

GITHUB



Nom: Gerard Sanchez
Nom: Valeria Santana
Módul: M8_Desplegament de Apps Web

INDEX

1. Introducción.....	2
2. Objetivo del manual.....	2
3. Configuración Inicial.....	2
4. Repositorios.....	3
4.1. Inicializar un repositorio nuevo.....	3
4.2. Clonar un repositorio existente.....	3
4.3. Agregar Cambios.....	4
4.4. Confirmar Cambios.....	4
5. Trabajar con Ramas.....	5
5.1. Crear Rama.....	5
5.2. Fusionar Ramas.....	5
5.3. Eliminar Ramas.....	6
6. Trabaja en Remoto.....	6
6.1. Subir Cambios.....	6
6.2. Actualizar Rama.....	7
7. Github en web.....	7

1. Introducción

Git es un sistema de control de versiones distribuido que permite a los desarrolladores rastrear cambios en el código, colaborar en proyectos y revertir cambios cuando sea necesario. A diferencia de los sistemas centralizados, Git ofrece a cada desarrollador una copia completa del repositorio, facilitando el trabajo en equipo y sin conexión.

GitHub es una plataforma basada en la web que utiliza Git para alojar proyectos y facilitar la colaboración. Proporciona una interfaz gráfica para gestionar repositorios, realizar revisiones de código y automatizar pruebas y despliegues. También permite crear sitios web estáticos mediante GitHub Pages.

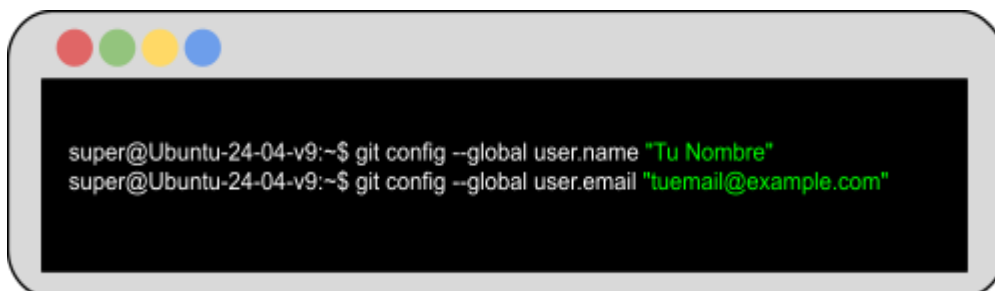
2. Objetivo del manual

Este manual ofrece una referencia rápida a los comandos más comunes de Git y GitHub, ideal tanto para principiantes como para usuarios experimentados. Al utilizar estas herramientas, mejorarás tu capacidad para gestionar proyectos y colaborar de manera eficiente en el desarrollo de software.

3. Configuración Inicial

Antes de empezar a usar Git, es necesario configurar tus datos personales (nombre de usuario y correo electrónico). Estos datos se adjuntará a cada commit que hagas.

- **user.name:** Define el nombre de usuario que se mostrará en los commits.
- **user.email:** Establece el correo que estará asociado a los commits



Podrás verificar la información con:

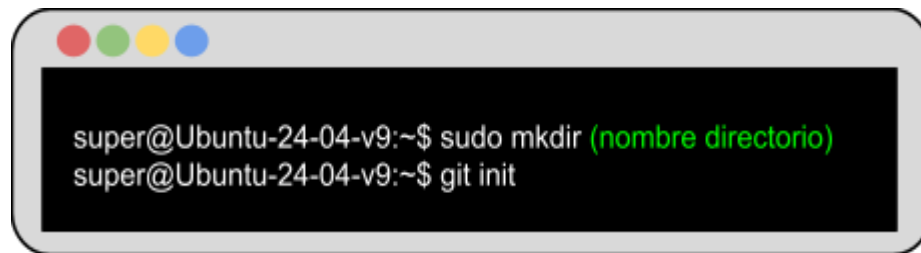


```
super@Ubuntu-24-04-v9:~$ git config --list
```

4. Repositorios

4.1. Inicializar un repositorio nuevo

Para comenzar un nuevo proyecto, abre la terminal y crea un directorio para preparar un repositorio Git que gestione tu historial. Utiliza el siguiente comando:

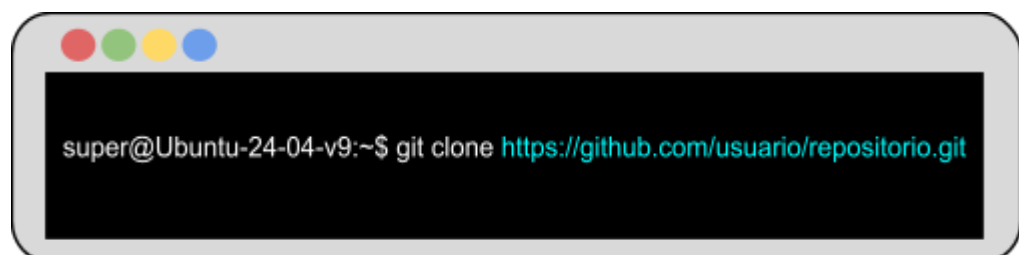


```
super@Ubuntu-24-04-v9:~$ sudo mkdir (nombre directorio)
super@Ubuntu-24-04-v9:~$ git init
```

Este comando inicializa un nuevo repositorio en el directorio actual. Crea una carpeta oculta `.git` que contiene toda la información necesaria sobre el historial del proyecto.

4.2. Clonar un repositorio existente

Si deseas trabajar en un proyecto que ya existe en GitHub o en otro servidor Git, utiliza el comando `git clone` para descargar una copia local del repositorio:




```
super@Ubuntu-24-04-v9:~$ git clone https://github.com/usuario/repositorio.git
```

La URL es del repositorio que deseas clonar.

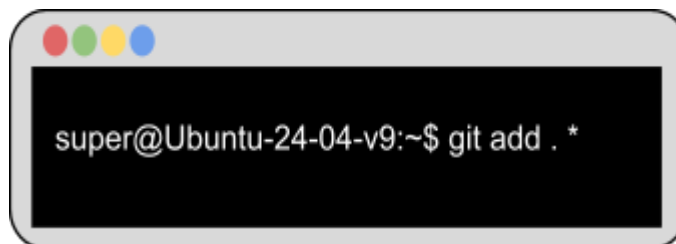
4.3. Agregar Cambios

Para agregar archivos al área de staging (staging area):



```
super@Ubuntu-24-04-v9:~$ git add nombre-del-archivo
```

Para agregar todos los archivos modificados:



```
super@Ubuntu-24-04-v9:~$ git add . *
```

4.4. Confirmar Cambios

Para confirmar los cambios añadidos:



```
super@Ubuntu-24-04-v9:~$ git commit -m 'mensaje del commit'
```

5. Trabajar con Ramas

5.1. Crear Rama

¿Qué es una Rama?

Una rama en Git es como una copia del proyecto donde puedes hacer cambios sin afectar la versión principal del código. Piensa en las ramas como diferentes caminos que puedes tomar mientras trabajas en un proyecto.

Creemos y nos movemos a la nueva branch:

A terminal window with a grey title bar and three colored window control buttons (red, green, blue) on the left. The terminal has a black background with white text. The prompt is 'super@Ubuntu-24-04-v9:~\$' and the command entered is 'git checkout -b nombre-de-la-rama', where 'nombre-de-la-rama' is highlighted in green.

```
super@Ubuntu-24-04-v9:~$ git checkout -b nombre-de-la-rama
```

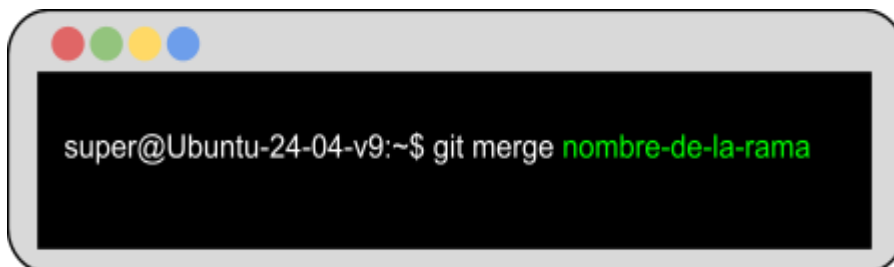
Con el **checkout** nos cambiaremos de ramas.

Y con el **git branch nombre-de-la-rama** solamente la crearemos

5.2. Fusionar Ramas

Fusionar ramas significa combinar los cambios de una rama en otra. Esto es útil cuando has terminado de trabajar en una nueva función o corregido un error y quieres que esos cambios se integren en la rama principal del proyecto.

Una vez que estás en la rama correcta, usa el siguiente comando para fusionar otra rama (por ejemplo, nueva-función) en la rama actual:

A terminal window with a grey title bar and three colored window control buttons (red, green, blue) on the left. The terminal has a black background with white text. The prompt is 'super@Ubuntu-24-04-v9:~\$' and the command entered is 'git merge nombre-de-la-rama', where 'nombre-de-la-rama' is highlighted in green.

```
super@Ubuntu-24-04-v9:~$ git merge nombre-de-la-rama
```

5.3. Eliminar Ramas

Para eliminar una rama:

A terminal window with a grey title bar and three colored window control buttons (red, green, blue) on the left. The terminal has a black background with white text. The prompt is 'super@Ubuntu-24-04-v9:~\$' and the command entered is 'git branch -d nombre-de-la-rama', where 'nombre-de-la-rama' is highlighted in green.

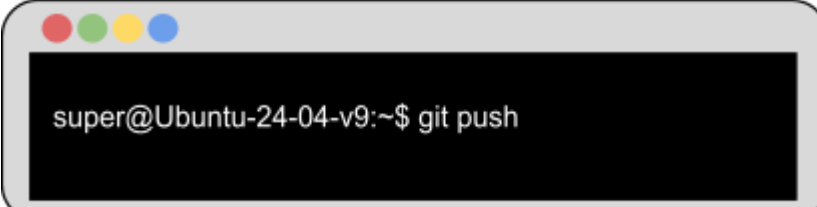
```
super@Ubuntu-24-04-v9:~$ git branch -d nombre-de-la-rama
```

6. Trabaja en Remoto

6.1. Subir Cambios

¿Qué es un Repositorio Remoto?

Un **repositorio remoto** es una versión de tu proyecto que se encuentra en un servidor. Puedes subir tus cambios a este repositorio o descargar cambios de otros colaboradores.

A terminal window with a grey title bar and three colored window control buttons (red, green, blue) on the left. The terminal has a black background with white text. The prompt is 'super@Ubuntu-24-04-v9:~\$' and the command entered is 'git push'.

```
super@Ubuntu-24-04-v9:~$ git push
```

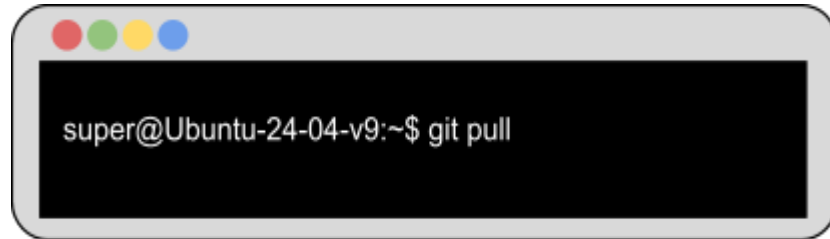
Después de ejecutar este comando, se te pedirá tu usuario de GitHub y, como contraseña, usarás el token que proporciona GitHub.

Para conseguir un token de acceso personal en GitHub, sigue estos pasos:

- Accede a **Settings** (Configuración) en tu perfil de GitHub.
- Navega a **Developer settings** (Configuración de desarrollador).
- Selecciona **Personal access tokens**.
- Haz clic en **Tokens (classic)**.
- Genera un nuevo token presionando **Generate new token**.

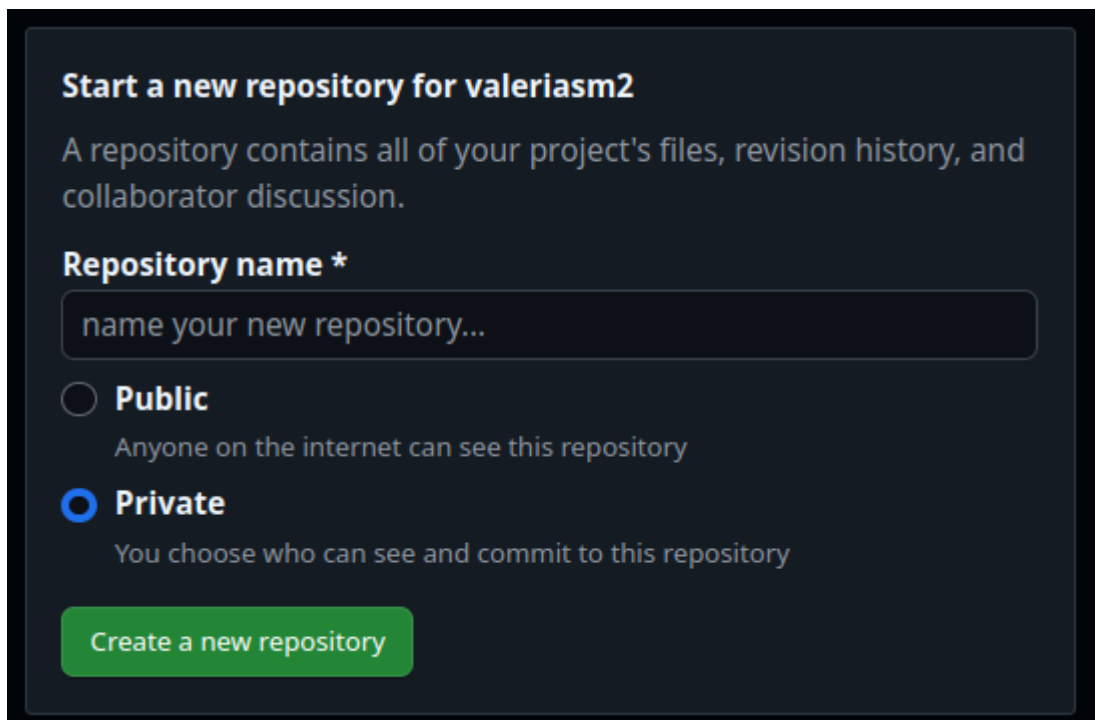
6.2. Actualizar Rama

Con el `git pull` bajaremos todos los cambios que se hayan añadido la última vez.



7. Github en web

En la pantalla principal de GitHub, verás un recuadro para crear un nuevo repositorio directamente en la web. Luego, es recomendable continuar haciendo los cambios desde la terminal.

A screenshot of the GitHub 'Start a new repository' form. The form has a dark gray background. At the top, it says 'Start a new repository for valeriam2'. Below that, it explains: 'A repository contains all of your project's files, revision history, and collaborator discussion.' There is a text input field for 'Repository name *' with the placeholder text 'name your new repository...'. Below the input field, there are two radio button options: 'Public' (with the description 'Anyone on the internet can see this repository') and 'Private' (with the description 'You choose who can see and commit to this repository'). The 'Private' option is selected. At the bottom, there is a green button labeled 'Create a new repository'.

Asignaremos un nombre a nuestro repositorio y elegiremos si queremos que sea público o privado para los demás usuarios. También podemos crear un nuevo repositorio haciendo clic en el botón verde "New" que aparece en la columna de la izquierda, junto al icono de un libro.