2025

Guía super mega básica para usar Laravel en un contenedor Docker

DOCKER Y LARAVEL

Miguel Castro

Contenido

Disclaimer

Esta guía proporciona instrucciones para configurar y ejecutar un entorno de desarrollo utilizando Docker con Laravel. Aunque he hecho todo lo posible para asegurar que los pasos sean precisos y fáciles de seguir, ten en cuenta lo siguiente:

Dependencias y Versiones: Las versiones de Docker, Docker Compose, PHP, Laravel y otros componentes pueden cambiar con el tiempo. Asegúrate de verificar que las versiones mencionadas en esta guía son compatibles con tu entorno de desarrollo. Los pasos proporcionados son específicos a la fecha de publicación y pueden no funcionar si las versiones de las herramientas cambian o si surgen actualizaciones.

Uso en Producción: Esta guía está destinada exclusivamente para entornos de desarrollo. No se recomienda usar la configuración propuesta en un entorno de **producción** sin realizar ajustes adicionales para optimizar la seguridad, el rendimiento y la escalabilidad.

Modificaciones: Los procedimientos descritos en esta guía pueden requerir modificaciones adicionales para adaptarse a configuraciones o necesidades específicas del proyecto. Se recomienda realizar pruebas exhaustivas antes de implementar cualquier solución en un entorno de producción.

Errores y Problemas Técnicos: Aunque los pasos se han verificado cuidadosamente, pueden surgir errores o problemas técnicos específicos en tu entorno. En caso de dificultades, se recomienda consultar la documentación oficial de Docker, Laravel y las imágenes de PHP para obtener más detalles o buscar asistencia en la comunidad.

Responsabilidad: El uso de esta guía es bajo tu propio riesgo. Ni yo ni el sitio web que proporciona esta guía seremos responsables por cualquier pérdida de datos, problemas técnicos o daños derivados de la implementación de las instrucciones proporcionadas.

¿Qué es y porque usar Docker y Laravel?

Docker es una herramienta que nos permite crear contenedores y con ello hacer portable y ejecutable en cualquier ambiente que lo requiera, dentro de estos contenedores se pueden agregar los servicios que se requieran por ejemplo PHP, Docker, Ngniz, Apache, etc.

Con Docker podrás tener proyectos de Laravel de cualquier versión sin necesidad de instalar todo el ambiente desde cero.

Instalación de Docker y Docker Compose

Como prerrequisitos es necesario contar con la instalación de Docker y Docker Compose sin importar el SO.

En caso de ser la primera vez que usaras Docker necesitaras instalarlo, por lo que es necesario que lo hagas descargándolo para tu SO desde la página oficial de Docker https://www.docker.com/get-started/

Una vez instalado desde consola, terminal o línea de comandos deberás de ejecutar:

```
docker --version
docker-compose --version
```

Al ejecutar dichos comandos deberá de mostrarte la versión con la que cuentas instalado.

Creación de contenedor

1. Crea la carpeta que contendrá tu contenedor y el proyecto que crearas, no importa si lo realizas desde línea de comandos o desde el ambiente gráfico, pero es importante que identifiques la ruta de acceso por la línea de comandos (opcional).
2. Dentro de la carpeta crea el archivo *Docker-compose.yml* dentro de este archivo se almacenarán las configuraciones de servicios y aplicaciones que

tendrá nuestro contenedor. Para la fecha de creación de esta guía se creó un archivo como el siguiente:

```
version: "3.8"

services:
  app:
    build:
      context: ./docker/php
    container_name: laravel_app
    restart: unless-stopped
    working_dir: /var/www
    volumes:
      - ./app:/var/www
    depends_on:
      - db
    ports:
      - "8000:80"

  db:
    image: mysql:8
    container_name: laravel_mysql
    restart: unless-stopped
    environment:
      MYSQL_ROOT_PASSWORD: root
      MYSQL_DATABASE: laravel
      MYSQL_USER: laraveluser
      MYSQL_PASSWORD: secret
    ports:
      - "3306:3306"
    volumes:
      - dbdata:/var/lib/mysql

volumes:
  dbdata:
```

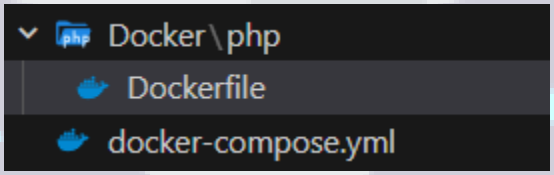
Explicación:

- `app`: Contenedor donde corre PHP con Apache.
- `db`: Contenedor con MySQL para la base de datos.
- `ports: "8000:80"` → Laravel estará disponible en `http://localhost:8000`.

- db: Base de datos MySQL para guardar información.

3. Se requiere configurar Apache y PHP para lo que crearemos una carpeta y un archivo dentro de nuestro proyecto que deberá de ver como lo siguiente:

Proyecto > Docker > php > Dockerfile



El archivo *Dockerfile* deberá de contener lo siguiente:

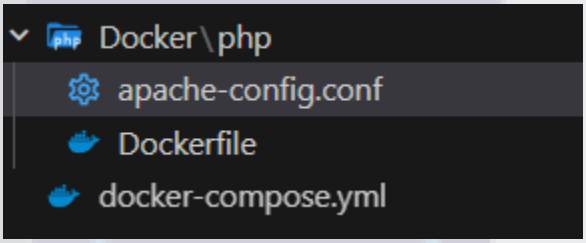
```
FROM php:8.2-apache

# Instalar dependencias necesarias
RUN apt-get update && apt-get install -y \
   libpng-dev \
   libjpeg-dev \
   libfreetype6-dev \
   zip unzip curl \
   libonig-dev \
   libzip-dev \
   && docker-php-ext-configure gd --with-freetype --with-jpeg \
   && docker-php-ext-install gd pdo_mysql mbstring zip

# Habilitar mod_rewrite para Apache
RUN a2enmod rewrite

# Copiar el archivo de configuración de Apache
COPY apache-config.conf /etc/apache2/sites-available/000-default.conf

# Establecer directorio de trabajo
WORKDIR /var/www

CMD ["apache2-foreground"]
```

Explicación:

- Usamos la imagen oficial php:8.2-apache, para más detalles puedes consultar aquí https://hub.docker.com/_/php al igual que lo n3ecesario para la versión de Laravel aquí https://laravel.com/docs/12.x/releases
- Instalamos las extensiones necesarias para Laravel.
- Activamos mod_rewrite en Apache.
- Copiamos una configuración personalizada para Apache.

4. Se requiere ahora configurar Apache para lo que se repite un proceso similar al anterior agregando el archivo *apache-config.conf* como se muestra a continuación:

Proyecto > Docker > php > apache-config.conf



Este archivo deberá de contener lo siguiente:

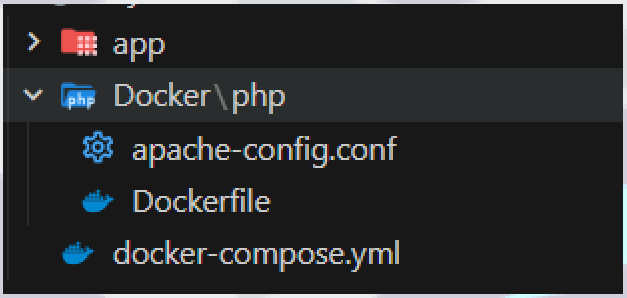
```
<VirtualHost *:80>
    DocumentRoot /var/www/public

    <Directory /var/www>
        AllowOverride All
        Require all granted
    </Directory>

    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>
```

Explicación:

- Apache servirá Laravel desde la carpeta public/.
- AllowOverride All permite que Laravel use .htaccess.

5. Se procede a crear la carpeta contenedora del proyecto que deberá de estar como lo siguiente:

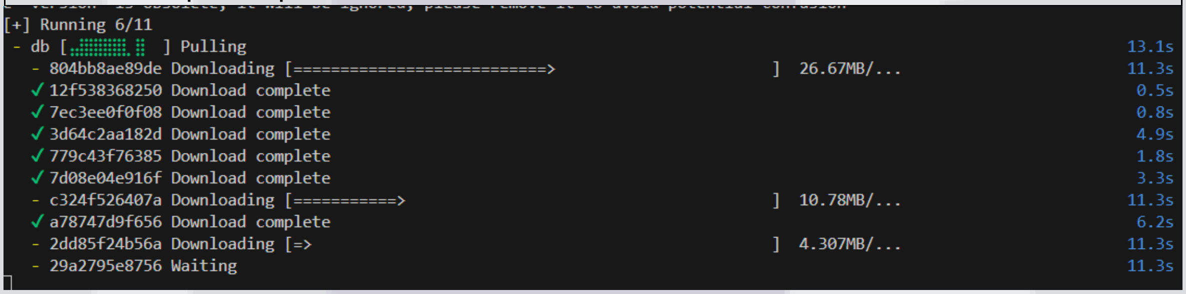
Proyecto > app



Construir y levantar contenedores

Para poder realizar esto se requiere que por la línea de comandos se ejecuten las siguientes instrucciones:

```
docker-compose up -d --build
```



```
[+] Running 6/11
 - db [⣿⣿⣿⣿⣿⣿⣿⣿⣀ ⣿ ] Pulling                                                    13.1s
   - 804bb8ae89de Downloading [=========================>          ] 26.67MB/...  11.3s
   ✓ 12f538368250 Download complete                                                0.5s
   ✓ 7ec3ee0f0f08 Download complete                                                0.8s
   ✓ 3d64c2aa182d Download complete                                                4.9s
   ✓ 779c43f76385 Download complete                                                1.8s
   ✓ 7d08e04e916f Download complete                                                3.3s
   - c324f526407a Downloading [===========>                        ] 10.78MB/...  11.3s
   ✓ a78747d9f656 Download complete                                                6.2s
   - 2dd85f24b56a Downloading [=>                                  ] 4.307MB/...  11.3s
   - 29a2795e8756 Waiting                                                         11.3s
```

Explicación:

- docker-compose up → Levanta los contenedores.
- -d → Los ejecuta en segundo plano.
- --build → Construye la imagen de PHP y Apache.

Puedes verificar si todo esta correcto ejecutando el siguiente comando:

```
docker ps
```

Si ves laravel_app y laravel_mysql, ¡todo va bien!

NOTA: para que todo lo anterior funcione deberás de estar posicionad@ sobre la ruta raíz de tu carpeta de Docker.

Instalación de Laravel

Si aun no cuentas con algún proyecto de Laravel dentro de la carpeta app de tu proyecto deberás de agregar lo siguiente.

```
composer create-project --prefer-dist laravel/laravel .
```

Y después al terminar de realizar esa instalación ejecutar exit para salir de ese proceso.

NOTA: Si deseas agregar un proyecto existente entonces deberás de pegar tu proyecto dentro de la carpeta de app.

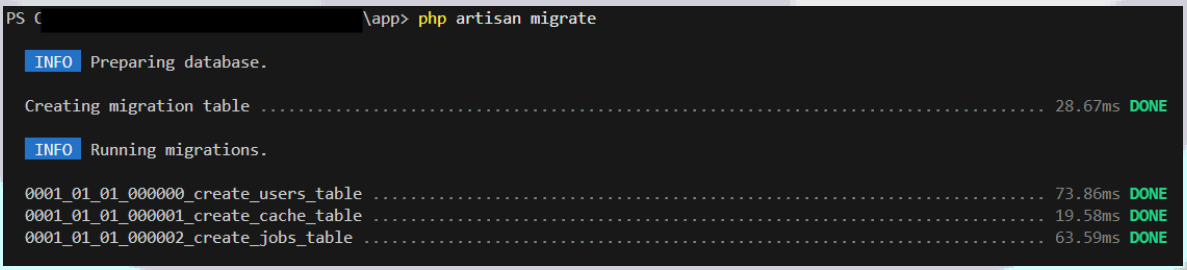
Configuración para bases de datos

Dentro de tu proyecto busca tu archivo *.env* dentro busca las los datos de base de datos (DB_) y puedes agregar los tuyos por ejemplo:

```
DB_CONNECTION=mysql
DB_HOST=db
DB_PORT=3306
DB_DATABASE=laravel
DB_USERNAME=laraveluser
DB_PASSWORD=secret
```

Y podrás correr las migraciones allí usando el comando:

```
php artisan migrate
```



Bonus

Comandos Útiles

docker-compose down

- **Detiene los contenedores en ejecución**
- **Elimina los contenedores**
- **Elimina la red creada por Docker Compose**
- **Elimina los volúmenes (si se usa --volumes)**

docker-compose restart

- **Detiene** los contenedores en ejecución.
- **Los vuelve a iniciar** con la misma configuración.
- **Los datos, volúmenes y configuraciones se mantienen intactos**.

docker-compose logs -f

- **Muestra los logs** de todos los contenedores que están corriendo bajo Docker Compose.
- La opción **-f** significa **"follow"** (seguir), lo que hace que los logs se actualicen en tiempo real. Esto permite ver los nuevos registros a medida que se generan sin tener que ejecutar el comando nuevamente.