

Práctico 2: Git y GitHub

Alumno Marcelo Giammona

Actividades

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

• ¿Qué es GitHub?

GitHub es una plataforma de desarrollo colaborativo basada en la nube que permite a los desarrolladores almacenar, gestionar y compartir su código.

• ¿Cómo crear un repositorio en GitHub?

Inicio sesión en mi cuenta de GitHub.

Hago clic en el icono "+" que aparece en la esquina superior derecha de la página.

Selecciono "New repository" en el menú desplegable.

Le doy un nombre del repositorio

Agrego una descripción

Selecciono la opción público o privado

Selecciono la opción para inicializar el repositorio con un archivo README

Hago clic en el botón "Create repository"

• ¿Cómo crear una rama en Git?

Utilizo el comando

```
git branch nombre-de-la-rama
```

• ¿Cómo cambiar a una rama en Git?

Utilizo el comando

```
git checkout nombre-de-la-rama
```

• ¿Cómo fusionar ramas en Git?

Me aseguro de estar en la rama donde quiero recibir los cambios (rama destino). Por lo general, esta será la rama principal main o master. Utilizo el comando

```
git checkout main
```

Me aseguro de que la rama destino esté actualizada con el comando

```
git pull
```

Para fusionar las ramas utilizo el comando

```
git merge nombre-de-la-rama-origen
```

• ¿Cómo crear un commit en Git?

Me aseguro de que los archivos que quiero incluir estén en el área de preparación (staging) con el comando:

```
git add .
```

Verifico qué archivos están en el área de preparación con el comando:

```
git status
```

Creo el commit con un mensaje descriptivo con el comando

```
git commit -m "Mensaje descriptivo del cambio"
```

• ¿Cómo enviar un commit a GitHub?

Me aseguro la rama que estoy utilizando
Si la rama es main utilizo el comando
`git push origin main`

• ¿Qué es un repositorio remoto?

Un repositorio remoto en Git es una versión de un proyecto que está alojada en Internet o en alguna red. Funciona como una copia central del código que permite la colaboración entre varios desarrolladores.

• ¿Cómo agregar un repositorio remoto a Git?

Abro la terminal

Me aseguro de estar en el directorio de mi proyecto Git local

Utilizo el comando

`git remote add nombre-remoto URL-del-repositorio`

Agrego el repositorio de como "origin" con el comando

`git remote add origin https://github.com/usuario/nombre-repositorio.git`

Verifico que el repositorio remoto se haya agregado correctamente con el comando

`git remote -v`

Para enviar los cambios al repositorio remoto utilizo el comando

`git push -u origin main`

El parámetro `-u` en el último comando establece la rama remota como la rama de seguimiento predeterminada, lo que significa que en futuras ocasiones puedo simplemente usar `git push` sin especificar el nombre del remoto y la rama.

• ¿Cómo empujar cambios a un repositorio remoto?

Guardo los cambios localmente con los comandos

`git add .`

`git commit -m "mensaje"`

Luego, empujo los cambios al repositorio remoto con el siguiente comando

`git push origin nombre de la rama`

• ¿Cómo tirar de cambios de un repositorio remoto?

Abro la terminal

Navego hasta el directorio de mi repositorio local

Ejecuto el comando

`git pull`

• ¿Qué es un fork de repositorio?

Es una copia independiente de un repositorio Git existente.

• ¿Cómo crear un fork de un repositorio?

Navego al repositorio que quiero hacer fork

Hago clic en el botón "Fork" que se encuentra en la esquina superior derecha de la página

Selecciono mi cuenta de usuario como destino del fork

Espero a que se complete el proceso de copia

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

Clono el fork a mi máquina local con el comando

```
git clone https://github.com/tu-usuario/nombre-repositorio.git
```

Creo una nueva rama para mis cambios con los siguientes comandos

```
git checkout -b nombre-de-rama
```

Creo una nueva rama para los cambios con el comando

```
git checkout -b nombre-de-rama
```

Realizo los cambios necesarios en el código y hago commit de los cambios con los comandos

```
git add .
```

```
git commit -m "Descripción de los cambios realizados"
```

Subo la rama a mi fork con el comando

```
git push origin nombre-de-la-rama
```

En mi fork en la plataforma GitHub busco el botón "New pull request" / "Create pull request"

Selecciono el repositorio original como base y mi rama como "compare"

Añado un título descriptivo y explico los cambios en la descripción

Hago clic en "Create pull request"

- ¿Cómo aceptar una solicitud de extracción?

Navego a la sección de Pull Requests en mi repositorio de GitHub

Hago clic en la pestaña "Pull requests"

Selecciono el pull request que deseo revisar y aceptar

Reviso el código modificado en la pestaña "Files changed"/"Changes"

Leo la descripción y los comentarios

Agrego mis comentarios si es necesario

Para aprobar la revisión hago clic en "Review changes"

Selecciono "Approve"

Añado comentarios si lo deseo

Hago clic en "Submit review"

Para fusionar el pull request hago clic en el botón "Merge pull request"/"Merge"

Selecciono el método de fusión si hay opciones (merge commit, squash, rebase)

Confirmo la acción

Cierro el pull request (normalmente se hace automáticamente tras la fusión)

- ¿Qué es una etiqueta en Git?

Es un marcador o referencia que se asigna a un punto específico en la historia del repositorio (normalmente a un commit)

- ¿Cómo crear una etiqueta en Git?

Para crear una etiqueta se utiliza el comando `git tag v1.0`

- ¿Cómo enviar una etiqueta a GitHub?

Para enviar una etiqueta a GitHub uso el comando

```
git push origin v1.0
```

- ¿Qué es un historial de Git?

El historial de Git es un registro cronológico de todos los cambios realizados en un repositorio.

• ¿Cómo ver el historial de Git?

Para ver el historial de Git utilizo el comando `git log`

• ¿Cómo buscar en el historial de Git?

Para buscar en el historial de Git, puedo utilizar distintos comandos en función de lo que quiero encontrar

Ej. 1: Buscar commits donde se agregó o eliminó texto específico

`git log -S"texto a buscar"`

Ej. 2: Buscar commits donde se modificaron líneas que coinciden con un patrón

`git log -G"expresión regular"`

• ¿Cómo borrar el historial de Git?

Para eliminar todo el historial de Git utilizo el siguiente comando `rm -rf .git`

• ¿Qué es un repositorio privado en GitHub?

Es un repositorio al que solo puedo acceder yo y las personas a las que le de permiso explícito. A diferencia de los repositorios públicos, los privados **no son visibles para el público** y requieren autenticación para acceder.

• ¿Cómo crear un repositorio privado en GitHub?

Inicio sesión en GitHub con mi cuenta

En la página principal de GitHub, hago clic en el icono de "+" en la esquina superior derecha y selecciono **"New repository"** (Nuevo repositorio)

Elige un nombre para el repositorio.

Agrego una breve descripción sobre lo que contiene el repositorio (esto es opcional).

Selecciono la opción **"Private"** para hacerlo privado.

Hago clic en **"Create repository"** para crear el repositorio privado.

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

Inicio sesión en mi cuenta de GitHub y voy al repositorio privado al que quiero invitar a alguien.

En la página principal del repositorio, hago clic en el botón **"Settings"** (Configuración) que se encuentra en la parte superior, justo debajo del nombre del repositorio.

En el menú de la izquierda, selecciono **"Manage access"** (Gestionar acceso). Esta opción me permite ver y controlar quién tiene acceso al repositorio.

Hago clic en el botón **"Invite a collaborator"** (Invitar a un colaborador) que aparece en la parte superior de la página de acceso.

Aparece una ventana donde puedo escribir el **nombre de usuario de GitHub** o la **dirección de correo electrónico** de la persona que quiero invitar.

Selecciono el nombre de usuario correcto de la lista que aparece.

Elijo el tipo de acceso que le daré a esa persona:

- **Read** (Lectura): Puede ver el código, pero no hacer cambios.

- **Write** (Escritura): Puede ver y modificar el código.
- **Admin** (Administración): Puede gestionar la configuración del repositorio, agregar colaboradores, etc.

Hago clic en el botón **"Add"** para enviar la invitación.

• ¿Qué es un repositorio público en GitHub?

Es un repositorio el cual puede ser visto por cualquier persona

• ¿Cómo crear un repositorio público en GitHub?

Inicio sesión en GitHub y accedo a mi cuenta.

Busca el botón verde que dice "New" o "Nuevo" en la parte superior de la página

Escribo un nombre para el repositorio.

Agrego una breve descripción de lo que trata el proyecto.

Selecciono la opción "Public" para que el repositorio sea público.

Hago clic en el botón verde que dice "Create repository" o "Crear repositorio".

• ¿Cómo compartir un repositorio público en GitHub?

Ingresando al repositorio y compartiendo el link

2) Realizar la siguiente actividad:

• Crear un repositorio.

- o Dale un nombre al repositorio.
- o Elije el repositorio sea público.
- o Inicializa el repositorio con un archivo.

github.com/new

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 * 2005Nico / Repository name * Repo_TP_02
Repo_TP_02 is available.

Great repository names are short and memorable. Need inspiration? How about [expert-fiesta](#) ?

Description (optional)
Repositorio TP 02

☒ 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.](#)

This will set [main](#) as the default branch. Change the default name in your [settings](#).

🔔 You are creating a public repository in your personal account.

[Create repository](#)

• Agregando un Archivo

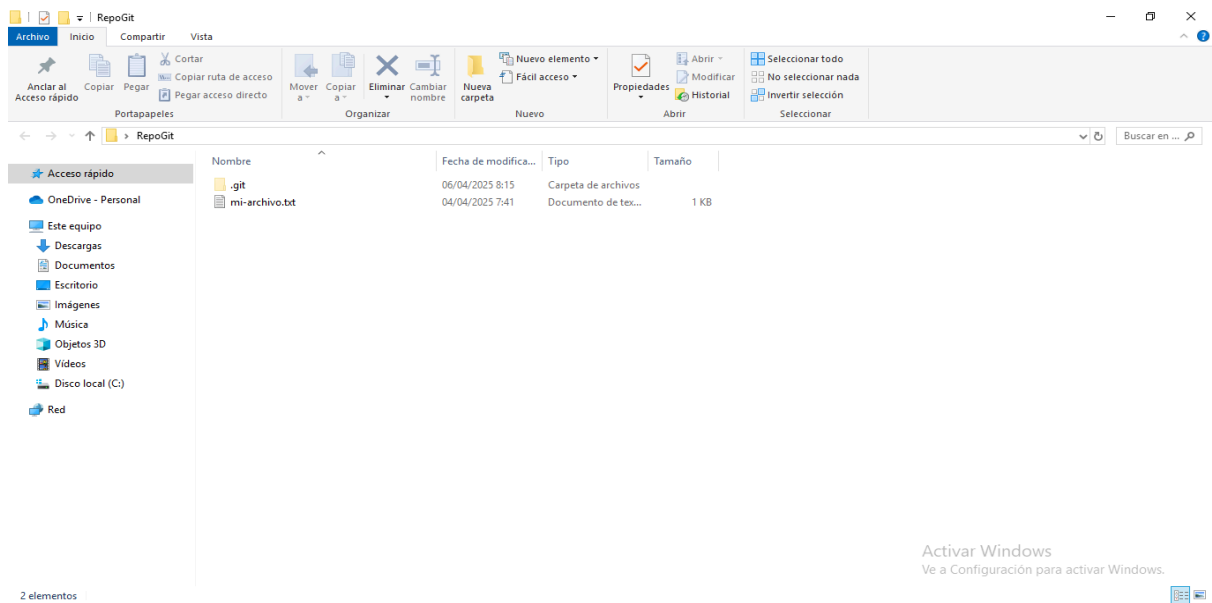
o Crea un archivo simple, por ejemplo, "mi-archivo.txt".

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

```
MINGW64/c/Users/Marcelo/Desktop/Repo_TP-02
Marcelo@Notebook_Marcelo MINGW64 ~/Desktop/Repo_TP-02 (master)
$ git add .

Marcelo@Notebook_Marcelo MINGW64 ~/Desktop/Repo_TP-02 (master)
$ git commit -m "Agregando mi-archivo.txt"
[master (root-commit) 3cedf34] Agregando mi-archivo.txt
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 mi-archivo.txt

Marcelo@Notebook_Marcelo MINGW64 ~/Desktop/Repo_TP-02 (master)
$ |
```

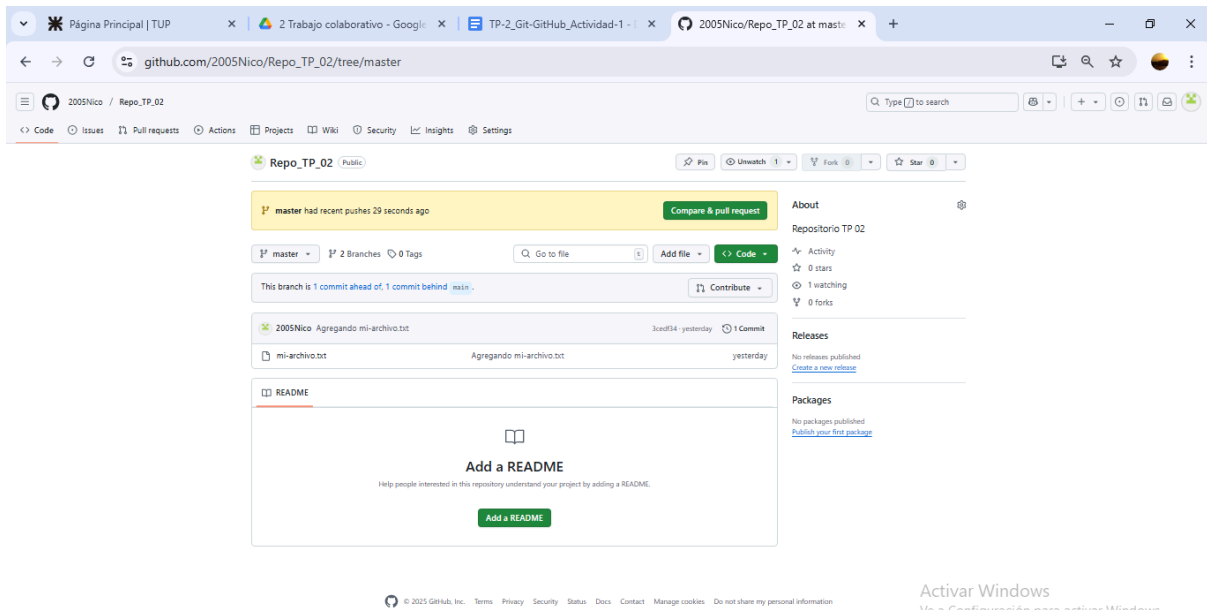


o Sube los cambios al repositorio en GitHub con `git push origin main` (o el nombre de la rama correspondiente).

```
MINGW64/c/Users/Marcelo/Desktop/Repo_TP-02

Marcelo@Notebook_Marcelo MINGW64 ~/Desktop/Repo_TP-02 (master)
$ git push -u origin master
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Writing objects: 100% (3/3), 226 bytes | 56.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
remote:
remote: Create a pull request for 'master' on GitHub by visiting:
remote:   https://github.com/2005Nico/Repo_TP-02/pull/new/master
remote:
To https://github.com/2005Nico/Repo_TP-02.git
 * [new branch]      master -> master
branch 'master' set up to track 'origin/master'.

Marcelo@Notebook_Marcelo MINGW64 ~/Desktop/Repo_TP-02 (master)
$ |
```

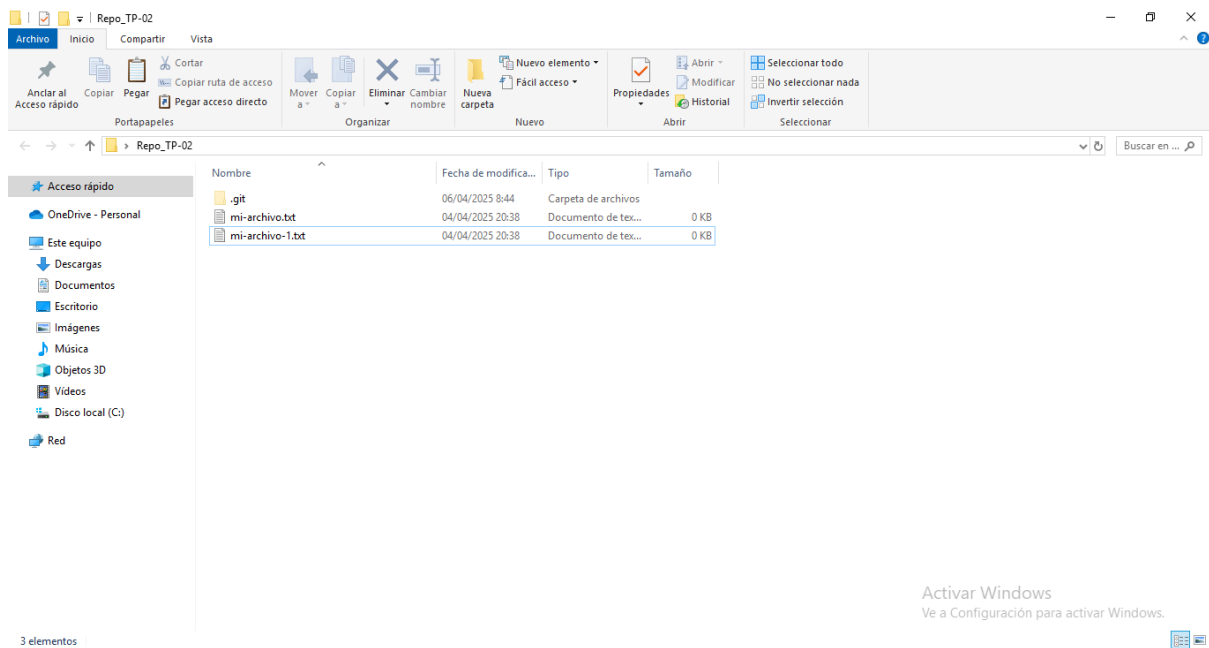


• Creando Branchs

o Crear una Branch



o Realizar cambios o agregar un archivo



```
MINGW64/c/Users/Marcelo/Desktop/Repo_TP-02

Marcelo@Notebook_Marcelo MINGW64 ~/Desktop/Repo_TP-02 (master)
$ git branch rama1

Marcelo@Notebook_Marcelo MINGW64 ~/Desktop/Repo_TP-02 (master)
$ git checkout rama1
Switched to branch 'rama1'

Marcelo@Notebook_Marcelo MINGW64 ~/Desktop/Repo_TP-02 (rama1)
$ git add .

Marcelo@Notebook_Marcelo MINGW64 ~/Desktop/Repo_TP-02 (rama1)
$ git commit -m "Creo la branch rama1 y agrego el archivo mi-archivo-1.txt"
[rama1 7b6d65e] Creo la branch rama1 y agrego el archivo mi-archivo-1.txt
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 mi-archivo-1.txt

Marcelo@Notebook_Marcelo MINGW64 ~/Desktop/Repo_TP-02 (rama1)
$ |
```

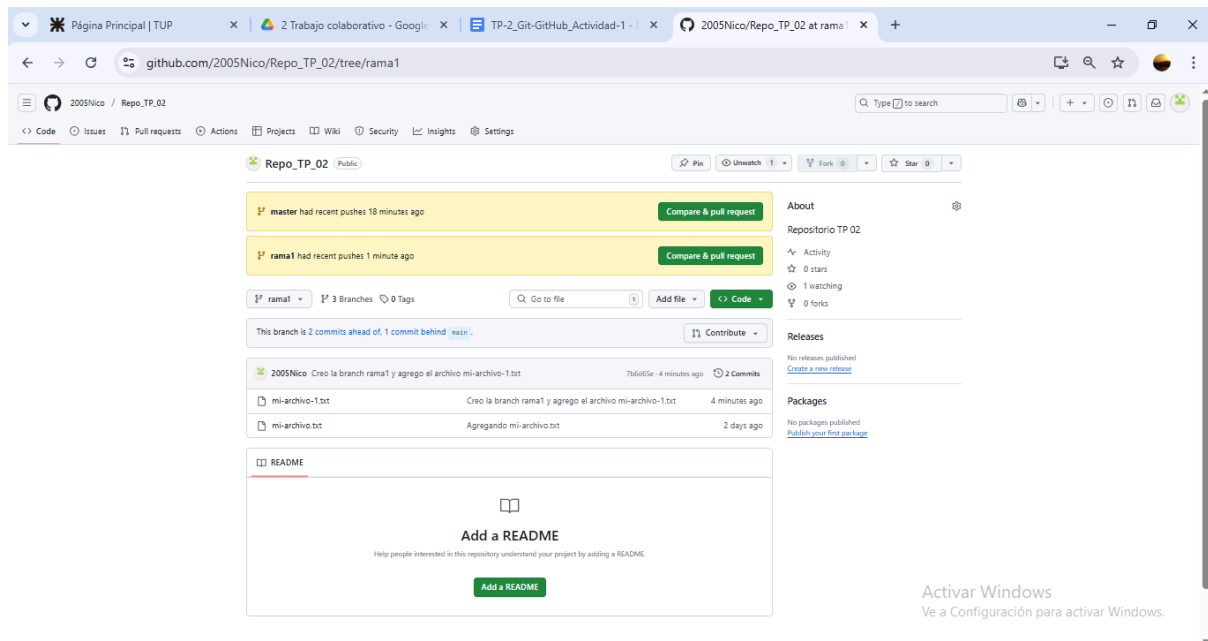
Activar Windows
Ve a Configuración para activar Windows.

o Subir la Branch

```
MINGW64/c/Users/Marcelo/Desktop/Repo_TP-02
Marcelo@Notebook_Marcelo MINGW64 ~/Desktop/Repo_TP-02 (rama1)
$ git push origin rama1
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Delta compression using up to 2 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (2/2), 278 bytes | 278.00 KiB/s, done.
Total 2 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
remote:
remote: Create a pull request for 'rama1' on GitHub by visiting:
remote:   https://github.com/2005Nico/Repo_TP_02/pull/new/rama1
remote:
To https://github.com/2005Nico/Repo_TP_02.git
 * [new branch]      rama1 -> rama1

Marcelo@Notebook_Marcelo MINGW64 ~/Desktop/Repo_TP-02 (rama1)
$
```

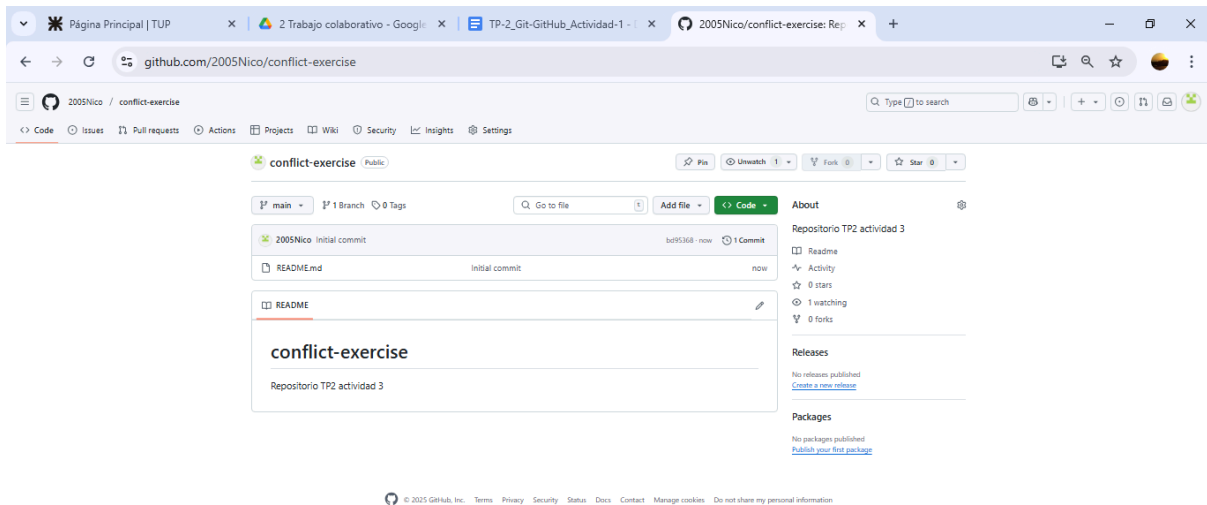
Activar Windows
Ve a Configuración para activar Windows.



3) Realizar la siguiente actividad:

Paso 1: Crear un repositorio en GitHub

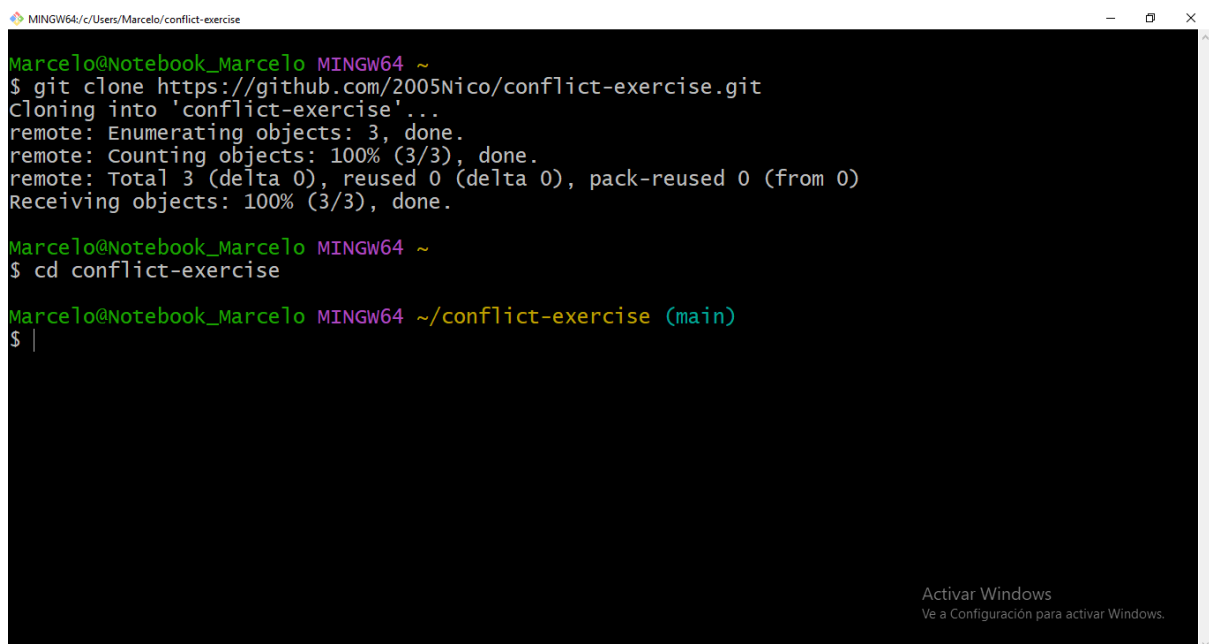
- Ve a GitHub e inicia sesión en tu cuenta.
- Haz clic en el botón "New" o "Create repository" para crear un nuevo repositorio.
- Asigna un nombre al repositorio, por ejemplo, conflict-exercise.
- Opcionalmente, añade una descripción.
- Marca la opción "Initialize this repository with a README".
- Haz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



Activar Windows
Ve a Configuración para activar Windows.

Paso 2: Clonar el repositorio a tu máquina local

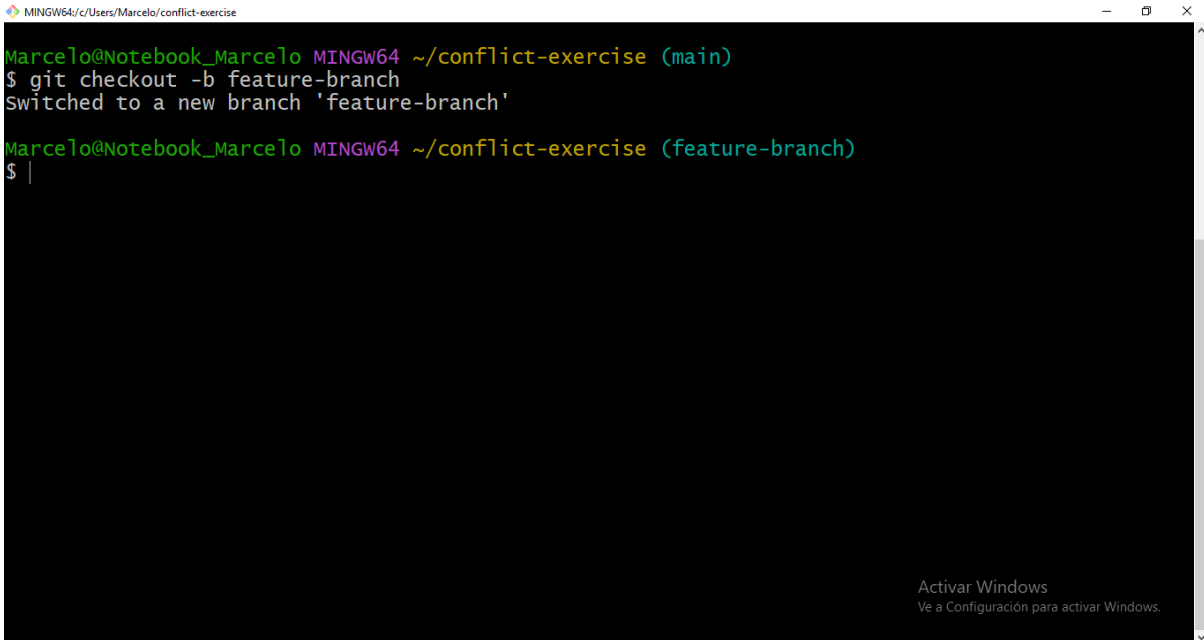
- Copia la URL del repositorio (usualmente algo como <https://github.com/tuusuario/conflict-exercise.git>).
- Abre la terminal o línea de comandos en tu máquina.
- Clona el repositorio usando el comando:
`git clone https://github.com/tuusuario/conflict-exercise.git`
- Entra en el directorio del repositorio: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



Paso 3: Crear una nueva rama y editar un archivo

- Crea una nueva rama llamada feature-branch:

git checkout -b feature-branch

A terminal window titled 'MINGW64/c/Users/Marcelo/conflict-exercise' showing the execution of 'git checkout -b feature-branch'. The prompt changes from '(main)' to '(feature-branch)'.

```
MINGW64/c/Users/Marcelo/conflict-exercise
Marcelo@Notebook_Marcelo MINGW64 ~/conflict-exercise (main)
$ git checkout -b feature-branch
Switched to a new branch 'feature-branch'

Marcelo@Notebook_Marcelo MINGW64 ~/conflict-exercise (feature-branch)
$ |
```

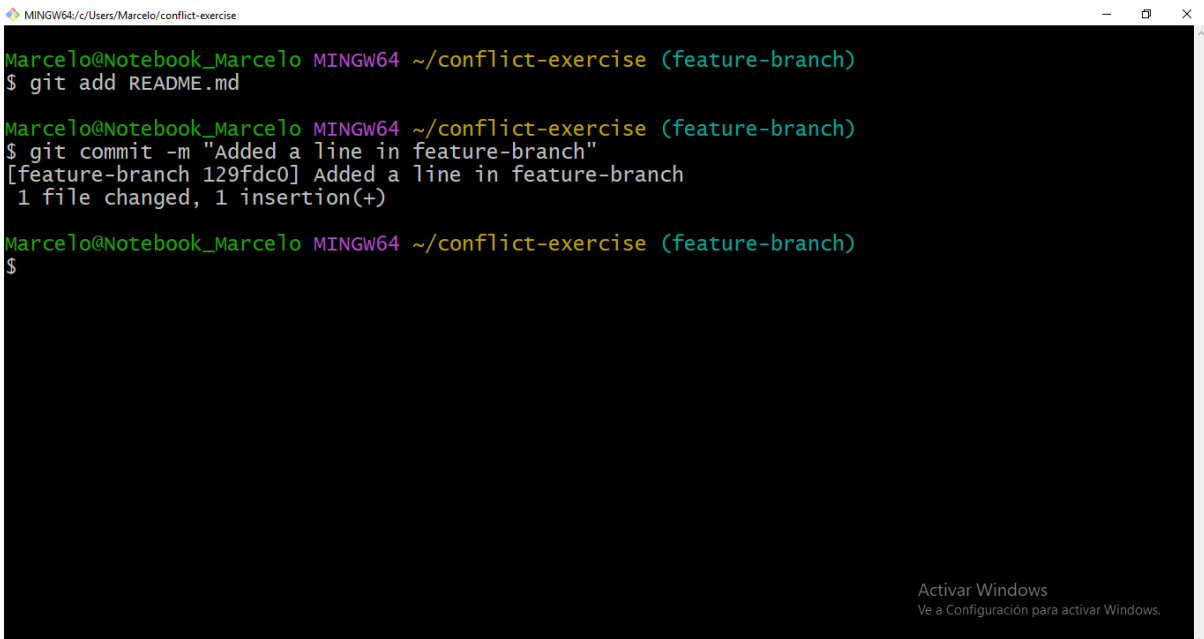
- Abre el archivo README.md en un editor de texto y añade una línea nueva, por ejemplo:

Este es un cambio en la feature branch.

- Guarda los cambios y haz un commit:

git add README.md

git commit -m "Added a line in feature-branch"

A terminal window showing the execution of 'git add README.md' and 'git commit -m "Added a line in feature-branch"'. The commit hash '129fdc0' is displayed.

```
MINGW64/c/Users/Marcelo/conflict-exercise
Marcelo@Notebook_Marcelo MINGW64 ~/conflict-exercise (feature-branch)
$ git add README.md

Marcelo@Notebook_Marcelo MINGW64 ~/conflict-exercise (feature-branch)
$ git commit -m "Added a line in feature-branch"
[feature-branch 129fdc0] Added a line in feature-branch
1 file changed, 1 insertion(+)

Marcelo@Notebook_Marcelo MINGW64 ~/conflict-exercise (feature-branch)
$
```

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

- Cambia de vuelta a la rama principal (main):

git checkout main

```
MINGW64/c/Users/Marcelo/conflict-exercise
Marcelo@Notebook_Marcelo MINGW64 ~/conflict-exercise (feature-branch)
$ git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.

Marcelo@Notebook_Marcelo MINGW64 ~/conflict-exercise (main)
$ |
```

Activar Windows
Ve a Configuración para activar Windows.

- Edita el archivo README.md de nuevo, añadiendo una línea diferente:
Este es un cambio en la main branch.
- Guarda los cambios y haz un commit:
git add README.md
git commit -m "Added a line in main branch"

```
MINGW64/c/Users/Marcelo/conflict-exercise
Marcelo@Notebook_Marcelo MINGW64 ~/conflict-exercise (main)
$ git add README.md

Marcelo@Notebook_Marcelo MINGW64 ~/conflict-exercise (main)
$ git commit -m "Added a line in main branch"
[main eaaca14] Added a line in main branch
1 file changed, 1 insertion(+)

Marcelo@Notebook_Marcelo MINGW64 ~/conflict-exercise (main)
$ |
```

Activar Windows
Ve a Configuración para activar Windows.

Paso 5: Hacer un merge y generar un conflicto

- Intenta hacer un merge de la feature-branch en la rama main:
git merge feature-branch
- Se generará un conflicto porque ambos cambios afectan la misma línea del archivo README.md.

```
MINGW64/c/Users/Marcelo/conflict-exercise
Marcelo@Notebook_Marcelo MINGW64 ~/conflict-exercise (main)
$ 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.

Marcelo@Notebook_Marcelo MINGW64 ~/conflict-exercise (main|MERGING)
$
```

Paso 6: Resolver el conflicto

- Abre el archivo README.md en tu editor de texto. Verás algo similar a esto:

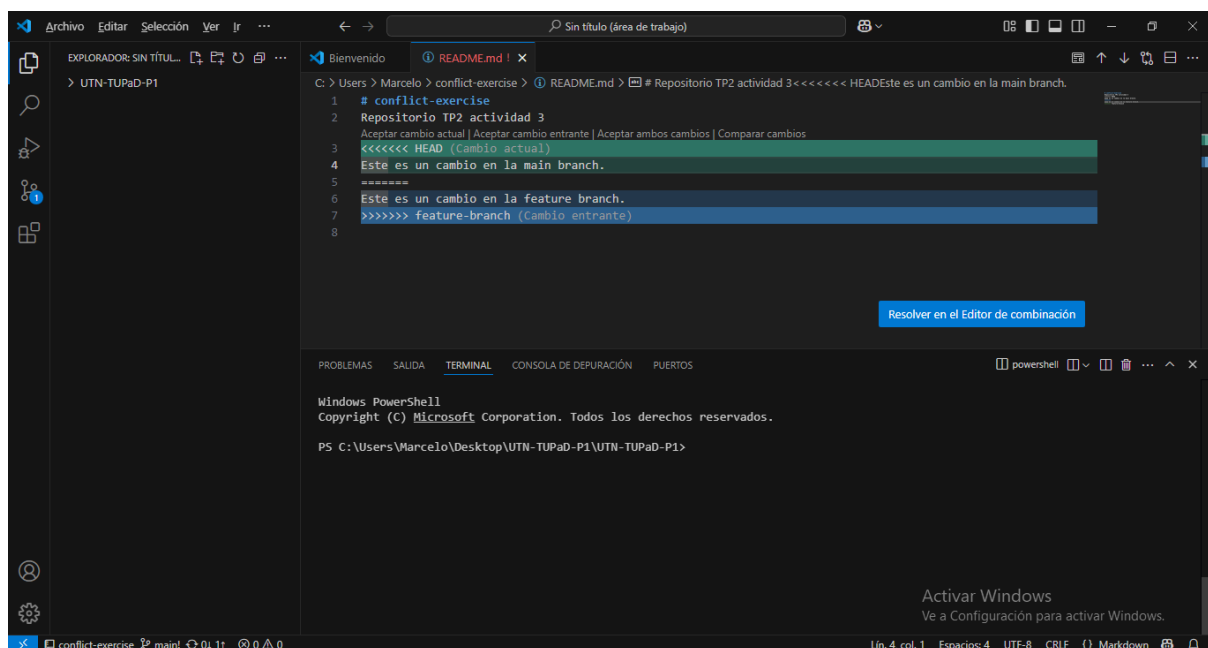
```
<<<<<<< HEAD
```

```
Este es un cambio en la main branch.
```

```
=====
```

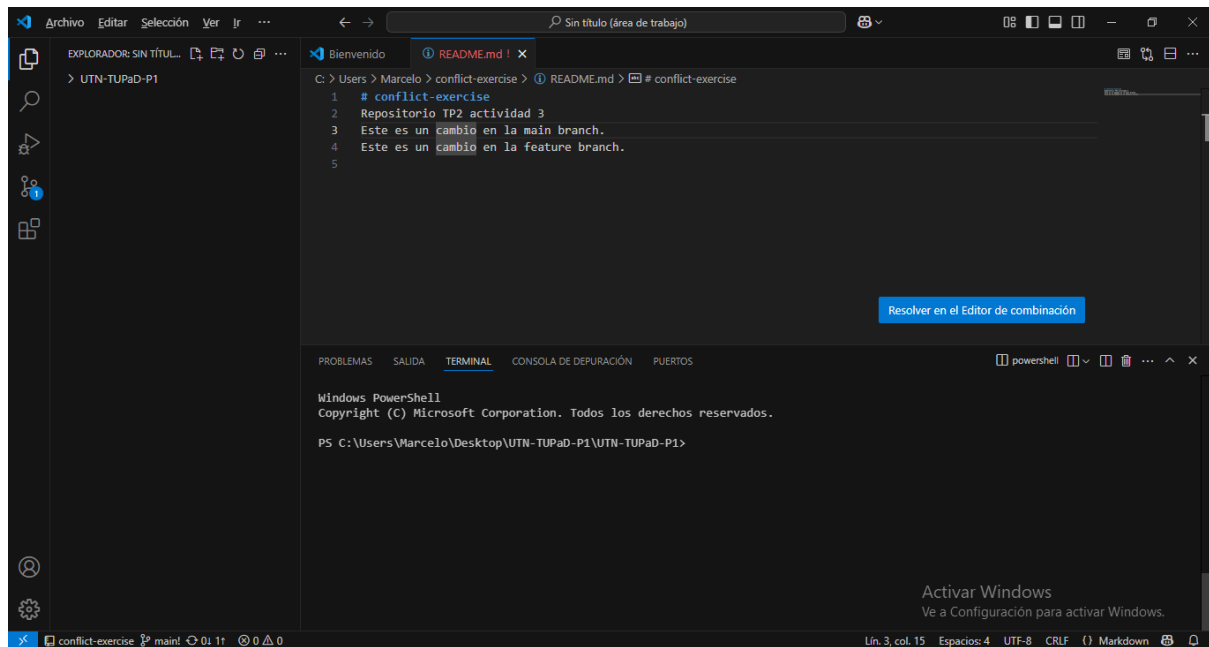
```
Este es un cambio en la feature branch.
```

```
>>>>>>> feature-branch
```



- Decide cómo resolver el conflicto. Puedes mantener ambos cambios, elegir uno de ellos, o fusionar los contenidos de alguna manera.
- 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

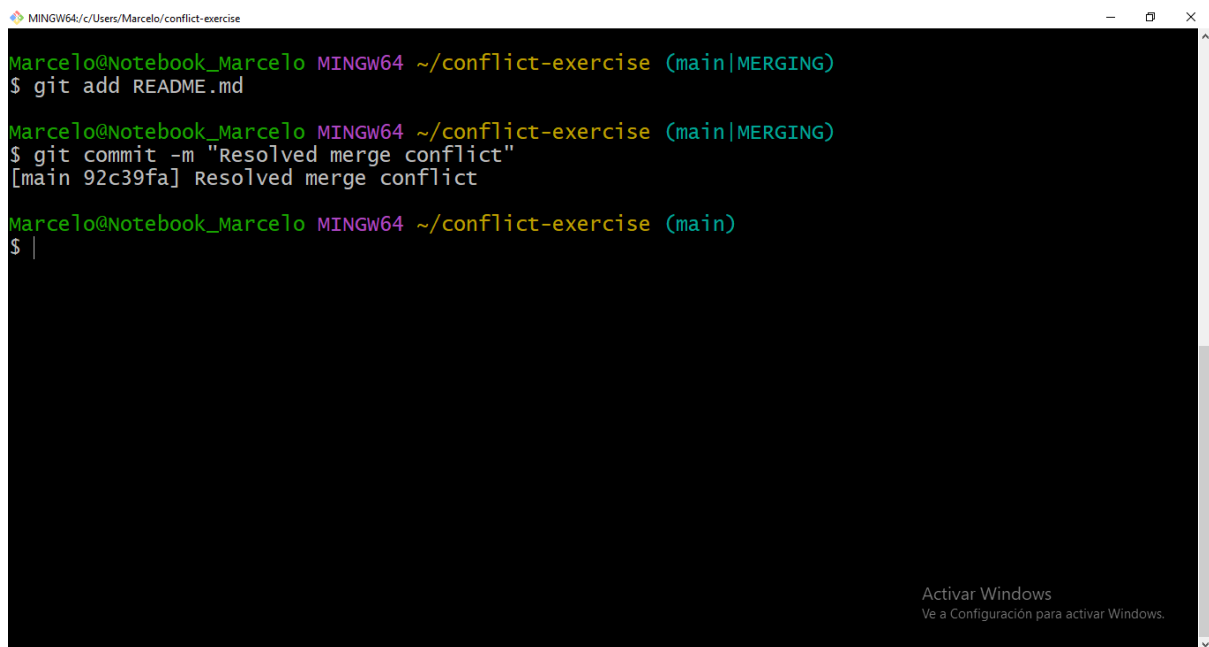
debe borrar la parte del texto que no se quiera dejar).



- Añade el archivo resuelto y completa el merge:

```
git add README.md
```

```
git commit -m "Resolved merge conflict"
```



Paso 7: Subir los cambios a GitHub

- Sube los cambios de la rama main al repositorio remoto en GitHub:

```
git push origin main
```

```
MINGW64/c/Users/Marcelo/conflict-exercise
Marcelo@Notebook_Marcelo MINGW64 ~/conflict-exercise (main)
$ git push origin main
Enumerating objects: 11, done.
Counting objects: 100% (11/11), done.
Delta compression using up to 2 threads
Compressing objects: 100% (6/6), done.
Writing objects: 100% (9/9), 828 bytes | 207.00 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/2005Nico/conflict-exercise.git
   bd95368..92c39fa  main -> main

Marcelo@Notebook_Marcelo MINGW64 ~/conflict-exercise (main)
$ |
```

- También sube la feature-branch si deseas:
git push origin feature-branch

```
MINGW64/c/Users/Marcelo/conflict-exercise
Marcelo@Notebook_Marcelo MINGW64 ~/conflict-exercise (main)
$ git push origin feature-branch
Everything up-to-date

Marcelo@Notebook_Marcelo MINGW64 ~/conflict-exercise (main)
$ |
```

Paso 8: Verificar en GitHub

- Ve a tu repositorio en GitHub y revisa el archivo README.md para confirmar que los cambios se han subido correctamente.
- Puedes revisar el historial de commits para ver el conflicto y su resolución.

2005Nico / conflict-exercise

2005Nico Resolved merge conflict 92:39fa - 9 minutes ago 4 Commits

README.md Resolved merge conflict 9 minutes ago

conflict-exercise

Repositorio TP2 actividad 3 Este es un cambio en la main branch. Este es un cambio en la feature branch.

Repositorio TP2 actividad 3

- Readme
- Activity
- 0 stars
- 1 watching
- 0 forks

Releases

No releases published
[Create a new release](#)

Packages

No packages published
[Publish your first package](#)

© 2025 GitHub, Inc. Terms Privacy Security Status Docs Contact Manage cookies Do not share my personal information

Activar Windows
Ve a Configuración para activar Windows.

conflict-exercise/README.md

2005Nico Resolved merge conflict 92:39fa - 10 minutes ago History

Preview Code Blame 4 lines (4 loc) - 125 bytes Code 55% faster with GitHub Copilot Raw

```
1 # conflict-exercise
2 Repositorio TP2 actividad 3
3 Este es un cambio en la main branch.
4 Este es un cambio en la feature branch.
```

Activar Windows
Ve a Configuración para activar Windows.