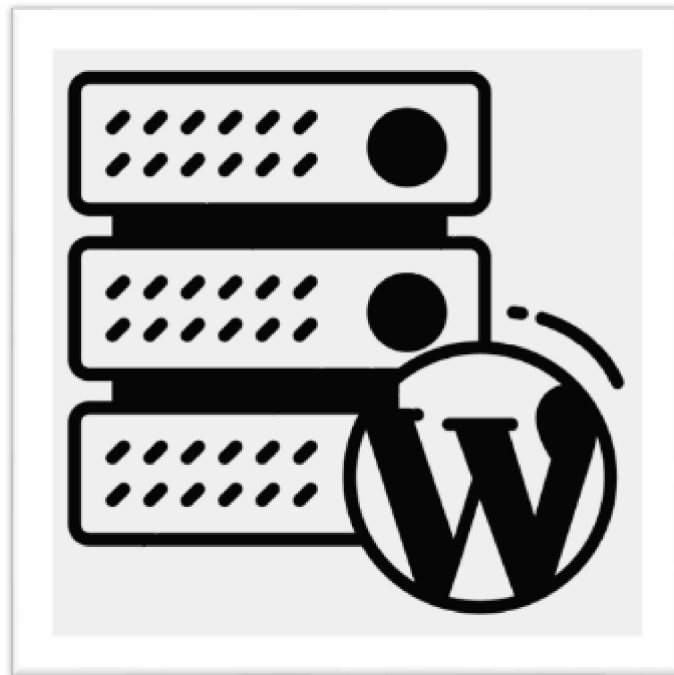


CONTENEDORES WORDPRESS



ÍNDICE

CMS WordPress	2
Despliegue del CMS WordPress	2
Pasos previos	2
Instalación.....	3
Parámetros	4
Ping	5
Bind-address	6
Drupal.....	7
TROUBLESHOOTING	10



CMS WordPress

En este manual vamos a explicar los pasos para desplegar el CMS WordPress.

WordPress es el sistema de gestión de contenidos, enfocado a la creación de cualquier tipo de página web, más famoso en todo el mundo.

Podremos ejecutar todos los comandos desde CMD usando WSL o desde la terminal de nuestro IDE (VSCode en mi caso).



Despliegue del CMS WordPress

Pasos previos

Antes de todo, deberemos tener 'Docker' y 'Docker-compose' instalados en la máquina. Y si estamos trabajando con GitHub en paralelo, tenerlo bien configurado para trabajar en nuestro repositorio.

En nuestra carpeta del proyecto creamos un archivo llamado 'docker-compose.yml' y añadimos la siguiente configuración:

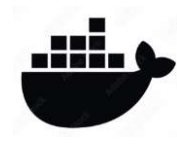


```
docker-compose.yml
Wordpress > docker-compose.yml
1 #version: '3.8'
2
3 services:
4   db:
5     image: mariadb:latest
6     container_name: servidor_mysql
7     networks:
8     - red_wp
9     volumes:
10    - db_data:/var/lib/mysql
11    environment:
12      MYSQL_DATABASE: bd_wp
13      MYSQL_USER: user_wp
14      MYSQL_PASSWORD: asdasd
15      MYSQL_ROOT_PASSWORD: asdasd
16
17   wordpress:
18     image: wordpress:latest
19     container_name: servidor_wp
20     networks:
21     - red_wp
22     ports:
23     - "80:80"
24     volumes:
25     - wordpress_data:/var/www/html/wp-content
26     environment:
27       WORDPRESS_DB_HOST: mysql
28       WORDPRESS_DB_USER: example username
29       WORDPRESS_DB_PASSWORD: example password
30       WORDPRESS_DB_NAME: wordpress
31
32   volumes:
33     db_data:
34     wordpress_data:
35
36   networks:
37     red_wp:
38     driver: bridge
39
```

Instalación

A continuación, pondremos los comandos necesarios en la terminal para crear el entorno de Wordpress:

> Creamos la red: `docker network create red_wp`



> Contenedor MySQL: `docker run -d --name servidor_mysql --network red_wp -v /opt/mysql_wp:/var/lib/mysql -e MYSQL_DATABASE=bd_wp -e MYSQL_USER=user_wp -e MYSQL_PASSWORD=asdasd -e MYSQL_ROOT_PASSWORD=asdasd mariadb`

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\Pablo\Documents\GitHub\ContenedoresDocker\Wordpress> docker network create red_wp
a296a30f8125387594c60fb81b3d6589223bd67169a5a854200a3da4b7ef61f9
PS C:\Users\Pablo\Documents\GitHub\ContenedoresDocker\Wordpress> docker run -d --name servidor_mysql --network red_wp -v /opt/
/mysql_wp:/var/lib/mysql -e MYSQL_DATABASE=bd_wp -e MYSQL_USER=user_wp -e MYSQL_PASSWORD=asdasd -e MYSQL_ROOT_PASSWORD=asdasd
mariadb
```

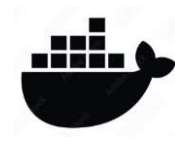
> Contenedor WordPress: `docker run -d --name servidor_wp --network red_wp -v /opt/wordpress:/var/www/html/wp-content -e WORDPRESS_DB_HOST=servidor_mysql -e WORDPRESS_DB_USER=user_wp -e WORDPRESS_DB_PASSWORD=asdasd -e WORDPRESS_DB_NAME=bd_wp -p 8080:80 wordpress`

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\Pablo\Documents\GitHub\ContenedoresDocker\Wordpress> docker run -d --name servidor_wp --network red_wp -v /opt/wor
dpress:/var/www/html/wp-content -e WORDPRESS_DB_HOST=servidor_mysql -e WORDPRESS_DB_USER=user_wp -e WORDPRESS_DB_PASSWORD=as
dasd -e WORDPRESS_DB_NAME=bd_wp -p 8080:80 wordpress
```

Parámetros

Vamos a ejecutar una instrucción Docker para visualizar el contenido del fichero 'wp-config.php' y verificar que los parámetros de conexión a la base de datos son los mismo que los indicados en las variables de entorno:

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\Pablo\Documents\GitHub\ContenedoresDocker\Wordpress> docker exec -it servidor_wp bash
root@bd6673829e5a:/var/www/html# cat /var/www/html/wp-config.php
```



```
Wordpress > docker-compose.yml
3  services:
17  wordpress:
24  volumes:
25  | - wordpress_data:/var/www/html/wp-content
26  environment:
27  | WORDPRESS_DB_HOST: mysql
28  | WORDPRESS_DB_USER: example username
29  | WORDPRESS_DB_PASSWORD: example password
30  | WORDPRESS_DB_NAME: wordpress
31
32  volumes:

}

// ** Database settings - You can get this info from your web host ** //
/** The name of the database for WordPress */
define( 'DB_NAME', getenv_docker('WORDPRESS_DB_NAME', 'wordpress') );

/** Database username */
define( 'DB_USER', getenv_docker('WORDPRESS_DB_USER', 'example username') );

/** Database password */
define( 'DB_PASSWORD', getenv_docker('WORDPRESS_DB_PASSWORD', 'example password') );

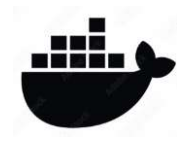
/**
 * Docker image fallback values above are sourced from the official WordPress install
 * https://github.com/WordPress/WordPress/blob/1356f6537220ffdc32b9dad2a6cdbe2d010b7a
 * (However, using "example username" and "example password" in your database is stro
 * ndom credentials!)
 */

/** Database hostname */
define( 'DB_HOST', getenv_docker('WORDPRESS_DB_HOST', 'mysql') );
```

Si los parámetros no coinciden modificar las variables en 'docker-compose.yml', guardar los cambios y reiniciar el contenedor (`docker-compose down, docker-compose up -d`)

Ping

Ahora vamos a ejecutar una instrucción docker para comprobar que desde WSL `servidor_wp` podemos hacer ping usando el nombre `servidor_mysql`. (Tendrás que instalar el paquete `iputils-ping` en el contenedor).



```

Administrador: Símbolo del sistema - docker exec -it servidor_wp bash

C:\Windows\System32>docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS
bd6673829e5a   wordpress     "docker-entrypoint.s..." 39 minutes ago Up 39 minutes 0.0.0.0:8080->80/tcp
5c575af57885   mariadb       "docker-entrypoint.s..." 41 minutes ago Up 41 minutes 3306/tcp
2e864f6677df   mysql        "docker-entrypoint.s..." 2 days ago    Up 6 seconds  3306/tcp, 33060/tcp, 0.0.0.0:3308->3308/tcp
roleplayinggame-database

C:\Windows\System32>docker exec -it servidor_wp bash
root@bd6673829e5a:/var/www/html# apt update && apt install -y iputils-ping

```

```

root@bd6673829e5a:/var/www/html# ping servidor_mysql
PING servidor_mysql (172.19.0.2) 56(84) bytes of data.
64 bytes from servidor_mysql.red_wp (172.19.0.2): icmp_seq=1 ttl=64 time=0.091 ms
64 bytes from servidor_mysql.red_wp (172.19.0.2): icmp_seq=2 ttl=64 time=0.048 ms
64 bytes from servidor_mysql.red_wp (172.19.0.2): icmp_seq=3 ttl=64 time=0.037 ms
64 bytes from servidor_mysql.red_wp (172.19.0.2): icmp_seq=4 ttl=64 time=0.078 ms
64 bytes from servidor_mysql.red_wp (172.19.0.2): icmp_seq=5 ttl=64 time=0.043 ms
64 bytes from servidor_mysql.red_wp (172.19.0.2): icmp_seq=6 ttl=64 time=0.043 ms
^C
--- servidor_mysql ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5140ms
rtt min/avg/max/mdev = 0.037/0.056/0.091/0.020 ms
root@bd6673829e5a:/var/www/html#

```

Bind-address

Después, vamos a visualizar el fichero ‘/etc/mysql/mariadb.conf.d/50-server.cnf’ del contenedor con la base de datos y comprobar cómo está configurado el parámetro ‘bind-address’:

Para ello, entramos a la BASH del contenedor ‘servidor_mysql’ ejecutando: `docker exec -it servidor_mysql bash`

Una vez dentro, visualizamos el fichero ‘/etc/mysql/mariadb.conf.d/50-server.cnf’

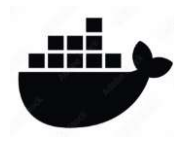
```

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  docker

PS C:\Users\Pablo\Documents\GitHub\ContenedoresDocker\Wordpress> docker exec -it servidor_mysql bash
root@5c575af57885:/# cat /etc/mysql/mariadb.conf.d/50-server.cnf

```

Comprobamos que el parámetro ‘bind-address’ está comentado. Esto quiere decir que generalmente permitirá conexiones desde todas las interfaces de red disponibles.




```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

# Broken reverse DNS slows down connections considerably and name resolve is
# safe to skip if there are no "host by domain name" access grants
#skip-name-resolve

# Instead of skip-networking the default is now to listen only on
# localhost which is more compatible and is not less secure.
#bind-address          = 127.0.0.1
```

Drupal

Como paso final, vamos a Instalar Drupal (CMS PHP) siguiendo la documentación de Docker Hub de la aplicación seleccionada:

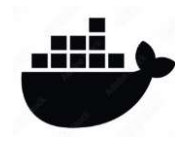
Ejecutaremos lo siguiente para instalarlo: `docker run --name drupal_db --network drupal_network -e MARIADB_ROOT_PASSWORD=root_password -e MARIADB_DATABASE=drupal -e MARIADB_USER=drupal_user -e MARIADB_PASSWORD=drupal_password -d mariadb:10.5`

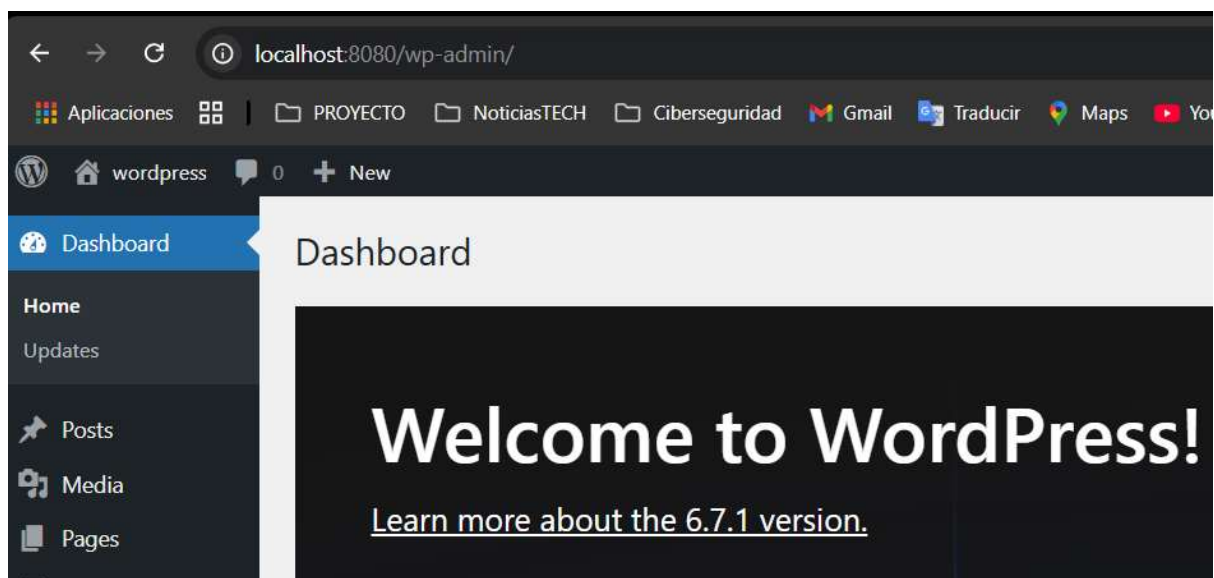
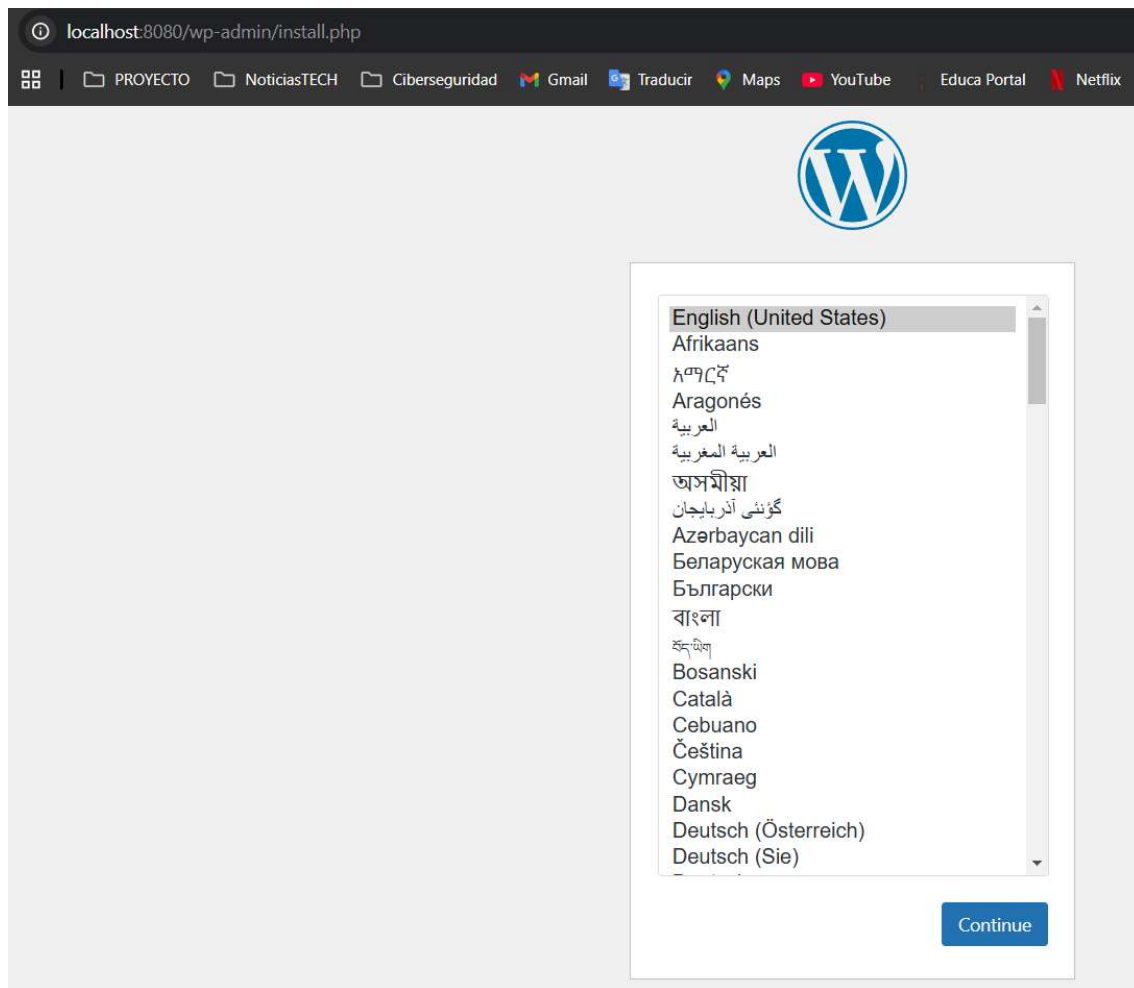
```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS powershell + - [] ... ^ X

PS C:\Users\Pablo\Documents\GitHub\ContenedoresDocker\Wordpress> docker run -d --name drupal --network red_wp -p 8081:80 drupal
Unable to find image 'drupal:latest' locally
latest: Pulling from library/drupal
193b424feb9d: Download complete
```

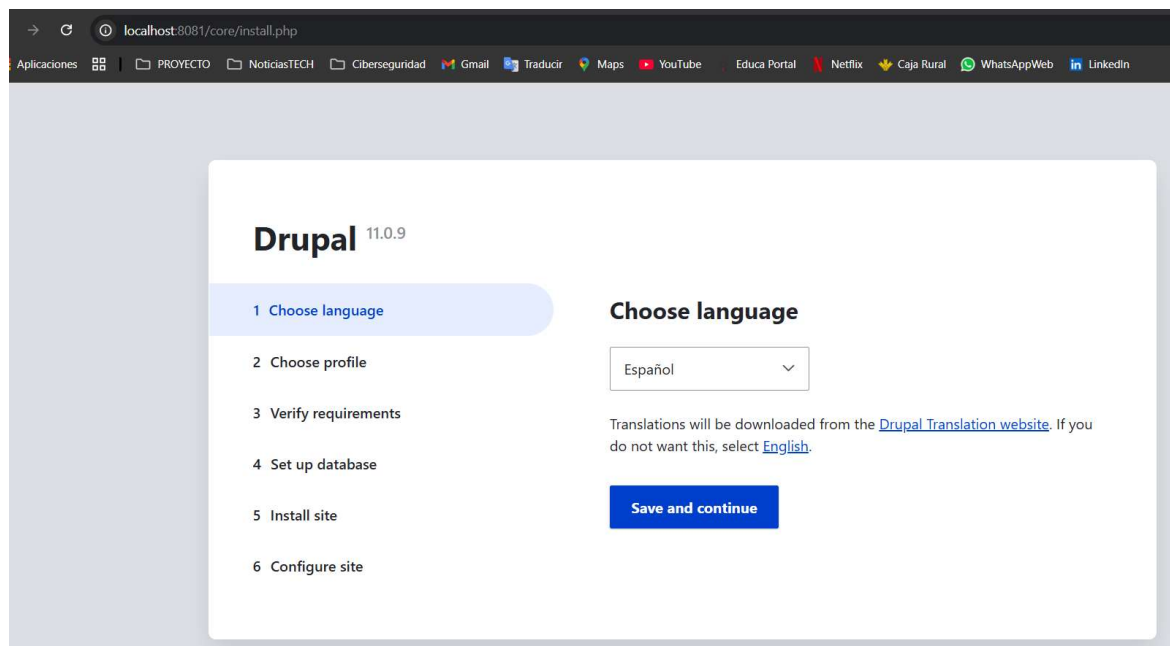
En nuestro navegador ejecutaremos:

<http://localhost:8080> para entrar a nuestra página de WordPress. Una vez dentro seguiríamos los pasos de instalación con los datos de nuestra configuración '.yaml' y comenzaríamos a trabajar.

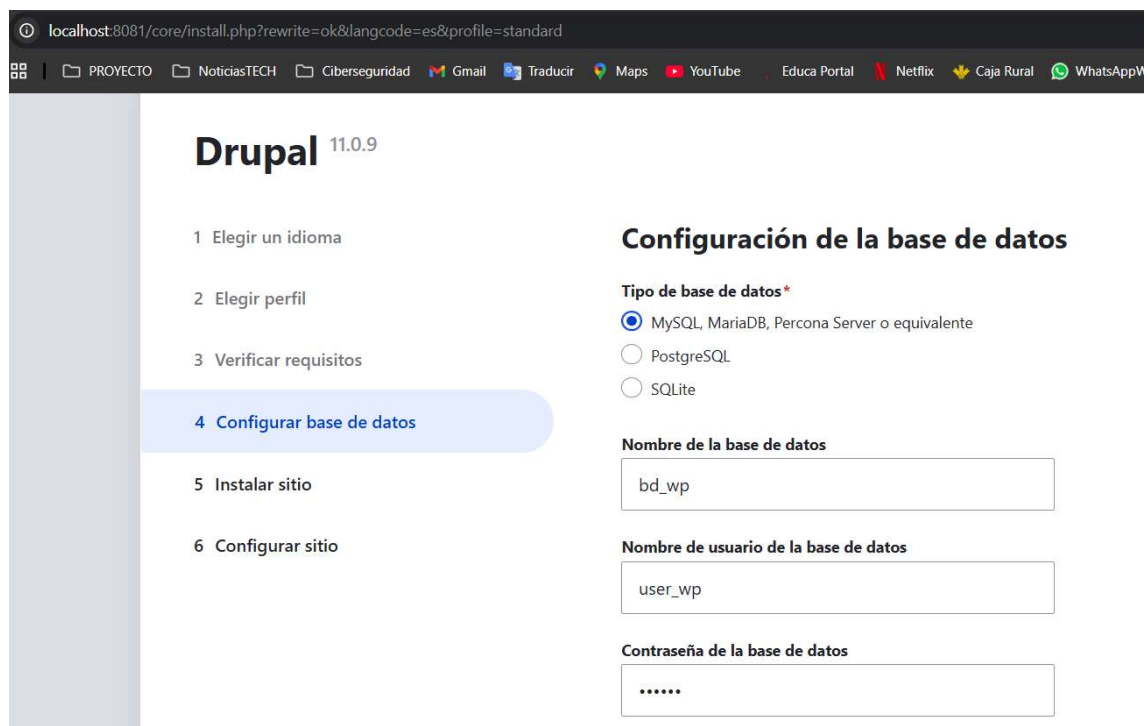




<http://localhost:8081> para entrar a nuestra página Drupal. Una vez dentro seguiríamos los pasos de configuración del sitio con nuestros datos configurados en '.yml' y comenzaríamos a trabajar.



The screenshot shows the Drupal 11.0.9 installation interface. On the left, a sidebar lists the installation steps: 1 Choose language, 2 Choose profile, 3 Verify requirements, 4 Set up database, 5 Install site, and 6 Configure site. The first step, 'Choose language', is highlighted. The main content area is titled 'Choose language' and features a dropdown menu with 'Español' selected. Below the dropdown, a note states: 'Translations will be downloaded from the [Drupal Translation website](#). If you do not want this, select [English](#).' A blue 'Save and continue' button is positioned at the bottom right of the main content area.



The screenshot shows the Drupal 11.0.9 installation interface at the 'Configuración de la base de datos' (Database configuration) step. The sidebar on the left lists the steps: 1 Elegir un idioma, 2 Elegir perfil, 3 Verificar requisitos, 4 Configurar base de datos, 5 Instalar sitio, and 6 Configurar sitio. The fourth step, 'Configurar base de datos', is highlighted. The main content area is titled 'Configuración de la base de datos' and includes a section for 'Tipo de base de datos *' (Database type). Three options are available: 'MySQL, MariaDB, Percona Server o equivalente' (selected with a radio button), 'PostgreSQL', and 'SQLite'. Below this, there are three text input fields: 'Nombre de la base de datos' (containing 'bd_wp'), 'Nombre de usuario de la base de datos' (containing 'user_wp'), and 'Contraseña de la base de datos' (containing six dots for a password).



TROUBLESHOOTING

- > Asegurarse de ejecutar los comandos de instalación y de ficheros en las rutas correspondientes.
- > Usar puertos que no estén en uso.
- > Vigilar la sintaxis de los comandos de instalación de los contenedores ya que son muy largos y pueden llevar a errores fácilmente.
- > Tener activada la opción de 'Use WSL2 Engine' en Docker Desktop para los pasos a realizar desde el CMD.

