

INSTALACIÓN DE WORDPRESS EN AWS CON SSH



MARIO CAMACHO PRIETO
2º ASIR / CAJA MÁGICA
06 / 01 / 2026

ÍNDICE

1. Preparación de seguridad local: Generación de claves SSH y configuración de permisos.
2. Configuración de AWS Academy: Acceso al entorno de laboratorio y gestión de claves en la nube.
3. Seguridad en la red: Creación de reglas de firewall (Security Groups) para el tráfico web y SSH.
4. Despliegue del servidor: Lanzamiento y configuración de la instancia EC2 con Ubuntu Server.
5. Administración remota y actualización: Conexión SSH al servidor y puesta al día del sistema.
6. Instalación del stack LAMP: Configuración de Apache, MySQL y PHP.
7. Automatización con Scripting: Creación y ejecución de un script para la instalación desatendida de WordPress.
8. Configuración final del CMS: Finalización de la instalación de WordPress a través del navegador.
9. Exposición pública con túnel seguro: Implementación de ngrok y ajuste final de URLs en la base de datos.

Utilicé estos comandos para crear el directorio donde guardaría mis llaves y otorgarle permisos exclusivos a mi usuario (lectura, escritura y ejecución), evitando que otros usuarios del sistema puedan acceder a él.

```
alumno16@A6Alumno16:~$ mkdir -p ~/.ssh
alumno16@A6Alumno16:~$ chmod 700 ~/.ssh
alumno16@A6Alumno16:~$ |
```

Con este comando generé mi par de llaves criptográficas (pública y privada). En la captura se observa el "randomart", que es la representación gráfica que confirma que la llave se generó correctamente

```
alumno16@A6Alumno16:~$ ssh-keygen -t ed25519 -f ~/.ssh/wordpress-key -C "mario@aws"
Generating public/private ed25519 key pair.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/alumno16/.ssh/wordpress-key
Your public key has been saved in /home/alumno16/.ssh/wordpress-key.pub
The key fingerprint is:
SHA256:ZiuOEVp8ZKUCVn8GVThVpfad9hF7fYz4qvWczkhRmXY mario@aws
The key's randomart image is:
+--[ED25519 256]--+
|  o.. ..o+o....|
|  . o . +o    + |
|    + = o.  B.E |
|  . = o    = *= |
|  o  S    o . =B |
|  o o .    o. = |
|  . . .    o . . |
|  + .    o * . |
|  . .    ..o.* |
+-----[SHA256]-----+
alumno16@A6Alumno16:~$ |
```

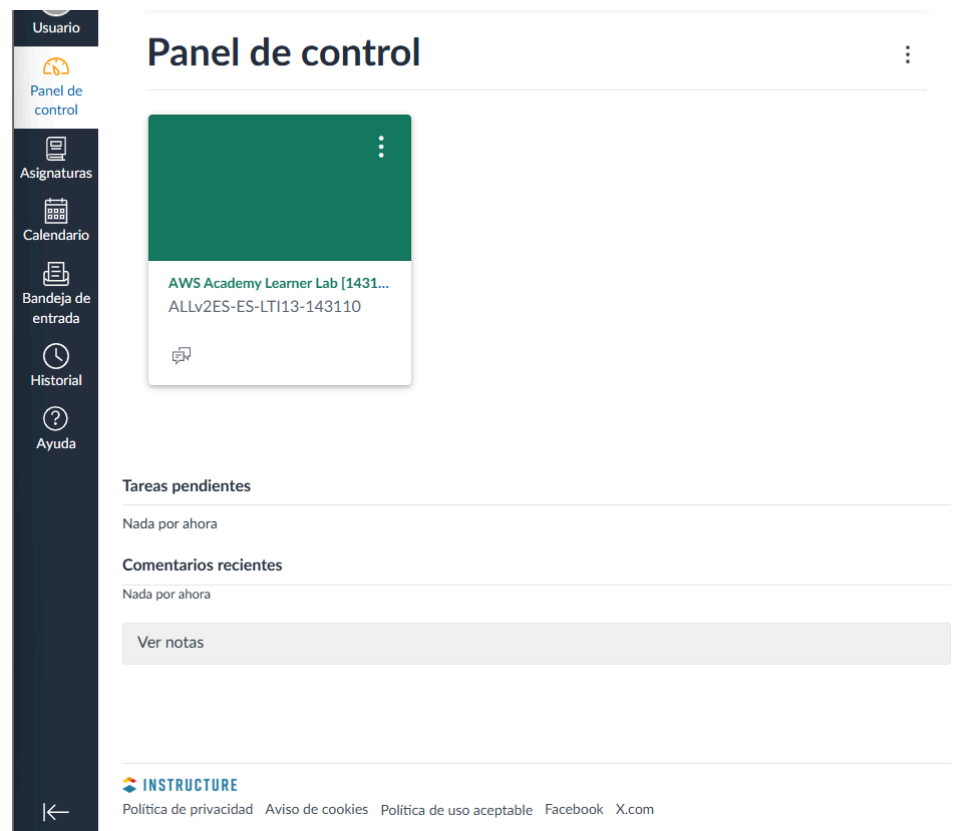
Verifiqué que los archivos .pub (público) y la llave privada existieran. Apliqué el permiso 400 a la llave privada para que sea solo de lectura para mí, algo que SSH exige por seguridad para funcionar.

```
alumno16@A6Alumno16:~$ ls -la ~/.ssh/wordpress-key*
-rw----- 1 alumno16 alumno16 399 Feb  2 09:49 /home/alumno16/.ssh/wordpress-key
-rw-r--r-- 1 alumno16 alumno16  91 Feb  2 09:49 /home/alumno16/.ssh/wordpress-key.pub
alumno16@A6Alumno16:~$ |
```

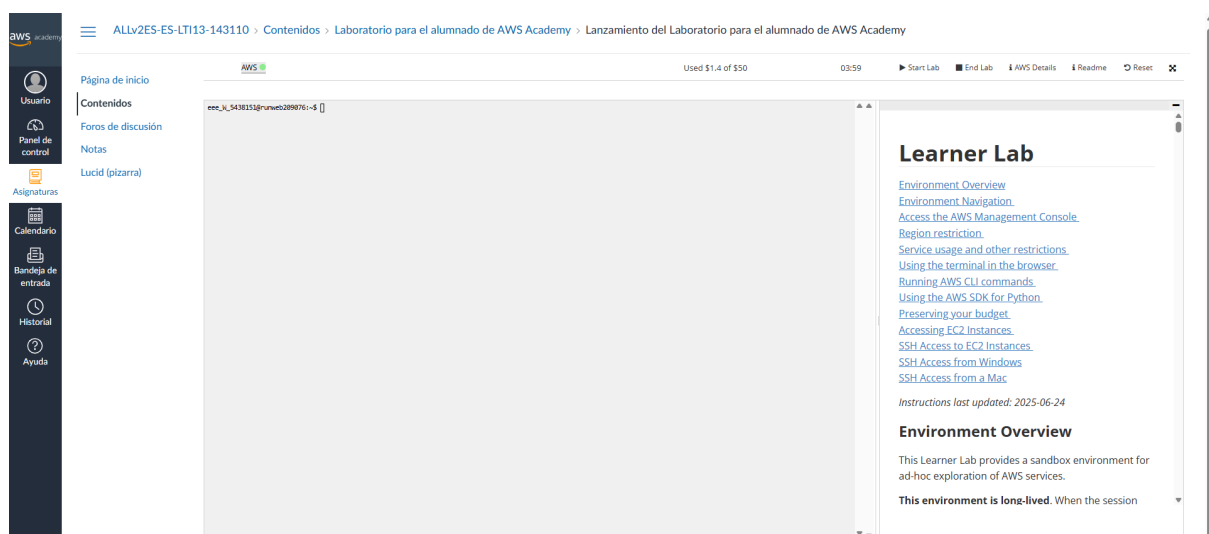
```
alumno16@A6Alumno16:~$ chmod 400 ~/.ssh/wordpress-key
alumno16@A6Alumno16:~$ |
```

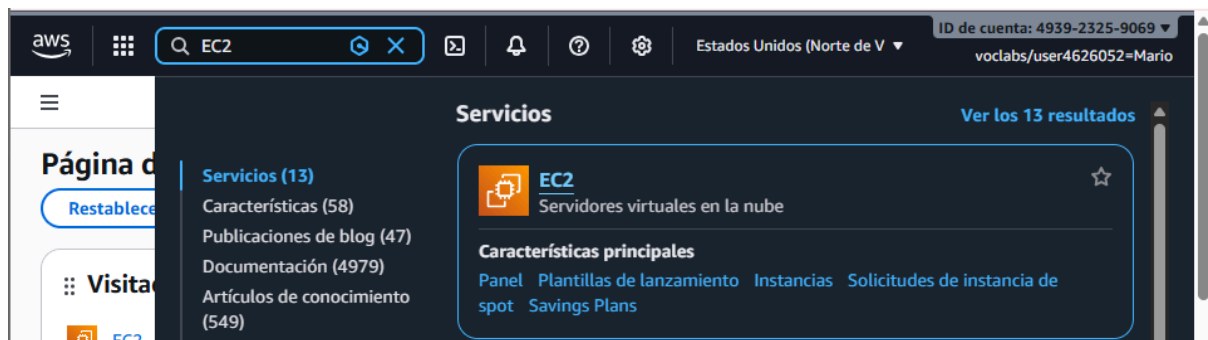
```
alumno16@A6Alumno16:~$ ls -la ~/.ssh/wordpress-key
-r----- 1 alumno16 alumno16 399 Feb  2 09:49 /home/alumno16/.ssh/wordpress-key
alumno16@A6Alumno16:~$ |
```

En la captura de pantalla muestro el estado "Ready" en verde, lo que indica que el entorno de AWS está activo y listo para desplegar máquinas.

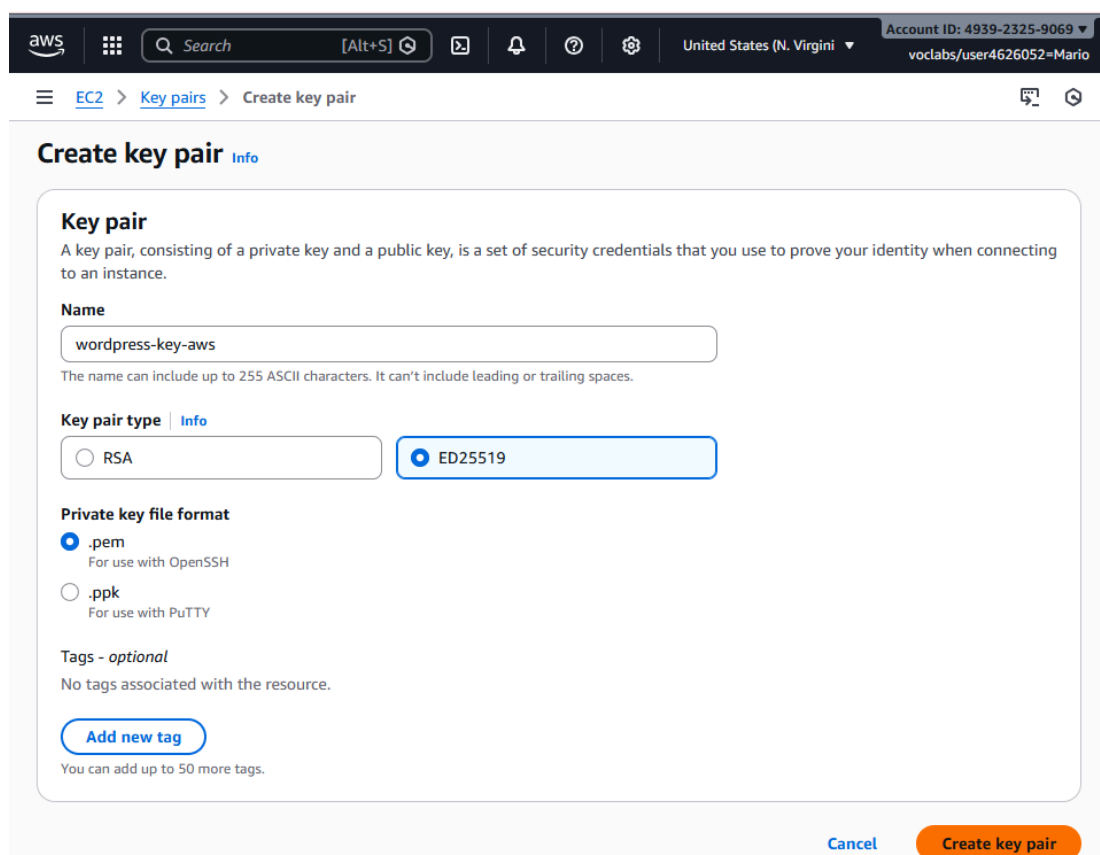


En la consola de AWS, creo la clave wordpress-key-aws en formato .pem. Primero entro en el botón de AWS para poder buscar la sección de EC2 en el buscador y poder entrar en la sección de Create Key Pair para crear la clave.





Creé una llave llamada wordpress-key-aws en formato .pem directamente en AWS para poder asociarla a mi futura instancia.



wordpress-key-aws.pem

Tras descargar la llave de AWS, la moví a mi carpeta .ssh y volví a asegurar su privacidad con chmod 400 para poder usarla en la conexión remota.

```
alumno16@A6Alumno16:~$ cp /mnt/c/Users/Alumno.DESKTOP-DI5KTUG/Downloads/wordpress-key-aws.pem ~/.ssh/
alumno16@A6Alumno16:~$ chmod 400 ~/.ssh/wordpress-key-aws.pem
alumno16@A6Alumno16:~$
```

He creado el grupo wordpress-aws-sg para establecer las reglas de tráfico necesarias que hagan mi servidor funcional y accesible. Configuré el puerto 22 (SSH) con el fin de poder administrar el servidor de forma remota mediante la terminal. También habilité el puerto 80 (HTTP) para que cualquier usuario en la red pueda entrar y visualizar mi página web. Adicionalmente, incluí el puerto 443 (HTTPS) para garantizar que el sitio pueda soportar conexiones cifradas y seguras en el futuro. Finalmente, realicé estos pasos en la consola de AWS para asegurar que mi instancia tenga un cortafuegos correctamente definido para WordPress.

EC2 > Security Groups > Create security group

Create security group [Info](#)

A security group acts as a virtual firewall for your instance to control inbound and outbound traffic. To create a new security group, complete the fields below.

Basic details

Security group name [Info](#)

Name cannot be edited after creation.

Description [Info](#)

VPC [Info](#)

Outbound rule 1 [Delete](#)

Type Info	Protocol Info	Port range Info
SSH	TCP	22
Destination type Info	Destination Info	Description - optional Info
Custom	<input type="text" value="0.0.0.0/0"/>	

Outbound rule 2 [Delete](#)

Type Info	Protocol Info	Port range Info
Custom TCP	TCP	80
Destination type Info	Destination Info	Description - optional Info
Anywhere-IPv4	<input type="text" value="0.0.0.0/0"/>	

Outbound rule 3 [Delete](#)

Type Info	Protocol Info	Port range Info
Custom TCP	TCP	443
Destination type Info	Destination Info	Description - optional Info
Anywhere-IPv4	<input type="text" value="0.0.0.0/0"/>	

[Add rule](#)

✓ Security group (sg-046754b6ad337a1f2 | wordpress-aws-sg) was created successfully
▶ Details

He configurado y lanzado oficialmente mi instancia "wordpress-server" utilizando una imagen de Ubuntu 24.04 LTS para asegurar un sistema operativo robusto y actualizado. Elegí el tipo de instancia t3.micro y asigné 15 GiB de almacenamiento gp3 para equilibrar el rendimiento con los beneficios de la capa gratuita de AWS. Para garantizar un acceso seguro, vinculé el par de llaves "wordpress-key-aws" y habilitéé la asignación automática de una IP pública. Finalmente, integré la instancia en mi red virtual (VPC) y definí un nuevo grupo de seguridad para controlar el tráfico entrante. Con estos pasos completados, mi servidor ya está inicializado y listo para comenzar la instalación de WordPress.

Launch an instance [Info](#)

Amazon EC2 allows you to create virtual machines, or instances, that run on the AWS Cloud. Quickly get started by following the simple steps below.

Name and tags [Info](#)

Name

wordpress-server

[Add additional tags](#)

▼ Application and OS Images (Amazon Machine Image) [Info](#)

An AMI contains the operating system, application server, and applications for your instance. If you don't see a suitable AMI below, use the search field or choose **Browse more AMIs**.

Recents

Quick Start

Amazon Linux

aws

macOS

Mac

Ubuntu

ubuntu

Windows

Microsoft

Red Hat

Red Hat

SUSE Linux

SUSE

Debian

debian

Browse more AMIs

Including AMIs from AWS, Marketplace and the Community

Amazon Machine Image (AMI)

Ubuntu Server 24.04 LTS (HVM), SSD Volume Type
ami-0b6c6ebd2801a5cb (64-bit (x86)) / ami-0071c8c431eea0edb (64-bit (Arm))
Virtualization: hvm ENA enabled: true Root device type: ebs

Free tier eligible

▼ Instance type [Info](#) | [Get advice](#)

Instance type

t3.micro
Family: t3 2 vCPU 1 GiB Memory Current generation: true
On-Demand Ubuntu Pro base pricing: 0.0139 USD per Hour
On-Demand SUSE base pricing: 0.0104 USD per Hour
On-Demand Linux base pricing: 0.0104 USD per Hour
On-Demand RHEL base pricing: 0.0392 USD per Hour
On-Demand Windows base pricing: 0.0196 USD per Hour

Free tier eligible

All generations

[Compare instance types](#)

Additional costs apply for AMIs with pre-installed software

▼ Key pair (login) [Info](#)

You can use a key pair to securely connect to your instance. Ensure that you have access to the selected key pair before you launch the instance.

Key pair name - *required*

wordpress-key-aws

[Create new key pair](#)

▼ Network settings [Info](#)

VPC - required [Info](#)

vpc-03428f9b8aac6e5af
172.31.0.0/16

(default) ▼



Subnet [Info](#)

No preference ▼



[Create new subnet](#)

Availability Zone [Info](#)

No preference ▼



[Enable additional zones](#)

Auto-assign public IP [Info](#)

Enable ▼

Firewall (security groups) [Info](#)

A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

☒ Create security group

☐ Select existing security group

Security group name - required

wordpress-aws-sg-4

This security group will be added to all network interfaces. The name can't be edited after the security group is created. Max length is 255 characters.
Valid characters: a-z, A-Z, 0-9, spaces, and ._-:/()#,@[]+=&:{}!\$*

▼ Configure storage [Info](#)

1x

15

GiB

gp3 ▼

Root volume, 3000 IOPS, Not encrypted

[Add new volume](#)

✓ Success

Successfully initiated launch of instance ([i-05341d29d2caa27d9](#))

<input checked="" type="checkbox"/>	Name ✎	Instance ID	Instance state	Instance type
<input checked="" type="checkbox"/>	wordpress-ser...	i-0430da0e7620cad63	Running 🔍 🔍	t3.micro

i-0430da0e7620cad63 (wordpress-server) [⚙️](#) [▼](#)

[<](#) [Details](#) [Status and alarms](#) [Monitoring](#) [Security](#) [Network](#) [>](#)

▼ Instance summary [Info](#)

Instance ID
[🔍](#) i-0430da0e7620cad63

Private IPv4 addresses
[🔍](#) 172.31.26.153

Instance state
[🟢](#) Running

Public IPv4 address
[🔍](#) 54.234.160.213 | [open address](#)

IPv6 address
-

Public DNS
[🔍](#) ec2-54-234-160-213.compute-1.amazonaws.com | [open address](#)

8

Una vez lanzada la instancia con éxito, utilicé el protocolo SSH para establecer una conexión segura desde mi terminal local. Empleé mi archivo de identidad .pem para autenticarme como el usuario ubuntu en la IP pública asignada. Este es el primer paso crítico para cualquier tarea de administración en la nube.

```
alumno16@A6Alumno16:~$ ssh -i ~/.ssh/wordpress-key-aws.pem ubuntu@3.228.20.44
The authenticity of host '3.228.20.44 (3.228.20.44)' can't be established.
ED25519 key fingerprint is SHA256:16ZiK++d3fo9cmpp8SAfM8a8MLoeEiP5/j1LkmKA.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? Yes
Warning: Permanently added '3.228.20.44' (ED25519) to the list of known hosts.
Welcome to Ubuntu 24.04.3 LTS (GNU/Linux 6.14.0-1018-aws x86_64)

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

System information as of Mon Feb  2 10:12:29 UTC 2026

System load:  0.0           Temperature:   -273.1 C
Usage of /:   12.9% of 13.49GB Processes:    119
Memory usage: 24%          Users logged in: 0
Swap usage:   0%           IPv4 address for ens5: 172.31.70.251

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

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

The list of available updates is more than a week old.
To check for new updates run: sudo apt update

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@ip-172-31-70-251:~$ |
```

Ejecuté `sudo apt update` y `upgrade` para parchear cualquier vulnerabilidad del sistema operativo y asegurar que todos los repositorios estuvieran al día.

```
ubuntu@ip-172-31-70-251:~$ sudo apt update
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease [126 kB]
Get:4 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Packages [15.0 MB]
Get:5 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Get:6 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe Translation-en [5982 kB]
Get:7 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Components [3871 kB]
Get:8 http://security.ubuntu.com/ubuntu noble-security/main amd64 Packages [1410 kB]
Get:9 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Component Metadata [301 kB]
69 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.

ubuntu@ip-172-31-70-251:~$ sudo apt upgrade -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Calculating upgrade... Done
The following packages will be upgraded:
  bsdxtrutils bsdxtrutils dirmngr eject fdisk fwupd gir1.2-glib-2.0 gnupg gnupg-l10n gnupg-utils gpg
  gpg-agent gpg-wks-client gpgconf gpgsm gpgv keyboxd klibc-utils kpartx libblkid1 libdrm-common
  libdrm2 libfdisk1 libfwupd2 libglib2.0-0t64 libglib2.0-bin libglib2.0-data libklibc libmbim-glib4
  libmbim-proxy libmbim-utils libmount1 libnss-systemd libnuma1 libpam-systemd libpng16-16t64
  libpython3.12-minimal libpython3.12-stdlib libpython3.12t64 libsmartcols1 libsodium23 libssl3t64
  libsystemd-shared libsystemd0 libtasn1-6 libudev1 libuuid1 libxml2 libxslt1.1 linux-tools-common
  mount multipath-tools numactl openssl python3-distupgrade python3-pyasn1 python3-urllib3
  python3.12 python3.12-minimal screen snapd systemd systemd-dev systemd-resolved systemd-sysv
  ubuntu-release-upgrader-core udev util-linux uuid-runtime
69 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
```

Instalé de forma conjunta el servidor web Apache, el motor de base de datos MySQL y el lenguaje PHP. Estos tres componentes forman el entorno "LAMP", el estándar de la industria necesario para que WordPress pueda procesar código y almacenar contenido.

```
ubuntu@ip-172-31-70-251:~$ sudo apt install apache2 php php-mysql libapache2-mod-php php-curl php-gd
php-mbstring php-xml php-xmlrpc php-intl php-zip mysql-server -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  apache2-bin apache2-data apache2-utils fontconfig-config fonts-dejavu-core fonts-dejavu-mono
  libaom3 libapache2-mod-php8.3 libapr1t64 libaprutil1-dbd-sqlite3 libaprutil1-ldap libaprutil1t64
  libb2-0 libbrotli1 libfontconfig1 libgd3 libheif-plugin-aomdec libheif-plugin-aomenc
  libheif-plugin-libde265 libheif1 libjpeg-turbo8 libjpeg8 liblerc4 liblua5.4-0
  libsharpyuv0 libtiff6 libwebp7 libxpm4 php-common php8.3 php8.3-cli php8.3-common php8.3-curl
```

Finalmente, utilicé `systemctl start` y `systemctl enable` para Apache y MySQL. Esto es crucial porque asegura que, si el servidor se reinicia por cualquier motivo, ambos servicios arranquen solos sin que yo tenga que intervenir manualmente.

```
ubuntu@ip-172-31-70-251:~$ sudo systemctl start apache2
ubuntu@ip-172-31-70-251:~$ sudo systemctl start mysql
ubuntu@ip-172-31-70-251:~$ sudo systemctl enable apache2
Synchronizing state of apache2.service with SysV service script with /usr/lib/systemd/systemd-sysv-instal
l.
Executing: /usr/lib/systemd/systemd-sysv-install enable apache2
ubuntu@ip-172-31-70-251:~$ sudo systemctl enable mysql
Synchronizing state of mysql.service with SysV service script with /usr/lib/systemd/systemd-sysv-instal
l.
Executing: /usr/lib/systemd/systemd-sysv-install enable mysql
ubuntu@ip-172-31-70-251:~$
```

Comprobé que ambos servicios estuvieran en estado "active (running)". Confirmé que Apache estaba listo para entregar páginas y que MySQL reportaba estar "operational", asegurando que la base del proyecto es totalmente estable.

```
ubuntu@ip-172-31-70-251:~$ sudo systemctl status apache2
● apache2.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/apache2.service; enabled; preset: enabled)
   Active: active (running) since Mon 2026-02-02 10:15:09 UTC; 18min ago
     Docs: https://httpd.apache.org/docs/2.4/
   Main PID: 16791 (apache2)
    Tasks: 6 (limit: 1008)
   Memory: 12.8M (peak: 14.1M)
      CPU: 114ms
   CGroup: /system.slice/apache2.service
           └─16791 /usr/sbin/apache2 -k start
             └─16796 /usr/sbin/apache2 -k start
               └─16797 /usr/sbin/apache2 -k start
                 └─16798 /usr/sbin/apache2 -k start
                   └─16799 /usr/sbin/apache2 -k start
                     └─16800 /usr/sbin/apache2 -k start

Feb 02 10:15:09 ip-172-31-70-251 systemd[1]: Starting apache2.service - The Apache HTTP Server...
Feb 02 10:15:09 ip-172-31-70-251 systemd[1]: Started apache2.service - The Apache HTTP Server.
ubuntu@ip-172-31-70-251:~$ sudo systemctl status mysql
● mysql.service - MySQL Community Server
   Loaded: loaded (/usr/lib/systemd/system/mysql.service; enabled; preset: enabled)
   Active: active (running) since Mon 2026-02-02 10:17:09 UTC; 16min ago
     Main PID: 17846 (mysqld)
    Status: "Server is operational"
    Tasks: 37 (limit: 1008)
   Memory: 351.7M (peak: 366.7M)
      CPU: 7.547s
   CGroup: /system.slice/mysql.service
           └─17846 /usr/sbin/mysqld

Feb 02 10:17:08 ip-172-31-70-251 systemd[1]: Starting mysql.service - MySQL Community Server...
Feb 02 10:17:09 ip-172-31-70-251 systemd[1]: Started mysql.service - MySQL Community Server.
ubuntu@ip-172-31-70-251:~$ |
```

Para finalizar, desarrollé un script de Bash (install-wordpress.sh) para automatizar la configuración crítica y la instalación del CMS:

```
ubuntu@ip-172-31-70-251:~$ sudo nano install-wordpress.sh
ubuntu@ip-172-31-70-251:~$ |
```

En esta primera sección, he definido las variables críticas para el entorno, generando contraseñas aleatorias y seguras mediante openssl para proteger el acceso a los datos. Posteriormente, he automatizado el robustecimiento de MySQL, eliminando usuarios anónimos y bases de datos de prueba para cerrar brechas de seguridad iniciales. Finalmente, el script crea de forma automática la base de datos wordpress y el usuario wpuser, otorgándole los privilegios necesarios para que el CMS pueda operar correctamente.

```
GNU nano 7.2                                install-wordpress.sh *
#!/bin/bash
set -e

echo "=== Iniciando instalación automatizada de WordPress ==="

# Variables
DB_NAME="wordpress"
DB_USER="wpuser"
DB_PASSWORD="$(openssl rand -base64 12)"
DB_ROOT_PASSWORD="$(openssl rand -base64 12)"
WP_HOME="http://localhost"
WP_SITEURL="http://localhost"

# Paso 1: Configurar MySQL
echo "Configurando MySQL..."
sudo mysql -e "ALTER USER 'root'@'localhost' IDENTIFIED BY '${DB_ROOT_PASSWORD}';"
sudo mysql -e "DELETE FROM mysql.user WHERE User='';"
sudo mysql -e "DELETE FROM mysql.user WHERE User='root' AND Host NOT IN ('localhost', '127.0.0.1', '::1');"
sudo mysql -e "DROP DATABASE IF EXISTS test;"
sudo mysql -e "DELETE FROM mysql.db WHERE Db='test' OR Db='test_%';"
sudo mysql -e "FLUSH PRIVILEGES;"

# Paso 2: Crear base de datos y usuario de WordPress
echo "Creando base de datos y usuario..."
sudo mysql -u root -p"${DB_ROOT_PASSWORD}" -e "CREATE DATABASE ${DB_NAME};"
sudo mysql -u root -p"${DB_ROOT_PASSWORD}" -e "CREATE USER '${DB_USER}'@'localhost' IDENTIFIED BY '${DB_PASSWORD}';"
sudo mysql -u root -p"${DB_ROOT_PASSWORD}" -e "GRANT ALL PRIVILEGES ON ${DB_NAME}.* TO '${DB_USER}'@'localhost';"
sudo mysql -u root -p"${DB_ROOT_PASSWORD}" -e "FLUSH PRIVILEGES;"
```

Este bloque se encarga de la gestión del contenido y la integración con el servidor web. Primero, he programado la descarga del paquete oficial de WordPress y su despliegue en el directorio /var/www/html/. Acto seguido, utilizo comandos de edición de texto para configurar el archivo wp-config.php en tiempo real, inyectando las credenciales de la base de datos generadas anteriormente. Para concluir, ajusto los permisos de propiedad al usuario del servidor web y habilito el módulo rewrite de Apache, garantizando que el sitio sea capaz de gestionar enlaces permanentes y rutas personalizadas.

```
GNU nano 7.2                                install-wordpress.sh *
```

```
# Paso 3: Descargar WordPress
echo "Descargando WordPress..."
cd /tmp
wget https://wordpress.org/latest.tar.gz -q
tar -xzf latest.tar.gz

# Paso 4: Instalar WordPress
echo "Copiando archivos a /var/www/html..."
sudo rm -rf /var/www/html/*
sudo cp -r wordpress/* /var/www/html/

# Paso 5: Configurar wp-config.php
echo "Configurando wp-config.php..."
sudo cp /var/www/html/wp-config-sample.php /var/www/html/wp-config.php
sudo sed -i "s/database_name_here/${DB_NAME}/g" /var/www/html/wp-config.php
sudo sed -i "s/username_here/${DB_USER}/g" /var/www/html/wp-config.php
sudo sed -i "s/password_here/${DB_PASSWORD}/g" /var/www/html/wp-config.php

# Paso 6: Permisos
echo "Configurando permisos..."
sudo chown -R www-data:www-data /var/www/html/
sudo chmod -R 755 /var/www/html/

# Paso 7: Habilitar mod_rewrite en Apache
echo "Habilitando mod_rewrite..."
sudo a2enmod rewrite
sudo systemctl restart apache2
```

La parte final del script está diseñada para la gestión administrativa y la verificación del éxito del proceso. He configurado la creación de un archivo de texto llamado wordpress-credentials.txt donde se exportan y guardan de forma organizada todas las claves y usuarios generados durante la instalación. Esto me permite tener un respaldo inmediato de la configuración del sitio, incluyendo los accesos a la base de datos y la raíz de MySQL. El script termina notificando que la instalación ha concluido satisfactoriamente y proporciona la dirección IP para acceder al instalador visual desde el navegador.

```
# Paso 8: Guardar credenciales
echo "Guardando credenciales en archivo..."
cat > ~/wordpress-credentials.txt << EOF
=== CREDENCIALES DE WORDPRESS ===
Base de datos: ${DB_NAME}
Usuario BD: ${DB_USER}
Contraseña BD: ${DB_PASSWORD}
Usuario root MySQL: root
Contraseña root MySQL: ${DB_ROOT_PASSWORD}
Acceso local: http://localhost
Acceso remoto: (se configurará con ngrok)
EOF

echo "=== Instalación completada ==="
echo "Credenciales guardadas en ~/wordpress-credentials.txt"
echo "Accede a http://TU-IP-PUBLICA para finalizar la instalación de WordPress"
```

En esta primera captura utilizo el comando scp para copiar el script install-wordpress.sh desde mi máquina local a la instancia EC2 en AWS.

Especifico la clave privada (wordpress-key-aws.pem) para autenticarme de forma segura y el usuario ubuntu junto con la IP pública del servidor.

La advertencia sobre la autenticidad del host aparece porque es la primera vez que me conecto a esa máquina. Al aceptar con yes, el servidor queda registrado en el archivo known_hosts, evitando que vuelva a preguntarme en futuras conexiones.

Gracias a este paso, consigo transferir el script de automatización al servidor remoto de forma segura usando SSH.

```
alumno16@A6Alumno16:~$ sudo scp -i ~/wordpress-key-aws.pem install-wordpress.sh ubuntu@54.234.160.213:~/
[sudo] password for alumno16:
The authenticity of host '54.234.160.213 (54.234.160.213)' can't be established.
ED25519 key fingerprint is SHA256:65Nr5CyL/DhfYyUbbL76hx1Ba6YdOr+gZt+zVJ4IWNM.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '54.234.160.213' (ED25519) to the list of known hosts.
install-wordpress.sh                               100% 1185      8.0KB/s   00:00
alumno16@A6Alumno16:~$ |
```

Con esta acción le otorgo permisos de ejecución al script, lo que es imprescindible para poder lanzarlo directamente desde la terminal.

Sin este permiso, el sistema no permitiría ejecutar el archivo como un programa.

```
alumno16@A6Alumno16:~$ sudo chmod +x ~/install-wordpress.sh
alumno16@A6Alumno16:~$ |
```

Ejecuto el script `install-wordpress.sh` con el comando `./install-wordpress.sh`, iniciando así la instalación automática de WordPress. El script se encarga de configurar MySQL, crear la base de datos y el usuario necesarios, descargar y desplegar WordPress en `/var/www/html` y ajustar los permisos de los archivos para su correcto funcionamiento. También habilita el módulo `rewrite` de Apache para permitir el uso de enlaces permanentes y guarda las credenciales generadas en un archivo de texto. Durante la ejecución aparecen advertencias de MySQL por el uso de contraseñas en la línea de comandos, algo normal en este tipo de scripts. El mensaje final confirma que la instalación se ha completado correctamente y que el servidor está listo para acceder a WordPress desde el navegador.

```
ubuntu@ip-172-31-26-153:~$ ./install-wordpress.sh
=== Iniciando instalación automatizada de WordPress ===
Configurando MySQL...
Creando base de datos y usuario...
mysql: [Warning] Using a password on the command line interface can be insecure.
mysql: [Warning] Using a password on the command line interface can be insecure.
mysql: [Warning] Using a password on the command line interface can be insecure.
mysql: [Warning] Using a password on the command line interface can be insecure.
Descargando WordPress...
Copiando archivos a /var/www/html...
Configurando wp-config.php...
Configurando permisos...
Habilitando mod_rewrite...
Enabling module rewrite.
To activate the new configuration, you need to run:
    systemctl restart apache2
Guardando credenciales en archivo...
=== Instalación completada ===
Credenciales guardadas en ~/wordpress-credentials.txt
Accede a http://TU-IP-PUBLICA para finalizar la instalación de WordPress
```

ejecuté el comando `cat ~/wordpress-credentials.txt` para visualizar el resumen de mi configuración. En pantalla puedo confirmar los datos clave de mi sitio, como el nombre de la base de datos wordpress, el usuario `wpuser` y sus respectivas contraseñas generadas. Además, verifiqué que el acceso local está listo en `http://localhost` y que la parte remota queda pendiente de configurar mediante `ngrok`.

```
ubuntu@ip-172-31-26-153:~$ cat ~/wordpress-credentials.txt
=== CREDENCIALES DE WORDPRESS ===
Base de datos: wordpress
Usuario BD: wpuser
Contraseña BD: QV+ZL4YRU2EXckfE
Usuario root MySQL: root
Contraseña root MySQL: x2/gGQfAkTUuonHp
Acceso local: http://localhost
Acceso remoto: (se configurará con ngrok)
```


Para asegurar que el servidor esté plenamente operativo, utilicé el comando `systemctl status` para verificar el estado de los servicios principales. Comprobé que tanto Apache2 como MySQL aparecen en color verde como "active (running)", lo que confirma que el servidor web y el motor de base de datos se están ejecutando correctamente y sin errores.

```
ubuntu@ip-172-31-26-153:~$ sudo systemctl status apache2
● apache2.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/apache2.service; enabled; preset: enabled)
   Active: active (running) since Tue 2026-02-03 09:28:16 UTC; 1min 22s ago
     Docs: https://httpd.apache.org/docs/2.4/
   Process: 18229 ExecStart=/usr/sbin/apachectl start (code=exited, status=0/SUCCESS)
   Main PID: 18232 (apache2)
     Tasks: 6 (limit: 1008)
    Memory: 13.7M (peak: 14.0M)
       CPU: 64ms
    CGroup: /system.slice/apache2.service
           └─18232 /usr/sbin/apache2 -k start
             └─18235 /usr/sbin/apache2 -k start
               └─18236 /usr/sbin/apache2 -k start
                 └─18237 /usr/sbin/apache2 -k start
                   └─18238 /usr/sbin/apache2 -k start
                     └─18239 /usr/sbin/apache2 -k start

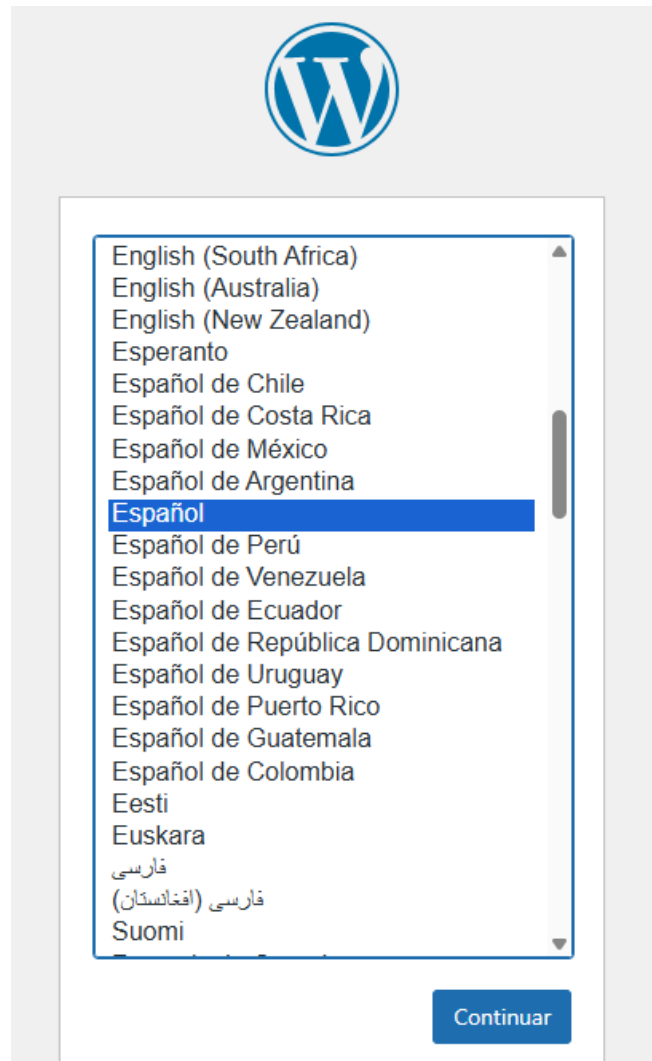
Feb 03 09:28:16 ip-172-31-26-153 systemd[1]: Starting apache2.service - The Apache HTTP Server...
Feb 03 09:28:16 ip-172-31-26-153 systemd[1]: Started apache2.service - The Apache HTTP Server.
```

```
ubuntu@ip-172-31-26-153:~$ sudo systemctl status mysql
● mysql.service - MySQL Community Server
   Loaded: loaded (/usr/lib/systemd/system/mysql.service; enabled; preset: enabled)
   Active: active (running) since Tue 2026-02-03 09:27:44 UTC; 2min 9s ago
   Main PID: 15667 (mysqld)
     Status: "Server is operational"
     Tasks: 38 (limit: 1008)
    Memory: 356.5M (peak: 383.2M)
       CPU: 1.883s
    CGroup: /system.slice/mysql.service
           └─15667 /usr/sbin/mysqld

Feb 03 09:27:43 ip-172-31-26-153 systemd[1]: Starting mysql.service - MySQL Community Server...
Feb 03 09:27:44 ip-172-31-26-153 systemd[1]: Started mysql.service - MySQL Community Server.
```


Una vez que tuve el servidor listo, procedí con la instalación visual de WordPress. Estos son los pasos que completé:

Seleccióne el idioma elegi el Español como idioma principal para mi sitio, asegurándome de que toda la interfaz de administración sea fácil de gestionar.



En este paso definí la identidad de mi proyecto. Le puse como título "PracticaAWS" y creé mi cuenta de administrador con el usuario admin. Para el correo electrónico de recuperación, utilicé mi dirección personal y, aunque la contraseña era sencilla, marqué la casilla para confirmar el uso de una contraseña débil y poder avanzar rápidamente en esta prueba técnica.

Hola

¡Este es el famoso proceso de instalación de WordPress en cinco minutos! Simplemente completa la información siguiente y estarás a punto de usar la más enriquecedora y potente plataforma de publicación personal del mundo.

Información necesaria

Por favor, proporciona la siguiente información. No te preocupes, siempre podrás cambiar estos ajustes más tarde.

Título del sitio

PracticaAWS

Nombre de usuario

admin

Los nombres de usuario pueden tener únicamente caracteres alfanuméricos, espacios, guiones bajos, guiones medios, puntos y el símbolo @.

Contraseña

•••••

Muy débil

Mostrar

Importante: Necesitas esta contraseña para acceder. Por favor, guárdala en un lugar seguro.

Confirma la contraseña

☒ Confirma el uso de una contraseña débil.

Tu correo electrónico

marioclaserebeca@gmail.com

Comprueba bien tu dirección de correo electrónico antes de continuar.

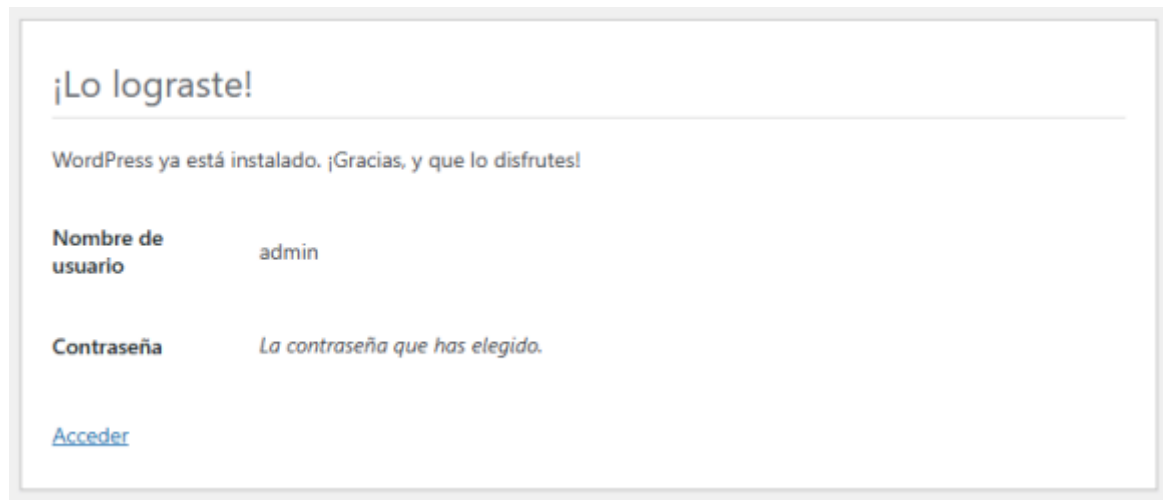
Visibilidad en los motores de búsqueda

☐ Pedir a los motores de búsqueda que no indexen este sitio

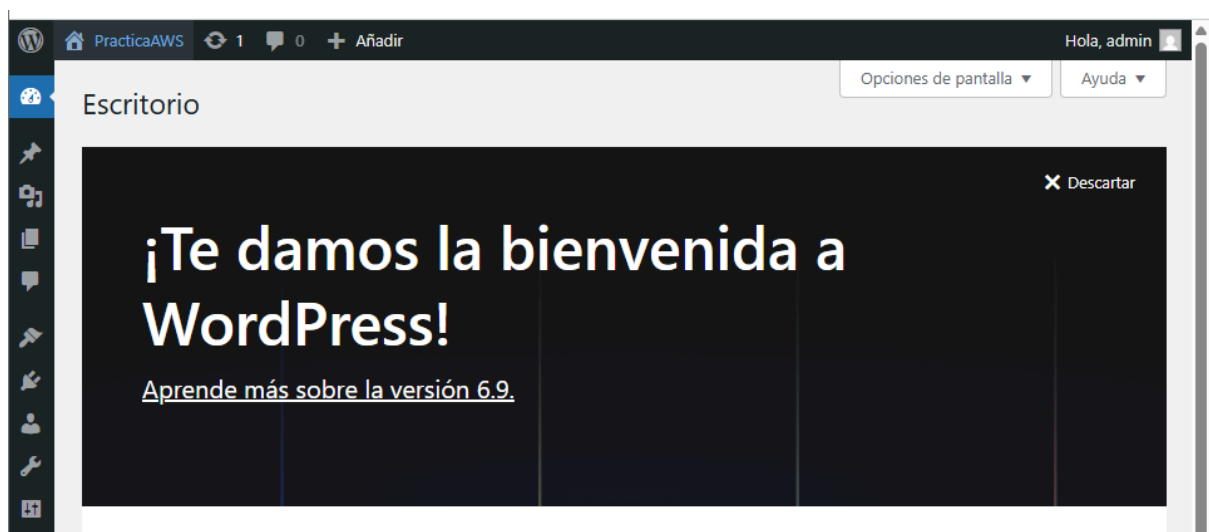
Depende de los motores de búsqueda atender esta petición o no.

Instalar WordPress

Tras pulsar el botón de instalar, recibí el mensaje de éxito: "¡Lo lograste!". Aquí confirmé que WordPress ya estaba instalado correctamente en mi instancia de AWS y listo para que yo pudiera acceder con mis credenciales.



Finalmente, inicié sesión y entré al Escritorio (Dashboard) de WordPress. Me recibió el mensaje de bienvenida a la versión 6.9, confirmando que tengo control total sobre el panel de administración y que el servidor Apache está sirviendo la página web sin problemas.



Primero, me situé en mi directorio personal con `cd ~` y descargué el paquete oficial de Ngrok para Linux (64 bits) usando `wget`.

Una vez descargado el archivo `.tgz`, lo descomprimí con `tar -xvzf ngrok-v3-stable-linux-amd64.tgz`, lo cual generó el archivo ejecutable llamado `ngrok`. Para poder ejecutarlo desde cualquier parte del sistema, moví el binario a la carpeta de ejecutables locales con `sudo mv ngrok /usr/local/bin/`.

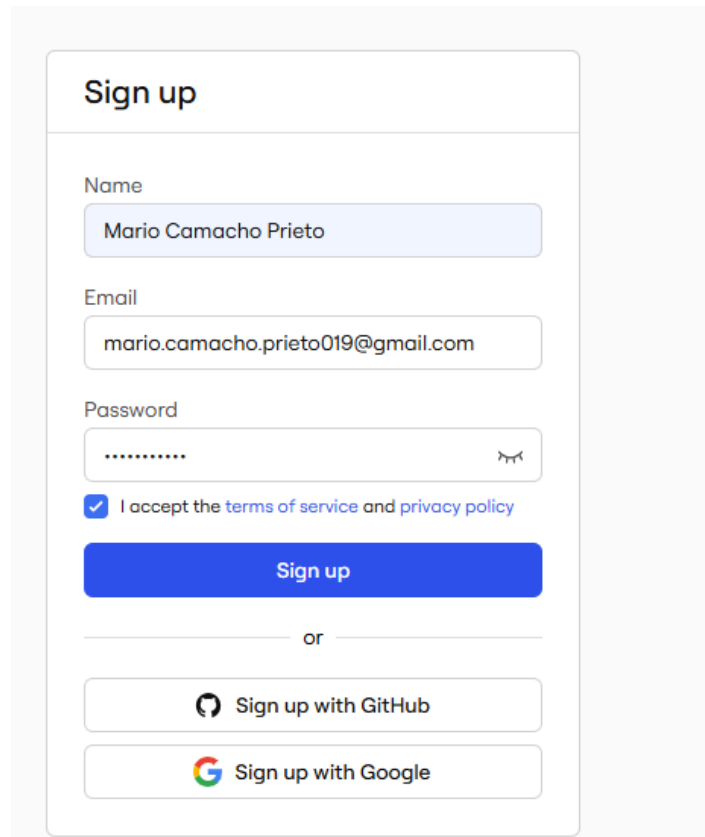
```
ubuntu@ip-172-31-26-153:~$ cd ~
ubuntu@ip-172-31-26-153:~$ wget https://bin.equinox.io/c/nyjmj1MqvV4c/ngrok-v3-stable-linux-amd64.tgz
--2026-02-03 09:33:38-- https://bin.equinox.io/c/nyjmj1MqvV4c/ngrok-v3-stable-linux-amd64.tgz
Resolving bin.equinox.io (bin.equinox.io)... 75.2.60.68, 99.83.220.108, 35.71.179.82, ...
Connecting to bin.equinox.io (bin.equinox.io)|75.2.60.68|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 11382588 (11M) [application/octet-stream]
Saving to: 'ngrok-v3-stable-linux-amd64.tgz'

ngrok-v3-stable-linux-amd 100%[=====>] 10.85M --.-KB/s in 0.05s

2026-02-03 09:33:39 (227 MB/s) - 'ngrok-v3-stable-linux-amd64.tgz' saved [11382588/11382588]

ubuntu@ip-172-31-26-153:~$ tar -xvzf ngrok-v3-stable-linux-amd64.tgz
ngrok
ubuntu@ip-172-31-26-153:~$ sudo mv ngrok /usr/local/bin/
```

Para que Ngrok funcione, tuve que registrarme en su web oficial (ngrok.com) creando una cuenta con mi nombre, correo y contraseña. Una vez dentro del panel de control de Ngrok, busqué la sección "Your Authtoken" para copiar mi clave personal secreta, la cual es necesaria para autenticar mi servidor con su servicio.





The image shows the Ngrok sign-up form. It has a title "Sign up" at the top. Below it are three input fields: "Name" with the value "Mario Camacho Prieto", "Email" with the value "mario.camacho.prieto019@gmail.com", and "Password" with a masked value ".....". There is a checkbox labeled "I accept the terms of service and privacy policy" which is checked. Below the checkbox is a blue "Sign up" button. Underneath the button is a horizontal line with the word "or" in the center. Below the line are two buttons: "Sign up with GitHub" and "Sign up with Google".

ngrok

Your Authtoken

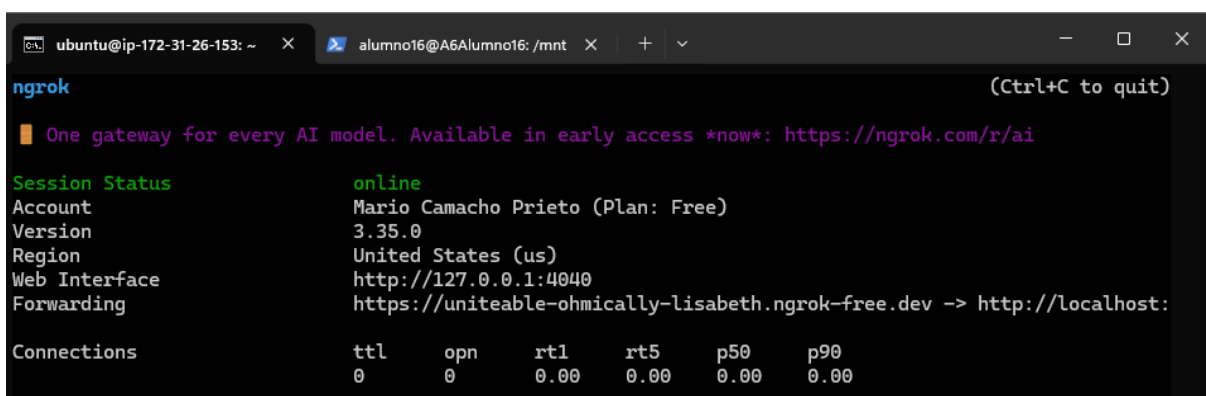
Use this personal Authtoken to authenticate ngrok agents, SDKs, and the Kubernetes Operator for your own projects. Keep it secret, like a password.

.....   Copy

Con el token copiado, regresé a mi terminal en el servidor de AWS y ejecuté el comando para guardar la configuración: `ngrok config add-authtoken`. El sistema me confirmó que el token se guardó correctamente en el archivo de configuración `.yaml`.

```
ubuntu@ip-172-31-26-153:~$ ngrok config add-authtoken 399c0dezK0o07ux68vQVr9bhW0W_2EFM1WfmMTGi7X1wD6J7q
Authtoken saved to configuration file: /home/ubuntu/.config/ngrok/ngrok.yaml
```

Finalmente, inicié el túnel para exponer mi servidor local (puerto 80) a Internet con el comando `ngrok http 80`. En la pantalla de estado de Ngrok, pude ver que el estado pasó a "online" y se generó una URL pública única (en este caso, una terminada en `.ngrok-free.dev`) que apunta directamente a mi `localhost:80`.



```
ngrok (Ctrl+C to quit)

One gateway for every AI model. Available in early access *now*: https://ngrok.com/r/ai

Session Status      online
Account             Mario Camacho Prieto (Plan: Free)
Version             3.35.0
Region              United States (us)
Web Interface        http://127.0.0.1:4040
Forwarding           https://uniteable-ohmically-lisabeth.ngrok-free.dev -> http://localhost:

Connections          ttl    opn    rt1    rt5    p50    p90
0                  0      0.00   0.00   0.00   0.00
```

He accedido con éxito a mi instancia de AWS mediante SSH, utilizando mi clave privada para identificarme como el usuario ubuntu. Al entrar, el sistema me presenta un resumen inmediato de la salud del servidor:

- Rendimiento y Carga: El servidor está operando con una carga de trabajo mínima (0.08), lo que significa que el procesador está prácticamente libre. Sin embargo, observo que la memoria RAM está siendo utilizada en un 74%, un nivel considerable para este tipo de instancia.
- Almacenamiento: El disco principal tiene una ocupación muy baja, del 20.8% sobre un total de 13.49 GB, por lo que tengo espacio de sobra para el despliegue.
- Anomalía Térmica: Un dato curioso que registra mi terminal es la temperatura de -273.1 °C. Es un error típico de los sensores en máquinas virtuales, ya que marca el límite físico del cero absoluto.
- Mantenimiento: El sistema me alerta de que tengo 3 actualizaciones de seguridad pendientes de instalar y que hay 3 paquetes adicionales que pueden actualizarse. Además, me recuerda que no he activado el soporte de mantenimiento extendido (ESM).

```
alumno16@A6Alumno16:/mnt/c/Users/Alumno.DESKTOP-DI5KTUG$ ssh -i ~/.ssh/wordpress-key-aws.pem ubuntu@54.234.160.213
Welcome to Ubuntu 24.04.3 LTS (GNU/Linux 6.14.0-1018-aws x86_64)

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

System information as of Tue Feb  3 09:46:10 UTC 2026

System load:  0.08           Temperature:   -273.1 C
Usage of /:   20.8% of 13.49GB Processes:      130
Memory usage: 74%           Users logged in: 1
Swap usage:   0%            IPv4 address for ens5: 172.31.26.153

* Ubuntu Pro delivers the most comprehensive open source security and
  compliance features.

  https://ubuntu.com/aws/pro

Expanded Security Maintenance for Applications is not enabled.

3 updates can be applied immediately.
3 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable

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

Last login: Tue Feb  3 09:22:33 2026 from 88.0.22.181
```

Me conecté de nuevo a la base de datos MySQL desde la terminal con el usuario que creamos antes: `mysql -u wpuser -p -D wordpress`.

```
ubuntu@ip-172-31-16-182:~$ mysql -u wpuser -p -D wordpress
Enter password:
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 50
Server version: 8.0.45-0ubuntu0.24.04.1 (Ubuntu)

Copyright (c) 2000, 2026, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
```

Para que el sitio funcione perfectamente con la nueva dirección pública, ejecuté dos comandos UPDATE dentro de MySQL para cambiar las opciones siteurl y home por la URL que me proporcionó Ngrok (por ejemplo: <https://uniteable-ohmically-lisabeth.ngrok-free.dev>)

```
mysql> UPDATE wp_options SET option_value='https://uniteable-ohmically-lisabeth.ngrok-free.dev' WHERE
option_name='siteurl';
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> UPDATE wp_options SET option_value='https://uniteable-ohmically-lisabeth.ngrok-free.dev' WHERE
option_name='home';
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```


Para terminar la práctica, abrí mi navegador web y escribí la dirección pública que me dio Ngrok.

Como puedes ver en la captura, el sitio carga perfectamente con todo el contenido, imágenes y estilos.

Probé a navegar por la "Página de ejemplo" y todo funciona de maravilla, lo que significa que el túnel está bien configurado y la base de datos responde correctamente a las peticiones externas.

