

PARTE 1: INSTALACIÓN Y CONFIGURACIÓN DE APACHE 1.

Adrián Aceitón Palomo

2ºASIR

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

```
alumno01@A6Alumno01:/mnt/c/Users/Alumno.DESKTOP-DI5KTUG$ sudo apt update && sudo apt upgrade -y
[sudo] password for alumno01:
Hit:1 http://security.ubuntu.com/ubuntu noble-security InRelease
Hit:2 http://archive.ubuntu.com/ubuntu noble InRelease
Get:3 http://archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Hit:4 http://archive.ubuntu.com/ubuntu noble-backports InRelease
Fetched 126 kB in 1s (129 kB/s)
Reading package lists... Done
Building dependency tree... Done
```

2. Instalar Apache2

Comando: sudo apt install apache2 -y

Descripción: Instala el servidor web Apache en tu sistema.

```
alumno01@A6Alumno01:/mnt/c/Users/Alumno.DESKTOP-DI5KTUG$ 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.

```

alumno01@A6Alumno01: /mn  X  +  v
GNU nano 7.2 /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.

```

GNU nano 7.2 /etc/apache2/sites-available/000-default.conf *
<VirtualHost *:8080>
  
```

5. Instalar PHP

Comando: `sudo apt install php libapache2-mod-php -y`

Descripción: Básicamente instalo PHP y ya.

```

alumno01@A6Alumno01:/mnt/c/Users/Alumno.DESKTOP-DISKUG$ sudo apt install php libapache2-mod-php -y
Reading package lists... Done
  
```

6. Reiniciar Apache

Comando: `sudo systemctl restart apache2`

Descripción: Reinicio de Apache para aplicar los cambios

```
alumno01@A6Alumno01:/mnt/c/Users/Alumno.DESKTOP-DI5KTUG$ sudo systemctl restart apache2
```

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`.

```
alumno01@A6Alumno01:/mnt/c/Users/Alumno.DESKTOP-DI5KTUG$ sudo apt install net-tools
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following package was automatically installed and is no longer required:
  libllvm19
Use 'sudo apt autoremove' to remove it.
The following NEW packages will be installed:
  net-tools
0 upgraded, 1 newly installed, 0 to remove and 12 not upgraded.
Need to get 204 kB of archives.
After this operation, 811 kB of additional disk space will be used.
Get:1 http://archive.ubuntu.com/ubuntu noble-updates/main amd64 net-tools amd64 2.10-0.1ubuntu4.4 [204
kB]
Fetched 204 kB in 0s (988 kB/s)
Selecting previously unselected package net-tools.
(Reading database ... 42208 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) ...
alumno01@A6Alumno01:/mnt/c/Users/Alumno.DESKTOP-DI5KTUG$ sudo netstat -tulpn | grep 8080
tcp6      0      0  ::::8080           :::*               LISTEN    5119/apache2
```

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.

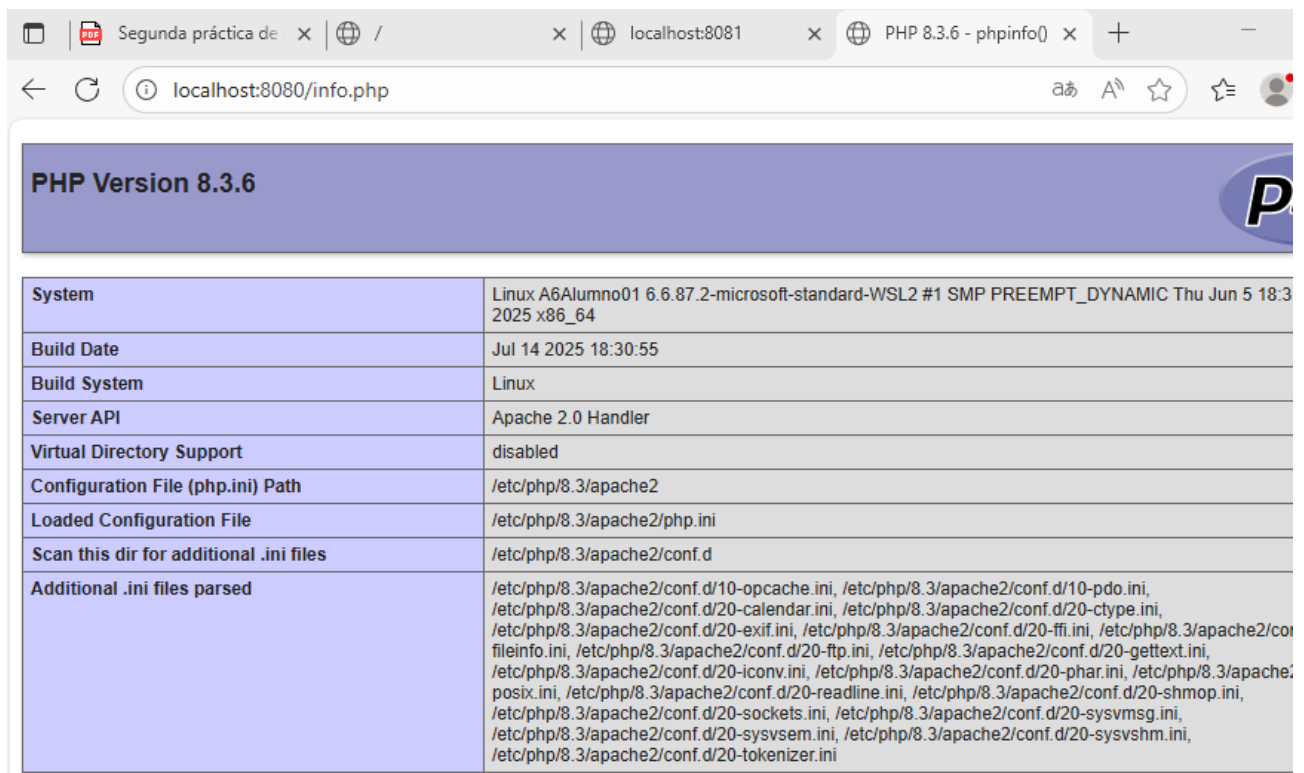
```
alumno01@A6Alumno01:/mnt/c/Users/Alumno.DESKTOP-DI5KTUG$ echo "<?php phpinfo(); ?>" | sudo tee /var/www/html/info.php
<?php phpinfo(); ?>
```


9. Probar Apache desde terminal

Comando: curl http://localhost:8080/info.php

Descripción: Esto verifica que Apache sirva correctamente el contenido PHP

```
alumno01@A6Alumno01: /mnt/c/Users/Alumno.DESKTOP-DI5KTUG$ curl http://localhost:8080/info.php
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "DTD/xhtml11-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;}
```



PARTE 2: INSTALACIÓN Y CONFIGURACIÓN DE NGINX

1. Instalar Nginx

Comando: `sudo apt install nginx -y`

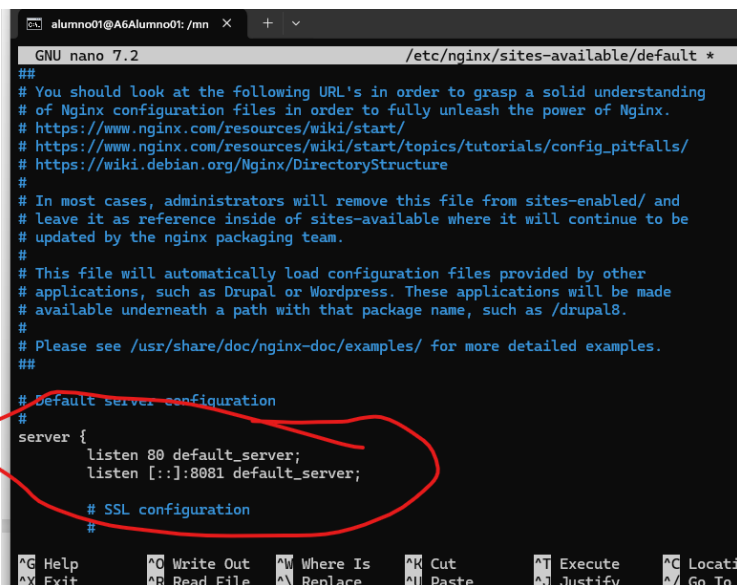
Descripción: Instalación del servidor web Nginx en tu sistema.

```
</div></body></html>alumno01@A6Alumno01:/mnt/c/Users/Alumno.DESKTOP-DISKUG$ sudo apt install nginx -y
Reading package lists... Done
Building dependency tree... Done
```

2. Configurar Nginx en puerto 8081

Comando: `sudo nano /etc/nginx/sites-available/default`

Descripción: Con este comando, entro a editar el archivo del sitio que Nginx carga por defecto. El objetivo es cambiar el puerto en el que Nginx "escucha". Para ello, busco la línea que dice `listen 80` (el puerto web estándar) y la modifico para que ponga `listen 8081`. Así, Nginx servirá esta web a través del nuevo puerto.



```
alumno01@A6Alumno01: /mnt  x  +  v
GNU nano 7.2 /etc/nginx/sites-available/default *
##
# You should look at the following URL's in order to grasp a solid understanding
# of Nginx configuration files in order to fully unleash the power of Nginx.
# https://www.nginx.com/resources/wiki/start/
# https://www.nginx.com/resources/wiki/start/topics/tutorials/config_pitfalls/
# https://wiki.debian.org/Nginx/DirectoryStructure
#
# In most cases, administrators will remove this file from sites-enabled/ and
# leave it as reference inside of sites-available where it will continue to be
# updated by the nginx packaging team.
#
# This file will automatically load configuration files provided by other
# applications, such as Drupal or Wordpress. These applications will be made
# available underneath a path with that package name, such as /drupal8.
#
# Please see /usr/share/doc/nginx-doc/examples/ for more detailed examples.
##

# Default server configuration
#
server {
    listen 80 default_server;
    listen [::]:8081 default_server;

    # SSL configuration
    #
```

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: Crea una página HTML identificable para Nginx.

```
alumno01@A6Alumno01:/mnt/c/Users/Alumno.DESKTOP-DI5KTUG$ echo "<h1>Servidor Nginx</h1><p>Funcionando en puerto 8081</p>" | sudo tee /var/www/html/index.html
<h1>Servidor Nginx</h1><p>Funcionando en puerto 8081</p>
alumno01@A6Alumno01:/mnt/c/Users/Alumno.DESKTOP-DI5KTUG$
```

4. Reiniciar Nginx

Comando: sudo systemctl restart nginx

Descripción: Aquí reinicio Nginx para aplicar los cambios de configuración.

```
alumno01@A6Alumno01:/mnt/c/Users/Alumno.DESKTOP-DI5KTUG$ sudo systemctl restart nginx
```

5. Verificar estado de Nginx

Comando: sudo systemctl status nginx

Descripción: Comprueba que Nginx está funcionando correctamente en el puerto 8081

```
alumno01@A6Alumno01:/mnt/c/Users/Alumno.DESKTOP-DI5KTUG$ sudo systemctl status nginx
● nginx.service - A high performance web server and a reverse proxy server
   Loaded: loaded (/usr/lib/systemd/system/nginx.service; enabled; preset: enabled)
   Active: active (running) since Tue 2025-10-14 09:04:07 CEST; 1min 5s ago
     Docs: man:nginx(8)
  Process: 2683 ExecStartPre=/usr/sbin/nginx -t -q -g daemon on; master_process on; (code=exited, status=0/SUCCESS)
  Process: 2685 ExecStart=/usr/sbin/nginx -g daemon on; master_process on; (code=exited, status=0/SUCCESS)
   Main PID: 2686 (nginx)
      Tasks: 17 (limit: 8250)
```

6. Probar Nginx desde terminal

Comando: curl http://localhost:8081

Descripción: Verifica que Nginx sirve correctamente el contenido HTML.

```
alumno01@A6Alumno01:/mnt/c/Users/Alumno.DESKTOP-DI5KTUG$ curl http://localhost:8081
<h1>Servidor Nginx</h1><p>Funcionando en puerto 8081</p>
```



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: Instala herramientas necesarias para añadir repositorios externos.

```
alumno01@A6Alumno01:/mnt/c/Users/Alumno.DESKTOP-DI5KTUG$ 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: Añade el repositorio oficial de Caddy a tu sistema para poder utilizarlo posteriormente.

```
alumno01@A6Alumno01:/mnt/c/Users/Alumno.DESKTOP-DI5KTUG$ curl -sLf 'https://dl.cloudsmith.io/public/caddy/stable/gpg.key'
| sudo gpg --dearmor -o /usr/share/keyrings/caddy-stable-archive-keyring.gpg
alumno01@A6Alumno01:/mnt/c/Users/Alumno.DESKTOP-DI5KTUG$
alumno01@A6Alumno01:/mnt/c/Users/Alumno.DESKTOP-DI5KTUG$ curl -sLf 'https://dl.cloudsmith.io/public/caddy/stable/debian.d
eb.txt' | sudo tee /etc/apt/sources.list.d/caddy-stable.list
# Source: Caddy
# Site: https://github.com/caddyserver/caddy
# Repository: Caddy / stable
```

3. Actualizar e instalar Caddy

Comando: sudo apt update && sudo apt install caddy -y

Descripción: Actualiza la lista de paquetes e instala Caddy.

```
alumno01@A6Alumno01:/mnt/c/Users/Alumno.DESKTOP-DI5KTUG$ sudo apt update && sudo apt install caddy -y
Hit:1 http://security.ubuntu.com/ubuntu noble-security InRelease
Hit:2 http://archive.ubuntu.com/ubuntu noble InRelease
Hit:3 http://archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:4 http://archive.ubuntu.com/ubuntu noble-backports InRelease
Get:5 https://dl.cloudsmith.io/public/caddy/stable/deb/debian any-version InRelease [14.8 kB]
Get:6 https://dl.cloudsmith.io/public/caddy/stable/deb/debian any-version/main amd64 Packages [4329 B]
Fetched 19.1 kB in 1s (24.6 kB/s)
Reading package lists... Done
```

4. Crear directorio para Caddy

Comando: sudo mkdir -p /var/www/caddy

Descripción: Crea un directorio específico para los archivos de Caddy

```
alumno01@A6Alumno01:/mnt/c/Users/Alumno.DESKTOP-DI5KTUG$ sudo mkdir -p /var/www/caddy
```

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: Con estos comandos, genero un archivo README.md y le añado contenido de ejemplo con formato Markdown (un título, un párrafo y una lista). El objetivo es tener un fichero de prueba en la carpeta web de Caddy. Así, cuando probemos el servidor, veremos este contenido y sabremos que está funcionando.

```
alumno01@A6Alumno01:/mnt/c/Users/Alumno.DESKTOP-DI5KTUG$ echo "# Bienvenido a Caddy" | sudo tee /var/www/caddy/README.md
# Bienvenido a Caddy
alumno01@A6Alumno01:/mnt/c/Users/Alumno.DESKTOP-DI5KTUG$ echo "# Bienvenido a Caddy" | sudo tee /var/www/caddy/README.md
# Bienvenido a Caddy

alumno01@A6Alumno01:/mnt/c/Users/Alumno.DESKTOP-DI5KTUG$ echo "Este servidor está funcionando correctamente." | sudo tee -a /var/www/caddy/README.md
Este servidor está funcionando correctamente.

alumno01@A6Alumno01:/mnt/c/Users/Alumno.DESKTOP-DI5KTUG$ echo "" | sudo tee -a /var/www/caddy/README.md

alumno01@A6Alumno01:/mnt/c/Users/Alumno.DESKTOP-DI5KTUG$ echo "## Características" | sudo tee -a /var/www/caddy/README.md
## Características
alumno01@A6Alumno01:/mnt/c/Users/Alumno.DESKTOP-DI5KTUG$ echo "- Servidor moderno" | sudo tee -a /var/www/caddy/README.md
- Servidor moderno
alumno01@A6Alumno01:/mnt/c/Users/Alumno.DESKTOP-DI5KTUG$ echo "- HTTPS automático" | sudo tee -a /var/www/caddy/README.md
- HTTPS automático
alumno01@A6Alumno01:/mnt/c/Users/Alumno.DESKTOP-DI5KTUG$ echo "- Fácil configuración" | sudo tee -a /var/www/caddy/README.md
- Fácil configuración
alumno01@A6Alumno01:/mnt/c/Users/Alumno.DESKTOP-DI5KTUG$
```

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

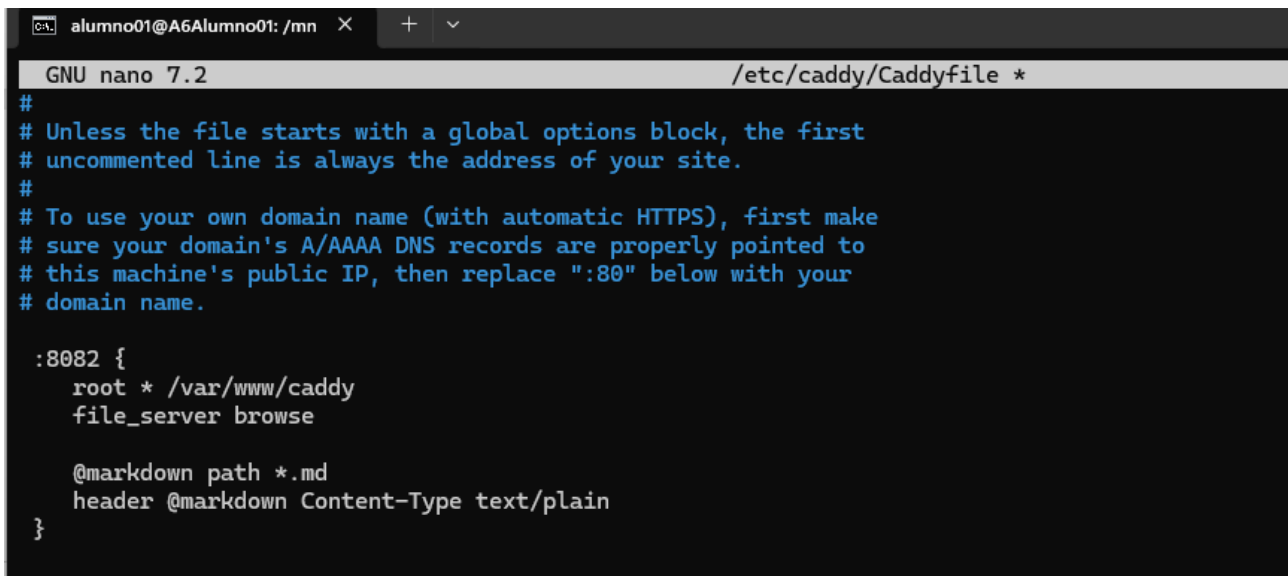
Descripción: Descargo una imagen de prueba usando curl y la muevo al directorio web de Caddy. El objetivo es tener un archivo estático para verificar que Caddy puede servirlo correctamente. Es importante ejecutar primero los comandos que se ven en la captura (mkdir y chmod) para asegurar que la carpeta /var/www/caddy esté creada y tenga los permisos necesarios.

```
alumno01@A6Alumno01:/mnt/c/Users/Alumno.DESKTOP-DI5KTUG$ sudo mkdir -p /var/www/caddy
alumno01@A6Alumno01:/mnt/c/Users/Alumno.DESKTOP-DI5KTUG$ sudo chmod -R 755 /var/www/caddy
alumno01@A6Alumno01:/mnt/c/Users/Alumno.DESKTOP-DI5KTUG$ curl -o /tmp/test-image.jpg "https://www.python.org/static/apple-touch-icon-144x144-precomposed.png"
% Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left   Speed
100 7382 100 7382    0     0  200k    0 --:--:-- --:--:-- --:--:-- 205k
alumno01@A6Alumno01:/mnt/c/Users/Alumno.DESKTOP-DI5KTUG$ sudo mv /tmp/test-image.jpg /var/www/caddy/test.jpg
alumno01@A6Alumno01:/mnt/c/Users/Alumno.DESKTOP-DI5KTUG$
```

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.



```
alumno01@A6Alumno01: /mn  X  +  v
GNU nano 7.2 /etc/caddy/Caddyfile *
#
# Unless the file starts with a global options block, the first
# uncommented line is always the address of your site.
#
# To use your own domain name (with automatic HTTPS), first make
# sure your domain's A/AAAA DNS records are properly pointed to
# this machine's public IP, then replace ":80" below with your
# domain name.

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

    @markdown path *.md
    header @markdown Content-Type text/plain
}
```

7, 8, 9, 10, 11.

Primeramente reiniciamos el servicio caddy y vemos que funciona correctamente, despues de esto comprobamos con el status que esté activo y funcionando. Luego mostramos un curl para ver si funciona en consola, y tambien abrimos en el navegador con localhost para ver si funciona, ademas nos adentramos en las subcarpetas para encontrar el archivo README markdown que creamos previamente y tambien la imagen que descargamos también previamente:

```
alumno01@A6Alumno01:/mnt/c/Users/Alumno.DESKTOP-DI5KTUG$ sudo systemctl restart caddy
alumno01@A6Alumno01:/mnt/c/Users/Alumno.DESKTOP-DI5KTUG$ sudo systemctl status caddy
● caddy.service - Caddy
   Loaded: loaded (/usr/lib/systemd/system/caddy.service; enabled; preset: enabled)
   Active: active (running) since Tue 2025-10-14 14:14:25 CEST; 4s ago
     Docs: https://caddyserver.com/docs/
  Main PID: 888 (caddy)
    Tasks: 12 (limit: 9350)
   Memory: 11.8M (peak: 12.4M)
      CPU: 60ms
   CGroup: /system.slice/caddy.service
           └─888 /usr/bin/caddy run --environ --config /etc/caddy/Caddyfile


Oct 14 14:14:25 A6Alumno01 caddy[888]: {"level":"info","ts":1760444065.5774353,"logger":"admin","msg":>
Oct 14 14:14:25 A6Alumno01 caddy[888]: {"level":"info","ts":1760444065.5777967,"logger":"tls.cache.ma>
Oct 14 14:14:25 A6Alumno01 caddy[888]: {"level":"warn","ts":1760444065.5778072,"logger":"http","msg":>
Oct 14 14:14:25 A6Alumno01 caddy[888]: {"level":"warn","ts":1760444065.5778415,"logger":"http","msg":>
Oct 14 14:14:25 A6Alumno01 caddy[888]: {"level":"info","ts":1760444065.577845,"logger":"http.log","ms>
Oct 14 14:14:25 A6Alumno01 caddy[888]: {"level":"info","ts":1760444065.5795205,"msg":"autosaved confi>
Oct 14 14:14:25 A6Alumno01 caddy[888]: {"level":"info","ts":1760444065.5795996,"msg":"serving initial>
Oct 14 14:14:25 A6Alumno01 systemd[1]: Started caddy.service - Caddy.
Oct 14 14:14:25 A6Alumno01 caddy[888]: {"level":"info","ts":1760444065.5898697,"logger":"tls","msg":>
Oct 14 14:14:25 A6Alumno01 caddy[888]: {"level":"info","ts":1760444065.5914307,"logger":"tls","msg":>
alumno01@A6Alumno01:/mnt/c/Users/Alumno.DESKTOP-DI5KTUG$
```

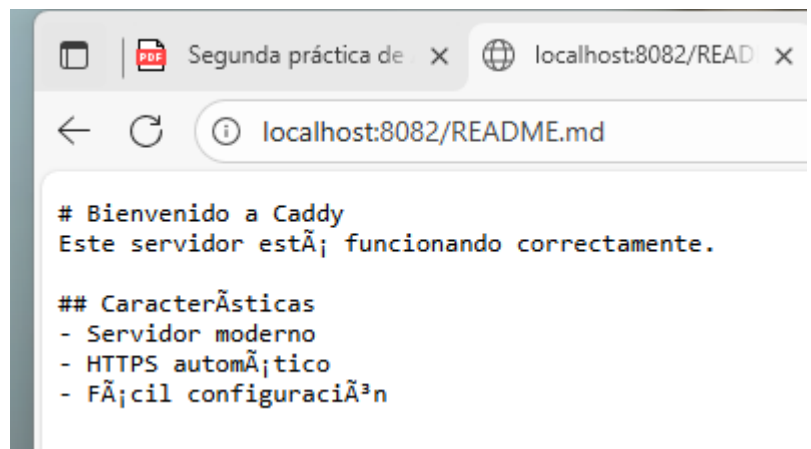
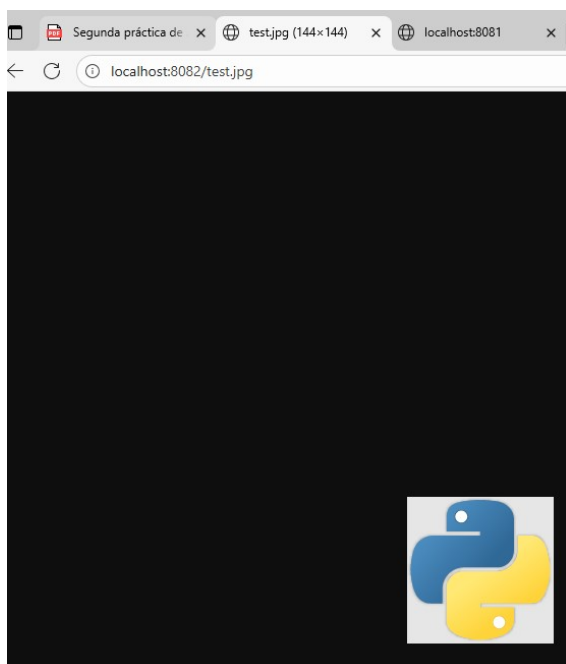
```
alumno01@A6Alumno01:/mnt/c/Users/Alumno.DESKTOP-DI5KTUG$ curl http://localhost:8082/README.md
# Bienvenido a Caddy
Este servidor está funcionando correctamente.

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

Activar Windows
Ve a Configuración para activar Windows.

0 directories 2 files 7.4 KiB total			List	Grid
NAME	SIZE	MODIFIED		
README.md	152 B	14/10/2025, 9:23:41		
test.jpg	7.2 KiB	14/10/2025, 9:29:35		

Served with  caddy



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: Instala Certbot y su integración con Apache para gestionar certificados SSL.

```
alumno01@A6Alumno01:/mnt/c/Users/Alumno.DESKTOP-DI5KTUG$ sudo apt install certbot python3-certbot-apache -y
[sudo] password for alumno01:
Reading package lists... 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: Crea un certificado autofirmado para practicar HTTPS localmente. Completa los campos solicitados (puedes usar valores por defecto)

Además metemos la información que se ve en la captura

[illegible]

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.


```
alumno01@A6Alumno01:/mnt/c/Users/Alumno.DESKTOP-DI5KTUG$ sudo systemctl restart apache2
alumno01@A6Alumno01:/mnt/c/Users/Alumno.DESKTOP-DI5KTUG$ sudo a2enmod ssl
Considering dependency mime for ssl:
Module mime already enabled
Considering dependency socache_shmcb for ssl:
Module socache_shmcb already enabled
Module ssl already enabled
```

Activar Windows
Ve a Configuración para activar

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).

```
alumno01@A6Alumno01:/mnt/c/Users/Alumno.DESKTOP-DI5KTUG$ sudo a2enmod ssl
Considering dependency mime for ssl:
Module mime already enabled
Considering dependency socache_shmcb for ssl:
Module socache_shmcb already enabled
Module ssl already enabled
alumno01@A6Alumno01:/mnt/c/Users/Alumno.DESKTOP-DI5KTUG$ sudo nano /etc/apache2/sites-available/default-ssl.conf
alumno01@A6Alumno01:/mnt/c/Users/Alumno.DESKTOP-DI5KTUG$ sudo nano /etc/apache2/ports.conf
alumno01@A6Alumno01:/mnt/c/Users/Alumno.DESKTOP-DI5KTUG$ sudo a2ensite default-ssl.conf
Enabling site default-ssl.
To activate the new configuration, you need to run:
systemctl reload apache2
alumno01@A6Alumno01:/mnt/c/Users/Alumno.DESKTOP-DI5KTUG$ sudo systemctl restart apache2
alumno01@A6Alumno01:/mnt/c/Users/Alumno.DESKTOP-DI5KTUG$ sudo netstat -tulpn | grep -E '8443'
tcp6      0      0 :::8443          :::*              LISTEN     1167/apache2
alumno01@A6Alumno01:/mnt/c/Users/Alumno.DESKTOP-DI5KTUG$ curl -i -k https://localhost:8443
HTTP/1.1 200 OK
Date: Thu, 16 Oct 2025 12:09:46 GMT
Server: Apache/2.4.58 (Ubuntu)
Last-Modified: Tue, 14 Oct 2025 07:01:02 GMT
ETag: "39-64118f2676028"
Accept-Ranges: bytes
Content-Length: 57
Content-Type: text/html

<h1>Servidor Nginx</h1><p>Funcionando en puerto 8081</p>
alumno01@A6Alumno01:/mnt/c/Users/Alumno.DESKTOP-DI5KTUG$
```

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

```
<h1>Servidor Nginx</h1><p>Funcionando en puerto 8081</p>
alumno01@A6Alumno01:/mnt/c/Users/Alumno.DESKTOP-DI5KTUG$ sudo systemctl status apache2 nginx caddy
[sudo] password for alumno01:
● apache2.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/apache2.service; enabled; preset: enabled)
   Active: active (running) since Thu 2025-10-16 14:09:16 CEST; 21min ago
     Docs: https://httpd.apache.org/docs/2.4/
   Process: 1164 ExecStart=/usr/sbin/apachectl start (code=exited, status=0/SUCCESS)
    Main PID: 1167 (apache2)
       Tasks: 6 (limit: 9350)
      Memory: 12.1M (peak: 12.9M)
         CPU: 181ms
    CGroup: /system.slice/apache2.service
            └─1167 /usr/sbin/apache2 -k start
              1169 /usr/sbin/apache2 -k start
              1170 /usr/sbin/apache2 -k start
              1171 /usr/sbin/apache2 -k start
              1172 /usr/sbin/apache2 -k start
              1173 /usr/sbin/apache2 -k start

Oct 16 14:09:16 A6Alumno01 systemd[1]: Starting apache2.service - The Apache HTTP Server...
Oct 16 14:09:16 A6Alumno01 systemd[1]: Started apache2.service - The Apache HTTP Server.

● nginx.service - A high performance web server and a reverse proxy server
   Loaded: loaded (/usr/lib/systemd/system/nginx.service; enabled; preset: enabled)
   Active: active (running) since Thu 2025-10-16 13:51:42 CEST; 38min ago
     Docs: man:nginx(8)
   Process: 168 ExecStartPre=/usr/sbin/nginx -t -q -g daemon on; master_process on; (code=exited, status=0/SUCCESS)
   Process: 224 ExecStart=/usr/sbin/nginx -g daemon on; master_process on; (code=exited, status=0/SUCCESS)
    Main PID: 241 (nginx)
       Tasks: 17 (limit: 9350)
      Memory: 12.2M (peak: 15.2M)
         CPU: 52ms
    CGroup: /system.slice/nginx.service
            └─241 "nginx: master process /usr/sbin/nginx -g daemon on; master_process on;"
              243 "nginx: worker process"
```

Descripción:
Muestra el estado de los tres servidores simultáneamente.

2. Verificar puertos en uso

Comando: `sudo netstat -tulpn | grep -E '8443'`

Descripción: Lista los puertos donde están escuchando los servidores.

```
alumno01@A6Alumno01:/mnt/c/Users/Alumno.DESKTOP-DI5KTUG$ sudo netstat -tulpn | grep -E '8443'
tcp6      0      0 :::8443          :::*              LISTEN      1167/apache2
alumno01@A6Alumno01:/mnt/c/Users/Alumno.DESKTOP-DI5KTUG$
```

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

```
alumno01@A6Alumno01:/mnt/c/Users/Alumno.DESKTOP-DI5KTUG$ curl -k https://localhost:8443
<h1>Servidor Nginx</h1><p>Funcionando en puerto 8081</p>
```

```
alumno01@A6Alumno01:/mnt/c/Users/Alumno.DESKTOP-DI5KTUG$ curl http://localhost:8080
<h1>Servidor Nginx</h1><p>Funcionando en puerto 8081</p>
alumno01@A6Alumno01:/mnt/c/Users/Alumno.DESKTOP-DI5KTUG$ curl http://localhost:8081
<h1>Servidor Nginx</h1><p>Funcionando en puerto 8081</p>
alumno01@A6Alumno01:/mnt/c/Users/Alumno.DESKTOP-DI5KTUG$ curl http://localhost:8082

<!DOCTYPE html>
<html>
  <head>
    <title></title>
```

