Basic commands:

* Git Init : initialize repo
* Git add . : add all the files to the repo
* Git commit -m ‘message’: to the repo
* Git status: know what is going on

We can also remove the staged file by using git restore –staged names.txt

To see the history of the project we can use: git log

**How to remove the commit?** We cannot remove any random commits because it uses hash and all the commits are built on top of each other, but what we can do it go to the point from where we do not need any other commits:

Text

Description automatically generated

Command: git reset {commit\_hash} -> this will reset to the commit we will paste

**When we want to put changes in backstage, meaning when we want changes only when we need it:** we can use “***git stash”*** we use this when we do not want to commit but at the same time do not want to lose any changes. When we want to get those changes, we can use “***git stash pop***”and it will recall all the changes that has been made. If we want to discard the changes, we can use “**git stash clear**”

# Connect to Github

Create the repo, copy the link and use this command: “***git remote add origin ${link\_of\_repo}***”

To check all the available remotes, we can use: “**git remote -v**”

Now our local repo is connected to github.

To push all the commits: “***git push origin main***”

# Branches

By default, the name of the branch is “main”. What is the use of the branch? When we are working on a new feature, always create the new branch. We should never commit on the “**main**” branch.

Create new branch: “**git branch feature**”

**Head** means the pointer that says all the new commits will be reflected on this branch. It will have the “**\***” symbol. We can change the pointer using “**git checkout ${branch\_name}**”

Once the code is finalized, we can merge the branch using command: “**git merge ${feature}**”