

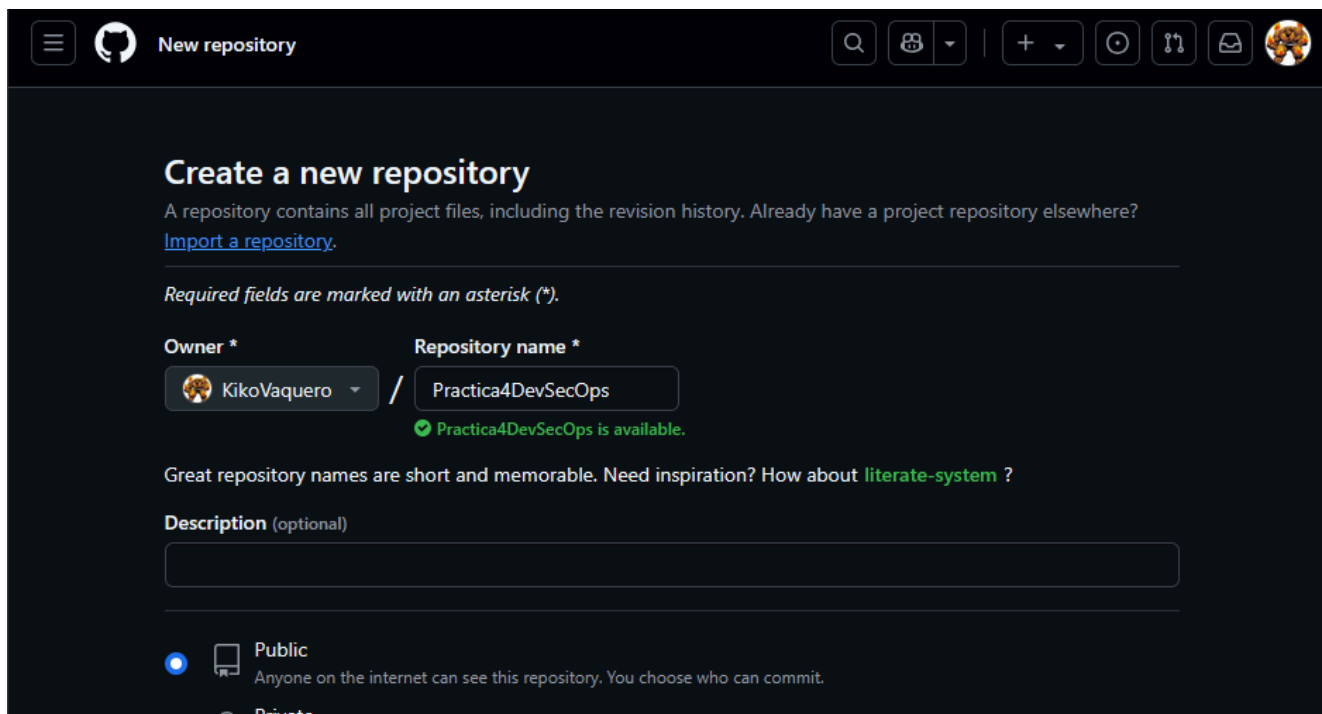
DevSecOps

- [GIT](#)
 - [Creación del repositorio](#)
- [Docker](#)
 - [Imagen PHP](#)
 - [Creación y ejecución del docker](#)
 - [Subida a git](#)
 - [Imagen Apache](#)
 - [Creación y ejecución del contenedor](#)
 - [Subida a git](#)
 - [20 contenedores a través de un comando](#)

GIT

Creación del repositorio

Creamos un nuevo repositorio en github:



Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Required fields are marked with an asterisk ().*

Owner * KikoVaquero / **Repository name *** Practica4DevSecOps

✔ Practica4DevSecOps is available.

Great repository names are short and memorable. Need inspiration? How about [literate-system](#) ?

Description (optional)

☒ **Public**
Anyone on the internet can see this repository. You choose who can commit.

☐ **Private**

Ejecutamos lo siguiente en el directorio que elijamos:

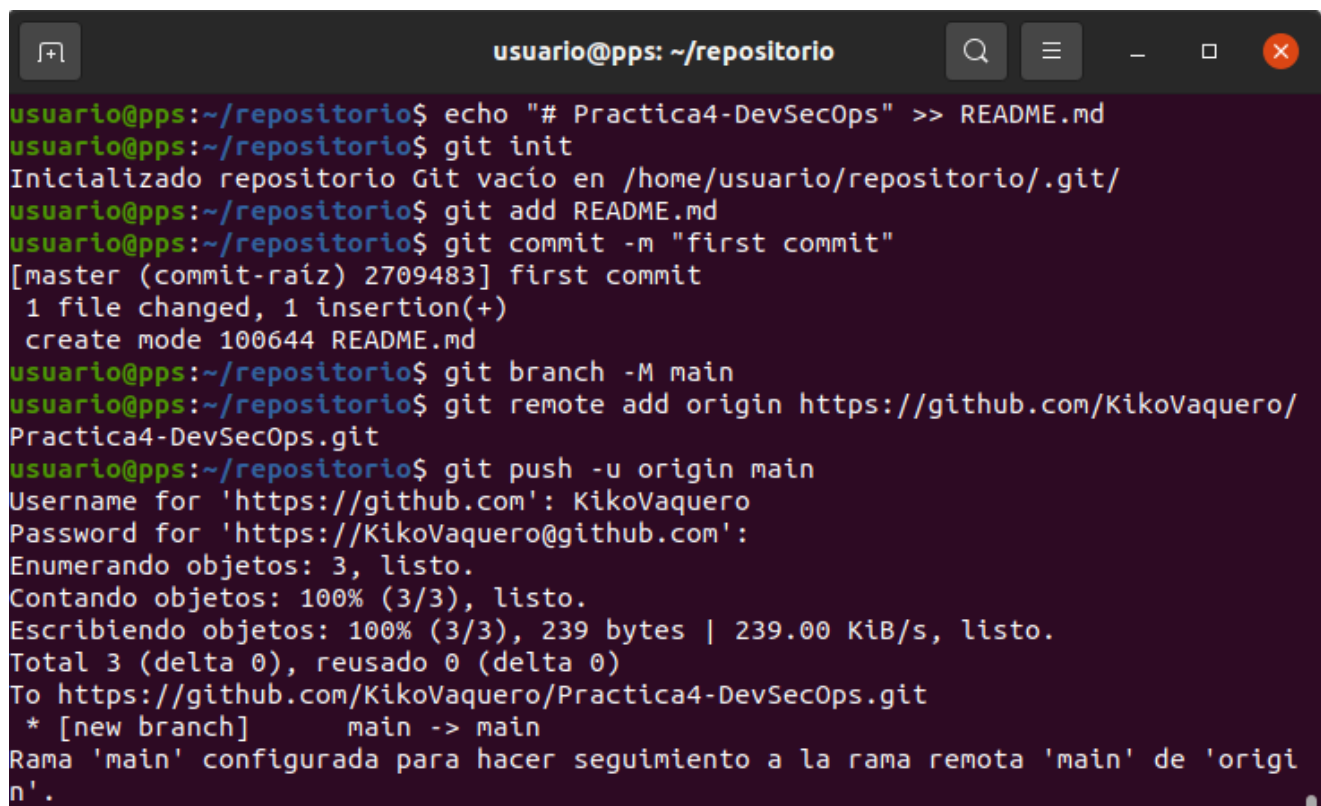
```
echo "# Practica4-DevSecOps" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
```

```
git remote add origin https://github.com/KikoVaquero/Practica4-DevSecOps.git
git push -u origin main
```

...or create a new repository on the command line

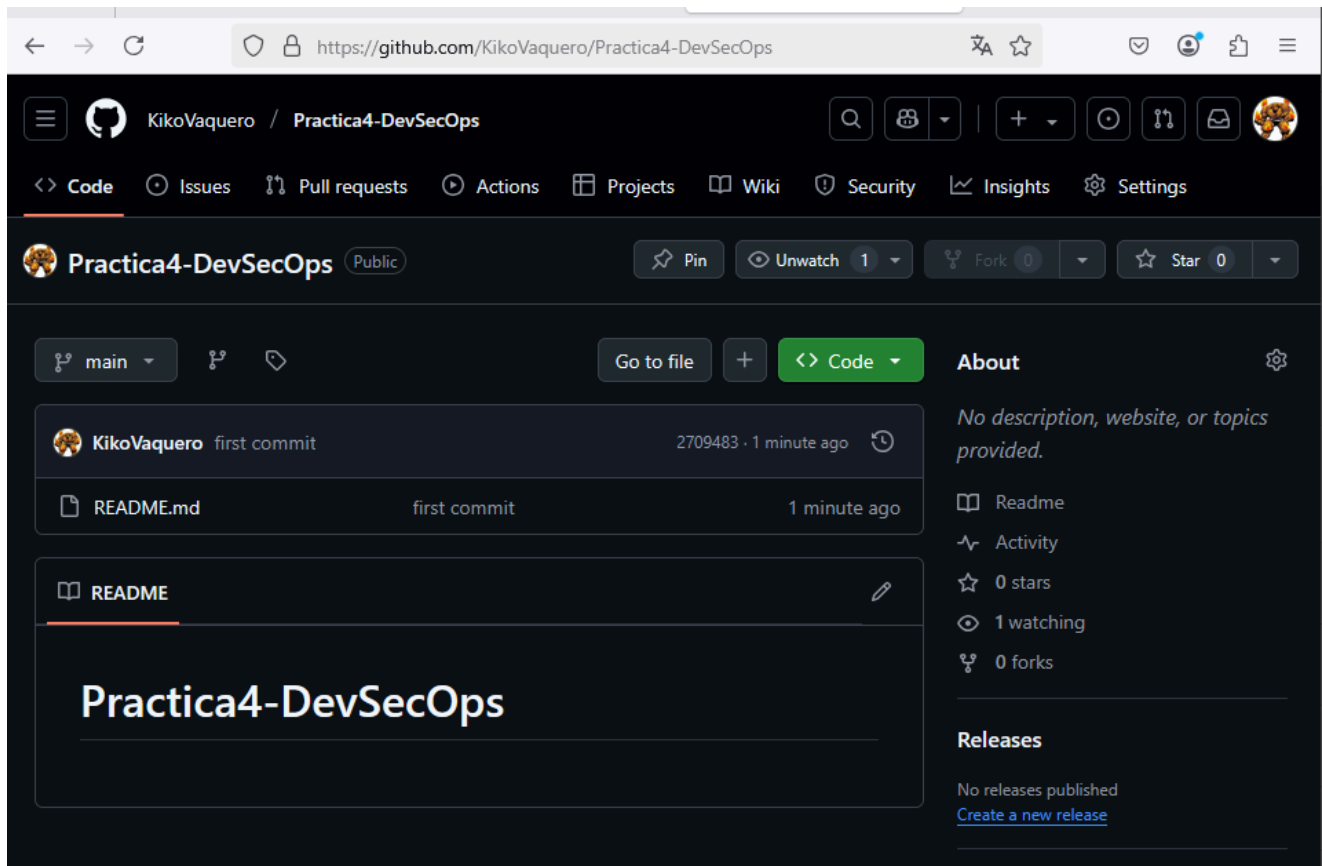
```
echo "# Practica4-DevSecOps" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin https://github.com/KikoVaquero/Practica4-DevSecOps.git
git push -u origin main
```

En este caso lo hago sobre el directorio vacío `repositorio`.



```
usuario@pps: ~/repositorio
usuario@pps:~/repositorio$ echo "# Practica4-DevSecOps" >> README.md
usuario@pps:~/repositorio$ git init
Inicializado repositorio Git vacío en /home/usuario/repositorio/.git/
usuario@pps:~/repositorio$ git add README.md
usuario@pps:~/repositorio$ git commit -m "first commit"
[master (commit-raíz) 2709483] first commit
1 file changed, 1 insertion(+)
create mode 100644 README.md
usuario@pps:~/repositorio$ git branch -M main
usuario@pps:~/repositorio$ git remote add origin https://github.com/KikoVaquero/Practica4-DevSecOps.git
usuario@pps:~/repositorio$ git push -u origin main
Username for 'https://github.com': KikoVaquero
Password for 'https://KikoVaquero@github.com':
Enumerando objetos: 3, listo.
Contando objetos: 100% (3/3), listo.
Escribiendo objetos: 100% (3/3), 239 bytes | 239.00 KiB/s, listo.
Total 3 (delta 0), reusado 0 (delta 0)
To https://github.com/KikoVaquero/Practica4-DevSecOps.git
 * [new branch]      main -> main
Rama 'main' configurada para hacer seguimiento a la rama remota 'main' de 'origin'.
```

Y con esto ya tenemos el archivo README.md en github y listo para subir archivos.



Docker

Imagen PHP

Creación y ejecución del docker

Primeramente creo la página básica de php:

```
<?php
echo "<h1>Esto es una prueba</h1>";
?>
```

```
usuario@pps: ~/practicadocker/php
usuario@pps:~/practicadocker/php$ cat php/index.php
<?php
echo "<h1>Esto es una prueba</h1>";
?>
```

Si buscamos en docker hub como lanzar una aplicación en php encontramos:

```
FROM php:8.2-cli
WORKDIR /usr/src/myapp
COPY . /usr/src/myapp
CMD [ "php", "-S", "0.0.0.0:80" ]
```

```
usuario@pps:~/practicadocker/php$ cat php/Dockerfile
FROM php:8.2-cli
WORKDIR /usr/src/myapp
COPY . /usr/src/myapp
CMD [ "php", "-S", "0.0.0.0:80" ]
```

Creamos la imagen:

```
docker build -t aplicacionphp .
```

Y lanzamos la aplicación:

```
docker run -d --name php1 -p 80:80 aplicacionphp
```

```
usuario@pps:~/practicadocker/php$ docker build -t aplicacionphp .
[+] Building 1.6s (8/8) FINISHED
=> [internal] load build definition from Dockerfile
=> => transferring dockerfile: 133B
=> [internal] load metadata for docker.io/library/php:8.2-cli
=> [internal] load .dockerignore
=> => transferring context: 2B
=> [1/3] FROM docker.io/library/php:8.2-cli@sha256:ed4385b854a7ef4aeee1108c75333443d64c937faaf7c7d28bf63a436df06428
=> [internal] load build context
=> => transferring context: 161B
=> CACHED [2/3] WORKDIR /usr/src/myapp
=> [3/3] COPY . /usr/src/myapp
=> exporting to image
=> => exporting layers
=> => writing image sha256:b85d4ffc3cb488f2bcbde407f14ceda54f35b438cc9c818e5c2342cd9e1a755f
=> => naming to docker.io/library/aplicacionphp
usuario@pps:~/practicadocker/php$ docker run -d --name php1 -p 80:80 aplicacionphp
dc75d726d3a15ae0eb12b98713f6e3203c3e0b2f60e06b63aada6ac4028ffd6b
```

Y si ahora accedemos al localhost:

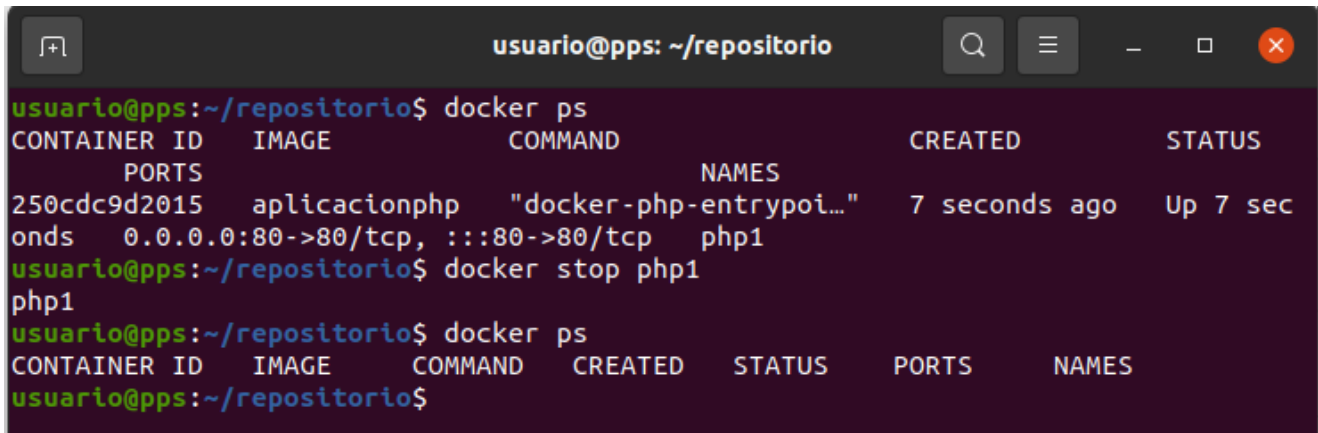


Si queremos parar el contenedor:

```
docker stop 250cdc9d2015
```

O por el nombre del contenedor:

```
docker stop php1
```



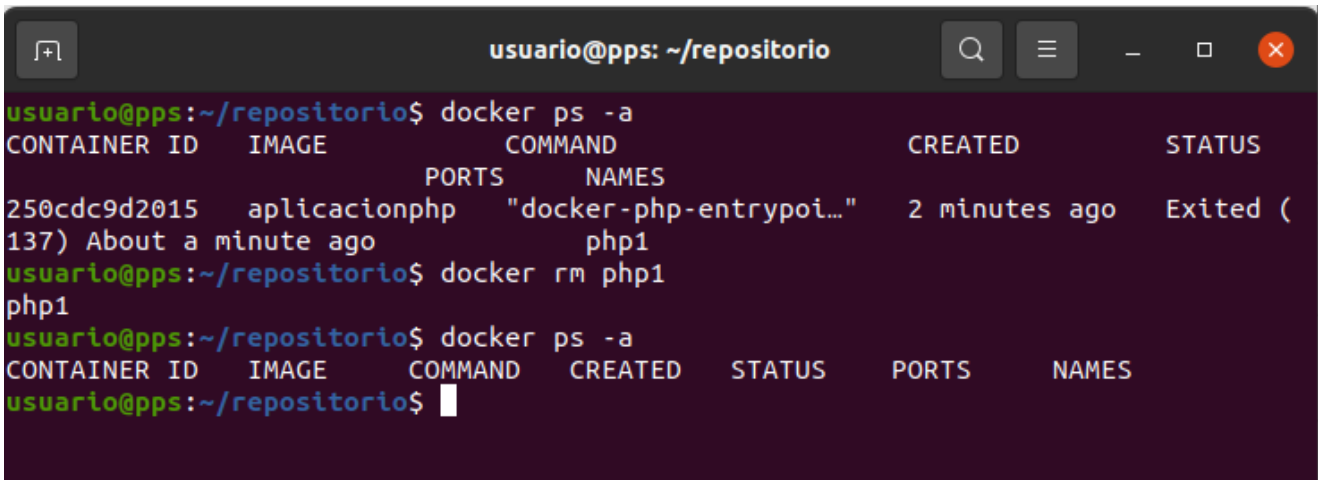
```

usuario@pps: ~/repositorio
usuario@pps:~/repositorio$ docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS
PORTS         NAMES
250cdc9d2015   aplicacionphp  "docker-php-entrypoi...  7 seconds ago Up 7 sec
onds          0.0.0.0:80->80/tcp, :::80->80/tcp  php1
usuario@pps:~/repositorio$ docker stop php1
php1
usuario@pps:~/repositorio$ docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS         NAMES
usuario@pps:~/repositorio$

```

Para eliminarlo igual, con el id o el nombre del contenedor:

```
docker rm php1
```



```

usuario@pps: ~/repositorio
usuario@pps:~/repositorio$ docker ps -a
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS
PORTS         NAMES
250cdc9d2015   aplicacionphp  "docker-php-entrypoi...  2 minutes ago Exited (
137) About a minute ago  php1
usuario@pps:~/repositorio$ docker rm php1
php1
usuario@pps:~/repositorio$ docker ps -a
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS         NAMES
usuario@pps:~/repositorio$

```

Subida a git

Para subirlo a git voy a copiar el directorio con el `Dockerfile` al repositorio:

```

usuario@pps:~/repositorio$ cp -R ../practicadocker/php/php/ .
usuario@pps:~/repositorio$ ls
php  README.md
usuario@pps:~/repositorio$

```

Una vez tenemos los cambios:

```

git add *
git commit -m "Dockerfile php"
git push origin main

```

```

usuario@pps:~/repositorio$ git add *
usuario@pps:~/repositorio$ git commit -m "Dockerfile php"
[main 8b7fb5e] Dockerfile php
 2 files changed, 7 insertions(+)
  create mode 100644 php/Dockerfile
  create mode 100644 php/index.php
usuario@pps:~/repositorio$ git push origin main
Username for 'https://github.com': KikoVaquero
Password for 'https://KikoVaquero@github.com':
Enumerando objetos: 6, listo.
Contando objetos: 100% (6/6), listo.
Compresión delta usando hasta 2 hilos
Comprimiendo objetos: 100% (4/4), listo.
Escribiendo objetos: 100% (5/5), 488 bytes | 488.00 KiB/s, listo.
Total 5 (delta 0), reusado 0 (delta 0)
To https://github.com/KikoVaquero/Practica4-DevSecOps.git
   2709483..8b7fb5e  main -> main
usuario@pps:~/repositorio$

```

Y ya tendríamos los archivos en el repositorio de github.

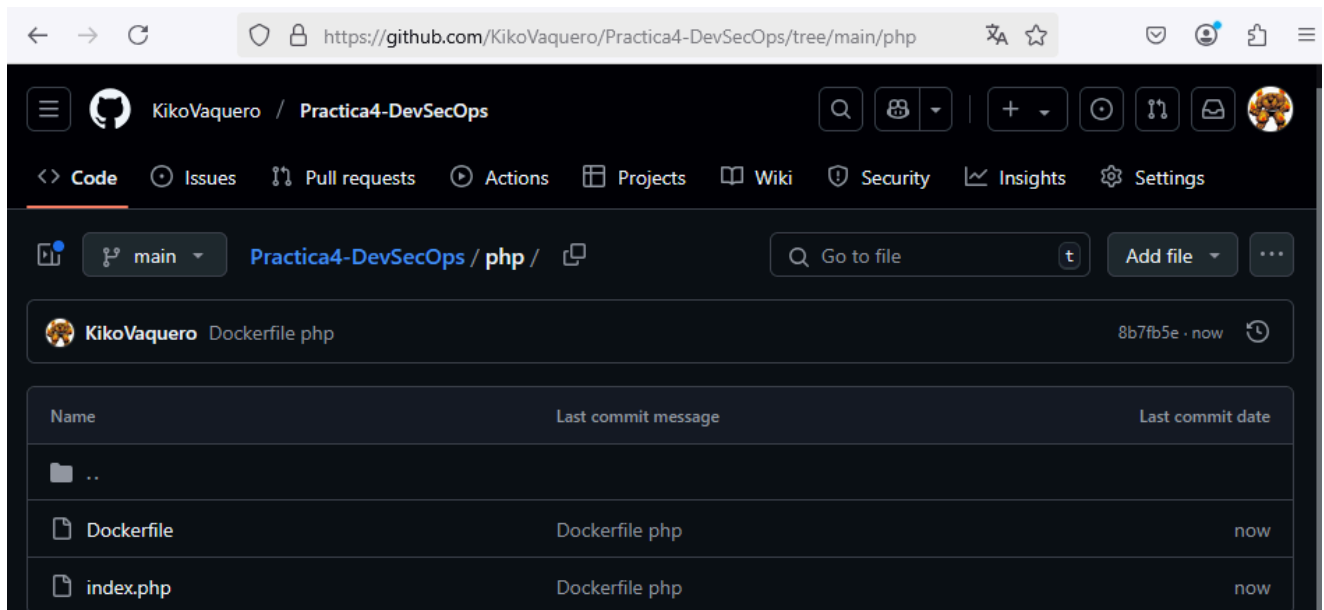


Imagen Apache

Creación y ejecución del contenedor

Lo aplicación que va a lanzar en este caso apache va a ser la que creamos en el anterior apartado, es decir voy a hacer un `git clone` de mi propio repositorio:

```

FROM ubuntu:20.04
ENV DEBIAN_FRONTEND=noninteractive
RUN apt update
RUN apt install -y apache2
RUN apt install -y apache2-utils
RUN apt install -y git php
RUN apt clean

```

```

WORKDIR /var/www/html
RUN mkdir app
WORKDIR /var/www/html/app
RUN git clone https://github.com/KikoVaquero/Practica4-DevSecOps/ .
EXPOSE 80
CMD ["apachectl", "-D", "FOREGROUND"]

```

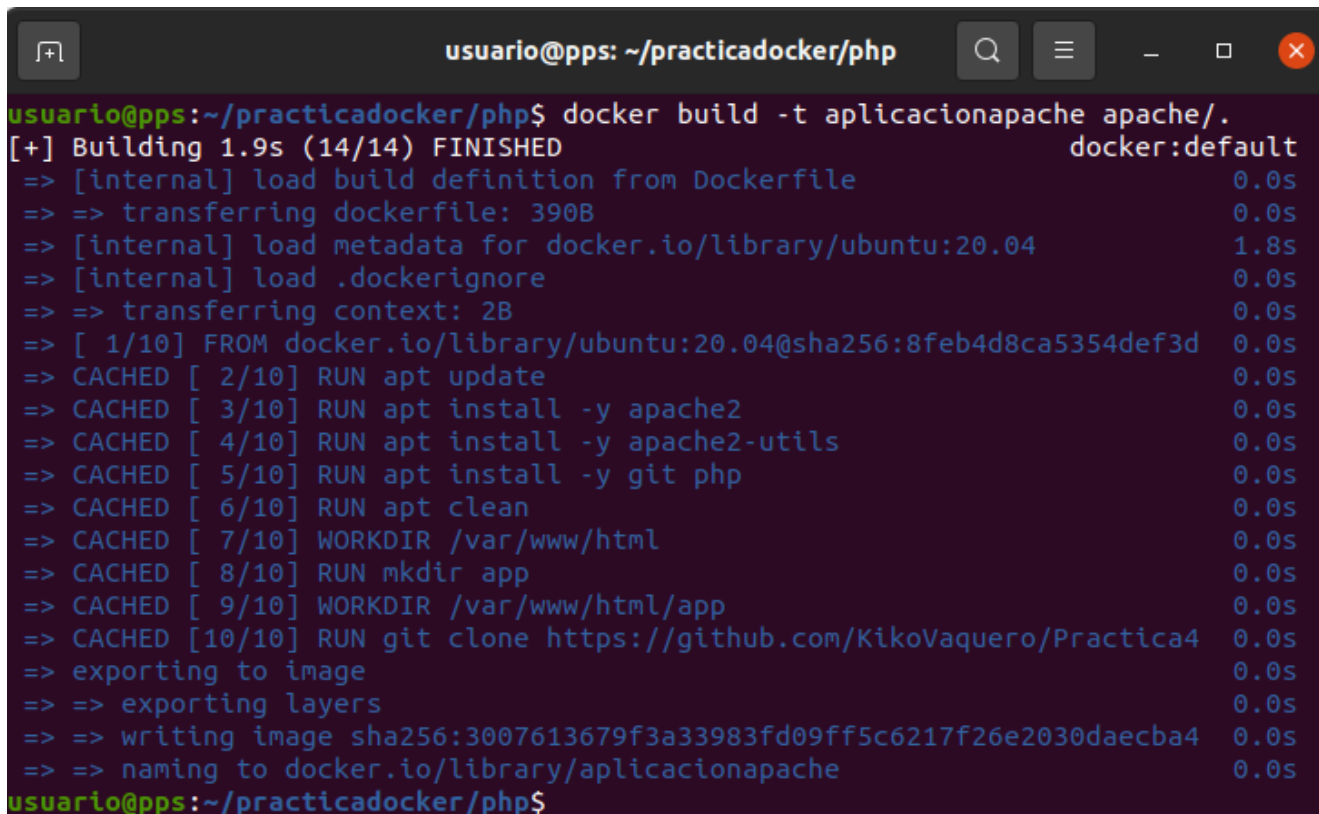
```

usuario@pps:~/practicadocker/php$ cat apache/Dockerfile
FROM ubuntu:20.04
ENV DEBIAN_FRONTEND=noninteractive
RUN apt update
RUN apt install -y apache2
RUN apt install -y apache2-utils
RUN apt install -y git php
RUN apt clean
WORKDIR /var/www/html
RUN mkdir app
WORKDIR /var/www/html/app
RUN git clone https://github.com/KikoVaquero/Practica4-DevSecOps/ .
EXPOSE 80
CMD ["apachectl", "-D", "FOREGROUND"]

```

Para crear la imagen vamos a ejecutar:

```
docker build -t aplicacionapache apache/.
```



```

usuario@pps: ~/practicadocker/php
usuario@pps:~/practicadocker/php$ docker build -t aplicacionapache apache/.
[+] Building 1.9s (14/14) FINISHED                                docker:default
=> [internal] load build definition from Dockerfile                0.0s
=> => transferring dockerfile: 390B                                0.0s
=> [internal] load metadata for docker.io/library/ubuntu:20.04    1.8s
=> [internal] load .dockerignore                                   0.0s
=> => transferring context: 2B                                       0.0s
=> [ 1/10] FROM docker.io/library/ubuntu:20.04@sha256:8feb4d8ca5354def3d 0.0s
=> CACHED [ 2/10] RUN apt update                                    0.0s
=> CACHED [ 3/10] RUN apt install -y apache2                        0.0s
=> CACHED [ 4/10] RUN apt install -y apache2-utils                 0.0s
=> CACHED [ 5/10] RUN apt install -y git php                       0.0s
=> CACHED [ 6/10] RUN apt clean                                     0.0s
=> CACHED [ 7/10] WORKDIR /var/www/html                             0.0s
=> CACHED [ 8/10] RUN mkdir app                                     0.0s
=> CACHED [ 9/10] WORKDIR /var/www/html/app                         0.0s
=> CACHED [10/10] RUN git clone https://github.com/KikoVaquero/Practica4 0.0s
=> exporting to image                                              0.0s
=> => exporting layers                                              0.0s
=> => writing image sha256:3007613679f3a33983fd09ff5c6217f26e2030daecba4 0.0s
=> => naming to docker.io/library/aplicacionapache                 0.0s
usuario@pps:~/practicadocker/php$

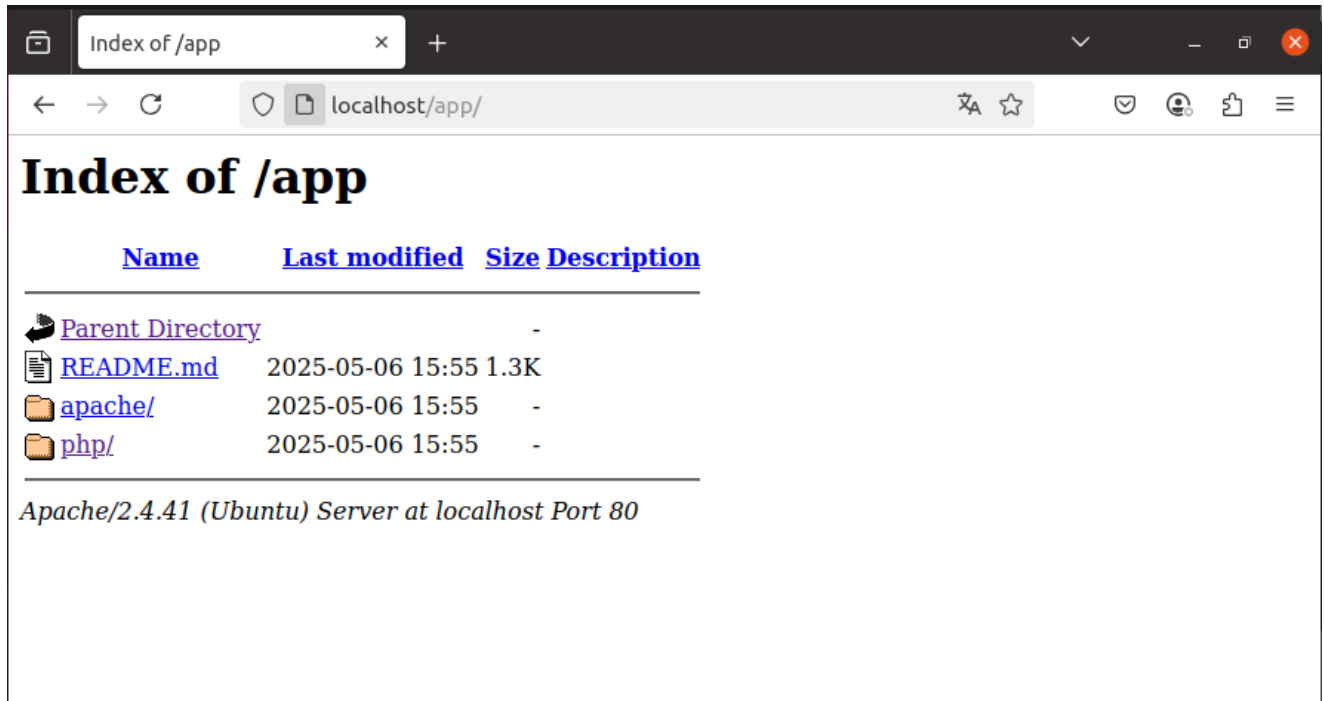
```

Para crear el contenedor y ejecutarlo:

```
docker run -d --name apache1 -p 80:80 aplicacionapache
```

```
usuario@pps: ~/practicadocker/php$ docker run -d --name apache1 -p 80:80 aplicaci
onapache
4810748dbb5546ff8be1124b2f2b99ee3b9e9a7ec089ede06f31e1c9dea401be
```

Y ya podríamos acceder a la página creada en el puerto 80:



Para parar el contenedor podemos utilizar:

```
docker stop 4810748dbb55
```



```

usuario@pps: ~/repositorio
usuario@pps:~/repositorio$ docker ps
CONTAINER ID   IMAGE             COMMAND                  CREATED        STATUS        PORTS
4810748dbb55   aplicacionapache  "apachectl -D FOREGR..." 21 minutes ago Up 2
1 minutes     0.0.0.0:80->80/tcp, :::80->80/tcp  apache1
usuario@pps:~/repositorio$ docker stop 4810748dbb55
4810748dbb55
usuario@pps:~/repositorio$ docker ps
CONTAINER ID   IMAGE             COMMAND                  CREATED        STATUS        PORTS        NAMES
usuario@pps:~/repositorio$

```

Para borrarlo:

```
docker rm 4810748dbb55
```

```

usuario@pps: ~/repositorio
usuario@pps:~/repositorio$ docker ps -a
CONTAINER ID   IMAGE             COMMAND                  CREATED        STATUS        PORTS        NAMES
4810748dbb55   aplicacionapache  "apachectl -D FOREGR..." 22 minutes ago Exit
ed (137) 35 seconds ago  apache1
usuario@pps:~/repositorio$ docker rm 4810748dbb55
4810748dbb55
usuario@pps:~/repositorio$ docker ps -a
CONTAINER ID   IMAGE             COMMAND                  CREATED        STATUS        PORTS        NAMES
usuario@pps:~/repositorio$

```

Subida a git

Copiamos los archivos al repositorio:

```

usuario@pps: ~/repositorio
usuario@pps:~/repositorio$ cp -R ../practicadocker/php/apache/ .
usuario@pps:~/repositorio$ ls
apache  php  README.md
usuario@pps:~/repositorio$

```

Los subimos al repositorio ejecutando:

```

git add *
git commit -m "Dockerfile php"
git push origin main

```

```

usuario@pps: ~/repositorio
usuario@pps:~/repositorio$ git add *
usuario@pps:~/repositorio$ git commit -m "Dockerfile php"
[main effb5c5] Dockerfile php
 1 file changed, 13 insertions(+)
 create mode 100644 apache/Dockerfile
usuario@pps:~/repositorio$ git push origin main
Username for 'https://github.com': KikoVaquero
Password for 'https://KikoVaquero@github.com':
Enumerando objetos: 5, listo.
Contando objetos: 100% (5/5), listo.
Compresión delta usando hasta 2 hilos
Comprimiendo objetos: 100% (3/3), listo.
Escribiendo objetos: 100% (4/4), 585 bytes | 585.00 KiB/s, listo.
Total 4 (delta 0), reusado 0 (delta 0)
To https://github.com/KikoVaquero/Practica4-DevSecOps.git
 8b7fb5e..effb5c5  main -> main
usuario@pps:~/repositorio$

```

Y los cambios ya se ven en el repositorio:

```

usuario@pps: ~/repositorio
usuario@pps:~/repositorio$ git add *
usuario@pps:~/repositorio$ git commit -m "Dockerfile apache"
[main 7fe2f7b] Dockerfile apache
 1 file changed, 13 insertions(+)
 create mode 100644 apache/Dockerfile
usuario@pps:~/repositorio$ git push origin main
Username for 'https://github.com': KikoVaquero
Password for 'https://KikoVaquero@github.com':
Enumerando objetos: 5, listo.
Contando objetos: 100% (5/5), listo.
Compresión delta usando hasta 2 hilos
Comprimiendo objetos: 100% (3/3), listo.
Escribiendo objetos: 100% (4/4), 586 bytes | 586.00 KiB/s, listo.
Total 4 (delta 0), reusado 0 (delta 0)
To https://github.com/KikoVaquero/Practica4-DevSecOps.git
 8b7fb5e..7fe2f7b  main -> main
usuario@pps:~/repositorio$

```

En el caso de que queramos deshacer el push que acabamos de hacer:

```

git reset --hard HEAD~1
git push --force          # fuerza el cambio en el remoto

```

20 contenedores a través de un comando

El comando utilizado es:

```

for i in {1..20};do docker run -d --name apache$i -p 30$i:80
aplicacionapache; done

```

```
usuario@pps: ~  
usuario@pps:~$ for i in {1..20};do docker run -d --name apache$i -p 30$i:80 aplicacionapache; done  
231e84a9f86a358c26aa58edc07f10cca86aaba703b5dcbc0cdfb07776a7274  
d4284ef3fe94fc8423203837304e51845566b84e95d3f2fe58a7b7a3acffde7e  
c55a017348ed14b362422b04343c0ce94d0461c79c8a89aec5ab603843a4e8f9  
b6f545e7c05106bcd381f33a96ab86f24630490c38494f8237642a6256f6d0bb  
173d21a5e93d48b6c8f1b1a846ef6e1a59b91ec23e5557298fb490f79dee36c7  
259270012e969506647c1b24df8430b22516beddb693aa92e9a11aa89cf43041  
32e1211ed7a501ad5855baede68c2517ebff49688e9d03c8f77cb827656ce4df  
5b48a049c7e1fa52146ca1ab3395de755c380bf7324af8488189e26fadec656b  
37da59b772552d321ece76a91f91a2d282d19a9fc1acb5f0c6beb5a066997573  
52da6ac9c1e238f26ca40ced7d5d4ff6e948e2806f77f747c1362cff554e515  
bbefce3c848a7e43da233c33f980d3baf85c97e2d92d6fec76e05f488cd0aa42  
3f191217b89756ea8471042040d484ef153a5f2094d5ebad454f961f0e403a7f  
6568625ce1d6638f89ea69208a253a8d86892b988a788100dfd08f9c455b02ca  
7462d5ed80b886357a1165d18da7d2625ba9e5d3fb4b88174e1e9c95cd7d567f  
4dffae95422dc7083030d381b319824405295688f04fa1dcd74694be908c5a50  
800e68e73008dbecf41cb90a9349619f7f52943fc6d9c8d5e965160dd6dc4e9f  
5b42b55e5323ec1f66088f69bd50b3b0c568eb8235e8ba86853a1886d541007b  
7f145846c5a97d8a340af8760ed4648e2d1994a4edaecbb8a1b587abd4a5419a  
e1d9a79bf0034352f558f64b1862ac6cf9ee813fe15a6d72a13881470b03e73c  
b159be5f4f36303b55ef6de0ed65a386fd92d7908a3109247435518fd253bb0a  
usuario@pps:~$
```

Para ver que los contenedores están corriendo:

```
docker ps
```

```

usuario@pps: ~
usuario@pps:~$ docker ps
CONTAINER ID   IMAGE               COMMAND                  CREATED        STATUS        PORTS
S
b159be5f4f36   aplicacionapache   "apachectl -D FOREGR..." 51 seconds ago Up 50 seconds 0.0.
0.0:3020->80/tcp, [::]:3020->80/tcp   apache20
e1d9a79bf003   aplicacionapache   "apachectl -D FOREGR..." 51 seconds ago Up 50 seconds 0.0.
0.0:3019->80/tcp, [::]:3019->80/tcp   apache19
7f145846c5a9   aplicacionapache   "apachectl -D FOREGR..." 51 seconds ago Up 50 seconds 0.0.
0.0:3018->80/tcp, [::]:3018->80/tcp   apache18
5b42b55e5323   aplicacionapache   "apachectl -D FOREGR..." 51 seconds ago Up 51 seconds 0.0.
0.0:3017->80/tcp, [::]:3017->80/tcp   apache17
800e68e73008   aplicacionapache   "apachectl -D FOREGR..." 51 seconds ago Up 51 seconds 0.0.
0.0:3016->80/tcp, [::]:3016->80/tcp   apache16
4dffae95422d   aplicacionapache   "apachectl -D FOREGR..." 52 seconds ago Up 51 seconds 0.0.
0.0:3015->80/tcp, [::]:3015->80/tcp   apache15
7462d5ed80b8   aplicacionapache   "apachectl -D FOREGR..." 52 seconds ago Up 51 seconds 0.0.
0.0:3014->80/tcp, [::]:3014->80/tcp   apache14
6568625ce1d6   aplicacionapache   "apachectl -D FOREGR..." 52 seconds ago Up 51 seconds 0.0.
0.0:3013->80/tcp, [::]:3013->80/tcp   apache13
3f191217b897   aplicacionapache   "apachectl -D FOREGR..." 52 seconds ago Up 52 seconds 0.0.
0.0:3012->80/tcp, [::]:3012->80/tcp   apache12
bbefce3c848a   aplicacionapache   "apachectl -D FOREGR..." 52 seconds ago Up 52 seconds 0.0.
0.0:3011->80/tcp, [::]:3011->80/tcp   apache11
52da6acf9c1e   aplicacionapache   "apachectl -D FOREGR..." 52 seconds ago Up 52 seconds 0.0.
0.0:3010->80/tcp, [::]:3010->80/tcp   apache10
37da59b77255   aplicacionapache   "apachectl -D FOREGR..." 53 seconds ago Up 52 seconds 0.0.
0.0:309->80/tcp, [::]:309->80/tcp     apache9
5b48a049c7e1   aplicacionapache   "apachectl -D FOREGR..." 53 seconds ago Up 52 seconds 0.0.
0.0:308->80/tcp, [::]:308->80/tcp     apache8
32e1211ed7a5   aplicacionapache   "apachectl -D FOREGR..." 53 seconds ago Up 52 seconds 0.0.
0.0:307->80/tcp, [::]:307->80/tcp     apache7
259270012e96   aplicacionapache   "apachectl -D FOREGR..." 53 seconds ago Up 53 seconds 0.0.
0.0:306->80/tcp, [::]:306->80/tcp     apache6
173d21a5e93d   aplicacionapache   "apachectl -D FOREGR..." 53 seconds ago Up 53 seconds 0.0.
0.0:305->80/tcp, [::]:305->80/tcp     apache5
b6f545e7c051   aplicacionapache   "apachectl -D FOREGR..." 54 seconds ago Up 53 seconds 0.0.
0.0:304->80/tcp, [::]:304->80/tcp     apache4
c55a017348ed   aplicacionapache   "apachectl -D FOREGR..." 54 seconds ago Up 53 seconds 0.0.
0.0:303->80/tcp, [::]:303->80/tcp     apache3
d4284ef3fe94   aplicacionapache   "apachectl -D FOREGR..." 54 seconds ago Up 53 seconds 0.0.
0.0:302->80/tcp, [::]:302->80/tcp     apache2
231e84a9f86a   aplicacionapache   "apachectl -D FOREGR..." 54 seconds ago Up 53 seconds 0.0.
0.0:301->80/tcp, [::]:301->80/tcp     apache1
usuario@pps:~$

```

Si ahora por ejemplo accedemos a <http://localhost:309/app/php>



Para parar todos los contenedores a la vez:

```
docker stop $(docker ps -aq)
```

```
usuario@pps:~$ docker stop $(docker ps -aq)
8020bf287a33
3ef719dd7fb8
e799cf1f7e47
9e7954db8ddc
f7c1c5ec8d6b
bacd35ebaa68
00e87f83c008
ed3f6109fa98
f23aaea29c0a
14d3b4cc4a1d
d2cfed04d33c
e70c0ff3357a
9036d057ff79
a3d7924f4b71
2bc67b0a310f
ea812c2d2538
18f0a6f149f8
52a948a920bf
dbaee8183cca
05b00037823a
usuario@pps:~$ docker ps
CONTAINER ID   IMAGE      COMMAND                  CREATED        STATUS        PORTS        NAMES
usuario@pps:~$
```

Para borrar todos a la vez:

```
docker rm $(docker ps -aq)
```

```
usuario@pps: ~  
usuario@pps:~$ docker rm $(docker ps -aq)  
8020bf287a33  
3ef719dd7fb8  
e799cf1f7e47  
9e7954db8ddc  
f7c1c5ec8d6b  
bacd35ebaa68  
00e87f83c008  
ed3f6109fa98  
f23aaea29c0a  
14d3b4cc4a1d  
d2cfed04d33c  
e70c0ff3357a  
9036d057ff79  
a3d7924f4b71  
2bc67b0a310f  
ea812c2d2538  
18f0a6f149f8  
52a948a920bf  
dbaee8183cca  
05b00037823a  
usuario@pps:~$ docker ps -a  
CONTAINER ID    IMAGE    COMMAND    CREATED    STATUS    PORTS    NAMES  
usuario@pps:~$
```