

## Contenedores con Docker. Proxy inverso con nginx

1. Se trata de emplear la imagen del servicio nginx de la práctica anterior, pero en este caso para habilitarlo como proxy inverso de tal modo que permita redirigir las peticiones que lleguen al puerto del proxy en la maquina anfitrión hacia las direcciones IP y puertos de los servidores web que tenemos en distintos contenedores. Para ello:

- a) Crea en tu sistema Docker dos contenedores de nombres c1 y c2 a partir de la imagen de ubuntu/apache2. Habilita un par de carpetas en la maquina real que permitan el acceso a los directorios /var/www/html de cada contenedor. No es necesario mapear puertos web en este caso ya que la gestión de acceso se realizará desde el proxy
- b) Personaliza cada una de las páginas de inicio index.html empleando las carpetas que se habilitaron anteriormente o bien copia directamente los archivos index.html con el comando:

```
docker cp index.html <nombre_contenedor>:/var/www/html
```

- c) Verifica con docker inspect o desde Docker Desktop, la dirección IP privada de cada uno de los contenedores (que estarán en la dirección de red 172.17.0.0/24)
- d) Inicia tu contenedor nginx y accede a una consola interactiva con este contenedor mediante el comando:

```
docker exec -it <nombre_contenedor> bash
```

o bien desde la consola accesible desde Docker Desktop

- e) Una vez dentro debes acceder a la carpeta /etc/nginx/conf.d y crear dentro de ella un par de archivos nuevos de nombres sitio1.conf y sitio2.conf con contenido similar a este:

```
server {
    listen      80;
    server_name www.sitioX.com;

    location / {
        proxy_pass http://172.17.0.X:80;
    }
}
```

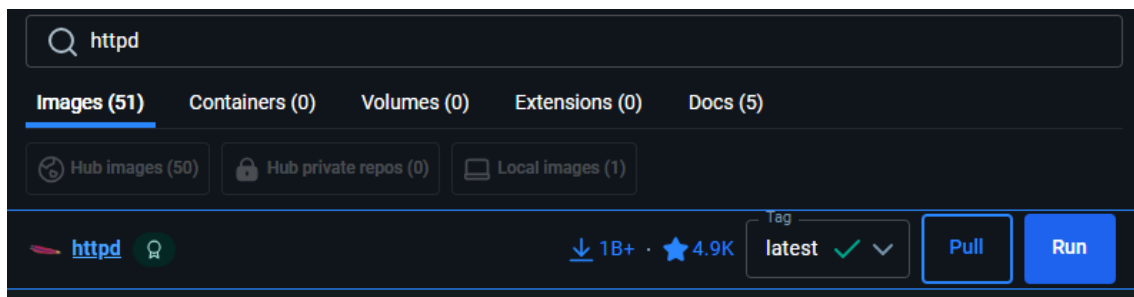
donde [www.sitioX.com](http://www.sitioX.com) y 172.17.0.X deben ajustarse por los valores adecuados

- f) Una vez completado debe pararse y arrancar de nuevo el contenedor nginx
- g) Finalmente, desde la maquina Windows 10 y en su fichero hosts debe establecerse los dos valores de [www.sitioX.com](http://www.sitioX.com) como localhost y verificar desde un navegador cualquiera que al acceder a las direcciones [www.sitio1.com](http://www.sitio1.com) y [www.sitio2.com](http://www.sitio2.com) se visualizan las páginas index.html de cada uno de los dos contenedores

2. Acceso a contenedores desde VS Code. Se trata de habilitar otra vía de acceso al sistema de ficheros de los contenedores en ejecución. Para ello:

- a) Se debe instalar en VS Code la extensión “Docker for Visual Studio Code”
- b) Se nos creará un acceso a la extensión en la barra de tareas lateral izquierda. Desde ella se deben abrir los directorios /var/www/html de cada contenedor con el propósito de modificar directamente el fichero index.html
- c) Verificar desde el navegador que los cambios son efectivos

1.- Primero haremos un pull de una imagen httpd.



En la documentación nos aparecerá como lanzar el contenedor, nos aparecerá con un volumen, pero nosotros modificaremos el comando y lo quitaremos.

```
H:\>docker run -dit --name c1 -p 8000:80 httpd:2.4
Unable to find image 'httpd:2.4' locally
2.4: Pulling from library/httpd
Digest: sha256:e19cdd61f51985351ca9867d384cf1b050487d26bb1b49c470f2fcda1b5f276c
Status: Downloaded newer image for httpd:2.4
9130b2fc013c42947728ab16205c8b86b4ad23d930a212b56bb72e5dc3bad3cc
H:\>
```

Lanzamos el contenedor c1 y c2.

```
H:\>docker run -dit --name c2 -p 8001:80 httpd:2.4
```

Los puertos de estos contenedores no hacen falta ponerlos ya que realizaremos una conexión por proxy.

2.- Ahora nos situaremos en la carpeta donde tengamos nuestro index.html, en este caso web.

```
C:\Users\guillermosr26\web>dir
El volumen de la unidad C no tiene etiqueta.
El número de serie del volumen es: CAA6-66B2

Directorio de C:\Users\guillermosr26\web

10/12/2025  09:04    <DIR>          .
18/12/2025  11:30    <DIR>          ..
10/12/2025  09:09                69 Dockerfile
09/01/2026  10:47               265 index.html
                2 archivos             334 bytes
                2 dirs  113.851.510.784 bytes libres

C:\Users\guillermosr26\web>docker cp index.html c1:/usr/local/apache2/htdocs/
```

Y copiaremos el index.html en la carpeta que nos dice la documentación.

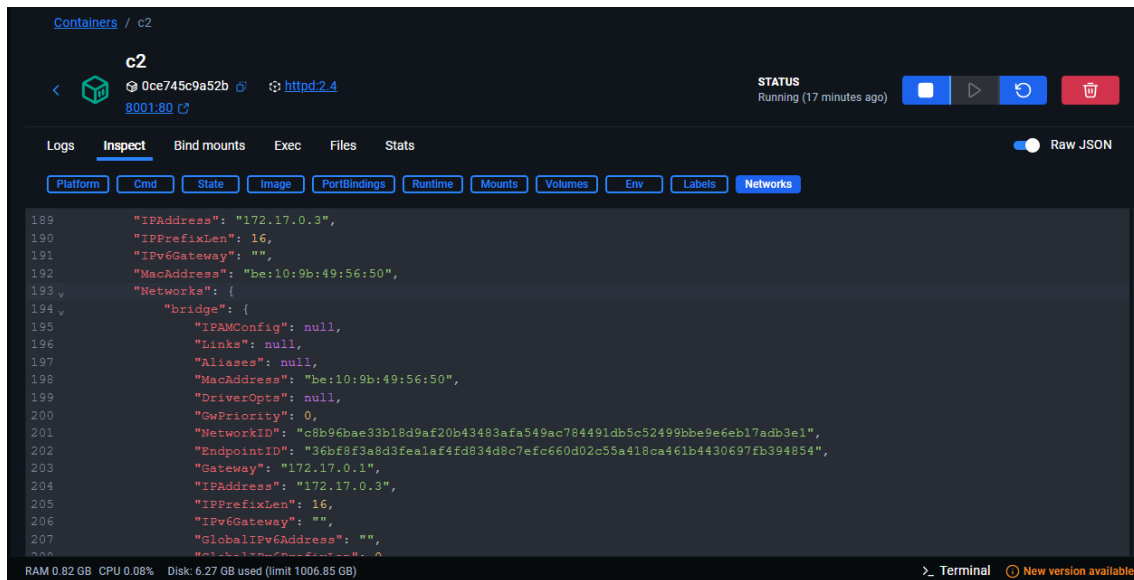
Haremos lo mismo para los dos contenedores.

```
C:\Users\guillermosr26\web>docker cp index.html c2:/usr/local/apache2/htdocs/
```

3.- Ahora con el comando docker inspect c1 podremos observar la ip del contenedor.

```
{
  "IPAddress": "3a:94:3e:2f:69:63",
  "Networks": {
    "bridge": {
      "IPAMConfig": null,
      "Links": null,
      "Aliases": null,
      "MacAddress": "3a:94:3e:2f:69:63",
      "DriverOpts": null,
      "GwPriority": 0,
      "NetworkID": "c8b96bae33b18d9af20b43483afa549ac784491db5c52499bbe9e6eb17adb3e1",
      "EndpointID": "006c4d2f3569f555ece8158614ebc7bfecb741f93066e9838a1fd2bfe146390",
      "Gateway": "172.17.0.1",
      "IPAddress": "172.17.0.2",
      "IPPrefixLen": 16,
      "IPv6Gateway": "",
      "GlobalIPv6Address": "",
      "GlobalIPv6PrefixLen": 0,
      "DNSNames": null
    }
  }
}
```

También se puede ver desde el docker desktop, de manera mas sencilla, para ello daremos clic en nuestro contenedor, le daremos al apartado inspect y finalmente networks.



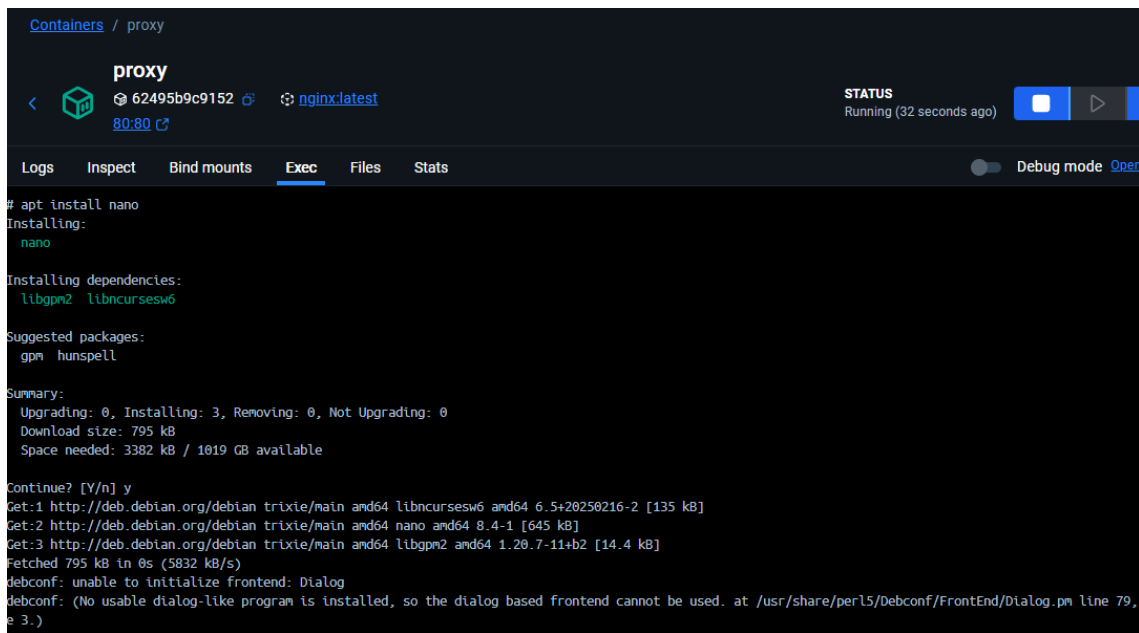
4.- Tras esto lanzaremos un contenedor de nginx, en un puerto que tengamos disponible, este puerto sí que es importante ya que es nuestro servidor proxy.

```
C:\Users\guillermosr26\web>docker run --name proxy -p 80:80 -d nginx
```

Tras lanzarlo debemos ir a la consola del docker y escribir los archivos con nano, si no tenemos este comando debemos instalarlo. Primero haremos un apt update.

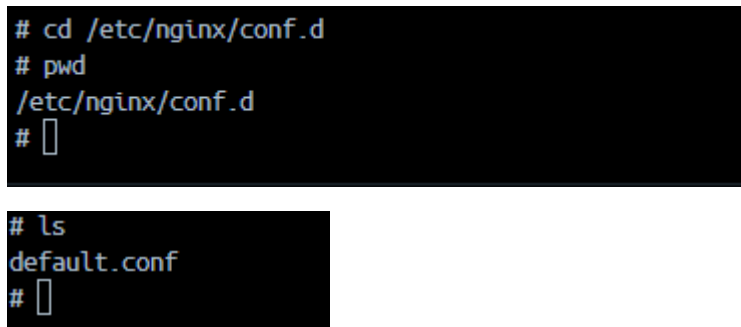
```
# nano
/bin/sh: 1: nano: not found
# apt update
Get:1 http://deb.debian.org/debian trixie InRelease [140 kB]
Get:2 http://deb.debian.org/debian trixie-updates InRelease [47.3 kB]
Get:3 http://deb.debian.org/debian-security trixie-security InRelease [43.4 kB]
Get:4 http://deb.debian.org/debian trixie/main amd64 Packages [9670 kB]
Get:5 http://deb.debian.org/debian trixie-updates/main amd64 Packages [5412 B]
Get:6 http://deb.debian.org/debian-security trixie-security/main amd64 Packages [94.5 kB]
Fetched 10.0 MB in 1s (7416 kB/s)
All packages are up to date.
#
```

Ahora ejecutamos apt install nano y ya nos dejará crear archivos con este comando.

A screenshot of a Docker container terminal window titled 'proxy'. The container ID is 62495b9c9152 and it is running the 'nginx:latest' image. The terminal shows the command '# apt install nano' being executed. The output indicates that nano is being installed along with its dependencies, libgpm2 and libncursesw6. The summary shows that 3 packages will be installed, requiring 795 kB of download size and 3382 kB of space. The installation proceeds successfully, and the terminal shows the fetched data and a message from debconf stating that the dialog-based frontend cannot be used because no usable dialog-like program is installed. The terminal interface includes tabs for Logs, Inspect, Bind mounts, Exec (active), Files, and Stats, and a 'Debug mode' toggle at the top right.

```
Containers / proxy
proxy
62495b9c9152 nginx:latest
80:80
STATUS
Running (32 seconds ago)
Logs Inspect Bind mounts Exec Files Stats
Debug mode
# apt install nano
Installing:
 nano
Installing dependencies:
 libgpm2 libncursesw6
Suggested packages:
 gpm hunspell
Summary:
 Upgrading: 0, Installing: 3, Removing: 0, Not Upgrading: 0
 Download size: 795 kB
 Space needed: 3382 kB / 1019 GB available
Continue? [Y/n] y
Get:1 http://deb.debian.org/debian trixie/main amd64 libncursesw6 amd64 6.5+20250216-2 [135 kB]
Get:2 http://deb.debian.org/debian trixie/main amd64 nano amd64 8.4-1 [645 kB]
Get:3 http://deb.debian.org/debian trixie/main amd64 libgpm2 amd64 1.20.7-11+b2 [14.4 kB]
Fetched 795 kB in 0s (5832 kB/s)
debconf: unable to initialize frontend: Dialog
debconf: (No usable dialog-like program is installed, so the dialog based frontend cannot be used. at /usr/share/perl5/Debconf/FrontEnd/Dialog.pm line 79,
e 3.)
```

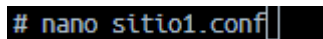
Nos situamos en la carpeta de conf.d de nginx y crearemos los archivos, con un ls vemos que nos encontramos donde está el default.conf.

Two terminal screenshots. The first shows the user navigating to the directory /etc/nginx/conf.d using 'cd' and confirming the current directory with 'pwd'. The second screenshot shows the user running 'ls' in that directory, which lists 'default.conf' as the only file.

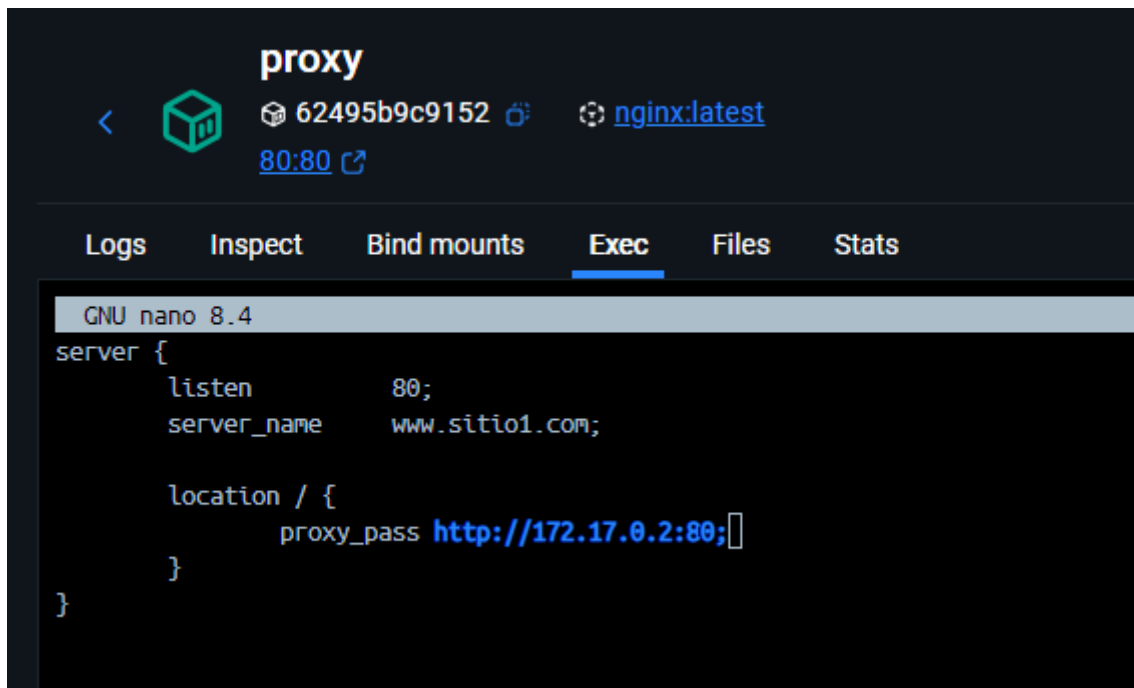
```
# cd /etc/nginx/conf.d
# pwd
/etc/nginx/conf.d
#

# ls
default.conf
#
```

Para acabar creamos el archivo sitio1.conf con el nano y pondremos lo siguiente.

A terminal screenshot showing the command '# nano sitio1.conf' being entered, with the cursor at the end of the line.

```
# nano sitio1.conf
```



```
proxy
62495b9c9152 nginx:latest
80:80

Logs Inspect Bind mounts Exec Files Stats

GNU nano 8.4
server {
    listen      80;
    server_name www.sitio1.com;

    location / {
        proxy_pass http://172.17.0.2:80;
    }
}
```

Aquí proxy\_pass es la ip de nuestro contenedor c1, en este caso 127.17.0.2.

En el contenedor c2 haremos lo mismo, la ip en este caso es 127.17.0.3.



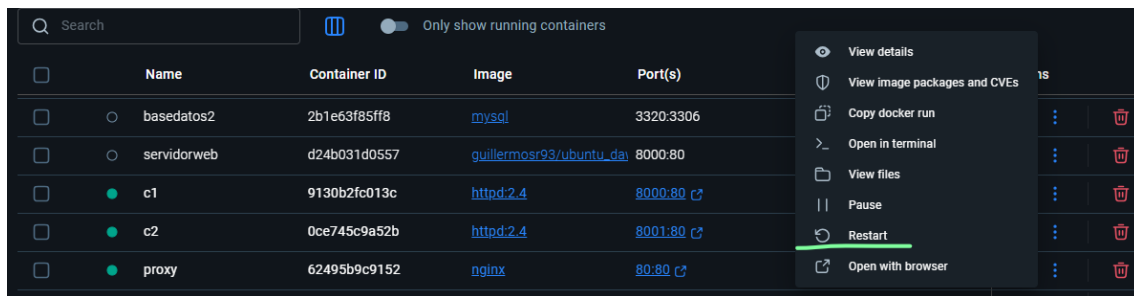
```
proxy
62495b9c9152 nginx:latest
80:80

Logs Inspect Bind mounts Exec Files Stats

GNU nano 8.4
server {
    listen      80;
    server_name www.sitio2.com;

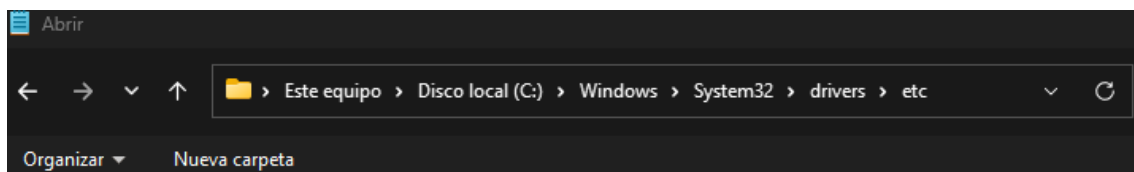
    location / {
        proxy_pass http://172.17.0.3:80;
    }
}
```

Para que funcione debemos de reiniciar nuestro contenedor de nginx.

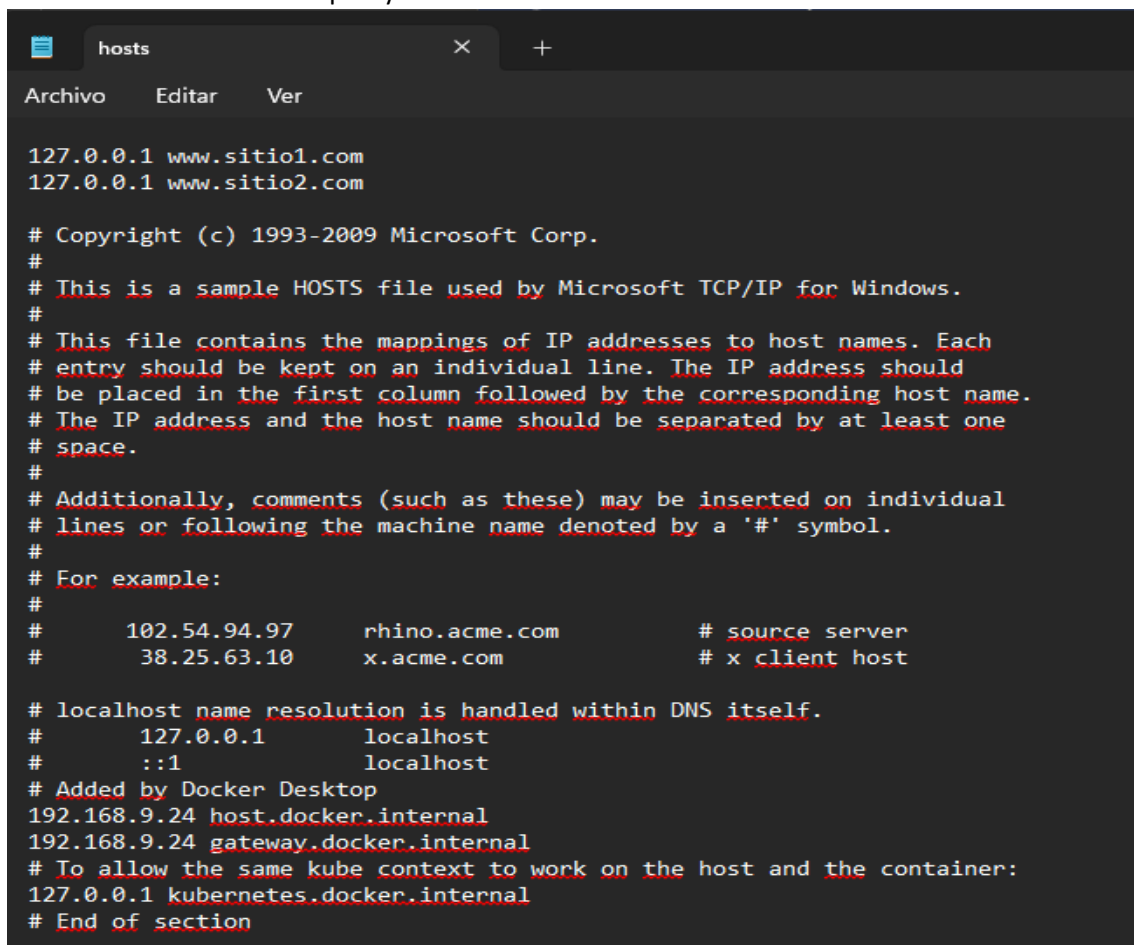


5.- Ahora debemos de configurar nuestro archivo hosts para que se reconozca la ip del proxy.

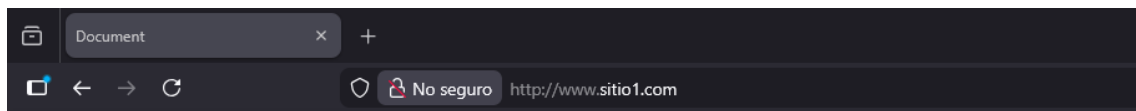
La ruta del archivo seria la siguiente:



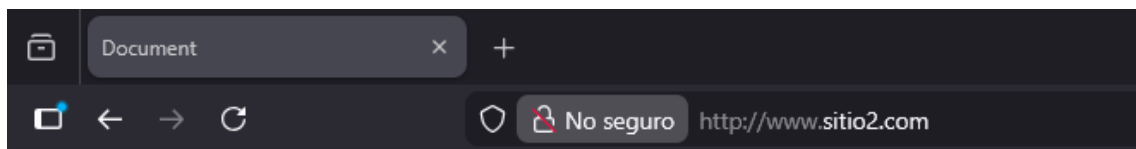
Ahora configuramos el archivo hosts para asociar los dominios www.sitio1.com y www.sitio2.com a la IP del proxy.



Como podemos comprobar, ya funcionan las URLs.



## Bienvenido al sitio web del contenedor c1



## Bienvenido al sitio web del contenedor c2

6.- Finalmente iremos a nuestro visual, lo abriremos en el archivo index.html que habíamos copiado anteriormente, y en las opciones de la izquierda, si tenemos la extensión instalada de docker, nos aparecerá un contenedor, si le damos clic, nos aparecerán los contenedores donde hemos copiado ese index.html.

Si vamos a la carpeta de destino que nos decía la documentación, en este caso `usr/local/apache2/htdocs`, podremos editar nuestro index.html.

