

# MANUAL TÉCNICO

## Implementación Docker

Elaboró: Oscar Jaime González

Ultima revisión 18 de septiembre de 2023

## Contenido

OBJETIVO .....	3
INTRODUCCIÓN .....	4
REQUERIMIENTOS DE FUNCIONAMIENTO .....	5
Contenedores: .....	5
Generales: .....	5
Particulares: .....	5
SOLUCIONES TECNOLÓGICAS .....	6
Docker .....	6
Instalación Windows .....	6
PHP-Apache .....	8
MySQL .....	10
CREACIÓN DE CONTENEDORES .....	11
BIBLIOGRAFÍA .....	15

## OBJETIVO

El siguiente manual técnico está elaborado con la finalidad de detallar las tecnologías, configuraciones y consideraciones especiales elegidas para la implementación de los requerimientos solicitados.

En el presente manual se detallan algunos conceptos y configuraciones necesarias para la ejecución de un Servidor Web y una base de datos, adicionalmente se establecen las mejores soluciones para obtener de manera simplificada el resultado solicitado y la integración de estos servicios en contenedores Docker.

Proporcionar una guía detallada de los pasos a seguir para la implementación de 2 contenedores Docker, los cuales contendrán un servidor web (PHP-Apache) y una base de datos relacional (MySQL). Estos contenedores estarán conectados por una Red Docker. Adicionalmente se detallan configuraciones necesarias para cumplir con requerimientos adicionales.

Adicionalmente se documenta un código PHP para observar el correcto funcionamiento de esta conexión (Servidor Web y Base de Datos).

## INTRODUCCIÓN

Para la implementación de los contenedores Docker se eligieron las siguientes soluciones tecnológicas:

- Servidor Web: Apache
- Lenguaje de desarrollo web: PHP, HTML
- Base de Datos: MySQL

Además de Docker para realizar la implementación y un navegador Web para la visualización.

A continuación, se detalla la instalación, configuración y pruebas de conexión para la implementación conjunta de los contenedores.

## REQUERIMIENTOS DE FUNCIONAMIENTO

### Contenedores:

#### Generales:

Red de comunicación entre contenedores.

Verificación de la salud de los contenedores.

Inicio automático tras encender el equipo de cómputo.

Almacenar los archivos de configuración en volúmenes y con acceso desde el host principal.

#### Particulares:

##### Contenedor 1: Servidor Web (PHP-apache)

- Puerto 1234

##### Contenedor 2: Base de Datos

- Puerto 4321

## SOLUCIONES TECNOLÓGICAS

### Docker

#### Instalación Windows

Para la instalación de la herramienta en Windows se utilizará una interfaz gráfica para la instalación.

Accede en un navegador web a la siguiente dirección: [Docker](https://docker.com)

Selecciona tu sistema operativo (Windows) y descarga el instalador.

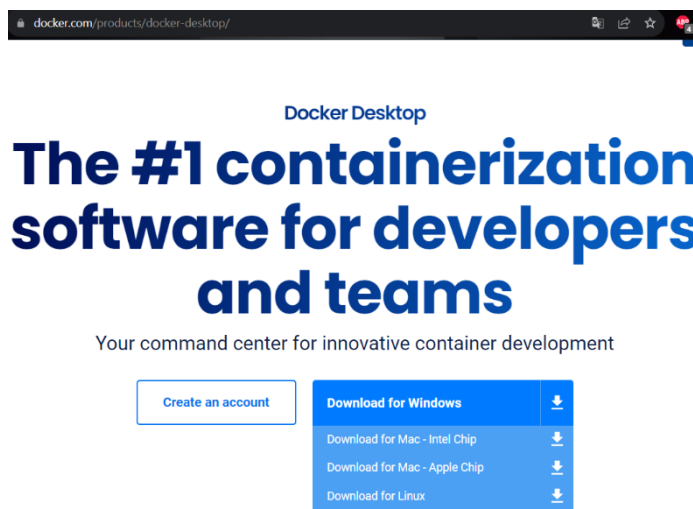


Imagen 1. Página web de Docker Desktop

Ejecutamos el instalador y habilitamos la casilla de **“Enable WSL 2 Windows Features”**.



Imagen 2. Instalador Docker Desktop

Docker nos pedirá cerrar y reiniciar el equipo.

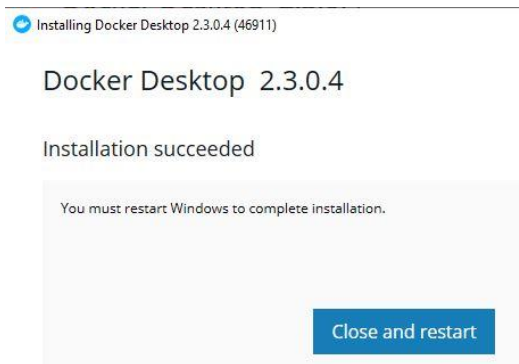


Imagen 3. Instalador Docker Desktop

Después de reiniciar el equipo abrimos la interfaz de Docker Desktop para verificar si se instaló correctamente.

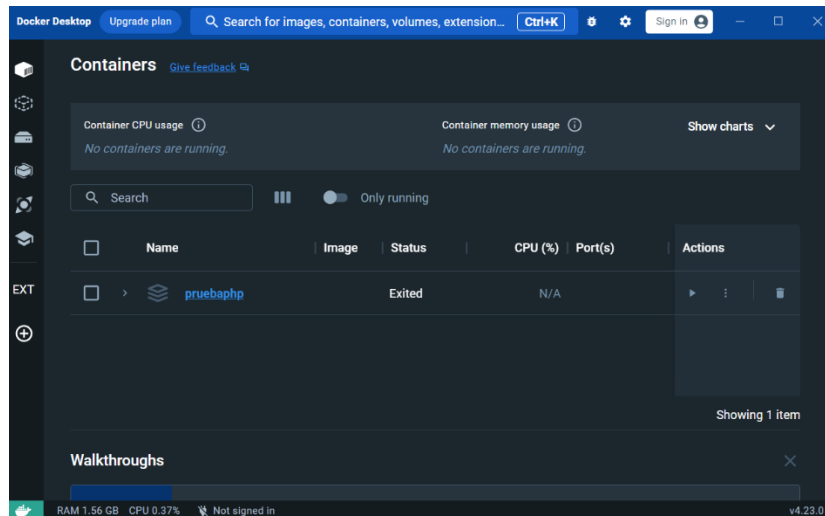


Imagen 4. Interfaz Docker Desktop

**Nota:** En caso de algún error en la instalación o en la ejecución de la interfaz gráfica puedes consultar la documentación de Docker aquí ([Documentación de Docker](#)).

## PHP-Apache

Crearemos un directorio llamado **PruebaPHP**. Dentro crearemos un archivo con nombre **“docker-compose.yml”**, este archivo tiene la función de almacenar la información de los contenedores y las configuraciones necesarias para su correcto funcionamiento.

Adicionalmente, crea el directorio **“www”** y **“db”**.

Dentro del directorio **“www”** creamos un archivo **“index.php”**. La estructura final nos quedara de esta manera:

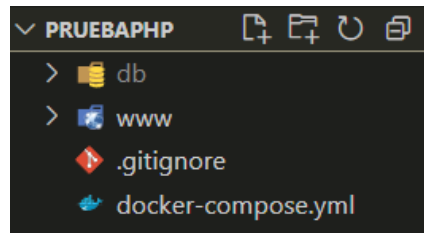


Imagen 5. Estructura del Proyecto, directorio PruebaPHP

Utilizando un editor de texto modificamos el archivo **“docker-compose.yml”**.

```
docker-compose.yml
1  version: '3.9'
2
3  services:
4    serv-apache:
5      image: php:7.4-apache
6      container_name: serv-apache
7      ports:
8        - "1234:80"
9      restart: always
10     volumes:
11       - ./www:/var/www/html
12     links:
13       - db-mysql
14     healthcheck:
15       test: curl --fail http://localhost || exit 1
16       interval: 10s
17       retries: 5
18       start_period: 5s
19       timeout: 10s
20
```

Imagen 6. Archivo “docker-compose.yml”.

A continuación, se detallan las configuraciones del archivo:



**Image:** es la imagen en la cual se basa para crear el contenedor, en este caso es “php:7.4-apache”, esta versión de PHP incluye una implementación de apache para generar el servidor web.

**container\_name:** es el nombre del contenedor para este caso el nombre es “serv-apache”.

**ports:** aquí definimos el puerto del anfitrión y el puerto del contenedor, para este caso el puerto por default del servidor apache es el 80 (en el contenedor) y por requerimiento asignamos el puerto 1234 en el anfitrión (PC).

**restart:** como parte de los requerimientos activamos el reinicio del contenedor. Para la ejecución automática del contenedor.

**volumes:** definimos un volumen donde se coloca el Código PHP para visualizar una página web. La ruta es nuestro directorio “pruebaPHP/www” y hace referencia al directorio default donde se colocan los archivos HTML en los servidores apache en este caso es “/var/www/html”

**links:** en este valor se define una Red Docker entre los contenedores y se asocia a este contenedor y el que se coloca como valor, en nuestro caso es el contenedor “db-mysql” el cual es nuestra base de datos.

**healthcheck:** en esta configuración se realiza la verificación de salud del contenedor con parámetros recomendados en la documentación de Docker. Para más información consultar:

[Compose V3-Healthcheck](#)

## MySQL

Para continuar con la configuración del contenedor de la Base de Datos se adicionará al archivo “docker-compose.yml” la información requerida para su creación, cuidando la indentación de los elementos que se anexen.

```
21 db-mysql:
22   image: mysql:8
23   container_name: db-mysql
24   environment:
25     MYSQL_DATABASE: prueba-tec
26     MYSQL_ROOT_PASSWORD: pswdummy1
27   ports:
28     - "4321:3306"
29   restart: always
30   volumes:
31     - ./db:/var/lib/mysql
```

Imagen 7. Archivo “docker-compose.yml”

A continuación, se detallan las configuraciones del archivo:

**Image:** es la imagen en la cual se basa para crear el contenedor, en este caso es “mysql:8.”

**container\_name:** es el nombre del contenedor para este caso el nombre es “db-mysql”.

**environment:** utilizaremos variables de entorno, las cuales nos servirán para realizar la conexión con la Base de Datos. Para esta implementación utilizaremos “MYSQL\_DATABASE” y “MYSQL\_ROOT\_PASSWORD”.

**ports:** aquí definimos el puerto del anfitrión y el puerto del contenedor, para este caso el puerto por default de MySQL es el 3306 (en el contenedor) y por requerimiento asignamos el puerto 4321 en el anfitrión (PC).

**restart:** como parte de los requerimientos activamos el reinicio del contenedor. Para la ejecución automática del contenedor.

**volumes:** definimos un volumen donde se coloca los datos de la BD. La ruta es nuestro directorio “pruebaPHP/db “ y hace referencia al directorio default donde se colocan los datos correspondientes a la Base de Datos en MySQL en este caso es “/var/lib/mysql”

## CREACIÓN DE CONTENEDORES

Para la creación de los contenedores el archivo “Docker-compose.yml” debe de contener el siguiente contenido.

```
docker-compose.yml
1  version: '3.9'
2
3  services:
4    serv-apache:
5      image: php:7.4-apache
6      container_name: serv-apache
7      ports:
8        - "1234:80"
9      restart: always
10     volumes:
11       - ./www:/var/www/html
12     links:
13       - db-mysql
14     healthcheck:
15       test: curl --fail http://localhost || exit 1
16       interval: 10s
17       retries: 5
18       start_period: 5s
19       timeout: 10s
20
21     db-mysql:
22       image: mysql:8
23       container_name: db-mysql
24       environment:
25         MYSQL_DATABASE: prueba-tec
26         MYSQL_ROOT_PASSWORD: pswdummy1
27       ports:
28         - "4321:3306"
29       restart: always
30       volumes:
31         - ./db:/var/lib/mysql
```

Imagen 8. “docker-compose.yml” versión final

Código del archivo “docker-compose.yml”:

```
version: '3.9'

services:
  serv-apache:
    image: php:7.4-apache
    container_name: serv-apache
    ports:
      - "1234:80"
    restart: always
    volumes:
```

```

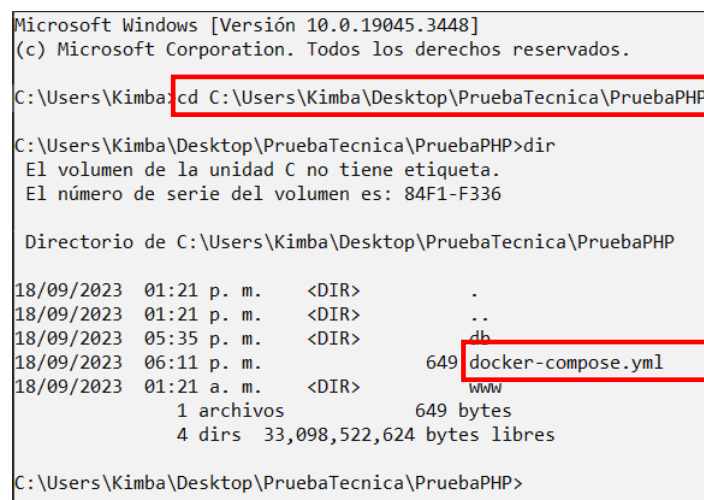
- ./www:/var/www/html
links:
- db-mysql
healthcheck:
  test: curl --fail http://localhost || exit 1
  interval: 10s
  retries: 5
  start_period: 5s
  timeout: 10s

db-mysql:
  image: mysql:8
  container_name: db-mysql
  environment:
    MYSQL_DATABASE: prueba-tec
    MYSQL_ROOT_PASSWORD: pswdummy1
  ports:
    - "4321:3306"
  restart: always
  volumes:
    - ./db:/var/lib/mysql

```

Nota: Al copiar el código se debe que cuidar la indentación.

A continuación, abrimos una ventana terminal y utilizaremos el comando `cd` y `dir` para movernos al directorio `pruebaPHP` y verificar que el archivo “`Docker-compose.yml`” está en el directorio.



```

Microsoft Windows [Versión 10.0.19045.3448]
(c) Microsoft Corporation. Todos los derechos reservados.

C:\Users\Kimba>cd C:\Users\Kimba\Desktop\PruebaTecnica\PruebaPHP

C:\Users\Kimba\Desktop\PruebaTecnica\PruebaPHP>dir
El volumen de la unidad C no tiene etiqueta.
El número de serie del volumen es: 84F1-F336

Directorio de C:\Users\Kimba\Desktop\PruebaTecnica\PruebaPHP
18/09/2023  01:21 p. m.    <DIR>          .
18/09/2023  01:21 p. m.    <DIR>          ..
18/09/2023  05:35 p. m.    <DIR>          db
18/09/2023  06:11 p. m.             649  docker-compose.yml
18/09/2023  01:21 a. m.    <DIR>          www
                    1 archivos             649 bytes
                    4 dirs 33,098,522,624 bytes libres

C:\Users\Kimba\Desktop\PruebaTecnica\PruebaPHP>

```

Imagen 9. CMD, ejecución de comandos `cd` y `dir`

Ejecutamos un comando de Docker, “docker compose up -d”. Docker comenzara a descargar las imágenes y a crear los contenedores y al finalizar los iniciara.

```
C:\Users\Kimba\Desktop\PruebaTecnica\PruebaPHP>docker compose up -d
[+] Running 25/18
  db-mysql 10 layers [██████████] 0B/0B Pulled 46.5s
  serv-apache 13 layers [██████████] 0B/0B Pulled 3.1s
```

Imagen 11. Descarga de imágenes y creación de contenedores.

```
[+] Running 3/3
  Network pruebapHP_default Created 0.1s
  Container db-mysql Started 5.3s
  Container serv-apache Started 0.1s
```

Imagen 10. Inicio de los contenedores

Podemos observar en Docker Desktop las imágenes que se utilizaron como base y los contenedores creados, adicionalmente podemos observar que están corriendo los logs generados.

<input type="checkbox"/>	Name	Tag	Status	Created	Size	Actions
<input type="checkbox"/>	mysql 8da80fe49fcf	8	<a href="#">In use</a>	3 days ago	577.37 MB	▶ ⋮ 🗑
<input type="checkbox"/>	php 20a3732f422b	7.4-apache	<a href="#">In use</a>	10 months ago	452.58 MB	▶ ⋮ 🗑

Imagen 12. imágenes descargadas y base de los contenedores.

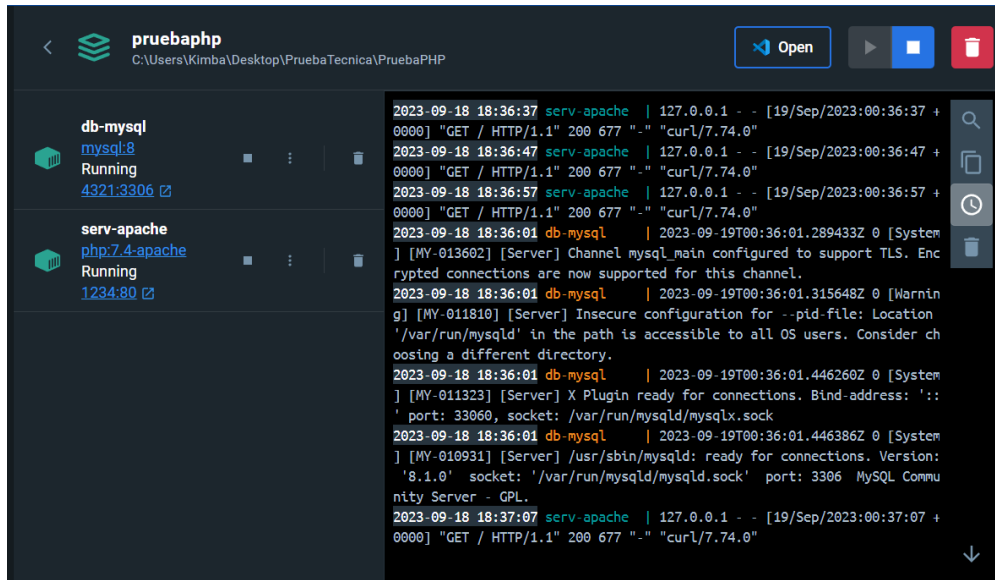


Imagen 13. Contenedores corriendo y logs.

## BIBLIOGRAFÍA

Se reviso documentación y sitios relacionados con la implementación

«*Install Docker desktop on Windows*». (2023, 14 septiembre). Docker Documentation.  
<https://docs.docker.com/desktop/install/windows-install/>

Docker. (s. f.). [https://hub.docker.com/\\_/php](https://hub.docker.com/_/php)

Docker. (s. f.-b). [https://hub.docker.com/\\_/mysql](https://hub.docker.com/_/mysql)

PHP: Documentation. (s. f.). <https://www.php.net/docs.php>