# 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: Shakthi Lakshmanan V Department : CSE



### **Introduction**

This guide shows how to use Git for version control. Git helps developers track changes, work on new features separately, and merge them back when ready. This keeps development organized and makes teamwork easier.

### **What You Will Learn**

1. How to set up a Git repository.
2. How to create and switch between branches.
3. How to save changes in different branches.
4. How to merge branches into the main project.
5. How to delete branches after finishing work.

### **Objectives**

* Set up a Git repository.
* Create and manage branches.
* Add and save changes using Git.
* Merge changes into the main branch.
* Handle merge conflicts if they happen.

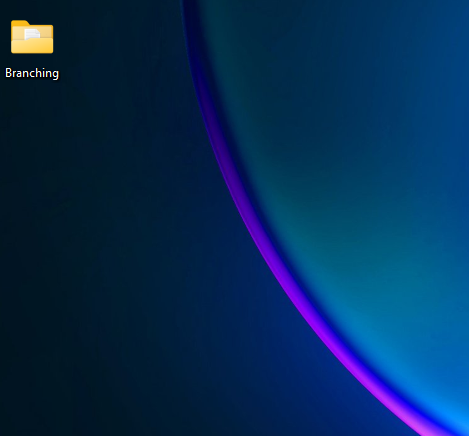
### **Why Use Git?**

* **Track Changes** – Keep a history of updates and go back if needed.
* **Work Together** – Multiple people can work on different features without issues.
* **Keep Things Organized** – Branches help separate work and keep the main project stable.
* **Improve Workflow** – Merging branches makes it easy to add new features.
* **Clean Up Easily** – Deleting branches after merging keeps the project tidy.

# Step-by-Step Overview

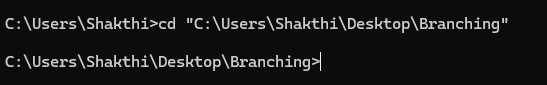
## Step 1:

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



## Step 2:

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



## 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:



## Step 5:

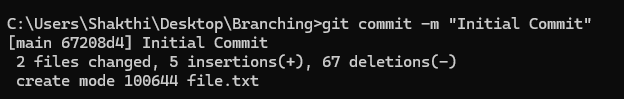
Add the File to Git

Tell Git to track this file:



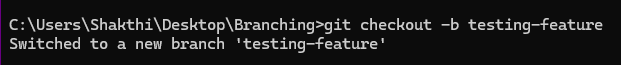
## Step 6:

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:



## Step 9:

Now, stage the changes:



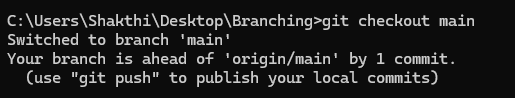
## Step 10:

Commit the changes:



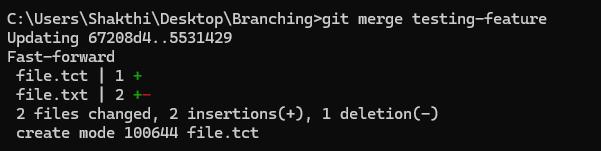
## Step 11:

Switch to the master Branch



## Step 12:

Merge Changes from testing-feature to master



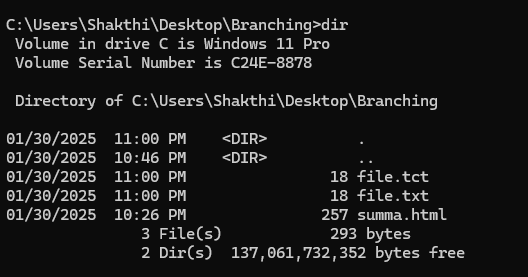
## Step 13:

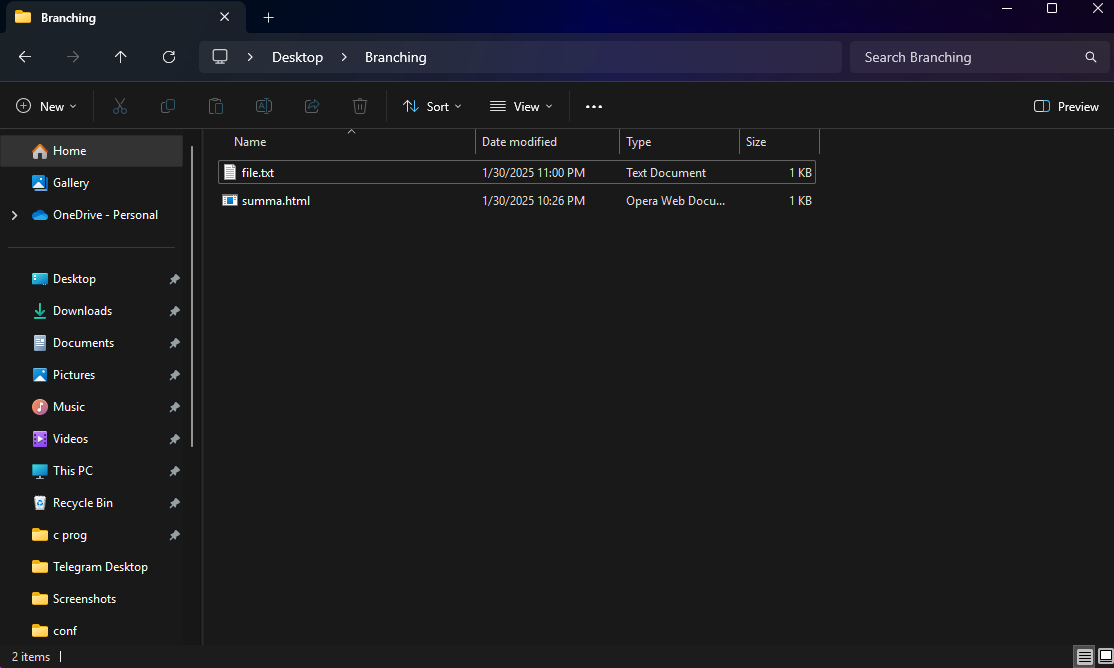
Once the merge is done, you can delete the testing-feature branch:



## 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.