

## 2 GUÍA CONFIGURACIÓN SERVIDOR WEB - APACHE

### MATERIAL

- Máquina Virtual Ubuntu 22.04 Desktop.
- Virtualbox
- Ordenador con S.O. Windows 10.

### 2.1 Configuración Servidor Web - APACHE

#### INTRODUCCIÓN

De la arquitectura del Servidor Web – Apache podemos destacar:

- Estructurado en módulos.
- Los módulos proveen una serie de funciones relativas a un aspecto concreto del servidor.
- La funcionalidad de los módulos puede ser activada o desactivada al arrancar el servidor.
- Hay tres categorías de módulos:
  - Módulos base. Realizan funciones básicas.
  - Módulos multiproceso. Se encargan de la conexión de los puertos de la máquina, aceptando y atendiendo las peticiones.
  - Módulos adicionales. Añaden otras funcionalidades al servidor.

El Servidor Apache es desarrollado en el Proyecto HTTP Server de la Apache Software Foundation distribuido bajo licencia de código abierto, permitiendo la distribución de derivados de código abierto y cerrado a partir de su código fuente original.

#### ESTRUCTURA DIRECTORIOS

En cuanto a la estructura de directorios cabe destacar dos partes importantes:

- Carpeta desde donde se sirven las páginas web **“/var/www/html”**, donde se colocan las páginas públicas (index.html). En la misma podemos crear carpetas que serán visibles desde el exterior.
- Carpeta archivos de configuración **“/etc/apache2/”**. En esta carpeta se encuentran los archivos de configuración de Apache. Esta ubicación también alberga el archivo de configuración principal de Apache **“apache2.conf”**, al que se pueden vincular otros archivos de configuración con la directiva **“Include”**. Además encontramos en **“/etc/apache2/”** el archivo de configuración de los puertos **“ports.conf”**.  
Destacar **“/etc/apache2/sites-enabled”** que es la **carpeta donde se guardan los archivos para servir varias páginas**. En ella por defecto, encontraremos el

archivo “000-dafault” que nos servirá como referencia para la creación de nuevos hosts, lo que se conoce como hosts virtuales (virtualhost).

## NÚMERO CONEXIONES SERVIDOR APACHE

El número de conexiones simultáneas o peticiones que puede atender nuestro servidor va en función del equipamiento hardware y software con el que esté dotada la máquina. En cuanto a los recursos hardware no hay establecida una regla si podemos tener en cuenta que el límite nos lo va a marcar la memoria **RAM que consume el servicio Apache** (RAM por conexión) del equipo. Para ello se puede obtener con el siguiente comando:

```
ps -ylC apache2 --sort:rss | awk '{SUM += $8; I += 1} END {print SUM/I/1024}'
```

El resultado obtenido son MB, también es necesario conocer el consumo de memoria RAM del resto de procesos, teniendo en cuenta de que debemos dejar memoria disponible para otras tareas.

El **consumo RAM del resto de procesos** lo podemos obtener con la siguiente instrucción:

```
ps -N -ylC apache2 --sort:rss | awk '{SUM += $8} END {print SUM/1024}'
```

El dato que nos proporcione también es en MB.

Para obtener el número de conexiones simultáneas que podemos permitir planteamos el siguiente cálculo:

$$N^{\circ} \text{ conexiones simultáneas} = \frac{RAM \text{ Total} - RAM \text{ Resto procesos}}{RAM \text{ por conexión}}$$

Imaginando que los datos fueran

- Equipo con 4GB de RAM.
- 800MB consumo resto procesos.
- 10MB consumo por conexión de Apache.

$$N^{\circ} \text{ conexiones simultáneas} = \frac{4096 - 800}{10} = 329$$

Este es el máximo de conexiones simultáneas que nos permite tener activas con los parámetros actuales de la máquina, no obstante, si ponemos este máximo (329), se corre el riesgo de saturar la máquina ya que ocuparíamos toda la memoria RAM. Esto lo podemos evitar si minoramos este parámetro en un 30%, de manera que quedaría en 230 conexiones simultáneas.

Para cambiar es parámetro lo podemos realizar en el archivo **apache2.conf**, en

- MaxKeepAliveRequests

Otro parámetro importante es el del tiempo de espera entre la primera conexión y la siguiente desde la misma conexión expresado en segundos.

- KeepAliveTimeout

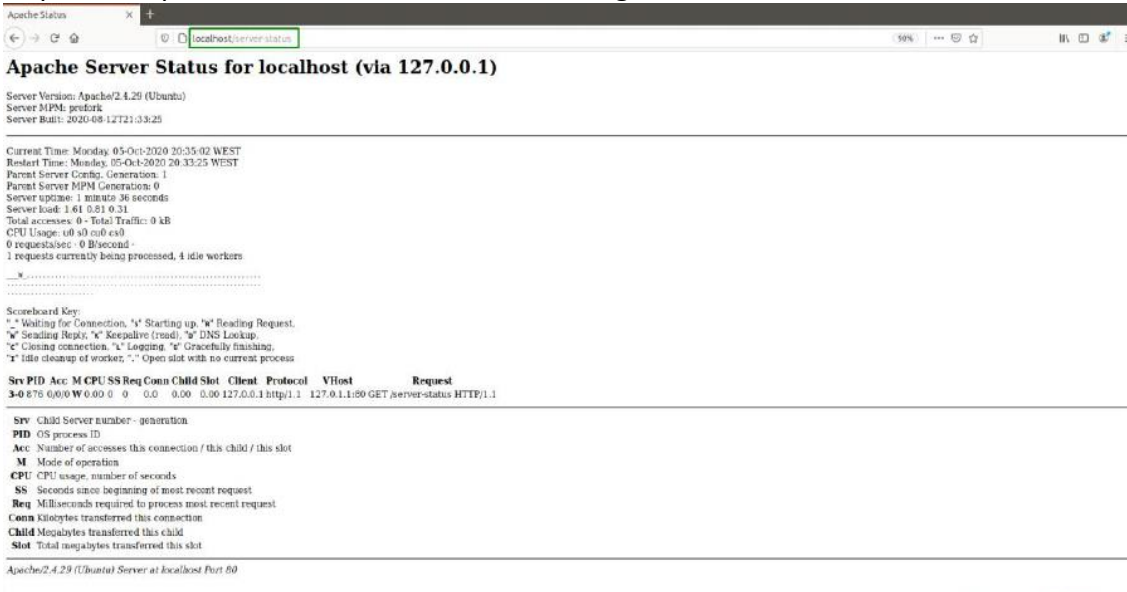
## MONITORIZACIÓN RECURSOS SERVIDOR WEB

El servidor web Apache nos proporciona herramientas para monitorizar es estado de los recursos del mismo.

El acceso al monitor de recursos, lo efectuamos poniendo la siguiente URL en el navegador de la máquina que alberga el servidor:

**localhost/server-status**

La pantalla que se nos muestra es similar a la siguiente:



```

Apache Server Status for localhost (via 127.0.0.1)
Server Version: Apache/2.4.29 (Ubuntu)
Server MPM: prefork
Server Built: 2020-08-12T21:03:25

Current Time: Monday, 05-Oct-2020 20:35:02 WEST
Restart Time: Monday, 05-Oct-2020 20:33:25 WEST
Parent Server Config: Generation: 1
Parent Server MPM: Generation: 0
Server uptime: 1 minute 36 seconds
Server load: 1.61 0.81 0.31
Total accesses: 0 - Total Traffic: 0 kB
CPU Usage: u0 s0 cu0 cs0
0 requests/sec - 0 B/second -
1 requests currently being processed, 4 idle workers

Scoreboard Key:
" " Waiting for Connection, "S" Starting up, "R" Reading Request,
"W" Sending Reply, "K" Keepalive (read), "D" DNS Lookup,
"C" Closing connection, "L" Logging, "G" Gracefully finishing,
"T" Idle cleanup of worker, "-" Open slot with no current process

Srv PID Acc M CPU SS Req Conn Child Slot Client Protocol VHost Request
3-0 876 0/0/0 W 0.00 0 0.0 0.00 0.00 127.0.0.1 http/1.1 127.0.1.1:80 GET /server-status HTTP/1.1

Srv Child Server number - generation
PID OS process ID
Acc Number of accesses this connection / this child / this slot
M Mode of operation
CPU CPU usage: number of seconds
SS Seconds since beginning of most recent request
Req Milliseconds required to process most recent request
Conn KiloBytes transferred this connection
Child MegaBytes transferred this child
Slot Total megabytes transferred this slot

Apache/2.4.29 (Ubuntu) Server at localhost Port 80

```

Para ver la actividad deberemos realizar peticiones al servidor web, bien desde la máquina del servidor web o desde el equipo principal. Si refrescamos la página podemos ver la actividad que se ha registrado en el servidor (resaltado en el cuadro verde).

Apache Status for localhost: Default P... +

localhost/server-status 50%

# Apache Server Status for localhost (via 127.0.0.1)

Server Version: Apache/2.4.29 (Ubuntu)  
Server MPM: prefork  
Server Built: 2020-08-12T21:33:25

---

Current Time: Monday, 05-Oct-2020 20:36:25 WEST  
Restart Time: Monday, 05-Oct-2020 20:33:25 WEST  
Parent Server Config. Generation: 1  
Parent Server MPM Generation: 0  
Server uptime: 2 minutes 59 seconds  
Server load: 0.80 0.75 0.33  
Total accesses: 2 - Total Traffic: 5 kB  
CPU Usage: 0.0 s0 cu0 cs0  
.0112 requests/sec - 260 B/second - 2560 B/request  
1 requests currently being processed, 5 idle workers

.....  
.....

Scoreboard Key:  
".." Waiting for Connection, "S" Starting up, "R" Reading Request,  
"w" Sending Reply, "k" Keepalive (read), "b" DNS Lookup,  
"c" Closing connection, "L" Logging, "G" Gracefully finishing,  
"I" Idle cleanup of worker, "X" Open slot with no current process

Srv	PID	Acc	M	CPU	SS	Req	Conn	Child	Slot	Client	Protocol	VHost	Request
2-0	675	0/0/0	W	0.00	0	0	0.00	0.00	127.0.0.1	http/1.1	127.0.1.1:80	GET /server-status HTTP/1.1	
3-0	676	0/1/1		0.00	78	23	0.00	0.00	127.0.0.1	http/1.1	127.0.1.1:80	GET /server-status HTTP/1.1	
4-0	677	0/1/1		0.00	11	15	0.00	0.00	127.0.0.1	http/1.1	127.0.1.1:80	GET / HTTP/1.1	

Srv Child process number - generation  
PID OS process ID  
Acc Number of accesses this connection / this child / this slot  
M Mode of operation  
CPU CPU usage, number of seconds  
SS Seconds since beginning of most recent request  
Req Milliseconds required to process most recent request  
Conn Kilobytes transferred this connection  
Child Megabytes transferred this child  
Slot Total megabytes transferred this slot

En la parte inferior tenemos la leyenda de lo que representa cada columna que describe la actividad del servidor.

- **Srv.** Número de servicio.
- **PID.** ID Proceso en S.O.
- **Acc.** Número de accesos a la conexión.
- **M.** Modo de operación (leyenda de modos en parte superior a la tabla).
- **CPU.** Número de segundos de uso de procesador.
- **SS.** Segundos desde el inicio de la solicitud más reciente.
- **Req.** Milisegundos para procesar la solicitud más reciente.
- **Conn.** KiloBytes transferidos en esta conexión.
- **Child.** MegaBytes transferidos secundario.
- **Slot.** Total MegaBytes transferidos en este slot.
- **Protocolo.** Indica el protocolo empleado.
- **VHOST.** La IP desde la que se realiza.
- **Request.** Tipo de solicitud.

## 2.2 Despliegue Web con tecnología de virtualización en contenedores

### 1. Instalar Docker engine en Ubuntu.

Para instalar el Docker Engine debemos de actualizar nuestro fichero de fuentes de software (sources)

**sudo apt-get update**

Agregamos los paquetes para intercambio de información por https, los certificados, el curl para el manejo de transferencia de archivos desde línea de comandos, así como el gnupg para el manejo de cifrados y firmas digitales.

**sudo apt-get install apt-transport-https ca-certificates curl gnupg**

```
root@informatica-villa:/# apt-get install apt-transport-https ca-certificates
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias... Hecho
Leyendo la información de estado... Hecho
ca-certificates ya está en su versión más reciente (20230311ubuntu0.22.04.1).
fijado ca-certificates como instalado manualmente.
Se instalarán los siguientes paquetes NUEVOS:
  apt-transport-https
0 actualizados, 1 nuevos se instalarán, 0 para eliminar y 27 no actualizados.
Se necesita descargar 1.510 B de archivos.
Se utilizarán 169 kB de espacio de disco adicional después de esta operación.
¿Desea continuar? [S/n] s
Des:1 http://es.archive.ubuntu.com/ubuntu jammy-updates/universe amd64 apt-transport-https all 2.4.10 [1.510 B]
Descargados 1.510 B en 0s (3.982 B/s)
Seleccionando el paquete apt-transport-https previamente no seleccionado.
(Leyendo la base de datos ... 206152 ficheros o directorios instalados actualmente.)
Preparando para desempaquetar .../apt-transport-https_2.4.10_all.deb ...
Desempaquetando apt-transport-https (2.4.10) ...
Configurando apt-transport-https (2.4.10) ...
root@informatica-villa:/#
```

Añadimos la key para poder descargar el docker-engine del repositorio de Docker.

Instalamos el directorio para las claves

**sudo install -m 0755 -d /etc/apt/keyrings**

Descargamos la clave para conectarnos al repositorio de Docker.

**curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --  
dearmor -o /etc/apt/keyrings/docker.gpg**



```
root@informatica-villa:/# sudo install -m 0755 -d /etc/apt/keyrings
root@informatica-villa:/# curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /etc/apt/keyrings/docker.gpg
```

Añadimos el repositorio al administrador de paquetes (APT)

```
echo \
    "deb [arch=$(dpkg --print-architecture) signed-
    by=/etc/apt/keyrings/docker.gpg] https://download.docker.com/linux/ubuntu \
    $(. /etc/os-release && echo "$VERSION_CODENAME") stable" | \
    sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
```

```
root@informatica-villa:/# echo \
    "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.gpg] https://download.docker.com/linux/ubuntu \
    $(. /etc/os-release && echo "$VERSION_CODENAME") stable" | \
    sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
root@informatica-villa:/#
```

Actualizamos la lista de paquetes

**sudo apt-get update**

Ahora estamos en disposición de instalar el motor de Docker (Docker Engine).

**sudo apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-plugin  
docker-compose-plugin**

```
root@informatica-villa:/# apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias... Hecho
Leyendo la información de estado... Hecho
Se instalarán los siguientes paquetes adicionales:
  docker-ce-rootless-extras git git-man liberror-perl libslirp0 pigz slirp4netns
Paquetes sugeridos:
  aufs-tools cgroupfs-mount | cgroup-lite git-daemon-run | git-daemon-sysvinit git-doc git-email git-gui gitk gitweb git-cvs git-mediawiki git-svn
Se instalarán los siguientes paquetes NUEVOS:
  containerd.io docker-buildx-plugin docker-ce docker-ce-cli docker-ce-rootless-extras docker-compose-plugin git git-man liberror-perl libslirp0 pigz
  slirp4netns
0 actualizados, 12 nuevos se instalarán, 0 para eliminar y 27 no actualizados.
Se necesita descargar 118 MB de archivos.
Se utilizarán 430 MB de espacio de disco adicional después de esta operación.
¿Desea continuar? [S/n] s
```

Comprobamos si está funcionando correctamente con la siguiente instrucción.

En este caso, ¡descargamos la imagen hello-world y la arrancamos recibiéndonos con el célebre mensaje en este caso de Hello from Docker!

**sudo docker run hello-world**

```
root@informatica-villa:/# docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
719385e32844: Pull complete
Digest: sha256:4f53e2564790c8e7856ec08e384732aa38dc43c52f02952483e3f003afbf23db
Status: Downloaded newer image for hello-world:latest

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
$ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
https://hub.docker.com/

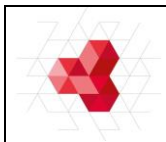
For more examples and ideas, visit:
https://docs.docker.com/get-started/
```

## 2. Instalar contenedor con Apache, MySQL y PHP y realizar pruebas de funcionamiento.

Descargamos la imagen del Docker hub.

```
sudo docker pull pensiero/apache-php-mysql
```





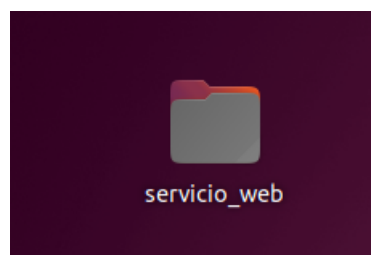
```
root@informatica-villa:/home/informatica# docker pull pensiero/apache-php-mysql
Using default tag: latest
latest: Pulling from pensiero/apache-php-mysql
171857c49d0f: Pull complete
419640447d26: Pull complete
61e52f862619: Pull complete
a6edc16f9ca1: Pull complete
ee073e22bbfb: Pull complete
57a6104d5bf3: Pull complete
0048b08fe1f7: Pull complete
c51e68409ca8: Pull complete
fd5d536e12a7: Pull complete
690db5093a0b: Pull complete
dc87e8478eb3: Pull complete
82cccd33ca41: Pull complete
60db8d11b9b6: Pull complete
e818ff2f3a9e: Pull complete
39e942945306: Pull complete
49e64dedaf54: Pull complete
68c380612628: Pull complete
Digest: sha256:35828429de43d00459387fd3f0c44032fc7d5b6470fcbdc5338c4469210fe05b
Status: Downloaded newer image for pensiero/apache-php-mysql:latest
docker.io/pensiero/apache-php-mysql:latest
```

Comprobamos que se ha cargado la imagen. Viendo que aparece la imagen de Hello World y la de apache-mysql-PHP.

### sudo docker images

```
root@informatica-villa:/home/informatica# docker images
REPOSITORY          TAG         IMAGE ID      CREATED        SIZE
hello-world         latest     9c7a54a9a43c  5 months ago  13.3kB
pensiero/apache-php-mysql latest     c04be4cb614f  2 years ago   589MB
root@informatica-villa:/home/informatica#
```

Creamos una carpeta en el Escritorio que se llama servicio\_web.



Dentro de esta carpeta crearemos el archivo index.php

**nano /home/*nombre\_usuario*/Escritorio/servicio\_web/index.php**

```
<?php

phpinfo()

?>
```



```
GNU nano 6.2 /home/informatica/Escritorio/servicio_web/index.php
?php
phpinfo()
?>
```

Para arrancar el contenedor pondremos:

```
sudo docker run -it -p 81:80 --name lamp -d -v /home/informatica/Escritorio/servicio_web/:/var/www/html pensiero/apache-php-mysql
```

Con el comando anterior hemos definido:

- it. Para que se ejecute de manera interactiva.
- p 81:80. Asociamos el puerto del host 81 al 80 del contenedor
- name lamp. Denominamos al contenedor lamp.
- d. Iniciar el demonio en segundo plano.
- v /home/informatica/Escritorio/servicio\_web/:/var/www/html. Definimos el lugar donde se guardarán los archivos públicos del servidor.
- pensiero/apache-php-mysql. Nombre de la imagen.

```
root@informatica-villa:/home/informatica# sudo docker run -it -p 81:80 --name lamp -d -v /home/informatica/Escritorio/servicio_web/:/var/www/html pensiero/apache-php-mysql
ef97264d99ce77dddf6a481c154e7f0501bd8a6abed53b094ed622ecc9dea91c
root@informatica-villa:/home/informatica#
```

Ejecutamos el contenedor en una consola para configurarlo.

```
docker exec -it lamp /bin/bash
```

```
root@informatica-villa:/home/informatica# docker exec -it lamp /bin/bash
root@ef97264d99ce:/var/www#
```

En este caso para que se nos ejecute al entrar en el contenedor con la consola vamos a realizar los siguientes pasos.

Instalamos el nano.

```
apt-get install nano
```

```
root@ef97264d99ce:/var/www# apt-get install nano
Reading package lists... Done
Building dependency tree
Reading state information... Done
Suggested packages:
  spell
The following NEW packages will be installed:
  nano
0 upgraded, 1 newly installed, 0 to remove and 5 not upgraded.
Need to get 231 kB of archives.
After this operation, 778 kB of additional disk space will be used.
Get:1 http://archive.ubuntu.com/ubuntu bionic/main amd64 nano amd64 2.9.3-2 [231 kB]
Fetched 231 kB in 1s (236 kB/s)
debconf: delaying package configuration, since apt-utils is not installed
Selecting previously unselected package nano.
(Reading database ... 16596 files and directories currently installed.)
Preparing to unpack .../nano_2.9.3-2_amd64.deb ...
Unpacking nano (2.9.3-2) ...
Setting up nano (2.9.3-2) ...
update-alternatives: using /bin/nano to provide /usr/bin/editor (editor) in auto mode
update-alternatives: warning: skip creation of /usr/share/man/man1/editor.1.gz because associated file /usr/share/man/man1/nano.1.gz (of link group editor) doesn't exist
update-alternatives: using /bin/nano to provide /usr/bin/pico (pico) in auto mode
update-alternatives: warning: skip creation of /usr/share/man/man1/pico.1.gz because associated file /usr/share/man/man1/nano.1.gz (of link group pico) doesn't exist
```

Editamos el archivo de los sitios por defecto (000-default) que se encuentra en la carpeta de configuraciones privadas en el sites-enabled.

### nano /etc/apache2/sites-enabled/000-default.conf

```

GNU nano 2.9.3 /etc/apache2/sites-enabled/000-default.conf
<VirtualHost *:80>
  DocumentRoot ${PROJECT_PATH}/${PROJECT_PUBLIC_DIR}
  DirectoryIndex index.php

  <Directory ${PROJECT_PATH}/>
    Options -Indexes +FollowSymLinks +MultiViews
    AllowOverride All
    Order Deny,Allow
    Allow from all
  </Directory>

  CustomLog /proc/self/fd/1 combined
  ErrorLog /proc/self/fd/2
</VirtualHost>

```

Debiendo quedar de la siguiente manera:

```

GNU nano 2.9.3 /etc/apache2/sites-enabled/000-default.conf
<VirtualHost *:80>
  DocumentRoot /var/www/html/
  DirectoryIndex index.php

  <Directory /var/www/>
    Options -Indexes +FollowSymLinks +MultiViews
    AllowOverride All
    Order Deny,Allow
    Allow from all
  </Directory>

  CustomLog /proc/self/fd/1 combined
  ErrorLog /proc/self/fd/2
</VirtualHost>

```

Reiniciamos el servicio apache.

### service apache2 restart

```

root@ef97264d99ce:/var/www# service apache2 restart
* Restarting Apache httpd web server apache2
Terminated

```

En la terminal de la máquina principal iniciamos el contenedor.

### docker ps -a

### docker start <nombre\_contenedor o id\_contenedor>

```

root@informatica-villa:/home/informatica# docker ps -a
CONTAINER ID   IMAGE                                COMMAND                  CREATED    STATUS    PORTS    NAMES
ef97264d99ce   pensiero/apache-php-mysql          "/usr/sbin/apache2ct..." 5 minutes ago   Exited (0) 32 seconds ago   443/tcp, 0.0.0.0:81->80/tcp, :::81->80/tcp   lamp
c58989596eec   hello-world                         "/hello"                 9 days ago     Exited (0) 9 days ago      optimistic_cori

root@informatica-villa:/home/informatica# docker start ef9
ef9
root@informatica-villa:/home/informatica# docker ps -a
CONTAINER ID   IMAGE                                COMMAND                  CREATED    STATUS    PORTS    NAMES
ef97264d99ce   pensiero/apache-php-mysql          "/usr/sbin/apache2ct..." 6 minutes ago   Up 3 seconds                443/tcp, 0.0.0.0:81->80/tcp, :::81->80/tcp   lamp
c58989596eec   hello-world                         "/hello"                 9 days ago     Exited (0) 9 days ago      optimistic_cori

```

Si accedemos desde el navegador web del servidor o la máquina anfitriona por medio del puerto 81, <http://localhost:81> nos debe de mostrar la siguiente página.

### Acceso desde el servidor

PHP 7.4.11 - phpinfo()

localhost:81

PHP Version 7.4.11	
System	Linux 132216cca47 6.2.0-34-generic #34-22.04.1-Ubuntu SMP PREEMPT_DYNAMIC Thu Sep 7 13:12:03 UTC 2 x86_64
Build Date	Oct 10 2020 19:44:50
Server API	Apache 2.0 Handler
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc/php/7.4/apache2
Loaded Configuration File	/etc/php/7.4/apache2/php.ini
Scan this dir for additional .ini files	/etc/php/7.4/apache2/conf.d
Additional .ini files parsed	/etc/php/7.4/apache2/conf.d/10-mysqld.ini, /etc/php/7.4/apache2/conf.d/10-opcache.ini, /etc/php/7.4/apache2/conf.d/10-pdo.ini, /etc/php/7.4/apache2/conf.d/15-xml.ini, /etc/php/7.4/apache2/conf.d/20-bcmath.ini, /etc/php/7.4/apache2/conf.d/20-calendar.ini, /etc/php/7.4/apache2/conf.d/20-ctype.ini, /etc/php/7.4/apache2/conf.d/20-curl.ini, /etc/php/7.4/apache2/conf.d/20-dom.ini, /etc/php/7.4/apache2/conf.d/20-ffi.ini, /etc/php/7.4/apache2/conf.d/20-ftp.ini, /etc/php/7.4/apache2/conf.d/20-gettext.ini, /etc/php/7.4/apache2/conf.d/20-gmp.ini, /etc/php/7.4/apache2/conf.d/20-gnupg.ini, /etc/php/7.4/apache2/conf.d/20-intl.ini, /etc/php/7.4/apache2/conf.d/20-ioncube.ini, /etc/php/7.4/apache2/conf.d/20-javascript.ini, /etc/php/7.4/apache2/conf.d/20-json.ini, /etc/php/7.4/apache2/conf.d/20-mbstring.ini, /etc/php/7.4/apache2/conf.d/20-mysql.ini, /etc/php/7.4/apache2/conf.d/20-pdo_mysql.ini, /etc/php/7.4/apache2/conf.d/20-phar.ini, /etc/php/7.4/apache2/conf.d/20-posix.ini, /etc/php/7.4/apache2/conf.d/20-readline.ini, /etc/php/7.4/apache2/conf.d/20-shmop.ini, /etc/php/7.4/apache2/conf.d/20-simplexml.ini, /etc/php/7.4/apache2/conf.d/20-sockets.ini, /etc/php/7.4/apache2/conf.d/20-sysmsg.ini, /etc/php/7.4/apache2/conf.d/20-sysvshm.ini, /etc/php/7.4/apache2/conf.d/20-sysvsem.ini, /etc/php/7.4/apache2/conf.d/20-tokenizer.ini, /etc/php/7.4/apache2/conf.d/20-xmlreader.ini, /etc/php/7.4/apache2/conf.d/20-xmlwriter.ini, /etc/php/7.4/apache2/conf.d/20-xsl.ini
PHP API	20190902
PHP Extension	20190902
Zend Extension	320190902
Zend Extension Build	API320190902.NTS
PHP Extension Build	API20190902.NTS
Debug Build	no
Thread Safety	disabled
Zend Signal Handling	enabled
Zend Memory Manager	enabled
Zend Multibyte Support	provided by mbstring
IPv6 Support	enabled
DTrace Support	available, disabled
Registered PHP Streams	https, ftps, compress.zlib, php, file, glob, data, http, ftp, phar
Registered Stream Socket Transports	tcp, udp, unix, udg, ssl, tls, tlsv1.0, tlsv1.1, tlsv1.2, tlsv1.3
Registered Stream Filters	zlib.*, string.rot13, string.toupper, string.tolower, string.strip_tags, convert.*, consumed, dechunk, convert.iconv.*

This program makes use of the Zend Scripting Language Engine:  
 Zend Engine v3.4.0, Copyright (c) Zend Technologies



### Acceso desde la máquina principal, anfitrión

PHP 7.4.11 - phpinfo()

No seguro | 192.168.1.137:81

PHP Version 7.4.11	
System	Linux 132216cca47 6.2.0-34-generic #34-22.04.1-Ubuntu SMP PREEMPT_DYNAMIC Thu Sep 7 13:12:03 UTC 2 x86_64
Build Date	Oct 10 2020 19:44:50
Server API	Apache 2.0 Handler
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc/php/7.4/apache2
Loaded Configuration File	/etc/php/7.4/apache2/php.ini
Scan this dir for additional .ini files	/etc/php/7.4/apache2/conf.d
Additional .ini files parsed	/etc/php/7.4/apache2/conf.d/10-mysqld.ini, /etc/php/7.4/apache2/conf.d/10-opcache.ini, /etc/php/7.4/apache2/conf.d/10-pdo.ini, /etc/php/7.4/apache2/conf.d/15-xml.ini, /etc/php/7.4/apache2/conf.d/20-bcmath.ini, /etc/php/7.4/apache2/conf.d/20-calendar.ini, /etc/php/7.4/apache2/conf.d/20-ctype.ini, /etc/php/7.4/apache2/conf.d/20-curl.ini, /etc/php/7.4/apache2/conf.d/20-dom.ini, /etc/php/7.4/apache2/conf.d/20-ffi.ini, /etc/php/7.4/apache2/conf.d/20-ftp.ini, /etc/php/7.4/apache2/conf.d/20-gettext.ini, /etc/php/7.4/apache2/conf.d/20-gmp.ini, /etc/php/7.4/apache2/conf.d/20-gnupg.ini, /etc/php/7.4/apache2/conf.d/20-intl.ini, /etc/php/7.4/apache2/conf.d/20-ioncube.ini, /etc/php/7.4/apache2/conf.d/20-javascript.ini, /etc/php/7.4/apache2/conf.d/20-json.ini, /etc/php/7.4/apache2/conf.d/20-mbstring.ini, /etc/php/7.4/apache2/conf.d/20-mysql.ini, /etc/php/7.4/apache2/conf.d/20-pdo_mysql.ini, /etc/php/7.4/apache2/conf.d/20-phar.ini, /etc/php/7.4/apache2/conf.d/20-posix.ini, /etc/php/7.4/apache2/conf.d/20-readline.ini, /etc/php/7.4/apache2/conf.d/20-shmop.ini, /etc/php/7.4/apache2/conf.d/20-simplexml.ini, /etc/php/7.4/apache2/conf.d/20-sockets.ini, /etc/php/7.4/apache2/conf.d/20-sysmsg.ini, /etc/php/7.4/apache2/conf.d/20-sysvshm.ini, /etc/php/7.4/apache2/conf.d/20-sysvsem.ini, /etc/php/7.4/apache2/conf.d/20-tokenizer.ini, /etc/php/7.4/apache2/conf.d/20-xmlreader.ini, /etc/php/7.4/apache2/conf.d/20-xmlwriter.ini, /etc/php/7.4/apache2/conf.d/20-xsl.ini
PHP API	20190902
PHP Extension	20190902
Zend Extension	320190902
Zend Extension Build	API320190902.NTS
PHP Extension Build	API20190902.NTS
Debug Build	no
Thread Safety	disabled
Zend Signal Handling	enabled
Zend Memory Manager	enabled
Zend Multibyte Support	provided by mbstring
IPv6 Support	enabled
DTrace Support	available, disabled
Registered PHP Streams	https, ftps, compress.zlib, php, file, glob, data, http, ftp, phar
Registered Stream Socket Transports	tcp, udp, unix, udg, ssl, tls, tlsv1.0, tlsv1.1, tlsv1.2, tlsv1.3
Registered Stream Filters	zlib.*, string.rot13, string.toupper, string.tolower, string.strip_tags, convert.*, consumed, dechunk, convert.iconv.*

Para salir del contenedor en la terminal de Ubuntu.

**exit**

Hemos salido, pero el contenedor denominado lamp sigue activo. Para comprobarlo lanzamos la siguiente orden:

**sudo docker ps -a**

```
root@informatica-villa:/home/informatica# docker ps -a
CONTAINER ID   IMAGE               COMMAND                  CREATED        STATUS              PORTS              NAMES
ef97264d99ce   pensiero/apache-p  "/usr/sbin/apache2ct...  5 minutes ago   Exited (0) 32 seconds ago              lamp
c58989596eec   hello-world        "/hello"                9 days ago     Exited (0) 9 days ago                  optimistic_corl
```

Podemos trabajar con los contenedores como si fueran servicios. En este caso parándolos (stop) y arrancándolos (start).

La regla general sería:

**sudo docker acción nombre\_containerID**

Si lo trasladamos al contenedor que hemos llamado lamp2, quedaría:

Haciéndolo por el nombre.

**sudo docker start lamp2**

```
root@informatica-villa:/home/informatica# docker start lamp2
lamp2
```

Por el contenedor ID quedaría.

**sudo docker start b58f66d7791b**

También sería válido en vez de poner todos los dígitos del id del contenedor poner sólo los primeros dígitos.

```
root@informatica-villa:/home/informatica# sudo docker start b58f66d7791b
b58f66d7791b
```

Exponer un puerto del contenedor para que salga fuera de la máquina.

La regla general es:

**sudo docker container run -d -p port\_host:port\_docker imagen**

Particularizando.

**sudo docker container run -d -p 82:80 --name lamp2 pensiero/apache-php-mysql**

```
root@informatica-villa:/home/informatica# docker ps -a
CONTAINER ID   IMAGE               COMMAND                  CREATED        STATUS              PORTS              NAMES
b58f66d7791b   pensiero/apache-p  "/usr/sbin/apache2ct...  3 minutes ago   Up 3 minutes       443/tcp, 0.0.0.0:82->80/tcp, :::82->80/tcp   lamp2
ef97264d99ce   pensiero/apache-p  "/usr/sbin/apache2ct...  34 minutes ago   Up 27 minutes      443/tcp, 0.0.0.0:81->80/tcp, :::81->80/tcp   lamp
c58989596eec   hello-world        "/hello"                9 days ago     Exited (0) 9 days ago                  optimistic_corl
root@informatica-villa:/home/informatica#
```

En el caso de que quisiéramos borrar un contenedor.

**sudo docker rm <nombre o containerID>**

```
root@informatica-villa:/home/informatica# docker rm b58f66d7791b
Error response from daemon: You cannot remove a running container b58f66d7791b58c7f1ee471cd52fc4f9cb8c6ec8cc329f3af4da11adca276022. Stop the container before attempting removal or force remove
root@informatica-villa:/home/informatica#
```

Para poder eliminar un contenedor primero debemos de pararlo y ya luego podemos borrarlo.

**sudo docker stop <nombre o containerID>**

**sudo docker rm <nombre o containerID>**

```
root@informatica-villa:/home/informatica# docker stop b58f66d7791b
b58f66d7791b
root@informatica-villa:/home/informatica# docker rm b58f66d7791b
b58f66d7791b
root@informatica-villa:/home/informatica#
```

La imagen de docker que hemos ejecutado anteriormente cuyo contenedor se ha denominado lamp, hemos tenido que configurarle ciertos parámetros dentro. Si la exploramos internamente comprobamos que no contiene el MySQL. Por lo que ahora vamos a emplear dos imágenes una para Apache-PHP y otra para MySQL, que son más sencillas de administrar. En la imagen de Apache-PHP no necesitaremos realizar ninguna configuración, mientras que en la imagen MySQL vamos a crear una base de datos.

Descargamos las imágenes de Apache-PHP y MySQL .

De las opciones que nos ofrece el docker hub para el PHP nos descargaremos la versión 7.0, esto lo conseguimos añadiendo al final del comando :7.0-apache.

**docker pull php:7.0-apache**

```
root@informatica-villa:/home/informatica# docker pull php:7.0-apache
7.0-apache: Pulling from library/php
Digest: sha256:1d34b2e491a02ba7a8d26478132015e197a5ffea37f0a93b42621d11cfe042cc
Status: Image is up to date for php:7.0-apache
docker.io/library/php:7.0-apache
```

En el caso de MySQL vamos a descargar la 5.7 puesto que no tiene una restricción. Esto puede ser práctico si tenemos que recrear escenarios de configuración de entornos de trabajo antiguos.

### docker pull mysql:5.7

```
root@informatica-villa:/home/informatica# docker pull mysql:5.7
5.7: Pulling from library/mysql
9ad776bc3934: Pull complete
9e4eda42c982: Pull complete
df6d882cf587: Pull complete
6c804e92b324: Pull complete
fd54ada0c48d: Pull complete
4ed8fb20ac8d: Pull complete
eec2b1bc5454: Pull complete
41c3423057b7: Pull complete
122b2c7b16c0: Pull complete
0d30e03d70e3: Pull complete
71c43898e898: Pull complete
Digest: sha256:4f9bfb0f7dd97739ceedb546b381534bb11e9b4abf013d6ad9ae6473fed66099
Status: Downloaded newer image for mysql:5.7
docker.io/library/mysql:5.7
root@informatica-villa:/home/informatica#
```

Mostramos las imágenes.

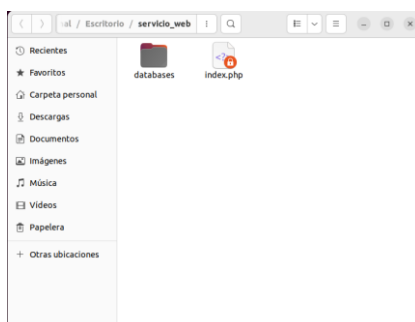
### docker images

```
root@informatica-villa:/home/informatica# docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
mysql	5.7	3b85be0b10d3	2 months ago	581MB
mysql	<none>	a5b7ceed4074	2 months ago	581MB
hello-world	latest	9c7a54a9a43c	5 months ago	13.3kB
pensiero/apache-php-mysql	latest	c04be4cb614f	2 years ago	589MB
php	7.0-apache	aa67a9c9814f	4 years ago	368MB

```
root@informatica-villa:/home/informatica#
```

Aprovechando la carpeta que ya creamos antes vamos a crear dentro la subcarpeta database para que se almacenen ahí los datos de la base de datos cuando se cierre el contenedor y no perder la información.





Ahora vamos a arrancar el contenedor donde configuramos la contraseña del mismo.

```
docker run -p 3307:3306 --name basedatos -v /home/informatica/Escritorio/servicioweb/database:/var/lib/mysql -e MYSQL_ROOT_PASSWORD=1234 -d mysql:5.7
```

-p 3307:3306. Enlazamos el puerto de salida al host el 3307 con el 3306 dentro del contenedor.

--name basedatos. Denominamos el contenedor con este nombre.

-v /home/informatica/Escritorio/servicioweb/database:/var/lib/mysql. Asociamos la carpeta del host donde se guardarán los datos con la interna del contenedor.

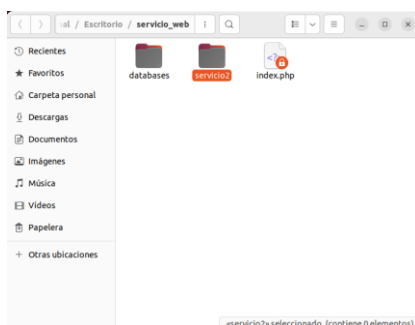
-e MYSQL\_ROOT\_PASSWORD=1234. Definimos la contraseña del root.

-d. Ejecución en segundo plano del demonio.

mysql:5.7. Nombre la de imagen a emplear y la versión.

```
root@informatica-villa:/home/informatica# docker run -p 3307:3306 --name basedatos -v /home/informatica/Escritorio/servicioweb/database:/var/lib/mysql -e MYSQL_ROOT_PASSWORD=1234 -d mysql:5.7
1fc75a631bc403eca7867f9b4f9a86a6173c2ab3787678ef126fee07dfca6db0
root@informatica-villa:/home/informatica#
```

Arrancamos el PHP-apache. En este caso vamos a crear dos servicios para ver la facilidad de crearlos. El primer servicio lo direccionaremos a la carpeta de servicio\_web (miservidorphp), mientras que el segundo lo direccionaremos a la carpeta de servicio2 (miservidorphp2).





Dentro de la subcarpeta servicio2 crearemos un **index.html** con el siguiente contenido, personalizar con el nombre de cada alumno/a en el código:

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <h1>Despliegue de Aplicaciones Web - DPL</h1>
    <br>
    <h2>Servicio Web 2</h2>
  </head>
  <body>
    <br>
    <a>Nombre y Apellidos Alumno-a</a>
    <br>
  </body>
</html>
```

```
GNU nano 6.2 /home/informatica/Escritorio/servicio_web/servicio2/index.html *
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <h1>Despliegue de Aplicaciones Web - DPL</h1>
    <br>
    <h2>Servicio Web 2</h2>
  </head>
  <body>
    <br>
    <a>Nombre y Apellidos Alumno-a</a>
    <br>
  </body>
</html>
```

Con el parámetro --link enlazamos nuestro contenedor con basedatos.

### Servicio 1

```
docker run -p 9090:80 -v /home/informatica/Escritorio/servicioweb/:/var/www/html
--name miservidorphp -d --link basedatos php:7.0-apache
```

### Servicio 2

```
docker run -p 9091:80 -v /home/informatica/Escritorio/servicio_web/servicio2/:/var/www/html
--name miservidorphp2 -d --link basedatos php:7.0-apache
```

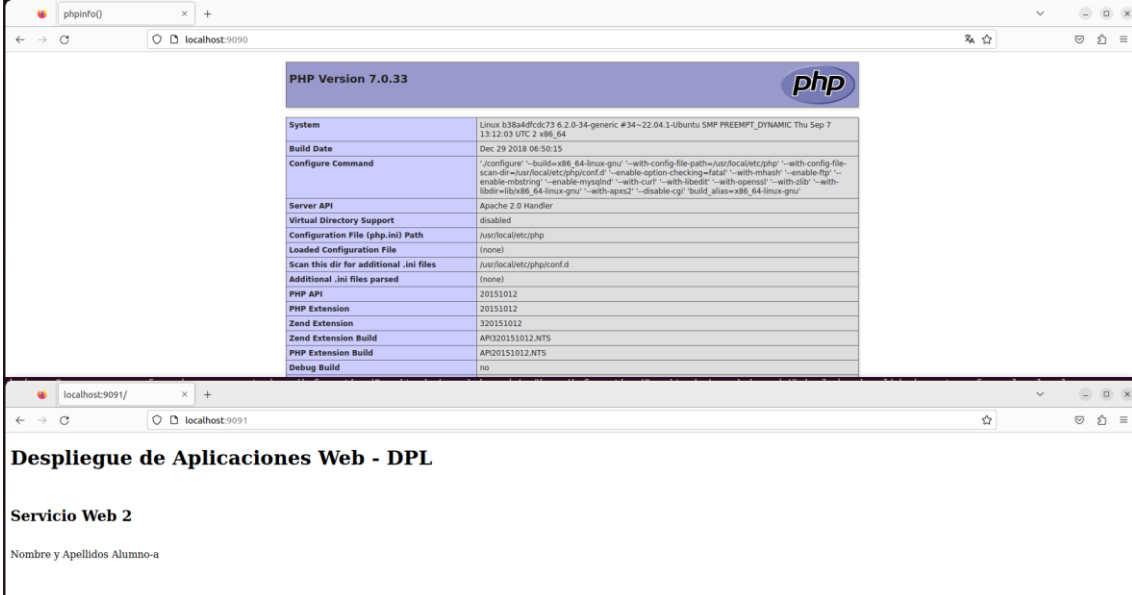
```
root@informatica-villa:/home/informatica# docker run -p 9090:80 -v /home/informatica/Escritorio/servicio_web/:/var/www/html --name miservidorphp -d --link basedatos php:7.0-apache
b38a4dfdc737b420fb13d756ad5788411ace287e945d91c7e298229f564d85d
root@informatica-villa:/home/informatica# docker run -p 9091:80 -v /home/informatica/Escritorio/servicio_web/servicio2/:/var/www/html --name miservidorphp2 -d --link basedatos php:7.0-apache
15a1daeb64b2017c3984cd74ff7ad04e1d4e9276e6c4f88540abc39b7a2007e
root@informatica-villa:/home/informatica#
```

Mostramos los contenedores

**docker ps -a**

```
root@informatica-villa:/home/informatica# docker ps -a
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS                               NAMES
15a1daeb64b   php:7.0-apache  "docker-php-entrypol..." About a minute ago Up About a minute   0.0.0.0:9091->80/tcp, :::9091->80/tcp  nlservidorphp2
b38a4dfcd73   php:7.0-apache  "docker-php-entrypol..." 2 minutes ago  Up 2 minutes     0.0.0.0:9090->80/tcp, :::9090->80/tcp  nlservidorphp
1fc75a631b04   mysql:5.7       "docker-entrypoint.s..." 14 minutes ago Up 14 minutes     33060/tcp, 0.0.0.0:3307->3306/tcp, :::3307->3306/tcp  basedatos
ef97264d99ce   pensiero/apache-php-mysql  "/usr/sbin/apache2ctl..." About an hour ago Up About an hour   443/tcp, 0.0.0.0:81->80/tcp, :::81->80/tcp  lamp
c58989596eec   hello-world     "hello"                  9 days ago    Exited (0) 9 days ago                               optimistc_cort
```

Comprobamos que están operativos los dos servicios por medio del navegador web.



Comprobamos que funciona el MySQL con la contraseña configurada y crearemos una cuenta.

**docker exec -it basedatos /bin/bash**

```
root@informatica-villa:/home/informatica# docker exec -it basedatos /bin/bash
bash-4.2#
```

Accedemos al MySQL y ponemos la contraseña configurada y deberemos acceder al prompt de MySQL.

**mysql -u root -p**

```
bash-4.2# mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 2
Server version: 5.7.43 MySQL Community Server (GPL)

Copyright (c) 2000, 2023, 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> █
```

Procedemos a crear la base de datos usuarios y la tabla clientes. Donde introduciremos un cliente a modo de prueba.

Mostramos primero las bases de datos existentes.

**show databases;**

```
mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| mysql |
| performance_schema |
| sys |
+-----+
4 rows in set (0.01 sec)

mysql>
```

Creamos la base de datos usuarios.

**create database usuarios;**

```
mysql> create database usuarios;
Query OK, 1 row affected (0.00 sec)
```

Mostramos que se ha creado.

**show databases;**

```
mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| mysql |
| performance_schema |
| sys |
| usuarios |
+-----+
5 rows in set (0.00 sec)

mysql>
```

Hacemos uso de la misma para crear la tabla con los campos username, nombre, correo y contra.

**use usuarios;**

**create table clientes(username varchar(20) primary key not null, nombre varchar(30), correo varchar (50), contra varchar(20));**

```
mysql> use usuarios;
Database changed
mysql> create table clientes(username varchar(20) primary key not null, nombre varchar(30), correo varchar (50), contra varchar(20));
Query OK, 0 rows affected (0.87 sec)
```

Mostramos las tablas para verificar que se ha creado y vemos los campos de los que dispone.

**show tables;**

```
mysql> show tables;
+-----+
| Tables_in_usuarios |
+-----+
| clientes |
+-----+
1 row in set (0.00 sec)
```

**select \* from clientes;**

```
mysql> select * from clientes;
Empty set (0.00 sec)
```

```
insert into clientes values ('cjguedes','cristobal','cristobal.guedes@cifpvilladeaguimes.es', '1234');
```

```
mysql> insert into clientes values('cjguedes','cristobal','cristobal.guedes@cifpvilladeaguimes.es', '1234');
Query OK, 1 row affected (0.15 sec)
```

```
select * from clientes
```

```
mysql> select * from clientes;
+-----+-----+-----+-----+
| username | nombre | correo | contra |
+-----+-----+-----+-----+
| cjguedes | cristobal | cristobal.guedes@cifpvilladeaguimes.es | 1234 |
+-----+-----+-----+-----+
1 row in set (0.00 sec)
```

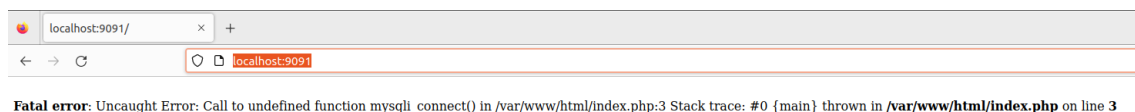
```
<?php
//Create connection
$conn=mysqli_connect("ip_servidor:3307","root","1234", "usuarios");

//Check connection
if (!$conn){
    die("Connection failed: " . mysqli_connect_error());
}
echo "Connected successfully";
?>
```

```
GNU nano 6.2 /home/informatica/Escritorio/servicio_web/servicio2/index.php
<?php
//Create connection
$conn = mysqli_connect("192.168.1.137:3307","root","1234", "usuarios");

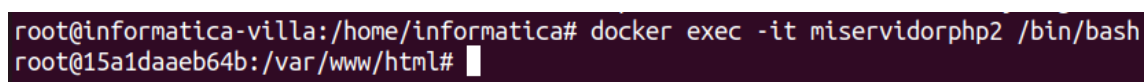
//Check connection
if (!$conn){
    die("Connection failed: " . mysqli_connect_error());
}
echo "Connected suessfully";
?>
```

Si comprobamos su funcionamiento desde el navegador veremos que nos arroja un fallo con la extensión mysqli.



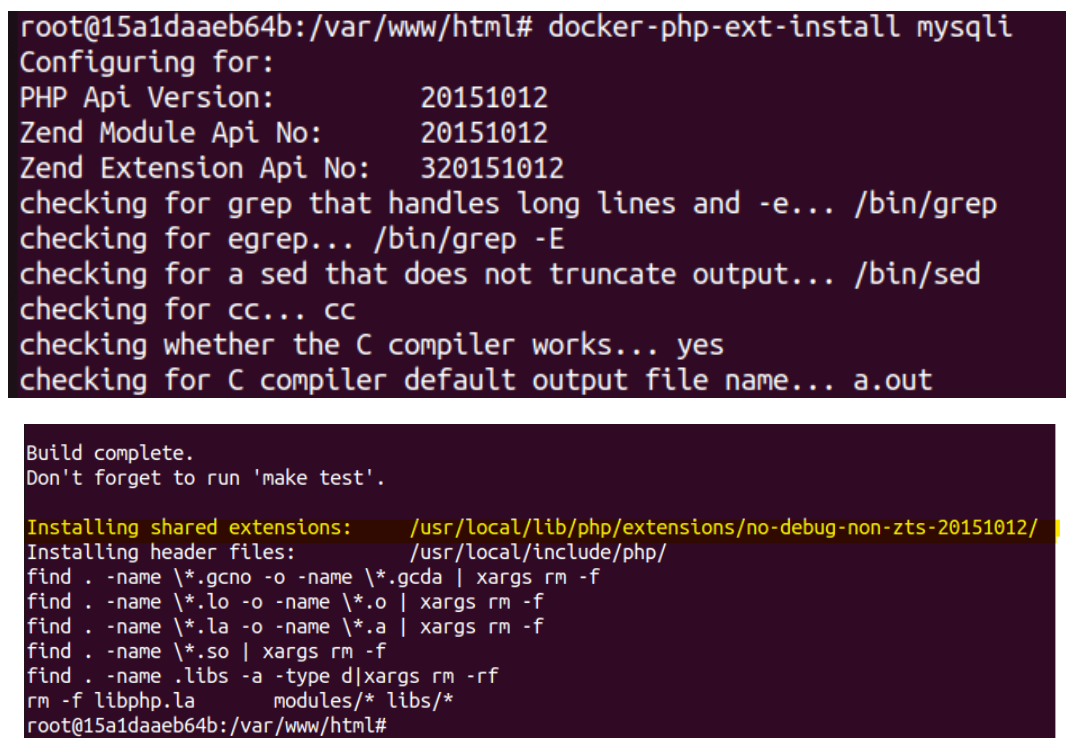
Esto sucede porque no está activa la extensión que se emplea para la conexión con MySQL en la imagen empleada del servicio de php-apache. Para solucionarlo debemos configurarlo.

**docker exec -it miservidorphp2 /bin/bash**



Instalamos la extensión de MySQL. Donde debemos de anotarnos la ruta que indica al final de "Installing shared extensions" ya que la deberemos referenciar para que se pueda hacer uso de ella.

**docker-php-ext-install mysqli**



Installing shared extensions:

**/usr/local/lib/php/extensions/no-debug-non-zts-20151012/**

Nos movemos a esa ruta.

**cd /usr/local/lib/php/extensions/no-debug-non-zts-20151012/**

```
root@15a1daaeb64b:/var/www/html# cd /usr/local/lib/php/extensions/no-debug-non-zts-20151012/
root@15a1daaeb64b:/usr/local/lib/php/extensions/no-debug-non-zts-20151012#
```

Comprobamos que se encuentra el archivo mysqli.so

**ls**

```
root@15a1daaeb64b:/usr/local/lib/php/extensions/no-debug-non-zts-20151012# ls
mysqli.so  opcache.so
```

Comprobado que tenemos la extensión, agregamos la ruta al mismo en el archivo php.ini

**cd /usr/local/etc/php**

```
root@15a1daaeb64b:/usr/local/lib/php/extensions/no-debug-non-zts-20151012# cd /usr/local/etc/php
root@15a1daaeb64b:/usr/local/etc/php#
```

Instalamos el editor de terminal nano. Primero hacemos una actualización de repositorios y luego ya podemos instalarlo.

**apt-get update**

**apt-get install nano**

Como no nos deja instalar el nano puesto que es una versión antigua de Debian. Lo consultamos con la siguiente orden:

**cat /etc/os-release**

```
root@15a1daaeb64b:/usr/local/etc/php# cat /etc/os-release
PRETTY_NAME="Debian GNU/Linux 9 (stretch)"
NAME="Debian GNU/Linux"
VERSION_ID="9"
VERSION="9 (stretch)"
ID=debian
HOME_URL="https://www.debian.org/"
SUPPORT_URL="https://www.debian.org/support"
BUG_REPORT_URL="https://bugs.debian.org/"
root@15a1daaeb64b:/usr/local/etc/php#
```

Optaremos por extraer el archivo del contenedor y editarlo en el host para posteriormente volverlo a introducir en la máquina.



El comando de manera genérica sería:

### Del Contenedor a Host

**docker cp NOMBRE\_CONTENEDOR:RUTA\_DEL\_CONTENEDOR RUTA\_LOCAL**

### De Host al Contenedor

**docker cp RUTA\_LOCAL NOMBRE\_CONTENEDOR:RUTA\_DEL\_CONTENEDOR**

En nuestro caso vamos a traernos dos archivos en la misma ruta:

- php.ini-development
- php.ini-production

**docker cp miservidorphp2://usr/local/etc/php/php.ini-development /home/informatica/Escritorio/**

**docker cp miservidorphp2://usr/local/etc/php/php.ini-production /home/informatica/Escritorio/**

```
root@informatica-villa:/home/informatica# docker cp miservidorphp2://usr/local/etc/php/php.ini-development /home/informatica/Escritorio/
Successfully copied 72.2kB to /home/informatica/Escritorio/
root@informatica-villa:/home/informatica# docker cp miservidorphp2://usr/local/etc/php/php.ini-production /home/informatica/Escritorio/
Successfully copied 72.2kB to /home/informatica/Escritorio/
root@informatica-villa:/home/informatica#
```

Procedemos a editarlo para luego volver a copiarlos al directorio original del contenedor de Docker.

### **nano /home/informatica/Escritorio/php.ini-development**

Buscamos dentro del editor nano con **ctrl+w** **dinamyc extensions**. Debemos descomentar la línea del path y actualizarla con la ruta de nuestra extensión (en rojo).

```
;;;;;;;;;;;;;;;;;;;;;;;;;;
; Dynamic Extensions ;
;;;;;;;;;;;;;;;;;;;;;;;;;;

; If you wish to have an extension loaded automatically, use the following
; syntax:
;
;   extension=modulename.extension
;
; For example, on Windows:
;
;   extension=msql.dll
;
; ... or under UNIX:
;
;   extension=msql.so
;
; ... or with a path:
;
;   extension=/path/to/extension/msql.so
;
; If you only provide the name of the extension, PHP will look for it in its
; default extension directory.
```

Debiendo quedar:

```

; Dynamic Extensions ;
;
; If you wish to have an extension loaded automatically, use the following
; syntax:
;
; extension=module_name.extension
;
; For example, on Windows:
;
; extension=msql.dll
;
; ... or under UNIX:
;
; extension=msql.so
;
; ... or with a path:
;
; extension=/usr/local/lib/php/extensions/no-debug-non-zts-20151012/msqli.so
;
; If you only provide the name of the extension, PHP will look for it in its
; default extension directory.

```

Repetimos la operación para el archivo php.ini-production.

**nano /home/informatica/Escritorio/php.ini-production**

Una vez hechos los cambios anteriores copiamos los archivos a la carpeta original de Docker.

**docker cp /home/informatica/Escritorio/php.ini-development miservidorphp2:/usr/local/etc/php/**

**docker cp /home/informatica/Escritorio/php.ini-production miservidorphp2:/usr/local/etc/php/**

```

root@informatica-villa:/home/informatica# docker cp /home/informatica/Escritorio/php.ini-development miservidorphp2:/usr/local/etc/php/
Successfully copied 72.2kB to miservidorphp2:/usr/local/etc/php/
root@informatica-villa:/home/informatica# docker cp /home/informatica/Escritorio/php.ini-production miservidorphp2:/usr/local/etc/php/
Successfully copied 72.7kB to miservidorphp2:/usr/local/etc/php/

```

Después de hacer las anteriores configuraciones reiniciamos el servicio de Docker de miservidorphp2.

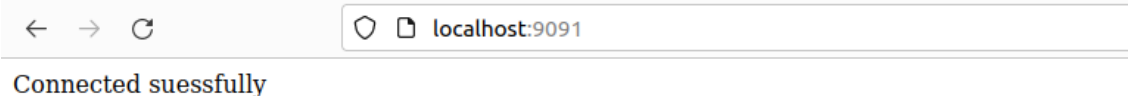
**docker restart miservidorphp2**

```

root@informatica-villa:/home/informatica# docker restart miservidorphp2
miservidorphp2
root@informatica-villa:/home/informatica#

```

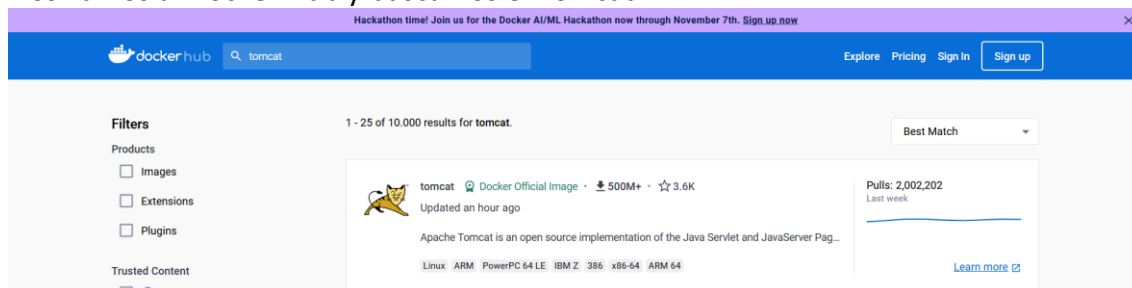
Ahora si accedemos desde el navegador web ya no tenemos el problema con el conector mysqli u nos saldrá el mensaje **Connected successfully**.



The screenshot shows a web browser window with the address bar displaying 'localhost:9091'. Below the address bar, the text 'Connected successfully' is visible.

### 3. Instalar contenedor con Tomcat y realizar pruebas de funcionamiento.

Nos vamos al Docker Hub y buscamos el Tomcat.



En este caso nos vamos a descargar la versión oficial, seleccionando la imagen oficial y copiando el comando siguiente y ejecutándolo en la terminal.



En este caso para mantener la sintonía con el Tomcat que hemos instalado en las actividades anteriores vamos a descargarnos las 9.0.80. Para ello buscamos en el listado de posibles imágenes disponibles y vemos que la que está disponible es la 9.0.81.

9.0.81-jdk11-temurin-jammy, 9.0-jdk11-temurin-jammy, 9-jdk11-temurin-jammy, 9.0.81-jdk11-temurin, 9.0-jdk11-temurin, 9-jdk11-temurin, 9.0.81-jdk11, 9.0-jdk11, 9-jdk11

#### **docker pull tomcat:9.0.81-jdk11-temurin-jammy**

```

root@informatica-villa:/home/informatica# docker pull tomcat:9.0.81-jdk11-temurin-jammy
9.0.81-jdk11-temurin-jammy: Pulling from library/tomcat
43f89b94cd7d: Pull complete
39769250f2be: Pull complete
7d78bd96cc6f: Pull complete
609b859ad36b: Pull complete
442b4ebcd69d: Pull complete
9f14cf35d8aa: Pull complete
5b4604023e80: Pull complete
a244838ff031: Pull complete
Digest: sha256:abbad54e2ea2b672344da9d2cdc4820e84eb6f2f0ce0c91265419be8130c7e2d
Status: Downloaded newer image for tomcat:9.0.81-jdk11-temurin-jammy
docker.io/library/tomcat:9.0.81-jdk11-temurin-jammy
root@informatica-villa:/home/informatica#

```

Comprobamos las imágenes cargadas.

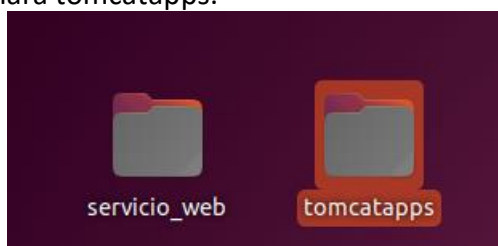
## **docker images**

```
root@informatica-villa:/home/informatica# docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
tomcat	9.0.81-jdk11-temurin-jammy	352d1953ce80	5 hours ago	412MB
mysql	5.7	3b85be0b10d3	2 months ago	581MB
mysql	<none>	a5b7ceed4074	2 months ago	581MB
hello-world	latest	9c7a54a9a43c	5 months ago	13.3kB
pensiero/apache-php-mysql	latest	c04be4cb614f	2 years ago	589MB
php	7.0-apache	aa67a9c9814f	4 years ago	368MB

```
root@informatica-villa:/home/informatica#
```

Para desplegar las aplicaciones ponemos una carpeta en el escritorio de nuestra máquina que se llamará tomcatapps.



Creamos el contenedor.

```
docker run -d --name servidortomcat -p 8081:8080 -v /home/informatica/Escritorio/tomcatapps:/usr/local/tomcat/webapps/ tomcat:9.0.81-jdk11-temurin-jammy
```

```
root@informatica-villa:/home/informatica# docker run -d --name servidortomcat -p 8081:8080 -v /home/informatica/Escritorio/tomcatapps:/usr/local/tomcat/webapps/ tomcat:9.0.81-jdk11-temurin-jammy
27b041053c8bace8dfec1527c68672365d249852412cf00e244a965ed4fde64a
root@informatica-villa:/home/informatica#
```

Comprobamos que se ha creado el contenedor.

## **docker ps -a**

```
root@informatica-villa:/home/informatica# docker ps -a
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
27b041053c8b	tomcat:9.0.81-jdk11-temurin-jammy	"catalina.sh run"	39 seconds ago	Up 38 seconds	0.0.0.0:8081->8080/tcp, :::8081->8080/tcp	servidortomcat
15a1daeb0b4b	php:7.0-apache	"docker-php-entrypoint..."	4 hours ago	Up 49 minutes	0.0.0.0:9091->80/tcp, :::9091->80/tcp	ntservidorphp2
b3a4edfcd73	php:7.0-apache	"docker-php-entrypoint..."	4 hours ago	Up 4 hours	0.0.0.0:9090->80/tcp, :::9090->80/tcp	ntservidorphp
1fc75a631bc4	mysql:5.7	"docker-entrypoint.s..."	4 hours ago	Up 49 minutes	33060/tcp, 0.0.0.0:3307->3306/tcp, :::3307->3306/tcp	basedatos
ef97264d99ce	pensiero/apache-php-mysql	"/usr/sbin/apache2ct..."	5 hours ago	Up 5 hours	443/tcp, 0.0.0.0:81->80/tcp, :::81->80/tcp	lamp
c589b9590ec	hello-world	"/hello"	9 days ago	Exited (0) 9 days ago		optimistic_cort

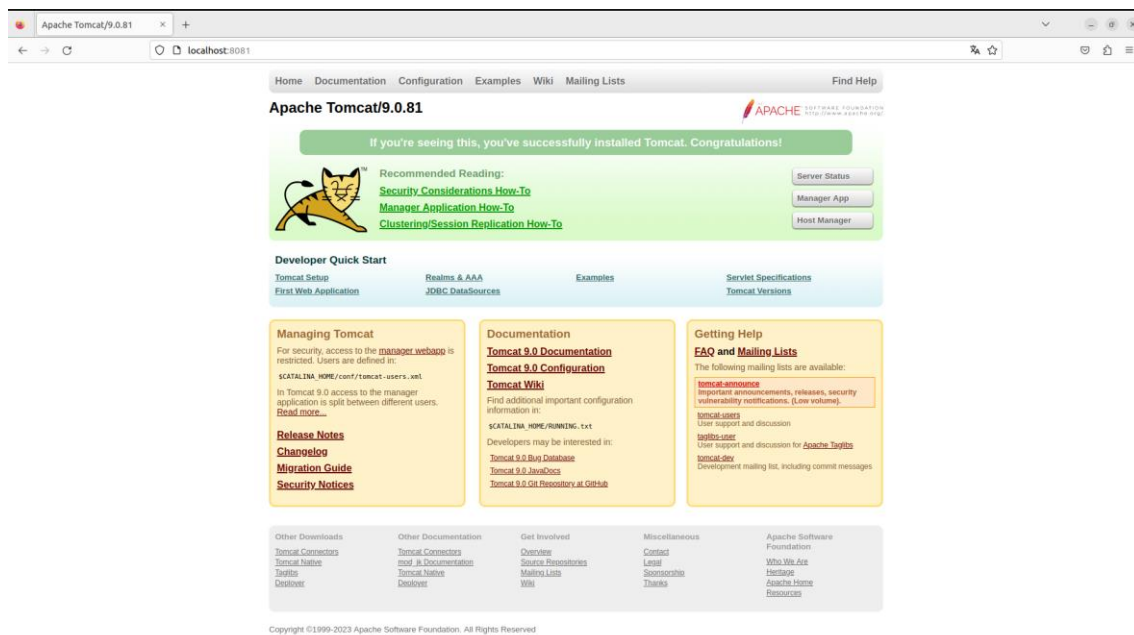
```
root@informatica-villa:/home/informatica#
```

Si queremos tener la página con la que nos recibe el Tomcat como en la instalación que hicimos en su momento podemos hacer una copia de la misma hacia la carpeta tomcatapps si disponemos de ella en este servidor si no la copiamos desde la otra máquina. Hay que copiar la carpeta ROOT de webapps y todo su contenido.

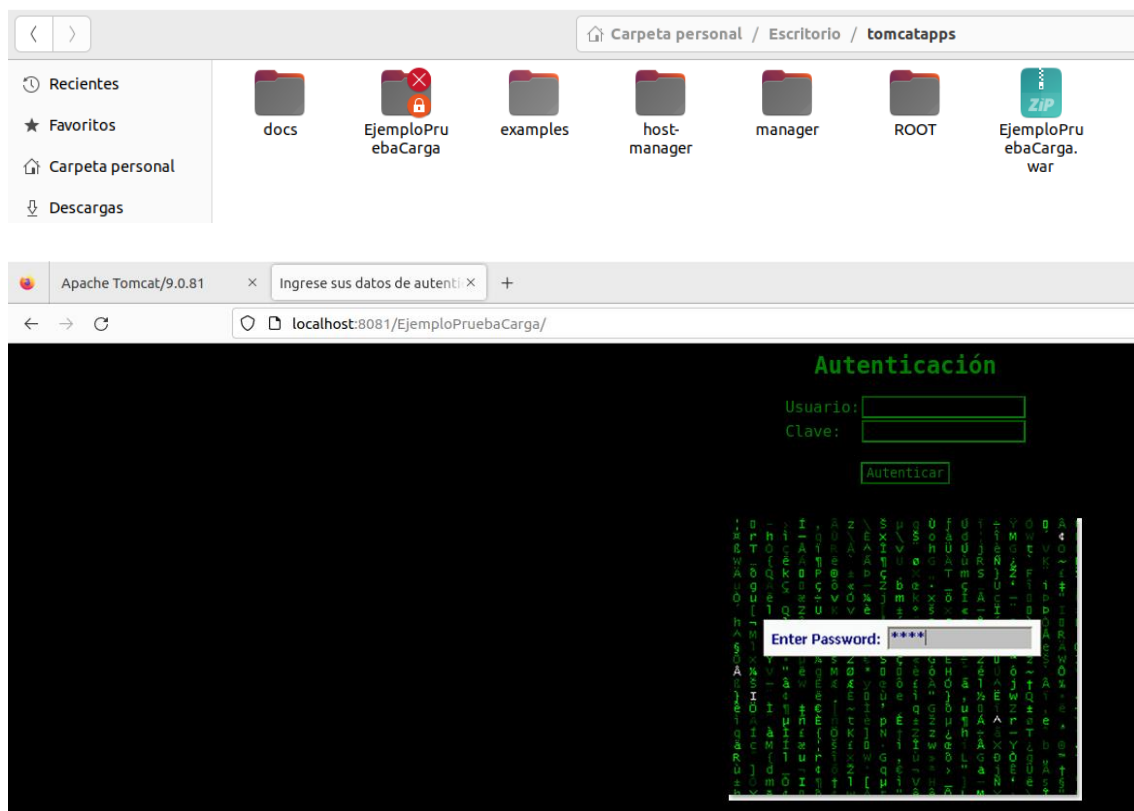
```
cd /opt/tomcat/webapps
```

```
cp -R ./ /home/informatica/Escritorio/tomcatapps/
```

Si todo ha ido bien nos debería de permitir ver en el navegador poniendo <http://localhost:8081> la página de inicio de Tomcat como veíamos cuando lo instalábamos de manera manual.

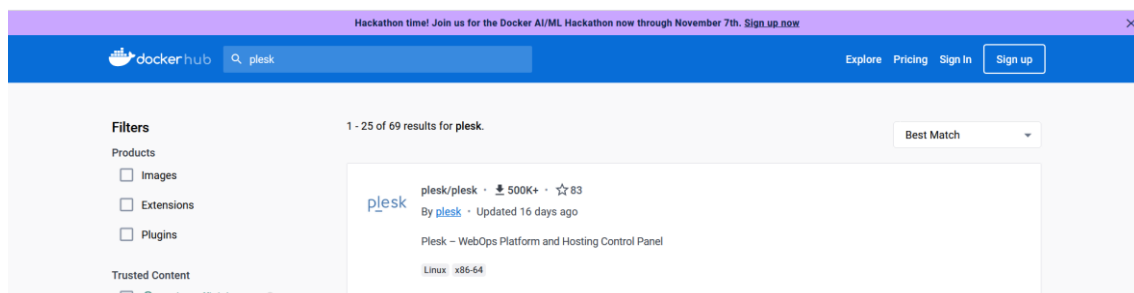


Permitiéndonos desplegar aplicaciones en la misma, para lo que tendríamos que configurar los xml correspondientes. O incluso podemos insertar de manera manual los archivos en la carpeta tomcatapps y podríamos acceder a las aplicaciones web.

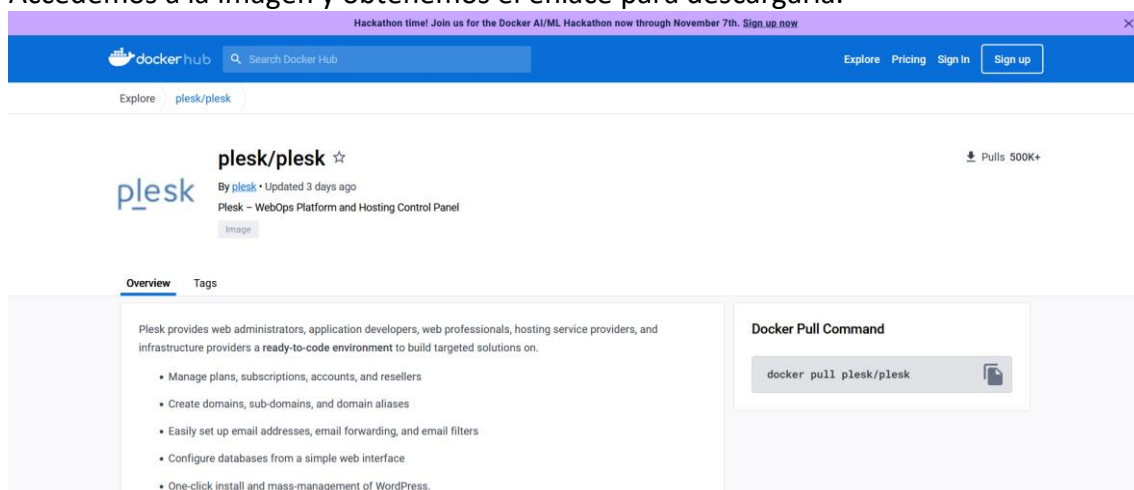


#### 4. Instalar contenedor de plesk y probarlo.

Al igual que hiciéramos con el Tomcat vamos a descargar de Docker Hub el Plesk.



Accedemos a la imagen y obtenemos el enlace para descargarla.



#### **docker pull plesk/plesk**

Si seguimos los pasos de la página (<https://hub.docker.com/r/plesk/plesk>) la podremos desplegar y configurar sin problemas. Incluso vienen las credenciales que debemos usar para acceder. En este caso deberíamos crear el contenedor hacia un puerto de la máquina host que no estuviera previamente ocupado p.e. 8082.