**INTRODUCTION TO VERSION CONTROL EXERCISE**

**QUES1-** Git Setup.

**Solution-**



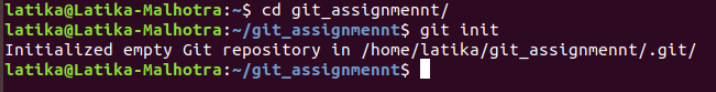






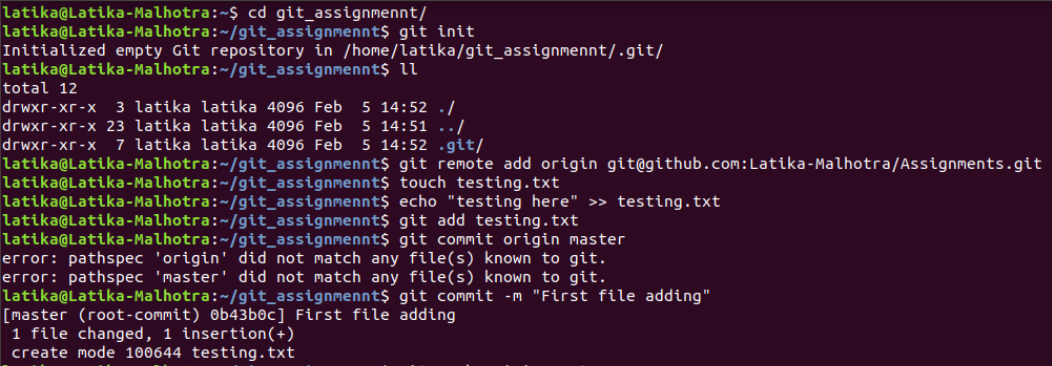
**QUES2-**Initialize a Git Repository

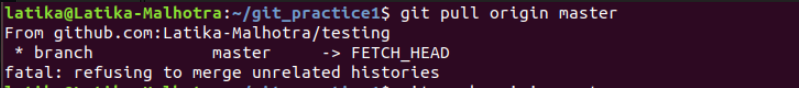
**Solution-**



**QUES3-**Add files to the repository

**Solution-**

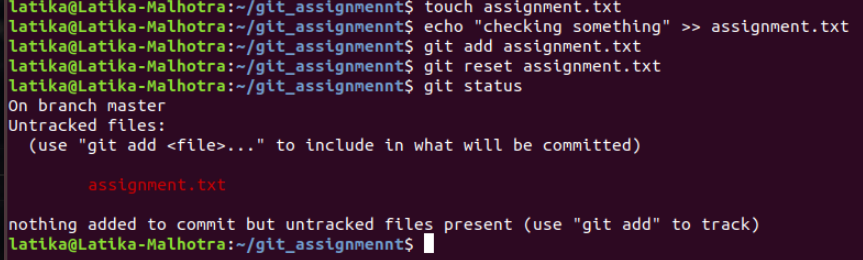






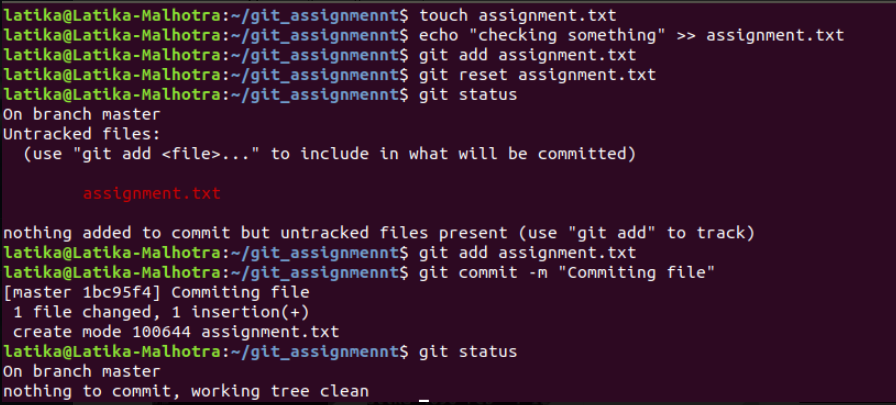
**QUES4-**Unstage 1 file

**Solution-**



**QUES5-**Commit the file

**Solution-**



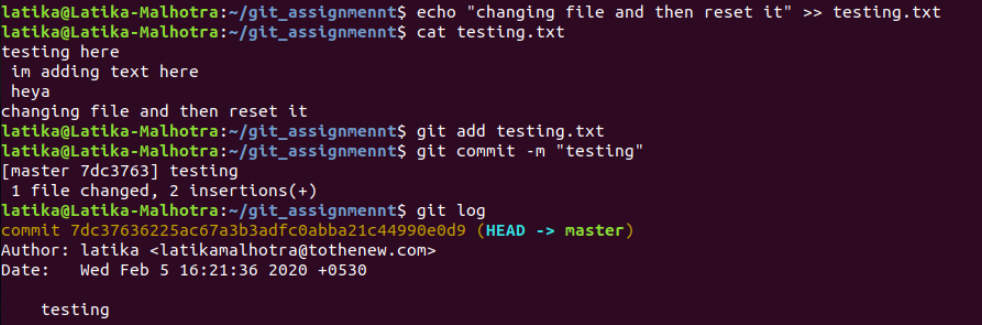
**QUES6-**Add a remote

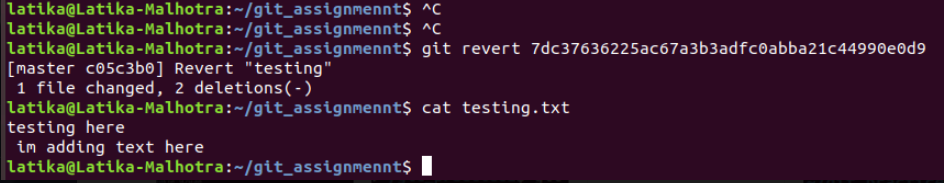
**Solution-**



**QUES7-**Undo changes to a particular file

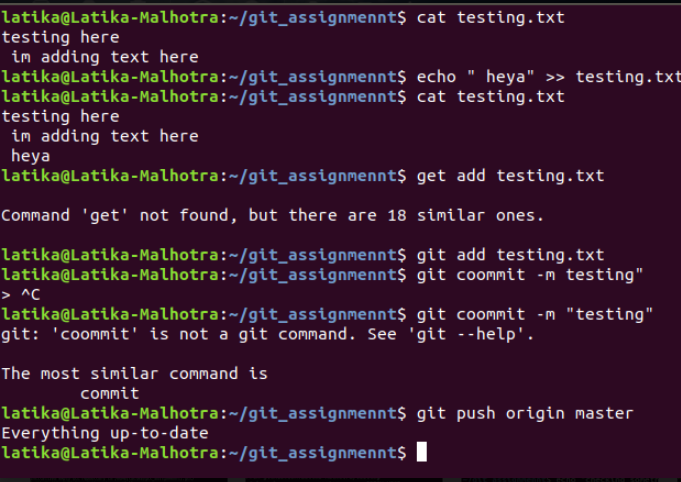
**Solution-**





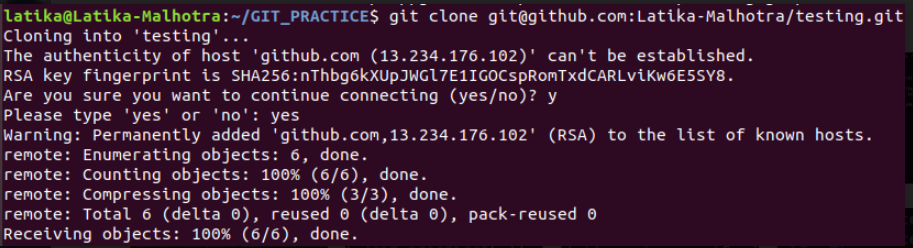
**QUES8-**Push changes to Github

**Solution-**



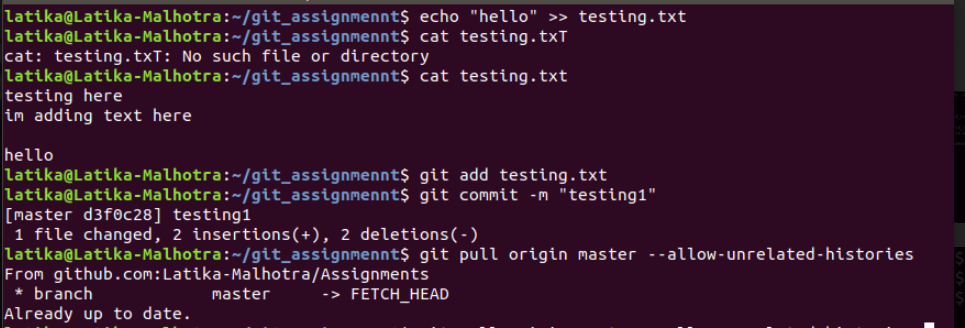
**QUES9-**Clone the repository

**Solution-**



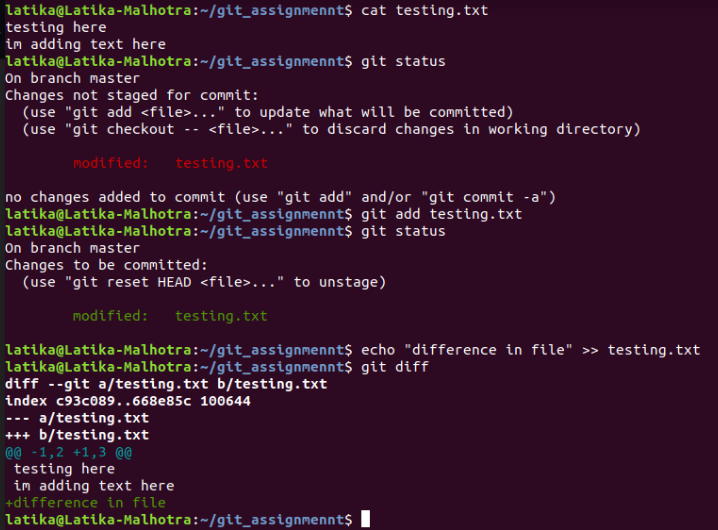
**QUES10-**Add changes to one of the copies and pull the changes in the other.

**Solution-**



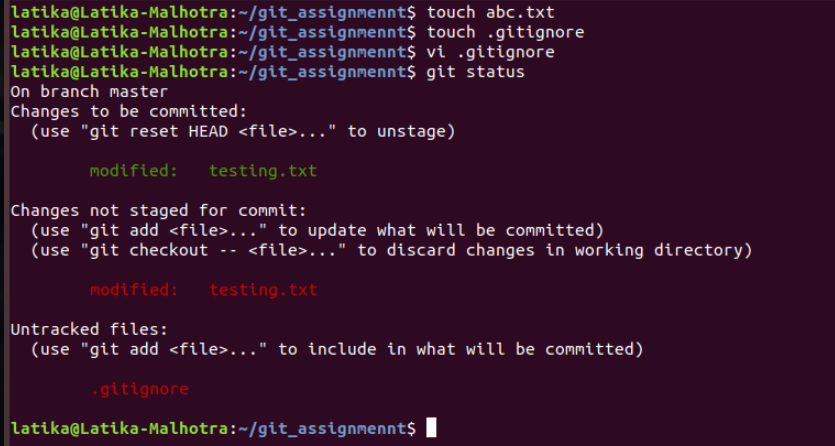
**QUES11-**Check differences between a file and its staged version

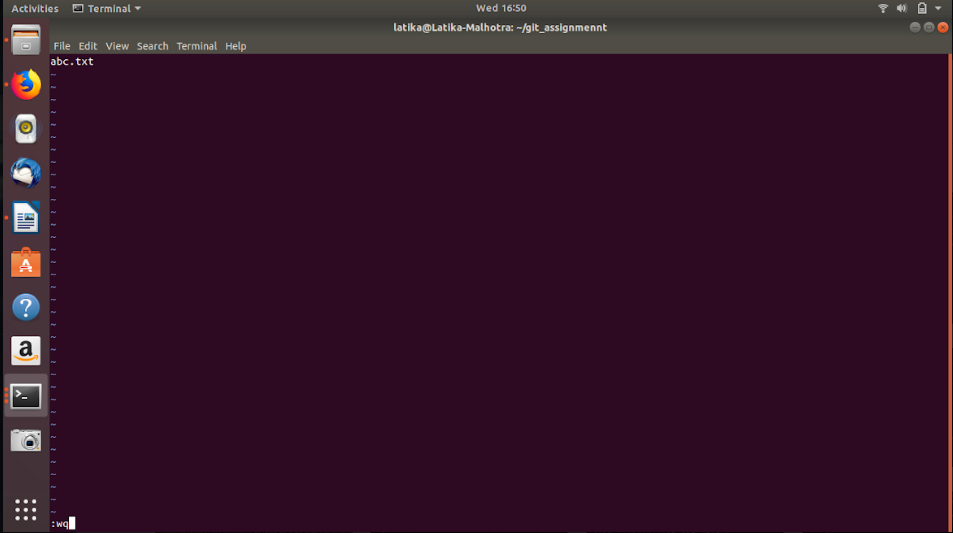
**Solution-**



**QUES12-**Ignore a few files to be checked in

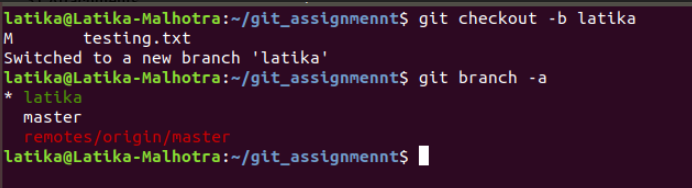
**Solution-**





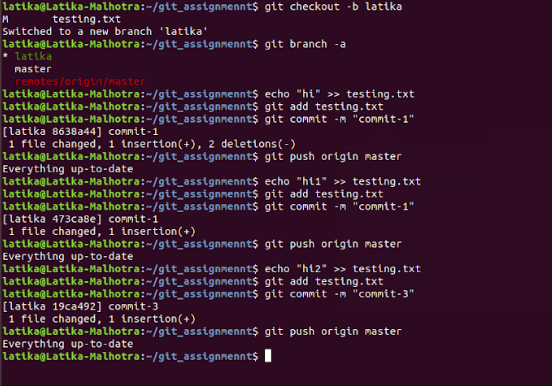
**QUES13-**Create a new branch.

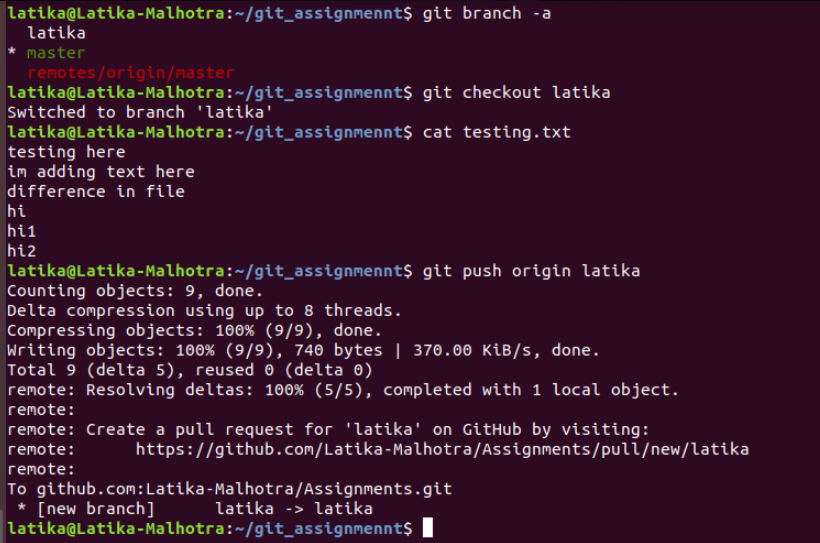
**Solution-**

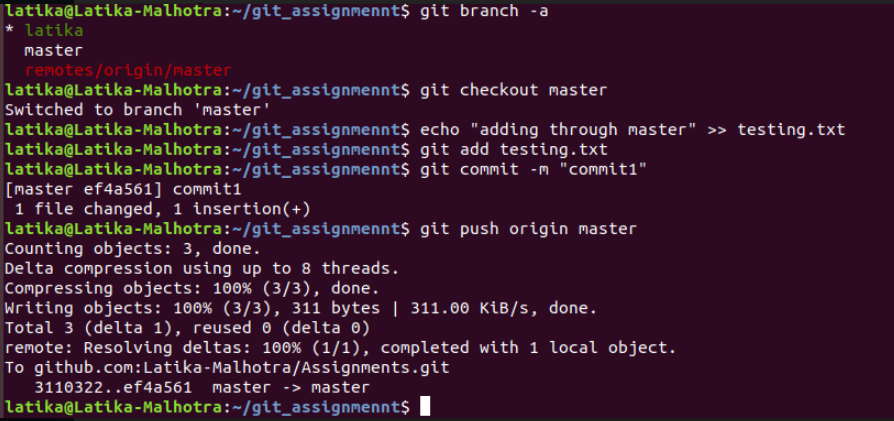


**QUES14-**Diverge them with commits

**Solution-**

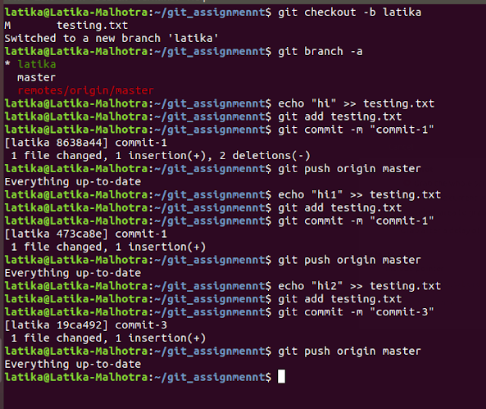


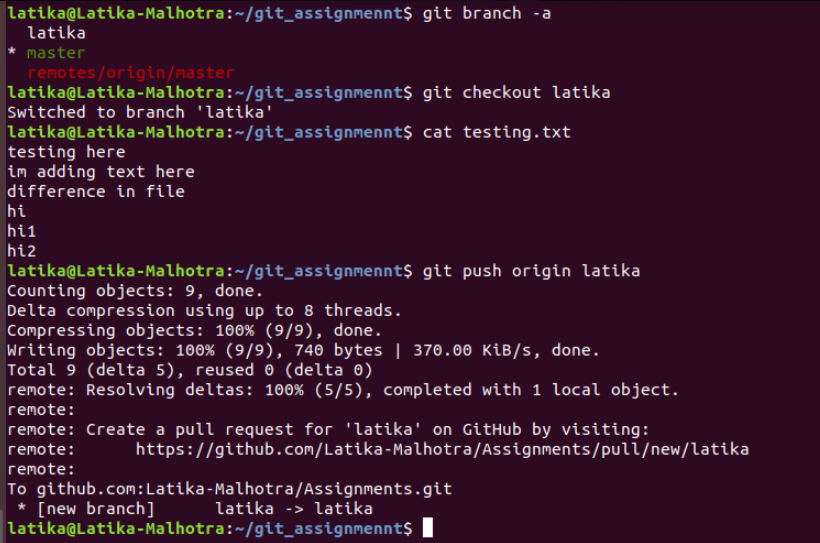


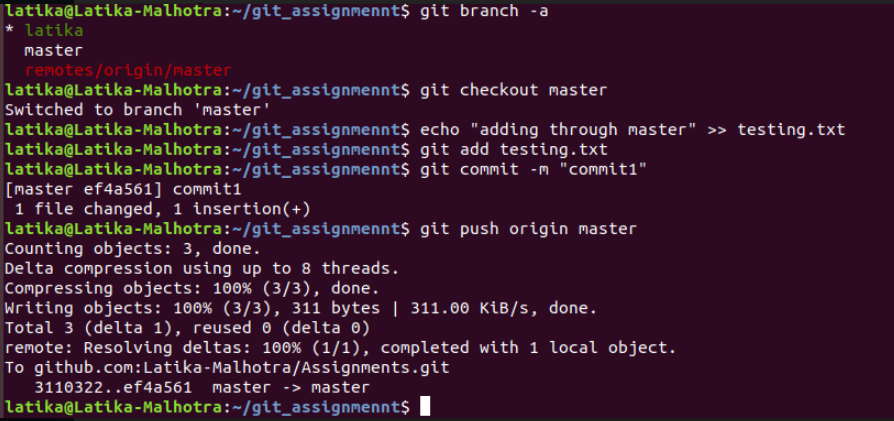


**QUES15-**Edit the same file at the same line on both branches and commit

**Solution-**

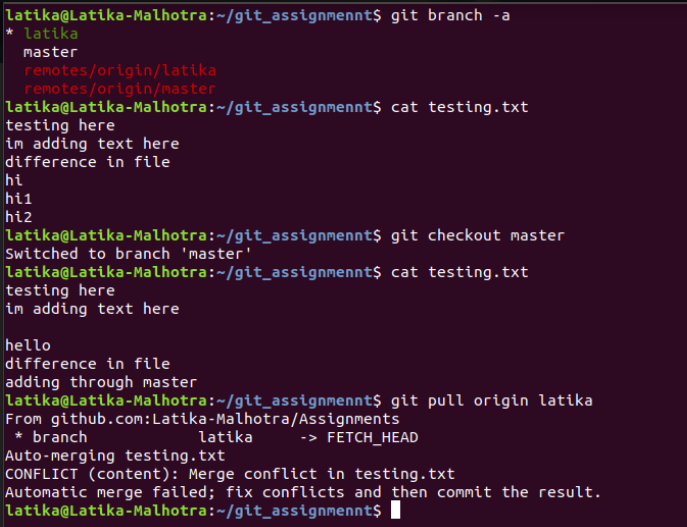


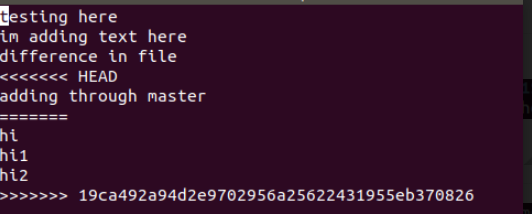


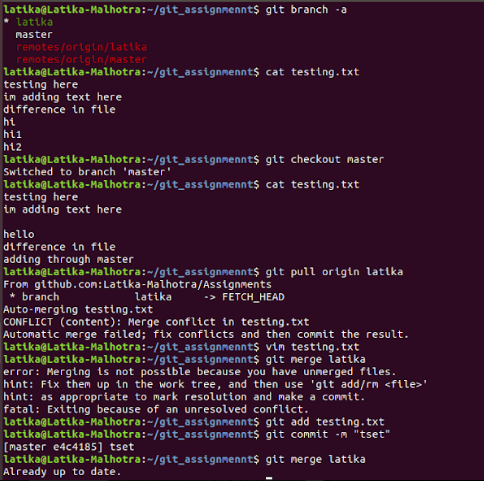


**QUES16-**Try merging and resolve merge conflicts

**Solution-**

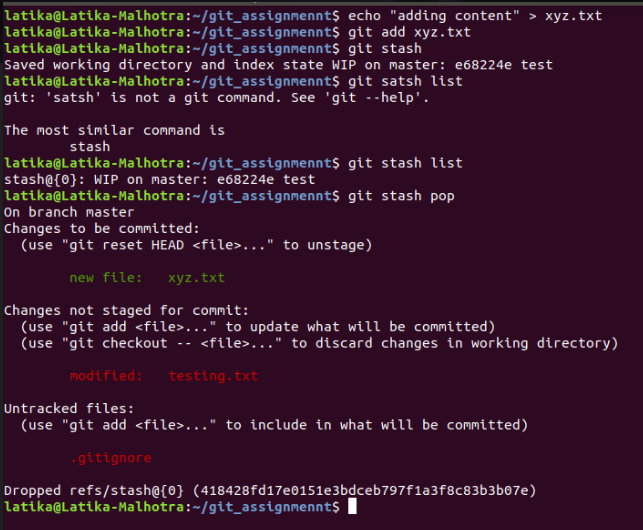






**QUES17-**Stash the changes and pop them

**Solution-**



**QUES18-**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[  
else  
PS1='\u@\h:\W $(parse\_git\_branch)\$ '  
fi  
unset color\_prompt force\_color\_prompt

**Solution-**

