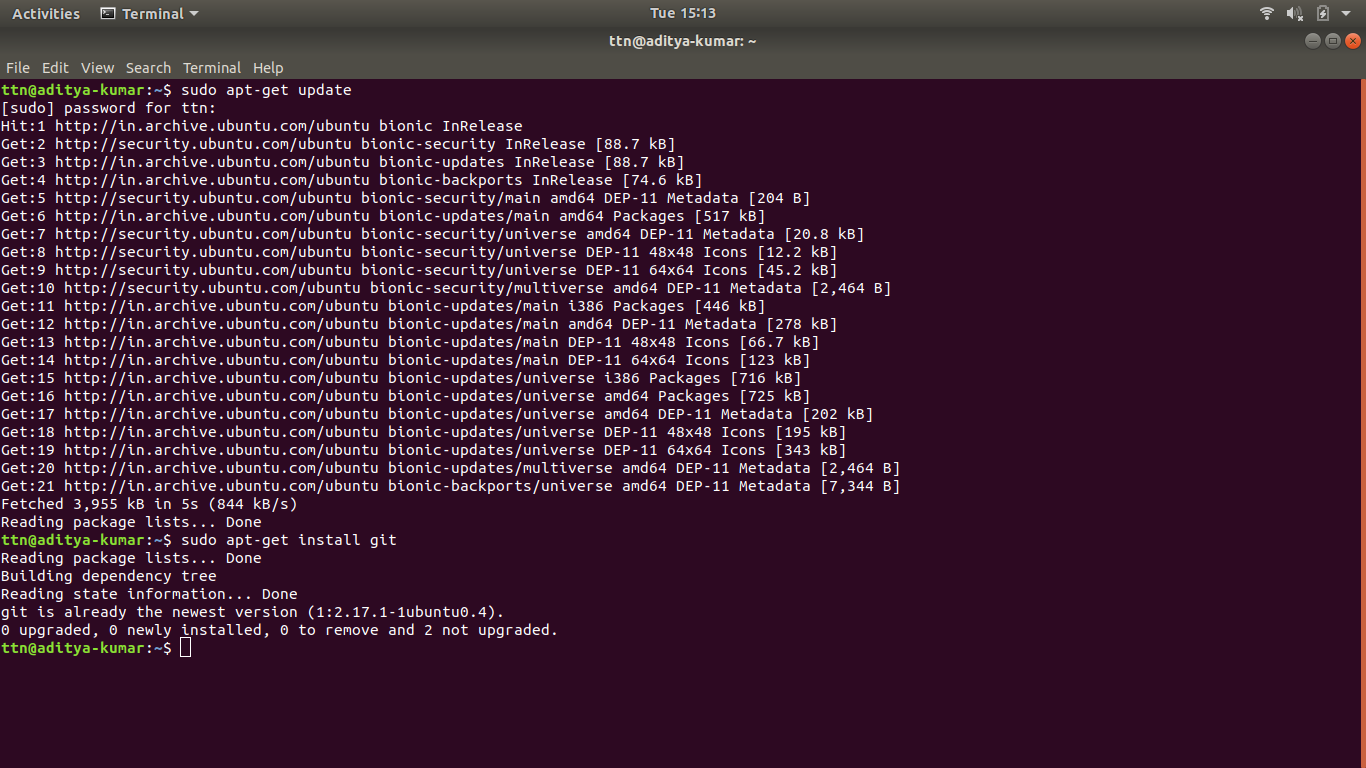
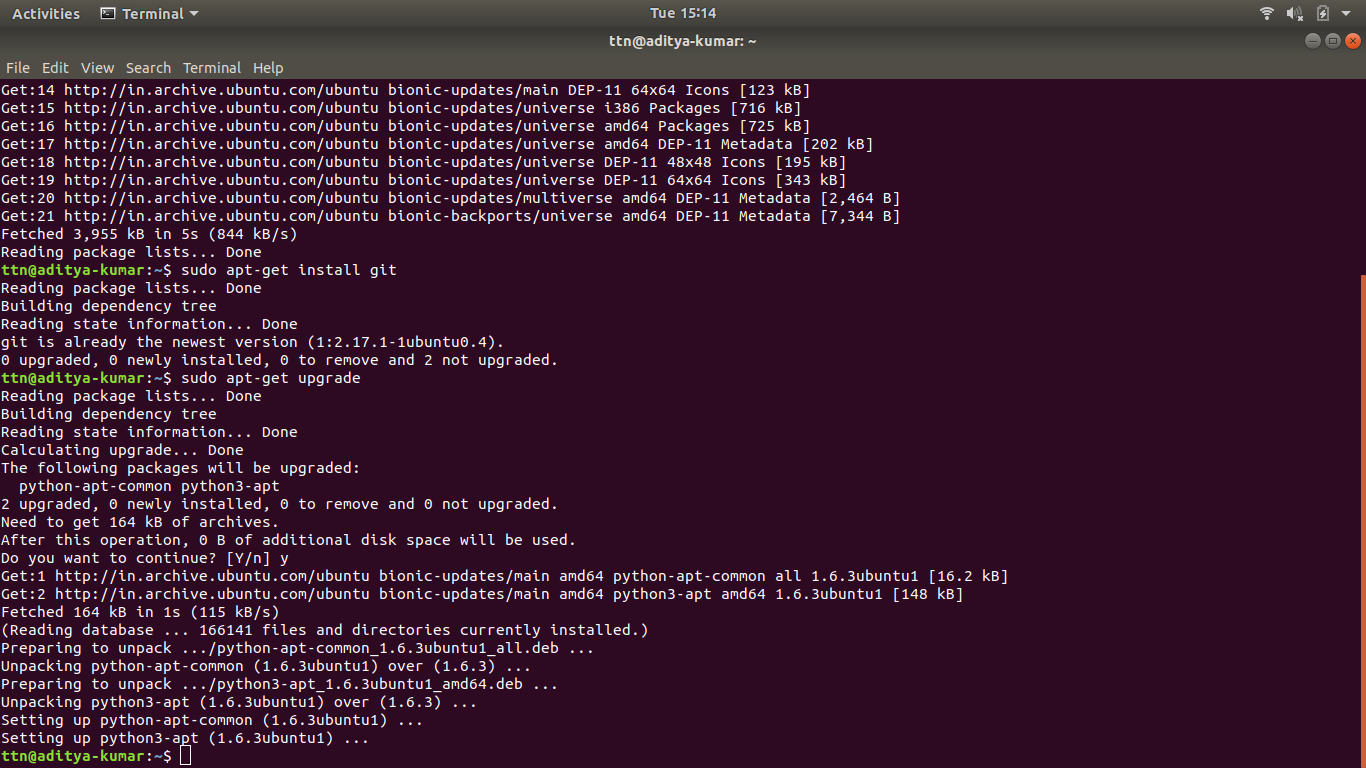
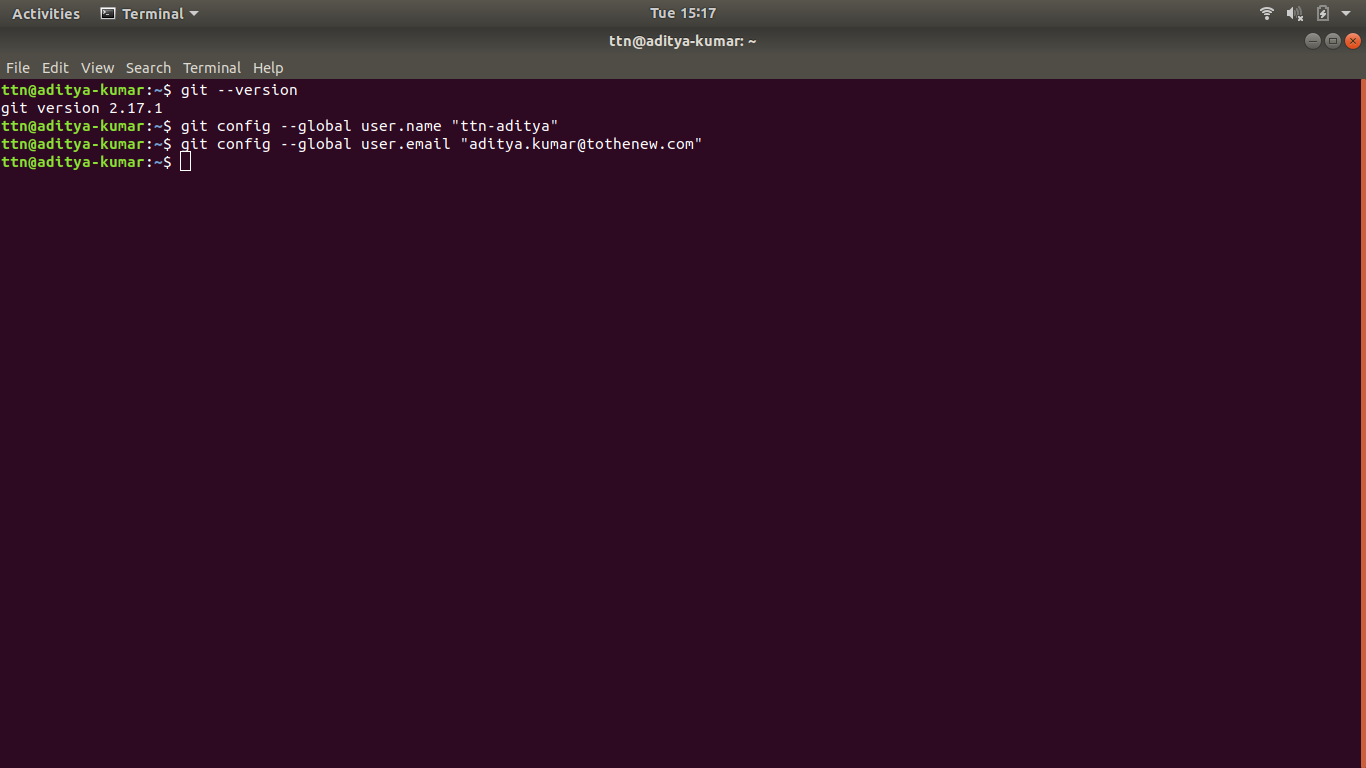
1.Git Setup<https://confluence.atlassian.com/bitbucket/set-up-git-744723531.html>



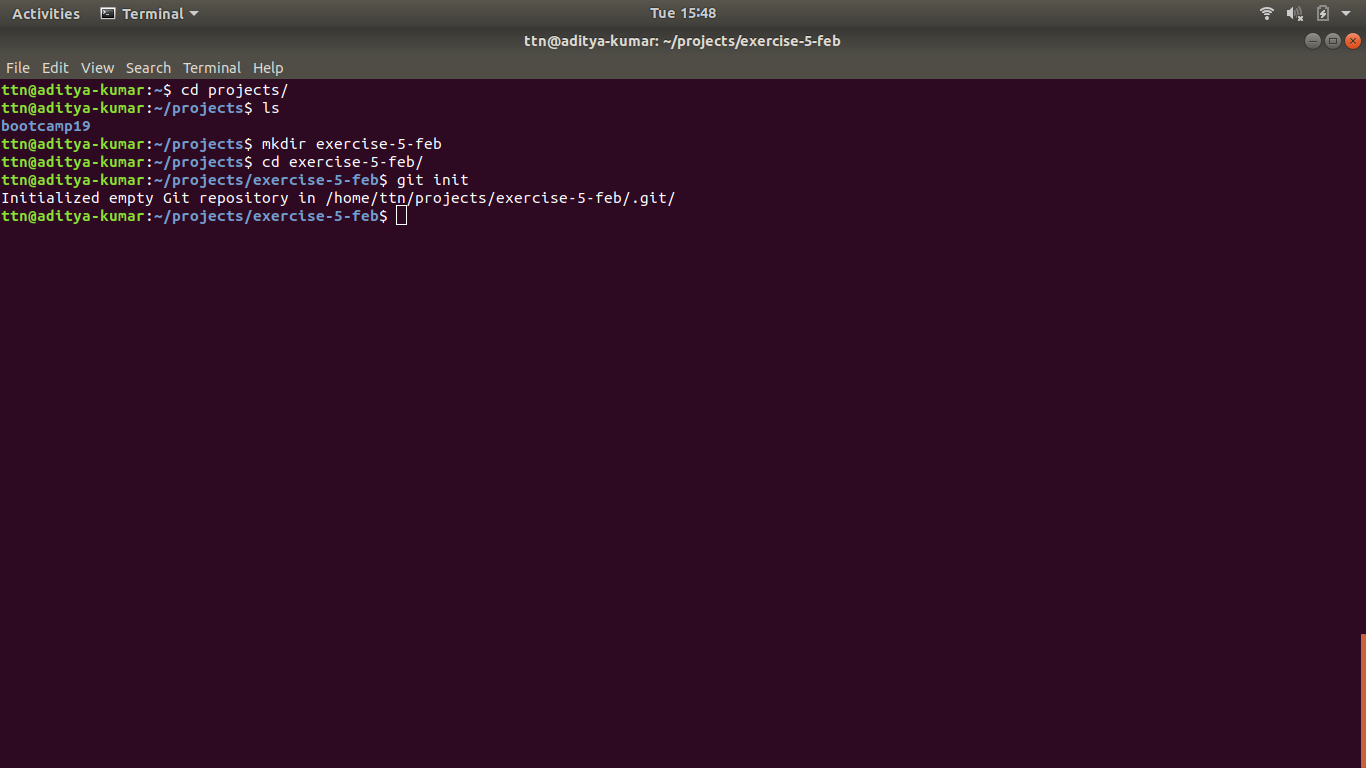
For update and install git package



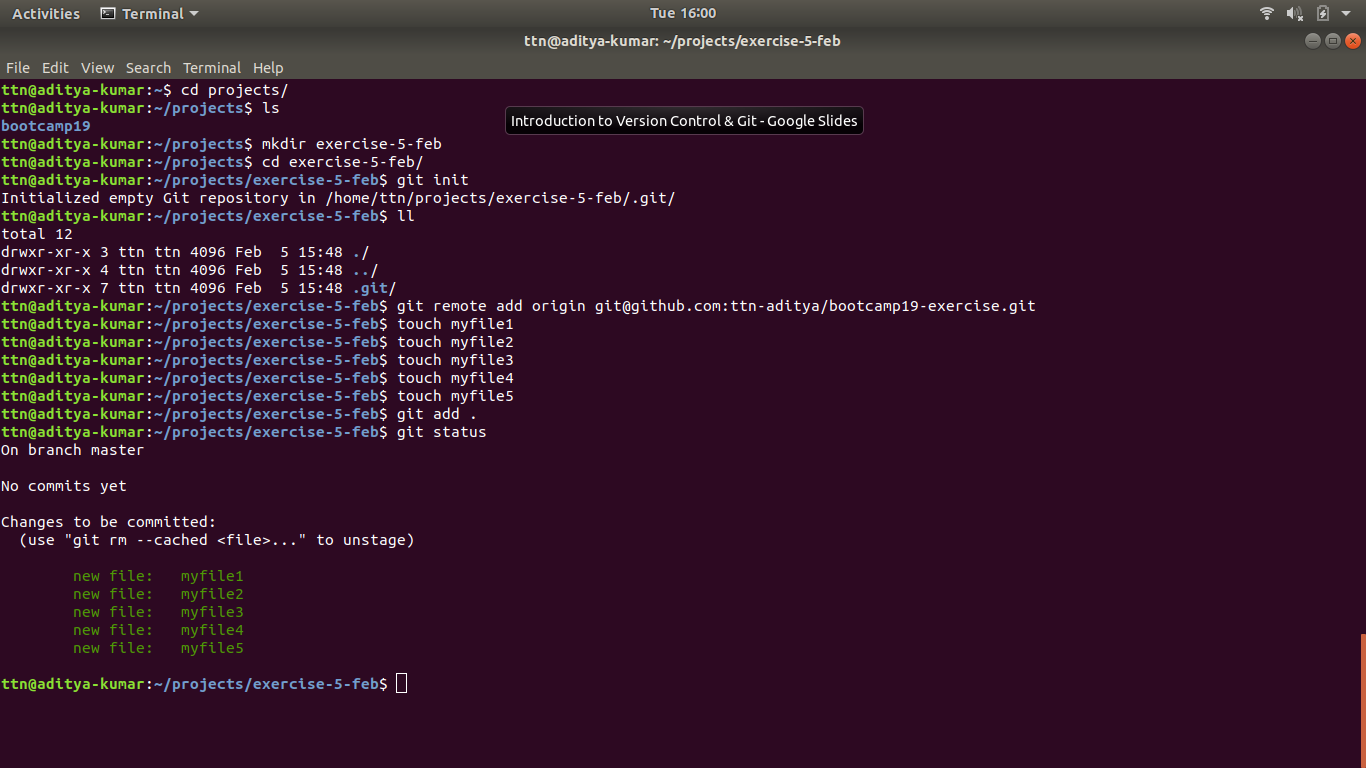
For upgrade

Setting up git username and user email

2.Initialize a Git Repository

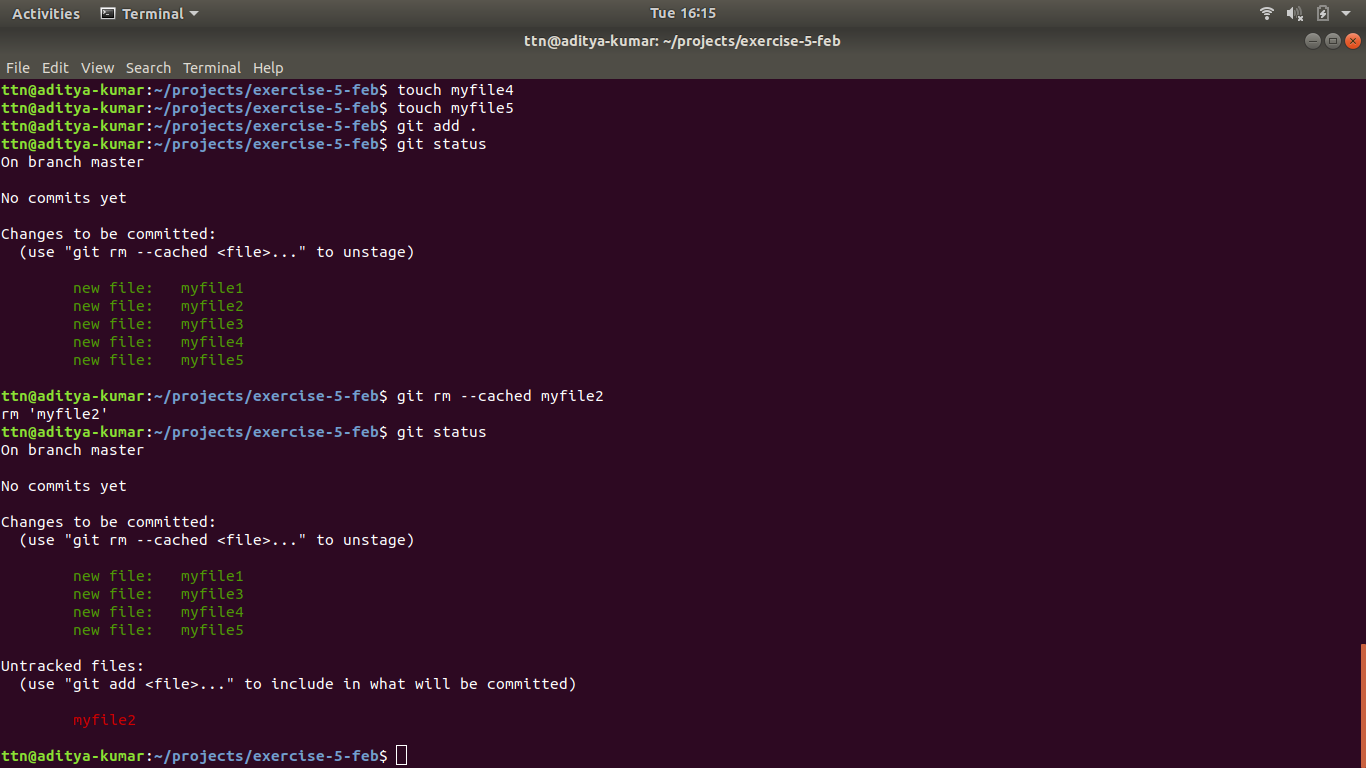


3.Add files to the repository



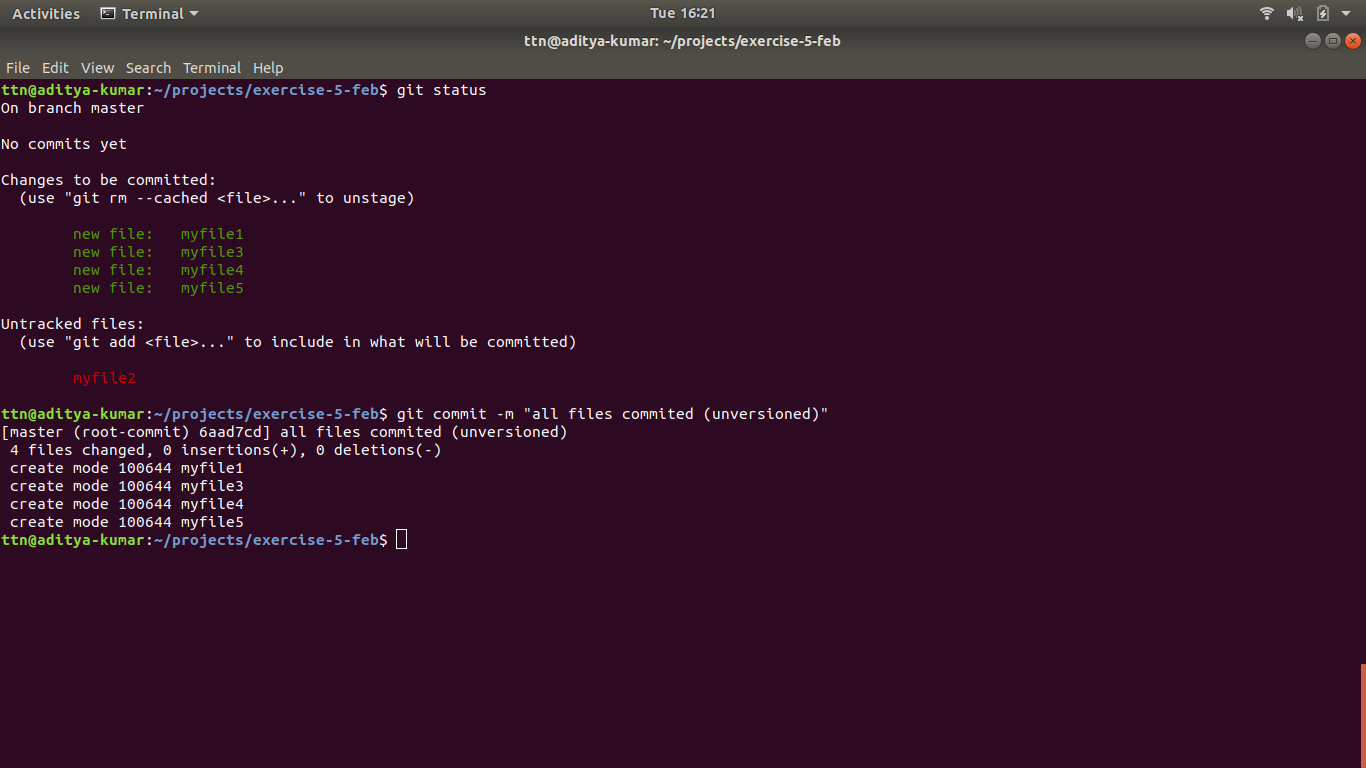
All the files were added to git

4.Unstage 1 file



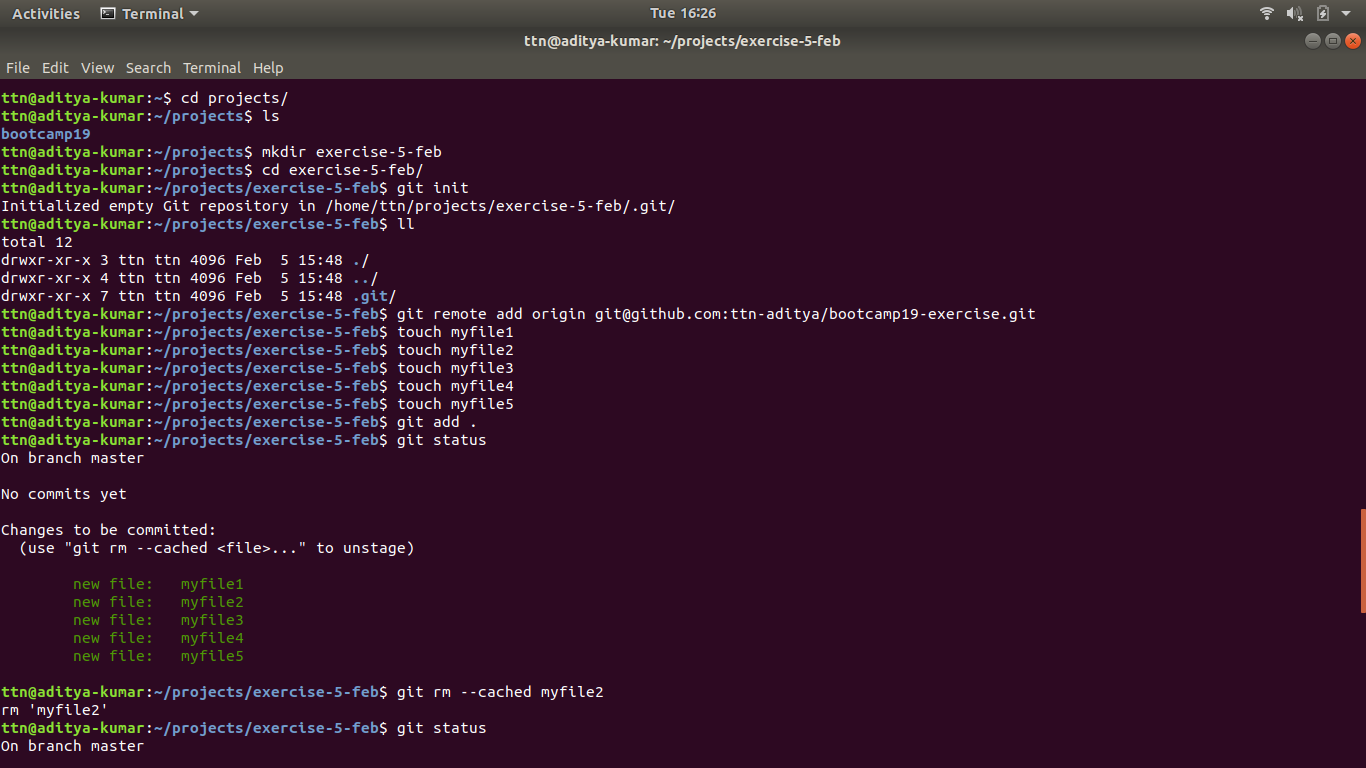
Since files were unversioned so “git rm --cached <file>...” worked successfully here

5.Commit the file

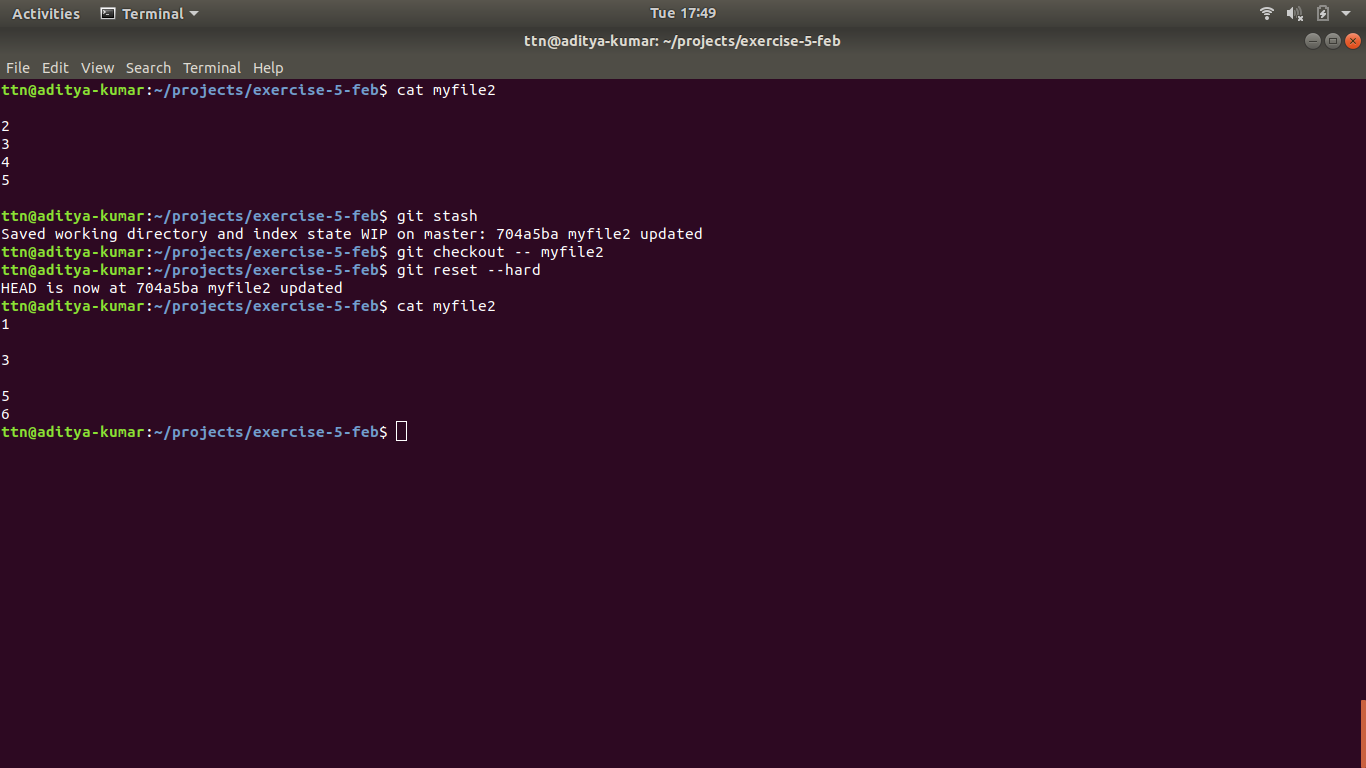


All unversioned files are now committed with message “all files commited (unversioned)”

6.Add a remote

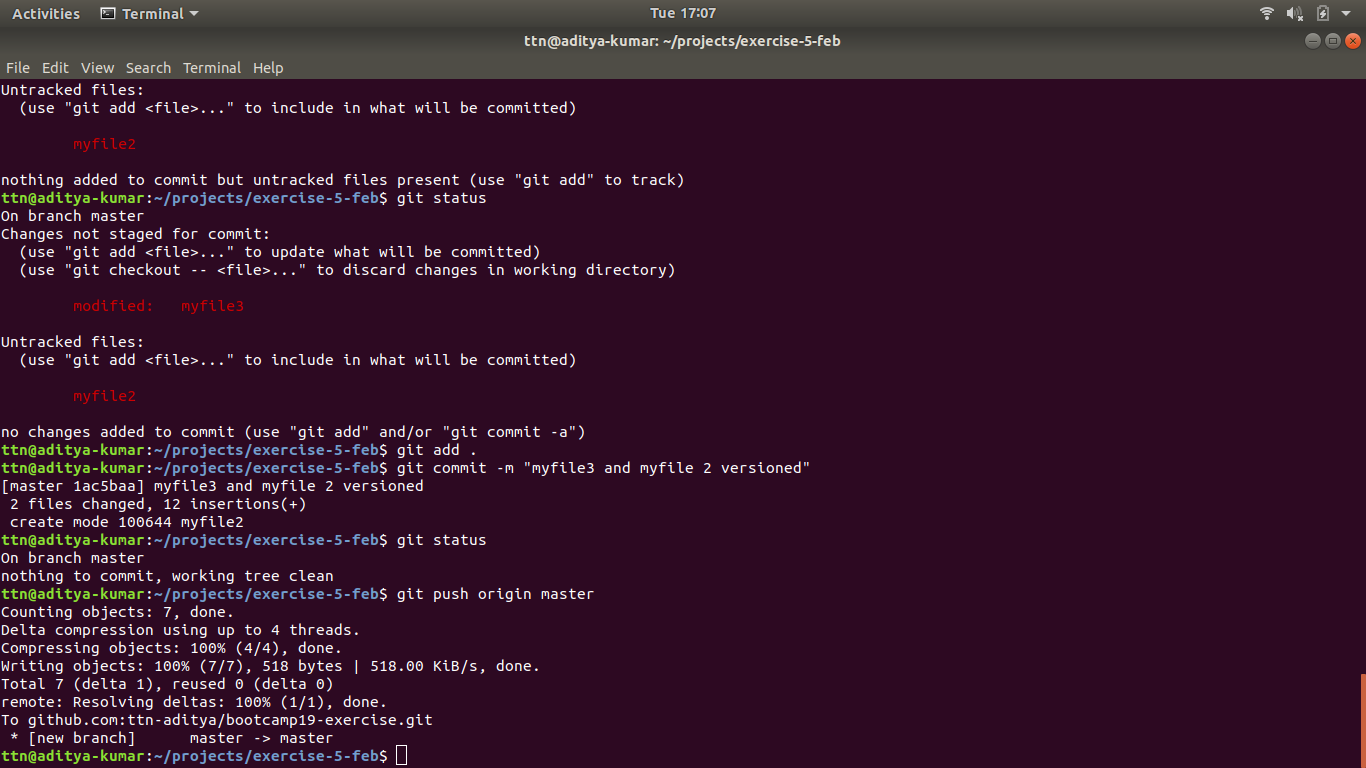


7.Undo changes to a particular file



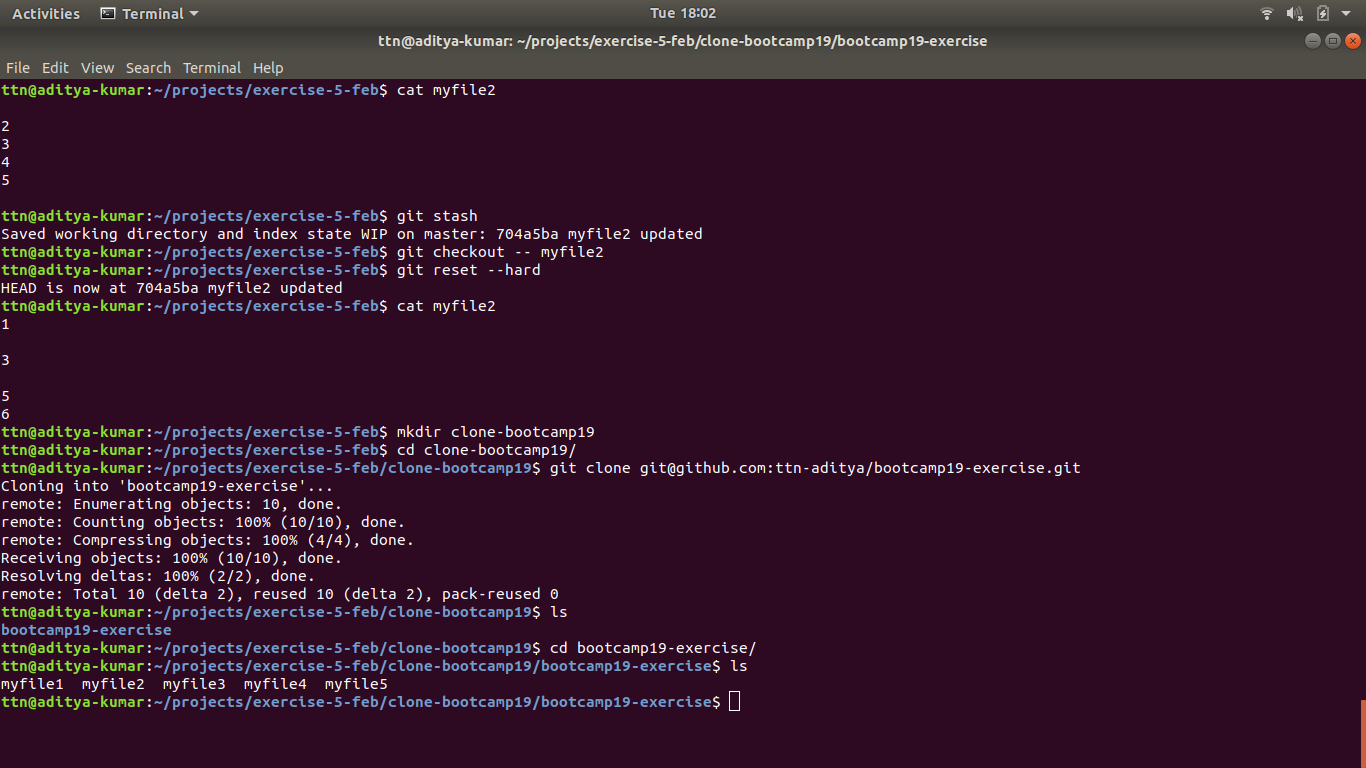
A file named myfile2 is pushed to remote server and then some modifications were made on myfile2 on local server.By using reset --hard the file on local server is being brought back to previous version.

8.Push changes to Github

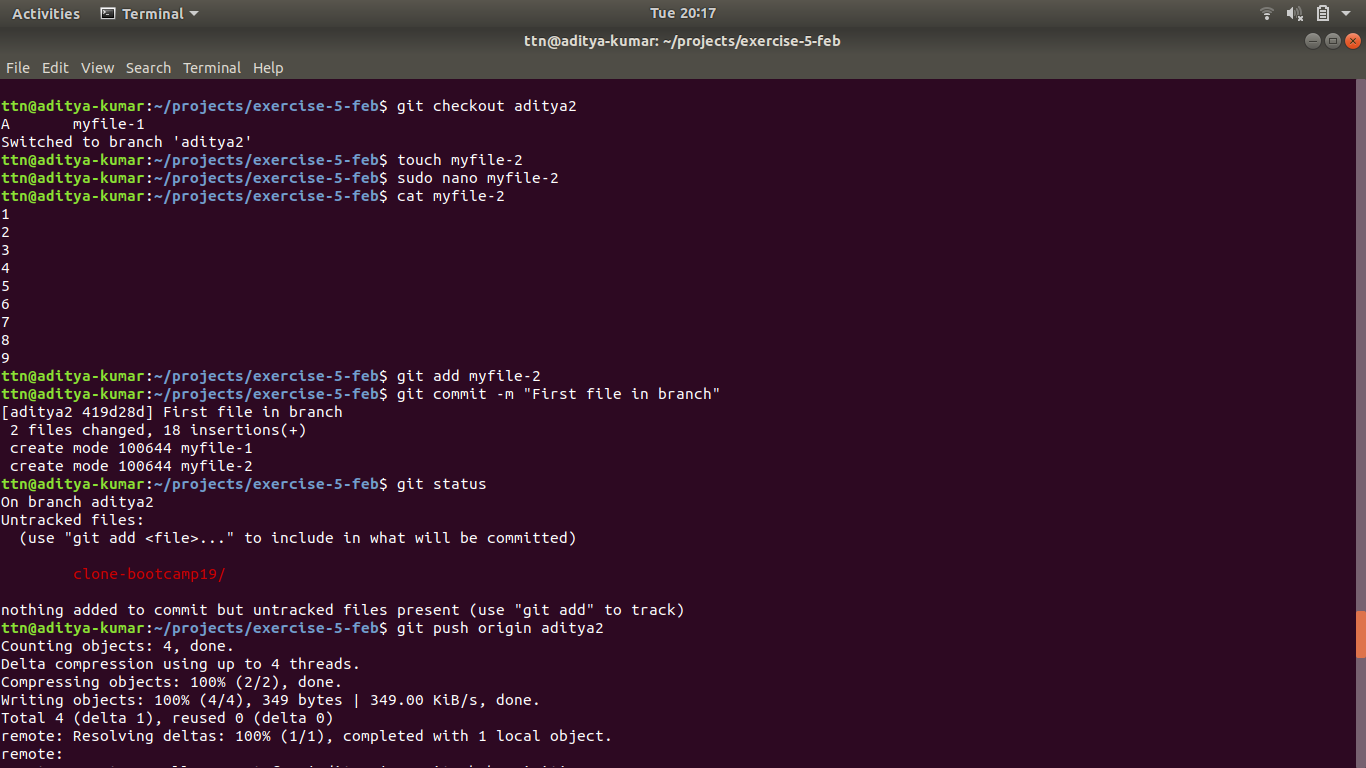


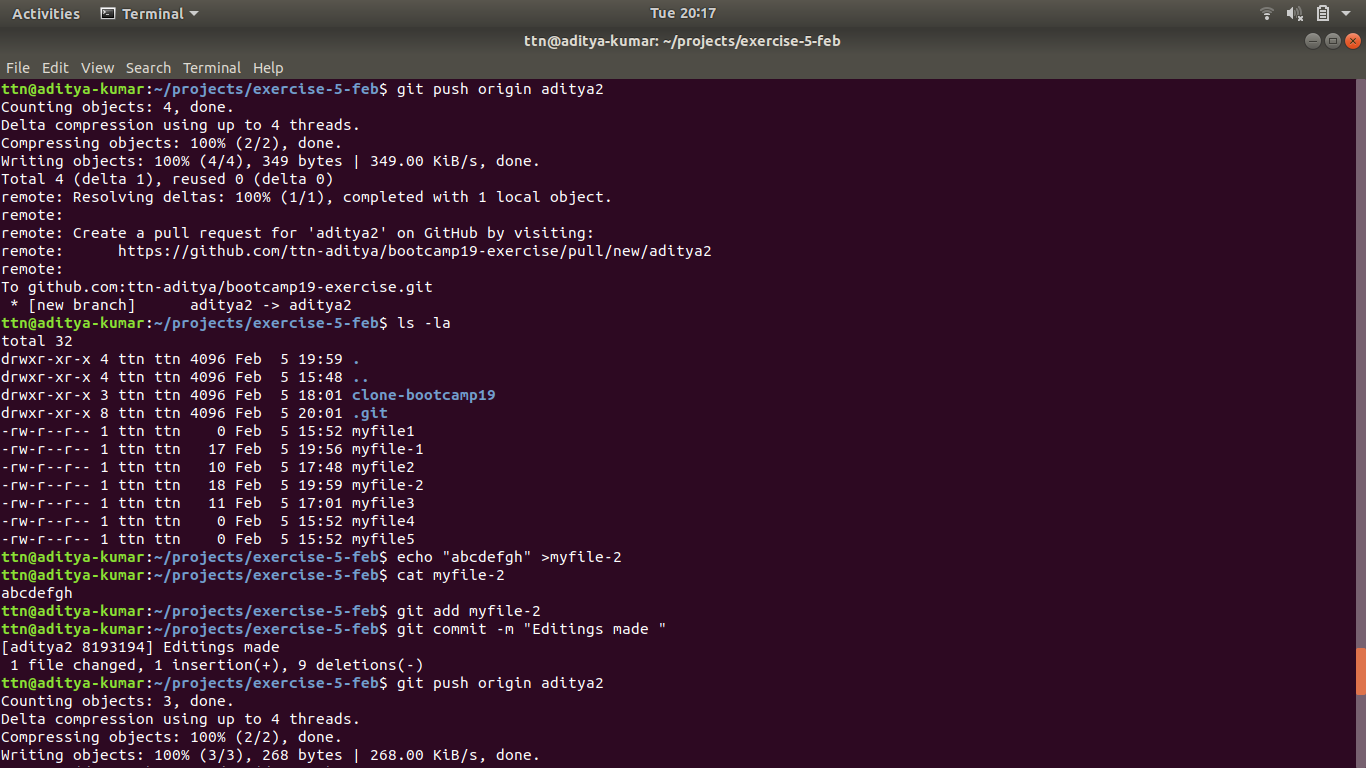
Unstaged file added to git repository and both myfile2 and myfile3 are committed with message “myfile1 and myfile2 (versioned)”

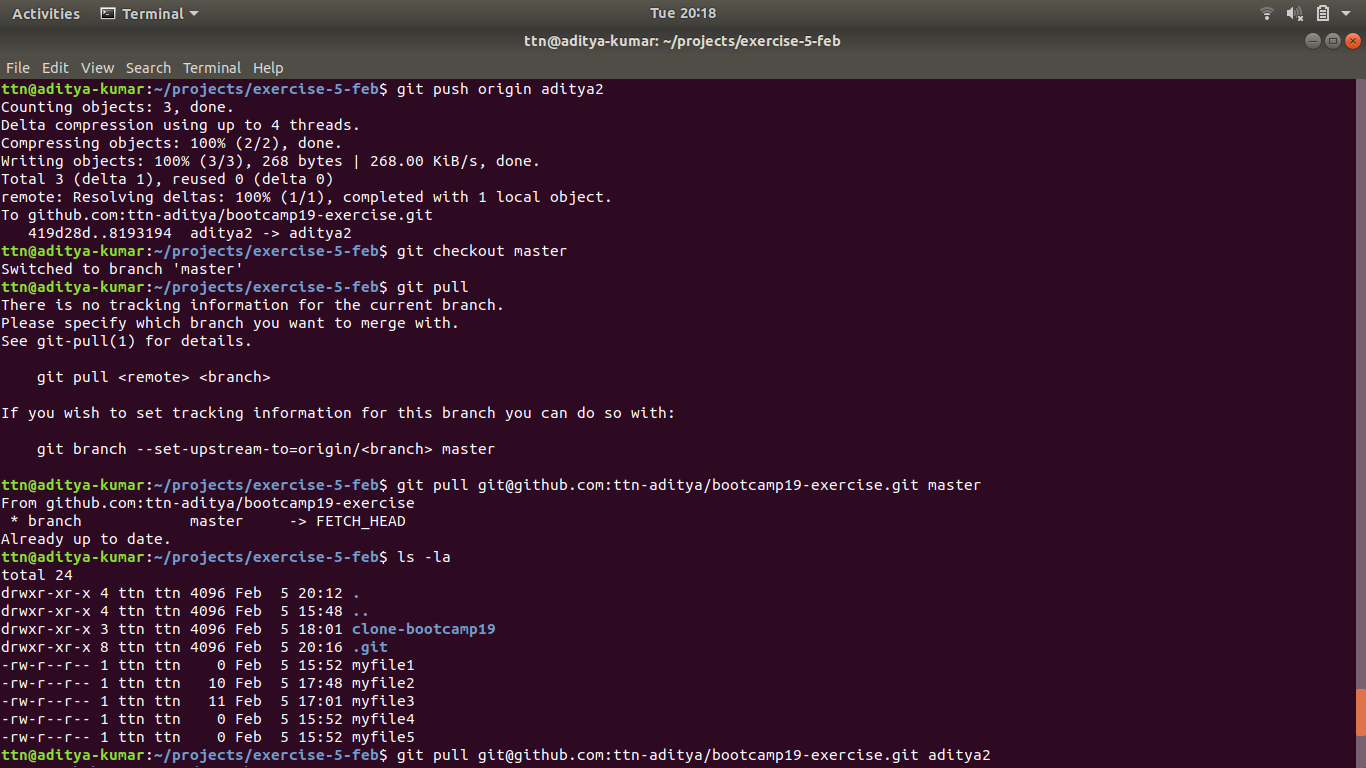
9.Clone the repository

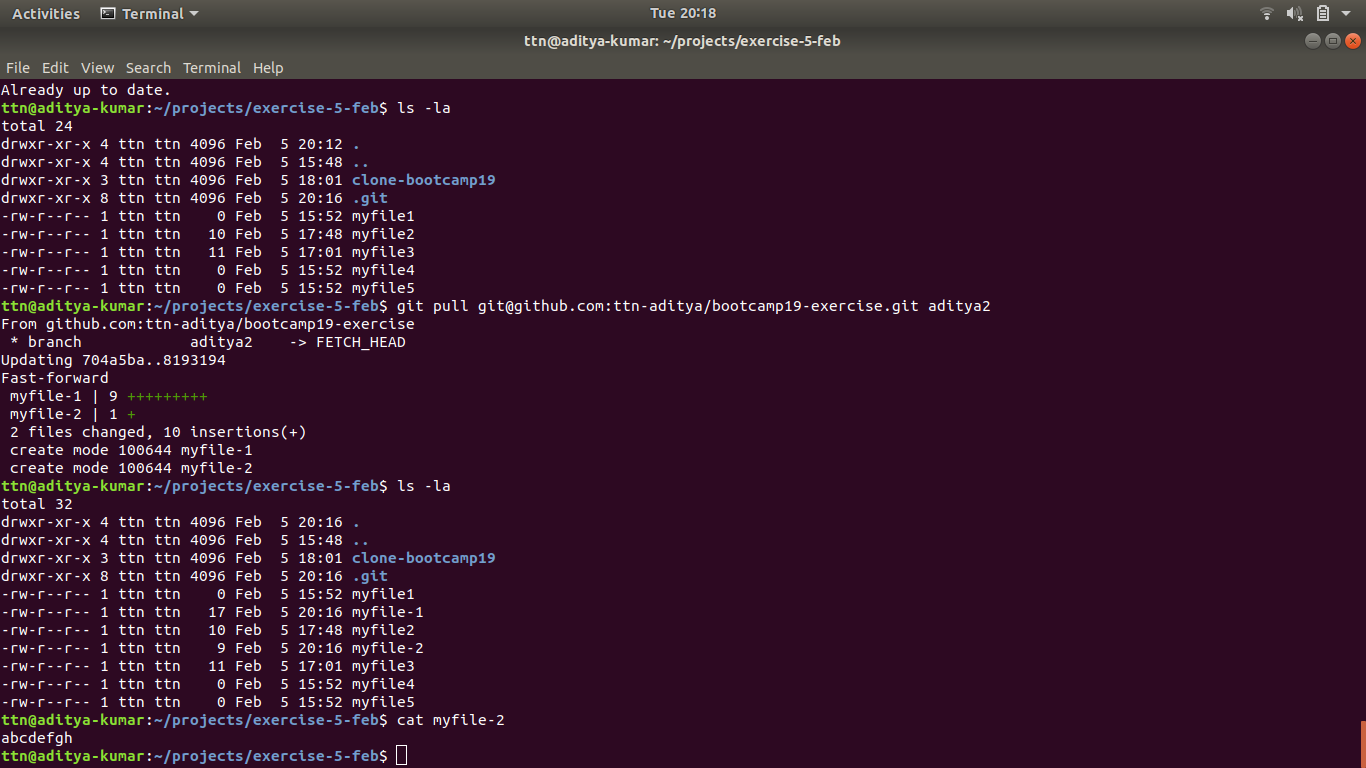


The complete repository is being cloned to a new folder named “clone-bootcamp”

10. Add changes to one of the copies and pull the changes in the other.





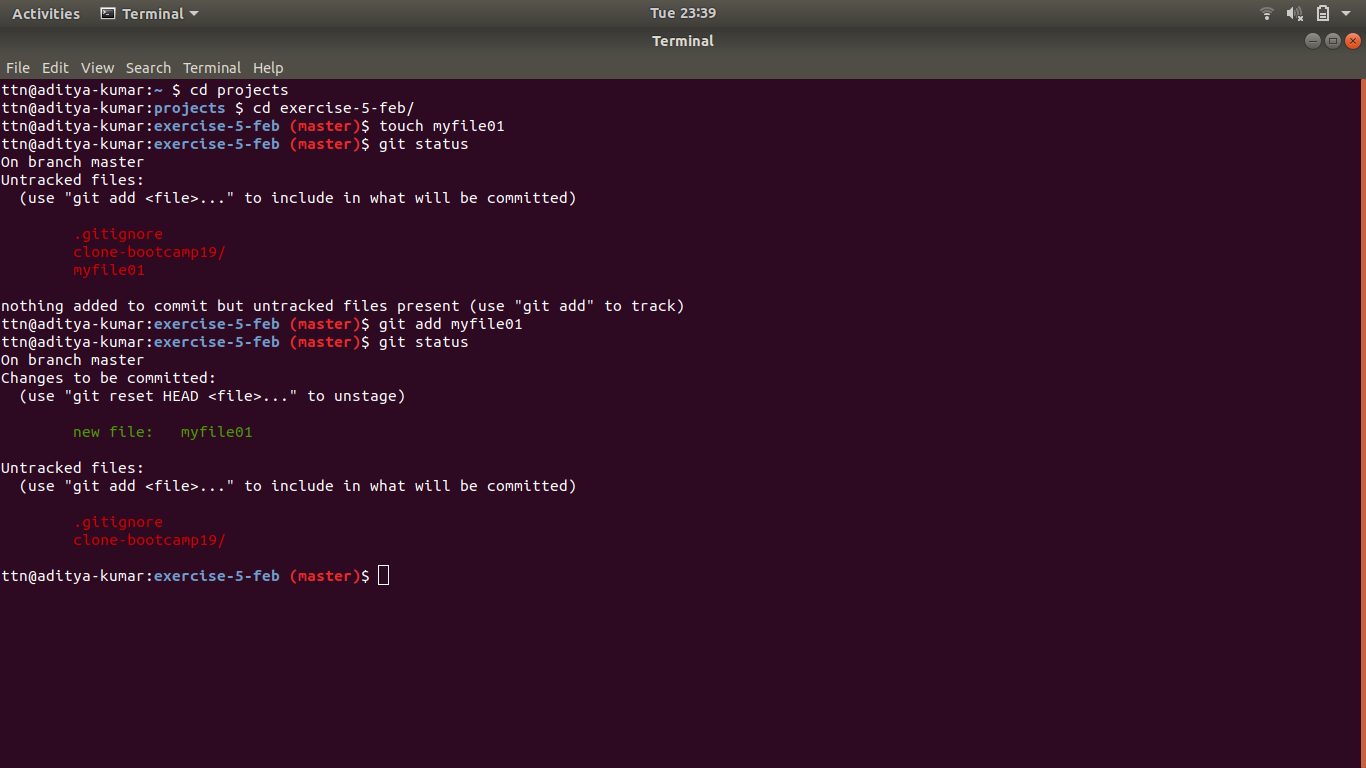


A file is created in branch aditya2 named myfile-2 with some data as shown in first fig.

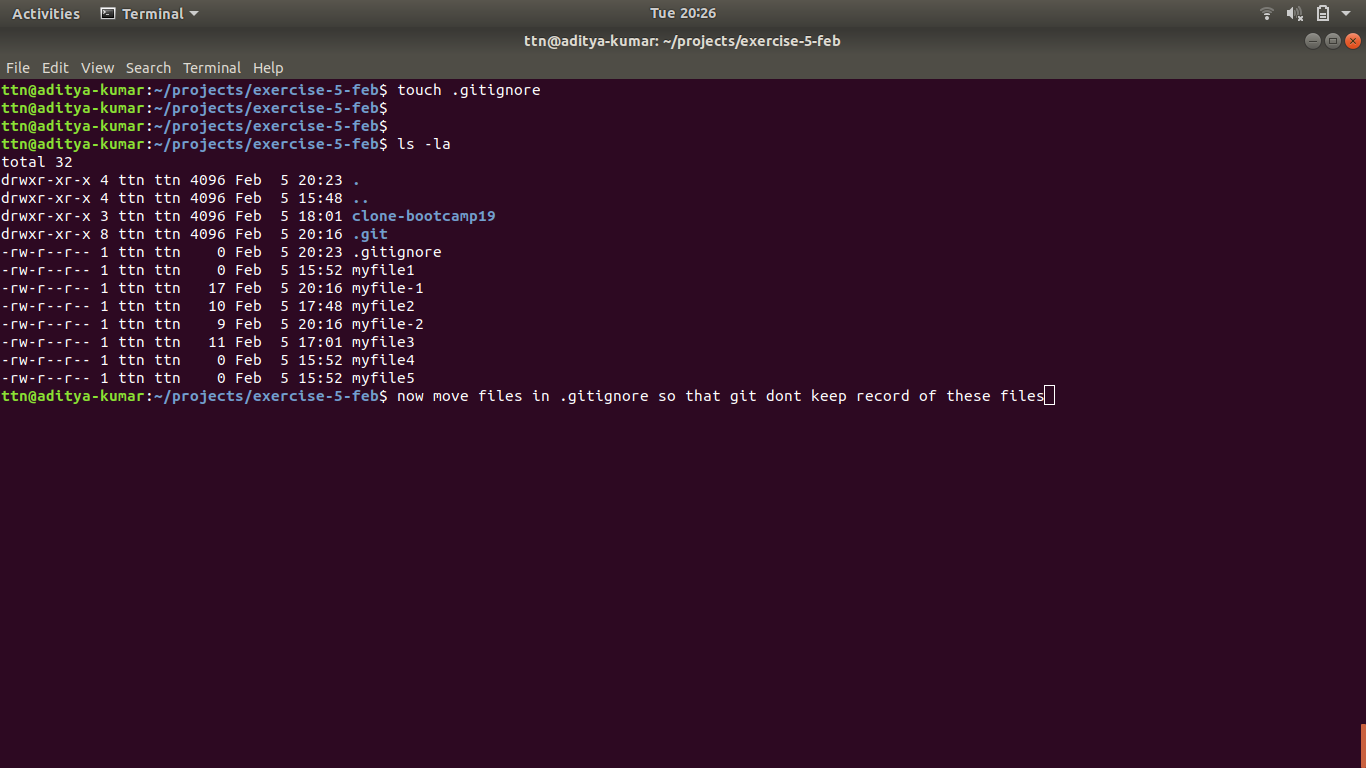
The data is then manipulated as shown in second fig.

The branch is then changed to master and then the whole data is copied to branch master from aditya2

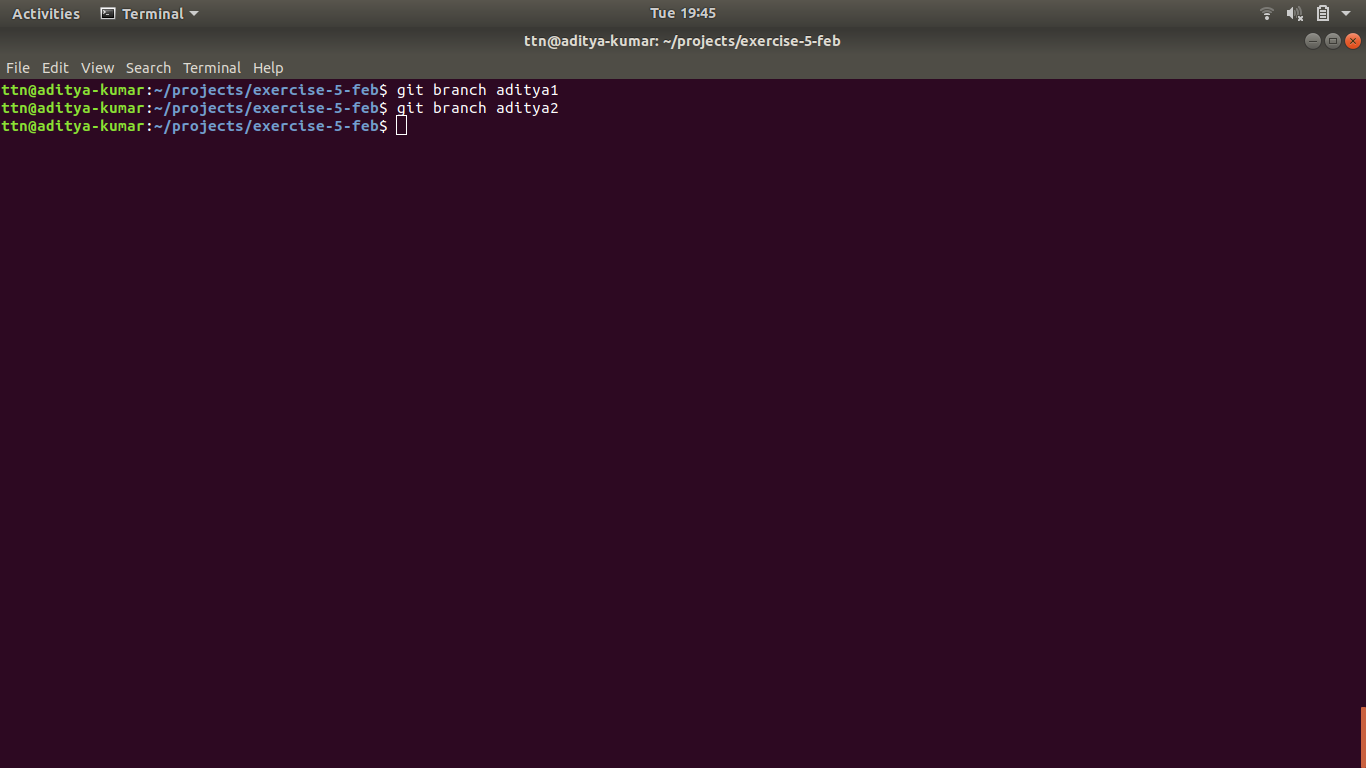
11.Check Differences between a file and its staged version



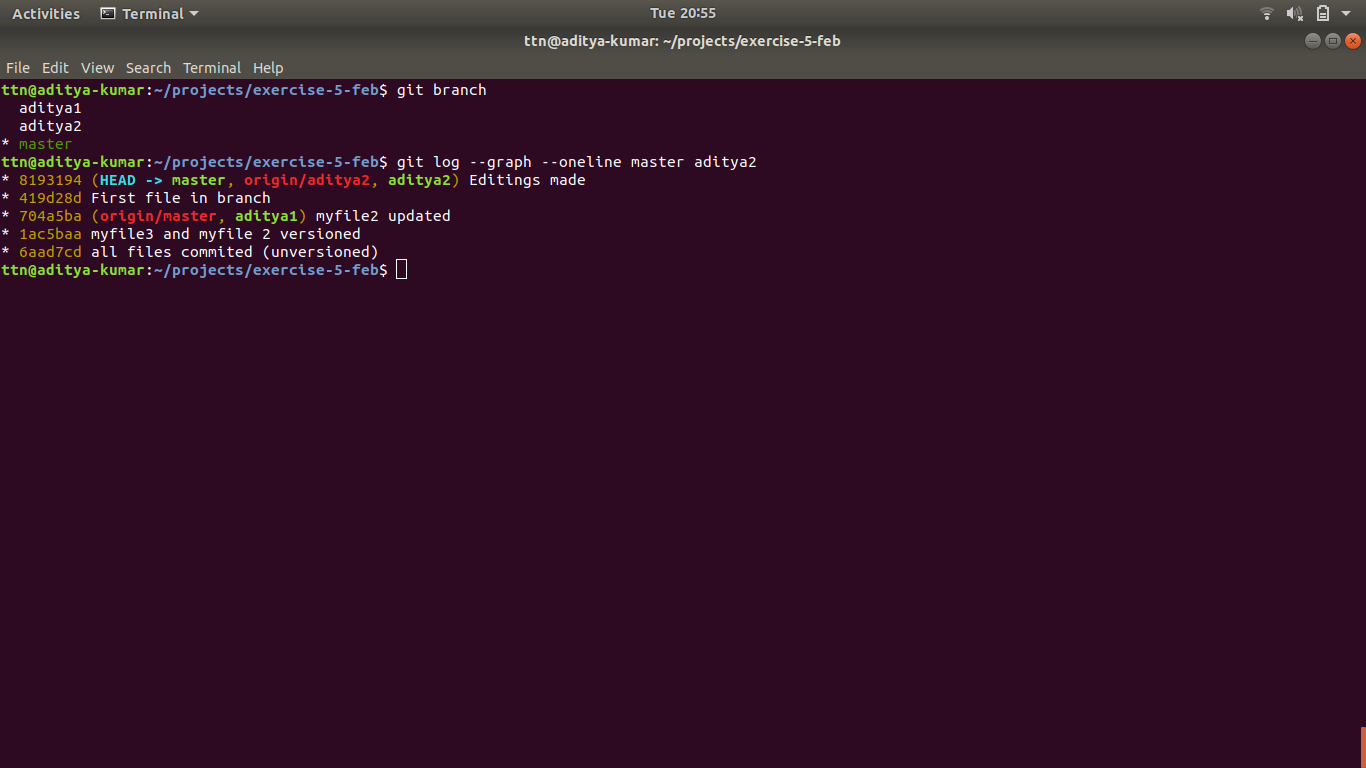
12.Ignore a few files to be checked in



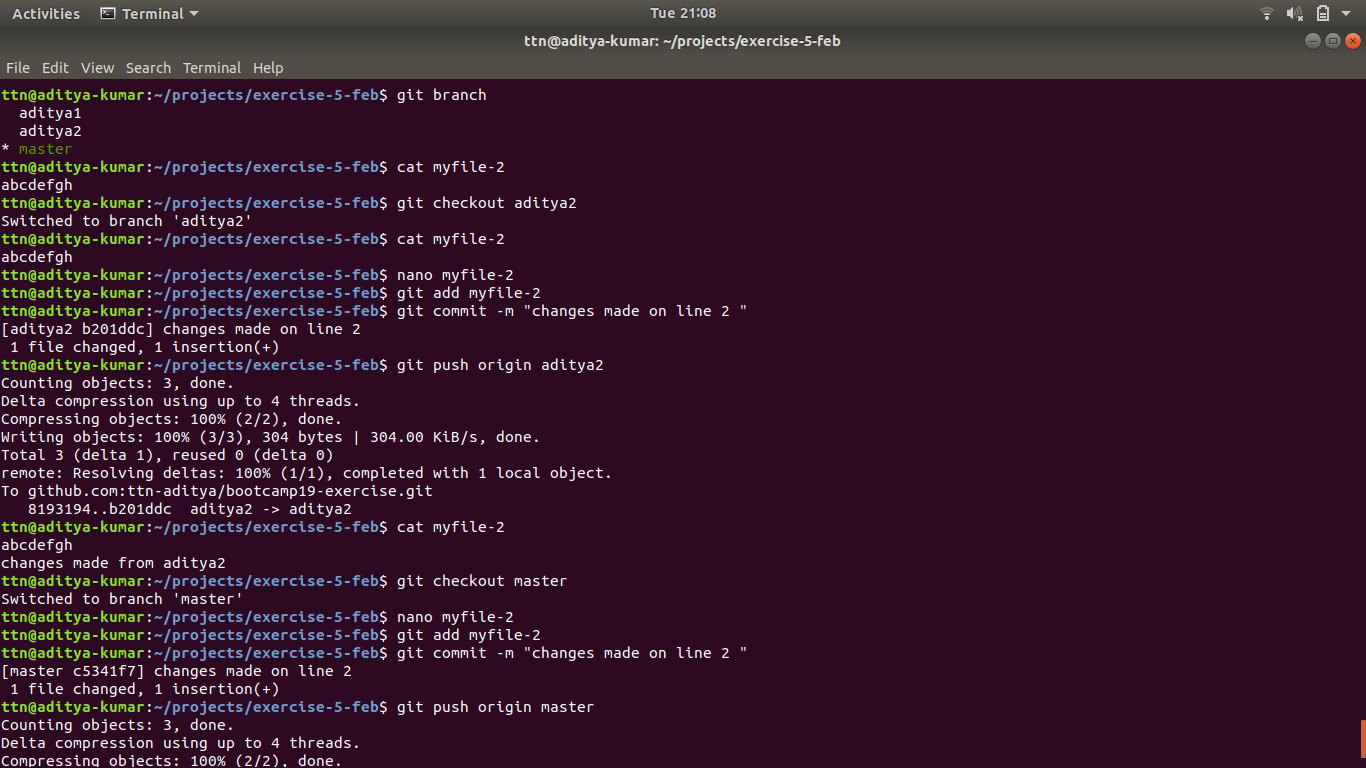
13.Create a new branch.

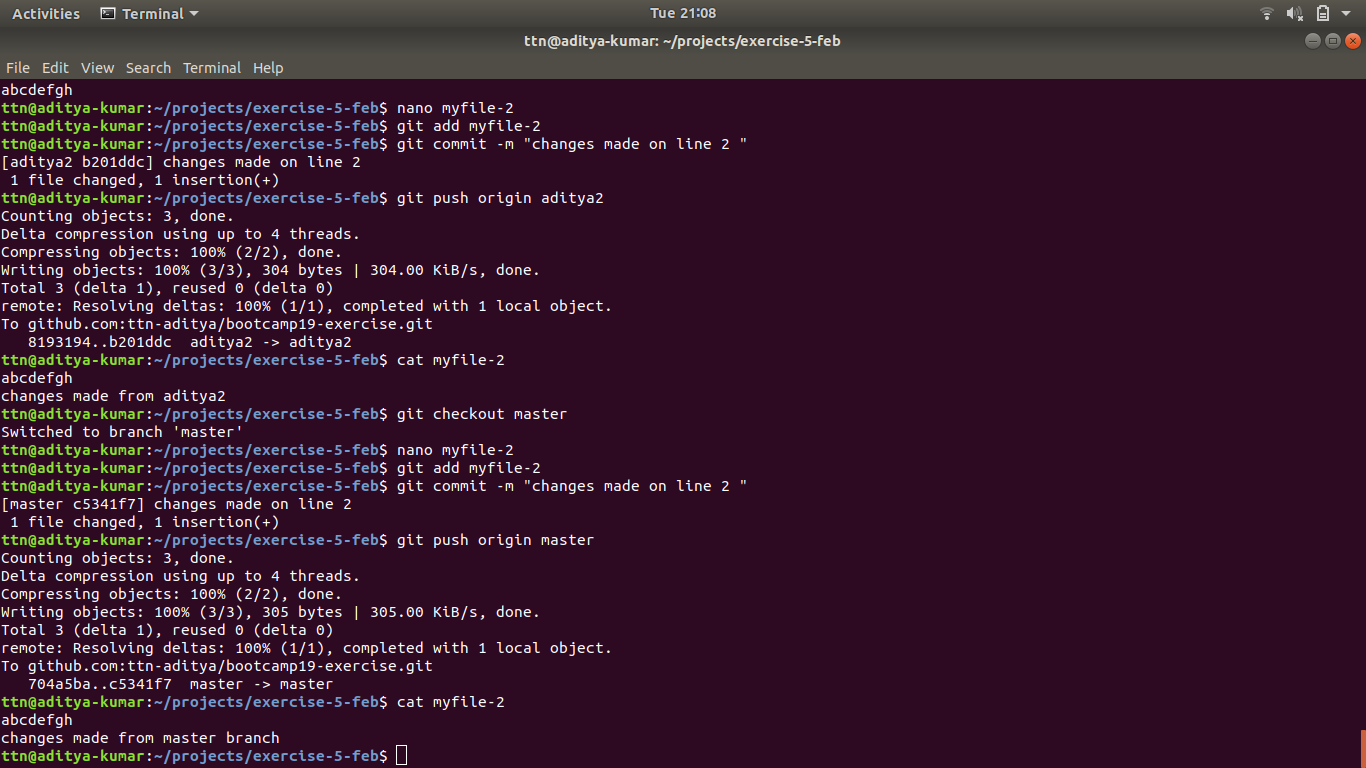


14.Diverge them with commits

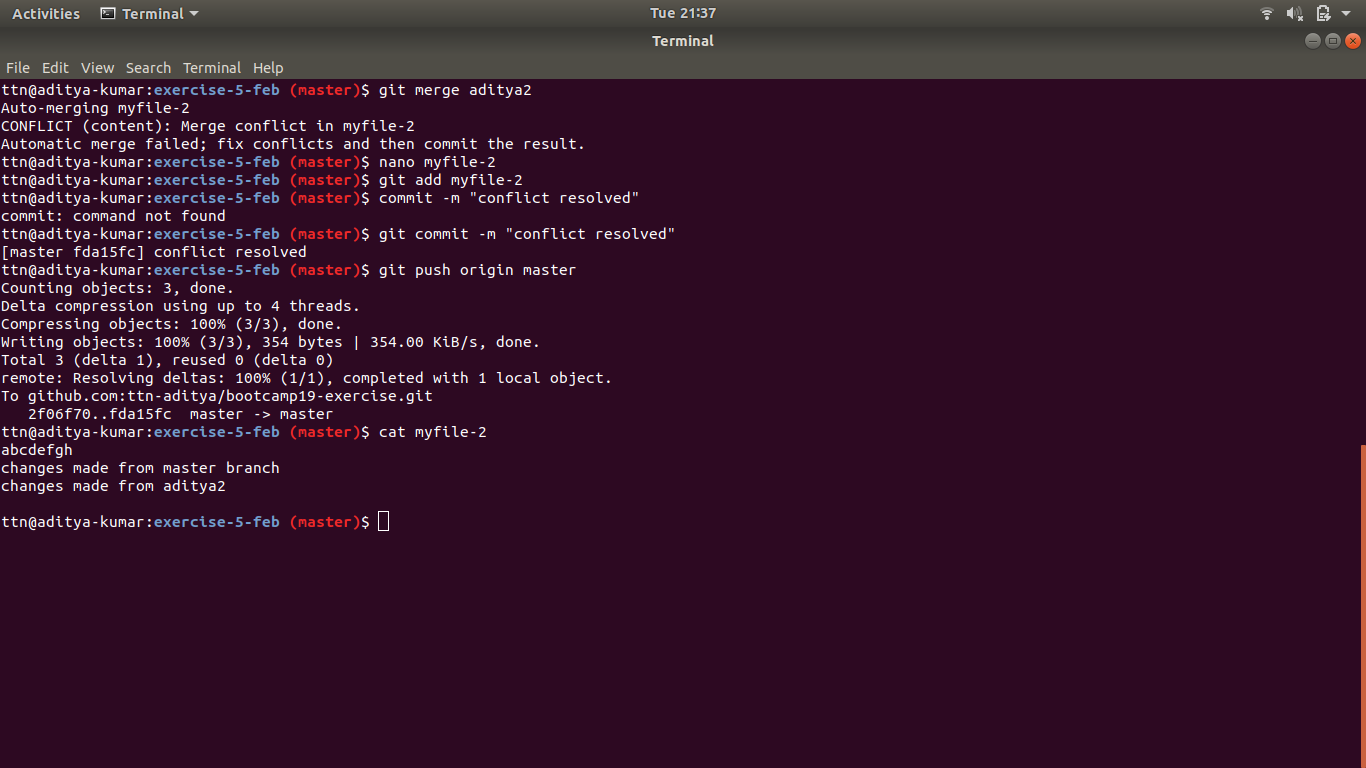


15.Edit the same file at the same line on both branches and commit.

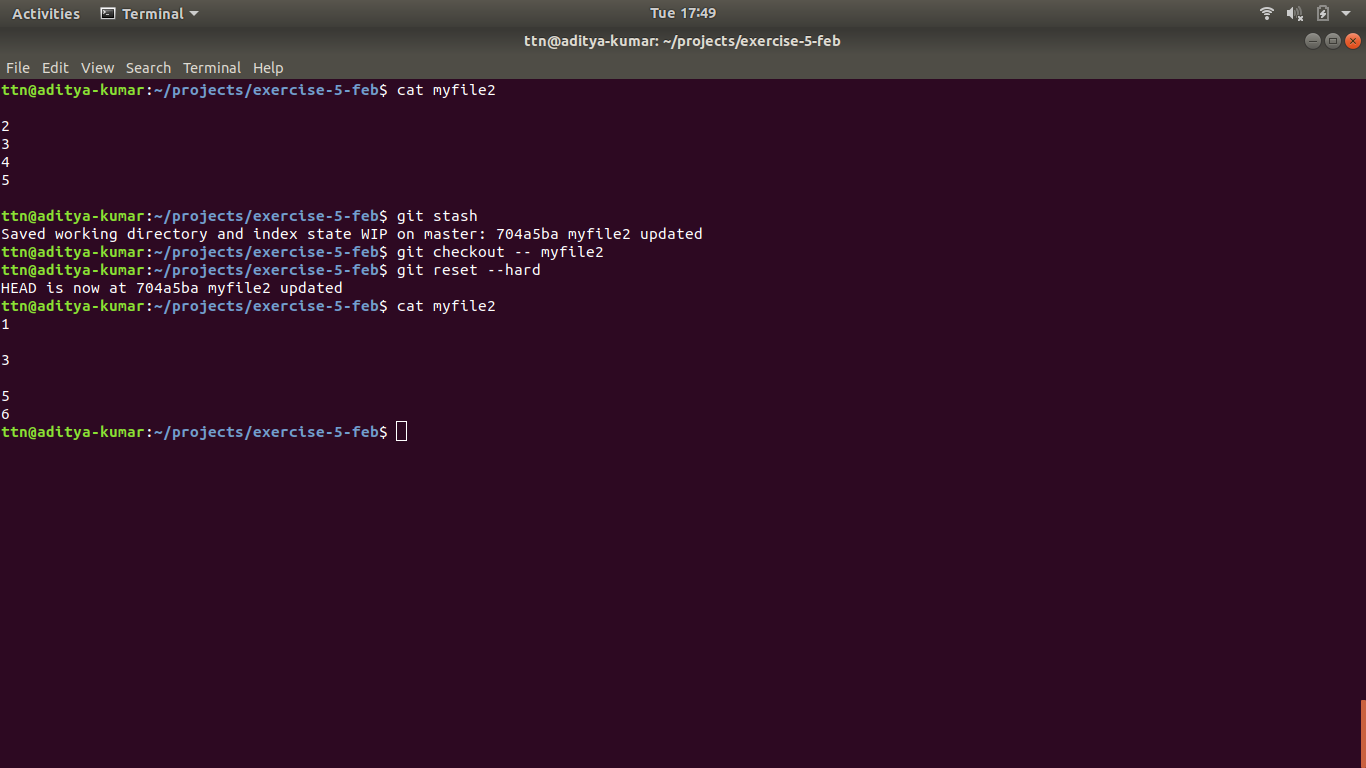




16.Try merging and resolve merge conflicts



17.Stash the changes and pop them



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

