

# PROGRAMACION 1

## Rotolo Martin

### Práctico 2: Git y GitHub

1) Contestar las siguientes preguntas utilizando las guías y documentación proporcionada

(Desarrollar las respuestas) :

- **¿Qué es GitHub?**

GitHub es una plataforma en línea que permite almacenar proyectos que usan el sistema de control de versiones Git. Es el servicio más popular del mundo para compartir y colaborar en proyectos de software, tanto de código abierto como privados. Almacena los repositorios en la nube, lo que facilita el trabajo colaborativo entre desarrolladores ubicados en diferentes partes del mundo.

Además de servir como un "espacio remoto" donde subir tu código, GitHub incluye herramientas para organizar, discutir, revisar y aprobar cambios, lo que lo convierte en una solución integral para el desarrollo de software moderno.

- **¿Cómo crear un repositorio en GitHub?**

- 1. Ingresa a GitHub**

Ve a <https://github.com> e inicia sesión con tu cuenta.

- 2. Crea un nuevo repositorio**

Haz clic en el botón "+" en la esquina superior derecha.

Selecciona "New repository".

- 3. Configura tu repositorio**

Rellena los siguientes campos:

**Repository name:** Elige un nombre para tu repositorio.

**Description (opcional):** Breve descripción del proyecto.

**Visibility:**

**Public:** Cualquiera puede verlo.

**Private:** Solo tú (o quienes invites) podrán verlo.

#### **4. Opciones adicionales**

Puedes marcar:

- “Initialize this repository with a README” si quieres que tenga un archivo README desde el inicio.

También puedes añadir un .gitignore (para ignorar archivos) y elegir una licencia.

#### **5. Crear**

Haz clic en el botón verde “Create repository”

- **¿Cómo crear una rama en Git?**

Comando: `git branch <nombre>`

- **¿Cómo cambiar a una rama en Git?**

Comando: `git checkout <nombre>`

- **¿Cómo fusionar ramas en Git?**

Comando: `git merge <nombre>`

- **¿Cómo crear un commit en Git?**

Comando: `git add .`

Comando: `git commit -m "mensaje"`

- **¿Cómo enviar un commit a GitHub?**

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

- **¿Qué es un repositorio remoto?**

Un repositorio remoto en Git es una versión de tu repositorio que está alojada en internet o en una red, generalmente en plataformas como GitHub, GitLab o Bitbucket. Este repositorio remoto permite que tú y otros colaboradores puedan trabajar juntos desde diferentes lugares.

- **¿Cómo agregar un repositorio remoto a Git?**

Comando: `git remote add origin url-del-repositorio`

- **¿Cómo empujar cambios a un repositorio remoto?**

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

- **¿Cómo tirar de cambios de un repositorio remoto?**

Comando: `git pull origin nombre-de-la-rama`

- **¿Qué es un fork de repositorio?**

Cuando haces un fork de un repositorio, creas una copia independiente del proyecto en tu cuenta de GitHub (u otra plataforma). Puedes trabajar en esa copia como deseas y realizar cambios sin interferir directamente con el repositorio original. • **¿Cómo crear un fork de un repositorio?**

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

**1. Sube tu rama con los cambios a tu fork en GitHub.**

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

## 2. Ve a tu repositorio en GitHub

Abre tu navegador y ve a tu fork en GitHub. Verás un mensaje que te sugiere "Compare & pull request".

## 3. Haz clic en "Compare & pull request"

Se abrirá la página para enviar una pull request.

Asegúrate de que la base (base repository) sea el repositorio original y la rama de destino, y que la comparación (compare) sea tu rama con los cambios.

## 4. Escribe un título y una descripción para tu pull request

Es importante ser claro en tu descripción. Explica qué cambios hiciste y por qué los estás proponiendo.

## 5. Envía la pull request

Cuando todo esté listo, haz clic en el botón "Create pull request".

## • ¿Cómo aceptar una solicitud de extracción?

### 1. Revisa la pull request

Primero, ve a tu repositorio en GitHub donde se ha creado la solicitud de extracción (pull request). Verás un ícono o un mensaje que te avisa sobre las pull requests abiertas. Haz clic en la pestaña "Pull requests" en la parte superior de la página del repositorio.

### 2. Selecciona la pull request que quieras aceptar

Verás una lista de las pull requests abiertas. Haz clic en la que deseas revisar y aceptar.

### 3. Revisa los cambios propuestos

Dentro de la pull request, puedes ver una descripción de los cambios propuestos, los archivos modificados y un diferencial (diff) que muestra las diferencias entre la rama propuesta y la rama principal. Puedes hacer comentarios y discutir los cambios con el colaborador antes de aceptarlos.

### 4. Resolver conflictos (si es necesario)

Si hay conflictos entre los cambios propuestos y tu rama principal (por ejemplo, entre tu rama main y la rama de la pull request), GitHub te lo indicará. Deberás resolver los conflictos manualmente en el editor de GitHub o localmente en tu máquina y luego hacer un nuevo commit.

**Para resolver los conflictos:**

Haz un pull de la rama de la pull request en tu repositorio local.

Resuelve los conflictos en tu editor.

Haz un commit para los cambios de resolución de conflictos.

Sube los cambios al repositorio remoto.

#### **5. Haz clic en "Merge pull request"**

Si los cambios son correctos y no hay conflictos, en la parte inferior de la pull request verás el botón "Merge pull request". Haz clic en este botón para aceptar la pull request.

#### **6. Elige el método de fusión**

Merge commit: Fusiona los cambios creando un commit de fusión. Esto es el valor predeterminado.

Squash and merge: Combina todos los commits de la pull request en un solo commit antes de fusionarlo en la rama principal.

Rebase and merge: Reaplica los commits de la pull request sobre la base de tu rama principal (es útil para mantener el historial más limpio).

Elige la opción que prefieras según las necesidades del proyecto.

#### **7. Confirma la fusión**

Después de hacer clic en "Merge pull request", GitHub te pedirá que confirmes la fusión. Haz clic en "Confirm merge".

#### **8. Eliminar la rama (opcional)**

Después de que la pull request haya sido fusionada, generalmente puedes eliminar la rama de la pull request (especialmente si era una rama de características específica).

GitHub te dará la opción de eliminar la rama automáticamente después de la fusión, o puedes hacerlo manualmente.

### • **¿Qué es un etiqueta en Git?**

Una etiqueta (o tag) en Git es una referencia o marcador que apunta a un punto específico en el historial del proyecto. Se usa principalmente para marcar versiones importantes o hitos en el proyecto, como lanzamientos (releases), versiones estables, o puntos donde se desea hacer una anotación especial.

- **¿Cómo crear una etiqueta en Git?**

Comando: `git tag nombre-de-etiqueta`

Etiqueta anotada:

Comando: `git tag -a nombre-de-etiqueta -m "Mensaje de la etiqueta"`

- **¿Cómo enviar una etiqueta a GitHub?**

Comando: `git push origin nombre-de-etiqueta`

- **¿Qué es un historial de Git?**

El historial de Git es una secuencia de commits que representan el registro completo de todos los cambios realizados en un repositorio de Git a lo largo del tiempo. Cada commit en el historial está asociado con un conjunto de cambios específicos que fueron realizados en el proyecto en un punto determinado, junto con información sobre quién hizo el cambio, cuándo lo hizo y por qué (generalmente en el mensaje del commit).

- **¿Cómo ver el historial de Git?**

Comando: `git log`

- **¿Cómo buscar en el historial de Git?**

Buscar por mensaje de commit: `git log --grep="término"`

Buscar por autor: `git log --author="nombre"`

Buscar por fecha: `git log --since="fecha" --until="fecha"`

Buscar en archivos específicos: `git log -- <archivo>`

Buscar en el contenido de los cambios: `git log -S "término"`

Ver un gráfico del historial: `git log --graph --oneline`

Ver un commit específico: `git log <commit-hash>`

- **¿Cómo borrar el historial de Git?**

Comando: `rm -rf .git`

- **¿Qué es un repositorio privado en GitHub?**

Un repositorio privado en GitHub es un tipo de repositorio que solo tú y las personas que tú autorices pueden ver y acceder. A diferencia de los repositorios públicos, que son visibles para cualquier persona en internet, los privados están ocultos para el público general.

- **¿Cómo crear un repositorio privado en GitHub?**

1. Inicia sesión en [github.com](https://github.com) y haz clic en el botón verde "New" o ve a: <https://github.com/new>.
2. Rellena los datos del repositorio: nombre, descripción, etc.
3. En la sección de visibilidad, selecciona Private.
4. (Opcional) Marca si quieres iniciar con README, .gitignore, etc.
5. Haz clic en "Create repository".

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

1. Ve a tu repositorio.
2. Haz clic en Settings > Collaborators and teams (o "Manage access").
3. Haz clic en "Invite a collaborator".
4. Escribe el nombre de usuario de GitHub de la persona y envía la invitación.

- **¿Qué es un repositorio público en GitHub?**

Un repositorio público en GitHub es un repositorio que cualquier persona puede ver, clonar y descargar desde internet. Es completamente abierto y visible para todos, aunque solo tú y tus colaboradores autorizados pueden hacer cambios directamente (a menos que aceptes contribuciones por pull requests).

- **¿Cómo crear un repositorio público en GitHub?**

1. Entra a [github.com](https://github.com) e inicia sesión.
2. Haz clic en el botón verde "New" (o ve a [github.com/new](https://github.com/new)).
3. Escribe el nombre del repositorio.
4. En la sección de visibilidad, elige Public.
5. Haz clic en "Create repository".

- **¿Cómo compartir un repositorio público en GitHub?**

1. Ve a tu repositorio en GitHub.
2. Copia la URL del navegador. Por ejemplo:

<https://github.com/tu-usuario/tu-repositorio>

Compártela por WhatsApp, correo, redes sociales, etc.

Cualquiera con ese enlace puede:

- Ver el código fuente.
- Clonar el repositorio con Git.
- Hacer sugerencias o pull requests

**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.

The screenshot shows the GitHub homepage with a dark theme. At the top left, there's a navigation bar with links like Egg LXP, Bootstrap v5.3, Gmail, WhatsApp, FIN, freeCodeCamp, Adobe Express, TensorArt | FREE..., ClipDrop, CSS & Tailwind, ChatGPT, Windguru, and KIREINA. The main header says "Dashboard". On the left, there's a sidebar titled "Top repositories" with a "New" button highlighted by a red circle. Below it is a search bar with placeholder text "Find a repository...". A repository card for "MartinRotolo/UTN-TUPaD-P1" is visible. The central area has a "Home" section with a "Ask Copilot" input field and a "Recent commits in torvalds/linux" link. To the right, there's a "GitHub Copilot" section with a "Available for free" message and a "Open Copilot" button. Below that is an "Explore repositories" section listing "weecology / DeepForest", "supabase / supabase", and "keepassxreboot / keepassxc".

## 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 \*** **Repository name \***  -  primer-repo-git is available.

Great repository names are short and memorable. Need inspiration? How about [turbo-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**

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](#).

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.

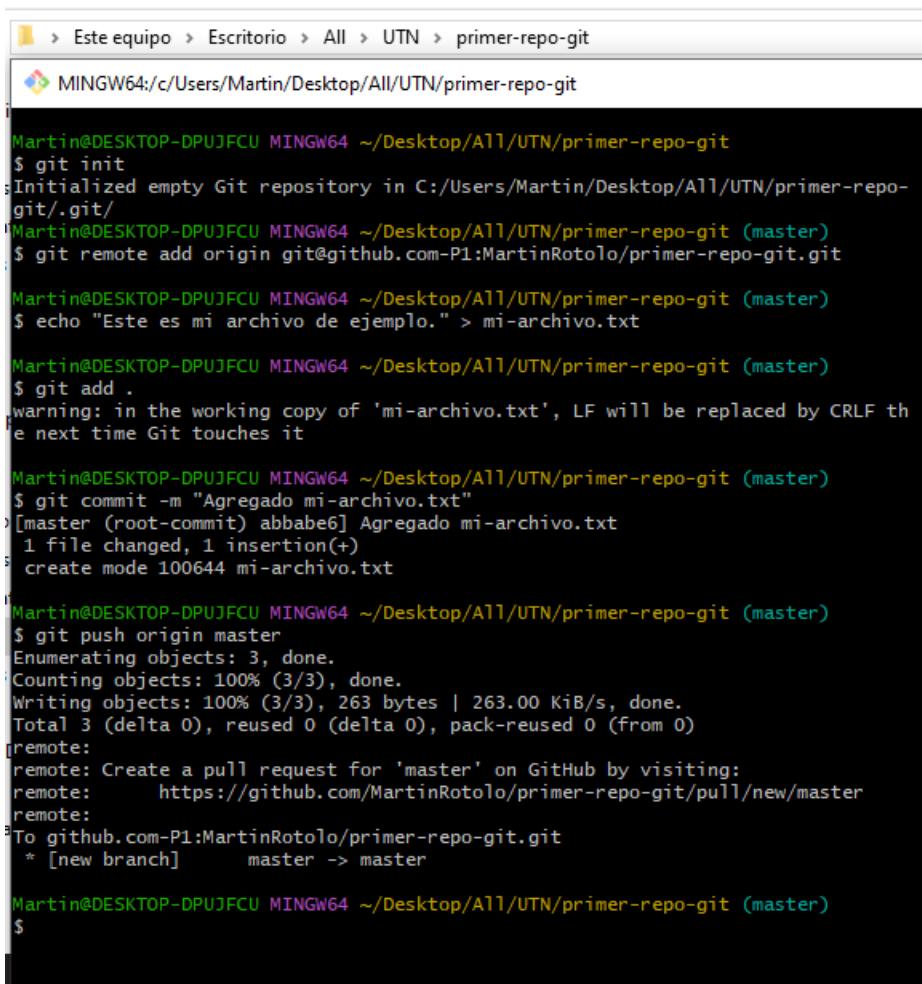
**Create repository**

- 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.

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



The screenshot shows a terminal window with the following session:

```
Martin@DESKTOP-DPUJFCU MINGW64 ~/Desktop/All/UTN/primer-repo-git
$ git init
:Initialized empty Git repository in C:/Users/Martin/Desktop/All/UTN/primer-repo-git/.git/
Martin@DESKTOP-DPUJFCU MINGW64 ~/Desktop/All/UTN/primer-repo-git (master)
$ git remote add origin git@github.com:P1:MartinRotolo/primer-repo-git.git
Martin@DESKTOP-DPUJFCU MINGW64 ~/Desktop/All/UTN/primer-repo-git (master)
$ echo "Este es mi archivo de ejemplo." > mi-archivo.txt

Martin@DESKTOP-DPUJFCU MINGW64 ~/Desktop/All/UTN/primer-repo-git (master)
$ git add .
warning: in the working copy of 'mi-archivo.txt', LF will be replaced by CRLF the next time Git touches it

Martin@DESKTOP-DPUJFCU MINGW64 ~/Desktop/All/UTN/primer-repo-git (master)
$ git commit -m "Agregado mi-archivo.txt"
>[master (root-commit) abbabe6] Agregado mi-archivo.txt
 1 file changed, 1 insertion(+)
 create mode 100644 mi-archivo.txt

Martin@DESKTOP-DPUJFCU MINGW64 ~/Desktop/All/UTN/primer-repo-git (master)
$ git push origin master
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Writing objects: 100% (3/3), 263 bytes | 263.00 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/MartinRotolo/primer-repo-git/pull/new/master
remote:
To github.com:P1:MartinRotolo/primer-repo-git.git
 * [new branch]      master -> master

Martin@DESKTOP-DPUJFCU MINGW64 ~/Desktop/All/UTN/primer-repo-git (master)
$
```

- Creando Branchs

- o Crear una Branch

- o Realizar cambios o agregar un archivo

## o Subir la Branch

```
MINGW64:/c/Users/Martin/Desktop/All/UTN/primer-repo-git
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/MartinRotolo/primer-repo-git/pull/new/master
remote:
To github.com:P1:MartinRotolo/primer-repo-git.git
 * [new branch]      master -> master

Martin@DESKTOP-DPUJFCU MINGW64 ~/Desktop/All/UTN/primer-repo-git (master)
$ git checkout -b nuevarama
Switched to a new branch 'nuevarama'

Martin@DESKTOP-DPUJFCU MINGW64 ~/Desktop/All/UTN/primer-repo-git (nuevarama)
$ echo "Cambios desde una rama" > cambios.txt

Martin@DESKTOP-DPUJFCU MINGW64 ~/Desktop/All/UTN/primer-repo-git (nuevarama)
$ git add .
warning: in the working copy of 'cambios.txt', LF will be replaced by CRLF the next time Git touches it

Martin@DESKTOP-DPUJFCU MINGW64 ~/Desktop/All/UTN/primer-repo-git (nuevarama)
$ git commit -m "agregado cambios.txt desde nueva rama"
[nuevarama 439c3c3] agregado cambios.txt desde nueva rama
 1 file changed, 1 insertion(+)
 create mode 100644 cambios.txt

Martin@DESKTOP-DPUJFCU MINGW64 ~/Desktop/All/UTN/primer-repo-git (nuevarama)
$ git push origin nuevarama
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), 328 bytes | 328.00 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/MartinRotolo/primer-repo-git/pull/new/nuevarama
remote:
To github.com:P1:MartinRotolo/primer-repo-git.git
 * [new branch]      nuevarama -> nuevarama

Martin@DESKTOP-DPUJFCU MINGW64 ~/Desktop/All/UTN/primer-repo-git (nuevarama)
$ |
```

### 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 \*** **Repository name \***

 MartinRotolo / conflict-exercise conflict-exercise is available.

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

**Description (optional)**  
Mi segundo repo P1

 **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/tuusuuario/conflict-exercise.git>).

- Abre la terminal o línea de comandos en tu máquina.

- Clona el repositorio usando el comando:

```
git clone https://github.com/tuusuuario/conflict-exercise.git
```

- Entra en el directorio del repositorio:

```
cd conflict-exercise
```

```
MINGW64:/c/Users/Martin/Desktop/All/UTN/Programacion 1/Repositorios/Segundo-repo/conflict-exercise
Martin@DESKTOP-DPUJFCU MINGW64 ~/Desktop/All/UTN/Programacion 1/Repositorios/Segundo-repo
$ git init
Initialized empty Git repository in C:/Users/Martin/Desktop/All/UTN/Programacion 1/Repositorios/Segundo-repo/.git/
Martin@DESKTOP-DPUJFCU MINGW64 ~/Desktop/All/UTN/Programacion 1/Repositorios/Segundo-repo (master)
$ git clone https://github.com/MartinRotolo/conflict-exercise
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.

Martin@DESKTOP-DPUJFCU MINGW64 ~/Desktop/All/UTN/Programacion 1/Repositorios/Segundo-repo (master)
$ cd conflict-exercise

Martin@DESKTOP-DPUJFCU MINGW64 ~/Desktop/All/UTN/Programacion 1/Repositorios/Segundo-repo/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"**

```
MINGW64:/c/Users/Martin/Desktop/All/UTN/Programacion 1/Repositorios/Segundo-repo/conflict-exercise
undo-repo
$ git init
Initialized empty Git repository in C:/Users/Martin/Desktop/All/UTN/Programacion 1/Repositorios/Segundo-repo/.git

Martin@DESKTOP-DPUJFCU MINGW64 ~/Desktop/All/UTN/Programacion 1/Repositorios/Segundo-repo (master)
$ git clone https://github.com/MartinRotolo/conflict-exercise
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.

Martin@DESKTOP-DPUJFCU MINGW64 ~/Desktop/All/UTN/Programacion 1/Repositorios/Segundo-repo (master)
$ cd conflict-exercise

Martin@DESKTOP-DPUJFCU MINGW64 ~/Desktop/All/UTN/Programacion 1/Repositorios/Segundo-repo/conflict-exercise (main)
$ git checkout -b feabute-branch
Switched to a new branch 'feabute-branch'

Martin@DESKTOP-DPUJFCU MINGW64 ~/Desktop/All/UTN/Programacion 1/Repositorios/Segundo-repo/conflict-exercise (feabute-branch)
$ git add README.md

Martin@DESKTOP-DPUJFCU MINGW64 ~/Desktop/All/UTN/Programacion 1/Repositorios/Segundo-repo/conflict-exercise (feabute-branch)
$ git commit -m "Added a line in feature-branch"
[feabute-branch 5b7d1d8] Added a line in feature-branch
 1 file changed, 1 insertion(+)

Martin@DESKTOP-DPUJFCU MINGW64 ~/Desktop/All/UTN/Programacion 1/Repositorios/Segundo-repo/conflict-exercise (feabute-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"**

```
MINGW64:/c/Users/Martin/Desktop/All/UTN/Programacion 1/Repositorios/Segundo-repo/conflict-exercise
Switched to a new branch 'feabute-branch'

Martin@DESKTOP-DPUJFCU MINGW64 ~/Desktop/All/UTN/Programacion 1/Repositorios/Segundo-repo/conflict-exercise (feabute-branch)
$ git add README.md

Martin@DESKTOP-DPUJFCU MINGW64 ~/Desktop/All/UTN/Programacion 1/Repositorios/Segundo-repo/conflict-exercise (feabute-branch)
$ git commit -m "Added a line in feature-branch"
[feabute-branch 5b7d1d8] Added a line in feature-branch
 1 file changed, 1 insertion(+)

Martin@DESKTOP-DPUJFCU MINGW64 ~/Desktop/All/UTN/Programacion 1/Repositorios/Segundo-repo/conflict-exercise (feabute-branch)
$ git checkout master
error: pathspec 'master' did not match any file(s) known to git

Martin@DESKTOP-DPUJFCU MINGW64 ~/Desktop/All/UTN/Programacion 1/Repositorios/Segundo-repo/conflict-exercise (feabute-branch)
$ git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.

Martin@DESKTOP-DPUJFCU MINGW64 ~/Desktop/All/UTN/Programacion 1/Repositorios/Segundo-repo/conflict-exercise (main)
$ git add README.md

Martin@DESKTOP-DPUJFCU MINGW64 ~/Desktop/All/UTN/Programacion 1/Repositorios/Segundo-repo/conflict-exercise (main)
$ git commit -m "Added a line in main branch"
[main 981bdce] Added a line in main branch
 1 file changed, 1 insertion(+)

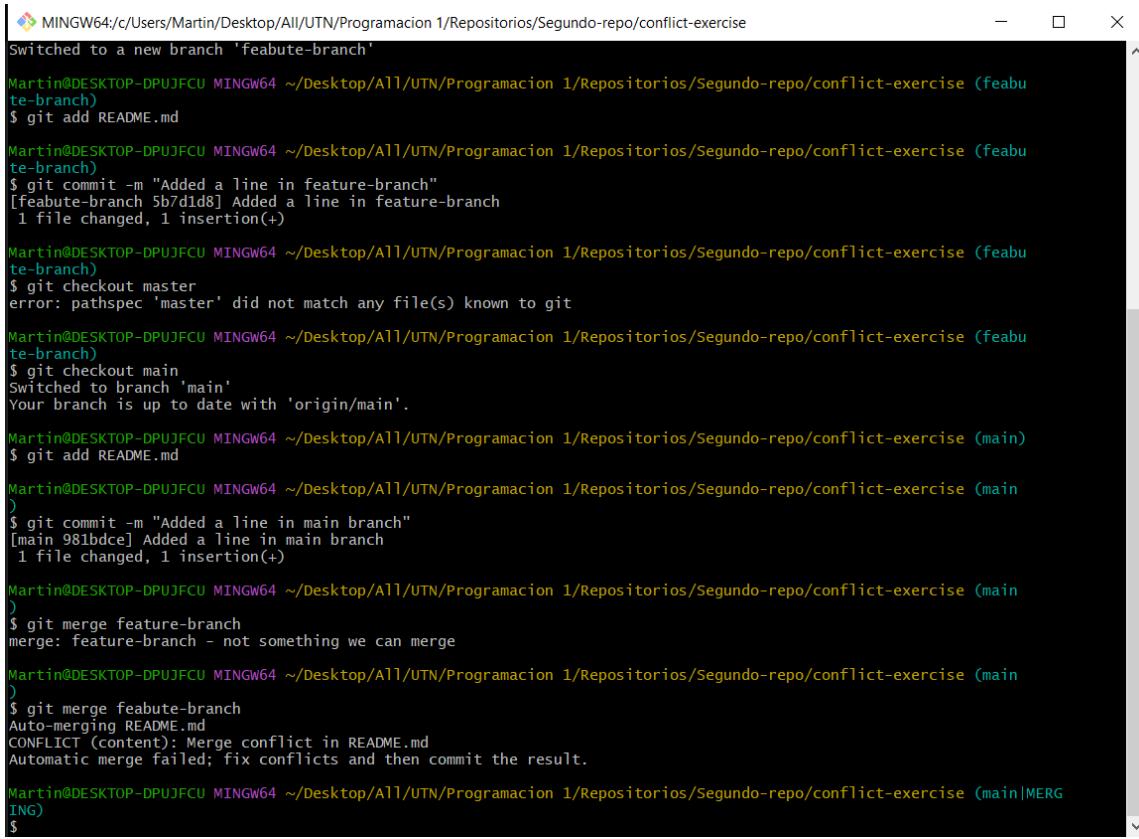
Martin@DESKTOP-DPUJFCU MINGW64 ~/Desktop/All/UTN/Programacion 1/Repositorios/Segundo-repo/conflict-exercise (main)
$
```

## 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.



```
MINGW64:/c/Users/Martin/Desktop/All/UTN/Programacion 1/Repositorios/Segundo-repo/conflict-exercise
Switched to a new branch 'feabute-branch'

Martin@DESKTOP-DPUJFCU MINGW64 ~/Desktop/All/UTN/Programacion 1/Repositorios/Segundo-repo/conflict-exercise (feabute-branch)
$ git add README.md

Martin@DESKTOP-DPUJFCU MINGW64 ~/Desktop/All/UTN/Programacion 1/Repositorios/Segundo-repo/conflict-exercise (feabute-branch)
$ git commit -m "Added a line in feature-branch"
[feabute-branch 5b7d1d8] Added a line in feature-branch
 1 file changed, 1 insertion(+)

Martin@DESKTOP-DPUJFCU MINGW64 ~/Desktop/All/UTN/Programacion 1/Repositorios/Segundo-repo/conflict-exercise (feabute-branch)
$ git checkout master
error: pathspec 'master' did not match any file(s) known to git

Martin@DESKTOP-DPUJFCU MINGW64 ~/Desktop/All/UTN/Programacion 1/Repositorios/Segundo-repo/conflict-exercise (feabute-branch)
$ git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.

Martin@DESKTOP-DPUJFCU MINGW64 ~/Desktop/All/UTN/Programacion 1/Repositorios/Segundo-repo/conflict-exercise (main)
$ git add README.md

Martin@DESKTOP-DPUJFCU MINGW64 ~/Desktop/All/UTN/Programacion 1/Repositorios/Segundo-repo/conflict-exercise (main)
$ git commit -m "Added a line in main branch"
[main 981bdce] Added a line in main branch
 1 file changed, 1 insertion(+)

Martin@DESKTOP-DPUJFCU MINGW64 ~/Desktop/All/UTN/Programacion 1/Repositorios/Segundo-repo/conflict-exercise (main)
$ git merge feature-branch
merge: feature-branch - not something we can merge

Martin@DESKTOP-DPUJFCU MINGW64 ~/Desktop/All/UTN/Programacion 1/Repositorios/Segundo-repo/conflict-exercise (main)
$ git merge feabute-branch
Auto-merging README.md
CONFLICT (content): Merge conflict in README.md
Automatic merge failed; fix conflicts and then commit the result.

Martin@DESKTOP-DPUJFCU MINGW64 ~/Desktop/All/UTN/Programacion 1/Repositorios/Segundo-repo/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

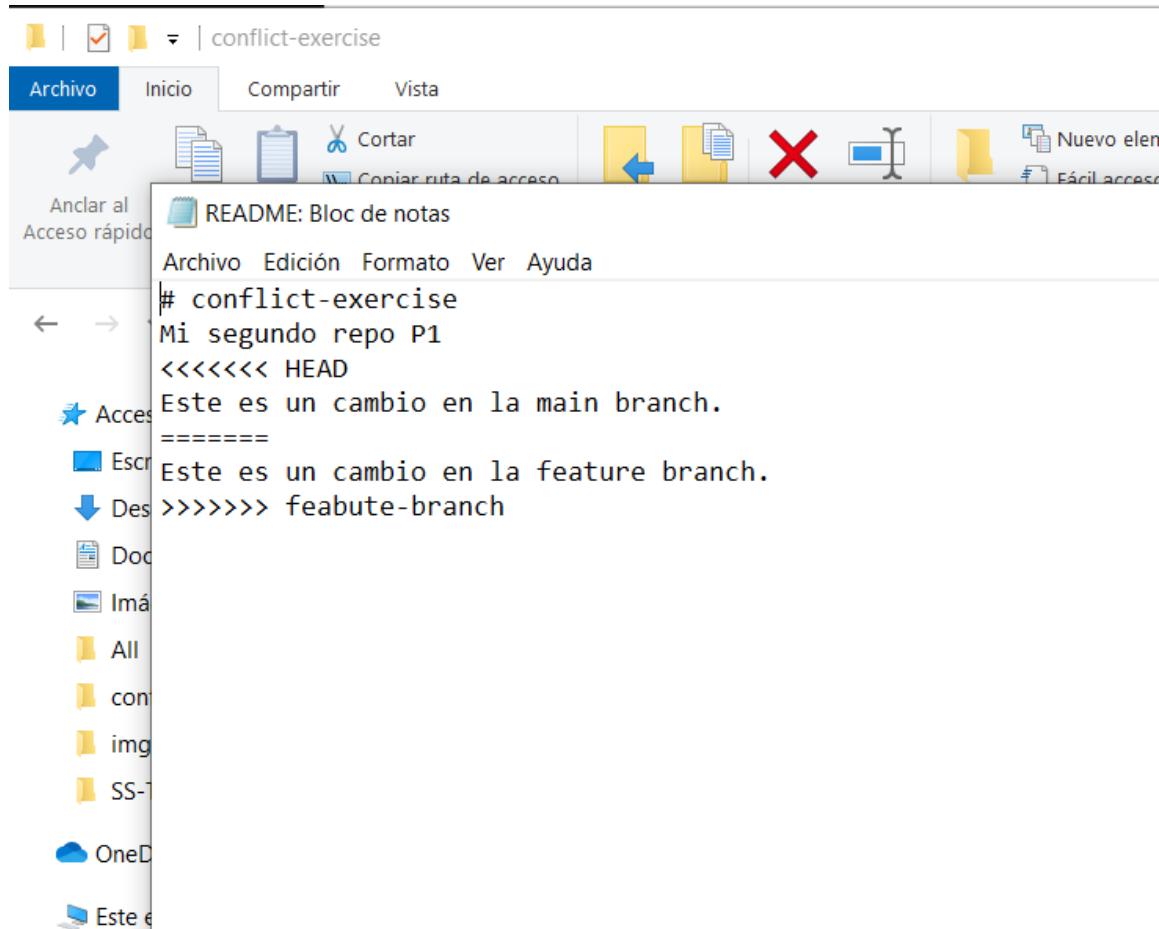
- Decide cómo resolver el conflicto. Puedes mantener ambos cambios, elegir uno de ellos, o fusionar los contenidos de alguna manera.
- Edita el archivo para resolver el conflicto y guarda los cambios (Se debe borrar lo marcado en verde en el archivo donde estes solucionando el conflicto. Y se

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"
```



```
MINGW64:/c/Users/Martin/Desktop/All/UTN/Programacion 1/Repositorios/Segundo-repo/conflict-exercise
te-branch)
$ git commit -m "Added a line in feature-branch"
[feabute-branch 5b7d1d8] Added a line in feature-branch
1 file changed, 1 insertion(+)

Martin@DESKTOP-DPUJFCU MINGW64 ~/Desktop/All/UTN/Programacion 1/Repositorios/Segundo-repo/conflict-exercise (feabute-branch)
$ git checkout master
error: pathspec 'master' did not match any file(s) known to git
:
Martin@DESKTOP-DPUJFCU MINGW64 ~/Desktop/All/UTN/Programacion 1/Repositorios/Segundo-repo/conflict-exercise (feabute-branch)
$ git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.

Martin@DESKTOP-DPUJFCU MINGW64 ~/Desktop/All/UTN/Programacion 1/Repositorios/Segundo-repo/conflict-exercise (main)
$ git add README.md

Martin@DESKTOP-DPUJFCU MINGW64 ~/Desktop/All/UTN/Programacion 1/Repositorios/Segundo-repo/conflict-exercise (main)
$ git commit -m "Added a line in main branch"
[main 981bdce] Added a line in main branch
1 file changed, 1 insertion(+)

Martin@DESKTOP-DPUJFCU MINGW64 ~/Desktop/All/UTN/Programacion 1/Repositorios/Segundo-repo/conflict-exercise (main)
$ git merge feature-branch
merge: feature-branch - not something we can merge
Martin@DESKTOP-DPUJFCU MINGW64 ~/Desktop/All/UTN/Programacion 1/Repositorios/Segundo-repo/conflict-exercise (main)
$ git merge feabute-branch
Auto-merging README.md
CONFLICT (content): Merge conflict in README.md
Automatic merge failed; fix conflicts and then commit the result.

Martin@DESKTOP-DPUJFCU MINGW64 ~/Desktop/All/UTN/Programacion 1/Repositorios/Segundo-repo/conflict-exercise (main|MERGING)
$ git add README.md

Martin@DESKTOP-DPUJFCU MINGW64 ~/Desktop/All/UTN/Programacion 1/Repositorios/Segundo-repo/conflict-exercise (main|MERGING)
$ git commit -m "Resolved merge conflict"
[main 93241ec] Resolved merge conflict

Martin@DESKTOP-DPUJFCU MINGW64 ~/Desktop/All/UTN/Programacion 1/Repositorios/Segundo-repo/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**

```
MINGW64:/c/Users/Martin/Desktop/All/UTN/Programacion 1/Repositorios/Segundo-repo/conflict-exercise (main)
$ git push origin main
ERROR: Permission to MartinRotolo/conflict-exercise.git denied to centroserviziostiaantica.
fatal: Could not read from remote repository.

Please make sure you have the correct access rights
and the repository exists.

Martin@DESKTOP-DPUJFCU MINGW64 ~/Desktop/All/UTN/Programacion 1/Repositorios/Segundo-repo/conflict-exercise (main)
$ git remote set-url origin git@github.com:MartinRotolo/conflict-exercise.git

Martin@DESKTOP-DPUJFCU MINGW64 ~/Desktop/All/UTN/Programacion 1/Repositorios/Segundo-repo/conflict-exercise (main)
$ git push origin main
Enumerating objects: 9, done.
Counting objects: 100% (9/9), done.
Delta compression using up to 16 threads
Compressing objects: 100% (5/5), done.
Writing objects: 100% (7/7), 692 bytes | 692.00 KiB/s, done.
Total 7 (delta 2), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (2/2), done.
To github.com:MartinRotolo/conflict-exercise.git
  1a862b4..93241ec main -> main

Martin@DESKTOP-DPUJFCU MINGW64 ~/Desktop/All/UTN/Programacion 1/Repositorios/Segundo-repo/conflict-exercise (main)
$ git push origin feabute-branch
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
remote:
remote: Create a pull request for 'feabute-branch' on GitHub by visiting:
remote:     https://github.com/MartinRotolo/conflict-exercise/pull/new/feabute-branch
remote:
To github.com:MartinRotolo/conflict-exercise.git
 * [new branch]      feabute-branch -> feabute-branch

Martin@DESKTOP-DPUJFCU MINGW64 ~/Desktop/All/UTN/Programacion 1/Repositorios/Segundo-repo/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.

 **conflict-exercise** Public

main ▾ 2 Branches 0 Tags

Go to file Add file <> Code

 centroserviziostiaantica Resolved merge conflict 93241ec · 37 minutes ago 4 Commits

README.md Added a line in main branch 45 minutes ago

 README

## conflict-exercise

Mi segundo repo P1 Este es un cambio en la main branch.

About Mi segundo repo P1

Readme Activity 0 stars 1 watching 0 forks

Releases No releases published Create a new release

Packages No packages published Publish your first package

Contributors 2

 centroserviziostiaantica

 MartinRotolo

### Commits

main ▾ All users All time

Commits on Apr 11, 2025

**Resolved merge conflict** 93241ec

 centroserviziostiaantica committed 38 minutes ago

**Added a line in main branch** 981bdce

 centroserviziostiaantica committed 47 minutes ago

**Added a line in feature-branch** 5b7did8

 centroserviziostiaantica committed 50 minutes ago

**Initial commit** 1a862b4

 MartinRotolo authored 1 hour ago Verified