LABORATORIO LOCAL

Laboratorio BÁSICO:

Paso 1: Crea un nuevo directorio para el laboratorio

Primero instalamos la última versión de Git utilizando el comando **- - version**

```
PS C:\Users\Mañana\Desktop\Bootcamp> git --version
git version 2.48.1.windows.1
PS C:\Users\Mañana\Desktop\Bootcamp> mkdir git-lab


    Directorio: C:\Users\Mañana\Desktop\Bootcamp


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----        17/03/2025     10:34                git-lab


PS C:\Users\Mañana\Desktop\Bootcamp> cd git-lab
PS C:\Users\Mañana\Desktop\Bootcamp\git-lab>
```

Ahora lo que debemos hacer es crear un directorio y meternos dentro de él, a través de los comandos de **mkdir** y **cd**

Paso 2: Inicializa un repositorio Git

Inicializamos un repositorio con el comando **git init**

```
PS C:\Users\Mañana\Desktop\Bootcamp\git-lab> git init
Initialized empty Git repository in C:/Users/Mañana/Desktop/Bootcamp/git-lab/.git/
```

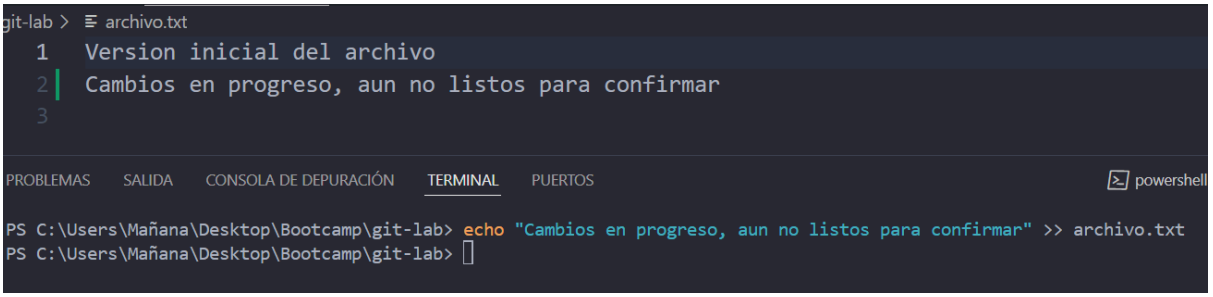
Paso 3: Crea un archivo y confírmalo en el repositorio

Creamos el **archivo.txt** y le hacemos **un git** para guardar el cambio y le añadimos el comentario para saber que estamos guardando en todo momento

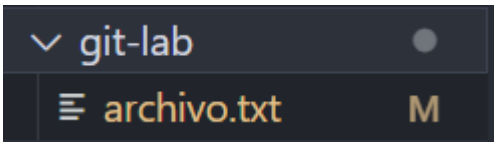
```
PS C:\Users\Mañana\Desktop\Bootcamp\git-lab> echo "Version inicial del archivo" > archivo.txt
PS C:\Users\Mañana\Desktop\Bootcamp\git-lab> git add archivo.txt
PS C:\Users\Mañana\Desktop\Bootcamp\git-lab> git commit -m "Agregar versión inicial del archivo.txt"
[master (root-commit) 6bfb40c] Agregar versión inicial del archivo.txt
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 archivo.txt
```

Paso 4: Haz algunos cambios no confirmados en el archivo

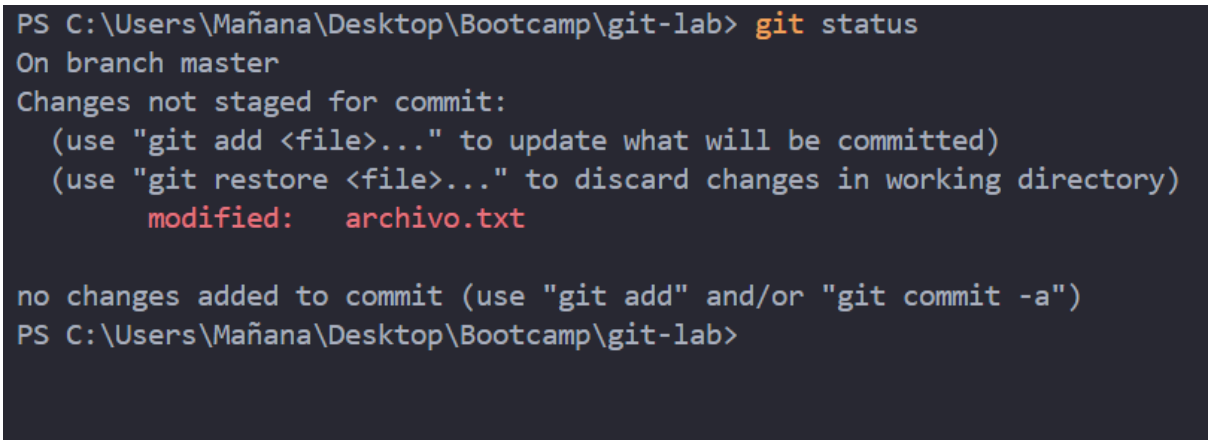
Ahora utilizando **echo** le indicamos un cambio en el archivo que todavía no han sido confirmados:



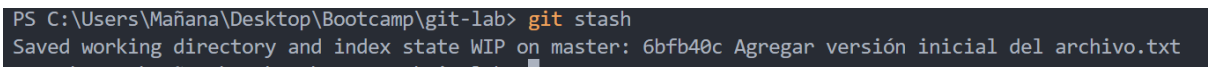
Esta **M en amarillo** nos está indicando que hemos **modificado el archivo**, pero aun no permanentemente, es decir, debemos hacer un **commit** para guardarlo en el historial de Git



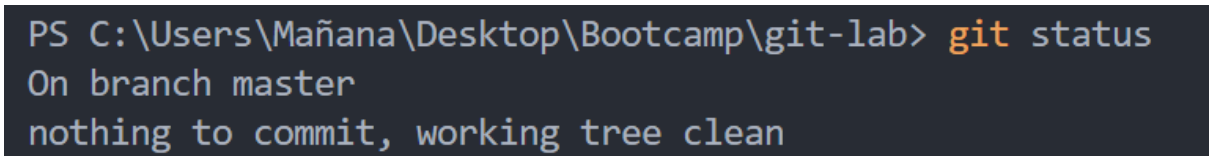
Como podemos ver, realizando un **git status** vemos los cambios aún no confirmados en nuestro **archivo.txt:**



Paso 5: Usa git stash para guardar los cambios no confirmados
Procedemos a realizar los cambios con **git stash** y vemos que se ha guardado correctamente:



Nos cercioramos que no hay cambios pendientes con **git status:**



Paso 6: Recupera los cambios guardados con git stash apply

A través del **git stash apply,** vuelves a tener cambios sin confirmar, es como que vuelves atrás, básicamente se aplican los cambios que habíamos guardado en el stash anteriormente :

```
PS C:\Users\Mañana\Desktop\Bootcamp\git-lab> git stash apply
On branch master
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:   archivo.txt

no changes added to commit (use "git add" and/or "git commit -a")
```

Paso 7: Confirma los cambios
Finalmente confirmamos los cambios:

```
PS C:\Users\Mañana\Desktop\Bootcamp\git-lab> git add archivo.txt
PS C:\Users\Mañana\Desktop\Bootcamp\git-lab> git commit -m "Aplicar cambios guardados con git stash"
[master baa0f8b] Aplicar cambios guardados con git stash
 1 file changed, 0 insertions(+), 0 deletions(-)
```

Paso 8: Limpia el stash

Para limpiar el stash podemos usar el **drop pero tienes que saber que no se aplican los cambios**

```
PS C:\Users\Mañana\Desktop\Bootcamp\git-lab> git stash drop
Dropped refs/stash@{0} (f4319a1c8e897b5a7b16ad8d806ed8e0acb1be65)
```

Si te fijas puedes ver que sigue apareciendo **el directorio con el archivo.txt:**

Para eliminar el stash y aplicar los cambios de una sola vez podemos usar el **git stash pop**

Laboratorio INTERMEDIO:

Paso 1: Crea el directorio y el repositorio

Ahora en este nuevo laboratorio vamos a crear el directorio e iniciar el repositorio como hemos hecho antes utilizando **mkdir** para crear el nuevo directorio, **cd** para meternos dentro de él y el **git init** para iniciar el repositorio:

```
PS C:\Users\Mañana\Desktop\Bootcamp> mkdir git-lab-complejo


    Directorio: C:\Users\Mañana\Desktop\Bootcamp


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----        17/03/2025     12:09                git-lab-complejo


PS C:\Users\Mañana\Desktop\Bootcamp> cd .\git-lab-complejo\
PS C:\Users\Mañana\Desktop\Bootcamp\git-lab-complejo> git init
Initialized empty Git repository in C:/Users/Mañana/Desktop/Bootcamp/git-lab-complejo/.git/
```

Paso 2: Crea la rama principal y agrega un archivo

Creamos la rama principal del main, utilizamos el **echo** para que muestre por pantalla y lo guarde dentro del archivo **main.txt** y luego le realizamos el **commit** para hacer un punto de control:

```
PS C:\Users\Mañana\Desktop\Bootcamp\git-lab-complejo> echo "Version inicial en main" > main.txt
PS C:\Users\Mañana\Desktop\Bootcamp\git-lab-complejo> git add main.txt
PS C:\Users\Mañana\Desktop\Bootcamp\git-lab-complejo> git commit -m "Agregar archivo inicial en main"
[master (root-commit) 691f7a1] Agregar archivo inicial en main
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 main.txt
```

Paso 3: Crea una nueva rama para una característica (feature)

Ahora hemos creado la **rama feature** y hemos hecho lo mismo de antes, crear un archivo txt (***caracteristica.txt***) y hemos realizado un **commit**:

```
PS C:\Users\Mañana\Desktop\Bootcamp\git-lab-complejo> git checkout -b feature/nueva-caracteristica
Switched to a new branch 'feature/nueva-caracteristica'
PS C:\Users\Mañana\Desktop\Bootcamp\git-lab-complejo> echo "Nueva caracteristica en desarrollo" > caracteristica.txt
PS C:\Users\Mañana\Desktop\Bootcamp\git-lab-complejo> git add caracteristica.txt
PS C:\Users\Mañana\Desktop\Bootcamp\git-lab-complejo> git commit -m "Comenzar nueva caracteristica"
[feature/nueva-caracteristica 53a1249] Comenzar nueva caracteristica
 1 file changed, 0 insertions(+), 0 deletions(-)
```

Paso 4: Realiza cambios en la nueva característica (sin confirmar)

Aqui basicamente **modificamos el documento sin confirmarlo** y lo verificamos con el **git status :**

```
PS C:\Users\Mañana\Desktop\Bootcamp\git-lab-complejo> echo "Progreso no terminado en la nueva caracteristica" >> caracteristica.txt
PS C:\Users\Mañana\Desktop\Bootcamp\git-lab-complejo> git status
On branch feature/nueva-caracteristica
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:   caracteristica.txt

no changes added to commit (use "git add" and/or "git commit -a")
```

Paso 5: Recibes una solicitud urgente para corregir un bug en la rama principal

Si nos damos cuenta este mensaje de error nos dice que no encuentra ninguna **rama main** debido a que nuestra rama se llama **master**, si hacemos un **git log** podremos darnos cuenta:

```
PS C:\Users\Mañana\Desktop\Bootcamp\git-lab-complejo> git checkout main
error: pathspec 'main' did not match any file(s) known to git
```

```
PS C:\Users\Mañana\Desktop\Bootcamp\git-lab-complejo> git log
commit 53a1249129b47da76f9aee06f6d21cd22a6edf5b (HEAD -> feature/nueva-caracteristica)
Author: ulippuden99 <146721319+ulippuden99@users.noreply.github.com>
Date:   Mon Mar 17 12:31:21 2025 +0100

    Comenzar nueva caracteristica

commit 691f7a1c844e9c39997e84f369c0f85c4d8121e0 (master)
Author: ulippuden99 <146721319+ulippuden99@users.noreply.github.com>
Date:   Mon Mar 17 12:21:36 2025 +0100

    Agregar archivo inicial en main
```

Antes de intentar cambiar de rama con el **checkout** vamos a cambiar el nombre de la rama principal master y la llamamos main, utilizaremos el comando **git branch -m NOMBRE NUEVO NOMBRE** :

```
PS C:\Users\Mañana\Desktop\Bootcamp\git-lab-complejo> git branch -m master main
```

Vamos a verificarlo con un **log:**
Podemos ver que ahora si se llama **main:**

```
PS C:\Users\Mañana\Desktop\Bootcamp\git-lab-complejo> git log
commit 53a1249129b47da76f9aee06f6d21cd22a6edf5b (HEAD -> feature/nueva-caracteristica)
Author: ulippuden99 <146721319+ulippuden99@users.noreply.github.com>
Date:   Mon Mar 17 12:31:21 2025 +0100

    Comenzar nueva caracteristica

commit 691f7a1c844e9c39997e84f369c0f85c4d8121e0 (main)
Author: ulippuden99 <146721319+ulippuden99@users.noreply.github.com>
Date:   Mon Mar 17 12:21:36 2025 +0100
```

Ahora si realizamos un **checkout main** para cambiar de rama sin haber confirmado el cambio de antes nos damos cuenta que no nos permite hacerlo y nos avisa de que tenemos que confirmar los cambios:

```
PS C:\Users\Mañana\Desktop\Bootcamp\git-lab-complejo> git checkout main
error: Your local changes to the following files would be overwritten by checkout:
        caracteristica.txt
Please commit your changes or stash them before you switch branches.
Aborting
```

Paso 6: Usa git stash para guardar tus cambios

Ahora vamos a guardar los cambios para poder cambiar de rama con el git stash:

```
PS C:\Users\Mañana\Desktop\Bootcamp\git-lab-complejo> git stash
Saved working directory and index state WIP on feature/nueva-caracteristica: 53a1249 Comenzar nueva caracterist
ica
```

Paso 7: Cambia a la rama main y corrige el bug

Vamos a cambiar de rama:

```
PS C:\Users\Mañana\Desktop\Bootcamp\git-lab-complejo> git checkout main
Switched to branch 'main'
```

Vamos a corregir el archivo: hacemos el **echo** para escribir o transmitir el cambio que queremos hacer, con el **add** lo mandamos y creamos un nuevo punto de control con el **commit**:

```
PS C:\Users\Mañana\Desktop\Bootcamp\git-lab-complejo> echo "Correccion de bug urgente" >> main.txt
PS C:\Users\Mañana\Desktop\Bootcamp\git-lab-complejo> git add main.txt
PS C:\Users\Mañana\Desktop\Bootcamp\git-lab-complejo> git commit -m  "Correccion de bug urgente en main,"
[main b27a493] Correccion de bug urgente en main,
 1 file changed, 0 insertions(+), 0 deletions(-)
```

Paso 8: Vuelve a la rama de tu característica

Ahora volvemos a la rama **feature**:

```
PS C:\Users\Mañana\Desktop\Bootcamp\git-lab-complejo> git checkout feature/nueva-caracteristica
Switched to branch 'feature/nueva-caracteristica'
```

Paso 9: Recupera tus cambios con git stash apply

Ahora lo que debemos hacer es un **git stash apply** para recuperar los cambios de antes(Básicamente volver atrás a los cambios del commit anterior):

```
PS C:\Users\Mañana\Desktop\Bootcamp\git-lab-complejo> git stash apply
On branch feature/nueva-caracteristica
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:   caracteristica.txt
```

Una vez recuperado los cambios podemos avanzar realizando otro **commit:**

```
PS C:\Users\Mañana\Desktop\Bootcamp\git-lab-complejo> git add caracteristica.txt
PS C:\Users\Mañana\Desktop\Bootcamp\git-lab-complejo> git commit -m "Continuar con el desarrollo de la nueva ca
racteristica"
[feature/nueva-caracteristica 6c28a22] Continuar con el desarrollo de la nueva caracteristica
 1 file changed, 0 insertions(+), 0 deletions(-)
```

Paso 10: Limpia el stash

Vamos a limpiar el stash:

```
PS C:\Users\Mañana\Desktop\Bootcamp\git-lab-complejo> git stash drop
Dropped refs/stash@{0} (85bbabbe182db46ff00174669a66bd091e6e0f33)
```

Una vez eliminado podemos verificarlo con el **log:**

Podemos apreciar que ya no aparece nuestra rama **main** ni la rama **feature**

```
PS C:\Users\Mañana\Desktop\Bootcamp\git-lab-complejo> git log
commit 6c28a224a6914c8b03525c5d52b11384cffa1120 (HEAD -> feature/nueva-caracteristica)
Author: ulippuden99 <146721319+ulippuden99@users.noreply.github.com>
Date:   Mon Mar 17 13:07:13 2025 +0100

    Continuar con el desarrollo de la nueva caracteristica

commit 53a1249129b47da76f9aee06f6d21cd22a6edf5b
Author: ulippuden99 <146721319+ulippuden99@users.noreply.github.com>
Date:   Mon Mar 17 12:31:21 2025 +0100

    Comenzar nueva caracteristica
```