SRE Training (Day 5) - GIT

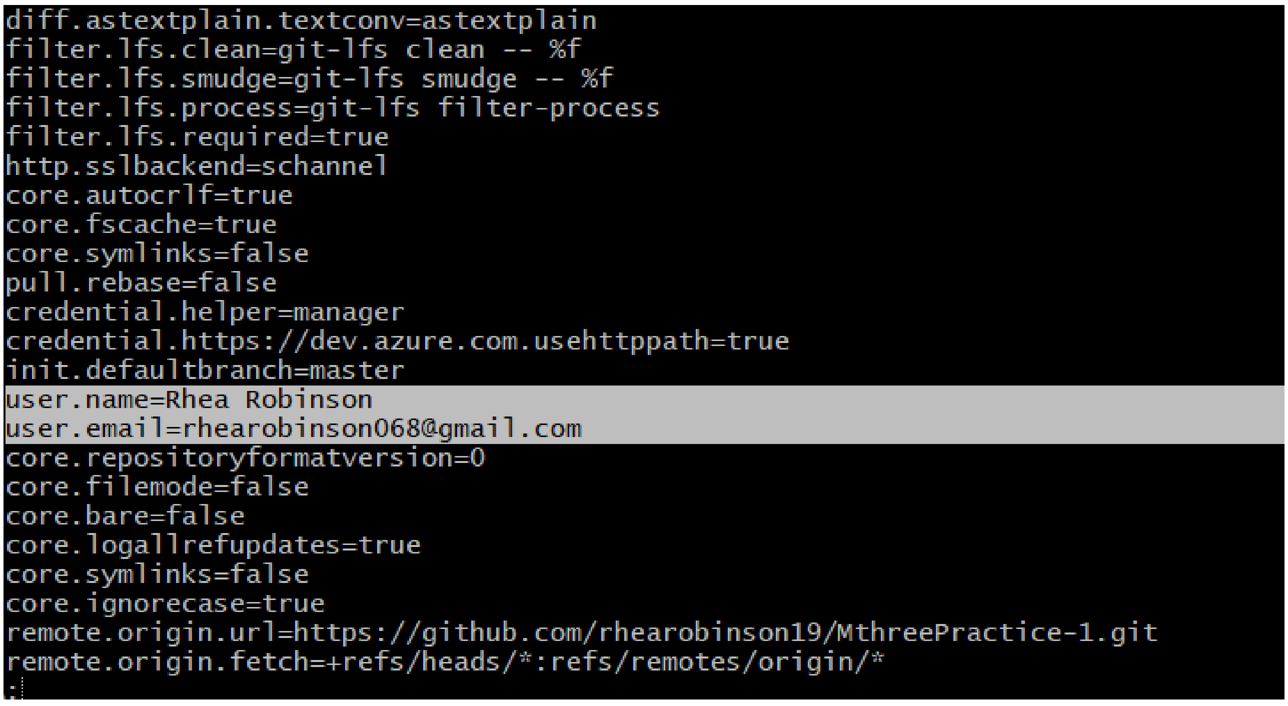
Version Control System (VCS) - tool that helps track changes to files over time, allowing multiple people to collaborate efficiently. It maintains a history of modifications, making it easy to revert to previous versions if needed.

GIT - Distributed Version Control Tool
GitHub - Cloud platform that uses GIT

I. GIT CONFIG

- **git config user.name "Rhea Robinson"**
- **git config user.email "rhearobinson00@gmail.com"**
  Local declaration (applies to a particular local git folder)

- **git config --global user.name "Rhea Robinson"**
  Global declaration (applies to all repos)
- **git config –list** - list configuration details

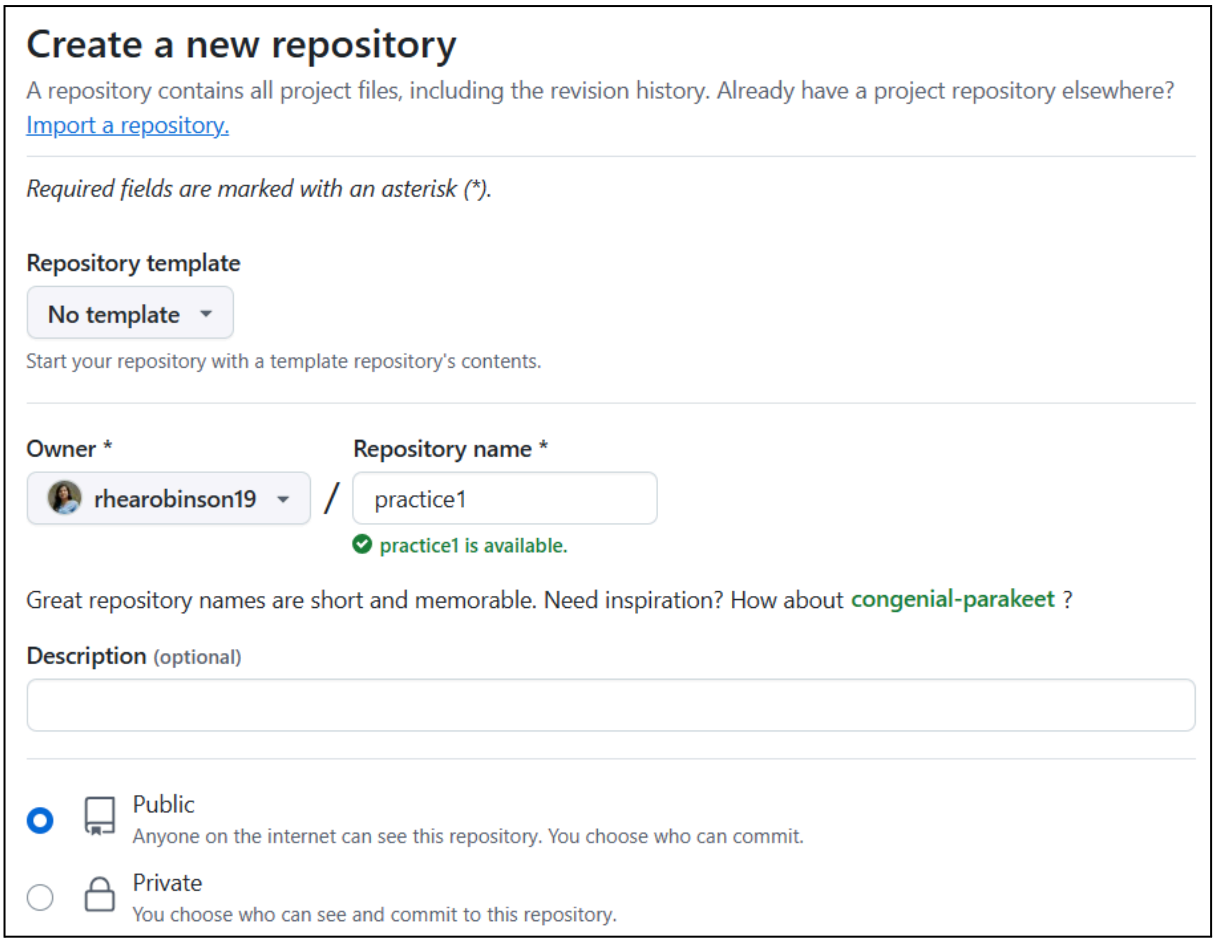


```
diff.astextplain.textconv=astextplain
filter.lfs.clean=git-lfs clean -- %f
filter.lfs.smudge=git-lfs smudge -- %f
filter.lfs.process=git-lfs filter-process
filter.lfs.required=true
http.sslbackend=schannel
core.autocrlf=true
core.fscache=true
core.symlinks=false
pull.rebase=false
credential.helper=manager
credential.https://dev.azure.com.usehttppath=true
init.defaultbranch=master
user.name=Rhea Robinson
user.email=rhearobinson068@gmail.com
core.repositoryformatversion=0
core.filemode=false
core.bare=false
core.logallrefupdates=true
core.symlinks=false
core.ignorecase=true
remote.origin.url=https://github.com/rhearobinson19/MthreePractice-1.git
remote.origin.fetch=+refs/heads/*:refs/remotes/origin/*
:
```

II. GETTING STARTED WITH YOUR FIRST REPOSITORY

1. CREATING GITHUB REPOSITORY

The github repository will be available remotely on the cloud platform. The steps are -> Create a new repo >> give it a name >> Optional README file (contains details about the repository)



Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? Import a repository.

Required fields are marked with an asterisk (*).

Repository template

No template

Start your repository with a template repository's contents.

Owner *: rhearobinson19 / Repository name *: practice1

practice1 is available.

Great repository names are short and memorable. Need inspiration? How about congenial-parakeet ?

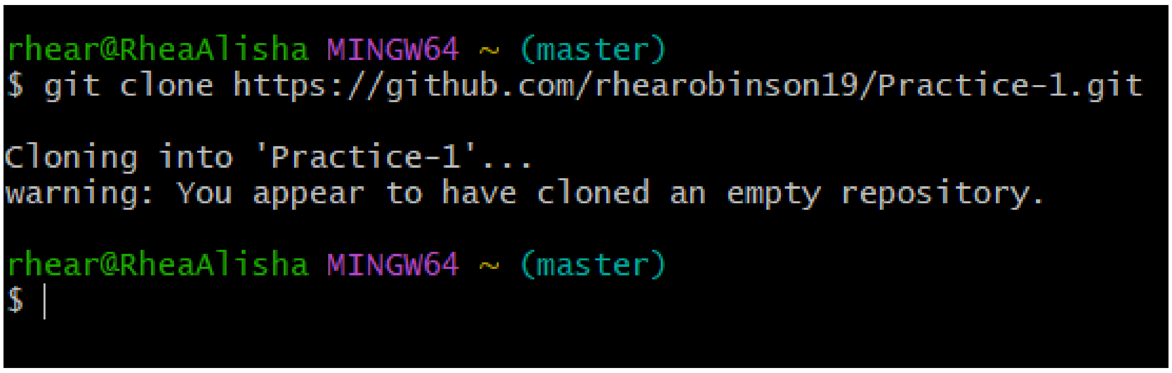
Description (optional)

[✓ Public] Anyone on the internet can see this repository. You choose who can commit.

Private — You choose who can see and commit to this repository.

2. CLONING TO THE LOCAL COMPUTER

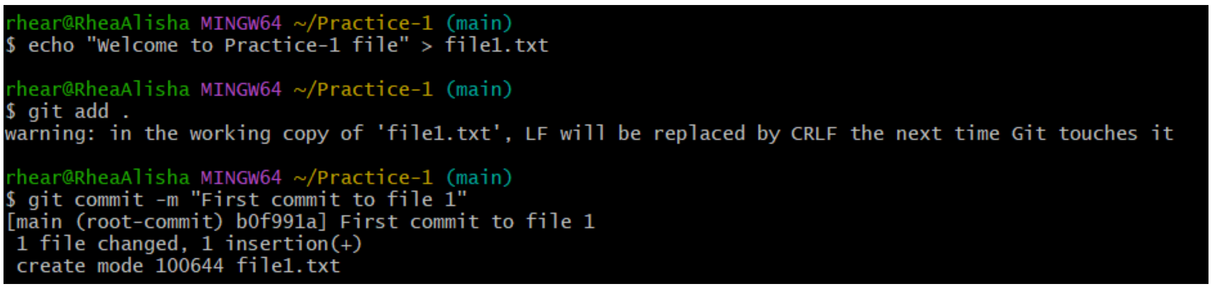
There are multiple cloning options (I have used https). Steps : Copy the https link to close -> use command - **git clone** https://github.com/rhearobinson19/Practice-1.git



```
rhear@RheaAlisha MINGW64 ~ (master)
$ git clone https://github.com/rhearobinson19/Practice-1.git

Cloning into 'Practice-1'...
warning: You appear to have cloned an empty repository.

rhear@RheaAlisha MINGW64 ~ (master)
$ |
```

3. CREATE A NEW FILE ON LOCAL REPO, ADD, COMMIT AND PUSH TO REMOTE

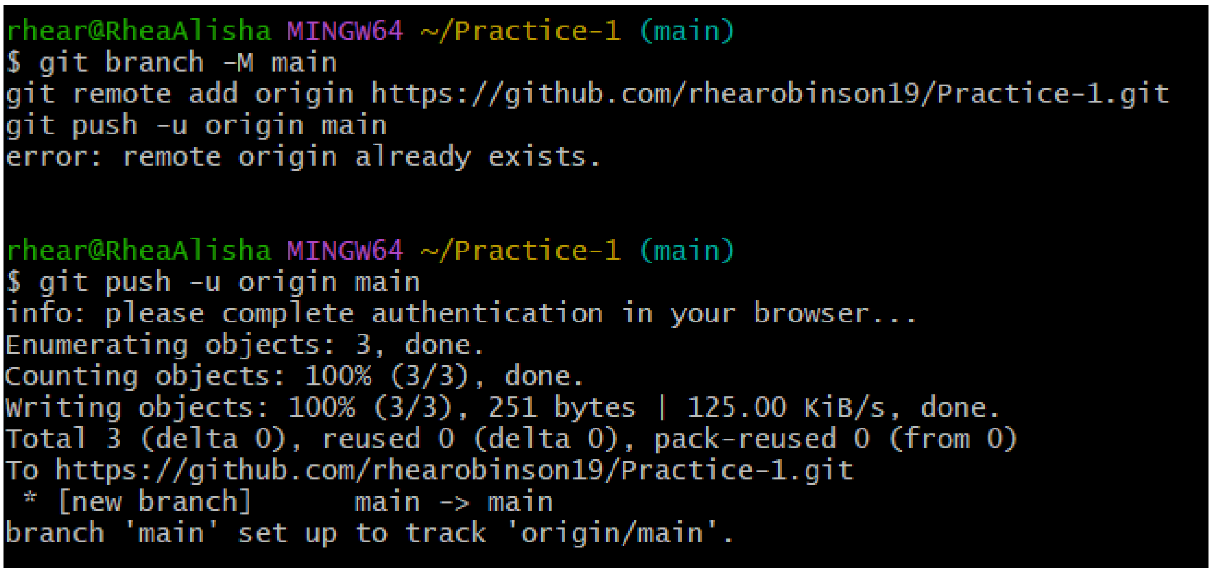
- To create a new file, we can use touch command or echo with redirection among other ways.
- Any changes made to a new file must be added using the git **add** command, which pushes it to the staging area, and then can be committed using **git commit** with a commit message.



```
rhear@RheaAlisha MINGW64 ~/Practice-1 (main)
$ echo "Welcome to Practice-1 file" > file1.txt

rhear@RheaAlisha MINGW64 ~/Practice-1 (main)
$ git add .
warning: in the working copy of 'file1.txt', LF will be replaced by CRLF the next time Git touches it

rhear@RheaAlisha MINGW64 ~/Practice-1 (main)
$ git commit -m "First commit to file 1"
[main (root-commit) b0f991a] First commit to file 1
 1 file changed, 1 insertion(+)
 create mode 100644 file1.txt
```

4. BRANCHES & GIT PUSH

- The command **git branch -M main** shown below is to override the default master branch, rename and use as main.
- The committed changes need to be pushed to the remote using **git push -u origin main** command
- If remote origin was not added, it can be added using git **remote add origin "repo_url"** command where you replace repo_url with the url of your github repository.

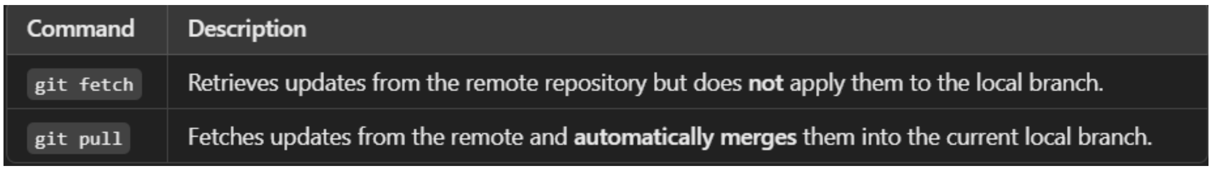


```
rhear@RheaAlisha MINGW64 ~/Practice-1 (main)
$ git branch -M main
git remote add origin https://github.com/rhearobinson19/Practice-1.git
git push -u origin main
error: remote origin already exists.


rhear@RheaAlisha MINGW64 ~/Practice-1 (main)
$ git push -u origin main
info: please complete authentication in your browser...
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Writing objects: 100% (3/3), 251 bytes | 125.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/rhearobinson19/Practice-1.git
 * [new branch]      main -> main
branch 'main' set up to track 'origin/main'.
```

5. GIT PULL AND FETCH

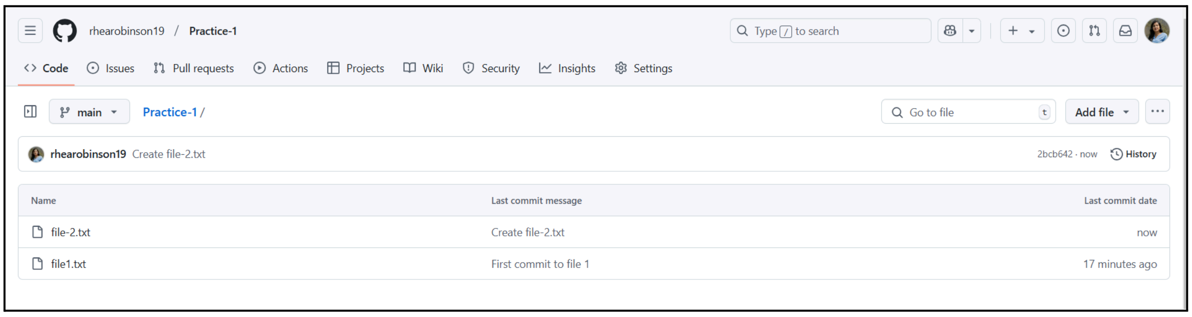
- Just like how **git push** reflects the local changes on the remote repository, similarly, **git pull** and **git fetch** retrieves the changes made in the remote repo onto your local branch.

- Difference between **git pull** and **git fetch**:



Command	Description
`git fetch`	Retrieves updates from the remote repository but does **not** apply them to the local branch.
`git pull`	Fetches updates from the remote and **automatically merges** them into the current local branch.

Below image shows a new file "file-2.txt" created on the remote repository

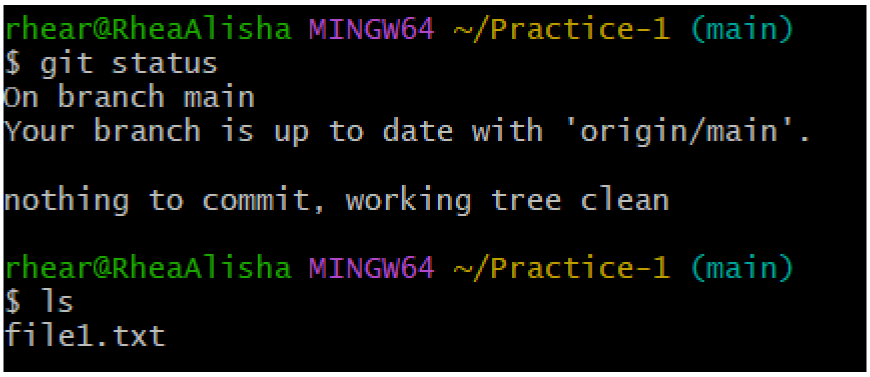


rhearobinson19 / Practice-1

rhearobinson19 Create file-2.txt 2bcb642 · now

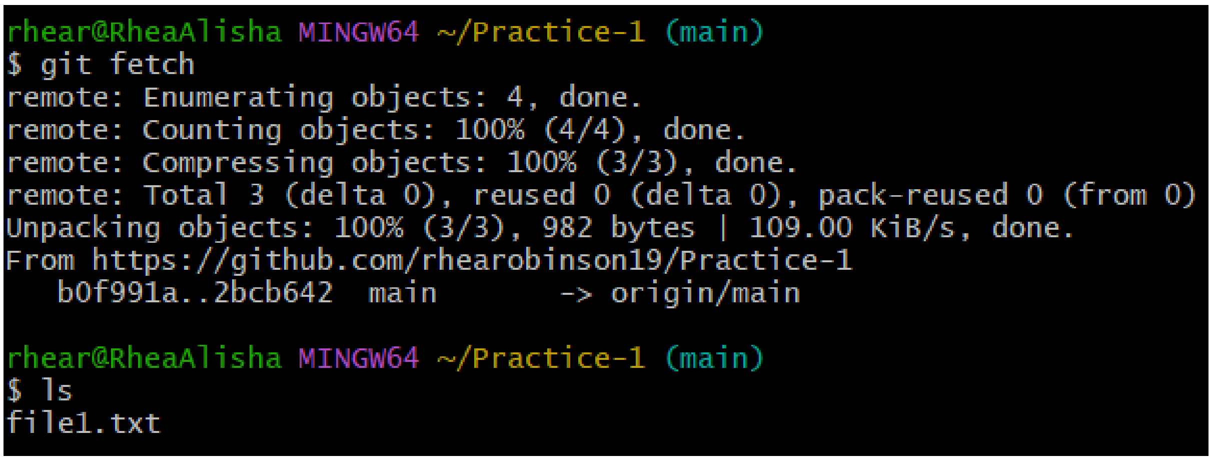
Name	Last commit message	Last commit date
file-2.txt	Create file-2.txt	now
file1.txt	First commit to file 1	17 minutes ago

The new file does not show up on your local, as shown



```
rhear@RheaAlisha MINGW64 ~/Practice-1 (main)
$ git status
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean

rhear@RheaAlisha MINGW64 ~/Practice-1 (main)
$ ls
file1.txt
```

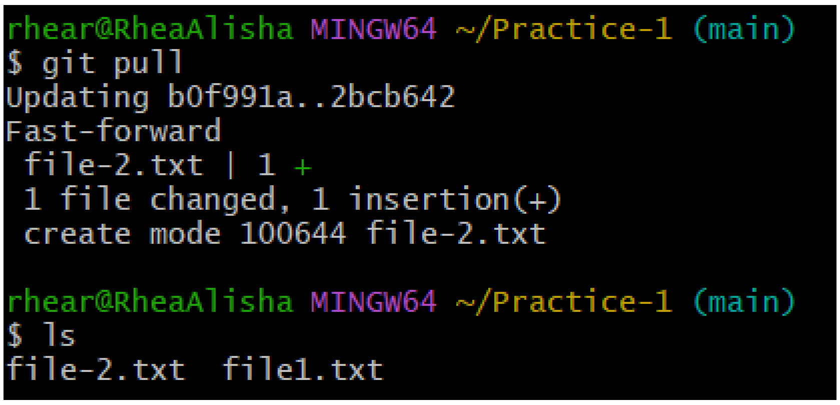
Now, results of **git fetch**



```
rhear@RheaAlisha MINGW64 ~/Practice-1 (main)
$ git fetch
remote: Enumerating objects: 4, done.
remote: Counting objects: 100% (4/4), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
Unpacking objects: 100% (3/3), 982 bytes | 109.00 KiB/s, done.
From https://github.com/rhearobinson19/Practice-1
   b0f991a..2bcb642  main       -> origin/main

rhear@RheaAlisha MINGW64 ~/Practice-1 (main)
$ ls
file1.txt
```

As observable, the command fetches the changes from the remote (proof of change is the change in COMMIT ID from `b0f991a` to `2bcb642`) but does not merge them into your local branch as seen in the results of the ls (file-2 not reflected).

Further, we have the results of **git pull**



```
rhear@RheaAlisha MINGW64 ~/Practice-1 (main)
$ git pull
Updating b0f991a..2bcb642
Fast-forward
 file-2.txt | 1 +
 1 file changed, 1 insertion(+)
 create mode 100644 file-2.txt

rhear@RheaAlisha MINGW64 ~/Practice-1 (main)
$ ls
file-2.txt  file1.txt
```

Again, as observable, the new commits were fetched, and merged into the local using fast forward merging as a result of which the ls command reflects the new file created.

III. BRANCHES

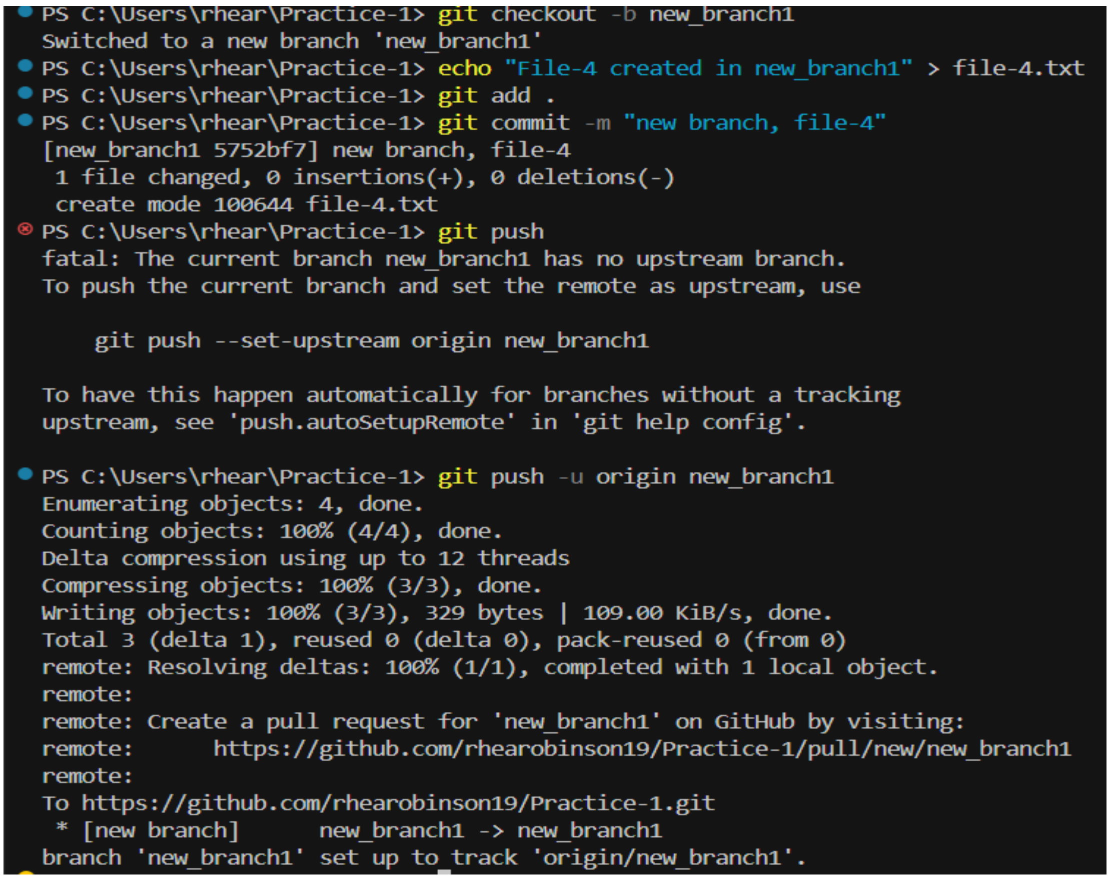
A **branch** in Git is like a separate version of your project where you can work on new features, fixes, or experiments without affecting the main codebase.

`git branch` → Lists all local branches.

`git branch -a` → Lists all local and remote branches.

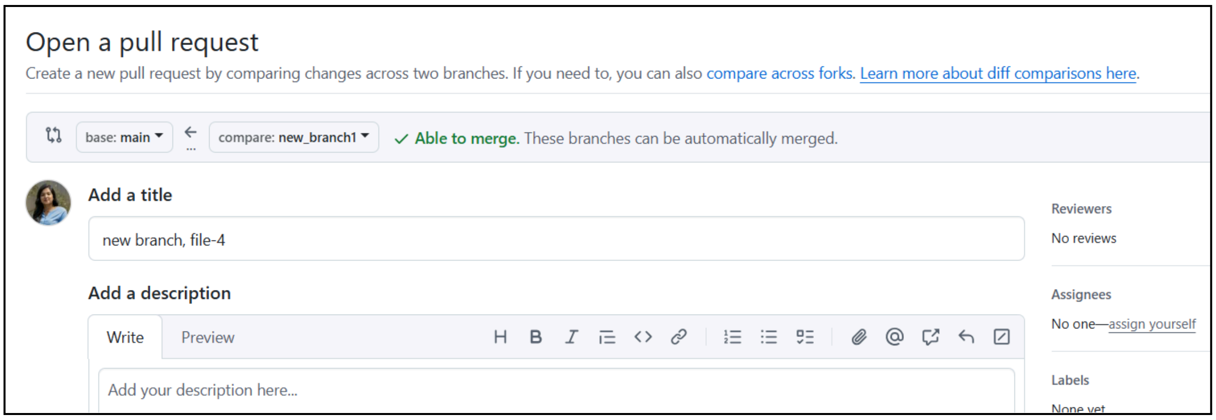
`git branch -r` → Lists only remote branch

1. CREATING A NEW BRANCH



```
PS C:\Users\rhear\Practice-1> git checkout -b new_branch1
Switched to a new branch 'new_branch1'
PS C:\Users\rhear\Practice-1> echo "File-4 created in new_branch1" > file-4.txt
PS C:\Users\rhear\Practice-1> git add .
PS C:\Users\rhear\Practice-1> git commit -m "new branch, file-4"
[new_branch1 5752bf7] new branch, file-4
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 file-4.txt
PS C:\Users\rhear\Practice-1> git push
fatal: The current branch new_branch1 has no upstream branch.
To push the current branch and set the remote as upstream, use

    git push --set-upstream origin new_branch1

To have this happen automatically for branches without a tracking
upstream, see 'push.autoSetupRemote' in 'git help config'.

PS C:\Users\rhear\Practice-1> git push -u origin new_branch1
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 12 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 329 bytes | 109.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
remote:
remote: Create a pull request for 'new_branch1' on GitHub by visiting:
remote:      https://github.com/rhearobinson19/Practice-1/pull/new/new_branch1
remote:
To https://github.com/rhearobinson19/Practice-1.git
 * [new branch]      new_branch1 -> new_branch1
branch 'new_branch1' set up to track 'origin/new_branch1'.
```

2. **MERGING**
You can review, compare and merge on github as a pull request


Open a pull request

Create a new pull request by comparing changes across two branches. If you need to, you can also compare across forks. Learn more about diff comparisons here.

base: main ← compare: new_branch1 ✓ Able to merge. These branches can be automatically merged.

Add a title

new branch, file-4

Add a description

Write Preview

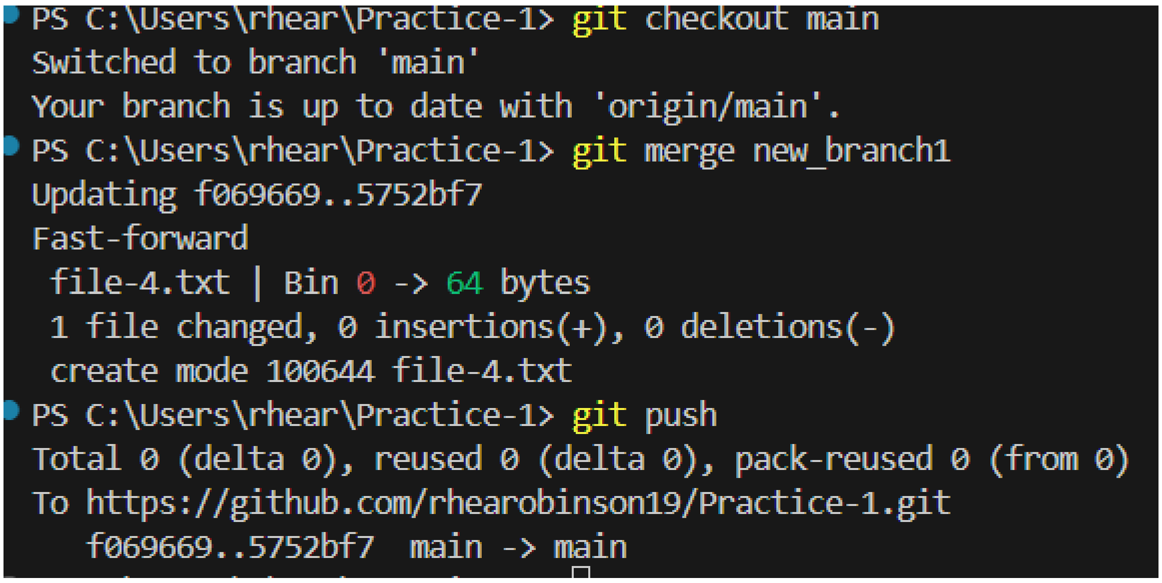
Add your description here...

Reviewers
No reviews

Assignees
No one—assign yourself

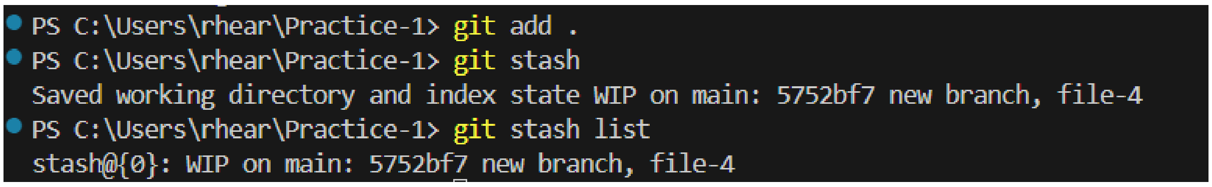
Labels
None yet

Merging on local and pushing changes to remote

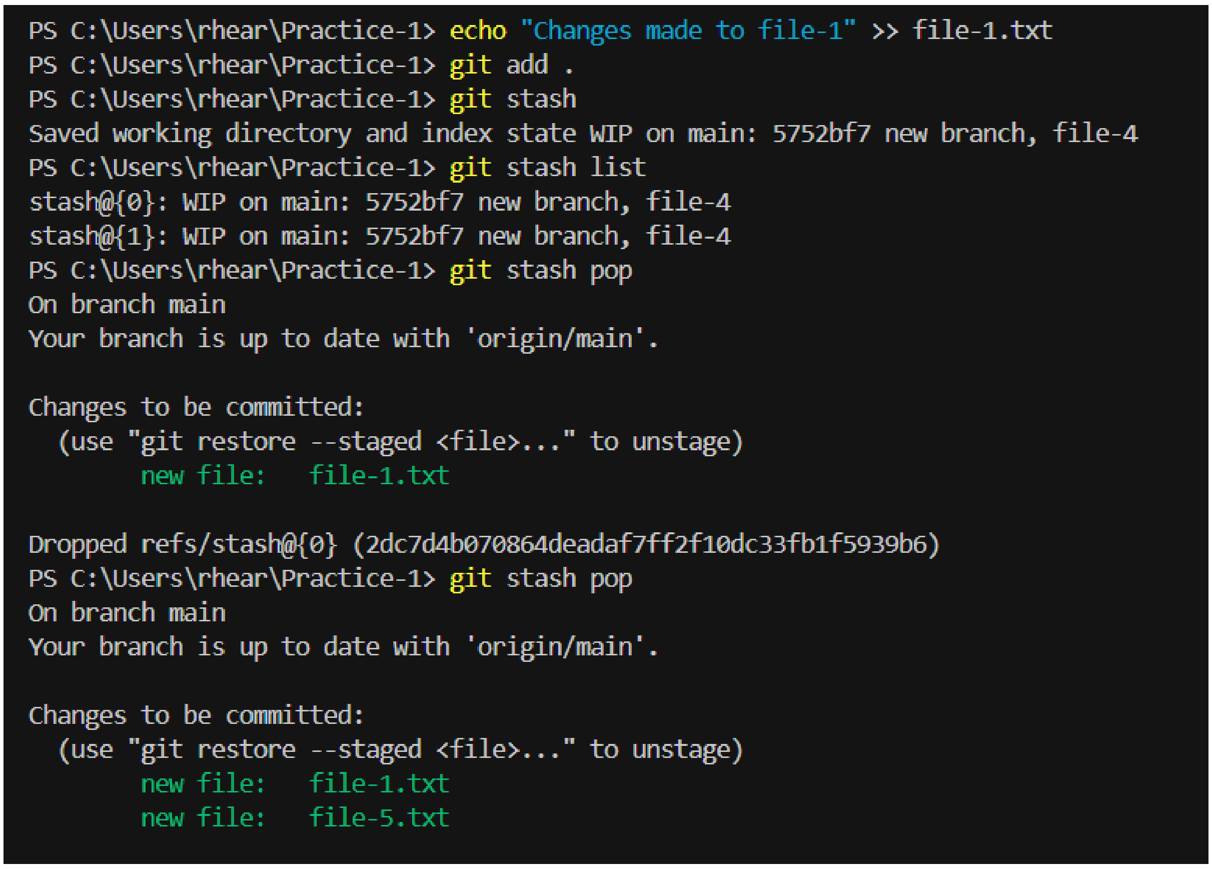

```
PS C:\Users\rhear\Practice-1> git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.
PS C:\Users\rhear\Practice-1> git merge new_branch1
Updating f069669..5752bf7
Fast-forward
 file-4.txt | Bin 0 -> 64 bytes
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 file-4.txt
PS C:\Users\rhear\Practice-1> git push
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/rhearobinson19/Practice-1.git
   f069669..5752bf7  main -> main
```

IV. GIT STASH

git stash is used to temporarily save uncommitted changes without committing them, allowing you to switch branches or perform other actions without losing progress.


```
PS C:\Users\rhear\Practice-1> git add .
PS C:\Users\rhear\Practice-1> git stash
Saved working directory and index state WIP on main: 5752bf7 new branch, file-4
PS C:\Users\rhear\Practice-1> git stash list
stash@{0}: WIP on main: 5752bf7 new branch, file-4
```

Applying the stash using **git stash pop** - executes the latest item stashed and removes it from the stash.

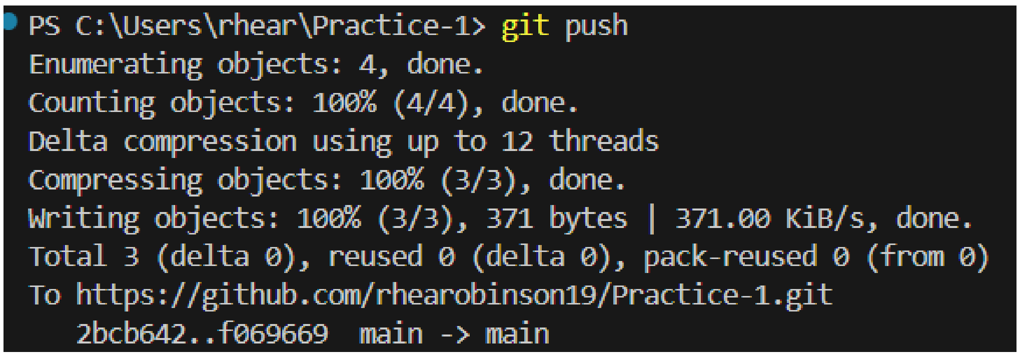


```
PS C:\Users\rhear\Practice-1> echo "Changes made to file-1" >> file-1.txt
PS C:\Users\rhear\Practice-1> git add .
PS C:\Users\rhear\Practice-1> git stash
Saved working directory and index state WIP on main: 5752bf7 new branch, file-4
PS C:\Users\rhear\Practice-1> git stash list
stash@{0}: WIP on main: 5752bf7 new branch, file-4
stash@{1}: WIP on main: 5752bf7 new branch, file-4
PS C:\Users\rhear\Practice-1> git stash pop
On branch main
Your branch is up to date with 'origin/main'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   file-1.txt

Dropped refs/stash@{0} (2dc7d4b070864deadaf7ff2f10dc33fb1f5939b6)
PS C:\Users\rhear\Practice-1> git stash pop
On branch main
Your branch is up to date with 'origin/main'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   file-1.txt
        new file:   file-5.txt
```

KEY LEARNINGS

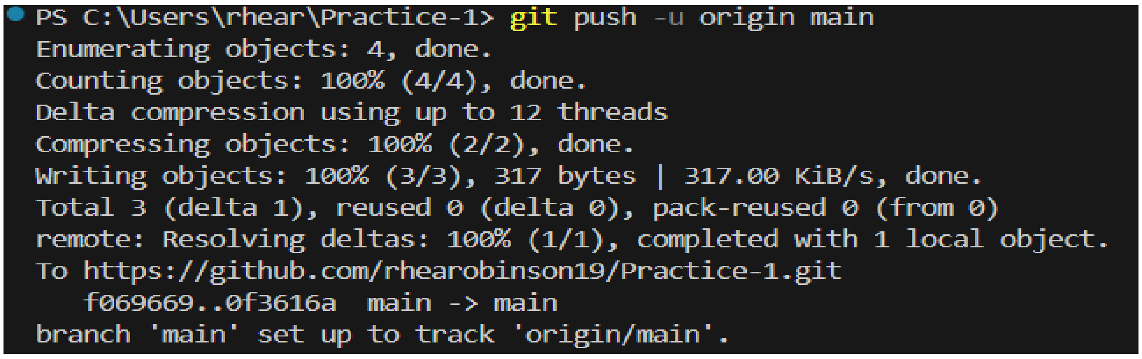
1. **git push v/s git push -u origin main**
   In terms of results, both commands work the same. However the only difference is git push -u origin main is used for the first time to set the upstream to the origin(remote) and push changes to the main branch. Once this is done, git push will work just fine to push changes directly to the remote without explicit mentions.

   git push



```
PS C:\Users\rhear\Practice-1> git push
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 12 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 371 bytes | 371.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/rhearobinson19/Practice-1.git
   2bcb642..f069669  main -> main
```

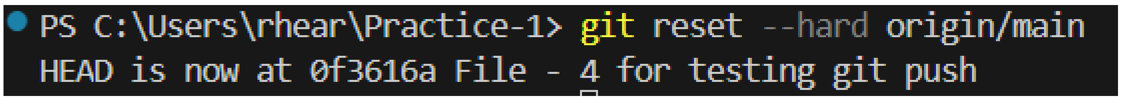
git push -u origin main



```
PS C:\Users\rhear\Practice-1> git push -u origin main
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 12 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 317 bytes | 317.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/rhearobinson19/Practice-1.git
   f069669..0f3616a  main -> main
branch 'main' set up to track 'origin/main'.
```

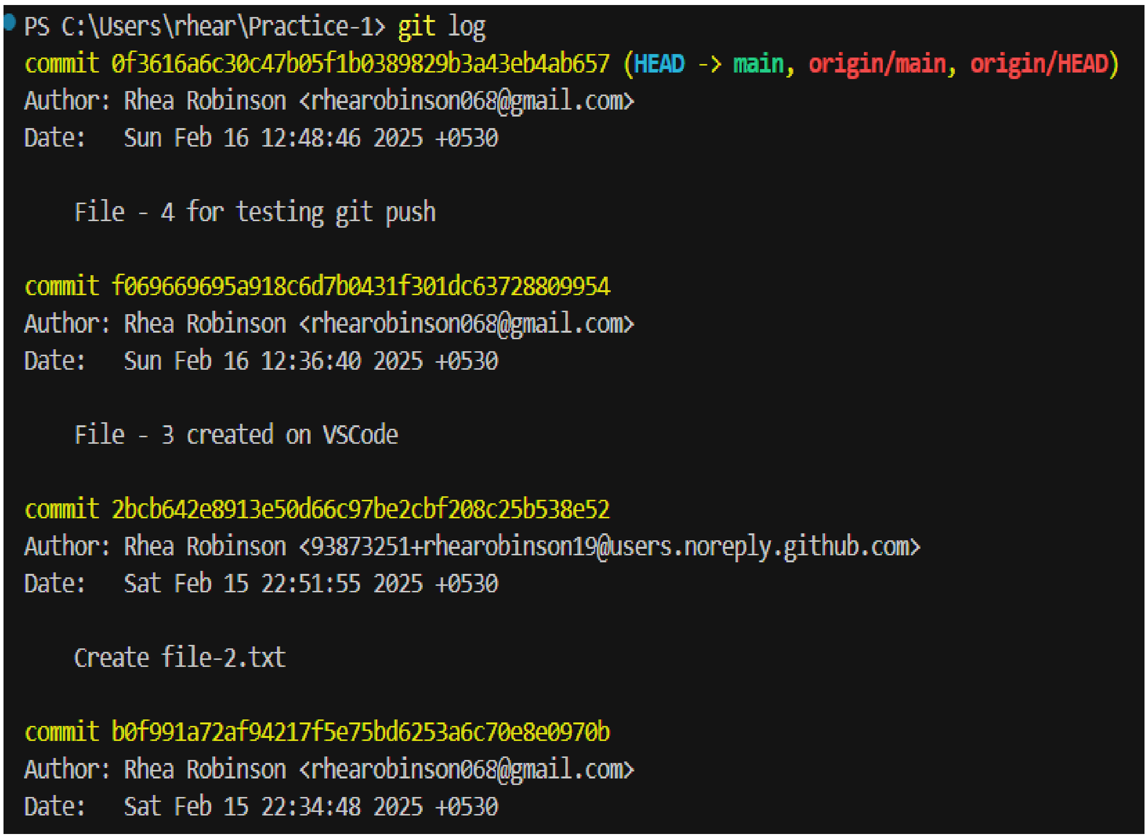
2. **git reset v/s git revert**

   Discards all local changes (staged & unstaged) and resets the working directory to a specific commit (latest commit if not mentioned)

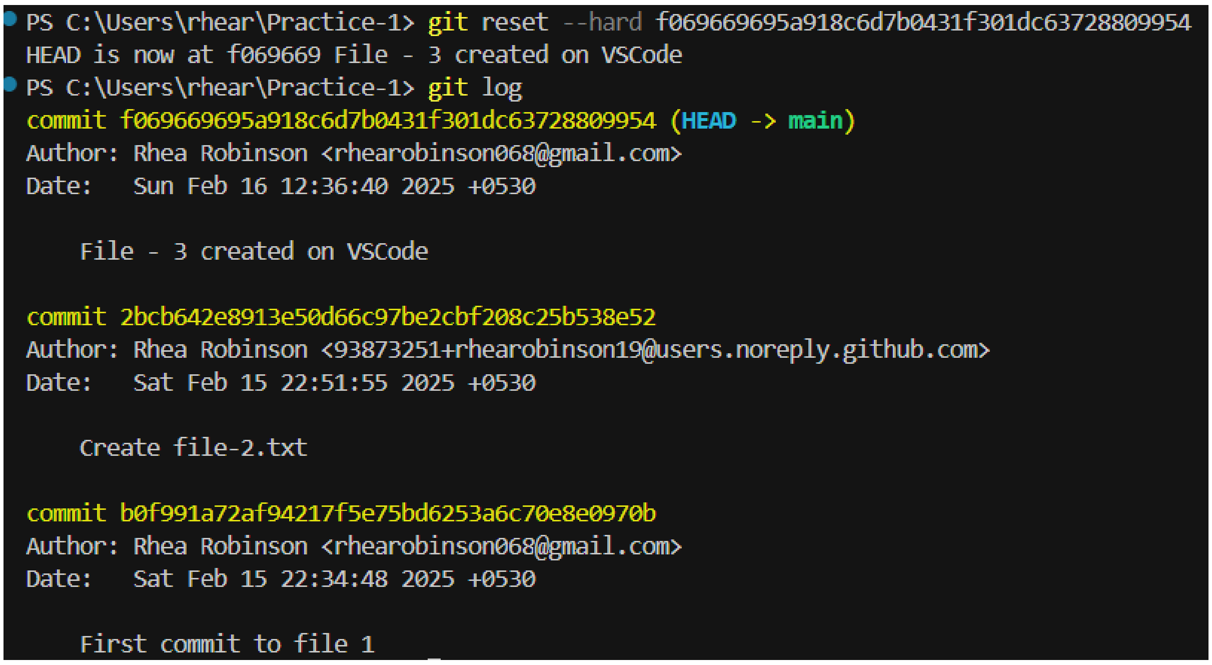


```
PS C:\Users\rhear\Practice-1> git reset --hard origin/main
HEAD is now at 0f3616a File - 4 for testing git push
```

Discarding the latest commit

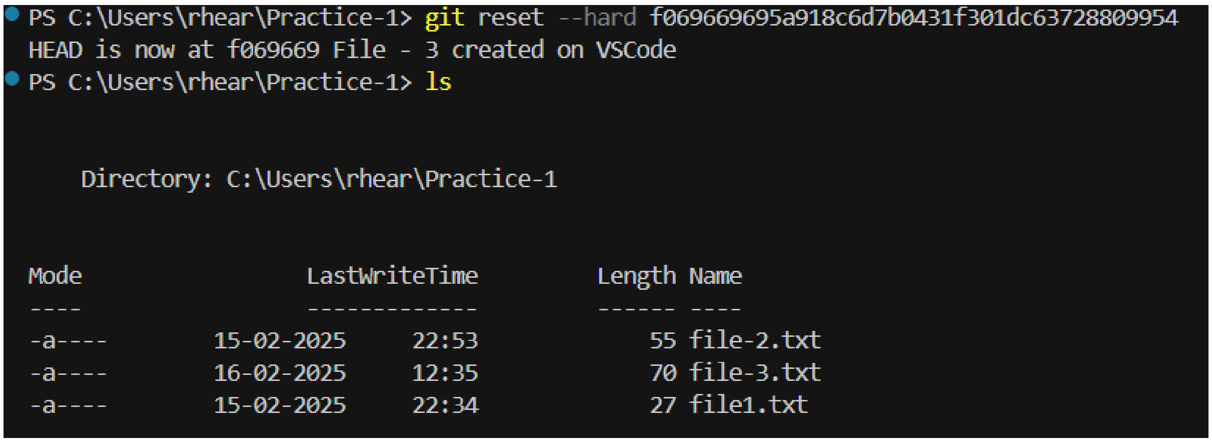


```
PS C:\Users\rhear\Practice-1> git log
commit 0f3616a6c30c47b05f1b0389829b3a43eb4ab657 (HEAD -> main, origin/main, origin/HEAD)
Author: Rhea Robinson <rhearobinson068@gmail.com>
Date:   Sun Feb 16 12:48:46 2025 +0530

    File - 4 for testing git push

commit f069669695a918c6d7b0431f301dc63728809954
Author: Rhea Robinson <rhearobinson068@gmail.com>
Date:   Sun Feb 16 12:36:40 2025 +0530

    File - 3 created on VSCode

commit 2bcb642e8913e50d66c97be2cbf208c25b538e52
Author: Rhea Robinson <93873251+rhearobinson19@users.noreply.github.com>
Date:   Sat Feb 15 22:51:55 2025 +0530

    Create file-2.txt

commit b0f991a72af94217f5e75bd6253a6c70e8e0970b
Author: Rhea Robinson <rhearobinson068@gmail.com>
Date:   Sat Feb 15 22:34:48 2025 +0530
```

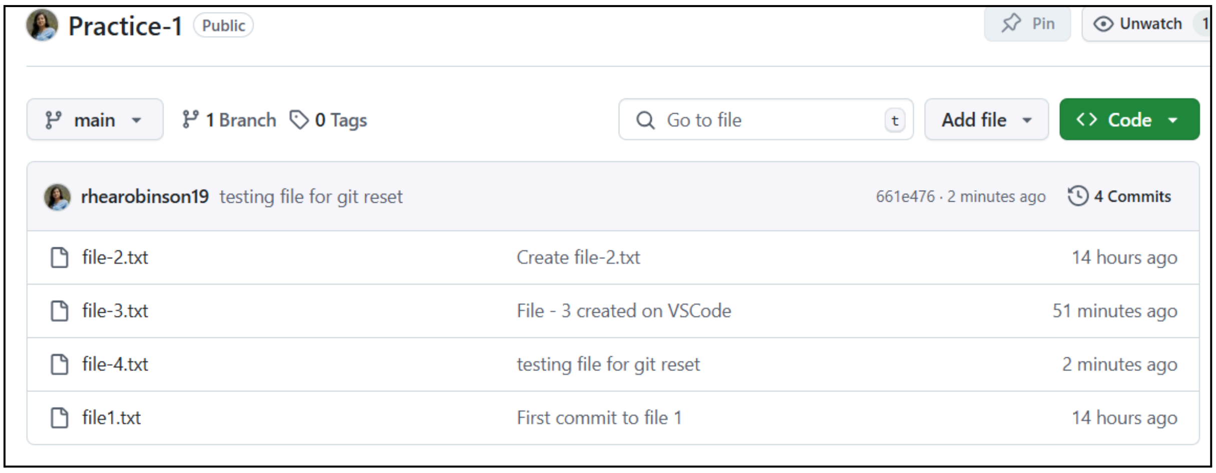


```
PS C:\Users\rhear\Practice-1> git reset --hard f069669695a918c6d7b0431f301dc63728809954
HEAD is now at f069669 File - 3 created on VSCode
PS C:\Users\rhear\Practice-1> git log
commit f069669695a918c6d7b0431f301dc63728809954 (HEAD -> main)
Author: Rhea Robinson <rhearobinson068@gmail.com>
Date:   Sun Feb 16 12:36:40 2025 +0530

    File - 3 created on VSCode

commit 2bcb642e8913e50d66c97be2cbf208c25b538e52
Author: Rhea Robinson <93873251+rhearobinson19@users.noreply.github.com>
Date:   Sat Feb 15 22:51:55 2025 +0530

    Create file-2.txt

commit b0f991a72af94217f5e75bd6253a6c70e8e0970b
Author: Rhea Robinson <rhearobinson068@gmail.com>
Date:   Sat Feb 15 22:34:48 2025 +0530

    First commit to file 1
```

When you just do git reset <commit_id> it just changes the head reference. You need to force push the changes to reflect it on the remote. Image below shows file-4 deleted from local but still showing up on remote



```
PS C:\Users\rhear\Practice-1> git reset --hard f069669695a918c6d7b0431f301dc63728809954
HEAD is now at f069669 File - 3 created on VSCode
PS C:\Users\rhear\Practice-1> ls


    Directory: C:\Users\rhear\Practice-1


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----        15-02-2025     22:53             55 file-2.txt
-a----        16-02-2025     12:35             70 file-3.txt
-a----        15-02-2025     22:34             27 file1.txt
```



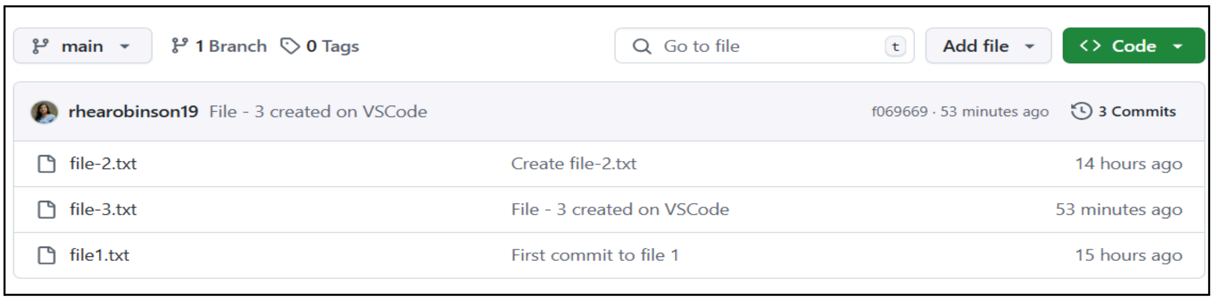
Practice-1 Public

main · 1 Branch · 0 Tags

rhearobinson19 testing file for git reset · 661e476 · 2 minutes ago · 4 Commits

file-2.txt	Create file-2.txt	14 hours ago
file-3.txt	File - 3 created on VSCode	51 minutes ago
file-4.txt	testing file for git reset	2 minutes ago
file1.txt	First commit to file 1	14 hours ago

git push --force origin main



rhearobinson19 File - 3 created on VSCode f069669 · 53 minutes ago 3 Commits

file-2.txt	Create file-2.txt	14 hours ago
file-3.txt	File - 3 created on VSCode	53 minutes ago
file1.txt	First commit to file 1	15 hours ago

***git revert is a better option because this creates a new commit that undoes the changes in `file-4`, instead of rewriting history.**