

- ¿Qué es GitHub?

es una plataforma en línea, que permite a los desarrolladores alojar y compartir proyectos de software, utilizando el sistema de control de versiones Git.

- ¿Cómo crear un repositorio en GitHub?

Creación de un repositorio a partir de la interfaz de usuario web

1. En la esquina superior derecha de cualquier página, selecciona y luego haz clic en **Nuevo repositorio**.
2. Otra opción para crear un repositorio con la estructura de directorios y los archivos de un repositorio existente es seleccionar el menú desplegable Escoger una plantilla y hacer clic un repositorio de plantilla. Verás repositorios de plantillas que te pertenecen a ti y a las organizaciones de las que eres miembro o bien repositorios de plantillas que has usado anteriormente. Para más información, consulta Crear un repositorio desde una plantilla.
3. Si decidió elegir una plantilla, también puede incluir la estructura de directorios y los archivos de todas las ramas en la plantilla, y no únicamente aquellos de la rama predeterminada, seleccione Include all branches.
4. Usa el menú desplegable Propietario para seleccionar la cuenta que quieres que sea propietaria del repositorio.
5. Teclea el nombre de tu repositorio, y una descripción opcional.
6. Elige la visibilidad del repositorio. Para más información, consulta Acerca de los repositorios.
7. Si no estás utilizando una plantilla, hay varios elementos opcionales que puedes pre-cargar en tu repositorio. Si estás importando un repositorio existente a GitHub, no elijas ninguna de estas opciones, ya que producirás un conflicto de fusión. Puedes agregar o crear nuevos archivos usando la interfaz de usuario o elegir agregar nuevos archivos usando luego la línea de comando. Para más información, consulta Importación de un repositorio de Git externo mediante la línea de comandos, Agregar un archivo a un repositorio y Cómo abordar los conflictos de combinación.

Puedes crear un README, que es un documento que describe tu proyecto. Para más información, consulta Acerca de los archivos README.

Puede crear un archivo .gitignore, que es un conjunto de reglas de omisión. Para más información, consulta Ignorar archivos.

Puedes elegir agregar una licencia de software a tu proyecto. Para obtener más información, consulta [Generar licencia para un repositorio](#).

8. Opcionalmente, si la cuenta personal o la organización en la que va a crear usa cualquier GitHub Apps de GitHub Marketplace, seleccione las aplicaciones que le gustaría usar en el repositorio.
9. Haga clic en [Create repository](#) (Crear repositorio).
10. En la parte inferior de la página de Configuración rápida resultante, en "Importar el código del repositorio anterior", puedes elegir importar un proyecto en tu nuevo repositorio. Para ello, haga clic en [Import code](#).

- ¿Cómo crear una rama en Git?

Puedes usar el comando `git branch` seguido del nombre de la rama. Por ejemplo, para crear una rama llamada `new_branch`, puedes usar `git branch new_branch`.

- ¿Cómo cambiar a una rama en Git?

puedes usar el comando `git checkout`. Por ejemplo, para cambiar a la rama `master` desde la rama `mi-función`, puedes usar `git checkout master`.

- ¿Cómo fusionar ramas en Git?

Colocarte en la rama a la que quieres fusionar la rama.

Usar el comando `git merge` seguido del nombre de la rama que quieres fusionar.

Por ejemplo, para crear un nuevo commit en la rama `master` con los cambios de la rama `cabecera`, usa este comando:

Código

```
git checkout master  
git merge cabecera
```

- ¿Cómo crear un commit en Git?

Seleccionar los cambios con `git add`

Usa `git add` para preparar los cambios que quieres incluir en la confirmación.

Puedes usar `git add -p` para separar los cambios, incluso si están en el mismo archivo.

Crear la confirmación con `git commit`

Usa git commit para crear una instantánea de los cambios preparados.

Puedes añadir un mensaje de confirmación usando el indicador -m. Por ejemplo, git commit -m "Add an anchor for the trial end sectionnnn."

- ¿Cómo enviar un commit a GitHub?

Hacer commit a los cambios

Ejecuta git add . para rastrear los cambios en tu carpeta

Ejecuta git commit -m "mensaje" para preparar los cambios para enviarlos a GitHub

El mensaje debe ser una breve descripción de los cambios

Enviar los cambios a GitHub

Ejecuta git push

Especifica el nombre remoto y el nombre de rama

Por ejemplo, git push origin main

Opcional: Subir un proyecto a GitHub usando un navegador

En la página principal del repositorio, selecciona el menú desplegable "Cargar archivos"

Haz clic en "Agregar archivos"

Arrastra y suelta los archivos en el explorador

En "Confirmar cambios", selecciona "Confirmar directamente en la rama principal"

Haz clic en "Confirmar cambios"

- ¿Qué es un repositorio remoto?

Es una copia de un proyecto que se encuentra en un servidor remoto, ya sea en internet o en una red. Los repositorios remotos se comparten entre varios usuarios de un equipo.

- ¿Cómo agregar un repositorio remoto a Git?

Estar en el directorio donde está el repositorio

Usar el comando git remote add en el terminal

Especificar un nombre remoto, por ejemplo, origin

También puedes clonar un repositorio remoto usando el comando git clone.
Para ello, necesitas la URL del repositorio.

Uso de git clone

Usar el comando git clone

Especificar la URL del repositorio

Se extraerá la última versión de los archivos del repositorio remoto en la rama principal

Se añadirá a una nueva carpeta con el mismo nombre que el repositorio

Una vez que has iniciado o clonado un repositorio, puedes:

Realizar commits en la versión del archivo usando los comandos git add y git commit

Enviar los cambios al repositorio remoto con git push

Extraer y descargar contenido desde un repositorio remoto con git pull

•¿Cómo empujar cambios a un repositorio remoto?

Para empujar cambios a un repositorio remoto, puedes usar el comando git push. Antes de hacerlo, debes confirmar los cambios en el repositorio local.

Sintaxis

git push <nombre del repositorio> <nombre de la rama>

•¿Cómo tirar de cambios de un repositorio remoto?

1. Ejecuta git pull
2. Git ejecutará git fetch en la rama local a la que apunta HEAD
3. Git descargará el contenido
4. Git entrará en un flujo de trabajo de fusión
5. Se creará una nueva confirmación de fusión
6. HEAD se actualizará para que apunte a la nueva confirmación

Git pull es una combinación de los comandos git fetch y git merge.

Otros comandos relacionados con repositorios remotos

- git remote rm Elimina la dirección URL de un repositorio remoto
- git remote rename Cambia el nombre de una conexión remota
- git remote set-url Cambia la URL de origen remota

•¿Qué es un fork de repositorio?

Un fork de repositorio es una copia exacta de un repositorio que se crea para colaborar en un proyecto. La palabra fork en inglés significa "bifurcación".

¿Para qué sirve un fork?

- Permite contribuir a un proyecto sin tener permisos de escritura
- Permite experimentar con nuevas características
- Permite colaborar con otros desarrolladores
- Permite contribuir a proyectos de código abierto
- Permite realizar cambios en el proyecto sin afectar el repositorio original

•¿Cómo crear un fork de un repositorio?

Ir a la página del repositorio que se desea bifurcar

Pulsar el botón "Fork" que se encuentra en la parte superior derecha de la página.

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

1. Realizar los cambios en una rama separada o en un repositorio bifurcado
2. Ir al repositorio en el sitio web de tu proveedor de alojamiento
3. Seleccionar la rama que contiene los cambios
4. Hacer clic en el botón "Solicitar extracción"
5. Seleccionar la rama en la que quieres combinar los cambios
6. Escribir un título y una descripción para tu solicitud de extracción
7. Hacer clic en el botón "Enviar solicitud de extracción"

•¿Cómo aceptar una solicitud de extracción?

En github

1. Revisa los cambios en la pestaña "Archivos modificados"
2. Haz clic en el botón "Revisar cambios"
3. Selecciona "Aprobar" en las opciones
4. Envía tu revisión

- ¿Qué es una etiqueta en Git?

Una etiqueta es una referencia que marca un punto específico en el historial de un repositorio. Se pueden usar para indicar versiones, cambios importantes o cualquier otro evento que sea relevante para el equipo de desarrollo.

- ¿Cómo crear una etiqueta en Git?

Para crear una etiqueta en Git, puedes usar el comando `git tag` seguido del nombre de la etiqueta. Por ejemplo, para crear una etiqueta llamada "v1.4", puedes usar `git tag v1.4`.

- ¿Cómo enviar una etiqueta a GitHub?

Para enviar etiquetas a GitHub, puedes usar la opción `--tags` en el comando `git push`. También puedes crear etiquetas desde la interfaz web de GitHub.

Desde la línea de comandos

1. Usa `git tag` para crear una etiqueta
2. Usa `git push --tags` para enviar las etiquetas al repositorio remoto

- ¿Qué es un historial de Git?

Es un registro de los cambios realizados en un repositorio, almacenado como un gráfico de confirmaciones.

- ¿Cómo ver el historial de Git?

puedes usar el comando `git log`. También puedes ver el historial de Git en la interfaz de usuario de GitLab o GitHub Desktop.

• ¿Cómo buscar en el historial de Git?

puedes usar los comandos `git log` y `git grep`. También puedes usar la interfaz de usuario de GitLab.

Comandos para buscar en el historial de Git

- `git log --[filename]` Muestra el historial de commits sobre un archivo
- `git log -L :[function]:[file]` Muestra el historial de una función o línea de código
- `git grep <pattern> $(git rev-list --all)` Busca un patrón en todo el historial de confirmaciones
- `git log -p -G <pattern>` Busca confirmaciones que introduzcan o eliminen un patrón
- `git log -p -S <string>` Busca confirmaciones que agreguen o eliminen una cadena específica
- `git log --grep=<pattern>` Busca un patrón en los mensajes de confirmación

¿Cómo borrar el historial de Git?

Se puede eliminar la carpeta `.git`, o bien se puede reescribir la historia de tu repositorio para eliminar un archivo o datos confidenciales.

• ¿Qué es un repositorio privado en GitHub?

Es un espacio de almacenamiento que solo es accesible para ti y las personas a las que le des permiso.

• ¿Cómo crear un repositorio privado en GitHub?

1. Ir a la esquina superior derecha de cualquier página de GitHub
2. Seleccionar Nuevo repositorio
3. Escribir un nombre para el repositorio
4. Añadir una descripción, si quieres
5. Elegir la visibilidad del repositorio, en este caso privado
6. Seleccionar Crear repositorio

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

Seguir estos pasos:

1. Ir al repositorio
2. Hacer clic en Configuración
3. En la barra lateral, hacer clic en Colaboradores y equipos
4. Hacer clic en Agregar personas
5. Buscar al usuario que se quiere invitar
6. Seleccionar el rol para el colaborador
7. Hacer clic en Invitar

- ¿Qué es un repositorio público en GitHub?

Es un espacio virtual donde se puede almacenar, organizar y compartir código, archivos y documentación. Los repositorios públicos son accesibles para cualquier persona con conexión a internet.

- ¿Cómo crear un repositorio público en GitHub?

1. Ir a [GitHub](#)
2. En la esquina superior derecha, seleccionar Nuevo repositorio
3. Escribir un nombre para el repositorio
4. Añadir una descripción opcional
5. Elegir la visibilidad del repositorio, en este caso pública
6. Seleccionar Inicializar este repositorio con un archivo Léame
7. Hacer clic en Crear repositorio

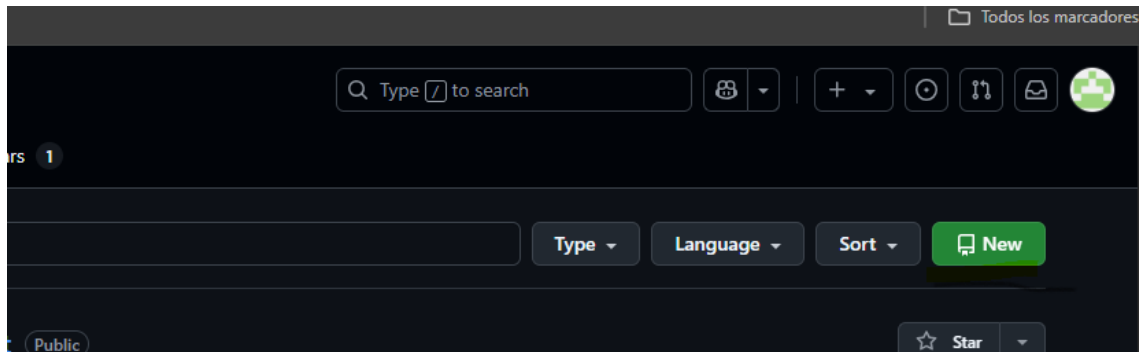
- ¿Cómo compartir un repositorio público en GitHub?

Invitar colaboradores

1. Ve a la página principal del repositorio
2. Haz clic en "Configuración"
3. En la barra lateral, haz clic en "Colaboradores y equipos"
4. Haz clic en "Agregar personas" o "Agregar equipos"

5. Escribe el nombre de la persona o el equipo a invitar
6. Selecciona el rol de repositorio que deseas otorgar
7. Haz clic en "Agregar NOMBRE AL REPOSITORIO"

2) Crear un repositorio




Ingresar nombre al repositorio

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Required fields are marked with an asterisk ().*

Owner * **Repository name ***

 lauoisin / primer-repo-gi

✔ primer-repo-gi is available.

Great repository names are short and memorable. Need inspiration? How about **cautious-carnival** ?

Description (optional)

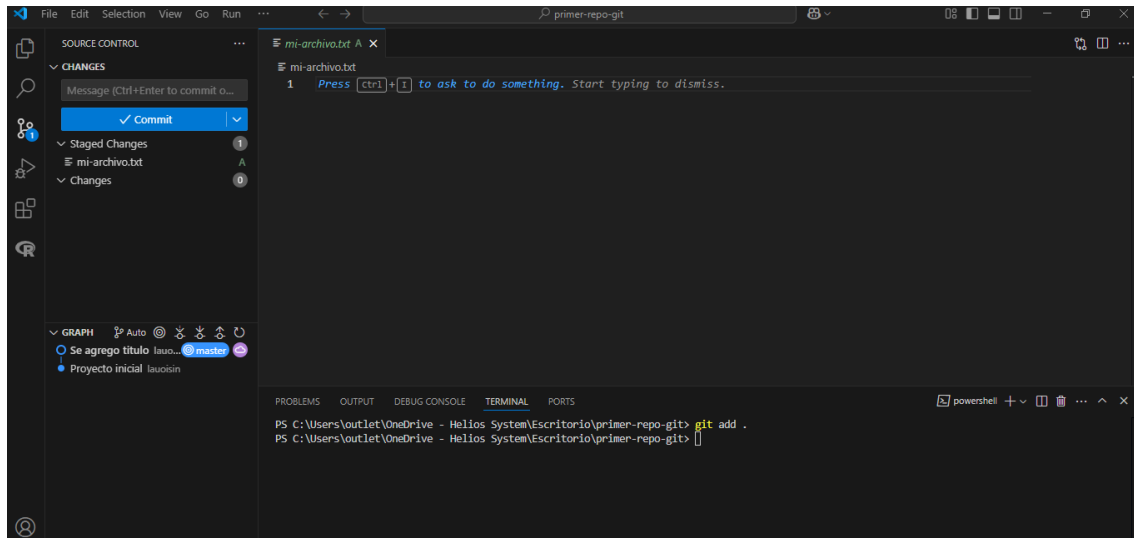
☒ **Public**
Anyone on the internet can see this repository. You choose who can commit.

☐ **Private**
You choose who can see and commit to this repository.

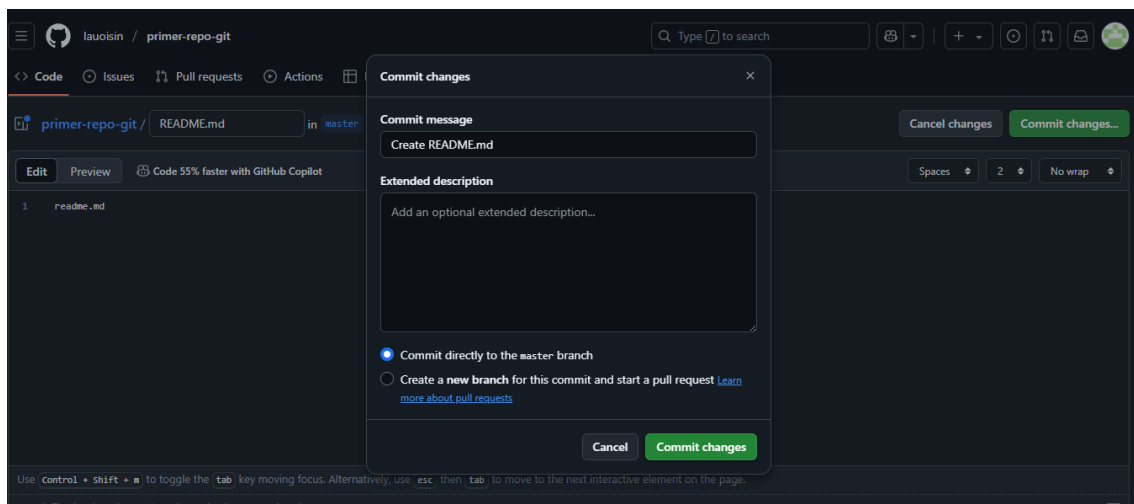
Initialize this repository with:

☐ Add a README file

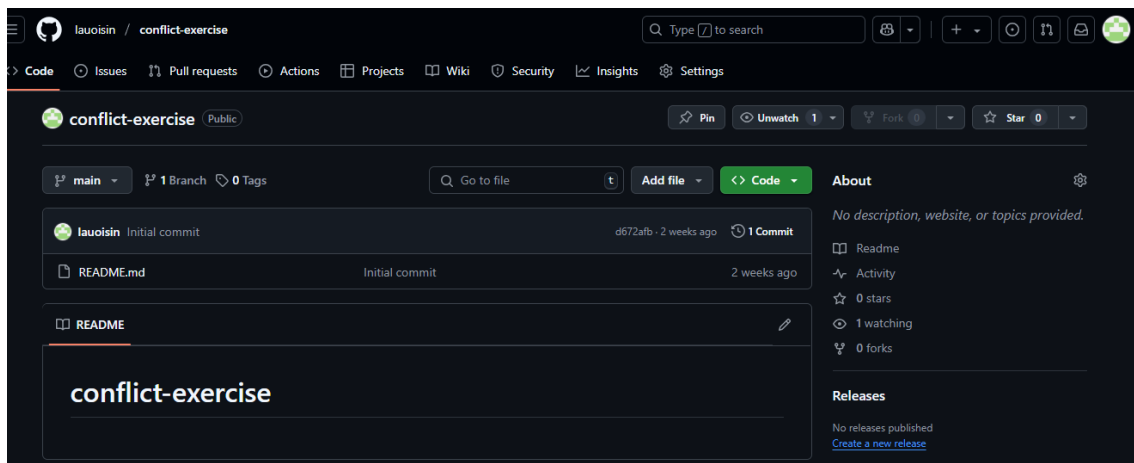
Agregar archivo mi-archivo.txt



Agregar readme.md



3) Paso 1: Crear un repositorio en GitHub con Nombre conflict-exercise



Paso 2: Clonar el repositorio a máquina local

```
PS C:\Users\outlet\OneDrive - Helios System\Escritorio\primer-repo-git> git clone https://github.com/lauoisin/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.
PS C:\Users\outlet\OneDrive - Helios System\Escritorio\primer-repo-git>
```

- Entra en el directorio del repositorio:

```
PS C:\Users\outlet\OneDrive - Helios System\Escritorio\primer-repo-git> cd conflict-exercise
PS C:\Users\outlet\OneDrive - Helios System\Escritorio\primer-repo-git\conflict-exercise>
```

git checkout -b feature-branch

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS
PS C:\Users\outlet\OneDrive - Helios System\Escritorio\primer-repo-git\conflict-exercise> git checkout -b feature-branch
Switched to a new branch 'feature-branch'
PS C:\Users\outlet\OneDrive - Helios System\Escritorio\primer-repo-git\conflict-exercise>
```

- Abre el archivo README.md en un editor de texto y añade una línea nueva y commitear los cambios.

```
PS C:\Users\outlet\OneDrive - Helios System\Escritorio\conflict-exercise\conflict-exercise> git add README.md
PS C:\Users\outlet\OneDrive - Helios System\Escritorio\conflict-exercise\conflict-exercise>
```

```
PS C:\Users\outlet\OneDrive - Helios System\Escritorio\conflict-exercise\conflict-exercise> git commit -m "Added a line in feature-branch"
[feature-branch b48614e] Added a line in feature-branch
1 file changed, 3 insertions(+), 1 deletion(-)
PS C:\Users\outlet\OneDrive - Helios System\Escritorio\conflict-exercise\conflict-exercise>
```

Paso 4: Volver a la rama principal y editar el mismo archivo (main)

```
PS C:\Users\outlet\OneDrive - Helios System\Escritorio\conflict-exercise\conflict-exercise> git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.
PS C:\Users\outlet\OneDrive - Helios System\Escritorio\conflict-exercise\conflict-exercise> █
```

Edita el archivo README.md de nuevo, añadiendo una línea diferente:

- Guarda los cambios y haz un commit:

```
PS C:\Users\outlet\OneDrive - Helios System\Escritorio\conflict-exercise\conflict-exercise> git add README.md
PS C:\Users\outlet\OneDrive - Helios System\Escritorio\conflict-exercise\conflict-exercise> █
```

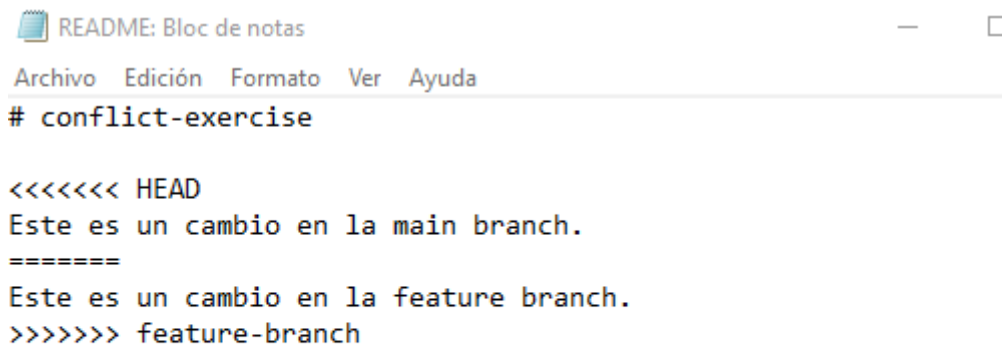
```
PS C:\Users\outlet\OneDrive - Helios System\Escritorio\conflict-exercise\conflict-exercise> git commit -m "Added a line in main branch"
[main ec544c5] Added a line in main branch
1 file changed, 3 insertions(+), 1 deletion(-)
PS C:\Users\outlet\OneDrive - Helios System\Escritorio\conflict-exercise\conflict-exercise> █
```

Paso 5: Hacer un merge y generar un conflicto

```
PS C:\Users\outlet\OneDrive - Helios System\Escritorio\conflict-exercise\conflict-exercise> 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.
PS C:\Users\outlet\OneDrive - Helios System\Escritorio\conflict-exercise\conflict-exercise> █
```

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

```

Decidir como resolver el conflicto

```
PS C:\Users\outlet\OneDrive - Helios System\Escritorio\conflict-exercise\conflict-exercise> git add README.md
PS C:\Users\outlet\OneDrive - Helios System\Escritorio\conflict-exercise\conflict-exercise> █
```

```
PS C:\Users\outlet\OneDrive - Helios System\Escritorio\conflict-exercise\conflict-exercise> git commit -m "Resolved merge conflict"
[main 548d0b5] Resolved merge conflict
PS C:\Users\outlet\OneDrive - Helios System\Escritorio\conflict-exercise\conflict-exercise> █
```

Paso 7: Subir los cambios a GitHub

```
PS C:\Users\outlet\OneDrive - Helios System\Escritorio\conflict-exercise\conflict-exercise> git push origin main
Enumerating objects: 11, done.
Counting objects: 100% (11/11), done.
Delta compression using up to 4 threads
Compressing objects: 100% (6/6), done.
Writing objects: 100% (9/9), 850 bytes | 283.00 KiB/s, done.
Total 9 (delta 1), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (1/1), done.
To https://github.com/lauoisin/conflict-exercise.git
d672afb..548d0b5  main -> main
PS C:\Users\outlet\OneDrive - Helios System\Escritorio\conflict-exercise\conflict-exercise> 
```

- También sube la feature-branch si deseas:

```
PS C:\Users\outlet\OneDrive - Helios System\Escritorio\conflict-exercise\conflict-exercise> 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/lauoisin/conflict-exercise/pull/new/feature-branch
remote:
To https://github.com/lauoisin/conflict-exercise.git
* [new branch]   feature-branch -> feature-branch
PS C:\Users\outlet\OneDrive - Helios System\Escritorio\conflict-exercise\conflict-exercise> 
```

Paso 8: Verificar en GitHub

The screenshot shows the GitHub 'Commits' page for the 'main' branch. The page is dark-themed and displays a list of commits. At the top, there are filters for the branch ('main') and the user ('lauoisin'). Below the filters, the commits are listed in chronological order, with the most recent at the top. The commits are grouped by date: 'Apr 13, 2025' and 'Mar 28, 2025'. Each commit entry includes a title, a description, the commit hash, and a link to the commit details. The 'Resolved merge conflict' commit is highlighted with a green background. The 'Initial commit' is marked as 'Verified'.

Commit Title	Commit Hash	Author	Time Ago
Resolved merge conflict	548d0b5	lauoisin	5 minutes ago
Added a line in main branch	ec544c5	lauoisin	10 minutes ago
Added a line in feature-branch	b48614e	lauoisin	16 minutes ago
Initial commit	d672afb	lauoisin	2 weeks ago

Commit 548d0b5

lauoisin committed 4 minutes ago

Resolved merge conflict

main

2 parents ec544c5 - b48614e commit 548d0b5

Filter files...

README.md

1 file changed +5 -1 lines changed

Search within code

README.md

@@ -1,3 +1,7 @@

1 1 # conflict-exercise

2 2

3 - Este es un cambio en la main branch.

3 + <<<<<< HEAD

4 + Este es un cambio en la main branch.

5 + =====

6 + Este es un cambio en la feature branch.

7 + >>>>>> feature-branch

Comments 0

Lock conversation

Alumna: Laura Mendez