# Placement Empowerment Program

***Cloud Computing and DevOps Centre***

Set Up Git Branching: Create a new branch in your Git repository for testing .

Add a new feature and merge it

Name: Kowshika V Department: CSE



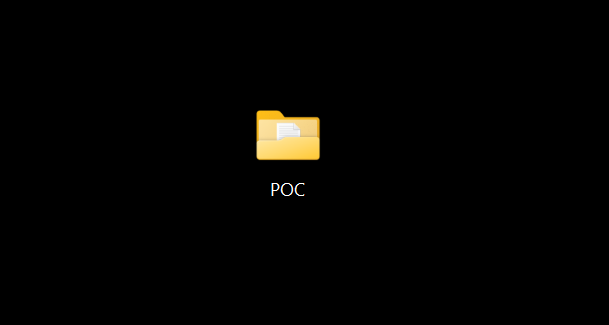
# Introduction:

In this Proof of Concept (POC), Git is utilized for version control to streamline the development process. It enables developers to work on new features in separate branches, keeping them independent from the main branch. Once the features are complete, they can be merged back, promoting structured and collaborative development

# Step-by-Step Overview

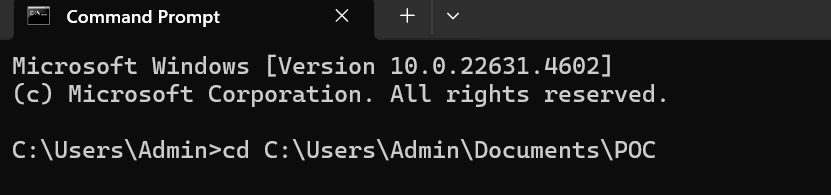
## Step 1:

Create a folder and name it (poc).



## Step 2:

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

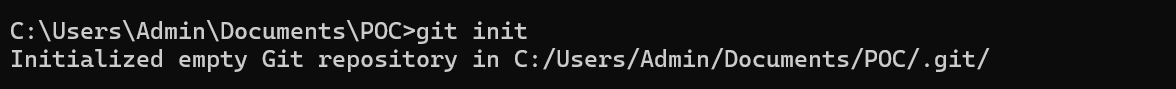


## Step 3:

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.



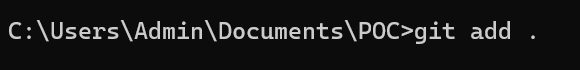
## Step 4:

Create a simple file to start the repository:

Screenshot 2025-01-30 205002

## Step 5:

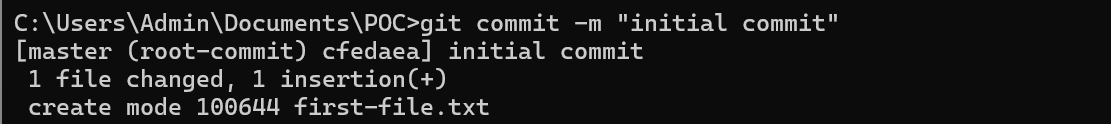
Add the File to Git



Tell Git to track this file:

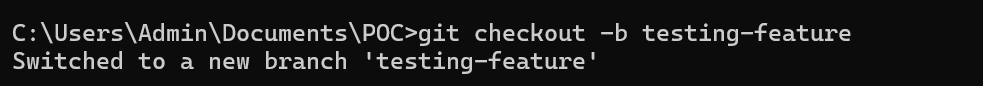
## Step 6:

Save this change in Git with a commit message.



## Step 7:

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



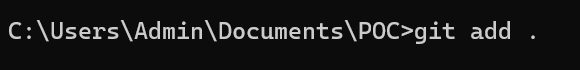
## Step 8:

Let’s add a new file for our feature:

Screenshot 2025-01-30 205002

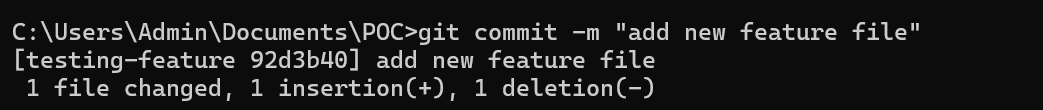
## Step 9:

Now, stage the changes:



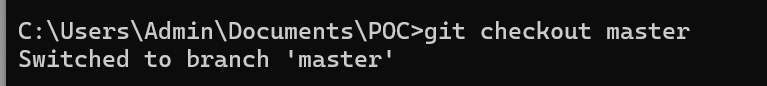
## Step 10:

Commit the changes:



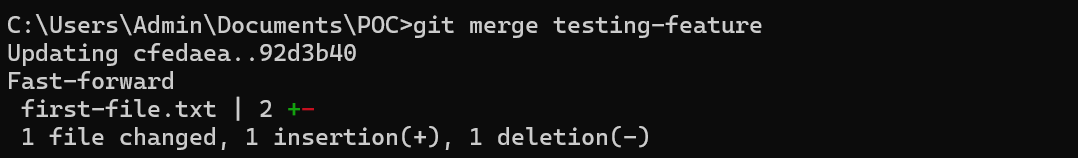
## Step 11:

Switch to the master Branch



## Step 12:

Merge Changes from testing-feature to master



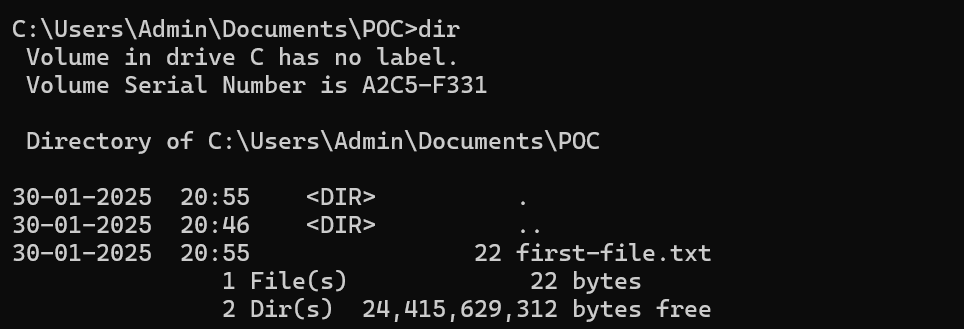
## Step 13:

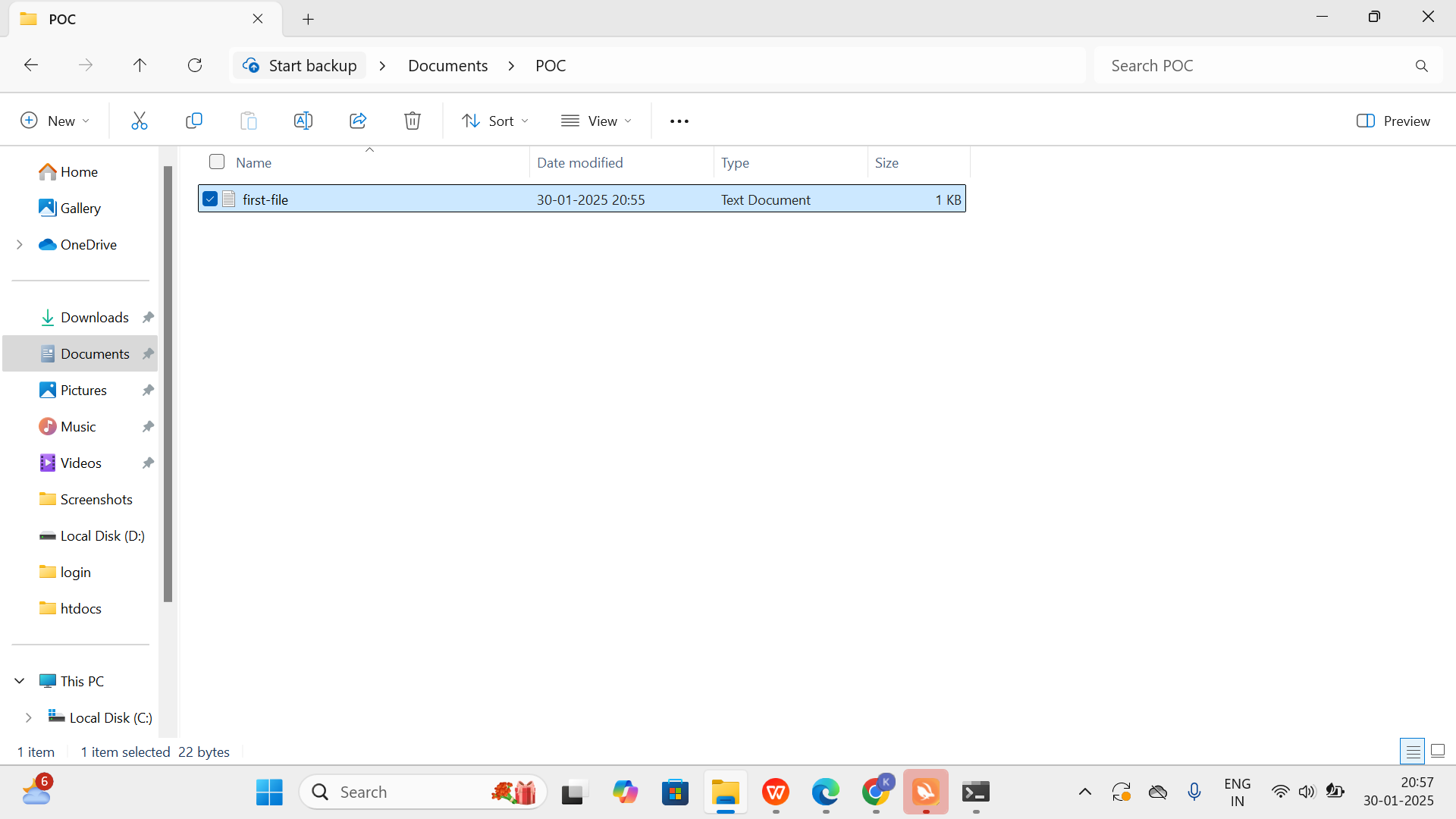
Once the merge is done, you can delete the testing-feature branch:



## Step 14:

Now, check the files in the folder:





## 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.