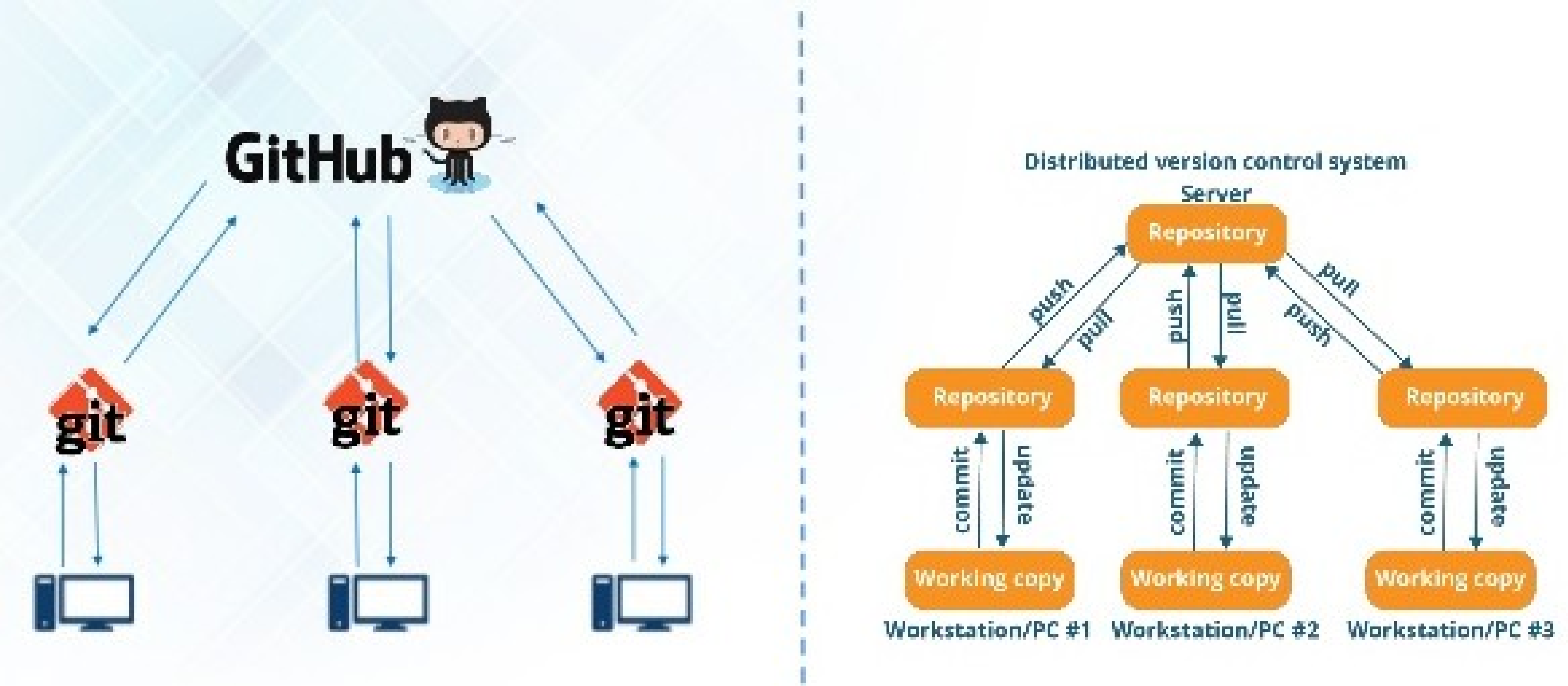
Introduction to git and github

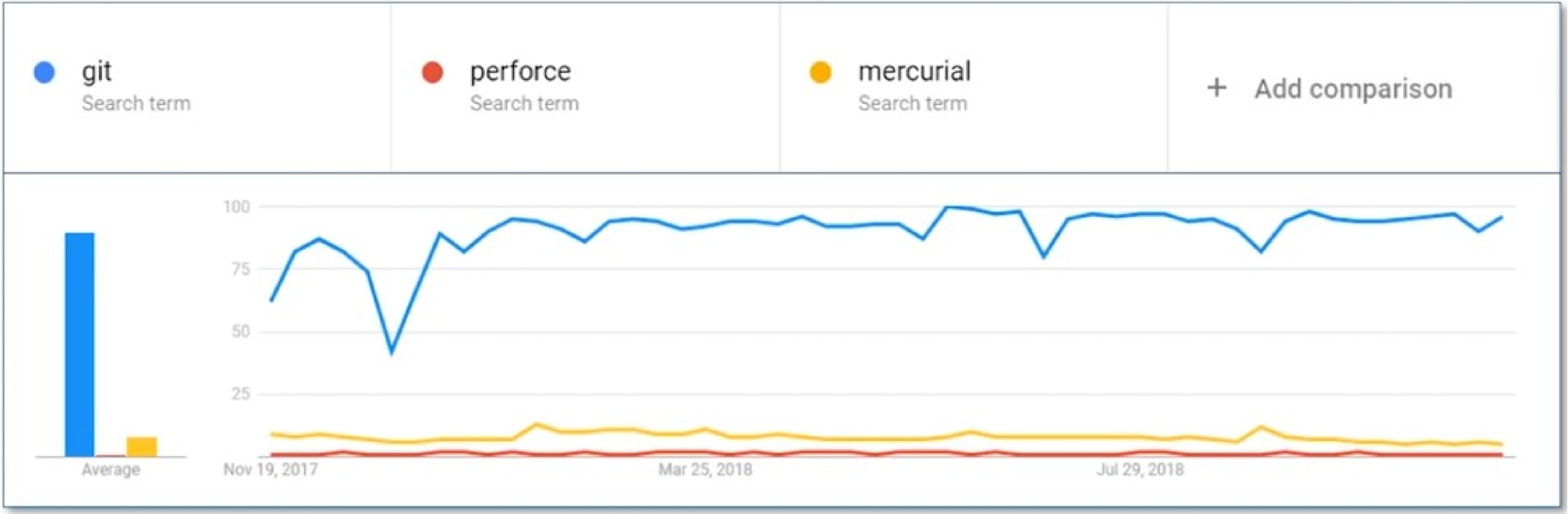
Git and github



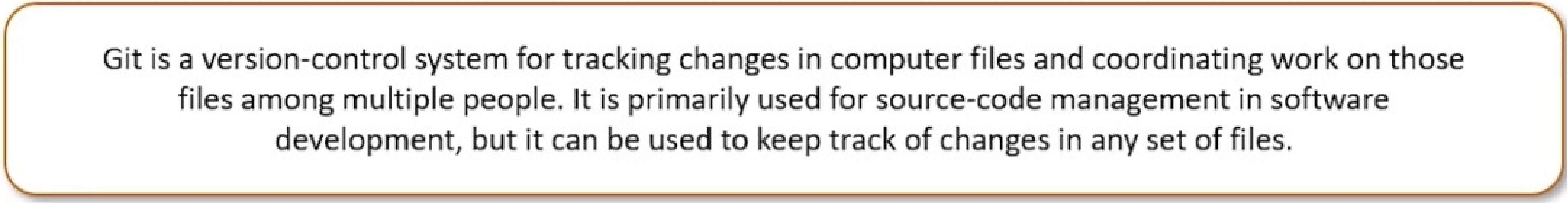
]

Why Git?

Git is the most popular tool among all the DVCS tools.

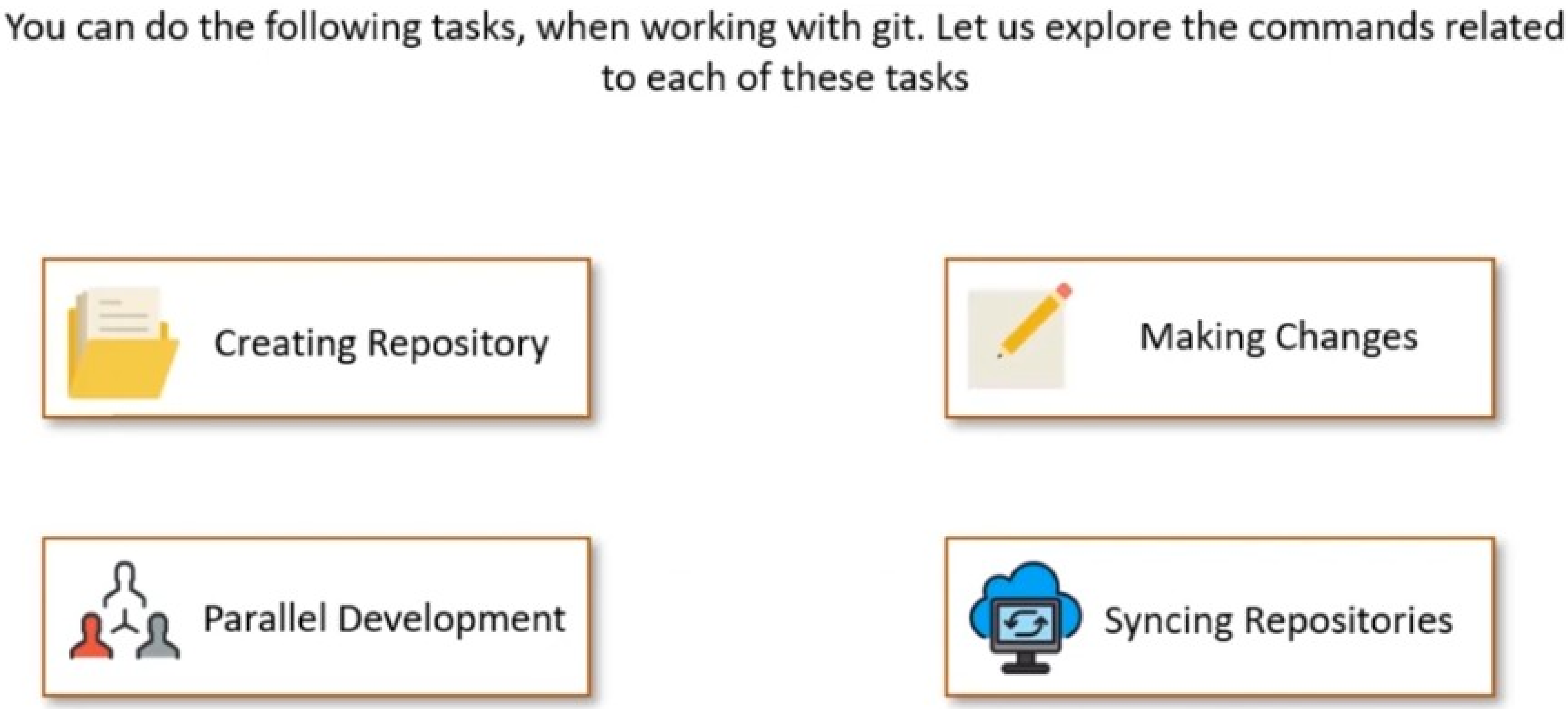


What is git?



Git is a version-control system for tracking changes in computer files and coordinating work on those files among multiple people. It is primarily used for source-code management in software development, but it can be used to keep track of changes in any set of files.

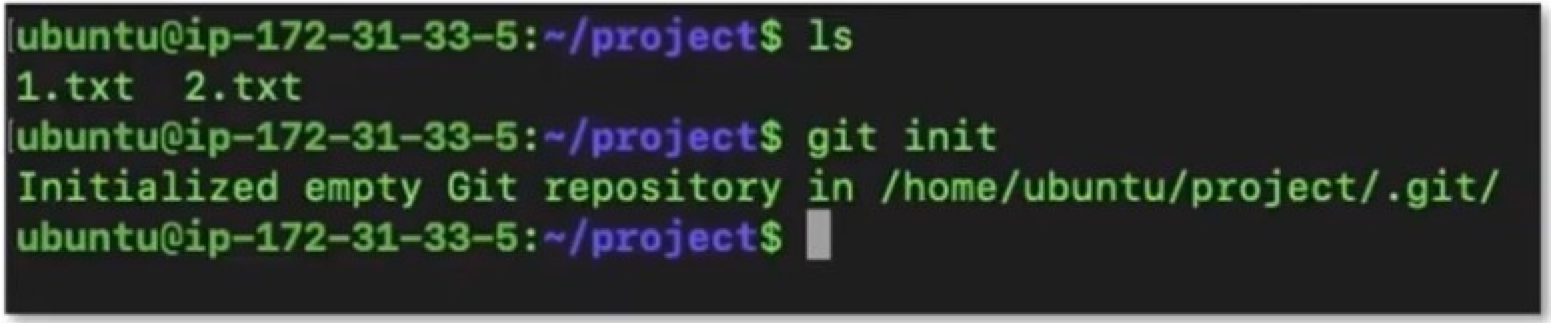
Common git commands



You can do the following tasks, when working with git. Let us explore the commands related to each of these tasks

Git init

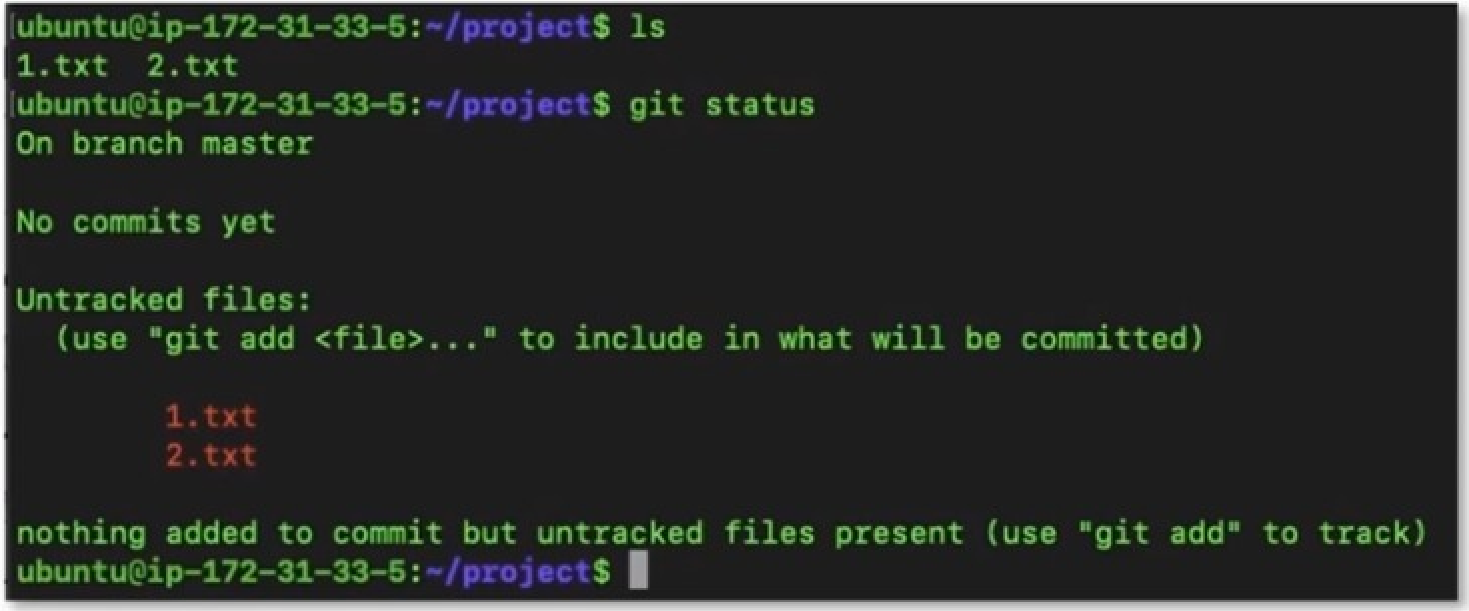
You can create a repository using the command git init. Navigate to your project folder and enter the command git init to initialize a git repository for your project on the local system



```
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$ 
```

Git status

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**

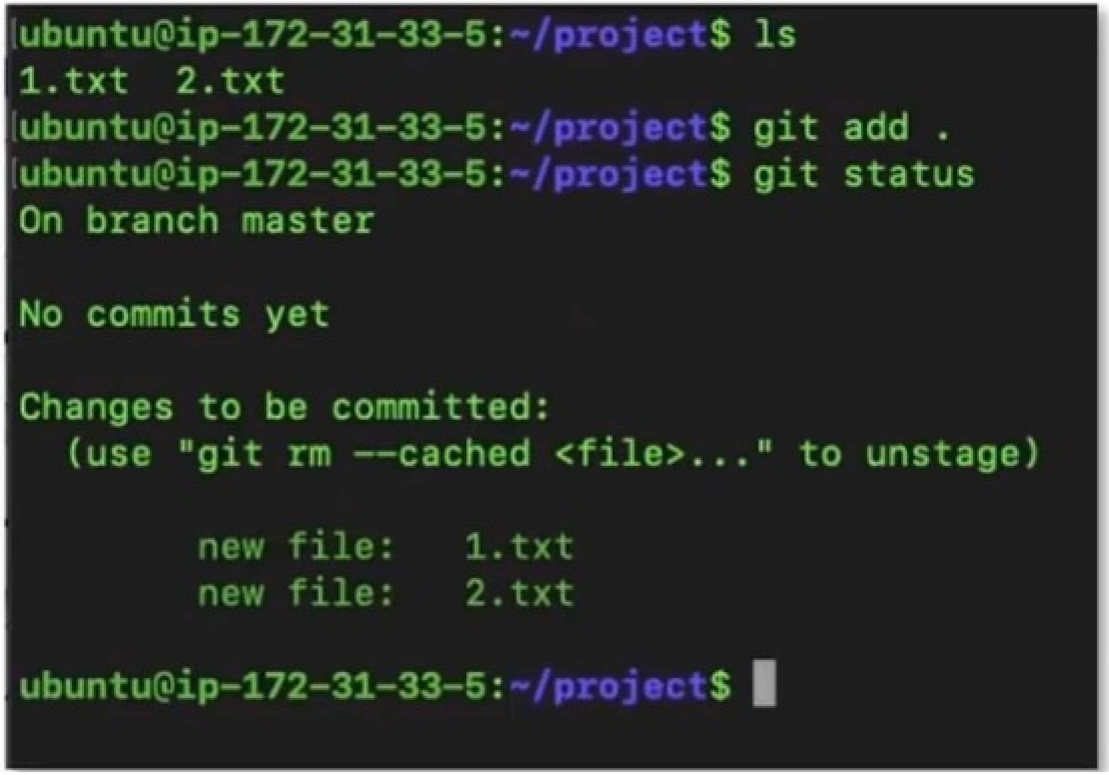


```
ubuntu@ip-172-31-33-5:~/project$ ls
1.txt  2.txt
ubuntu@ip-172-31-33-5:~/project$ git status
On branch master

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$
```

Git add

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,

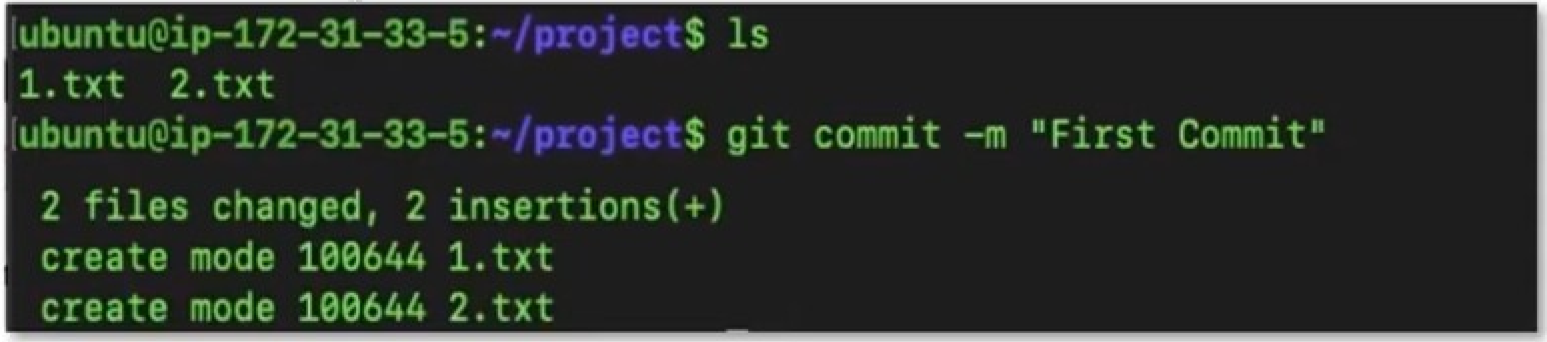
git add .



```
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$ 
```

Git commit

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"**

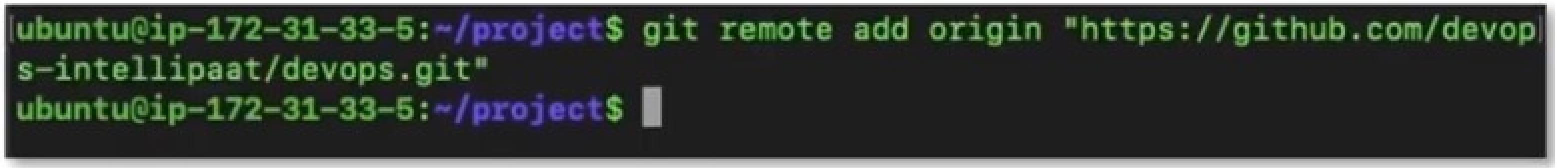


```
ubuntu@ip-172-31-33-5:~/project$ ls
1.txt  2.txt
ubuntu@ip-172-31-33-5:~/project$ git commit -m "First Commit"

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

Git remote

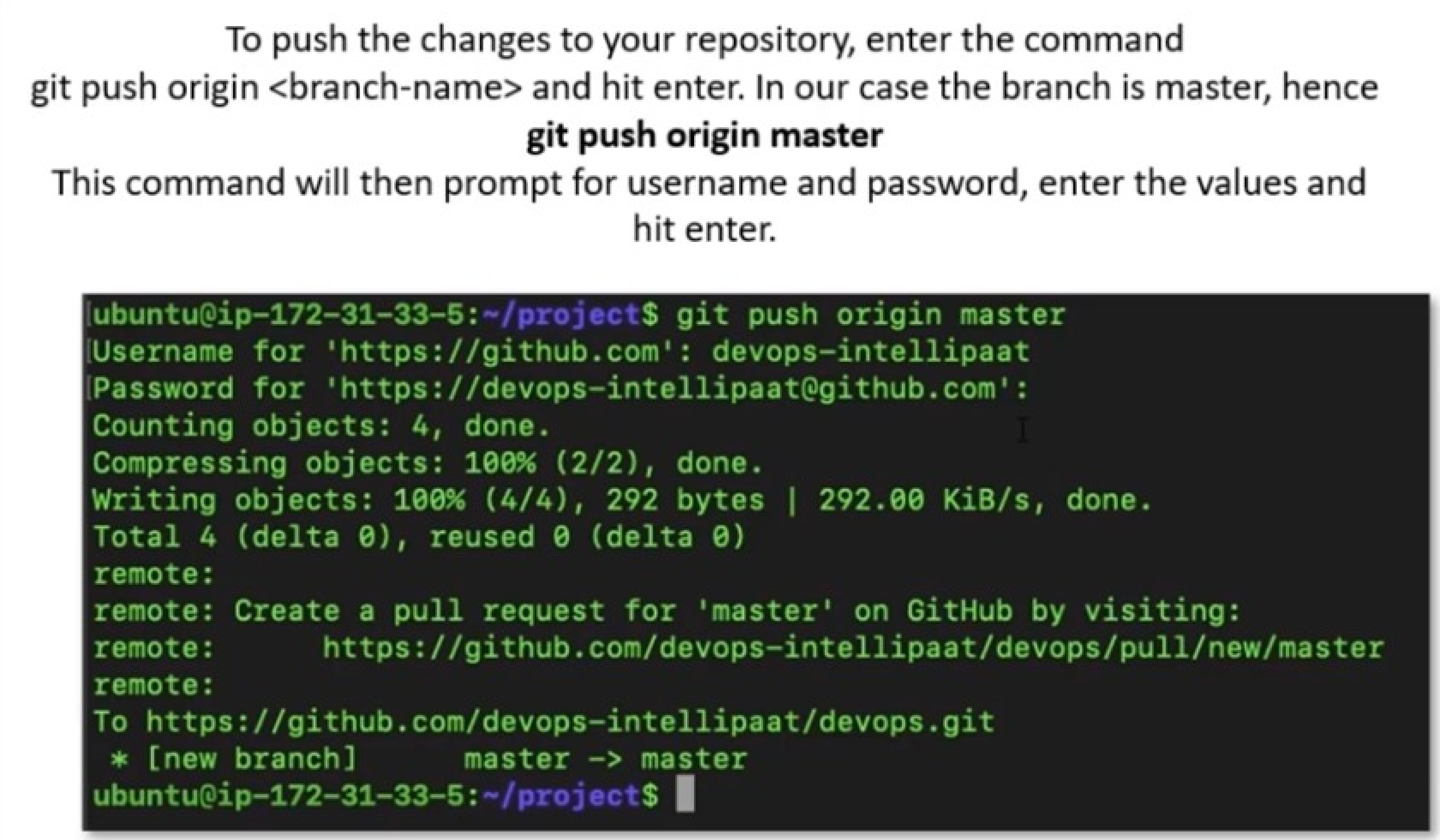
Once everything is ready on our local, we can start pushing our changes to the remote repository. Copy your repository link and paste it in the command

git remote add origin “<URL to repository>”



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

Git push

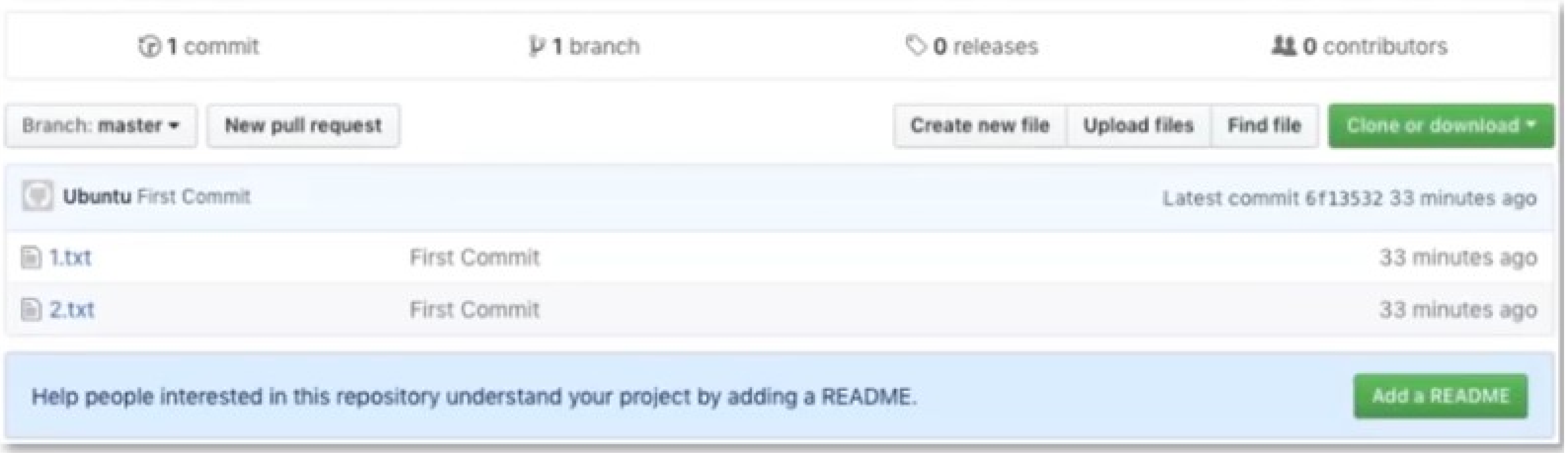


To push the changes to your repository, enter the command
git push origin <branch-name> and hit enter. In our case the branch is master, hence
git push origin master
This command will then prompt for username and password, enter the values and hit enter.

```
ubuntu@ip-172-31-33-5:~/project$ git push origin master
Username for 'https://github.com': devops-intellipaat
Password for 'https://devops-intellipaat@github.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-intellipaat/devops/pull/new/master
remote:
To https://github.com/devops-intellipaat/devops.git
 * [new branch]      master -> master
ubuntu@ip-172-31-33-5:~/project$ 
```

Git push

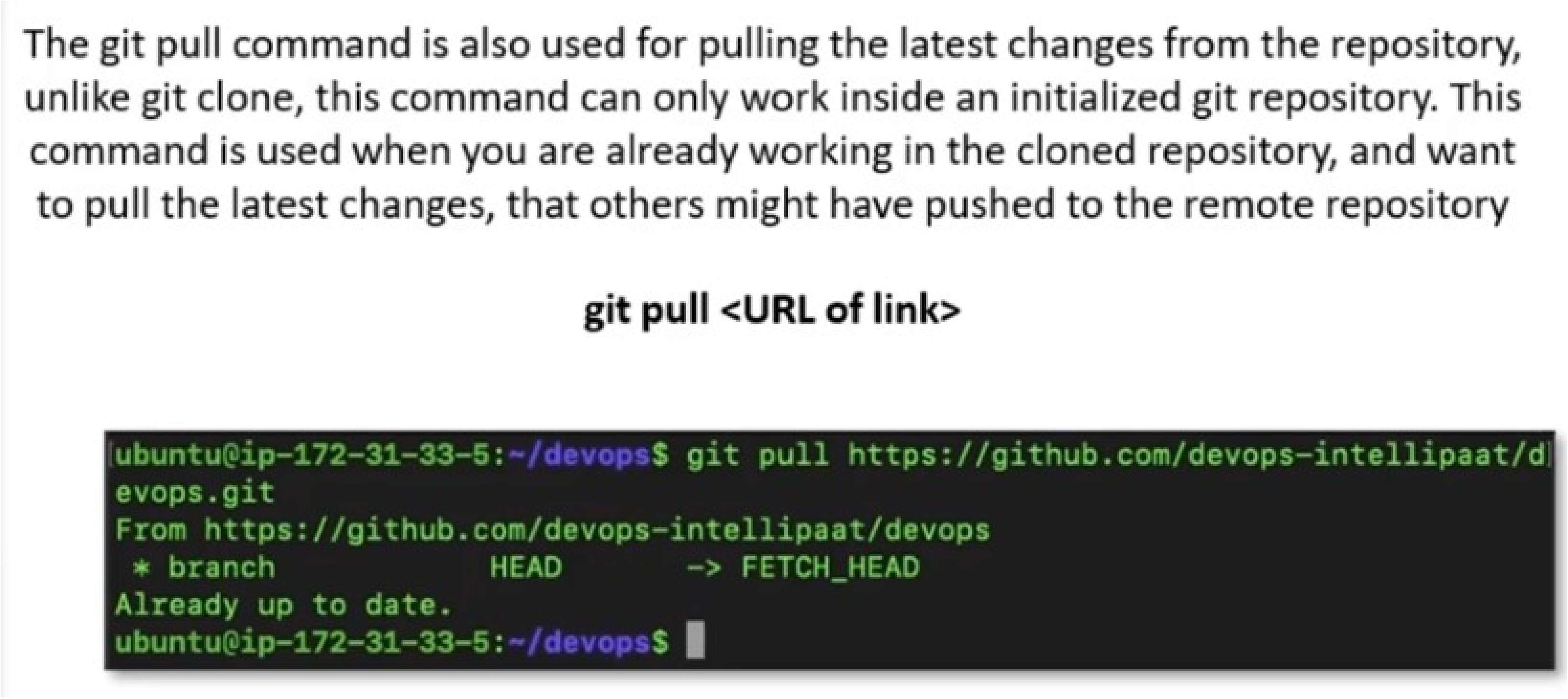
Your local repository is now synced with the remote repository on github



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.

Git pull

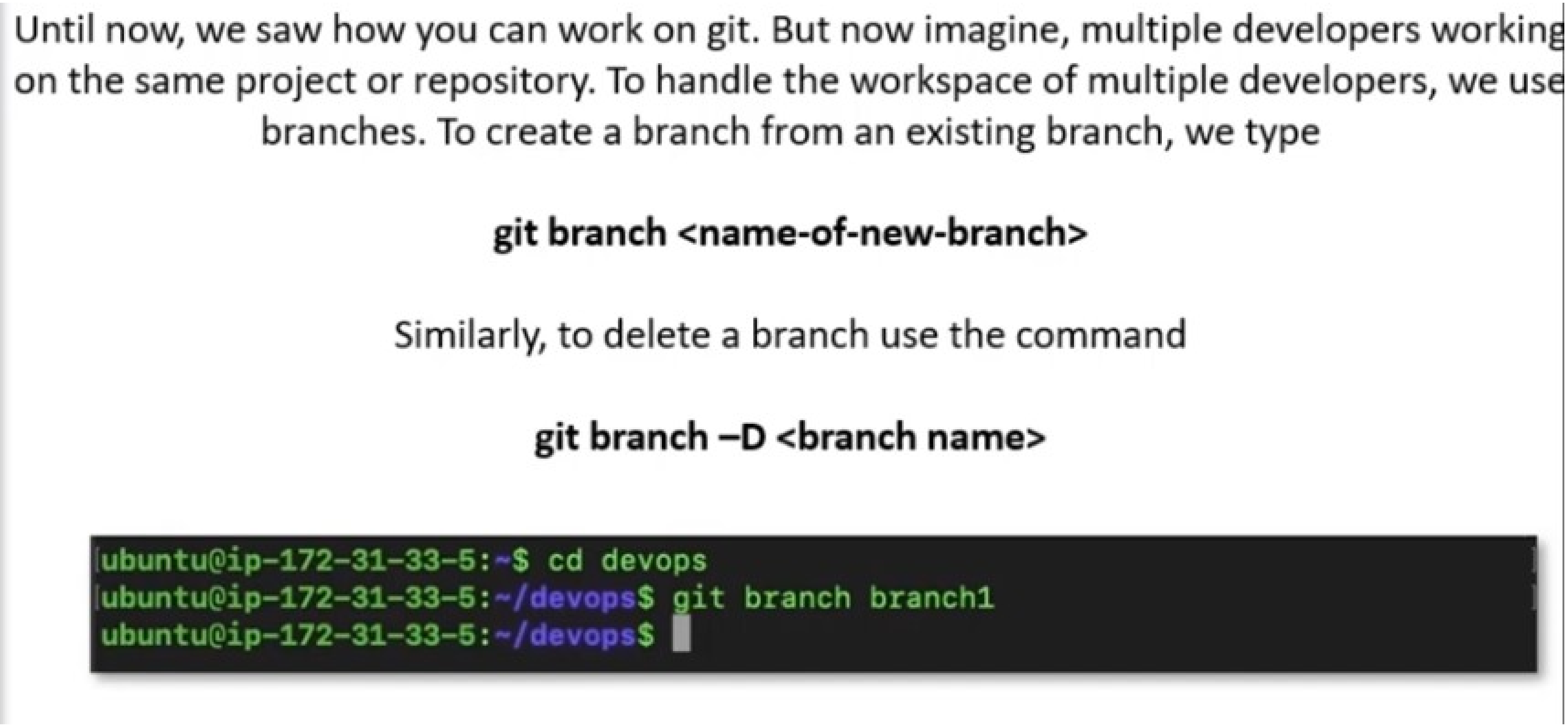


The git pull command is also used for pulling the latest changes from the repository, unlike git clone, this command can only work inside an initialized git repository. This command is used when you are already working in the cloned repository, and want to pull the latest changes, that others might have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

git pull <URL of link>

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

Git branch



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

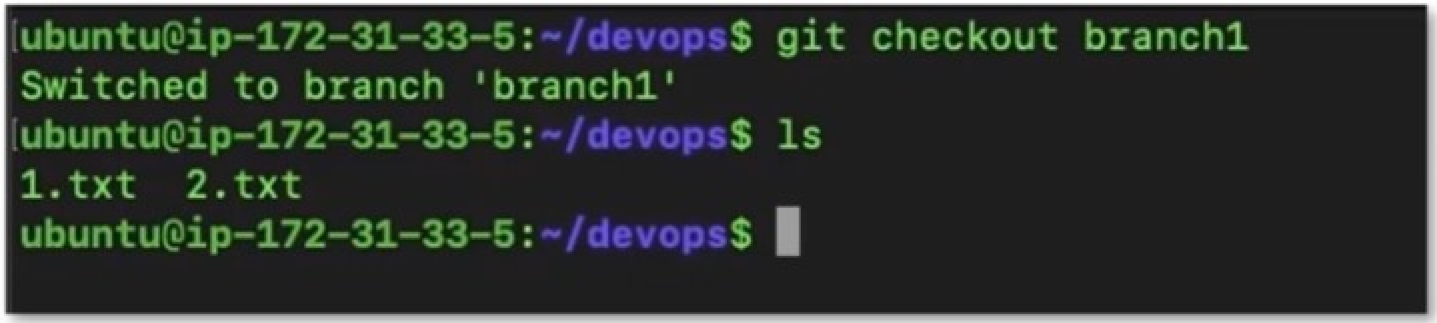
git branch –D <branch name>

```
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$
```

Switch to new branch

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$
```