

## Configuración de un servidor Web en Ubuntu

### Instala el servidor Apache

```
sudo apt install apache2
```

Esto instalará el servidor web Apache y sus dependencias.

### Gestiona y verifica el estado de Apache

Iniciar Apache:

```
sudo systemctl start apache2
```

Verificar que el servicio está activo:

```
sudo systemctl status apache2
```

Configura para que se inicie en el arranque:

```
sudo systemctl enable apache2
```

### Permite el tráfico web (ajuste del firewall)

Si tienes UFW activado, ejecuta:

```
sudo ufw allow 'Apache Full'
```

Esto abrirá los puertos necesarios para HTTP y HTTPS.

### Accede desde un navegador:

*http://IP\_DE\_TU\_SERVIDOR*

Deberías ver la página de bienvenida de Apache.

### Instala soporte adicional (opcional)

Para PHP:

```
sudo apt install php libapache2-mod-php
```

Para bases de datos:

```
sudo apt install mysql-server
```

### Sube tus archivos web

Pon tus archivos HTML, PHP o recursos en /var/www/html

## Habilitar un puerto por ejemplo el 8000

### Permite también el puerto 8000

Puedes agregar la regla antes o después de habilitar UFW:

```
sudo ufw allow ssh # para impedir que se corte la sesión
```

```
sudo ufw allow 80/tcp
```

```
sudo ufw allow 443/tcp
```

```
sudo ufw allow 8000/tcp
```

Esto permitirá el tráfico TCP en el puerto 8000.

### Activa UFW

```
sudo ufw enable
```

Te mostrará una advertencia sobre posible desconexión SSH. Ya agregaste la regla SSH, así que puedes responder y (sí).

### Verifica el estado y las reglas

```
sudo ufw status
```

Deberías ver que UFW está activo y tanto SSH como el puerto 8000 están permitidos.

## Python

### Actualizar el sistema

```
sudo apt update && sudo apt upgrade -y
```

### Instalar y configurar Python (3.x)

Instala Python3, pip, venv y dependencias para compilar paquetes nativos

```
sudo apt install -y python3 python3-pip python3-venv python3-dev build-essential libssl-dev  
libffi-dev
```

### Verifica versiones

```
python3 --version
```

```
pip3 --version
```

### Crear y usar un entorno virtual (recomendado para cada proyecto):

crea carpeta de proyecto y venv

```
mkdir -p ~/projects/mi_proyecto  
cd ~/projects/mi_proyecto  
python3 -m venv venv
```

activa

```
source venv/bin/activate
```

dentro del venv, actualiza pip e instala lo que necesites:

```
pip install --upgrade pip  
pip install fastapi[standard]
```

### Gestión del servidor uvicorn

ejecutar el servidor uvicorn en segundo plano

```
nohup unicorn main:app --host 0.0.0.0 --port 8000 > server.log 2>&1 &
```

Ver el log del servidor uvicorn

```
cat server.log
```

### Parar el servicio en 2do plano

Primero necesitas ver el PID (process ID) del proceso uvicorn que está corriendo:

```
ps aux | grep unicorn
```

Te mostrará algo como:

```
ubuntu 6523 0.2 1.1 55512 23456 ? S 18:15 0:02 unicorn main:app --host 0.0.0.0 --port  
8000
```

Aquí el PID es 6523.

Luego simplemente lo detienes con:

*kill 6523*

**Salir del venv**

deactivate

## Git

### Instalar y configurar Git

instalar git

```
sudo apt install -y git
```

verificar

```
git --version
```

configurar usuario global (cambia por tus datos)

```
git config --global user.name "Tu Nombre"
```

```
git config --global user.email "tu@correo.com"
```

revisar la configuración

```
git config --list
```

### Generar clave SSH (para usar repos remotos vía SSH — GitHub, GitLab):

generar clave ed25519 (más moderna). Presiona Enter para aceptar ruta por defecto y opcionalmente una passphrase.

```
ssh-keygen -t ed25519 -C "tu@correo.com"
```

inicia el agente SSH y añade la clave

```
eval "$(ssh-agent -s)"
```

```
ssh-add ~/.ssh/id_ed25519
```

muestra la clave pública para copiarla y pegarla en GitHub/GitLab (Settings → SSH keys)

```
cat ~/.ssh/id_ed25519.pub
```

verifica la conexión

```
ssh -T git@github.com
```

## Ayuda Git

### Crear el repositorio en GitHub (desde la web)

1. Entra a GitHub y haz clic en New repository
2. Nombre del repo: por ejemplo “mi\_primer\_python”
3. Opcional: descripción
4. Marca Public o Private
5. NO marques Add README, ni Add .gitignore, ni Add license (porque ya tienes tu proyecto local y evitarás conflictos)
6. Clic en Create repository

Después de eso, GitHub te mostrará instrucciones, algo como esto (lo usaremos en el paso 3):

```
git remote add origin git@github.com:tu_usuario/mi_primer_python.git  
git branch -M main  
git push -u origin main
```

### Clonar el repositorio (solo si aún no tienes carpeta local)

Si el proyecto ya existe en tu Ubuntu → pasa al paso 3

Si aún NO lo has creado → clonamos el repo vacío:

```
git clone git@github.com:tu_usuario/mi_primer_python.git  
cd mi_primer_python
```

### Subir tu proyecto local al repositorio (primer push)

Si ya tienes una carpeta con tu programa (hola.py, etc.), entra en esa carpeta:

```
cd ~/mi_primer_python
```

Inicializa git:

```
git init
```

Agrega el remoto (cambia tu\_usuario por tu usuario real):

```
git remote add origin git@github.com:tu_usuario/mi_primer_python.git
```

Verifica que quedó bien:

```
git remote -v
```

Luego agrega todos los archivos y haz el primer commit:

```
git add .  
git commit -m "Primer commit - programa inicial en Python"
```

Sube al repositorio:

```
git branch -M main
```

```
git push -u origin main
```

Si todo va bien, lo verás en GitHub

### Cómo actualizar el repositorio cuando hagas cambios

Cada vez que edites algo:

```
git status      # ver cambios
```

```
git add .       # agregar todo
```

```
git commit -m "Descripción del cambio"
```

```
git push        # subir a GitHub
```

Ejemplo:

```
git add hola.py
```

```
git commit -m "Agrego interacción con el usuario"
```

```
git push
```

### Cómo bajar cambios desde GitHub (pull)

Si alguien más modifica el repo o lo editas desde la web:

```
git pull
```

Errores comunes:

Error	Solución
fatal: remote origin already exists	git remote remove origin y luego volver a agregar
permission denied (publickey)	No configuraste SSH: ssh -T git@github.com
updates were rejected because the remote contains work	Primero git pull --rebase origin main
branch 'main' not found	Cambiaste a master: git branch -M main