UNIDAD 1

Implantación, configuración y administración de servidores web.

Ejercicio 1: Clasificación de las Aplicaciones web

Ejercicio 2: Arquitecturas web

Ejercicio 3: Instalando nuestro servidor web

Ejercicio 4: Haciendo las comunicaciones más seguras

Ejercicio 5: Instalando un servidor de aplicaciones

Ejercicio 6: Hosts virtuales

Ejercicios realizados por José Carlos López Henestrosa

Ejercicio 1: 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.

Del lado del cliente, las tecnologías que se usan son el cliente web, el cual suele ser un navegador, como Chrome o Firefox. Por otro lado, el navegador tiene que ejecutar código para permitir al usuario ver e interactuar con página, la cual está programada en HTML, CSS y JavaScript. Lo descrito anteriormente se define con el término "front-end".

Por otra parte, en el lado del servidor, se diseña la lógica y las soluciones para que todas las acciones solicitadas en una página web sean ejecutadas de manera correcta. Esto se realiza a través de lenguajes como PHP (normalmente, se usa un framework como Laravel o Symfony), Java (Spring), Python (Django), Ruby (Ruby on Rails), Javascript (Node. JS) o C# (ASP.NET Core). Esto se considera como el "back-end" de la aplicación web.

Ejercicio 2: 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.

En Linux, la plataforma LAMP incluye componentes de software libre. Su nombre son siglas que denotan las iniciales de los componentes que integra:

- Linux: sistema operativo.
- Apache: servidor web.
- MySQL: gestor de bases de datos.
- PHP/Python/Perl: lenguaje interpretado para generar contenido interactivo y dinámico.

En Windows, la plataforma WISA usa software propietario de Microsoft. Está compuesta de los siguientes componentes:

- 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.

No obstante, hay arquitecturas multiplataforma como XAMPP que incluyen software libre cuyas siglas significan lo siguiente:

- X: Cross-Platform,
- Apache: servidor web.

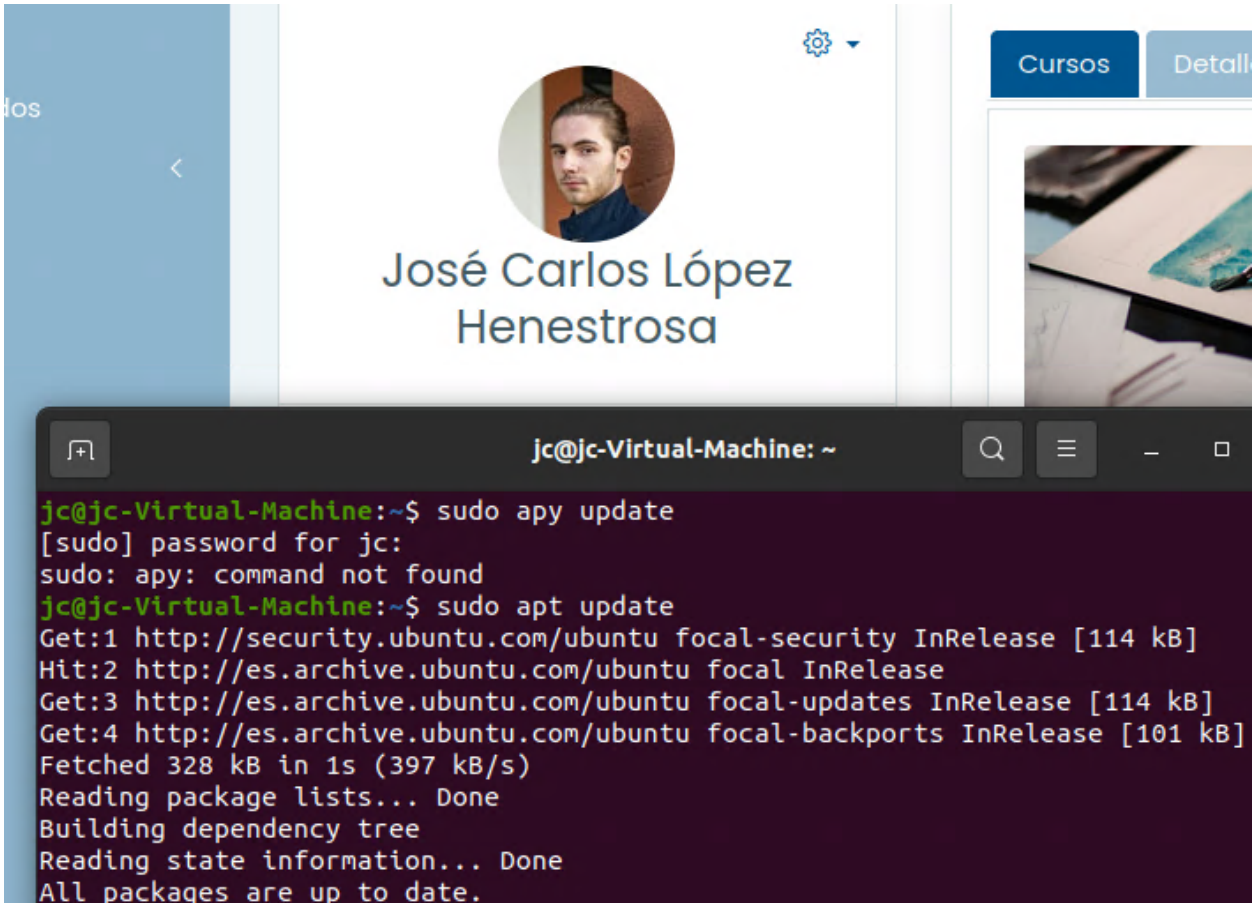
- MySQL: gestor de bases de datos.
- PHP/Python/Perl: lenguaje interpretado para generar contenido interactivo y dinámico.

Ejercicio 3: Instalando nuestro servidor web. Dispones de una máquina (o máquina virtual) que cuenta con el sistema operativo Ubuntu 14/16/18/20 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:

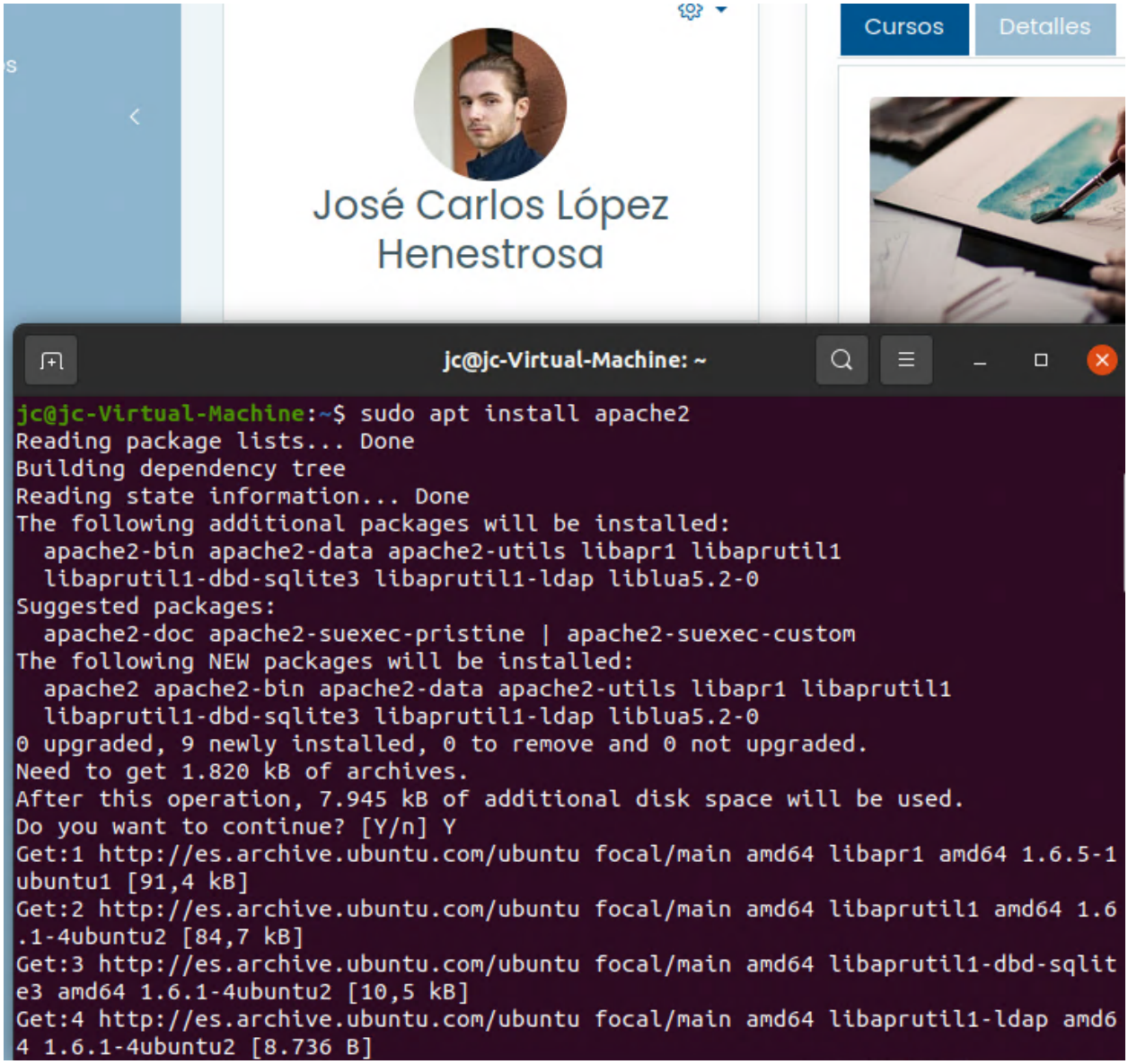
- Instalar el servidor web Apache desde el terminal de comandos.
- Arrancar, reiniciar, comprobar el estado y parar el servidor web Apache.
- Comprobar que está funcionando el servidor Apache desde un navegador web.
- Cambiar el puerto por el cual está escuchando Apache pasándolo al puerto 8088 y comprueba 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.

Para instalar el servidor web Apache, tenemos que seguir los siguientes pasos:

1. Actualizamos el índice de paquetes local para obtener las versiones más recientes de los paquetes.



2. Instalamos el paquete apache2 con el comando sudo apt install apache2

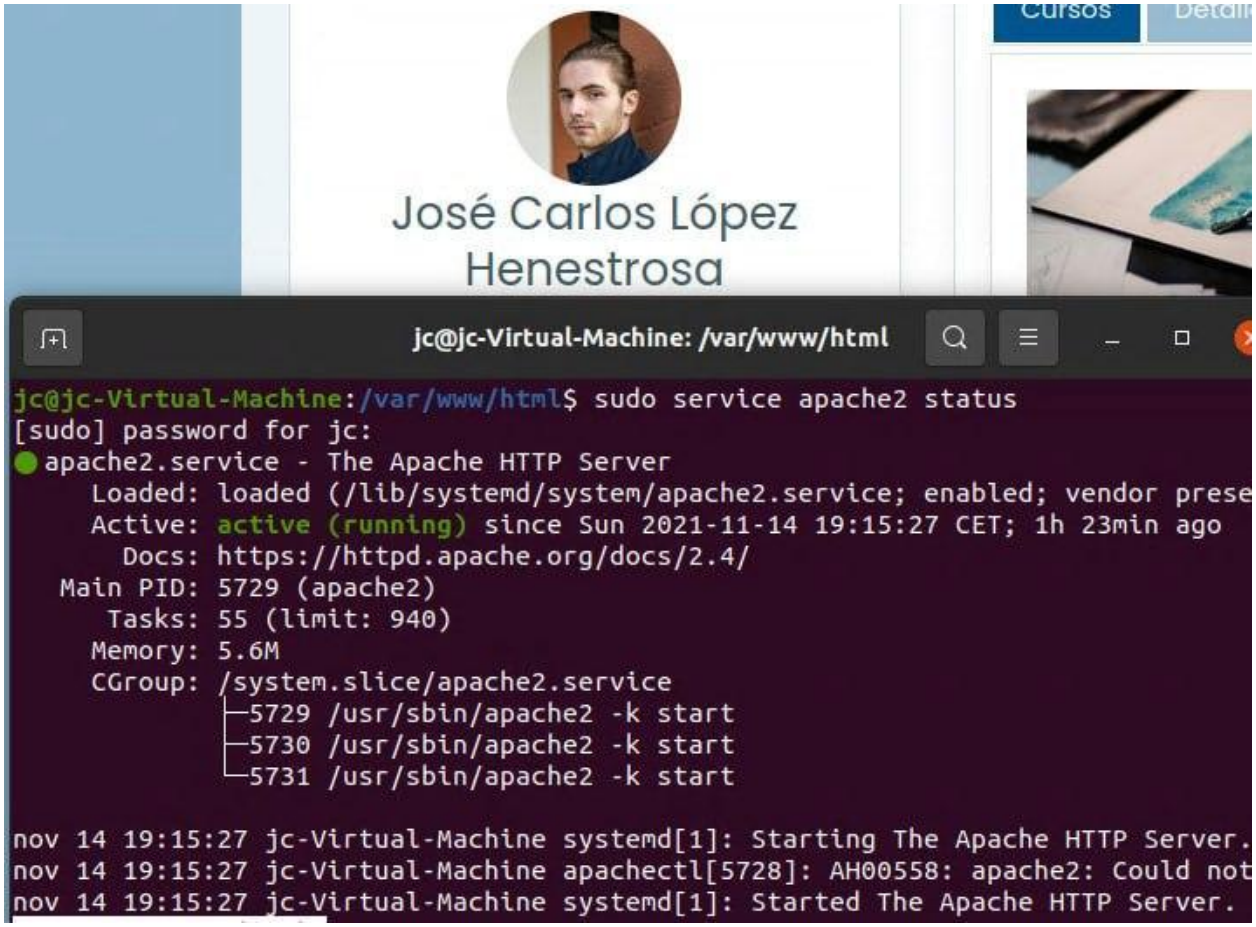


Para arrancar, reiniciar, comprobar el estado y parar el servidor web Apache:

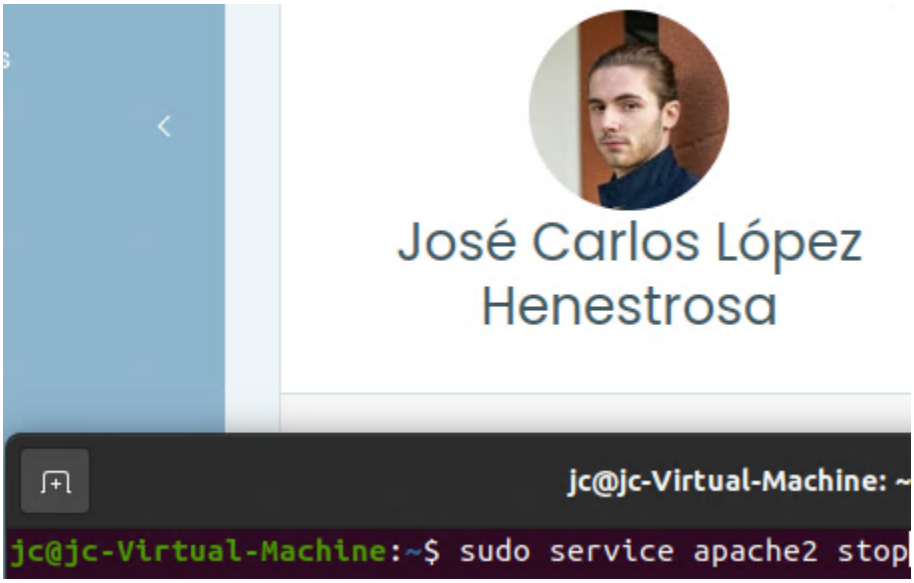
1. Iniciamos el servicio de apache2 con el comando sudo service apache2 start

2. Reiniciamos el servicio con el comando sudo service apache2 restart

3. Comprobamos el estado del servicio con el comando sudo service apache2 status

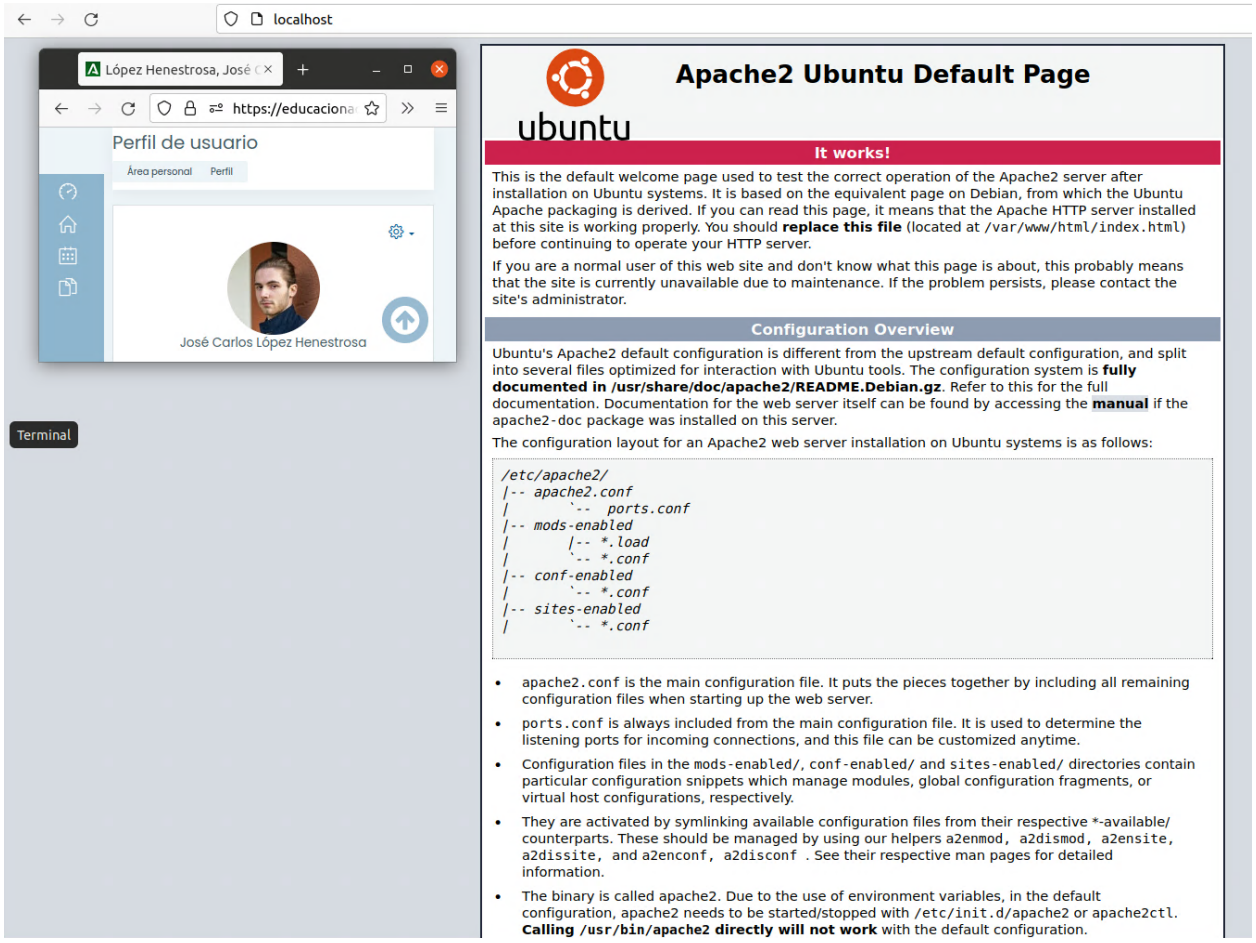


4. Paramos el servicio con el comando sudo service apache2 stop



Para comprobar que está funcionando el servidor Apache desde un navegador web:

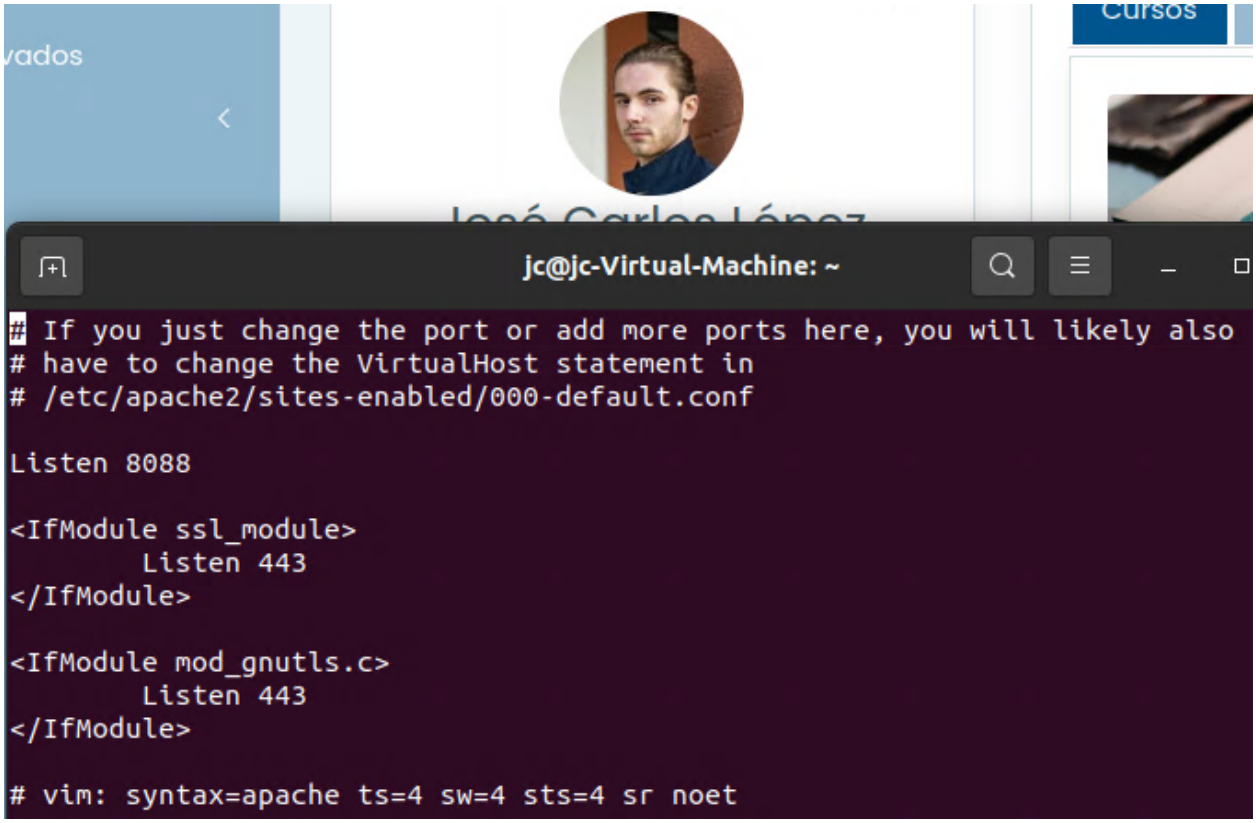
1. Para comprobar que el servidor está funcionando desde un navegador web, introducimos localhost en la URL.



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

1. Introducir el comando sudo vi /etc/apache2/ports.conf para cambiar la configuración de los puertos.

2. Editar la línea Listen 80 (por defecto) a Listen 8088

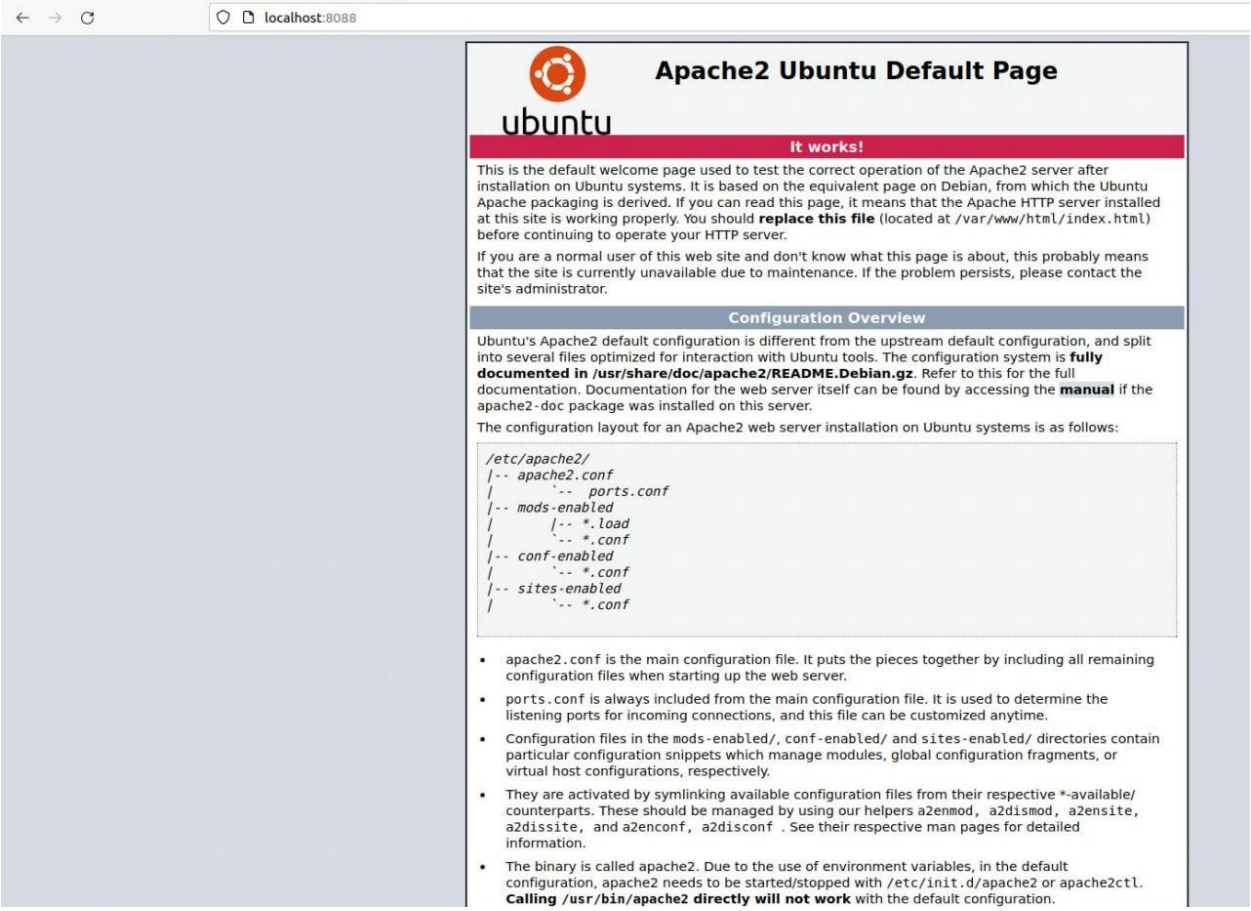


3. Después, accedemos a la configuración por defecto con el comando sudo vi /etc/apache2/sites-enabled/000-default.conf

4. Cambiar <VirtualHost *:80> por <VirtualHost *:8088>

5. Tras ello, hay que reiniciar el servicio para que se ejecuten los cambios con el comando sudo service apache2 restart, visto anteriormente.

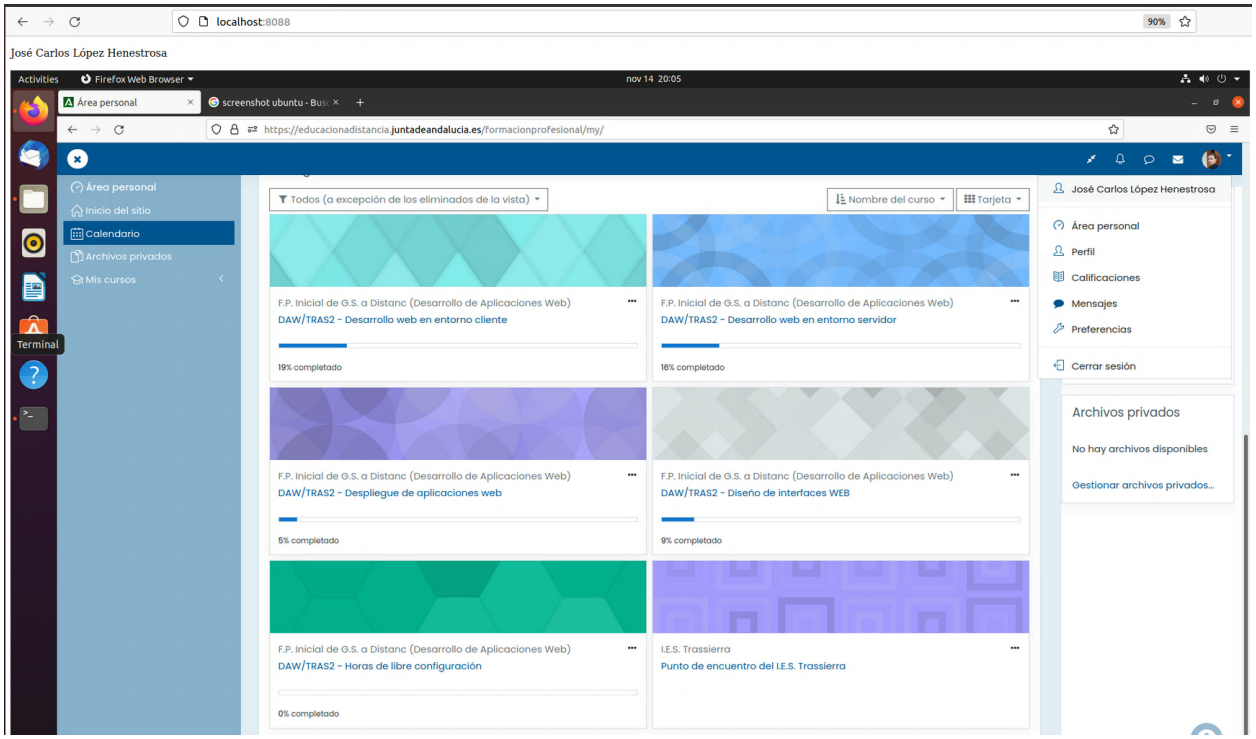
6. Introducimos localhost:8088 en la URL.



Para cambiar la página web por defecto para que aparezcan tus apellidos más un pantallazo de tu inicio del curso:

1. Modificamos el archivo index.html dentro del directorio /var/www/html con el comando sudo nano index.html

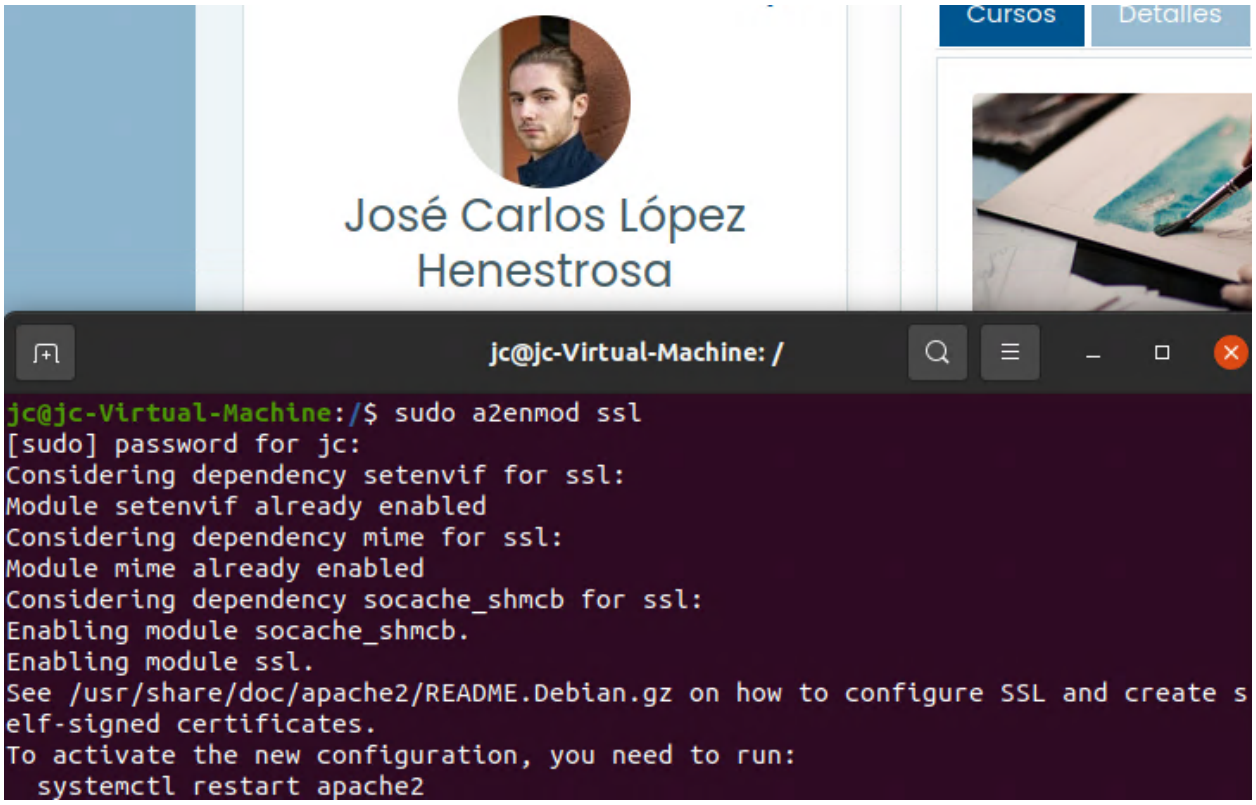
2. El resultado quedaría tal que así:



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).

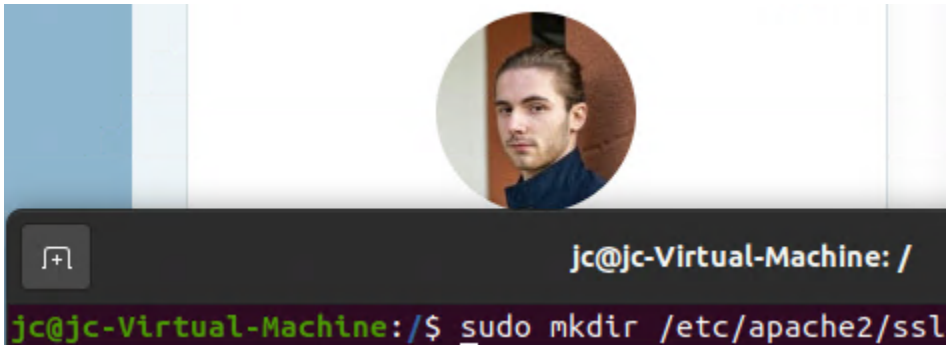
---

1. Tenemos que ejecutar el comando sudo a2enmod ssl para habilitar el módulo mod_ssl

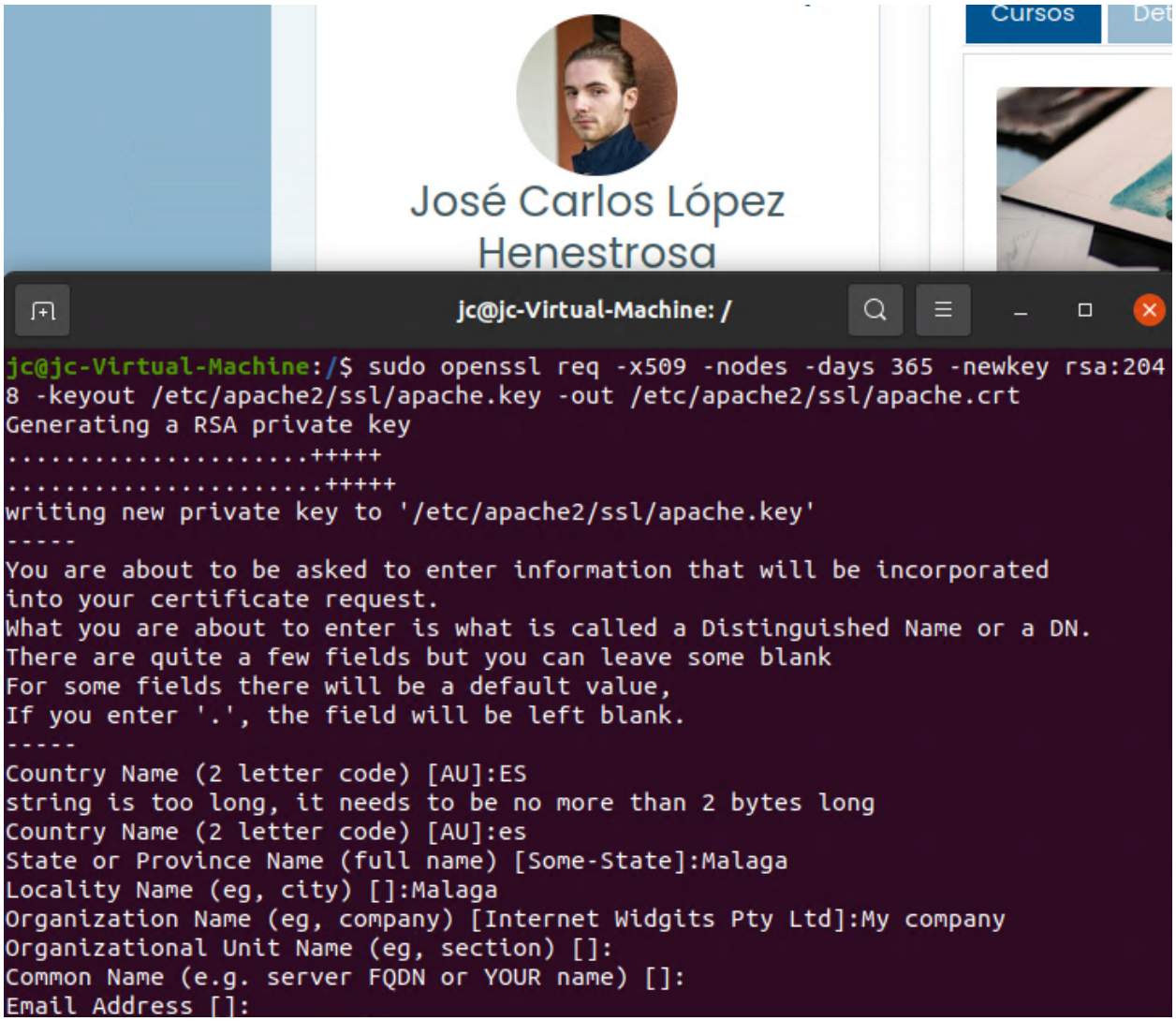


2. Tras ello, reiniciamos el servicio con sudo service apache2 restart

3. Creamos un directorio dentro de la carpeta de configuración de apache con sudo mkdir /etc/apache2/ssl



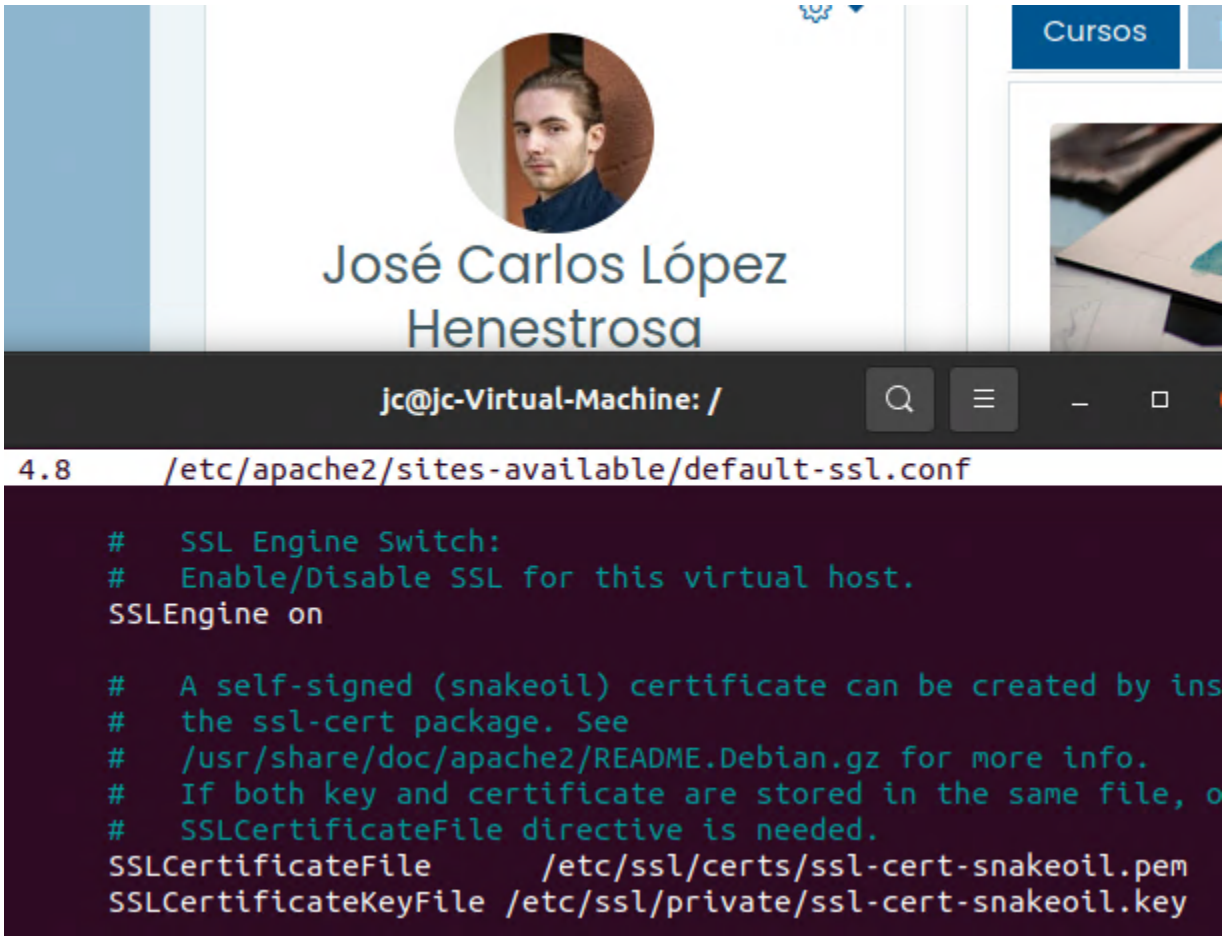
4. Creamos la clave RSA, necesaria para el certificado, con el comando sudo openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout /etc/apache2/ssl/apache.key -out /etc/apache2/ssl/apache.crt



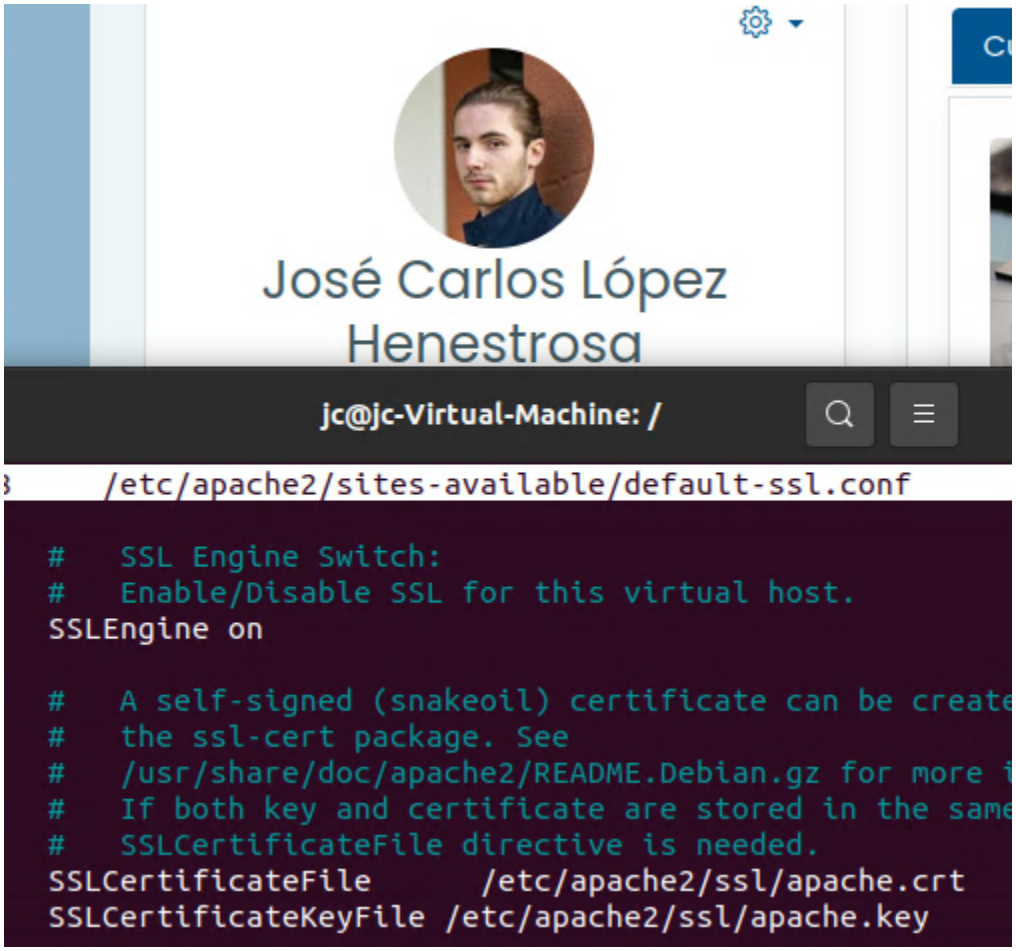
5. Abrimos la configuración por defecto SSL de apache

6. Cambiamos las rutas por defecto de SSLCertificateFile y SSLCertificateKeyFile.

Por defecto:

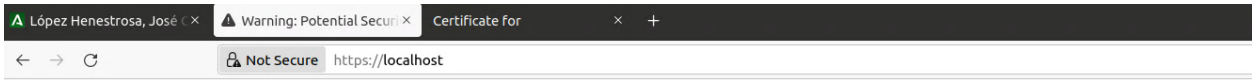


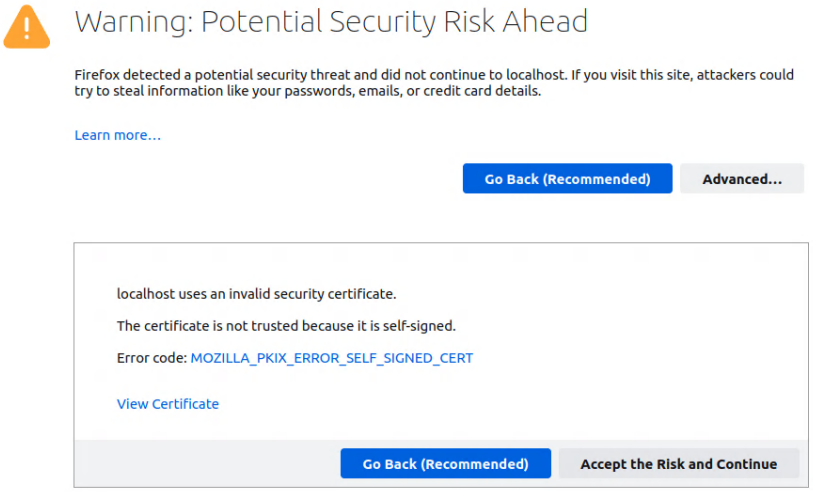
Cambiadas:



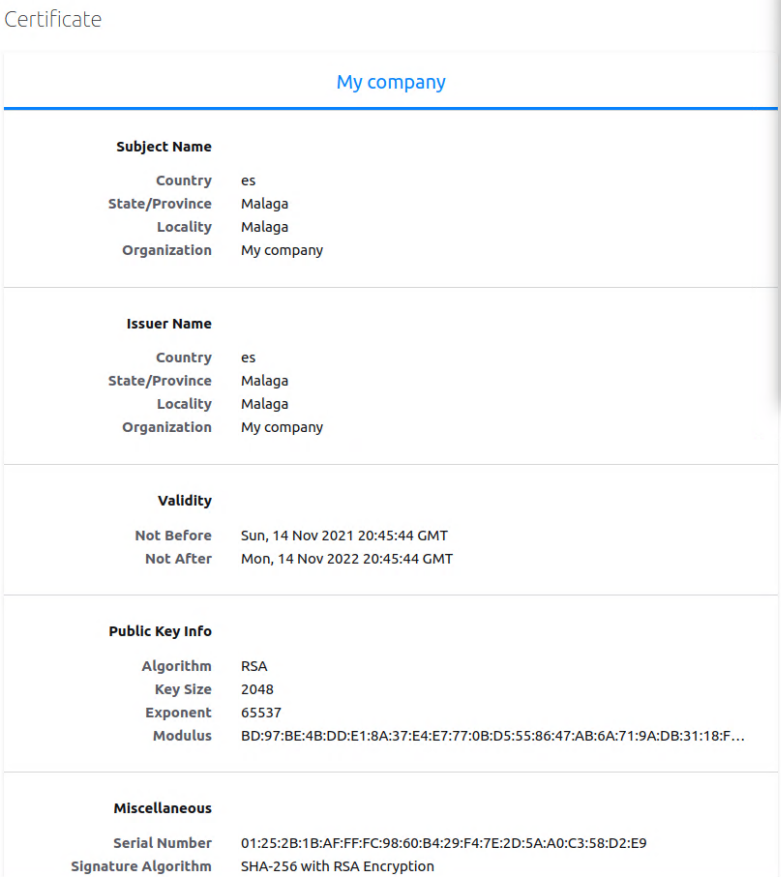
7. Establecemos la configuración SSL por defecto para a2ensite y reinciamos

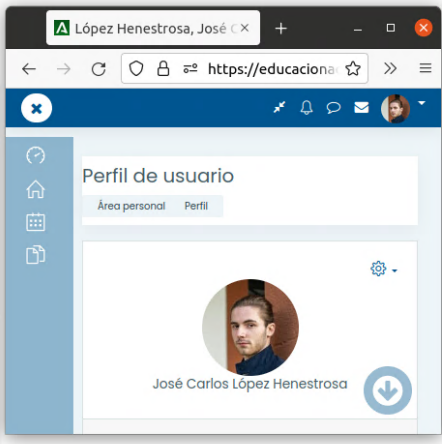
8. Al intentar abrir la página, saldrá un aviso diciendo que la conexión no es segura.





9. Al abrir los detalles del certificado, encontramos que ha sido aplicado correctamente.

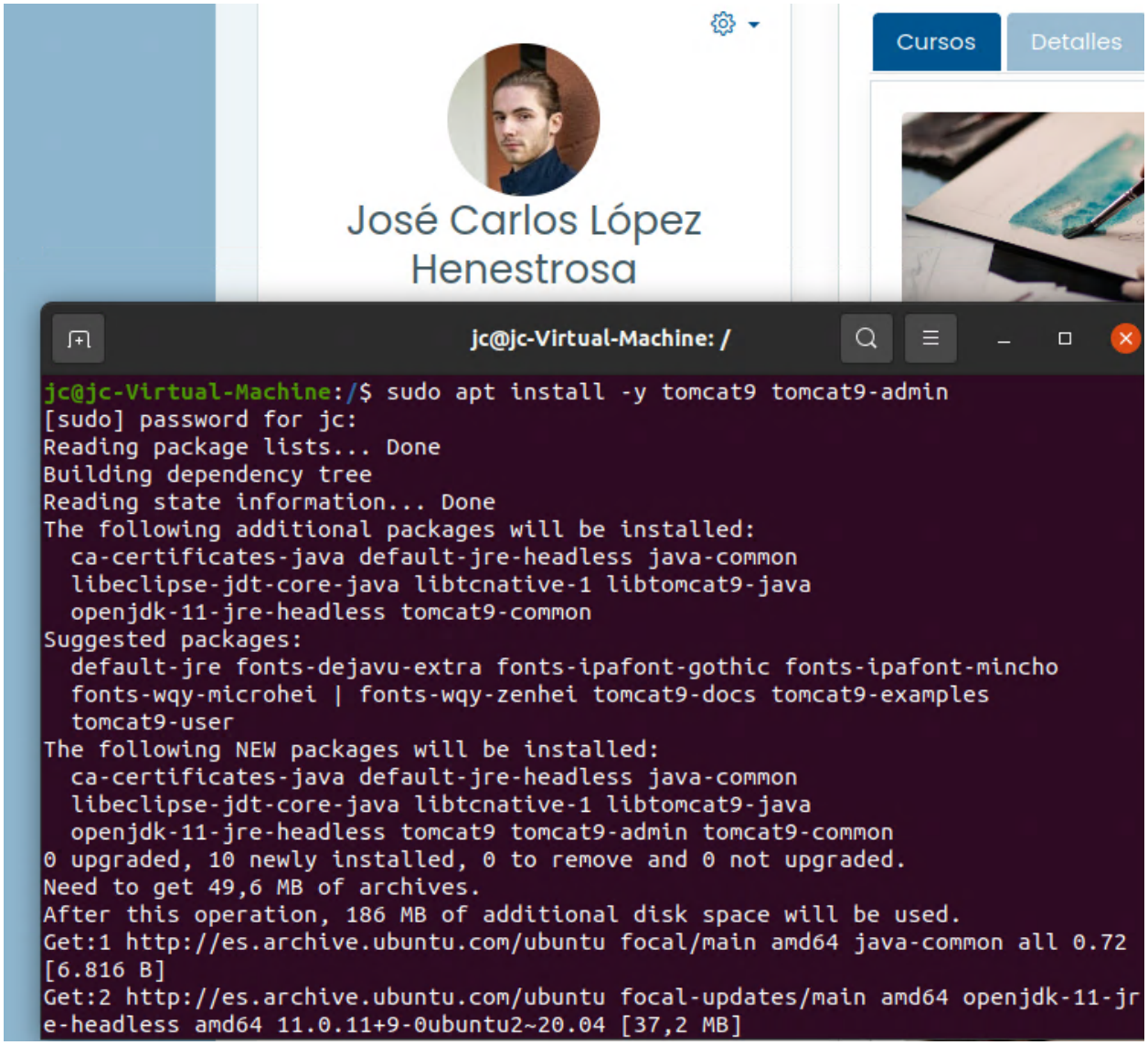




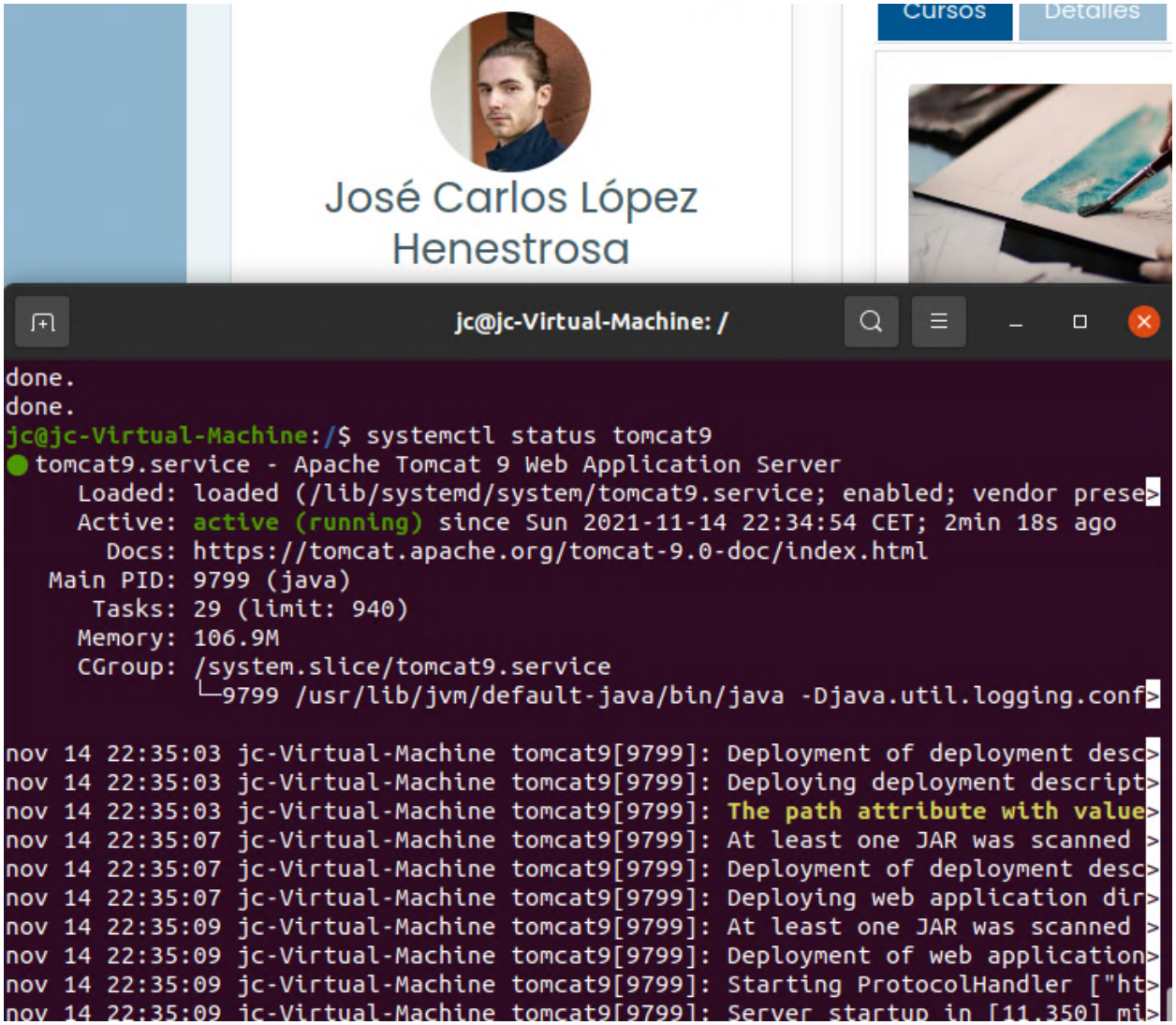
Ejercicio 5: Instalando un servidor de aplicaciones. Partiendo de la instalación de Apache del ejercicio 3, realiza la instalación del servidor de aplicaciones Apache Tomcat (es recomendable instalar una versión lo más reciente posible, la 9 o 10 al menos). Para ayudarte en el proceso de instalación 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.

---

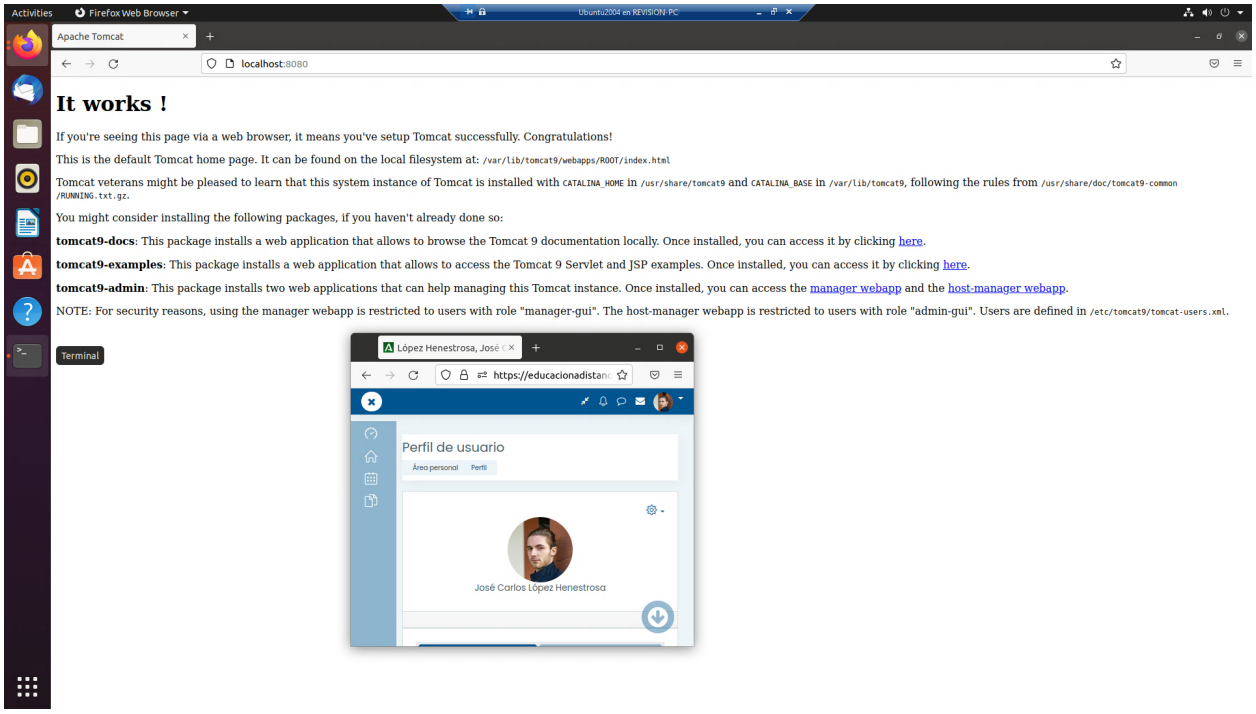
1. Introducimos el comando sudo apt install -y tomcat9 tomcat9-admin para su instalación.



2. Comprobamos el estado ejecutando systemctl status tomcat9



3. También, introducimos localhost:8080 en el navegador para comprobar que la página se muestra correctamente.

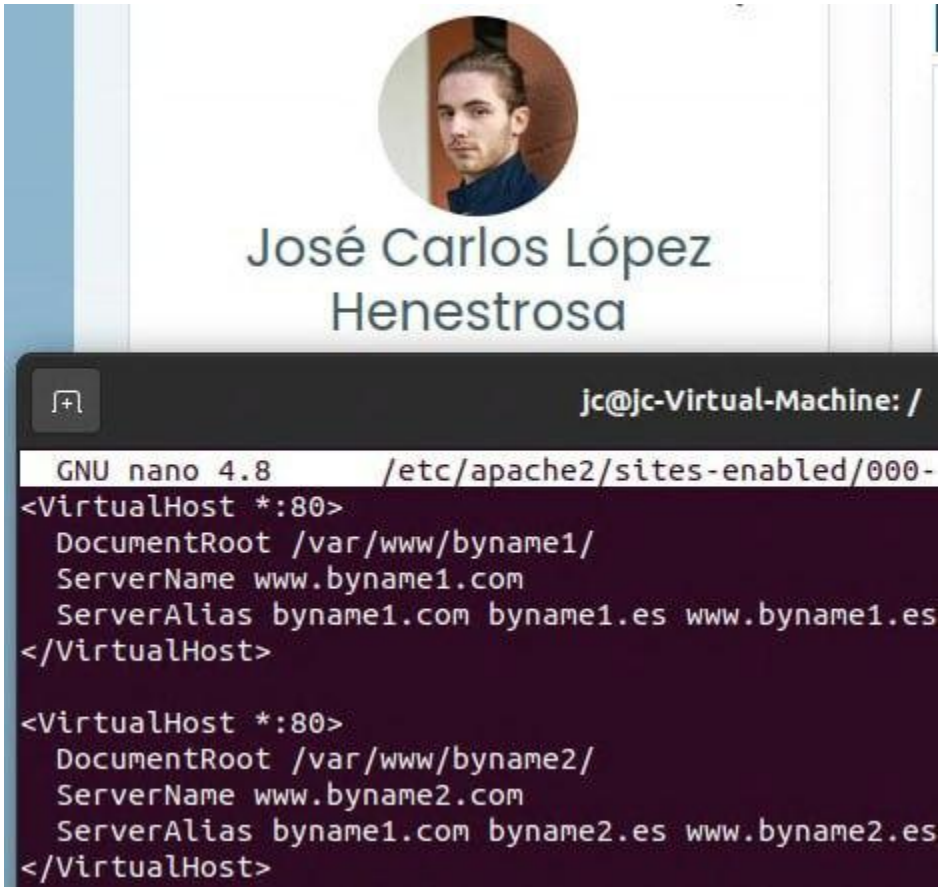
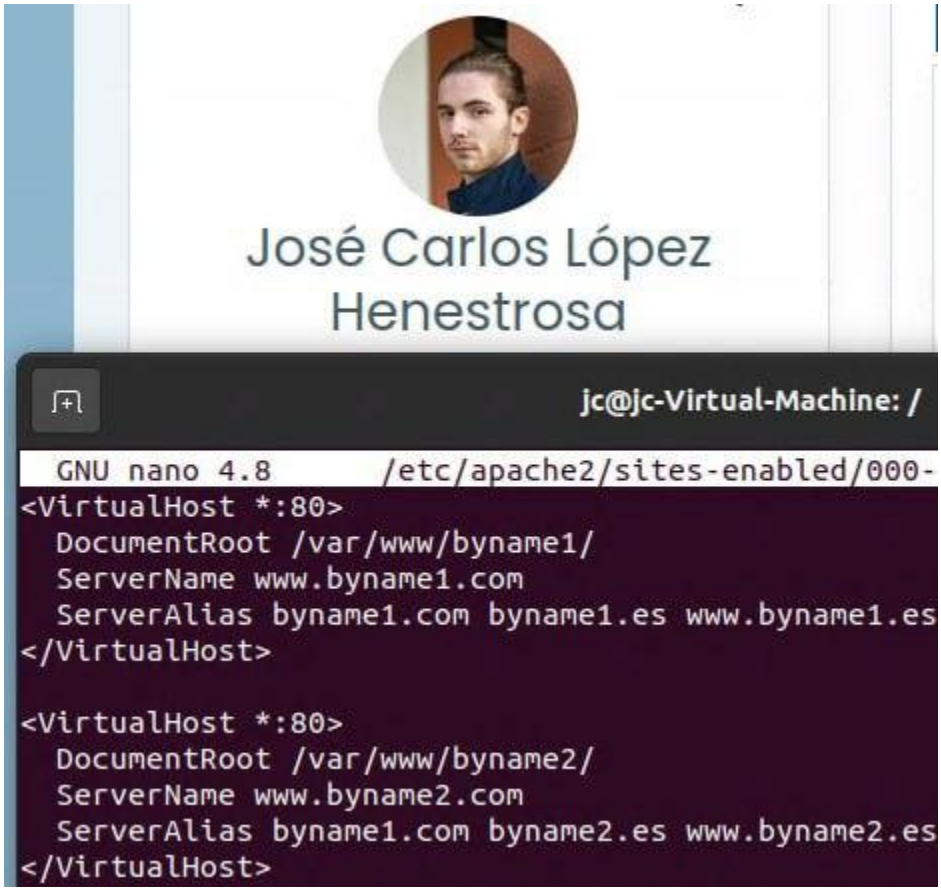


Ejercicio 6: Hosts virtuales. Ya sabemos que Apache permite tener más de un sitio web en un servidor, donde cada sitio puede estar asociado a un dominio diferente. Revisa el punto 5 de la documentación y explica qué es, para que se utiliza y cuál es la diferencia entre virtualhost por nombre y por IP. Pon un ejemplo de configuración de cada uno, usando las directiva 'VirtualHost', incluyendo capturas del fichero de configuración.

El ServerName apunta al dominio y DocumentRoot señala el directorio en el que están almacenadas las páginas del sitio web. Estos dos parámetros pueden ser configurados en un VirtualHost o a nivel global. Un VirtualHost es una entidad de configuración que permite que una sola máquina host se parezca a varias máquinas host.

Los VirtualHost por nombre hacen que varios nombres de máquinas apunten a la misma dirección IP, mientras que los VirtualHost por IP implican que un mismo servidor tenga varias direcciones IP, lo cual es menos común ya que supone un gasto más elevado de recursos (más tarjetas de red y más dificultad de mantenimiento).

Ejemplo de archivo de configuración por nombre:



Ejemplo de archivo de configuración por IP: