**4. Git-HOL**

**Hands-on Lab: Resolving Merge Conflicts in Git**

**Step 1:** Verify if the master branch is in a clean state.

**Step 2:** Create a branch named 'GitWork' and add a file named 'hello.xml'.

**Step 3**: Update the content of 'hello.xml' and check the status.

**Step 4:** Commit the changes to reflect them in the branch.

**Step 5:** Switch to the master branch.

**Step 6:** Add a file 'hello.xml' to the master branch with different content than the one in the branch.

**Step 7:** Commit the changes to the master branch.

**Step 8:** View the commit history using: git log –oneline –graph –decorate –all

**Step 9**: Check the differences with the Git diff tool.

**Step 10:** For better visualization, use the P4Merge tool to list all differences between master and branch.

**Step 11:** Merge the branch into the master.

**Step 12**: Observe the Git markup after merging.

**Step 13:** Use a 3-way merge tool to resolve any conflicts.

**Step 14:** Commit the changes to the master once conflicts are resolved.

**Step 15:** Check the Git status and add any backup files to the .gitignore file.

**Step 16:** Commit the updated .gitignore file.

**Step 17:** List out all available branches.

**Step 18:** Delete the branch that was merged into master.

**Step 19:** View the commit history again using: git log –oneline –graph –decorate