Essential Git Commands

For Version Control

What It Git ?

Git is a tool that helps developers work together on software projects. It keeps **track of changes in the code**, making it easy for multiple people to **collaborate** without messing things up.

Git manages your code versions, and **GitHub** provides a platform for you to store, share, and collaborate on those versions with others.

Git Commands

Configuring Git

Initialize a new Git repositor

```
git config --global user.name "name"
```

```
git config --global user.email "email"
```

```
git config --list
```

To view the configuration settings specific to a particular Git repository

Clone : Cloning a repository on our local machine

```
git clone <repository_url>
```

Init : Initialize a new Git repository

```
git init
```

version : Displays the installed Git version


```
git version
```

Status : Displays the state of the code

```
git status
```

add : adds new or changed files in your working directory to the Git staging area.

```
git add <-file name->
```

```
git add .          //adds all
```

commit : It is the record of change.

```
git commit -m  “some message“
```

amend : Changes to the most recent commit | --amend -m “New Msg“

```
git commit --amend "
```

remote : Add a remote repository

```
git remote add origin <repository_url>
```

```
git remote -v          // to verify remote
```


branch : independent lines of devlpmnt

```
git branch          // To check branch
```

```
git branch -M <name>     // to rename
```

```
git checkout  <b-name>  // to navigate
```

```
git checkout  -b <new-b> // Create new
```

```
git checkout  -d <b-name> // Delete b
```

push : upload local repo content to remote repo | 🚩**-u** upstream

```
git push origin <branch> or main
```

pull : Fetch and merge changes from a remote repository

```
git pull origin  <branch> or main
```

---

diff : to compare commits, branches, files & more

```
git diff <branch_name>
```

log : Display commit history

```
git log
```

Undoing Changes

Case 1 : **staged** changes

```
git reset <file_name>
```

```
git reset    // Reset all
```

Case 2 : **commited** changes (for one)

```
git reset HEAD~1
```

Case 3 : **commited** changes (for many)

```
git reset <commit-hash>
```

```
git reset --hard <commit-hash>
```

⚠ it removes all changes

merge : Merges one branch into another | with 2 ways

1. Using git **merge** command

```
git merge <branch_name>
```

2. Using **pull request** (GitHub)

---

untracked : new files that git doesn't yet track

modified : changed

staged : file is ready to be committed

unmodified : unchanged