

herramienta ec2 de aws

Sistema Informáticos (1º DAM)



Antonio Salces Alcaraz (1º DAM)

C.P.I.F.P. Alan Turing

18/01/2025



Índice

[1. Crear instancia EC2 1](#_Toc188309743)

[2. Acceder a la nueva instancia con PuTTy y operar con ella 5](#_Toc188309744)

[2.1. Acceder con PuTTy 5](#_Toc188309745)

[2.2. Actualizar paquetes y descargar docker.io 7](#_Toc188309746)

[2.3. Pasar web estática con “WinSCP” 8](#_Toc188309747)

[2.4. Arrancar contenedor con “nginx” para poder ver la web en Internet. 10](#_Toc188309748)

[3. Trabajar con la instancia desde la terminal de Linux 12](#_Toc188309749)

[3.1. Convertir clave .ppk a .pem 12](#_Toc188309750)

[3.2. Acceder a la instancia mediante SSH 13](#_Toc188309751)

[3.3. Enviar página web mediante “SCP” 13](#_Toc188309752)

[3.4. Descomprimir y arrancar contenedor 13](#_Toc188309753)

[4. Crear AMI 16](#_Toc188309754)

[4.1. Como crear una AMI de una instancia 16](#_Toc188309755)

[5. Lanzar instancia con la nueva AMI y trabajar con docker-compose y Wordpress. 18](#_Toc188309756)

[5.1. Lanzar instancia con la AMI 18](#_Toc188309757)

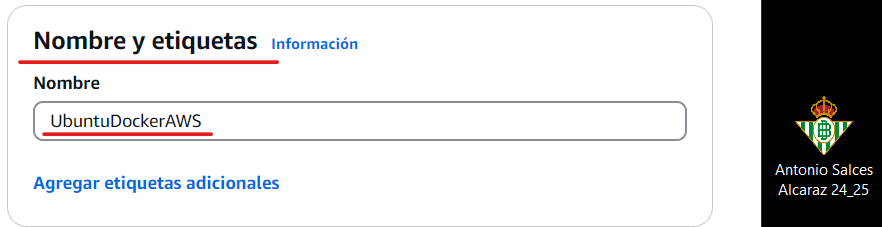
[5.2. Instalar “docker-compose” y desplegar wordpress en la nueva instancia 18](#_Toc188309758)

# Crear instancia EC2

En este primer ejercicio, tenemos que crear una instancia en EC2 con las siguientes características:

* **Nombre**: UbuntuDockerAWS
* **Plataforma**: Ubuntu
* **Tipo de instancia**: t2.micro
* **Acceso por clave**: vockey.ppk / vockey.pem
* **Configuración del firewall**: permitir SSH (22), HTTP (80 y 8080) y HTTPS (443)
* **Almacenamiento**: 8 GB (por defecto)

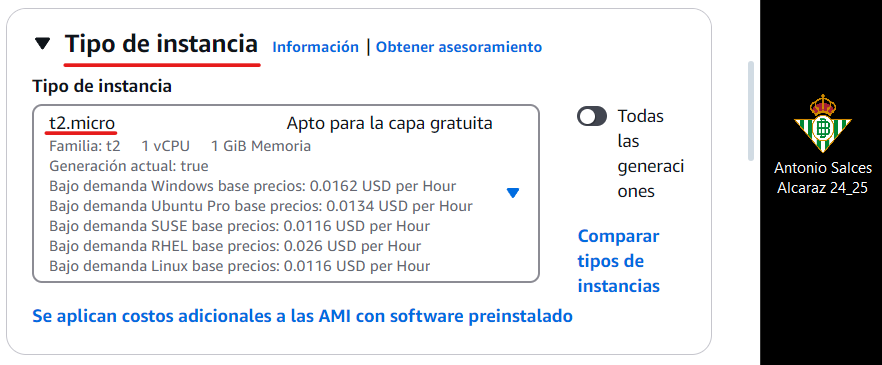
En la pestaña “Nombre y etiquetas”, elegiremos el nombre.

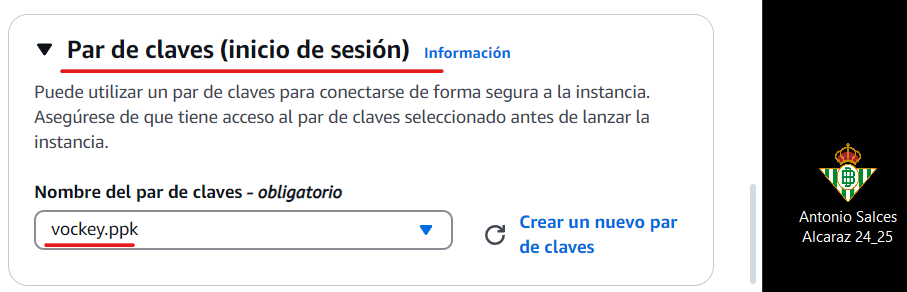


En la pestaña “Imágenes de aplicaciones y sistemas operativos (Imagen de máquina de Amazon”, elegiremos el Sistema Operativo.

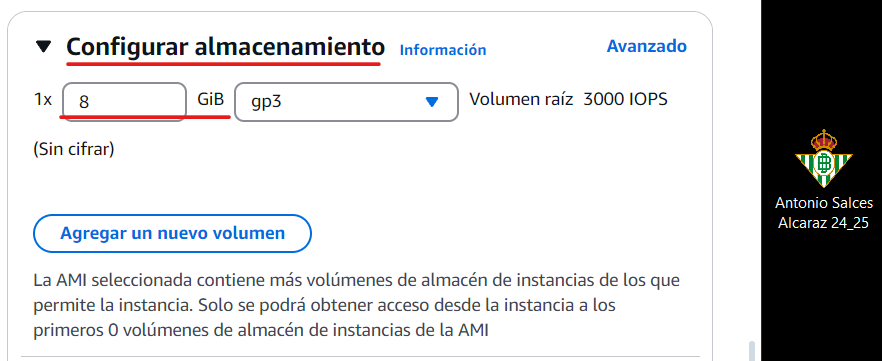


En la pestaña “Tipo de instancia”, elegiremos el tipo de máquina que queremos dentro de las opciones de Amazon.

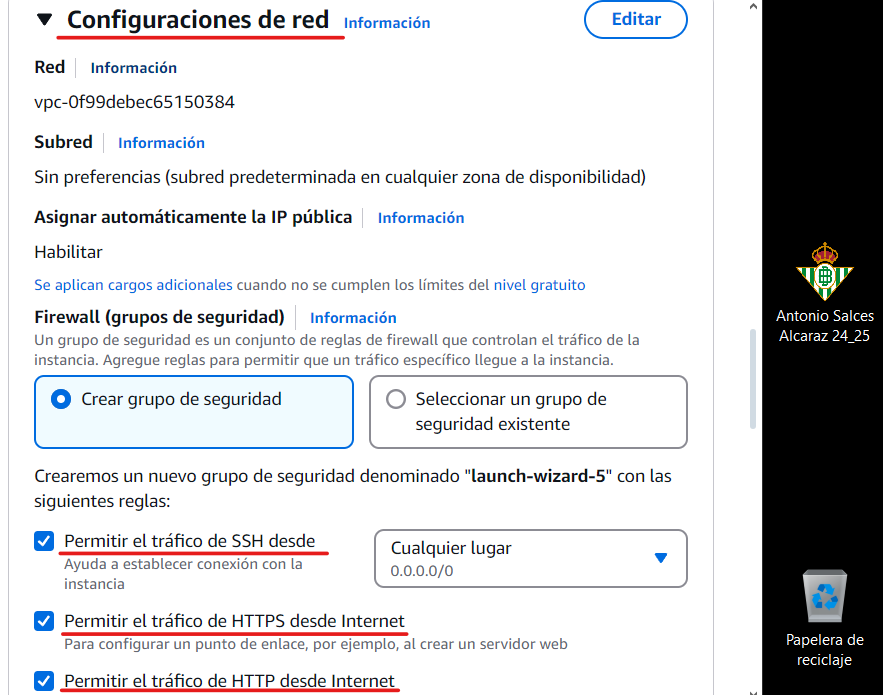


En la pestaña “Par de claves (inicio de sesión)”, elegiremos la clave a utilizar. 

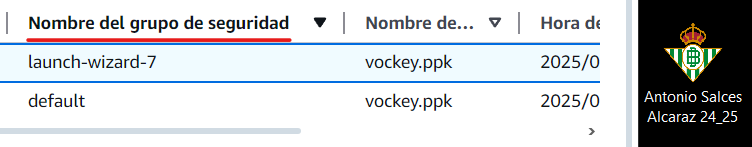
En la pestaña “Configurar almacenamiento”, elegiremos el almacenamiento que queramos.



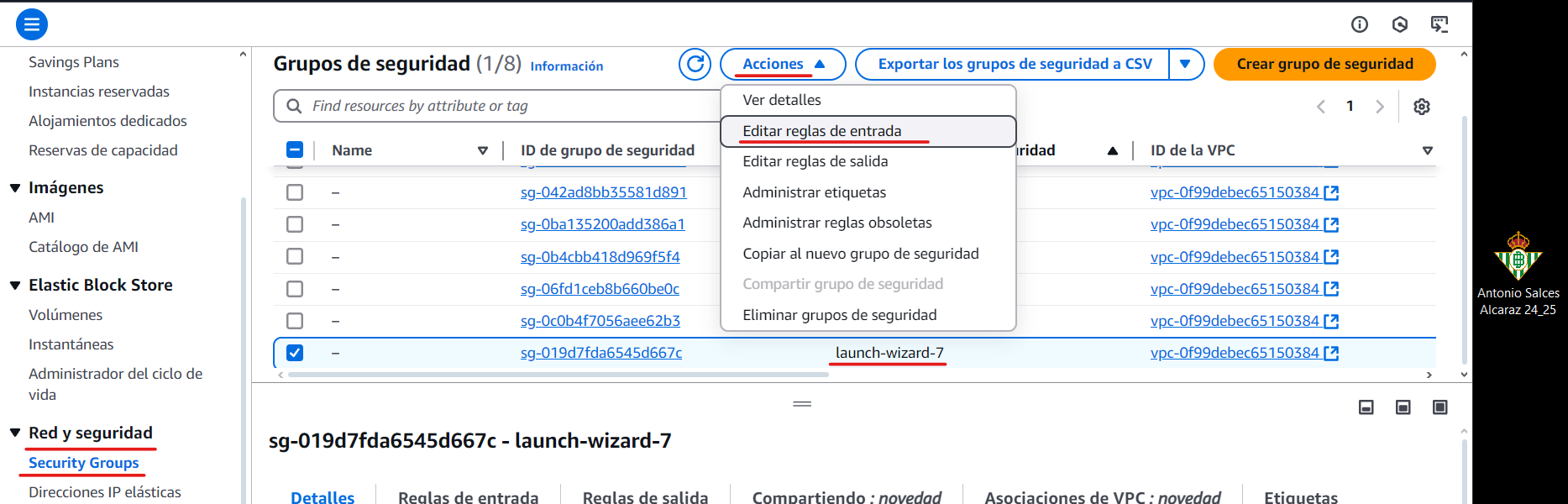
En la pestaña “Configuraciones de red”, podremos activar SSH, HTTP y HTTPS activando las tres casillas.



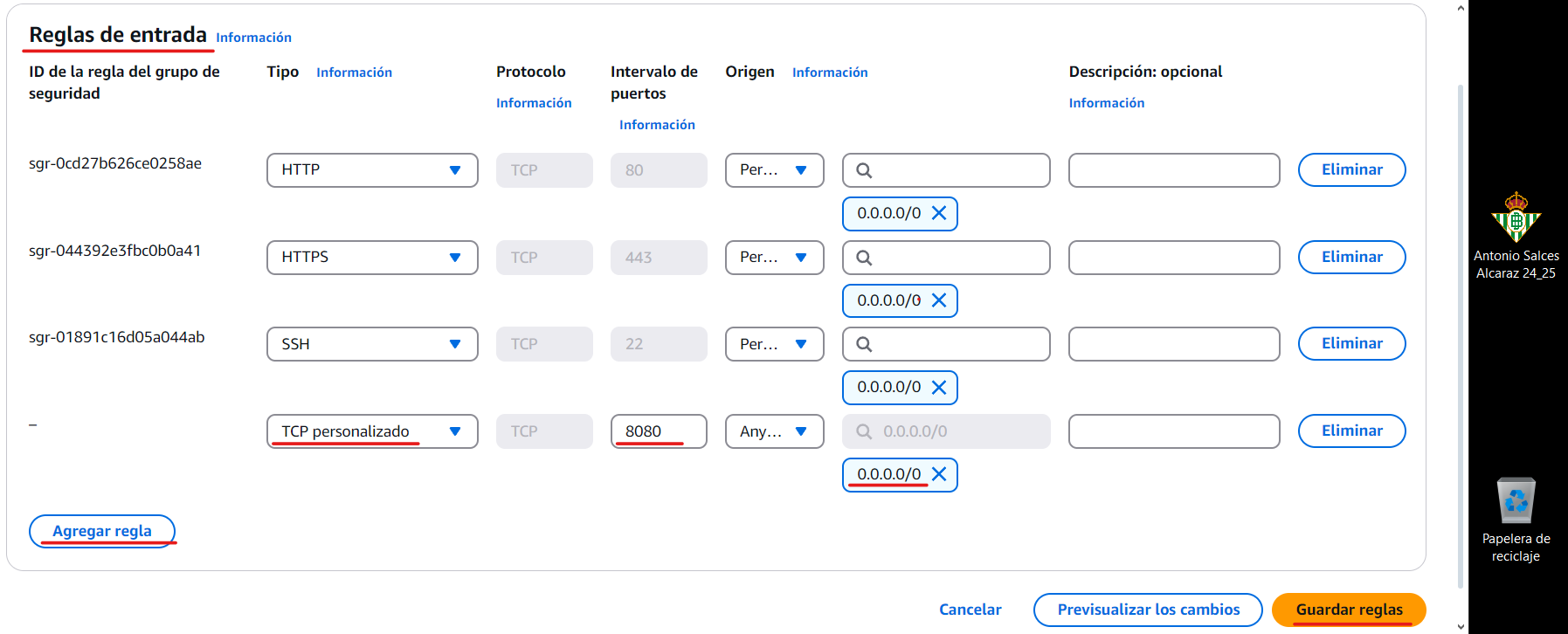
Para activar el puerto 8080 en la instancia, tenemos primero que saber el grupo de seguridad de la misma. Para ello, iremos a la pantalla donde se ven nuestras instancias, y buscaremos “Nombre del grupo de seguridad”.



Ahora nos digiremos a “Red y seguridad > Security Groups”, seleccionamos el grupo de seguridad de la instancia, y pulsaremos sobre “Acciones > Editar reglas de entrada”.



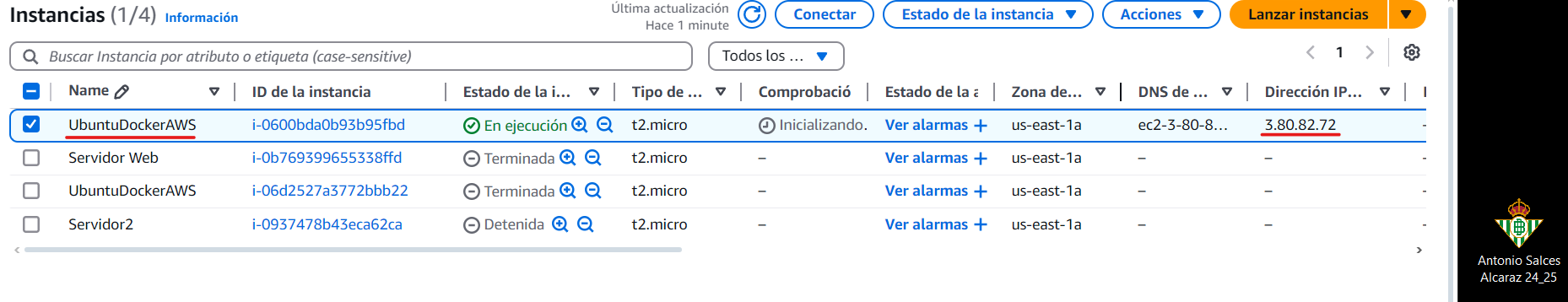
Tendremos que hacer click sobre “Agregar regla” y, en el menú desplegable, seleccionar TCP personalizado, pondremos entonces el puerto deseado, y pondremos las IPs válidas para conectarnos a ese puerto. Si ponemos “0.0.0.0/0”, podremos conectarnos desde cualquier IP. Cuando lo hayamos puesto todo, clickaremos sobre “Guardar regla”.



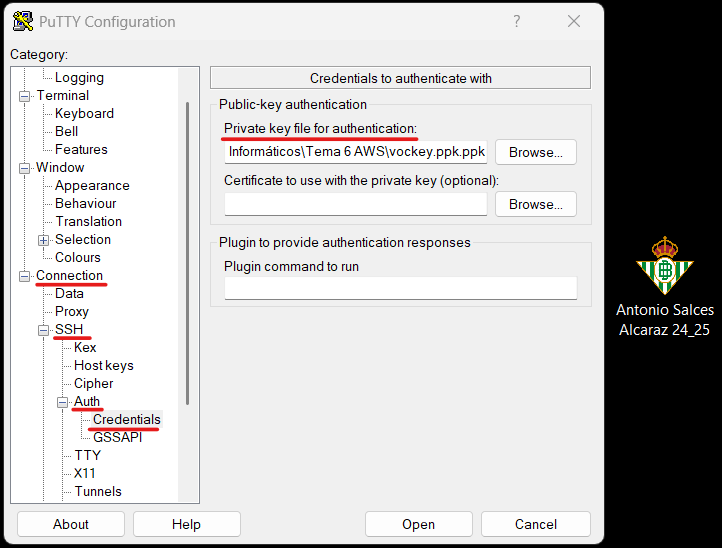
# Acceder a la nueva instancia con PuTTy y operar con ella

## Acceder con PuTTy

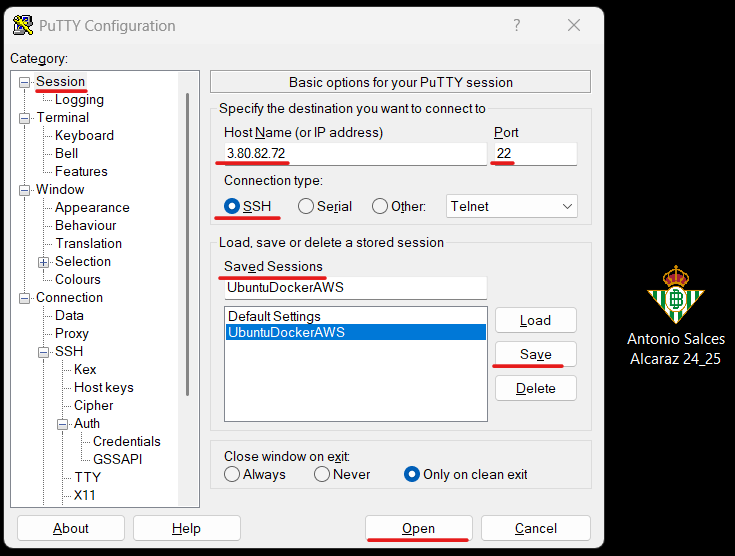
En primer lugar, debemos de saber la IP que tiene nuestra instancia en el momento de conectarnos mediante “PuTTy”. Para ellos, vamos a nuestras instancias, y buscamos la IP de la instancia.



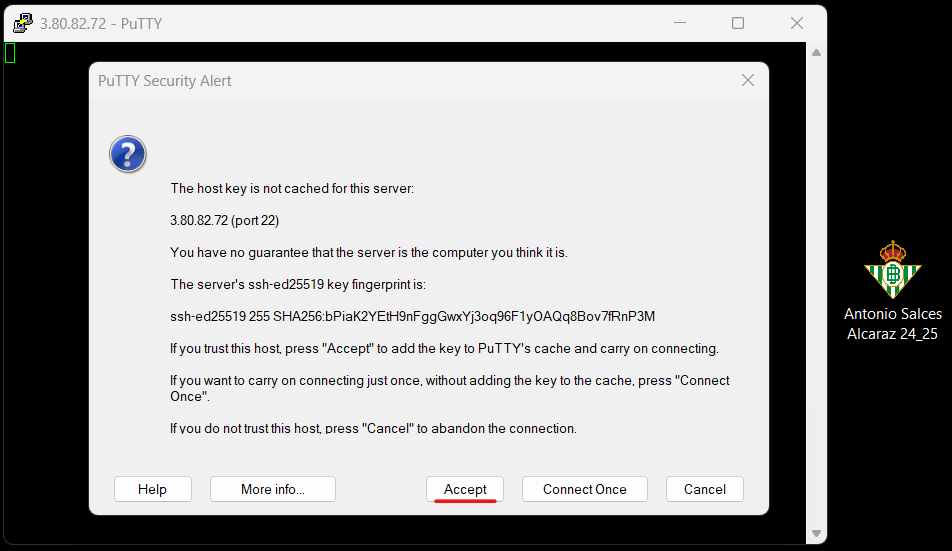
Ahora abrimos “PuTTy”. Iremos a la sección “Connection > SSH > Auth > Credentials”, y una vez dentro, pondremos la clave “vockey.ppk” dentro de “Private key file for authentication”.

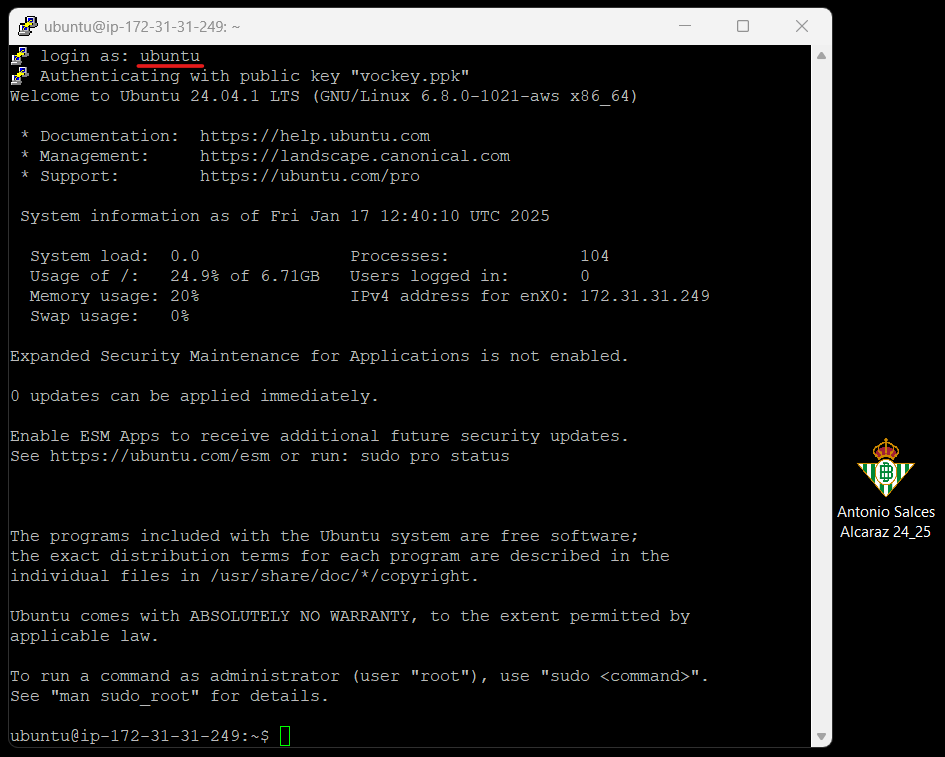


Ahora vamos a la pestaña “Session”. Primero pondremos la IP dentro de “Host Name (or IP address)”, y el puerto 22 dentro de “Port”. Seleccionaremos SSH dentro de “Connection type”. Para guardar la configuración para futuras conexiones, pondremos el nombre deseado de “Saved Sessions” y clickamos sobre “Save”. Una vez hayamos guardado la configuración, haremos click en “Open” para acceder a la instancia.



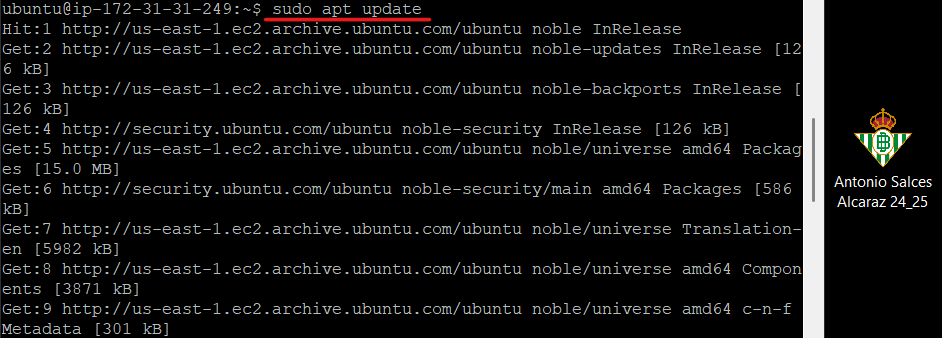
Cuando entremos por primera en cualquier instancia, nos saltará un aviso de seguridad. Clickando sobre “Accept”, podremos acceder a la instancia.



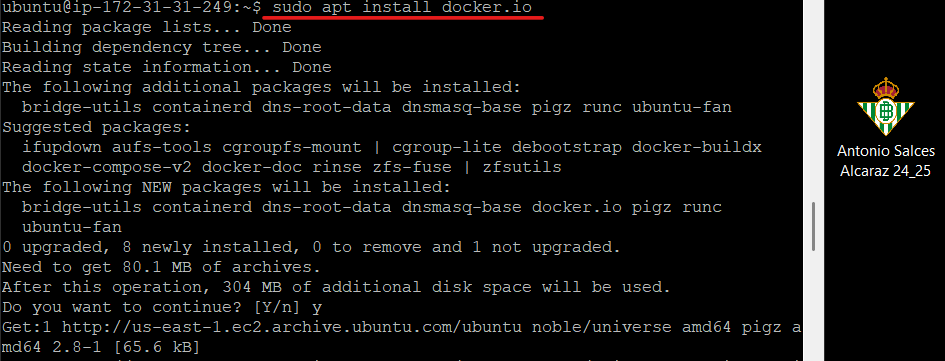
Una vez accedemos a la instancia, tendremos que loguearnos como “ubuntu", y PuTTy se encargará de enviar las claves para poder acceder. Como podemos ver en la imagen, ya podemos acceder y utilizar comando en la máquina. 

## Actualizar paquetes y descargar docker.io

Primero, debemos de actualizar los paquetes con “***sudo apt update***”

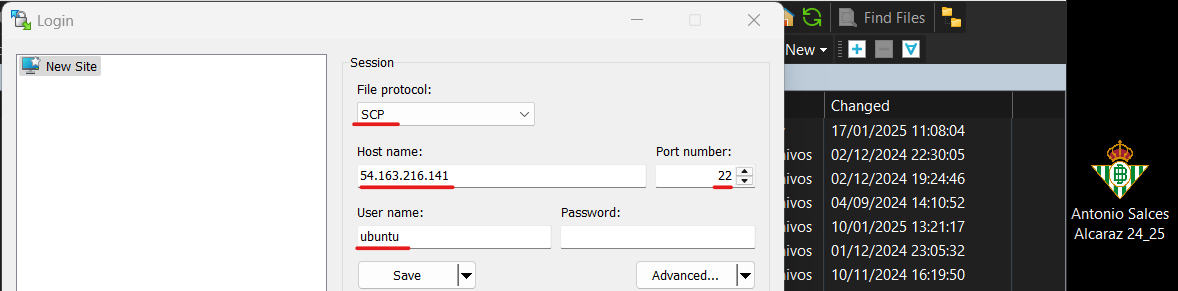


Ahora utilizaremos “***sudo apt install <paquete>***” para instalar “docker.io”.

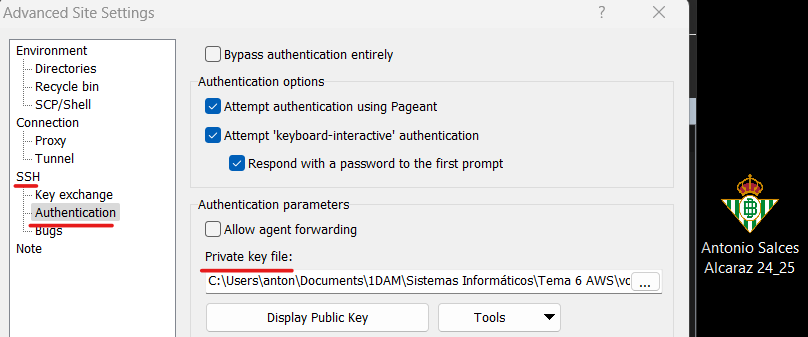


## Pasar web estática con “WinSCP”

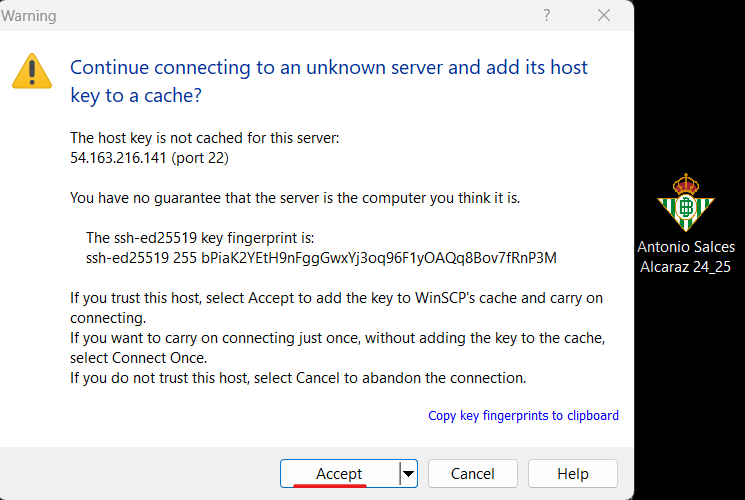
Abriremos “WinSCP”. En “File protocol” elegiremos “SCP”, en “Host name” pondremos la IP, y en “User name” pondremos “Ubuntu”. Ahora nos iremos a la sección “Advanced…”.



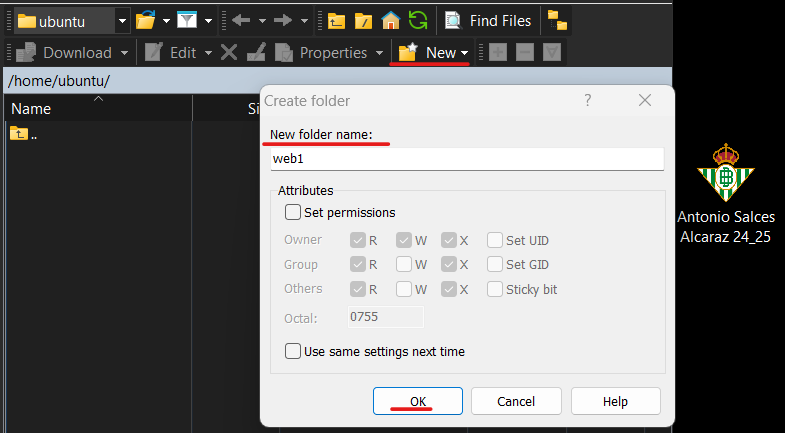
Nos dirigiremos a “SSH > Authentication”, y en “Private key file:” pondremos la clave generada anteriormente. Tras colocarla, podremos conectarnos.



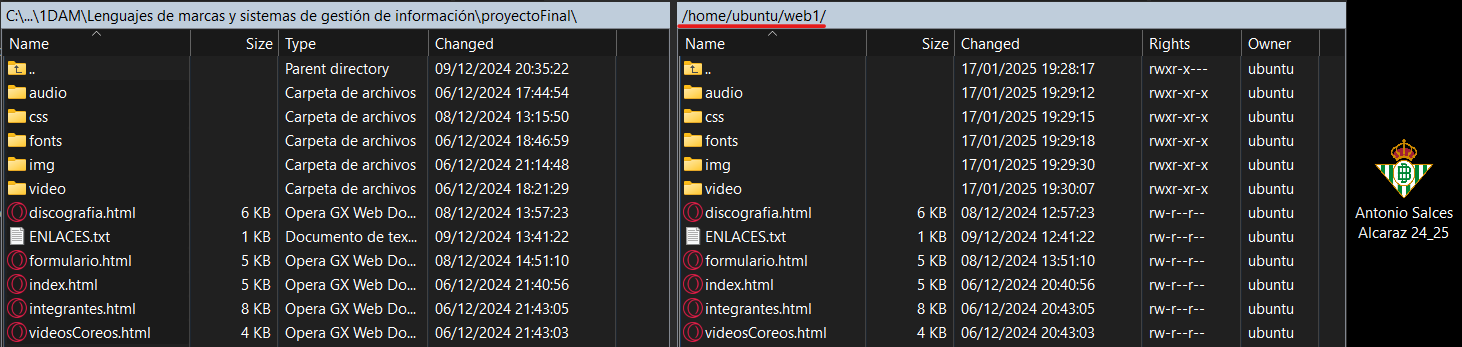
Al igual que “PuTTy”, nos dará un warning de seguridad. Simplemente tenemos que darle a “Accept”.



Ahora clickaremos sobre el botón señalado para crear una nueva carpeta dentro de “/home/ubuntu", y en “New folder name:” pondremos el nombre deseado, y daremos a “OK”.

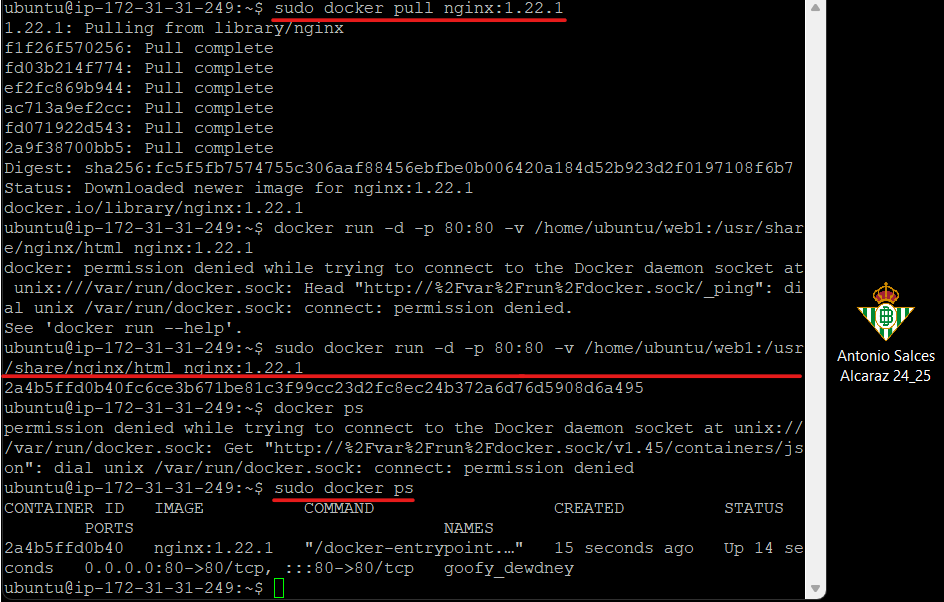


Ahora nos dirigiremos a la carpeta de nuestro S.O. que tenga los archivos de la página web a publicar, entonces, los seleccionaremos y los pegaremos a “/home/ubuntu/web1” de nuestra instancia.

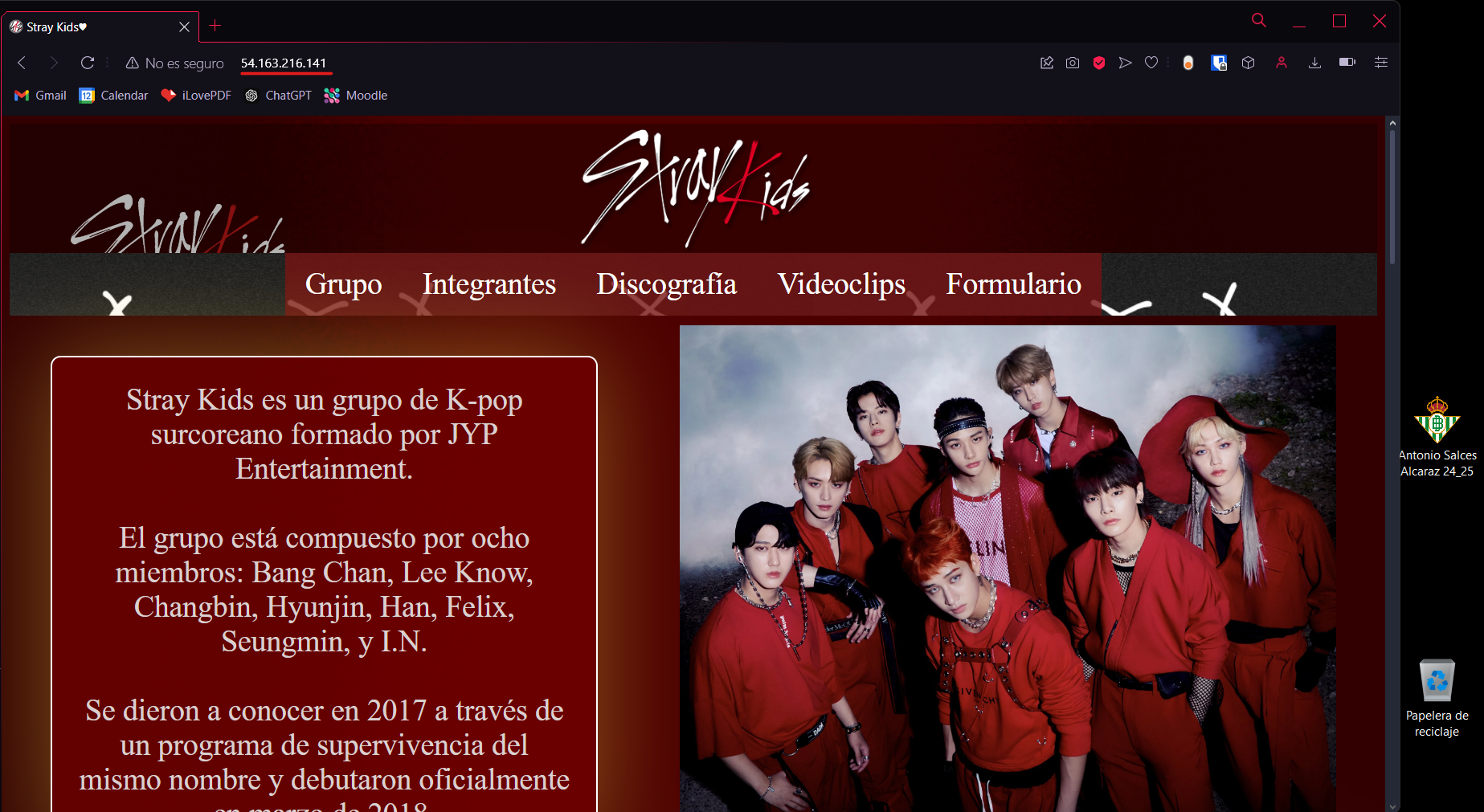


## Arrancar contenedor con “nginx” para poder ver la web en Internet.

En primer lugar, haremos “***sudo docker pull <imagen>***” para descargar “nginx:1.22.1”. Una vez lo hayamos descargado, utilizaremos el comando “***sudo docker run [-d] [-p <puerto>] [-v <ruta de ubuntu:ruta de nginx>] <imagen>***” para arrancar el contenedor y enlazar la carpeta que contiene nuestra web, a la carpeta publica del contenedor. Por último, utilizaremos “***docker ps***” para comprobar que el contenedor está funcionando.



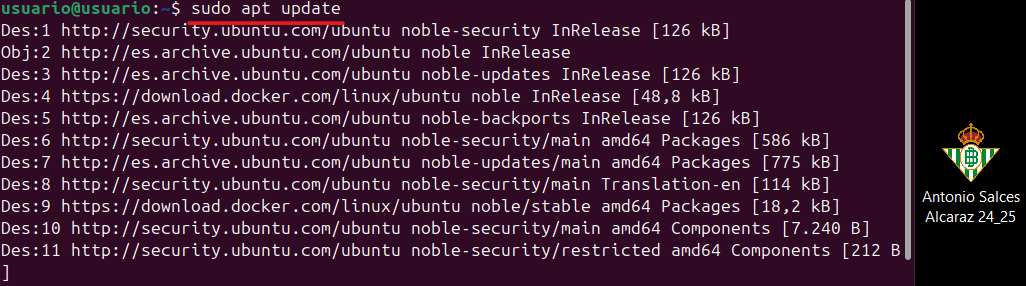
Ahora, si buscamos la IP de nuestra instancia en cualquier buscador, podremos ver la página web que hemos subido a Ubuntu.



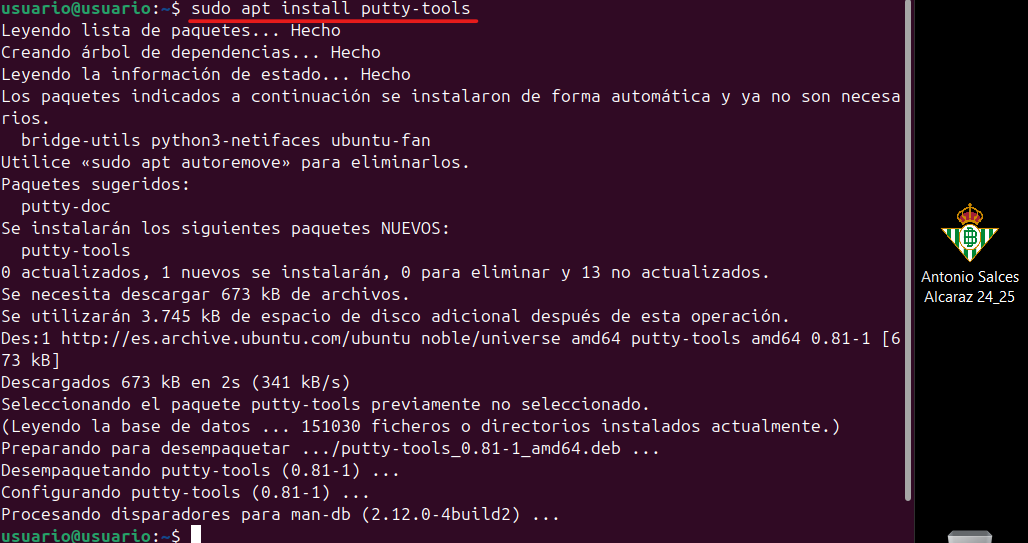
# Trabajar con la instancia desde la terminal de Linux

## Convertir clave .ppk a .pem

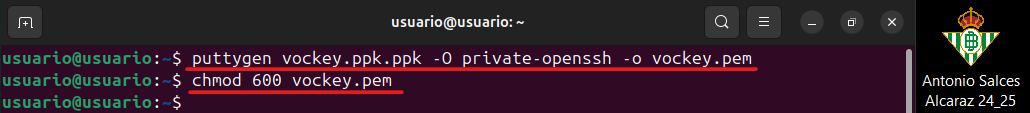
En primer lugar, debemos de convertir el archivo “.ppk” en “.pem” para conectarnos en la terminal de Ubuntu. Para ello, actualizaremos el sistema con “***apt update***” como primer paso.



Tras actualizar los paquetes y las dependencias, instalaremos el paquete “putty-tools” con “***apt install <paquete>***”.

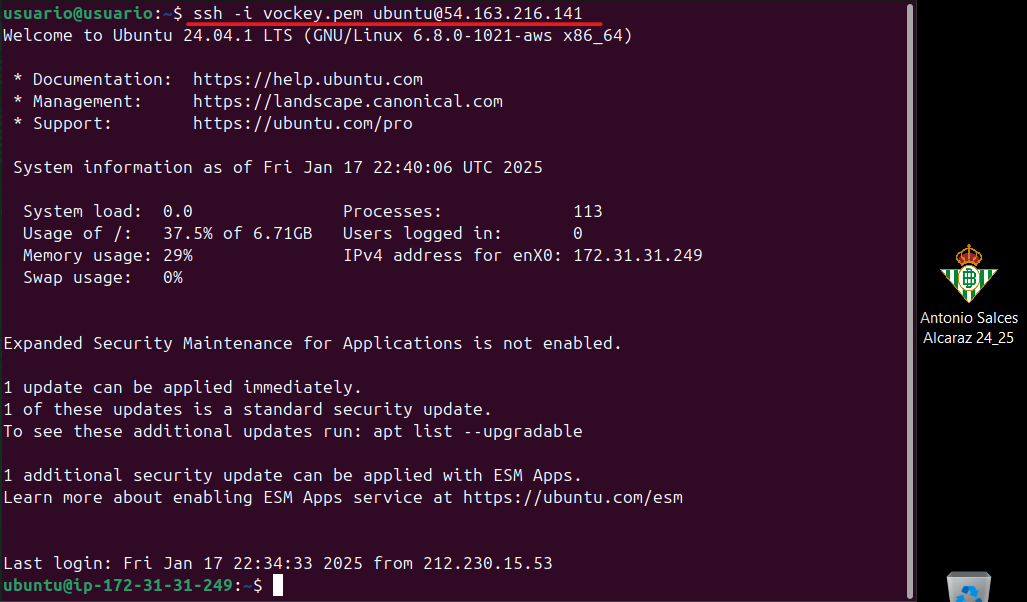


Tras instalar el paquete mencionado, con el comando “***puttygen <archivo de origen> -O private-openssh -o <archivo de destino>***” podremos convertir la antigua clave .ppk en una nueva .pem. Además, le tendremos que dar pocos permisos porque Ubuntu tiene unas directivas de seguridad estrictas con SSH. Por lo tanto, tendremos que ejecutar el comando “***chmod 600 <archivo>***” para que no de problemas con los permisos.



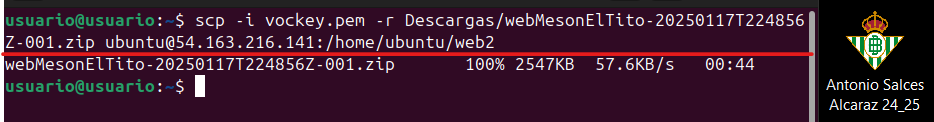
## Acceder a la instancia mediante SSH

Para acceder a la instancia, debemos utilizar el comando “***ssh <-i clave> <usuario>@<IP>***”.



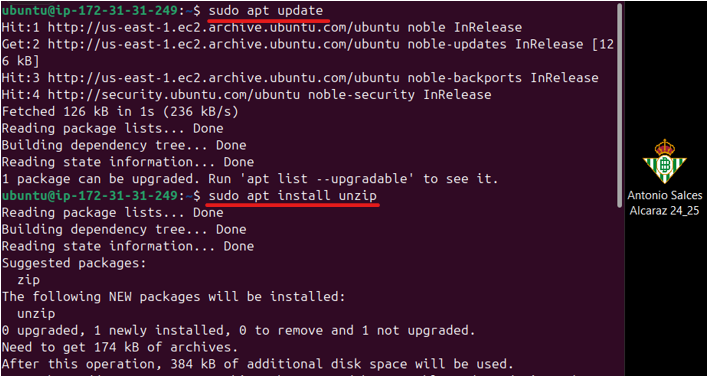
## Enviar página web mediante “SCP”

Para enviar la web mediante SCP, debemos utilizar “***scp <-i clave> <-r ruta local> <usuario@IP>:<ruta>***”. Como podemos ver en la imagen, envío un .zip a la máquina, por lo que luego tendremos que descomprimirlo.

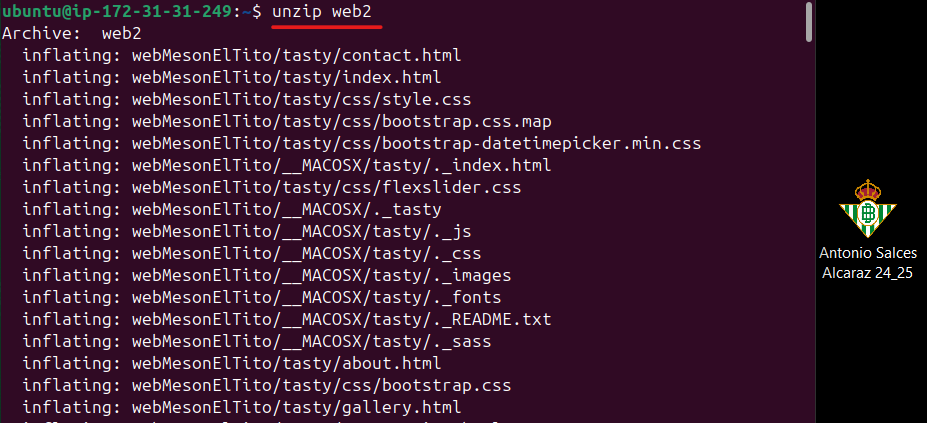


## Descomprimir y arrancar contenedor

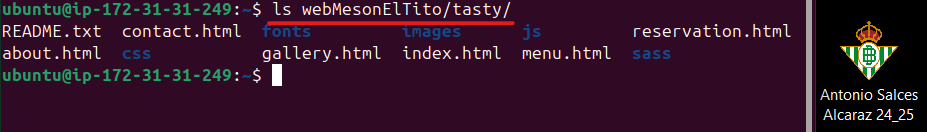
Primero tenemos que instalar el paquete “unzip”, por lo que ejecutaremos “***apt update***” y, tras ello, “***apt install <paquete>***”.



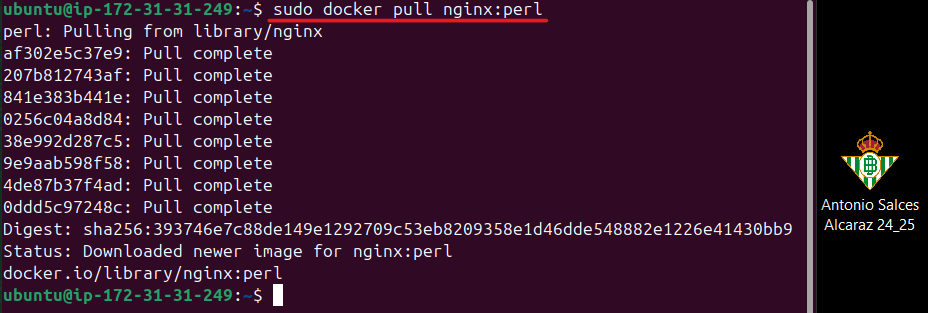
Tras instalar el paquete “unzip”, ejecutaremos “***unzip <archivo>***” para descomprimirlo.



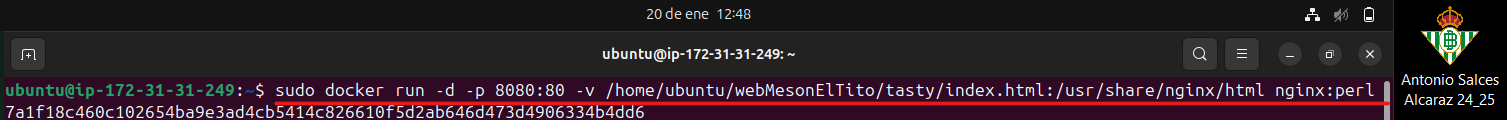
Ahora haremos un “***ls***” por los archivos del archivo descomprimido para comprobar que está todo.



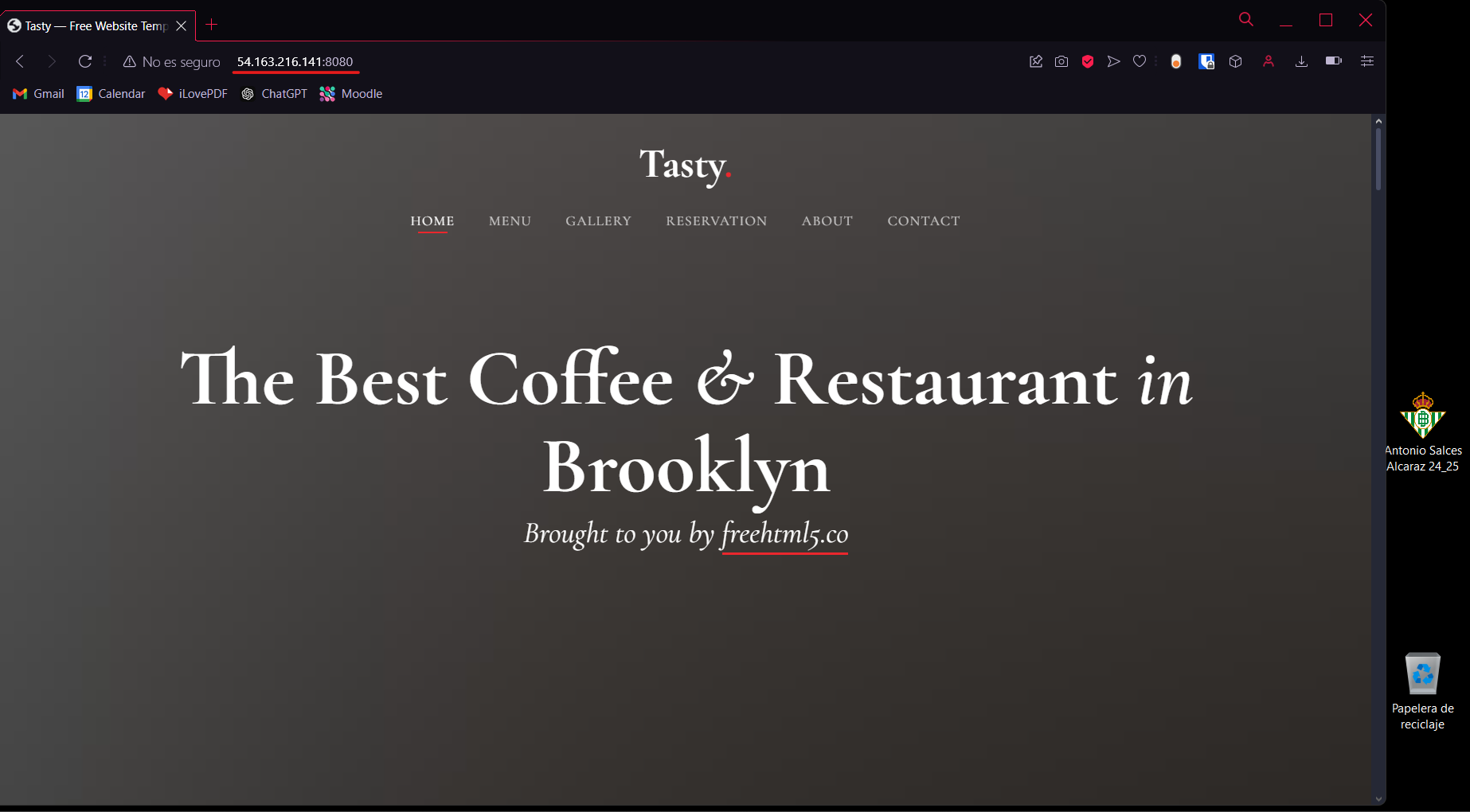
Necesitamos hacer un contenedor con “nginx:perl”, por lo que ejecutaremos el comando “***docker pull <paquete>***”.



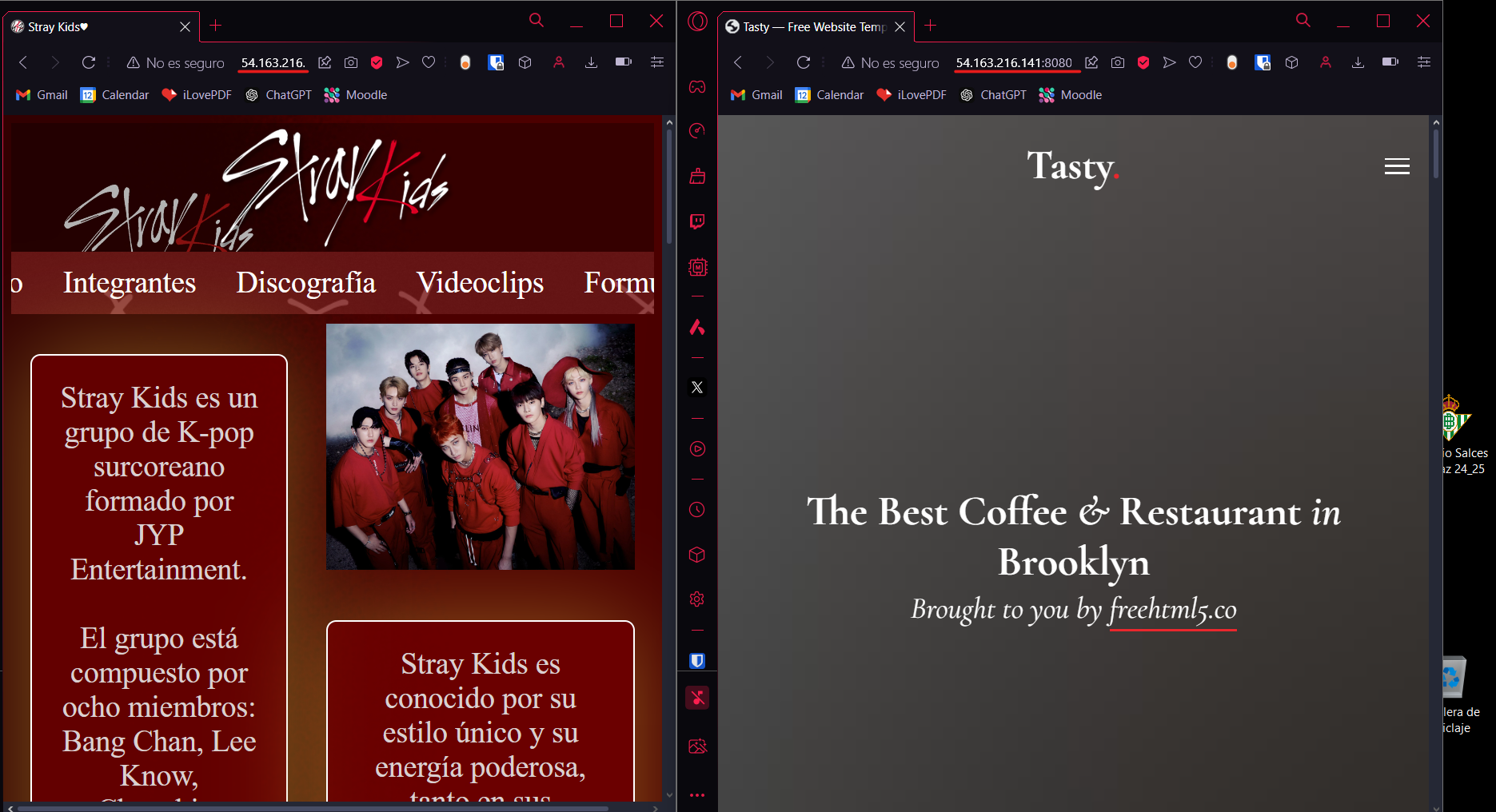
Ahora solo tendremos que arrancar el contenedor con “***docker run [-d] [-p <puerto> [-v <ruta local>:<ruta al contenedor>] imagen***”.



Tras arrancar el contenedor, si accedemos mediante el navegador a la IP por el puerto “8080”, podremos ver la nueva web.



Y aquí podremos ver como en la misma instancia, podemos tener dos páginas web en dos puertos diferentes.

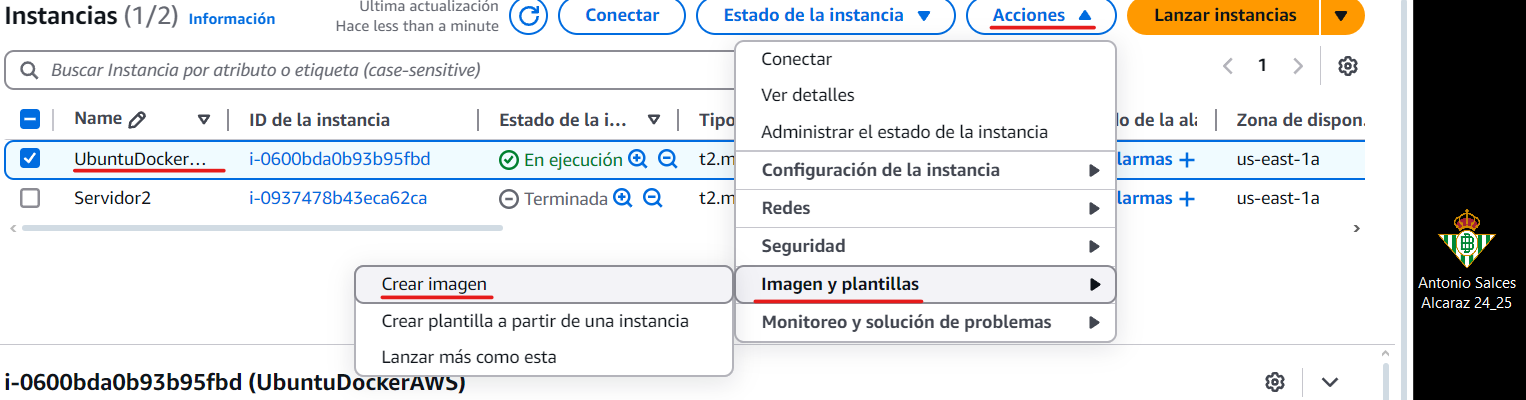


# Crear AMI

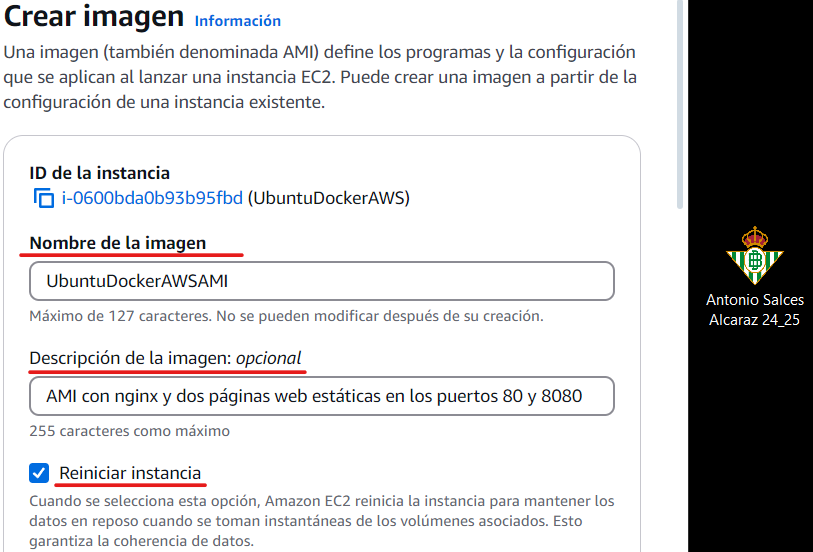
Una **AMI** es una plantilla preconfigurada que proporciona la información necesaria para lanzar instancias EC2 en **AWS. Está incluye información sobre el S.O., software preinstalado, y permisos.**

## Como crear una AMI de una instancia

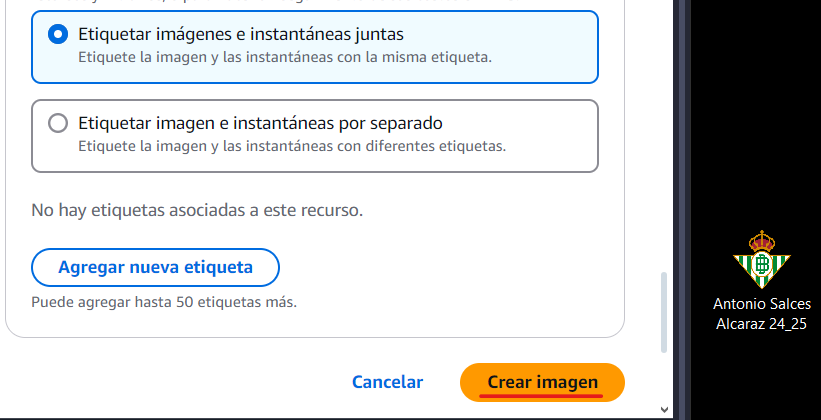
En primer lugar, debemos de ir a la pantalla donde tenemos todas nuestras instacias. Marcaremos la instancia de la que queremos hacer la AMI, clickaremos sobre “Acciones”, nos iremos a “Imagen y plantillas”, y seleccionaremos “Crear imagen”.



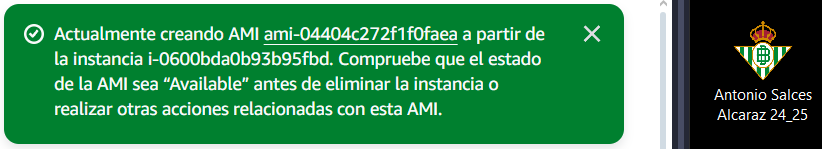
Ahora pondremos el nombre a la imagen y una descripción si así lo quisiéramos. Marcaremos la opción “Reiniciar instancia” para que no haya problemas con los datos mientras se está creando la AMI.



Ahora bajaremos al final de la pantalla y clickaremos sobre “Crear imagen”.



Y como podemos ver, nos sale un mensaje de que la AMI se está creando.



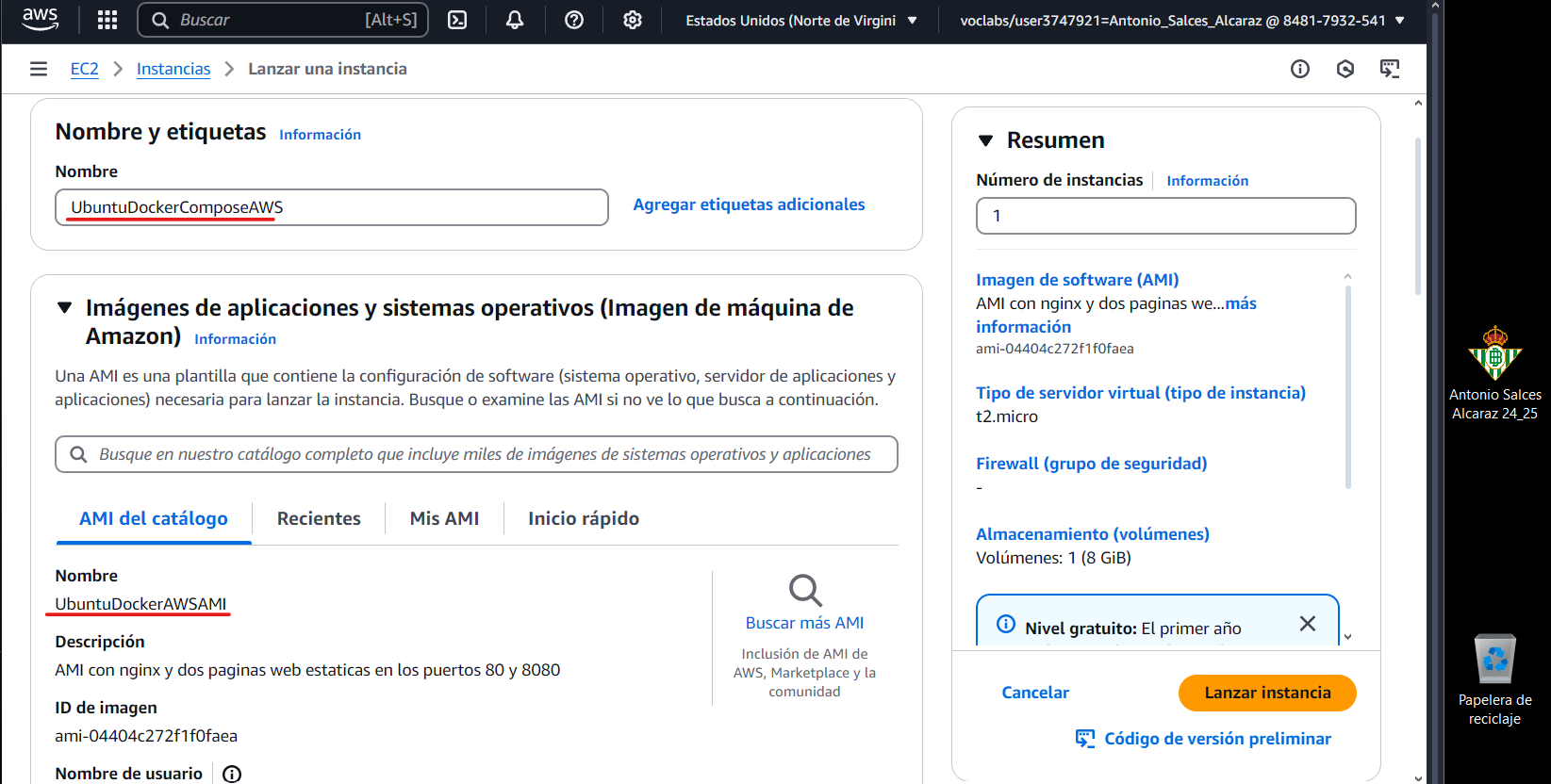
# Lanzar instancia con la nueva AMI y trabajar con docker-compose y Wordpress.

## Lanzar instancia con la AMI

En primer lugar, nos iremos a “Imágenes > AMI” dentro de la barra lateral izquierda. Buscaremos la AMI que acabamos de crear y clickaremos sobre “Lanzar instancia a partir de una AMI”.



Ahora pondremos el nombre y verificaremos que estamos utilizando la AMI creada anteriormente. Configuraremos la máquina y revisaremos toda la configuración al igual que en la primera actividad. Cuando este todo listo, haremos click en “Lanzar instancia”.



## Instalar “docker-compose” y desplegar wordpress en la nueva instancia

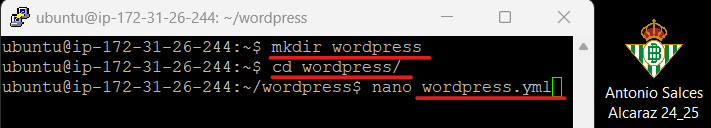
En primer lugar, tenemos que actualizar los paquetes del sistema con “***apt install***”.



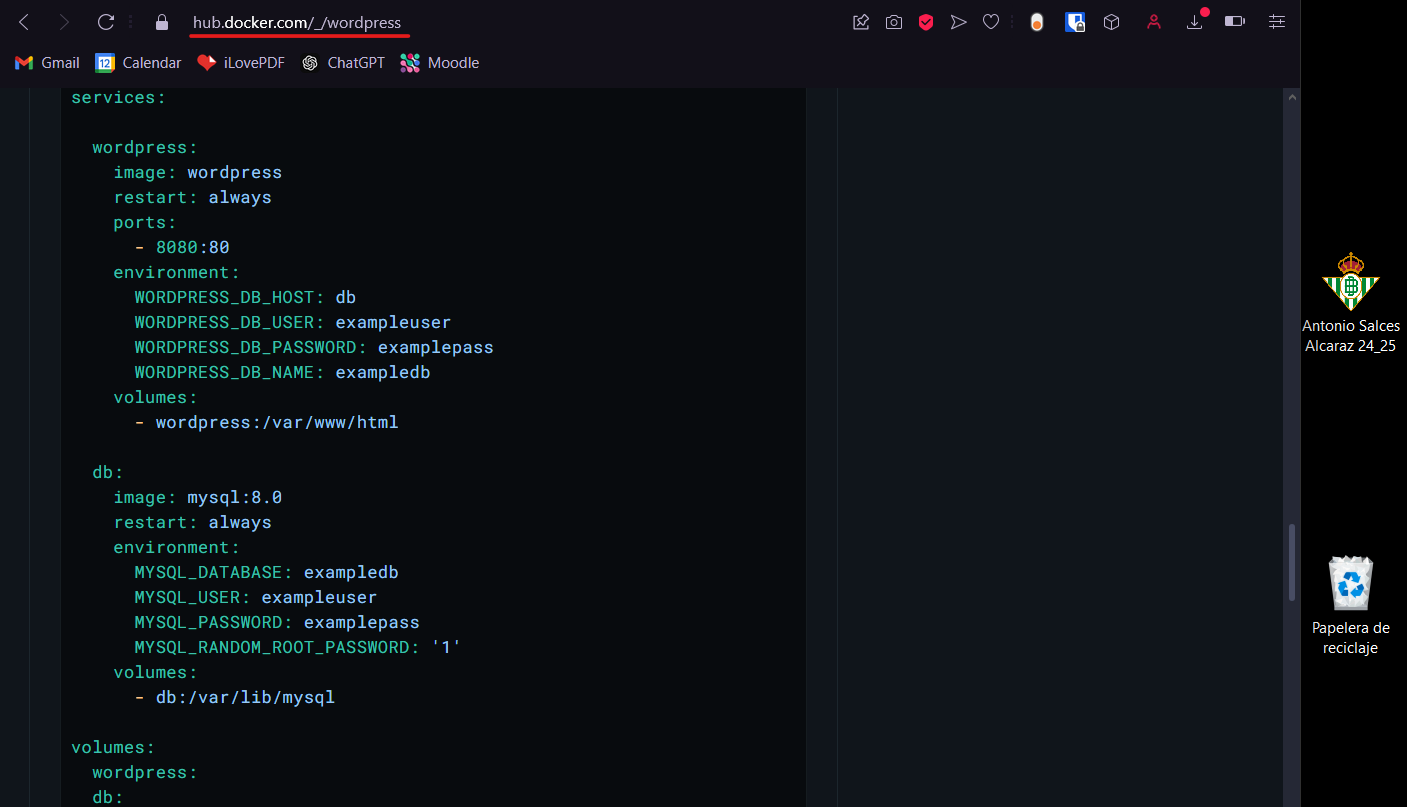
Ahora instalamos “docker compose v2” utilizando “***apt install <paquete>***”.

.

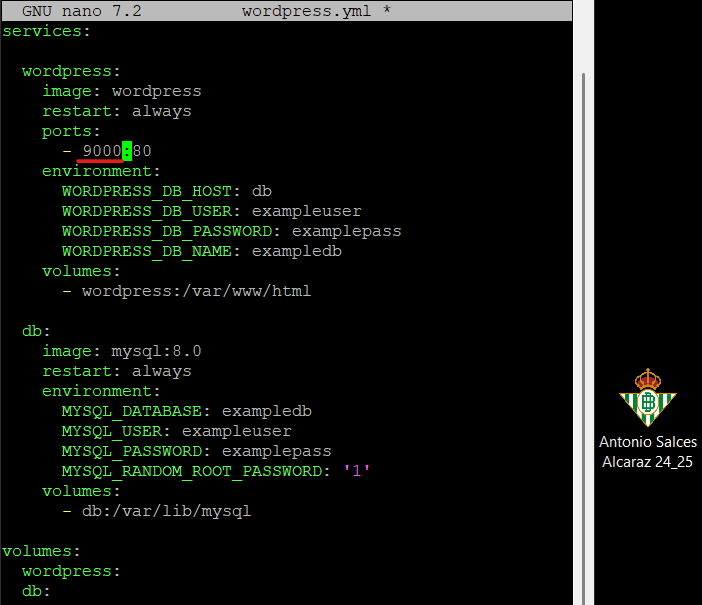
Para desplegar el archivo .yml crearemos una carpeta con “***mkdir <nombre***”. También crearemos y editaremos el archivo con “***nano <nombre>***”. En este paso se puede ver como hemos entrado desde “PuTTy”, en las siguientes imágenes se explicará el por qué.



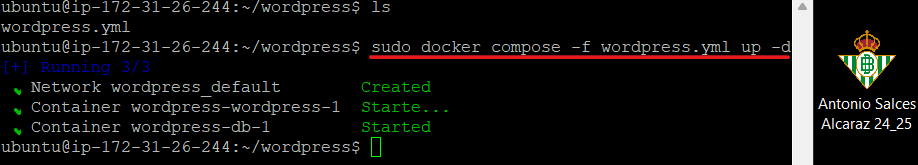
Podemos encontrar en la entrada oficial de “[Wordpress](https://hub.docker.com/_/wordpress)” en la página “[Docker Hub](https://hub.docker.com/)” un ejemplo de archivo .yml. Lo copiaremos en el portapapeles.



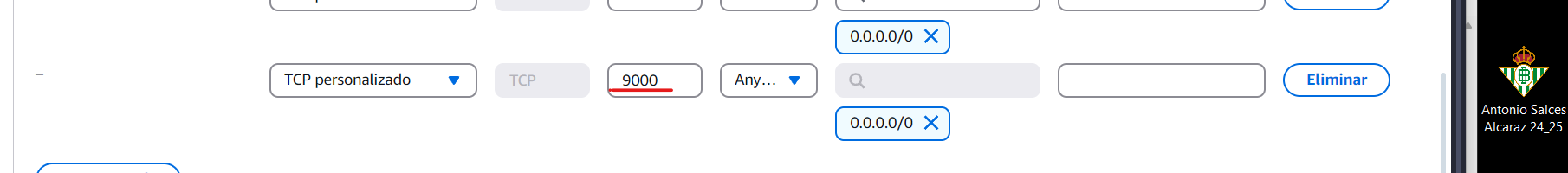
Desde “PuTTy”, ahora podremos pegar más cómodamente, por eso hemos accedido al sistema con él. Copiaremos todo el contenido en el archivo .yml creado, y le cambiaremos el puerto al indicado en el enunciado (9000).



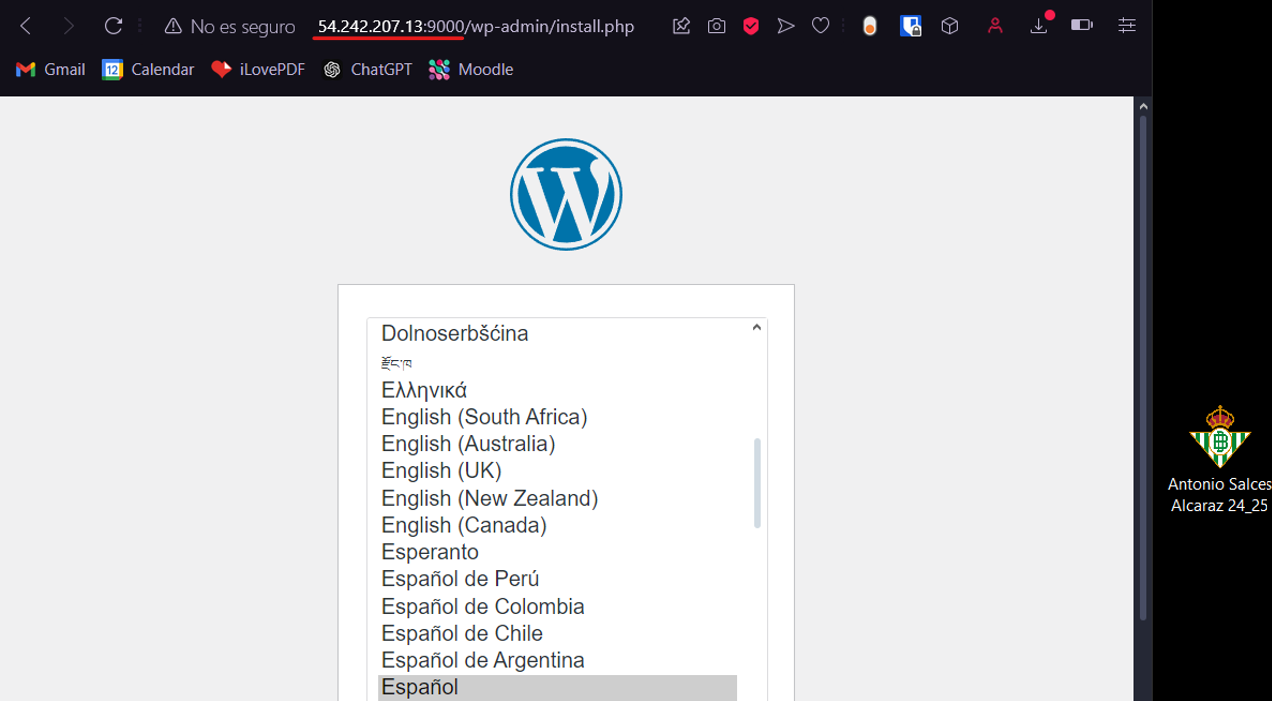
Ahora solo debemos de poner en marcha el archivo con “***docker compose [-f <archivo>] up [-d]***”.



Para poder conectarnos a la instancia por Internet al puerto de “Wordpress”, tenemos que abrir el puerto asignado como se ha visto en el punto 1.



Como podemos ver en la siguiente imagen, si buscamos en el buscado la IP seguida del puerto (***<IP>:<puerto>***), podremos ver como entramos en el proceso de instalación de “Wordpress”.



Una vez lo hayamos instalado, podemos empezar a crear páginas que podremos ver en Internet con el enlace proporcionado por “Wordpress”.

