

Actividades

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

- ¿Qué es GitHub?

GitHub es un servicio de alojamiento para los repositorios de Git. (Git es un sistema de control de versiones de código abierto, diseñado para pequeños y grandes proyectos).

- ¿Cómo crear un repositorio en GitHub?

1. Ingresar a GitHub en <https://github.com> e iniciar sesión.
2. Para crear un repositorio, hacer clic en el icono + (esquina superior derecha) y elegir Nuevo repositorio.
3. Una vez creado se debe completar los datos, como nombre del repositorio, descripción del proyecto y elegir que sea privado o público.

- ¿Cómo crear una rama en Git?

1. Desde VS code con un proyecto ya inicializado con Git, abrir la terminal
2. Ver en qué rama nos encontramos con el comando `git branch`
3. Crear una nueva rama con `git branch "nombre-rama"`
4. Una vez creada, con el siguiente comando `git checkout "nombre-rama"`, nos movemos a esa rama para iniciar el trabajo.
5. Una alternativa para crear y pasar a la nueva rama de una vez, es utilizar el comando `git checkout -b "nombre-rama"`

- ¿Cómo cambiar a una rama en Git?

Con el siguiente comando `git checkout "nombre-rama"`, nos movemos a esa rama para iniciar el trabajo.

- ¿Cómo fusionar ramas en Git?

1. Para fusionar ramas, como primera medida guardar los cambios con `git add .` y luego `git commit -m "Describir los cambios"`

Con esto nos aseguramos que el trabajo quede guardado.

2. Cambiar a la rama donde se quiere traer los cambios, supongamos que va a ser en el master, utilizamos el comando `git checkout "master"`.
3. Posicionados en la rama adecuada, traemos los cambios de la otra rama con `git merge "nombre-rama"`.

Si al fusionar hay conflictos entre las ramas, te avisa para que puedas resolver manualmente el problema.

- ¿Cómo crear un commit en Git?

1. Estando en la carpeta del proyecto en el cual se trabaja con Git, verificar en que rama nos encontramos con *git branch*.
2. Ver cuales archivos se cambiaron desde el último *commit*, con *git status*.
3. El siguiente paso es agregar los cambios con *git add*.
4. Agregado los cambios, se procede a realizar el commit, *git commit -m "Mensaje descriptivo"*

- ¿Cómo enviar un commit a GitHub?

1-Para enviar un commit, antes verificar que el proyecto esté conectado a un repositorio en GitHub, para verificarlo utilizar el comando *git remote -v*, si no está inicializado hacerlo de la siguiente manera *git remote add origin https://github.com/tu-usuario/tu-repo.git*, para comprobarlo si se encuentra inicializado utilizar *git remote -v*.

2- Abrir la terminal y ejecutar los siguientes comandos:

- a) *git add* . para agregar los archivos y cambios,
- b) luego *git commit -m "Descripción"* , para confirmar los cambios,
- c) y por ultimo un *git push origin "nombre de la rama"*, subir los cambios al repositorio remoto.

- ¿Qué es un repositorio remoto?

Un repositorio remoto es una versión del proyecto que está alojada en un servidor, normalmente en la nube como GitHub. Es accesible desde cualquier computadora con acceso a Internet y sirve para compartir tu código con otras personas o simplemente respaldarlo.

- ¿Cómo agregar un repositorio remoto a Git?

- 1- Con un repositorio ya creado en GitHub, copiar el link del repositorio remoto.
- 2- Dentro de VScode abrir la carpeta del proyecto e iniciar git con *git init*.
- 3- Agregar el repositorio remoto del link que copiamos de esta manera *git remote add origin link copiado*.
- 4- Podemos verificar que se agrego con el comando *git remote -v*

Con eso realizado se conectó el proyecto local con el repositorio remoto de GitHub.

- ¿Cómo empujar cambios a un repositorio remoto?

Una vez agregados los cambios con *git add* . y luego un *git commit -m "Descripción de cambios"* , proceder de la siguiente manera.

- 1- Si es la primera vez que se realiza un push (empujar cambios), utilizar *git push -u origin master*, -u vincula la rama local con la rama remota.
- 2- La próxima vez solo usar *git push*

- ¿Cómo tirar de cambios de un repositorio remoto?

Encontradome en la carpeta del proyecto, verifico en que rama estoy con *git branch*, paso siguiente es tirar de los cambios con *git pull origin nombre de la rama*

- ¿Qué es un fork de repositorio?

Un fork es una copia de un repositorio creado en una cuenta diferente permitiendo desarrollar cambios sin afectar el original.

- ¿Cómo crear un fork de un repositorio?

- 1- Ingresar al repositorio que se quiere copiar en GitHub.
- 2- Hacer click en el botón fork, arriba a la derecha.
- 3- GitHub crear una copia del proyecto en el perfil del usuario.

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

Un Pull request es una forma de decirle al dueño del repositorio que se hicieron cambios en el repositorio que se copió, y se requiere que los revise para incorporarlo al original.

Pasos para enviar la solicitud:

- 1- Una vez realizados los cambios en el código, hacer un *git push* al repositorio en cual se hizo el fork.
- 2- Dentro de GitHub vamos al repositorio donde nos mostrara un cartel donde dice *Compare & pull request*.
- 3- Hacemos click sobre Compare & pull request y paso siguiente tenemos que completar título, descripción y elegir si la solicitud la hacemos sobre el repositorio base (original) o sobre el repositorio del fork que creamos.
- 4- Y por último hacer clic en crear la solicitud de extracción.

- ¿Cómo aceptar una solicitud de extracción?

Esto se hace desde GitHub

- 1- Ir al repositorio y entrar al pestaña Pull request, y junto al botón Código, va a ver un listado de solicitudes pendientes.
- 2- Hacer clic sobre pull request sobre la que se quiere revisar, un vez ingresado se puede visualizar los archivos que fueron modificados, quien lo envió y desde que rama.
- 3- Si todo está bien, paso siguiente es fusionar las ramas con el botón que dice *merge pull request*

- ¿Qué es un etiqueta en Git?

Una etiqueta en Git es una marca que se coloca en un punto específico del historial de confirmaciones, generalmente para indicar una versión importante del proyecto.

- ¿Cómo crear una etiqueta en Git?

Dentro del proyecto con el Git inicializado, crear la etiqueta con el siguiente comando:

- 1- *git tag -a nombre -m "mensaje"*
- a: crea una etiqueta anotada
-Donde *nombre* es la etiqueta que puede ser la versión ej: v1.0
-m: mensaje que describe esa versión

- ¿Cómo enviar una etiqueta a GitHub?

Para subir una etiqueta a GitHub, utilizar el comando *git push origin nombre de la etiqueta*; y para subir todas la etiquetas utilizar *git push origin --tags*

- ¿Qué es un historial de Git?

El historial de Git es el registro completo de todos los cambios (commits) que se hicieron en un repositorio a lo largo del tiempo, como que archivos fueron modificados, quien hizo el cambio, un mensaje describiendo el cambio y cuando se hizo.

- ¿Cómo ver el historial de Git?

Desde la terminal usar el comando *git log*, esto nos muestra la fecha, mensaje, quien lo hizo.

- ¿Cómo buscar en el historial de Git?

Desde la terminal utilizar el comando *git log*, esto muestra:

Autor

Fecha

Mensaje

- ¿Cómo borrar el historial de Git?

Se utiliza el *git reset*, que permite retroceder el puntero de historial a un commit anterior, es decir puede deshacer uno o más commits recientes, pero sin borrar todo el historial

Hay diferentes modos:

Git reset - -soft <identificador del commit> : vuelve al commit especificado, pero mantiene los cambios del área.

Git reset - -mixed <identificador del commit> : Borra el stage pero mantiene los archivos modificado en el directorio de trabajo.

Git reset - -hard <identificador del commit> : Elimina los commits y también los archivos modificados.

- ¿Qué es un repositorio privado en GitHub?

Un repositorio privado en GitHub es un proyecto solo visible para el usuario que lo creo y para los usuarios que se les dé permiso, o sea no está abierto públicamente y no es visible en internet a menos que tengas los permisos.

- ¿Cómo crear un repositorio privado en GitHub?

- 1- Haber iniciado sesión en <https://github.com>
- 2- En la parte superior derecha hacer clic en +, elegir nuevo repositorio
- 3- Completar los datos como nombre, descripción etc...
- 4- Seleccionar la opción "Privado"
- 5- Inicializar el repositorio con un Readme (opcional), esto es para tener un archivo inicial
- 6- Hacer clic en el botón verde "Crear repositorio"

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

- 1- Ir a “Configuración” arriba a la derecha del repositorio.
- 2- En el menú de la izquierda arriba clic donde dice “Colaboradores”.
- 3- Hacer clic en “Invitar/agregar personas/colaboradores”.
- 4- Escribir el nombre de usuario de GitHub del colaborador.
- 5- Seleccionar cuando aparezca y luego en Añadir.

- ¿Qué es un repositorio público en GitHub?

Un repositorio público es un proyecto que puede ser visto por cualquier colaborador en internet. No es necesario iniciar sesión ni pedir permisos para acceder el código.

- ¿Cómo crear un repositorio público en GitHub?

- 1- Haber iniciado sesión en <https://github.com>
- 2- En la parte superior derecha hacer clic en +, elegir nuevo repositorio
- 3- Completar los datos como nombre, descripción etc...
- 4- Seleccionar la opción “Publico”
- 5- Inicializar el repositorio con un Readme (opcional), esto es para tener un archivo inicial
- 6- Hacer clic en el botón verde “Crear repositorio”

- ¿Cómo compartir un repositorio público en GitHub?

- 1- Copiar el enlace del repositorio, el que se encuentra en la barra de navegación.
- 2- Enviar enlace al colaborador que se quiera.

2) Realizar la siguiente actividad:


- Crear un repositorio.
 - Dale un nombre al repositorio.
 - Elije el repositorio sea público.
 - Inicializa el repositorio con un archivo.

Crear un nuevo repositorio

Un repositorio contiene todos los archivos del proyecto, incluido el historial de revisiones. ¿Ya tienes un repositorio de proyectos en otro lugar? [Importa un repositorio](#).

Los campos obligatorios están marcados con un asterisco (*).

Dueño * **Nombre del repositorio ***


 Claudio482 / RepodePractica-UTN-Clau


✓ RepodePractica-UTN-Claudio está disponible.

Los buenos nombres de repositorios son cortos y fáciles de recordar. ¿Necesitas inspiración? ¿Qué te parece? [invención crujiente](#) ?

Descripción (opcional)

Primer Repositorio creado para practicar

☒  **Público**
Cualquier persona en internet puede ver este repositorio. Tú decides quién puede contribuir.

☐  **Privado**
Tú eliges quién puede ver y comprometerse con este repositorio.

Inicialice este repositorio con:

☒ **Agregar un archivo README**
Aquí puedes escribir una descripción detallada de tu proyecto. [Obtén más información sobre los archivos README](#).

Agregar .gitignore


Plantilla .gitignore: Ninguna

Seleccione los archivos que no desea rastrear de una lista de plantillas. [Obtenga más información sobre cómo ignorar archivos](#).

Elija una licencia

Licencia: Ninguna

Una licencia indica a otros qué pueden y no pueden hacer con tu código. [Obtén más información sobre las licencias](#).

Esto establecerá  **principal** Como rama predeterminada. Cambia el nombre predeterminado en la [configuración](#).

① Estás creando un repositorio público en tu cuenta personal.

Crear repositorio

• 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).

 InformePráctica-UTN-Claudio Público Alfiler Dejar de ver 1

 principal 1 sucursal 0 etiquetas t Agregar archivo <> Código

 Claudio482	Agregando mi-archivo.txt	ae76e73 · hace 4 minutos	🕒 2 confirmaciones
 README.md	Compromiso inicial	hace 1 hora	
 mi-archivo.txt	Agregando mi-archivo.txt	hace 4 minutos	

 **LÉAME** 

InformePráctica-UTN-Claudio

Primer Repositorio creado para practicar

- Creando Branchs

- Crear una Branch
- Realizar cambios o agregar un archivo
- 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".

Crear un nuevo repositorio

Un repositorio contiene todos los archivos del proyecto, incluido el historial de revisiones. ¿Ya tienes un repositorio de proyectos en otro lugar? [Importa un repositorio](#).

Los campos obligatorios están marcados con un asterisco (*).

Dueño * **Nombre del repositorio ***

 Claudio482 / conflict-exercise

✓ El ejercicio de conflicto está disponible.

Los buenos nombres de repositorios son cortos y fáciles de recordar. ¿Necesitas inspiración? ¿Qué te parece? [super-duper-cuchara](#) ?

Descripción (opcional)

Ejercicio de práctica de conflictos en Git

☐ Público
Cualquier persona en internet puede ver este repositorio. Tú decides quién puede contribuir.

☒ Privado
Tú eliges quién puede ver y comprometerse con este repositorio.

Inicialice este repositorio con:

☒ Agregar un archivo README
Aquí puedes escribir una descripción detallada de tu proyecto. [Obtén más información sobre los archivos README](#).

Agregar .gitignore

Plantilla .gitignore : Ninguna

Seleccione los archivos que no desea rastrear de una lista de plantillas. [Obtenga más información sobre cómo ignorar archivos](#).

Elija una licencia


Licencia : Ninguna




Una licencia indica a otros qué pueden y no pueden hacer con tu código. [Obtén más información sobre las licencias](#).


Esto establecerá [principal](#) como rama predeterminada. Cambia el nombre predeterminado en la [configuración](#).


❗ Estás creando un repositorio privado en tu cuenta personal.



Crear repositorio

 **conflict-exercise** Private Unwatch 1

 main  1 Branch  0 Tags Add file Code

 Claudio482 Initial commit e8476f0 · now 1 Commit

 README.md Initial commit now

 README 

conflict-exercise

Ejercicio de práctica de conflictos en Git



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
```


✓ SIN TÍTULO (ÁREA DE TRABAJO)    

✓ ConflictExercise

✓ conflict-exercise

 README.md

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"
```

```
[feature-branch 511409f] 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"
```

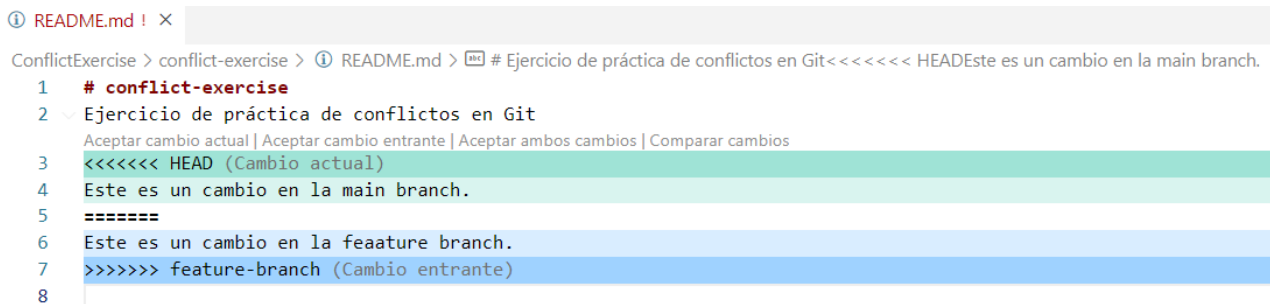
```
[main 55b3ca4] Added a line in main branch  
1 file changed, 1 insertion(+)
```

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.



```
1 # conflict-exercise
2 Ejercicio de práctica de conflictos en Git
3 <<<<<< HEAD (Cambio actual)
4 Este es un cambio en la main branch.
5 =====
6 Este es un cambio en la feaature branch.
7 >>>>>> feature-branch (Cambio entrante)
8
```

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"
```

```
[main 070f00f] 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
```

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.

Commits


 main



🔍 All users

📅 All time


 Commits on Mar 30, 2025



Resolved merge conflict

 Claudio482 committed 9 minutes ago


070f00f  



Added a line in main branch


 Claudio482 committed 24 minutes ago

55b3ca4  


Added a line in feature-branch

 Claudio482 committed 32 minutes ago

511409f  

 Commits on Mar 29, 2025

Initial commit

 Claudio482 authored 6 hours ago

Verified e8476f0 