

2023

DESPLIEGUE DE APLICACIONES WEB

GUÍA DE RESOLUCIÓN DE LA TAREA 01

Contenido

Ejercicio 1 Arquitecturas web.	3
Ejercicio 2 Clasificación de las Aplicaciones web.	5
Ejercicio 3. Instalando nuestro servidor web.	7
3.1. Instalar el servidor web Apache desde terminal de comandos.	7
3.2. Arrancar, reiniciar, comprobar el estado y parar el servidor web Apache.	8
3.3. Comprobar que está funcionando el servidor Apache desde un navegador web.	9
3.4. Cambiar el puerto por el cual está escuchando Apache pasándolo al puerto 8088 y comprueba de nuevo desde tu navegador que está funcionando.	11
3.5. Cambiar la página web por defecto para que aparezcan tus apellidos, más un pantallazo de tu inicio del curso.	12
Ejercicio 4. Haciendo las comunicaciones más seguras.	14
Ejercicio 5. Instalando un servidor de aplicaciones.	18
Ejercicio 6. Hosts virtuales	22

Ejercicio 1 Arquitecturas web.

Una plataforma o arquitectura web es el entorno empleado para diseñar, desarrollar y ejecutar un sitio web. Hay muchos modelos de plataformas y una de las piezas más importantes y determinantes a la hora de elegirla es el sistema operativo. Describe dos plataformas web, una basada en Linux y otra basada en Windows, indicando qué tecnologías utilizan para cada una de las capas, es decir, para cubrir aspectos como el servicio web (HTTP), el contenido dinámico, o el acceso a datos.

[Esta actividad está basada en el apartado 2.5 de la unidad]

Una plataforma web es el entorno de desarrollo de software empleado para diseñar y ejecutar un sitio web. En términos generales, una plataforma web consta de cuatro componentes básicos:

1. El **sistema operativo**, bajo el cual opera el equipo donde se hospedan las páginas web y que representa la base misma del funcionamiento del computador. En ocasiones limita la elección de otros componentes.
2. El **servidor web** es el software que maneja las peticiones desde equipos remotos a través de la Internet. En el caso de páginas estáticas, el servidor web simplemente provee el archivo solicitado, el cual se muestra en el navegador. En el caso de sitios dinámicos, el servidor web se encarga de pasar las solicitudes a otros programas que puedan gestionarlas adecuadamente.
3. El **gestor de bases de datos** se encarga de almacenar sistemáticamente un conjunto de registros de datos relacionados para ser usados posteriormente.
4. Un **lenguaje de programación interpretado** que controla las aplicaciones de software que corren en el sitio web.

Diferentes combinaciones de los cuatro componentes señalados, basadas en las distintas opciones de software disponibles en el mercado, dan lugar a numerosas plataformas web, aunque, sin duda, hay dos que sobresalen del resto por su popularidad y difusión: LAMP y WISA.

La plataforma LAMP trabaja enteramente con componentes de **software libre** y no está sujeta a restricciones propietarias. El nombre **LAMP** surge de las iniciales de los componentes de software que la integran:

- **Linux**: sistema operativo.
- **Apache**: servidor web.
- **MySQL**: gestor de bases de datos.
- **PHP**: lenguaje interpretado PHP, aunque a veces se sustituye por Perl o Python.

La plataforma **WISA** está basada en tecnologías desarrolladas por la compañía Microsoft; se trata, por lo tanto, de **software propietario**. La componen los siguientes elementos:

- **Windows**: sistema operativo.
- **Internet Information Services (IIS)**: servidor web.
- **SQL Server**: gestor de bases de datos.
- **ASP o ASP.NET**: como lenguaje para scripting del lado del servidor.

Existen otras plataformas, como por ejemplo la configuración Windows-Apache-MySQLPHP que se conoce como **WAMP**. Es bastante común pero sólo como plataforma de desarrollo local.

De forma similar, un servidor Windows puede correr con IIS, y con MySQL y PHP. A esta configuración se la conoce como plataforma **WIMP**.

Entre los servidores web más utilizados hoy en día, aparte de Apache e IIS tenemos también a **Nginx**, un servidor web software libre que está demostrando un alto rendimiento, y que es capaz de atender una gran cantidad de peticiones simultáneas, y que según las estadísticas tiene un mejor rendimiento que sus competidores al servir contenido estático. En su contra, es un servidor cuya configuración es menos flexible que otros.

Nginx también puede integrarse con PHP, por lo que podemos encontrar plataformas tipo LNMP, resultado de integrar Linux con Nginx, MySQL o [MariaDB](#) y PHP. Esta opción es también posible en Windows, dando lugar a plataformas WNMP (Windows + Nginx + MySQL o MariaDB + PHP).

Existen muchas otras plataformas que trabajan con distintos sistemas operativos (Unix, MacOS, Solaris), servidores web (incluyendo algunos que se han cobrado relativa popularidad como Lighttpd y LiteSpeed), bases de datos (MariaDB, [PostgreSQL](#), [MongoDB](#)) y otros lenguajes de programación.

Ejercicio 2 Clasificación de las Aplicaciones web.

Enumera y explica brevemente cada una de las diferentes tecnologías asociadas a las aplicaciones web que se ejecutarán tanto del lado del servidor como del cliente, especificando lo que corresponde a cada uno de los casos. [Este ejercicio está basado en el apartado 2.2. de la unidad].

Las aplicaciones web emplean páginas dinámicas, éstas se ejecutan en un servidor web y se muestran en el navegador de un equipo cliente que es el que ha realizado previamente la solicitud. Cuando una página web llega al navegador, es posible que también incluya algún programa o fragmento de código que se deba ejecutar. Ese código, normalmente en lenguaje JavaScript, **lo ejecutará el propio navegador**. Es por ello que en este apartado nos centraremos en las tecnologías asociadas a las aplicaciones web que se ejecutarán tanto del lado del servidor como del cliente, especificando lo que corresponda en cada uno de los casos.

Tecnologías en el lado del servidor:

- **CGI (Common Gateway Interface):** la "Interface Común de Entrada" es uno de los estándares más antiguos relacionado con las aplicaciones web. Está pensado para permitir que un cliente web pueda acceder a un programa que se ejecuta en un servidor web, donde este generará contenido dinámico. **Este estándar** es utilizado para acceder a información almacenada en bases de datos, para implementar motores de búsqueda, gestionar datos de formularios, generar emails de forma automática, foros, comercio electrónico, juegos en línea, etc. Las rutinas de CGI son habitualmente escritas en lenguajes interpretados como Perl o por lenguajes compilados como C.
- **ASP (Active Server Pages):** las "Páginas Activas" se ejecutan del lado del servidor, de este modo se forman los resultados que luego se mostrarán en el navegador de cada equipo cliente en virtud de la petición que se ha realizado. Un buen ejemplo de ello son los buscadores, donde un usuario realiza una petición de información y el servidor nos entrega un resultado a medida de nuestra petición. Existen versiones de ASP para Unix y Linux, a pesar de que fue una tecnología desarrollada por Microsoft para la creación dinámica de páginas web ofrecida junto a su servidor IIS. Hoy en día ASP ha evolucionado hasta [ASP.NET](#), y se ha convertido en un framework de desarrollo web libre.
- **PHP (Hypertext Preprocessor):** este lenguaje es, al igual que ASP, ejecutado en el lado del servidor. PHP es similar a ASP y puede ser usado en circunstancias similares. Es muy eficiente, permitiendo el acceso a bases de datos empleando servidores como [MySQL](#) y, por lo tanto, suele utilizarse para crear páginas dinámicas complejas.
- **Java:** en el ecosistema de Java existe un conjunto de tecnologías pensadas para desarrollo de aplicaciones web que se ejecutan en el lado del servidor, generalmente enmarcadas en lo que se conoce como Java EE (Java Enterprise Edition). Las tecnologías más conocidas en el entorno web de Java son **JSP** (JavaServer Pages), **JSF** (JavaServer Faces) y los **servlets**. JSP es conceptualmente similar a PHP y ASP, pero con el trasfondo de java.
- **JavaScript:** dentro de la idea de "full stack development", en la que se persigue que un mismo desarrollador o desarrolladora pueda implementar tanto la parte que se ejecuta en el servidor (back-end) como la página que se visualiza en el cliente (front-end), han surgido cada vez más tecnologías que permiten desarrollar en JavaScript aplicaciones web que se ejecutan en el servidor (la misma usada en el navegador web), como es el caso de Express.js

y Hapi.js (ambos funcionan sobre Node.js), y Meteor.js. Pero esta idea no es nueva, ya en 1995 el servidor web Netscape Enterprise Server ya contemplaba el uso de javascript en el servidor (server-side javascript).

Tecnologías en el lado del cliente:

- **HTML (HiperText Markup Language):** como es de suponer, ya sabrás que HTML es el lenguaje de marcas que se utiliza maquetar el contenido de una página web. Hoy día va por la versión 5 (HTML5).
- **CSS (Cascading Style Sheets):** las "Hojas de Estilo en Cascada" se usan para formatear las páginas web; se trata de separar el contenido de un documento de su presentación. Cualquier cambio en el estilo marcado para un elemento en la CSS afectará a todas las páginas vinculadas a esa CSS.
- **Java:** es un lenguaje que también podemos encontrarlo en el lado del cliente, ejecutándose de forma incrustada en un navegador web. A las aplicaciones Java que se pueden embeber en una página web se las conoce como "applets".
- **JavaScript:** posiblemente el uso más extendido de este lenguaje es hacer las páginas web más interactivas. Es un lenguaje que como ya se introdujo antes, puede interpretarse y ejecutarse en un navegador web. Es útil para realizar tareas tales como mover imágenes por la pantalla, crear menús de navegación interactivos, juegos, etc. En las páginas web suele preferirse JavaScript porque es aceptado por muchos más navegadores que VBScript (creado por Microsoft).
- **VBScript (Visual Basic Scripting):** fue la respuesta de Microsoft a JavaScript. VBScript es un lenguaje que ha entrado en desuso, dado que solo se podía usar casi exclusivamente en el navegador Microsoft Internet Explorer. El código en VBScript puede, además, estar diseñado para su ejecución en el lado del cliente o en el del servidor, la diferencia es que un código que se ejecuta en el lado del servidor no es visible en el lado del cliente. Éste recibe los resultados, pero no el código.

Ejercicio 3. Instalando nuestro servidor web.

Dispones de una máquina (o máquina virtual) que cuenta con el sistema operativo Ubuntu **20/22** recientemente actualizado, con el entorno de red configurado y conexión a Internet. Además, estás trabajando con la cuenta del usuario root. Indica cada uno de los pasos, y comandos implicados en ellos, para conseguir hacer lo siguiente:

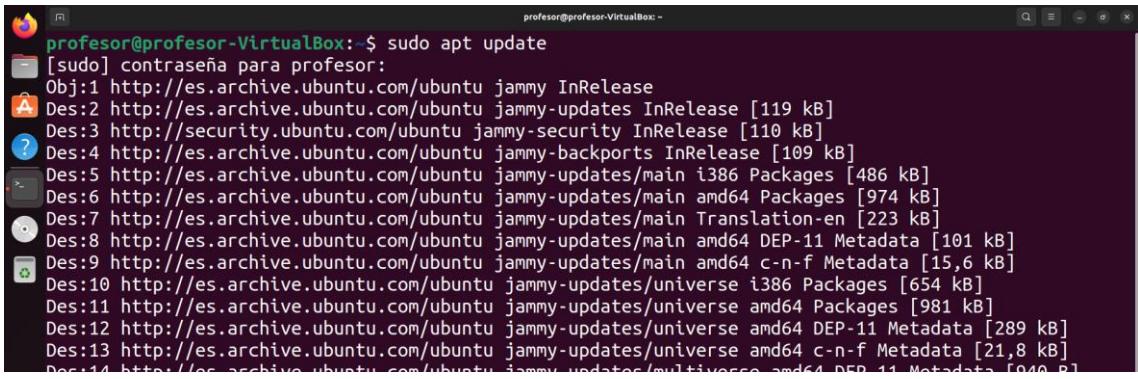
(**Recuerda que debes hacer las capturas de pantalla necesarias, que sean legibles, indicativas de que has realizado lo que se pide, que demuestren que todo ha funcionado correctamente, y en las que aparezca tu perfil de la plataforma de enseñanza**)

- Instalar el servidor web Apache desde terminal de comandos.
- Arrancar, reiniciar, comprobar el estado y parar el servidor web Apache.
- Comprobar que está funcionando el servidor Apache desde un navegador web.
- Cambiar el puerto por el cual está escuchando Apache pasándolo al puerto 8088 y comprueba de nuevo desde tu navegador que está funcionando.
- Cambiar la página web por defecto para que aparezcan **tus apellidos más un pantallazo de tu inicio del curso**.

3.1. Instalar el servidor web Apache desde terminal de comandos.

Vamos a instalar el servidor web Apache en Ubuntu 22.04 LTS Jammy Jellyfish desde los repositorios de la propia distribución, por lo que los actualizamos:

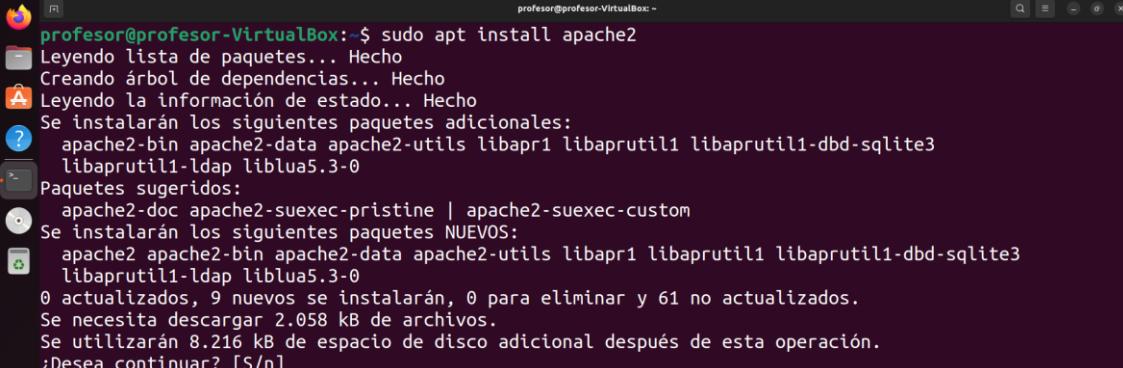
\$ apt update ó apt-get update



```
profesor@profesor-VirtualBox:~$ sudo apt update
[sudo] contraseña para profesor:
Obj:1 http://es.archive.ubuntu.com/ubuntu jammy InRelease
Des:2 http://es.archive.ubuntu.com/ubuntu jammy-updates InRelease [119 kB]
Des:3 http://security.ubuntu.com/ubuntu jammy-security InRelease [110 kB]
Des:4 http://es.archive.ubuntu.com/ubuntu jammy-backports InRelease [109 kB]
Des:5 http://es.archive.ubuntu.com/ubuntu jammy-updates/main i386 Packages [486 kB]
Des:6 http://es.archive.ubuntu.com/ubuntu jammy-updates/main amd64 Packages [974 kB]
Des:7 http://es.archive.ubuntu.com/ubuntu jammy-updates/main Translation-en [223 kB]
Des:8 http://es.archive.ubuntu.com/ubuntu jammy-updates/main amd64 DEP-11 Metadata [101 kB]
Des:9 http://es.archive.ubuntu.com/ubuntu jammy-updates/main amd64 c-n-f Metadata [15,6 kB]
Des:10 http://es.archive.ubuntu.com/ubuntu jammy-updates/universe i386 Packages [654 kB]
Des:11 http://es.archive.ubuntu.com/ubuntu jammy-updates/universe amd64 Packages [981 kB]
Des:12 http://es.archive.ubuntu.com/ubuntu jammy-updates/universe amd64 DEP-11 Metadata [289 kB]
Des:13 http://es.archive.ubuntu.com/ubuntu jammy-updates/universe amd64 c-n-f Metadata [21,8 kB]
Des:14 http://es.archive.ubuntu.com/ubuntu jammy-updates/multiverse amd64 DEP-11 Metadata [940 kB]
```

Una vez actualizado el repositorio, procedemos a realizar la instalación del paquete de Apache Web Server:

\$ apt install apache2 ó apt-get install apache2



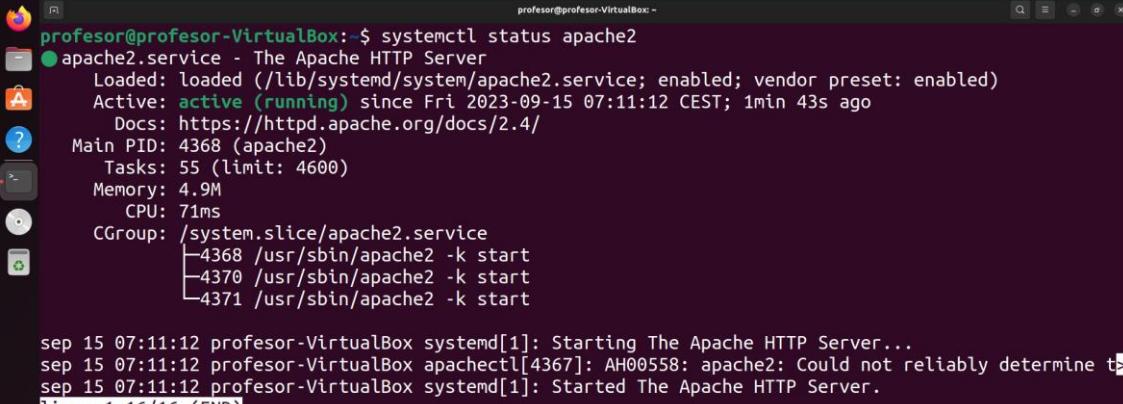
```
profesor@profesor-VirtualBox:~$ sudo apt install apache2
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias... Hecho
Leyendo la información de estado... Hecho
Se instalarán los siguientes paquetes adicionales:
  apache2-bin apache2-data apache2-utils libapr1 libaprutil1 libaprutil1-dbd-sqlite3
  libaprutil1-ldap liblua5.3-0
Paquetes sugeridos:
  apache2-doc apache2-suexec-pristine | apache2-suexec-custom
Se instalarán los siguientes paquetes NUEVOS:
  apache2 apache2-bin apache2-data apache2-utils libapr1 libaprutil1 libaprutil1-dbd-sqlite3
  libaprutil1-ldap liblua5.3-0
0 actualizados, 9 nuevos se instalarán, 0 para eliminar y 61 no actualizados.
Se necesita descargar 2.058 kB de archivos.
Se utilizarán 8.216 kB de espacio de disco adicional después de esta operación.
¿Desea continuar? [S/n]
```

3.2. Arrancar, reiniciar, comprobar el estado y parar el servidor web Apache.

Tras la descarga e instalación de este paquete y sus dependencias se crea un nuevo servicio en Ubuntu 22.04 LTS, el servicio apache2 o apache2.service que queda iniciado y habilitado para su arranque automático junto al sistema.

Para comprobar si ya se ha iniciado:

\$ systemctl status apache2 ó systemctl status apache2.service

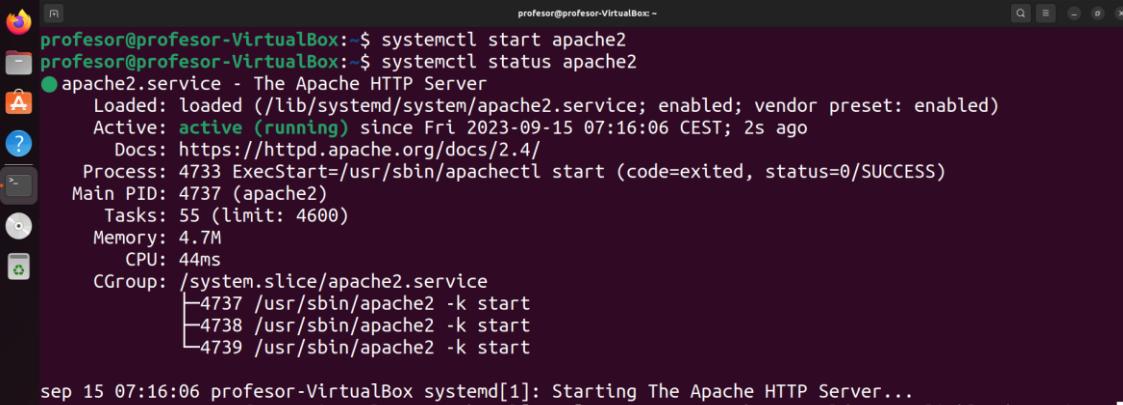


```
profesor@profesor-VirtualBox:~$ systemctl status apache2
● apache2.service - The Apache HTTP Server
   Loaded: loaded (/lib/systemd/system/apache2.service; enabled; vendor preset: enabled)
   Active: active (running) since Fri 2023-09-15 07:11:12 CEST; 1min 43s ago
     Docs: https://httpd.apache.org/docs/2.4/
         Main PID: 4368 (apache2)
            Tasks: 55 (limit: 4600)
           Memory: 4.9M
              CPU: 71ms
             CGroup: /system.slice/apache2.service
                     ├─4368 /usr/sbin/apache2 -k start
                     ├─4370 /usr/sbin/apache2 -k start
                     └─4371 /usr/sbin/apache2 -k start

sep 15 07:11:12 profesor-VirtualBox systemd[1]: Starting The Apache HTTP Server...
sep 15 07:11:12 profesor-VirtualBox apachectl[4367]: AH00558: apache2: Could not reliably determine the parent PID of the current process, ignoring error.
sep 15 07:11:12 profesor-VirtualBox systemd[1]: Started The Apache HTTP Server.
1lines 1-16/16 (END)
```

Si queremos iniciar el servidor web:

\$ systemctl start apache2 ó systemctl start apache2.service

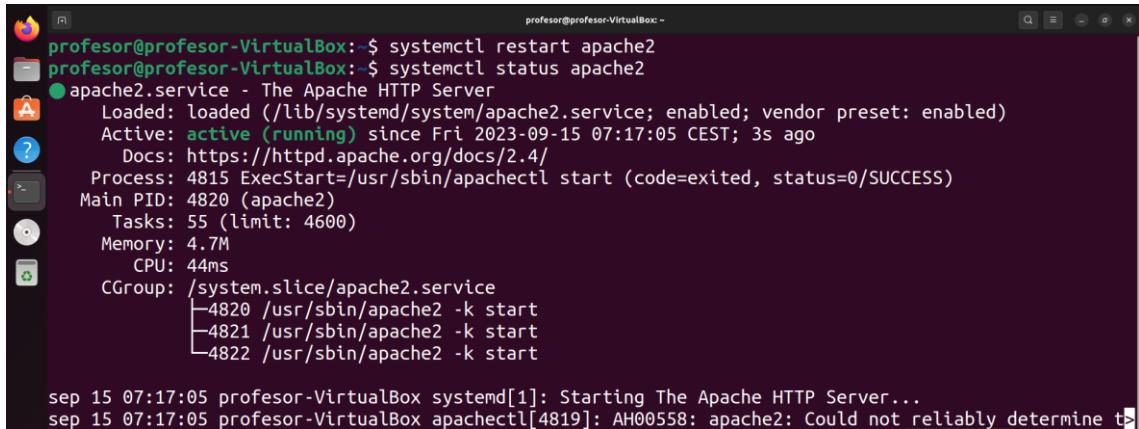


```
profesor@profesor-VirtualBox:~$ systemctl start apache2
profesor@profesor-VirtualBox:~$ systemctl status apache2
● apache2.service - The Apache HTTP Server
   Loaded: loaded (/lib/systemd/system/apache2.service; enabled; vendor preset: enabled)
   Active: active (running) since Fri 2023-09-15 07:16:06 CEST; 2s ago
     Docs: https://httpd.apache.org/docs/2.4/
        Process: 4733 ExecStart=/usr/sbin/apachectl start (code=exited, status=0/SUCCESS)
      Main PID: 4737 (apache2)
         Tasks: 55 (limit: 4600)
        Memory: 4.7M
           CPU: 44ms
          CGroup: /system.slice/apache2.service
                  ├─4737 /usr/sbin/apache2 -k start
                  ├─4738 /usr/sbin/apache2 -k start
                  └─4739 /usr/sbin/apache2 -k start

sep 15 07:16:06 profesor-VirtualBox systemd[1]: Starting The Apache HTTP Server...
sep 15 07:16:06 profesor-VirtualBox apachectl[4736]: AH00558: apache2: Could not reliably determine t
```

Si queremos reiniciar el servidor web:

\$ systemctl restart apache2 ó systemctl restart apache2.service

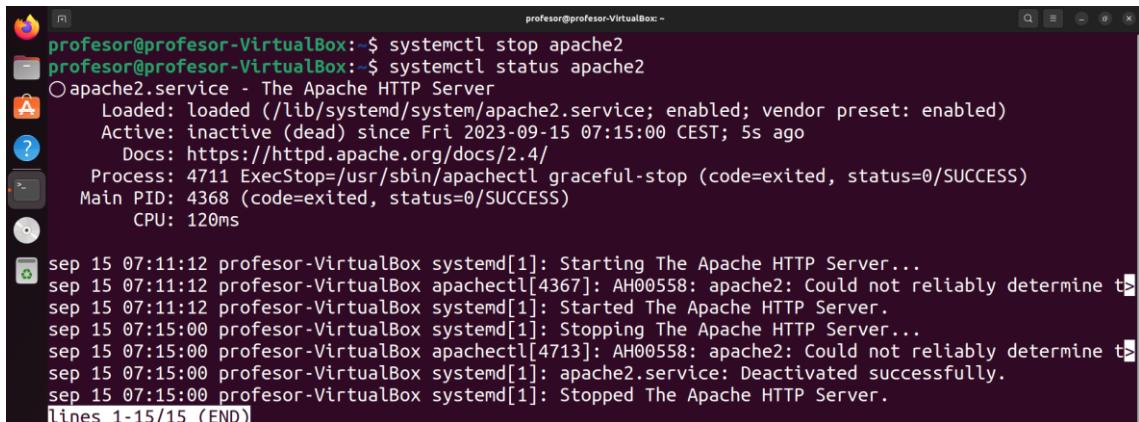


```
profesor@profesor-VirtualBox:~$ systemctl restart apache2
profesor@profesor-VirtualBox:~$ systemctl status apache2
● apache2.service - The Apache HTTP Server
   Loaded: loaded (/lib/systemd/system/apache2.service; enabled; vendor preset: enabled)
   Active: active (running) since Fri 2023-09-15 07:17:05 CEST; 3s ago
     Docs: https://httpd.apache.org/docs/2.4/
   Process: 4815 ExecStart=/usr/sbin/apachectl start (code=exited, status=0/SUCCESS)
 Main PID: 4820 (apache2)
    Tasks: 55 (limit: 4600)
   Memory: 4.7M
      CPU: 44ms
     CGroup: /system.slice/apache2.service
             └─4820 /usr/sbin/apache2 -k start
                 ├─4821 /usr/sbin/apache2 -k start
                 ├─4822 /usr/sbin/apache2 -k start

sep 15 07:17:05 profesor-VirtualBox systemd[1]: Starting The Apache HTTP Server...
sep 15 07:17:05 profesor-VirtualBox apachectl[4819]: AH00558: apache2: Could not reliably determine the server's fully qualified domain name, using localhost. in /etc/apache2/envvars
```

Y para detener el servidor web utilizamos el comando:

\$ systemctl stop apache2 ó systemctl stop apache2.service



```
profesor@profesor-VirtualBox:~$ systemctl stop apache2
profesor@profesor-VirtualBox:~$ systemctl status apache2
● apache2.service - The Apache HTTP Server
   Loaded: loaded (/lib/systemd/system/apache2.service; enabled; vendor preset: enabled)
   Active: inactive (dead) since Fri 2023-09-15 07:15:00 CEST; 5s ago
     Docs: https://httpd.apache.org/docs/2.4/
   Process: 4711 ExecStop=/usr/sbin/apachectl graceful-stop (code=exited, status=0/SUCCESS)
 Main PID: 4368 (code=exited, status=0/SUCCESS)
      CPU: 120ms

sep 15 07:11:12 profesor-VirtualBox systemd[1]: Starting The Apache HTTP Server...
sep 15 07:11:12 profesor-VirtualBox apachectl[4367]: AH00558: apache2: Could not reliably determine the server's fully qualified domain name, using localhost. in /etc/apache2/envvars
sep 15 07:11:12 profesor-VirtualBox systemd[1]: Started The Apache HTTP Server.
sep 15 07:15:00 profesor-VirtualBox systemd[1]: Stopping The Apache HTTP Server...
sep 15 07:15:00 profesor-VirtualBox apachectl[4713]: AH00558: apache2: Could not reliably determine the server's fully qualified domain name, using localhost. in /etc/apache2/envvars
sep 15 07:15:00 profesor-VirtualBox systemd[1]: apache2.service: Deactivated successfully.
sep 15 07:15:00 profesor-VirtualBox systemd[1]: Stopped The Apache HTTP Server.
lines 1-15/15 (END)
```

Si necesitas saber qué versión exacta del servidor web Apache está corriendo en tu máquina Ubuntu 22.04 LTS puedes saberlo con el comando apachectl:

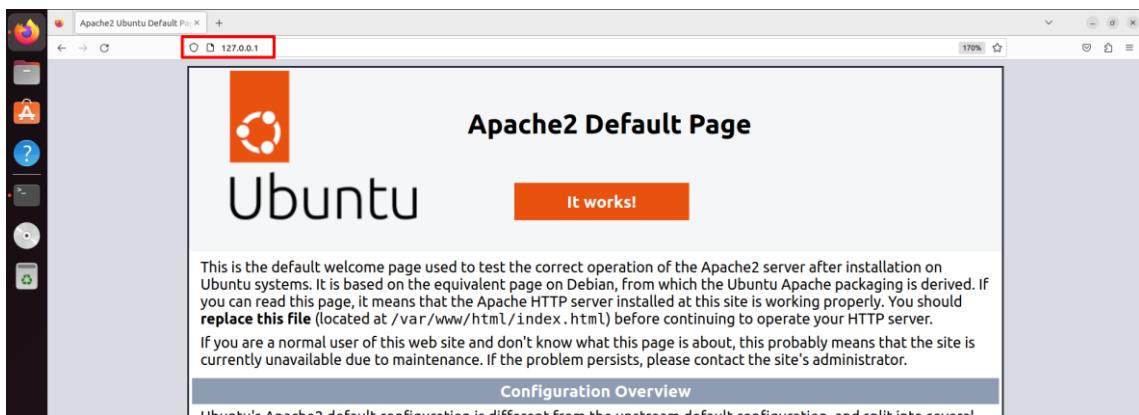
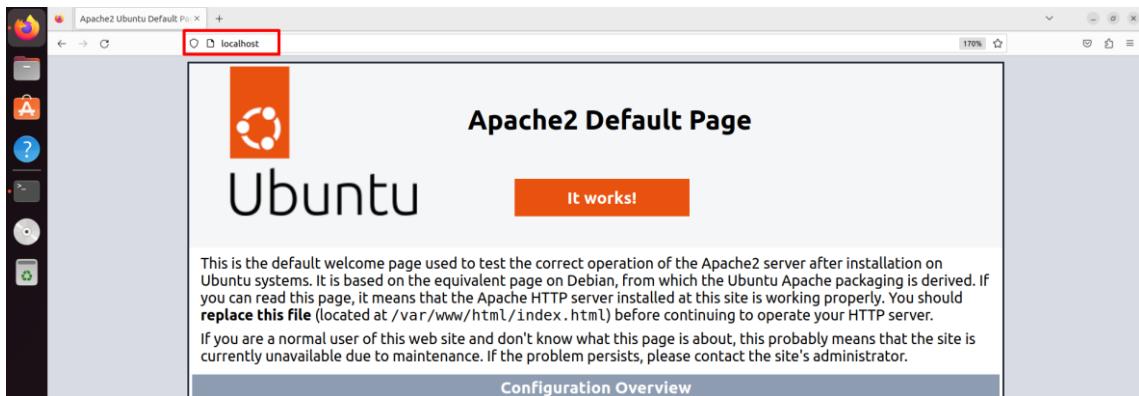


```
profesor@profesor-VirtualBox:~$ apachectl -v
Server version: Apache/2.4.52 (Ubuntu)
Server built: 2023-05-03T20:02:51
```

3.3. Comprobar que está funcionando el servidor Apache desde un navegador web.

Para esta comprobación, abrimos nuestro navegador en la máquina Ubuntu, por ejemplo Mozilla Firefox y en la barra de navegación escribimos la url:

- <http://localhost>
- <http://127.0.0.1>

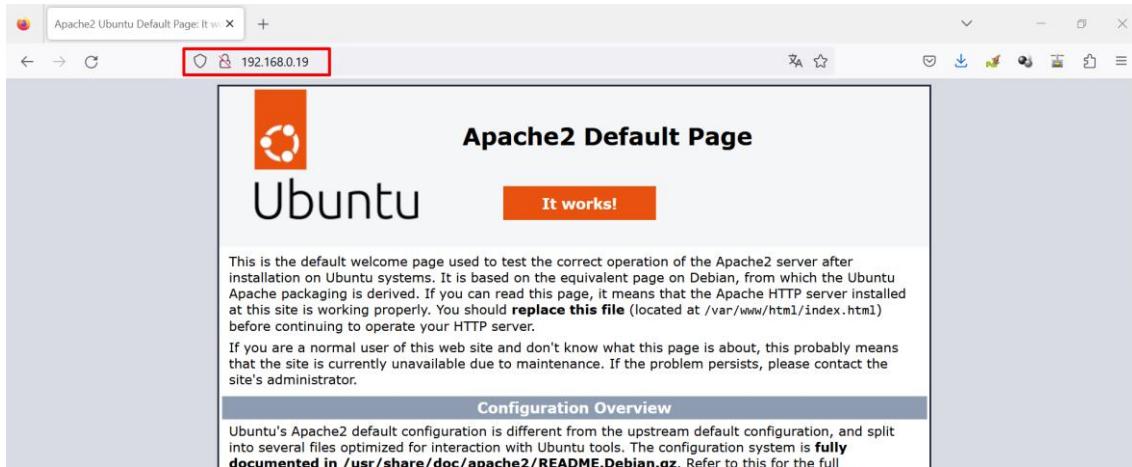


Si en lugar de hacerlo desde el navegador de la máquina local, lo queremos hacer desde un navegador externo a Ubuntu (desde el navegador de nuestra máquina anfitriona) debemos sustituir la ip 127.0.0.1 por la ip de nuestra máquina Ubuntu. Para obtener esta ip usaremos el comando:

```
$ ip addr
```

```
profesor@profesor-VirtualBox:~$ ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
        inet 127.0.0.1/8 scope host lo
            valid_lft forever preferred_lft forever
        inet6 ::1/128 scope host
            valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:8f:a2:0d brd ff:ff:ff:ff:ff:ff
        inet 192.168.0.19/24 brd 192.168.0.255 scope global dynamic noprefixroute enp0s3
            valid_lft 86393sec preferred_lft 86393sec
        inet6 fe80::375:4956:6d94:5b2d/64 scope link noprefixroute
            valid_lft forever preferred_lft forever
```

En este caso vemos que la ip asignada a la interfaz de red enp0s3 es la 192.168.0.19. Si escribimos esta ip en el navegador de nuestra máquina anfitriona, veremos que también carga la página por defecto del servidor web:



3.4. Cambiar el puerto por el cual está escuchando Apache pasándolo al puerto 8088 y comprueba de nuevo desde tu navegador que está funcionando.

El comando `ss` se usa para mostrar información de red en sistemas Gnu/Linux. Para poder comprobar los puertos a la escucha, no habrá más que abrir una terminal y escribir en ella:

`$ ss -ntl`

```
profesor@profesor-VirtualBox:~$ ss -ntl
State      Recv-Q    Send-Q      Local Address:Port          Peer Address:Port      Process
LISTEN     0          4096        127.0.0.53%lo:53        0.0.0.0:*
LISTEN     0          128         127.0.0.1:631           0.0.0.0:*
LISTEN     0          128         [::]:631              [::]:*
LISTEN     0          511         *:80                  *:*
```

Para modificar el puerto de escucha de Apache debemos editar el fichero de configuración `ports.conf` que se encuentra en la carpeta `/etc/apache2`. Para ello podemos usar cualquier editor de texto, como por ejemplo nano:

`$ nano /etc/apache2/ports.conf`

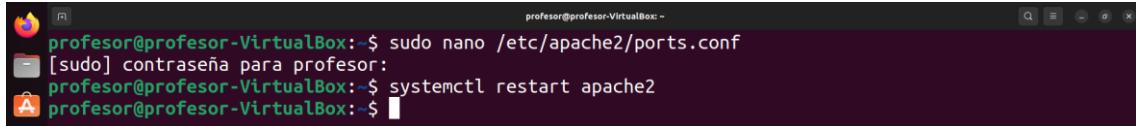
```
profesor@profesor-VirtualBox:~$ sudo nano /etc/apache2/ports.conf
```

Buscamos la línea en la que indica el puerto de escucha con la directiva `Listen` y cambiamos el valor actual (80) por el nuevo valor (8088):

```
GNU nano 6.2                               /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 8088
<IfModule ssl_module>
    Listen 443
</IfModule>
<IfModule mod_gnutls.c>
    Listen 443
</IfModule>
# vim: syntax=apache ts=4 sw=4 sts=4 sr noet
```

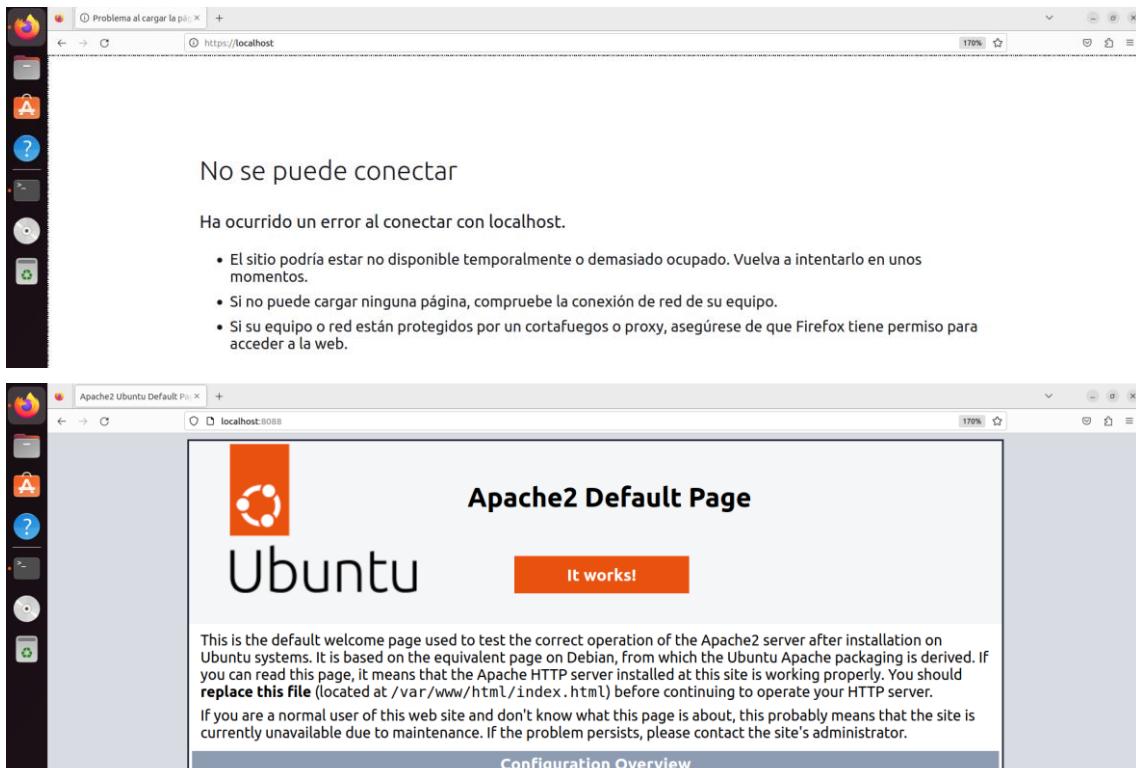
Para que los cambios tengan efecto, debemos reiniciar el servidor web:

```
$ systemctl restart apache2
```



```
profesor@profesor-VirtualBox:~$ sudo nano /etc/apache2/ports.conf
[sudo] contraseña para profesor:
profesor@profesor-VirtualBox:~$ systemctl restart apache2
profesor@profesor-VirtualBox:~$
```

Ahora abrimos nuestro navegador y comprobamos que, para acceder a la página por defecto de Apache, es necesario especificar el nuevo puerto, escribiendo `http://localhost:8088`, ya que solo con `http://localhost` vemos que no funciona.



Especificando el nuevo puerto, sí que funciona

3.5. Cambiar la página web por defecto para que aparezcan tus apellidos, más un pantallazo de tu inicio del curso.

Para esta actividad debemos hacer una captura de la plataforma y guardarla en formato jpg/gif/png en nuestro servidor web.

La forma más sencilla de cambiar la página principal es sustituir la página `index.html` que viene por defecto por una que hagamos nosotros. En mi caso voy a renombrar la actual y luego crear una nueva:

```
$ mv /var/www/html/index.html /var/www/html/index.html.old
```

```
$ nano /var/www/html/index.html
```

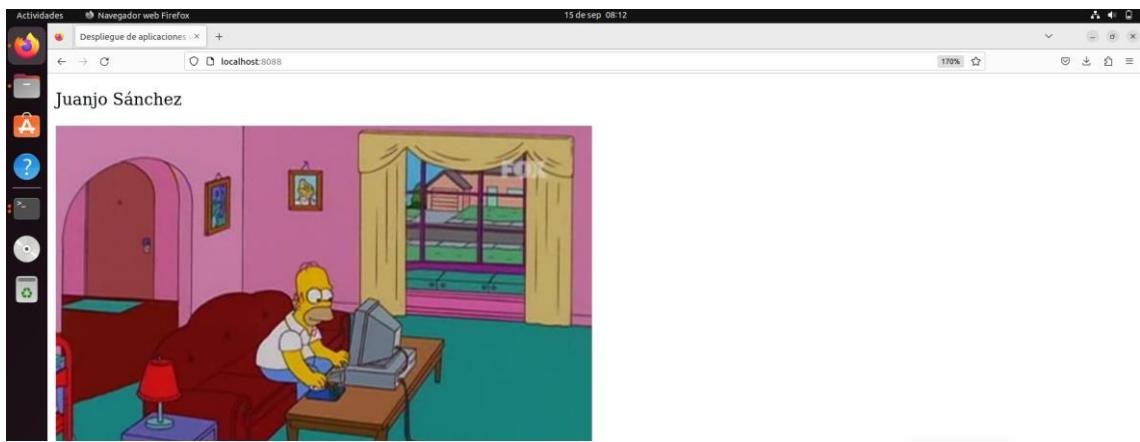


```
profesor@profesor-VirtualBox:~$ sudo mv /var/www/html/index.html /var/www/html/index.html.old
profesor@profesor-VirtualBox:~$ sudo nano /var/www/html/index.html
```

```
profesor@profesor-VirtualBox: ~
/var/www/html/index.html *
```

```
GNU nano 6.2
<!DOCTYPE html>
<html>
  <head>
    <title>Despliegue de aplicaciones web</title>
  </head>
  <body>
    <p>Juanjo Sánchez</p>
    
  </body>
</html>
```

Ahora abrimos el navegador y comprobamos que se ha cambiado la web por defecto:

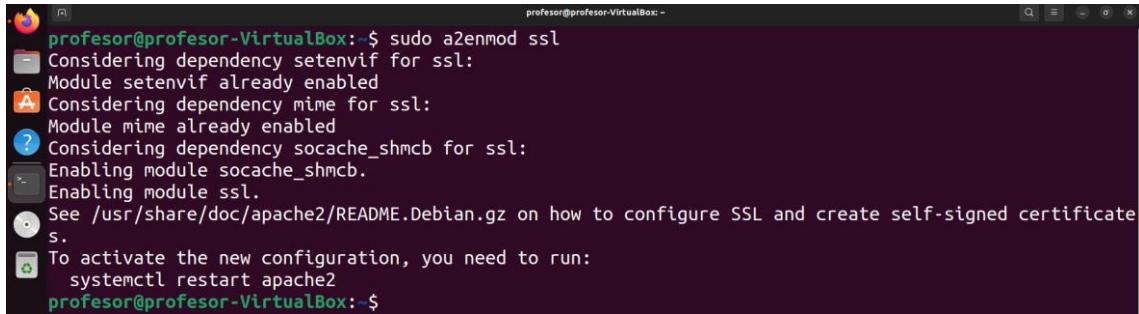


Ejercicio 4. Haciendo las comunicaciones más seguras.

Partiendo de la instalación de Apache del ejercicio anterior, instala un certificado de seguridad (SSL) en tu servidor y habilita el módulo de Apache correspondiente para que funcione correctamente. Comprueba desde tu navegador que ahora puedes acceder al servidor usando el protocolo seguro (https).

Lo primero que vamos a hacer es habilitar el módulo de Apache2 que se encarga de la gestión de las gestiones de ssl:

```
$ sudo a2enmod ssl
```



```
profesor@profesor-VirtualBox:~$ sudo a2enmod ssl
[sudo] contraseña para profesor:
Considering dependency setenvif for ssl:
Module setenvif already enabled
Considering dependency mime for ssl:
Module mime already enabled
Considering dependency socache_shmcb for ssl:
Enabling module socache_shmcb.
Enabling module ssl.
See /usr/share/doc/apache2/README.Debian.gz on how to configure SSL and create self-signed certificates.
To activate the new configuration, you need to run:
    systemctl restart apache2
profesor@profesor-VirtualBox:~$
```

Tal y como nos indica, debemos de

```
$ systemctl restart apache2
```



```
profesor@profesor-VirtualBox:~$ systemctl restart apache2
profesor@profesor-VirtualBox:~$
```

Con el módulo de cifrado SSL ya activo necesitamos archivos de claves de cifrado y certificados públicos para cada uno de los dominios que queramos configurar. Para un servidor local o en pruebas podemos crear claves y certificados autofirmados con la herramienta openssl. Procedemos a instalar openSSL para poder generar certificados:

```
$ apt install openssl
```



```
profesor@profesor-VirtualBox:~$ sudo apt install openssl
[sudo] contraseña para profesor:
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias... Hecho
Leyendo la información de estado... Hecho
openssl ya está en su versión más reciente (3.0.2-0ubuntu1.10).
fijado openssl como instalado manualmente.
0 actualizados, 0 nuevos se instalarán, 0 para eliminar y 61 no actualizados.
profesor@profesor-VirtualBox:~$ openssl version
OpenSSL 3.0.2 15 Mar 2022 (Library: OpenSSL 3.0.2 15 Mar 2022)
profesor@profesor-VirtualBox:~$
```

Ahora ejecutamos openssl con los parámetros necesarios para que nos genere un certificado en ese directorio. En los parámetros de este comando especificamos la ruta y nombre del certificado y clave (es buena idea usar el dominio con el que estarán relacionados):

```
$ sudo openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout
/etc/ssl/private/apache-clave.key -out /etc/ssl/certs/apache-certificado.crt
```

```
profesor@profesor-VirtualBox:~$ sudo openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout /etc/ssl/private/apache-clave.key -out /etc/ssl/certs/apache-certificado.crt
[sudo] contraseña para profesor:
```

El siguiente paso es modificar el fichero por defecto de Apache, en este caso el default-ssl.conf, que se encuentra en la carpeta /etc/apache2/sites-available

\$ nano /etc/apache2/sites-available/default-ssl.conf



Buscamos estas líneas y cambiamos los valores por defecto por las rutas de certificado y clave que acabamos de crear:

```
GNU nano 6.2 /etc/apache2/sites-available/default-ssl.conf *
# following line enables the CGI configuration for this host only
# after it has been globally disabled with "a2disconf".
#Include conf-available/serve-cgi-bin.conf

#   SSL Engine Switch:
#   # Enable/Disable SSL for this virtual host.
SSLEngine on

#   A self-signed (snakeoil) certificate can be created by installing
#   the ssl-cert package. See
#   /usr/share/doc/apache2/README.Debian.gz for more info.
#   If both key and certificate are stored in the same file, only the
#   SSLCertificateFile directive is needed.
SSLCertificateFile /etc/ssl/certs/apache-certificado.crt
SSLCertificateKeyFile /etc/ssl/private/apache-clave.key

#   Server Certificate Chain:
```

Guardamos el fichero, habilitamos el sitio web seguro por defecto:

\$ a2ensite default-ssl.conf

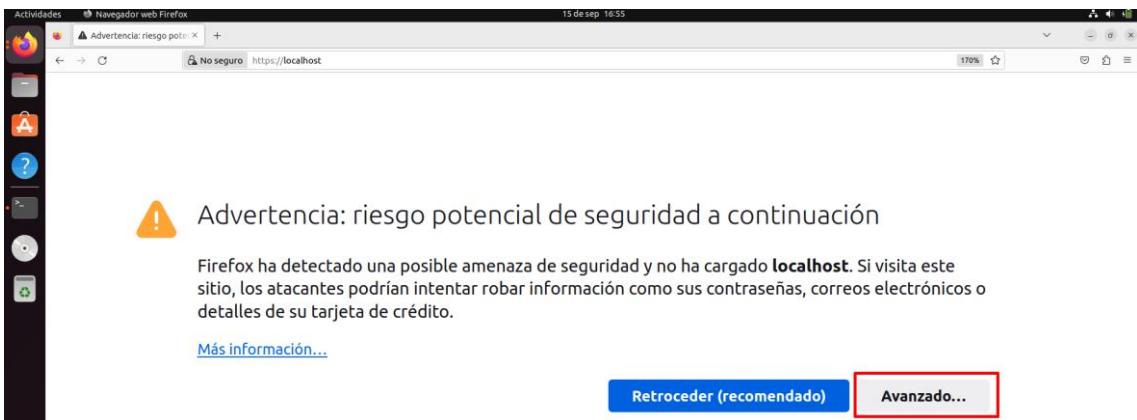
```
profesor@profesor-VirtualBox:~$ sudo a2ensite default-ssl.conf
Enabling site default-ssl.
To activate the new configuration, you need to run:
  systemctl reload apache2
profesor@profesor-VirtualBox:~$
```

Y recargamos la configuración del servidor web:

\$ systemctl restart apache2

```
profesor@profesor-VirtualBox:~$ systemctl restart apache2
```

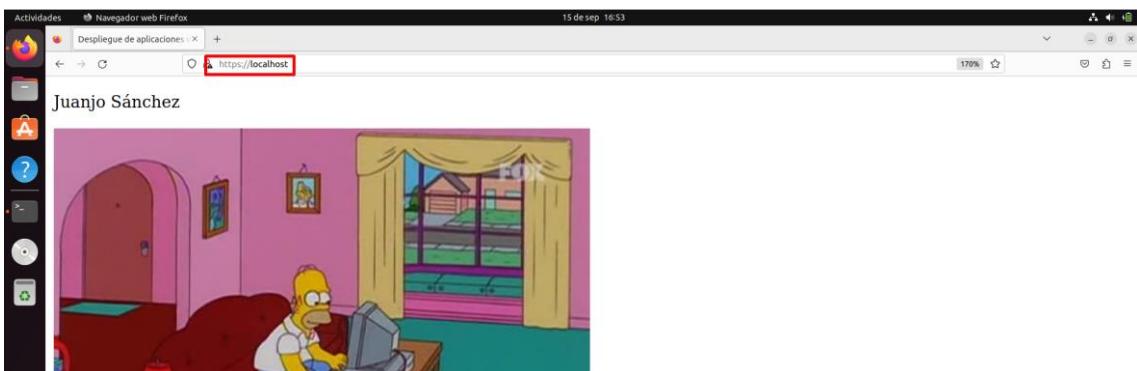
Ahora vamos a nuestro navegador e intentamos cargar la url https://localhost



Nos muestra un aviso, ya que el certificado es auto-generado y no está emitido por una entidad de confianza. Si pinchamos en el botón “Avanzado...” nos mostrará esta información y nos permitirá continuar con la navegación aceptando los riesgos.



Si clicamos en “Aceptar el riesgo y continuar” nos mostrará nuestra página:



Si pinchamos en el candado podemos ver la información del certificado que está usando el servidor web:

Firefox 15 de sep 16:54

Certificado para DAW

Firefox about:certificate?cert=MIIDkzCCAnugAwIBAgIUBfWQk2tfqD0hFLCDG6FRE2LhvxDQYJKoZIhvcNAQELBQAwWTELMAkGA1UEBhMCU1AxEDAOE

Certificado

DAW

Nombre del asunto

País	SP
Estado/Provincia	Almeria
Localidad	Aguadulce
Organización	IES Aguadulce
Nombre común	DAW

Nombre del emisor

País	SP
Estado/Provincia	Almeria
Localidad	Aguadulce
Organización	IES Aguadulce
Nombre común	DAW

Validez

No antes	Fri, 15 Sep 2023 14:42:43 GMT
No después	Sat, 14 Sep 2024 14:42:43 GMT

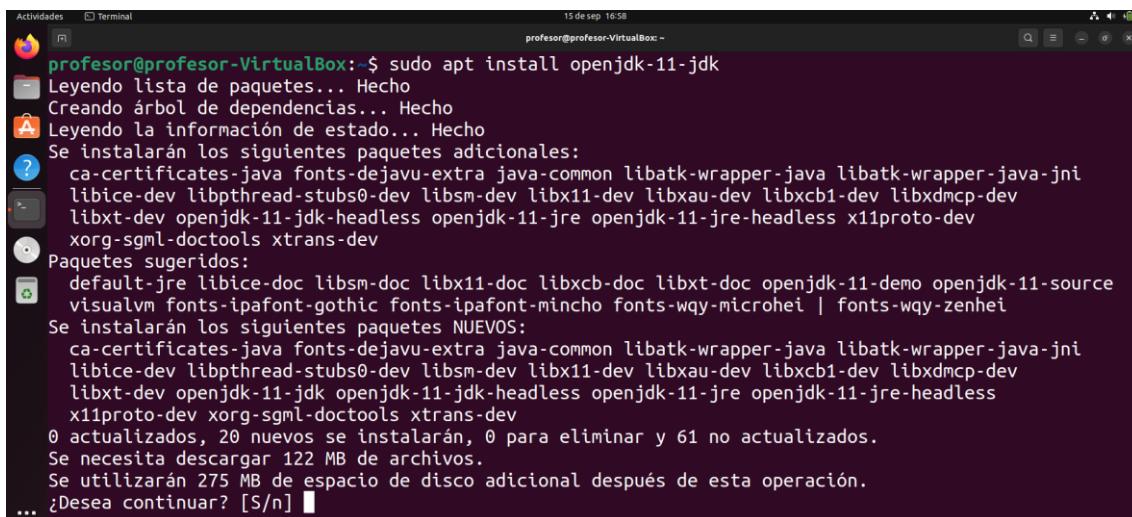
Información de clave pública

Ejercicio 5. Instalando un servidor de aplicaciones.

Partiendo de la instalación de Apache del ejercicio 3, realiza la instalación del servidor de aplicaciones **Apache Tomcat** (es recomendable instalar una versión lo más reciente posible, tomcat10). Para ayudarte en el proceso de instalación encontrarás un foro dedicado a esta tarea, con guías y vídeos explicativos. **Importante:** para esta actividad solo hay que hacer la instalación y comprobar desde nuestro navegador que tenemos acceso a la página principal del servidor de aplicaciones. No hay que configurar el acceso al panel de administración ni crear usuarios.

Lo primero que tenemos que hacer es instalarnos las librerías de Java si no las tenemos: \$

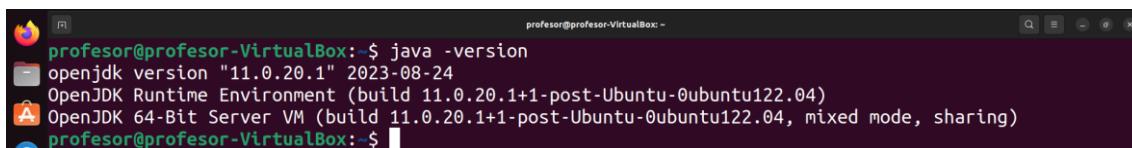
```
$ sudo apt install openjdk-11-jdk
```



```
profesor@profesor-VirtualBox:~$ sudo apt install openjdk-11-jdk
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias... Hecho
Leyendo la información de estado... Hecho
Se instalarán los siguientes paquetes adicionales:
  ca-certificates-java fonts-dejavu-extra java-common libatk-wrapper-java libatk-wrapper-java-jni
  libice-dev libpthread-stubs0-dev libsm-dev libx11-dev libxau-dev libxcb1-dev libxdmcp-dev
  libxt-dev openjdk-11-jdk-headless openjdk-11-jre openjdk-11-jre-headless x11proto-dev
  xorg-sgml-doctools xtrans-dev
Paquetes sugeridos:
  default-jre libice-doc libsm-doc libx11-doc libxcb-doc libxt-doc openjdk-11-demo openjdk-11-source
  visualvm fonts-ipafont-gothic fonts-ipafont-mincho fonts-wqy-microhei | fonts-wqy-zenhei
Se instalarán los siguientes paquetes NUEVOS:
  ca-certificates-java fonts-dejavu-extra java-common libatk-wrapper-java libatk-wrapper-java-jni
  libice-dev libpthread-stubs0-dev libsm-dev libx11-dev libxau-dev libxcb1-dev libxdmcp-dev
  libxt-dev openjdk-11-jdk openjdk-11-jdk-headless openjdk-11-jre openjdk-11-jre-headless
  x11proto-dev xorg-sgml-doctools xtrans-dev
0 actualizados, 20 nuevos se instalarán, 0 para eliminar y 61 no actualizados.
Se necesita descargar 122 MB de archivos.
Se utilizarán 275 MB de espacio de disco adicional después de esta operación.
¿Desea continuar? [S/n] ■
```

Una vez instalado, podemos comprobarla con:

```
$ java -version
```



```
profesor@profesor-VirtualBox:~$ java -version
openjdk version "11.0.20.1" 2023-08-24
OpenJDK Runtime Environment (build 11.0.20.1+1-post-Ubuntu-0ubuntu122.04)
OpenJDK 64-Bit Server VM (build 11.0.20.1+1-post-Ubuntu-0ubuntu122.04, mixed mode, sharing)
profesor@profesor-VirtualBox:~$
```

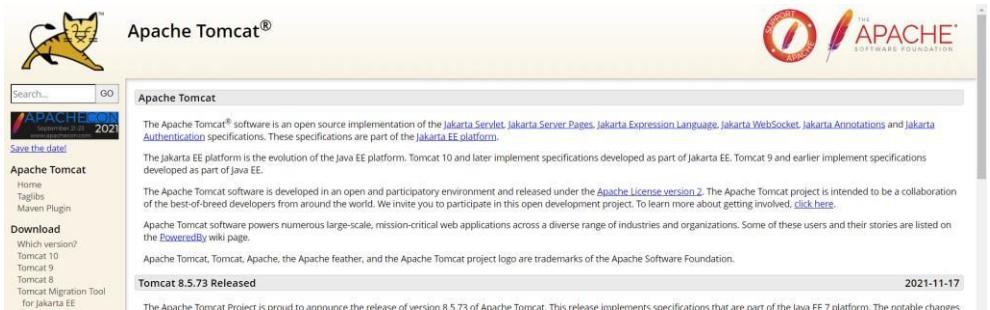
En segundo lugar vamos a crear un usuario que será el que se encargue de ejecutar Tomcat como un servicio, ya que ejecutar Tomcat bajo el usuario root podría ser peligroso. El siguiente comando crea un nuevo usuario y grupo cuyo directorio de trabajo es /opt/tomcat y será el que ejecutará el servicio Tomcat:

```
$ sudo useradd -m -U -d /opt/tomcat -s /bin/false tomcat
```



```
profesor@profesor-VirtualBox:~$ sudo useradd -r -m -U -d /opt/tomcat -s /bin/false tomcat
```

Ahora descargamos los ficheros de Tomcat desde la página oficial: <https://tomcat.apache.org/>



En el menú lateral seleccionamos la versión que necesitamos. En nuestro caso la 10. Ahora nos desplazamos hasta la sección donde aparezca la versión y revisión concreta y en el enlace de descarga (el tarball comprimido), lo pinchamos con el botón derecho y seleccionamos la opción ‘copiar dirección de enlace’.

Si pegamos el enlace en un bloc de notas o en nuestro terminal, tendrá un aspecto similar a esto:

`https://dlcdn.apache.org/tomcat/tomcat-10/v10.0.14/bin/apache-tomcat-10.0.14.tar.gz`

Ahora realizamos la descarga en nuestro servidor usando el comando wget.

`$ wget https://dlcdn.apache.org/tomcat/tomcat-10/v10.1.13/bin/apache-tomcat-10.1.13.tar.gz`

```
profesor@profesor-VirtualBox:~$ wget https://dlcdn.apache.org/tomcat/tomcat-10/v10.1.13/bin/apache-tomcat-10.1.13.tar.gz
--2023-09-15 17:06:12-- https://dlcdn.apache.org/tomcat/tomcat-10/v10.1.13/bin/apache-tomcat-10.1.13.tar.gz
Resolviendo dlcdn.apache.org (dlcdn.apache.org)... 151.101.2.132
Conectando con dlcdn.apache.org (dlcdn.apache.org)[151.101.2.132]:443... conectado.
Petición HTTP enviada, esperando respuesta... 200 OK
Longitud: 12377776 (12M) [application/x-gzip]
Guardando como: 'apache-tomcat-10.1.13.tar.gz'

apache-tomcat-10.1.13.tar 100%[=====] 11,80M 27,9MB/s en 0,4s
2023-09-15 17:06:13 (27,9 MB/s) - 'apache-tomcat-10.1.13.tar.gz' guardado [12377776/12377776]
```

Descomprimimos el tarball, usando como carpeta de destino el directorio de tomcat:

`$ sudo tar -xf /tmp/apache-tomcat-10.0.14.tar.gz -C /opt/tomcat/`

```
profesor@profesor-VirtualBox:~$ sudo tar xzf apache-tomcat-*.tar.gz -C /opt/tomcat
```

Tomcat se actualiza periódicamente. Las actualizaciones incluyen correcciones de errores, parches de seguridad y nuevas funciones. Para tener más control sobre las versiones y actualizaciones, crearemos un enlace simbólico llamado latest, que apuntará al directorio de instalación de Tomcat, y así no tener que modificar los ficheros de configuración de Tomcat cada vez que se actualice. En caso de actualizar, lo único que tendríamos que hacer sería crear nuevos enlaces simbólicos a la nueva versión, sin tener que modificar los ficheros de configuración de Tomcat.

`$ sudo ln -s /opt/tomcat/apache-tomcat-10.1.13 /opt/tomcat/latest`

```
profesor@profesor-VirtualBox:~$ sudo ln -s /opt/tomcat/apache-tomcat-10.1.13 /opt/tomcat/latest
```

También vamos a cambiar el propietario del directorio /opt/tomcat y además le vamos a otorgar permisos de ejecución en dicho directorio para los ficheros de tipo script:

`$ sudo chown -R tomcat: /opt/tomcat`



```
profesor@profesor-VirtualBox:~$ sudo chown -R tomcat: /opt/tomcat
$ sudo sh -c 'chmod +x /opt/tomcat/latest/bin/*.sh'
profesor@profesor-VirtualBox:~$ sudo sh -c 'chmod +x /opt/tomcat/latest/bin/*.sh'
```

Para no tener que ejecutar manualmente los scripts de arranque y parada de Tomcat, vamos a crear un servicio para que el manejo sea similar al de Apache Web Server.

Tenemos que crear el siguiente fichero *tomcat.service* en la carpeta */etc/systemd/system/* e incluir el contenido que hay un poco más abajo:

```
$ sudo nano /etc/systemd/system/tomcat.service
```



El código que tenemos que incluir es:

```
[Unit]
Description=Apache Tomcat 10 Web Application Server
After=network.target

[Service]
Type=forking

User=tomcat
Group=tomcat

Environment="JAVA_HOME=/usr/lib/jvm/java-11-openjdk-amd64"
Environment="CATALINA_HOME=/opt/tomcat"
Environment="CATALINA_BASE=/opt/tomcat"
Environment="CATALINA_PID=/opt/tomcat/temp/tomcat.pid"
Environment="CATALINA_OPTS=-Xms512M -Xmx1024M -server -XX:+UseParallelGC"

ExecStart=/opt/tomcat/bin/startup.sh
ExecStop=/opt/tomcat/bin/shutdown.sh

[Install]
WantedBy=multi-user.target
```

Ojo! La variable de entorno Environment="JAVA_HOME=/usr/lib/jvm/openjdk-11" dependerá de qué versión tengamos de JAVA. Revisa el directorio /usr/lib/jvm para saber cómo se ha instalado.

Guardamos el fichero y ejecutamos el siguiente comando para informar al sistema operativo que hemos creado un nuevo fichero de sistema:

```
$ sudo systemctl daemon-reload
```

Habilitamos y arrancamos el servicio de Tomcat que acabamos de crear:

```
$ sudo systemctl enable --now tomcat
```

Verificamos el estado del servicio Tomcat:

```
$ sudo systemctl status tomcat
```

Podemos iniciar, parar y reiniciar Tomcat de la misma forma que cualquier otro servicio:

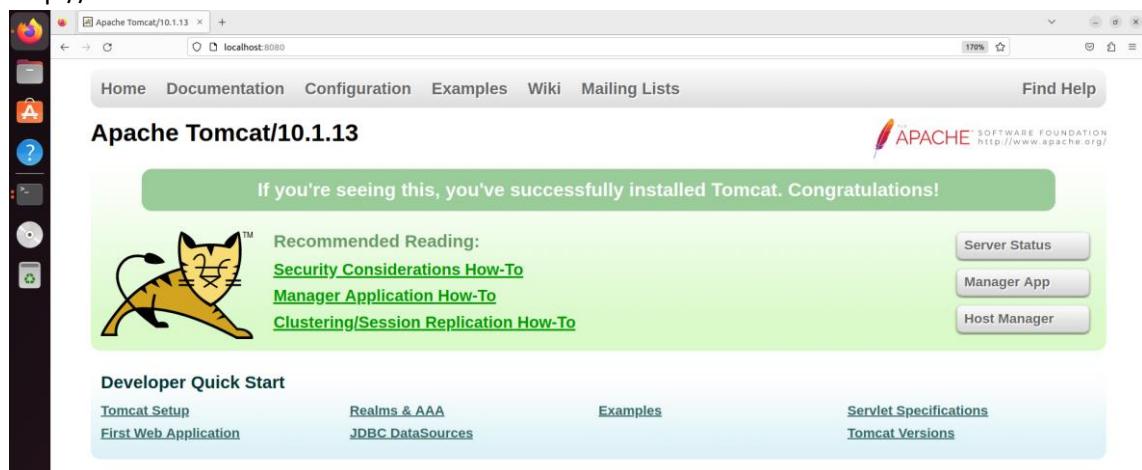
```
$ sudo systemctl start tomcat
```

\$ sudo systemctl stop tomcat

```
$ sudo systemctl restart tomcat
```

Ahora solo queda abrir el navegador y acceder a la página principal de Tomcat: <http://localhost:8080>

<http://127.0.0.1:8080>



Ejercicio 6. Hosts virtuales

Ya sabemos que Apache permite tener más de un sitio web en un servidor, donde cada sitio puede estar asociado a un dominio diferente. Revisa el punto 5 de la documentación y explica qué es, para que se utiliza y cuál es la diferencia entre virtualhost por nombre y por IP. Pon un ejemplo de configuración de cada uno, usando la directiva 'VirtualHost', incluyendo capturas del fichero de configuración.

Esto viene en el punto 5 de la unidad

- Un servidor web con una única dirección IP pública, y que sirve diferentes sitios web (**www.midominio.local**, **www.misitio.local**, **www.miempresa.local**, etc.). El mismo servidor web está configurado para servir diferente contenido dependiendo del dominio que se solicite por el cliente. Esta configuración se la conoce como **hosts** virtuales basados en nombre.
- Un servidor web tiene asignadas varias direcciones IP públicas, y una dirección IP está asociada a un sitio web y otra dirección IP está asociada a otro sitio web. Cada dominio por tanto necesita su propia dirección IP. Esta configuración se la conoce como **hosts** virtuales basados en IP.

Ejemplo de virtualhost basado en nombre:

Creamos la carpeta que alojará el contenido de la nueva web:



```
profesor@profesor-VirtualBox:~$ sudo mkdir /var/www/despliegue.com
```

Creamos el archivo de configuración del host virtual con el siguiente contenido:



```
profesor@profesor-VirtualBox:~$ sudo nano /etc/apache2/sites-available/despliegue.com.conf
```

```
<VirtualHost *:8088>
    ServerName despliegue.com
    DocumentRoot /var/www/despliegue.com
</VirtualHost>
```

Editamos el archivo /etc/hosts para que asociar nuestra dirección ip al dominio del nuevo sitio web:



```
GNU nano 6.2
profesor@profesor-VirtualBox:~$ /etc/hosts
127.0.0.1      localhost
127.0.1.1      profesor-VirtualBox
192.168.0.19   despliegue.com
?
# The following lines are desirable for IPv6 capable hosts
::1      ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
```

Habilitamos el nuevo sitio web:



```
profesor@profesor-VirtualBox:~$ sudo a2ensite despliegue.com
```

Y reiniciamos el servicio apache:



```
profesor@profesor-VirtualBox:~$ sudo systemctl reload apache2
```

Y ya tenemos nuestro host virtual funcionando:

