

chat necesito un entorno de desarrollo rápido y fácilmente instalable para PHP, MySQL, estaba pensando en un Docker o un Nginx pero no quiero exponer mi Debian a tanto para no usarlo perpetuo prefiero algo rápido y configurable y fácil de instalación y que pueda replicar rápido. porque haré una página web conectada a MySQL y Composer

Perfecto, lo que describes **encaja exactamente con Docker**, y además no "ensucia" tu **Debian**, es reproducible y lo puedes levantar/bajar en segundos. Te explico **la mejor opción** y luego **alternativas** por si quieras algo aún más simple.

” Ask ChatGPT

### ✓ Opción recomendada: Docker + Docker Compose (PHP + MySQL + Nginx + Composer)

#### ¿Por qué es ideal para ti?

- ✓ No instala PHP/MySQL en tu sistema
- ✓ Replicable en cualquier máquina
- ✓ Configurable
- ✓ Rápido de levantar ( docker compose up )
- ✓ Compatible con Composer
- ✓ Fácil de versionar (Git)

### 🧱 Arquitectura típica

pgsql

 Copy code

```
Proyecto
├── docker/
│   ├── nginx/
│   │   └── default.conf
│   └── php/
│       └── Dockerfile
└── src/      ← tu código PHP
    └── docker-compose.yml
    └── composer.json
```

### 🔧 Instalación mínima (una sola vez)

bash

 Copy code

```
sudo apt install docker docker-compose-plugin
sudo usermod -aG docker $USER
reboot
```

### 📦 docker-compose.yml (ejemplo funcional)

yaml

 Copy code

```
version: "3.9"

services:
```

```

web:
  image: nginx:alpine
  ports:
    - "8080:80"
  volumes:
    - ./src:/var/www/html
    - ./docker/nginx/default.conf:/etc/nginx/conf.d/default.conf
  depends_on:
    - php

php:
  build: ./docker/php
  volumes:
    - ./src:/var/www/html
  working_dir: /var/www/html

mysql:
  image: mysql:8.0
  environment:
    MYSQL_DATABASE: app
    MYSQL_USER: app
    MYSQL_PASSWORD: secret
    MYSQL_ROOT_PASSWORD: root
  volumes:
    - mysql_data:/var/lib/mysql
  ports:
    - "3306:3306"

volumes:
  mysql_data:

```

” Ask ChatGPT

## 🐘 Dockerfile PHP (con Composer)

dockerfile

 Copy code

```

FROM php:8.3-fpm

RUN apt-get update && apt-get install -y \
    zip unzip git curl \
    && docker-php-ext-install pdo pdo_mysql

# Composer
COPY --from=composer:latest /usr/bin/composer /usr/bin/composer

```

## 🌐 Configuración Nginx

nginx

 Copy code

```

server {
  listen 80;
  root /var/www/html/public;
  index index.php;

  location / {
    try_files $uri /index.php?$query_string;
  }

  location ~ \.php$ {
    fastcgi_pass php:9000;
    fastcgi_index index.php;
  }
}

```

” Ask ChatGPT

” Ask ChatGPT

```
    include fastcgi_params;
    fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name;
}
}
```

## 🚀 Uso diario

bash

 Copy code

```
docker compose up -d      # levantar todo
docker compose down        # apagar
docker compose exec php bash
composer install
```

 Ask ChatGPT

Accedes desde:

arduino

 Copy code

<http://localhost:8080>

## 🟡 Alternativas (si quieres aún más simple)

### 1 Laravel Sail

Si usarás Laravel:

bash

 Copy code

```
curl -s https://laravel.build/app | bash
cd app
./vendor/bin/sail up
```

- ✓ Ultra rápido
- ✗ Más acoplado a Laravel

### 2 DDEV

Muy cómodo y potente:

bash

 Copy code

```
sudo apt install ddev
ddev config
ddev start
```

- ✓ Cero configuración
- ✗ Más pesado que Docker puro

### 3 ✗ XAMPP / LAMP

No recomendado en tu caso:

- Ensucia Debian
- Poco replicable
- Difícil de versionar

## 🏁 Conclusión clara

👉 Docker + docker-compose es EXACTAMENTE lo que buscas

👉 No compromete tu sistema

👉 Es profesional

👉 Escalable

👉 Replicable en segundos

Siquieres, en el próximo mensaje puedo:

- Ajustarlo a **Laravel / Symfony / PHP puro**
- Optimizarlo para **desarrollo vs producción**
- Añadir **phpMyAdmin**
- Hacerlo **100% copy-paste** listo

¿Con qué framework piensas trabajar?



” Ask ChatGPT

Aqui te paso mi sistema, nada de asumir que se cosas me das todo desde cero: n3o\_usr@endpoint-debianusr:~\$ fastfetch

```
_met$$$$$gg.      n3o_usr@endpoint-debianusr
,g$$$$$$$$$$$$$P.  -----
,g$$P""  ""Y$$. OS: Debian GNU/Linux 13 (trixie)
x86_64
,$$P'      $$$.     Host: Latitude 5490
,$$P ,ggs. $$b:    Kernel: Linux
6.12.57+deb13-amd64
d$$' , $P"'. .   $$$.     Uptime: 11
mins
$$P d$, , $$P Packages: 1955 (dpkg), 10 (flatpak), 7
(snap)
$: $$-. - ,d$$' Shell: bash 5.2.37
$$; Y$b._ ,d$P' Display (BOE072F): 1920x1080 @
60 Hz in 14" [Built-in] ."Y$$$$$P"
$$b " - .__ DE: GNOME
48.4
Y$$b                      WM: Mutter
(Wayland)
Y$$.                      WM Theme:
Adwaita
$$b.                     Theme: Adwaita
[GTK2/3/4]
Y$$b.                      Icons: Adwaita
[GTK2/3/4]
"Y$b._                   Font: Cantarell
(11pt) [GTK2/3/4]
"      Cursor: Adwaita
(24px)
Terminal: GNOME Terminal 3.56.2
Terminal Font: Monospace (11pt)
CPU: Intel(R) Core(TM) i7-8650U (8) @
4.20 GHz
GPU: Intel UHD Graphics 620 @ 1.15 GHz
[Integrated]
Memory: 3.43 GiB / 31.21 GiB (11%)
Swap: 0 B / 31.86 GiB (0%)
Disk (/): 66.73 GiB / 905.52 GiB (7%) - ext4
Local IP (wlp2s0): 10.246.100.217/24
Battery (DELL MT31P92): 68% (4 hours, 22
```

mins remaining) [Discharging]  
Locale: en\_US.UTF-8

Perfecto, gracias por el **fastfetch** 👍

Voy a hacerlo **100% desde cero**, sin asumir **ningún conocimiento previo**, y **adaptado a tu Debian 13 (trixie)**.

La idea es que al final tengas:

- ✓ PHP
- ✓ MySQL
- ✓ Composer
- ✓ Nginx
- ✓ Todo aislado (NO ensucia Debian)
- ✓ Replicable en minutos
- ✓ Fácil de borrar si no te gusta

” Ask ChatGPT

## 🧠 Idea general (explicado simple)

- 👉 Docker = cajitas aisladas donde corren programas
- 👉 Cada cajita tiene **una sola cosa** (PHP, MySQL, Nginx)
- 👉 Tu Debian solo ejecuta Docker, **nada más**
- 👉 Si borras el proyecto → **no queda rastro**

No es “usar Docker perpetuo”, es **usar contenedores desechables**.

## PASO 1 — Instalar Docker (una sola vez)

Abre una terminal y copia **tal cual**:

bash

 Copy code

```
sudo apt update  
sudo apt install -y docker.io docker-compose-plugin
```

Habilitamos Docker al arrancar:

bash

 Copy code

```
sudo systemctl enable docker  
sudo systemctl start docker
```

Permiso para usar Docker sin sudo :

bash

 Copy code

```
sudo usermod -aG docker $USER
```

⚠️ Reinicia el sistema ahora

bash

 Copy code

reboot

Después del reinicio, comprueba:

bash

 Copy code

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

Si eso funciona → Docker está listo ✓

## PASO 2 — Crear tu proyecto (estructura básica)

En tu HOME:

bash

 Copy code

```
mkdir webdev  
cd webdev
```

Creamos carpetas:

bash

 Copy code

```
mkdir -p docker/php docker/nginx src/public
```

” Ask ChatGPT

## PASO 3 — Archivo principal: docker-compose.yml

Crea el archivo:

bash

 Copy code

```
nano docker-compose.yml
```

Pega TODO esto:

yaml

 Copy code

```
services:  
  nginx:  
    image: nginx:alpine  
    ports:  
      - "8080:80"  
    volumes:  
      - ./src:/var/www/html  
      - ./docker/nginx/default.conf:/etc/nginx/conf.d/default.conf  
    depends_on:  
      - php  
  
  php:  
    build: ./docker/php  
    volumes:  
      - ./src:/var/www/html  
  
  mysql:  
    image: mysql:8.0  
    environment:  
      MYSQL_DATABASE: app  
      MYSQL_USER: app  
      MYSQL_PASSWORD: secret  
      MYSQL_ROOT_PASSWORD: root  
    volumes:  
      - mysql_data:/var/lib/mysql  
  
volumes:  
  mysql_data:
```

Guarda con:

objectivec

 Copy code

CTRL + O → ENTER → CTRL + X

## PASO 4 — PHP con MySQL + Composer

Crea el Dockerfile:

bash

 Copy code

nano docker/php/Dockerfile

Pega esto:

dockerfile

 Copy code

FROM php:8.3-fpm

RUN apt-get update && apt-get install -y git unzip zip curl \&& docker-php-ext-install pdo pdo\_mysql

 Ask ChatGPT

COPY --from=composer:latest /usr/bin/composer /usr/bin/composer

## PASO 5 — Configuración de Nginx

bash

 Copy code

nano docker/nginx/default.conf

Pega:

nginx

 Copy code

```
server {  
    listen 80;  
    root /var/www/html/public;  
    index index.php index.html;  
  
    location / {  
        try_files $uri /index.php?$query_string;  
    }  
  
    location ~ \.php$ {  
        fastcgi_pass php:9000;  
        fastcgi_index index.php;  
        include fastcgi_params;  
        fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name;  
    }  
}
```

## PASO 6 — Tu primer PHP

bash

 Copy code

nano src/public/index.php

Pega:

php

 Copy code

```
<?php  
phpinfo();
```

## PASO 7 — Arrancar TODO 🚀

Desde la carpeta webdev :

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

 Copy code

La primera vez tarda un poco (descarga imágenes).

” Ask ChatGPT

## PASO 8 — Probar en el navegador

Abre:

```
arduino  
http://localhost:8080
```

 Copy code

Si ves la página de `phpinfo()` → 🎉 TODO FUNCIONA

## PASO 9 — Usar Composer (MUY IMPORTANTE)

Entrar al contenedor PHP:

```
bash  
docker compose exec php bash
```

 Copy code

Dentro del contenedor:

```
bash  
composer --version
```

 Copy code

Ejemplo:

```
bash  
composer require vlucas/phpdotenv
```

 Copy code

Salir:

```
bash  
exit
```

 Copy code

## PASO 10 — Conectar PHP a MySQL

- ❖ **Host:** mysql
- ❖ **Base:** app
- ❖ **Usuario:** app
- ❖ **Password:** secret

Ejemplo PHP:

```
php
```

```
$pdo = new PDO(  
    "mysql:host=mysql;dbname=app",  
    "app",  
    "secret"  
) ;
```

## ✍ Apagar o borrar TODO

Apagar:

bash

 Copy code

```
docker compose down
```

 Ask ChatGPT

Borrar absolutamente todo:

bash

 Copy code

```
docker compose down -v  
rm -rf ~/webdev
```

Tu Debian queda **limpio como antes** ✓

## 🧠 Resumen claro

- ✓ No instalaste PHP
- ✓ No instalaste MySQL
- ✓ No tocaste Nginx del sistema
- ✓ Todo es reproducible
- ✓ Ideal para desarrollo real

Siquieres, en el siguiente mensaje puedo:

- Añadir **phpMyAdmin**
- Adaptarlo a **Laravel / Symfony**
- Explicarte **qué es cada cosa** con dibujo mental
- Optimizarlo para **poca RAM**
- Enseñarte a **clonar y levantar en otra máquina en 1 comando**

Dime **qué sigue** y lo hacemos paso a paso 🤗

