

Name - Vaibhav Dixit

1. Set your username and email in git config

For setting username, we can use this git command -

```
git config --global user.name "Your Username"
```

For setting email -

```
git config --global user.email "your.email@example.com"
```

2. Create a new branch named "feature-branch" and switch to it.

```
git checkout -b feature-branch
```

3. List all branches in the repository.

```
git branch
```

4. Delete the branch "feature-branch"

```
git branch -d feature-branch
```

5. How do you undo the last commit

```
git reset HEAD~1
```

6. Create a new branch names "conflict-branch"

```
git checkout -b conflict-branch
```

7. Create a another branch named "feature1"

```
git checkout -b feature1
```

8. Make some changes in to feature1 branch

Assuming that we are at branch feature1 as we have checked out to it while creating it in the previous step. I would make some file changes and then

Git add .

Lets assume that we are making change to file1 at line 2 only -

Git add file1

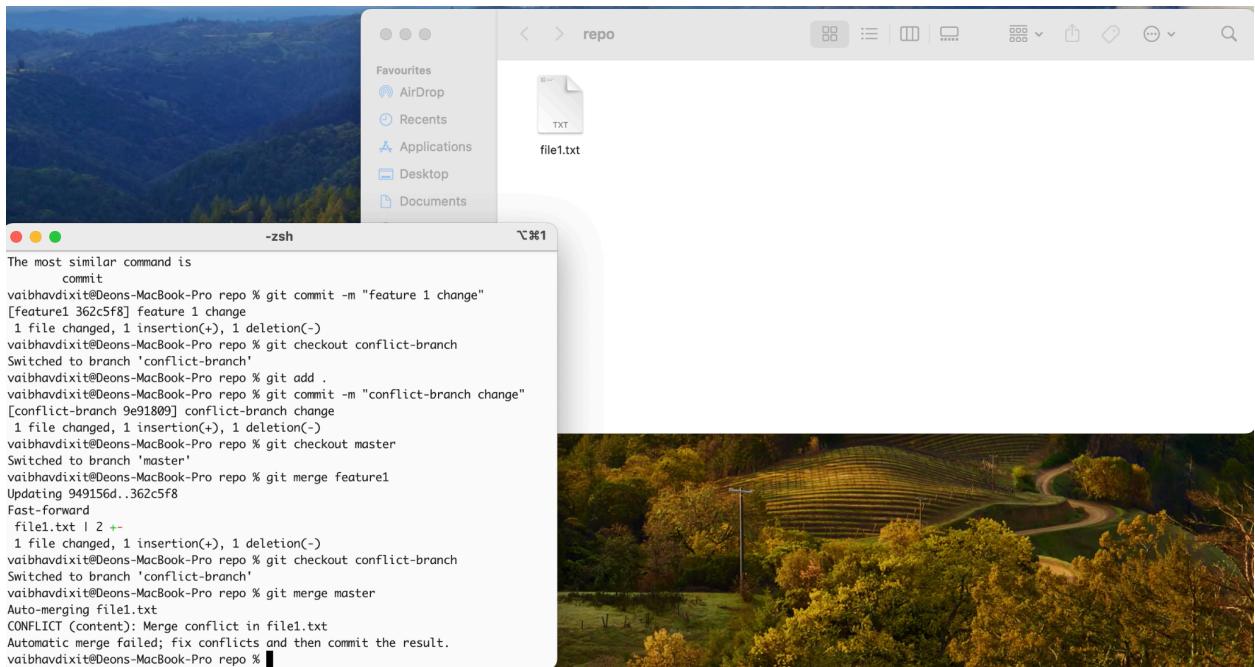
9. Merge “feature1” branch into main branch

```
git checkout main  
git merge feature1
```

10. Make changes in “conflict-branch”, in the same file and line that you had made changes in feature1

```
git checkout conflict-branch  
# Making changes in the same file (file1) and line (line 2) as in feature1  
git add file1
```

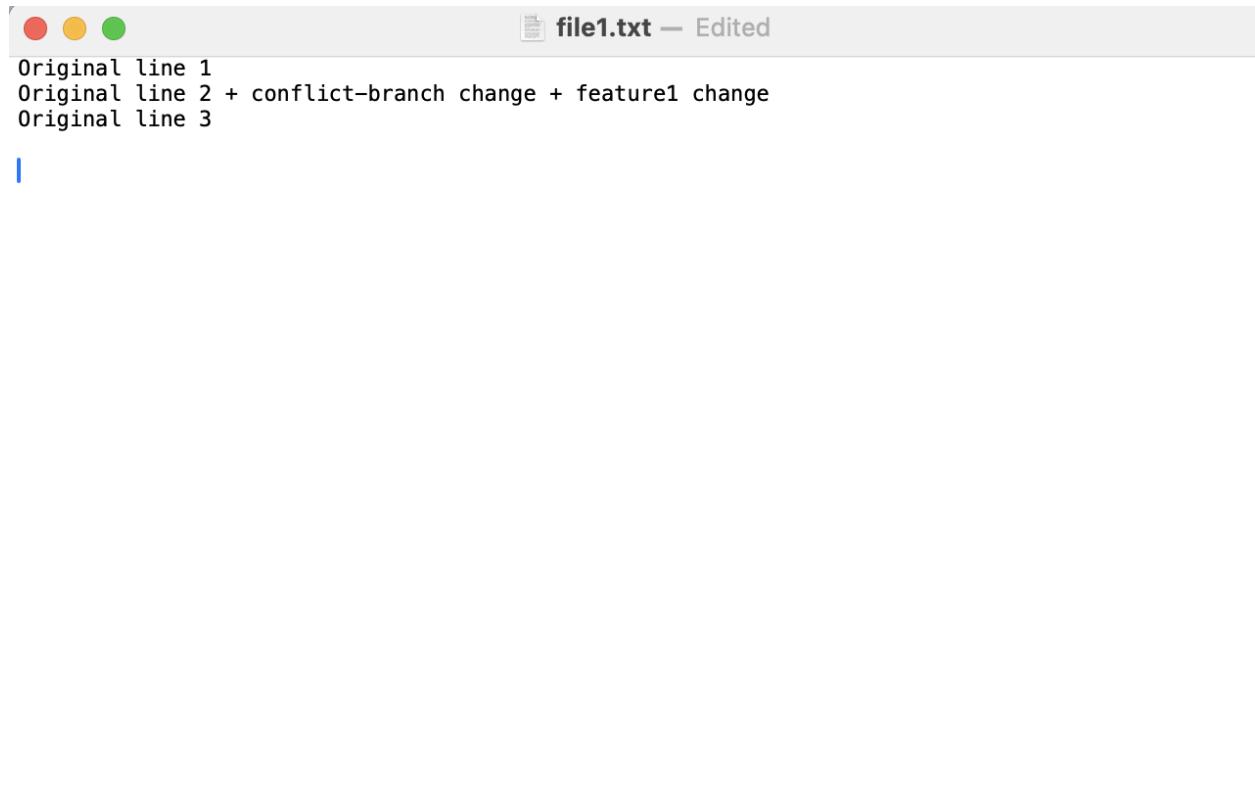
11. Merge master into conflict-branch [Attach screenshot of terminal & file]



A screenshot of a code editor window titled "file1.txt". The content of the file is as follows:

```
Original line 1
<<<<< HEAD
Original line 2 + conflict-branch change
=====
Original line 2 + feature1 change
>>>>> master
Original line 3
```

12. Resolve merge conflicts



file1.txt — Edited

```
Original line 1
Original line 2 + conflict-branch change + feature1 change
Original line 3
```

```
git add .
git commit -m "Resolved merge conflict"
```

13. Add a remote named "origin" pointing to a GitHub repository.

```
Git remote add origin https://github.com/flow6979/git-practice
```

```
it-practice
Cloning into 'git-practice'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (3/3), done.
vaibhavdixit@Deons-MacBook-Pro desktop % git init
hint: Using 'master' as the name for the initial branch. This default branch nam
e
hint: is subject to change. To configure the initial branch name to use in all
hint: of your new repositories, which will suppress this warning, call:
hint:
hint:   git config --global init.defaultBranch <name>
hint:
hint: Names commonly chosen instead of 'master' are 'main', 'trunk' and
hint: 'development'. The just-created branch can be renamed via this command:
hint:
hint:   git branch -m <name>
Initialized empty Git repository in /Users/vaibhavdixit/Desktop/.git/
vaibhavdixit@Deons-MacBook-Pro desktop % git remote add origin https://github.co
m/flow6979/git-practice
vaibhavdixit@Deons-MacBook-Pro desktop % git remote -v
origin  https://github.com/flow6979/git-practice (fetch)
origin  https://github.com/flow6979/git-practice (push)
vaibhavdixit@Deons-MacBook-Pro desktop %
```

git remote -v (for verifying)

14. Fork a repository on GitHub and clone it to your local machine.

distributed-e-commerce Private

forked from [VandanaKumari18/distributed-e-commerce](#)

main 1 Branch 0 Tags

This branch is up to date with [VandanaKumari18/distributed-e-commerce:main](#).

flow6979 Merge pull request [VandanaKumari18#1](#) from flow6979/Vandana/readme

README.md modified : Readme.md file

Git clone <https://github.com/flow6979/distributed-e-commerce.git>

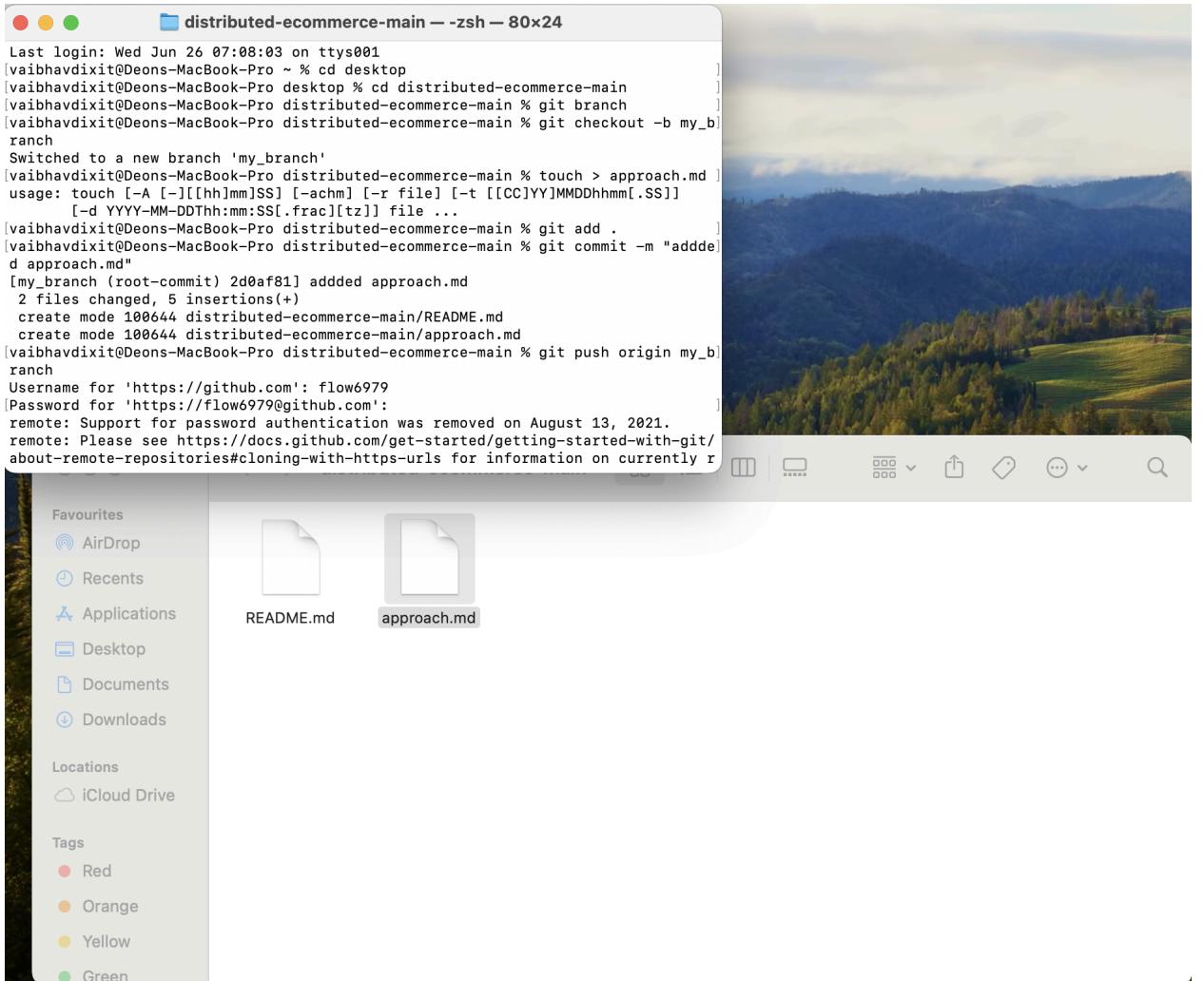
15. Create a new branch on your fork, make changes, and open a pull request to the original repository.

```
git checkout -b my_branch
```

Added approach.md

```
git add .  
git commit -m "Added approach.md"
```

```
git push origin my_branch
```



There is some authentication issue while cloning and pushing

16. Comment on a PR and suggest improvements

B

The screenshot shows a GitHub commit page for a file named README.md. The commit message is '# distributed-e-commerce' and the commit hash is e55e9f3. A comment from user flow6979 is visible, suggesting to 'use br less often'. The interface includes standard GitHub UI elements like file navigation, commit details, and a comment section.

17. Create a Git alias for the command `git log --oneline` named `gitlol`.

```
git config --global alias.gitlol "log --oneline"
```

Now, git lol & git log --oneline should do same thing

18. Create a pre-commit hook

Steps -

1. cd .git/hooks (inside our repo)
2. touch pre-commit (creating new file)
3. chmod +x pre-commit (making it executable)
4. nano pre-commit (editing our file using text editor)
5. Add pre-commit hook script, then save and exit the editor

Pre-commit Hook: This script runs before each commit is processed. It allows you to enforce policies or checks (like code linting, tests, etc.) before allowing a commit to proceed.

19. You have made local changes in your branch, but you need to switch to another branch urgently without committing. How would you handle this situation?

Stash

```
git stash  
git checkout branch2 (going from branch1 to branch2)  
git stash apply
```

We can also create other worktree for other branch

20. You accidentally deleted a file in your local repository. How do you restore it using Git?

`git checkout HEAD -- deleted_file_path`

retrieves the specified file from the most recent commit (HEAD) in your Git history and restores it to your working directory.

21. You have committed changes to your branch but forgot to include a file. How do you add the file to the last commit without creating a new commit?

`Git add file_path`

`Git commit --amend --no-edit`

`--amend`: Incorporates changes into the last commit instead of creating a new one.

`--no-edit`: Prevents Git from opening the commit message editor, preserving the existing commit message.

22. You want to discard all changes in your working directory and revert to the last commit.

What Git command would you use?

`Git reset`

or

`git reset --hard HEAD`

(`--hard` -> reset both staging area and working directory)

23. You need to view a specific commit's changes. What Git command can be used to show the changes introduced by a particular commit?

`Git show commit-hash`

`commit -hash` -> unique alphanumeric string assigned to each commit in git.

24. You want to change a commit message, after you have already committed, how do you do so?

`git commit --amend -m "New commit message"`

Or

```
git commit --amend
```

Edit in editor -> save and close the editor

25. Your colleague has made changes in their branch, and you want to incorporate those changes into your branch without merging. How do you achieve this?

Steps -

1. Identify commit hashes from colleague's branch
git checkout colleague-branch
git log
2. git checkout your-branch
3. git cherry-pick <commit-hash>
Or
git cherry-pick <commit-hash1> <commit-hash2> .. (for specific hashes)
Or
git cherry-pick <commit-hash1>..<commit-hash2> (for range)

26. You've made several commits on a branch, but you want to club them into a single commit before pushing to the remote repository. How would you do that?

Commit squashing -

1. Start an interactive rebase
git rebase -i HEAD~n (n = number of commits to merge)
2. An editor will open with our commits prefixed with pick, except for first replace all picks with squash. Then save and close editor. Another editor will open to edit combined commit message.

A force push may require to rewrite the commit history (git push origin feature-branch --force)

27. You accidentally staged a file that you don't want to commit. How do you unstage it?

Git reset

Or

Git reset HEAD file_name

28. You don't want to commit files that have .yml in the end, and also files inside folder config. How do you do that?

Use gitignore

touch .gitignore

Add rules in .gitignore, like -

*.yml

config/

29. You want to see a list of all the files changed in the last commit. What Git command would you use?

git show --name-only HEAD

Or

git diff --name-only HEAD~1 HEAD

30. You realize that your local branch is outdated, and you want to fetch the latest changes from the remote repository. How do you do this without merging?

git fetch

31. You accidentally deleted a branch. How do you recover it?

1. git reflog (to find commit hash of last commit on deleted branch)
2. git checkout -b new_branch <commit-hash>

32. You want to remove untracked files and directories from your working directory. What Git command would you use?

1. Dry run: git clean -n
2. Removing untracked files: git clean -f
3. Removing untracked directories: git clean -fd
4. Removing ignored files: git clean -fdx

33. You have a commit from a feature branch that you want to apply to the main branch without merging the entire feature branch.

1. git log --oneline feature-branch
2. git checkout main

3. git cherry-pick <commit-hash>
 4. Resolve conflicts
 - git add resolved_file
 - git cherry-pick --continue
34. You mistakenly committed a change to the wrong branch and need to apply that commit to the correct branch.
 1. git log --oneline (identify commit hash)
 2. git checkout correct_branch
 3. git cherry-pick <commit-hash>
 4. Resolve conflicts
 - git add resolved_file
 - git cherry-pick --continue
35. There is a series of commits on a feature branch, but you only want to cherry-pick a specific range of commits.
 1. Identify commit hashes
 - git log --oneline feature_branch
 2. git checkout our_branch
 3. git cherry-pick <start-commit>^..<end-commit>
 4. Resolve conflicts
 - git add resolved_file
 - git cherry-pick --continue
36. You want to clone a GitHub repository onto your local machine, but you only need a specific branch. How can you achieve this?

`git clone -b branch_name repository_url`
37. You've made changes to your local repository and want to push them to your fork on GitHub. What Git commands would you use?

`git add .`
`git commit -m "commit message"`
`git push origin branch_name`
38. You want to create a new branch both locally and on GitHub to work on a new feature. What commands would you use?

Creating locally: git checkout -b branch_name
Pushing to github: git push -u origin branch_name

39. You want to see the commit history of a GitHub repository. How can you do this using Git commands?

git log

40. You've accidentally committed sensitive information and want to remove the commit from both your local and remote repositories on GitHub. What commands would you use?

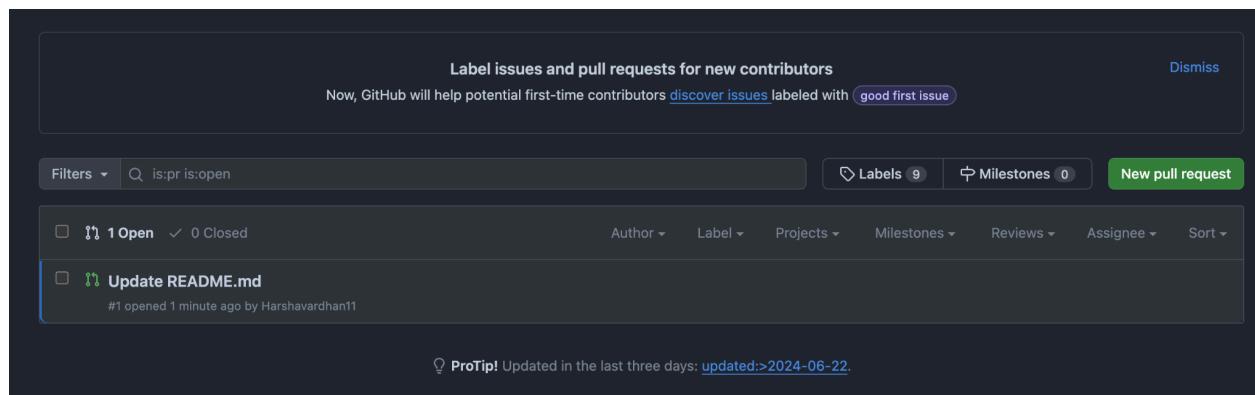
1. git reset --hard HEAD~1 (resetting branch head to point before commit, maybe just before current head)
2. git push --force origin branch_name (force push the updated branch to github repo to overwrite remote history)

41. You want to delete a remote branch on GitHub. What Git command would you use?

1. git branch -d branch-name (deleting locally)
2. git push origin --delete branch-name (deleting on github)

42. Create a git repository for all your assignments and upload them in it. Ask your peers to code review it, and you need to code review your peers assignments

Done



Got pr for review -

A screenshot of a GitHub pull request interface. At the top, a green button says "Open" and a status message says "Harshavardhan11 wants to merge 1 commit into flow6979:main from Harshavardhan11:harshwardhan". Below this are tabs for "Conversation 0", "Commits 1", "Checks 0", and "Files changed 1". The main area shows a comment from "Harshavardhan11" saying "pulling the request" with a smiley face emoji. Below it is a commit "Update README.md" with a small profile picture. A callout box on the right says "First-time contributor" and "Looks like you got your first contribution! Visit your community profile to learn more about recommended open source practices." with a "Go to community profile" button. Another comment from "flow6979" says "okay worthy change" with a smiley face emoji. On the far right, there's an "Owner" section.

Merged a PR -

A screenshot of a GitHub pull request interface showing a merged pull request. At the top, a purple button says "Merged" and a status message says "flow6979 merged 1 commit into flow6979:main from Harshavardhan11:harshwardhan now". Below this are tabs for "Conversation 0", "Commits 1", "Checks 0", and "Files changed 1". The main area shows a comment from "Harshavardhan11" saying "pulling the request" with a smiley face emoji. Below it is a commit "Update README.md" with a small profile picture and a "Verified" badge with the hash "f2d3ab6". A callout box on the right says "(Contributor)" and "Contributor". Another comment from "flow6979" says "okay worthy change" with a smiley face emoji. At the bottom, a message says "flow6979 merged commit abda0dd into flow6979:main now" with a "Revert" button. On the far right, there's an "Owner" section.

Raised an issue

43. Create a pull request on any open source library on github, attach the pull request link to the readme file of this project's repository