

**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: SREEVATSAN V P Department : AIML



**Introduction**

**This Proof of Concept (POC) showcases the use of Git for version control in managing the development process. Git enables developers to create separate branches for new features, keeping them isolated from the main branch until they are ready to be merged. This approach promotes an organized and collaborative workflow.**

**Overview**

**This POC covers:**

1. **Setting up a Git repository.**
2. **Creating and switching between branches.**
3. **Committing changes within different branches.**
4. **Merging feature branches into the main branch.**
5. **Removing branches after their tasks are completed.**

**Objectives**

1. **Initialize and configure a Git repository.**
2. **Create and manage feature branches (e.g., testing-feature).**
3. **Add, commit, and merge code changes effectively.**
4. **Remove branches once they are no longer needed.**
5. **Understand how to handle merge conflicts when they occur.**

**Significance**

1. **Version Control – Tracks modifications, enables rollbacks, and minimizes conflicts.**
2. **Team Collaboration – Allows multiple developers to work on different features simultaneously without interference.**
3. **Branching Strategy – Ensures new features and bug fixes remain isolated, maintaining stability in the main branch.**
4. **Efficient Development – Facilitates seamless integration of new changes without disrupting ongoing work.**
5. **Organized Workflow – Keeps the repository clean by deleting feature branches post-merge.**

**Step-by-Step Overview**   
  
 Step1:

A yellow folder with green check mark and white text

Description automatically generatedCreate a folder and name it (Git\_Branching).

Step 2:

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

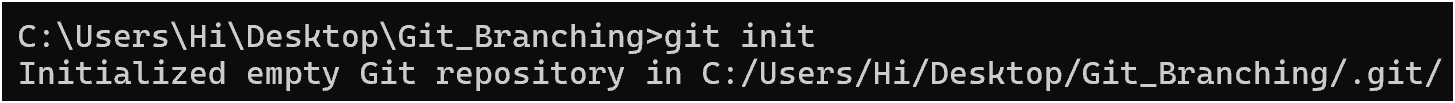


Step 3:

Initialize Git by typing this command:

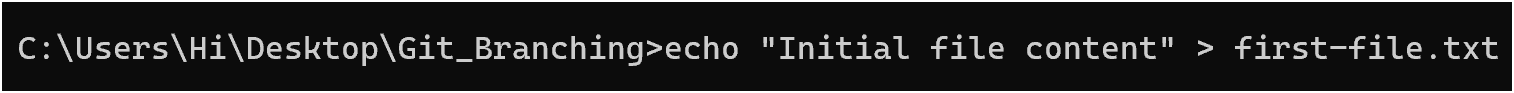
**git init**

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:

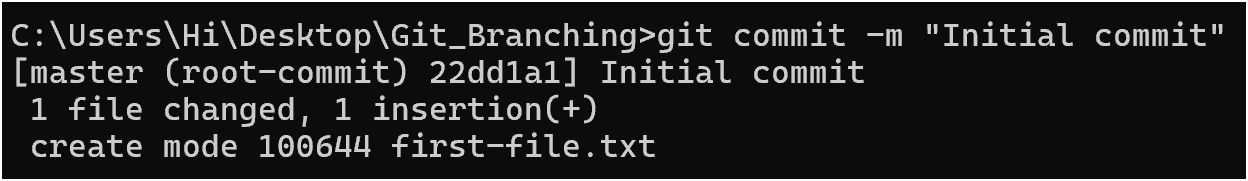
Add the File to Git

Tell Git to track this file:



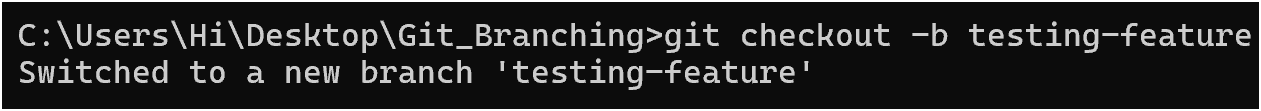
Step 6:

Save this change in Git with a commit message.



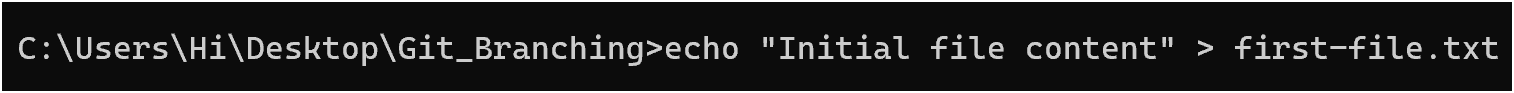
Step 7:

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



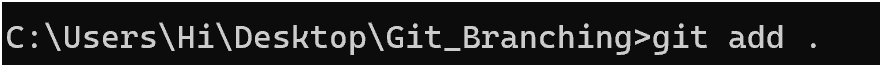
Step 8:

Let’s add a new file for our feature:



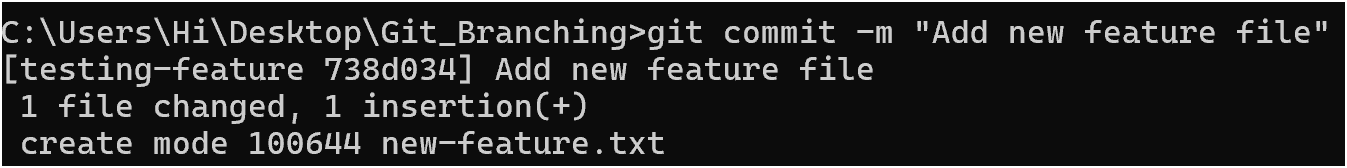
Step 9:

Now, stage the changes:



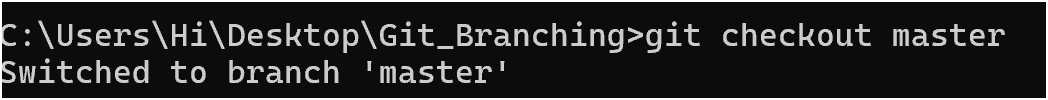
Step 10:

Commit the changes:



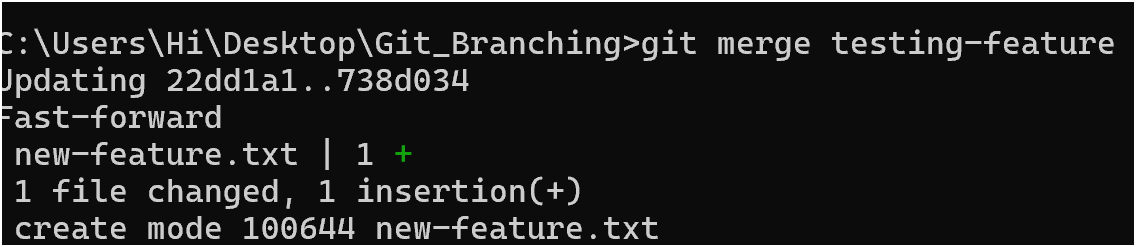
Step 11:

Switch to the master Branch



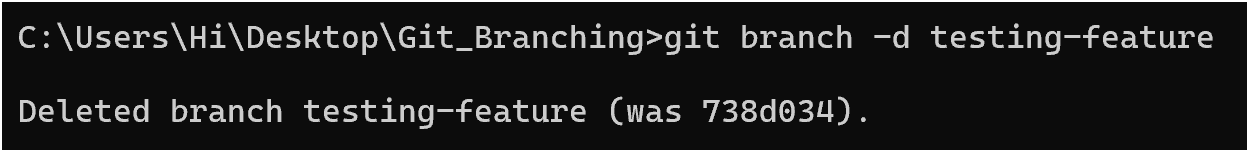
Step 12:

Merge Changes from testing-feature to master



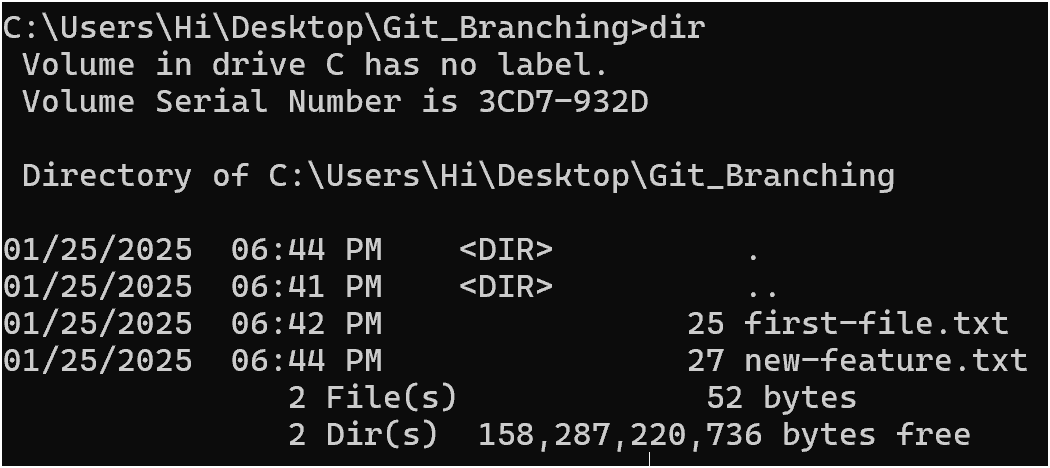
Step 13:

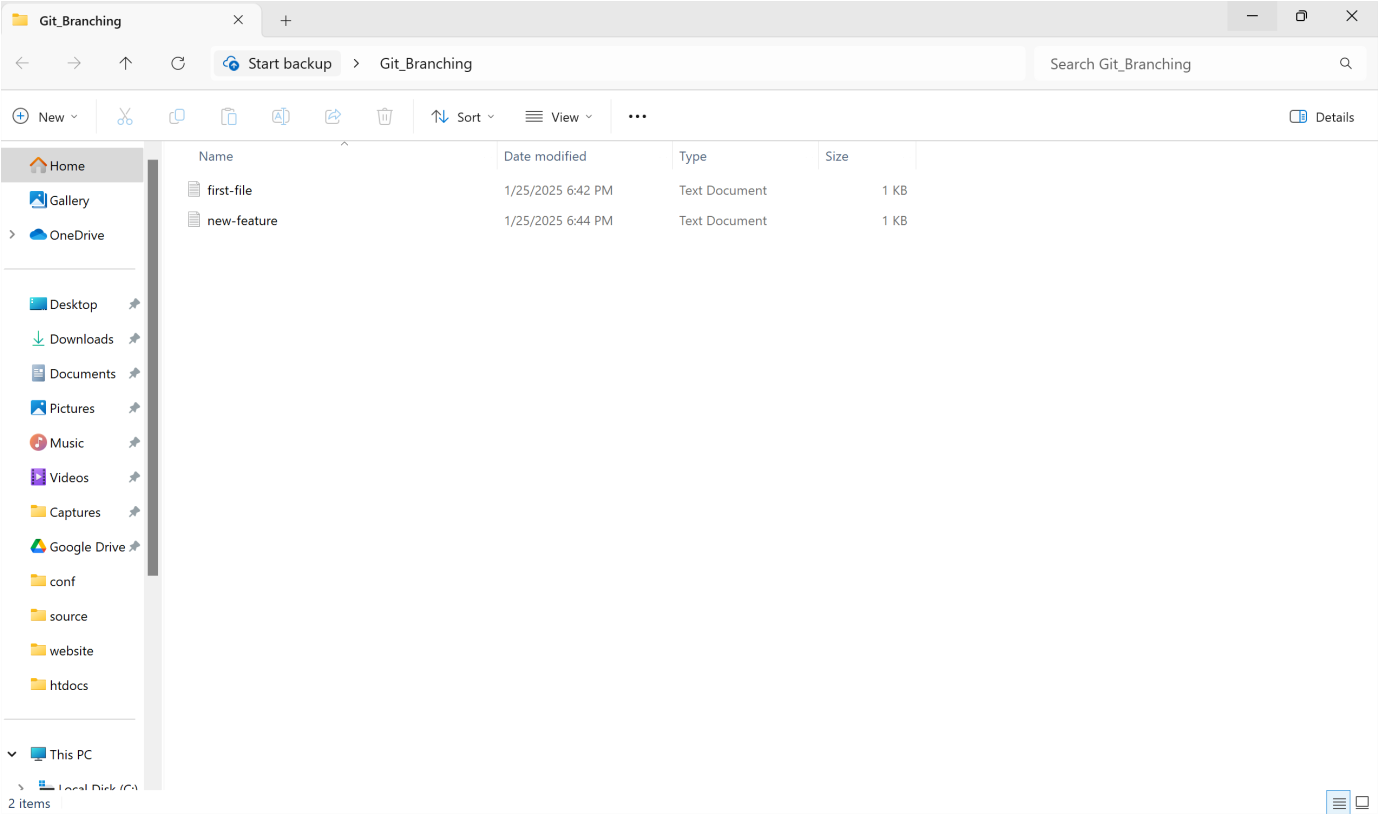
Once the merge is done, you can delete the testing-feature branch:



Step 14:

Now, check the files in the folder:





Outcome

1. Successfully set up a Git repository in your local project directory.

2. Create, switch, and manage multiple branches for feature development and testing.

3. Track modifications and commit changes across different branches.

4. Merge feature branches into the main branch while preserving project stability.

5. Gain practical experience with essential Git commands like git init, git add, git commit, git checkout, and git merge.