## **4. Git-HOL**

**Task 4: Resolving Merge Conflicts in Git**

**Objective:**

* Learn how to handle merge conflicts when different changes occur in the same file on multiple branches.
* Use manual or tool-based conflict resolution to finalize the merge.

**Step-by-Step Instructions with Commands**

**Step 1: Navigate to your local repository**

cd ~/Documents/gitdemo

**Step 2: Verify current branch is clean**

git status

**Step 3: Create and switch to a new branch GitWork**

git branch GitWork

git checkout GitWork

**Step 4: Create a file hello.xml in the repo folder**

* Add initial content:

<message>Hello from GitWork branch</message>

**Step 5: Stage and commit the new file**

git add hello.xml

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

**Step 6: Update hello.xml content**

<message>Hello from GitWork branch - updated</message>

* Save the file.

**Step 7: Stage and commit the updated file**

git add hello.xml

git commit -m "Update hello.xml in GitWork branch"

**Step 8: Switch back to main branch**

git checkout main

**Step 9: Create a conflicting hello.xml in main**

* Add different content:

<message>Hello from main branch</message>

**Step 10: Stage and commit the file in main**

git add hello.xml

git commit -m "Add hello.xml with different content in main branch"

**Step 11: View commit log graph**

git log --oneline --graph --decorate --all

**Step 12: Check differences between branches**

git diff main GitWork

**Step 13: Attempt to merge GitWork into main**

git merge GitWork

* You will get a **merge conflict error**.

**Step 14: Check files with conflicts**

git status

**Step 15: Open conflicted file (hello.xml) and manually resolve conflicts**

* Remove conflict markers <<<<<<<, =======, >>>>>>>.
* Decide final content and save.

**Step 16: Stage the resolved file**

git add hello.xml

**Step 17: Commit the resolved merge**

git commit -m "Resolved merge conflict in hello.xml"

**Step 18: Update .gitignore to ignore backup files**

* Add lines:

\*.orig

\*.backup

**Step 19: Stage and commit .gitignore**

git add .gitignore

git commit -m "Update .gitignore to ignore backup files"

**Step 20: Delete the branch GitWork locally**

git branch -d GitWork

**Step 21: Push changes to remote repository**

git push origin main

**Step 22: Verify final commit history**

git log --oneline --graph --decorate

**Notes:**

* Always check for conflicts after a merge; Git stops the process until resolved.
* Use a merge tool or manual edits to resolve conflicts.
* Commit after resolving conflicts to finalize merges.
* Clean up branches to keep repo organized.