

**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: MADHUMETHRA AP

Department: IT



**Introduction**

Git is a powerful version control system that enables developers to track changes, collaborate efficiently, and manage code effectively. One of Git’s essential features is branching, which allows multiple developers to work on different features simultaneously without affecting the main project.

### **Overview**

Understand Git branching and its workflow.

Be able to create and manage branches effectively.

Learn how to merge changes safely into the main branch.

Gain confidence in handling version control for team collaboration.

**Objectives**

The objective of this guide is to help you:

Create a new branch in a Git repository.

Add a new feature to the branch.

Merge the branch back into the main branch.

Understand the importance of using Git branches for version control.

**Importance**

Git branches provide an isolated workspace for new development, allowing developers to test features, fix bugs, and implement enhancements without modifying the stable codebase.

**Step-by-Step Overview**

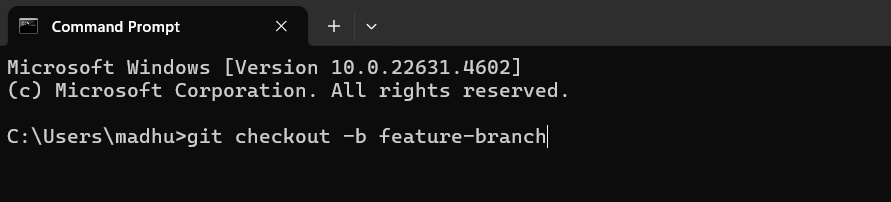
Step 1

\*Create a New Branch\*

First, navigate to your Git repository in the terminal and check out a new branch for your feature:

bash

git checkout -b feature-branch



Step 2

This creates and switches you to a new branch called feature-branch.

Make changes to your code to add the new feature. Once you’ve made the necessary changes, save the files.

Step 3

Stage the Changes After editing your files, stage the changes for commit:

bash

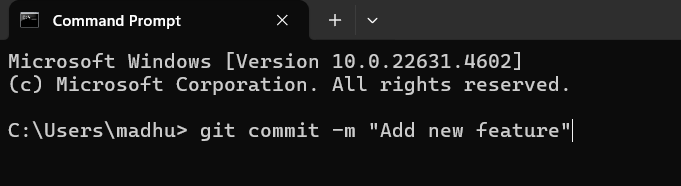
git add .

Step 4

Next, commit the changes with a relevant message:

bash

git commit -m "Add new feature"



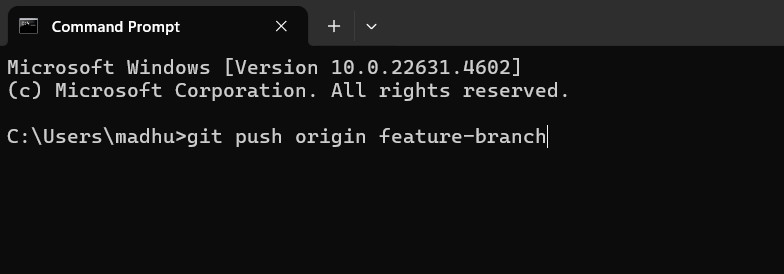
Step 5

Push the New Branch\*

If you’re working with a remote repository (like GitHub or GitLab), push the branch to the remote:

bash

git push origin feature-branch



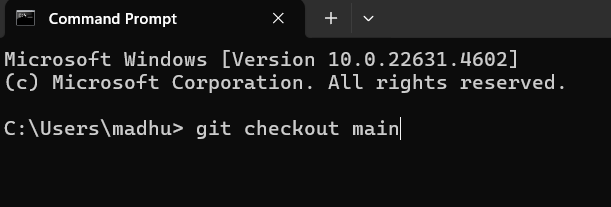
Step 6

Switch Back to the Main Branch

Once you’re done with the feature, switch back to the main branch (or whatever your main branch is called, e.g., main or master):

bash

git checkout main

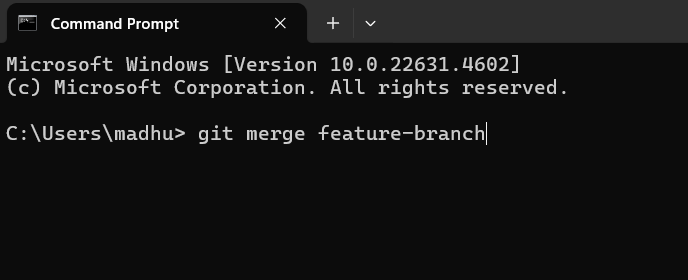


Step 7

Now, merge the new feature branch into the main branch:

bash

git merge feature-branch



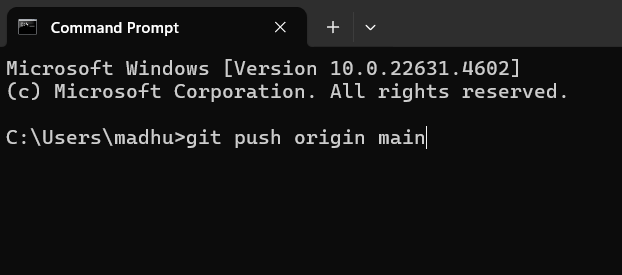
Step 8

Push the Changes to the Remote\*

If everything looks good, push the updated main branch to the remote repository:

bash

git push origin main



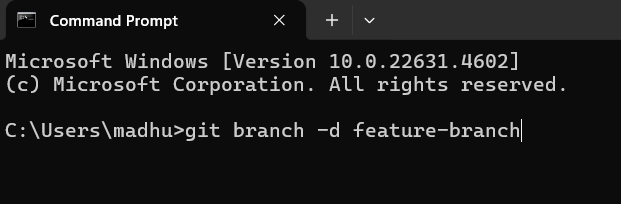
Step 9

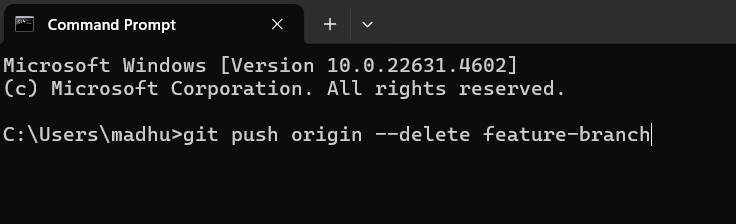
Once the feature is merged, you can delete the feature branch both locally and remotely if no longer needed:

bash

git branch -d feature-branch

git push origin --delete feature-branch





**Output**

1. Created a new branch: Successfully created a new branch in the Git repository for testing.
2. Added a new feature: Implemented a new feature in the testing branch.

3. Merged the branch: Merged the testing branch into the main branch, integrating the new feature.