Experiment 02 - Version Control

Roll No.	70
Name	MAYURI SHRIDATTA YERANDE
Class	D15-B
Subject	DevOps Lab
LO Mapped	LO1: To understand the fundamentals of DevOps engineering and be fully proficient with DevOps terminologies, concepts, benefits, and deployment options to meet your business requirements LO2: To obtain complete knowledge of the "version control system" to effectively track changes augmented with Git and GitHub

Aim: To understand Version Control System / Source Code Management, install git and create a GitHub account.

Introduction:

Version control

Software is developed to solve a user problem. Increasingly, these solutions have many different forms (e.g. mobile, embedded, SaaS) and run a variety of environments, such as cloud, on-prem, or Edge. As organizations accelerate delivery of their software solutions through DevOps, controlling and managing different versions of application artifacts - from code to configuration and from design to deployment - becomes increasingly difficult. Velocity without robust version control and traceability is like driving a car without a seatbelt.
Version control facilitates coordination, sharing, and collaboration across the entire software development team. Version control software enables teams to work in distributed and asynchronous environments, manage changes and versions of code and artifacts, and resolve merge conflicts and related anomalies.

Types of Version Control systems

- Distributed version control system

A distributed version control system (DVCS) is a type of version control system that allows users to access a repository from multiple locations. DVCSs are often used by developers who need to work on projects from multiple computers or who need to collaborate with other developers remotely.

- Centralized version control system

A centralized version control system (CVCS) is a type of version control system (VCS) where all users are working with the same central repository. This central repository can be located on a server or on a developer's local machine. CVCSs are typically used in software development projects where a team of developers needs to share code and track changes.

- Lock-based version control system

A lock-based version control system is a type of version control system that uses locks to manage concurrent access to files and resources. Locking prevents two or more users from making conflicting changes to the same file or resource.

- Optimistic version control system

In an optimistic version control system, every user has their own private workspace. When they want to share their changes with the rest of the team, they submit a request to the server. The server then looks at all the changes and determines which ones can be safely merged together

Benefits of version control

- Quality
  Teams can review, comment, and improve each other's code and assets.
- Acceleration
  Branch code, make changes, and merge commits faster.
- Visibility
  Understand and spark team collaboration to foster greater release build and release patterns.

Importance of Version Control

Version control plays an important role in keeping track of the changes and every group member working off the latest version of the code. The version control software can be used for all files, code, and assets that several group members will work together on. Version control systems do more than just tracking and managing files. It helps to develop and ship the products faster. This is important, especially for teams who practice DevOps. Using the right version control system does the following things;

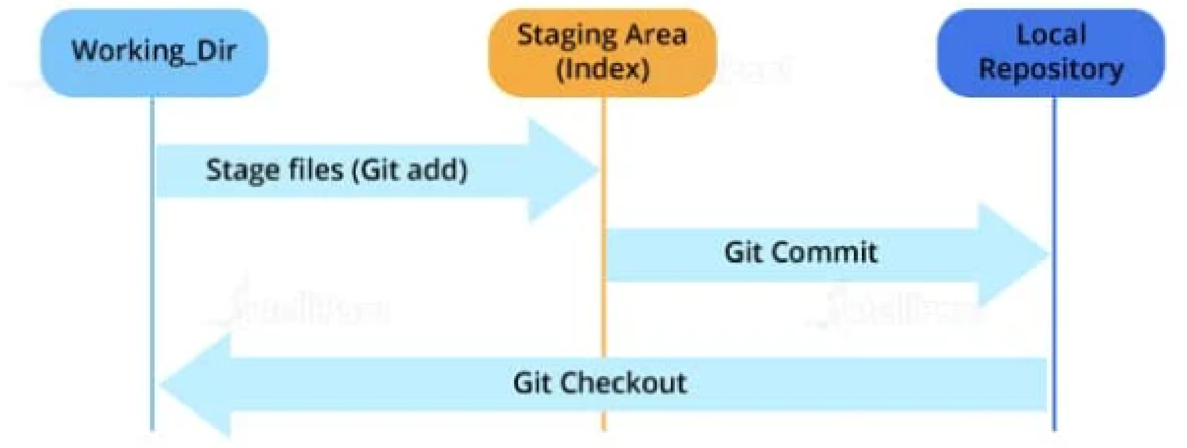
- Accelerates the delivery of the product
- Improves visibility
- And helps teams collaborating across the world.

Git:

Introduction

When you check for the definition of Git online, the best you can go something along these lines:Gitis a distributed version control system (DVCS) for tracking changes to files. But what does that mean? Git is an open-source VCS, which is not file-based, unlike other systems. Rather, it stores information as snapshots. Being a VCS, helps coders revert to their previous code when they hit a roadblock in the newer version, without affecting the original source code. On the other hand, what makes it different from other VCS is the way it sees data, which is more like a series of snapshots. It basically clicks a picture of how all your files look at the moment and saves the changes made to them over time.

Git Architecture



The three layers are:

- **Working directory:** This is created when a Git project is initialized onto your local machine and allows you to edit the source code copied.
- **Staging area:** Post the edits, the code is staged in the staging area by applying the command, git add. This displays a preview for the next stage. In case further modifications are made in the working directory, the snapshots for these two layers will be different. However, these can be synced by using the same 'git add' command.
- **Local repository:** If no further edits are required to be done, then you can go ahead and apply the git commit command. This replicates the latest snapshots in all three stages, making them in sync with each other.

Git Terminologies

Branch:- A branch is a version of the repository that diverges from the main working project. It is an essential feature available in most modern version control systems. A Git project can have more than one branch. We can perform many operations on Git branch-like rename, list, delete, etc.ge the whole branch. You can revert the commit and cherry-pick it on another branch.

Clone:- The **git clone** is a Git command-line utility. It is used to make a copy of the target repository or clone it. If I want a local copy of my repository from GitHub, this tool allows creating a local copy of that repository on your local directory from the repository URL.

Fetch:- It is used to fetch branches and tags from one or more other repositories, along with the objects necessary to complete their histories. It updates the remote-tracking branches.

HEAD:- HEAD is the representation of the last commit in the current checkout branch. We can think of the head like a current branch. When you switch branches with git checkout, the HEAD revision changes, and points the new branch.

Index:- The Git index is a staging area between the working directory and repository. It is used as the index to build up a set of changes that you want to commit together.

Origin:- In Git, "origin" is a reference to the remote repository from a project was initially cloned. More precisely, it is used instead of that original repository URL to make referencing much easier.

Pull/Pull Request:- The term Pull is used to receive data from GitHub. It fetches and merges changes on the remote server to your working directory. The **git pull command** is used to make a Git pull.

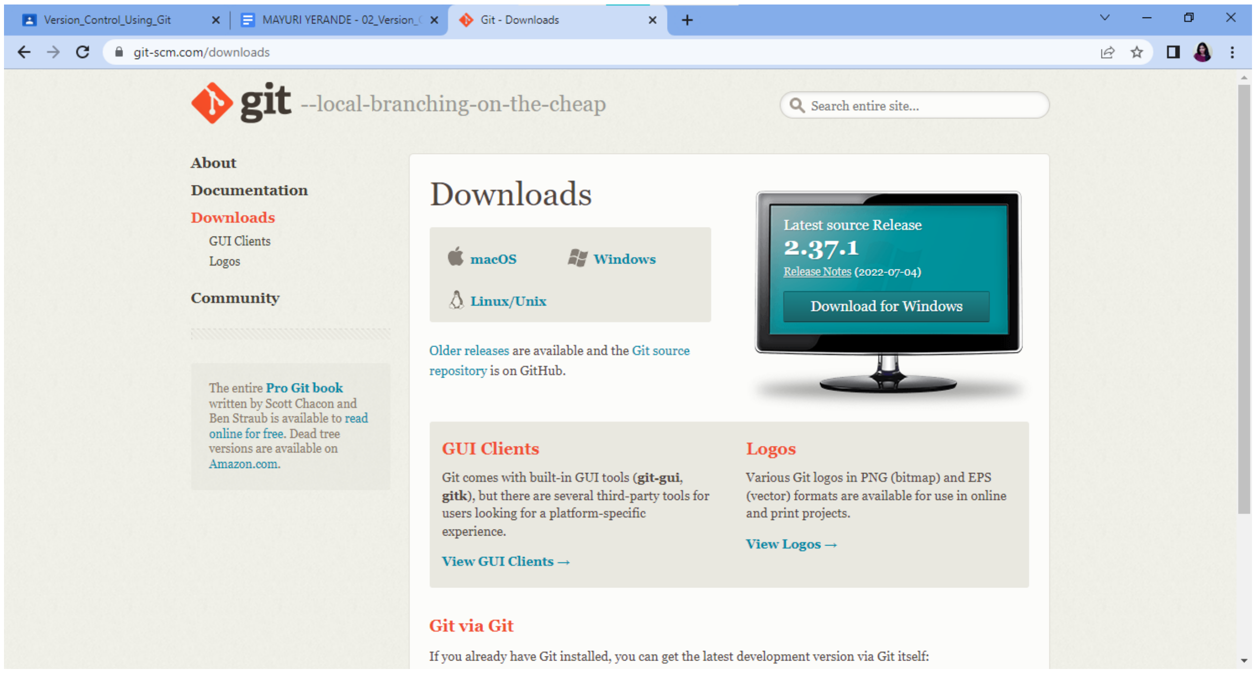
Pull requests are a process for a developer to notify team members that they have completed a feature. Once their feature branch is ready, the developer files a pull request via their remote server account. Pull request announces all the team members that they need to review the code and merge it into the master branch.

Push:- The push term refers to upload local repository content to a remote repository. Pushing is an act of transfer commits from your local repository to a remote repository. Pushing is capable of overwriting changes; caution should be taken when

Installation:

Link:- https://git-scm.com/downloads

- **Choose your operating system**



Downloads

macOS Windows
Linux/Unix

Latest source Release
2.37.1
Release Notes (2022-07-04)
Download for Windows

Older releases are available and the Git source repository is on GitHub.

The entire Pro Git book written by Scott Chacon and Ben Straub is available to read online for free. Dead tree versions are available on Amazon.com.

GUI Clients

Git comes with built-in GUI tools (git-gui, gitk), but there are several third-party tools for users looking for a platform-specific experience.

View GUI Clients →

Logos

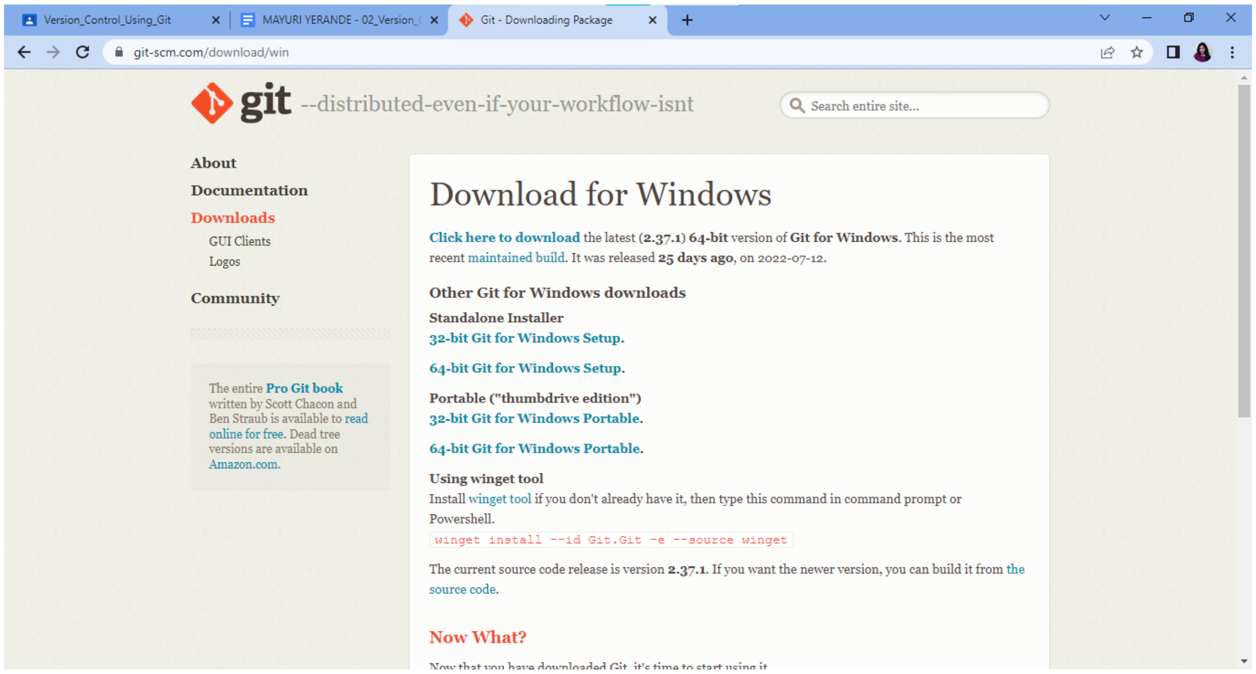
Various Git logos in PNG (bitmap) and EPS (vector) formats are available for use in online and print projects.

View Logos →

Git via Git

If you already have Git installed, you can get the latest development version via Git itself:

- **Choose 64 or 32 bit based on your device and choose the standalone installer**



Version_Control_Using_Git
MAYURI YERANDE - 02_Version_
Git - Downloading Package
git-scm.com/download/win
git --distributed-even-if-your-workflow-isnt
Search entire site...
About
Documentation
Downloads
GUI Clients
Logos
Community

Download for Windows

Click here to download the latest (2.37.1) 64-bit version of Git for Windows. This is the most recent maintained build. It was released 25 days ago, on 2022-07-12.

Other Git for Windows downloads

Standalone Installer
32-bit Git for Windows Setup.

64-bit Git for Windows Setup.

Portable ("thumbdrive edition")
32-bit Git for Windows Portable.

64-bit Git for Windows Portable.

Using winget tool
Install winget tool if you don't already have it, then type this command in command prompt or Powershell.

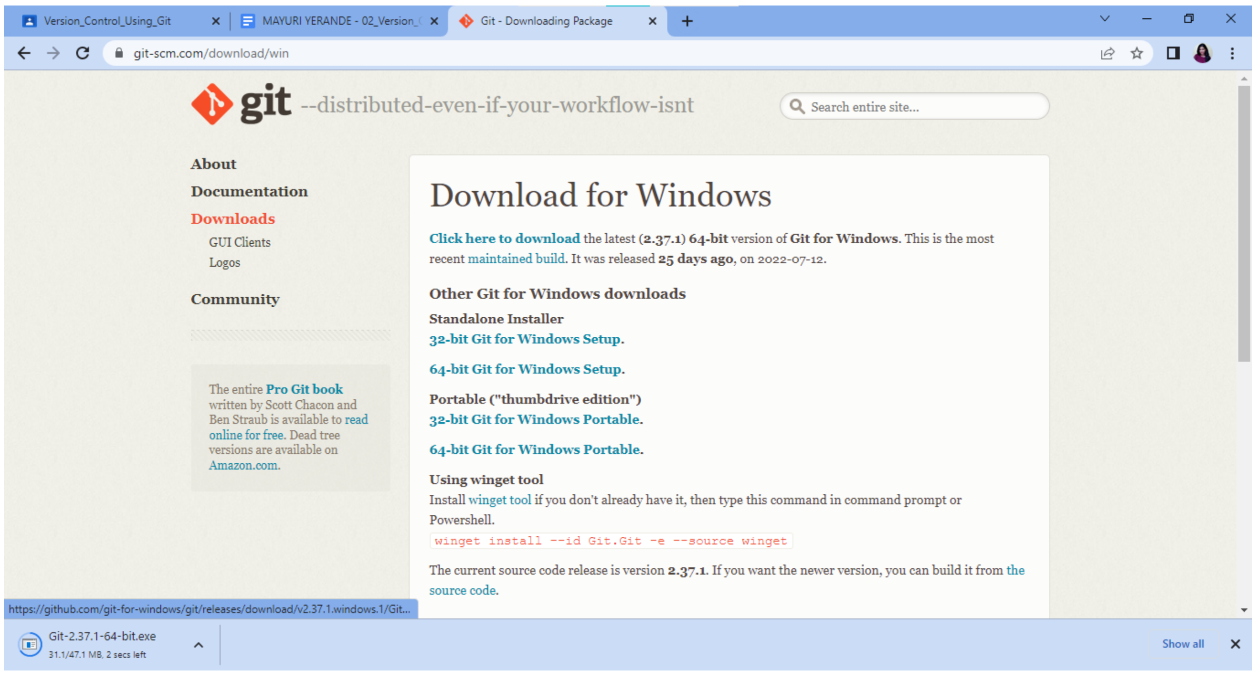
```
winget install --id Git.Git -e --source winget
```

The current source code release is version 2.37.1. If you want the newer version, you can build it from the source code.

Now What?

The entire Pro Git book written by Scott Chacon and Ben Straub is available to read online for free. Dead tree versions are available on Amazon.com.

- **Git will start downloading automatically**


Version_Control_Using_Git
MAYURI YERANDE - 02_Version_
Git - Downloading Package
git-scm.com/download/win
git --distributed-even-if-your-workflow-isnt
Search entire site...
About
Documentation
Downloads
GUI Clients
Logos
Community

The entire **Pro Git book** written by Scott Chacon and Ben Straub is available to read online for free. Dead tree versions are available on Amazon.com.

Download for Windows

Click here to download the latest (**2.37.1**) **64-bit** version of **Git for Windows**. This is the most recent maintained build. It was released **25 days ago**, on 2022-07-12.

Other Git for Windows downloads

Standalone Installer

32-bit Git for Windows Setup.

64-bit Git for Windows Setup.

Portable ("thumbdrive edition")

32-bit Git for Windows Portable.

64-bit Git for Windows Portable.

Using winget tool

Install winget tool if you don't already have it, then type this command in command prompt or Powershell.

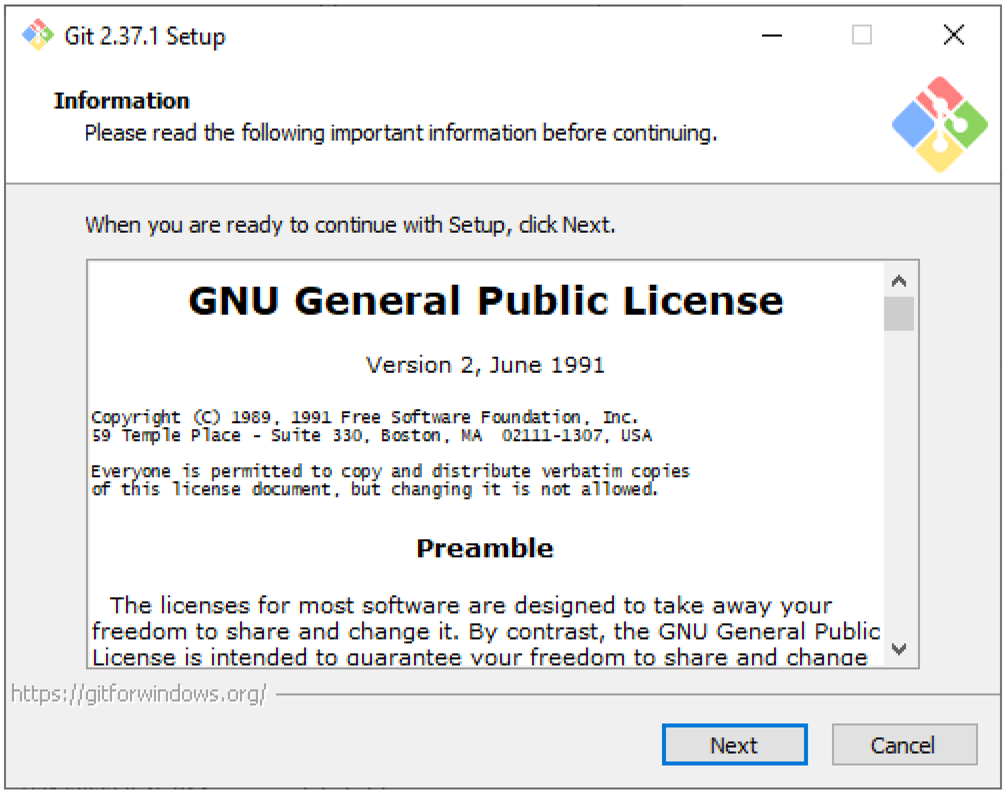
```
winget install --id Git.Git -e --source winget
```

The current source code release is version **2.37.1**. If you want the newer version, you can build it from the source code.

https://github.com/git-for-windows/git/releases/download/v2.37.1.windows.1/Git...

Git-2.37.1-64-bit.exe
31.1/47.1 MB, 2 secs left

- **Open the downloaded file and this screen will get displayed**



Git 2.37.1 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

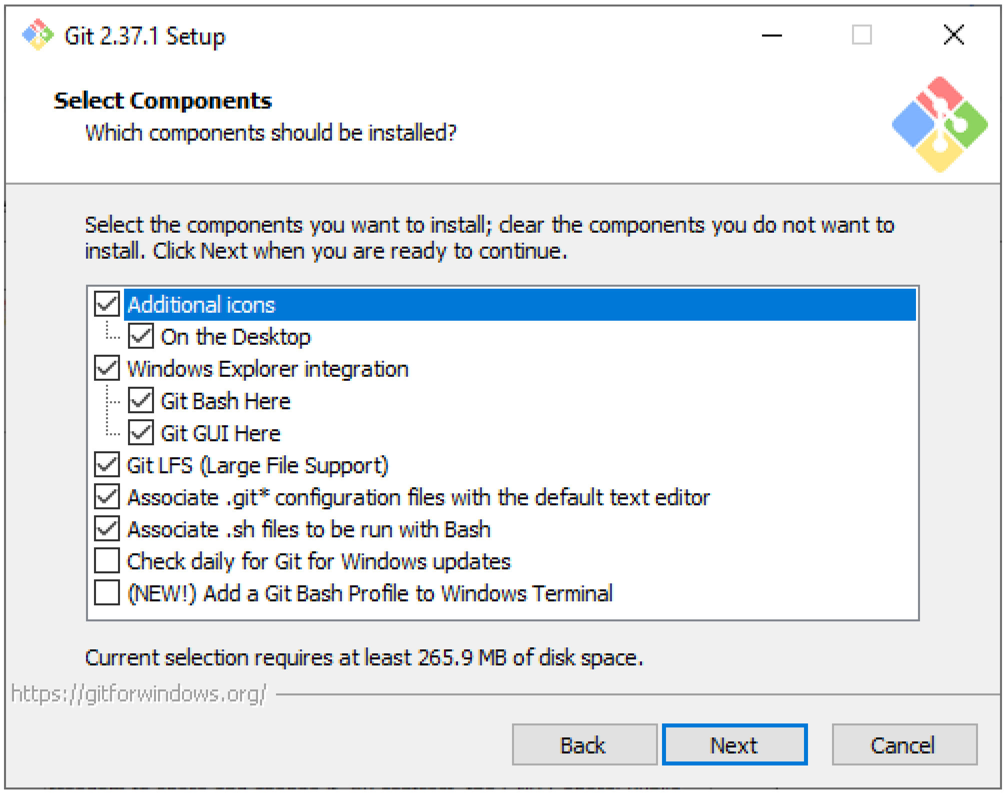
```
Copyright (C) 1989, 1991 Free Software Foundation, Inc.
59 Temple Place - Suite 330, Boston, MA  02111-1307, USA

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

- **Click on the "Additional icons" to get git icon on your desktop**



Git 2.37.1 Setup

Select Components

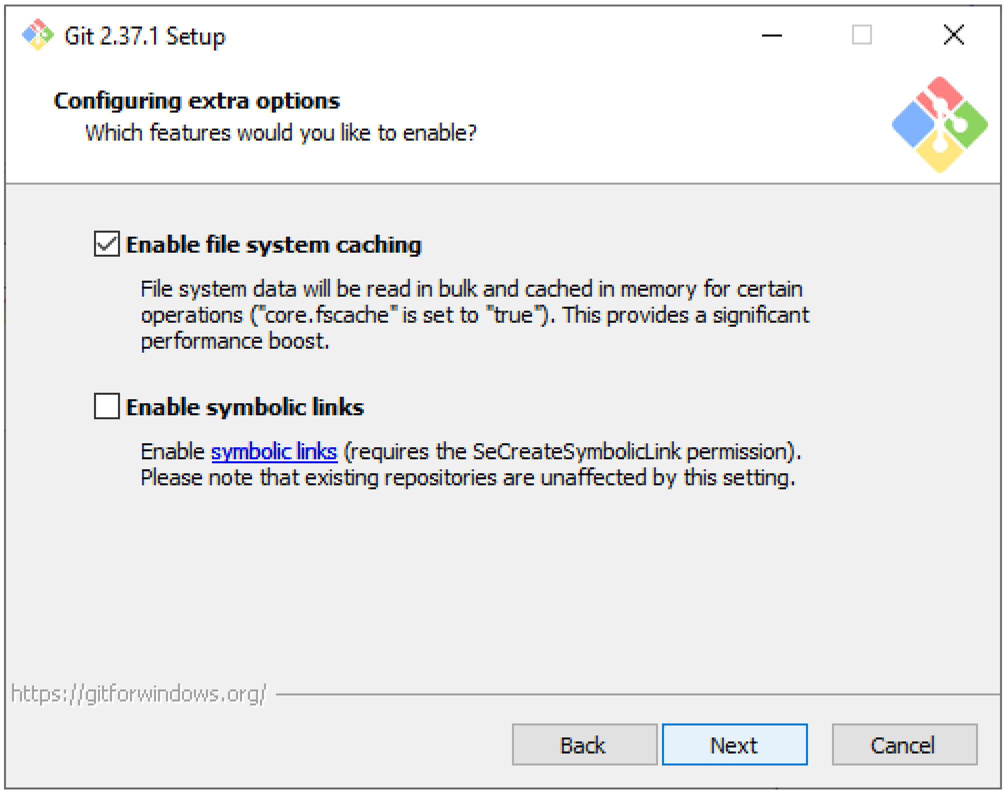
Which components should be installed?

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

- ☑ Additional icons
  - ☑ On the Desktop
- ☑ Windows Explorer integration
  - ☑ Git Bash Here
  - ☑ Git GUI Here
- ☑ Git LFS (Large File Support)
- ☑ Associate .git* configuration files with the default text editor
- ☑ 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 265.9 MB of disk space.

- **Click on "next"**



Git 2.37.1 Setup

Configuring extra options

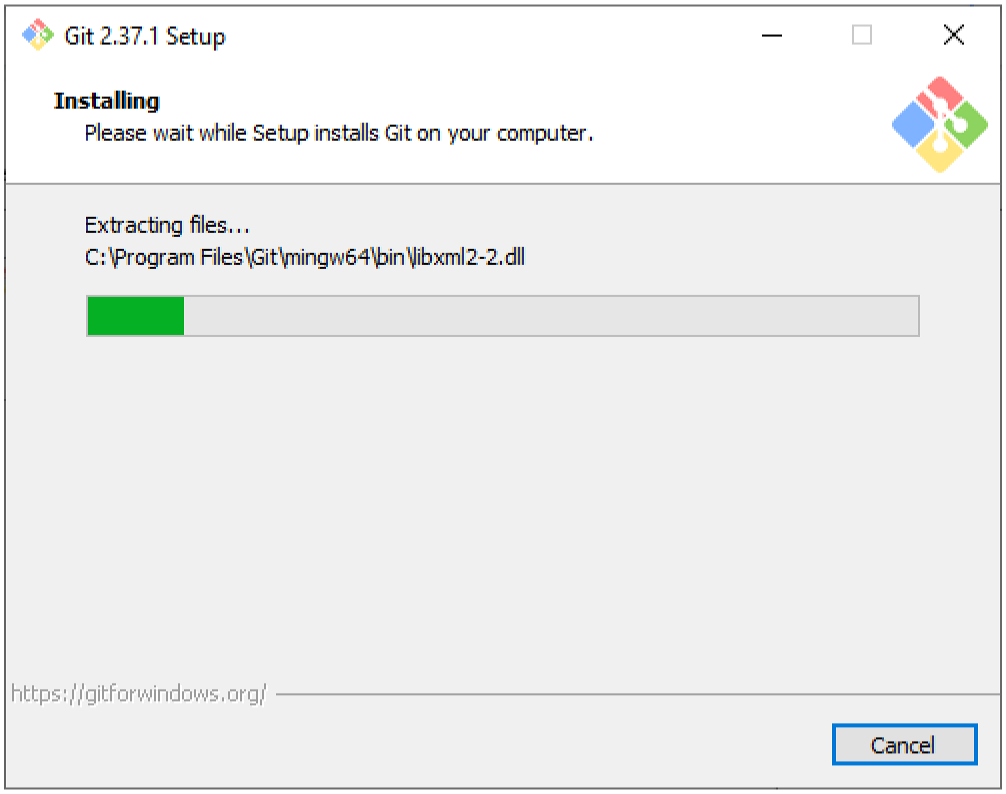
Which features would you like to enable?

☑ **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.



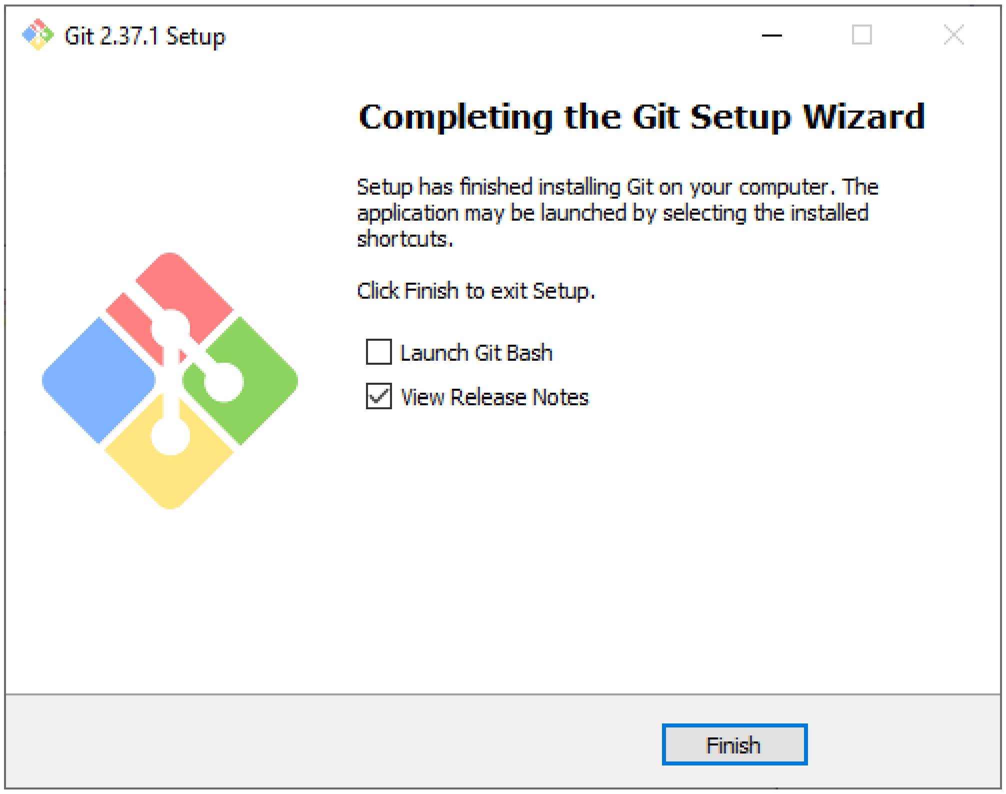
Git 2.37.1 Setup

Installing

Please wait while Setup installs Git on your computer.

Extracting files...
C:\Program Files\Git\mingw64\bin\libxml2-2.dll

https://gitforwindows.org/



Git 2.37.1 Setup

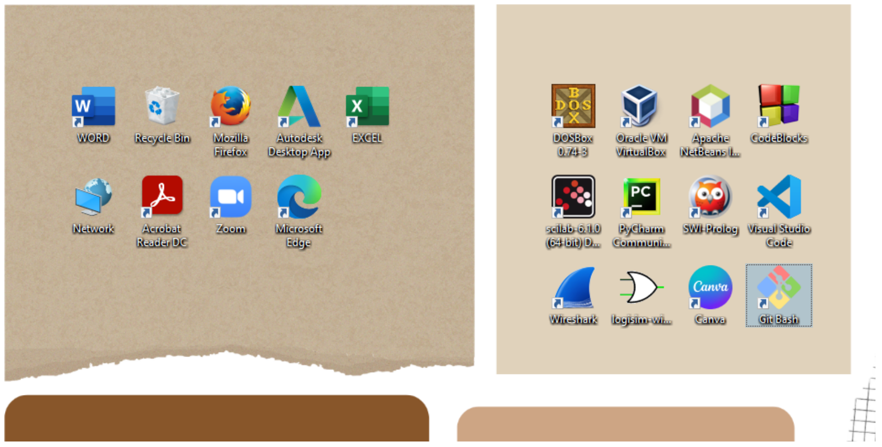
Completing the Git Setup Wizard

Setup has finished installing Git on your computer. The application may be launched by selecting the installed shortcuts.

Click Finish to exit Setup.

- [ ] Launch Git Bash
- [x] View Release Notes

- **Click on "finish"**
- **You will see the git icon on you. Thus git is successfully installed.**



Github:

- **Go to https://github.com/join in a web browser.** You can use any web browser on your computer, phone, or tablet to join.

- **Enter your personal details.** In addition to creating a username and entering an email address, you'll also have to create a password. Your password must be at least 15 characters in length *or* at least 8 characters with at least one number and lowercase letter.

- **Click the green Create an account button.** It's below the form.
  Complete the CAPTCHA puzzle. The instructions vary by puzzle, so just follow the on-screen instructions to confirm that you are a human.

- **Click the Choose button for your desired plan.** Once you select a plan, GitHub will send an email confirmation message to the address you entered. The plan options are

  Free: Unlimited public and private repositories, up to 3 collaborators, issues and bug tracking, and project management tools.

  Pro: Unlimited access to all repositories, unlimited collaborators, issue & bug tracking, and advanced insight tools.

  Team: All of the aforementioned features, plus team access controls and user management.

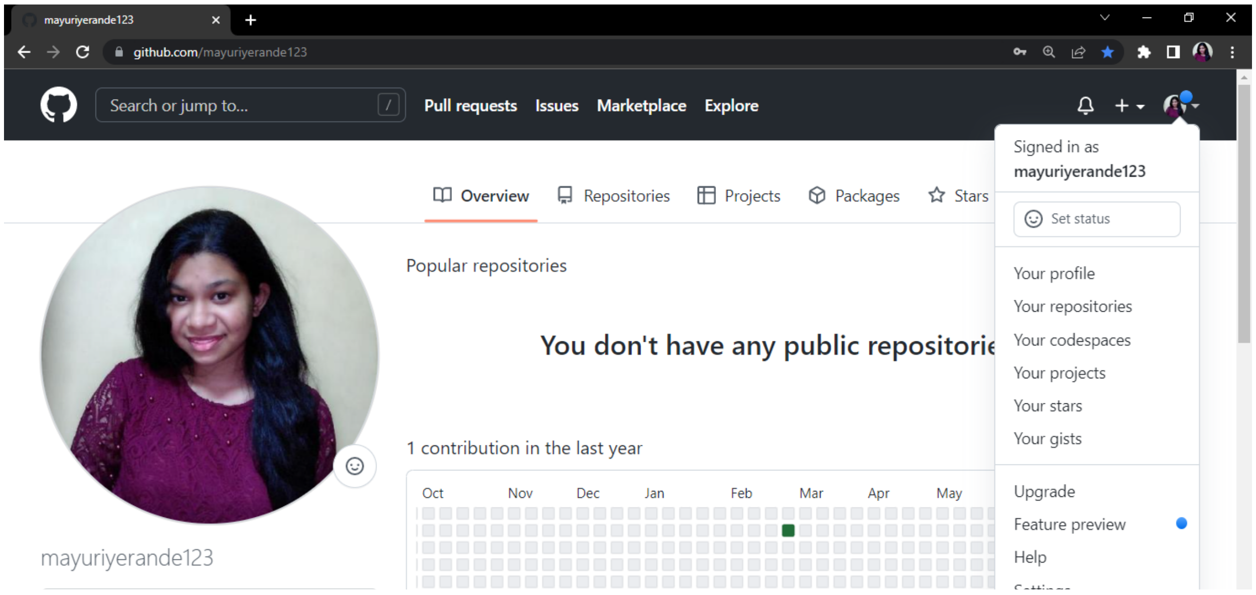
Enterprise: All of the features of the Team plan, plus self-hosting or cloud hosting, priority support, single sign-on support, and more.

- **Click the Verify email address button in the message from GitHub.** This confirms your email address and returns you to the sign-up process.

- **Review your plan selection and click Continue.** You can also choose whether you want to receive updates from GitHub via email by checking or unchecking the "Send me updates" box. If you chose a paid plan, you'll have to enter your payment information as requested before you can continue.

- **Select your preferences and click Submit.** GitHub displays a quick survey that can help you tailor your experience to match what you're looking for. Once you make your selection, you'll be taken to a screen that allows you to set up your first repository.

- Your account is ready to use.



Popular repositories

You don't have any public repositorie

1 contribution in the last year

mayuriyerande123

<u>**Conclusion**</u>: We have successfully learnt and implemented version control,git and github.