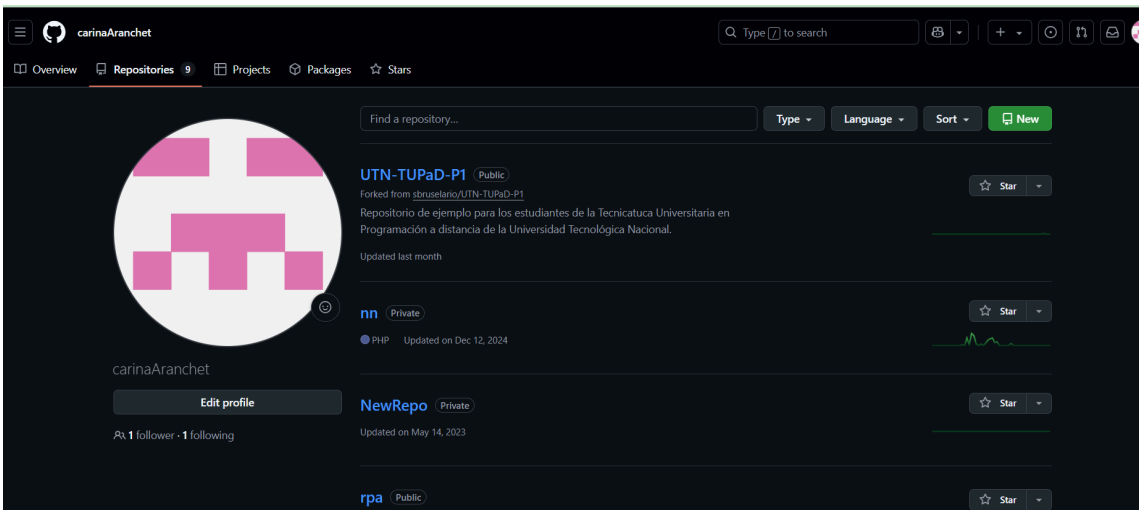
Actividades

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

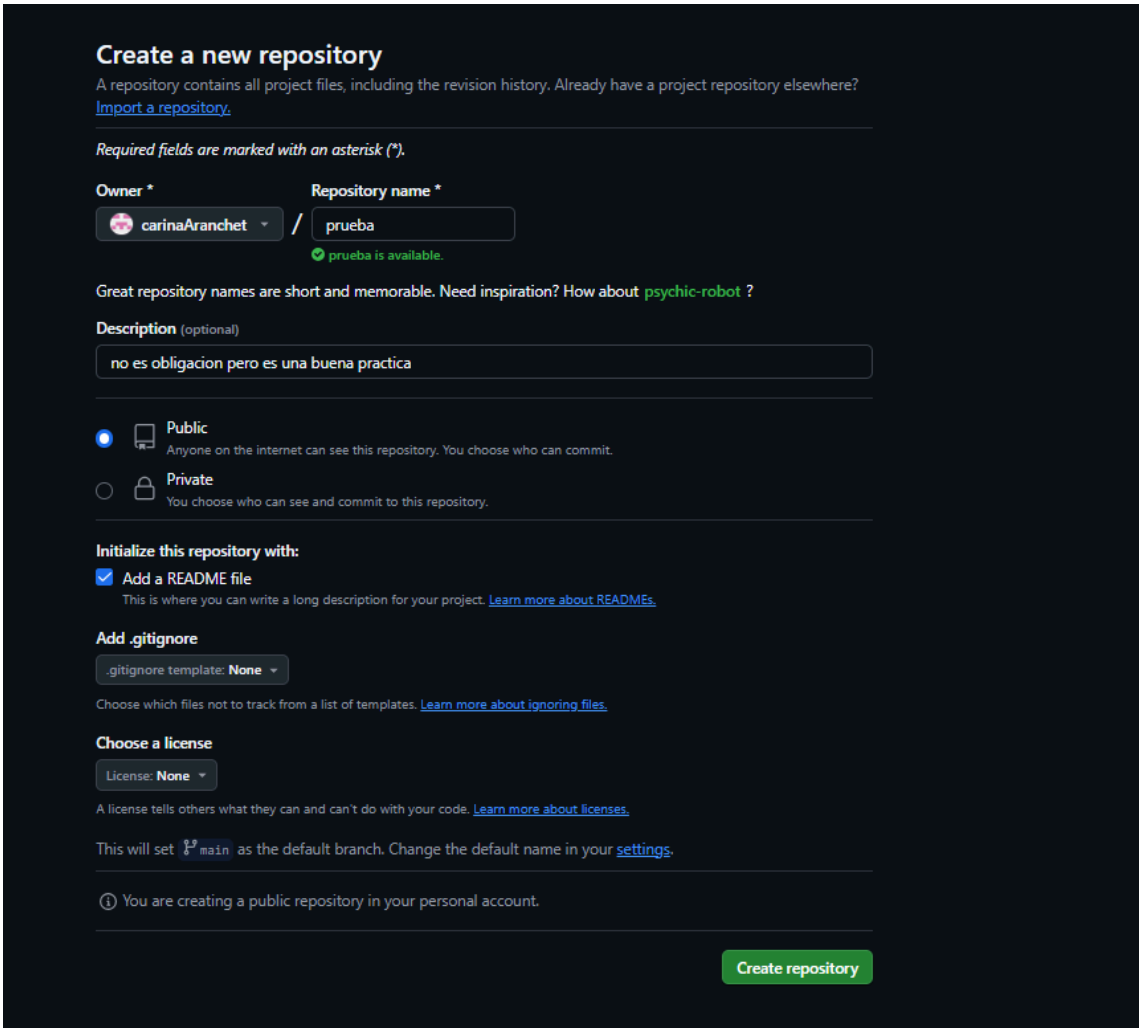
• ¿Qué es GitHub?

Es una plataforma online de desarrollo colaborativo, en el cual se puede administrar y versionar proyectos mediante el uso del sistema de control de versiones **Git**.

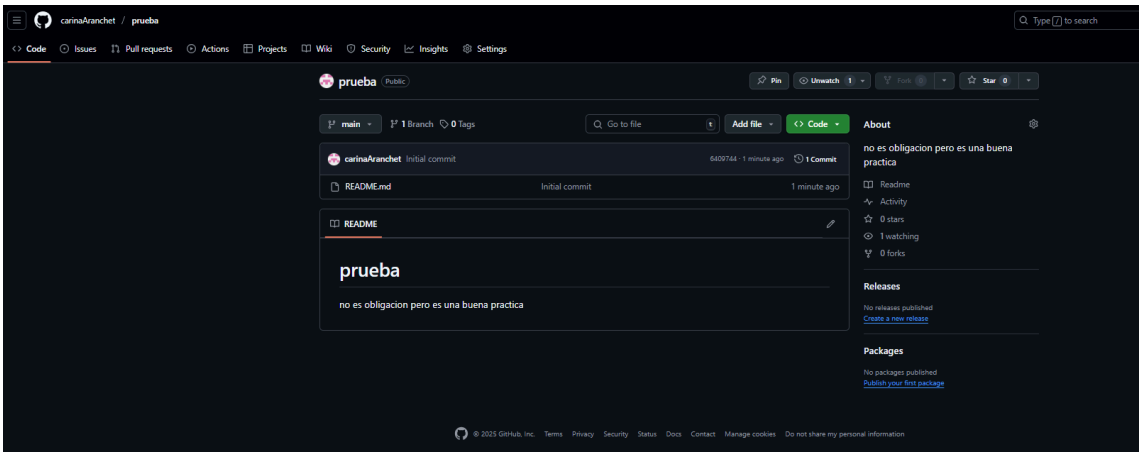
• ¿Cómo crear un repositorio en GitHub?



Presionamos la opción “New”



Ingresamos el nombre del repositorio, la descripción en caso de que se quiera agregar, y por ultimo se puede inicializar el repositorio con el archivo README.md. Y clickeamos en la opción “Create repository”



Finalmente vemos creado el repositorio.

• ¿Cómo crear una rama en Git?

Para crear una rama en git, utilizamos el comando git branch nombreDeLaRama

• ¿Cómo cambiar a una rama en Git?

Para movernos a la rama que acabamos de crear, utilizamos el siguiente comando:

```
git checkout nombreDeLaRama
```

• ¿Cómo fusionar ramas en Git?

Se utiliza el siguiente comando:

```
git merge nombreDeLaRama
```

• ¿Cómo crear un commit en Git?

```
git commit -m “algún comentario”
```

• ¿Cómo enviar un commit a GitHub?

```
git push origin nombreDeLaRama
```

• ¿Qué es un repositorio remoto?

Es una versión del repositorio que se encuentra alojado en una plataforma online.

• ¿Cómo agregar un repositorio remoto a Git?

Para agregar un repositorio remoto a Git, se debe utilizar el comando :

```
git remote add origin URL-del-repositorio
```

previamente se tiene que haber creado el repositorio remoto.

• ¿Cómo empujar cambios a un repositorio remoto?

Se utiliza el comando:

```
git push origin nombreDeLaRama
```

• ¿Cómo tirar de cambios de un repositorio remoto?

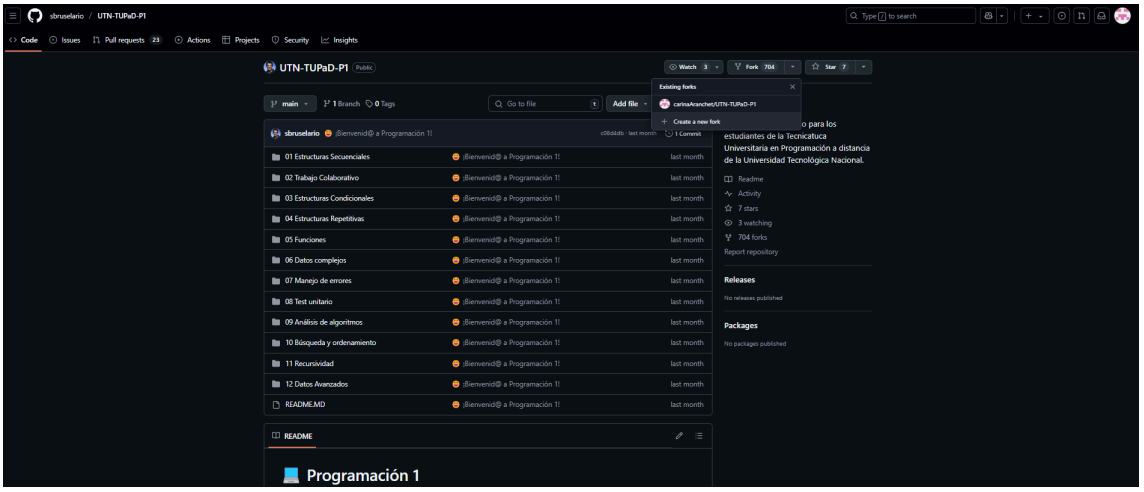
```
git pull origin nombreDeLaRama
```

• ¿Qué es un fork de repositorio?

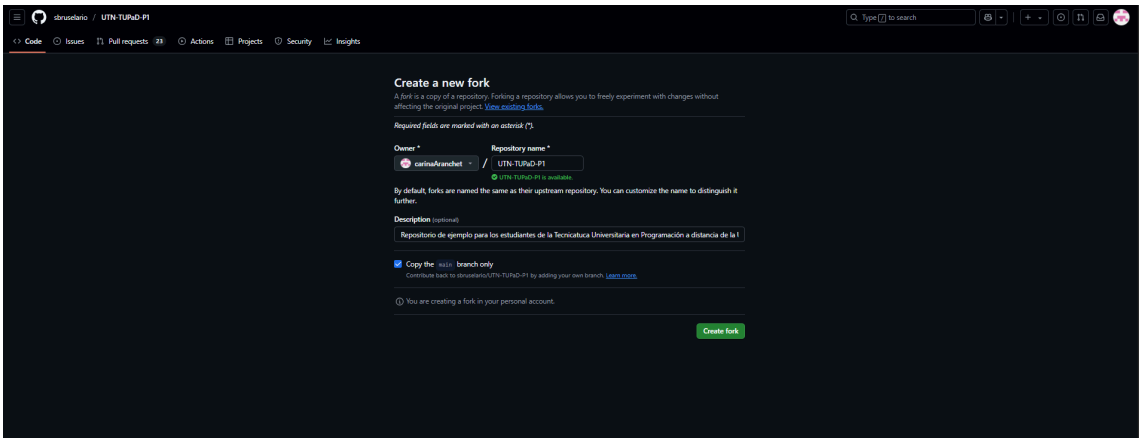
Un fork de un repositorio es una copia de un repositorio publico de otro usuario en nuestra cuenta.

• ¿Cómo crear un fork de un repositorio?

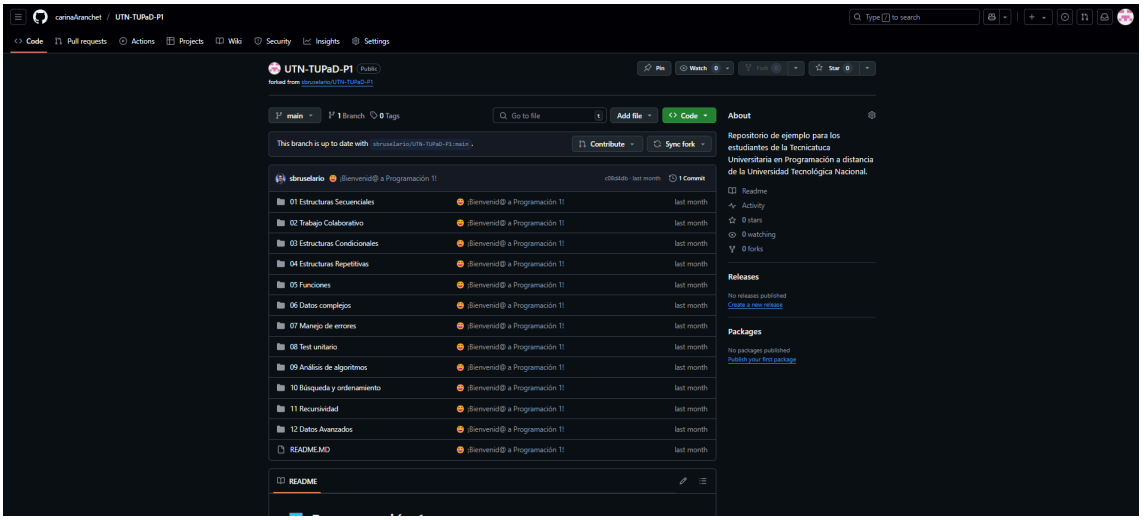
Se accede al repositorio en donde se quiere hacer un fork:



Luego hay que clickear la opción de fork >> new fork



Clickeamos la opción “créate fork”



Listo, copia realizada!!

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

Para enviar una **pull request** (PR) a un repositorio en GitHub, sigue estos pasos:

1. **Haz un fork** del repositorio original si no tienes acceso directo.
2. **Clona** el repositorio forked a tu máquina local.
3. Crea una **nueva rama** para trabajar en tus cambios.
4. **Realiza tus cambios** en esa rama.
5. **Commit y push** tus cambios al repositorio forked.
6. En GitHub, ve a tu repositorio forked y haz clic en el botón **"New pull request"**.
7. **Selecciona la rama** de tu repositorio forked que contiene los cambios y compara con la rama del repositorio original.
8. **Envía la pull request** describiendo los cambios que has hecho.

• ¿Cómo aceptar una solicitud de extracción?

Para aceptar una **pull request** en GitHub:

1. Revisa el contenido de la pull request.
2. Si todo está correcto y sin conflictos, haz clic en **"Merge pull request"**.
3. Después de hacer el merge, puedes eliminar la rama de la pull request si ya no es necesaria.

• ¿Qué es un etiqueta en Git?

Una **etiqueta** (tag) en Git es una referencia que señala un punto específico en el historial de commits, generalmente utilizado para marcar versiones o puntos importantes del proyecto (por ejemplo, una versión de lanzamiento).

• ¿Cómo crear una etiqueta en Git?

```
git tag nombre_de_etiqueta
```

• ¿Cómo enviar una etiqueta a GitHub?

```
git push origin nombre_de_etiqueta
```

• ¿Qué es un historial de Git?

Es el registro completo de todos los commits realizados en el repositorio

• ¿Cómo ver el historial de Git?

```
git log
```

• ¿Cómo buscar en el historial de Git?

Por mensaje de commit :

```
git log --grep="texto a buscar"
```

Por cambios en archivos:

```
git log -S "texto"
```

• ¿Cómo borrar el historial de Git?

Borrar el historial de Git es una operación avanzada y generalmente no se recomienda. Se puede reescribir el historial utilizando comandos como git rebase o git filter-branch

• ¿Qué es un repositorio privado en GitHub?

Un repositorio privado es aquel que solo pueden ver y acceder las personas a las que se les otorga permiso. Es ideal para proyectos que no deseas compartir públicamente.

• ¿Cómo crear un repositorio privado en GitHub?

Al crear un repositorio en GitHub, selecciona la opción “Private” (Privado) en lugar de “Public” en la sección de visibilidad. Esto restringe el acceso a usuarios autorizados.

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

Dentro del repositorio, en “settings”, “Manage access” se invita al usuario ingresando su nombre o correo electronico y asignandole el rol adecuado

• ¿Qué es un repositorio público en GitHub?

Un repositorio público es accesible para cualquier persona en Internet.

• ¿Cómo crear un repositorio público en GitHub?

Cuando creas el repositorio seleccionas la opción “public” en la seccion de visibilidad.

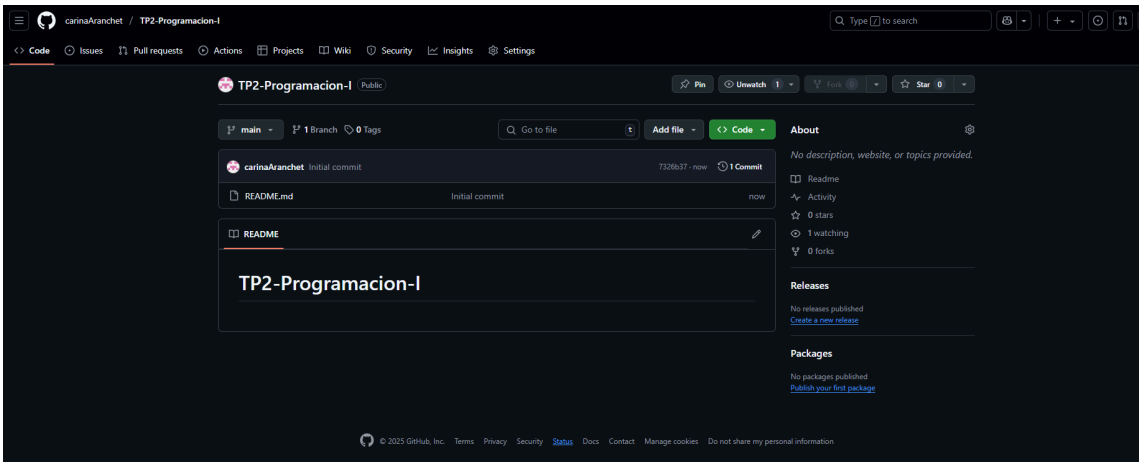
• ¿Cómo compartir un repositorio público en GitHub?

Compartir un repositorio público es tan sencillo como proporcionar la URL del repositorio.

2) Realizar la siguiente actividad:

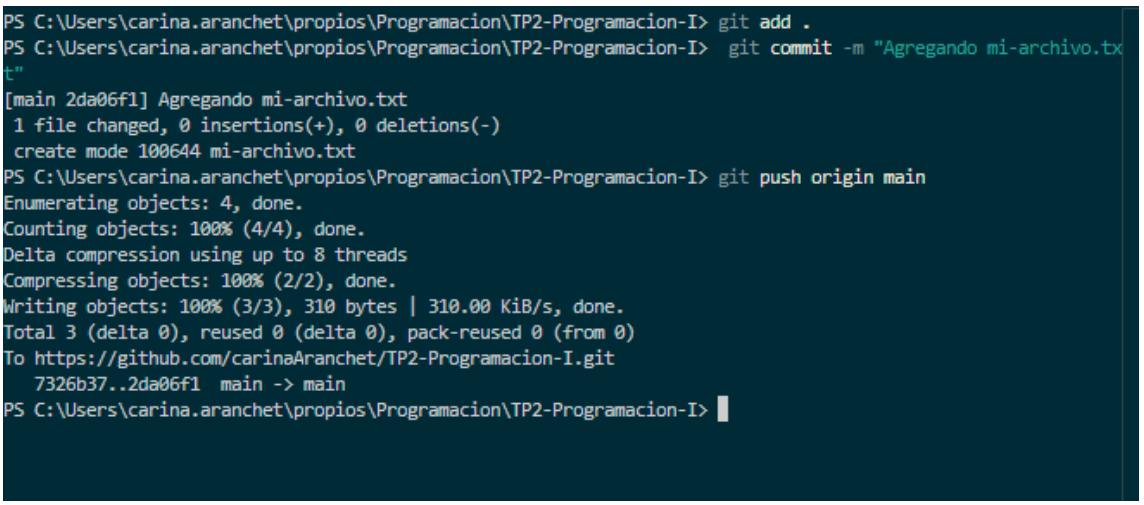
• Crear un repositorio:

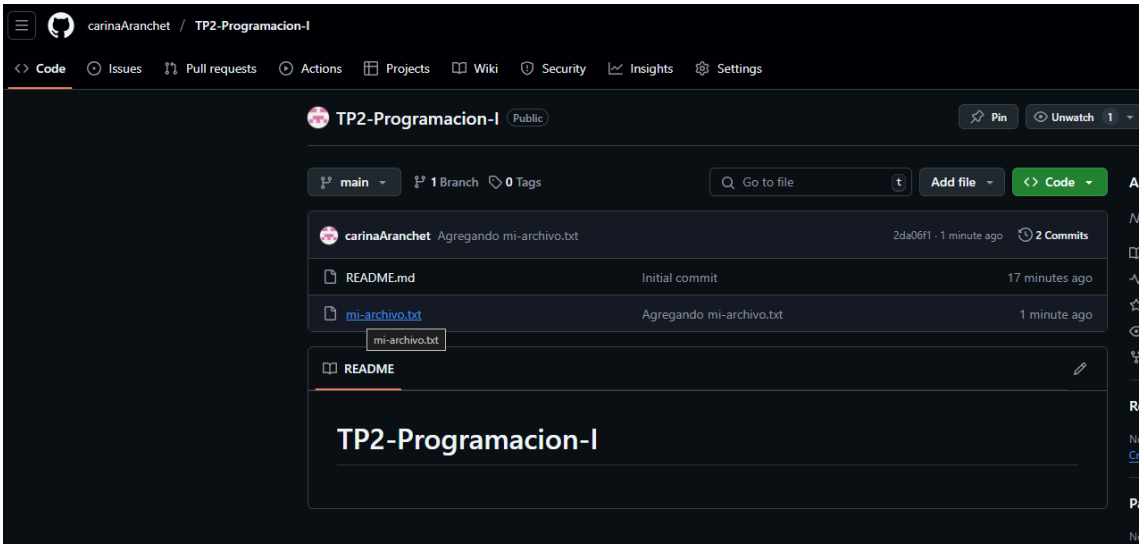
1) Dale un nombre al repositorio.
2) Elije el repositorio sea público.
3) Inicializa el repositorio con un archivo.



• Agregando un Archivo:

1) Crea un archivo simple, por ejemplo, "mi-archivo.txt".
2) Realiza los comandos git add . y git commit -m "Agregando mi-archivo.txt" en la línea de comandos.
3) Sube los cambios al repositorio en GitHub con git push origin main (o el nombre de la rama correspondiente).

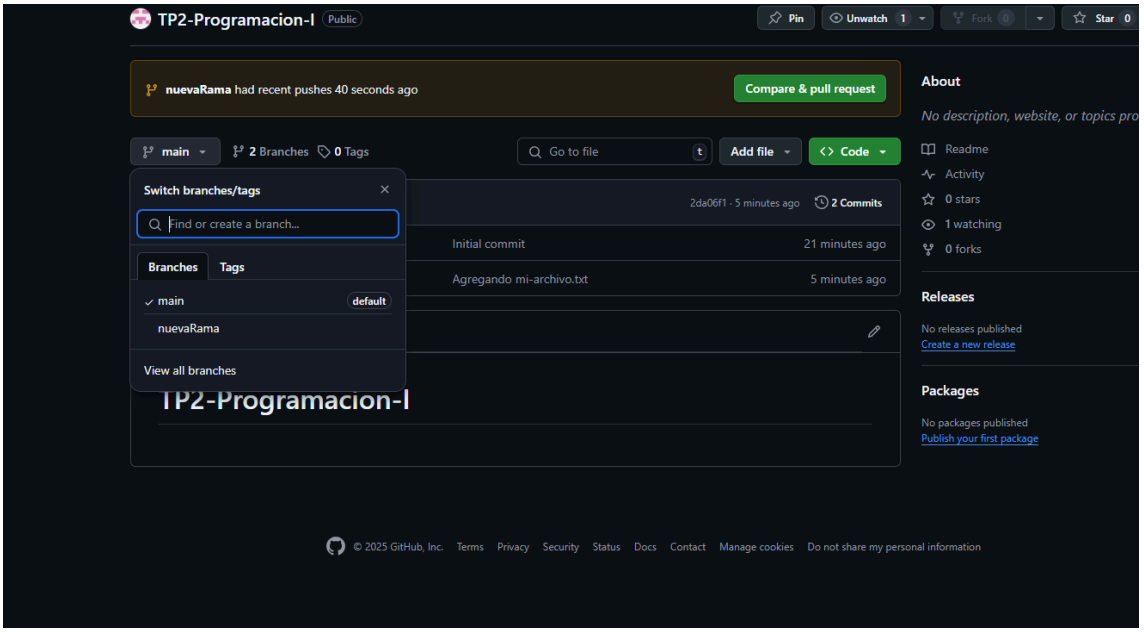




- Creando Branchs:

1) Crear una Branch
2) Realizar cambios o agregar un archivo
3) Subir la Branch

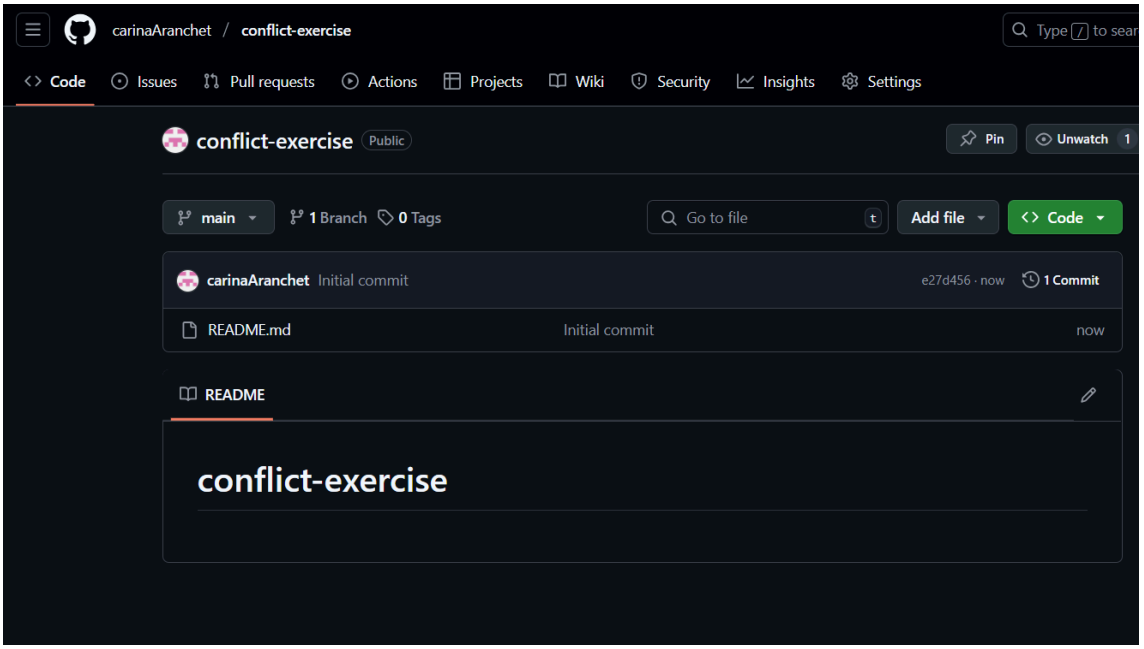
```
PS C:\Users\carina.aranchet\propios\Programacion\TP2-Programacion-I> git checkout -b nuevaRama
Switched to a new branch 'nuevaRama'
PS C:\Users\carina.aranchet\propios\Programacion\TP2-Programacion-I> echo "Archivo nuevo" > archivo.txt
PS C:\Users\carina.aranchet\propios\Programacion\TP2-Programacion-I> git add .
PS C:\Users\carina.aranchet\propios\Programacion\TP2-Programacion-I> git commit -m "Agregué archivo.txt con
contenido nuevo"
[nuevaRama c9834f0] Agregué archivo.txt con contenido nuevo
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 archivo.txt
PS C:\Users\carina.aranchet\propios\Programacion\TP2-Programacion-I> git push origin nuevaRama
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 8 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 379 bytes | 379.00 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/carinaAranchet/TP2-Programacion-I/pull/new/nuevaRama
remote:
To https://github.com/carinaAranchet/TP2-Programacion-I.git
 * [new branch]      nuevaRama -> nuevaRama
PS C:\Users\carina.aranchet\propios\Programacion\TP2-Programacion-I>
```



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".



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

```
PS C:\Users\carina.aranchet\propios\Programacion> git clone https://github.com/carinaAranchet/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.
PS C:\Users\carina.aranchet\propios\Programacion> cd conflict-exercise
PS C:\Users\carina.aranchet\propios\Programacion\conflict-exercise>
```

Paso 3: Crear una nueva rama y editar un archivo

- Crea una nueva rama llamada feature-branch: git checkout -b feature-branch
- Abre el archivo README.md en un editor de texto y añade una línea nueva, por ejemplo: Este es un cambio en la feature branch.
- Guarda los cambios y haz un commit: git add README.md git commit -m "Added a line in feature-branch"

```
PS C:\Users\carina.aranchet\propios\Programacion\conflict-exercise> git checkout -b feature-branch
Switched to a new branch 'feature-branch'
```

```
PS C:\Users\carina.aranchet\propios\Programacion\conflict-exercise> git add README.md
PS C:\Users\carina.aranchet\propios\Programacion\conflict-exercise> git commit -m "Added a line in feature-branch"
[feature-branch cc7da1e] Added a line in feature-branch
 1 file changed, 3 insertions(+), 1 deletion(-)
PS C:\Users\carina.aranchet\propios\Programacion\conflict-exercise>
```

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"

```
PS C:\Users\carina.aranchet\propios\Programacion\conflict-exercise> git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.
PS C:\Users\carina.aranchet\propios\Programacion\conflict-exercise> git add README.md
PS C:\Users\carina.aranchet\propios\Programacion\conflict-exercise> git commit -m "Added a line in main branch"
[main e4d7ea1] Added a line in main branch
 1 file changed, 5 insertions(+), 1 deletion(-)
```

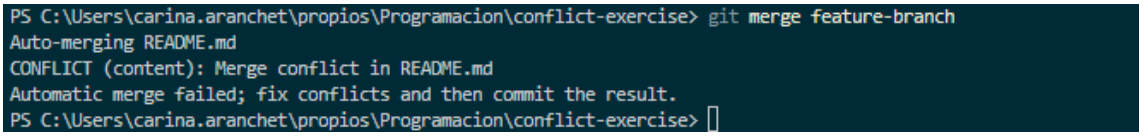
Paso 5: Hacer un merge y generar un conflicto

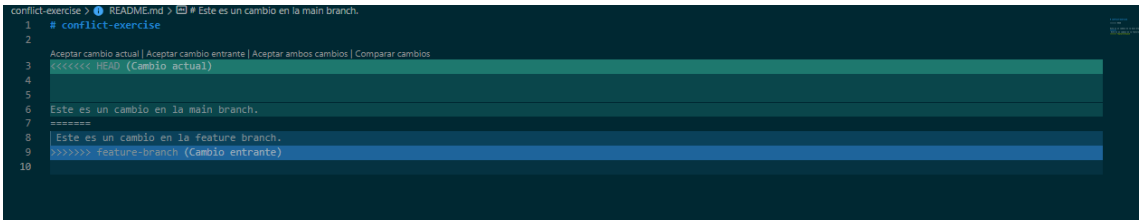
- Intenta hacer un merge de la feature-branch en la rama main:

  git merge feature-branch

- Se generará un conflicto porque ambos cambios afectan la misma línea del

archivo README.md





Paso 6: Resolver el conflicto

• Abre el archivo README.md en tu editor de texto. Verás algo similar a esto:

<<<<<<< HEAD

Este es un cambio en la main branch.

=======

Este es un cambio en la feature branch.

>>>>>>> feature-branch

• Decide cómo resolver el conflicto. Puedes mantener ambos cambios, elegir uno de ellos, o fusionar los contenidos de alguna manera.

• Edita el archivo para resolver el conflicto y guarda los cambios(Se debe borrar lo marcado en verde en el archivo donde estes solucionando el conflicto. Y se debe borrar la parte del texto que no se quiera dejar).

• Añade el archivo resuelto y completa el merge:

git add README.md

git commit -m "Resolved merge conflict"

Paso 7: Subir los cambios a GitHub

• Sube los cambios de la rama main al repositorio remoto en GitHub:

git push origin main

• También sube la feature-branch si deseas:

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.

```
PS C:\Users\carina.aranchet\propios\Programacion\conflict-exercise> git add README.md
PS C:\Users\carina.aranchet\propios\Programacion\conflict-exercise> git commit -m "Resolved merge conflict"
[main 60b6078] Resolved merge conflict
PS C:\Users\carina.aranchet\propios\Programacion\conflict-exercise> git push origin main
Enumerating objects: 11, done.
Counting objects: 100% (11/11), done.
Delta compression using up to 8 threads
Compressing objects: 100% (6/6), done.
Writing objects: 100% (9/9), 808 bytes | 808.00 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/carinaAranchet/conflict-exercise.git
   e27d456..60b6078  main -> main
PS C:\Users\carina.aranchet\propios\Programacion\conflict-exercise>
```