

Pichulman Miguel Angel  
**PRACTICO 2**

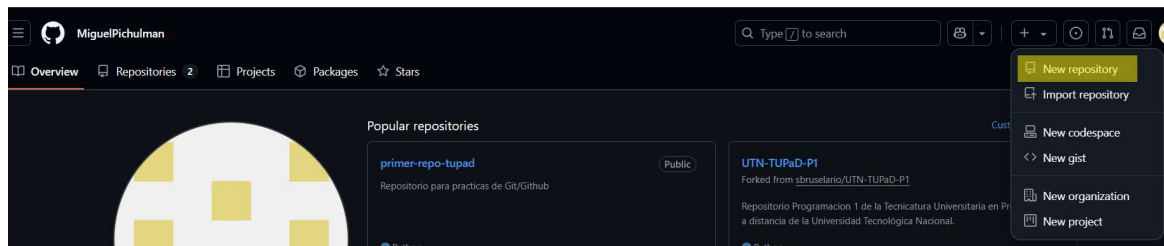
1)Contestar las siguientes preguntas utilizando las guías y documentación proporcionada (Desarrollar las respuestas):

- ¿Qué es GitHub?

GitHub es una plataforma de desarrollo de software que permite guardar de manera remota nuestros proyectos, sus cambios y cada una de sus versiones, lo que permite trabajar en conjunto con otros desarrolladores al utilizar el sistema de control de versiones GIT.

- ¿Cómo crear un repositorio en GitHub?

Primero hay q crear una cuenta en GitHub. Una vez dentro del perfil, hacer clic en “+”--->New repository



Luego Nombrarlo y crearlo

**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 \* MiguelPichulman / Repository name \*

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

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.](#)

ⓘ You are creating a public repository in your personal account.

**Create repository**

Ahora el repositorio esta listo para linkearlo con nuestro repositorio local.

*Pichulman Miguel Angel*

- ¿Cómo crear una rama en Git?

Utilizando el comando `git branch <nombre_rama>`

- ¿Cómo cambiar a una rama en Git?

Utilizando el comando `git checkout <nombre_rama>`

- ¿Cómo fusionar ramas en Git?

Primero debemos ubicarnos en la rama a la cual queremos fusionar los datos utilizando `git checkout master` (por ejemplo). Luego usamos el comando `git merge <nombre_rama>` para fusionar `<nombre_rama>` en la rama master

- ¿Cómo crear un commit en Git?

Primero se deben añadir el archivo que se quiera incluir en el commit utilizando el comando `git add <archivo>` o `git add` para incluir todos los q se encuentren en el repositorio. Luego se utiliza el comando `git commit -m "descripción"`

- ¿Cómo enviar un commit a GitHub?

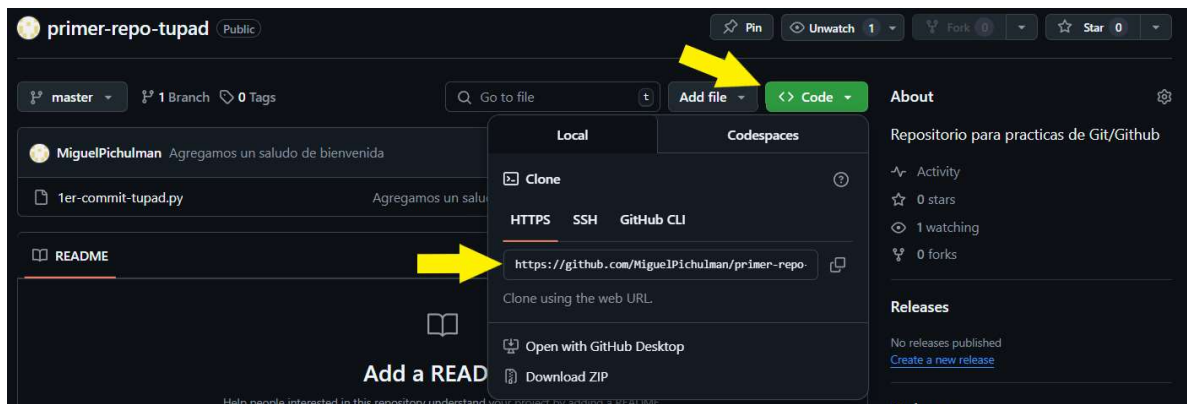
Utilizando el comando `git push origin <nombre_rama>`

- ¿Qué es un repositorio remoto?

Es una copia de un repositorio Git que se encuentra alojada en un servidor externo, accesible a través de internet.

- ¿Cómo agregar un repositorio remoto a Git?

Una vez tengamos creado nuestro repositorio en GitHub, debemos conocer su ruta. Para ello hacemos click en `<>Code` y luego copiamos la url asignada.



*Pichulman Miguel Angel*

Luego abrimos una terminal en el repositorio local que deseamos enviar a GitHub y tipeamos el comando `git remote add origin https://github.com/tu-usuario/tu-repositorio`.

Antes que nada debemos verificar si nuestro repositorio local tiene un repositorio remoto utilizando el comando `git remote -v`. En caso de que no salga nada listado deberemos utilizar el comando `git remote add origin https://github.com/tu-usuario/tu-repositorio.git`.

- ¿Cómo empujar cambios a un repositorio remoto?

Utilizando el comando `git push -u origin master`.

- ¿Cómo tirar de cambios de un repositorio remoto?

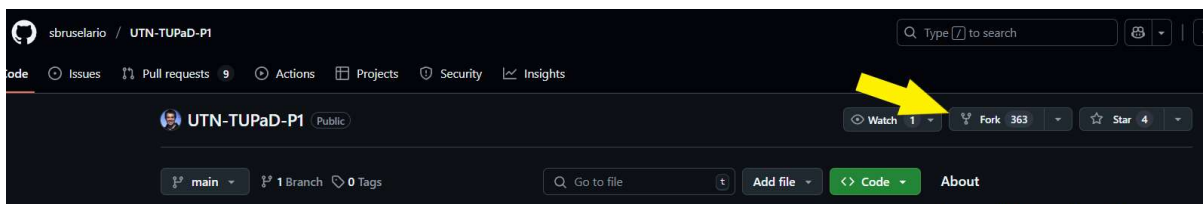
Utilizando el comando `git pull origin <nombre-de-la-rama>`.

- ¿Qué es un fork de repositorio?

Es realizar una copia de un repositorio en una cuenta diferente permitiendo hacer cambios sin afectar al original.

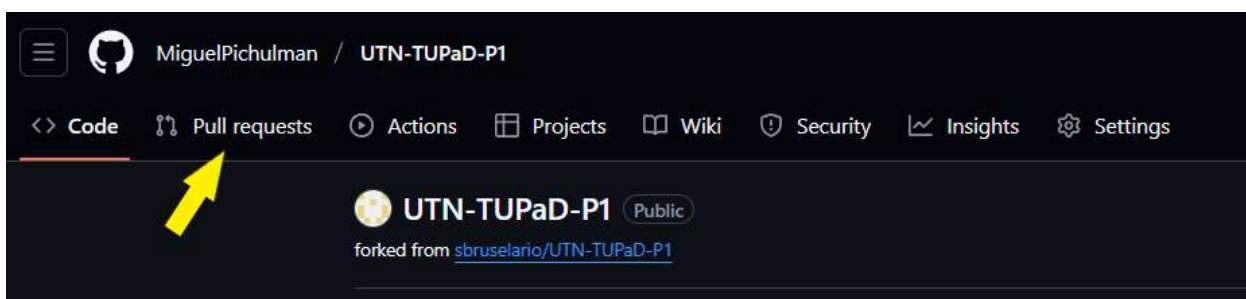
- ¿Cómo crear un fork de un repositorio?

Se debe acceder al repositorio original y luego apretar en el boton “Fork”

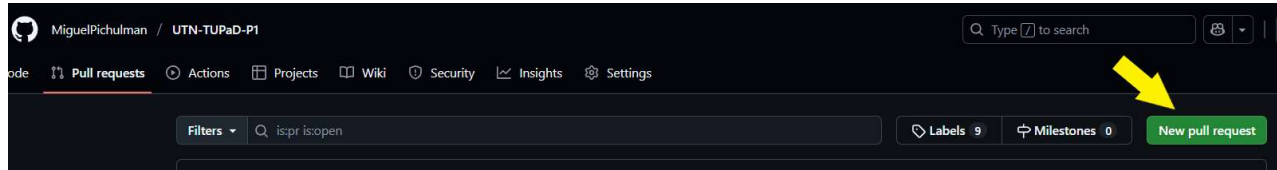


- ¿Cómo enviar una solicitud de extracción (pull request) a un repositorio?

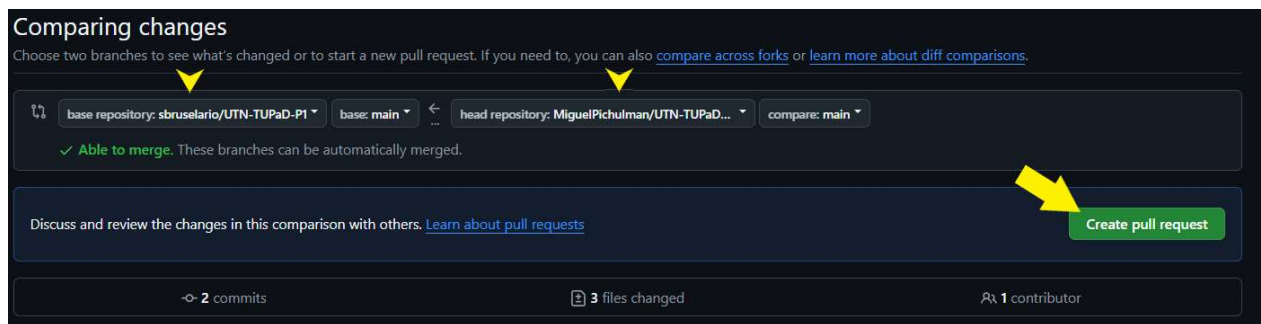
Dentro de nuestra cuenta de GitHub nos debemos ubicar en el repositorio que hemos “forkeado” y hacer click en “Pull Requests”



Luego click en “New pull request”



Luego de seleccionar/verificar que este seleccionado el repositorio original asi como también el repositorio q ha sufrido los cambios, click en “Create pull request”



- ¿Cómo aceptar una solicitud de extracción?

El autor del repositorio verá en sus pull requests el mensaje que le hemos enviado, para que lo pueda observar y si lo considera realizar el cambio pertinente (además de poder responderle al usuario que le ha propuesto ese cambio). Lo bueno de todo esto es que si el usuario original considera que esta modificación es buena y no genera conflictos con la rama maestra de su repositorio local remoto, puede clickear en “Merge pull request” y de esta manera sumará a su repositorio los cambios que hizo un usuario.



- ¿Qué es una etiqueta en Git?

Git tiene la posibilidad de etiquetar puntos específicos del historial como importantes. Esta funcionalidad se usa típicamente para marcar versiones de lanzamiento (v1.0, por ejemplo).



- ¿Cómo crear una etiqueta en Git?

Utilizando el comando *git tag <etiqueta>*

- ¿Cómo enviar una etiqueta a GitHub?

Utilizando el comando *git push origin <etiqueta>*

- ¿Qué es un historial de Git?

Es un registro de todos los cambios realizados en el repositorio.

- ¿Cómo ver el historial de Git?

Utilizando el comando *git log*.

- ¿Cómo buscar en el historial de Git?

Para buscar commits que contengan una palabra o frase específica en el mensaje de commit, usa git log con la opción *--grep*: *git log --grep="palabra clave"*

Para buscar commits que han modificado un archivo específico, usa git log seguido del nombre del archivo: *git log - nombre\_del\_archivo*

Para buscar commits en un rango de fechas específico, usa las opciones *--since* y *--until*:

*git log --since="2024-01-01" --until="2024-01-31"*

Para encontrar commits hechos por un autor específico, usa *--author*:

*git log --author="Nombre del Autor"*

- ¿Cómo borrar el historial de Git?

Utilizando el comando *rm -rf .git* (elimina todos los commits del repositorio)

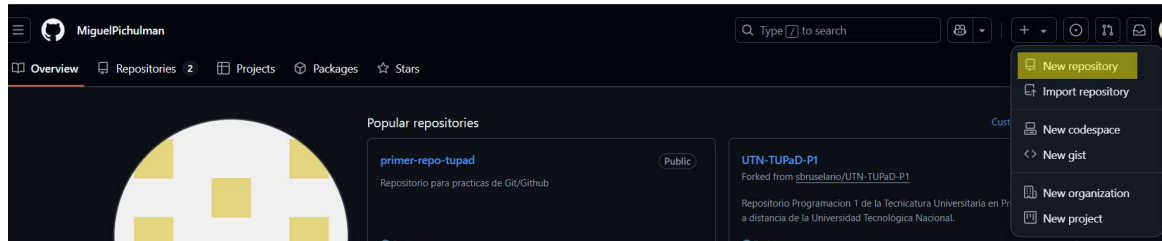
- ¿Qué es un repositorio privado en GitHub?

Son aquellos que solo son visibles para el propietario del repositorio y los colaboradores que especifique.

*Pichulman Miguel Angel*

- ¿Cómo crear un repositorio privado en GitHub?

Primero hay q crear una cuenta en GitHub. Una vez dentro del perfil, hacer clic en “+”--->New repository




Luego Nombrarlo, marcar la opción “Private” y crearlo

**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 \***

 MiguelPichulman /

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


**.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.](#)

 You are creating a private repository in your personal account.

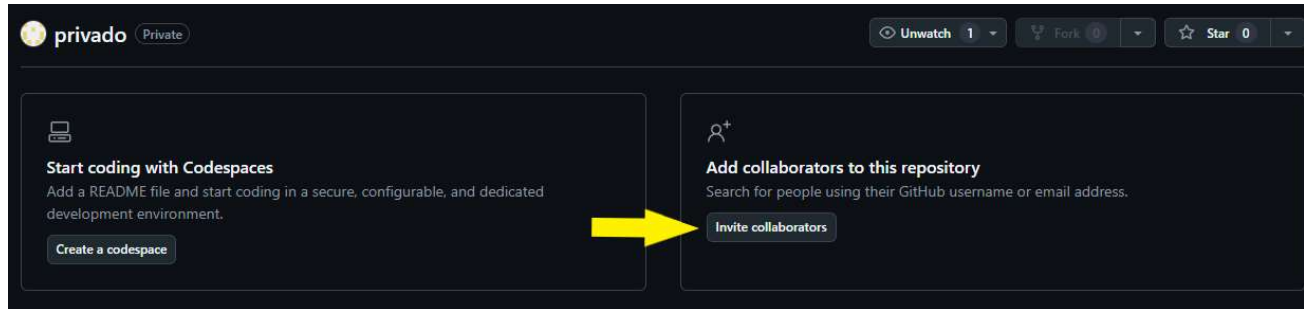
**Create repository**



*Pichulman Miguel Angel*

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

Entrar al repositorio y hacer click en “Invite collaborators”

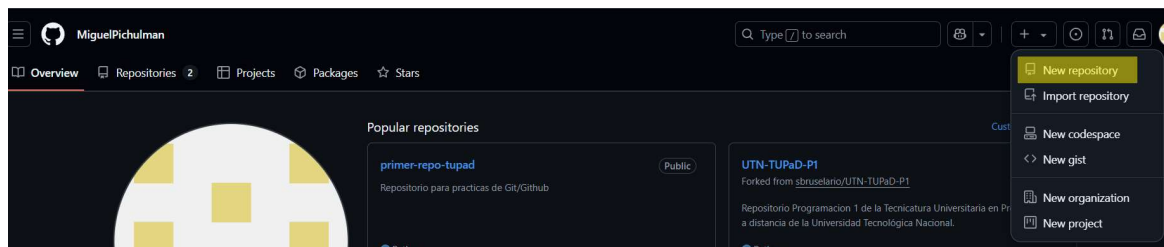


- Qué es un repositorio público en GitHub?

Es aquel que puede ser visto por cualquiera, incluso por alguien que no sea usuario de GitHub.

- ¿Cómo crear un repositorio público en GitHub?

Primero hay q crear una cuenta en GitHub. Una vez dentro del perfil, hacer clic en “+”---->New repository



Luego Nombrarlo, verificar que este marcada la opción “Public” y crearlo

**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 \* MiguelPichulman / Repository name \*

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

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.](#)

 You are creating a public repository in your personal account.



**Create repository**

- ¿Cómo compartir un repositorio público en GitHub?

Compartiendo la url del repositorio. Por ejemplo: <https://github.com/usuario/nombre-del-repositorio>





*Pichulman Miguel Angel*

2) Realizar la siguiente actividad.

- Crear un repositorio.

Dale un nombre al repositorio.

Elije el repositorio sea público.

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 \*

MiguelPichulman

Repository name \*

repo-TP2

repo-TP2 is available.

Great repository names are short and memorable. Need inspiration? How about [sturdy-rotary-phone](#) ?

Description (optional)

Repositorio creado para el ejercicio2 del TP2 de Programacion1

☒ 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](#).

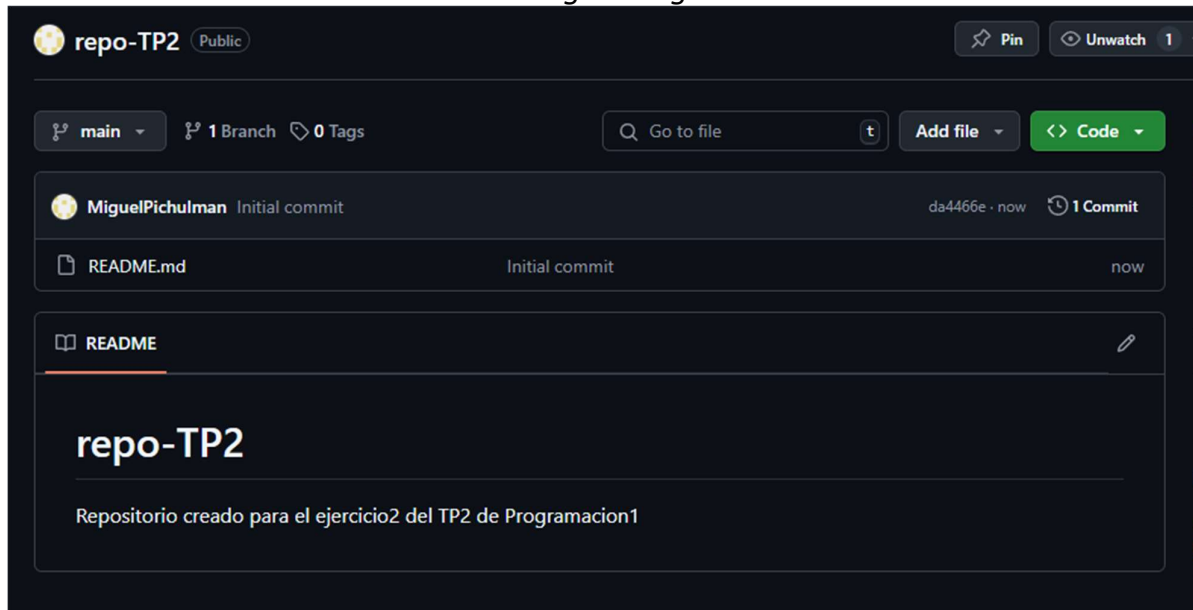
?

You are creating a public repository in your personal account.

Create repository



*Pichulman Miguel Angel*



```
MINGW64:/c/Users/pichu/Desktop/UTN/Programacion1/repo-TP2

pichu@DESKTOP-KQ77ULJ MINGW64 ~/Desktop/UTN/Programacion1/repo-TP2
$ git init
Initialized empty Git repository in C:/Users/pichu/Desktop/UTN/Programacion1/repo-TP2/.git/

pichu@DESKTOP-KQ77ULJ MINGW64 ~/Desktop/UTN/Programacion1/repo-TP2 (master)
$ |
```

- Agregando un Archivo

Crea un archivo simple, por ejemplo, "mi-archivo.txt".

```
MINGW64:/c/Users/pichu/Desktop/UTN/Programacion1/repo-TP2

pichu@DESKTOP-KQ77ULJ MINGW64 ~/Desktop/UTN/Programacion1/repo-TP2 (main)
$ echo "Archivo de practica TP2" > mi-archivo.txt

pichu@DESKTOP-KQ77ULJ MINGW64 ~/Desktop/UTN/Programacion1/repo-TP2 (main)
$ dir
README.md  mi-archivo.txt

pichu@DESKTOP-KQ77ULJ MINGW64 ~/Desktop/UTN/Programacion1/repo-TP2 (main)
$ |
```



*Pichulman Miguel Angel*

Realiza los comandos git add . y git commit -m "Agregando mi-archivo.txt" en la línea de comandos.

```
MINGW64:/c/Users/pichu/Desktop/UTN/Programacion1/repo-TP2

pichu@DESKTOP-KQ77ULJ MINGW64 ~/Desktop/UTN/Programacion1/repo-TP2 (main)
$ git add .
warning: in the working copy of 'mi-archivo.txt', LF will be replaced by CRLF the next time Git touches it

pichu@DESKTOP-KQ77ULJ MINGW64 ~/Desktop/UTN/Programacion1/repo-TP2 (main)
$ git commit -m "Agregando mi-archivo.txt"
[main 11ec6f8] Agregando mi-archivo.txt
1 file changed, 1 insertion(+)
create mode 100644 mi-archivo.txt

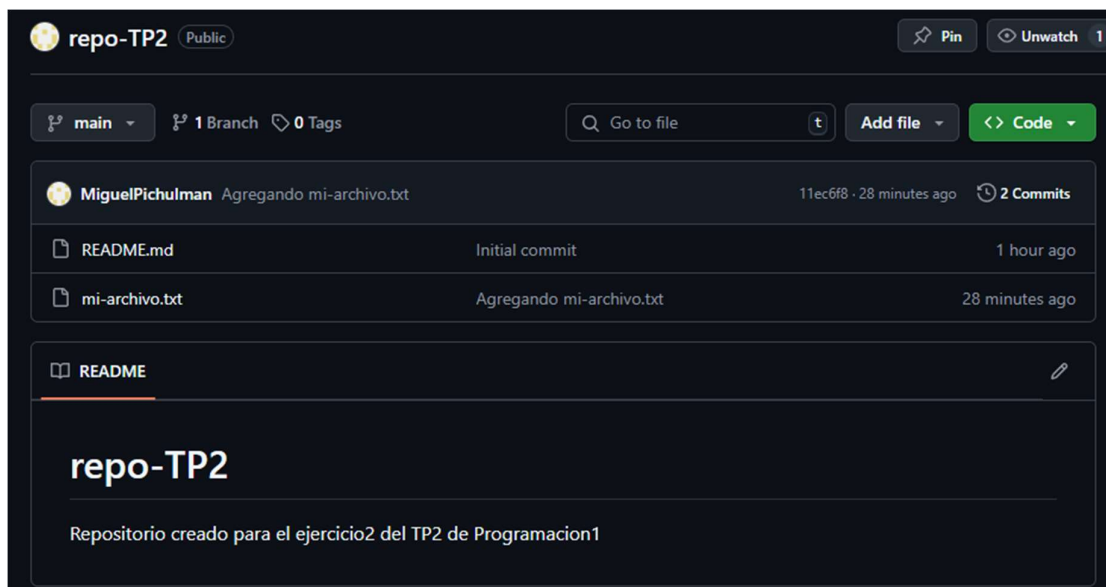
pichu@DESKTOP-KQ77ULJ MINGW64 ~/Desktop/UTN/Programacion1/repo-TP2 (main)
$ |
```

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

```
MINGW64:/c/Users/pichu/Desktop/UTN/Programacion1/repo-TP2

pichu@DESKTOP-KQ77ULJ MINGW64 ~/Desktop/UTN/Programacion1/repo-TP2 (main)
$ git push
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 12 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 325 bytes | 325.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/MiguelPichulman/repo-TP2.git
a0eac54..11ec6f8 main -> main

pichu@DESKTOP-KQ77ULJ MINGW64 ~/Desktop/UTN/Programacion1/repo-TP2 (main)
$ .....
```





*Pichulman Miguel Angel*

- Creando Branchs

Crear una Branch

```
MINGW64:/c/Users/pichu/Desktop/UTN/Programacion1/repo-TP2

pichu@DESKTOP-KQ77ULJ MINGW64 ~/Desktop/UTN/Programacion1/repo-TP2 (main)
$ git branch nuevaRama

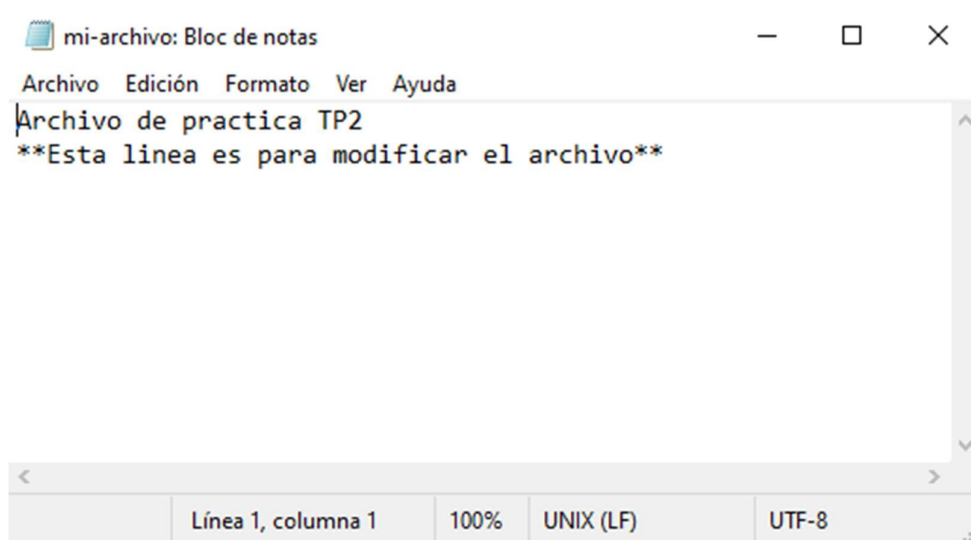
pichu@DESKTOP-KQ77ULJ MINGW64 ~/Desktop/UTN/Programacion1/repo-TP2 (main)
$ git branch
* main
  nuevaRama

pichu@DESKTOP-KQ77ULJ MINGW64 ~/Desktop/UTN/Programacion1/repo-TP2 (main)
$ git switch nuevaRama
Switched to branch 'nuevaRama'

pichu@DESKTOP-KQ77ULJ MINGW64 ~/Desktop/UTN/Programacion1/repo-TP2 (nuevaRama)
$ git branch
  main
* nuevaRama

pichu@DESKTOP-KQ77ULJ MINGW64 ~/Desktop/UTN/Programacion1/repo-TP2 (nuevaRama)
$ |
```

Realizar cambios o agregar un archivo





*Pichulman Miguel Angel*

Subir la Branch

```
MINGW64:/c/Users/pichu/Desktop/UTN/Programacion1/repo-TP2

pichu@DESKTOP-KQ77ULJ MINGW64 ~/Desktop/UTN/Programacion1/repo-TP2 (nuevaRama)
$ git add .
warning: in the working copy of 'mi-archivo.txt', LF will be replaced by CRLF the next time Git touches it

pichu@DESKTOP-KQ77ULJ MINGW64 ~/Desktop/UTN/Programacion1/repo-TP2 (nuevaRama)
$ git status
On branch nuevaRama
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   mi-archivo.txt

pichu@DESKTOP-KQ77ULJ MINGW64 ~/Desktop/UTN/Programacion1/repo-TP2 (nuevaRama)
$ git commit -m "Agregando miduifacion a mi-archivo.txt desde nuevaRama"
[nuevaRama 4b32e94] Agregando miduifacion a mi-archivo.txt desde nuevaRama
1 file changed, 1 insertion(+)

pichu@DESKTOP-KQ77ULJ MINGW64 ~/Desktop/UTN/Programacion1/repo-TP2 (nuevaRama)
$ git status
On branch nuevaRama
nothing to commit, working tree clean
```

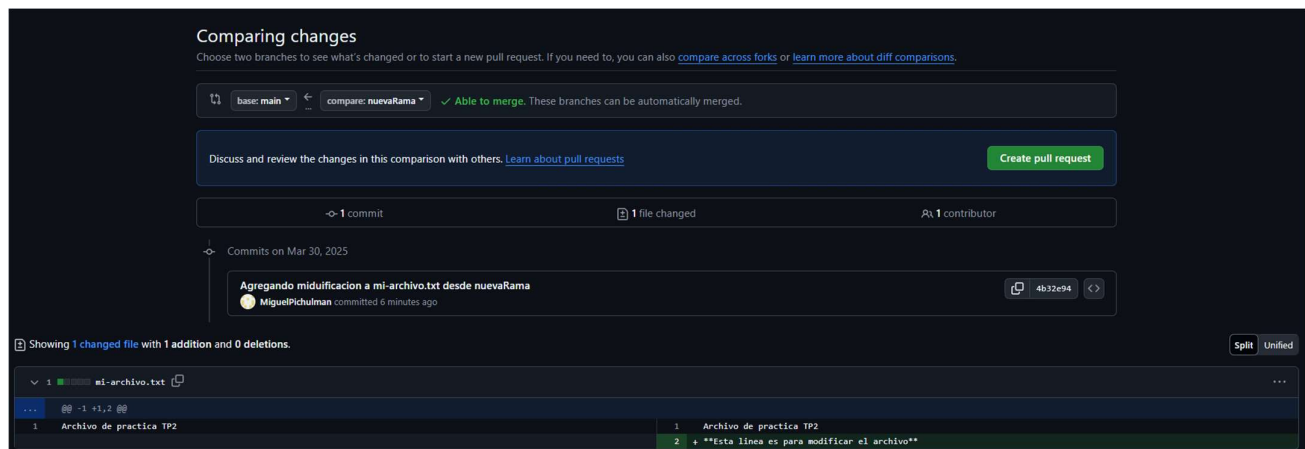
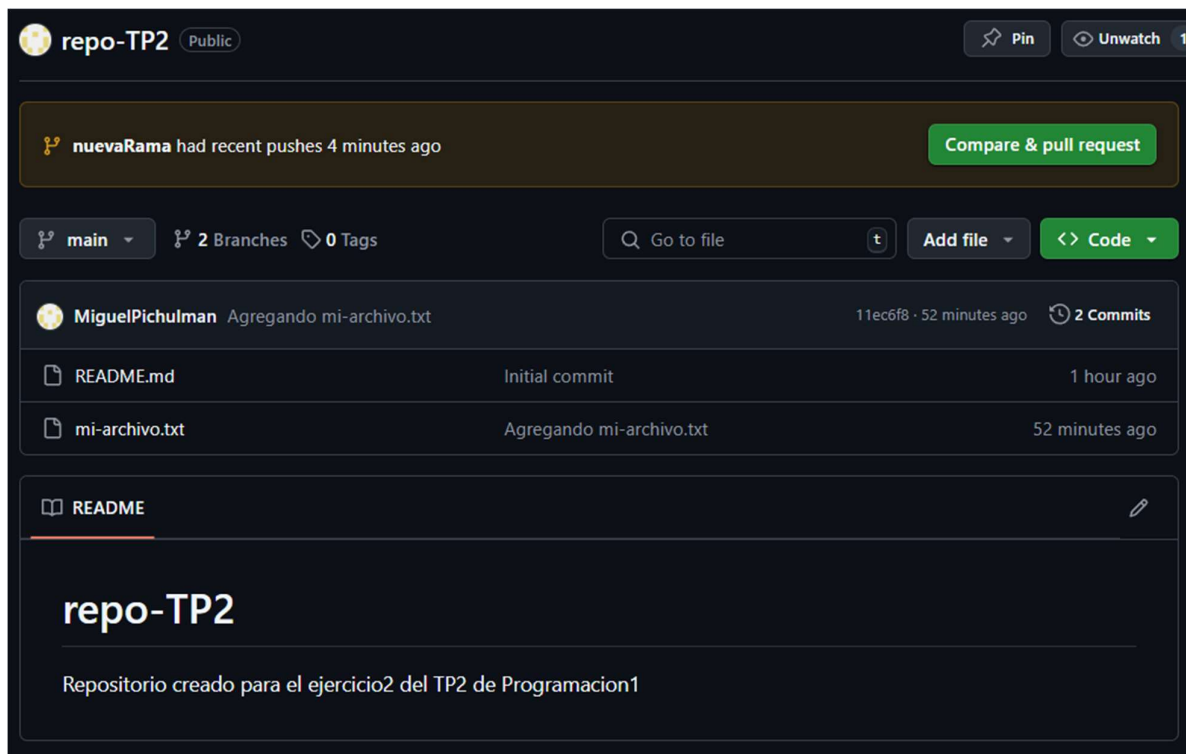
```
MINGW64:/c/Users/pichu/Desktop/UTN/Programacion1/repo-TP2

pichu@DESKTOP-KQ77ULJ MINGW64 ~/Desktop/UTN/Programacion1/repo-TP2 (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% (3/3), done.
Writing objects: 100% (3/3), 385 bytes | 385.00 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/MiguelPichulman/repo-TP2/pull/new/nuevaRama
remote:
To https://github.com/MiguelPichulman/repo-TP2.git
 * [new branch]      nuevaRama -> nuevaRama

pichu@DESKTOP-KQ77ULJ MINGW64 ~/Desktop/UTN/Programacion1/repo-TP2 (nuevaRama)
$ |
```



*Pichulman Miguel Angel*



### 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".





*Pichulman Miguel Angel*

cd• 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 \*



MiguelPichulman

Repository name \*

conflict-exercise

✔ conflict-exercise is available.

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

Description (optional)

Repositorio del ejercicio3 del TP2 de Programacion1



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



*Pichulman Miguel Angel*

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
```

```
MINGW64/c/Users/pichu/Desktop/UTN/Programacion1/conflict-exercise

pichu@DESKTOP-KQ77ULJ MINGW64 ~/Desktop/UTN/Programacion1
$ git clone https://github.com/MiguelPichulman/conflict-exercise.git
Cloning into 'conflict-exercise'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (3/3), done.

pichu@DESKTOP-KQ77ULJ MINGW64 ~/Desktop/UTN/Programacion1
$ cd conflict-exercise/

pichu@DESKTOP-KQ77ULJ MINGW64 ~/Desktop/UTN/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
```

```
MINGW64/c/Users/pichu/Desktop/UTN/Programacion1/conflict-exercise

pichu@DESKTOP-KQ77ULJ MINGW64 ~/Desktop/UTN/Programacion1/conflict-exercise (main)
$ git checkout -b feature-branch
Switched to a new branch 'feature-branch'

pichu@DESKTOP-KQ77ULJ MINGW64 ~/Desktop/UTN/Programacion1/conflict-exercise (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.



*Pichulman Miguel Angel*

```
File Edit Selection View Go ... Search
Restricted Mode is intended for safe code browsing. Trust this window to enable all features. Manage Learn More
README.md X
C: > Users > pichu > Desktop > UTN > Programacion1 > conflict-exercise > README.md > # conflict-exercise
1 # conflict-exercise
2 Repositorio del ejercicio3 del TP2 de Programacion1
3 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/pichu/Desktop/UTN/Programacion1/conflict-exercise
pichu@DESKTOP-KQ77ULJ MINGW64 ~/Desktop/UTN/Programacion1/conflict-exercise (feature-branch)
$ git add README.md

pichu@DESKTOP-KQ77ULJ MINGW64 ~/Desktop/UTN/Programacion1/conflict-exercise (feature-branch)
$ git commit -m "added a line in featured-branch"
[feature-branch 5e84d8e] added a line in featured-branch
1 file changed, 1 insertion(+)

pichu@DESKTOP-KQ77ULJ MINGW64 ~/Desktop/UTN/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
```

```
MINGW64:/c/Users/pichu/Desktop/UTN/Programacion1/conflict-exercise
pichu@DESKTOP-KQ77ULJ MINGW64 ~/Desktop/UTN/Programacion1/conflict-exercise (feature-branch)
$ git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.

pichu@DESKTOP-KQ77ULJ MINGW64 ~/Desktop/UTN/Programacion1/conflict-exercise (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"
```



*Pichulman Miguel Angel*

```
MINGW64:/c/Users/pichu/Desktop/UTN/Programacion1/conflict-exercise

pichu@DESKTOP-KQ77ULJ MINGW64 ~/Desktop/UTN/Programacion1/conflict-exercise (main)
$ git add README.md

pichu@DESKTOP-KQ77ULJ MINGW64 ~/Desktop/UTN/Programacion1/conflict-exercise (main)
$ git commit -m "added a line in main-branch"
[main f22eb1e] added a line in main-branch
1 file changed, 1 insertion(+), 1 deletion(-)

pichu@DESKTOP-KQ77ULJ MINGW64 ~/Desktop/UTN/Programacion1/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/pichu/Desktop/UTN/Programacion1/conflict-exercise

pichu@DESKTOP-KQ77ULJ MINGW64 ~/Desktop/UTN/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.

pichu@DESKTOP-KQ77ULJ MINGW64 ~/Desktop/UTN/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



*Pichulman Miguel Angel*

```
File Edit Selection View Go ... Search
Restricted Mode is intended for safe code browsing. Trust this window to enable all features. Manage Learn More
README.md X
C: > Users > pichu > Desktop > UTN > Programacion1 > conflict-exercise > README.md > # conflict-exercise
1 # conflict-exercise
  Accept Current Change | Accept Incoming Change | Accept Both Changes | Compare Changes
2 <<<<<< HEAD (Current Change)
3 Este es un cambio en la main branch
4 =====
5 Repositorio del ejercicio3 del TP2 de Programacion1
6 Este es un cambio en la feature branch.
7 >>>>>> feature-branch (Incoming Change)
8
```

- 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/pichu/Desktop/UTN/Programacion1/conflict-exercise
pichu@DESKTOP-KQ77ULJ MINGW64 ~/Desktop/UTN/Programacion1/conflict-exercise (main|MERGING)
$ git add README.md

pichu@DESKTOP-KQ77ULJ MINGW64 ~/Desktop/UTN/Programacion1/conflict-exercise (main|MERGING)
$ git commit -m "Resolved merge conflict"
[main 9f61847] Resolved merge conflict

pichu@DESKTOP-KQ77ULJ MINGW64 ~/Desktop/UTN/Programacion1/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
```





*Pichulman Miguel Angel*

```
MINGW64:/c/Users/pichu/Desktop/UTN/Programacion1/conflict-exercise

pichu@DESKTOP-KQ77ULJ MINGW64 ~/Desktop/UTN/Programacion1/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), 850 bytes | 850.00 KiB/s, done.
Total 9 (delta 2), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (2/2), done.
To https://github.com/MiguelPichulman/conflict-exercise.git
   de4d251..9f61847  main -> main

pichu@DESKTOP-KQ77ULJ MINGW64 ~/Desktop/UTN/Programacion1/conflict-exercise (main)
$ |
```

- También sube la feature-branch si deseas:

git push origin feature-branch

```
MINGW64:/c/Users/pichu/Desktop/UTN/Programacion1/conflict-exercise

pichu@DESKTOP-KQ77ULJ MINGW64 ~/Desktop/UTN/Programacion1/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/MiguelPichulman/conflict-exercise/pull/new/feature-branch
remote:
To https://github.com/MiguelPichulman/conflict-exercise.git
 * [new branch]      feature-branch -> feature-branch

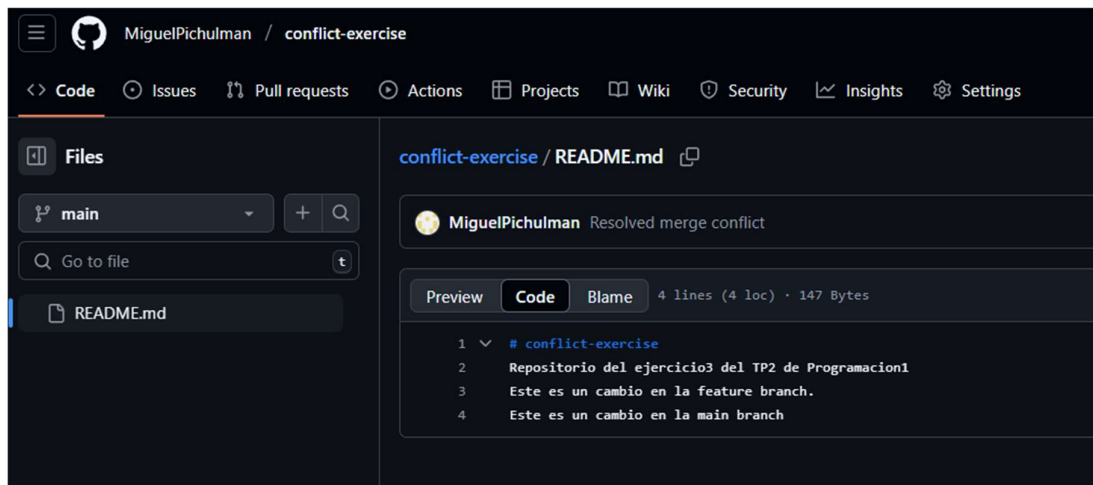
pichu@DESKTOP-KQ77ULJ MINGW64 ~/Desktop/UTN/Programacion1/conflict-exercise (main)
$ |
```

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.



*Pichulman Miguel Angel*



```
MINGW64:/c:/Users/pichu/Desktop/UTN/Programacion1/conflict-exercise

pichu@DESKTOP-KQ77ULJ MINGW64 ~/Desktop/UTN/Programacion1/conflict-exercise (main)
$ git log
commit 9f6184705b2a82f6fefa6b02ce94db1f86a2c5d7 (HEAD -> main, origin/main, origin/HEAD)
Merge: f22eb1e 5e84d8e
Author: Miguel Pichulman <miguel.a.pichulman@gmail.com>
Date: Sun Mar 30 21:00:16 2025 -0300

    Resolved merge conflict

commit f22eb1e35f8ea7ac7154745e8f1861571225f18a
Author: Miguel Pichulman <miguel.a.pichulman@gmail.com>
Date: Sun Mar 30 20:43:37 2025 -0300

    added a line in main-branch

commit 5e84d8e5953e0cf4132939f21c68d542517ccc01 (origin/feature-branch, feature-branch)
Author: Miguel Pichulman <miguel.a.pichulman@gmail.com>
Date: Sun Mar 30 20:36:18 2025 -0300

    added a line in featured-branch

commit de4d251b3083fccd67ef09fa5abf12d50a582dff
Author: MiguelPichulman <miguel.a.pichulman@gmail.com>
Date: Sun Mar 30 20:18:34 2025 -0300

    Initial commit

pichu@DESKTOP-KQ77ULJ MINGW64 ~/Desktop/UTN/Programacion1/conflict-exercise (main)
$ |
```