Placement Empowerment Program

Cloud Computing and DevOps Centre

TASK-5

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

Name:ARUN SINGH Department:CSE

St. JOSEPH'S COLLEGE OF ENGINEERING | St. JOSEPH'S INSTITUTE OF TECHNOLOGY

AUTONOMOUS INSTITUTIONS, AFFILIATED TO ANNA UNIVERSITY

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.

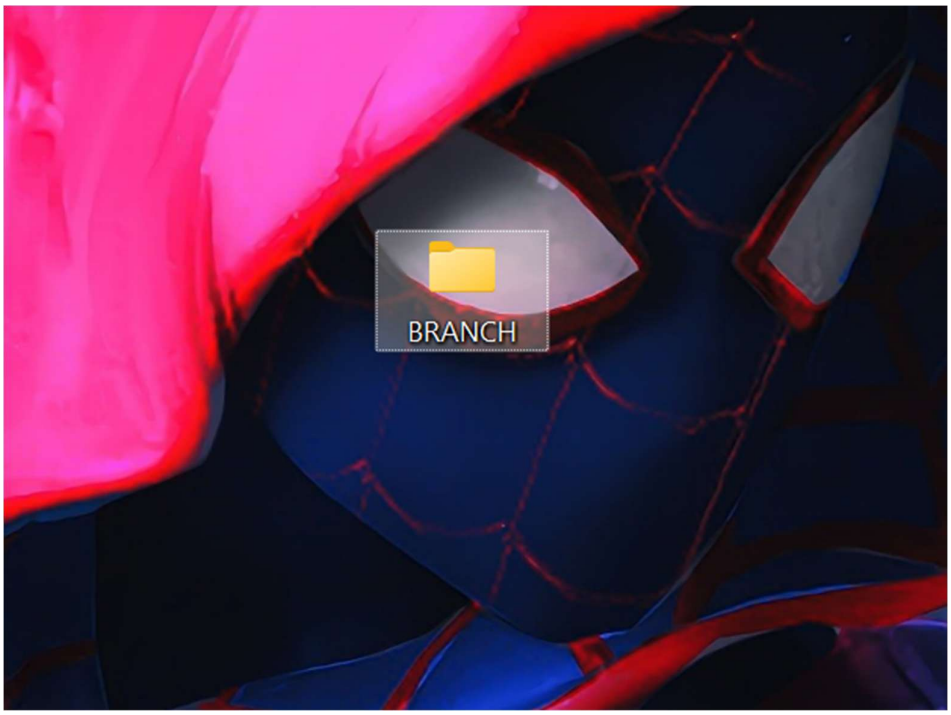
Objectives:

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.

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\Staff>cd C:\Users\Staff\Desktop\BRANCH
```

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

Step 4:

Create a simple file to start the repository:

```
C:\Users\Staff\Desktop\BRANCH>echo "initial file" > file.txt
```

Step 5:

Add the File to Git

Tell Git to track this file:

```
C:\Users\Staff\Desktop\BRANCH>git add .
```

Step 6:

Save this change in Git with a commit message.

```
C:\Users\Staff\Desktop\BRANCH>git commit -m "add"
[master (root-commit) cbb2a78] add
 1 file changed, 1 insertion(+)
 create mode 100644 file.txt
```

Step 7:

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

```
C:\Users\Staff\Desktop\BRANCH>git checkout -b new
Switched to a new branch 'new'
```

Step 8:

Let’s add a new file for our feature:

```
C:\Users\Staff\Desktop\BRANCH>echo "file commit" > file.txt
```

Step 9:

Now, stage the changes

```
C:\Users\Staff\Desktop\BRANCH>git add .
```

Step 10:

Commit the changes:

```
C:\Users\Staff\Desktop\BRANCH>git commit -m "add branch"
[new 2b38b87] add branch
 1 file changed, 1 insertion(+), 1 deletion(-)
```

Step 11:

Switch to the master Branch.

```
C:\Users\Staff\Desktop\BRANCH>git checkout master
Switched to branch 'master'
```

Step 12:

Merge Changes from testing-feature to master.

```
C:\Users\Staff\Desktop\BRANCH>git merge new
Updating cbb2a78..2b38b87
Fast-forward
 file.txt | 2 +-
 1 file changed, 1 insertion(+), 1 deletion(-)
```

Step 13:

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

```
C:\Users\Staff\Desktop\BRANCH>git branch -d new
Deleted branch new (was 2b38b87).
```

Step 14:

Now, check the files in the folder:

```
 Directory of C:\Users\Staff\Desktop\BRANCH

31-01-2025  00:01    <DIR>          .
30-01-2025  23:37    <DIR>          ..
31-01-2025  00:01                16 file.txt
               1 File(s)             16 bytes
               2 Dir(s)  86,832,152,576 bytes free

C:\Users\Staff\Desktop\BRANCH>
```