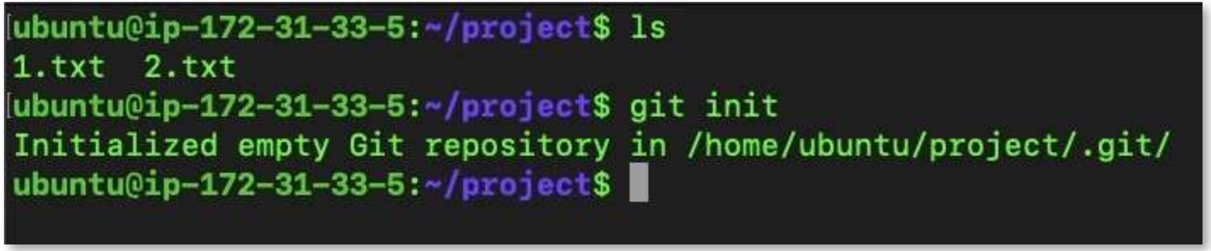
Common Git Commands

Common Git Commands – git init

- Creating Repository
- Making Changes
- Syncing Repositories
- Parallel Development

You can create a repository using the command git init. Navigate to your project folder and enter the command git init to initialize a git repository for your project on the local system



```
ubuntu@ip-172-31-33-5:~/project$ ls
1.txt  2.txt
ubuntu@ip-172-31-33-5:~/project$ git init
Initialized empty Git repository in /home/ubuntu/project/.git/
ubuntu@ip-172-31-33-5:~/project$ 
```

Common Git Commands – git status

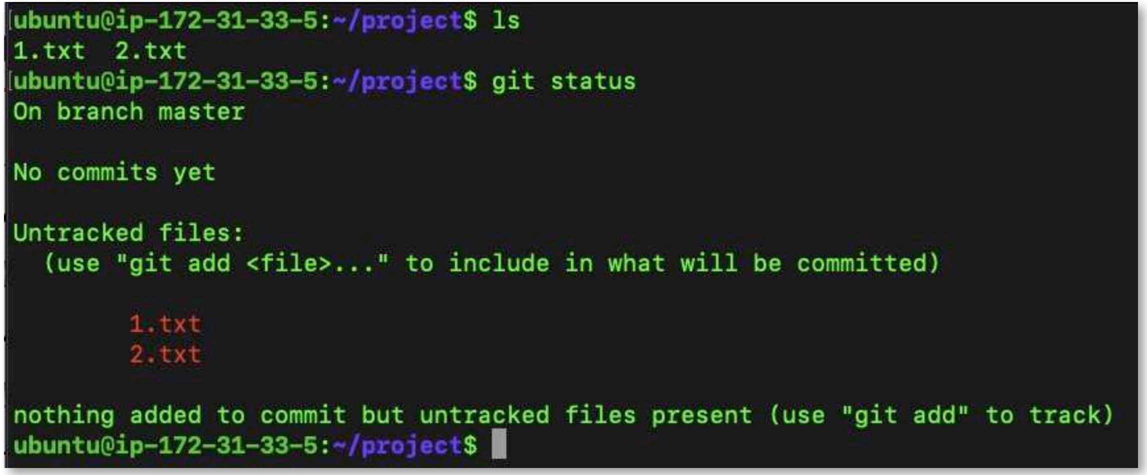
Creating Repository

Making Changes

Syncing Repositories

Parallel Development

Once the directory has been initialized you can check the status of the files, whether they are being tracked by git or not, using the command **git status**



```
ubuntu@ip-172-31-33-5:~/project$ ls
1.txt  2.txt
ubuntu@ip-172-31-33-5:~/project$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)

        1.txt
        2.txt

nothing added to commit but untracked files present (use "git add" to track)
ubuntu@ip-172-31-33-5:~/project$
```

Common Git Commands – git add

Creating Repository

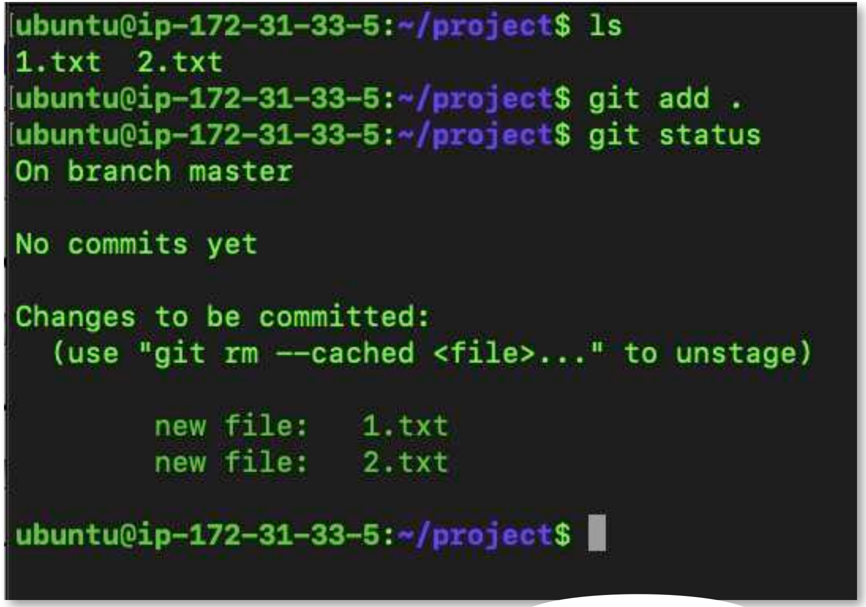
Making Changes

Syncing Repositories

Parallel Development

Since no files are being tracked right now, let us now stage these files. For that, enter the command **git add.** If we want to track all the files in the project folder, we can type the command,

git add .



```
ubuntu@ip-172-31-33-5:~/project$ ls
1.txt  2.txt
ubuntu@ip-172-31-33-5:~/project$ git add .
ubuntu@ip-172-31-33-5:~/project$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)

        new file:   1.txt
        new file:   2.txt

ubuntu@ip-172-31-33-5:~/project$
```

Common Git Commands – git commit

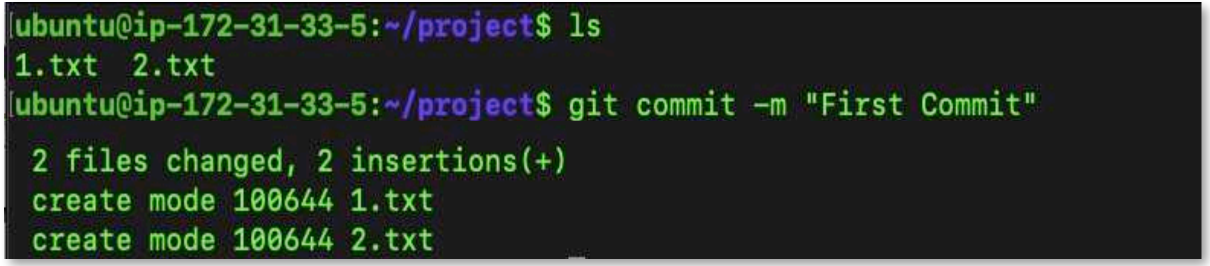
Creating Repository

Making Changes

Syncing Repositories

Parallel Development

Once the files or changes have been staged, we are ready to commit them in our repository. We can commit the files using the command **git commit –m "custom message"**



```
ubuntu@ip-172-31-33-5:~/project$ ls
1.txt  2.txt
ubuntu@ip-172-31-33-5:~/project$ git commit -m "First Commit"

 2 files changed, 2 insertions(+)
 create mode 100644 1.txt
 create mode 100644 2.txt
```

Common Git Commands – git remote

Creating Repository

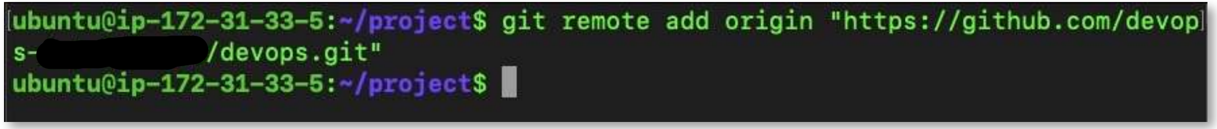
Making Changes

Syncing Repositories

Parallel Development

Once everything is ready on our local, we can start pushing our changes to the remote repository. Copy your repository link and paste it in the command

git remote add origin “<URL to repository>”



```
ubuntu@ip-172-31-33-5:~/project$ git remote add origin "https://github.com/devop
s-[redacted]/devops.git"
ubuntu@ip-172-31-33-5:~/project$ 
```

Common Git Commands – git push

Creating Repository

Making Changes

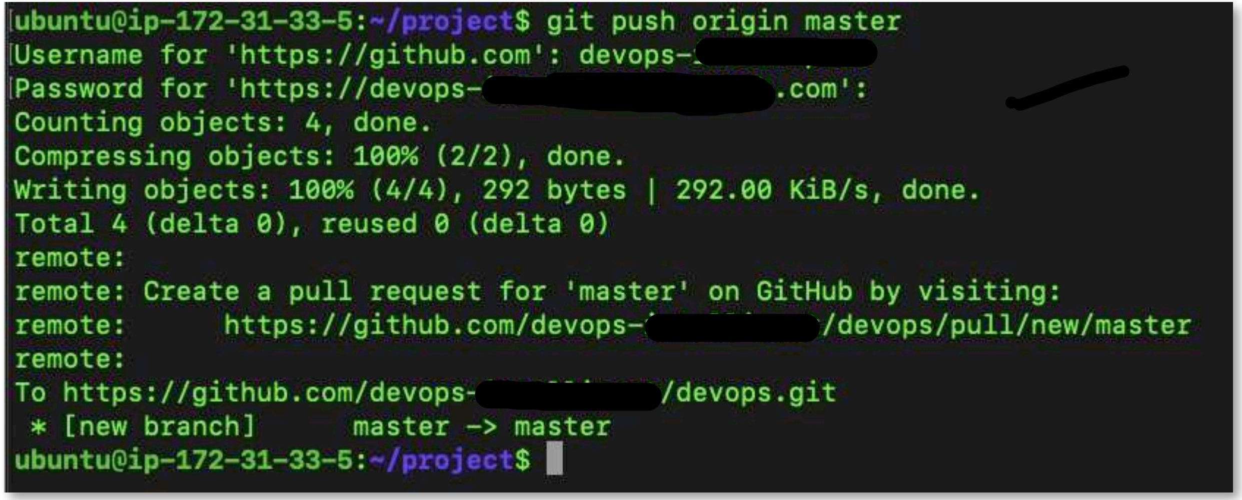
Syncing Repositories

Parallel Development

To push the changes to your repository, enter the command git push origin <branch-name> and hit enter. In our case the branch is master, hence

git push origin master

This command will then prompt for username and password, enter the values and hit enter.



```
ubuntu@ip-172-31-33-5:~/project$ git push origin master
Username for 'https://github.com': devops-[redacted]
Password for 'https://devops-[redacted].com':
Counting objects: 4, done.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (4/4), 292 bytes | 292.00 KiB/s, done.
Total 4 (delta 0), reused 0 (delta 0)
remote:
remote: Create a pull request for 'master' on GitHub by visiting:
remote:      https://github.com/devops-[redacted]/devops/pull/new/master
remote:
To https://github.com/devops-[redacted]/devops.git
 * [new branch]      master -> master
ubuntu@ip-172-31-33-5:~/project$
```

Common Git Commands – git push

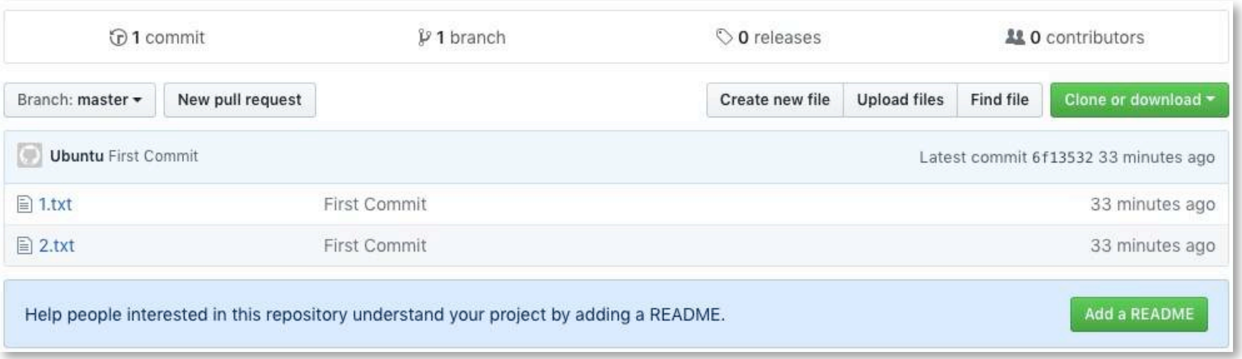
Creating Repository

Making Changes

Syncing Repositories

Parallel Development

Your local repository is now synced with the remote repository on github



1 commit | 1 branch | 0 releases | 0 contributors

Ubuntu First Commit | Latest commit 6f13532 33 minutes ago

1.txt	First Commit	33 minutes ago
2.txt	First Commit	33 minutes ago

Help people interested in this repository understand your project by adding a README.

Common Git Commands – git clone

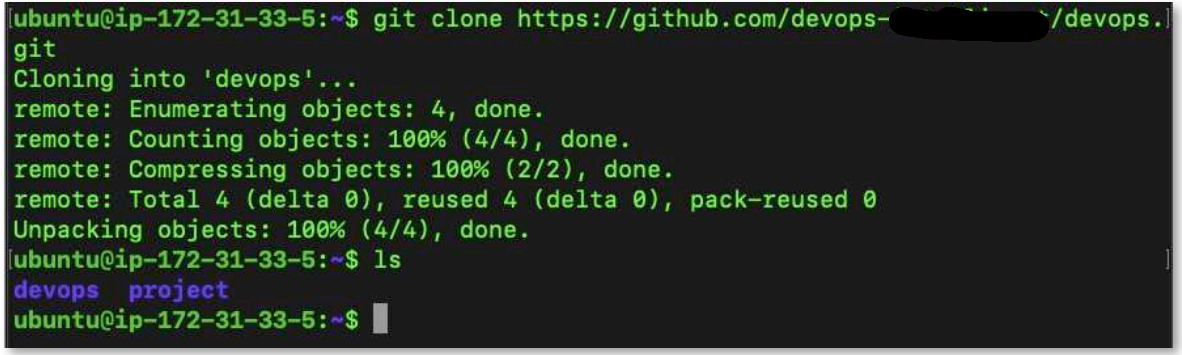
Creating Repository

Making Changes

Syncing Repositories

Parallel Development

Similarly, if we want to download the remote repository to our local system, we can use the command **git clone <URL>**. This command will create a folder with the repository name, and download all the contents of the repository inside this folder. In our example, repository contents were downloaded into the ”devops” folder.



```
ubuntu@ip-172-31-33-5:~$ git clone https://github.com/devops-[redacted]/devops.
git
Cloning into 'devops'...
remote: Enumerating objects: 4, done.
remote: Counting objects: 100% (4/4), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 4 (delta 0), reused 4 (delta 0), pack-reused 0
Unpacking objects: 100% (4/4), done.
ubuntu@ip-172-31-33-5:~$ ls
devops  project
ubuntu@ip-172-31-33-5:~$ 
```

Common Git Commands – git pull

Creating Repository

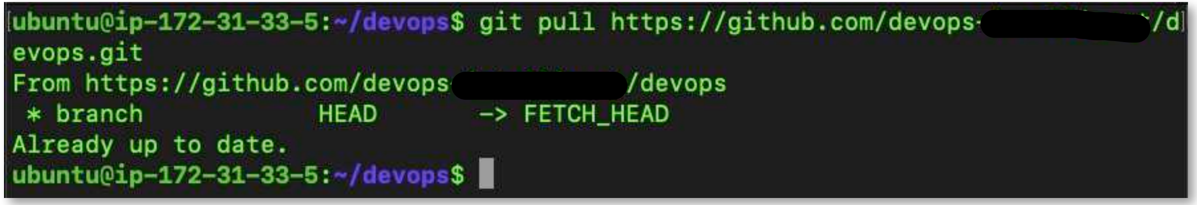
Making Changes

Syncing Repositories

Parallel Development

The git pull command is also used for pulling the latest changes from the repository, unlike git clone, this command can only work inside an initialized git repository. This command is used when you are already working in the cloned repository, and want to pull the latest changes, that others might have pushed to the remote repository

git pull <URL of link>



```
[ubuntu@ip-172-31-33-5:~/devops$ git pull https://github.com/devops-[redacted]/d
evops.git
From https://github.com/devops[redacted]/devops
 * branch            HEAD       -> FETCH_HEAD
Already up to date.
ubuntu@ip-172-31-33-5:~/devops$
```

Common Git Commands – git branch

Creating Repository

Making Changes

Syncing Repositories

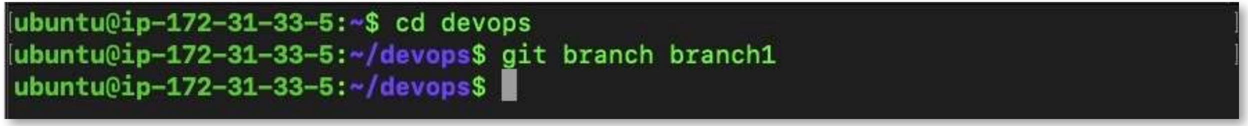
Parallel Development

Until now, we saw how you can work on git. But now imagine, multiple developers working on the same project or repository. To handle the workspace of multiple developers, we use branches. To create a branch from an existing branch, we type

git branch <name-of-new-branch>

Similarly, to delete a branch use the command

git branch –D <branch name>



```
ubuntu@ip-172-31-33-5:~$ cd devops
ubuntu@ip-172-31-33-5:~/devops$ git branch branch1
ubuntu@ip-172-31-33-5:~/devops$ 
```

Common Git Commands – git checkout

Creating Repository

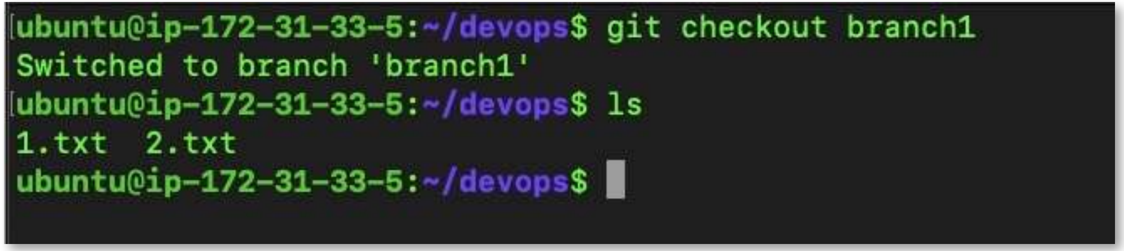
Making Changes

Syncing Repositories

Parallel Development

To switch to the new branch, we type the command

git checkout <branch-name>

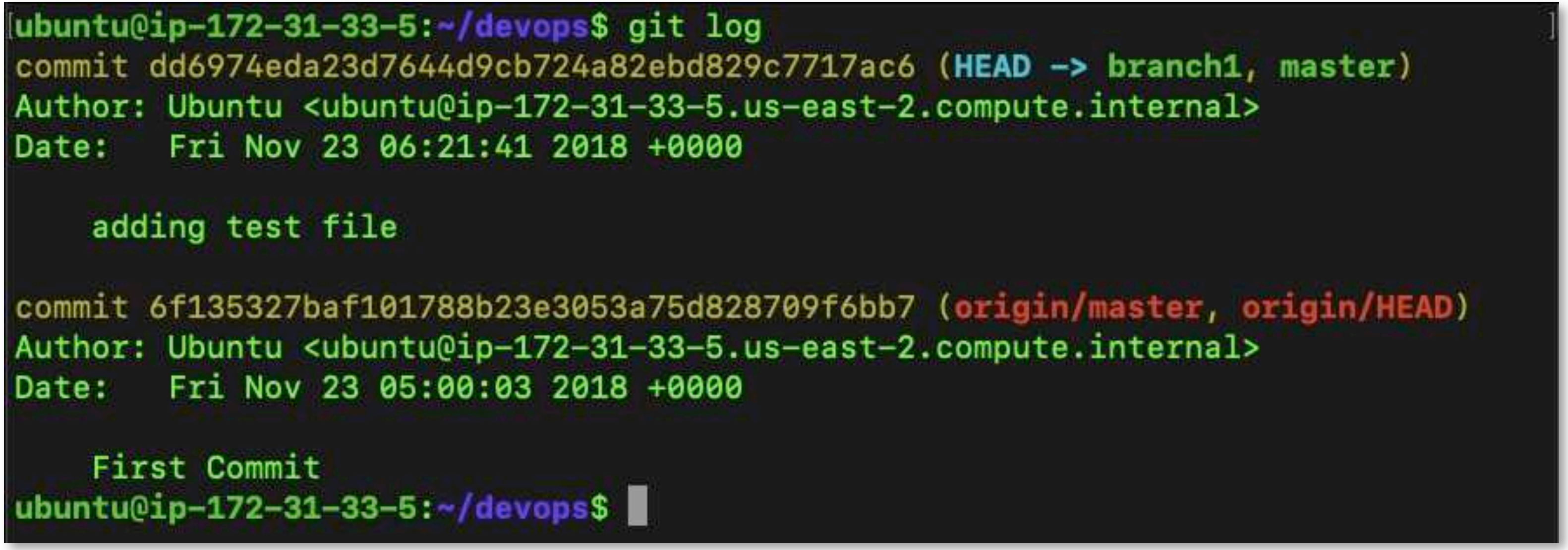


```
ubuntu@ip-172-31-33-5:~/devops$ git checkout branch1
Switched to branch 'branch1'
ubuntu@ip-172-31-33-5:~/devops$ ls
1.txt  2.txt
ubuntu@ip-172-31-33-5:~/devops$
```

Common Git Commands – git log

Want to check the log for every commit detail in your repository?
You can accomplish that using the command

git log



```
ubuntu@ip-172-31-33-5:~/devops$ git log
commit dd6974eda23d7644d9cb724a82ebd829c7717ac6 (HEAD -> branch1, master)
Author: Ubuntu <ubuntu@ip-172-31-33-5.us-east-2.compute.internal>
Date:   Fri Nov 23 06:21:41 2018 +0000

    adding test file

commit 6f135327baf101788b23e3053a75d828709f6bb7 (origin/master, origin/HEAD)
Author: Ubuntu <ubuntu@ip-172-31-33-5.us-east-2.compute.internal>
Date:   Fri Nov 23 05:00:03 2018 +0000

    First Commit
ubuntu@ip-172-31-33-5:~/devops$
```

Common Git Commands – git stash

Creating Repository

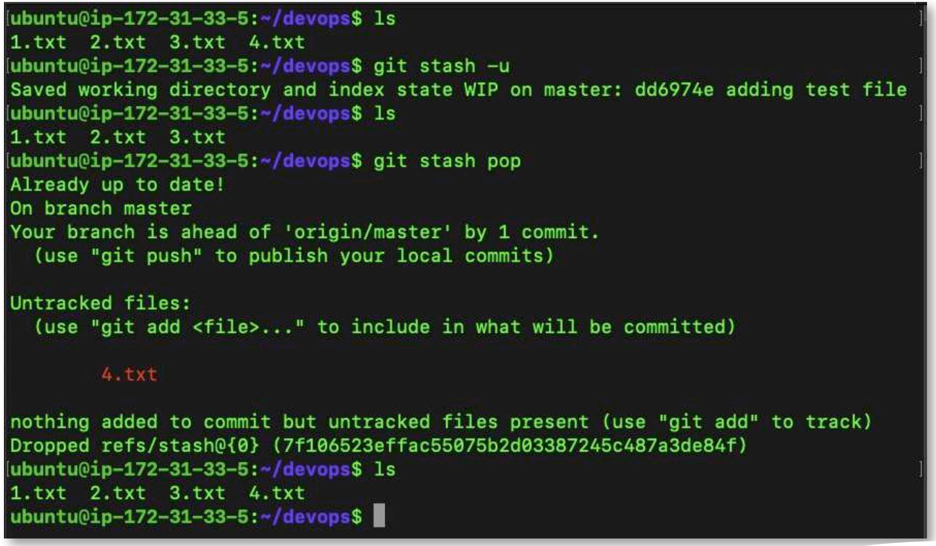
Making Changes

Syncing Repositories

Parallel Development

Want to save your work without committing the code? Git has got you covered. This can be helpful when you want to switch branches, but do not want to save your work to your git repository. To stash your staged files without committing just type in **git stash.** If you want to stash your untracked files as well, type **git stash –u**.

Once you are back and want to retrieve working, type in **git stash pop**



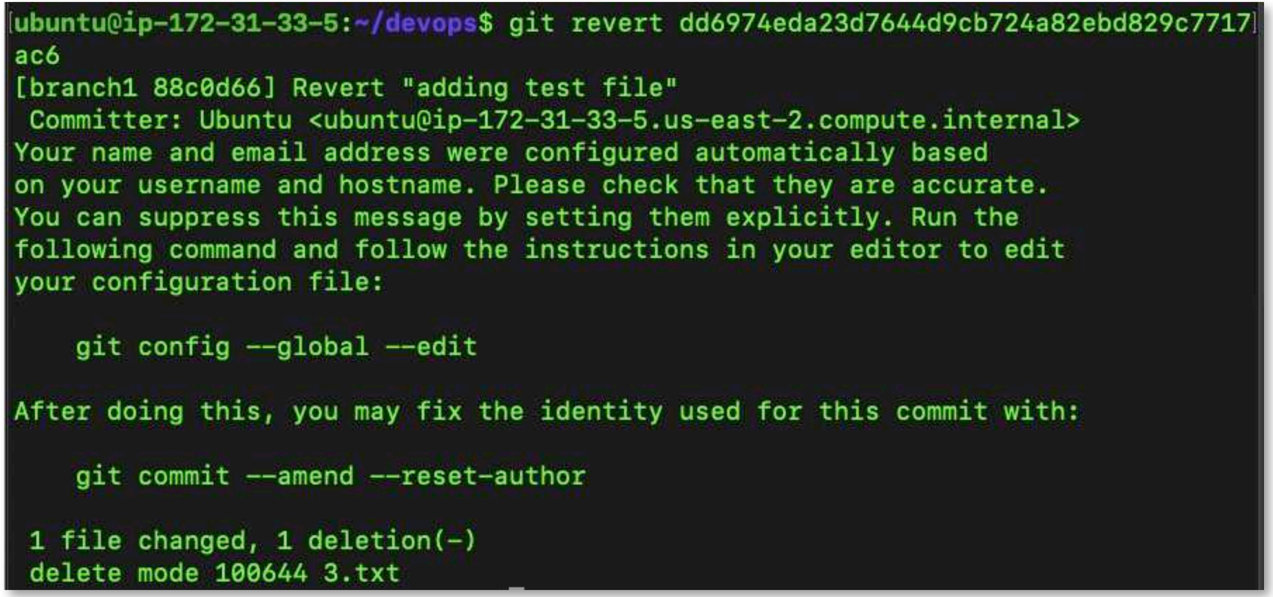
```
ubuntu@ip-172-31-33-5:~/devops$ ls
1.txt  2.txt  3.txt  4.txt
ubuntu@ip-172-31-33-5:~/devops$ git stash -u
Saved working directory and index state WIP on master: dd6974e adding test file
ubuntu@ip-172-31-33-5:~/devops$ ls
1.txt  2.txt  3.txt
ubuntu@ip-172-31-33-5:~/devops$ git stash pop
Already up to date!
On branch master
Your branch is ahead of 'origin/master' by 1 commit.
  (use "git push" to publish your local commits)

Untracked files:
  (use "git add <file>..." to include in what will be committed)

        4.txt

nothing added to commit but untracked files present (use "git add" to track)
Dropped refs/stash@{0} (7f106523effac55075b2d03387245c487a3de84f)
ubuntu@ip-172-31-33-5:~/devops$ ls
1.txt  2.txt  3.txt  4.txt
ubuntu@ip-172-31-33-5:~/devops$ 
```

Common Git Commands – git revert

This command helps you in reverting a commit, to a previous version

git revert <commit-id>

<commit-id> can be obtained from the output of **git log**



```
ubuntu@ip-172-31-33-5:~/devops$ git revert dd6974eda23d7644d9cb724a82ebd829c7717
ac6
[branch1 88c0d66] Revert "adding test file"
 Committer: Ubuntu <ubuntu@ip-172-31-33-5.us-east-2.compute.internal>
Your name and email address were configured automatically based
on your username and hostname. Please check that they are accurate.
You can suppress this message by setting them explicitly. Run the
following command and follow the instructions in your editor to edit
your configuration file:

    git config --global --edit

After doing this, you may fix the identity used for this commit with:

    git commit --amend --reset-author

 1 file changed, 1 deletion(-)
 delete mode 100644 3.txt
```