***GitHub & Git Notes***

**General Points**

1. ***– git add .*** = will track files and changes in git.
2. ***– git add <file-name>*** = change the tracking stage of specific mentioned file.
3. ***– git commit –m “<commit-message>”*** = save your files in git.
4. ***– git push origin <branch-name>*** = makes the code on your local devices upload to GitHub.
5. ***– git push –u origin main =*** -u here says that whenever we use only “git push” the git will automatically upload to origin main as **“-u”** work is to assign default branch to push your code when no branch is mentioned. It goes not affect the basic way to push or pull your code. –u sets a default upstream.
6. **Origin** = it is a naming convention pointing to original repository. It refer to the default remote repo connected to local repo or where it was cloned. I guess the local repo is treated as copy of remote repo.
7. ***– git pull origin <branch-name>*** = pulls changes from the branch on remote site with similar name as requested branch name of the origin remote repo regardless of the current branch tracking configuration.
8. ***– git remote add origin <remote-repo-URL>*** = the local repo and remote repo which is on GitHub connects with this command.
9. ***– git branch –M main =*** renames the default branch to main which may be called master. *–M* represent –force or –move.
10. ***– git status =*** check the status of tracked files and commits.
11. ***– git checkout <branch-name>*** = moves the attention of git to mentioned branch.
12. ***– git checkout –b <branch-name>*** = creates a new branch of that specified name and switch the attention to the newly created one.
13. ***– git clone <repo-URL>*** = will clone any remote on GitHub to your local machine. But if you want to make a copy of other’s repo then fork it repository using your account then clone the forked repo in you local machine and make changes. You can create a pull request to the original owner of that repo and ask him to review the pull request, if the change suites him you can **COLLABORATE** or become a **CONTRIBUTER** on the original git repo.
14. ***– git remote –v*** =helps to see the URL your remote repo which your local sends the data.
15. {git pointer on<branch\_1>}: -- git diff <branch\_2> = to check the difference of files and there data between two separate branches.
16. {git pointer on<branch\_1>}: -- git merge <branch\_2> = to merge the two different branches offline locally.
17. – git commit –am “<Commit-message>” = -am stands for add modified. With this –am prompt we can commit a modified file without changing its stage because it is been tracked but git already. –am will not work for files who are newly introduced as its not modified file as tracking is important for git to do its job.
18. – git diff = to check the difference in the old and new code.
19. –git reset <file-name>| --git reset = unstage and staged file.
20. –git reset HEAD~1 = this command will uncommit and unstage the previous commit by one step. I guess HEAD points to list of commits but from bottom to top

**::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::**

**Q&A**

* What is publish branch in Git hub? And why when I pushed by code via VS code the same did not reflected in Git hub website and after I published the branch the changes reflected!
* Do people make a branch for update, bug fix etc. do the changes in the code deploy on GitHub, get merged with main branch and then delete that very branch, is this practice is good?