🔷 Git Terminology

Term	Meaning
Repository (Repo)	A project folder tracked by Git that contains code and its history.
Commit	A snapshot of your code at a specific time.
Staging Area (Index)	A "waiting room" where files are added before committing.
Branch	A separate line of development inside a repo (default is `main` or `master`).
Remote	A copy of your repo hosted elsewhere (like GitHub).
Origin	The default name of your remote repo on GitHub.
HEAD	A pointer to your current branch/commit.
Stash	A temporary storage for changes you don't want to commit yet.
Merge	Combine changes from one branch into another.
Rebase	Move or replay commits on top of another branch for a cleaner history.
Tag	A label (usually for versions, like `v1.0`).

🔷 Git Commands with Explanation and Examples

Configuration

```
git config --global user.name "Your Name"
git config --global user.email "you@example.com"

Sets your identity for commits (use the same email linked to GitHub).
```

git config --list
Shows all settings.

Start a Repository

```
git init
```

Starts a new Git repository in the current folder.

```
git clone git@github.com:username/repo.git
```

Copies (clones) a remote repository from GitHub to your computer.

Basic Workflow

`git status`

Shows which files are changed, staged, or untracked.

`git add filename.txt`

Adds `filename.txt` to the staging area.

`git add .`

Adds all changes to staging.

`git commit -m "Add new feature"`

Saves staged changes into the repo history with a message.

`git log`

Shows commit history.

Branches

`git branch`

Lists all branches.

`git branch feature-branch`

Creates a new branch.

`git checkout feature-branch`

Switches to the branch.

`git checkout -b feature-branch`

Creates and switches in one step.

`git merge feature-branch`

Merges changes from `feature-branch` into the current branch.

Remote Repositories

`git remote add origin git@github.com:username/repo.git`

Links local repo to a GitHub repo.

```
git remote -v
```

Shows linked remotes.

```
git push -u origin main
```

Pushes commits to GitHub and sets tracking.

```
git push
```

Pushes commits to remote.

```
git pull origin main
```

Fetches and merges changes from GitHub.

```
git fetch
```

Downloads changes but does **not** merge them.

Undoing Changes

```
git checkout -- filename.txt
```

Discards changes in a file (restores last committed version).

```
git reset HEAD filename.txt
```

Unstages a file (keeps changes).

```
git reset --hard
```

Removes all changes (back to last commit). ⚠️ Dangerous!

```
git revert <commit_id>
```

Makes a new commit that undoes a specific commit.

Stashing

```
git stash
```

Saves changes temporarily without committing.

```
git stash list
```

Shows saved stashes.

```
git stash apply
```

Restores latest stash (keeps it saved).

`git stash pop`

Restores and deletes stash.

Tagging

`git tag v1.0`

Creates a tag (useful for marking releases).

`git push origin --tags`

Pushes tags to GitHub.

Viewing

`git log --oneline`

Shows a simplified commit history.

`git show <commit_id>`

Shows details of a commit.

`git log -- filename.txt`

Shows history of a specific file.

`git diff`

Shows unstaged changes.

`git diff --staged`

Shows staged changes.

Advanced

`git rebase main`

Moves your branch commits on top of `main`.

`git cherry-pick <commit_id>`

Applies a commit from another branch to your current one.

```
git blame filename.txt
```

Shows who changed each line in a file.

Git Workflow

1. `git init` or `git clone` → start repo
2. `git status` → check changes
3. `git add` → stage files
4. `git commit -m "message"` → commit
5. `git push` → send to GitHub
6. `git pull` → get updates