
«Dockerizar» una aplicación LAMP

Implantación de Aplicaciones Web

Curso 2025/2026

Índice general

1	«Dockerizar» una aplicación LAMP	1
1.1	Tareas a realizar	1
1.2	Requisitos del archivo Dockerfile	1
1.3	Requisitos del archivo docker-compose.yml	2
1.3.1	Servicio de MySQL	2
1.3.2	Networks	2
1.3.3	Docker restart policies	3
1.3.4	Variables de entorno	3
1.3.5	Orden en el que se inician los servicios	3
2	Referencias	4
3	Licencia	5

1 «Dockerizar» una aplicación LAMP

En esta práctica tendremos que crear un archivo [Dockerfile](#) para crear una imagen [Docker](#) que contenga una **aplicación web LAMP**. Posteriormente deberá realizar la implantación del sitio web en [Amazon Web Services \(AWS\)](#) haciendo uso de contenedores [Docker](#) y de la herramienta [Docker Compose](#).

1.1. Tareas a realizar

A continuación se describen **muy brevemente** algunas de las tareas que tendrá que realizar.

1. Crear una máquina virtual [Amazon EC2](#).
2. Instalar y configurar [Docker](#) y [Docker compose](#) en la máquina virtual.
3. Crear un archivo [Dockerfile](#) para crear una imagen que contenga el servicio de **Apache** con la siguiente aplicación web:
 - <https://github.com/josejuansanchez/iaw-practica-lamp>
4. Crear un archivo **docker-compose.yml** para poder desplegar los servicios de **Apache**, **MySQL** y **phpMyAdmin**. Deberá utilizar las imágenes oficiales de [Docker Hub](#).
5. Busque cuál es la dirección IP pública de su instancia y compruebe que puede acceder a los servicios de **Apache** y **phpMyAdmin** desde una navegador web.

1.2. Requisitos del archivo Dockerfile

Tendrá que crear un archivo [Dockerfile](#) con los siguientes requisitos:

- Como imagen base deberá utilizar la versión de [ubuntu](#) que está etiquetada como [focal](#).
- Instala el software necesario para poder ejecutar el servicio de **Apache** y servir una **aplicación web escrita en PHP** que hace uso de una base de datos **MySQL**.
- Para realizar la instalación de los paquetes será necesario configurar la variable de entorno [DEBIAN_FRONTEND](#) con el valor [noninteractive](#).

Ejemplo:

```
1 ENV DEBIAN_FRONTEND=noninteractive
```

- Deberá copiar el código de la aplicación web en el directorio `/var/www/html`, que es el directorio que utiliza Apache para servir el contenido. Tenga en cuenta que la aplicación web está en el siguiente repositorio de GitHub:
 - <https://github.com/josejuansanchez/iaw-practica-lamp>
- Tenga en cuenta que tendrá que modificar el archivo de configuración `config.php` y **sustituir el nombre de `localhost` por el nombre del servicio donde se estará ejecutando el servicio de MySQL.**
- Recuerde que **Apache** está configurado por defecto para darle prioridad a los archivos `index.html`, sin embargo, el archivo principal de la web se llama `index.php`.
- El puerto que usará la imagen para ejecutar el servicio de Apache será el puerto 80.

1.3. Requisitos del archivo `docker-compose.yml`

1.3.1. Servicio de MySQL

Deberá utilizar la última versión de la imagen de **MySQL** disponible en [Docker Hub](#).

Tenga en cuenta que la imagen oficial de **MySQL** está preparada para permitirnos importar un script SQL con la base de datos inicial de nuestra aplicación. Para importar la base de datos de la aplicación web puede crear un volumen de tipo *bind mount* entre el directorio de su máquina local donde está el script SQL y el directorio `/docker-entrypoint-initdb.d` de la imagen oficial de MySQL. Esto hará que la primera vez que se instancie la base de datos leerá todos los archivos con extensión `.sql` que estén en este directorio y se ejecutarán.

A continuación se muestra un ejemplo de cómo tendría que hacerlo en el archivo `docker-compose.yml`:

```
1 ...
2   volumes:
3     - mysql_data:/var/lib/mysql
4     - ./sql:/docker-entrypoint-initdb.d
5 ...
```

1.3.2. Networks

Los servicios definidos en el archivo **`docker-compose.yml`** deberán usar dos redes:

- *frontend-network*
- *backend-network*

En la red *frontend-network* estarán los servicios:

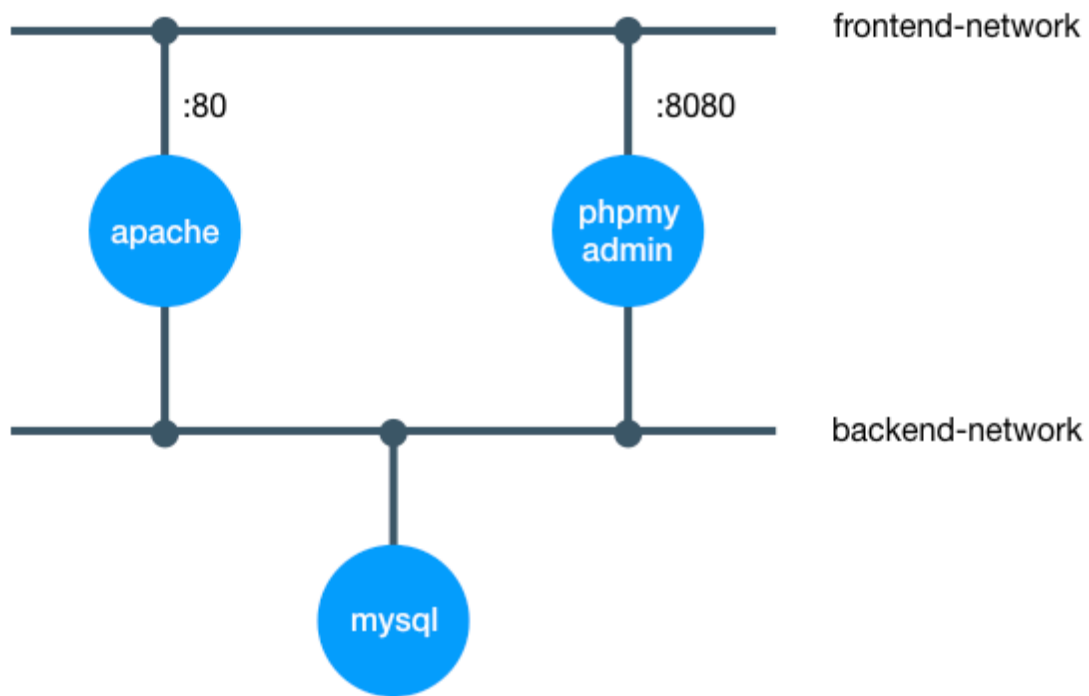
- **Apache**
- **phpMyAdmin**

Y en la red *backend-network* sólo estará el servicio:

■ MySQL

Sólo los servicios que están en la red *frontend-network* expondrán sus puertos en el host. Por lo tanto, el servicio de **MySQL** no deberá estar accesible desde el host.

A continuación se muestra un diagrama con las redes y los servicios que tiene que crear:



1.3.3. Docker restart policies

Deberá utilizar alguna política de reinicio para que los contenedores se reinicien cada vez que se detengan de forma inesperada.

1.3.4. Variables de entorno

Deberá hacer uso de un archivo **.env** para almacenar todas las variables de entorno que necesite en el archivo **docker-compose.yml**.

1.3.5. Orden en el que se inician los servicios

Deberá indicar el orden en el que se deben iniciar los servicios con la opción `depends_on`. Se recomienda la lectura del artículo [Control startup and shutdown order in Compose](#)

2 Referencias

- [Quickstart: Compose and WordPress.](#)
- [Curso de introducción a Docker.](#)
- [Docker](#)
- [Docker Compose](#)
- [Control startup and shutdown order in Compose](#)

3 Licencia

Esta página forma parte del curso Implantación de Aplicaciones Web de José Juan Sánchez Hernández y su contenido se distribuye bajo una licencia Creative Commons Reconocimiento-NoComercial-CompartirIgual 4.0 Internacional.