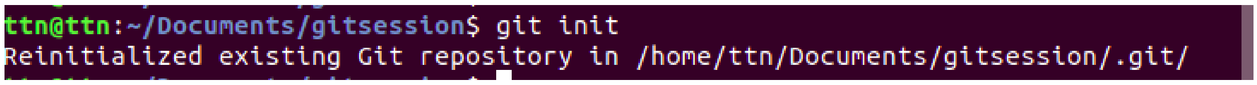
1. Git Setup https://confluence.atlassian.com/bitbucket/set-up-git-744723531.html

sudo apt-get update

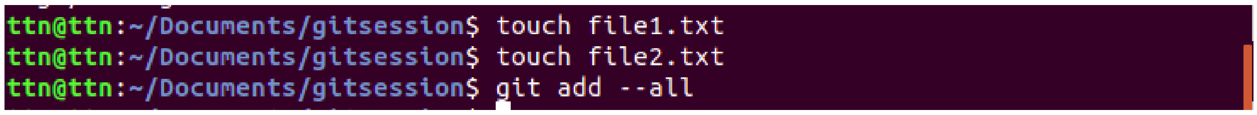
sudo apt-get install git

2. Initialize a Git Repository



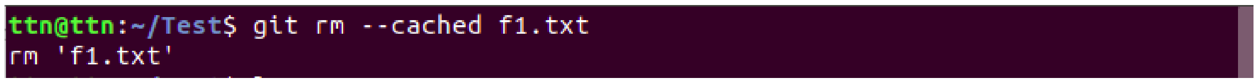
```
ttn@ttn:~/Documents/gitsession$ git init
Reinitialized existing Git repository in /home/ttn/Documents/gitsession/.git/
```

3. Add files to the repository



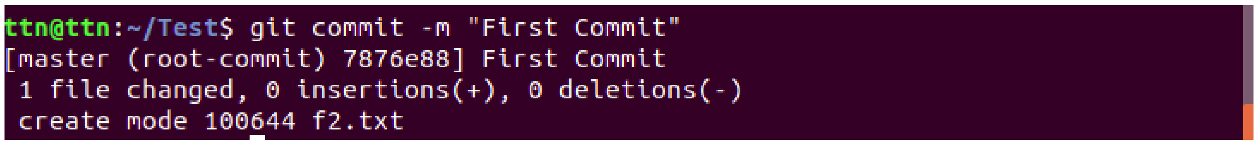
```
ttn@ttn:~/Documents/gitsession$ touch file1.txt
ttn@ttn:~/Documents/gitsession$ touch file2.txt
ttn@ttn:~/Documents/gitsession$ git add --all
```

4. Unstage 1 file



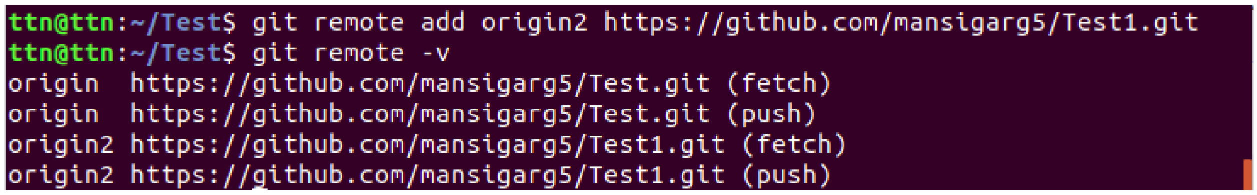
```
ttn@ttn:~/Test$ git rm --cached f1.txt
rm 'f1.txt'
```

5. Commit the file



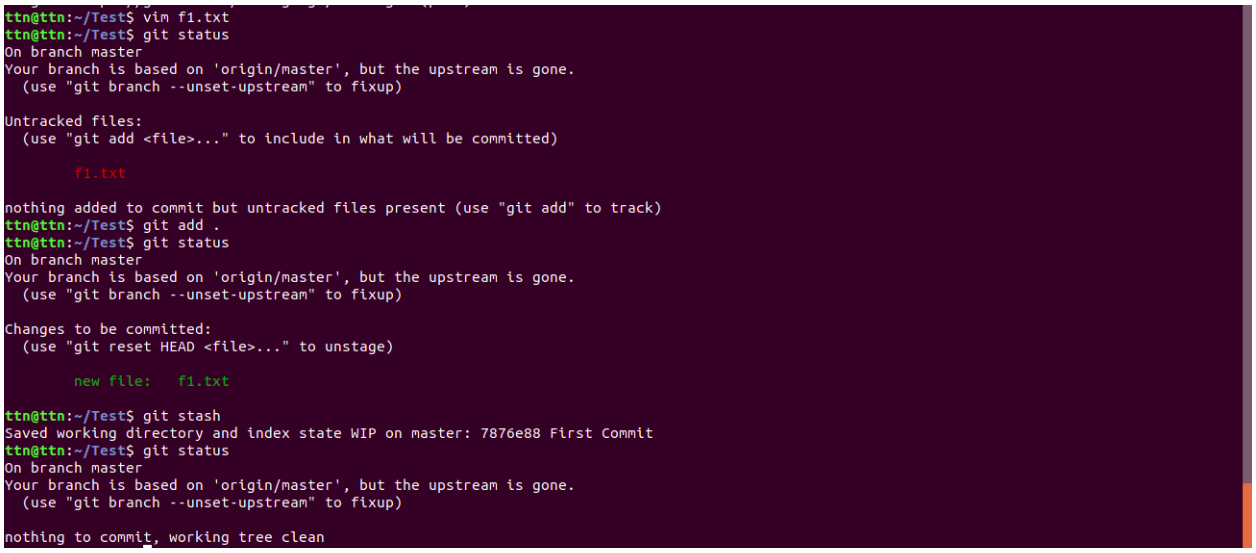
```
ttn@ttn:~/Test$ git commit -m "First Commit"
[master (root-commit) 7876e88] First Commit
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 f2.txt
```

6. Add a remote



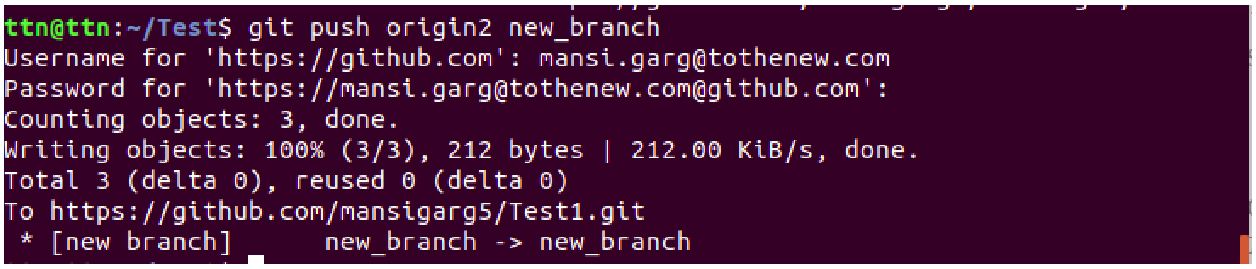
```
ttn@ttn:~/Test$ git remote add origin2 https://github.com/mansigarg5/Test1.git
ttn@ttn:~/Test$ git remote -v
origin  https://github.com/mansigarg5/Test.git (fetch)
origin  https://github.com/mansigarg5/Test.git (push)
origin2 https://github.com/mansigarg5/Test1.git (fetch)
origin2 https://github.com/mansigarg5/Test1.git (push)
```

7. Undo changes to a particular file



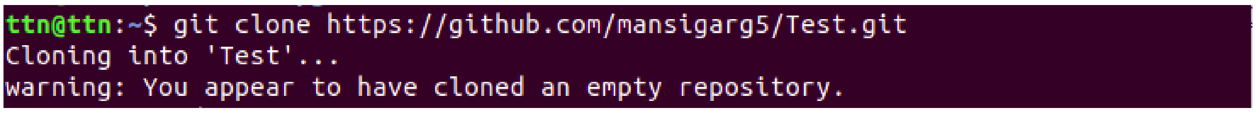
```
ttn@ttn:~/Test$ vim f1.txt
ttn@ttn:~/Test$ git status
On branch master
Your branch is based on 'origin/master', but the upstream is gone.
  (use "git branch --unset-upstream" to fixup)

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

        f1.txt

nothing added to commit but untracked files present (use "git add" to track)
ttn@ttn:~/Test$ git add .
ttn@ttn:~/Test$ git status
On branch master
Your branch is based on 'origin/master', but the upstream is gone.
  (use "git branch --unset-upstream" to fixup)

Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

        new file:   f1.txt

ttn@ttn:~/Test$ git stash
Saved working directory and index state WIP on master: 7876e88 First Commit
ttn@ttn:~/Test$ git status
On branch master
Your branch is based on 'origin/master', but the upstream is gone.
  (use "git branch --unset-upstream" to fixup)

nothing to commit, working tree clean
```

8. Push changes to Github



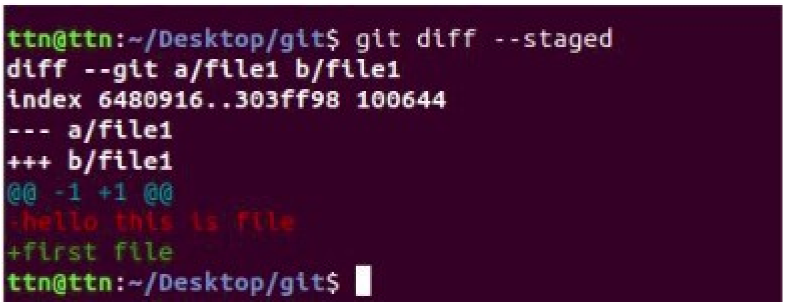
```
ttn@ttn:~/Test$ git push origin2 new_branch
Username for 'https://github.com': mansi.garg@tothenew.com
Password for 'https://mansi.garg@tothenew.com@github.com':
Counting objects: 3, done.
Writing objects: 100% (3/3), 212 bytes | 212.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To https://github.com/mansigarg5/Test1.git
 * [new branch]      new_branch -> new_branch
```

9. Clone the repository



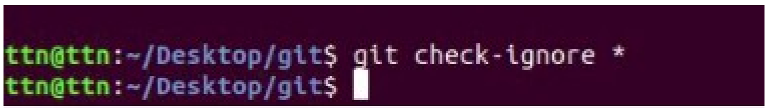
```
ttn@ttn:~$ git clone https://github.com/mansigarg5/Test.git
Cloning into 'Test'...
warning: You appear to have cloned an empty repository.
```

11. Check differences between a file and its staged version



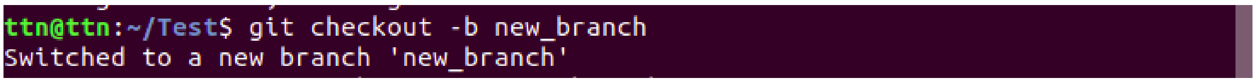
```
ttn@ttn:~/Desktop/git$ git diff --staged
diff --git a/file1 b/file1
index 6480916..303ff98 100644
--- a/file1
+++ b/file1
@@ -1 +1 @@
-hello this is file
+first file
ttn@ttn:~/Desktop/git$
```

12. Ignore a few files to be checked in



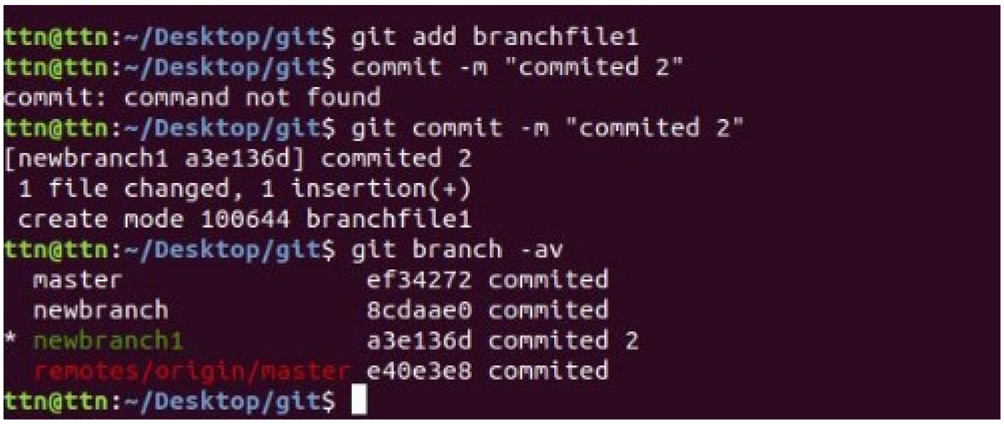
```
ttn@ttn:~/Desktop/git$ git check-ignore *
ttn@ttn:~/Desktop/git$
```

13. Create a new branch.



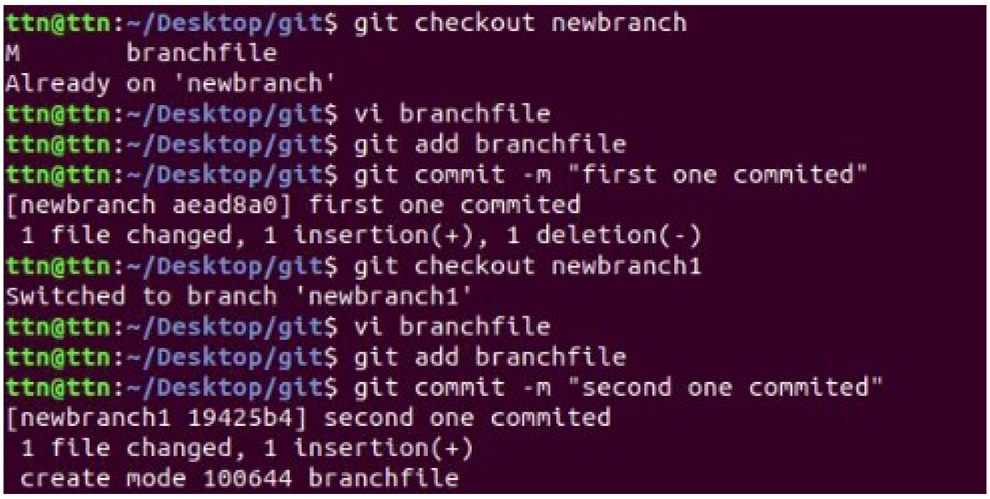
```
ttn@ttn:~/Test$ git checkout -b new_branch
Switched to a new branch 'new_branch'
```

14. Diverge them with commits



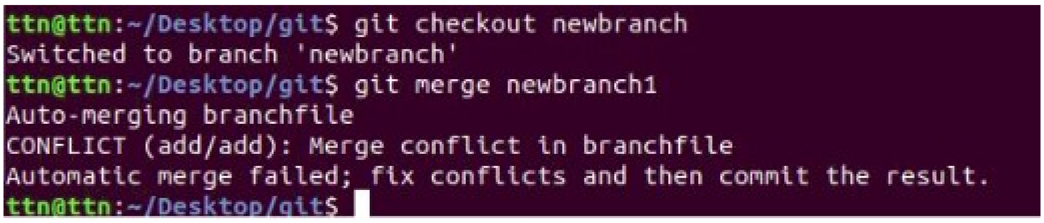
```
ttn@ttn:~/Desktop/git$ git add branchfile1
ttn@ttn:~/Desktop/git$ commit -m "commited 2"
commit: command not found
ttn@ttn:~/Desktop/git$ git commit -m "commited 2"
[newbranch1 a3e136d] commited 2
 1 file changed, 1 insertion(+)
 create mode 100644 branchfile1
ttn@ttn:~/Desktop/git$ git branch -av
  master                 ef34272 commited
  newbranch              8cdaae0 commited
* newbranch1             a3e136d commited 2
  remotes/origin/master  e40e3e8 commited
ttn@ttn:~/Desktop/git$ 
```

15. Edit the same file at the same line on both branches and commit

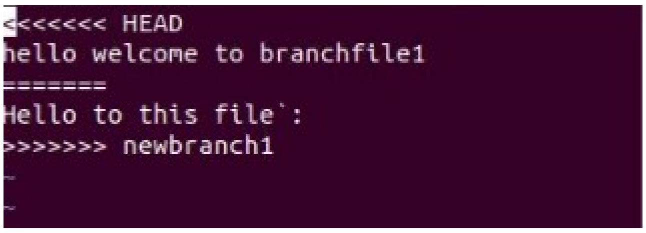


```
ttn@ttn:~/Desktop/git$ git checkout newbranch
M       branchfile
Already on 'newbranch'
ttn@ttn:~/Desktop/git$ vi branchfile
ttn@ttn:~/Desktop/git$ git add branchfile
ttn@ttn:~/Desktop/git$ git commit -m "first one commited"
[newbranch aead8a0] first one commited
 1 file changed, 1 insertion(+), 1 deletion(-)
ttn@ttn:~/Desktop/git$ git checkout newbranch1
Switched to branch 'newbranch1'
ttn@ttn:~/Desktop/git$ vi branchfile
ttn@ttn:~/Desktop/git$ git add branchfile
ttn@ttn:~/Desktop/git$ git commit -m "second one commited"
[newbranch1 19425b4] second one commited
 1 file changed, 1 insertion(+)
 create mode 100644 branchfile
```

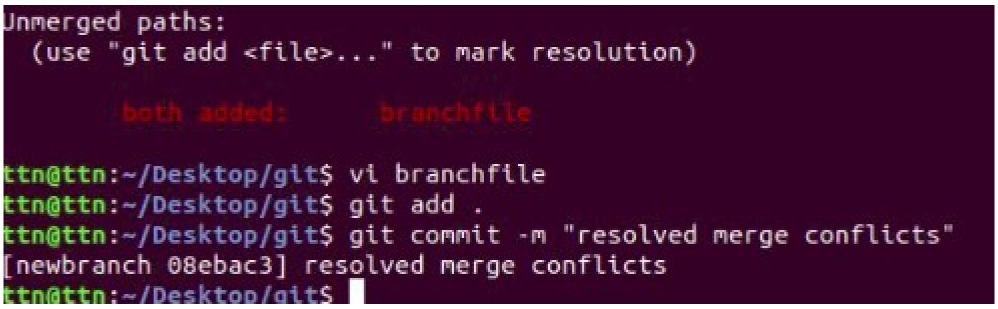
16. Try merging and resolve merge conflicts



```
ttn@ttn:~/Desktop/git$ git checkout newbranch
Switched to branch 'newbranch'
ttn@ttn:~/Desktop/git$ git merge newbranch1
Auto-merging branchfile
CONFLICT (add/add): Merge conflict in branchfile
Automatic merge failed; fix conflicts and then commit the result.
ttn@ttn:~/Desktop/git$
```

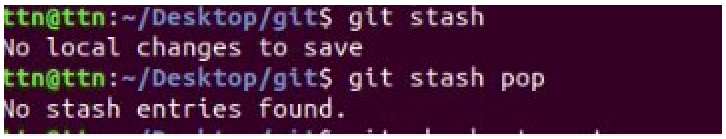


```
<<<<<<< HEAD
hello welcome to branchfile1
=======
Hello to this file`:
>>>>>>> newbranch1
~
~
```



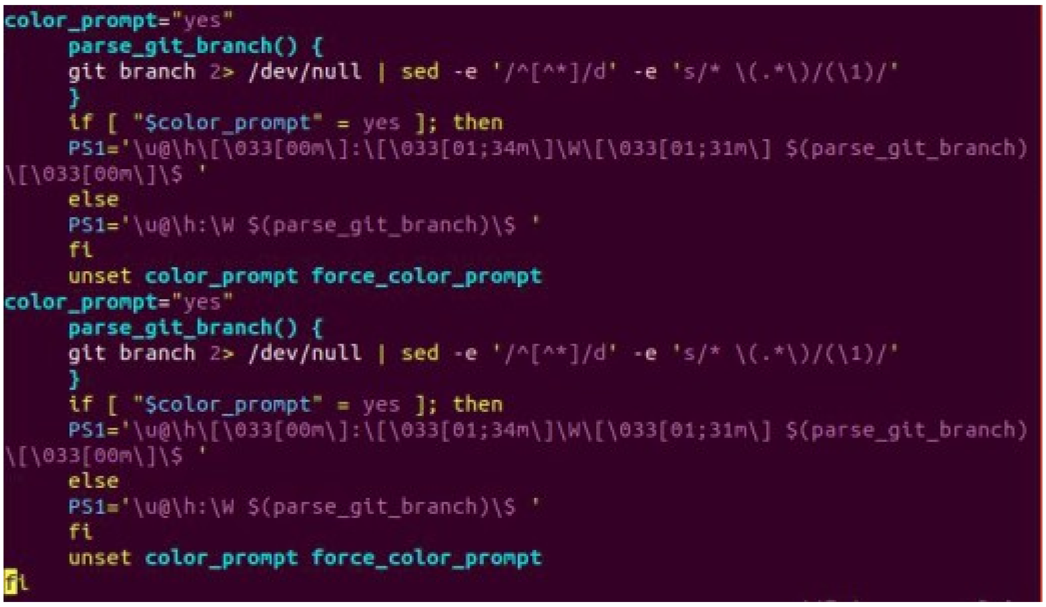
```
Unmerged paths:
  (use "git add <file>..." to mark resolution)

        both added:      branchfile

ttn@ttn:~/Desktop/git$ vi branchfile
ttn@ttn:~/Desktop/git$ git add .
ttn@ttn:~/Desktop/git$ git commit -m "resolved merge conflicts"
[newbranch 08ebac3] resolved merge conflicts
ttn@ttn:~/Desktop/git$
```

17. Stash the changes and pop them



```
ttn@ttn:~/Desktop/git$ git stash
No local changes to save
ttn@ttn:~/Desktop/git$ git stash pop
No stash entries found.
```

18. Add the following code to your .bashrc file



```
color_prompt="yes"
     parse_git_branch() {
     git branch 2> /dev/null | sed -e '/^[^*]/d' -e 's/* \(.*\)/(\1)/'
     }
     if [ "$color_prompt" = yes ]; then
     PS1='\u@\h\[\033[00m\]:\[\033[01;34m\]\W\[\033[01;31m\] $(parse_git_branch)
\[\033[00m\]\$ '
     else
     PS1='\u@\h:\W $(parse_git_branch)\$ '
     fi
     unset color_prompt force_color_prompt
color_prompt="yes"
     parse_git_branch() {
     git branch 2> /dev/null | sed -e '/^[^*]/d' -e 's/* \(.*\)/(\1)/'
     }
     if [ "$color_prompt" = yes ]; then
     PS1='\u@\h\[\033[00m\]:\[\033[01;34m\]\W\[\033[01;31m\] $(parse_git_branch)
\[\033[00m\]\$ '
     else
     PS1='\u@\h:\W $(parse_git_branch)\$ '
     fi
     unset color_prompt force_color_prompt
fi
```