Trabajo Práctico 2: Git y GitHub

Alumna: Astrid Ayelen Añazco

Comisión: 24

Actividades

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

• ¿Qué es GitHub? GitHub es una plataforma de desarrollo colaborativo basada en Git. Allí se alojan repositorios de código, permite colaborar en proyectos de software, revisar cambios y gestionar versiones.

• ¿Cómo crear un repositorio en GitHub?

Para crear un repositorio en GitHub :

1. Iniciar sesión o crear una cuenta de GitHub
2. Ir a “new repositoy” (nuevo repositorio) que aparece en la pagina principal o en el menú
3. Ponerle un nombre y descripción
4. Seleccionar si será público (gratuito) o privado (se paga)
5. Hacer clic en “Create Repository” para finalizar

• ¿Cómo crear una rama en Git?

Para crear una rama en Git, ejecutar en la terminal:

git branch nombre_de_la_rama

• ¿Cómo cambiar a una rama en Git?

git checkout nombre_de_la_rama

• ¿Cómo fusionar ramas en Git?

git merge nombre_de_la_rama

• ¿Cómo crear un commit en Git?

Primero añado los cambios al area de staging con

git add archivo_o_carpeta

Creo el commit con una descripcion

git commit –m “Descripcion”

• ¿Cómo enviar un commit a GitHub?

git push origin nombre_de_la_rama

• ¿Qué es un repositorio remoto?

Es la versión del repositorio que se aloja en un servidor (en nuestro caso, GitHub). Permite colaboración e intercambio de código entre diferentes desarrolladores

• ¿Cómo agregar un repositorio remoto a Git?

git remote add origin https://github.com/tu-usuario/nombre-del-repositorio.git

• ¿Cómo empujar cambios a un repositorio remoto?

git push origin nombre_de_la_rama

• ¿Cómo tirar de cambios de un repositorio remoto?

git pull origin nombre_de_la_rama

• ¿Qué es un fork de repositorio?

Es una copia de un repositorio que se crea en GitHub. Permite modificar el código del repositorio original sin afectar el proyecto fuente.

• ¿Cómo crear un fork de un repositorio?

1. Ingresar al repositorio original en Github
2. Hacer clic en el boton “Fork”
3. Se creará una copia del repositorio en tu propia cuenta

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

Luego de subir los cambios a gitHub, en la página del repositorio original hacer clic en “New Pull Request”, seleccionar la rama del fork con los cambios realizados y se creara una solicitud de extraccción para que los cambios sean revisados e integrados si les parece correcto.

• ¿Cómo aceptar una solicitud de extracción?

Si esta todo correcto con la opcion “Merge pull request” se integran los cambios

• ¿Qué es un etiqueta en Git?

Se utilizan para señalar puntos especificos en la historia del repositorio

• ¿Cómo crear una etiqueta en Git?

git tag nombre_etiqueta

• ¿Cómo enviar una etiqueta a GitHub?

git push origin nombre_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.

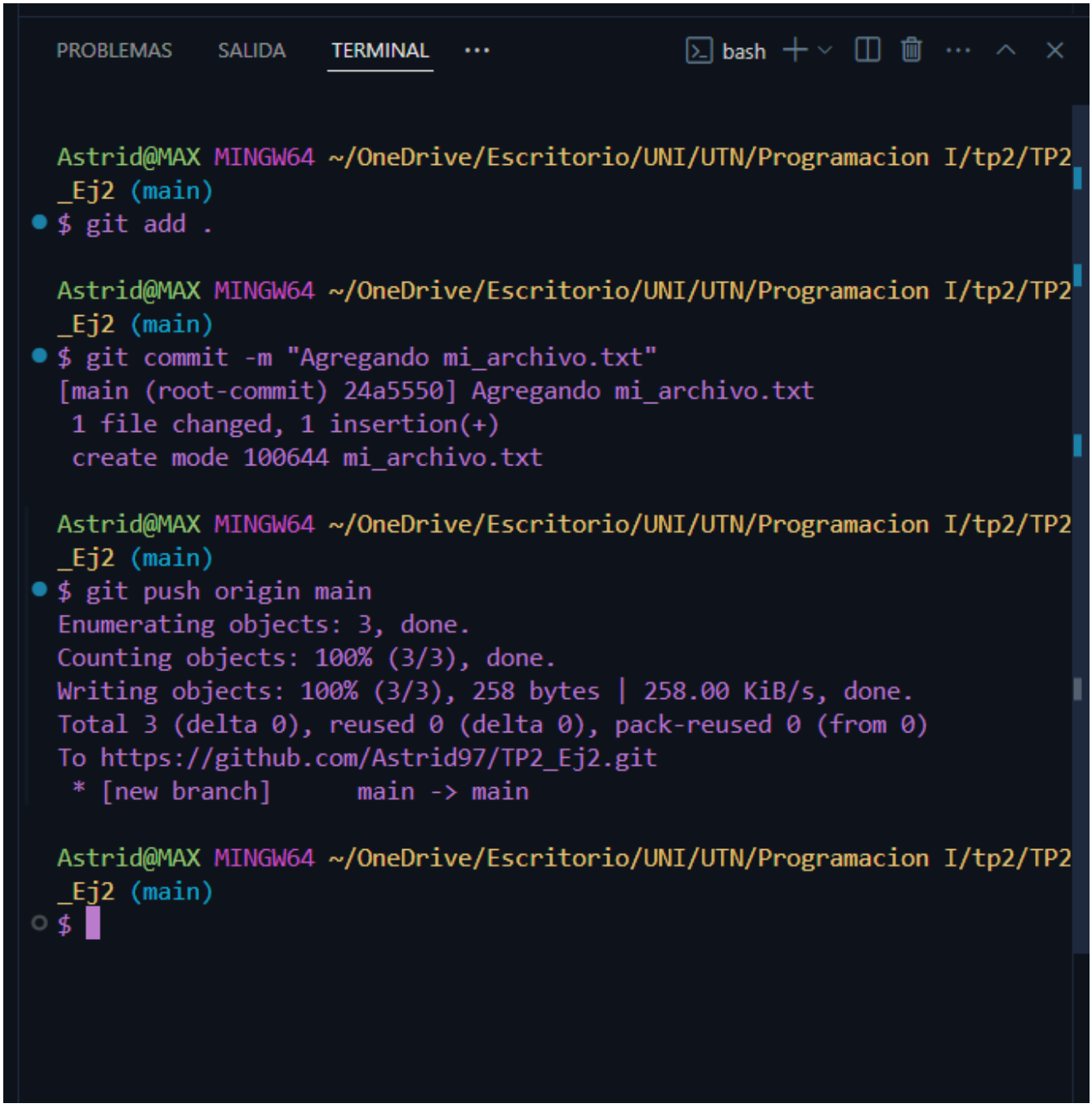
2) Realizar la siguiente actividad:

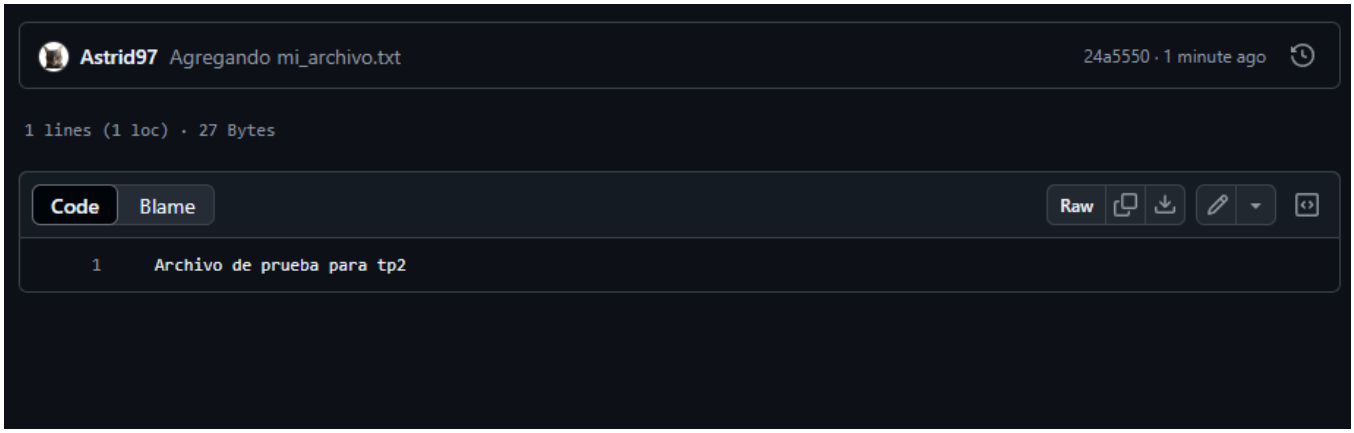
• Crear un repositorio:

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).

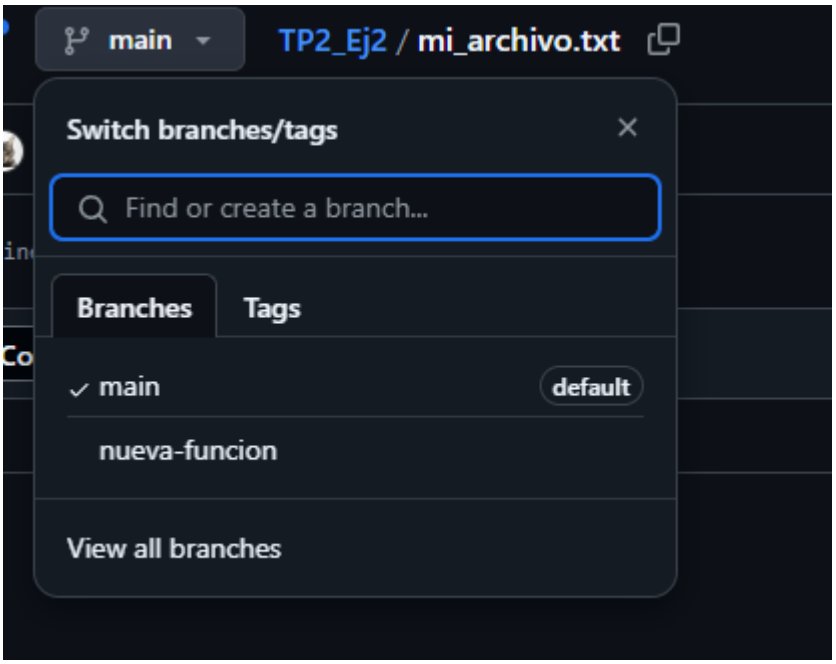




- Creando Branchs:

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

```
PS C:\Users\Astrid\OneDrive\Escritorio\UNI\UTN\Programacion I\tp2\TP2_Ej2> git branch nueva-funcion
PS C:\Users\Astrid\OneDrive\Escritorio\UNI\UTN\Programacion I\tp2\TP2_Ej2> git checkout nueva-funcion
M       mi_archivo.txt
Switched to branch 'nueva-funcion'
PS C:\Users\Astrid\OneDrive\Escritorio\UNI\UTN\Programacion I\tp2\TP2_Ej2> git add .
PS C:\Users\Astrid\OneDrive\Escritorio\UNI\UTN\Programacion I\tp2\TP2_Ej2> git commit -m "agregando cambios en la nueva rama"
[nueva-funcion 6420f25] agregando cambios en la nueva rama
 1 file changed, 1 insertion(+)
PS C:\Users\Astrid\OneDrive\Escritorio\UNI\UTN\Programacion I\tp2\TP2_Ej2> git push -u origin nueva-funcion
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), 308 bytes | 308.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
remote:
remote: Create a pull request for 'nueva-funcion' on GitHub by visiting:
remote:      https://github.com/Astrid97/TP2_Ej2/pull/new/nueva-funcio
remote:
To https://github.com/Astrid97/TP2_Ej2.git
 * [new branch]      nueva-funcion -> nueva-funcion
branch 'nueva-funcion' set up to track 'origin/nueva-funcion'.
PS C:\Users\Astrid\OneDrive\Escritorio\UNI\UTN\Programacion I\tp2\TP2_Ej2>
```



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

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"

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"

```
Astrid@MAX MINGW64 ~/OneDrive/Escritorio/UNI/UTN/Programacion I/tp2ejercicio3/Conflict-exercise (feature-branch)
$ git commit -m "Added a line in feature-branch"
[feature-branch 2e0116a] Added a line in feature-branch
 1 file changed, 1 insertion(+)

Astrid@MAX MINGW64 ~/OneDrive/Escritorio/UNI/UTN/Programacion I/tp2ejercicio3/Conflict-exercise (feature-branch)
$ git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.

Astrid@MAX MINGW64 ~/OneDrive/Escritorio/UNI/UTN/Programacion I/tp2ejercicio3/Conflict-exercise (main)
$ git add README.md

Astrid@MAX MINGW64 ~/OneDrive/Escritorio/UNI/UTN/Programacion I/tp2ejercicio3/Conflict-exercise (main)
$ git commit -m "Added a line in main branch"
[main 27d4bad] 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.

```
Astrid@MAX MINGW64 ~/OneDrive/Escritorio/UNI/UTN/Programacion I/tp2ejercicio3/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.
```

Paso 6: Resolver el conflicto

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

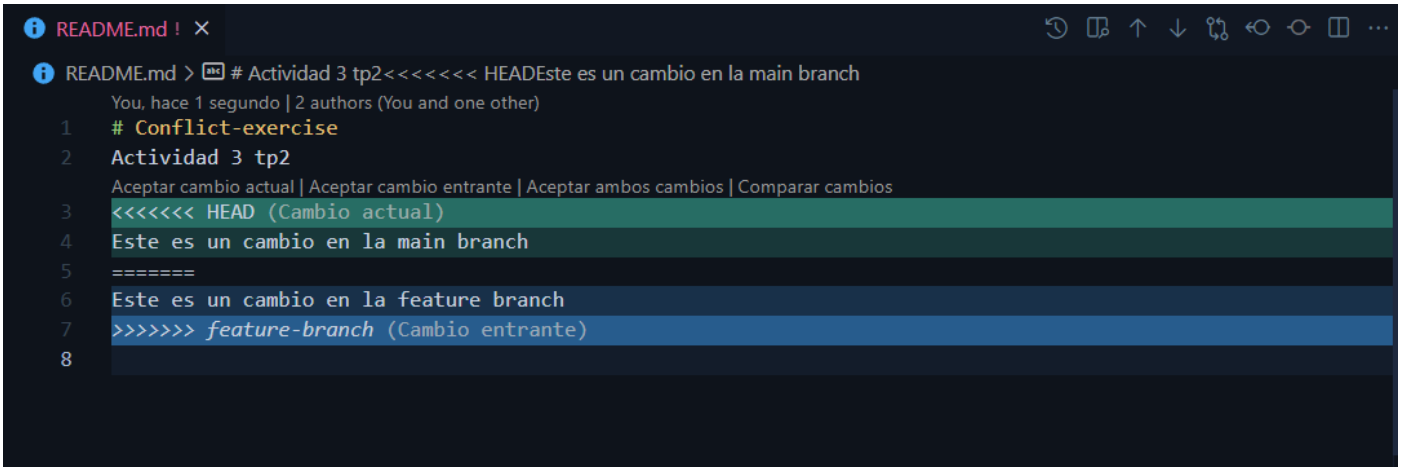
<<<<<<< HEAD

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

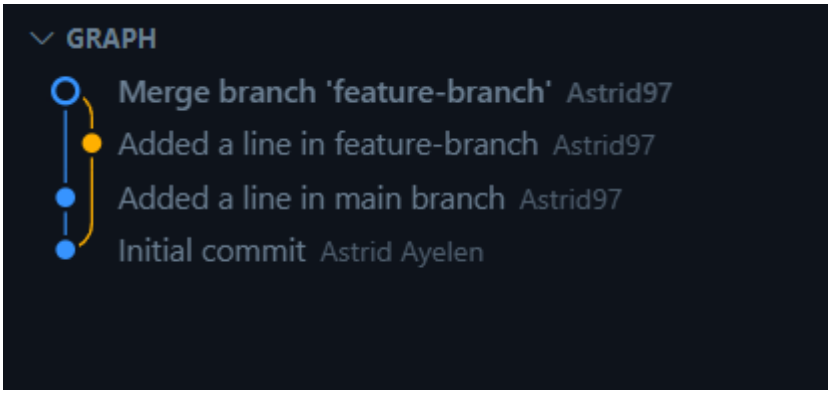
• También sube la feature-branch si deseas:

git push origin feature-branch

```
Astrid@MAX MINGW64 ~/OneDrive/Escritorio/UNI/UTN/Programacion I/tp2ejercicio3/Conflict-exercise (main)
$ git push origin main
Everything up-to-date

Astrid@MAX MINGW64 ~/OneDrive/Escritorio/UNI/UTN/Programacion I/tp2ejercicio3/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/Astrid97/Conflict-exercise/pull/new/feature-branch
remote:
To https://github.com/Astrid97/Conflict-exercise.git
 * [new branch]      feature-branch -> feature-branch

Astrid@MAX MINGW64 ~/OneDrive/Escritorio/UNI/UTN/Programacion I/tp2ejercicio3/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.