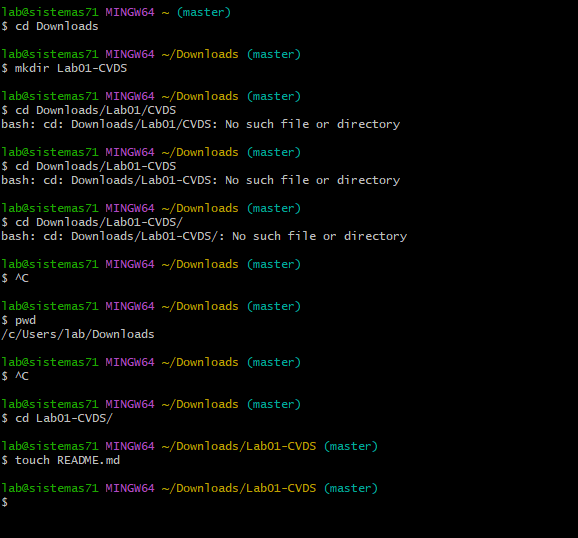
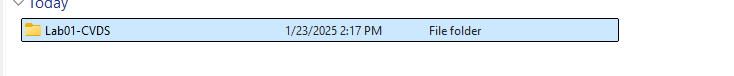
Cristian David Silva perilla

1 y 2)

Procedemos a crear el repositorio de manera local en la terminal de git bash



Como se recomienda agregar el archive readme lo agregamos con el comando touch





**3) como se usan estos comandos git add y git commit -m “mensaje”?**

**git add:** Este comando sirve para agregar archivos o cambios especificos, pero es una preparación para el commit mas no lo realiza.

Ejemplo de uso:

git add archivo.txt(Esto agrega el archivo archivo.txt al área de preparación.)

Para agregar todo:

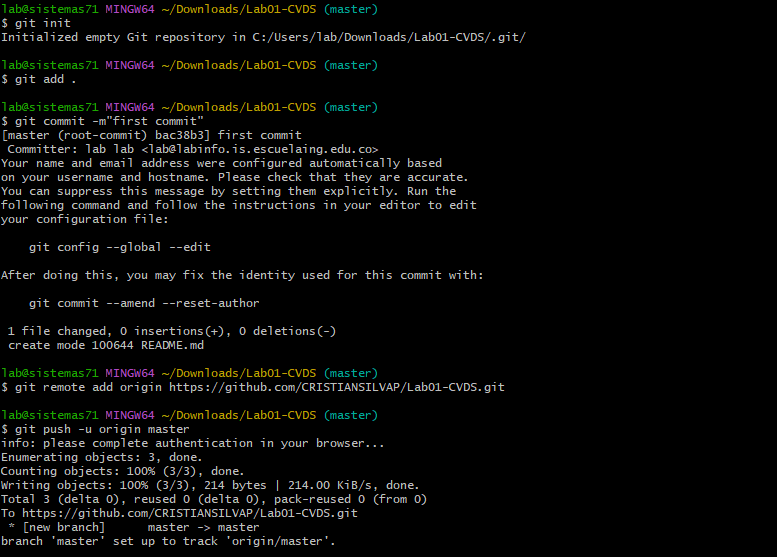
git add . (Esto agregará todos los cambios de los archivos que han sido modificados o creados.)

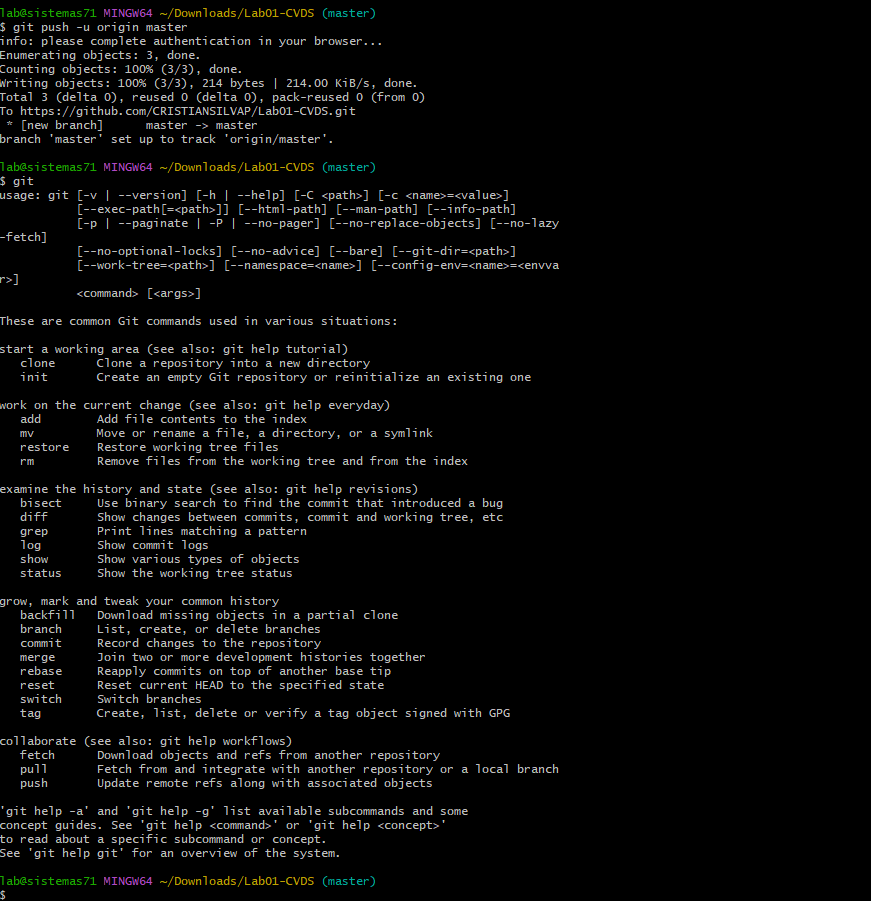
**git commit**: Este comando es el que realiza los cambios que se hallan añadido al area de preparacion y el mensaje se puede usar para describir lo que se haya hecho en el cambio.

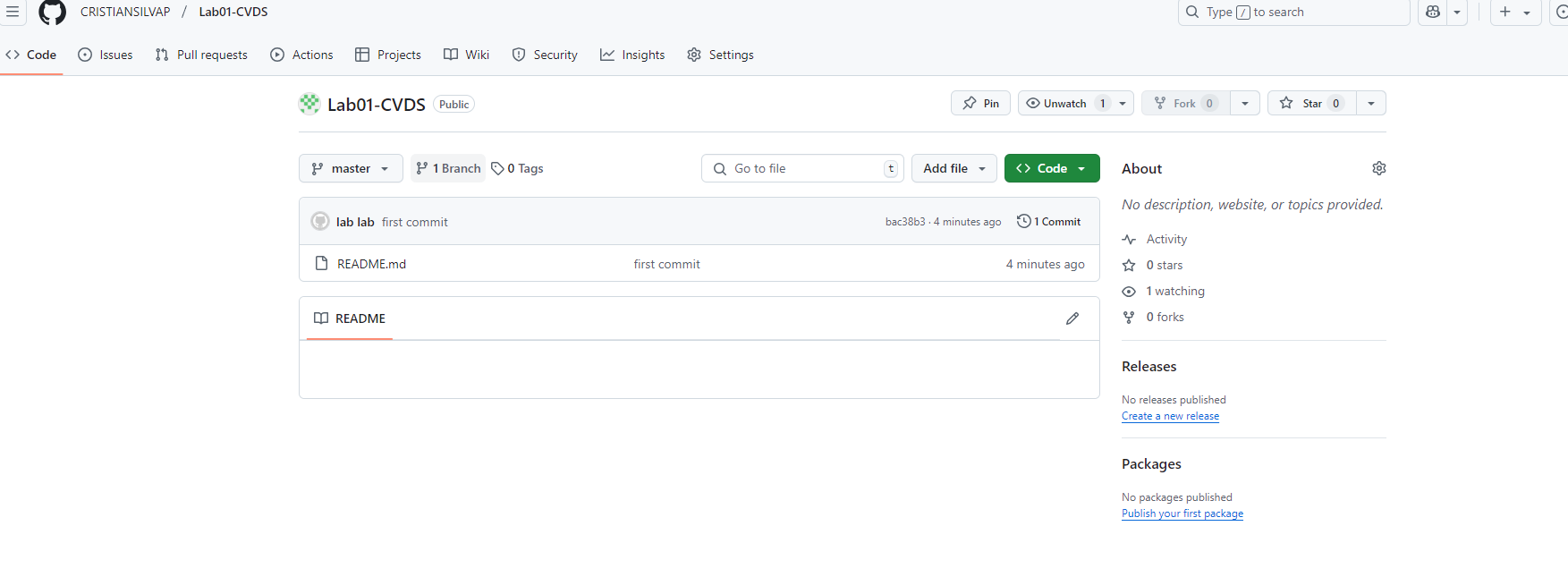
Ejemplo de uso:

git commit -m "Añadir nueva funcionalidad al archivo de inicio" (La idea es que el mensaje sea directo y consciso especificando el cambio realizado).

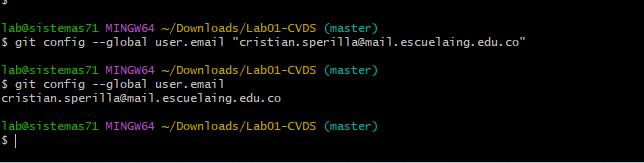
6 y 7)







8)

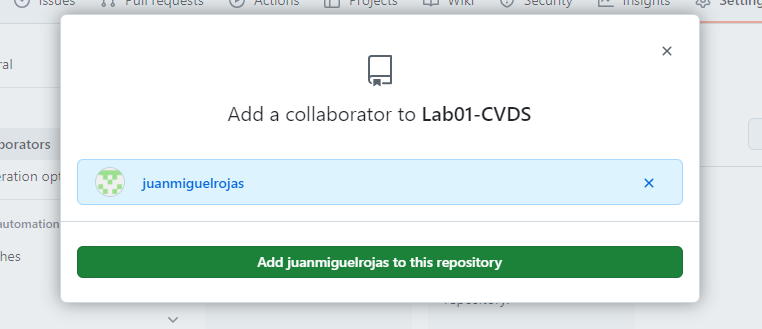


**PARTE II (Trabajo en parejas)**

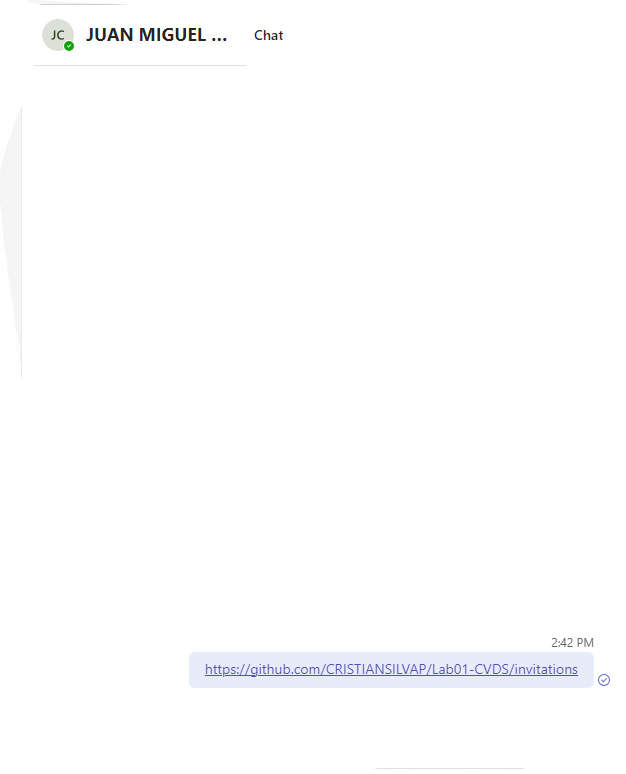
1. Se escogen los roles para trabajar en equipo, una persona debe escoger ser "Owner" o Propietario del repositorio y la otra "Collaborator" o Colaborador en el repositorio.

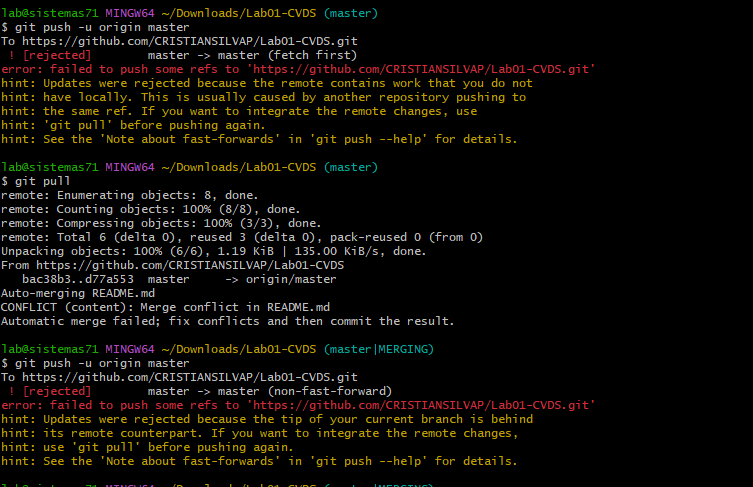
Cristian silva será el owner y juan Miguel rojas será el collaborator

1. El owner agrega al colaborador con permisos de escritura en el repositorio que creó en la parte 1



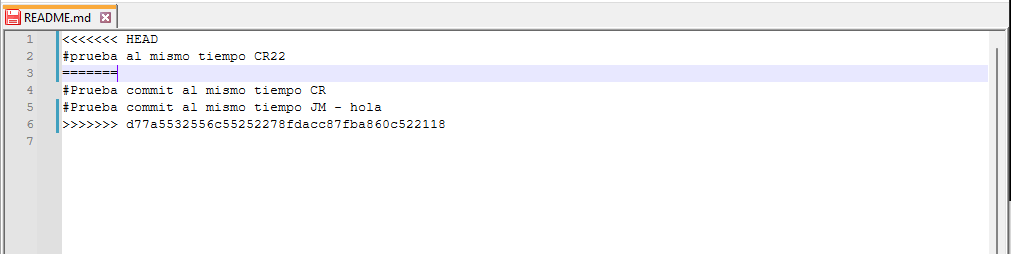
1. El owner le comparte la url via Teams al colaborador

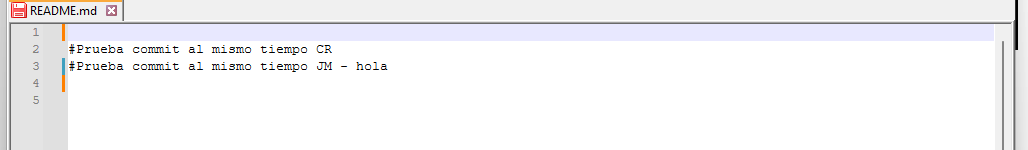


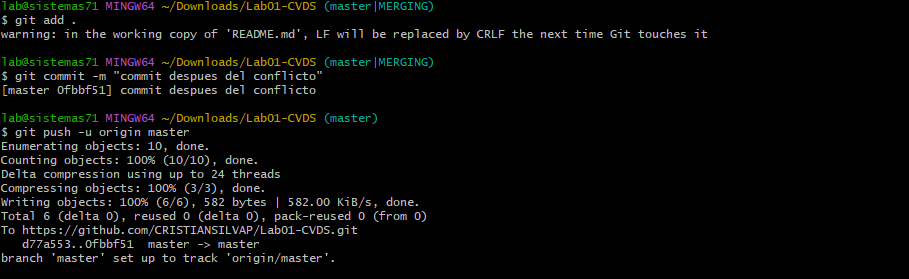
1. El colaborador acepta la invitación al repositorio
2. Owner y Colaborador editan el archivo README.md al mismo tiempo e intentan subir los cambios al mismo tiempo.  
    
3. ¿Que sucedió?

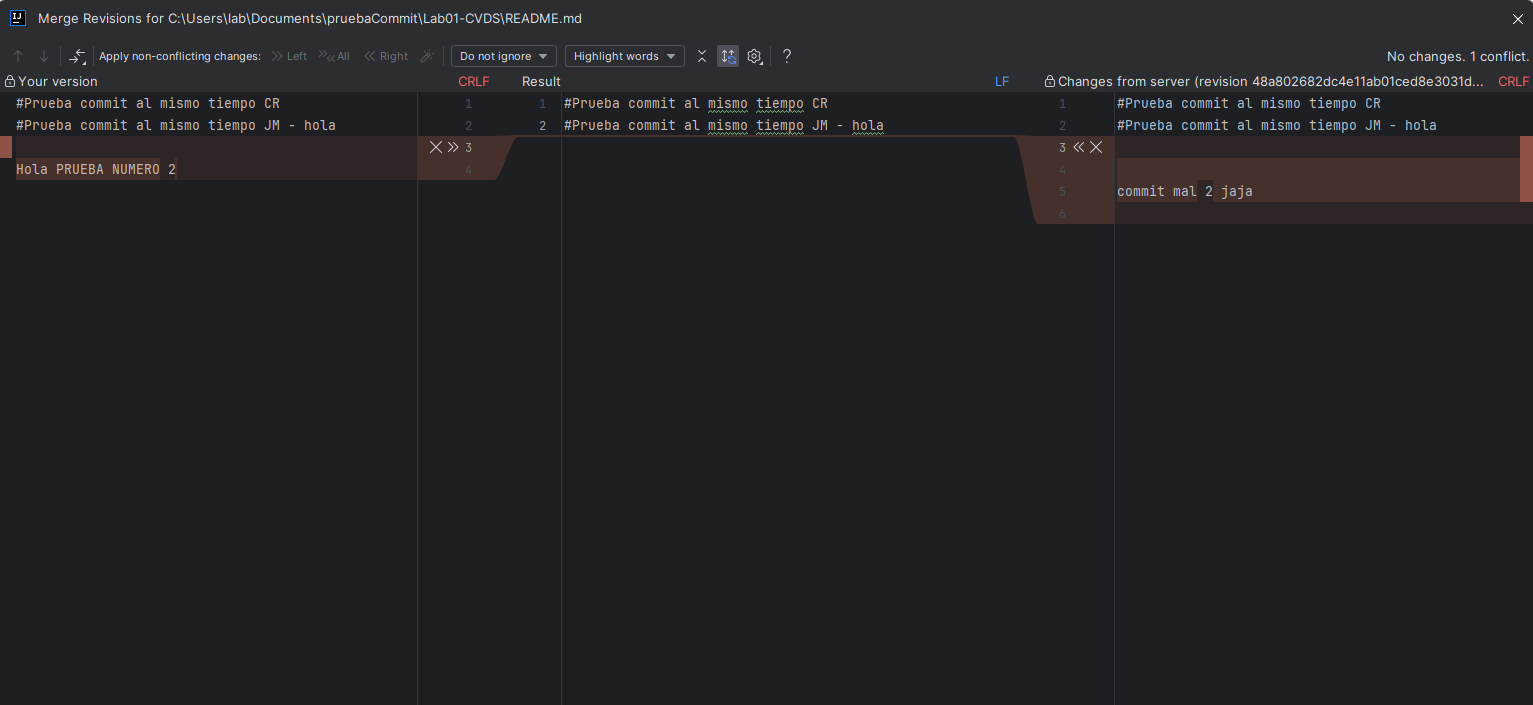
Lo que pasa es que al intentar hacer los cambios al tiempo el git toma el que primero llegué poniendo en conflicto el que llegue despues como se evidencia en la imagen anterior

1. La persona que perdió la competencia de subir los cambios, tiene que resolver los conflictos, cúando haces pull de los cambios, los archivos tienen los símbolos <<< === y >>> (son normales en la resolución de conflictos), estos conflictos debes resolverlos manualmente.



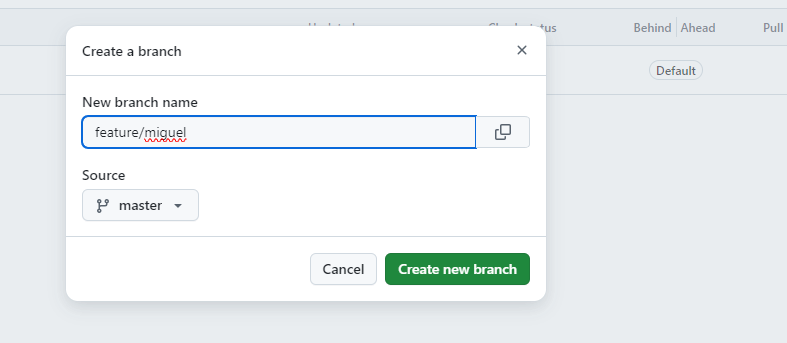
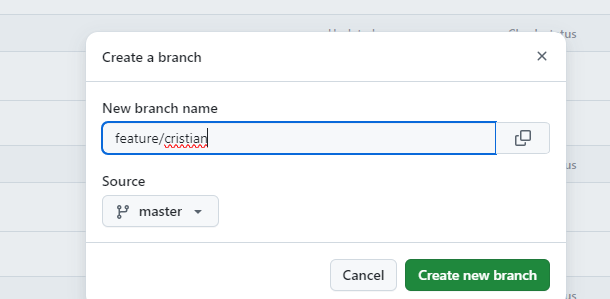
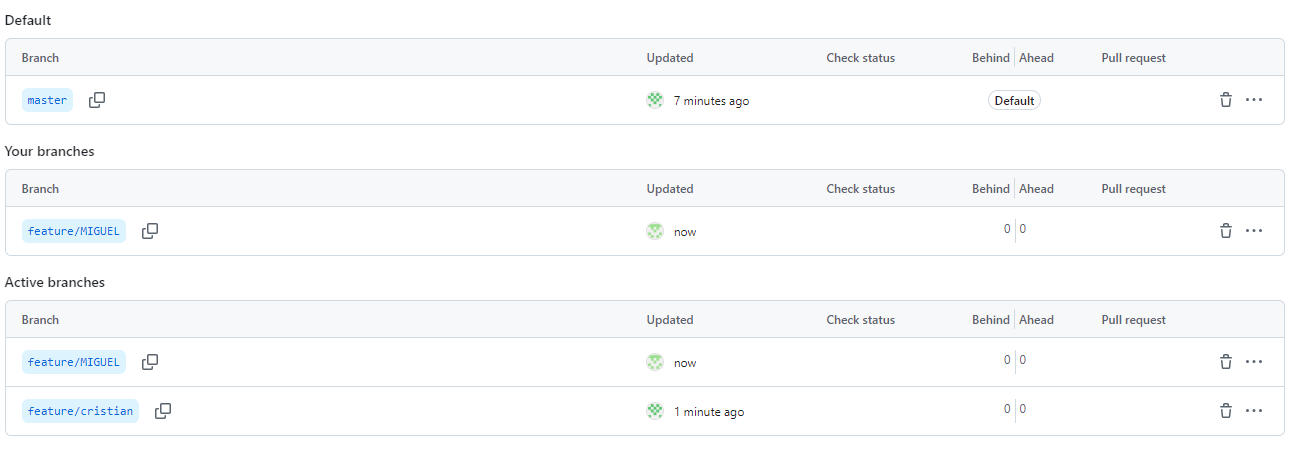
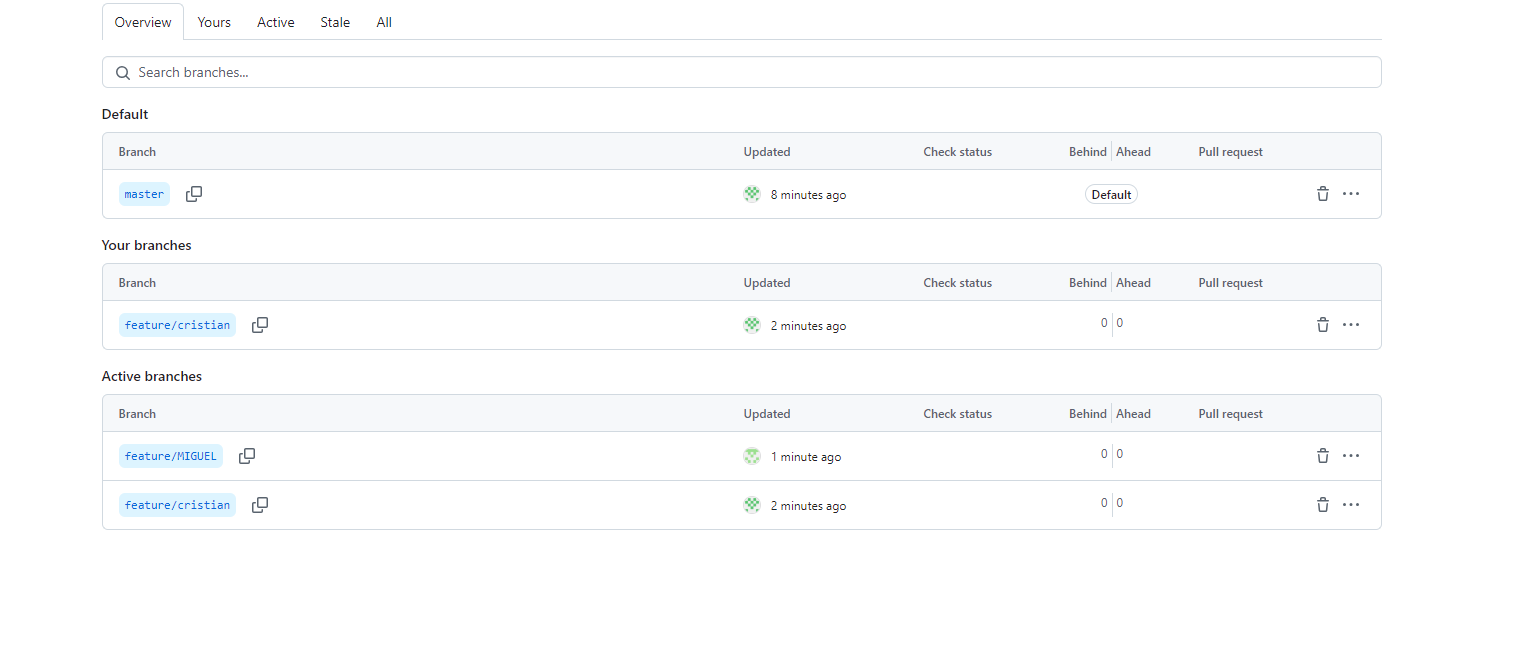




1. Volver a repetir un cambio sobre el README.md ambas personas al tiempo para volver a tener conflictos.
2. Resuelvan el conflicto con IntelliJ si es posible   
   

De esta forma ya sabes resolver conflictos directamente sobre los archivos y usando un IDE como IntelliJ, esto te será muy útil en los futuros trabajos en equipo con Git.

**PARTE III (Trabajo de a parejas)**

1. ¿Hay una mejor forma de trabajar con git para no tener conflictos?
2. ¿Qué es y como funciona el **Pull Request**?  
   Un Pull Request (PR) es una solicitud para integrar cambios de una rama a otra, comúnmente usada en plataformas como GitHub, GitLab o Bitbucket. Cuando trabajas en una rama (por ejemplo, una rama de características o de corrección de errores), un pull request te permite pedirle a otra persona (o al mismo equipo) que revise y apruebe tus cambios antes de fusionarlos con la rama principal.
3. Creen una rama cada uno y suban sus cambioS  
     
     
     
   

1. Tanto owner como colaborador hacen un cambio en el README.md y hacen un Pull Request (PR) a la rama main/master

**(Recomendación : deben trabajar en equipo y distribuirse de mejor manera quien va a modificar qué, para evitar modificar los mismos archivos al mismo tiempo, si no se editan los mismos archivos la resolución de conflictos es automática)**

1. Teniendo en cuenta la recomendación, mezclen los cambios a la rama main a través de PR con el check/review/approval del otro compañero (Cuando se hace merge se deberían borrar las ramas en github)

***Como Borrar Ramas después de un Pull Request***

Se dirigen a la configuración de su repositorio:

Y en general

Se dirigen al final en el área del pull requests y seleccionan “Automatically delete head branches”

***Aprobación Pull Request***

Nos dirigimos (todavía en configuraciones) a Branches, en esta visualizarán donde daremos protección de nuestras ramas, seleccionamos Add rule

Aquí damos el nombre de nuestra rama (Verificar el nombre tal cual lo tenemos en nuestro repositorio) y seleccionamos la primera opción como se muestra, así estamos requiriendo que cuando se haga ese pull request en nuestra rama se necesita aprobación de otro compañero

Vamos al final y damos clic en Create

Y así protegimos nuestra rama principal, esto se vuelve muy relevante cuando trabajamos en parejas o en equipos, deberían tener un mensaje final que se vea así