-----------------

|  |  |
| --- | --- |
| git config  git config --global user.email [you@example.com](mailto:you@example.com)  git config --global user.name "your name" | to set up the configuration |
| pwd | present working directory |
| cd  cd”c/” | change directory |
| git status | to see the status of the repository |
| git init | to make a local directory a git repository |
| git add  \*git add --a / git add.  \*\*git add <filename.ex>t | \*to stage all the files in a directory  \*\* to stage the single file |
| git commit -m ”comment” | to commit the change |
| git clone <the address> <folder\_name> | to clone a existing github repository in a local repository |
| git log | to trace the committed change in the files |
| rm -rf .git | to delete the git repository |
| git ignore  \*touch error.log  \*\*touch .gitignore  git add .gitignore | \* will create a error.log file in the repository  \*\* will create .gitignore file the the repository  if we merge the files which we just want to ignore we can merge them with .gitignore by ( specifying the names, address or mentioning ant patterns) |
| git diff | compare the staging area with working directory  (if one file is modified after the staging- it will show what was the modification) |
| git dif --staged | compare previous commit with the current staging |
| git commit -a -m ”comments”  direct commit | it will commit all the tracked file in the repository but not the untracked one- which is new created or was never staged. |
| git rm <filename.ex> | to remove a file from the repository |
| git mv <prev\_file.ex> <new\_file.ex> | to rename a file presented in the repository |
| git rm --cached <filename.ext> | to stop the tracking of this file |
| \*git log -p  \*\*git log -p -n | \*it will show the log with git diff(difference)  \*\* it will show the diff for n commits |
| git log --stat | it will provide short overview of commits and difference |
| \*git log --pretty=0ne line  \*\*git log --pretty= short  \*\*\*git log --pretty=full | \*it will show the comments in one line  \*\* it will show shortly only author and commients the have made  \*\*\*it will show the author name (who has build the file) and commit name(who has made change) |
| git log --since=2.days/weeks/months | we can see the commit of different timestamps |
| git log --pretty=format:”%h-- %an” | it will show the log with <hash commit—author name > this format |
| git commit --amend | it will open a new terminal where we can edit the existing commits / merge the difference in a single commit |
| git restore –staged <file name> | to unstage a staged filr |
| \*git checkout -- <file name>  \*\* git checkout -f | \*it will reverse the file as previous commit  \*\* it will reverse all the files as previous commit  ! if we run this before commit we will loss the modification permanently |
| git remote add <url> | to adda remote repository |
| git remote -v | it will show the push(fetch) and pull repository |
| git push -u origin master  \*origin = name of thr repository  \* master/ main = brunch type | to push the code in the remote repository |
| git config –global alias.st status  $ git config --global alias.unstage 'resote --staged--' | it will change the status alias as s  it will small down the unstaged commad as unstage |
| git checkout -b <name>  git check out -b develop | it will create a new brunch  it will create a parallel brunch named develop |
| git checkout master  git checkout develop | we can switch between two brunches and works independently |
| git merge <branch name>  if conflict happened then solve the conflict and then git add and git commit | To merge the development done in the parallel branch with master branch |
| git brach -v | commit hash and commit message will show with branches name |
| git branch --merged  git branch --no-merged | Already merged branches  Not merged branches |
| git branch -d <branch name> | Gives caution if not merged and ask to add capital”D” to delete the branch |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |