

## Práctico 2: Git y GitHub

### Actividades

1) Contestar las siguientes preguntas utilizando las guías y documentación proporcionada

(Desarrollar las respuestas) :

- ¿Qué es GitHub?
- ¿Cómo crear un repositorio en GitHub?
- ¿Cómo crear una rama en Git?
- ¿Cómo cambiar a una rama en Git?
- ¿Cómo fusionar ramas en Git?
- ¿Cómo crear un commit en Git?
- ¿Cómo enviar un commit a GitHub?
- ¿Qué es un repositorio remoto?
- ¿Cómo agregar un repositorio remoto a Git?
- ¿Cómo empujar cambios a un repositorio remoto?
- ¿Cómo tirar de cambios de un repositorio remoto?
- ¿Qué es un fork de repositorio?
- ¿Cómo crear un fork de un repositorio?
- ¿Cómo enviar una solicitud de extracción (pull request) a un repositorio?
- ¿Cómo aceptar una solicitud de extracción?
- ¿Qué es un etiqueta en Git?
- ¿Cómo crear una etiqueta en Git?
- ¿Cómo enviar una etiqueta a GitHub?
- ¿Qué es un historial de Git?
- ¿Cómo ver el historial de Git?
- ¿Cómo buscar en el historial de Git?
- ¿Cómo borrar el historial de Git?
- ¿Qué es un repositorio privado en GitHub?
- ¿Cómo crear un repositorio privado en GitHub?
- ¿Cómo invitar a alguien a un repositorio privado en GitHub?
- ¿Qué es un repositorio público en GitHub?
- ¿Cómo crear un repositorio público en GitHub?
- ¿Cómo compartir un repositorio público en GitHub?

### ¿Qué es GitHub?

GitHub es una plataforma de desarrollo colaborativo que permite a los desarrolladores almacenar, gestionar y seguir el desarrollo de proyectos de software utilizando el sistema de control de versiones Git. Ofrece características como la gestión de repositorios, seguimiento de problemas, revisiones de código, y colaboración a través de "pull requests".

## ¿Cómo crear un repositorio en GitHub?

Pasos para crear un repositorio en GitHub:

1. Inicia sesión en tu cuenta de GitHub.
2. Haz clic en el botón "New" o "Nuevo" en la página principal de tu perfil.
3. Completa el formulario con el nombre del repositorio, una descripción opcional y selecciona si deseas que sea público o privado.
4. Haz clic en "Create repository" para finalizar.

## ¿Cómo crear una rama en Git?

Para crear una rama en Git, utilizar el siguiente comando en la terminal:

- `git branch nombre-de-la-rama`

Esto creará una nueva rama basada en la rama actual.

## ¿Cómo cambiar a una rama en Git?

Para cambiar a una rama existente, utiliza el siguiente comando:

- `git checkout nombre-de-la-rama`

## ¿Cómo fusionar ramas en Git?

Para fusionar una rama en la rama actual, primero asegúrate de estar en la rama en la que deseas realizar la fusión y luego ejecuta:

- `git merge nombre-de-la-rama-a-fusionar`

## ¿Cómo crear un commit en Git?

Para crear un commit, primero añade los cambios al área de preparación (staging area) con:

- `git add .`

Luego, ejecuta el siguiente comando para crear el commit:

- `git commit -m "Mensaje del commit"`

### **¿Cómo enviar un commit a GitHub?**

Para enviar tus commits a GitHub, utiliza el comando:

- `git push origin nombre-de-la-rama`

Esto enviará tus cambios a la rama especificada en el repositorio remoto.

### **¿Qué es un repositorio remoto?**

Un repositorio remoto es una versión de tu proyecto que está alojada en un servidor en línea, como GitHub. Permite a los desarrolladores colaborar y compartir su trabajo a través de Internet.

### **¿Cómo agregar un repositorio remoto a Git?**

Para agregar un repositorio remoto, utilizar el siguiente comando:

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

Reemplaza `url-del-repositorio` con la URL de tu repositorio en GitHub.

### **¿Cómo empujar cambios a un repositorio remoto?**

Para empujar cambios a un repositorio remoto, utiliza:

- `git push origin nombre-de-la-rama`

### **¿Cómo tirar de cambios de un repositorio remoto?**

Para obtener los cambios más recientes de un repositorio remoto, utiliza:

- `git pull origin nombre-de-la-rama`

### **¿Qué es un fork de repositorio?**

Un fork es una copia de un repositorio que permite a los desarrolladores experimentar y realizar cambios sin afectar el repositorio original. Es común en proyectos de código abierto.

### **¿Cómo crear un fork de un repositorio?**

Para crear un fork de un repositorio en GitHub, ve a la página del repositorio y haz clic en el botón "Fork" en la esquina superior derecha. Esto creará una copia del repositorio en tu cuenta de GitHub.

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

Para enviar una pull request:

1. Realiza cambios en tu fork y súbelos a GitHub.
2. Ve a la página de tu fork y haz clic en "Pull request".
3. Selecciona la rama que deseas fusionar y la rama del repositorio original.
4. Completa el formulario y haz clic en "Create pull request".

### **¿Cómo aceptar una solicitud de extracción?**

Para aceptar una pull request:

1. Ve a la página del repositorio original.
2. Haz clic en "Pull requests" y selecciona la solicitud que deseas aceptar.
3. Revisa los cambios y haz clic en "Merge pull request" para fusionar.

### **¿Qué es una etiqueta en Git?**

Una etiqueta es una referencia que apunta a un commit específico en el historial de Git. Se utiliza comúnmente para marcar versiones de lanzamiento.

### **¿Cómo crear una etiqueta en Git?**

Para crear una etiqueta, utiliza el siguiente comando:

- `git tag -a nombre-de-la-etiqueta -m "Mensaje de la etiqueta"`

### **¿Cómo enviar una etiqueta a GitHub?**

Para enviar una etiqueta a GitHub, utiliza:

- `git push origin nombre-de-la-etiqueta`

### **¿Qué es un historial de Git?**

El historial de Git es un registro de todos los commits realizados en un repositorio. Permite a los desarrolladores ver los cambios a lo largo del tiempo.

### **¿Cómo ver el historial de Git?**

Para ver el historial de commits, utiliza:

- `git log`

### **¿Cómo buscar en el historial de Git?**

Puedes buscar en el historial utilizando:

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

### **¿Cómo borrar el historial de Git?**

Borrar el historial de Git no es una práctica común, pero si necesitas hacerlo, puedes usar comandos avanzados como `git rebase` o `git reset`. Sin embargo, ten cuidado, ya que esto puede afectar la integridad del repositorio.

### **¿Qué es un repositorio privado en GitHub?**

Un repositorio privado es un repositorio que solo puede ser visto y accedido por usuarios específicos. No es visible para el público en general.

### **¿Cómo crear un repositorio privado en GitHub?**

Al crear un nuevo repositorio, selecciona la opción "Private" en la configuración del repositorio.

### **¿Cómo invitar a alguien a un repositorio privado en GitHub?**

Para invitar a alguien a un repositorio privado:

1. Ve a la página del repositorio.

2. Haz clic en "Settings" y luego en "Manage access".
3. Haz clic en "Invite a collaborator" y escribe el nombre de usuario o correo electrónico de la persona.

### **¿Qué es un repositorio público en GitHub?**

Un repositorio público es un repositorio que puede ser visto y accedido por cualquier persona en Internet.

### **¿Cómo crear un repositorio público en GitHub?**

Al crear un nuevo repositorio, selecciona la opción "Public" en la configuración del repositorio.

### **¿Cómo compartir un repositorio público en GitHub?**

Para compartir un repositorio público, simplemente comparte la URL del repositorio con otros. También puedes utilizar las funciones de "Share" en GitHub para enviar enlaces directos.

2) Realizar la siguiente actividad:

Crear un repositorio.

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

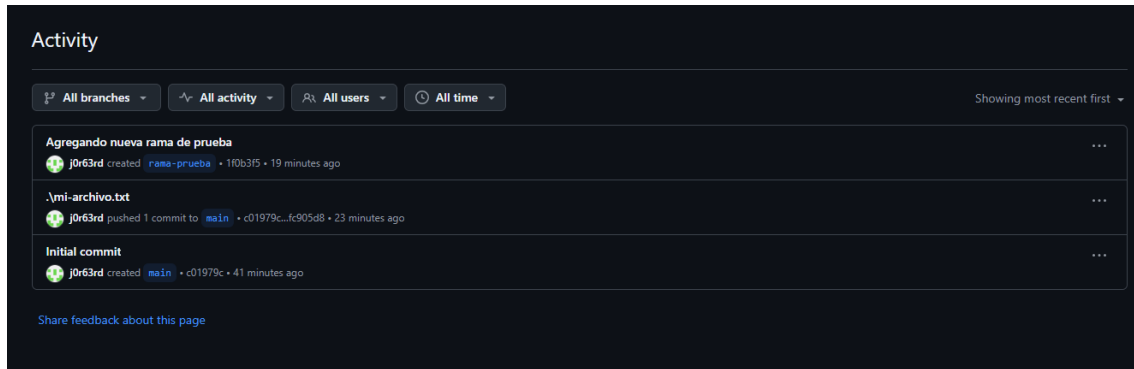
Agregando un Archivo

- 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).

## Creando Branchs

- Crear una Branch
- Realizar cambios o agregar un archivo
- Subir la Branch

[https://github.com/j0r63rd/Practico\\_2\\_Git-GitHub.git](https://github.com/j0r63rd/Practico_2_Git-GitHub.git)



### 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"`
- 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

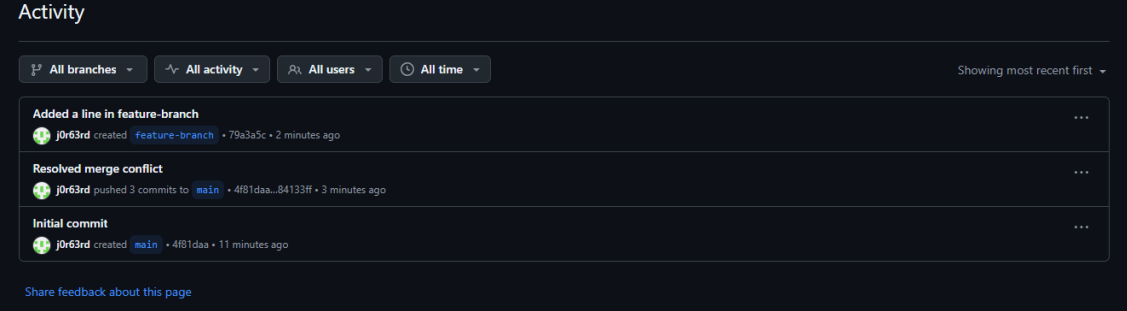
- Sube los cambios de la rama main al repositorio remoto en GitHub:



git push origin main

- También sube la feature-branch si deseas:
- 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.

<https://github.com/j0r63rd/conflict-exercise.git>



The screenshot shows the GitHub Activity page for the repository `j0r63rd/conflict-exercise.git`. The page has a dark theme. At the top, there are filters for "All branches", "All activity", "All users", and "All time". The activity list shows three recent events:

- Added a line in feature-branch**: `j0r63rd` created `feature-branch` • 79a3a5c • 2 minutes ago
- Resolved merge conflict**: `j0r63rd` pushed 3 commits to `main` • 4f81daa..84133ff • 3 minutes ago
- Initial commit**: `j0r63rd` created `main` • 4f81daa • 11 minutes ago

At the bottom, there is a link to "Share feedback about this page".