

Exposición



Git y GitHub



Integrantes grupo 10:

Agostina Finucci

Micaela Hermosa

Nallive Trujillo

Sofía Rincón

Sofía Speciali



Git y GitHub
en 10 minutos





- Software de control de versiones.
- Permite llevar un registro de todos los cambios que ocurren dentro de un conjunto de archivos o directorio.
- Así, se puede tener control de todos los cambios realizados y acceder a las distintas versiones cuando se desee.
- Muy útil para compartir y coordinar el trabajo que varias personas realizan sobre un mismo proyecto.

Glosario

Secciones de git

1. Working directory (Directorio de trabajo): se crean y modifican archivos.

Untracked

Modified

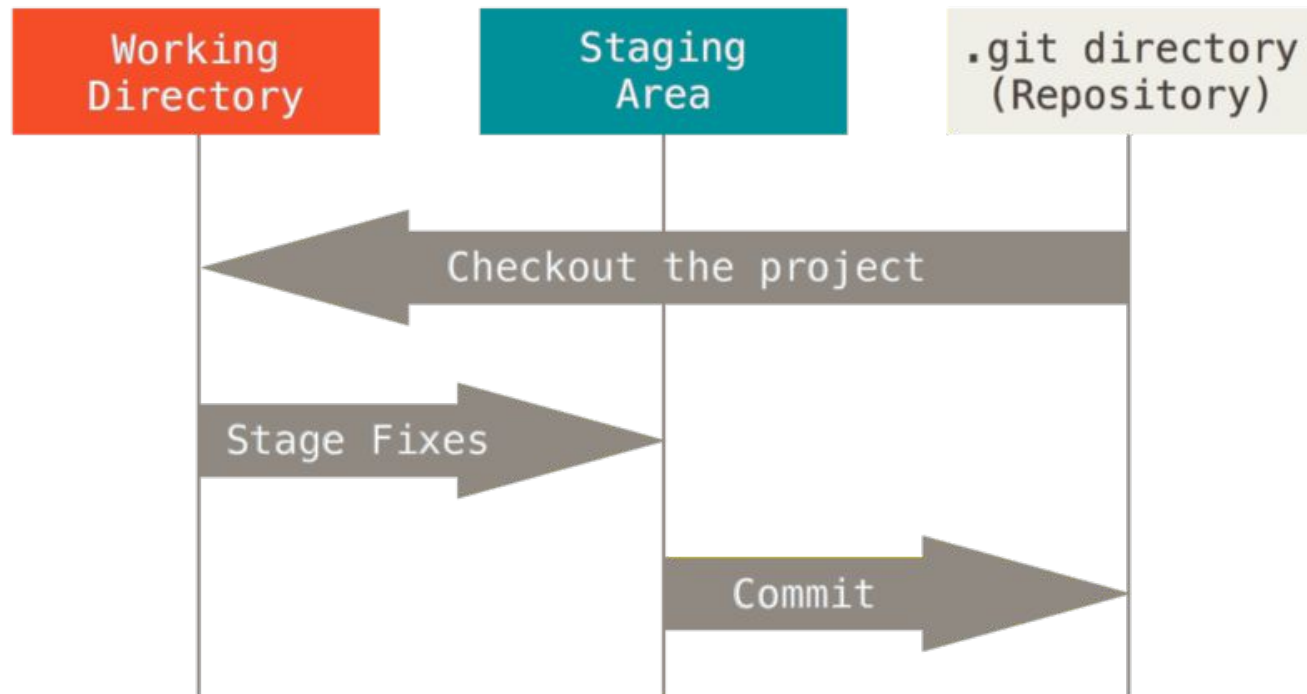
2. Staging area (área de preparación): agregar archivos creados y/o modificados.

Staged

3. Local repository (Repositorio local): guarda las versiones del proyecto al realizar commit.

Committed

Secciones de git



Repositorio

- Es el lugar donde se irán almacenando los archivos de nuestro proyecto.

Repositorio local



Repositorios que se alojan en la computadora.

Repositorio remoto



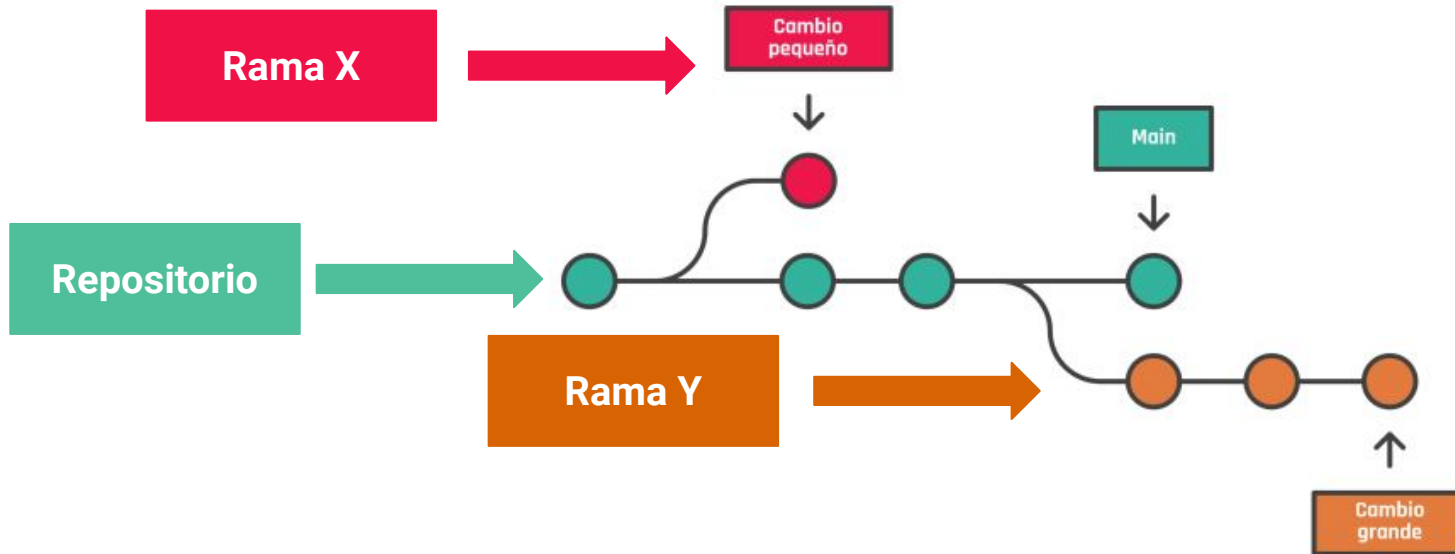
Repositorios que se alojan en GitHub.

Commit

- Permite hacer seguimiento de los cambios realizados, ya que genera un punto cronológico en la línea del tiempo del proyecto.

Rama o branch

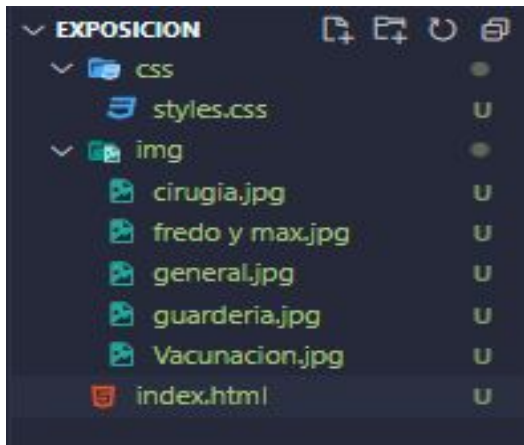
- Es una línea de desarrollo que tiene como base el repositorio, pero que no actúa directamente sobre él, o sea que no lo modifica.



Crea e Inicia Git

1. **git init**: crea e inicia un repositorio llamado .git y es el encargado de llevar registro de los cambios que sucedan en los archivos.

```
ASUS@DESKTOP-5R9JJ5G MINGW64 ~/Desktop/Exposicion
$ git init
Initialized empty Git repository in C:/Users/ASUS/Desktop/Exposicion/.git/
```



Untracked: Archivos sin seguimiento.

Configurando Git

El comando **git config** es una función útil que sirve para definir valores de configuración de Git en el ámbito de un proyecto global o local:

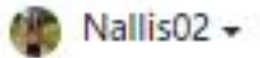
Global

```
git config --global user.name "usuarioGitHub"
```

```
git config --global user.email correogithub@example.com
```

```
ASUS@DESKTOP-5R9JJ5G MINGW64 ~/Desktop/Exposicion (master)
$ git config --global user.name "Nallis02"

ASUS@DESKTOP-5R9JJ5G MINGW64 ~/Desktop/Exposicion (master)
$ git config user.name
Nallis02
```



```
ASUS@DESKTOP-5R9JJ5G MINGW64 ~/Desktop/Exposicion (master)
$ git config --global user.email nallive@gmail.com

ASUS@DESKTOP-5R9JJ5G MINGW64 ~/Desktop/Exposicion (master)
$ git config user.email
nallive@gmail.com
```

nallive14@gmail.com – Primary ⓘ


Configurando Git

Local

```
$ git config user.name "usuariogithub"
```

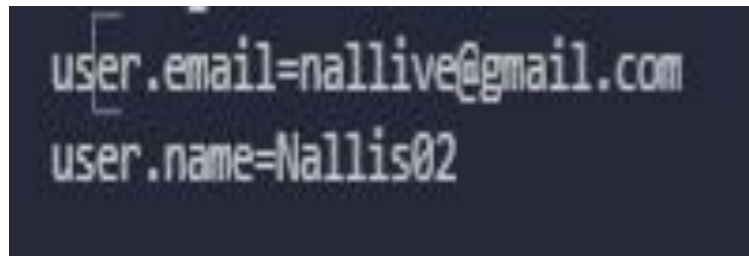
```
$ git config user.email correogithub@example.com
```

```
$ git config --list (Ver la configuración de Git)
```



A terminal window with a dark background. The prompt shows the user is on a Windows machine (MINGW64) in the directory ~/Desktop/Exposicion, on the master branch. The command `$ git config --list` has been entered.

```
ASUS@DESKTOP-5R9JJ5G MINGW64 ~/Desktop/Exposicion (master)  
$ git config --list
```



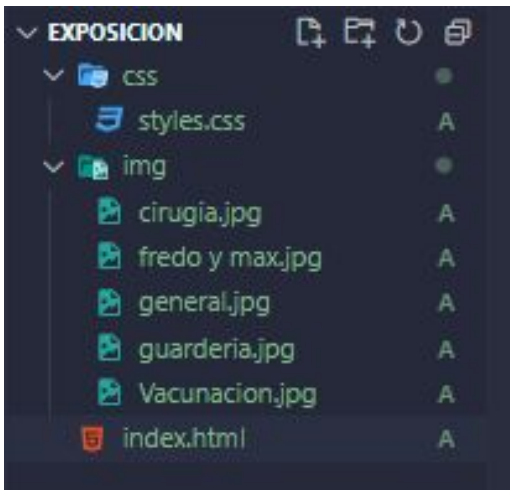
A terminal window showing the output of the `git config --list` command. It displays two lines of configuration: `user.email=nallive@gmail.com` and `user.name=Nallis02`.

```
user.email=nallive@gmail.com  
user.name=Nallis02
```

Del Working Directory al Staging Area

2. **git add .** : agrega todos los cambios actuales. Estado agregado (A), en seguimiento.

```
ASUS@DESKTOP-5R9JJ5G MINGW64 ~/Desktop/Exposicion (master)
$ git add .
```

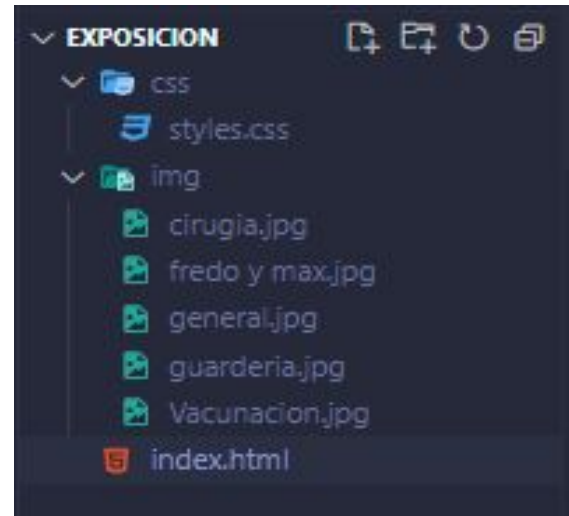


Staged: preparado para ser committed (confirmado).

Del Staging Area al Local Repository

3. **git commit -m "mensaje"**: crea un commit. Este es como una fotografía del archivo que se agrega al repositorio. En cualquier momento se puede regresar en el tiempo a cualquier foto que se haya tomado.

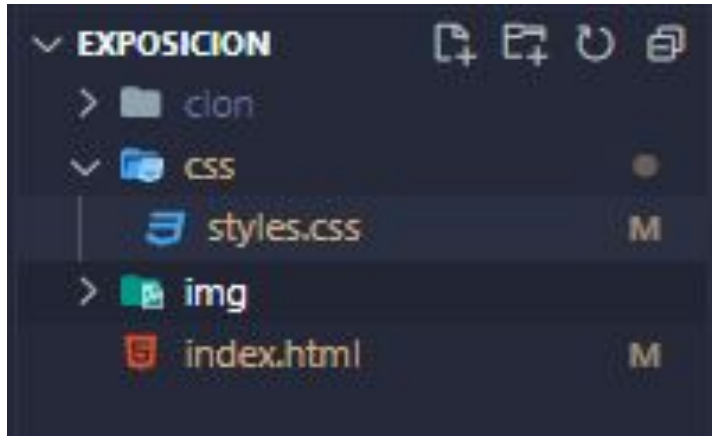
```
ASUS@DESKTOP-5R9JJ5G MINGW64 ~/Desktop/Exposicion (master)
$ git commit -m "Primer commit"
[master (root-commit) 4093e55] Primer commit
7 files changed, 21 insertions(+)
create mode 100644 css/styles.css
create mode 100644 img/Vacunacion.jpg
create mode 100644 img/cirugia.jpg
create mode 100644 img/fredo y max.jpg
create mode 100644 img/general.jpg
create mode 100644 img/guarderia.jpg
create mode 100644 index.html
```



Committed

Estado Modificado

Cuando se realiza alguna modificación a los archivos ya agregados en el repositorio se muestra la letra M (modificado) al lado del archivo al que se le haya hecho la modificación.



Modified: modificado

```
ASUS@DESKTOP-5R9JJ5G MINGW64 ~/Desktop/Exposicion (login)
$ git status
On branch login
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   css/styles.css
        modified:   index.html

no changes added to commit (use "git add" and/or "git commit -a")
```

Estado Deleted

Cuando se elimina alguno de los archivos ya agregados en el repositorio aparecerá un D de **Deleted** para indicar que ese archivo fue eliminado.

```
ASUS@DESKTOP-5R9JJ5G MINGW64 ~/Desktop/Exposicion (login)
$ git rm img/cirugia.jpg
rm 'img/cirugia.jpg'
```

```
ASUS@DESKTOP-5R9JJ5G MINGW64 ~/Desktop/Exposicion (login)
$ git status
On branch login
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        deleted:    img/cirugia.jpg
```



Deleted: eliminado

Comandos más usados

4. **git log:** muestra el historial de commits.

```
ASUS@DESKTOP-5R9JJ5G MINGW64 ~/Desktop/Exposicion (main)
$ git log
commit 4093e55b89540465ecd1370b41bd1251a15643a9 (HEAD -> main)
Author: Nallis <nallive14@gmail.com>
Date:   Wed Jul 14 18:33:29 2021 -0500

    Primer commit
```

5. **git status:** muestra el estado de los archivos en el repositorio.

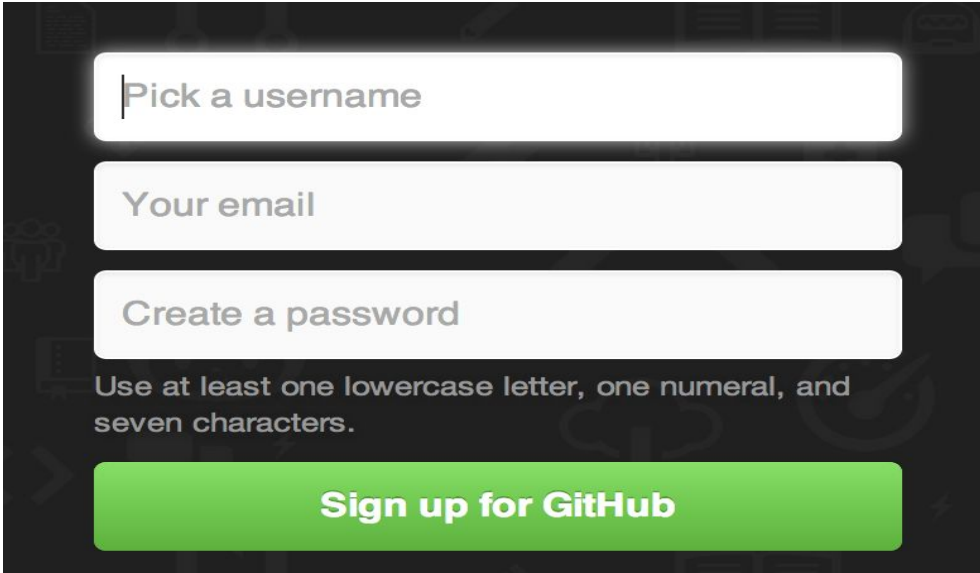
```
ASUS@DESKTOP-5R9JJ5G MINGW64 ~/Desktop/Exposicion (main)
$ git status
On branch main
nothing to commit, working tree clean
```




- GitHub es un espacio en la nube que sirve para alojar proyectos utilizando el sistema de control de versiones de Git.
- Se utiliza principalmente para la creación de código fuente de programas de ordenador.
- Es la red social de los programadores.

Crear cuenta GitHub

Para usar GitHub debemos crear una cuenta con usuario, correo y contraseña en la página de <https://github.com/>

A screenshot of the GitHub sign-up form. It features three white input fields on a dark background. The first field is labeled 'Pick a username', the second 'Your email', and the third 'Create a password'. Below the password field is a note: 'Use at least one lowercase letter, one numeral, and seven characters.' At the bottom is a green button with the text 'Sign up for GitHub'.

Pick a username

Your email

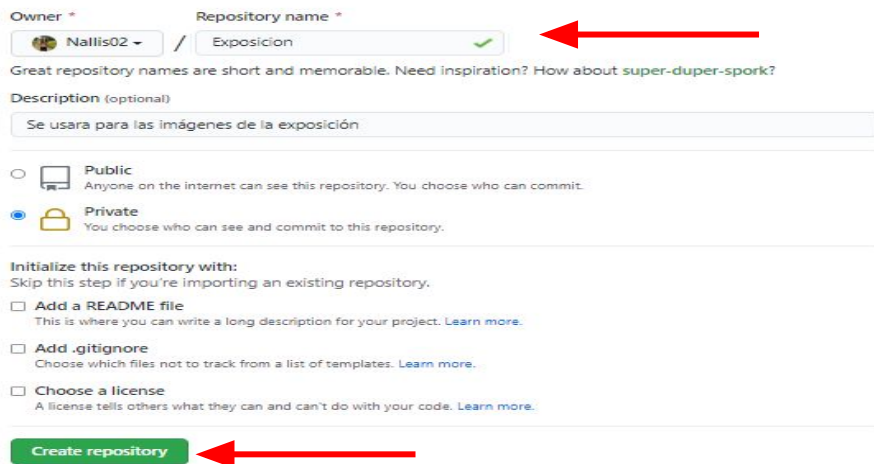
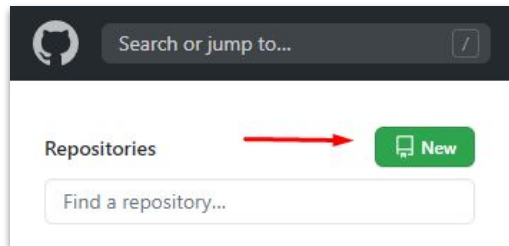
Create a password

Use at least one lowercase letter, one numeral, and seven characters.

Sign up for GitHub

Crear un repositorio en GitHub

1. En nuestra cuenta de GitHub hacemos clic en New.
2. Elegimos un nombre para el repositorio y hacemos clic en Create repository.

A screenshot of the 'Create new repository' form on GitHub. The 'Owner' field is set to 'Nallis02'. The 'Repository name' field is set to 'Exposicion' and has a green checkmark next to it. A red arrow points from the right side of the form to the 'Repository name' field. Below the name field is a 'Description (optional)' field with the text 'Se usara para las imágenes de la exposición'. There are two radio buttons for visibility: 'Public' (unselected) and 'Private' (selected). Below these are options to 'Initialize this repository with:' including 'Add a README file', 'Add .gitignore', and 'Choose a license'. At the bottom is a green 'Create repository' button with a red arrow pointing to it from the right.

Comandos más usados

git remote add origin <<https://link-con-nombre-del-repositorio>>: conecta el repositorio remoto con el local.

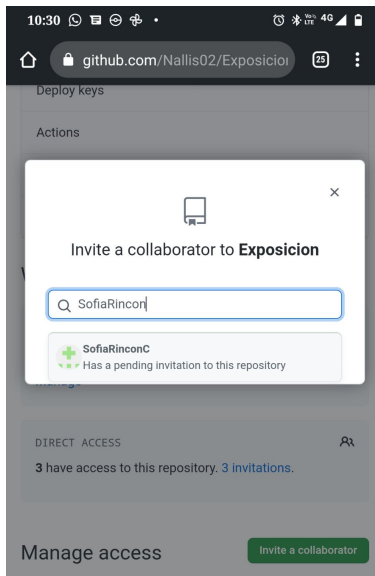
git remote --v: muestra las URLs que Git ha asociado al nombre y que serán usadas al leer y escribir en ese remoto.

```
ASUS@DESKTOP-5R9JJ5G MINGW64 ~/Desktop/Exposicion (master)
$ git remote add origin https://github.com/Nallis02/Exposicion.git

ASUS@DESKTOP-5R9JJ5G MINGW64 ~/Desktop/Exposicion (master)
$ git remote -v
origin https://github.com/Nallis02/Exposicion.git (fetch)
origin https://github.com/Nallis02/Exposicion.git (push)
```

Invitar colaboradores

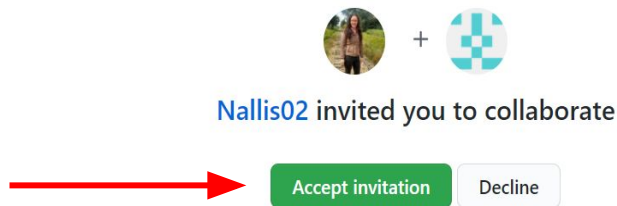
GitHub nos da la opción de trabajar de manera colaborativa en un mismo repositorio. Para ello, desde Configuración debemos invitar a los usuarios que queremos invitar a colaborar en nuestro proyecto.



Aceptar invitación

Los pasos a seguir cuando nos invitan a colaborar en un repositorio son:

1. Revisar nuestra casilla de correo y hacer clic donde dice [accept or decline](#);
2. El link nos redireccionará a GitHub donde aceptaremos la invitación a colaborar;
3. Una vez aceptada la invitación y ejecutado el comando **git clone**, está todo listo para trabajar en colaboración en el mismo repositorio.



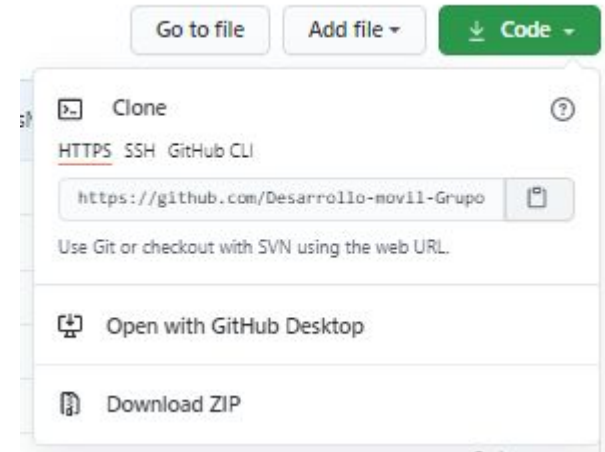
🔒 Owners of Exposition will be able to see:

- Your public profile information
- [Certain activity](#) within this repository
- Country of request origin
- Your access level for this repository
- Your IP address

Clonar repositorio

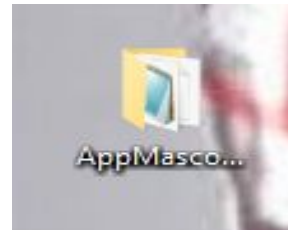
`git clone <url repositorio remoto a clonar>`: clona el repositorio remoto.

```
ASUS@DESKTOP-5R9JJ5G MINGW64 ~/Desktop
$ git clone https://github.com/Desarrollo-movil-Grupo-4/AppMascotas.git
Cloning into 'AppMascotas'...
remote: Enumerating objects: 215, done.
remote: Counting objects: 100% (215/215), done.
remote: Compressing objects: 100% (115/115), done.
Receiving objects: 97% (209/215) 201 (delta 50), pack-reused 0R
Receiving objects: 100% (215/215), 221.90 KiB | 927.00 KiB/s, done.
Resolving deltas: 100% (63/63), done.
```



```
ASUS@DESKTOP-5R9JJ5G MINGW64 ~/Desktop
$ cd AppMascotas

ASUS@DESKTOP-5R9JJ5G MINGW64 ~/Desktop/AppMascotas (main)
$
```



Escritorio

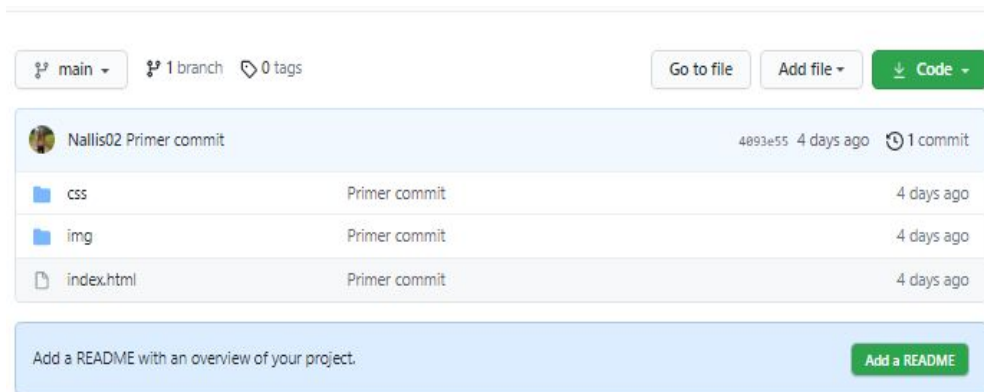
Subiendo y bajando archivos

git pull origin main: descarga los cambios del repositorio remoto.

```
ASUS@DESKTOP-5R9JJ5G MINGW64 ~/Desktop/AppMascotas (main)
$ git pull origin main
From https://github.com/Desarrollo-movil-Grupo-4/AppMascotas
 * branch          main          -> FETCH_HEAD
Already up to date.
```

git push origin main: envía cambios desde el repositorio local al remoto.

```
ASUS@DESKTOP-5R9JJ5G MINGW64 ~/Desktop/Exposicion (login)
$ git push origin main
Enumerating objects: 11, done.
Counting objects: 100% (11/11), done.
Delta compression using up to 8 threads
Compressing objects: 100% (9/9), done.
Writing objects: 100% (11/11), 160.03 KiB | 14.55 MiB/s, done.
Total 11 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/Nallis02/Exposicion.git
 * [new branch]      main -> main
```



The screenshot shows a GitHub repository interface. At the top, there's a navigation bar with 'main' selected as the branch, '1 branch', and '0 tags'. Below this, the repository name 'Nallis02 Primer commit' is displayed, along with the commit hash '4093e55' and the time '4 days ago'. A table lists the files in the repository: 'css', 'img', and 'index.html', each with a 'Primer commit' label and a '4 days ago' timestamp. At the bottom, there's a section titled 'Add a README with an overview of your project.' with a green 'Add a README' button.

File	Commit	Time
css	Primer commit	4 days ago
img	Primer commit	4 days ago
index.html	Primer commit	4 days ago

Uso de ramas

`git branch -M <main>`: renombra rama principal.

```
ASUS@DESKTOP-5R9JJ5G MINGW64 ~/Desktop/Exposicion (master)
$ git branch -M main

ASUS@DESKTOP-5R9JJ5G MINGW64 ~/Desktop/Exposicion (main)
$
```

`git branch <nueva_rama>`: crea una nueva rama.

```
ASUS@DESKTOP-5R9JJ5G MINGW64 ~/Desktop/Exposicion (main)
$ git branch login
```

Uso de ramas

git branch: permite ver las ramas del proyecto.

```
ASUS@DESKTOP-5R9JJ5G MINGW64 ~/Desktop/Exposicion (main)
$ git branch
  login
* main
```

git checkout <nombre_rama>: permite cambiar de rama.

```
ASUS@DESKTOP-5R9JJ5G MINGW64 ~/Desktop/Exposicion (main)
$ git checkout login
Switched to branch 'login'

ASUS@DESKTOP-5R9JJ5G MINGW64 ~/Desktop/Exposicion (login)
$ █
```

Unión de ramas

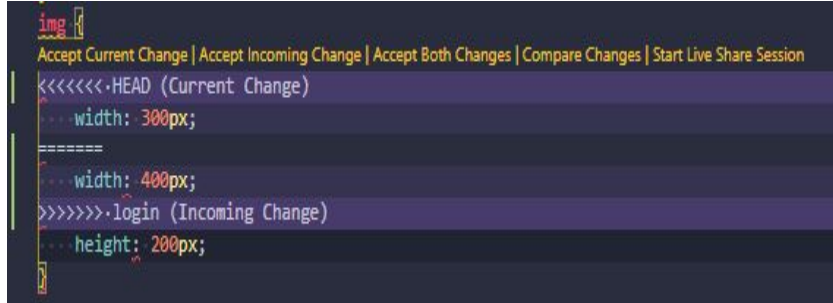
git merge <rama_secundaria>: para realizar un merge o unión de rama debemos posicionarnos en la rama principal (main o master).

```
ASUS@DESKTOP-5R9JJ5G MINGW64 ~/Desktop/Exposicion (main)
$ git merge login
Updating 4093e55..79db224
Fast-forward
 css/styles.css | 15 ++++++
 img/cirugia.jpg | Bin 7707 -> 0 bytes
 index.html      | 14 ++++++-----
 3 files changed, 23 insertions(+), 6 deletions(-)
 delete mode 100644 img/cirugia.jpg
```

Conflictos en el código

Se generan cuando hacemos un push o un merge y otro ha trabajado sobre los mismos archivos que uno.

Para resolver estos conflictos debemos comunicarnos con la persona que hizo la modificación y llegar a un acuerdo sobre qué código se dejará en el archivo.



```
img
Accept Current Change | Accept Incoming Change | Accept Both Changes | Compare Changes | Start Live Share Session
<<<<<< HEAD (Current Change)
  width: 300px;
=====
  width: 400px;
>>>>>> login (Incoming Change)
  height: 200px;
```

- Aceptar cambio actual
- Aceptar cambio entrante
- Aceptar ambos cambios
- Comparar cambios

¡Gracias!