* What is Version control :- let suppose we have a software 'A' and we need to change something in that software 'A', so we do change and after that we say, this is the updated version of the software 'A'. and also have previous version of software 'A'.

Eg:- window 7 ⟶ window 8 ⟶ window 10 ⟶ window 11
(window 11 ← updated version of "windows")

* So after all this, the System which manage all this is know as version control system.

* Need of version control system :- If there are many Developer do changes is software 'A' at a time & after that they need to update the main Software so they can collabrate with each other on one plateform.

* Principle of Version Control :-

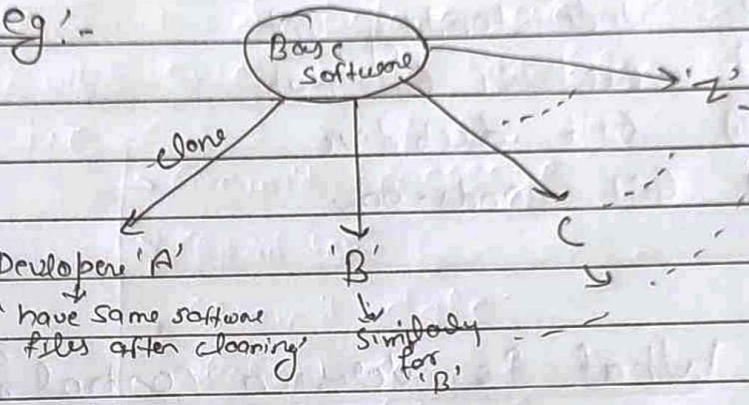
Result = Start (Base software) + Sum of all changes

* * * "GIT" is a "distributed file system".

↓

"De centralized version control system"

eg :-

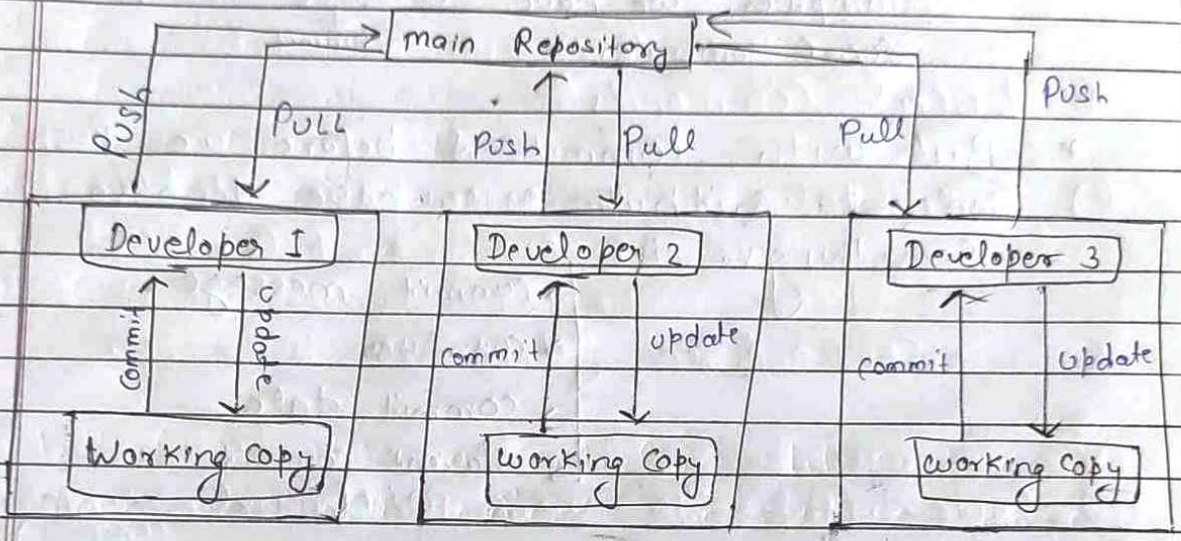


* It clones the whole software file at local computer, ~~when~~ to every developer. that's why it take more storage space.

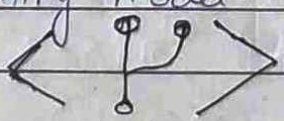
i) Commit mean → Save
ii) Clone mean → Copy

Understanding GIT

Distributed Version Control System



Git & Git hub :-

GIT	GIT Hub
i) first develop in 2005	i) Git hub is designed as a Git repository hosting service
ii) Git is installed and maintained on your local system (rather than in the cloud)	ii) Git hub is exclusively cloud-based
iii) One thing that really sets Git apart is its branching model. 	iii) You can share your code with other, give them the power to make revisions or edits.
iv) Git is a higher quality version control system.	iv) Git hub is a cloud-based hosting service

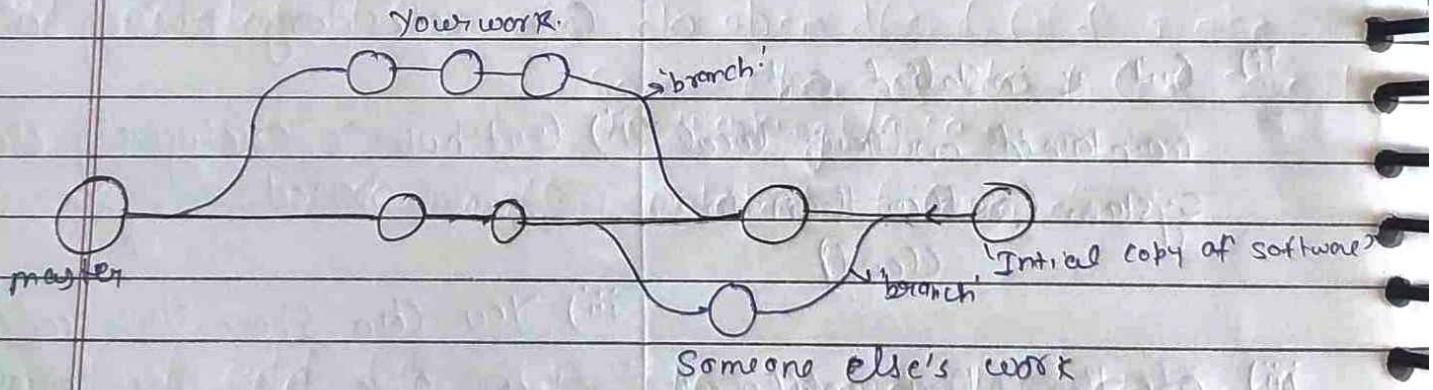
GIT deep-dive

1) * **Commit** :- commit means Save, i.e (whatever changes we have made on my local computer we have to save it in our local computer. → commit

* But with commit & before we Push the code to Github we have to also add Some things. i.e

- commit message
- Commiter
- commit date
- Parent commit
  - i.e (intial satate before latest commit)

2) * **Branches** :-



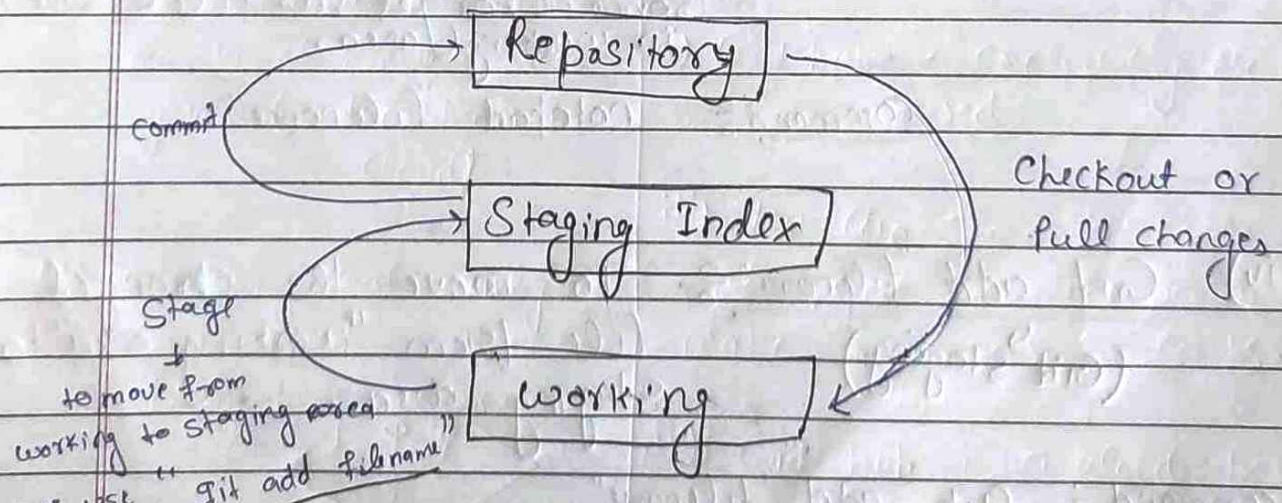
GIT Hands-on

i) Installing git
ii) Creating a github account
iii) Creating a new repo on Github
iv) cloning a repository (repo)
v) three stages of git
vi) Local file status
vii) Git staging / un-staging
viii) Git diff

* **Cloning a repo :-** cloning a repository (repo) means, make a copy of new repo to your local computer.

Command :- git clone https://github.com/yourreponame/
~~git clone~~

* **Three stages of Git :-**



Git Commands

There are various commands in Git which are used for "Local computer & Remote computer"

* Local Commands :-

i) **Git Status :-** To view the status of current file. i.e (local file status)

Command :- git status

ii) **Git diff :-** To view the difference b/w initial file & file after changes

Command :- git diff

iii) **notepad "filename" :-** To open the file & make some changes to file & save it in working area.

Command :- notepad "filename"

iv) **Git add "filename" :-** (Git staging) To move the changed file from "working area" to Staging area.

command :- git add "filename" → "for specific file."

command :- git add.

↳ to add all changed files from working area to staging area.

v) Git unstage :- To move file from staging area to working area. ~~remove all changes~~.

Commands :- git reset Head "filename"

OR

git restore --staged "filename"

OR

git rm "filename"

vi) Git Commit :- After we move file from working area to staging area then we use commit command to save the file locally.

Command :- git commit

Note :- when we use "git commit" command we have to also provide a commit message & save that commit message, if we not give commit message then the file will be not commited locally.

vii) Git log :- It is to view all the commits we have made from intial file to leftest change.

Command :- git log

viii) Git checkout :- checkout command is used to create new branch and also used for move from one branch to another branch.

Command :- git checkout -b "branch name"

* Inside that branch we have to add file using "Touch" command. & after that we can make changes in that file. which is in different branch.

* To get back to previous (i.e initial) branch we also use "checkout" command.

Command :- git checkout "Previous (initial) branch"

ix) Git branch :- Git branch is to view total number of branches & we are on which branch also.

2) * Remote Commands :-

i) Git Push :- Push command is used to make changes visible on "Remote" which we have done on "locally".

Command :- git Push

ii) git fetch :- fetch command is used to copy which changes we have made on Remote to locally. but it does not apply changes locally.

Command :- git fetch

iii) git Pull :- Pull command is used to apply changes locally which we have done on Remote.

(fetch + merge) "do both"

Command :- git Pull

iv) Resolving conflicts :- if someone do changes remotely and Push & commit before me and the Parent is same for both then when I will Push my changes it will not do that. because someone did changes already in that parent. So we have to Resolve conflict.

To Resolve conflict we have to "Pull Privious Changes" & Resolve it in note Pad by Removing "<<<< >>>>" → this

And after change we will again add & commit with commit message then we will "Push" it on "Remote" then we can see all changes on the Remote.

Note:- "gitk" command is a UI to see all changes & commits visually.

* "git init" command is used to creat a new Git repository.