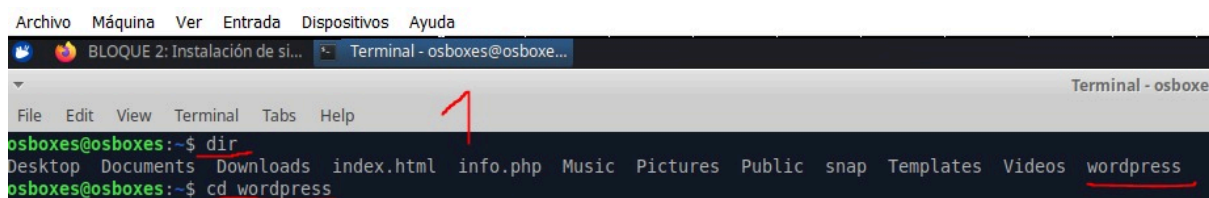


Sistemas Informáticos

Actividad Docker 4

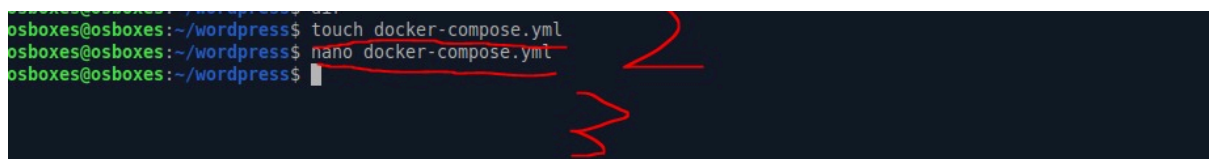
En esta práctica vamos a aprender a desplegar la aplicación Wordpress + MariaDB cada uno dentro de sus respectivos volúmenes.

1. Empezaremos creando un archivo dentro del cual introduciremos de una sola vez el nombre de las imágenes, el contenedor, el entorno, el puerto (para Wordpress) y el volumen correspondiente para cada una.
2. A la hora de guardar el archivo, si accedemos a nuestra dirección IP desde el navegador deberíamos acceder a Wordpress.



```
Archivo  Máquina  Ver  Entrada  Dispositivos  Ayuda
BLOQUE 2: Instalación de si... Terminal - osboxes@osboxe...
File Edit View Terminal Tabs Help
osboxes@osboxes:~$ dir
Desktop Documents Downloads index.html info.php Music Pictures Public snap Templates Videos wordpress
osboxes@osboxes:~$ cd wordpress
```

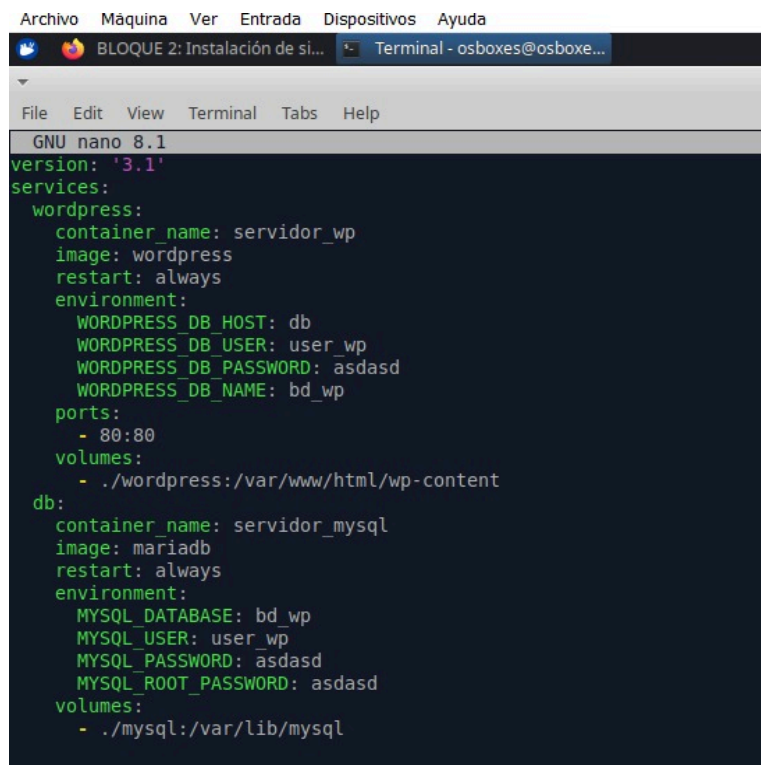
1) Aquí vemos con el comando `dir` el directorio “**wordpress**” que hemos creado de antemano. Esto ya hemos aprendido a hacerlo en las tareas anteriores. Accedemos usando el comando “**cd**”.



```
osboxes@osboxes:~/wordpress$ touch docker-compose.yml
osboxes@osboxes:~/wordpress$ nano docker-compose.yml
osboxes@osboxes:~/wordpress$
```

2) Usamos el comando “**touch docker-compose.yml**”. (Si nos deniega el acceso, escribimos “**sudo**” delante) Esto creará el fichero.

3) Usamos “**nano docker-compose.yml**” para acceder al editor.



```

Archivo  Máquina  Ver  Entrada  Dispositivos  Ayuda
BLOQUE 2: Instalación de si...  Terminal - osboxes@osboxe...
File  Edit  View  Terminal  Tabs  Help
GNU nano 8.1
version: '3.1'
services:
  wordpress:
    container_name: servidor_wp
    image: wordpress
    restart: always
    environment:
      WORDPRESS_DB_HOST: db
      WORDPRESS_DB_USER: user_wp
      WORDPRESS_DB_PASSWORD: asdasd
      WORDPRESS_DB_NAME: bd_wp
    ports:
      - 80:80
    volumes:
      - ./wordpress:/var/www/html/wp-content
  db:
    container_name: servidor_mysql
    image: mariadb
    restart: always
    environment:
      MYSQL_DATABASE: bd_wp
      MYSQL_USER: user_wp
      MYSQL_PASSWORD: asdasd
      MYSQL_ROOT_PASSWORD: asdasd
    volumes:
      - ./mysql:/var/lib/mysql

```

Dentro del editor, el texto a escribir es el que aparece en imagen. Esto creará las imágenes, contenedores, entornos y volúmenes automáticamente. Además asignará el puerto a la imagen de wordpress. Pulsamos **ctrl** y **x**, escribimos Y para “**SI**” y damos a enter.

El texto es:

```

version: '3.1'

services:
  wordpress:
    container_name: servidor_wp
    image: wordpress
    restart: always
    environment:
      WORDPRESS_DB_HOST: db
      WORDPRESS_DB_USER: user_wp
      WORDPRESS_DB_PASSWORD: asdasd
      WORDPRESS_DB_NAME: bd_wp
    ports:
      - 80:80
    volumes:
      - ./wordpress:/var/www/html/wp-content
  db:
    container_name: servidor_mysql
    image: mariadb
    restart: always
    environment:
      MYSQL_DATABASE: bd_wp
      MYSQL_USER: user_wp
      MYSQL_PASSWORD: asdasd
      MYSQL_ROOT_PASSWORD: asdasd
    volumes:
      - ./mysql:/var/lib/mysql

```

```

osboxes@osboxes:~/wordpress$ docker-compose up -d
Creating network "wordpress default" with the default driver
Pulling wordpress (wordpress:...)...
latest: Pulling from library/wordpress
2d429b9e73a6: Already exists
8e3574ead1d9: Pull complete
33ddd73cf168: Downloading [====>] 6.433MB/104.3MB
03e622ab6113: Download complete
3d465c9a467d: Downloading [=====>] 4.639MB/20.12MB
c99b33b2d2df: Download complete
8944b2c2d493: Download complete
b95e19029c21: Downloading [=====>] 8.735MB/12.27MB
3ecc49d93144: Waiting
413b3a10b41e: Waiting
93e37cdea03d: Waiting
2c3cdaf28ff9: Waiting
bebb38845b62: Waiting
4f4fb700ef54: Waiting
1062c4bf27a2: Waiting
0f3c44dd6c5b: Waiting
205f781f096b: Waiting
147b34766441: Waiting
cc509c872df2: Waiting
a6ef3423d3cc: Waiting
8bd2c82cab52: Waiting
cdd30a8da961: Waiting

```

4) Escribimos el comando “docker-compose up -d” y la creación comenzará.

```

osboxes@osboxes:~/wordpress$ docker-compose ps

```

Name	Command	State	Ports
servidor_mysql	docker-entrypoint.sh mariadb	Up	3306/tcp
servidor_wp	docker-entrypoint.sh apach ...	Up	0.0.0.0:80->80/tcp, :::80->80/tcp

```

osboxes@osboxes:~/wordpress$ ifconfig
br-895898e11e87: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.1 netmask 255.255.0.0 broadcast 172.18.255.255
    inet6 fe80::42:eeff:fe29:ce3a prefixlen 64 scopeid 0x20<link>
    ether 02:42:ee:29:ce:3a txqueuelen 0 (Ethernet)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 27 bytes 3898 (3.8 KB)
    TX errors 0 dropped 17 overruns 0 carrier 0 collisions 0

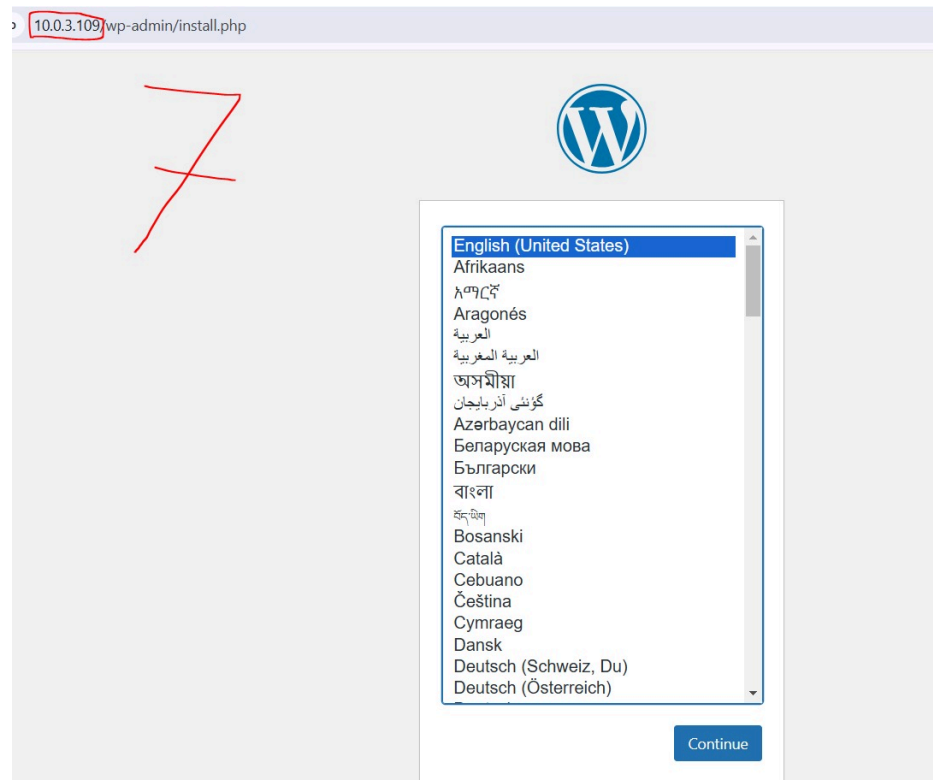
docker0: flags=4099<UP,BROADCAST,MULTICAST> mtu 1500
    inet 172.17.0.1 netmask 255.255.0.0 broadcast 172.17.255.255
    ether 02:42:af:46:2b:60 txqueuelen 0 (Ethernet)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 0 bytes 0 (0.0 B)
    TX errors 0 dropped 22 overruns 0 carrier 0 collisions 0

enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 10.0.3.109 netmask 255.255.0.0 broadcast 10.0.255.255
    inet6 fe80::60dc:a57e:7f12:6b59 prefixlen 64 scopeid 0x20<link>
    ether 08:00:27:89:18:65 txqueuelen 1000 (Ethernet)
    RX packets 290312 bytes 407896038 (407.8 MB)
    RX errors 0 dropped 172 overruns 0 frame 0
    TX packets 102883 bytes 8395635 (8.3 MB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

```

5) Con el comando docker-compose ps veremos los contenedores listados.

6) Si usamos ifconfig comprobaremos nuestra dirección IP como hemos hecho otras veces antes.



- 7) Al escribir nuestra IP en el navegador deberíamos poder acceder sin problema, tal como se ve en la imagen. Ni siquiera necesitamos escribir el puerto esta vez.

```
osboxes@osboxes:~/wordpress$ docker-compose stop
Stopping servidor_mysql ... done
Stopping servidor_wp ... done
osboxes@osboxes:~/wordpress$ docker-compose down -v
Removing servidor_mysql ... done
Removing servidor_wp ... done
Removing network wordpress_default
osboxes@osboxes:~/wordpress$
```

- 8) Con el comando **docker-compose stop** detendremos ambos contenedores a la vez.
- 9) Con el comando **docker-compose down -v** eliminaremos no solamente los contenedores, sino todo el entorno. Con esto. el ejercicio está terminado.