

## # Working with Git Ignore

• gitignore

Just for example → अगर कोई database या कोई logs कि file बनी हो तो उसे उसे commit नहीं करना हो, तो एक .gitignore नाम कि file बना के Database & logs कि file को लिखा देगे तो ये file नहीं commit होगी।

Git Clean

↓  
Clean the Dummy file

→ git clean -n  
→ git clean -f

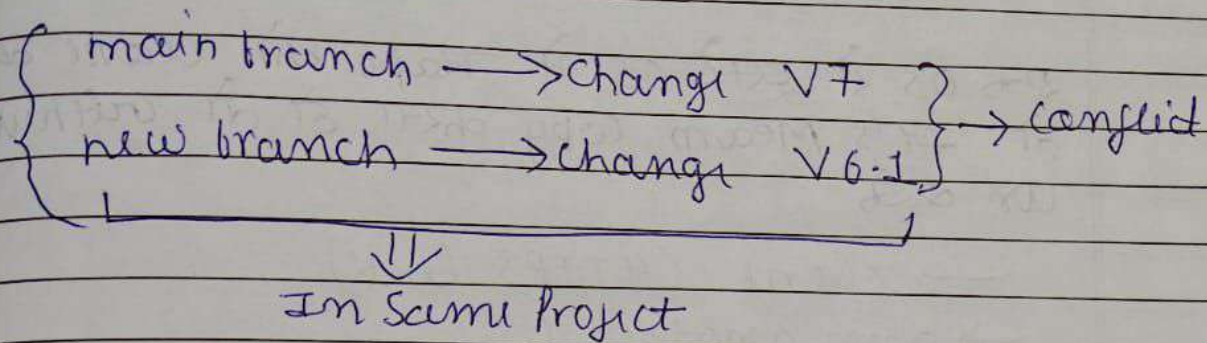
## # What are Git Tags

Commit Id को एक नाम दे दे जिससे याद करना आसान हो जाय।

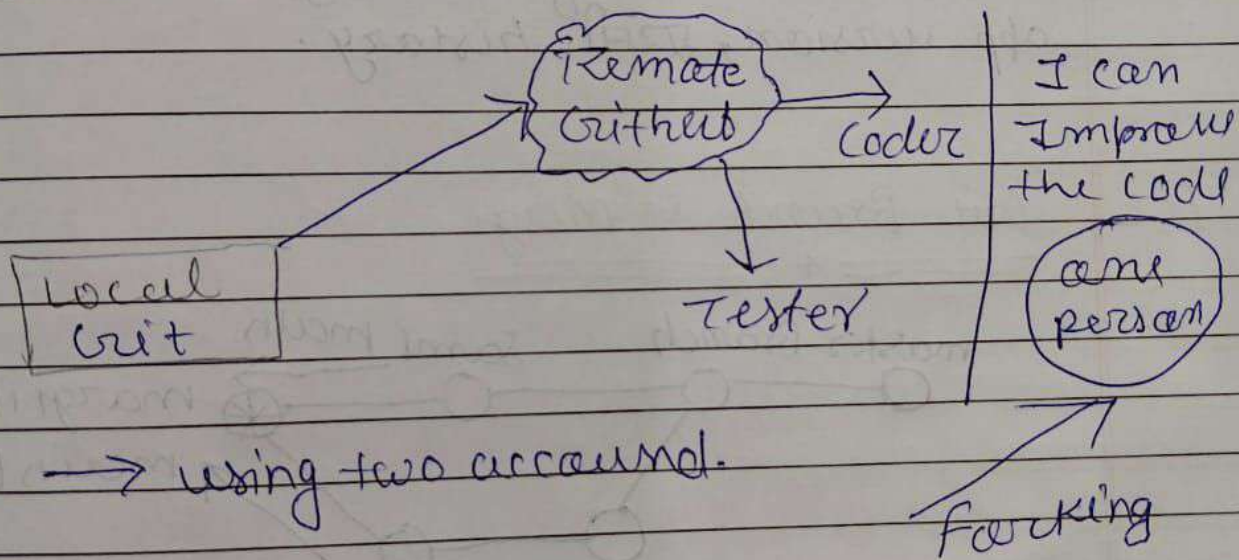
Git  
→ git tag -a name -m "This is V7"  
→ git tag  
⇒ git push origin --tags  
⇒ git tag -d <tag-no>



## What is Merge Conflict



## Git Forking and Pull Request





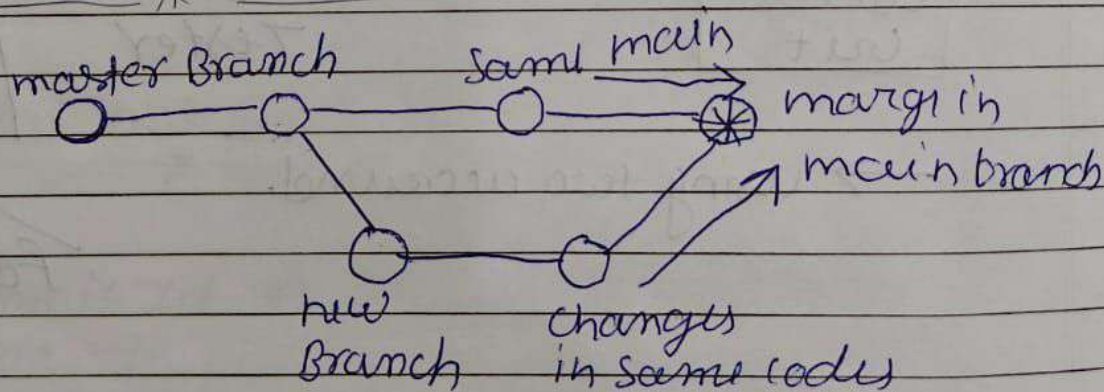
## # Understand Git client \*

एक OS से दूसरे OS में Repo को clone करना हो तो git's Meam copy करना हो तो GitHub का use करें

- clone (HTTPS link)
- open another OS
- git clone url
- you have seen the all file in another OS.

इसके साथ -2 सारे log सब चले आ जाता है, सारे app version, पुरानी history.

## # Git Branch & Merge



- git branch branch-name (create new branch)
- git checkout branch-name (change branch)
- git branch (check all branch)
- git merge branch-name (merge the branch)
- git Push - main branch
- git Push origin branch-name (to accept it in the branch and it is pushed to github to sync it)



## GitHub

classmate

Date \_\_\_\_\_  
Page \_\_\_\_\_

अब हम सारे changes को कही Remotally store करने हैं जिससे कुछ issue solve हो जायेगा जैसे data corruption का koi issue नहीं होगा, Team में काम कर रहे हो तो इन changes को provide करना etc. (Remote Repository)

- Create a repo
- अपनी local file को Remotally में Push करना होता है पहले —
- `git remote add origin https://github.com/ARPIT311095/RepoName`
- `git branch -M main`
- `git push -u origin main`
- connect to Github.

### # Flow for Git PUSH

after changing the file content  
after commit  
then → `git push` (Tab ja ke github में jayega)

### # Understand Git Pull

Remote to local system

- add new file
- then commit
- `git pull`



Case 3 { में पहले change चाहता हूँ like - V4 then add  
उसके बाद कुछ गलत add हो गए तो उस Type  
के case में -

→ git restore --worktree filename

Case 4 { commit karu ché baté →

→ git reset --soft HEAD^ } back the  
--hard HEAD^ } previous  
file version

## # Useful Tips of (git log)

- git log -P-2 (last two commit with diff)
- git log --stat (summary of changes)
- git log --pretty=oneline (one line commit)
- git log -s "hl" (hl add kiga)



## Get old Version with Git Checkout

Rapo se man lo mere pass बहुत सारी files है और सब को मुझे check करना है

⇒ अगर मुझे latest version से पुराने version कि files पे जाना हो तो checkout का use करेंगे और अगर पुराने version से नये version पे आना हो तब भी।

- version Id is Required

→ Git checkout VersionId -- space \*

→ Git checkout master -- \*

## Working with Git Restore

अगर file में कुछ galti से add हो गया हो तो उसे कैसे सही करें

→ git restore --staged

→ git status

→ git restore

Case - 1  
without  
add

add operation Case - 2

↳ staging

without commit



## ## Project 1 → Git commit \*

→ index.html

→ git status

→ git add index.html (\*)

→ index.html changes the status check using git status

→ commit the file → git commit

-m "message"

→ git log (check the commit status)

commit → eggplant → salad

## Project 2 → version-2 & version-3

changes the same property in index.html

→ git diff (different V1 & V2)

## ## tracking the older file (show)

file Id → Requirement jisko track  
करना है।

→ git show Id: filename



## Configure GIT with username & Email

- VS code Editor (using)
- Git hub Terminal — Git bash.

```
git config --global user.name "Parth"  
git config --global user.email arpit9090@  
git config --list
```

## Creating First Git Repo ⇒

Storage Box hata hai Jahan ap apne project ki Saari files aur unka history Rakhta hai. Me Local me hota hai. tracking the files. Meरे Pass एक folder hai अब मुझे इस folder मे Saari files rakhni hai और उसको मुझे track करना है।

अब मे' Git Repo को बताऊंगा कि इसी मे' Repository बना रहा हू अब इसके अन्दा कि file को track करेगा है। ↓

git init



first step

↳ ls -la

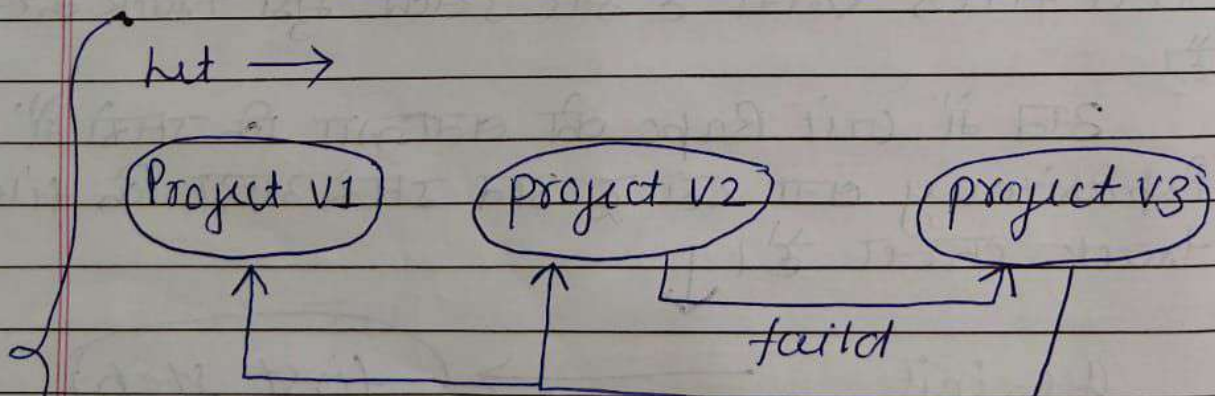
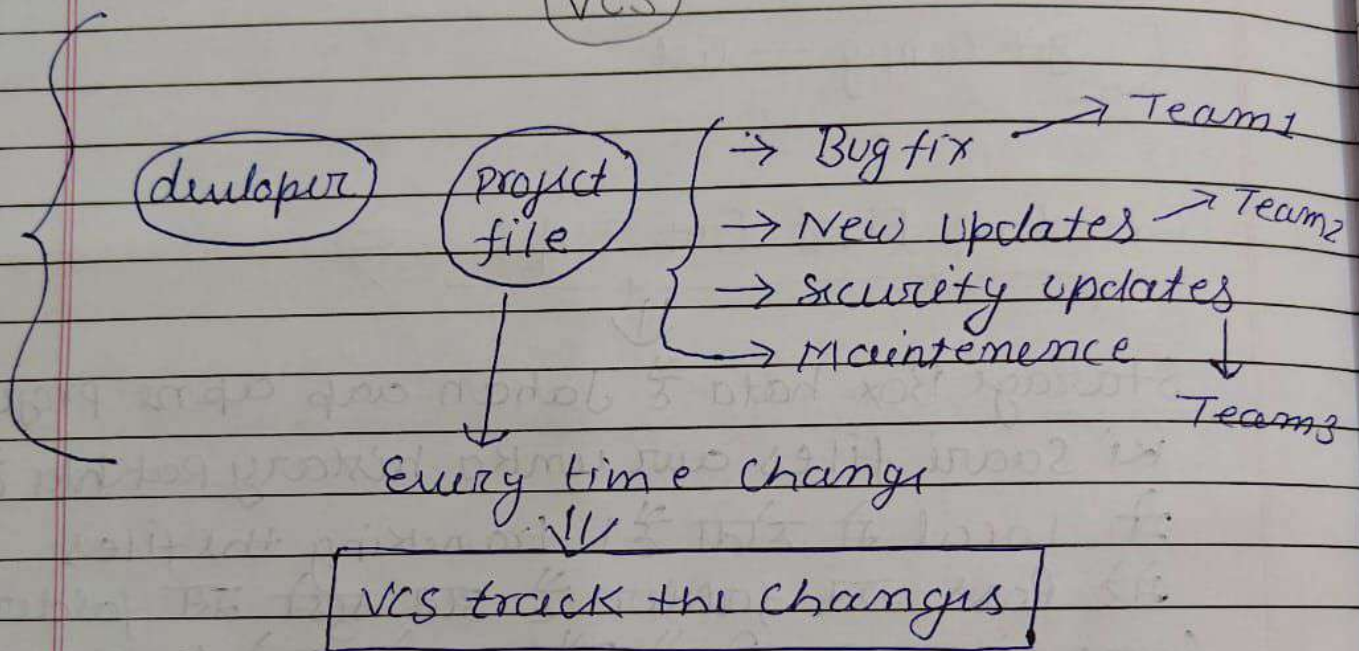
↓ (.git)



## Git and GitHub

Git is a distributed (VCS) → Version Control System

मे सारा काम आपके Local System में रहेगा।  
(VCS)



VCS को use करके हम Previous version pe jaa sakte hai

- Multiple people can work on a project → Same