el CIFP Laboral</p>
14  </body>
15 </html>
```

```
index.html estilos.css fecha.php Dockerfile
1 body {
2   font-family: Arial, sans-serif;
3   line-height: 1.6;
4   margin: 0;
5   padding: 20px;
6   background-color: #f4f4f4;
7   color: #333;
8 }
9
10 h1 {
11   color: #0066cc;
12 }
```

```
index.html estilos.css fecha.php Dockerfile
1 <?php
2   setlocale(LC_TIME, "es_ES.UTF-8");
3   $mes_actual = strftime("%B");
4   $fecha_actual = date("d/m/Y");
5   $hora_actual = date("H:i:");
6   echo "<h1>Informacion</h1>";
7   echo "<p>Hoy es $fecha_actual</p>";
8   echo "<p>El mes es: <strong>$mes_actual</strong></p>";
9   echo "<p>Hora: $hora_actual</p>";
10 ?>
```

Una vez creados estos, creamos el archivo Dockerfile en el directorio raíz, el cual se compone de 3 elementos:

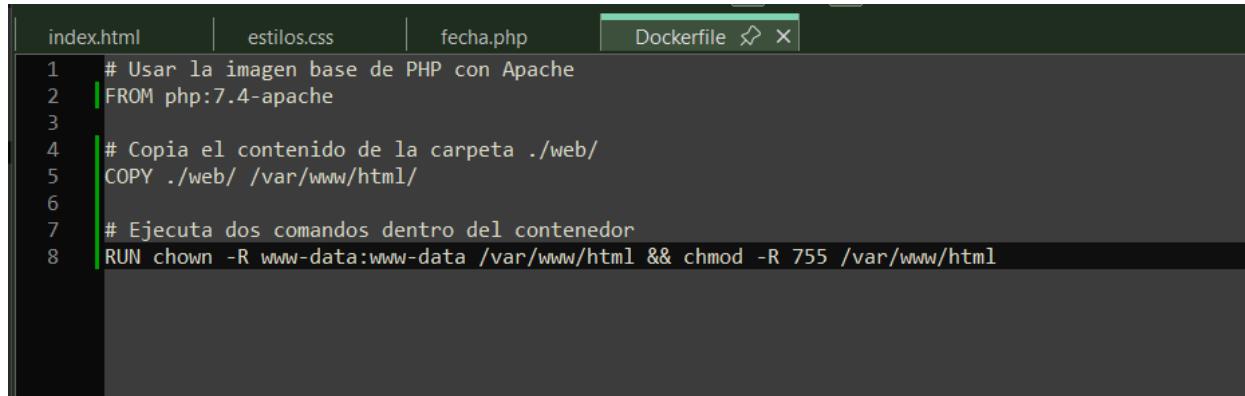
1. FROM: Indica la imagen base para construir la imagen Docker.
2. COPY: Copia el contenido de la carpeta web al directorio `/var/www/html/`.

3. RUN: Ejecuta los comandos `chown -R www-data:www-data /var/www/html` cambia el grupo y el propietario de los archivos al directorio que usa el servidor apache y `chmod -R 755 /var/www/html` modifica los permisos para que el propietario pueda leer, escribir y ejecutar y el grupo solo leer y ejecutar.

```
# Usar la imagen base de PHP con Apache
FROM php:7.4-apache

# Copia el contenido de la carpeta ./web/
COPY ./web/ /var/www/html/

# Ejecuta dos comandos dentro del contenedor
RUN chown -R www-data:www-data /var/www/html && chmod -R 755 /var/www/html
```



```
index.html | estilos.css | fecha.php | Dockerfile ⌂ X
1 # Usar la imagen base de PHP con Apache
2 FROM php:7.4-apache
3
4 # Copia el contenido de la carpeta ./web/
5 COPY ./web/ /var/www/html/
6
7 # Ejecuta dos comandos dentro del contenedor
8 RUN chown -R www-data:www-data /var/www/html && chmod -R 755 /var/www/html
```

Creación de la imagen

Para crear la imagen usamos el siguiente comando:

```
docker build -t adriangarciaejerciciodokerfile/ejercicio3:v1 .
```

El comando se compone de las siguientes partes:

- docker build: Es el comando para crear la imagen.
- -t: Es la etiqueta que usamos para dar un nombre a la imagen.
- adriangarciaejerciciodokerfile: Es el nombre de la imagen, en este caso comienda con mi nombre y primer apellido (`adrian_garcia`) al que se le ha agregado el nombre del ejercicio.
- ejercicio3: Se trata del nombre de la imagen.
- v1: Es la etiqueta de la imagen.
- . : Indica la ruta en la que están los archivos necesarios como es la actual se indica con "..".

The screenshot shows a terminal window in Mobaxterm. The title bar indicates the current directory is `/home/mobaxterm/Desktop/Ejercicio3-Dockerfile`. The menu bar includes Terminal, Sessions, View, X server, Tools, Games, Settings, Macros, Help, Session, Servers, Tools, Games, Sessions, View, Split, MultiExec, Tunneling, Packages, Settings, Help, X server, and Exit. On the left, there's a sidebar with 'Quick connect...', 'User sessions' (marked with a yellow star), and 'PUTTY sessions'. The main terminal area displays the output of a `docker build` command:

```
[+] Building 20.7s (9/9) FINISHED
--> [internal] load build definition from Dockerfile          docker:desktop-linux
--> => transferring dockerfile: 296B                         0.1s
--> [internal] load metadata for docker.io/library/php:7.4-apache   0.0s
--> [auth] library/php:pull token for registry-1.docker.io      2.5s
--> [internal] load .dockerignore                                0.0s
--> => transferring context: 2B                               0.1s
--> [internal] load build context                            0.0s
--> => transferring context: 3.84kB                          0.2s
--> [1/3] FROM docker.io/library/php:7.4-apache@sha256:c9d7e608f73832673 12.9s
--> => resolve docker.io/library/php:7.4-apache@sha256:c9d7e608f738326734 0.1s
--> => sha256:ab590b48ea476386dd7b07c34de9eff7cf2103c4668 2.46kB / 2.46kB 0.4s
--> => sha256:05e465aaa99a358add4acecdade8f39843089069f31fea0 892B / 892B 0.4s
--> => sha256:80692ae2d067c8358112c56490a2a9f769ef395fd8f7662 246B / 246B 0.3s
--> => sha256:d2c43c5efbc861f83ee6565c7102ca660d6f35e15 10.20MB / 10.20MB 1.1s
--> => sha256:66d98f73acb62e86c0c226f9eedcbc7eda305df0c1e171c 491B / 491B 0.3s
--> => sha256:d14eb2ed1e17ae00f5fcba4b0d562e2867c401c20 10.76MB / 10.76MB 1.3s
--> => sha256:fe42347c4ecfc90333acd9cad13912387eaa39d13827a25 514B / 514B 0.3s
--> => sha256:9b233e420ac7bbca645bb82c213029762acf1742400c076 475B / 475B 0.2s
--> => sha256:25f85b498fd5bfc6cce951513219fe480850daba7 19.25MB / 19.25MB 2.6s
--> => sha256:fb5a4c8af82f00730b7427e47bda7f76cea2e2b9aea4217 270B / 270B 0.2s
--> => sha256:156740b07ef8a632f9f7bea4e57e4ee5541ade376 91.63MB / 91.63MB 6.4s
--> => sha256:c428f1a494230852524a2a5957cc5199c36c8b403305e0e 226B / 226B 0.2s
--> => sha256:a603fa5e3b4127f210503aaa6189abf6286ee5a73 31.41MB / 31.41MB 3.4s
--> => extracting sha256:a603fa5e3b4127f210503aaa6189abf6286ee5a73deeaaab4 2.6s
--> => extracting sha256:f428f1a494230852524a2a5957cc5199c36c8b403305e0e8 0.1s
--> => extracting sha256:156740b07ef8a632f9f7bea4e57e4ee5541ade376adf9169 2.9s
--> => extracting sha256:fb5a4c8af82f00730b7427e47bda7f76cea2e2b9aea42175 0.0s
--> => extracting sha256:25f85b498fd5bfc6cce951513219fe480850daba71e6e997 0.4s
--> => extracting sha256:9b233e420ac7bbca645bb82c213029762acf1742400c0763 0.0s
--> => extracting sha256:fe42347c4ecfc90333acd9cad13912387eaa39d13827a25c 0.0s
--> => extracting sha256:d14eb2ed1e17ae00f5fcba4b0d562e2867c401c20372829e 0.1s
--> => extracting sha256:66d98f73acb62e86c0c226f9eedcbc7eda305df0c1e171ca 0.0s
--> => extracting sha256:d2c43c5efbc861f83ee6565c7102ca660d6f35e158324fbb 0.4s
--> => extracting sha256:ab590b48ea476386dd7b07c34de9eff7cf2103c4668ade98 0.0s
--> => extracting sha256:80692ae2d067c8358112c56490a2a9f769ef395fd8f7662a 0.0s
--> => extracting sha256:05e465aaa99a358add4acecdade8f39843089069f31fea02 0.0s
--> [2/3] COPY ./web/ /var/www/html/                           0.8s
--> [3/3] RUN chown -R www-data:www-data /var/www/html && chmod -R 755 /v 2.2s
--> => exporting to image                                     1.6s
--> => exporting layers                                      0.9s
--> => exporting manifest sha256:5d09b2fb002e9cec9a6831668938763ee417f0c4 0.0s
--> => exporting config sha256:8d2af1234f011e8191ef9175f03d8a354fea86cf84 0.0s
--> => exporting attestation manifest sha256:a2185b58fdf4258c343cb924134f 0.1s
--> => exporting manifest list sha256:969c42a5cef539c7c8250b662ee04ef079e 0.1s
--> => naming to docker.io/adriangarciaejericiodokerfile/ejercicio3:v1 0.0s
--> => unpacking to docker.io/adriangarciaejericiodokerfile/ejercicio3:v 0.3s
```

View build details: <https://docker-desktop://dashboard/build/desktop-linux/desktop-linux/mhe7lo4xlv5zvqow25kwikjj>

The bottom status bar shows the date and time as 2025-04-17 16:50:29.

UNREGISTERED VERSION - Please support Mobaxterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>

Comprobamos que ha creado la imagen:

`docker images`

<code>docker images</code>			
REPOSITORY	TAG	IMAGE ID	CREATED
adriangarciaejericiodokerfile/ejercicio3	v1	969c42a5cef5	7 minutes ago
docker/welcome-to-docker	latest	eedaff45e3c7	17 months ago

Ejecutamos el contenedor en modo desatendido (`-d`) en el puerto 8080:

```
docker run -d -p 8080:80 adriangarciaejerciciodokerfile/ejercicio3:v1
```

Como podemos ver en Docker Desktop ya vemos la imagen creada:

Docker Desktop Personal

Images

Local Docker Hub repositories

483.97 MB / 21.67 MB in use 2 images Last refresh: 25 minutes ago

Name	Tag	Image ID	Created	Size	Actions
docker/welcome-to-docker	latest	eedaff45e3c7	1 year ago	29.46 MB	...
adriangarciaejerciciodol	v1	969c42a5cef5	15 minutes ago	647.25 MB	...

Showing 2 items

Walkthroughs

FROM node How do I run a container? Run Docker Hub images Terminal ✓ v4.40.0

Comprobamos si se accede a través del navegador:

Ejercicio3.Dockerfile

Asignatura Despliegue de aplicaciones web

Mi nombre es Adrián García de la Cera

Este es el ejercicio 3 de Dockerfile del modulo de despliegue del FP de Desarrollo de aplicaciones Web impartido en el el CIP Laboral

Comprobaciones

Subida de imágenes

Para subir una imagen a Docker Hub lo primero es logarse con el comando `docker login`

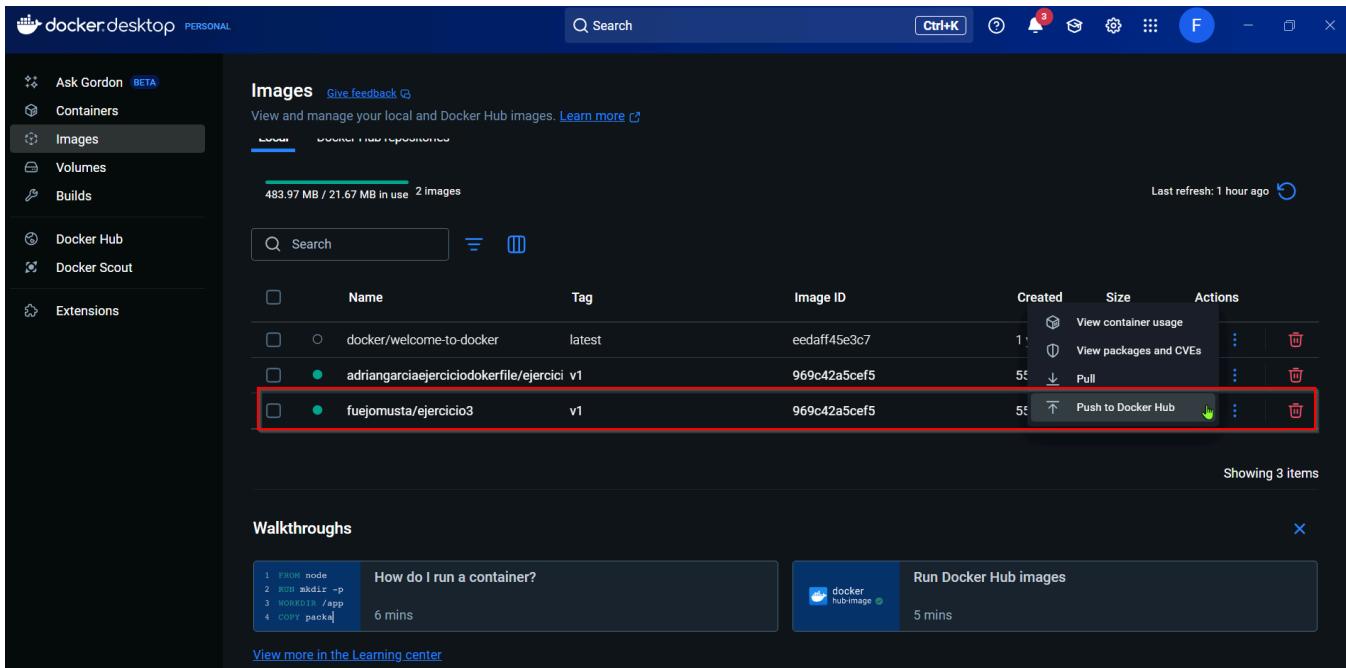
```
2025-04-17 17:16.33 /home/mobaxterm/Desktop/Ejercicio3-Dockerfile docker login
Authenticating with existing credentials... [Username: fuejomusta]
Info → To login with a different account, run 'docker logout' followed by 'docker login'

Login Succeeded
```

En este caso al intentar subir la imagen a Docker hub da error por que el nombre indicado en la imagen no coincide con el usuario de Docker hub, por lo que procedemos a modificar la etiqueta de la imagen de `adriangarciaejeriodokerfile/ejercicio3:v1` a `fuejomusta/ejercicio3:v1`, creando una nueva etiqueta para esa imagen y permitiendo subirla:

```
docker tag adriangarciaejeriodokerfile/ejercicio3:v1 fuejomusta/ejercicio3:v1
```

Una vez solucionado el problema de la etiqueta tenemos 2 opciones para subir la imagen a Docker Hub, a través de Docker Desktop, de manera muy sencilla, junto a la etiqueta aparecen 3 simbolos, un botón de play, un cubo de basura y 3 puntos verticales que al pulsar sobre ellos nos dejará indicar `Push to Docker Hub`, lo que iniciará la subida:

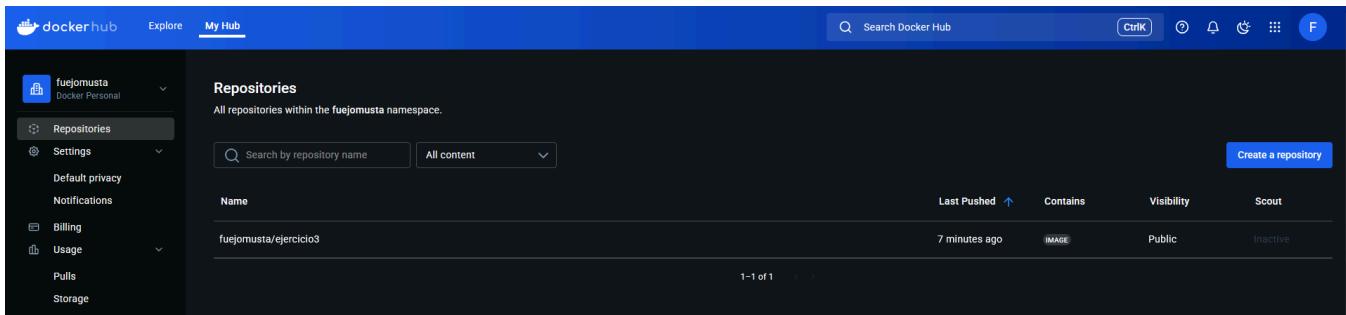


The screenshot shows the Docker Desktop interface. On the left, there's a sidebar with options like Ask Gordon, Containers, Images (which is selected), Volumes, Builds, Docker Hub, Docker Scout, and Extensions. The main area is titled 'Images' and shows 'View and manage your local and Docker Hub images'. It lists three images: 'docker/welcome-to-docker' (latest tag, 483.97 MB), 'adriangarciaejeriodokerfile/ejercicio3 v1' (969c42a5cef5, 56 MB), and 'fuejomusta/ejercicio3 v1' (969c42a5cef5, 56 MB). The third image's Actions menu is open, and the 'Push to Docker Hub' button is highlighted with a red box.

Otra manera es a través de la terminal con el código:

```
docker push fuejomusta/ejercicio3:v1
```

Una vez confirmado que ha subido la imagen, accedemos a nuestro repositorio en la web de Docker Hub y confirmamos que está la imagen creada.



The screenshot shows the Docker Hub 'My Hub' page. On the left, there's a sidebar with 'fuejomusta' selected, showing options like Repositories, Settings, Default privacy, Notifications, Billing, Usage, Pulls, and Storage. The main area is titled 'Repositories' and shows 'All repositories within the fuejomusta namespace.' It lists one repository: 'fuejomusta/ejercicio3'. The details for this repository show it was last pushed 7 minutes ago, contains an image, is public, and is inactive.

Borrado y descarga de imagen

Una vez subida la imagen procedemos a borrar la imagen local bien con el botón con símbolo de basura de Docker Desktop o con el comando:

```
docker rmi fuejomusta/ejercicio3:v1
```

Es posible que nos dé error por estar trabajando el contenedor, en tal caso será necesario comprobar el contenedor (`docker ps -a`), parar la ejecución del contenedor (`docker stop nombre contenedor`) y luego borrarlo (`docker rm -f nombre contenedor`).

Una vez borrado comprobamos que ya no existe la imagen en Docker Desktop o con comando:

```
docker images
```

Docker Desktop interface showing the Images tab. The sidebar has options like Ask Gordon, Containers, Images (selected), Volumes, Builds, Docker Hub, Docker Scout, and Extensions. The main area shows 1 image: docker/welcome-to-docker (latest tag, eedaff45e3c7, created 1 year ago, 29.46 MB). Below the table, it says "Showing 1 item". At the bottom, there's a Walkthroughs section with "How do I run a container?" and a "Run Docker Hub images" button.

```
2025-04-17 17:54:35 /home/mobaxterm/Desktop/Ejercicio3-Dockerfile docker images
REPOSITORY          TAG      IMAGE ID   CREATED        SIZE
docker/welcome-to-docker  latest   eedaff45e3c7  17 months ago  29.5MB
```

Para descargar la imagen de nuevo, al igual que para eliminarla existen 2 posibilidades, buscarla desde Docker Desktop y descargarla con el botón "Pull" o directamente con el comando:

```
docker pull fuejomusta/ejercicio3:v1
```

Y comprobamos de nuevo tenemos la imagen

Docker Desktop interface showing the Images tab. The sidebar has options like Ask Gordon, Containers, Images (selected), Volumes, Builds, Docker Hub, Docker Scout, and Extensions. The main area shows 2 images: docker/welcome-to-docker (latest tag, eedaff45e3c7, created 1 year ago, 29.46 MB) and fuejomusta/ejercicio3 (v1 tag, 969c42a5cef5, created 2 hours ago, 647.25 MB). The Actions column for each image includes a trash icon.

```
2025-04-17 17:59:57 /home/mobaxterm/Desktop/Ejercicio3-Dockerfile docker pull fuejomusta/ejercicio3:v1
v1: Pulling from fuejomusta/ejercicio3
2968821aa7c5: Pull complete
845fb9d5029: Pull complete
Digest: sha256:969c42a5cef539c7c8250b662ee04ef079e177758b1227ba0493ab2f0381db75
Status: Downloaded newer image for fuejomusta/ejercicio3:v1
docker.io/fuejomusta/ejercicio3:v1
```

```
2025-04-17 18:22:27 /home/mobaxterm/Desktop/Ejercicio3-Dockerfile docker images
REPOSITORY          TAG      IMAGE ID      CREATED       SIZE
fuejomusta/ejercicio3   v1      969c42a5cef5  2 hours ago  647MB
docker/welcome-to-docker  latest  eedaff45e3c7  17 months ago  29.5MB
```

Una vez descargada, es necesario iniciarla, para ello usamos el comando:

```
docker run -d -p 8080:80 fuejomusta/ejercicio3:v1
```

The screenshot shows a web browser window with the URL `localhost:8080`. The page content is as follows:

Ejercicio3.Dockerfile

Asignatura Despliegue de aplicaciones web

Mi nombre es Adrián García de la Cera

Este es el ejercicio 3 de Dockerfile del modulo de despliegue del FP de Desarrollo de aplicaciones Web impartido en el el CIFP Laboral

