**Git Fetch vs Git Pull**

Git Fetch is the command that tells the local repository that there are changes available in the remote repository without bringing the changes into the local repository. Git Pull on the other hand brings the copy of the remote directory changes into the local repository. Let us look at Git Fetch and Git Pull separately with the help of an example.

**Git Fetch**

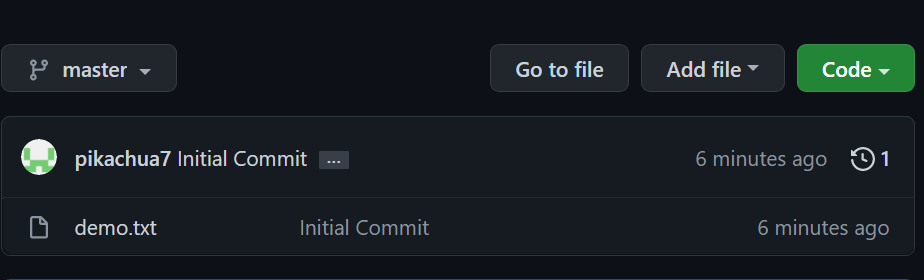
Let us create a file called ***demo.txt***with “***Hello Geeks”*** content inside it initialize the directory to a git repository and push the changes to a remote repository.

* *git init*
* *git add <Filename>*
* *git commit -m <Commit Message>*
* *git remote add origin <Link to your remote repository>*
* *git push origin <branch name>*

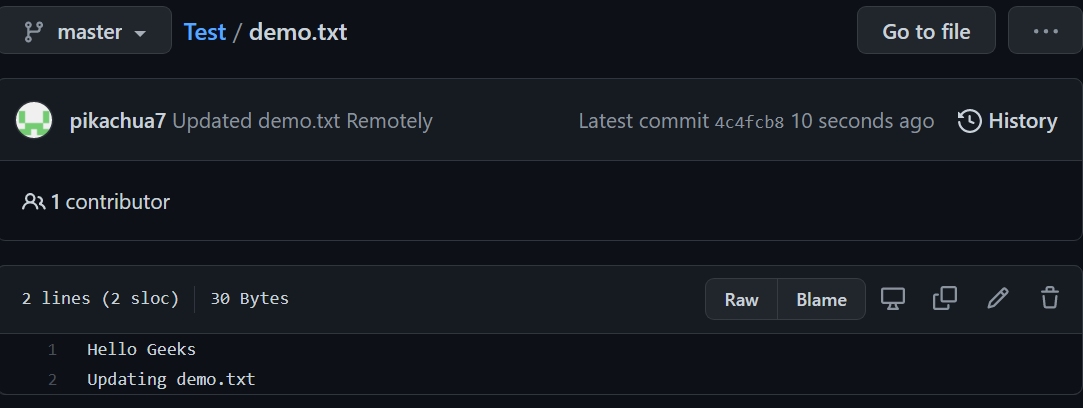
Text

Description automatically generated

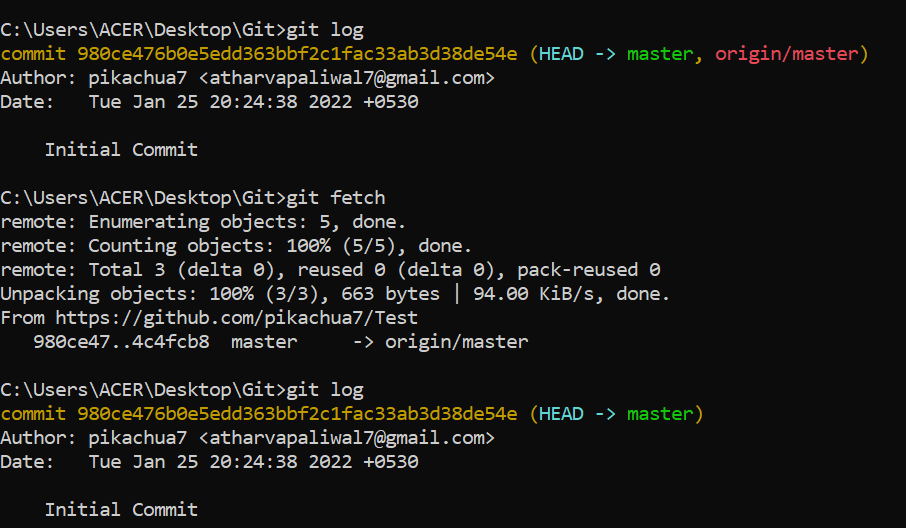
Now, we have my ***demo.txt*** in the remote repository.



The local and the remote repositories are now in sync and have the same content at both places. Let’s now update our ***demo.txt*** in the remote repository.



Now since we have updated our ***demo.txt*** remotely, let’s bring the changes to our local repository. Our local repository has only 1 commit while the remote repository now has 2 commits (*observe the second commit starting from****4c4fcb8***). Let’s use the ***git fetch***command to see in the local repository whether we have a change in the remote repository or not. Before that let’s use the ***git log***command to see our previous commits.

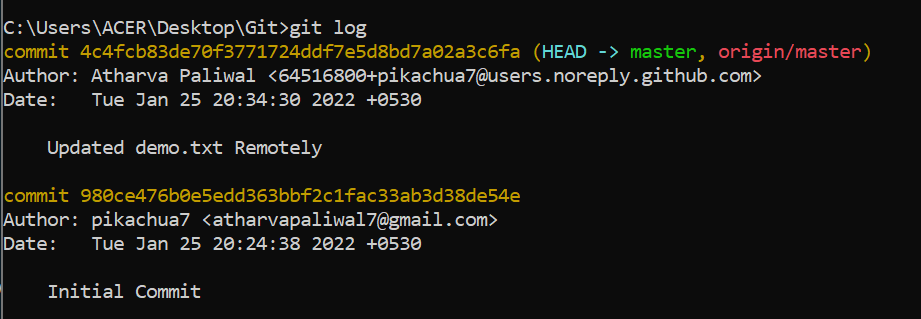


We can see that after using ***git fetch***we get the information that there is some commit done in the remote repository. (*notice the****4c4fcb8****which is the initials of our 2nd commit in a remote repository*). To merge these changes into our local repository, we need to use the ***git merge origin/<branch name>***command.

Text

Description automatically generated

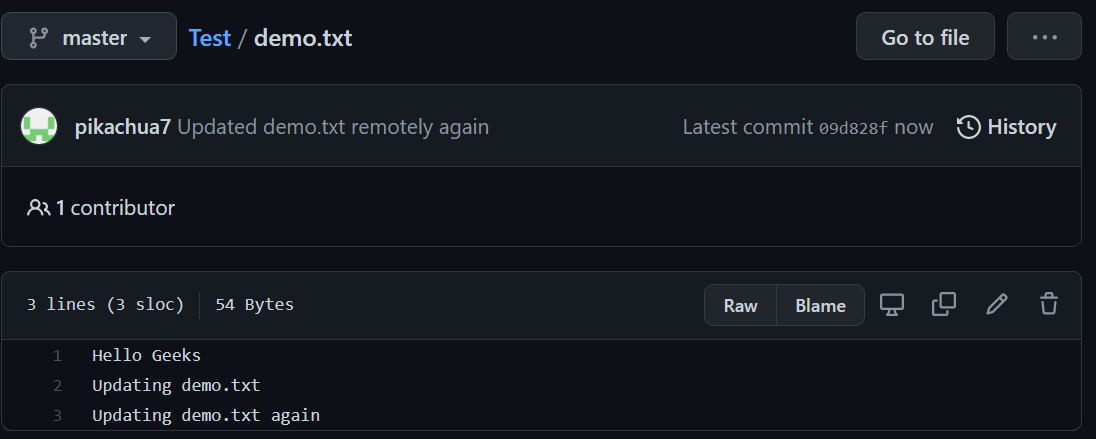
Let us have a look at our commits in the local repository usingthe ***git log***command.



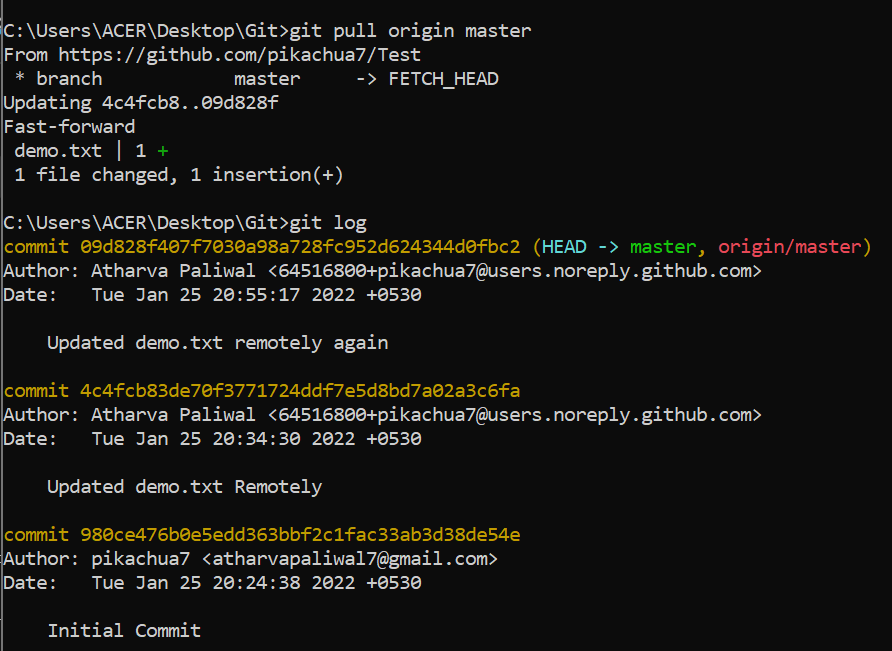
And we got our remote repository commit in our local repository. This is how git fetch works. Let us now have a look at the ***git pull***command.

**Git Pull**

Let’s make more changes to our demo.txt file at the remote repository.



Now, we have 3 commits at our remote repository whereas 2 commits at our local repository. (*Notice the third commit starting with****09d828f***). Let us now bring this change to our local repository using the ***git pull origin <branch name>*** command.



We can see that with the help of just ***git pull*** command we directly fetched and merged our remote repository with the local repository.

***git pull = git fetch + git merge***

Let us see what our demo.txt in the local repository looks like –

Graphical user interface, text, application

Description automatically generated

And now our remote and local repositories are again in sync with each other. So, from the above examples, we can conclude that –

**Difference Table**

| **Git Fetch** | **Git Pull** |
| --- | --- |
| Gives the information of a new change from a remote repository without merging into the current branch | Brings the copy of all the changes from a remote repository and merges them into the current branch |
| Repository data is updated in the .git directory | The local repository is updated directly |
| Review of commits and changes can be done | Updates the changes to the local repository immediately. |
| No possibility of merge conflicts. | Merge conflicts are possible if the remote and the local repositories have done changes at the same place. |
| Command for Git fetch is git fetch<remote> | Command for Git Pull is git pull<remote><branch> |
| Git fetch basically imports the commits to local branches so as to keep up-to-date that what everybody is working on. | Git Pull basically brings the local branch up-to-date with the remote copy that will also updates the other remote tracking branches. |