

🔧 Documentación: Despliegue de Laravel con Docker

Este documento detalla paso a paso el proceso de despliegue de una aplicación Laravel utilizando Docker, así como la explicación del [Dockerfile](#) utilizado en el proceso.

🔧 1. Prerrequisitos

Antes de comenzar, asegúrate de tener instalado:

- [Docker](#)
- [Docker Compose \(opcional\)](#)
- Git (para clonar el repositorio si es necesario)

📁 2. Ubicación del [Dockerfile](#)

El [Dockerfile](#) se encuentra en la ruta **raíz** del proyecto Laravel. Esto significa que al construir la imagen, todo el código fuente del proyecto estará disponible en la imagen Docker.

📄 3. Contenido del [Dockerfile](#)

El [Dockerfile](#) se encarga de configurar un entorno basado en PHP con Apache para ejecutar Laravel dentro de un contenedor. A continuación, se explica su contenido:

```
# Usamos la imagen oficial de PHP con Apache
FROM php:8.3.0-apache-bullseye

# Instalamos dependencias necesarias
RUN apt-get update && apt-get install -y \
    git \
    curl \
    libpng-dev \
    libonig-dev \
    libxml2-dev \
    libzip-dev \
    zip \
    unzip \
    && apt-get clean && rm -rf /var/lib/apt/lists/*

# Instalamos extensiones PHP necesarias para Laravel
RUN docker-php-ext-install pdo_mysql zip mbstring exif pcntl bcmath gd

# Copiamos Composer desde una imagen oficial
COPY --from=composer:latest /usr/bin/composer /usr/bin/composer

# Configuración de PHP (modo desarrollo)
RUN cp $PHP_INI_DIR/php.ini-development $PHP_INI_DIR/conf.d/php.ini

# Configuramos Apache para servir Laravel desde la carpeta "public"
RUN sed -i 's#DocumentRoot /var/www/html#DocumentRoot /var/www/html/public#g'
```

```
/etc/apache2/sites-available/000-default.conf
RUN a2enmod rewrite

# Definimos la carpeta de trabajo
WORKDIR /var/www/html

# Copiamos el código fuente del proyecto al contenedor
COPY . .

# Instalamos las dependencias de Laravel
RUN composer install --no-dev --optimize-autoloader

# Ajustamos permisos para que Apache pueda acceder a los archivos
RUN chown -R www-data:www-data /var/www/html

# Exponemos el puerto 80 para servir la aplicación
EXPOSE 80

# Iniciamos Apache al ejecutar el contenedor
CMD ["apache2-foreground"]
```

🔍 Explicación del Dockerfile

1. Imagen Base

- FROM `php:8.3.0-apache-bullseye` → Utiliza PHP 8.3 con Apache como servidor web.

2. Instalación de Dependencias

- Se instalan las bibliotecas necesarias para Laravel (`git`, `curl`, `zip`, `mbstring`, etc.).

3. Extensiones PHP

- Se instalan las extensiones necesarias para Laravel (`pdo_mysql`, `mbstring`, `bcmath`, `gd`, etc.).

4. Composer

- Se copia Composer desde una imagen optimizada para no tener que instalarlo manualmente.

5. Configuración de PHP

- Se habilita el archivo `php.ini-development`.

6. Configuración de Apache

- Se modifica el `DocumentRoot` para que apunte a la carpeta `public` de Laravel.
- Se habilita el módulo `mod_rewrite` para que las rutas de Laravel funcionen correctamente.

7. Definición del Directorio de Trabajo

- WORKDIR `/var/www/html` → Define la raíz del proyecto dentro del contenedor.

8. Copia del Código y Dependencias

- `COPY . .` → Copia el código fuente dentro del contenedor.
- `RUN composer install --no-dev --optimize-autoloader` → Instala las dependencias sin las de desarrollo, optimizando el autoloader.

9. Permisos

- `chown -R www-data:www-data /var/www/html` → Ajusta los permisos para Apache.

10. Exposición del Puerto y Inicio de Apache

- `EXPOSE 80` → Expone el puerto 80 del contenedor.
- `CMD ["apache2-foreground"]` → Inicia Apache cuando el contenedor arranca.

🚀 4. Uso de `docker-compose.yml`

Se ha configurado un archivo `docker-compose.yml` que permite ejecutar Laravel de manera sencilla. El contenido del archivo es el siguiente:

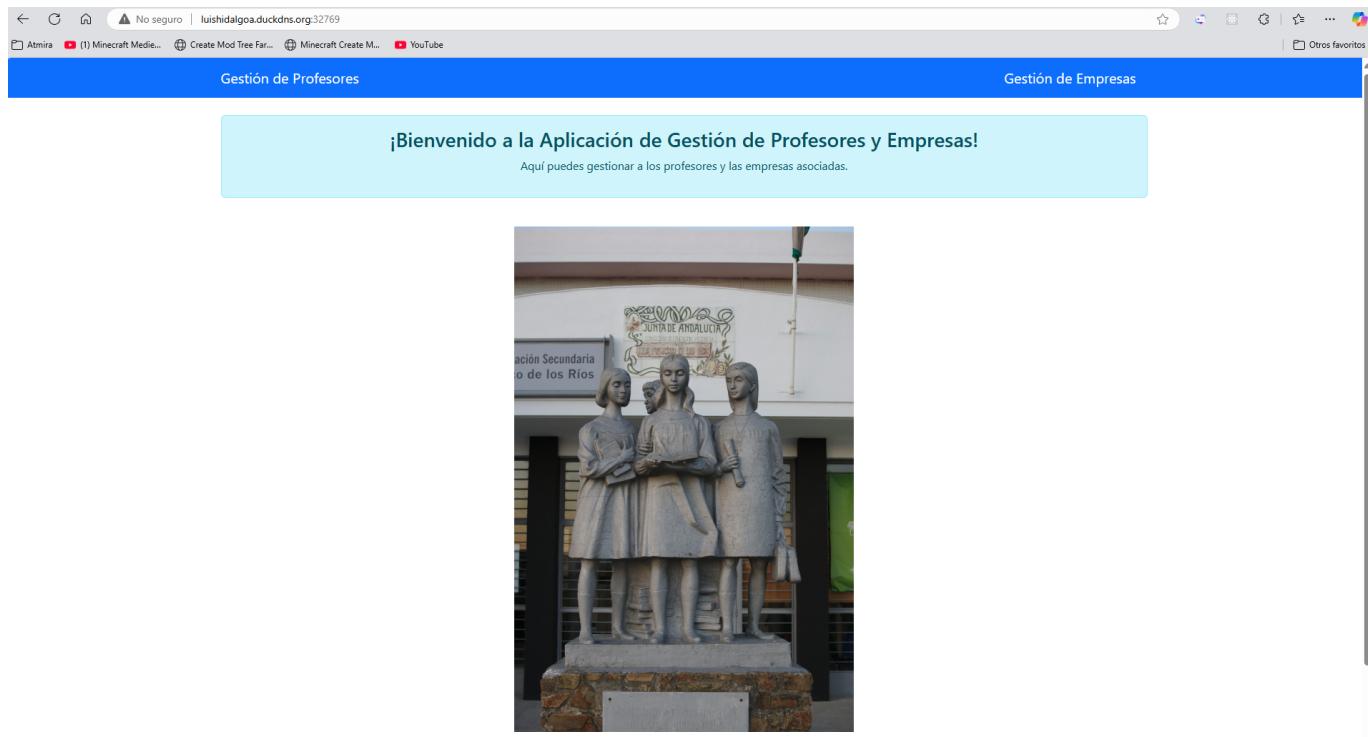
```
services:  
  app:  
    build:  
      context: ./  
    ports:  
      - "${HTTPD_PORT}:80"  
    working_dir: /var/www/html  
    networks:  
      - laravel_network  
  
networks:  
  laravel_network:  
    driver: bridge
```

🔗 4.1. Levantar los contenedores con `docker-compose`

Para iniciar todos los servicios definidos en `docker-compose.yml`, ejecuta:

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

ahora podras acceder a tu aplicación Laravel en <http://localhost:8000>. En mi caso lo he desplegado en AWS por lo que accedere a traves de <http://luishidalgoa.duckdns.org:32769>



❖ 4.2. Detener los contenedores

```
docker-compose down
```

Esto detendrá y eliminará los contenedores creados.

⌚ 5. Repositorio del Proyecto

Este despliegue ha sido realizado sobre el siguiente repositorio: [Laravel_Proyecto_FPDual](#)