

# Actividad - Ejercicios Docker

Tarea realizada por Alejandro Cortina Santos, Rubén Gómez Regüela y Miguel Valle Miranda

## Actividad - Ejercicios

### Docker

Ejercicio 1. Inicial

Ejercicio 2. Trabajo con imágenes

Servidor web

Servidor de base de datos

Ejercicio 3. Almacenamiento

Ejercicio 4. Redes

Despliegue de contenedores en red: Adminer y MariaDB

Ejercicio 5. Crea una imagen con Dockerfile

Crear una imagen con un servidor web que sirva un sitio web

## Ejercicio 1. Inicial

Crear un contenedor demonio a partir de la imagen `nginx`, el contenedor se debe llamar `servidor_web` y se debe acceder a él utilizando el puerto 8181 del ordenador donde tengas instalado Docker.

Entrega un documento con los siguientes pantallazos, y los comandos empleados para resolver cada apartado:

1. Pantallazo donde se vea la creación del contenedor y podamos comprobar que el contenedor está funcionando.

```
docker run -d --name servidor_web -p 8181:80 nginx
```

```
cliente@cliente-VirtualBox:~$ docker run -d --name servidor_web -p 8181:80 nginx
Unable to find image 'nginx:latest' locally
latest: Pulling from library/nginx
5eb5b503b376: Pull complete
1ae07ab881bd: Pull complete
78091884b7be: Pull complete
091c283c6a66: Pull complete
55de5851019b: Pull complete
b559bad762be: Pull complete
Digest: sha256:2834dc507516af02784808c5f48b7cbe38b8ed5d0f4837f16e78d00deb7e7767
Status: Downloaded newer image for nginx:latest
58198e190322ddf953dd9144d779ad517b812686d73694e21042e6f6e84133ae
```

```
docker ps
```

```
cliente@cliente-VirtualBox:~$ docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED        STATUS
PORTS          NAMES
58198e190322   nginx    "/docker-entrypoint.    3 minutes ago Up 3 minutes
0.0.0.0:8181->80/tcp, :::8181->80/tcp   servidor_web
```

2. Pantallazo donde se vea el acceso al servidor web utilizando un navegador web (recuerda que tienes que acceder a la ip del ordenador donde tengas instalado docker)



3. Pantallazo donde se vean las imágenes que tienes en tu registro local.

```
docker images
```

```
cliente@cliente-VirtualBox:~$ docker images
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
nginx         latest    c316d5a335a5   7 days ago    142MB
```

4. Pantallazo donde se vea cómo se elimina el contenedor (recuerda que antes debe estar parado el contenedor).

```
docker stop servidor_web
docker rm servidor_web
docker ps -a
```

```
cliente@cliente-VirtualBox:~$ docker stop servidor_web
servidor_web
cliente@cliente-VirtualBox:~$ docker rm servidor_web
servidor_web
cliente@cliente-VirtualBox:~$ docker ps -a
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
cliente@cliente-VirtualBox:~$
```

## Ejercicio 2. Trabajo con imágenes

### Servidor web

1. Arranca un contenedor que ejecute una instancia de la imagen `php:7.4-apache`, que se llame `web` y que sea accesible desde un navegador en el puerto 8000.

```
docker run -d -it --name web -p 8000:80 php:7.4-apache
```

```
docker@docker-VirtualBox:~$ sudo docker run -d -it --name web -p 8000:80 php:7.4-apache
59d403c5a59cbfb02a50bcd2f5d59e59a965de61a2c4db5e668a177a2218901
docker@docker-VirtualBox:~$ sudo docker ps -a
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS
59d403c5a59c   php:7.4-apache "docker-php-entrypoint..." 6 seconds ago   Up 4 seconds   0.0.0.0:8000->80/tcp
```

2. Colocar en el directorio raíz del servicio web ( `/var/www/html` ) de dicho contenedor un fichero llamado `index.html` con el siguiente contenido: `HOLA SOY XXXXXXXXXXXXXXXXX`

```
docker exec -it web bash
vim index.html
```

```
docker@docker-VirtualBox:~$ sudo docker exec -it web bash
root@59d403c5a59c:/var/www/html# vim index.html
```

```
Archivo  Editar  Ver  Buscar  Terminal  Ayuda
<html>
  <head>
</head>
  <body>
    <h1>HOLA SOY ALEJANDRO CORTINA</h1>
  </body>
</html>
```



**HOLA SOY ALEJANDRO CORTINA**

3. Colocar en ese mismo directorio raíz un archivo llamado `mes.php` que muestre el nombre del mes actual. Ver la salida del script en el navegador.

```
vim mes.php
```

```
root@59d403c5a59c:/var/www/html# vim mes.php
root@59d403c5a59c:/var/www/html# exit
```

```
docker@docker-VirtualBox:~$ sudo docker exec -it web bash
Archivo  Editar  Ver  Buscar  Terminal  Ayuda
?php
$fechaact=date("F");
echo $fechaact;
```



#### 4. Borrar el contenedor

```
docker stop web
docker rm web
docker ps -a
```

```
docker@docker-VirtualBox:~$ docker stop web
web
docker@docker-VirtualBox:~$ docker rm web
web
docker@docker-VirtualBox:~$ docker ps -a
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
```

#### 5. Captura del tamaño del contenedor después de crear los archivos

```
docker@docker-VirtualBox: ~
Archivo Editar Ver Buscar Terminal Ayuda
docker@docker-VirtualBox:~$ docker info
Client:
Context: default
Debug Mode: false
Plugins:
  app: Docker App (Docker Inc., v0.9.1-beta3)
  buildx: Docker Buildx (Docker Inc., v0.7.1-docker)
  scan: Docker Scan (Docker Inc., v0.12.0)
Server:
Containers: 1
  Running: 1
  Paused: 0
  Stopped: 0
Images: 2
Server Version: 20.10.12
Storage Driver: overlay2
  Backing Filesystem: extfs
  Supports d_type: true
  Native Overlay Diff: true
  userxattr: false
Logging Driver: json-file
Cgroup Driver: cgroupfs
Cgroup Version: 1
Plugins:
  Volume: local
  Network: bridge host ipvlan macvlan null overlay
  Log: awslogs fluentd gcplogs gelf journald json-file local logentries splunk syslog
Swarm: inactive
Runtimes: io.containerd.runc.v2 io.containerd.runtime.v1.linux runc
Default Runtime: runc
Init Binary: docker-init
  containerd version: 7b11cfaabd73bb80907dd23182b9347b4245eb5d
  runc version: v1.0.2-0-g52b36a2
  init version: de40ad0
Security Options:
  apparmor
  seccomp
   Profile: default
Kernel Version: 5.4.0-84-generic
Operating System: Ubuntu 18.04.6 LTS
OSType: linux
Architecture: x86_64
CPUs: 2
Total Memory: 1.941GiB
```

## Servidor de base de datos

1. Arrancar un contenedor que se llame **bbdd** y que ejecute una instancia de la imagen **mariadb** para que sea accesible desde el puerto 3336.

La contraseña de root sea **root**

Crear una base de datos automáticamente que se llame **prueba**

Crear el usuario **invitado** con la contraseña **invitado**

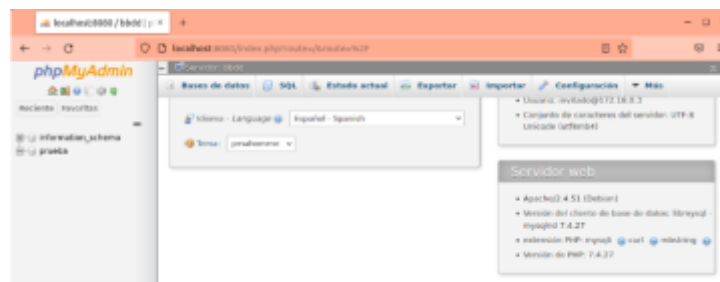
```
sudo docker run -d --name bdd -p 3306:3306 -e MARIADB_USER=invitado -e
MARIADB_PASSWORD=invitado -e MARIADB_ROOT_PASSWORD=root -e
MARIADB_DATABASES=prueba mariadb:latest
```

```
docker@docker-VirtualBox: ~
Archivo Editar Ver Buscar Terminal Ayuda
docker@docker-VirtualBox:~$ sudo docker run -d --name bdd -p 3306:3306 -e MARIADB_USER=invitado -e MARIADB_PASSWORD=invitado
-e MARIADB_ROOT_PASSWORD=root -e MARIADB_DATABASES=prueba mariadb:latest
[sudo] contraseña para docker:
Unable to find image 'mariadb:latest' locally
latest: Pulling from library/mariadb
08c01a0ec47e: Pull complete
a2bcb14c13a1: Pull complete
29c56760f879: Pull complete
a95000a218fc: Pull complete
a765d76e68d9: Pull complete
c6945738f085: Pull complete
62787b7c58c5: Pull complete
d2987a30cfe4: Pull complete
cbc65983d8b5: Pull complete
db216f91595e: Pull complete
Digest: sha256:ca31f38b6e325ece985d857db7eba1fe59928b4fd83ff8a55cb912c9684b9e43
Status: Downloaded newer image for mariadb:latest
903ed263f09d46dda821d27b0fb512a0c33eae8137a4ce14f86a32c35be801
docker@docker-VirtualBox:~$
```

## Cliente de bdd

```
sudo docker run -d -it --name conectarbdd --link bdd -p 8080:80
phpmyadmin:latest
```

```
docker@docker-VirtualBox:~$ sudo docker run -d -it --name conectarbdd --link bdd -p 8080:80 phpmyadmin:latest
Unable to find image 'phpmyadmin:latest' locally
latest: Pulling from library/phpmyadmin
5eb5b503b376: Already exists
8b1ad84cf101: Already exists
38c937d4deb7: Already exists
6a2f1dc96e59: Already exists
f8c3f82c39d4: Already exists
90fc6462bd8e: Already exists
c670d99116c9: Already exists
12c48eb41a93: Already exists
0c537b2632b2: Already exists
21a949b7ce7d: Already exists
6e7c48790426: Already exists
e1b625ec6dbf: Already exists
345e72046e34: Already exists
fe1f13e1098e: Pull complete
e52518400d12: Pull complete
8bc94ee83c2b: Pull complete
f9712993d0b2: Pull complete
0dc16e3654c6: Pull complete
Digest: sha256:48e667339e129195358b61d47be4f65a84e513663604d8d64070e3e17ab9f6c9
Status: Downloaded newer image for phpmyadmin:latest
d6dc46500cc01c7e0ad2b41d29d15c4bfe059f21e3faf7f85a2cdec464950825
docker@docker-VirtualBox:~$
```



Intentamos borrar imagen **mariadb**

```
sudo docker rmi mariadb
```

```
docker@docker-VirtualBox:~$ sudo docker rmi mariadb
Error response from daemon: conflict: unable to remove repository
reference "mariadb" (must force) - container 903ed263f09d is using
its referenced image 45a5a43e143a
docker@docker-VirtualBox:~$
```

## Ejercicio 3. Almacenamiento

1. Crea una carpeta llamada **saludo** y dentro de ella crea un fichero llamado **index.html** con el siguiente contenido (Deberás sustituir ese XXXXX por tu nombre.):

```
mkdir saludo
cd saludo
echo "<h1>HOLA SOY RUBEN</h1>" > index.html
```

```
cliente@cliente-VirtualBox:~$ mkdir saludo
cliente@cliente-VirtualBox:~$ cd saludo
```

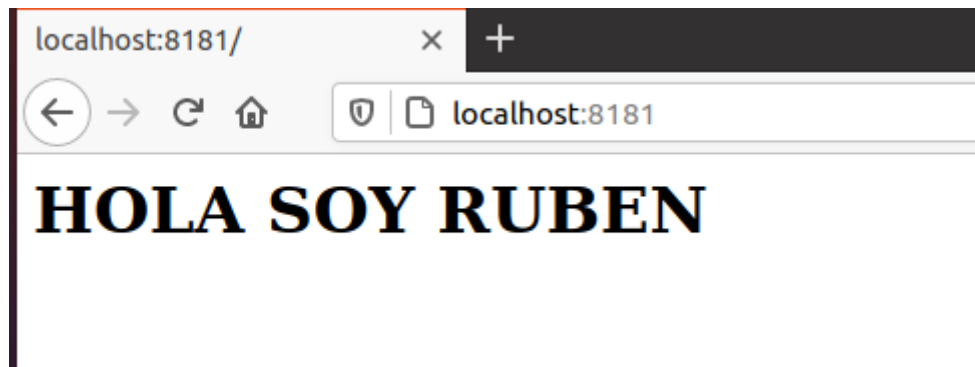
```
cliente@cliente-VirtualBox:~/saludo$ echo "<h1>HOLA SOY RUBEN</h1>" > index.htm
l
```

2. Una vez hecho esto arrancar dos contenedores basados en la imagen **php:7.4-apache** que hagan un bind mount de la carpeta **saludo** en la carpeta **/var/www/html** del contenedor. Uno de ellos vamos a acceder con el puerto 8181 y el otro con el 8282. Y su nombres serán **c1** y **c2**.

**c1**

```
docker run -d --name c1 -p 8181:80 -v /home/cliente/saludo:/var/www/html php:7.4-apache
```

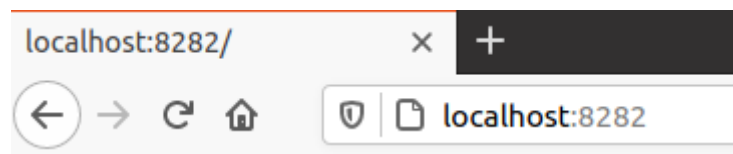
```
cliente@cliente-VirtualBox:~/saludo$ docker run -d --name c1 -p 8181:80 -v /home/cliente/saludo:/var/www/html php:7.4-apache
Unable to find image 'php:7.4-apache' locally
7.4-apache: Pulling from library/php
5eb5b503b376: Already exists
8b1ad84cf101: Pull complete
38c937dadeb7: Pull complete
6a2f1dc96e59: Pull complete
f8c3f82c39d4: Pull complete
90fc6462bd8e: Pull complete
c670d99116c9: Pull complete
12c48eb41a93: Pull complete
0c537b2632b2: Pull complete
21a949b7ce7d: Pull complete
6e7c48790426: Pull complete
e1b625ec6dbf: Pull complete
345e72046e34: Pull complete
Digest: sha256:23d69eeef4f12618dea1202e38b345591ff04dc7faedd5b2c48b1d973eb04269
Status: Downloaded newer image for php:7.4-apache
4a2835557642ab8395f1121e191b8cfcc270a786673a285107b54c770770d334
```



c2

```
docker run -d --name c2 -p 8282:80 -v /home/cliente/saludo:/var/www/html php:7.4-apache
```

```
cliente@cliente-VirtualBox:~/saludo$ docker run -d --name c2 -p 8282:80 -v /home/cliente/saludo:/var/www/html php:7.4-apache
635657ee2546ca2570b34ec61d88c209f8b089e715902689b33dd9744cea8255
```



# HOLA SOY RUBEN

3. Modifica el contenido del fichero `~/saludo/index.html` .

```
sudo nano index.html
```

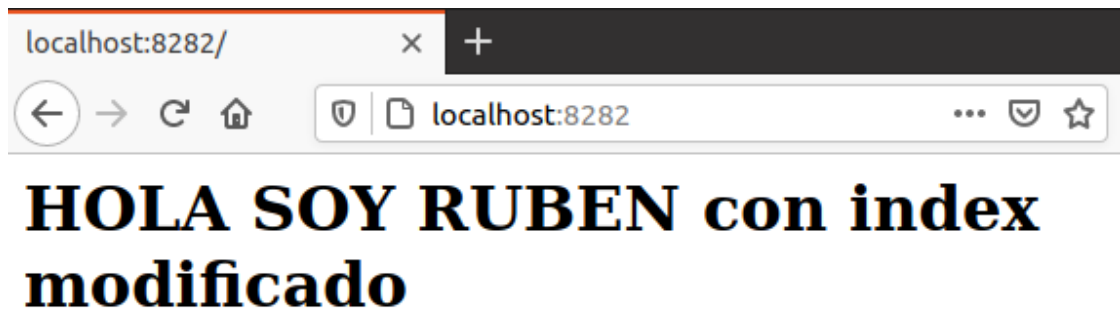
```
cliente@cliente-VirtualBox:~/saludo$ sudo nano index.html
[sudo] contraseña para cliente:
```

```
GNU nano 4.8                                index.html
<h1>HOLA SOY RUBEN con index modificado</h1>
```

4. Comprueba que puedes seguir accediendo a los contenedores, sin necesidad de reiniciarlos.

Accedemos a c1 y c2 después de modificar index.html sin reiniciar los contenedores





5. Borra los contenedores utilizados.

```
docker stop c1
docker stop c2
docker rm c1
docker rm c2
docker ps -a
```

```
cliente@cliente-VirtualBox:~$ docker stop c1
c1
cliente@cliente-VirtualBox:~$ docker stop c2
c2
cliente@cliente-VirtualBox:~$ docker rm c1
c1
cliente@cliente-VirtualBox:~$ docker rm c2
c2
```

```
cliente@cliente-VirtualBox:~$ docker ps -a
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
--------------	-------	---------	---------	--------	-------	-------

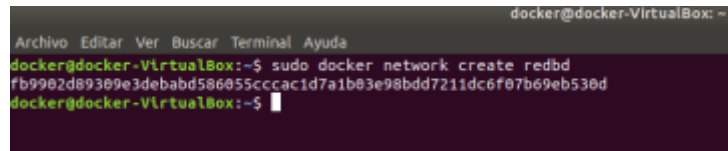


## Ejercicio 4. Redes

### Despliegue de contenedores en red: Adminer y MariaDB

1. Crea una red bridge `redbd`

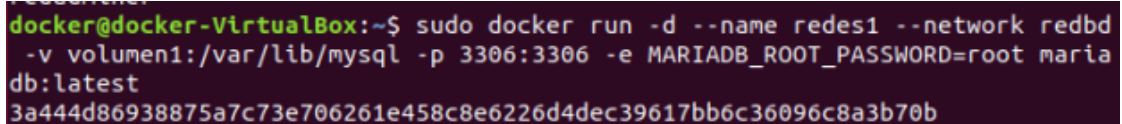
```
sudo docker network create redbd
```



```
docker@docker-VirtualBox: ~  
Archivo Editar Ver Buscar Terminal Ayuda  
docker@docker-VirtualBox:~$ sudo docker network create redbd  
fb9902d89309e3debabd586055cccac1d7a1b03e98bdd7211dc6f07b69eb530d  
docker@docker-VirtualBox:~$
```

2. Crea un contenedor con una imagen de `mariaDB` que estará en la red `redbd`. Este contenedor se ejecutará en segundo plano, y será accesible a través del puerto 3306. (Es necesario definir la contraseña del usuario `root` y un volumen de datos persistente)

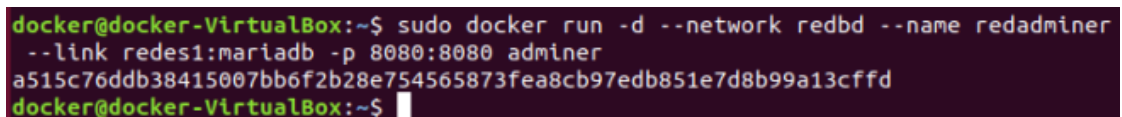
```
sudo docker run -d --name redes1 -p 3306:3306 -e MARIADB_ROOT_PASSWORD=root -v  
volumen1:/var/lib/mysql --network redbd mariadb:latest
```



```
docker@docker-VirtualBox:~$ sudo docker run -d --name redes1 --network redbd  
-v volumen1:/var/lib/mysql -p 3306:3306 -e MARIADB_ROOT_PASSWORD=root maria  
db:latest  
3a444d86938875a7c73e706261e458c8e6226d4dec39617bb6c36096c8a3b70b  
docker@docker-VirtualBox:~$
```

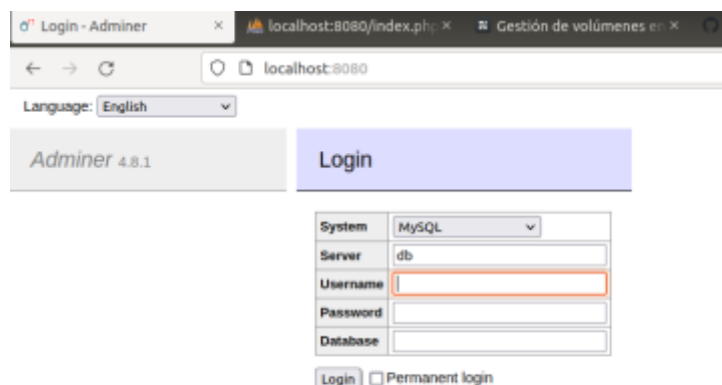
3. Crear un contenedor con Adminer que se pueda conectar al contenedor de la BD

```
sudo docker run -d --network redbd --name redadminer --link redes1:mariadb -p  
8080:8080 adminer
```

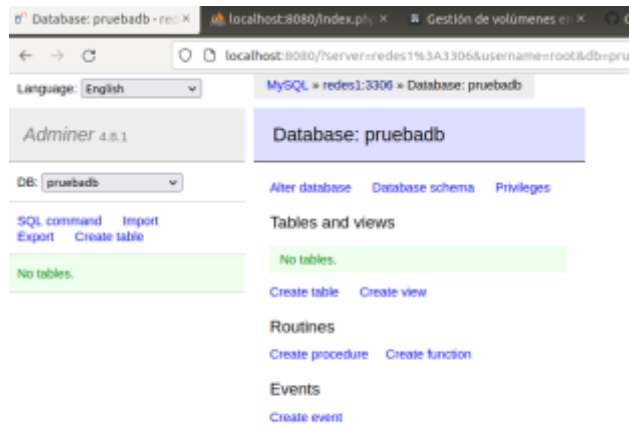


```
docker@docker-VirtualBox:~$ sudo docker run -d --network redbd --name redadminer  
--link redes1:mariadb -p 8080:8080 adminer  
a515c76ddb38415007bb6f2b28e754565873fea8cb97edb851e7d8b99a13cffd  
docker@docker-VirtualBox:~$
```

4. Comprobar que el contenedor Adminer puede conectar con el contenedor mysql abriendo un navegador web y accediendo a la URL: `http://localhost:8080`



The screenshot shows a web browser window with the Adminer login page. The URL bar shows 'localhost:8080'. The page has a 'Login' button and a form with fields for 'System' (MySQL), 'Server' (db), 'Username', 'Password', and 'Database'. There is also a 'Login' button and a 'Permanent login' checkbox.



Borramos contenedor red y sus volúmenes utilizados.

```
docker rm -v redes1
```

```
docker@doker-VirtualBox:~$ docker rm -v redes1
```

## Ejercicio 5. Crea una imagen con Dockerfile

### Crear una imagen con un servidor web que sirva un sitio web

1. Desplegar una plantilla, o un trabajo de clase, que tenga, al menos, un index.html y una carpeta para estilos, imágenes, etc.

```
mkdir apache
cd apache
echo "<h1>Prueba de funcionamiento de contenedor Docker</h1>" > index.html
```

```
mike@dockerLinux:~$ mkdir apache
mike@dockerLinux:~$ cd apache
mike@dockerLinux:~/apache$ echo "<h1>Prueba de funcionamiento de contenedor Docker</h1>">index.html
```

2. Basar la imagen en nginx o apache

Comando al que equivale el dockerfile.

```
docker build -t miguelvm75/apache2:lastest .
```

```
mike@dockerLinux:~/apache$ docker build -t miguelvm75/apache2:lastest .
```

```
GNU nano 4.8                                mike@dockerLinux: ~/apache 203x48
FROM debian                                  Dockerfile
MAINTAINER Miguel Valle Miranda "miguelvm75@educastur.es"

RUN apt-get update && apt-get install -y apache2 && apt-get clean && rm -rf /var/lib/apt/lists/*

ENV APACHE_RUN_USER www-data
ENV APACHE_RUN_GROUP www-data
ENV APACHE_LOG_DIR /var/log/apache2

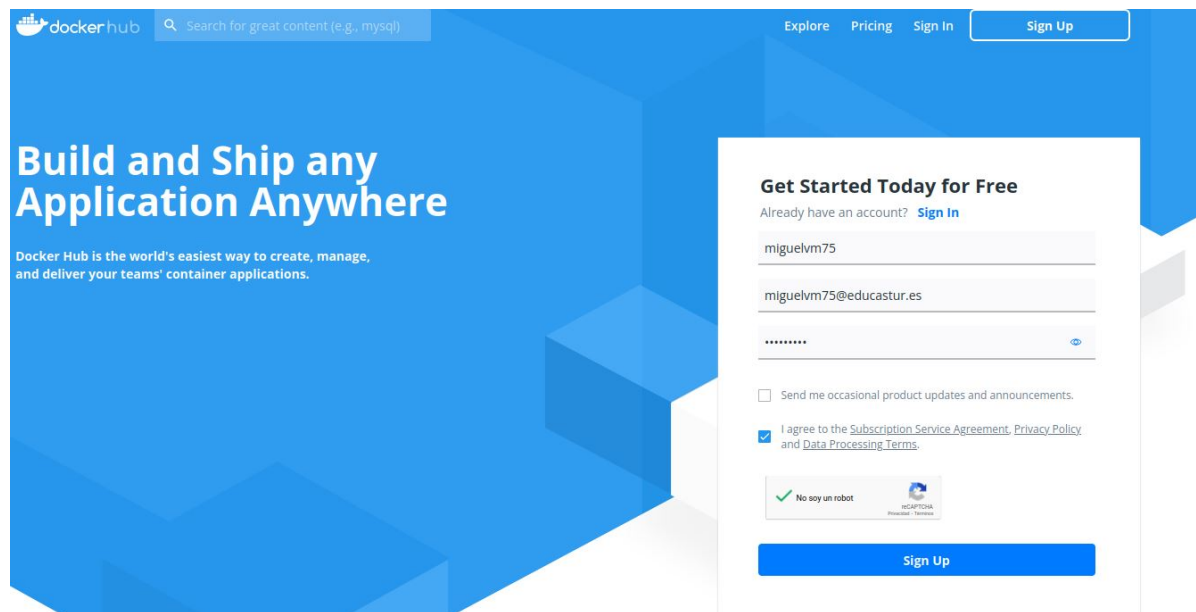
EXPOSE 80
ADD ["index.html", "/var/www/html/"]

ENTRYPOINT ["/usr/sbin/apache2ctl", "-D", "FOREGROUND"]
```

Mostramos imagen creada.

```
mike@dockerLinux:~/apache$ docker images
REPOSITORY          TAG          IMAGE ID          CREATED           SIZE
miguelvm75/apache2  latest      d53ac0671d8d     11 minutes ago   235MB
```

Creamos cuenta en Dockerhub.



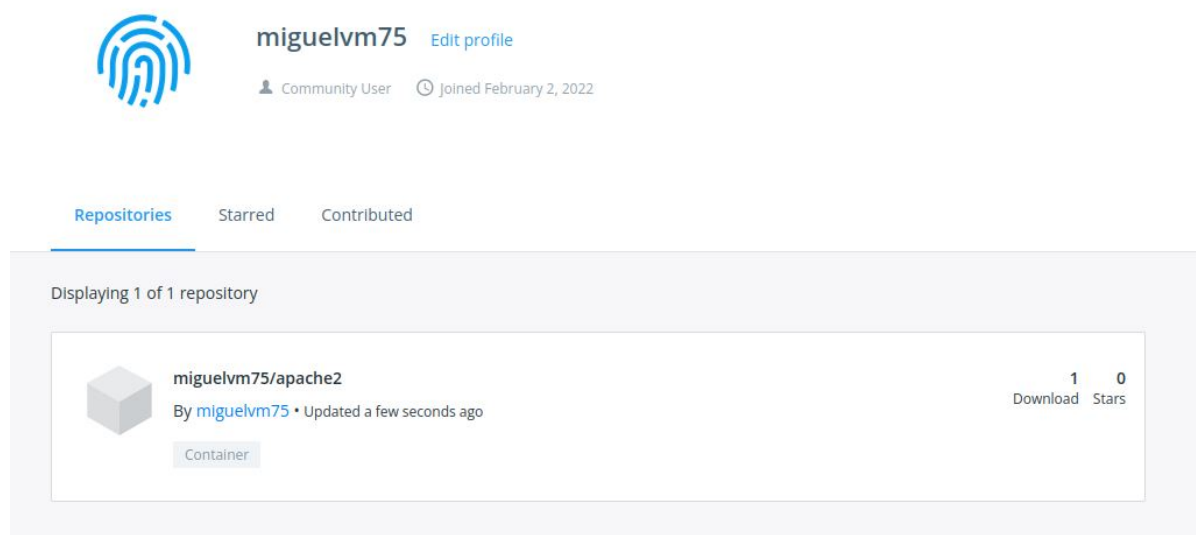
Iniciamos sesión.

```
mike@dockerLinux:~$ docker login
Login with your Docker ID to push and pull images from Docker Hub. If you don't have a Docker ID, head over to https://hub.docker.com to create one.
Username: miguelfm75
```

Subimos el repositorio a Dockerhub.

```
docker push miguelfm75/apache2:lastest

mike@dockerLinux:~$ docker push miguelfm75/apache2:lastest
The push refers to repository [docker.io/miguelfm75/apache2]
f3093709ac04: Pushed
8f31fb2fe168: Pushing [==>] 4.84MB/110.7MB
```



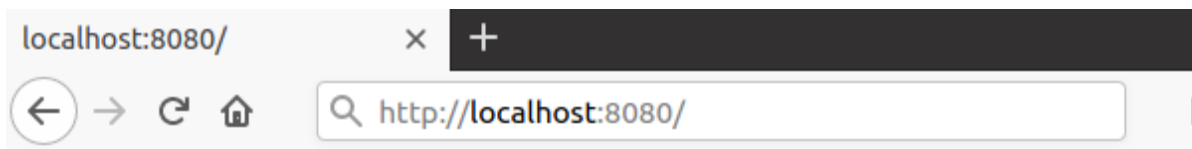
Bajada de la imagen y creación de un contenedor

```
docker pull miguelvm75/apache2:lastest
```

```
cliente@cliente-VirtualBox:~/saludo$ docker pull miguelvm75/apache2:lastest
lastest: Pulling from miguelvm75/apache2
0e29546d541c: Pull complete
39a9e64bac11: Pull complete
0d97eb421eb9: Pull complete
Digest: sha256:32720b8f1ca60092e51e82978f3ead73d2cf959da912a145f9335062e9b9e317
Status: Downloaded newer image for miguelvm75/apache2:lastest
docker.io/miguelvm75/apache2:lastest
```

```
cliente@cliente-VirtualBox:~/saludo$ docker run -d --name prueba miguelvm75/apache2:lastest
c5f0ba739cae0a7e62a4100b66690bc027bd4c9f5b12f2b5e3f25c0a23c0b307
```

Acceso al navegador con el sitio servido



## Prueba de funcionamiento de contenedor Docker