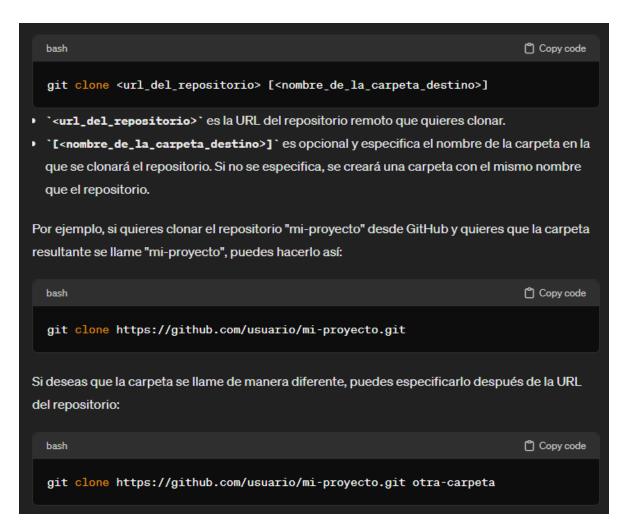


INICIAR LA APLICACIÓN DE GIT EN LA CARPETA PRINCIPAL DEL PROYECTO A TRABAJAR LOS COMANDO BASICOS QUE SE USARAN DE FORMA RECURRENTE SON:

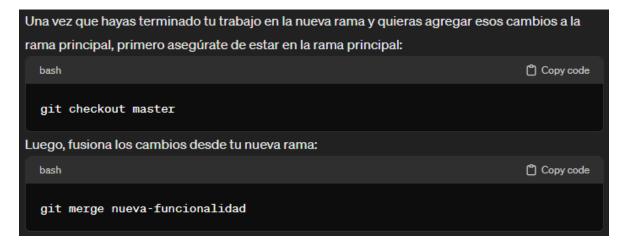
- GIT INIT (SE USA SOLO UNA VEZ, NORMALMENTE ES SOLO CUANDO SE QUIERE SUBIR DOCUMENTOS A UN REPOSITORIO REMOTO; LUEGO DE ESO SE PUEDE OMITIR)
- GIT CLONE (PARA PODER DESCARGAR EL REPOSITORIO REMOTO DE MANERA RAPIDA Y FACIL)



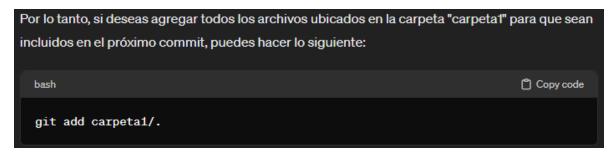
GIT BASH (SIRVE PARA VER LA RAMA EN LA QUE SE ESTA TRABAJANDO). CABE RECALCAR
QUE SE PUEDE CREAR NUEVAS RAMAS PARA TRABAJAR Y REALIZAR CAMBIOS SIN
PREOCUPARSE DE LA RAMA PRINCIPAL (MAIN) DE LA SIGUIENTE MANERA

1. Crear una nueva rama:	
Para crear una nueva rama, puedes usar el comando 'git branch' seguido del nombre de la	
nueva rama. Por ejemplo, si quieres llamar a tu nueva rama "nueva-funcionalidad", ejecutarías:	
bash	Copy code
git branch nueva-funcionalidad	
O puedes hacerlo y cambiar inmediatamente a esa rama utilizando `-b`:	
bash	Copy code
git checkout -b nueva-funcionalidad	

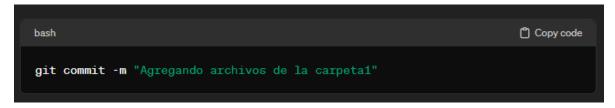
Y PARA PODER JUNTAR O COMBINAR LOS CAMBIOS REALIZADOS DE LA RAMA NUEVA A LA PRINCIPAL SE REALIZA DE LA SIGUEINTE MANERA.



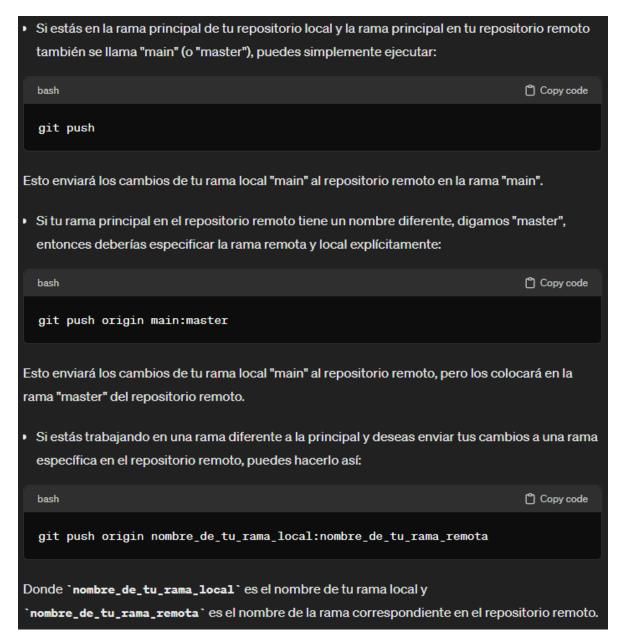
 GIT ADD (SE USA PARA AGREGAR LOS ARCHIVOS QUE QUERAMOS CARGAR AL REPOSITORIO REMOTO, SI TENEMOS VARIOS ARCHIVOS CON CAMBIOS QUE QUEREMOS SUBIR PODEMOS HACERLO DE LA SIGUIENTE MANERA)



 GIT COMMIT (SE UTILIZA PARA CONFIRMAR LOS CAMBIOS REALIZADOS A LOS ARCHIVOS QUE HEMOS AGREGADO CON 'GIT ADD')



 GIT PUSH (SE USA PARA SUBIR LOS ARCHIVOS DEL REPOSITORIO LOCAL AL REPOSITORIO REMOTO)



 GIT PULL (SE USA PARA PODER ACTUALIZAR Y DESCARGAR LOS CAMBIOS SUBIDOS POR LOS DEMAS INTEEGRANTES DE UN REPOSITORIO)

CADA USUARIO O INTEGRANTE DEL REPOSITORIO ESTRARA TRABAJANDO EL REPOSITORIO REMOTO DE FORMA LOCAL, POR LO QUE LOS COMMIT QUE SE HAGAN EN CADA MAQUINA SERAN PERSONALES HASTA QUE SE SUBAN AL REPOSRITORIO REMOTO CON PUSH. SI SE DESEA REGRESAR A UN COMMIT ANTERIOR PARA SEGUIR TRABAJANDO DESDE UNA VERSION ANTERIOR A LOS CAMBIOS QUE SE HAN HECHOS SE PEUDE REALIZAR DE LA SIGUIENTE MANERA:

 GIT LOG (PARA VER EL HISTORIAL DE COMMIT'S QUE SE HAN REALIZADO EN EL REPOSITORIO REMOTO Y LOCAL)

```
commit 5f3a1e2d6e50a4e4a95b7b83161f0b29c1eef3e3
Author: Autor <email@ejemplo.com>
Date: Mié Abr 6 18:02:21 2024 +0200

Mensaje del commit anterior

commit 2fd67b4e9d9911d3b53a4f5b80ee5a52f2b4fb10 (HEAD -> rama_actual)
Author: Autor <email@ejemplo.com>
Date: Mié Abr 6 17:58:11 2024 +0200

Mensaje del commit actual
```

• GIT CHEKOUT (PARA VOLVER O RECUPERAR UN COMMIT ANTERIOR)

Por ejemplo, si el hash del commit es `5f3a1e2d6e50a4e4a95b7b83161f0b29c1eef3d usar solo los primeros caracteres para identificarlo:	<b>≘3</b> `, puedes
bash	Copy code
git checkout 5f3a1e2	
Sin embargo, para mayor comodidad, Git permite que uses cualquier prefijo del hash que sea lo suficientemente único para identificarlo. Entonces, en realidad, podrías es	
bash	Copy code
git checkout 5f3a1e2d	
O incluso menos caracteres si ese prefijo es lo suficientemente único para identificar	el commit.
Si prefieres no usar el hash del commit, también puedes usar referencias relativas, co	omo
`HEAD~n`, donde `n` es el número de commits atrás desde el commit actual, o referencias	
simbólicas como "HEAD", "HEAD", "HEAD", etc.	
Por ejemplo:	
bash	Copy code
git checkout HEAD~1	
Esto te llevará al commit anterior al commit actual.	