

# Enrique Crespo Ramirez

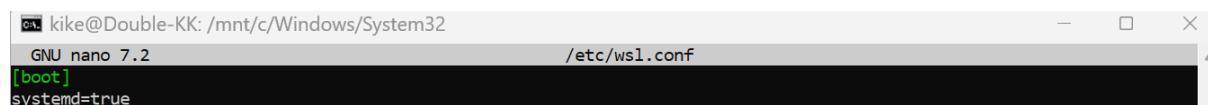
## Objetivos

- **Qué haremos:** levantar **dos servicios web en Docker**:
  - **Apache + PHP** (para ejecutar `info.php` y una página PHP propia).
  - **Nginx** (sirviendo HTML estático).
- **Cómo lo haremos:** con **Docker Compose** (un archivo `compose.yml` que arranca ambos servicios a la vez).

## Activa systemd en Ubuntu de WSL

**Por qué:** El servicio de Docker (`dockerd`) se maneja con **systemd**. En WSL hay que habilitarlo para poder usar `systemctl enable --now docker`, autoinicio, etc.

**`sudo nano /etc/wsl.conf`**



```
kike@Double-KK: /mnt/c/Windows/System32
GNU nano 7.2 /etc/wsl.conf
[boot]
systemd=true
```

## Añade la clave GPG y el repositorio oficial

Sin el repo de Docker, `apt` no encontrará las versiones oficiales.

**`sudo apt-get update`**

**`sudo apt-get install -y ca-certificates curl gnupg`**

## Carpeta para llaves

crea el directorio donde guardaremos las **llaves GPG** (keyrings) que usa `apt` para **verificar la firma** de los paquetes del repositorio. Aquí es donde guardaremos la GPG del Docker.

**`sudo install -m 0755 -d /etc/apt/keyrings`**

## Clave GPG de Docker

- **GPG** = *GNU Privacy Guard*. Es un sistema de **criptografía de clave pública** (tienes una clave privada para firmar y una clave pública para verificar).
- La **“GPG de Docker”** (o *clave pública de Docker*) es la clave que Docker publica para que tu sistema **verifique** que los paquetes del **repositorio oficial de Docker** (los `.deb` y el índice de `apt`) **vienen realmente de Docker y no han sido alterados**.

```
sudo curl -fsSL https://download.docker.com/linux/ubuntu/gpg -o  
/etc/apt/keyrings/docker.asc sudo chmod a+r  
/etc/apt/keyrings/docker.asc
```

repositorio (usa automáticamente tu codename:  
noble, jammy, etc.)

Este bloque añade el repositorio oficial de Docker a APT de forma segura y actualiza el índice.

```
echo \
```

```
"deb [arch=$(dpkg --print-architecture) signed-  
by=/etc/apt/keyrings/docker.asc] \
```

```
https://download.docker.com/linux/ubuntu
```

```
$(. /etc/os-release && echo ${UBUNTU_CODENAME:-  
$VERSION_CODENAME}) stable" \
```

```
| sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
```

## instalamos Docker Engine + CLI + containerd + Compose plugin

```
sudo apt-get install -y docker-ce docker-ce-cli containerd.io docker-buildx-plugin  
docker-compose-plugin
```

- docker-ce: motor de Docker
  - docker-ce-cli: el comando docker
  - containerd.io: runtime de contenedores
  - docker-buildx-plugin: builder avanzado
  - docker-compose-plugin: **docker compose** (v2) integrado
- Tras esto, Docker suele arrancar solo

```
sudo systemctl enable --now docker
```

```
sudo systemctl status docker --no-pager (comprobar estado del docker)
```

```
kike@Double-KK:/mnt/c/Windows/System32$ sudo systemctl status docker --no-pager
● docker.service - Docker Application Container Engine
   Loaded: loaded (/usr/lib/systemd/system/docker.service; enabled; preset: enabled)
   Active: active (running) since Wed 2025-10-08 18:55:41 CEST; 31s ago
     TriggeredBy: ● docker.socket
    Docs: https://docs.docker.com
   Main PID: 1803 (dockerd)
      Tasks: 13
  Memory: 22.3M (peak: 25.3M)
     CPU: 266ms
    CGroup: /system.slice/docker.service
            └─1803 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock

Oct 08 18:55:41 Double-KK dockerd[1803]: time="2025-10-08T18:55:41.404659638+02:00" level=info msg="CDI directo.../run/cdi
Oct 08 18:55:41 Double-KK dockerd[1803]: time="2025-10-08T18:55:41.415500462+02:00" level=info msg="Creating a ..out=1m0s
Oct 08 18:55:41 Double-KK dockerd[1803]: time="2025-10-08T18:55:41.483969795+02:00" level=info msg="Loading con... start."
Oct 08 18:55:41 Double-KK dockerd[1803]: time="2025-10-08T18:55:41.705959564+02:00" level=info msg="Loading con...: done."
Oct 08 18:55:41 Double-KK dockerd[1803]: time="2025-10-08T18:55:41.763230326+02:00" level=info msg="Docker daem..n=28.5.1
Oct 08 18:55:41 Double-KK dockerd[1803]: time="2025-10-08T18:55:41.763553031+02:00" level=info msg="Initializin...uildkit"
Oct 08 18:55:41 Double-KK dockerd[1803]: time="2025-10-08T18:55:41.791792472+02:00" level=info msg="Completed b...ization"
Oct 08 18:55:41 Double-KK dockerd[1803]: time="2025-10-08T18:55:41.794450499+02:00" level=info msg="Daemon has ..ization"
Oct 08 18:55:41 Double-KK dockerd[1803]: time="2025-10-08T18:55:41.794503775+02:00" level=info msg="API listen ...er.sock"
Oct 08 18:55:41 Double-KK systemd[1]: Started docker.service - Docker Application Container Engine.
Hint: Some lines were ellipsized, use -l to show in full.
kike@Double-KK:/mnt/c/Windows/System32$
```

## Creamos una estructura para guardar el proyecto

```
cd ~
```

```
mkdir -p proyectos/docker-web-lab/{apache-php/src,nginx/html,docs/capturas}
```

Entramos a la estructura

`cd ~/proyectos/docker-web-lab`

```
like@Double-KK:~/proyectos/docker-web-lab$ ls -R
:
apache-php docs nginx

/apache-php:
src

/apache-php/src:

/docs:
capturas

/docs/capturas:

/nginx:
html

/nginx/html:
like@Double-KK:~/proyectos/docker-web-lab$
```

## Archivos para Apache + PHP

Dentro de la ruta **docker-web-lab** que Creamos para meter la configuracion php

Ese bloque crea **apache-php/src/index.php**, una **página PHP mínima** que muestra dos líneas de HTML.

- **Dónde y por qué ahí:** se guarda en `apache-php/src` porque esa carpeta se **monta** en el contenedor (en `/var/www/html`) mediante `compose.yml`. Así, lo que edites en tu host aparece dentro de Apache.
- **Qué hace el código:** entre `<?php ... ?>` se ejecuta PHP; con `echo` imprime un `<h1>` y un `<p>`.
- **Para qué sirve:** es una **prueba funcional** para confirmar 3 cosas:
  - que el contenedor **Apache+PHP** está corriendo,
  - que **PHP se interpreta** (no se ve el código en crudo),
  - que el **volumen** está bien mapeado (ves los cambios al refrescar).

```
sudo nano apache-php/src/index.php
```

```
<?php
```

```
echo "<h1>Apache + PHP en Docker</h1>";
```

```
echo "<p>Contenedor ejecutándose correctamente.</p>";
```

```
?>
```

Página PHP mínima para comprobar que **PHP ejecuta código** dentro del contenedor de Apache.



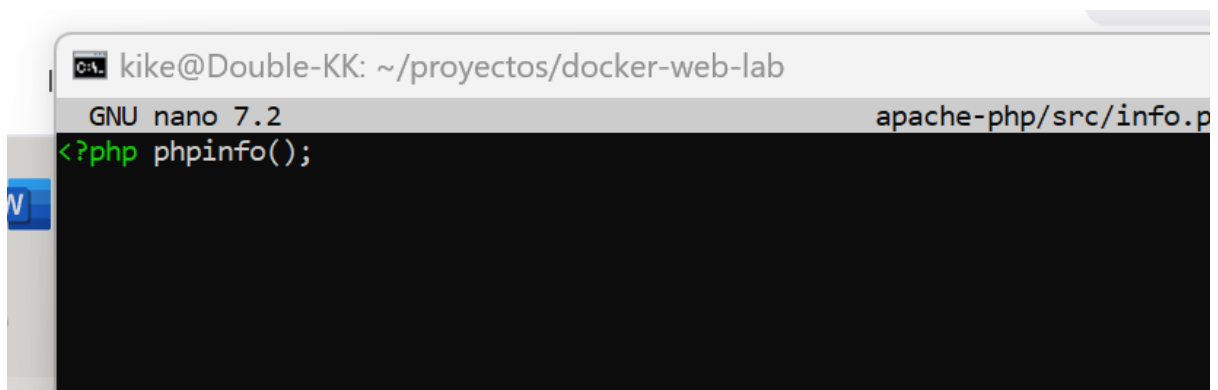
A screenshot of a terminal window. The title bar shows 'kike@Double-KK: ~/proyectos/docker-web-lab'. The terminal prompt is 'GNU nano 7.2' and the file being edited is 'apache-php/src/index.php'. The code in the editor is:

```
<?php
echo "<h1>Apache + PHP en Docker</h1>";
echo "<p>Contenedor ejecutándose correctamente.</p>";
?>
```

## Info.php

```
sudo nano apache-php/src/info.php
```

Muestra **toda la configuración de PHP** (versión, extensiones, variables)



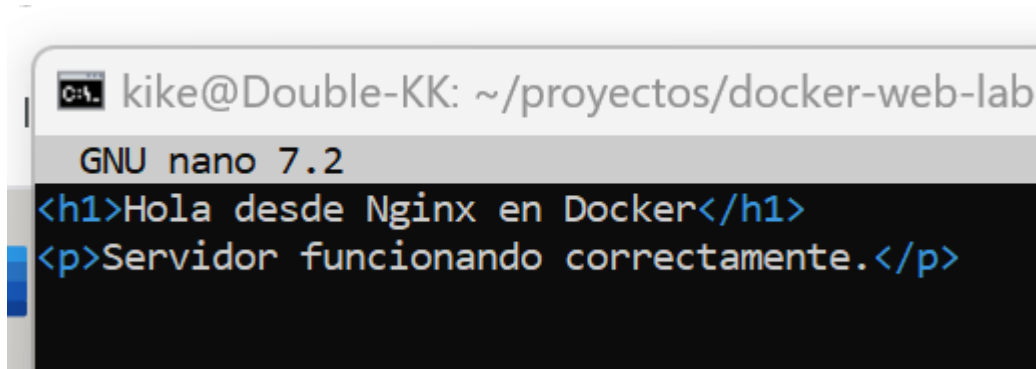
A screenshot of a terminal window. The title bar shows 'kike@Double-KK: ~/proyectos/docker-web-lab'. The terminal prompt is 'GNU nano 7.2' and the file being edited is 'apache-php/src/info.php'. The code in the editor is:

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

## Archivos para Nginx

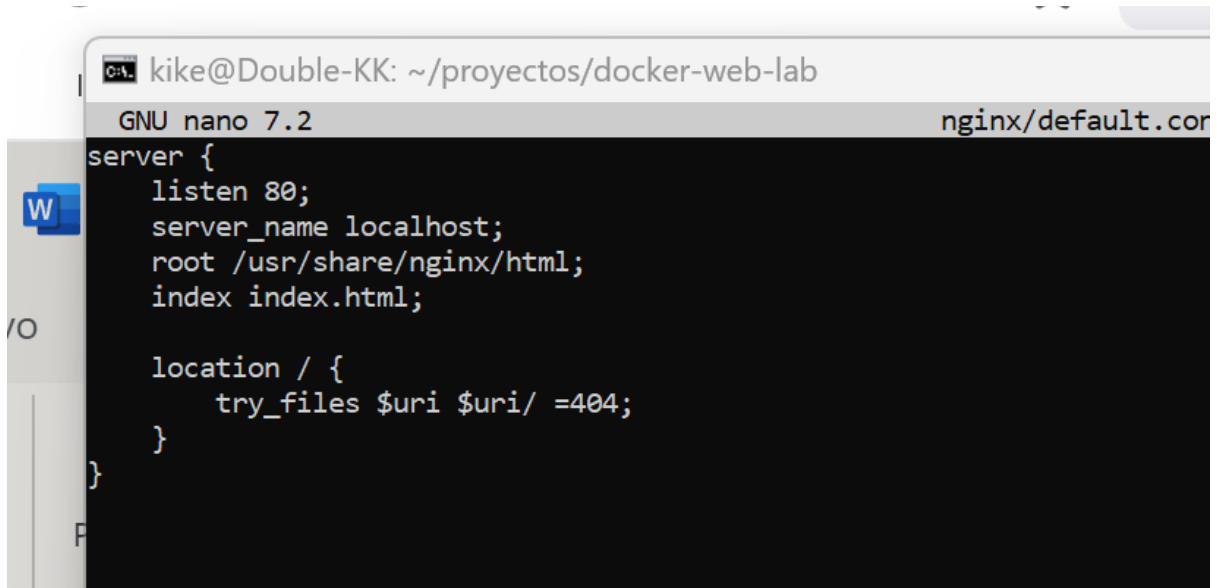
`sudo nano nginx/html/index.html`

Página estática simple para validar que **Nginx sirve contenido**.



A terminal window showing the nano editor editing the file `index.html`. The prompt is `kike@Double-KK: ~/proyectos/docker-web-lab`. The editor title is `GNU nano 7.2`. The content of the file is:

```
<h1>Hola desde Nginx en Docker</h1>
<p>Servidor funcionando correctamente.</p>
```



A terminal window showing the nano editor editing the file `default.conf`. The prompt is `kike@Double-KK: ~/proyectos/docker-web-lab`. The editor title is `GNU nano 7.2` and the file path is `nginx/default.conf`. The content of the file is:

```
server {
    listen 80;
    server_name localhost;
    root /usr/share/nginx/html;
    index index.html;

    location / {
        try_files $uri $uri/ =404;
    }
}
```

## Default.conf (virtual host de Nginx)

`sudo nano nginx/default.conf`

`server {`

`listen 80;`

```
server_name localhost;

root /usr/share/nginx/html;

index index.html;


location / {

    try_files $uri $uri/ =404;

}

}
```

Define que Nginx escuche en **80** y sirva archivos desde /usr/share/nginx/html (donde montaremos tu carpeta nginx/html). try\_files evita errores raros si la ruta no existe.

## Crear Docker Compose (compose.yml)

**Crea en la raíz del proyecto:**

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

```
services: apache: image: php:8.2-apache container_name: web_apache
ports: - "8080:80" # Host:Contenedor → http://localhost:8080
volumes: - ./apache-php/src:/var/www/html:rw

nginx: image: nginx:stable container_name: web_nginx ports: -
"8081:80" # Host:Contenedor → http://localhost:8081 volumes: -
./nginx/html:/usr/share/nginx/html:ro -
./nginx/default.conf:/etc/nginx/conf.d/default.conf:ro
```

```
kike@Double-KK: ~/proyectos/docker-web-lab
GNU nano 7.2                                compose.yml *

services:
  apache:
    image: php:8.2-apache
    container_name: web_apache
    ports:
      - "8080:80"                                # Host:Contenedor → http://Localhost:8080
    volumes:
      - ./apache-php/src:/var/www/html:rw

  nginx:
    image: nginx:stable
    container_name: web_nginx
    ports:
      - "8081:80"                                # Host:Contenedor → http://Localhost:8081
    volumes:
      - ./nginx/html:/usr/share/nginx/html:ro
      - ./nginx/default.conf:/etc/nginx/conf.d/default.conf:ro
```

### Explicación clave:

- image: usamos **imágenes oficiales** (menos errores, cero configuración extra).
- ports: mapeo **host:contenedor** → podrás entrar con el navegador a localhost:8080 (Apache/PHP) y localhost:8081 (Nginx).
- volumes: montamos tus carpetas locales dentro del contenedor → **editas archivos y recargas**, no hay que reconstruir nada.

## Arrancar los servicios

**sudo docker compose up -d**

Este comando **inicia todo el entorno definido en docker-compose.yml** (por ejemplo, un servidor web, una base de datos y un proxy, todos a la vez) y **los deja corriendo en segundo plano** para que sigan funcionando incluso si cierras la terminal.



```
kike@Double-KK: ~/proyectos/docker-web-lab
[+] Running 24/24
✔ nginx Pulled 66.2s
✔ 5c32499ab806 Pull complete 34.3s
✔ ae7b49ada9e3 Pull complete 40.5s
✔ 8e5924dfa87c Pull complete 40.5s
✔ 1d9a18bc0c05 Pull complete 40.5s
✔ be90cf255959 Pull complete 49.2s
✔ 82eb62151b9d Pull complete 49.2s
✔ 08d4f638eff8 Pull complete 49.2s
✔ apache Pulled 50.0s
✔ 8c7716127147 Pull complete 12.2s
✔ 3f814cc06e5a Pull complete 12.2s
✔ 349592d2c6d1 Pull complete 16.0s
✔ e9d7b3818d3e Pull complete 16.0s
✔ 042f7bbd46e8 Pull complete 16.3s
✔ 4a03afdd8816 Pull complete 16.3s
✔ 08a4a8d8574b Pull complete 16.3s
✔ 634ab520a54a Pull complete 16.4s
✔ 2e39efed0f04 Pull complete 21.7s
✔ ec1283305d46 Pull complete 22.3s
✔ 4ad1ce053292 Pull complete 22.3s
✔ b29d636b5d63 Pull complete 32.1s
✔ cdb5bbef0e17 Pull complete 32.2s
✔ 50e2b6face72 Pull complete 32.2s
✔ 4f4fb700ef54 Pull complete 33.0s
[+] Running 3/3
✔ Network docker-web-lab_default Created 0.0s
✔ Container web_apache Started 0.3s
✔ Container web_nginx Started 0.3s
kike@Double-KK:~/proyectos/docker-web-lab$
```

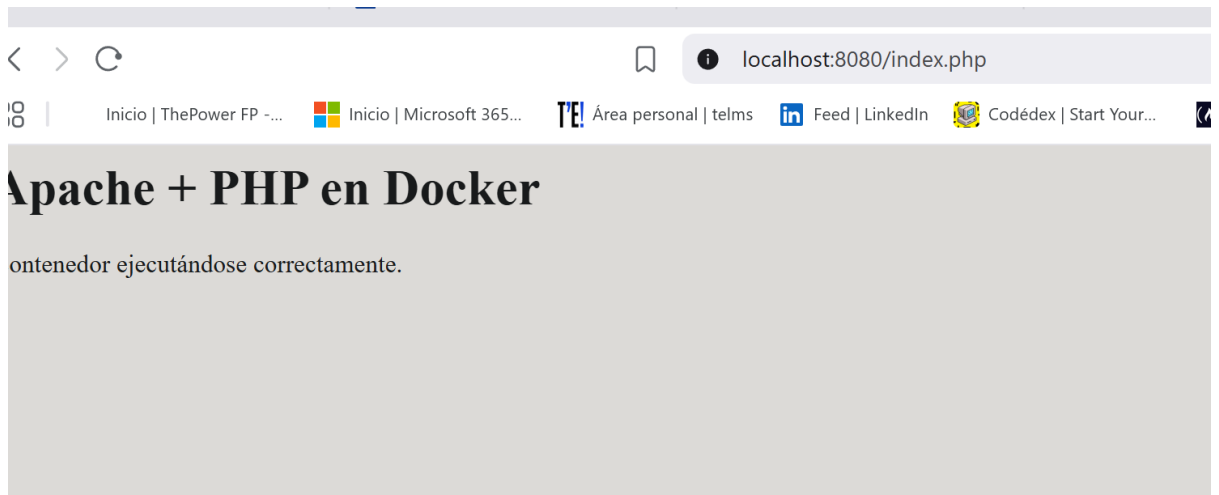
- Los contenedores **ya están creados y en ejecución**.
- Docker asignó los puertos:
  - Apache+PHP → localhost:8080
  - Nginx → localhost:8081

Comprobar docker en marcha

**sudo docker ps**

```
kike@Double-KK:~/proyectos/docker-web-lab$ sudo docker ps
CONTAINER ID   IMAGE      COMMAND                  CREATED        STATUS        PORTS
58601889faae   php:8.2-apache "docker-php-entrypoi..." About a minute ago Up About a minute 0.0.0.0:8080->80/tcp,
[::]:8080->80/tcp   web_apache
1ca3336f15f9   nginx:stable "/docker-entrypoint..." About a minute ago Up About a minute 0.0.0.0:8081->80/tcp,
[::]:8081->80/tcp   web_nginx
kike@Double-KK:~/proyectos/docker-web-lab$
```

<http://localhost:8080/index.php> → Página PHP.



<http://localhost:8081/> → Página de Nginx.



## Estructura del proyecto

```
kike@Double-KK:~/proyectos/docker-web-lab$ ls -R
.:
apache-php  compose.yml  docs  nginx

./apache-php:
src

./apache-php/src:
index.php  info.php

./docs:
capturas

./docs/capturas:

./nginx:
default.conf  html

./nginx/html:
index.html
kike@Double-KK:~/proyectos/docker-web-lab$
```

Ap  
onten

```
docker-web-lab/
├─ compose.yml
├─ apache-php/
│   └─ src/
│       ├── index.php
│       └─ info.php
└─ nginx/
    ├── default.conf
    └─ html/
        └─ index.html
```



Containers [Give feedback](#)

Container CPU usage   
 0.01% / 1600% (16 CPUs available)

Container memory usage   
 37.21MB / 7.44GB

[Show charts](#)

☒ Only show running containers

<input type="checkbox"/>	Name	Container ID	Image	Port(s)	CPU (%)	Last started	Actions
<input type="checkbox"/>	<div><div></div>welcome-to-docker</div>	e92cb142aec0	<a href="#">docker/welcome-to-docker:latest</a>	8088:80	0%	2 hours ago	<div><div></div><div></div><div></div><div></div></div>
<input type="checkbox"/>	<div><div></div>docker-web-lab</div>	-	-	-	0.01%	2 hours ago	<div><div></div><div></div><div></div><div></div></div>
<input type="checkbox"/>	<div><div></div>apache-php</div>	b6e74fa88a04	<a href="#">php:8.2-apache</a>	8080:80	0.01%	2 hours ago	<div><div></div><div></div><div></div><div></div></div>
<input type="checkbox"/>	<div><div></div>nginx-web</div>	62ad0201cd0b	<a href="#">nginx:stable</a>	8081:80	0%	2 hours ago	<div><div></div><div></div><div></div><div></div></div>

[← Learning.cs](#)