Git Github

Open VS terminal

Git → Local

→ git clone "repo_link (origin)" → This will bring all the files from repo

Open Repository > Code

There will be link.

This command gets git repo into local machine.

→ cd "folder name"

This command will open the folder to work on

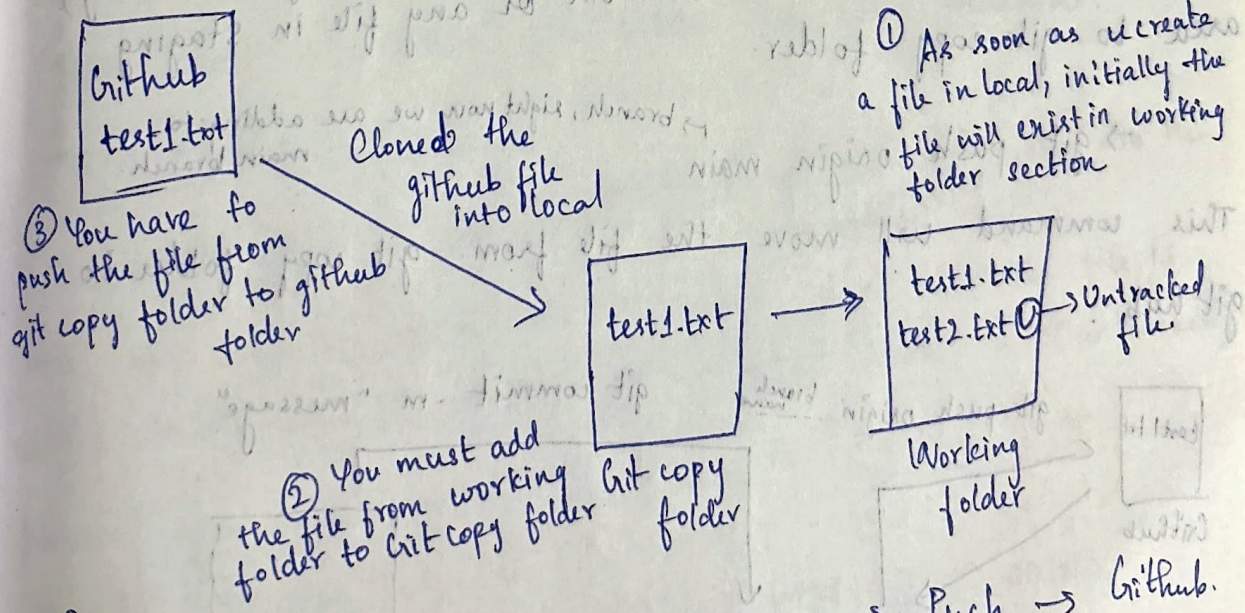
Local → Git.

Let us say u are creating new file inside git folder on your local machine "test2.txt"

This file will be indicated by "U" symbol meaning untracked file (only in local not synced up to git)

→ git status

This command will give status of folder u r working on.



How:

Insert (Any modification) → Add → Commit → Push. → Github.

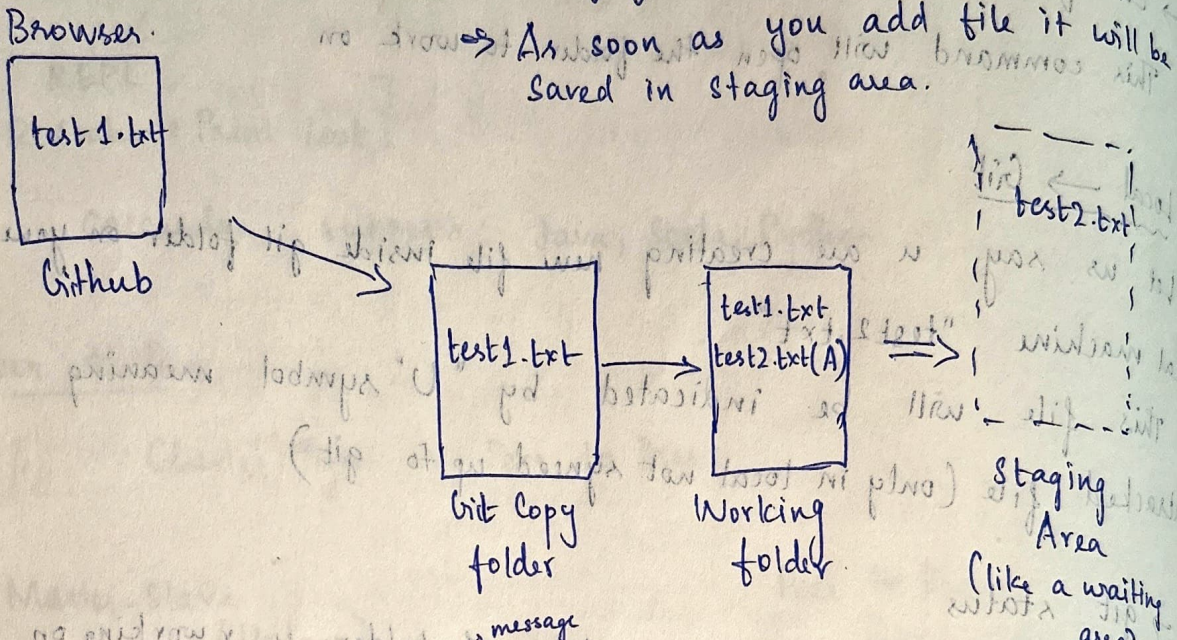
→ git add "test2.txt"

This command will change the symbol from "U" to "A" indicating the file has been added to ~~git copy folder~~ staging area waiting to be committed.

→ git status

This will show that the file has to be committed.

This means the file is in staging area.



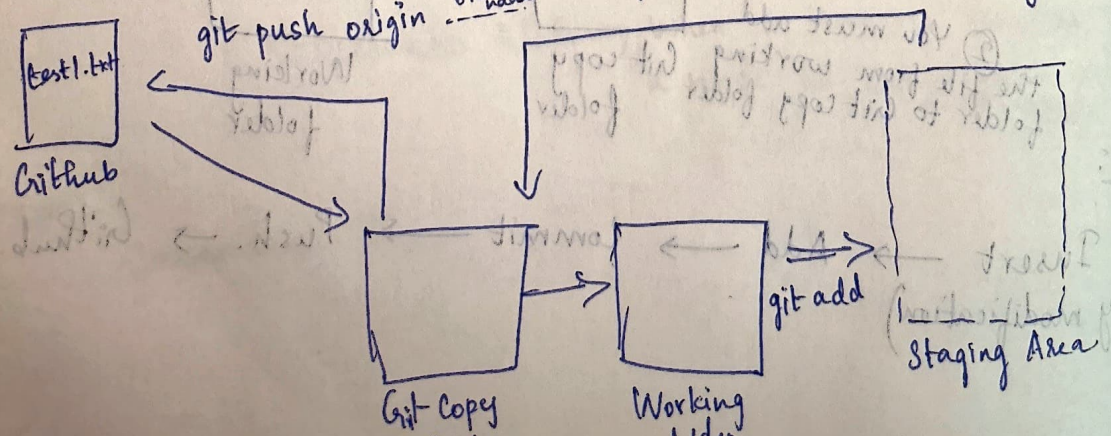
→ As soon as you add file it will be saved in staging area.

→ git commit -m "test2.txt file added"
↳ message
↳ commit message (mandatory)

This command will move test2.txt or any file in staging area to git copy folder.

→ git push origin main
↳ branch, right now we are adding to main branch.

This command will move the file from git copy folder to git hub.



Now let's say you need "specific file from Github.

git clone command will get all the file

Let's say on Github you created a new file after you cloned the folder into your local machine. Now you just want the newly created file into your git copy folder/working folder in your local machine.

→ git pull

This command will update the git copy folder/working folder with changes made on github(browser)

Folder 1 > Folder 2

cd.. (Goes one step back to the folder.

Folder 1.

Let's say you are creating folder in your local machine and want to upload it to git repository

You have to create file/folder in local machine.

cd folder/filename

git init ⇒ This command will add .git file into the folder

This file will be added to the folder and whatever file is within that folder, those will be marked with 'U' (untracked)

Now you can either add file one by one using

git add "filename".

→ To add all the file within the folder all at once.

→ git add . ⇒ This will mark all the file as "changes to be committed" status, meaning all the file will be moved to "staging" area

Now → git commit -m "message"

This command will commit all the modification.

If we have push from local machine, we need origin link.

1) Create a repository/branch in github
2) Get the origin link.

→ git remote add origin "origin link"

This command will connect to github repository

Now run

→ git branch ⇒ To get branch name

→ git push origin main → branch name

You can change branchname

→ git branch -M branchname

To get list of branches

→ git branch -a

To change from one branch to another branch

- git checkout branchname you want to work

Creating a new branch:

→ git branch newbranch ← branch name.

To navigate to new branch

→ git checkout newbranch

To add the new branch to staging area and then to git

→ git add . ⇒ All modification will be moved to staging area.

→ git commit -m "Added new branch" ⇒ All modif. will be committed

To push the new branch into git:

→ git push origin newbranch.

Now to merge the new branch to main branch ⇒ Raising a PULL req.

Creating pull requests:

Pull request on Git → Select the branches → Give message/description → Create pull req.

To merge the request

User will be notified with pull req. user can click on merge button → Confirm merge.

(code will be merged with main branch.)

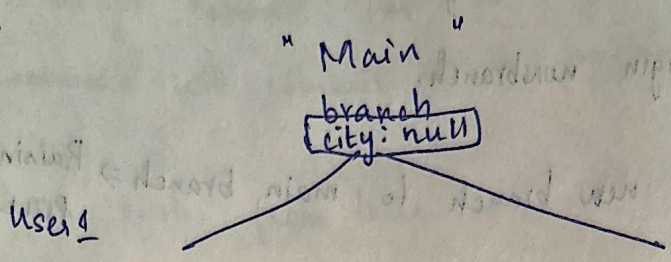
Now let's say user 1 working on branch 1
user 2 working on branch 2.

Now both the users have created branches from "main"

They both started working on same line of code, In this case, if both the users have created pull requests for merge.

While the reviewer can merge one branch with main without issue, but if tries to merge the next branch to main, then conflict arises as both the branches contain changes on same line of code.

eg:



branch 1

city: chennai

↓

Pull request 1

branch 2

city: bangalore

↓

Pull request 2

Meeging:

Main ← Pull request 1 ← branch 1 (Succes

Main ← Pull request 2 ← branch 2 (Conflict

↳ In this case user have to click on resolve conflict

Once the user clicks on resolve conflict button, user can review both the changes on screen.

eg: >> branch 1
city: chennai
>>> branch 2
city: bangalore
>> main

User can keep the code they want to merge and delete rest of it

eg: city: chennai

Click on Resolved button, now user can merge the cod