**Git Hands-On Lab: Conflict Resolution (Git-T03-HOL\_001)**

1. Please follow the instructions to complete the hands-on. Each instruction expect a command for the Git Bash.

1. Verify if master is in clean state.
2. Create a branch **“GitWork”.** Add a file “hello.xml”.
3. Update the content of “hello.xml” and observe the status
4. Commit the changes to reflect in the branch
5. Switch to master.
6. Add a file **“hello.xml”** to the master and add some different content than previous.
7. Commit the changes to the master
8. Observe the log by executing **“git log –oneline –graph –decorate –all”**
9. Check the differences with Git diff tool
10. For better visualization, use P4Merge tool to list out all the differences between master and branch
11. Merge the bran to the master
12. Observe the git mark up.
13. Use 3-way merge tool to resolve the conflict
14. Commit the changes to the master, once done with conflict
15. Observe the git status and add backup file to the .gitignore file.
16. Commit the changes to the .gitignore
17. List out all the available branches
18. Delete the branch, which merge to master.
19. Observe the log by executing **“git log –oneline –graph –decorate”**

**Solution:**

**Prerequisites:**

• Hands-on ID: Git-T03-HOL\_001

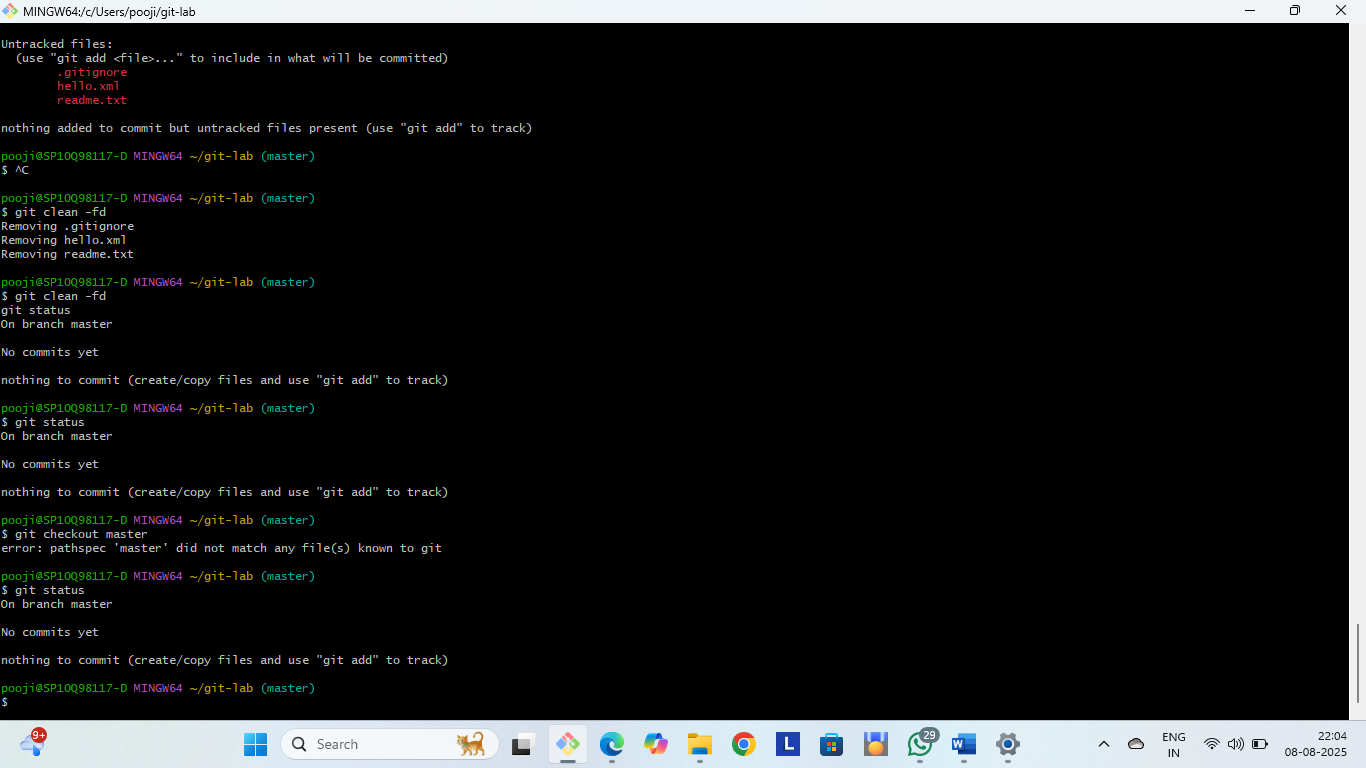
• Git installed

• GitHub account

**Steps Performed:**

**1. Verify if master is in clean state:** Ensures you are on the master branch and have no uncommitted changes**.**

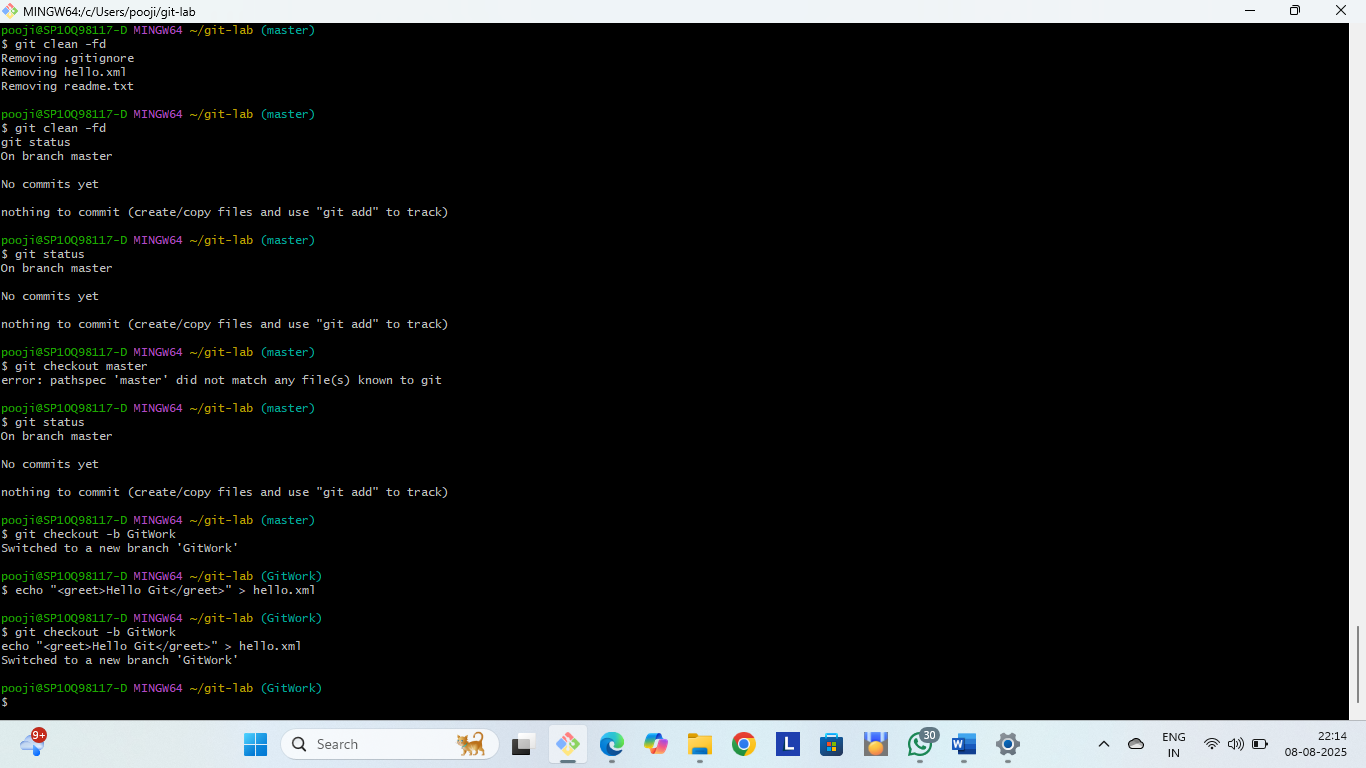
**Command:**

* git checkout master
* git status

**2. Create a branch “GitWork” and add a file *hello.xml*:** Creates and switches to GitWork branch, then creates the file.

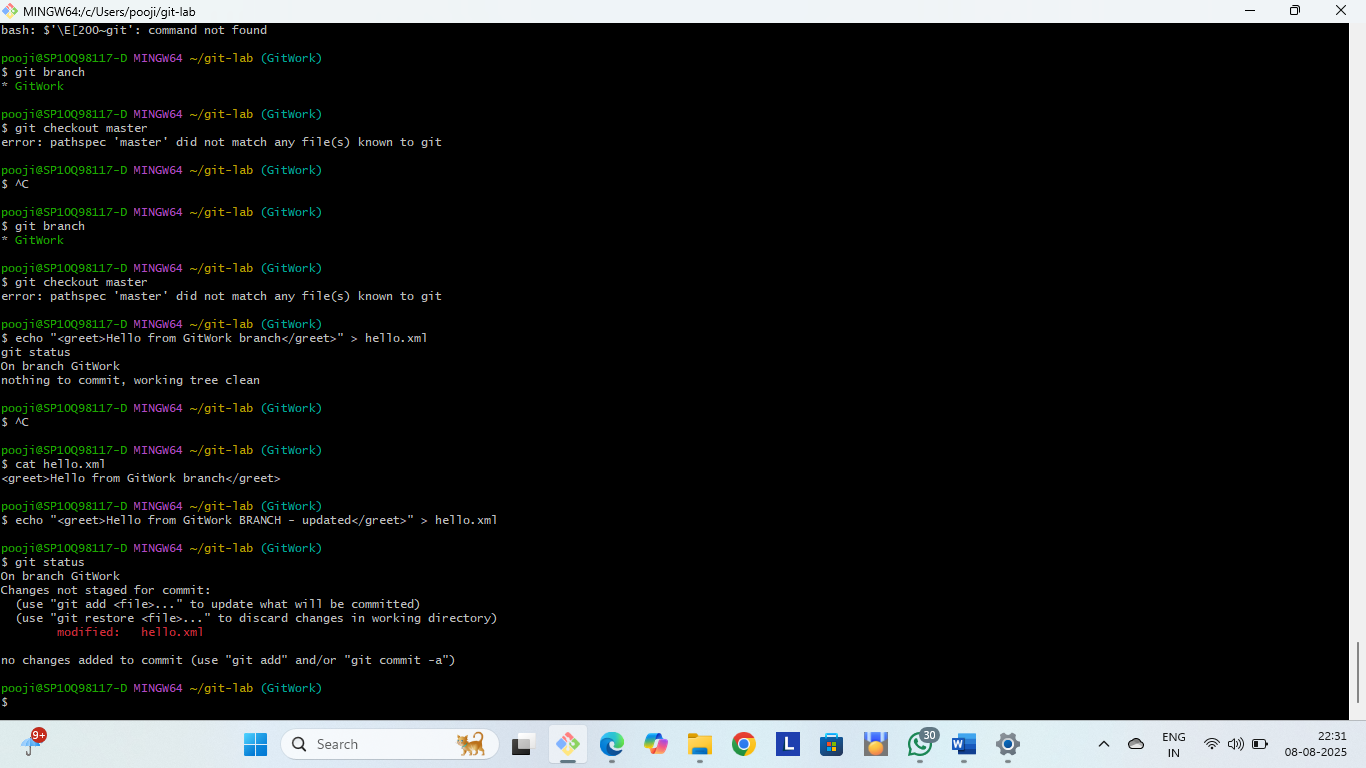
**Command:**

* git checkout -b GitWork

 echo "<greet>Hello Git</greet>" > hello.xml

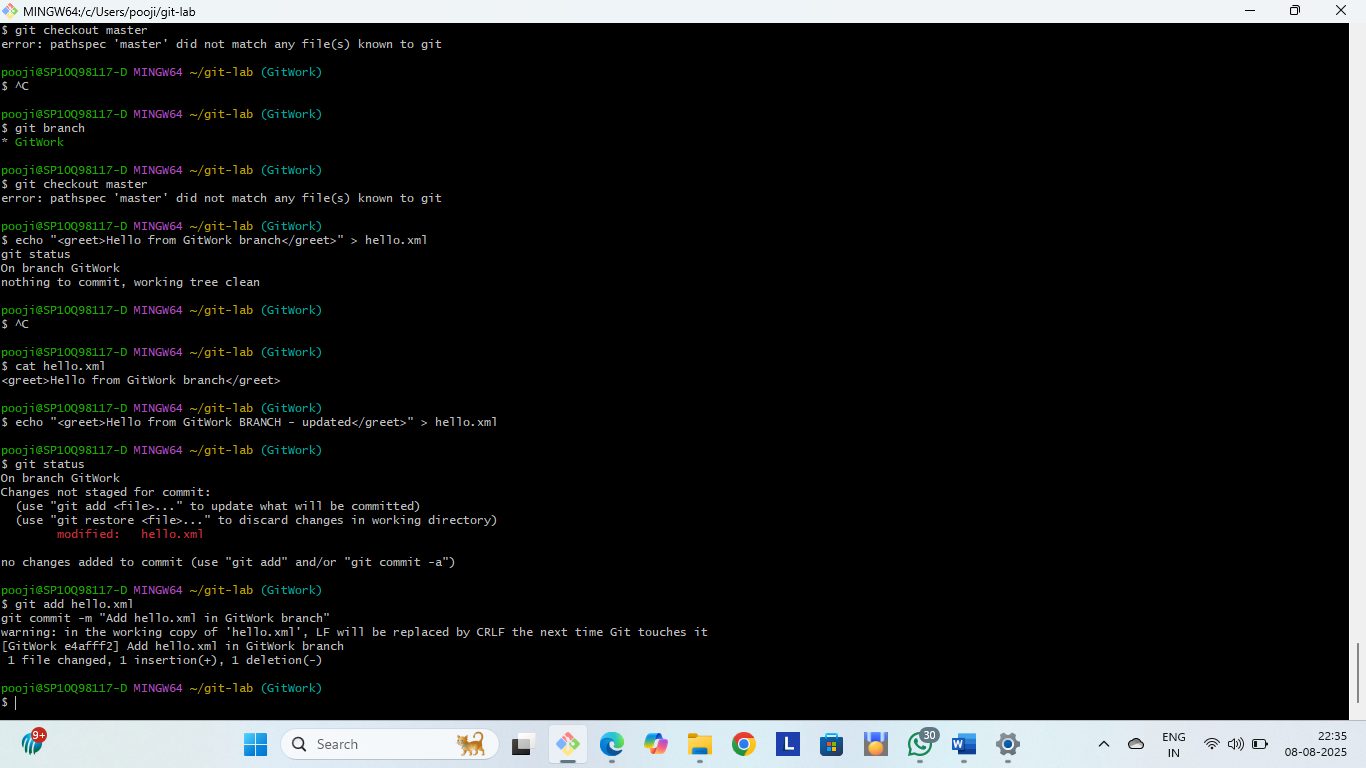
**3. Update the content of hello.xml and observe the status :** Shows that the file has been modified but not yet committed.

**Command:**

* echo "<greet>Hello from GitWork branch</greet>" > hello.xml
* git status

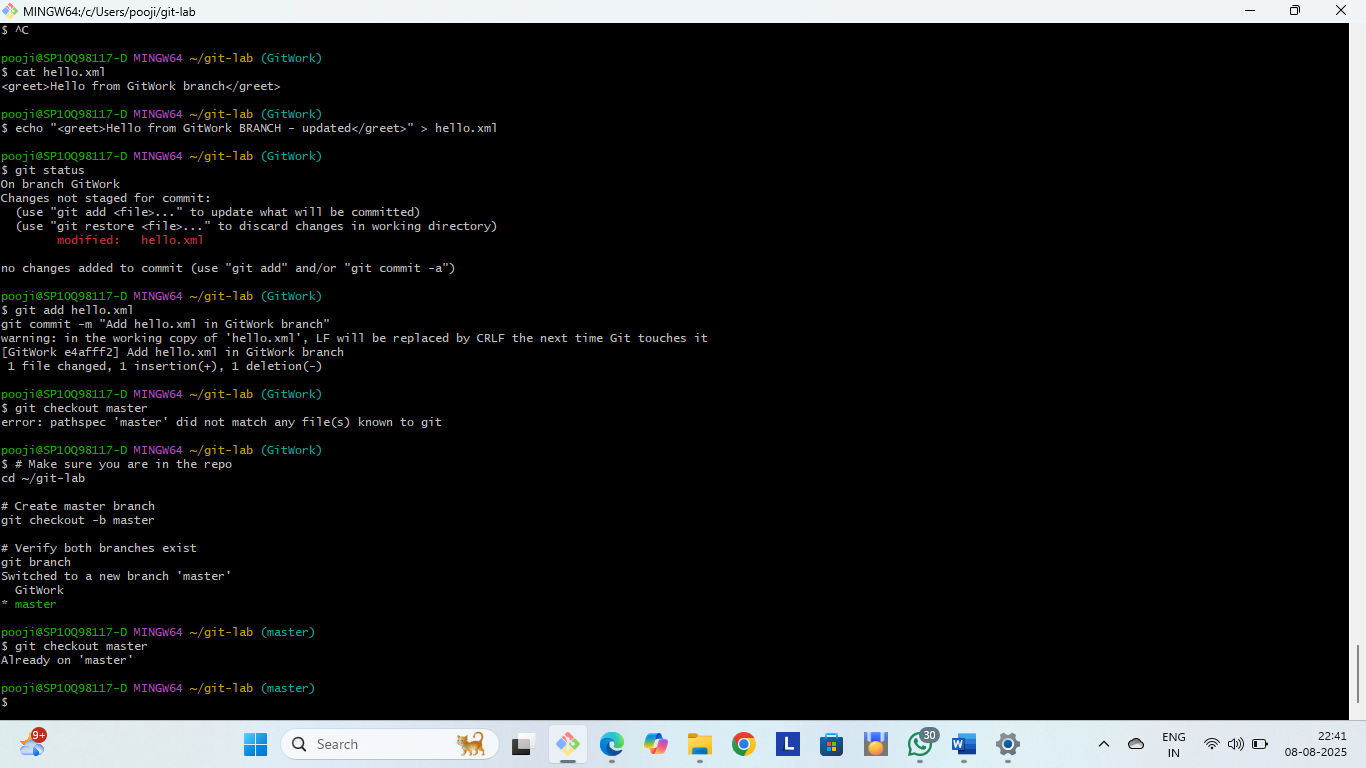
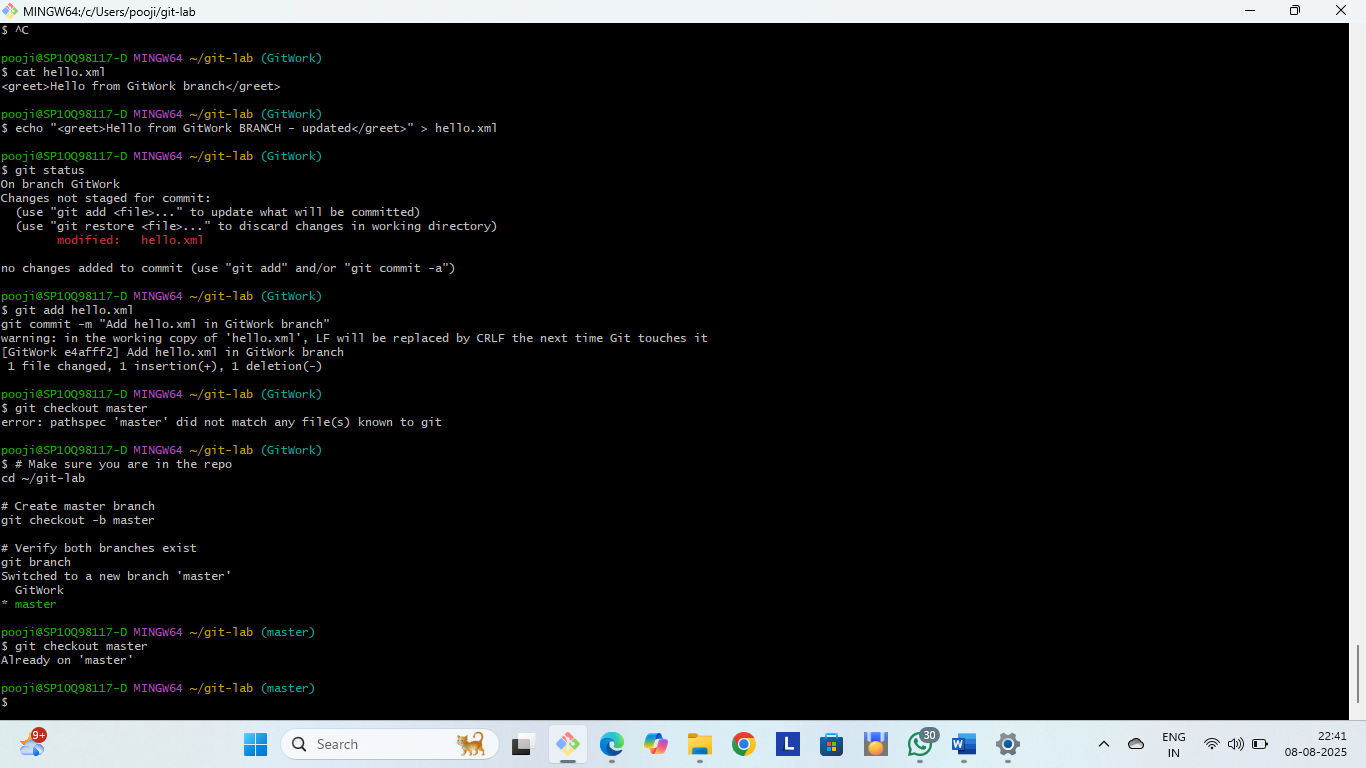
**4. Commit changes to GitWork:** Saves changes to GitWork branch.

**Command:**

* git add hello.xml
* git commit -m "Add hello.xml in GitWork branch"

**5. Switch to master:** Back to master to make separate edits.

**Command:**

* git checkout master

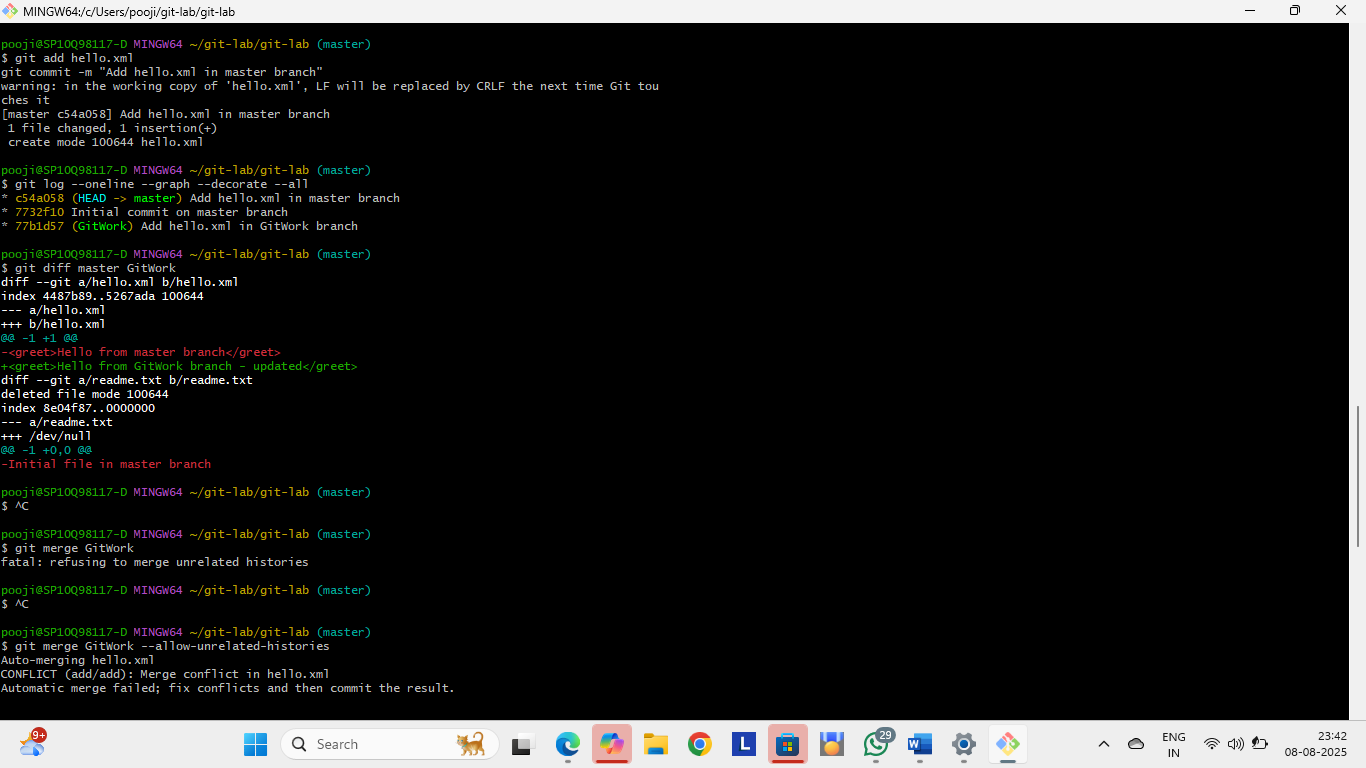
**6. Add hello.xml to master with different content:** Creates master’s own version of the file.

**Command:**

* echo "<greet>Hello from master branch</greet>" > hello.xml

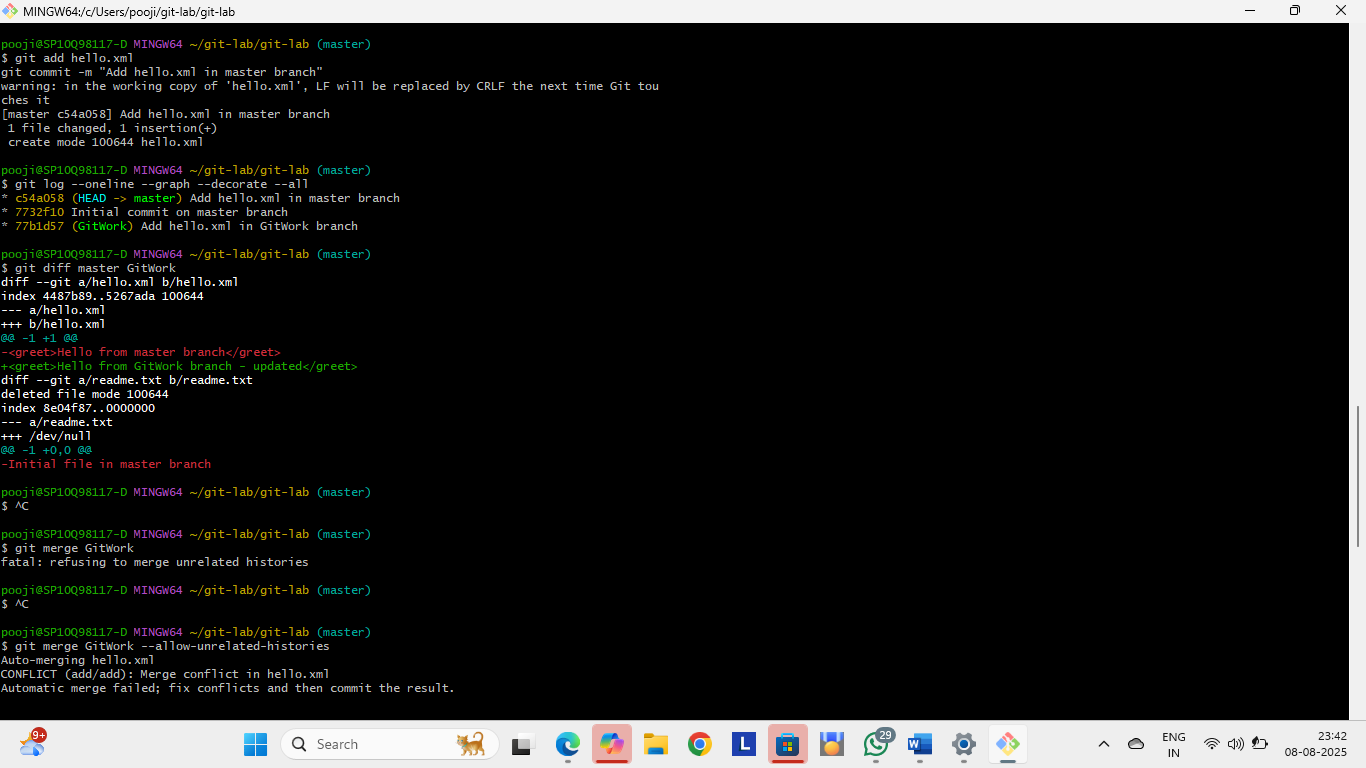
**7. Commit changes to master:** Saves master’s version.

**Command:**

* git add hello.xml
* git commit -m "Add hello.xml in master branch"

**8. View branch history:** Shows commits for both branches.

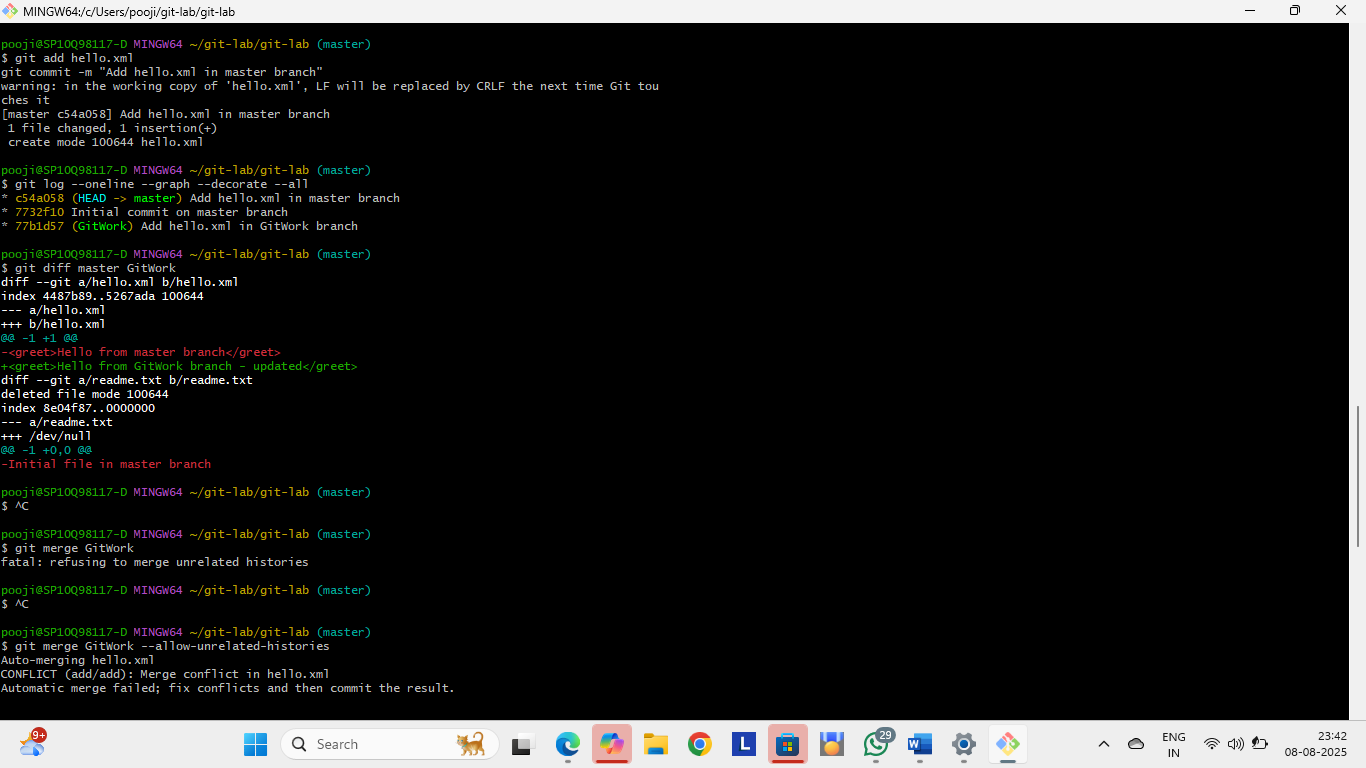
**Command:**

* git log --oneline --graph --decorate –all

**9. Compare branches:** Shows content differences in hello.xml.

**Command:**

* git diff master GitWork



**10. Use P4Merge for visual diff *(if configured):*** Opens a side-by-side diff in P4Merge.

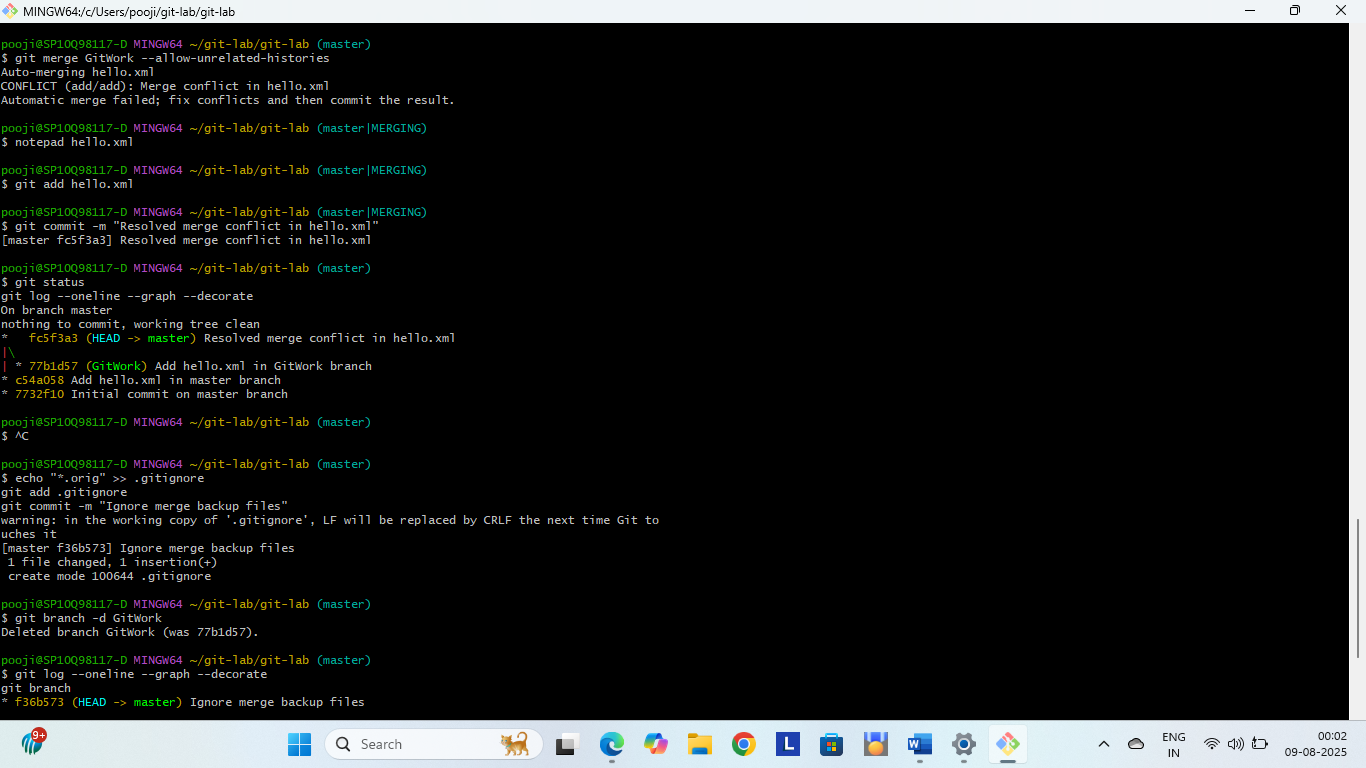
**Command:**

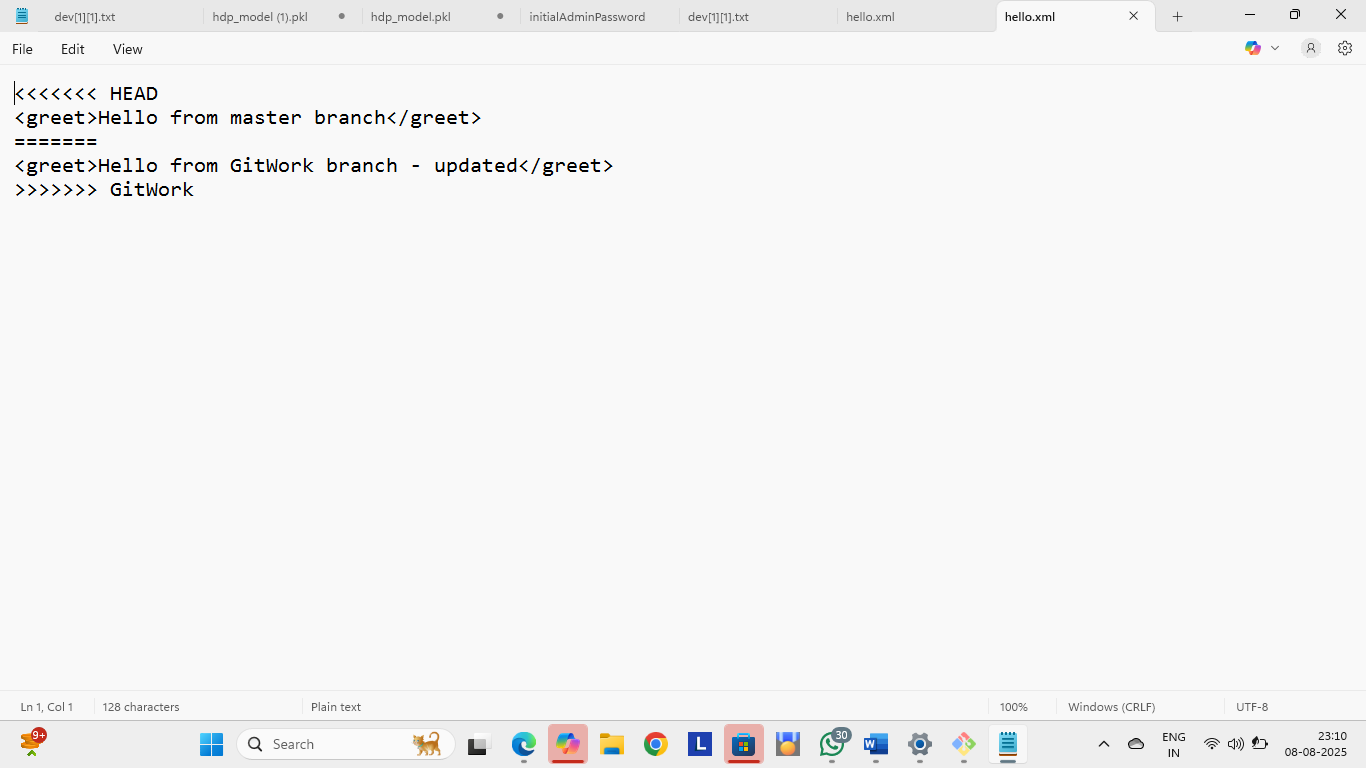
* git difftool master GitWork

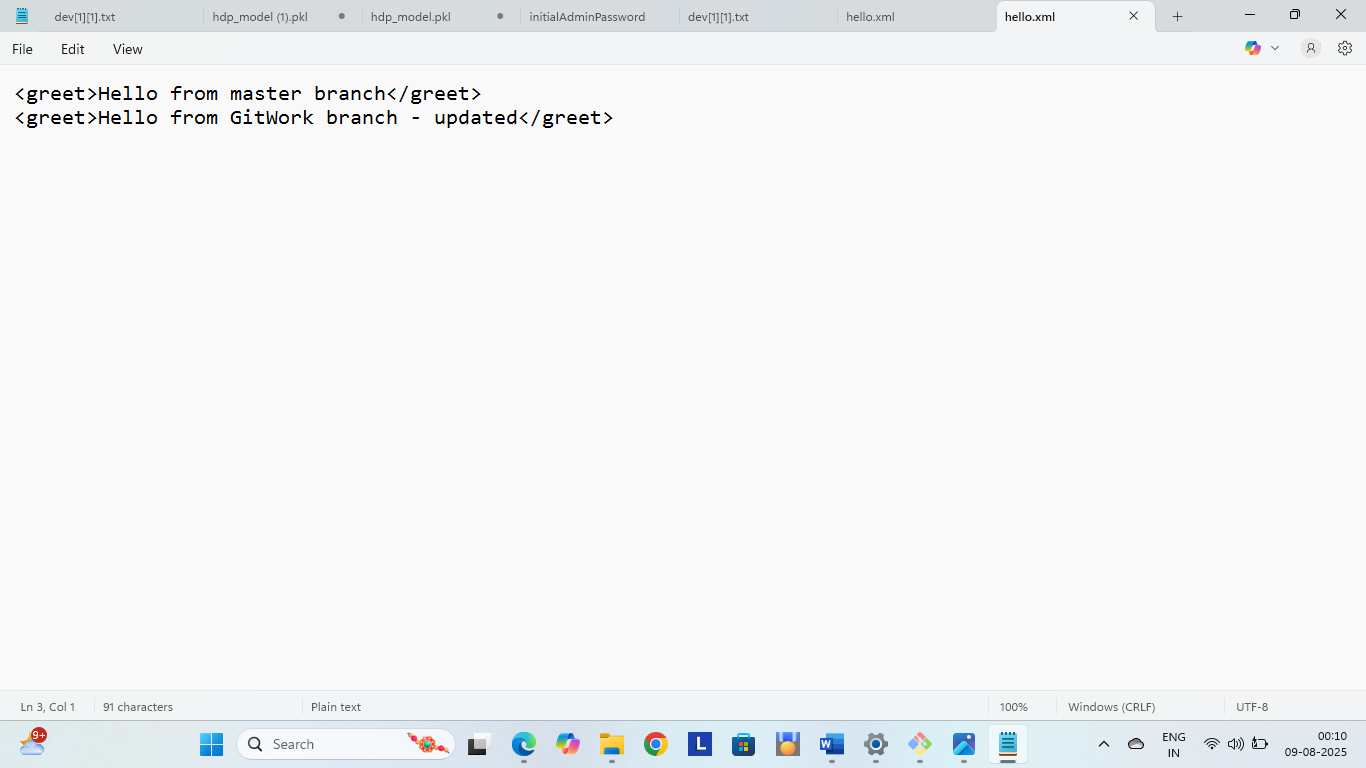
**11. Merge GitWork into master:** Merge starts but conflict occurs.

**Command:**

* git merge GitWork



**12. Check conflict markers in hello.xml :** Git shows both versions to resolve manually.



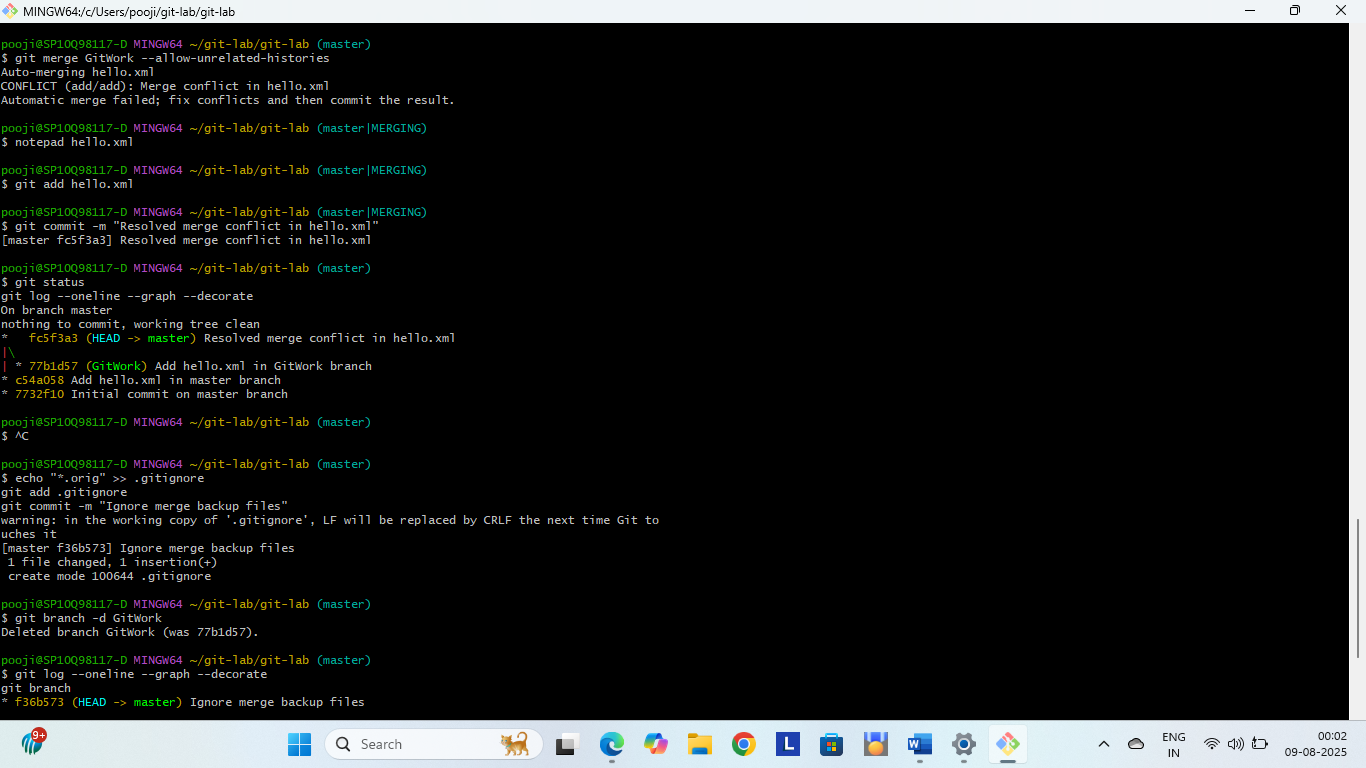
**13. Resolve with merge tool:** Opens merge UI to combine both versions.

**Command:**

* git mergetool

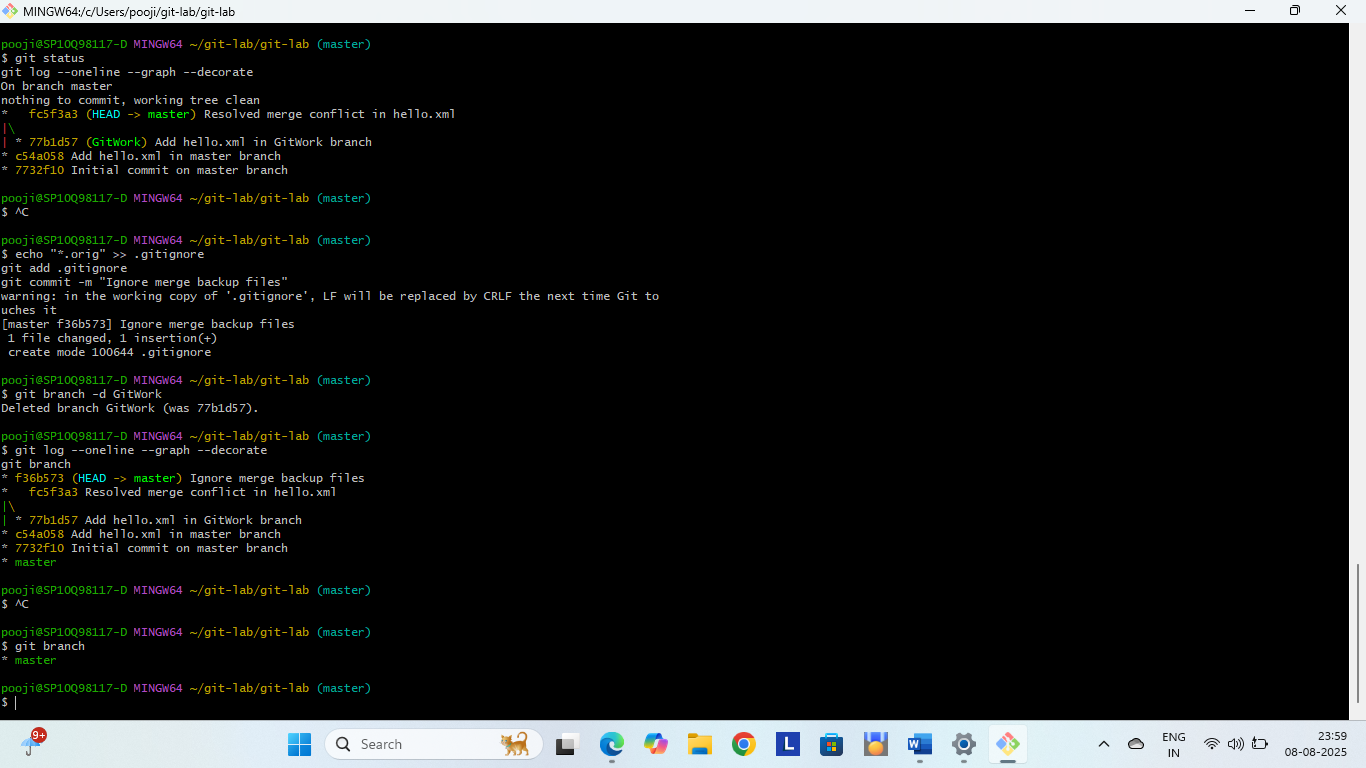
**14. Commit resolved merge:** Saves the resolved merge.

**Command:**

* git add hello.xml
* git commit -m "Merge GitWork into master - resolved conflict"

**15. Ignore merge backup files:** Prevents merge backup files from being tracked.

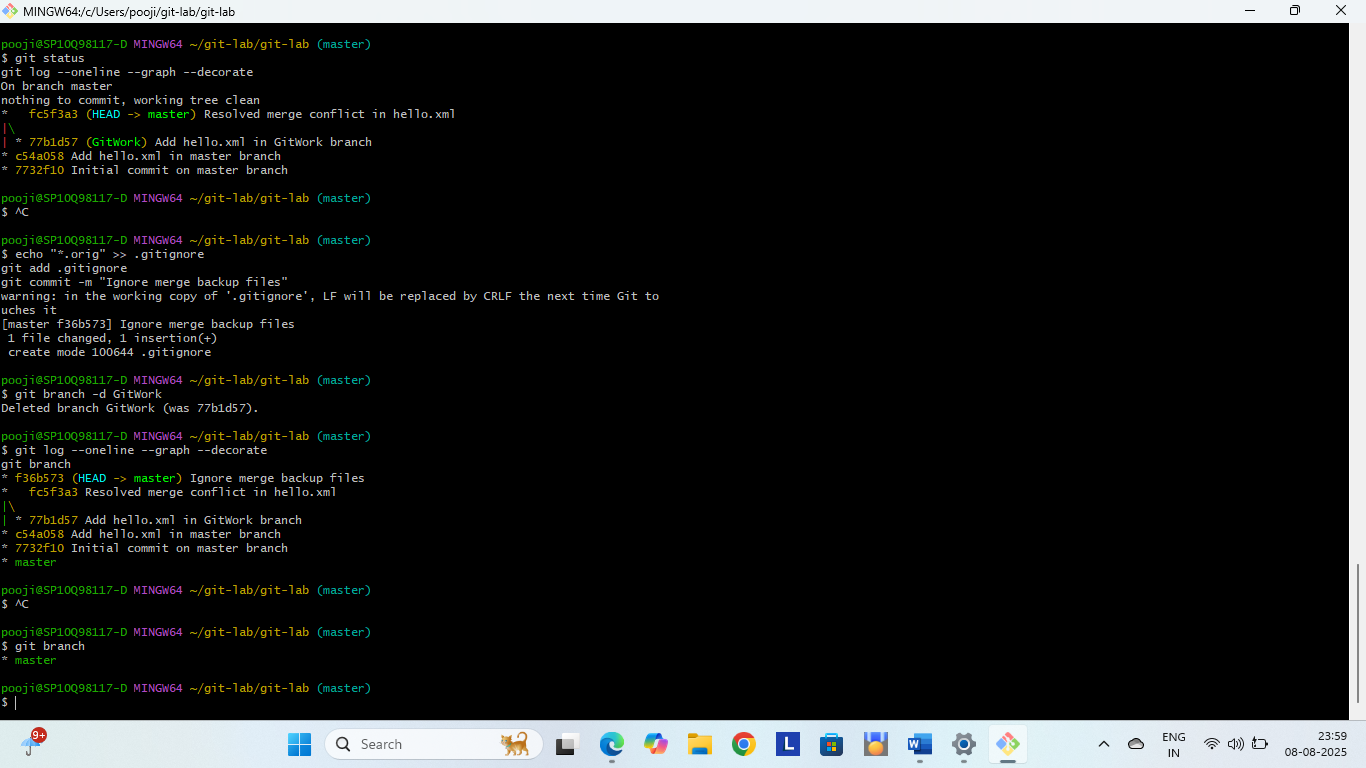
**Command:**

* echo "\*.orig" >> .gitignore
* git status

**16. Commit .gitignore:** Saves ignore rules.

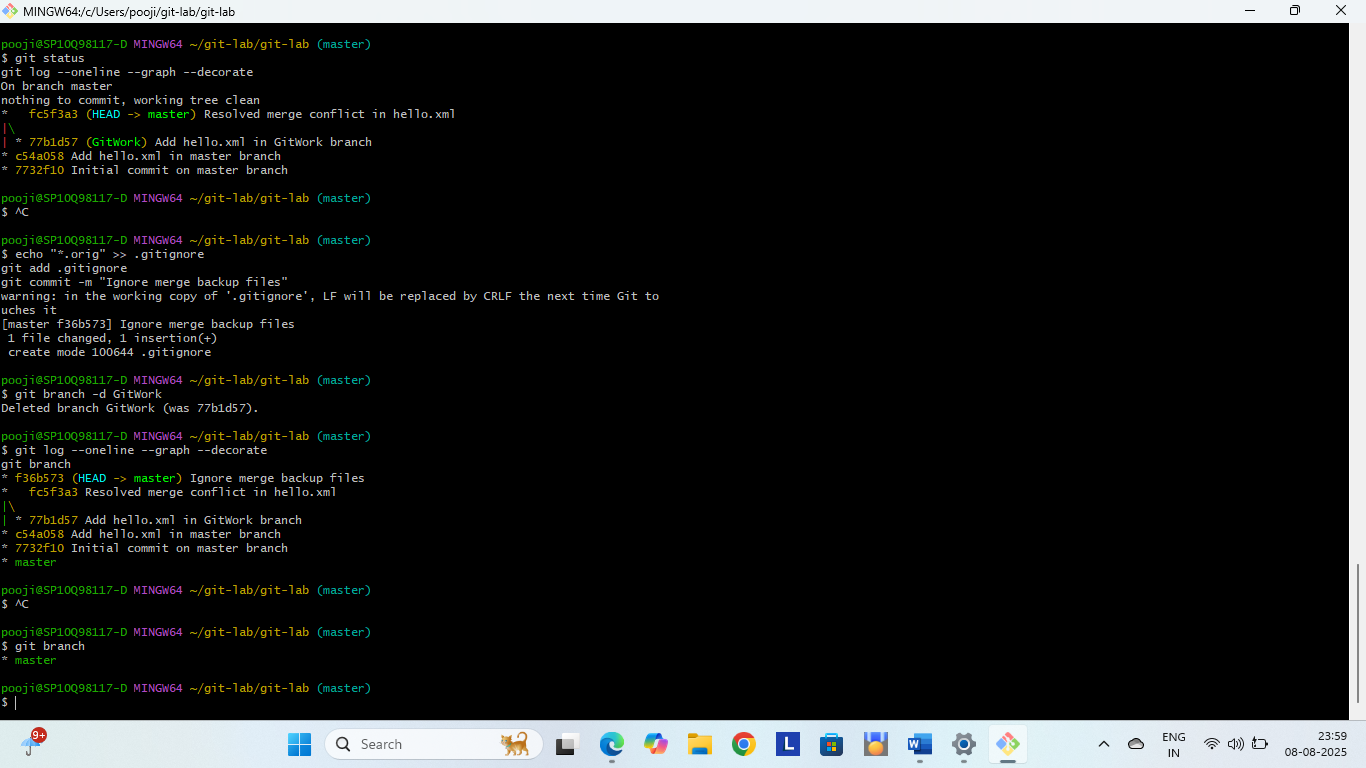
**Command:**

* git add .gitignore
* git commit -m "Ignore merge backup files"



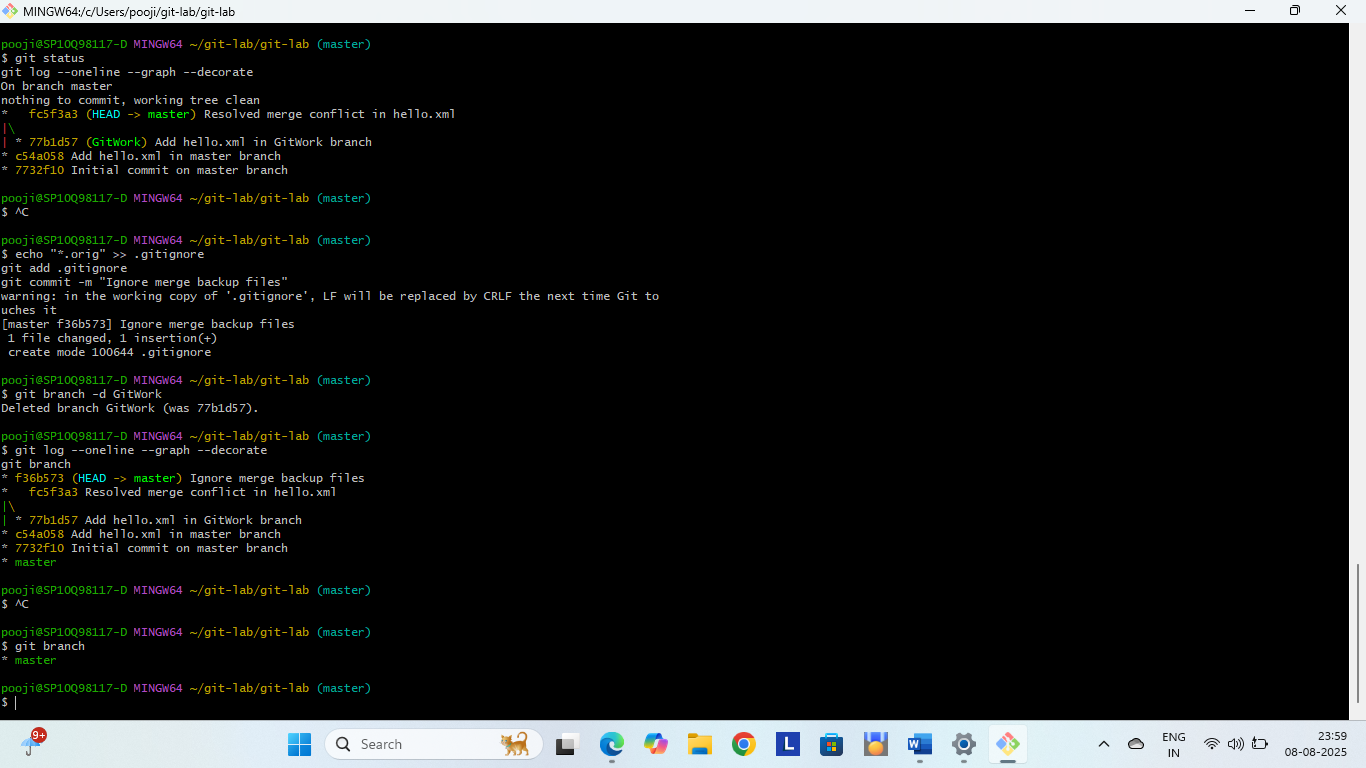
**17. List branches:** Shows both branches.

**Command:**

* git branch

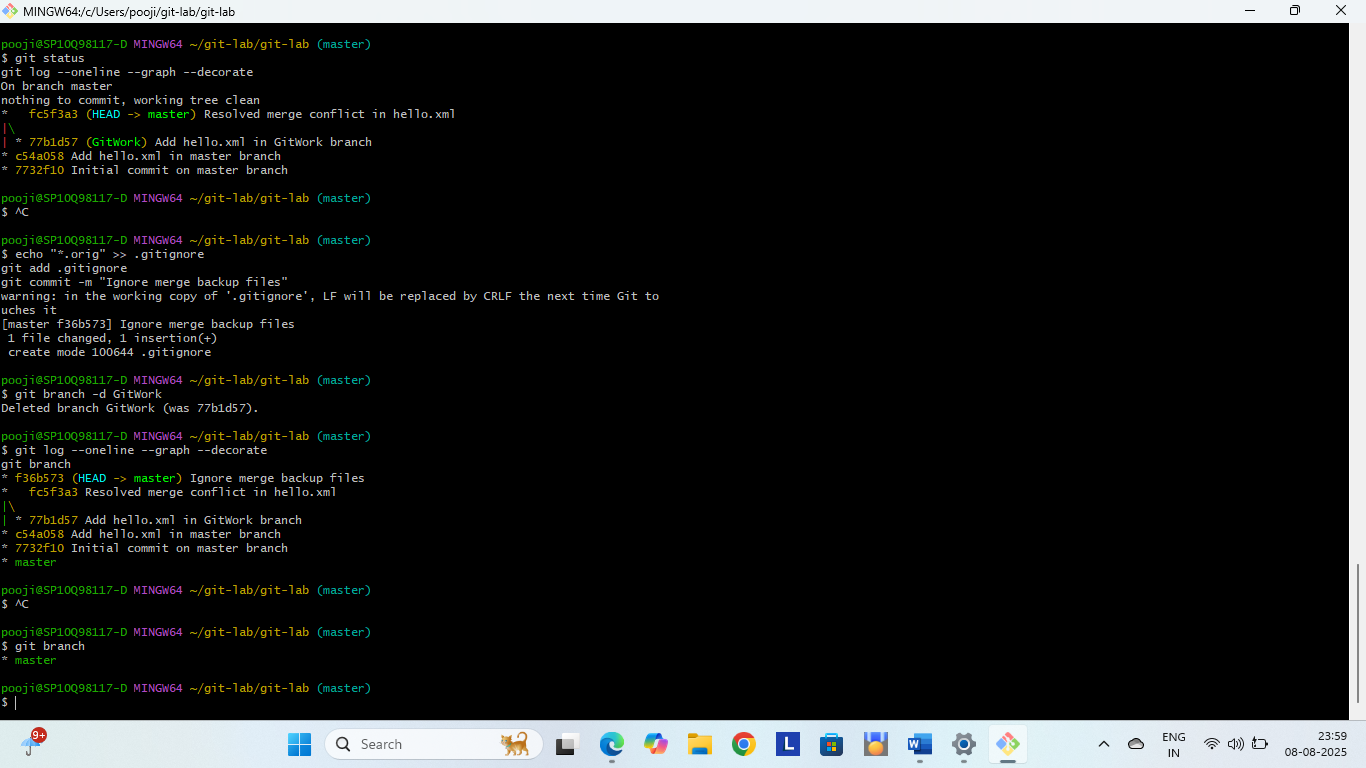
**18. Delete GitWork:** Removes GitWork after merge.

**Command:**

* git branch -d GitWork

**19. Final log:** Shows clean commit history after merge

**Command:**

* git log --oneline --graph --decorate