

GIT Installation

Download Git:

<https://git-scm.com/downloads>



Download for Windows

[Click here to download](#) the latest (2.45.0) 64-bit version of Git for Windows. This is the most recent maintained build. It was released 8 days ago, on 2024-04-29.


Other Git for Windows downloads

Standalone Installer

32-bit Git for Windows Setup.

64-bit Git for Windows Setup.

Installer File (Git-xxx.exe file)



Git-2.45.0-64-bit.
exe

Git 2.45.0 Setup

Information

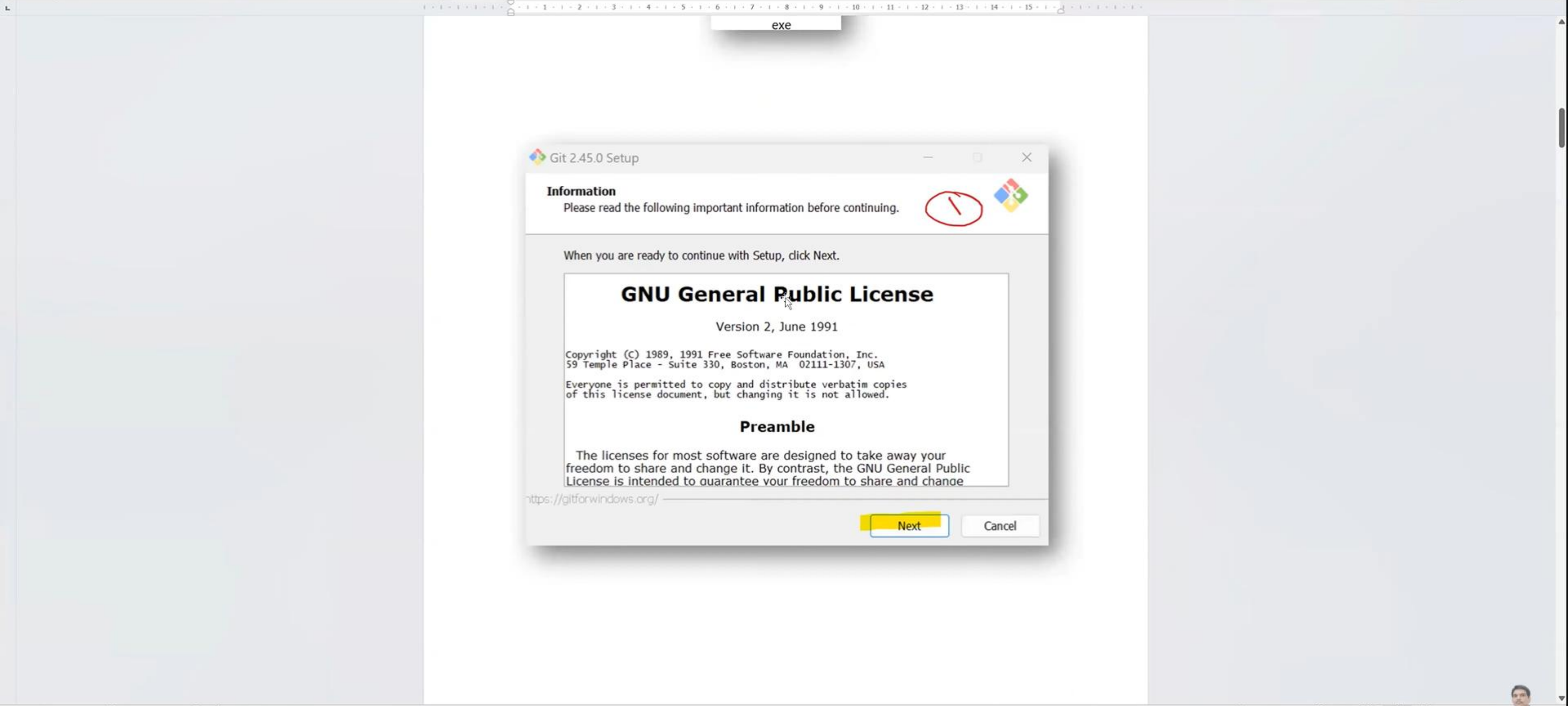
Please read the following important information before continuing.

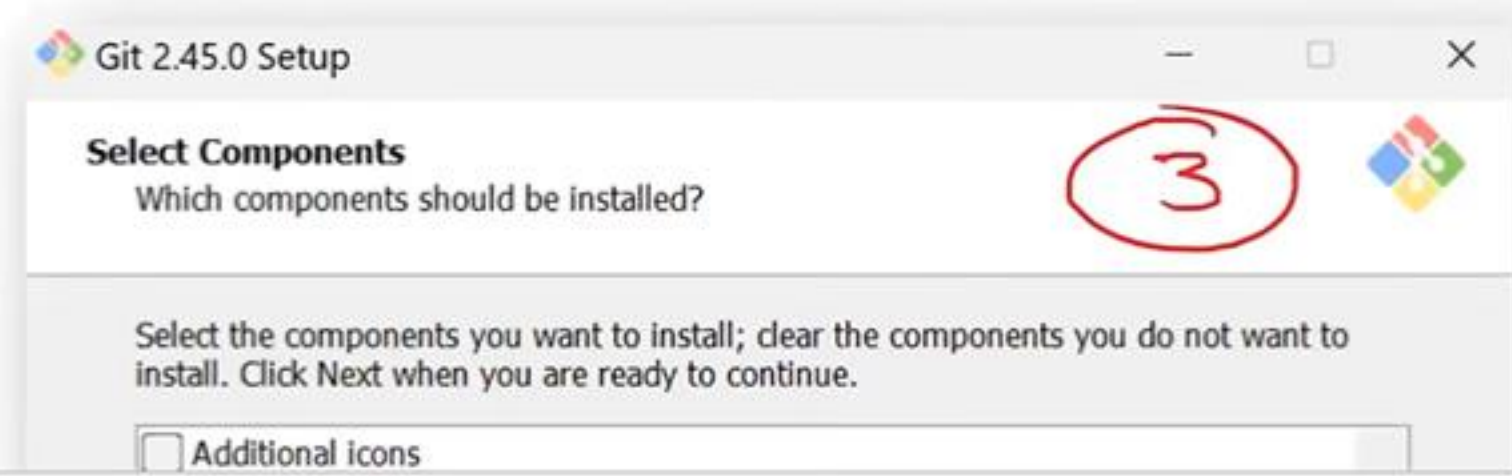
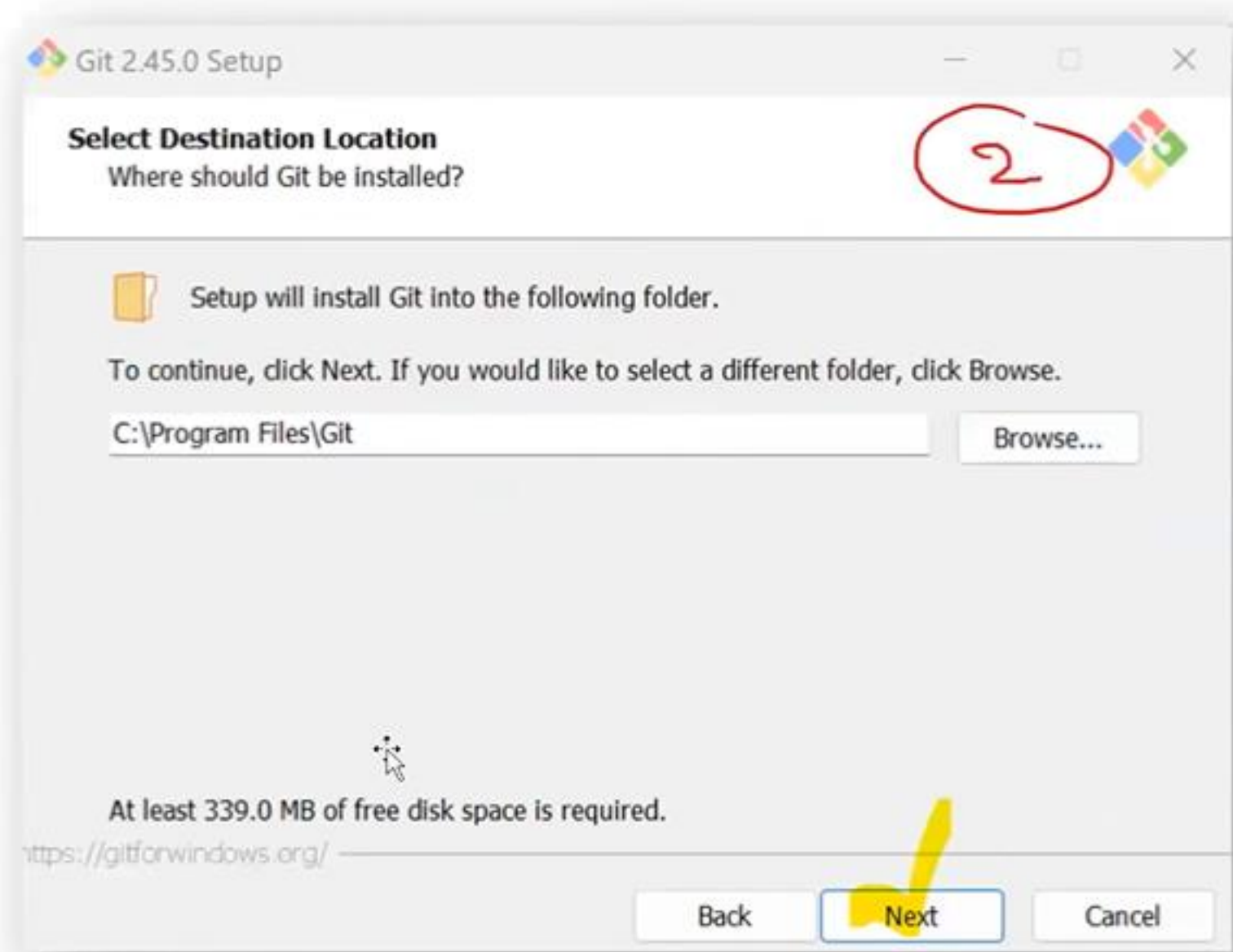
When you are ready to continue with Setup, click Next.

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





Back Next Cancel

Git 2.45.0 Setup

Select Components

Which components should be installed?

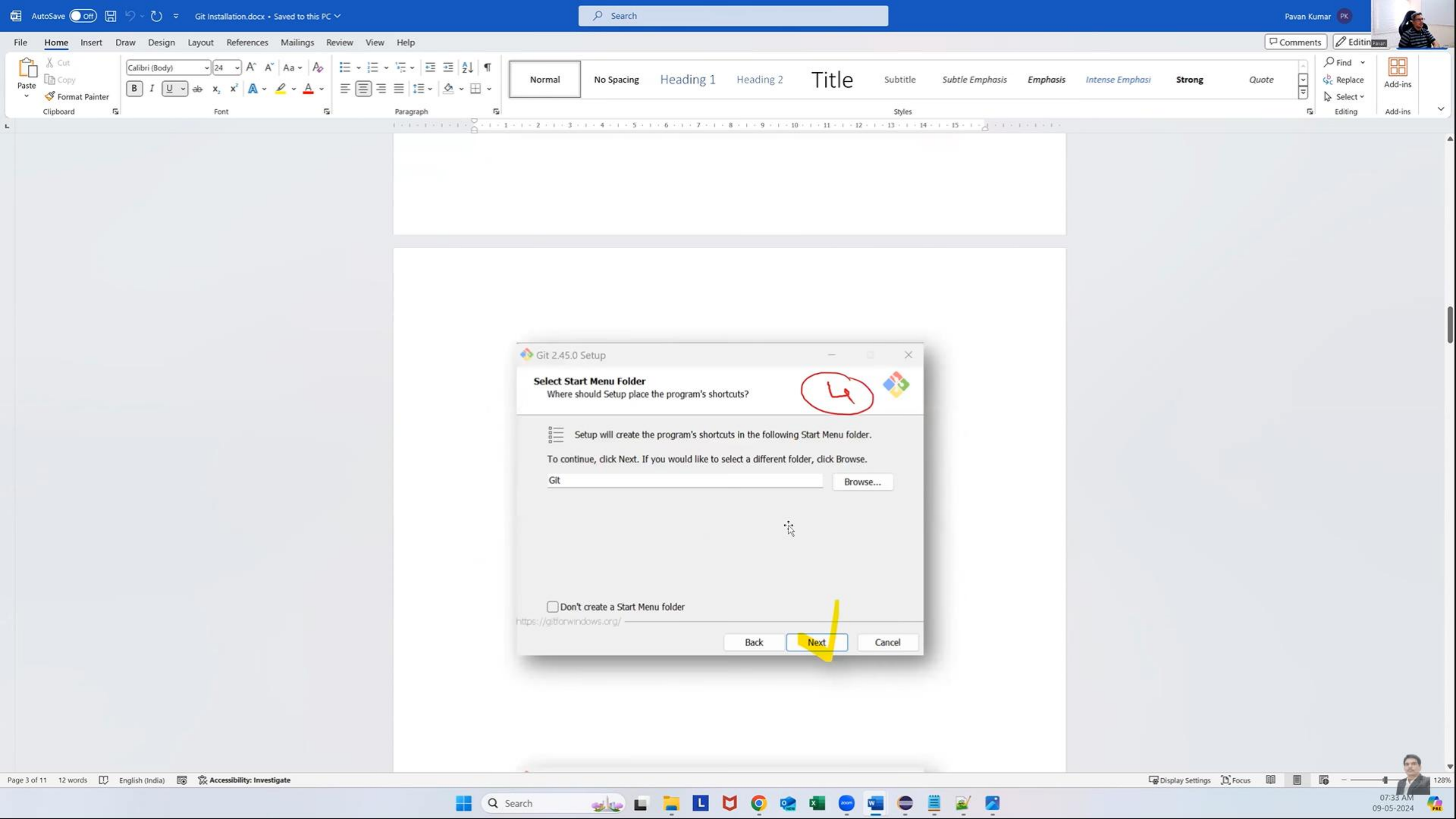
Select the components you want to install; clear the components you do not want to install. Click Next when you are ready to continue.

- ☐ Additional Icons
 - ☐ On the Desktop
- ☒ Windows Explorer integration
 - ☒ Open Git Bash here
 - ☒ Open 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 339.0 MB of disk space.

<https://gitforwindows.org/>

Back Next Cancel



Back Next Cancel

Git 2.45.0 Setup

Choosing the default editor used by Git

Which editor would you like Git to use?

Use Vim (the ubiquitous text editor) as Git's default editor

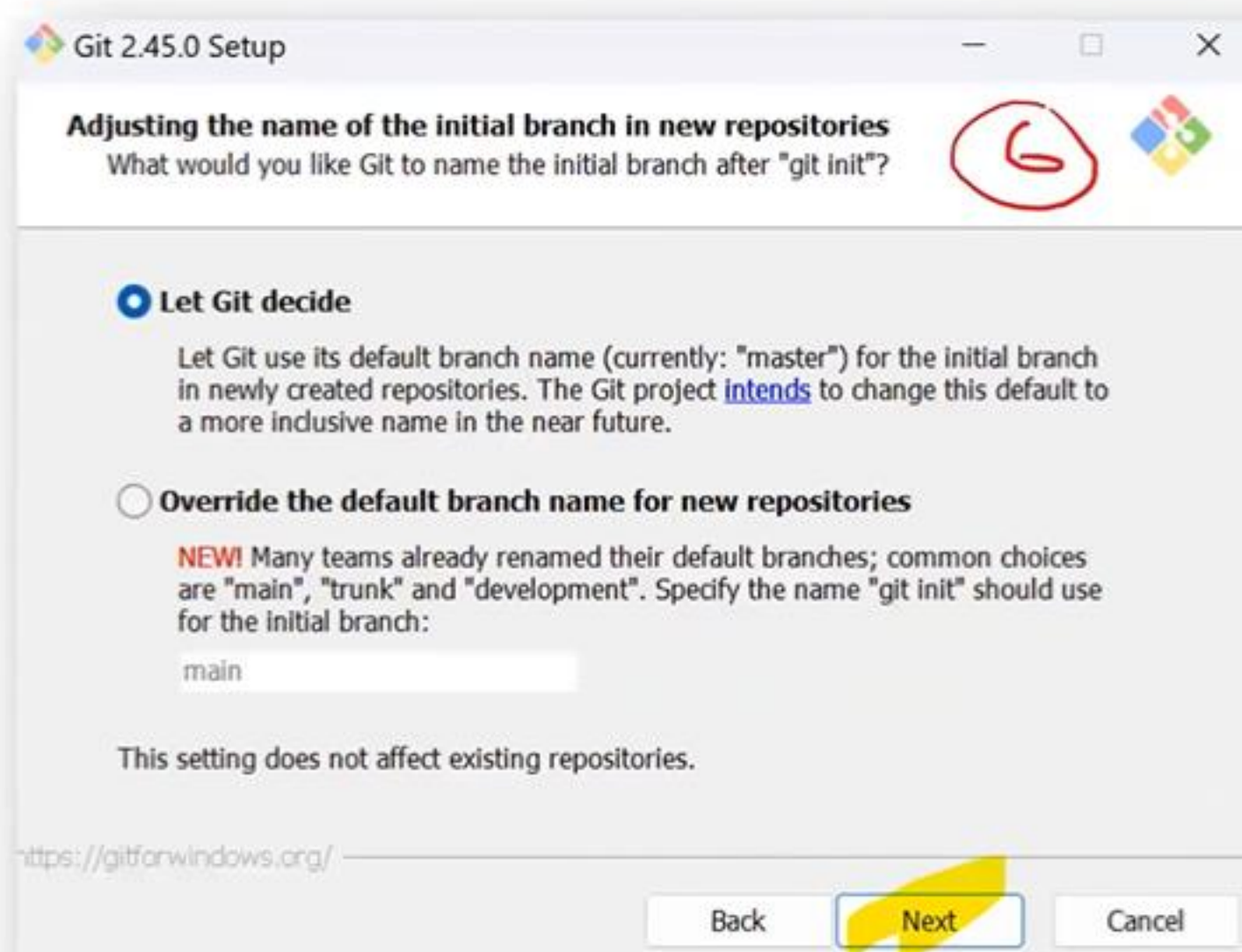
The [Vim editor](#), while powerful, [can be hard to use](#). Its user interface is unintuitive and its key bindings are awkward.

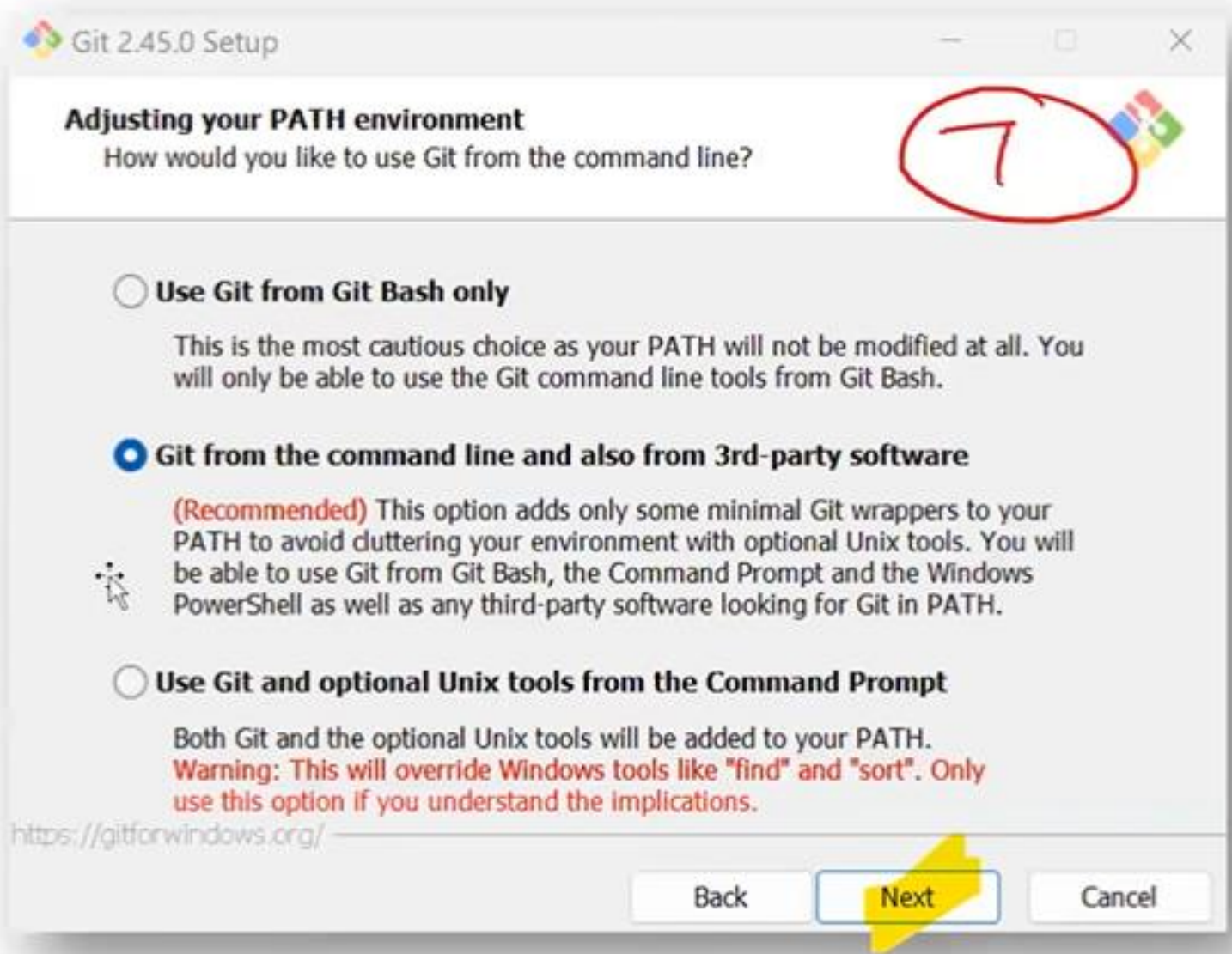
Note: Vim is the default editor of Git for Windows only for historical reasons, and it is highly recommended to switch to a modern GUI editor instead.

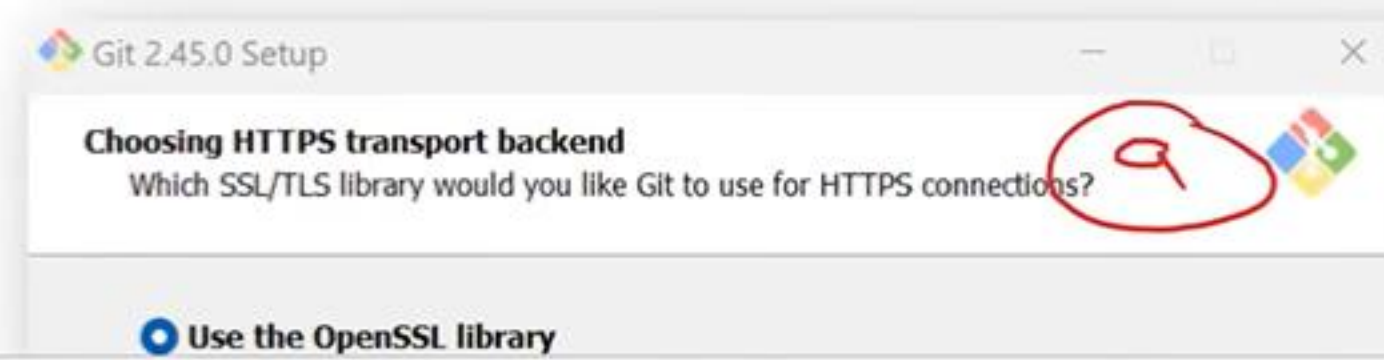
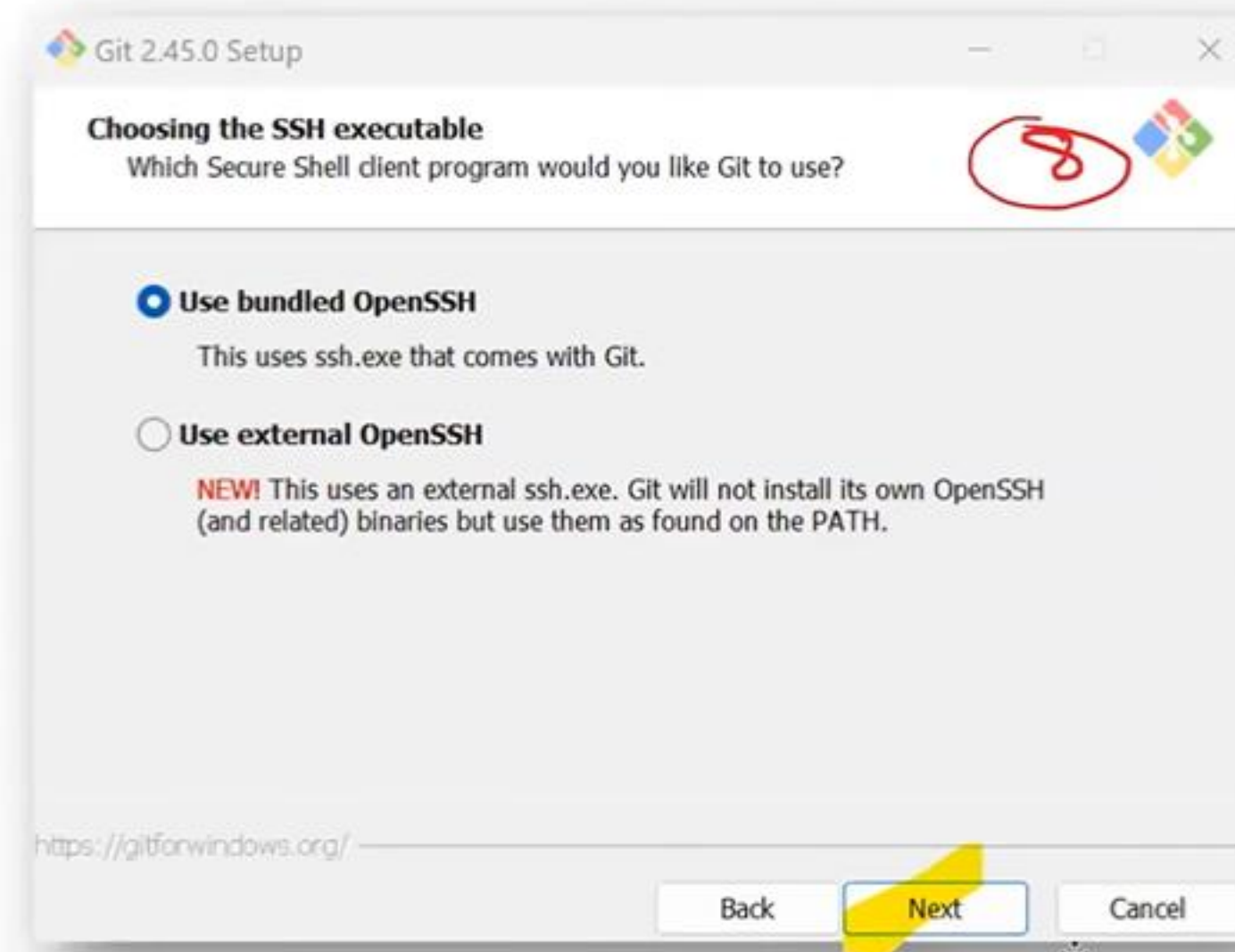
Note: This will leave the 'core.editor' option unset, which will make Git fall back to the 'EDITOR' environment variable. The default editor is Vim - but you may set it to some other editor of your choice.

<https://gitforwindows.org/>

Back Next Cancel







<https://gitforwindows.org/>

Back Next Cancel

Git 2.45.0 Setup

Choosing HTTPS transport backend

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

☒ **Use the OpenSSL library**

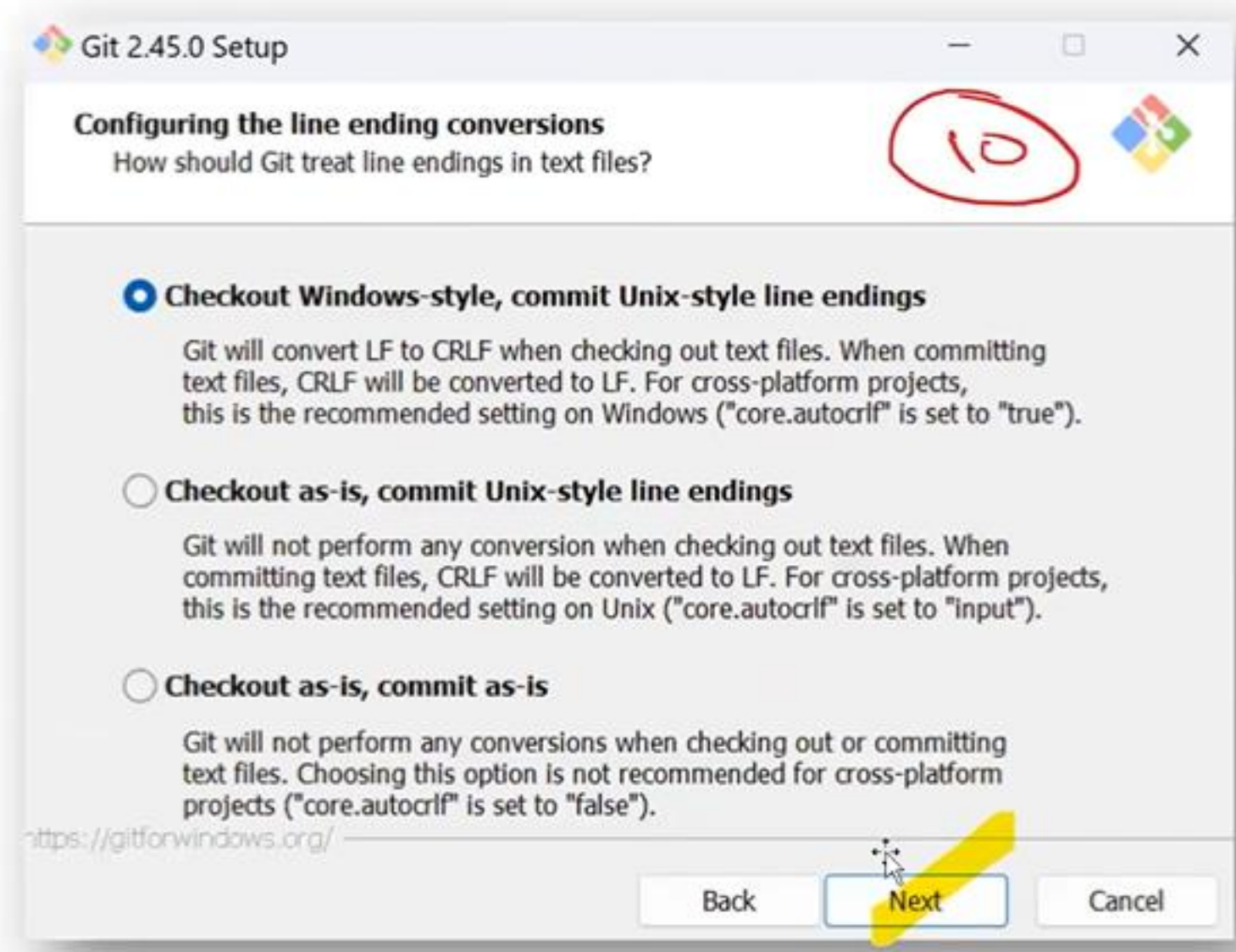
Server certificates will be validated using the ca-bundle.crt file.

☐ **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.

<https://gitforwindows.org/>

Back Next Cancel



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").

<https://gitforwindows.org/>

Back Next Cancel

Git 2.45.0 Setup

Configuring the terminal emulator to use with Git Bash

Which terminal emulator do you want to use with your Git Bash?

☒ 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 (such 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 well 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.

<https://gitforwindows.org/>

Back Next Cancel

Back Next Cancel

Git 2.45.0 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'.

<https://gitforwindows.org/>

Back Next Cancel

Git 2.45.0 Setup

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.

<https://gitforwindows.org/>

Back Next Cancel

Git 2.45.0 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.fsckache" 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.

<https://gitforwindows.org/>

Back Next Cancel

<https://gitforwindows.org/>

Back Next Cancel

Git 2.45.0 Setup

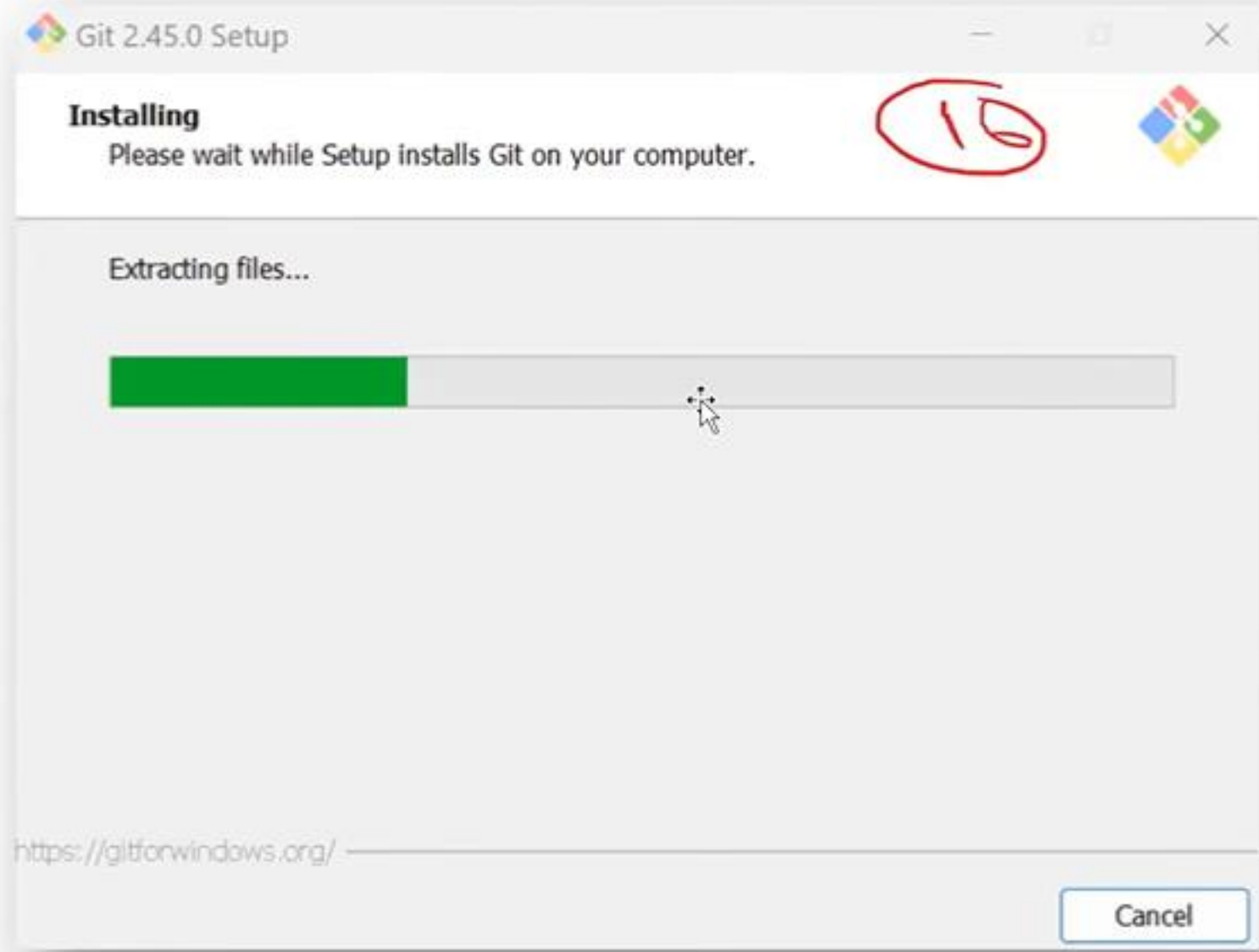
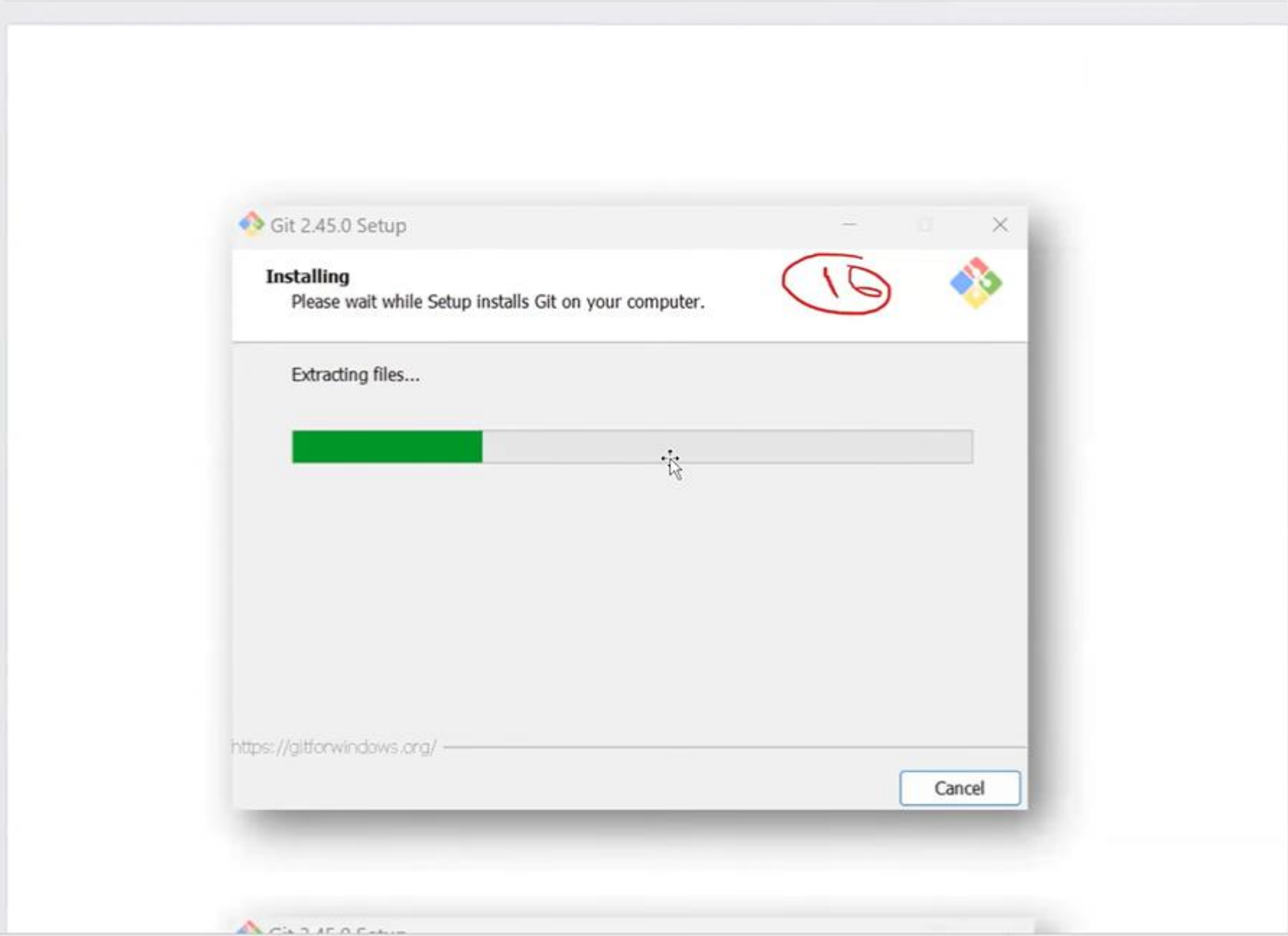
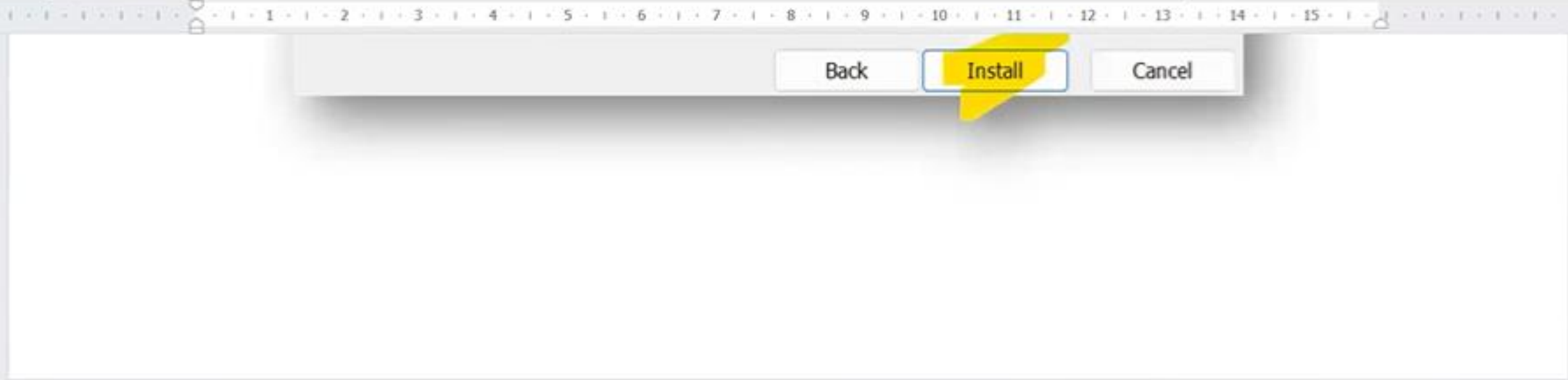
Configuring experimental options
These features are developed actively. Would you like to try them?

☐ **Enable experimental support for pseudo consoles.**
This allows running native console programs like Node or Python in a Git Bash window without using winpty, but is unfortunately not quite stable yet.

☐ **Enable experimental built-in file system monitor**
(NEW!) Automatically run a [built-in file system watcher](#), to speed up common operations such as ``git status``, ``git add``, ``git commit``, etc in worktrees containing many files.

<https://gitforwindows.org/>

Back Install Cancel



<https://gitforwindows.org/>

Cancel

Git 2.45.0 Setup

Completing the Git Setup Wizard

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

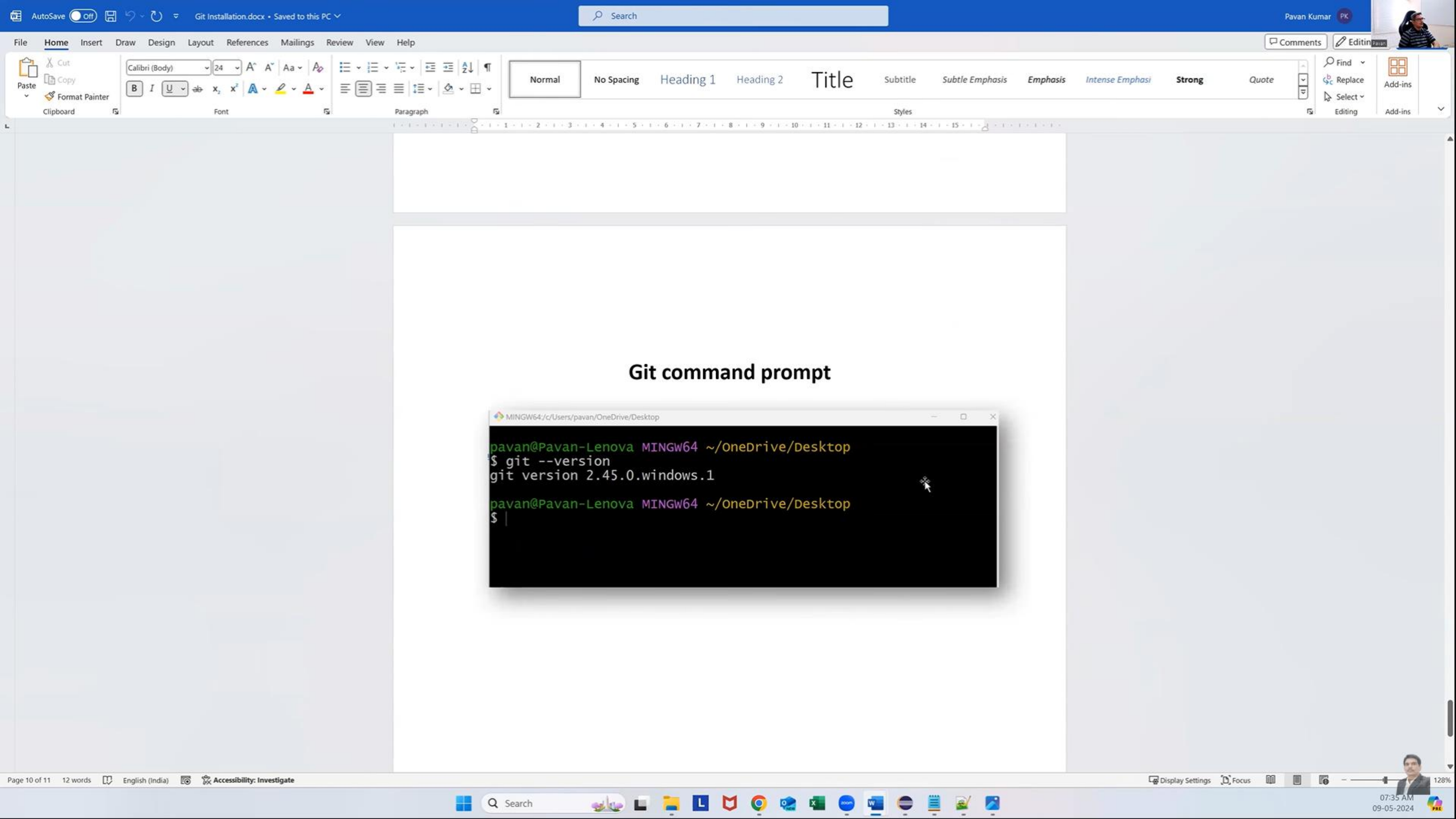
Click Finish to exit Setup.

☐ Launch Git Bash

☐ View Release Notes

17

Finish



Git Commands

Git workflow from creating a local repository to pushing changes to a remote repository.



1) Create a New Local Git Repository

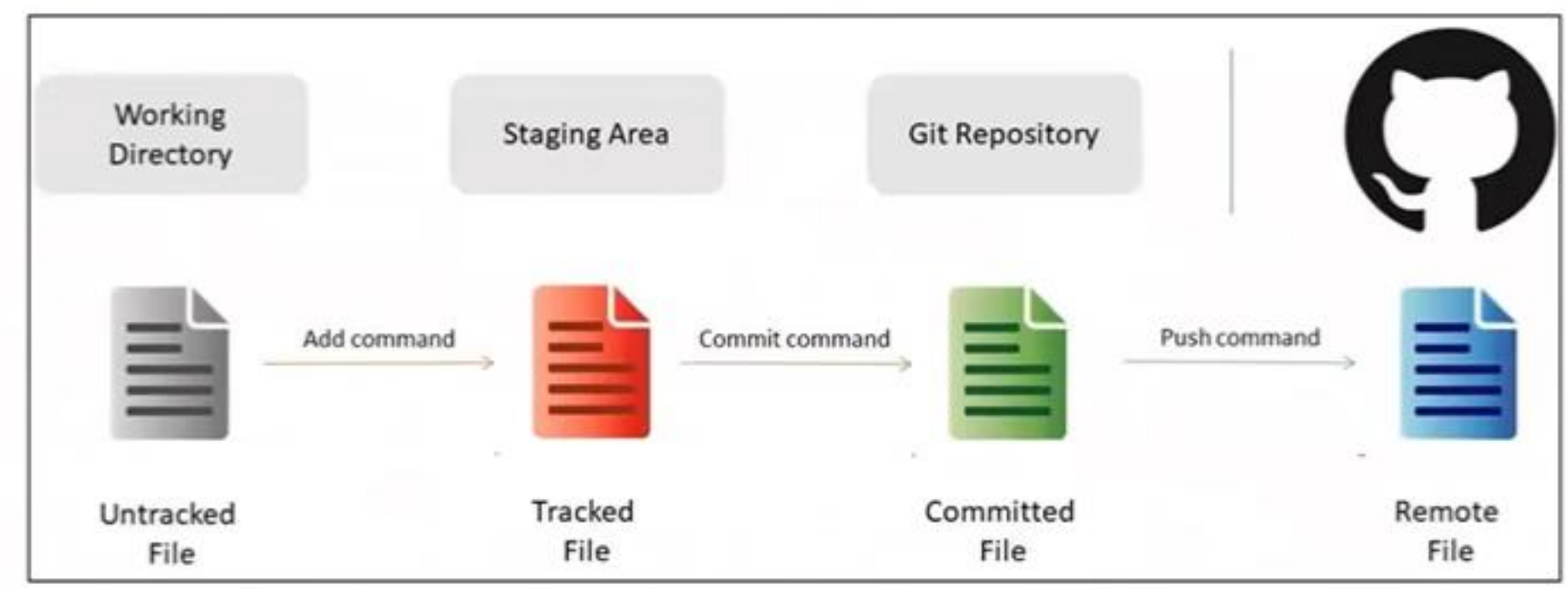
To create a new local Git repository, navigate to the desired directory and run **git init**:

```
git init
```

2) Provide User Info to Git Repo (One Time)

You need to configure your username and email for Git, which is usually

Git workflow from creating a local repository to pushing changes to a remote repository.



1) Create a New Local Git Repository

To create a new local Git repository, navigate to the desired directory and run **git init**:

```
git init
```

2) Provide User Info to Git Repo (One Time)

You need to configure your username and email for Git, which is usually done once. Replace "**your name**" and "**your email**" with your actual name and email:

```
git config --global user.name "your name"
git config --global user.email "your email"
```

3) Adding Files or Folders to Staging

To add files or folders to the staging area, use the **git add** command. You

3) Adding Files or Folders to Staging

To add files or folders to the staging area, use the **git add** command. You can add specific files, all files, or files matching a certain pattern:

```
git add -A           # Add all files and folders to staging
```

```
git add filename     # Add a specific file to staging
git add *.java        # Add all Java files to staging
git add foldername    # Add all files within a folder to staging
```

4) Commit the Code into Local (Git) Repository


```
git add filename           # Add a specific file to staging
git add *.java             # Add all Java files to staging
git add foldername        # Add all files within a folder to staging
```

4) Commit the Code into Local (Git) Repository

After adding files to the staging area, you commit them to the local repository along with a commit message using **git commit -m "commit message"**:

```
git commit -m "commit message"
```

5) Connect Local Repository with Remote Repository (One Time)

If you haven't already, you need to connect your local repository with a remote repository. This is typically done using the **git remote add** command followed by the remote repository URL:

remote repository. This is typically done using the **git remote add** command followed by the remote repository URL:

```
git remote add origin "https://github.com/pavanoltraining/opencart.git"
```

6) Push the Code into Remote Repository

Finally, to push your committed changes from the local repository to the remote repository, use the **git push** command followed by the name of the remote repository (often **origin**) and the branch you want to push (often **master** for the main branch):

```
git push origin master
```

These steps cover the basic Git workflow from creating a repository to pushing changes to a remote repository. Remember to replace **"your name"**, **"your email"**, and the repository URL with your actual information.