

TRABAJO PRÁCTICO N.º 2

GIT Y GITHUB

Nombre: Mariano Rodríguez Arce

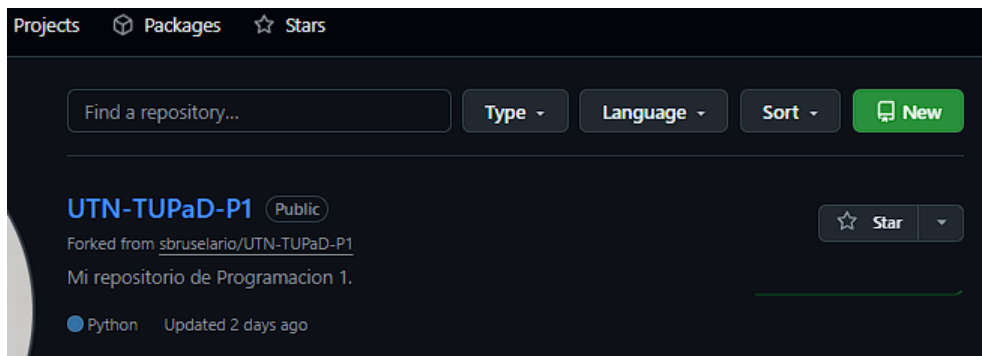
1) Contestar las siguientes preguntas utilizando las guías y documentación proporcionada (Desarrollar las respuestas):

a. • ¿Qué es GitHub?

GitHub es una plataforma que nos permite compartir nuestros repositorios de manera pública o privada. Podemos realizar diferentes acciones como seguir otros usuarios, ver código de proyectos de otras personas y hasta poder clonarlos y guardarlos.

b. • ¿Cómo crear un repositorio en GitHub?

Para crear un repositorio en GitHub tendremos que ingresar a nuestro perfil y hacer clic en el siguiente botón.



Luego podremos elegir un nombre para el repositorio junto con una descripción opcional y demás configuraciones.


Create a new repository

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

Required fields are marked with an asterisk ().*

Owner *


Repository name *

 Mariano-RA


 /

Great repository names are short and memorable. Need inspiration? How about **solid-dollop** ?

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:

☐ Add a README file

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

Add .gitignore


.gitignore template: None

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

Choose a license

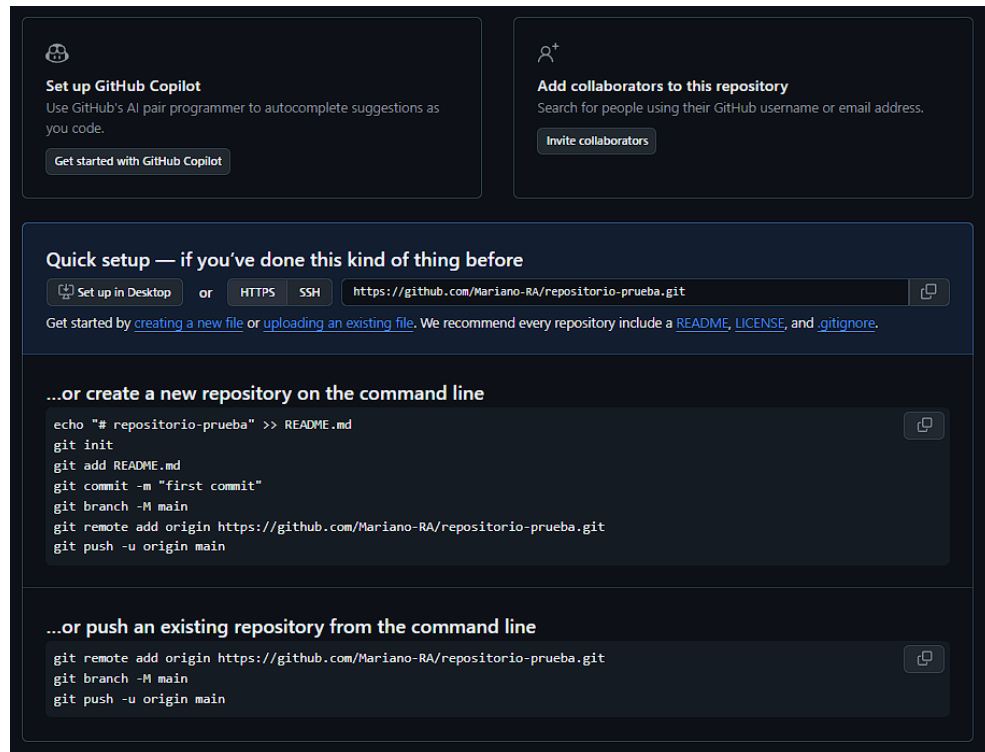
License: None

A license tells others what they can and can't do with your code. [Learn more about licenses.](#)

 You are creating a public repository in your personal account.

Create repository

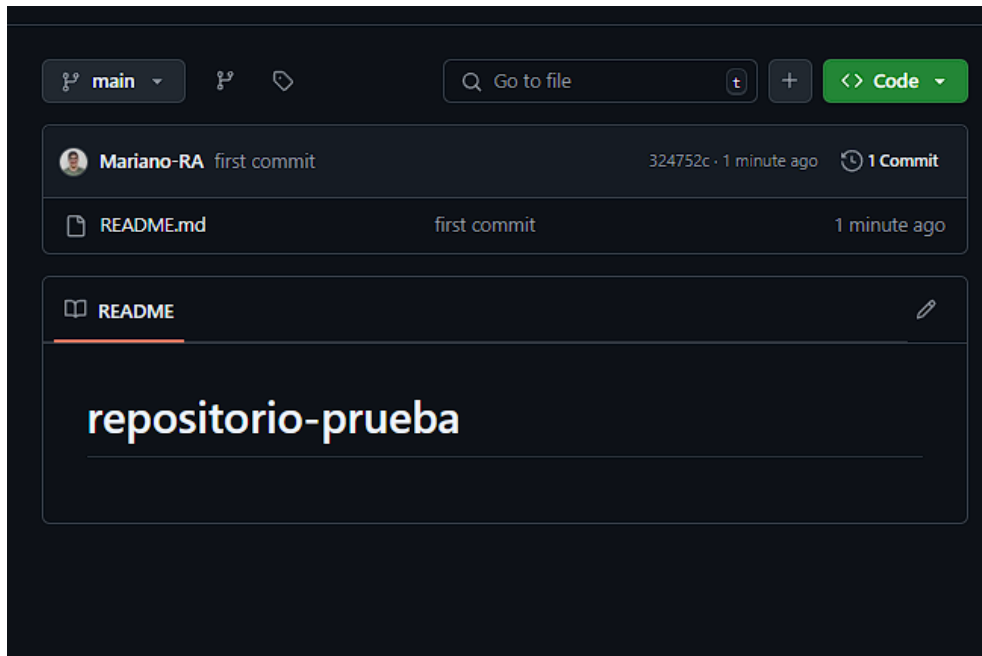
Una vez creado el repositorio podremos tener acceso al link para poder clonarlo y crear el repositorio local con el comando **Git init** o también pushear un repositorio git creado anteriormente



De manera local podremos ver el resultado de los comandos de la siguiente forma

```
>> git init
>> git add README.md
>> git commit -m "first commit"
>> git branch -M main
>> git remote add origin https://github.com/Mariano-RA/repositorio-prueba.git
>> git push -u origin main
Initialized empty Git repository in C:/Users/nanos/Documents/Tecnicatura/Programacion/repoPrueba/.git/
[master (root-commit) 324752c] first commit
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 README.md
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Writing objects: 100% (3/3), 263 bytes | 263.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/Mariano-RA/repositorio-prueba.git
 * [new branch]      main -> main
branch 'main' set up to track 'origin/main'.
```

Y en Github se podrá visualizar de la siguiente manera



- c. • ¿Cómo crear una rama en Git?

Git branch nombre_rama -> permite crear una rama

Git checkout -b nombre_rama -> permite crear y cambiar a la rama creada

- d. • ¿Cómo cambiar a una rama en Git?

Git checkout nombre_rama

- e. • ¿Cómo fusionar ramas en Git?

Git merge rama -> Permite fusionar la branch rama con la branch en la que nos encontramos actualmente

- f. • ¿Cómo crear un commit en Git?

Git add . -> Para poder agregar los archivos a stage

Git commit -m "Mensaje" -> Para confirmar los cambios

- g. • ¿Cómo enviar un commit a GitHub?

`Git push -u origin master`

- h. • ¿Qué es un repositorio remoto?

Es una copia de un proyecto que se encuentra subido a la nube y a la cual podemos acceder siempre que tengamos la URL necesaria

- i. • ¿Cómo agregar un repositorio remoto a Git?

`Git remote add origin url`

- j. • ¿Cómo empujar cambios a un repositorio remoto?

`Git push -u origin nombre_rama`

- k. • ¿Cómo tirar de cambios de un repositorio remoto?

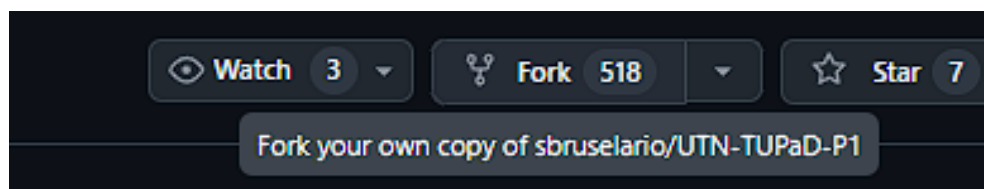
`Git pull origin nombre_rama`

- l. • ¿Qué es un fork de repositorio?

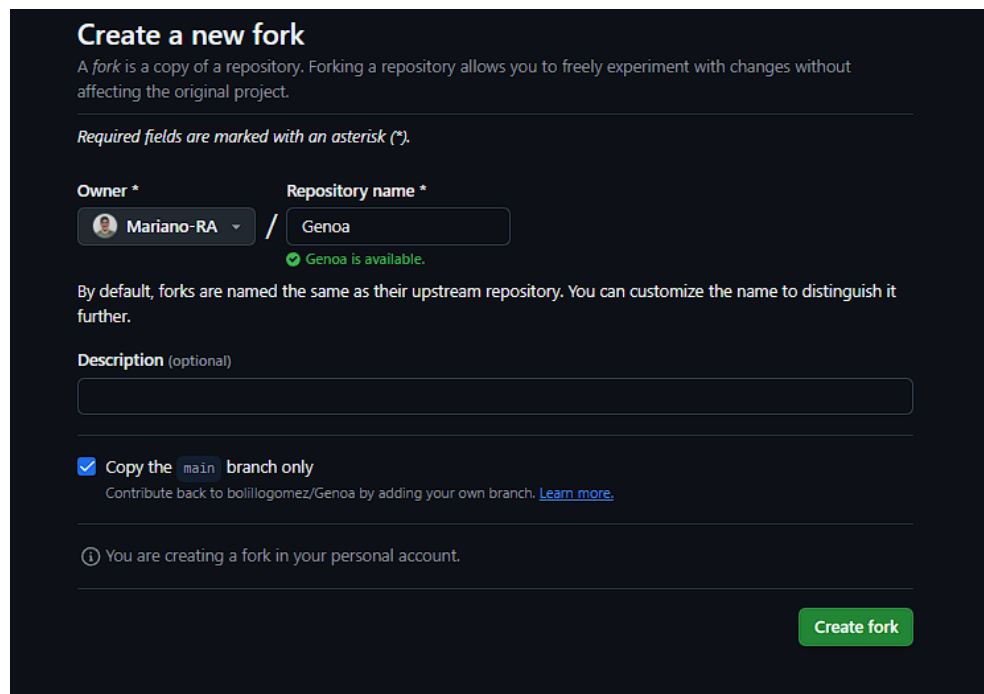
Fork es cuando realizamos una copia de un repositorio de algún usuario de GitHub en nuestra cuenta lo que nos permite modificar el código libremente sin afectar el repositorio original

- m. • ¿Cómo crear un fork de un repositorio?

Tenemos que ingresar al repositorio que queremos copiar



Completar los datos que nos pide como por ejemplo el nombre que le vamos a asignar y una breve descripción




Create a new fork

A fork is a copy of a repository. Forking a repository allows you to freely experiment with changes without affecting the original project.

Required fields are marked with an asterisk (*).

Owner * **Repository name ***

 Mariano-RA / Genoa


✔ Genoa is available.

By default, forks are named the same as their upstream repository. You can customize the name to distinguish it further.

Description (optional)

☒ Copy the `main` branch only

Contribute back to bolillogomez/Genoa by adding your own branch. [Learn more.](#)

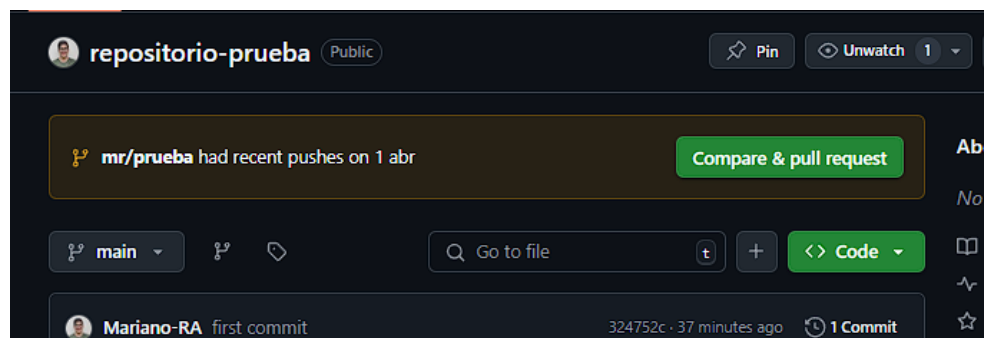
 You are creating a fork in your personal account.

[Create fork](#)

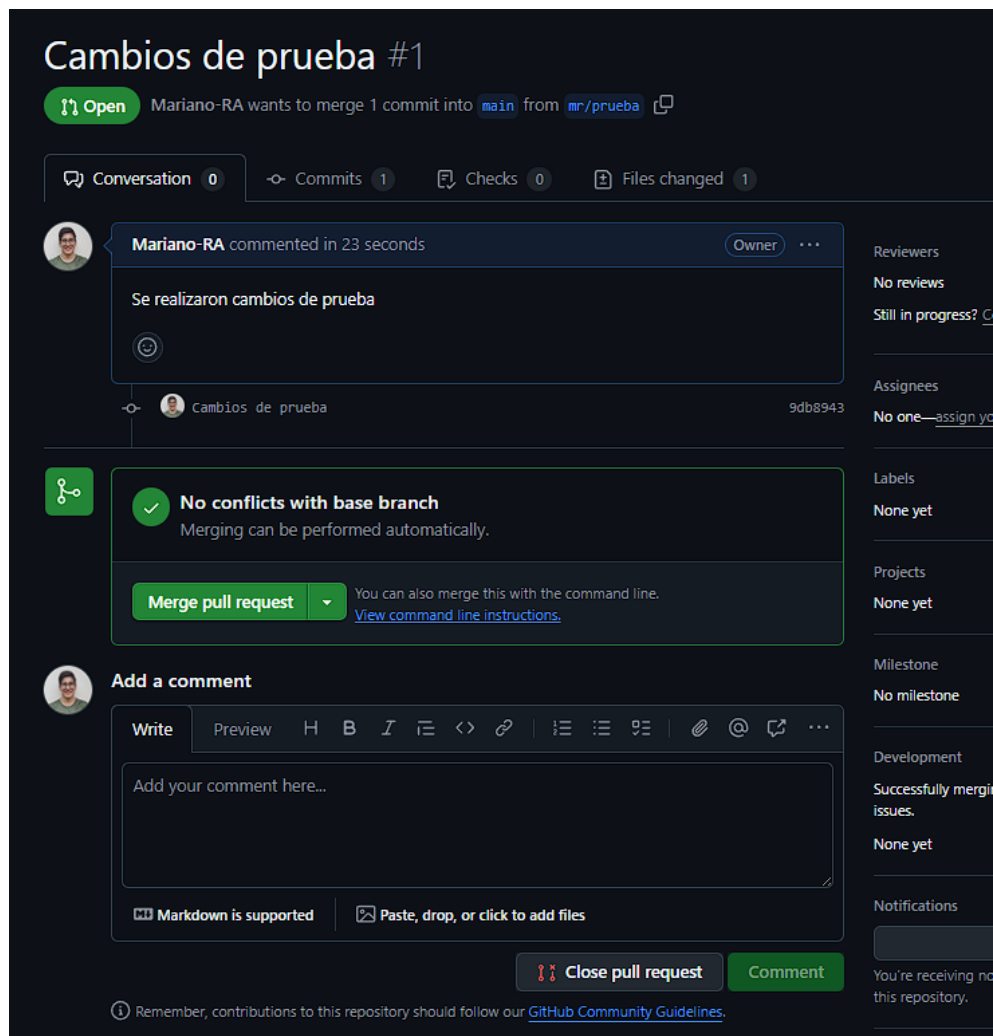
Y una vez realizado ya podremos acceder al repositorio desde nuestra cuenta

n. ¿Cómo enviar una solicitud de extracción (pull request) a un repositorio?

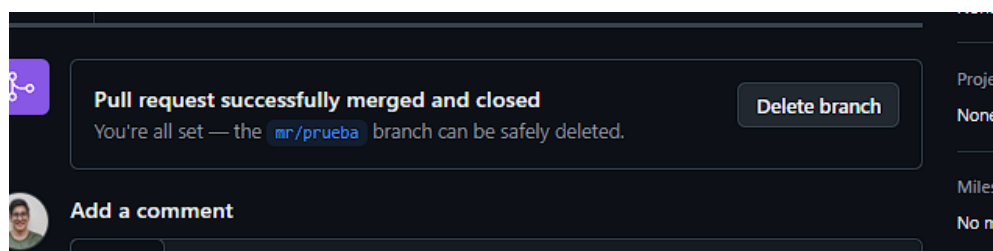
Tendremos que crear una branch sobre la cual vamos a trabajar, luego que hagamos los cambios necesarios y de pushearlos a nuestro repositorio podremos visualizarlos en la interfaz de GitHub.



Haciendo clic en el botón vamos a poder acceder a una pagina que nos permite cargar un título y una descripción al pull request

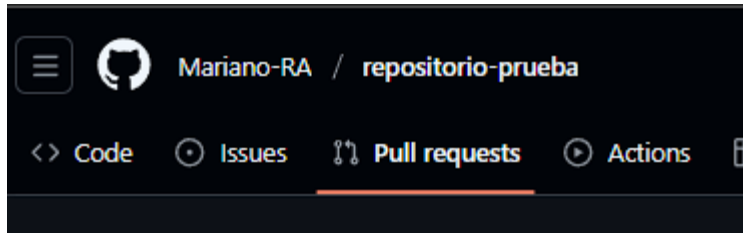


Luego de crear el título y descripción de manera automática GitHub va a chequear si existen conflictos entre la branch desde la cual estamos realizando el pull request y la branch hacia donde estamos pusheando. En el caso de que no haya conflictos podremos aceptar el merge y borrar la branch en el caso que sea necesario.

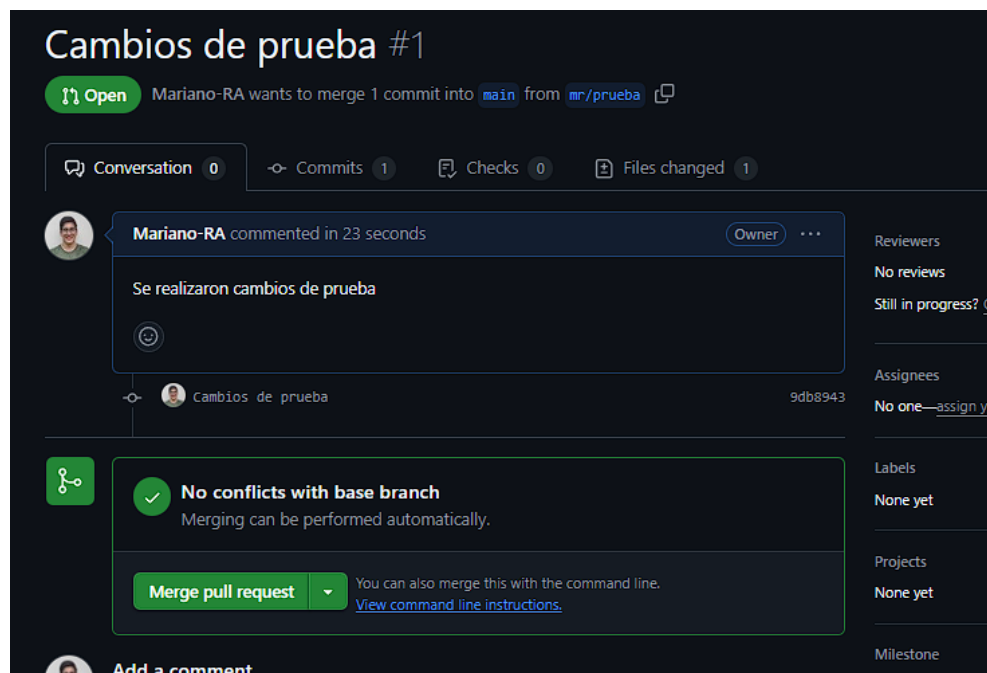


- o. • ¿Cómo aceptar una solicitud de extracción?

Cuando accedemos a la pestaña de pull request podremos ver los que están pendientes y asignados a nosotros



Una vez que se chequee que no existan conflictos y que los archivos que se están por subir son los correctos podremos hacer clic en Merge para poder finalizar el pull request



- p. • ¿Qué es una etiqueta en Git?

Git tiene la posibilidad de etiquetar puntos específicos del historial como importantes. Esta funcionalidad se usa típicamente para marcar versiones de lanzamiento

- q. • ¿Cómo crear una etiqueta en Git?

Git tag nombre -> Para poder crear la etiqueta

Git tag -a nombre -m mensaje -> Para poder crear un tag con un mensaje específico

- r. • ¿Cómo enviar una etiqueta a GitHub?

Git push --tags -> Para poder subir cambios incluyendo todos los tags

- s. • ¿Qué es un historial de Git?

El historial de Git es el registro de todos los commits realizados sobre un repositorio.

- t. • ¿Cómo ver el historial de Git?

Git log -> Para poder visualizar todos los commits

Git log --oneline -> Para poder visualizar los commits pero de manera más resumida

- u. • ¿Cómo buscar en el historial de Git?

Git log --author "nombre de autor" -> Nos permite filtrar el historial por autor

Git log --grep="texto" -> Nos permite buscar palabras o grupos de palabras en la rama en la que se esté trabajando

- i. • ¿Cómo borrar el historial de Git?

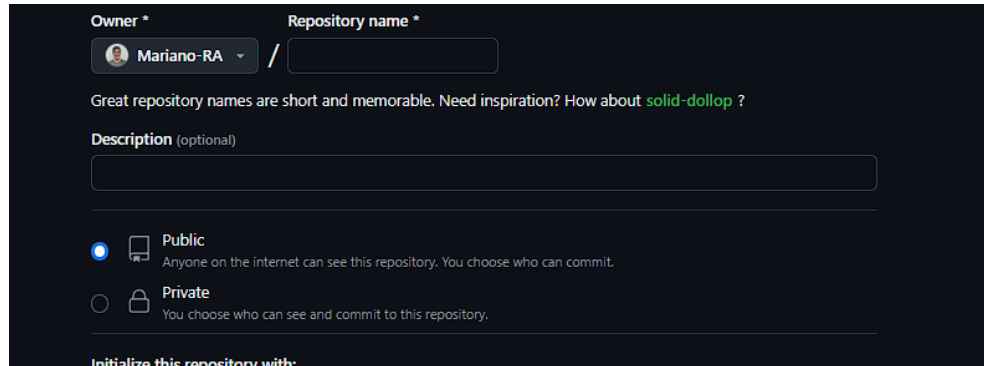
De manera local solamente se pueden borrar los commits que no se han pusheado. Una vez que son pusheados no es posible eliminar el commit

- v. • ¿Qué es un repositorio privado en GitHub?

Un repositorio privado es aquel el cual es solo accesible por los usuarios autorizados y no se puede visualizar públicamente.

- w. • ¿Cómo crear un repositorio privado en GitHub?

Cuando creamos el repositorio debemos seleccionar la opción Private



The screenshot shows the GitHub repository creation interface. At the top, there are two fields: 'Owner *' with a dropdown menu showing 'Mariano-RA' and a profile picture, and 'Repository name *' with an empty text box. Below these is a hint: 'Great repository names are short and memorable. Need inspiration? How about solid-dollop ?'. Then, there is a 'Description (optional)' text area. The visibility options are 'Public' (selected with a blue radio button and a globe icon) and 'Private' (unselected with a white radio button and a lock icon). The 'Private' option has a description: 'You choose who can see and commit to this repository.' At the bottom, there is a link that says 'Initialize this repository with:'.

- x. • ¿Cómo invitar a alguien a un repositorio privado en GitHub?

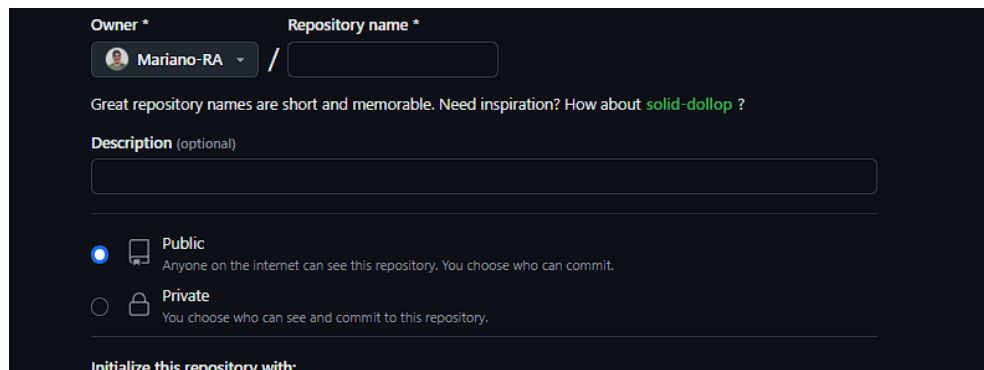
Debemos entrar a la configuración del proyecto, hacer clic en Colaboradores e ingresar el nombre de usuario o correo de las personas que queremos invitar. También podremos que seleccionar que rol le queremos asignar

- y. • ¿Qué es un repositorio público en GitHub?

Un repositorio público es aquel que es accesible por cualquier usuario de GitHub sin restricciones

- z. • ¿Cómo crear un repositorio público en GitHub?

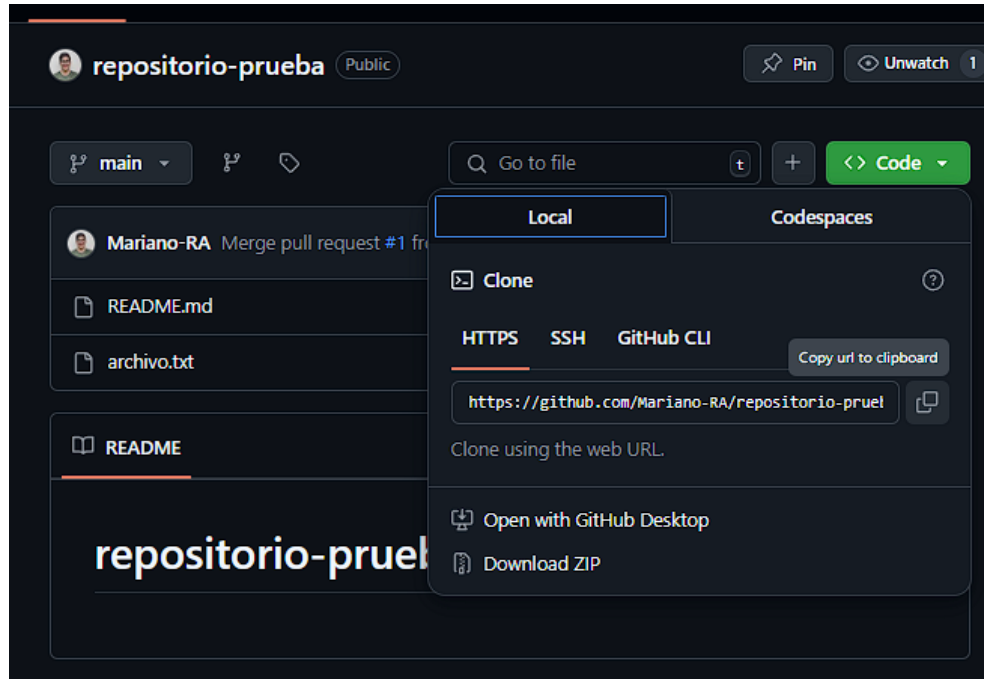
Cuando creamos el repositorio debemos seleccionar la opción Public



This screenshot is identical to the one above, showing the GitHub repository creation form. In this instance, the 'Public' option is selected with a blue radio button and a globe icon. The 'Public' option has a description: 'Anyone on the internet can see this repository. You choose who can commit.' The 'Private' option remains unselected.

aa. • ¿Cómo compartir un repositorio público en GitHub?

Una vez creado el repositorio deberemos compartir la URL que nos proporciona GitHub



2) Realizar la siguiente actividad:

- a. • Crear un repositorio.
 - i. o Dale un nombre al repositorio.
 - ii. o Elije el repositorio sea público.
 - iii. o Inicializa el repositorio con un archivo.
- b. • Agregando un Archivo
 - i. o Crea un archivo simple, por ejemplo, "mi-archivo.txt".
 - ii. o Realiza los comandos git add . y git commit -m "Agregando mi-archivo.txt" en la línea de comandos.
 - iii. o Sube los cambios al repositorio en GitHub con git push origin main (o el nombre de la rama correspondiente).

```
PS C:\Users\nanos\Documents\Tecnicatura\Programacion> cd .\repositorio-prueba\  
PS C:\Users\nanos\Documents\Tecnicatura\Programacion\repositorio-prueba> git add .  
PS C:\Users\nanos\Documents\Tecnicatura\Programacion\repositorio-prueba> git commit -m 'Agregando mi-archivo.txt'  
[main 2c17dbc] Agregando mi-archivo.txt  
1 file changed, 1 insertion(+)  
create mode 100644 mi-archivo.txt
```

- c. • Creando Branchs
 - i. o Crear una Branch
 - ii. o Realizar cambios
 - iii. o agregar un archivo o Subir la Branch

```
PS C:\Users\nanos\Documents\Tecnicatura\Programacion\repositorio-prueba> git branch branch-prueba  
PS C:\Users\nanos\Documents\Tecnicatura\Programacion\repositorio-prueba> git checkout branch-prueba  
Switched to branch 'branch-prueba'  
PS C:\Users\nanos\Documents\Tecnicatura\Programacion\repositorio-prueba> git add .  
PS C:\Users\nanos\Documents\Tecnicatura\Programacion\repositorio-prueba> git commit -m 'se agrega nuevo archivo a la branch'  
[branch-prueba 87a18e6] se agrega nuevo archivo a la branch  
1 file changed, 1 insertion(+)  
create mode 100644 nuevo-archivo.txt
```

URL: <https://github.com/Mariano-RA/repositorio-prueba>

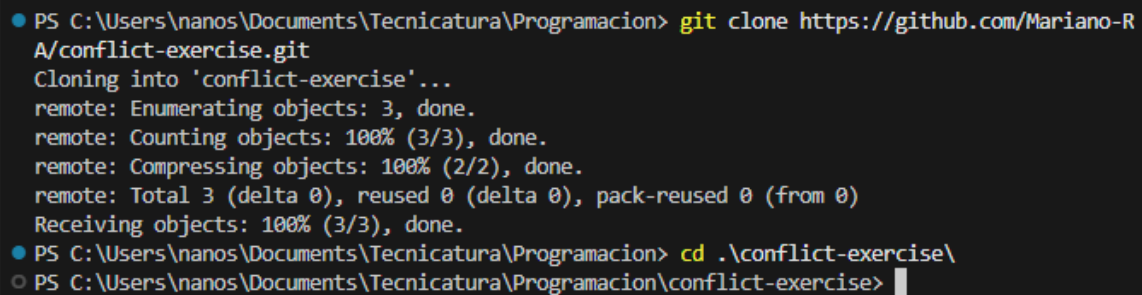
3) Realizar la siguiente actividad:

a. Paso 1: Crear un repositorio en GitHub

- i. • Ve a GitHub e inicia sesión en tu cuenta.
- ii. • Haz clic en el botón "New" o "Create repository" para crear un nuevo repositorio.
- iii. • Asigna un nombre al repositorio, por ejemplo, conflict-exercise.
- iv. • Opcionalmente, añade una descripción.
- v. • Marca la opción "Initialize this repository with a README".
- vi. • Haz clic en "Create repository".

b. Paso 2: Clonar el repositorio a tu máquina local

- i. • Copia la URL del repositorio (usualmente algo como <https://github.com/tuusuario/conflict-exercise.git>).
- ii. • Abre la terminal o línea de comandos en tu máquina.
- iii. • Clona el repositorio usando el comando:
`git clone https://github.com/tuusuario/conflict-exercise.git`
- iv. • Entra en el directorio del repositorio:
`cd conflict-exercise`



```
PS C:\Users\nanos\Documents\Tecnicatura\Programacion> git clone https://github.com/Mariano-R
A/conflict-exercise.git
Cloning into 'conflict-exercise'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (3/3), done.
PS C:\Users\nanos\Documents\Tecnicatura\Programacion> cd .\conflict-exercise\
PS C:\Users\nanos\Documents\Tecnicatura\Programacion\conflict-exercise>
```

c. Paso 3: Crear una nueva rama y editar un archivo

- i. • Crea una nueva rama llamada feature-branch:
`git checkout -b feature-branch`
- ii. • Abre el archivo README.md en un editor de texto y añade una línea nueva, por ejemplo: Este es un cambio en la feature branch.
- iii. • Guarda los cambios y haz un commit:
`git add README.md`
`git commit -m "Added a line in feature-branch"`


```
conflict-exercise > ① README.md > # <<<<<< HEAD Este es un cambio en la main branch.  
You, 1 second ago | 2 authors (You and one other)  
1 # conflict-exercise  
2 Repositorio creado para practicar  
3  
Accept Current Change | Accept Incoming Change | Accept Both Changes | Compare Changes  
4 <<<<<< HEAD (Current Change)  
5 Este es un cambio en la main branch.  
6 =====  
7 Este es un cambio en la feature branch  
8 >>>>>> feature-branch (Incoming Change)  
9
```

[Resolve in Merge Editor](#)

```
PROBLEMS 5 OUTPUT DEBUG CONSOLE TERMINAL PORTS ... Python - conflict-exercise + v [ ] [X] ... X  
PS C:\Users\nanos\Documents\Tecnicatura\Programacion\conflict-exercise> git merge feature-branch  
Auto-merging README.md  
CONFLICT (content): Merge conflict in README.md  
Automatic merge failed; fix conflicts and then commit the result.  
PS C:\Users\nanos\Documents\Tecnicatura\Programacion\conflict-exercise>
```

- f. Paso 6: Resolver el conflicto
- i.
 - Abre el archivo README.md en tu editor de texto. Verás algo similar a esto:

```
<<<<<<< HEAD
Este es un cambio en la main branch.
=====
Este es un cambio en la feature branch.
>>>>>>> feature-branch
```
 - ii.
 - Decide cómo resolver el conflicto. Puedes mantener ambos cambios, elegir uno de ellos, o fusionar los contenidos de alguna manera.
 - iii.
 - Edita el archivo para resolver el conflicto y guarda los cambios(Se debe borrar lo marcado en verde en el archivo donde estes solucionando el conflicto. Y se debe borrar la parte del texto que no se quiera dejar).
 - iv.
 - Añade el archivo resuelto y completa el merge:

```
PS C:\Users\nanos\Documents\Tecnatura\Programacion\conflict-exercise> git add README.md
PS C:\Users\nanos\Documents\Tecnatura\Programacion\conflict-exercise> git commit -m "Resolved merge conflict"
[main 393abae] Resolved merge conflict
PS C:\Users\nanos\Documents\Tecnatura\Programacion\conflict-exercise>
```

- ```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS ... Python - conflict-exercise + - [X] [X] [X] [X]
```
- ```
PS C:\Users\nanos\Documents\Tecnicatura\Programacion\conflict-exercise> git push origin main  
Enumerating objects: 9, done.  
Counting objects: 100% (9/9), done.  
Delta compression using up to 12 threads  
Compressing objects: 100% (5/5), done.  
Writing objects: 100% (7/7), 661 bytes | 661.00 KiB/s, done.  
Total 7 (delta 3), reused 0 (delta 0), pack-reused 0  
remote: Resolving deltas: 100% (3/3), done.  
To https://github.com/Mariano-RA/conflict-exercise.git  
c0e5f9a..393abae main -> main  
  
PS C:\Users\nanos\Documents\Tecnicatura\Programacion\conflict-exercise> git push origin feature-branch  
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0  
remote:  
remote: Create a pull request for 'feature-branch' on GitHub by visiting:  
remote:   https://github.com/Mariano-RA/conflict-exercise/pull/new/feature-branch  
remote:  
To https://github.com/Mariano-RA/conflict-exercise.git  
 * [new branch]      feature-branch -> feature-branch  
  
💡 PS C:\Users\nanos\Documents\Tecnicatura\Programacion\conflict-exercise>
```



```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS ... Python - conflict-exercise + v [ ] [ ] ...

• PS C:\Users\nanos\Documents\Tecnatura\Programacion\conflict-exercise> git log
commit 393abae37733b34f8a0d3fd72f9134cd8630e5c1 (HEAD -> main, origin/main, origin/HEAD)
Merge: 6fa0c81 0c1fc65
Author: Mariano RA <rodriguezarce.mariano@gmail.com>
Date: Tue Apr 8 19:19:11 2025 -0300

    Resolved merge conflict

commit 6fa0c81fcab45f0edc23095496505cde6e73a80e
Author: Mariano RA <rodriguezarce.mariano@gmail.com>
Date: Tue Apr 8 19:16:57 2025 -0300

    Added a line in main branch

commit 0c1fc652cc6e6f4454d6ac800e90f4e3051bec41 (origin/feature-branch, feature-branch)
Author: Mariano RA <rodriguezarce.mariano@gmail.com>
Date: Tue Apr 8 19:16:00 2025 -0300

    Added a line in feature-branch

commit c0e5f9a7e057b5199a5e37ef254bf1e1c073a581
Author: Mariano-RA <100177646+Mariano-RA@users.noreply.github.com>
Date: Tue Apr 8 18:53:10 2025 -0300

    Initial commit

• PS C:\Users\nanos\Documents\Tecnatura\Programacion\conflict-exercise> 
```

URL: <https://github.com/Mariano-RA/conflict-exercise>