

Tu primer Pull Request en GitHub

Vas a clonar un repositorio real, crear tu propia rama, hacer un cambio, y abrir un Pull Request — el flujo que se usa todos los días en cualquier equipo de desarrollo.

QUÉ VAS A HACER

- Clonar el repositorio del curso en tu computadora
- Crear una rama con tu nombre para trabajar sin tocar el código de nadie más
- Crear un archivo con tu presentación
- Commitear y pushear tu rama a GitHub
- Abrir un Pull Request para que tu cambio sea revisado y aceptado

ANTES DE EMPEZAR — VERIFICÁ QUE TENÉS ESTO

- ☐ Git instalado en tu computadora (`git --version` en la terminal no debe dar error)
- ☐ Una cuenta creada en **github.com** y haber iniciado sesión
- ☐ Una terminal abierta (PowerShell en Windows, Terminal en Mac/Linux)

PARTE 1 — CONFIGURACIÓN Y CLONADO

1

Configurá tu identidad en Git

La primera vez que usás Git en una computadora, tenés que decirle cómo llamarte. Este nombre y email van a aparecer en todos tus commits.

```
terminal
$ git config --global user.name "Tu Nombre"
$ git config --global user.email "tu@email.com"
# Verificá que quedó guardado
$ git config --global user.name
Tu Nombre
```

Reemplazá con tu nombre real y el email con el que te registraste en GitHub. Esto se hace una sola vez por computadora.

2

Cloná el repositorio del curso

Este comando descarga el repositorio completo a tu computadora — los archivos, el historial de commits, y la conexión a GitHub. Todo en un solo paso.

```
terminal
$ git clone https://github.com/micadevops/camino-devops-en-aws.git
Cloning into 'camino-devops-en-aws'...
remote: Counting objects: ..., done.
```

Receiving objects: 100% (...), done.

- ✓ Se creó una carpeta llamada **camino-devops-en-aws** en el lugar donde estaba tu terminal.

3 Entrá a la carpeta del proyecto

Ahora tenés que moverte adentro de esa carpeta. Desde acá van a correr todos los comandos de Git que siguen.

```
terminal

$ cd camino-devops-en-aws
# Listá los archivos para ver qué hay adentro
$ ls
  README.md  sesion2/  ...
```

4 Mirá en qué rama estás y el historial

Acabás de clonar el repo. Estás en la rama principal (**main**). Veamos también los commits que ya existen.

```
terminal

# Ver en qué rama estás (* = rama actual)
$ git branch
* main

# Ver el historial de commits
$ git log --oneline
a3f2b9c (HEAD → main) ...
b1c2d3e ...
```

*Siempre que clones o abris un repo, lo primero es saber dónde estás. **git branch** te lo dice al instante.*

PARTE 2 — TU RAMA Y TU CAMBIO

5 Creá tu propia rama

Vas a trabajar en tu propia rama para no tocar el código de nadie más. Nombrá tu rama con tu nombre en minúscula, sin espacios. Ejemplos: **mica/mi-presentacion**, **sofia/mi-presentacion**.

```
terminal

# Creá la rama y cambiáte a ella — todo en uno
$ git checkout -b tu-nombre/mi-presentacion
Switched to a new branch 'tu-nombre/mi-presentacion'

# Confirmá que estás en la nueva rama
$ git branch
  main
```

* `tu-nombre/mi-presentacion` ← acá estás ahora

✓ El * al lado de tu rama confirma que ya estás trabajando ahí. Main sigue intacto.

6 Creá tu archivo de presentación

Creá una carpeta llamada `participantes` (si no existe) y dentro de ella un archivo de texto con tu nombre. Abrilo con cualquier editor de texto (VS Code, el Bloc de Notas, nano) y escribí algo sobre vos.

```
terminal — opción rápida desde la terminal

# Crear la carpeta participantes
$ mkdir participantes

# Crear tu archivo (reemplazá "tu-nombre" con tu nombre real)
$ echo "Hola! Soy Tu Nombre. Estoy aprendiendo DevOps en AWS Girls UG UY."
> participantes/tu-nombre.txt
```

Si preferís usar VS Code: abrí la carpeta `camino-devops-en-aws` en VS Code, creá la carpeta `participantes` y adentro creá un archivo `tu-nombre.txt`. Escribí lo que quieras y guardalo.

PARTE 3 — COMMIT Y PUSH

7 Verificá el estado — git status

Antes de commitear, siempre verificás qué cambió. Git te muestra el archivo nuevo como "Untracked" porque todavía no se lo mostraste.

```
terminal

$ git status
Untracked files:
  participantes/tu-nombre.txt
nothing added to commit but untracked files present
```

Rojo = Git lo ve pero no lo rastrea. Todavía no le dijiste que lo incluya.

8 Agregá el archivo al staging — git add

Le decís a Git: "este archivo quiero incluir en el próximo commit". Con el punto (`.`) se agregan todos los archivos nuevos o modificados de una vez.

```
terminal

$ git add .

# Verificá que ahora está en staging
$ git status
Changes to be committed:
```

new file: participantes/*tu-nombre*.txt

✓ Verde = está en el staging area. Listo para ser commiteado.

9

Creá el commit — git commit

Guardás la foto del momento con un mensaje descriptivo. Este commit queda guardado en el historial local de tu rama.

```
terminal

$ git commit -m "feat: agrego presentación de tu-nombre"
[tu-nombre/mi-presentacion a1b2c3d] feat: agrego presentación de tu-nombre
1 file changed, 1 insertion(+)

# Verificá que el commit aparece en el historial
$ git log --oneline
a1b2c3d (HEAD → tu-nombre/mi-presentacion) feat: agrego presentación de ...
... (commits anteriores de main)
```

✓ Tu commit aparece como el más nuevo. Está guardado en tu máquina — falta subirlo a GitHub.

10

Subí tu rama a GitHub — git push

Como es la primera vez que subís esta rama, necesitás usar `-u origin` para que Git sepa a dónde enviarla. La próxima vez con `git push` solo alcanza.

```
terminal

$ git push -u origin tu-nombre/mi-presentacion
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
To https://github.com/micadevops/camino-devops-en-aws.git
* [new branch] tu-nombre/mi-presentacion → tu-nombre/mi-presentacion
```

✓ Tu rama y tu commit están en GitHub. Ahora cualquiera puede verlos.

PARTE 4 — ABRIR TU PULL REQUEST

11

Abrí el Pull Request en GitHub

Después de pushear, GitHub casi siempre muestra un banner amarillo con "Compare & pull request". Hacé click ahí. Si no aparece, seguí estos pasos:

- Andá a **github.com/micadevops/camino-devops-en-aws**
- Hacé click en la pestaña **Pull requests**
- Hacé click en el botón verde **New pull request**
- En "compare", seleccioná tu rama (`tu-nombre/mi-presentacion`)
- Hacé click en **Create pull request**

¿Qué escribir en el PR?

Título: feat: agrego presentación de [tu nombre]

Descripción: "Agrego mi archivo de presentación en la carpeta participantes/. Soy [tu nombre] y estoy aprendiendo DevOps con AWS Girls UG UY."



¡Terminaste! Tu Pull Request está abierto en GitHub. Micaela va a poder verlo, comentarlo, y cuando lo apruebe, tu cambio va a quedar en main para siempre.

BONUS — EXPLORÁ EL REPOSITORIO



Ver los PRs de tus compañeras

En la pestaña "Pull requests" de GitHub vas a ver todos los PRs abiertos — los de tus compañeras también. Podés leer el código que cambiaron, hacer comentarios, o simplemente explorar. Eso es exactamente lo que hacen los equipos reales todos los días.



Ver el historial completo del repo

En GitHub, hacé click en el número de commits que aparece arriba de los archivos (algo como "23 commits"). Vas a ver el historial completo — quién hizo cada cambio, cuándo, y qué cambió exactamente. Hacé click en cualquier commit para ver el diff (verde = líneas agregadas, rojo = borradas).



Ver tus ramas en GitHub

En el repo, hacé click en el selector de rama (dice "main") y vas a ver todas las ramas que subieron las participantes. Cada una trabajó en su propia línea del tiempo sin pisarse con las demás.

terminal — también podés verlas desde acá

```
# Bajá info de todas las ramas remotas
$ git fetch --all
$ git branch -r
  origin/main
  origin/tu-nombre/mi-presentacion
  origin/mica/mi-presentacion
  origin/sofia/mi-presentacion ...
```

Resumen de lo que hiciste hoy:

git clone → git branch → git checkout -b → (editar archivo) → git add → git commit → git push → abrir PR en GitHub.

Este es el flujo de trabajo de cualquier desarrolladora en cualquier empresa del mundo. Lo acabás de hacer.

