

INTRODUCCIÓN A GIT

TABLA DE CONTENIDO

1. OVERVIEW DE GIT/GITHUB.....	3
2. COMENZANDO CON GITHUB.....	6
3. CREAR Y MEZCLAR RAMAS EN GITHUB.....	14
4. PRE-REQUISITOS PARA LA INTERFAZ DESDE LÍNEA DE COMANDOS.....	22
5. CONFIGURANDO SSH PARA ACCEDER AL REPOSITORIO.....	24
6. CREANDO UN REPOSITORIO EN GITHUB.....	27
7. RAMIFICACIONES Y FUSIONES VÍA LÍNEA DE COMANDOS.....	37
8. CONTRIBUYENDO A UN PROYECTO.....	41
9. RECUPERAR VERSIONES ANTERIORES DE ARCHIVOS.....	44

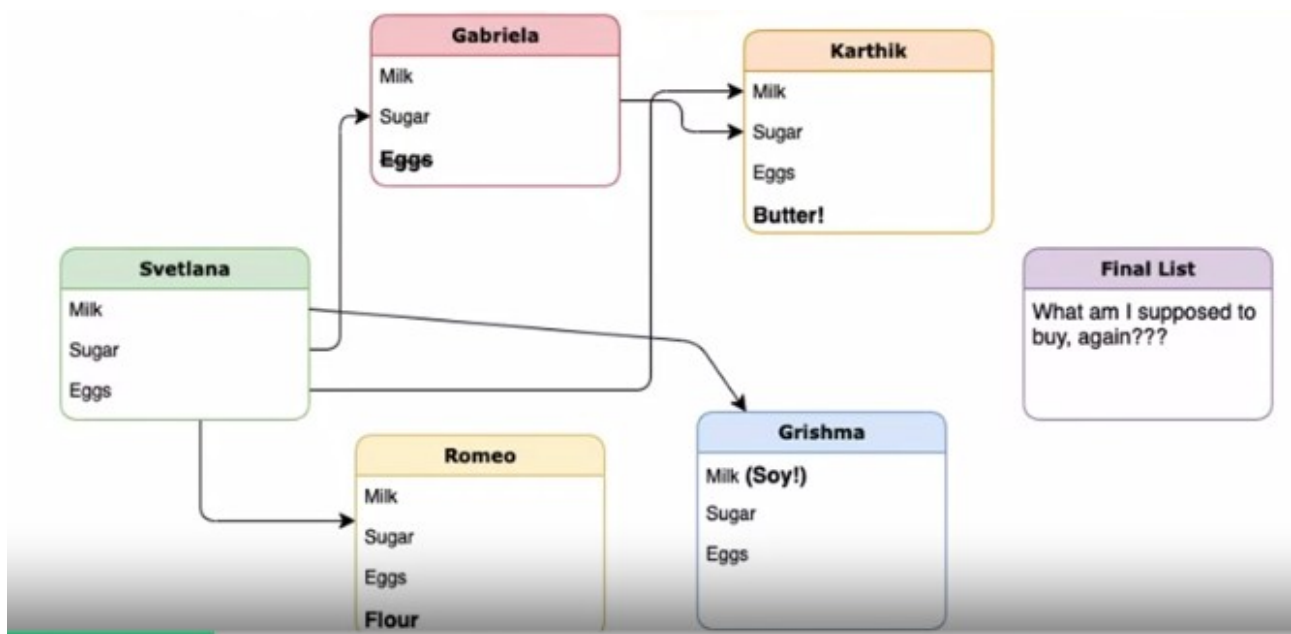
1. OVERVIEW DE GIT/GITHUB

No es posible hablar acerca de Git y GitHub sin un entendimiento básico acerca de qué es el control de versiones.

Un **sistema de control de versiones** permite seguir un rastro de los cambios en los documentos. Esto hace que sea más fácil:

- Recuperar versiones viejas de un documento si se comete un error.
- Trabajar en colaboración con otros.

Aquí un ejemplo de cómo funciona un control de versiones: digamos que tienes una lista de compras y quieres que tus compañeros te confirmen las cosas que necesitan junto con ítems adicionales. Sin el control de versiones, tienes un gran desastre que limpiar antes de ir de compras. Con el control de versiones, sabes EXACTAMENTE lo que necesitas luego de que cada uno haya contribuido con sus ideas.



Working with Version Control

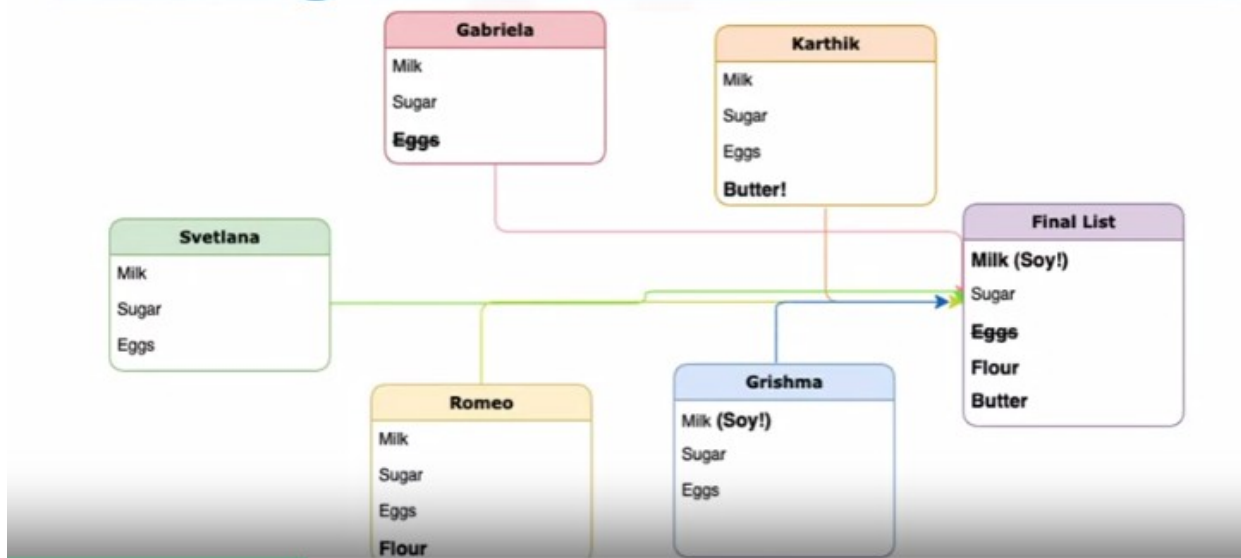


Figura 1. Explicando el control de versiones con un ejemplo de la vida real.

GIT es:

- Software free y open source, distribuido bajo la licencia general pública GNU.
- Un **sistema de control de versiones distribuido**.
 - Lo que significa que un usuario en cualquier parte del mundo puede tener una copia de su proyecto en su computadora personal; y cuando realiza cambios, pueden sincronizar su versión con un servidor remoto para compartirla contigo.

Git no es el único sistema de control de versiones, pero su aspecto distribuido es una de las razones por las que es uno de los más utilizados.

Los sistemas de control de versiones se usan mucho para todo aquello que involucre código, pero también pueden usarse para el control de versiones de imágenes, documentos, etc.

Puede usarse Git sin una interfaz web desde la línea de comandos.

GitHub es uno de los servicios de hosting web para repositorios Git. Otros son GitLab y BitBucket.

Hay algunos términos básicos que deben conocerse antes de empezar:

- **Protocolo SSH**
 - Es un método para lograr un inicio de sesión seguro de una computadora a otra.
- **Repositorio**

- Contiene las carpetas del proyecto que están configuradas para el control de versiones.
- **Fork**
 - Es una copia del repositorio.
- **Pull Request**
 - Es la forma en la que se solicita que alguien revise y apruebe los cambios antes de que se vuelvan finales.
- **Directorio de trabajo (working directory)**
 - Contiene los archivos y subdirectorios en su computadora que están asociados con un repositorio Git.

Hay algunos comandos básicos que deben conocerse:

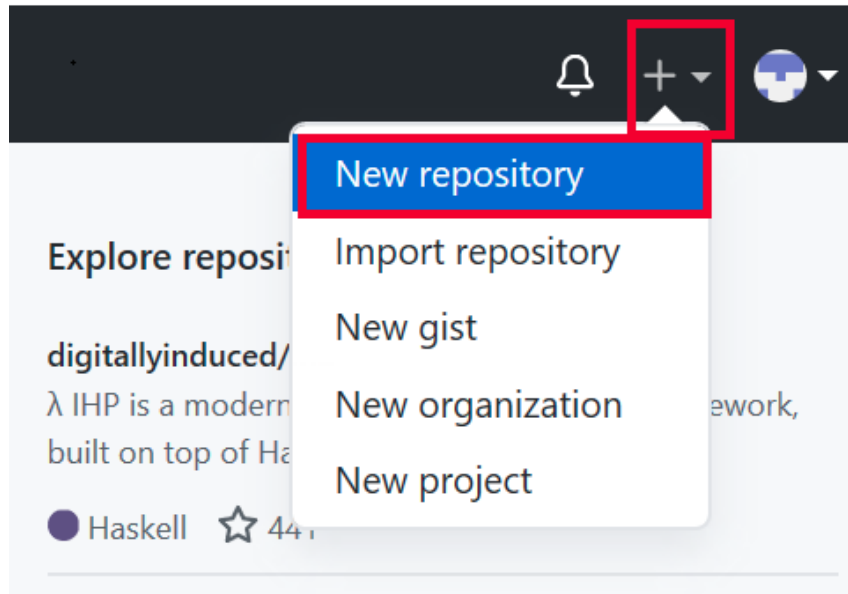
- **git init**
 - Cuando se comienza con un nuevo repositorio, sólo se necesita crearlo una vez: ya sea localmente y haciendo luego un push a GitHub o clonando un repositorio existente usando el comando “git init”.
- **git add**
 - Mueve los cambios del working directory al staging area (área de ensayo).
- **git status**
 - Permite ver el estado de su working directory y la instantánea preparada de sus cambios.
- **git commit**
 - Toma su instantánea preparada de cambios y la confirma (commit) con el proyecto.
- **git reset**
 - deshace los cambios que realizó en el working directory.
- **git log**
 - Habilita navegar a través de cambios previos en el proyecto.
- **git branch**
 - Permite crear un ambiente aislado dentro de su repositorio para realizar cambios.
- **git checkout**
 - Permite ver y cambiar ramas existentes.
- **git merge**
 - Permite volver a poner todo junto otra vez.

2. COMENZANDO CON GITHUB

Primero creamos una cuenta en <https://github.com/join>.

Ejercicio: Agregar un proyecto/repo

En el símbolo de + hacemos clic en “New Repository”:



Le damos un nombre al repositorio y lo inicializamos con un archivo README.me vacío:

Create a new repository

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

Owner *

 Malika-s ▾

Repository name *

/ testrepo ✓

Great repository names are short and memorable. Need inspiration? How about [urban-octo-waffle?](#)

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.

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



Initialize this repository with a README

This will let you immediately clone the repository to your computer.

Add .gitignore: None ▾

Add a license: None ▾

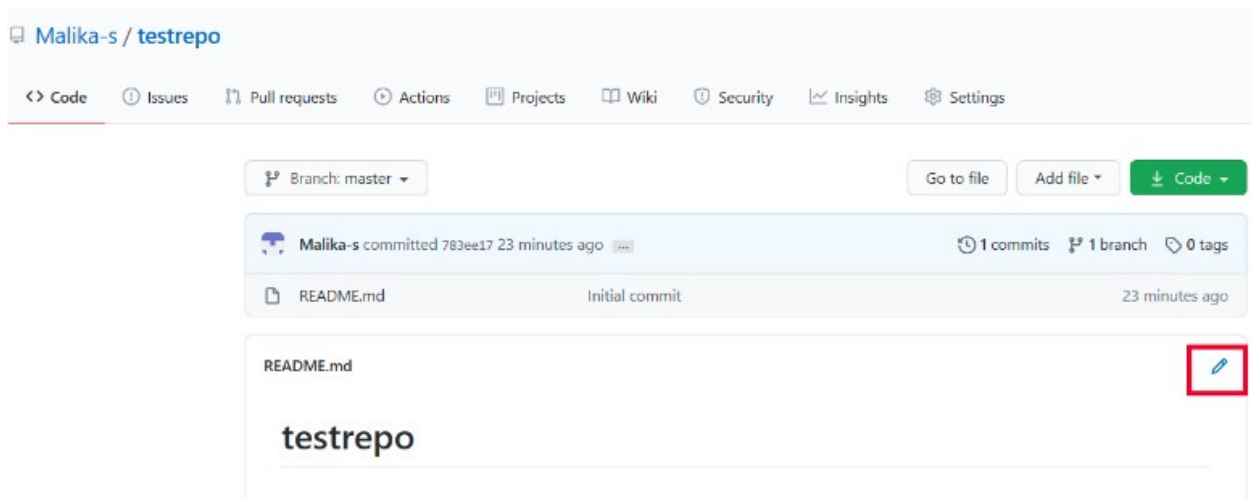


Create repository

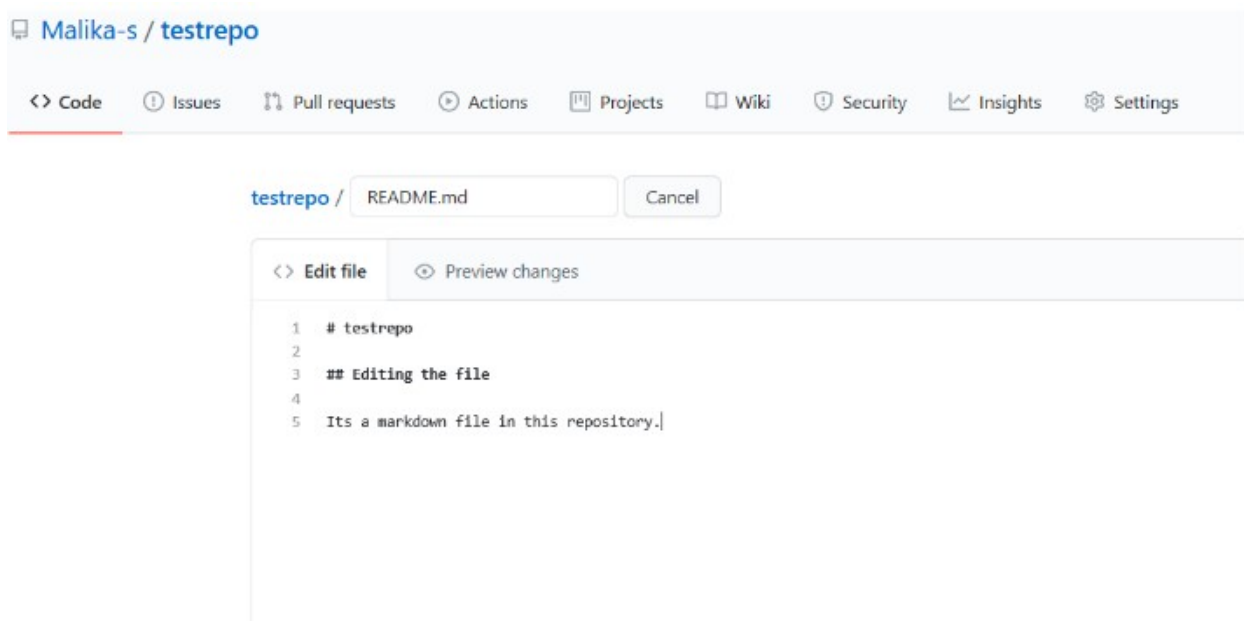
Hacemos clic en "Create repository". Ahora seremos redireccionados al repositorio recientemente creado.

Ejercicio: Crear/editar un archivo

Una vez que el repositorio es creado, el archivo raíz es listado por defecto y contiene un único archivo: README.me. Para editarlo se debe hacer clic en el lápiz:



Luego le agregamos texto al archivo:



Bajamos la página y hacemos clic en “Commit Changes”.



Commit changes

Update README.md

Add an optional extended description...

- ☒ Commit directly to the `master` branch.
- ☐ Create a new branch for this commit and start a pull request. [Learn more about pull requests.](#)

Commit changes

Cancel

Ejercicio: crear un nuevo archivo

Haga clic en el nombre del repositorio para volver a la rama master:

Malika-s / testrepo

<> Code ⓘ Issues 🔗 Pull requests ▶ Actions 📁 Projects 📖 Wiki ⓘ Security 📊 Insights ⚙ Settings

Branch: master ▼

Go to file

Add file ▼

Code ▼



Malika-s committed 3861fb4 4 minutes ago

2 commits 1 branch 0 tags



README.md

Update README.md

4 minutes ago

README.md



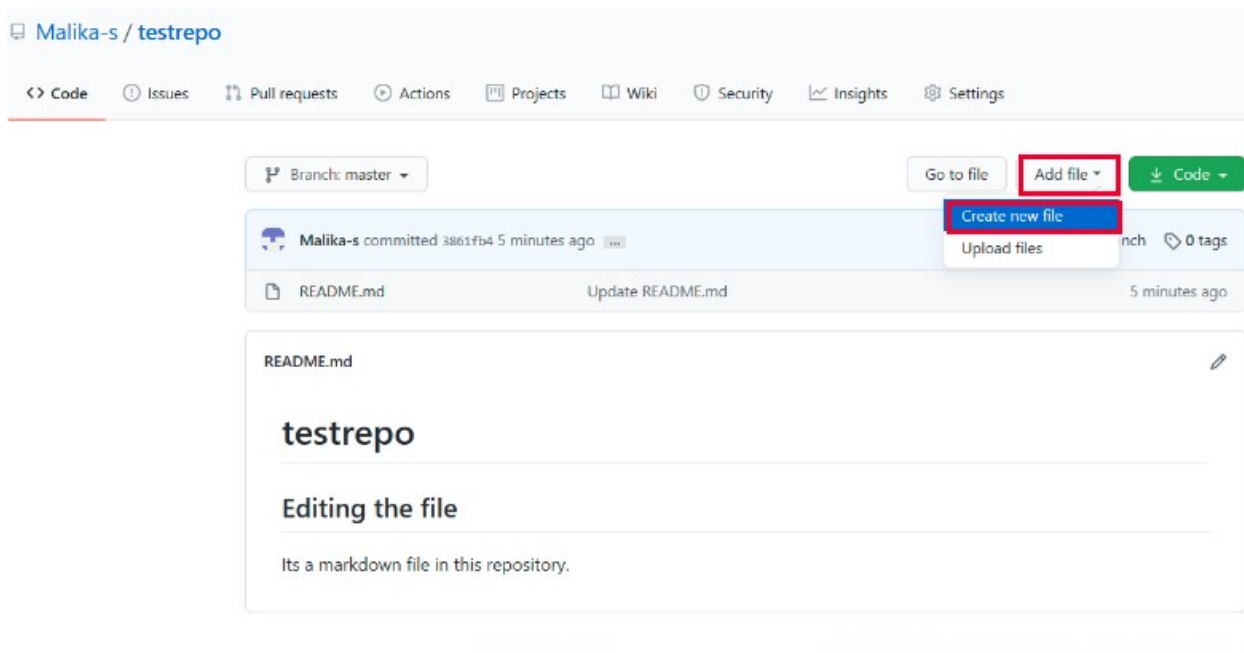
testrepo

Editing the file

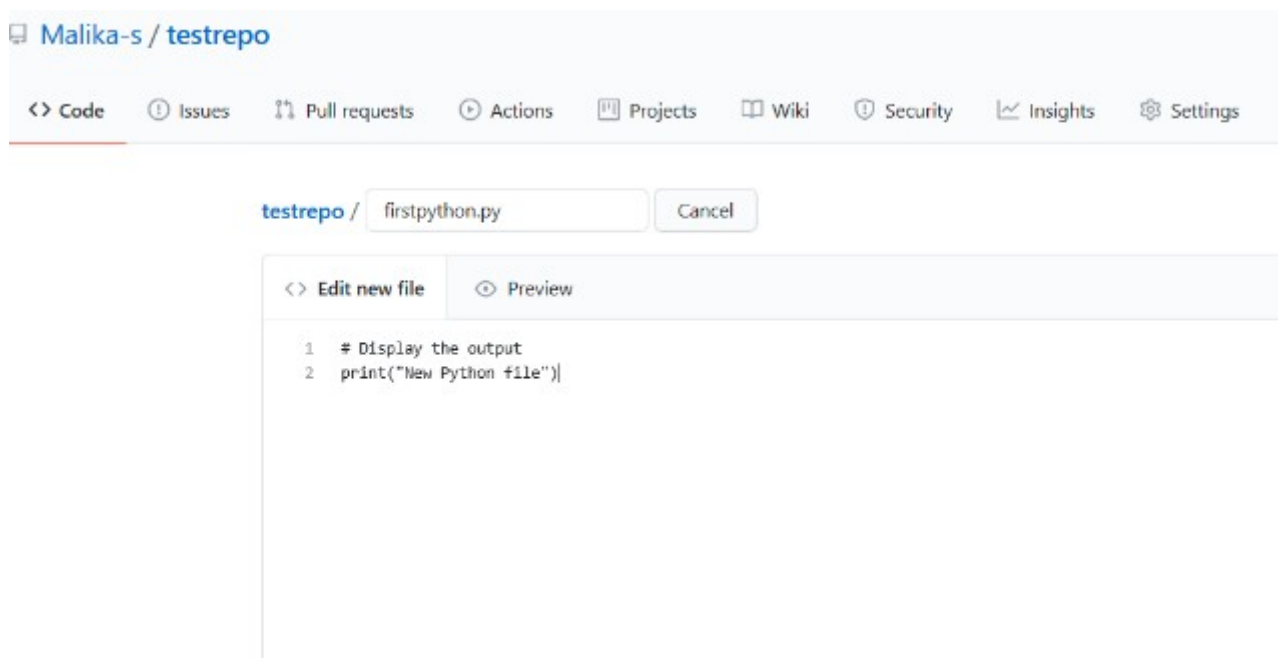
Its a markdown file in this repository.

malikarepo.png (1730 x 666)

Haga clic en “Add File” y seleccione “Create New” para crear un archivo en el repositorio:



Proporcione el nombre del archivo y la extensión. Por ejemplo: “firstpython.py” y agregue las líneas.



Baje en la página. Agregue una descripción si así lo desea y haga clic en “Commit new file”.

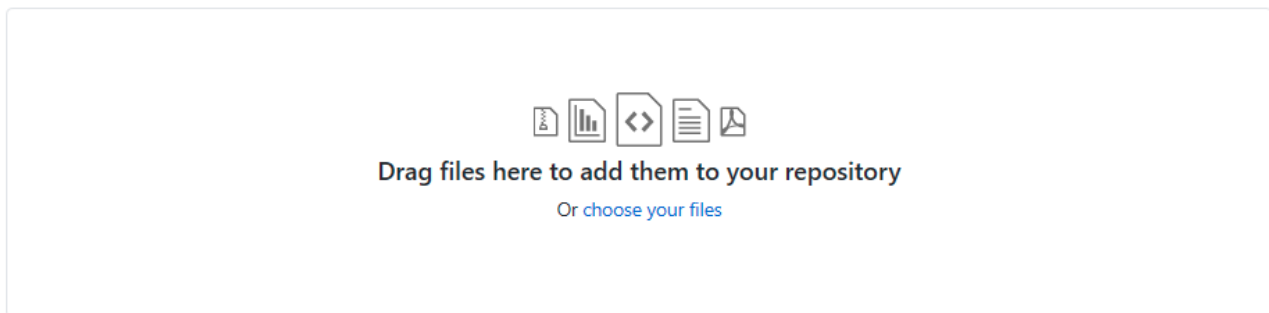
Su archivo fue agregado al repositorio.

Ejercicio: Suba un archivo y confírmelo

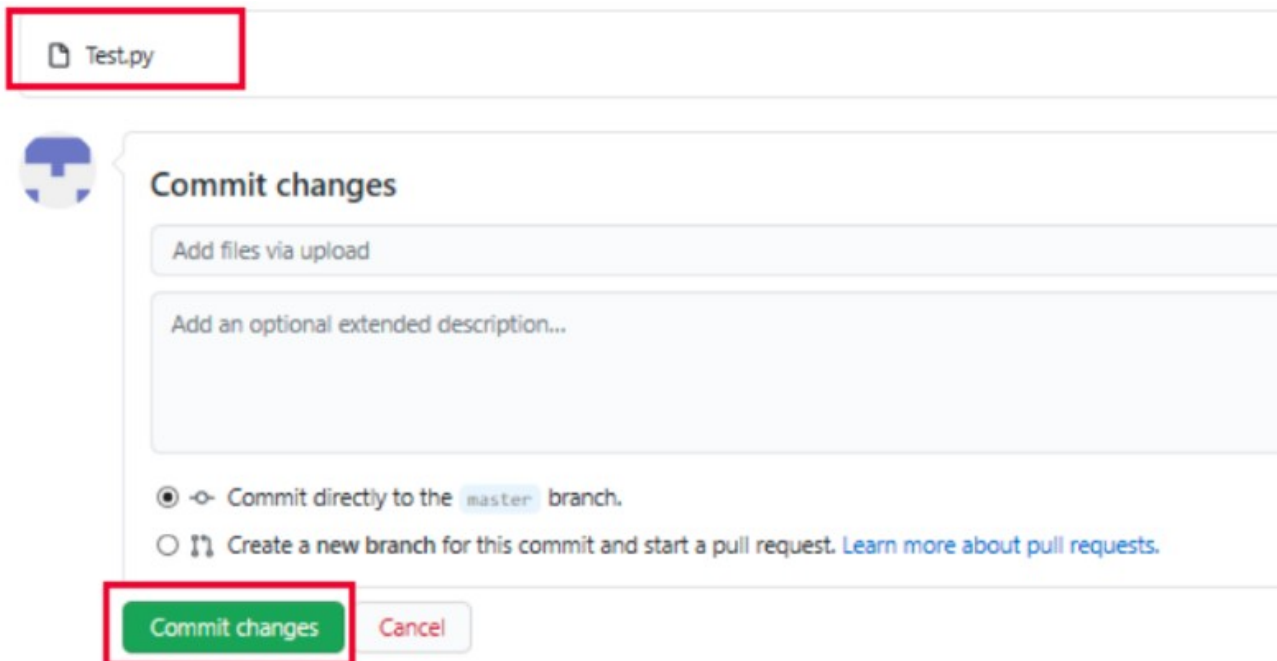
Haga clic en “Add file” y seleccione “Upload files” para subir un archivo al repositorio desde su máquina local.

Seleccione sus archivos.

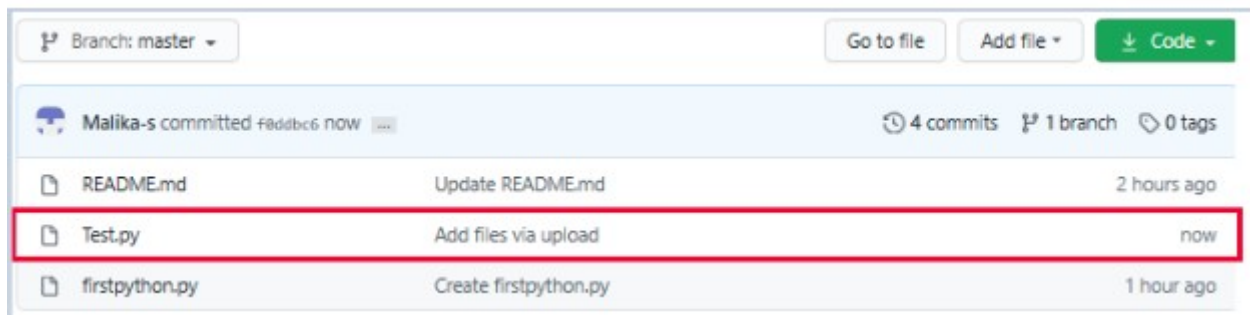
testrepo /



Cuando se terminen de subir, haga clic en “Commit changes”.



Su archivo quedó en el repositorio.



Branch: master

Go to file Add file Code

Malika-s committed f8ddb66 now

4 commits 1 branch 0 tags

README.md	Update README.md	2 hours ago
Test.py	Add files via upload	now
firstpython.py	Create firstpython.py	1 hour ago

3. CREAR Y MEZCLAR RAMAS EN GITHUB

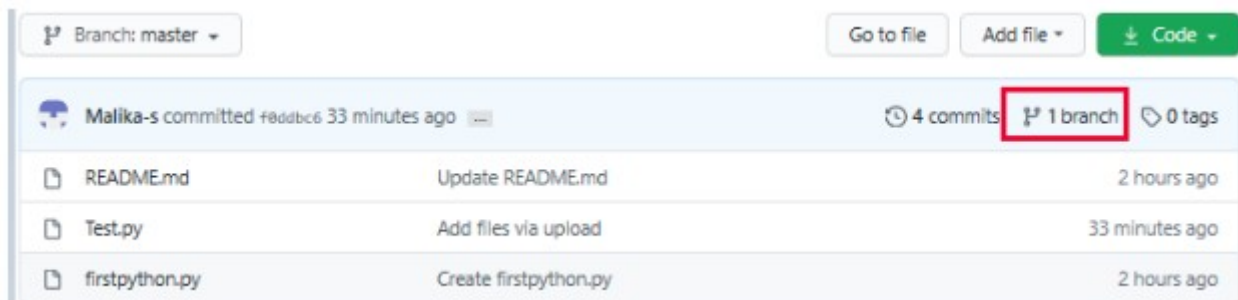
Haremos lo siguiente:

- Crear una rama.
- Confirmar cambios en la rama hija.
- Abrir un pull request (PR)
- Fusionar (merge) la PR en la rama master.

Ejercicio: Crear una rama

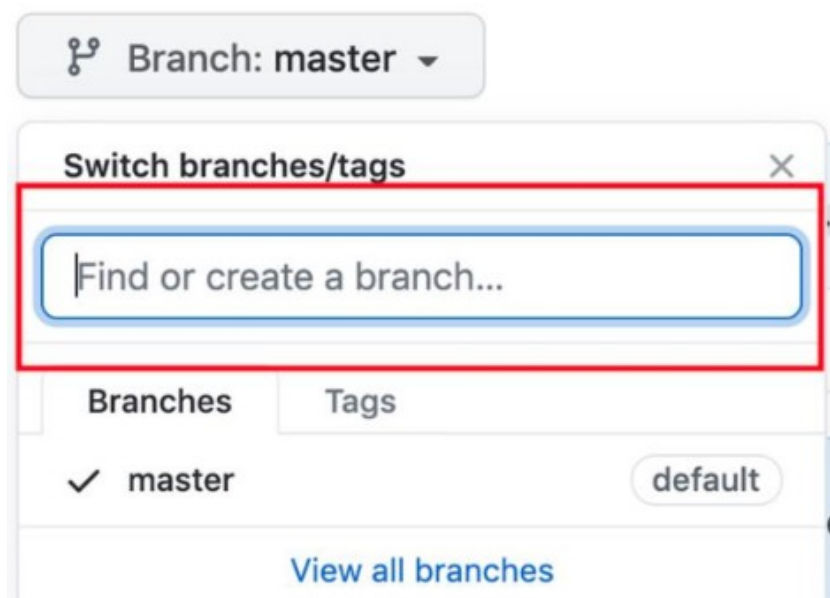
Puede crear o borrar ramas directamente en GitHub.

Paso 1. Actualmente, hay una rama, como se muestra abajo.

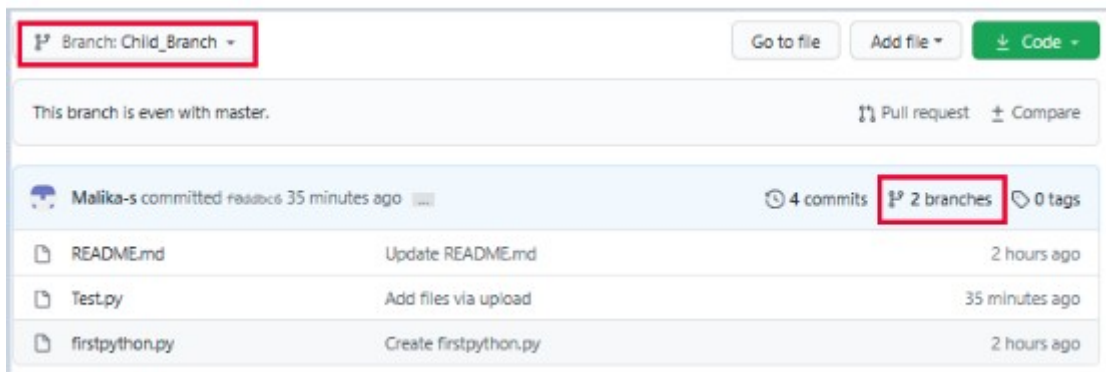


Paso 2. En GitHub, navegar a la página principal del repo.

Paso 3. Hacer clic en el selector de ramas. Ingres el nombre de su rama y presione ENTER.



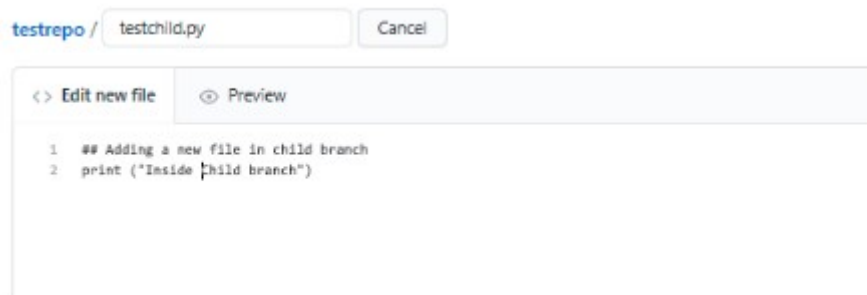
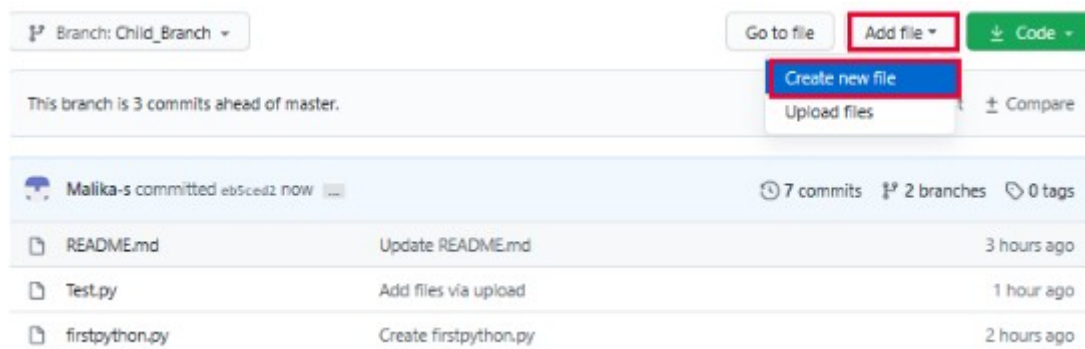
Paso 4. Observe que su repo tiene 2 ramas: Master y Child:Branch.



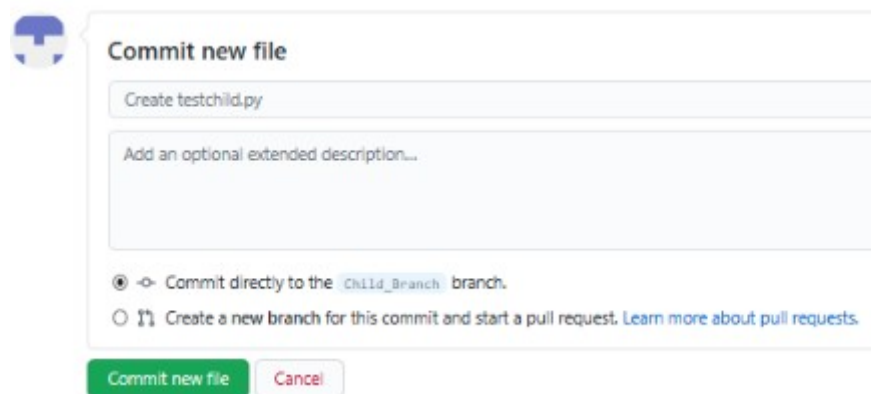
Lo que sea que haya en la rama master es copiado a la rama hija (child branch). Pero cuando agregamos o editamos cualquier archivo esto NO se verá reflejado en la rama Master.

Ejercicio 2. Agregar un archivo en la rama hija.

Paso 1. Cree un nuevo archivo. Por ejemplo testchild.py.



Agregue una descripción (opcional) y de clic en “Commit new file”.

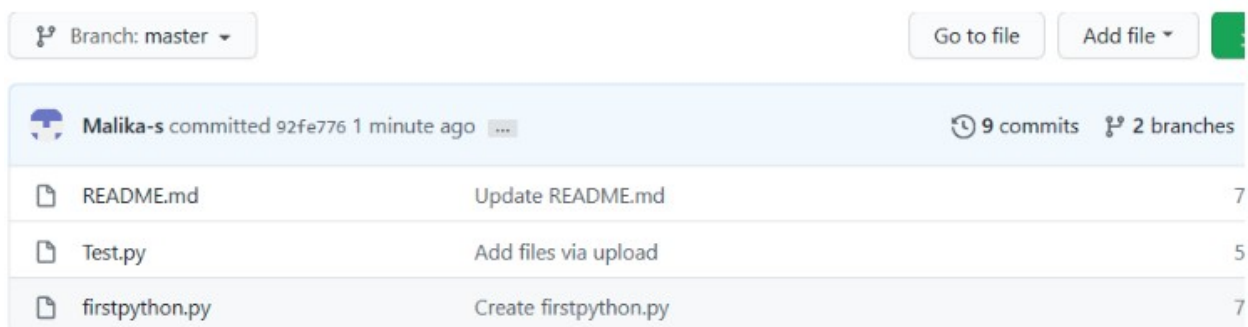


The screenshot shows the 'Commit new file' dialog box in GitHub. It has a title bar with the GitHub logo and the text 'Commit new file'. Below the title bar, there is a text input field containing 'Create testchild.py'. Underneath that is a larger text area with the placeholder text 'Add an optional extended description...'. At the bottom of the dialog, there are two radio buttons. The first is selected and is labeled 'Commit directly to the Child_Branch branch.' The second is labeled 'Create a new branch for this commit and start a pull request. Learn more about pull requests.' At the very bottom, there are two buttons: 'Commit new file' (green) and 'Cancel' (red).

El archivo ha sido agregado a la rama hija.

Ejercicio 3: Abrir un Pull Request

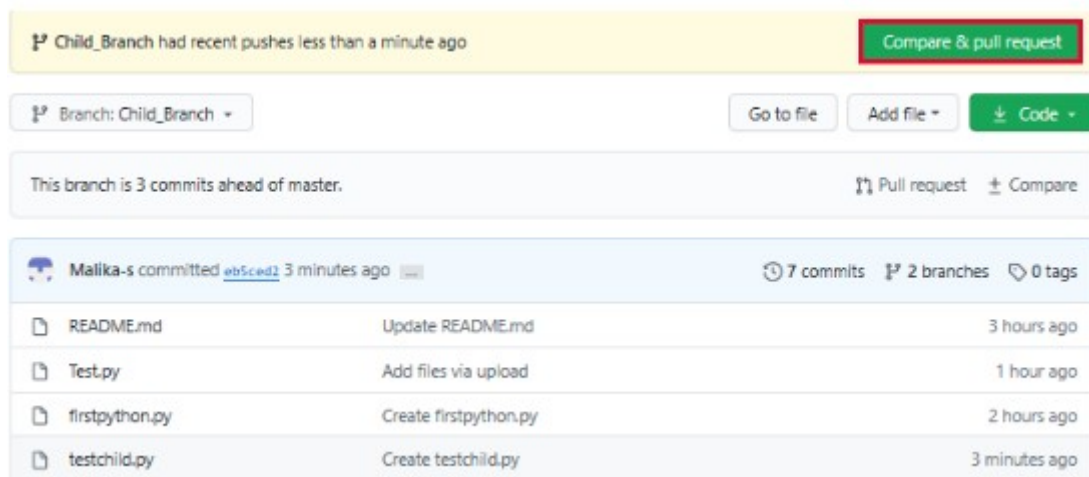
Puede chequear que en la rama master no hay ningún archivo testchild.py seleccionando la rama en el menú selector de ramas:



The screenshot shows the GitHub repository file list for the 'master' branch. At the top, there is a dropdown menu showing 'Branch: master'. To the right of the dropdown are buttons for 'Go to file' and 'Add file'. Below the dropdown, there is a summary bar showing 'Malika-s committed 92fe776 1 minute ago' and '9 commits 2 branches'. The main part of the screenshot is a table of files in the repository.

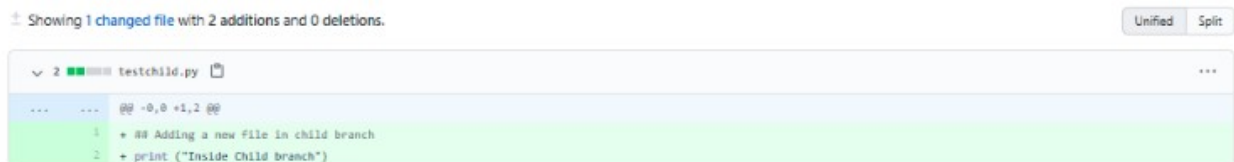
File	Commit Message	Size
README.md	Update README.md	7
Test.py	Add files via upload	5
firstpython.py	Create firstpython.py	7

También puede comparar el archivo con la opción que se muestra abajo “Compare & pull request”.



Paso 1. Baje en la página, obtendrá un 1 archivo cambiado:

Step 1: Scroll down the page, you will get 1 file changed



Paso 2. Suba y cree un pull request usando la opción “Create Pull Request”. En la imagen, en la parte sobresaltada, puede verse una flecha que indica que se está comparando y creando un pull request.

Puede agregar comentarios si así lo desea.

Open a pull request

Create a new pull request by comparing changes across two branches. If you need to, you can also [compare across forks](#).

base: master ← compare: Child_Branch ✓ Able to merge. These branches can be automatically merged.

Child branch

Write Preview

Want to change in the master branch

Attach files by dragging & dropping, selecting or pasting them.

Create pull request




Remember, contributions to this repository should follow our [GitHub Community Guidelines](#).

Ejercicio 4. Merge the Pull Request

Para aceptar el pull request haga clic en la pestaña Pull Requests para ver un resumen de los pull requests pendientes. Si estás de acuerdo con los cambios, haz clic en Merge Pull request para aceptar la pull request y realizar la fusión (merge).


Child branch #1

 Open Malika-s wants to merge 3 commits into `master` from `Child_Branch` 

 Conversation **0**  Commits **3**  Checks **0**  Files changed **1**



Malika-s commented 2 minutes ago

Owner  ...

Want to change in the master branch.



Malika-s added 3 commits 25 minutes ago



Create testchild

Verified 9767921



Delete testchild

Verified 48b4479



Create testchild.py

Verified eb5ced2

Add more commits by pushing to the `Child_Branch` branch on Malika-s/testrepo.



Continuous integration has not been set up

[GitHub Actions](#) and [several other apps](#) can be used to automatically catch bugs and enforce style.



This branch has no conflicts with the base branch

Merging can be performed automatically.

Merge pull request




You can also open this in [GitHub Desktop](#) or view [command line instructions](#).

Una vez que se haga clic en Merge Pull request, verá un botón de confirmación.


Child branch #1




[Open](#) Malika-s wants to merge 3 commits into `master` from `Child_Branch`

[Conversation](#) 0 [Commits](#) 3 [Checks](#) 0 [Files changed](#) 1


 Malika-s commented 4 minutes ago Owner  

Want to change in the master branch.

 Malika-s added 3 commits 27 minutes ago

-  Create testchild Verified 9767921
-  Delete testchild Verified 48b4479
-  Create testchild.py Verified eb5ced2


Add more commits by pushing to the `Child_Branch` branch on Malika-s/testrepo.

 Merge pull request #1 from Malika-s/Child_Branch

Child branch

[Confirm merge](#) [Cancel](#)

La fusión ha sido exitosa.


 Pull request successfully merged and closed





You're all set—the `child_branch` branch can be safely deleted.

[Delete branch](#)

Ahora, la rama hija se ha fusionado con la rama Master. Puede chequear que ahora la rama Master tiene el archivo testchild.py.

Branch: `master` [Go to file](#) [Add file](#) [Code](#)

 Malika-s committed 7a7c576 3 minutes ago 8 commits 2 branches 0 tags

 README.md	Update README.md	3 hours ago
 Test.py	Add files via upload	1 hour ago
 firstpython.py	Create firstpython.py	3 hours ago
 testchild.py	Create testchild.py	21 minutes ago

Hemos visto como crear una rama, editar y confirmar cambios, abrir un pull request y fusionar la solicitud.

4. PRE-REQUISITOS PARA LA INTERFAZ DESDE LÍNEA DE COMANDOS

En Linux, la instalación de git es simple:

sudo apt install git-all

Veamos ahora la generación de la clave SSH:

Una **clave SSH** es una credencial de acceso en el protocolo SSH. Su función es similar a la de los nombres de usuario y contraseña, pero las claves se usan principalmente para procesos automáticos.

1. Lanzar git bash

2. Escriba lo siguiente, reemplazando la dirección de correo por la de su cuenta de GitHub:

```
ssh-keygen -t rsa -b 4096 -C "<your email address>"
```

Esto va a generar una nueva clave SSH.

3. Se le pedirá que ingrese un directorio para guardar la clave. La ubicación por defecto es una carpeta .ssh en el directorio home.

En otras palabras, podrá localizar la clave en ~/.ssh/id_rsa.

4. Se le pedirá una frase de contraseña. Puede dejarla en blanco.

Ahora navegue al directorio .ssh con:

```
cd ~/.ssh
```

con ls puede listar todo el contenido del directorio .ssh. Deberá encontrar allí id_rsa id_rsa.pub.

id_rsa es la versión privada de la clave, e id_rsa.pub la pública.

5. Debe agregarse la clave SSH al ssh-agent, que está destinado a ayudar con el proceso de autenticación.

Para esto se debe iniciar primero el ssh-agent:

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

6. Agregue la clave al agente ejecutando lo siguiente en el terminal Git Bash:

```
ssh-add ~/.ssh/id_rsa
```

5. CONFIGURANDO SSH PARA ACCEDER AL REPOSITORIO

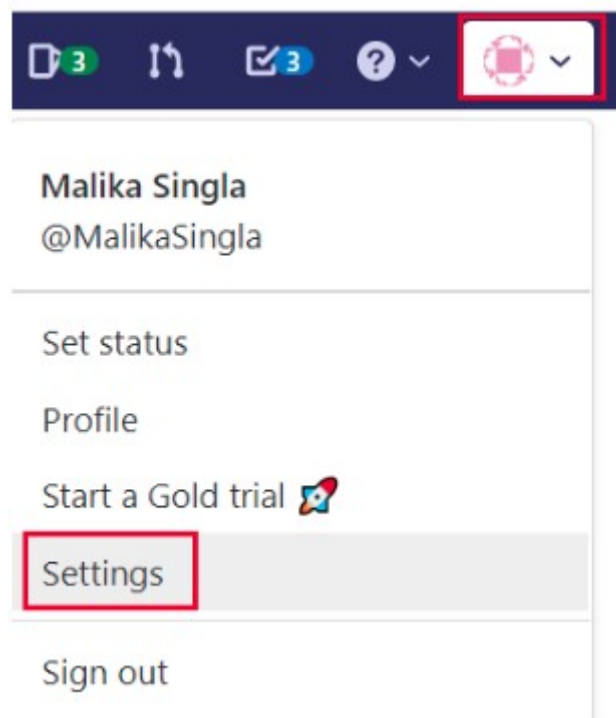
Abra GitBash para copiar la clave SSH generada previamente.

1. Copie el comando usando:

```
cat ~/.ssh/id_rsa.pub | clip
```

Si clip no funciona use `cat ~/.ssh/id_rsa.pub` en la línea de comandos y copie la salida.

2. Abra GitHub y vaya a Settings:



3. Seleccione “SSH and GPG keys”

Malika-s
Personal settings

Profile

Account

User security

Security log

Security & analysis

Emails

Notifications

Billing

SSH and GPG keys

Blocked users

Repositories

Organizations

Saved replies

Public profile

Name

Your name may appear around GitHub where you contribute or are mentioned. You can remove it at any time.

Public email

Select a verified email to display

You have set your email address to private. To toggle email privacy, go to [email settings](#) and uncheck "Keep my email address private."

Bio

Tell us a little bit about yourself

You can @mention other users and organizations to link to them.

URL

Twitter username

4. Haga clic para agregar una nueva clave SSH:

Malika-s
Personal settings

Profile

Account

User security

Security log

Security & analysis

Emails

Notifications

Billing

SSH and GPG keys

SSH keys

New SSH key

There are no SSH keys associated with your account.

Check out our guide to [generating SSH keys](#) or troubleshoot [common SSH Problems](#).


GPG keys

New GPG key

There are no GPG keys associated with your account.

Learn how to [generate a GPG key](#) and add it to your account.

5. Brinde un título y pegue la clave. La clave pegada debe tener su email al final.

 **Malika-s**
Personal settings

Profile

Account

User security

Security log

Security & analysis

Emails

Notifications

Billing

SSH and GPG keys

Blocked users

Personalization

SSH keys / Add new

Title

AddingKey

Key

AAAAAB3NzaC1yc2EAAAADAQABAAQACQC5b03ZdC9zRTA7ZRJ384mn5ZMTFDkWqbpqZjXTv2dlMv6xbsDkwzEkH8bYxEgUubj07K0uiXIZsL3atfx4E3Wm+B8zVnBxb897CbH/fCDpPXa7eSLT91NyZUEJLpt0F6BmWRqzqtV4kdyddNKG9R7c2btk3CQ3jGR57z/MX9/zXK3Mjk9erUke4CWKFlwR2GeXDOat86an97odssi6PIRjklI7m6Qs5odJUP1ljsrLjsAmlUsQPd6bpfm3EtgvKfhUTGm/jMOllws9MYASP5VOhte2s5nXAY/gIA3obe6ZPWuJpenSxRd1qZgeRhs9WHw0YpSrKrUf5u0tEy+QcB7lcShkrRqgBssSsrcb7/bTZZ/5YU0SUNMBDkW7/uSLw3Gah6jKBEX4RqjsjnfAQ+5d3N6WNMI2+jL7Fn18+ugguRY9CcfsG7qlqZD/SZfedol7k6gWnXKrGCZ2mPTTvfIKJTRGbfFtWAVAmuWoSI5kblI8fYbOs8Hfm+iD2GyA2RVQDQMhPEVx+4QyJG/wzPu0IHBEyNwiZMhm6JjrJwwLV3wkwKa9DGdwhbMsjSoW3W8nGDnNRFRFHGbeU5Bh+R3or0lvDS0RIBfDjVkcGHXak1VwMP3Czo4FwGs5C4zCigysjfyV3bZ9Icz6R/Nzt2WRGP9eGS8fEDKAzdzjQ==
Your email address

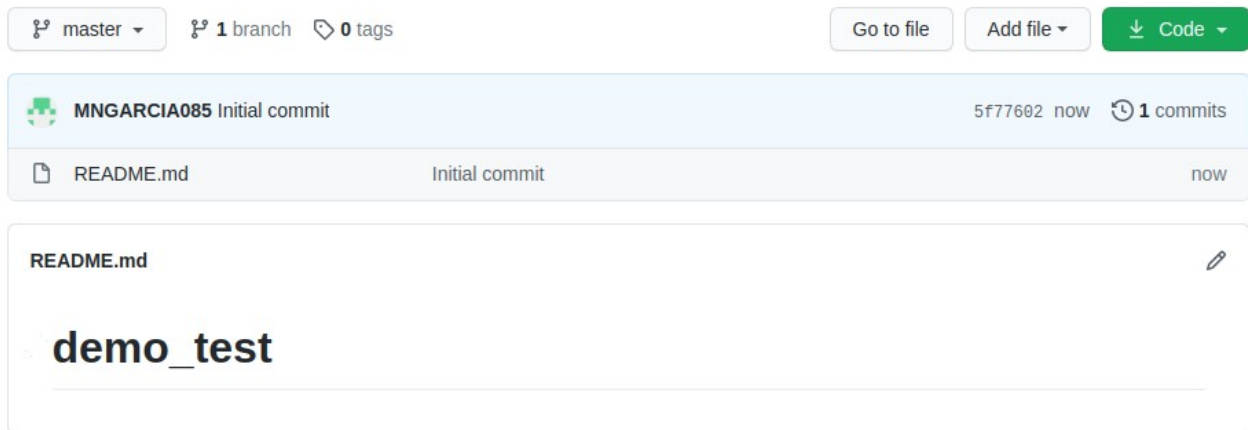
Add SSH key

Haga clic en “Add SSH Key”.

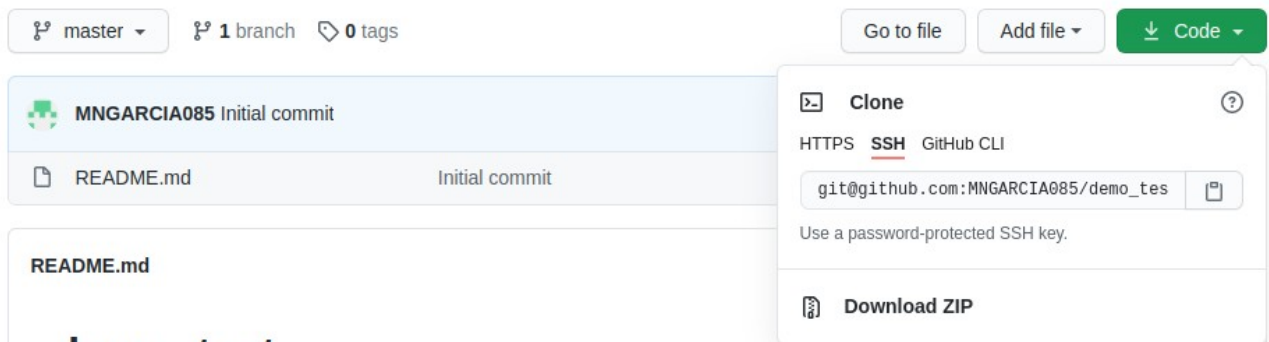
Ahora, la clave fue agregada a su cuenta.

6. CREANDO UN REPOSITORIO EN GITHUB

1. Creamos un nuevo repositorio llamado “demo_test” en GitHub. Una vez creado se verá así:



2. Para descargarlo tenemos varias opciones, entre ellas, copiarlo a un repositorio local usando SSH y HTTPS.



3. En este caso lo clonaremos usando el link SSH.

Vamos a la carpeta donde queremos clonar el proyecto y usamos el comando:

“git clone <SSHRepositoryLink>”

En este caso el proyecto lo clonaremos en: SSHRepositoryLink

El link es: [git@github.com:MNGARCIA085/demo_test.git](https://github.com/MNGARCIA085/demo_test)

```
marcos@a1:~/Escritorio/GIT$ git clone  
git@github.com:MNGARCIA085/demo_test.git
```

Clonando en 'demo_test'...

Warning: Permanently added the RSA host key for IP address '140.82.113.3' to the list of known hosts.

remote: Enumerating objects: 3, done.

remote: Counting objects: 100% (3/3), done.

remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0

Recibiendo objetos: 100% (3/3), listo.

Ahora, el proyecto fue copiado a su máquina local.

```
marcos@a1:~/Escritorio/GIT$ ls  
demo_test
```

Creamos un nuevo archivo:

```
gedit prueba.txt
```

```
marcos@a1:~/Escritorio/GIT/demo_test$ gedit prueba.txt  
marcos@a1:~/Escritorio/GIT/demo_test$ ls  
prueba.txt README.md
```

Lo agregamos al repo con "git add":

```
marcos@a1:~/Escritorio/GIT/demo_test$ git add prueba.txt
```

Chequeamos el status con "git status":

```
marcos@a1:~/Escritorio/GIT/demo_test$ git status
```

En la rama master

Tu rama está actualizada con 'origin/master'.

Cambios a ser confirmados:

(usa "git restore --staged <archivo>..." para sacar del área de stage)
nuevo archivo: prueba.txt

Confirmamos los cambios con "git commit -m <mensaje>"

```
marcos@a1:~/Escritorio/GIT/demo_test$ git commit -m "mensaje"
```

*** Por favor cuéntame quien eres.

Corre

```
git config --global user.email "you@example.com"  
git config --global user.name "Tu Nombre"
```

para configurar la identidad por defecto de tu cuenta.
Omite --global para configurar tu identidad solo en este repositorio.

fatal: no es posible auto-detectar la dirección de correo (se obtuvo 'marcos@a1.(none)')

```
marcos@a1:~/Escritorio/GIT/demo_test$ git config user.email  
"mngarcia085@gmail.com"
```

```
marcos@a1:~/Escritorio/GIT/demo_test$ git commit -m "mensaje"  
[master 6cb7906] mensaje  
1 file changed, 1 insertion(+)  
create mode 100644 prueba.txt
```

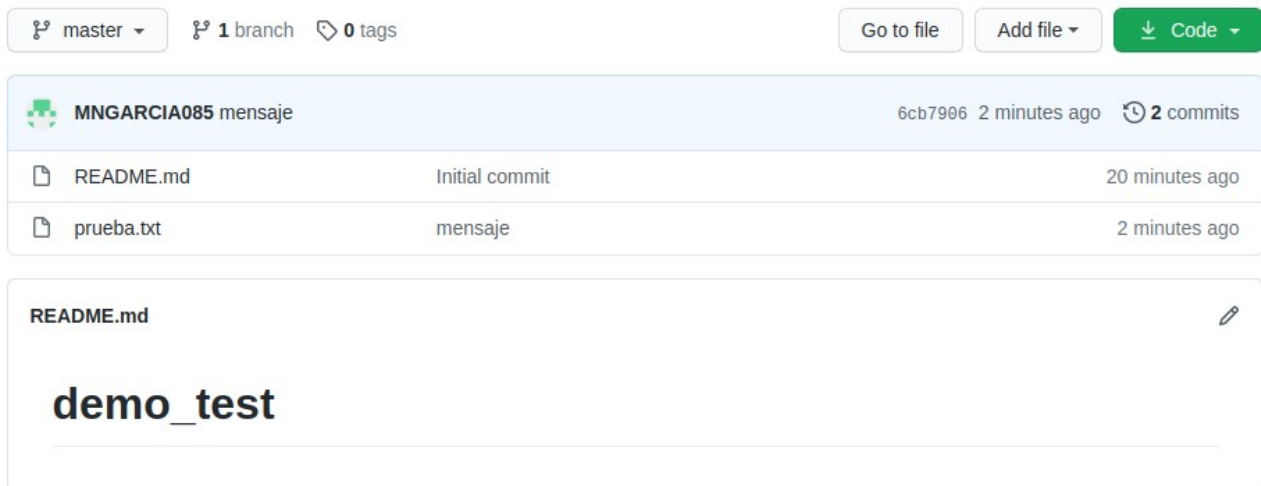
Subo los cambios al repo con "git push"

```
marcos@a1:~/Escritorio/GIT/demo_test$ git push
```

Enumerando objetos: 4, listo.
Contando objetos: 100% (4/4), listo.
Compresión delta usando hasta 4 hilos
Comprimiendo objetos: 100% (2/2), listo.
Escribiendo objetos: 100% (3/3), 277 bytes | 277.00 KiB/s, listo.

Total 3 (delta 0), reusado 0 (delta 0)
To github.com:MNGARCIA085/demo_test.git
5f77602..6cb7906 master -> master

Ahora pueden verse los cambios en el repo:



The screenshot shows the GitHub interface for the repository MNGARCIA085/demo_test.git. At the top, there are buttons for 'Go to file', 'Add file', and 'Code'. Below this, a commit history table is displayed:

File	Commit Message	Time
README.md	Initial commit	20 minutes ago
prueba.txt	mensaje	2 minutes ago

Below the table, the content of the README.md file is shown, which includes the text 'demo_test'.

RESUMEN

1. Creamos un nuevo repositorio en GitHub
2. Para descargarlo tenemos varias opciones, entre ellas, copiarlo a un repositorio local usando SSH y HTTPS.
3. Copie el link SSH.
4. Abra si terminal y muévase al directorio donde desea trabajar. En mi caso /home/Escritorio/GIT
5. Clone el repositorio a su sistema local:

git clone <repolink>

6. Ahora, la carpeta fue copiada a su máquina local, dentro de *home...GIT*.

7. Haga `cd demo`

8. Para listar los archivos puede usar `ls`, ver el contenido `cat README.md` y crear un nuevo archivo `gedit test.txt`.

9. Agregue un archivo al repo:

```
git add test.txt
```

Chequee el status: `git status`

10. Confirme los cambios usando:


```
git commit -m "aquí un mensaje"
```

11. Haga un push del archivo ejecutando:

```
git push
```


Ahora puede ver los cambios en su repo.


Ahora crearemos un nuevo repositorio “demo_test1” sin el archivo README.md.

Owner *  mskill / Repository name * demo_test1 ✓

Great repository name demo_test1 is available. Need inspiration? How about [stunning-octo-succotash?](#)


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.

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

☐ **Initialize this repository with a README**
This will let you immediately clone the repository to your computer.

Add .gitignore: **None** ▼ | Add a license: **None** ▼ 

Create repository

Veremos la siguiente pantalla:

Quick setup — if you've done this kind of thing before

or `git@github.com:MNGARCIA085/demo_test1.git`

Get started by [creating a new file](#) or [uploading an existing file](#). We recommend every repository include a [README](#), [LICENSE](#), and [.gitignore](#).

...or create a new repository on the command line

```
echo "# demo_test1" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M master
git remote add origin git@github.com:MNGARCIA085/demo_test1.git
git push -u origin master
```

...or push an existing repository from the command line

```
git remote add origin git@github.com:MNGARCIA085/demo_test1.git
git branch -M master
git push -u origin master
```

...or import code from another repository

You can initialize this repository with code from a Subversion, Mercurial, or TFS project.

Copiamos el link como antes:

[git@github.com:MNGARCIA085/demo_test1.git](https://github.com/MNGARCIA085/demo_test1.git)

Creamos en nuestra máquina local un directorio con el nombre del proyecto y nos movemos a él:

```
marcos@a1:~/Escritorio/GIT$ mkdir demo_test1
marcos@a1:~/Escritorio/GIT$ cd demo_test1
marcos@a1:~/Escritorio/GIT/demo_test1$
```

Creamos el archivo readme:

```
echo "#demo1" >> README.md
```

Inicializamos el directorio:

git init

Agregamos el archivo README.md

git add README.md

Chequeamos el status:

git status

Confirmamos los cambios

git commit -m "primer commit"

Agregamos el origen dónde haremos push del archivo. Este es el link SSH copiado previamente.

git remote add origin [git@github.com:MNGARCIA085/demo_test1.git](ssh://git@github.com:MNGARCIA085/demo_test1.git)

Hacemos push del archivo:

git push -u origin master

```
marcos@a1:~/Escritorio/GIT/demo_test1$ echo "#demo1" >> README.md
```

```
marcos@a1:~/Escritorio/GIT/demo_test1$ git init
```

```
Inicializado repositorio Git vacío en /home/marcos/Escritorio/GIT/demo_test1/.git/
```

```
marcos@a1:~/Escritorio/GIT/demo_test1$ git add README.md
```

```
marcos@a1:~/Escritorio/GIT/demo_test1$ git status
```

En la rama master

No hay commits todavía

Cambios a ser confirmados:

(usa "git rm --cached <archivo>..." para sacar del área de stage)
nuevo archivo: README.md

```
marcos@a1:~/Escritorio/GIT/demo_test1$ git commit -m "primer commit"
```

```
[master (commit-raíz) 5d4752c] primer commit  
1 file changed, 1 insertion(+)  
create mode 100644 README.md
```

```
marcos@a1:~/Escritorio/GIT/demo_test1$ git remote add origin  
git@github.com:MNGARCIA085/demo_test1.git
```

```
+marcos@a1:~/Escritorio/GIT/demo_test1$ git push -u origin master
```

Warning: Permanently added the RSA host key for IP address '140.82.113.4' to the list of known hosts.

Enumerando objetos: 3, listo.

Contando objetos: 100% (3/3), listo.

Escribiendo objetos: 100% (3/3), 222 bytes | 222.00 KiB/s, listo.

Total 3 (delta 0), reusado 0 (delta 0)

To github.com:MNGARCIA085/demo_test1.git

* [new branch] master -> master

Rama 'master' configurada para hacer seguimiento a la rama remota 'master' de 'origin'.

Ahora el archivo README fue creado en nuestro repositorio.

master


1 branch

0 tags


Go to file

Add file

Code

 MNGARCIA085 primer commit


5d4752c 1 minute ago 1 commits

 README.md

primer commit

1 minute ago

README.md



```
"#demo1"
```

7. RAMIFICACIONES Y FUSIONES VÍA LÍNEA DE COMANDOS

1. Creamos un nuevo repositorio en GitHub. Se llamará RAMAS.

2. Clonamos el repo en nuestra máquina local:

```
git clone git@github.com:MNGARCIA085/RAMAS.git
```

3. Agregamos un nuevo archivo “newfile.txt” en GitHub.

Al agregar el archivo remotamente, éste no se encontrará en el directorio local, esto puede chequearse usando ls.

```
marcos@marcos-HP-Laptop-15-bs0xx:~/Escritorio/GIT/RAMAS$ ls  
README.md
```

4. Para hacer un pull del archivo que fue agregado en el repo remoto al repo local usamos el comando “git pull”

```
marcos@marcos-HP-Laptop-15-bs0xx:~/Escritorio/GIT/RAMAS$ git pull
```

```
remote: Enumerating objects: 4, done.  
remote: Counting objects: 100% (4/4), done.  
remote: Compressing objects: 100% (2/2), done.  
Desempaquetando objetos: 100% (3/3), 669 bytes | 334.00 KiB/s, listo.  
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0  
Desde github.com:MNGARCIA085/RAMAS  
  0ef47c2..c01b315 main    -> origin/main  
Actualizando 0ef47c2..c01b315  
Fast-forward  
 newfile.txt | 1 +  
 1 file changed, 1 insertion(+)  
 create mode 100644 newfile.txt
```

Luego de hacer el pull, tenemos ahora 2 archivos en el repo local:

```
marcos@marcos-HP-Laptop-15-bs0xx:~/Escritorio/GIT/RAMAS$ ls  
newfile.txt README.md
```

5. Para agregar una nueva rama en la rama master: **git branch branchname**

Para cambiar de rama: **git checkout branchname**

Agregar un archivo en la rama: **echo “#content”>> filename.txt**

Luego agregue y haga un push del archivo. Para crear la rama remotamente debemos usar “**git push –set-upstream origin branchname**”.

git branch branchname

git checkout branchname

echo "#stuff on branch" >> stuffonbranch.txt

git add stuffonbranch.txt

git commit -m "agregar a la rama"

git push

git push --set-upstream origin branchname

```
marcos@marcos-HP-Laptop-15-bs0xx:~/Escritorio/GIT/RAMAS$ git branch branchname
marcos@marcos-HP-Laptop-15-bs0xx:~/Escritorio/GIT/RAMAS$ git checkout
branchname
```

Cambiado a rama 'branchname'

```
marcos@marcos-HP-Laptop-15-bs0xx:~/Escritorio/GIT/RAMAS$ echo "#stuff on branch"
>> stuffonbranch.txt
marcos@marcos-HP-Laptop-15-bs0xx:~/Escritorio/GIT/RAMAS$ git add
stuffonbranch.txt
marcos@marcos-HP-Laptop-15-bs0xx:~/Escritorio/GIT/RAMAS$ git commit -m "add"
```

```
[branchname 0ed5ddd] add
1 file changed, 1 insertion(+)
create mode 100644 stuffonbranch.txt
```

```
marcos@marcos-HP-Laptop-15-bs0xx:~/Escritorio/GIT/RAMAS$ git push
```

fatal: La rama actual branchname no tiene una rama upstream.
Para realizar un push de la rama actual y configurar el remoto como upstream, use

git push --set-upstream origin branchname

```
marcos@marcos-HP-Laptop-15-bs0xx:~/Escritorio/GIT/RAMAS$ git push --set-upstream
origin branchname
```

Warning: Permanently added the RSA host key for IP address '140.82.112.4' to the list of known hosts.

Enumerando objetos: 4, listo.

Contando objetos: 100% (4/4), listo.

Compresión delta usando hasta 4 hilos

Comprimiendo objetos: 100% (2/2), listo.

Escribiendo objetos: 100% (3/3), 335 bytes | 335.00 KiB/s, listo.

Total 3 (delta 0), reusado 0 (delta 0)

remote:

remote: Create a pull request for 'branchname' on GitHub by visiting:

remote: <https://github.com/MNGARCIA085/RAMAS/pull/new/branchname>

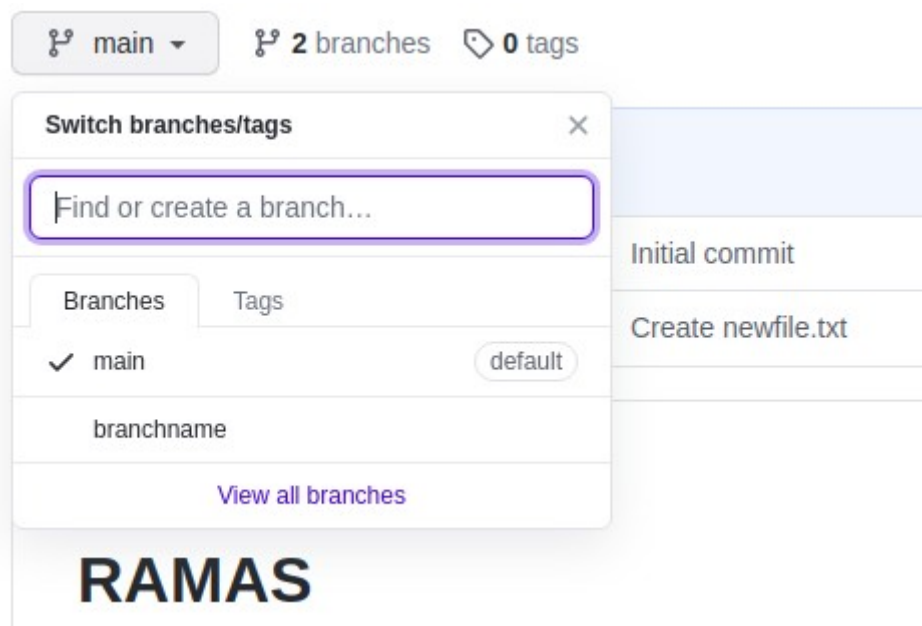
remote:

To github.com:MNGARCIA085/RAMAS.git

* [new branch] branchname -> branchname

Rama 'branchname' configurada para hacer seguimiento a la rama remota 'branchname' de 'origin'.

Ahora tenemos 2 ramas en GitHub:



6. Cambie a la rama master usando git checkout master.

7. Use el comando git merge mybranch para realizar un merge de las ramas.

git checkout main

git merge branchname

git push

marcos@marcos-HP-Laptop-15-bs0xx:~/Escritorio/GIT/RAMAS\$ **git checkout main**

Cambiado a rama 'main'

Tu rama está actualizada con 'origin/main'.

```
marcos@marcos-HP-Laptop-15-bs0xx:~/Escritorio/GIT/RAMAS$ git merge branchname
Actualizando c01b315..0ed5ddd
Fast-forward
 stuffonbranch.txt | 1 +
 1 file changed, 1 insertion(+)
 create mode 100644 stuffonbranch.txt
```

```
marcos@marcos-HP-Laptop-15-bs0xx:~/Escritorio/GIT/RAMAS$ git push
```


Warning: Permanently added the RSA host key for IP address '140.82.114.4' to the list of known hosts.


Total 0 (delta 0), reusado 0 (delta 0)


To github.com:MNGARCIA085/RAMAS.git

c01b315..0ed5ddd main -> main

Con el comando de merge, la nueva rama será mezclada con la master y el nuevo archivo será insertado en ella. Antes teníamos 2 archivos, ahora tenemos 3, stuffonbranch.txt fue agregado.


 main ▾



 2 branches




 0 tags


Go to file

Add file ▾

 Code ▾

 MNGARCIA085 add 0ed5ddd 8 minutes ago  3 commits

 README.md	Initial commit	18 minutes ago
 newfile.txt	Create newfile.txt	14 minutes ago
 stuffonbranch.txt	add	8 minutes ago

README.md 

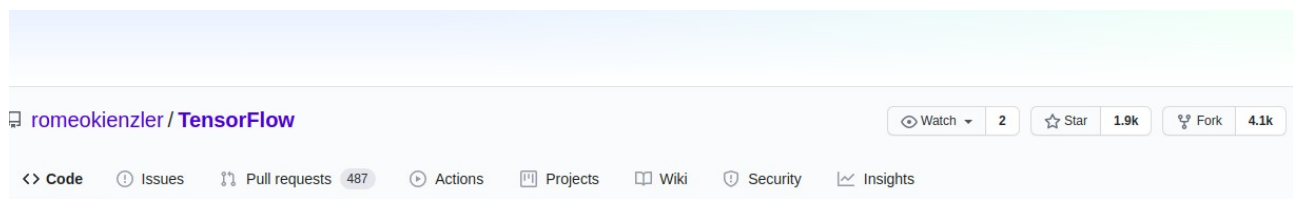
8. CONTRIBUYENDO A UN PROYECTO

Objetivo:

- Hacer fork de un repositorio en GitHub.
- Realizarle cambios al repositorio que se le hizo fork y confirmar los cambios.
- Crear pull requests.

Abrir el link <https://github.com/romeokienzler/TensorFlow/>

Haga clic en Fork y copie el repo en su cuenta.



Clone el repo:

git clone [git@github.com:MNGARCIA085/TensorFlow.git](https://github.com/MNGARCIA085/TensorFlow.git)

Edite y guarde cualquier archivo. Por ejemplo LICENCE. Aquí le agrego el texto 'hOLA' al final.

git clone [git@github.com:MNGARCIA085/TensorFlow.git](https://github.com/MNGARCIA085/TensorFlow.git)

luego agreguelo y confirme los cambios:

git add .

git commit -m "mensaje"

```
(base) marcos@marcos-HP-Laptop-15-bs0xx:~/Escritorio/GIT/TensorFlow$ git add .
```

```
(base) marcos@marcos-HP-Laptop-15-bs0xx:~/Escritorio/GIT/TensorFlow$ git commit -m "mensaje"
```

```
[master c49638d] mensaje  
1 file changed, 3 insertions(+)
```

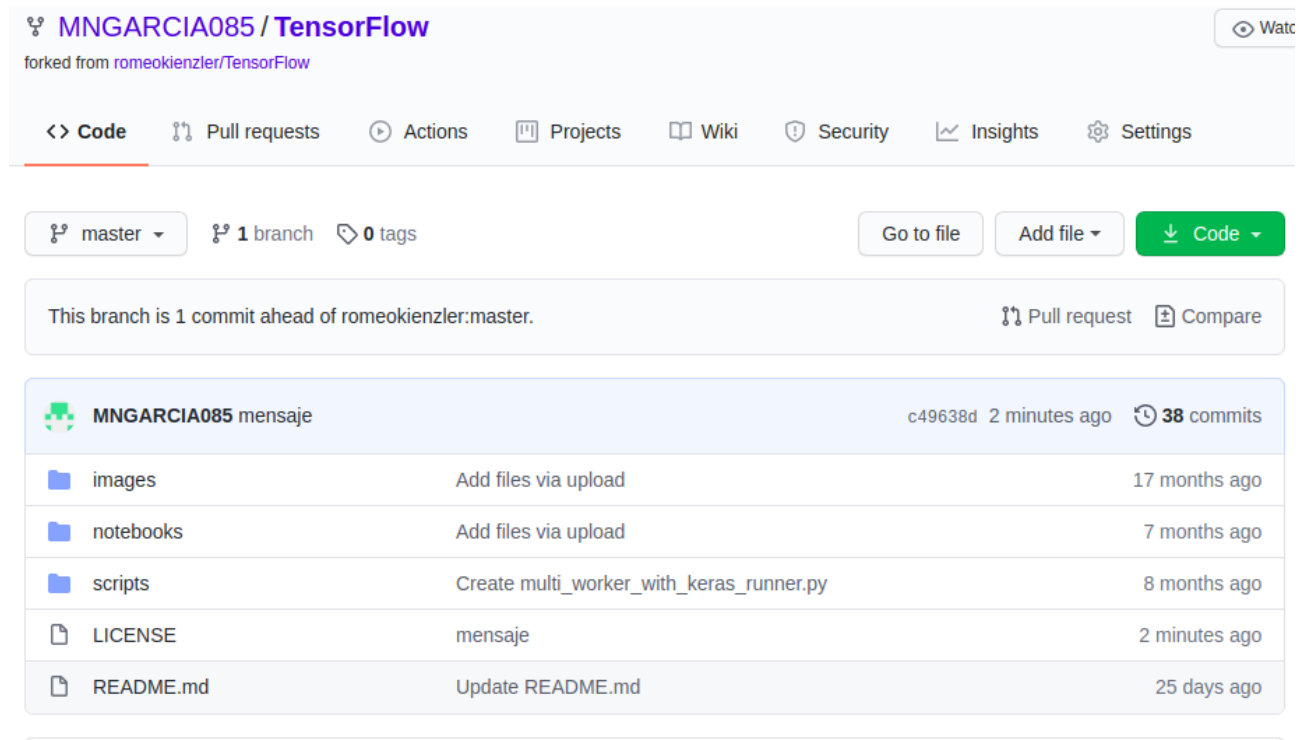
Haga git push para realizar los cambios en el repo remoto:

```
(base) marcos@marcos-HP-Laptop-15-bs0xx:~/Escritorio/GIT/TensorFlow$ git push
```

Warning: Permanently added the RSA host key for IP address '18.228.52.138' to the list of known hosts.

Enumerando objetos: 5, listo.
Contando objetos: 100% (5/5), listo.
Compresión delta usando hasta 4 hilos
Comprimiendo objetos: 100% (3/3), listo.
Escribiendo objetos: 100% (3/3), 405 bytes | 405.00 KiB/s, listo.
Total 3 (delta 1), reusado 0 (delta 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To github.com:MNGARCIA085/TensorFlow.git
cf5ea4b..c49638d master -> master

Chequee el repo para verificar:



MNGARCIA085 / TensorFlow Watch

forked from romeokienzler/TensorFlow

[Code](#) [Pull requests](#) [Actions](#) [Projects](#) [Wiki](#) [Security](#) [Insights](#) [Settings](#)

[master](#) [1 branch](#) [0 tags](#) [Go to file](#) [Add file](#) [Code](#)

This branch is 1 commit ahead of romeokienzler:master. [Pull request](#) [Compare](#)

	MNGARCIA085 mensaje	c49638d 2 minutes ago	38 commits
images	Add files via upload	17 months ago	
notebooks	Add files via upload	7 months ago	
scripts	Create multi_worker_with_keras_runner.py	8 months ago	
LICENSE	mensaje	2 minutes ago	
README.md	Update README.md	25 days ago	

Haga clic en “Compare” para comparar los cambios:

Comparing changes

Choose two branches to see what's changed or to start a new pull request. If you need to, you can also [compare across forks](#).

base repository: romeokienzler/TensorFlow

base: master

head repository: MNGARCIA085/TensorFlow

compare: master

✓ **Able to merge.** These branches can be automatically merged.

Discuss and review the changes in this comparison with others. [Learn about pull requests](#) [Create pull request](#)

1 commit

1 file changed

0 comments

1 contributor

Commits on Oct 08, 2020

mensaje

c49638d

Showing 1 changed file with 3 additions and 0 deletions.

Unified Split

3 LICENSE

@@ -1,3 +1,6 @@

1 1 This work is licensed under the Creative Commons Attribution-ShareAlike 4.0 International License.

2 2 To view a copy of this license, visit <http://creativecommons.org/licenses/by-sa/4.0/>

3 3 or send a letter to Creative Commons, PO Box 1866, Mountain View, CA 94042, USA.

4 +

5 +

6 + hOLA

Cree un Pull Request para realizar los cambios en el archivo original.

Nota. El pull request será enviado al repo del autor y si éste acepta los cambios, los mismos serán realizados en el repo original.

El mejor proceso para contribuir en la solución de un bug a un repositorio externo es hacer un fork del repositorio, actualizar el fork y crear un pull request.

9. RECUPERAR VERSIONES ANTERIORES DE ARCHIVOS

Creo un nuevo repositorio “**revertir**” en GitHub:

Lo clono:

```
git clone git@github.com:MNGARCIA085/revertir.git
```

Creo un archivo y lo agrego:

```
gedit prueba.txt
```

```
git add prueba.txt
```

```
git commit -m “primer commit”
```

```
git push
```

```
(base) marcos@marcos-HP-Laptop-15-bs0xx:~/Escritorio/GIT/revertir$ gedit prueba.txt  
(base) marcos@marcos-HP-Laptop-15-bs0xx:~/Escritorio/GIT/revertir$ git add prueba.txt  
(base) marcos@marcos-HP-Laptop-15-bs0xx:~/Escritorio/GIT/revertir$ git commit -m  
"primer commit"
```

```
[master f334ed8] primer commit  
1 file changed, 1 insertion(+)  
create mode 100644 prueba.txt
```

```
(base) marcos@marcos-HP-Laptop-15-bs0xx:~/Escritorio/GIT/revertir$ git push
```

Enumerando objetos: 4, listo.

Contando objetos: 100% (4/4), listo.

Compresión delta usando hasta 4 hilos

Comprimiendo objetos: 100% (2/2), listo.

Escribiendo objetos: 100% (3/3), 290 bytes | 290.00 KiB/s, listo.

Total 3 (delta 0), reusado 0 (delta 0)

To github.com:MNGARCIA085/revertir.git

9e5b375..f334ed8 master -> master

master 1 branch 0 tags

Go to file Add file Code

MNGARCIA085 primer commit f334ed8 44 seconds ago 2 commits

README.md	Initial commit	4 minutes ago
prueba.txt	primer commit	44 seconds ago

README.md

master revertir / prueba.txt

MNGARCIA085 primer commit

1 contributor

1 lines (1 sloc) | 12 Bytes

1 primer vers

Ahora modifico el archivo (le escribo segunda vers) y vuelvo a subirlo:

gedit prueba.txt

git add prueba.txt

git commit -m "segundo commit"

git push




```
(base) marcos@marcos-HP-Laptop-15-bs0xx:~/Escritorio/GIT/revertir$ gedit prueba.txt
(base) marcos@marcos-HP-Laptop-15-bs0xx:~/Escritorio/GIT/revertir$ git add prueba.txt
(base) marcos@marcos-HP-Laptop-15-bs0xx:~/Escritorio/GIT/revertir$ git commit -m
"segundo commit"
```

```
[master 864aa5e] segundo commit
1 file changed, 1 insertion(+), 1 deletion(-)
```


```
(base) marcos@marcos-HP-Laptop-15-bs0xx:~/Escritorio/GIT/revertir$ git push
```



Warning: Permanently added the RSA host key for IP address '18.228.67.229' to the list of known hosts.

Enumerando objetos: 5, listo.
Contando objetos: 100% (5/5), listo.
Compresión delta usando hasta 4 hilos
Comprimiendo objetos: 100% (2/2), listo.
Escribiendo objetos: 100% (3/3), 292 bytes | 292.00 KiB/s, listo.
Total 3 (delta 0), reusado 0 (delta 0)
To github.com:MNGARCIA085/revertir.git
f334ed8..864aa5e master -> master

 master ▾  1 branch  0 tags

[Go to file](#) [Add file ▾](#) [Code ▾](#)

 MNGARCIA085 segundo commit 864aa5e 30 seconds ago ⌚ 3 commits

 README.md	Initial commit	7 minutes ago
 prueba.txt	segundo commit	30 seconds ago

 master ▾ **revertir / prueba.txt**

 MNGARCIA085 segundo commit

 1 contributor

1 lines (1 sloc) | 13 Bytes

1 segunda vers

Repito con una tercera versión (que es la que “no va a funcionar”)

gedit prueba.txt

git add prueba.txt

git commit -m “tercer commit”

git push

master ▾
 1 branch
 0 tags
 Go to file
Add file ▾
Code ▾

MNGARCIA085 tercer commit		090718a 17 seconds ago	🕒 4 commits
📄 README.md	Initial commit	9 minutes ago	
📄 prueba.txt	tercer commit	17 seconds ago	

master ▾
 revertir / prueba.txt

MNGARCIA085 tercer commit

1 contributor

1 lines (1 sloc) | 13 Bytes

1 tercera vers

Ahora digamos que lo que subí recién causó un error, para volver a la versión anterior hago:

git checkout HEAD prueba.txt

Pero si quiero volver a otra versión, por ejemplo a la primera debo hacer:

git log --oneline

(base) marcos@marcos-HP-Laptop-15-bs0xx:~/Escritorio/GIT/revertir\$ **git log --oneline**

```

090718a (HEAD -> master, origin/master, origin/HEAD) tercer commit
864aa5e segundo commit
f334ed8 primer commit
9e5b375 Initial commit
    
```

Ahora vuelvo a la primera versión, utilizando el identificador que obtuve a través del comando “git log --oneline”:

git checkout f334ed8 prueba.txt

y vuelvo a subirlo:

git add prueba.txt

git commit -m "restaurar prueba.txt a la primera versión"

(base) marcos@marcos-HP-Laptop-15-bs0xx:~/Escritorio/GIT/revertir\$ **git checkout f334ed8 prueba.txt**

Actualizada 1 ruta para 8f49826

(base) marcos@marcos-HP-Laptop-15-bs0xx:~/Escritorio/GIT/revertir\$ **git add prueba.txt**
(base) marcos@marcos-HP-Laptop-15-bs0xx:~/Escritorio/GIT/revertir\$ **git commit -m "restaurar prueba.txt a la primera vers"**

[master c76fbc4] restaurar prueba.txt a la primera vers
1 file changed, 1 insertion(+), 1 deletion(-)

(base) marcos@marcos-HP-Laptop-15-bs0xx:~/Escritorio/GIT/revertir\$ **git push**

Enumerando objetos: 5, listo.

Contando objetos: 100% (5/5), listo.

Compresión delta usando hasta 4 hilos

Comprimiendo objetos: 100% (2/2), listo.

Escribiendo objetos: 100% (3/3), 310 bytes | 310.00 KiB/s, listo.

Total 3 (delta 0), reusado 0 (delta 0)

To github.com:MNGARCIA085/revertir.git

090718a..c76fbc4 master -> master

master

1 branch

0 tags

Go to file

Add file

Code

MNGARCIA085 restaurar prueba.txt a la primera vers

c76fbc4 1 minute ago 5 commits

README.md	Initial commit	17 minutes ago
prueba.txt	restaurar prueba.txt a la primera vers	1 minute ago

README.md

master

revertir / prueba.txt

MNGARCIA085 restaurar prueba.txt a la primera vers

1 contributor

1 lines (1 sloc) | 12 Bytes

1 primer vers

Para revertir todo el proyecto a la primera versión en lugar de solamente el archivo “prueba.txt” hacemos:

git checkout f334ed8