



GOBIERNO
DE ESPAÑA

MINISTERIO
DE EDUCACIÓN
Y FORMACIÓN PROFESIONAL



T3A1 - GESTIÓN BÁSICA DE CONTENEDORES DOCKER

1. PRIMEROS PASOS (1 punto)

Instala Docker engine en tu Ubuntu nativo, realiza la configuración necesaria para ejecutarlo sin ser superusuario, instala Visual studio code y su extensión de Docker y hazte una cuenta de Docker hub.

Instalaremos docker engine en una máquina Ubuntu virtual en un principio, si en un futuro tenemos problemas en la máquina virtual, crearemos una máquina Ubuntu nativa, por el momento, lo intentaremos con una máquina virtual.

Empezaremos por instalar docker engine, podemos seguir cualquiera de estas dos páginas para ello:

- <https://docs.docker.com/engine/install/ubuntu/>
- <https://www.digitalocean.com/community/tutorials/how-to-install-and-use-docker-on-ubuntu-20-04-es> (yo usaré este)

Añadiremos la clave de GPG para el repositorio oficial de Docker en el sistema

```
root@ubuntu: /home/eric
root@ubuntu:/home/eric# curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
OK
root@ubuntu:/home/eric#
```

Agregaremos el repositorio de Docker a las fuentes de APT

```
root@ubuntu: /home/eric
root@ubuntu:/home/eric# sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu focal stable"
Des:1 https://download.docker.com/linux/ubuntu focal InRelease [57,7 kB]
Obj:2 http://ppa.launchpad.net/olsf/suricata-stable/ubuntu focal InRelease
Des:3 https://download.docker.com/linux/ubuntu focal/stable amd64 Packages [40,1 kB]
Obj:4 http://us.archive.ubuntu.com/ubuntu focal InRelease
Obj:5 http://security.ubuntu.com/ubuntu focal-security InRelease
Obj:6 http://us.archive.ubuntu.com/ubuntu focal-updates InRelease
Obj:7 http://us.archive.ubuntu.com/ubuntu focal-backports InRelease
Descargados 97,8 kB en 1s (103 kB/s)
Leyendo lista de paquetes... Hecho
root@ubuntu:/home/eric#
```

Ahora ejecutamos un `sudo apt update` para actualizar los paquetes de Docker del repositorio recién agregado.

Ahora, ya podremos instalar docker engine en Ubuntu

```

root@ubuntu: /home/eric
root@ubuntu:/home/eric# sudo apt install docker-ce
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias
Leyendo la información de estado... Hecho
Se instalarán los siguientes paquetes adicionales:
  containerd.io docker-buildx-plugin docker-ce-cli docker-ce-rootless-extras docker-compose-plugin git
  git-man liberror-perl pigz slirp4netns
Paquetes sugeridos:
  aufs-tools cgroupfs-mount | cgroup-lite git-daemon-run | git-daemon-sysvinit git-doc git-el git-email
  git-gui gitk gitweb git-cvs git-mediawiki git-svn
Se instalarán los siguientes paquetes NUEVOS:
  containerd.io docker-buildx-plugin docker-ce docker-ce-cli docker-ce-rootless-extras
  docker-compose-plugin git git-man liberror-perl pigz slirp4netns
0 actualizados, 11 nuevos se instalarán, 0 para eliminar y 61 no actualizados.
Se necesita descargar 125 MB de archivos.
Se utilizarán 467 MB de espacio de disco adicional después de esta operación.
¿Desea continuar? [Y/n] Y
Des:1 https://download.docker.com/linux/ubuntu focal/stable amd64 containerd.io amd64 1.6.28-2 [29,7 MB]
Des:2 http://us.archive.ubuntu.com/ubuntu focal/universe amd64 pigz amd64 2.4-1 [57,4 kB]
Des:3 https://download.docker.com/linux/ubuntu focal/stable amd64 docker-buildx-plugin amd64 0.13.1-1-ubun
tu.20.04-focal [29,5 MB]
Des:4 http://us.archive.ubuntu.com/ubuntu focal/main amd64 liberror-perl all 0.17029-1 [26,5 kB]
Des:5 http://us.archive.ubuntu.com/ubuntu focal-updates/main amd64 git-man all 1:2.25.1-1ubuntu3.11 [887 k
B]

```

Tras instalarlo podemos ejecutar un service docker status para ver que está en función

```

root@ubuntu: /home/eric
root@ubuntu:/home/eric# service docker status
● docker.service - Docker Application Container Engine
   Loaded: loaded (/lib/systemd/system/docker.service; enabled; vendor preset: enabled)
   Active: active (running) since Sun 2024-04-07 18:59:35 PDT; 4min 54s ago
 TriggeredBy: ● docker.socket
    Docs: https://docs.docker.com
   Main PID: 33510 (dockerd)
     Tasks: 9
    Memory: 29.8M
    CGroup: /system.slice/docker.service
            └─33510 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock

abr 07 18:59:34 ubuntu systemd[1]: Starting Docker Application Container Engine...
abr 07 18:59:34 ubuntu dockerd[33510]: time="2024-04-07T18:59:34.716978803-07:00" level=info msg="Startin>
abr 07 18:59:34 ubuntu dockerd[33510]: time="2024-04-07T18:59:34.718375158-07:00" level=info msg="detecte>
abr 07 18:59:34 ubuntu dockerd[33510]: time="2024-04-07T18:59:34.867485748-07:00" level=info msg="Loading>
abr 07 18:59:35 ubuntu dockerd[33510]: time="2024-04-07T18:59:35.146974886-07:00" level=info msg="Loading>
abr 07 18:59:35 ubuntu dockerd[33510]: time="2024-04-07T18:59:35.207041721-07:00" level=info msg="Docker >
abr 07 18:59:35 ubuntu dockerd[33510]: time="2024-04-07T18:59:35.207540429-07:00" level=info msg="Daemon >
abr 07 18:59:35 ubuntu dockerd[33510]: time="2024-04-07T18:59:35.252153507-07:00" level=info msg="API lis>
abr 07 18:59:35 ubuntu systemd[1]: Started Docker Application Container Engine.
lines 1-20/20 (END)

```

Para conseguir que un usuario no root ejecute docker sin ser superusuario (sin sudo) usaremos el siguiente comando:

```

root@ubuntu: /home/eric
root@ubuntu:/home/eric# usermod -aG docker eric
root@ubuntu:/home/eric#

```

Ahora el usuario Eric podrá ejecutar docker sin necesidad de sudo

```

eric@ubuntu:~$ docker
Usage:  docker [OPTIONS] COMMAND

A self-sufficient runtime for containers

Common Commands:
run      Create and run a new container from an image
exec     Execute a command in a running container
ps       List containers
build    Build an image from a Dockerfile
pull     Download an image from a registry
push     Upload an image to a registry
images   List images
login    Log in to a registry
logout   Log out from a registry
search   Search Docker Hub for images
version  Show the Docker version information
info     Display system-wide information

Management Commands:
builder  Manage builds

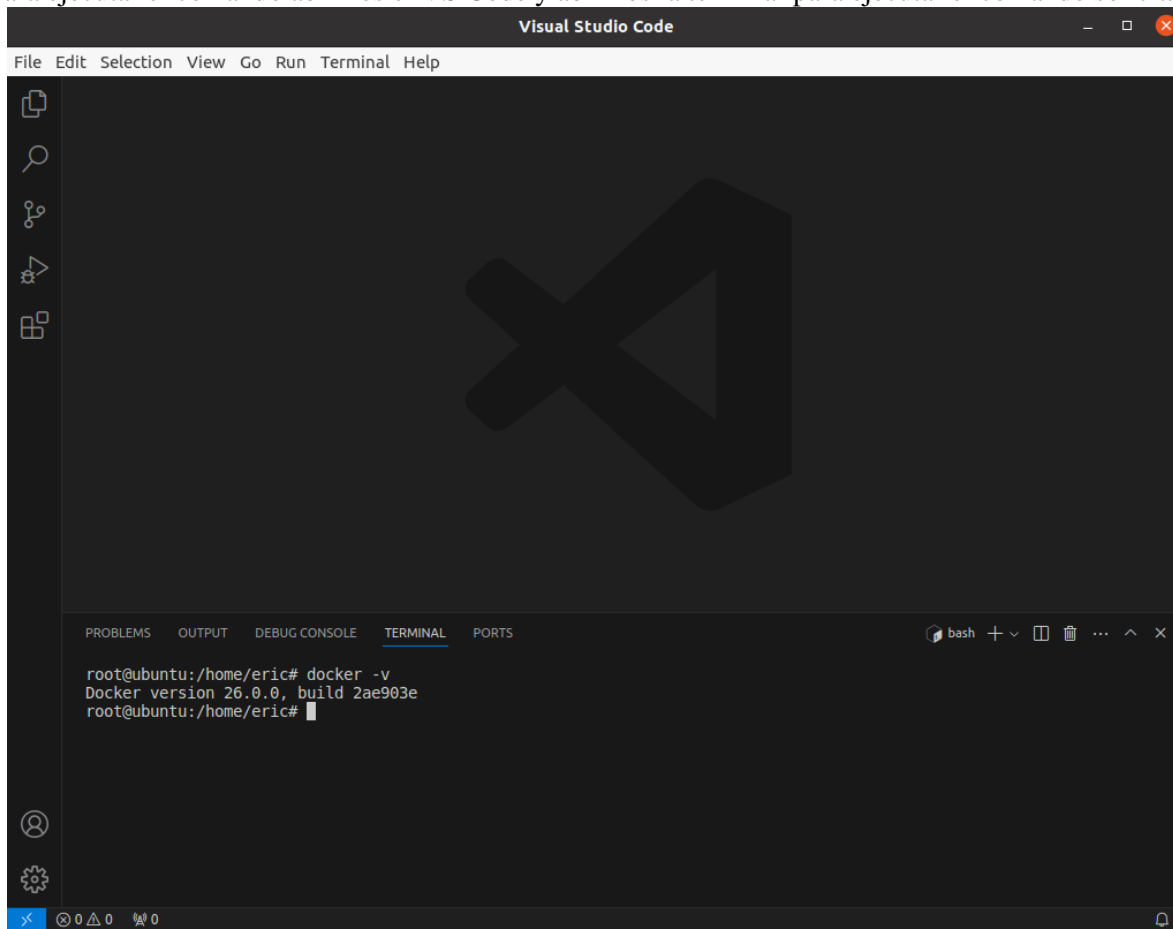
```

Aunque el usuario esté preparado para usar docker sin ser superusuario, por comodidad seguiremos con root.

Adjunta:

a) Captura de pantalla con la ejecución del comando docker -v en VS code.

Para ejecutar el comando abrimos el VS Code y abrimos la terminal para ejecutar el comando solicitado



```
Visual Studio Code
File Edit Selection View Go Run Terminal Help

root@ubuntu:/home/eric# docker -v
Docker version 26.0.0, build 2ae903e
root@ubuntu:/home/eric#
```

b) Captura de pantalla con la ejecución del contenedor:

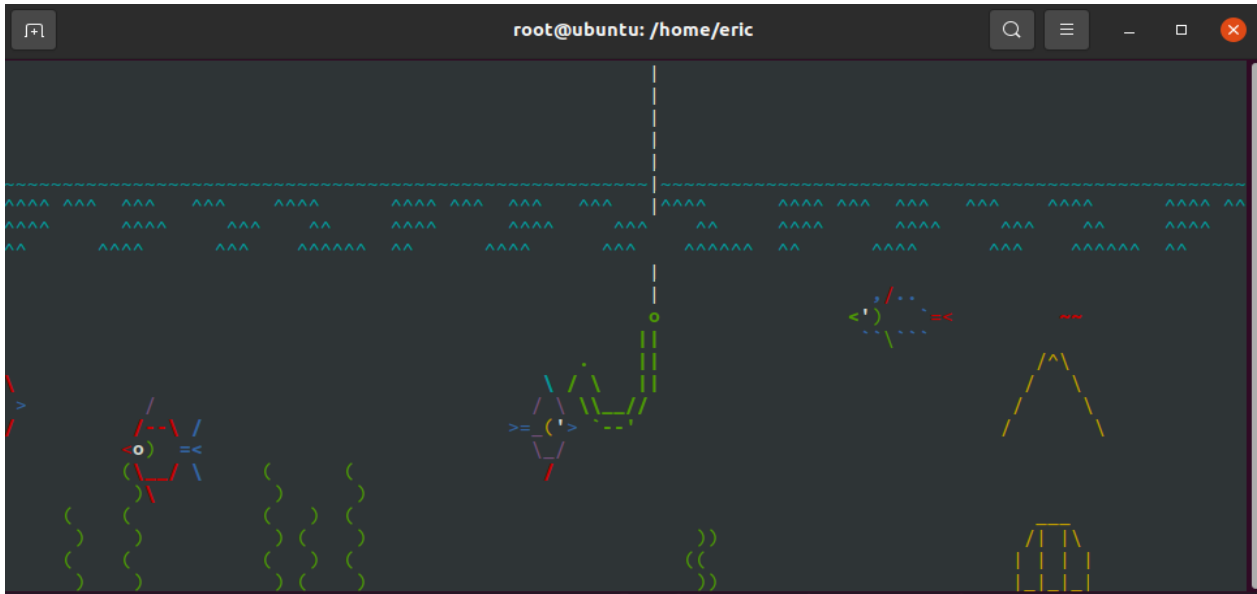
`docker run -it --rm danielkraic/asciiquarium`

Cerramos VS Code una vez probada la compatibilidad con docker, y abrimos una terminal Linux, en ella ejecutaremos el comando pedido arriba.

Como el contenedor que nos pide no lo tenemos descargado aún, el comando descargará dicho contenedor y lo ejecutará

```
root@ubuntu:/home/eric# docker run -it --rm danielkraic/asciiquarium
Unable to find image 'danielkraic/asciiquarium:latest' locally
latest: Pulling from danielkraic/asciiquarium
2ec1fbfd44b7: Pull complete
05fa48dc8a10: Pull complete
Digest: sha256:a9aaa046d674a28cd9d5091e7946c4885d27ac9ead48ce7b7ff5c12c7917fa4d
Status: Downloaded newer image for danielkraic/asciiquarium:latest
```

Una vez se descargue el contenedor, se ejecutará automáticamente.



c) Explica qué ocurre cuando lanzamos el contenedor anterior pero sin el flag “-it”

Cuando lanzamos el contenedor sin el **flag -it**, el contenedor se ejecutará en segundo plano (**modo detached**) y no se adjuntará a la terminal actual.

Esto significa que no podremos interactuar directamente con el contenedor a través de la terminal.

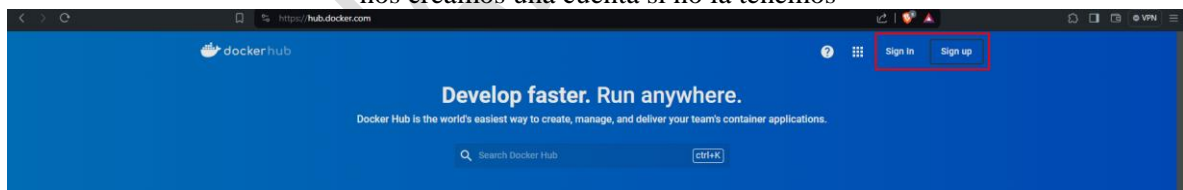
d) Explica para qué sirve el flag --rm

El **flag --rm** se utiliza para indicar a Docker que elimine automáticamente el contenedor después de que hayamos finalizado su ejecución.

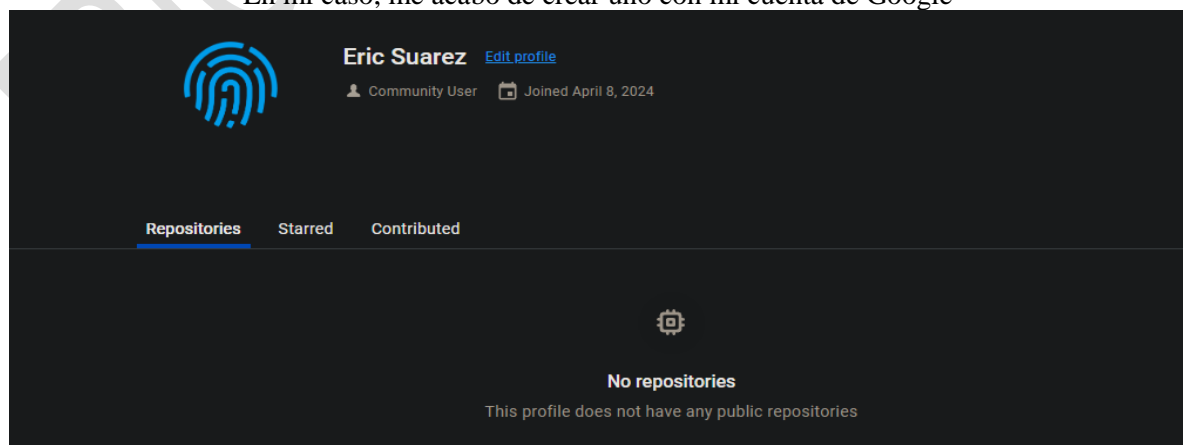
Esto es útil para limpiar **automáticamente** los contenedores que ya no son necesarios, evitando que se acumulen en el sistema.

e) Enlace de tu usuario en dockerhub

Nos dirigimos a la página web oficial de docker <https://hub.docker.com/> e iniciamos sesión o en su defecto, nos creamos una cuenta si no la tenemos



En mi caso, me acabo de crear uno con mi cuenta de Google



Este es el enlace: <https://hub.docker.com/u/ericsuarezv>

2. DESCARGANDO Y LANZANDO CONTENEDORES (4 puntos)

Descargar las siguientes imágenes:

- ubuntu:18.04
- centos:8
- debian:9
- httpd
- php:7.3-apache
- mariadb:latest

Para descargar todas estas imágenes, ejecutaremos el comando **docker pull + la imagen correspondiente**, de esta manera:

```
root@ubuntu:/home/eric# docker pull ubuntu:18.04
18.04: Pulling from library/ubuntu
7c457f213c76: Pull complete
Digest: sha256:152dc042452c496007f07ca9127571cb9c29697f42acbfad72324b2bb2e43c98
Status: Downloaded newer image for ubuntu:18.04
docker.io/library/ubuntu:18.04
root@ubuntu:/home/eric#
```

Y lo haremos igual para todas las demás máquinas.

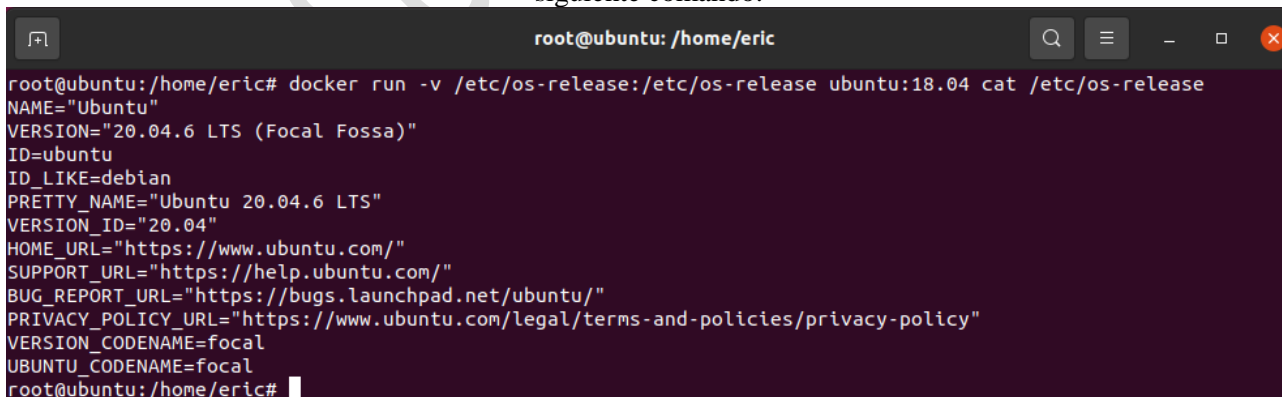
- a) Adjunta captura de pantalla con la ejecución de la orden correspondiente de **docker** que las lista.

Para ver el listado de los contenedores descargados usaremos el comando **docker images**

```
root@ubuntu:/home/eric# docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
httpd                latest             147ddc9b1d39       3 days ago         174MB
mariadb              latest             bc6434c28e9a       6 weeks ago        405MB
ubuntu               18.04              f9a80a55f492       10 months ago      63.2MB
debian               9                  662c05203bab       21 months ago      101MB
php                  7.3-apache         35da9118b3c0       2 years ago         451MB
centos                8                  5d0da3dc9764       2 years ago         231MB
danielkraic/asciiq  latest             7bab964067d2       5 years ago         309MB
root@ubuntu:/home/eric#
```

- b) Lanza la imagen de ubuntu y sin entrar en el contenedor, mostrar por pantalla el fichero **/etc/os-release** y capturar un pantallazo del resultado.

Para lanzar la imagen sin entrar al contenedor y mostrar por pantalla el fichero seleccionado ejecutaremos el siguiente comando:



```
root@ubuntu:/home/eric# docker run -v /etc/os-release:/etc/os-release ubuntu:18.04 cat /etc/os-release
NAME="Ubuntu"
VERSION="20.04.6 LTS (Focal Fossa)"
ID=ubuntu
ID_LIKE=debian
PRETTY_NAME="Ubuntu 20.04.6 LTS"
VERSION_ID="20.04"
HOME_URL="https://www.ubuntu.com/"
SUPPORT_URL="https://help.ubuntu.com/"
BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
VERSION_CODENAME=focal
UBUNTU_CODENAME=focal
root@ubuntu:/home/eric#
```

Este comando ejecutará un contenedor basado en la imagen de Ubuntu 18.04 y mostrará el archivo **/etc/os-release**

c) **Lanza la imagen httpd entrando de manera interactiva en el contenedor poniéndole un nombre y adjunta captura de pantalla.**

Para lanzar la imagen **httpd** interactivamente y asignarle un nombre, podemos utilizar el siguiente comando

```
root@ubuntu: /home/eric
root@ubuntu:/home/eric# docker run -it --name mi_httpd httpd /bin/bash
root@376e8b56d801:/usr/local/apache2# ls
bin build cgi-bin conf error htdocs icons include logs modules
root@376e8b56d801:/usr/local/apache2#
```

Con esto, ya estaremos dentro del contenedor y podremos navegar por él.

d) **¿Qué versión de sistema operativo hay en el contenedor? ¿Qué ocurre cuando intentamos ejecutar el comando “nano”? Explica a qué es debido.**

Para conocer el sistema operativo del contenedor ejecutamos el anterior comando mostrado: **cat /etc/os-release** dentro del contenedor

```
root@ubuntu: /home/eric
root@376e8b56d801:/usr/local/apache2# cat /etc/os-release
PRETTY_NAME="Debian GNU/Linux 12 (bookworm)"
NAME="Debian GNU/Linux"
VERSION_ID="12"
VERSION="12 (bookworm)"
VERSION_CODENAME=bookworm
ID=debian
HOME_URL="https://www.debian.org/"
SUPPORT_URL="https://www.debian.org/support"
BUG_REPORT_URL="https://bugs.debian.org/"
root@376e8b56d801:/usr/local/apache2#
```

Cuando intentamos ejecutar el comando **nano** dentro del contenedor, obtenemos un mensaje de error que indica que el comando no está disponible.

Esto se debe a que el contenedor está basado en la imagen **httpd**, que es una imagen de Apache HTTP Server, estas imágenes suelen ser versiones mínimas de Linux diseñadas específicamente para ejecutar el servicio web y no incluyen herramientas de edición de texto como **nano**.

```
root@ubuntu: /home/eric
root@376e8b56d801:/usr/local/apache2# nano
bash: nano: command not found
root@376e8b56d801:/usr/local/apache2#
```

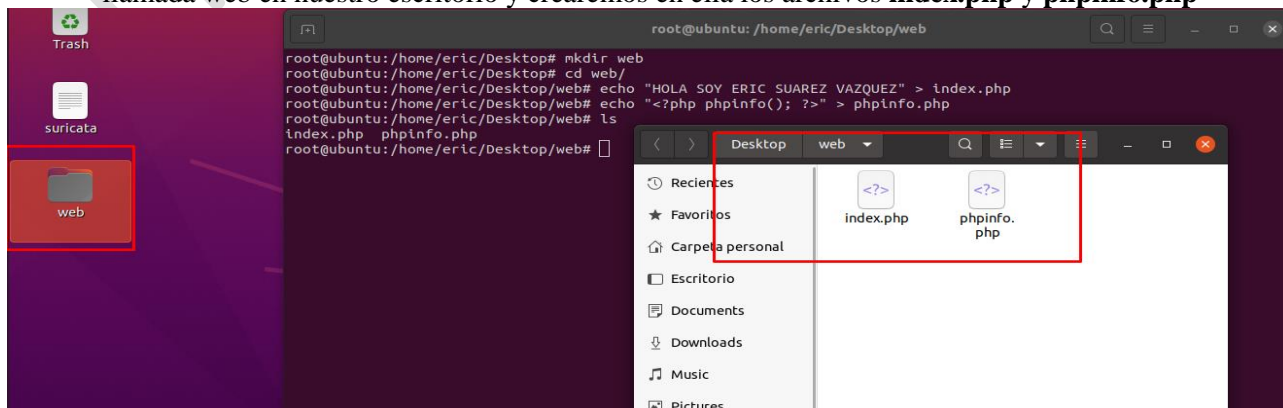
3. INFO SOBRE IMÁGENES Y CONTENEDORES (4 puntos)

- ☐ Arranca un contenedor que ejecute una instancia de la imagen **php:7.3-apache**, que se llame **web** y que sea accesible desde tu equipo en el puerto 8181.

Colocar en el directorio raíz del servicio web de dicho contenedor un fichero llamado **index.php** con el siguiente contenido:

HOLA SOY NOMBRE+APELLIDOS

Empezamos desde aquí creando aquí creando la estructura de directorios, para ello crearemos una carpeta llamada **web** en nuestro escritorio y crearemos en ella los archivos **index.php** y **phpinfo.php**



Visualiza la web desde el anfitrión.

Tras crear todo la estructura de directorios y los archivos, ejecutamos el siguiente comando para iniciar la imagen docker en la carpeta destino por el puerto 8181 e iremos a la url localhost:8181 para ver el **index.php**



```

root@ubuntu: /home/eric/Desktop
root@ubuntu:/home/eric/Desktop# docker run -d --name web -p 8181:80 -v "$(pwd)/web":/var/www/html php:7.3-apache
b80c96e1b655c9b01eba12889e658f74d67b4f6615b9aefe6877b7f601c69e8a
root@ubuntu:/home/eric/Desktop#
    
```

Colocar en ese mismo directorio raíz un archivo llamado **index.php** con el siguiente contenido:

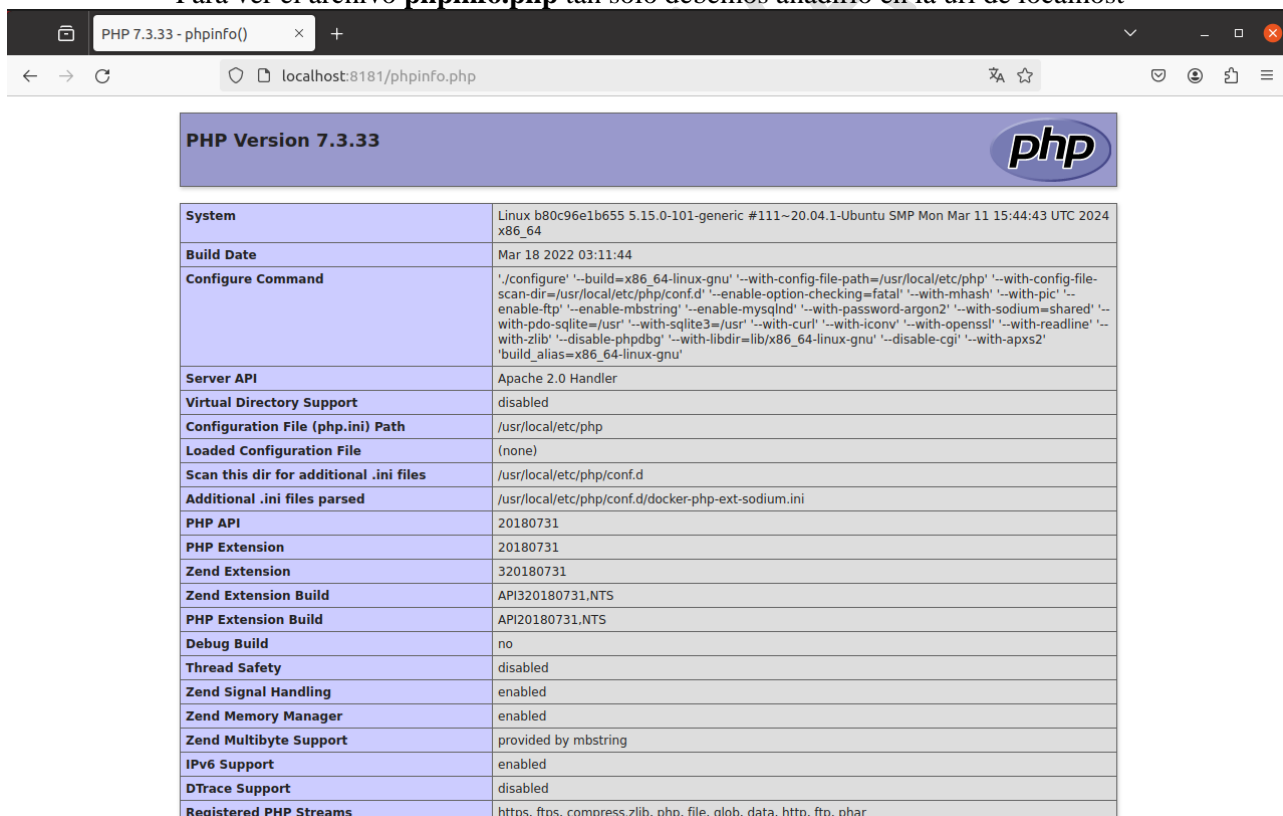
```

<?php
phpinfo();
?>
    
```

Atención: esto ya lo hemos hecho antes y, para no perder la información escrita en **index.php**, hemos creado otro archivo.php diferente que es **phpinfo.php**,

Visualiza nuevamente la web desde el navegador del anfitrión.

Para ver el archivo **phpinfo.php** tan solo debemos añadirlo en la url de localhost



PHP Version 7.3.33	
System	Linux b80c96e1b655 5.15.0-101-generic #111~20.04.1-Ubuntu SMP Mon Mar 11 15:44:43 UTC 2024 x86_64
Build Date	Mar 18 2022 03:11:44
Configure Command	'./configure' '--build=x86_64-linux-gnu' '--with-config-file-path=/usr/local/etc/php' '--with-config-file-scan-dir=/usr/local/etc/php/conf.d' '--enable-option-checking=fatal' '--with-mhash' '--with-pic' '--enable-ftp' '--enable-mbstring' '--enable-mysqlnd' '--with-password-argon2' '--with-sodium=shared' '--with-pdo-sqlite=/usr' '--with-sqlite3=/usr' '--with-curl' '--with-iconv' '--with-openssl' '--with-readline' '--with-zlib' '--disable-phpdbg' '--with-libdir=lib/x86_64-linux-gnu' '--disable-cgi' '--with-apxs2' 'build_alias=x86_64-linux-gnu'
Server API	Apache 2.0 Handler
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/usr/local/etc/php
Loaded Configuration File	(none)
Scan this dir for additional .ini files	/usr/local/etc/php/conf.d
Additional .ini files parsed	/usr/local/etc/php/conf.d/docker-php-ext-sodium.ini
PHP API	20180731
PHP Extension	20180731
Zend Extension	320180731
Zend Extension Build	API320180731.NTS
PHP Extension Build	API20180731.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, ftps, compress.zlib, php, file, glob, data, http, ftp, phar

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

Antes de arrancarlo visitar la página del contenedor en Docker Hub (https://hub.docker.com/_/mariadb) y establecer las variables de entorno necesarias para que:

- ☐ La contraseña de root sea root.
- ☐ Crear el usuario “invitado” con la contraseña “invitado”.

Para arrancar un contenedor que se llame **bbdd** con la imagen de mariadb y por el puerto 3336, con las características root y usuario explicadas arriba, debemos ejecutar el siguiente comando

```
root@ubuntu:/home/eric/Desktop# docker run -d --name bbdd -p 3336:3306 -e MYSQL_ROOT_PASSWORD=root -e MYSQL_USER=invitado -e MYSQL_PASSWORD=invitado mariadb
93b071c401492fd24f859d355ab84306bbd7c01874ba243b131b68574977231e
root@ubuntu:/home/eric/Desktop#
```

Comprueba que la configuración es correcta entrando en la base de datos de la forma que prefieras. Conexión con el servidor en Mariadb [aquí](#).

Comprobamos que la configuración es correcta entrando a la base de datos mediante la conexión con **mysql**

```
root@ubuntu:/home/eric/Desktop# mysql -u invitado -p -h 127.0.0.1 -P 3336
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 4
Server version: 11.3.2-MariaDB-1:11.3.2+maria-ubu2204 mariadb.org binary distribution

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]>
```

(Debemos tener mariadb-client o mysql-client instalado)

- ☐ Ejecutando la orden adecuada, obtén la siguiente información:
 - ☐ Dirección IP del contenedor **web** y captura de acceso desde el anfitrión

Obtenemos la dirección IP del contenedor **web** con el siguiente comando

```
root@ubuntu:/home/eric/Desktop# docker inspect -f '{{range .NetworkSettings.Networks}}{{.IPAddress}}{{end}}' web
172.17.0.2
root@ubuntu:/home/eric/Desktop#
```

- ☐ Redirección de puertos del contenedor **web**

Obtenemos la dirección del puerto del contenedor **web** con el siguiente comando

```
root@ubuntu:/home/eric/Desktop# docker port web
80/tcp -> 0.0.0.0:8181
80/tcp -> [::]:8181
root@ubuntu:/home/eric/Desktop#
```

- ☐ Dirección IP del contenedor **bbdd** y captura de acceso desde el anfitrión

☐ Obtenemos la dirección IP del contenedor **bbdd** con el siguiente comando

```
root@ubuntu:/home/eric/Desktop# docker inspect -f '{{range .NetworkSettings.Networks}}{{.IPAddress}}{{end}}' bbdd
172.17.0.3
root@ubuntu:/home/eric/Desktop#
```


Requiere instalar un cliente mysql remoto. Como por ejemplo [mycli](#)

Ya tenemos la dirección IP necesaria para obtener el acceso del anfitrión pero primero debemos instalar la herramienta **mycli**

```
root@ubuntu: /home/eric/Desktop
root@ubuntu:/home/eric/Desktop# pip install mycli
Collecting mycli
  Downloading mycli-1.27.2-py2.py3-none-any.whl (68 kB)
    | 68 kB 4.4 MB/s
Collecting sqlglot>=5.1.3
  Downloading sqlglot-23.7.0-py3-none-any.whl (383 kB)
    | 383 kB 12.2 MB/s
Collecting pyaes>=1.6.1
  Downloading pyaes-1.6.1.tar.gz (28 kB)
Collecting configobj>=5.0.5
  Downloading configobj-5.0.8-py2.py3-none-any.whl (36 kB)
Requirement already satisfied: cryptography>=1.0.0 in /usr/lib/python3/dist-packages (from mycli) (2.8)
Collecting cli_helpers[styles]>=2.2.1
  Downloading cli_helpers-2.3.1-py3-none-any.whl (19 kB)
Requirement already satisfied: click>=7.0 in /usr/lib/python3/dist-packages (from mycli) (7.0)
Collecting pyperclip>=1.8.1
  Downloading pyperclip-1.8.2.tar.gz (20 kB)
Collecting PyMySQL>=0.9.2
  Downloading PyMySQL-1.1.0-py3-none-any.whl (44 kB)
    | 44 kB 7.0 MB/s
Collecting sqlparse<0.5.0,>=0.3.0
  Downloading sqlparse-0.4.4-py3-none-any.whl (41 kB)
```

Una vez termine la descarga tendremos todo listo para obtener el acceso del anfitrión con el siguiente comando de **mycli**

```
root@ubuntu:/home/eric/Desktop# mycli -h 172.17.0.3 -P 3336 -u invitado -p invitado
```

Como no hemos hecho nada en la base de datos al entrar, al poner el comando solo nos muestra los datos de la base de datos y el acceso del anfitrión

❑ Redirección de puertos del contenedor **bbdd**

Obtenemos la dirección del puerto del contenedor **bbdd** con el siguiente comando

```
root@ubuntu:/home/eric/Desktop
root@ubuntu:/home/eric/Desktop# docker port bdd
3306/tcp -> 0.0.0.0:3336
3306/tcp -> [::]:3336
root@ubuntu:/home/eric/Desktop#
```

4. LIMPIEZA (1 punto)

Adjunta captura de pantalla de las siguientes operaciones:

a) lista de contenedores activos e inactivos e imágenes

Para obtener una lista de contenedores activos e inactivos disponibles, ejecutaremos el siguiente comando

```
root@ubuntu:/home/eric/Desktop
root@ubuntu:/home/eric/Desktop# docker ps -a
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
93b071c40149	mariadb	"docker-entrypoint.s..."	About an hour ago	Up About an hour	0.0.0.0:3336->3306/tcp, :::3336->3306/tcp	bbdd
b80c96e1b655	php:7.3-apache	"docker-php-entrypoi..."	2 hours ago	Up 2 hours	0.0.0.0:8181->80/tcp, :::8181->80/tcp	web
376e8b56d801	httpd	"/bin/bash"	2 hours ago	Exited (255) 2 hours ago	80/tcp	mi_httpd
96e171d03a19	ubuntu:18.04	"cat /etc/os-release"	2 hours ago	Exited (0) 2 hours ago		nystifying_zhukovsky

A continuación ejecutaremos este otro comando para listar las imágenes de docker que tengamos

```
root@ubuntu:/home/eric/Desktop# docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
httpd	latest	147ddc9b1d39	3 days ago	174MB
mariadb	latest	bc6434c28e9a	6 weeks ago	405MB
ubuntu	18.04	f9a80a55f492	10 months ago	63.2MB
debian	9	662c05203bab	21 months ago	101MB
php	7.3-apache	35da9118b3c0	2 years ago	451MB
centos	8	5d0da3dc9764	2 years ago	231MB
danielkraic/asciiaquarium	latest	7bab964067d2	5 years ago	309MB

b) borra con un solo comando las imágenes de las que no dependan ningún contenedor

Podemos utilizar el siguiente comando para eliminar todas las imágenes que no estén siendo utilizadas por ningún contenedor, y lo mostramos con el comando anterior

```
root@ubuntu:/home/eric/Desktop# docker image prune -a
WARNING! This will remove all images without at least one container associated to them.
Are you sure you want to continue? [y/N] Y
Deleted Images:
untagged: centos:8
untagged: centos@sha256:a27fd8080b517143cbbbab9dfb7c8571c40d67d534bbdee55bd6c473f432b177
deleted: sha256:5d0da3dc976460b72c77d94c8a1ad043720b0416bfc16c52c45d4847e53fad6
deleted: sha256:74ddd0ec08fa43d09f32636ba91a0a3053b02cb4627c35051aff89f853606b59
untagged: debian:9
untagged: debian@sha256:c5c5200ff1e9c73ffbf188b4a67eb1c91531b644856b4aefe86a58d2f0cb05be
deleted: sha256:662c05203bab4c59568d24689fa5c3955439360a35c483178598d226b9a5ad10
deleted: sha256:47dde6f41d1fcca77df950cac35c7103996a9333244a7406248c9b6898215469
untagged: danielkraic/asciiquarium:latest
untagged: danielkraic/asciiquarium@sha256:a9aaa046d674a28cd9d5091e7946c4885d27ac9ead48ce7b7ff5c12c7917fa4d
deleted: sha256:7bab964067d2af9f5438194a7184de2e2edbc94b0e567466625004f67870d2cc
deleted: sha256:a5b5b0bc65280271ddac7d7ff3071ea2caa9225b6608dd6d4e3e9bc6018df166
deleted: sha256:97b94d418dac4ec6da11f188c162831ddf674b08c496d1fd9ebaf5d497f94e08

Total reclaimed space: 640.9MB
root@ubuntu:/home/eric/Desktop# docker images
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
httpd          latest    147ddc9b1d39   3 days ago    174MB
mariadb        latest    bc6434c28e9a   6 weeks ago   405MB
ubuntu         18.04     f9a80a55f492   10 months ago 63.2MB
php            7.3-apache 35da9118b3c0   2 years ago   451MB
root@ubuntu:/home/eric/Desktop#
```

c) borra todos los contenedores y todas las imágenes

Para borrar todos los contenedores y todas las imágenes, ejecutaremos los siguientes comandos

- Borrar contenedores

```
root@ubuntu:/home/eric/Desktop# docker rm -f $(docker ps -aq)
93b071c40149
b80c96e1b655
376e8b56d801
96e171d03a19
```

- Borrar imágenes

```
root@ubuntu:/home/eric/Desktop# docker rmi -f $(docker images -aq)
Untagged: httpd:latest
Untagged: httpd@sha256:8a1bbe23f2733589625dedde0fb9687419d02cf2ec542c5bd411798b4b47ada3
Deleted: sha256:147ddc9b1d39c13d6ca2c0260958cdddc964c6076d9e1e0d5d60caf2f2a0c31
Deleted: sha256:1dfc2bda2b118df6cf6732d5116ea9d3c6c3a18adb95dc115e3c59ac79fcb0d2
Deleted: sha256:119da33aba778ccbda3af1a3e95df4afb35d0552ba75d3127cdb5a6a8d144435
Deleted: sha256:ee1cf6338fbd00e32428c8610860490eea4ce4407a37ce25dbbcd01789713ec7
Deleted: sha256:304a1d544ec8439199cbbda34c9012f09e48efa1a8627e38bfff88cd29dac78be
Deleted: sha256:ab60a1b2df537aee1390e40f0ed38acb83c2a7e84aacf3dfb22a373c5cd46013
Deleted: sha256:a483da8ab3e941547542718cacd3258c6c705a63e94183c837c9bc44eb608999
Untagged: mariadb:latest
Untagged: mariadb@sha256:b5e508abc5d889425e90212541e30d29279b7ed34dd74bab5bb715b2f2aeeb7b
Deleted: sha256:bc6434c28e9a9ca8559a04481c70873457ed32555abcf73002136b608f08738d
Deleted: sha256:eb4bc0576128a42b4deb94c27e3402caea4a8d23a9bbace69062b6baefd8166c
Deleted: sha256:a41676253ddf5639a70f0c6784a15b5bb3b30b6c3a951c93614706e7c8e279d0
Deleted: sha256:ae8f3798e18019fd2c5e3f00b3fafcdd8747ac336232637dfefbcfd6af97bfca
Deleted: sha256:934e32e3c780f53245ababf8b9f932625614391df2af620bef96aa4bbf55abbc
Deleted: sha256:1caabc41bbd4f1a65d150e0f1225b0012e3355d5741a71b088e6d900ce95edc4
Deleted: sha256:8e44cf75df2b748d0813e8a97635c16c604e42e62a6df09bf622a67201bee5eb
Deleted: sha256:bdd1a6100fb28db33ba5e9187006c6dbda2db93c75e4b3e02b4887104004fb4
Deleted: sha256:5498e8c22f6996f25ef193ee58617d5b37e2a96decf22e72de13c3b34e147591
Untagged: ubuntu:18.04
Untagged: ubuntu@sha256:152dc042452c496007f07ca9127571cb9c29697f42acbfa72324b2bb2e43c98
Deleted: sha256:f9a80a55f492e823bf5d51f1bd5f87ea3eed1cb31788686aa99a2fb61a27af6a
Deleted: sha256:548a79621a426b4eb077c926eabac5a8620c454fb230640253e1b44dc7dd7562
Untagged: php:7.3-apache
Untagged: php@sha256:b9872cd287ef72bc17d45d713aa2742f3d3bcf2503fea2506fd93aa94995219f
Deleted: sha256:35da9118b3c0980fbaf84a4d410653642c9d55fb2f6c40011cc78de303837d64
Deleted: sha256:30189ef121b167a2f357c086c2fe7066360aee308ef40acce334deb72066a9f
Deleted: sha256:f43e4641413d6e45d6e2c02bd58f94018228edc2deb26a95de77232d0f29e429
Deleted: sha256:8280d7b6d895dc07cb8601673f05d96951995c76b5e258c97b2deee30dca6121
Deleted: sha256:7eba25ddc4e69fb5ba9d7f7104989ab3680ff323c1373fd49de90c869ad51115
Deleted: sha256:5f7c985fe7348d9824bb6db19fe570c9590d33c235b8065140a49c1e2e8746b6
Deleted: sha256:1c8d008a05ca951d28ea4530bf0cd8f246aa9301237e4080e6b44cbc71d0bd63
Deleted: sha256:34f8af19ffbea1f8e352455f43b8bcd3e3790f0d80e2562b714e4fe2e5848993
Deleted: sha256:1ed7805d18d6f65d73b776a46c223733f1931f51eaf70b36ccd7f50f9c290938
Deleted: sha256:73f0954bc88b5540a775c8381e848fdc3823e9f665eb2244e571de53d716de3d
Deleted: sha256:86a7dcd3813e9f09ddade57bce78717fd49df1665f8342887e07f50db19ffffbd
Deleted: sha256:2c609a6dfc4e2acb8c8a508661a32fb258487077bc5168b3bf84a4bc64768d17
Deleted: sha256:40872cc04308a7b31ed7ea9c7a0edfc417f5cd81d13c35dbd48e53297f28bc11
Deleted: sha256:03ae2846e2f2b5c439e92a930db2f43a51c0def8b78f7aea66449790a041dea6
Deleted: sha256:3a626bb08c24b5cc968d312bf5694aa87b6d9961c5f182c6bc138d8ca8ac13ee
root@ubuntu:/home/eric/Desktop# docker images
REPOSITORY TAG IMAGE ID CREATED SIZE
root@ubuntu:/home/eric/Desktop#
```