

# **Índice**

<b>Práctica 1</b>	<b>Página 2</b>
<b>Práctica 2</b>	<b>Página 8</b>
<b>Práctica 3</b>	<b>Página 16</b>
<b>Práctica 4</b>	<b>Página 18</b>
<b>Práctica 5</b>	<b>Página 21</b>

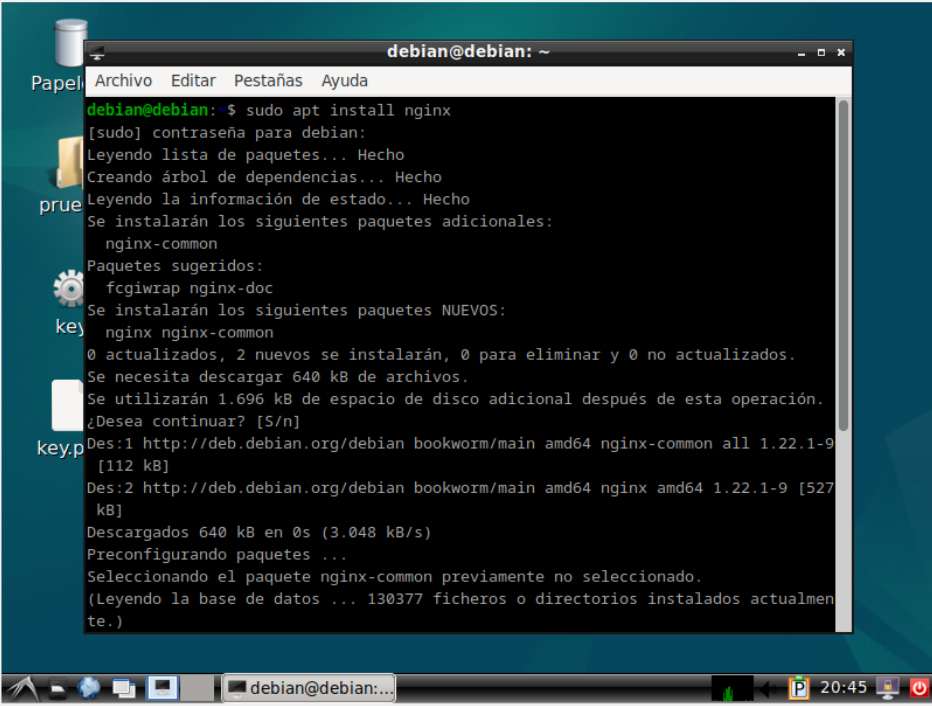
# PRÁCTICA 1

## Instalación de Nginx y configuración de una página

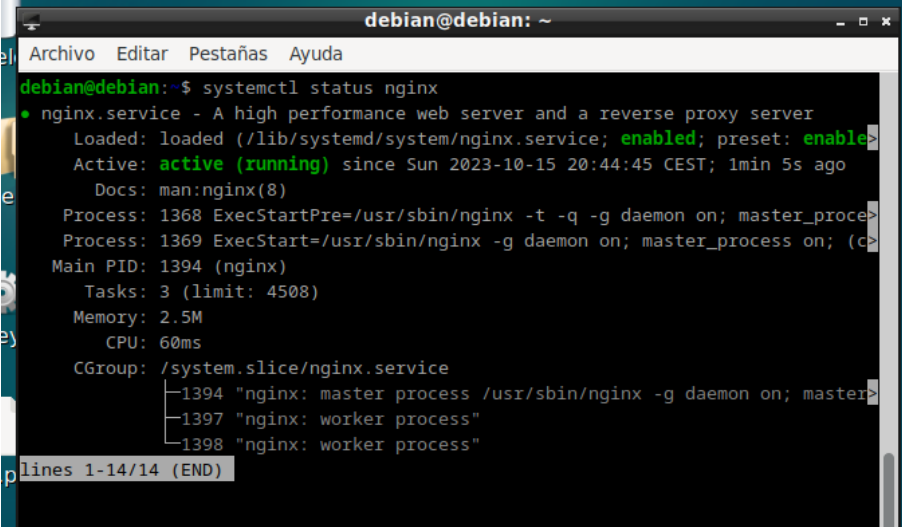
En este documento se describen los pasos a seguir para realizar la instalación del servidor web Nginx y realizar la configuración de una primera página web estática. Debes seguir los siguientes pasos y realizar las capturas que se pidan. Una vez se hayan hecho todas las prácticas de la Unidad 2, deberás crear un documento con todas las capturas pedidas y las explicaciones de las distintas configuraciones que hemos realizado con Nginx.

1. En este primer paso vamos a realizar la instalación y comprobación de Nginx:

- *sudo apt install nginx*
- *systemctl status nginx*



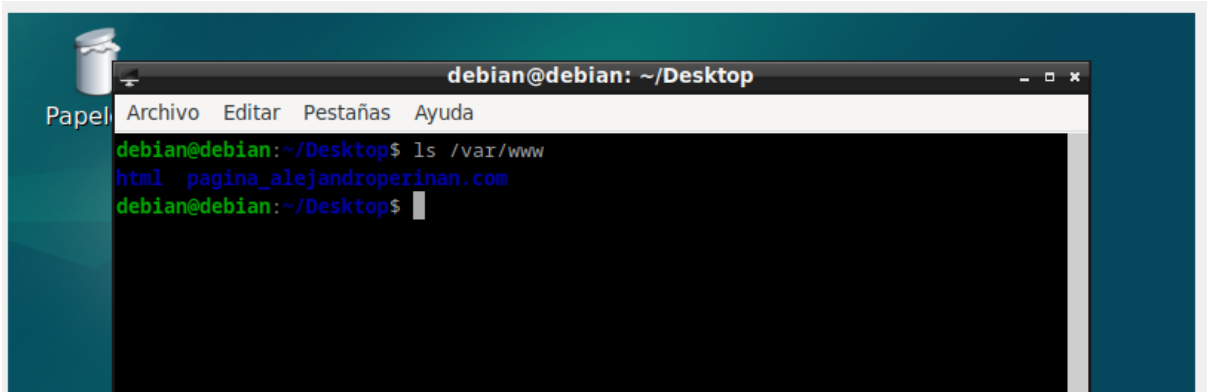
```
debian@debian: ~  
Papel Archivo Editar Pestañas Ayuda  
debian@debian:~$ sudo apt install nginx  
[sudo] contraseña para debian:  
Leyendo lista de paquetes... Hecho  
Creando árbol de dependencias... Hecho  
Leyendo la información de estado... Hecho  
Se instalarán los siguientes paquetes adicionales:  
  nginx-common  
Paquetes sugeridos:  
  fcgiwrap nginx-doc  
Se instalarán los siguientes paquetes NUEVOS:  
  nginx nginx-common  
0 actualizados, 2 nuevos se instalarán, 0 para eliminar y 0 no actualizados.  
Se necesita descargar 640 kB de archivos.  
Se utilizarán 1.696 kB de espacio de disco adicional después de esta operación.  
¿Desea continuar? [S/n]  
Des:1 http://deb.debian.org/debian bookworm/main amd64 nginx-common all 1.22.1-9 [112 kB]  
Des:2 http://deb.debian.org/debian bookworm/main amd64 nginx amd64 1.22.1-9 [527 kB]  
Descargados 640 kB en 0s (3.048 kB/s)  
Preconfigurando paquetes ...  
Seleccionando el paquete nginx-common previamente no seleccionado.  
(Leyendo la base de datos ... 130377 ficheros o directorios instalados actualmente.)
```



```
debian@debian: ~  
el Archivo Editar Pestañas Ayuda  
debian@debian:~$ systemctl status nginx  
• nginx.service - A high performance web server and a reverse proxy server  
Loaded: loaded (/lib/systemd/system/nginx.service; enabled; preset: enable  
Active: active (running) since Sun 2023-10-15 20:44:45 CEST; 1min 5s ago  
Docs: man:nginx(8)  
Process: 1368 ExecStartPre=/usr/sbin/nginx -t -q -g daemon on; master_proce  
Process: 1369 ExecStart=/usr/sbin/nginx -g daemon on; master_process on; (c  
Main PID: 1394 (nginx)  
Tasks: 3 (limit: 4508)  
Memory: 2.5M  
CPU: 60ms  
CGroup: /system.slice/nginx.service  
├─1394 "nginx: master process /usr/sbin/nginx -g daemon on; master  
├─1397 "nginx: worker process"  
└─1398 "nginx: worker process"  
lines 1-14/14 (END)
```

2. Vamos a crear la carpeta de nuestro sitio web:
- `sudo mkdir -p /var/www/pagina_tunombre.com`

Realiza una captura del contenido que hay en el directorio /var/www

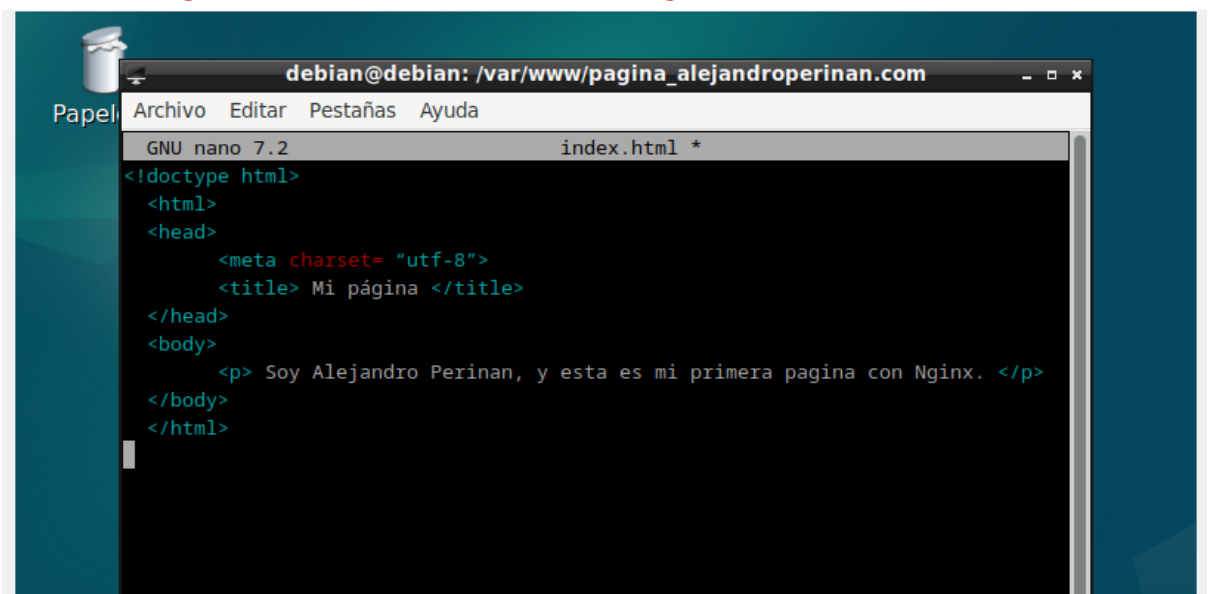


A terminal window titled 'debian@debian: ~/Desktop' showing the command `ls /var/www` and its output: `html pagina_alejandroperinan.com`.

3. Vamos a crear el index de nuestra página en el directorio anterior y editarlo:

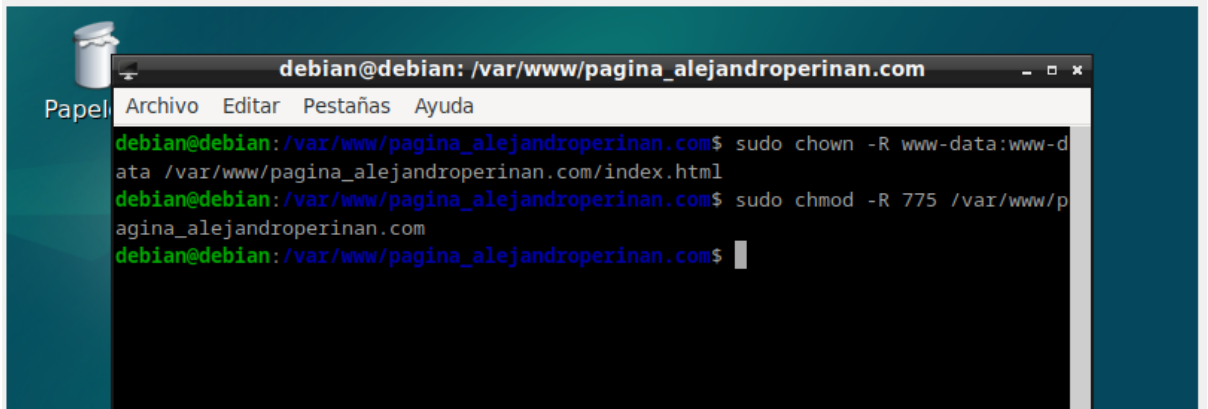
- `sudo nano index.html`
- `<!doctype html>`
- `<html>`
- `<head>`
  - `<meta charset= "utf-8">`
  - `<title> Mi página </title>`
- `</head>`
- `<body>`
  - `<p> Soy tu_nombre, y esta es mi primera página con Nginx. </p>`
- `</body>`
- `</html>`

Realiza una captura del contenido del archivo html que has hecho.



A terminal window titled 'debian@debian: /var/www/pagina\_alejandroperinan.com' showing the content of `index.html` in the nano editor. The content is: `<!doctype html>`, `<html>`, `<head>`, `<meta charset= "utf-8">`, `<title> Mi página </title>`, `</head>`, `<body>`, `<p> Soy Alejandro Perinan, y esta es mi primera pagina con Nginx. </p>`, `</body>`, and `</html>`.

4. Damos permisos para que no haya problemas al intentar acceder a la página anterior:
- `sudo chown -R www-data:www-data /var/www/pagina_tunombre.com/index.html`
  - `sudo chmod -R 755 /var/www/pagina_tunombre.com`



The screenshot shows a terminal window titled 'debian@debian: /var/www/pagina\_alejandroperinan.com'. The window has a menu bar with 'Papel', 'Archivo', 'Editar', 'Pestañas', and 'Ayuda'. The terminal content shows the following commands and their outputs:

```
debian@debian:/var/www/pagina_alejandroperinan.com$ sudo chown -R www-data:www-data /var/www/pagina_alejandroperinan.com/index.html
debian@debian:/var/www/pagina_alejandroperinan.com$ sudo chmod -R 755 /var/www/pagina_alejandroperinan.com
debian@debian:/var/www/pagina_alejandroperinan.com$
```

5. Creamos y configuramos el archivo para nuestro sitio en el directorio sites-available:
- `sudo nano pagina_tunombre.com`
  - `server {`

```
listen 82;
listen [::]:82;

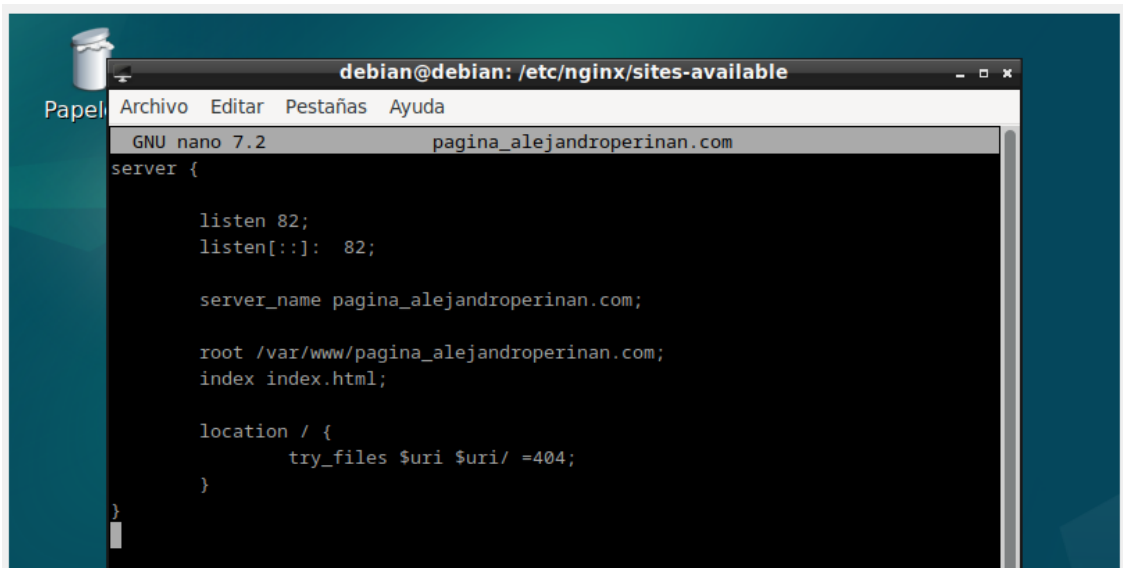
server_name pagina_tunombre.com;

root /var/www/pagina_tunombre.com;
index index.html;
```

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

}
```

Realiza una captura del contenido del archivo html que acabas de configurar.



```
debian@debian: /etc/nginx/sites-available
GNU nano 7.2 pagina_alejandroperinan.com
server {

    listen 82;
    listen[::]: 82;

    server_name pagina_alejandroperinan.com;

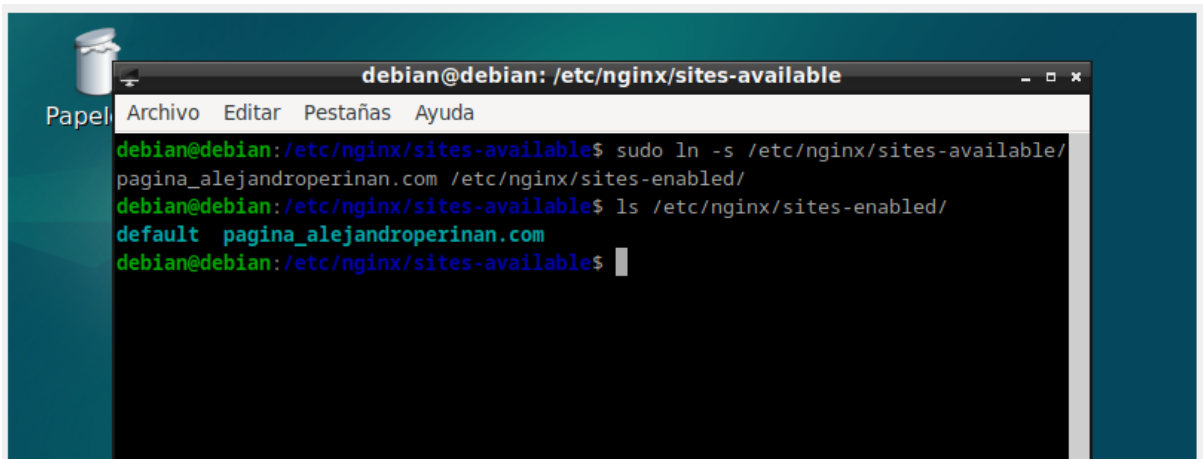
    root /var/www/pagina_alejandroperinan.com;
    index index.html;

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

}
```

6. Ahora tenemos que vincular este archivo con el correspondiente en el directorio sites-enabled:

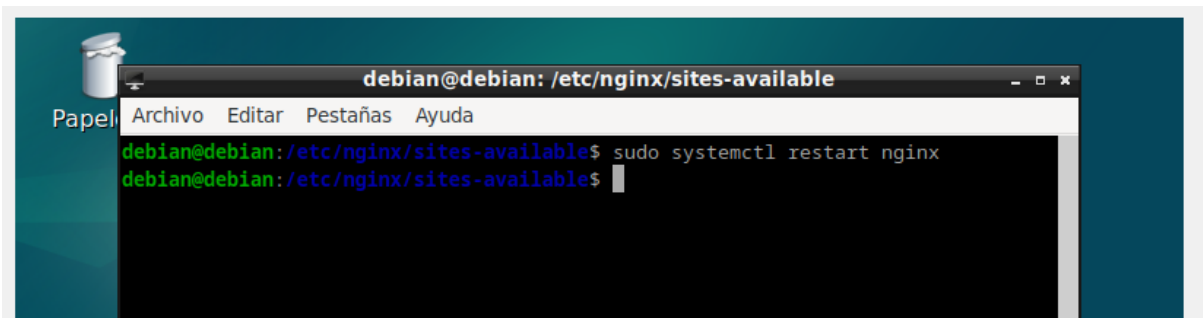
- `sudo ln -s /etc/nginx/sites-available/pagina_tunombre.com /etc/nginx/sites-enabled/`  
Haz una captura donde se muestre el aspecto de los archivos del directorio sites-enabled, y comprueba que se ha generado el archivo correspondiente.



```
debian@debian: /etc/nginx/sites-available
debian@debian:/etc/nginx/sites-available$ sudo ln -s /etc/nginx/sites-available/
pagina_alejandroperinan.com /etc/nginx/sites-enabled/
debian@debian:/etc/nginx/sites-available$ ls /etc/nginx/sites-enabled/
default pagina_alejandroperinan.com
debian@debian:/etc/nginx/sites-available$
```

7. Hecho todo esto, debemos reiniciar Nginx:

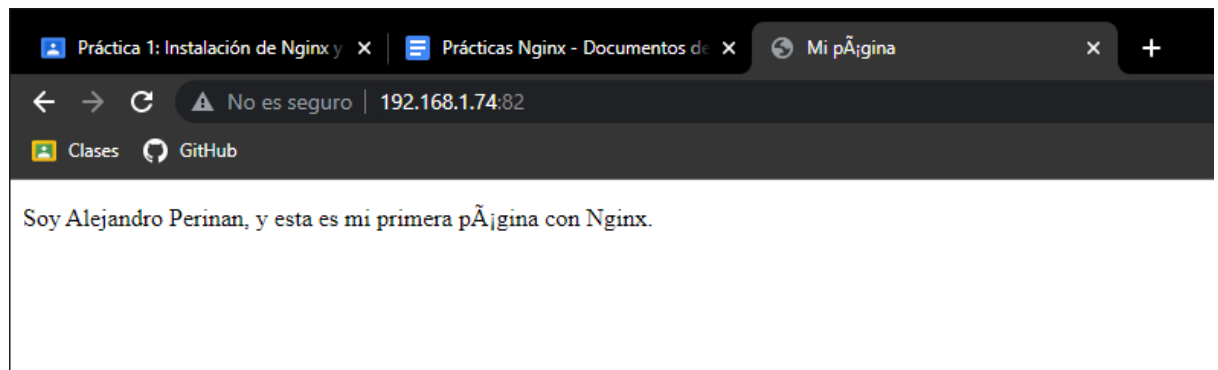
- `sudo systemctl restart nginx`



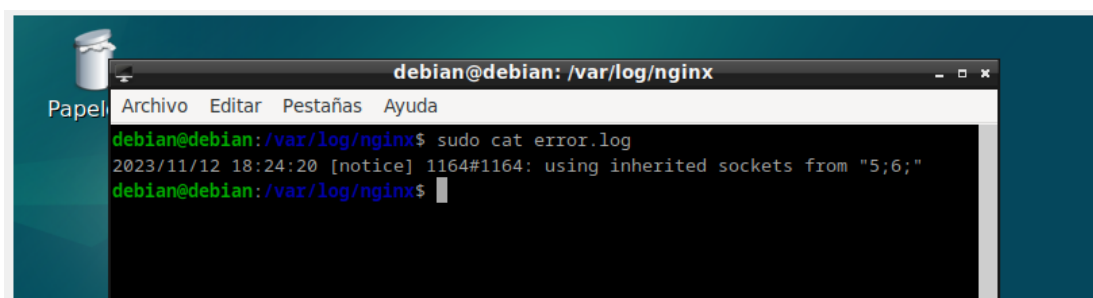
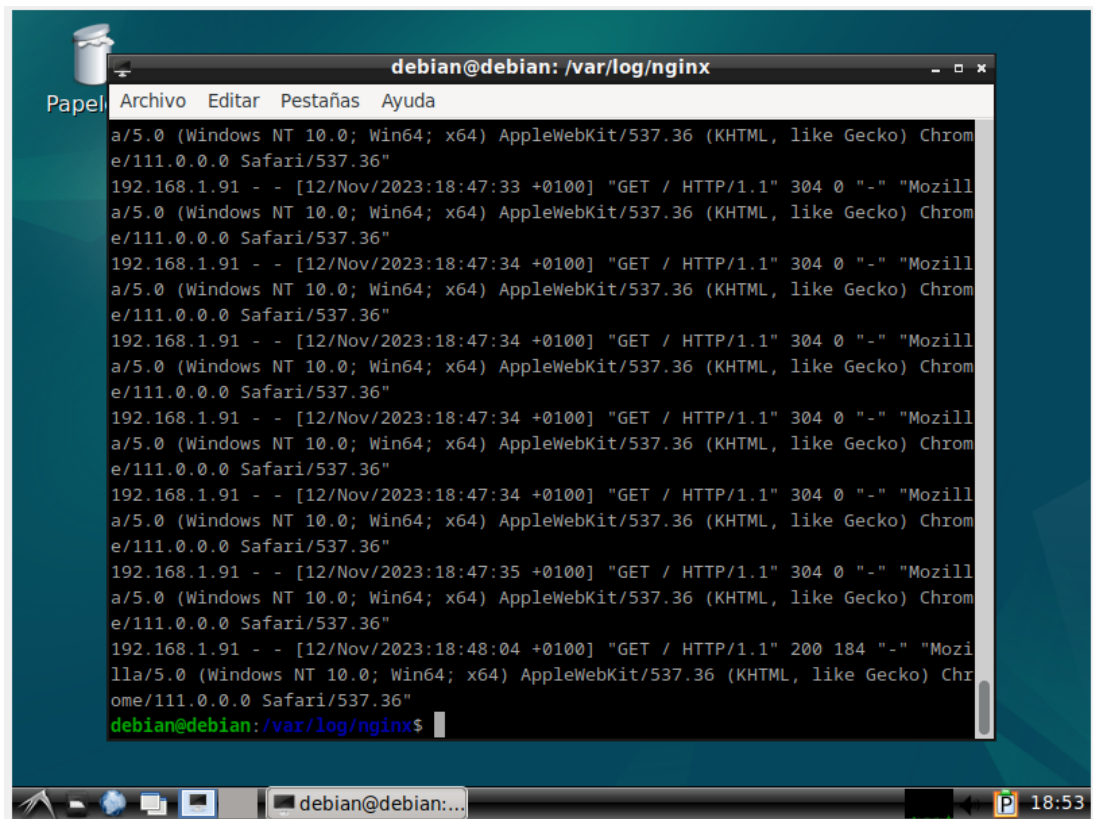
```
debian@debian: /etc/nginx/sites-available
debian@debian:/etc/nginx/sites-available$ sudo systemctl restart nginx
debian@debian:/etc/nginx/sites-available$
```

8. Comprueba que se está sirviendo la página que acabas de hacer desde un navegador del cliente, accediendo a la página mediante la IP del servidor.

Haz una captura de la página.



9. Haz una captura del contenido de los archivos de logs, tanto del de access como del de error.



Puede ser que al reiniciar Nginx os aparezcan una serie de errores. En caso de no poder solventarlos, hay que realizar una nueva instalación. Para ellos desinstalamos Nginx con los siguientes comandos y volveremos al paso 1 del documento:

- `sudo apt --purge remove nginx`
- `sudo apt --purge remove nginx-common`
- `sudo apt --purge remove nginx-core`

A veces, el error se deberá simplemente a algún fallo de sintaxis. Los archivos de configuración son muy sensibles a espacios, sangrías, etc. Con el comando de abajo, podremos averiguar rápidamente si tenemos algún error de este tipo:

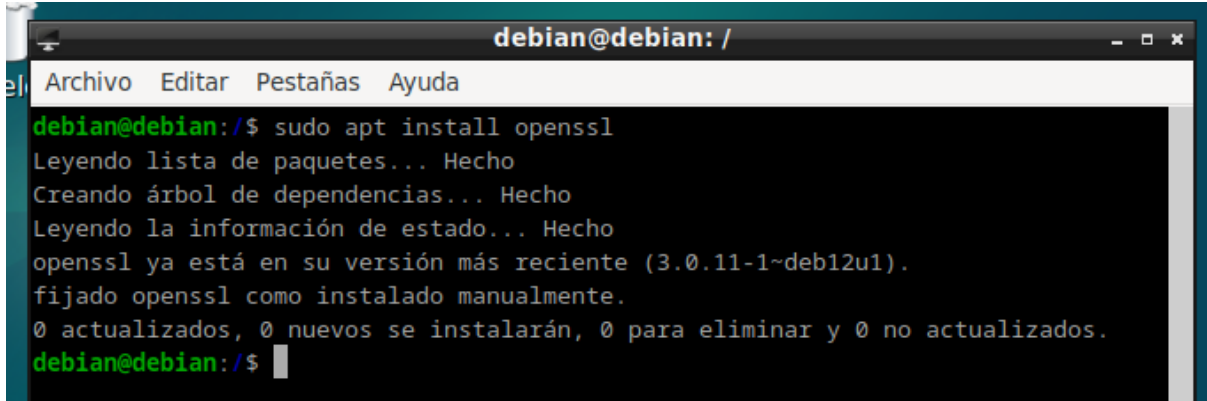
- `sudo nginx -t`

## PRÁCTICA 2

### Autenticación y restricciones por IP con Nginx

En esta práctica debemos instalar openssl, pues es la herramienta que vamos a emplear para la creación de contraseñas.

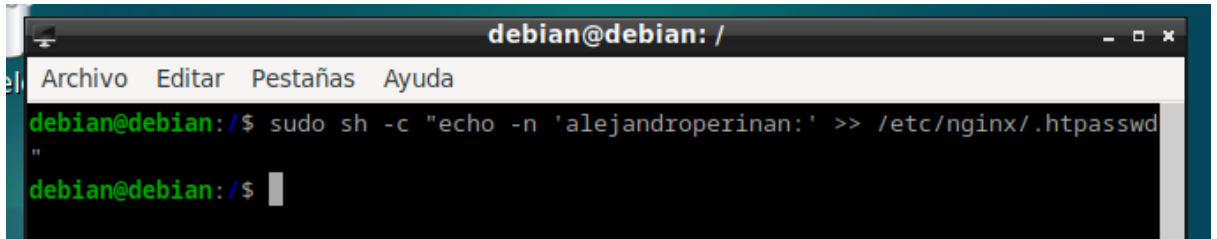
Es necesario instalar openssl: *sudo apt install openssl*



```
debian@debian: /
Archivo Editar Pestañas Ayuda
debian@debian:/$ sudo apt install openssl
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias... Hecho
Leyendo la información de estado... Hecho
openssl ya está en su versión más reciente (3.0.11-1~deb12u1).
fijado openssl como instalado manualmente.
0 actualizados, 0 nuevos se instalarán, 0 para eliminar y 0 no actualizados.
debian@debian:/$
```

1. Creación de un archivo “.htpasswd” donde vamos a almacenar los usuarios junto con sus contraseñas:

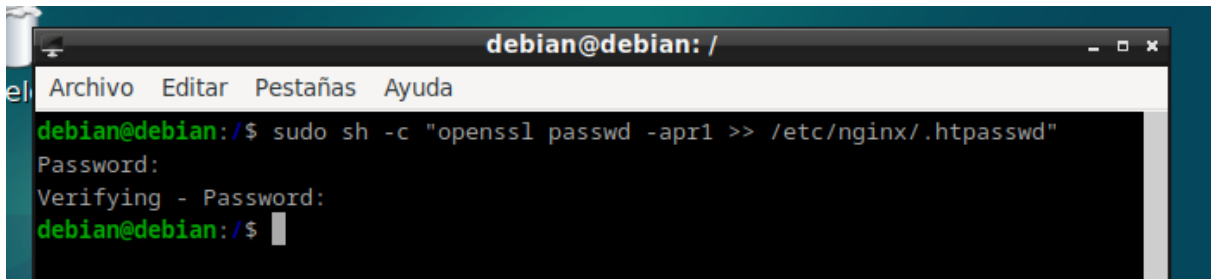
- *sudo sh -c "echo -n 'tu\_nombre:' >> /etc/nginx/.htpasswd"*



```
debian@debian: /
Archivo Editar Pestañas Ayuda
debian@debian:/$ sudo sh -c "echo -n 'alejandropereira:' >> /etc/nginx/.htpasswd"
debian@debian:/$
```

2. Creación de la contraseña correspondiente al usuario anterior:

- *sudo sh -c "openssl passwd -apr1 >> /etc/nginx/.htpasswd"*

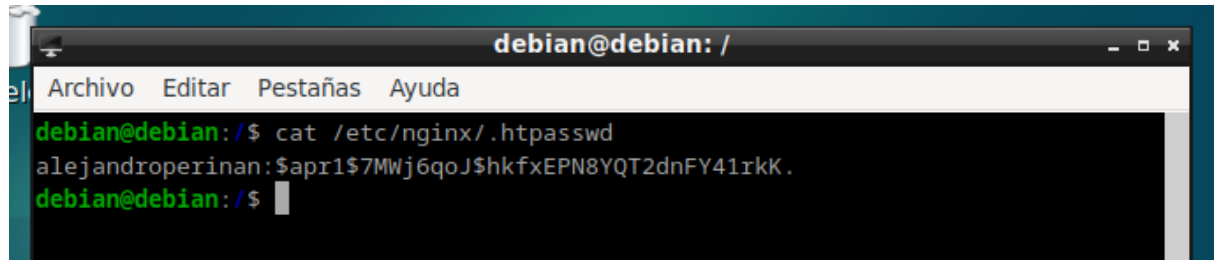


```
debian@debian: /
Archivo Editar Pestañas Ayuda
debian@debian:/$ sudo sh -c "openssl passwd -apr1 >> /etc/nginx/.htpasswd"
Password:
Verifying - Password:
debian@debian:/$
```



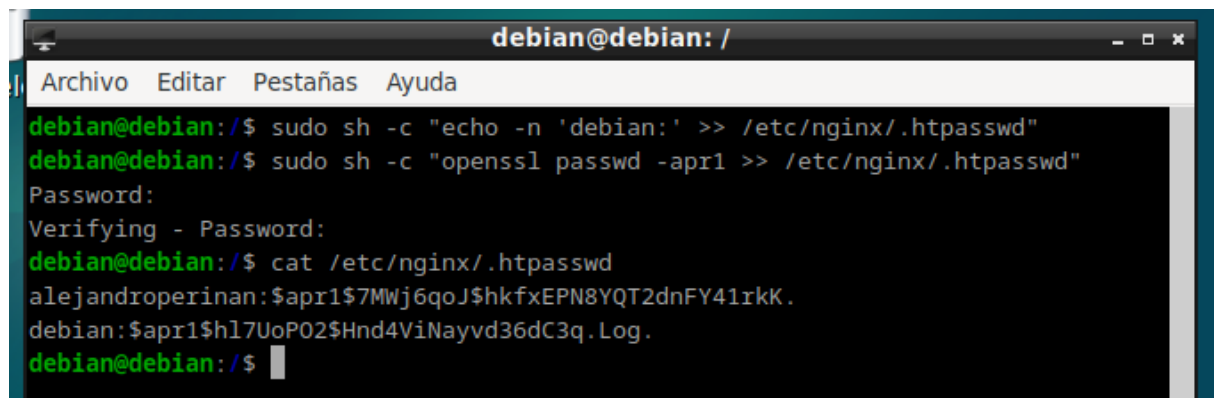
3. Vamos a comprobar que aparece el nombre del usuario anterior junto con su contraseña cifrada cifrados en el fichero `.htpasswd`

- `cat /etc/nginx/.htpasswd`



```
debian@debian: /
Archivo Editar Pestañas Ayuda
debian@debian:/$ cat /etc/nginx/.htpasswd
alejandropereiran:$apr1$7MWj6qoJ$hkfxEPN8YQT2dnFY41rkK.
debian@debian:/$
```

4. Añade un usuario más con su correspondiente contraseña y haz una captura con el contenido del archivo `.htpasswd` visto en la consola.



```
debian@debian: /
Archivo Editar Pestañas Ayuda
debian@debian:/$ sudo sh -c "echo -n 'debian:' >> /etc/nginx/.htpasswd"
debian@debian:/$ sudo sh -c "openssl passwd -apr1 >> /etc/nginx/.htpasswd"
Password:
Verifying - Password:
debian@debian:/$ cat /etc/nginx/.htpasswd
alejandropereiran:$apr1$7MWj6qoJ$hkfxEPN8YQT2dnFY41rkK.
debian:$apr1$h17UoP02$Hnd4ViNayvd36dC3q. Log.
debian@debian:/$
```

5. Ahora debemos modificar la configuración de nuestro archivo localizado en el directorio `sites-available` para que tenga en cuenta los credenciales anteriores:

- `sudo nano /etc/nginx/sites-available/pagina_tunombre.com`

Hay que añadir las siguientes líneas dentro del área de `location`:

- `auth_basic "Credenciales";`

- `auth_basic_user_file /etc/nginx/.htpasswd;`

Haz una captura donde se muestre el nuevo aspecto del archivo `pagina_tunombre.com` y explica cuál es la función de estas 2 nuevas líneas.



```
debian@debian: /
Archivo Editar Pestañas Ayuda
GNU nano 7.2 /etc/nginx/sites-available/pagina_alejandropereiran.com
server {

    listen 82;
    listen [::]:82;

    server_name pagina_alejandropereiran.com;

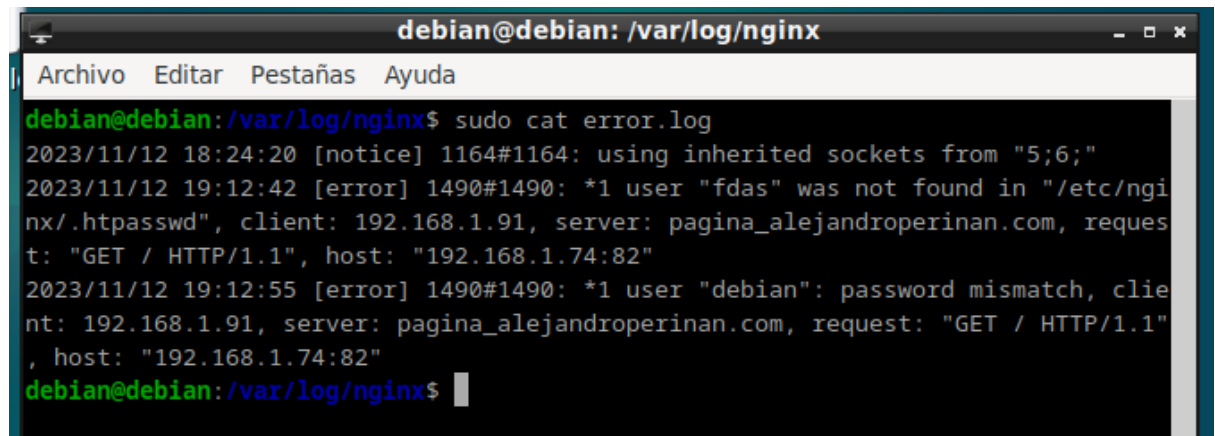
    root /var/www/pagina_alejandropereiran.com;
    index index.html;

    location / {
        try_files $uri $uri/ =404;
        auth_basic "Credenciales";
        auth_basic_user_file /etc/nginx/.htpasswd;
    }

}
```

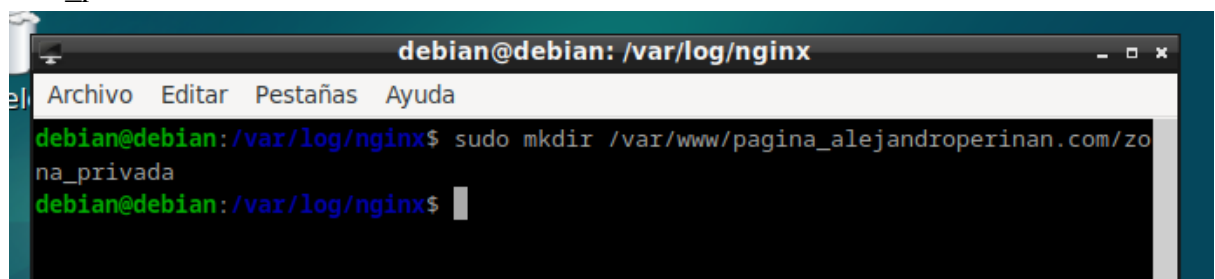
6. Reiniciamos Nginx para que se aplique la nueva configuración. Intenta acceder a la web desde el cliente. Hazlo utilizando un usuario correcto y otro incorrecto. Di los errores que aparecen y revisa el fichero de logs. Intenta acceder cancelando la autenticación. ¿Qué mensaje de error aparece? ¿Qué sucede cuando intentas acceder a la misma página por segunda vez? **Describe esto en la memoria de la práctica.**

Al entrar, se solicita al usuario que introduzca sus datos, si los datos no se encuentran en el archivo configurado anteriormente se recarga el procedimiento, en caso contrario extremos con éxito en la página.

A terminal window titled 'debian@debian: /var/log/nginx' showing the output of 'sudo cat error.log'. The logs show a successful connection followed by two failed login attempts. The first attempt for user 'fdas' failed because the user was not found in the password file. The second attempt for user 'debian' failed due to a password mismatch. The terminal text is as follows:

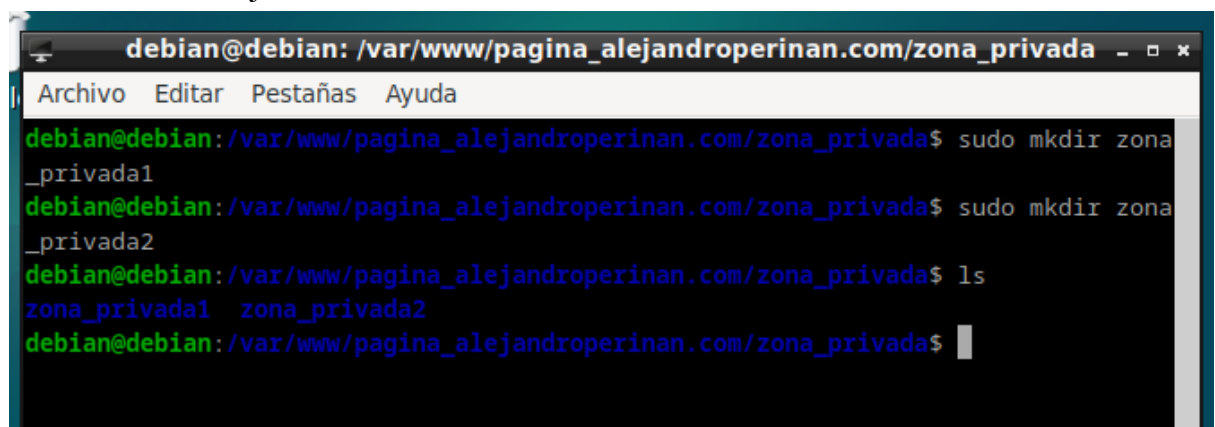
```
debian@debian:/var/log/nginx$ sudo cat error.log
2023/11/12 18:24:20 [notice] 1164#1164: using inherited sockets from "5;6;"
2023/11/12 19:12:42 [error] 1490#1490: *1 user "fdas" was not found in "/etc/nginx/.htpasswd", client: 192.168.1.91, server: pagina_alejandroperinan.com, request: "GET / HTTP/1.1", host: "192.168.1.74:82"
2023/11/12 19:12:55 [error] 1490#1490: *1 user "debian": password mismatch, client: 192.168.1.91, server: pagina_alejandroperinan.com, request: "GET / HTTP/1.1", host: "192.168.1.74:82"
debian@debian:/var/log/nginx$
```

Hasta ahora estamos aplicando autenticación a toda nuestra página. Vamos a ver cómo hacerlo solo a ciertas partes. Para ello, dentro del directorio en el que estamos incluyendo los distintos archivos de nuestra página (/var/www/pagina\_tunombre.com), debemos crear otro directorio. Por ejemplo, zona\_privada.

A terminal window titled 'debian@debian: /var/log/nginx' showing the command 'sudo mkdir /var/www/pagina\_alejandroperinan.com/zona\_privada' being executed successfully. The terminal text is as follows:

```
debian@debian:/var/log/nginx$ sudo mkdir /var/www/pagina_alejandroperinan.com/zona_privada
debian@debian:/var/log/nginx$
```

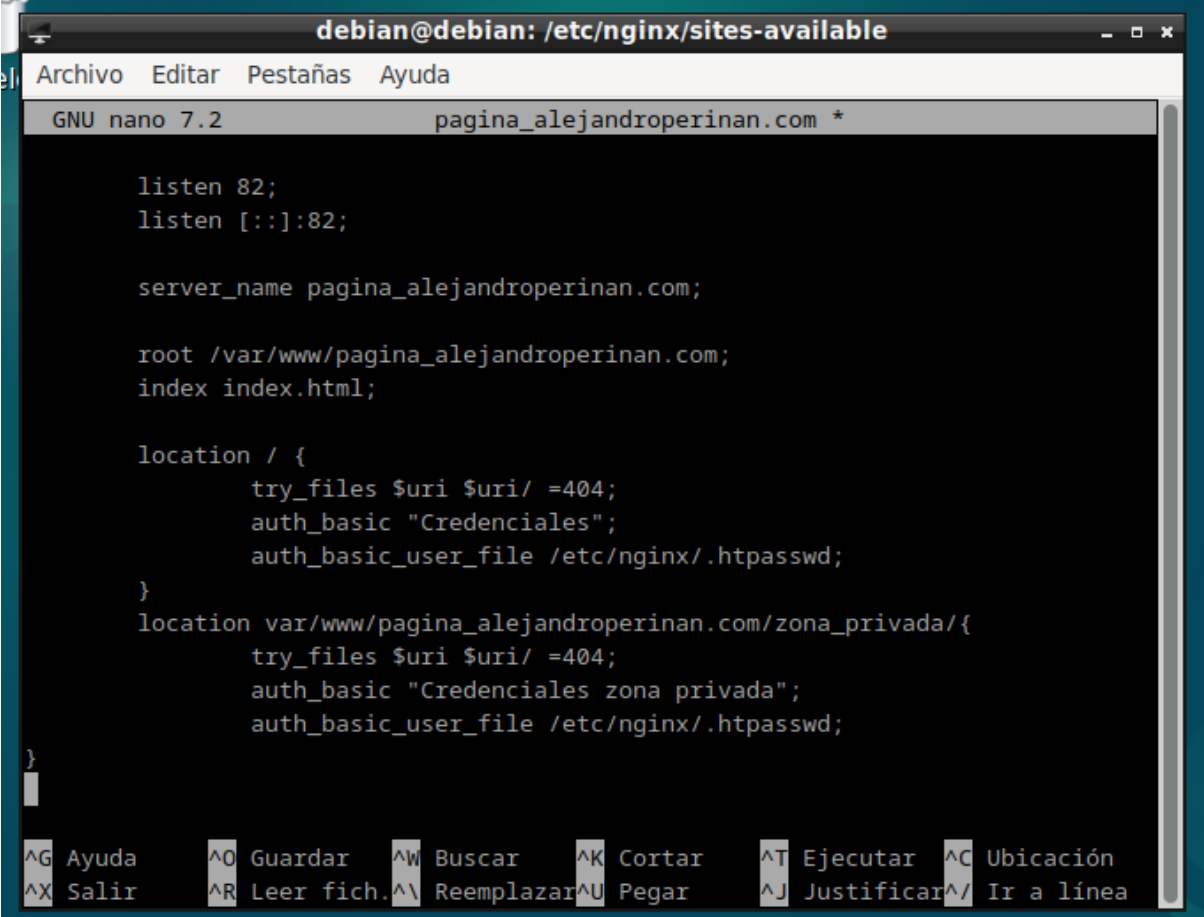
7. Dentro de este nuevo directorio, crea dos nuevas páginas html. Por ejemplo: zona\_privada1 y zona\_privada2. Configura cada una para que al acceder a ellas muestren un mensaje diferente.

A terminal window titled 'debian@debian: /var/www/pagina\_alejandroperinan.com/zona\_privada' showing the creation of two subdirectories, 'zona\_privada1' and 'zona\_privada2', using 'sudo mkdir'. The 'ls' command is then used to verify their creation. The terminal text is as follows:

```
debian@debian:/var/www/pagina_alejandroperinan.com/zona_privada$ sudo mkdir zona_privada1
debian@debian:/var/www/pagina_alejandroperinan.com/zona_privada$ sudo mkdir zona_privada2
debian@debian:/var/www/pagina_alejandroperinan.com/zona_privada$ ls
zona_privada1 zona_privada2
debian@debian:/var/www/pagina_alejandroperinan.com/zona_privada$
```

8. Volvemos al archivo de la carpeta sites-available. Y lo vamos a modificar para que solo se realice el acceso previa autenticación en las páginas que acabamos de crear. Debemos crearnos un nuevo bloque location y especificar la ruta que deseamos restringir, en este caso: /zona\_privada/

Realiza una captura donde se muestren los cambios realizados en el archivo de configuración.



```
debian@debian: /etc/nginx/sites-available
GNU nano 7.2 pagina_alejandroperinan.com *

listen 82;
listen [::]:82;

server_name pagina_alejandroperinan.com;

root /var/www/pagina_alejandroperinan.com;
index index.html;

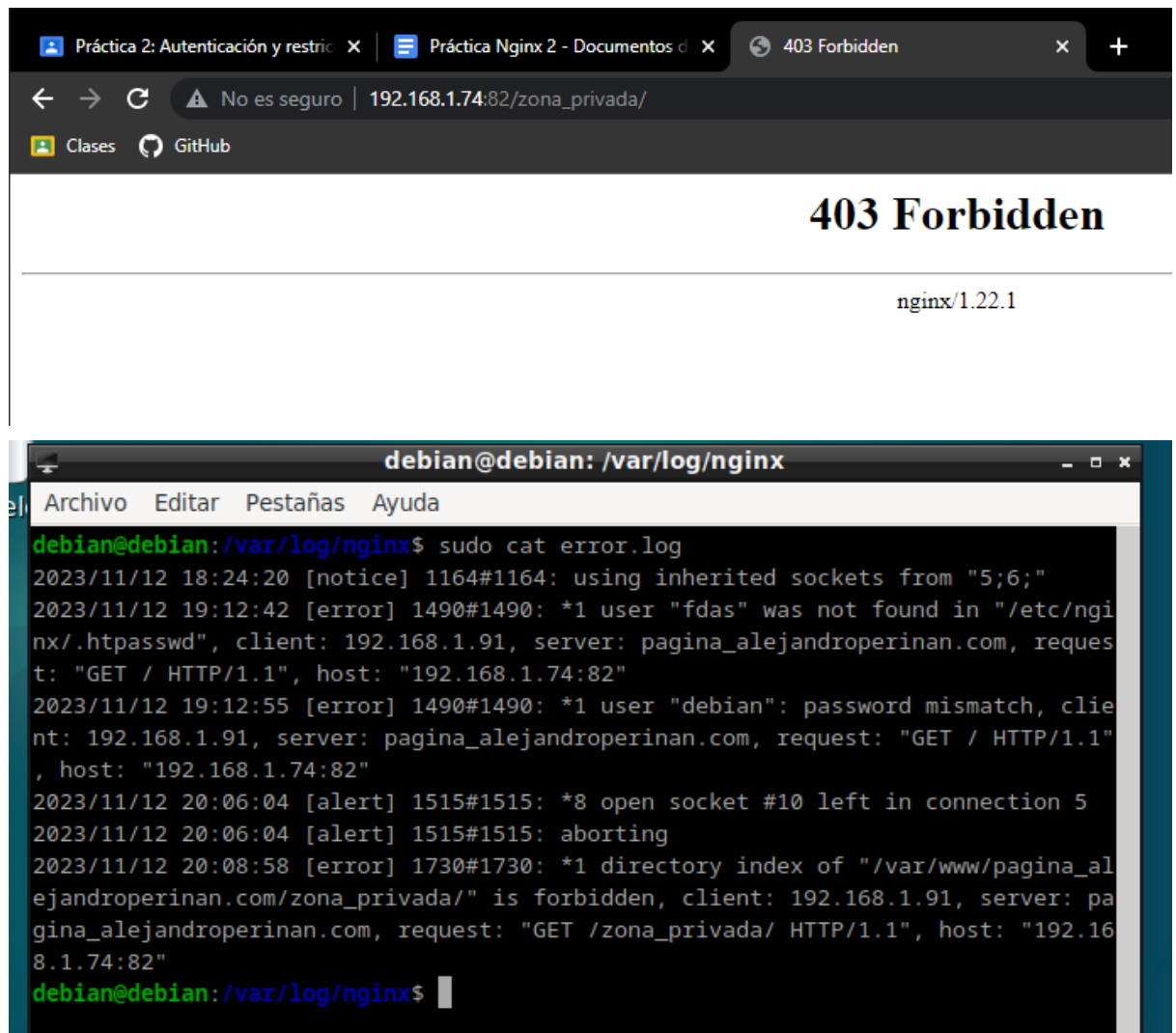
location / {
    try_files $uri $uri/ =404;
    auth_basic "Credenciales";
    auth_basic_user_file /etc/nginx/.htpasswd;
}
location var/www/pagina_alejandroperinan.com/zona_privada/{
    try_files $uri $uri/ =404;
    auth_basic "Credenciales zona privada";
    auth_basic_user_file /etc/nginx/.htpasswd;
}

^G Ayuda ^O Guardar ^W Buscar ^K Cortar ^T Ejecutar ^C Ubicación
^X Salir ^R Leer fich. ^\ Reemplazar ^U Pegar ^J Justificar ^_ Ir a línea
```

Ahora, además de la autenticación mediante usuario y contraseña, vamos a aplicar restricciones de acceso a la página mediante IP.

9. Explica cómo se deben emplear las directivas deny y allow. Configura el archivo localizado en sites available de forma que se deniegue el acceso a la máquina anfitriona (nuestro cliente). ¿Cuál es el error que nos aparece en el navegador? Revisa también el archivo error.log. Incluye estas capturas y explicación en la memoria de la práctica.

Estas directivas permiten controlar qué usuarios o direcciones IP tienen permiso para acceder a recursos específicos en el servidor.



10. ¿En qué se diferencian las directivas `satisfy all` y `satisfy any`? Sigue realizando modificaciones en la configuración del archivo de `sites-available`. En primer lugar, modifícalo para que sea necesaria autenticación o tener una cierta IP para poder acceder a la web. En segundo lugar, modifícalo para que, si queremos acceder a la `zona_privada`, sea necesario tanto tener una IP determinada como estar autenticado. Realiza una captura del archivo de configuración resultante en ambos casos, y explica qué has hecho.

Las directivas `satisfy all` y `satisfy any` son configuraciones utilizadas para especificar los requisitos de autorización que deben cumplirse para permitir el acceso a un recurso.

```
debian@debian: /etc/nginx/sites-available
Archivo  Editar  Pestañas  Ayuda
GNU nano 7.2  pagina_alejandroperinan.com *
server {

    listen 82;
    listen [::]:82;

    server_name pagina_alejandroperinan.com;

    root /var/www/pagina_alejandroperinan.com;
    index index.html;

    location / {
        try_files $uri $uri/ =404;
        satisfy all;
        allow 192.168.1.91;
        deny all;
        auth_basic "Credenciales";
        auth_basic_user_file /etc/nginx/.htpasswd;
    }
    location var/www/pagina_alejandroperinan.com/zona_privada/{
        try_files $uri $uri/ =404;
    }
}

^G Ayuda  ^O Guardar  ^W Buscar  ^K Cortar  ^T Ejecutar  ^C Ubicación
^X Salir  ^R Leer fich. ^\ Reemplazar ^U Pegar  ^J Justificar ^/ Ir a línea
```

```
debian@debian: /etc/nginx/sites-available
Archivo  Editar  Pestañas  Ayuda
GNU nano 7.2  pagina_alejandroperinan.com *
server {

    listen 82;
    listen [::]:82;

    server_name pagina_alejandroperinan.com;

    root /var/www/pagina_alejandroperinan.com;
    index index.html;

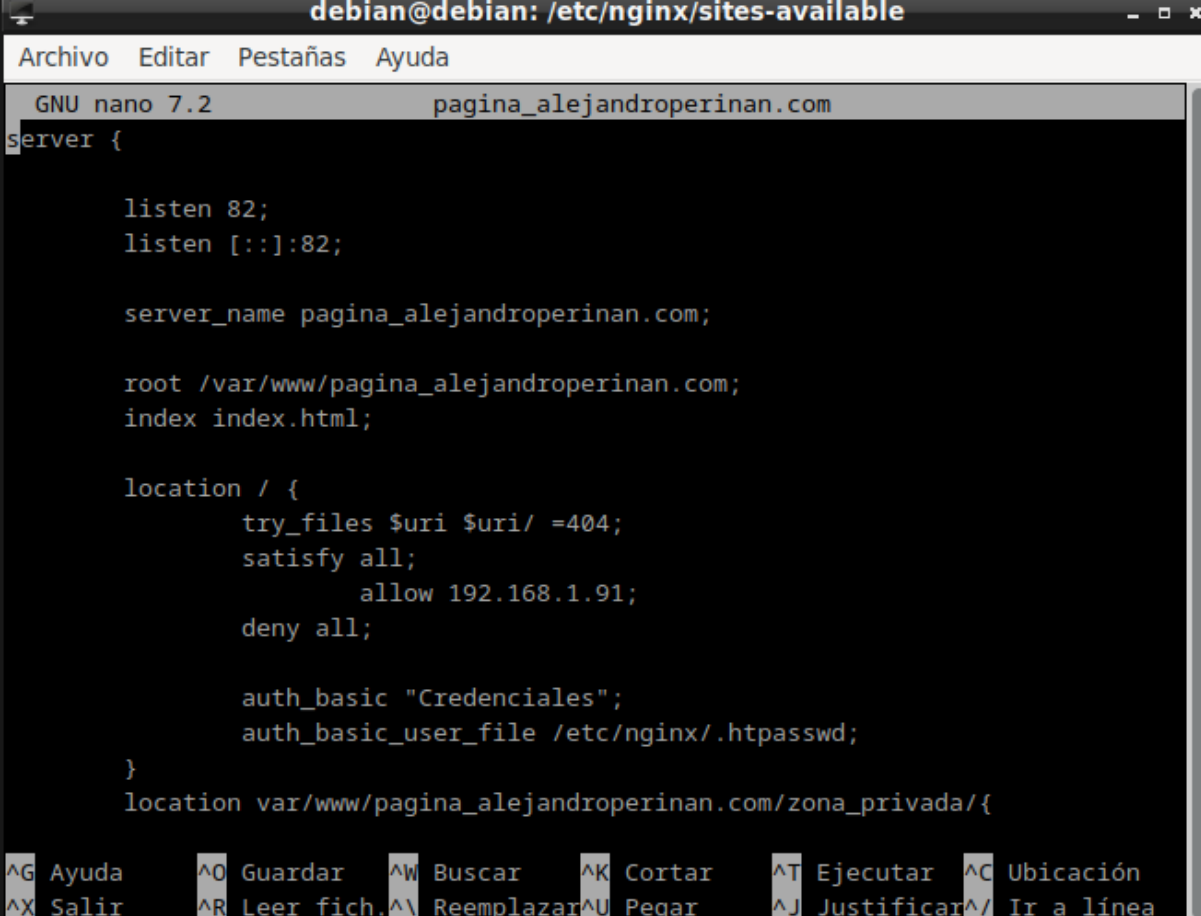
    location / {
        try_files $uri $uri/ =404;
        satisfy all;
        deny all;
        deny 192.168.1.91;
        auth_basic "Credenciales";
        auth_basic_user_file /etc/nginx/.htpasswd;
    }
    location var/www/pagina_alejandroperinan.com/zona_privada/{
        try_files $uri $uri/ =404;
    }
}

^G Ayuda  ^O Guardar  ^W Buscar  ^K Cortar  ^T Ejecutar  ^C Ubicación
^X Salir  ^R Leer fich. ^\ Reemplazar ^U Pegar  ^J Justificar ^/ Ir a línea
```

11. ¿Qué líneas añadirías en caso de que quieras que tu página sea accesible a las direcciones del grupo 192.168.59.1/24 y 192.168.80.1/24, exceptuando la dirección 192.168.59.20? No queremos que ninguna otra dirección IP pueda acceder a nuestro sitio.

```
allow 192.168.59.0/24;  
allow 192.168.80.0/24;  
deny 192.168.59.20;  
deny all;
```

12. Configura el archivo para que puedas acceder con la IP del cliente y además también se necesite realizar la autenticación. **Añade una captura del archivo de configuración.**



The screenshot shows a terminal window titled "debian@debian: /etc/nginx/sites-available". The window contains the nano 7.2 editor editing the file "pagina\_alejandroperinan.com". The configuration is as follows:

```
server {  
  
    listen 82;  
    listen [::]:82;  
  
    server_name pagina_alejandroperinan.com;  
  
    root /var/www/pagina_alejandroperinan.com;  
    index index.html;  
  
    location / {  
        try_files $uri $uri/ =404;  
        satisfy all;  
        allow 192.168.1.91;  
        deny all;  
  
        auth_basic "Credenciales";  
        auth_basic_user_file /etc/nginx/.htpasswd;  
    }  
    location var/www/pagina_alejandroperinan.com/zona_privada/{
```

At the bottom of the terminal, there is a row of keyboard shortcuts for nano:

^G	Ayuda	^O	Guardar	^W	Buscar	^K	Cortar	^T	Ejecutar	^C	Ubicación
^X	Salir	^R	Leer fich.	^\\	Reemplazar	^U	Pegar	^J	Justificar	^/	Ir a línea

13. Imagina que mi IP es 192.168.20.24, y quiero acceder al directorio “series” de una página web que tiene la configuración de la imagen adjunta. ¿Podré acceder a esa sección? ¿Por qué?

```
location /series/ {  
  
    satisfy all;  
    allow 192.168.20.22;  
    deny all;  
    allow 192.168.20.1/24;  
  
    auth_basic "Esta zona está restringida";  
    auth_basic_user_file /.htpasswd;  
  
    try_files $uri $uri/ =404;  
}
```

No podrá, ya que del grupo de IPs 192.168.20.1/24 solo esta admitida la IP 192.168.20.22.

## PRÁCTICA 3

### Configuración de Nginx como Proxy Inverso

En esta práctica queremos configurar Nginx como Proxy Inverso, siguiendo el esquema dibujado en la pizarra. El objetivo es que, a diferencia de en las prácticas anteriores, no podamos acceder a la página web que hemos creado directamente desde el servidor donde se aloja. Lo que pretendemos es hacerlo por medio de un servidor intermedio o Proxy Inverso. A continuación se describen algunos de los pasos a seguir, pero la idea es que esta práctica sea más autónoma e investiguéis las distintas configuraciones por vuestra cuenta. Todo lo que realicéis se debe documentar mediante capturas y explicaciones en la memoria de la práctica.

Habrá que realizar un clon de nuestra máquina actual, a la que llamaremos proxy inverso (por ejemplo). En ella deberemos instalar Nginx. Al clonarla, debemos indicarle que utilice una nueva MAC, para que así el proxy pueda obtener una dirección IP diferente al servidor.

Dentro del proxy, tendremos que realizar la configuración para que efectivamente actúe como intermediario. Es decir, reciba las peticiones de la máquina anfitriona, las entregue al servidor origen y devuelva la respuesta correspondiente al cliente.

En el navegador web del cliente, acceder al menú inspeccionar, para comprobar la IP desde la que se nos está sirviendo la página web. Añadir una captura de pantalla de esto.

Añadir las capturas de las configuraciones realizadas indicando a qué corresponden.

Para crear el proxy, he clonado la máquina servidor añadiendo nuevas direcciones MAC como marca el enunciado. Ahora, iniciamos la máquina virtual clonada o proxy, y lo que debemos hacer es en el archivo de configuración que tenemos en sites-available especificar qué dirección IP y por qué puerto queremos que se redirija las solicitudes. Esto lo hacemos dejando el archivo de configuración de la siguiente manera:



```
debian@debian: /etc/nginx/sites-available
GNU nano 7.2 pagina_alejandroperinan.com
server {

    listen 82;
    listen [::]:82;

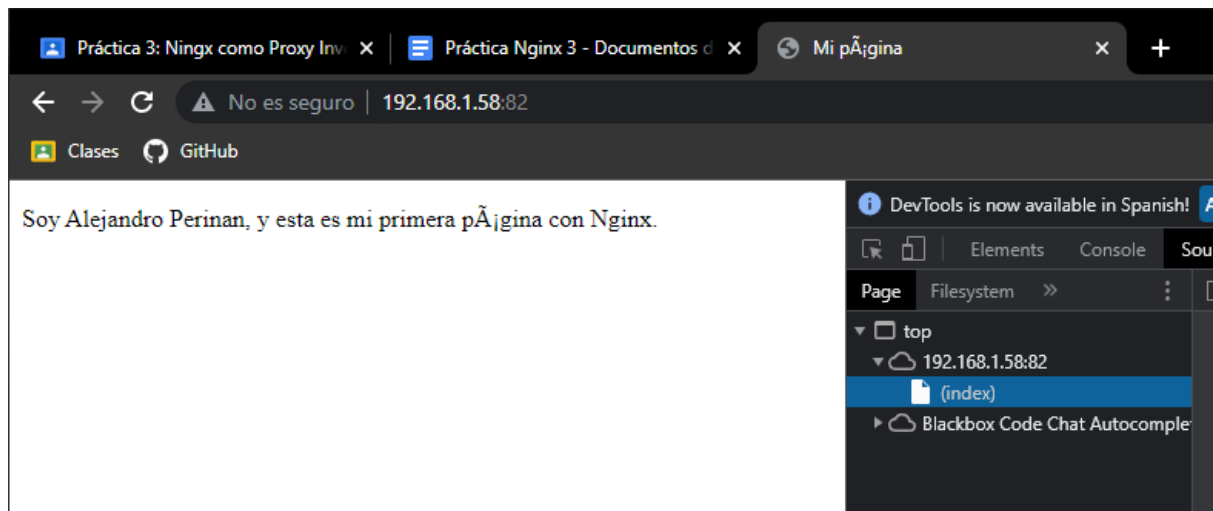
    server_name pagina_alejandroperinan.com;

    root /var/www/pagina_alejandroperinan.com;
    index index.html;

    location / {
        proxy_pass http://192.168.1.74:82;
    }
}
```



Ahora si accedemos a el sitio mediante la IP del proxy por el mismo puerto tenemos acceso.



## PRÁCTICA 4

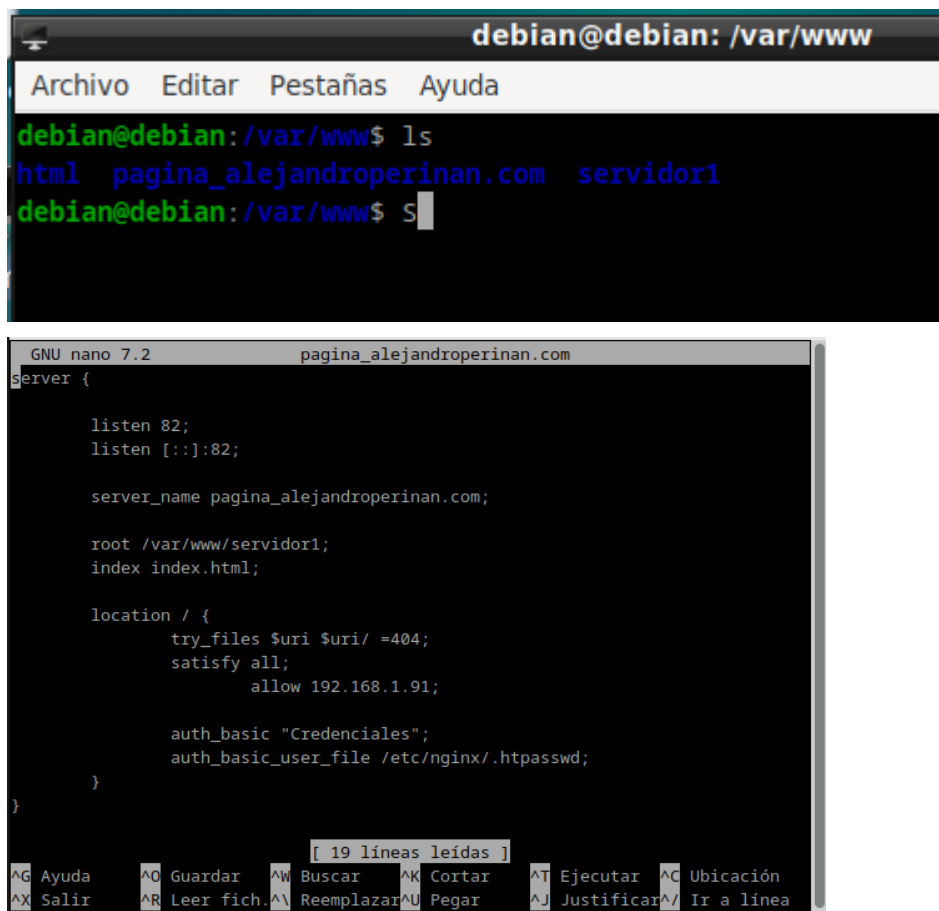
### Configuración de Nginx como balanceador de carga

En esta práctica queremos configurar Nginx como Proxy Inverso, siguiendo el esquema dibujado en la pizarra. El objetivo es que el servidor que se encuentra al frente, sea capaz de repartir las solicitudes de los clientes entre los distintos servidores origen que podamos tener. Ya hemos comentado las ventajas que nos aporta realizar el balanceo de carga.

Habría que realizar otro clon de nuestra primera máquina virtual (la borraremos una vez terminadas las prácticas 4 y 5). Al clonarla, debemos indicarle que utilice una nueva MAC, para que así el proxy pueda obtener una dirección IP diferente al servidor.

En la máquina virtual original (la primera debian que instalamos), vamos a crear una nueva página, que se llame servidor 1. Debe servir una página donde se muestre claramente qué servidor es. En la máquina que hemos creado en el segundo párrafo, crearemos otra página servidor 2. Esta página deberá mostrar que está siendo servida por el servidor 2.

Es conveniente que configuremos las IPs de las dos máquinas para que sean estáticas y no tengamos que estar cambiando la configuración del balanceador.



```
debian@debian: /var/www
Archivo Editar Pestañas Ayuda
debian@debian:/var/www$ ls
html pagina_alejandroperinan.com servidor1
debian@debian:/var/www$ s

GNU nano 7.2 pagina_alejandroperinan.com
server {

    listen 82;
    listen [::]:82;

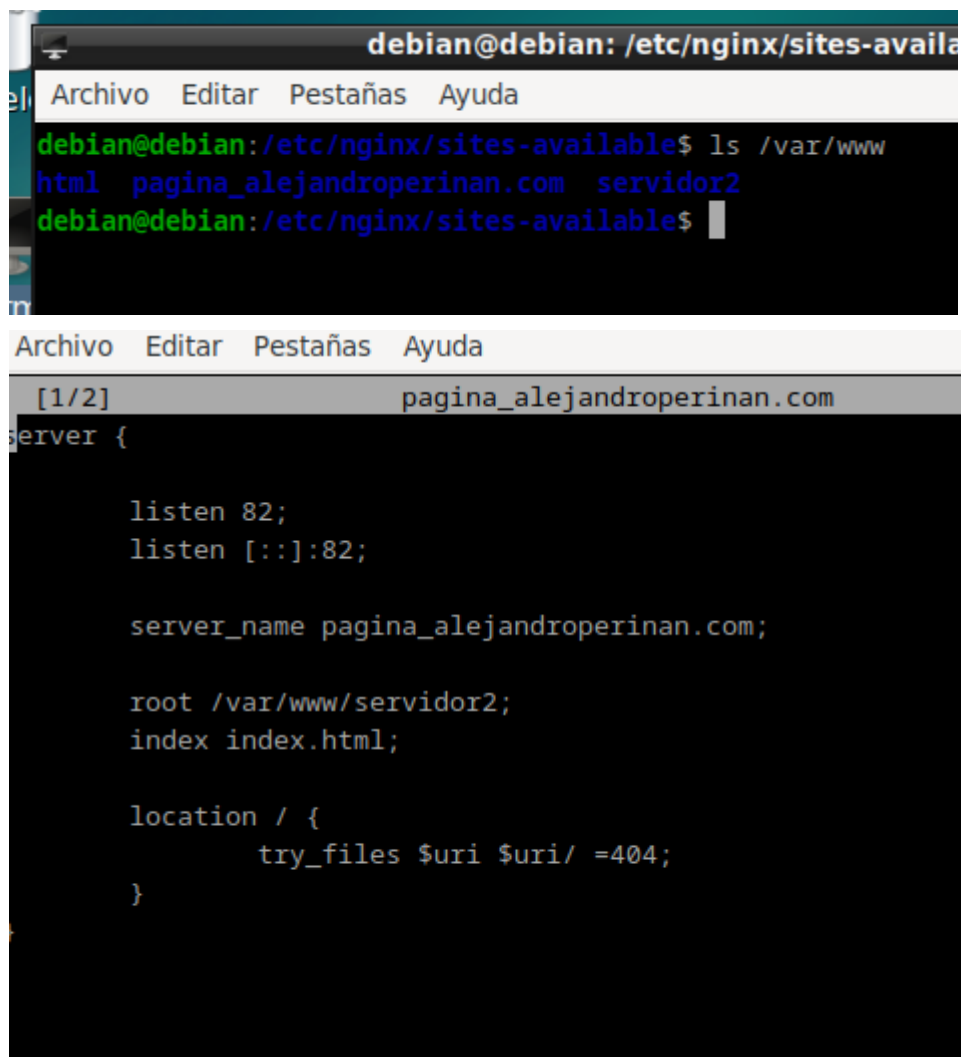
    server_name pagina_alejandroperinan.com;

    root /var/www/servidor1;
    index index.html;

    location / {
        try_files $uri/ =404;
        satisfy all;
        allow 192.168.1.91;

        auth_basic "Credenciales";
        auth_basic_user_file /etc/nginx/.htpasswd;
    }
}

[ 19 líneas leídas ]
^G Ayuda ^O Guardar ^W Buscar ^K Cortar ^T Ejecutar ^C Ubicación
^X Salir ^R Leer fich. ^_ Reemplazar ^U Pegar ^J Justificar ^/ Ir a línea
```



```
debian@debian: /etc/nginx/sites-available
Archivo  Editar  Pestañas  Ayuda
debian@debian:/etc/nginx/sites-available$ ls /var/www
html  pagina_alejandroperinan.com  servidor2
debian@debian:/etc/nginx/sites-available$

[1/2] pagina_alejandroperinan.com
server {

    listen 82;
    listen [::]:82;

    server_name pagina_alejandroperinan.com;

    root /var/www/servidor2;
    index index.html;

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

Con esto configuramos los 2 servidores.

Nos queda realizar la configuración del balanceador (será el proxy\_inverso de la práctica anterior). Sigue el siguiente código:

```
upstream backend_hosts {
    random;
    server (IP y puerto del servidor 1);
    server (IP y puerto del servidor 2);
}

server {
```

```

listen 82;
server_name (IP del proxy);

location / {
    proxy_pass http://backend_hosts;
}
}

```

The screenshot shows a terminal window titled 'debian@debian: /etc/nginx/sites-available'. Inside, the GNU nano 7.2 editor is open, editing a file named 'pagina\_alejandroperinan.com'. The configuration file contains the following content:

```

upstream backend_hosts {
    random;
    server 192.168.1.58:24;
    server 192.168.1.74:24;
}

server {

    listen 82;
    listen [::]:82;

    server_name pagina_alejandroperinan.com;

    location / {
        proxy_pass http://backend_hosts;
    }
}

```

At the bottom of the terminal, a status bar indicates '[ 16 lineas leidas ]' and a menu of keyboard shortcuts is visible, including Ayuda, Guardar, Buscar, Cortar, Ejecutar, Ubicación, Salir, Leer fich, Reemplazar, Pegar, Justificar, and Ir a línea.

¿Qué otras opciones tenemos aparte de random en el bloque de los servidores del backend? ¿Para qué sirven las principales?

round-robin: Este es el método de equilibrio de carga predeterminado en Nginx y distribuye las solicitudes en un ciclo secuencial a través de los servidores en el grupo.

least\_conn: Dirige las solicitudes al servidor con la menor cantidad de conexiones activas en ese momento.

ip\_hash: Utiliza la dirección IP del cliente para determinar a qué servidor enviar la solicitud. Esto garantiza que las solicitudes del mismo cliente siempre se dirijan al mismo servidor, útil para sesiones persistentes.

least\_time: Selecciona el servidor con el tiempo de respuesta más corto o la menor carga, dependiendo de los parámetros que especifiques.

Accede al navegador de la máquina cliente. Busca la dirección IP del balanceador y recarga la página. ¿Por qué cambia?

Por qué de forma aleatoria cada vez que recargamos la página el balanceador de carga nos redirige a un servidor de los 2 que tenemos, que cada uno tiene su IP

## PRÁCTICA 5

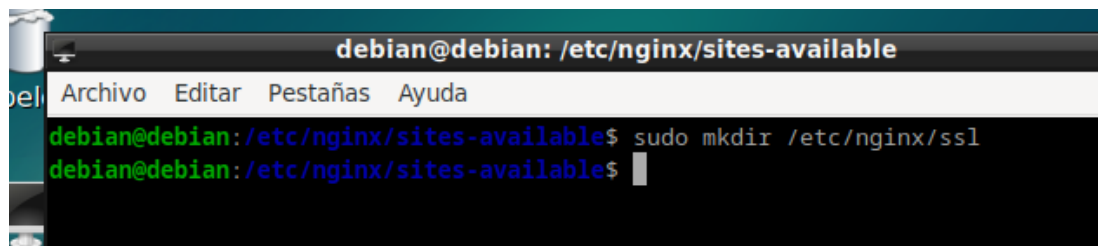
### Nginx como Proxy Inverso/Balanceador con cifrado

En esta práctica última práctica de la unidad 2, modificaremos lo que llevamos hecho hasta ahora de manera que la comunicación que se lleva en nuestro sistema sea cifrada, con el objetivo de añadir una mayor seguridad.

Puesto que nuestros servidores (los 2 servidores origen y el proxy) se encuentran en una red privada, no sería necesario que la comunicación entre éstos se cifrara. Por tanto, el cifrado solo se llevará a cabo entre el proxy y las peticiones de clientes que se reciban. De esta forma, también vamos a conseguir liberar a los servidores origen de tareas como el cifrado/descifrado, quedando totalmente disponibles para servir páginas web.

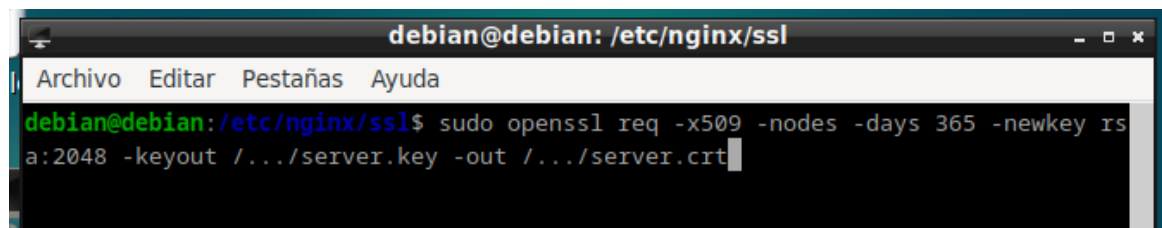
Vamos a utilizar el protocolo HTTPS, que se basa en el uso de certificados digitales. Por tanto, lo primero que haremos será crearnos un certificado (que autofirmaremos):

1. Nos creamos un directorio llamado *ssl* dentro de */etc/nginx/*.



```
debian@debian: /etc/nginx/sites-available
Archivo Editar Pestañas Ayuda
debian@debian:/etc/nginx/sites-available$ sudo mkdir /etc/nginx/ssl
debian@debian:/etc/nginx/sites-available$
```

2. Nos creamos el certificado, junto con el par de claves (pública y privada que lleva asociadas), con el siguiente comando: *sudo openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout ../../server.key -out ../../server.crt*



```
debian@debian: /etc/nginx/ssl
Archivo Editar Pestañas Ayuda
debian@debian:/etc/nginx/ssl$ sudo openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout ../../server.key -out ../../server.crt
```

3. Investiga qué especifica cada una de las partes del comando anterior. Al ejecutar el comando anterior, se nos pedirán una serie de datos que iremos rellenando. Comprobar que tenemos los ficheros de clave y certificado en el directorio correspondiente.

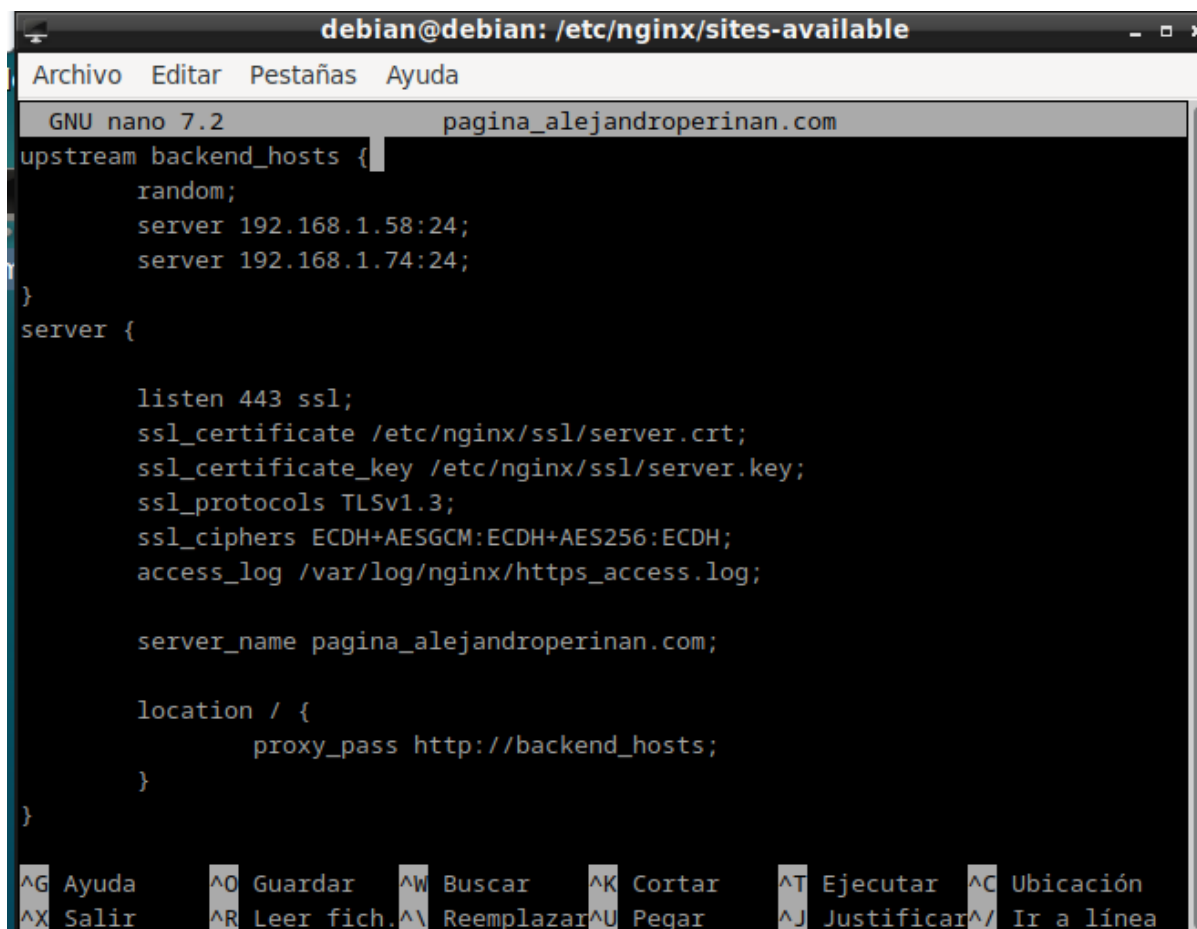
req -x509: Sirve para generar un CSR X509.

-days 365: Sirve para que esté activo el certificado durante 365 días.

-newkey rsa:2048: Sirve para generar una nueva clave privada RSA de 2048 bits.

4. Ahora nos queda modificar el archivo de configuración del proxy para adaptarlo al protocolo HTTPS. Se debe añadir lo siguiente en el bloque del servidor:

```
listen 443 ssl;
ssl_certificate /etc/nginx/ssl/server.crt;
ssl_certificate_key /etc/nginx/ssl/server.key;
ssl_protocols TLSv1.3;
ssl_ciphers ECDH+AESGCM:ECDH+AES256:ECDH;
server_name 192.168.18.20;
access_log /var/log/nginx/https_access.log;
```



The screenshot shows a terminal window titled 'debian@debian: /etc/nginx/sites-available'. The window contains the nano 7.2 editor with the following configuration for 'pagina\_alejandroperinan.com':

```
upstream backend_hosts {
    random;
    server 192.168.1.58:24;
    server 192.168.1.74:24;
}

server {

    listen 443 ssl;
    ssl_certificate /etc/nginx/ssl/server.crt;
    ssl_certificate_key /etc/nginx/ssl/server.key;
    ssl_protocols TLSv1.3;
    ssl_ciphers ECDH+AESGCM:ECDH+AES256:ECDH;
    access_log /var/log/nginx/https_access.log;

    server_name pagina_alejandroperinan.com;

    location / {
        proxy_pass http://backend_hosts;
    }

}
```

The bottom of the window shows the nano editor's command palette with options: Ayuda, Guardar, Buscar, Cortar, Ejecutar, Ubicación, Salir, Leer fich., Reemplazar, Pegar, Justificar, and Ir a línea.

5. Indica para qué sirven las líneas anteriores.
- listen 443 ssl: sirve para escuchar por el puerto 443 que es el del protocolo https y también por ssl
  - ssl\_certificate /etc/nginx/ssl/server.crt: Indica dónde está el certificado.
  - ssl\_certificate\_key /etc/nginx/ssl/server.key: Indica la clave del certificado.
  - ssl\_protocols TLSv1.3: indica la versión del protocolo ssl.
  - ssl\_ciphers ECDH+AESGCM:ECDH+AES256:ECDH: son suites de ssl.
  - access\_log /var/log/nginx/https\_access.log: Indica la ruta del log del https.

6. Accede ahora al servidor e indica qué te aparece en el navegador. ¿Por qué aparece ese mensaje?

Aparece un error 400 debido a que se intenta acceder mediante el protocolo http en lugar de https.

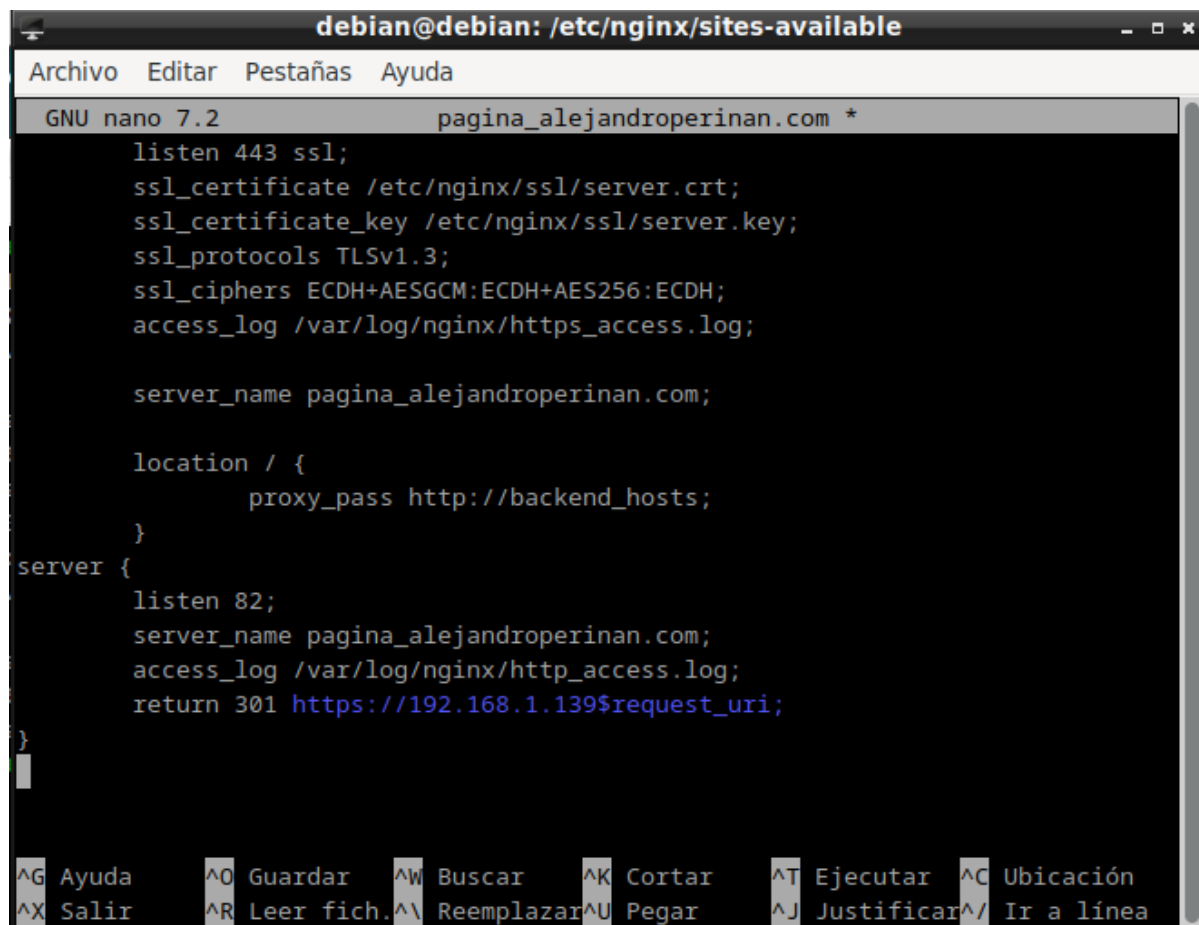
7. Accede al certificado desde el navegador del cliente y muestra una captura de éste.

10.1.4.202:443

92:C3:A9:F1:5D:A1:C2:17:B7:6C:88:B0:18:02:EC:9C:...

8. Vamos a hacer una última modificación al archivo de configuración. El objetivo es que si intentamos acceder a las páginas con HTTP (es decir, sin cifrado), el proxy nos redirija automáticamente a la versión segura. Añade lo siguiente al archivo de configuración:

```
server {  
    listen 82;  
    server_name 192.168.18.20;  
    access_log /var/log/nginx/http_access.log;  
    return 301 https://192.168.18.20$request_uri;  
}
```



The screenshot shows a terminal window titled "debian@debian: /etc/nginx/sites-available". Inside, the nano 7.2 editor is open, editing the file "pagina\_alejandroperinan.com \*". The configuration file contains the following content:

```
listen 443 ssl;  
ssl_certificate /etc/nginx/ssl/server.crt;  
ssl_certificate_key /etc/nginx/ssl/server.key;  
ssl_protocols TLSv1.3;  
ssl_ciphers ECDH+AESGCM:ECDH+AES256:ECDH;  
access_log /var/log/nginx/https_access.log;  
  
server_name pagina_alejandroperinan.com;  
  
location / {  
    proxy_pass http://backend_hosts;  
}  
  
server {  
    listen 82;  
    server_name pagina_alejandroperinan.com;  
    access_log /var/log/nginx/http_access.log;  
    return 301 https://192.168.1.139$request_uri;  
}
```

At the bottom of the terminal, there is a status bar with various keyboard shortcuts for nano editor operations:

^G Ayuda	^O Guardar	^W Buscar	^K Cortar	^T Ejecutar	^C Ubicación
^X Salir	^R Leer fich.	^Y Reemplazar	^U Pegar	^J Justificar	^_ Ir a línea

9. Indica qué significa el código HTTP 301. ¿Qué hace la última línea de la imagen?

HTTP 301 indica que la dirección fue movida a otro sitio.

Redirigir a la página que se ponga detrás del return 301.

Incluye capturas de todas las configuraciones realizadas, indicando a qué corresponde cada una. Responde a todas las preguntas planteadas.