

## **Placement Empowerment Program**

### ***Cloud Computing and DevOps Centre***

Create a new branch in your Git repository for testing .  
Add a new feature and merge it

Name: ERIC JEEVAN A

Department: ADS

# Introduction:

In this Proof of Concept (POC), Git is used for version control to manage the development workflow. Git allows developers to create separate branches for new features, isolate them from the main branch, and merge them back after completion. This ensures organized and collaborative development.

## Overview:

This POC demonstrates how to:

1. Initialize a Git repository.
2. Create and switch between branches.
3. Commit changes in different branches.
4. Merge feature branches into the main branch.
5. Delete branches after completing the work.

## Objectives:

1. To initialize and set up a Git repository.
2. To create and manage feature branches (e.g., testing-feature).
3. To demonstrate adding, committing, and merging code.
4. To showcase how to delete branches after their purpose is served.
5. To learn how to resolve merge conflicts if any arise during the process.

# Importance:

- 1. Version Control:** Helps track changes, revert to previous versions, and avoid conflicts in the codebase.
- 2. Collaboration:** Different team members can work on separate features simultaneously without interfering with each other's work.
- 3. Branching:** Isolates new features or bug fixes, ensuring stability in the main branch (master or main).
- 4. Efficiency:** Merging branches allows rapid integration of new features without disrupting ongoing work.
- 5. Clean Workflow:** Deleting feature branches after merging keeps the repository clean and manageable.

## Step-by-Step Overview

### Step 1:

Create a folder and name it (Git\_Branching).



## Step 2:

Set the path to the folder created in first step (Git\_Branching).

```
C:\Users\samni>cd "C:\Users\samni\Desktop\Branching"
```

## Step 3:

Initialize Git by typing this command:

**git init**

This command will create a .git folder inside your folder, which tells Git to start tracking your files.

```
C:\Users\samni\Desktop\Git Branching>git init  
Initialized empty Git repository in C:/Users/samni/Desktop/Git Branching/.git/
```

## Step 4:

Create a simple file to start the repository:

```
C:\Users\samni\Desktop\Git Branching>echo "1st file content" > first-file.txt
```

## Step 5:

Add the File to Git

Tell Git to track this file:

```
C:\Users\samni\Desktop\Git Branching>git add .
```

## Step 6:

Save this change in Git with a commit message.

```
C:\Users\samni\Desktop\Git Branching>git commit -m "Initial Commit"
[master (root-commit) 33245af] Initial Commit
1 file changed, 1 insertion(+)
create mode 100644 first-file.txt
```

## Step 7:

Create and switch to a new branch called testing-feature.

```
C:\Users\samni\Desktop\Git Branching>git checkout -b testing-feature
Switched to a new branch 'testing-feature'
```

## Step 8:

Let's add a new file for our feature:

```
C:\Users\samni\Desktop\Git Branching>echo "1st file content" > first-file.txt
```

## Step 9:

Now, stage the changes:

```
C:\Users\samni\Desktop\Git Branching>git add .
```

## Step 10:

Commit the changes:

```
C:\Users\samni\Desktop\Git Branching>git commit -m "add new feature"
```

## Step 11:

Switch to the master Branch

```
C:\Users\samni\Desktop\Git Branching>git checkout master  
Switched to branch 'master'
```

## Step 12:

Merge Changes from testing-feature to master

```
C:\Users\samni\Desktop\Git Branching>git merge testing-feature  
Updating 33245af..a3ae022  
Fast-forward  
 first-file.text | 1 +  
 1 file changed, 1 insertion(+)  
 create mode 100644 first-file.text
```

## Step 13:

Once the merge is done, you can delete the testing-feature branch:

```
C:\Users\samni\Desktop\Git Branching>git branch -d testing-feature  
Deleted branch testing-feature (was a3ae022).
```

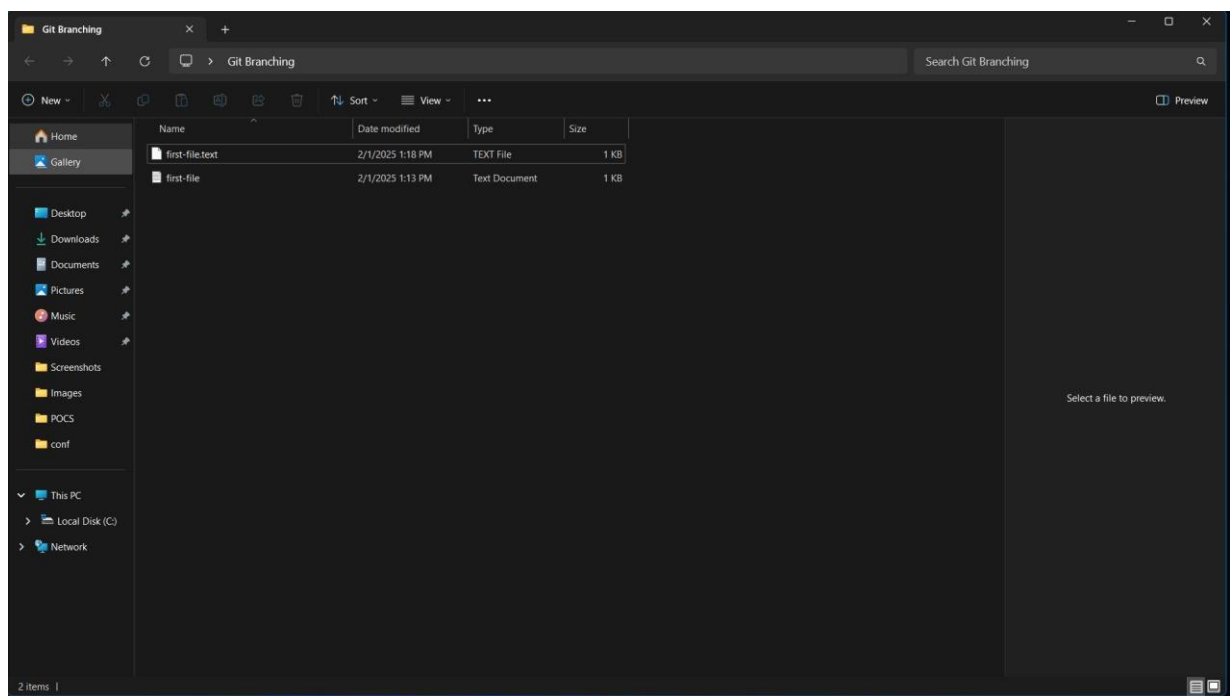
## Step 14:

Now, check the files in the folder:

```
C:\Users\samni\Desktop\Git Branching>dir
Volume in drive C has no label.
Volume Serial Number is E29E-BC01

Directory of C:\Users\samni\Desktop\Git Branching

02/01/2025  01:18 PM    <DIR>          .
02/01/2025  01:11 PM    <DIR>          ..
02/01/2025  01:18 PM                21 first-file.text
02/01/2025  01:13 PM                21 first-file.txt
                2 File(s)                42 bytes
                2 Dir(s)  236,272,271,360 bytes free
```



# Outcome

By completing this PoC of managing branches in Git for a local repository, you will:

1. Successfully initialize a Git repository in your local project folder.
2. Create and manage multiple branches for feature development and experimentation.
3. Track and commit changes made to files in different branches.
4. Merge feature branches back into the main branch while maintaining project integrity.
5. Gain hands-on experience with key Git commands such as `git init`, `git add`, `git commit`, `git checkout`, and `git merge`.