

1) Create Merge Conflict Scenario and resolve Merge Conflicts with 2 branches changed using same file

```
MINGW64~/Downloads/devops/sandeep/demo
sandeep@torlikonda\SandeepTorlikonda MINGW64 ~/Downloads/devops/sandeep/demo (feature_b)
$ git commit -m "Change in branch B"
[feature_b 9bf36b] Change in branch B
1 file changed, 1 insertion(+)

sandeep@torlikonda\SandeepTorlikonda MINGW64 ~/Downloads/devops/sandeep/demo (feature_b)
$ git commit -m "Change in branch B"
on branch feature_b
nothing to commit, working tree clean

sandeep@torlikonda\SandeepTorlikonda MINGW64 ~/Downloads/devops/sandeep/demo (feature_b)
$ git commit -m "Change in branch B"
on branch feature_b
nothing to commit, working tree clean

sandeep@torlikonda\SandeepTorlikonda MINGW64 ~/Downloads/devops/sandeep/demo (feature_b)
$ git merge feature_A
Auto-merging demo.txt
CONFLICT (content): Merge conflict in demo.txt
Automatic merge failed; fix conflicts and then commit the result.

sandeep@torlikonda\SandeepTorlikonda MINGW64 ~/Downloads/devops/sandeep/demo (feature_b|MERGING)
$ git merge feature_A
error: Merging is not possible because you have unmerged files.
hint: Fix them up in the work tree, and then use 'git add/rm <file>'
hint: as appropriate to mark resolution and make a commit.
fatal: Exiting because of an unresolved conflict.

sandeep@torlikonda\SandeepTorlikonda MINGW64 ~/Downloads/devops/sandeep/demo (feature_b|MERGING)
$ =====
Change made in Branch B.
=====
Change made in Branch A.
>>>>>> branch_A
bash: syntax error near unexpected token '<<<'
bash: Change: command not found
bash: =====: command not found
bash: Change: command not found
bash: syntax error near unexpected token '>>>'

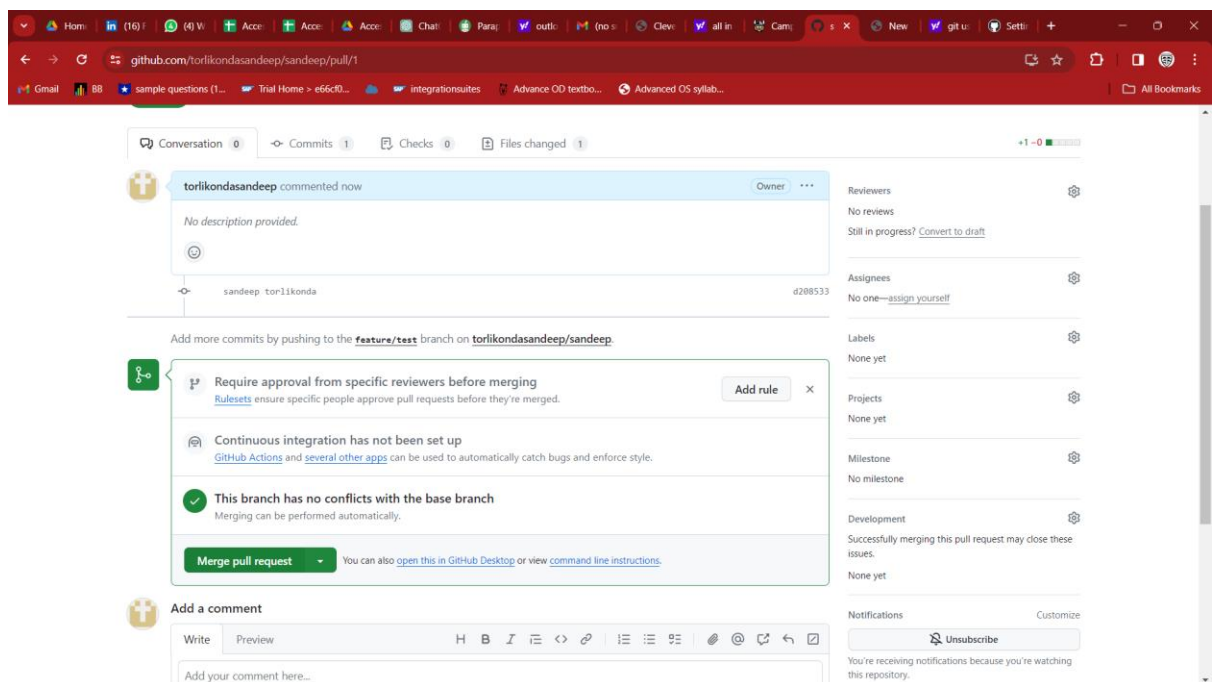
sandeep@torlikonda\SandeepTorlikonda MINGW64 ~/Downloads/devops/sandeep/demo (feature_b|MERGING)
$ git add demo.txt

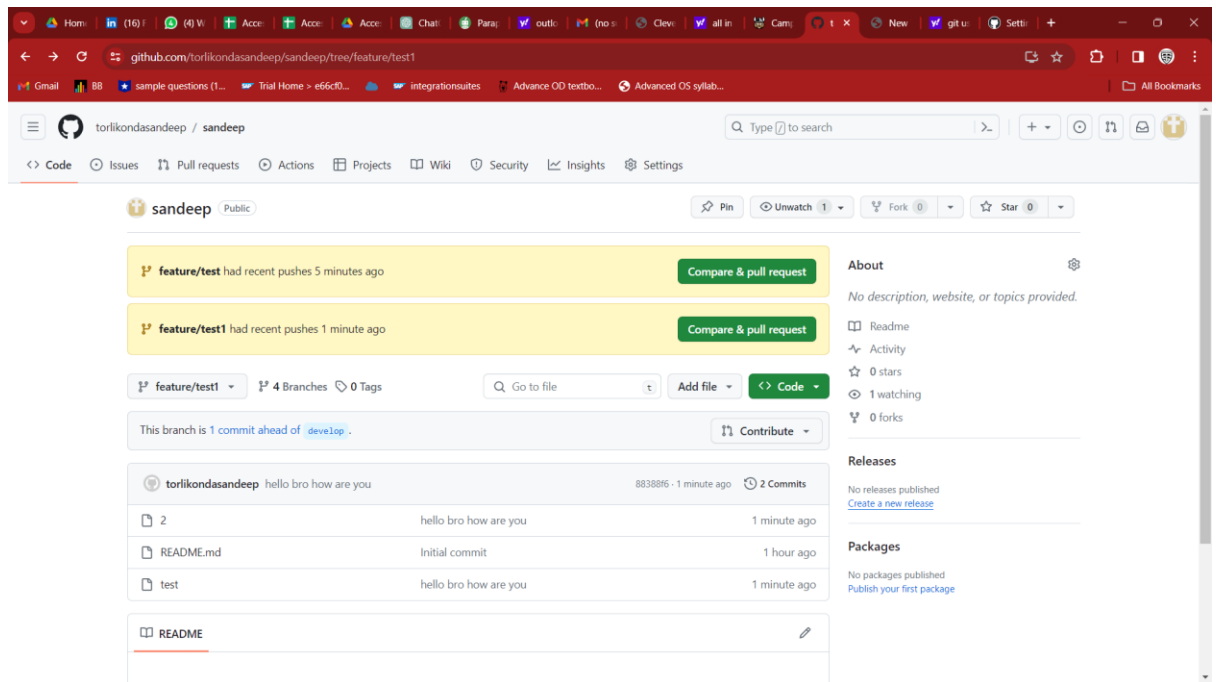
sandeep@torlikonda\SandeepTorlikonda MINGW64 ~/Downloads/devops/sandeep/demo (feature_b|MERGING)
$ git commit -m "resolved"
[feature_b 34fea05] resolved

sandeep@torlikonda\SandeepTorlikonda MINGW64 ~/Downloads/devops/sandeep/demo (feature_b)
$ git checkout master
Switched to branch 'master'

sandeep@torlikonda\SandeepTorlikonda MINGW64 ~/Downloads/devops/sandeep/demo (master)
$ git merge feature_B
Updating 233df85..34fea05
Fast-forward
 demo.txt | 5 +++++
 1 file changed, 5 insertions(+)

sandeep@torlikonda\SandeepTorlikonda MINGW64 ~/Downloads/devops/sandeep/demo (master)
$
```





2) Merge vs rebase (observer commit log)

Merging preserves the entire history of your repository by creating new commit, while rebasing creates a linear history by moving your feature branch onto the tip of main.

3) git pull vs fetch

git pull : is used to fetch and integrate changes from a remote repository into the current branch

Syntax: git pull [options] [remote] [branch]

Git fetch: It doesn't get files transferred, it just checks if there are new changes