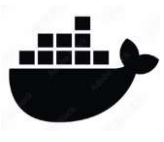
PÁGINA WEB (VERSIÓN PHP) SOBRE LAMP

ÍNDICE



Pasos previos

Lo primero de todo, debemos tener clara la infraestructura del proyecto y qué tenemos que incluir en él.

Teniendo en cuenta que estoy usando Docker Desktop y VSCode como IDE realizaremos los siguientes pasos.

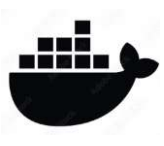
EN VSCode creamos una nueva carpeta llamada 'paginaWEB' en la que vamos a crear una subcarpeta 'html' donde copiaremos todos los archivos de nuestra página web: index.html, index.php, db.php, etc. Si preferimos, podemos crear otra subcarpeta para los archivos 'sql'.

Estructura del proyecto

Deberíamos tener una carpeta con una forma parecida a ésta en nuestro VSCode:

Configuración de archivo docker-compose.yml

Crearemos el archivo docker-compose.yml con la configuración de los servicios:

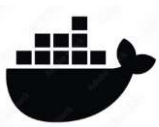


paginaWEB > docker-compose.yml

```yaml

services:
  web:
    build:
      context: .
      dockerfile: Dockerfile
    container_name: lamp_web
    ports:
      - "8080:80"                 # Exponemos el puerto 80 del contenedor al 8080 del host
    volumes:
      - ./html:/var/www/html      # Montamos los archivos de la página web
    depends_on:
      - db                        # Aseguramos que el contenedor de MySQL se inicie antes
    networks:
      - lamp_net

  db:
    image: mariadb:latest         # Usamos MariaDB como base de datos
    container_name: lamp_db
    environment:
      MYSQL_DATABASE: proyectos       # Nombre de la base de datos
      MYSQL_USER: root         # Usuario
      MYSQL_PASSWORD: root_password # Contraseña
      MYSQL_ROOT_PASSWORD: root_password
    volumes:
      - db_data:/var/lib/mysql     # Almacenamos los datos de la base de datos
      - ./sql:/docker-entrypoint-initdb.d # Opcional: scripts SQL iniciales
    ports:
      - "3306:3306"               # Exponemos el puerto de MySQL
    networks:
      - lamp_net

volumes:
  db_data:

networks:
  lamp_net:
    driver: bridge
```

Configuración de archivo dockerfile

Creamos el archivo Dockerfile para el contenedor web:

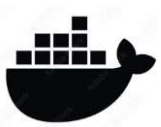


paginaWEB > dockerfile > COPY

```
1
2   # Usamos la imagen oficial de PHP con Apache
3   FROM php:8.1-apache
4
5   # Instalar extensiones necesarias para PHP y Apache
6   RUN apt-get update && apt-get install -y \
7       libzip-dev \
8       unzip \
9       mariadb-client \
10      && docker-php-ext-install pdo_mysql mysqli
11
12  # Habilitamos módulos de Apache (opcional: mod_rewrite para URLs amigables)
13  RUN a2enmod rewrite
14
15  # Copiamos la carpeta de tu sitio web al directorio de Apache (opcional si no usas volúmenes)
16  COPY ./html /var/www/html
17
18  # Configuramos permisos para el directorio
19  RUN chown -R www-data:www-data /var/www/html && chmod -R 755 /var/www/html
20
21  # Exponemos el puerto 80 del contenedor
22  EXPOSE 80
23
24  # Comando para iniciar Apache
25  CMD ["apache2-foreground"]
```

Archivo PHP

Deberemos tener un archivo parecido a ‘db.php’ en el que se configura la conexión a la base de datos. Tendremos que fijarnos que los parámetros sean los mismo a los de nuestro archivo ‘docker-compose.yml’:



paginaWEB > html > db.php

```php
<?php
// Parámetros de conexión
$servername = "db";
$username = "root";
$password = "root_password";
$dbname = "proyectos";

// Crear conexión
$conn = new mysqli($servername, $username, $password, $dbname);

// Comprobar conexión
if ($conn->connect_error) {
    die("Conexión fallida: " . $conn->connect_error);
} else {
    echo "Conexión exitosa a la base de datos.";
}
?>
```

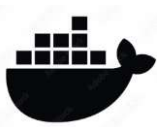
Archivo ‘index.php’

En el archivo ‘index.php’, verificaremos que todo cuadre en la consulta para obtener los proyectos:

paginaWEB > html > index.php

```php

<?php
// Incluir el archivo de conexión
include 'db.php';

// Consulta para obtener los proyectos
$sql = "SELECT * FROM proyectos";
$result = $conn->query($sql);
?>
```

Con los demás archivos de la página web no debería haber problema y no tendríamos que hacer configuraciones extras.



Contenedores

Construiremos los contenedores web y db ejecutando en la terminal de VSCode lo siguiente (ejecutar los comandos en la ruta del proyecto): docker-compose build

```
PS C:\Users\Pablo\Documents\GitHub\ContenedoresDocker\paginaWEB> docker-compose build
[+] Building 2.1s (11/11) FINISHED
                                                                    docker:desktop-linux
 => [web internal] load build definition from dockerfile
                                                                                   0.0s
 => => transferring dockerfile: 769B
                                                                                   0.0s
 => [web internal] load metadata for docker.io/library/php:8.1-apache
                                                                                   1.1s
 => [web internal] load .dockerignore
                                                                                   0.0s
 => => transferring context: 2B
                                                                                   0.0s
 => [web 1/5] FROM docker.io/library/php:8.1-apache@sha256:e3d4b282a162873e05dfa6df3429e45605049095a5a
                                                                                   0.0s
```

Después, tendremos que levantarlos con: docker-compose up –d

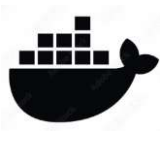
```
PS C:\Users\Pablo\Documents\GitHub\ContenedoresDocker\paginaWEB> docker-compose up -d
[+] Running 3/3
 ✓ Network paginaweb_lamp_net  Created                                             0.0s
 ✓ Container lamp_db           Started                                             0.5s
 ✓ Container lamp_web          Started                                             0.7s
```

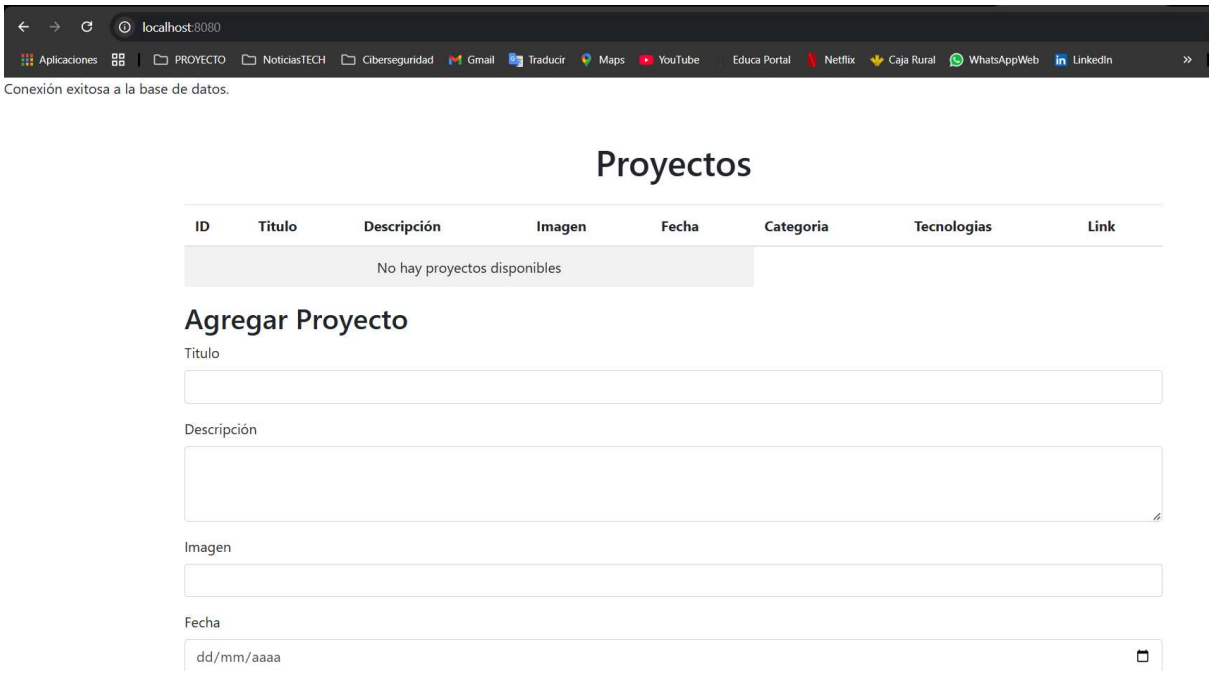
Y para verificar que están en funcionamiento ejecutaremos: docker ps

```
PS C:\Users\Pablo\Documents\GitHub\ContenedoresDocker\paginaWEB> docker ps
CONTAINER ID   IMAGE           COMMAND                  CREATED          STATUS          PORTS
          NAMES
7fcc2b5c021c   paginaweb-web   "docker-php-entrypoi…"   40 seconds ago   Up 40 seconds   0.0.0.0:8080->80/tcp
          lamp_web
fc716c8ca567   mariadb:latest  "docker-entrypoint.s…"   40 seconds ago   Up 40 seconds   0.0.0.0:3306->3306/tcp
          lamp_db
2e864f6677df   mysql           "docker-entrypoint.s…"   5 days ago       Up 23 seconds   3306/tcp, 33060/tcp, 0.0.0.0:3308->
3308/tcp   roleplayinggame-database
PS C:\Users\Pablo\Documents\GitHub\ContenedoresDocker\paginaWEB>
```

Acceder desde navegador

Si nada ha fallado, deberíamos poder visualizar nuestra aplicación desde http://localhost:8080 en el navegador.





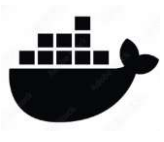
TROUBLESHOOTING

- Problema 'Connection refused' al conectarse a la base de datos:
  - Causa: Usar "ocalhost" en vez del nombre del servicio Docker.
  - Solución: en 'db.php' usar $servername = "db" en vez de "localhost"

```php
<?php
// Parámetros de conexión
$servername = "db";
$username = "root";
$password = "root_password";
$dbname = "proyectos";
```

- Problema tabla 'proyectos' no existe:
  - Causa: la tabla no fue creada
  - Solución: acceder al contenedor de la base de datos y verifica las tablas con: docker exec –it lamp_db bash, y una vez dentro: mysql –u root –p (USE proyectos; SHOW TABLES;)

    Si la tabla no existe, la creamos con: CREATE TABLE proyectos y la configuración que le queramos dar.



- Problema permisos incorrectos en /var/www/html
  - Causa: los permisos del directorio no son los adecuados
  - Solución: ajustar los permisos dentro del contenedor:

    ```
    docker exec -it lamp_web bash

    chown -R www-data:www-data /var/www/html

    chmod -R 755 /var/www/html
    ```