

EJERCICIOS SISTEMAS

INFORMÁTICOS TEMA 4:

PRÁCTICAS DE DOCKER



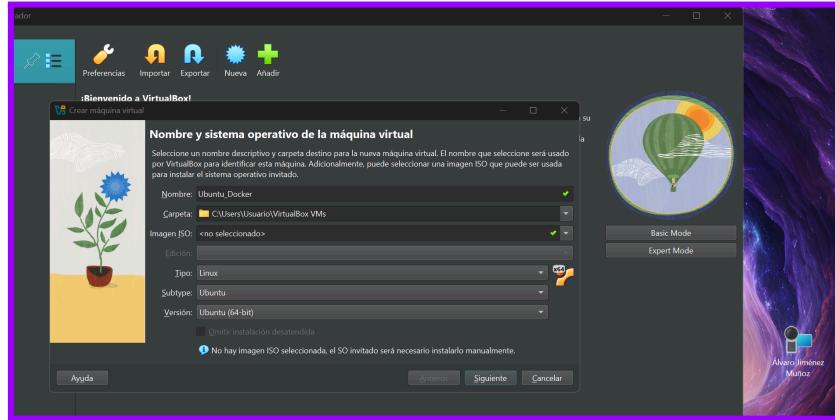
1.º DAM

ÍNDICE

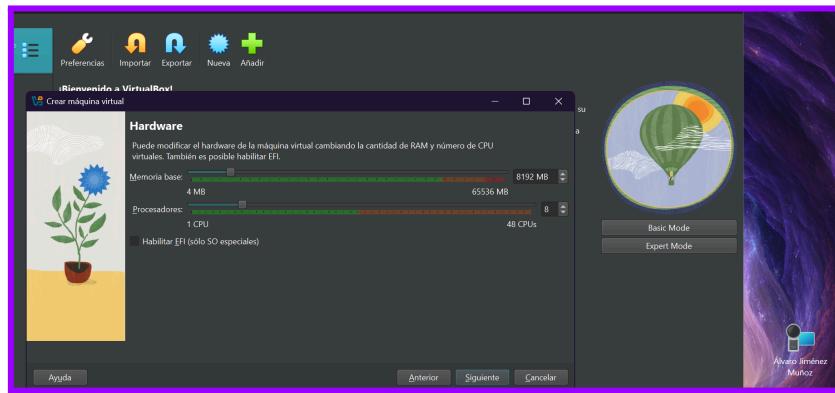
Pág. 3. <u>Creación de máquina e instalación de Docker con terminal</u>
Pág. 8. <u>Actividad 1</u>
Pág. 12. <u>Actividad 2</u>
Pág. 17. <u>Actividad 3</u>
Pág. 21. <u>Actividad 4</u>
Pág. 24. <u>Actividad 5</u>
Pág. 29. <u>Actividad 6</u>
Pág. 34. <u>Actividad 7</u>
Pág. 38. <u>BIBLIOGRAFÍA</u>

Para descargar Docker en una máquina virtual de Ubuntu a través de la terminal con VirtualBox, lo primero que se debe hacer es crear dicha máquina como de costumbre, estableciendo todos sus parámetros iniciales de la siguiente manera:

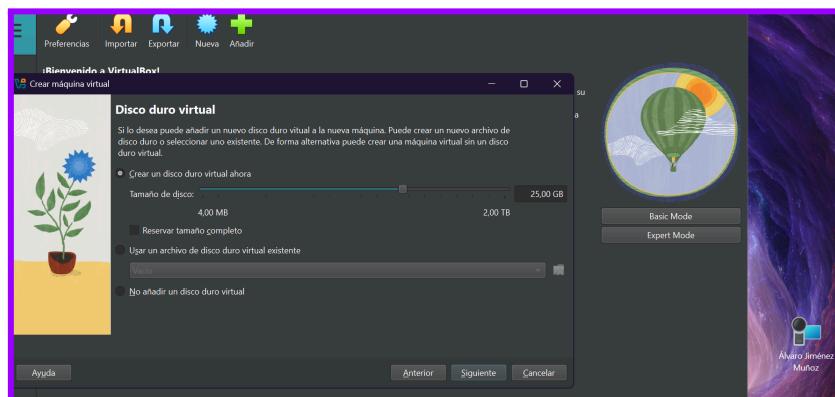
Se selecciona nombre, ruta y tipo de sistema operativo:



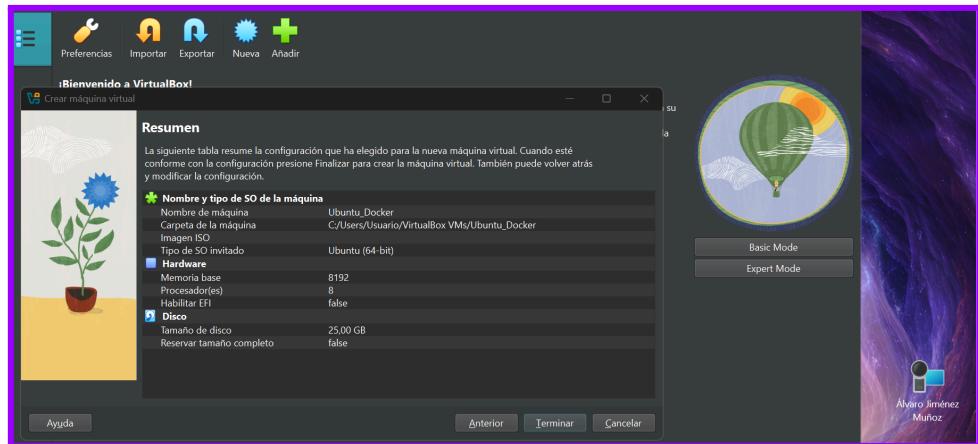
Memoria y número de procesador de la VM:



Tamaño del disco duro virtual:



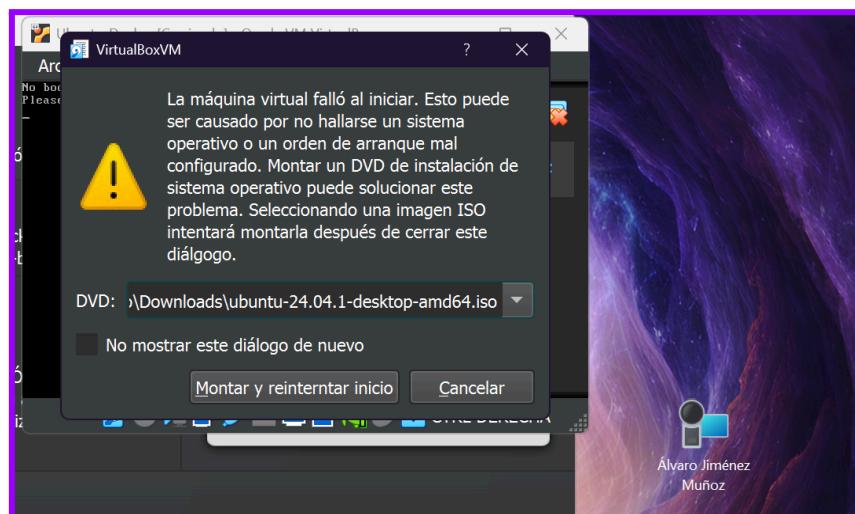
Así quedaría:



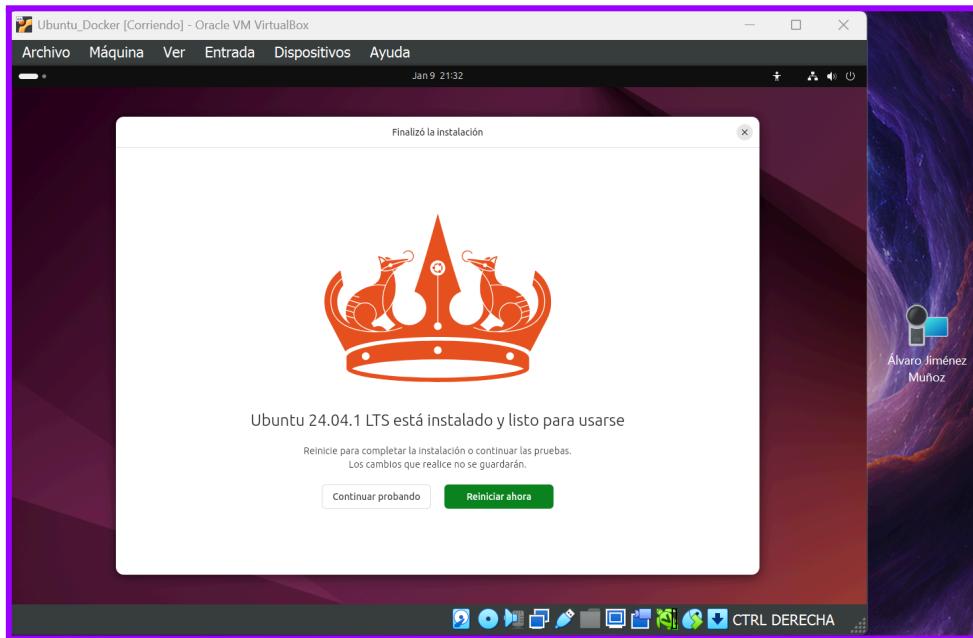
Después se inicia por primera vez:



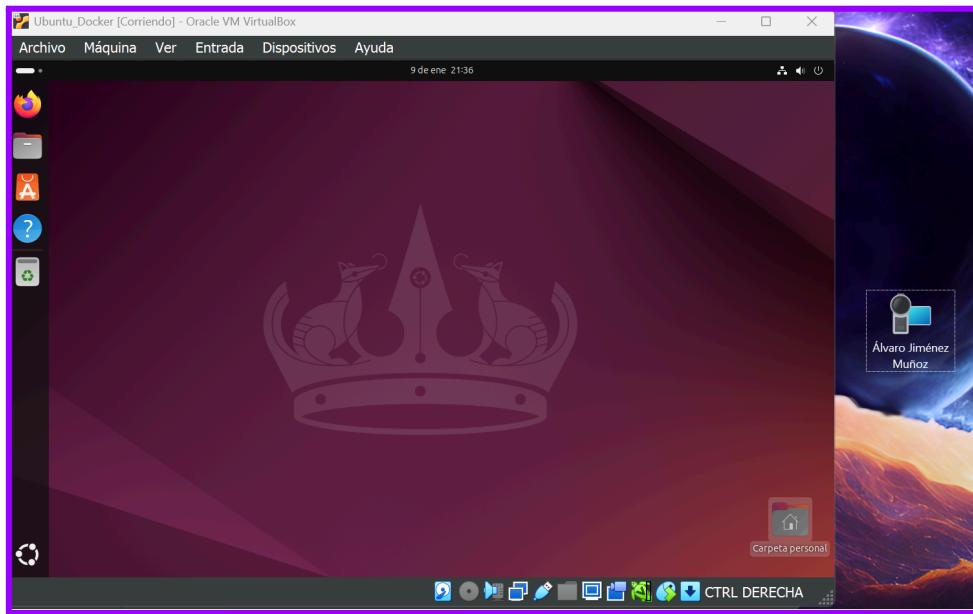
Al hacerlo, pedirá la ISO del sistema operativo, por lo que se pone la más reciente de Ubuntu:



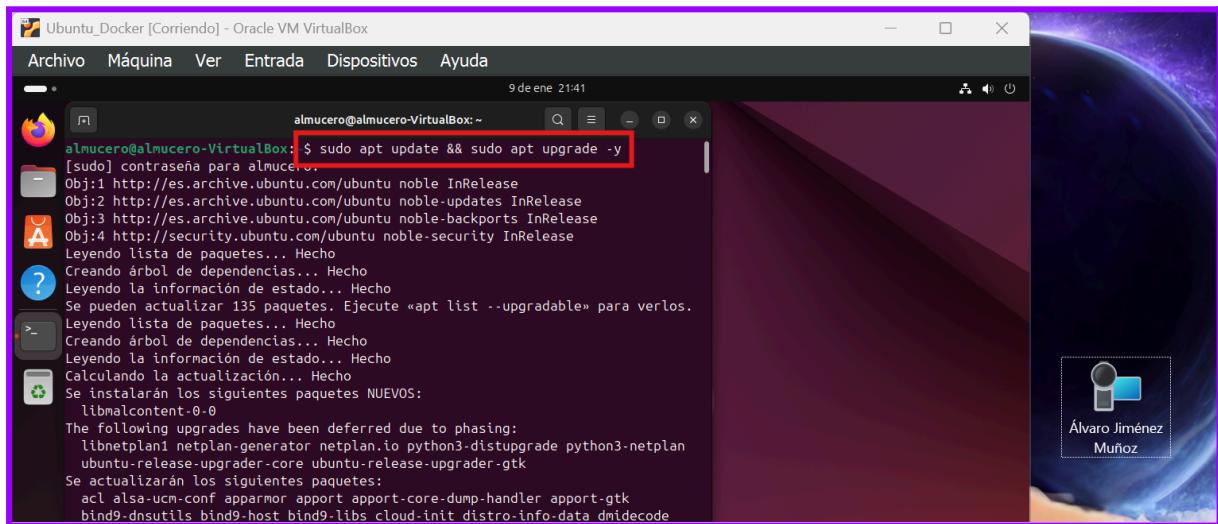
Después de eso se continúa con todo el proceso de instalación que aparecerá hasta el final:



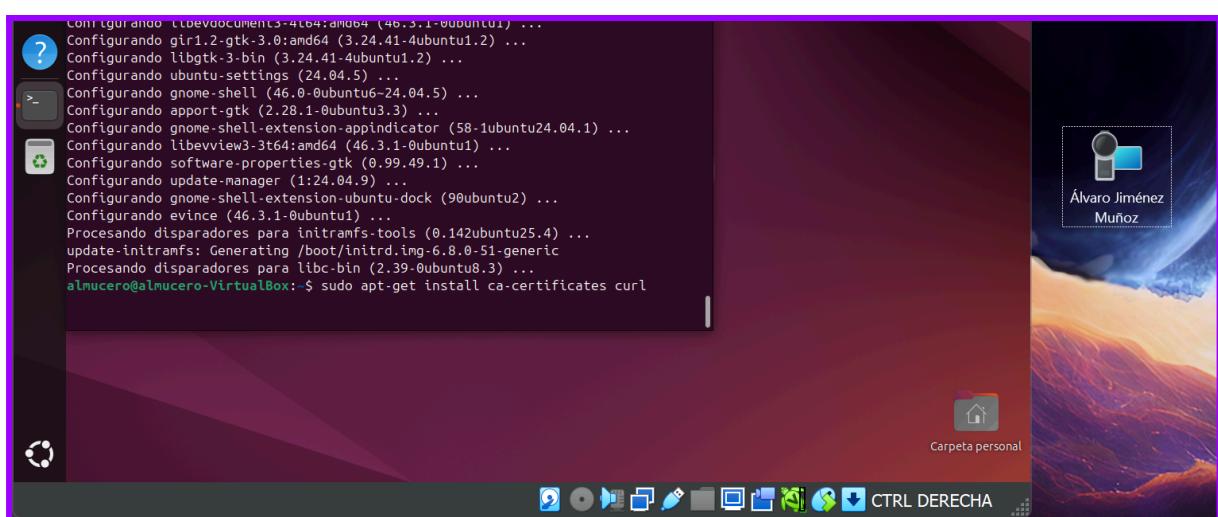
Tras reiniciar, la máquina estará completamente lista y funcional con Ubuntu instalado:



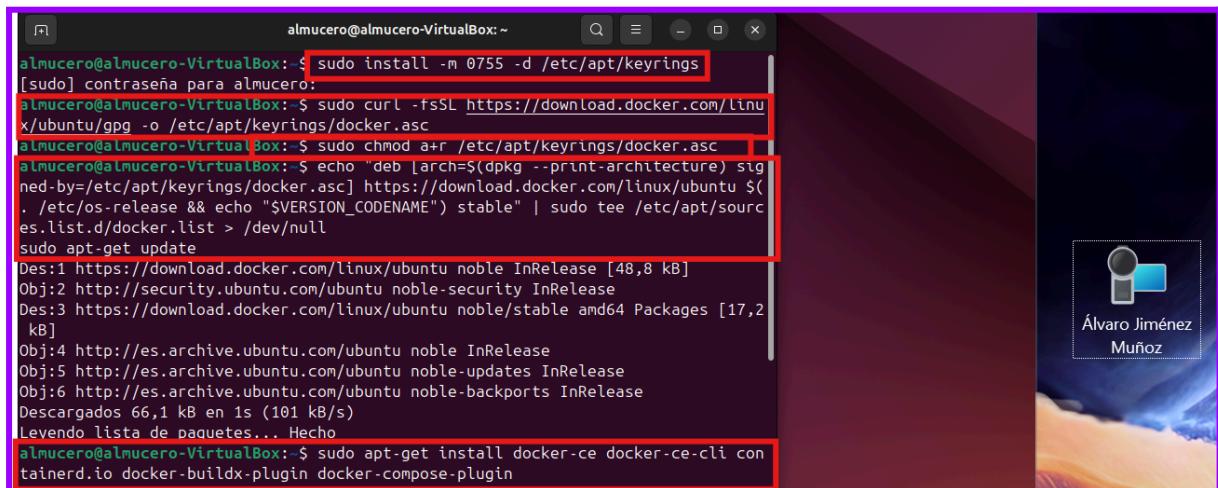
Ahora, para instalar Docker a través del terminal de la misma, se deben de introducir los siguientes comandos en orden:



```
Ubuntu_Docker [Corriendo] - Oracle VM VirtualBox
Archivo Máquina Ver Entrada Dispositivos Ayuda
9 de ene 21:41
almucero@almucero-VirtualBox: ~
[sudo] contraseña para almucero:
Obj:1 http://es.archive.ubuntu.com/ubuntu noble InRelease
Obj:2 http://es.archive.ubuntu.com/ubuntu noble-updates InRelease
Obj:3 http://es.archive.ubuntu.com/ubuntu noble-backports InRelease
Obj:4 http://security.ubuntu.com/ubuntu noble-security InRelease
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias... Hecho
Leyendo la información de estado... Hecho
Se pueden actualizar 135 paquetes. Ejecute «apt list --upgradable» para verlos.
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias... Hecho
Leyendo la información de estado... Hecho
Calculando la actualización... Hecho
Se instalarán los siguientes paquetes NUEVOS:
libmcalcontent-0-0
The following upgrades have been deferred due to phasing:
 libnetplan netplan-generator netplan.io python3-distupgrade python3-netplan
 ubuntu-release-upgrader-core ubuntu-release-upgrader-gtk
Se actualizarán los siguientes paquetes:
 acl alsas-ucm-conf apparmor apport apport-core-dump-handler apport-gtk
 bind9-dnsutils bind9-host bind9-libs cloud-init distro-info-data dmidecode
```

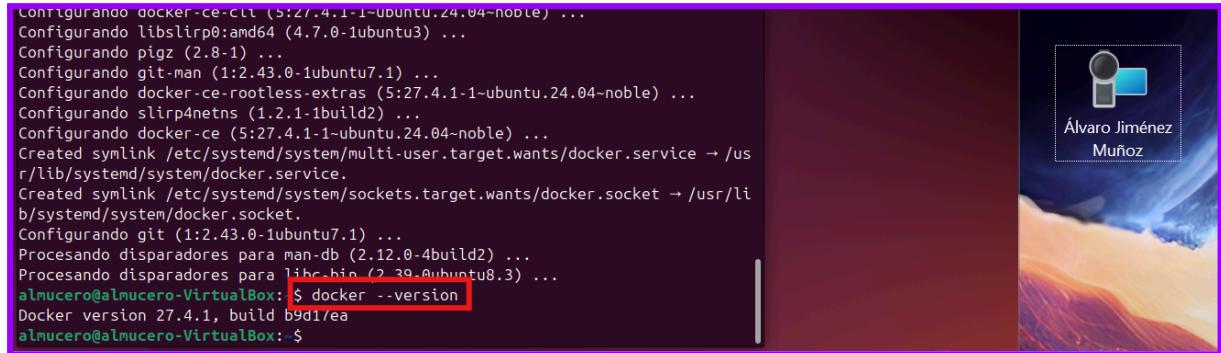


```
Configurando libevview3-3764:amd64 (46.3.1-0ubuntu1) ...
Configurando gir1.2-gtk-3.0:amd64 (3.24.41-4ubuntu1.2) ...
Configurando libgtk-3-bin (3.24.41-4ubuntu1.2) ...
Configurando ubuntu-settings (24.04.5) ...
Configurando gnome-shell (46.0-0ubuntu6-24.04.5) ...
Configurando apport-gtk (2.28.1-0ubuntu3.3) ...
Configurando gnome-shell-extension-appindicator (58-1ubuntu24.04.1) ...
Configurando libevview3-3764:amd64 (46.3.1-0ubuntu1) ...
Configurando software-properties-gtk (0.99.49.1) ...
Configurando update-manager (1:24.04.9) ...
Configurando gnome-shell-extension-ubuntu-dock (90ubuntu2) ...
Configurando evince (46.3.1-0ubuntu1) ...
Procesando disparadores para initramfs-tools (0.142ubuntu25.4) ...
update-initramfs: Generating /boot/initrd.img-6.8.0-51-generic
Procesando disparadores para libc-bin (2.39-0ubuntu8.3) ...
almucero@almucero-VirtualBox: ~$ sudo apt-get install ca-certificates curl
```



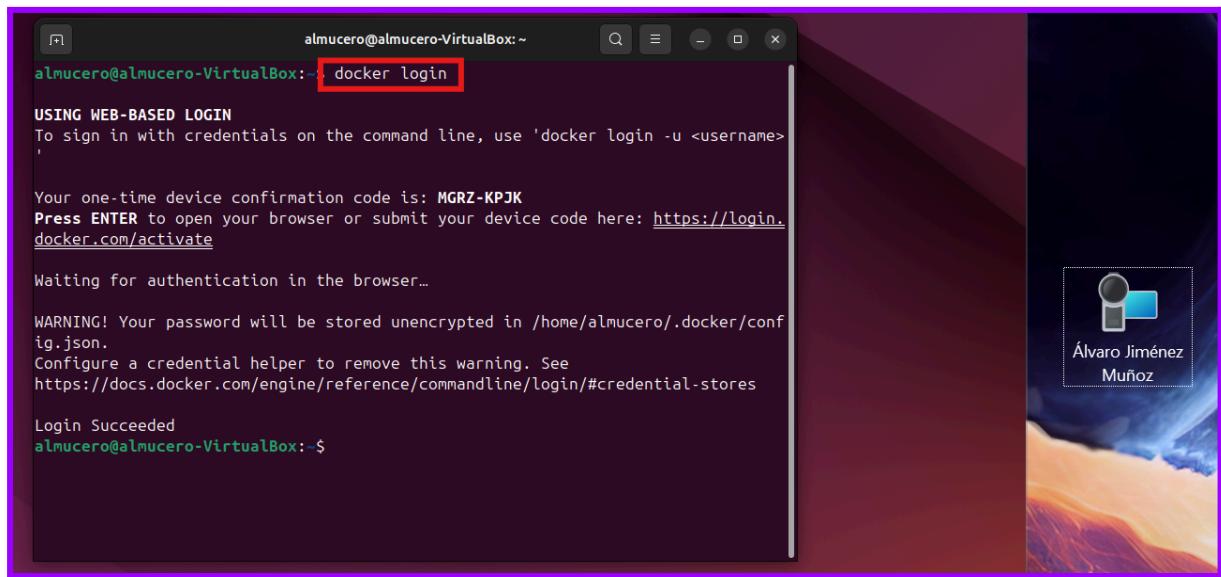
```
almucero@almucero-VirtualBox: ~$ sudo install -m 0755 -d /etc/apt/keyrings
[sudo] contraseña para almucero:
almucero@almucero-VirtualBox: ~$ sudo curl -fsSL https://download.docker.com/linux/ubuntu/gpg | gpg --dearmor > /etc/apt/keyrings/docker.asc
almucero@almucero-VirtualBox: ~$ sudo chmod a+r /etc/apt/keyrings/docker.asc
almucero@almucero-VirtualBox: ~$ echo 'deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.asc] https://download.docker.com/linux/ubuntu $( . /etc/os-release && echo "$VERSION_CODENAME" ) stable' | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
sudo apt-get update
Des:1 https://download.docker.com/linux/ubuntu noble InRelease [48,8 kB]
Obj:2 http://security.ubuntu.com/ubuntu noble-security InRelease
Des:3 https://download.docker.com/linux/ubuntu/stable amd64 Packages [17,2 kB]
Obj:4 http://es.archive.ubuntu.com/ubuntu noble InRelease
Obj:5 http://es.archive.ubuntu.com/ubuntu noble-updates InRelease
Obj:6 http://es.archive.ubuntu.com/ubuntu noble-backports InRelease
Descargados 66,1 kB en 1s (101 kB/s)
Leyendo lista de paquetes... Hecho
almucero@almucero-VirtualBox: ~$ sudo apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin
```

Después de esto, Docker ya se habrá instalado satisfactoriamente; para comprobarlo, se ejecuta el siguiente comando, que notificará de la versión actual de Docker instalado en la VM. Si da una respuesta como aquí se muestra, es que se ha instalado:



```
Configurando docker-ce-clt (5:27.4.1-1~ubuntu.24.04-noble) ...
Configurando libslirp0:amd64 (4.7.0-1ubuntu3) ...
Configurando pigz (2.8-1) ...
Configurando git-man (1:2.43.0-1ubuntu7.1) ...
Configurando docker-ce-rootless-extras (5:27.4.1-1~ubuntu.24.04-noble) ...
Configurando slirp4netns (1.2.1-1build2) ...
Configurando docker-ce (5:27.4.1-1~ubuntu.24.04-noble) ...
Created symlink /etc/systemd/system/multi-user.target.wants/docker.service → /usr/lib/systemd/system/docker.service.
Created symlink /etc/systemd/system/sockets.target.wants/docker.socket → /usr/lib/systemd/system/docker.socket.
Configurando git (1:2.43.0-1ubuntu7.1) ...
Procesando disparadores para man-db (2.12.0-4build2) ...
Procesando disparadores para libc-bin (2.39-0ubuntu8.3) ...
almucero@almucero-VirtualBox: $ docker --version
Docker version 27.4.1, build b9d1/ea
almucero@almucero-VirtualBox: $
```

Por último, para evitar posibles problemas a la hora de descargar imágenes y demás, se inicia sesión en Docker con este comando y se siguen los pasos indicados:



```
almucero@almucero-VirtualBox: ~ $ docker login
USING WEB-BASED LOGIN
To sign in with credentials on the command line, use 'docker login -u <username>'

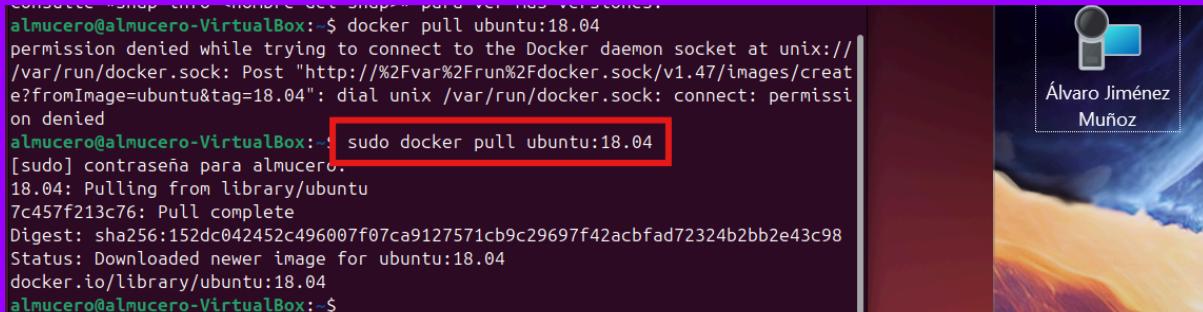
Your one-time device confirmation code is: MGRZ-KPJK
Press ENTER to open your browser or submit your device code here: https://login.docker.com/activate

Waiting for authentication in the browser...
WARNING! Your password will be stored unencrypted in /home/almucero/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credential-stores

Login Succeeded
almucero@almucero-VirtualBox: ~ $
```

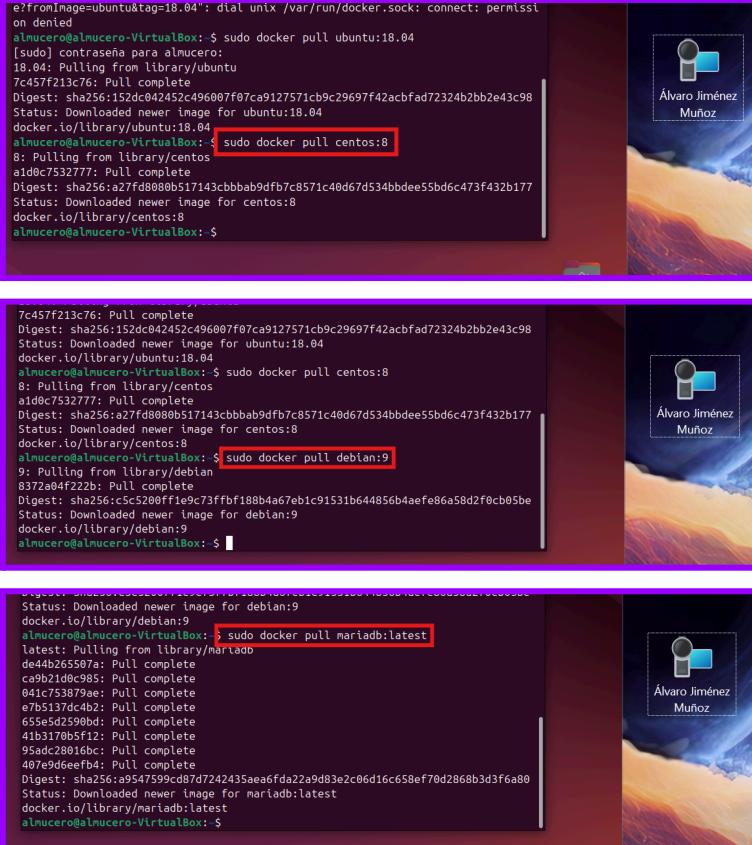
Actividad 1

Para la primera parte de la actividad, descargas diversas imágenes de Docker Hub (`ubuntu:18.04`, `centos:8`, `debian:9`, `mariadb:latest`, `mysql:5.7`, `httpd`, `tomcat:9.0.39-jdk11`, `jenkins/jenkins:lts`, `php:7.3-apache`). Se ejecuta el siguiente comando seguido de la versión que se descargará de la imagen en cuestión:

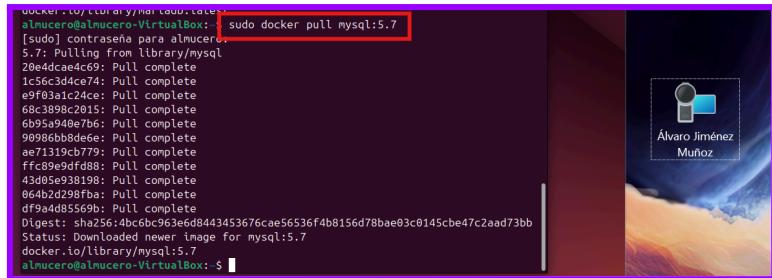


```
... Consulte https://hub.docker.com/ para ver las versiones.  
almucero@almucero-VirtualBox: $ docker pull ubuntu:18.04  
permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock: Post "http://%2Fvar%2Frun%2Fdocker.sock/v1.47/images/create?fromImage=ubuntu&tag=18.04": dial unix /var/run/docker.sock: connect: permission denied  
[sudo] contraseña para almucero.  
18.04: Pulling from library/ubuntu  
7c457f213c76: Pull complete  
Digest: sha256:1152dc042452c496007f07ca9127571cb9c29697f42acbfad72324b2bb2e43c98  
Status: Downloaded newer image for ubuntu:18.04  
docker.io/library/ubuntu:18.04  
almucero@almucero-VirtualBox: $
```

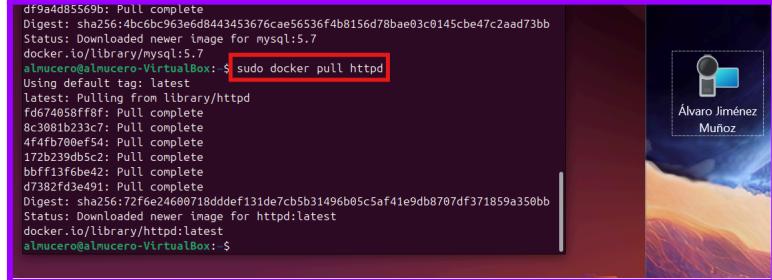
Se hace lo mismo con el resto de imágenes:



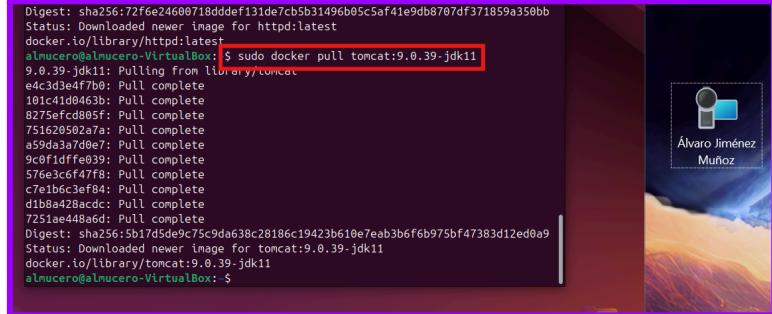
```
e?fromImage=ubuntu&tag=18.04": dial unix /var/run/docker.sock: connect: permission denied  
[sudo] contraseña para almucero:  
18.04: Pulling from library/ubuntu  
7c457f213c76: Pull complete  
Digest: sha256:1152dc042452c496007f07ca9127571cb9c29697f42acbfad72324b2bb2e43c98  
Status: Downloaded newer image for ubuntu:18.04  
docker.io/library/ubuntu:18.04  
almucero@almucero-VirtualBox: $ sudo docker pull centos:8  
8: Pulling from library/centos  
alid0c7532777: Pull complete  
Digest: sha256:a27fd8080b517143cbbb9dfb7c8571c40d67d534bbdee55bd6c473f432b177  
Status: Downloaded newer image for centos:8  
docker.io/library/centos:8  
almucero@almucero-VirtualBox: $  
  
7c457f213c76: Pull complete  
Digest: sha256:1152dc042452c496007f07ca9127571cb9c29697f42acbfad72324b2bb2e43c98  
Status: Downloaded newer image for ubuntu:18.04  
docker.io/library/ubuntu:18.04  
almucero@almucero-VirtualBox: $ sudo docker pull centos:8  
8: Pulling from library/centos  
alid0c7532777: Pull complete  
Digest: sha256:a27fd8080b517143cbbb9dfb7c8571c40d67d534bbdee55bd6c473f432b177  
Status: Downloaded newer image for centos:8  
docker.io/library/centos:8  
almucero@almucero-VirtualBox: $ sudo docker pull debian:9  
9: Pulling from library/debian  
8372a04f222b: Pull complete  
Digest: sha256:c5c520ff1e9c73ffbf188b4a67eb1c91531b644856b4aefef86a58d2f0cb05be  
Status: Downloaded newer image for debian:9  
docker.io/library/debian:9  
almucero@almucero-VirtualBox: $  
  
Status: Downloaded newer image for debian:9  
docker.io/library/debian:9  
almucero@almucero-VirtualBox: $ sudo docker pull mariadb:latest  
latest: Pulling from library/mariadb  
de440265507a: Pull complete  
ca9b210c985: Pull complete  
041c75879ae: Pull complete  
e705137dc4b2: Pull complete  
655e5d2590bd: Pull complete  
41b317065f12: Pull complete  
95adec28016bc: Pull complete  
407e9d0eefb4: Pull complete  
Digest: sha256:a9547599cd87d7242435aea6fda22a9db83e2c06d16c658ef70d2868b3d3f6a80  
Status: Downloaded newer image for mariadb:latest  
docker.io/library/mariadb:latest  
almucero@almucero-VirtualBox: $
```



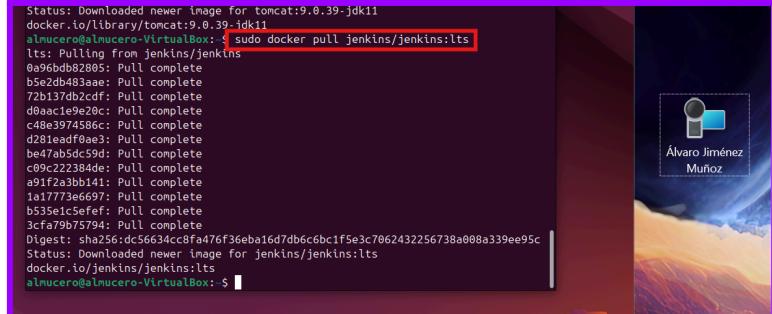
```
[sudo] contraseña para almucero.
[almucero@almucero-VirtualBox: ~] sudo docker pull mysql:5.7
5.7: Pulling from library/mysql
20e04dcae4c69: Pull complete
1c5c53d3dce74: Pull complete
e9f93a1c24ce: Pull complete
68c3898c2015: Pull complete
6b95a940e7b6: Pull complete
90986bb8de66: Pull complete
ae71319cb779: Pull complete
ffc39e9df088: Pull complete
43d05e938198: Pull complete
064b2d298fb9: Pull complete
df94ad85569b: Pull complete
Digest: sha256:4bc6b963e6d8443453676cae56536f4b8156d78bae03c0145cbe47c2aad73bb
Status: Downloaded newer image for mysql:5.7
docker.io/library/mysql:5.7
[almucero@almucero-VirtualBox: ~]
```

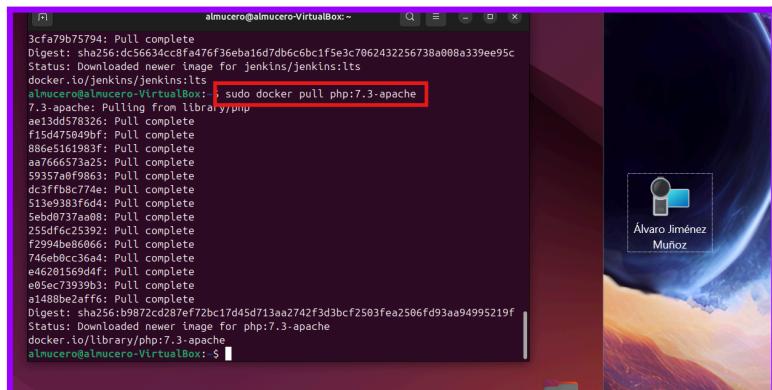
```
df9a4d85569b: Pull complete
Digest: sha256:4bc6b963e6d8443453676cae56536f4b8156d78bae03c0145cbe47c2aad73bb
Status: Downloaded newer image for mysql:5.7
docker.io/library/mysql:5.7
[almucero@almucero-VirtualBox: ~] sudo docker pull httpd
Using default tag: latest
latest: Pulling from library/httpd
fd674058ffbf: Pull complete
8c3081b233c7: Pull complete
4f4fb700ef54: Pull complete
172b239db5c2: Pull complete
bbf13f6be42: Pull complete
d7382fd3e491: Pull complete
Digest: sha256:72f6e24600718dddef131de7cb5b31496b05c5af41e9db8707df371859a350bb
Status: Downloaded newer image for httpd:latest
docker.io/library/httpd:latest
[almucero@almucero-VirtualBox: ~]
```

```
Digest: sha256:72f6e24600718dddef131de7cb5b31496b05c5af41e9db8707df371859a350bb
Status: Downloaded newer image for httpd:latest
docker.io/library/httpd:latest
[almucero@almucero-VirtualBox: ~] sudo docker pull tomcat:9.0.39-jdk11
9.0.39-jdk11: Pulling from library/tomcat
e4cd3d3e4f7b0: Pull complete
101c41d0463b: Pull complete
8275fcfd005f: Pull complete
751620502a7a: Pull complete
a595da3a7d0e7: Pull complete
9c0f1dffe039: Pull complete
576e3c6f47f8: Pull complete
c7e1b6c3e8f84: Pull complete
d1b8a428acd: 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
[almucero@almucero-VirtualBox: ~]
```

```
Status: Downloaded newer image for tomcat:9.0.39-jdk11
docker.io/library/tomcat:9.0.39-jdk11
[almucero@almucero-VirtualBox: ~] sudo docker pull jenkins/jenkins:lts
lts: Pulling from jenkins/jenkins
0a96bdb82805: Pull complete
b5e20b483aae: Pull complete
72ba137db2cdf: Pull complete
d0aac1e9e20c: Pull complete
c48e3974586c: Pull complete
d281eadf0ae3: Pull complete
bed7ab5dc59d: Pull complete
c09c222384dc: Pull complete
a91f2aabb141: Pull complete
1a17773e6697: Pull complete
b535e1c5effe: Pull complete
3ca79b757794: Pull complete
Digest: sha256:dc56634cc8ffa476f36eba16d7db6c6bc1f5e3c7062432256738a008a339ee95c
Status: Downloaded newer image for jenkins/jenkins:lts
docker.io/jenkins/jenkins:lts
[almucero@almucero-VirtualBox: ~]
```

```
3ca79b757794: Pull complete
Digest: sha256:dc56634cc8ffa476f36eba16d7db6c6bc1f5e3c7062432256738a008a339ee95c
Status: Downloaded newer image for jenkins/jenkins:lts
docker.io/jenkins/jenkins:lts
[almucero@almucero-VirtualBox: ~] sudo docker pull php:7.3-apache
7.3-apache: Pulling from library/php
ae13d5d7832c: Pull complete
f15d475049bf: Pull complete
88e6161983f: Pull complete
ea7666573a25: Pull complete
59357a0f9863: Pull complete
dc3ffbb6c774a: Pull complete
513e383f6d4: Pull complete
5eb0d737a08b: Pull complete
255df6c25392: Pull complete
f2994be86066: Pull complete
746eb0c36d4: Pull complete
e46281569d4f: Pull complete
e05ec73939b5: Pull complete
a1488be2ff66: Pull complete
Digest: sha256:b9872cd87ef72bc17d45d713aa2742f3d3bcf2503fea2506fd93aa94995219f
Status: Downloaded newer image for php:7.3-apache
docker.io/library/php:7.3-apache
[almucero@almucero-VirtualBox: ~]
```

Tras ejecutar esos comandos, se habrán descargado todas las imágenes; para comprobar que eso realmente ha ocurrido y que no falta ninguna, se ejecuta el siguiente comando, que muestra todas las imágenes descargadas:

```
/var/run/docker.sock: Head "http://%2Fvar%2Frun%2Fdocker.sock/_ping": dial unix  
/var/run/docker.sock: connect: permission denied  
[sudo] contraseña para almucero:  
REPOSITORY TAG IMAGE ID CREATED SIZE  
jenkins/jenkins lts 44c1caef796 32 hours ago 468MB  
mariadb latest 6722945a6940 7 weeks ago 407MB  
httpd latest 4ce47c750a58 5 months ago 147MB  
mysql 5.7 5107333e08a8 13 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  
almucero@almucero-VirtualBox:~$
```

Aquí pueden verse todas las imágenes descargadas; luego, para arrancar un contenedor de la imagen **ubuntu:18.04** llamado **ubuntu**, se ejecuta el siguiente comando:

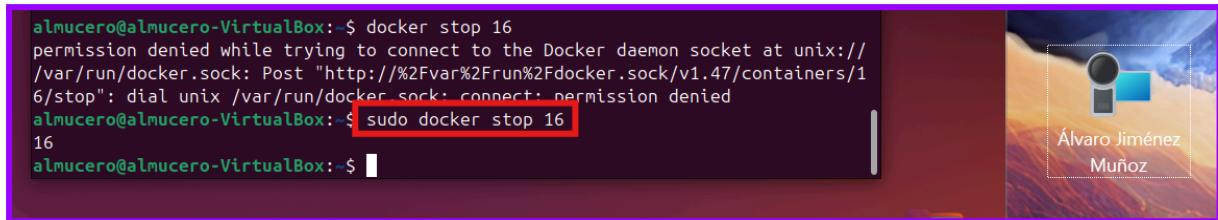
```
unix:///var/run/docker.sock: Head "http://%2Fvar%2Frun%2Fdocker.sock/_ping": di  
al unix /var/run/docker.sock: connect: permission denied.  
See 'docker run --help'.  
[sudo] contraseña para almucero:  
almucero@almucero-VirtualBox: $ sudo docker run -dit --name ubuntu ubuntu:18.04  
163276e8c75ea7e8bc7da2c4e3abeff314c619f3f17fb06ef1634ac6f619b7b6  
almucero@almucero-VirtualBox: $
```

Y con este comando se comprueba si se ha arrancado de verdad:

```
al unix /var/run/docker.sock: connect: permission denied.  
See 'docker run --help'.  
[sudo] contraseña para almucero:  
almucero@almucero-VirtualBox: $ sudo docker run -dit --name ubuntu ubuntu:18.04  
163276e8c75ea7e8bc7da2c4e3abeff314c619f3f17fb06ef1634ac6f619b7b6  
almucero@almucero-VirtualBox: $ sudo docker ps -a  
CONTAINER ID IMAGE COMMAND CREATED STATUS  
PORTS NAMES  
163276e8c75e ubuntu:18.04 "/bin/bash" About a minute ago Up About a minu  
te  
tue  
ubuntu  
almucero@almucero-VirtualBox: $
```

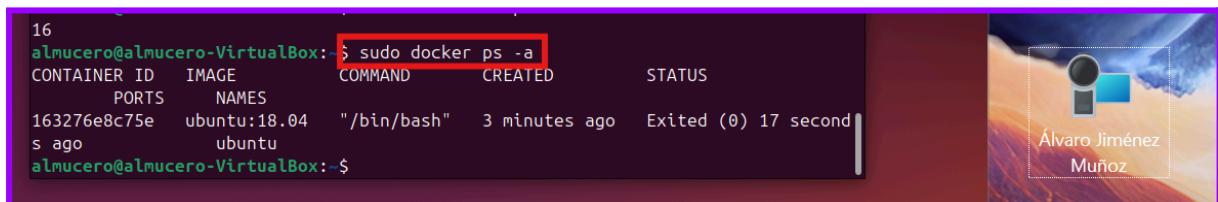
Para el siguiente paso, salir del contenedor, se ejecuta el siguiente comando:

(Aquí no es necesario poner por completo el nombre del contenedor, ya que con poner las iniciales del mismo de forma que se diferencie del resto ya debe valer).



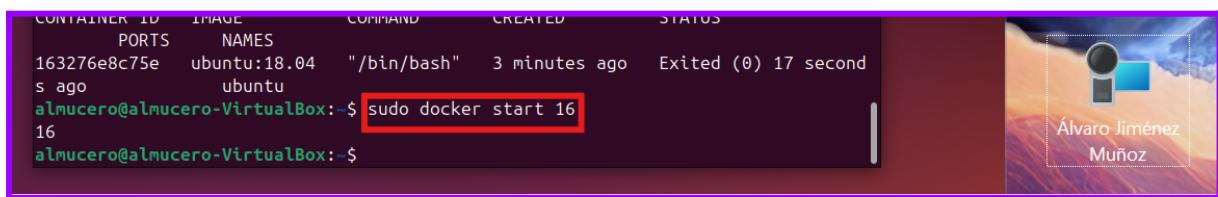
```
almucero@almucero-VirtualBox: $ docker stop 16
permission denied while trying to connect to the Docker daemon socket at unix://
/var/run/docker.sock: Post "http://%2Fvar%2Frun%2Fdocker.sock/v1.47/containers/1
6/stop": dial unix /var/run/docker.sock: connect: permission denied
almucero@almucero-VirtualBox: $ sudo docker stop 16
16
almucero@almucero-VirtualBox: $
```

De nuevo, con el mismo comando de antes, se comprueba que está parado de verdad:



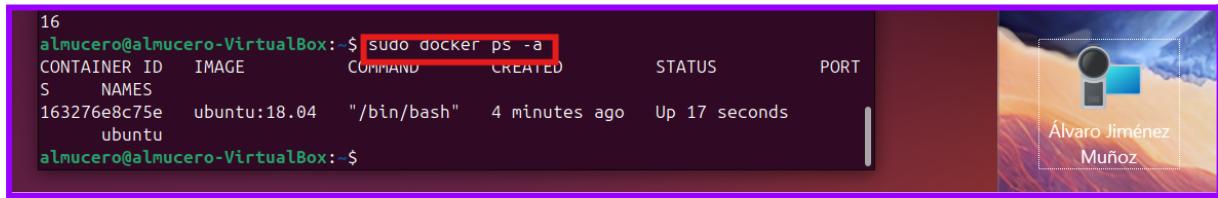
```
16
almucero@almucero-VirtualBox: $ sudo docker ps -a
CONTAINER ID IMAGE COMMAND CREATED STATUS
PORTS NAMES
163276e8c75e ubuntu:18.04 "/bin/bash" 3 minutes ago Exited (0) 17 seconds ago
ubuntu
almucero@almucero-VirtualBox:~$
```

Para arrancar de nuevo el contenedor, se hace lo siguiente (usando solo las iniciales nuevamente):



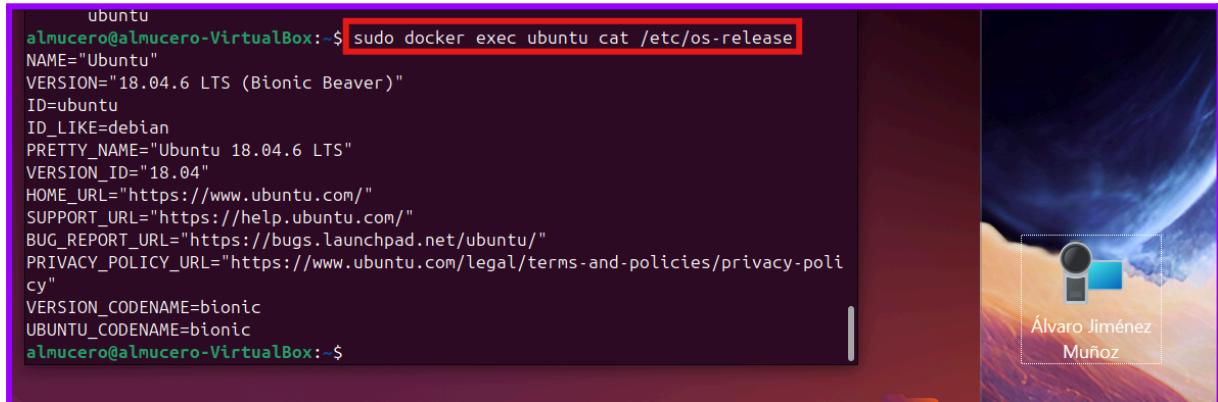
```
CONTAINER ID IMAGE COMMAND CREATED STATUS
PORTS NAMES
163276e8c75e ubuntu:18.04 "/bin/bash" 3 minutes ago Exited (0) 17 seconds ago
ubuntu
almucero@almucero-VirtualBox: $ sudo docker start 16
16
almucero@almucero-VirtualBox: $
```

Y se comprueba nuevamente que se ha vuelto a arrancar:



```
16
almucero@almucero-VirtualBox: $ sudo docker ps -a
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
163276e8c75e ubuntu:18.04 "/bin/bash" 4 minutes ago Up 17 seconds
ubuntu
almucero@almucero-VirtualBox: $
```

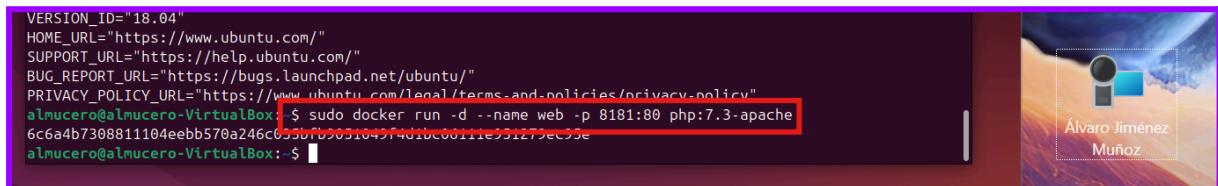
Para mostrar en la terminal el fichero “/etc/os-release”, sin entrar en el contenedor, se hace lo siguiente, usando “cat”:



```
ubuntu
almucero@almucero-VirtualBox: $ sudo 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
almucero@almucero-VirtualBox: $
```

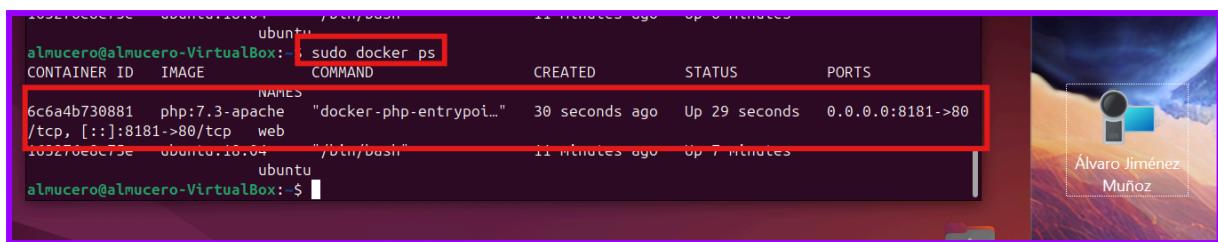
Actividad 2:

Para arrancar un contenedor que ejecute una instancia de la imagen **php:7.3-apache** llamada “web” y accesible desde el puerto **8181**, se siguen los siguientes pasos en la terminal:



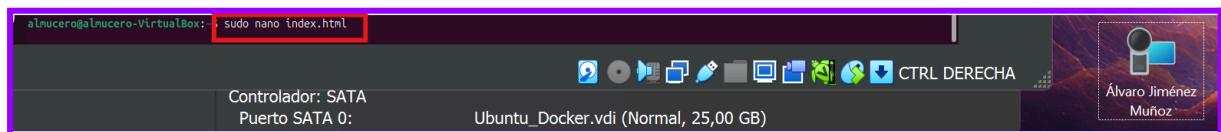
```
VERSION_ID="18.04"
HOME_URL="https://www.ubuntu.com/"
SUPPORT_URL="https://help.ubuntu.com/"
BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
almucero@almucero-VirtualBox: $ sudo docker run -d --name web -p 8181:80 php:7.3-apache
6c6a4b7308811104ebeb570a246c055bfbd30310497401bc0011e031279ec05e
almucero@almucero-VirtualBox: $
```

Se comprueba que ha funcionado:

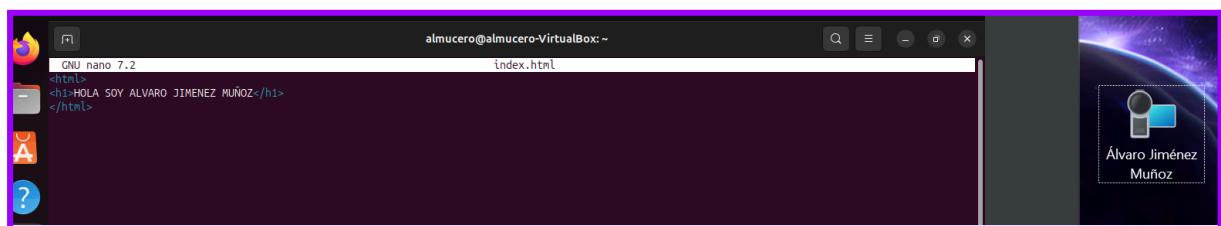


```
10:52:00 user@almucero-VirtualBox: ~ % sudo docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED            STATUS              PORTS
NAMES
6c6a4b730881      php:7.3-apache     "docker-php-entrypoi..."   30 seconds ago    Up 29 seconds    0.0.0.0:8181->80
/tcp, [::]:8181->80/tcp   web
10:52:00 user@almucero-VirtualBox: ~ %
```

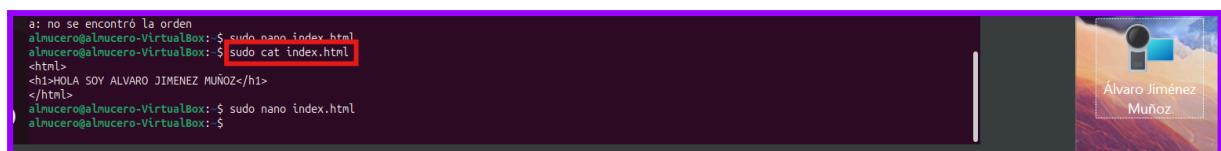
Luego, para colocar en el directorio raíz del servicio web del contenedor creado un fichero llamado “index.html”, se ejecuta el siguiente comando, que abrirá el archivo en cuestión:



Una vez dentro de este, se escriben sus contenidos en lenguaje **HTML** y, una vez hecho, se cierra el fichero guardando sus contenidos. Para ello se hace **Ctrl+X** y cuando diga si se quieren guardar los contenidos, se le da que **sí**. Después se establece el nombre, dejando el mismo, y se le da **Enter**:



Con el siguiente comando, **cat**, por terminal se ven los contenidos del fichero, y se comprueba si todo ha ido bien:



Después, se mueve este fichero ya finalizado al directorio raíz del servicio web:

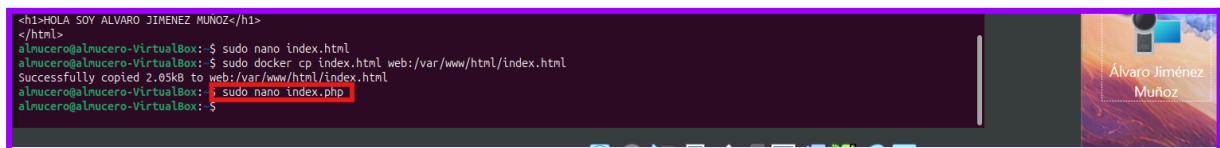


Finalmente, para comprobar que ha funcionado todo, se escribe en la barra superior del navegador “localhost:8181”, y se debería mostrar lo siguiente, indicando así que ha funcionado:

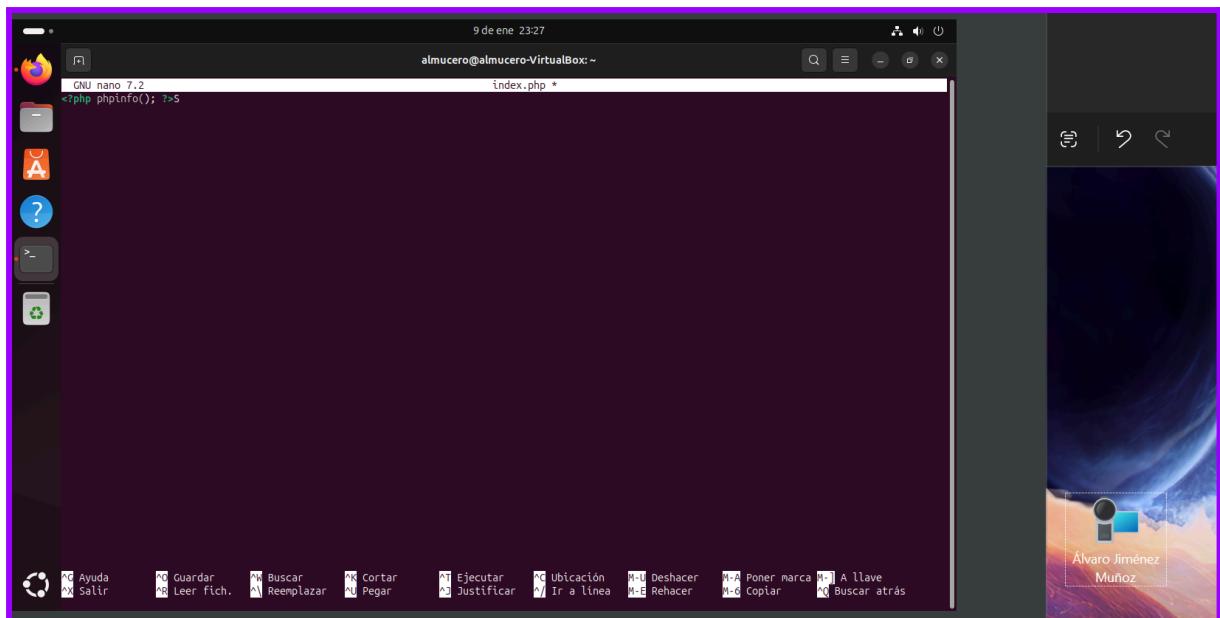


Una vez más, para colocar en ese mismo directorio raíz un nuevo archivo llamado “index.php”, se siguen los mismos pasos que antes:

Se crea el fichero:



Se establecen sus contenidos y se cierra guardando estos mismos, de la misma forma que antes:



Se comprueba que estos contenidos se han creado correctamente:



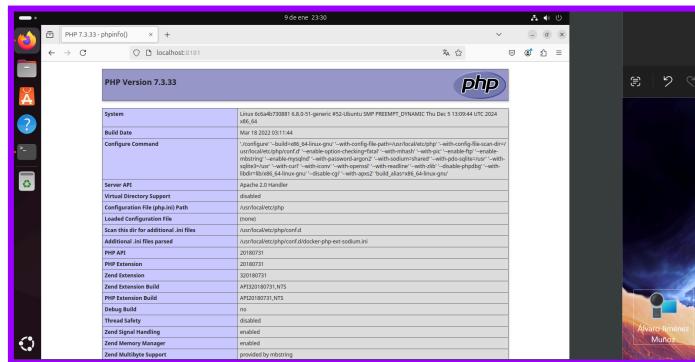
```
<HTML>
</HTML>
almucero@almucero-VirtualBox: ~$ sudo nano index.html
almucero@almucero-VirtualBox: ~$ sudo docker cp index.html web:/var/www/html/index.html
almucero@almucero-VirtualBox: ~$ ls -l /var/www/html/index.html
almucero@almucero-VirtualBox: ~$ curl http://127.0.0.1/index.html
almucero@almucero-VirtualBox: ~$ sudo cat index.php
<?php phinfo(); ?>
almucero@almucero-VirtualBox: ~$
```

Se mueve el fichero al directorio raíz del servicio web:



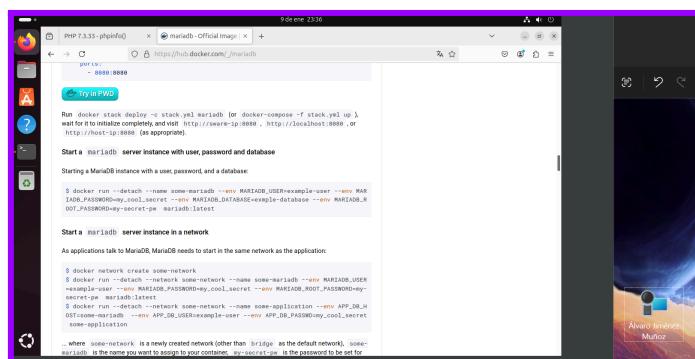
```
almucero@almucero-VirtualBox: ~$ sudo nano index.php
almucero@almucero-VirtualBox: ~$ sudo cat index.php
<?php phinfo(); ?>
almucero@almucero-VirtualBox: ~$ docker cp index.php web:/var/www/html/index.php
Successfully copied 0B to web/var/www/html/index.php
permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock. Put "http://<IPv4>:2375" in /etc/docker/daemon.json or use -H<IPv4>:2375 when running docker.
permision denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock: Put "http://<IPv4>:2375" in /etc/docker/daemon.json or use -H<IPv4>:2375 when running docker.
almucero@almucero-VirtualBox: ~$ sudo docker cp index.php web:/var/www/html/index.php
Successfully copied 2.058B to web/var/www/html/index.php
almucero@almucero-VirtualBox: ~$
```

Y se abre en el navegador:

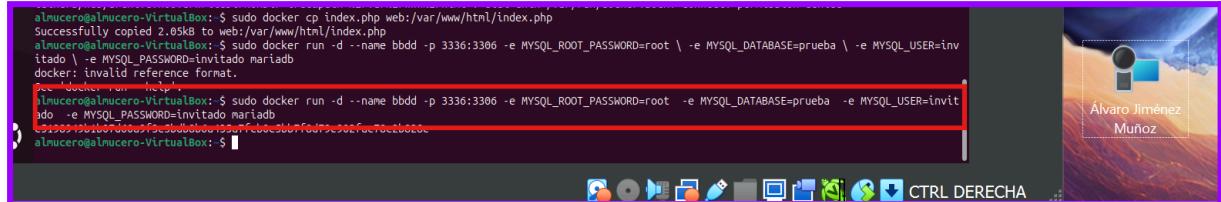


Para la siguiente etapa, arrancar un contenedor bajo el nombre “**bbdd**” que ejecute una instancia de la imagen mariadb, antes descargada, y que sea accesible desde el puerto **3336**; se siguen los siguientes pasos:

(Antes de nada, se visita la página del contenedor en Docker Hub y se mira cuáles serán algunos de los comandos necesarios para llevar a cabo el proceso, es decir, establecer las variables de entorno del contenedor a crear):

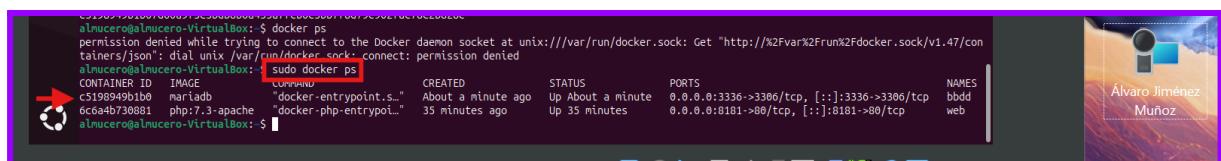


Una vez visitada la página, se crea el contenedor como de costumbre, pero añadiendo las variables de entorno que hacen que la contraseña de root sea root, se cree una base de datos automáticamente llamada “prueba” y se cree un usuario “invitado” con la contraseña “invitado”:



```
alnucero@alnucero-VirtualBox: $ sudo docker cp index.php web:/var/www/html/index.php
Successfully copied 2.05kB to web:/var/www/html/index.php
alnucero@alnucero-VirtualBox: $ sudo docker run -d --name bbdd -p 3336:3306 -e MYSQL_ROOT_PASSWORD=root -e MYSQL_DATABASE=prueba -e MYSQL_USER=invitado -e MYSQL_PASSWORD=invitado mariadb
docker: invalid reference format.
alnucero@alnucero-VirtualBox: $ ls
alnucero@alnucero-VirtualBox: $ sudo docker ps
alnucero@alnucero-VirtualBox: $
```

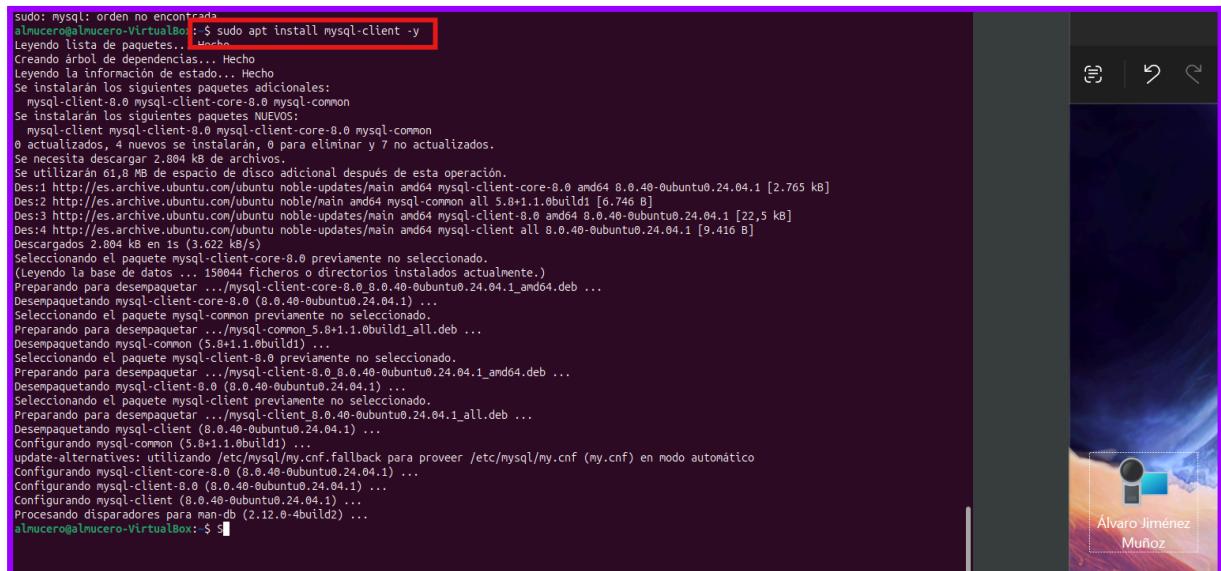
Se comprueba que se ha creado correctamente:



```
alnucero@alnucero-VirtualBox: $ docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
c5198949b100 mariadb "docker-entrypoint.s..." About a minute ago Up About a minute 0.0.0.0:3336->3306/tcp, [::]:3336->3306/tcp, 0.0.0.0:8181->80/tcp, [::]:8181->80/tcp web
bbdd
alnucero@alnucero-VirtualBox: $
```

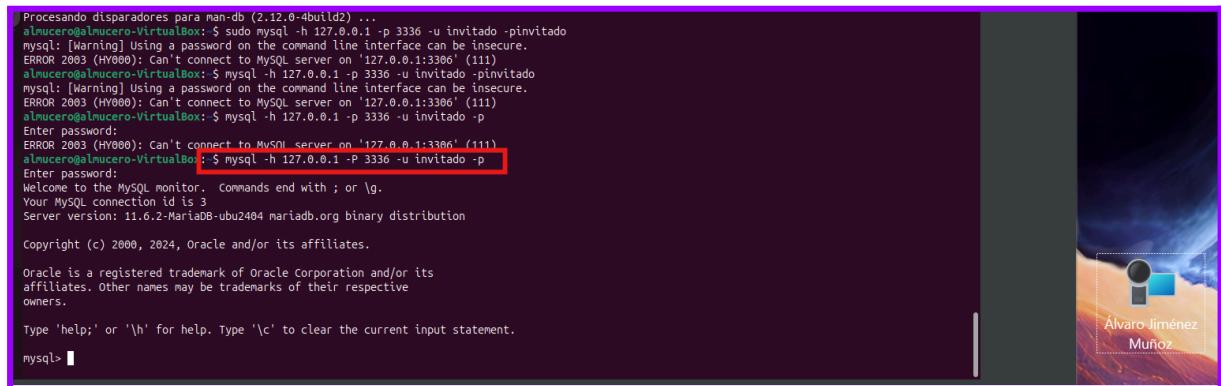
Para conectarnos al servidor de base de datos con el usuario antes creado y observar que verdaderamente se ha creado la base de datos “prueba” desde un cliente de base de datos, se hace lo siguiente:

Primero de todo, se instala la imagen MySQL necesaria para poder llevar a cabo el proceso mencionado:



```
sudo: MySQL: orden no encontrada
alnucero@alnucero-VirtualBox: $ sudo apt install mysql-client -y
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias... Hecho
Leyendo la información de estado... Hecho
Se instalarán los siguientes paquetes adicionales:
 mysql-client-8.0 mysql-client-core-8.0 mysql-common
Se instalarán los siguientes paquetes NUEVOS:
 mysql-client mysql-client-8.0 mysql-client-core-8.0 mysql-common
0 actualizados, 4 nuevos se instalarán, 0 para eliminar y 7 no actualizados.
Se necesita descargar 2.894 kB de archivos.
Se utilizarán 61,8 MB de espacio de disco adicional después de esta operación.
Descripción: 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]
Descripción: http://es.archive.ubuntu.com/ubuntu/noble/main amd64 mysql-common all 5.0+1.1.0build1 [6.746 B]
Descripción: http://es.archive.ubuntu.com/ubuntu/noble-updates/main amd64 mysql-client-8.0 amd64 8.0.40-0ubuntu0.24.04.1 [22,5 kB]
Descripción: http://es.archive.ubuntu.com/ubuntu/noble-updates/main amd64 mysql-client all 8.0.40-0ubuntu0.24.04.1 [9.416 B]
Descargados 2.894 kB en 1s (3.622 kB/s)
Seleccionando el paquete mysql-client-core-8.0 previamente no seleccionado.
(Leyendo la base de datos ... 150844 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 ...
Seleccionando el paquete mysql-common previamente no seleccionado.
Preparando para desempaquetar .../mysql-common_5.0+1.1.0build1_all.deb ...
Desempaquetando mysql-common_5.0+1.1.0build1 ...
Seleccionando el paquete mysql-client-8.0 previamente no seleccionado.
Preparando para desempaquetar .../mysql-client-8.0_8.0.40-0ubuntu0.24.04.1_amd64.deb ...
Desempaquetando mysql-client-8.0 (8.0.40-0ubuntu0.24.04.1) ...
Seleccionando el paquete mysql-client previamente no seleccionado.
Preparando para desempaquetar .../mysql-client_8.0.40-0ubuntu0.24.04.1_all.deb ...
Desempaquetando mysql-client_8.0.40-0ubuntu0.24.04.1 ...
Configurando mysql-common (5.0+1.1.0build1) ...
update-alternatives: utilizando /etc/mysql/my.cnf.fallback para proveer /etc/mysql/my.cnf (my.cnf) en modo automático
Configurando mysql-client-core-8.0 (8.0.40-0ubuntu0.24.04.1) ...
Configurando mysql-client-8.0 (8.0.40-0ubuntu0.24.04.1) ...
Configurando mysql-client (8.0.40-0ubuntu0.24.04.1) ...
Procesando dependencias para man-db (2.12.0-4build2) ...
alnucero@alnucero-VirtualBox: $
```

Después, con el siguiente comando nos conectamos mediante el usuario “**invitado**” al servidor de base de datos, tras lo cual pedirá la contraseña establecida, “**invitado**” también:



```
Procesando disparadores para man-db (2.12.0-4build2) ...
alvucero@alvucero-VirtualBox: ~$ sudo mysql -h 127.0.0.1 -p 3336 -u invitado -p
mysql: [Warning] Using a password on the command line interface can be insecure.
ERROR 2003 (HY000): Can't connect to MySQL server on '127.0.0.1:3306' (111)
alvucero@alvucero-VirtualBox: ~$ mysql -h 127.0.0.1 -p 3336 -u invitado -p
mysql: [Warning] Using a password on the command line interface can be insecure.
ERROR 2003 (HY000): Can't connect to MySQL server on '127.0.0.1:3306' (111)
alvucero@alvucero-VirtualBox: ~$ mysql -h 127.0.0.1 -P 3336 -u invitado -p
Enter password:
ERROR 2003 (HY000): Can't connect to MySQL server on '127.0.0.1:3306' (111)
alvucero@alvucero-VirtualBox: ~$ 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 3
Server version: 11.6.2-MariaDB-ubuntu2404 mariadb.org binary distribution

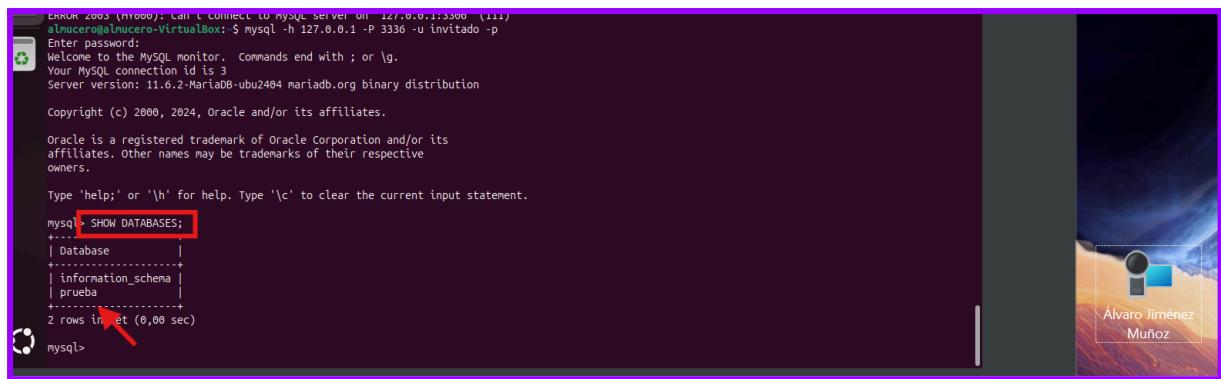
Copyright (c) 2000, 2024, Oracle and/or its affiliates.

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> 
```

Una vez hecho, solo quedaría ejecutar el siguiente comando de SQL para mostrar las bases de datos creadas:



```
ERROR 2003 (HY000): Can't connect to MySQL server on '127.0.0.1:3306' (111)
alvucero@alvucero-VirtualBox: ~$ 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 3
Server version: 11.6.2-MariaDB-ubuntu2404 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 (0,00 sec)

mysql> 
```

Ahí se puede ver claramente cómo la base de datos “**prueba**” está ahí creada.

(Para salir de ahí y poder seguir usando la terminal, sin que sea la de MySQL, se introduce el comando “**exit;**”.)

Actividad 3

En base a los contenedores en ejecución creados en la tarea anterior, se ejecutan las siguientes órdenes de docker cli para obtener tanto la dirección IP de los contenedores en cuestión, así como la redirección de puertos de los mismos:

Para el contenedor “web”, se ejecuta el siguiente comando para mostrar la dirección IP:

```
mysql> exit;
Bye
álmucero@almucero-VirtualBox: $ sudo docker inspect -f '{{range .NetworkSettings.Networks}}{{.IPAddress}}{{end}}' web
172.17.0.3
álmucero@almucero-VirtualBox: $
```

Y este para la redirección de puertos:

```
mysql> exit;
Bye
álmucero@almucero-VirtualBox: $ sudo docker inspect -f '{{range .NetworkSettings.Networks}}{{.IPAddress}}{{end}}' web
172.17.0.3
álmucero@almucero-VirtualBox: $ sudo docker port web
80/tcp -> 0.0.0.0:8181
80/tcp -> [::]:8181
álmucero@almucero-VirtualBox: $
```

Para el contenedor “bbdd” se hace lo mismo:

Dirección IP:

```
mysql> exit;
Bye
álmucero@almucero-VirtualBox: $ sudo docker inspect -f '{{range .NetworkSettings.Networks}}{{.IPAddress}}{{end}}' web
172.17.0.3
álmucero@almucero-VirtualBox: $ sudo docker port web
80/tcp -> 0.0.0.0:8181
80/tcp -> [::]:8181
álmucero@almucero-VirtualBox: $ sudo docker inspect -f '{{range .NetworkSettings.Networks}}{{.IPAddress}}{{end}}' bbdd
172.17.0.2
álmucero@almucero-VirtualBox: $
```

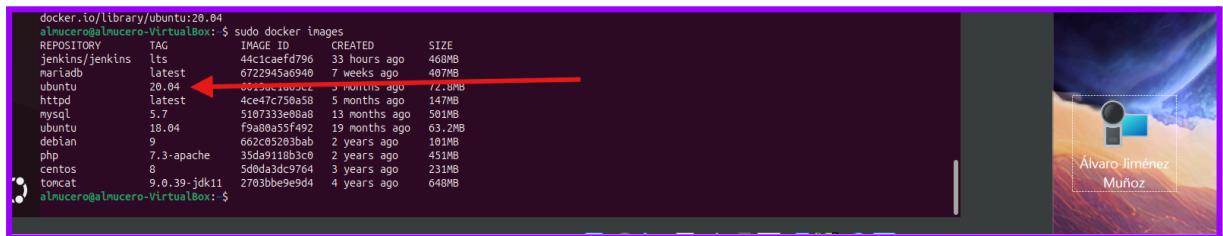
Redirección de puertos:

```
172.17.0.3
álmucero@almucero-VirtualBox: $ sudo docker port web
80/tcp -> 0.0.0.0:8181
80/tcp -> [::]:8181
álmucero@almucero-VirtualBox: $ sudo docker inspect -f '{{range .NetworkSettings.Networks}}{{.IPAddress}}{{end}}' bbdd
172.17.0.2
álmucero@almucero-VirtualBox: $ sudo docker port bbdd
3306/tcp -> 0.0.0.0:3336
3306/tcp -> [::]:3336
álmucero@almucero-VirtualBox: $
```

Lo siguiente será descargar una nueva imagen de Docker Hub, la imagen “Ubuntu:20.04”, de la misma forma que en el ejercicio 1:

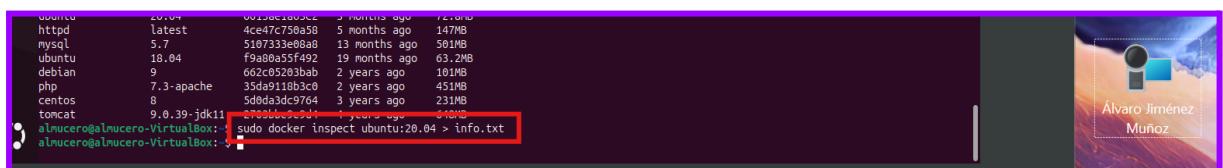
```
álmucero@almucero-VirtualBox: $ sudo docker port bbdd
3306/tcp -> 0.0.0.0:3336
3306/tcp -> [::]:3336
álmucero@almucero-VirtualBox: $ sudo docker pull ubuntu:20.04
20.04: Pulling from library/ubuntu
d9802f032d67: Pull complete
Digest: sha256:8e5c4f0285ecbb4ead078431d29b576a530d3166df73ec44afffc1cd27555141b
Status: Downloaded newer image for ubuntu:20.04
docker.io/library/ubuntu:20.04
álmucero@almucero-VirtualBox: $
```

Se ejecuta el siguiente comando, que permite ver todas las imágenes creadas, y como la flecha indica, ahí está la que se acaba de crear:



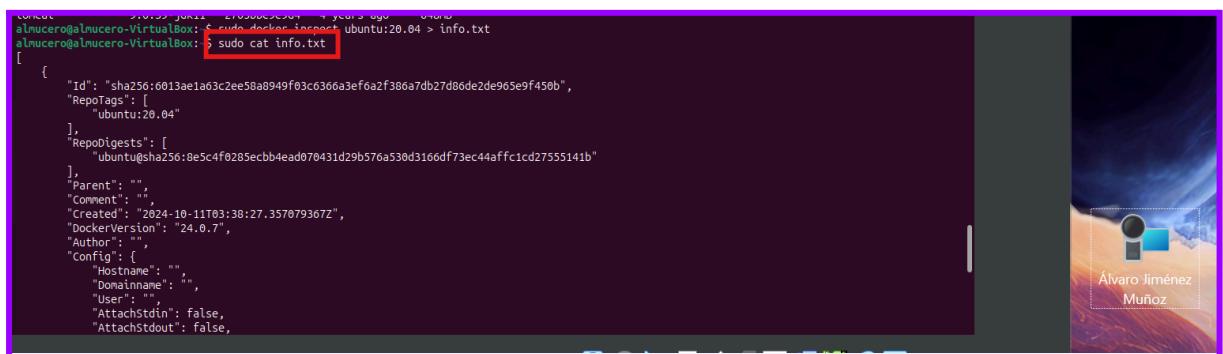
```
docker.io/library/ubuntu:20.04$ sudo docker images
REPOSITORY          TAG        IMAGE ID      CREATED             SIZE
jenkins/jenkins    lts       44c1caeafdf796   33 hours ago    468MB
mariadb            latest    6722945a6940   7 weeks ago     407MB
ubuntu              20.04    0013ae1a63c2...  3 months ago    72.4MB
httpd               latest    4ce47c750a50   5 months ago    147MB
mysql               5.7       5107333e00a0   13 months ago   501MB
ubuntu              18.04    f9a80a5f492...  19 months ago   63.2MB
debian              9         662c05203bab   2 years ago     101MB
php                 7.3-apache 35da9110b3c0   2 years ago     451MB
centos              8         5dd0a3dc9764   3 years ago     231MB
tomcat              9.0.39-jdk11 2703b0ebe9d4   4 years ago     648MB
ubuntu              20.04    0013ae1a63c2...  3 months ago    72.4MB
almucero@almucero-VirtualBox: $
```

Para obtener toda la información de la imagen y volcarla en el fichero “info.txt”, se hace lo siguiente:



```
docker.io/library/ubuntu:20.04$ sudo docker inspect ubuntu:20.04 > info.txt
almucero@almucero-VirtualBox: $ sudo docker inspect ubuntu:20.04 > info.txt
almucero@almucero-VirtualBox: $
```

Usando de nuevo el comando “cat” se ven los contenidos de este fichero, viendo así además que se ha creado correctamente:



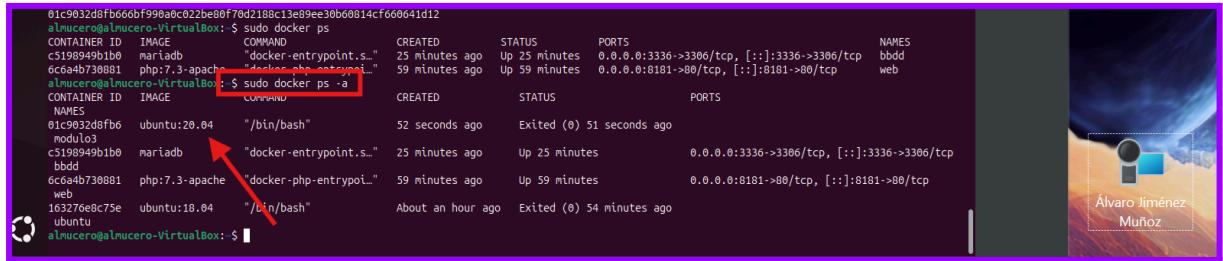
```
cat: info.txt: No such file or directory
almucero@almucero-VirtualBox: $ sudo docker inspect ubuntu:20.04 > info.txt
almucero@almucero-VirtualBox: $ sudo cat info.txt
[{"Id": "sha256:6013ae1a63c2ee50a8949f03c6366a3ef0af386a7db27d86de2de905e9f450b", "RepoTags": ["ubuntu:20.04"], "RepoDigests": ["ubuntu@sha256:8e5c4f0285ecbb4ead070431d29b576a530d3166df73ec44afffc1cd27555141b"], "Parent": "", "Comment": "", "Created": "2024-10-11T03:30:27.357079367Z", "DockerVersion": "24.0.7", "Author": "", "Config": {"Hostname": "", "Domainname": "", "User": "", "AttachStdin": false, "AttachStdout": false, "Env": [], "Label": {}}, "Metadata": {"LastTagTime": "2001-01-01T00:00:00Z"}}
]
```

Después, para instanciar dicha imagen creando un nuevo contenedor llamado “modulo3”, se siguen los siguientes pasos:



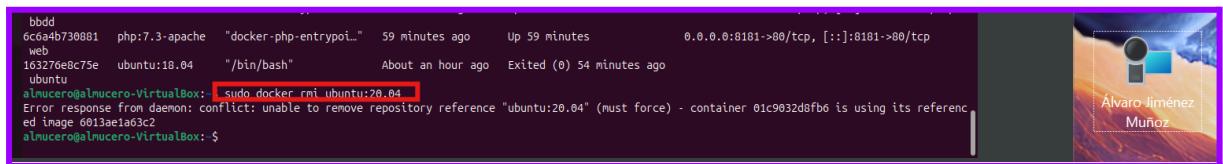
```
], "Metadata": {"LastTagTime": "2001-01-01T00:00:00Z"} }
]
almucero@almucero-VirtualBox: $ sudo docker run -d --name modulo3 ubuntu:20.04
01c9032dfb660bf990a0c022be80f4d2108c13e809e20b60014cf660641d12
almucero@almucero-VirtualBox: $
```

Se comprueba que ha funcionado:



```
01c9032d0fb660bf990a0c022be80f70d2180c13e89ee30b60014cf600641d12
almucero@almucero-VirtualBox: $ sudo docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
c5198949b1b0 mariadb "docker-entrypoint.s..." 25 minutes ago Up 25 minutes 0.0.0.0:3336->3306/tcp, [::]:3336->3306/tcp bbdd
0c6a6db730881 php:7.3-apache "docker-php-entrypoi..." 59 minutes ago Up 59 minutes 0.0.0.0:8101->80/tcp, [::]:8101->80/tcp web
almucero@almucero-VirtualBox: $ sudo docker ps -a
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
NAMES
01c9032d0fb6 ubuntu:20.04 "/bin/bash" 52 seconds ago Exited (0) 51 seconds ago
modulo3
c5198949b1b0 mariadb "docker-entrypoint.s..." 25 minutes ago Up 25 minutes 0.0.0.0:3336->3306/tcp, [::]:3336->3306/tcp bbdd
0c6a6db730881 php:7.3-apache "docker-php-entrypoi..." 59 minutes ago Up 59 minutes 0.0.0.0:8101->80/tcp, [::]:8101->80/tcp web
163276e8c75e ubuntu:18.04 "/bin/bash" About an hour ago Exited (0) 54 minutes ago
ubuntu
almucero@almucero-VirtualBox: $
```

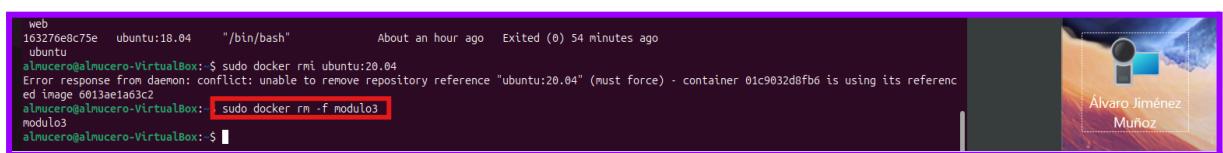
Ahora, si se intenta borrar la imagen recién creada, ocurrirá lo siguiente:



```
bbdd
6c6a6db730881 php:7.3-apache "docker-php-entrypoi..." 59 minutes ago Up 59 minutes 0.0.0.0:8101->80/tcp, [::]:8101->80/tcp
web
163276e8c75e ubuntu:18.04 "/bin/bash" About an hour ago Exited (0) 54 minutes ago
ubuntu
almucero@almucero-VirtualBox: $ sudo docker rmi ubuntu:20.04
Error response from daemon: conflict: unable to remove repository reference "ubuntu:20.04" (must force) - container 01c9032d0fb6 is using its reference
ed image 6013ae1a63c2
almucero@almucero-VirtualBox: $
```

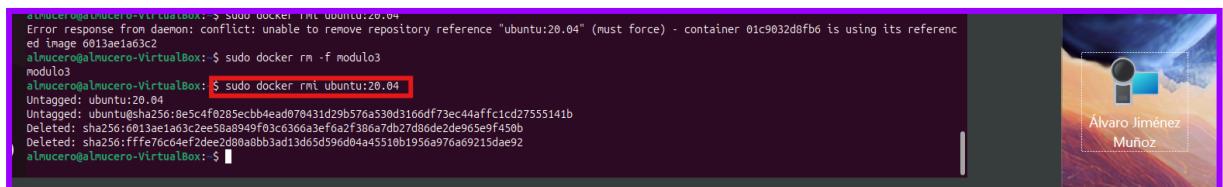
Y es que no se puede borrar ya que está siendo utilizada por algún contenedor, es decir, el contenedor “modulo3” creado previamente.

Por lo que, para que no ocurra esto y se pueda borrar definitivamente la imagen, se deben borrar los contenedores que están utilizando la imagen, en este caso solo “modulo3”:



```
web
163276e8c75e ubuntu:18.04 "/bin/bash" About an hour ago Exited (0) 54 minutes ago
ubuntu
almucero@almucero-VirtualBox: $ sudo docker rmi ubuntu:20.04
Error response from daemon: conflict: unable to remove repository reference "ubuntu:20.04" (must force) - container 01c9032d0fb6 is using its reference
ed image 6013ae1a63c2
almucero@almucero-VirtualBox: $ sudo docker rm -f modulo3
modulo3
almucero@almucero-VirtualBox: $
```

Y volver a ejecutar el comando anterior, que al ya no existir “modulo3”, no volverá a dar fallo:



```
almucero@almucero-VirtualBox: $ sudo docker rmi ubuntu:20.04
Error response from daemon: conflict: unable to remove repository reference "ubuntu:20.04" (must force) - container 01c9032d0fb6 is using its reference
ed image 6013ae1a63c2
almucero@almucero-VirtualBox: $ sudo docker rm -f modulo3
modulo3
almucero@almucero-VirtualBox: $ sudo docker rmi ubuntu:20.04
Untagged: ubuntu:20.04
Deleted: sha256:0b13ae1a63c2ee58a8949f03c366a3ef6a2f386a7db27d80de2de965e9f450b
Deleted: sha256:ffff7cc044ef2dec2d80aabbb3a13d65d599dd04a4510b1956a970a69215dae92
almucero@almucero-VirtualBox: $
```

Se comprueba que verdaderamente se ha borrado:

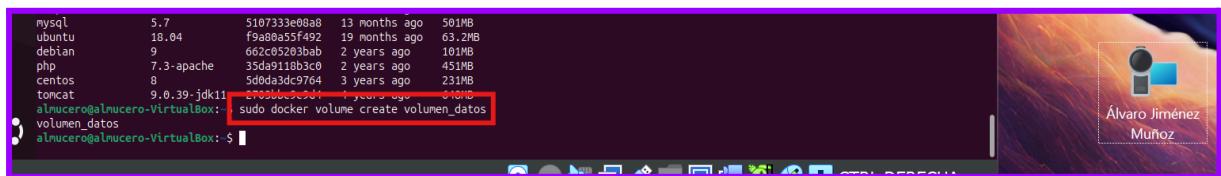


```
163276e8c75e  ubuntu:18.04  "/bin/bash"          About an hour ago  Exited (0) 57 minutes ago
ubuntu
almucero@almucero-VirtualBox: ~$ sudo docker images
REPOSITORY      TAG      IMAGE ID      CREATED        SIZE
jenkins/jenkins  lts      44c1caeafdf96  33 hours ago  468MB
mariadb         latest   6722945a6940  7 weeks ago   407MB
httdp           latest   4ce47c758a58  5 months ago  147MB
mysql            5.7     5107333e90a8  13 months ago  501MB
ubuntu           18.04   f9a80a5f492   19 months ago  63.2MB
debian           9       662c05203bab  2 years ago   101MB
php              7.3-apache 35da911b03c0  2 years ago   451MB
centos          8       5d0da3dc9764  3 years ago   231MB
tomcat           9.0.39-jdk11 2703bbe9e9d4  4 years ago   640MB
almucero@almucero-VirtualBox: ~$
```

Actividad 4

Lo primero en esta actividad será crear los volúmenes “volumen_datos” y “volumen_web”; para ello se hace lo siguiente:

Para “volumen_datos”:



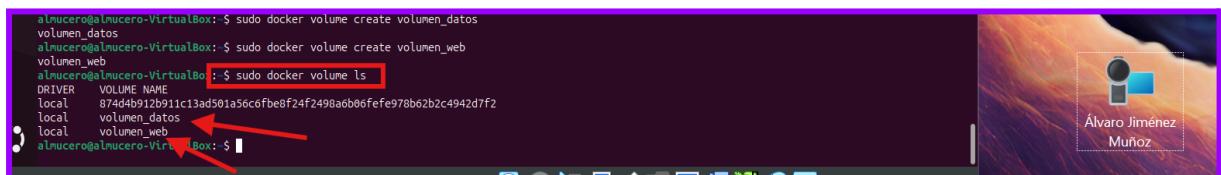
```
mysql      5.7      5107333e90a8  13 months ago  501MB
ubuntu    18.04   f9a80a5f492   19 months ago  63.2MB
debian     9       662c05203bab  2 years ago   101MB
php        7.3-apache 35da911b03c0  2 years ago   451MB
centos    8       5d0da3dc9764  3 years ago   231MB
tomcat    9.0.39-jdk11 2703bbe9e9d4  4 years ago   640MB
almucero@almucero-VirtualBox: ~$ sudo docker volume create volumen_datos
volumen_datos
almucero@almucero-VirtualBox: ~$
```

Y para “volumen_web”:



```
debian     9       662c05203bab  2 years ago   101MB
php        7.3-apache 35da911b03c0  2 years ago   451MB
centos    8       5d0da3dc9764  3 years ago   231MB
tomcat    9.0.39-jdk11 2703bbe9e9d4  4 years ago   640MB
almucero@almucero-VirtualBox: ~$ sudo docker volume create volumen_datos
volumen_datos
almucero@almucero-VirtualBox: ~$ sudo docker volume create volumen_web
volumen_web
almucero@almucero-VirtualBox: ~$
```

Se comprueba que correctamente se han creado ambos volúmenes:



```
almucero@almucero-VirtualBox: ~$ sudo docker volume create volumen_datos
volumen_datos
almucero@almucero-VirtualBox: ~$ sudo docker volume create volumen_web
volumen_web
almucero@almucero-VirtualBox: ~$ sudo docker volume ls
DRIVER    VOLUME NAME
local     874d4bb912b911c13ad501a56cc6fbe8f24f2498a6b06fe978b62b2c494d7f2
local     volumen_datos
local     volumen_web
almucero@almucero-VirtualBox: ~$
```

Una vez creados estos contenedores, se arranca otro llamado “c1” sobre la imagen “php:7.4-apache”, que monte “volumen_web” en la ruta “/var/www/html”; para ello se hace:

(Al no estar descargada la imagen en cuestión, esta se descargará automáticamente al introducir el comando antes de hacer lo que en él se especifica).

```
local ~ volumen_web
almucero@almucero-VirtualBox: ~$ sudo docker run -d --name c1 -v volumen_web:/var/www/html php:7.4-apache
Unable to find image "php:7.4-apache": locally
7.4-apache: Pulling from library/php
a0e03f5ae3b4: Pull complete
c420f1a49423: Pull complete
156748bb7fe8: Pull complete
fb5a4dc8fb2f: Pull complete
25f85b498fd5: Pull complete
9b233e420a07: Pull complete
fe42347c4ecf: Pull complete
d14eb2ed1e17: Pull complete
6dd98f73ac6b: Pull complete
d2c43c5efcb8: Pull complete
ab590d40e47: Pull complete
80692zaed667: Pull complete
05e405aa9a99: Pull complete
Digest: sha256:c5d7e00877382673479770d06aacc8100011ec751d1905ff63fae3fe2e0ca6d
Status: Downloaded newer image for php:7.4-apache
a43530e569961666ed2e145e0e2b55324e80fae2be083073370315ea41
almucero@almucero-VirtualBox: ~$
```

Se comprueba que ha funcionado:

80692ae2d067: Pull complete
05e465aa99a: Pull complete
Digest: sha256:c9d7e008f73832673479770d66aacc8100011ec751d1905ff03fae3fe2e0ca6d
Status: Downloaded newer image for php:7.4-apache
b43530e56998a1666edc2e540e2eb55324e80afae2b2e03873370315ea41
alnucero@alnucero-VirtualBox: ~ \$ sudo docker ps

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
b43530e56998	php:7.4-apache	"docker-php-entrypoint"	26 seconds ago	Up 24 seconds	80/tcp	c1
c5198949b1b0	mariaadb	"docker-entrypoints-s..."	42 minutes ago	Up 42 minutes	0.0.0.0:3336->3306/tcp, [::]:3336->3306/tcp	bbdd
6c0a4d730881	php:7.3-apache	"docker-php-entrypoint"	About an hour ago	Up About an hour	0.0.0.0:8181->80/tcp, [::]:8181->80/tcp	web

alnucero@alnucero-VirtualBox: ~ \$

A red arrow points from the top right towards the "NAMES" column of the Docker ps output table.

Después, para arrancar otro contenedor bajo el nombre ahora “c2” sobre la imagen “mariadb”, esta vez sí descargada de antemano, que se monte “volumen_datos” en la ruta “/var/lib/mysql” y con contraseña de root “admin”, se hace lo siguiente (lo mismo que antes):

```
--tlskey string      Path to TLS key file (default "/root/.docker/key.pem")
--tlsversion int     Use TLS and verify the remote
-v, --version        Print version information and quit

Run 'docker COMMAND --help' for more information on a command.

For more help on how to use Docker, head to https://docs.docker.com/go/guides/

alucero@alucero-VirtualBox: ~$ sudo docker run -d --name c2 -v volumen_datos:/var/lib/mysql -e MYSQL_ROOT_PASSWORD=admin mariadb
a52883fcbb0d868a1404ccbf99149c6f150c05000000751a2f73...159870
alucero@alucero-VirtualBox: ~$
```

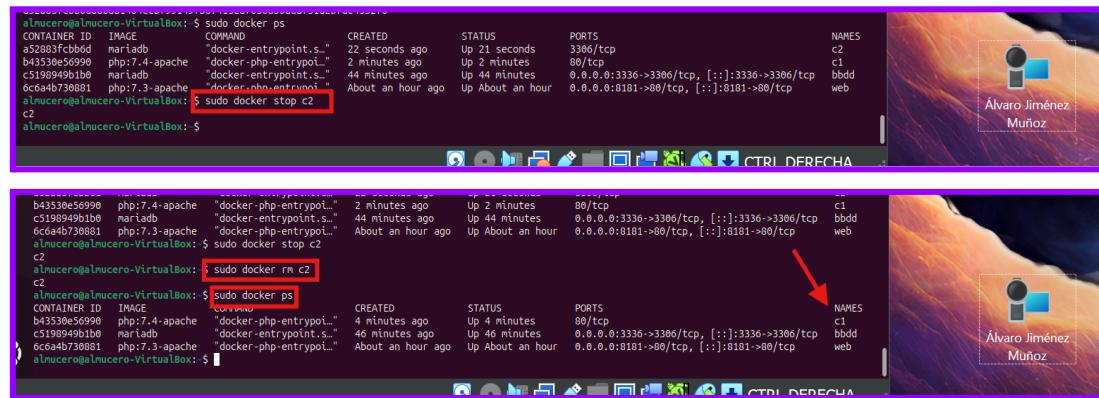
De nuevo se comprueba si funcionó:

```
sun 'docker COMMAND --help' for more information on a command.  
or more help on how to use Docker, head to https://docs.docker.com/go/guides/  
  
almucero@almucero-VirtualBox: ~$ sudo docker run -d --name c2 -v volumen_datos:/var/lib/mysql -e MYSQL_ROOT_PASSWORD=admin mariadb  
52883fcb6bdd8ea81404ccb991497bfaf13caf036836aa751a2bdf4e532f0  
almucero@almucero-VirtualBox: ~$ sudo docker ps  


| CONTAINER ID | IMAGE         | COMMAND                  | CREATED           | STATUS           | PORTS                                       | NAMES |
|--------------|---------------|--------------------------|-------------------|------------------|---------------------------------------------|-------|
| 52883fcb6bdd | mariadb       | "docker-entrypoint.s..." | 22 seconds ago    | Up 21 seconds    | 3306/tcp                                    | c2    |
| 43530e56990  | php7.4-apache | "docker-php-entrypi..."  | 2 minutes ago     | Up 2 minutes     | 80/tcp                                      | c1    |
| 516949461b6  | mariadb       | "docker-entrypoint.s..." | 44 minutes ago    | Up 44 minutes    | 0.0.0.0:3336->3306/tcp, [::]:3336->3306/tcp | bddd  |
| 5c04ab730881 | php7.3-apache | "docker-php-entrypoi..." | About an hour ago | Up About an hour | 0.0.0.0:8181->80/tcp, [::]:8181->80/tcp     | web   |

  
Álvaro Jiménez  
Muñoz
```

Luego, para borrar el volumen “volumen_datos”, se debe, al igual que antes, parar y borrar previamente el contenedor “c2”:



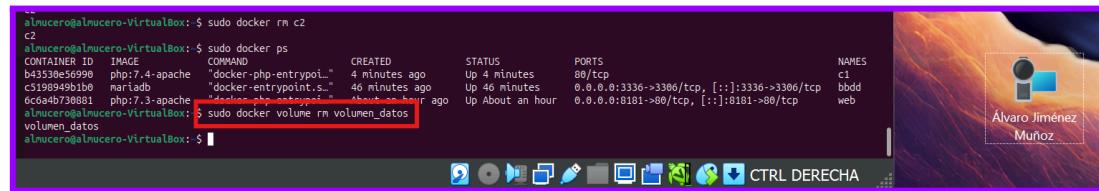
```

almucero@almucero-VirtualBox:~$ sudo docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
b43530e56990 php:7.4-apache "docker-php-entrypoint..." 22 seconds ago Up 21 seconds 3306/tcp c2
c5198949b1b0 mariadb "docker-entrypoint.s..." 2 minutes ago Up 2 minutes 0.0.0.0:3306->3306/tcp, [::]:3336->3306/tcp, [::]:8010->80/tcp c1
5198949b1b0 php:7.4-apache "docker-php-entrypoint..." 44 minutes ago Up 44 minutes 0.0.0.0:3336->3306/tcp, [::]:3336->3306/tcp, [::]:8010->80/tcp bbdd
6c6a4b730881 php:7.3-apache "docker-php-entrypoint..." About an hour ago Up About an hour 0.0.0.0:8181->80/tcp, [::]:8181->80/tcp web
almucero@almucero-VirtualBox:~$ sudo docker stop c2
c2
almucero@almucero-VirtualBox:~$ sudo docker rm c2
c2
almucero@almucero-VirtualBox:~$ sudo docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
b43530e56990 php:7.4-apache "docker-php-entrypoint..." 4 minutes ago Up 4 minutes 3306/tcp c1
c5198949b1b0 mariadb "docker-entrypoint.s..." 46 minutes ago Up 46 minutes 0.0.0.0:3336->3306/tcp, [::]:3336->3306/tcp, [::]:8010->80/tcp bbdd
6c6a4b730881 php:7.3-apache "docker-php-entrypoint..." About an hour ago Up About an hour 0.0.0.0:8181->80/tcp, [::]:8181->80/tcp web
almucero@almucero-VirtualBox:~$ 

```

Ahí se ve como el contenedor “c2” ya no está, quedando solo “c1”.

Luego, se borra el volumen:

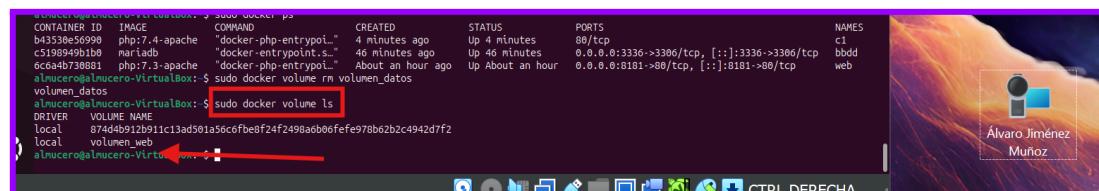


```

almucero@almucero-VirtualBox:~$ sudo docker rm c2
c2
almucero@almucero-VirtualBox:~$ sudo docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
b43530e56990 php:7.4-apache "docker-php-entrypoint..." 4 minutes ago Up 4 minutes 3306/tcp c1
c5198949b1b0 mariadb "docker-entrypoint.s..." 46 minutes ago Up 46 minutes 0.0.0.0:3336->3306/tcp, [::]:3336->3306/tcp, [::]:8010->80/tcp bbdd
6c6a4b730881 php:7.3-apache "docker-php-entrypoint..." About an hour ago Up About an hour 0.0.0.0:8181->80/tcp, [::]:8181->80/tcp web
almucero@almucero-VirtualBox:~$ sudo docker volume rm volumen_datos
volumen_datos
almucero@almucero-VirtualBox:~$ 

```

Se comprueba que efectivamente se ha borrado y solo queda el otro volumen creado:

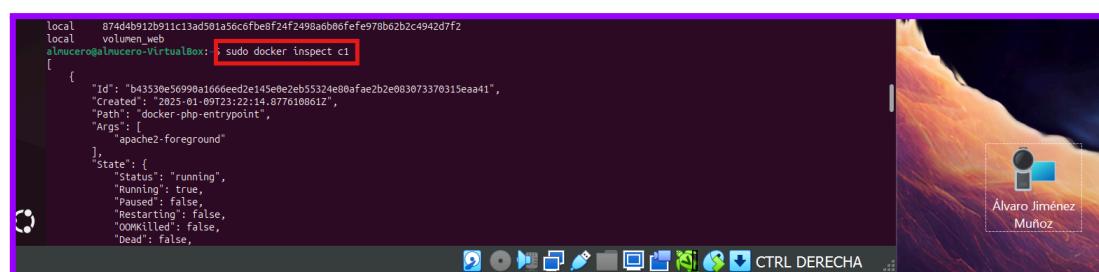


```

almucero@almucero-VirtualBox:~$ sudo docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
b43530e56990 php:7.4-apache "docker-php-entrypoint..." 4 minutes ago Up 4 minutes 3306/tcp c1
c5198949b1b0 mariadb "docker-entrypoint.s..." 46 minutes ago Up 46 minutes 0.0.0.0:3336->3306/tcp, [::]:3336->3306/tcp, [::]:8010->80/tcp bbdd
6c6a4b730881 php:7.3-apache "docker-php-entrypoint..." About an hour ago Up About an hour 0.0.0.0:8181->80/tcp, [::]:8181->80/tcp web
almucero@almucero-VirtualBox:~$ sudo docker volume ls
DRIVER VOLUME NAME
local 874d4b912b911c13ad501a56c6fbefbf24f2498aa0b06feff978be2b2c4942d7f2
local volumen_web
almucero@almucero-VirtualBox:~$ 

```

Ya para finalizar la actividad, se usa “docker inspect” para verificar que el contenedor “c1” usa “volumen_web”:

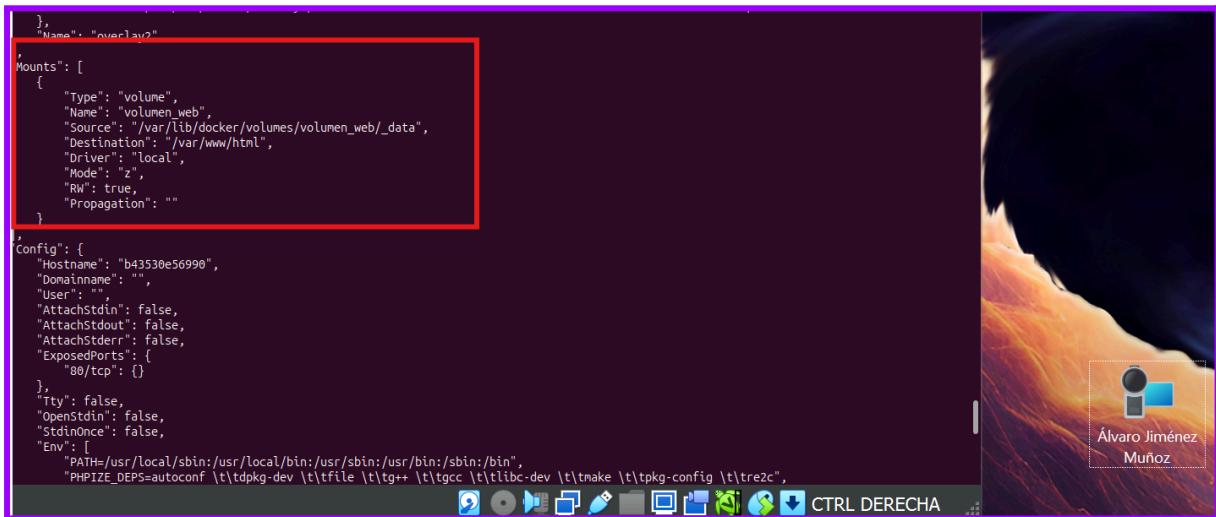


```

local 874d4b912b911c13ad501a56c6fbefbf24f2498aa0b06feff978be2b2c4942d7f2
local volumen_web
almucero@almucero-VirtualBox:~$ sudo docker inspect c1
[
  {
    "Id": "b43530e56990a166eed2e145e0e2eb5324e80afae2b2e083073370315eaa41",
    "Created": "2025-01-09T23:22:14.876108617Z",
    "Path": "docker-php-entrypoint",
    "Args": [
      "apache2-foreground"
    ],
    "State": {
      "Status": "running",
      "Running": true,
      "Paused": false,
      "Restarting": false,
      "OOMKilled": false,
      "Dead": false,
      "Pid": 1000
    },
    "NetworkSettings": {
      "Bridge": "bridge",
      "ContainerID": "b43530e56990a166eed2e145e0e2eb5324e80afae2b2e083073370315eaa41",
      "IPAddress": "172.17.0.2",
      "IPPrefixLen": 16,
      "MacAddress": "0A:0A:0A:0A:0A:0A",
      "Gateway": "172.17.0.1"
    }
  }
]

```

Se navega hasta la sección de “Mounts”, donde se aprecia que está siendo usado el volumen:

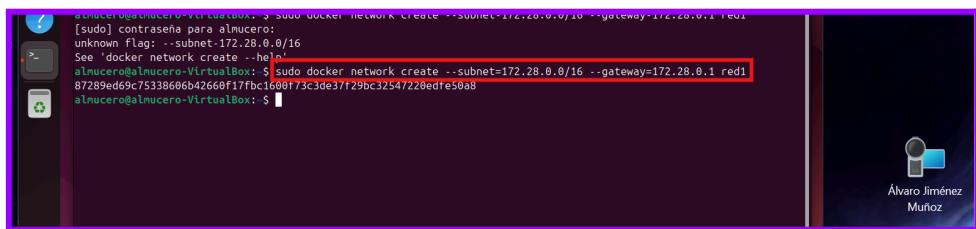


```
        },
        "Name": "overlay2"
    },
    "Mounts": [
        {
            "Type": "volume",
            "Name": "volumen_web",
            "Source": "/var/lib/docker/volumes/volumen_web/_data",
            "Destination": "/var/www/html",
            "Driver": "local",
            "Mode": "z",
            "RW": true,
            "Propagation": ""
        }
    ],
    "Config": {
        "Hostname": "b43530e56990",
        "Domainname": "",
        "User": "",
        "AttachStdin": false,
        "AttachStdout": false,
        "AttachStderr": false,
        "ExposedPorts": {
            "80/tcp": {}
        },
        "Tty": false,
        "OpenStdin": false,
        "StdinOnce": false,
        "Env": [
            "PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin",
            "PHPIZE_DEPS=autoconf \t\tdpkg-dev \t\tfile \t\tg++ \t\tgcc \t\tmake \t\tpkg-config \t\tre2c",
            "ALMUCERO=almucero"
        ]
    }
}
```

Actividad 5

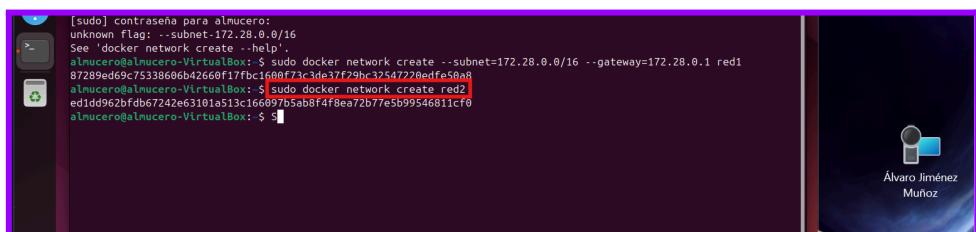
Para crear las redes, ambas en modo **bridged**, el modo por defecto, por lo que no es necesario especificar nada, se debe hacer lo siguiente con ambas, partiendo ya con todo lo anterior hecho e instalado:

“red1”, asignando valores:



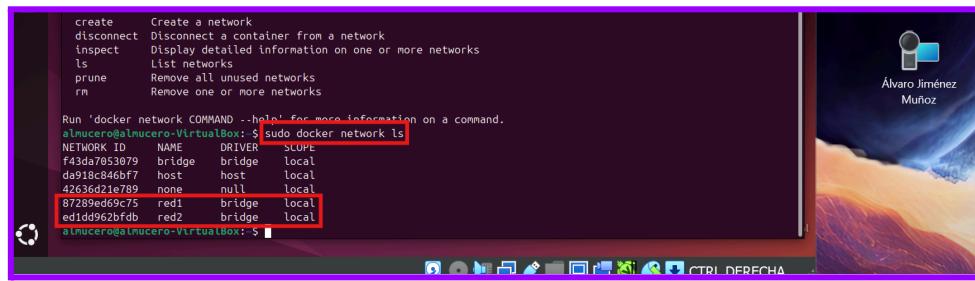
```
almucero@almucero-VirtualBox: ~ $ sudo docker network create --subnet=172.28.0.0/16 --gateway=172.28.0.1 red1
[sudo] contraseña para almucero:
unknown flag: --subnet=172.28.0.0/16
See 'docker network create -h'.
almucero@almucero-VirtualBox: ~ $ sudo docker network create --subnet=172.28.0.0/16 --gateway=172.28.0.1 red1
87289ed69c75338606b42660f17fb1600f73c3de3f729bc32547220edfe50a8
almucero@almucero-VirtualBox: ~ $
```

“red2”, valores asignados automáticamente por Docker:



```
almucero@almucero-VirtualBox: ~ $ sudo docker network create red2
87289ed69c75338606b42660f17fb1600f73c3de3f729bc32547220edfe50a8
almucero@almucero-VirtualBox: ~ $ sudo docker network create red2
e1dd962bfb67242e63101a513c166997b5ab8f4f8ea72b77e5b99546811cf0
almucero@almucero-VirtualBox: ~ $
```

Usando el siguiente comando se puede ver cómo ambas redes han sido creadas correctamente:

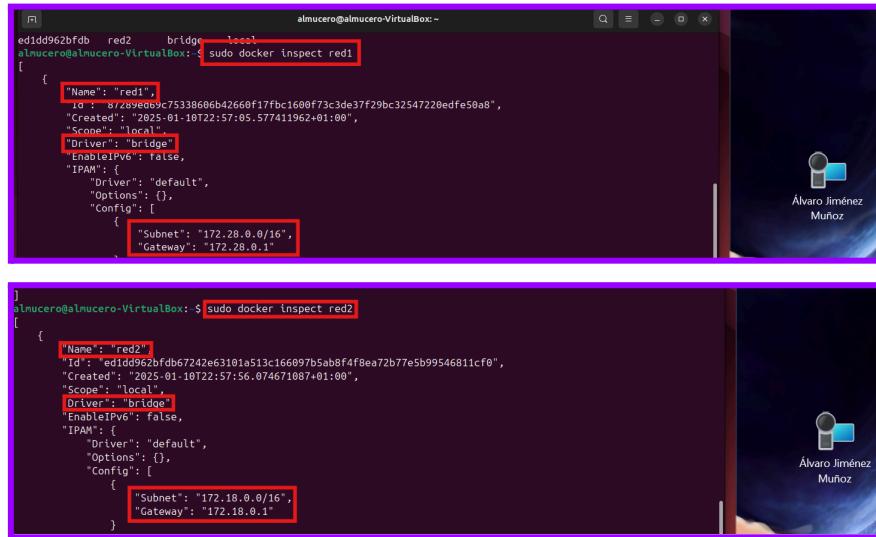


```
create      Create a network
disconnect  Disconnect a container from a network
inspect     Display detailed information on one or more networks
ls          List networks
prune      Remove all unused networks
rm          Remove one or more networks

Run 'docker network COMMAND --help' for more information on a command.

almucero@almucero-VirtualBox:~$ sudo docker network ls
NETWORK ID     NAME      DRIVER    SCOPE
f43da7653079   bridge    bridge    local
da918cd8bf7   host      host      local
42636d21e789   none      null     local
87289ed69c75   red1      bridge    local
ed1dd962fdb   red2      bridge    local
almucero@almucero-VirtualBox:~$
```

Y usando la orden “**inspect**” se puede ver si, aparte de haber sido creadas, también han sido configuradas como se ha indicado previamente:

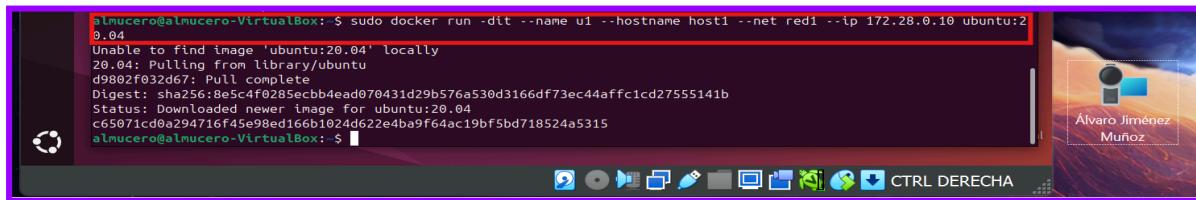


```
almucero@almucero-VirtualBox:~$ sudo docker inspect red1
[{"Name": "red1", "Id": "ed1dd962fdb67242e63101a513c166097b5ab8f4f8ea72b77e5b99546811cf0", "Created": "2015-01-10T22:57:05.577411962+01:00", "Scope": "local", "Driver": "bridge", "EnableIPv6": false, "IPAM": {"Driver": "default", "Options": {}, "Config": [{"Subnet": "172.28.0.0/16", "Gateway": "172.28.0.1"}]}}

almucero@almucero-VirtualBox:~$ sudo docker inspect red2
[{"Name": "red2", "Id": "ed1dd962fdb67242e63101a513c166097b5ab8f4f8ea72b77e5b99546811cf0", "Created": "2015-01-10T22:57:05.074671087+01:00", "Scope": "local", "Driver": "bridge", "EnableIPv6": false, "IPAM": {"Driver": "default", "Options": {}, "Config": [{"Subnet": "172.18.0.0/16", "Gateway": "172.18.0.1"}]}}
```

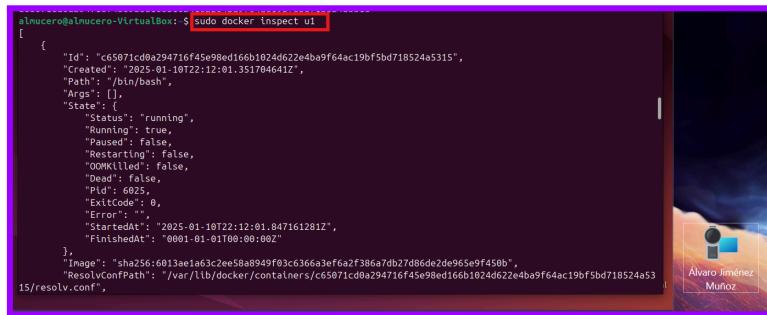
En este segundo caso se puede ver cómo los valores asignados a la red en cuanto a dirección de red, máscara y gateway han sido establecidos automáticamente por Docker de manera aleatoria.

Después, para poner en ejecución un contenedor con hostname “**host1**”, IP “**172.20.0.10**” y conectado a “**red1**” de la imagen “**ubuntu:20.04**” bajo el nombre “**u1**”, se hace lo siguiente:



```
almucero@almucero-VirtualBox:~$ sudo docker run -dit --name u1 --hostname host1 --net red1 --ip 172.20.0.10 ubuntu:20.04
Unable to find image 'ubuntu:20.04' locally
20.04: Pulling from library/ubuntu
d9802f032d67: Pull complete
Digest: sha256:8e5c4f0295ecbb4ead070431d29b576a530d3166df73ec44afffc1cd27555141b
Status: Downloaded newer image for ubuntu:20.04
c65071cd0a294716f45e98ed166b1024d622e4ba9f64ac19bf5bd718524a5315
almucero@almucero-VirtualBox:~$
```

Para ver la configuración de red de este contenedor creado, de nuevo, se usa “inspect”:



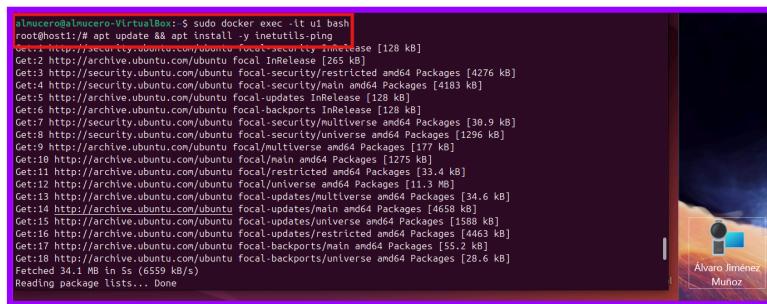
```
almucero@almucero-VirtualBox:~$ sudo docker inspect ui
[{"Id": "c65071cd0a294716f45e980d166b1024d622e4ba9f64ac19bf5bd718524a5315", "Created": "2025-01-10T22:12:01.351704641Z", "Path": "/bin/bash", "Args": [], "State": {"Status": "running", "Running": true, "Paused": false, "Restarting": false, "OOMKilled": false, "Dead": false, "Pid": 6025, "ExitCode": 0, "Error": ""}, "StartedAt": "2025-01-10T22:12:01.847161281Z", "FinishedAt": "2025-01-10T08:00:00Z", "Image": "sha256:6013ae1a63c2ee58a8949f03c6366a3ef6a2f386a7db27d86de2de965e9f450b", "ResolvePath": "/var/lib/docker/containers/c65071cd0a294716f45e980d166b1024d622e4ba9f64ac19bf5bd718524a5315/resolv.conf", "HostConfig": {}}]
```

Usar este comando con el contenedor dará mucha información, y entre ella también se puede encontrar la configuración de red del contenedor, donde se puede ver toda la información acerca de la red creada previamente a la que está conectado:



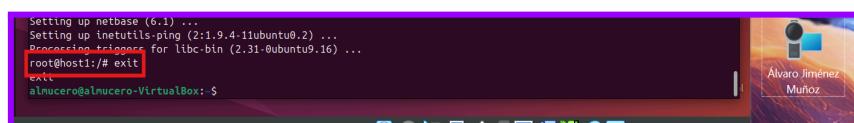
```
"MacAddress": "", "Networks": { "red1": { "IPAMConfig": { "IPv4Address": "172.28.0.10" }, "Links": null, "MacAddress": "02:42:ac:1c:00:0a", "DriverOpts": null, "NetworkID": "97389ed69c753396b42660f17fb1600f73c3de37f29bc32547220edfe50a8", "EndpointID": "adfd3d80d1ac099cd2a86cb62a81d0eda22f4986304006c05a96462fc088ce9d2", "Gateway": "172.28.0.1", "IPAddress": "172.28.0.10", "IPPrefixLen": 16, "IPv6Gateway": "", "GlobalIPv6Address": "", "GlobalIPv6PrefixLen": 0, "DNSNames": [ "ui", "c65071cd0a29", "host1" ] } } }
```

Una vez hecho eso, para entrar en el contenedor y preparar el entorno instalando la aplicación “ping”, se utilizan los siguientes comandos:



```
almucero@almucero-VirtualBox:~$ sudo docker exec -it ui bash
root@host1:/# apt update & apt install -y inetutils-ping
Get:1 http://archive.ubuntu.com/ubuntu focal InRelease [265 kB]
Get:2 http://archive.ubuntu.com/ubuntu focal-security/restricted amd64 Packages [4276 kB]
Get:3 http://security.ubuntu.com/ubuntu focal-security/main amd64 Packages [4183 kB]
Get:5 http://archive.ubuntu.com/ubuntu focal-updates InRelease [128 kB]
Get:6 http://archive.ubuntu.com/ubuntu focal-backports InRelease [128 kB]
Get:7 http://security.ubuntu.com/ubuntu focal-security/multiverse amd64 Packages [30.9 kB]
Get:8 http://archive.ubuntu.com/ubuntu focal-security/universe amd64 Packages [1296 kB]
Get:9 http://archive.ubuntu.com/ubuntu focal/multiverse amd64 Packages [177 kB]
Get:10 http://archive.ubuntu.com/ubuntu focal/universe amd64 Packages [11.3 MB]
Get:11 http://archive.ubuntu.com/ubuntu focal/restricted amd64 Packages [33.4 kB]
Get:12 http://archive.ubuntu.com/ubuntu focal/universe amd64 Packages [11.3 MB]
Get:13 http://archive.ubuntu.com/ubuntu focal-updates/multiverse amd64 Packages [34.6 kB]
Get:14 http://archive.ubuntu.com/ubuntu focal-updates/main amd64 Packages [4658 kB]
Get:15 http://archive.ubuntu.com/ubuntu focal-updates/universe amd64 Packages [1588 kB]
Get:16 http://archive.ubuntu.com/ubuntu focal-updates/restricted amd64 Packages [4463 kB]
Get:17 http://archive.ubuntu.com/ubuntu focal-backports/main amd64 Packages [55.2 kB]
Get:18 http://archive.ubuntu.com/ubuntu focal-backports/universe amd64 Packages [28.6 kB]
Fetched 54.1 kB in 5s (6559 kB/s)
Reading package lists... Done
```

Para salir del mismo se hace lo siguiente:



```
Setting up netbase (6.1) ...
Setting up inetutils-ping (2:1.9.4-11ubuntu0.2) ...
Processing triggers for libc-bin (2.31-0ubuntu10.16) ...
root@host1:/# exit
exit
almucero@almucero-VirtualBox:~$
```

Ahora, para crear un nuevo contenedor con nombre “u2” de la misma imagen con hostname “host2” y conectado a “red2”, con la diferencia de que este tendrá la dirección IP de la red elegida por Docker, se siguen los siguientes pasos, muy similares a los anteriores:

```
exit  
almucero@almucero-VirtualBox:~$ sudo docker run -dit --name u2 --hostname host2 --net red2 ubuntu:20.04  
f5a83536a1a8322c8accee40457651c38892a3310ffef9b5728af6f9a3e41a10  
almucero@almucero-VirtualBox:~$
```

Álvaro Jiménez
Muñoz

Se comprueba su configuración de red de la misma forma:

```
almucero@almucero-VirtualBox:~$ sudo docker run -dit --name u2 --hostname host2 --net red2 ubuntu:20.04  
f5a83536a1a8322c8accee40457651c38892a3310ffef9b5728af6f9a3e41a10  
almucero@almucero-VirtualBox:~$ sudo docker inspect u2  
[  
  {  
    "Id": "f5a83536a1a8322c8accee40457651c38892a3310ffef9b5728af6f9a3e41a10",  
    "Created": "2025-01-10T22:24:16.228838124Z",  
    "Path": "/bin/bash",  
    "Args": [],  
    "State": {  
      "Status": "running",  
      "Running": true,  
      "Paused": false,  
      "Restarting": false,  
      "OOMKilled": false,  
      "Dead": false,  
      "Pid": 6449,  
      "ExitCode": 0,  
      "Error": "",  
      "StartedAt": "2025-01-10T22:24:16.511844059Z",  
    },  
    "Networks": {  
      "red2": {  
        "IPAMConfig": null,  
        "Links": null,  
        "Aliases": null,  
        "MacAddress": "02:42:ac:12:00:02",  
        "DriverOpts": null,  
        "NetworkID": "ed1dd962bfdb67242e63101a513c166097b5ab8f4f8ea72b77e5b99546811cf0",  
        "EndpointID": "4c974e06d34862a5436be68ee7eb23c12c7004117fb65fe14ddc63f3cad99a4",  
        "Gateway": "172.18.0.1",  
        "IPAddress": "172.18.0.2",  
        "IPPrefixLen": 16,  
        "IPv6Gateway": "",  
        "GlobalIPv6Address": "",  
        "GlobalIPv6PrefixLen": 0,  
        "DNSNames": [  
          "u2",  
          "f5a83536a1a8",  
          "host2"  
        ]  
      }  
    }  
  }]
```

```
'Networks': {  
  "red2": {  
    "IPAMConfig": null,  
    "Links": null,  
    "Aliases": null,  
    "MacAddress": "02:42:ac:12:00:02",  
    "DriverOpts": null,  
    "NetworkID": "ed1dd962bfdb67242e63101a513c166097b5ab8f4f8ea72b77e5b99546811cf0",  
    "EndpointID": "4c974e06d34862a5436be68ee7eb23c12c7004117fb65fe14ddc63f3cad99a4",  
    "Gateway": "172.18.0.1",  
    "IPAddress": "172.18.0.2",  
    "IPPrefixLen": 16,  
    "IPv6Gateway": "",  
    "GlobalIPv6Address": "",  
    "GlobalIPv6PrefixLen": 0,  
    "DNSNames": [  
      "u2",  
      "f5a83536a1a8",  
      "host2"  
    ]  
  }  
}
```



Se instala “ping”, de nuevo, de la misma manera:

```
For more help on how to use Docker, head to https://docs.docker.com/go/guides/  
almucero@almucero-VirtualBox:~$ sudo docker exec -it u2 bash  
root@host2:/# apt update && apt install -y inetutils-ping  
Get:1 http://archive.ubuntu.com/ubuntu focal-security InRelease [128 kB]  
Get:2 http://archive.ubuntu.com/ubuntu focal-security InRelease [128 kB]  
Get:3 http://archive.ubuntu.com/ubuntu focal-updates InRelease [128 kB]  
Get:4 http://archive.ubuntu.com/ubuntu focal-backports InRelease [128 kB]
```

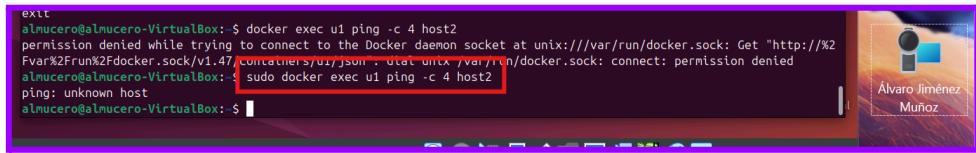


Y ya por último se sale nuevamente:

```
Setting up libltdl11:amd64 (1.3.3-2.2ubuntu2) ...  
Setting up netbase (6.1) ...  
Setting up inetutils-ping (2:1.9.4-11ubuntu0.2) ...  
Processing triggers for libc-bin (2.31-0ubuntu9.16) ...  
root@host2:/# exit  
exit  
almucero@almucero-VirtualBox:~$
```

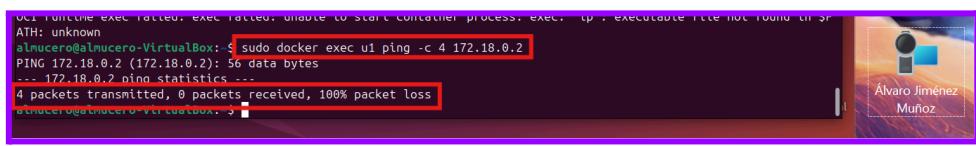


Habiendo hecho ya todo esto, debido a los parámetros establecidos durante los procesos anteriores, se puede observar cómo desde ninguno de los 2 contenedores se puede hacer ping al otro:



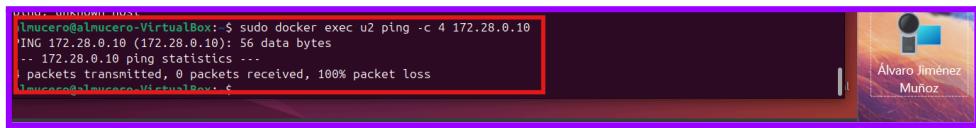
```
ext
almucero@almucero-VirtualBox: $ docker exec u1 ping -c 4 host2
permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock: Get "http://%2Fvar%2Frun%2Fdocker.sock/v1.47/containers/u1/json": dial unix /var/run/docker.sock: connect: permission denied
almucero@almucero-VirtualBox: $ sudo docker exec u1 ping -c 4 host2
ping: unknown host
almucero@almucero-VirtualBox: $
```

En ese caso puede verse como desde el contenedor “u1” no es posible hacer ping a “u2” mediante el nombre, pero como se ve a continuación, tampoco se puede mediante dirección IP:



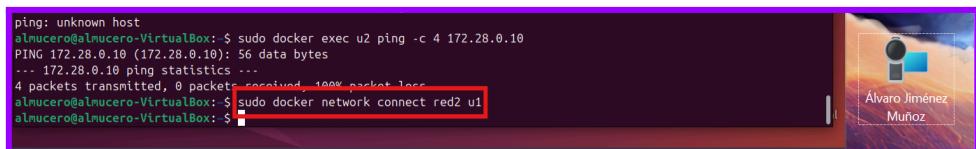
```
OCI runtime exec failed: exec failed: unable to start container process: exec: "ping": executable file not found in $PATH
almucero@almucero-VirtualBox: $ sudo docker exec u1 ping -c 4 172.18.0.2
PING 172.18.0.2 (172.18.0.2): 56 data bytes
... 172.18.0.2 ping statistics ...
4 packets transmitted, 0 packets received, 100% packet loss
almucero@almucero-VirtualBox: $
```

Lo mismo ocurre en la dirección opuesta:



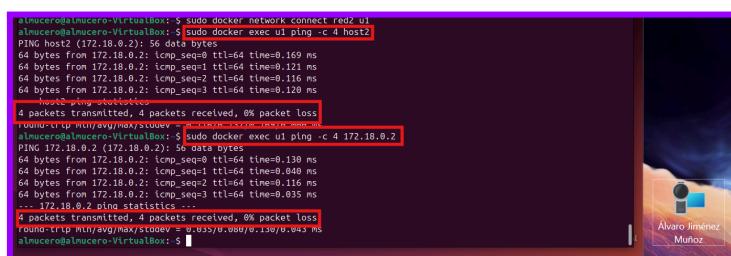
```
almucero@almucero-VirtualBox: $ sudo docker exec u2 ping -c 4 172.28.0.10
PING 172.28.0.10 (172.28.0.10): 56 data bytes
... 172.28.0.10 ping statistics ...
packets transmitted, 0 packets received, 100% packet loss
almucero@almucero-VirtualBox: $
```

Habiendo ya comprobado que ambos contenedores están en redes diferentes y aisladas, la “red2” se conectaría al contenedor “u1” con la orden “**docker network connect**” de esta forma:



```
ping: unknown host
almucero@almucero-VirtualBox: $ sudo docker exec u2 ping -c 4 172.28.0.10
PING 172.28.0.10 (172.28.0.10): 56 data bytes
... 172.28.0.10 ping statistics ...
4 packets transmitted, 0 packets received, 100% packet loss
almucero@almucero-VirtualBox: $ sudo docker network connect red2 u1
almucero@almucero-VirtualBox: $
```

Habiendo hecho esta conexión, si se vuelven a ejecutar los comandos anteriores, el contenedor “u1” ahora sí podrá establecer conexión mediante “ping” con el contenedor “u2”, ya bien sea con el nombre o con la dirección IP:

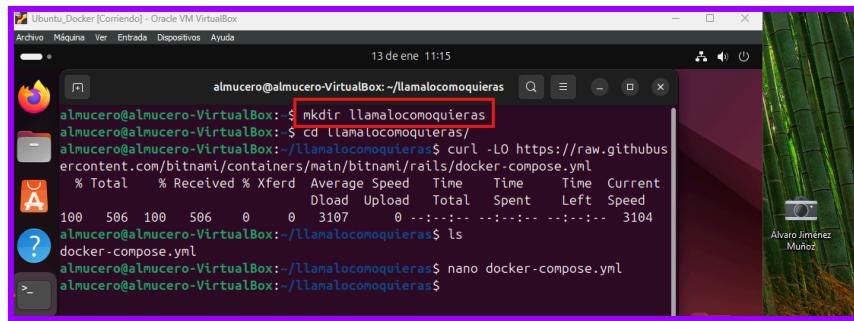


```
almucero@almucero-VirtualBox: $ sudo docker network connect red2 u1
almucero@almucero-VirtualBox: $ sudo docker exec u1 ping -c 4 host2
PING host2 (172.18.0.2): 56 data bytes
64 bytes From 172.18.0.2: icmp_seq=1 ttl=64 time=0.169 ms
64 bytes From 172.18.0.2: icmp_seq=2 ttl=64 time=0.121 ms
64 bytes From 172.18.0.2: icmp_seq=3 ttl=64 time=0.116 ms
64 bytes From 172.18.0.2: icmp_seq=4 ttl=64 time=0.120 ms
...
4 packets transmitted, 4 packets received, 0% packet loss
round-trip min/avg/max/stddev = 0.035/0.060/0.139/0.045 ms
almucero@almucero-VirtualBox: $ sudo docker exec u1 ping -c 4 172.18.0.2
PING 172.18.0.2 (172.18.0.2): 56 data bytes
64 bytes From 172.18.0.2: icmp_seq=0 ttl=64 time=0.130 ms
64 bytes From 172.18.0.2: icmp_seq=1 ttl=64 time=0.130 ms
64 bytes From 172.18.0.2: icmp_seq=2 ttl=64 time=0.116 ms
64 bytes From 172.18.0.2: icmp_seq=3 ttl=64 time=0.035 ms
...
4 packets transmitted, 4 packets received, 0% packet loss
round-trip min/avg/max/stddev = 0.035/0.060/0.139/0.045 ms
almucero@almucero-VirtualBox: $
```

Actividad 6

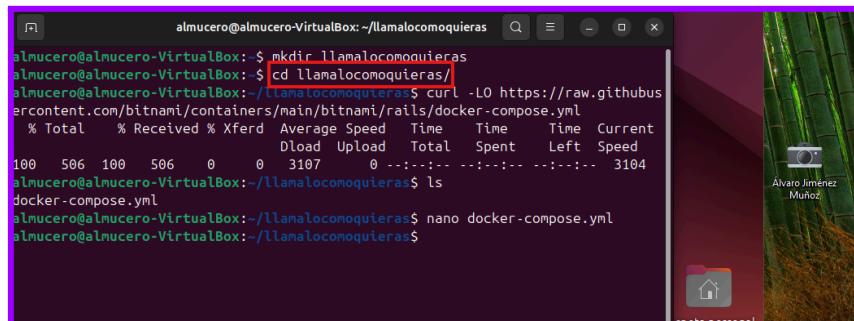
Para montar un entorno de desarrollo para el framework Ruby on Rails utilizando como base el “**docker-compose.yml**” de Bitnami, se debe de introducir el siguiente comando por terminal dentro de la máquina, a poder ser dentro de una carpeta creada:

Para crear la carpeta se hace lo siguiente, introduciendo un nombre cualquiera:



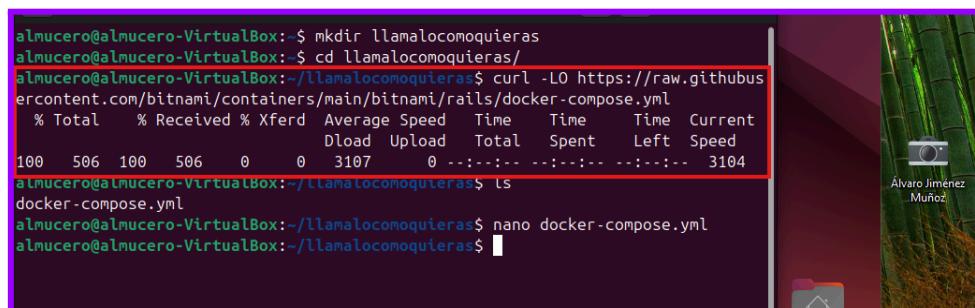
```
Ubuntu_Docker [Comiendo] - Oracle VM VirtualBox
Archivo Máquina Ver Entrada Dispositivos Ayuda
13 de ene 11:15
almucero@almucero-VirtualBox: ~/llamalocomoquieras
almucero@almucero-VirtualBox: $ mkdir llamalocomoquieras
almucero@almucero-VirtualBox: $ cd llamalocomoquieras/
almucero@almucero-VirtualBox: ~/llamalocomoquieras$ curl -LO https://raw.githubusercontent.com/bitnami/containers/main/bitnami/rails/docker-compose.yml
% Total % Received % Xferd Average Speed Time Time Current
          Dload Upload Total Spent Left Speed
100 506 100 506 0 0 3107 0 --:-- --:-- --:-- 3104
almucero@almucero-VirtualBox: ~/llamalocomoquieras$ ls
docker-compose.yml
almucero@almucero-VirtualBox: ~/llamalocomoquieras$ nano docker-compose.yml
almucero@almucero-VirtualBox: ~/llamalocomoquieras$
```

Así se accede a ella:



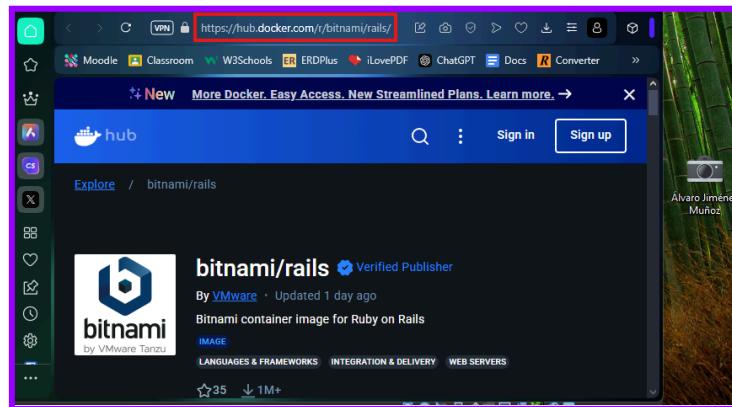
```
Ubuntu_Docker [Comiendo] - Oracle VM VirtualBox
Archivo Máquina Ver Entrada Dispositivos Ayuda
13 de ene 11:15
almucero@almucero-VirtualBox: ~/llamalocomoquieras
almucero@almucero-VirtualBox: $ mkdir llamalocomoquieras
almucero@almucero-VirtualBox: $ cd llamalocomoquieras/
almucero@almucero-VirtualBox: ~/llamalocomoquieras$ curl -LO https://raw.githubusercontent.com/bitnami/containers/main/bitnami/rails/docker-compose.yml
% Total % Received % Xferd Average Speed Time Time Current
          Dload Upload Total Spent Left Speed
100 506 100 506 0 0 3107 0 --:-- --:-- --:-- 3104
almucero@almucero-VirtualBox: ~/llamalocomoquieras$ ls
docker-compose.yml
almucero@almucero-VirtualBox: ~/llamalocomoquieras$ nano docker-compose.yml
almucero@almucero-VirtualBox: ~/llamalocomoquieras$
```

Una vez dentro de la carpeta, el comando que se debía introducir es el siguiente:

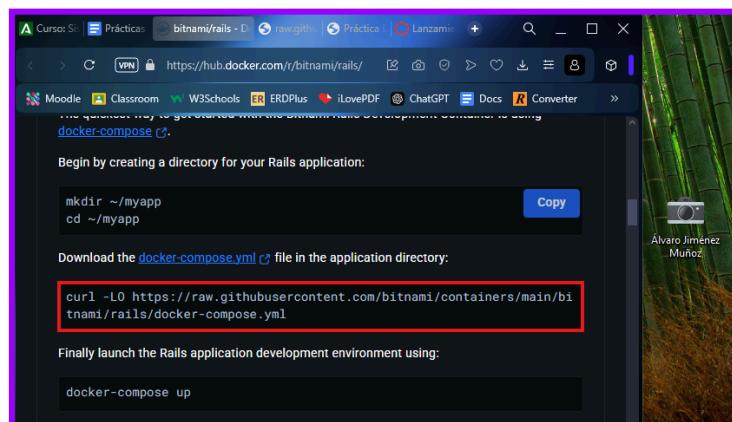


```
Ubuntu_Docker [Comiendo] - Oracle VM VirtualBox
Archivo Máquina Ver Entrada Dispositivos Ayuda
13 de ene 11:15
almucero@almucero-VirtualBox: ~/llamalocomoquieras
almucero@almucero-VirtualBox: $ curl -LO https://raw.githubusercontent.com/bitnami/containers/main/bitnami/rails/docker-compose.yml
% Total % Received % Xferd Average Speed Time Time Current
          Dload Upload Total Spent Left Speed
100 506 100 506 0 0 3107 0 --:-- --:-- --:-- 3104
almucero@almucero-VirtualBox: ~/llamalocomoquieras$ ls
docker-compose.yml
almucero@almucero-VirtualBox: ~/llamalocomoquieras$ nano docker-compose.yml
almucero@almucero-VirtualBox: ~/llamalocomoquieras$
```

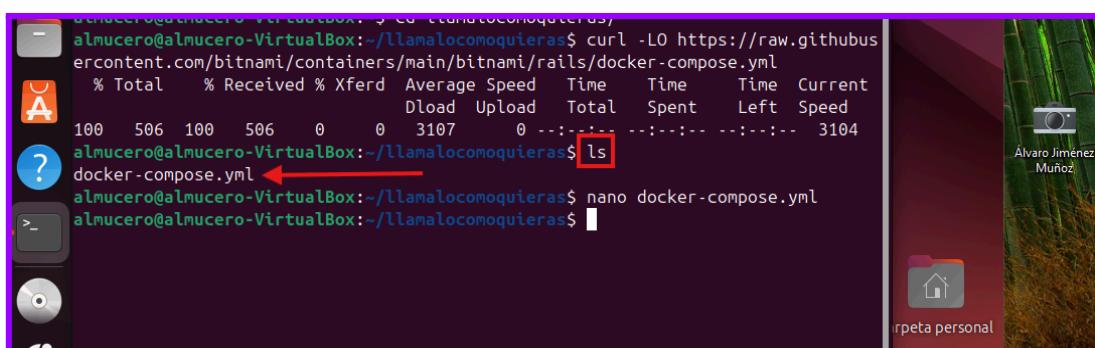
Dicho comando descarga el archivo `docker-compose.yml` de Bitnami y puede ser encontrado, junto a otros muchos más comandos relacionados, en la siguiente web:



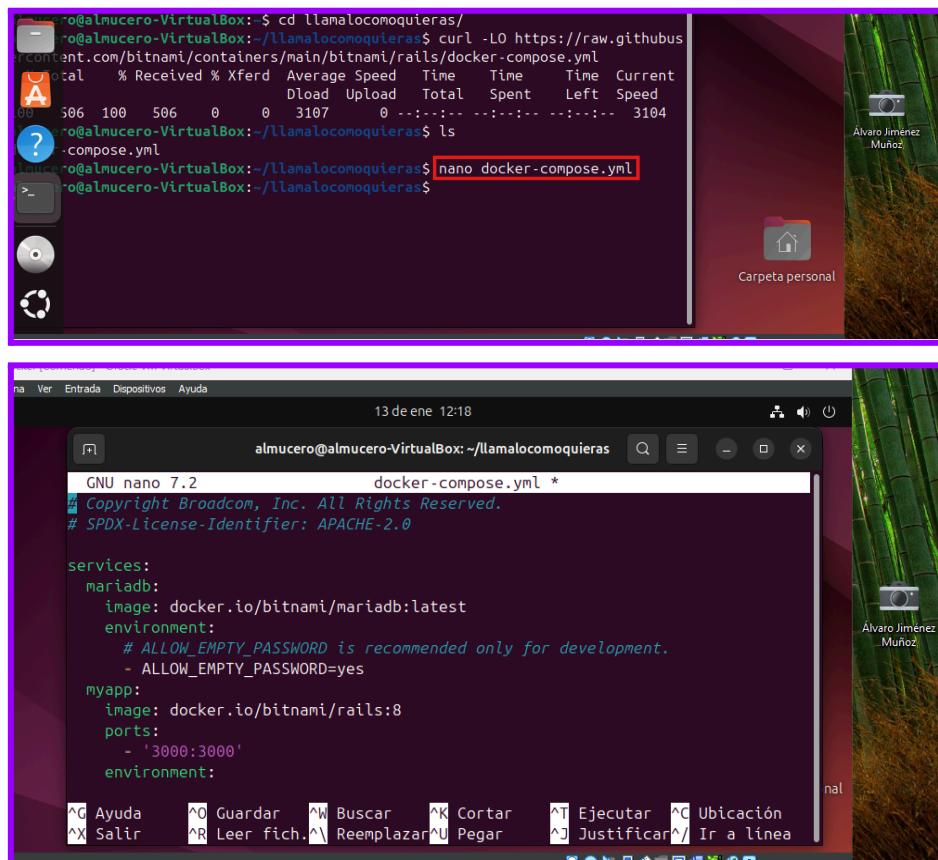
Concretamente, es este de aquí el comando:



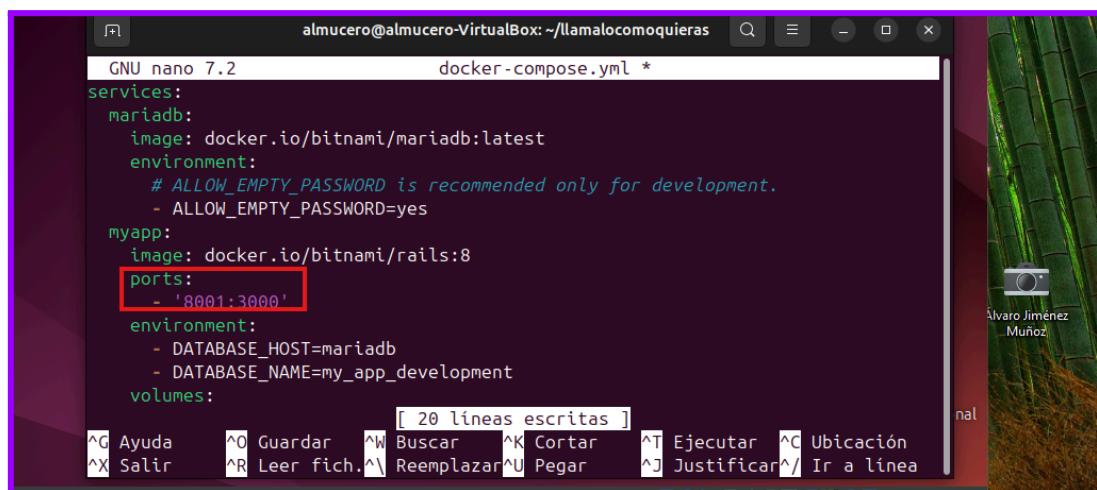
Una vez hecho eso, se comprueba si se ha descargado el archivo usando el comando “ls”:



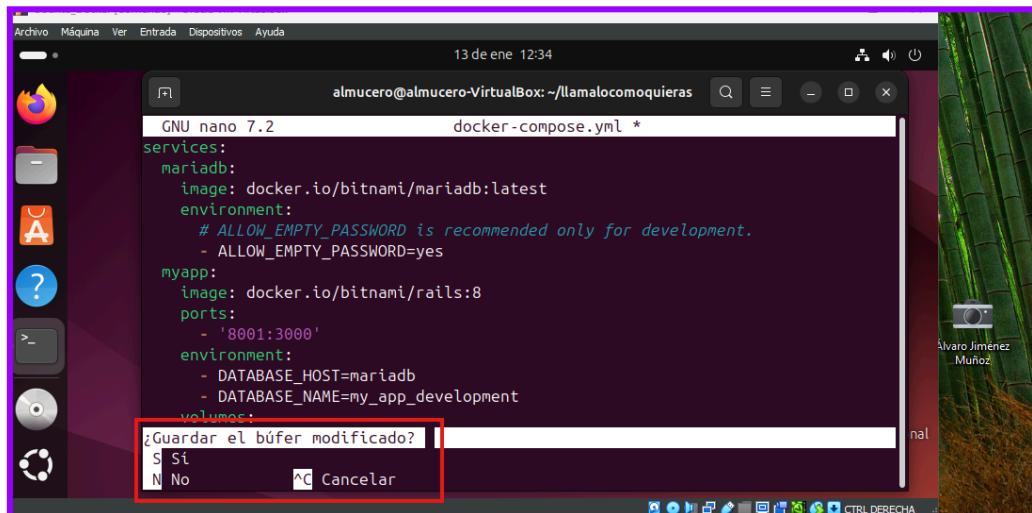
Ya, una vez descargado el archivo y comprobada su instalación, lo siguiente es acceder al archivo para poder modificarlo. Para hacerlo, se debe usar el comando “**`nano`**” como de costumbre:



Al usarlo, se accederá a los contenidos del archivo donde se podrán modificar sus contenidos sin restricciones. Lo único que se va a modificar ahora es el puerto de redirección, cambiándolo al 8001:



Una vez cambiado, se sale del archivo usando Ctrl + X, tras lo cual se guardarán los cambios y se establecerá un nombre:

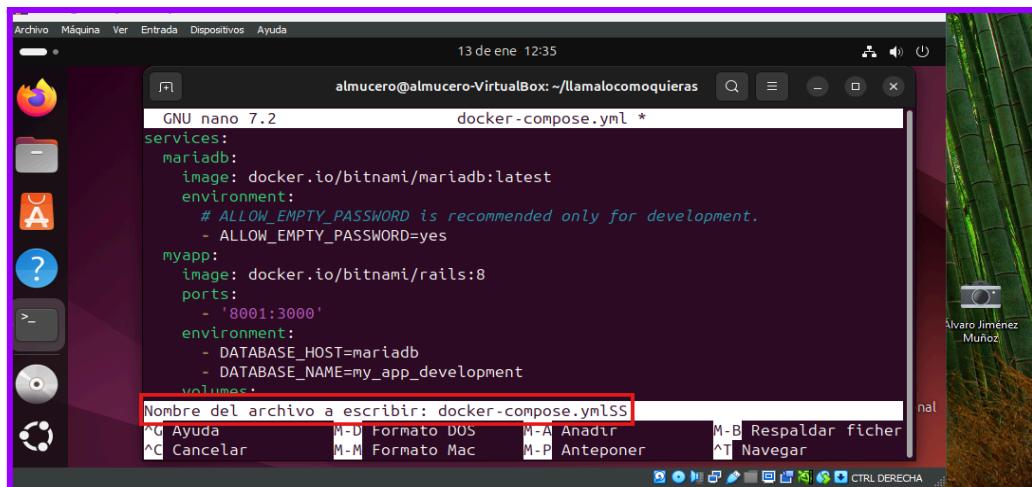


```
GNU nano 7.2          docker-compose.yml *
```

```
services:
  mariadb:
    image: docker.io/bitnami/mariadb:latest
    environment:
      # ALLOW_EMPTY_PASSWORD is recommended only for development.
      - ALLOW_EMPTY_PASSWORD=yes
  myapp:
    image: docker.io/bitnami/rails:8
    ports:
      - '8001:3000'
    environment:
      - DATABASE_HOST=mariadb
      - DATABASE_NAME=my_app_development
    volumes:
```

¿Guardar el búfer modificado?

S Sí N No ^C Cancelar



```
Nombre del archivo a escribir: docker-compose.ymlSS
```

^G Ayuda M-D Formato DOS M-A Anadir M-B Respaldar fichero
^C Cancelar M-M Formato Mac M-P Anteponer ^T Navegar

Después, se inicia la aplicación multicapa con “**docker-compose up**”:

Al hacerlo, es posible que dé error y pida hacer un par de cosas antes:



```
sudo snap install docker      # version 27.2.0, or
sudo apt install docker-compose # version 1.29.2-6
Consulte «snap info docker» para ver más versiones.
almucero@almucero-VirtualBox:~/llamalocomoquieras$ sudo docker-compose up
[sudo] contraseña para almucero:
sudo: docker-compose: orden no encontrada
almucero@almucero-VirtualBox:~/llamalocomoquieras$ docker-compose up
No se ha encontrado la orden «docker-compose», pero se puede instalar con:
sudo snap install docker      # version 27.2.0, or
sudo apt install docker-compose # version 1.29.2-6
Consulte «snap info docker» para ver más versiones.
almucero@almucero-VirtualBox:~/llamalocomoquieras$
```

Por lo que, en caso de ocurrir, se instala lo que indica:



```
[sudo] contraseña para almucero:  
sudo: docker-compose: orden no encontrada  
almucero@almucero-VirtualBox:~/llamalocomoquieras$ docker-compose up  
No se ha encontrado la orden «docker-compose», pero se puede instalar con:  
sudo snap install docker      # versión 27.2.0, or  
sudo apt install docker-compose # versión 1.29.2-6  
Consulte «snap info docker» para ver más versiones.  
almucero@almucero-VirtualBox:~/llamalocomoquieras$ sudo snap install docker  
docker 27.2.0 from Canonical✓ installed  
almucero@almucero-VirtualBox:~/llamalocomoquieras$
```



```
Consulte «snap info docker» para ver más versiones.  
almucero@almucero-VirtualBox:~/llamalocomoquieras$ sudo snap install docker  
docker 27.2.0 from Canonical✓ installed  
almucero@almucero-VirtualBox:~/llamalocomoquieras$ sudo apt install docker-compo  
se  
Leyendo lista de paquetes... Hecho  
Creando árbol de dependencias... Hecho  
Leyendo la información de estado... Hecho  
Los paquetes indicados a continuación se instalaron de forma automática y ya no  
son necesarios.  
    docker-ce-rootless-extras libssl1.0_0 python3-pipfakes slirp4netns
```

Habiendo ya instalado eso, de ser necesario, se vuelve a intentar hacer “**docker-compose up**”, dando lugar a una respuesta muy larga:

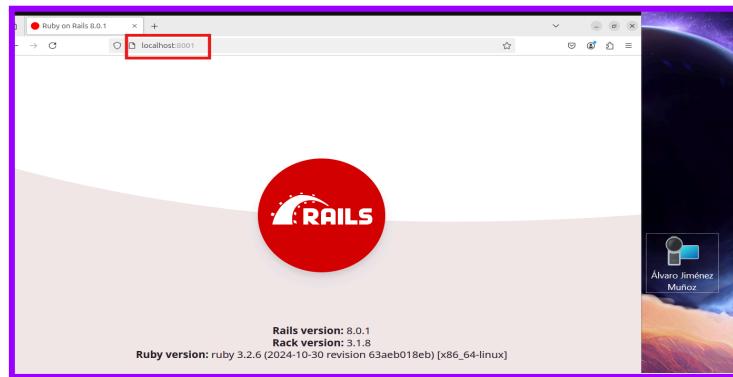


```
almucero@almucero-VirtualBox:~/llamalocomoquieras$ sudo docker compose up  
[sudo] contraseña para almucero:  
[+] Running 4/4  
 ✓ mariadb Pulled          342.9s  
   ✓ e64d741a6e4f Pull complete  
 ✓ myapp Pulled            340.7s  
   ✓ 216a615696b1 Pull complete  
[+] Running 3/3  
 ✓ Network llamalocomoquieras_default     Created          429.7s  
 ✓ Container llamalocomoquieras-mariadb-1 Created          427.9s  
 ✓ Container llamalocomoquieras-myapp-1   Created          0.5s  
Attaching to mariadb-1, myapp-1  
mariadb-1 | mariadb 00:07:20.56 INFO  ==>  
mariadb-1 | mariadb 00:07:20.56 INFO  ==> Welcome to the Bitnami mariadb container  
mariadb-1 | mariadb 00:07:20.56 INFO  ==> Subscribe to project updates by watching https://github.com/bitnami/containers  
mariadb-1 | mariadb 00:07:20.58 INFO  ==> Did you know there are enterprise versions of the Bitnami catalog?  
For enhanced secure software supply chain features, unlimited pulls from Docker, LTS support, or application customization, see Bitnami Premium or Tanzu Application Catalog. See https://www.arrow.com/globalecs-na/vendors/bitnami/ for more information.  
mariadb-1 | mariadb 00:07:20.59 INFO  ==>  
mariadb-1 | mariadb 00:07:20.61 INFO  ==> ** Starting MariaDB setup **
```



```
yapp-1 | Cannot render console from 172.19.0.1! Allowed networks: 127.0.0.0/127.255.255.255, ::1  
yapp-1 | (72.6ms)  CREATE TABLE `schema_migrations` (`version` varchar(255) NOT NULL PRIMARY KEY) /*application='App'*/  
yapp-1 | (30.7ms)  CREATE TABLE `ar_internal_metadata` (`key` varchar(255) NOT NULL PRIMARY KEY, `value` varchar(255), `created_at` datetime(6) NOT NULL, `updated_at` datetime(6) NOT NULL) /*application='App'*/  
yapp-1 | ActiveRecord::SchemaMigration Load (0.5ms)  SELECT `schema_migrations`.`version` FROM `schema_migrations` ORDER BY `schema_migrations`.`version` ASC /*application='App'*/  
yapp-1 | Processing by Rails::WelcomeController#index as HTML  
yapp-1 | Rendering /opt/bitnami/ruby/lib/ruby/gems/3.2.0/gems/railties-8.0.1/lib/rails/templates/rails/welcome/index.html.erb  
yapp-1 | Rendered /opt/bitnami/ruby/lib/ruby/gems/3.2.0/gems/railties-8.0.1/lib/rails/templates/rails/welcome/index.html.erb (Duration: 2.8ms | GC: 0.0ms)  
yapp-1 | Completed 200 OK in 54ms (Views: 8.5ms | ActiveRecord: 0.0ms (0 queries, 0 cached) | GC: 0.2ms)  
yapp-1 |  
yapp-1 |  
yapp-1 | Enable Watch
```

Si funciona el comando tal y como ahí se ve, lo único que faltaría sería comprobar que la aplicación base del framework verdaderamente está en ejecución, para ello, se escribe en la barra de búsqueda del navegador de la máquina virtual lo siguiente:



Ese debería ser el resultado de la búsqueda en caso de que todo haya ido correctamente.

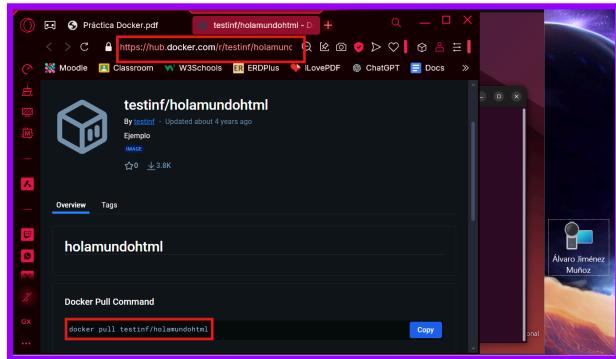
Actividad 7

Lo primero de la actividad será descargar la siguiente imagen no firmada mediante la orden “**docker pull**”:

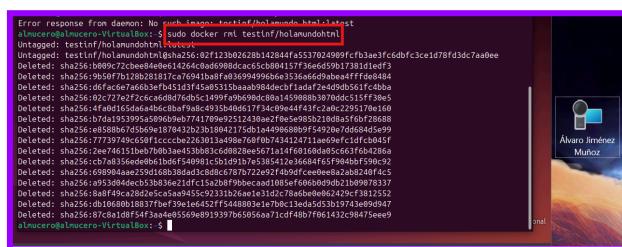
A screenshot of a terminal window on a VirtualBox machine. The terminal prompt is "almucero@almucero-VirtualBox: ~". The user runs the command "sudo docker pull testinf/holamundohtml". The output shows the progress of pulling the image from the Docker registry, with many layers being downloaded. The process is completed successfully with a status message: "Status: Downloaded newer image for testinf/holamundohtml:latest".

```
almucero@almucero-VirtualBox: ~$ sudo docker pull testinf/holamundohtml
[sudo] contraseña para almucero.
Using default tag: latest
latest: Pulling from testinf/holamundohtml
6ec7b7d162b2: Pull complete
db606474d60c: Pull complete
afb30f0cd8e0: Pull complete
3bb2e8051594: Pull complete
4c761b44e2cc: Pull complete
c2199db96575: Pull complete
1b9a9381eea8: Pull complete
50f0689715a3: Pull complete
8a6cc018dd45: Pull complete
052299cf2d76: Pull complete
ee83da709c88: Pull complete
5b10df91e6d0: Pull complete
a2eb858e27d8: Pull complete
ff8ee7220e5: Pull complete
Digest: sha256:02f123b02628b142844fa5537024909fcfb3ae3fc6dbfc3ce1d78fd3dc7aa0ee
Status: Downloaded newer image for testinf/holamundohtml:latest
docker.io/testinf/holamundohtml:latest
almucero@almucero-VirtualBox: ~$
```

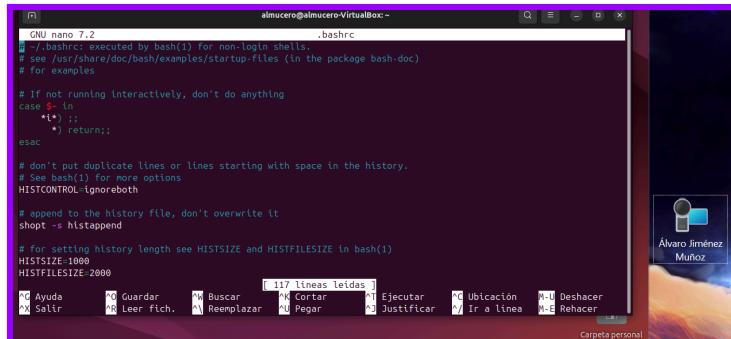
La imagen en cuestión, junto a la orden para descargarla, puede encontrarse en esta web:



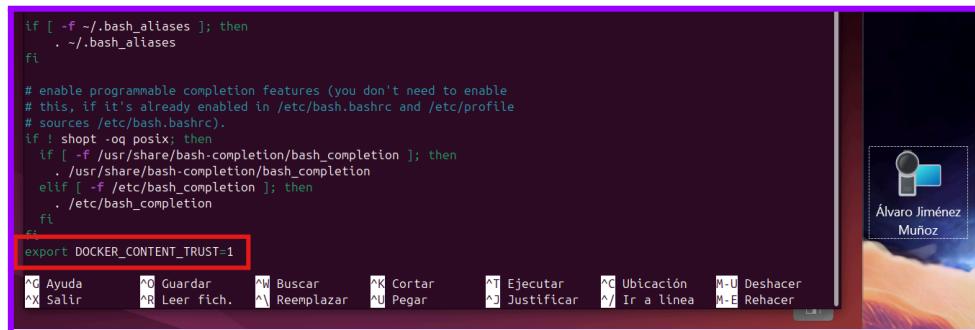
Una vez descargada la imagen se borra con la orden “rmi”:



Habiendo ya borrado la imagen, ahora se debe de habilitar la variable de entorno “**DOCKER_CONTENT_TRUST**” que hace que solo puedan ser descargadas imágenes firmadas, para ello lo primero de todo es acceder al fichero “**.bashrc**” usando el comando “**nano**”:



Una vez dentro, se navega hasta el final del fichero y se añade la siguiente línea de texto:

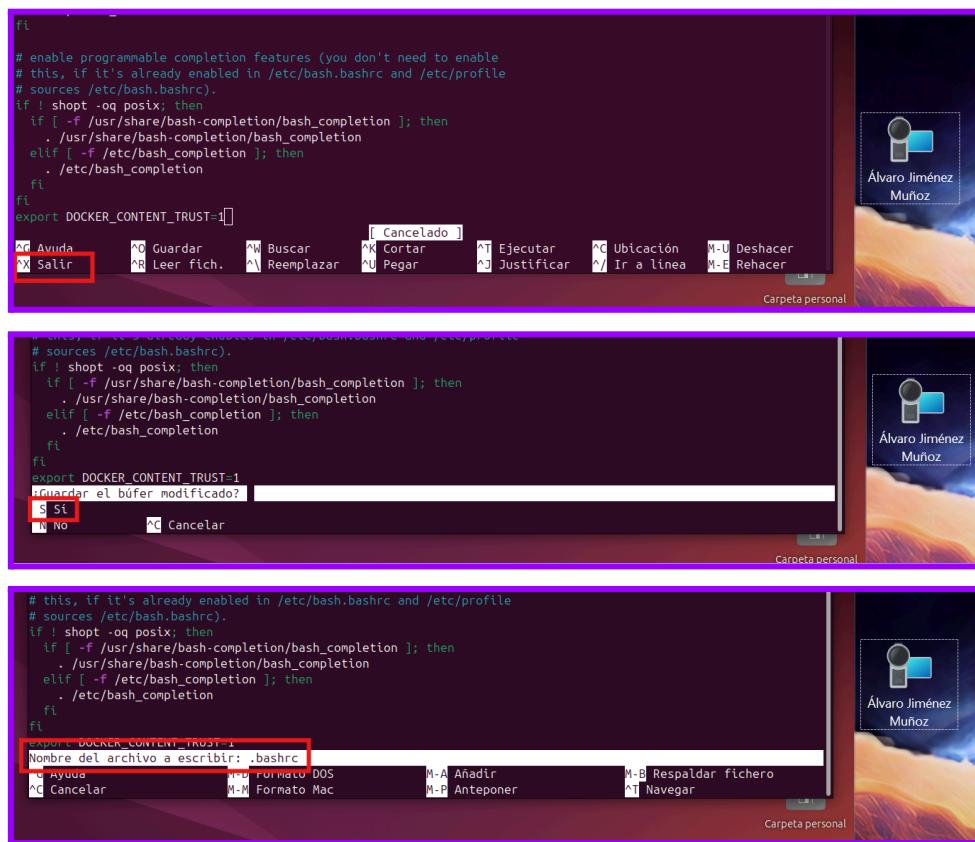


```
if [ -f ~/.bash_aliases ]; then
    . ~/.bash_aliases
fi

# enable programmable completion features (you don't need to enable
# this, if it's already enabled in /etc/bash.bashrc and /etc/profile
# sources /etc/bash.bashrc).
if ! shopt -q posix; then
    if [ -f /usr/share/bash-completion/bash_completion ]; then
        . /usr/share/bash-completion/bash_completion
    elif [ -f /etc/bash_completion ]; then
        . /etc/bash_completion
    fi
fi
export DOCKER_CONTENT_TRUST=1
```

Añadir esa línea y asignarle como valor 1 básicamente lo que hace es habilitar/deshabilitar la realización de subcomandos **PULL/RUN/BUILD** sobre imágenes no firmadas. De esta forma, únicamente se permite trabajar con variables imágenes firmadas, incluso aunque estas hayan sido creadas por el usuario.

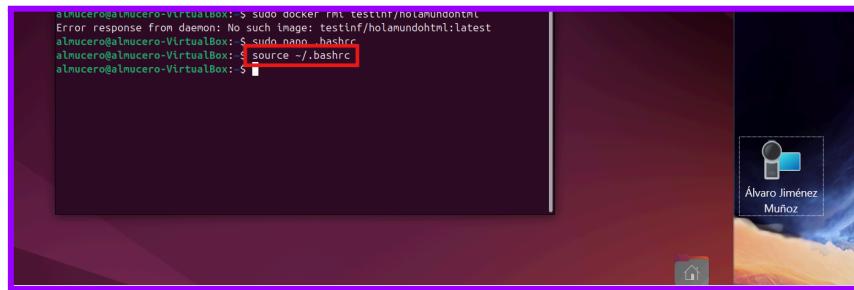
Después, tocará cerrar el fichero guardando los contenidos, igual que la última vez:



The screenshots show the following sequence of events:

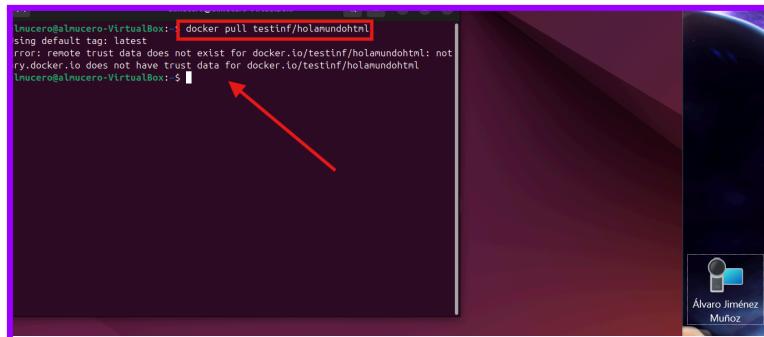
- The terminal window shows the file content with the `export DOCKER_CONTENT_TRUST=1` line added.
- A confirmation dialog box appears asking "¿Guardar el búfer modificado?". The "Sí" button is highlighted with a red box.
- The terminal window shows the file saved with the message "Nombre del archivo a escribir: .bashrc". The file path ".bashrc" is highlighted with a red box.

Tras guardar los cambios, lo primero a hacer antes de nada para poder continuar la actividad es recargar el “.bashrc”, para que la modificación realizada previamente se aplique de verdad utilizando la siguiente orden:



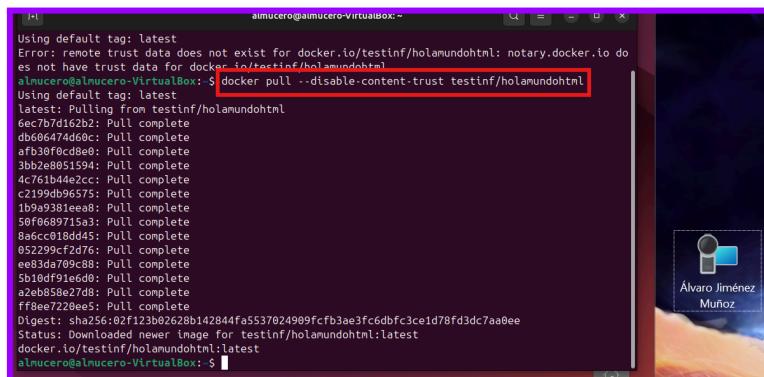
```
almucero@almucero-VirtualBox: ~$ sudo docker rmi testinf/holamundohtml
Error response from daemon: No such image: testinf/holamundohtml:latest
almucero@almucero-VirtualBox: ~$ sudo nano .bashrc
almucero@almucero-VirtualBox: ~$ source ~/.bashrc
almucero@almucero-VirtualBox: ~$
```

Ya con los cambios guardados, si se intentase descargar de nuevo la imagen ocurriría lo siguiente (esta vez no se debe usar sudo, ya que en caso de hacerlo, la imagen se descarga independientemente de la variable de entorno aplicada, y en caso de no usarlo directamente no se descargará nada):



```
almucero@almucero-VirtualBox: ~$ docker pull testinf/holamundohtml
Using default tag: latest
Error: remote trust data does not exist for docker.io/testinf/holamundohtml: notary.docker.io does not have trust data for docker.io/testinf/holamundohtml
almucero@almucero-VirtualBox: ~$
```

Ahora bien, aun con la variable aplicada y la descarga fallando, la imagen todavía puede descargarse forzosamente usando el flag “**--disable-content-trust**” usando la siguiente orden por terminal:



```
almucero@almucero-VirtualBox: ~$ docker pull --disable-content-trust testinf/holamundohtml
Using default tag: latest
latest: Pulling from testinf/holamundohtml
6ec7b7d1g2b2: Pull complete
db6b6474d66c: Pull complete
af30f0c0d8e0: Pull complete
3bb2e8051594: Pull complete
4c761b44e2cc: Pull complete
cz199db96575: Pull complete
1b9a9381ee08: Pull complete
50f0689715a3: Pull complete
8a6cc018dd45: Pull complete
052299cf2d76: Pull complete
ee83da709c88: Pull complete
5b10df91e600: Pull complete
a2eb858e2708: Pull complete
ff8ee7229ee5: Pull complete
Digest: sha256:02f123b02628b142844fa5537024909fcfb3ae3fc0dbfc3ce1d78fd3dc7aa0ee
Status: Downloaded newer image for testinf/holamundohtml:latest
docker.io/testinf/holamundohtml:latest
almucero@almucero-VirtualBox: ~$
```

De esa forma, como se puede ver, la imagen ha podido ser descargada a pesar de que la variable seguía activa.

BIBLIOGRAFÍA

https://www.virtualbox.org/wiki/Downloads
https://kinsta.com/es/blog/volumenes-docker-compose/
https://gist.github.com/codewithleader/4fb24e08d623858e329c625932900947
https://chatgpt.com/
https://atareao.es/tutorial/docker/gestionar-imagenes-con-docker/
https://www.aluracursos.com/blog/creando-volumenes-con-docker
https://www.ionos.com/es-us/digitalguide/servidores/know-how/comandos-de-docker/?srsltid=AfmBOoqhyNLyW9pILHwG_GBbUMUXnlWpz1-OzbMOlgSmSUHUMHCEYVrE
https://www.ionos.com/es-us/digitalguide/servidores/know-how/docker-container-volumes/?srsltid=AfmBOoqCz3c53p7y-bxLmHTTC-p6S91sa9OSMx8EQkiK5U7nNbobl3fm
https://imagineinformacion.com/tutoriales/comandos-docker
https://www.youtube.com/watch?v=DIDel70dFII
https://trifulcas.com/courses/docker/lessons/comandos-para-imagenes/
https://certidevs.com/aprender-docker-imagenes
https://soporte.donweb.com/hc/es/articles/23259975086484--C%C3%B3mo-crear-vol%C3%A1menes-en-Docker
https://www.google.com/
https://www.youtube.com/
Apuntes