## **Objectives**

* Explain branching and merging
* Explain about creating a branch request in GitLab
* Explain about creating a merge request in GitLab

In this hands-on lab, you will learn how to:

* Construct a branch, do some changes in the branch, and merge it with master (or trunk)

## **Prerequisites**

The following are the pre-requisites to complete this hands-on lab:

* Setting up Git environment with P4Merge tool for Windows

Notes\*:

Please follow the below steps for creating a free account in GitHub.

Do not use cognizant credentials to login to GitHub.

Estimated time to complete this lab: **30 minutes.**

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

**Branching:**

1. Create a new branch **“GitNewBranch”.**
2. List all the local and remote branches available in the current trunk. Observe the “\*” mark which denote the current pointing branch.
3. Switch to the newly created branch. Add some files to it with some contents.
4. Commit the changes to the branch.
5. Check the status with **“git status”** command.

**Merging:**

1. Switch to the master
2. List out all the differences between trunk and branch. These provide the differences in command line interface.
3. List out all the visual differences between master and branch using **P4Merge tool**.
4. Merge the source branch to the trunk.
5. Observe the logging after merging using **“git log –oneline –graph –decorate”**
6. Delete the branch after merging with the trunk and observe the git status.

**THEORY**

1. **Branching**
   1. A branch in Git is like a separate workspace where you can make changes without affecting the main code (usually master or main).
   2. You can create multiple branches to work on new features, bug fixes, or experiments.
   3. This keeps the main branch stable while development happens in isolation.
2. **Merging**
   1. Merging is the process of integrating changes from one branch into another.
   2. For example, after finishing work in a feature branch, you merge it into the master branch so that the new changes become part of the main project.
3. **Creating a Merge Request in GitLab**
   1. A merge request (MR) in GitLab is a way to propose changes from one branch into another, allowing others to review and approve before merging.
   2. Steps in GitLab:
      1. Push your branch to the remote repository.
      2. Go to GitLab, select your project.
      3. Click **Merge Requests** → **New Merge Request**.
      4. Choose the source branch and target branch.
      5. Add a title, description, and reviewers, then submit.

### **Step 1 – Create a new branch**

$ git branch GitNewBranch

Check branches:

$ git branch  
\* master  
 GitNewBranch

### **Step 2 – Switch to the new branch**

$ git checkout GitNewBranch  
Switched to branch 'GitNewBranch'

Check again:

$ git branch  
 master  
\* GitNewBranch

### **Step 3 – Add files and commit**

$ echo "This is a new feature" > feature.txt  
$ git add feature.txt  
$ git commit -m "Added feature.txt in GitNewBranch"  
[GitNewBranch 1a2b3c4] Added feature.txt in GitNewBranch  
 1 file changed, 1 insertion(+)  
 create mode 100644 feature.txt

Status check:

$ git status  
On branch GitNewBranch  
nothing to commit, working tree clean

### **Step 4 – Switch back to master**

$ git checkout master  
Switched to branch 'master'

### **Step 5 – Show differences**

$ git diff master GitNewBranch

$ git mergetool

### **Step 6 – Merge the branch into master**

$ git merge GitNewBranch  
Updating 9b8c7d6..1a2b3c4  
Fast-forward  
 feature.txt | 1 +  
 1 file changed, 1 insertion(+)  
 create mode 100644 feature.txt

**Step 7 – View log after merge**

$ git log --oneline --graph --decorate  
\* 1a2b3c4 (HEAD -> master, GitNewBranch) Added feature.txt in GitNewBranch  
\* 9b8c7d6 Initial commit

**Step 8 – Delete the branch**

$ git branch -d GitNewBranch  
Deleted branch GitNewBranch (was 1a2b3c4).

Status check:

$ git status  
On branch master  
nothing to commit, working tree clean