GIT TOOLS

1.Branch

Branches allow you to develop features, fix bugs, or safely experiment with new ideas in a contained area of your repository.

Two types

- Master branch
- Feature branch

A feature branch is a copy of the main codebase where an individual or team of software developers can work on a new feature until it is complete.

#git branch ---->use to show which branch now (eg. * master) --->* means active branch

#git branch dev ----->branch create

#git checkout dev --->change branch

2.MERGE

Merging is the process of combining the changes in two distinct branches.

merge can done only through matser to dev

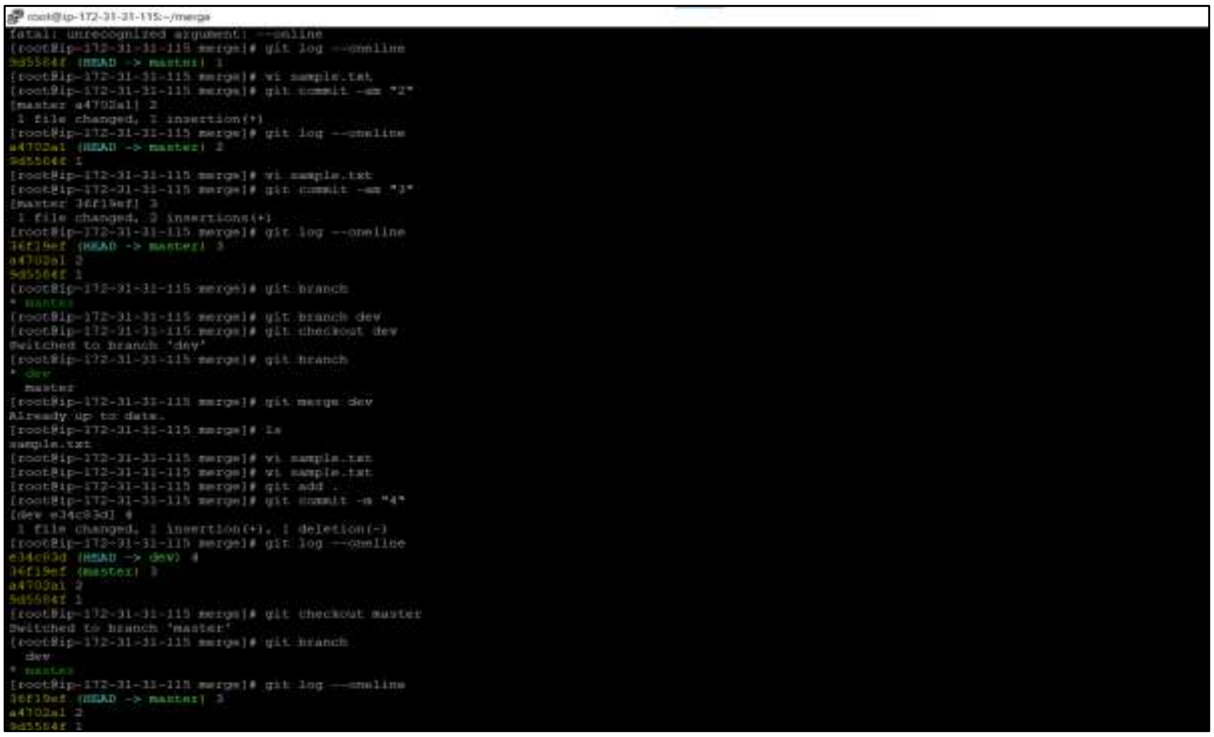
dev to master merge is not possible in merege concept

step1:3 commit in master branch

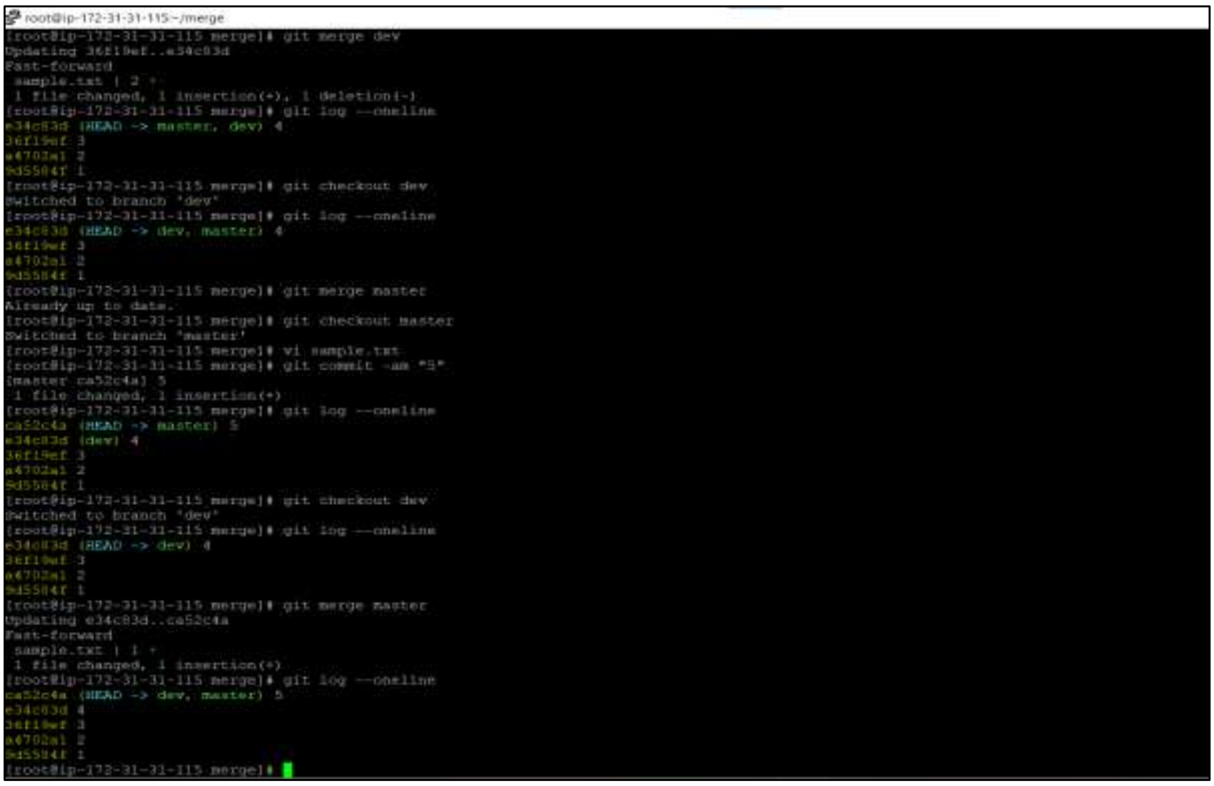
step2:1 commit in dev branch

now merge dev to master --->#git checkout dev--->
#git merge master

merge master to dev--->#git checkout master ---->#git merge dev


root@ip-172-31-31-115:~/merge

```
fatal: unrecognized argument: --online
[root@ip-172-31-31-115 merge]# git log --oneline
9d5584f (HEAD -> master) 1
[root@ip-172-31-31-115 merge]# vi sample.txt
[root@ip-172-31-31-115 merge]# git commit -am "2"
[master a4702a1] 2
 1 file changed, 1 insertion(+)
[root@ip-172-31-31-115 merge]# git log --oneline
a4702a1 (HEAD -> master) 2
9d5584f 1
[root@ip-172-31-31-115 merge]# vi sample.txt
[root@ip-172-31-31-115 merge]# git commit -am "3"
[master 36f19ef] 3
 1 file changed, 2 insertions(+)
[root@ip-172-31-31-115 merge]# git log --oneline
36f19ef (HEAD -> master) 3
a4702a1 2
9d5584f 1
[root@ip-172-31-31-115 merge]# git branch
* master
[root@ip-172-31-31-115 merge]# git branch dev
[root@ip-172-31-31-115 merge]# git checkout dev
Switched to branch 'dev'
[root@ip-172-31-31-115 merge]# git branch
* dev
  master
[root@ip-172-31-31-115 merge]# git merge dev
Already up to date.
[root@ip-172-31-31-115 merge]# ls
sample.txt
[root@ip-172-31-31-115 merge]# vi sample.txt
[root@ip-172-31-31-115 merge]# vi sample.txt
[root@ip-172-31-31-115 merge]# git add .
[root@ip-172-31-31-115 merge]# git commit -m "4"
[dev e34c83d] 4
 1 file changed, 1 insertion(+), 1 deletion(-)
[root@ip-172-31-31-115 merge]# git log --oneline
e34c83d (HEAD -> dev) 4
36f19ef (master) 3
a4702a1 2
9d5584f 1
[root@ip-172-31-31-115 merge]# git checkout master
Switched to branch 'master'
[root@ip-172-31-31-115 merge]# git branch
  dev
* master
[root@ip-172-31-31-115 merge]# git log --oneline
36f19ef (HEAD -> master) 3
a4702a1 2
9d5584f 1
```


root@ip-172-31-31-115:~/merge

```
[root@ip-172-31-31-115 merge]# git merge dev
Updating 36f19ef..e34c83d
Fast-forward
 sample.txt | 2 +-
 1 file changed, 1 insertion(+), 1 deletion(-)
[root@ip-172-31-31-115 merge]# git log --oneline
e34c83d (HEAD -> master, dev) 4
36f19ef 3
a4702a1 2
9d5584f 1
[root@ip-172-31-31-115 merge]# git checkout dev
Switched to branch 'dev'
[root@ip-172-31-31-115 merge]# git log --oneline
e34c83d (HEAD -> dev, master) 4
36f19ef 3
a4702a1 2
9d5584f 1
[root@ip-172-31-31-115 merge]# git merge master
Already up to date.
[root@ip-172-31-31-115 merge]# git checkout master
Switched to branch 'master'
[root@ip-172-31-31-115 merge]# vi sample.txt
[root@ip-172-31-31-115 merge]# git commit -am "5"
[master ca52c4a] 5
 1 file changed, 1 insertion(+)
[root@ip-172-31-31-115 merge]# git log --oneline
ca52c4a (HEAD -> master) 5
e34c83d (dev) 4
36f19ef 3
a4702a1 2
9d5584f 1
[root@ip-172-31-31-115 merge]# git checkout dev
Switched to branch 'dev'
[root@ip-172-31-31-115 merge]# git log --oneline
e34c83d (HEAD -> dev) 4
36f19ef 3
a4702a1 2
9d5584f 1
[root@ip-172-31-31-115 merge]# git merge master
Updating e34c83d..ca52c4a
Fast-forward
 sample.txt | 1 +
 1 file changed, 1 insertion(+)
[root@ip-172-31-31-115 merge]# git log --oneline
ca52c4a (HEAD -> dev, master) 5
e34c83d 4
36f19ef 3
a4702a1 2
9d5584f 1
[root@ip-172-31-31-115 merge]#
```

3.MERGE CONFLICT

A merge conflict can occur when the merged branches edit the same file line differently, or when one branch modifies a file and another branch deletes it.

```
#mkdir merge conflict

#cd merge conflict

#git config --global user.name  “jayaseelan4053”

#git config --global user.email “jayasee4053@gmail.com”

#git config --global core.editor vim

#git config --global core.compression 2

#git config --global diff.tool vim.diff

#git init

#vi index.txt --->add 1

#git add .

#git commit –m “1”

#vi index.txt --->add 2

#git commit –am “2”

#vi index.txt -->add 3 -->#git commit –am “3”

#git branch dev ---->#git checkout dev

#git log –oneline --->it will showns master branch commits

#vi index.txt ----> add 5

#git add .  --->git commit –m “4”

#git checkout master

#vi index.txt --->add 6

#git commit –am “5”

#git checkout dev
```

#git merge master --->shown(Automatic merge failed; fix conflicts and then commit the result.)

Now install git conflict

#git config --global merge.toolvimdiff

#git config --global merge.conflictstyle diff3

#git config --global mergetool.prompt false

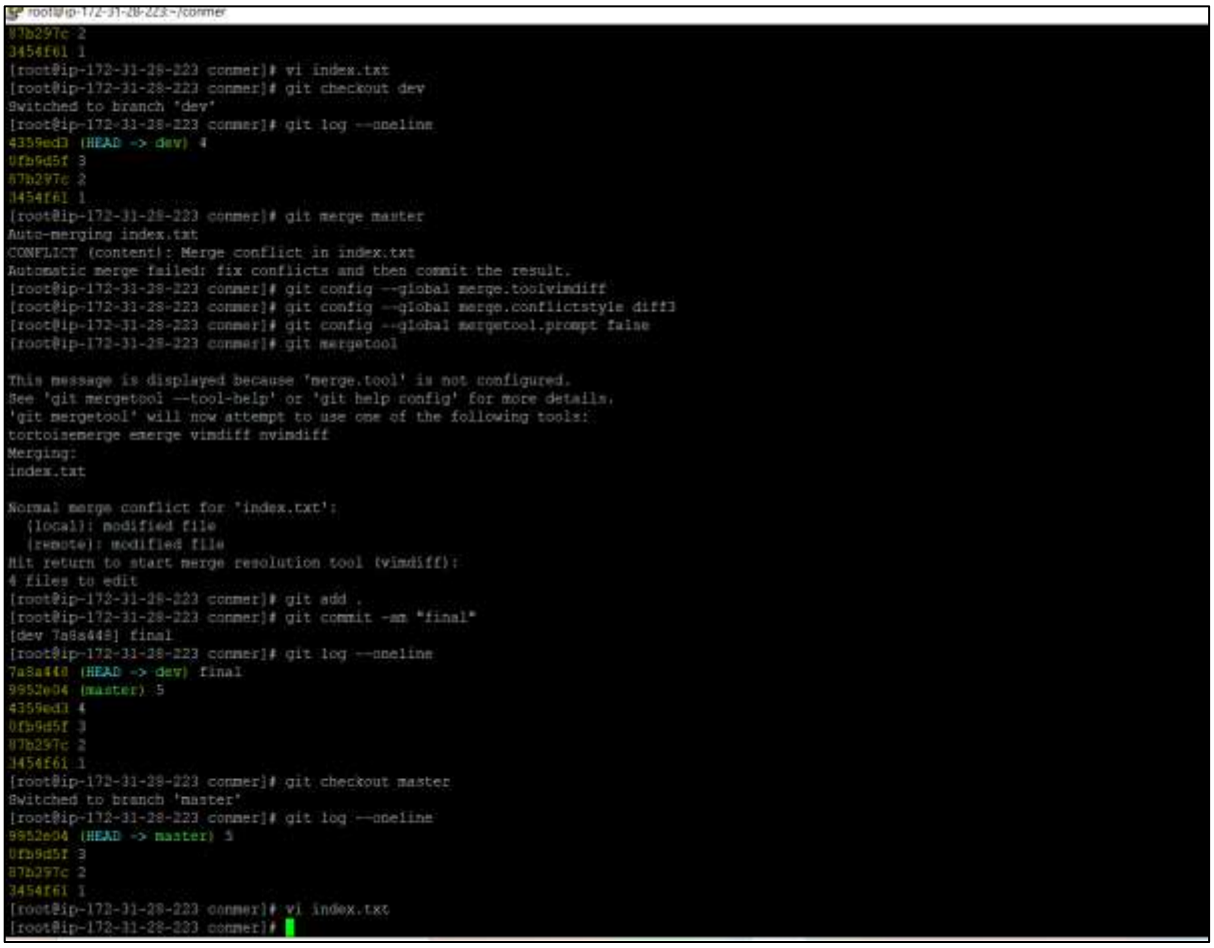
#git mergetool --->enter

Use ctrl+ww --->bottom extra line delete --->save:wqa!

#git add .

#git commit –am “final”

#git log --oneline ----->it will now merged and shown.

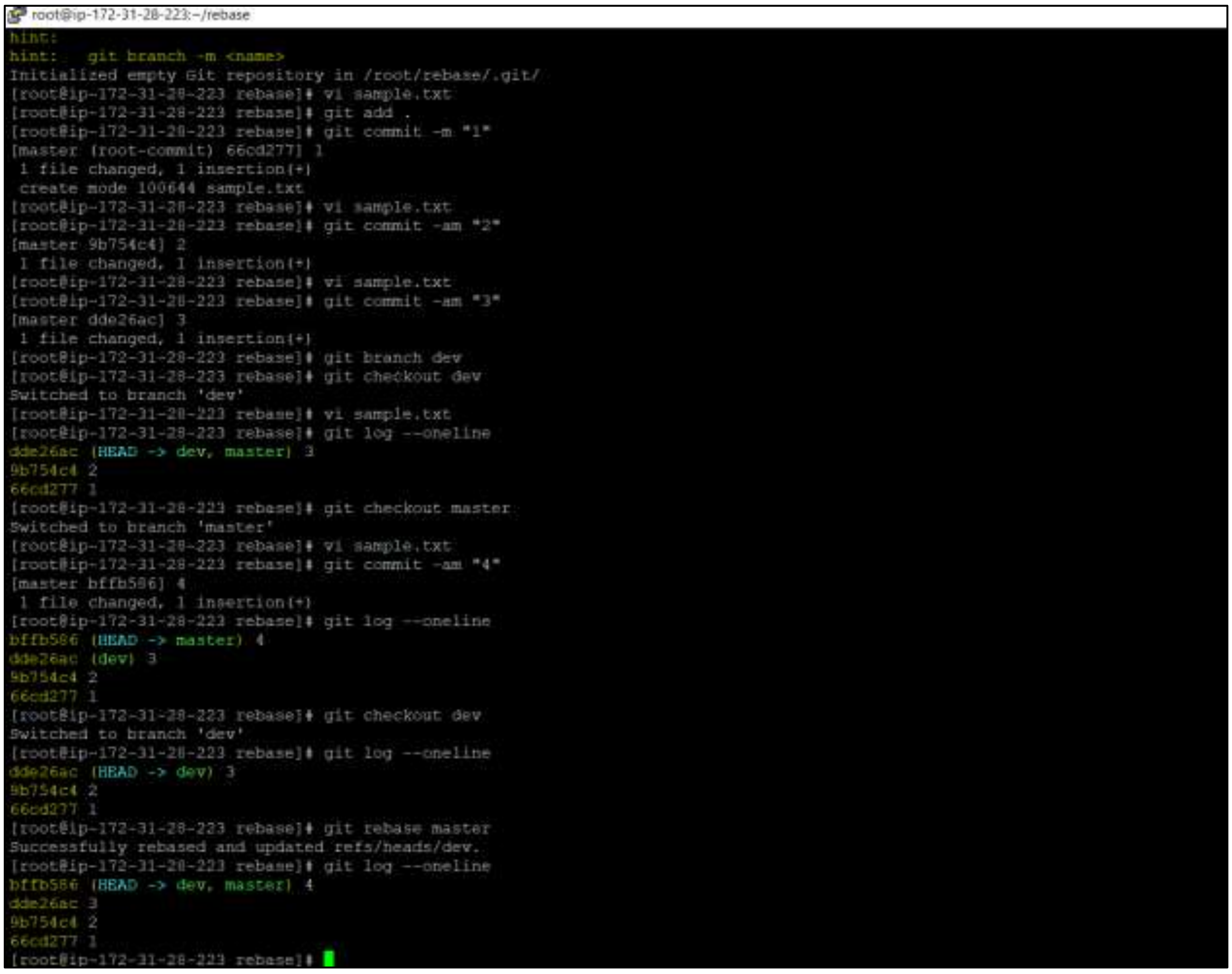


```
root@ip-172-31-28-223:~/conmer
87b297c 2
3454f61 1
[root@ip-172-31-28-223 conmer]# vi index.txt
[root@ip-172-31-28-223 conmer]# git checkout dev
Switched to branch 'dev'
[root@ip-172-31-28-223 conmer]# git log --oneline
4359ed3 (HEAD -> dev) 4
0fb9d5f 3
87b297c 2
3454f61 1
[root@ip-172-31-28-223 conmer]# git merge master
Auto-merging index.txt
CONFLICT (content): Merge conflict in index.txt
Automatic merge failed; fix conflicts and then commit the result.
[root@ip-172-31-28-223 conmer]# git config --global merge.toolvimdiff
[root@ip-172-31-28-223 conmer]# git config --global merge.conflictstyle diff3
[root@ip-172-31-28-223 conmer]# git config --global mergetool.prompt false
[root@ip-172-31-28-223 conmer]# git mergetool

This message is displayed because 'merge.tool' is not configured.
See 'git mergetool --tool-help' or 'git help config' for more details.
'git mergetool' will now attempt to use one of the following tools:
tortoisemerge emerge vimdiff nvimdiff
Merging:
index.txt

Normal merge conflict for 'index.txt':
  {local}: modified file
  {remote}: modified file
Hit return to start merge resolution tool (vimdiff):
4 files to edit
[root@ip-172-31-28-223 conmer]# git add .
[root@ip-172-31-28-223 conmer]# git commit -am "final"
[dev 7a8a448] final
[root@ip-172-31-28-223 conmer]# git log --oneline
7a8a448 (HEAD -> dev) final
9952e04 (master) 5
4359ed3 4
0fb9d5f 3
87b297c 2
3454f61 1
[root@ip-172-31-28-223 conmer]# git checkout master
Switched to branch 'master'
[root@ip-172-31-28-223 conmer]# git log --oneline
9952e04 (HEAD -> master) 5
0fb9d5f 3
87b297c 2
3454f61 1
[root@ip-172-31-28-223 conmer]# vi index.txt
[root@ip-172-31-28-223 conmer]#
```

4.REBASE

Rebasing is the process of moving or combining a sequence of commits to a new base commit.

- Master branch --->3 commit add
- Then checkout dev branch --->it will show master branch commits(git log –oneline)
- Then checkout master branch and another commit in master branch
- Now checkout dev branch --->#git rebase master --->#git log –oneline(it will show all commits include master)

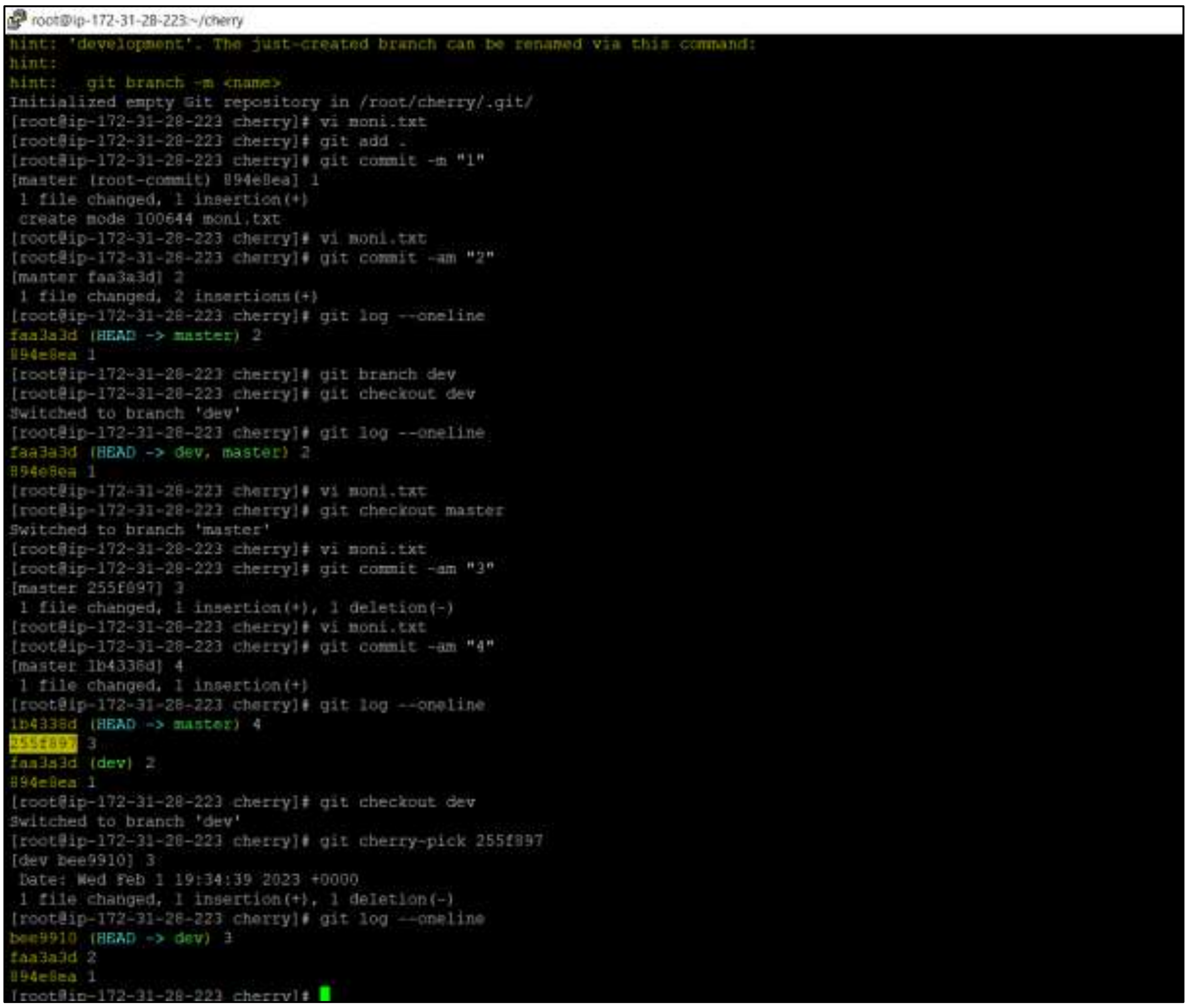


```
root@ip-172-31-28-223:~/rebase
hint:
hint:   git branch -m <name>
Initialized empty Git repository in /root/rebase/.git/
[root@ip-172-31-28-223 rebase]# vi sample.txt
[root@ip-172-31-28-223 rebase]# git add .
[root@ip-172-31-28-223 rebase]# git commit -m "1"
[master (root-commit) 66cd277] 1
 1 file changed, 1 insertion(+)
 create mode 100644 sample.txt
[root@ip-172-31-28-223 rebase]# vi sample.txt
[root@ip-172-31-28-223 rebase]# git commit -am "2"
[master 9b754c4] 2
 1 file changed, 1 insertion(+)
[root@ip-172-31-28-223 rebase]# vi sample.txt
[root@ip-172-31-28-223 rebase]# git commit -am "3"
[master dde26ac] 3
 1 file changed, 1 insertion(+)
[root@ip-172-31-28-223 rebase]# git branch dev
[root@ip-172-31-28-223 rebase]# git checkout dev
Switched to branch 'dev'
[root@ip-172-31-28-223 rebase]# vi sample.txt
[root@ip-172-31-28-223 rebase]# git log --oneline
dde26ac (HEAD -> dev, master) 3
9b754c4 2
66cd277 1
[root@ip-172-31-28-223 rebase]# git checkout master
Switched to branch 'master'
[root@ip-172-31-28-223 rebase]# vi sample.txt
[root@ip-172-31-28-223 rebase]# git commit -am "4"
[master bffb586] 4
 1 file changed, 1 insertion(+)
[root@ip-172-31-28-223 rebase]# git log --oneline
bffb586 (HEAD -> master) 4
dde26ac (dev) 3
9b754c4 2
66cd277 1
[root@ip-172-31-28-223 rebase]# git checkout dev
Switched to branch 'dev'
[root@ip-172-31-28-223 rebase]# git log --oneline
dde26ac (HEAD -> dev) 3
9b754c4 2
66cd277 1
[root@ip-172-31-28-223 rebase]# git rebase master
Successfully rebased and updated refs/heads/dev.
[root@ip-172-31-28-223 rebase]# git log --oneline
bffb586 (HEAD -> dev, master) 4
dde26ac 3
9b754c4 2
66cd277 1
[root@ip-172-31-28-223 rebase]#
```

5.CHERRYPICK

Git cherry-pick applies the changes from specific commits from one branch to another branch.

- 2 commit add master branch
- Change dev branch
- Now 3 commit add master branch
- Now 5th commit only I need show that only take master to dev
- #git checkout dev
- #git cherry-pick (commit ip)-->that particular commit only add dev branch



```
root@ip-172-31-28-223:~/cherry
hint: 'development'. The just-created branch can be renamed via this command:
hint:
hint:   git branch -m <name>
Initialized empty Git repository in /root/cherry/.git/
[root@ip-172-31-28-223 cherry]# vi moni.txt
[root@ip-172-31-28-223 cherry]# git add .
[root@ip-172-31-28-223 cherry]# git commit -m "1"
[master (root-commit) 894e8ea] 1
 1 file changed, 1 insertion(+)
 create mode 100644 moni.txt
[root@ip-172-31-28-223 cherry]# vi moni.txt
[root@ip-172-31-28-223 cherry]# git commit -am "2"
[master faa3a3d] 2
 1 file changed, 2 insertions(+)
[root@ip-172-31-28-223 cherry]# git log --oneline
faa3a3d (HEAD -> master) 2
894e8ea 1
[root@ip-172-31-28-223 cherry]# git branch dev
[root@ip-172-31-28-223 cherry]# git checkout dev
Switched to branch 'dev'
[root@ip-172-31-28-223 cherry]# git log --oneline
faa3a3d (HEAD -> dev, master) 2
894e8ea 1
[root@ip-172-31-28-223 cherry]# vi moni.txt
[root@ip-172-31-28-223 cherry]# git checkout master
Switched to branch 'master'
[root@ip-172-31-28-223 cherry]# vi moni.txt
[root@ip-172-31-28-223 cherry]# git commit -am "3"
[master 255f897] 3
 1 file changed, 1 insertion(+), 1 deletion(-)
[root@ip-172-31-28-223 cherry]# vi moni.txt
[root@ip-172-31-28-223 cherry]# git commit -am "4"
[master 1b4338d] 4
 1 file changed, 1 insertion(+)
[root@ip-172-31-28-223 cherry]# git log --oneline
1b4338d (HEAD -> master) 4
255f897 3
faa3a3d (dev) 2
894e8ea 1
[root@ip-172-31-28-223 cherry]# git checkout dev
Switched to branch 'dev'
[root@ip-172-31-28-223 cherry]# git cherry-pick 255f897
[dev bee9910] 3
 Date: Wed Feb 1 19:34:39 2023 +0000
 1 file changed, 1 insertion(+), 1 deletion(-)
[root@ip-172-31-28-223 cherry]# git log --oneline
bee9910 (HEAD -> dev) 3
faa3a3d 2
894e8ea 1
[root@ip-172-31-28-223 cherry]#
```

6.STASH

Temporary location to save your working file.

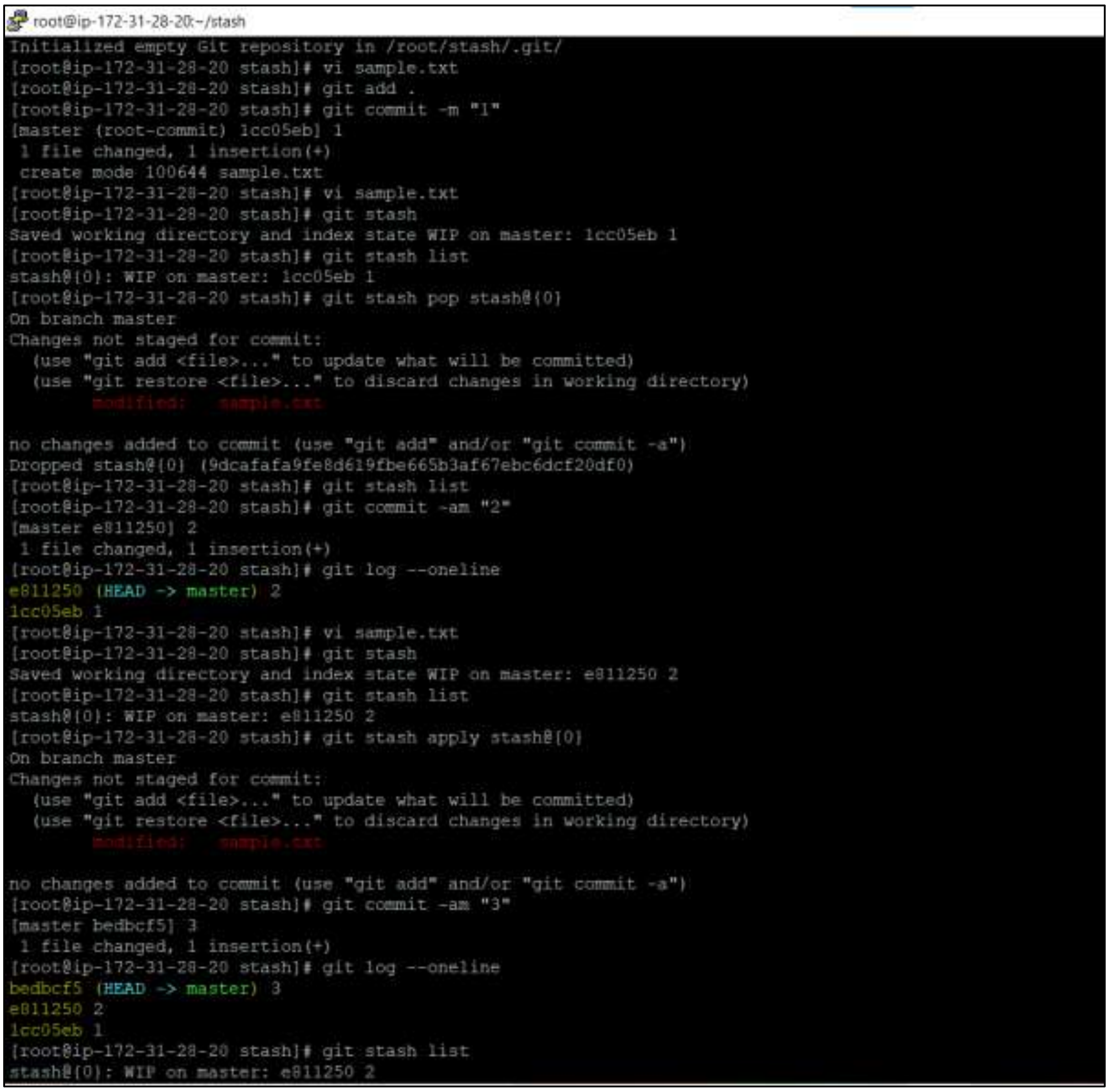
Two types

- Pop ------>cut & paste
- Apply ---->copy & paste

```
1.#vi sample.txt --->add 1

 #git add .

 #git commit –m "1"

 # vi sample.txt --->add 2 --->:wq!

 #git stash ---->#git stash list

 #git stash pop stash@{0}
```

Stash id

```
 #git commit –am "2"

 #git log --oneline

 #git stash list ---->can't show because pop it means cut stash and commit


2.# vi sample.txt --->add 3 --->:wq!

 #git stash ---->#git stash list

 #git stash apply stash@{0}

 #git commit –am "3"

  #git log --oneline

  #git stash list ---->show stash because apply  it means copy stash and
commit
```



```
root@ip-172-31-28-20:~/stash
Initialized empty Git repository in /root/stash/.git/
[root@ip-172-31-28-20 stash]# vi sample.txt
[root@ip-172-31-28-20 stash]# git add .
[root@ip-172-31-28-20 stash]# git commit -m "1"
[master (root-commit) 1cc05eb] 1
 1 file changed, 1 insertion(+)
 create mode 100644 sample.txt
[root@ip-172-31-28-20 stash]# vi sample.txt
[root@ip-172-31-28-20 stash]# git stash
Saved working directory and index state WIP on master: 1cc05eb 1
[root@ip-172-31-28-20 stash]# git stash list
stash@{0}: WIP on master: 1cc05eb 1
[root@ip-172-31-28-20 stash]# git stash pop stash@{0}
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:   sample.txt

no changes added to commit (use "git add" and/or "git commit -a")
Dropped stash@{0} (9dcafafa9fe8d619fbe665b3af67ebc6dcf20df0)
[root@ip-172-31-28-20 stash]# git stash list
[root@ip-172-31-28-20 stash]# git commit -am "2"
[master e811250] 2
 1 file changed, 1 insertion(+)
[root@ip-172-31-28-20 stash]# git log --oneline
e811250 (HEAD -> master) 2
1cc05eb 1
[root@ip-172-31-28-20 stash]# vi sample.txt
[root@ip-172-31-28-20 stash]# git stash
Saved working directory and index state WIP on master: e811250 2
[root@ip-172-31-28-20 stash]# git stash list
stash@{0}: WIP on master: e811250 2
[root@ip-172-31-28-20 stash]# git stash apply stash@{0}
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:   sample.txt

no changes added to commit (use "git add" and/or "git commit -a")
[root@ip-172-31-28-20 stash]# git commit -am "3"
[master bedbcf5] 3
 1 file changed, 1 insertion(+)
[root@ip-172-31-28-20 stash]# git log --oneline
bedbcf5 (HEAD -> master) 3
e811250 2
1cc05eb 1
[root@ip-172-31-28-20 stash]# git stash list
stash@{0}: WIP on master: e811250 2
```