

# Ayudantía 6

Nginx y express en un vps, git y github



Sebastián Riquelme

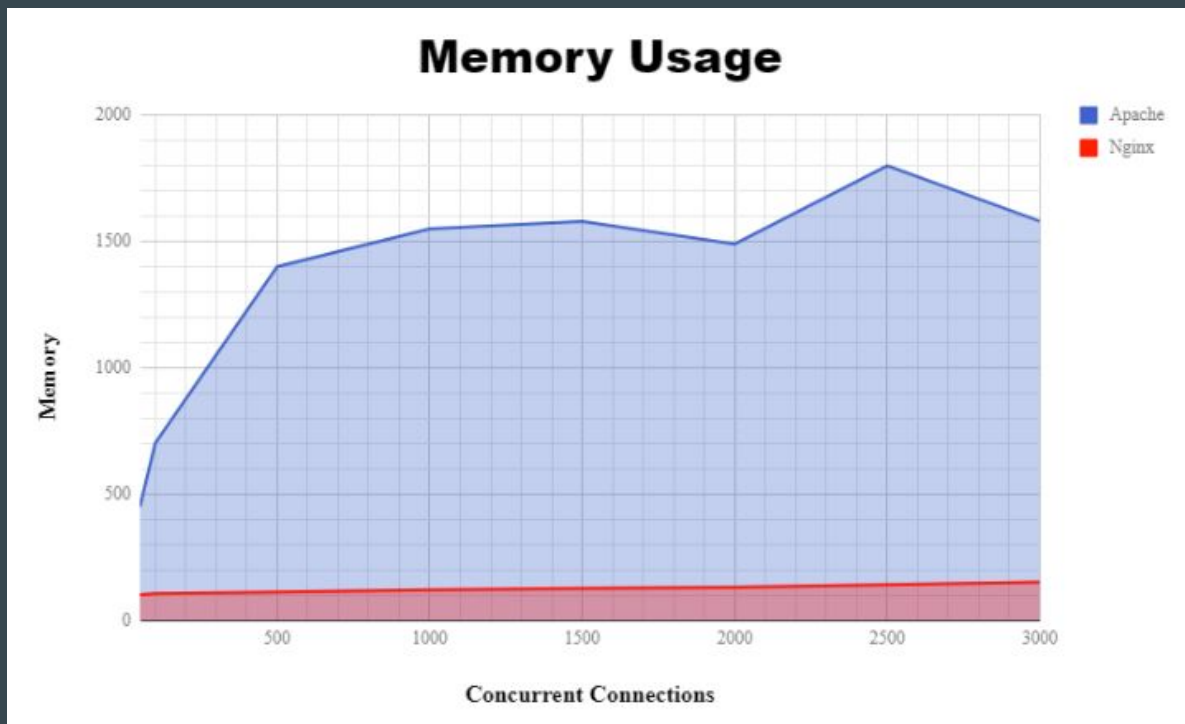
# ¿Qué es Nginx?

Nginx es un servidor web/proxy inverso ligero de alto rendimiento. Es software libre y de código abierto.



# Ventaja de Nginx frente a apache

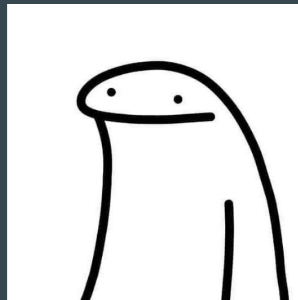
Nginx consume menos recursos que Apache, esto es notable en el consumo de RAM.



# Instalar y configurar Nginx

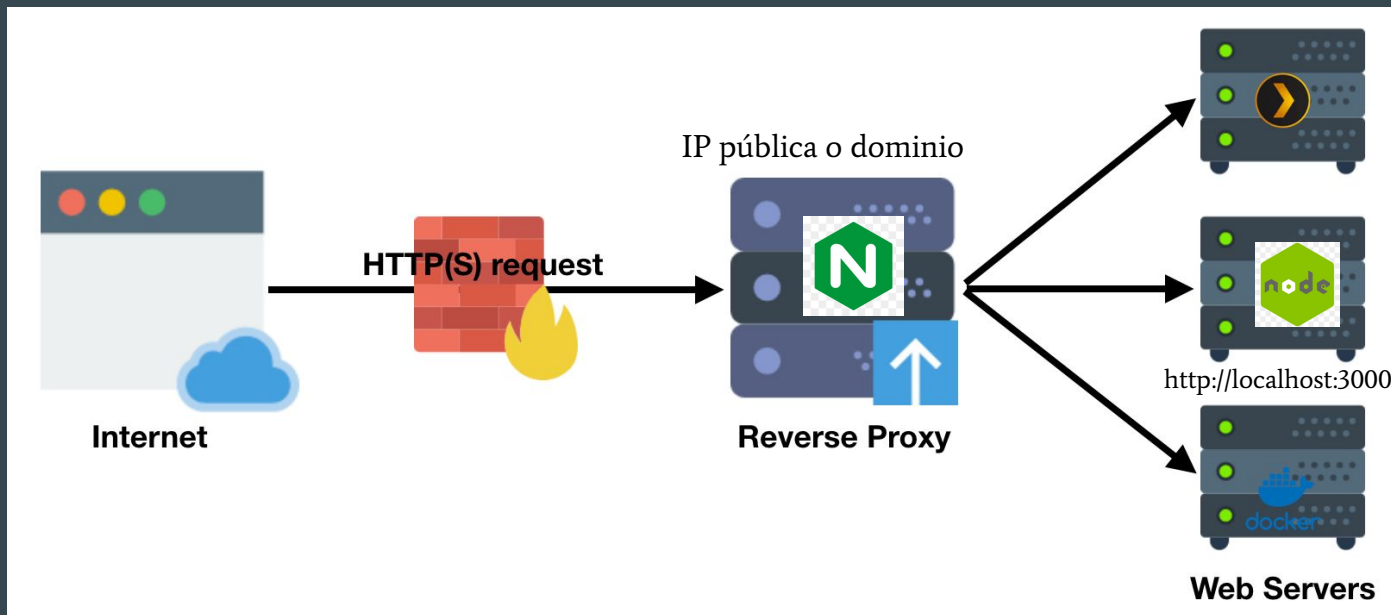
- **Instalar Nginx:**  
sudo apt update  
sudo apt install nginx
- **Configurar firewall**  
sudo ufw allow 'Nginx HTTP'
- **Status servidor web**  
systemctl status nginx
- **Visitar servidor**  
http://IP\_servidor
- **Editar archivo de configuración Nginx**  
nano /etc/nginx/sites-enabled/default.conf
- **Contenido del archivo de configuración**  
Editar el archivo con los siguientes datos:  
[github\\_default.conf](#)
- **Reiniciar Nginx**  
sudo systemctl restart nginx

Con esto habremos  
configurado un reverse proxy!



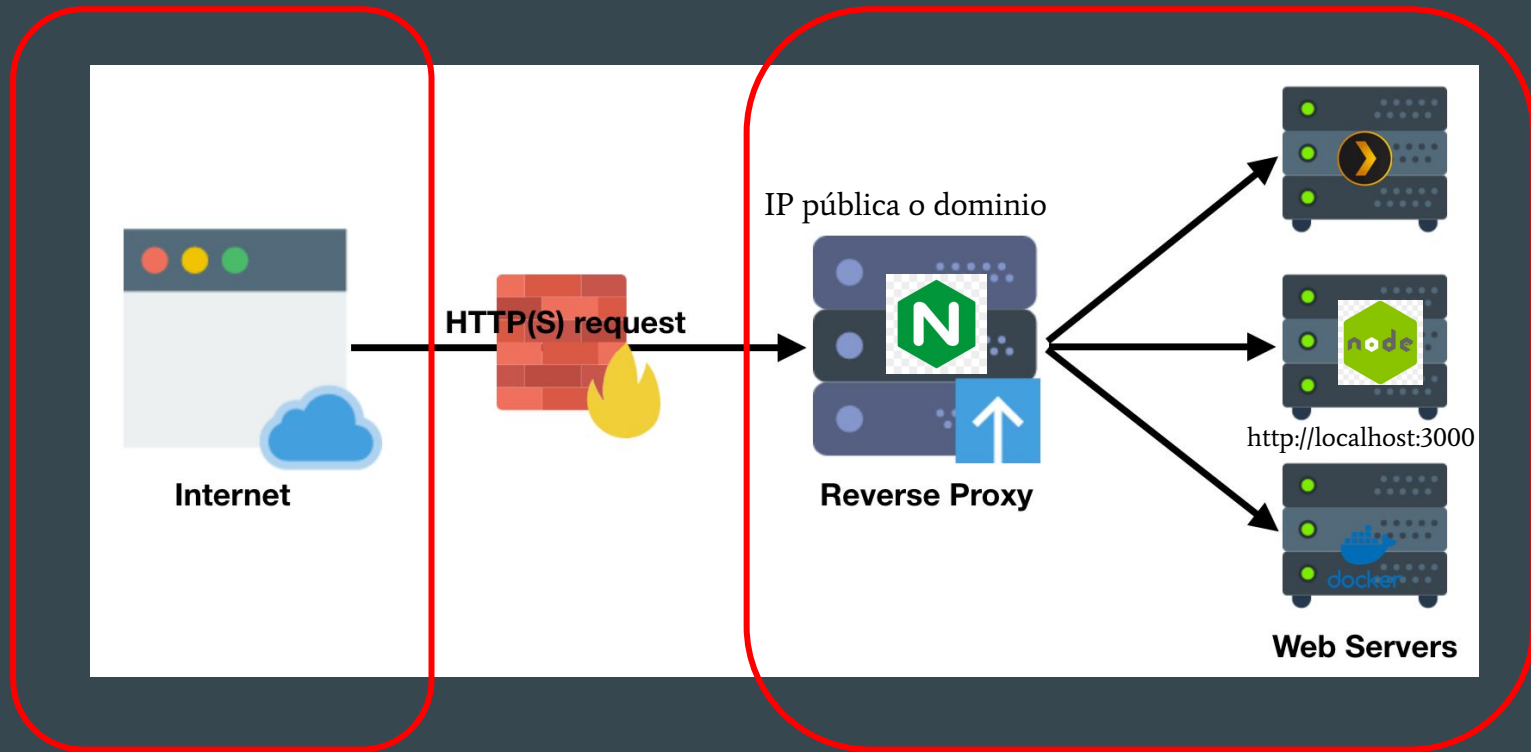
# ¿Qué es un reverse proxy?

Recibimos consultas y las traducimos a uno de nuestros servicios.

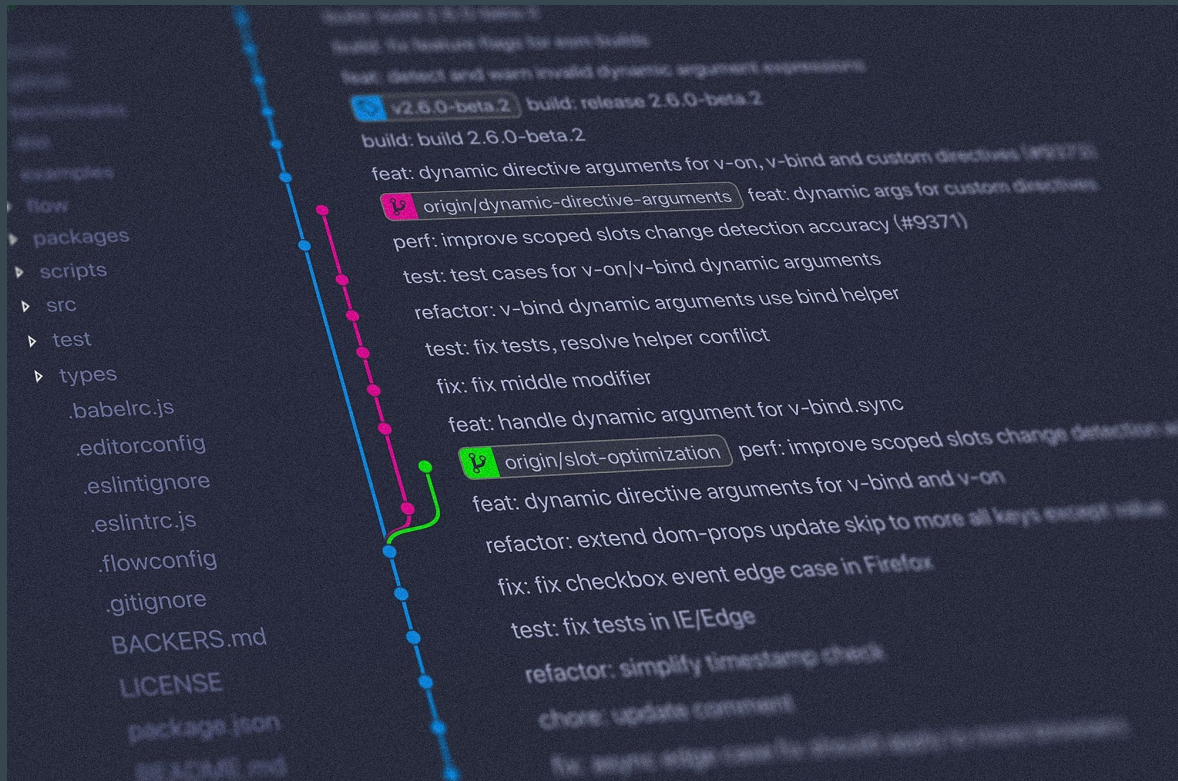


Cliente (Ej: navegador)

Instancia EC2 (VPS)



# Git y github



# ¿Qué es Git?

Git es un sistema de control de versiones distribuido, diseñado por Linus Torvalds (Creador de linux). Está pensando en la eficiencia y la confiabilidad del mantenimiento de versiones de aplicaciones cuando estas tienen un gran número de archivos de código fuente.

Git está optimizado para guardar cambios de forma incremental.

Permite contar con un historial, regresar a una versión anterior y agregar funcionalidades.

Lleva un registro de los cambios que otras personas realicen en los archivos.

Git fue diseñado para operar en un entorno Linux. Actualmente, es multiplataforma, es decir, es compatible con Linux, MacOS y Windows. En la máquina local se encuentra Git, se utiliza bajo la terminal o línea de comandos y tiene comandos como merge, pull, add, commit y clone, entre otros.



# ¿Qué es github?

Github es un servicio de alojamiento en la nube que ofrece a los desarrolladores repositorios de software usando el sistema de control de versiones de git.



# Por si es confuso

Aquí una imagen comparativa



**Herramienta de  
control de versiones  
distribuida**

**Herramienta de  
código abierto que  
los desarrolladores  
instalan localmente  
para gestionar el  
código fuente**



**Plataforma  
basada en la  
nube**

**Servicio en línea al  
que los  
desarrolladores que  
utilizan Git pueden  
conectarse y cargar  
o descargar recursos**

# Crear un repositorio en github (nube)

## 1. Creamos el repo en github


### Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

---

Owner \*

Repository name \*


 SebastianRiquelmeM ▾

 /


Great repository names are short and memorable. Need inspiration? How about [fictional-parakeet?](#)

Description (optional)

---

☒  **Public**

Anyone on the Internet can see this repository. You choose who can commit.

☐  **Private**

You choose who can see and commit to this repository.

---

**Initialize this repository with:**

Skip this step if you're importing an existing repository.

☐ **Add a README file**

This is where you can write a long description for your project. [Learn more.](#)

**Add .gitignore**

Choose which files not to track from a list of templates. [Learn more.](#)

.gitignore template: None ▾

# Nomenclatura git y github

- **Repositorio**

Es la base de datos donde se almacena el historial del código (Podemos ver los diferentes commit)

- **Commit**

Registro de cambios en el código, queda en el historial del repositorio.

- **Branch (Rama en español)**

Entorno de trabajo independiente. Podemos tener varias branch dentro de un mismo repo.

- **Checkout**

Cambiar de una rama a otra.

- **Merge**

Combinar dos ramas en una.

# Nomenclatura git y github

- **Rebase**

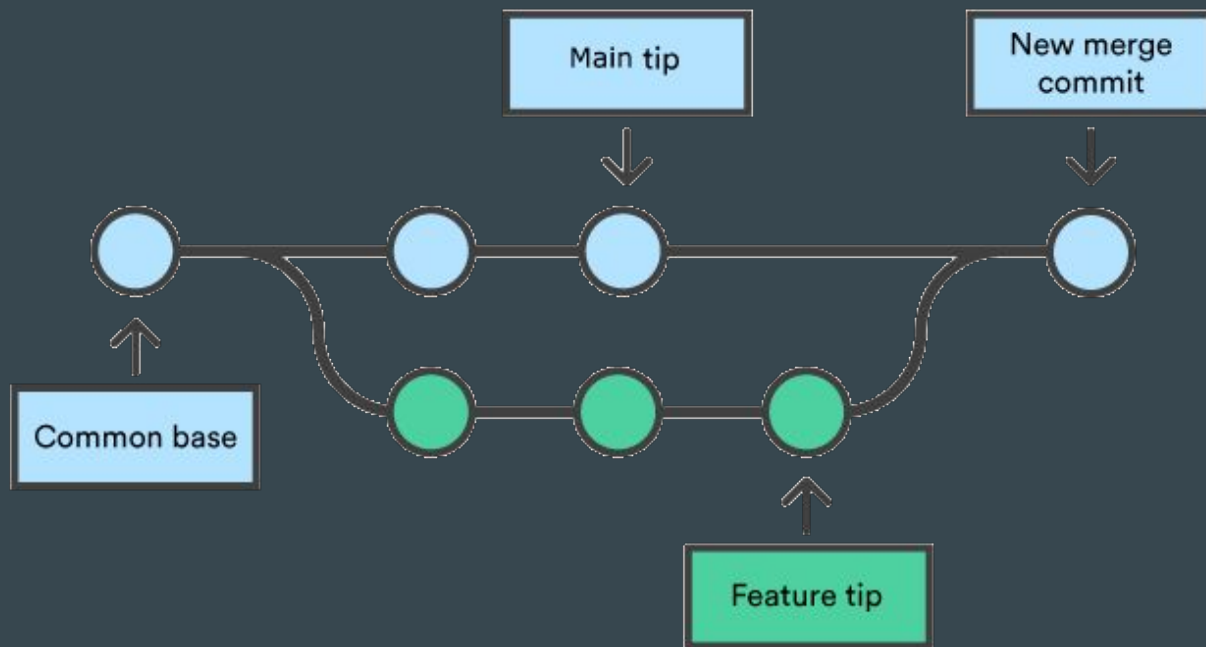
Reemplazar una rama continuando con los cambios de otra.

- **Pull**

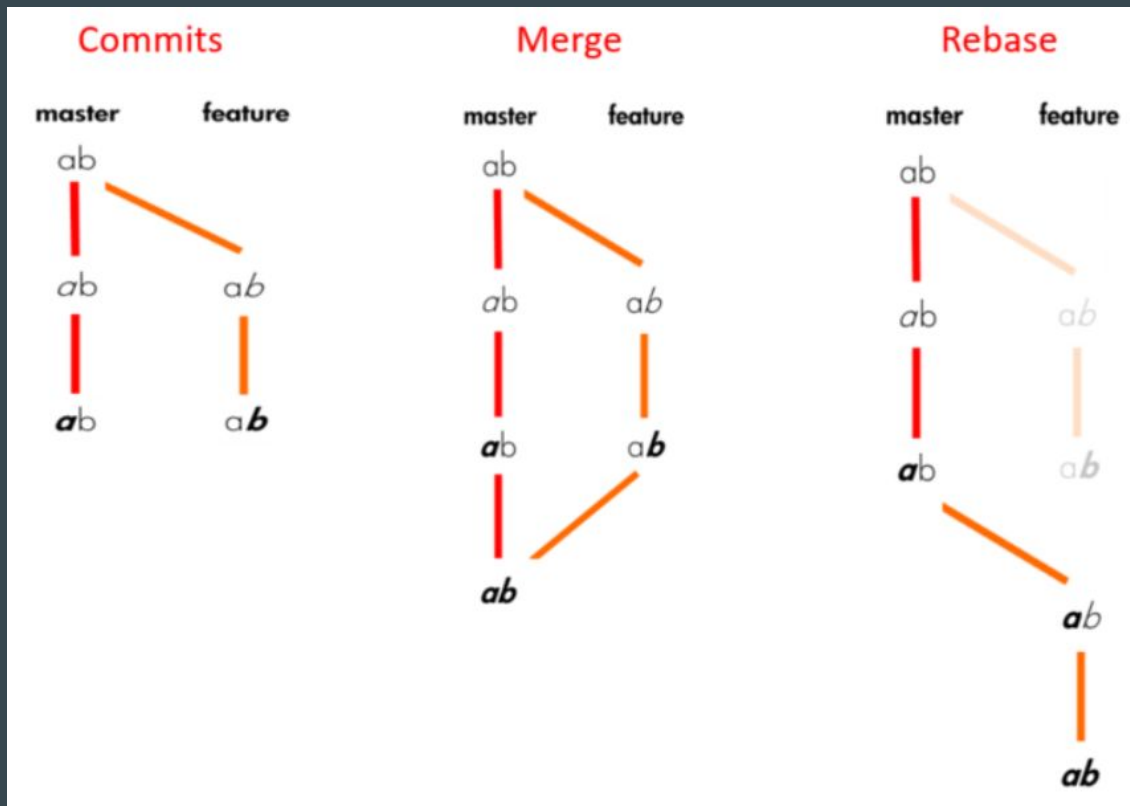
Actualizar repo local con los nuevos cambios del repo remoto.

# Nomenclatura git y github

- Un diagrama de git merge (juntar dos ramas)



# Diagrama para diferenciar commit, merge y rebase



# Instalar git

apt install git

Sino, ver <https://git-scm.com/download/linux>





# Conectarnos a github desde la terminal local

## Github CLI

Esta es la opción que más me gusta, pues es muy cómoda y es la que menos problemas me ha dado.

- **Instrucciones de instalación**

- **Windows y mac**

- <https://cli.github.com/manual/installation>

- **Linux**

- [https://github.com/cli/cli/blob/trunk/docs/install\\_linux.md](https://github.com/cli/cli/blob/trunk/docs/install_linux.md)

- **Iniciar sesión**

- `gh auth login`

Con esto tendremos pleno acceso a nuestro github, para descargar y modificar repositorios públicos y privados. ¡Cuidado con dejar su sesión abierta en PC públicos!

Cabe destacar que también hay otros métodos como SSH para github.

# Luego seguimos las instrucciones

A terminal window with a dark background. The title bar shows 'seba@seba-pc: ~' and standard window controls. The terminal text shows the command 'gh auth login' being executed, followed by a prompt to select an account. The user has selected 'GitHub.com' from a list that also includes 'GitHub Enterprise Server'.

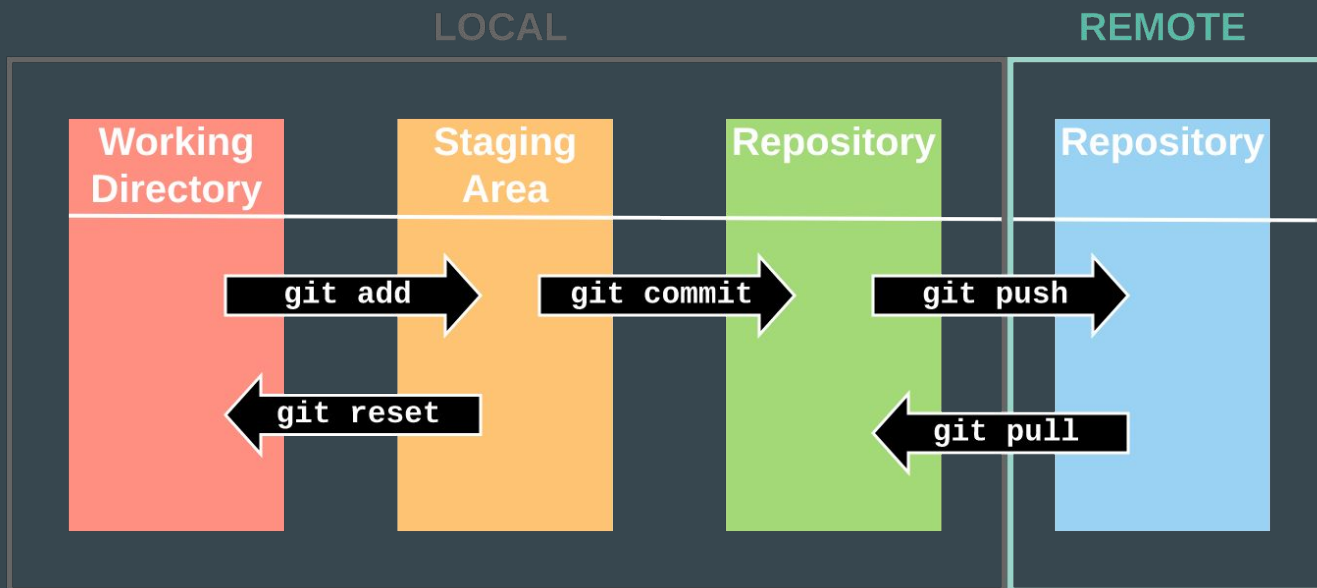
```
seba@seba-pc: ~  
seba@seba-pc:~$ gh auth login  
? What account do you want to log into? [Use arrows to move, type to filter]  
  
> GitHub.com  
   GitHub Enterprise Server
```

## 2. Gestión entre git y github

- **Iniciamos un repositorio en local**  
git init
- **Añadimos todos los archivos de la carpeta local (Antes es importante el gitignore)**  
El punto se refiere al directorio actual en que estoy situado  
git add .
- **Creamos una rama**  
git branch -M main
- **Añadimos a git el repositorio remoto (github)**  
git remote add origin <URL>.git  
Por ejemplo:  
git remote add origin [https://github.com/SebastianRiquelmeM/repositorio\\_prueba.git](https://github.com/SebastianRiquelmeM/repositorio_prueba.git)
- **Subir a github nuestro git local**  
git push -u origin main

# Diagrama de trabajo entre git y github

Local es git y remote git



# Subir nuevos cambios (Ya teniendo iniciado el repo local)

- **Añadimos todos los archivos de la carpeta, es importante estar situado en la carpeta inicial del repo (Si tenemos activado ver archivos ocultos es donde hay un .git)**  
git add .
- **Creamos un nuevo commit**  
git commit -m "Mensaje descriptivo"
- **Subimos el commit**  
git push

# Cargar archivos a nuestro servidor con git (O en cualquier carpeta)

- Hacemos un clone en la carpeta que queramos tener nuestro proyecto  
Esto copia todos los archivos de nuestro github, también podemos crear commits desde aquí si tenemos los permisos correctos para el repositorio remoto.

```
git clone <URL>.git .
```

Cabe destacar que podemos descargar cualquier repositorio de github con este comando.

- Cuando ya tenemos el repositorio clonado, pero queremos cargar nuevos cambios:

```
git pull
```

# Existen más repositorios remotos aparte de github

Podemos trabajar con ellos desde git.

