🧼 Everyday Hygiene

Task	Command
Check current status	`git status`
See commit history	`git log --oneline --graph --decorate`
Add files	`git add <file>` or `git add .`
Commit	`git commit -m "Message"`
Undo last commit (keep changes)	`git reset --soft HEAD~1`
Undo last commit (unstage files)	`git reset --mixed HEAD~1`
Remove commit and file changes	`git reset --hard HEAD~1`

🌿 Branching & Fixing Mistakes

Situation	What to Do
Create new branch	`git checkout -b feature/xyz`
Switch branch	`git switch <branch>` or `git checkout <branch>`
Rename branch	`git branch -m new-name`
You forgot to switch branch before commits	

✅ `git branch new-branch`

✅ `git reset --hard HEAD~n`

✅ `git switch new-branch`

Move 1 commit to another branch

✅ `git checkout -b temp-branch`

✅ `git cherry-pick <commit>`

✅ `git switch original`

✅ `git reset --hard HEAD~1`

🧽 Stashing & Cleanup

Task	Command
Stash current changes	`git stash`
List stashes	`git stash list`
Show stash contents	`git stash show -p stash@{0}`
Apply latest stash	`git stash apply`
Clean untracked files	`git clean -fxn` (preview) / `-fx` (execute)

📌 Rebase & Squash Commits

Use Case	Command
Rebase onto another branch	`git rebase <branch>`
Interactive squash	`git rebase -i HEAD~N`
Edit commit message	Replace `pick` with `reword` in rebase editor
Fix conflict during rebase	Resolve files ➤ `git add .` ➤ `git rebase --continue`
Abort rebase	`git rebase --abort`

🔀 Syncing & Collaboration

Task	Command
Fetch all updates	`git fetch --all`

See local vs remote difference	`git log HEAD..origin/<branch>`
Set upstream tracking	`git branch --set-upstream-to=origin/<remote-branch> <local>`
Track remote with different name	`git checkout -b local-name origin/remote-name`
Push current branch	`git push -u origin <branch>`
Force push (use with care!)	`git push --force`
Recover from force push	`git reset --hard <good-commit>` ➤ `git push --force`

📜 Tags & CI/CD

Task	Command
Create tag	`git tag -a v1.0 -m "Release v1.0"`
Push tag	`git push origin v1.0`
See tags	`git tag`
Latest tag info	`git describe` or `git describe --tags --abbrev=0`

🕵️ Debugging Code History

Task	Command
See who added a line	`git blame <file>`
See diff between 2 commits	`git diff <commit1> <commit2>`
See file in a past commit	`git show <commit>:<file-path>`

🧠 Lifesaver Commands for Panic Moments

Situation	Use This
Wrong file committed	`git reset HEAD~1` ➤ `git add` right files ➤ `git commit`
Forgot what's stashed	`git stash list` ➤ `git stash show -p stash@{0}`
Accidentally deleted branch	`git reflog` ➤ Find last HEAD ➤ `git checkout -b recovery <hash>`
Merge conflict?	Resolve ➤ `git add` ➤ `git commit` or `git rebase --continue`
Restore file from another branch	`git checkout <branch> -- <file>`

🎁 Bonus: Shortcuts to Look Cool 😎

Alias	Command
git lg	`git log --oneline --graph --decorate --all`
git undo	`git reset --soft HEAD~1`
git last	`git log -1 HEAD`
git gone	`git remote prune origin`