

## Práctico 2: Git y GitHub

**Alumno:** *Emiliano Bisio*

### Actividad 1

- **¿Qué es GitHub?**

*GitHub es una plataforma basada en la nube donde puedes almacenar, compartir y trabajar junto con otros usuarios para escribir código. El trabajo colaborativo, una de las características fundamentales de GitHub, es posible gracias al software de código abierto Git, en el que se basa GitHub.*

- **¿Cómo crear un repositorio en GitHub?**

1. En la esquina superior derecha de cualquier página, selecciona + y luego haz clic en “*New repository*”.
2. En el cuadro “*Repository name*”, escriba un nombre para su repositorio.
3. En el cuadro “*Description*”, escriba una breve descripción (es optativo).
4. Seleccione si el repositorio será Público o Privado.
5. Seleccione “*Add a README file*” (es optativo).
6. Haga clic en “*Create repository*”.

- **¿Cómo crear una rama en Git?**

Con el comando “**git checkout -b** <nombre de la rama>”.  
Por ejemplo: `git checkout -b feature-branch`

- **¿Cómo cambiar a una rama en Git?**

Se usa el comando “**git checkout** <nombre de la rama>”.  
Por ejemplo: `git checkout develop`

- **¿Cómo fusionar ramas en Git?**

El comando “**git merge**” mas el nombre de la rama que queremos fusionar.  
Por ejemplo, si estoy en la rama “main” y quiero fusionarla con la rama “develop” tengo que ejecutar el comando: `git merge develop`.

- **¿Cómo crear un commit en Git?**

Usando el comando “**git commit -m** “*Una descripción breve de los cambios realizados*”.

- **¿Cómo enviar un commit a GitHub?**

Con “**git push**” puedo enviar un commit a GitHub.

Ejemplo: `git push origin main`, donde “origin” es el nombre del repositorio remoto y “main” la rama.

- **¿Qué es un repositorio remoto?**

Es el lugar donde se almacena nuestro código en un servidor que se puede acceder a través de una red. Las plataformas como GitHub nos ofrecen repositorios remotos que facilitan el desarrollo colaborativo, permitiendo compartir código.

- **¿Cómo agregar un repositorio remoto a Git?**

Usamos el comando “**git remote add**” seguido de un nombre que elijamos para el remoto y la URL del repositorio remoto.

Por ejemplo: `git remote add origin https://github.com/usuario/nombre-repo.git`

- **¿Cómo empujar cambios a un repositorio remoto?**

El comando “**git push**” seguido del nombre del remoto y la rama a la queremos empujar los cambios.

Por ejemplo: `git push origin main`

- **¿Cómo tirar de cambios de un repositorio remoto?**

Con “**git pull**” seguido del nombre del remoto y la rama.

Por ejemplo: `git pull origin main`

- **¿Qué es un fork de repositorio?**

Es una copia completa de un repositorio original que luego podemos utilizar como un repositorio Git cualquiera. Cuando hacemos un fork de un repositorio, se crea un nuevo repositorio en nuestra cuenta de GitHub con una URL diferente al repositorio original.

- **¿Cómo crear un fork de un repositorio?**

1. Ingresamos al repositorio de GitHub que queremos hacer el fork.
2. Hacemos clic en el botón de “Fork”.
3. Seleccionamos nuestra cuenta de GitHub.
4. Ahora tenemos una copia completa del repositorio original en nuestra cuenta.

- **¿Cómo enviar una solicitud de extracción (pull request) a un repositorio?**

1. Clonar en repositorio del repositorio para colaborar.
2. Creo una rama nueva para trabajar en los cambios.
3. Agrego y confirmo los cambios.
4. Envié la rama al repositorio remoto.
5. En el repositorio de GitHub hago clic en el botón “Pull request”.

- **¿Cómo aceptar una solicitud de extracción?**

1. En el repositorio, hacer clic en la pestaña “Pull request”.
2. Busco la solicitud a la cual quiero hacer la revisión.
3. Hacer clic en la opción “Files changed”.
4. Hacer clic en el botón “Review changes” sobre el código cambiado.
5. Clic en “Approve” para aceptar los cambios propuestos en el pull request y luego en “Submit review”.

- **¿Qué es un etiqueta en Git?**

Sirve para hacer referencia a puntos específicos del historial. Se utilizan para marcar momentos importantes, como el lanzamiento de una versión.

Existen 2 tipos de etiquetas en Git, las ligeras y las anotadas.

- **¿Cómo crear una etiqueta en Git?**

Crear una etiqueta ligera: `git tag nombre-etiqueta`

Crear un etiqueta anotada: `git tag -a nombre-etiqueta -m “Descripción de la etiqueta”`.

- **¿Cómo enviar una etiqueta a GitHub?**

Usando el comando: `git push origin nombre-etiqueta`

- **¿Qué es un historial de Git?**

Es el registro completo de todos los cambios realizados en un repositorio.

- **¿Cómo ver el historial de Git?**

Con el comando: `git log`

- **¿Cómo buscar en el historial de Git?**

Una de las formas de buscar sería utilizando el comando: `git log --grep="palabra clave"`

- **¿Cómo borrar el historial de Git?**

Si deseamos borrar el historial completo, podemos eliminar el repositorio. Es decir, eliminar la carpeta `.git` y luego volver a iniciar un proyecto git en la carpeta.

- **¿Qué es un repositorio privado en GitHub?**

Un repositorio privado en GitHub es un tipo de repositorio donde el contenido y el historial del proyecto están restringidos y sólo son accesibles para las personas que tienen permisos específicos. Esto significa que nadie fuera de los colaboradores autorizados podrá ver ni interactuar con el repositorio.

- **¿Cómo crear un repositorio privado en GitHub?**

En tu perfil de GitHub y haz clic en el botón *“New repository”*. Luego completa los campos como el nombre del repositorio y la descripción. En la sección de *“Repository visibility”* selecciona *“Private”*. Por último configura los permisos e invita a colaboradores según sea necesario.

- **¿Cómo invitar a alguien a un repositorio privado en GitHub?**

Ir al repositorio privado al cual quieres invitar a alguien. Hacer clic en la pestaña *“Settings”* en la parte superior del repositorio. Luego, en el menú de la izquierda, selecciona *“Collaborators and teams”* dentro de la sección *“Access”*. En la sección de *“Collaborators”*, busca el campo para añadir usuarios. Escribe el nombre de usuario o la dirección de correo electrónico de la persona que deseas invitar. Haz clic en el botón *“Add collaborator”*. Esto enviará una invitación a la persona para que acepte el acceso al repositorio. El usuario invitado deberá aceptar la invitación desde su cuenta de GitHub. Para hacerlo, recibirá un correo electrónico o verá una notificación en GitHub. Puedes establecer el nivel de acceso que tendrá el colaborador (por ejemplo, solo lectura o permisos de escritura). Esto lo puedes ajustar en la misma sección donde agregaste al colaborador.

- **¿Qué es un repositorio público en GitHub?**

Es un tipo de repositorio que está disponible para cualquier persona en el mundo. Su contenido puede ser visto, copiado y utilizado por cualquier usuario de GitHub, incluso si no está directamente involucrado en el proyecto. Es ideal para proyectos que deseas compartir abiertamente o colaborar con la comunidad.

- **¿Cómo crear un repositorio público en GitHub?**

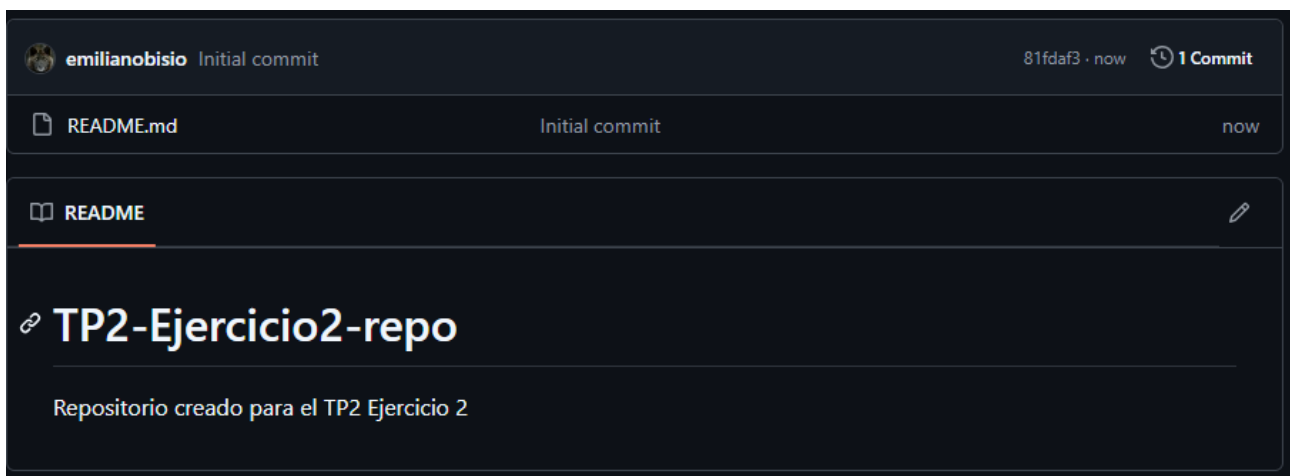
1. En la esquina superior derecha de cualquier página, selecciona + y luego haz clic en “New repository”.
2. En el cuadro “Repository name”, escriba un nombre para su repositorio.
3. En el cuadro “Description”, escriba una breve descripción (es optativo).
4. Seleccione que repositorio será Público.
5. Seleccione Agregar un archivo README (es optativo).
6. Haga clic en “Create repository”.

- **¿Cómo compartir un repositorio público en GitHub?**

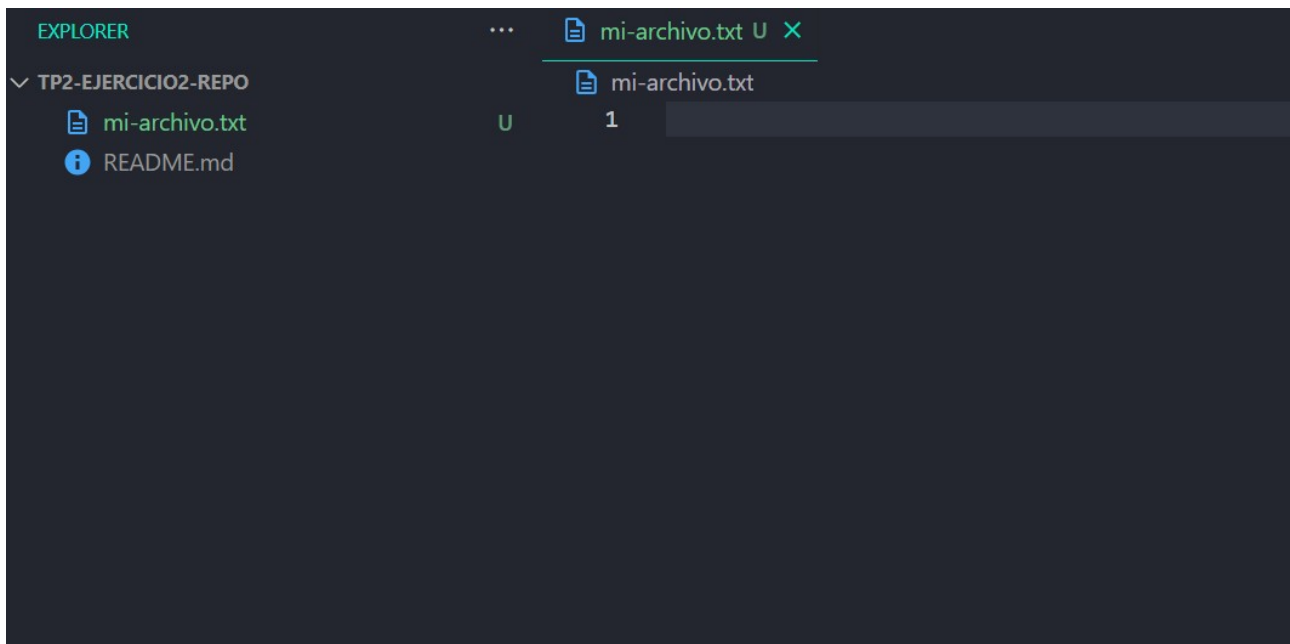
Ir a la pagina principal del repositorio en GitHub que queremos compartir y copiar la URL desde la barra de navegación de nuestro navegador.

## Actividad 2

- **Crea un repositorio e inicializa el repositorio con un archivo.**



- **Crea un archivo simple, por ejemplo, "mi-archivo.txt".**



- Realiza los comandos `git add .` y `git commit -m "Agregando mi-archivo.txt"` en la línea de comandos.

```
$ git commit -m "Added mi-archivo.txt"
[main e651ef8] Added mi-archivo.txt
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 mi-archivo.txt
```

- Sube los cambios al repositorio en GitHub con `git push origin main` (o el nombre de la rama correspondiente).

```
$ git push origin main
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 4 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 285 bytes | 285.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/emilianobisio/TP2-Ejercicio2-repo
81fdaf3..e651ef8  main -> main
```

- Crear una Branch

```
$ git checkout -b feature-branch
Switched to a new branch 'feature-branch'
```

- Subir la Branch

```
$ git push origin feature-branch
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Delta compression using up to 4 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (2/2), 299 bytes | 299.00 KiB/s, done.
Total 2 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
remote:
remote: Create a pull request for 'feature-branch' on GitHub by visiting:
remote:   https://github.com/emilianobisio/TP2-Ejercicio2-repo/pull/new/feature-branch
remote:
To https://github.com/emilianobisio/TP2-Ejercicio2-repo
 * [new branch]      feature-branch -> feature-branch
```

main

2 Branches

0 Tags

Go to file

Add file

<> Code

emilianobisio

Added mi-archivo.txt

e651ef8 · 6 minutes ago

2 Commits

README.md

Initial commit

11 minutes ago

mi-archivo.txt

Added mi-archivo.txt

6 minutes ago

README

TP2-Ejercicio2-repo

Repositorio creado para el TP2 Ejercicio 2

Branches

New branch

Overview

Yours

Active

Stale

All

Search branches...

Default

Branch	Updated	Check status	Behind	Ahead	Pull request
main	6 minutes ago				

Your branches

Branch	Updated	Check status	Behind	Ahead	Pull request
feature-branch	2 minutes ago		0	1	

Active branches

Branch	Updated	Check status	Behind	Ahead	Pull request
feature-branch	2 minutes ago		0	1	

## Actividad 3


- **Paso 1: Crear un repositorio 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.](#)

---

*Required fields are marked with an asterisk (\*).*

**Owner \***  
 **emilianobisio** ▾


**Repository name \***


✔ **conflict-exercise** is available.

Great repository names are short and memorable. Need inspiration? How about **curly-disco** ?

**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**

▾

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



- **Paso 2: Clonar el repositorio a tu máquina local**


```
$ git clone https://github.com/emilianobisio/conflict-exercise
Cloning into 'conflict-exercise'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (3/3), done.
```



- Paso 3: Crear una nueva rama y editar un archivo

```
$ git checkout -b feature-branch  
Switched to a new branch 'feature-branch'
```

 README.md M 

 README.md >  # conflict-exercise

1     # conflict-exercise

2     Este es un cambio en la feature branch

```
$ git commit -m "Added a line in feature-branch"  
[feature-branch 61ff056] Added a line in feature-branch  
1 file changed, 2 insertions(+), 1 deletion(-)
```

- Paso 4: Volver a la rama principal y editar el mismo archivo

```
$ git checkout main  
Switched to branch 'main'  
Your branch is up to date with 'origin/main'.
```

```
$ git commit -m "Added a line in main branch"  
[main a3c7681] Added a line in main branch  
1 file changed, 2 insertions(+), 1 deletion(-)
```

- **Paso 5: Hacer un merge y generar un conflicto**

```
$ 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.
```

```
1  README.md > # <<<<<< HEADEste es un cambio en la rama main
2  # conflict-exercise
3  Accept Current Change | Accept Incoming Change | Accept Both Changes | Compare Changes
4  <<<<<< HEAD (Current Change)
5  Este es un cambio en la rama main
6  =====
7  Este es un cambio en la feature branch
8  >>>>>> feature-branch (Incoming Change)
```

- **Paso 6: Resolver el conflicto**

```
$ git commit -m "Resolved merge conflict"
[main bcfa38b] Resolved merge conflict
```

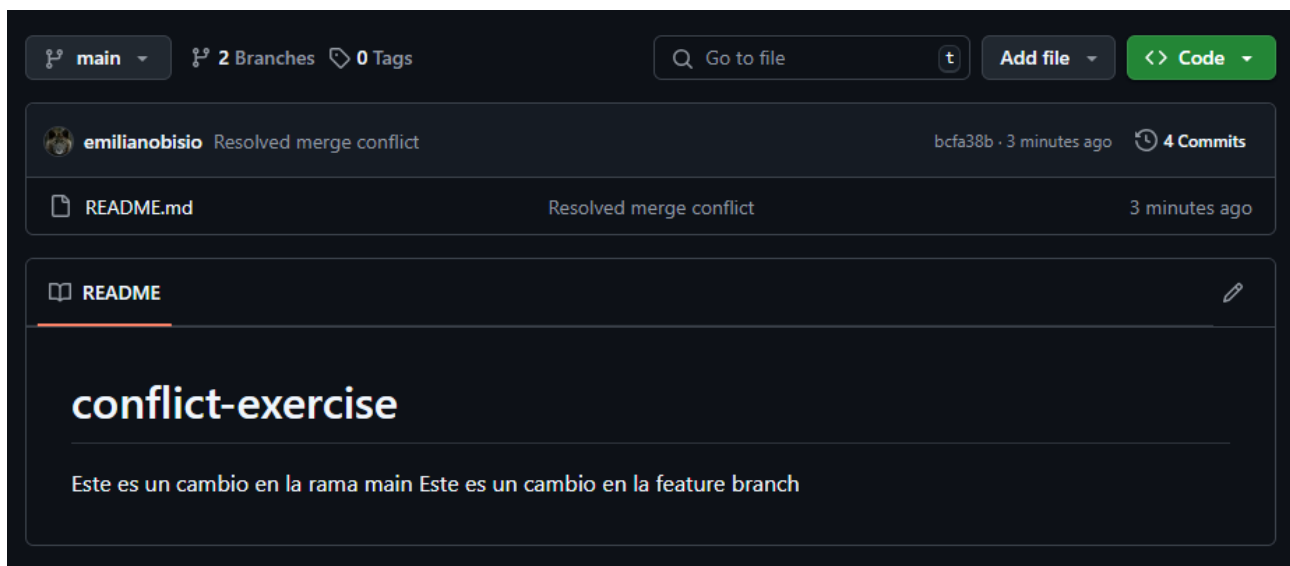
```
1  README.md X
2  README.md > # conflict-exercise
3  # conflict-exercise
4  Este es un cambio en la rama main
5  Este es un cambio en la feature branch
```

- **Paso 7: Subir los cambios a GitHub**

```
$ git push origin main
Enumerating objects: 11, done.
Counting objects: 100% (11/11), done.
Delta compression using up to 4 threads
Compressing objects: 100% (6/6), done.
Writing objects: 100% (9/9), 790 bytes | 395.00 KiB/s, done.
Total 9 (delta 2), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (2/2), done.
To https://github.com/emilianobisio/conflict-exercise
402c46e..bcfa38b  main -> main
```

```
$ git push origin feature-branch
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
remote:
remote: Create a pull request for 'feature-branch' on GitHub by visiting:
remote:   https://github.com/emilianobisio/conflict-exercise/pull/new/feature-branch
remote:
To https://github.com/emilianobisio/conflict-exercise
 * [new branch]      feature-branch -> feature-branch
```

- **Paso 8: Verificar en GitHub**



The screenshot shows the GitHub interface for the repository 'conflict-exercise' by user 'emilianobisio'. The top navigation bar includes the 'main' branch selector, '2 Branches', '0 Tags', a search bar, and buttons for 'Add file' and 'Code'. The main content area displays a commit titled 'Resolved merge conflict' by 'emilianobisio' with hash 'bcfa38b' and '4 Commits', made '3 minutes ago'. Below this, a file 'README.md' is shown with the same commit information. The 'README' file content is displayed, showing the title 'conflict-exercise' and a message: 'Este es un cambio en la rama main Este es un cambio en la feature branch'.