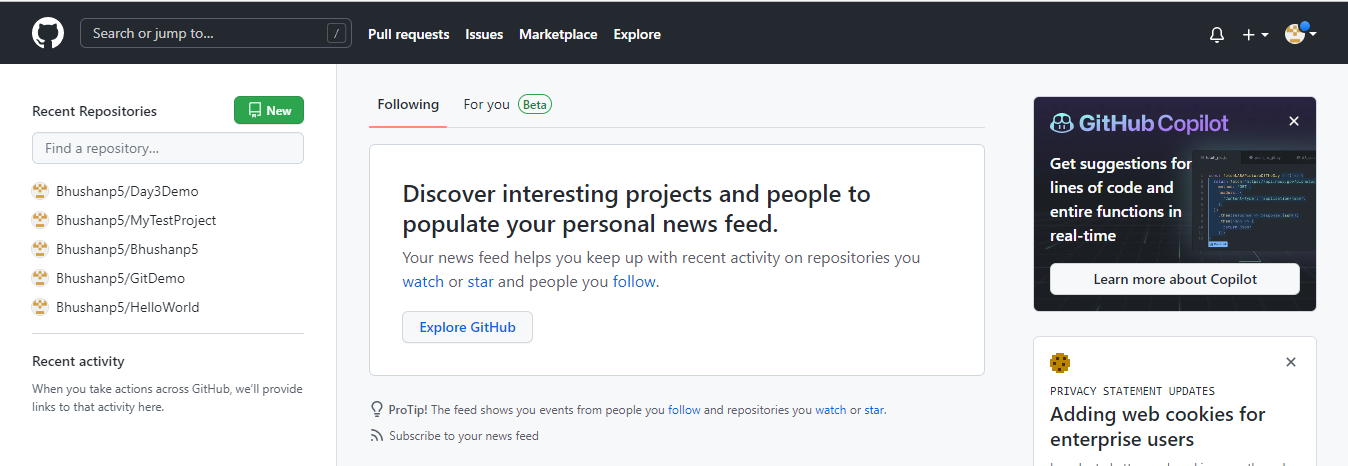
git clone vs git remote add

git clone:

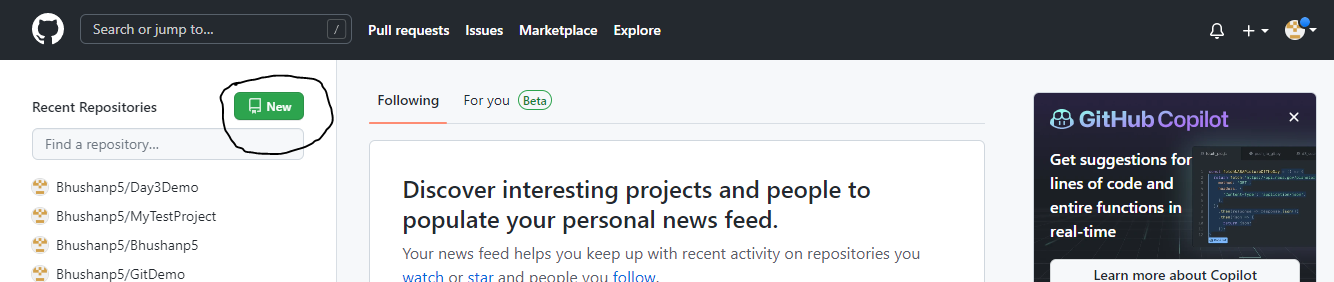
I hope you are aware of clone and remote add commands.

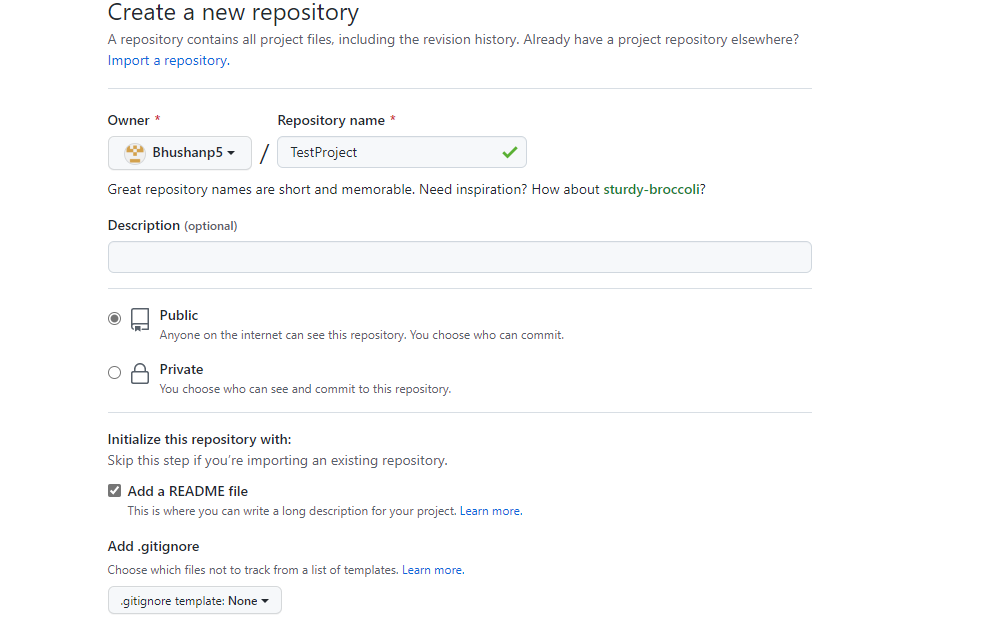
But let’s see what is different between them.

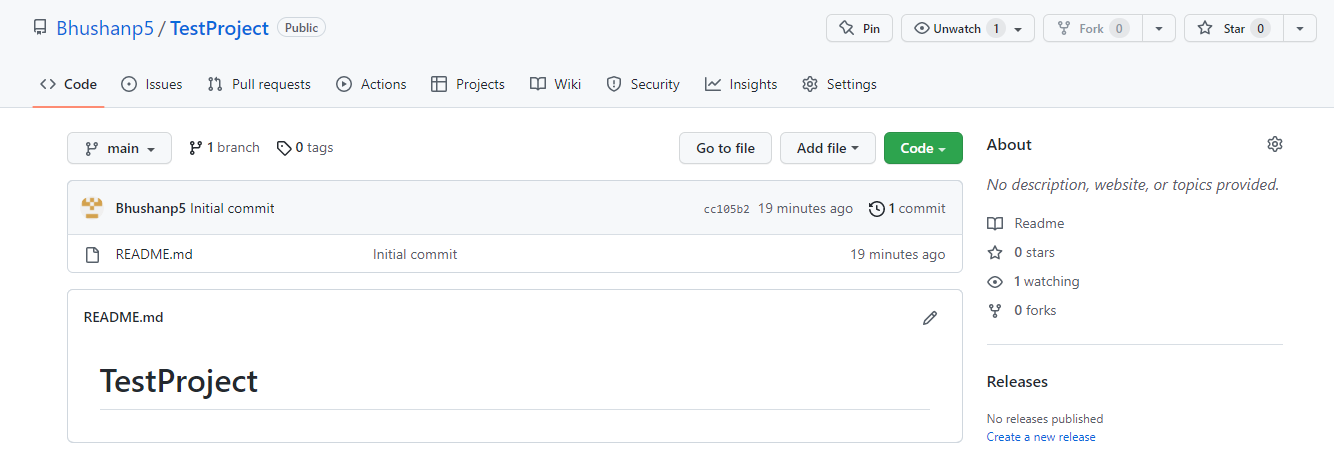
1. Visit and login <https://github.com/>



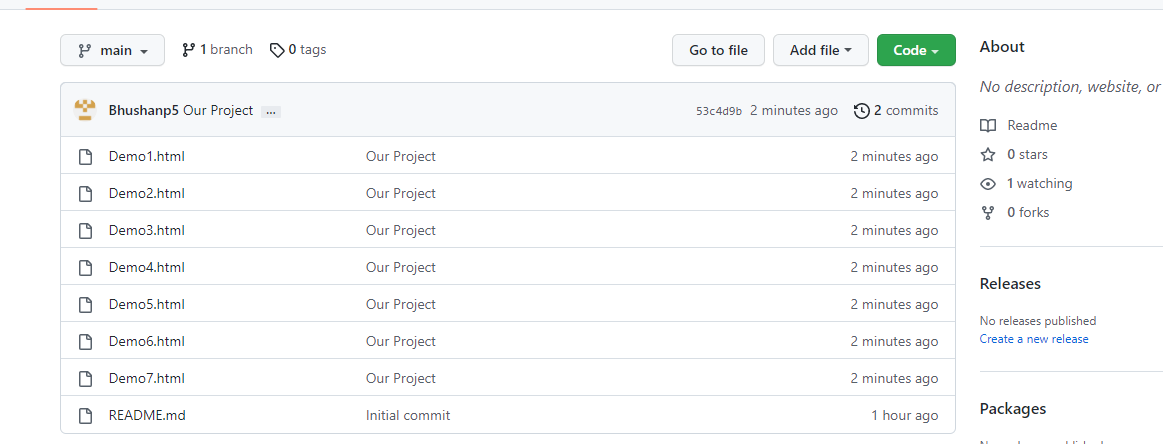
1. Create new repository.







Once it will be done then click on add files and upload files and commit so we can see how to clone from GitHub.



Now, create a new folder for cloning this project.

Right click and open git bash and fire given below commands:

BhushanP5@IMCCBCP48-MSL1 MINGW64 ~/Desktop/MyData/MyDrive/OneDrive - CitiusTech/Freshers Training July 2022/GIT/Day 3/TestProject

$ git config user.name Bhushan

fatal: not in a git directory

BhushanP5@IMCCBCP48-MSL1 MINGW64 ~/Desktop/MyData/MyDrive/OneDrive - CitiusTech/Freshers Training July 2022/GIT/Day 3/TestProject

$ git init

Initialized empty Git repository in C:/Users/Bhushanp5/Desktop/MyData/MyDrive/OneDrive - CitiusTech/Freshers Training July 2022/GIT/Day 3/TestProject/.git/

BhushanP5@IMCCBCP48-MSL1 MINGW64 ~/Desktop/MyData/MyDrive/OneDrive - CitiusTech/Freshers Training July 2022/GIT/Day 3/TestProject (master)

$ git config user.name Bhushanp5

BhushanP5@IMCCBCP48-MSL1 MINGW64 ~/Desktop/MyData/MyDrive/OneDrive - CitiusTech/Freshers Training July 2022/GIT/Day 3/TestProject (master)

$ git config user.email Bhushan.Paradkar@citiustech.com

BhushanP5@IMCCBCP48-MSL1 MINGW64 ~/Desktop/MyData/MyDrive/OneDrive - CitiusTech/Freshers Training July 2022/GIT/Day 3/TestProject (master)

$ git clone https://github.com/Bhushanp5/TestProject.git

Cloning into 'TestProject'...

remote: Enumerating objects: 12, done.

remote: Counting objects: 100% (12/12), done.

remote: Compressing objects: 100% (10/10), done.

remote: Total 12 (delta 1), reused 0 (delta 0), pack-reused 0

Receiving objects: 100% (12/12), 4.39 KiB | 4.39 MiB/s, done.

Resolving deltas: 100% (1/1), done.

BhushanP5@IMCCBCP48-MSL1 MINGW64 ~/Desktop/MyData/MyDrive/OneDrive - CitiusTech/Freshers Training July 2022/GIT/Day 3/TestProject (master)

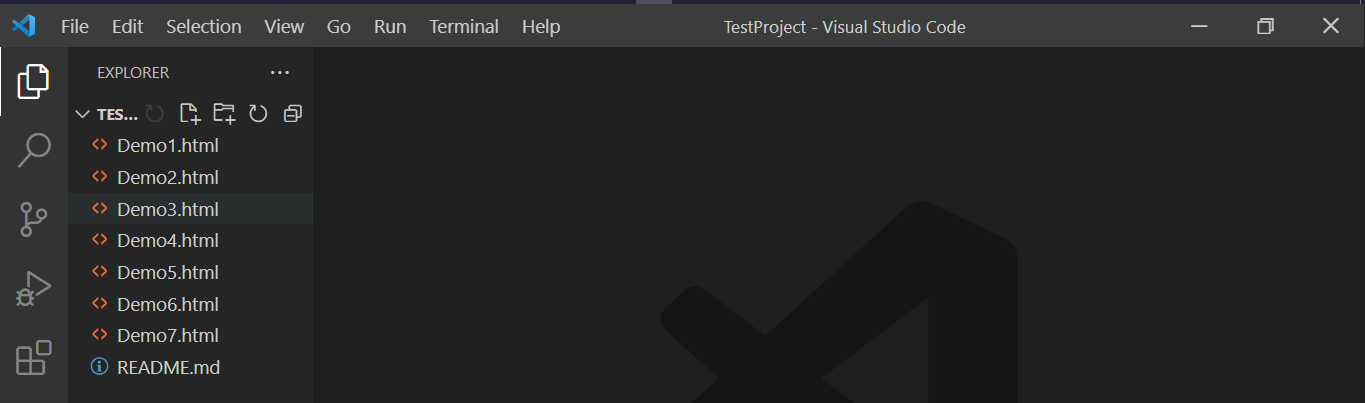
$ cd TestProject

BhushanP5@IMCCBCP48-MSL1 MINGW64 ~/Desktop/MyData/MyDrive/OneDrive - CitiusTech/Freshers Training July 2022/GIT/Day 3/TestProject/TestProject (main)

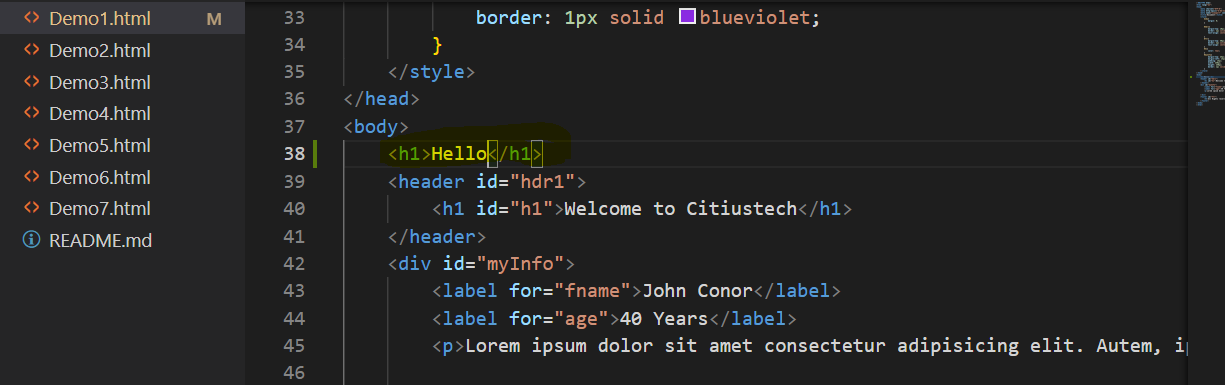
$ ls

Demo1.html Demo2.html Demo3.html Demo4.html Demo5.html Demo6.html Demo7.html README.md

Now, Next step is open TestProject folder and open it in VS Code.

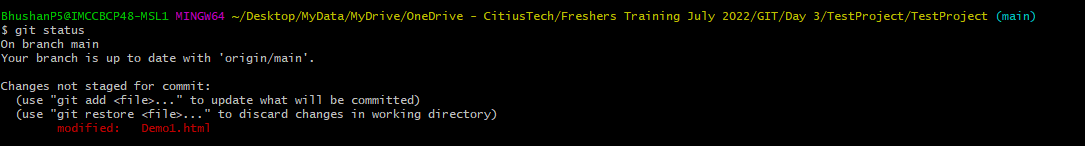


Open any html file and do some changes.



I did the change which I marked.

Now, next step is open our git bash and check status.



Follow given below steps which you already tried.

BhushanP5@IMCCBCP48-MSL1 MINGW64 ~/Desktop/MyData/MyDrive/OneDrive - CitiusTech/Freshers Training July 2022/GIT/Day 3/TestProject/TestProject (main)

$ git add --all

BhushanP5@IMCCBCP48-MSL1 MINGW64 ~/Desktop/MyData/MyDrive/OneDrive - CitiusTech/Freshers Training July 2022/GIT/Day 3/TestProject/TestProject (main)

$ git status

On branch main

Your branch is up to date with 'origin/main'.

Changes to be committed:

(use "git restore --staged <file>..." to unstage)

modified: Demo1.html

BhushanP5@IMCCBCP48-MSL1 MINGW64 ~/Desktop/MyData/MyDrive/OneDrive - CitiusTech/Freshers Training July 2022/GIT/Day 3/TestProject/TestProject (main)

$ git commit -m "Changes done on 31st"

[main 605a8d4] Changes done on 31st

1 file changed, 1 insertion(+)

BhushanP5@IMCCBCP48-MSL1 MINGW64 ~/Desktop/MyData/MyDrive/OneDrive - CitiusTech/Freshers Training July 2022/GIT/Day 3/TestProject/TestProject (main)

$ git push origin master

error: src refspec master does not match any

error: failed to push some refs to 'https://github.com/Bhushanp5/TestProject.git'

BhushanP5@IMCCBCP48-MSL1 MINGW64 ~/Desktop/MyData/MyDrive/OneDrive - CitiusTech/Freshers Training July 2022/GIT/Day 3/TestProject/TestProject (main)

$ git remote add origin https://github.com/Bhushanp5/TestProject.git

error: remote origin already exists.

BhushanP5@IMCCBCP48-MSL1 MINGW64 ~/Desktop/MyData/MyDrive/OneDrive - CitiusTech/Freshers Training July 2022/GIT/Day 3/TestProject/TestProject (main)

$ git branch -M main

BhushanP5@IMCCBCP48-MSL1 MINGW64 ~/Desktop/MyData/MyDrive/OneDrive - CitiusTech/Freshers Training July 2022/GIT/Day 3/TestProject/TestProject (main)

$ git push -u origin main

Enumerating objects: 5, done.

Counting objects: 100% (5/5), done.

Delta compression using up to 8 threads

Compressing objects: 100% (3/3), done.

Writing objects: 100% (3/3), 330 bytes | 330.00 KiB/s, done.

Total 3 (delta 2), reused 0 (delta 0), pack-reused 0

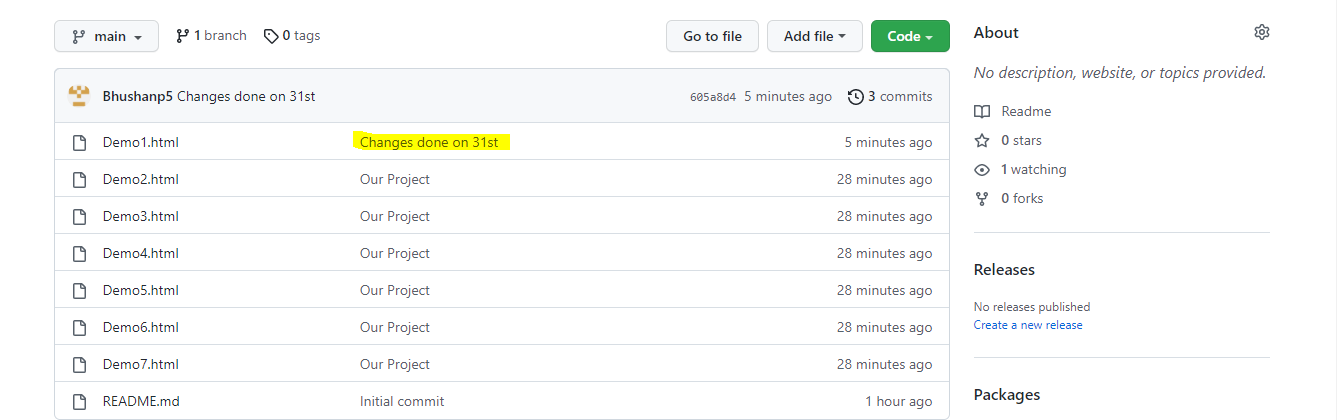
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.

To https://github.com/Bhushanp5/TestProject.git

53c4d9b..605a8d4 main -> main

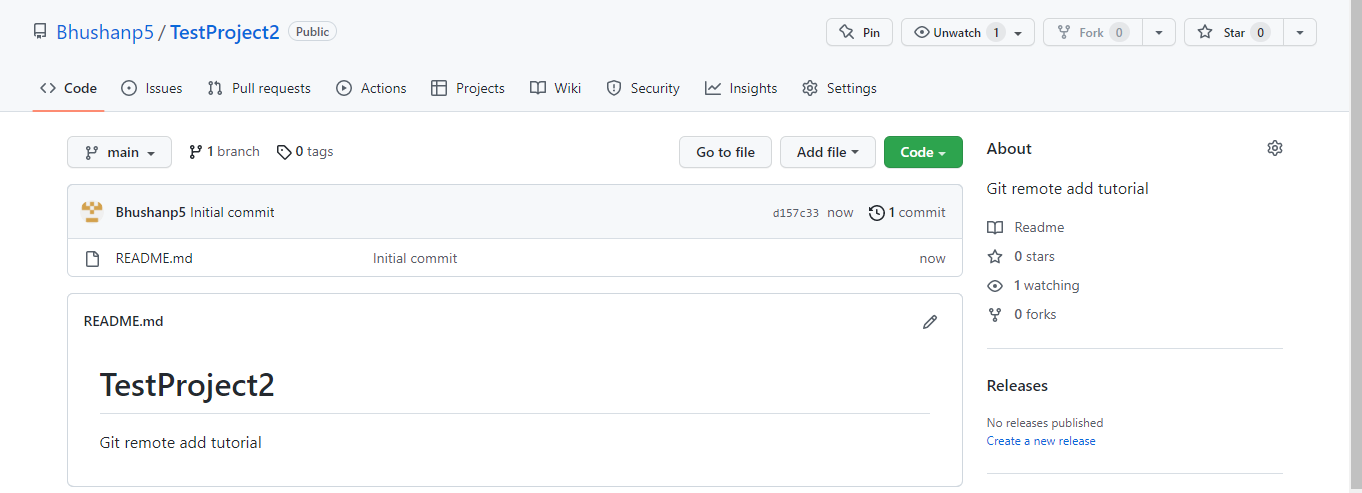
Branch 'main' set up to track remote branch 'main' from 'origin'.

Now, we can see changes are updated in remote repo too.

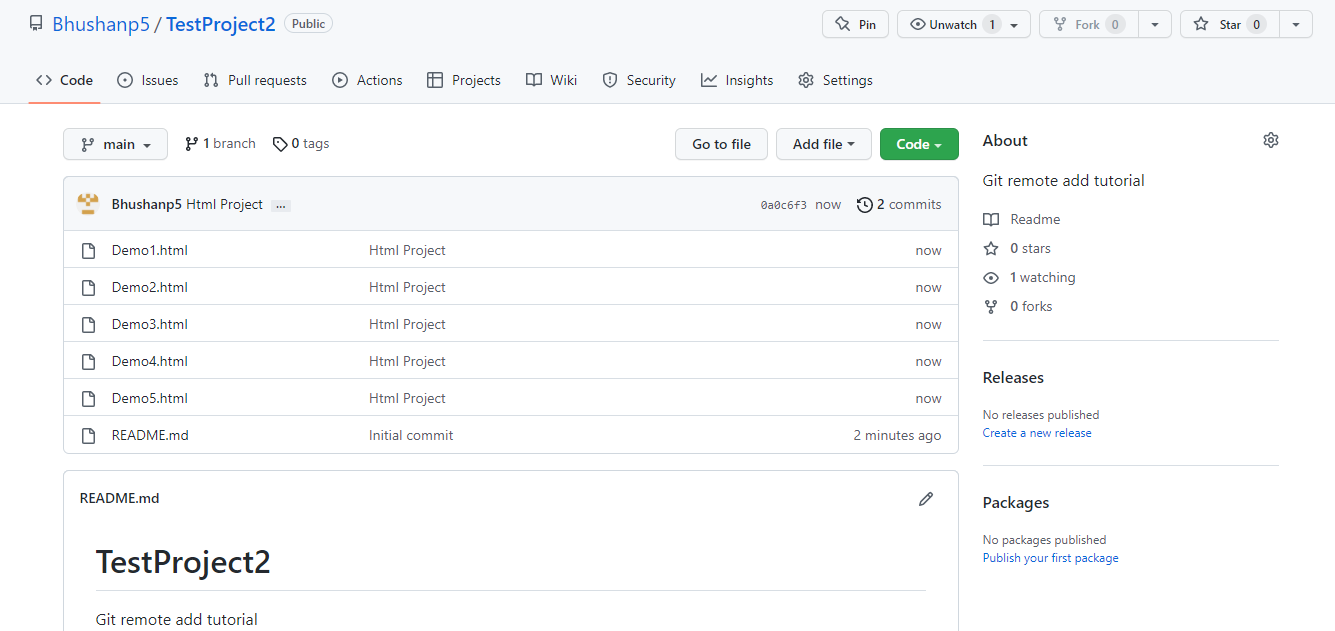


**git remote add:**

For git remote add we will create another repository. Open GitHub and create new repo then add some file in it.



Add files like previous project.



Now again create new folder, follow git initial commands which we did for previous demo.

1. Create folder
2. Open git bash

BhushanP5@IMCCBCP48-MSL1 MINGW64 ~/Desktop/MyData/MyDrive/OneDrive - CitiusTech/Freshers Training July 2022/GIT/Day 3/TestProject/TestProject2 (master)

$ git init

Initialized empty Git repository in C:/Users/Bhushanp5/Desktop/MyData/MyDrive/OneDrive - CitiusTech/Freshers Training July 2022/GIT/Day 3/TestProject/TestProject2/.git/

BhushanP5@IMCCBCP48-MSL1 MINGW64 ~/Desktop/MyData/MyDrive/OneDrive - CitiusTech/Freshers Training July 2022/GIT/Day 3/TestProject/TestProject2 (master)

$ git remote add origin https://github.com/Bhushanp5/TestProject2.git

Now, trying to pull master branch but can’t get it and reason for this is there is no master branch present so, we are getting this error.

BhushanP5@IMCCBCP48-MSL1 MINGW64 ~/Desktop/MyData/MyDrive/OneDrive - CitiusTech/Freshers Training July 2022/GIT/Day 3/TestProject/TestProject2 (master)

$ git pull origin master

fatal: couldn't find remote ref master

Either you can search for correct branch name or you can create new branch for you and use this branch.

BhushanP5@IMCCBCP48-MSL1 MINGW64 ~/Desktop/MyData/MyDrive/OneDrive - CitiusTech/Freshers Training July 2022/GIT/Day 3/TestProject/TestProject2 (master)

$ git pull origin dev

remote: Enumerating objects: 10, done.

remote: Counting objects: 100% (10/10), done.

remote: Compressing objects: 100% (8/8), done.

remote: Total 10 (delta 1), reused 0 (delta 0), pack-reused 0

Unpacking objects: 100% (10/10), 3.45 KiB | 7.00 KiB/s, done.

From https://github.com/Bhushanp5/TestProject2

\* branch dev -> FETCH\_HEAD

\* [new branch] dev -> origin/dev

BhushanP5@IMCCBCP48-MSL1 MINGW64 ~/Desktop/MyData/MyDrive/OneDrive - CitiusTech/Freshers Training July 2022/GIT/Day 3/TestProject/TestProject2 (master)

$ ls

Demo1.html Demo2.html Demo3.html Demo4.html Demo5.html README.md

BhushanP5@IMCCBCP48-MSL1 MINGW64 ~/Desktop/MyData/MyDrive/OneDrive - CitiusTech/Freshers Training July 2022/GIT/Day 3/TestProject/TestProject2 (master)

$ git status

On branch master

nothing to commit, working tree clean

But there are some technical differences between git clone and git remote add.

**GIT REMOTE add** just creates an entry in your git configuring  that specifies a name for a particular URL. You must have an existing git report  to use this.

**GIT CLONE** creates a new git repository by copying an existing one located at the URL you specify.

**The differences between git clone and git remote:**

**git clone:**

Will physically download the files into your computer. It will take space from your computer. If the repo is 200Mb, then it will download that all and place it in the directory you cloned.

**git remote add:**

Won't take space! It's more like a pointer! It doesn't increase your disk consumption. It just gets a snapshot of what branches are available and their git commit history I believe. It doesn't contain the actual file/folders of your project.

If you do:

git remote add TechLeadRepo git://github.com/user/test.git

then you haven't added anything to your computer. After you've added it in your remote branches then you're able to get a list of all branches on that remote by doing:

BhushanP5@IMCCBCP48-MSL1 MINGW64 ~/Desktop/MyData/MyDrive/OneDrive - CitiusTech/Freshers Training July 2022/GIT/Day 3/TestProject/TestProject2 (master)

$ git pull origin dev

remote: Enumerating objects: 10, done.

remote: Counting objects: 100% (10/10), done.

remote: Compressing objects: 100% (8/8), done.

remote: Total 10 (delta 1), reused 0 (delta 0), pack-reused 0

Unpacking objects: 100% (10/10), 3.45 KiB | 7.00 KiB/s, done.

From https://github.com/Bhushanp5/TestProject2

\* branch dev -> FETCH\_HEAD

\* [new branch] dev -> origin/dev

We can also use command to fetch all branches that is git fetch –all.

Using the -amend option with the git commit command.

--amend is use for modifying last commit.

Some important point about amend:

1. It is used to modify last commit
2. It actually deletes last commit but keep changes of last commit but create new commit with old changes. We will check it out with example.

For this amend we will create new folder and create local git repo.

I hope you know how to create git repo.

BhushanP5@IMCCBCP48-MSL1 MINGW64 ~/Desktop/MyData/MyDrive/OneDrive - CitiusTech/Freshers Training July 2022/GIT/Day 3/TestProject/ammendDemo (master)

$ git init

Initialized empty Git repository in C:/Users/Bhushanp5/Desktop/MyData/MyDrive/OneDrive - CitiusTech/Freshers Training July 2022/GIT/Day 3/TestProject/ammendDemo/.git/

BhushanP5@IMCCBCP48-MSL1 MINGW64 ~/Desktop/MyData/MyDrive/OneDrive - CitiusTech/Freshers Training July 2022/GIT/Day 3/TestProject/ammendDemo (master)

$ git config --global --edit

BhushanP5@IMCCBCP48-MSL1 MINGW64 ~/Desktop/MyData/MyDrive/OneDrive - CitiusTech/Freshers Training July 2022/GIT/Day 3/TestProject/ammendDemo (master)

$ touch MyInfo.txt

BhushanP5@IMCCBCP48-MSL1 MINGW64 ~/Desktop/MyData/MyDrive/OneDrive - CitiusTech/Freshers Training July 2022/GIT/Day 3/TestProject/ammendDemo (master)

$ git status

On branch master

No commits yet

Untracked files:

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

MyInfo.txt

nothing added to commit but untracked files present (use "git add" to track)

BhushanP5@IMCCBCP48-MSL1 MINGW64 ~/Desktop/MyData/MyDrive/OneDrive - CitiusTech/Freshers Training July 2022/GIT/Day 3/TestProject/ammendDemo (master)

$ git add .

BhushanP5@IMCCBCP48-MSL1 MINGW64 ~/Desktop/MyData/MyDrive/OneDrive - CitiusTech/Freshers Training July 2022/GIT/Day 3/TestProject/ammendDemo (master)

$ git commit -m "MyInfo Added"

[master (root-commit) 31bbd37] MyInfo Added

1 file changed, 0 insertions(+), 0 deletions(-)

create mode 100644 MyInfo.txt

BhushanP5@IMCCBCP48-MSL1 MINGW64 ~/Desktop/MyData/MyDrive/OneDrive - CitiusTech/Freshers Training July 2022/GIT/Day 3/TestProject/ammendDemo (master)

$ git status

On branch master

nothing to commit, working tree clean

BhushanP5@IMCCBCP48-MSL1 MINGW64 ~/Desktop/MyData/MyDrive/OneDrive - CitiusTech/Freshers Training July 2022/GIT/Day 3/TestProject/ammendDemo (master)

$ git log

commit 31bbd37940e10f9c04c0f19bb5296b3988ae0b9d (HEAD -> master)

Author: Bhushanp5 <bhushan.paradkar@citiustech.com>

Date: Thu Sep 1 00:16:06 2022 +0530

MyInfo Added

But now I want to change commit message then for this purpose I can use --amend

BhushanP5@IMCCBCP48-MSL1 MINGW64 ~/Desktop/MyData/MyDrive/OneDrive - CitiusTech/Freshers Training July 2022/GIT/Day 3/TestProject/ammendDemo (master)

$ git commit --amend -m "MyInfo Added on 31st"

[master 78ec23d] MyInfo Added on 31st

Date: Thu Sep 1 00:16:06 2022 +0530

1 file changed, 0 insertions(+), 0 deletions(-)

create mode 100644 MyInfo.txt

BhushanP5@IMCCBCP48-MSL1 MINGW64 ~/Desktop/MyData/MyDrive/OneDrive - CitiusTech/Freshers Training July 2022/GIT/Day 3/TestProject/ammendDemo (master)

$ git log

commit 78ec23dd98cab61f7b623f0ee3e2342da46330fc (HEAD -> master)

Author: Bhushanp5 <bhushan.paradkar@citiustech.com>

Date: Thu Sep 1 00:16:06 2022 +0530

MyInfo Added on 31st

But observe one thing here. After git log I got following output. Note down Hashcode of this commit and now again we will do amend

BhushanP5@IMCCBCP48-MSL1 MINGW64 ~/Desktop/MyData/MyDrive/OneDrive - CitiusTech/Freshers Training July 2022/GIT/Day 3/TestProject/ammendDemo (master)

$ git commit --amend -m "MyInfo Added on 31st at 12 am"

[master 1ceb884] MyInfo Added on 31st at 12 am

Date: Thu Sep 1 00:16:06 2022 +0530

1 file changed, 0 insertions(+), 0 deletions(-)

create mode 100644 MyInfo.txt

BhushanP5@IMCCBCP48-MSL1 MINGW64 ~/Desktop/MyData/MyDrive/OneDrive - CitiusTech/Freshers Training July 2022/GIT/Day 3/TestProject/ammendDemo (master)

$ git log

commit 1ceb884af65e68d91c05e0f6d3688598baa3dd9f (HEAD -> master)

Author: Bhushanp5 <bhushan.paradkar@citiustech.com>

Date: Thu Sep 1 00:16:06 2022 +0530

MyInfo Added on 31st at 12 am

So, now you can see here that hash code got changed it means it’s simply deleting previous commit and creating new.

Using amend we can also change name and email too.

Observe given below example.

BhushanP5@IMCCBCP48-MSL1 MINGW64 ~/Desktop/MyData/MyDrive/OneDrive - CitiusTech/Freshers Training July 2022/GIT/Day 3/TestProject/ammendDemo (master)

$ git commit --amend --author="Suyog <Suyog@Citiustech.com>"

[master 21e87a8] MyInfo Added on 31st at 12 am

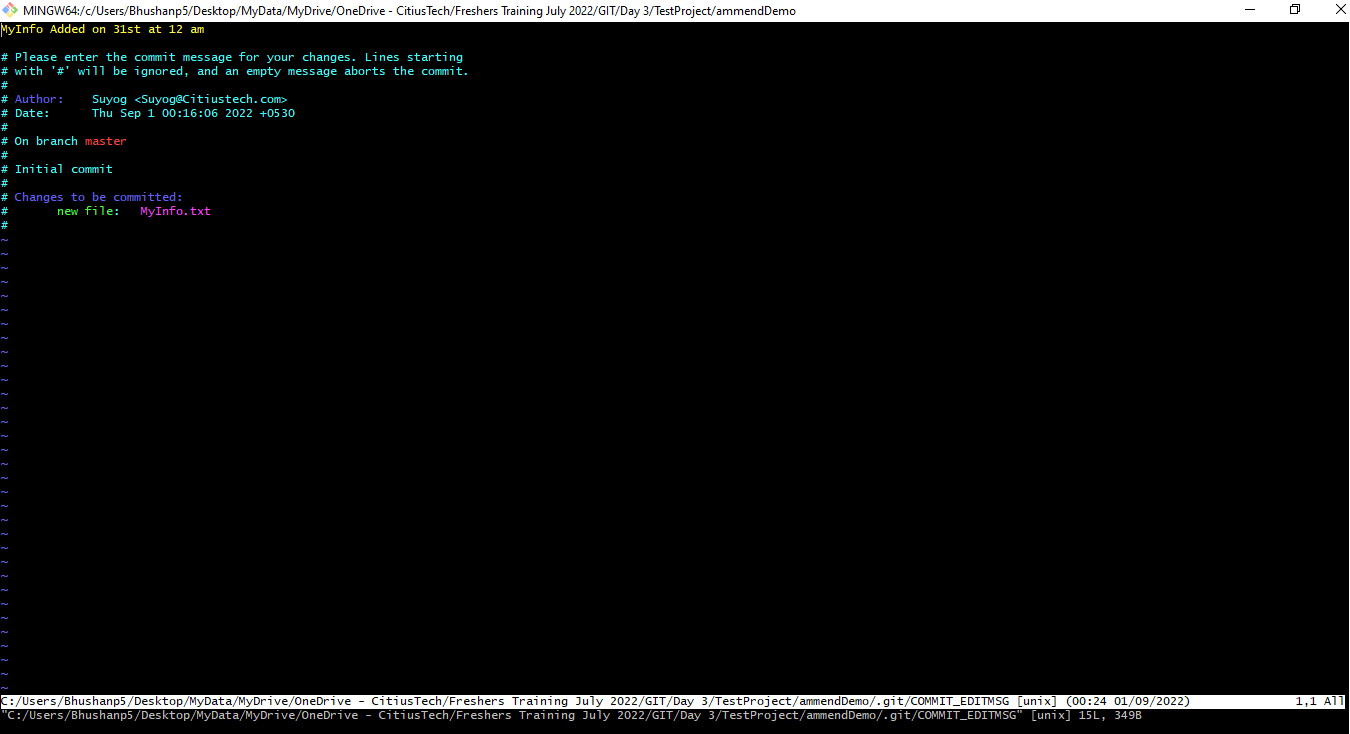
Author: Suyog <Suyog@Citiustech.com>

Date: Thu Sep 1 00:16:06 2022 +0530

1 file changed, 0 insertions(+), 0 deletions(-)

create mode 100644 MyInfo.txt

It will open VM editor to make confirm as given below.



Press :wq thenpress enter

BhushanP5@IMCCBCP48-MSL1 MINGW64 ~/Desktop/MyData/MyDrive/OneDrive - CitiusTech/Freshers Training July 2022/GIT/Day 3/TestProject/ammendDemo (master)

$ git commit --amend --author="Suyog <Suyog@Citiustech.com>"

[master 21e87a8] MyInfo Added on 31st at 12 am

Author: Suyog <Suyog@Citiustech.com>

Date: Thu Sep 1 00:16:06 2022 +0530

1 file changed, 0 insertions(+), 0 deletions(-)

create mode 100644 MyInfo.txt

You will be here.

BhushanP5@IMCCBCP48-MSL1 MINGW64 ~/Desktop/MyData/MyDrive/OneDrive - CitiusTech/Freshers Training July 2022/GIT/Day 3/TestProject/ammendDemo (master)

$ git log

commit 21e87a8f2584bd7be4c1aae7d6fc03afbc893303 (HEAD -> master)

Author: Suyog <Suyog@Citiustech.com>

Date: Thu Sep 1 00:16:06 2022 +0530

MyInfo Added on 31st at 12 am

Here you can see, we have changed name and email of Author too using –amend.

In this way amend is going to work.

Cherry-picking.

It is not part of TOC but as additional knowledge you should have idea about it.

**What is cherry picking?**

**Ans:** Cherry picking in gits means to choose a commit from one branch and apply it to other. Sometimes this conflict with concept of merge and rebase.

**Note:** Whenever you want to apply a commit on any branch then make sure you are on that branch.

We will carry forward with same example.

First, we will create branch “development” in same git bash of amend.

BhushanP5@IMCCBCP48-MSL1 MINGW64 ~/Desktop/MyData/MyDrive/OneDrive - CitiusTech/Freshers Training July 2022/GIT/Day 3/TestProject/ammendDemo (master)

$ git branch development

BhushanP5@IMCCBCP48-MSL1 MINGW64 ~/Desktop/MyData/MyDrive/OneDrive - CitiusTech/Freshers Training July 2022/GIT/Day 3/TestProject/ammendDemo (master)

$ git branch

development

\* master

BhushanP5@IMCCBCP48-MSL1 MINGW64 ~/Desktop/MyData/MyDrive/OneDrive - CitiusTech/Freshers Training July 2022/GIT/Day 3/TestProject/ammendDemo (master)

$ git checkout development

Switched to branch 'development'

Checkout to development branch and now add two text files separately with separate commit in development branch.

BhushanP5@IMCCBCP48-MSL1 MINGW64 ~/Desktop/MyData/MyDrive/OneDrive - CitiusTech/Freshers Training July 2022/GIT/Day 3/TestProject/ammendDemo (development)

$ touch text1.txt

BhushanP5@IMCCBCP48-MSL1 MINGW64 ~/Desktop/MyData/MyDrive/OneDrive - CitiusTech/Freshers Training July 2022/GIT/Day 3/TestProject/ammendDemo (development)

$ git add .

BhushanP5@IMCCBCP48-MSL1 MINGW64 ~/Desktop/MyData/MyDrive/OneDrive - CitiusTech/Freshers Training July 2022/GIT/Day 3/TestProject/ammendDemo (development)

$ commit -m "text1.txt added"

bash: commit: command not found

BhushanP5@IMCCBCP48-MSL1 MINGW64 ~/Desktop/MyData/MyDrive/OneDrive - CitiusTech/Freshers Training July 2022/GIT/Day 3/TestProject/ammendDemo (development)

$ git commit -m "text1.txt added"

[development 9c8fb99] text1.txt added

1 file changed, 0 insertions(+), 0 deletions(-)

create mode 100644 text1.txt

BhushanP5@IMCCBCP48-MSL1 MINGW64 ~/Desktop/MyData/MyDrive/OneDrive - CitiusTech/Freshers Training July 2022/GIT/Day 3/TestProject/ammendDemo (development)

$ touch text2.txt

BhushanP5@IMCCBCP48-MSL1 MINGW64 ~/Desktop/MyData/MyDrive/OneDrive - CitiusTech/Freshers Training July 2022/GIT/Day 3/TestProject/ammendDemo (development)

$ git add .

BhushanP5@IMCCBCP48-MSL1 MINGW64 ~/Desktop/MyData/MyDrive/OneDrive - CitiusTech/Freshers Training July 2022/GIT/Day 3/TestProject/ammendDemo (development)

$ git commit -m "text2.txt added"

[development f4cef3f] text2.txt added

1 file changed, 0 insertions(+), 0 deletions(-)

create mode 100644 text2.txt

BhushanP5@IMCCBCP48-MSL1 MINGW64 ~/Desktop/MyData/MyDrive/OneDrive - CitiusTech/Freshers Training July 2022/GIT/Day 3/TestProject/ammendDemo (development)

$ git log

commit f4cef3f3766bb55dde5d60ab2fd394499c49eac9 (HEAD -> development)

Author: Bhushanp5 <bhushan.paradkar@citiustech.com>

Date: Thu Sep 1 00:53:44 2022 +0530

text2.txt added

commit 9c8fb995048138d5c17895ed4f781add2ab5678c

Author: Bhushanp5 <bhushan.paradkar@citiustech.com>

Date: Thu Sep 1 00:53:17 2022 +0530

text1.txt added

commit 21e87a8f2584bd7be4c1aae7d6fc03afbc893303 (master)

Author: Suyog <Suyog@Citiustech.com>

Date: Thu Sep 1 00:16:06 2022 +0530

MyInfo Added on 31st at 12 am

So, here in output we can see that there are two commits we did in development branch. Now I want to apply a commit from branch development to master.

So, first thing that we will have to do is, need to checkout to master.

BhushanP5@IMCCBCP48-MSL1 MINGW64 ~/Desktop/MyData/MyDrive/OneDrive - CitiusTech/Freshers Training July 2022/GIT/Day 3/TestProject/ammendDemo (development)

$ git checkout master

Switched to branch 'master'

Guys, we have already seen merge command so, if I will use command

Git merge development then what will happen?

Definitely, both commits of development will merge with master but I don’t want to do this. I want to merge only one. So, in such kind of scenarios we can use

Cherry-pick. It’s very easy to use but only thing you should know that is hash of commit which you want to merge with another branch.

BhushanP5@IMCCBCP48-MSL1 MINGW64 ~/Desktop/MyData/MyDrive/OneDrive - CitiusTech/Freshers Training July 2022/GIT/Day 3/TestProject/ammendDemo (master)

$ git cherry-pick f4cef3f3766bb55dde5d60ab2fd394499c49eac9

[master 195f580] text2.txt added

Date: Thu Sep 1 00:53:44 2022 +0530

1 file changed, 0 insertions(+), 0 deletions(-)

create mode 100644 text2.txt

In this way our one commit is merge with master branch.

BhushanP5@IMCCBCP48-MSL1 MINGW64 ~/Desktop/MyData/MyDrive/OneDrive - CitiusTech/Freshers Training July 2022/GIT/Day 3/TestProject/ammendDemo (master)

$ git log

commit 195f580830839d0b827334e81b9e6467e1144cd8 (HEAD -> master)

Author: Bhushanp5 <bhushan.paradkar@citiustech.com>

Date: Thu Sep 1 00:53:44 2022 +0530

text2.txt added

commit 21e87a8f2584bd7be4c1aae7d6fc03afbc893303

Author: Suyog <Suyog@Citiustech.com>

Date: Thu Sep 1 00:16:06 2022 +0530

MyInfo Added on 31st at 12 am

Now, for master we got two commits now.

Git-reset.

Guys now we will discuss the next topic that is Git reset.

For better understanding of this we will create new folder as project.

As additional information, we will use VSCode to fire git commands.

**Step 1.** Create a folder and open VS Code with respect to that folder.

**Step 2.** Once your VS Code will open simply press ‘ctrl+`’ or simply click on terminal and open new terminal.

**Step 3.** Fire first command that is git init to make it local repository.

**Step 4.** Create new file as index.html for understanding and add any element in it.

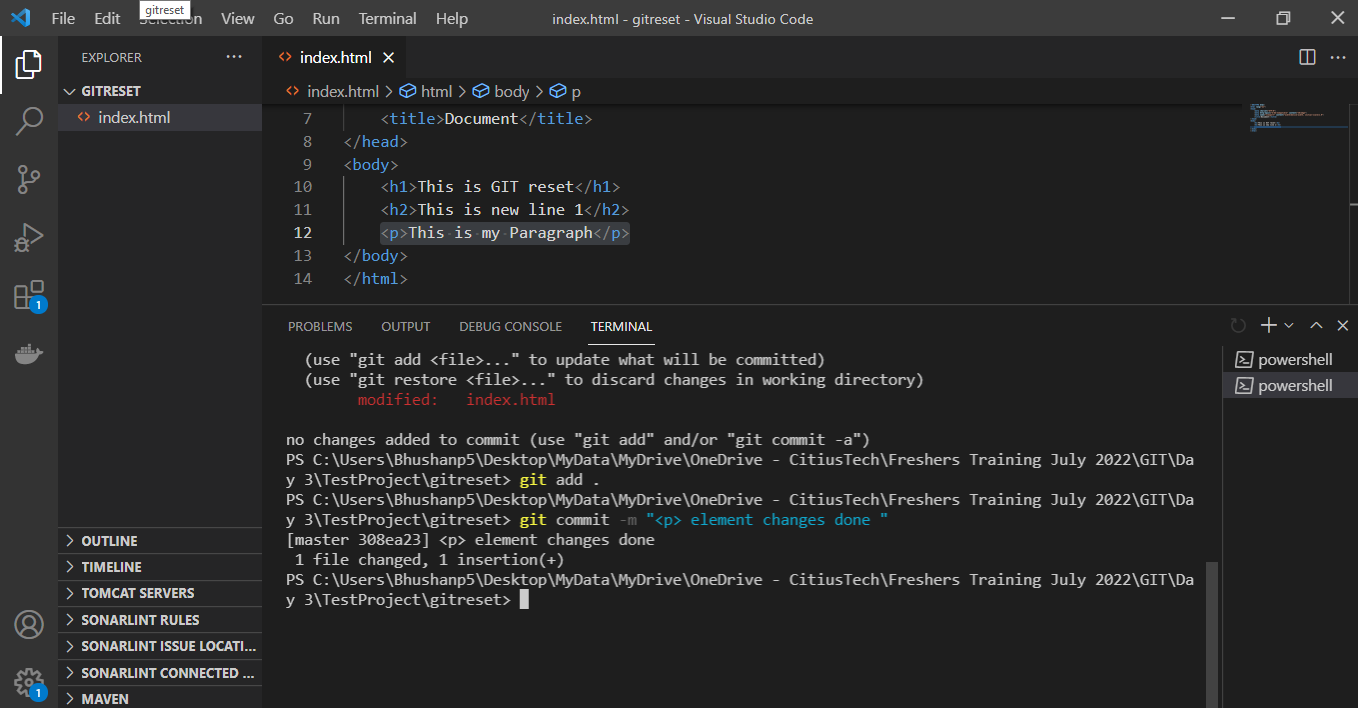
**Step 5.** Fire **git status**, then **git add** . then **git commit -m “index.html is create and committed”**

**Step 6.** Again, add some elements in this index.html file.

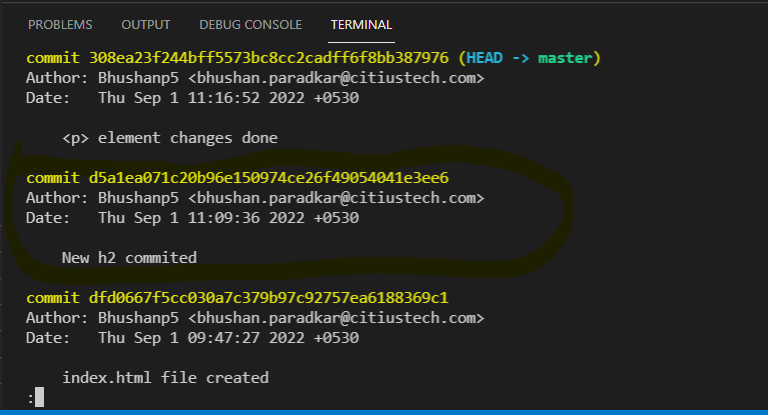
**Step 7.** git status, git add ., git commit -m “new changes made committed”

**Step 8.** So, we have two commits so far. Now add one more element in index.html for understanding of git reset. For example: <p>This is my Paragraph</p>

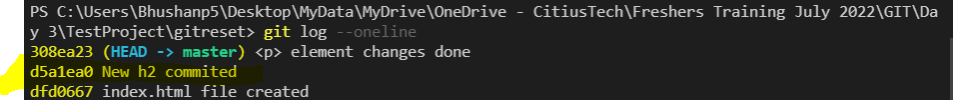
**Step 9.** git status, git add ., git commit -m “new changes <p> made committed”



Now our target is to refuse the changes we made to our index.html file and I want to

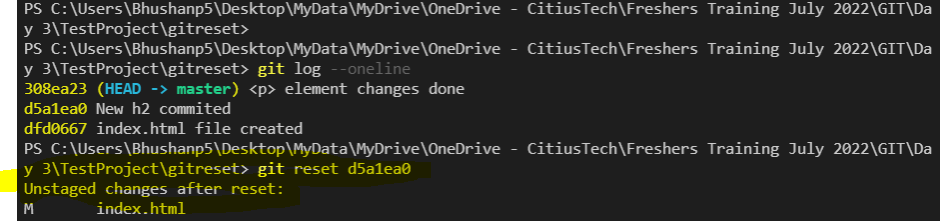


I want to revert changes to this commit.

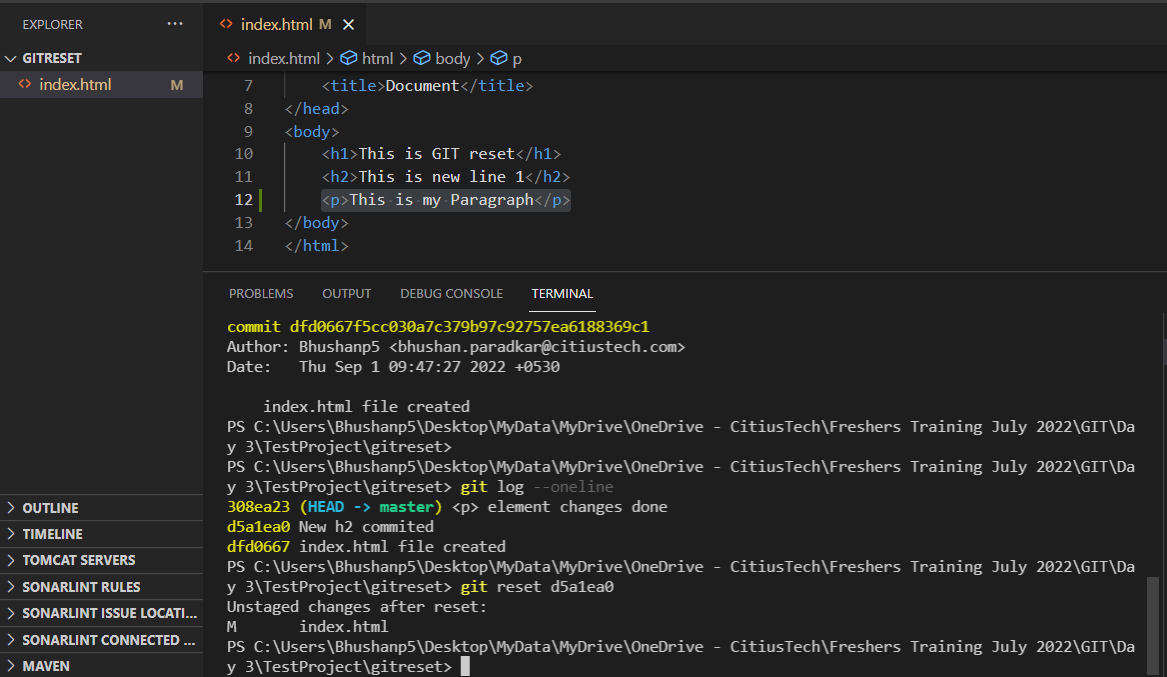


In order to reset to this commit position. We can use reset command.

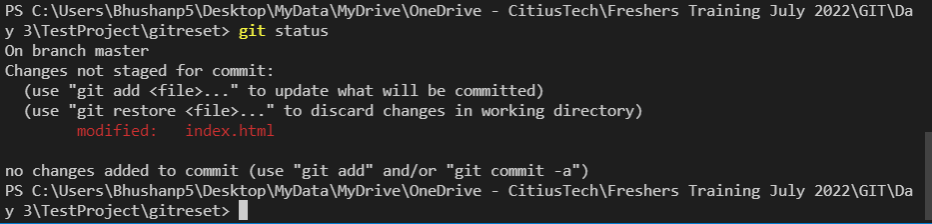
For this purpose we will use git log --oneline which will shows me commit history, and I will have to get hash for that commit.



But we can see here nothing happened here. But we got message that is unstaged changes after reset.

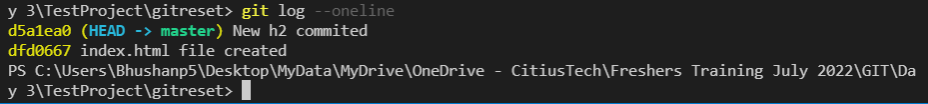


Now, if I will fire command that is git status



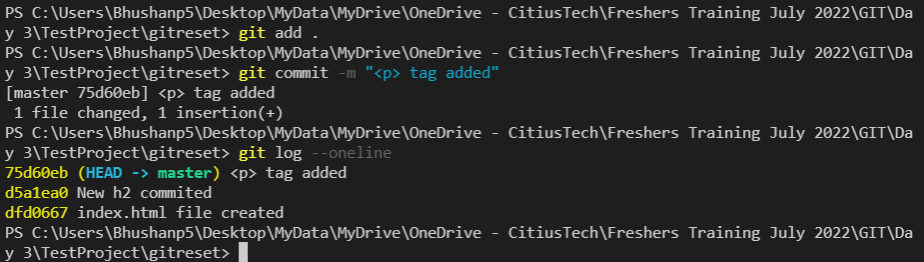
What we can conclude from this is, git reset command kept your changes but it undo commits.

Now, we will see again by git log --oneline which commits are there?



Here we can see last commit is undone.

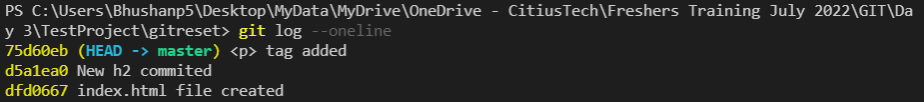
But let’s say you want to keep this <p> tag then again fire command git add ., git commit -m “<p> added again”



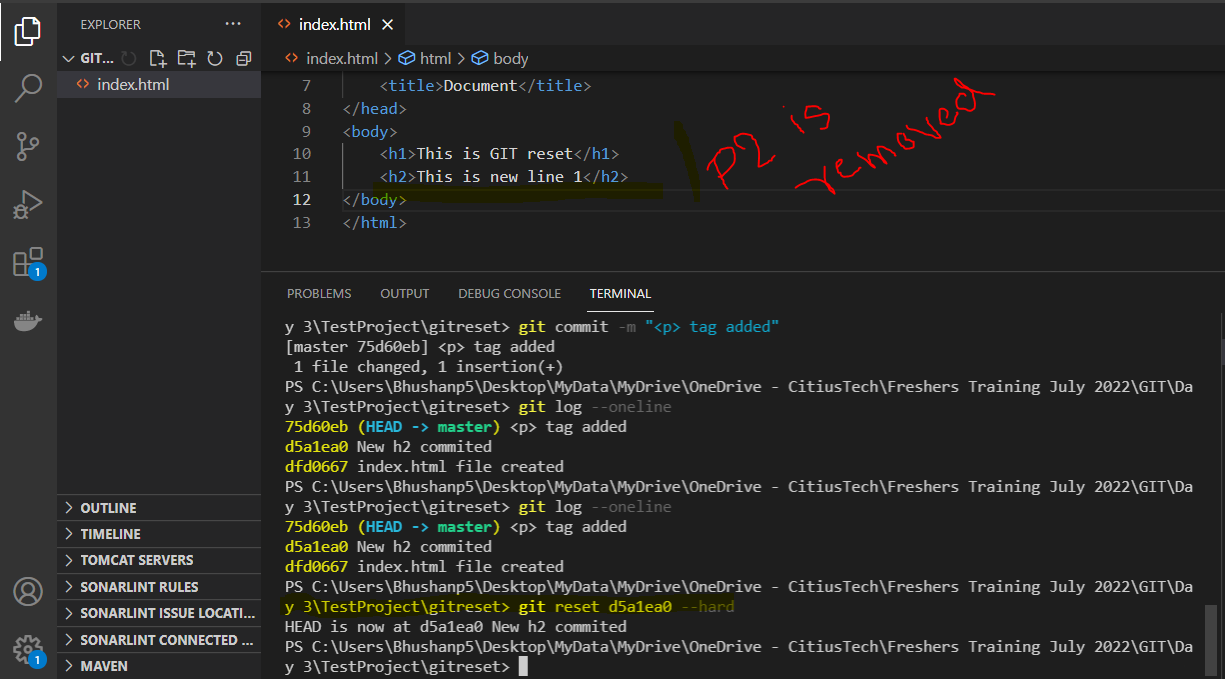
But now here you can see, commit hash got changed. It means earlier commit got removed and new commit is committed.

But now, I don’t want to add that <p> tag and I want to take my file content to previous commit and I also want to delete this <p> tag from my html.

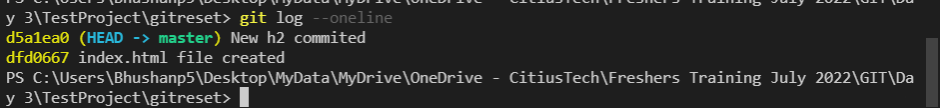
So, first I will check list of commits from which I can select to which commit I want to jump back.



Now I want to revert to new h2 commited. So, I will have to get hash of that commit**.**



Now, we can see the commit is also removed and the <p> tag in html is also removed. And after using command git log --oneline, we can see only two commits.

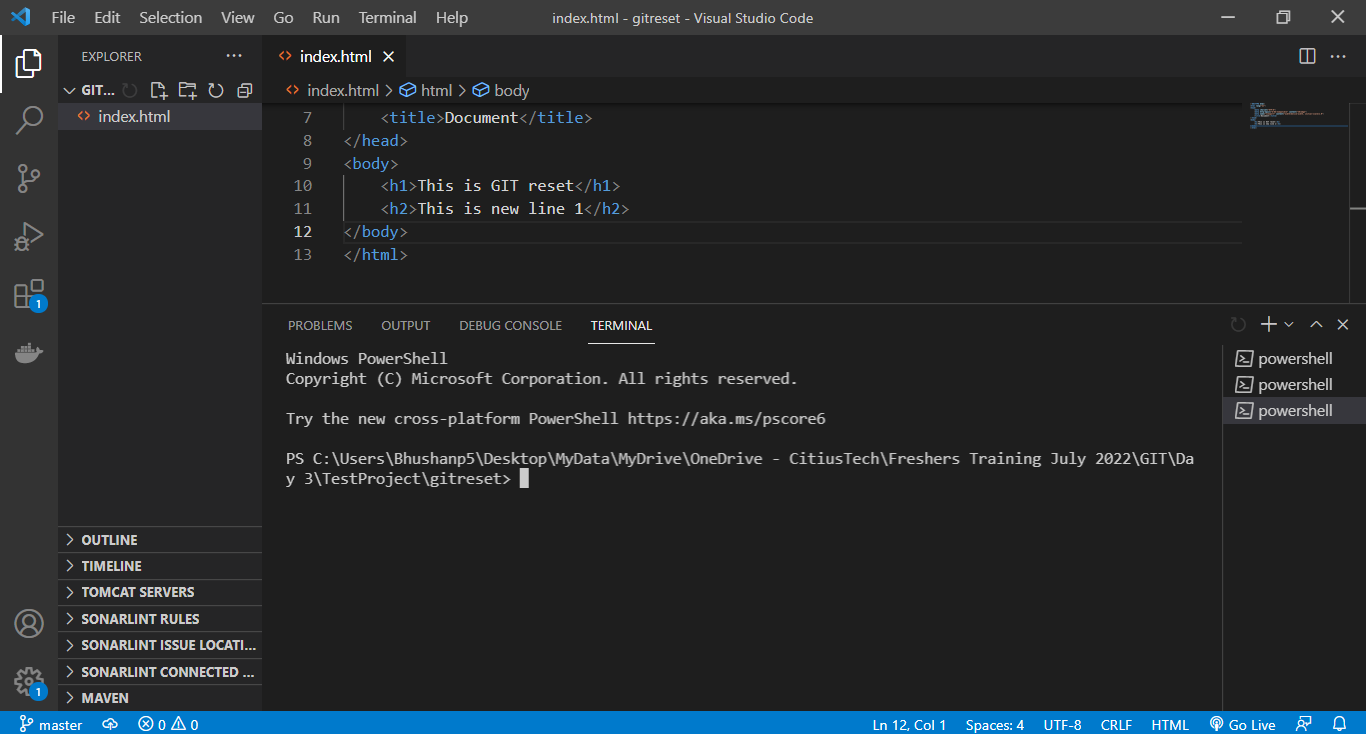


But guys, be very careful while using this command because it will be removed completely.

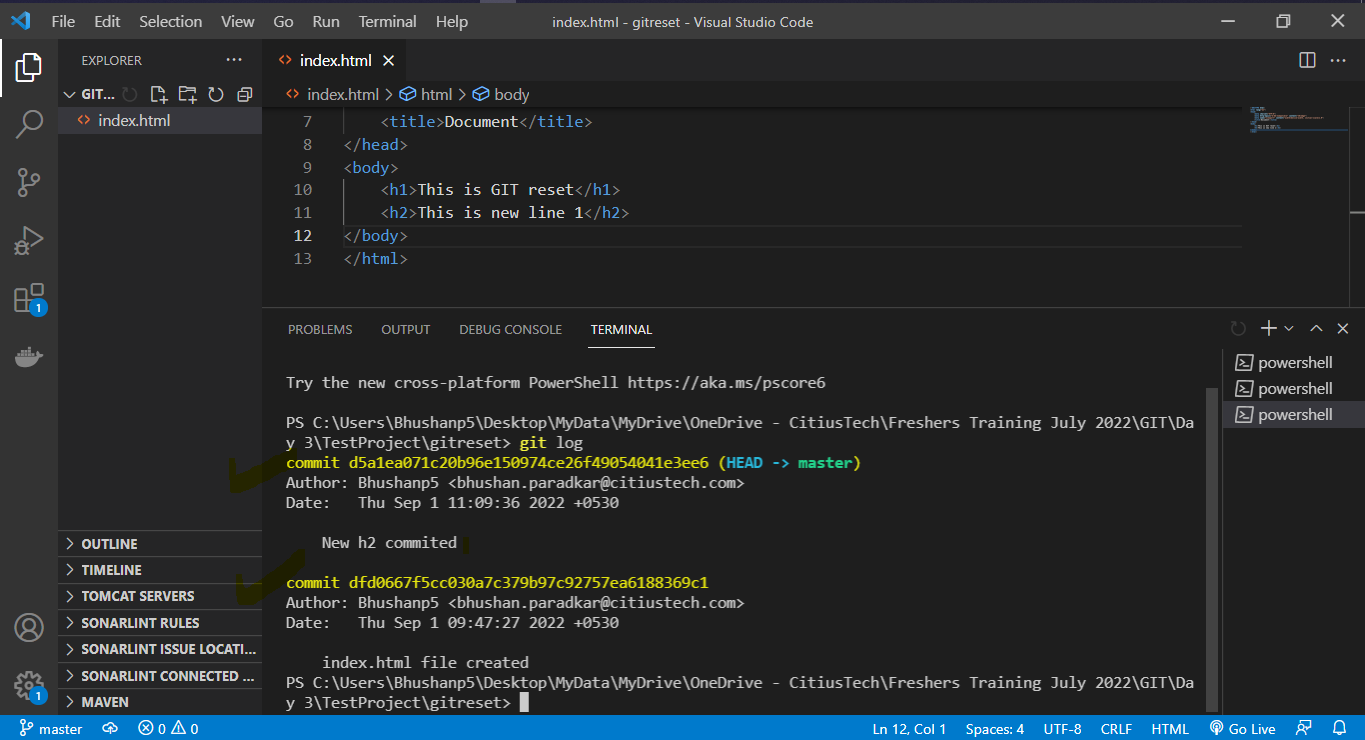
Git-head.

Head is reference to the most recent commit in the current branch.

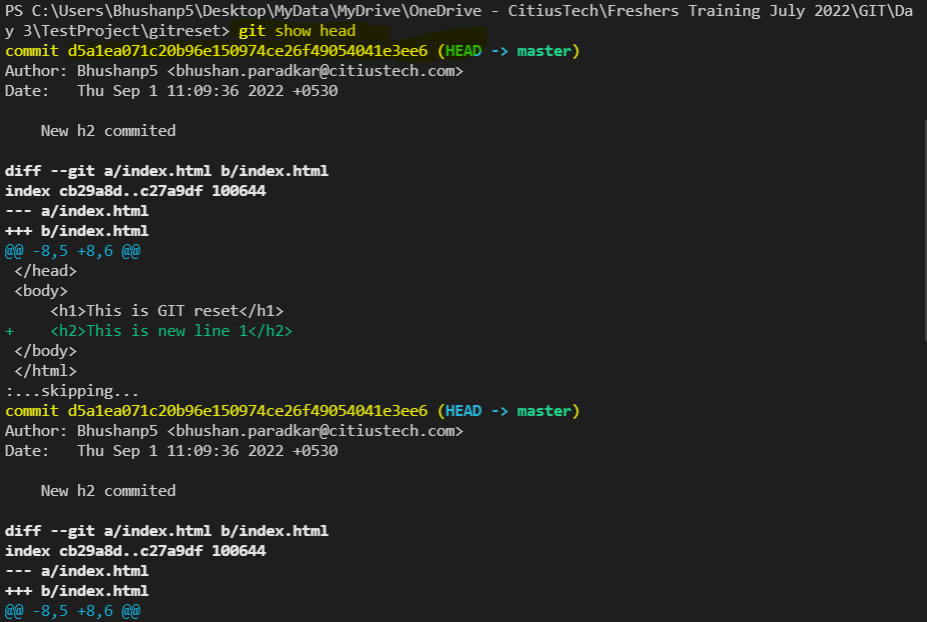
For understanding head, we will use **same demo which we used for reset.**



In this example we have two commits.

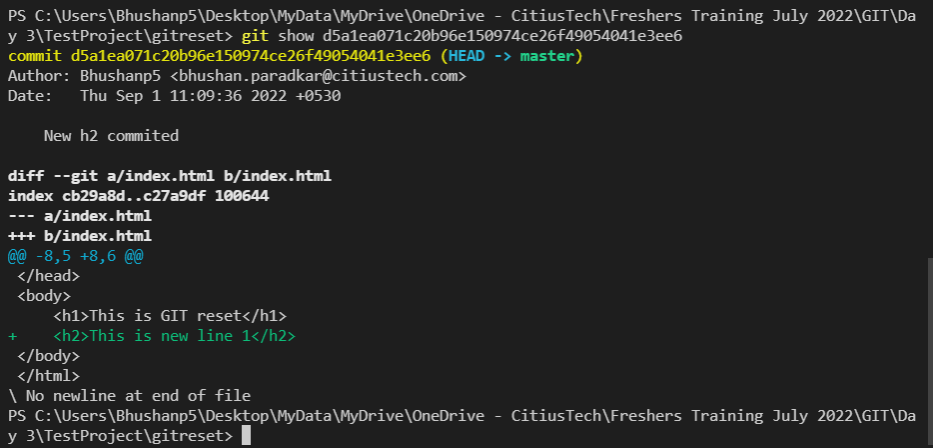


At first, we will fire command that is **git show head**.



It is the same thing If I will fire

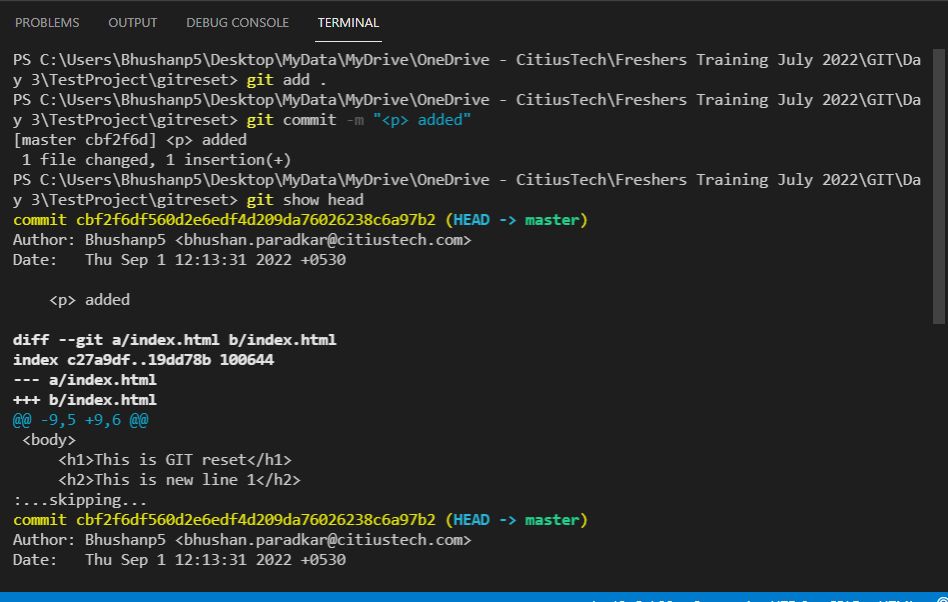
git show d5a1ea071c20b96e150974ce26f49054041e3ee6



Both cases we got same output.

Now, we will do one change in index.html and do git add and git commit.

And fir git show head again.



Here we got our head on <p> added commit.

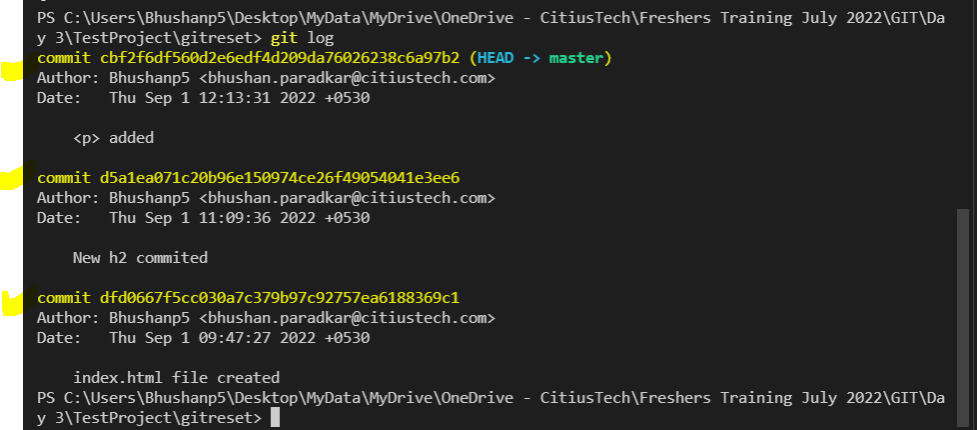
So, here we can conclude that head pointer is mostly pointing to recent commit.

So, you might have doubt, what is need of head. Actually we have some tool for getting difference between two commits like difftool which take two parameters. These parameters are nothing but hash code of commits.

So, whenever you will have such kind of scenarios then we can use head.

Because along with head you can use ~ and number to pointing to that commit.

**For example:**



If I want to point to first commit that is index.html file create then I can use

Head~2, If I want to point to second commit is New h2 committed then I can

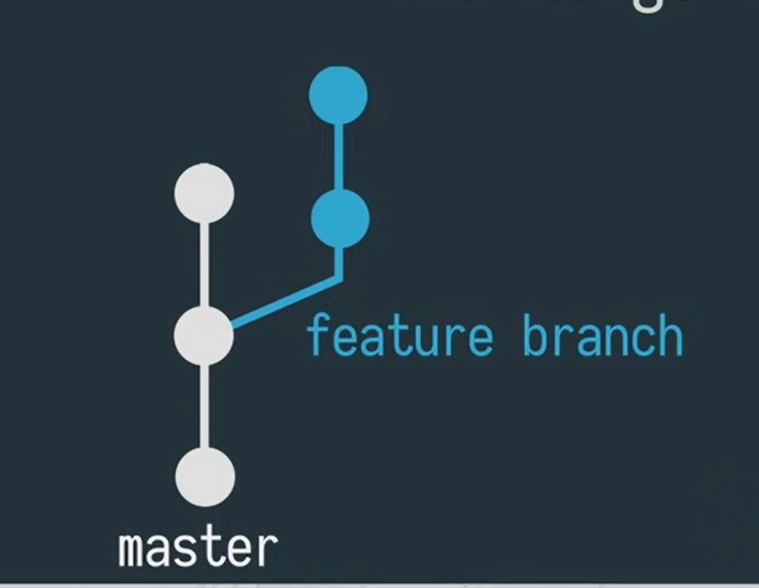
use Head~1 and for recent we have Head only.

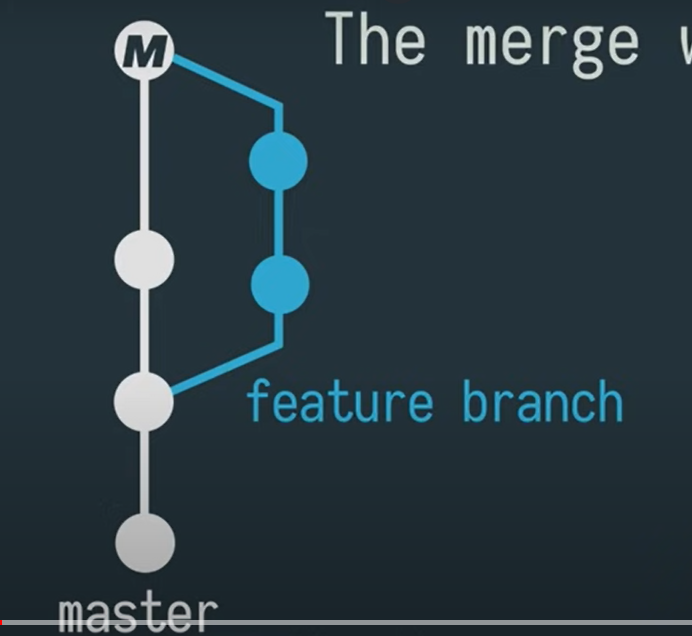
In .git file there is head file in which you can see where it is referring.

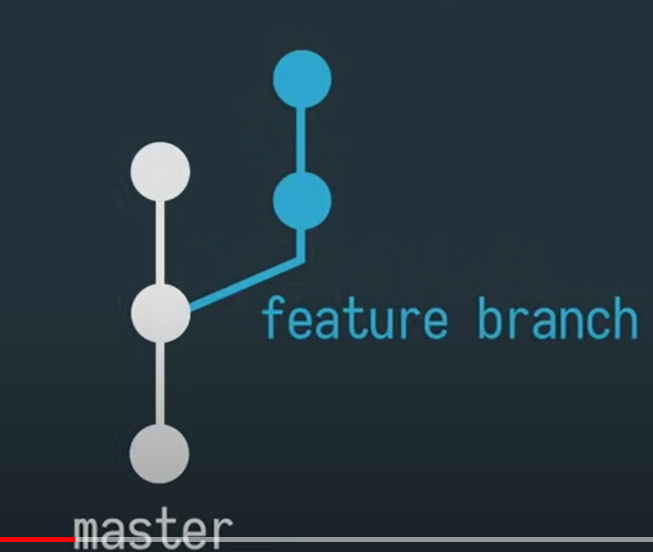
When you change branch automatically the branch in this head file will also change.

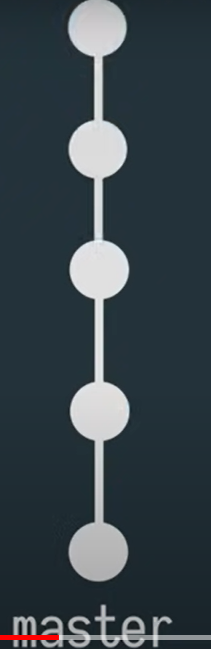
Git-Rebase vs merge

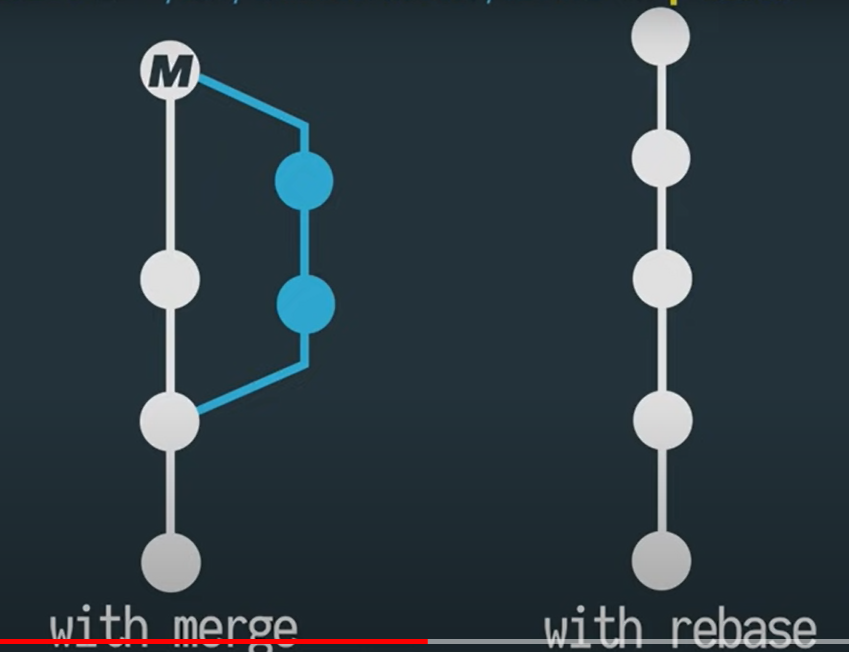
If we will observe merge and rebase then we can see that both are using for same purpose but it does it in different ways. Both commands are introduced to integrate changes from one branch to another branch.











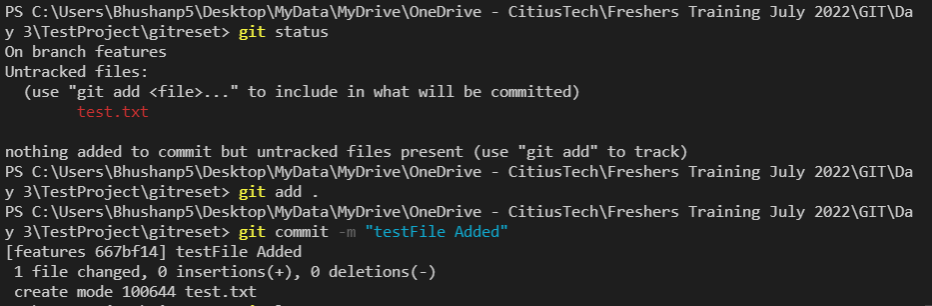
Let’s understand it with example.

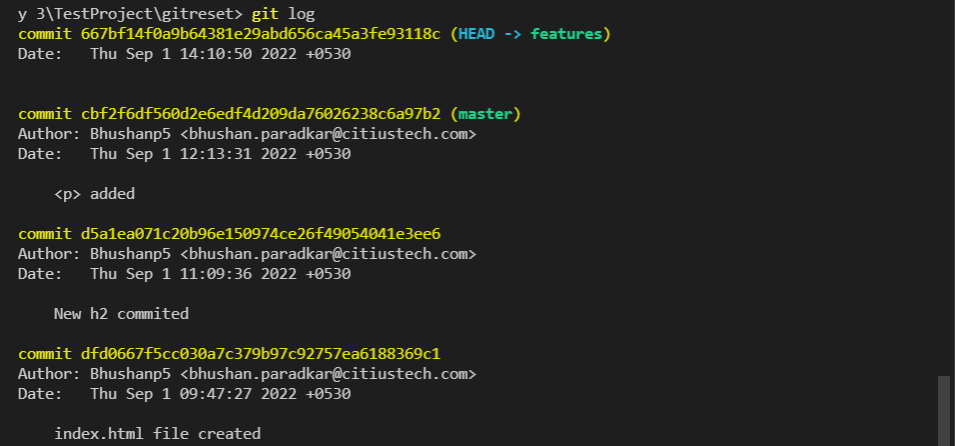
We will use same git bash which we are using.

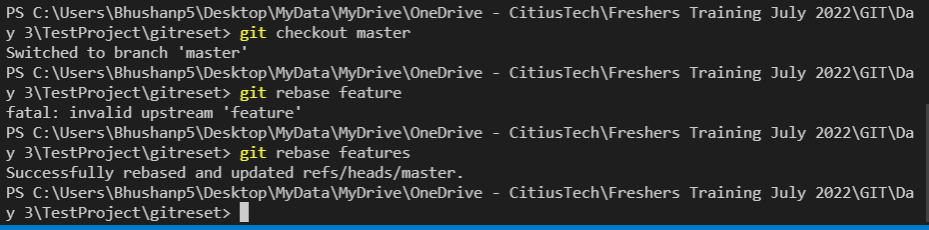
First, we will create a branch using command

**git checkout -b features**

**for understanding we will create a file test.txt in same project.**







**Note: In order to delete branch, we will have to use command.**

**git branch -d branch\_name.**

**IMP 🡪 sFor, Git Tutorial - Branches in git (Create, Merge, Squash merge, etc) - complete video**

[**https://www.youtube.com/watch?v=GDof3qQfyho**](https://www.youtube.com/watch?v=GDof3qQfyho)

**Git merge Conflict**

**<https://www.youtube.com/watch?v=dx7qqbUlWFs>**