

Práctico 2: Git y GitHub

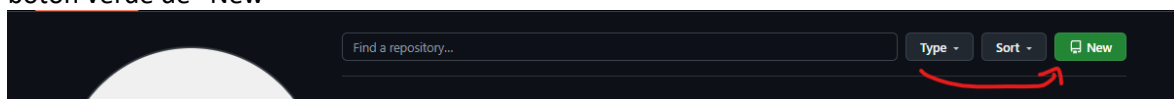
Fernando Joaquín Aguillón Basabilbaso

Actividades

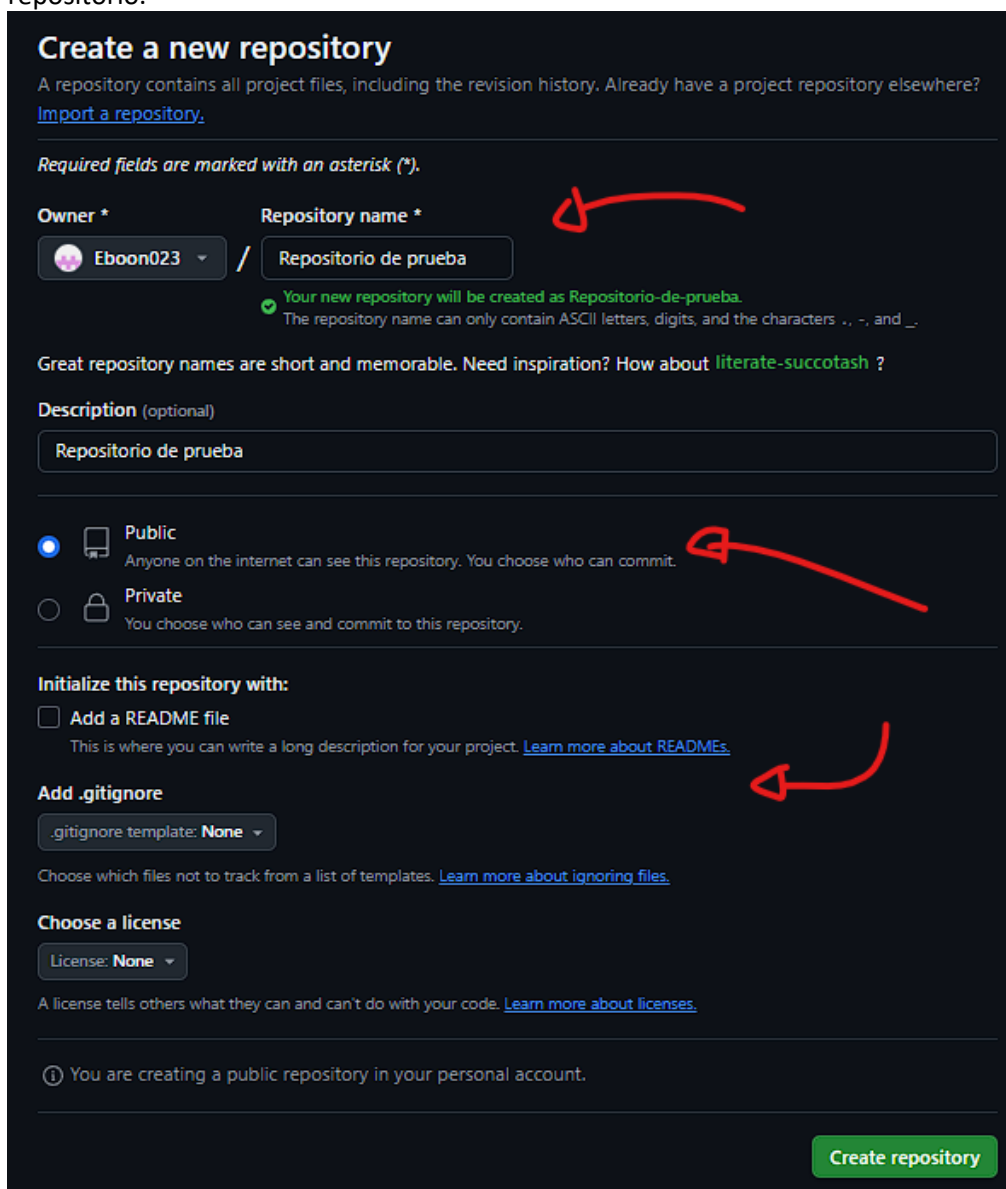
1)

-GitHub es una plataforma que permite a los desarrolladores almacenar, compartir y colaborar en proyectos de código. Se basa en el sistema de control de versiones Git

-Primero para crear un repositorio en GitHub, es necesario tener una cuenta creada. Una vez listo ese paso, es necesario ir al icono de tu cuenta y seleccionar "Your repositories". Luego darle al botón verde de "New"



Ahora nos aparecerá una pagina donde debemos completar las preferencias de nuestro repositorio.



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 * Eboon023 / **Repository name *** Repositorio de prueba

✓ Your new repository will be created as Repositorio-de-prueba.
The repository name can only contain ASCII letters, digits, and the characters ., -, and _.

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

Description (optional)

Repositorio de prueba

☒ **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.](#)

① You are creating a public repository in your personal account.

Create repository

Y listo, ya tenemos un repositorio en GitHub, ahora solo queda llenarlo con nuestros archivos

-Para crear una rama nueva en Git, se debe escribir el comando “git checkout (Nombre de la rama nueva)” desde la terminal del proyecto.

-Para cambiar a una rama ya creada en Git se necesita ejecutar el comando “git checkout -b ((Nombre de la rama))” desde la terminal del proyecto.

-Para fusionar ramas en git primero se necesita estar en la rama que desea usar como base en la fusión, ósea la rama que absorba los cambios de la otra rama.
Luego se debe ejecutar el comando “git merge (Nombre de la rama a absorber)”

-Para hacer un commit en Git es necesario primero ejecutar el comando “git add (Nombre del archivo modificado)” o “git add .(para seleccionar todos los archivos)” para guardarlos en un área de preparación. Luego se ejecuta el comando “git commit -m “(Mensaje)””, donde es necesario dejar un mensaje para reconocer la modificación en el historial.

-Un repositorio remoto es una copia de un proyecto que se almacena en un servidor remoto, ya sea en internet o en una red. Se puede compartir entre varios miembros de un equipo.

-Primero se debe crear un vinculo entre el repositorio local y remoto con el comando “git remote add (nombre del remoto) (link)”. Para verificar la conectividad, porque en algunos casos se necesita de iniciar sesión, se usa el comando “git remote show (nombre del remoto)”.
Para pasar al local los archivos del remoto se usa el comando “git pull (nombre del remoto) (rama a seleccionar)”

-Primero es recomendable descargar los archivos mas actuales del repositorio remoto para evitar conflictos, después para empujar los cambios al remoto se usa el comando “git push (nombre del remoto) (nombre de la rama)”

-Para tirar de cambios de un repositorio remoto se usa el comando “git pull (nombre del remoto) (rama a seleccionar)”

-Hacer un fork de un repositorio es crear una copia de un repositorio en tu propio github, esto con tal de usar un repositorio ajeno como propio y modificarlo a gusto.

-Para realizar un fork, basta con dirigirnos a un repositorio no creado por nosotros y apretar en el botón “FORK”.

-Para solicitar una pull request, debes dirigirte a la rama que contiene los cambios en el repositorio, y hacer clic en "Comparar y solicitud de incorporación de cambios", elegir las ramas base y la de comparación, escribir un titulo y una descripción, y finalmente hacer clic en "Crear solicitud de incorporación de cambios".

-Al dueño del repositorio le llegara una solicitud para una solicitud de incorporación de cambios, allí puede ver los cambios, y si no genera un conflicto puede simplemente aceptarlos.

-Una etiqueta es una marca que se aplica a una confirmación de un proyecto. Sirve para identificar un punto importante en el historial del repositorio y para crear instantáneas del repositorio.

-Para crear una etiqueta en Git, se debe ejecutar el comando “git tag (Nombre de la etiqueta) (Rama a la cual se le asigna) -m (Mensaje de la etiqueta).

- Para enviar la etiqueta del repositorio local a GitHub se ejecuta el comando `"git push --tags"`.
- El historial de git es una manera de ver todos los commits que se realizaron en un proyecto.
- Para ver el historial de git se ejecuta el comando `"git log"`. Si el historial es muy extenso se puede agregar el comando `"-x"` donde x es el número de commits que desea ver.
- Se puede buscar de varias formas, donde algunas son:
 - Por palabra clave con `"git log -grep=(palabra clave)"`
 - Por autor con `"git log --author=(Nombre del autor)"`
 - Por rango de fechas con `"git log --since=(fecha inicial) --until=(fecha final)"`
- Para borrar el historial de Git se ejecuta el comando `"git reset (Nombre del Archivo)"`
- Un repositorio privado en github a diferencia de un público, es inaccesible para cualquier persona, solo el administrador del mismo puede dar acceso a este mismo.
- Para crear un repositorio privado solo basta con seleccionarlo como privado en la opción que te da al crear cualquier repositorio.
- Para invitar a alguien como colaborador a un proyecto privado se debe ingresar a las configuraciones de dicho repositorio, al apartado de "Colaboradores" y seleccionar el botón "Add People". Ahí se debe ingresar el nombre de GitHub del colaborador y seleccionar el nivel de permisos que se le dará.
- Un repositorio publico es aquel que esta disponible para que cualquier persona lo pueda ver y clonar, y hasta colaborar en el mismo.
- Para crear un repositorio publico solo basta con seleccionarlo como público(default) en la opción que te da al crear cualquier repositorio.
- Simplemente con enviar el link directo, basta para compartir el repositorio público.

2) Realizar la siguiente actividad:

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


Required fields are marked with an asterisk (*).


Owner * / Repository name *

✓ Prueba-TUP is available.

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

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.](#)

Choose a license

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](#)

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

```
Initial commit

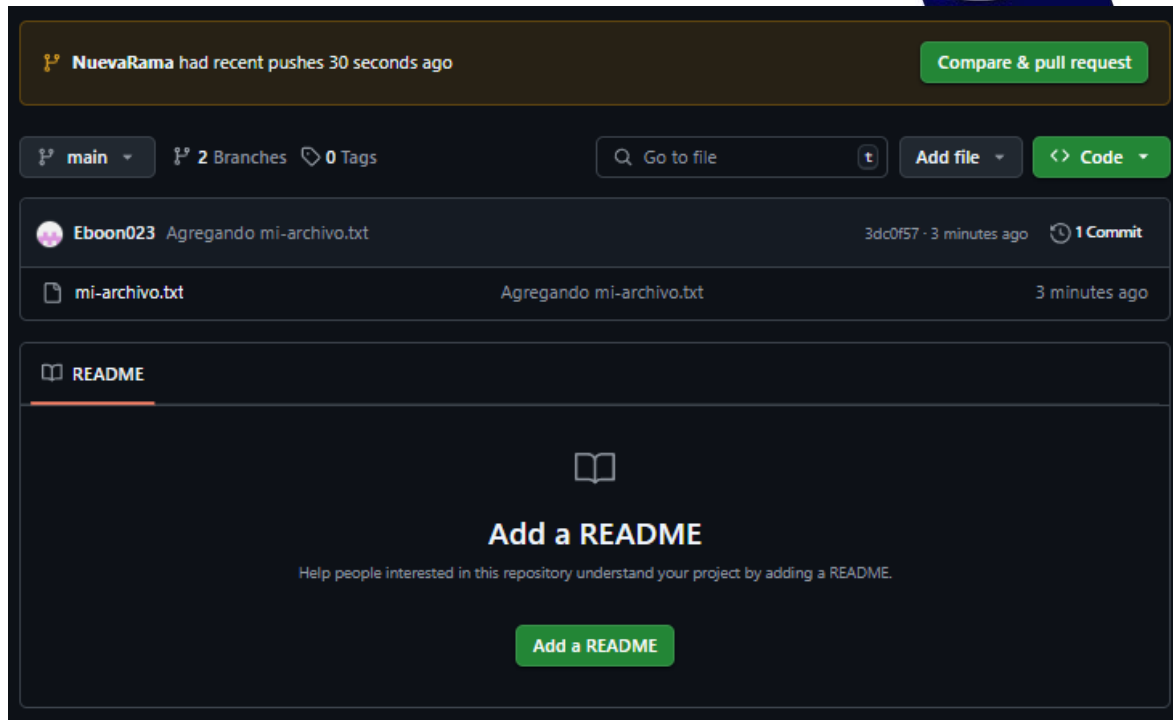
Untracked files:
  (use "git add <file>..." to include in what will be committed)
      mi-archivo.txt

nothing added to commit but untracked files present (use "git add" to track)
error: src refsPEC main does not match any
error: failed to push some refs to 'https://github.com/Eboon023/Prueba-TUP.git'
PS C:\Users\Fernando A\Documents\Prueba-TUP> git init
>> git add .
>> git commit -m "Agregando mi-archivo.txt"
>> git branch -M main
>> git remote add origin https://github.com/Eboon023/Prueba-TUP.git
>> git push -u origin main
Reinitialized existing Git repository in C:/Users/Fernando A/Documents/Prueba-TUP/.git/
[main (root-commit) 3dc0f57] Agregando mi-archivo.txt
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 mi-archivo.txt
error: remote origin already exists.
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Writing objects: 100% (3/3), 231 bytes | 115.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/Eboon023/Prueba-TUP.git
 * [new branch]      main -> main
branch 'main' set up to track 'origin/main'.
PS C:\Users\Fernando A\Documents\Prueba-TUP> 
```

Creando Branchs

- Crear una Branch
- Realizar cambios o agregar un archivo
- Subir la Branch

```
PS C:\Users\Fernando A\Documents\Prueba-TUP> git checkout -b NuevaRama
Switched to a new branch 'NuevaRama'
PS C:\Users\Fernando A\Documents\Prueba-TUP> git add .
PS C:\Users\Fernando A\Documents\Prueba-TUP> git commit -m "Subiendo archivo modificado a la nueva rama"
[NuevaRama 0af5623] Subiendo archivo modificado a la nueva rama
 1 file changed, 1 insertion(+)
PS C:\Users\Fernando A\Documents\Prueba-TUP> git push origin NuevaRama
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Writing objects: 100% (3/3), 305 bytes | 152.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
remote:
remote: Create a pull request for 'NuevaRama' on GitHub by visiting:
remote:   https://github.com/Eboon023/Prueba-TUP/pull/new/NuevaRama
remote:
To https://github.com/Eboon023/Prueba-TUP.git
 * [new branch]      NuevaRama -> NuevaRama
PS C:\Users\Fernando A\Documents\Prueba-TUP> 
```



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".

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 *

 Eboon023

Repository name *

/ conflict-exercise

✓ conflict-exercise is available.

Great repository names are short and memorable. Need inspiration? How about [bug-free-dollop](#) ?

Description (optional)

Ejercicio de practica sobre conflictos en git



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

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
```



```
MINGW64:/c/Users/Fernando A/Documents

Fernando A@DESKTOP-Fernando MINGW64 ~/Documents
$ git clone https://github.com/Eboon023/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.

Fernando A@DESKTOP-Fernando MINGW64 ~/Documents
$
```

Paso 3: Crear una nueva rama y editar un archivo

- Crea una nueva rama llamada feature-branch:

`git checkout -b feature-branch`

```
MINGW64:/c/Users/Fernando A/Documents/conflict-exercise

Fernando A@DESKTOP-Fernando MINGW64 ~/Documents
$ git clone https://github.com/Eboon023/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.

Fernando A@DESKTOP-Fernando MINGW64 ~/Documents
$ cd conflict-exercise

Fernando A@DESKTOP-Fernando MINGW64 ~/Documents/conflict-exercise (main)
$ git checkout -b feature-branch
Switched to a new branch 'feature-branch'

Fernando A@DESKTOP-Fernando MINGW64 ~/Documents/conflict-exercise (feature-branch)
$
```

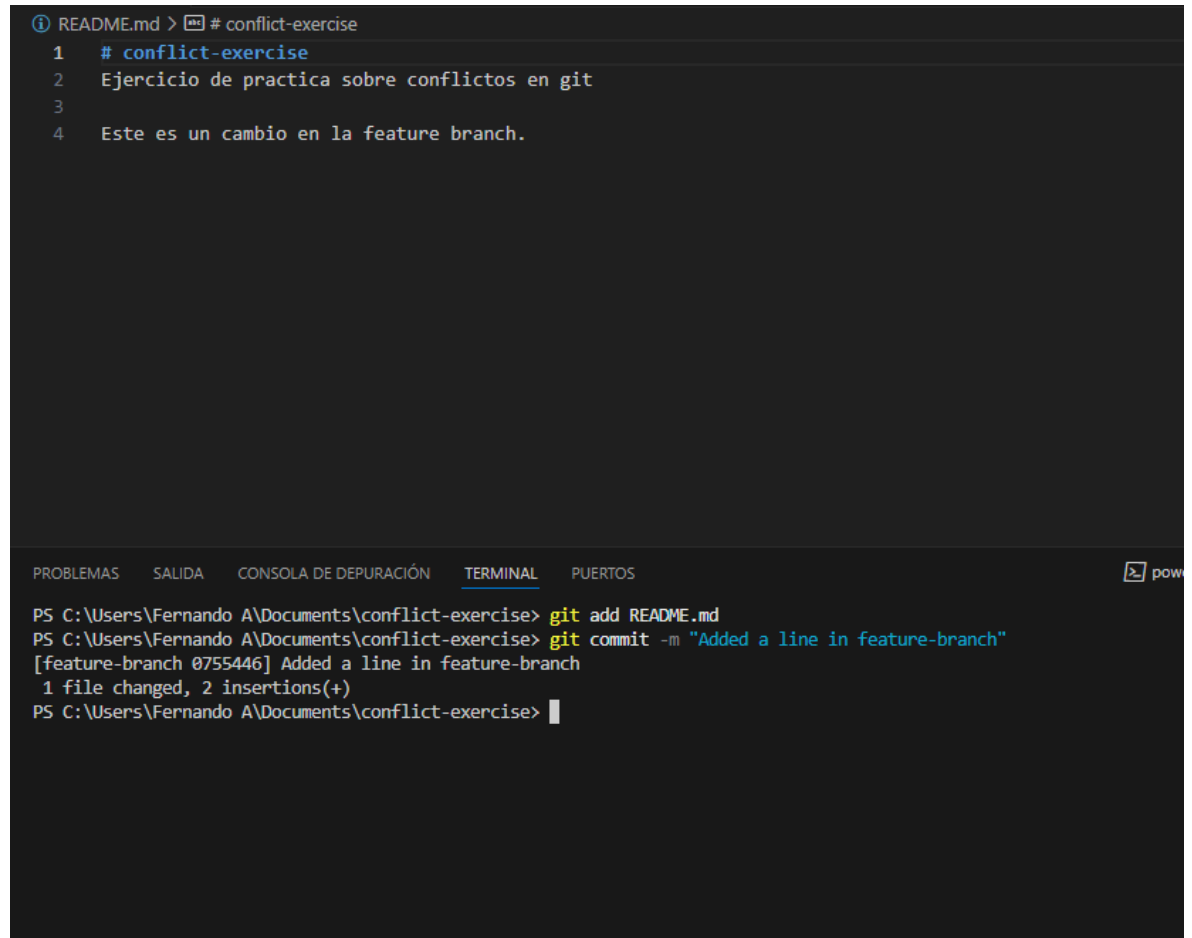
- Abre el archivo README.md en un editor de texto y añade una línea nueva, por ejemplo:

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"
```



The screenshot shows a code editor with a dark theme. The top part displays the content of a file named README.md, which includes a title, a description, and a line about a change in the feature branch. Below the editor, a terminal window is open, showing the execution of git commands: adding the README.md file and committing the changes with a specific message. The terminal output confirms the commit was successful.

```
① README.md > # conflict-exercise
1 # conflict-exercise
2 Ejercicio de practica sobre conflictos en git
3
4 Este es un cambio en la feature branch.

PROBLEMAS SALIDA CONSOLA DE DEPURACIÓN TERMINAL PUERTOS
PS C:\Users\Fernando A\Documents\conflict-exercise> git add README.md
PS C:\Users\Fernando A\Documents\conflict-exercise> git commit -m "Added a line in feature-branch"
[feature-branch 0755446] Added a line in feature-branch
1 file changed, 2 insertions(+)
PS C:\Users\Fernando A\Documents\conflict-exercise>
```

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

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

```
git checkout main
```

- 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"
```

① README.md > # conflict-exercise

```
1 # conflict-exercise
2 Ejercicio de practica sobre conflictos en git
3
4 Este es un cambio en la main branch.
```

PROBLEMAS SALIDA CONSOLA DE DEPURACIÓN TERMINAL PUERTOS

```
PS C:\Users\Fernando A\Documents\conflict-exercise> git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.
PS C:\Users\Fernando A\Documents\conflict-exercise> git add README.md
PS C:\Users\Fernando A\Documents\conflict-exercise> git commit -m "Added a line in main branch"
[main 80a1f78] Added a line in main branch
1 file changed, 2 insertions(+)
PS C:\Users\Fernando A\Documents\conflict-exercise> |
```

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.

```
① README.md ! ✕
① README.md > # <<<<<<< HEAD Este es un cambio en la main branch.
1 # conflict-exercise
2 Ejercicio de practica sobre conflictos en git
3
4 Aceptar cambio actual | Aceptar cambio entrante | Aceptar ambos cambios | Comparar cambios
4 <<<<<<< HEAD (Cambio actual)
5 Este es un cambio en la main branch.
6 =====
7 Este es un cambio en la feature branch.
8 >>>>>>> feature-branch (Cambio entrante)
9

PROBLEMAS  SALIDA  CONSOLA DE DEPURACIÓN  TERMINAL  PUERTOS

PS C:\Users\Fernando A\Documents\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\Fernando A\Documents\conflict-exercise> |
```

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"
```

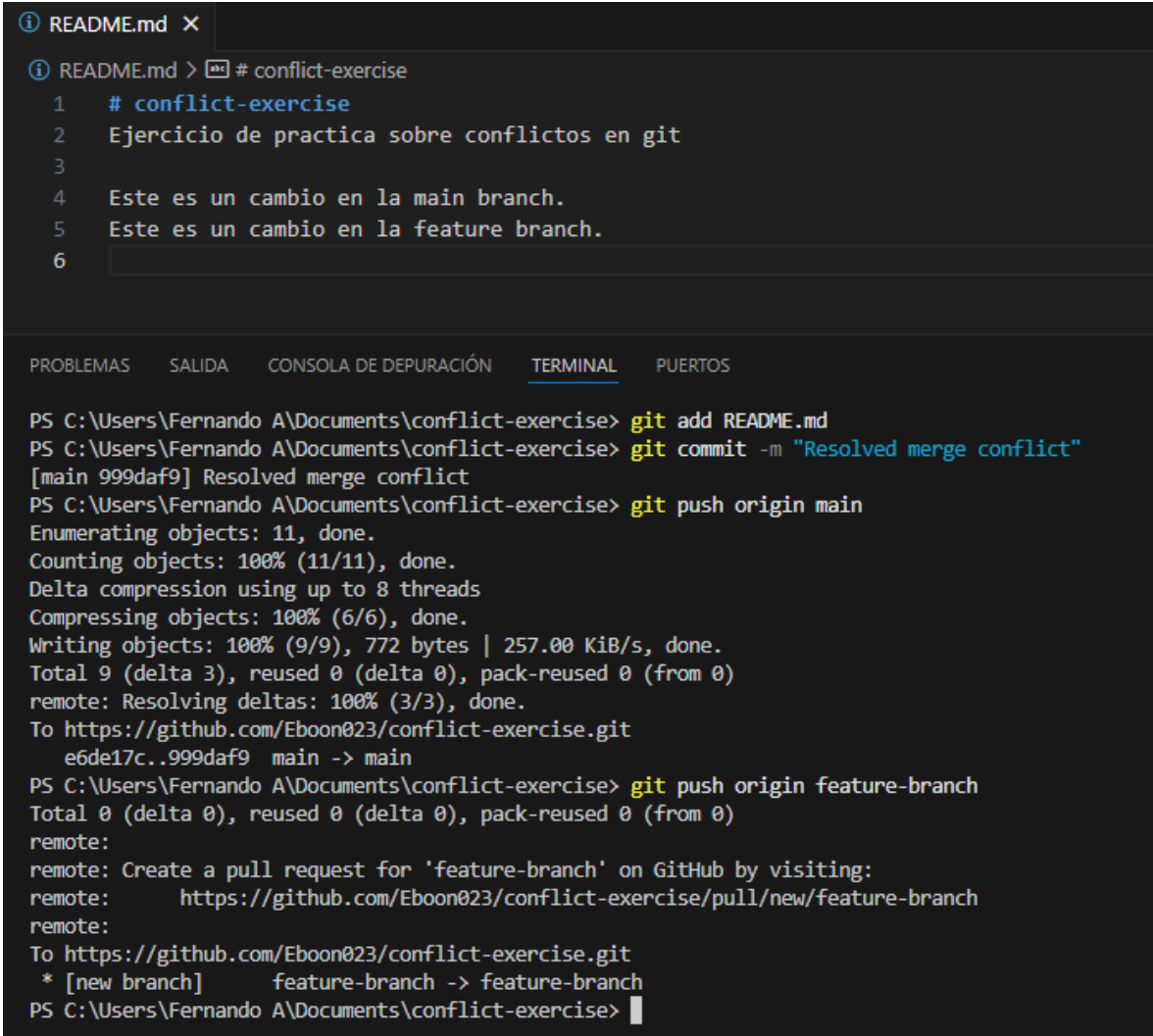
Paso 7: Subir los cambios a GitHub

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

```
git push origin main
```

- También sube la feature-branch si deseas:

```
git push origin feature-branch
```



```

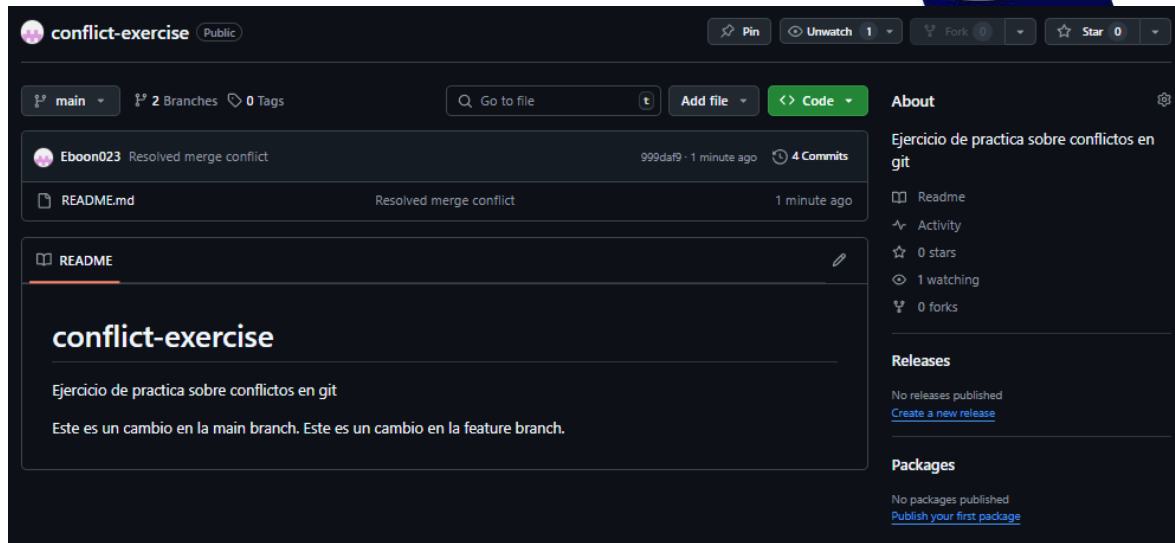
i README.md X
i README.md > # conflict-exercise
1 # conflict-exercise
2 Ejercicio de practica sobre conflictos en git
3
4 Este es un cambio en la main branch.
5 Este es un cambio en la feature branch.
6

PROBLEMAS  SALIDA  CONSOLA DE DEPURACIÓN  TERMINAL  PUERTOS

PS C:\Users\Fernando A\Documents\conflict-exercise> git add README.md
PS C:\Users\Fernando A\Documents\conflict-exercise> git commit -m "Resolved merge conflict"
[main 999daf9] Resolved merge conflict
PS C:\Users\Fernando A\Documents\conflict-exercise> git push origin main
Enumerating objects: 11, done.
Counting objects: 100% (11/11), done.
Delta compression using up to 8 threads
Compressing objects: 100% (6/6), done.
Writing objects: 100% (9/9), 772 bytes | 257.00 KiB/s, done.
Total 9 (delta 3), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (3/3), done.
To https://github.com/Eboon023/conflict-exercise.git
 e6de17c..999daf9  main -> main
PS C:\Users\Fernando A\Documents\conflict-exercise> 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/Eboon023/conflict-exercise/pull/new/feature-branch
remote:
To https://github.com/Eboon023/conflict-exercise.git
 * [new branch]      feature-branch -> feature-branch
PS C:\Users\Fernando A\Documents\conflict-exercise>
```

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.



The screenshot shows the GitHub interface for a repository named 'conflict-exercise'. The repository is public and has 2 branches and 0 tags. The main branch is selected. The repository has 4 commits, with the latest commit by 'Eboon023' resolving a merge conflict 1 minute ago. The README file is highlighted, showing the title 'conflict-exercise' and the description 'Ejercicio de practica sobre conflictos en git'. The README content states: 'Este es un cambio en la main branch. Este es un cambio en la feature branch.' The right sidebar shows the repository's statistics: 0 stars, 1 watching, and 0 forks. It also includes sections for Releases and Packages, both of which are currently empty.

conflict-exercise Public

main 2 Branches 0 Tags

Go to file Add file Code

Eboon023 Resolved merge conflict 999daf9 · 1 minute ago 4 Commits

README.md Resolved merge conflict 1 minute ago

README

conflict-exercise

Ejercicio de practica sobre conflictos en git

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

About

Ejercicio de practica sobre conflictos en git

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

Releases

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

Packages

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