

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

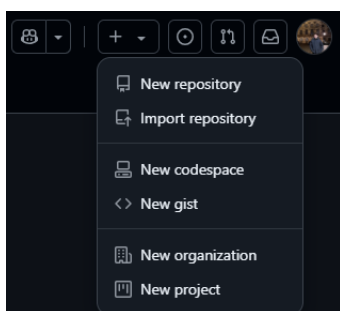
- 1) Contestar las siguientes preguntas utilizando las guías y documentación proporcionada (Desarrollar las respuestas) :

- **¿Qué es GitHub?**

GitHub es una plataforma basada en la nube donde puedes almacenar, compartir y trabajar junto con otros usuarios para escribir código

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

En la esquina superior derecha de cualquier página, selecciona el icono + (Create new...) y luego haz clic en Nuevo repositorio.

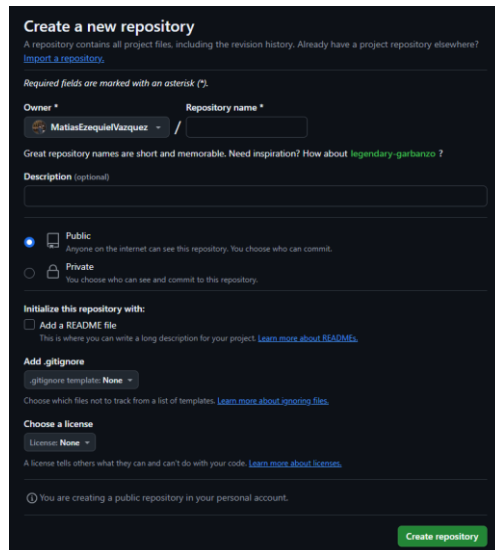


En el cuadro "Nombre del repositorio", escriba un nombre para el proyecto.

En el cuadro "Descripción", escriba una breve descripción.

Seleccione si el repositorio será Público o Privado.

Haga clic en Create repository



- **¿Cómo crear una rama en Git?**

Escriba el comando git branch “nombre de la rama” en su terminal para crear una nueva rama

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

Escriba el comando git checkout “nombre de la rama” en su terminal para cambiar a una rama en GIT

- **¿Cómo fusionar ramas en Git?**

Ubíquese en un rama (ejemplo “Main”) y escriba el comando git merge “rama_ejemplo” a fin de fusionar “rama_ejemplo” con “Main”.

- **¿Cómo crear un commit en Git?**

Escriba el comando git commit una vez el archivo se encuentra en el staging área, puede agregar el comando -m para agregar un mensaje descriptivo al commit.

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

Escriba el comando `git push -u origin master` si es la primera vez que sube los cambios a el repositorio remoto, luego de la primera vez solo debe escribir el comando `git push`

- **¿Qué es un repositorio remoto?**

Un repositorio comprende toda la colección de archivos y carpetas asociados con un proyecto, en conjunto con el historial de revisión de cada archivo. GitHub hospeda repositorios de Git en la nube, estos mismos se denominan repositorios remotos.

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

Escriba el comando `git remote add origin "url del repositorio remoto"` en la terminal desde la ubicación donde se encuentra iniciado su repositorio GIT.

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

Escriba el comando `git push` para empujar los cambios a un repositorio remoto luego de su realizar el commit en git si es que el comando `git remote add origin "url del repositorio remoto"` ya se ejecuto.

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

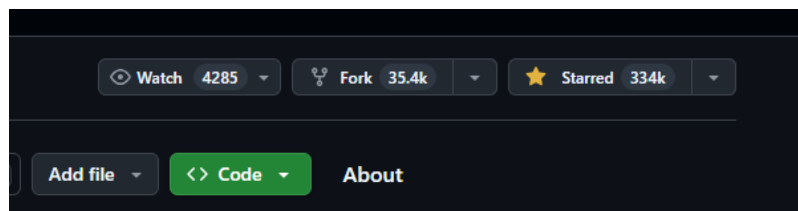
Escriba el comando `git pull` para aplicar cambios desde un repositorio remoto.

- **¿Qué es un fork de repositorio?**

Una bifurcación (fork) es un nuevo repositorio que comparte la configuración de visibilidad y código con el repositorio original y es copiado a su perfil de GitHub.

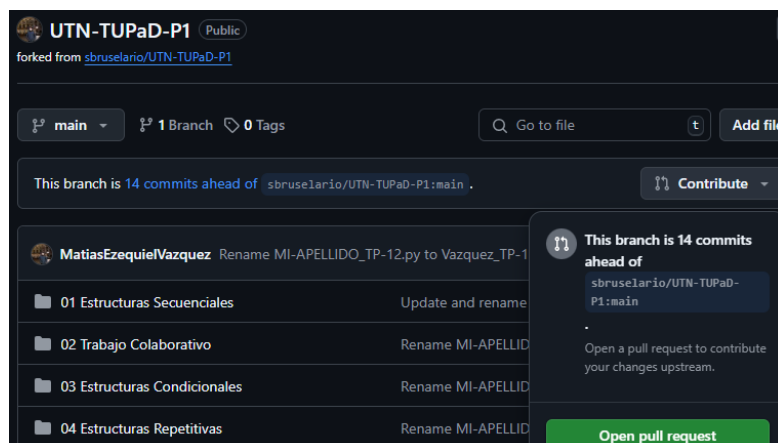
- **¿Cómo crear un fork de un repositorio?**

Haga click en el botón de "Fork" dentro de GitHub y se le solicitara agregar un nombre y descripción a su copia del repositorio original.



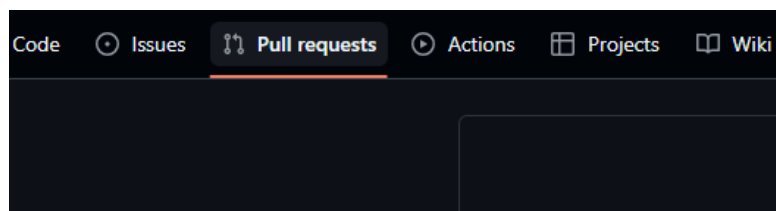
- **¿Cómo enviar una solicitud de extracción (pull request) a un repositorio?**

Dentro de GitHub una vez se encuentre en su repositorio el cual creo en base a un fork puede seleccionar la opción Contribute > Open Pull Request para enviar la solicitud de extracción a un repositorio.



- **¿Cómo aceptar una solicitud de extracción?**

Navegar hasta la sección Pull requests y seleccionar la PR que se desea aceptar, se confirmara esto una vez se haga click en las opciones Merge Pull request > Confirm merge.



- **¿Qué es un etiqueta en Git?**

Una etiqueta es una referencia que apunta a un commit específico, usualmente utilizadas para dejar asentados cambios importantes en el historial de un proyecto.

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

Escriba el comando `git tag "mensaje_tag"` en donde se agregara el tag al commit actual.

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

Escriba el comando `git push origin "ejemplo_tag"` para enviar un tag específico al repositorio remoto o utilizar el comando `git push --tags` para enviar todas las etiquetas.

- **¿Qué es un historial de Git?**

En GIT el historial contiene información de los cambios realizados a lo largo del tiempo en el repositorio, cada commit queda registrado con información relevante como mensajes asociados, autor, archivos modificados, hash (identificador único del commit), fecha y hora, etc.

- **¿Cómo ver el historial de Git?**

Escriba el comando git log para ver el historial en Git.

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

Existen diferentes formas de buscar dentro del historial de Git, dependiendo las necesidades se pueden utilizar los siguientes comandos:

git log → Ver commits

git log --oneline → Ver commits (una línea c/u)

git log --decorate --all --graph --oneline → Ver commits (graficado)

git log ramaB..ramaA → Ver commits en ramaA que no estén en ramaB

git log --follow archivo → Ver commits donde el archivo cambió

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

Para borrar o modificar el historial de Git se suele utilizar el comando git reset en cualquiera de sus tres opciones dependiendo las necesidades:

git reset --soft | git reset --mixed | git reset --hard

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

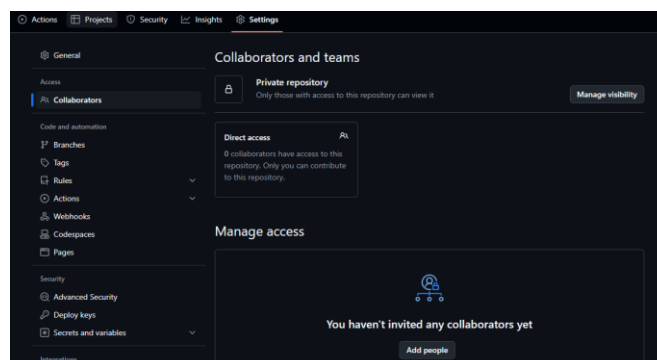
Un repositorio privado en GitHub es un repositorio remoto en donde el acceso esta restringido al creador y usuarios invitados a colaborar en el mismo.

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

Al momento de crear un nuevo repositorio en GitHub deberá seleccionar la opción “Private”, por defecto esta seleccionada la opción “Public”.

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

Acceda al repositorio, en la pestaña Settings seleccione Collaborators y agregue a los usuarios con los que quiera compartir el repositorio privado.



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

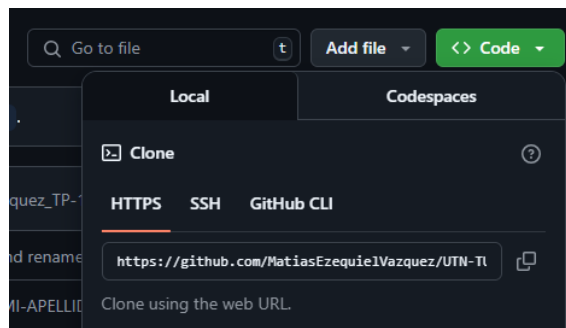
Por el contrario, a los repositorios Privados, un repositorio Publico en GitHub es un accesible para cualquier persona en internet, el contenido del repositorio es visible para cualquier usuario.

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

Al momento de crear un nuevo repositorio en GitHub deberá seleccionar la opción “Public”, por defecto la misma se encuentra seleccionada.

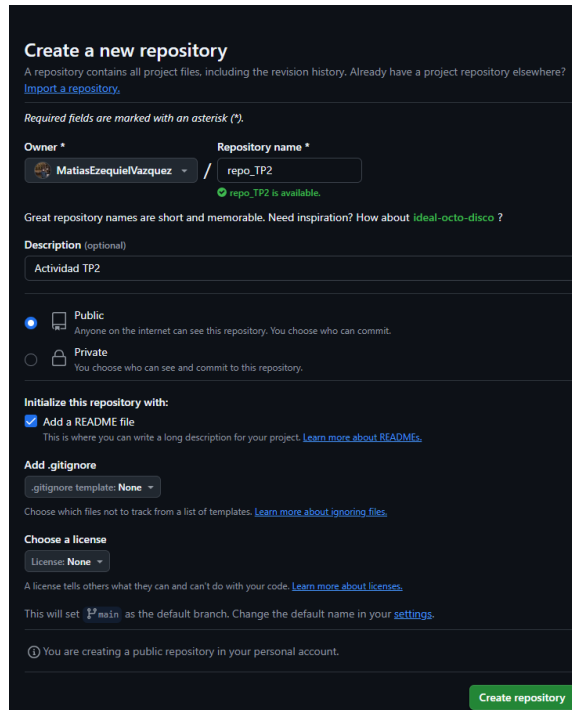
- **¿Cómo compartir un repositorio público en GitHub?**

Simplemente comparta el URL del repositorio, accesible desde Repositorio > Code.



2) Realizar la siguiente actividad:

- Crear un repositorio.
 - Dale un nombre al repositorio.
 - Elije que 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).

```
zuraAD+MatiasEzequielVazquez@LAPTOP-S7NMOM0J MINGW64 ~/source/UTN/TP2 Programaci
on1 (main)
$ git add mi-archivo.txt

zuraAD+MatiasEzequielVazquez@LAPTOP-S7NMOM0J MINGW64 ~/source/UTN/TP2 Programaci
on1 (main)
$ git status
On branch main

no commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   mi-archivo.txt
```

```
AzureAD+MatiasEzequielVazquez@LAPTOP-S7NMOM0J MINGW64 ~/source/UTN/TP2 Programaci
on1 (main)
$ git status
On branch main

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
    new file:   mi-archivo.txt

AzureAD+MatiasEzequielVazquez@LAPTOP-S7NMOM0J MINGW64 ~/source/UTN/TP2 Programaci
on1 (main)
$ git commit -m "primer commit"
[main (root-commit) 532331c] primer commit
 1 file changed, 1 insertion(+)
 create mode 100644 mi-archivo.txt

AzureAD+MatiasEzequielVazquez@LAPTOP-S7NMOM0J MINGW64 ~/source/UTN/TP2 Programaci
on1 (main)
$ git remote add origin https://github.com/MatiasEzequielVazquez/repo_TP2.git

AzureAD+MatiasEzequielVazquez@LAPTOP-S7NMOM0J MINGW64 ~/source/UTN/TP2 Programaci
on1 (main)
$ git push origin main
Enumerating objects: 6, done.
Counting objects: 100% (6/6), done.
Delta compression using up to 12 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (5/5), 541 bytes | 60.00 KiB/s, done.
Total 5 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/MatiasEzequielVazquez/repo_TP2.git
 33b842a..0e20ef2  main -> main
```

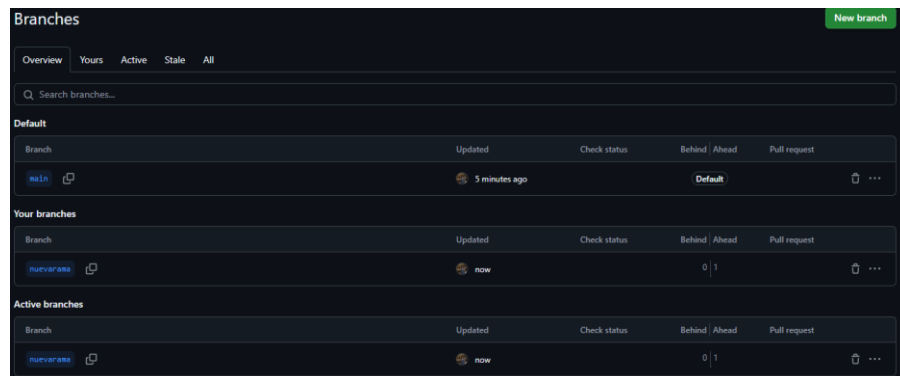
- Creando Branchs
 - Crear una Branch
 - Realizar cambios o agregar un archivo
 - Subir la Branch

```
AzureAD+MatiasEzequielVazquez@LAPTOP-S7NMOM0J MINGW64 ~/source/UTN/TP2 Programacion1 (main)
$ git checkout -b nuevarama
Switched to a new branch 'nuevarama'
```

```
AzureAD+MatiasEzequielVazquez@LAPTOP-S7NMOM0J MINGW64 ~/source/UTN/TP2 Programacion1 (nuevarama)
$ git commit -m "cambios guardados en ramanueva"
[nuevarama e027241] cambios guardados en ramanueva
 1 file changed, 2 insertions(+), 1 deletion(-)
```

```
AzureAD+MatiasEzequielVazquez@LAPTOP-S7NMOM0J MINGW64 ~/source/UTN/TP2 Programacion1 (nuevarama)
$ git push origin nuevarama
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 12 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 318 bytes | 35.00 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/MatiasEzequielVazquez/repo_TP2/pull/new/nuevarama
remote:
To https://github.com/MatiasEzequielVazquez/repo_TP2.git
 * [new branch]   nuevarama -> nuevarama
```

https://github.com/MatiasEzequielVazquez/repo_TP2.git



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 ***

 MatiasEzequielVazquez /

✓ conflict-exercise is available.

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

① 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**

```
AzureAD+MatiasEzequielVazque@LAPTOP-S7NMOM03 MINGW64 ~/source/UTN/TP2 - Actividad 3 (main)
$ git init
Initialized empty Git repository in C:/Users/MatiasEzequielVazque/source/UTN/TP2 - Actividad 3/.git/

AzureAD+MatiasEzequielVazque@LAPTOP-S7NMOM03 MINGW64 ~/source/UTN/TP2 - Actividad 3 (master)
$ git clone https://github.com/MatiasEzequielVazquez/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.

AzureAD+MatiasEzequielVazque@LAPTOP-S7NMOM03 MINGW64 ~/source/UTN/TP2 - Actividad 3 (master)
$ git branch -m master main

AzureAD+MatiasEzequielVazque@LAPTOP-S7NMOM03 MINGW64 ~/source/UTN/TP2 - Actividad 3 (main)
$ cd ./conflict-exercise

AzureAD+MatiasEzequielVazque@LAPTOP-S7NMOM03 MINGW64 ~/source/UTN/TP2 - Actividad 3/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"

```
AzureAD+MatiasEzequielVazque@LAPTOP-S7NMOM03 MINGW64 ~/source/UTN/TP2 - Actividad 3/conflict-exercise (main)
$ git checkout -b feature-branch
Switched to a new branch 'feature-branch'

AzureAD+MatiasEzequielVazque@LAPTOP-S7NMOM03 MINGW64 ~/source/UTN/TP2 - Actividad 3/conflict-exercise (feature-branch)
$ git status
On branch feature-branch
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
    modified:   README.md

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

AzureAD+MatiasEzequielVazque@LAPTOP-S7NMOM03 MINGW64 ~/source/UTN/TP2 - Actividad 3/conflict-exercise (feature-branch)
$ git add README.md

AzureAD+MatiasEzequielVazque@LAPTOP-S7NMOM03 MINGW64 ~/source/UTN/TP2 - Actividad 3/conflict-exercise (feature-branch)
$ git commit -m "Added a line in feature-branch"
[feature-branch 6e3caf5] Added a line in feature-branch
1 file changed, 1 insertion(+)


```

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  # conflict-exercise
2  Repo actividad 3 - TP2
3  Este es un cambio en la main branch.
4

$ git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.

$ git status
On branch main
Your branch is up to date with 'origin/main'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   README.md

no changes added to commit (use "git add" and/or "git commit -a")
$ git add README.md
$ git commit -m "Added a line in main branch"
[main 54f3578] Added a line in main branch
1 file changed, 1 insertion(+)
```

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.

```
$ 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.
```

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"
```

```
AzureAD+MatiasEzequielVazquez@LAPTOP-S7NMOM03 MINGW64 ~/source/UTN/TP2 - Actividad 3/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.

AzureAD+MatiasEzequielVazquez@LAPTOP-S7NMOM03 MINGW64 ~/source/UTN/TP2 - Actividad 3/conflict-exercise (main|MERGING)
$ git add README.md

AzureAD+MatiasEzequielVazquez@LAPTOP-S7NMOM03 MINGW64 ~/source/UTN/TP2 - Actividad 3/conflict-exercise (main|MERGING)
$ git commit -m "Resolved merge conflict"
[main 8bf2bf9] Resolved merge conflict

AzureAD+MatiasEzequielVazquez@LAPTOP-S7NMOM03 MINGW64 ~/source/UTN/TP2 - Actividad 3/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:

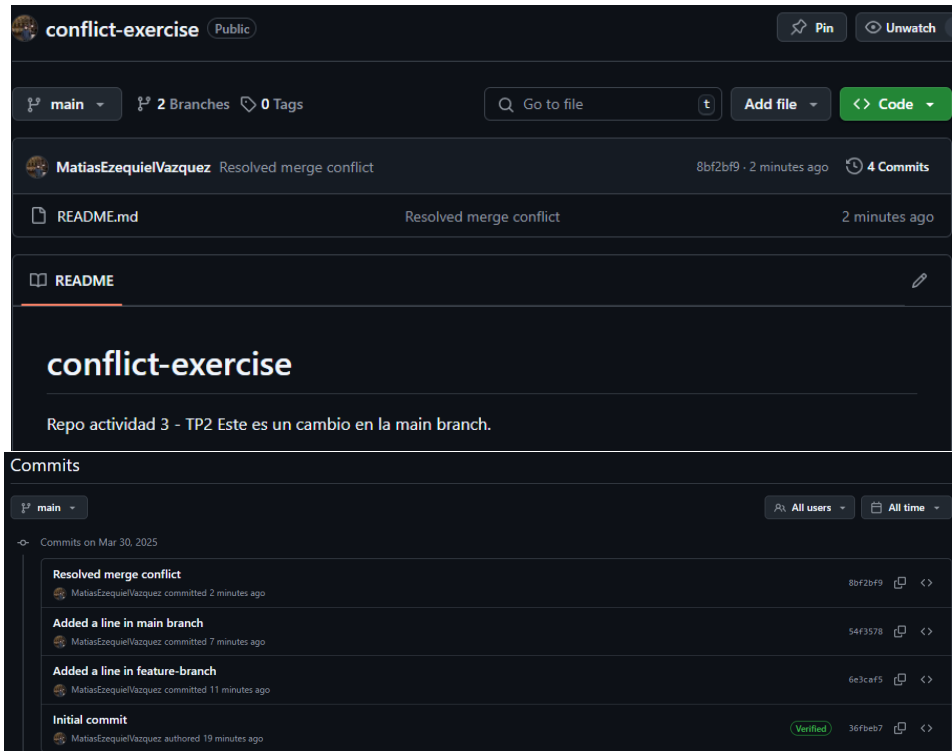
```
git push origin feature-branch
```

```
AzureAD+MatiasEzequielVazquez@LAPTOP-S7NMOM03 MINGW64 ~/source/UTN/TP2 - Actividad 3/conflict-exercise (main)
$ git push origin main
Enumerating objects: 11, done.
Counting objects: 100% (11/11), done.
Delta compression using up to 12 threads
Compressing objects: 100% (6/6), done.
Writing objects: 100% (9/9), 780 bytes | 65.00 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/MatiasEzequielVazquez/conflict-exercise.git
 36fbeb7..8bf2bf9 main -> main

AzureAD+MatiasEzequielVazquez@LAPTOP-S7NMOM03 MINGW64 ~/source/UTN/TP2 - Actividad 3/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/MatiasEzequielVazquez/conflict-exercise/pull/new/feature-branch
remote:
To https://github.com/MatiasEzequielVazquez/conflict-exercise.git
 * [new branch]   feature-branch -> 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 displays the GitHub interface for a repository named "conflict-exercise". At the top, the repository name is shown with a "Public" badge, along with "Pin" and "Unwatch" buttons. Below this, the main branch is selected, showing "2 Branches" and "0 Tags". A search bar "Go to file" and buttons for "Add file" and "Code" are present. The commit history shows a recent commit by "MatiasEzequielVazquez" titled "Resolved merge conflict" with hash "8bf2bf9" and "2 minutes ago", and "4 Commits" in total. Below the commit list, the "README.md" file is shown with the text "Resolved merge conflict" and "2 minutes ago". The README content includes the title "conflict-exercise" and the description "Repo actividad 3 - TP2 Este es un cambio en la main branch." Below the README, the "Commits" section is visible, showing a list of commits for the "main" branch, filtered by "All users" and "All time". The commits list includes:

- Resolved merge conflict** by MatiasEzequielVazquez committed 2 minutes ago (hash: 8bf2bf9)
- Added a line in main branch** by MatiasEzequielVazquez committed 7 minutes ago (hash: 54f3578)
- Added a line in feature-branch** by MatiasEzequielVazquez committed 11 minutes ago (hash: 6e3caf5)
- Initial commit** by MatiasEzequielVazquez authored 19 minutes ago (hash: 36fbeb7, marked as Verified)