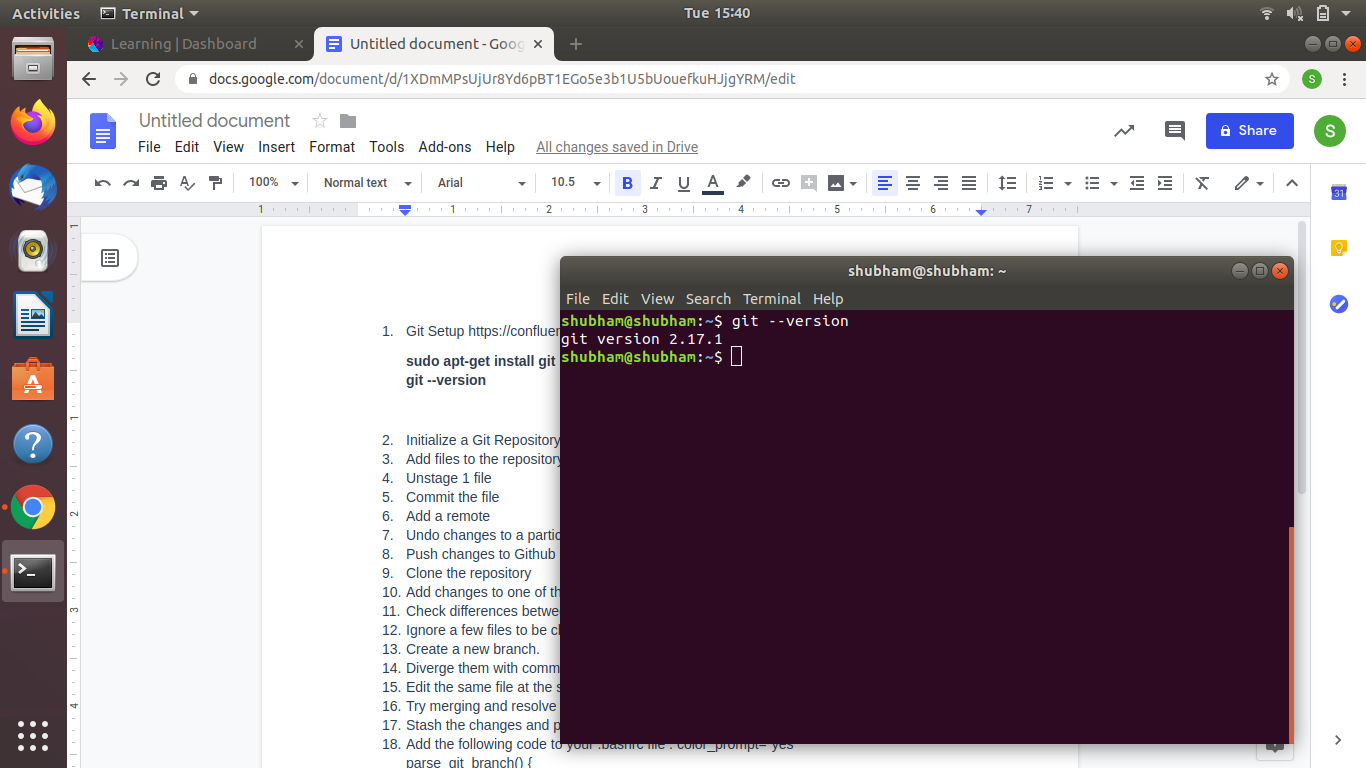
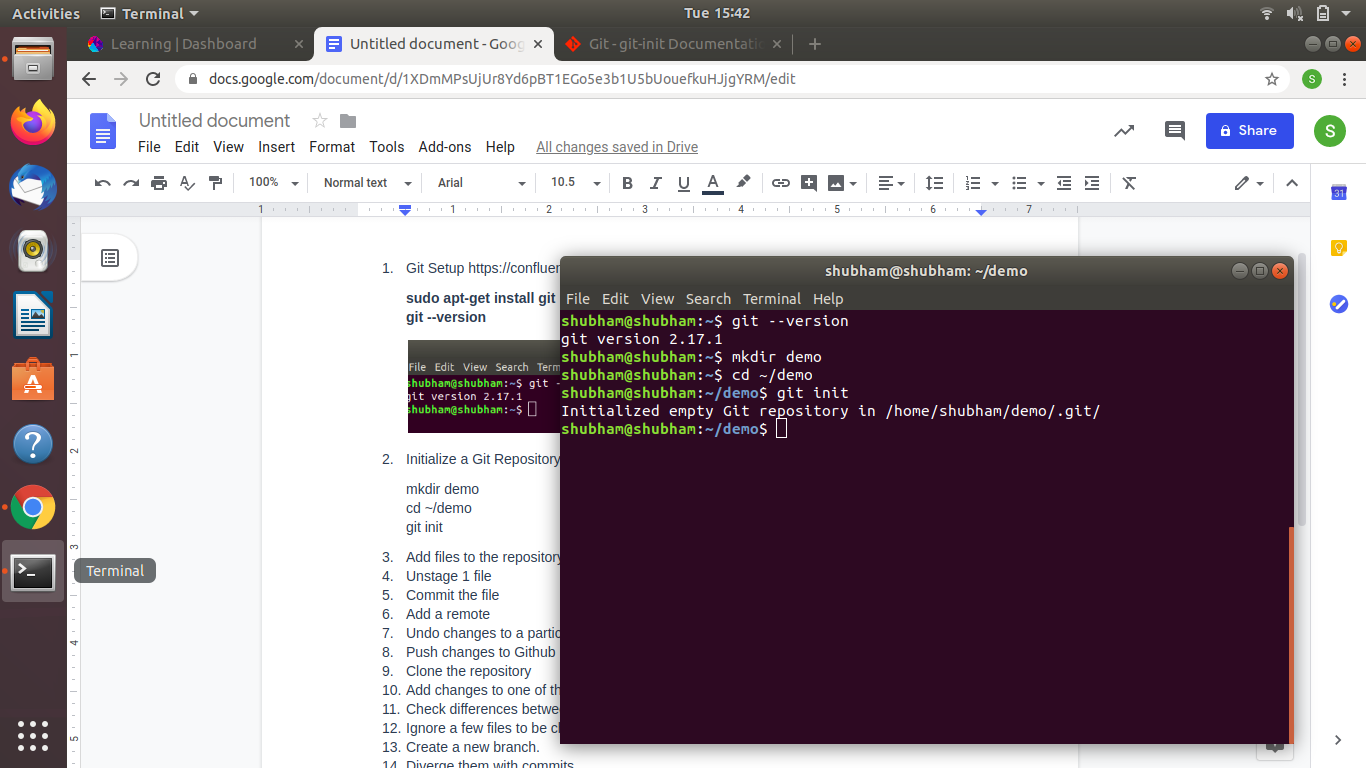
1. Git Setup https://confluence.atlassian.com/bitbucket/set-up-git-744723531.html

**sudo apt-get install git  
git --version**

****

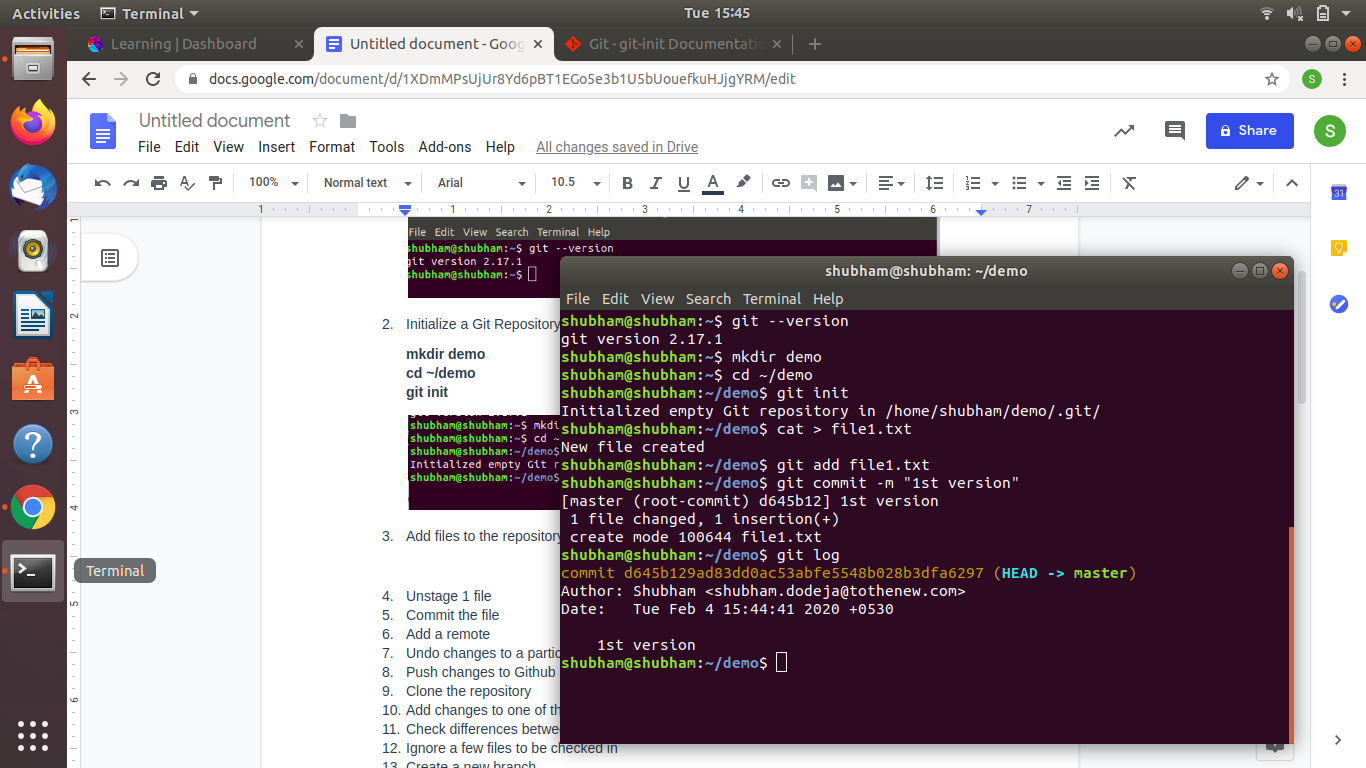
1. Initialize a Git Repository

**mkdir demo  
cd ~/demo  
git init**

****

1. Add files to the repository

**cat > file1.txt  
git add file1.txt (stage)**

****

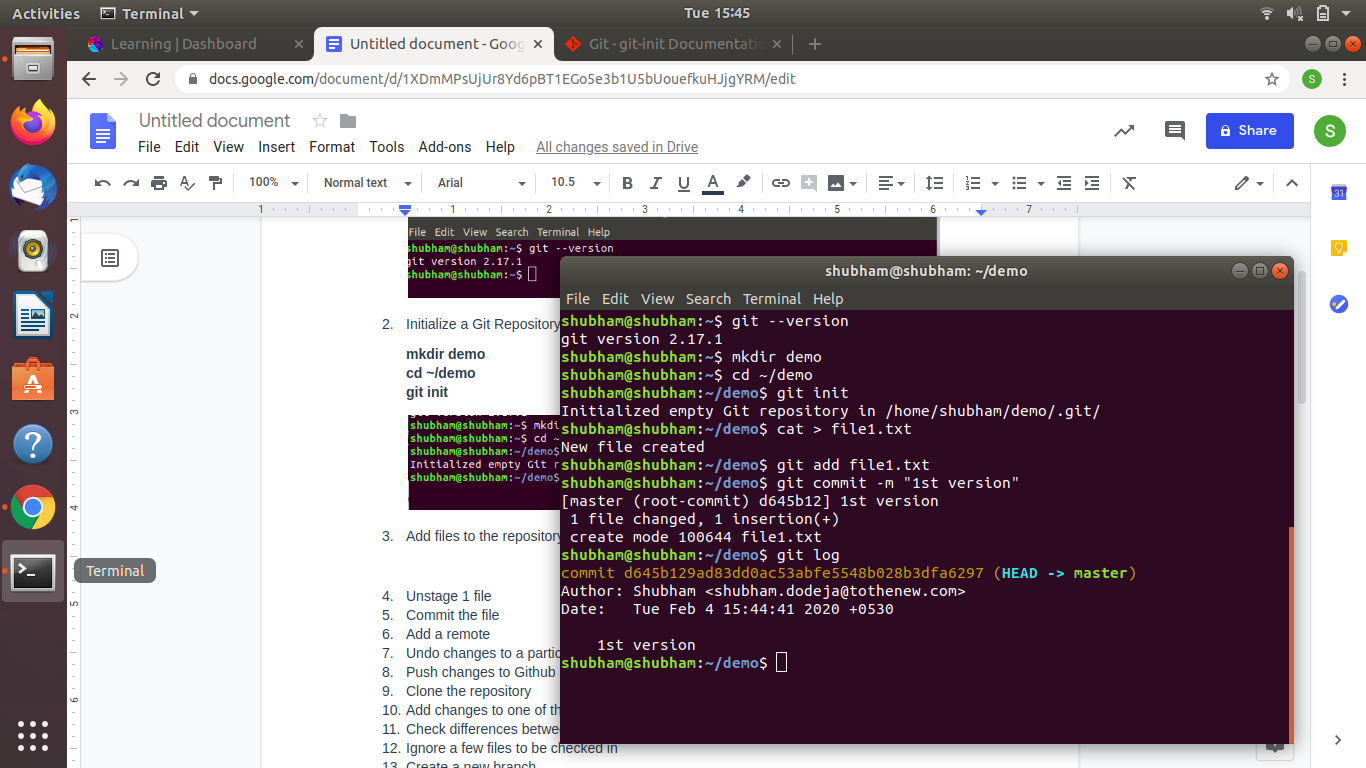
1. Unstage 1 file

**git add file1.txt (stage)  
git reset -- file1.txt (unstage)**



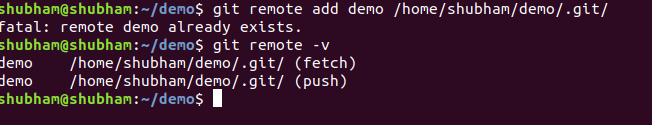
1. Commit the file

**cat > file1.txt (creating file)  
git add file1.txt (staging)**  
**git commit -m "1st version" (commit)**

****

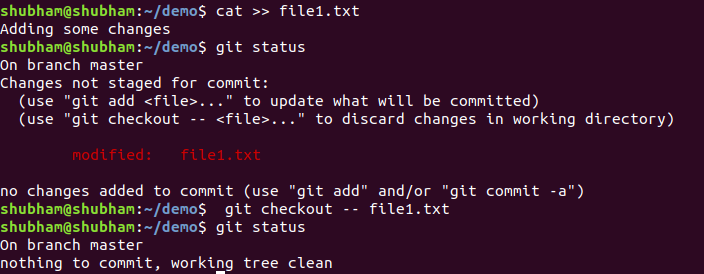
1. Add a remote

**git remote add demo /home/shubham/demo/.git/  
git remote -v**



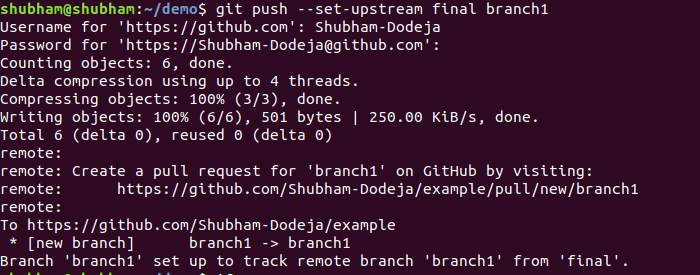
1. Undo changes to a particular file

**git checkout -- file1.txt**



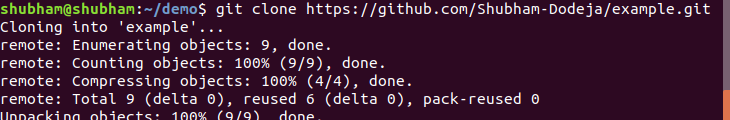
1. Push changes to Github

**Git push --set-upstream final branch1**



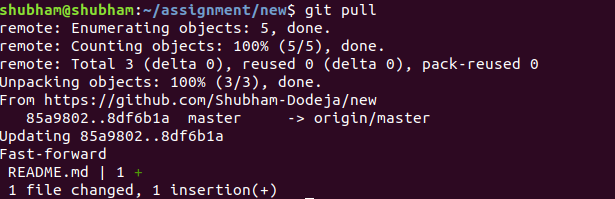
1. Clone the repository

**git clone https://github.com/Shubham-Dodeja/example.git**



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

**git pull**

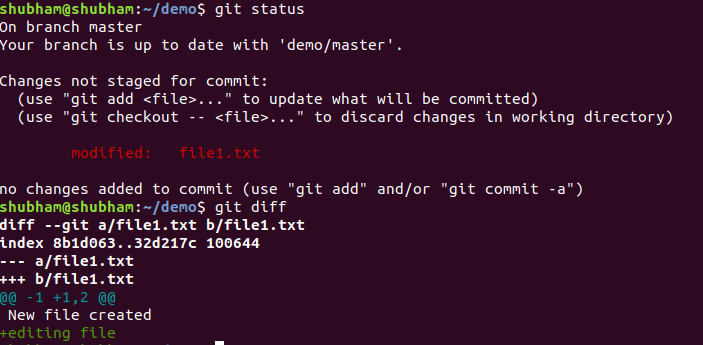


1. Check differences between a file and its staged version

**git diff**

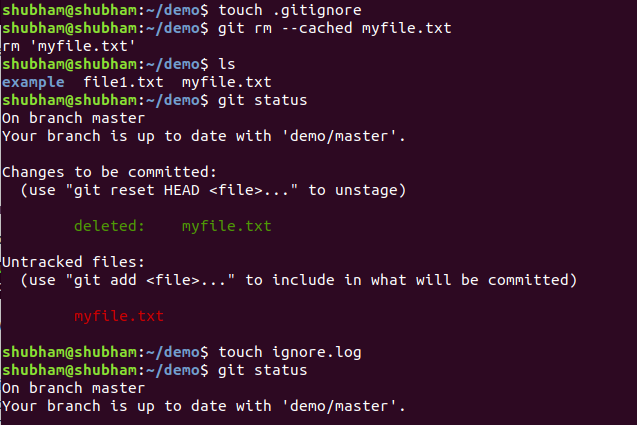
**A file which isn’t stage generally appears red in the terminal while one that is staged appears in green. Moreover a staged file is saved in the index while one which isn’t is processed in workspace.**

**A normal file is Untracked file while a staged file can be commited**



1. Ignore a few files to be checked in

**touch .gitignore  
git rm --cached myfile.txt  
Git status**



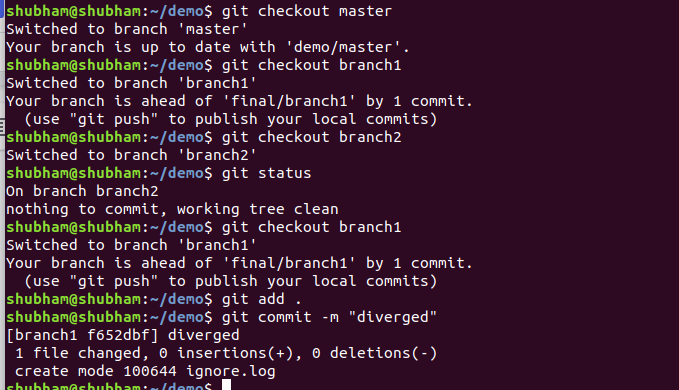
1. Create a new branch.

**git checkout -b "branch1"**



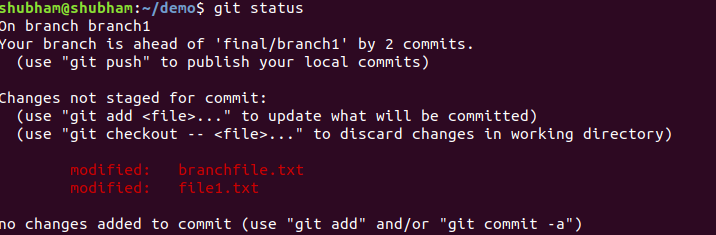
1. Diverge them with commits

**git checkout branch1  
edit a file  
Git commit -m “message”**



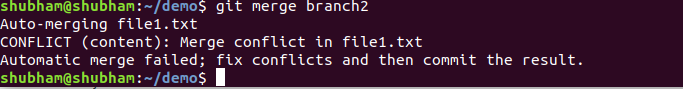
1. Edit the same file at the same line on both branches and commit

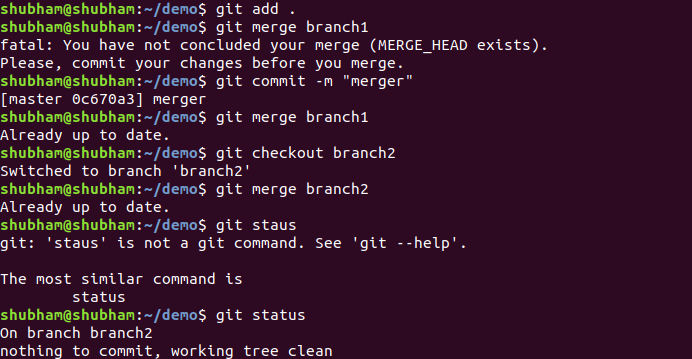
**gedit file1.txt  
git status**



1. Try merging and resolve merge conflicts

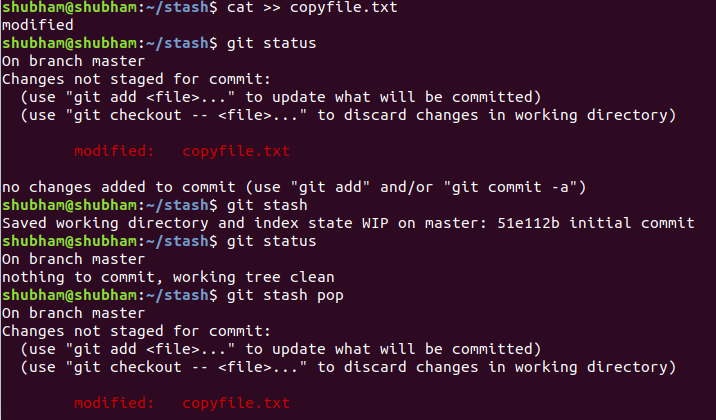
**git merge branch2**

**The most direct way to resolve a merge conflict is to edit the conflicted file. Open the file1.txt file in editor.**



1. Stash the changes and pop them

**cat >> copyfile.txt  
git status  
git stash  
git status  
git stash pop**



1. 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)\$ '

**gedit ~/.bashrc**

