



Source Code Management

Lab Record

NAME: Abhipsha Jena

ENROLLMENT NUMBER: A866175124002

FACULTY: Dr. Monit Kapoor

COURSE CODE: CSE2015

SLOT: L15+L16

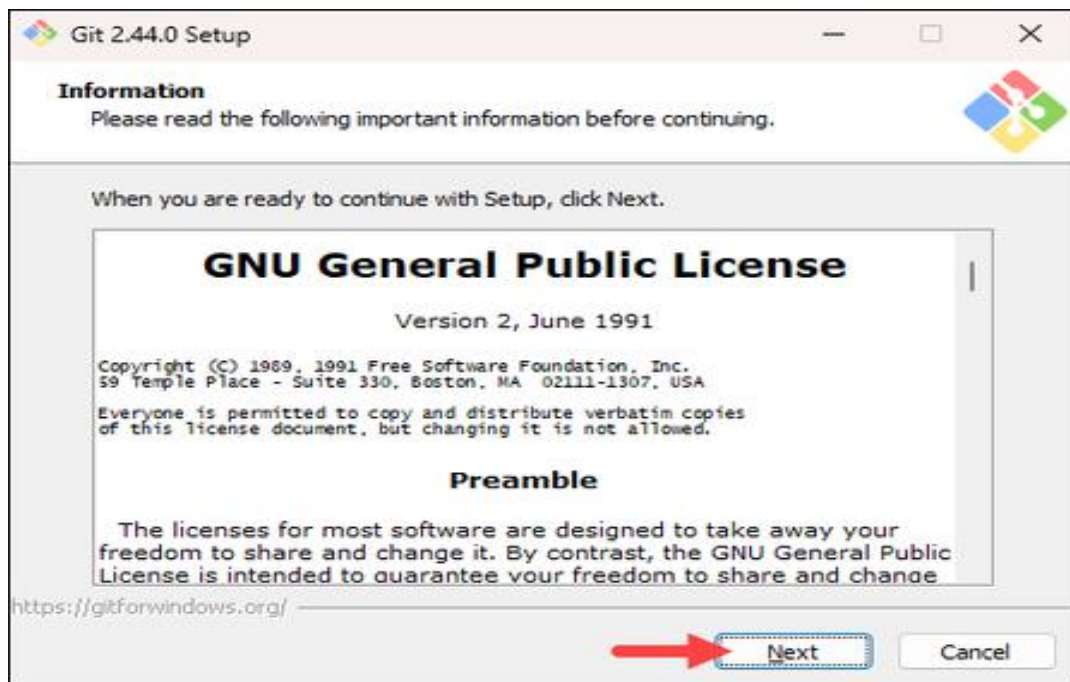
GIT INSTALLATION

- Navigate to the official Git downloads page and click the download link for the latest Git version

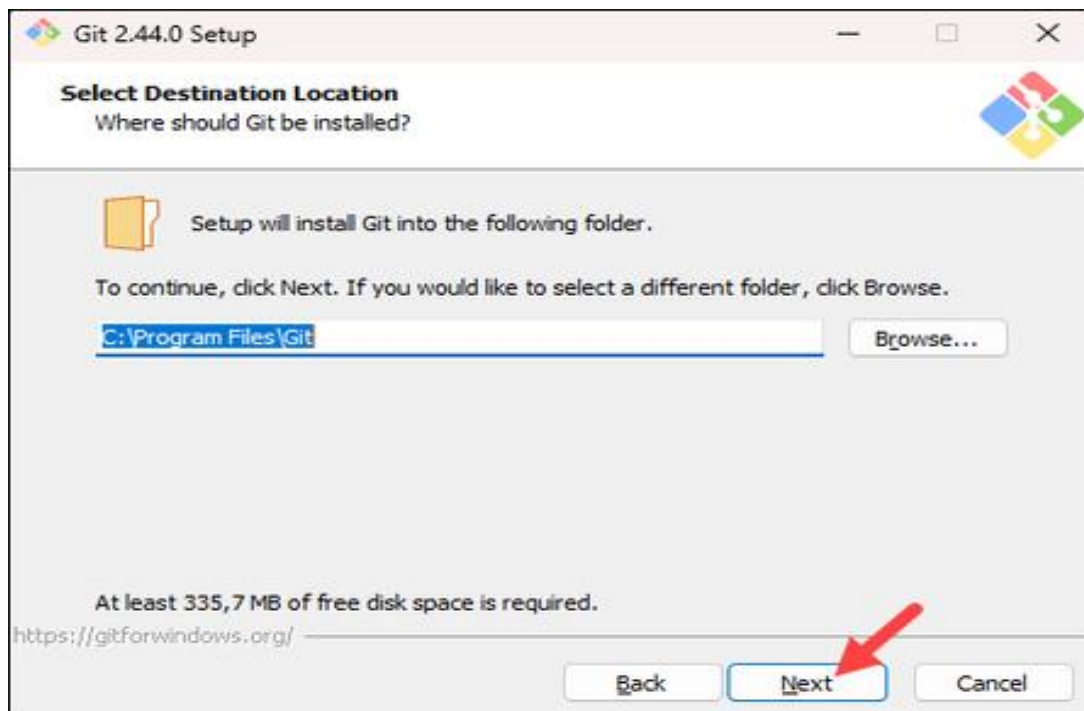
for Windows:



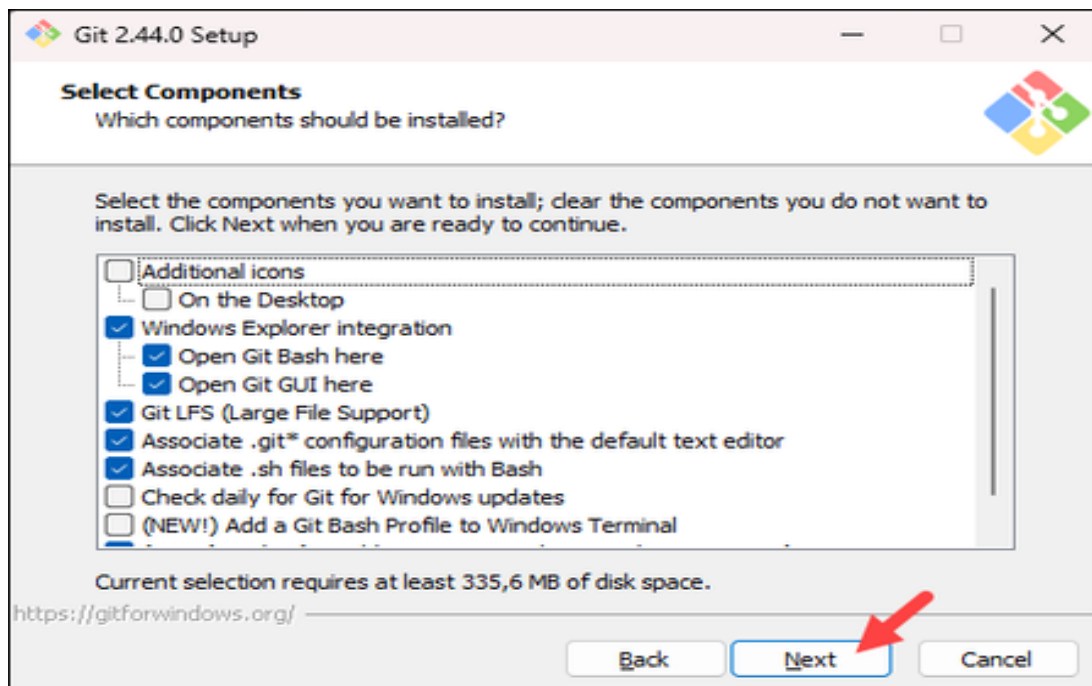
- Double-click the downloaded file to extract and launch the installer
- Review the GNU General Public License, and when you are ready to install, click Next.



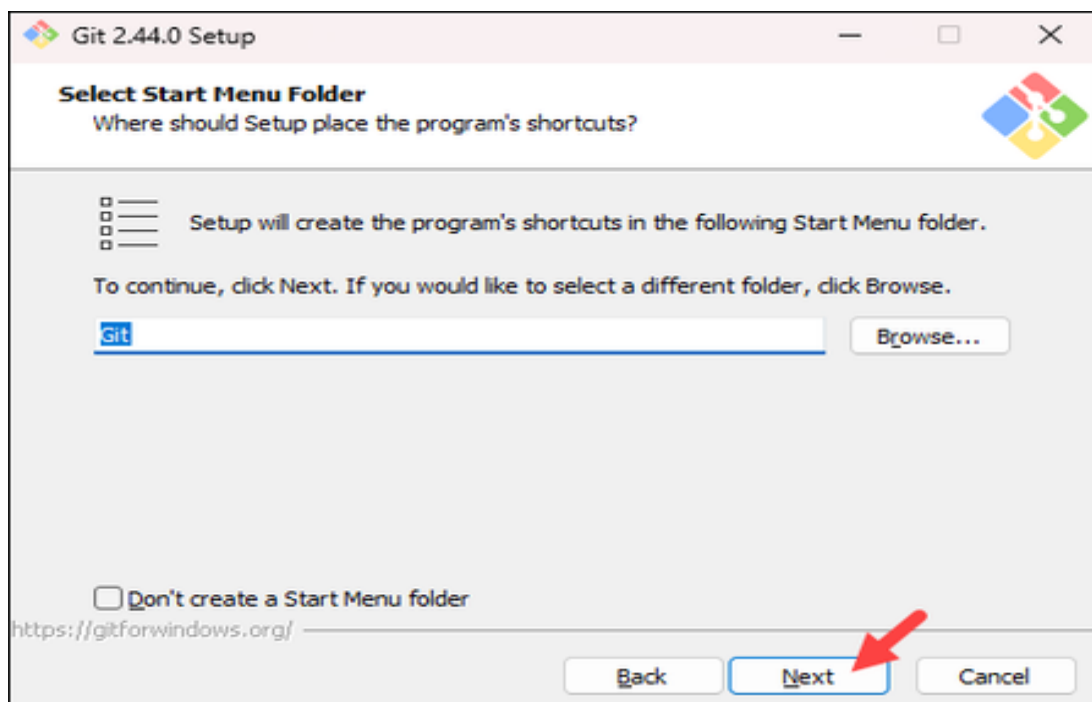
- The installer prompts you for an installation location. Leave the default one unless you want to change it and click Next.



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

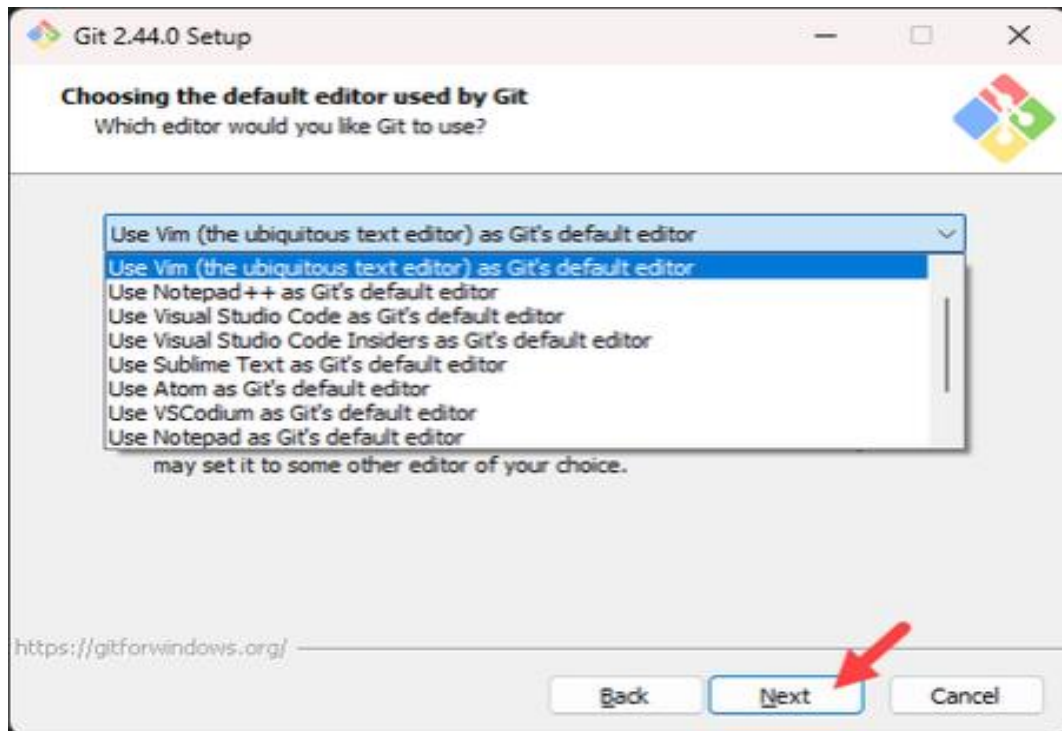


- The installer offers to create a start menu folder. Click Next to accept and proceed to the next step.

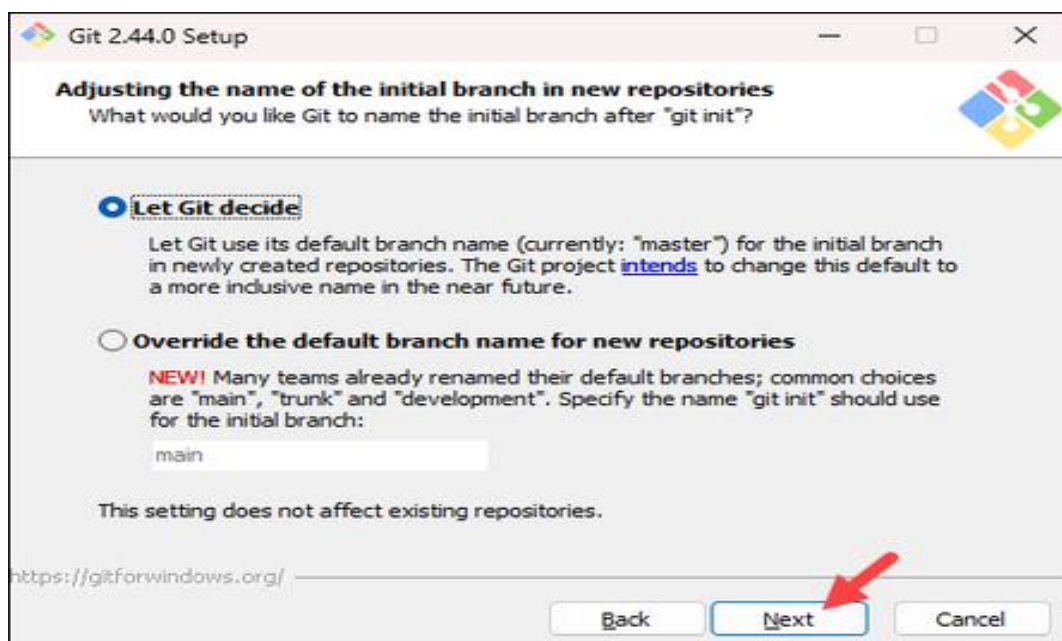


- Select a text editor you want to use with Git. Use the drop-down menu to select Notepad++ (or whichever text editor you prefer) and click Next.

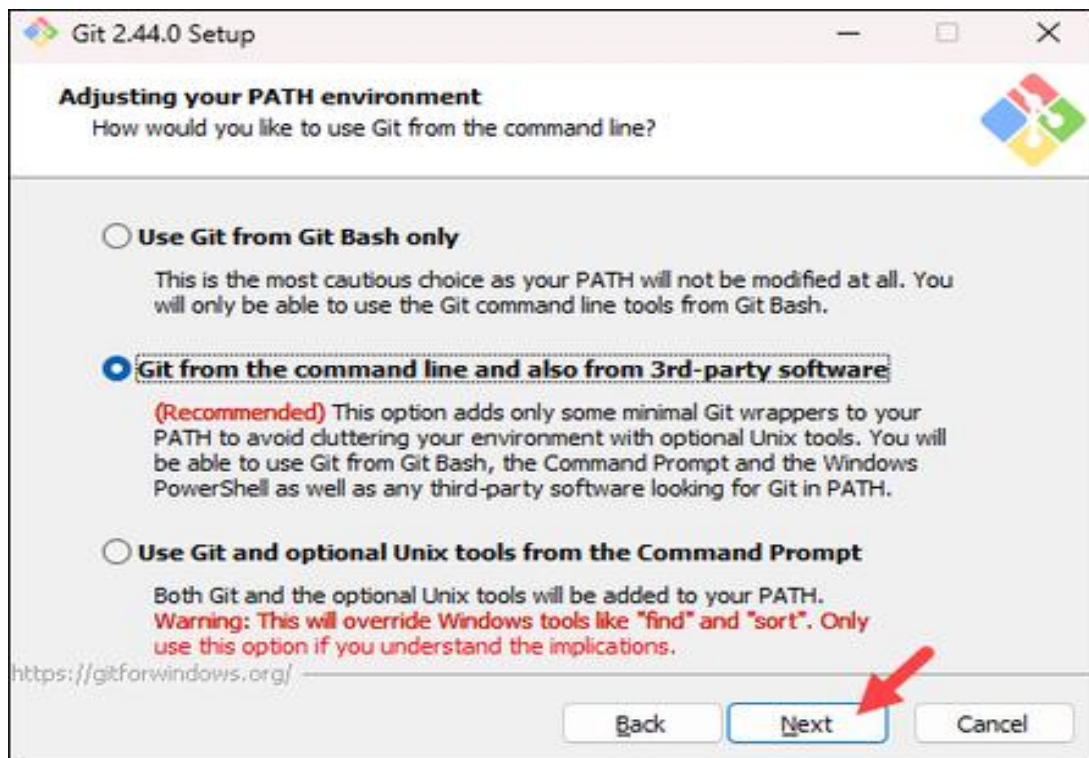
- If you prefer to use a CLI text editor in Git Bash, select nano or Vim from the list.



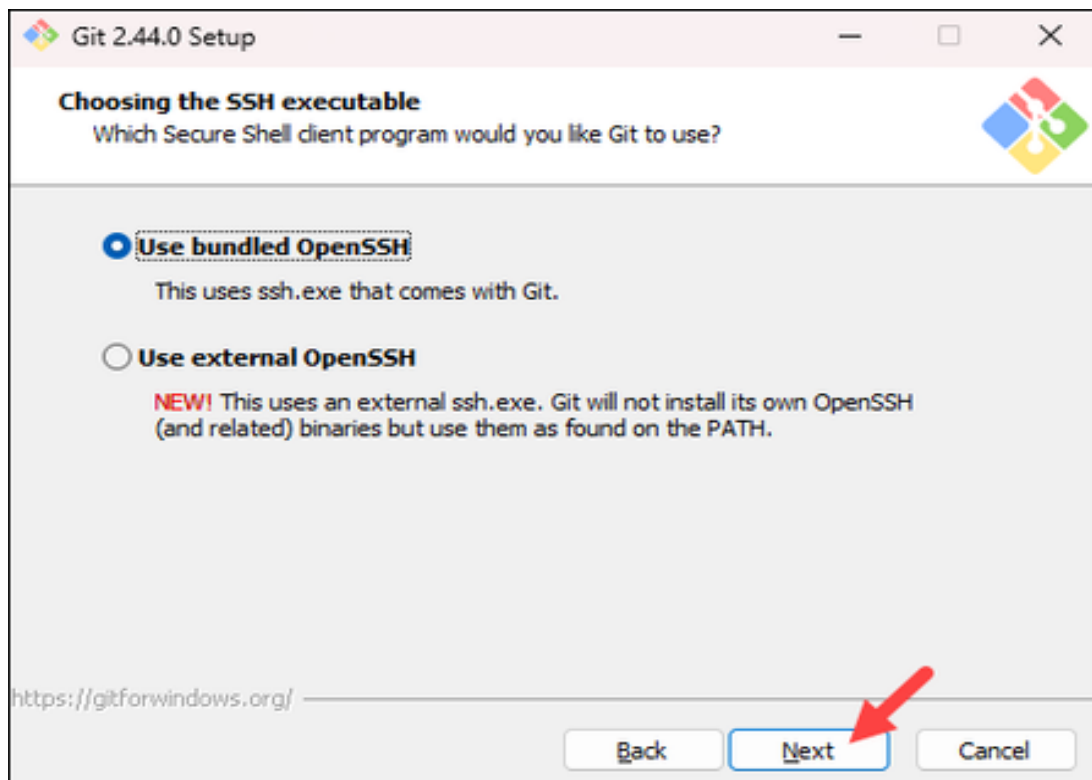
- The next step allows you to choose a different name for your initial branch. The default is master.
- Unless you are working in a team that requires a different name, leave the default option and click Next.



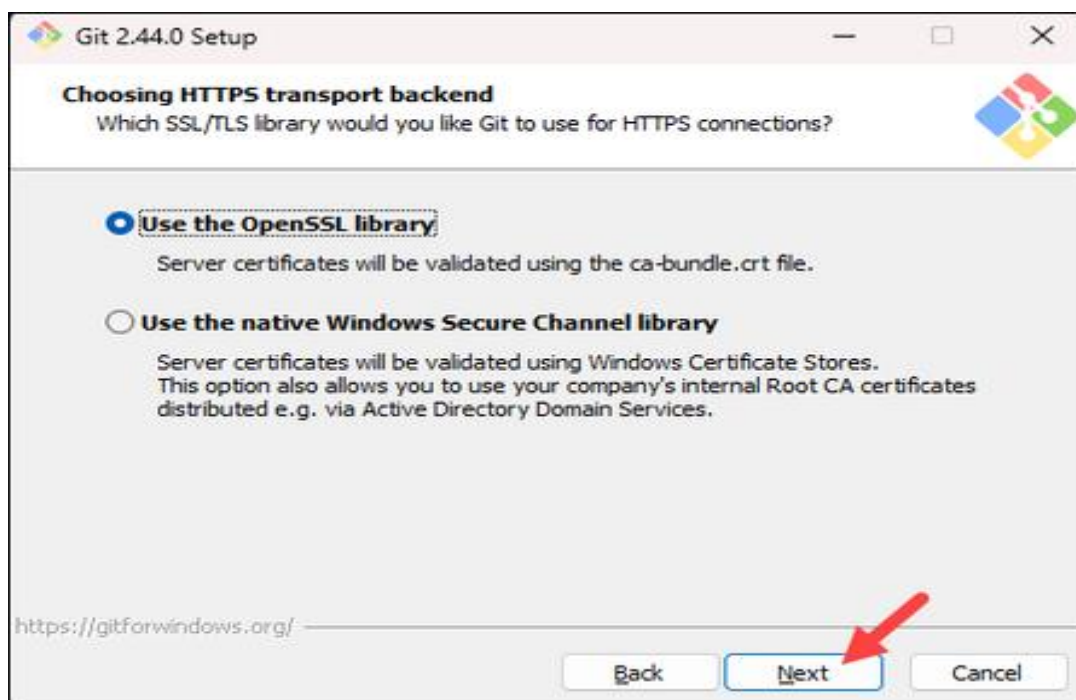
- The next step allows you to change the PATH environment. The PATH is the default set of directories included when you run a command from the command line. Keep the middle selection and click Next.



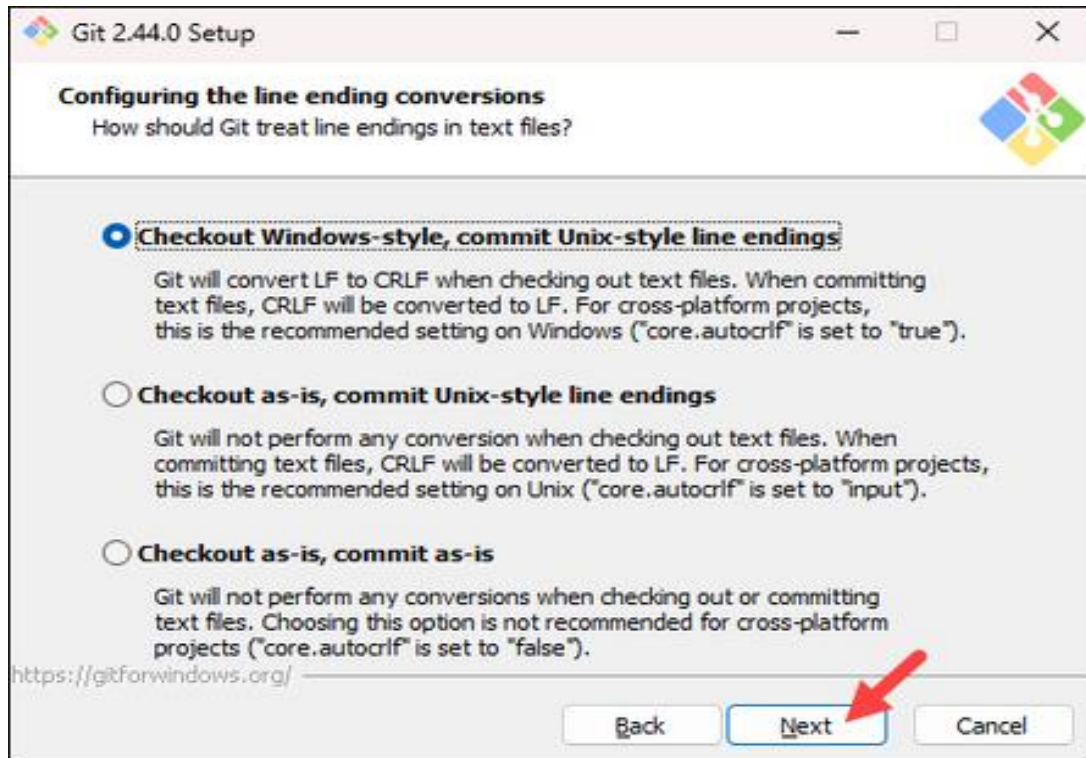
- The installer prompts you to select the SSH client for Git to use. Git already comes with its own SSH client, so if you don't need a specific one, leave the default option and click Next.



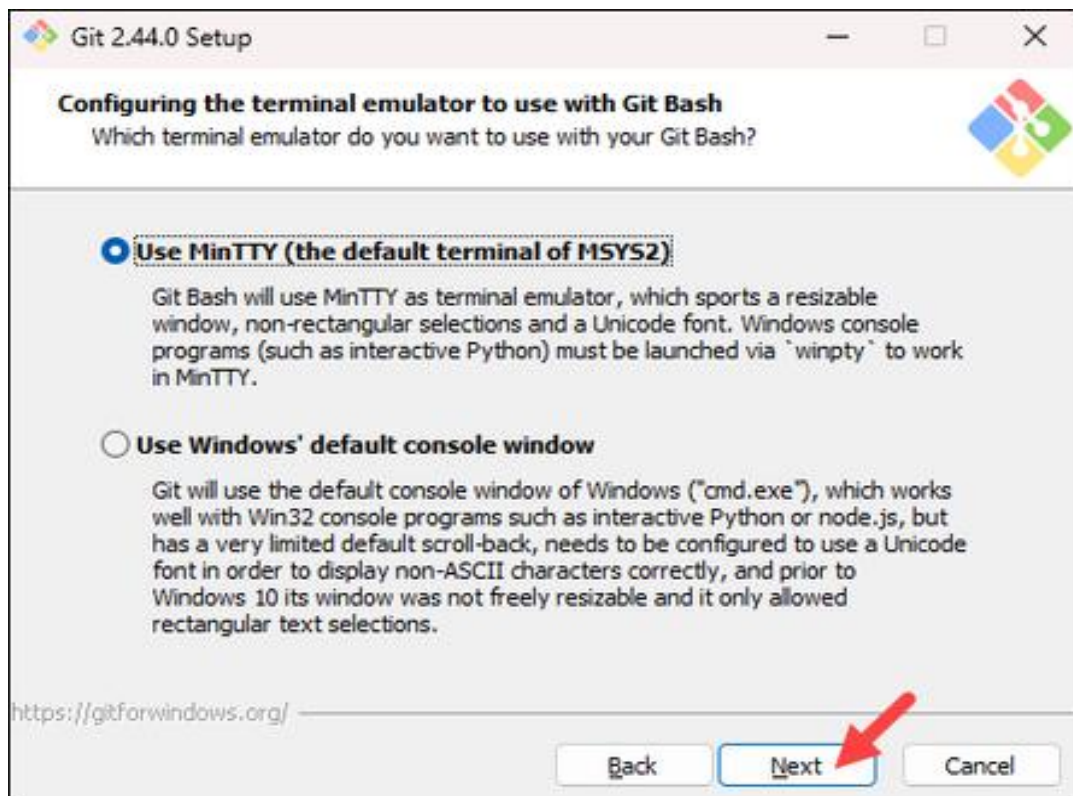
- The next option relates to server certificates. The default option is recommended for most users. If you work in an Active Directory environment; you may need to switch to Windows Store certificates.
- Select your preferred option and click Next.



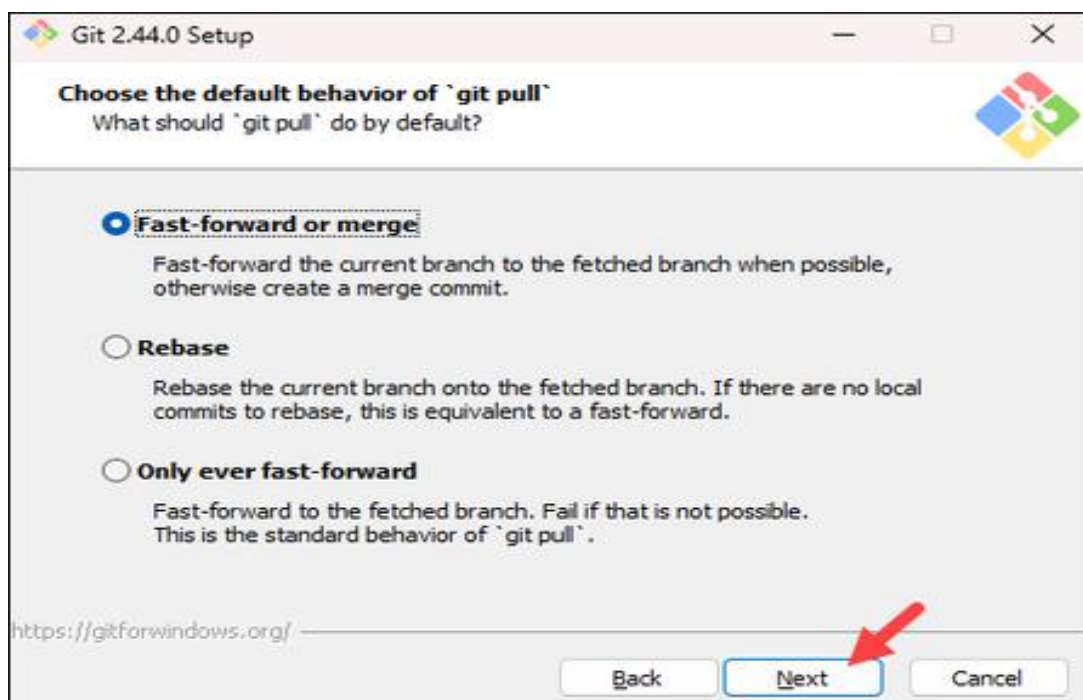
- The following selection configures line-ending conversion, which relates to the way data is formatted. The default selection is recommended for Windows. Click Next to proceed.



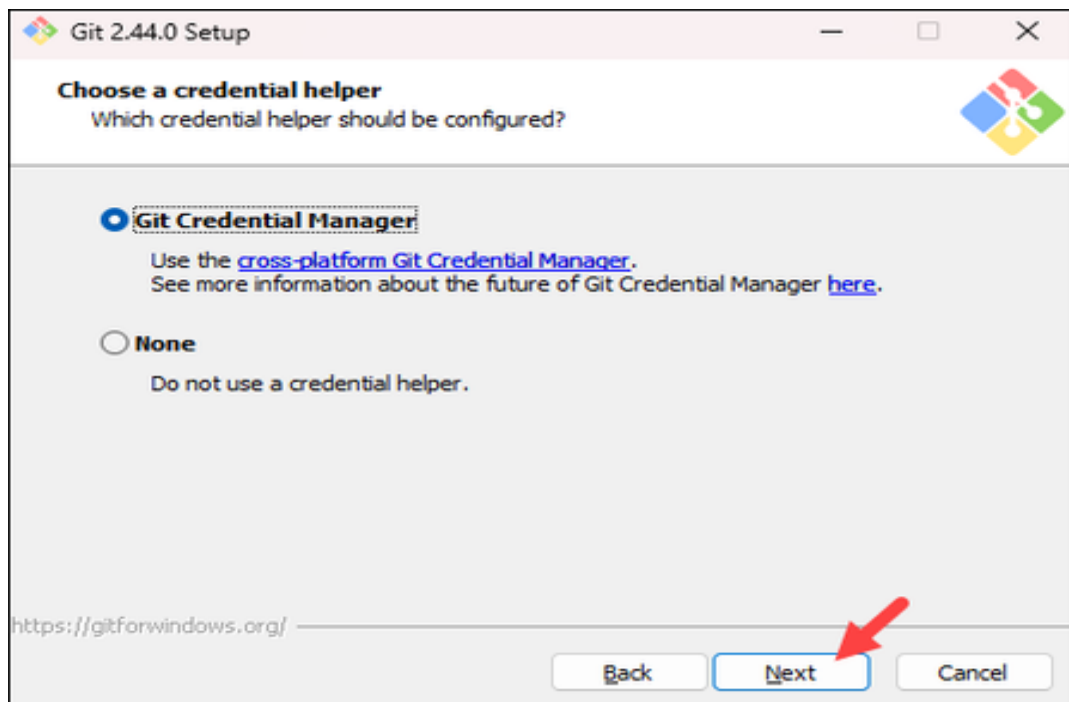
- Choose the terminal emulator you want to use. The default MinTTY is recommended for its features. Click Next to continue.



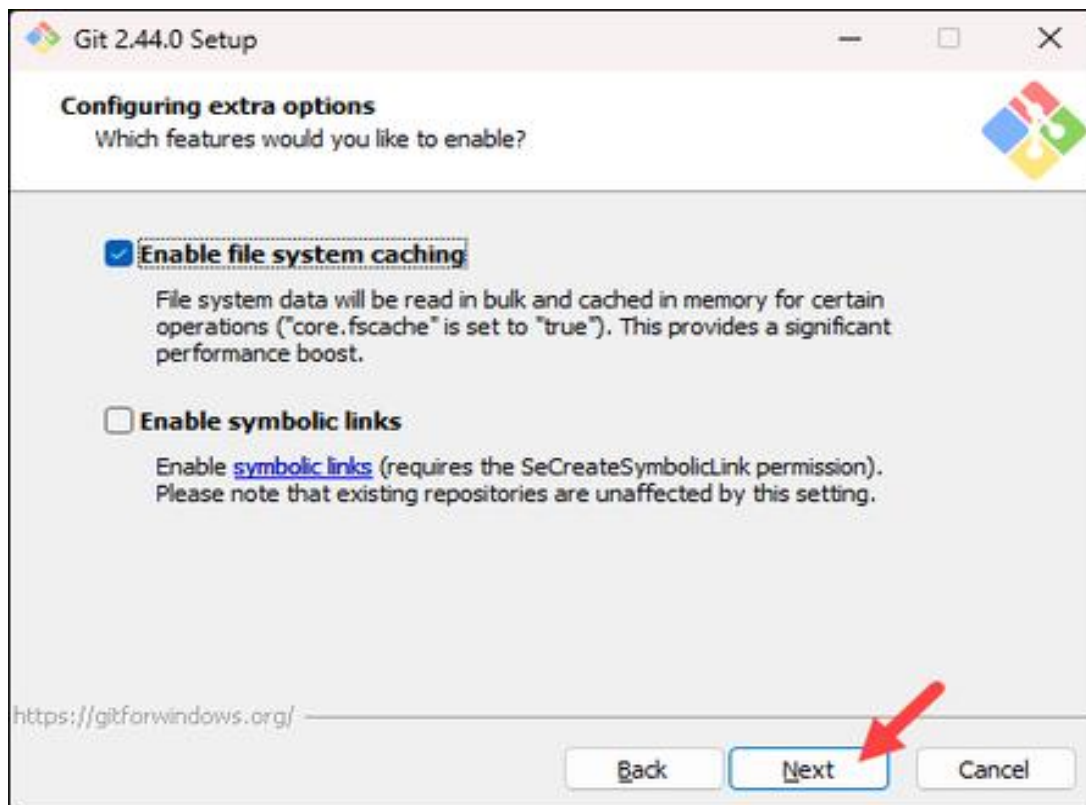
- The next step allows you to choose what the git pull command will do. The default option is recommended unless you specifically need to change its behaviour. Click Next to continue with the installation.



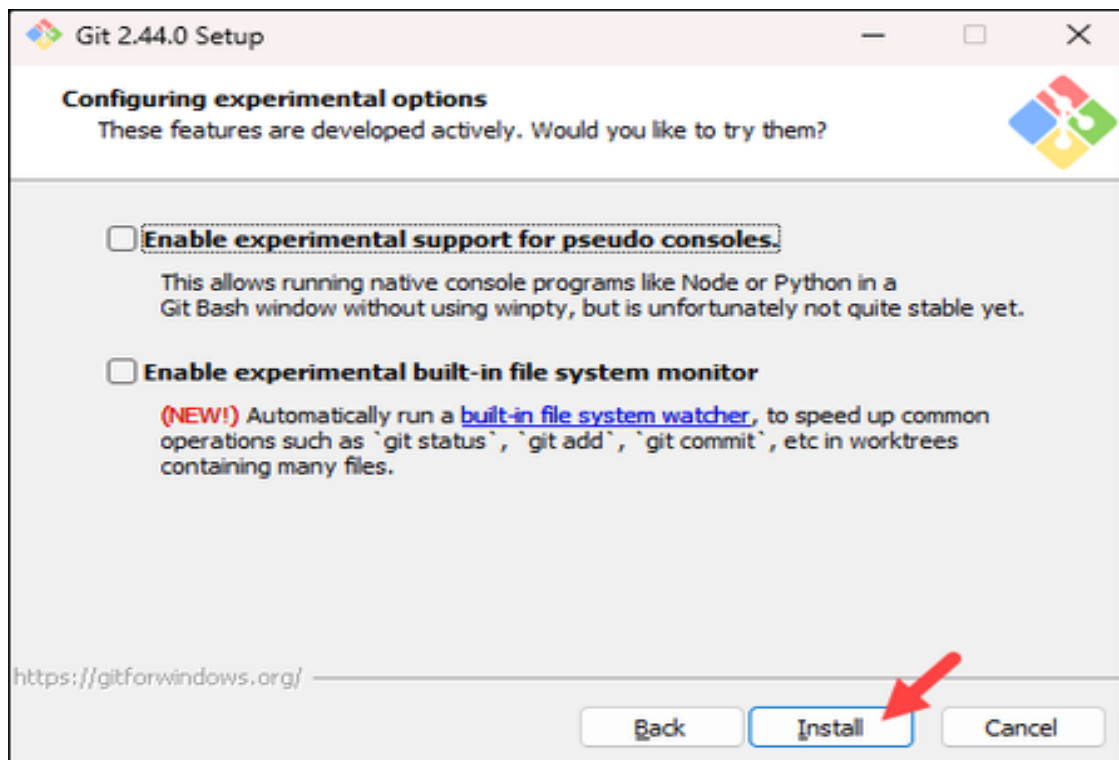
- The next step is to choose which credential helper to use. Git uses credential helpers to fetch or save credentials. The default option is the most stable one. Select your preferred credential manager and click Next.



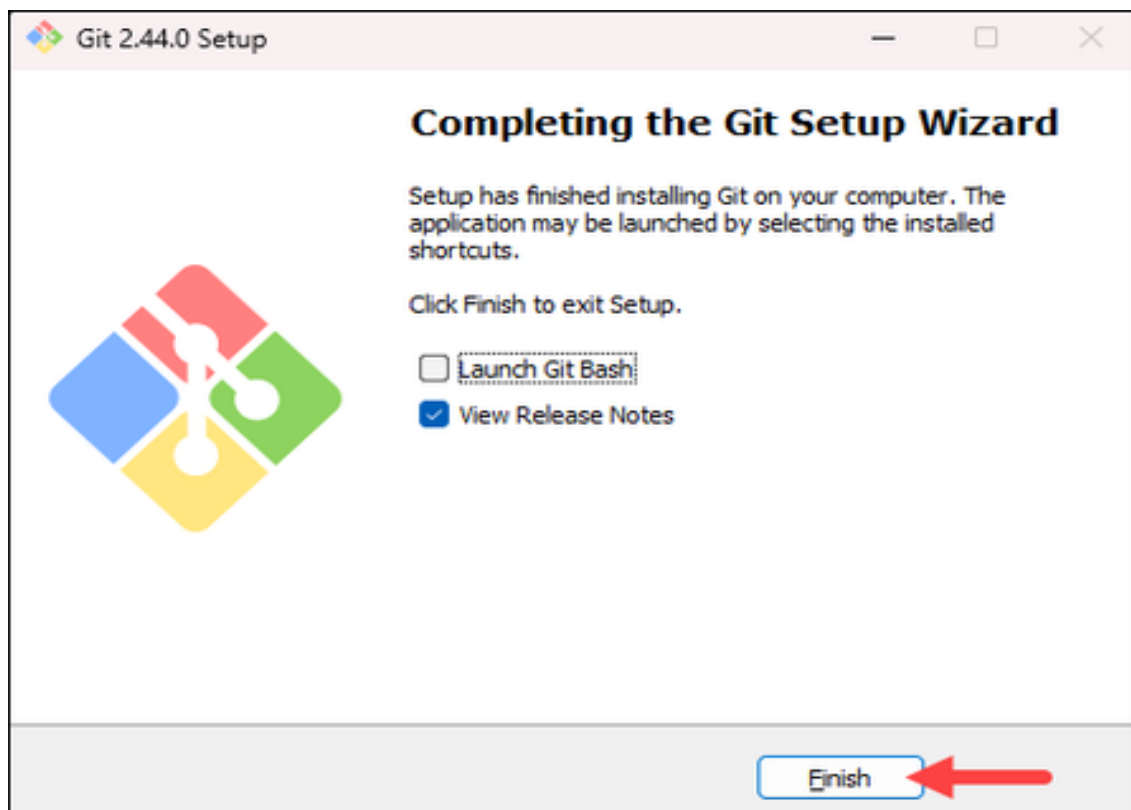
- The next step lets you decide which extra options to enable. If you use symbolic links, which represent shortcuts for the command line, tick the box. Keep file system caching checked and click Next.



- 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.



- 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.



○

COMMANDS

- **pwd**

(Presenting work directory)

It is used to view the current directory working on

```
sripr@LAPTOP-TB8KOFPM MINGW64 ~  
$ pwd  
/c/Users/sripr
```

- **cd <directory name>**

(change directory)

It is used to change the directory

```
sripr@LAPTOP-TB8KOFPM MINGW64 ~  
$ cd Git  
  
sripr@LAPTOP-TB8KOFPM MINGW64 ~/Git  
$
```

- **cd ..**

It is used to change the directory to the parent directory

```
sripr@LAPTOP-TB8KOFPM MINGW64 ~/Git/SCM  
$ cd ..  
  
sripr@LAPTOP-TB8KOFPM MINGW64 ~/Git  
$ |
```

- **ls**

It is used to list all the folders

```
sripr@LAPTOP-TB8KOFPM MINGW64 ~  
$ ls  
AppData/      Downloads/    'My Documents'@  
'Application Data'@  Favorites/    NTUSER.DAT  
Contacts/     Git/          NTUSER.DAT{a15fe73d-00a7-11f0-9671-a4dd710a5f61}.TM.b1f  
Cookies@      Links/        NTUSER.DAT{a15fe73d-00a7-11f0-9671-a4dd710a5f61}.TM.Container00000000000000000001.regtrans-ms  
Desktop/     'Local Settings'@ NTUSER.DAT{a15fe73d-00a7-11f0-9671-a4dd710a5f61}.TM.Container00000000000000000002.regtrans-ms  
Documents/   Music/        NetHood@  
OneDrive/    SendTo@      ntuser.dat.LOG2  
Pictures/    'Start Menu'@ ntuser.ini  
PrintHood@  Templates@  
Recent@     Videos/  
'Saved Games'/ 'WPS Cloud Files'/  
Searches/   ntuser.dat.LOG1
```


- **vi <file name>**

(vim mode/vi editor)

It is used to create a file and edit it

```
sripr@LAPTOP-TB8KOFPM MINGW64 ~/Git/SCM/lab
$ vi fun.txt
```

Click the key “I” to insert or edit your file.

[illegible]

Then press esc key to return from insert mode

```
~  
~  
~  
~  
~  
fun.txt[+] [unix] (23:16 05/06/2025)
```

Enter :wq (write and quit) to return

```
~  
~  
~  
~  
~  
fun.txt[+] [unix] (23:16 05/06/2025)  
:wq
```

- **cat <file name>**

It is used to view the file contents.

```
sripr@LAPTOP-TB8KOFPM MINGW64 ~/Git/SCM/lab  
$ cat fun.txt  
hello
```

- **ls -l**

It is used to list all files

```
sripr@LAPTOP-TB8KOFPM MINGW64 ~/Git/SCM/lab  
$ ls -l  
total 1  
-rw-r--r-- 1 sripr 197609 7 Jun  5 23:19 fun.txt
```

- **ls -ah**

It is used to list all the files including the hidden files

```
sripr@LAPTOP-TB8KOFPM MINGW64 ~/Git/SCM/lab  
$ ls -ah  
./ ../ fun.txt
```

- **history**

It is used to view history of commands

```
sripr@LAPTOP-TB8KOFPM MINGW64 ~/Git/SCM/lab
$ history
1  CLEAR
2  cd
3  pwd
4  cd D:
5  mkdir Git
6  cd Git
7  mkdir SCM
8  cd SCM
9  git config --global user.name "boingin"
10 git config --global user.email "abhipsha.jena@s.amity.edu"
11 git init CodeSmiths
12 ls
13 cd CodeSmiths
14 git branch branch1
15 git branch branch2
```

- **mkdir <folder name>**

It is used to create folder

```
sripr@LAPTOP-TB8KOFPM MINGW64 ~/Git/SCM
$ mkdir lab
```

```
sripr@LAPTOP-TB8KOFPM MINGW64 ~/Git/SCM
$ cd lab

sripr@LAPTOP-TB8KOFPM MINGW64 ~/Git/SCM/lab
$
```

:ls nameoffolder

confirmation to check if the folder is created

:cd folder/

It is used to change the present directory as the folder also after this create one new file by using vi (then insert i)

- **git init**

It is used to initialize the repository
(working directory to empty repository)

```
sripr@LAPTOP-TB8KOFPM MINGW64 ~/Git/SCM/lab
$ git init
Initialized empty Git repository in C:/Users/sripr/Git/SCM/lab/.git/
sripr@LAPTOP-TB8KOFPM MINGW64 ~/Git/SCM/lab (master)
$
```

- **git status**

It is used to display the current status of the working directory.

```
sripr@LAPTOP-TB8KOFPM MINGW64 ~/Git/SCM/lab (master)
$ git status
On branch master

No commits yet

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

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

- **git add <file name>**

It is used to add the file

After this edit the file (using vi, insert i)

```
sripr@LAPTOP-TB8KOFPM MINGW64 ~/Git/SCM/lab (master)
$ git add fun.txt
warning: in the working copy of 'fun.txt', LF will be replaced by CRLF the next time Git touches it
```

- **git add .**

It is used to add all the files in the folder

```
sripr@LAPTOP-TB8KOFPM MINGW64 ~/Git/SCM/lab (master)
$ git add .
```

- **git commit -m "<message>"**

It is used to commit a change

```
sripr@LAPTOP-TB8KOFPM MINGW64 ~/Git/SCM/lab (master)
$ git commit -m "Added a text"
[master (root-commit) 2f7f8b2] Added a text
1 file changed, 1 insertion(+)
create mode 100644 fun.txt
```

- **git config**

It is used to configure username and email

git config --global user.mail "mail"

git config --global user.name "username"

```
git config --global user.name "boingin"
git config --global user.email "abhipsha.jena@s.amity.edu"
```

- **git log**

It is used to view status/entries of commits

```
sripr@LAPTOP-TB8KOFPM MINGW64 ~/Git/SCM/lab (master)
$ git log
commit 2f7f8b23e6951dacf55b4c9d9e71c370f9904832 (HEAD -> master)
Author: boingin <abhipsha.jena@s.amity.edu>
Date: Thu Jun 5 23:32:07 2025 +0530

    Added a text
```

- **git diff**

It shows the changes made in the working directory since the last commit

```
sripr@LAPTOP-TB8KOFPM MINGW64 ~/Git/SCM/lab (master)
$ git diff dd23bde 2f7f8b2
diff --git a/fun.txt b/fun.txt
index d275f2d..f2aa86d 100644
--- a/fun.txt
+++ b/fun.txt
@@ -1,2 +1 @@
-hello
-Added another line
+hello
```

- **git branch<branch name>**

It is used to create a new branch

```
sripr@LAPTOP-TB8KOFPM MINGW64 ~/Git/SCM/lab (master)
$ git branch b1
```

- **git branch**

It lists all the branches in the repository

```
sripr@LAPTOP-TB8KOFPM MINGW64 ~/Git/SCM/lab (master)
$ git branch
b1
* master
```

- **git log --oneline**

It views a concise commit history

```
sripr@LAPTOP-TB8KOFPM MINGW64 ~/Git/SCM/lab (master)
$ git log --oneline
dd23bde (HEAD -> master) Added a line
2f7f8b2 Added a text
```

- **git checkout <branch name>**

It switches to the given branch

```
sripr@LAPTOP-TB8KOFPM MINGW64 ~/Git/SCM/lab (master)
$ git checkout b1
Switched to branch 'b1'

sripr@LAPTOP-TB8KOFPM MINGW64 ~/Git/SCM/lab (b1)
$
```

- **git merge <branch name>**

It is used to combine changes from one branch to your current branch

```
sripr@LAPTOP-TB8KOFPM MINGW64 ~/Git/SCM/lab (master)
$ git merge b1
Already up to date.
```

- **git branch -M <branch name>**

It is used to rename your current branch to a new name

```
sripr@LAPTOP-TB8KOFPM MINGW64 ~/Git/SCM/lab (master)
$ git branch -M main

sripr@LAPTOP-TB8KOFPM MINGW64 ~/Git/SCM/lab (main)
```

- **git remote add origin <url>**

It is used to connect your local repository to a remote Git repository and gives it the name origin

```
sripr@LAPTOP-TB8KOFPM MINGW64 ~/Git/SCM/lab (master)
$ git remote add origin https://github.com/boingin/lab
```

- **git push -u origin <branch name>**

It is used to upload your local commits into the connected remote repository

```
sripr@LAPTOP-TB8KOFPM MINGW64 ~/Git/SCM/lab (main)
$ git push -u origin main
Enumerating objects: 6, done.
Counting objects: 100% (6/6), done.
Delta compression using up to 8 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (6/6), 454 bytes | 454.00 KiB/s, done.
Total 6 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/boingin/lab
 * [new branch]      main -> main
branch 'main' set up to track 'origin/main'.
```

- **git clone <url>**

It is used to download an existing Git repository from a remote server to your local machine

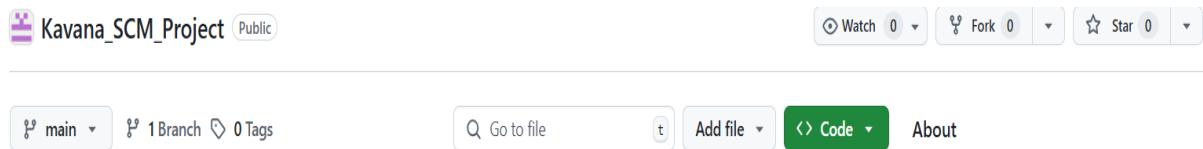
```
sripr@LAPTOP-TB8KOFPM MINGW64 ~/Git/SCM
$ git clone https://github.com/boingin/Kavana_SCM_Project.git
Cloning into 'Kavana_SCM_Project'...
remote: Enumerating objects: 43, done.
remote: Counting objects: 100% (43/43), done.
remote: Compressing objects: 100% (23/23), done.
remote: Total 43 (delta 20), reused 39 (delta 19), pack-reused 0 (from 0)
Receiving objects: 100% (43/43), 895.22 KiB | 1.50 MiB/s, done.
Resolving deltas: 100% (20/20), done.
```

- **git pull**

It is used to download changes from the remote repository and integrate them into your current local branch

HOW TO FORK A REPOSITORY

- Open the repository you want to fork and click on the fork option on the top right corner



- You will come to the page which says to create a new fork
- You can select the owner from the list and type on the desired name you want the repository to be forked on
- By unticking the “Copy the main branch only” might help you to fork all the branches from the person’s repository which might help you in the contributing better to the projects as you will have access to all their previous codes

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 *


 boingin / Kavana_SCM_Project

✔ Kavana_SCM_Project is available.

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

Description (optional)

☒ Copy the `main` branch only
Contribute back to L15-16-WS/Kavana_SCM_Project by adding your own branch. [Learn more.](#)

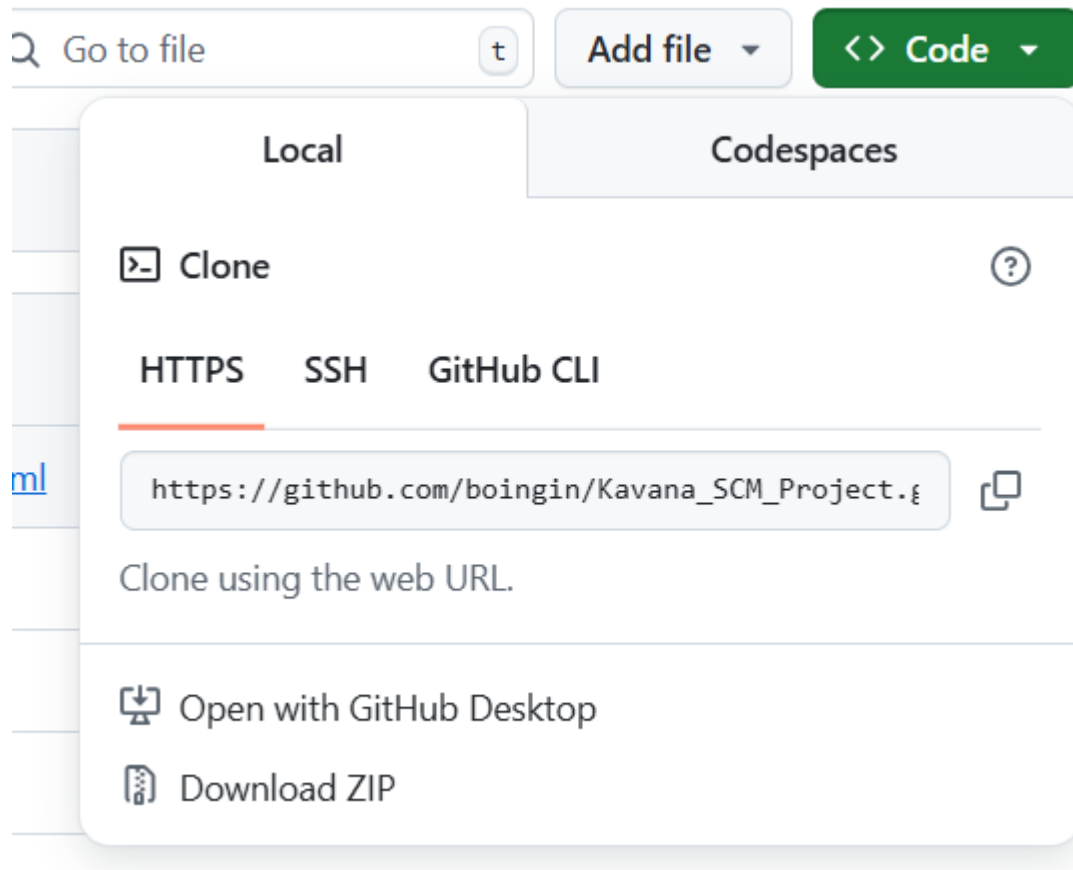
 You are creating a fork in your personal account.

Create fork

- After forking the repository will appear in your profile



- Now you can make contribution to their project by cloning it.
- To clone the repository, tap on the code option
- With the url provided you can copy it and use the git clone command to clone the forked repository into your system to work on the contribution



- After cloning, open their repository (cd <the file name you saved their repository as>)
- Create a separate branch and then make your contribution