# Git And GitHub

# Notes

Developer Sequence

4th September, 2025

# ## Git

Git is a distributed version control system (VCS) designed to track changes in source code during software development. It allows multiple developers to collaborate on projects efficiently by managing revisions, facilitating team coordination, and enabling version control for files. Git tracks changes in a repository, stores these changes with metadata like authorship and timestamps, and supports branching and merging to facilitate parallel development efforts. It's widely used in software development to manage codebase versions and ensure project integrity across teams.

- git  --version

# ## Git Hub

GitHub is a cloud-based platform that uses Git for version control and collaboration. It allows developers to store, manage, and share their code repositories online. GitHub provides features like:

* Remote Repository Hosting: Store and access Git repositories online.

-Collaboration Tools: Supports team collaboration with pull requests, code reviews, and discussions.

-Branching and Merging: Developers can work on different features or fixes simultaneously.

-Issue Tracking: Manage bugs and feature requests efficiently.

-CI/CD Integration: Automate testing and deployment with GitHub Actions.

-Security & Access Control: Control who can view or edit your repositories.

### Remote to local Config

## SETUP

Configuring user information used across all local repositories

- git config --global user.name “[firstname lastname]”

set a name that is identifiable for credit when review version history

- git config --global user.email “[valid-email]”

set an email address that will be associated with each history marker

- git config  --list

### Remote to Local

It show list configuration

- git clone <url>

- git status

- git add <filename>  or git add .

- git  commit -m "commit msg"

- git push origin main

### Local to Remote

- git init

- git remote add origin  <link>

- git remote -v

- git branch

#Leave current branch

- Switch to a branch:

- git  checkout <branch name>

---------------or----------------------

-git switch feature-branch

#Switch new branch

- Create and switch to a new branch in one step

-git checkout -b new-branch

---------------or----------------------

-git switch -c new-branch

- git branch -d <branch name>

- git branch -M main

A branch in Git is like a separate workspace that allows you to develop features, fix bugs, or experiment without affecting the main project.

- Think of it as a parallel version of your codebase.
- You can create, switch, and merge branches without interfering with other developers' work.

- git push -u origin main  (upstream)

### ### Merging code

- git diff <branch name >

- git merge <branch name>

This are the branches

1) main

2) front-end

3) Back-end

4) Testing

### ### create pull request

git pull origin main

* A Pull Request (PR) in GitHub is a request to merge changes from one branch into another, typically from a feature branch to the main branch. It allows for code review and collaboration before merging.

* yes, we can  download in local diff which are  done on remote.

## Undoing changes

- git reset <filename>

- git reset

* committed  changes

- git  reset HEAD~1

The command git reset HEAD~1 moves the current branch one commit back (HEAD~1 refers to the previous commit), effectively removing the latest commit while keeping the changes in the working directory.

git log-history of commit

-  git log

-  git reset --hard  [hashvalue]

The command git reset --hard <hashvalue> moves the current branch pointer to a specific commit (identified by <hashvalue>) and discards all changes after that commit, including staged and unstaged files.

## Resolve merge conflict

- git merge main

- git merge feture

Easy Additional:

1️⃣ Rename the Current Branch

If you are on the branch you want to rename:

```
- git branch -m new-branch-name
```

2️⃣ Rename a Different Branch

If you want to rename a branch without switching to it:

```
- git branch -m old-branch-name new-branch-name
```

Updating the Remote Repository (GitHub, GitLab, etc.)

**If the branch has already been pushed, you need to update the remote:**

```
- git push origin --delete old-branch-name  # Delete old branch from remote
- git push origin new-branch-name  # Push the renamed branch
- git branch --unset-upstream  # Remove tracking of the old branch
- git branch -u origin/new-branch-name  # Set tracking for the new branch
```

*You accidentally named your branch login_fet instead of login_feature.

```
- git branch -m login_fet login_feature           # Rename branch
- git push origin --delete login_fet           # Remove old branch from remote
- git push origin login_feature                # Push new name
- git branch -u origin/login_feature          # Track the new branch
```