Git init

it will initialize the git repository

Git status

It will show the current status of the project like which files are tracked and which not.

Git add .

it will add all the untracked file to staging(tracking) area.

Git commit

It will take a screenshot of the project at the level it will currently be at.

Git restore –staged filename.txt

It will remove the tracked/staged files from the staging area to the untracked area.

Git log

It will show all the history/commit of the project from start to end.

Git reset [commit hash code like 45h2jkl5h235jkh2345234k523u3]

it will delete all the commit that was above on this commit (not this commit).

NOTE: The commits that we deleted now, all the added or modified file be come into the staged area from commit area.

Git stash

This command will store the modified and added file that was in staging area to the somewhere and now whenever I need those files I can fetch them. Those files are not committed nor loosed and nor saved in the project history. And when you do git log the message will be working tree is clean. There will be no sign of the files that you added or modified.

Git stash pop

Fetch all the files and folders from the backstage to the staged area

Git stash clear

It will delete all the files and folders that were stashed or that was in the backstage.

Git remote add origin URL

It will add the local project to the remote project

Git remote -v

It will tell that on which the current project is linked.

Git remote add upstream URL

it is similar to "**git remote add origin URL**" .

For example: we want to contribute to some project on git hub we can not make changes directly to the someone other projects. So we fork that project after that a copy of that project start showing on our GitHub profile also. Now we can clone that project and the URL will be "**git remote add origin URL**" this URL will refer to our GitHub profile where the copy of the project is placed **But** if we want to add the upstream to our copied project then we use this command "**git remote add upstream URL**" here the url is the one from the original project from someone others accounts.

Git branch branch-name

It will create a branch with the name of "branch-name"

Git checkout branch-name

It will switch from the current branch to this "branch-name" branch.

Or I say the head will start pointing towards the "branch-name" rather then the current branch we were having.

Git push origin jamal -f

When to use this ?

For example: you made two commits now you have deleted one commit from your local branch using reset and stash commands, but as you know the commits on local and remote are interlinked so at this point the remote branch has two commits but our local branch has one commit, for reflecting the local changes to the remote we use the above command with -f flag it means force. Now we are forcing the local changes on the remote branch.

⇒If you want updated your forked project with the main project the following command is helpful.

Git fetch –all –prune

It will fetch all the commits and the files that my forked project is behind to the main project. “prune” means it will also fetch all the commits that was deleted.

Git checkout main

Switch to the main branch

Git reset –hard upstream/main

It will fetch all the commits and the folders/files that your main branch was behind to your remote main branch.

Now my main branch is as the original project main branch.

Git push origin main

Why use this command? Our main branch was already updated ?

The reason is that the files and folders were on remote branch not on local branch so we have to use this branch to push the remote change of main branch to the local branch.

⇒You can also use the following command instead of above command to make your forked project updated with the origin project main branch.

Git pull upstream main == git fetch –all –prune and reset –hard upstream/main

And then

Git push origin main

=>Rebase

Git rebase -i hash_code_of_commit

It will show a new terminal window where you have to do pick and squash if you squash any commit it will be merged into the above commit that is picked. Like

Before squash:

Pick jh2345 1

Pick jh2345 2

Pick jh2345 3

Pick jh2345 4

After squash:

Pick jh2345 1

S jh2345 2

S jh2345 3

S jh2345 4

If using vim use the following command otherwise you are good to go by saving the file

After this hit ESC key then :x it will give you the option to write the commit message for those all the squash and pick commit.

=>Deleting the commit and also the files and folder that were made in that commit at once

Git reset –hard hash_code

The hash code will be the one from above all the commits you want to delete or reset.