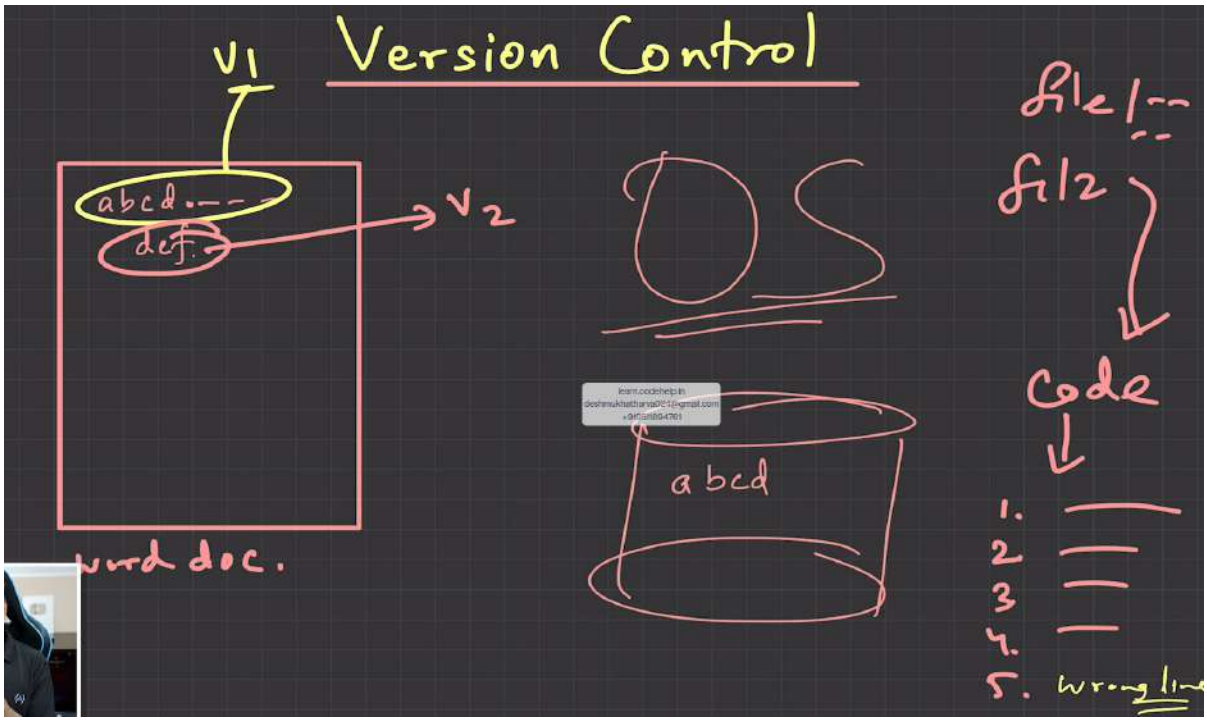
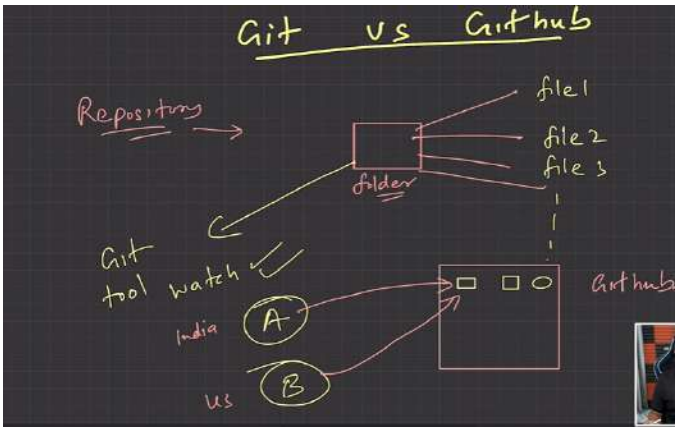
Github & Git

▼ Version Controll (Git vs GitHub)

A software development practice that **tracks and manages changes to files and code** over time.



- **Whenever we do changes in the File the Version Controll Tool Watchs all the files that we are changeing.**
- **Git is a version controll tool...**
- **ex:- git , subversion , perforce**
- Youtube is Hosting Service for Videos such like github is the hosting service for a repositries.



▼ Installation of git

References & Links

https://git-scm.com/downloads

Open Command Prompt for Following Commands

cd git - - version

Create a Account on a Github Account from a Website

https://github.com/

git configuration

1. git config - - global user.name ‘Atharva’
2. git config - - global user.name ‘username’

   Note : Same User Name as you created in github account.
3. git config - - global user.email “deshmukhatharva024@gmail.com”
4. git config - - list → Command to get the All Information of user , email set....

Github Desktop

Installation References

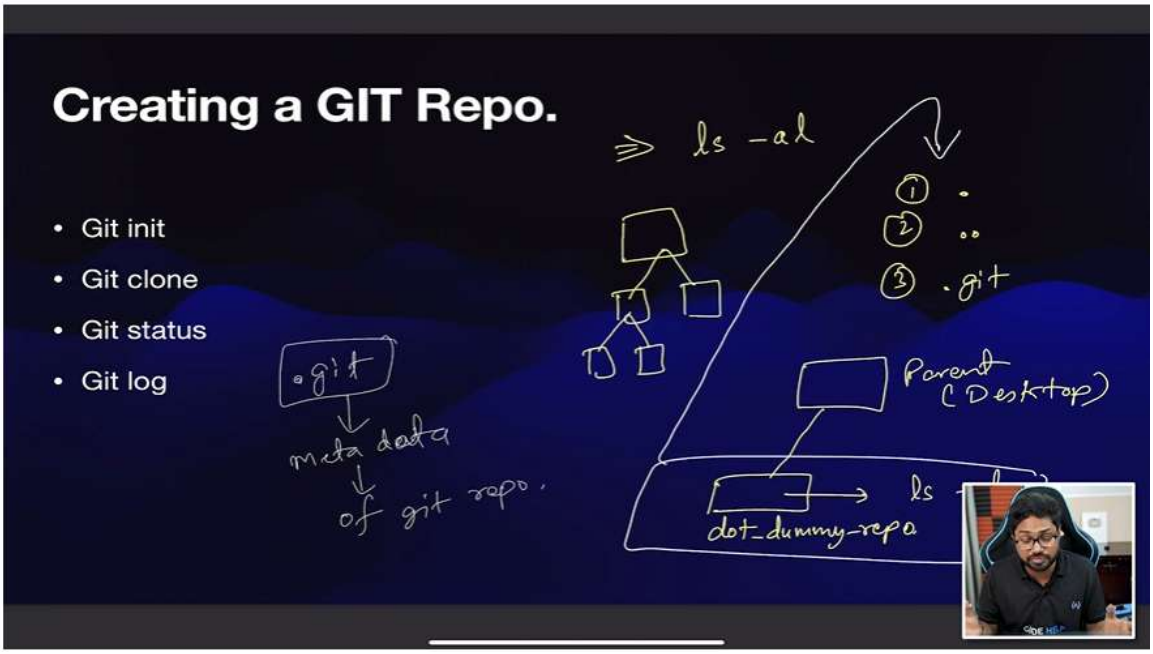
https://desktop.github.com/download/

▼ Creating a Git Repo

Commands

1. **ls → To see whether anything is present in that folder or not.**
2. **git init → To Convert the normal Folder into a empty git Repo**

   As .git folder is is created inside that normal folder. (As By Default that .git folder is hidden).
3. **ls -al → to show the following file that are present inside a project Folder.**



4. **git Status**

   To Check Which File is being added or existing file is being modified or any file is being deleted.

To Clone Others Repo and Contribute in it

1. git Clone <link> to Clone the Repo in our Local Machines.

2. git Status

To Check Which File is being added or existing file is being modified or any file is being deleted.

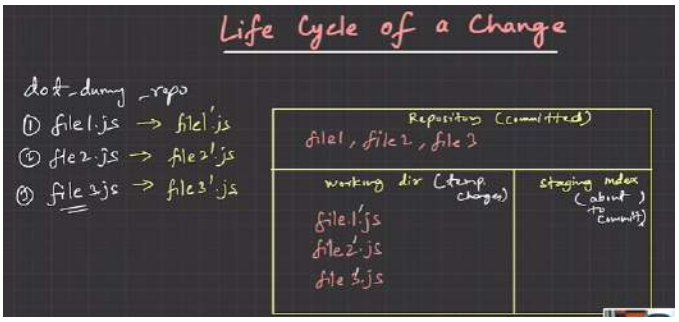
3. git diff

Through This Command we are able to see the following changes that happened in the file such that which of the following line is removed and which of the following are added.

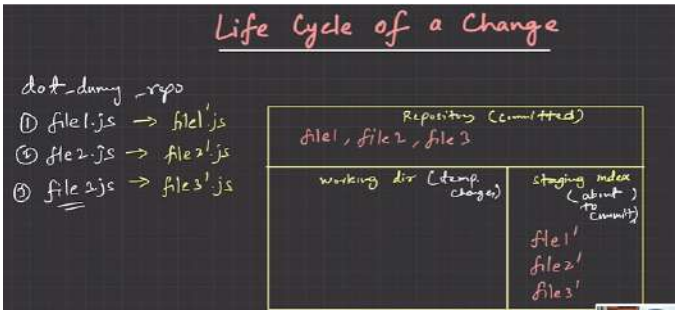
```
lovebabbar@192 myBTP % git diff
diff --git a/GOA.py b/GOA.py
index 6ff3dd1..ab2b1bc 100644
--- a/GOA.py
+++ b/GOA.py
@@ -7,7 +7,7 @@ import settings
 from sklearn.model_selection import train_test_split
 from sklearn.metrics import accuracy_score

-minimum_no_of_hidden_neuron = 2
+minimum_no_of_hidden_neuron = 50


 def give_a_random_solution(no_of_features):
lovebabbar@192 myBTP %
```

▼ Life Cycle of a Change

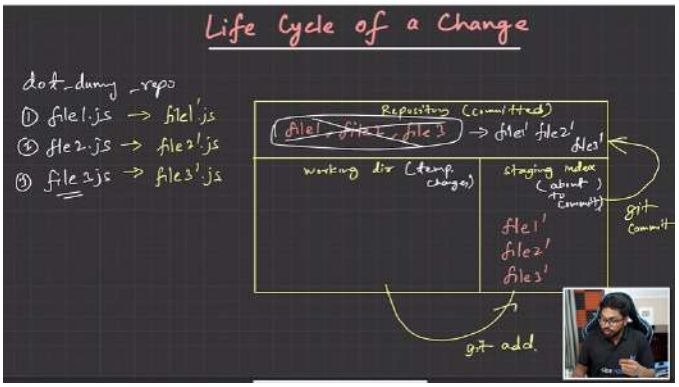
The Following Files that are Modified but not operation performed rather than git status.



After git add . command the all modified files would move from a working dir (Temporary Changes) to Staging Index (Commit).

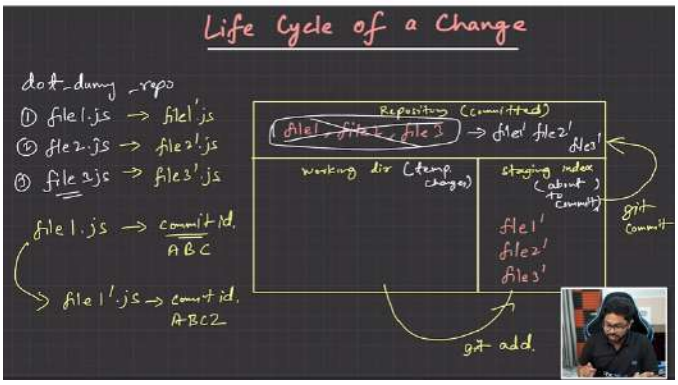


After the git commit -m ‘ ’ the Files that are changed are being finalized and moves in repository (commited)



After the Respository (commited) the Modified File Replaces the Original File or previous File.

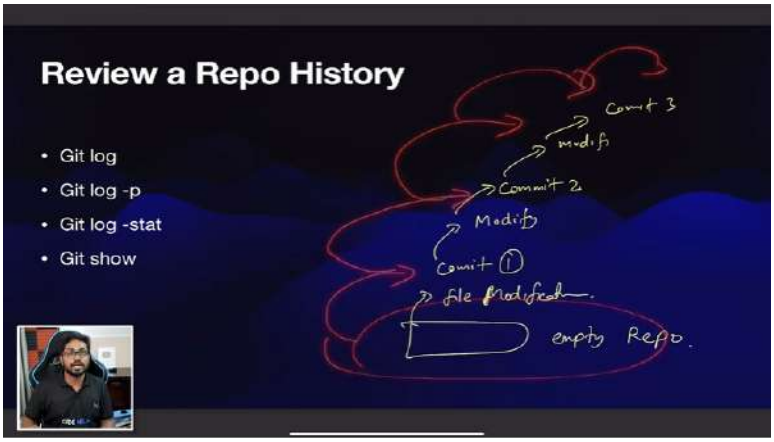
Note :- Git Has the Habit of Maintaining the previous aswell as new id of commit so that we are able to see the changes that we have done in the earlier commits .



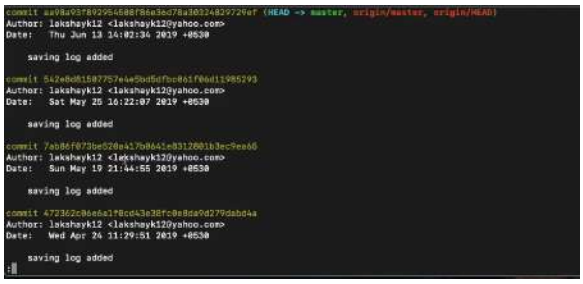
▼ Review a Repo History

- Git log
- Git log -p
- Git log -stat
- Git show

Normal Process



git log :- It is used to tell me which is the top & (latest) commit and till last older commit means it actually shows from latest to older commits.



It is used to give us in hash id...

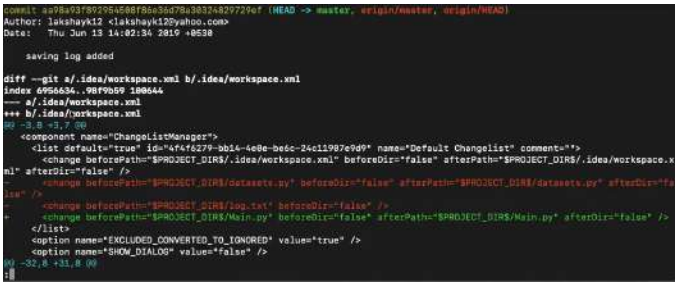
Note :- To See the latest 3 Commits the Following Command

git log -3

Syntax:- git log -(How many Latest Commit in Number)

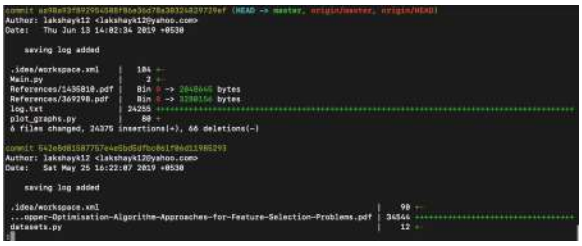
git log -p

It gives the information regarding the commit there author and date , Aswell as it also shows diff (git diff) also of a code



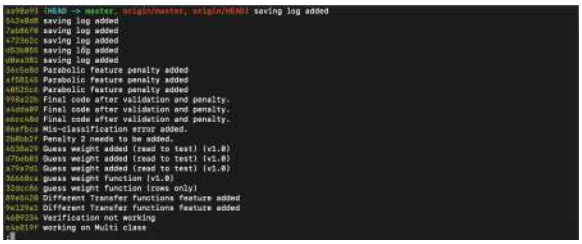
git log - -stat

Now Just I Want to see the FileName That are being Modified...



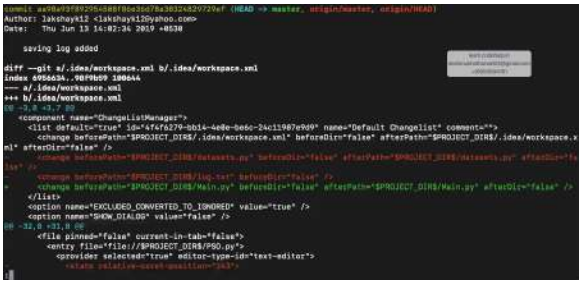
git log - -oneline

We are able to see the sha ids aswell as commit messages



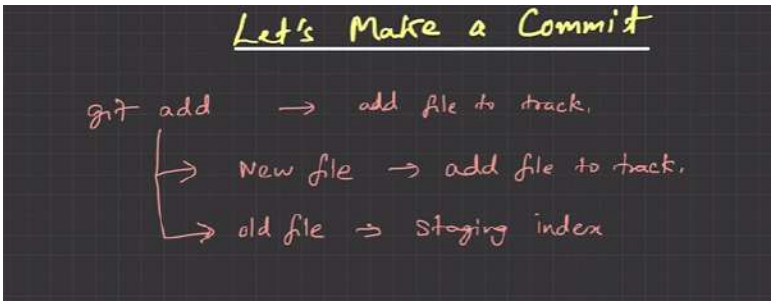
git show sha ID

Now I Want to See in particular Commit What are the Following Changed That Happened



▼ Doing your First Commit

1. git init → Create a Empty git repo
2. git status —→ looking for changes that happened in file
3. git add . → To track a File (Add File to track)



4. git commit -m ‘New File Added’
5. git status ——→ as new File is Added But there is no change
6. git show shaID → To See the Following Changes that are happened in this particular commit

```
lovebabbar@192 myrepo % git show 73ff9cdddfacc23d13aa5e688c3e8543f7bea62a
commit 73ff9cdddfacc23d13aa5e688c3e8543f7bea62a (HEAD -> master)
Author: Love Babbar <lovebabbar@192.168.0.215>
Date:   Fri Dec 23 01:32:34 2022 +0530

    init main.cpp

diff --git a/main.cpp b/main.cpp
new file mode 100644
index 0000000..63d2bbf
--- /dev/null
+++ b/main.cpp
@@ -0,0 +1,5 @@
+#include <iostream>
+
+int main(){
+        std::cout<<"Hello World";
+}
\ No newline at end of file
lovebabbar@192 myrepo %
```

7. Now We Modified a File
8. git status → File has been Modified...
9. git diff filename → What Changes that are being modified in that particular file....
10. git add . ——→ To Move the Fill in the staging Index....

Note :- git add . ——→ is not Recommend instead add name....

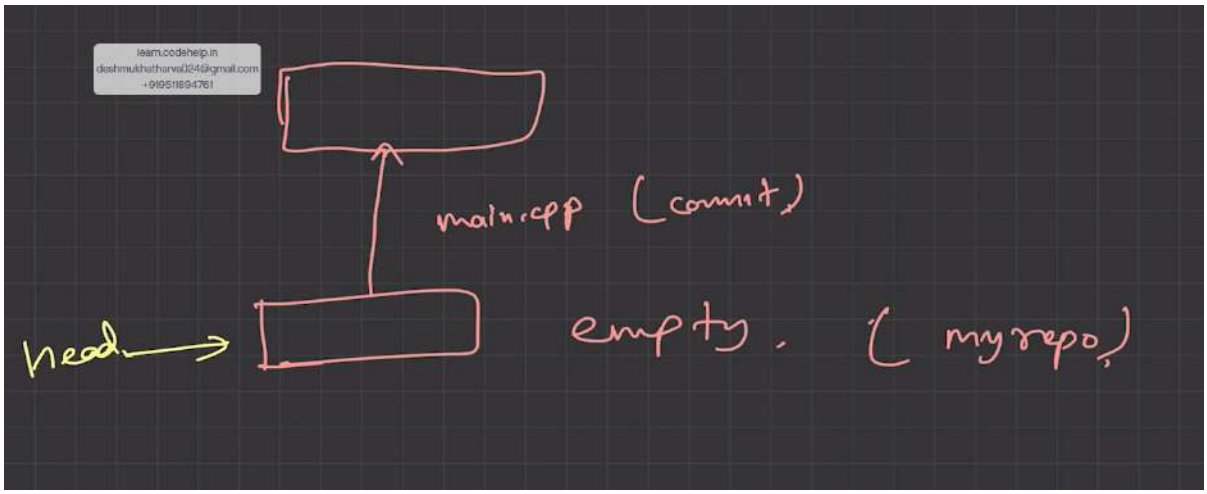
`git add main.cpp`

11. git Status → For the File to be Commited
12. git commit -m ‘Text File Modified’
13. git log ——→ To See the Commits

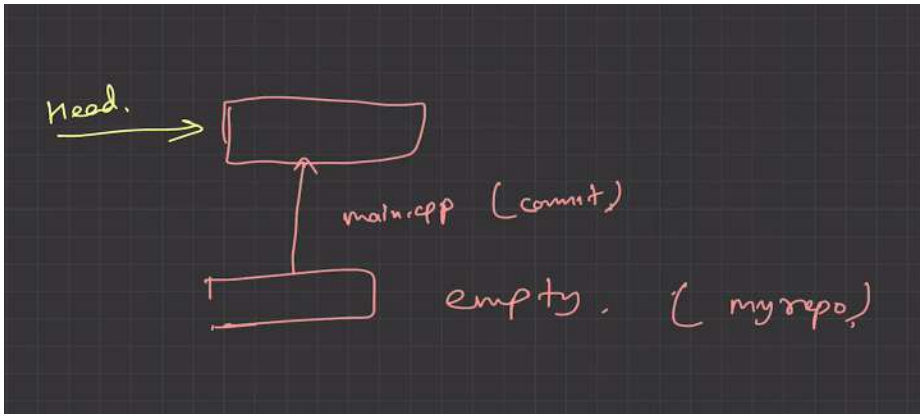
The Meaning of Head

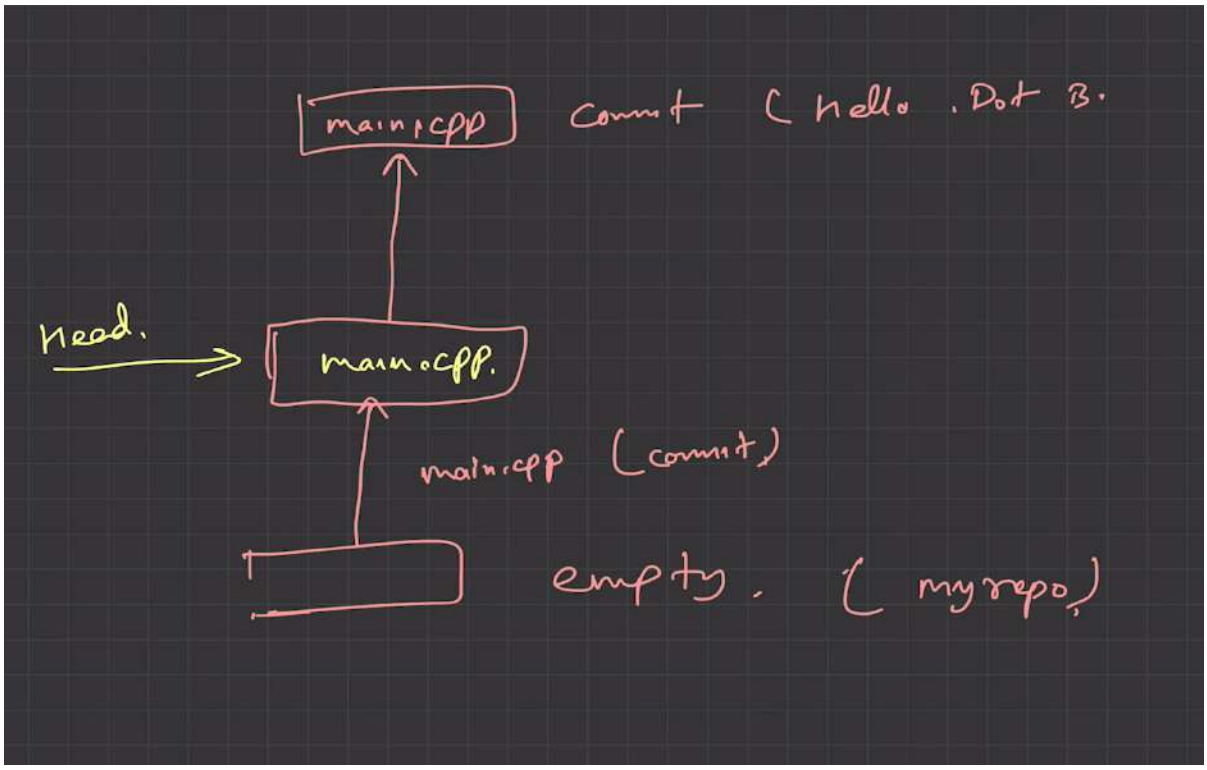
```
commit 30814b0da4ad9c55fd1c8160a93af3ce8915a3b6 (HEAD -> master)
```

First head Points to the Empty Repo

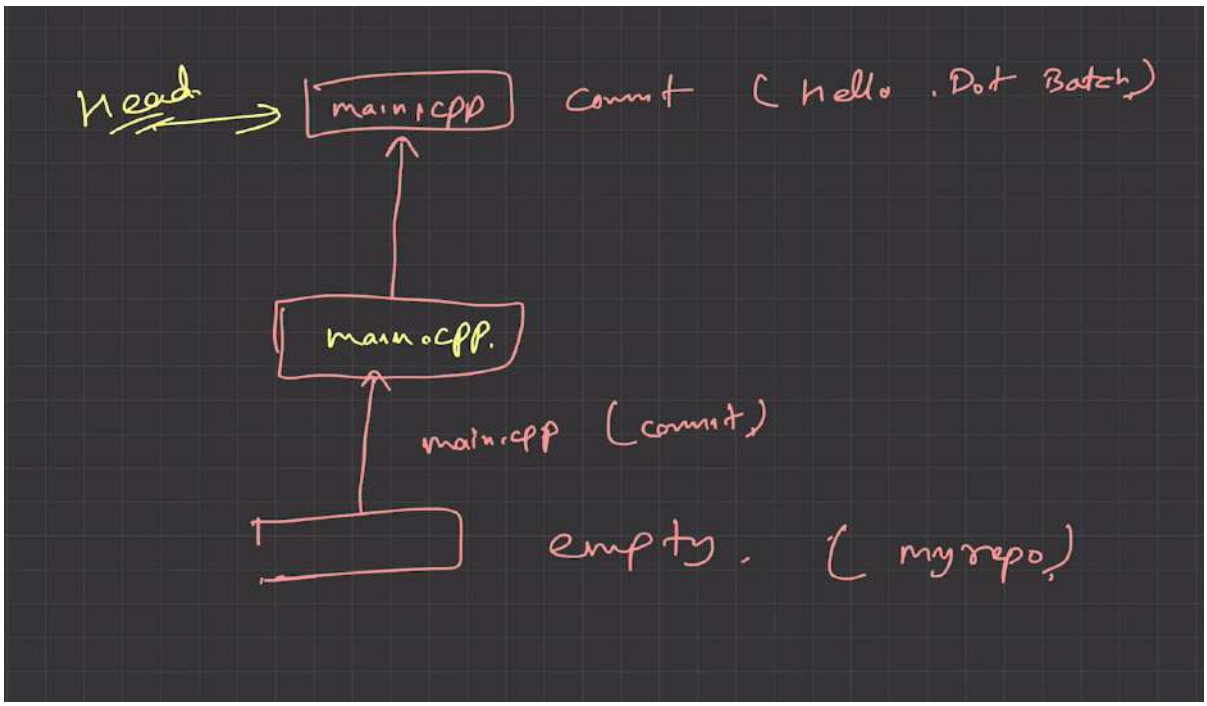


When there is a Recent Change Then Head Starts Pointing to that File





Note :- Head Always Points Towards the Latest Commit



14. When you Accidently change some of the file that you donot want to change and now you want to discard that file.

git restore filename (git restore main.cpp)

As that File will restore in head which is the latest recent commit

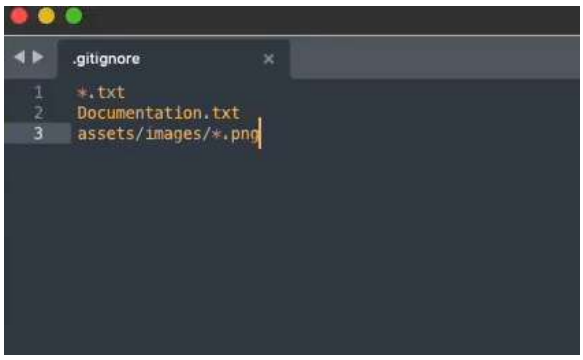
```
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   main.cpp

no changes added to commit (use "git add" and/or "git commit -a")
lovebabbar@192 myrepo % git diff
diff --git a/main.cpp b/main.cpp
index 14a6901..838d97a 100644
--- a/main.cpp
+++ b/main.cpp
@@ -3,4 +3,5 @@
 int main(){
        std::cout<<"Hello World";
        std:cout<<"Hello .Dot Batch :)";
+       abcd;
 }
\ No newline at end of file
lovebabbar@192 myrepo % git restore main.cpp
lovebabbar@192 myrepo % git status
On branch master
nothing to commit, working tree clean
lovebabbar@192 myrepo %
```

15. As We are Keeping more Files Such as .cpp , .js , .css and many files in the git repo as you have also added a files such as (.docx , .rtf) as you donot want to track those file then use this following command.

Put in the GitIgnore File (.gitignore)

*.txt —> All related to .txt file are not being tracked & (ignored).

/document.txt ——→ We can Add a Path also in it....



16. Adding one Note

To Merge a Branch code of another in our Branch

Such that on Another Branch tell him to do

git add .

git commit -m ‘text Add’

donot use git push command

Then Switch on that branch where you need to take the code of another branch

git switch branch name

donot use git switch -b branch name It is used to Create a new Branch

After Switching on That Branch
git merge another branch name

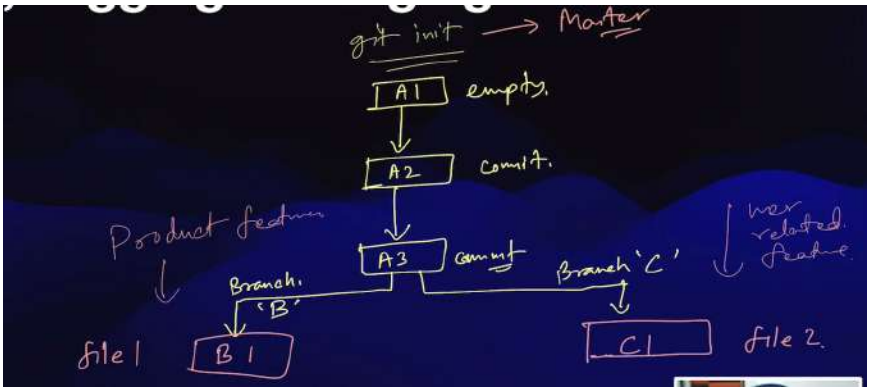
So Now the Following Changes of another with commit only will come to your branch.

▼ Branching, Tagging & Merging

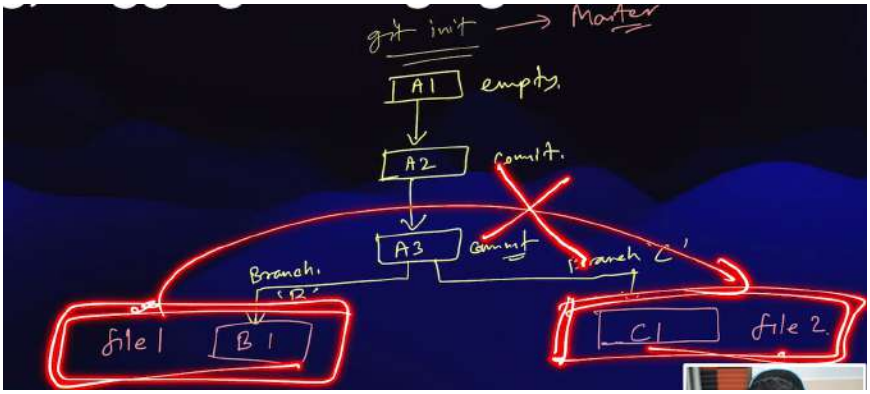
- Git tag
- Git branch ⟶ It is used to list how many branches are present in that particular repo.
- Git stash
- Git checkout
- Git Merge

▼ Branching

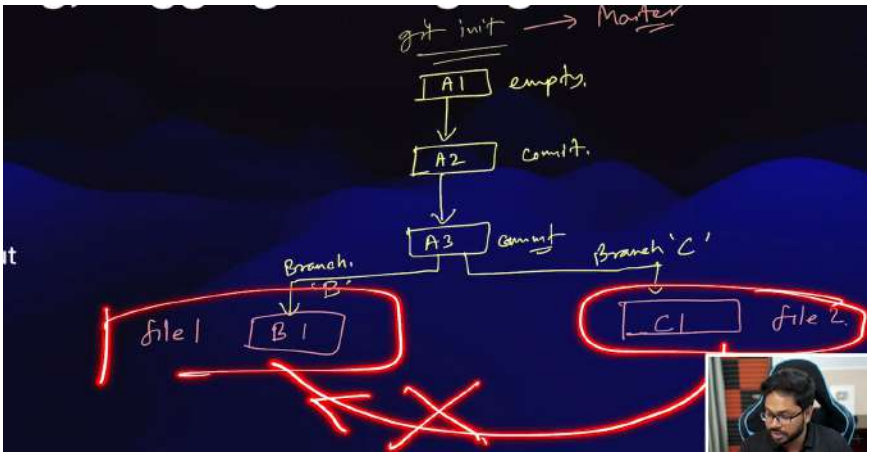
1. Firstly When We Run a Command git init a Empty Repo is created with .git hidden folder.



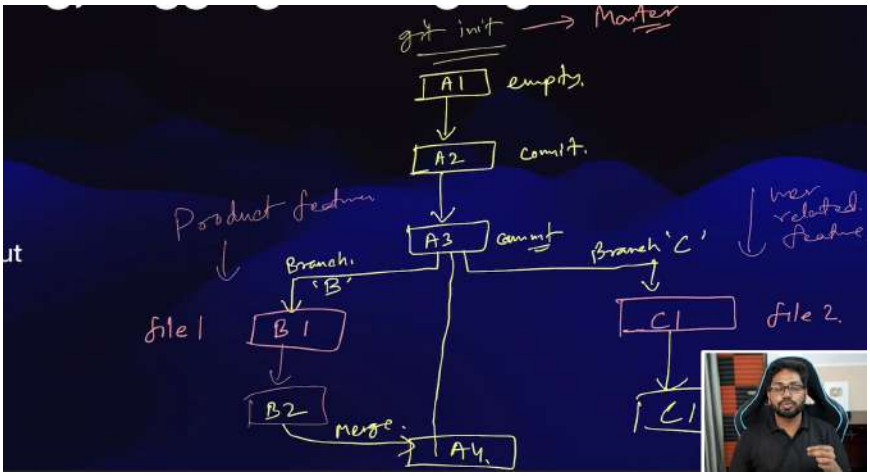
2. The Following Change regarding the File 1 that is at the branch A will not be not there at the branch B



3. The Following Changes regarding the File 2 that is at the branch B will not be there at the branch A



4. Whenever the work is over regarding the features on B1 Branch It will merge in A3 branch



Git branch

It is used to list how many branches are present in that particular repo. and in Which Branch you are currently present with * sign

```
lovebabbar@192 myrepo % git branch
* master
lovebabbar@192 myrepo %
```

If Branch is already present and you need to switch the branch then we can use this command also

`git branch branch_name`

or alternative

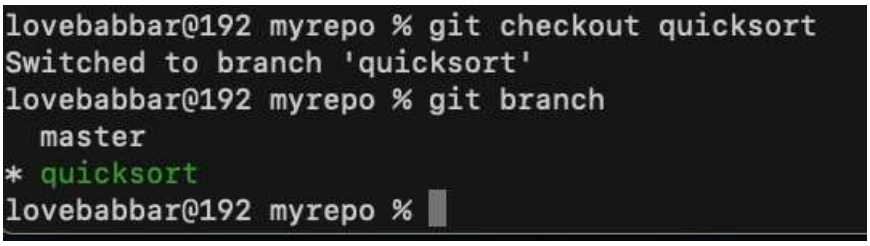
`git checkout branch_name`

To Create a New Branch use this command

`git checkout -b branchname`

```
lovebabbar@192 myrepo % git branch quicksort
lovebabbar@192 myrepo % git branch
* master
  quicksort
lovebabbar@192 myrepo %
```

Note :- Now Currently also we are on Master Branch not on newly created branch.



git branch branch-name

1. If we run a git branch branch-name

the new branch will be created but you will not be shifted to new branch you will remain on a existing branch only

2. **To shift a branch you need to run a command git checkout branch-name**

git checkout -b branch-name

1. if we run a git checkout -b branch-name command the newly branch will be created but also you will be shifted to new branch so pointer points on the new branch.

Git Merge

To Merge a Branch

Syntax:- git merge branch-name

ex:- git merge bubblessort

To Delete a Branch

command :- **git branch -d bubblesort**

```
lovebabbar@192 myrepo % git branch
  bubblesort
* master
  quicksort
lovebabbar@192 myrepo % git branch -d bubblesort
Deleted branch bubblesort (was 8f3e69f).
lovebabbar@192 myrepo %
```

```
lovebabbar@192 myrepo % git branch
  bubblesort
* master
  quicksort
lovebabbar@192 myrepo % git branch -d bubblesort
Deleted branch bubblesort (was 8f3e69f).
lovebabbar@192 myrepo % git branch
* master
  quicksort
lovebabbar@192 myrepo %
```

Note :- **GitHub try to do the automerging But Failed due to Merge Conflict**

```
lovebabbar@192 myrepo % git merge quicksort
Auto-merging main.cpp
CONFLICT (content): Merge conflict in main.cpp
Automatic merge failed; fix conflicts and then commit the result.
lovebabbar@192 myrepo %
```

Solution ➖

1. Solve the Merge Conflict and push with the regular commands.
2. If we not wanted to slove merge conflict use this command git merge - - abort so the following changes came would be discarded.

Important Point

Note :-
As We Run Two Command

1. git add . ——→ To Go Files in the Staging Index....
2. git commit -m ‘Text File Added’ ——→ Which is the Final Commit

If We Want to Perform both command in a Single Command then use this command

`git -am`

Add & Commit at one time.

ex:- git -am ‘Text File Modified’

▼ Tagging

As for Example :- Both of the Branches are Merge to the Main Branch So Now This is My Beta release.

So github tells that we have some better Idea for That.

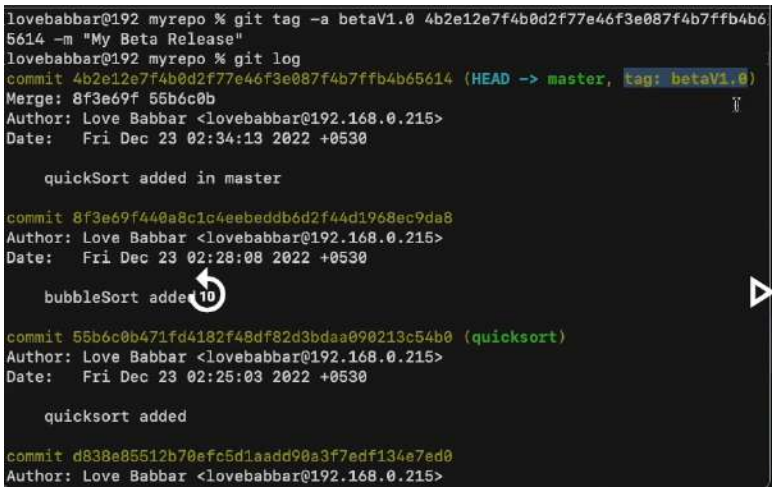
1. If we Tag a Specific Sha ID or Commit.

Using this command

git tag -a betaV1.0 sha ID -m ‘”My Beta Release”

```
lovebabbar@192 myrepo % git tag -a betaV1.0 4b2e12e7f4b0d2f77e46f3e087f4b7ffb4b6
5614 -m "My Beta Release"
```

So Now If We Do the Git log we will go



- If We Change Again the file and Use the Regular Commands
- **git add .**
- **git status**
- **git commit -m ‘Text File Modified’**
- **git push**

```
commit 9e7459a27aa627f68bd25a86fe9637cb84c69b0d (HEAD -> master)
Author: Love Babbar <lovebabbar@192.168.0.215>
Date:   Fri Dec 23 02:39:02 2022 +0530

    merge sort added

commit 4b2e12e7f4b0d2f77e46f3e087f4b7ffb4b65614 (tag: betaV1.0)
Merge: 8f3e69f 55b6c0b
Author: Love Babbar <lovebabbar@192.168.0.215>
Date:   Fri Dec 23 02:34:13 2022 +0530

    quickSort added in master

commit 8f3e69f440a8c1c4eebeddb6d2f44d1968ec9da8
Author: Love Babbar <lovebabbar@192.168.0.215>
Date:   Fri Dec 23 02:28:08 2022 +0530

    bubbleSort added

commit 55b6c0b471fd4182f48df82d3bdaa090213c54b0 (quicksort)
Author: Love Babbar <lovebabbar@192.168.0.215>
Date:   Fri Dec 23 02:25:03 2022 +0530
```

So Now Master is Ahead of a Tag Version as Head is Pointing Towards master

So If we want to delete the tag for that tag version which is one Commit Behind of a Master Use This Command

git tag -d betaV1.0

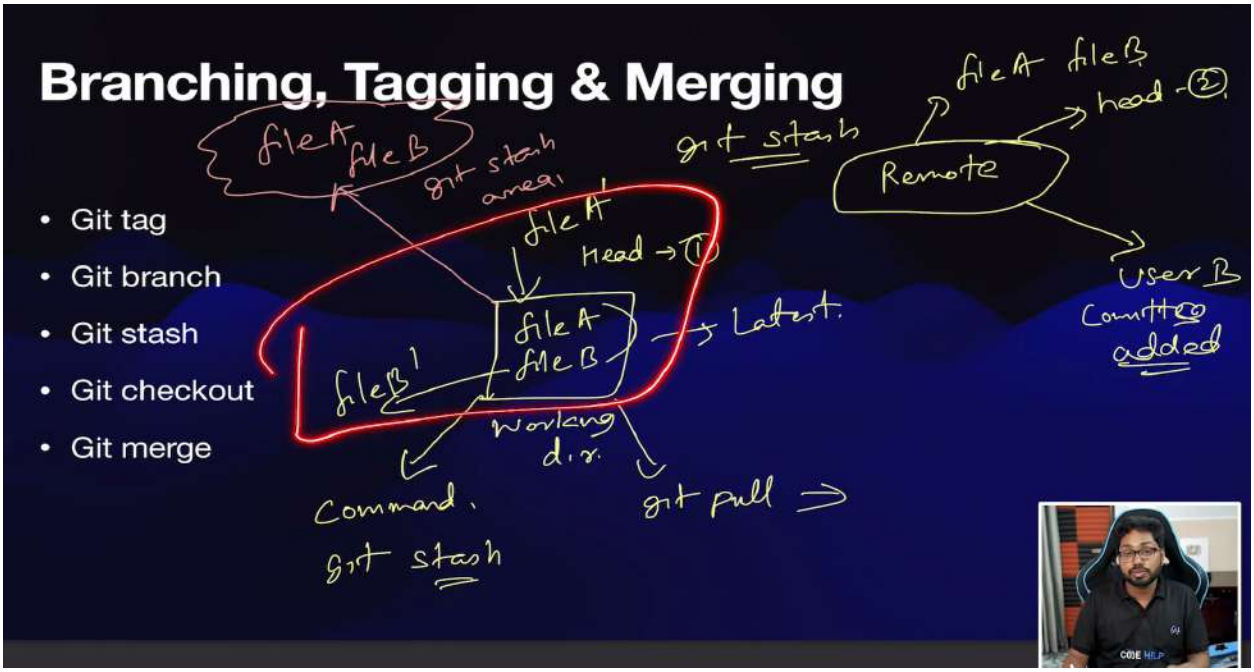
```
commit 4b2e12e7f4b0d2f77e46f3e087f4b7ffb4b65614 (tag: betaV1.0)
Merge: 8f3e69f 55b6c0b
Author: Love Babbar <lovebabbar@192.168.0.215>
Date:   Fri Dec 23 02:34:13 2022 +0530

    quickSort added in master

commit 8f3e69f440a8c1c4eebeddb6d2f44d1968ec9da8
Author: Love Babbar <lovebabbar@192.168.0.215>
Date:   Fri Dec 23 02:28:08 2022 +0530

    bubbleSort added

commit 55b6c0b471fd4182f48df82d3bdaa090213c54b0 (quicksort)
Author: Love Babbar <lovebabbar@192.168.0.215>
Date:   Fri Dec 23 02:25:03 2022 +0530

lovebabbar@192 myrepo % git tag -d betaV1.0
Deleted tag 'betaV1.0' (was 2911d5b)
lovebabbar@192 myrepo %
```

- So I Want to Add a tag for a master Branch Use this Command Again....

git tag -a betaV1.0 sha ID -m ‘”My Beta Release”

So Now Tag will be Added to a master Branch

Git Stash



Scenario Based

Two Commands Required

1. git stash ——→ To Store Your Changes in Temporary Storage
2. git stash apply —→ To Get Back Your Changes From a Temporary Storage

After taking a Git pull from a branch, if we run `git stash apply`, the changes stored in the temporary stash will be applied to the files. If there is a merge conflict, it will need to be resolved manually.

```
  (use "git restore <file>..." to discard changes in working directory)
        modified:   GOA.py

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

no changes added to commit (use "git add" and/or "git commit -a")
lovebabbar@192 myBTP % git pull
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), 4.89 KiB | 1.63 MiB/s, done.
From https://github.com/lakshayk12/ANN_optimization_BTP
   aa98a93..1896d5f  master     -> origin/master
Updating aa98a93..1896d5f
error: Your local changes to the following files would be overwritten by merge:
        GOA.py
Please commit your changes or stash them before you merge.
Aborting
lovebabbar@192 myBTP % git stash
Saved working directory and index state WIP on master: aa98a93 saving log added
lovebabbar@192 myBTP % git
```

git fetch - -all

To fetch all the things for ex:-
If anyone have created a branch then the name of that branch will displayed in the console...

git push -u origin master

origin means It is base Root From Which a Branches are created....

- Basically pushing a Code on a Master Branch

How to push the Code on Colaboration Repos.... ?

Undo Commits

1. Git commit - - amend

2. Git revert
3. Git reset

How to Undo a Commits....?

1. Git commit - - amend ———→ amend the most recent Commits
2. Git revert ———————→ revert the given commit (with the help of sha ID)
3. Git reset ———————→ It is used to Delete the Commit...

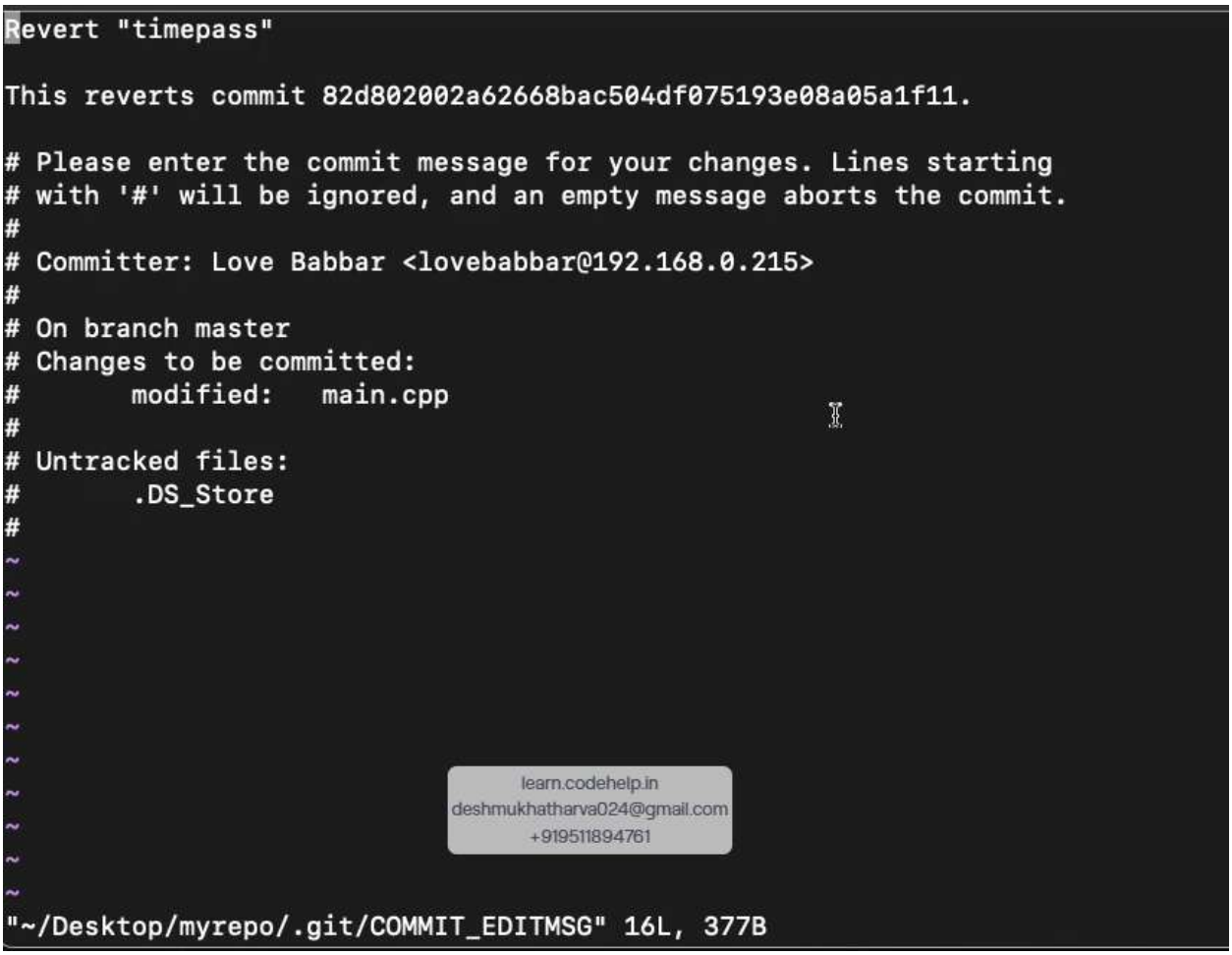
First Case

1. git add .
2. git status
3. git commit -m ‘File Modified’

Now If you want to revert the The Commit that you have made

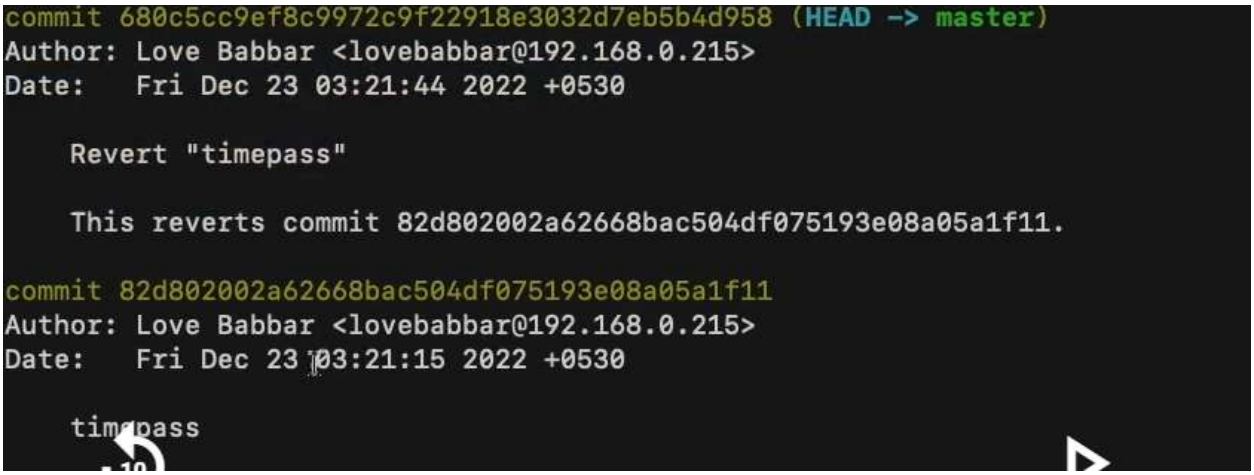
use This Command

git revert 82d802002a62668bac504df075193e08a05a1f11 → (sha ID).....



use this escape + coln(;) + wq + enter

So the Following Changes Will be Reverted....



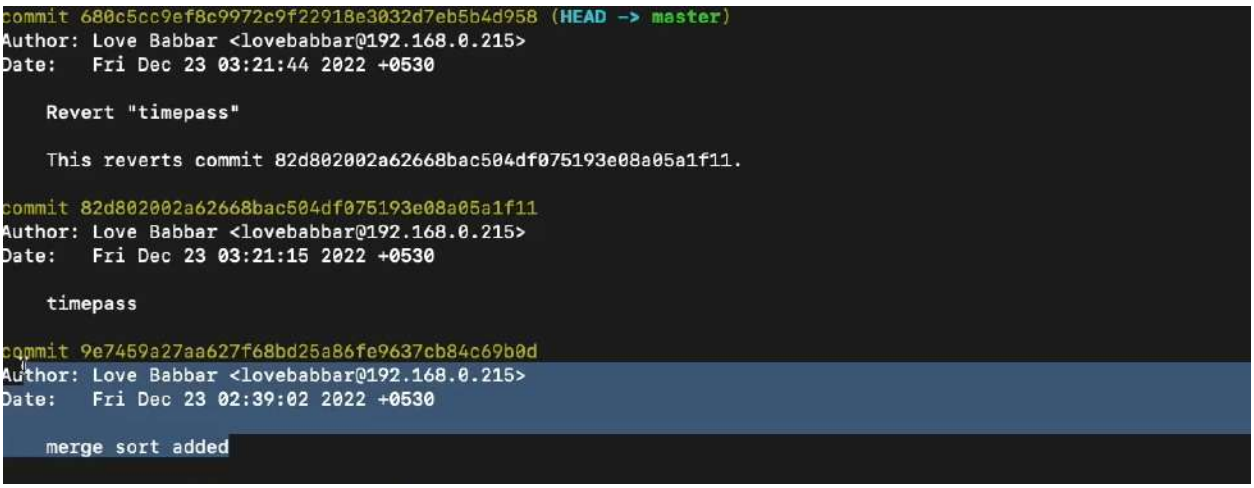
Alternative Option to Git Revert is Git Reset

As Head is Pointing Towards the Master Branch

So the Git Reset Command Does to Do Delete the Whole Commit Only . that we have created by Mistake and also Head Starts Pointing towards the Previous Commit.

Scenario —

1. Firstly Head is Pointing Towards the Master .



2. git reset - - soft 82d802002a62668bac504df075193e08a05a1f11

So After the Execution of the Above commands the following commits are being deleted....

- - - Soft :-
- - - mixed :-
- - - hard :-

When We use the Git Reset Hard the Following changes that were Present in a File at Local Will be Discarded

When we use the Git Reset Mixed the Following Changes that were Present in a File At Local (Working Dir) will be shown as the modifications every Time....

When we use the Git reset Soft the Following Changes that were present in a File at Local It will Shown in the Stage Indexing.

Note :- We Generally Use Soft , mixed and hard is Completely Avoided to Use.

- If we use Hard Then the Changes that are Present in Working Dir Would be Deleted Completely

Git commit - - amend

It will amend (delete) the Most Recent Commit