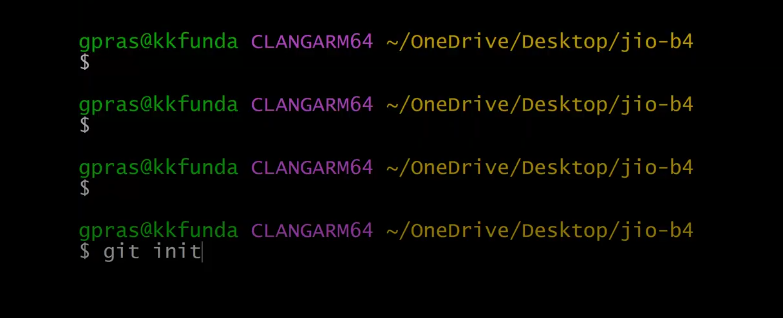
GItHub   
  
Git Commands :  
  
In this class we study How to move the file from local computer to Git Repository

Create a Folder on the desktop “Jio-b4”  
  
Now open the folder Jio\_b2 in Git Bash and type the first command “ git init “  
  


The above command git init Initializes a new Git repository in your current directory.  
  
🡪 Creates a hidden .git/ folder in your directory.

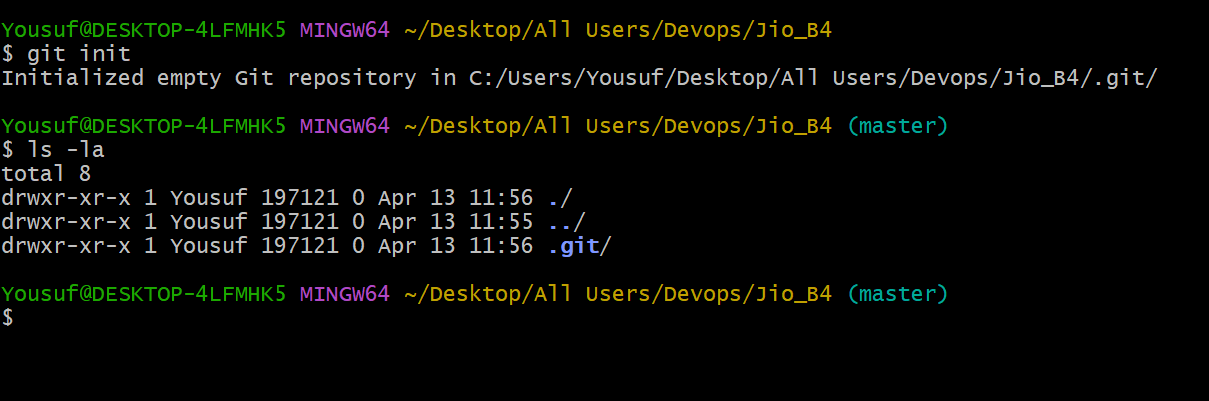
🡪This folder contains all Git tracking info (like commits, branches, config).

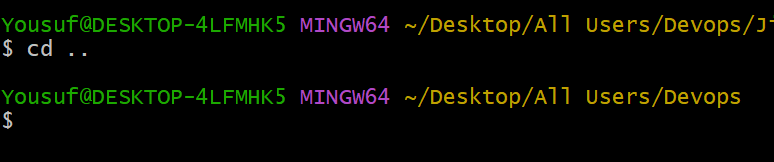
🡪 After this, your folder becomes a **local Git repo**, ready for staging, committing, and pushing.

After executing git init command , you will see a confirmation message similar to:

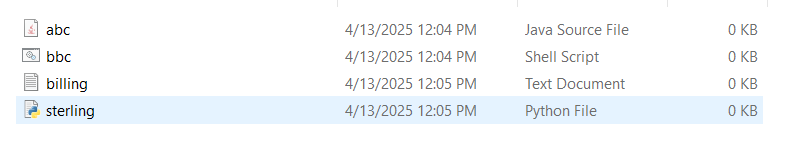
Initialized empty Git repository in ~/Desktop/Jio-b4/.git/

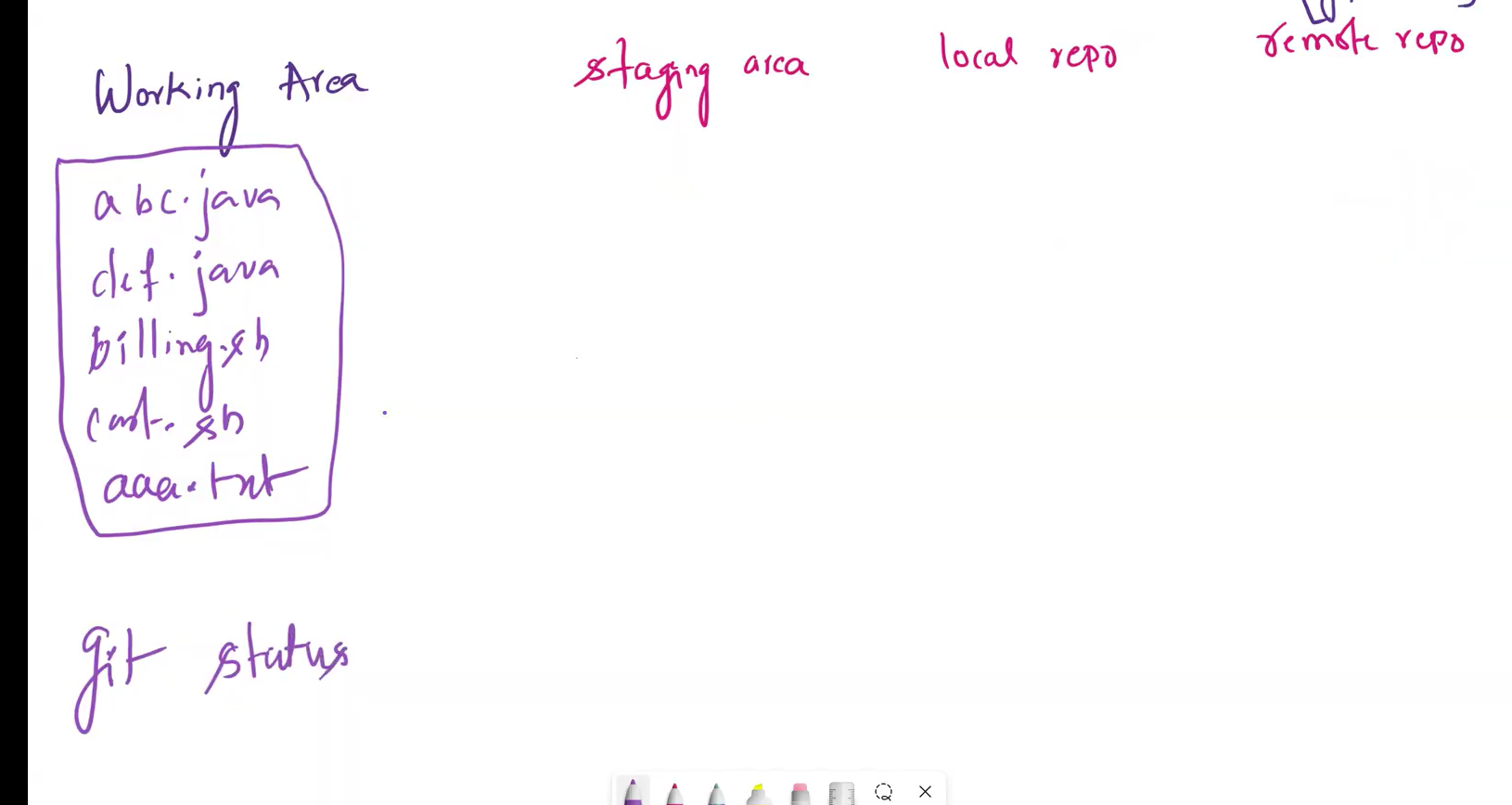
A hidden folder named `.git` is created within the directory.



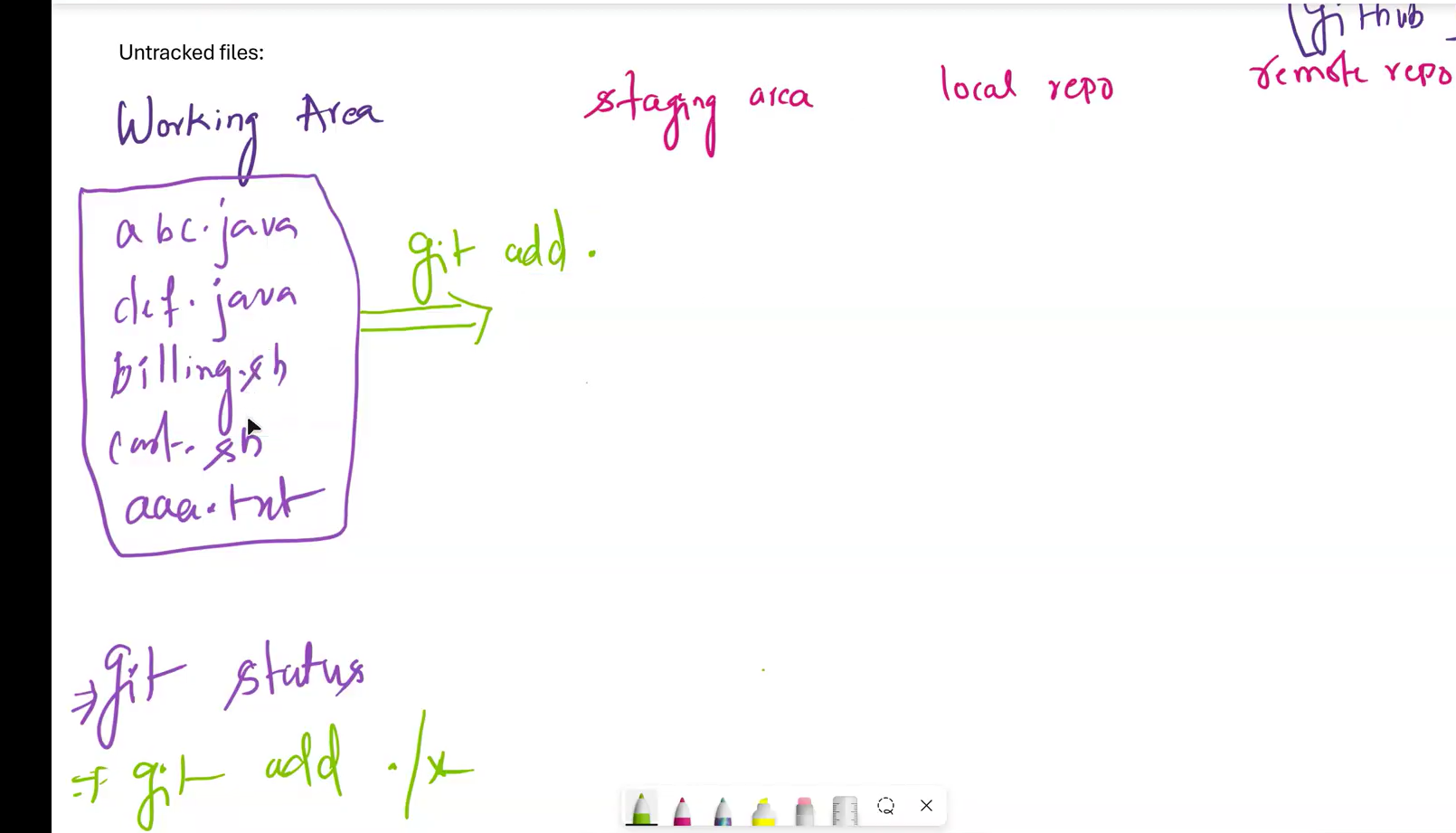
Now execute the command **cd..** to get back to the folder Jio\_B4  
  


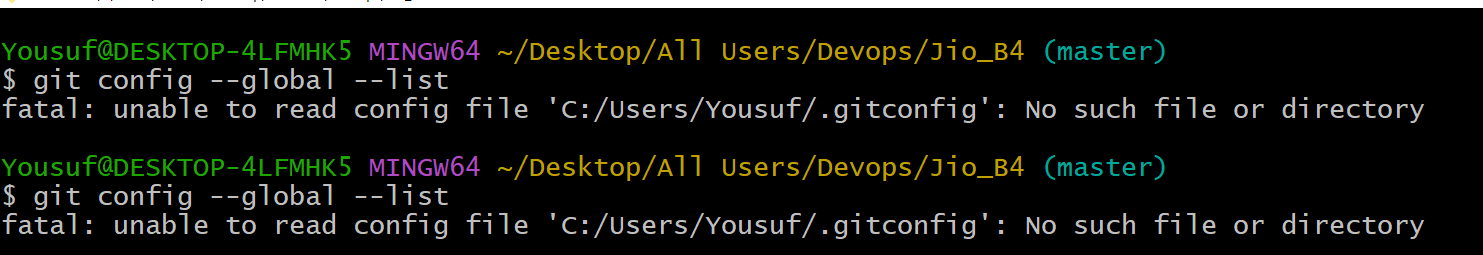
After excuting the command “git init” a **.git** directory would be installed   
  
Now in the folder **Jio\_B4** create some files with .java , .txt , .py , .sh extensions.



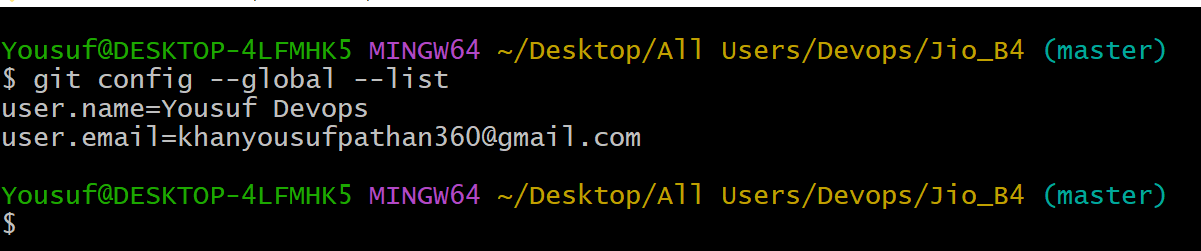


Command **“git status**”   
The **git status** command is used to **check the current state of your working directory and staging area**.  
  
By using this command we can find the current status of the files whether it is in ‘working area’ or in ‘staging area’  
  

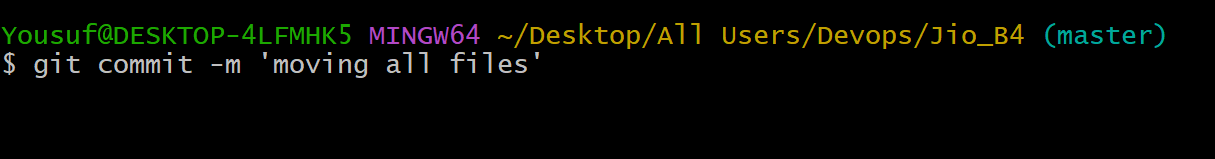
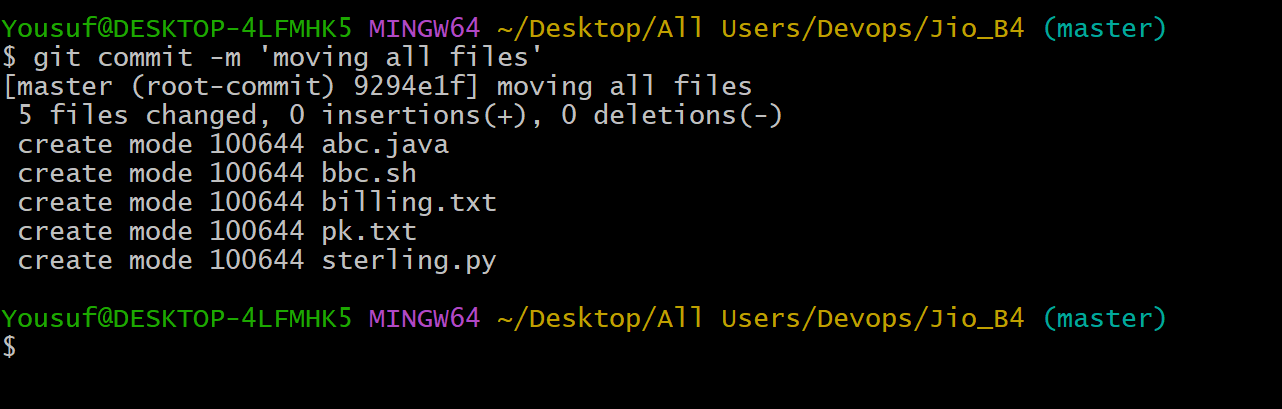

To move the files from ‘working area’ to ‘staging area’ we have to execute the command ‘git add. (or) git add \*’  
  
Then all the files will be moved from working area to staging area  
  
  


As you can see the below image after executing the command git add \* all the files present in ‘**working are**a’ is moved to ‘**staging area’  
  
  
  
Now** all the files are in staging area have to move from **staging area** to **Local repository** but we have to set up a configuration first before moving the files from staging to Local Repo  
  
Execute :: **git config --global --list**-------------------------------------------------------------------------------------------------  
The result would empty  
  
  
  
As we haven’t set up the global Git config (like your name and email), so the file .gitconfig hasn't been created yet.  
  
Set up your global user info:

git config --global user.name "Yousuf Devops"

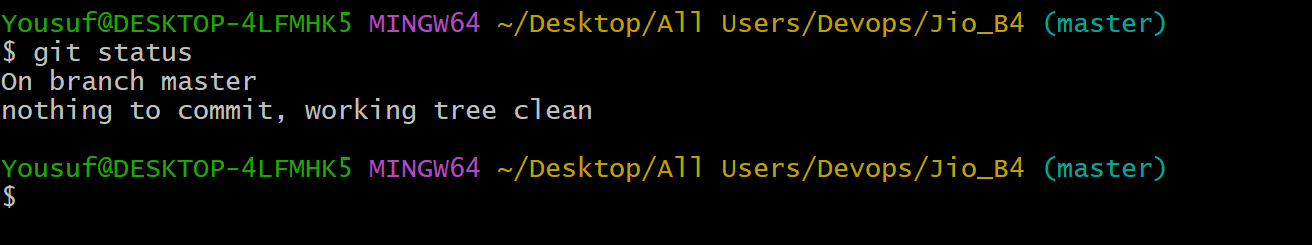
git config --global user.email [khanyousufpathan360@gmail.com](mailto:khanyousufpathan360@gmail.com)  
  
In the below image it is hsowing that we perfectly configured out github credentials & it’s a one time activity  
  


Now Let’s try to move the files from “staging area” to “Local Repository” for that we have to execute an command **‘git commit –m message’**

Execute the command : **git commit –m ‘moving all Files’**  
  
  
  
After executing the command **git commit –m ‘moving all Files’** check the below image  
  
  


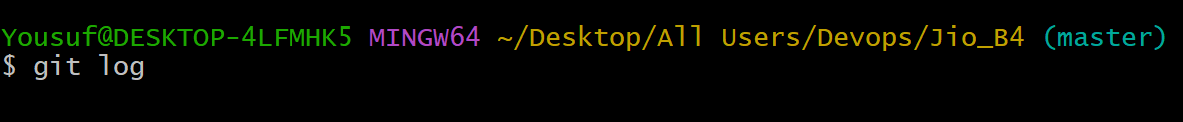
### git commit –m ‘moving all Files’ 📘 What it does:

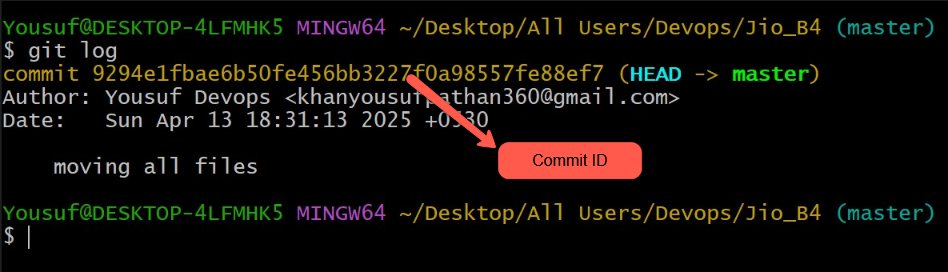
* git commit creates a snapshot of your changes.
* -m adds a commit message: 'moving all Files'.

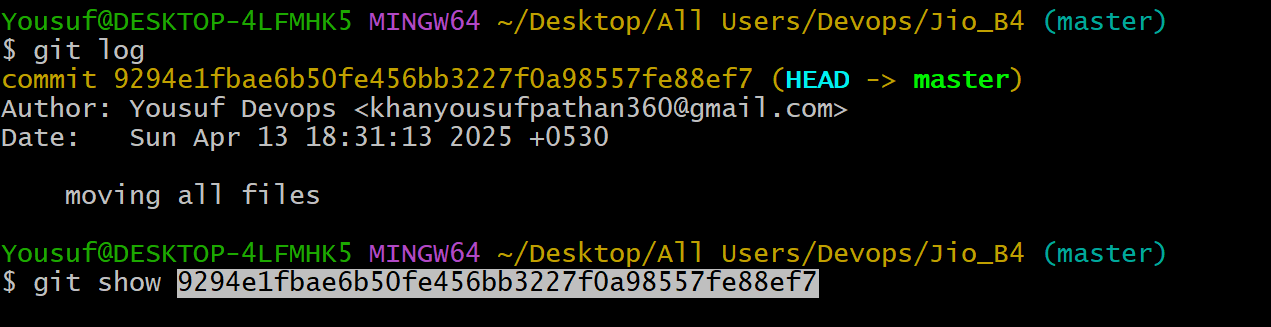
Now check the status The working area(working tree clean) would be clean  
  
git status: 

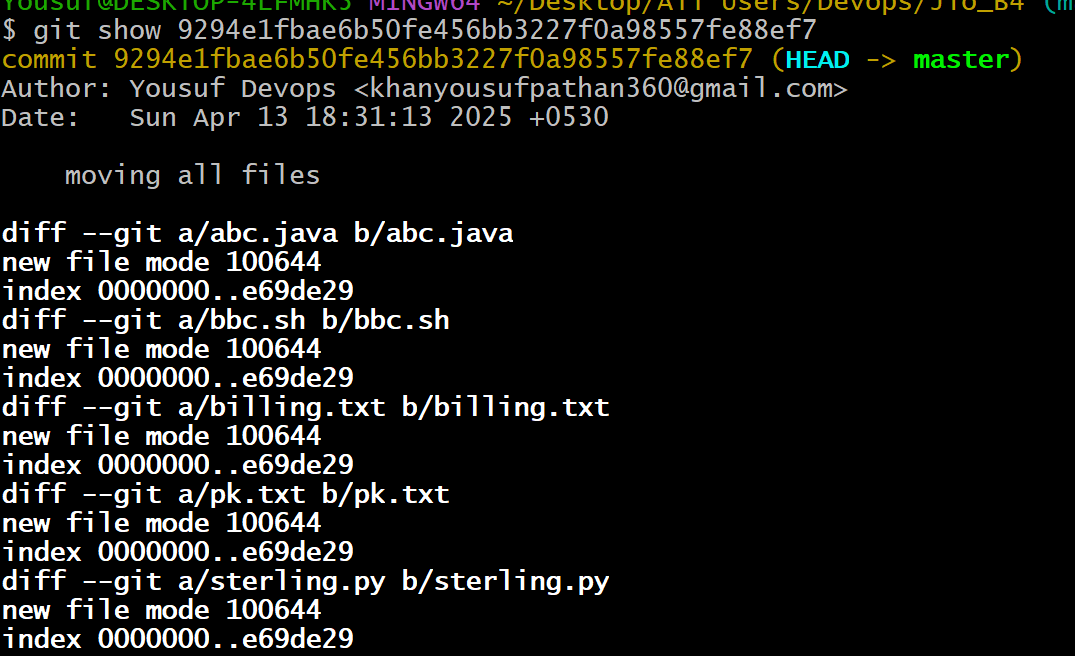
A commit ID get generate for the moved files from staging area  
  
That commit ID associated with the message “moving all Files”  
  
The files are moved from staging area to Local repository  
  
To list commit Id we have to execute the command ‘ git log ‘ command  
  
Interview question :: How would you list the commit IDs?

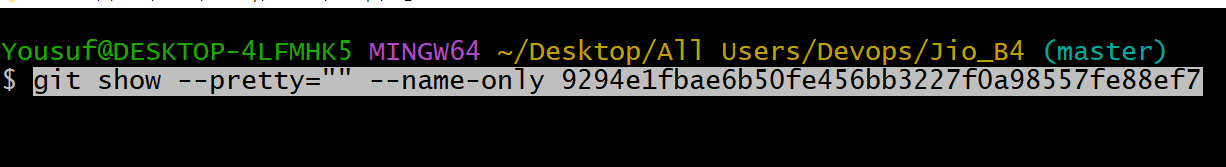
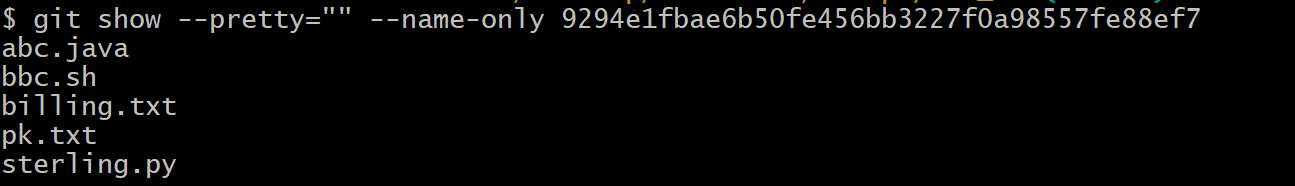
Ans) By executing the command ‘git log’ we can list the commit Ids





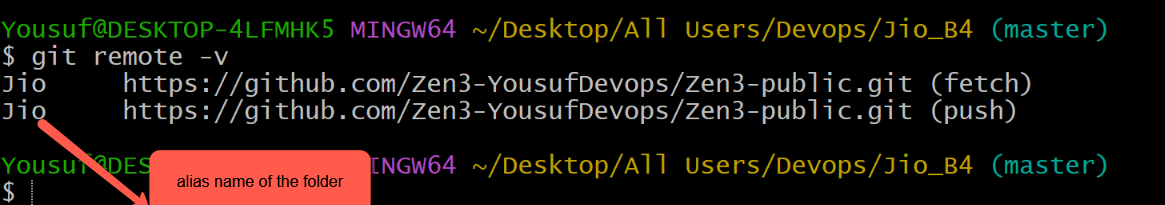
In order to find out the files that are saved in commit id then we have to execute the command   
  
git show commitid  
  
example **: git show 9294e1fbae6b50fe456bb3227f0a98557fe88ef7**

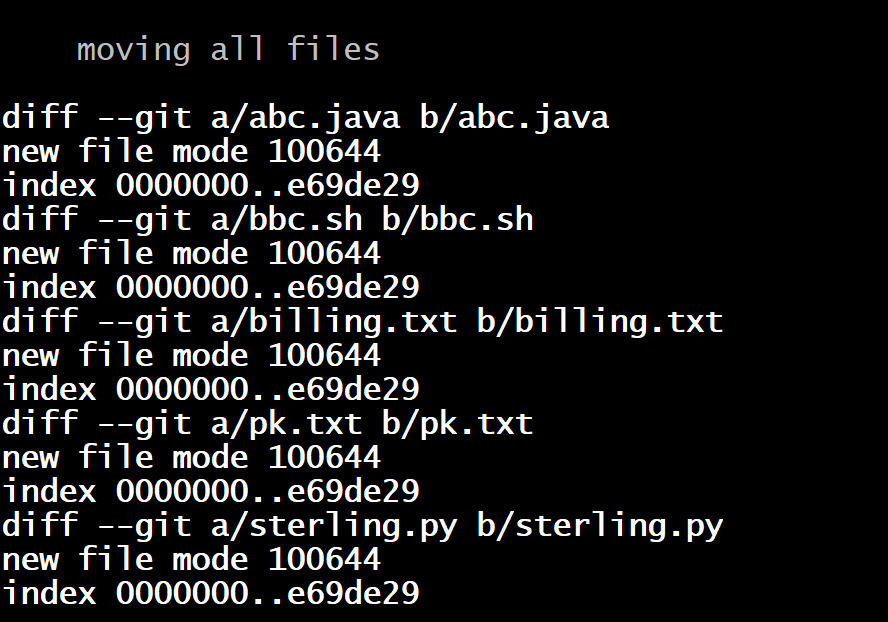
**Press Enter**

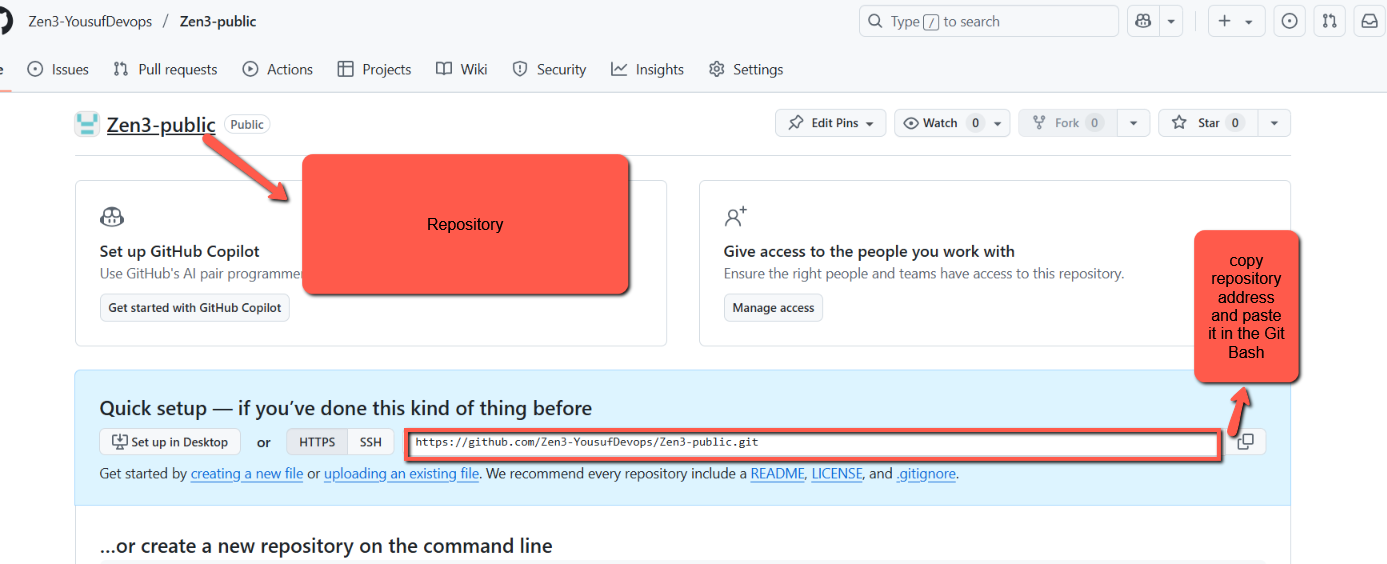
**git show --pretty="" --name-only 9294e1fbae6b50fe456bb3227f0a98557fe88ef7**This command is used to **see which files were changed** in a specific commit (without showing their contents). It's helpful when auditing or reviewing code changes.  

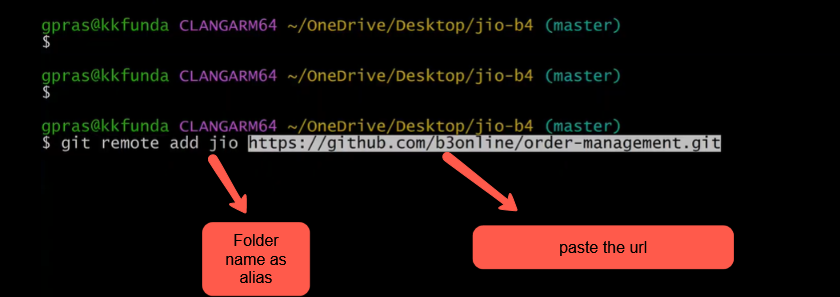
### Now let’s try to move the files from Local Repository to Remote Repository to check the folder alias name we have to execute the comand git remote –v

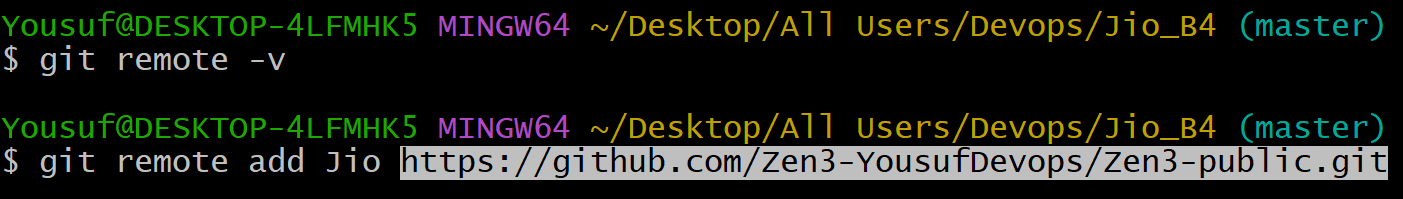
### What it does:

* Lists the **remote repositories** linked to your local Git repository.
* The -v flag stands for **verbose**, so it shows the **URLs** associated with each remote name.
*   
    
  Now lets move the files from Local Repo to Remote Repo

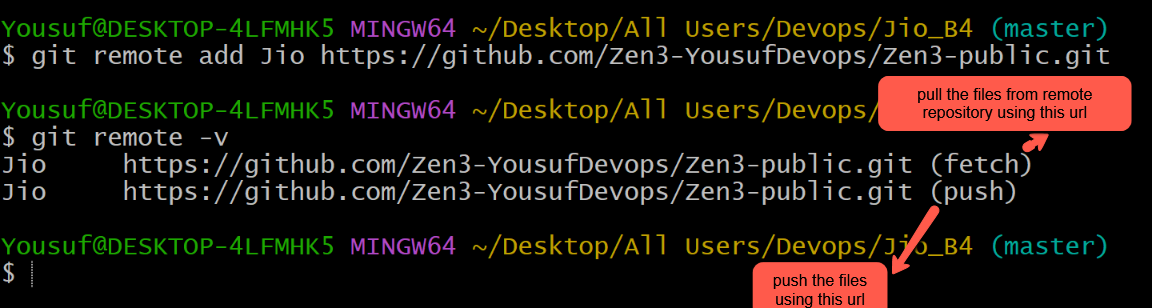


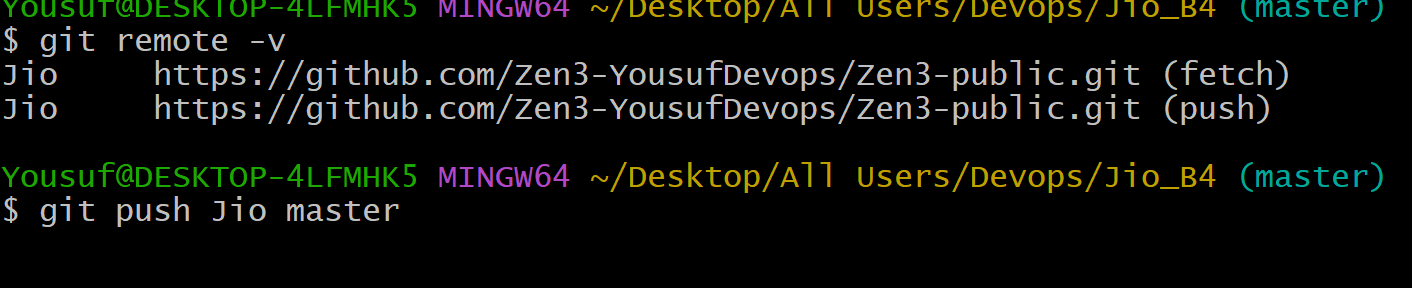
Let’s push the above files in to Remote Repositotry So first copy the repository address from your github account  
  


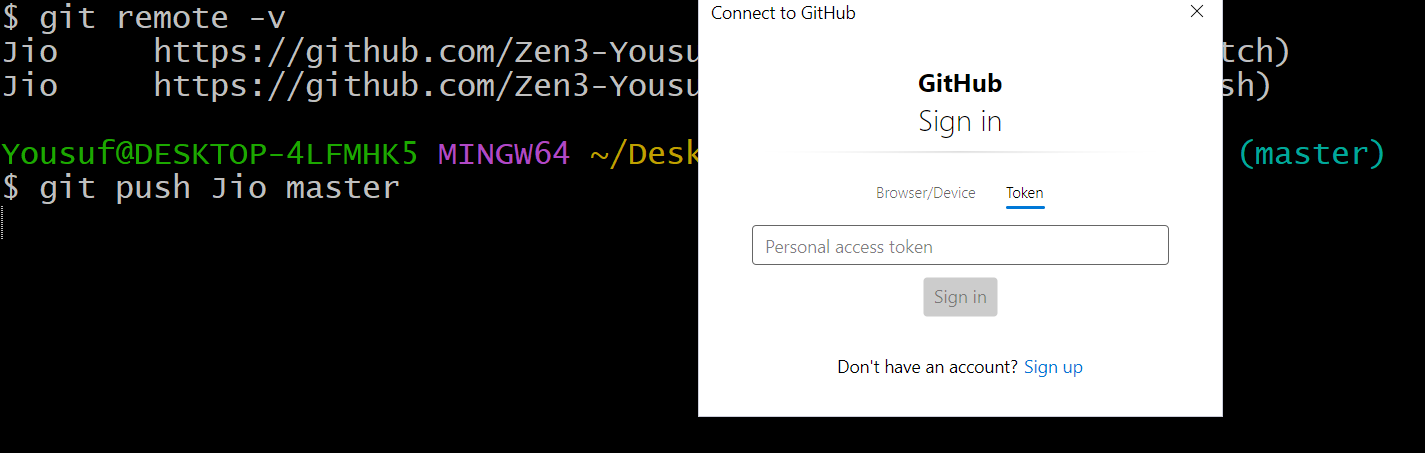


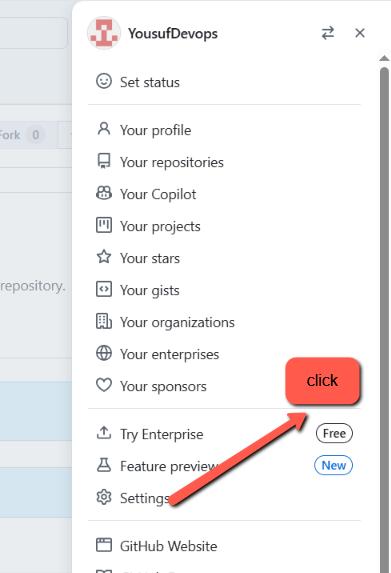


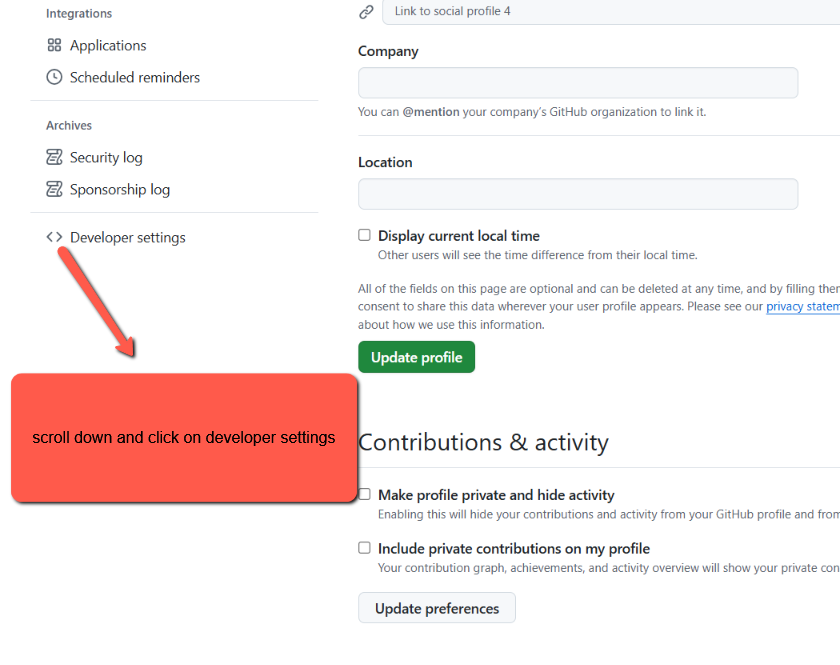
Press enter

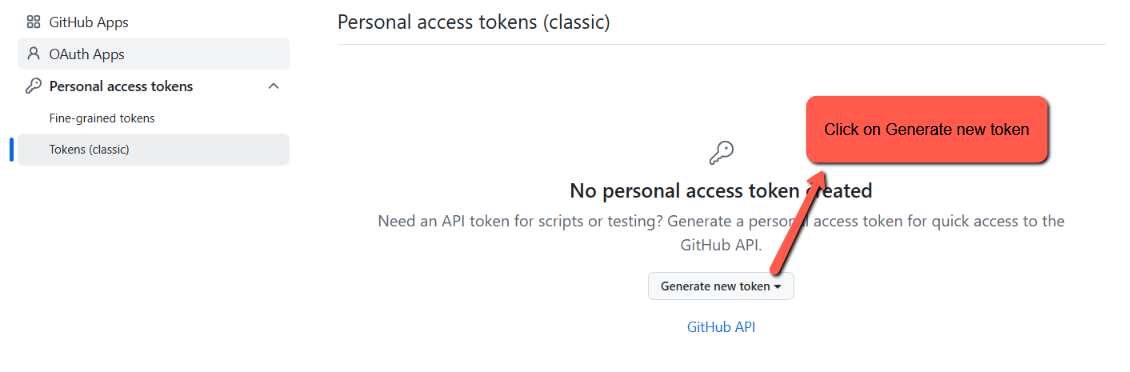


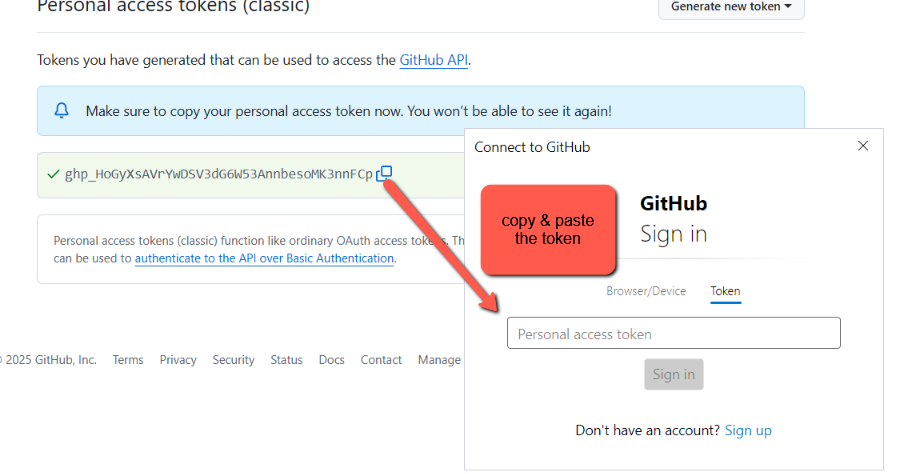
Zen3-public is the remote repository . so we are trying to push the files from Jio-B4 (local repository) to Remote Repository ie (Zen3-public) Lets run the command   
  
“ git push Jio master “ to push the files from Jio\_B4 in to Remote Repository  
  


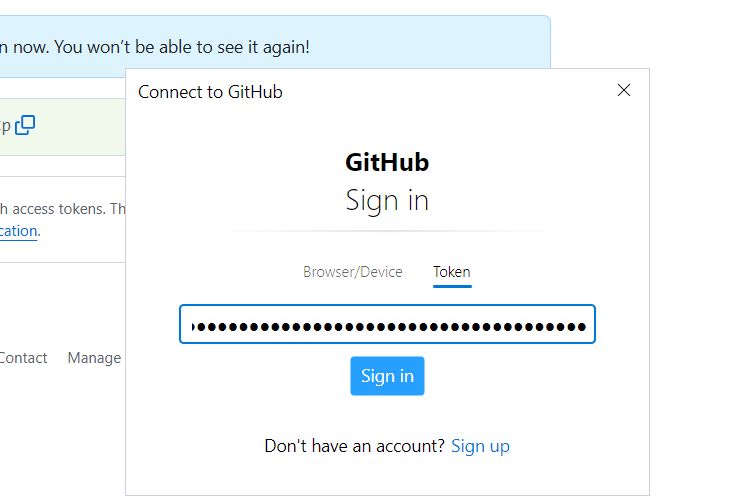
It will ask for personal access token   
  


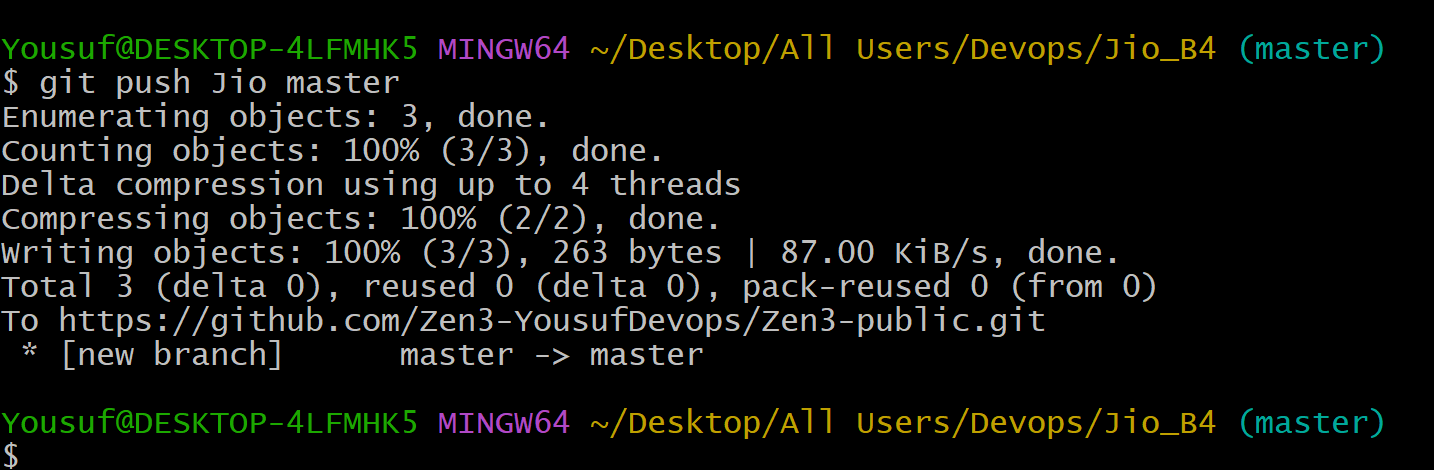
So Let’s generate the access token from the github account  
  
1. Click on the profile and click on the settings  
  


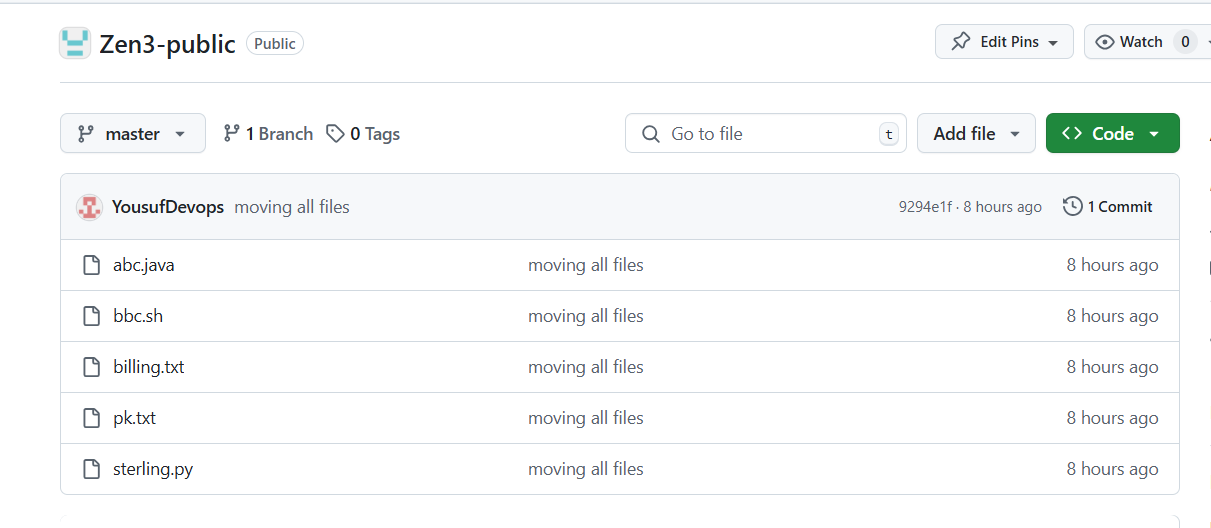
2. 

3. 

4. 

5. click on sign in

6. Files were sent to GitHub successfully check the below message saying counting object 100%.

Check the github repository .voila the files are successfully copied in to the remote repository Ze3-public  


* We can move modified files **directly from working area to local repo** using git commit –am
* ❗ But only if the files are already tracked (i.e., were added previously using git add or committed before)  
    
  or we can move the modified file in traditional way by using the command
* git add filename
* git commit -m "your message"
* use the command git push Jio master to update the Remote Repository

|  |
| --- |
|  |

We will discus about (master) branch in the upcoming session  


Standard Up calls starts from :: 1:16:00 please check the video