Working with Branches in Git

Objective

The objective of this lab assignment is to guide students through the process of creating and managing branches in Git. By the end of this assignment, students will be able to:

1. Create a new branch.
2. Switch between branches.
3. Merge branches.
4. Resolve merge conflicts.
5. Delete branches.

Prerequisites

1. Basic understanding of Git.
2. Git installed on your local machine.
3. A local Git repository.

Instructions

Step 1: Open Terminal or Command Prompt

Open your terminal (on macOS or Linux) or Command Prompt (on Windows).

Step 2: Navigate to Your Local Repository

Use the `cd` command to navigate to the directory of your local project. For example:

cd path/to/your/local/repository

Step 3: Check the Current Branch

Check which branch you are currently on using:

git branch

Example:

$ git branch

* master

Step 4: Create a New Branch

Create a new branch called `feature-branch` using:

git branch feature-branch

Example:

$ git branch feature-branch

Step 5: Switch to the New Branch

Switch to the newly created branch using:

git checkout feature-branch

Example:

$ git checkout feature-branch

Switched to branch 'feature-branch'

Step 6: Make Changes in the New Branch

Make some changes to your files in the `feature-branch`. For example, create a new file called `feature.txt` and add some content to it.

Example:

$ echo "This is a new feature." > feature.txt

$ git add feature.txt

$ git commit -m "Add new feature"

[feature-branch abcdef2] Add new feature

1 file changed, 1 insertion(+)

create mode 100644 feature.txt

Step 7: Switch Back to the Master Branch

Switch back to the `master` branch using:

git checkout master

Example:

$ git checkout master

Switched to branch 'master'

Step 8: Merge the Feature Branch into Master

Merge the changes from `feature-branch` into `master` using:

git merge feature-branch

Example:

$ git merge feature-branch

Updating abcdef1..abcdef2

Fast-forward

feature.txt | 1 +

1 file changed, 1 insertion(+)

create mode 100644 feature.txt

Step 9: Resolve Merge Conflicts (if any)

If there are any merge conflicts, Git will notify you. Open the conflicted files, resolve the conflicts, and then add the resolved files to the staging area using `git add`. Finally, commit the merge using `git commit`.

Example:

$ git add conflicted-file.txt

$ git commit -m "Resolve merge conflict"

Step 10: Delete the Feature Branch

Delete the `feature-branch` after merging it into `master` using:

git branch -d feature-branch

Example:

$ git branch -d feature-branch

Deleted branch feature-branch (was abcdef2).