

Práctico 2: Git y GitHub

Actividades

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

- ¿Qué es GitHub?

Es una plataforma de desarrollo que permite llevar un control y registro del ciclo de desarrollo de un proyecto, a través de un control de versiones.

- ¿Cómo crear un repositorio en GitHub?

Desde la web directa:

Hacer clic en el botón "+" en la esquina superior derecha y seleccionar "New repository". Ingresar un nombre para el repositorio. Elegir visibilidad (pública o privada). Hacer clic en "Create repository".

Desde terminal Git:

Iniciar un repositorio local en una carpeta que elijamos: `git init`

Agregar archivos: `git add .`

Hacer el primer commit: `git commit -m "Primer commit"` (Dentro de las comillas iría la descripción de lo hecho)

Conectar con GitHub: `git remote add origin URL_DEL_REPOSITORIO`

Subir los cambios: `git push -u origin main`

- ¿Cómo crear una rama en Git?

`git branch NOMBRE`

- ¿Cómo cambiar a una rama en Git?

`git checkout NOMBRE`

- ¿Cómo fusionar ramas en Git?

`git merge NOMBRE`

- ¿Cómo crear un commit en Git?

`git commit -m "MENSAJE"`

- ¿Cómo enviar un commit a GitHub?

`git push origin NOMBRE (de la rama)`

- ¿Qué es un repositorio remoto?

Es una versión del proyecto alojada en un servidor externo, como GitHub, que permite compartir y sincronizar cambios entre diferentes colaboradores o dispositivos.

- ¿Cómo agregar un repositorio remoto a Git?

`git remote add origin URL_DEL_REPOSITORIO`

- ¿Cómo empujar cambios a un repositorio remoto?

`git push origin NOMBRE (de la rama)`

- ¿Cómo tirar de cambios de un repositorio remoto?

`git pull origin NOMBRE_DE_LA_RAMA`

- ¿Qué es un fork de repositorio?

Un fork es una función para realizar una copia de un repositorio en GitHub y que se almacena en tu nuestra cuenta personal, permitiéndote realizar cambios sin afectar el repositorio original.

- ¿Cómo crear un fork de un repositorio?

Hacer clic en el botón Fork en la esquina superior derecha, del repositorio que queramos copiar. Se creará una copia del repositorio en mi cuenta.

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

Ir al repositorio y hacer clic en Compare & pull request . Selecciona las ramas, escribir un título y descripción para la solicitud, hacer clic en Create pull request

- ¿Cómo aceptar una solicitud de extracción?

Ir al repositorio original en GitHub, en la pestaña Pull requests , seleccionar la solicitud y revisar los cambios propuestos. Si todo está bien, hacer clic en Merge pull request para aceptarla.

- ¿Qué es un etiqueta en Git?

Una etiqueta en Git es un marcador o referencia estática a un punto específico en la historia del repositorio, generalmente usado para marcar versiones de lanzamiento.

- ¿Cómo crear una etiqueta en Git?

`git tag NOMBRE_DE_LA_ETIQUETA`

- ¿Cómo enviar una etiqueta a GitHub?

`git push origin NOMBRE_DE_LA_ETIQUETA`

- ¿Qué es un historial de Git?

El historial de Git es un registro de todos los cambios realizados en un repositorio, mostrando commits, ramas, fusiones y etiquetas.

- ¿Cómo ver el historial de Git?

`Git log`

- ¿Cómo buscar en el historial de Git?

`git log --grep="TEXTO_A_BUSCAR"`

- ¿Cómo borrar el historial de Git?

`rm -rf .git`

- ¿Qué es un repositorio privado en GitHub?

No es visible para el público, lo que lo hace ideal para proyectos confidenciales o de uso interno.

- ¿Cómo crear un repositorio privado en GitHub?

Seleccionar la opción Private y hacer clic en Create repository,

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

Ir al repositorio en GitHub, hacer clic en Settings (arriba a la derecha), seleccionar Collaborators and teams en el menú lateral, hacer clic en Add people y escribir el nombre o correo del usuario, elegir su nivel de acceso y enviar la invitación

- ¿Qué es un repositorio público en GitHub?

Es un repositorio visible para cualquier persona en internet. Cualquiera puede ver, clonar y/o contribuir (según los permisos) al repositorio, lo que lo hace ideal para proyectos abiertos o de código abierto.

- ¿Cómo crear un repositorio público en GitHub?

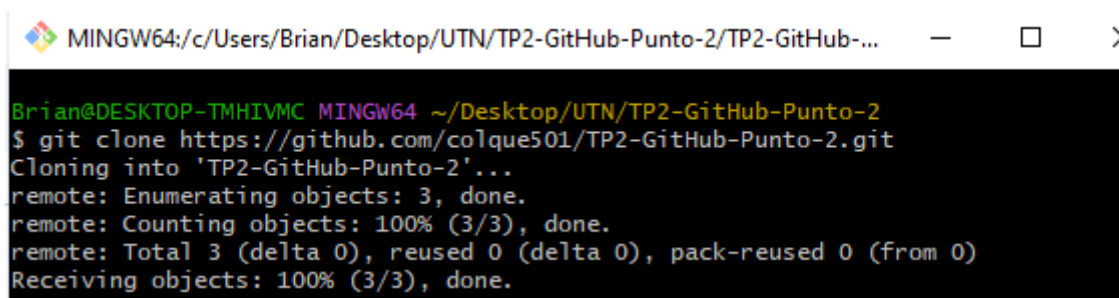
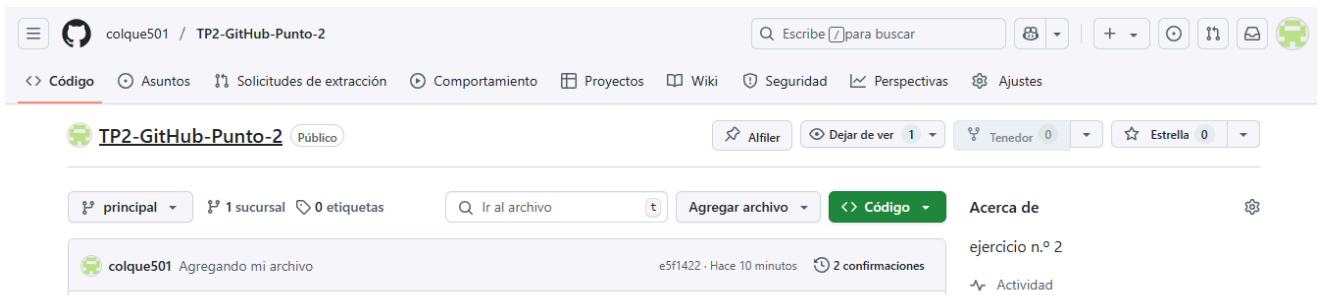
Hacer clic en New repository en GitHub, ingresar un nombre para el repositorio, seleccionar la opción Public (por defecto), hacer clic en Create repository.

- ¿Cómo compartir un repositorio público en GitHub?

Ir al repositorio público en GitHub, copia la URL del repositorio (disponible en la barra de direcciones) y compartir la URL con otras personas.

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.

Link: <https://github.com/colque501/TP2-GitHub-Punto-2.git>




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

```
Brian@DESKTOP-TMHIVMC MINGW64 ~/Desktop/UTN/TP2-GitHub-Punto-2
$ cd TP2-GitHub-Punto-2


Brian@DESKTOP-TMHIVMC MINGW64 ~/Desktop/UTN/TP2-GitHub-Punto-2/TP2-GitHub-Punto-2 (main)
$ git add .



Brian@DESKTOP-TMHIVMC MINGW64 ~/Desktop/UTN/TP2-GitHub-Punto-2/TP2-GitHub-Punto-2 (main)
$ git commit -m "Agregando mi archivo"
[main e5f1422] Agregando mi archivo
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 mi-archivo2.txt

Brian@DESKTOP-TMHIVMC MINGW64 ~/Desktop/UTN/TP2-GitHub-Punto-2/TP2-GitHub-Punto-2 (main)
$ 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), 287 bytes | 71.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/colque501/TP2-GitHub-Punto-2.git
42ccfd8..e5f1422 main -> main
```

 TP2-GitHub-Punto-2 Público Alfiler Dejar de ver 1

principal 1 sucursal 0 etiquetas Agregar archivo Código

 colque501 Agregando mi archivo e5f1422 · 14 minutos ago 2 confirmaciones

 mi-archivo.txt	Crear mi-archivo.txt	25 minutos ago
 mi-archivo2.txt	Agregando mi archivo	14 minutos ago

Creando Branches o Crear una Branch o Realizar cambios o agregar un archivo o Subir la Branch

```
Brian@DESKTOP-TMHIVMC MINGW64 ~/Desktop/UTN/TP2-GitHub-Punto-2/TP2-GitHub-Punto-2 (main)
$ git branch NuevaRama

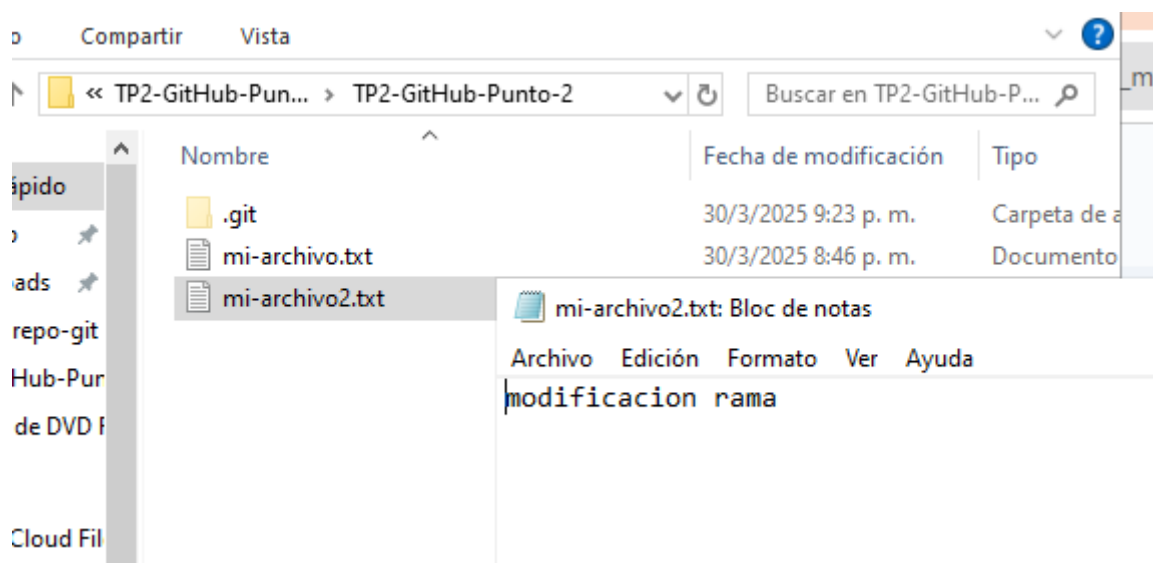
Brian@DESKTOP-TMHIVMC MINGW64 ~/Desktop/UTN/TP2-GitHub-Punto-2/TP2-GitHub-Punto-2 (main)
$ git branch
  NuevaRama
* main
```

Archivo modificado desde nueva rama

```
Brian@DESKTOP-TMHIVMC MINGW64 ~/Desktop/UTN/TP2-GitHub-Punto-2/TP2-GitHub-Punto-2 (NuevaRama)
$ echo "modificacion rama" > mi-archivo2.txt
```

```
Brian@DESKTOP-TMHIVMC MINGW64 ~/Desktop/UTN/TP2-GitHub-Punto-2/TP2-GitHub-Punto-2 (NuevaRama)
$ git add .

Brian@DESKTOP-TMHIVMC MINGW64 ~/Desktop/UTN/TP2-GitHub-Punto-2/TP2-GitHub-Punto-2 (NuevaRama)
$ git push origin NuevaRama
Total 0 (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/colque501/TP2-GitHub-Punto-2/pull/new/NuevaRama
remote:
To https://github.com/colque501/TP2-GitHub-Punto-2.git
 * [new branch]      NuevaRama -> NuevaRama
```



3) 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".

Link: <https://github.com/colque501/conflict-exercise.git>

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 *  colque501 / Repository name * conflict-exercise

✔ conflict-exercise is available.

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

Description (optional)

TP2 - Ejercicio n3

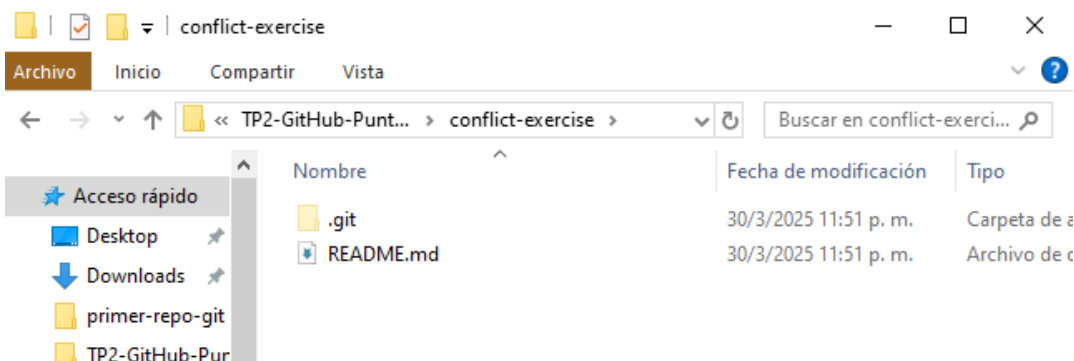
- ☒  Public
Anyone on the internet can see this repository. You choose who can commit.
- ☐  Private
You choose who can see and commit to this repository.

Initialize this repository with:

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

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:

```
MINGW64:/c/Users/Brian/Desktop/UTN/TP2-GitHub-Punto-2
Brian@DESKTOP-TMHIWMC MINGW64 ~/Desktop/UTN/TP2-GitHub-Punto-2
$ git clone https://github.com/colque501/conflict-exercise.git
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 • Crea una nueva rama llamada feature-branch: `git checkout -b 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"`

```

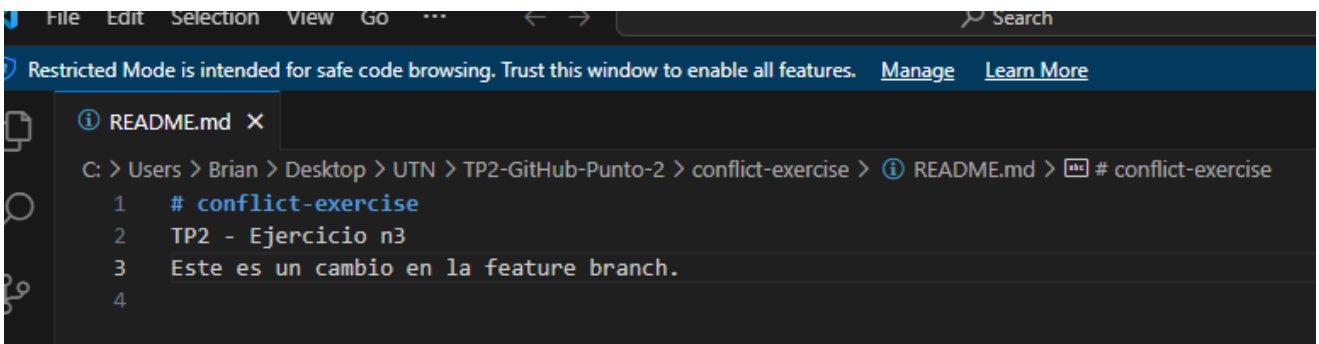
Brian@DESKTOP-TMHIVMC MINGW64 ~/Desktop/UTN/TP2-GitHub-Punto-2/conflict-exercise
(main)
$ git checkout -b feature-branch
Switched to a new branch 'feature-branch'

Brian@DESKTOP-TMHIVMC MINGW64 ~/Desktop/UTN/TP2-GitHub-Punto-2/conflict-exercise
(feature-branch)
$ git add README.md

Brian@DESKTOP-TMHIVMC MINGW64 ~/Desktop/UTN/TP2-GitHub-Punto-2/conflict-exercise
(feature-branch)
$ git commit -m "Added a line in feature-branch"
[feature-branch 7d51a60] Added a line in feature-branch
1 file changed, 1 insertion(+)

Brian@DESKTOP-TMHIVMC MINGW64 ~/Desktop/UTN/TP2-GitHub-Punto-2/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 • 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"

```

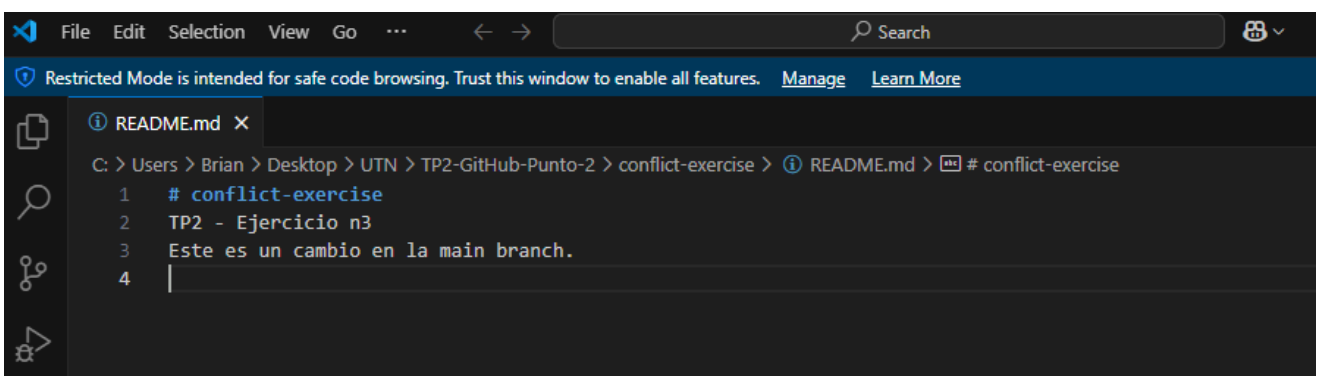
Brian@DESKTOP-TMHIVMC MINGW64 ~/Desktop/UTN/TP2-GitHub-Punto-2/conflict-exercise
(feature-branch)
$ git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.

Brian@DESKTOP-TMHIVMC MINGW64 ~/Desktop/UTN/TP2-GitHub-Punto-2/conflict-exercise
(main)
$ git add README.md

Brian@DESKTOP-TMHIVMC MINGW64 ~/Desktop/UTN/TP2-GitHub-Punto-2/conflict-exercise
(main)
$ git commit -m "Added a line in main branch"
[main 2653dd6] Added a line in main branch
1 file changed, 1 insertion(+)

Brian@DESKTOP-TMHIVMC MINGW64 ~/Desktop/UTN/TP2-GitHub-Punto-2/conflict-exercise
(main)
$

```

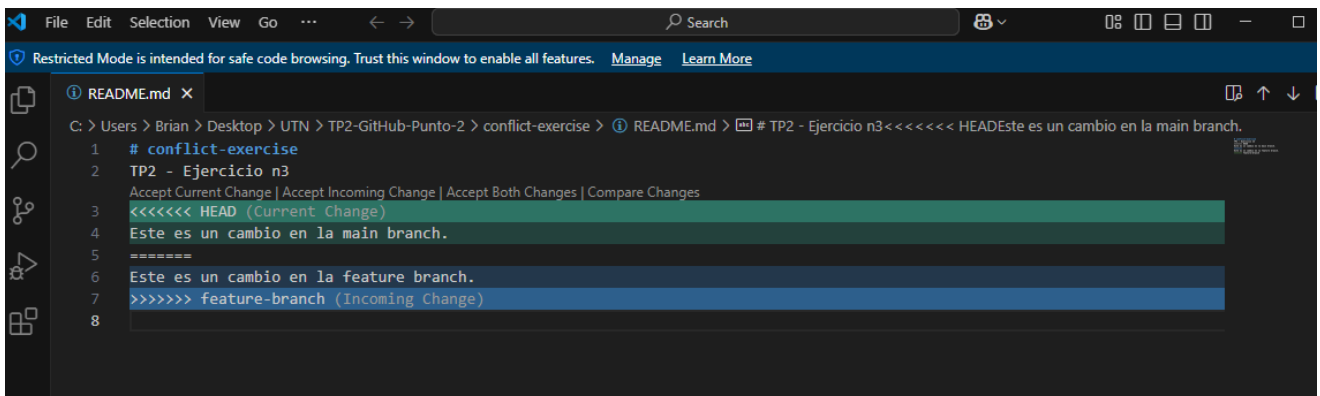


Paso 5: Hacer un merge y generar un conflicto • Intenta hacer un merge de la feature-branch en la rama main: `git merge feature-branch` • Se generará un conflicto porque ambos cambios afectan la misma línea del archivo README.md.

```
Brian@DESKTOP-TMHIVMC MINGW64 ~/Desktop/UTN/TP2-GitHub-Punto-2/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.

Brian@DESKTOP-TMHIVMC MINGW64 ~/Desktop/UTN/TP2-GitHub-Punto-2/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"`



```
1 # conflict-exercise
2 TP2 - Ejercicio n3
3 Este es un cambio en la main branch.
4 Este es un cambio en la feature branch.
5
```

```
Brian@DESKTOP-TMHIVMC MINGW64 ~/Desktop/UTN/TP2-GitHub-Punto-2/conflict-exercise
(main|MERGING)
$ git add README.md

Brian@DESKTOP-TMHIVMC MINGW64 ~/Desktop/UTN/TP2-GitHub-Punto-2/conflict-exercise
(main|MERGING)
$ git commit -m "Resolved merge conflict"
[main 14aed15] Resolved merge conflict

Brian@DESKTOP-TMHIVMC MINGW64 ~/Desktop/UTN/TP2-GitHub-Punto-2/conflict-exercise
(main)
$
```

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`


```

Brian@DESKTOP-TMHIVMC MINGW64 ~/Desktop/UTN/TP2-GitHub-Punto-2/conflict-exercise
(main)
$ 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), 777 bytes | 70.00 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/colque501/conflict-exercise.git
    d1fe358..14aed15  main -> main

Brian@DESKTOP-TMHIVMC MINGW64 ~/Desktop/UTN/TP2-GitHub-Punto-2/conflict-exercise
(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/colque501/conflict-exercise/pull/new/feature-branch
remote:
To https://github.com/colque501/conflict-exercise.git
 * [new branch]      feature-branch -> feature-branch

Brian@DESKTOP-TMHIVMC MINGW64 ~/Desktop/UTN/TP2-GitHub-Punto-2/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.

colque501 committed 8 minutes ago

Resolved merge conflict

main 2 parents 2653dd6 + 7d51a60 commit 14aed15

Filter files... README.md

1 file changed +1-0 lines changed

Search within code

1 1 # conflict-exercise

2 2 TP2 - Ejercicio n3

3 3 Este es un cambio en la main branch.

4 + Este es un cambio en la feature branch.

Comments 0 Lock conversation