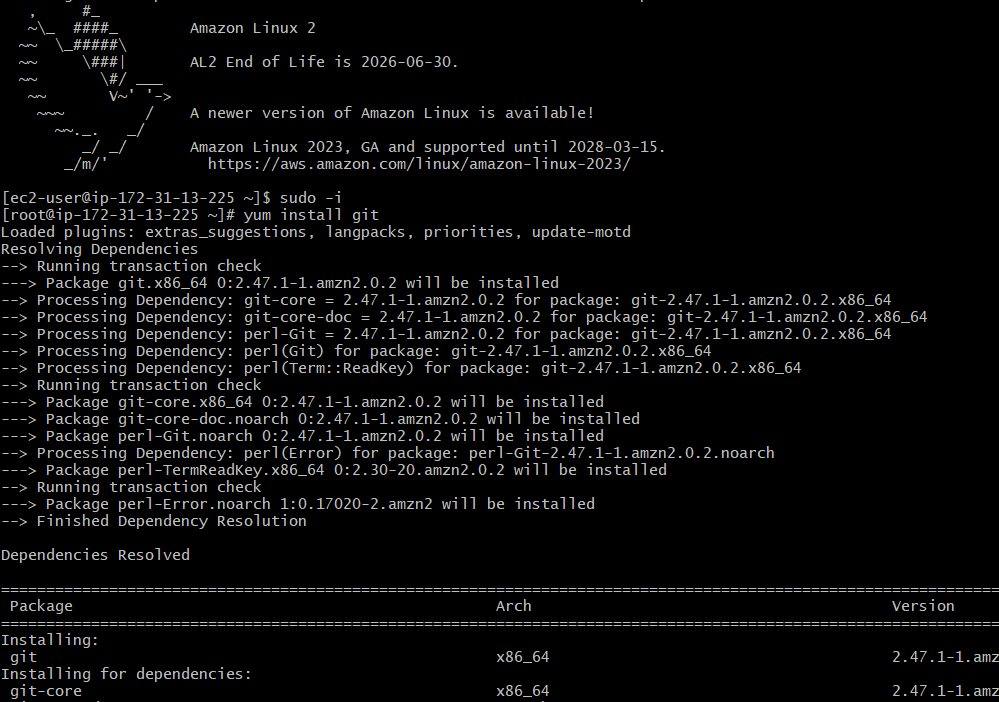
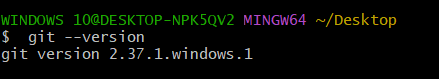
**GIT & GIT-HUB Task -02**

**1)Install git.**

**yum install git** – To install git in server.

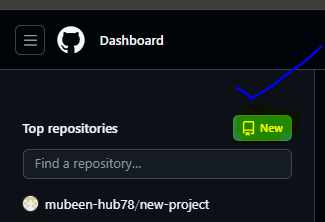


Mostly we have Git inbuilt in **windows**, if it is not available we can download it from browser and we can install it and we can check it in gitbash.  
  
**git --version** – command is used to check whether git is installed or not. For borth server and windows(Gitbash).

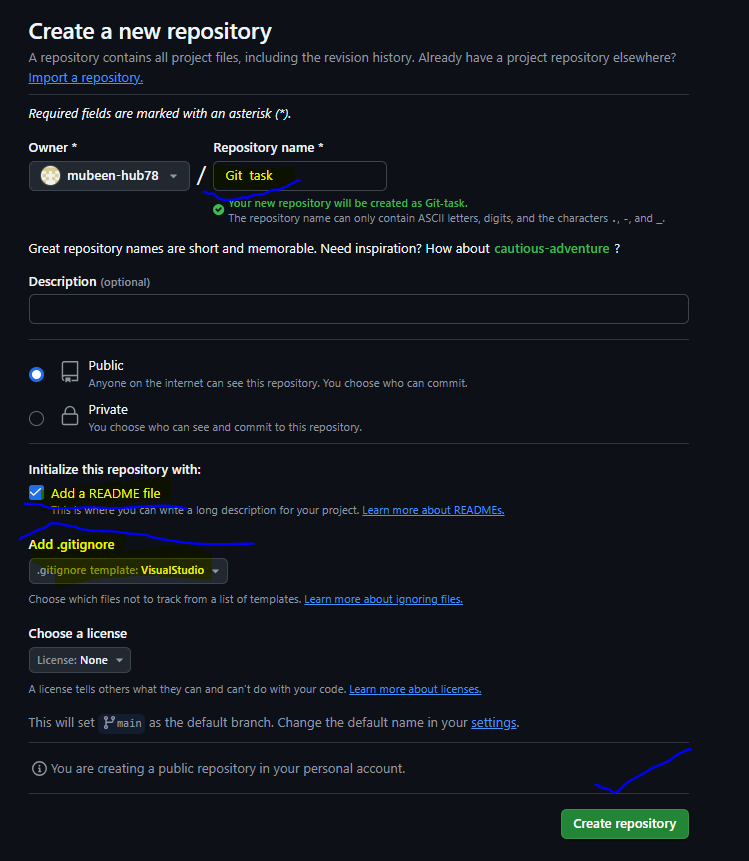


**2)Create a repo in github with README.md and .ignore file.**

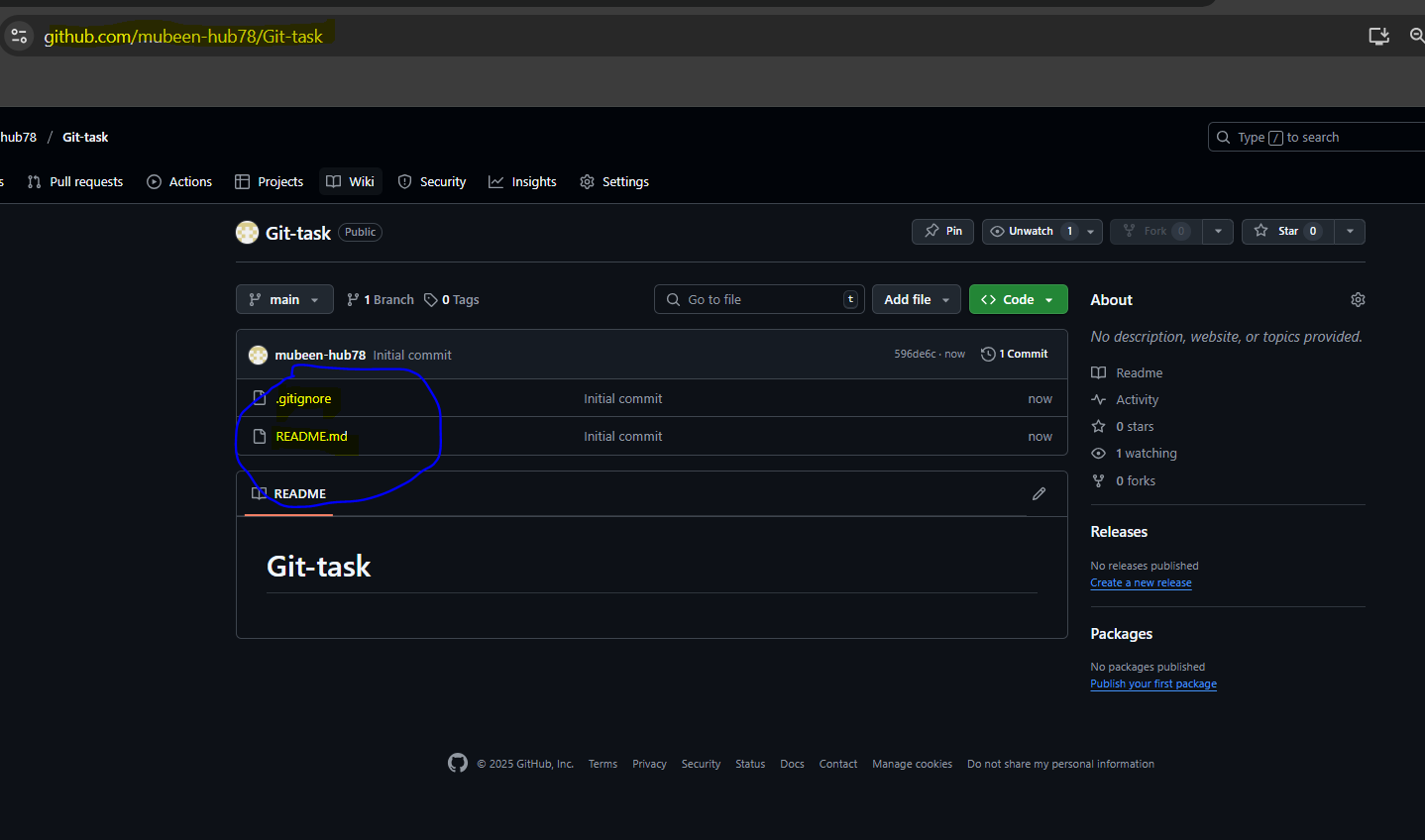
In the dashboard click on **New** to create a Repository.



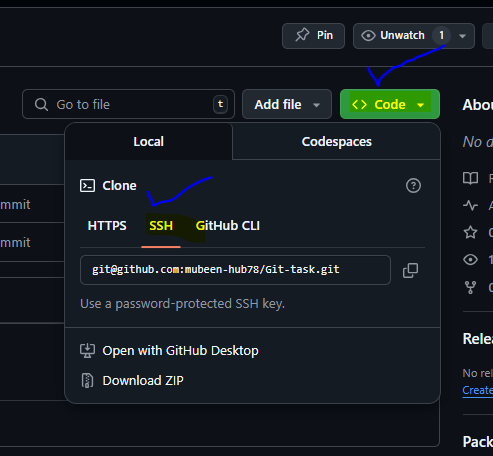
Select both README.md and .gitignore in **“Initialize this repository with”** while creating repository then **create repository**.



Repo with README.md & .gitignore is created.



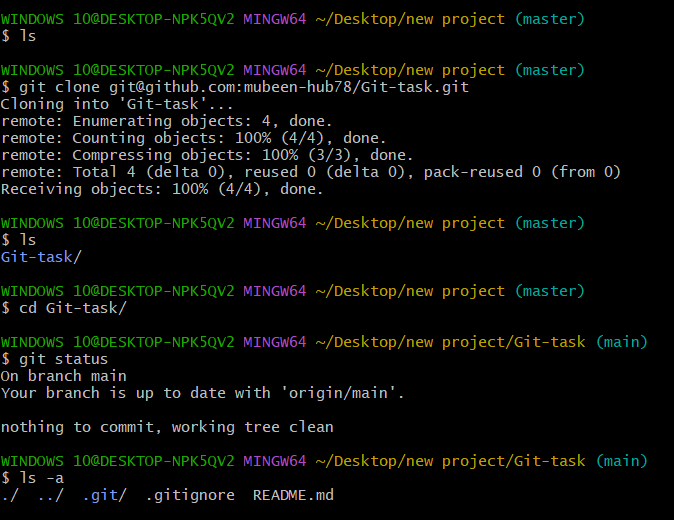
**3)Clone the created repo to local.**



**git clone <ssh url> -** Command used to clone.

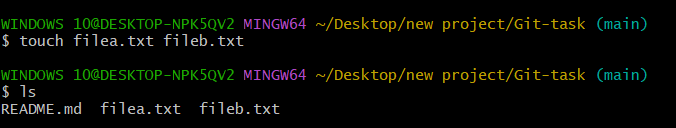
Copy SSH url from code.

Repository is added to the local machine.



**4)Create two files in local repo.**

**touch <file name>** – command used to create files.



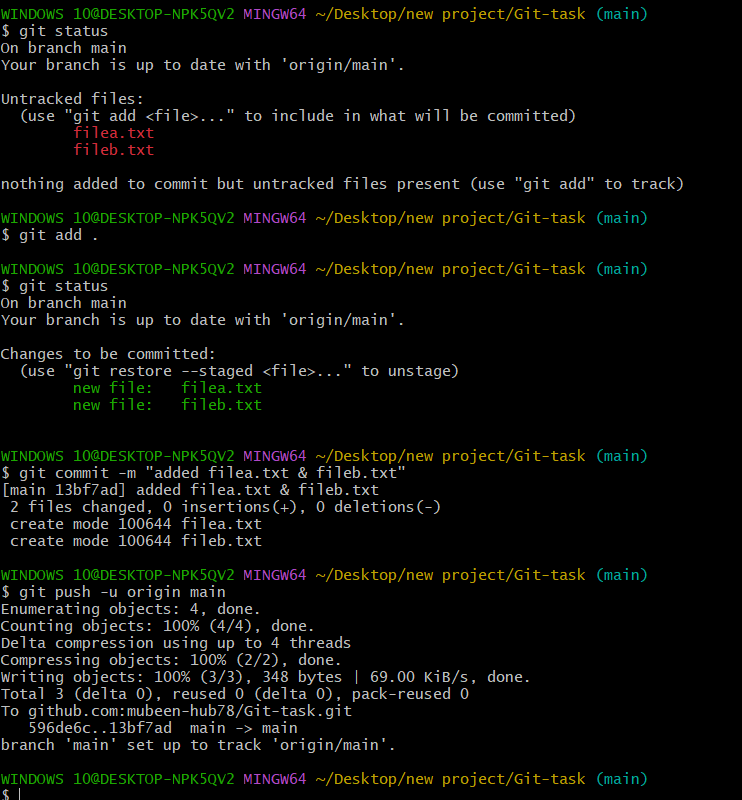
**5)Commit two files and push to central Repository.**

**git add .** – Used to add files from untracking area to tracking area.

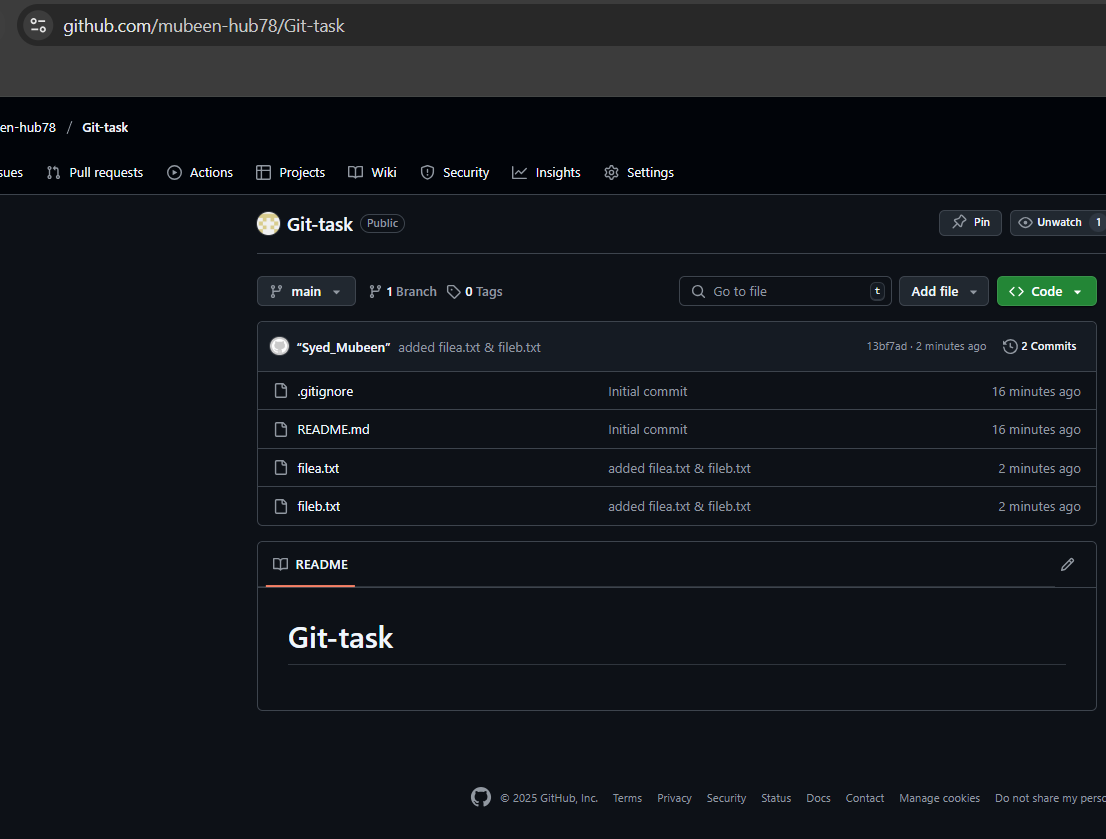
**git status** – Used to check files status whether files are in untracking area or tracking area.

**git commit -m “message”** – Used to commit changes with commit message.

**git push -u origin <branch\_name>** - To push the changes in to the Central Repo.



Changes in Central Repo after push from Local repo.



**6)Create a branch in local and create a sample file and push to central.**

**git branch <new\_branch\_name>** - To create a branch.

**git branch –-list** – To check available branches.

**git checkout <Branch\_name\_to\_be\_changed>** - To switch the branch.

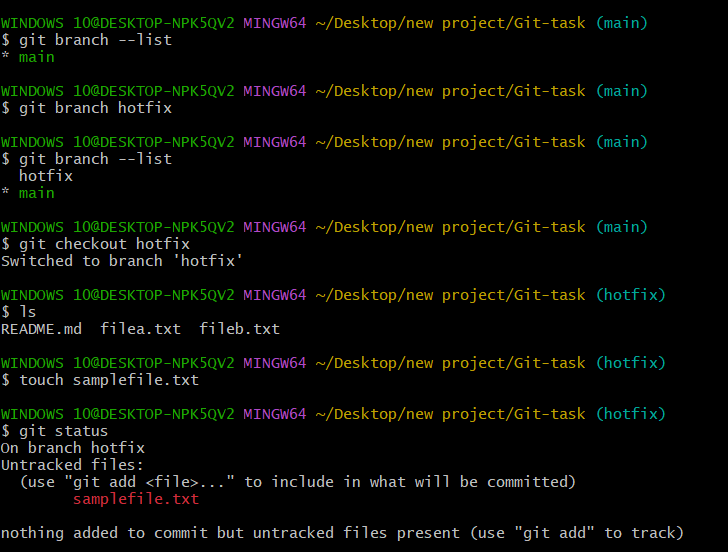
**touch <file name>** – command used to create files.

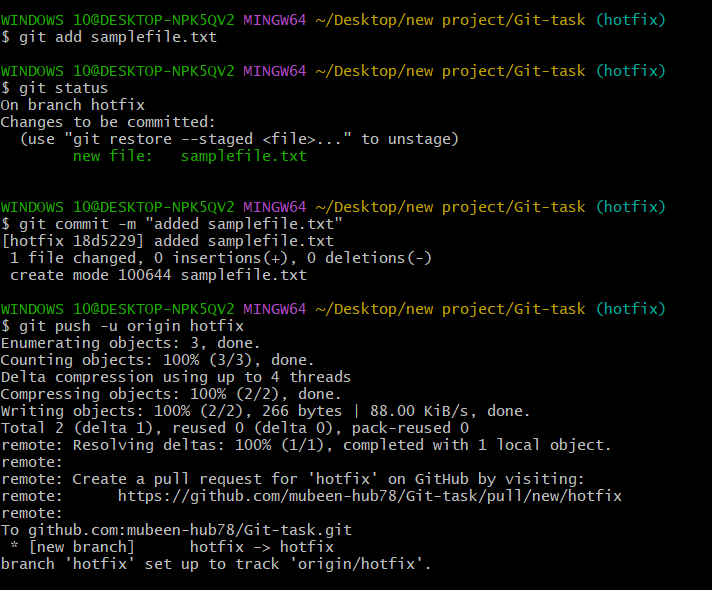
**git add .** – Used to add files from untracking area to tracking area.

**git status** – Used to check files status whether files are in untracking area or tracking area.

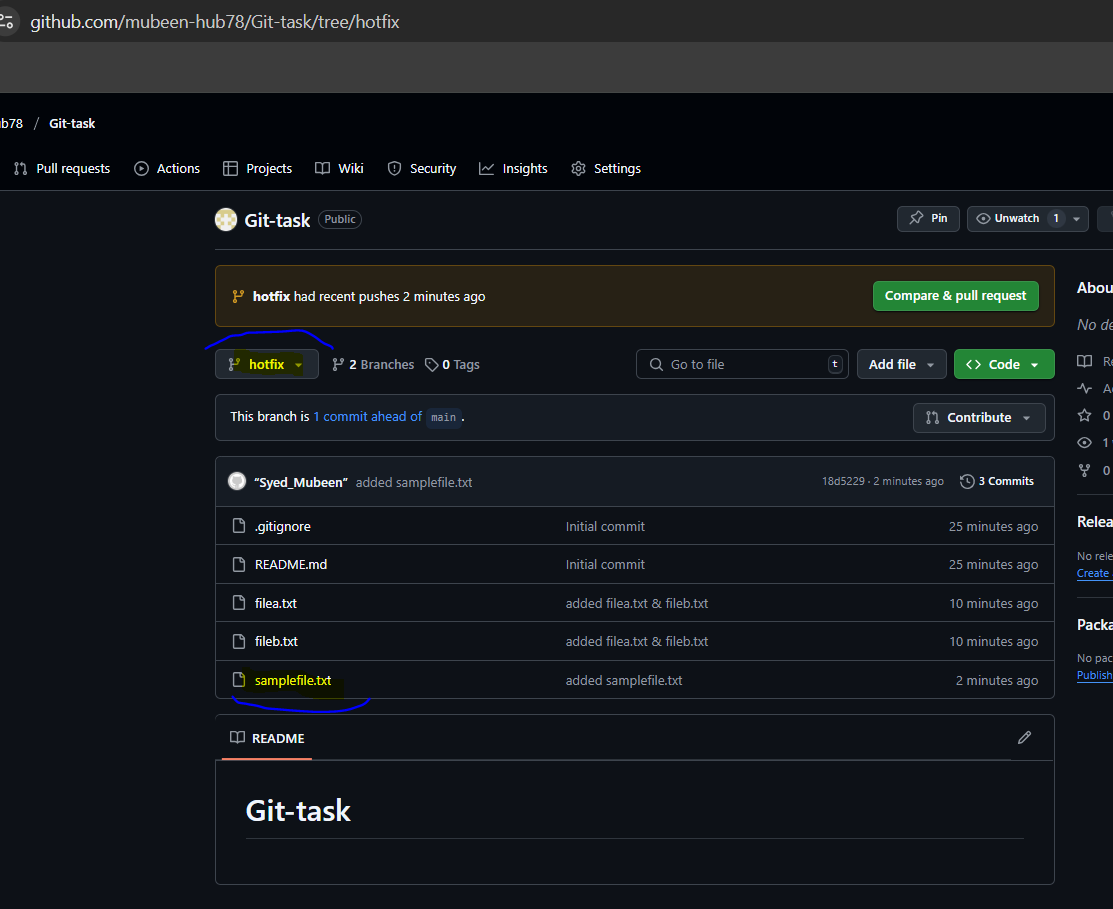
**git commit -m “message”** – Used to commit changes with commit message.

**git push -u origin <branch\_name>** - To push the changes in to the Central Repo.



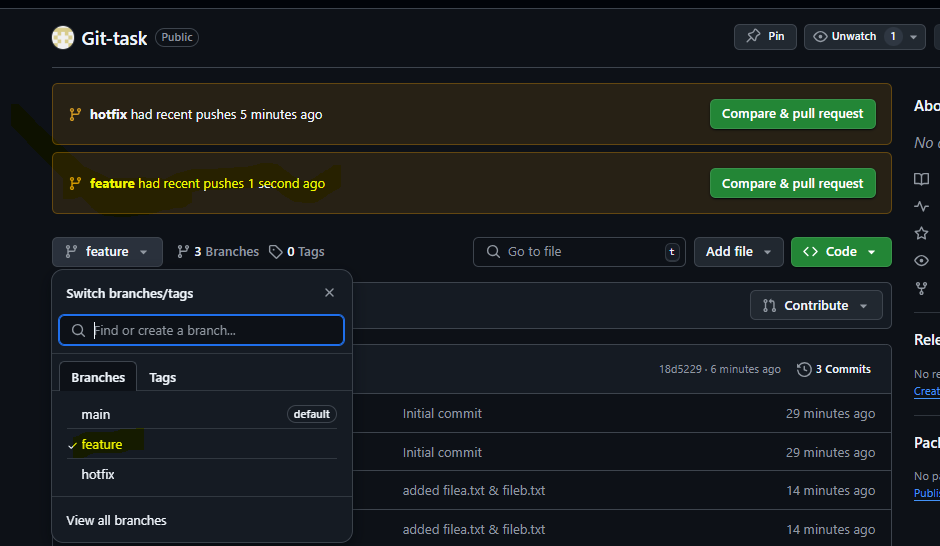


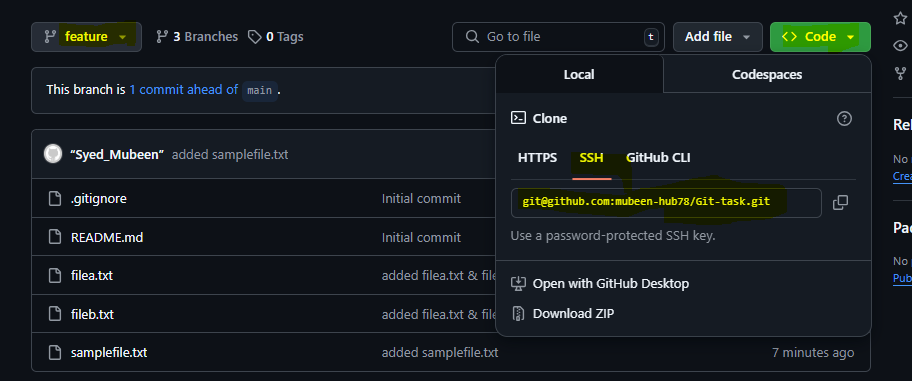
Git-hub GUI looks like this after updating changes.



**7)Create a branch in github and clone that to local.**

To create branch in Github, In the dashboard of Repo click on main enter branch name then create branch from main.





**git clone <ssh url> -** Command used to clone.

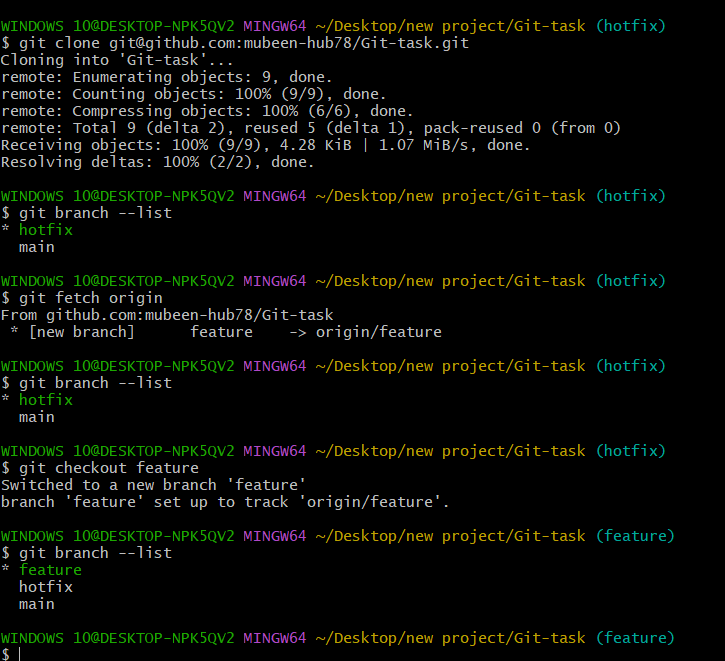
Copy SSH url from code.

**git fetch** – To pull all configuration changes from the central Repo.

**git checkout <Branch\_name\_to\_be\_changed>** - To switch the branch

**git branch –-list** – To check available branches. Where \* shows the current branch.

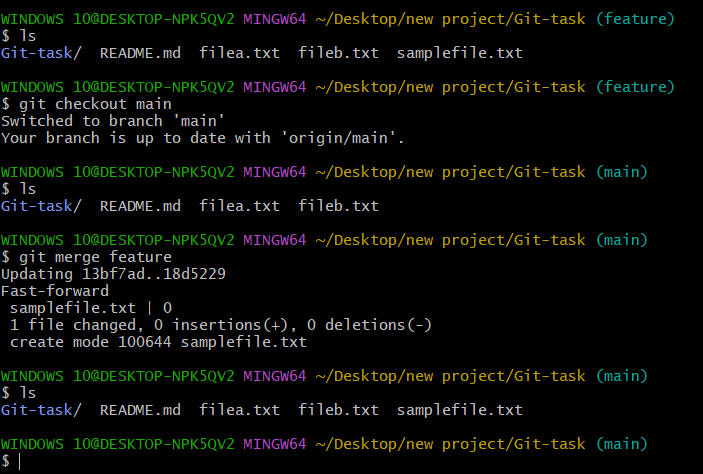
.



**8)Merge the created branch with master in git local.**

First switch to master/main branch then,

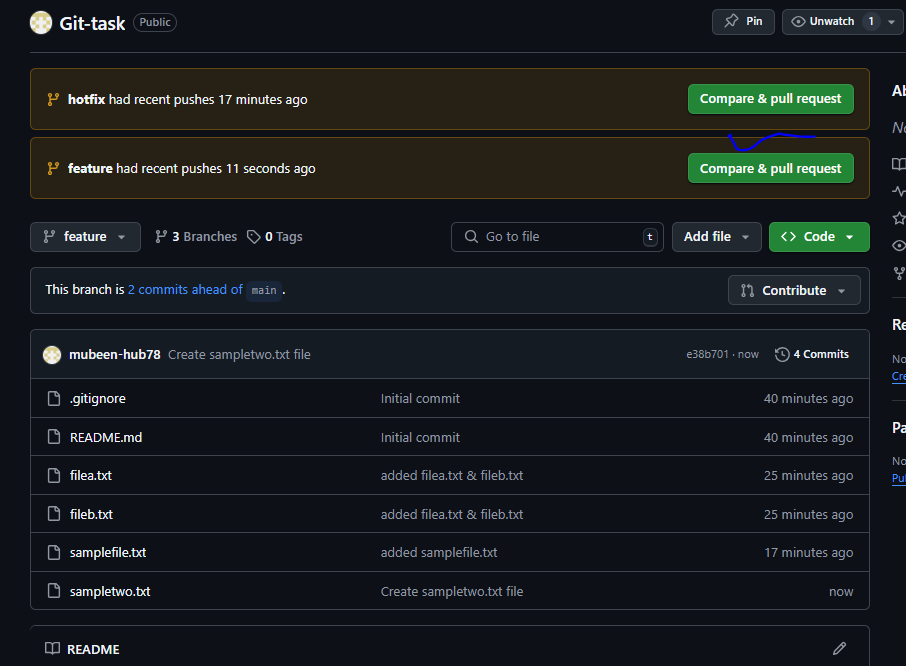
**git merge <branch\_name\_to\_be\_merged>** - Command used merge 2 branches, where the changes made in other branch are tracked and Collab’s with main/master branch.



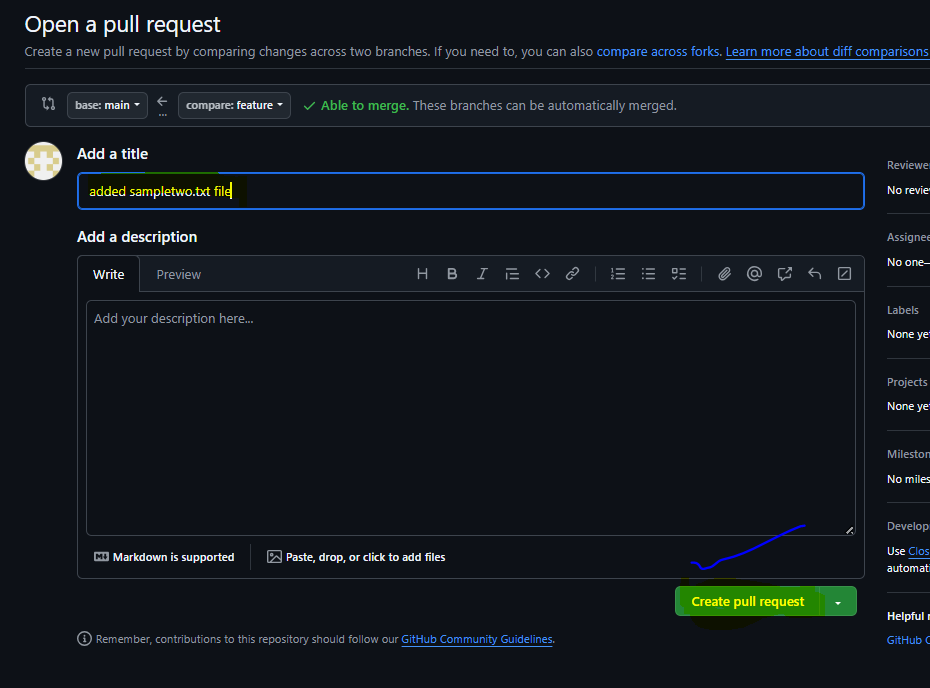
**9)Merge the created branch with master in github by sending a pull request.**

**Pull Request (PR)**, is same as merge but it requires review by reviewers before merging the branches.

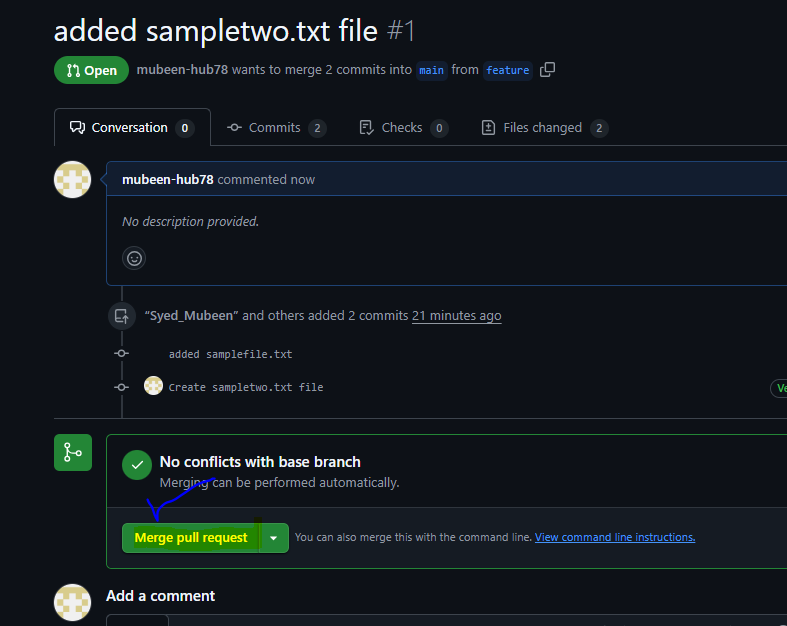
Click on **“Compare & pull request”** showing beside the branch name for the branch to be merged.



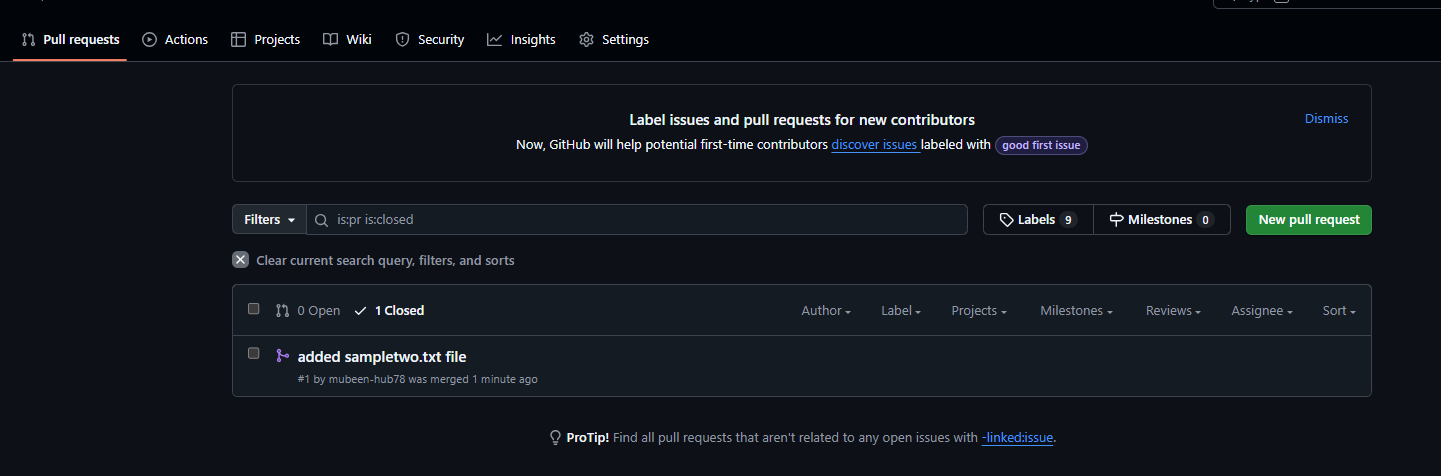
Add title and description relevant to PR and click on **“Create pull request”**.



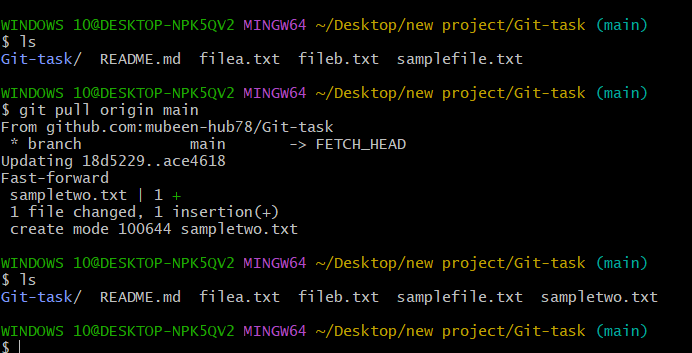
After that reviewers can review the PR and respond accordingly then, anyone can Complete the merge.



Then confirm merge by checking Pull requests.



**git pull –** Command use for pulling changes from central repo and changes are updated.



**10)create a file in local and send that to branch in github.**

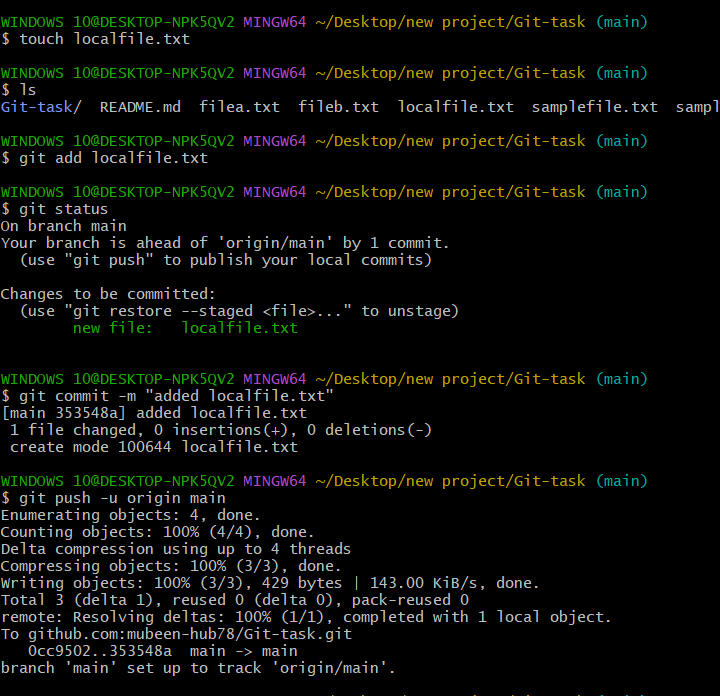
**touch <file name>** – command used to create files.

**git add <filename>** – Used to particular file from untracking area to tracking area.

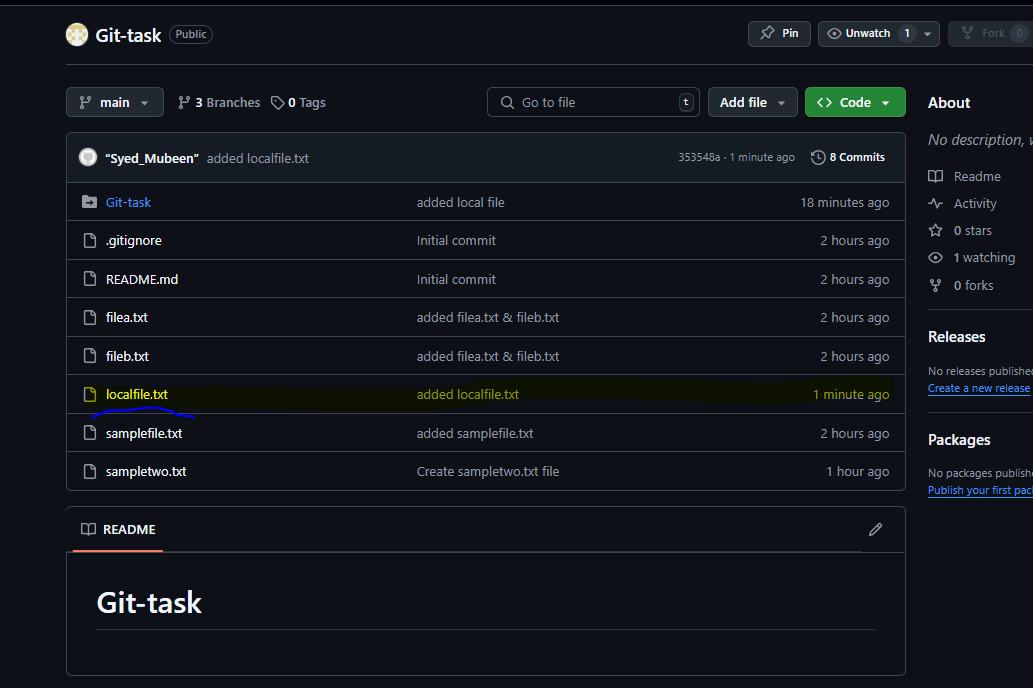
**git status** – Used to check files status whether files are in untracking area or tracking area.

**git commit -m “message”** – Used to commit changes with commit message.

**git push -u origin <branch\_name>** - To push the changes in to the Central Repo.

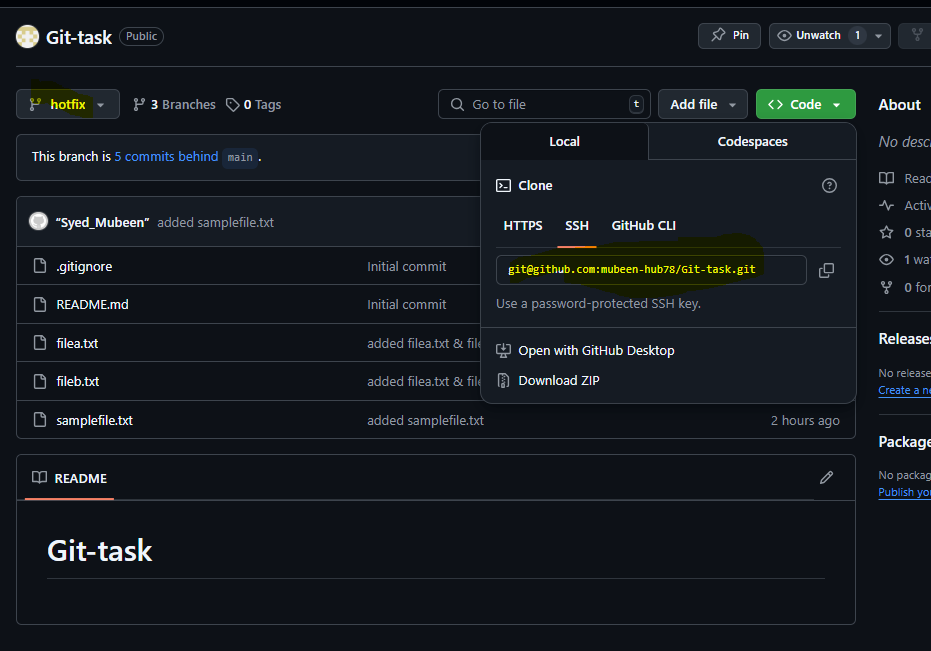


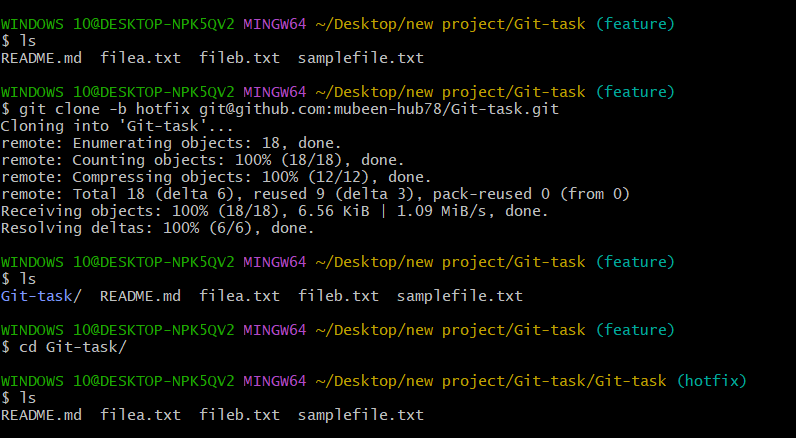
Changes in Repo after push.



**11)clone only a branch from github to local.**

**git clone -b <branch\_name> <ssh url> -** Used to clone particular branch.





**12)create a file with all passwords and make that untrackable with git.**

Create a file of passwords and add that passwords file name into the file .gitignore.

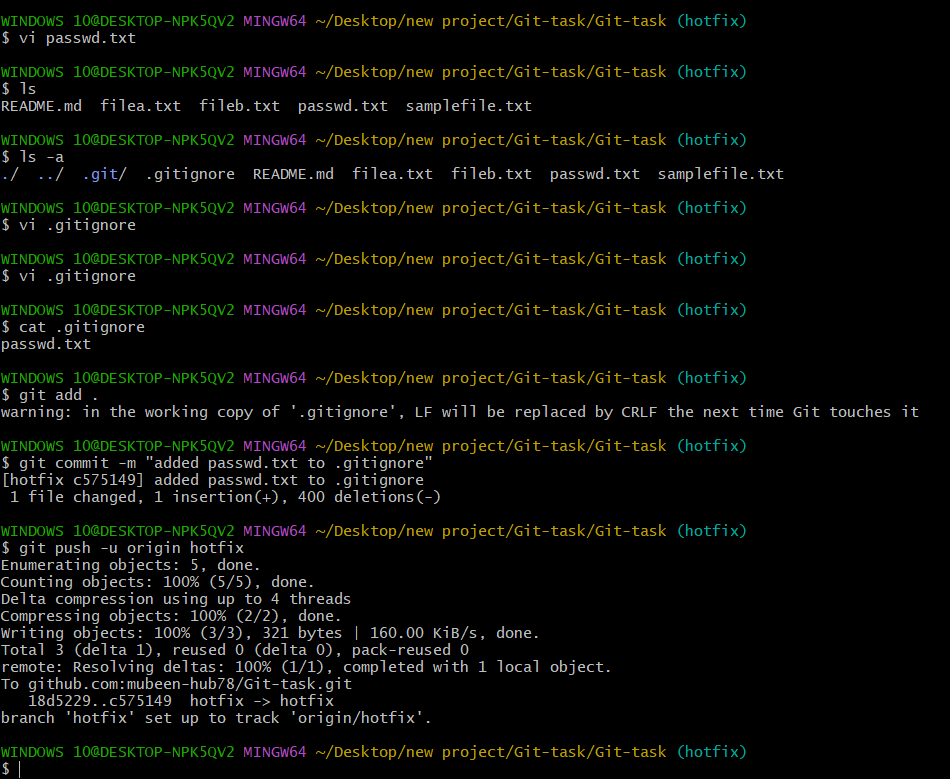
Commands used, **Vi <filename>** - To edit the file.

**cat <filename> -** To view the content.

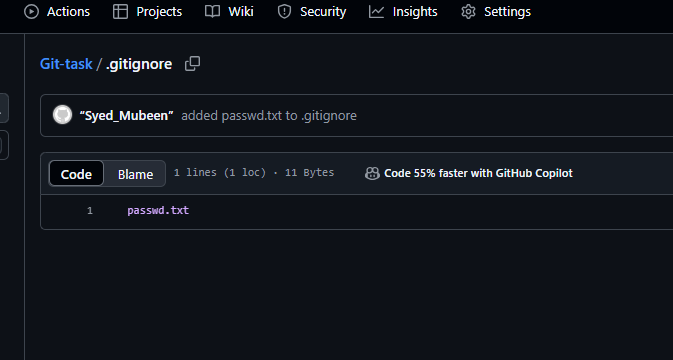
**git add <filename>** – Used to particular file from untracking area to tracking area.

**git commit -m “message”** – Used to commit changes with commit message.

**git push -u origin <branch\_name>** - To push the changes in to the Central Repo.

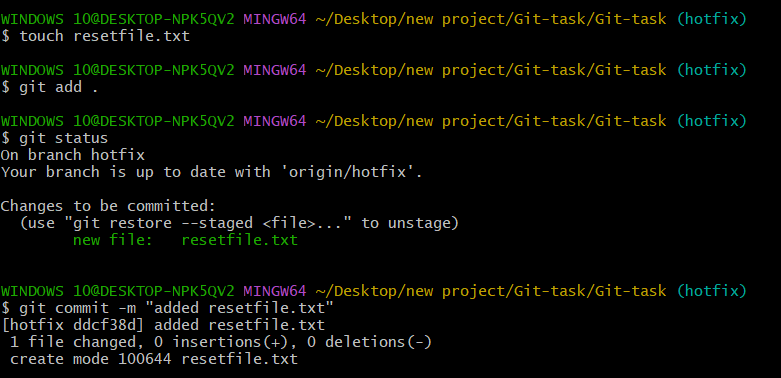


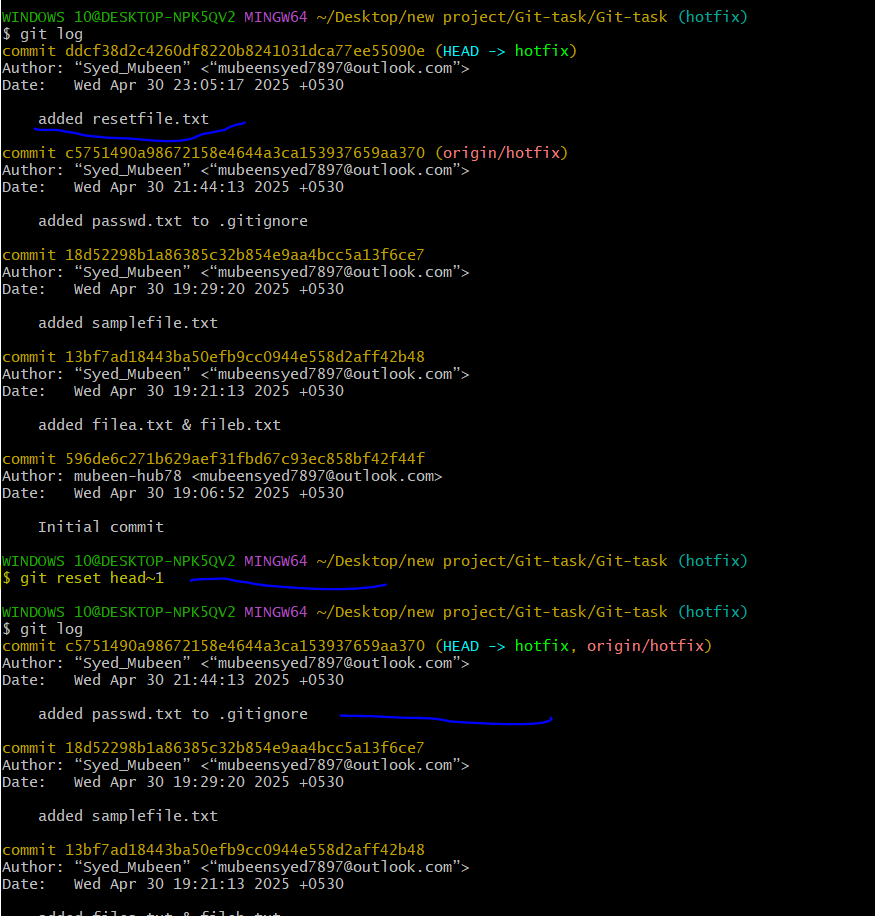
Git hub GUI to check .gitignore file. Purpose this task is to make passwords un trackable for security issues.



**13)make a commit and make that commit reset without savings changes.**

**git reset head~n** – To undo committed changes, when head is for first lines and n for no.of lines.

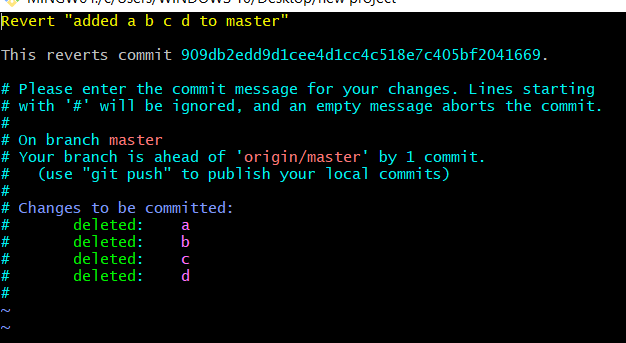


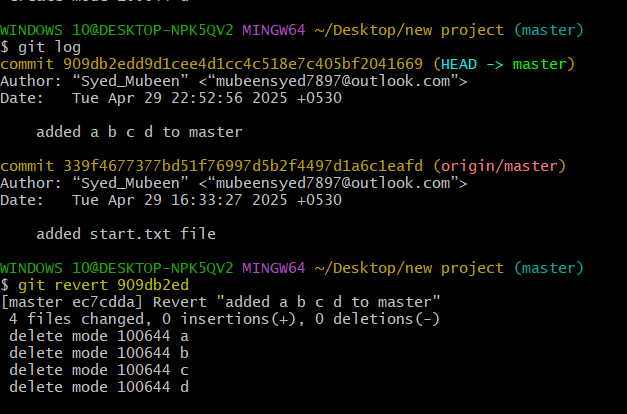


**14)Revert a commited commit to the older version.**

**git revert <commit id> -** To Rollback/undo/revert commits to the older version.

Commit id can be found in logs file by using command **git log**.



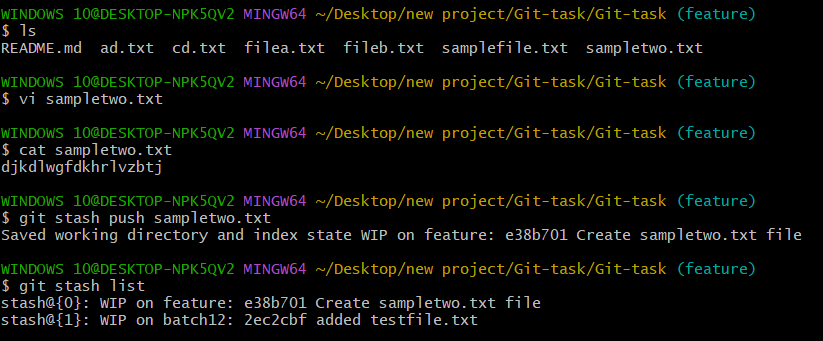


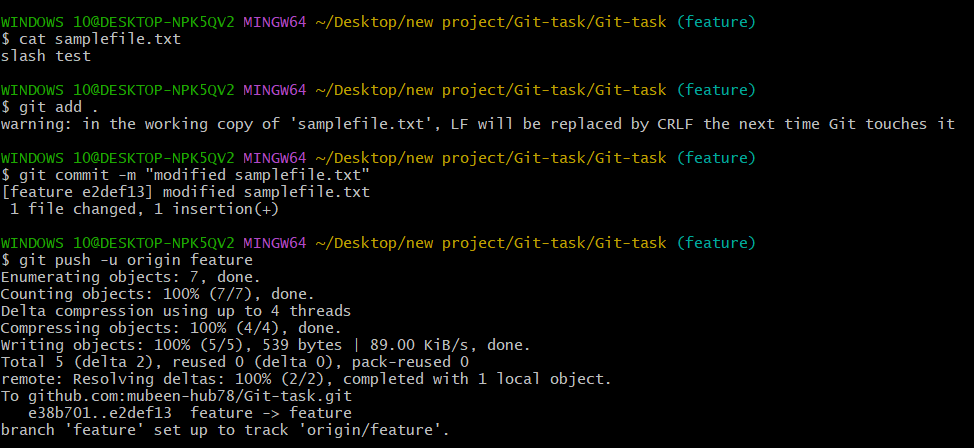
**15)push a file to stash without savings the changes and work on another file.**

**git stash push <filename> -** To push file to the stash memory so the changes cannot be committed or pushed to central repo.

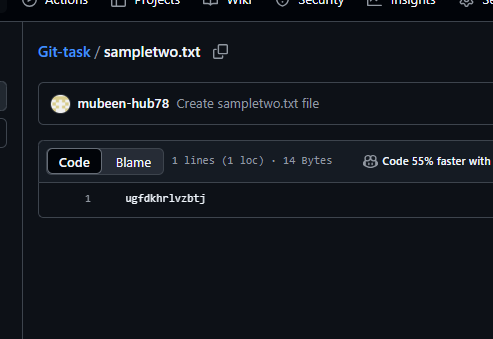
**git stash list** – To check files available in stash memory.

Except the file in the stash memory other all other files can be modified, committed and can be pushed to central repo.

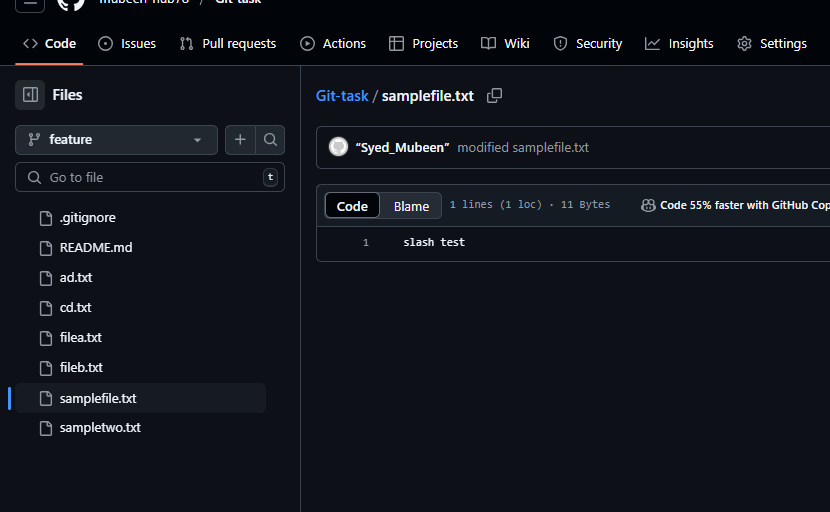




No changes updated in the file moved to stash memory.

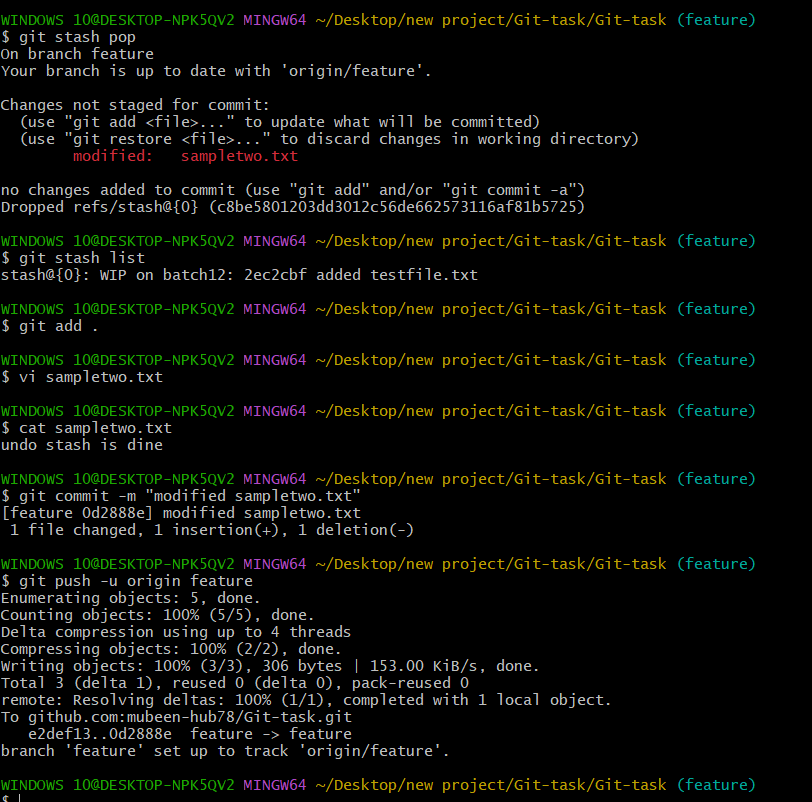


Changes are updated in the other file.

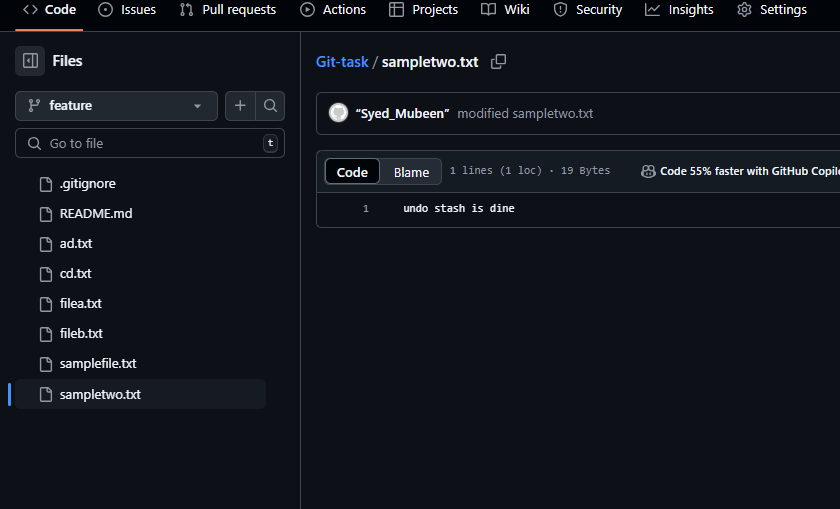


**16)undo the stash file and start working on that again.**

**git stash pop –** To get back file from stash memory**.**

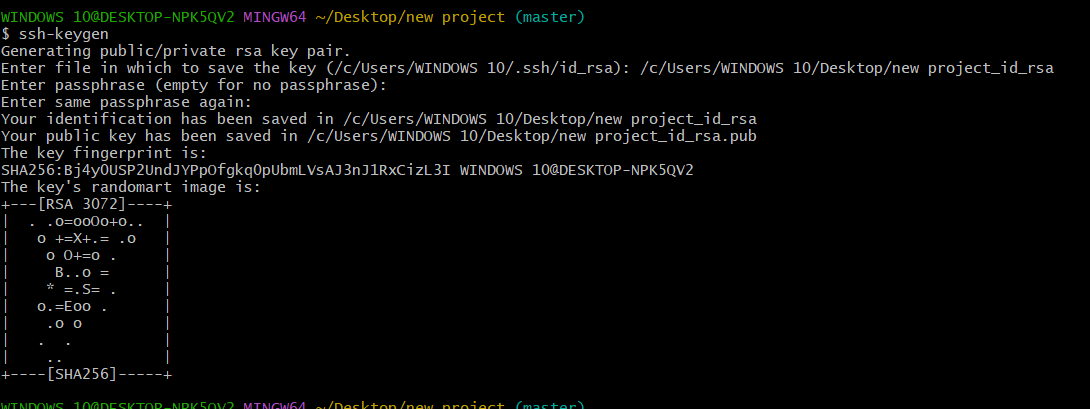


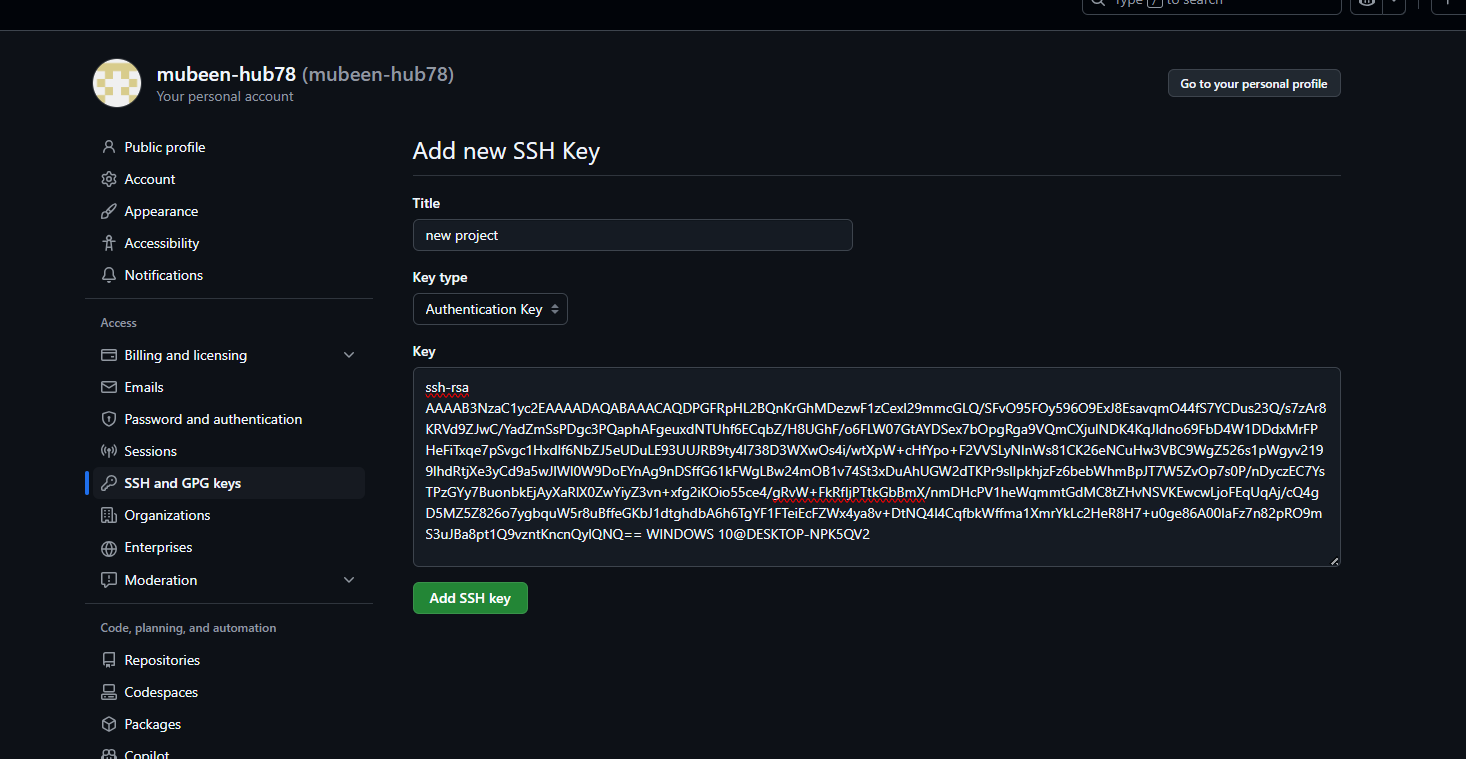
Now changes are updated after push.



**17)generate a ssh-keygen and configure into github.**

**ssh-keygen** – To generate keys.  
Copy public key and paste in Github settings --- **Add new SSH key**.

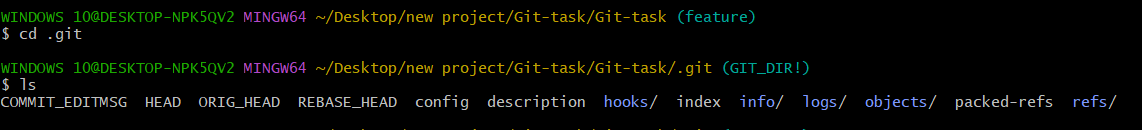




**18)configure webhooks to github.**

On hold for now.

**19)basic understanding of .git file.**



**COMMIT\_EDITMSG**: Stores the last commit message.

**HEAD**: Points to the current branch or commit.

**ORIG\_HEAD**: Holds the previous state of HEAD.

**REBASE\_HEAD**: Stores rebase info.

**config**: Repository-specific Git settings.

**description**: Describes the repo (used by GitWeb).

**hooks/**: Contains Git hook scripts.

**index**: The staging area for files before commit.

**info/**: Includes Git info, like exclude file.

**logs/**: Keeps logs of ref changes.

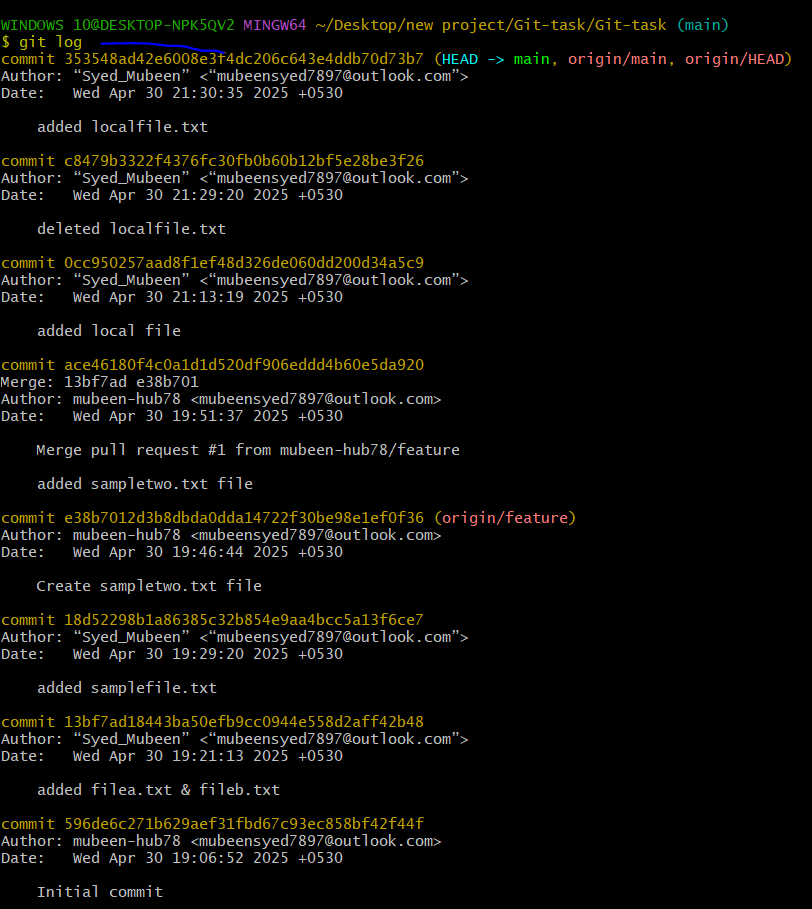
**objects/**: Stores commit data and file contents.

**packed-refs**: Stores refs in a packed format.

**refs/**: Contains branch and tag references.

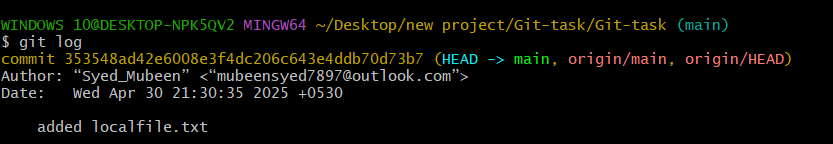
**20)Check all the logs of git.**

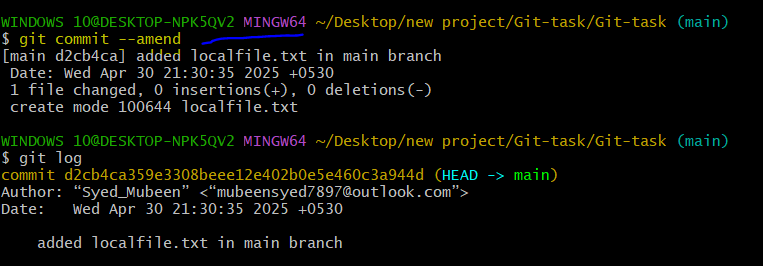
**git log** – Command used to view all the logs.



**21)Rename the commit message.**

Not recommended to do,   
**git commit --amend** – Used to rename the commit message.





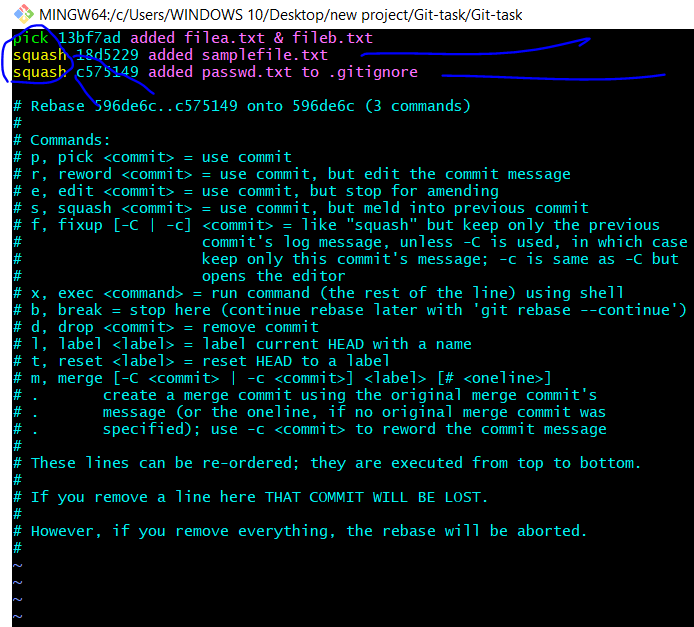
**22)Merge multiple commits into single commit.**

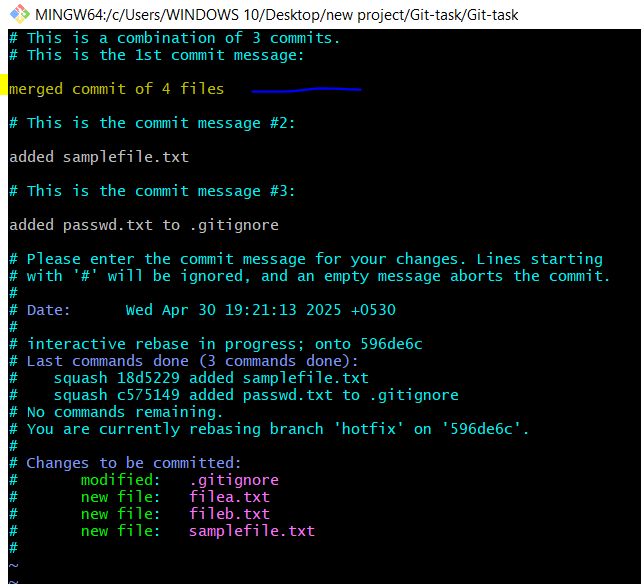
**git rebase head~n** – Used to merge multiple commits, where n is for no.of commits.

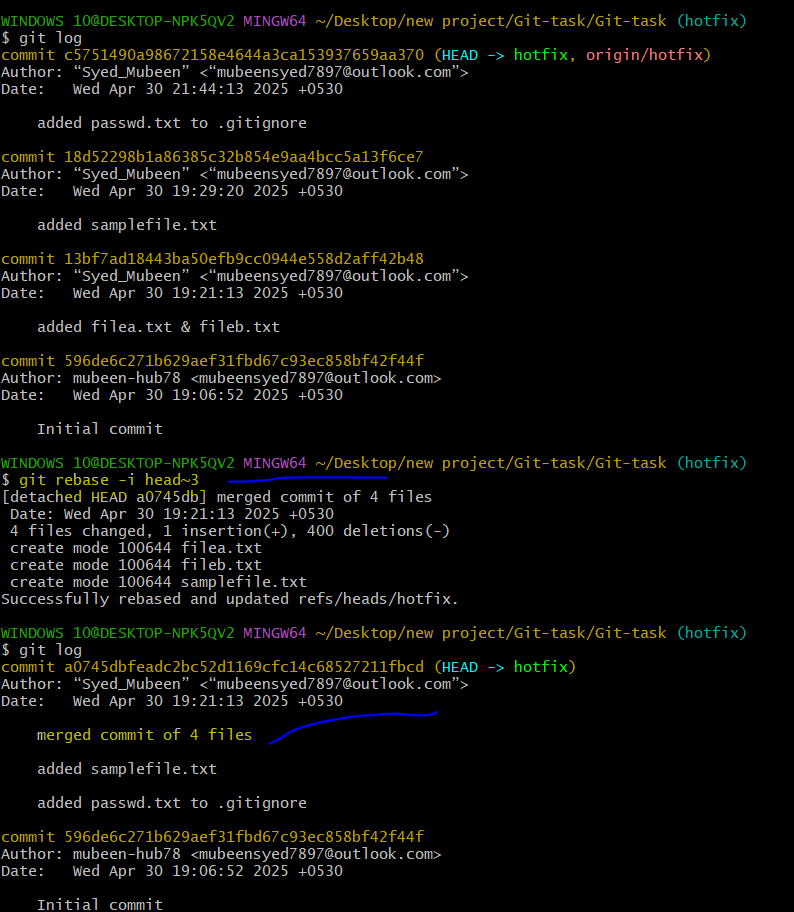
When we execute this command one file editor will be opened, we need to leave one line as **pick** beside the commit id and change to **squash** beside all other commit id’s then save the file.

One more file editor will be opened with merge commit details where we can name the merge commit and save the file.

Check with **git log** whether the commits are merged or not.







**The-End**