GIT :

1. Local tool installed on your computer.
2. Version control for code
3. Organized in “repositories”

GITHUB :

1. Cloud hosting provider for “Git repositories”
2. Collaboration in and contribution to development projects
3. The world’s largest development platform

GUI – Graphical user interface

CLI-command line interface

The fastest and most common way to efficiently work with Git and GitHub is using CLI.

Few commands ---

-

Pwd – prints current directory

Ls – list items inside the current folder

Cd / - change the directory to root directory.

Cd – to change to home directory.

Cd ..-to move one directory up

Mkdir <dir name> - to create a directory.

Touch <file ame with extension> - to create a file inside any directory.

Rm <file name>– to delete a file.

Rmdir <directoryname>-to remove a directory.

GIT – Consists of two parts.

1. **Working directory (current project folder)** – is the current project directory. Basically, the current code version we interact with now where we implement changes.
2. **Repository (All tracked files and folders)** – This repository is a hidden folder/area which we can’t see. But it has all tracked files and folders, meaning all files and folders git manages for us.

If we add new file, git keeps one initial version of that file and it tracks all changes by comparing the current state with the latest state.

Commits- Code snapshot.

Branch- where commits are stored.

Git init – to initialize the repository.

1. Git add <filename/.> – All mentioned files are being tracked and the files are added to staging area. Staging area is the area where the developers want to include this file or changes applied in the next snapshot. That should be stored. It’s in between our working directory and the commit that we want to create.
2. Git config --global –edit the credentials.
3. Git config –global user.name – to get the already set username.
4. Git config –global user.name “”– to set the new username.
5. Esc:X then Enter – it’s the default key combination to access.
6. Git log – it gives the overview of all commits inside our current branch (master or feature) with commit id, author, and date.
7. Git branch – shows us all branches.
8. Git checkout -b “<new branch name>” – this creates a new branch and checks that out.
9. Git merge <branch name> - We must be in the main branch where we want to code to be merged then hit the command
10. To delete a file from the git we do by these options:
11. Delete the file or folder manually and commit the changes.

If we delete the file from working directory then we have to add that changes to the staging area first

1. git rm <file name> - Go to the directory whr this file is present then run command to delete the file. By this way Git automatically removes the file from the staging area for us
2. Git reset --hard HEAD~1 – (HEAD is just a pointer) This command resets and goes one commit back. We can also write like git reset –hard HEAD~(2/3/4/5) to go back steps back.
3. Git branch -D <branch name> - We must go to another branch thn try deleting a particular branch.
4. Git checkout -- . – (to revert upstaged changes) This will simply reset the branch to the stage of the latest commit.
5. Git reset <file name> - will reset the staged changes.
6. Add the URL for the remote repository where your local repository will be pushed.
   1. git remote add origin <REMOTE\_URL>
   2. git push -u origin main