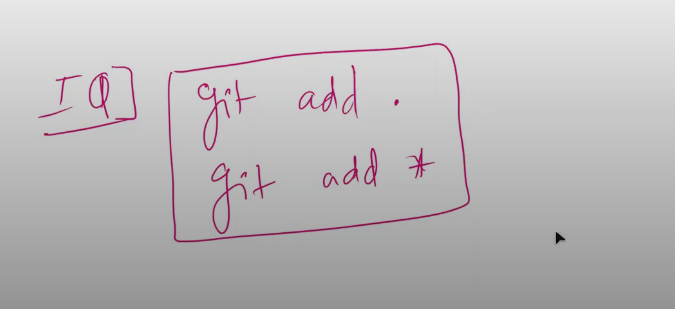
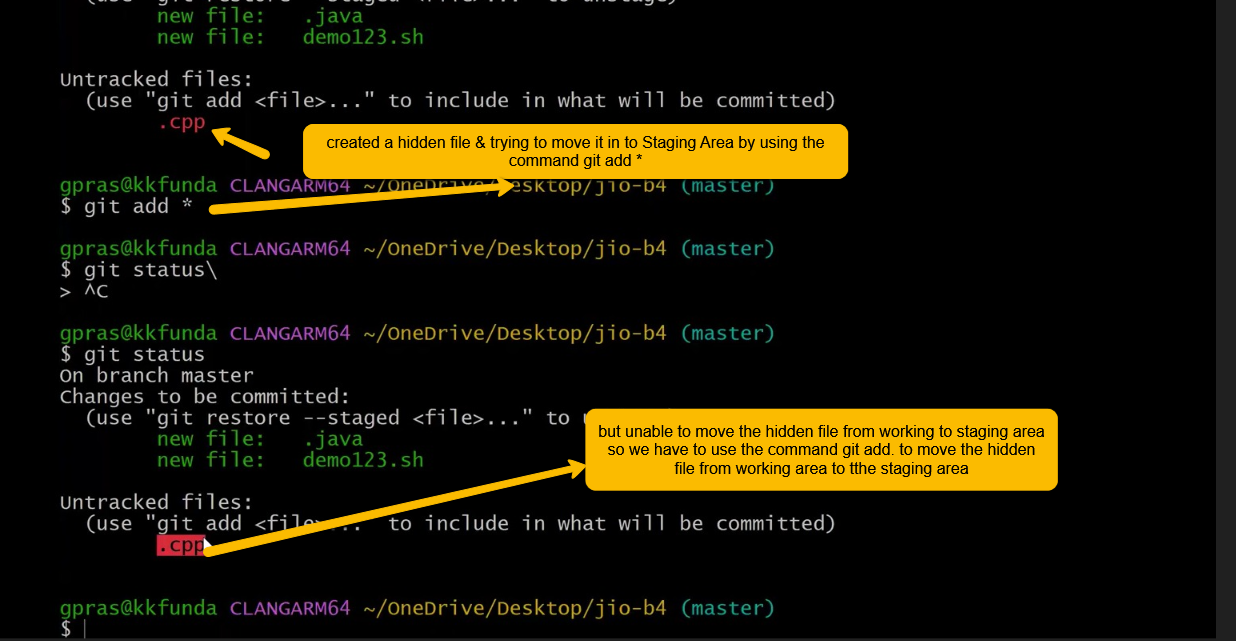
Let’s start this class with a n Interview Question   
Q) what is the difference between **git add .** & **git add \* ?**

1. **git add . (It will move all files and Directories from working area to staging area including hidden files also)  
     
   check the below example where we have created hidden file “.cpp” and trying to move it to the staging area using git add \* but got miserably failed to move it** 
2. **Git add \* command will move only new and modified files from working Area to Staging area and ignore the hidden files (.cpp)  
     
     
   below Another Answer from chatgpt   
     
   ------------------------------------------------------------------------------------------------------  
     
   Q) What is the difference between `git add .` and `git add \*`?**

**A)**

**- `git add .` adds all changes (new files, modified files, and deleted files) from the current directory and its subdirectories to the staging area.**

**- `git add \*` adds only new and modified files in the current directory (not subdirectories), and ignores hidden files (like `.gitignore`, `.env`, etc.).**

|  |  |  |  |
| --- | --- | --- | --- |
| **Command** | **Adds Subdirectories?** | **Adds Deleted Files?** | **Adds Hidden Files?** |
| **git add .** | **✅ Yes** | **✅ Yes** | **✅ Yes** |
| **git add \*** | **❌ No** | **❌ No** | **❌ No** |

## Example: Subdirectories in Git

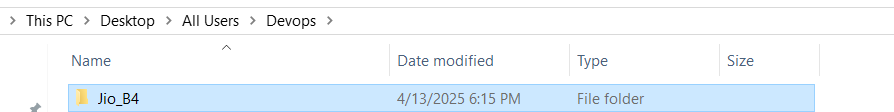
**Consider the following folder structure:**

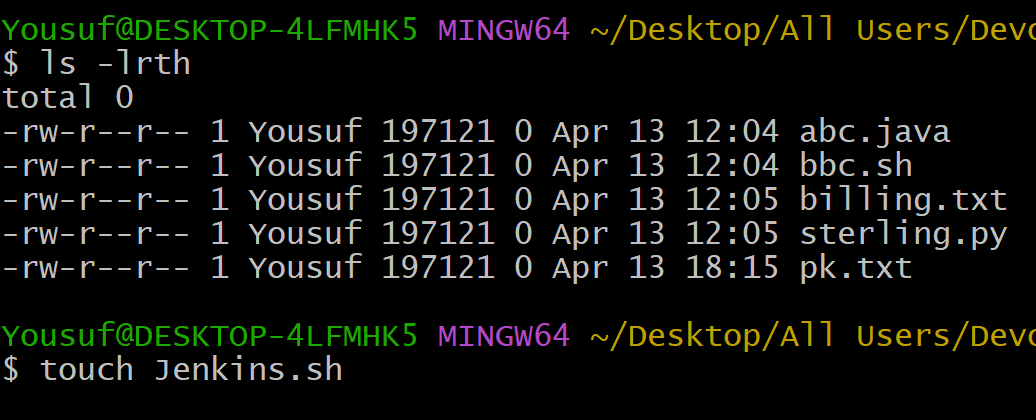
**my-project/  
├── index.html  
├── style.css  
├── script.js  
└── assets/  
 ├── logo.png  
 └── banner.jpg**

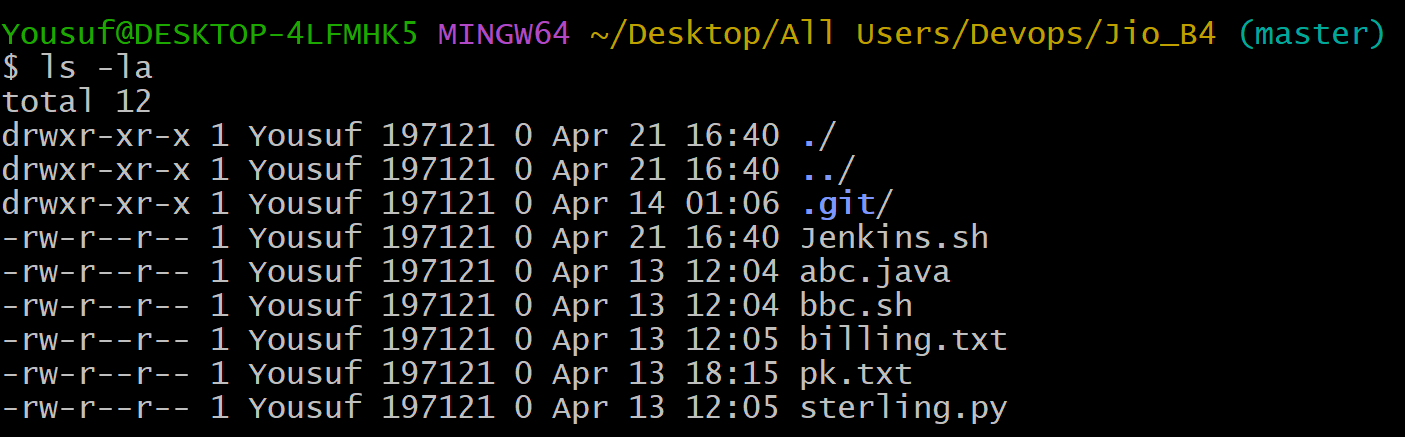
**If you are in the `my-project/` directory:**

**- `git add \*` will stage only: index.html, style.css, script.js**

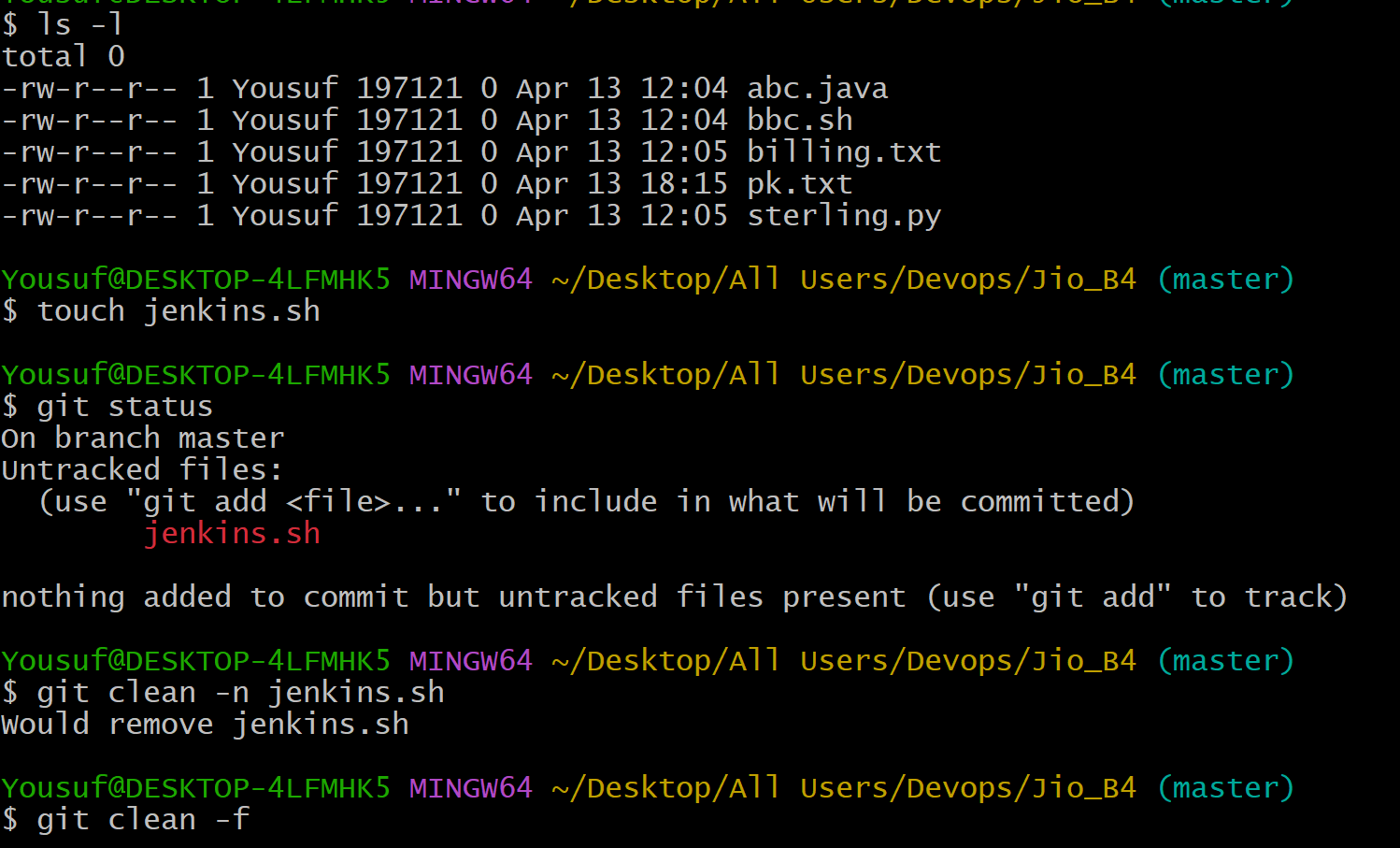
**- `git add .` will stage all of the above, including: assets/logo.png, assets/banner.jpg  
  
---------------------------------------------------------------------------------------------------------------------------------------**

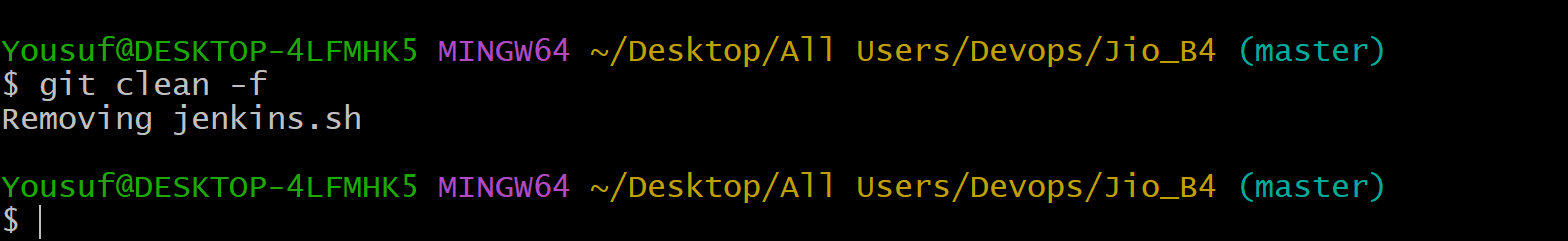
**Let’s create a file Jenkins.sh in the current directory Jio\_B4  
**

****

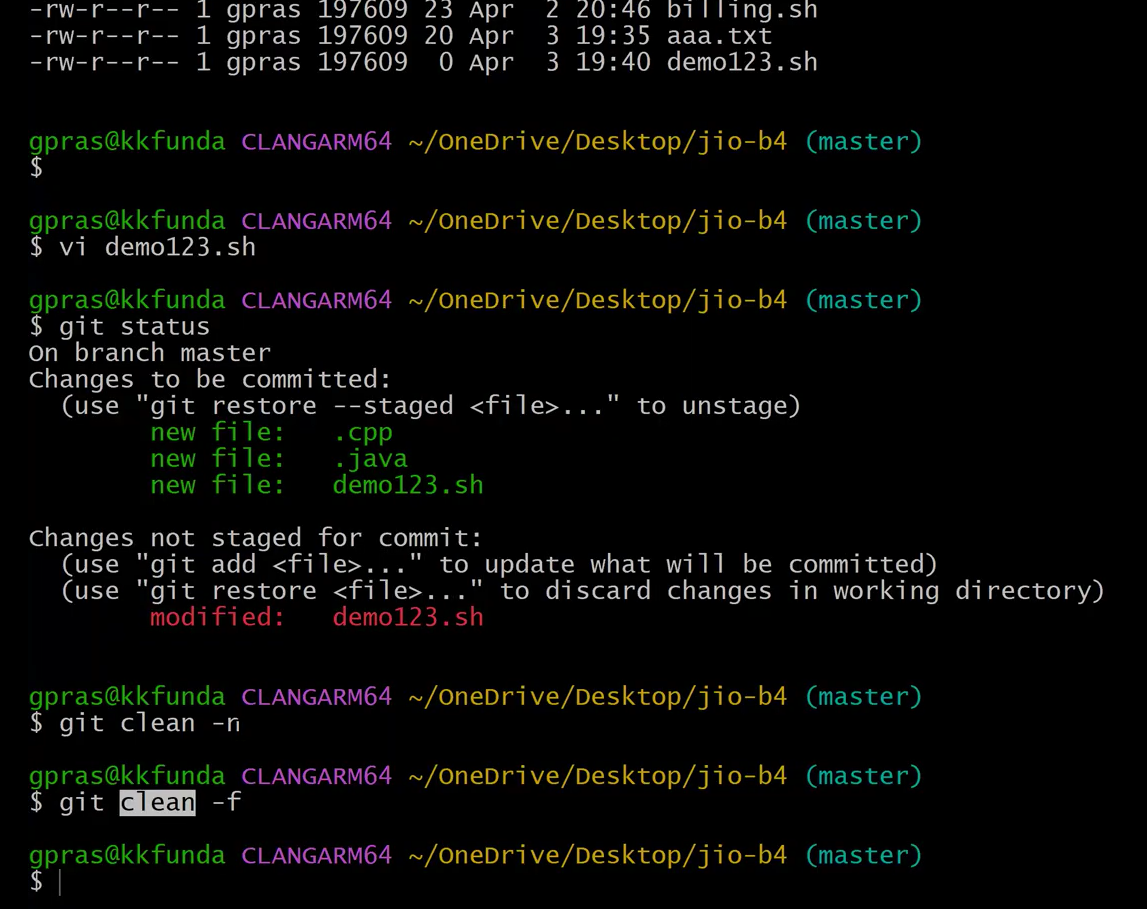
Command **ls -la** 

Now to delete the file Jenkins.sh we can use either the Linus shell command ‘**rm Jenkins.sh**’ or we can use the git command ‘git clean -n Jenkins.sh ’ and then git clean -f so the file Jenkins.sh will get deleted but   
this git command (git clean -n and git clean -f ) only works only for the untracked files it won’t work if the file is moved from working area to staging area.  
  
In the below example we can the process of deleting file using git command



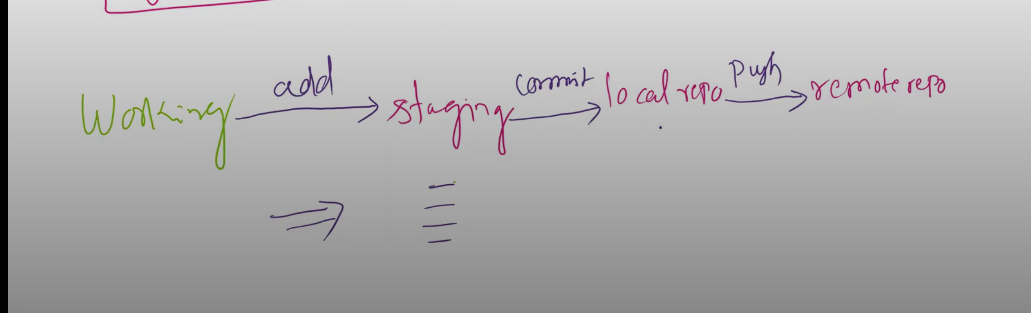
Press Enter  
  


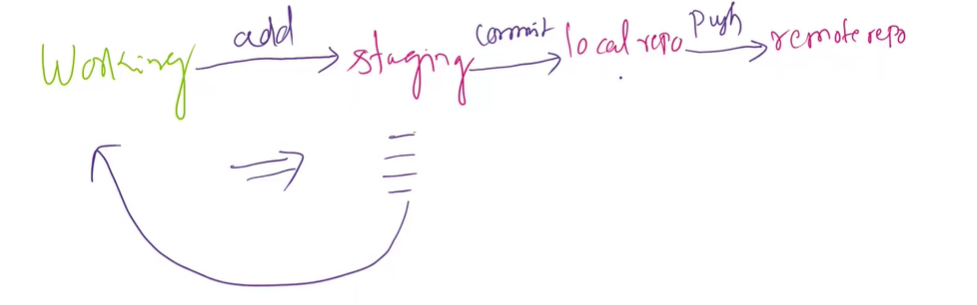
So the File Jenkins.sh(untracked file) is removed by using the git command from working area

Git clean command will help us to remove the untracked files but not the other files like Modified files in the below example it has been shown that   
  


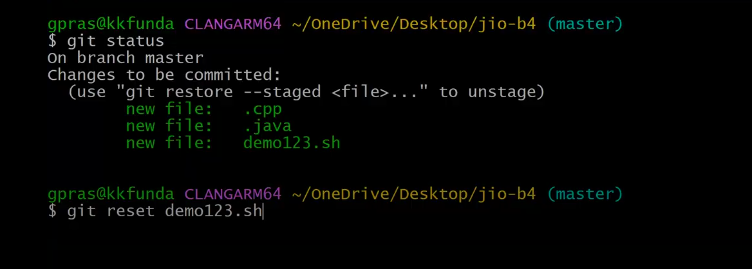
As you can see in the above image **git clean –n** demo123.sh and **git clean –f** these two commands are not working for modified file : demo123.sh

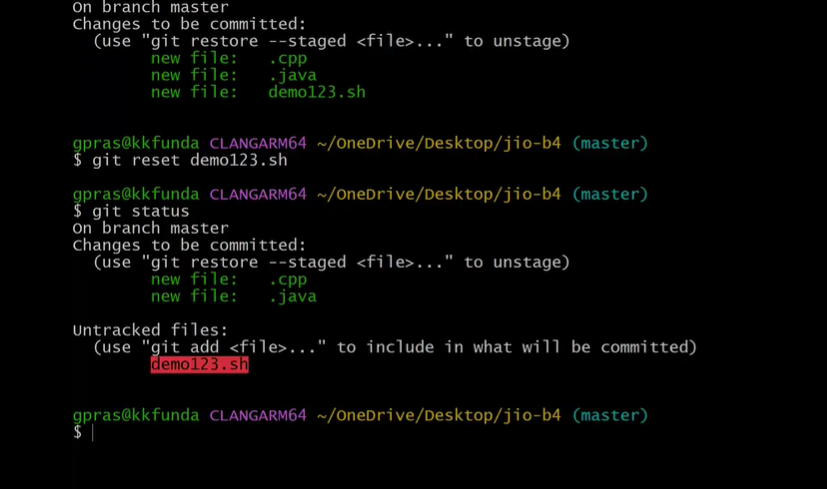
By mistake I have moved the files from working area to staging area and now I want to revert it so is it possible to get the files from staging area to working area?



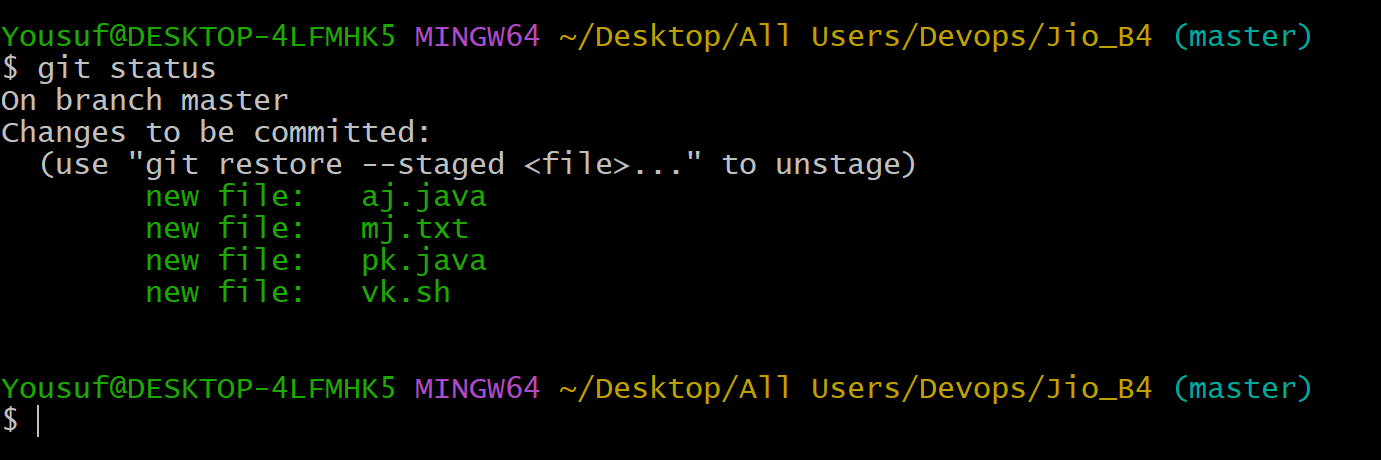


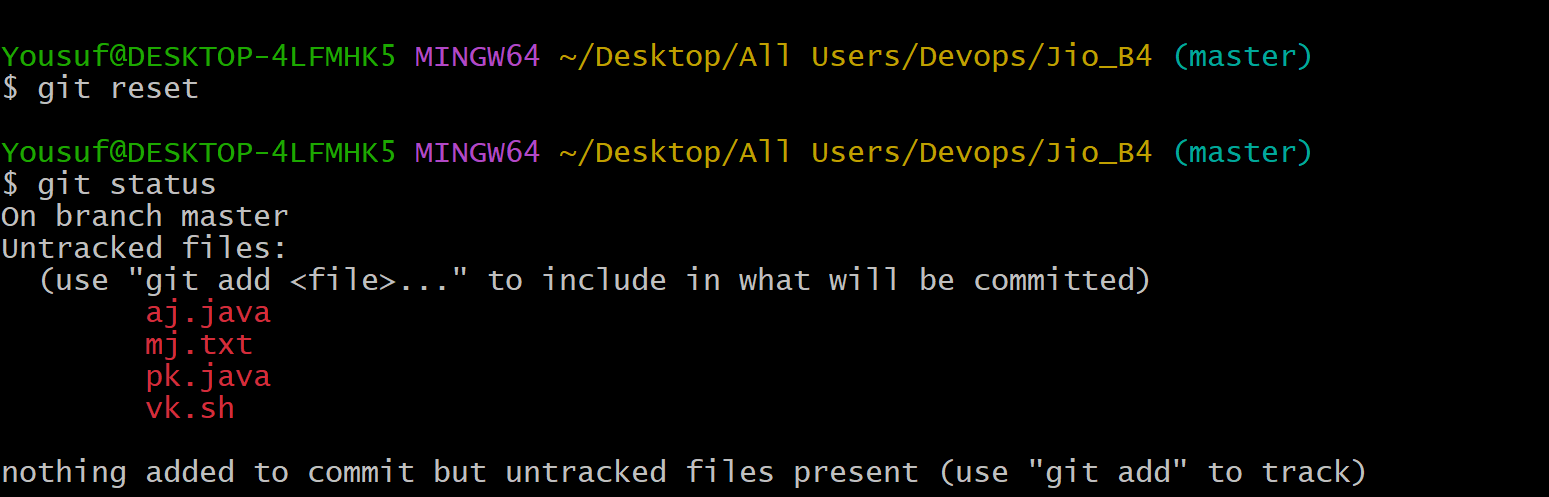
* Ans ) Yes! It’s possible to get back the files from staging Area to working Area by using the command ‘git reset filename’  
    
  shown in the below example

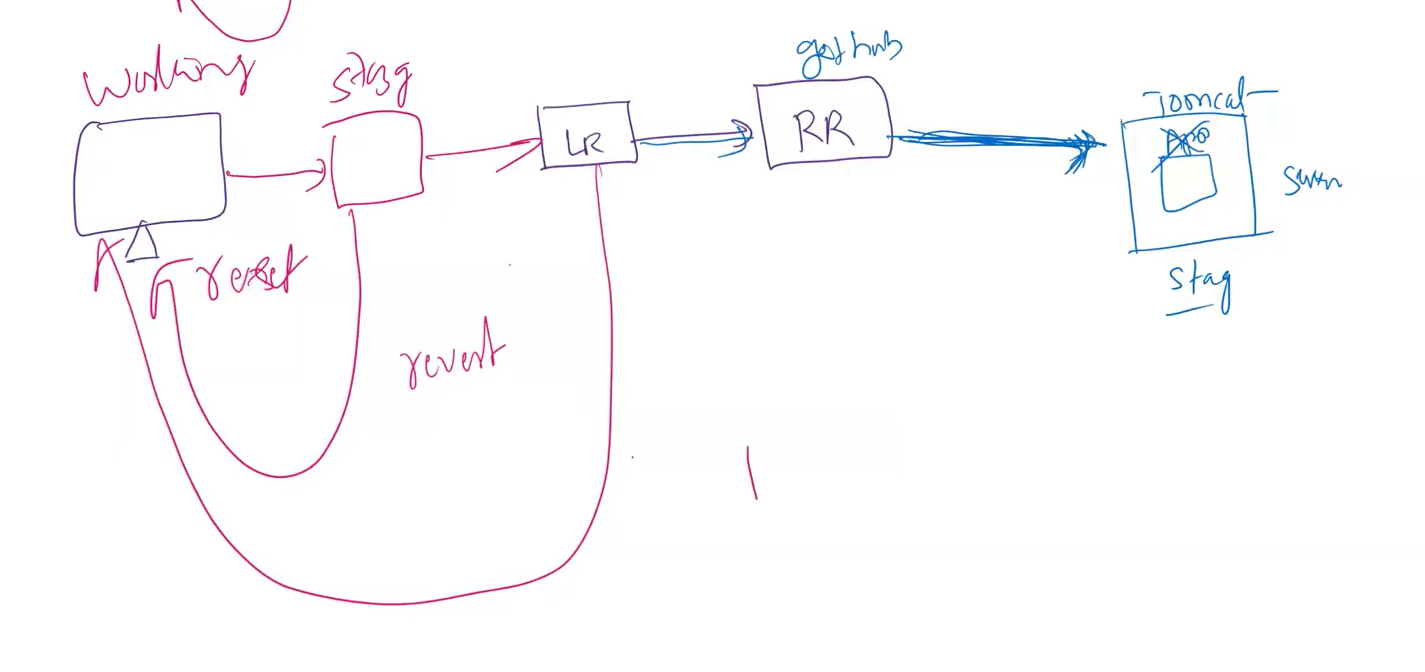




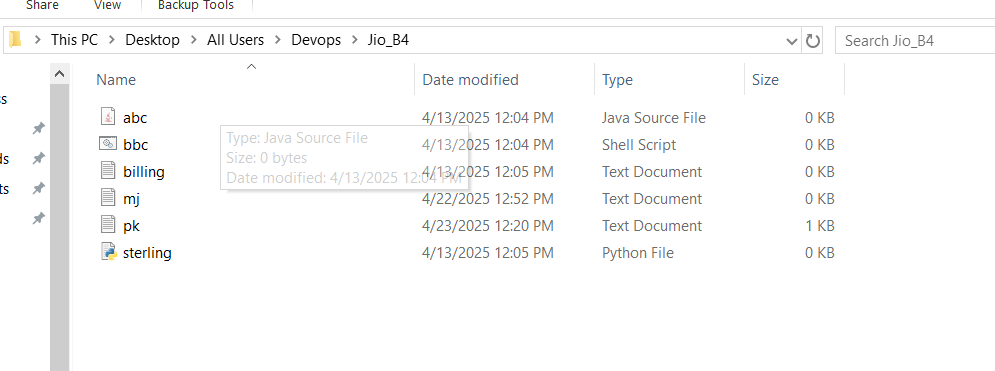
To revert all the files from staging area to working area we have to use the command ‘**git reset**’



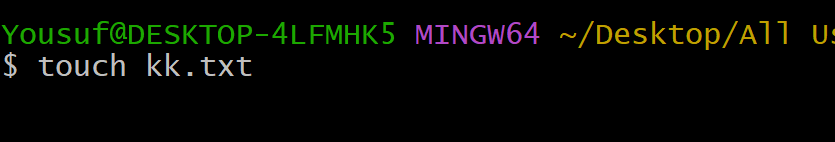


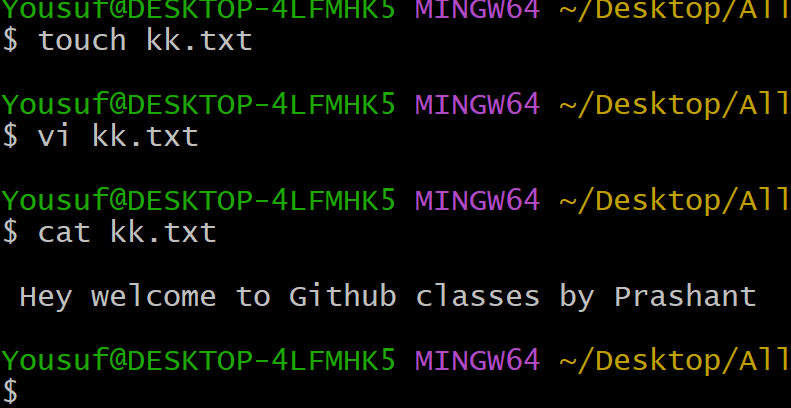


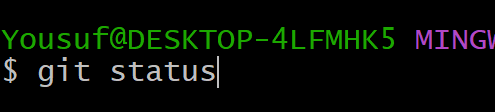
So far we have studied about the command ‘git reset’ which is use to send back the file from staging area to working area.  
  
Now let’s have a look with the next command ‘git revert commit id’

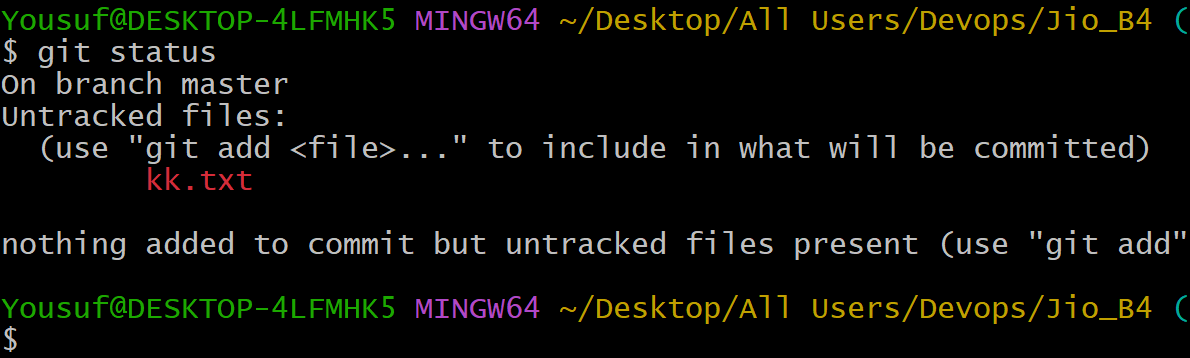
So to delete file or group of files that are associated with recent commit id present in Local repository we have to use the command ‘git revert with commit id ’  
  
Let’s see git revert with an example  
so create a file with the name kk.txt  


Open this folder Jio\_B4 in Git Bash and create a file KK.txt

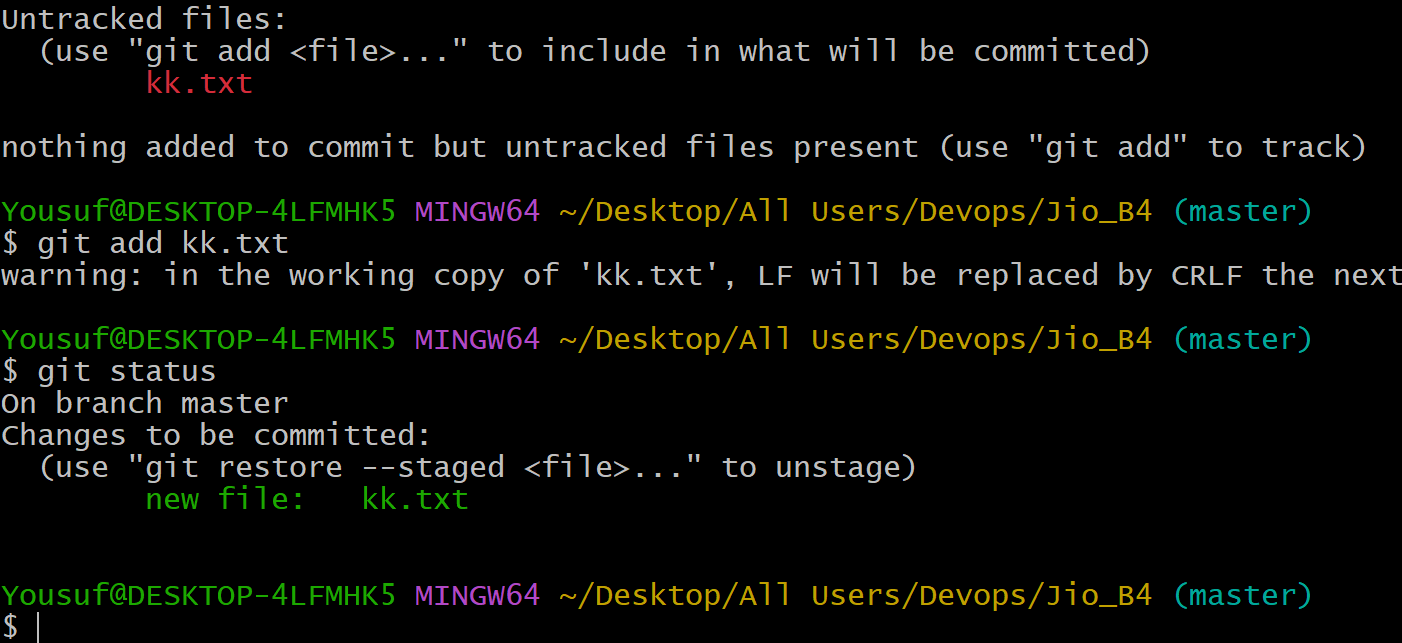


Add some data in to the file KK.txt  
  


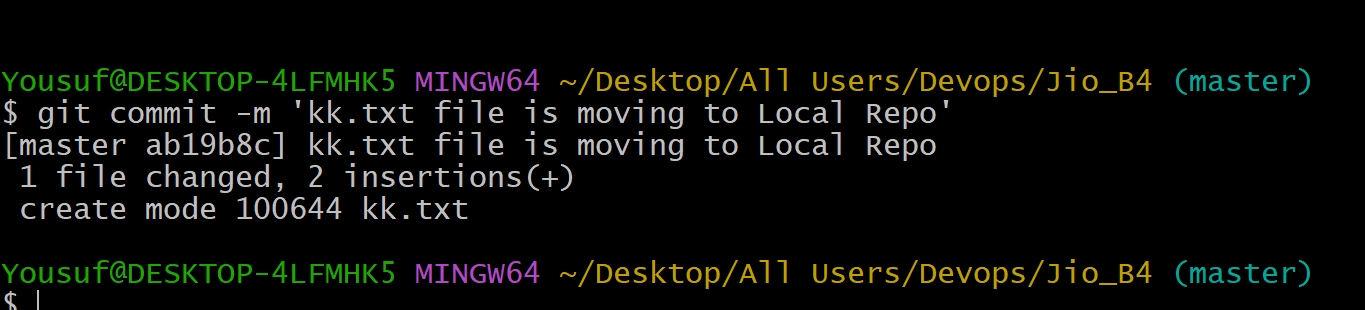
Now let’s move this file KK.txt from working area to staging area  
check the status first   


Press enter  


Execute the command ‘git add’ to move the file from working area to staging area



Now move kk.txt to Local repository using the command ‘git commit –m ‘



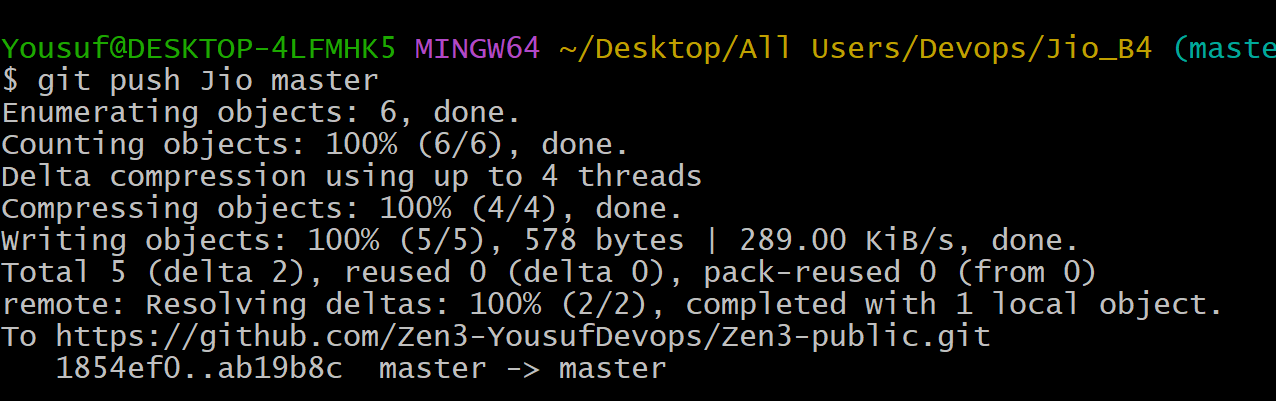
Execute the git log -2 to check the recent two commits with commit Ids

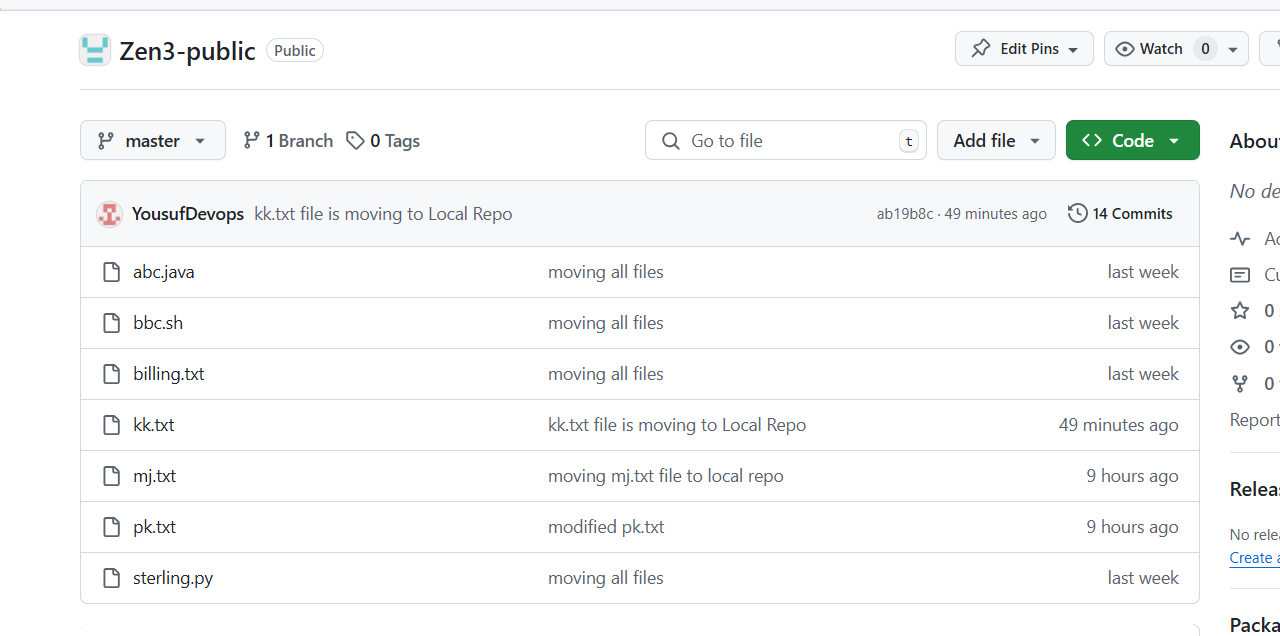


Now copy the commit Id shown in the below  

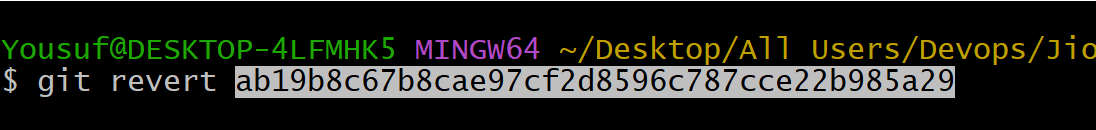

Now push the Jio\_B4 directory to the Remote Repository using the command

‘git push Jio master’ & press enter  
  
In the below image you can see directory is completelycopied to Ze3 repository of My github account

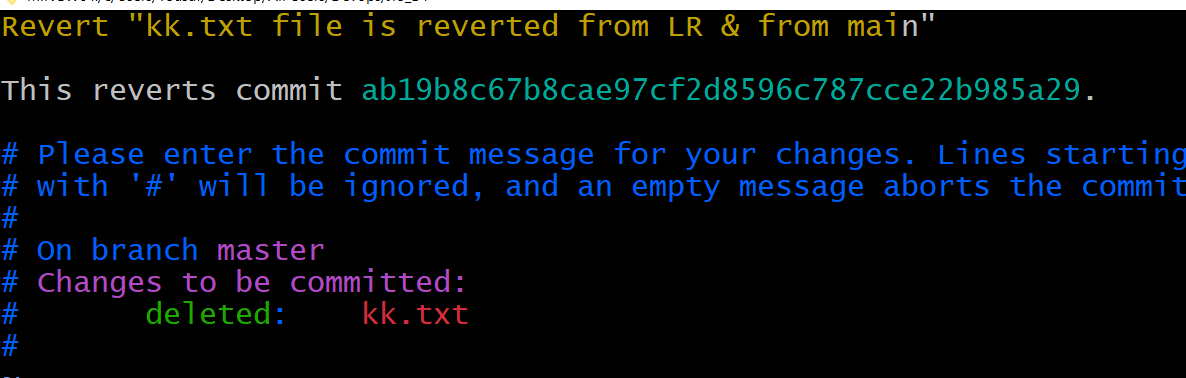


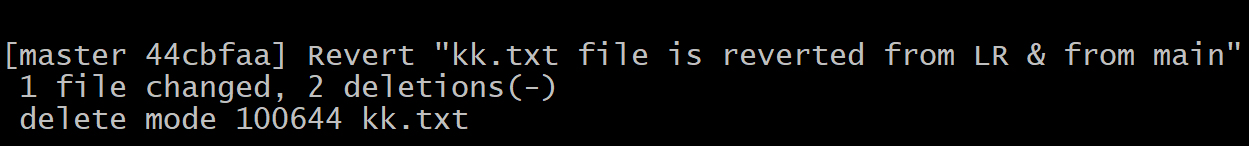
Now let’s check the github remote repository and let’s see the File kk.txt is added or not  
  


As you can the kk.txt is added  
  
Now In order to remove the kk.txt file from local repo and as well as main directory we have to execute the command ‘git revert commit id’

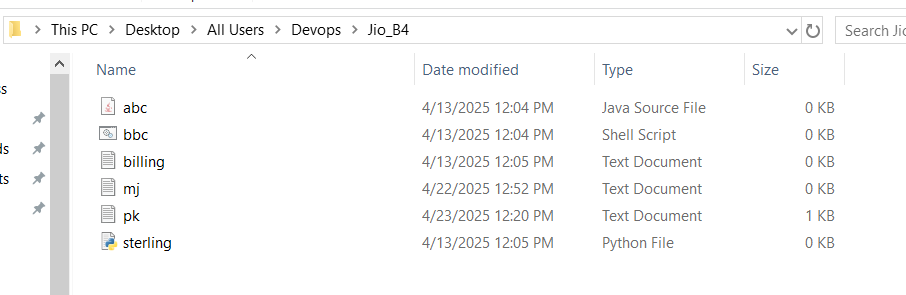
So execute git revert command  


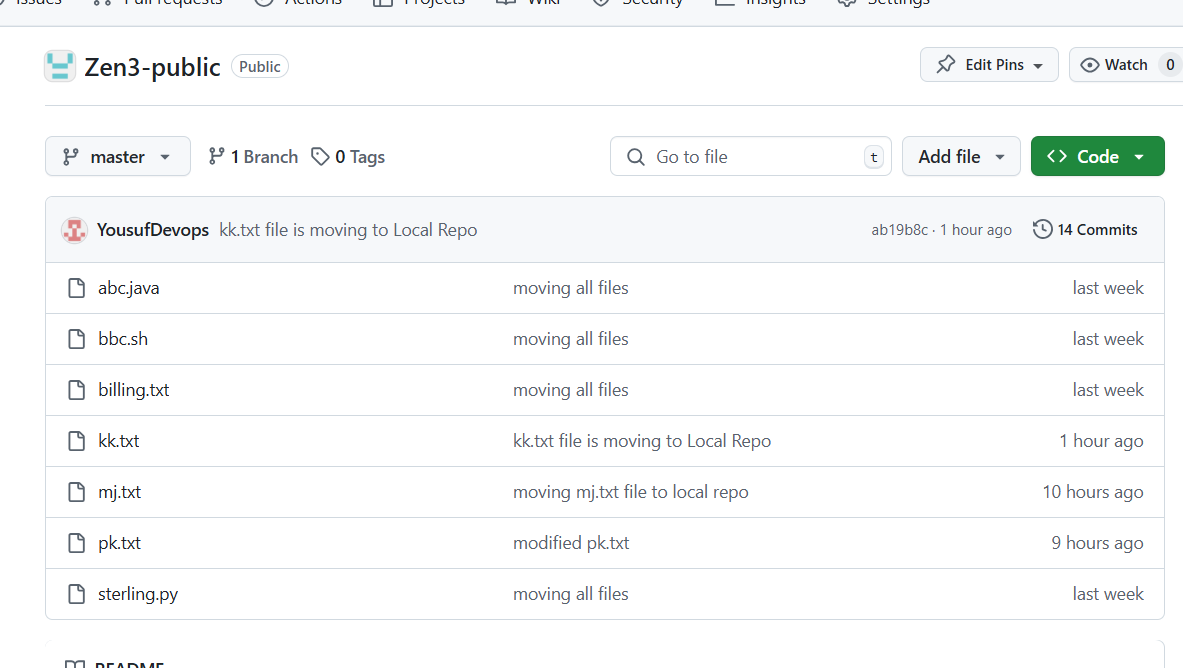
Press enter

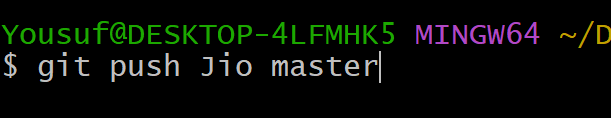


Press ESC & type :wq  
  
check the message saying file is changed and data is deleted  


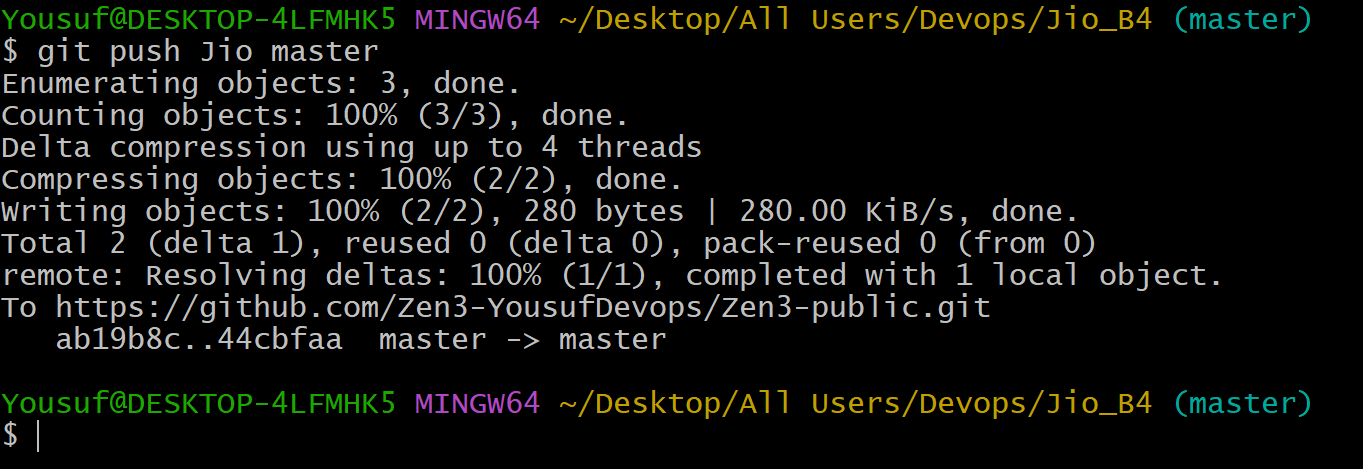
Now check the main directory and github repository  
  
In Main Directory ‘jio-B4’ the file kk.txt is not available

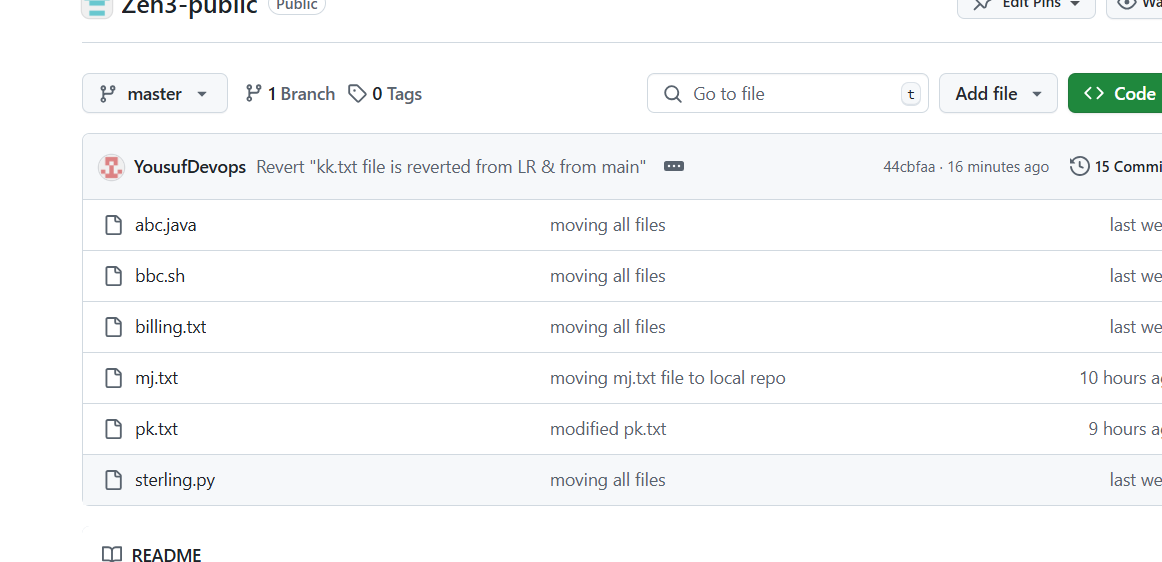


But in Github Remote Repository kk.txt is available didn’t get deleted check the below image   


The reason is because it was not updated   
we have to update the Remote Repository by using executing command   
git push Jio master

Press enter



Now check the remote repository and as you can see the changes can be seen in Remote Repository the kk.txt is not there  
  


Now we revoke the deleted kk.txt file again by using the command