# Essential Git Commands for DevOps Engineers

## 🔧 Setup & Configuration

git config --global user.name "Your Name"

git config --global user.email "you@example.com"

git config --list 🡺Verify configuration

## 📥 Repository Management

git init 🡺Initialize a new Git repo

git init <repo name> 🡺 create an initialized empty git repo

git clone <repo-url> 🡺Clone an existing repo

git remote -v 🡺View configured remote repos

git remote add <link of remote empty repository>

## 📄 Working with Files

git status 🡺Show modified, staged, and untracked files

git add <file> 🡺Stage a specific file

git add . 🡺Stage all changes

git rm <file> 🡺Remove file from staging and disk

git rm -rf <file/folder name> 🡺forcefully remove

git mv <old> <new> 🡺Rename or move file

## 💾 Committing Changes

git commit -m "Your message" 🡺Commit staged changes

git commit -am "msg" 🡺Add & commit tracked files in one step

## 🕗 Viewing History

git log 🡺View commit history

git log --oneline --graph --all 🡺Compact, visual commit tree

git diff 🡺View unstaged changes

git diff --staged 🡺View staged changes

## 🔁 Branching & Merging [a merge after rebase will become ff merge kind of rebase so use --no-ff]

git branch 🡺List branches

git branch -r 🡺 list remote branches

git branch -a 🡺 list all local and remote

git branch <name> 🡺Create a new branch

git branch -m <new name> 🡺 rename the branch

git branch -d <branch name> 🡺 delete the branch

git checkout <branch> 🡺Switch branches

git checkout -b <name> 🡺Create & switch to new branch

git merge <branch> 🡺Merge branch into current

git merge --ff-only <branch> 🡺 fast forward merge which is basically a rebase

git merge --no-ff <branch> 🡺 forced merge which is basic without ff

git rebase <branch> 🡺Reapply commits on top of another base

git merge/rebase --abort 🡺 unmerge or unbase

## ☁️ Remote Collaboration

git pull 🡺Fetch + merge changes from remote

git fetch 🡺Only download changes

git push 🡺Upload changes to remote

git push -u origin <branch> 🡺Push new branch to remote & track

### ✅ ****1. Undo Working Directory Changes****

| **Task** | **Command** |
| --- | --- |
| Discard changes in a file | git restore <file> |
| Discard all changes | git restore . |
| Discard and unstage | git restore --staged <file> + git restore <file> |
| Discard new (untracked) files | git clean -f |
| Discard new untracked directories | git clean -fd |
| Preview what clean will delete | git clean -n |

### ✅ ****2. Undo Staging (Unstage Files)****

| **Task** | **Command** |
| --- | --- |
| Unstage a specific file | git reset <file> |
| Unstage all files | git reset |
| Unstage and keep changes | git restore --staged <file> |

### ✅ ****3. Undo Commits****

| **Task** | **Command** |
| --- | --- |
| Undo the last commit but keep changes | git reset --soft HEAD~1 |
| Undo last commit and unstage changes | git reset --mixed HEAD~1 |
| Undo last commit and discard changes | git reset --hard HEAD~1 |
| Undo multiple commits | git reset --hard HEAD~N |
| Revert a commit (safe, creates new commit) | git revert <commit\_hash> |

### ✅ ****4. Clean Untracked Files****

| **Task** | **Command** |
| --- | --- |
| List what will be deleted | git clean -n |
| Delete untracked files | git clean -f |
| Delete untracked files and dirs | git clean -fd |
| Delete ignored files too | git clean -fx |
| Clean everything: untracked, dirs, ignored | git clean -fdx |

### ✅ ****5. Stash Changes (Temporary Undo)****

| **Task** | **Command** |
| --- | --- |
| Save uncommitted changes | git stash |
| View stashes | git stash list |
| Apply last stash | git stash apply |
| Apply and delete stash | git stash pop |
| Drop a stash | git stash drop stash@{0} |
| Clear all stashes | git stash clear |

### 🚨 ****Warnings****

* git reset --hard and git clean **permanently delete changes**. Use with caution.
* Use git status a

## 🛠️ Stashing & Cherry-picking

git stash 🡺Save work temporarily

git stash pop 🡺Re-apply stashed changes

git cherry-pick <commit> 🡺Apply a specific commit

## 🔍 Debugging & Auditing

git blame <file> 🡺Show who changed what

git show <commit> 🡺Show changes of a commit

git bisect 🡺Find commit that introduced a bug