Exemple de comenzi git, utilizare SourseTree (material ajutător)

Contents

[1. Create a local repository in lab4 3](#_Toc86021470)

[2. Create SSH Key if it’s necessary 3](#_Toc86021471)

[3. Create remote GitHub public repo 6](#_Toc86021472)

[4. Solving the conflict using the text editor 14](#_Toc86021473)

[5. git merge --squash 22](#_Toc86021474)

[6. git revert 27](#_Toc86021475)

[7. git push origin/master 30](#_Toc86021476)

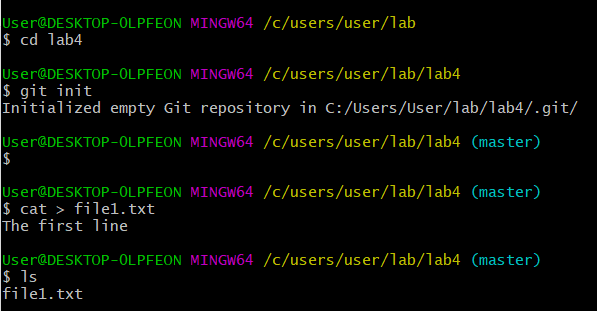
[8. git rebase 31](#_Toc86021477)

[9. git –amend 37](#_Toc86021478)

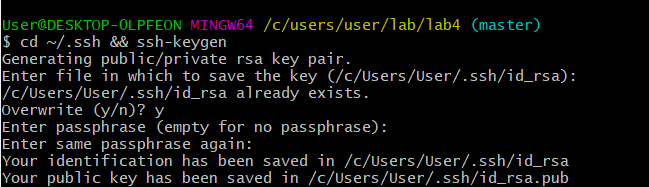
[10. git cherry-pick 38](#_Toc86021479)

[11. git reset 39](#_Toc86021480)

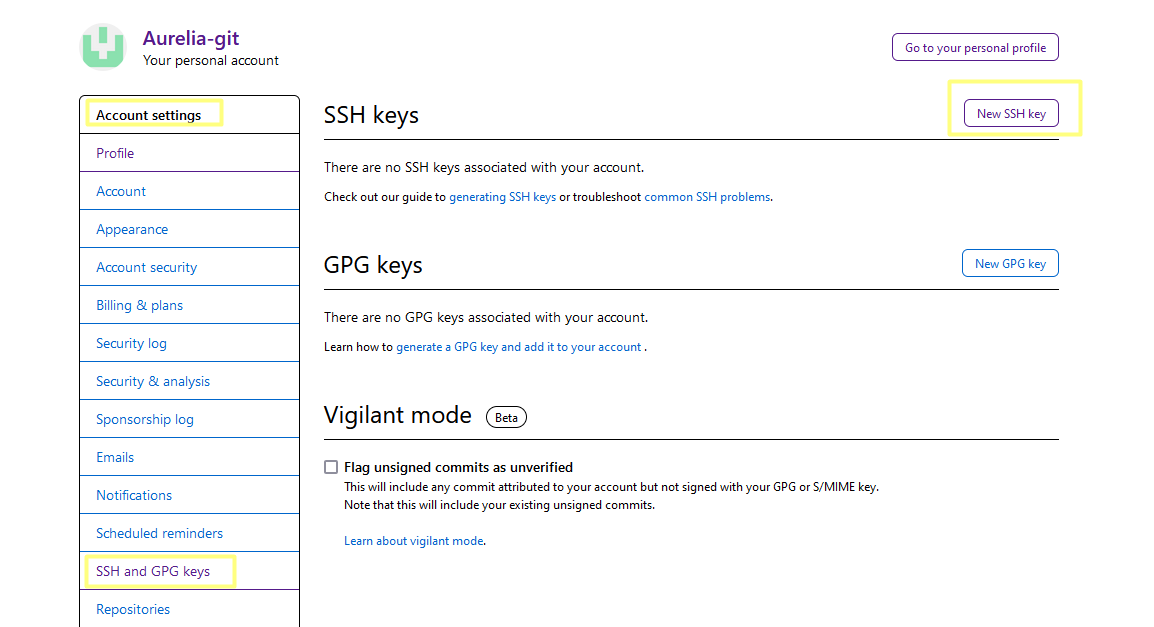
# Create a local repository in lab4



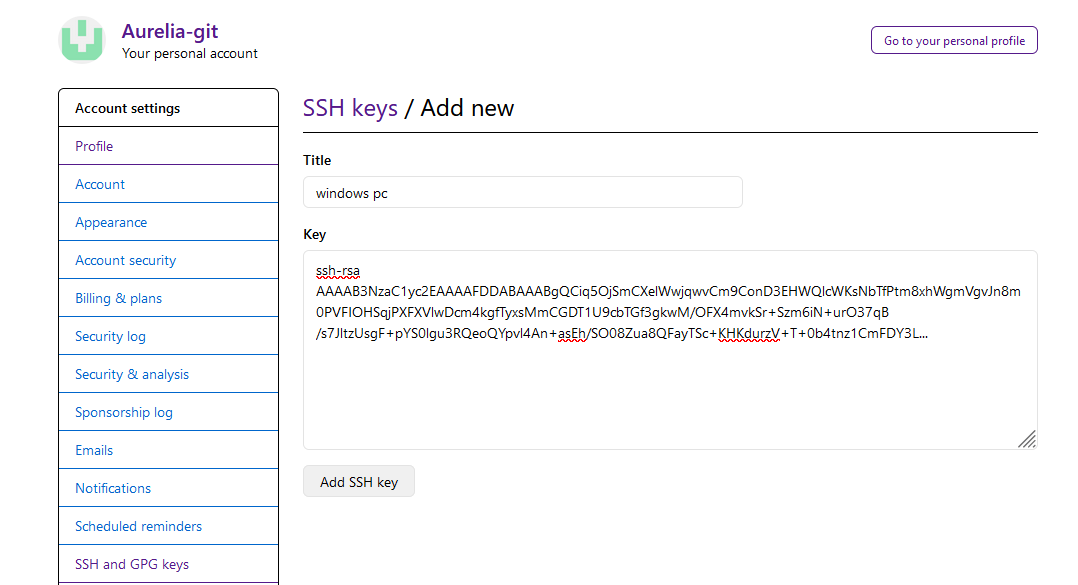
# Create SSH Key, if it’s necessary



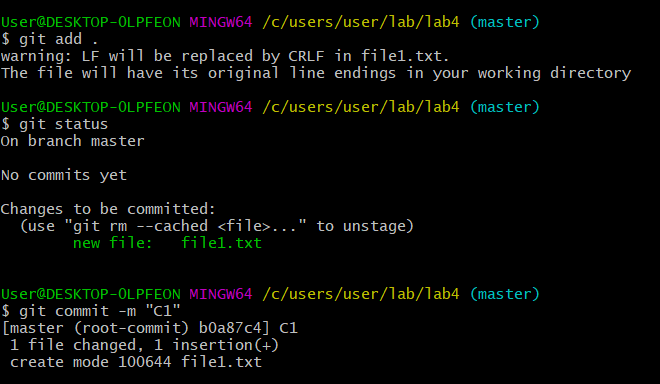
If you didn’t introduce another name for the key, your public key has been saved in id\_rsa.pub file



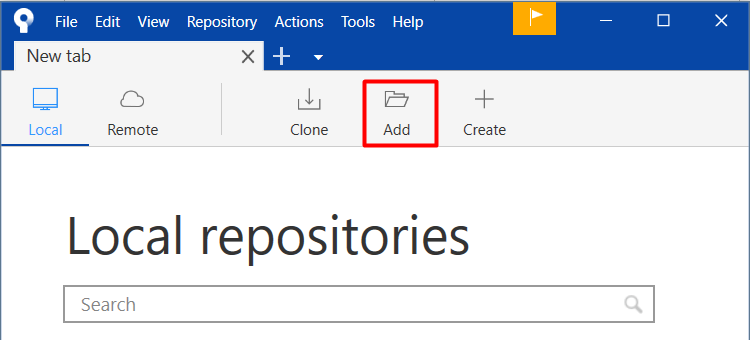
Copy the content of id\_rsa.pub to github and press “Add SSH key”:

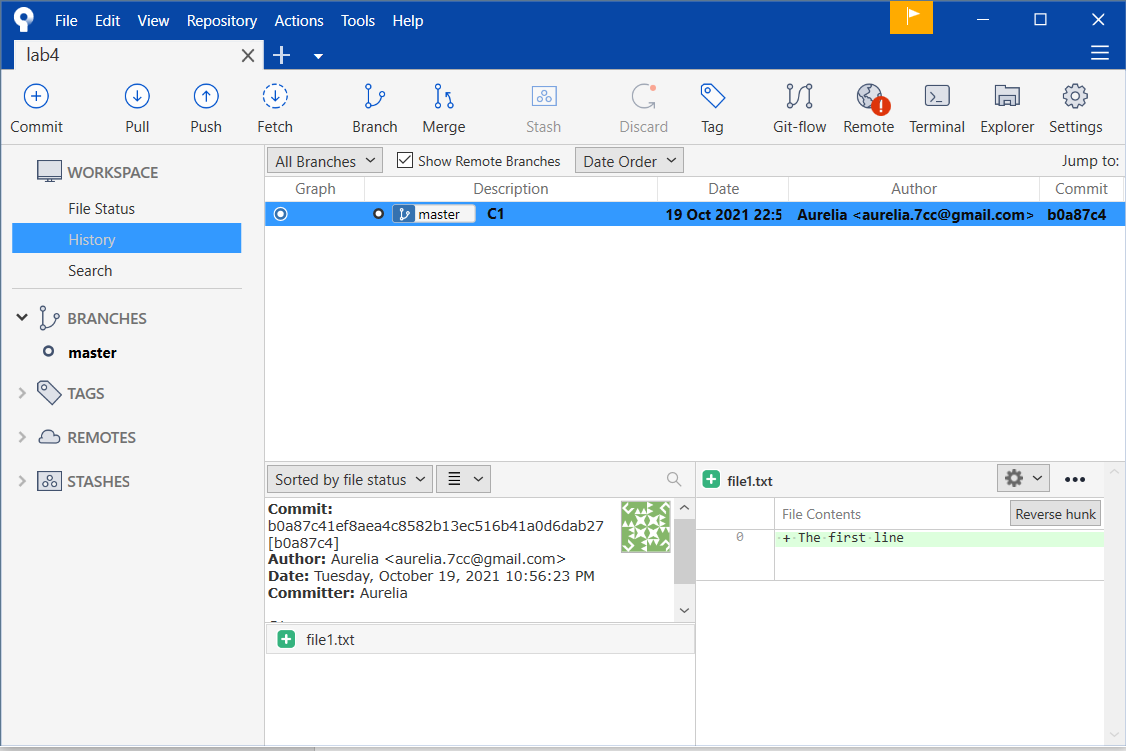


Create the first commit – C1 in the local repo:

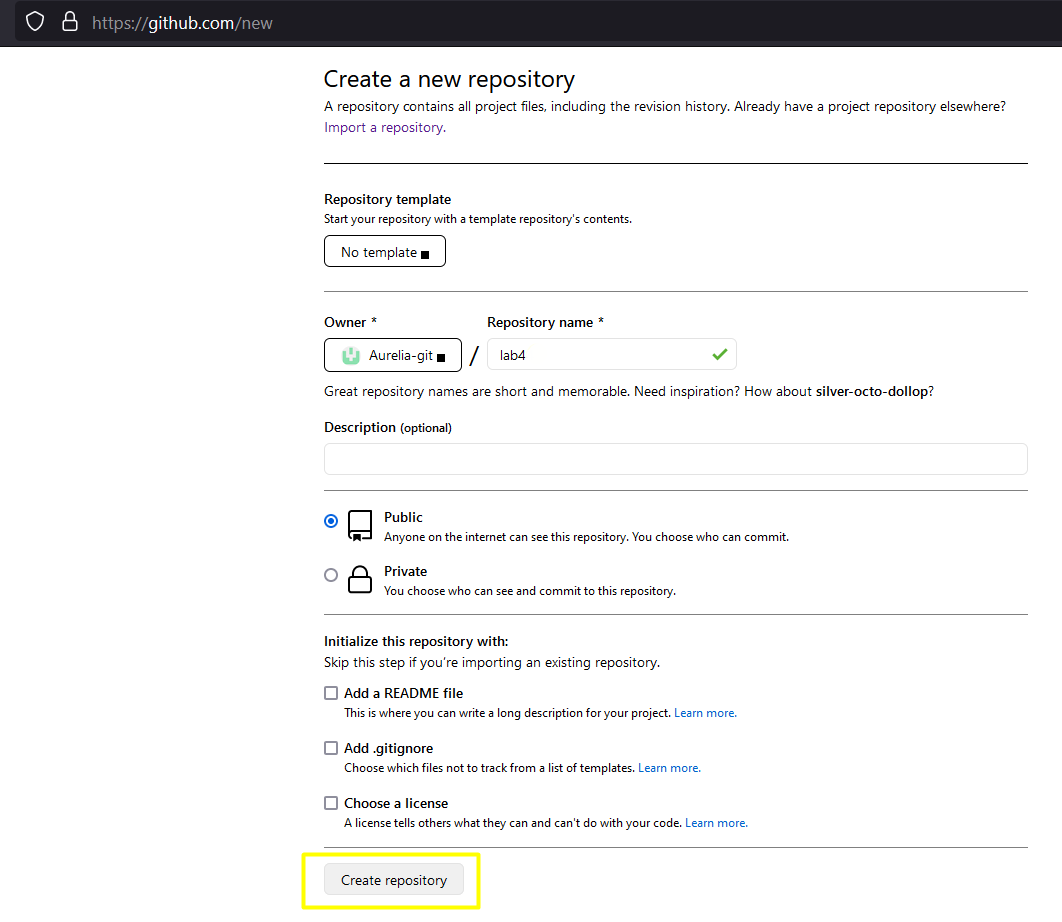


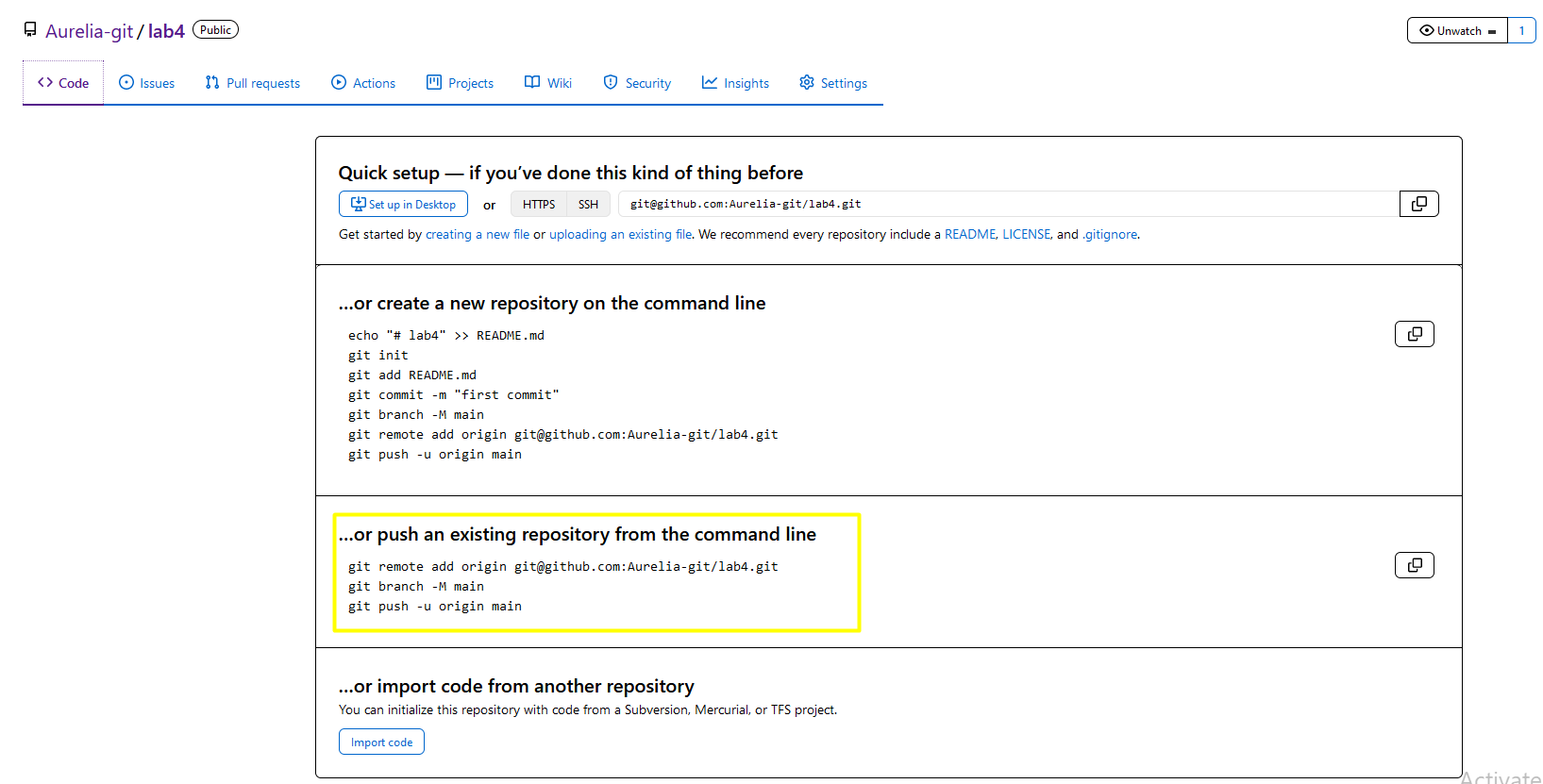
Add a local repo (lab4) to SourceTree:





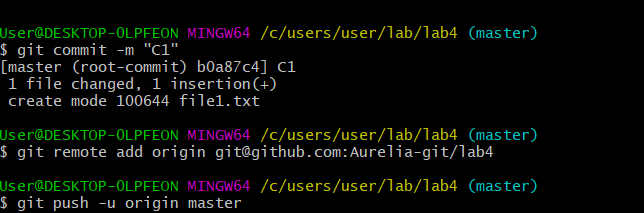
# Create remote GitHub public repo





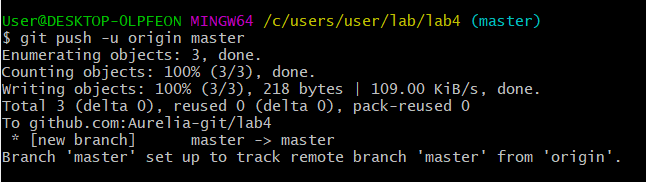
Add a new remote repo:

Branch 'master' set up to track remote branch 'master' from 'origin'.

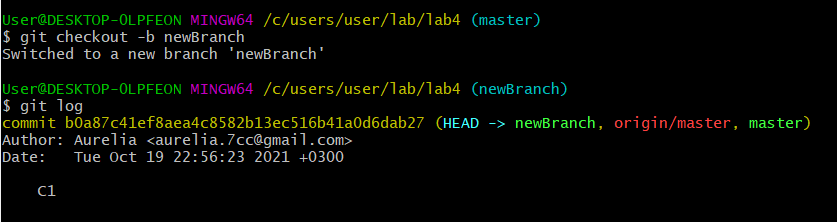


To add a new remote, use the **git remote add** command on the terminal, in the directory your repository is stored at.

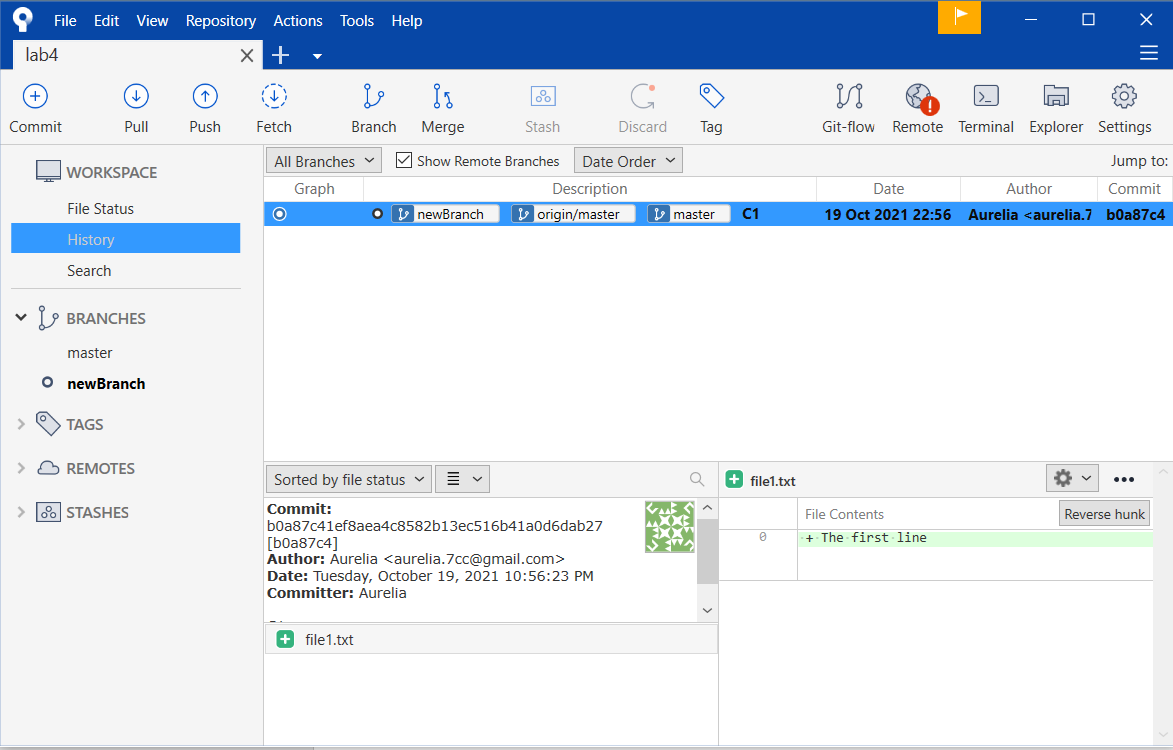
**git push -u origin master** is used for pushing local content to GitHub. In the code, the origin is your default remote repository name and '-u' flag is upstream, which is equivalent to '-set-upstream. ' and the master is the branch name.

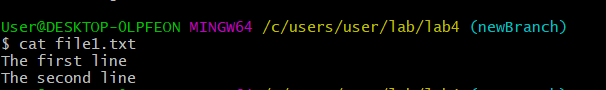


**Now we create a new branch newBranch and HEAD will point to it:**

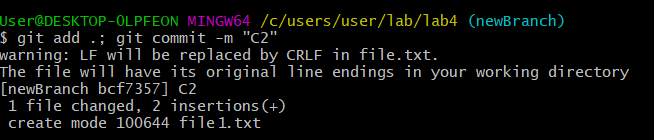


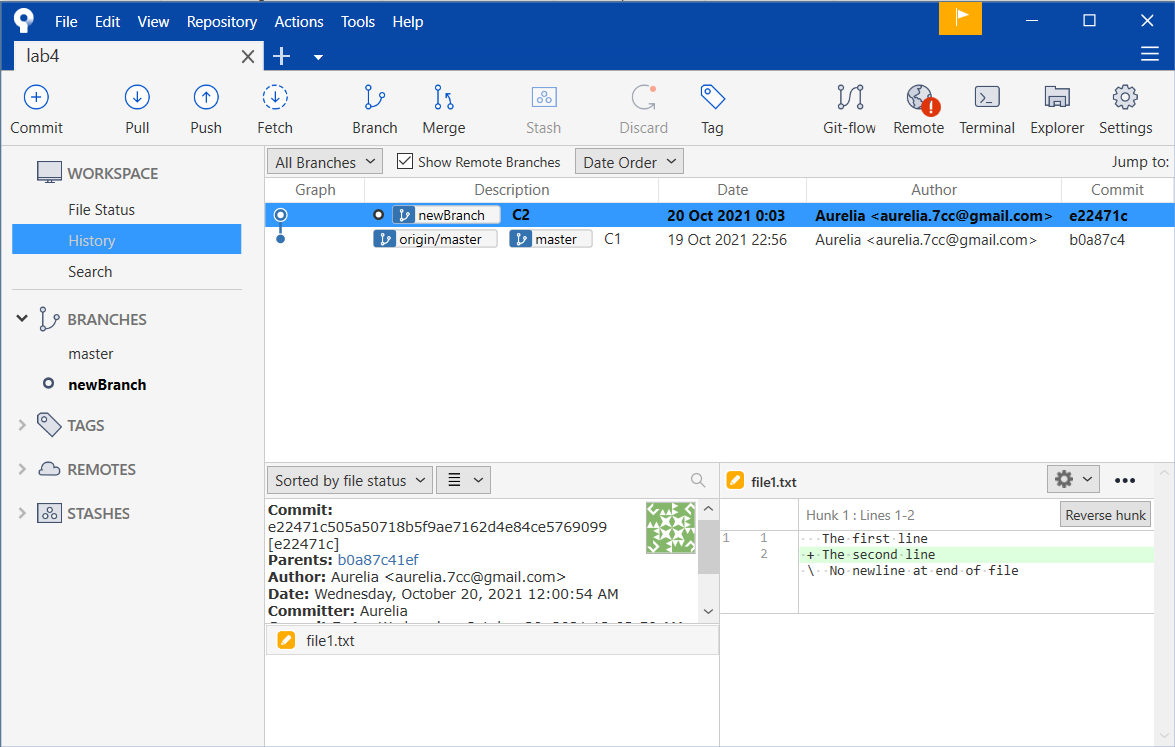
All our branches (master, origin/master and newBranch) point to the same commit C1:

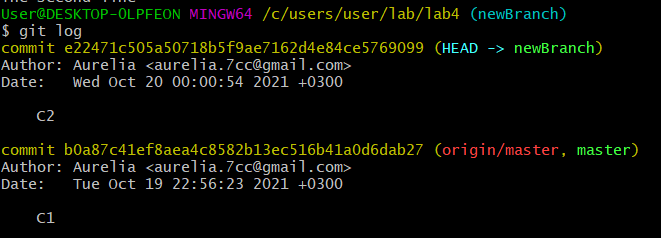




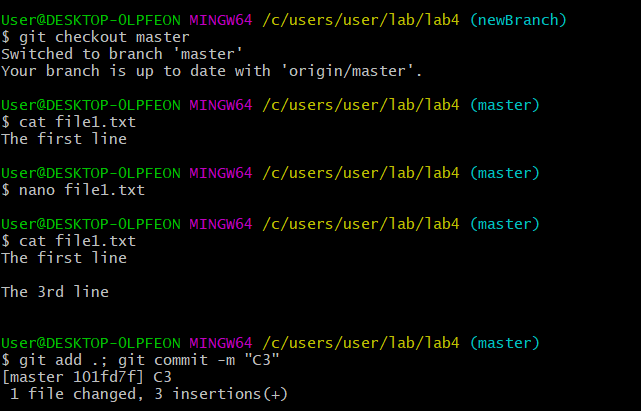
Create the commit C2 on newBranch:

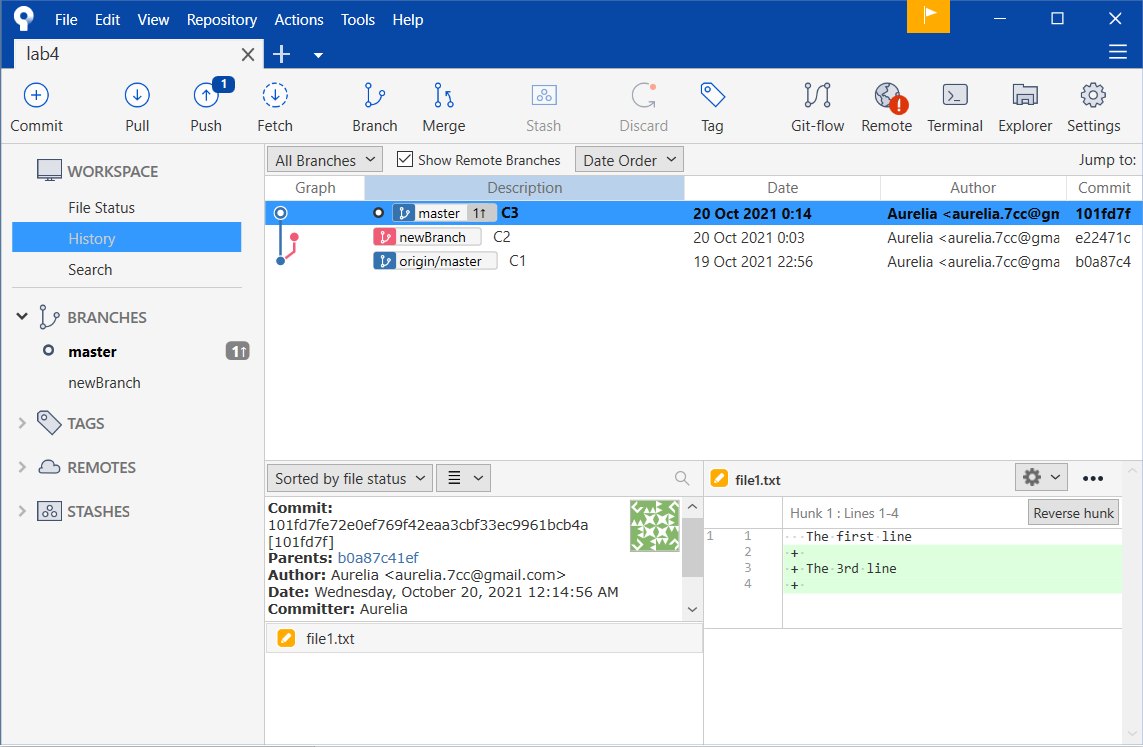




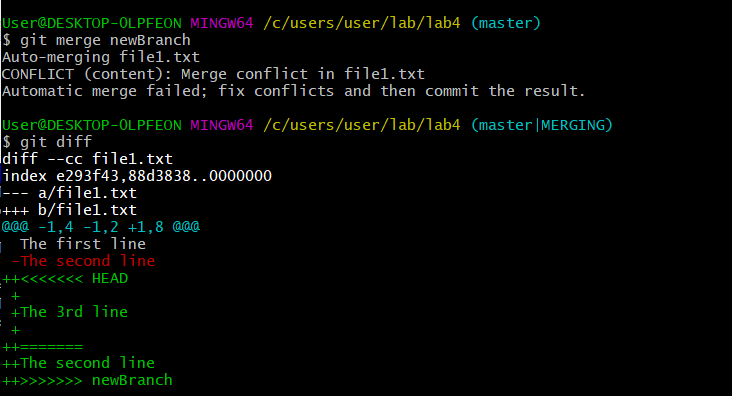


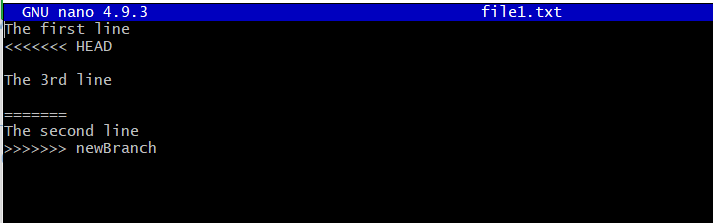
Checkout master branch and make C3 commit





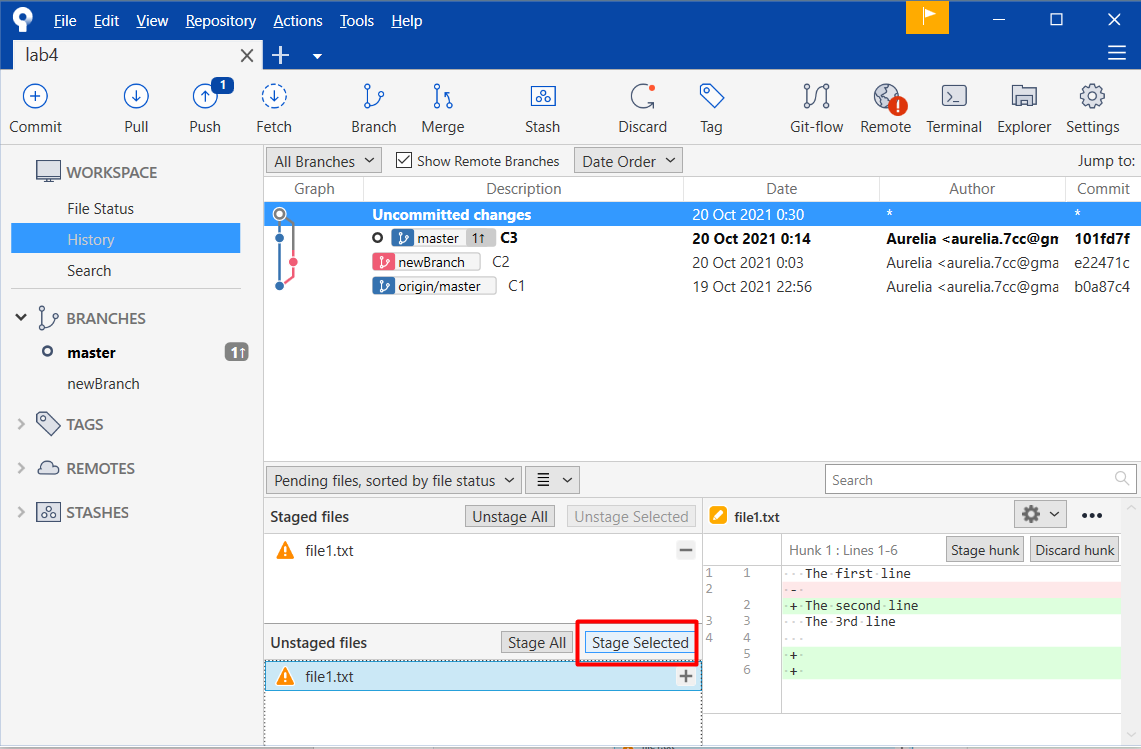
**git merge:**

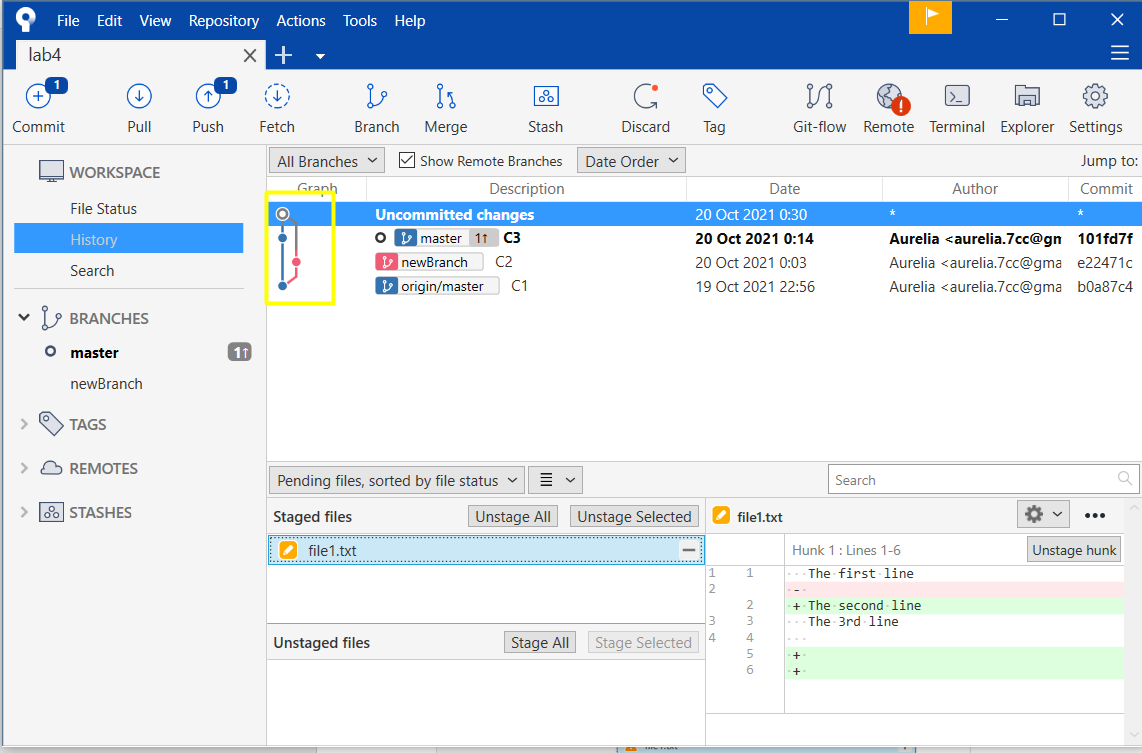


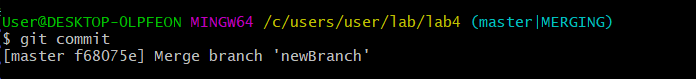


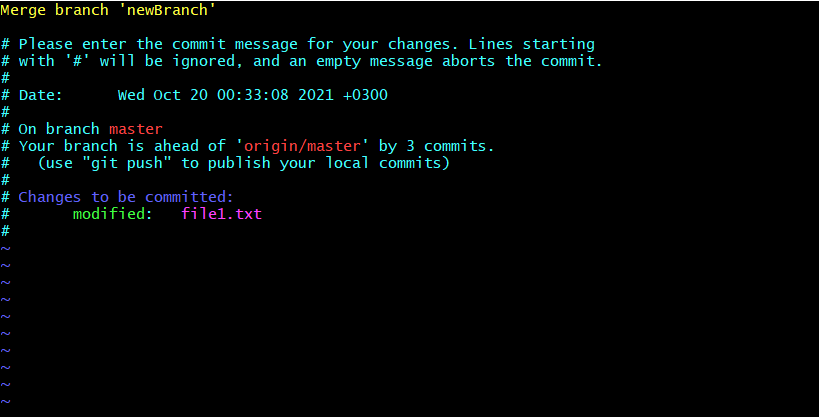
# Solving the conflict using the text editor

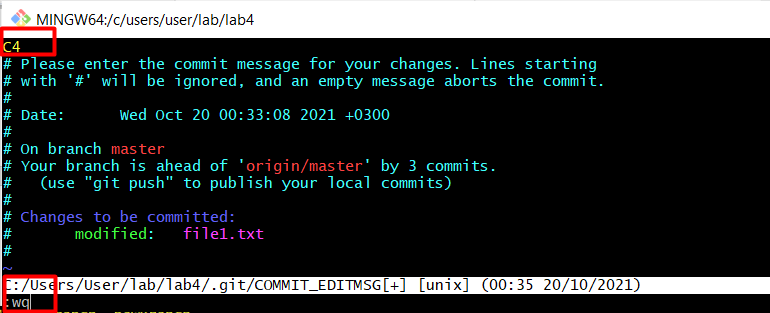


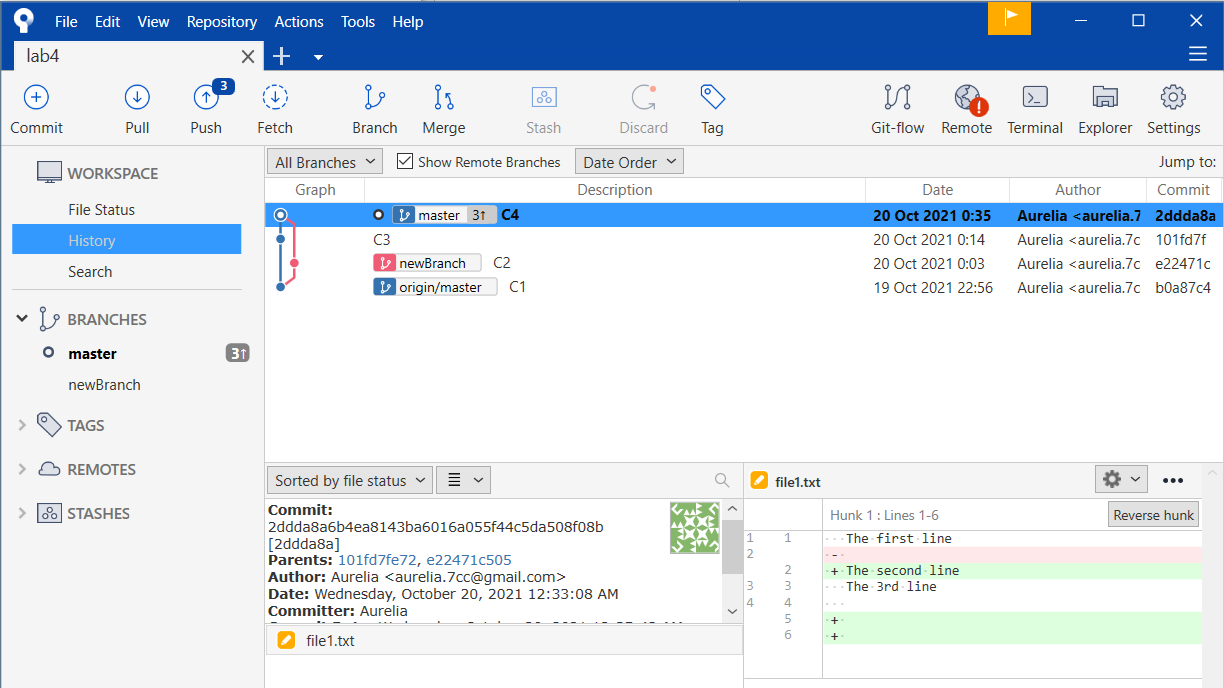




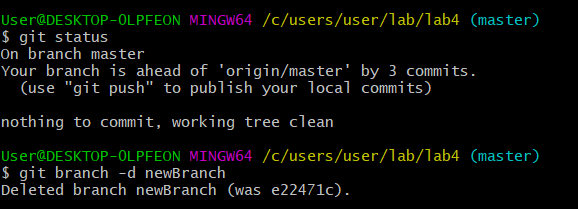


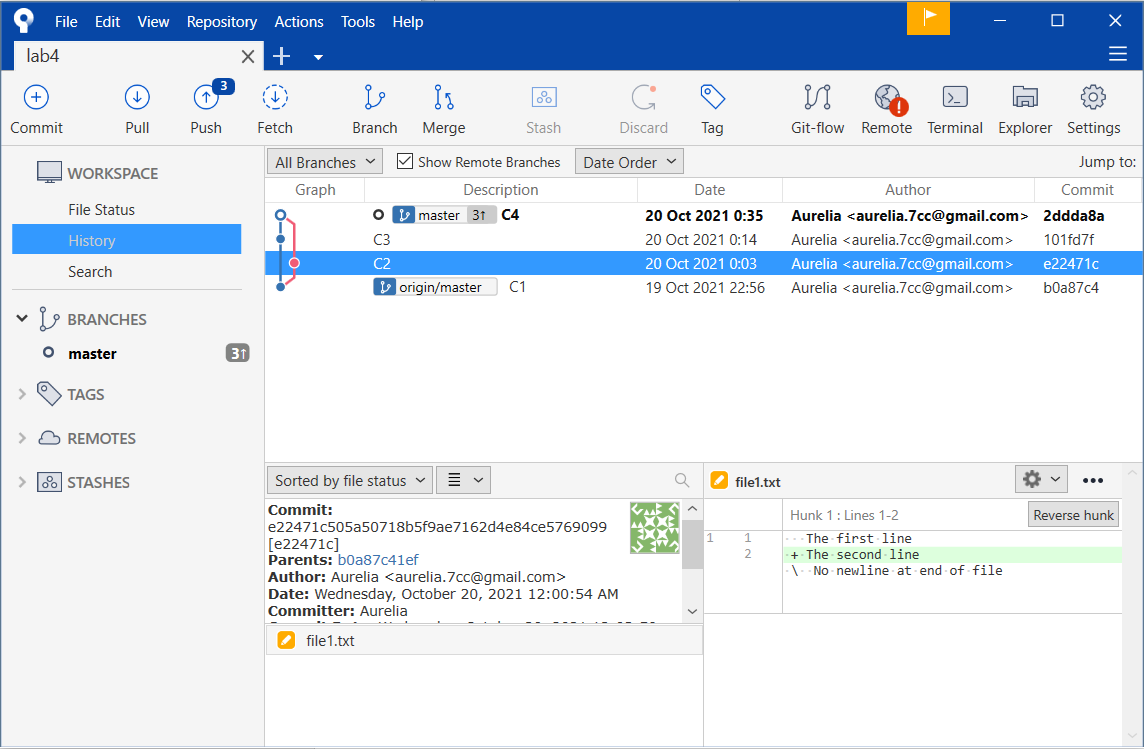




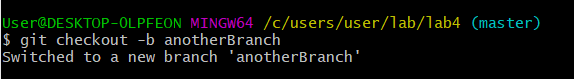


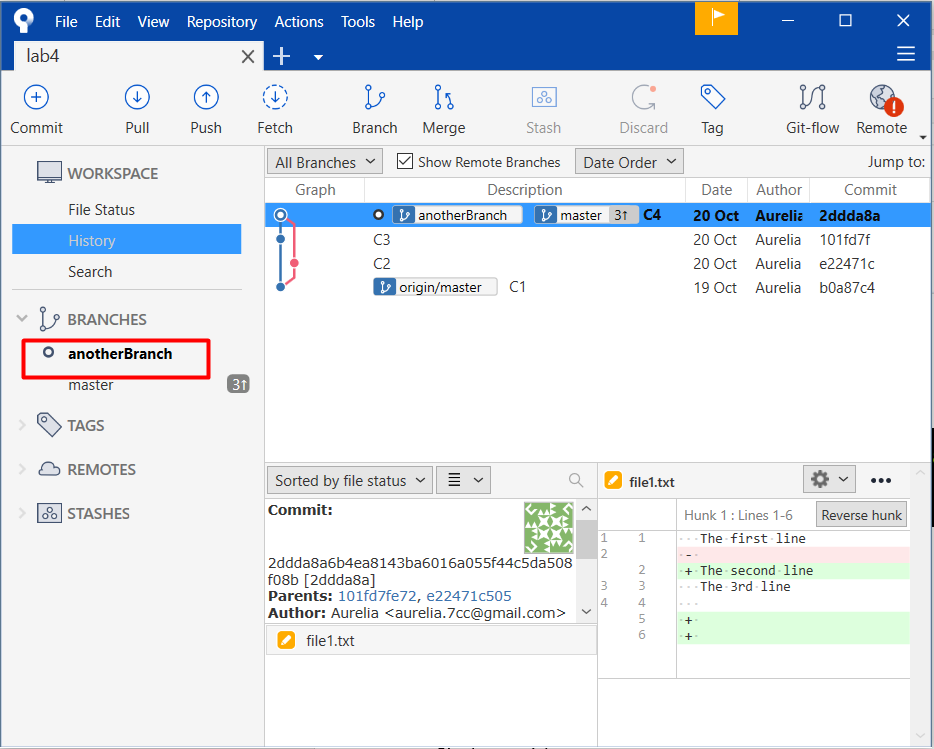
Deleting newBranch:

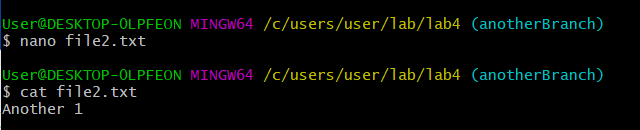


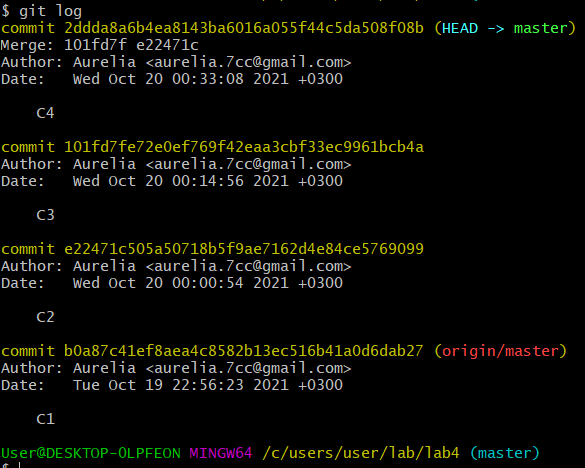


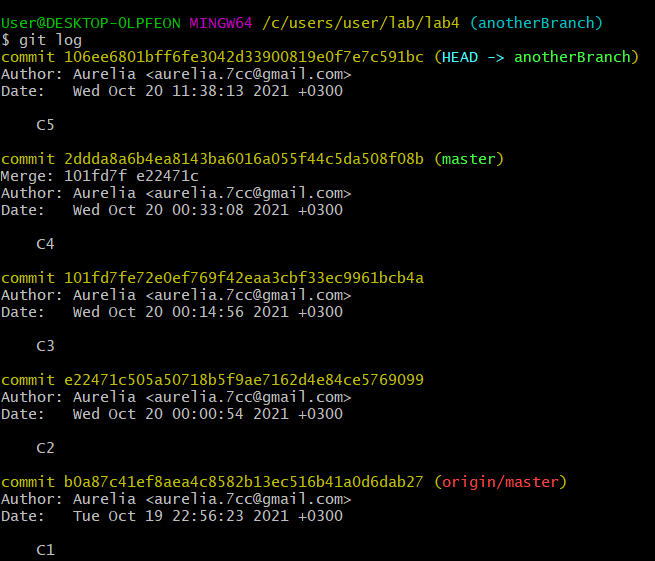
Make anotherBranch:

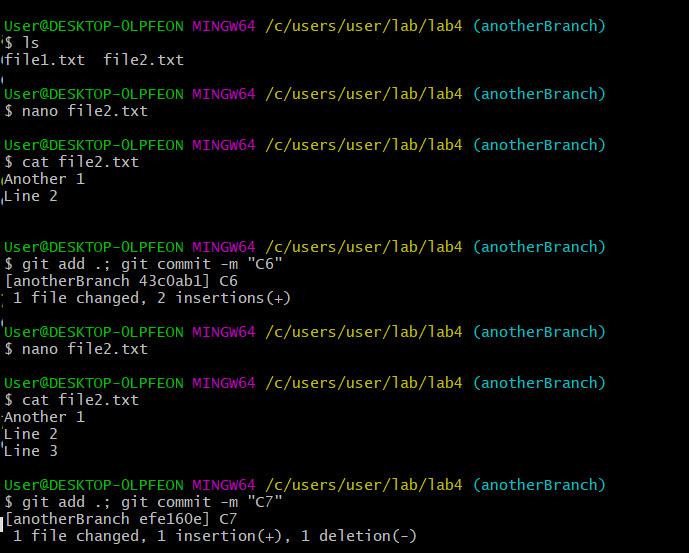


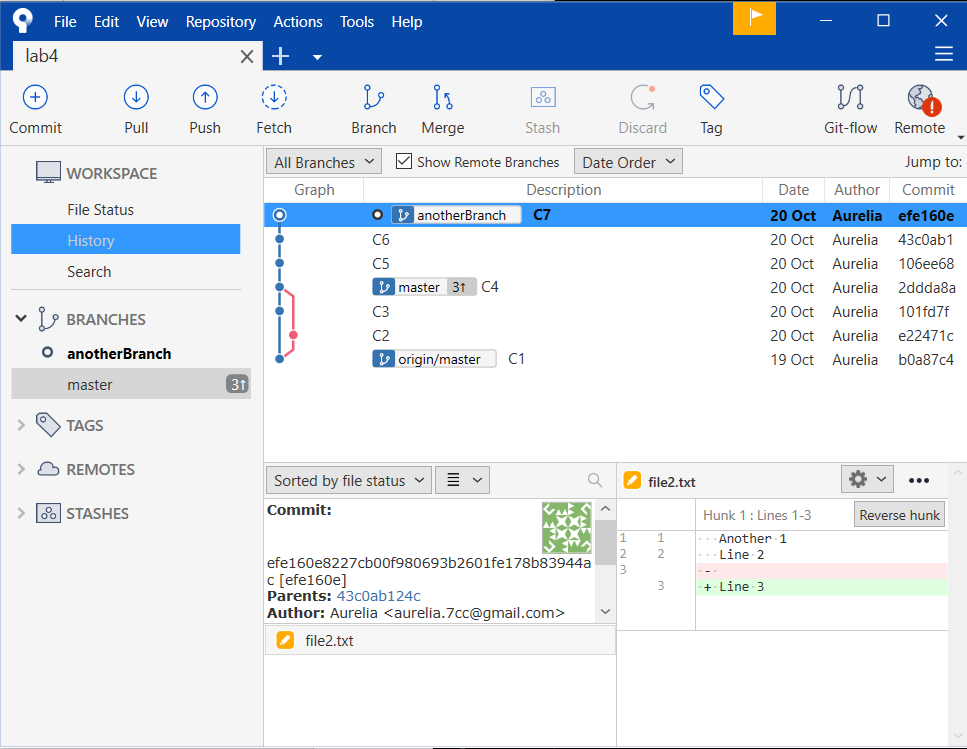




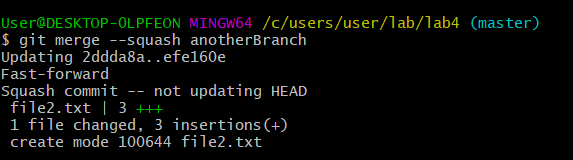


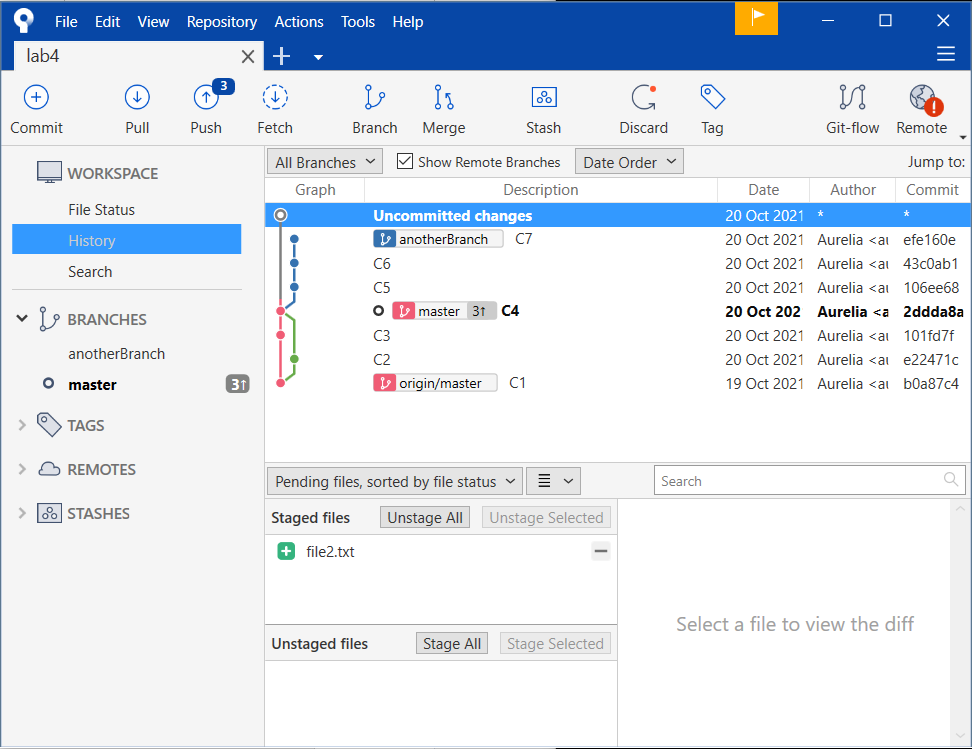




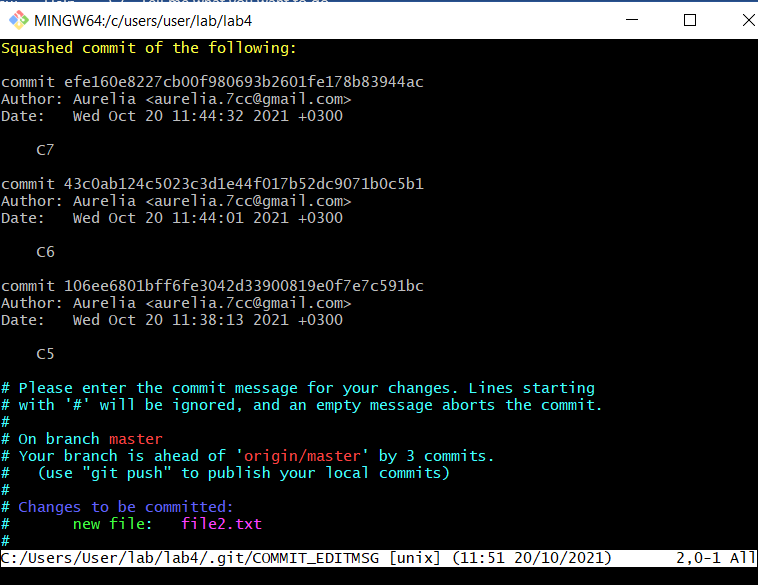


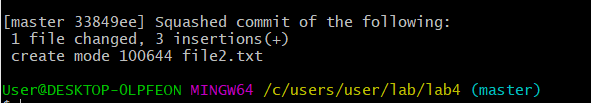
# git merge --squash

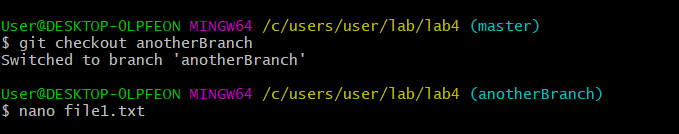


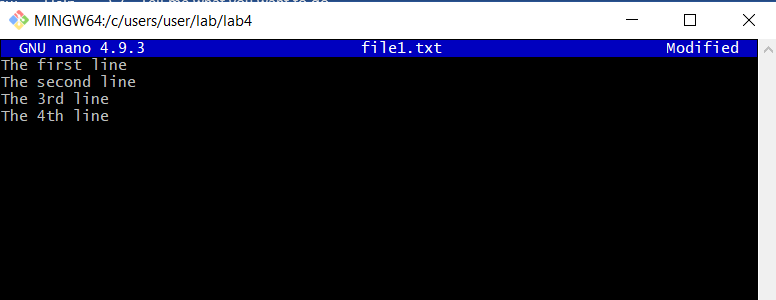


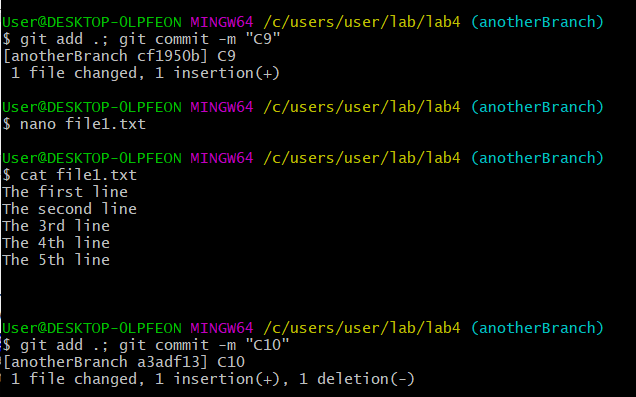
git commit

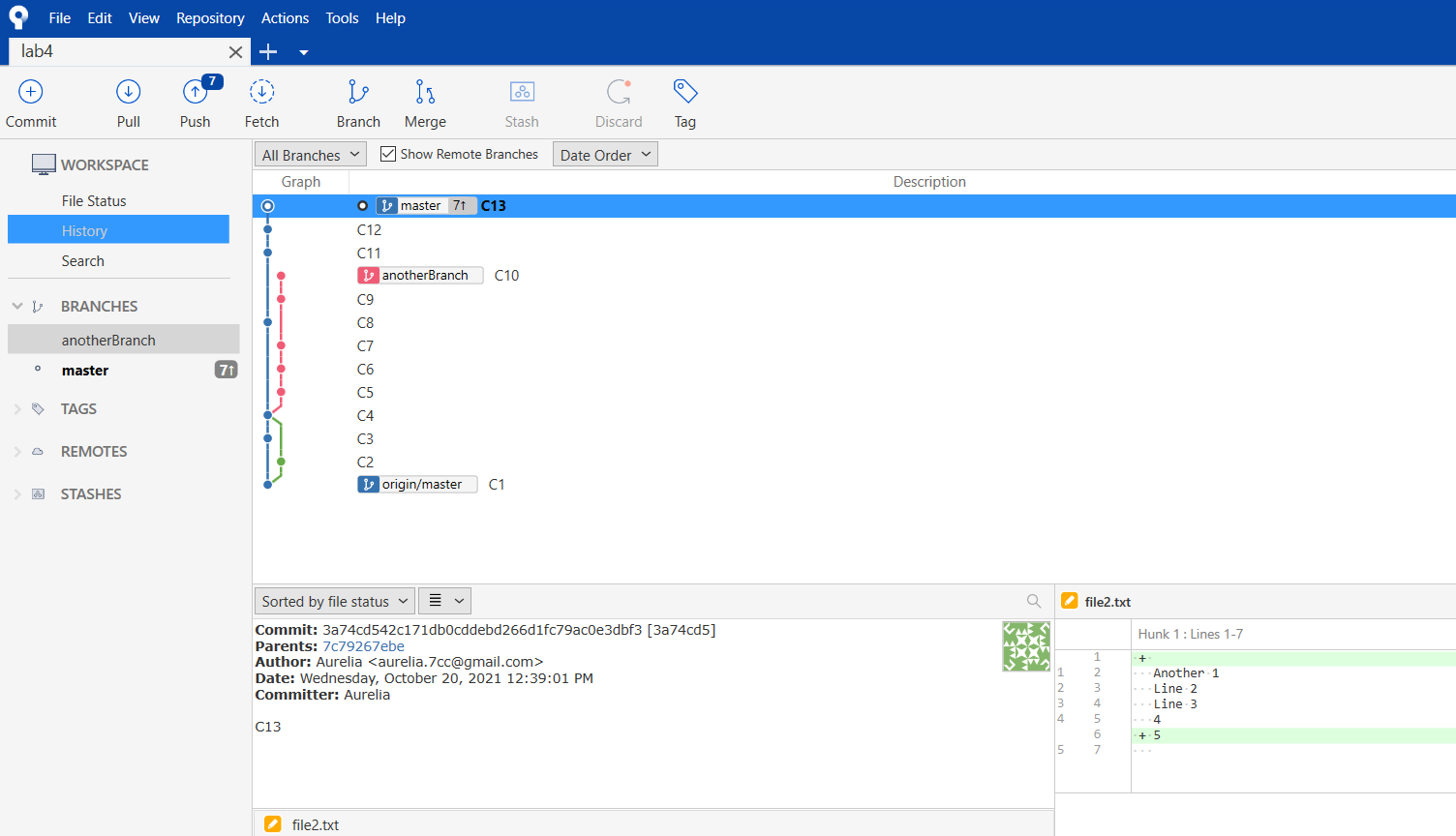


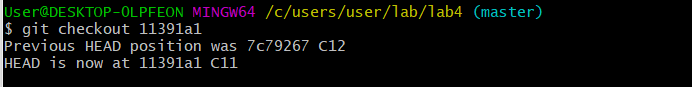


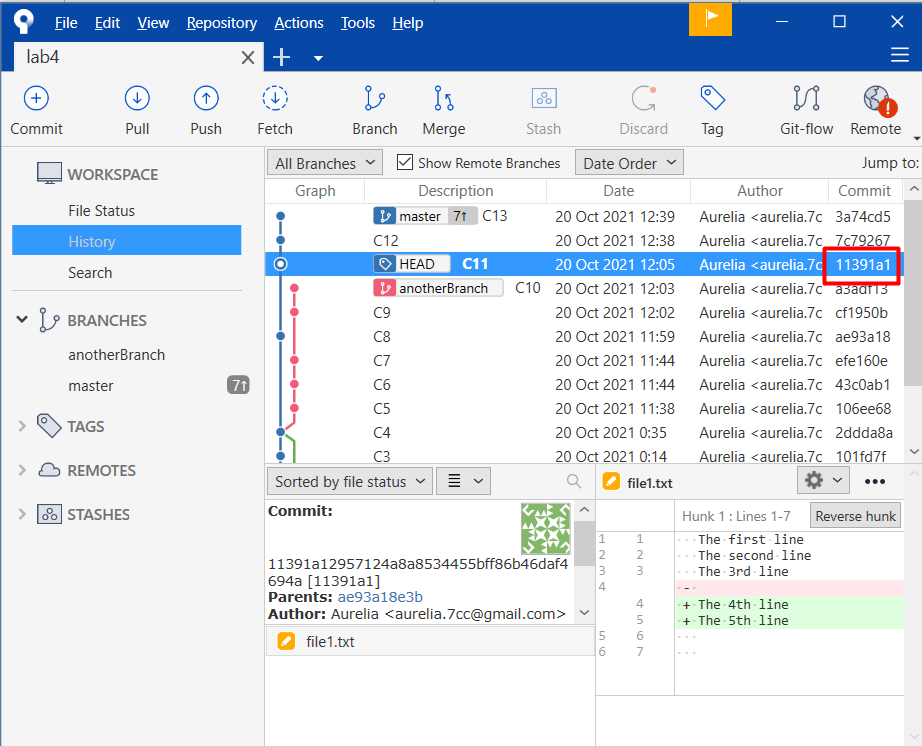






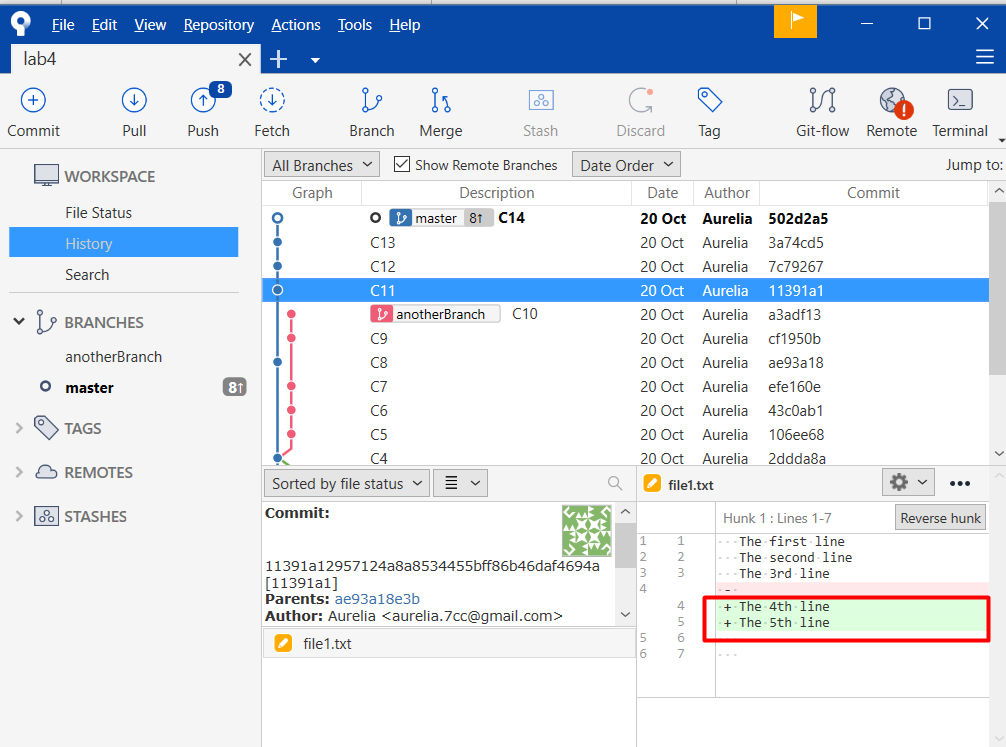


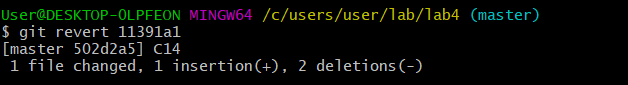


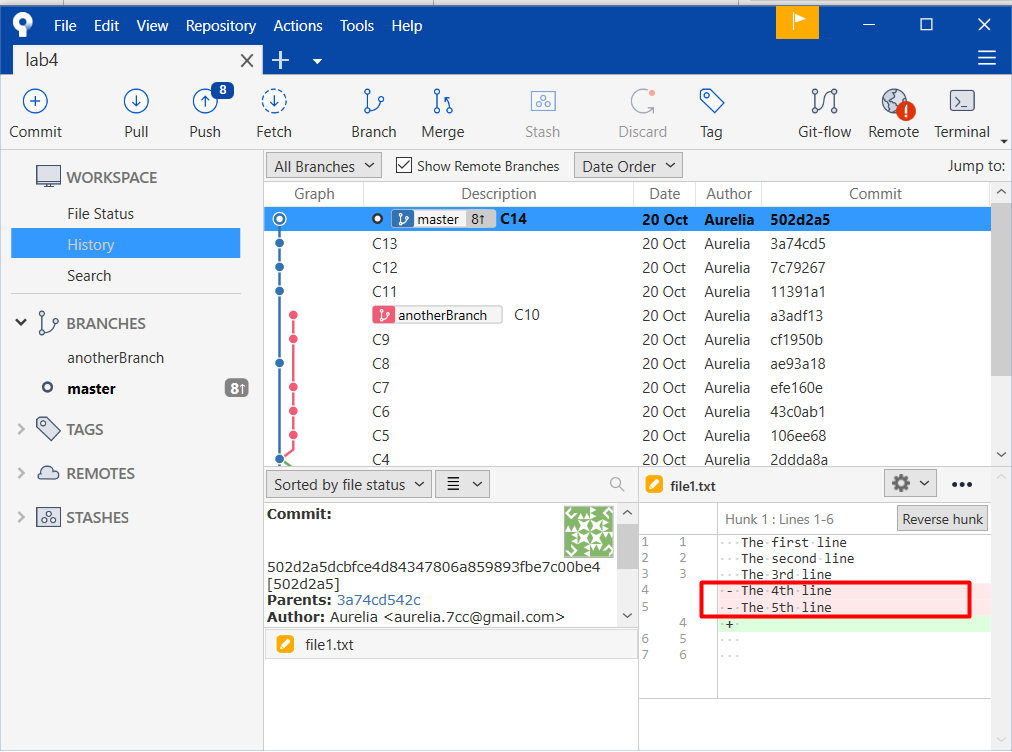


Or double click from Sourcetree

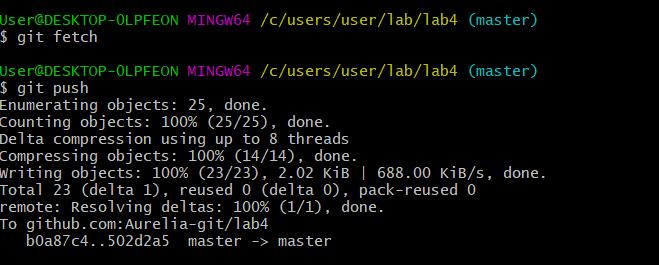
# git revert



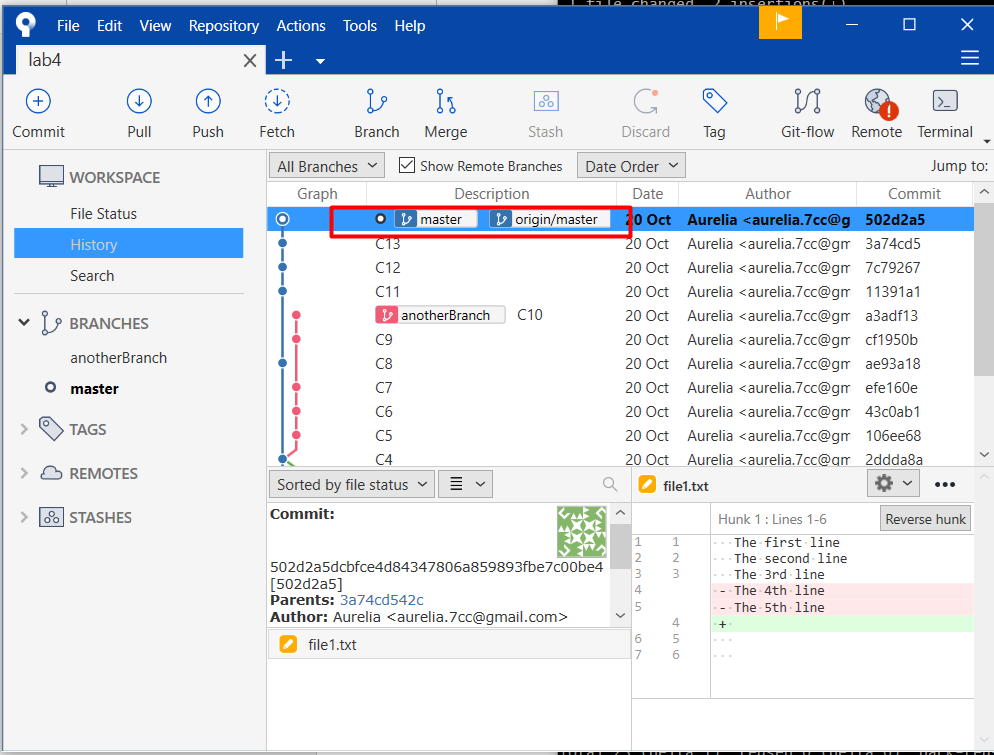




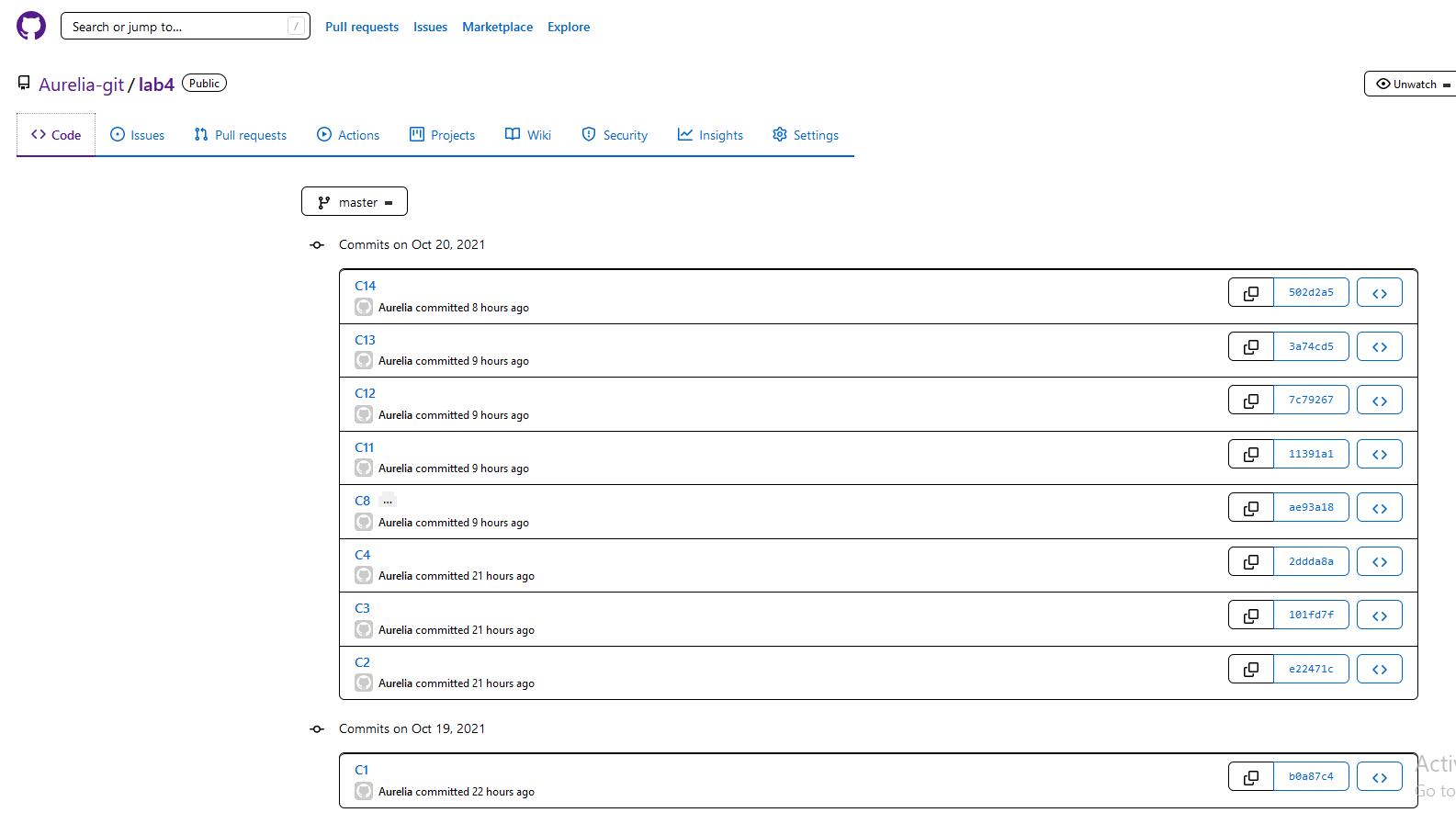
From the C14 2 lines have been deleted, C14 contains a revert of commit C11:



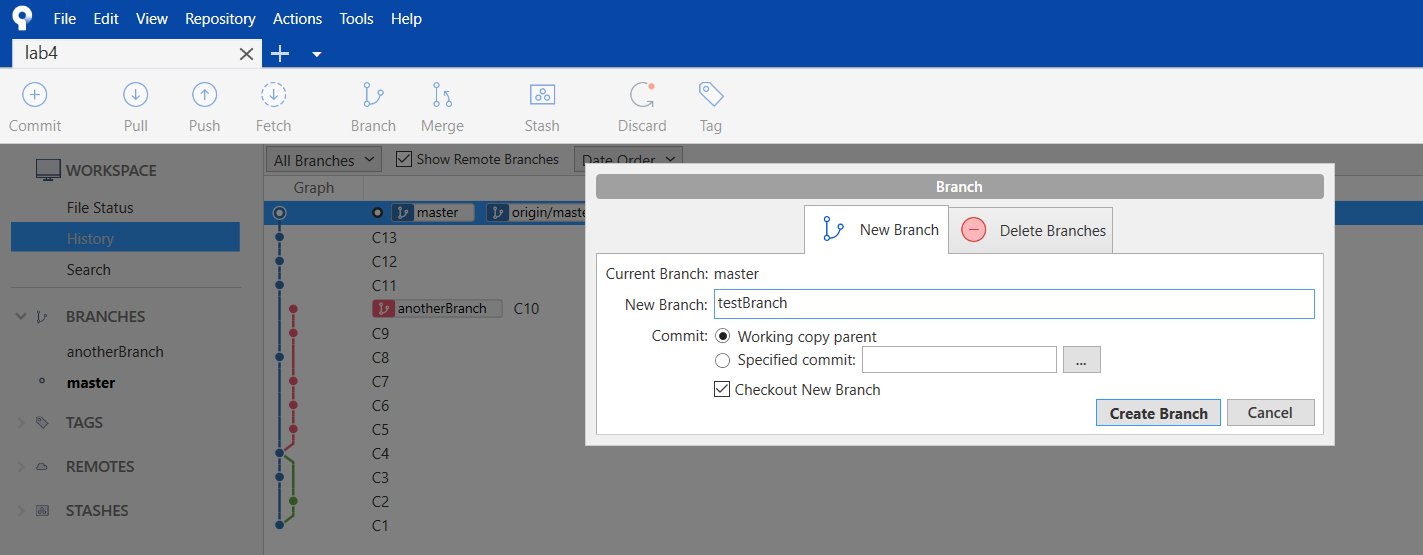
# git push origin/master

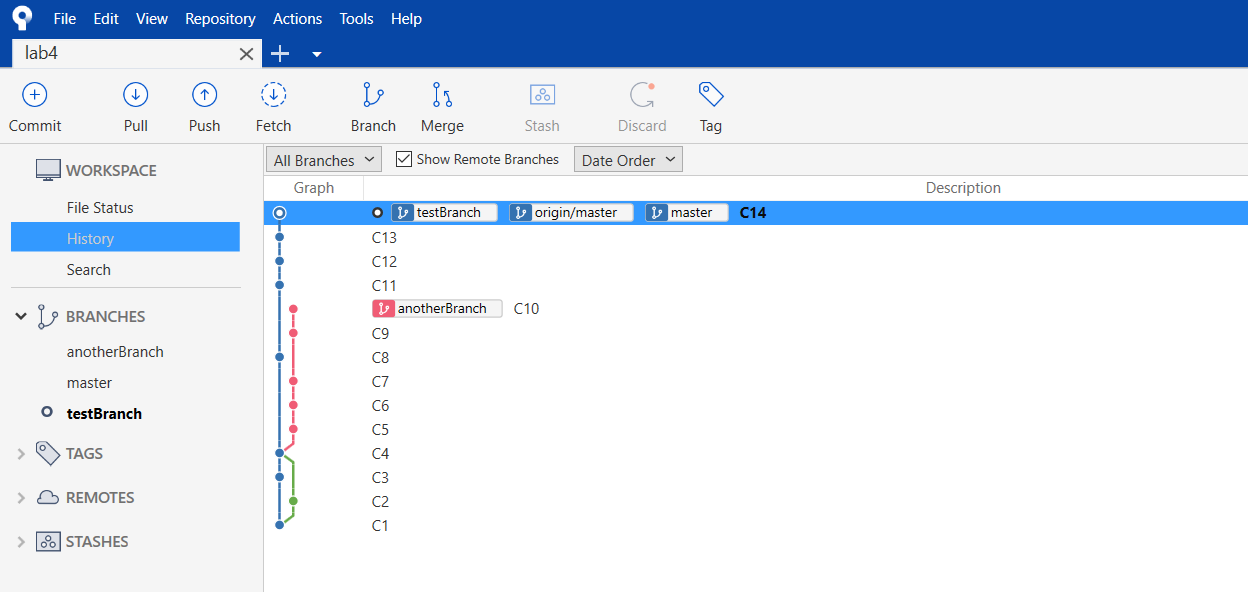


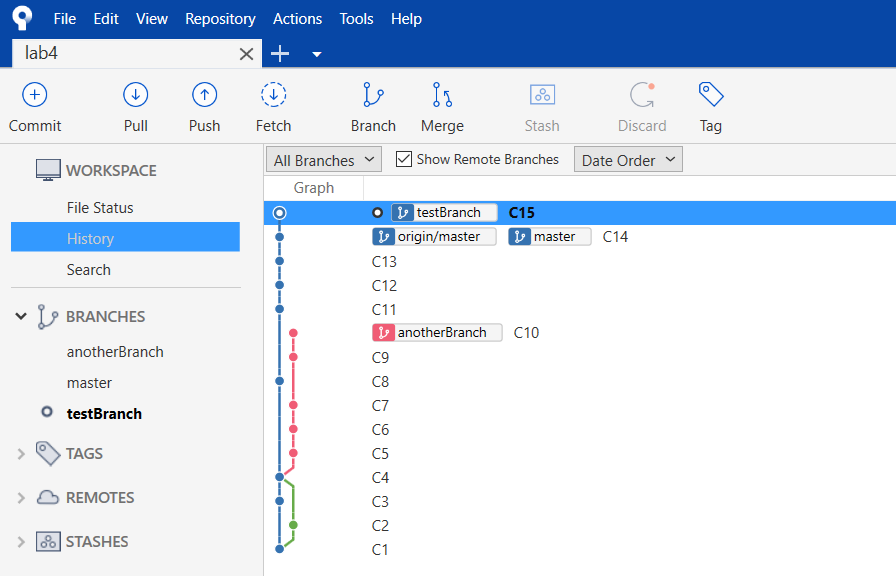
Origin/master:

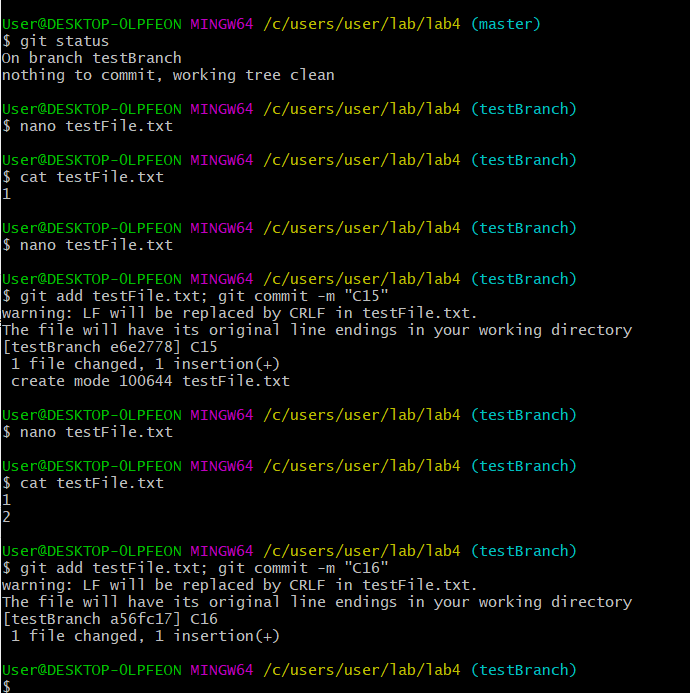


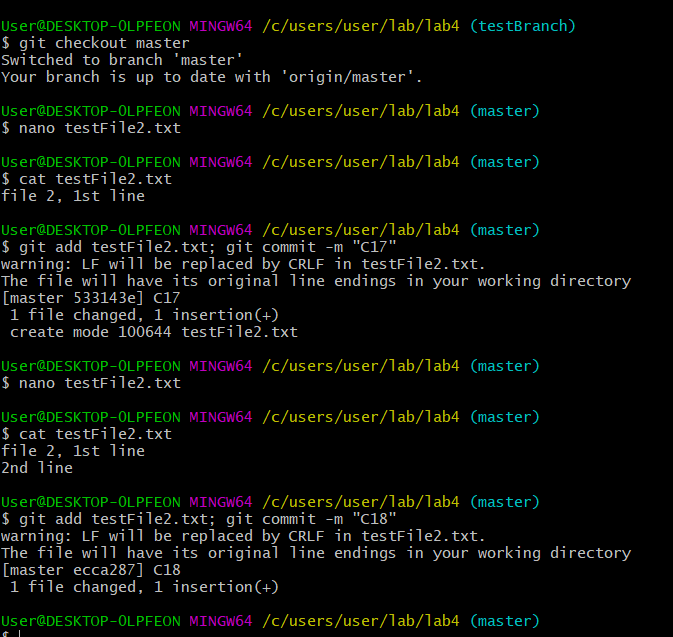
# git rebase

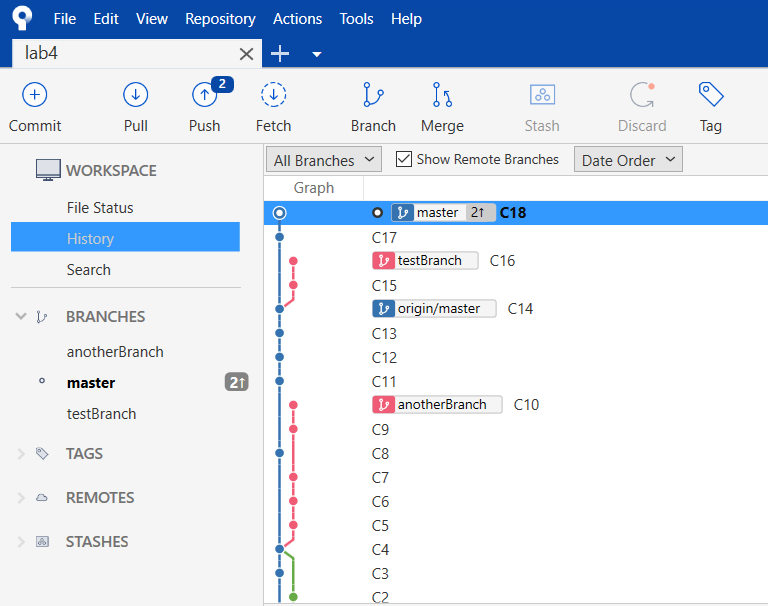


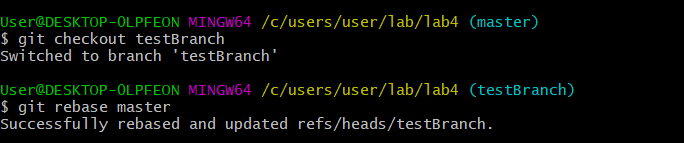


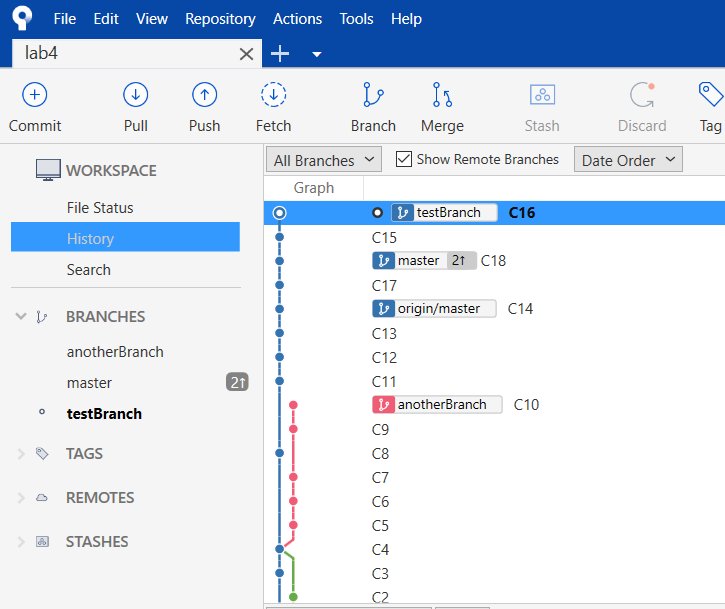


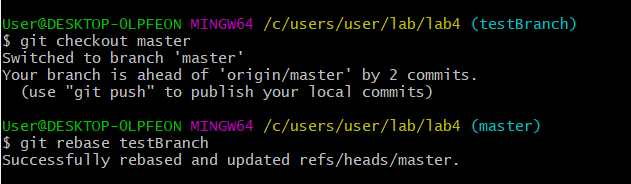


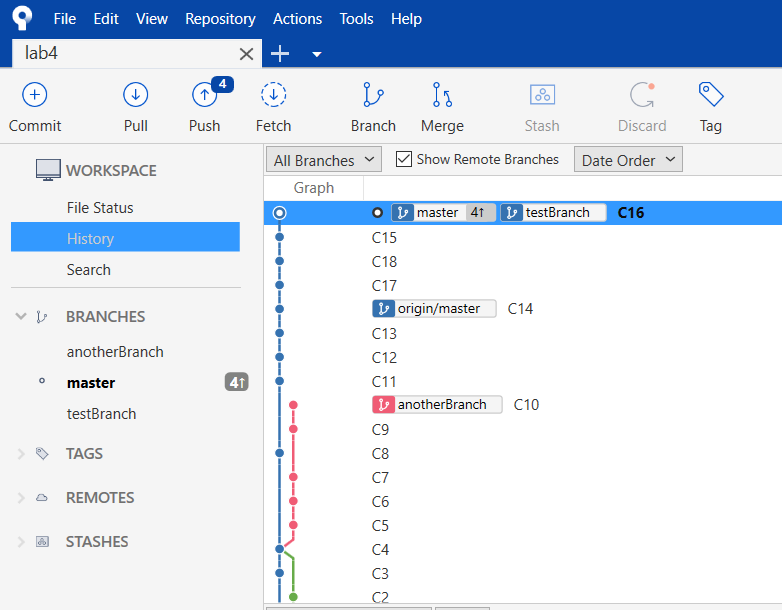






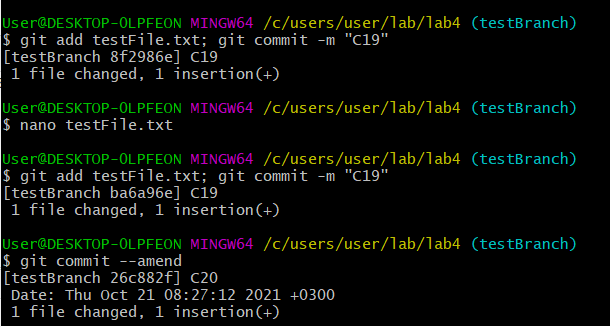




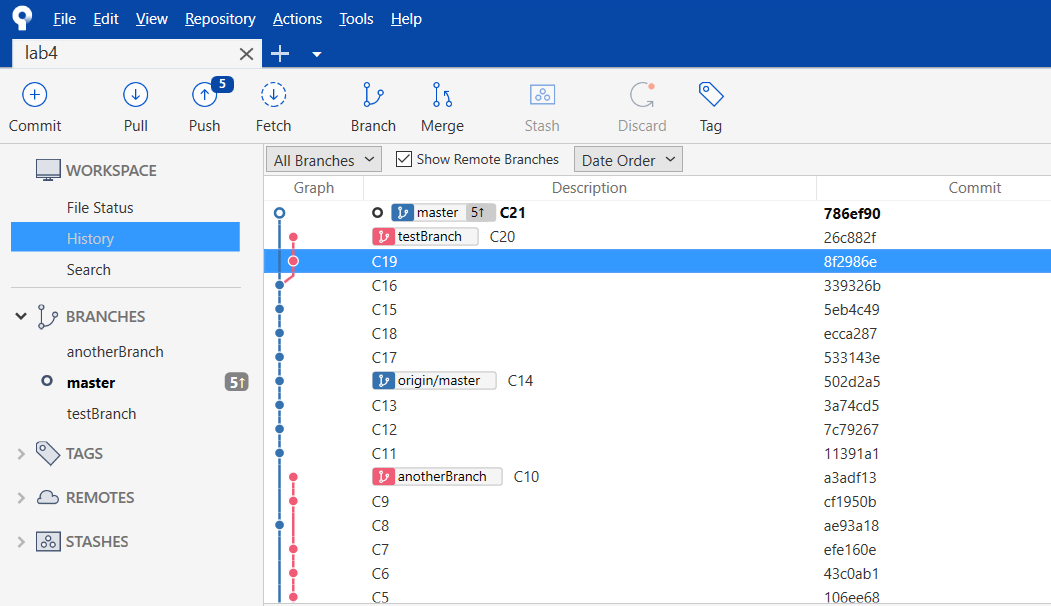


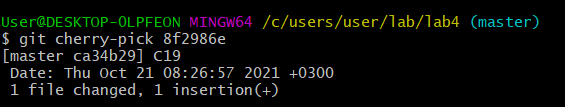
# git –amend

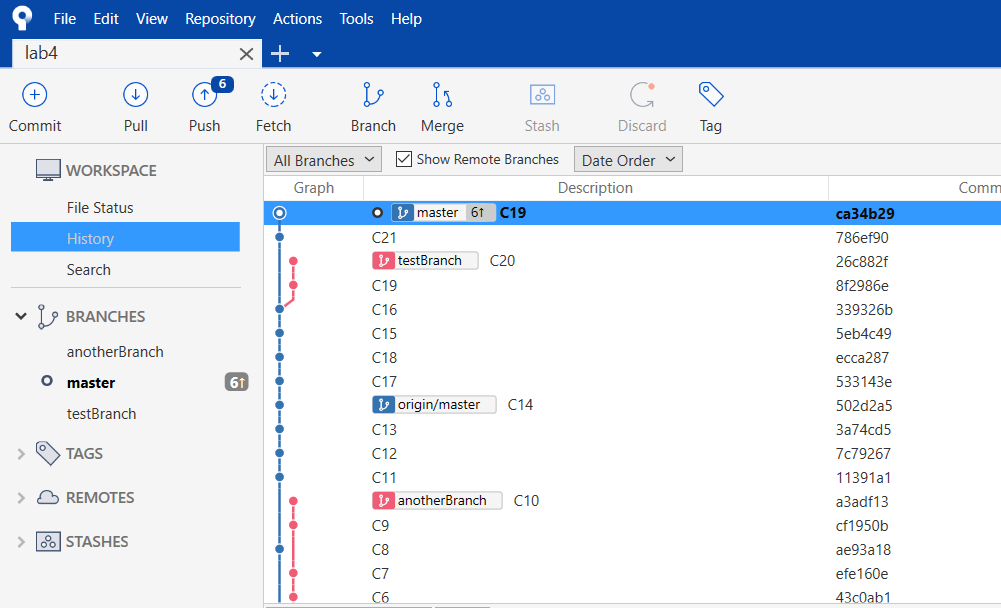
(Git –amend –no-edit)



# git cherry-pick

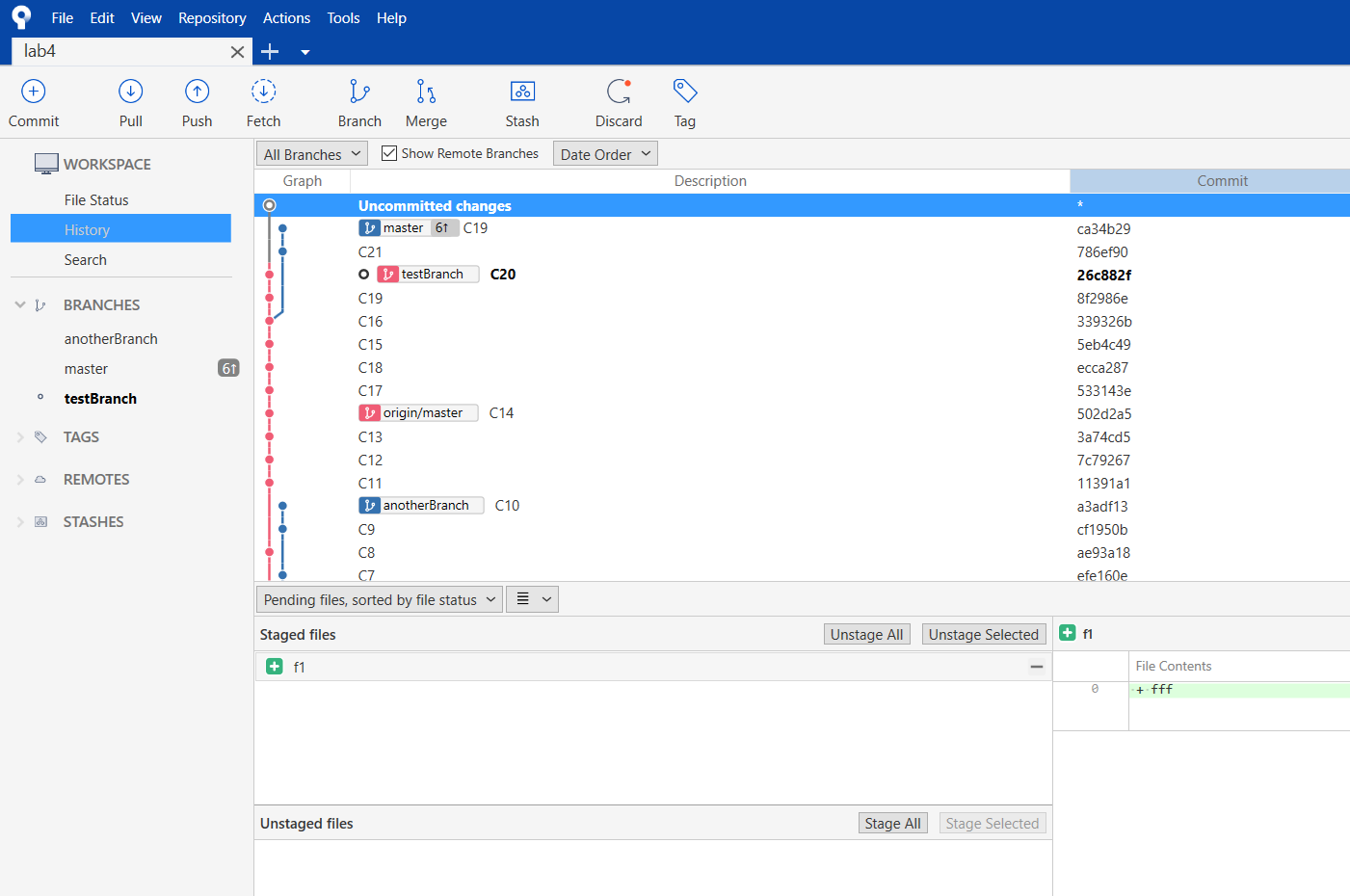


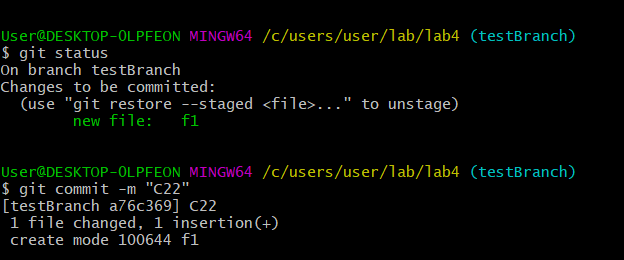


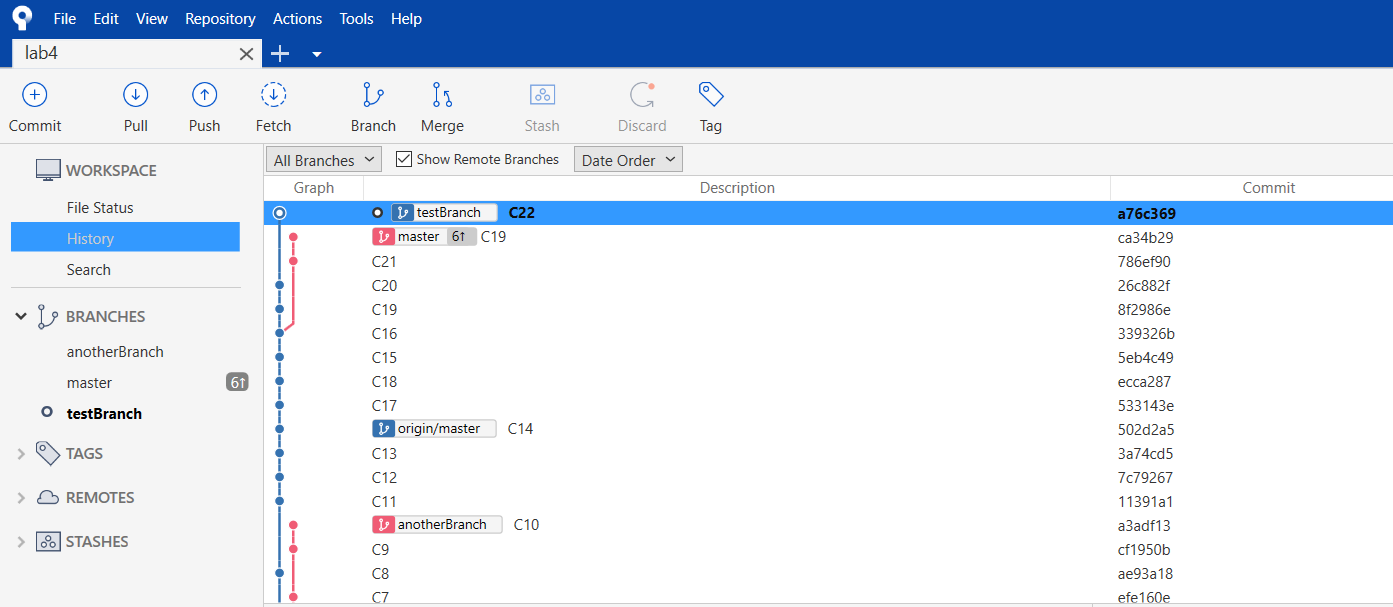


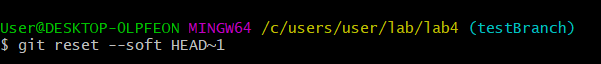
# git reset

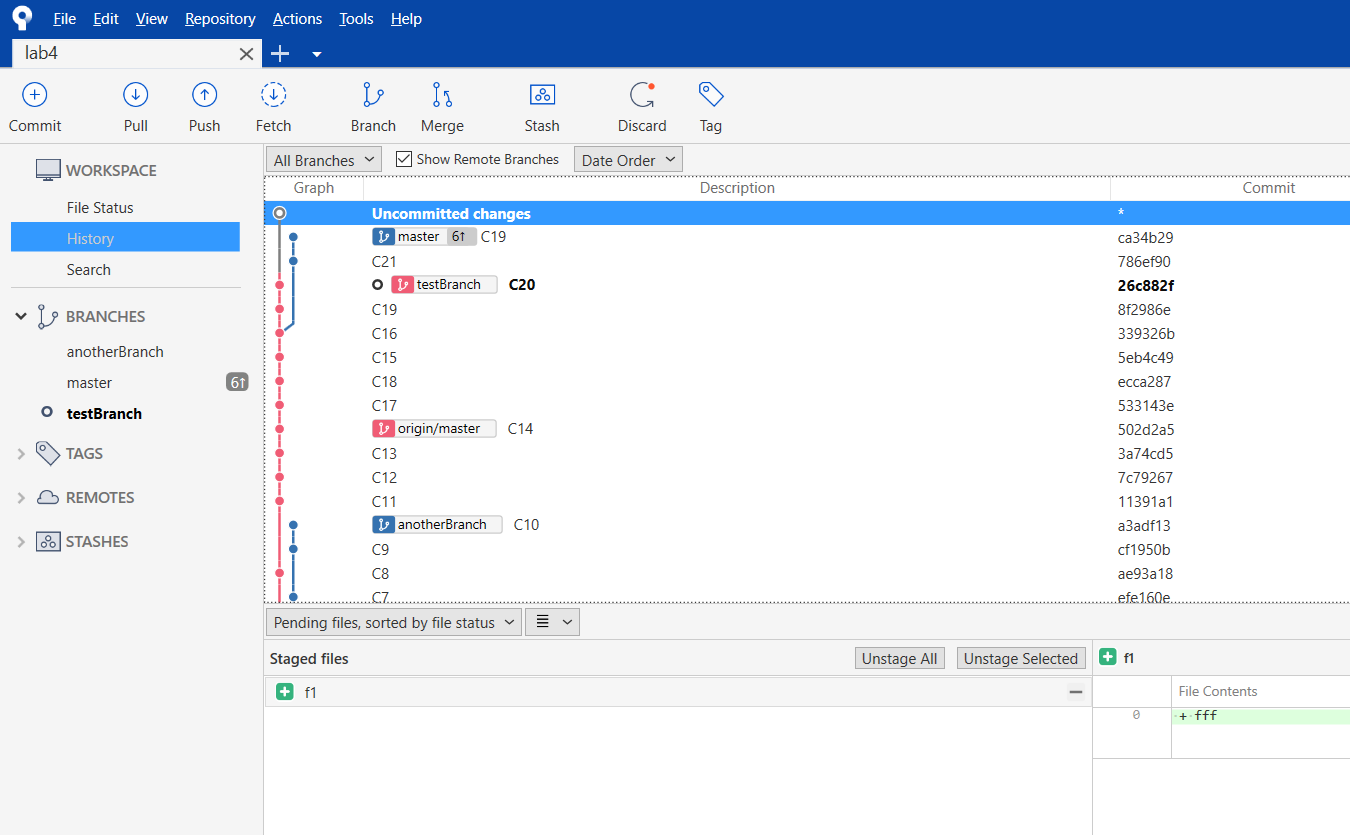
1. **git reset –soft HEAD~1** (undo the last commit, keeping the changes in all files)



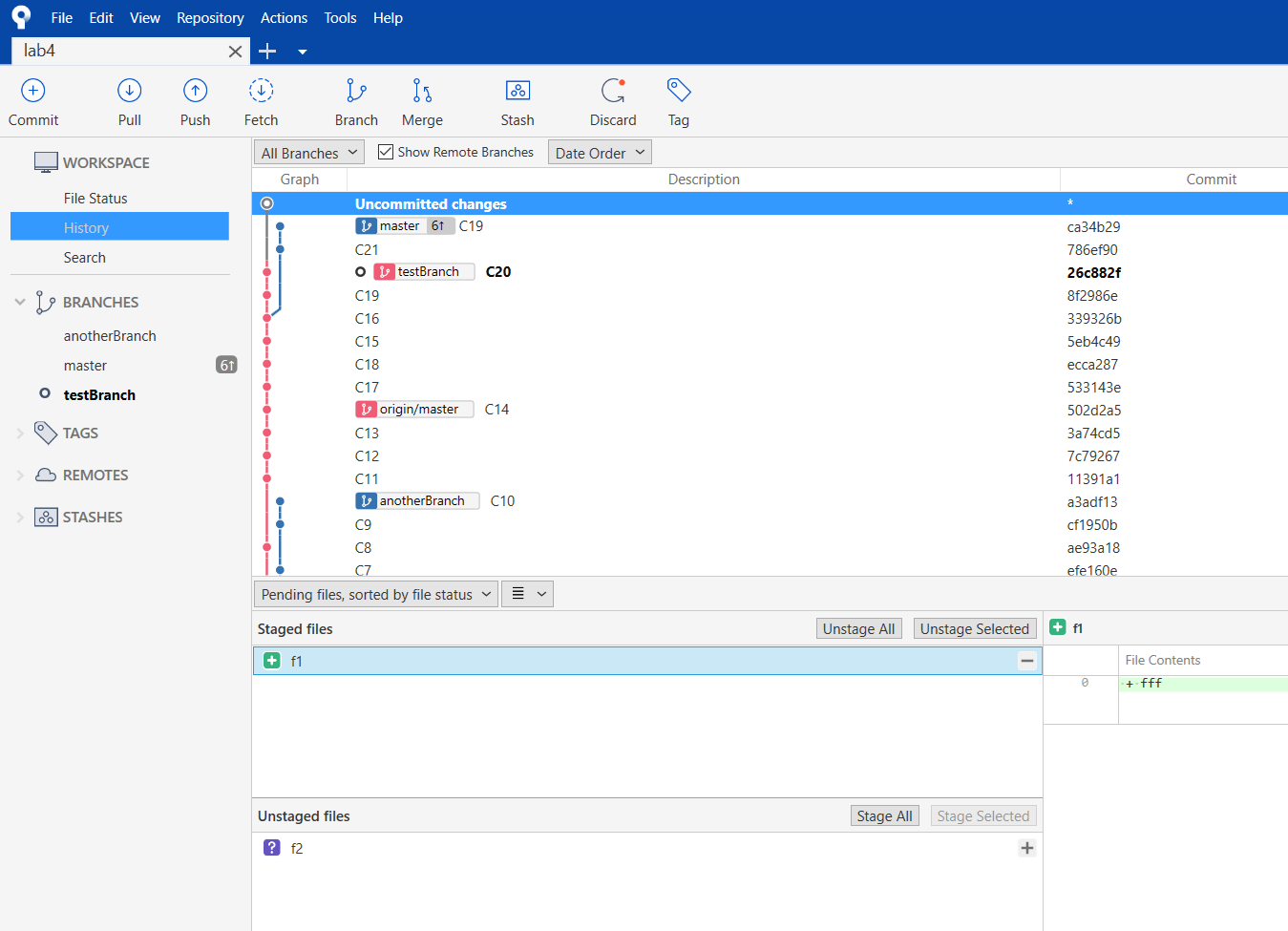
\

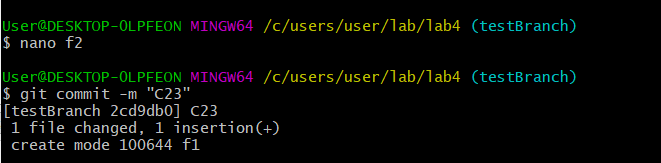


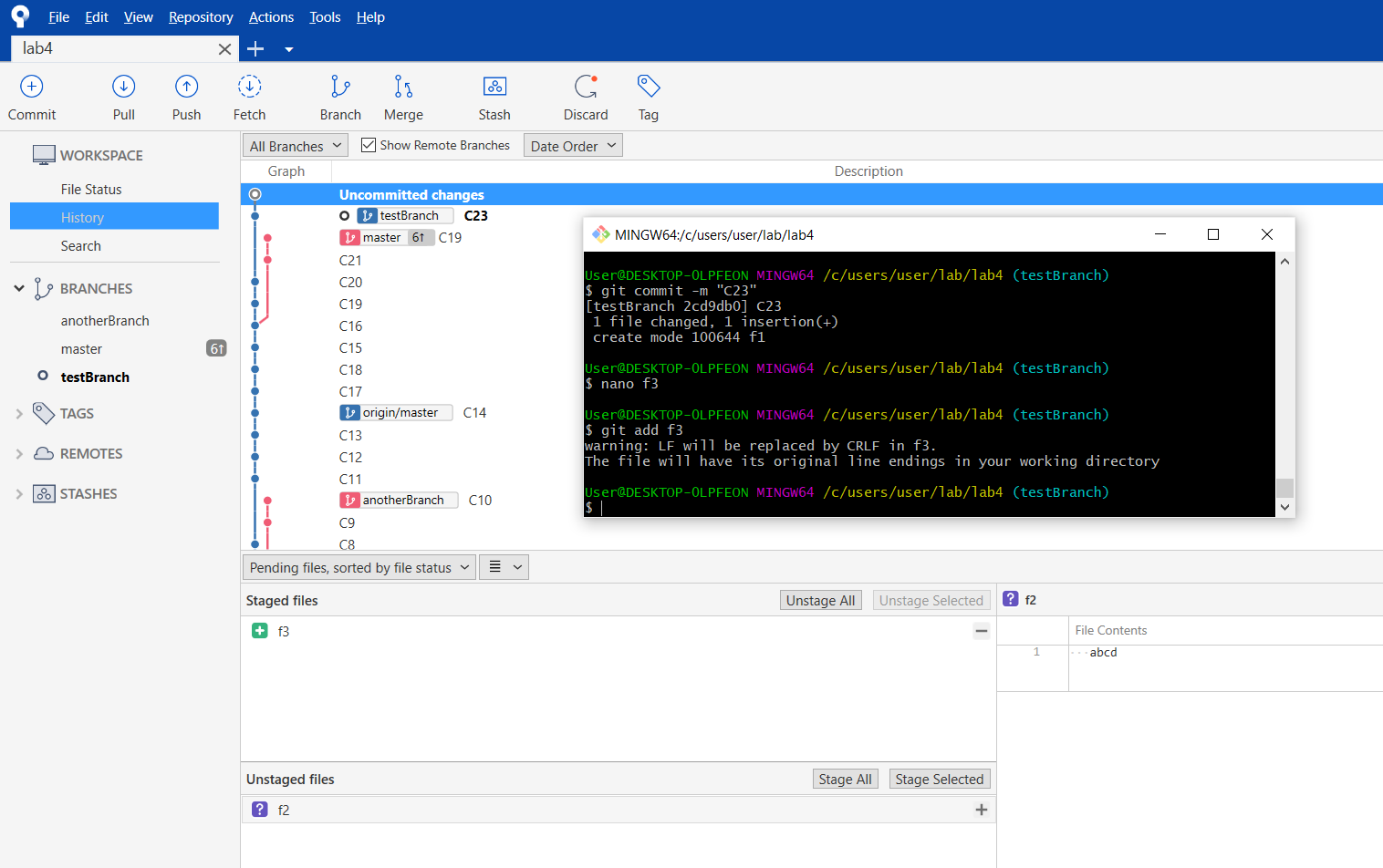


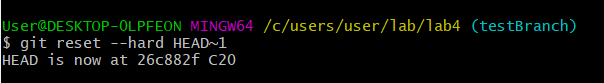


1. **git reset --hard HEAD~1**

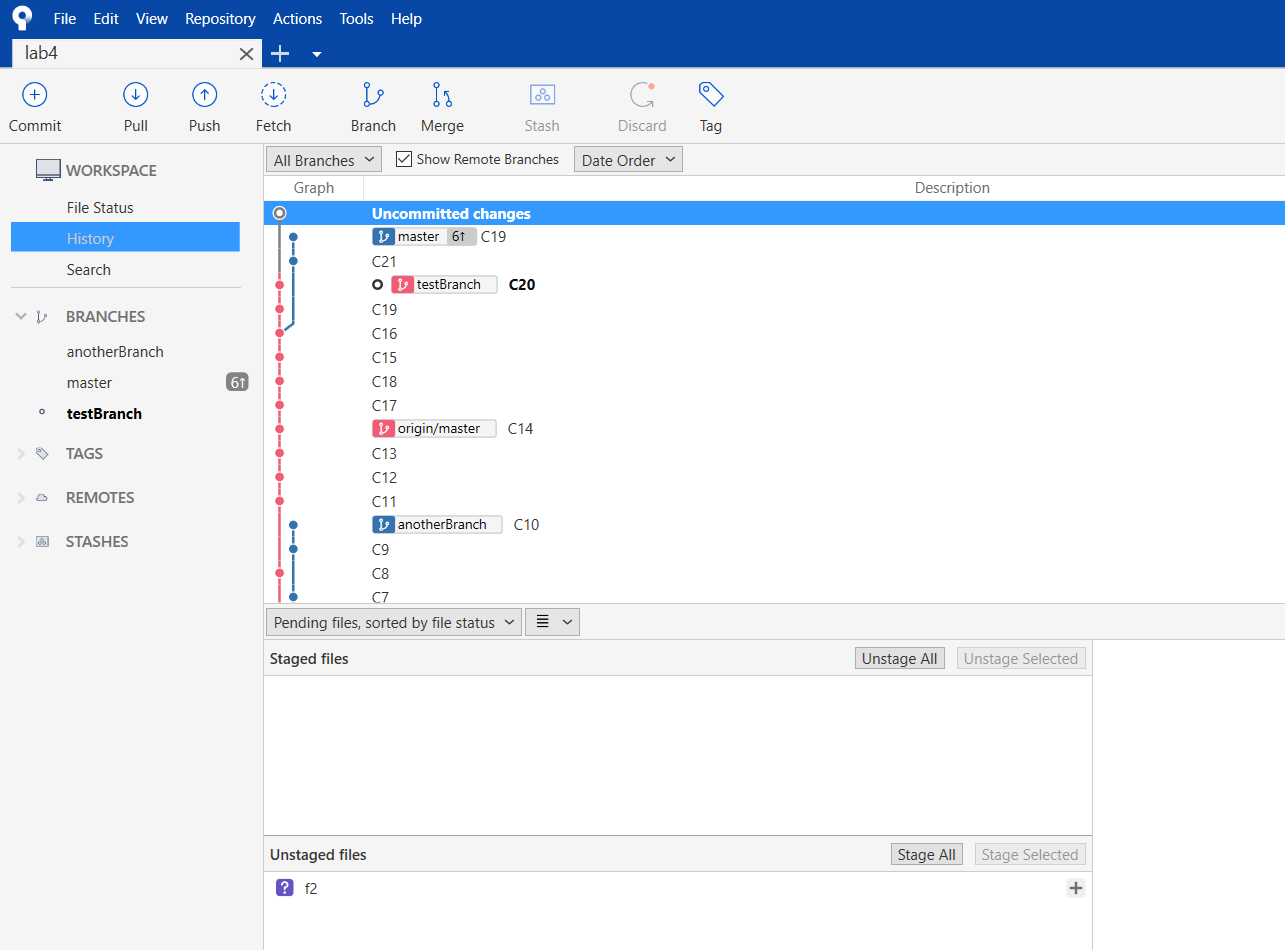




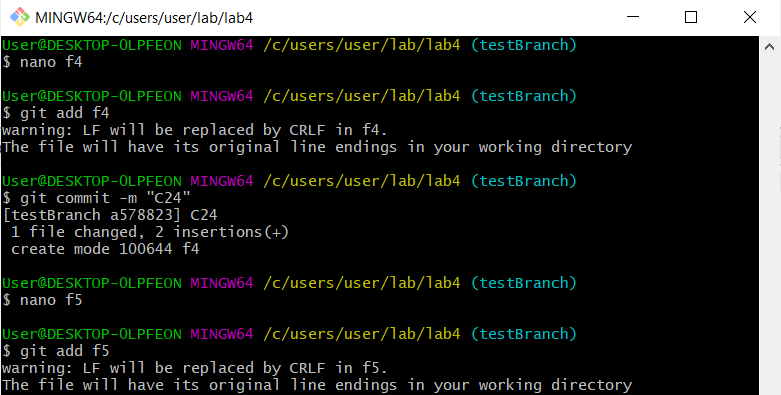


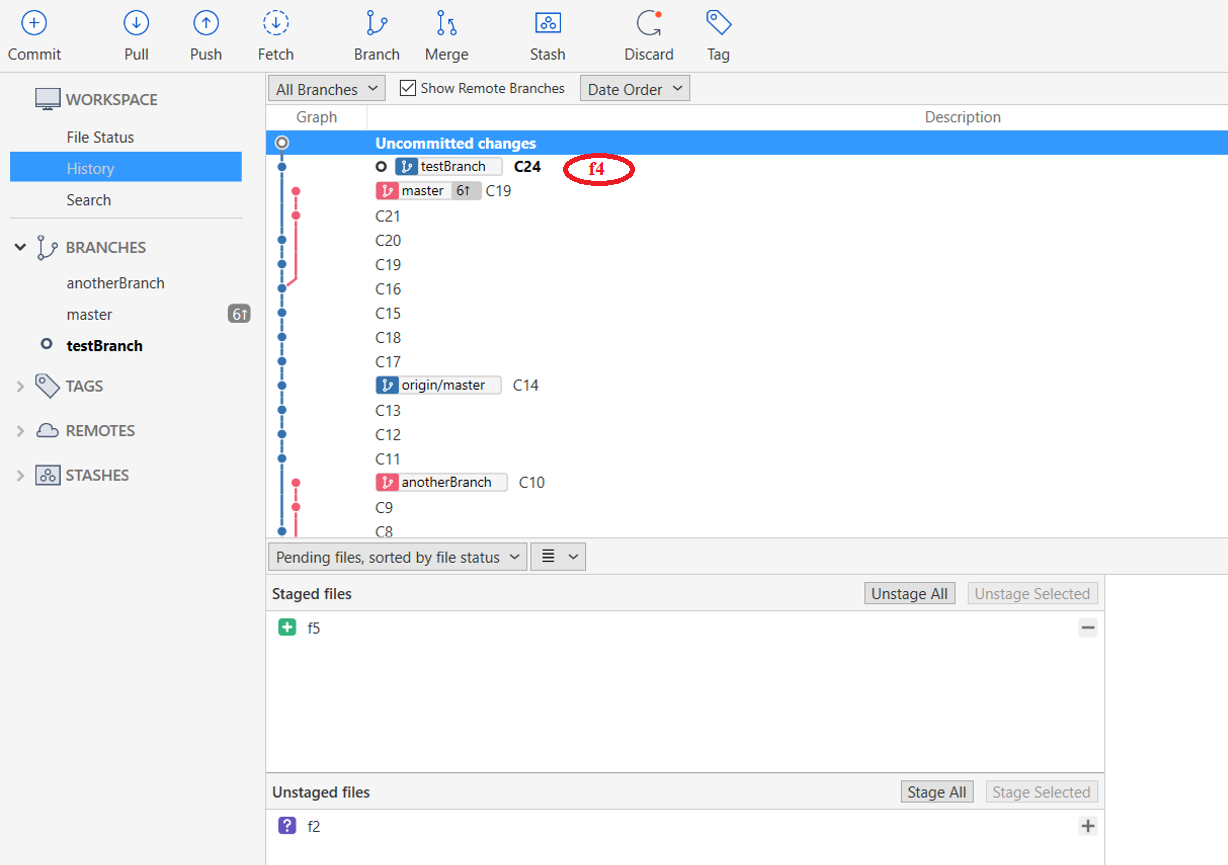


This command removed the commit C23, all together with all changes from the staged area.



1. **git reset –mixed HEAD~1**







This command discards the commit C24 and all the changes from C24 and the staging area and they remain only in the Working directory (unstaged files).

