AMITY UNIVERSITY

SOURCE CODE MANAGEMENT LABORATORY RECORD

Name : Vikas D

Enrollment No. : A866175124018

Program : B tech CSE (AIML)

Submitted to : Dr Monit Kapoor

Sl No. **Index**

Lab Exercise 1 :

Introduction to Gitbash

Git Bash

- Git Bash is an application for Microsoft Windows that provides a command-line interface to use Git, the version control system.

- It emulates a Bash (Bourne Again Shell) environment, allowing users to run Linux-style commands on Windows.

- It is mainly used by developers to:

1. Execute Git commands (e.g., git init, git commit)
2. Manage code repositories
3. Interact with remote repositories

-

  Key Features:

1. Supports Git version control operations.
2. Provides Unix-style command-line tools (like ssh, scp, ls, etc.).
3. Helps users practice command-line Git workflows.

- **Steps to Install Git Bash:**

1. Go to the official Git website:

- https://git-scm.com

2. Download Git for Windows:

- Click the “Download for Windows” button.
- The .exe file will start downloading.

•

3. Run the Installer:

Double-click the downloaded .exe file.

4. Follow the Setup Wizard:

- Click Next on the welcome screen.
- Choose the default options (recommended for beginners).
- Select the text editor (e.g., Notepad or VS Code).
- Choose “Git from the command line and also from 3rd-party software”.
- Continue clicking Next until you reach Install.

5. Click Install:

- Wait for the installation to complete.

6. Finish Setup:

- Click Finish and leave “Launch Git Bash” checked.

7. Start Using Git Bash:

-
- Git Bash will open in a terminal window.

Basic Git Commands in Git Bash:

1. Git init

Initializes a new Git repository in your folder.

2. Git clone <repo-url>

Copies (clones) a remote repository to your local machine.

3. Git status

Shows the status of files (tracked, modified, staged).

4. Git add <filename>

Adds a specific file to the staging area.

Use git add . to add all files.

•

5. **Git commit -m “Your message”**

Saves the staged changes with a message.

6. <u>Git push</u>

Uploads your commits to the remote repository (GitHub).

7. <u>Git pull</u>

Downloads changes from the remote repository and merges them.

8. <u>Git remote add origin</u> <repo-url>

Connects your local repo to a GitHub repository.

9. <u>Git log</u>

Shows the commit history.

10. <u>Git branch</u>

Lists all branches.

Use git branch <name> to create a new branch.

11. <u>Git checkout <branch-name></u>

Switches to a different branch.

12. <u>Git merge <branch-name></u>

Merges changes from one branch into the current one.

Lab Exercise : 2

Introduction to GitHub

Git Hub.

- GitHub is a web-based platform for hosting and sharing Git repositories.

- It is widely used for collaboration, version control, and open-source development.

- GitHub allows users to:

1. Store code in repositories
2. Track changes using Git
3. Collaborate with others through pull requests and issues
4. Manage projects with built-in tools like GitHub Projects and Actions

- Key Features:

1. Cloud-based hosting for Git repositories
2. Social coding features (followers, stars, forks)
3. Integration with CI/CD, project management, and automation tools
4. Access control and team collaboration

- **Steps to Install GitHub Desktop:**

1. Go to: https://desktop.github.com

2. Click “Download for Windows” (or Mac).

3. Open the downloaded file and run the installer.

4. Follow the setup wizard.

5. After installation, open GitHub Desktop.

6. Sign in with your GitHub account.

7. You can now clone repositories, make commits, and push changes using a user-friendly interface.

Lab Exercise 3 :

Gitbash and GitHub

Git.

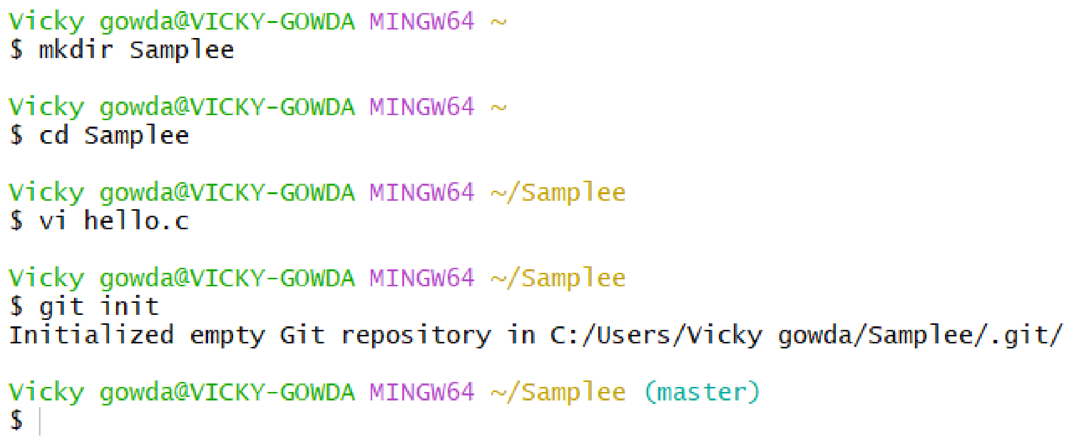
Git is a version control system that helps developers track and manage changes to code over time. It allows multiple people to work on a project at the same time without overwriting each other's work.

- **Key Features of Git:**

1. Version Tracking: Keeps a history of changes made to files.
2. Branching: Lets you work on new features or fixes in isolation.
3. Merging: Combines changes from different branches.
4. Collaboration: Enables teams to work together using services like GitHub, GitLab, or Bitbucket.

- **Step-by-Step Workflow :**

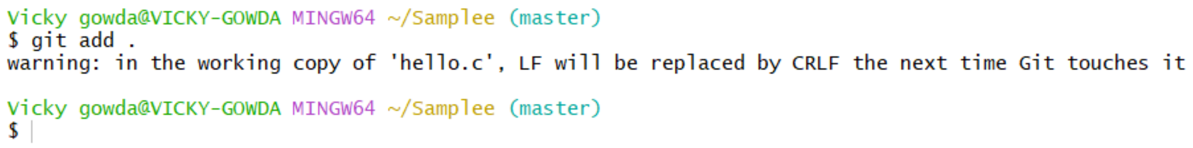
Step 1: Initialize Git



```
Vicky gowda@VICKY-GOWDA MINGW64 ~
$ mkdir Samplee

Vicky gowda@VICKY-GOWDA MINGW64 ~
$ cd Samplee

Vicky gowda@VICKY-GOWDA MINGW64 ~/Samplee
$ vi hello.c

Vicky gowda@VICKY-GOWDA MINGW64 ~/Samplee
$ git init
Initialized empty Git repository in C:/Users/Vicky gowda/Samplee/.git/

Vicky gowda@VICKY-GOWDA MINGW64 ~/Samplee (master)
$
```

- Creates a new Git repository in your project folder.
- Git starts tracking changes to files in this folder.

<u>Step 2:</u> Add Files to Staging

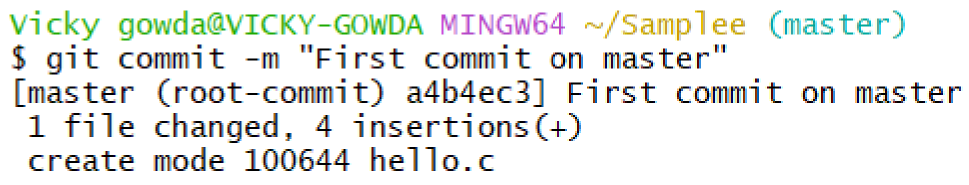


```
Vicky gowda@VICKY-GOWDA MINGW64 ~/Samplee (master)
$ git add .
warning: in the working copy of 'hello.c', LF will be replaced by CRLF the next time Git touches it

Vicky gowda@VICKY-GOWDA MINGW64 ~/Samplee (master)
$ 
```

- Adds all files to the staging area (preparing them to be committed).
- You can also use git add filename to add specific files.

Step 3: Commit Changes

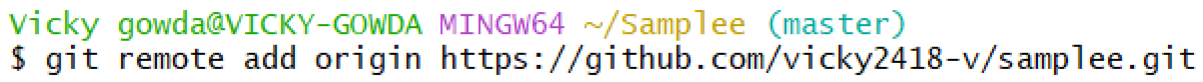
- Records your changes in Git history with a message.



```
Vicky gowda@VICKY-GOWDA MINGW64 ~/Samplee (master)
$ git commit -m "First commit on master"
[master (root-commit) a4b4ec3] First commit on master
 1 file changed, 4 insertions(+)
 create mode 100644 hello.c
```

- This saves your code locally (not yet on GitHub).

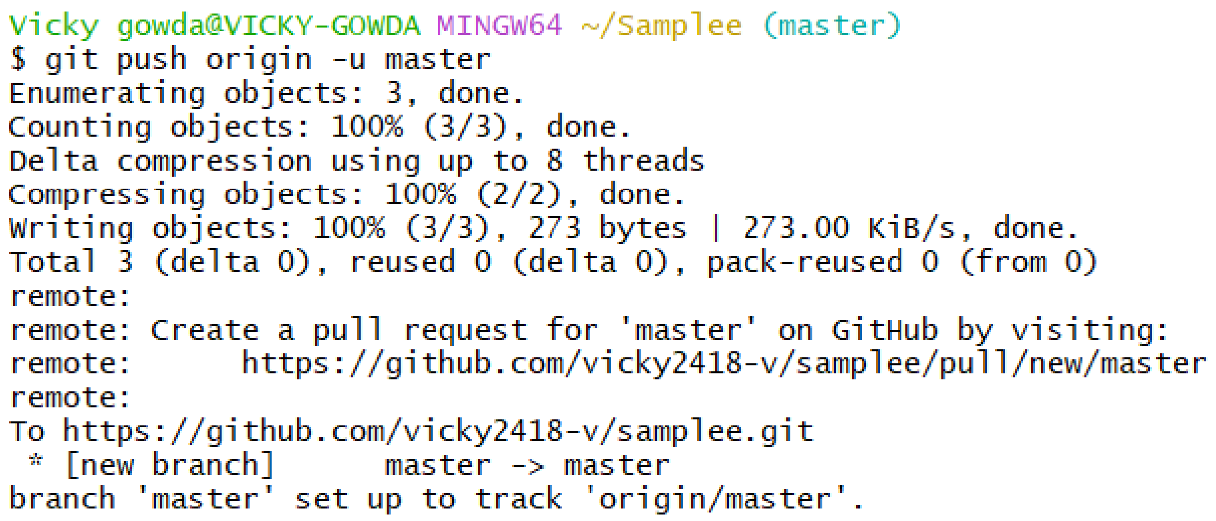
Step 4: Link to GitHub Repository



```
Vicky gowda@VICKY-GOWDA MINGW64 ~/Samplee (master)
$ git remote add origin https://github.com/vicky2418-v/samplee.git
```

- Connects your local Git project to a remote GitHub repository.

Step 5: Push to GitHub


```
Vicky gowda@VICKY-GOWDA MINGW64 ~/Samplee (master)
$ git push origin -u master
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Delta compression using up to 8 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 273 bytes | 273.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
remote:
remote: Create a pull request for 'master' on GitHub by visiting:
remote:      https://github.com/vicky2418-v/samplee/pull/new/master
remote:
To https://github.com/vicky2418-v/samplee.git
 * [new branch]      master -> master
branch 'master' set up to track 'origin/master'.
```

- Uploads your committed code to GitHub.

- After this, your project appears online in the linked GitHub repo.

Lab Exercise 4 :

File Creation with commit and push command

PART 1: Create the GitHub repository

1. Go to GitHub (https://github.com) and sign in.

2. Click “New repository”.

3. Enter a repository name (e.g., my-repo).

4. Choose Public or Private.

5. Don’t add README, .gitignore, or license here — we will push an existing repo.

6. Click “Create repository”.

7. Copy the HTTPS URL shown (e.g., https://github.com/yourusername/my-repo.git).

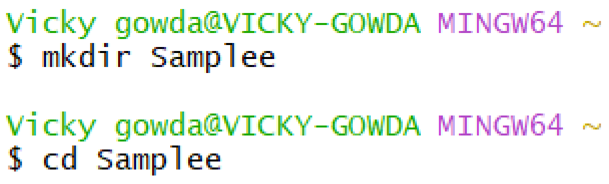
This URL is where you will push your code.

PART 2: Using Git Bash

Step 1: Open Git Bash

This is your command line tool for running Git commands.

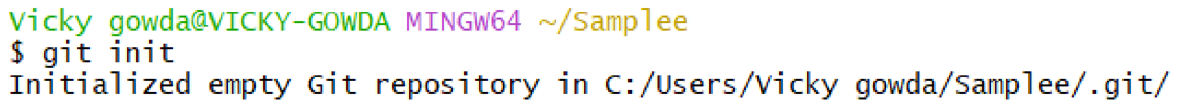
Step 2: Create a new local folder and navigate into it



```
Vicky gowda@VICKY-GOWDA MINGW64 ~
$ mkdir Samplee

Vicky gowda@VICKY-GOWDA MINGW64 ~
$ cd Samplee
```

- Mkdir my-repo: makes a new directory (folder) called my-repo.
- Cd my-repo: changes into that folder so your commands affect this directory.

Step 3: Initialize Git in this folder

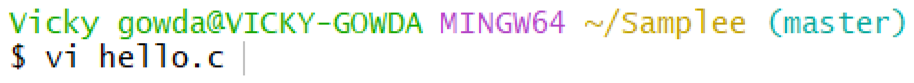


```
Vicky gowda@VICKY-GOWDA MINGW64 ~/Samplee
$ git init
Initialized empty Git repository in C:/Users/Vicky gowda/Samplee/.git/
```

- This creates a hidden .git folder that tracks changes to files here.

- Now this folder is a Git repository.

Step 4: Create a file

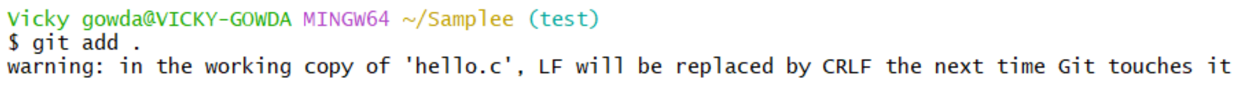


```
Vicky gowda@VICKY-GOWDA MINGW64 ~/Samplee (master)
$ vi hello.c
```

- This writes the text This is a sample file. Into a file named sample.txt.

- You can also create files using any editor.

Step 5: Stage the file

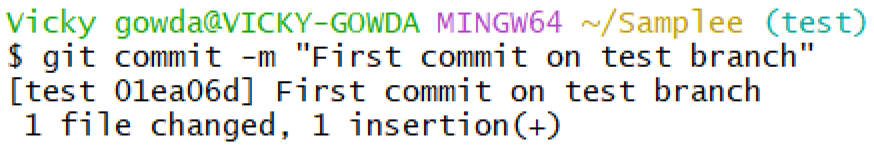


```
Vicky gowda@VICKY-GOWDA MINGW64 ~/Samplee (test)
$ git add .
warning: in the working copy of 'hello.c', LF will be replaced by CRLF the next time Git touches it
```

- This tells Git to “stage” (prepare) the file sample.txt for committing.

- Staging means you are telling Git what changes to include in the next commit.

<u>Step 6:</u> Commit the staged file



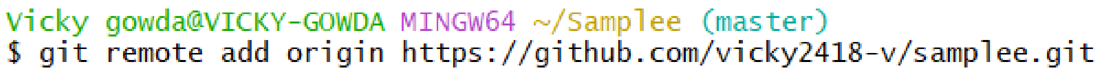
```
Vicky gowda@VICKY-GOWDA MINGW64 ~/Samplee (test)
$ git commit -m "First commit on test branch"
[test 01ea06d] First commit on test branch
 1 file changed, 1 insertion(+)
```

- This creates a commit, which is like a snapshot of your project.

- The -m flag adds a message describing the commit ("Add sample.txt").

- Commits save your work history.

<u>Step 7:</u> Add the remote repository URL



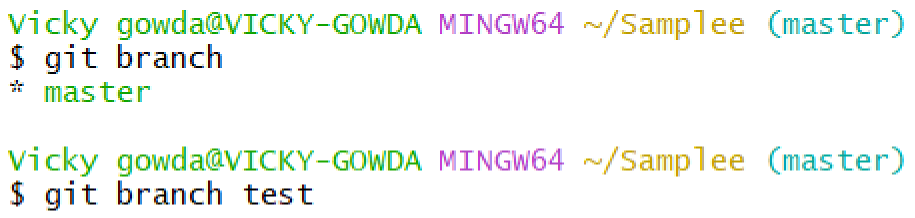
```
Vicky gowda@VICKY-GOWDA MINGW64 ~/Samplee (master)
$ git remote add origin https://github.com/vicky2418-v/samplee.git
```

- This links your local Git repository to the remote one on GitHub.

- Origin is the default name for your remote repository.

- Replace the URL with your actual GitHub repo URL.

Step 8: Push your commit to GitHub



```
Vicky gowda@VICKY-GOWDA MINGW64 ~/Samplee (master)
$ git branch
* master

Vicky gowda@VICKY-GOWDA MINGW64 ~/Samplee (master)
$ git branch test
```

- Git branch -M main: renames your branch to main (modern default branch name).

- Git push -u origin main: uploads your local commits to GitHub.

- The -u flag sets the remote origin/main as the default upstream branch.

PART 3: Verify

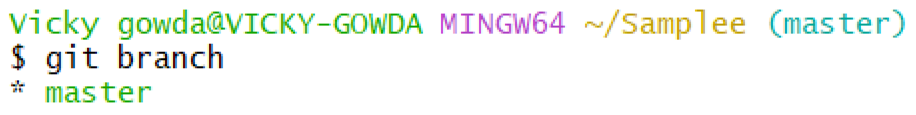
- Go to your GitHub repository page in a browser.
- You should see sample.txt uploaded there.

Lab Exercise : 5

Branches Creation

Branches in Git allow you to work on different features, bug fixes, or experiments without affecting the main codebase. Here’s how to create and manage branches using Git Bash:

1. View Current Branches

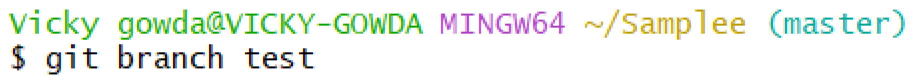


```
Vicky gowda@VICKY-GOWDA MINGW64 ~/Samplee (master)
$ git branch
* master
```

- Shows all local branches.
- The currently active branch is highlighted with *.

---

2. Create a New Branch



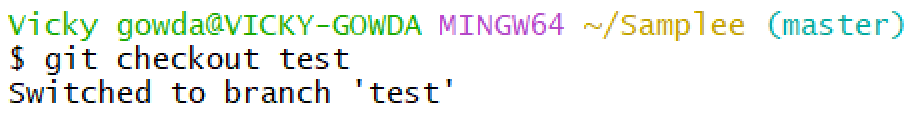
```
Vicky gowda@VICKY-GOWDA MINGW64 ~/Samplee (master)
$ git branch test
```

- This creates a new branch called new-feature, but does not switch to it.

---

3. Switch to the New Branch

- Changes your working directory to the new-feature branch.



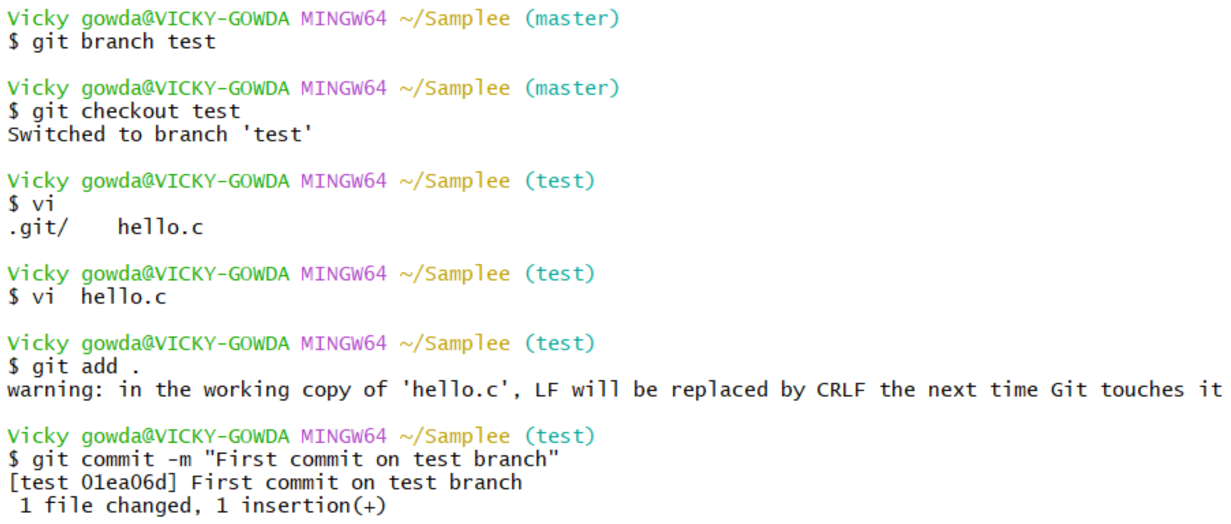
```
Vicky gowda@VICKY-GOWDA MINGW64 ~/Samplee (master)
$ git checkout test
Switched to branch 'test'
```

OR **create and switch in one step:**

Git checkout -b test

---

4. Make Changes and Commit (on the new branch)



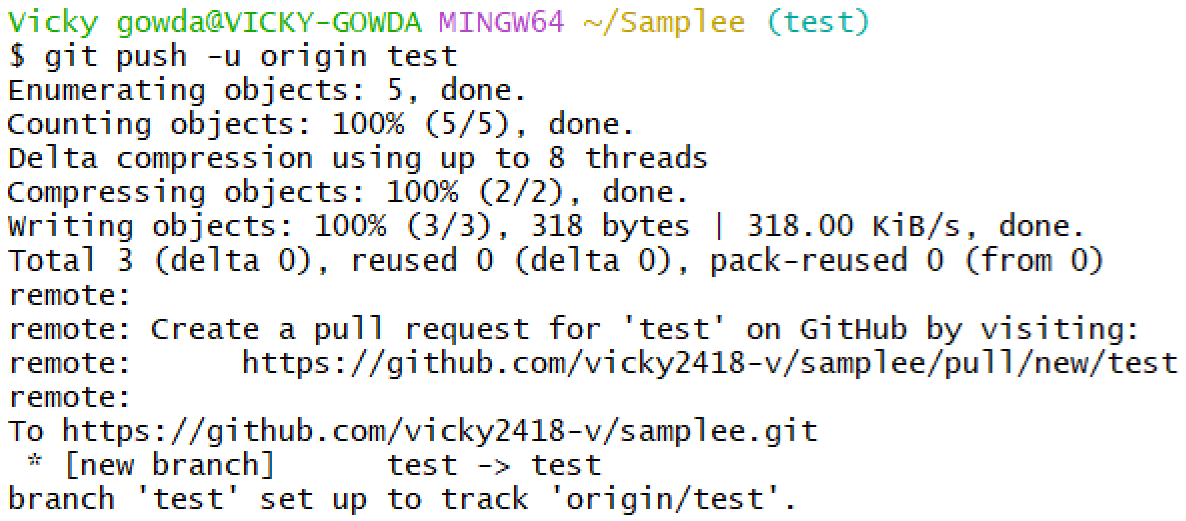
```
Vicky gowda@VICKY-GOWDA MINGW64 ~/Samplee (master)
$ git branch test

Vicky gowda@VICKY-GOWDA MINGW64 ~/Samplee (master)
$ git checkout test
Switched to branch 'test'

Vicky gowda@VICKY-GOWDA MINGW64 ~/Samplee (test)
$ vi
.git/    hello.c

Vicky gowda@VICKY-GOWDA MINGW64 ~/Samplee (test)
$ vi  hello.c

Vicky gowda@VICKY-GOWDA MINGW64 ~/Samplee (test)
$ git add .
warning: in the working copy of 'hello.c', LF will be replaced by CRLF the next time Git touches it

Vicky gowda@VICKY-GOWDA MINGW64 ~/Samplee (test)
$ git commit -m "First commit on test branch"
[test 01ea06d] First commit on test branch
 1 file changed, 1 insertion(+)
```

- Git add feature.txt

- Git commit -m “Add feature.txt in new-feature branch”

---

5. Push the New Branch to GitHub

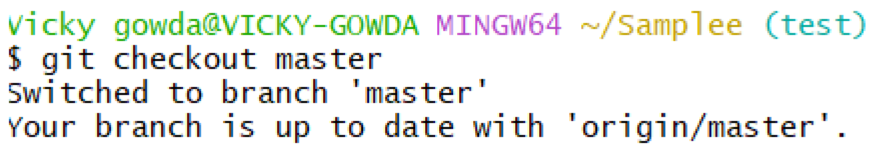


```
Vicky gowda@VICKY-GOWDA MINGW64 ~/Samplee (test)
$ git push -u origin test
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 8 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 318 bytes | 318.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
remote:
remote: Create a pull request for 'test' on GitHub by visiting:
remote:      https://github.com/vicky2418-v/samplee/pull/new/test
remote:
To https://github.com/vicky2418-v/samplee.git
 * [new branch]      test -> test
branch 'test' set up to track 'origin/test'.
```

- The -u flag sets the remote new-feature branch as the default upstream for this branch.

---

6. Switch Back to Main Branch

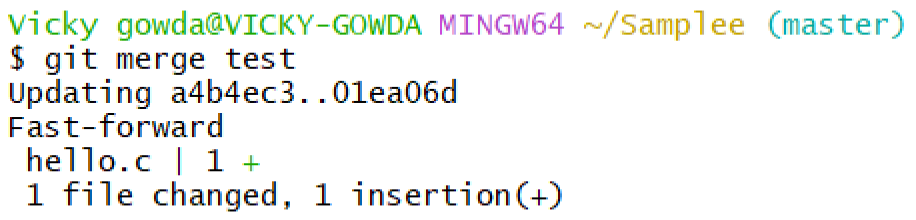


```
Vicky gowda@VICKY-GOWDA MINGW64 ~/Samplee (test)
$ git checkout master
Switched to branch 'master'
Your branch is up to date with 'origin/master'.
```

---

8. Merge New Branch into Main (Optional)

- First, make sure you’re on the master branch:



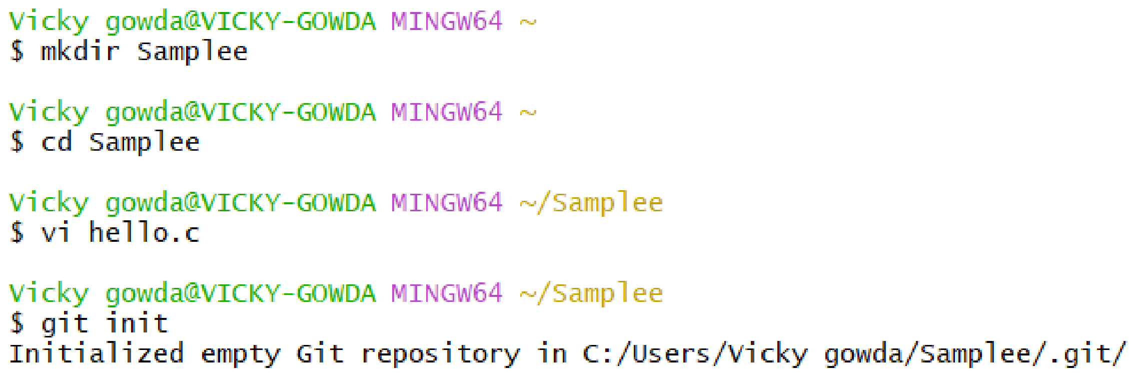
```
Vicky gowda@VICKY-GOWDA MINGW64 ~/Samplee (master)
$ git merge test
Updating a4b4ec3..01ea06d
Fast-forward
 hello.c | 1 +
 1 file changed, 1 insertion(+)
```

- This merges the new-feature branch into master.

Lab Exercise : 6

Git commit and Merge (Merge Request)

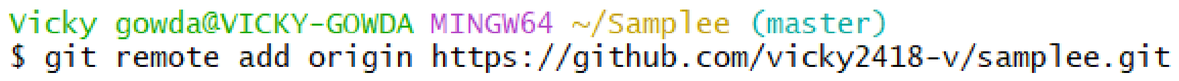
STEP 1: Create a Local Git Repository (if not already done)



```
Vicky gowda@VICKY-GOWDA MINGW64 ~
$ mkdir Samplee

Vicky gowda@VICKY-GOWDA MINGW64 ~
$ cd Samplee

Vicky gowda@VICKY-GOWDA MINGW64 ~/Samplee
$ vi hello.c

Vicky gowda@VICKY-GOWDA MINGW64 ~/Samplee
$ git init
Initialized empty Git repository in C:/Users/Vicky gowda/Samplee/.git/
```

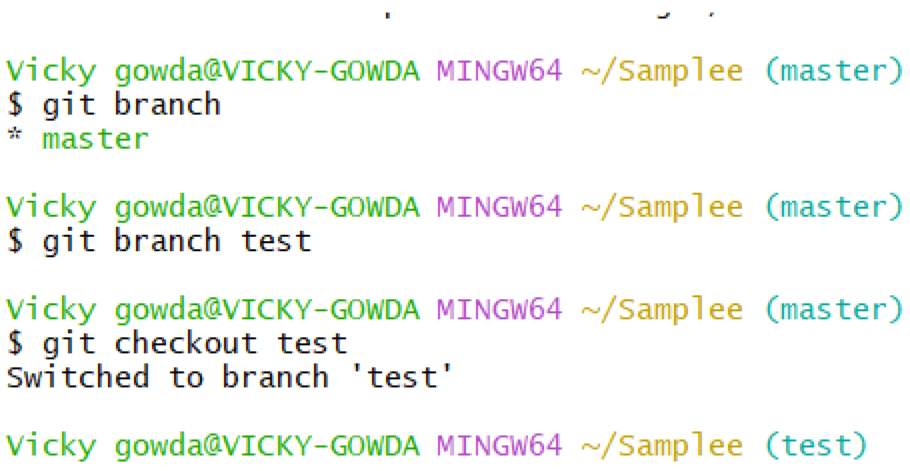
STEP 2: Add a Remote GitHub Repository

- Create a new repo on GitHub, then connect:



```
Vicky gowda@VICKY-GOWDA MINGW64 ~/Samplee (master)
$ git remote add origin https://github.com/vicky2418-v/samplee.git
```

STEP 3: Create and Switch to a New Branch

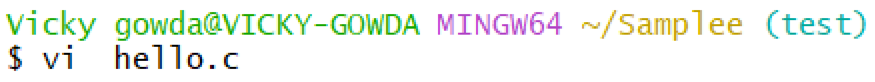


```
Vicky gowda@VICKY-GOWDA MINGW64 ~/Samplee (master)
$ git branch
* master

Vicky gowda@VICKY-GOWDA MINGW64 ~/Samplee (master)
$ git branch test

Vicky gowda@VICKY-GOWDA MINGW64 ~/Samplee (master)
$ git checkout test
Switched to branch 'test'

Vicky gowda@VICKY-GOWDA MINGW64 ~/Samplee (test)
```

- This is your test branch — where you make changes.

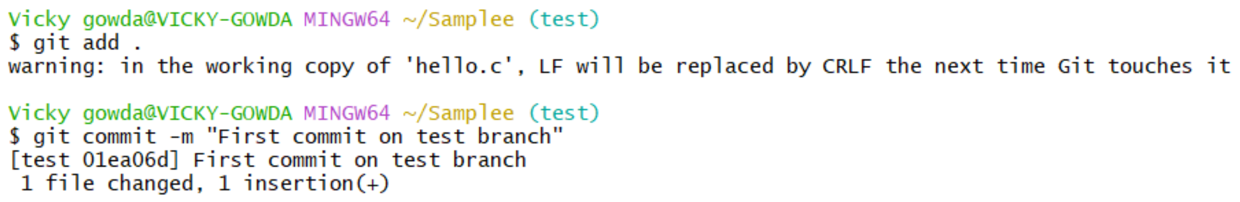
<u>STEP 4:</u> Add or Modify Files

Example:



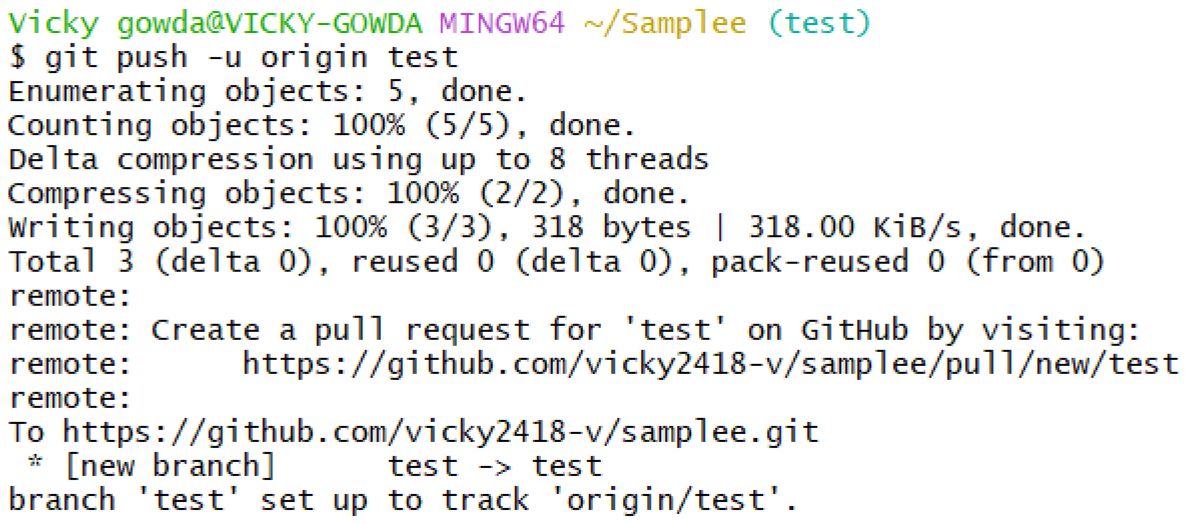
```
Vicky gowda@VICKY-GOWDA MINGW64 ~/Samplee (test)
$ vi  hello.c
```

STEP 5: Stage and Commit Changes



```
Vicky gowda@VICKY-GOWDA MINGW64 ~/Samplee (test)
$ git add .
warning: in the working copy of 'hello.c', LF will be replaced by CRLF the next time Git touches it

Vicky gowda@VICKY-GOWDA MINGW64 ~/Samplee (test)
$ git commit -m "First commit on test branch"
[test 01ea06d] First commit on test branch
 1 file changed, 1 insertion(+)
```

STEP 6: Push test Branch to GitHub



```
Vicky gowda@VICKY-GOWDA MINGW64 ~/Samplee (test)
$ git push -u origin test
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 8 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 318 bytes | 318.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
remote:
remote: Create a pull request for 'test' on GitHub by visiting:
remote:      https://github.com/vicky2418-v/samplee/pull/new/test
remote:
To https://github.com/vicky2418-v/samplee.git
 * [new branch]      test -> test
branch 'test' set up to track 'origin/test'.
```

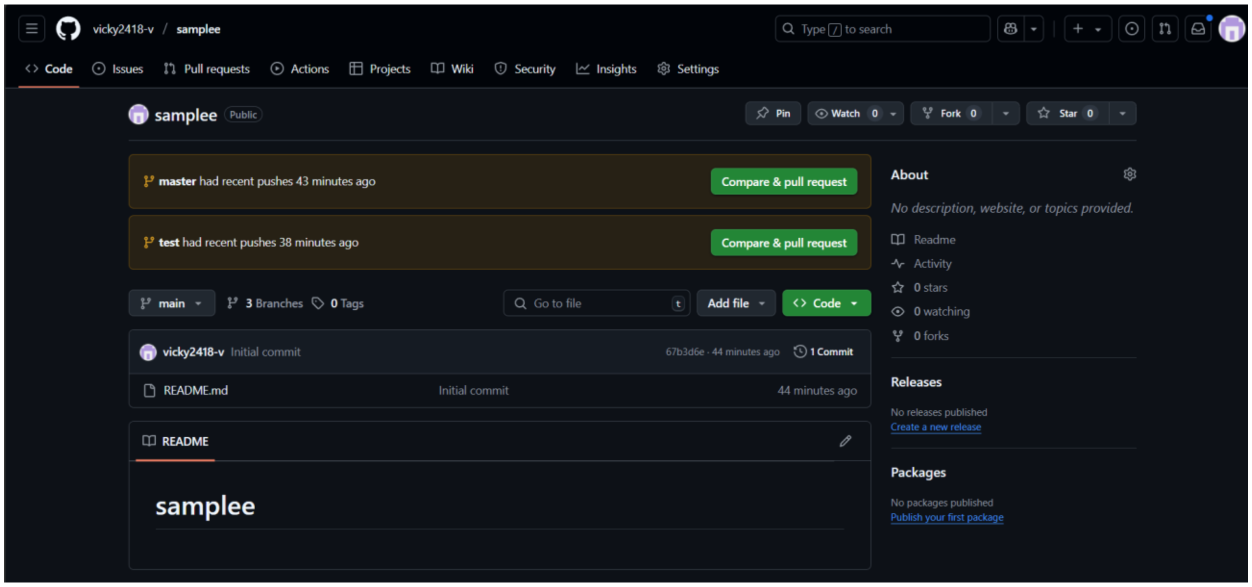
- This uploads your branch to GitHub.

<u>STEP 7:</u> Create a Merge Request (Pull Request) on GitHub

1. Go to your repo on GitHub

2. GitHub will show a “Compare & pull request” button — click it

(or go to the Pull Requests tab > click New pull request)



master had recent pushes 43 minutes ago

test had recent pushes 38 minutes ago

vicky2418-v Initial commit 67b3d6e · 44 minutes ago 1 Commit

README.md Initial commit 44 minutes ago

samplee

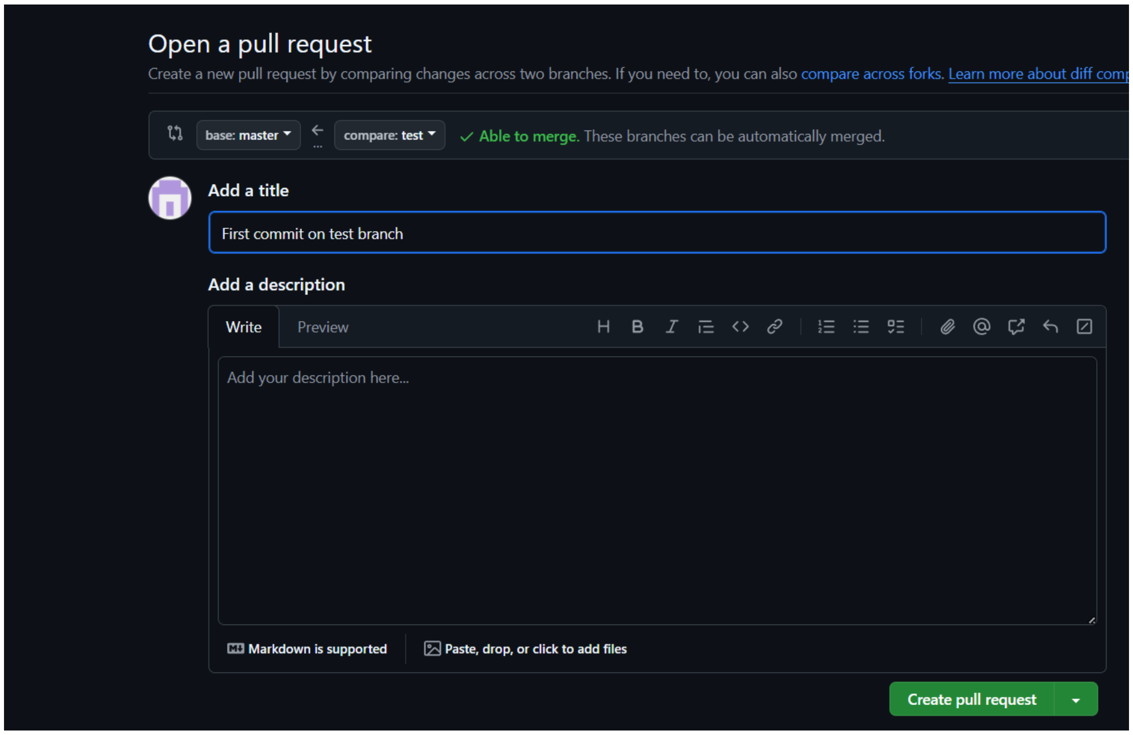
No description, website, or topics provided.

No releases published

No packages published

3. Set:

- Base branch: master (target)
- Compare branch: test (your work)


Open a pull request

Create a new pull request by comparing changes across two branches. If you need to, you can also compare across forks. Learn more about diff comp

base: master ← compare: test ✓ Able to merge. These branches can be automatically merged.

Add a title

First commit on test branch

Add a description

Write Preview

Add your description here...

Markdown is supported Paste, drop, or click to add files

Create pull request

4. Add a title and description

5. Click “Create pull request”

<u>STEP 8:</u> Review and Merge

- You or a teammate reviews the code

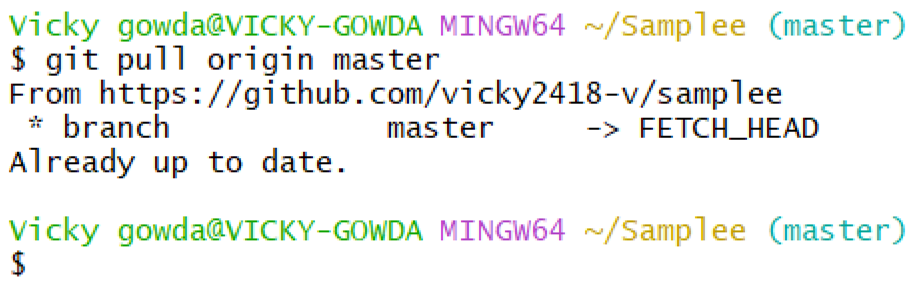
- If everything looks good, click “Merge pull request”
- Confirm the merge

STEP 9: Delete the Feature Branch (Optional)

- GitHub will offer an option to delete the branch. Or do it locally:

```
Git branch -d test
```

STEP 10: Update Local Master Branch



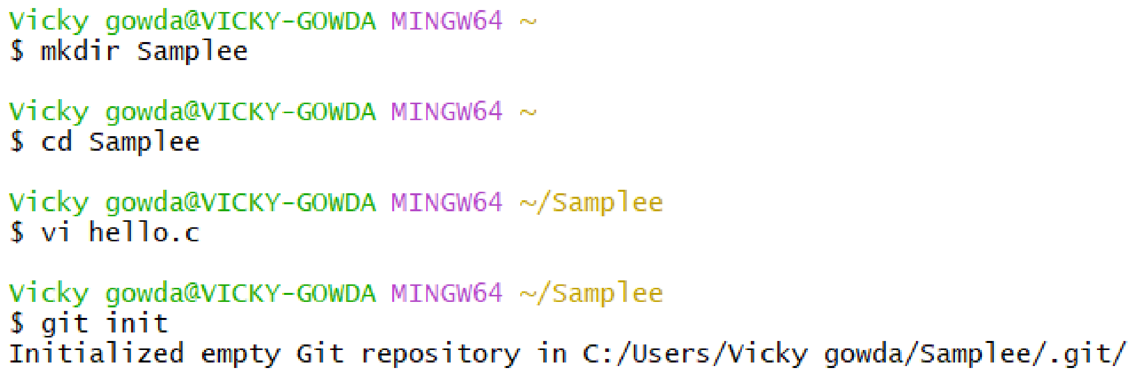
```
Vicky gowda@VICKY-GOWDA MINGW64 ~/Samplee (master)
$ git pull origin master
From https://github.com/vicky2418-v/samplee
 * branch            master     -> FETCH_HEAD
Already up to date.

Vicky gowda@VICKY-GOWDA MINGW64 ~/Samplee (master)
$
```

Lab Exercise 7 :

Open and Close Pull Request

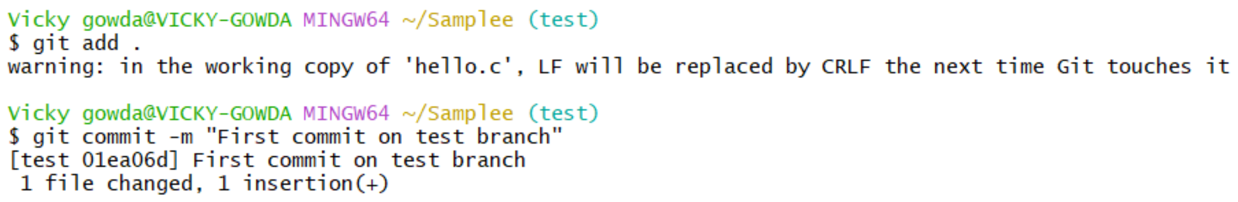
STEP-BY-STEP WORKFLOW :

1. Initialize Local Repo (Git Bash)



```
Vicky gowda@VICKY-GOWDA MINGW64 ~
$ mkdir Samplee

Vicky gowda@VICKY-GOWDA MINGW64 ~
$ cd Samplee

Vicky gowda@VICKY-GOWDA MINGW64 ~/Samplee
$ vi hello.c

Vicky gowda@VICKY-GOWDA MINGW64 ~/Samplee
$ git init
Initialized empty Git repository in C:/Users/Vicky gowda/Samplee/.git/
```

2. Create a file ,Commit it

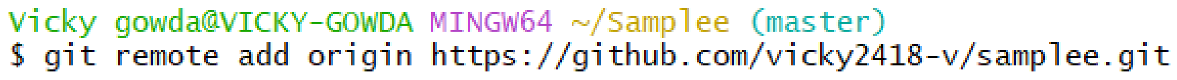


```
Vicky gowda@VICKY-GOWDA MINGW64 ~/Samplee (test)
$ git add .
warning: in the working copy of 'hello.c', LF will be replaced by CRLF the next time Git touches it

Vicky gowda@VICKY-GOWDA MINGW64 ~/Samplee (test)
$ git commit -m "First commit on test branch"
[test 01ea06d] First commit on test branch
 1 file changed, 1 insertion(+)
```

3.connect to GitHub Repository

On GitHub:

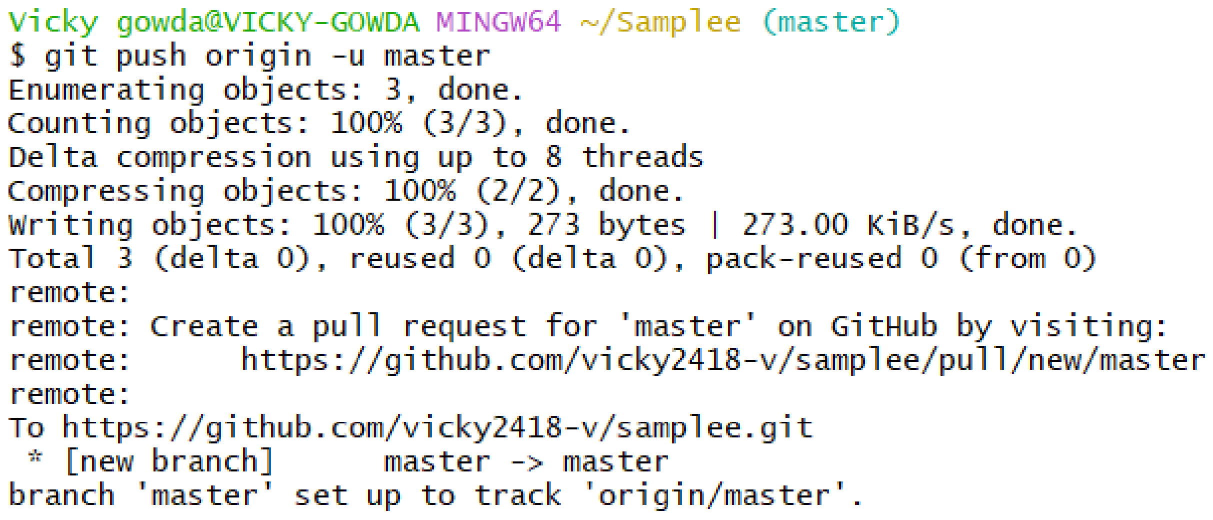
Create a new repo: my-project Back in Git Bash:



```
Vicky gowda@VICKY-GOWDA MINGW64 ~/Samplee (master)
$ git remote add origin https://github.com/vicky2418-v/samplee.git
```

---

4. Push to Master Branch



```
Vicky gowda@VICKY-GOWDA MINGW64 ~/Samplee (master)
$ git push origin -u master
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Delta compression using up to 8 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 273 bytes | 273.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
remote:
remote: Create a pull request for 'master' on GitHub by visiting:
remote:      https://github.com/vicky2418-v/samplee/pull/new/master
remote:
To https://github.com/vicky2418-v/samplee.git
 * [new branch]      master -> master
branch 'master' set up to track 'origin/master'.
```

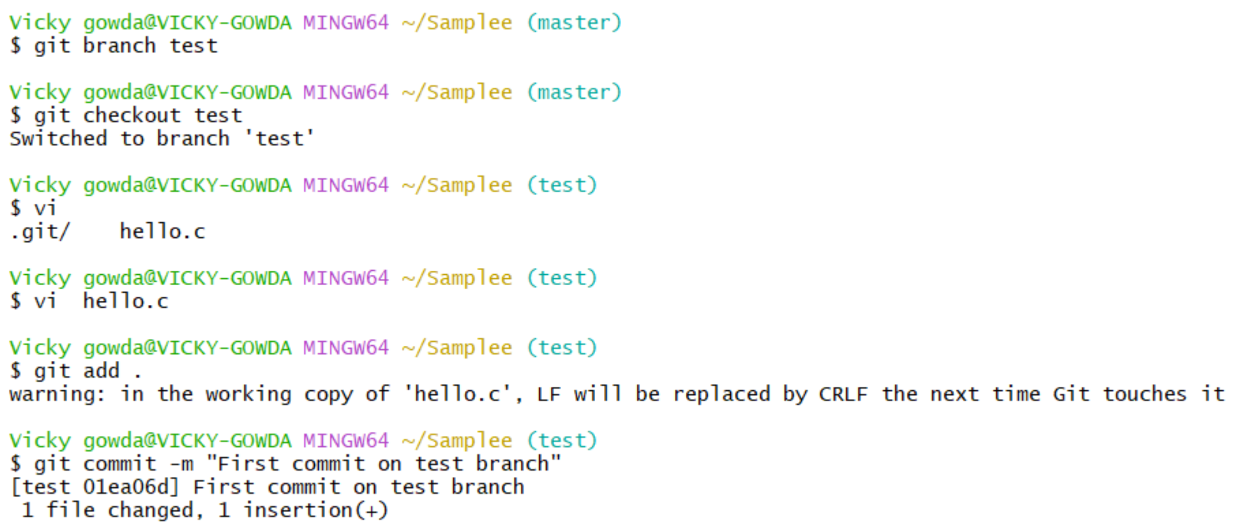
---

5. Create a Feature Branch and Switch to It

Git checkout -b test

---

6. Add Another File and Commit



```
Vicky gowda@VICKY-GOWDA MINGW64 ~/Samplee (master)
$ git branch test

Vicky gowda@VICKY-GOWDA MINGW64 ~/Samplee (master)
$ git checkout test
Switched to branch 'test'

Vicky gowda@VICKY-GOWDA MINGW64 ~/Samplee (test)
$ vi
.git/    hello.c

Vicky gowda@VICKY-GOWDA MINGW64 ~/Samplee (test)
$ vi  hello.c

Vicky gowda@VICKY-GOWDA MINGW64 ~/Samplee (test)
$ git add .
warning: in the working copy of 'hello.c', LF will be replaced by CRLF the next time Git touches it

Vicky gowda@VICKY-GOWDA MINGW64 ~/Samplee (test)
$ git commit -m "First commit on test branch"
[test 01ea06d] First commit on test branch
 1 file changed, 1 insertion(+)
```

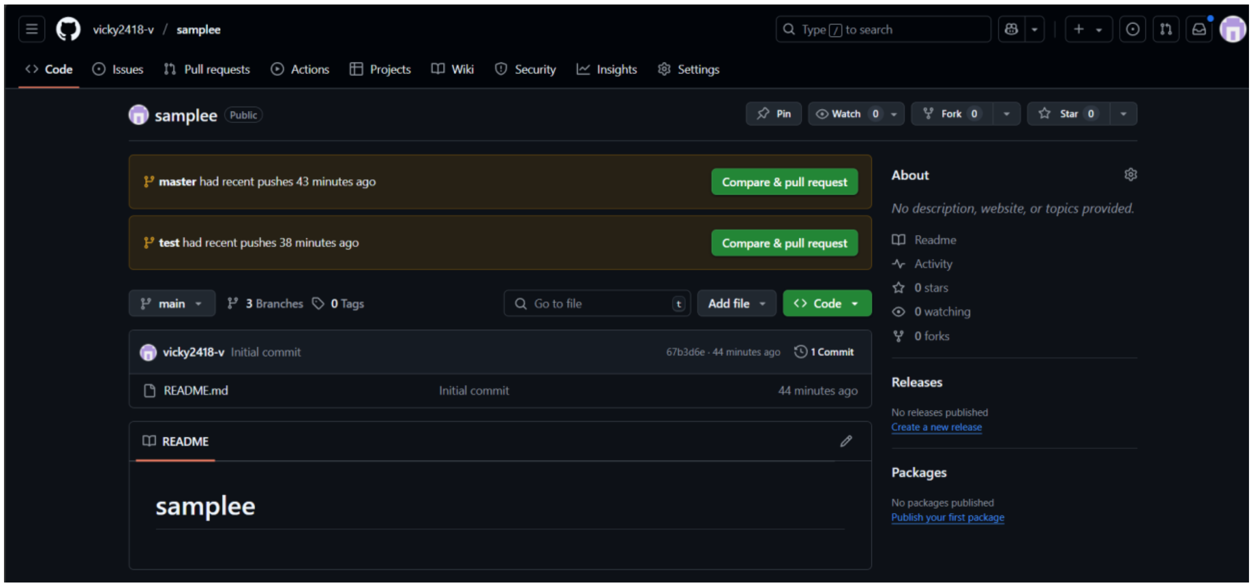
---

7. Push the Feature Branch to GitHub Git push -u origin test

---

8. Open a Pull Request (on GitHub)

1. Go to your repo on GitHub.
2. GitHub will show “Compare & pull request” button — click it.



master had recent pushes 43 minutes ago

test had recent pushes 38 minutes ago

vicky2418-v Initial commit 67b3d6e · 44 minutes ago 1 Commit

README.md Initial commit 44 minutes ago

samplee

About

No description, website, or topics provided.

Readme
Activity
0 stars
0 watching
0 forks

Releases

No releases published
Create a new release

Packages

No packages published
Publish your first package

3. Set:

- Base: master
- Compare: test

4. Add:

- Title: Add About Page
- Description: This PR adds about.html with basic content

5. Click “Create pull request”

⭘ Pull request is now open!

---

9. Close a Pull Request (Without Merging)

1. Scroll to the bottom of the pull request page.
2. Click “Close pull request”
3. (Optional) Add a comment like “Work in progress” or “Not needed”
4. Click “Close”

⭘ Pull request is now closed and not merged.

10. Reopen (Optional)

- Go to the closed pull request
- Click “Reopen pull request”