

# {SOURCE CODE MANAGEMENT CSE2002}

LAB MANUAL

**SUBMITTED BY:**

**SRUJANA.N**

**A866175124026**

**B.TECH IN CSE(AIML) 2<sup>ND</sup> SEMESTER**

**BATCH-07**



SI. NO.	INDEX
▪	Basics of Linux
▪	Introduction-Git Bash
▪	Git Bash And Git Hub
▪	File Creation with commit and push command
▪	Branches Creation
▪	Merge Request
▪	Open and Close Pull Request
▪	Complete Git Process

➤ **EXPERIMENT-01:**

➤ **Basics Of Linux:**

## ✓ **1.What is Linux?**

- ✓ Linux is a family of free and open-source operating systems based on the Linux kernel. Operating systems based on Linux are known as Linux distributions or distros. Examples include Debian, Ubuntu, Fedora, CentOS, Gentoo, Arch Linux, and many others.

## ✓ **2. Linux Distributions (Distros)**

- ✓ Different versions of Linux tailored for various needs:
- ✓ Ubuntu - User-friendly, great for beginners.
- ✓ Debian - Stable and well-tested.
- ✓ Fedora - Latest features, cutting-edge.
- ✓ Arch Linux - Lightweight and customizable.
- ✓ CentOS/RHEL - Used in enterprise environments.

## ✓ **3. Linux File System Structure**

- ✓ Linux follows a hierarchical structure:
- ✓ bin → essential binary programs
- ✓ etc → configuration files
- ✓ home → user directories
- ✓ root → root user's home directory
- ✓ var variable data (logs, etc.)
- ✓ tmp → temporary files
- ✓ usr → user-installed software

## ✓ **4. Basic Linux Commands**

### ✓ **Command Description**

- ✓ ls. - List files in a directory
- ✓ Cd - Change directory
- ✓ Pwd -Print working directory
- ✓ Mkdir. Make a new directory

- ✓ Rm - Remove files/directories
- ✓ Cp. - Copy files/directories
- ✓ Mv. - Move or rename files
- ✓ Cat.- Display file content
- ✓ Sudo. - Execute a command as superuser
- ✓ man
- ✓ Show manual/help for a command

## ✓ **5. File Permissions**

- ✓ Linux controls file access with permissions.
- ✓ Use ls -l to view them (e.g., -rwxr-xr--).
- ✓ Modify with chmod, chown, or chgrp.

## ✓ **6. Package Management**

- ✓ Install/update software using package managers:
- ✓ Debian/Ubuntu: apt (e.g., sudo apt install package-name)
- ✓ Red Hat/Fedora: dnf or yum
- ✓ Arch Linux: pacman

## ✓ **7. Shell and Terminal**

- ✓ The shell interprets commands (e.g., bash, zsh).
- ✓ Terminal is where you type the commands.

## ✓ **8. Users and Permissions**

- ✓ Regular users vs. root (superuser).
- ✓ Manage users with adduser, usermod, and passwd

## ✓ **Experiment-02:**


### ▪ **Introduction -Git Bash**

#### ✓ **What Is Git Bash?**

- ✓ Git Bash is a command-line tool for Windows that provides:
- ✓ Git command-line tools
- ✓ A Bash (Unix-style) shell

- ✓ . A way to run shell commands similar to those used in Linux/macOS
- ✓ It's especially useful for developers using Git on Windows who want a Unix-like experience.
- ✓ **Key Features:**
  - **Bash Emulation:**
- ✓ Bash (Bourne Again SHell) lets you run Linux-style commands (e.g., ls, pwd, rm).
- ✓ This makes it easier to follow tutorials written for Linux/macOS.

- **Git Integration:**

- ✓  You can run Git commands like git clone, git status, git commit directly in Git Bash.
- ✓ Useful for version control and managing code repositories.

- **Cross-Platform Compatibility:**

- ✓ Lets Windows users interact with remote Linux servers or repositories more easily.
- ✓ How to Install Git Bash:
- ✓ Go to <https://git-scm.com>
- ✓ Download the installer for your OS (choose Windows).
- ✓ Run the installer and select "Git Bash" when prompted about default terminal.
- ✓ After installation, right-click anywhere and choose "Git Bash Here" to open the terminal.
- ✓ **Common Commands in Git Bash:**

Command.	Description
Ls.	List files and directories
Pwd	Print current directory path
Cd.	Change directory
Mkdir myfolder	Create a new folder
rm filename	Remove a file
git init	Initialize a new Git repositor

Command.	Description
git clone URL.	Clone a repository from GitHub
git status.	Show current Git status

git add.                      Stage all changes

git commit -m "message" Commit staged changes

✓ **Use Cases:**

- ✓ Managing Git repositories
- ✓ Running shell scripts
- ✓ Navigating your project with Unix-style commands
- ✓ Automating development tasks

✓ **Experiment-03:**

✓ **Git Bash And GitHub**

✓ **What is Git?**

- ✓ Before connecting GitHub and Git Bash, it's important to know:
- ✓ Git is a version control system to track changes in code.
- ✓ GitHub is a hosting service for Git repositories.
- ✓ ? Git Bash lets you use Git on Windows with Linux-style
- ✓ commands.

✓ **☞How They Work Together**

✓ **Here's the typical workflow:**

- ✓ You install Git and Git Bash on your Windows
- ✓ machine.
- ✓ You create or clone a Git repository using Git Bash.
- ✓ You make changes to your files locally.
- ✓ You use Git commands in Git Bash to:
  - stage changes: git add
  - commit changes: git commit
  - push changes: git push (to GitHub)

**You use GitHub to:**

- ✓ store your code in the cloud

- ✓ collaborate with others
- ✓ view project history, pull requests, issues, etc.

### Example: Basic Workflow

#### Step-by-step using Git Bash & GitHub:

- ✓ Clone a GitHub repo:
- ✓ Make changes to our project files.
- ✓ Stage and commit changes:

```
sruja@SRUJANA7795 MINGW64 ~/SHRU (master)
$ ls
4026/ SHRU.txt web.html

sruja@SRUJANA7795 MINGW64 ~/SHRU (master)
$ vi web.html

sruja@SRUJANA7795 MINGW64 ~/SHRU (master)
$ git add .
warning: in the working copy of 'web.html', LF will be replaced by CRLF the next time Git touches it
warning: adding embedded git repository: 4026
hint: You've added another git repository inside your current repository.
hint: Clones of the outer repository will not contain the contents of
hint: the embedded repository and will not know how to obtain it.
hint: If you meant to add a submodule, use:
hint:   git submodule add <url> 4026
hint: If you added this path by mistake, you can remove it from the
hint: index with:
hint:   git rm --cached 4026
hint: See "git help submodule" for more information.
hint: Disable this message with "git config set advice.addEmbeddedRepo false"
warning: in the working copy of 'SHRU.txt', LF will be replaced by CRLF the next time Git touches it

sruja@SRUJANA7795 MINGW64 ~/SHRU (master)
$ git commit -m "i changed a padding"
[master a2ffe9e] i changed a padding
3 files changed, 3 insertions(+), 1 deletion(-)
create mode 160000 4026
create mode 100644 SHRU.txt

sruja@SRUJANA7795 MINGW64 ~/SHRU (master)
$
```

 MINGW64:/c/Users/sruja/SHRU

```
sruja@SRUJANA7795 MINGW64 ~ (master)
$ git clone https://github.com/Srujana433/SHRU.git
fatal: destination path 'SHRU' already exists and is not an empty directory.

sruja@SRUJANA7795 MINGW64 ~ (master)
$ cd SHRU

sruja@SRUJANA7795 MINGW64 ~/SHRU (master)
$
```

- ✓ Push changes to GitHub:

```
sruja@SRUJANA7795 MINGW64 ~/SHRU (master)
$ git push -u origin
Enumerating objects: 6, done.
Counting objects: 100% (6/6), done.
Delta compression using up to 12 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (4/4), 355 bytes | 177.00 KiB/s, done.
Total 4 (delta 1), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/Srujana433/4026.git
    bc89089..a2ffe9e  master -> master
branch 'master' set up to track 'origin/master'.
```

- ✓ 

```
sruja@SRUJANA7795 MINGW64 ~/SHRU (master)
$
```

## Experiment-04:

### File Creation With Commit And Push


#### Command.

❓ To Create a file, commit it and push it to a remote repository using Git Bash

#### 1. Create a File.

- ✓ Open Git Bash and create a folder :
- ✓ Using cd and mkdir create a folder:



 MINGW64:/c/folder

```
sruja@SRUJANA7795 MINGW64 ~ (master)
```

```
$ cd c:
```

```
sruja@SRUJANA7795 MINGW64 /c
```

```
$ mkdir folder
```

```
sruja@SRUJANA7795 MINGW64 /c
```

```
$ cd folder
```

```
sruja@SRUJANA7795 MINGW64 /c/folder
```

```
$ |
```

- ✓ Then git init command to create a new Git repository

```
sruja@SRUJANA7795 MINGW64 /c/folder
```

```
$ git init
```

```
Initialized empty Git repository in C:/folder/.git/
```

```
sruja@SRUJANA7795 MINGW64 /c/folder (master)
```

```
$
```

- ✓ Create a new file using vi command in Linux is used to open

and edit files using the vi editor, a powerful text editor available on most Unix-based systems:

```
sruja@SRUJANA7795 MINGW64 /c/folder (master)
```

```
$ vi file.txt
```

```
sruja@SRUJANA7795 MINGW64 /c/folder (master)
```

```
␣
```

```
sruja@SRUJANA7795 MINGW64 /c/folder (master)
$ git add file.txt
warning: in the working copy of 'file.txt', LF will be replaced by CRLF the next time Git touches it
sruja@SRUJANA7795 MINGW64 /c/folder (master)
$
```

O To add all files in the directory:

```
sruja@SRUJANA7795 MINGW64 /c/folder (master)
$ git add .
```

```
sruja@SRUJANA7795 MINGW64 /c/folder (master)
$
```

### 3.Commit the File:

O Commit the changes with a message.

```
sruja@SRUJANA7795 MINGW64 /c/folder (master)
$ git commit -m"i written my name"
[master (root-commit) 69b082b] i written my name
1 file changed, 2 insertions(+)
create mode 100644 file.txt
```

```
sruja@SRUJANA7795 MINGW64 /c/folder (master)
$ |
```

### 4.Add and push to remote repository:

Push the changes to Git Hub (assuming origin is the remote and Main is the branch.)

```
sruja@SRUJANA7795 MINGW64 /c/folder (master)
$ git pull --rebase folder master
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 3 (delta 0), pack-reused 0 (from 0)
Unpacking objects: 100% (3/3), 212 bytes | 9.00 KiB/s, done.
From https://github.com/Srujana433/C1
* branch          master      -> FETCH_HEAD
* [new branch]     master      -> folder/master
Successfully rebased and updated refs/heads/master.
```

```
sruja@SRUJANA7795 MINGW64 /c/folder (master)
$ git push -u folder master
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 12 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 314 bytes | 157.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/Srujana433/C1.git
  9578550..321b277  master -> master
branch 'master' set up to track 'folder/master'.
```

## Experiment-05:

### Branches Creation

In Git branches allow developers to work on different

Features or fixes without affecting the main

Codebase. Here's how we can create and manage branches.

#### ○ Check the branches:

Check the branches using git branch command:

```
sruja@SRUJANA7795 MINGW64 /c/folder (master)
$ git branch
* master
```

To create new branch ;

```
sruja@SRUJANA7795 MINGW64 /c/folder (master)
$ git branch 123
```

### O Switching to a Branch:

To switch to the newly created branch:

```
sruja@SRUJANA7795 MINGW64 /c/folder (master)
$ git checkout 123
Switched to branch '123'
```

```
sruja@SRUJANA7795 MINGW64 /c/folder (123)
$ git branch
* 123
  master
```

### Experiment-06:

#### Merge Request

O A Merge Request(MR) is a feature used in Gitbased platforms like GitLab to propose and review Changes before merging them into the main Branch.It is similar to a Pull Request(PR) in GitHub.

#### How Merge Request Works:

##### 1. Create a Branch

A developer creates a separate branch from the main

Branch (often called main or master) to work on a specific Feature, bug fix, or enhancement.

## **2. Make Changes**

The developer writes code, commits changes to their Branch, and pushes the branch to the remote repository.

## **3. Open a Merge Request**

In the Git platform (e.g., GitLab, GitHub):

- ☐ The developer opens a merge request from their Feature branch into the target branch (usually main).
- ☐ They add a description, possibly link related issues, And assign reviewers.

## **4. Code Review**

- ☐ Reviewers (team members or maintainers) review The code.
- ☐ Automated checks (e.g., tests, linters) are run via CI/CD pipelines.
- ☐ Feedback might be given; the author can make Additional commits to address it.

## **6. Merge**

The MR is merged into the target branch:

- ☐ Options include merge, squash and merge, or Rebase and merge.
- ☐ Once merged, the feature branch can be deleted if no Longer needed.

**Steps to Merge the Branch:**

1.Switch to the main branch (master):

```
sruja@SRUJANA7795 MINGW64 /c/folder (123)
$ git checkout master
Switched to branch 'master'
Your branch is up to date with 'folder/master'.
```

```
sruja@SRUJANA7795 MINGW64 /c/folder (master)
$ |
```

## 2.Merge the branch:

Using git mergetool and git merge main command:

```
sruja@SRUJANA7795 MINGW64 /c/folder (master)
$ git mergetool
```

```
This message is displayed because 'merge.tool' is not configured.
See 'git mergetool --tool-help' or 'git help config' for more details.
'git mergetool' will now attempt to use one of the following tools:
opendiff kdiff3 tkdiff xxdiff meld tortoisemerge gvimdiff diffuse diffmerge ecmerge p4merge araxis bc codecompare smerge emerge vimdiff nvimdiff
No files need merging
```

```
sruja@SRUJANA7795 MINGW64 /c/folder (master)
$ git merge test
fatal: refusing to merge unrelated histories
```

Now Press enter to edit the files and windows are opening

, then remove some red lines and add some lines:



To remove merging commit one line:

```
sruja@SRUJANA7795 MINGW64 /c/folder (master)
$ git commit -m"i writen by name"
On branch master
Your branch is up to date with 'folder/master'.

nothing to commit, working tree clean
```

Using the command `graph` we can see the graph of

Commits:

```
sruja@SRUJANA7795 MINGW64 /c/folder (master)
$ git log --oneline
321b277 (HEAD -> master, folder/master, folder/HEAD, 123) i written my name
9578550 test
```

```
sruja@SRUJANA7795 MINGW64 /c/folder (master)
```



Benefits:

- ❑ Facilitates code review
- ❑ Triggers automated tests
- ❑ Maintains a clear change history
- ❑ Encourages collaborative development

## **Experiment-07:**

### **❑ Open and Close Pull Request**

#### **1. Open a Pull Request**

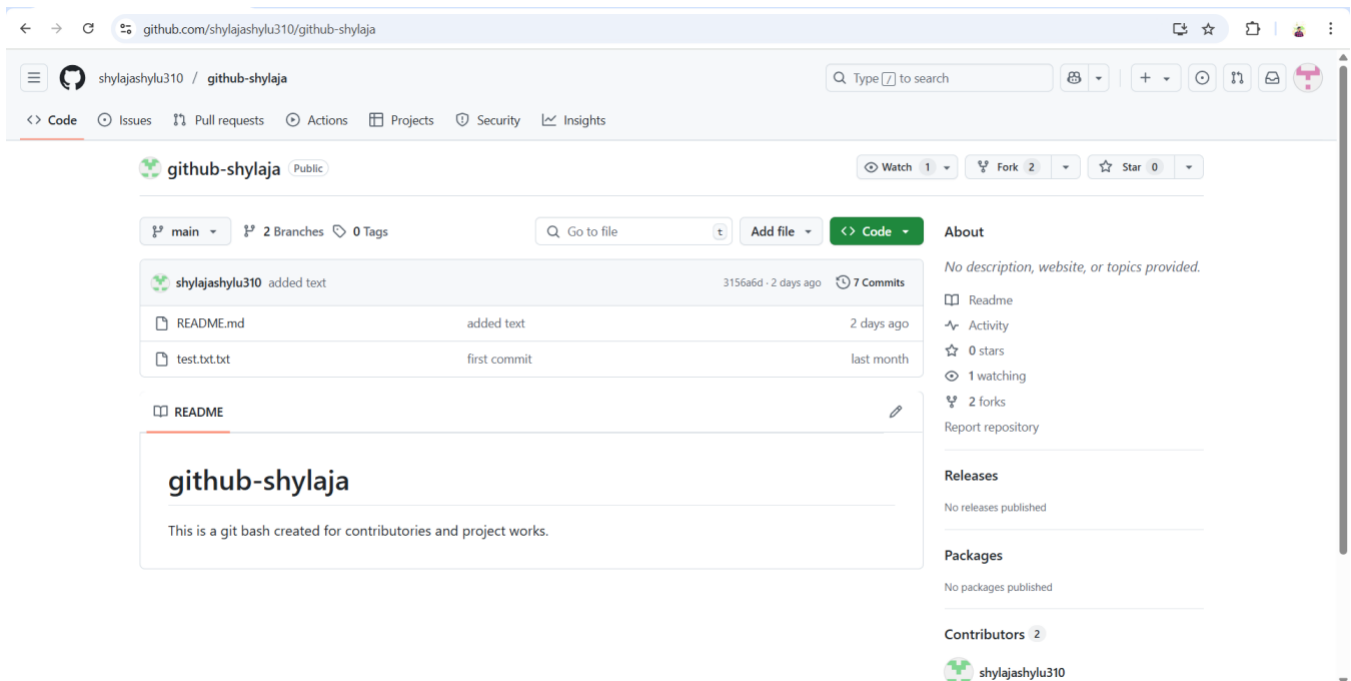
1. Push your changes to a branch on your fork or the same repository.
2. Go to GitHub, navigate to the repository.
3. You'll see a "Compare & pull request" button — click it.
4. Add a title and description for your PR.
5. Click "Create pull request".

#### **2. Close a Pull Request**

1. Click "Merge pull request".
2. Confirm by clicking "Confirm merge".
3. Optionally, delete the branch.

In the git hub account select the user which whom you

Want to merge and select the repo and fork it:



Now copy the link and using git clone command open that  
File:

```
sruja@SRUJANA7795 MINGW64 /c/folder (master)
$ git clone https://github.com/shylajashylu310/github-shylaja.git
Cloning into 'github-shylaja'...
remote: Enumerating objects: 17, done.
remote: Counting objects: 100% (17/17), done.
remote: Compressing objects: 100% (14/14), done.
remote: Total 17 (delta 1), reused 5 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (17/17), 4.94 KiB | 843.00 KiB/s, done.
Resolving deltas: 100% (1/1), done.
```

```

sruja@SRUJANA7795 MINGW64 /c
$ git commit -m"i witten something"
fatal: not a git repository (or any of the parent directories): .git

sruja@SRUJANA7795 MINGW64 /c
$ cd git-folder

sruja@SRUJANA7795 MINGW64 /c/git-folder (main)
$ ls
README.md

sruja@SRUJANA7795 MINGW64 /c/git-folder (main)
$ vi README.md

sruja@SRUJANA7795 MINGW64 /c/git-folder (main)
$ git add .
warning: in the working copy of 'README.md', LF will be replaced by CRLF the next time Git touches it

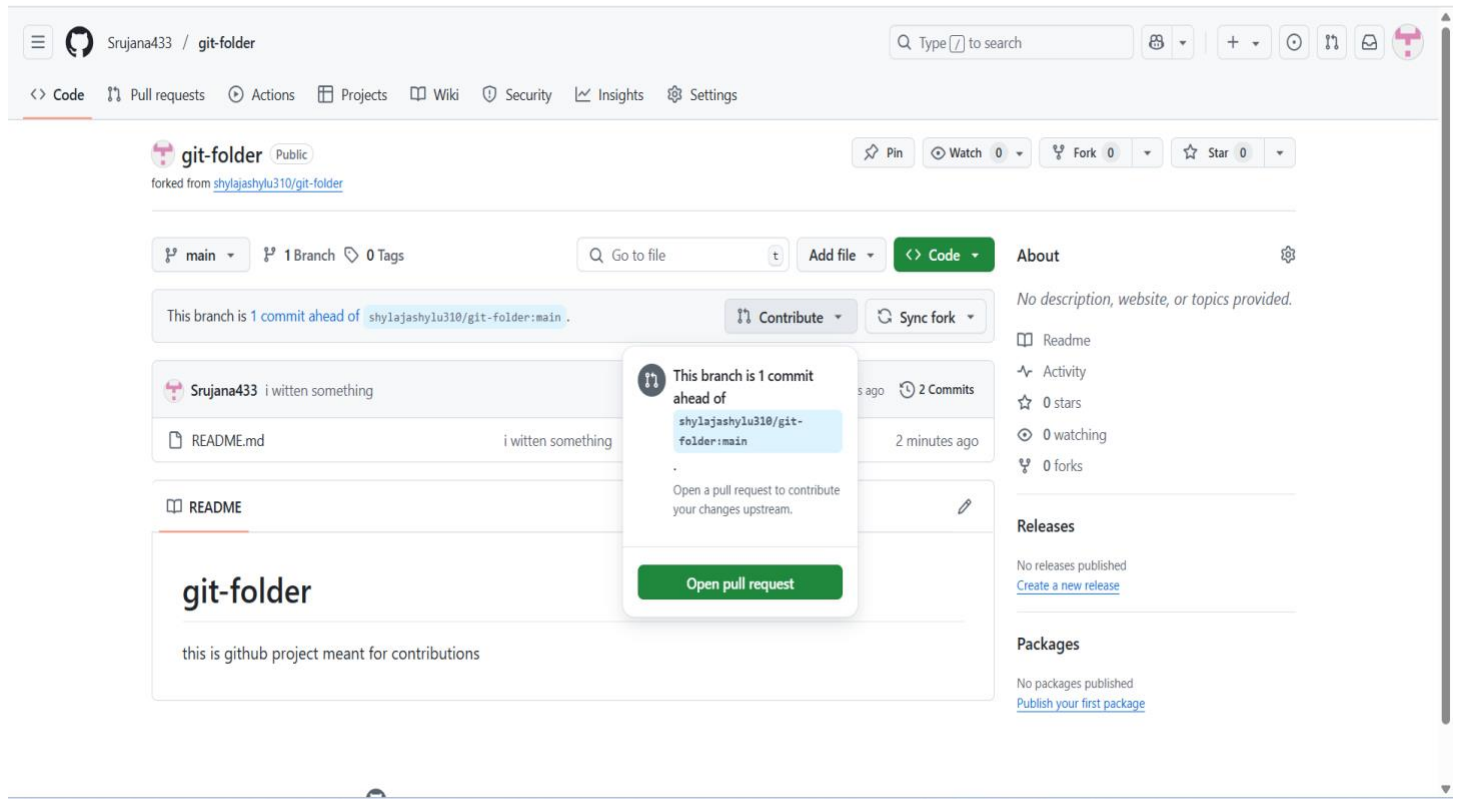
sruja@SRUJANA7795 MINGW64 /c/git-folder (main)
$ git commit -m"i witten something"
[main 2bad205] i witten something
1 file changed, 3 insertions(+), 1 deletion(-)

sruja@SRUJANA7795 MINGW64 /c/git-folder (main)
$ git remote
origin

sruja@SRUJANA7795 MINGW64 /c/git-folder (main)
$ git push origin main
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 12 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 305 bytes | 152.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/Srujana433/git-folder.git
ca214d5..2bad205 main -> main

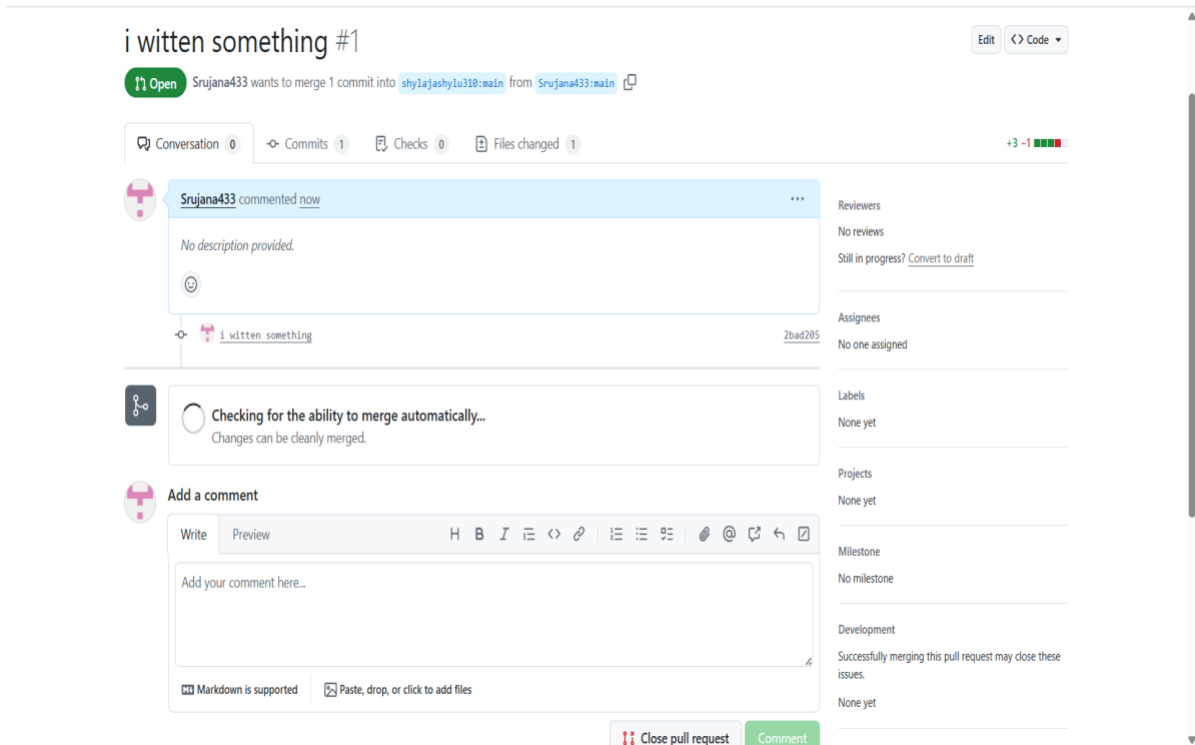
```

In the Git Hub account contribute to open the pull request:



## Steps to Close a Pull Request on GitHub:

1. Go to the repository on GitHub.
2. Click on the “Pull requests” tab
3. Find the pull request you want to close and click on  
It.
4. Scroll to the bottom of the PR page.
5. Click the “Close pull request” button.



## Experiment 08:

### ? Complete Git Process.

#### 1. Install Git Bash

- Download Git for Windows from **Git's official Website**.

- Run the installer and follow the setup instructions.

- Choose Git Bash as the default terminal option.

#### 2. Initialize a Repository

- ? Open Git Bash and navigation to our project folder.

```

sruja@SRUJANA7795 MINGW64 ~ (master)
$ pwd
/c/users/sruja

sruja@SRUJANA7795 MINGW64 ~ (master)
$ ls
3apr1/      Documents/
4026/       Downloads/
4027/       Favorites/
AppData/    Jana/
'Application Data'  L-35/
CI/         Lab/
Cloned/     'Links'
contacts/   'Local Settings'
cookies/    'My Documents'
CrossDevice/ NTUSER.DAT

NTUSER.DAT{a12d501b-29d8-11ef-943d-58cd9938ebe}.TM.btf
NTUSER.DAT{a12d501b-29d8-11ef-943d-58cd9938ebe}.TMCcontainer00000000000000000001.regtrans-ms
NTUSER.DAT{a12d501b-29d8-11ef-943d-58cd9938ebe}.TMCcontainer000000000000000000002.regtrans-ms
NetHood/
'New Folder'/
OneDrive/
Pr-inHood/
Pycharmp/
Recent/
SRU4026/

'Saved Games'/
Searches/
Sendto/
Shylaja/
'Start Menu'/
Task-Titans/
Task-Titans-/
Teaplaties/
Titans/
Untitled-1.py

c/
clean/
clean_4210/
d/
file/
from/
git/
helloworld.py
interspers/
ntuser.dat.LOG1

ntuser.dat.LOG2
ntuser.ini
p.txt
project/
scm6/
scm_labl/
scum/
scum_lab2/
scum_lab3/
shreya/

shru/
sruja/
string.py
task/
tasktitans/
test/
yashaswini-4210/

```

## ❓ Initialize a new git repository

```
sruja@SRUJANA7795 MINGW64 /c/123
$ git init
Initialized empty Git repository in C:/123/.git/
```

### 3.Create and Modify Files

❓ Create a file using vi command:

```
sruja@SRUJANA7795 MINGW64 /c/123 (master)
$ vi 123.txt
```

#### 4. Check Repository Status

[View changes](#)

```
sruja@SRUJANA7795 MINGW64 /c/123 (master)
$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    123.txt

nothing added to commit but untracked files present (use "git add" to track)
```

## 6. Stage and Commit Changes

- Add files to the staging area:
- Commit changes:

```
+ .....  
sruja@SRUJANA7795 MINGW64 /c/123 (master)  
$ git add .  
warning: in the working copy of '123.txt', LF will be replaced by CRLF the next time Git touches it  
warning: in the working copy of '765.txt', LF will be replaced by CRLF the next time Git touches it  
  
sruja@SRUJANA7795 MINGW64 /c/123 (master)  
$ git commit -m "i written something"  
[master (root-commit) df3be85] i written something  
2 files changed, 2 insertions(+)  
create mode 100644 123.txt  
create mode 100644 765.txt
```

## 6. Create and Manage Branches

- Create a new branch:

```
sruja@SRUJANA7795 MINGW64 /c/123 (master)  
$ git branch  
* master
```

```
sruja@SRUJANA7795 MINGW64 /c/123 (master)  
$ git branch test
```

- Switch to the branch:

```
sruja@SRUJANA7795 MINGW64 /c/123 (master)  
$ git checkout test  
Switched to branch 'test'
```

```
sruja@SRUJANA7795 MINGW64 /c/123 (test)  
$
```

---

- Merge the branch into the main branch:

```
sruja@SRUJANA7795 MINGW64 /c/folder (master)
$ git log --oneline
321b277 (HEAD -> master, folder/master, folder/HEAD, 123) i written my name
9578550 test
```

```
sruja@SRUJANA7795 MINGW64 /c/folder (master)
$
```

```
sruja@SRUJANA7795 MINGW64 /c/folder (master)
$ git mergetool --tool=codecompare
No files need merging
```

```
sruja@SRUJANA7795 MINGW64 /c/folder (master)
$ git merge test
merge: test - not something we can merge
```

## 7. Git Clone:

```
sruja@SRUJANA7795 MINGW64 /c/123 (test)
$ git clone https://github.com/Srujana433/git-folder.git
Cloning into 'git-folder'...
remote: Enumerating objects: 6, done.
remote: Counting objects: 100% (6/6), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 6 (delta 0), reused 3 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (6/6), done.
```

- Update local repository:



## 8. Pull Changes from Remote Repository

```
sruja@SRUJANA7795 MINGW64 ~/SHRU (master)
$ git push -u origin
Enumerating objects: 6, done.
Counting objects: 100% (6/6), done.
Delta compression using up to 12 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (4/4), 355 bytes | 177.00 KiB/s, done.
Total 4 (delta 1), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/Srujana433/4026.git
   bc89089..a2ffe9e  master -> master
branch 'master' set up to track 'origin/master'.

sruja@SRUJANA7795 MINGW64 ~/SHRU (master)
$
```

## 10. Open and Close Pull Requests

- Open a Pull Request on GitHub and merge changes.
- Close the Pull Request if needed.

The screenshot shows a GitHub Pull Request (PR) page for the repository 'shylajashylu310/git-folder'. The PR is titled 'i witten something #1' and is currently open. It shows a merge from 'Srujana433:main' to 'shylajashylu310:main'. The PR has 1 commit, 0 checks, and 1 file changed. A comment by 'Srujana433' is visible, stating 'No description provided.' The PR is currently in progress, with a status 'Checking for the ability to merge automatically...' and a message 'Changes can be cleanly merged.' The right sidebar shows various metadata: Reviewers (No reviews), Assignees (No one assigned), Labels (None yet), Projects (None yet), Milestone (No milestone), and Development (Successfully merging this pull request may close these issues). At the bottom, there are buttons for 'Close pull request' and 'Comment'.