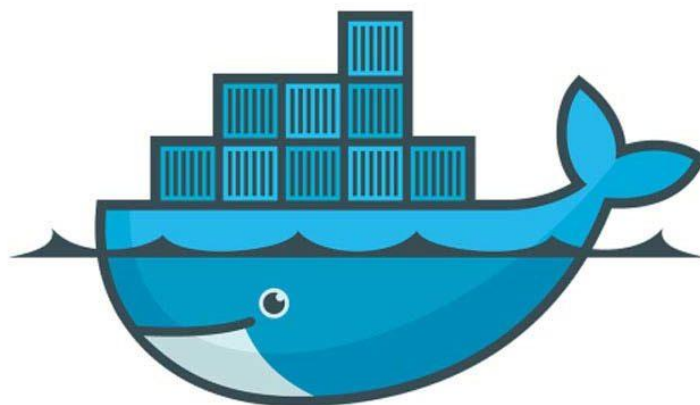




DOCKER

Sistemas Informáticos (1º DAM)



[FECHA]

[NOMBRE DE LA COMPAÑÍA]

[Dirección de la compañía]

Índice

Contenido

1. ACTIVIDAD 1: DESCARGAR IMÁGENES Y OPERACIONES BÁSICAS	1
1.1. DESCARGAR IMÁGENES DE DOCKER HUB.	1
1.2. MOSTRAR IMÁGENES	3
1.3. CONTENEDOR CON UBUNTU 18.04, NOMBRE "UBUNTU"	4
1.3.1. PARAR CONTENEDOR	4
1.3.2. REARRANCAR CONTENEDOR	4
1.3.3. MOSTRAR FICHERO SIN ENTRAR AL CONTENEDOR	5
2. ACTIVIDAD 2: TRABAJAR CON PHP Y MARIADB	6
2.1. CONTENEDOR CON PHP, NOMBRE "WEB" Y PUERTO 8181	6
2.2. COLOCAR FICHEROS INDEX.HTML E INDEX.PHP EN EL DIRECTORIO WEB DEL CONTENEDOR	6
2.3. CONTENEDOR CON MARIADB CON DIFERENTES PARÁMETROS	7
2.4. CONECTAR CON UN CLIENTE DE BASE DE DATOS	7
3. ACTIVIDAD 3: GESTIÓN DE IMÁGENES	9
3.1. DESCARGAR IMAGEN UBUNTU:20.04	9
3.2. VOLCAR LA INFORMACIÓN DE LA NUEVA IMAGEN EN UN ARCHIVO	9
3.3. CREAR CONTENEDOR CON UBUNTU:20.04 LLAMADA "MODULO3"	10
3.4. BORRAR CONTENEDOR CON UBUNTU:20.04	10
3.5. HACER ACCIONES NECESARIAS PARA BORRAR EL CONTENEDOR UBUNTU:20.04	10
4. ACTIVIDAD 4: OPERACIONES CON VOLÚMENES	11
4.1. CREAR VOLÚMENES	11
4.2. INSTANCIAR CONTENEDORES EN LOS VOLÚMENES	11
4.3. PARAR Y BORRAR CONTENEDOR "C2" Y VOLUMEN "VOLUMEN_DATOS"	12
4.4. COMPROBAR QUE "C1" ESTÁ MONTADO SOBRE "VOLUMEN_WEB"	12

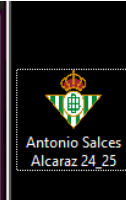
1. Actividad 1: descargar imágenes y operaciones básicas

1.1. Descargar imágenes de Docker Hub.

Para descargar los dockers, debemos de utilizar el comando ***“docker pull <imagen[:versión]>”***. Por defecto, cogerá la versión “latest” si no ponemos versión.

Ubuntu:18.04

```
usuario@usuario:~$ sudo docker pull ubuntu:18.04
[sudo] contraseña para usuario:
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
usuario@usuario:~$ S
```



Centos:8

```
usuario@usuario:~$ sudo docker pull centos:8
8: Pulling from library/centos
a1d0c7532777: Pull complete
Digest: sha256:a27fd8080b517143cbbab9dfb7c8571c40d67d534bbdee55bd6c473f432b177
Status: Downloaded newer image for centos:8
docker.io/library/centos:8
usuario@usuario:~$ S
```



Debian: 9

```
usuario@usuario:~$ sudo docker pull debian:9
9: Pulling from library/debian
8372a04f222b: Pull complete
Digest: sha256:c5c5200ff1e9c73ffbf188b4a67eb1c91531b644856b4aefe86a58d2f0cb05be
Status: Downloaded newer image for debian:9
docker.io/library/debian:9
```



MariaDB:latest

```
usuario@usuario:~$ sudo docker pull mariadb
Using default tag: latest
latest: Pulling from library/mariadb
de44b265507a: Pull complete
ca9b21d0c985: Pull complete
041c753879ae: Pull complete
e7b5137dc4b2: Pull complete
655e5d2590bd: Pull complete
41b3170b5f12: Pull complete
95adc28016bc: Pull complete
407e9d6eefb4: Pull complete
Digest: sha256:a9547599cd87d7242435aea6fda22a9d83e2c06d16c658ef70d2868b3d3f6a80
Status: Downloaded newer image for mariadb:latest
docker.io/library/mariadb:latest
usuario@usuario:~$ S
```



MySQL:5.7

```
usuario@usuario:~$ sudo docker pull mysql:5.7
5.7: Pulling from library/mysql
20e4dcae4c69: Pull complete
1c56c3d4ce74: Pull complete
e9f03a1c24ce: Pull complete
68c3898c2015: Pull complete
6b95a940e7b6: Pull complete
90986bb8de6e: Pull complete
ae71319cb779: Pull complete
ffc89e9dfd88: Pull complete
43d05e938198: Pull complete
064b2d298fba: Pull complete
df9a4d85569b: Pull complete
Digest: sha256:4bc6bc963e6d8443453676cae56536f4b8156d78bae03c0145cbe47c2aad73bb
Status: Downloaded newer image for mysql:5.7
docker.io/library/mysql:5.7
usuario@usuario:~$ S
```

httpd

```
usuario@usuario:~$ sudo docker pull httpd
Using default tag: latest
latest: Pulling from library/httpd
bc0965b23a04: Pull complete
d7ad38c6dd97: Pull complete
4f4fb700ef54: Pull complete
79b49624e34b: Pull complete
7d9f97915db2: Pull complete
9bd25d4f7b77: Pull complete
Digest: sha256:f4c5139eda466e45814122d9bd8b886d8ef6877296126c09b76dbad72b03c336
Status: Downloaded newer image for httpd:latest
docker.io/library/httpd:latest
```

tomcat:9.0.39-jdk11

```
usuario@usuario:~$ sudo docker pull tomcat:9.0.39-jdk11
9.0.39-jdk11: Pulling from library/tomcat
e4c3d3e4f7b0: Pull complete
101c41d0463b: Pull complete
8275efcd805f: Pull complete
751620502a7a: Pull complete
a59da3a7d0e7: Pull complete
9c0f1dffe039: Pull complete
576e3c6f47f8: Pull complete
c7e1b6c3ef84: Pull complete
d1b8a428acdc: Pull complete
7251ae448a6d: Pull complete
Digest: sha256:5b17d5de9c75c9da638c28186c19423b610e7eab3b6f6b975bf47383d12ed0a9
Status: Downloaded newer image for tomcat:9.0.39-jdk11
docker.io/library/tomcat:9.0.39-jdk11
usuario@usuario:~$
```



jenkins/jenkins:lts

```

usuario@usuario:~$ sudo docker pull jenkins/jenkins:lts
[sudo] contraseña para usuario:
lts: Pulling from jenkins/jenkins
b2b31b28ee3c: Pull complete
768595d27f0b: Pull complete
2902ddfaf8af: Pull complete
1944ded7dbca: Pull complete
37b0412849e4: Pull complete
9e6f96481dc6: Pull complete
8d5cd706e369: Pull complete
e1d3077f0c0c: Pull complete
66714a60a07a: Pull complete
e37c8a6a1d29: Pull complete
0867b45f78b4: Pull complete
d0238388e632: Pull complete
Digest: sha256:e728082cd6a2710840ef7d9fdcdc93408eb488aa05d10bc92f4454254e22cc4e
Status: Downloaded newer image for jenkins/jenkins:lts
docker.io/jenkins/jenkins:lts
usuario@usuario:~$

```



Antonio Salces Alcaraz 24_25

php:7.3-apache

```

usuario@usuario:~$ sudo docker pull php:7.3-apache
7.3-apache: Pulling from library/php
ae13dd578326: Pull complete
f15d475049bf: Pull complete
886e5161983f: Pull complete
aa7666573a25: Pull complete
59357a0f9863: Pull complete
dc3ffb8c774e: Pull complete
513e9383f6d4: Pull complete
5ebd0737aa08: Pull complete
255df6c25392: Pull complete
f2994be86066: Pull complete
746eb0cc36a4: Pull complete
e46201569d4f: Pull complete
e05ec73939b3: Pull complete
a1488be2aff6: Pull complete
Digest: sha256:b9872cd287ef72bc17d45d713aa2742f3d3bcf2503fea2506fd93aa94995219f
Status: Downloaded newer image for php:7.3-apache
docker.io/library/php:7.3-apache
usuario@usuario:~$

```



Antonio Salces Alcaraz 24_25

1.2. Mostrar imágenes

Para hacer esto, debemos de hacer uso del comando ***"docker images"***

```

usuario@usuario:~$ sudo docker images

```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
jenkins/jenkins	lts	e699ac6c40dc	3 weeks ago	467MB
mariadb	latest	6722945a6940	4 weeks ago	407MB
httpd	latest	494b2b45fd74	5 months ago	147MB
mysql	5.7	5107333e08a8	12 months ago	501MB
ubuntu	18.04	f9a80a55f492	19 months ago	63.2MB
debian	9	662c05203bab	2 years ago	101MB
php	7.3-apache	35da9118b3c0	2 years ago	451MB
centos	8	5d0da3dc9764	3 years ago	231MB
tomcat	9.0.39-jdk11	2703bbe9e9d4	4 years ago	648MB

```

usuario@usuario:~$

```



Antonio Salces Alcaraz 24_25

1.3. Contenedor con Ubuntu 18.04, nombre "ubuntu"

Para arrancar el contenedor y darle un nombre, utilizaremos el comando **"docker run [-dit] [--name <nombre>] <imagen:versión> [comando]"**. También utilizaremos el comando **"docker ps -a"** para ver todos los contenedores del sistema y así poder saber el ID del docker de ubuntu.

```

usuario@usuario:~$ sudo docker run -dit --name ubuntu ubuntu:18.04 /bin/bash
0ee7f08d0498e914c9e26f407efc740862b4853d24c0115bc6f2cb2fba4f960c
usuario@usuario:~$ docker ps -a
permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock: Get "http://%2Fvar%2Frun%2Fdocker.sock/v1.47/containers/json?all=1": dial unix /var/run/docker.sock: connect: permission denied
usuario@usuario:~$ sudo docker ps -a
CONTAINER ID   IMAGE     COMMAND                  CREATED          STATUS          PORTS          NAMES
0ee7f08d0498   ubuntu:18.04   "/bin/bash"             24 seconds ago   Up 22 seconds   -             ubuntu
usuario@usuario:~$

```

1.3.1. Parar contenedor

Para parar el contenedor, ejecutaremos el comando **"docker stop <ID>"**. Basta con poner el inicio del ID, no es necesario ponerlo entero. Para comprobar que está parado, basta con poner **"docker ps -a"** y comprobar el estado, donde se ve que está parado.

```

usuario@usuario:~$ sudo docker stop 0e
0e
usuario@usuario:~$ docker ps -a
permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock: Get "http://%2Fvar%2Frun%2Fdocker.sock/v1.47/containers/json?all=1": dial unix /var/run/docker.sock: connect: permission denied
usuario@usuario:~$ sudo docker ps -a
CONTAINER ID   IMAGE     COMMAND                  CREATED          STATUS          PORTS          NAMES
0ee7f08d0498   ubuntu:18.04   "/bin/bash"             8 minutes ago    Exited (0) 3 minutes ago   -             ubuntu
usuario@usuario:~$

```

1.3.2. Rearrancar contenedor

Para rearrancar el contenedor, utilizaremos **"docker start <ID>"**, y para comprobar si se ha iniciado, **"docker ps -a"**.

```

root@usuario:/home/usuario# docker start 0e
0e
root@usuario:/home/usuario# docker ps -a
CONTAINER ID   IMAGE     COMMAND                  CREATED          STATUS          PORTS          NAMES
0ee7f08d0498   ubuntu:18.04   "/bin/bash"             13 minutes ago   Up 5 seconds    -             ubuntu
root@usuario:/home/usuario#

```

1.3.3. Mostrar fichero sin entrar al contenedor

Para ello, debemos utilizar el comando ***"docker exec <nombre> cat /etc/os-release"***. Docker exec nos permite ejecutar comandos del contenedor desde fuera del mismo, mientras que "cat" se utiliza para visualizar archivos. En lugar del nombre, también podemos poner el ID.

```
root@usuario:/home/usuario# docker exec ubuntu cat /etc/os-release
NAME="Ubuntu"
VERSION="18.04.6 LTS (Bionic Beaver)"
ID=ubuntu
ID_LIKE=debian
PRETTY_NAME="Ubuntu 18.04.6 LTS"
VERSION_ID="18.04"
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=bionic
UBUNTU_CODENAME=bionic
root@usuario:/home/usuario# S
```



2. Actividad 2: trabajar con PHP y MariaDB

2.1. Contenedor con PHP, nombre “web” y puerto 8181

Para iniciar el contenedor, utilizaremos el comando “**docker run [-d] [--name <nombre>] [-p <puerto>] <imagen>**”. Con “-d” ejecutaremos el docker en segundo plano, y con “-p” podremos elegir el puerto para el docker, en este caso el 8181.

```
root@usuario:/home/usuario# docker run -d --name web -p 8181:80 php:7.3-apache
e869274605951551dd83d2e5d8f7e1aa5c254d61e62f42f676e7af5925e6eb4d
root@usuario:/home/usuario# docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS                               NAMES
e86927460595   php:7.3-apache  "docker-php-entrypoi..." 12 seconds ago Up 10 seconds 0.0.0.0:8181->80/tcp, [::]:8181->80/tcp   web
```

2.2. Colocar ficheros index.html e index.php en el directorio web del contenedor

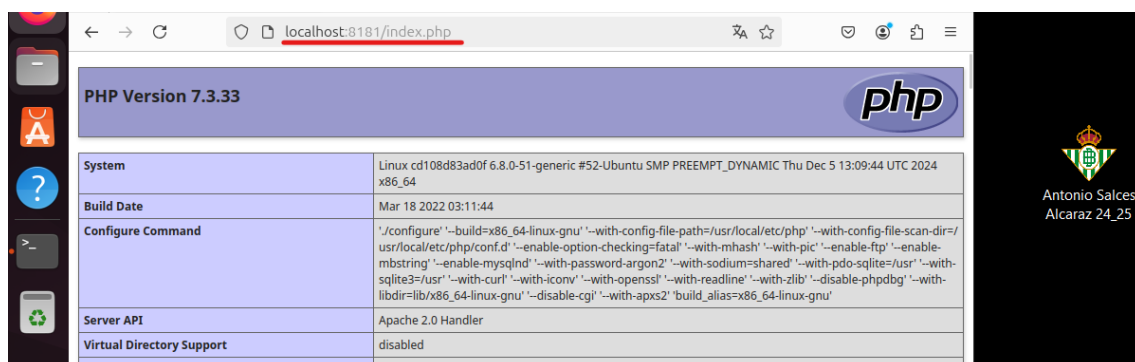
En primer lugar, accederemos al directorio web del contenedor con el primer comando que se ve en la imagen (“**docker exec -it [nombre o ID] bash**”). Una vez estemos en el contenedor, utilizaremos “**echo <contenido del archivo> > <ubicación del archivo>**”. Para comprobar el contenido del archivo, podemos utilizar “**cat <nombre>**”. Haremos lo mismo con el fichero “index.php”.

```
root@usuario:/home/usuario# docker exec -it web bash
root@e86927460595:/var/www/html# echo "<h1>HOLA SOY ANTONIO SALCES ALCARAZ</h1>" > /var/www/html/index.html
root@e86927460595:/var/www/html# cat index.html
<h1>HOLA SOY ANTONIO SALCES ALCARAZ</h1>
root@e86927460595:/var/www/html# echo "<?php phpinfo(); ?>" > /var/www/html/index.php
root@e86927460595:/var/www/html# cat index.php
<?php phpinfo(); ?>
root@e86927460595:/var/www/html#
```

Si buscamos la siguiente dirección en la maquina virtual, podremos ver el fichero index.html que hemos creado.



Y esto saldrá si buscamos el fichero index.php.



2.3. Contenedor con mariadb con diferentes parámetros

Debemos arrancar un contenedor que contenga “mariadb” con los siguientes parámetros:

- Nombre: bddd.
- Puerto: 3336.
- Contraseña de root: root.
- Crear base de datos con nombre: prueba.
- Contraseña del usuario: usuario.

```

root@usuario:/home/usuario# docker run --name bddd -e MYSQL_ROOT_PASSWORD=root -e MYSQL_DATABASE=prueba -e MYSQL_USER=invitado -e MYSQL_PASSWORD=invitado -p 3336:3306 -d mariadb
69f98e31a59a9d13b02c0cc148f469a83157fc31d13ba399c1e7b944f69baa6
root@usuario:/home/usuario# docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED        STATUS        PORTS                               NAMES
69f98e31a59a   mariadb   "docker-entrypoint.s..." 3 seconds ago  Up 2 seconds  0.0.0.0:3336->3306/tcp, :::3336->3306/tcp  bddd
cd108d83ad0f   php:7.3-apache  "docker-php-entrypoi..." 11 minutes ago Up 11 minutes  0.0.0.0:8181->80/tcp, :::8181->80/tcp    web

```

Los parámetros que se han utilizado son los siguientes:

- --name bddd: el nombre del docker es “bddd”.
- -e MYSQL_ROOT_PASSWORD=root: la contraseña del root será “root”.
- -e MYSQL_DATABASE=prueba: crea la base de datos llamada “prueba”.
- -e MYSQL_USER=invitado: crea el usuario “invitado”.
- -e MYSQL_PASSWORD=invitado: la contraseña del usuario “invitado” será “invitado”.
- -p 3336:3306: indica el puerto a utilizar.
- -d mariadb: lanzaremos “mariadb” en segundo plano.

2.4. Conectar con un cliente de base de datos

En primer lugar, debemos de instalar un cliente de base de datos. En mi caso, utilizaré un cliente de terminal (mysql-client). Utilizaremos el siguiente comando para instalarlo: “**apt install <paquete>**”.

```

root@usuario:/home/usuario# apt install mysql-client-core-8.0
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias... Hecho
Leyendo la información de estado... Hecho
Se instalarán los siguientes paquetes NUEVOS:
  mysql-client-core-8.0
0 actualizados, 1 nuevos se instalarán, 0 para eliminar y 4 no actualizados.
Se necesita descargar 2.765 kB de archivos.
Se utilizarán 61,6 MB de espacio de disco adicional después de esta operación.
Des:1 http://es.archive.ubuntu.com/ubuntu noble-updates/main amd64 mysql-client-core-8.0 amd64 8.0.40-0ubuntu0.24.04.1 [2.765 kB]
Descargados 2.765 kB en 2s (1.578 kB/s)
Seleccionando el paquete mysql-client-core-8.0 previamente no seleccionado.
(Leyendo la base de datos ... 150724 ficheros o directorios instalados actualmente.)
Preparando para desempaquetar .../mysql-client-core-8.0_8.0.40-0ubuntu0.24.04.1_amd64.deb ...
Desempaquetando mysql-client-core-8.0 (8.0.40-0ubuntu0.24.04.1) ...
Configurando mysql-client-core-8.0 (8.0.40-0ubuntu0.24.04.1) ...
Procesando disparadores para man-db (2.12.0-4build2) ...
root@usuario:/home/usuario#

```

Con el primer comando conseguiremos conectarnos con la base de datos del contenedor. El comando es el siguiente: “**mysql <-h IP> <-P puerto> <-u usuario> [-p]**”. “-p” indica que tenemos que introducir una contraseña para entrar.

Una vez hayamos conectado con la base de datos, podemos utilizar el comando “**SHOW DATABASES;**” para ver las bases de datos creadas. Podemos ver en la imagen que la base de datos “prueba” se ha creado.

```
root@usuario:/home/usuario# mysql -h 127.0.0.1 -P 3336 -u invitado -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 5
Server version: 11.6.2-MariaDB-ubu2404 mariadb.org binary distribution

Copyright (c) 2000, 2024, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

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

mysql> SHOW DATABASES;
+-----+
| Database |
+-----+
| information_schema |
| prueba |
+-----+
2 rows in set (0,01 sec)

mysql> 
```



3. Actividad 3: gestión de imágenes

3.1. Descargar imagen Ubuntu:20.04

Para descargar la imagen, utilizaremos el comando ***"docker pull ubuntu:20.04"***, y una vez se haya completado la descarga, para comprobar que está en nuestro sistema, utilizaremos el comando ***"docker images"***.

```
root@usuario:/home/usuario# docker pull ubuntu:20.04
20.04: Pulling from library/ubuntu
d9802f032d67: Pull complete
Digest: sha256:8e5c4f0285ecbb4ead070431d29b576a530d3166df73ec44affc1cd27555141b
Status: Downloaded newer image for ubuntu:20.04
docker.io/library/ubuntu:20.04
root@usuario:/home/usuario# docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
jenkins/jenkins	lts	e699ac6c40dc	5 weeks ago	467MB
mariadb	latest	6722945a6940	5 weeks ago	407MB
ubuntu	20.04	6013ae1a63c2	2 months ago	72.8MB
httpd	latest	4ce47c750a58	5 months ago	147MB
mysql	5.7	5107333e08a8	12 months ago	501MB
ubuntu	18.04	f9a80a55f492	19 months ago	63.2MB
debian	9	662c05203bab	2 years ago	101MB
php	7.3-apache	35da9118b3c0	2 years ago	451MB
centos	8	5d0da3dc9764	3 years ago	231MB
tomcat	9.0.39-jdk11	2703bbe9e9d4	4 years ago	648MB

```
root@usuario:/home/usuario#
```


Antonio Salces
Alcaraz 24_25

3.2. Volcar la información de la nueva imagen en un archivo

Para ver la información de la nueva imagen y volcarla en un archivo, utilizaremos el comando ***"docker inspect <imagen> > <ruta>"***. Para ver la información guardada en el archivo, utilizamos el comando ***"cat <ruta>"***.

```
root@usuario:/home/usuario# docker inspect ubuntu:20.04 > /home/usuario/Escritorio/info.txt
root@usuario:/home/usuario# cat /home/usuario/Escritorio/info.txt
```

```
[
  {
    "Id": "sha256:6013ae1a63c2ee58a8949f03c6366a3ef6a2f386a7db27d86de2de965e9f450b",
    "RepoTags": [
      "ubuntu:20.04"
    ],
    "RepoDigests": [
      "ubuntu@sha256:8e5c4f0285ecbb4ead070431d29b576a530d3166df73ec44affc1cd27555141b"
    ],
    "Parent": "",
    "Comment": "",
    "Created": "2024-10-11T03:38:27.357079367Z",
    "DockerVersion": "24.0.7",
    "Author": "",
    "Config": {
      "Hostname": "",
      "Domainname": "",
      "User": "",
      "AttachStdin": false,
      "AttachStdout": false,
      "AttachStderr": false,
```


Antonio Salces
Alcaraz 24_25

3.3. Crear contenedor con Ubuntu:20.04 llamada "modulo3"

Utilizaremos el comando **"docker run -d --name modulo3 ubuntu:20.04"** para crear el contenedor con los parámetros especificados. Con el comando **"docker ps -a"** veremos el contenedor.

```
root@usuario:/home/usuario# docker run -d --name modulo3 ubuntu:20.04
1b5a179cb2d8de09fb18c201312ec7dc92f083177ac41e31cb62f1cfc1edf6b6
root@usuario:/home/usuario# docker ps -a
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
1b5a179cb2d8	ubuntu:20.04	"/bin/bash"	5 seconds ago	Exited (0) 3 seconds ago		modulo3
69f98e31a59a	mariadb	"docker-entrypoint.s..."	3 days ago	Exited (0) 2 days ago		bbdd
cd108d83ad0f	php:7.3-apache	"docker-php-entrypoi..."	3 days ago	Exited (0) 3 days ago		web

```
root@usuario:/home/usuario#
```



3.4. Borrar contenedor con Ubuntu:20.04

Para borrar una imagen, utilizaremos el comando **"docker rmi <imagen>"**. Como podemos ver, no podemos borrar la imagen. Esto es debido a que la imagen se está utilizando en un contenedor, por lo que no podremos borrarla mientras el contenedor exista.

```
root@usuario:/home/usuario# docker rmi ubuntu:20.04
Error response from daemon: conflict: unable to remove repository reference "ubuntu:20.04" (must force) - container 1b5a179cb2d8 is using its referenced image 6013ae1a63c2
root@usuario:/home/usuario#
```



3.5. Hacer acciones necesarias para borrar el contenedor Ubuntu:20.04

Para borrar la imagen, en primer lugar debemos parar el contenedor con **"docker stop <contenedor>"**, luego borrar con **"docker rm <contenedor>"**, y por ultimo borrar la imagen con **"docker rmi <imagen>"**.

```
root@usuario:/home/usuario# docker stop modulo3
modulo3
root@usuario:/home/usuario# docker rm modulo3
modulo3
root@usuario:/home/usuario# docker rmi ubuntu:20.04
Untagged: ubuntu:20.04
Untagged: ubuntu@sha256:8e5c4f0285ecbb4ead070431d29b576a530d3166df73ec44affc1cd27555141b
Deleted: sha256:6013ae1a63c2ee58a8949f03c6366a3ef6a2f386a7db27d86de2de965e9f450b
Deleted: sha256:fffe76c64ef2dee2d80a8bb3ad13d65d596d04a45510b1956a976a69215dae92
root@usuario:/home/usuario#
```



4. Actividad 4: operaciones con volúmenes

4.1. Crear volúmenes

Con el comando **"docker volume create <nombre>"** podremos crear los volúmenes con el nombre que queramos. Luego, utilizaremos **"docker volume ls"** para ver los volúmenes creados.

```
root@usuario:/home/usuario# docker volume create volumen_datos
volumen_datos
root@usuario:/home/usuario# docker volume create volumen_web
volumen_web
root@usuario:/home/usuario# docker volume ls
DRIVER      VOLUME NAME
local       e6893b55a515fce92167af832f8e9fe5ca29cb3911b67d258916613f6732f019
local       volumen_datos
local       volumen_web
root@usuario:/home/usuario#
```



4.2. Instanciar contenedores en los volúmenes

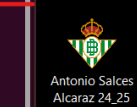
Con el comando **"docker run [-d] [--name <nombre>] [-v <volumen>:][ruta] <imagen>"** podremos crear un contenedor y elegir en que volumen y que ruta queremos asignarle. Con **"docker ps"** veremos que el contenedor está activo

```
root@usuario:/home/usuario# docker run -d --name c1 -v volumen_web:/var/www/html php:7.4-apache
4fcb4f9fca57a0fa960701bd6405a1a17efb3e3be00ac8319eb777fb7bcd3920
root@usuario:/home/usuario# docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED        STATUS        PORTS        NAMES
4fcb4f9fca57   php:7.4-apache   "docker-php-entrypoi..."   9 seconds ago   Up 7 seconds   80/tcp       c1
root@usuario:/home/usuario#
```



Con el comando **"docker run [-d] [--name <nombre>] >] [-v <volumen>:][ruta] [-e<MYSQL_ROOT_PASSWORD>=<contraseña>] <imagen>"** podremos crear un contenedor, elegir el volumen y la ruta donde se aloja y, en este caso (se trata de MariaDB) podemos asignar también una contraseña al usuario "root".

```
root@usuario:/home/usuario# docker run -d --name c2 -v volumen_datos:/var/lib/mysql -e MYSQL_ROOT_PASSWORD=admin mariadb
fea2b1eb1d73e6e259b0295058b05b4f7af654c00d07e711676af81e869a4192
root@usuario:/home/usuario# docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED        STATUS        PORTS        NAMES
fea2b1eb1d73   mariadb   "docker-entrypoint.s..."   15 seconds ago   Up 14 seconds   3306/tcp     c2
4fcb4f9fca57   php:7.4-apache   "docker-php-entrypoi..."   4 minutes ago   Up 4 minutes   80/tcp       c1
root@usuario:/home/usuario#
```



4.3. Parar y borrar contenedor “c2” y volumen “volumen_datos”

En primer lugar, paramos el contenedor “c2” con “**docker stop <nombre o ID>**”, luego, lo borramos con “**docker rm <nombre o ID>**”, y a continuación borramos el volumen con “**docker volume rm <nombre>**”. Para acabar, comprobaremos que se han eliminado con “**docker ps**” y “**docker volume ls**” y que ya solo queda el contenedor “c1”.

```
root@usuario:/home/usuario# docker stop c2
c2
root@usuario:/home/usuario# docker rm c2
c2
root@usuario:/home/usuario# docker volume rm volumen_datos
volumen_datos
root@usuario:/home/usuario# docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS        NAMES
4fcb4f9fca57   php:7.4-apache "docker-php-entrypoi..." 14 minutes ago Up 14 minutes 80/tcp       c1
root@usuario:/home/usuario# docker volumen ls
docker: 'volumen' is not a docker command.
See 'docker --help'
root@usuario:/home/usuario# docker volume ls

DRIVER          VOLUME NAME
local           73f52a5ff2bbefc42903e4315da181884b4b5fb6d8f46a64fcb48e2f15dbf043
local           e6893b55a515fce92167af832f8e9fe5ca29cb3911b67d258916613f6732f019
local           volumen_web
root@usuario:/home/usuario#
```

4.4. Comprobar que “c1” está montado sobre “volumen_web”

Debemos utilizar la orden “**docker inspect <nombre o ID>**”.

```
2 de ene 19:19
root@usuario:/home/usuario
root@usuario:/home/usuario# docker inspect c1
[
  {
```

Ahora bajamos hasta la parte de “Mounts” y comprobamos en que volumen está. En este caso, está sobre “volumen_web”.

```
"Mounts": [
  {
    "Type": "volume",
    "Name": "volumen_web",
    "Source": "/var/lib/docker/volumes/volumen_web/_data",
    "Destination": "/var/www/html",
    "Driver": "local",
    "Mode": "z",
    "RW": true,
    "Propagation": ""
  }
]
```