**GIT BASH**

**THERE ARE FOUR STAGES IN GIT**

**1 -: UNTRACKED**

**2 -: STAGGED**

**3 -: MODIFIED**

**4 -: UNMODIFIED**

**# we can check the files by using git status (command**)

* if some files is present then,
* we will first add these files by using git add (file name) or using. or using -A (command),
* by adding these files moved to stagged area,
* then we will use git commit -m"(any comment you want to use)" or git commit.
* if we use git commit, a editor mode will open after this you will press (i) button and starts inserting the comment

**\*TO CLOSE THIS EDITOR WINDOW FIRST ESC -> : -> wq**

* **Touch command use to create file in the folder**

**# git checkout command is used to**

match your present file(modified) to previous committed file and restore previous commit.

* #git checkout -f is used for all files present in the folder

**#git log command will show**

* all the committed statements with time and author
* TO FILTER GIT LOG
* #git log -p -number (as many as you required)
* #q is used to return in main menu

**# git diff will compare**

the working directory file to stagging area (file is same in both cases)

git diff (command)

**TO SKIP STAGGING AREA**

* use git commit -a -m"any comment"
* $git rm (command) is used to delete the file
* git rm FileName
* OR
* git rm --cached FileName is used to directly remove file from stagging area to untracked file

**#git removing command**

* git rm –cached (file name) is use to remove file from stagging area to untracked area.
* Git rm(file name) is used to delete the file

**#Git Ignore**

There are so many files that you don’t want to track such as log files,tmp files

* First create touch .gitignorae
* Create log file such as mylogs.log or useless folder
* Add these logs file into .gitignore (full name ,\*extension, /file name (only ignore file in its root directory(in simple language where is gitignore only those places you will ignore that file , folder name/ (/ is used to ignore any folder)

**#git branch**

You can create more than master ,branch which will help to integrate /merge the other features branchs to main branch

* Git branch feature1(it will create another branch name feature1)
* You can use feature1 branch to create another code which will you later integrate to master
* To switch between branches use git checkout branchname(where you want to switch)
* To merge different branches use git merge branchname
* #Short Hand for creating new branch with switching i.e git checkout -b BranchName