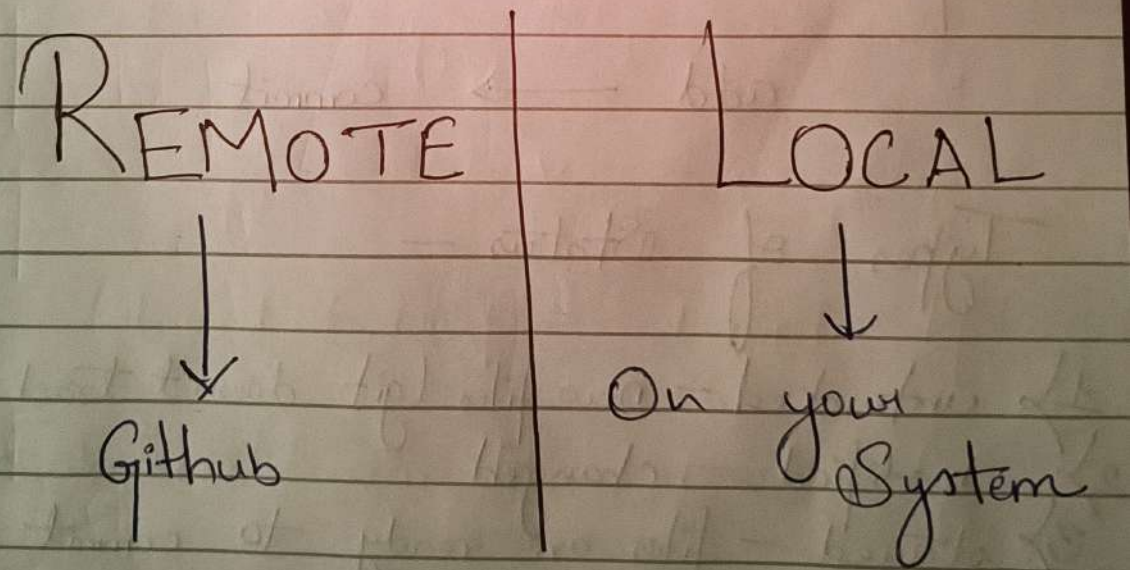
Git

To check if git is installed or not:

```
git --version
```

Configuring git:

```
git config --global user.name "My Name"
git config --global user.email "email@gmail.com"
git config --list
```



Clone - Cloning a repo on our local machine

git clone ← some link →

Status - displays the state of code

git status

→ Jaise hi hum apne code ko modify karte hai, waise hi hume 2 steps follow karne hote hai.

add ⟶ commit

Types of Status -

1) untracked - new files (git doesn't track yet)
2) modified - changed
3) staged - files are ready to commit
4) unmodified - unchanged.

change / new file
(modified) (untracked)

↓

add (staged)

↓

commit (unchanged)

Add - adds new or changed files in your working directory to the Git staging area.

git add <file name>

OR

git add . {To add all files}

Commit - it is the record of change

git commit -m "some message"

push – upload local repo content to remote repo.

git push origin main.

- git → Main cmd
- push →
- origin → remote github ka naam (can be changed)
- main → branch

Init – used to create a new git repo.

git init

git remote add origin <– link –>

git remote -v (to verify remote)

git branch (to check branch)

git branch -M main (to rename branch)

git push origin main.

(Whenever we create a new repo locally, it is not a git repo. We can check this by doing ls -a command. If . .. shows, it means it's not a git repository.

When we hit git init, and then do ls -a, it will show . .. .git

Create a GitHub repo from Terminal

gh auth login {Login using GitHub CLI}

gh repo create your-repo-name --public --source=. --remote=origin --push

--public can be changed to --private

--source=. means the current directory

--push will push all the code instantly.

NOTE: Instead of writing git push origin main again and again to push the code, if we are working on the same branch for a long time, we can use:

git push -u origin main

-u → to set upstream.

Now we just need to write git push to push the code.

Workflow:

GitHub Repo

↓

clone

↓

changes

↓

add

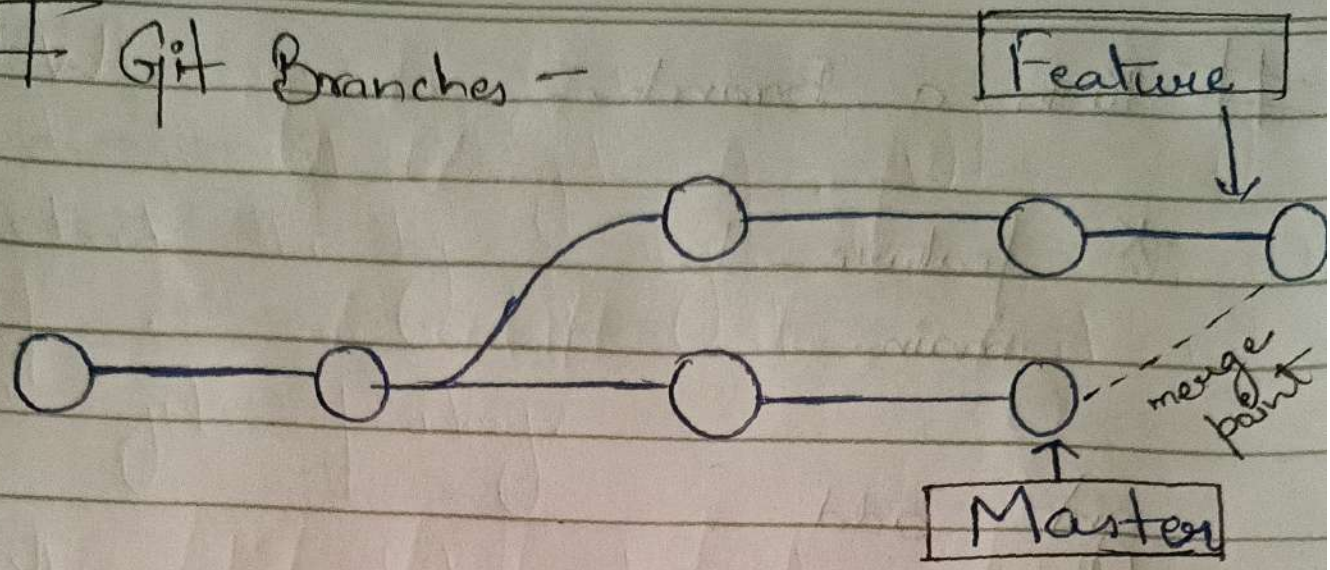
↓

commit

↓

push

Git Branches -



Branch Commands -

git branch (to check branch)

git branch -M main (to rename branch)

git checkout <- branch name -> (to navigate)

git checkout -b <- new branch name ->
(to create new branch)

git branch -d <- branch name ->
(to delete branch)

Note: Jis branch pe ho, usko delete nahi kar sakte nahi toh error aayega. Move to a newer/ other branch first.

Push a Branch:

* feature-1
  main

git add .

git commit -m "New features"

git push origin feature-1

Merging Code:

Way 1:-

git diff <- branch name ->
(to compare commits, branches, files & more)

git merge <- branch name ->
(to merge 2 branches)

Way 2:- (Using Github)
Create a PR (Pull Request)

pull - used to fetch and download content from a remote repo and immediately update the local repo to match the content.

```
git pull origin main;
```

Resolving Merge Conflicts:

An event which takes place when git is unable to automatically resolve differences in code between two commits.

Undoing Changes -

→ Case 1 - staged changes

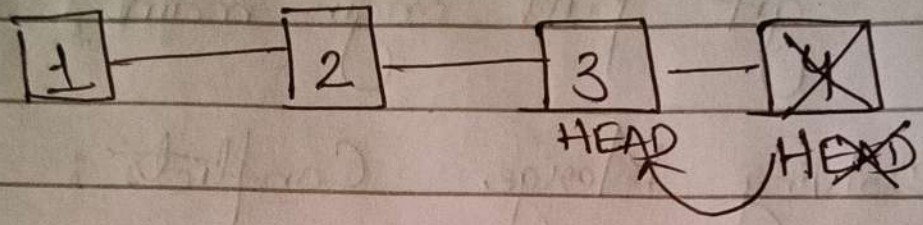
```
git reset <- file name ->
```

```
git reset
```

(To add huye par commit nahi huye)

→ Case 2 - commited changes (for 1 commit)

git reset HEAD~1



Use git log to check all commits.

→ Case 3- committed changes (for multiple commits)

git reset <- commit hash ->

git reset --hard <-commit hash->

Fork — To make a rough copy on your remote repo of someone else's repo.

To merge commits on your forked repo to the official one, create a pull request.

Note - Do not create unnecessary PR's. Always add something useful and then create a PR to become a contributor. Your commits can be seen by everyone who has your Github profile.

X — X — X — X