**git init (Used to initialize current directory as an git repositry)**

**git add . (Moving all files from untracked to tracked in staged area)**

**git add \*.js (Moving all .js file from untacked to tracked in stage area)**

**git rm –cached <filename> : (it will revert file back from staging area)**

**git commit –m “your-comment (Moving file to local repository for the purpose to push it later on main repositry)**

**git log –oneline : (display log with only starting 7 SHA char and message)**

**git log –stat : (display log with stat about file and line change)**

**git log –p : (display log with detail change code adding –w flag will ignore white spaces changes)**

**git show : (show full detail including pach and stat details of most recent commit)**

**git show f60ac82 : (show full detail including pach and stat details of SHA starting from f60ac82 )**

**git diff : (It show the difference between file that is commited and file in working directory)**

**git reset (Moving back all the files from staged area to unstaged area)**

**git reset –hard (Moving back all the files from staged area to unstaged area and also removing all the changes before adding to staged area)**

**git tag –a <your tag> <start-7 SHA> : (Used to add tag to the commit with specified )**

**git tag : (Used to view tag)**

**git tag –a <your tag> : (Used to add tag to the most recent commit , tag is associated with commit)**

**git tag –d <tag-to-delete> : (Used to delete tag)**

**git checkout HEAD <file name> : (undoing changes that are not commited yet)**

**git revert <commitHash> : (undoing changes that are commited)**

**git reset –hard <commitHash> : (undoing changes that are commited)**

**For ignoring files we can create a file with only (.gitignore) extension and define rules for instance we can wite test.php now it will ignore test.php , .gitignore itself should be commited**

**==================== Branches ==================**

**git branch : (Show list of branches)**

**git branch –v : (Show list of branches with additional info)**

**git branch <branch-name> : : (creating new branch)**

**git branch <branch-name> <start 7 SHA commit>: : (creating new branch and point it to given commit SHA)**

**git checkout <newChange> : (switch to newChange branch from current branch name)**

**git checkout –b <newBranchName> : (creating and switching to new branch)**

**git merge <newChange> : (merge newChange branch with current branch)**

**git log new-dev…master : (Show commit defference in two branches)**

**git branch –d <branch name> : (deleting specified branch , D is used for forced deletion)**

**git log –oneline –graph –all : The --graph flag adds the bullets and lines to the leftmost part of the output. This shows the actual branching that's happening. The --all flag is what displays all of the branches in the repository**

**git reset –hard HEAD^ : (if you make a merge on the wrong branch revert it back)**

**git merge <branchname> : (Merge the specified branc with current branch)**

**git commit –amend : (you can change thing inlast commit)**

**edit the file(s)**

**save the file(s)**

**stage the file(s)**

**and run git commit --amend**

**git revert <SHA of comit to revert>: (In revert git will apposite it like If I Add Char A git will del it , and make a new commit)**

**git reset <SHA of comit to revert>: (In resert git will remove commit)**

**--hard : used to trash the commit**

**--mixed : used to put in Working Dir**

**--soft : used to put in staging area**

**Merge Conflict Indicators Explanation**

**The editor has the following merge conflict indicators:**

* **<<<<<<< HEAD everything below this line (until the next indicator) shows you what's on the current branch**
* **||||||| merged common ancestors everything below this line (until the next indicator) shows you what the original lines were**
* **======= is the end of the original lines, everything that follows (until the next indicator) is what's on the branch that's being merged in**
* **>>>>>>> heading-update is the ending indicator of what's on the branch that's being merged in (in this case, the heading-update branch)**

**=========================== Stash =======================**

**git stash : (save local changes)**

**git stash save myChanges : (Save local changes with specified name)**

**git stash list : (Show list of stashes)**

**git stash pop : (It apply the last stash and remove it from clipboard)**

**git stash apply stashname : (apply stash with specified stashname)**

**==================== Git-Hub Remote Reposiotory ==================**

**git remote : (it is used to create or manage remote repositories)**

**git remote add origin <remote-repo url> : (it is used to create a short name of origin that points the project on specified remote url , origin is the remote short name)**

**git remote –v : (it will show the url of remote repository of current project)**

**git push <remote-shortname> <branch-to-push> : (it will push the specified branch to the specified remote repo)**

**git pull <remote-shortname><branch-to-pull> : (it will pull the specified branch to the from remote repo to local repo, it also move local master)**

**git clone <git-repositry-link> : (to clone the github repository from github)**

**git push : (push the local repository to server repository)**

**git fetch <remote-shortname> <branch-to-push> : (fetch changes from remote central repositories, it does not move local master we have to merge it)**

**git merge <remote-shortname>/<branch-to-merger-with> : (fetch changes are merged by the local changes)**

**git pull : (it will fetch and merge both)**

**git shortlog : (It is used to see how many commits each contributor has added , -s flag is used show just no of commits each developer has made , -n flag order numerically)**

**git log –author=”<authorname>” : (Used to search commit made by specified author)**

**git log –grep=<char-to-search> : (Used to search commit with specified char in messaged)**

**git remote show origin : (Show details of origin)**

**git remote add myremote <repository url> : (this remote command will link local repostiry to remote repository)**

**git branch –dr <branch name> : (deleting remote github branch)**

**=========================== Rebase =============================**

**Git rebase –I <Head~comits-to-combine> : (combing multiple commits in a single commit)**

**git rebase <branch name> : (rebasing the specified branch with current branch)**