# 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: Goutham K Department: EEE



# Introduction:

In this Proof of Concept (POC), Git is utilized for version control to streamline the development process. Git enables developers to create separate branches for new features, keeping them isolated from the main branch. Once the new features are completed, they can be merged back into the main branch. This approach fosters organized, collaborative development and maintains a clean, stable codebase throughout the project's lifecycle.

# 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.

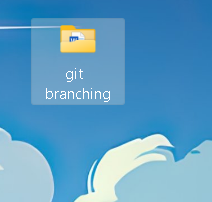
# Importance:

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:

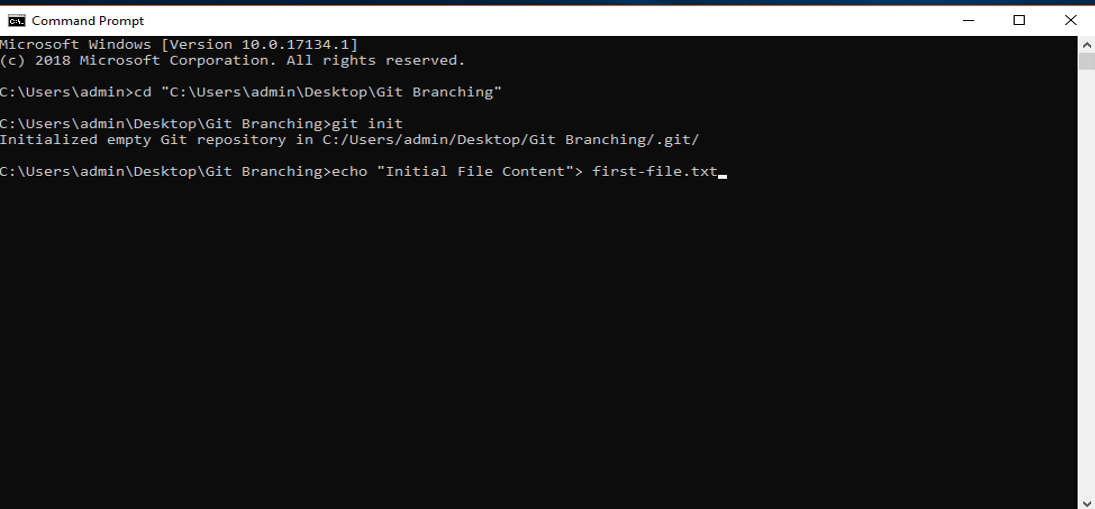
Set the path to the folder created in first step (Git\_Branching).

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.

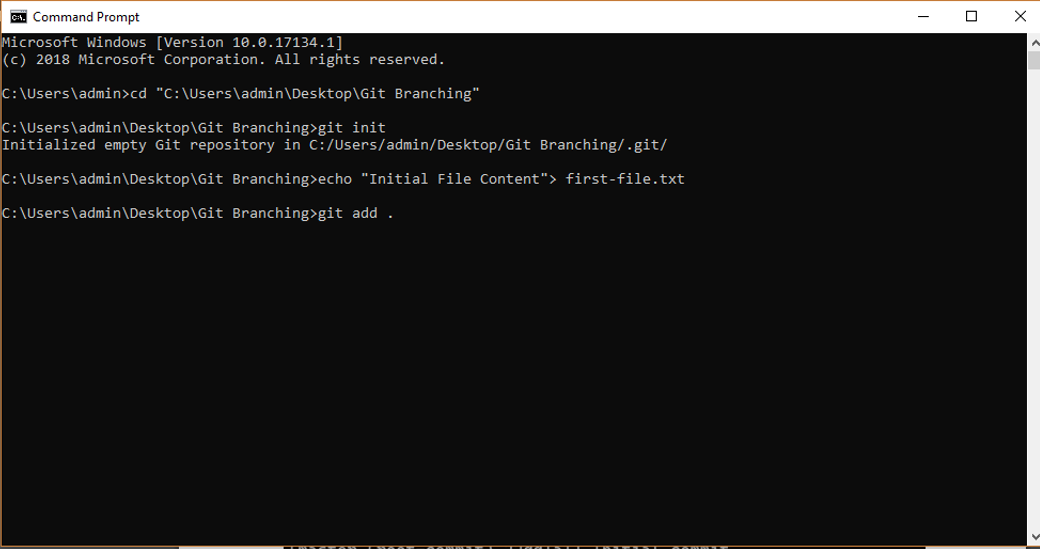
Create a simple file to start the repository:



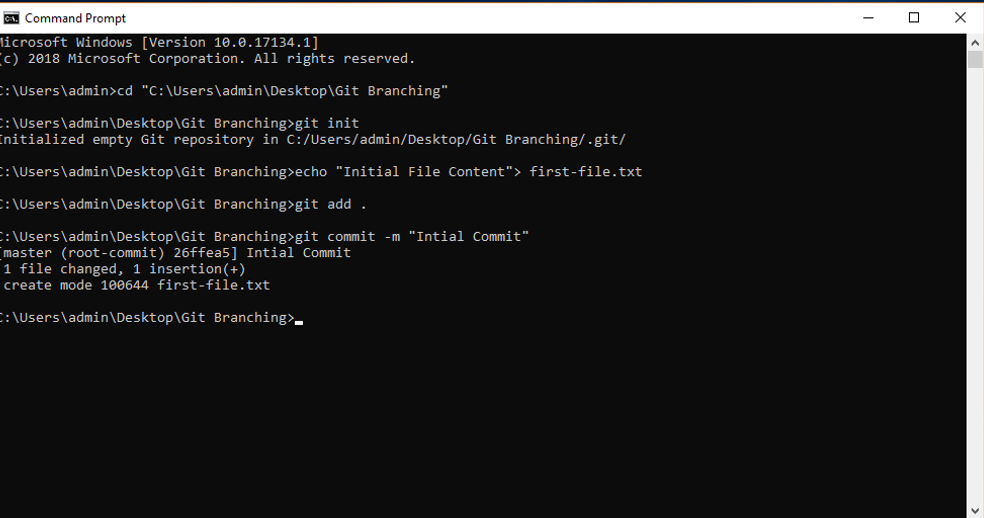
## Step 4:

Add the File to Git

Tell Git to track this file:

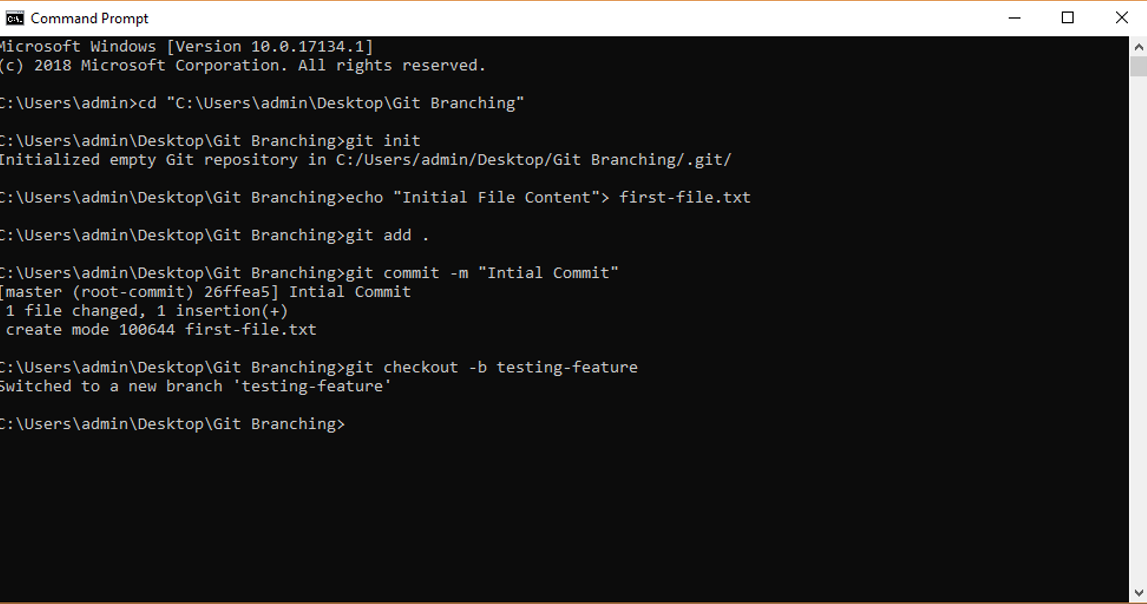


## Step 5:

Save this change in Git with a commit message. 

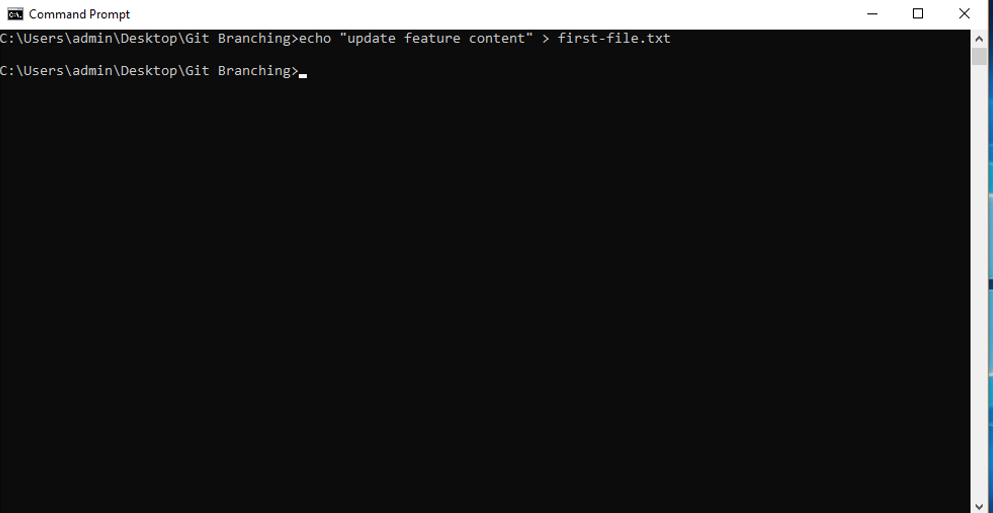
## Step 6:

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



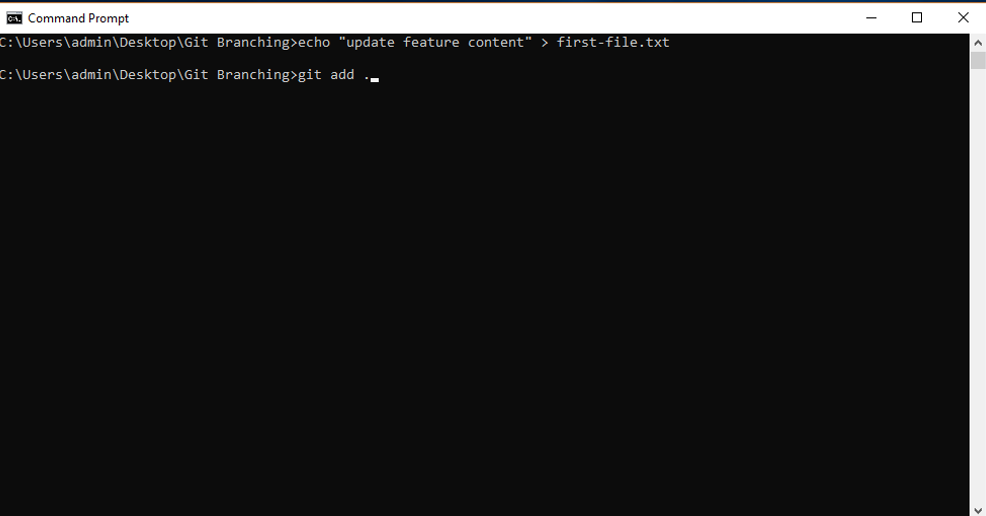
## Step 7:

Let’s add a new file for our feature:



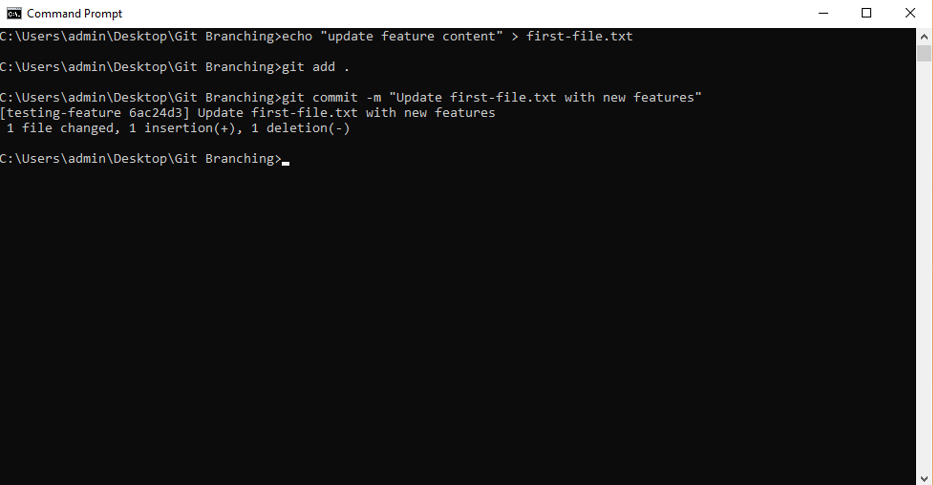
## Step 8:

Now, stage the changes:



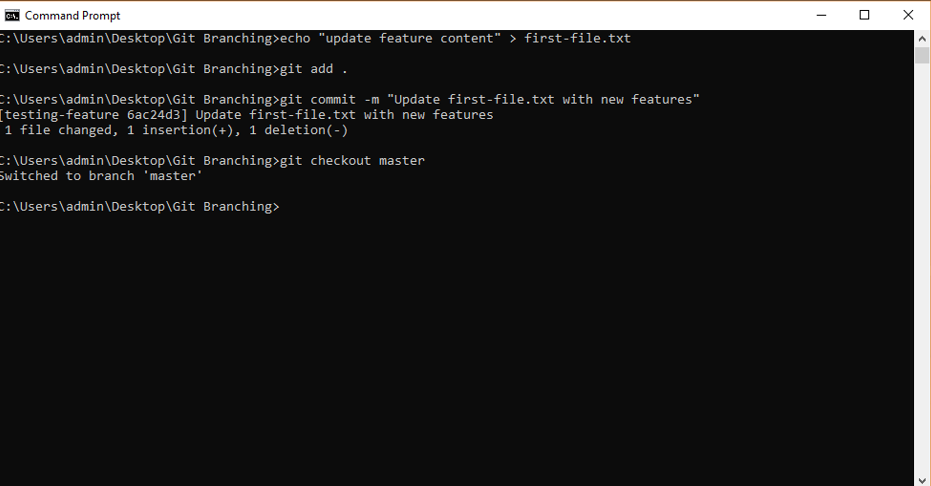
## Step 9:

Commit the changes:



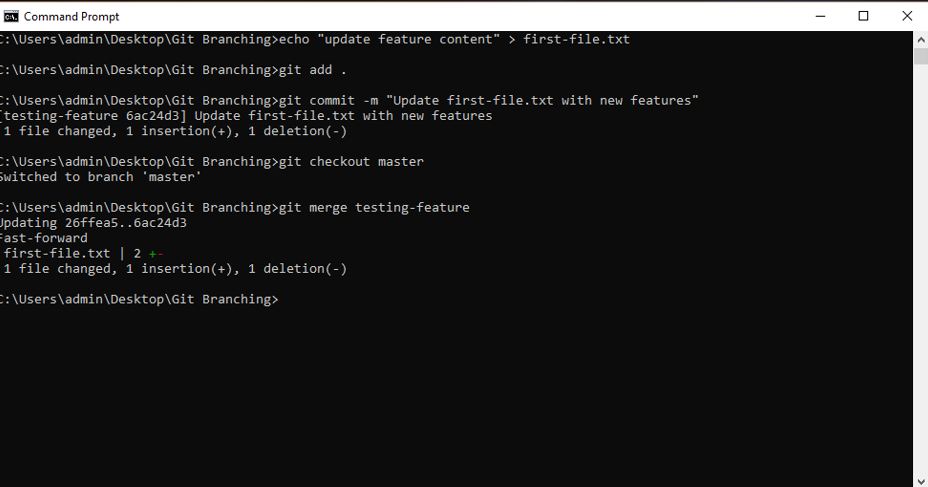
## Step 10:

Switch to the master Branch



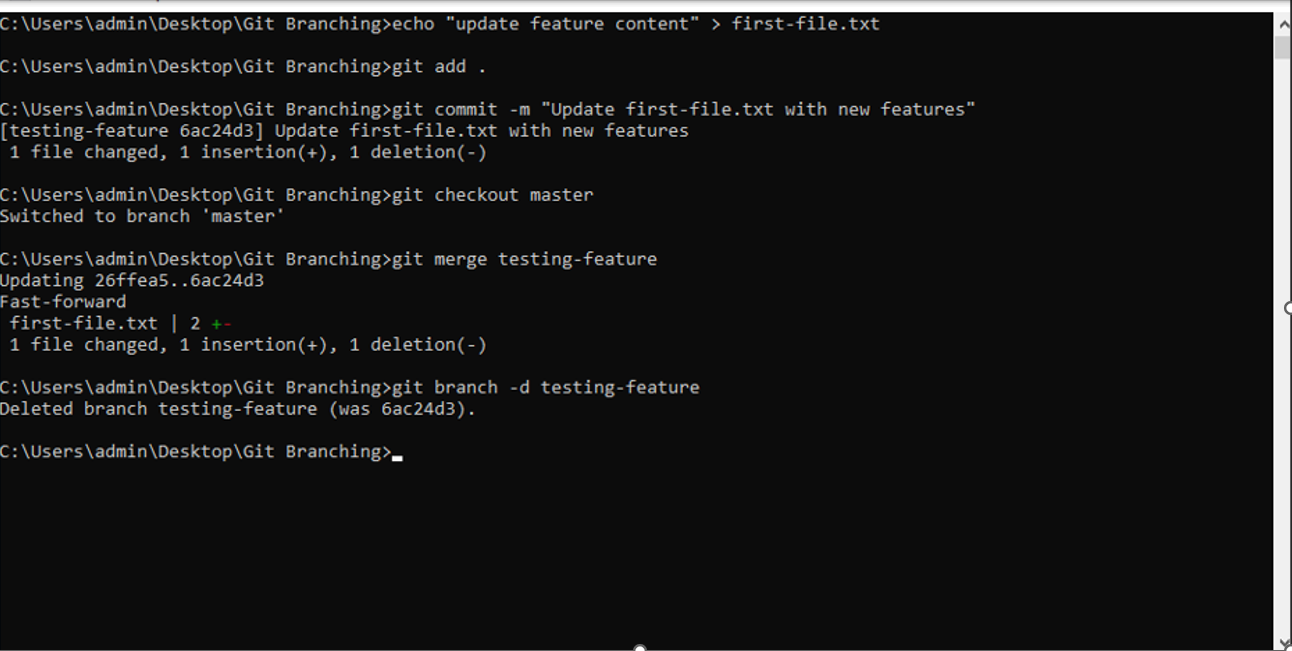
## Step 11:

Merge Changes from testing-feature to master



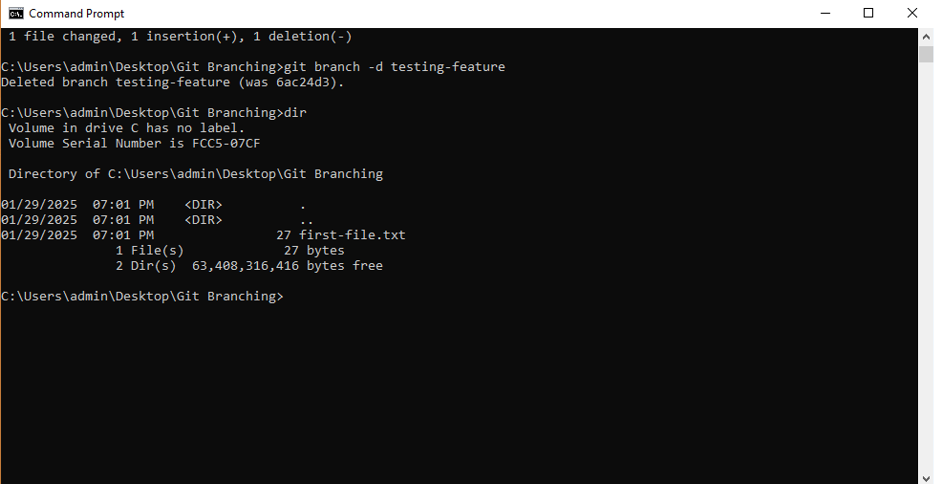
## Step 12:

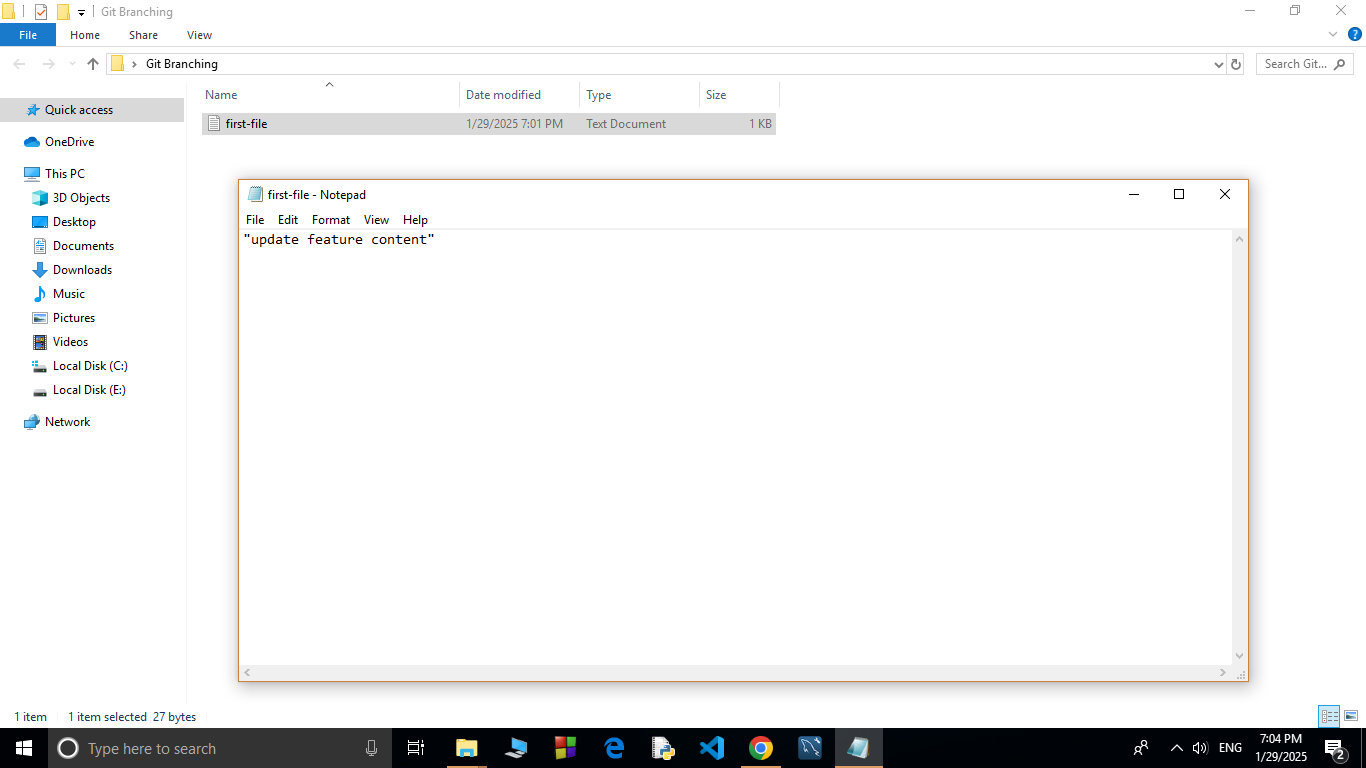
Once the merge is done, you can delete the testing-feature branch:



## Step 13:

Now, check the files in the folder:





## Outcome

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

* Initialize a Git repository in your local project folder.
* Create and manage multiple branches for feature development and experimentation.
* Track and commit changes made to files in various branches.
* Merge feature branches back into the main branch while preserving project integrity.
* Gain practical experience with essential Git commands such as git init, git add, git commit, git checkout, and git merge.