

Actividad 1:

- ¿Qué es GitHub?

GitHub es una comunidad donde podemos compartir nuestros repositorios de forma pública o privada

- ¿Cómo crear un repositorio en GitHub?

1) Inicia sesión en GitHub.

2) Haz clic en el botón **"New"** o ve a **Repositories > New**.

3) Escribe un nombre para el repositorio.

4) Elige si será **público** o **privado**.

5) (Opcional) Agrega un README, .gitignore o licencia.

6) Haz clic en **"Create repository"**.

- ¿Cómo crear una rama en Git?

`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?

`git commit -m "Mensaje del commit"`

- ¿Cómo enviar un commit a GitHub?

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

- ¿Qué es un repositorio remoto?

Un **repositorio remoto** es una versión del repositorio que está alojada en un servidor, como GitHub. Permite colaborar con otros usuarios y almacenar el código en la nube. Se accede a través de Internet y se puede sincronizar con tu repositorio local.

- ¿Cómo agregar un repositorio remoto a Git?

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

- ¿Cómo empujar cambios a un repositorio remoto?

`git push -u origin master`

- ¿Cómo tirar de cambios de un repositorio remoto?

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

- ¿Qué es un fork de repositorio?

Un **fork** es una copia de un repositorio en tu cuenta de GitHub. Te permite modificarlo sin afectar el original y contribuir mediante **pull requests**. Se usa para colaborar en proyectos de terceros.

- ¿Cómo crear un fork de un repositorio?

Para crear un **fork** de un repositorio en GitHub

1) Ve al repositorio que quieres forkar.

2) Haz clic en el botón **"Fork"** (arriba a la derecha).

3) Selecciona tu cuenta o una organización.

4) GitHub creará una copia del repositorio en tu cuenta.

5) Ahora puedes clonar tu fork y hacer cambios sin afectar el original.

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

1) **Sube tus cambios** a tu fork:

2) Ve al **repositorio original** en GitHub.

3) Haz clic en **"Pull Requests" > "New pull request"**.

4) Selecciona tu **rama con los cambios**.

5) Escribe un **título y descripción**.

6) Haz clic en **"Create pull request"**.

- ¿Cómo aceptar una solicitud de extracción?

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

1) Ve al repositorio en GitHub.

2) Haz clic en la pestaña **"Pull Requests"**.

3) Selecciona la **pull request** que quieres aceptar.

4) Revisa los cambios y comentarios.

5) Haz clic en **"Merge pull request"**.

6) Confirma con **"Confirm merge"**.

- ¿Qué es un etiqueta en Git?

Una **etiqueta (tag)** en Git es un marcador usado para señalar versiones específicas de un proyecto, como lanzamientos o hitos importantes. Se usa comúnmente para versiones estables.

- ¿Cómo crear una etiqueta en Git?

Para crear una **etiqueta (tag)** en Git:

1) **Etiqueta ligera** (simple marcador): `git tag nombre-de-la-etiqueta`

2) **Etiqueta anotada** (con mensaje y metadatos): `git tag -a nombre-de-la-etiqueta -m "Descripción de la etiqueta"`

- ¿Cómo enviar una etiqueta a GitHub?

Enviar la etiqueta al repositorio remoto: `git push origin nombre-de-la-etiqueta`

- ¿Qué es un historial de Git?

El **historial en Git** es el registro de todos los commits realizados en un repositorio. Permite ver los cambios, autores y fechas de cada modificación.

- ¿Cómo ver el historial de Git?

Para ver el historial, usa: `git log`

- ¿Cómo buscar en el historial de Git?

Para buscar en el historial de Git, usa estos comandos:

Buscar por mensaje de commit: `git log --grep="palabra clave"`

Buscar por autor: `git log --author="nombre o email"`

Buscar por cambios en un archivo: `git log -- nombre-del-archivo`

Buscar por una palabra dentro del código: `git log -S "texto a buscar"`

- ¿Cómo borrar el historial de Git?

Borrar el historial localmente (manteniendo los archivos):

```
rm -rf .git
```

```
git init
```

```
git add .
```

```
git commit -m "Reiniciando historial"
```

- ¿Qué es un repositorio privado en GitHub?

Un **repositorio privado** en GitHub es un repositorio cuyo contenido está restringido solo a ciertos usuarios o colaboradores que tengan acceso explícito. Esto significa que solo las personas que hayan sido invitadas o autorizadas pueden ver, clonar, modificar o contribuir al código almacenado en ese repositorio.

- ¿Cómo crear un repositorio privado en GitHub?

1) **Inicia sesión** en GitHub.

2) Haz clic en el icono de **"+"** en la esquina superior derecha y selecciona **"New repository"**.

3) Escribe el **nombre** del repositorio y, si quieres, agrega una **descripción**.

4) En la opción **"Visibility"**, selecciona **"Private"**.

5) Haz clic en **"Create repository"**.

6) Para invitar colaboradores, ve a **Settings > Manage access > Invite a collaborator**.

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

Para invitar colaboradores, ve a **Settings > Manage access > Invite a collaborator**

- ¿Qué es un repositorio público en GitHub?

Un **repositorio público** en GitHub es un repositorio cuyo contenido es accesible para cualquier persona en internet. Esto significa que cualquier persona puede ver, clonar, forkar, y contribuir al proyecto, siempre y cuando se sigan las reglas de colaboración definidas por el propietario del repositorio.

- ¿Cómo crear un repositorio público en GitHub?

1) **Inicia sesión** en GitHub.

2) Haz clic en el icono de **"+"** en la esquina superior derecha y selecciona **"New repository"**.

3) Escribe el **nombre** del repositorio y, si quieres, agrega una **descripción**.

4) En la opción **"Visibility"**, selecciona **"Public"**.

5) Haz clic en **"Create repository"**.

- ¿Cómo compartir un repositorio público en GitHub?

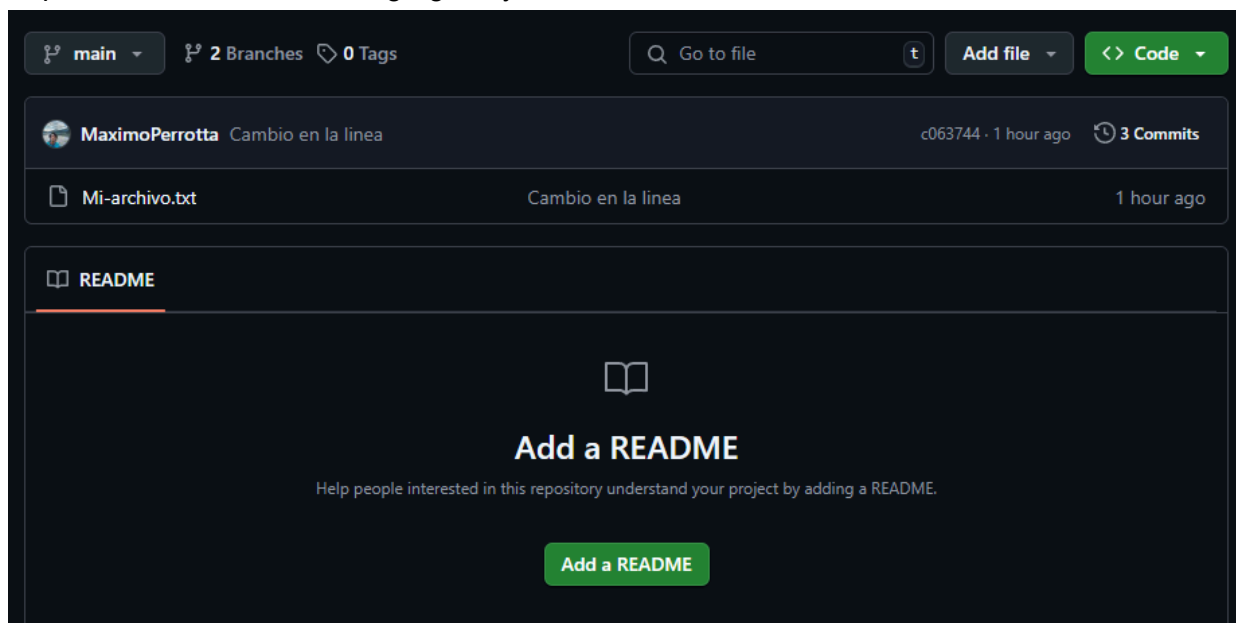
1) Abre el **repositorio público** en GitHub.

2) Copia la **URL** que aparece en la barra de direcciones de tu navegador (por ejemplo,

3) Comparte esa **URL** con quien quieras, ya sea por correo, redes sociales, etc.

Actividad 2:

Repositorio creado, archivo agregado y rama creada.



Actividad 3:

Repositorio creado, clonado, rama creada, merge con conflicto generado y solucionado y todo subido y verificado en github

The screenshot shows the GitHub interface for the repository 'conflict-exercie'. A 'Switch branches/tags' modal is open on the left, displaying a search bar and a list of branches: 'main' (marked as default) and 'feature-branch'. Below the modal, the commit history is visible, showing a commit titled 'segundo cambio'. The top navigation bar includes options to 'Pin' or 'Unwatch' the repository. The main content area shows the commit history with a search bar and buttons for 'Add file' and 'Code'.

This screenshot displays the commit history for the 'main' branch. The commits are listed in chronological order, with the most recent at the top. The first commit is titled 'Conflicto resuelto' and was committed 7 minutes ago. The second and third commits are titled 'Línea añadida en feature-branch' and were committed 11 and 12 minutes ago, respectively. The final commit is titled 'Initial commit' and was authored 23 minutes ago. Each commit entry includes the commit hash, a link to the commit, and a link to the code.

The screenshot shows the file diff view for the 'main' branch. The file 'README.md' is selected, and the diff shows changes between the current commit and the previous one. The diff indicates that 2 lines were added and 1 line was removed. The changes are highlighted in green (additions) and red (deletions). The diff shows the following changes:

```
@@ -1,2 +1,3 @@
1 1 # conflict-exercie
2 - hola cambio
2 2 + segundo cambio
3 3 +
```

Maximo Perrotta.