Guía Básica para Usar GitHub desde la Terminal

---

1. Iniciar Sesión en la Terminal

Para autenticarte en GitHub desde la terminal, puedes usar HTTPS:

Usando HTTPS (con token de acceso personal - PAT)

git config --global user.name "TuNombreDeUsuario"

git config --global user.email "tuemail@example.com"

Si necesitas autenticarte en GitHub, puedes usar:

git credential reject https://github.com

git credential approve https://github.com

Cuando realices git pull o git push, te pedirá tus credenciales.

---

2. Crear un Repositorio Local y Enviarlo a GitHub

git init nombre-del-repositorio

cd nombre-del-repositorio

git remote add origin https://github.com/TuUsuario/TuRepositorio.git

git branch -M main

git push -u origin main

Esto inicializa el repositorio, lo vincula con GitHub y sube la rama main.

---

3. Trabajando con Ramas en Git

Crear una Rama Localmente

git checkout -b nombre-de-la-rama

Esto crea y cambia a la nueva rama.

Subir una Rama Local al Repositorio Remoto

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

---

4. Realizar Cambios y Subirlos a GitHub

Agregar Cambios y Realizar un Commit

git add .

git commit -m "Mensaje del commit"

Esto guarda los cambios en la historia del repositorio.

Enviar los Cambios al Repositorio Remoto

git push origin nombre-de-la-rama

---

5. Obtener Cambios del Repositorio Remoto

Actualizar el Repositorio Local con los Cambios Remotos

git pull origin main

Esto sincroniza el repositorio local con la última versión de la rama main.

Descargar y Cambiar a una Rama Remota

git fetch origin

git checkout nombre-de-la-rama-remota

Esto te permite trabajar en una rama que no tenías localmente.

---

6. Fusionar Cambios y Resolver Conflictos

Realizar un Merge

Para fusionar una rama con main:

git checkout main

git merge nombre-de-la-rama

Si hay conflictos, Git los mostrará y deberás resolverlos manualmente en los archivos afectados antes de continuar con:

git add .

git commit -m "Resolviendo conflictos"

Resolver Conflictos en un Merge o Pull Request

Cuando Git detecta conflictos en un merge, abrirá los archivos afectados y marcará las diferencias así:

<<<<<<< HEAD

Código de la rama actual

=======

Código de la rama que se intenta fusionar

>>>>>> nombre-de-la-rama

Para resolverlo, edita el archivo manualmente, eliminando las marcas y manteniendo solo el código correcto. Luego, guarda el archivo y ejecuta:

git add .

git commit -m "Conflicto resuelto en archivo X"

Si el conflicto ocurrió en un **Pull Request en GitHub**, GitHub te permite resolverlo directamente desde la interfaz web.

Hacer un Pull Request en GitHub

1. Sube los cambios con git push origin nombre-de-la-rama.
2. Ve a GitHub y crea un **Pull Request (PR)** desde la rama.
3. Espera la revisión y aprobación antes de fusionar con main.

Revertir un Push (Deshacer Cambios en Remoto)

Si has enviado cambios por error, puedes revertirlos:

git reset --soft HEAD~1 # Deshace el último commit, pero mantiene los cambios en staging

git reset --hard HEAD~1 # Elimina completamente el último commit

Si ya hiciste push, puedes forzar la reversión:

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

---

7. Importancia de las Ramas en Git

Rama main

Es la rama principal y estable del proyecto. Aquí solo deben integrarse cambios ya probados y listos para producción.

Rama develop

Esta rama se usa para la integración de nuevas funcionalidades antes de ser fusionadas con main. Permite pruebas y revisión de cambios.

Ramas feature

Son ramas temporales creadas para desarrollar nuevas funcionalidades o corregir errores. Se crean desde develop y, una vez finalizadas, se fusionan nuevamente en develop.

Con esta guía básica, podrás trabajar con Git y GitHub desde la terminal de manera eficiente. ¡Feliz codificación! 🚀