**GIT and GITHUB**

GIT is a free and Open-source version control system. Unlike other centralized version control system such as SVN (Apache Subversion) and CVS (Concurrent versions System), Git is distributed i.e., every developer has the full history of their code repository locally. This makes the initial clone of the repository slower, but subsequent operation such as commit, blame, diff, merge, and log faster.

HOW GIT WORKS

* Create a “repository” (project) with a git hosting tool (like Bitbucket)
* Copy (or clone) the repository to your local machine
* Add a file to your local repo and “commit” (save) the changes
* “push” your changes to main branch
* Make a change to your file with git hosting tool and commit the changes
* “pull” the changes to your local machine
* Create a “branch” (version), make a change, commit the change
* Open a “pull request” (propose) changes to the main branch)
* “merge” your branch to the main branch

GIT Command

1. Create Folder

*$ mkdir foldername*



1. Change directory

*$ cd path*

change folder



Change disk



1. List all the item in the folder

*$ ls*

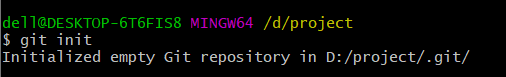


**.git FOLDER**

This folder contains all information that is necessary for the project and all information relating commits, remote repository address, etc. it also contains a log that stores the commit history. This log can help us to roll back to the desired version of the code.

1. Creating empty git repository

*$ git init*

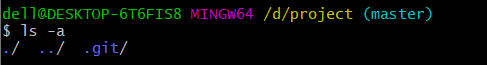


Now we will not be able to see the .git folder as it is hidden so the simple *ls* command will not work

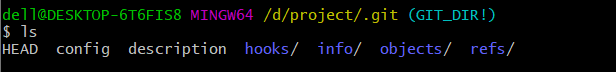


So, we use

1. *$ ls -a*

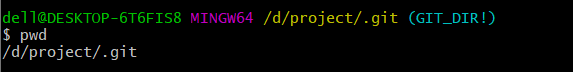
**

Now inside .git folder we get all the file



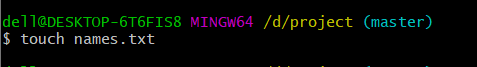
1. To get the current file location

$ *pwd*



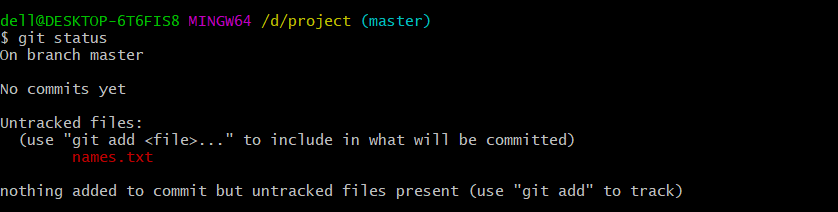
1. To create new file

$ touch filename

**

1. To get the status of the added/removed file

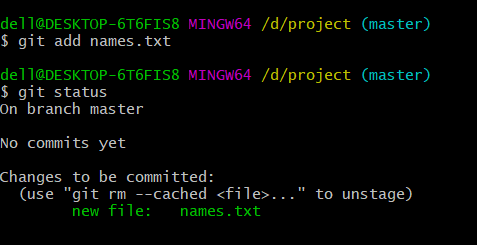
*$ git status*



Right now, no one knows that the file is added

1. To add file

*$ git add filename*



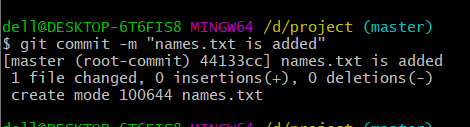
9.b to add all the file at once

*$ git add .*

10.to commit changes

*$ git commit -m “names.txt file is added”*

-m is used for message

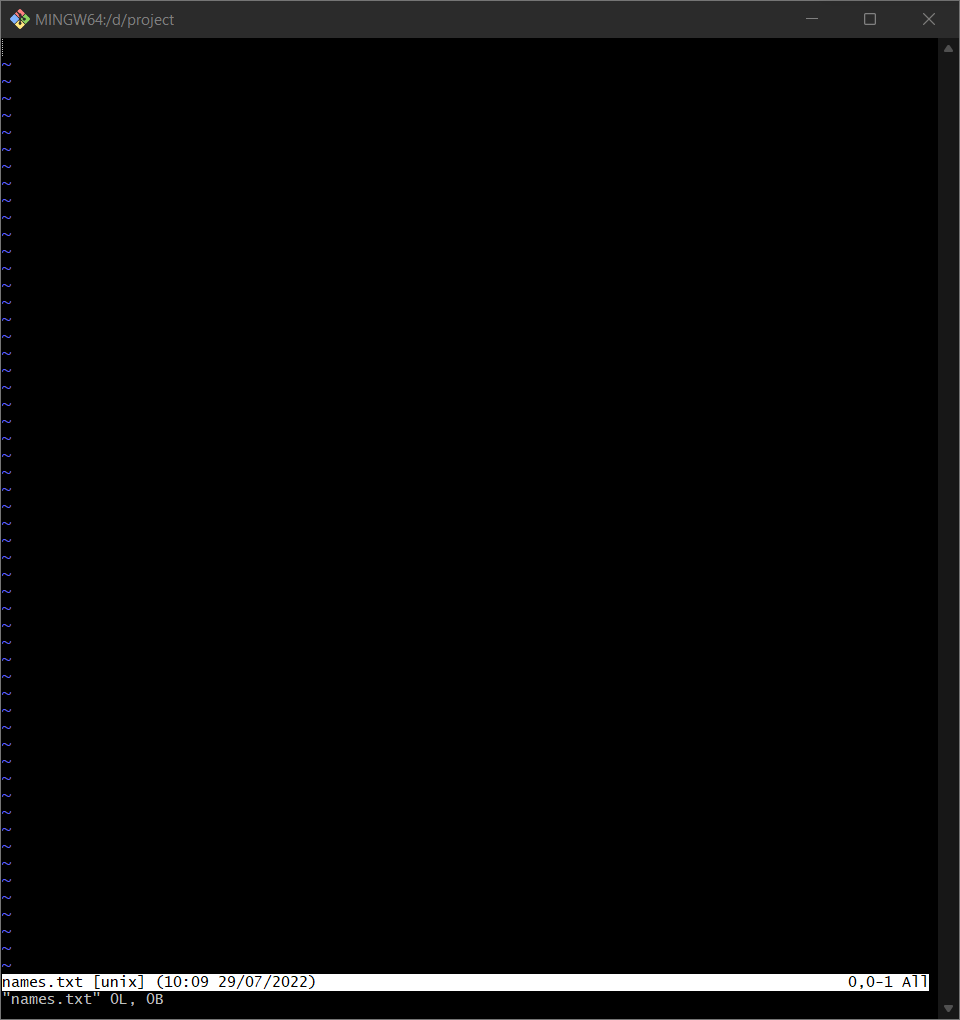


Here names.txt is added and in file there are 0 insertions, 0 deletion.

Now we will use vi editor

11.command to invoke vi editor

*$vi filename*



Now we can edit our text file

After editing file

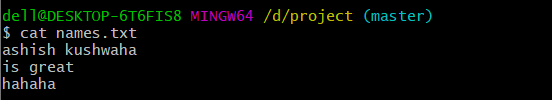
Click esc

The :wq

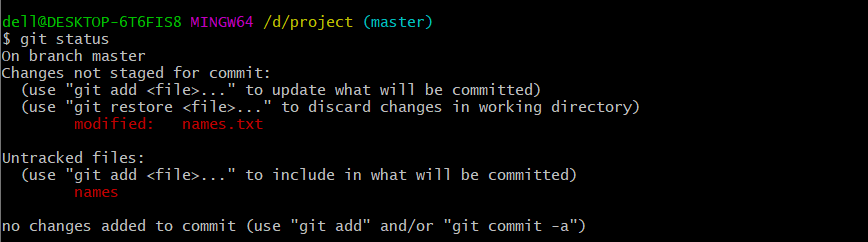
File will be saved and exit from vi editor

12. to see the all the content of file

*$ cat filename*

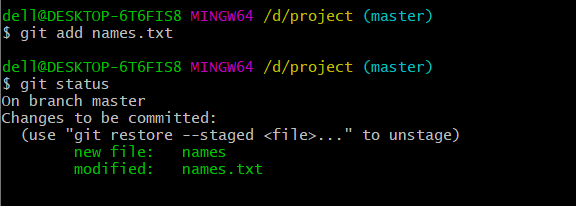


Now we check the changes



This says names.txt is modified

Now we add the file

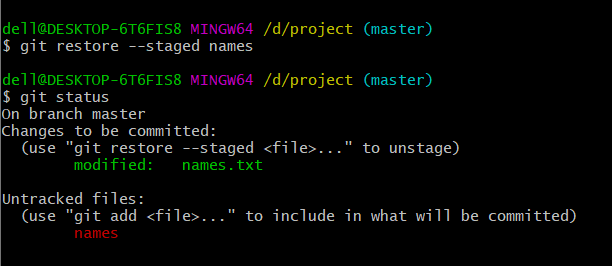


The files are on the stage

From here also if we don’t want to add files, we do that. For example, I have created one extra file names

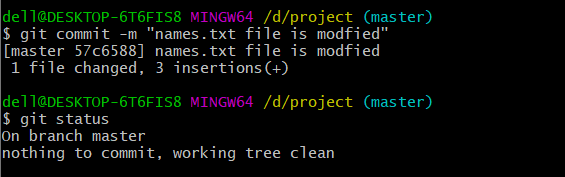
13. command to unstage the file

*$ git restore --staged names*



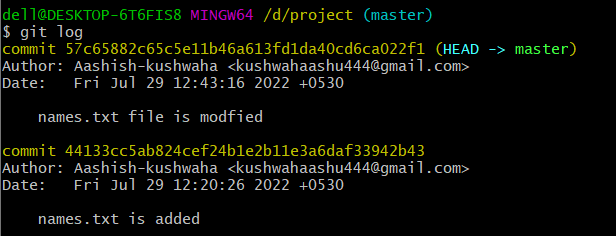
Now we can see ‘names’ is out of stage

Now we commit names.txt



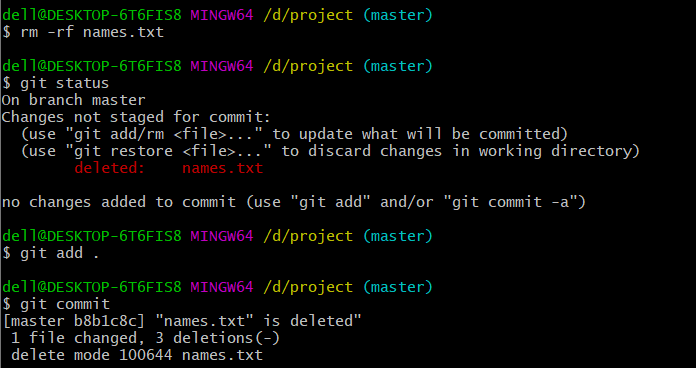
14. command to check all the history

*$ git log*



15. command to delete the file

$ rm -rf names.txt

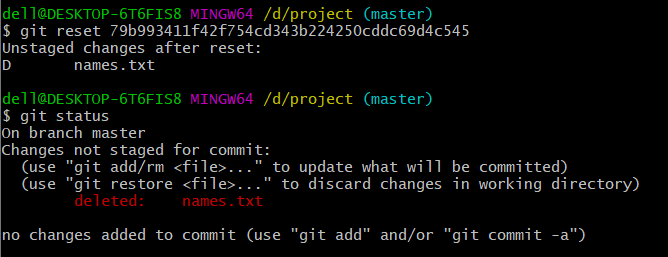


We check using log



Now if we want go back any of the commits (the stage of project which we want to see again) we use the HASHKEY associated with that commit

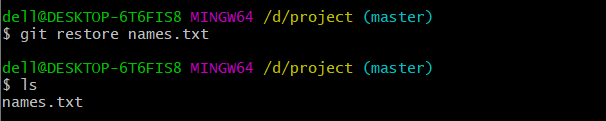
16 $ git reset hashkey



All the files are now on the unstage area

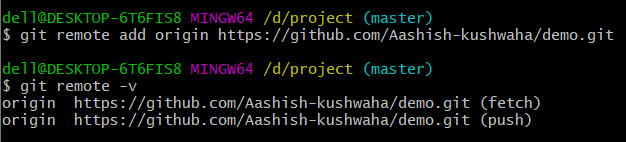
17. To restore the file from unstage area

$ git restore filename



18. Connecting remote repository to local repository

$ git remote add origin <repository link>



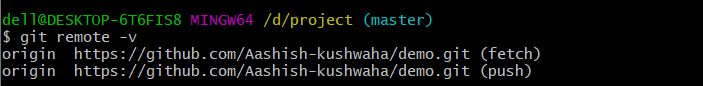
Remote means we are working with URLs (of remote repository)

add means that we just added an URL

origin just mean you name of the URL you want to add

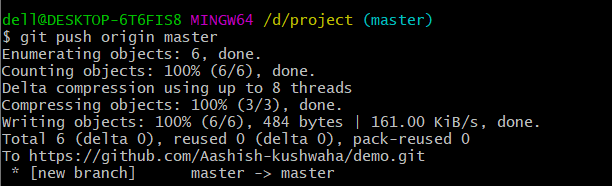
19.to see all the URLs attached

$ git remote -v



20. to push the changes to the BRANCH

$ git push origin <branch name>



jshhdjddkakk