**4. Git-HOL**

**Objectives:**1. Explain how to resolve the conflict during merge.  
A merge conflict occurs when two branches make changes to the same part of a file and Git cannot decide which version to keep. In this lab, you will simulate such a conflict using the hello.xml file, and resolve it using both manual editing and the P4Merge visual merge tool. You'll also learn how to finalize and clean up the merge process properly.

**Steps:**

1. Verified that master is in clean state using:  
 git checkout master  
 git status

2. Created a new branch:  
 git checkout -b GitWork

3. Created hello.xml with content in GitWork:  
 echo "<message>Hello from GitWork branch</message>" > hello.xml

4. Checked status:  
 git status

5. Committed hello.xml in GitWork:  
 git add hello.xml  
 git commit -m "Added hello.xml in GitWork"

6. Switched back to master:  
 git checkout master

7. Created a conflicting hello.xml in master:  
 echo "<message>Hello from master branch</message>" > hello.xml

8. Committed hello.xml in master:  
 git add hello.xml  
 git commit -m "Added conflicting hello.xml in master"

9. Viewed log with graph:  
 git log --oneline --graph --decorate --all

10. Checked branch differences:  
 git diff master GitWork

11. Used P4Merge for visual diff:  
 git difftool master GitWork

12. Attempted to merge GitWork into master:  
 git merge GitWork (got conflict)

13. Opened hello.xml to inspect Git conflict markup:  
 notepad++ hello.xml

14. Used 3-way merge tool P4Merge to resolve conflict:  
 git mergetool

15. Marked conflict resolved and committed:  
 git add hello.xml  
 git commit -m "Resolved conflict using P4Merge"

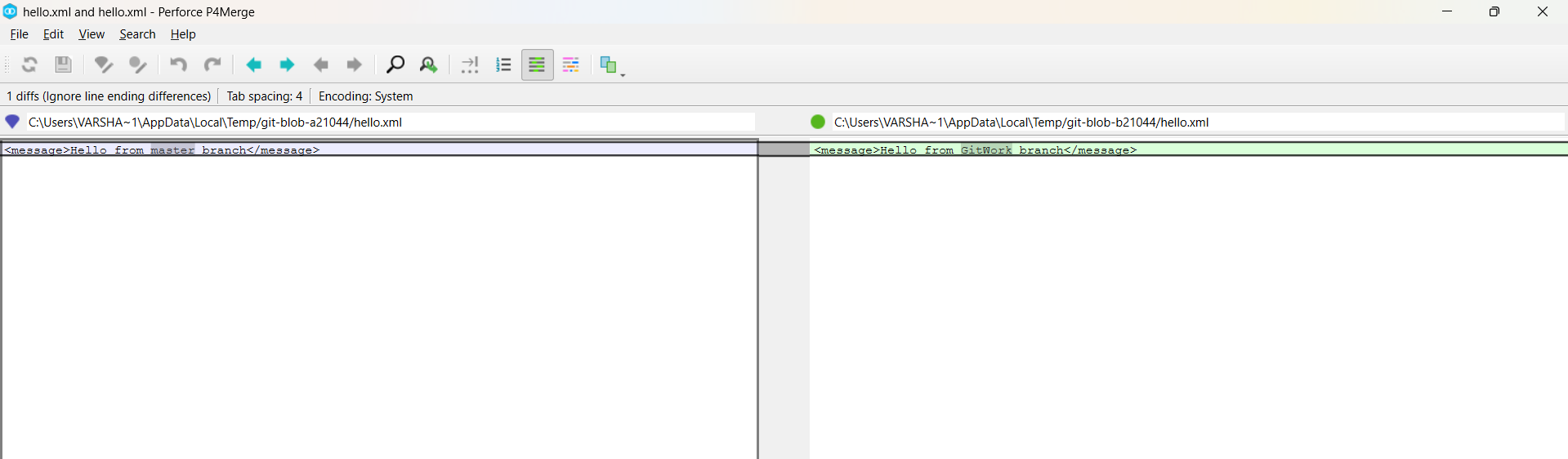
16. Checked status to confirm:  
 git status

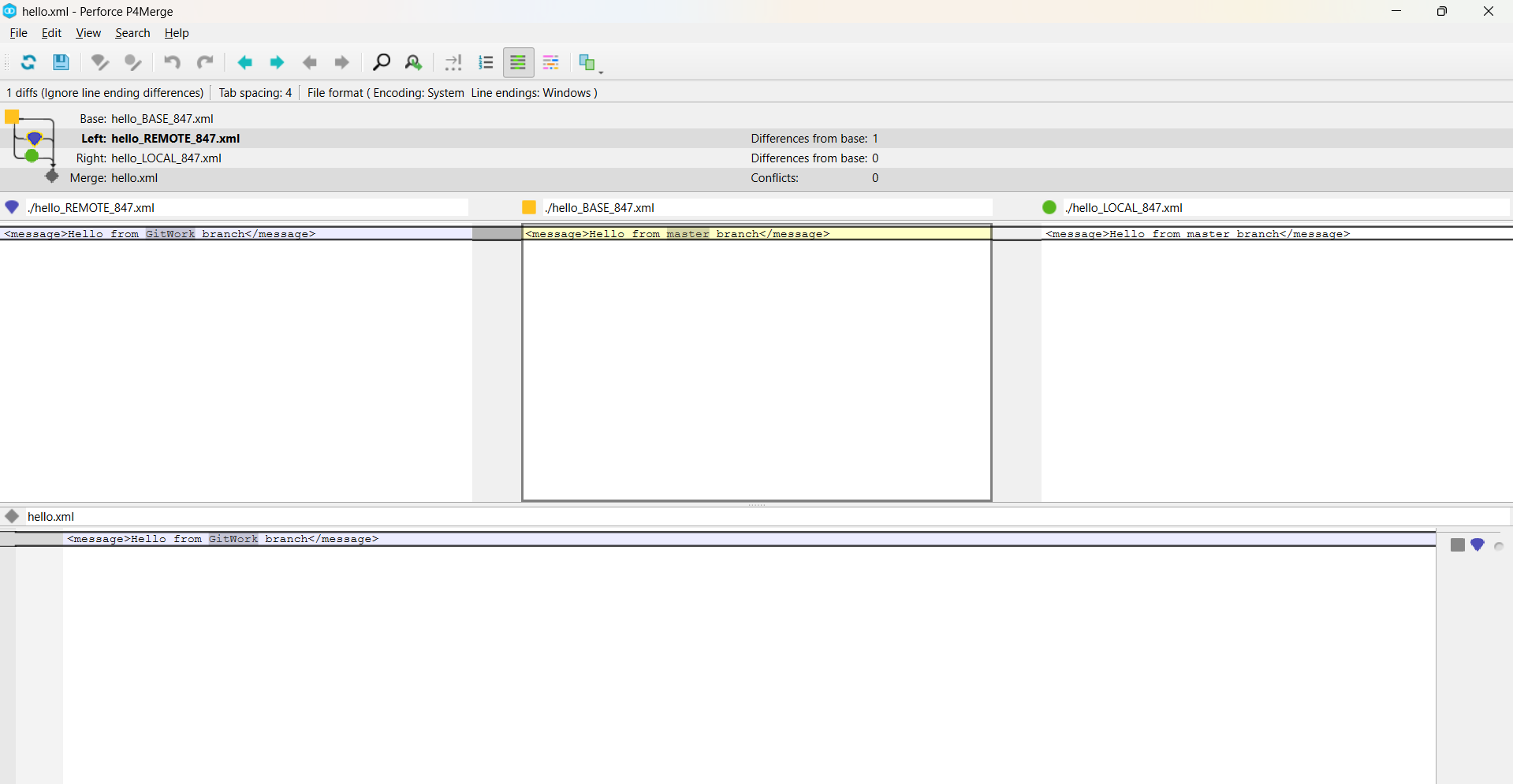
17. Added backup merge files to .gitignore:  
 echo "\*.orig" >> .gitignore  
 git add .gitignore  
 git commit -m "Added .orig to .gitignore"

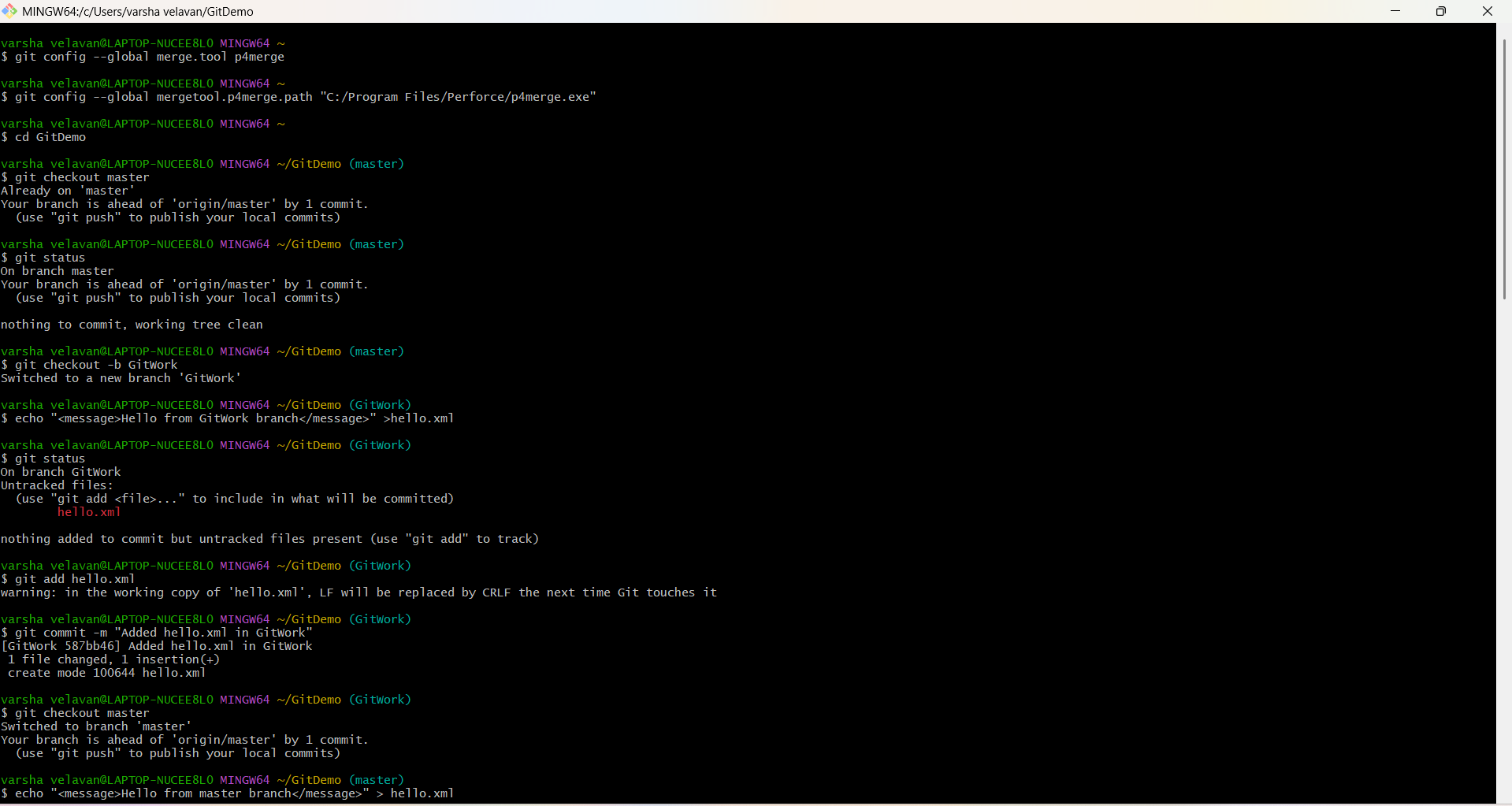
18. Listed all branches:  
 git branch -a

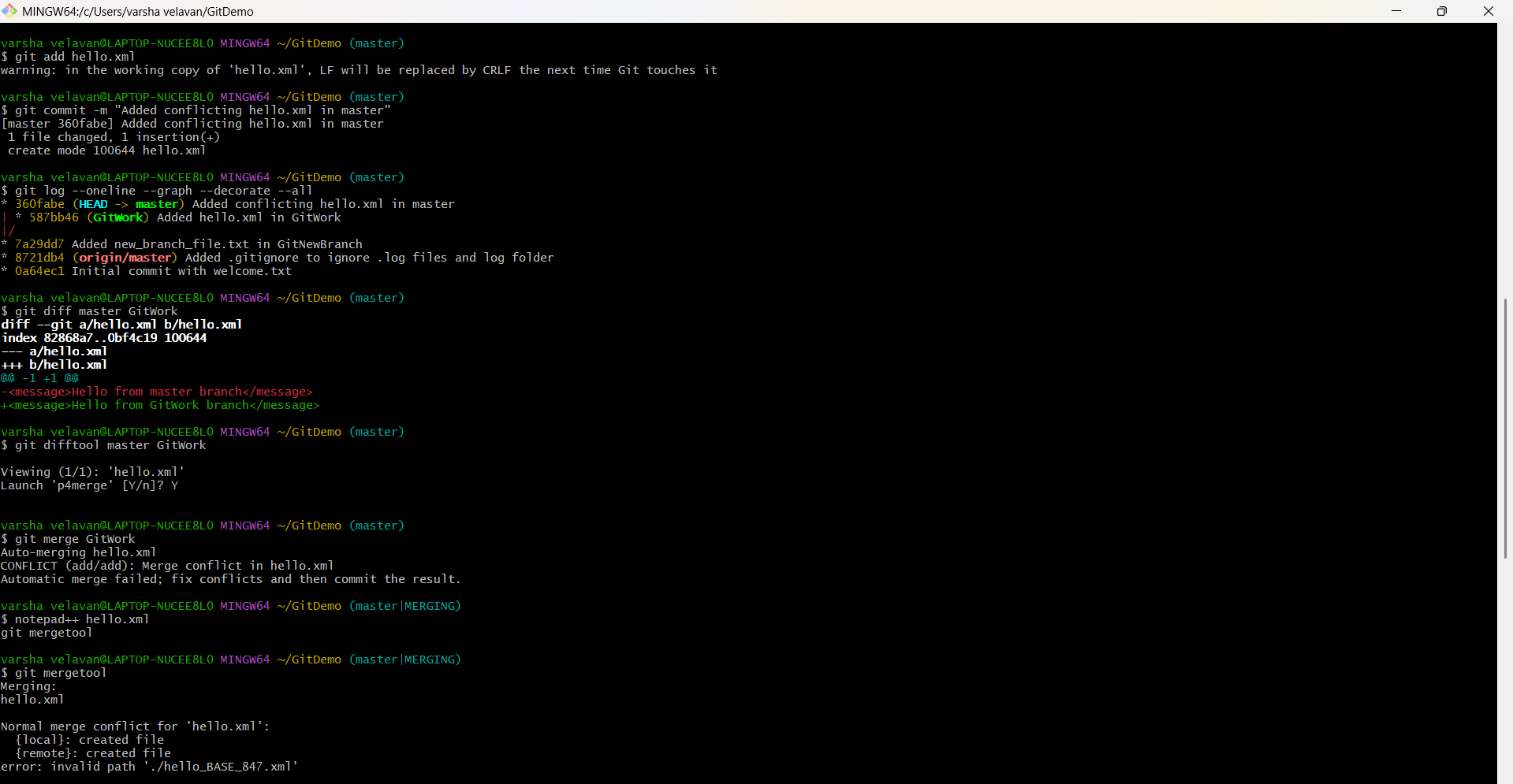
19. Deleted the GitWork branch and viewed final log:  
 git branch -d GitWork  
 git log --oneline --graph –decorate

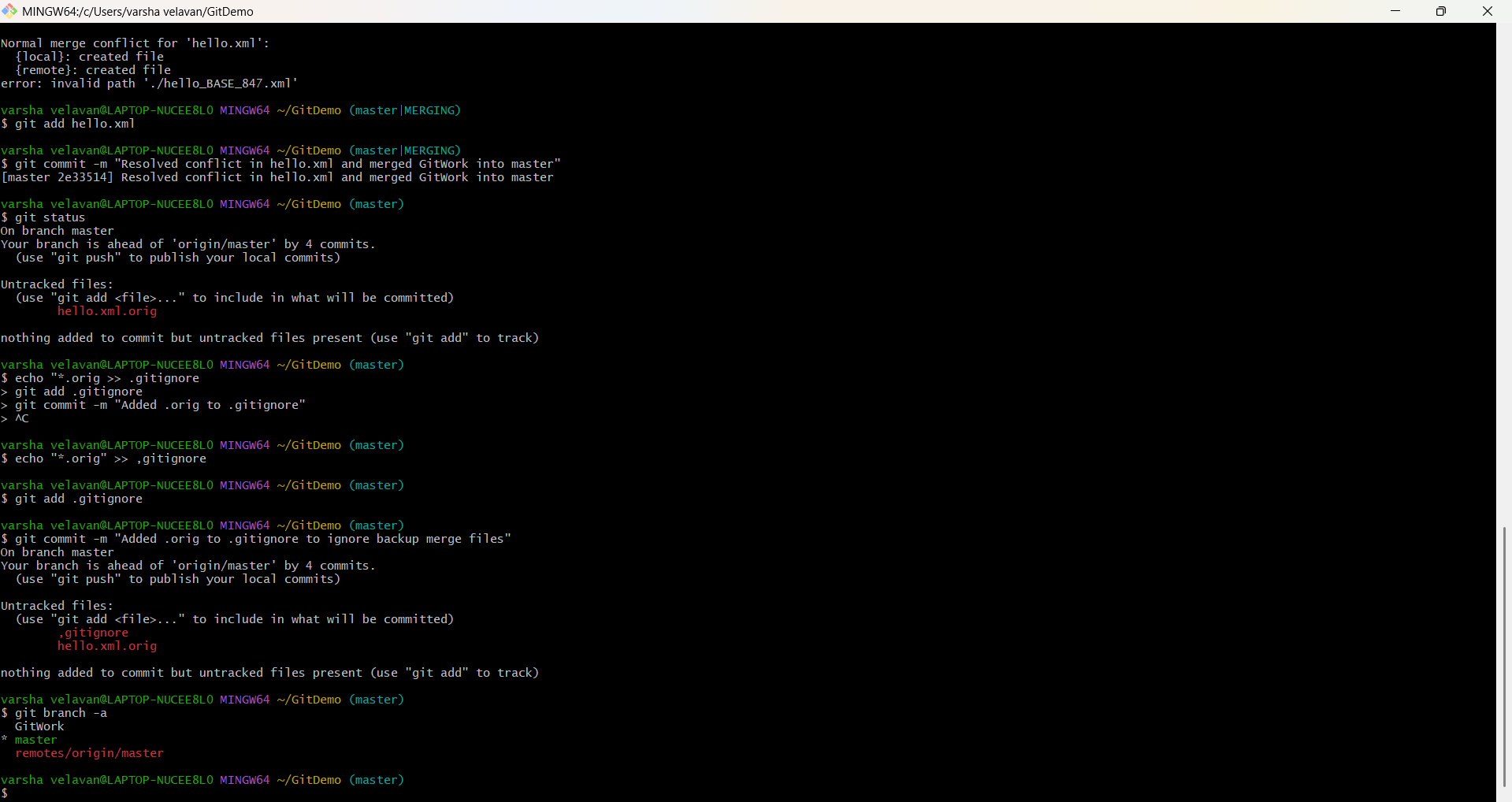
**Output:**

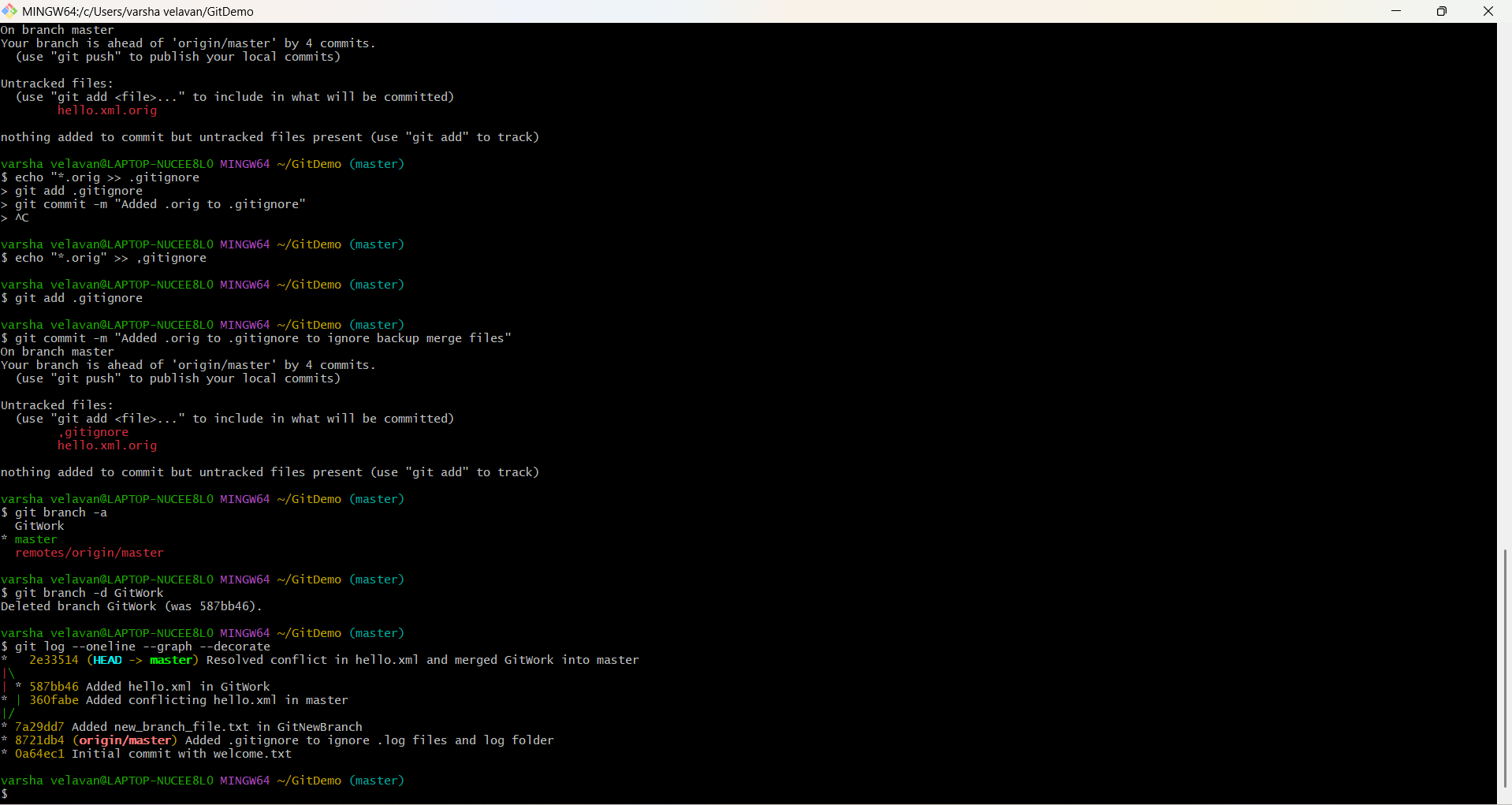
****

****

****

****

****

****