TEMARIO:

ARQUITECTURA EN LA NUBE -

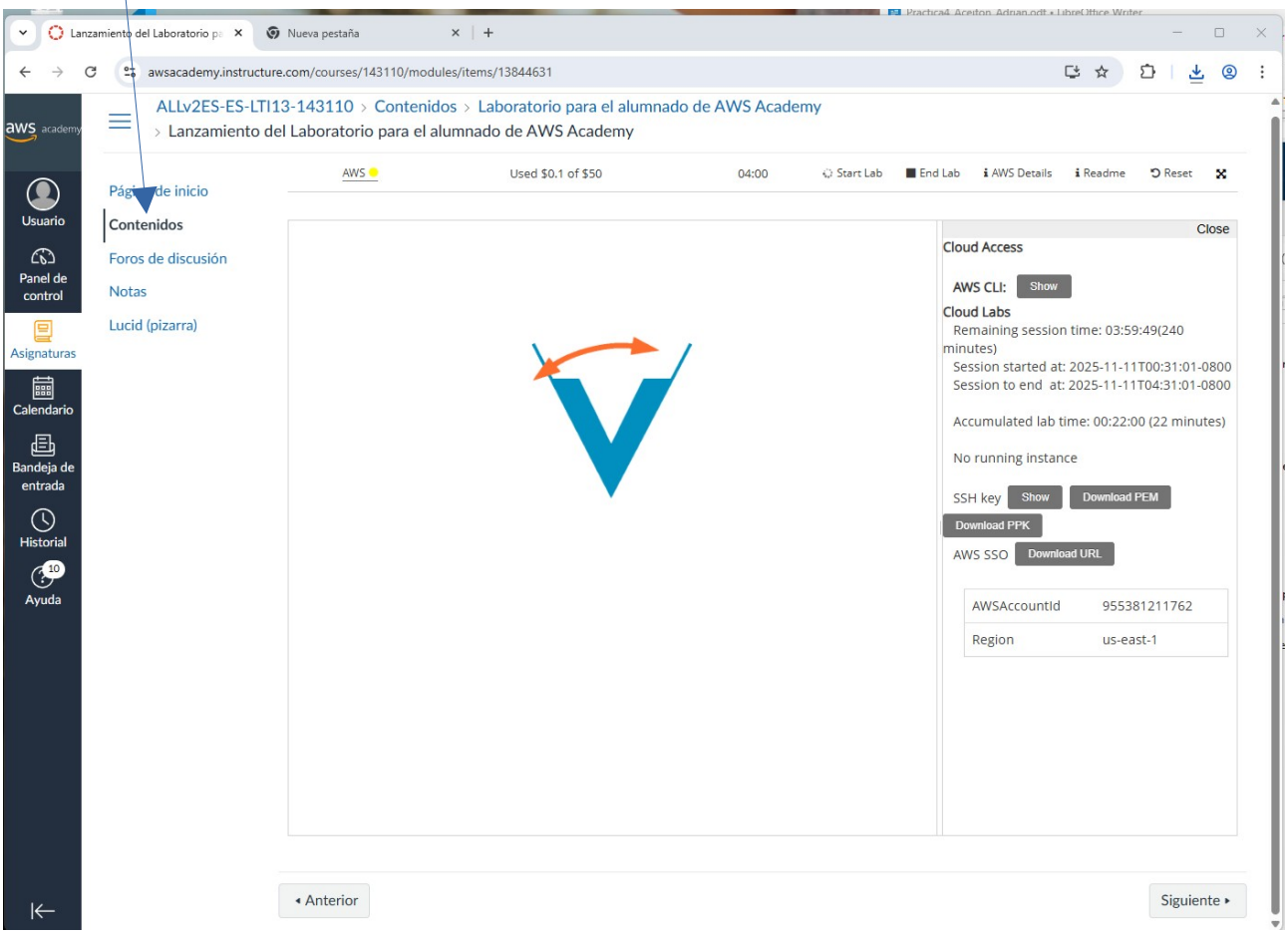
ADMINISTRACIÓN REMOTA DE SERVIDORES WEB EN AWS A TRAVÉS DE SSH

ADRIÁN ACEITÓN PALOMO

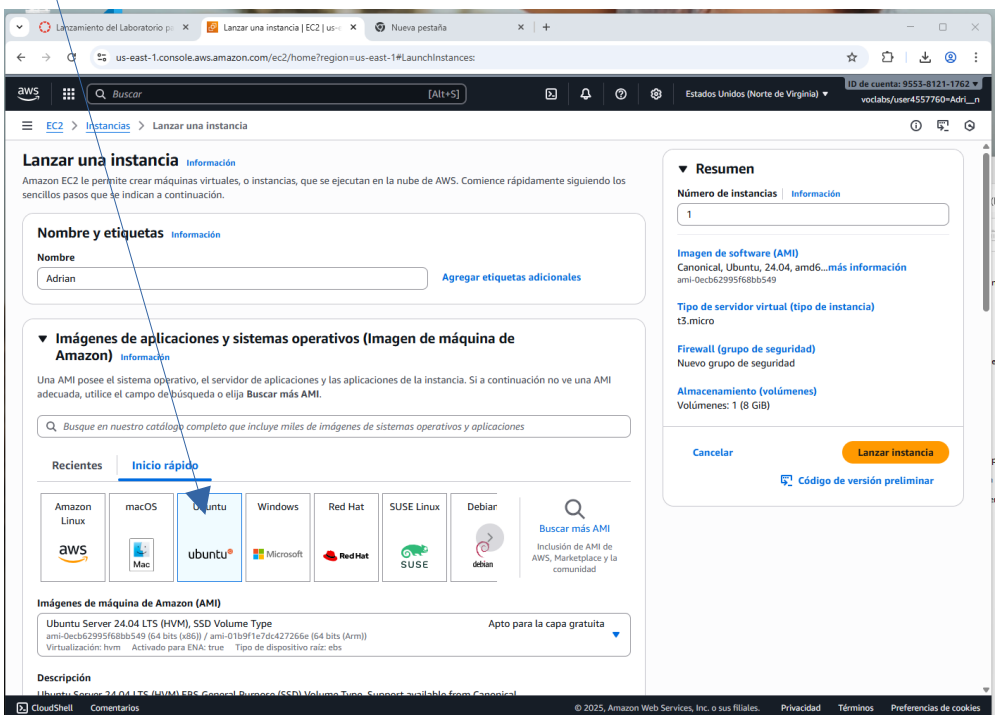
2º ASIR

Primeramente he tenido que registrarme en el AWS Instructure. Una vez registrado he tenido que crear una instancia, para ello he accedido a el apartado de asignaturas, después a contenidos, dentro de contenidos en laboratorio para el alumnado de AWS Academy. Luego en lanzamiento del laboratorio para el alumnado de AWS Academy, cuando le demos ahí se tira un rato cargando, una vez cargado tenemos que descargar el .pem (esto se descarga en AWS Details y una vez dentro en Download PEM). Después tenemos que cambiar los puertos que eso esta en donde pone AWS arriba a la izquierda y luego entramos a la instancia creada, dentro de esa instancia entramos a el apartado de grupos de seguridad, después seleccionamos la instancia y ponemos los puertos y activamos la opción de aceptar todas las IPV4. (Te pongo las capturas enumeradas con orden abajo para que lo entiendas bien). TODO ESTO ES PARA CREAR UNA INSTANCIA EC2 que es básicamente un ordenador linux virtual en la nube.

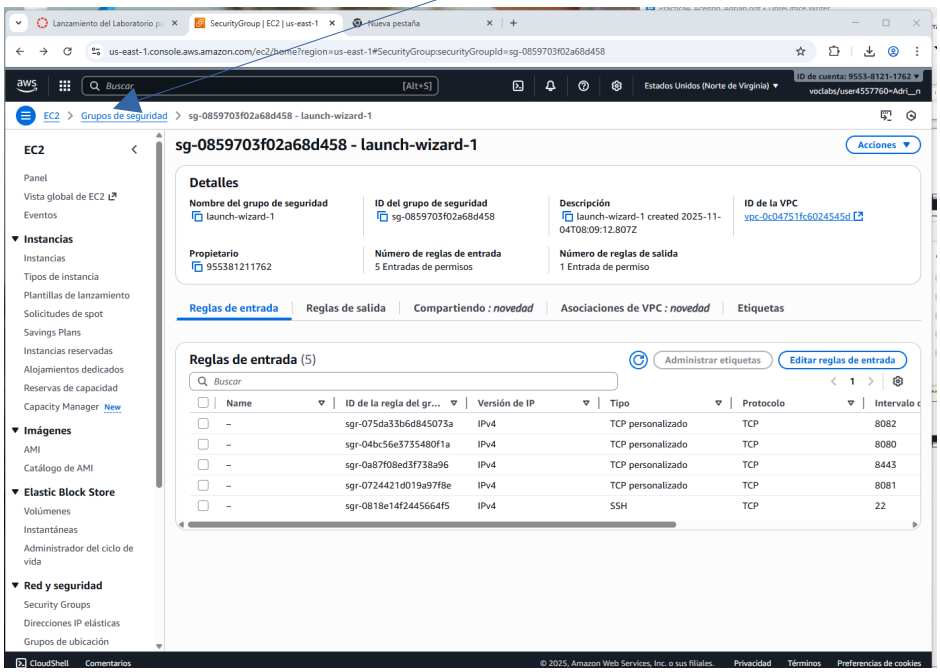
Primera captura: Entramos a los contenidos y a laboratorio para el alumnado de AWS Academy para crear la instancia



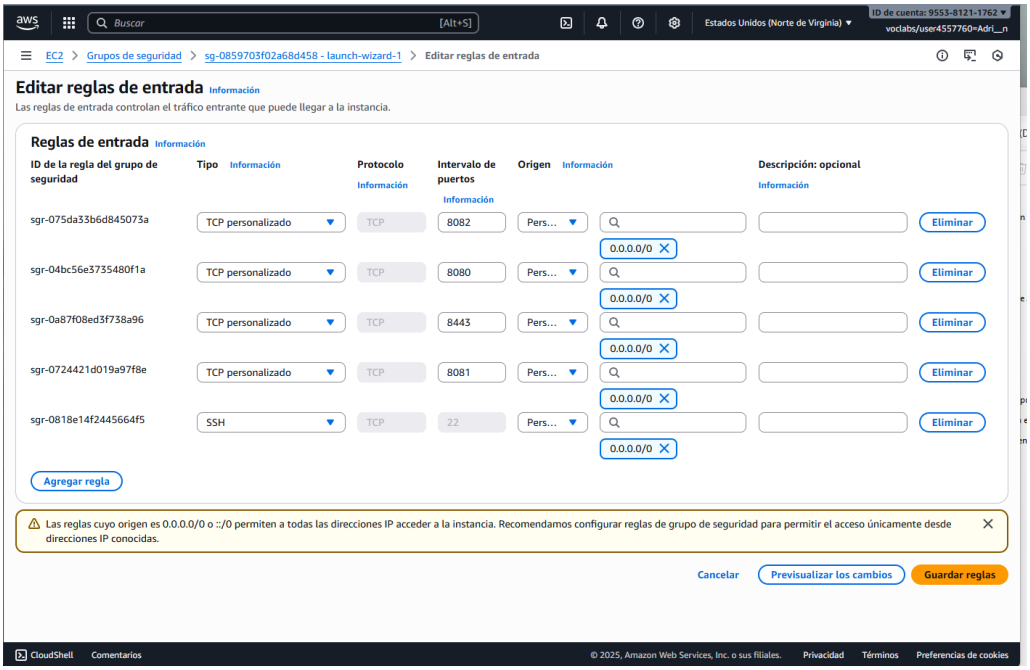
Segunda captura



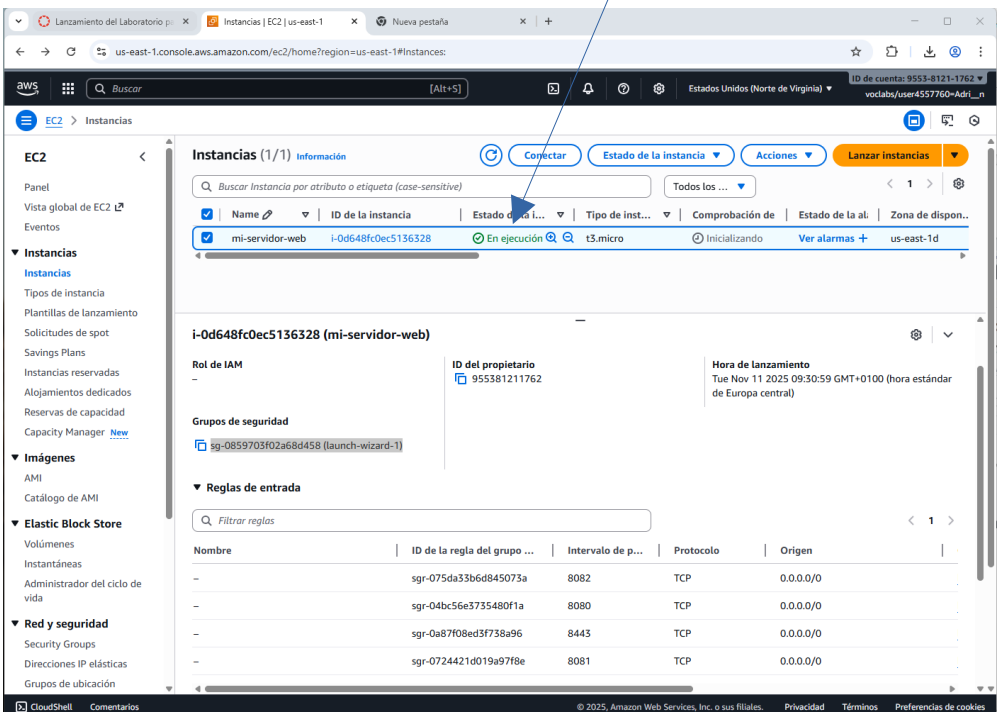
Tercera Captura: Aquí se pueden ver los grupos de seguridad de la instancia que he creado.



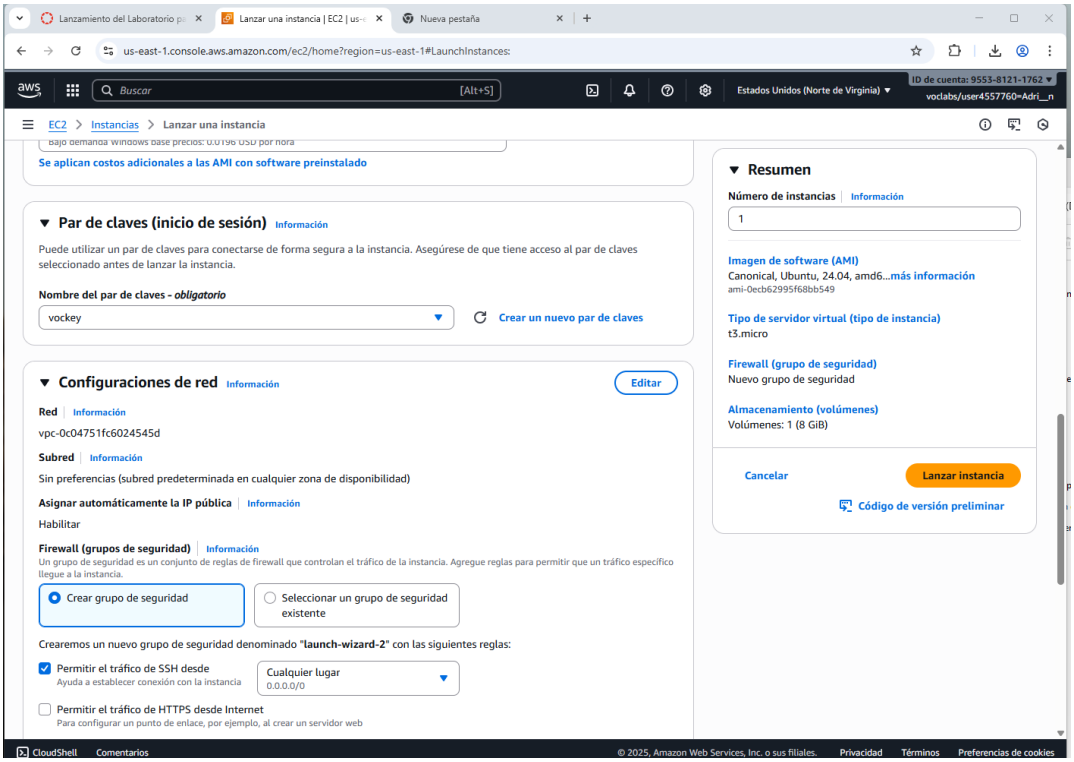
Cuarta Captura: Aquí se ven los puertos configurados correctamente y permitiendo que acceda cualquier IPV4.



Quinta Captura: Aquí vemos que ya está la instancia creada y bien configurada en ejecución.



Esta es una captura extra para ver como se crean los grupos de seguridad y sus configuraciones que expliqué anteriormente.



PARTE 1: CONFIGURACIÓN EN AWS (*Interfaz Visual*)

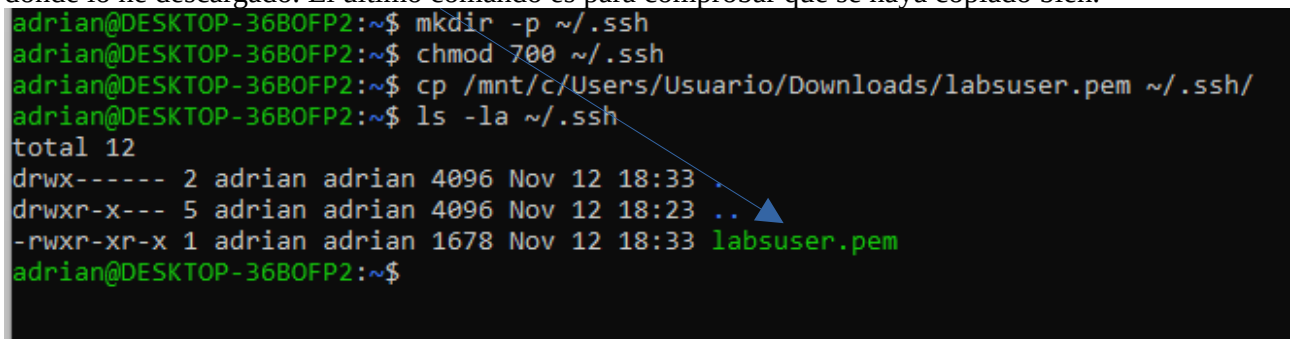
1. Descargar la clave PEM desde la página del laboratorio

1. Localiza el botón o enlace **"Download key"** o **"Download .pem"** 2. Se descargará un archivo con extensión .pem (*por ejemplo: labsuser.pem*)
3. **Importante:** Guarda este archivo en una ubicación segura, preferiblemente en WSL

Pasos para mover la clave a WSL:

Si descargaste el archivo en Windows:
cp /mnt/c/Users/TU-USUARIO/Downloads/labsuser.pem ~/.ssh/
O si prefieres descargarlo directamente en WSL:
cp ~/Downloads/labsuser.pem ~/.ssh/

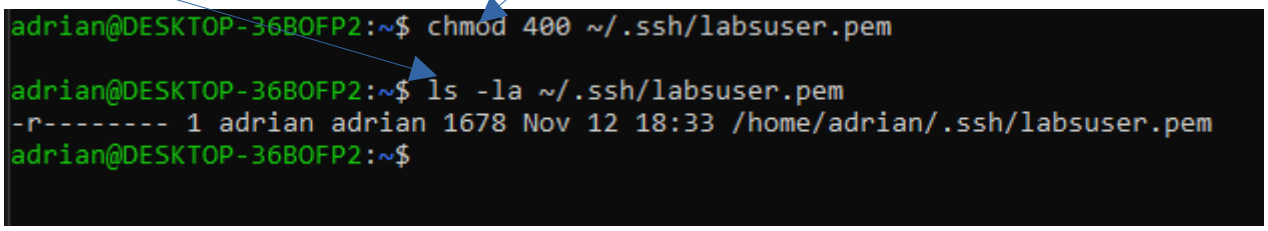
Aquí básciamente he tenido que crear la carpeta, modificar sus permisos y copiar el archivo .pem desde donde lo he descargado. El último comando es para comprobar que se haya copiado bien.



2. Configurar permisos de la clave PEM

chmod 400 ~/.ssh/labsuser.pem
Verifica los permisos:
ls -la ~/.ssh/labsuser.pem
Deberías ver: -r--------

Aquí básicamente estoy configurando los permisos para la clave .pem y luego los compruebo con un LS



3. Crear instancia EC2
1. **Nombre:** servidor-web-practica 2. **AMI:** Ubuntu 22.04 LTS o Ubuntu 24.04 LTS
3. **Tipo:** t2/3.micro (*Free Tier*)

4. **Red:** VPC y Subred por defecto

5. IP pública: Habilitada

6. **Almacenamiento:** 8 GiB, tipo gp2

7. **Etiquetas (*opcional*):** Curso → ArquitecturaNube 8. **Grupo de Seguridad:** sg-servidores-web

PUNTO 4

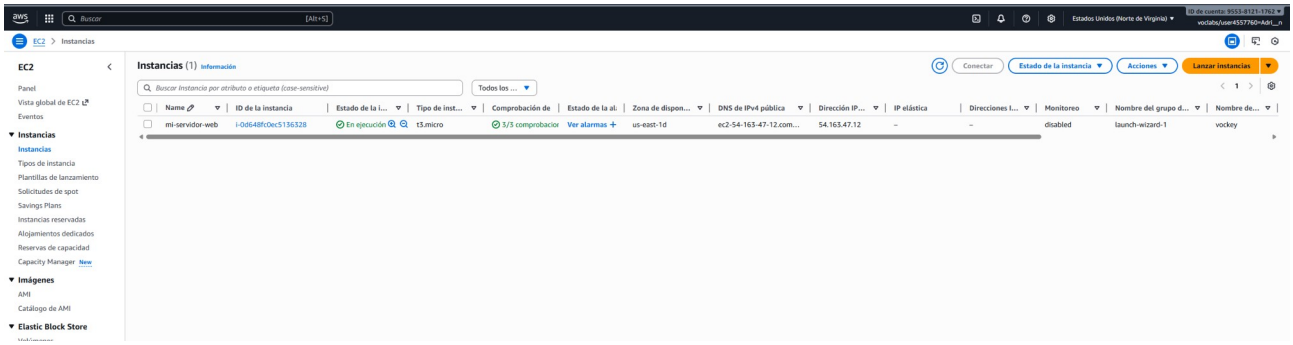
4. Configurar Security Group (*Reglas de entrada*) Puerto	**Protocolo**	**Tipo**	**Descripción**
22	TCP	SSH	Acceso SSH remoto
8080	TCP	HTTP personalizado	Apache HTTP
8081	TCP	HTTP personalizado	Nginx
8082	TCP	HTTP personalizado	Caddy
8443	TCP	HTTPS personalizado	Apache HTTPS (SSL)

5. Especificar la clave PEM al lanzar la instancia
Selecciona **Use existing key pair** y elige la clave labsuser.pem.
Si la instancia fue creada sin clave, termina la instancia y crea una nueva seleccionando la clave correcta.

TODO ESTO YA ESTA DOCUMENTADO CORRECTAMENTE AL PRINCIPIO (puntos 3,4 y 5).

2. Conectar por SSH
ssh -i ~/.ssh/labsuser.pem ubuntu@TU-IP-PUBLICA
Ejemplo:
ssh -i ~/.ssh/labsuser.pem ubuntu@54.123.45.67

Aquí he tenido que ir a instancias y copiar su dirección IP para poder conectarme por SSH.



Aquí ya se ve que me he podido conectar perfectamente a mi maquina AWS a través de SSH.

```
adrian@DESKTOP-36BOFP2:~$ ssh -i ~/.ssh/labsuser.pem ubuntu@54.163.47.12
The authenticity of host '54.163.47.12 (54.163.47.12)' can't be established.
ED25519 key fingerprint is SHA256:Qv4wzfCarSlUHzCsHS1P+ACrcL/IiUHpBvripbyPCuE.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '54.163.47.12' (ED25519) to the list of known hosts.
Welcome to Ubuntu 24.04.3 LTS (GNU/Linux 6.14.0-1016-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

 System information as of Wed Nov 12 18:27:43 UTC 2025

  System load:  0.0               Temperature:           -273.1 C
  Usage of /:   33.6% of 6.71GB   Processes:             110
  Memory usage: 26%               Users logged in:       0
  Swap usage:   0%                IPv4 address for ens5: 172.31.22.228


Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status


Last login: Tue Nov  4 08:12:49 2025 from 18.206.107.28
ubuntu@ip-172-31-22-228:~$
```

Ahora voy a proceder a hacer la practica 2 pero aquí desde WSL conectado remotamente por SSH a AWS.

PARTE 1: INSTALACIÓN Y CONFIGURACIÓN DE APACHE 1.

Adrián Aceitón Palomo

1.Actualizar el sistema

Comando: sudo apt update && sudo apt upgrade -y
Descripción: Esto actualiza la lista de paquetes y mejora el sistema a las últimas versiones

```
ubuntu@ip-172-31-22-228:~$ sudo apt update && sudo apt upgrade -y
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Hit:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease
Hit:4 http://security.ubuntu.com/ubuntu noble-security InRelease
Reading package lists... Done
```

2. Instalar Apache2

Comando: sudo apt install apache2 -y
Descripción: Instala el servidor web Apache en tu sistema.

```
ubuntu@ip-172-31-22-228:~$ sudo apt install apache2 -y
Reading package lists... Done
Building dependency tree... Done
```

3. Configurar Apache en puerto 8080

Comando: sudo nano /etc/apache2/ports.conf
Descripción: Esto abre el archivo de configuración de puertos. Cambio el puerto Listen 80 por Listen 8080.

```
ubuntu@ip-172-31-22-228:~$ sudo sed -i 's/^Listen 80$/Listen 8080/' /etc/apache2/ports.conf
ubuntu@ip-172-31-22-228:~$ sudo sed -i 's/<VirtualHost \*:80>/<VirtualHost *:8080>/' /etc/apache2/sites-available/000-de
fault.conf
ubuntu@ip-172-31-22-228:~$
```

Verificamos que está todo bien.

```
ubuntu@ip-172-31-22-228:~$ cat /etc/apache2/ports.conf
# If you just change the port or add more ports here, you will likely also
# have to change the VirtualHost statement in
# /etc/apache2/sites-enabled/000-default.conf

Listen 8080

<IfModule ssl_module>
        Listen 443
</IfModule>

<IfModule mod_gnutls.c>
        Listen 443
</IfModule>
```

4. Modificar el VirtualHost

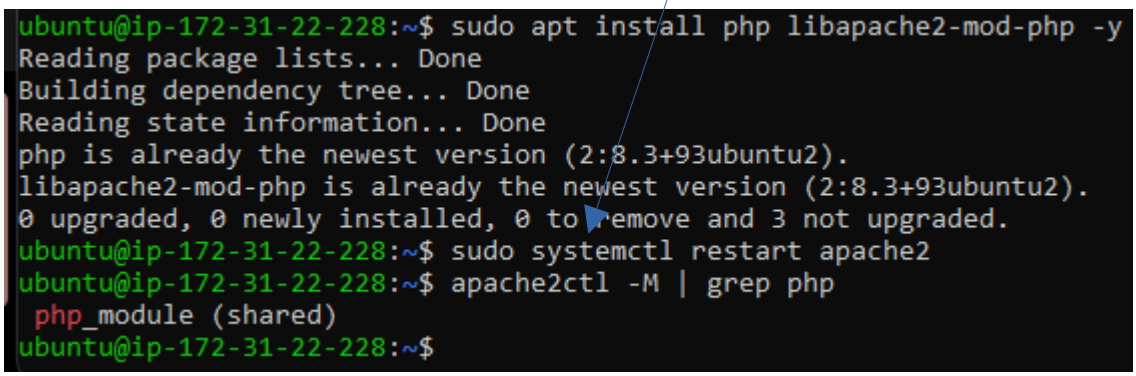
Comando: sudo nano /etc/apache2/sites-available/000-default.conf
Descripción: Para que el cambio de puerto sea efectivo, no es suficiente con el archivo de configuración general. Es necesario también modificar el archivo del sitio (000-default.conf). Cambié la línea <VirtualHost *:80> a <VirtualHost *:8080> para que, de esta forma, el sitio web por defecto de Apache sepa que debe atender las peticiones que lleguen por ese nuevo puerto.

5. Instalar PHP

Comando: sudo apt install php libapache2-mod-php -y
Descripción: Basicamente instalo PHP y ya, también lo verifico para ver que todo esté correcto.

6. Reiniciar Apache

Comando: sudo systemctl restart apache2
Descripción: Reinicio de Apache para aplicar los cambios



7. Verificar estado de Apache

Comando: sudo systemctl status apache2 con el comando sudo netstat -tulpn | grep 8080.
Descripción: Verifico que Apache esté operando correctamente y escuchando de forma específica en el puerto 8080. Para poder usar el comando netstat (que es el que nos permite ver los puertos), es necesario instalar primero el paquete net-tools.

```
ubuntu@ip-172-31-22-228:~$ sudo apt install -y net-tools
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following NEW packages will be installed:
  net-tools
0 upgraded, 1 newly installed, 0 to remove and 3 not upgraded.
Need to get 204 kB of archives.
After this operation, 811 kB of additional disk space will be used.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 net-tools amd64 2.
Fetched 204 kB in 0s (11.1 MB/s)
Selecting previously unselected package net-tools.
(Reading database ... 104228 files and directories currently installed.)
Preparing to unpack .../net-tools_2.10-0.1ubuntu4.4_amd64.deb ...
Unpacking net-tools (2.10-0.1ubuntu4.4) ...
Setting up net-tools (2.10-0.1ubuntu4.4) ...
Processing triggers for man-db (2.12.0-4build2) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
ubuntu@ip-172-31-22-228:~$ sudo netstat -tulpn | grep 8080
tcp6       0      0 :::8080                 :::*                    LISTEN      13733/apache2
ubuntu@ip-172-31-22-228:~$
```

8. Crear archivo PHP de prueba

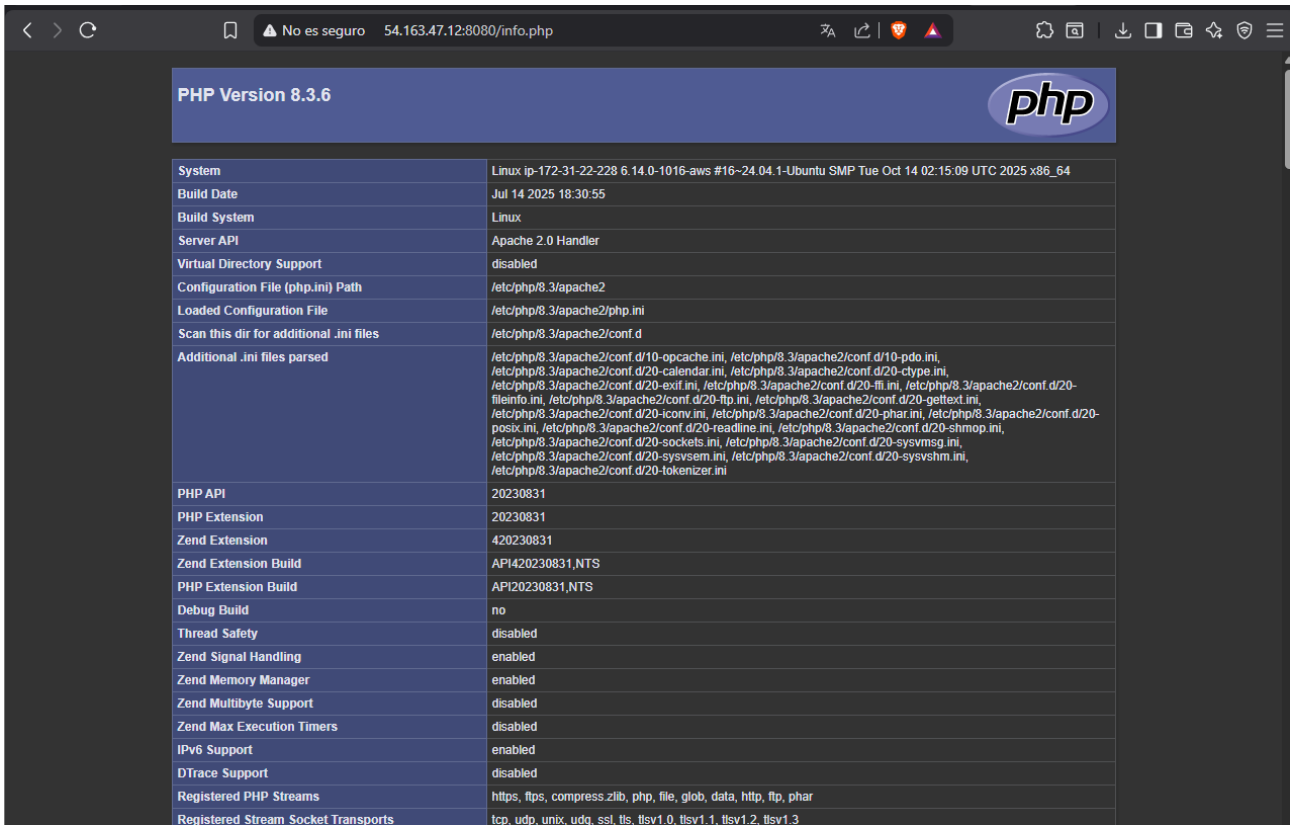
Comando: echo "" | sudo tee /var/www/html/info.php
Descripción: Crear un archivo que muestre información del PHP instalado.

```
ubuntu@ip-172-31-22-228:~$ echo "<?php phpinfo(); ?>" | sudo tee /var/www/html/info.php
<?php phpinfo(); ?>
ubuntu@ip-172-31-22-228:~$
```

9. Probar Apache desde terminal

Comando: curl http://localhost:8080/info.php
Descripción: Esto verifica que Apache sirve correctamente el contenido PHP

```
ubuntu@ip-172-31-22-228:~$ curl http://localhost:8080/info.php
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"><head>
<style type="text/css">
body {background-color: #fff; color: #222; font-family: sans-serif;}
pre {margin: 0; font-family: monospace;}
a:link {color: #009; text-decoration: none; background-color: #fff;}
a:hover {text-decoration: underline;}
table {border-collapse: collapse; border: 0; width: 934px; box-shadow: 1px 2px 3px rgba(0, 0, 0, 0.2);}
.center {text-align: center;}
.center table {margin: 1em auto; text-align: left;}
.center th {text-align: center !important;}
td, th {border: 1px solid #666; font-size: 75%; vertical-align: baseline; padding: 4px 5px;}
th {position: sticky; top: 0; background: inherit;}
h1 {font-size: 150%;}
h2 {font-size: 125%;}
h2 a:link, h2 a:visited{color: inherit; background: inherit;}
```



System	Linux ip-172-31-22-228 6.14.0-1016-aws #16~24.04.1-Ubuntu SMP Tue Oct 14 02:15:09 UTC 2025 x86_64
Build Date	Jul 14 2025 18:30:55
Build System	Linux
Server API	Apache 2.0 Handler
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc/php/8.3/apache2
Loaded Configuration File	/etc/php/8.3/apache2/php.ini
Scan this dir for additional .ini files	/etc/php/8.3/apache2/conf.d
Additional .ini files parsed	/etc/php/8.3/apache2/conf.d/10-opcache.ini, /etc/php/8.3/apache2/conf.d/10-pdo.ini, /etc/php/8.3/apache2/conf.d/20-calendar.ini, /etc/php/8.3/apache2/conf.d/20-ctype.ini, /etc/php/8.3/apache2/conf.d/20-exif.ini, /etc/php/8.3/apache2/conf.d/20-ffi.ini, /etc/php/8.3/apache2/conf.d/20-fileinfo.ini, /etc/php/8.3/apache2/conf.d/20-ftp.ini, /etc/php/8.3/apache2/conf.d/20-gettext.ini, /etc/php/8.3/apache2/conf.d/20-iconv.ini, /etc/php/8.3/apache2/conf.d/20-phar.ini, /etc/php/8.3/apache2/conf.d/20-posix.ini, /etc/php/8.3/apache2/conf.d/20-readline.ini, /etc/php/8.3/apache2/conf.d/20-shmop.ini, /etc/php/8.3/apache2/conf.d/20-sockets.ini, /etc/php/8.3/apache2/conf.d/20-sysvmsg.ini, /etc/php/8.3/apache2/conf.d/20-sysvsem.ini, /etc/php/8.3/apache2/conf.d/20-sysvshm.ini, /etc/php/8.3/apache2/conf.d/20-tokenizer.ini
PHP API	20230831
PHP Extension	20230831
Zend Extension	420230831
Zend Extension Build	API420230831,NTS
PHP Extension Build	API20230831,NTS
Debug Build	no
Thread Safety	disabled
Zend Signal Handling	enabled
Zend Memory Manager	enabled
Zend Multibyte Support	disabled
Zend Max Execution Timers	disabled
IPv6 Support	enabled
DTrace Support	disabled
Registered PHP Streams	https, ftps, compress.zlib, php, file, glob, data, http, ftp, phar
Registered Stream Socket Transports	tcp, udp, unix, udg, ssl, tls, tlsv1.0, tlsv1.1, tlsv1.2, tlsv1.3

PARTE 2: INSTALACIÓN Y CONFIGURACIÓN DE NGINX

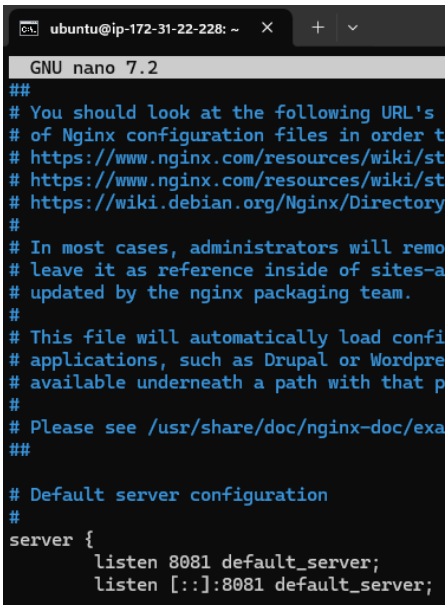
1. Instalar Nginx

Comando: sudo apt install nginx -y
Descripción: Instalación del servidor web Nginx en tu sistema.

```
ubuntu@ip-172-31-22-228:~$ sudo apt install nginx -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  nginx-common
```

2. Configurar Nginx en puerto 8081

Comando: sudo nano /etc/nginx/sites-available/default
Descripción: Con este comando abro el archivo principal del sitio que Nginx carga por defecto. Aquí es donde se define en qué puerto está escuchando el servidor web. Básicamente lo que hago es buscar la línea donde aparece listen 80, que es el puerto web típico, y la cambio por listen 8081. De esta forma, le indico a Nginx que en lugar de trabajar por el puerto 80, ahora su página va a funcionar por el puerto 8081.



3. Crear página HTML personalizada

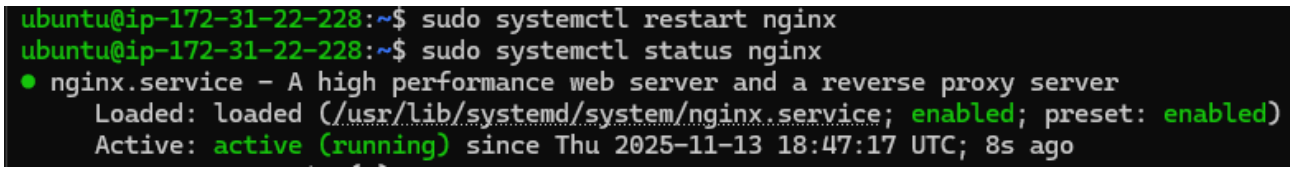
Comando: echo "Servidor Nginx Funcionando en puerto 8081" | sudo tee /var/www/html/html/index.html (LA RUTA DE LA PRACTICA 1 ES LA CORRECTA, LA DE LA PRACTICA 2 ESTA MAL.)
Descripción: Creación de una página HTML para Nginx.

4. Reiniciar Nginx

Comando: sudo systemctl restart nginx
Descripción: Aquí reinicio Nginx para aplicar los cambios de configuración

5. Verificar estado de Nginx

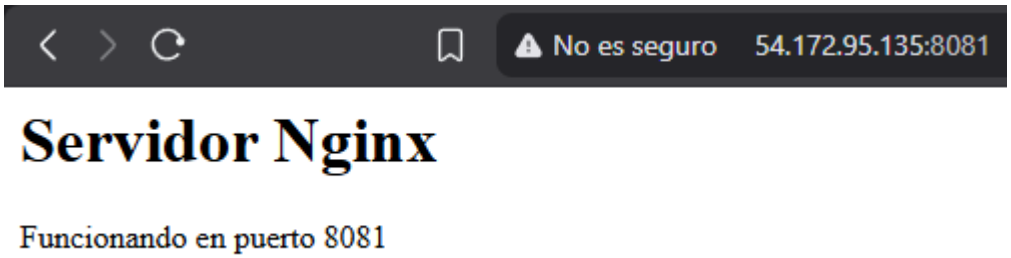
Comando: sudo systemctl status nginx
Descripción: Comprueba que Nginx está funcionando correctamente en el puerto 8081



6. Probar Nginx desde terminal

Comando: curl http://localhost:8081
Descripción: Verifica que Nginx sirve correctamente el contenido HTML.

```
ubuntu@ip-172-31-22-228:~$ curl http://localhost:8081
<h1>Servidor Nginx</h1><p>Funcionando en puerto 8081</p>
ubuntu@ip-172-31-22-228:~$
```



PARTE 3: INSTALACIÓN Y CONFIGURACIÓN DE CADDY

1. Instalar dependencias necesarias

Comando: sudo apt install -y debian-keyring debian-archive-keyring apt-transport-https curl
Descripción: Aquí simplemente instalo algunas herramientas que necesita el sistema para poder añadir repositorios externos sin problemas. Básicamente son claves y utilidades que permiten descargar software de fuentes seguras. Es un paso previo antes de poder instalar Caddy.

```
ubuntu@ip-172-31-22-228:~$ sudo apt install -y debian-keyring debian-archive-keyring apt-transport-https curl
Reading package lists... Done
```

2. Agregar repositorio de Caddy

Comando: curl -1sLf 'https://dl.cloudsmith.io/public/caddy/stable/gpg.key' | sudo gpg - dearmor -o /usr/share/keyrings/caddy-stable-archive-keyring.gpg curl -1sLf y 'https://dl.cloudsmith.io/public/caddy/stable/debian.deb.txt' | sudo tee /etc/apt/sources.list.d/caddy-stable.list
Descripción: Aquí lo que hago es agregar el repositorio oficial de Caddy para poder instalar siempre la última versión estable.
Primero descargo la clave GPG del repositorio y la convierto en formato compatible con APT para que el sistema pueda verificar que los paquetes son auténticos.
Después descargo el archivo del repositorio y lo guardo dentro de /etc/apt/sources.list.d/.

```
ubuntu@ip-172-31-22-228:~$ curl -1sLf 'https://dl.cloudsmith.io/public/caddy/stable/gpg.key' | \
sudo gpg --dearmor -o /usr/share/keyrings/caddy-stable-archive-keyring.gpg
ubuntu@ip-172-31-22-228:~$ curl -1sLf 'https://dl.cloudsmith.io/public/caddy/stable/debian.deb.txt' | \
sudo tee /etc/apt/sources.list.d/caddy-stable.list
# Source: Caddy
# Site: https://github.com/caddyserver/caddy
# Repository: Caddy / stable
# Description: Fast, multi-platform web server with automatic HTTPS
```

3. Actualizar e instalar Caddy

Comando: sudo apt update && sudo apt install caddy -y
Descripción: Actualiza la lista de paquetes e instala Caddy.

```
ubuntu@ip-172-31-22-228:~$ sudo apt update && sudo apt install caddy -y
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
```

4. Crear directorio para Caddy

Comando: sudo mkdir -p /var/www/caddy
Descripción: Aquí creo un directorio específico para los archivos de Caddy

```
ubuntu@ip-172-31-22-228:~$ sudo mkdir -p /var/www/caddy
ubuntu@ip-172-31-22-228:~$
```

5. Crear archivo Markdown de prueba

Comando: echo "# Bienvenido a Caddy" | sudo tee /var/www/caddy/README.md echo "" | sudo tee -a /var/www/caddy/README.md echo "Este servidor está funcionando correctamente." | sudo tee -a /var/www/caddy/README.md echo "" | sudo tee -a /var/www/caddy/README.md echo "## Características" | sudo tee -a /var/www/caddy/README.md echo "- Servidor moderno" | sudo tee -a /var/www/caddy/README.md echo "- HTTPS automático" | sudo tee -a /var/www/caddy/README.md echo "- Fácil configuración" | sudo tee -a /var/www/caddy/README.md
Descripción: Aquí lo que hago es crear un archivo README.md dentro de la carpeta de Caddy. Le voy metiendo varias líneas en formato Markdown: un título, un pequeño texto y una lista de características. La idea es simplemente tener un archivo de prueba para comprobar luego que Caddy está mostrando bien su contenido. Básicamente dejo preparado un fichero para verificar que el servidor funciona.

```
ubuntu@ip-172-31-22-228:~$ sudo mkdir -p /var/www/caddy
ubuntu@ip-172-31-22-228:~$ echo "# Bienvenido a Caddy" | sudo tee /var/www/caddy/README.md
# Bienvenido a Caddy
ubuntu@ip-172-31-22-228:~$ echo "" | sudo tee -a /var/www/caddy/README.md

ubuntu@ip-172-31-22-228:~$ echo "Este servidor está funcionando correctamente." | sudo tee -a /var/www/caddy/README.md
Este servidor está funcionando correctamente.
ubuntu@ip-172-31-22-228:~$ echo "" | sudo tee -a /var/www/caddy/README.md

ubuntu@ip-172-31-22-228:~$ echo "## Características" | sudo tee -a /var/www/caddy/README.md
## Características
ubuntu@ip-172-31-22-228:~$ echo "- Servidor moderno" | sudo tee -a /var/www/caddy/README.md
- Servidor moderno
ubuntu@ip-172-31-22-228:~$ echo "- HTTPS automático" | sudo tee -a /var/www/caddy/README.md
- HTTPS automático
ubuntu@ip-172-31-22-228:~$ echo "- Fácil configuración" | sudo tee -a /var/www/caddy/README.md
- Fácil configuración
ubuntu@ip-172-31-22-228:~$
```

6. Crear imagen de prueba (cuidado wsl hay que hacer ajustes)

Comando: curl -o /tmp/test-image.jpg "https://www.python.org/static/apple-touch icon-144x144-precomposed.png" sudo mv /tmp/test-image.jpg /var/www/caddy/test.jpg
Descripción: Aquí descargo una imagen cualquiera usando curl y la guardo temporalmente en /tmp. Luego la muevo a /var/www/caddy, que es la carpeta que Caddy va a servir.
Antes de eso creo la carpeta y le doy permisos para evitar problemas con WSL.
Al final queda un archivo estático listo para comprobar si Caddy sirve imágenes sin errores.

```
ubuntu@ip-172-31-22-228:~$ sudo mkdir -p /var/www/caddy
ubuntu@ip-172-31-22-228:~$ sudo chmod -R 755 /var/www/caddy
ubuntu@ip-172-31-22-228:~$ curl -o /tmp/test-image.jpg "https://www.python.org/static/apple-touch-icon-144x144-precompos
ed.png"
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  7382  100  7382    0     0   172k      0 --:--:-- --:--:-- --:--:--  171k
ubuntu@ip-172-31-22-228:~$ sudo mv /tmp/test-image.jpg /var/www/caddy/test.jpg
ubuntu@ip-172-31-22-228:~$
```

7. Crear Caddyfile personalizado

Comando: sudo nano /etc/caddy/Caddyfile
Descripción: Con este comando edito el Caddyfile, que es el archivo de configuración principal de Caddy. Introduzco un bloque de configuración para definir un nuevo sitio en el puerto 8082. Dentro de este bloque, le indico que la carpeta raíz de los archivos web es /var/www/caddy y activo el servidor de archivos (file_server) con la opción browse, que nos permite ver un listado de los contenidos de las carpetas. También incluyo una configuración para que los archivos Markdown (.md) se sirvan como texto plano.

```
:8082 {
    root * /var/www/caddy
    file_server browse

    @markdown path *.md
    header @markdown Content-Type text/plain
}
        # Set this path to your site's directory.

        # Enable the static file server.

        # Another common task is to set up a reverse proxy:
        # reverse_proxy localhost:8080

        # Or serve a PHP site through php-fpm:
        # php_fastcgi localhost:9000

# Refer to the Caddy docs for more information:
# https://caddyserver.com/docs/caddyfile
```

7, 8, 9, 10, 11.

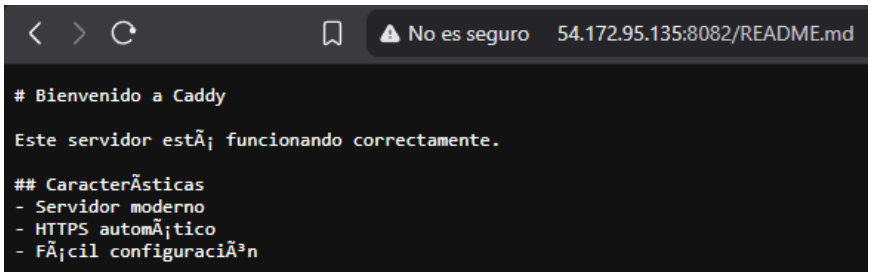
Aquí simplemente reinicio el servicio de Caddy para aplicar toda la configuración que puse en el Caddyfile. Después compruebo con `status` que esté activo y funcionando. También hago un `curl` al README.md para asegurarme de que Caddy está sirviendo correctamente los archivos markdown. Por último, abro en el navegador LA IP PUBLICA DEL SERVIDOR AWS`:8082` y reviso tanto el README como la imagen de prueba para confirmar que todo carga bien.

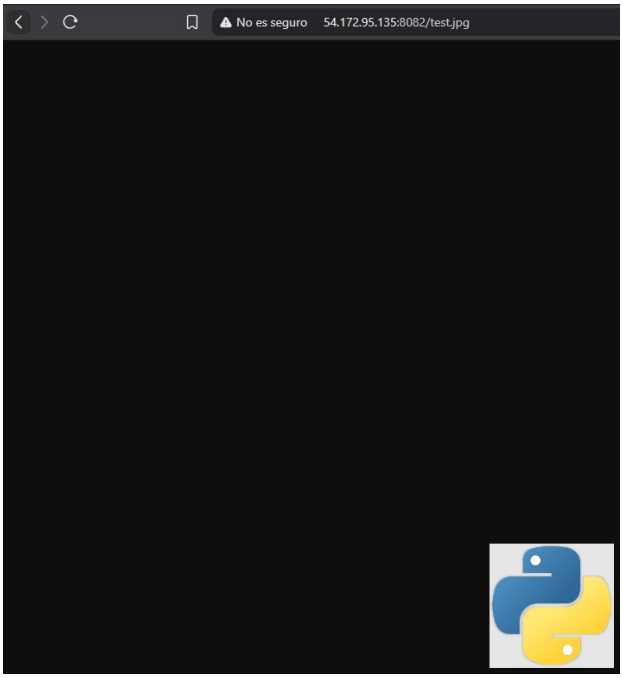
```
ubuntu@ip-172-31-22-228:~$ sudo systemctl restart caddy
ubuntu@ip-172-31-22-228:~$ sudo systemctl status caddy
● caddy.service - Caddy
     Loaded: loaded (/usr/lib/systemd/system/caddy.service; enabled; preset: enabled)
     Active: active (running) since Thu 2025-11-13 19:25:46 UTC; 3s ago
```

```
ubuntu@ip-172-31-22-228:~$ curl http://localhost:8082/README.md
# Bienvenido a Caddy

Este servidor está funcionando correctamente.

## Características
- Servidor moderno
- HTTPS automático
- Fácil configuración
```





PARTE 4: CONFIGURACIÓN DE HTTPS CON CERTBOT EN APACHE

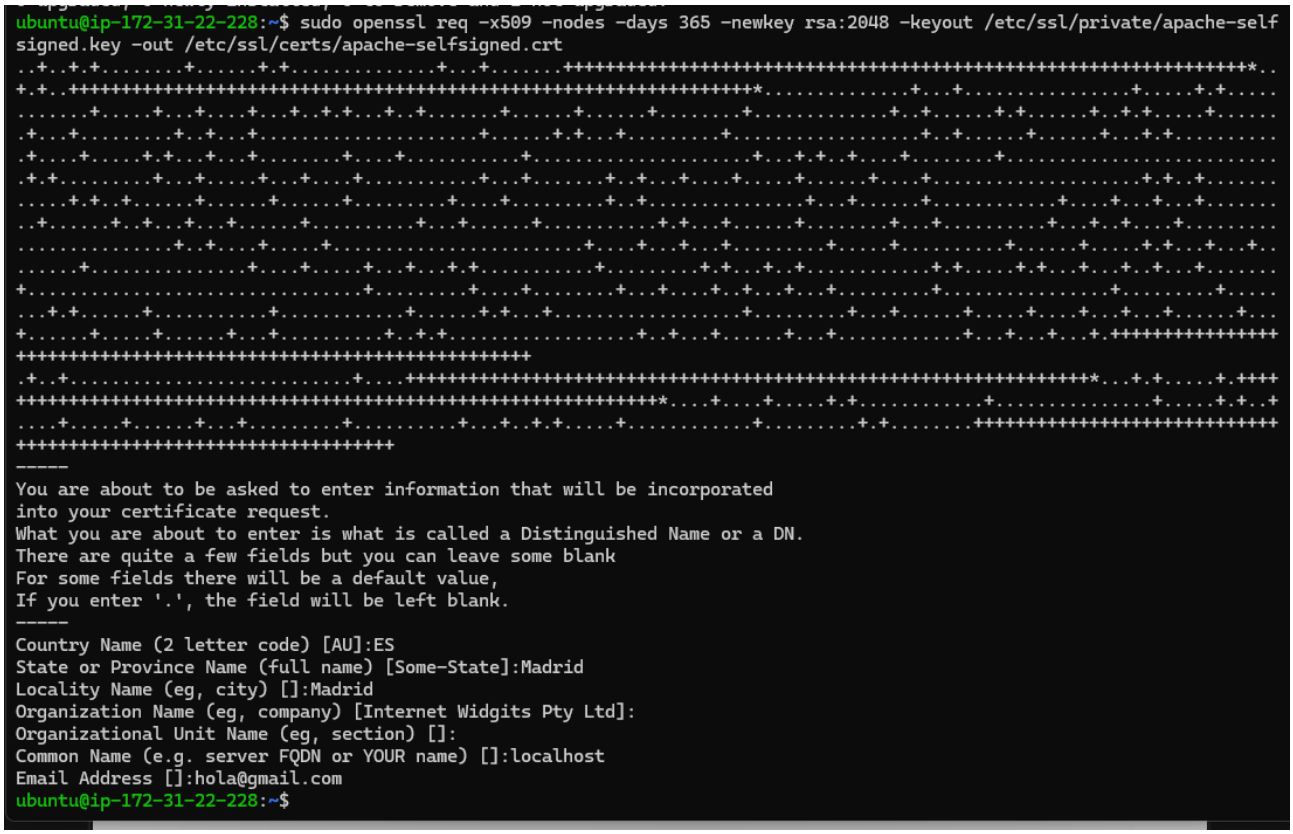
1. Instalar Certbot y el plugin de Apache

Comando: sudo apt install certbot python3-certbot-apache -y
Descripción: Aquí instalo Certbot y su integración con Apache para gestionar certificados SSL.

```
ubuntu@ip-172-31-22-228:~$ sudo apt install certbot python3-certbot-apache -y
Reading package lists... Done
Building dependency tree... Done
```

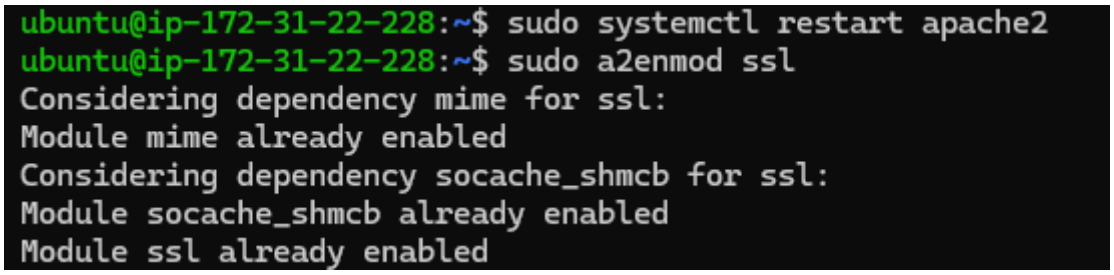
2. Verificar dominio o usar localhost

Nota: Para obtener certificados reales de Let's Encrypt necesitas un dominio público. Para esta práctica usaremos certificados autofirmados.
Comando: sudo openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout /etc/ssl/private/apache-selfsigned.key -out /etc/ssl/certs/apache-selfsigned.crt
Descripción: Con este comando creo un certificado autofirmado para poder probar HTTPS sin necesidad de un dominio real. Me va a pedir varios datos (país, ciudad, nombre común, etc.), y simplemente relleno lo básico como en la captura. El dato importante es el Common Name, donde pongo localhost, porque es donde voy a usar el certificado.



3. Habilitar módulo SSL en Apache

Comando: sudo a2enmod ssl
Descripción: Activa el módulo SSL necesario para HTTPS en Apache.
Antes de nada hay que reiniciar el servicio apach2 para que te deje activar el módulo ssl.



4. Crear configuración SSL para Apache

Comando: sudo nano /etc/apache2/sites-available/default-ssl.conf
Descripción: Edita el archivo y asegúrate de que incluye estas líneas dentro de *:443>: SSLEngine on SSLCertificateFile /etc/ssl/certs/apache-selfsigned.crt SSLCertificateKeyFile /etc/ssl/private/apache-selfsigned.key

(ESTA HECHO ABAJO)

TODAS ESTAS YA ESTAN HECHAS EN UNA SOLA CAPTURA

5. Cambiar puerto SSL

Comando: sudo nano /etc/apache2/ports.conf
Descripción: Añade la línea Listen 8443 para que Apache escuche HTTPS en puerto 8443.

6. Modificar VirtualHost SSL

Comando: sudo nano /etc/apache2/sites-available/default-ssl.conf
Descripción: Cambia por .

7. Habilitar sitio SSL

Comando: sudo a2ensite default-ssl.conf
Descripción: Activa la configuración SSL en Apache.

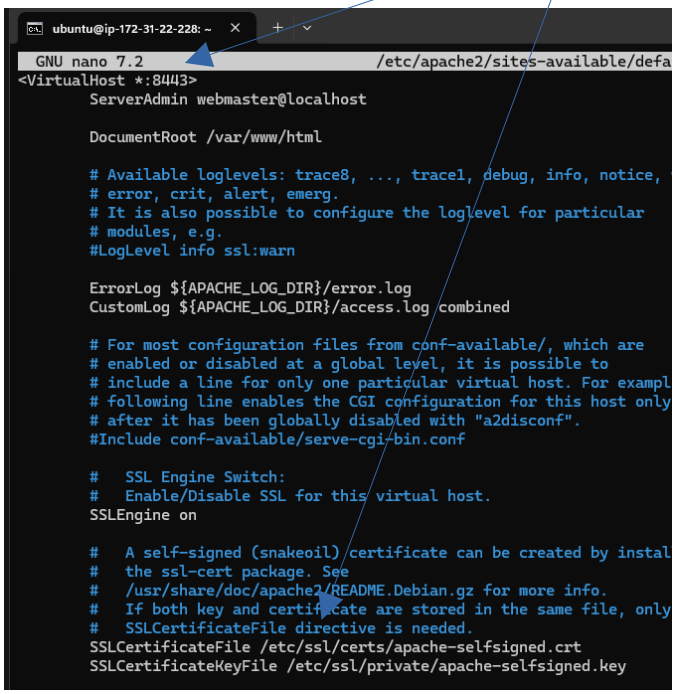
8. Reiniciar Apache

Comando: sudo systemctl restart apache2
Descripción: Aplica todos los cambios de configuración SSL.

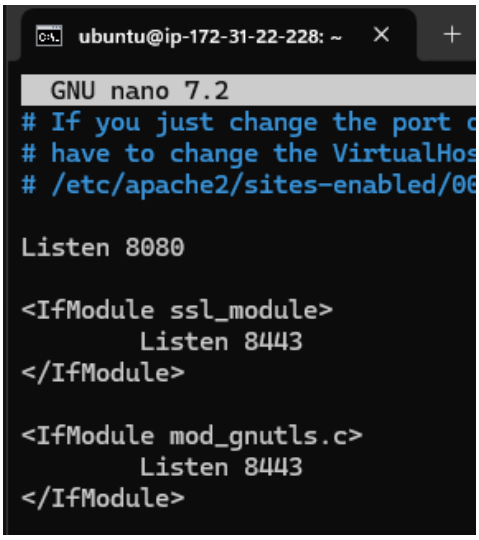
9. Verificar HTTPS

Comando: curl -i -k https://localhost:8443
Descripción: Prueba la conexión HTTPS (el flag -k ignora el aviso del certificado autofirmado).

Aquí lo que he hecho es cambiar los certificados ssl para que sean autofirmados, pero lo principal era cambiar el puerto del 443 al 8443

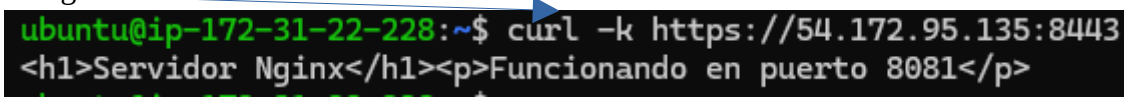


Listen 8443 (he tenido que cambiar el puerto de escucha al 8443 y ya).

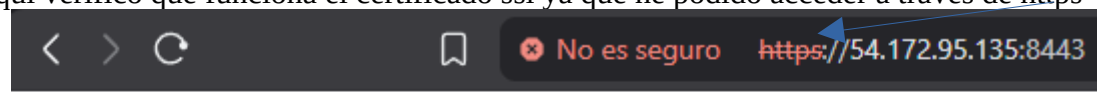


Aquí compruebo que esté todo lo de los puertos correcto.

Aquí hago una consulta



Aquí verifico que funciona el certificado ssl ya que he podido acceder a traves de https



Servidor Nginx

Funcionando en puerto 8081

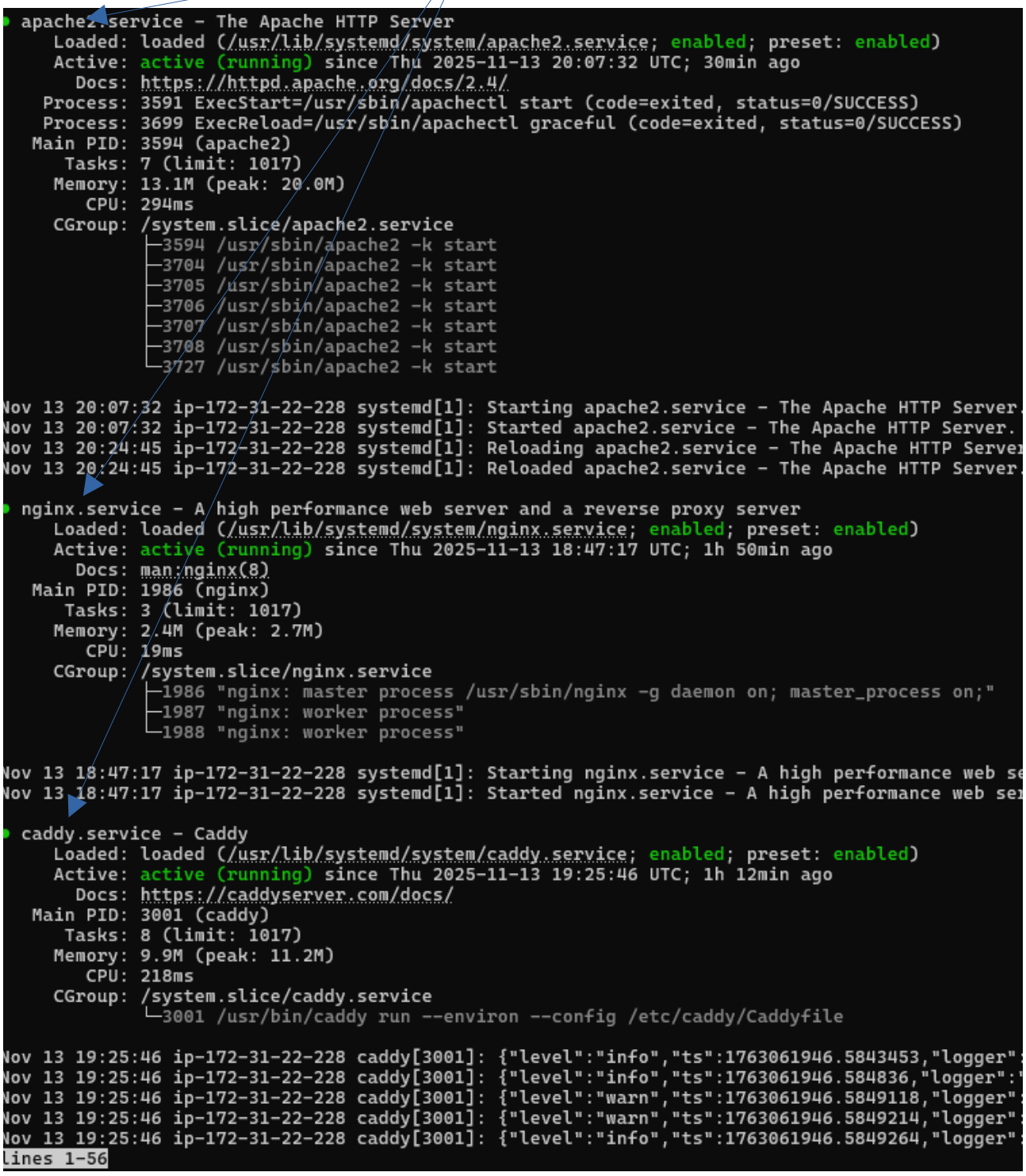
PARTE 5: VERIFICACIÓN FINAL DE LOS TRES SERVIDORES

1. Verificar que todos los servicios están activos

HAY QUE PONER EN EL FICHERO EL PUERTO 8443

Comando: sudo systemctl status apache2 nginx caddy

Descripción: Aquí básicamente he puesto los 3 servicios y se ven que estan activos y corriendo bien.



2. Verificar puertos en uso

Comando: sudo netstat -tulpn | grep -E '8443'
Descripción: Lista los puertos donde están escuchando los servidores.

```
ubuntu@ip-172-31-22-228:~$ sudo netstat -tulnp | grep -E '8443'
tcp6       0      0 :::8443                 :::*                    LISTEN      3594/apache2
```

3. Probar todos los servidores

Comandos: curl http://localhost:8080 curl http://localhost:8081 curl http://localhost:8082 curl -k https://localhost:8443
Descripción: Esto verifica que cada servidor responde correctamente en su puerto asignado, hay que hacer curls a todos los servidores y puertos

```
ubuntu@ip-172-31-22-228:~$ curl http://localhost:8080
<h1>Servidor Nginx</h1><p>Funcionando en puerto 8081</p>
```

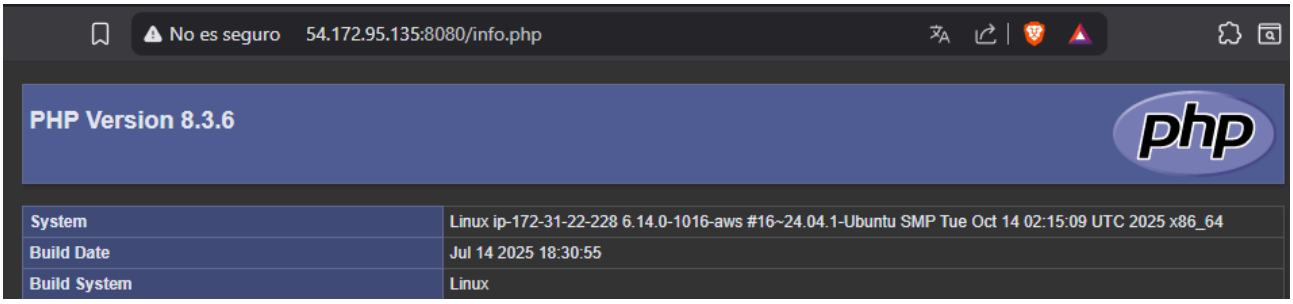
q

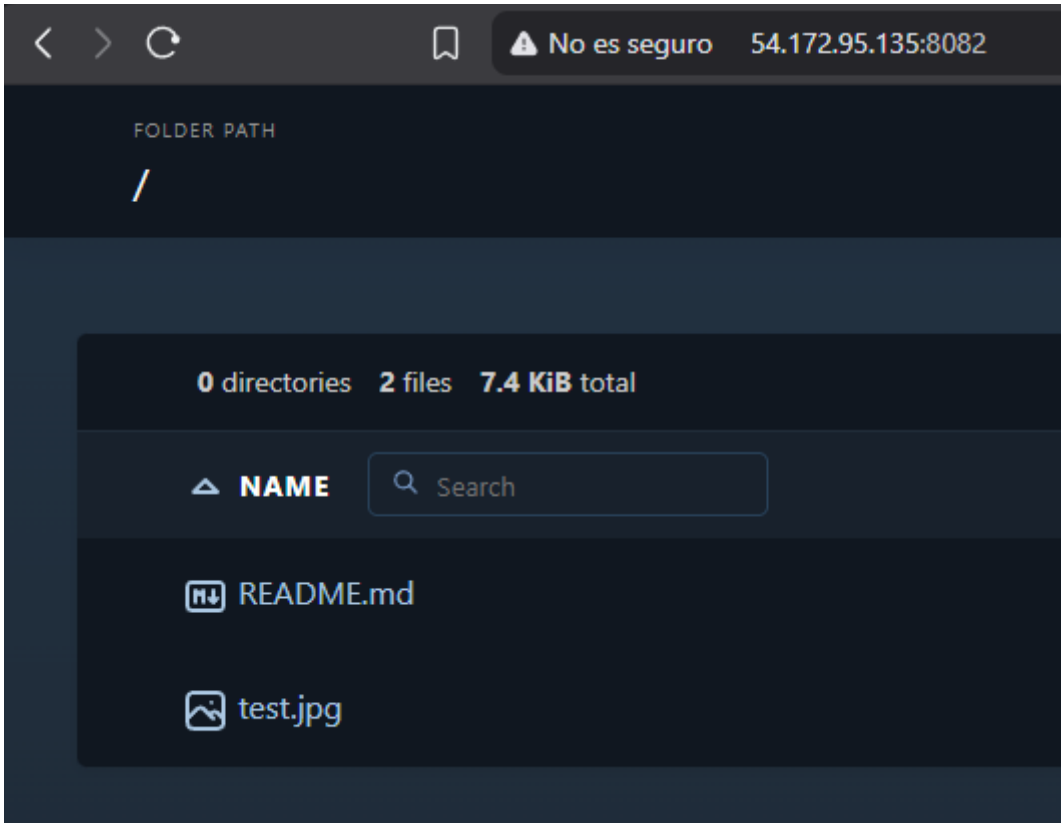
```
ubuntu@ip-172-31-22-228:~$ curl http://localhost:8081
<h1>Servidor Nginx</h1><p>Funcionando en puerto 8081</p>
```

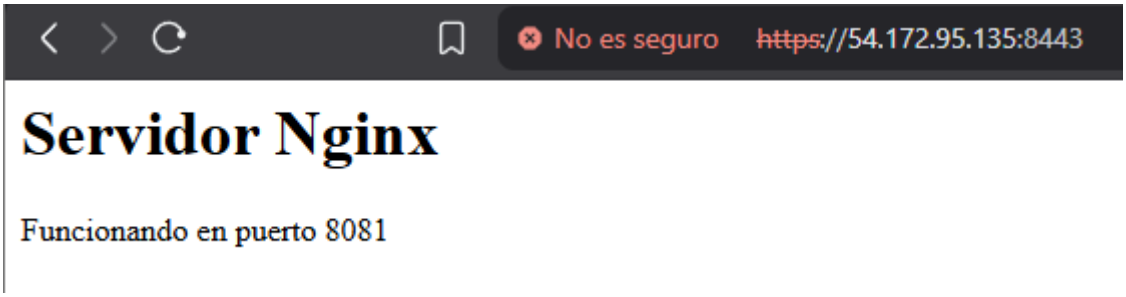
```
ubuntu@ip-172-31-22-228:~$ curl http://localhost:8082

<!DOCTYPE html>
<html>
```

```
ubuntu@ip-172-31-22-228:~$ curl -k https://localhost:8443
<h1>Servidor Nginx</h1><p>Funcionando en puerto 8081</p>
```




No es seguro
54.172.95.135:8082
FOLDER PATH
/
0 directories
2 files
7.4 KiB total
NAME
Search
README.md
test.jpg


No es seguro
https://54.172.95.135:8443
Servidor Nginx
Funcionando en puerto 8081