

MANUAL DE USUARIO
GESTIONAR UNA PAGINA WEB EN UN ENTORNO LAMP



Indice:

- 1- Introduccion
- 2- Instalación
- 3- Troubleshooting
- 4- Comprobacion

Introducción:

En esta práctica, se debe implantar una pagina web (PHP) en un entorno LAMP con Docker y Docker compose, utilizando xls.

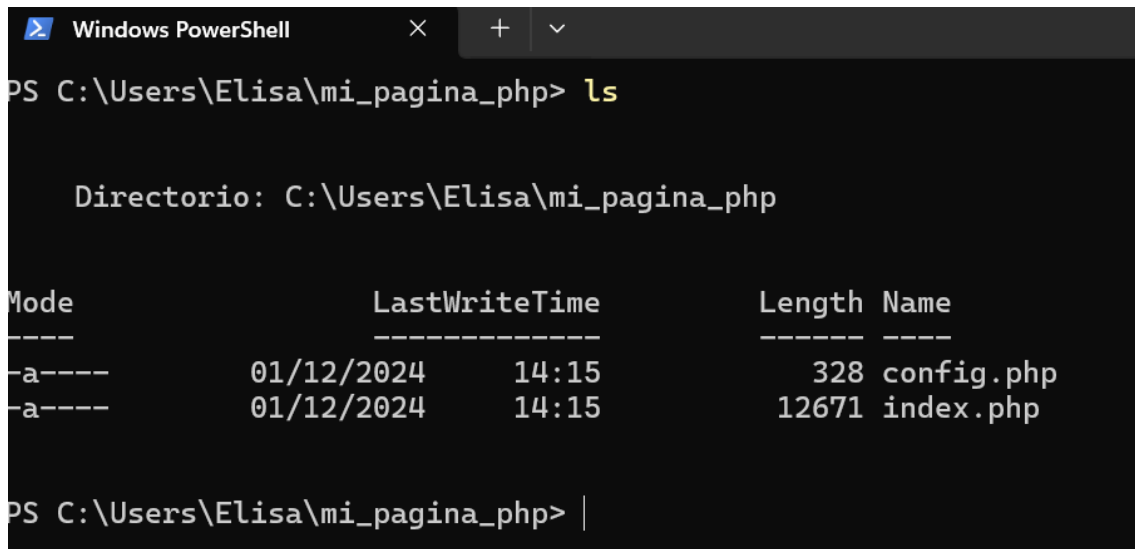
Instalación:

Los pasos para desarrollar la práctica:

- Crear un directorio mi_pagina_php

Entrar en el directorio y poner los ficheros php

En este caso hemos puesto el index.php de la práctica y el fichero config.php, puesto que tenemos que relacionar la conexión de MySQL, con los parámetros que aparecen en docker-compose.yml



```
Windows PowerShell
PS C:\Users\Elisa\mi_pagina_php> ls

Directorio: C:\Users\Elisa\mi_pagina_php

Mode                LastWriteTime         Length Name
----                -
-a----           01/12/2024   14:15             328 config.php
-a----           01/12/2024   14:15          12671 index.php

PS C:\Users\Elisa\mi_pagina_php> |
```

Editamos el fichero index.php y colocamos la conexión que aparece en config.php.

El fichero index tendrá un comienzo un poco distinto al original. El resto de fichero es igual.

```

docker-compose.yml  index.php
> Users > Elisa > mi_pagina_php > index.php
1  <?php
2      // Conectar a la base de datos
3      $servername = "db";
4      $username = "user";
5      $password = "userpassword";
6      $dbname = "my_database";
7      # ojo con los parametros
8
9      // Crear la conexión
10     $conn = new mysqli($servername, $username, $password, $dbname);
11
12     // Verificar la conexión
13     if ($conn->connect_error) {
14         die("Conexión fallida: " . $conn->connect_error);
15     }
16     echo "Conexión exitosa!";
17
18
19     // Consulta para obtener los proyectos
20     $sql = "SELECT titulo, descripcion, imagen FROM proyectos";
21     $result = $conn->query($sql);

```

Se sale del directorio o carpeta \$ cd -

Se genera el fichero docker-compose.yml

```

services:
# Servicio para MySQL

db:
    image: mysql:8.0
    container_name: mysql-container
    environment:
        MYSQL_ROOT_PASSWORD: rootpassword    # Contraseña para el usuario root de
MySQL
        MYSQL_DATABASE: mydatabase          # Nombre de la base de datos a crear
        MYSQL_USER: user                    # Nombre del usuario
        MYSQL_PASSWORD: userpassword        # Contraseña del usuario
    volumes:
        - mysql_data:/var/lib/mysql        # Persistencia de datos para MySQL
    networks:
        - lamp_network

# Servicio para Apache con PHP

web:
    image: php:7.4-apache

```

```
container_name: apache-php-container

volumes:
  - ./mi_pagina_php:/var/www/html      # Mapea el directorio local ./php al directorio
de trabajo en Apache

depends_on:
  - db

ports:
  - "8080:80"                          # Mapea el puerto 8080 de tu máquina al puerto 80 del
contenedor

networks:
  - lamp_network

# Servicio phpMyAdmin
phpmyadmin:
image: phpmyadmin/phpmyadmin
container_name: phpmyadmin-container
environment:
  PMA_HOST: db                          # Nombre del contenedor de MySQL
  PMA_PORT: 3306                        # Puerto del contenedor de MySQL
  PMA_USER: user                        # Usuario de la base de datos
  PMA_PASSWORD: userpassword           # Contraseña de la base de datos
ports:
  - "8081:80"                          # Puerto para acceder a phpMyAdmin
network:
  - lamp-network

volumes:
mysql_data:                            # Volumen persistente para MySQL
  driver: local

networks:
```

```
lamp_network:  
  driver: bridge
```

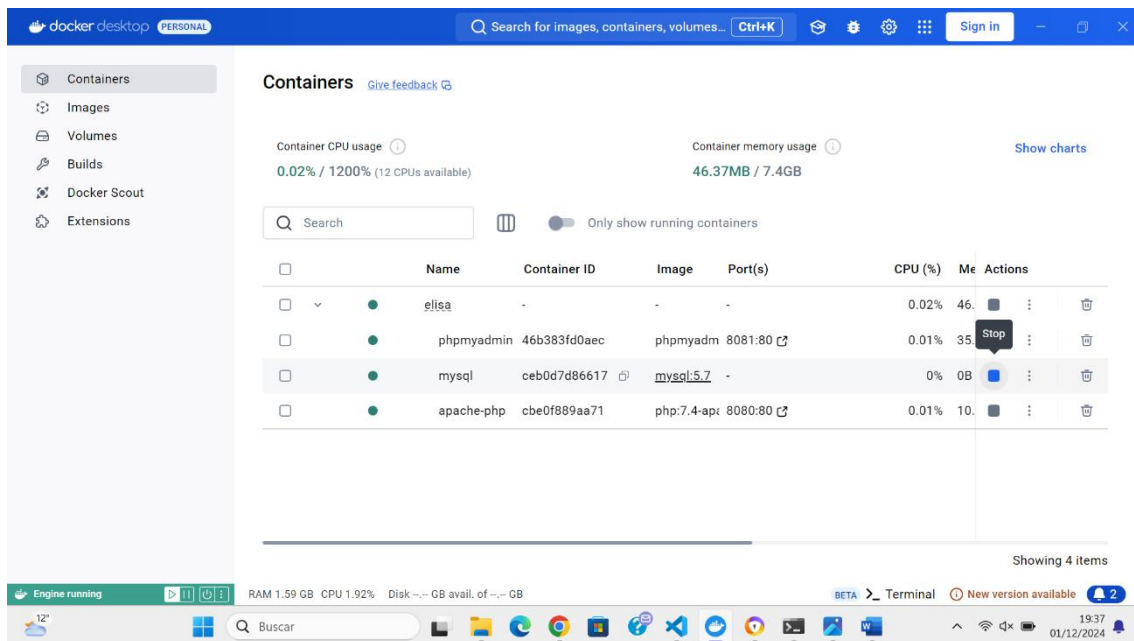
Se sale del directorio o carpeta \$ cd -

Se genera el fichero docker-compose.yml

Se ejecutan los contenedores

```
alberto@AlbertoCPort:/mnt/c/Users/alber$ docker-compose up -d  
[+] Running 3/3  
✓ Container phpmyadmin-container Running  
✓ Container mysql-container Running  
✓ Container apache-php-container Running
```

Se pasa ahora, al Docker Desktop y se comprueba que están los contenedores funcionando.



Si se pulsa en el contenedor apache, puerta 8080:80, se tendrá la página php

Troubleshooting:

Uncaught Error: Class 'mysqli' not found in /var/www/html/index.php:14



tack trace: #0 {main} thrown in /var/www/html/index.php on line 14

Si se pulsa en el contenedor phpmyadmin, puerta 8081:80, se accederá a phpMyadmin, aparece que no existe conexión con MySQL



Fatal error: Uncaught Error: Class 'mysqli' not found in /var/www/html/index.php:14
Stack trace: #0 {main} thrown in /var/www/html/index.php on line 14

Mirando la línea 14, parece ser que no se conecta con el servidor mysql

El problema radica en que no está activada la extensión mysqli ide conexión a MySQL.

Para ello, se va a comprobar si está activada, o no.

1.- Acceder al servidor PHP

\$ docker exec -it apache-php bash

2.- Comprobar si está instalada, ejecutamos este comando en el servidor

php -m | grep mysqli

Se comprueba que no está habilitada, puesto que no aparece en la lista

```
elisa@ELISA-C-H:/mnt/c/Users/Elisa$ docker exec -it apache-php bash
root@cbe0f889aa71:/var/www/html# php -m | grep mysqli
root@cbe0f889aa71:/var/www/html#
```

Como el servidor sigue ejecutándose

\$ docker exec -it apache-php bash

apt-get update

```

elisa@ELISA-C-H:/mnt/c/Users/Elisa$ docker exec -it apache-php bash
root@fea3e374a582:/var/www/html# apt-get update
Get:1 http://deb.debian.org/debian bullseye InRelease [116 kB]
Get:2 http://deb.debian.org/debian-security bullseye-security InRelease [27.2 kB]
Get:3 http://deb.debian.org/debian bullseye-updates InRelease [44.1 kB]
Get:4 http://deb.debian.org/debian bullseye/main amd64 Packages [8066 kB]
Get:5 http://deb.debian.org/debian-security bullseye-security/main amd64 Packages [317 kB]
Get:6 http://deb.debian.org/debian bullseye-updates/main amd64 Packages [18.8 kB]
Fetched 8589 kB in 2s (4181 kB/s)
Reading package lists... Done
root@fea3e374a582:/var/www/html# |

```

apt-get install -y libmysqlclient-dev

docker-php-ext-install mysqli

```

+ strip --strip-all modules/mysqli.so
Installing shared extensions:      /usr/local/lib/php/extensions/no-debug-non-zts-20190902/
Installing header files:          /usr/local/include/php/
find . -name \*.gcno -o -name \*.gcda | xargs rm -f
find . -name \*.lo -o -name \*.o | xargs rm -f
find . -name \*.la -o -name \*.a | xargs rm -f
find . -name \*.so | xargs rm -f
find . -name *.libs -a -type d|xargs rm -rf
rm -f libphp.la      modules/* libs/*
root@cbe0f889aa71:/var/www/html# exit
exit
elisa@ELISA-C-H:/mnt/c/Users/Elisa$ |

```

Al finalizar la instalación, después de salir del contenedor

Se reinicia al contenedor PHP

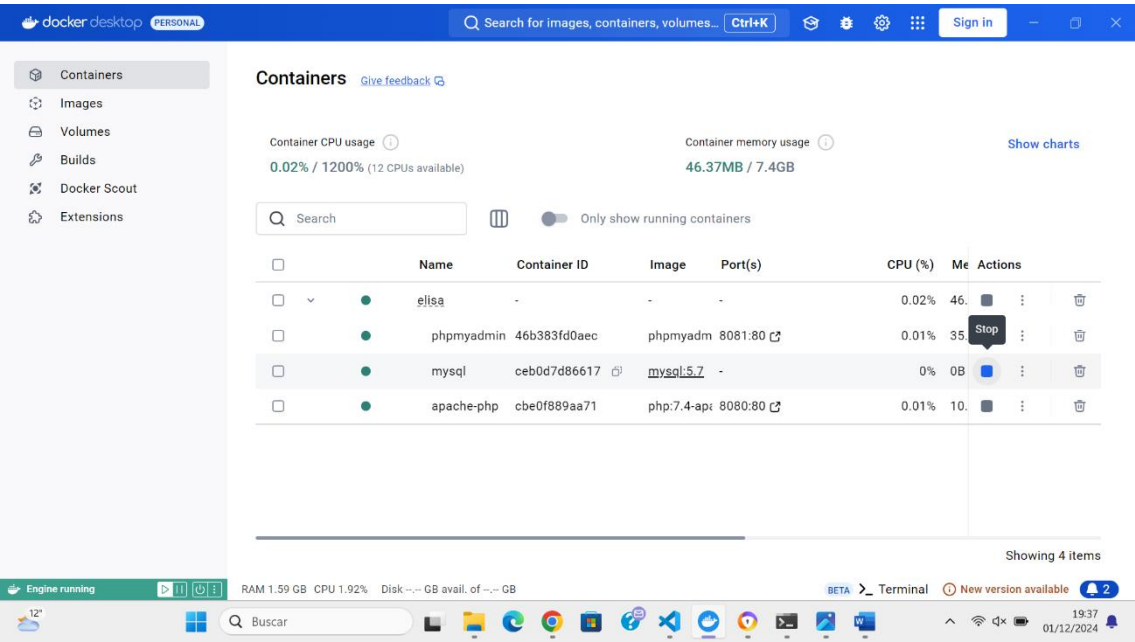
docker restart apache-php

```

root@cbe0f889aa71:/var/www/html# exit
exit
elisa@ELISA-C-H:/mnt/c/Users/Elisa$ docker restart apache-php
apache-php
elisa@ELISA-C-H:/mnt/c/Users/Elisa$ |

```

Se vuelve otra vez al docker desktop y se vuelve a ir a las páginas web.



Si se pulsa en el contenedor apache, puerta 8080:80, se tendrá la página php.



Si se pulsa en el contenedor phpmyadmin, puerta 8081:80, se accederá a phpMyadmin

