

## Práctico 2: Git y GitHub

### Objetivo:

El estudiante desarrollará competencias para trabajar con Git y GitHub, aplicando conceptos fundamentales de control de versiones, colaboración en proyectos y resolución de conflictos, en un entorno simulado y guiado.

### Resultados de aprendizaje:

1. Comprender los conceptos básicos de Git y GitHub: Identificar y explicar los principales términos y procesos asociados con Git y GitHub, como repositorios, ramas, commits, forks, etiquetas y repositorios remotos.
2. Manejar comandos esenciales de Git: Ejecutar comandos básicos para crear, modificar, fusionar y gestionar ramas, commits y repositorios, tanto en local como en remoto.
3. Aplicar técnicas de colaboración en GitHub: Configurar y utilizar repositorios remotos, realizar forks, y gestionar pull requests para facilitar el trabajo colaborativo.
4. Resolver conflictos en un entorno de control de versiones: Identificar, analizar y solucionar conflictos de merge generados en un flujo de trabajo con múltiples ramas.

### Actividades

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

- ¿Qué es GitHub?

GitHub es una página web para guardar los repositorios creados con git, para poder mostrarlos, compartirlos o trabajar en equipo.

- ¿Cómo crear un repositorio en GitHub?

Para crear un repositorio, luego de iniciar sesión se debe apretar en "New repository", una vez dentro se debe colocar el nombre, si es publico o privado y luego apretar en "Create repository".

- ¿Cómo crear una rama en Git?

Para crear una rama en Git se debe ingresar desde la terminal el siguiente comando:  
`git branch nombre-de-la-rama`

- ¿Cómo cambiar a una rama en Git?

Para cambiar a otra rama en Git, se debe ingresar el siguiente comando:

`Git checkout nombre-de-la-rama`

- ¿Cómo fusionar ramas en Git?

Para fusionar una rama, primero se debe cambiar a la rama principal y luego ingresar el siguiente comando con el nombre de la rama a fusionar:

`git merge nombre-de-la-rama`

- ¿Cómo crear un commit en Git?

Primero se debe agregar los archivos con el comando “git add .”

Luego realizar el commit con un mensaje aclarando de forma concisa que acción se realizó: “git commit -am cambio-realizado”

- ¿Cómo enviar un commit a GitHub?

Para enviar el commit primero se debe copiar la url de un repositorio remoto creado o ya existente de Github y conectarlo con el repositorio local:

git remote add origin <https://github.com/usuario/nombre-del-repo.git>

Luego enviar el commit:

git push -u origin nombre-rama

- ¿Qué es un repositorio remoto?

Un repositorio remoto es una copia del proyecto guardado en internet, en paginas como Github.

- ¿Cómo agregar un repositorio remoto a Git?

Para agregar un repositorio remoto a Git, se debe conseguir la url del remoto y realizar el siguiente comando en la terminal:

git remote add origin <https://github.com/usuario/repositorio.git>

- ¿Cómo empujar cambios a un repositorio remoto?

Se debe ingresar el comando:

git push origin nombre-de-la-rama

- ¿Cómo tirar de cambios de un repositorio remoto?

Con el comando:

git pull origin nombre-de-la-rama

- ¿Qué es un fork de repositorio?

Un fork es una copia de un proyecto de un tercero, que lo agregas a tu perfil para poder trabajar en el y realizar cambios sin afectar al proyecto original.

- ¿Cómo crear un fork de un repositorio?

Se debe ir al repositorio de otro usuario y apretar en donde dice “Fork”.

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

Después de realizar cambios en el fork, en la página del repositorio original hacés clic en "Pull Request" para proponer los nuevos cambios.

- ¿Cómo aceptar una solicitud de extracción?

Una vez revisado los cambios del pull request, y función y se esta de acuerdo con los cambios realizados se debe realizar click en "Merge".

- ¿Qué es un etiqueta en Git?

Es como una marca que se pone en un momento importante del proyecto, por ejemplo una versión final.

- ¿Cómo crear una etiqueta en Git?

Para crear una etiqueta se debe ingresar el comando:

```
git tag nombre-de-la-etiqueta
```

- ¿Cómo enviar una etiqueta a GitHub?

Se debe ingresar el comando:

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

- ¿Qué es un historial de Git?

Es el registro de todos los cambios que se hicieron en el proyecto.

- ¿Cómo ver el historial de Git?

Se puede ver el historial ingresando el comando "git log".

- ¿Cómo buscar en el historial de Git?

Se ingresa el comando con la palabra clave: `git log --grep="palabra clave"`.

- ¿Cómo borrar el historial de Git?

Para borrar el historial se debe ingresar el comando: `rm -rf .git`

- ¿Qué es un repositorio privado en GitHub?

Es un repositorio que solo el usuario autor lo puede ver y a los que decide invitar.

- ¿Cómo crear un repositorio privado en GitHub?

Al momento de crearlo se debe apretar en la opción "Privado".

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

Se debe entrar al repositorio > "Settings" > "Collaborators" y agregar el nombre del usuario.

- ¿Qué es un repositorio público en GitHub?

Es un repositorio que cualquier persona puede ver.

- ¿Cómo crear un repositorio público en GitHub?

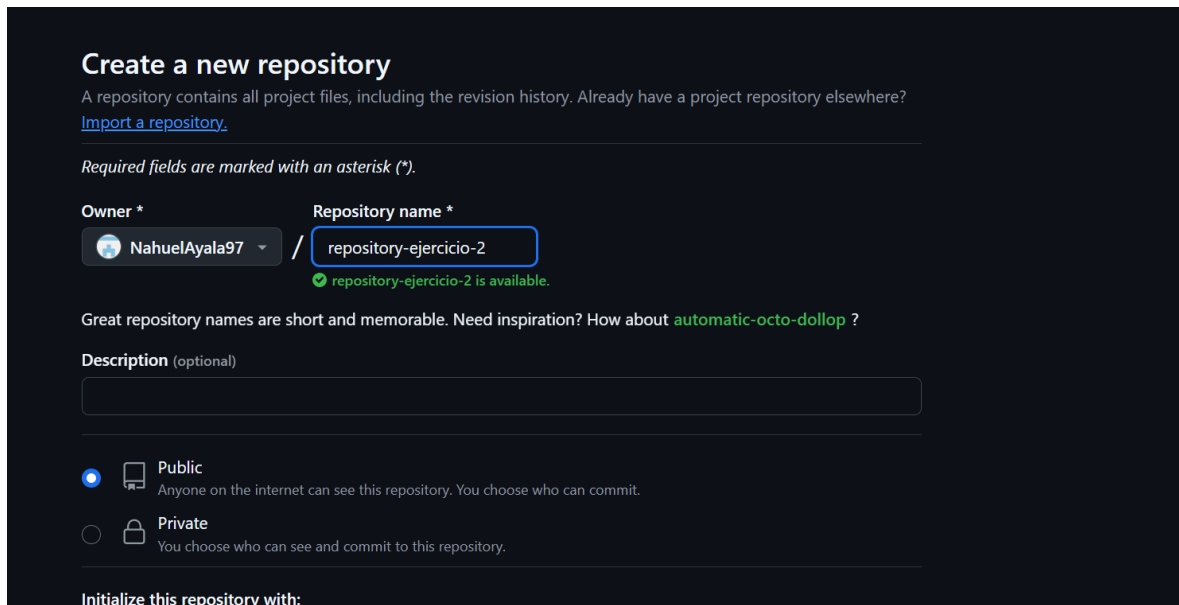
Al crearlo se debe elegir la opción “Publico”.

- ¿Cómo compartir un repositorio público en GitHub?

Se puede compartir copiando el link del repositorio y enviarlo.

## 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.



**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 \*** NahuelAyla97 / **Repository name \*** repository-ejercicio-2  
repository-ejercicio-2 is available.

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

**Description** (optional)

☐ **Public**  
Anyone on the internet can see this repository. You choose who can commit.

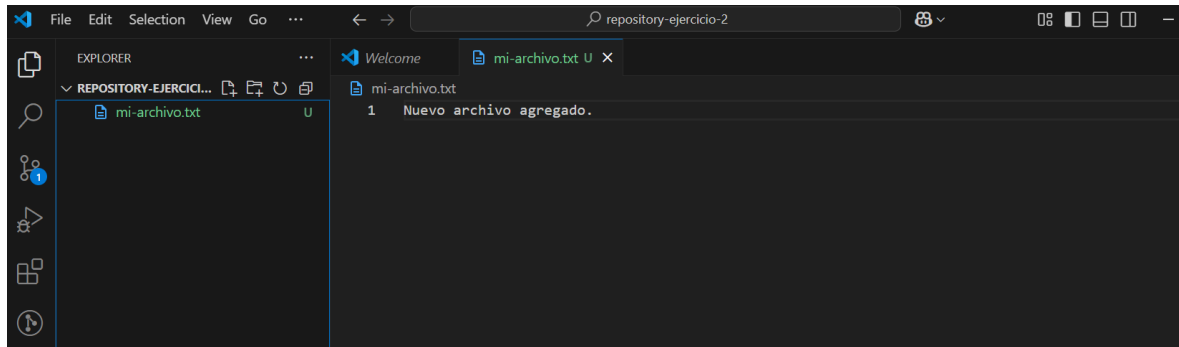
☐ **Private**  
You choose who can see and commit to this repository.

**Initialize this repository with:**

```
Nahuel.Ayala@nayala-ntb MINGW64 ~/Desktop/programacion1
$ git clone https://github.com/NahuelAyla97/repository-ejercicio-2.git
Cloning into 'repository-ejercicio-2'...
warning: You appear to have cloned an empty repository.

Nahuel.Ayala@nayala-ntb MINGW64 ~/Desktop/programacion1
$
```

- 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).



```
No commits yet

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)

Nahuel.Ayala@nayala-ntb MINGW64 ~/Desktop/programacion1/repository-ejercicio-2 (main)
$ git add .

Nahuel.Ayala@nayala-ntb MINGW64 ~/Desktop/programacion1/repository-ejercicio-2 (main)
$ git commit -am "Agregando mi-archivo.txt"
[main (root-commit) 9cb81ab] Agregando mi-archivo.txt
Committer: Nahuel Ayala <Nahuel.Ayala@bplay.com.ar>
Your name and email address were configured automatically based
on your username and hostname. Please check that they are accurate.
You can suppress this message by setting them explicitly. Run the
following command and follow the instructions in your editor to edit
your configuration file:

    git config --global --edit

After doing this, you may fix the identity used for this commit with:

    git commit --amend --reset-author

1 file changed, 1 insertion(+)
create mode 100644 mi-archivo.txt

Nahuel.Ayala@nayala-ntb MINGW64 ~/Desktop/programacion1/repository-ejercicio-2 (main)
```

```
Nahuel.Ayala@nayala-ntb MINGW64 ~/Desktop/programacion1/repository-ejercicio-2 (main)
$ git push origin main
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Writing objects: 100% (3/3), 253 bytes | 253.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/NahuelAyala97/repository-ejercicio-2.git
 * [new branch]      main -> main

Nahuel.Ayala@nayala-ntb MINGW64 ~/Desktop/programacion1/repository-ejercicio-2 (main)
$
```

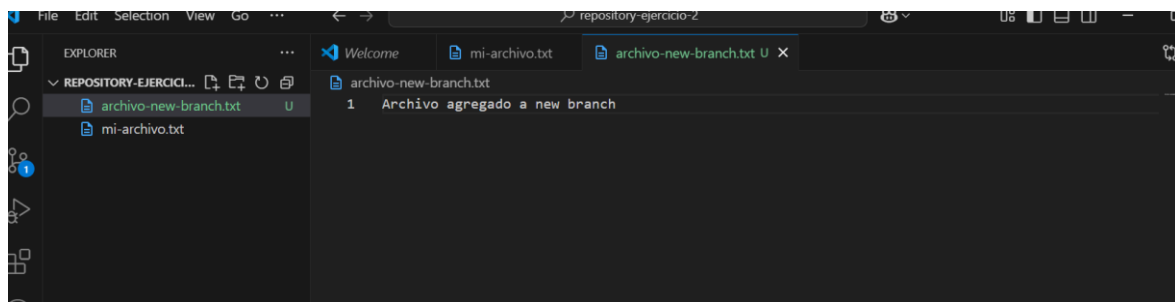
- Creando Branchs
  - o Crear una Branch
  - o Realizar cambios o agregar un archivo
  - o Subir la Branch

```
Nahuel.Ayala@nayala-ntb MINGW64 ~/Desktop/programacion1/repository-ejercicio-2 (main)
$ git branch new-branch

Nahuel.Ayala@nayala-ntb MINGW64 ~/Desktop/programacion1/repository-ejercicio-2 (main)
$ git checkout new-branch
Switched to branch 'new-branch'

Nahuel.Ayala@nayala-ntb MINGW64 ~/Desktop/programacion1/repository-ejercicio-2 (new-branch)
$ code .

Nahuel.Ayala@nayala-ntb MINGW64 ~/Desktop/programacion1/repository-ejercicio-2 (new-branch)
$
```



```
Nahuel.Ayala@nayala-ntb MINGW64 ~/Desktop/programacion1/repository-ejercicio-2 (new-branch)
$ git status
On branch new-branch
Untracked files:
  (use "git add <file>..." to include in what will be committed)
        archivo-new-branch.txt

nothing added to commit but untracked files present (use "git add" to track)

Nahuel.Ayala@nayala-ntb MINGW64 ~/Desktop/programacion1/repository-ejercicio-2 (new-branch)
$ git add .

Nahuel.Ayala@nayala-ntb MINGW64 ~/Desktop/programacion1/repository-ejercicio-2 (new-branch)
$ git commit -am "Creando new branch y se agrega archivo."
[new-branch eb09382] Creando new branch y se agrega archivo.
Committer: Nahuel Ayala <Nahuel.Ayala@bplay.com.ar>
Your name and email address were configured automatically based
on your username and hostname. Please check that they are accurate.
You can suppress this message by setting them explicitly. Run the
following command and follow the instructions in your editor to edit
your configuration file:

    git config --global --edit

After doing this, you may fix the identity used for this commit with:

    git commit --amend --reset-author

1 file changed, 1 insertion(+)
create mode 100644 archivo-new-branch.txt
```

```
Nahuel.Ayala@nayala-ntb MINGW64 ~/Desktop/programacion1/repository-ejercicio-2 (new-branch)
$ git push origin -u new-branch
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 16 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 340 bytes | 340.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
remote:
remote: Create a pull request for 'new-branch' on GitHub by visiting:
remote:   https://github.com/NahuelAyala97/repository-ejercicio-2/pull/new/new-branch
remote:
To https://github.com/NahuelAyala97/repository-ejercicio-2.git
 * [new branch]      new-branch -> new-branch
branch 'new-branch' set up to track 'origin/new-branch'.

Nahuel.Ayala@nayala-ntb MINGW64 ~/Desktop/programacion1/repository-ejercicio-2 (new-branch)
$
```

### 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".

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

Owner \* NahuelAyala97 / Repository name \* conflict-exercise  
✔ conflict-exercise is available.

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

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  
.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`

```
Nahuel.Ayala@nayala-ntb MINGW64 ~/Desktop/programacion1
$ git clone https://github.com/NahuelAyala97/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.

Nahuel.Ayala@nayala-ntb MINGW64 ~/Desktop/programacion1
$ cd conflict-exercise/

Nahuel.Ayala@nayala-ntb MINGW64 ~/Desktop/programacion1/conflict-exercise (main)
$
```

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.

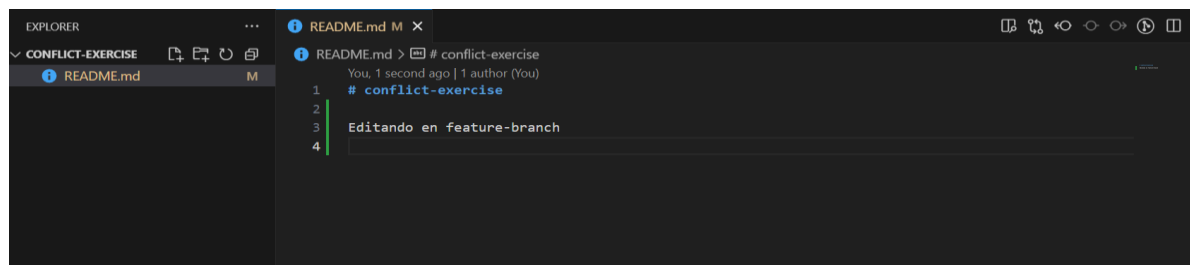
- Guarda los cambios y haz un commit:

```
git add README.md
```

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

```
Nahuel.Ayala@nayala-ntb MINGW64 ~/Desktop/programacion1/conflict-exercise (main)
$ git checkout -b feature-branch
Switched to a new branch 'feature-branch'

Nahuel.Ayala@nayala-ntb MINGW64 ~/Desktop/programacion1/conflict-exercise (feature-branch)
$ code .
```



```
Nahuel.Ayala@nayala-ntb MINGW64 ~/Desktop/programacion1/conflict-exercise (feature-branch)
$ git add .

Nahuel.Ayala@nayala-ntb MINGW64 ~/Desktop/programacion1/conflict-exercise (feature-branch)
$ ^[[200~git commit -m "Added a line in feature-branch"
bash: $'\E[200~git': command not found

Nahuel.Ayala@nayala-ntb MINGW64 ~/Desktop/programacion1/conflict-exercise (feature-branch)
$ git commit -am "Added a line in feature-branch"
[feature-branch cd6eaa1] Added a line in feature-branch
Committer: Nahuel Ayala <Nahuel.Ayala@bplay.com.ar>
Your name and email address were configured automatically based
on your username and hostname. Please check that they are accurate.
You can suppress this message by setting them explicitly. Run the
following command and follow the instructions in your editor to edit
your configuration file:

    git config --global --edit

After doing this, you may fix the identity used for this commit with:

    git commit --amend --reset-author

1 file changed, 3 insertions(+), 1 deletion(-)

Nahuel.Ayala@nayala-ntb MINGW64 ~/Desktop/programacion1/conflict-exercise (feature-branch)
$
```

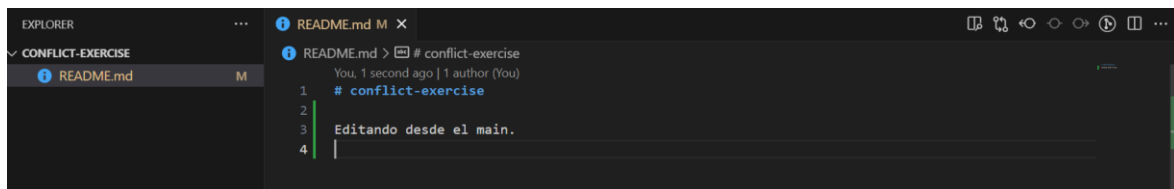


Paso 4: Volver a la rama principal y editar el mismo archivo

- 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"

```
Nahuel.Ayala@nayala-ntb MINGW64 ~/Desktop/programacion1/conflict-exercise (feature-branch)
$ git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.

Nahuel.Ayala@nayala-ntb MINGW64 ~/Desktop/programacion1/conflict-exercise (main)
$ code .
```



```
Nahuel.Ayala@nayala-ntb MINGW64 ~/Desktop/programacion1/conflict-exercise (main)
$ git add .

Nahuel.Ayala@nayala-ntb MINGW64 ~/Desktop/programacion1/conflict-exercise (main)
$ git commit -am "Added a line in main branch"
[main db3eeaa] Added a line in main branch
Committer: Nahuel Ayala <Nahuel.Ayala@bplay.com.ar>
Your name and email address were configured automatically based
on your username and hostname. Please check that they are accurate.
You can suppress this message by setting them explicitly. Run the
following command and follow the instructions in your editor to edit
your configuration file:

    git config --global --edit

After doing this, you may fix the identity used for this commit with:

    git commit --amend --reset-author

1 file changed, 3 insertions(+), 1 deletion(-)
```

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.

```
Nahuel.Ayala@nayala-ntb MINGW64 ~/Desktop/programacion1/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.

Nahuel.Ayala@nayala-ntb MINGW64 ~/Desktop/programacion1/conflict-exercise (main|MERGING)
```

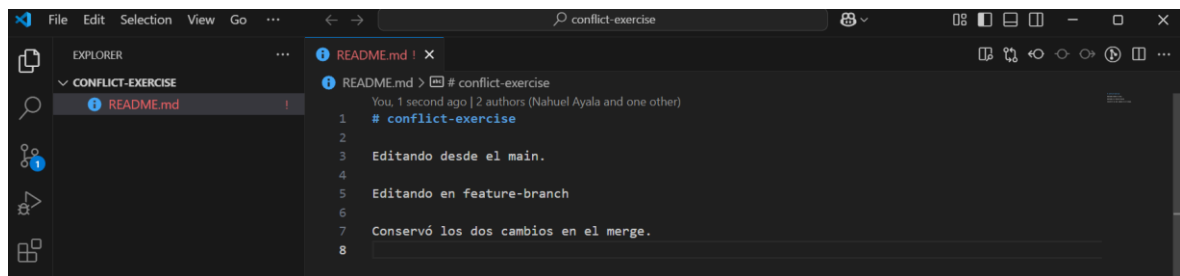
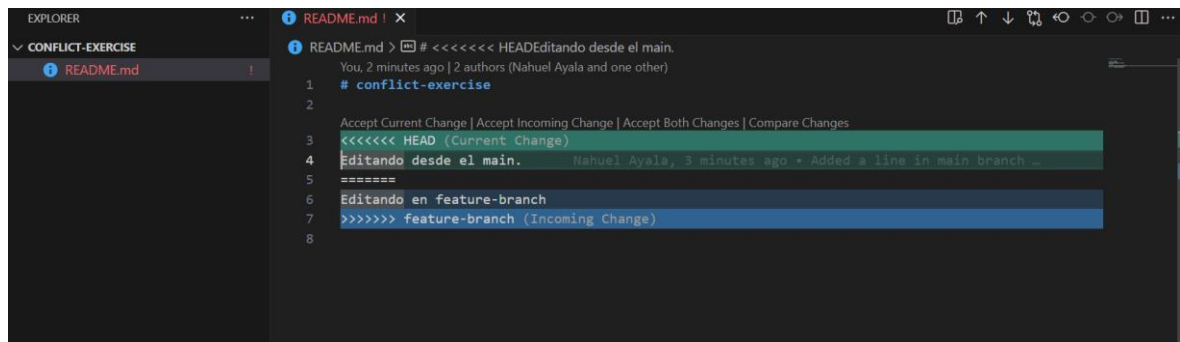
#### Paso 6: Resolver el conflicto

- 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"
```



```

Nahuel.Ayala@nayala-ntb MINGW64 ~/Desktop/programacion1/conflict-exercise (main|MERGING)
$ git status
On branch main
Your branch is ahead of 'origin/main' by 1 commit.
(use "git push" to publish your local commits)

You have unmerged paths.
  (fix conflicts and run "git commit")
  (use "git merge --abort" to abort the merge)

Unmerged paths:
  (use "git add <file>..." to mark resolution)
    both modified:   README.md

no changes added to commit (use "git add" and/or "git commit -a")

Nahuel.Ayala@nayala-ntb MINGW64 ~/Desktop/programacion1/conflict-exercise (main|MERGING)
$ git add .

Nahuel.Ayala@nayala-ntb MINGW64 ~/Desktop/programacion1/conflict-exercise (main|MERGING)
$ git commit -m "Resolved merge conflict"
[main a98c1c3] Resolved merge conflict
  Committer: Nahuel Ayala <Nahuel.Ayala@bplay.com.ar>
  Your name and email address were configured automatically based
  on your username and hostname. Please check that they are accurate.
  You can suppress this message by setting them explicitly. Run the
  following command and follow the instructions in your editor to edit
  your configuration file:

    git config --global --edit

After doing this, you may fix the identity used for this commit with:

    git commit --amend --reset-author

```

#### 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

```

Nahuel.Ayala@nayala-ntb MINGW64 ~/Desktop/programacion1/conflict-exercise (main)
$ git push
Enumerating objects: 11, done.
Counting objects: 100% (11/11), done.
Delta compression using up to 16 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (9/9), 844 bytes | 844.00 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/NahuelAyala97/conflict-exercise.git
   0e851cd..a98c1c3  main -> main

Nahuel.Ayala@nayala-ntb MINGW64 ~/Desktop/programacion1/conflict-exercise (main)
$ git push origin -u 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/NahuelAyala97/conflict-exercise/pull/new/feature-branch
remote:
To https://github.com/NahuelAyala97/conflict-exercise.git
 * [new branch]      feature-branch -> feature-branch
branch 'feature-branch' set up to track 'origin/feature-branch'.

Nahuel.Ayala@nayala-ntb MINGW64 ~/Desktop/programacion1/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.

