

Git

1. Configuration

```
git config --global user.email "mashfiq.rayhan.ovi@gmail.com" # Set global email
```

```
git config --global user.name "mashfiq-rayhan" # Set global username
```

2. Initialize Repository

```
git init # Initialize a new Git repository
```

3. Basic Commands

```
git status # Show the working tree status
```

```
git add . # Stage all changes for commit
```

```
git commit -m "commit message" # Commit staged changes with a message
```

```
git log # Show commit history
```

4. Branching & Switching

```
git branch # List all branches
```

```
git branch <branchname> # Create a new branch
```

```
git checkout <branchname> # Switch to an existing branch
```

```
git checkout -b <branchname> # Create and switch to a new branch
```

```
git switch <branchname> # Alternative to checkout (recommended)
```

```
git switch -c <branchname> # Create and switch to a new branch
```

5. Merging & Rebasing

```
git merge <branchname> # Merge specified branch into current branch
```

```
git merge --squash <branchname> # Merge changes but keep commits unsynced
```

```
git commit -m "message" # Commit the merged changes
```

```
git merge --no-ff <branchname> # Merge with a non-fast-forward option
```

```
git rebase <branchname> # Reapply commits on top of another branch
```

```
git merge --abort # Abort an in-progress merge
```

6. Undo Changes

```
git checkout <commit-id> # Switch to a specific commit (detached HEAD)

git checkout <filename> # Restore file from last commit

git checkout . # Restore all files from last commit

git restore path/<filename> # Restore specific file from last commit

git reset # Unstage all changes

git restore --staged <filename> # Unstage a specific file

git reset --soft HEAD~1 # Undo last commit but keep changes staged

git reset HEAD~1 # Undo last commit and unstage changes

git reset --hard HEAD~1 # Undo last commit and remove changes from working directory
```

7. Deleting Branches

```
git branch -d <branchname> # Delete merged branch

git branch -D <branchname> # Force delete unmerged branch
```

8. Stashing Changes

```
git stash # Save uncommitted changes

git stash list # List all stashes

git stash apply [index] # Apply a stash (default is latest)

git stash push -m "message" # Stash with a message

git stash pop [index] # Apply and remove stash

git stash drop [index] # Delete a specific stash

git stash clear # Remove all stashes
```

9. Cleaning Untracked Files

```
git clean -dn # Show a list of untracked files to be deleted

git clean -df # Delete untracked files permanently
```

10. Cherry-picking Commits

```
git cherry-pick <commit-id> # Apply a specific commit from another branch
```

11. Viewing Changes

```
git diff # Show unstaged changes

git log --merge # Show commits affecting the merge
```

12. Reflog & History

```
git reflog # Show reference logs (past commit history for 30 days)
```

13. Tags

Lightweight Tags

```
git tag <tag-id> <commit-id> # Create a lightweight tag
```

```
git show <tag-id> # Show details of a tag
```

```
git tag -d <tag-id> # Delete a tag
```

Annotated Tags

```
git tag -a <tag-id> -m "message" # Create an annotated tag
```

```
git tag -d <tag-id> # Delete an annotated tag
```

14. Local to Remote

```
git remote # List remote servers

git branch -a # Show all local and remote branches

git branch -r # Show remote tracking branches

git remote show origin # Show details of remote origin

git remote add origin <repo-link> # Add a remote repository

git branch -vv # Show local-remote tracking branches

git push origin master # Push changes to the remote master branch

git branch --track [localTrackingBranch] origin/[remoteTrackingBranch] # Track a remote branch

git clone <url> # Clone a repository

git ls-remote # List references in a remote repository

git branch --delete --remotes [remoteTrackingBranch] # Delete remote tracking branch

git push origin --delete [remoteBranch] # Delete a remote branch
```

15. Sync Forked Repository

```
git remote -v # Verify the remote URL of your fork

git remote add upstream <original-repo-url> # Add the original repository as upstream

git fetch upstream # Fetch updates from the original repository

git checkout main # Switch to the main branch

git merge upstream/main # Merge upstream changes into your local main branch

git push origin main # Push the synced changes to your forked repository
```