Github is a version control repositry and Internet hosting service mostly used for code. Founded in February 8, 2008, it is used for distributed version control and source code management by software developers and coders around the world. Tom Preston-Werner created Github and the site was launched by him in April 2008 along with Chris Wanstrath and PJ Hyett who are also the co-founders. Some similar platforms are GitLab, BitBucket, SourceForge, Gogs,etc. We need to use such platforms to implement version control of our projects and to manage the source code effectively.

The things I learnt from the online GitHub tutorial are as follows:

- To initialize Git repository, we use:

git init

- To see what the current status of our project is, we use:

git status

- To tell Git to start tracking the changes made to the file octocat.txt, we add it to the staging area by using this command:

git add octocat.txt

- To store our staged changes, we run commit command as follows with a desription in double quotes on what we have changed:

git commit -m "Add octocat story"

- We can use wildcards to add many files to the staging area of same type whether they are stored at root or in some directory. For example to add all the txt files, we use:

git add '\*.txt'

- We use git commit -m again to add all the text files from staging area.

git commit -m 'Add all the octocat txt files'

- We can use git log to see all the changes we have committed so far.

git log

- To push the local repo to remote repo we use git remote add “remote\_repo\_name” “remote repo url”

git remote add origin <https://github.com/try-git/try_git.git>

- The push command tells Git where to put the commits. To push local changes to the origin repo, we use:

git push -u origin master (-u tells git to remember the parameters, so next time we can simply use git push)

- We can check for changes by other people on our GitHub repository by other people and pull down any new changes by running:

git pull origin master

- For most comparing with recent Git Commit

git diff HEAD

- To see the changes we have staged

git diff --staged

- We can unstage files by using the git reset command.

git reset octofamily/octodog.txt

- To get rid of all the changes since the last commit for octocat.txt

git checkout -- octocat.txt

- To create a branch called clean\_up:

git branch clean\_up

- To see the current branches:

git branch

- We can switch branches using the git checkout <branch> command

git checkout clean\_up

- Use git rm command which will not only remove the actual files from disk, but will also stage the removal of the files

git rm '\*.txt'

- To commit our changes again:

git commit -m "Remove all the cats"

- To switch back to the master branch so we can copy (or merge) our changes from the clean\_up branch back into the masterbranch

git checkout master

- If we're already on the master branch, so we just need to tell Git to merge the clean\_up branch into it

git merge clean\_up

- Use git branch -d <branch name> to delete a branch

git branch -d clean\_up

- To push everything we've been working on to our remote repository use git push

git push