Experiment -1.1

Student Name: Yash Dwivedi	**UID:** 22BDO10019
Branch: AIT-CSE(DevOps)	**Section/Group**: 22BCD-1/A
Semester: 4th	**Date of Performance:** 17/01/2024
Subject Name: Git and Hub	**Subject Code:** 22CSH-293

1. **Aim/Overview of the practical:** Install Git and creating a repository.

2. **Software Used:** Git Bash, GitHub.

3. **Steps for experiment/practical:**

   1. Select the operating system on which you want to install Git as shown in image 1.

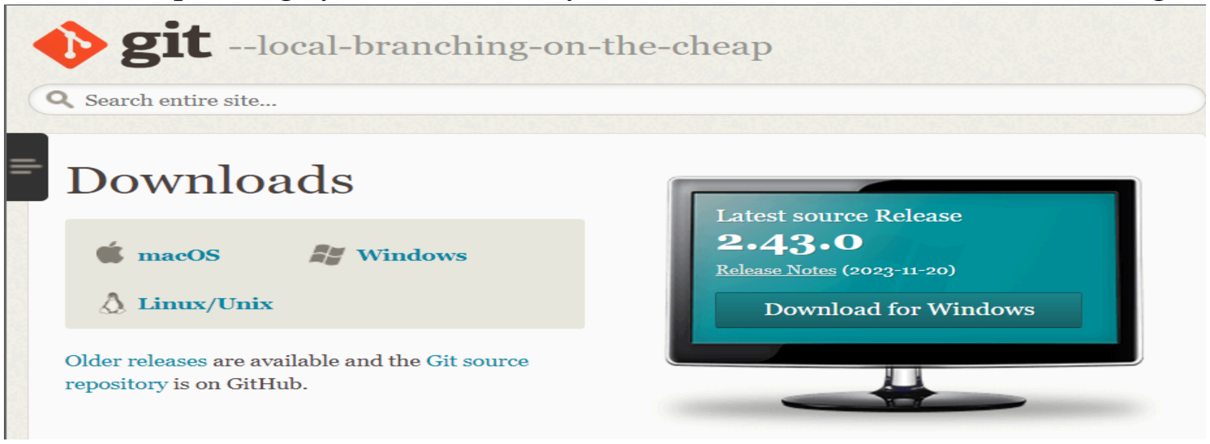
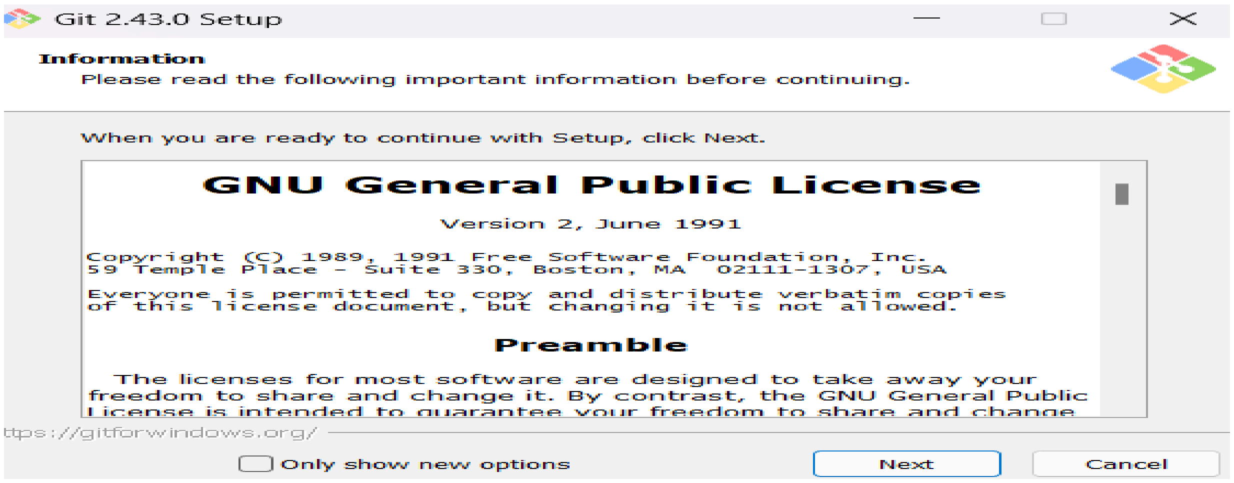

git --local-branching-on-the-cheap
Search entire site...
Downloads
macOS
Windows
Linux/Unix
Older releases are available and the Git source repository is on GitHub.
Latest source Release
2.43.0
Release Notes (2023-11-20)
Download for Windows

Image 1

   2. Select the installer, x32-bit or x64-bit, according to your operating system.

   3. Click on 'next'.

   4. Accept the agreement and click next (in image 2). Select the shown options and click on next (in image 3).


Git 2.43.0 Setup

Information
Please read the following important information before continuing.

When you are ready to continue with Setup, click Next.

GNU General Public License

Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.
59 Temple Place - Suite 330, Boston, MA 02111-1307, USA

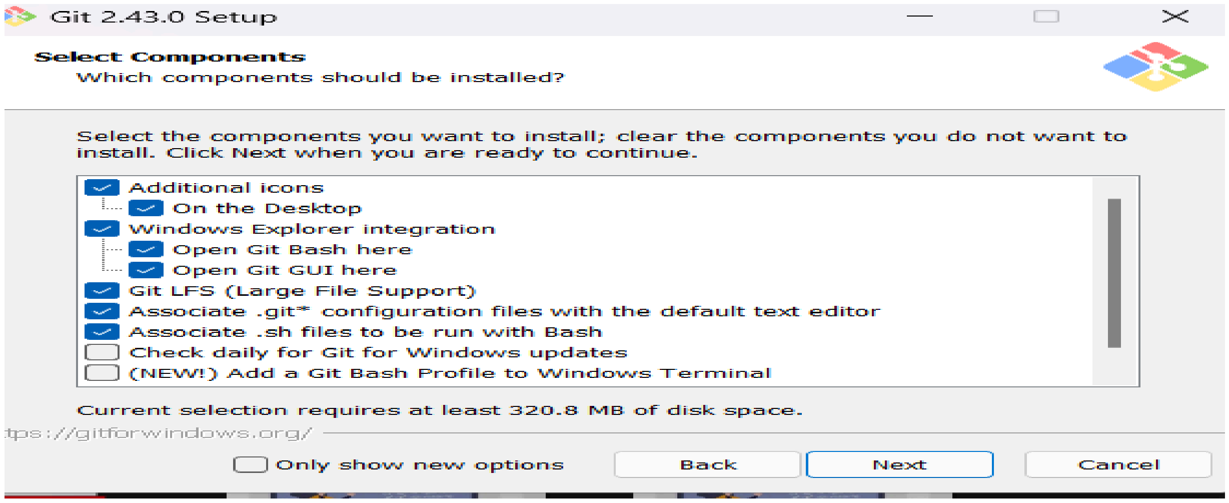
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your
freedom to share and change it. By contrast, the GNU General Public
License is intended to guarantee your freedom to share and change

ttps://gitforwindows.org/

Image 2


Git 2.43.0 Setup

Select Components
Which components should be installed?

Select the components you want to install; clear the components you do not want to
install. Click Next when you are ready to continue.

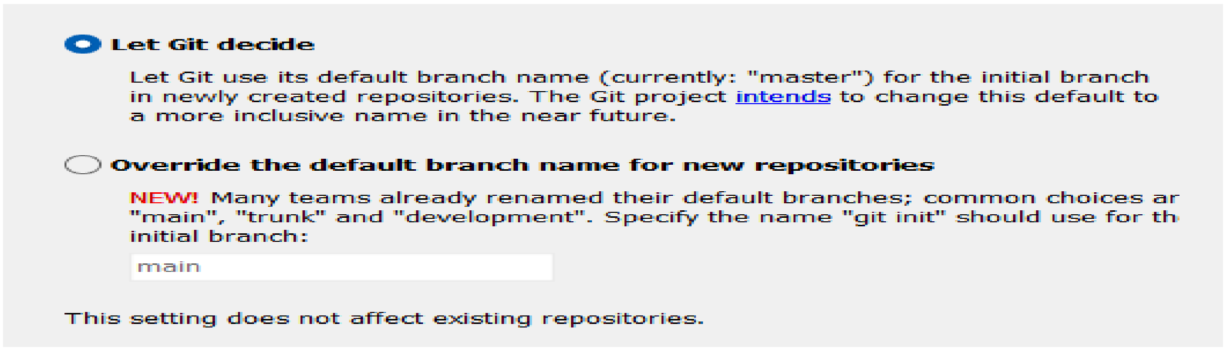
- [x] Additional icons
  - [x] On the Desktop
- [x] Windows Explorer integration
  - [x] Open Git Bash here
  - [x] Open Git GUI here
- [x] Git LFS (Large File Support)
- [x] Associate .git* configuration files with the default text editor
- [x] Associate .sh files to be run with Bash
- [ ] Check daily for Git for Windows updates
- [ ] (NEW!) Add a Git Bash Profile to Windows Terminal

Current selection requires at least 320.8 MB of disk space.

tps://gitforwindows.org/

Image 3

5. Keep on clicking the Next button and select the options as shown in the following images (in images from 4 to 13) .


● **Let Git decide**

Let Git use its default branch name (currently: "master") for the initial branch in newly created repositories. The Git project intends to change this default to a more inclusive name in the near future.

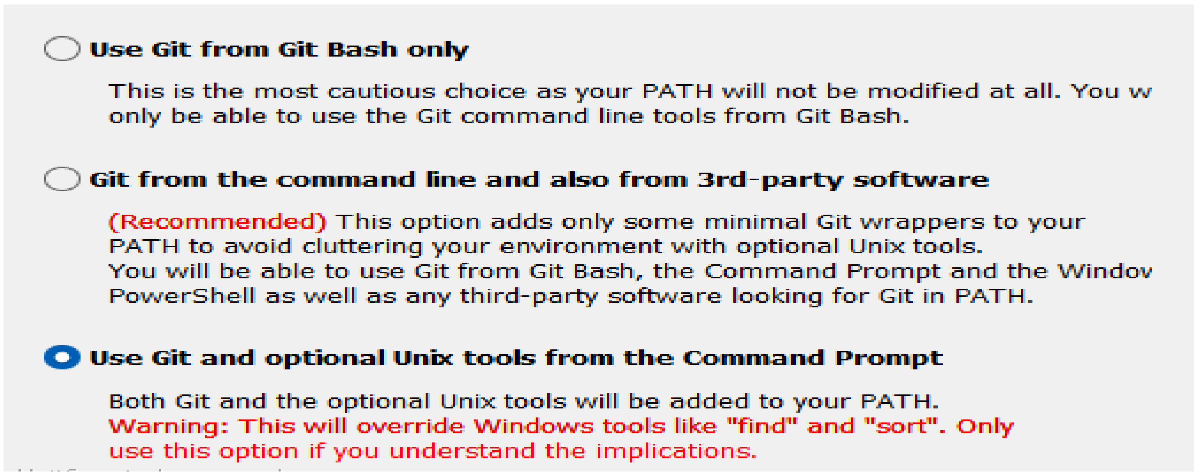
○ **Override the default branch name for new repositories**

NEW! Many teams already renamed their default branches; common choices ar "main", "trunk" and "development". Specify the name "git init" should use for th initial branch:

main

This setting does not affect existing repositories.

Image 4



○ **Use Git from Git Bash only**

This is the most cautious choice as your PATH will not be modified at all. You w
only be able to use the Git command line tools from Git Bash.

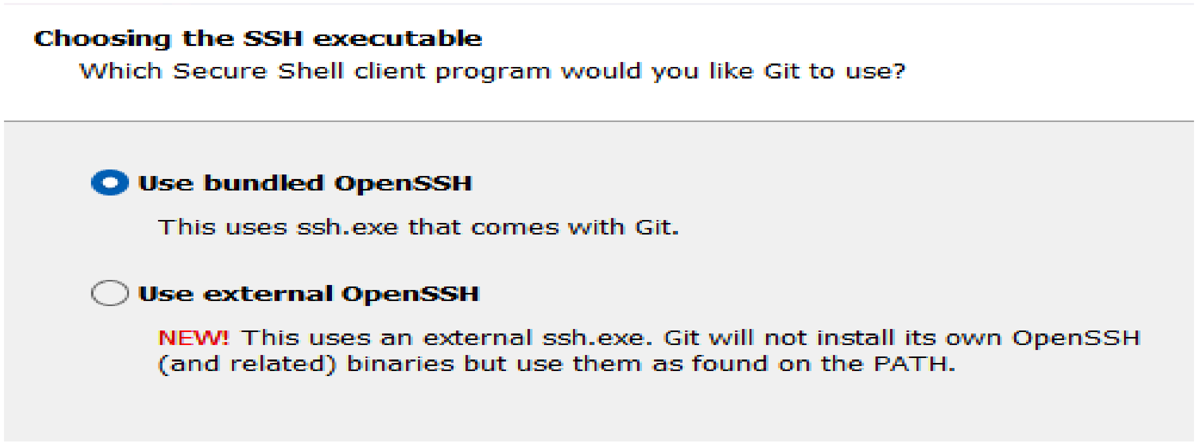
○ **Git from the command line and also from 3rd-party software**

(Recommended) This option adds only some minimal Git wrappers to your
PATH to avoid cluttering your environment with optional Unix tools.
You will be able to use Git from Git Bash, the Command Prompt and the Windov
PowerShell as well as any third-party software looking for Git in PATH.

● **Use Git and optional Unix tools from the Command Prompt**

Both Git and the optional Unix tools will be added to your PATH.
Warning: This will override Windows tools like "find" and "sort". Only
use this option if you understand the implications.

Image 5



Choosing the SSH executable
Which Secure Shell client program would you like Git to use?

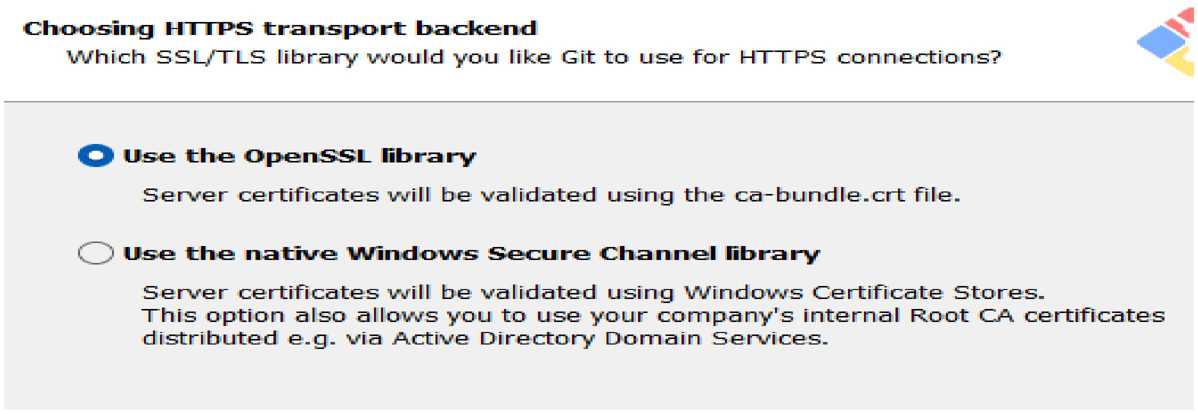
● **Use bundled OpenSSH**

This uses ssh.exe that comes with Git.

○ **Use external OpenSSH**

NEW! This uses an external ssh.exe. Git will not install its own OpenSSH
(and related) binaries but use them as found on the PATH.

Image 6



Choosing HTTPS transport backend
Which SSL/TLS library would you like Git to use for HTTPS connections?

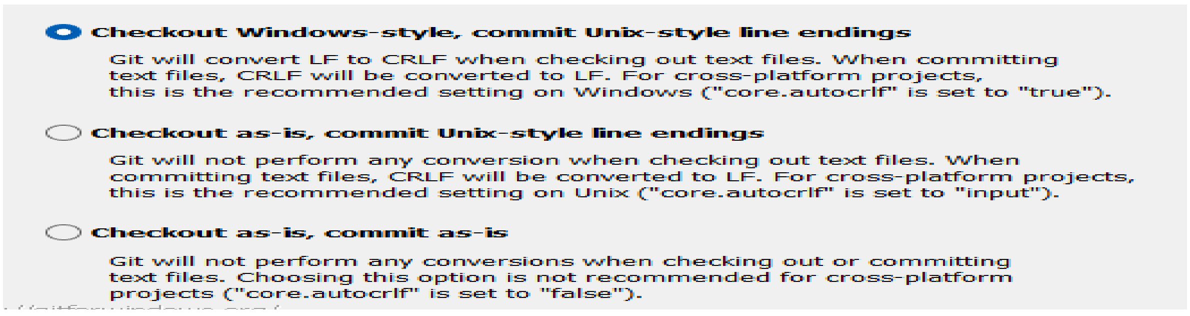
● **Use the OpenSSL library**

Server certificates will be validated using the ca-bundle.crt file.

○ **Use the native Windows Secure Channel library**

Server certificates will be validated using Windows Certificate Stores.
This option also allows you to use your company's internal Root CA certificates
distributed e.g. via Active Directory Domain Services.

Image 7


[✓ **Checkout Windows-style, commit Unix-style line endings**]

Git will convert LF to CRLF when checking out text files. When committing text files, CRLF will be converted to LF. For cross-platform projects, this is the recommended setting on Windows ("core.autocrlf" is set to "true").

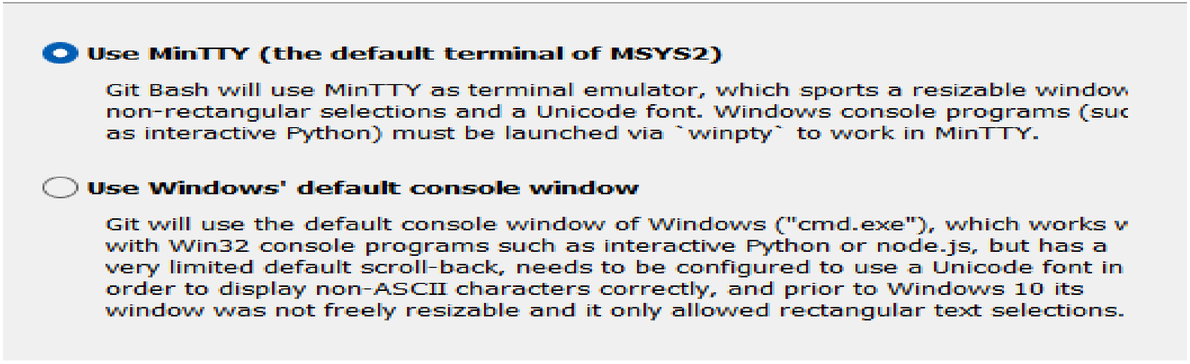
Checkout as-is, commit Unix-style line endings

Git will not perform any conversion when checking out text files. When committing text files, CRLF will be converted to LF. For cross-platform projects, this is the recommended setting on Unix ("core.autocrlf" is set to "input").

Checkout as-is, commit as-is

Git will not perform any conversions when checking out or committing text files. Choosing this option is not recommended for cross-platform projects ("core.autocrlf" is set to "false").

Image 8

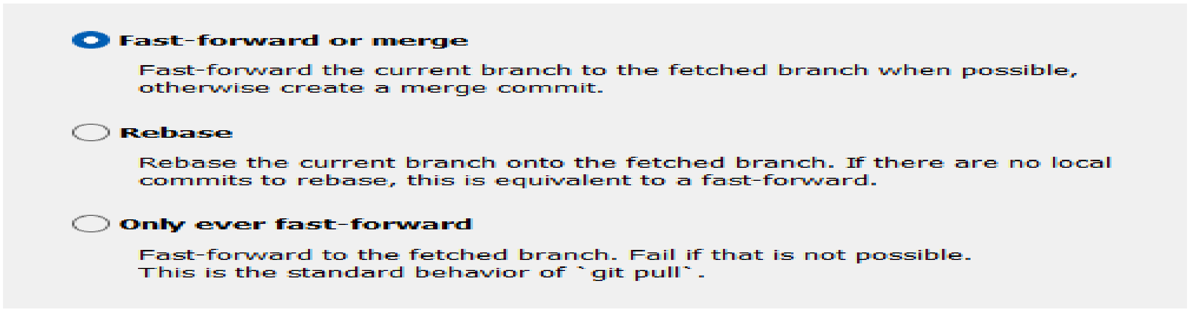

[✓ **Use MinTTY (the default terminal of MSYS2)**]

Git Bash will use MinTTY as terminal emulator, which sports a resizable window non-rectangular selections and a Unicode font. Windows console programs (suc as interactive Python) must be launched via `winpty` to work in MinTTY.

Use Windows' default console window

Git will use the default console window of Windows ("cmd.exe"), which works v with Win32 console programs such as interactive Python or node.js, but has a very limited default scroll-back, needs to be configured to use a Unicode font in order to display non-ASCII characters correctly, and prior to Windows 10 its window was not freely resizable and it only allowed rectangular text selections.

Image 9


[✓ **Fast-forward or merge**]

Fast-forward the current branch to the fetched branch when possible, otherwise create a merge commit.

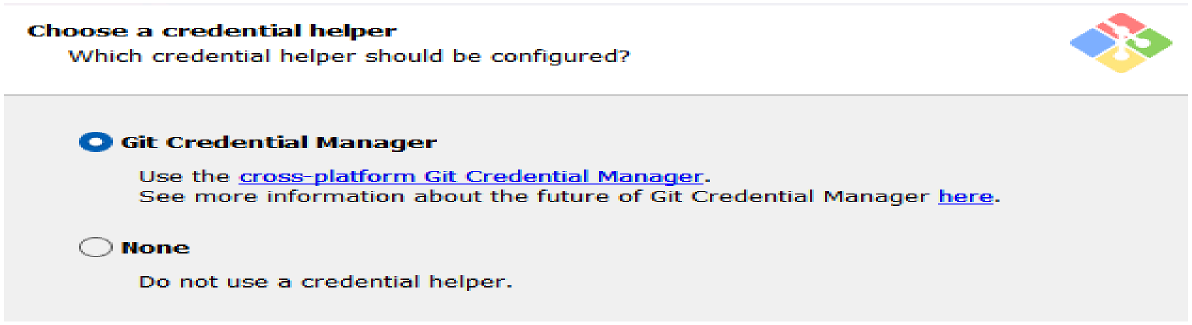
Rebase

Rebase the current branch onto the fetched branch. If there are no local commits to rebase, this is equivalent to a fast-forward.

Only ever fast-forward

Fast-forward to the fetched branch. Fail if that is not possible. This is the standard behavior of `git pull`.

Image 10


Choose a credential helper

Which credential helper should be configured?

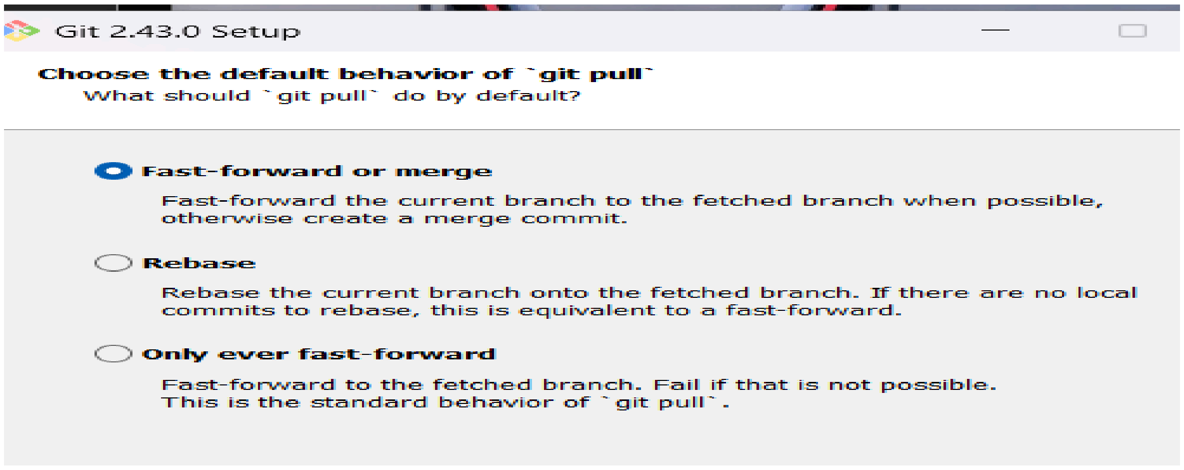
[✓ **Git Credential Manager**]

Use the cross-platform Git Credential Manager.
See more information about the future of Git Credential Manager here.

None

Do not use a credential helper.

Image 11

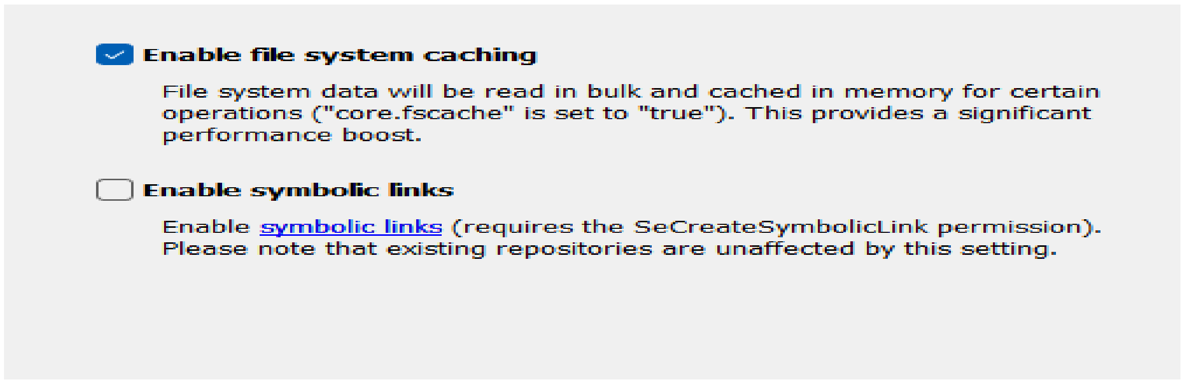


Git 2.43.0 Setup

Choose the default behavior of `git pull`
What should `git pull` do by default?

- [✓] **Fast-forward or merge**
  Fast-forward the current branch to the fetched branch when possible, otherwise create a merge commit.
- [ ] **Rebase**
  Rebase the current branch onto the fetched branch. If there are no local commits to rebase, this is equivalent to a fast-forward.
- [ ] **Only ever fast-forward**
  Fast-forward to the fetched branch. Fail if that is not possible. This is the standard behavior of `git pull`.

Image 12



- [✓] **Enable file system caching**
  File system data will be read in bulk and cached in memory for certain operations ("core.fscache" is set to "true"). This provides a significant performance boost.
- [ ] **Enable symbolic links**
  Enable symbolic links (requires the SeCreateSymbolicLink permission). Please note that existing repositories are unaffected by this setting.

Image 13

6. Click on the Install button and wait for Git to be installed on your system. After the installation, you will be able to see **Git Bash** and **Git GUI** (in image 14).

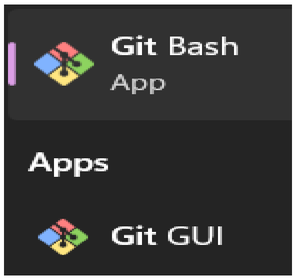
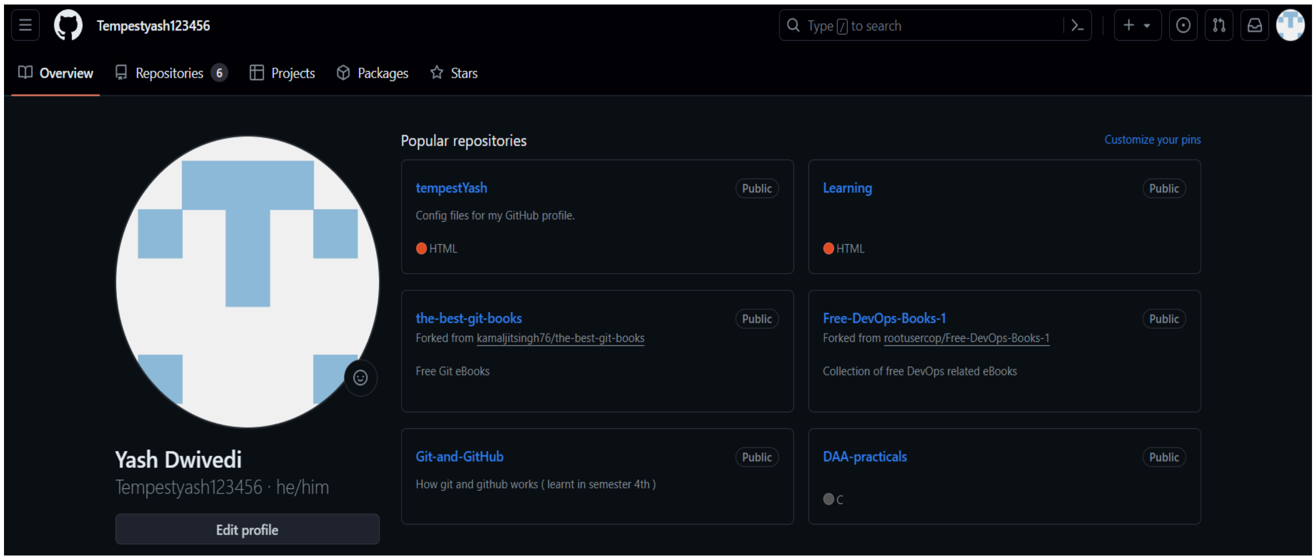


Image 14

7. Now, we will create an account on GitHub.



Popular repositories

tempestYash Public
Config files for my GitHub profile.
HTML

Learning Public
HTML

the-best-git-books Public
Forked from kamaljitsingh76/the-best-git-books
Free Git eBooks

Free-DevOps-Books-1 Public
Forked from rootusercop/Free-DevOps-Books-1
Collection of free DevOps related eBooks

Git-and-GitHub Public
How git and github works (learnt in semester 4th)

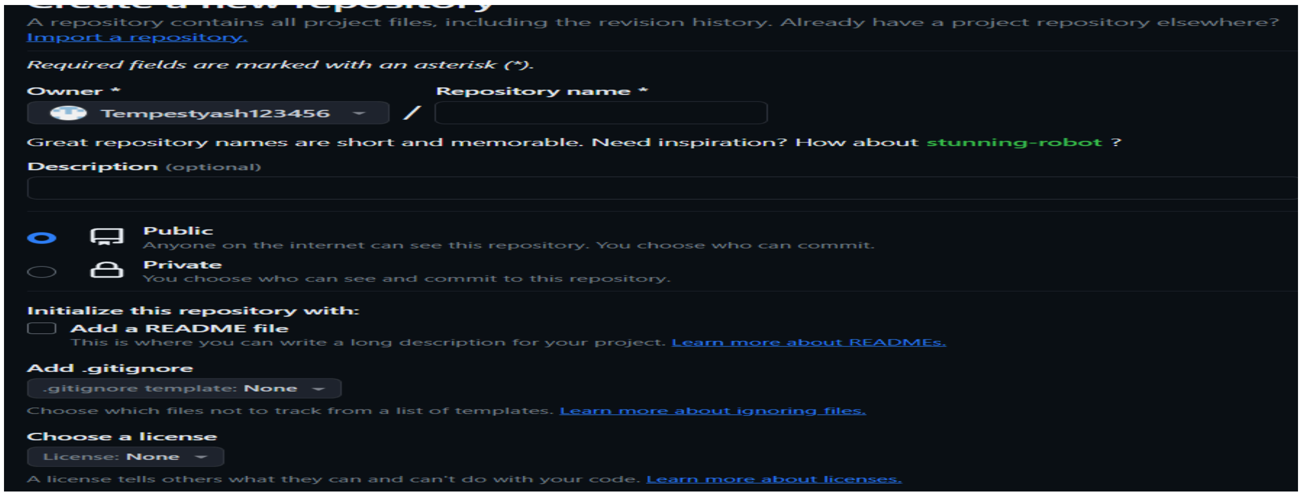
DAA-practicals Public
C

Yash Dwivedi
Tempestyash123456 · he/him

Image 15

8. Now, we will be creating a sample repository on Github (in image 16).

- Click on repositories
- Click on New
- Name the repository
- write the description of the repository(this is optional)
- Make the repository public or private.
- Add a README, .gitignore file and provide the GNU license



A repository contains all project files, including the revision history. Already have a project repository elsewhere? Import a repository.

Required fields are marked with an asterisk (*).

Owner * Tempestyash123456 / Repository name *

Great repository names are short and memorable. Need inspiration? How about stunning-robot ?

Description (optional)

Public
Anyone on the internet can see this repository. You choose who can commit.

Private
You choose who can see and commit to this repository.

Initialize this repository with:

Add a README file
This is where you can write a long description for your project. Learn more about READMEs.

Add .gitignore
.gitignore template: None
Choose which files not to track from a list of templates. Learn more about ignoring files.

Choose a license
License: None
A license tells others what they can and can't do with your code. Learn more about licenses.

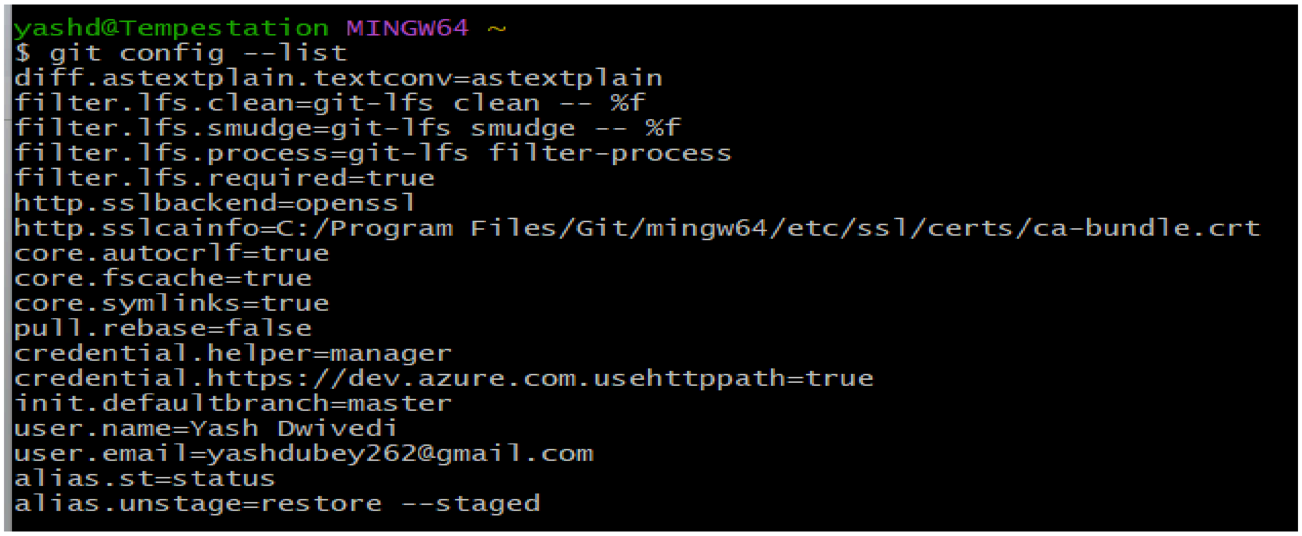
Image 16

9. Now, we will be configuring Git with our Github account. Open Git Bash and run the following git commands to configure your name and email respectively.

- **git config --global user.name "Yash Dwivedi"**
- **git config --global user.email "yashdubey262@gmail.com"**

Use the following git command to check whether the user has configured or not.

- **git config --list** (in image 17)



```
yashd@Tempestation MINGW64 ~
$ git config --list
diff.astextplain.textconv=astextplain
filter.lfs.clean=git-lfs clean -- %f
filter.lfs.smudge=git-lfs smudge -- %f
filter.lfs.process=git-lfs filter-process
filter.lfs.required=true
http.sslbackend=openssl
http.sslcainfo=C:/Program Files/Git/mingw64/etc/ssl/certs/ca-bundle.crt
core.autocrlf=true
core.fscache=true
core.symlinks=true
pull.rebase=false
credential.helper=manager
credential.https://dev.azure.com.usehttppath=true
init.defaultbranch=master
user.name=Yash Dwivedi
user.email=yashdubey262@gmail.com
alias.st=status
alias.unstage=restore --staged
```

Image 17

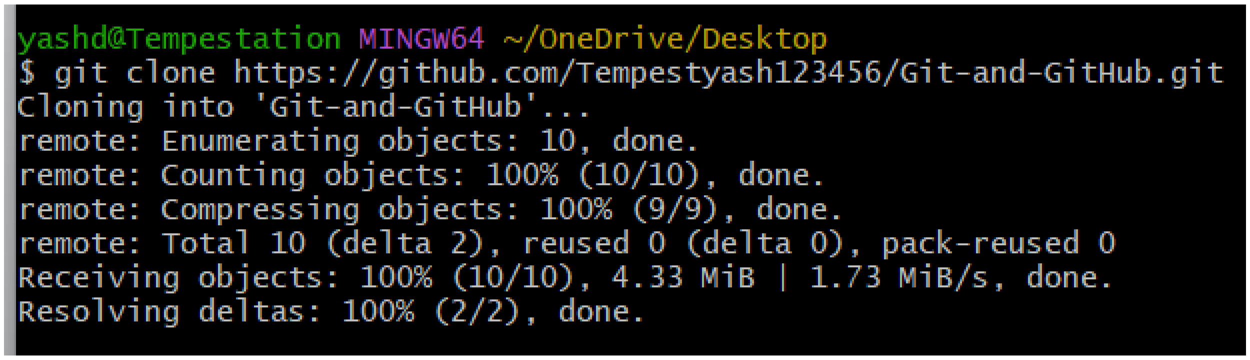
10. Now, we will be cloning a repository from github to our local environment using Git bash.

- Copy the HTTPS link of the repository from GitHub.
- Execute the following git command on git bash.

  git clone https://github.com/Tempestyash123456/Git-and-GitHub.git (in image 18)

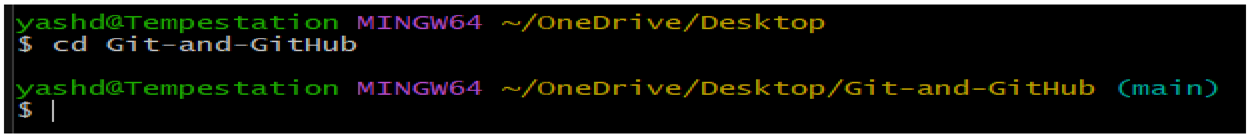
- Now, open the same repository on our git bash.

  cd Git-and-GitHub (in image 19)



```
yashd@Tempestation MINGW64 ~/OneDrive/Desktop
$ git clone https://github.com/Tempestyash123456/Git-and-GitHub.git
Cloning into 'Git-and-GitHub'...
remote: Enumerating objects: 10, done.
remote: Counting objects: 100% (10/10), done.
remote: Compressing objects: 100% (9/9), done.
remote: Total 10 (delta 2), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (10/10), 4.33 MiB | 1.73 MiB/s, done.
Resolving deltas: 100% (2/2), done.
```

Image 18



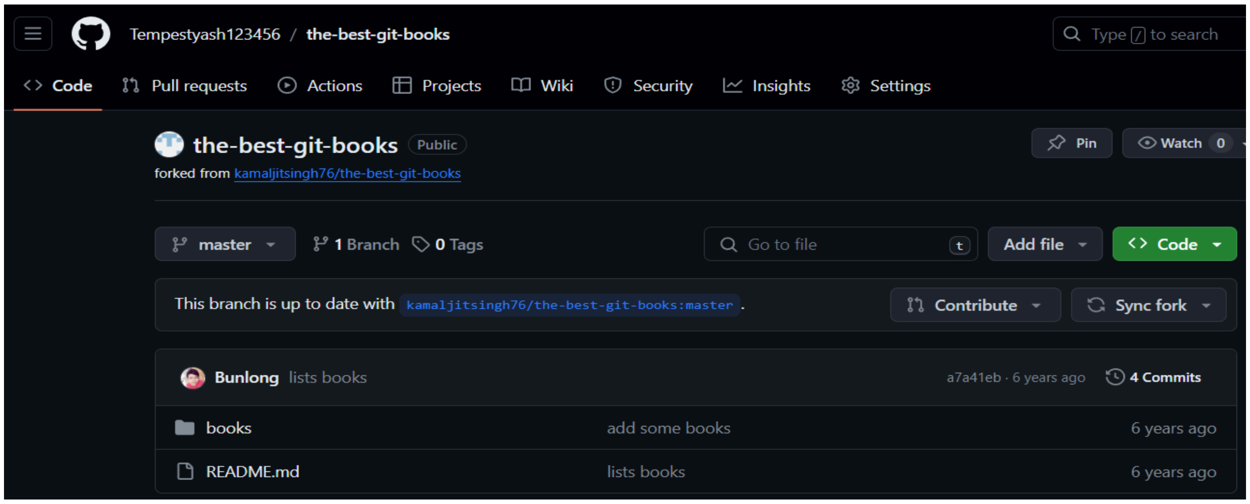
```
yashd@Tempestation MINGW64 ~/OneDrive/Desktop
$ cd Git-and-GitHub

yashd@Tempestation MINGW64 ~/OneDrive/Desktop/Git-and-GitHub (main)
$ |
```

Image 19

11. Now, we will be forking an existing repository as our own Github repository.

- Go to the repository.
- Click on the Fork button.
- Choose the destination and Repository name.
- Click on the **Create fork** button



the-best-git-books Public
forked from kamaljitsingh76/the-best-git-books

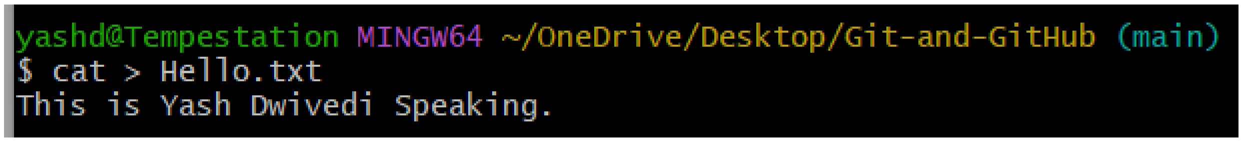
This branch is up to date with kamaljitsingh76/the-best-git-books:master .

Bunlong lists books	a7a41eb · 6 years ago	4 Commits
books	add some books	6 years ago
README.md	lists books	6 years ago

Image 20

12. Now, we will be creating a file on our local system and pushing it on Github.

- Create a file and write some content to it (in image 21).



```
yashd@Tempestation MINGW64 ~/OneDrive/Desktop/Git-and-GitHub (main)
$ cat > Hello.txt
This is Yash Dwivedi Speaking.
```

Image 21

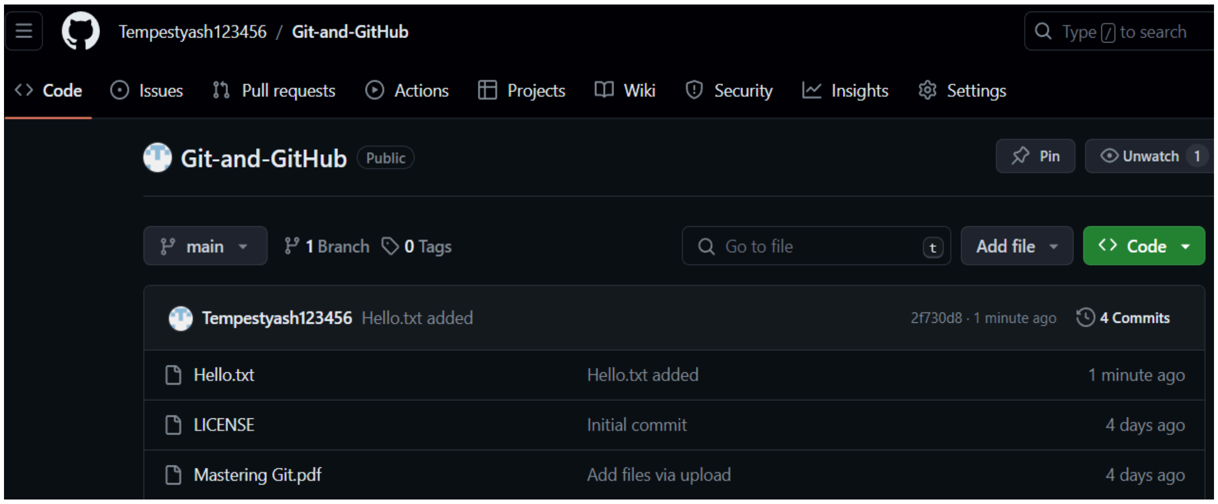
- Add the file to the staging area, commit the changes, and push it to the remote repository using the following git commands (in image 22).

  git add Hello.txt

  git commit -m "I have added Hello.txt file to the repository"

  git push origin main

- The file will be pushed onto the repository on the remote server.



Tempestyash123456 Hello.txt added 2f730d8 · 1 minute ago 4 Commits

Hello.txt	Hello.txt added	1 minute ago
LICENSE	Initial commit	4 days ago
Mastering Git.pdf	Add files via upload	4 days ago

Image 22

4. <u>Result/Output/Writing Summary</u>:
In this experiment we installed git, configured it with our GitHub account and write some commands such as clone to pull remote repository to our local machine, cd, cat, then add and commit to update changes to our remote repositories and forking a repository.

Learning outcomes (What I have learnt):

1. Learnt how to install git.

2. Learnt how to configure git with GitHub account.

3. Learnt about some basic commands such as cd and cat.

4. Learnt using git clone command.

5. Also learnt how to add and commit updates to the GitHub account.

Evaluation Grid (To be created as per the SOP and Assessment guidelines by the faculty):

Sr. No.	Parameters	Marks Obtained	Maximum Marks
1.			
2.			
3.			