

Práctico 2: Git y GitHub

Andrelo Luciano Aldo

Fork: [andrelo/UTN-TUPaD-P1: Repositorio Programacion1](https://github.com/andrelo/UTN-TUPaD-P1: Repositorio Programacion1)

Actividades

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

- **¿Qué es GitHub?** Es una comunidad donde podemos compartir nuestros repositorios de forma pública o privada, permite trabajar con repositorios remotos y trabajar en proyectos de terceros de forma colaborativa
- **¿Cómo crear un repositorio en GitHub?** Una vez iniciada tu sesión, haz click en el botón create repository, asigna un nombre al repositorio y elige si quieres un README, si sera publico o privado
- **¿Cómo crear una rama en Git?** una rama (o branch) se crea con el comando git branch y le asignamos el nombre
- **¿Cómo cambiar a una rama en Git?** para cambiar a esta rama debemos usar git checkout mas el nombre de la rama, esto nos permite desviarnos de la “rama principal” y experimentar sin comprometer.
- **¿Cómo fusionar ramas en Git?** Con el comando git merge (nombre_rama), puede presentar conflictos que serán resueltos por git o por el programador
- **¿Cómo crear un commit en Git?** con el comando git commit, registra los cambios, dejando una especie de ‘checkpoint’, también permite dejar un mensaje
- **¿Cómo enviar un commit a GitHub?** Utilizando git push origin (nombre_rama), permite subir los commits al repositorio remoto.
- **¿Qué es un repositorio remoto?** Es una copia de tu repositorio (carpeta) local almacenada en la nube, en este caso GitHub, que tiene fácil acceso
- **¿Cómo agregar un repositorio remoto a Git?** Conectando mi repositorio local al remoto (github) usando el comando git remote add origin (url_del_repositorio)
- **¿Cómo empujar cambios a un repositorio remoto?** Para subir nuestros cambios al repositorio remoto se utiliza git push origin (nombre_rama)
- **¿Cómo tirar de cambios de un repositorio remoto?** Para descargar los cambios del remoto a tu repositorio local se utiliza el codigo git pull origin (nombre_rama)
- **¿Qué es un fork de repositorio?** Es una copia de un repositorio creada en una cuenta diferente permitiendo desarrollar cambios sin afectar el original
- **¿Cómo crear un fork de un repositorio?** Haciendo click en el botón Fork del repositorio original

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

En github, haz click en New pull request desde tu fork para proponer cambios en el repositorio original.

- **¿Cómo aceptar una solicitud de extracción?** En GitHub, desde la pestaña Pull Requests, podras revisar cambios propuestos y aplicarlos haciendo click en Merge pull request
- **¿Qué es un etiqueta en Git?** Es un marcador que se utiliza para señalar un commit específico, generalmente se usa para etiquetar la versión del proyecto
- **¿Cómo crear una etiqueta en Git?** Con el comando `git tag -a (nombre-etiqueta) -m "(descripcion de etiqueta)"`, para verlas se escribe `git tag` y las mostrará en orden alfabetico
- **¿Cómo enviar una etiqueta a GitHub?** con `git push origin (nombre_etiqueta)`
- **¿Qué es un historial de Git?** Es el registro de todos los commits hechos en el repositorio, permite inspeccionar detalles, como el autor, fecha o mensaje
- **¿Cómo ver el historial de Git?** con el comando `git log`, busca mensajes de commits
- **¿Cómo buscar en el historial de Git?** con el comando `git log -grep= "texto a buscar"`
 - **¿Cómo borrar el historial de Git?** con el comando `rm -rf .git`, luego `git init`
- **¿Qué es un repositorio privado en GitHub?** Es un repositorio al que solo tienen acceso personas con invitación, sirve para tener algunos proyectos privados
- **¿Cómo crear un repositorio privado en GitHub?** En GitHub, cuando creas un repositorio, selecciona la opción Private y sigue los pasos
- **¿Cómo invitar a alguien a un repositorio privado en GitHub?** Desde GitHub, en Settings/Collaborators, agregar el nombre del usuario y enviar invitación
- **¿Qué es un repositorio público en GitHub?** Es un repositorio accesible para cualquier usuario de internet, facilita la colaboración en proyectos
- **¿Cómo crear un repositorio público en GitHub?** Desde Github, haz click en Create Repository y tilda la opcion Public y sigue los pasos
- **¿Cómo compartir un repositorio público en GitHub?** Copia la URL del repositorio y compártela directamente, esto da acceso a otros usuarios para explorar o colaborar en tus proyectos.

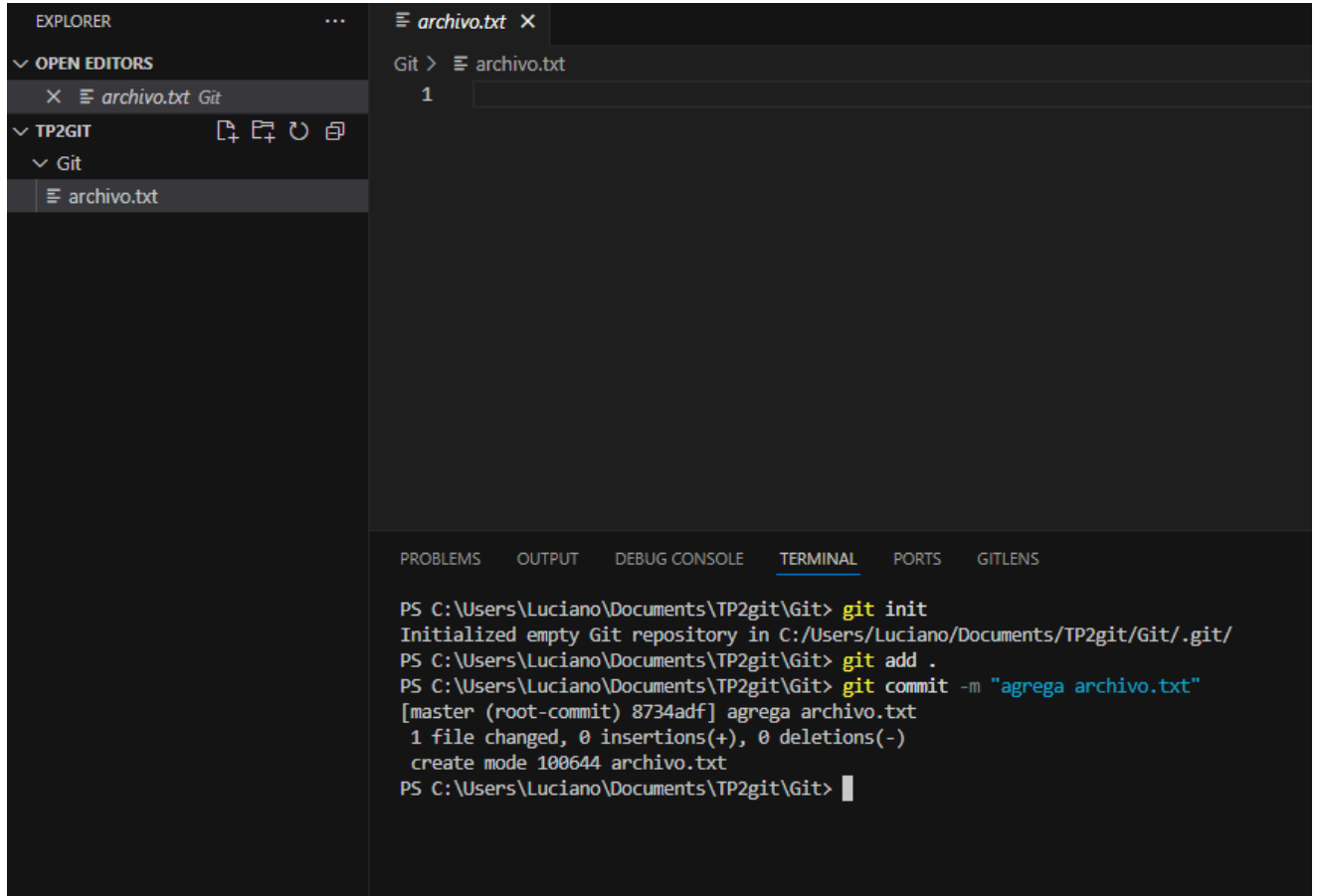
2) Realizar la siguiente actividad:

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

- Creando Branchs

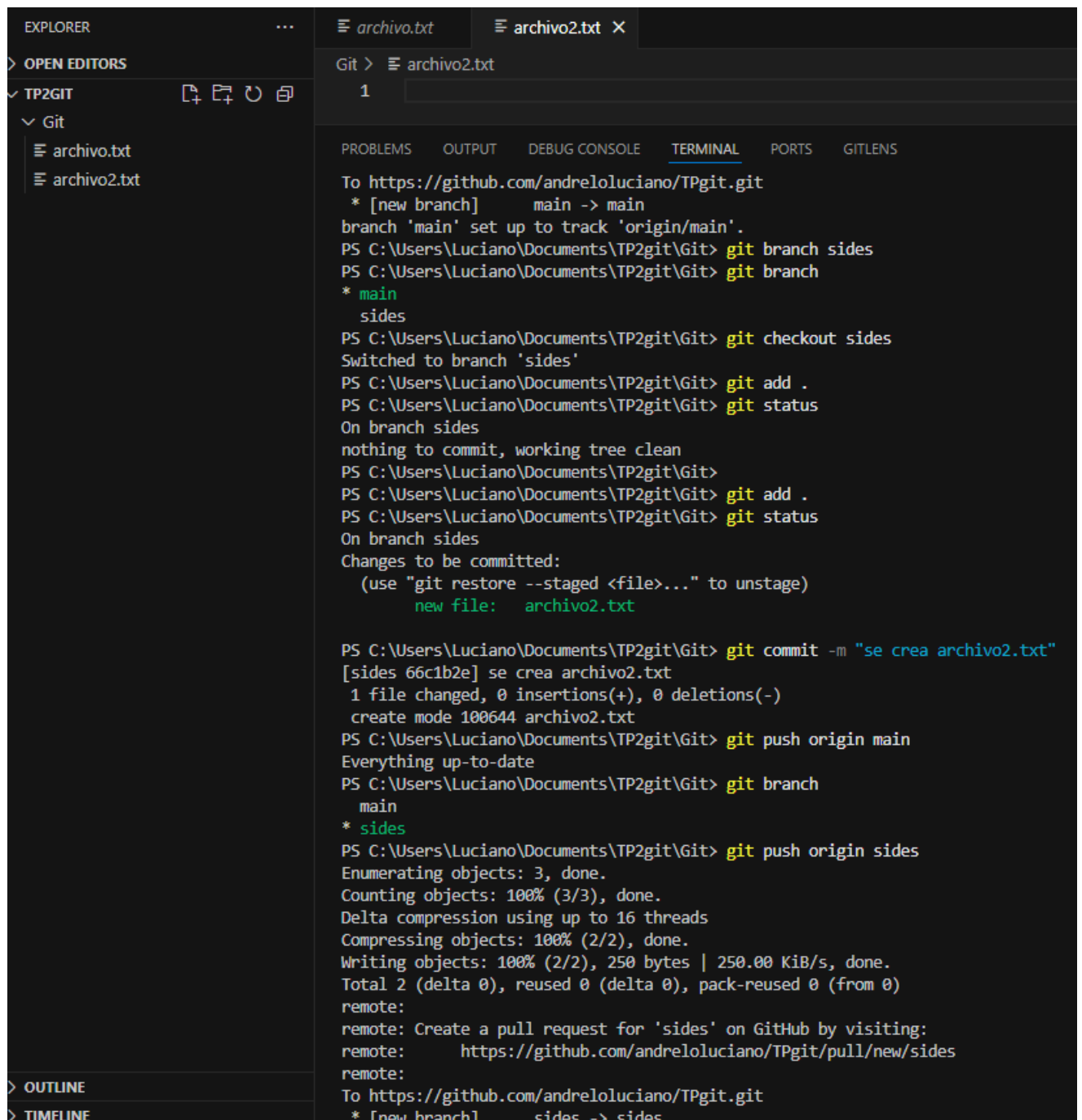
- Crear una Branch
- Realizar cambios o agregar un archivo
- Subir la Branch

[andreloluciano/TPgit: TP2 Consigna 2](#)



The screenshot shows the Visual Studio Code interface with a dark theme. On the left, the Explorer sidebar shows a project named 'TP2GIT' with a subfolder 'Git' containing a file 'archivo.txt'. The Open Editors sidebar shows 'archivo.txt' is open in the 'Git' workspace. The main editor area shows the 'archivo.txt' file with a single line of text: '1'. At the bottom, the Terminal panel is active, displaying the following commands and output:

```
PS C:\Users\Luciano\Documents\TP2git\Git> git init
Initialized empty Git repository in C:/Users/Luciano/Documents/TP2git/Git/.git/
PS C:\Users\Luciano\Documents\TP2git\Git> git add .
PS C:\Users\Luciano\Documents\TP2git\Git> git commit -m "agrega archivo.txt"
[master (root-commit) 8734adf] agrega archivo.txt
1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 archivo.txt
PS C:\Users\Luciano\Documents\TP2git\Git> |
```



```
EXPLORER
...
OPEN EDITORS
TP2GIT
  Git
    archivo.txt
    archivo2.txt

Git > archivo2.txt
1

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS GITLENS

To https://github.com/andreloluciano/TPgit.git
* [new branch]    main -> main
branch 'main' set up to track 'origin/main'.
PS C:\Users\Luciano\Documents\TP2git\Git> git branch sides
PS C:\Users\Luciano\Documents\TP2git\Git> git branch
* main
  sides
PS C:\Users\Luciano\Documents\TP2git\Git> git checkout sides
Switched to branch 'sides'
PS C:\Users\Luciano\Documents\TP2git\Git> git add .
PS C:\Users\Luciano\Documents\TP2git\Git> git status
On branch sides
nothing to commit, working tree clean
PS C:\Users\Luciano\Documents\TP2git\Git>
PS C:\Users\Luciano\Documents\TP2git\Git> git add .
PS C:\Users\Luciano\Documents\TP2git\Git> git status
On branch sides
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    new file:   archivo2.txt

PS C:\Users\Luciano\Documents\TP2git\Git> git commit -m "se crea archivo2.txt"
[sides 66c1b2e] se crea archivo2.txt
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 archivo2.txt
PS C:\Users\Luciano\Documents\TP2git\Git> git push origin main
Everything up-to-date
PS C:\Users\Luciano\Documents\TP2git\Git> git branch
  main
* sides
PS C:\Users\Luciano\Documents\TP2git\Git> git push origin sides
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Delta compression using up to 16 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (2/2), 250 bytes | 250.00 KiB/s, done.
Total 2 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
remote:
remote: Create a pull request for 'sides' on GitHub by visiting:
remote:   https://github.com/andreloluciano/TPgit/pull/new/sides
remote:
To https://github.com/andreloluciano/TPgit.git
* [new branch]    sides -> sides
```

3) Realizar la siguiente actividad:

[andreloluciano/conflict-exercise](https://github.com/andreloluciano/conflict-exercise)

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".

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
```

- 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"
```

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"
```

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.

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
```

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 a VS Code editor with a file explorer on the left and a terminal at the bottom. The file explorer shows a project structure with a folder 'TAREA' containing a subfolder 'utn\conflict-exercise' and a file 'README.md'. The editor window shows the 'README.md' file with a conflict resolution. The content of the file is as follows:

```

utn > conflict-exercise > README.md > # este es un cambio en la main branch
You, 1 second ago | 1 author (You)
1 # conflict-exercise
2
3 # este es un cambio en la main branch      You, 1 second ago • Uncommitted changes

```

The terminal window shows the following commands and output:

```

PS C:\Users\Luciano\Documents\Tarea\utn\conflict-exercise> git checkout feature branch
error: pathspec 'feature' did not match any file(s) known to git
error: pathspec 'branch' did not match any file(s) known to git
PS C:\Users\Luciano\Documents\Tarea\utn\conflict-exercise> git checkout feature-branch
error: pathspec 'feature-branch' did not match any file(s) known to git
PS C:\Users\Luciano\Documents\Tarea\utn\conflict-exercise> git branch feature-branch
PS C:\Users\Luciano\Documents\Tarea\utn\conflict-exercise> git branch
* main
PS C:\Users\Luciano\Documents\Tarea\utn\conflict-exercise> git checkout feature-branch
Switched to branch 'feature-branch'
PS C:\Users\Luciano\Documents\Tarea\utn\conflict-exercise> git add README.md
PS C:\Users\Luciano\Documents\Tarea\utn\conflict-exercise> git commit -m "Added a line in featubre-branch"
[feature-branch a799e50] Added a line in featubre-branch
1 file changed, 2 insertions(+), 1 deletion(-)
PS C:\Users\Luciano\Documents\Tarea\utn\conflict-exercise> git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.
PS C:\Users\Luciano\Documents\Tarea\utn\conflict-exercise> git add README.md
PS C:\Users\Luciano\Documents\Tarea\utn\conflict-exercise> git commit -m "Added a line in main branch"
[main e50bd8e] Added a line in main branch
1 file changed, 3 insertions(+), 1 deletion(-)
PS C:\Users\Luciano\Documents\Tarea\utn\conflict-exercise>

```

EXPLORER

OPEN EDITORS

TAREA

utn\conflict-exercise

README.md

utn > conflict-exercise > README.md

Este es un cambio en la feature-branch

You, 1 second ago | 1 author (You)

1 # conflict-exercise

2 Accept Current Change | Accept Incoming Change | Accept Both Changes | Compare Changes

3 <<<<<< HEAD (Current Change)

4

5 # este es un cambio en la main branch

6 =====

7 # Este es un cambio en la feature-branch

8 >>>>>> feature-branch (Incoming Change)

Resolve in Merge Editor

PROBLEMS

OUTPUT

DEBUG CONSOLE

TERMINAL

PORTS

GITLENS

error: pathspec 'branch' did not match any file(s) known to git
PS C:\Users\Luciano\Documents\Tarea\utn\conflict-exercise> git checkout feature-branch
error: pathspec 'feature-branch' did not match any file(s) known to git
PS C:\Users\Luciano\Documents\Tarea\utn\conflict-exercise> git branch feature-branch
PS C:\Users\Luciano\Documents\Tarea\utn\conflict-exercise> git branch
feature-branch
* main
PS C:\Users\Luciano\Documents\Tarea\utn\conflict-exercise> git checkout feature-branch
Switched to branch 'feature-branch'
PS C:\Users\Luciano\Documents\Tarea\utn\conflict-exercise> git add README.md
PS C:\Users\Luciano\Documents\Tarea\utn\conflict-exercise> git commit -m "Added a line in feautubre-branch"
[feature-branch a799e50] Added a line in feautubre-branch
1 file changed, 2 insertions(+), 1 deletion(-)
PS C:\Users\Luciano\Documents\Tarea\utn\conflict-exercise> git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.
PS C:\Users\Luciano\Documents\Tarea\utn\conflict-exercise> git add README.md
PS C:\Users\Luciano\Documents\Tarea\utn\conflict-exercise> git commit -m "Added a line in main branch"
[main e50bd8e] Added a line in main branch
1 file changed, 3 insertions(+), 1 deletion(-)
PS C:\Users\Luciano\Documents\Tarea\utn\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\Luciano\Documents\Tarea\utn\conflict-exercise> |

master* Launchpad 0 Not Committed Yet Ln 8, Col 1 Spaces: 4 UTF-8 CRLF Markdown

EXPLORER

OPEN EDITORS 1 unsaved

README.md utn\conflict-exer...

TAREA

utn\conflict-exercise

README.md

utn > conflict-exercise > README.md

conflict-exercise

You, 2 minutes ago | 1 author (You)

1 # conflict-exercise

2 # este es un cambio en la main branch

3

4

5

PROBLEMS

OUTPUT

DEBUG CONSOLE


TERMINAL

PORTS

GITLENS

1 file changed, 3 insertions(+), 1 deletion(-)
PS C:\Users\Luciano\Documents\Tarea\utn\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\Luciano\Documents\Tarea\utn\conflict-exercise> git add README.md
PS C:\Users\Luciano\Documents\Tarea\utn\conflict-exercise> git commit -m "Resolved merge conflict"
[main cdb971] Resolved merge conflict
PS C:\Users\Luciano\Documents\Tarea\utn\conflict-exercise> git push origin main
Enumerating objects: 11, done.
Counting objects: 100% (11/11), done.
Delta compression using up to 16 threads
Compressing objects: 100% (6/6), done.
Writing objects: 100% (9/9), 870 bytes | 870.00 KiB/s, done.
Total 9 (delta 1), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (1/1), done.
To https://github.com/andreoluciano/conflict-exercise.git
6a434e6..cdb971 main -> main
PS C:\Users\Luciano\Documents\Tarea\utn\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/andreoluciano/conflict-exercise/pull/new/feature-branch
remote:
To https://github.com/andreoluciano/conflict-exercise.git
* [new branch] feature-branch -> feature-branch
PS C:\Users\Luciano\Documents\Tarea\utn\conflict-exercise> |


master* Launchpad 0 You, 4 minutes ago Luciano (4 minutes ago) Ln 1, Col 20 Spaces: 4 UTF-8

 **conflict-exercise** Public

Pin Unwatch

main 2 Branches 0 Tags

t Add file <> Code

 **andreloluciano** Resolved merge conflict cdb971 · 1 minute ago 🕒 4 Commits

README.md Resolved merge conflict 1 minute ago

README 📄 ✎ ☰

conflict-exercise

<<<<<<< HEAD

este es un cambio en la main branch

=====

Este es un cambio en la feature-branch

| | | | | | | feature-branch

[andreloluciano/TPgit: TP2 Consigna 2](#)

[andreloluciano/conflict-exercise](#)

Fork: [andreloluciano/UTN-TUPaD-P1: Repositorio Programacion1](#)