Exp No. 1	Git Essentials – Commit, Push, Merge, and Pull Requests
Date:	

Aim

The aim is to demonstrate pushing changes to a GitHub repository, covering setup, managing visibility, and executing basic Git operations**.**

Description

The aim of the experiment is to demonstrate the process of pushing changes to a GitHub repository. This involves:

1. Setting up Git: Ensure Git is installed on your computer.
2. Managing Repository Visibility: Understand the effects of making a repository private or public on GitHub.
3. Changing Repository Visibility: Learn how to change the visibility of a repository on GitHub.
4. Pushing Changes to the Repository: Make changes to a local repository, commit them, and push them to the remote repository on GitHub**.**

Introduction of GIT

Git is one of the ways of implementing the idea of version control. It is Distributed Version Control System.

Installing GIT

Before you start using Git, you have to make it available on your computer.

it's already installed, it's probably a good idea to update to the latest version. You can either install it as a package or via another installer, or download the source code and compile it yourself.

Installing on Windows

There are also a few ways to install Git on Windows. The most official build is available for download on the Git website. Just go to https://git-scm.com/download/win and the download will start automatically

To get an automated installation you can use the Git Chocolatey package.

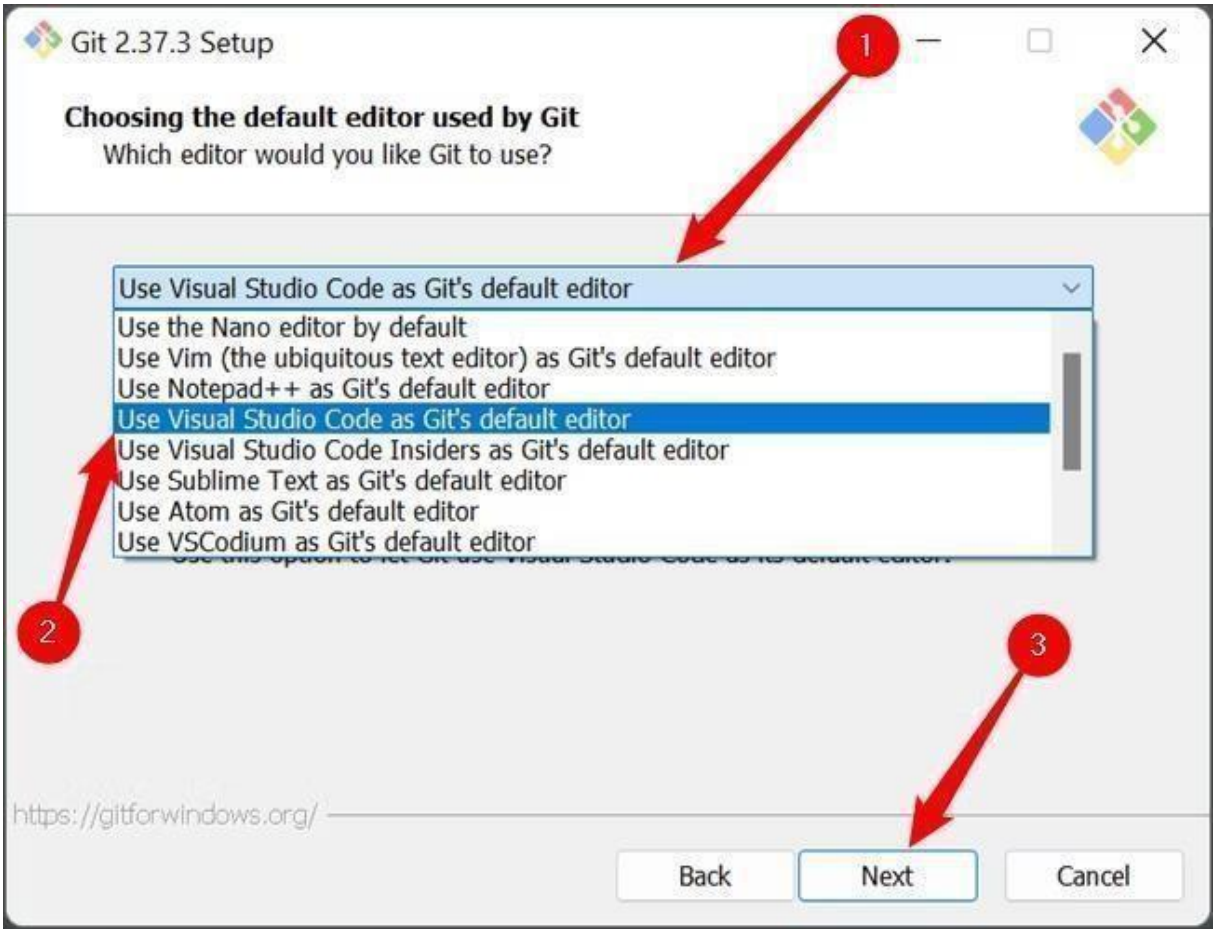
The easiest way to get Git is to download the executable from the Git website.

Click "64-bit Git for Windows Setup" to start the download, and then wait a moment — the download is only about 50 megabytes, so it shouldn't take very long.

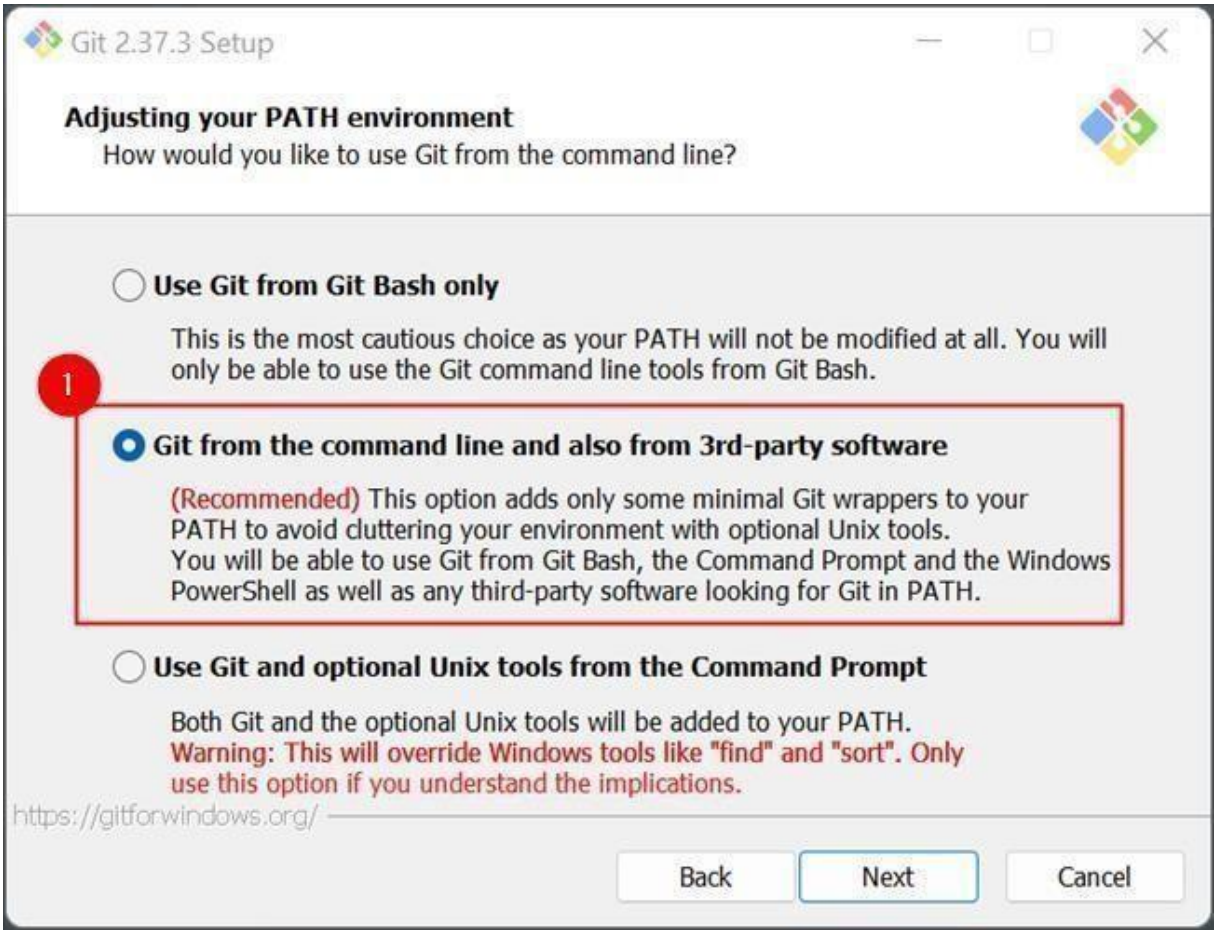
Double-click the executable you just downloaded, then click "Next" to move through the installation prompts.

The first is the text editor Git will use. The default selection is Vim. Vim is ubiquitous and a hallmark of command-line interfaces everywhere but learning to use its idiosyncratic

commands can be daunting. You should probably pick something else instead, like VisualStudio Code, Sublime, NotePad++, or any other plain text editor you like.



The second is the way Git integrates itself into your PC's PATH. Make sure that the "Git From The Command Line And Also From 3rd-Party Software" is selected.



Click through the remaining options, and wait for everything to finish downloading. The time requires to download everything will vary depending on what you chose to install. The default selection results in a download that is about 270 megabytes.

Managing Private and Public RepositoryMaking a repository Private

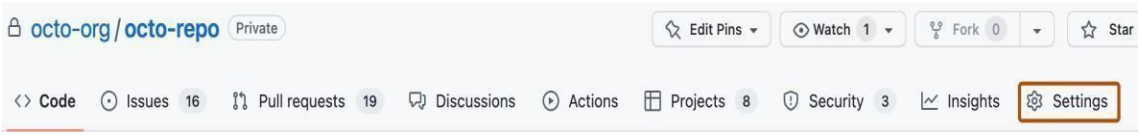
- GitHub will detach public forks of the public repository and put them into a new network. Public forks are not made private.
- If you're using GitHub Free for personal accounts or organizations, some features won't be available in the repository after you change the visibility to private. Any published GitHub Pages site will be automatically unpublished. If you added a customdomain to the GitHub Pages site, you should remove or update your DNS records before making the repository private, to avoid the risk of a domain takeover.
- GitHub will no longer include the repository in the GitHub Archive Program.
- GitHub Advanced Security features, such as code scanning, will stop working.

Making a repository Public

- GitHub will detach private forks and turn them into a standalone private repository.
- If you're converting your private repository to a public repository as part of a move toward creating an open source project.
- Once your repository is public, you can also view your repository's community profile to see whether your project meets best practices for supporting contributors.
- The repository will automatically gain access to GitHub Advanced Security features.

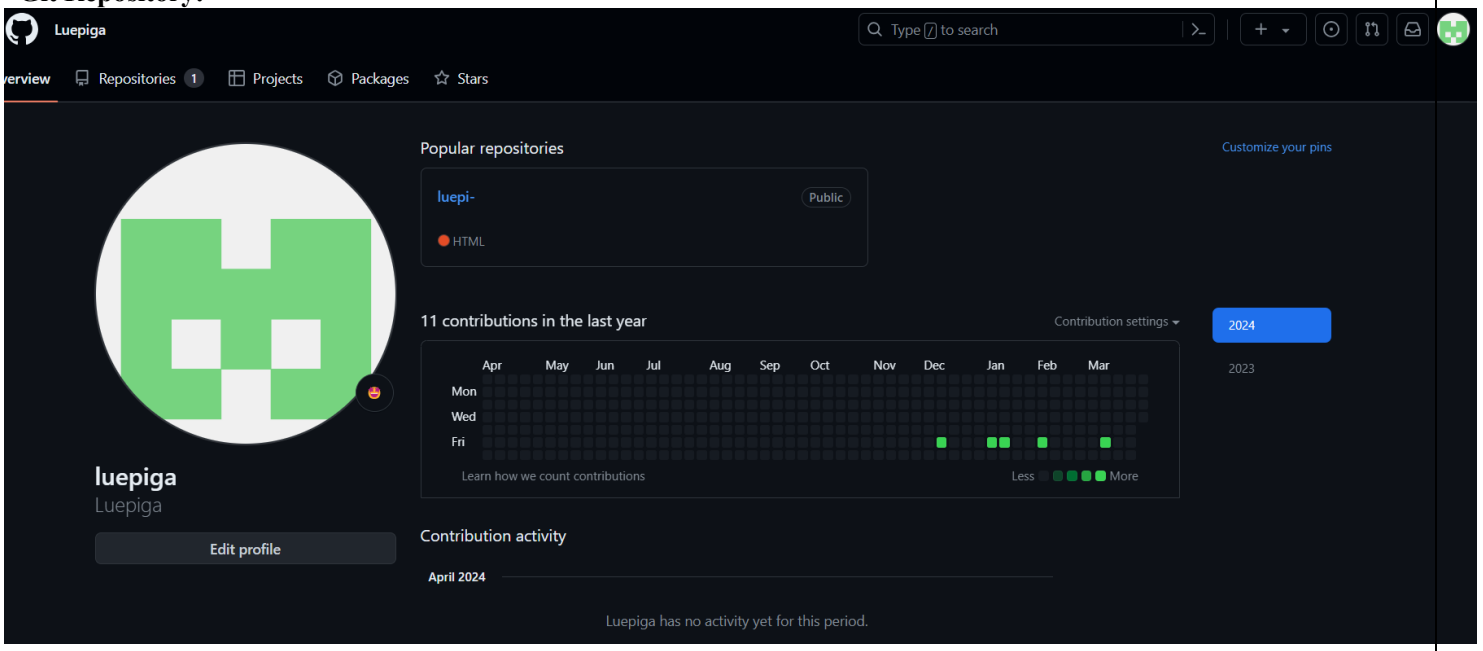
Changing a repository's Visibility

1. On GitHub.com, navigate to the main page of the repository.
2. Under your repository name, click **Settings**. If you cannot see the "Settings" tab, select the dropdown menu, then click **Settings**.



3. In the "Danger Zone" section, to the right of to "Change repository visibility", click **Change visibility**.
4. Select a visibility.
5. To verify that you're changing the correct repository's visibility, type the name of the repository you want to change the visibility of.
6. Click **I understand, change repository visibility**.

Git Repository:



Git Basic Commands:

1. **Initializing a Repository:**
git init: Initializes a new Git repository in the current directory. This creates a hidden .git directory to store the repository's data.
Code

```
git init
```

2. **Cloning a Repository:**
git clone <repository_url>: Creates a local copy of an existing remote repository.
Code

```
git clone https://github.com/user/repository.git
```

3. **Staging Changes:**
git add <file_name>: Adds specific files to the staging area, preparing them for the next commit.
git add . or git add -A: Adds all modified and new files in the working directory to the staging area.
Code

```
git add index.html
git add .
```

4. **Committing Changes:**
git commit -m "Commit message": Records the staged changes to the repository's history with a descriptive message.
Code

```
git commit -m "Initial commit of the project"
```

5. **Checking Status:**
git status: Shows the status of your working directory and staging area, indicating untracked, modified, and staged files.
Code

```
git status
```

6. **Viewing History:**

git log: Displays a detailed history of commits in the current branch.
Code

```
git log
```

7. **Working with Branches:**
git branch: Lists all local branches.
git branch <branch_name>: Creates a new branch.
git checkout <branch_name>: Switches to an existing branch.
git checkout -b <new_branch_name>: Creates a new branch and switches to it.
Code

```
git branch feature/new-feature
git checkout feature/new-feature
git checkout -b bugfix/fix-login
```

8. **Merging Branches:**
git merge <branch_name>: Merges the specified branch's history into the current branch.
Code

```
git merge feature/new-feature
```

9. **Interacting with Remote Repositories:**
git remote add origin <remote_repository_url>: Connects your local repository to a remote repository (commonly named 'origin').
git push origin <branch_name>: Pushes your local commits to the remote repository.
git pull origin <branch_name>: Fetches and merges changes from the remote repository into your local branch.
Code

```
git remote add origin https://github.com/user/repository.git
git push origin main
git pull origin main
```