

# SCM LAB RECORD



**Name: Dayashree S**

**Branch: B.Tech**

**Section: CSE**

**Course: Source Code Management**

**SEN: A86605223011**

**Faculty: Dr. Monit Kapoor**

**Date of Submission: 06 June 2025**

# GIT FUNDAMENTALS

## 1. Introduction to Git

Git is a **distributed version control system (DVCS)** developed by Linus Torvalds in 2005. It is used to manage and track changes in source code during software development, allowing multiple developers to collaborate efficiently.

## 2. Features of Git

- **Distributed:** Every developer has a full copy of the repository.
- **Speed:** Git operations are very fast due to local repositories.
- **Data Integrity:** Every file and commit is checksummed using SHA-1.
- **Support for Branching and Merging:** Git handles branching and merging efficiently.
- **Lightweight:** Branches and tags are lightweight and quick to create.

## 3. Basic Git Terminology

- **Repository (Repo):** A directory that contains your project work tracked by Git.
- **Commit:** A snapshot of changes made to files in a repository.
- **Branch:** A pointer to a specific commit; used for parallel development.
- **Merge:** The process of combining changes from different branches.
- **Staging Area:** A place where changes are grouped before committing.
- **Working Directory:** The local directory where files are modified by the user.

## 4. Branching in Git

Branching allows developers to work on different features, fixes, or experiments independently from the main codebase. A branch represents a separate line of development.

Common branch types:

- **Main or Master:** Stable and deployable version of the code.
- **Development (dev):** Integration branch for all features.
- **Feature Branches:** Created for developing individual features.
- **Hotfix Branches:** Used to fix urgent issues in production code.

After work on a branch is complete, it can be merged back into the main or development branch.

## 5. Merge Conflicts

A merge conflict occurs when changes in two branches affect the same part of a file and Git cannot automatically reconcile them. These conflicts must be resolved manually by the developer.

Once resolved, the merge can be completed, ensuring that the code integrates correctly without overwriting important changes.

## 6. .gitignore File

The .gitignore file is used to specify files and directories that Git should ignore and not track. This is useful for excluding temporary files, compiled code, environment configuration files, and other data that shouldn't be versioned.

Examples of ignored files:

- Log files
- System files like .DS\_Store
- Dependency folders like node\_modules
- Build folders like dist or out

## 7. Advantages of Git

- Facilitates collaboration in teams.
- Maintains a detailed history of code changes.
- Allows easy switching between different versions of code.
- Enhances productivity through branching and merging.
- Supports open-source development and contribution through platforms like GitHub, GitLab, etc.

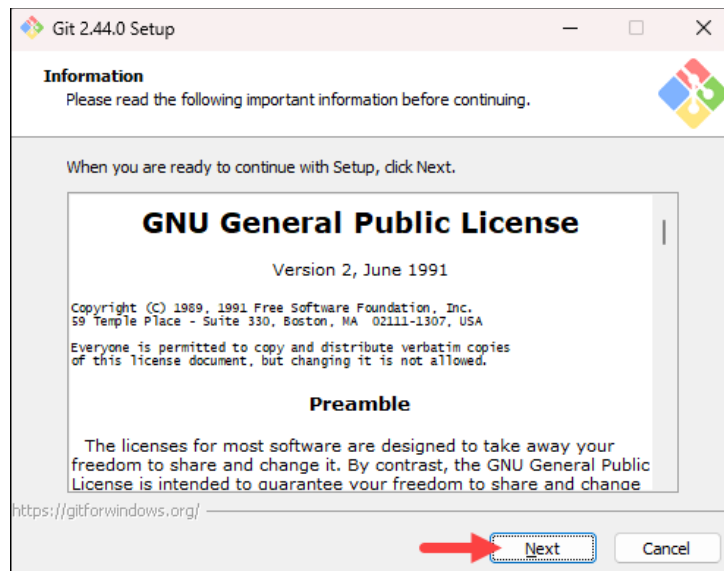
# GIT INSTALLATION

1. Navigate to the [official Git downloads page](#) and click the download link for the latest Git version

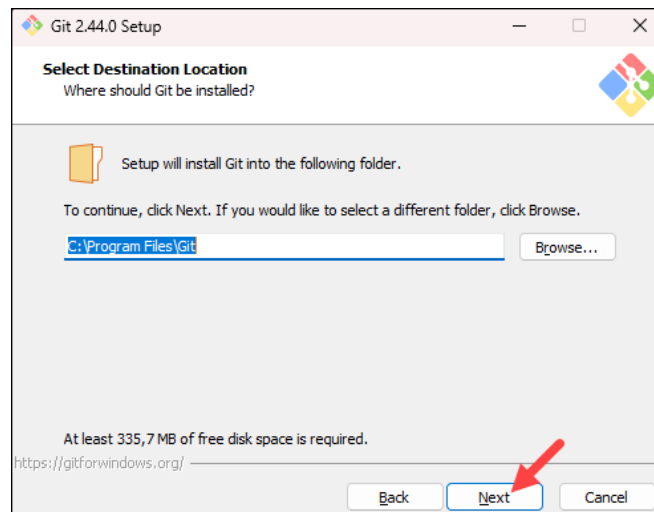


for Windows:

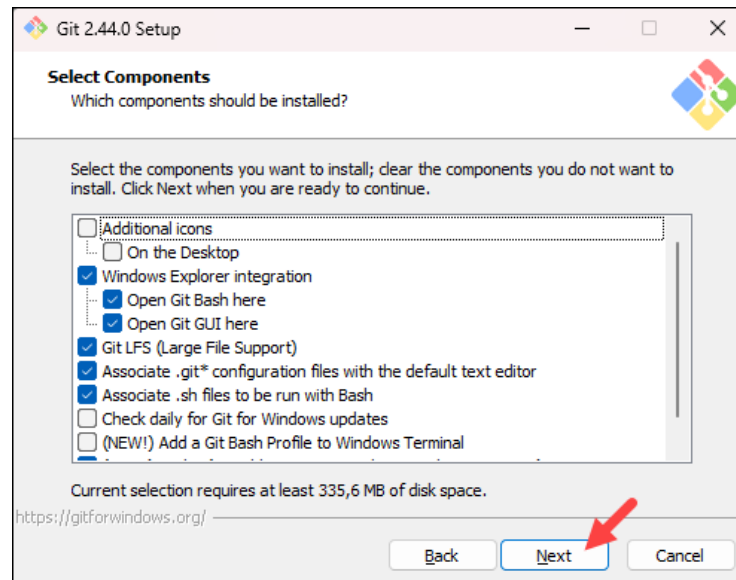
2. Double-click the downloaded file to extract and launch the installer
3. Review the GNU General Public License, and when you are ready to install, click **Next**.



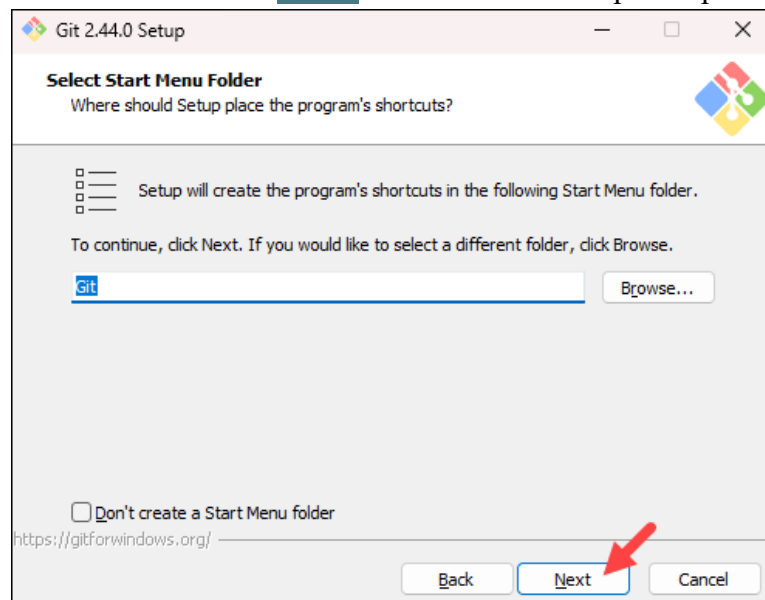
4. The installer prompts you for an installation location. Leave the default one unless you want to change it, and click **Next**.



5. In the component selection screen, leave the defaults unless you need to change them and click **Next**.

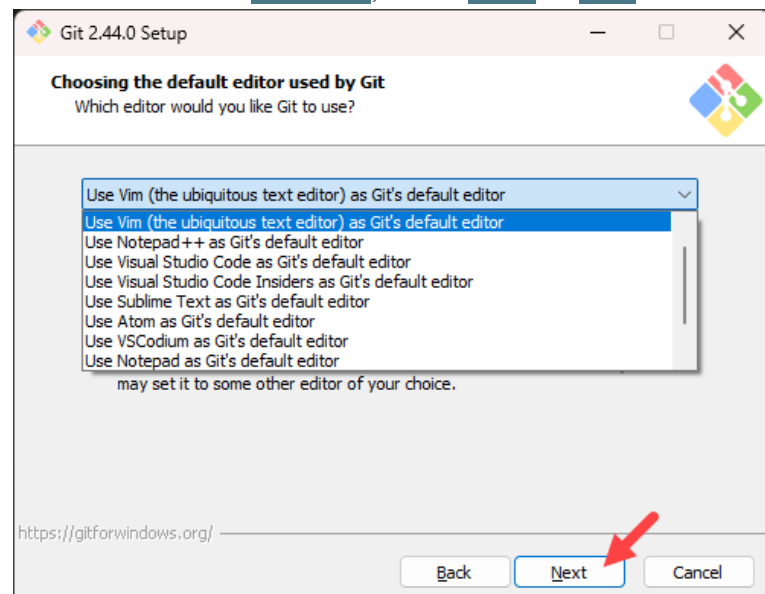


6. The installer offers to create a start menu folder. Click **Next** to accept and proceed to the next step.

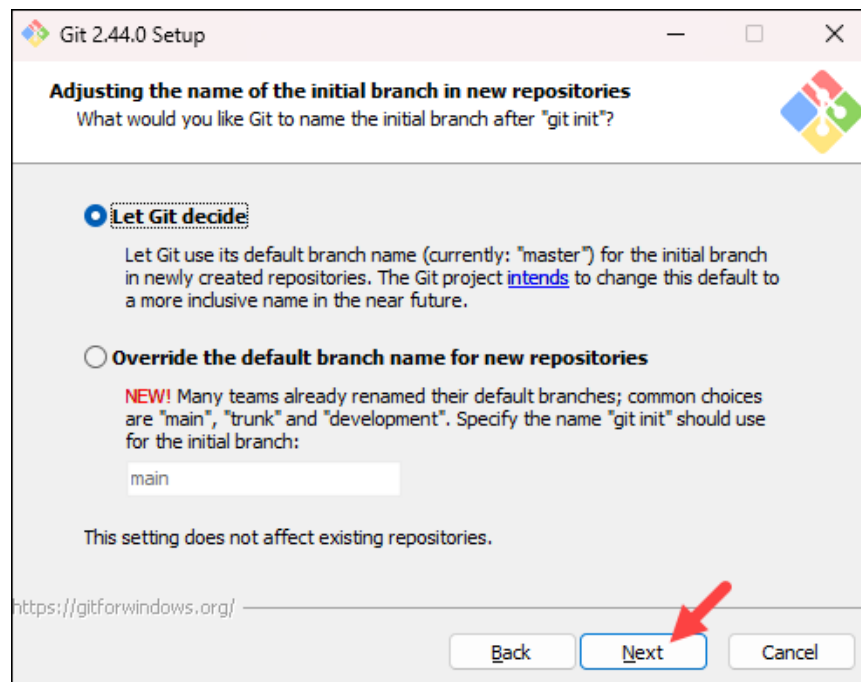


7. Select a text editor you want to use with Git. Use the drop-down menu to select Notepad++ (or whichever text editor you prefer) and click **Next**.

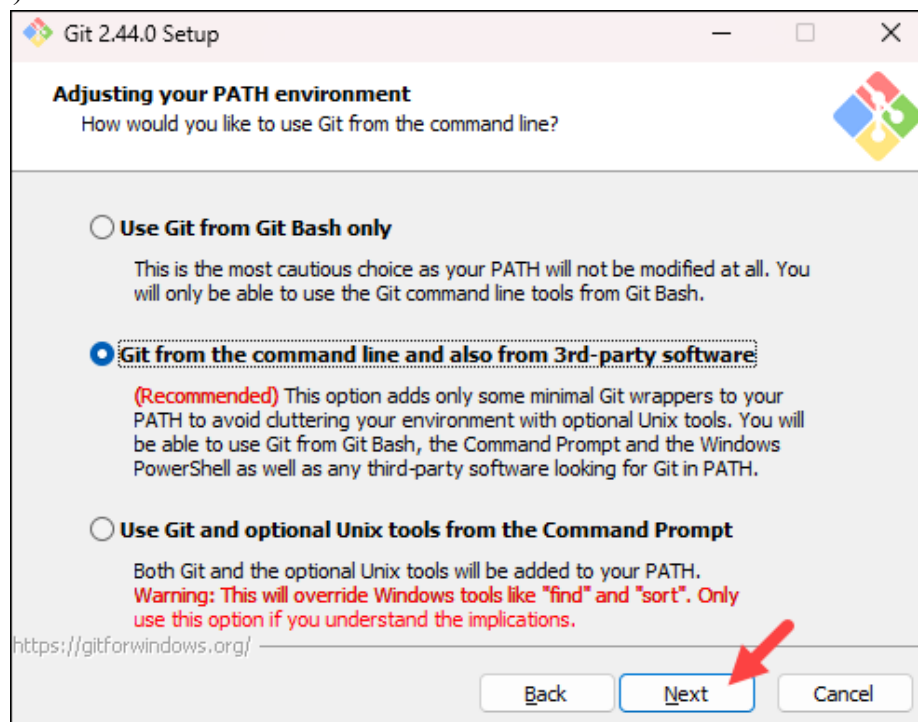
If you prefer to use a CLI text editor in Git Bash, select nano or Vim from the list.



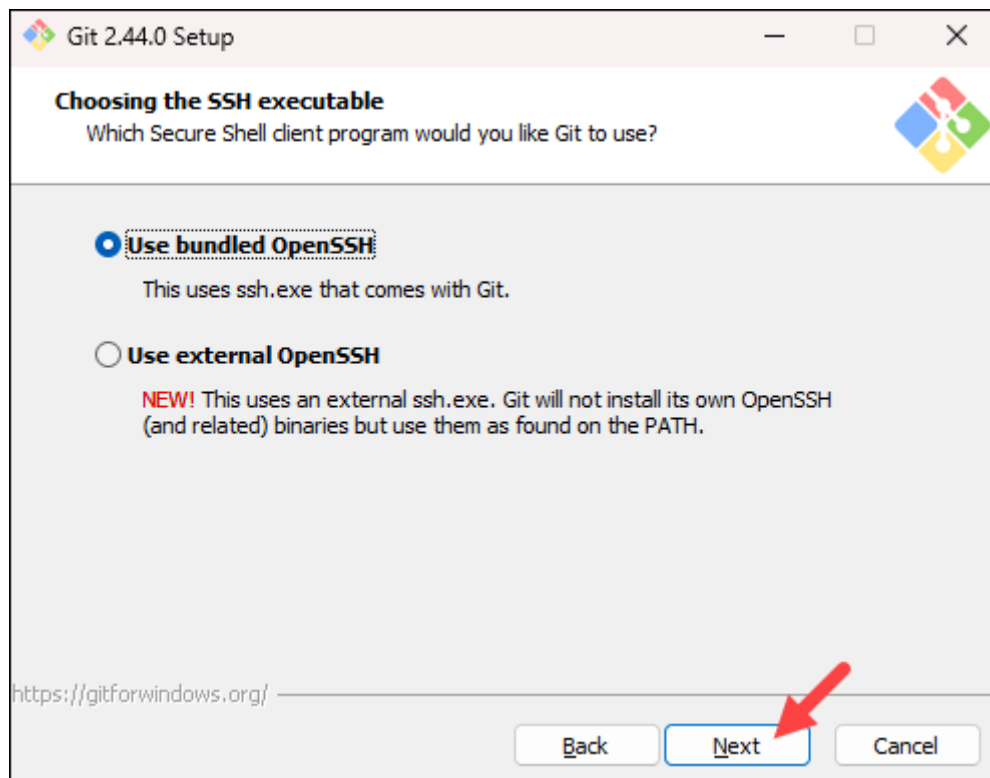
8. The next step allows you to choose a different name for your initial branch. The default is **master**. Unless you are working in a team that requires a different name, leave the default option and click **Next**.



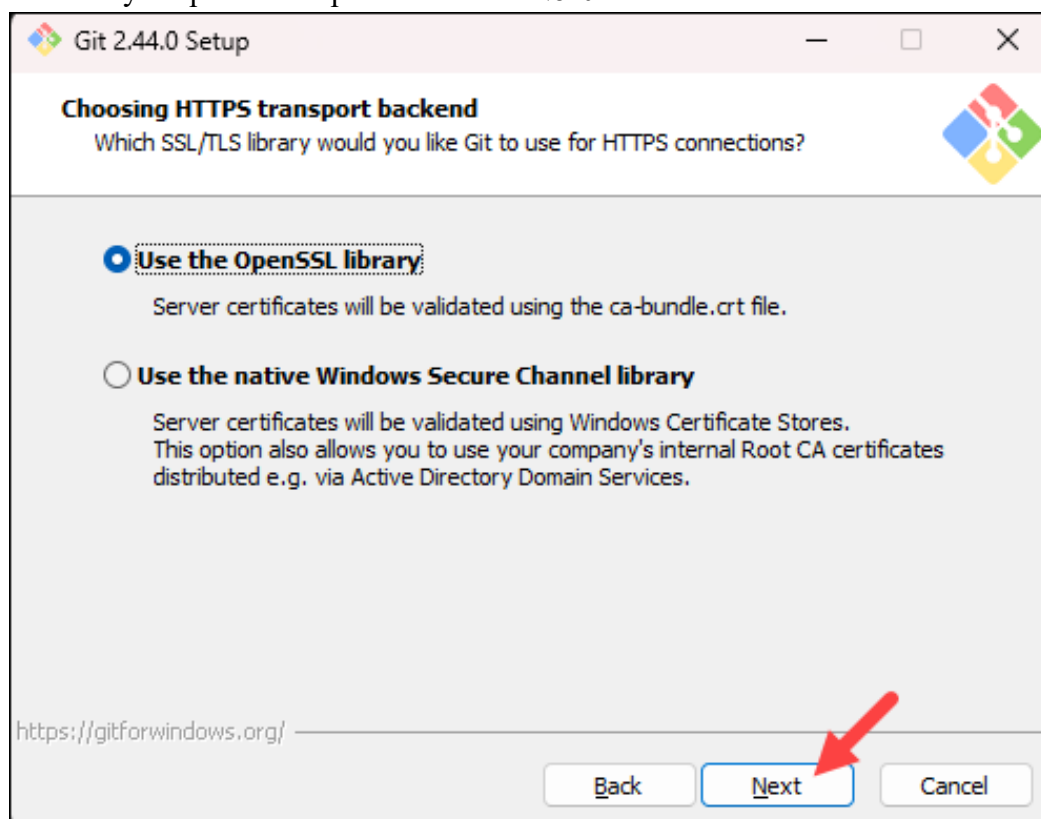
9. The next step allows you to change the **PATH environment**. The **PATH** is the default set of directories included when you run a command from the command line. Keep the middle (recommended) selection and click **Next**.



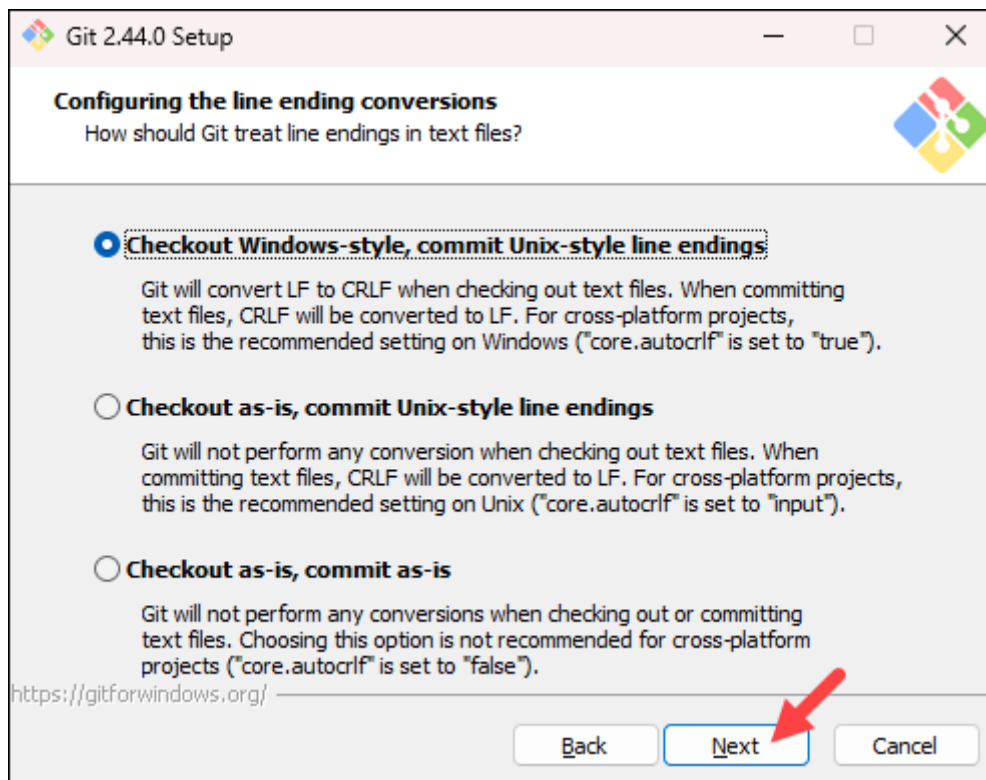
10. The installer prompts you to select the SSH client for Git to use. Git already comes with its own SSH client, so if you don't need a specific one, leave the default option and click **Next**.



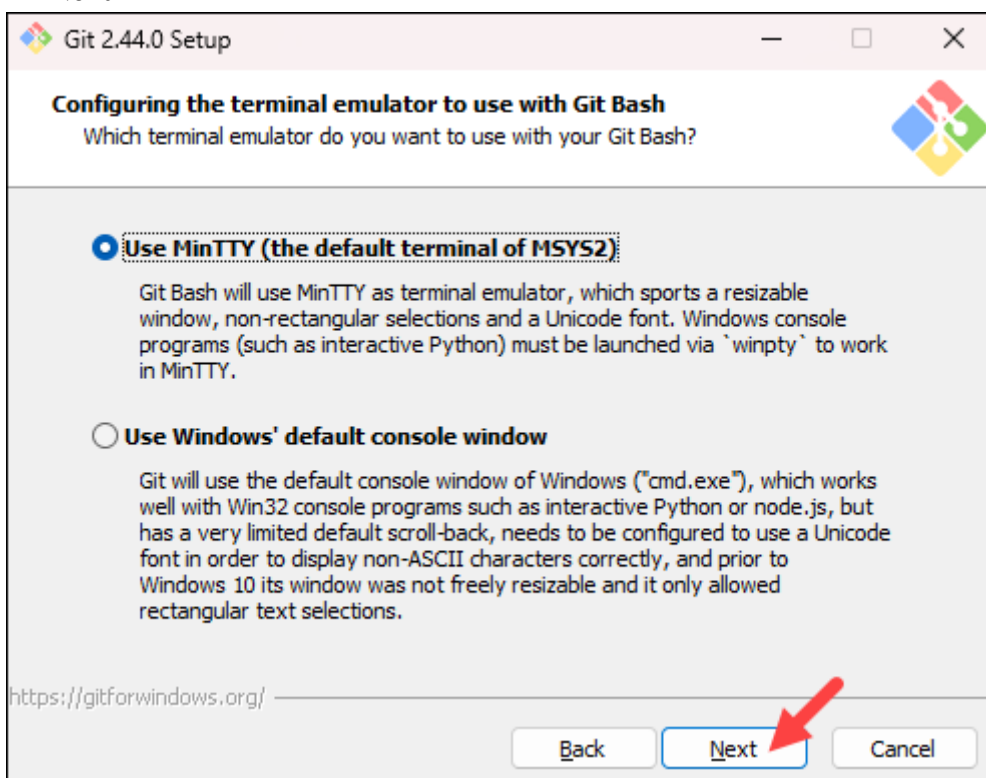
11. The next option relates to server certificates. The default option is recommended for most users. If you work in an Active Directory environment, you may need to switch to Windows Store certificates. Select your preferred option and click **Next**.



12. The following selection configures line-ending conversion, which relates to the way data is formatted. The default selection is recommended for Windows. Click **Next** to proceed.

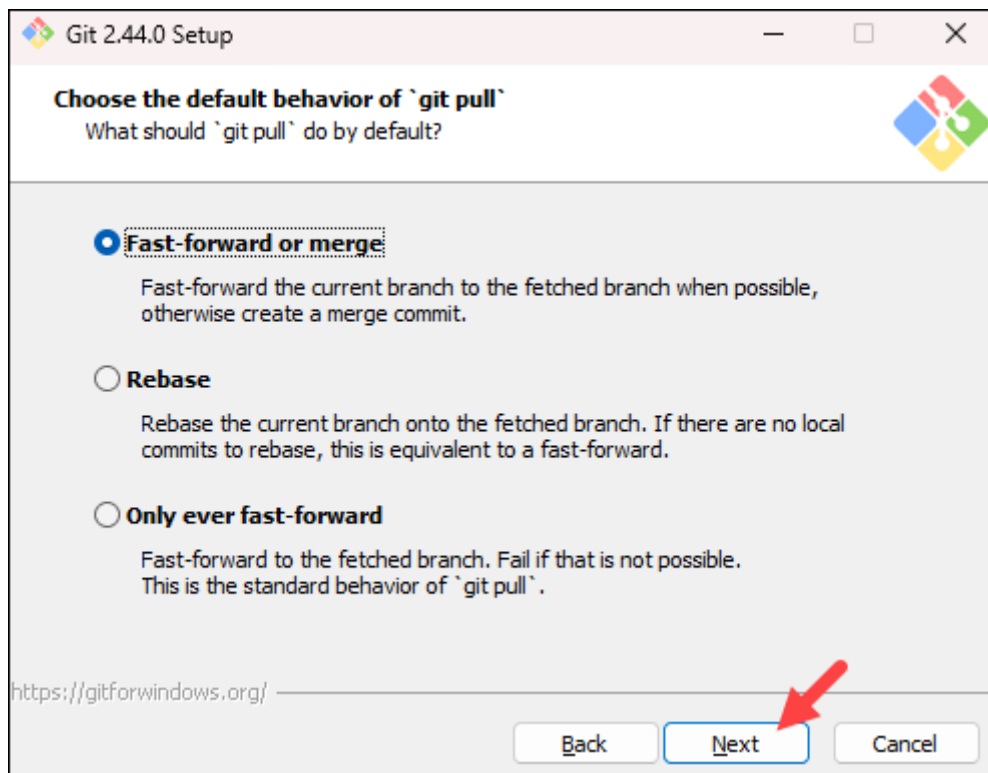


13. Choose the terminal emulator you want to use. The default MinTTY is recommended for its features. Click **Next** to continue.

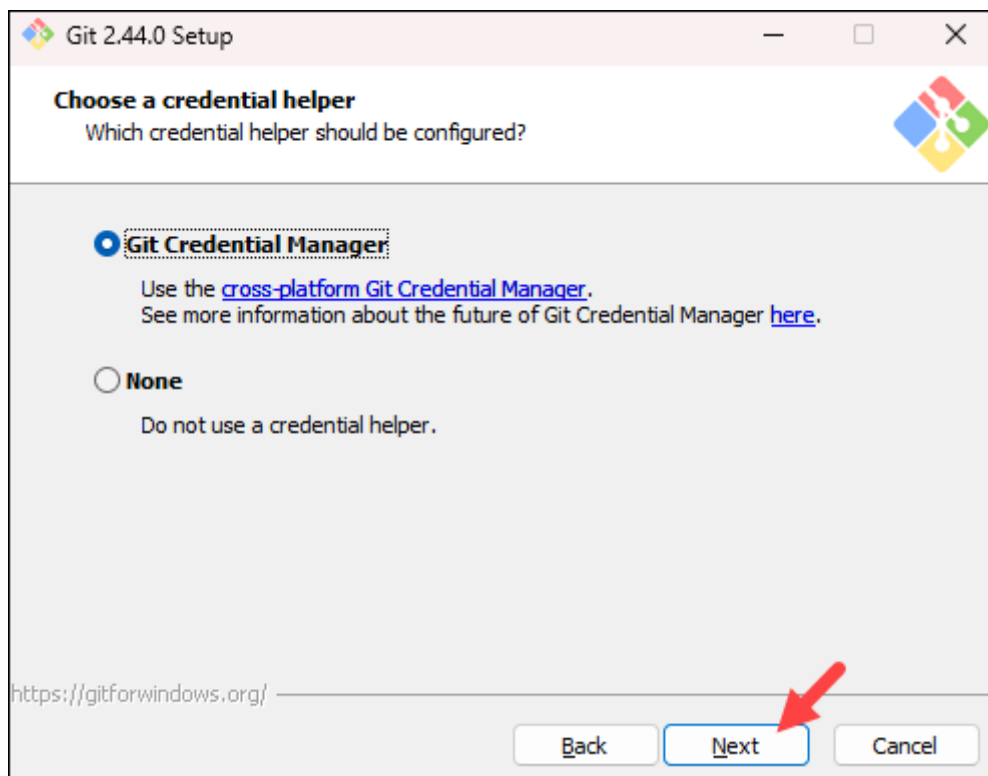


14. The next step allows you to choose what the **git pull** command will do. The default option is recommended unless you specifically need to change its behavior. Click **Next** to continue with the installation.

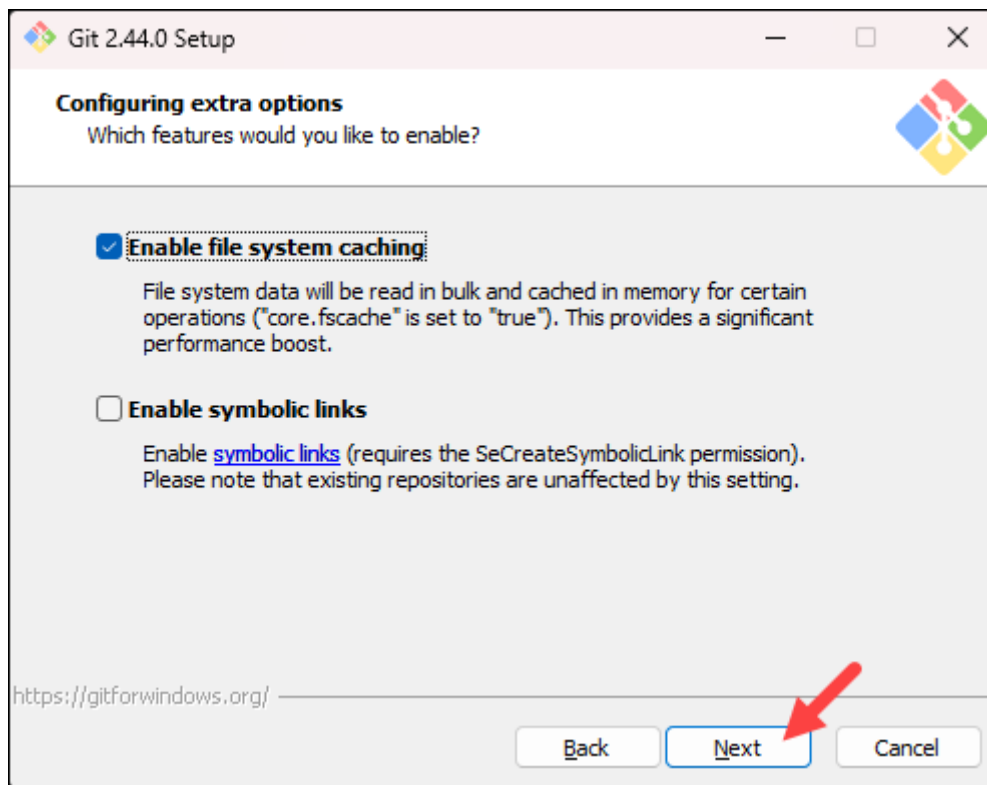




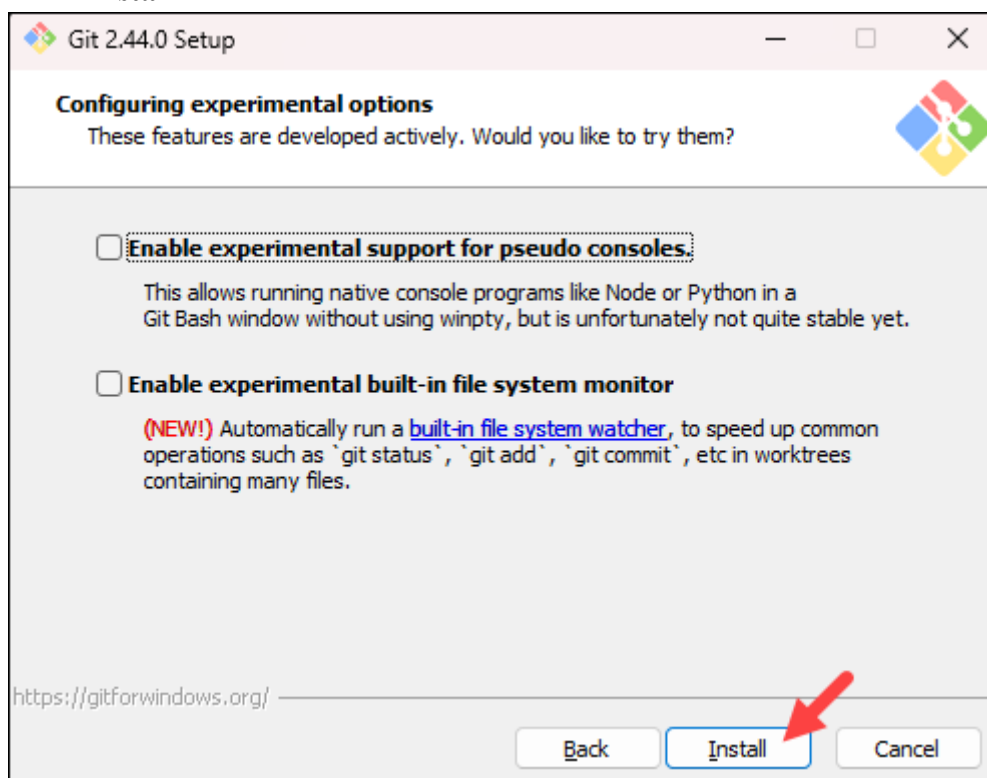
15. The next step is to choose which credential helper to use. Git uses credential helpers to fetch or save credentials. The default option is the most stable one. Select your preferred credential manager and click **Next**.



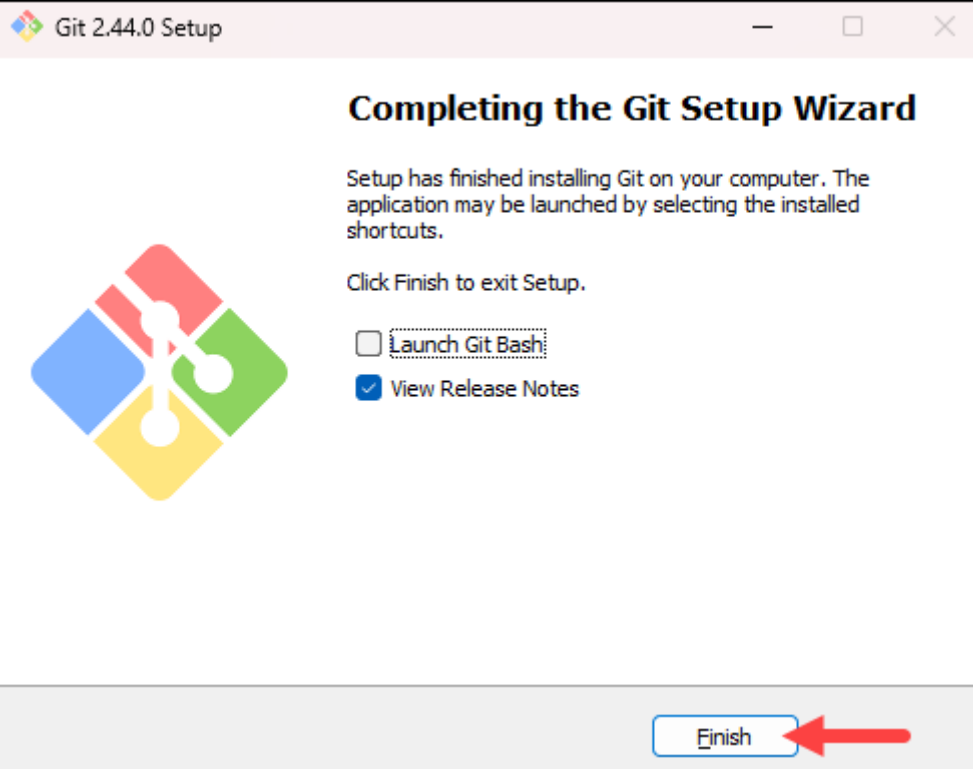
16. The next step lets you decide which extra options to enable. If you use [symbolic links](#), which represent shortcuts for the command line, tick the box. Keep [file system](#) caching checked and click **Next**.



17. Depending on which Git version you are installing, it may offer to install experimental features. At the time this article was written, the installer offered options to include support for pseudo controls and a built-in file system monitor. For the most stable operation, do not install experimental features and click **Install**.



18. Once the installation is complete, tick the boxes to view the Release Notes or launch Git Bash if you want to start using Git right away, and click **Finish**.



# BASIC COMMANDS IN GIT

## 1. pwd

- *Purpose:* Displays the Present Working Directory.
- *Usage:* Use this to find the directory you're currently working in.

```
dayashree@DESKTOP-J19GU2A MINGW64 ~  
$ pwd  
/c/Users/dayashree
```

## 2. cd

- *Purpose:* Change directory.
- *Usage:* Navigate to a different folder.

```
dayashree@DESKTOP-J19GU2A MINGW64 ~  
$ cd ..  
  
dayashree@DESKTOP-J19GU2A MINGW64 /c/Users  
$ |
```

## 3. ls

- *Purpose:* Lists all folders in the current directory.

```
dayashree@DESKTOP-J19GU2A MINGW64 /c/Users  
$ ls  
'All Users'@  Default/  'Default User'@  Public/  black/  dayashree/  desktop.ini
```

## 4. Vi Command

- *Purpose:* Open the Vim/Vi editor to create or edit files.
- *Usage:* vi filename. Press i to insert, esc to exit insert mode, and :wq to save and quit.

## 5. cat

- *Purpose:* View file contents.
- *Usage:* cat filename.

```
dayashree@DESKTOP-J19GU2A MINGW64 ~/project-3  
$ vi apple.txt  
  
dayashree@DESKTOP-J19GU2A MINGW64 ~/project-3  
$ cat apple.txt  
hello world
```

## 6. ls -l

- *Purpose:* Lists files in long format with additional details (permissions, owners, size).

```

dayashree@DESKTOP-J19GU2A MINGW64 ~/project-2 (master)
$ ls -l
total 2
-rw-r--r-- 1 dayashree 197121 39 Feb  7 17:06 bye.py
-rw-r--r-- 1 dayashree 197121 49 Feb 13 23:52 hello.txt
drwxr-xr-x 1 dayashree 197121  0 Feb  7 18:47 python-1/

dayashree@DESKTOP-J19GU2A MINGW64 ~/project-2 (master)
$ |

```

## 7. ls -ah

- *Purpose:* Lists all files, including hidden ones.

```

dayashree@DESKTOP-J19GU2A MINGW64 ~/project-2 (master)
$ ls -ah
-/.  ../  .git/  bye.py  hello.txt  python-1/

dayashree@DESKTOP-J19GU2A MINGW64 ~/project-2 (master)
$ |

```

## 8. history

- *Purpose:* Displays the history of previously run commands.

```

dayashree@DESKTOP-J19GU2A MINGW64 ~/project-2 (master)
$ history
 1  --version
 2  git --version
 3  git --version
 4  ls
 5  pwd
 6  git
 7  pwd
 8  cd ..
 9  clear
10  pwd
11  ls
12  cd
13  ls
14  vi
15  vi apple
16  vi orange.txt
17  vi orange.txt
18  cat orange.txt
19  ls-l
20  ls-l
21  ls-ah
22  ls -l
23  history

```

## 9. mkdir

- *Purpose:* Create a new folder.
- *Usage:* mkdir foldername.

```

dayashree@DESKTOP-J19GU2A MINGW64 ~
$ mkdir project-3

dayashree@DESKTOP-J19GU2A MINGW64 ~
$ cd project-3

dayashree@DESKTOP-J19GU2A MINGW64 ~/project-3
$ vi apple.txt

```

## 10. git init

- *Purpose:* Initializes an empty Git repository in the current directory.

```
dayashree@DESKTOP-J19GU2A MINGW64 ~/project-3
$ git init
Initialized empty Git repository in C:/Users/dayashree/project-3/.git/
```

## 11. git status

- *Purpose:* Displays the current status of the working directory and staging area, showing:
- Modified files not yet staged.
- Staged files ready for commit.
- Untracked files.
- Current branch and whether it's up-to-date with the remote branch.

```
dayashree@DESKTOP-J19GU2A MINGW64 ~/project-3 (master)
$ git status
On branch master

No commits yet

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

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

dayashree@DESKTOP-J19GU2A MINGW64 ~/project-3 (master)
$ |
```

## 12. git add

- *Purpose:* Stage a file to prepare it for commit.
- *Usage:* git add filename.

```
dayashree@DESKTOP-J19GU2A MINGW64 ~/project-3 (master)
$ git add .
warning: in the working copy of 'apple.txt', LF will be replaced by CRLF the next time Git touches it

dayashree@DESKTOP-J19GU2A MINGW64 ~/project-3 (master)
$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   apple.txt
```

## 13. git commit

- *Purpose:* Commit changes to the repository with a message.
- *Usage:* git commit -m "Message".

```

dayashree@DESKTOP-J19GU2A MINGW64 ~/project-3 (master)
$ git commit -m "first commit"
[master (root-commit) 5d12280] first commit
1 file changed, 1 insertion(+)
create mode 100644 apple.txt

dayashree@DESKTOP-J19GU2A MINGW64 ~/project-3 (master)
$ |

```

#### 14. git config

- *Purpose:* Configure Git with username and email.
- *Usage:*
  - `git config --global user.mail "email@example.com"`
  - `git config --global user.name "Your Name"`

#### 15. git log

- *Purpose:* View commit history.

```

dayashree@DESKTOP-J19GU2A MINGW64 ~/project-3 (master)
$ git log
commit 16d95c1063df26a6948cc4cc21acba21ae8489de (HEAD -> master, test)
Author: Dayashree.S <dayashree.s@s.amity.edu>
Date:   Fri Feb 14 00:19:40 2025 +0530

    second modification

commit 5d122803f4ccf1724e937a49879e198bda81f36a
Author: Dayashree.S <dayashree.s@s.amity.edu>
Date:   Fri Feb 14 00:16:16 2025 +0530

    first commit

```

#### 15. git diff

- *Purpose:* Shows the changes made in the working directory since the last commit.

```

dayashree@DESKTOP-J19GU2A MINGW64 ~/project-3 (master)
$ git diff 5d12280 16d95c1
diff --git a/apple.txt b/apple.txt
index 3b18e51..77b054a 100644
--- a/apple.txt
+++ b/apple.txt
@@ -1,3 @@
 hello world
+
+second change

dayashree@DESKTOP-J19GU2A MINGW64 ~/project-3 (master)
$ |

```

#### 16. git branch <branch name>

- *Purpose:* Creates a new branch.

```

dayashree@DESKTOP-J19GU2A MINGW64 ~/project-3 (master)
$ git branch test

```

## 17. git branch

- *Purpose:* Lists all branches in the repository.

```
dayashree@DESKTOP-J19GU2A MINGW64 ~/project-3 (master)
$ git branch
* master
  test

dayashree@DESKTOP-J19GU2A MINGW64 ~/project-3 (master)
$
```

## 18. git log --oneline

- *Purpose:* View a concise commit history.

```
dayashree@DESKTOP-J19GU2A MINGW64 ~/project-3 (master)
$ git log --oneline
16d95c1 (HEAD -> master) second modification
5d12280 first commit

dayashree@DESKTOP-J19GU2A MINGW64 ~/project-3 (master)
$ |
```

## 19. git checkout <branch name>

- *Purpose:* Switch to the master branch.

```
dayashree@DESKTOP-J19GU2A MINGW64 ~/project-3 (master)
$ git checkout test
Switched to branch 'test'

dayashree@DESKTOP-J19GU2A MINGW64 ~/project-3 (test)
$ |
```

## 20. git remote add origin " "

- *Purpose:* Add a new remote repository named origin with the provided ,it would typically be git remote add origin <repository\_url>.

## 21. git remote

- *Purpose:* List the names of remote repositories you have configured.

```
dayashree@DESKTOP-J19GU2A MINGW64 ~/python-1 (master)
$ mk dir
bash: mk: command not found

dayashree@DESKTOP-J19GU2A MINGW64 ~/python-1 (master)
$ ls
add.py  if-else.py  if.py

dayashree@DESKTOP-J19GU2A MINGW64 ~/python-1 (master)
$ git remote add origin https://github.com/daya321217/test.git

dayashree@DESKTOP-J19GU2A MINGW64 ~/python-1 (master)
$ git remote
origin
```

## 22. git push -u origin master



- Purpose: Push the master branch from your local repository to the origin remote repository and set origin as the upstream for the master branch (so subsequent git pull and git push commands can be used without specifying the remote and branch).

```
dayashree@DESKTOP-J19GU2A MINGW64 ~/python-1 (master)
$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   add.py
        new file:   if-else.py
        new file:   if.py

dayashree@DESKTOP-J19GU2A MINGW64 ~/python-1 (master)
$ git add .

dayashree@DESKTOP-J19GU2A MINGW64 ~/python-1 (master)
$ git commit -m "Final commit"
[master (root-commit) 7a95277] Final commit
 3 files changed, 15 insertions(+)
 create mode 100644 add.py
 create mode 100644 if-else.py
 create mode 100644 if.py

dayashree@DESKTOP-J19GU2A MINGW64 ~/python-1 (master)
$ git push -u origin master
info: please complete authentication in your browser...
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 8 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (5/5), 437 bytes | 437.00 KiB/s, done.
Total 5 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/daya321217/test.git
 * [new branch]      master -> master
branch 'master' set up to track 'origin/master'.

dayashree@DESKTOP-J19GU2A MINGW64 ~/python-1 (master)
$
```

## GIT MERGE

- **Production Code** is the finalized, deployable version of software used in a live environment.
- **Branches** in Git allow multiple versions of code to exist simultaneously, starting from the project's inception.
- **Merging** happens when different branches need to be combined. Git does not automatically decide what to keep—it requires manual resolution.
- **Conflict Resolution:** If conflicts arise, Git will halt the merge. A merge tool helps resolve differences.

```
dayashree@DESKTOP-J19GU2A MINGW64 ~/documents
$ mkdir merge_project

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents
$ cd merge_project

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/merge_project
$ git init
Initialized empty Git repository in C:/Users/dayashree/Documents/merge_project/.git/

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/merge_project (master)
$ echo "Initial content for main branch." > index.html

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/merge_project (master)
$ git status
On branch master

No commits yet

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

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

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/merge_project (master)
$ ls
index.html
```

```
dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/merge_project (master)
$ cat index.html
Initial content for main branch.

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/merge_project (master)
$ git add index.html
warning: in the working copy of 'index.html', LF will be replaced by CRLF the next time Git touches it
```

```
dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/merge_project (master)
$ git add index.html

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/merge_project (master)
$ git commit -m "Initial commit on main branch"
[master (root-commit) 88c532f] Initial commit on main branch
1 file changed, 1 insertion(+)
create mode 100644 index.html

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/merge_project (master)
$ git branch feature branch
fatal: not a valid object name: 'branch'
```

```
dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/merge_project (master)
$ git branch feature-branch

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/merge_project (master)
$ git checkout feature_branch
error: pathspec 'feature_branch' did not match any file(s) known to git
```

```
dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/merge_project (master)
$ git checkout feature-branch
Switched to branch 'feature-branch'
```

```

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/merge_project (feature-branch)
$ git status
On branch feature-branch
nothing to commit, working tree clean

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/merge_project (feature-branch)
$ echo "Adding a new feature." >> index.html

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/merge_project (feature-branch)
$ echo "Feature-specific logic" > script.js

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/merge_project (feature-branch)
$ git add index.html script.js
warning: in the working copy of 'index.html', LF will be replaced by CRLF the
warning: in the working copy of 'script.js', LF will be replaced by CRLF the n

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/merge_project (feature-branch)
$ git add index.html script.js

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/merge_project (feature-branch)
$ git commit -m "Implemented new feature and script"
[feature-branch 4e09ac8] Implemented new feature and script
2 files changed, 2 insertions(+)
create mode 100644 script.js

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/merge_project (feature-branch)
$ git checkout master
Switched to branch 'master'

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/merge_project (master)
$ ls
index.html

```

```

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/merge_project (master)
$ git merge feature-branch
Updating 88c532f..4e09ac8
Fast-forward
 index.html | 1 +
 script.js  | 1 +
2 files changed, 2 insertions(+)
create mode 100644 script.js

```

## Handle a simple merge conflict

```

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/merge_project (master)
$ echo "Master's line 1" > conflict_file.txt

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/merge_project (master)
$ git add conflict_file.txt
warning: in the working copy of 'conflict_file.txt', LF will be replaced by

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/merge_project (master)
$ git add conflict_file.txt

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/merge_project (master)
$ git commit -m "Added conflict file on master"
[master 4c88e31] Added conflict file on master
1 file changed, 1 insertion(+)
create mode 100644 conflict_file.txt

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/merge_project (master)
$ git checkout feature-branch
Switched to branch 'feature-branch'

```

```
dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/merge_project (feature-branch)
$ echo "Feature's line 1" > conflict_file.txt

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/merge_project (feature-branch)
$ git add conflict_file.txt
warning: in the working copy of 'conflict_file.txt', LF will be replaced by CRLF
$ git add conflict_file.txt
```

```
dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/merge_project (feature-branch)
$ git commit -m "Modified conflict file on feature"
[feature-branch 52b6bca] Modified conflict file on feature
1 file changed, 1 insertion(+)
create mode 100644 conflict_file.txt
```

```
dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/merge_project (feature-branch)
$ git checkout master
Switched to branch 'master'
```

```
dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/merge_project (master)
$ git merge feature-branch
Auto-merging conflict_file.txt
CONFLICT (add/add): Merge conflict in conflict_file.txt
Automatic merge failed; fix conflicts and then commit the result.
```

```
dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/merge_project (master)
$ vi conflict_file.txt
```

## Vim editor

```
|<<<<<< HEAD
Master's line 1
=====
Feature's line 1
>>>>>> feature-branch
~
~
~
```

```
dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/merge_project (master|MERGING)
$ git add conflict_file.txt

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/merge_project (master|MERGING)
$ git status
On branch master
All conflicts fixed but you are still merging.
(use "git commit" to conclude merge)

Changes to be committed:
  modified:   conflict_file.txt

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/merge_project (master|MERGING)
$ git commit -m "Resolved merge conflict in conflict_file.txt"
[master 646efe4] Resolved merge conflict in conflict_file.txt
```

# GIT IGNORE

```
dayashree@DESKTOP-J19GU2A MINGW64 ~/documents
$ mkdir gitignore_project

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents
$ cd gitignore_project

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/gitignore_project
$ git init
Initialized empty Git repository in C:/Users/dayashree/Documents/gitignore_project/.git/

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/gitignore_project (master)
$ echo "My awesome code." > app.js

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/gitignore_project (master)
$ echo "Temp data" > temp.log
```

```
dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/gitignore_project (master)
$ mkdir build

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/gitignore_project (master)
$ echo "Compiled output" > build/output.exe

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/gitignore_project (master)
$ echo "API_KEY=12345" > config.env

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/gitignore_project (master)
$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    app.js
    build/
    config.env
    temp.log

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

```
dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/gitignore_project (master)
$ vi .gitignore
```

```
# Ignore all files ending with .log
*.log

# Ignore the entire 'build' directory
build/

# Ignore a specific sensitive file
config.env

# You can also ignore by specific file name anywhere
my_temp_file.tmp
```

These are the files to be ignored

```
dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/gitignore_project (master)
$ git status
On branch master

No commits yet

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

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

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/gitignore_project (master)
$ git add .
warning: in the working copy of '.gitignore', LF will be replaced by CRLF the
warning: in the working copy of 'app.js', LF will be replaced by CRLF the next

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/gitignore_project (master)
$ git add .

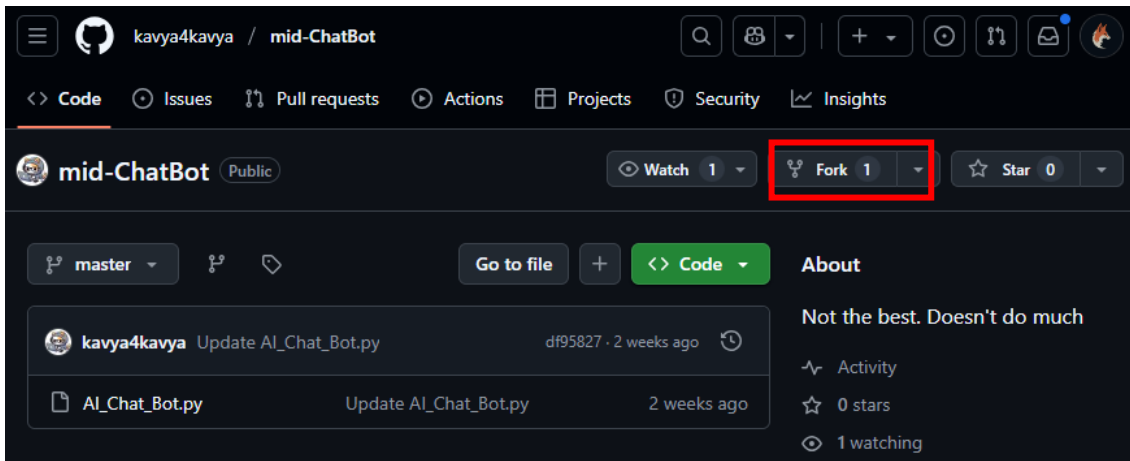
dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/gitignore_project (master)
$ git commit -m "Initial commit with .gitignore and app.js"
[master (root-commit) 96d0de5] Initial commit with .gitignore and app.js
2 files changed, 9 insertions(+)
 create mode 100644 .gitignore
 create mode 100644 app.js

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/gitignore_project (master)
$ echo "Another temp log which should be ignored" > new_temp.log

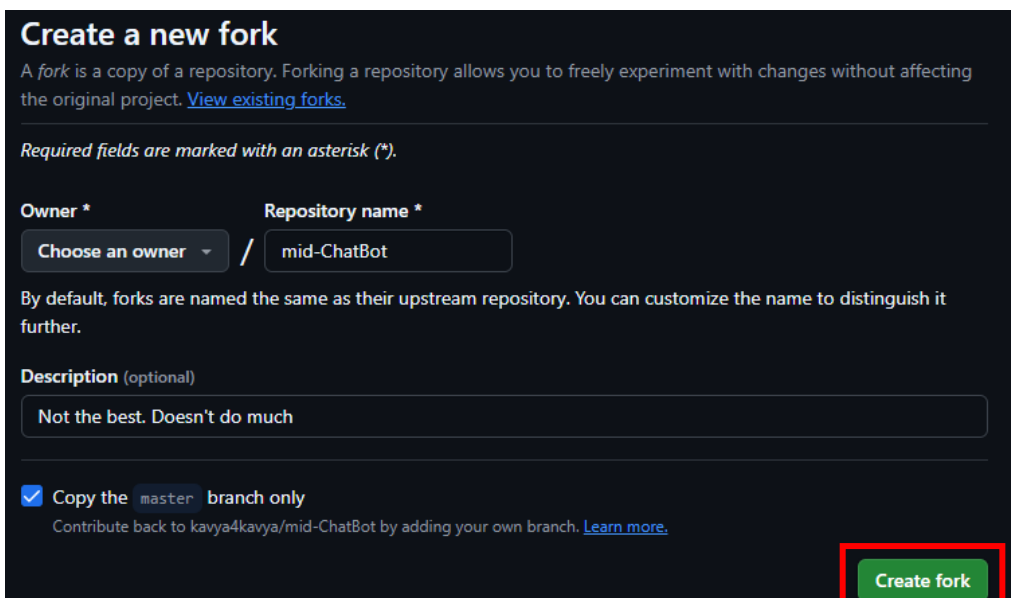
dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/gitignore_project (master)
$ git status
On branch master
nothing to commit, working tree clean
```

# GIT CLONE

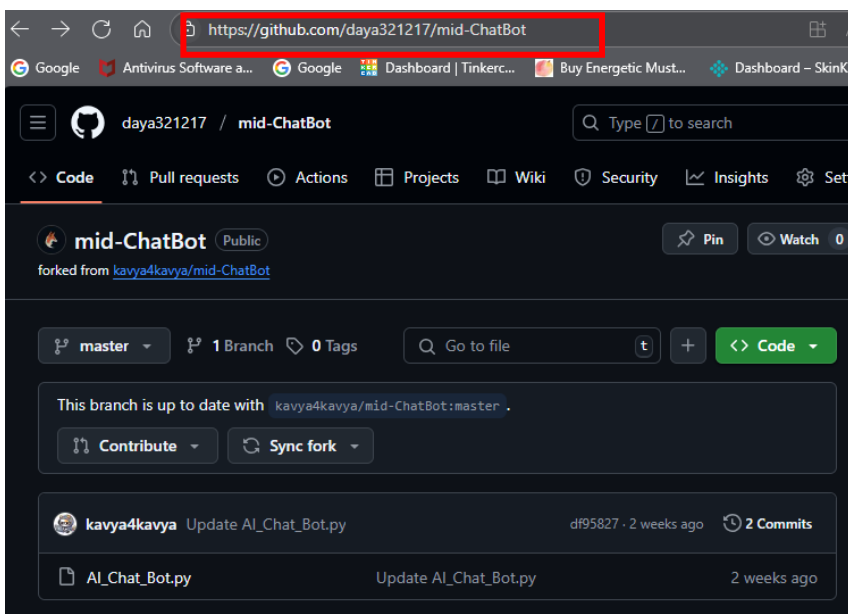
Navigate a project that you are interested in and fork it



Fork it



Copy the URL



Go to git bash and paste it with the following commands

```
dayashree@DESKTOP-J19GU2A MINGW64 ~/documents
$ mkdir git_clone_project
```

```
dayashree@DESKTOP-J19GU2A MINGW64 ~/documents
$ cd git_clone_project
```

```
dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/git_clone_project
$ git init
Initialized empty Git repository in C:/Users/dayashree/Documents/git_clone_project/.git/
```

```
dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/git_clone_project
$ git clone https://github.com/daya321217/mid-ChatBot
Cloning into 'mid-ChatBot'...
remote: Enumerating objects: 6, done.
remote: Counting objects: 100% (6/6), done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 6 (delta 1), reused 2 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (6/6), done.
Resolving deltas: 100% (1/1), done.
```

```
dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/git_clone_project (master)
$ ls
mid-ChatBot/
```

```
dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/git_clone_project (master)
$ cd mid-ChatBot

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/git_clone_project/mid-ChatBot (master)
$ ls
AI_Chat_Bot.py
```

Now the file can be accessed



# GIT CHEAT SHEET

## INSTALLATION & GUIs

With platform specific installers for Git, GitHub also provides the ease of staying up-to-date with the latest releases of the command line tool while providing a graphical user interface for day-to-day interaction, review, and repository synchronization.

### GitHub for Windows

<https://windows.github.com>

### GitHub for Mac

<https://mac.github.com>

For Linux and Solaris platforms, the latest release is available on the official Git web site.

### Git for All Platforms

<http://git-scm.com>

## SETUP

Configuring user information used across all local repositories

```
git config --global user.name "[firstname lastname]"
```

set a name that is identifiable for credit when review version history

```
git config --global user.email "[valid-email]"
```

set an email address that will be associated with each history marker

```
git config --global color.ui auto
```

set automatic command line coloring for Git for easy reviewing

## SETUP & INIT

Configuring user information, initializing and cloning repositories

```
git init
```

initialize an existing directory as a Git repository

```
git clone [url]
```

retrieve an entire repository from a hosted location via URL

## STAGE & SNAPSHOT

Working with snapshots and the Git staging area

```
git status
```

show modified files in working directory, staged for your next commit

```
git add [file]
```

add a file as it looks now to your next commit (stage)

```
git reset [file]
```

unstage a file while retaining the changes in working directory

```
git diff
```

diff of what is changed but not staged

```
git diff --staged
```

diff of what is staged but not yet committed

```
git commit -m "[descriptive message]"
```

commit your staged content as a new commit snapshot

## BRANCH & MERGE

Isolating work in branches, changing context, and integrating changes

```
git branch
```

list your branches. a \* will appear next to the currently active branch

```
git branch [branch-name]
```

create a new branch at the current commit

```
git checkout
```

switch to another branch and check it out into your working directory

```
git merge [branch]
```

merge the specified branch's history into the current one

```
git log
```

show all commits in the current branch's history

## INSPECT & COMPARE

Examining logs, diffs and object information

<b>git log</b>
show the commit history for the currently active branch
<b>git log branchB..branchA</b>
show the commits on branchA that are not on branchB
<b>git log --follow [file]</b>
show the commits that changed file, even across renames
<b>git diff branchB...branchA</b>
show the diff of what is in branchA that is not in branchB
<b>git show [SHA]</b>
show any object in Git in human-readable format

## TRACKING PATH CHANGES

Versioning file removes and path changes

<b>git rm [file]</b>
delete the file from project and stage the removal for commit
<b>git mv [existing-path] [new-path]</b>
change an existing file path and stage the move
<b>git log --stat -M</b>
show all commit logs with indication of any paths that moved

## IGNORING PATTERNS

Preventing unintentional staging or committing of files

<b>logs/ *.notes pattern*/</b>
Save a file with desired patterns as .gitignore with either direct string matches or wildcard globs.
<b>git config --global core.excludesfile [file]</b>
system wide ignore pattern for all local repositories

## SHARE & UPDATE

Retrieving updates from another repository and updating local repos

<b>git remote add [alias] [url]</b>
add a git URL as an alias
<b>git fetch [alias]</b>
fetch down all the branches from that Git remote
<b>git merge [alias]/[branch]</b>
merge a remote branch into your current branch to bring it up to date
<b>git push [alias] [branch]</b>
Transmit local branch commits to the remote repository branch
<b>git pull</b>
fetch and merge any commits from the tracking remote branch

## REWRITE HISTORY

Rewriting branches, updating commits and clearing history

<b>git rebase [branch]</b>
apply any commits of current branch ahead of specified one
<b>git reset --hard [commit]</b>
clear staging area, rewrite working tree from specified commit

## TEMPORARY COMMITS

Temporarily store modified, tracked files in order to change branches

<b>git stash</b>
Save modified and staged changes
<b>git stash list</b>
list stack-order of stashed file changes
<b>git stash pop</b>
write working from top of stash stack
<b>git stash drop</b>
discard the changes from top of stash stack

# SCM Project

The project was to make a repository in GitHub, make 3 branches and merge it with the main branch and access all 4 team-mate's repositories, fork it, clone it, make some changes and merge them.

First, make your own repositories and make 3 branches and add files and merge with the main branch.

Then create a dev branch where the team members will contribute.

```
dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 DAYS
$ cd SCM_project11

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 DAYS/SCM_project11
$ git init
Initialized empty Git repository in C:/Users/dayashree/Documents/100 DAYS/SCM_project11/.git/

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 DAYS/SCM_project11 (master)
$ touch Home.html

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 DAYS/SCM_project11 (master)
$ git add Home.html

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 DAYS/SCM_project11 (master)
$ git commit -m "Initial commit on master"
[master (root-commit) fe5af14] Initial commit on master
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 Home.html

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 DAYS/SCM_project11 (master)
$ git checkout master
Already on 'master'

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 DAYS/SCM_project11 (master)
$ git branch About

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 DAYS/SCM_project11 (master)
$ gto branch Community
bash: gto: command not found

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 DAYS/SCM_project11 (master)
$ git branch Community

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 DAYS/SCM_project11 (master)
$ git branch Home

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 DAYS/SCM_project11 (master)
$ git checkout Home
Switched to branch 'Home'

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 DAYS/SCM_project11 (Home)
$ ls
Home.html

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 DAYS/SCM_project11 (Home)
$ vi Home.html

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 DAYS/SCM_project11 (Home)
$ vi Home.css

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 DAYS/SCM_project11 (Home)
$ git add Home.css Home.html
warning: in the working copy of 'Home.html', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'Home.css', LF will be replaced by CRLF the next time Git touches it

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 DAYS/SCM_project11 (Home)
$ git add Home.css Home.html

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 DAYS/SCM_project11 (Home)
$ git commit -m " Home page commit on home branch"
[Home 0be08b0] Home page commit on home branch
2 files changed, 887 insertions(+)
create mode 100644 Home.css
```

```

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 DAYS/SCM_project11 (Home)
$ git checkout About
Switched to branch 'About'

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 DAYS/SCM_project11 (About)
$ ls
Home.html

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 DAYS/SCM_project11 (About)
$ vi About.html

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 DAYS/SCM_project11 (About)
$ vi About.css

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 DAYS/SCM_project11 (About)
$ git add About.css About.html
warning: in the working copy of 'About.css', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'About.html', LF will be replaced by CRLF the next time Git touches it

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 DAYS/SCM_project11 (About)
$ git add About.css About.html

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 DAYS/SCM_project11 (About)
$ git commit -m "About page commit on About branch"
[About 37738ac] About page commit on About branch
2 files changed, 214 insertions(+)
create mode 100644 About.css
create mode 100644 About.html

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 DAYS/SCM_project11 (About)
$ git branch
* About
  Community
  Home
  master

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 DAYS/SCM_project11 (About)
$ git checkout Community
Switched to branch 'Community'

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 DAYS/SCM_project11 (Community)
$ vi Community.html

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 DAYS/SCM_project11 (Community)
$ vi Community.css

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 DAYS/SCM_project11 (Community)
$ git add Community.css Community.html
warning: in the working copy of 'Community.css', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'Community.html', LF will be replaced by CRLF the next time Git touches it

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 DAYS/SCM_project11 (Community)
$ git add Community.css Community.html

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 DAYS/SCM_project11 (Community)
$ git commit -m "Commit on Community branch"
[Community aa234ee] Commit on Community branch
2 files changed, 216 insertions(+)
create mode 100644 Community.css
create mode 100644 Community.html

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 DAYS/SCM_project11 (Community)
$ git checkout master
Switched to branch 'master'

```

From the Above figure we can see that the three branches are created with their sub files and required commits are made.

Now we have to merge it into main (here master) branch.

```

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 DAYS/SCM_project11 (Community)
$ git checkout master
Switched to branch 'master'

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 DAYS/SCM_project11 (master)
$ git branch
  About
  Community
  Home
* master

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 DAYS/SCM_project11 (master)
$ git merge About --no-ff -m "Merge About to master"
Merge made by the 'ort' strategy.
 About.css | 137 +++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
 About.html | 77 +++++++++++++++++++++++++++++++++++++
 2 files changed, 214 insertions(+)
 create mode 100644 About.css
 create mode 100644 About.html

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 DAYS/SCM_project11 (master)
$ git merge Community --no-ff -m "Merge Community to master"
Merge made by the 'ort' strategy.
 Community.css | 131 +++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
 Community.html | 85 +++++++++++++++++++++++++++++++++++++
 2 files changed, 216 insertions(+)
 create mode 100644 Community.css
 create mode 100644 Community.html

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 DAYS/SCM_project11 (master)
$ git merge Home --no-ff -m "Merge Home to master"
Merge made by the 'ort' strategy.
 Home.css | 829 +++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
 Home.html | 58 +++++
 2 files changed, 887 insertions(+)
 create mode 100644 Home.css

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 DAYS/SCM_project11 (master)
$ git checkout master
Already on 'master'

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 DAYS/SCM_project11 (master)
$ ls
About.css About.html Community.css Community.html Home.css Home.html

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 DAYS/SCM_project11 (master)
$ git branch dev

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 DAYS/SCM_project11 (master)
$ git remote add origin https://github.com/daya321217/SCM_project.git

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 DAYS/SCM_project11 (master)
$ git branch -M main

```

merge the branches into main one by one and create a dev branch from master which acts like a copy of master as well as a branch for pull requests merge.

```

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 DAYS/SCM_project11 (main)
$ git push -u origin main
Enumerating objects: 20, done.
Counting objects: 100% (20/20), done.
Delta compression using up to 8 threads
Compressing objects: 100% (18/18), done.
Writing objects: 100% (20/20), 6.25 KiB | 1.56 MiB/s, done.
Total 20 (delta 8), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (8/8), done.
To https://github.com/daya321217/SCM_project.git
 * [new branch]      main -> main
branch 'main' set up to track 'origin/main'.

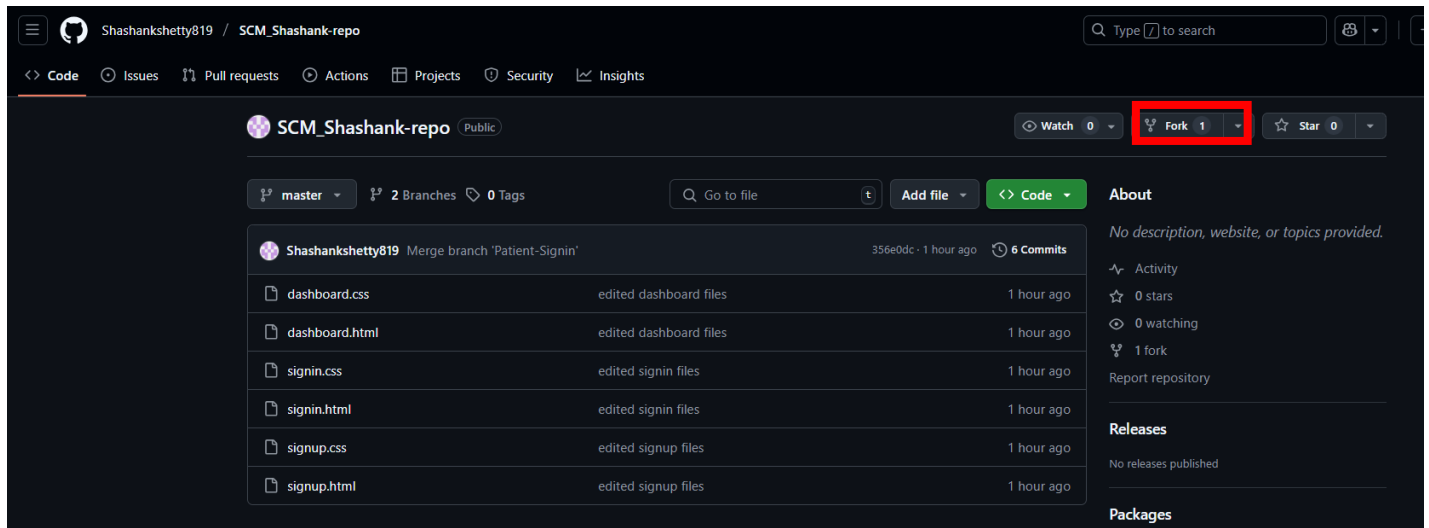
dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 DAYS/SCM_project11 (main)
$ git push -u origin dev
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
remote:
remote: Create a pull request for 'dev' on GitHub by visiting:
remote:   https://github.com/daya321217/SCM_project/pull/new/dev
remote:
To https://github.com/daya321217/SCM_project.git
 * [new branch]      dev -> dev
branch 'dev' set up to track 'origin/dev'.

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 DAYS/SCM_project11 (main)
$ git branch
  About
  Community
  Home
  dev
* main

```

Push it into Github from where the team members shall access the code

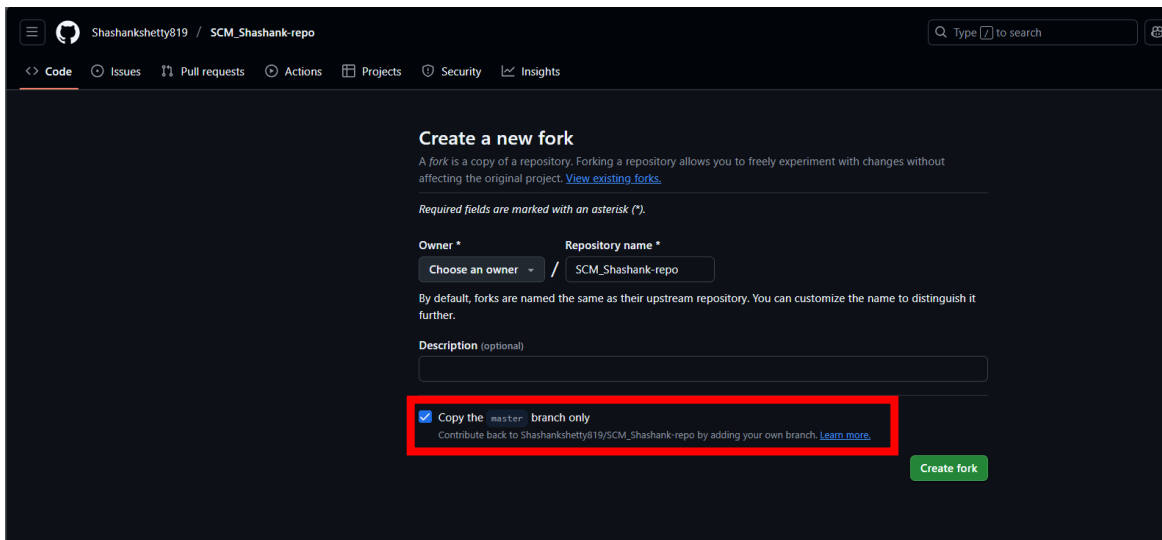
## FORKING AND PULL REQUESTS



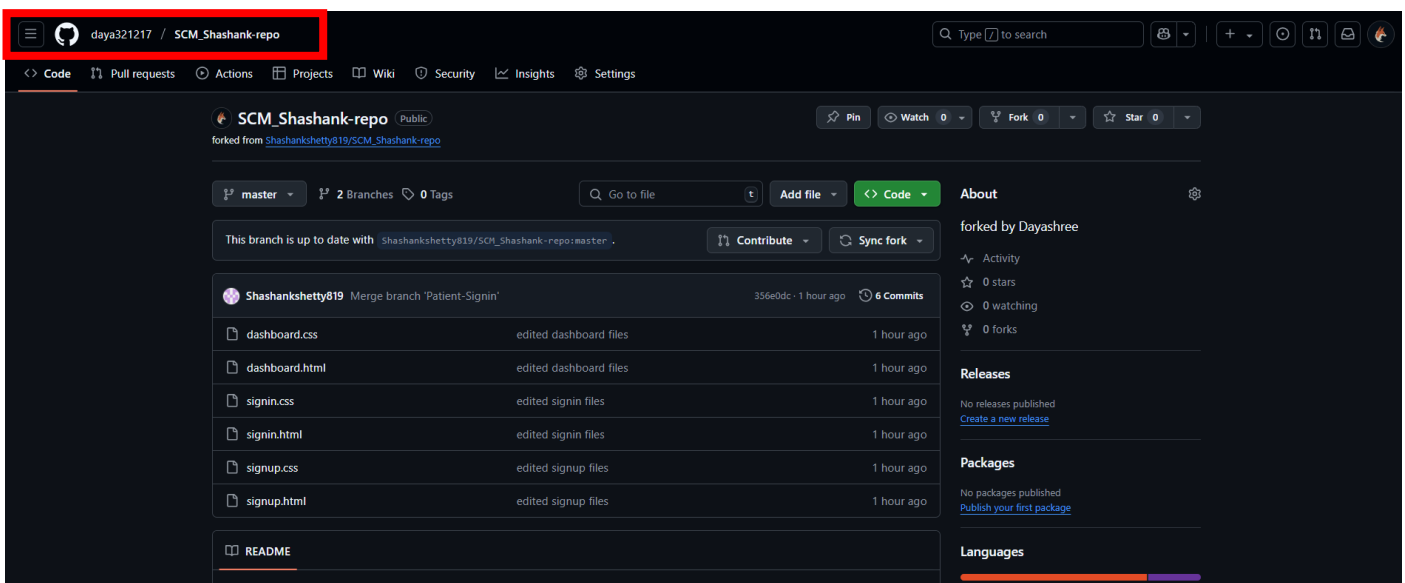
The screenshot shows the GitHub interface for a repository named 'SCM\_Shashank-repo' by user 'Shashankshetty819'. The repository is public and has 2 branches and 0 tags. The 'Fork' button is highlighted with a red box, indicating the next step in the process. The repository contains several files: dashboard.css, dashboard.html, signin.css, signin.html, signup.css, and signup.html, all edited 1 hour ago. The 'About' section on the right indicates no description, website, or topics are provided.

Go to your team member's repository and fork it to your own GitHub account.

**Important:** While forking, make sure to **uncheck** the option that says “Copy the master branch only” — we need access to all branches, especially the dev branch, for pushing our changes.



Now its forked as one of your repositories



```
dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 DAYS
$ git clone https://github.com/daya321217/SCM_Shashank-repo
Cloning into 'SCM_Shashank-repo'...
remote: Enumerating objects: 24, done.
remote: Counting objects: 100% (24/24), done.
remote: Compressing objects: 100% (10/10), done.
remote: Total 24 (delta 14), reused 24 (delta 14), pack-reused 0 (from 0)
Receiving objects: 100% (24/24), 11.18 KiB | 1.60 MiB/s, done.
Resolving deltas: 100% (14/14), done.
```

Clone it into your local machine

And make edits wherever required.



```

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 DAYS
$ cd SCM_Shashank-repo

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 DAYS/SCM_Shashank-repo (master)
$ ls
dashboard.css  dashboard.html  signin.css  signin.html  signup.css  signup.html

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 DAYS/SCM_Shashank-repo (master)
$ git checkout -b new-feature-dashboard-branch
Switched to a new branch 'new-feature-dashboard-branch'

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 DAYS/SCM_Shashank-repo (new-feature-dashboard-branch)
$ ls
dashboard.css  dashboard.html  signin.css  signin.html  signup.css  signup.html

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 DAYS/SCM_Shashank-repo (new-feature-dashboard-branch)
$ vi dashboard.css

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 DAYS/SCM_Shashank-repo (new-feature-dashboard-branch)
$ vi dashboard.html

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 DAYS/SCM_Shashank-repo (new-feature-dashboard-branch)
$ git add dashboard.css dashboard.html

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 DAYS/SCM_Shashank-repo (new-feature-dashboard-branch)
$ git commit -m "New dashboard Feature added"
[new-feature-dashboard-branch 35348ec] New dashboard Feature added
2 files changed, 239 insertions(+), 451 deletions(-)

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 DAYS/SCM_Shashank-repo (new-feature-dashboard-branch)
$ git push origin new-feature-dashboard-branch
Enumerating objects: 7, done.
Counting objects: 100% (7/7), done.
Delta compression using up to 8 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (4/4), 2.60 KiB | 2.60 MiB/s, done.
Total 4 (delta 3), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (3/3), completed with 3 local objects.
remote:
remote: Create a pull request for 'new-feature-dashboard-branch' on GitHub by visiting:
remote:   https://github.com/daya321217/SCM_Shashank-repo/pull/new/new-feature-dashboard-branch
remote:
To https://github.com/daya321217/SCM_Shashank-repo
 * [new branch]      new-feature-dashboard-branch -> new-feature-dashboard-branch

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 DAYS/SCM_Shashank-repo (new-feature-dashboard-branch)
$ !

```

We've cloned the "SCM\_Shashank-repo." For the edits, a new branch was created from the main branch, and all modifications were made there. These changes were then added, committed, and the specific branch was pushed to GitHub.





## 2nd Teammate

The screenshot shows the GitHub interface for the repository 'SCM\_Koushik' by user 'koushik-0211'. The repository is public and has 0 watches, 0 forks, and 0 stars. The main branch is 'main'. A recent commit 'Merge branch 'Sign-up'' is shown with a list of files: Appointment.css, Appointment.html, Sign-in.css, Sign-in.html, Sign-up.css, and Sign-up.html, all initialized 11 hours ago. The right sidebar contains sections for 'About' (no description), 'Activity' (0 stars, 0 watching, 0 forks), 'Report repository', and 'Releases' (no releases published).

Navigation: Code, Issues, Pull requests, Actions, Projects, Security, Insights

Repository: SCM\_Koushik (Public)

Buttons: Watch 0, Fork 0, Star 0

Branch: main

Commit: koushik-0211 Merge branch 'Sign-up' 9801df1 · 10 hours ago

File	Description	Time
Appointment.css	Appointment code insertion	11 hours ago
Appointment.html	Appointment code insertion	11 hours ago
Sign-in.css	Sign in page initialized	11 hours ago
Sign-in.html	Sign in page initialized	11 hours ago
Sign-up.css	Sign up page initialized	11 hours ago
Sign-up.html	Sign up page initialized	11 hours ago

About: No description, website, or topics provided.

Activity: 0 stars, 0 watching, 0 forks

Releases: No releases published

Navigate the repository in teammates account and fork it

The screenshot shows the 'Create a new fork' page for the repository 'SCM\_Koushik'. It explains that a fork is a copy of a repository and allows for experimentation without affecting the original project. The form includes fields for 'Owner' (daya32127) and 'Repository name' (SCM\_Koushik), with a confirmation message 'SCM\_Koushik is available.' The 'Description' field is optional and contains 'forked by Dayashree'. There is a checkbox for 'Copy the main branch only' and a note about contributing back. A green 'Create fork' button is at the bottom right.

### Create a new fork

A fork is a copy of a repository. Forking a repository allows you to freely experiment with changes without affecting the original project.

Required fields are marked with an asterisk (\*).

Owner \* Repository name \*

daya32127 / SCM\_Koushik

✔ SCM\_Koushik is available.

By default, forks are named the same as their upstream repository. You can customize the name to distinguish it further.

Description (optional)

forked by Dayashree

☐ Copy the main branch only

Contribute back to koushik-0211/SCM\_Koushik by adding your own branch. [Learn more.](#)

*i* You are creating a fork in your personal account.

Create fork

```

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 Days
$ git clone https://github.com/daya321217/SCM_Koushik
Cloning into 'SCM_Koushik'...
remote: Enumerating objects: 19, done.
remote: Counting objects: 100% (19/19), done.
remote: Compressing objects: 100% (9/9), done.
remote: Total 19 (delta 9), reused 19 (delta 9), pack-reused 0 (from 0)
Receiving objects: 100% (19/19), done.
Resolving deltas: 100% (9/9), done.

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 Days
$ ls
'04 may'/   'New folder'/   SCM_Shashank-repo/   SCM_project11/   'day2 - cooking recipe portal'/   'day9-sales Forecasting'/
Games/      SCM_Koushik/     SCM_dayashree/       'day1 - Pixel art'/   'day4 - Python project'/         flask/

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 Days
$ cd SCM_Koushik

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 Days/SCM_Koushik (main)
$ ls
Appointment.css  Appointment.html  Sign-in.css  Sign-in.html  Sign-up.css  Sign-up.html

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 Days/SCM_Koushik (main)
$ git checkout -b new-feature-Appointment-branch
Switched to a new branch 'new-feature-Appointment-branch'

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 Days/SCM_Koushik (new-feature-Appointment-branch)
$ ls
Appointment.css  Appointment.html  Sign-in.css  Sign-in.html  Sign-up.css  Sign-up.html

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 Days/SCM_Koushik (new-feature-Appointment-branch)
$ vi Appointment.css

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 Days/SCM_Koushik (new-feature-Appointment-branch)
$ git add Appointment.css

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 Days/SCM_Koushik (new-feature-Appointment-branch)
$ git commit -m "Final Appointment.css file edited"
[new-feature-Appointment-branch 2ec9259] Final Appointment.css file edited
1 file changed, 54 insertions(+)

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 Days/SCM_Koushik (new-feature-Appointment-branch)
$ vi Appointment.html

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 Days/SCM_Koushik (new-feature-Appointment-branch)
$ git add Appointment.html

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 Days/SCM_Koushik (new-feature-Appointment-branch)
$ git commit -m "Appointment.html edited"
[new-feature-Appointment-branch 25eab73] Appointment.html edited
1 file changed, 51 insertions(+), 1 deletion(-)

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 Days/SCM_Koushik (new-feature-Appointment-branch)
$ git add .

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 Days/SCM_Koushik (new-feature-Appointment-branch)
$ git commit -m "two Apointment edited in new feature Appointment branch"
On branch new-feature-Appointment-branch
nothing to commit, working tree clean

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 Days/SCM_Koushik (new-feature-Appointment-branch)
$ git push origin new-feature-Appointment-branch
Enumerating objects: 9, done.
Counting objects: 100% (9/9), done.
Delta compression using up to 8 threads
Compressing objects: 100% (6/6), done.
Writing objects: 100% (6/6), 2.55 KiB | 523.00 KiB/s, done.
Total 6 (delta 3), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (3/3), completed with 2 local objects.

```

Once done check for pull requests on your own repository where others have contributed on your code

Accept the pull request send the teammates, got to your git bash local repo and checkout to dev branch  
git pull origin dev – will pull all the edited/ contributions from other teammates to your local machine

Once done it shows the amt of files edited  
later checkout to main branch and merge dev into main

And finally push main.

```

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 Days/SCM_project11 (dev)
$ git checkout dev
Already on 'dev'
Your branch is up to date with 'origin/dev'.

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 Days/SCM_project11 (dev)
$ git pull origin dev
remote: Enumerating objects: 12, done.
remote: Counting objects: 100% (10/10), done.
remote: Compressing objects: 100% (6/6), done.
remote: Total 6 (delta 1), reused 3 (delta 0), pack-reused 0 (from 0)
Unpacking objects: 100% (6/6), 5.71 KiB | 22.00 KiB/s, done.
From https://github.com/daya321217/SCM_project
* branch      dev      -> FETCH_HEAD
   07fde49..4db2857 dev    -> origin/dev
Updating 07fde49..4db2857
Fast-forward
 About.css | 160 +++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
 About.html | 70  +++++++++++++++++++++++++++++++++++++-
 2 files changed, 229 insertions(+), 1 deletion(-)

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 Days/SCM_project11 (dev)
$ git log --oneline --graph --all
* 4db2857 (HEAD -> dev, origin/dev) Merge pull request #2 from Shashankshetty819/feature-About
|
| * 25a9393 About page edits done
| * | 07fde49 Merge pull request #1 from koushik-0211/new-Community
|/|
|/|
| * 9ad65ae Community web page edited
|/
| * 8f2e1a8 (origin/main, main) Merge Home to master
|/
| * 0be08b0 (Home) Home page commit on home branch
| * | 2b2fe2e Merge Community to master
|/|
| * | aa234ee (Community) Commit on Community branch
|/|
| * | b7ef798 Merge About to master
|/|
| * 37738ac (About) About page commit on About branch
|/
* fe5af14 Initial commit on master

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 Days/SCM_project11 (dev)
$ git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 Days/SCM_project11 (main)
$ git merge dev
Updating 8f2e1a8..4db2857
Fast-forward
 About.css | 160 +++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
 About.html | 70  +++++++++++++++++++++++++++++++++++++-
 Community.css | 41 +++++
 Community.html | 14 +++++
 4 files changed, 284 insertions(+), 1 deletion(-)

dayashree@DESKTOP-J19GU2A MINGW64 ~/documents/100 Days/SCM_project11 (main)
$ git push origin main
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)

```

Once this is done the project is good to go once the main branch is pushed

This is the Venn diagram of the initial branching system

