Trabajo Practico 2 Programación I

Cortes Leonardo Nahuel

• ¿Qué es GitHub?

Es una plataforma de alojamiento de repositorios Git basada en la nube.

• ¿Cómo crear un repositorio en GitHub?

1. Inicia sesión en GitHub.
2. Haz clic en **+** (arriba derecha) → **New repository**.
3. Completa:
    - **Nombre del repositorio** (ej: mi-proyecto).
    - Visibilidad (**Público** o **Privado**).
4. Haz clic en **Create repository**.

• ¿Cómo crear una rama en Git?

Se crea con los siguientes comandos por consola.

git branch nombre-rama # Crea la rama

git checkout nombre-rama # Cambia a ella

• ¿Cómo cambiar a una rama en Git?

Se cambia utilizando los siguientes comandos por consola:

git checkout nombre-rama # Método clásico

git switch nombre-rama # Método moderno (recomendado)

• ¿Cómo fusionar ramas en Git?

Con los siguientes pasos:

git checkout main #Cambia a la rama destino

git merge rama #Fusiona la rama

• ¿Cómo crear un commit en Git?

Con los siguientes Pasos:

1. Añade los cambios al **staging area**:

git add archivo.txt # Archivo específico

git add . # Todos los cambios

2. Crea el **commit** con un mensaje descriptivo:
   git commit -m "Mensaje del commit"

• ¿Cómo enviar un commit a GitHub?

Con los siguientes comandos:

git push repositorioRemoto nombre-rama # Sube los commits a GitHub

• ¿Qué es un repositorio remoto?

Es una copia del repositorio Git alojada en un servidor.

• ¿Cómo agregar un repositorio remoto a Git?

Con el siguiente comando:

git remote add origin URL-del-repositorio

• ¿Cómo empujar cambios a un repositorio remoto?

Con el siguiente comando:

git push -u origin nombre-rama

• ¿Cómo tirar de cambios de un repositorio remoto?

git pull origin nombre-rama # Descarga cambios y fusiona

• ¿Qué es un fork de repositorio?

Es una copia personal de un repositorio ajeno en tu cuenta de GitHub.

• ¿Cómo crear un fork de un repositorio?

Con los siguientes pasos:

1. Ve al repositorio en GitHub (ej: usuario/proyecto).
2. Haz clic en **Fork** (arriba derecha).
3. Elige tu cuenta como destino.
4. Clona tu fork localmente:

   git clone https://github.com/tu-usuario/proyecto.git

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

Con los siguientes pasos:

1. Haz un **fork** del repositorio original (en GitHub).
2. Clona tu fork localmente:

   git clone https://github.com/tu-usuario/repo.git

3. Crea una rama para tus cambios:

   git checkout -b mi-rama

4. Realiza commits y súbelos a tu fork:

   git push origin mi-rama

5. En GitHub, ve al repositorio original y haz clic en Pull Request → New Pull Request.
6. Selecciona tu rama (mi-rama) y describe los cambios.
7. Haz clic en **Create Pull Request**.

• ¿Cómo aceptar una solicitud de extracción?

Con los siguientes pasos:

1. Ve a la pestaña **Pull Requests** en GitHub.
2. Revisa los cambios y los comentarios.
3. Si todo está correcto, haz clic en **Merge Pull Request**
4. Confirma con **Confirm Merge**.

• ¿Qué es un etiqueta en Git?

Son marcadores estáticos que apuntan a commits específicos (ej: versiones como v1.0.0). No son lo mismo que las ramas (que sí cambian).

• ¿Cómo crear una etiqueta en Git?

Con los siguientes comandos:

git tag -a v1.0.0 -m "Versión 1.0.0" # Etiqueta anotada

git tag v1.0.0 # Etiqueta ligera

• ¿Cómo enviar una etiqueta a GitHub?

Con los siguientes comandos:

git push origin v1.0.0 # Sube una etiqueta específica

git push --tags # Sube todas las etiquetas

• ¿Qué es un historial de Git?

Es el registro cronológico de commits en un repositorio, que muestra: autores, fechas, mensajes de commit, cambios realizados.

• ¿Cómo ver el historial de Git?

Con los siguientes comandos:

git log # Historial completo

git log --oneline # Versión resumida (1 línea por commit)

git log --graph # Muestra ramas y merges visualmente

• ¿Cómo buscar en el historial de Git?

Se puede buscar filtrando por:

-Autor:

git log --author="nombre"

-Mensaje de commit:

git log --grep="palabra"

- Archivo específico:

git log -- archivo.txt

• ¿Cómo borrar el historial de Git?

Se puede borrar de las siguientes maneras:

Borrar commits recientes (sin eliminar archivos):

git reset --soft HEAD~1 # Borra el último commit, mantiene cambios

Borrar todo el historial (crear un nuevo commit raíz):

git checkout --orphan nueva-rama

git commit -m "Nuevo inicio"

• ¿Qué es un repositorio privado en GitHub?

Es un repositorio solo accesible para usuarios con permisos explícitos.

• ¿Cómo crear un repositorio privado en GitHub?

1. Haz clic en **+** → **New repository**.
2. Elige **Private** en visibilidad.
3. Completa los demás campos y haz clic en **Create repository**.

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

1. Ve a **Settings** → **Collaborators**.
2. Haz clic en **Add people**.
3. Ingresa el nombre de usuario o email y elige permisos (**Read**, **Write**, o **Admin**).
4. El usuario recibirá una invitación por email.

• ¿Qué es un repositorio público en GitHub?

Es un repositorio visible para cualquier persona (aunque solo los colaboradores pueden editarlo).

• ¿Cómo crear un repositorio público en GitHub?

1. Haz clic en **+** → **New repository**.
2. Elige **Public** en visibilidad.
3. Completa los demás campos y haz clic en **Create repository**.

• ¿Cómo compartir un repositorio público en GitHub?

De las siguiente manera:

- **URL directa**: Copia el enlace del repositorio (ej: https://github.com/usuario/repo).
- **README.md**: GitHub muestra automáticamente este archivo en la página principal.
- **GitHub Pages**: Habilita un sitio web estático vinculado al repositorio.

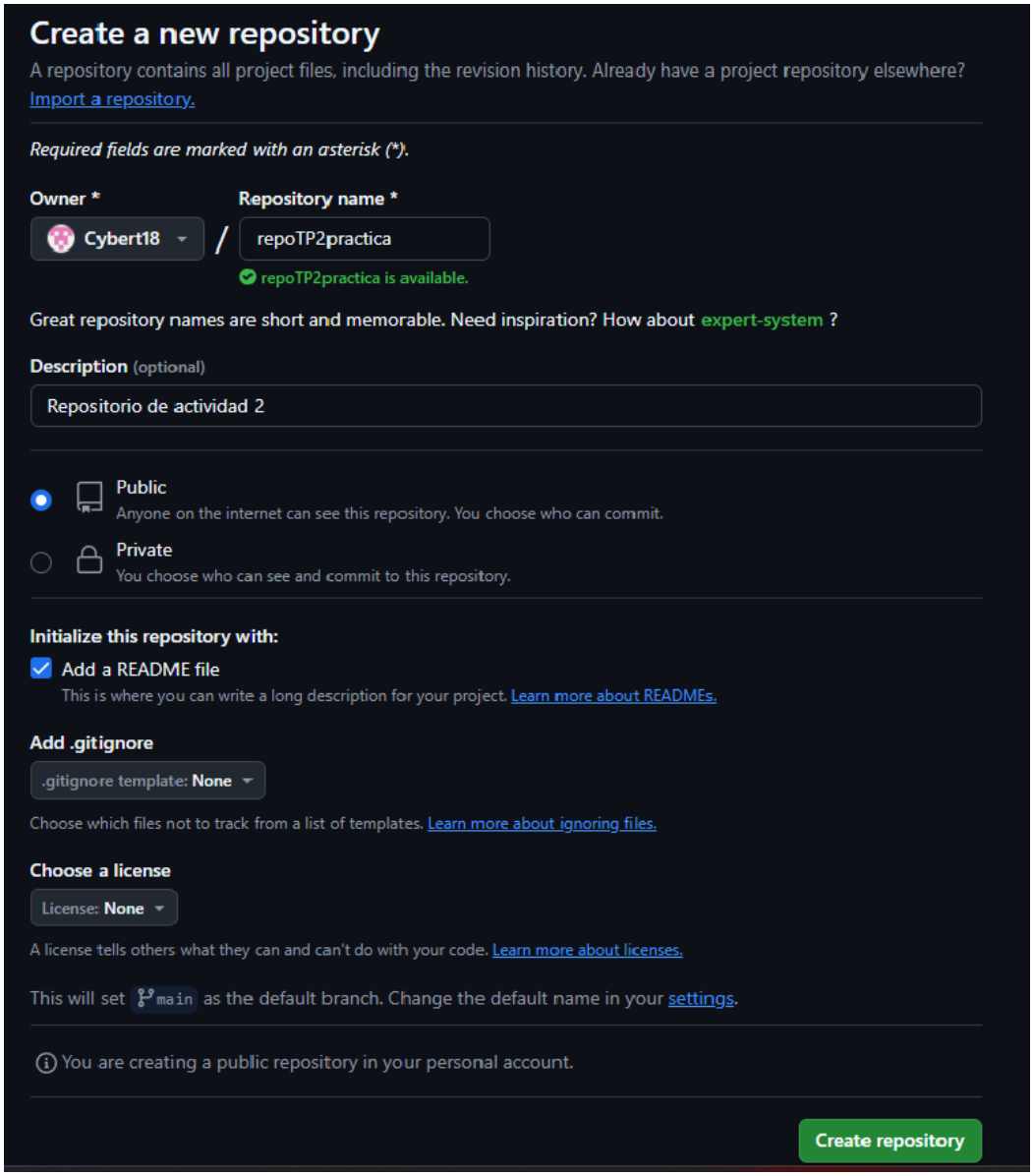
2) Realizar la siguiente actividad:

• Crear un repositorio.

o Dale un nombre al repositorio.

o Elije el repositorio sea público.

o Inicializa el repositorio con un archivo.

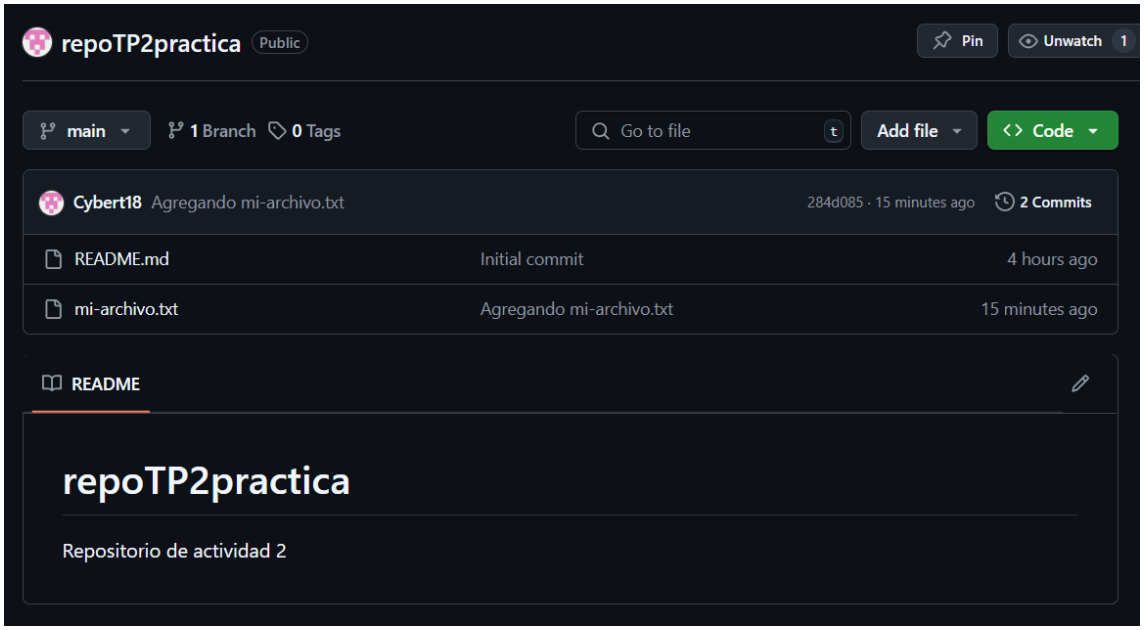


- Agregando un Archivo

    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).

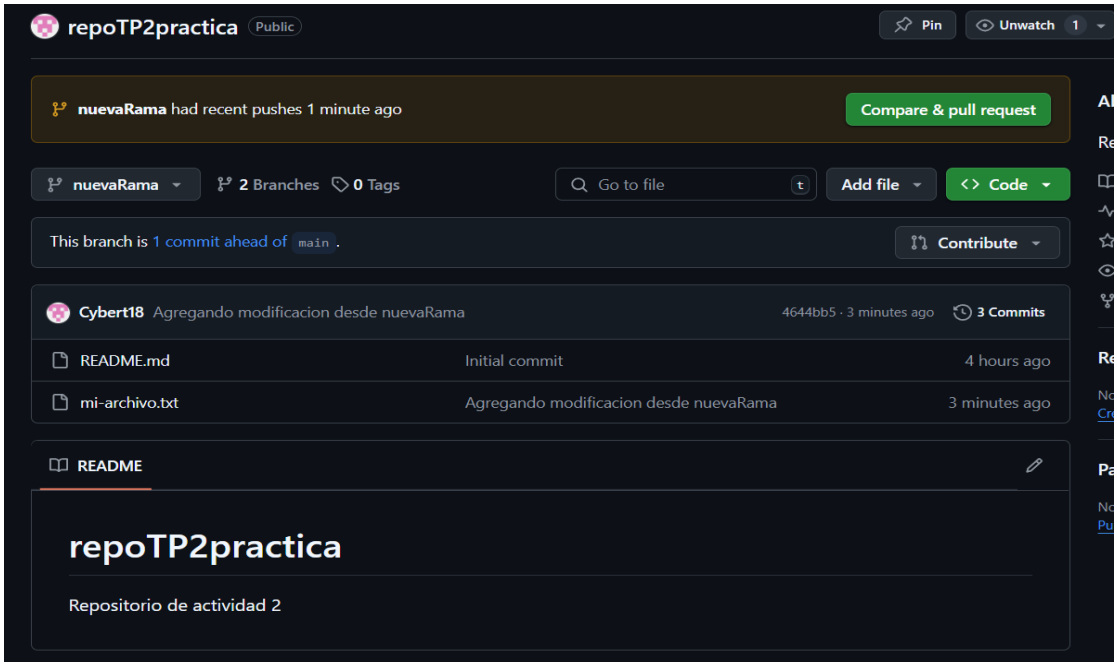


- Creando Branchs

    o Crear una Branch

    o Realizar cambios o agregar un archivo

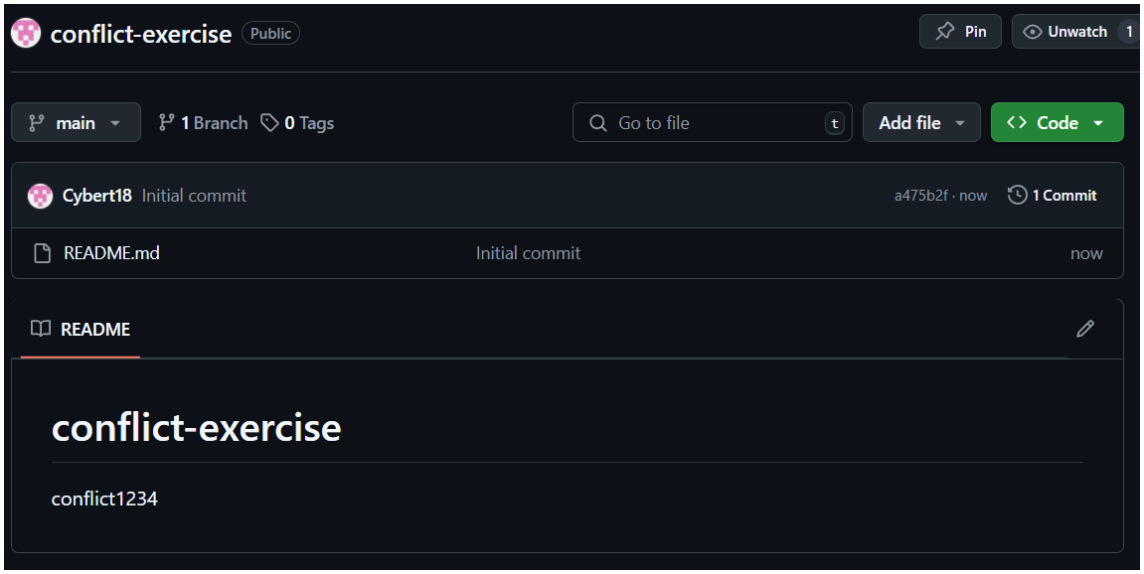
    o Subir la Branch



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

```
C:\Users\Leo Cortez\Documents\UTN\Programación I\s2>git clone https://github.com/Cybert18/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.
```

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"

```
C:\Users\Leo Cortez\Documents\UTN\Programación I\s2>cd conflict-exercise

C:\Users\Leo Cortez\Documents\UTN\Programación I\s2\conflict-exercise>git checkout -b feature-branch
Switched to a new branch 'feature-branch'

C:\Users\Leo Cortez\Documents\UTN\Programación I\s2\conflict-exercise>git add README.md

C:\Users\Leo Cortez\Documents\UTN\Programación I\s2\conflict-exercise>git commit -m "Added a line in feature-branch"
On branch feature-branch
nothing to commit, working tree clean

C:\Users\Leo Cortez\Documents\UTN\Programación I\s2\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"

```
C:\Users\Leo Cortez\Documents\UTN\Programación I\s2\conflict-exercise>git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.

C:\Users\Leo Cortez\Documents\UTN\Programación I\s2\conflict-exercise>git add README.md

C:\Users\Leo Cortez\Documents\UTN\Programación I\s2\conflict-exercise>git commit -m "Added a line in main branch"
[main 4af60a3] Added a line in main branch
 1 file changed, 1 insertion(+)

C:\Users\Leo Cortez\Documents\UTN\Programación I\s2\conflict-exercise>
```

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.