Comandos Básicos de Git

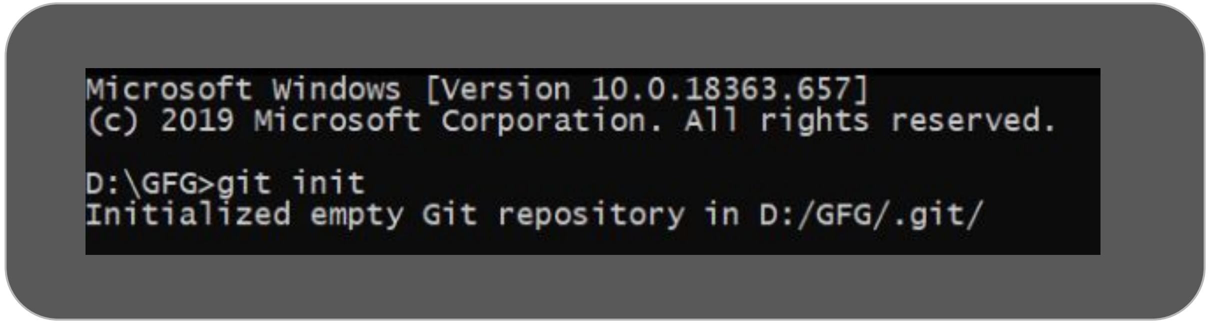
Josefina Mendoza, Felipe Cheker y Francisco Sosa

git init

. Inicializa un nuevo proyecto en git

. Configura el ambiente para el proyecto

. Crea una carpeta oculta que brinda la estructura de datos dentro del proyecto

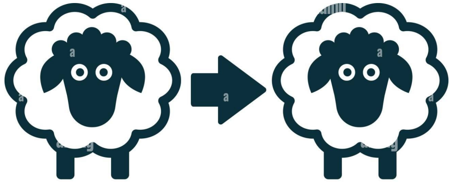


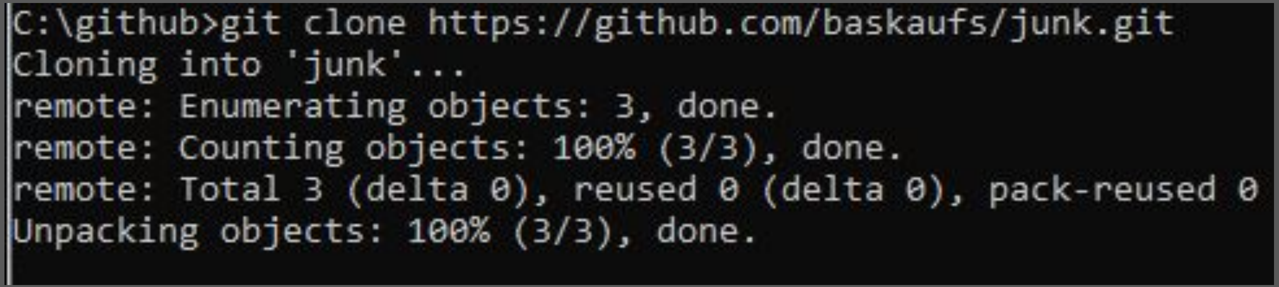
```
Microsoft Windows [Version 10.0.18363.657]
(c) 2019 Microsoft Corporation. All rights reserved.

D:\GFG>git init
Initialized empty Git repository in D:/GFG/.git/
```

git clone

. Genera una copia del repositorio original

. El repositorio puede ser local o remoto





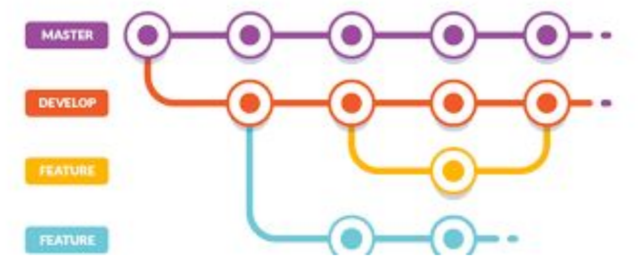
```
C:\github>git clone https://github.com/baskaufs/junk.git
Cloning into 'junk'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), done.
```

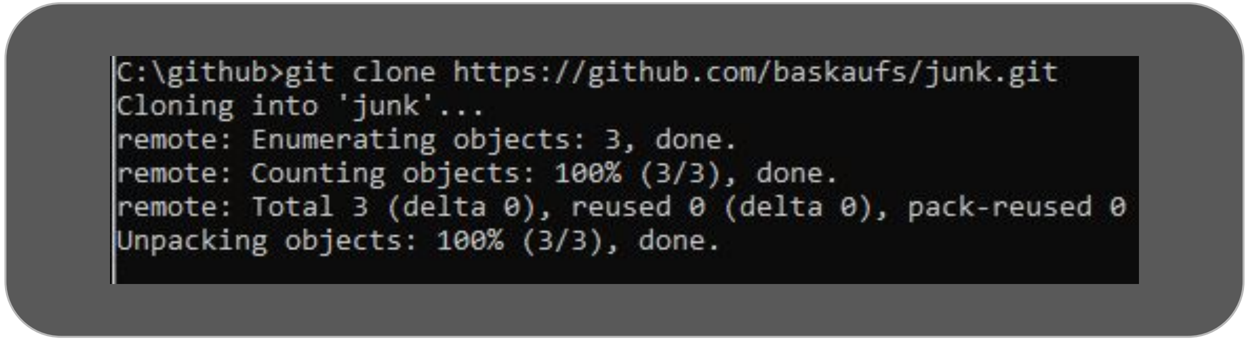
git branch

. Crea una rama de desarrollo independiente

. Se puede listar todas las ramas creadas

. Se puede eliminar ramas





```
C:\github>git clone https://github.com/baskaufs/junk.git
Cloning into 'junk'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), done.
```

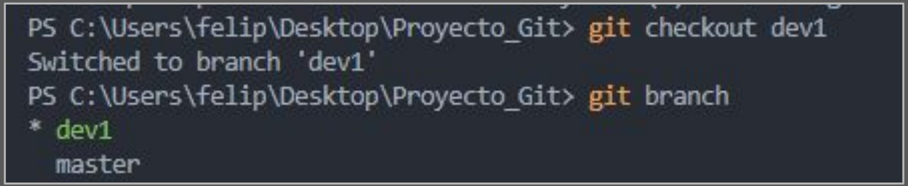
git checkout

. Permite moverse entre ramas

. Permite crear una rama (git checkout -b <rama>)

. Permite hacer checkout en un commit

. Permite reparar archivos (git checkout -- <archivo>)

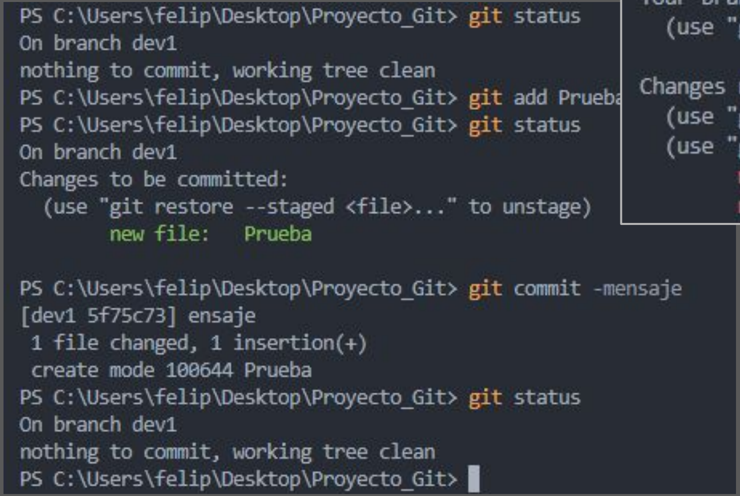


```
PS C:\Users\felip\Desktop\Proyecto_Git> git checkout dev1
Switched to branch 'dev1'
PS C:\Users\felip\Desktop\Proyecto_Git> git branch
* dev1
  master
```

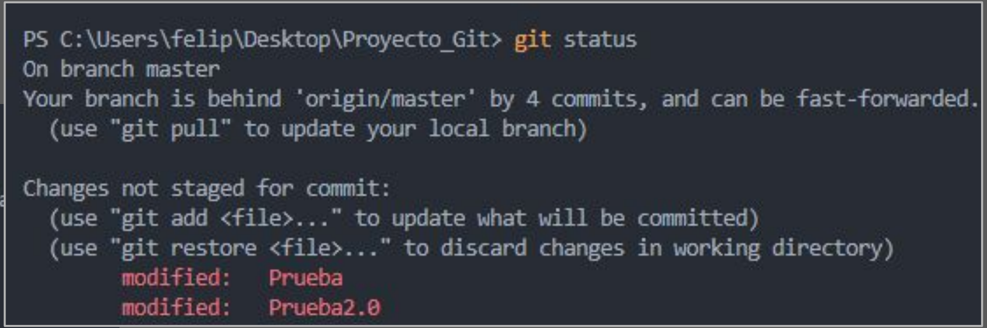
git status

. Permite visualizar si algún archivo tuvo un cambio

. Permite visualizar si hay algún archivo listo para hacer commit



```
PS C:\Users\felip\Desktop\Proyecto_Git> git status
On branch dev1
nothing to commit, working tree clean
PS C:\Users\felip\Desktop\Proyecto_Git> git add Prueba
PS C:\Users\felip\Desktop\Proyecto_Git> git status
On branch dev1
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   Prueba

PS C:\Users\felip\Desktop\Proyecto_Git> git commit -mensaje
[dev1 5f75c73] ensaje
 1 file changed, 1 insertion(+)
 create mode 100644 Prueba
PS C:\Users\felip\Desktop\Proyecto_Git> git status
On branch dev1
nothing to commit, working tree clean
PS C:\Users\felip\Desktop\Proyecto_Git>
```

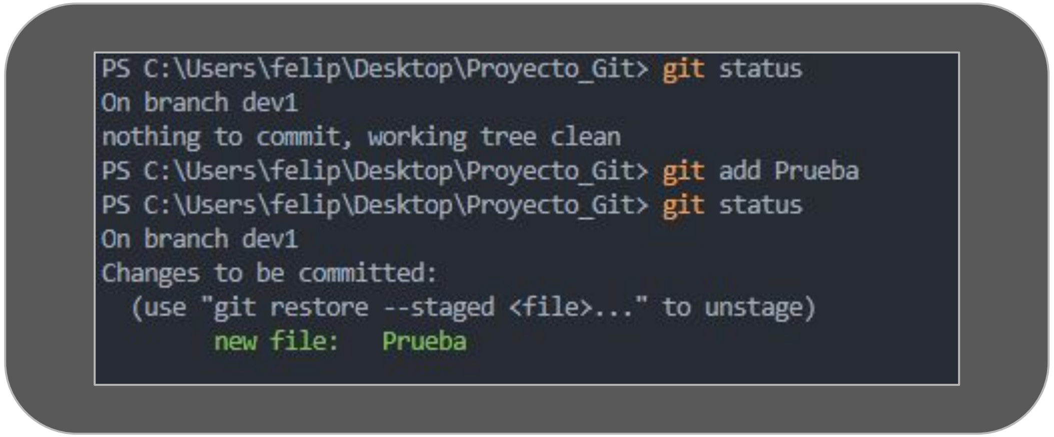


```
PS C:\Users\felip\Desktop\Proyecto_Git> git status
On branch master
Your branch is behind 'origin/master' by 4 commits, and can be fast-forwarded.
  (use "git pull" to update your local branch)

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   Prueba
        modified:   Prueba2.0
```

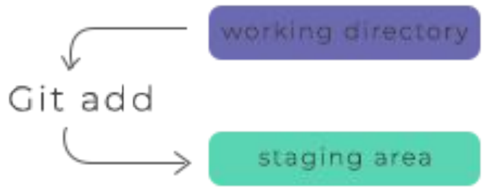
git add

. Permite agregar al ambiente “stage” todos los archivos que hayan tenido al menos un cambio

. Es el paso previo a realizar un commit



```
PS C:\Users\felip\Desktop\Proyecto_Git> git status
On branch dev1
nothing to commit, working tree clean
PS C:\Users\felip\Desktop\Proyecto_Git> git add Prueba
PS C:\Users\felip\Desktop\Proyecto_Git> git status
On branch dev1
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   Prueba
```

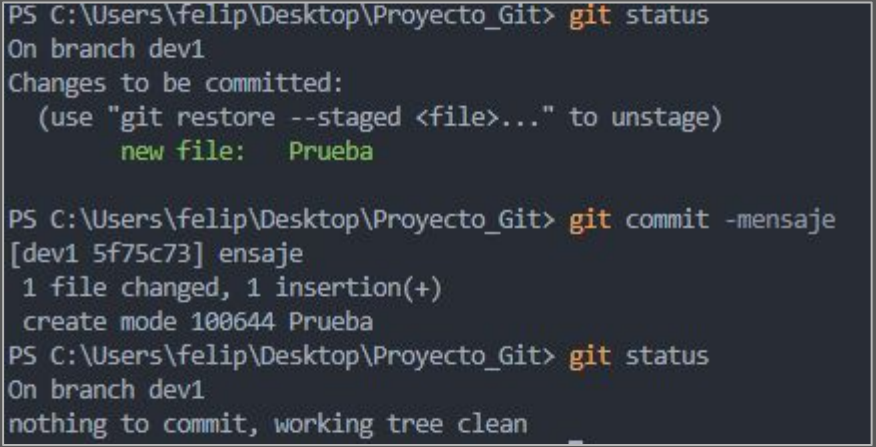


git commit

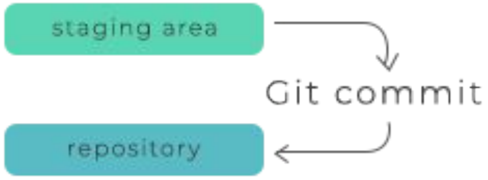
. Para realizar commit necesitamos tener los archivos en el ambiente “stage”

. Establece un punto de control

. git commit -m “Mensaje”



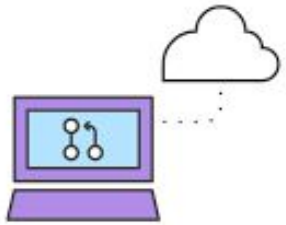
```
PS C:\Users\felip\Desktop\Proyecto_Git> git status
On branch dev1
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   Prueba

PS C:\Users\felip\Desktop\Proyecto_Git> git commit -mensaje
[dev1 5f75c73] ensaje
 1 file changed, 1 insertion(+)
 create mode 100644 Prueba
PS C:\Users\felip\Desktop\Proyecto_Git> git status
On branch dev1
nothing to commit, working tree clean
```

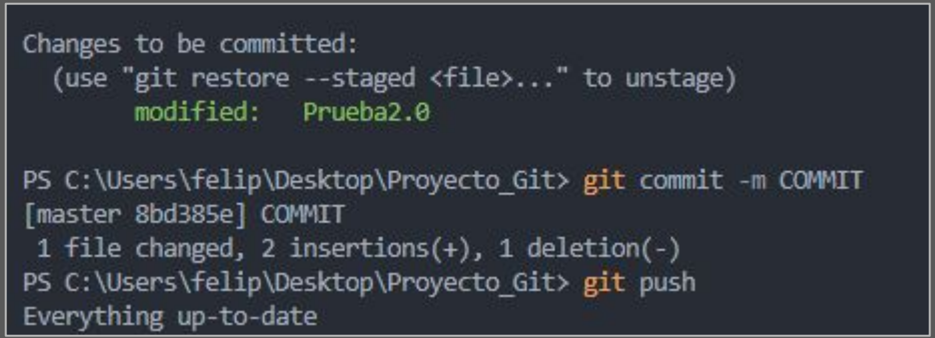


git push

. Este comando envía tus cambios al servidor remoto

. Agregando “-set-upstream <nombre-remoto> <nombre-de-tu-rama>” permite que Git realice un seguimiento automático de la relación entre la rama local y la rama remota

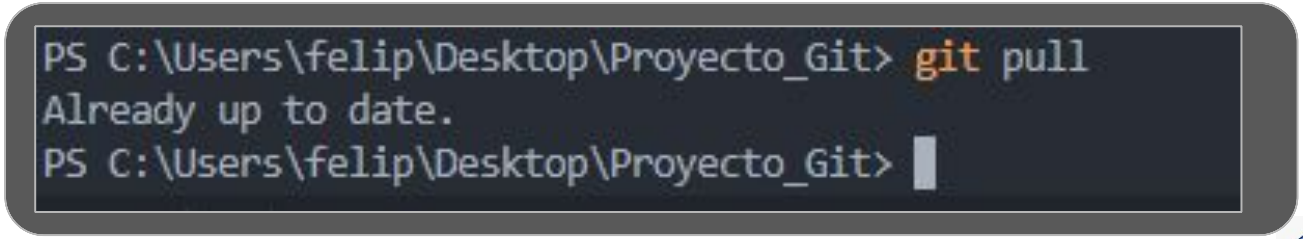




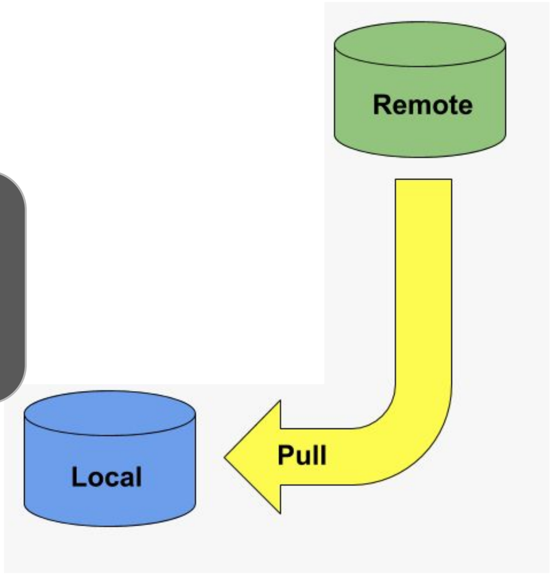
```
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   Prueba2.0

PS C:\Users\felip\Desktop\Proyecto_Git> git commit -m COMMIT
[master 8bd385e] COMMIT
 1 file changed, 2 insertions(+), 1 deletion(-)
PS C:\Users\felip\Desktop\Proyecto_Git> git push
Everything up-to-date
```

git pull

. Este comando se utiliza para descargar contenido desde un repositorio remoto y actualizar al instante el repositorio local para reflejar ese contenido



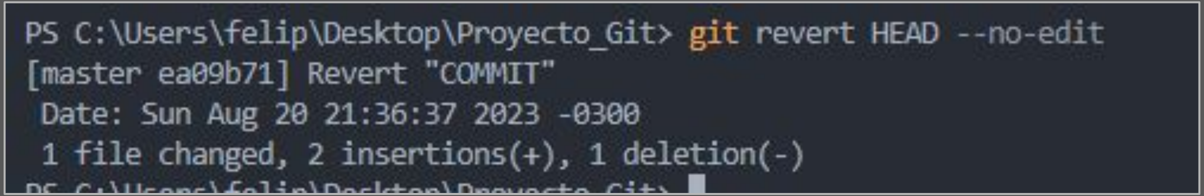
```
PS C:\Users\felip\Desktop\Proyecto_Git> git pull
Already up to date.
PS C:\Users\felip\Desktop\Proyecto_Git>
```



git revert

. Te permite establecer una rama a un commit anterior. Básicamente rebobina el estado de su rama

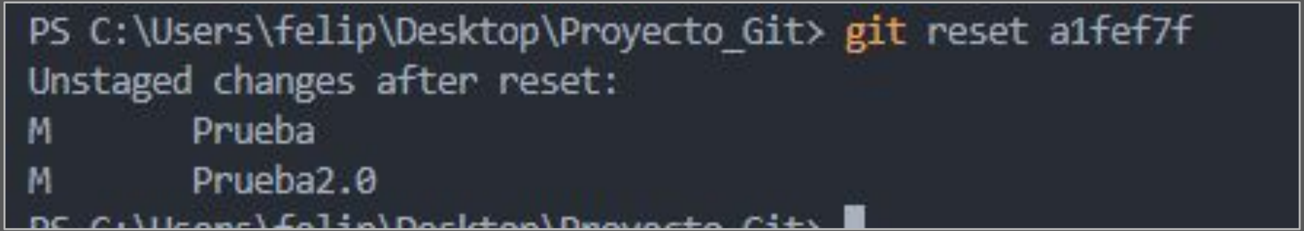
. Este comando genera un commit nuevo deshaciendo los comandos anteriores.



```
PS C:\Users\felip\Desktop\Proyecto_Git> git revert HEAD --no-edit
[master ea09b71] Revert "COMMIT"
 Date: Sun Aug 20 21:36:37 2023 -0300
 1 file changed, 2 insertions(+), 1 deletion(-)
PS C:\Users\felip\Desktop\Proyecto_Git>
```

git reset

. Comando que usamos cuando queremos mover el repositorio a uno anterior commit, descartando cualquier cambio realizado después de eso commit



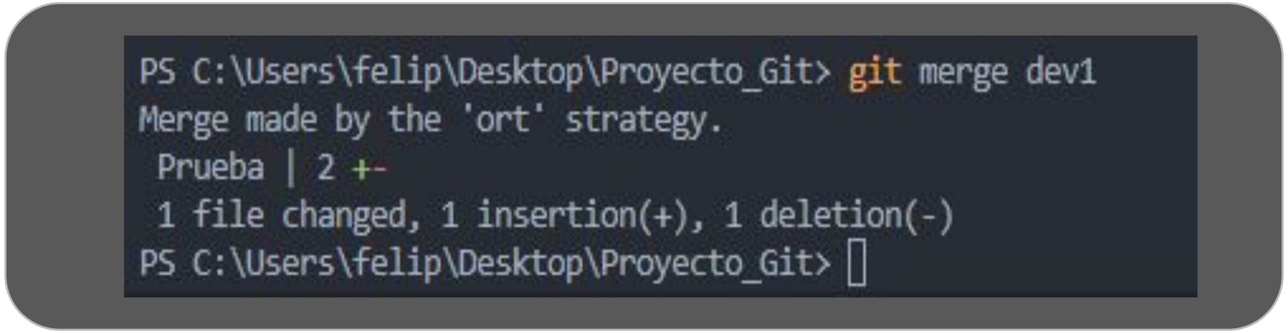
```
PS C:\Users\felip\Desktop\Proyecto_Git> git reset a1fef7f
Unstaged changes after reset:
M       Prueba
M       Prueba2.0
```

Se puede observar que quedaron archivos en el ambiente de desarrollo ya que en el punto que se hizo reset habían archivos distintos a los que hay en la última versión.

git merge

. El comando git merge se usa para fusionar las ramas.

. Para utilizar git merge primero tenemos que pararnos en la rama a la que queremos llevar los cambios, luego aplicamos el comando.



```
PS C:\Users\felip\Desktop\Proyecto_Git> git merge dev1
Merge made by the 'ort' strategy.
 Prueba | 2 +-
 1 file changed, 1 insertion(+), 1 deletion(-)
PS C:\Users\felip\Desktop\Proyecto_Git> 
```

Recursos de aprendizaje

https://github.com/pcottle/learnGitBranching - flujo de ramas

https://learngitbranching.js.org/?locale=es_AR - Aprender git a través de un juego

https://ndpsoftware.com/git-cheatsheet.html#loc=remote_repo; - Definiciones de comandos de git en sus respectivos ambientes.

https://gitexercises.fracz.com/ - Ejercicios de git

https://marklodato.github.io/visual-git-guide/index-en.html#basic-usage - Explicación de git con imágenes

Demostración Práctica

Recomendaciones de uso

1. Detallar el contenido y el objetivo del proyecto, utilizar organización de carpetas con nombres claros.
2. Commits pequenos: Generar commits pequeños y coherentes, cada uno con dicho propósito. Esto hace que sea más fácil el entendimiento de cada cambio y deshacerlos si es necesario.
3. Ramas: Utilizar las ramas para trabajar en nuevas características o correcciones de errores. Esto hace que mantengamos el código principal estable mientras realizamos pruebas o solucionamos problemas.
4. Pull requests: Al trabajar en equipo, debemos hacer uso de Pull Requests (PR) para incorporar cambios a la rama principal. Esto facilita la revisión del código y genera posibles cambios antes de la fusión.

Bibliografía

https://docs.github.com/es/get-started/using-git/about-git

https://git-scm.com/book/es/v2/Ap%C3%A9ndice-C%3A-Comandos-de-Git-Obtener-y-Crear-Proyectos

https://git-scm.com/docs/git-clone

https://github.com/git-guides/git-clone

https://www.atlassian.com/git/tutorials/setting-up-a-repository/git-clone

https://git-scm.com/docs/git-branch

https://git-scm.com/docs/git-checkout

https://www.atlassian.com/git/tutorials/using-branches/git-checkout

ChatGPT - “git checkout”

foto diapositiva 7 y 8 - https://www.w3docs.com/learn-git/git-add.html