**Commands used in GIT and GIT HUB**

1. **cd :-** Used to change the current working directory.
2. **pwd :-** Show the present working directory in which the user is currently
   1. Present.
3. **git config -- global user.name** “your username”
4. **git config -- global user.email** “your email address”
5. **git config -- list**
6. **git config -- global core.editor**
7. **git config user.name**
8. **git config user.email**
9. **git config --list**
10. **git status :-**To show the status of the file i.e the file are in staged stage or not the files are already commit or show that a git file is created.
11. **git config --global core.editor editor-name** :-Used to select editor.
12. **git init :-**Used to create a GIT repository.
13. **git add -- a :-**Put all the files in the staging area.
14. **git commit -m “your message” :-**Commit the files with the message. Be sure
15. that the message should be relevant.
16. **git log :-**Show how many commits are made on a git repository.
17. **git add file-name :-**Put the specific file in the stage area.
18. **rm -rf .git :-**Delete all the git files/repository.
19. **touch file-name.extension :-**Create an empty file.
20. **touch .gitignore :-**Create a file in which put the name of that files whom we don’t want to affected by the git add -- a.(If we write the \*.extension in the .gitignore file then all the files with that extension are not affected by the git add -- a.
21. **git diff :-**Used to compare the current working directory and the staging directory.
22. **git diff --staged :-**Compare with the previous commit with the current staging area.
23. **git commit -a -m “your message” :-**Skip the staging area and make a commitnot commit untrack files this command will track only the tracked files.
24. **git rm file-name.extension :-**Remove the file or directory.
25. **git mv file-name.extension new-name.extension :-**Rename the files.
26. **git rm -- cached file-name.extension :-**Can’t track the specified files.
27. **git log -p :-**Show’s what changes are made in the files.
28. **git log -p -5 :-**Show’s only the last 5 commits. We can replace the number(5) with any number.
29. **git log -- stat :-**Show’s the changes in the form of summary.
30. **git log -- pretty – oneline/short/less :-**Show’s all the commits only in the one line.
31. **git log -since=2days/weeks/months/years :-**Show’s the files according to the time stamp.
32. **git commit --amend :-**To change the current file.
33. **git restore -- staged file-name :-**To unstaged the file with the specific name.
34. **git checkout --file-name :-**To unmodify and undo files. But can’t affect the files in the staging area.
35. **git checkout -f :-**Step back all the changes in all the files.
36. **git remote :-**Create a remote repository.
37. **git remote add origin** [git@github.com:repositorymam/description.git](mailto:git@github.com:repositorymam/description.git)
38. **git remote -v :-**Show where we push and pull changes.
39. Steps to put files on to the repository:-

* Login to your git account.
* Go to setting, SSH and GPG keys.
* Create new SSH key and title.
* Copy SSH key from the GIT HUB.
* Follow the steps that are show on the GIT HUB page.

1. **git config --global alias.new-name command :-**Create an alias for the original name.
2. **for example:-** git config --global alias.st status
3. **NOTE:-** If we use the alias name which is too long then put the name in the quotes(“ “).
4. **git checkout -b branch-name :-**Creates a new branch.
5. **git checkout master/new branch-name :-**Used to switch between branches.
6. **git merge branch-name :-**Merge all the branches to the master branch.
7. **git branch -v :-**Show last commit of every branch.
8. **git branch --merged :-**Show all the merged branches.
9. **git branch --no-merged :-**Show all the branches that are not merged.
10. **git branch :-**Show how many branches we created.
11. **git branch -d branch-name :-**Delete the branch. If the branch is not commit then the error will be showed. To overcome to this error commit the branch or used the following command:-
12. **git branch -D branch-name :-**Delete the branch without any error even commited or not.
13. **git checkout branch-name :-**Move to the specific branch.
14. **git push origin branch-name :-**Push specific branch to the remote repository.
15. **git push origin branch-name:new-name :-** Push specific branch to the remote repository with the new name.
16. **git push -d origin branch-name :-**Delete the specified branch from the remote repository.

**SOURCE** :- https://www.youtube.com/playlist?list=PLu0W\_9lII9agwhy658ZPA0MTStKUJTWPi