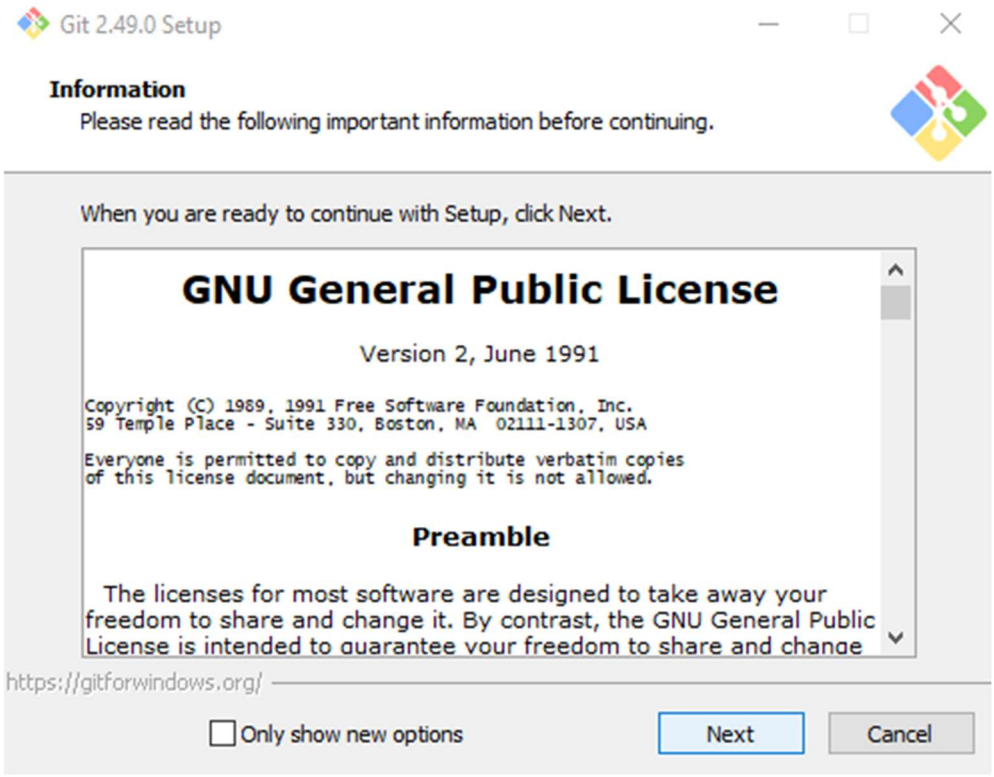
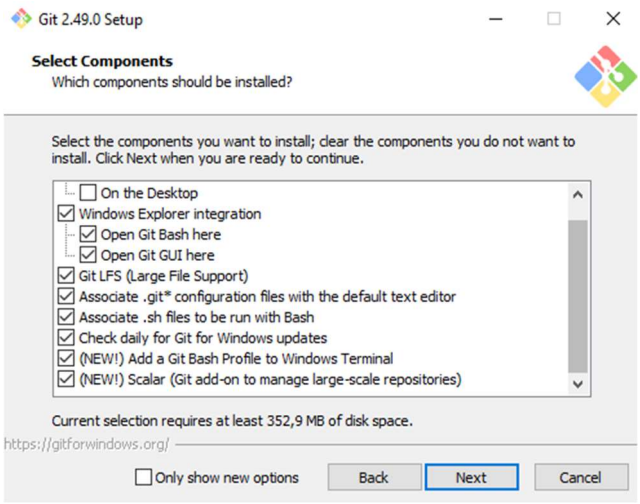
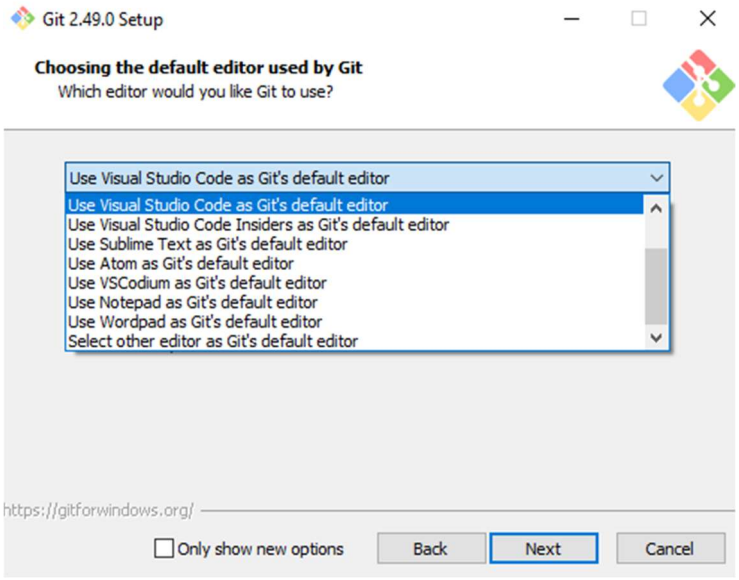
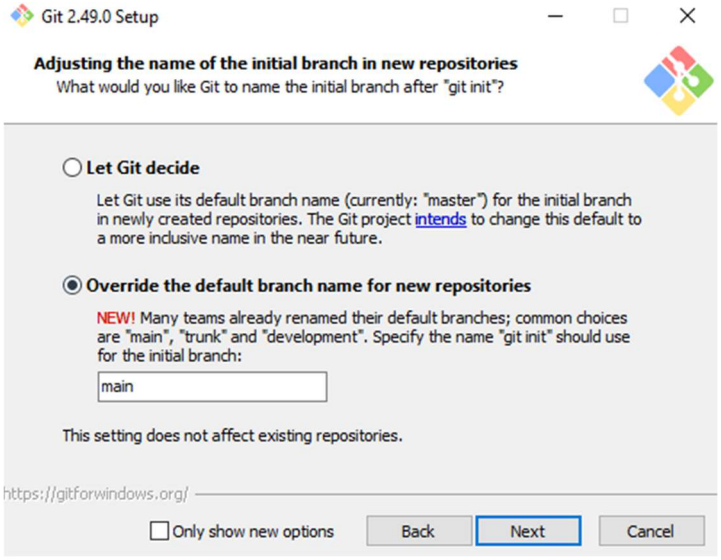
FASE DE INSTALACIÓN

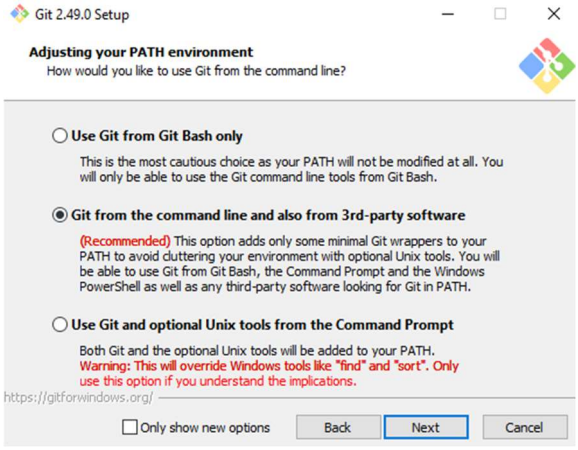
A continuación, se le adjunta las capturas, estas comprenden todo el proceso de instalación de ‘GIT’:

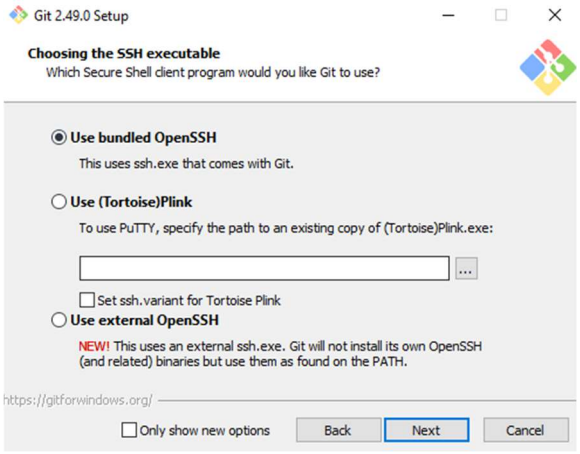


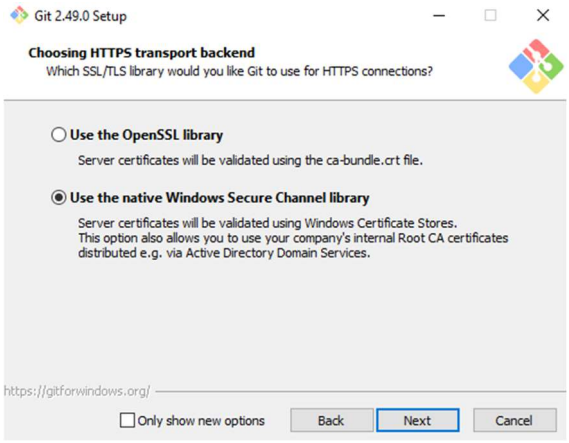


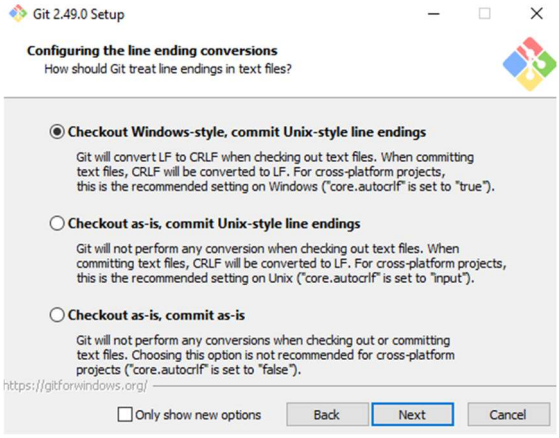

Git 2.49.0 Setup
Choosing the default editor used by Git
Which editor would you like Git to use?
Use Visual Studio Code as Git's default editor
Use Visual Studio Code as Git's default editor
Use Visual Studio Code Insiders as Git's default editor
Use Sublime Text as Git's default editor
Use Atom as Git's default editor
Use VSCodium as Git's default editor
Use Notepad as Git's default editor
Use Wordpad as Git's default editor
Select other editor as Git's default editor
https://gitforwindows.org/
Only show new options
Back
Next
Cancel

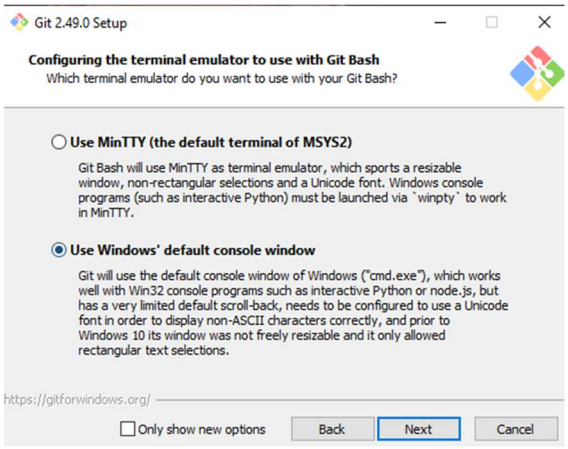

Git 2.49.0 Setup
Adjusting the name of the initial branch in new repositories
What would you like Git to name the initial branch after "git init"?
Let Git decide
Let Git use its default branch name (currently: "master") for the initial branch in newly created repositories. The Git project intends to 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 are "main", "trunk" and "development". Specify the name "git init" should use for the initial branch:
main
This setting does not affect existing repositories.
https://gitforwindows.org/
Only show new options
Back
Next
Cancel

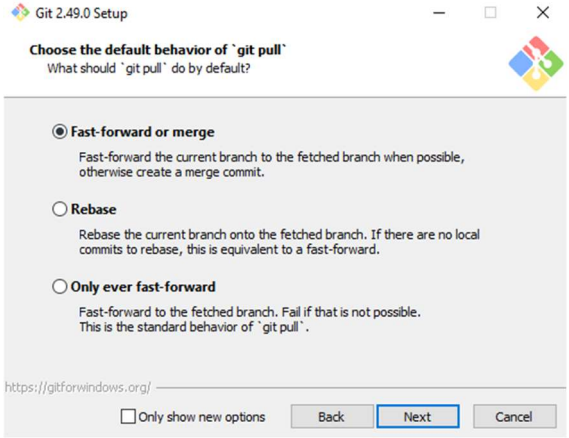

Git 2.49.0 Setup
Adjusting your PATH environment
How would you like to use Git from the command line?
Use Git from Git Bash only
This is the most cautious choice as your PATH will not be modified at all. You will only be able to use the Git command line tools from Git Bash.
Git from the command line 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 Windows 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.
https://gitforwindows.org/
Only show new options
Back
Next
Cancel

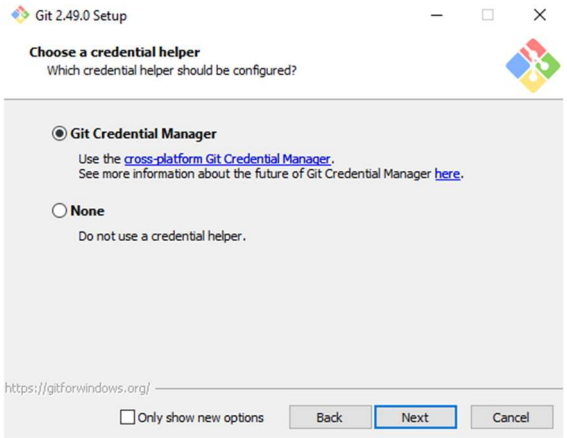

Git 2.49.0 Setup
Choosing the SSH executable
Which Secure Shell client program would you like Git to use?
Use bundled OpenSSH
This uses ssh.exe that comes with Git.
Use (Tortoise)Plink
To use PuTTY, specify the path to an existing copy of (Tortoise)Plink.exe:
...
Set ssh.variant for Tortoise Plink
Use 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.
https://gitforwindows.org/
Only show new options
Back
Next
Cancel

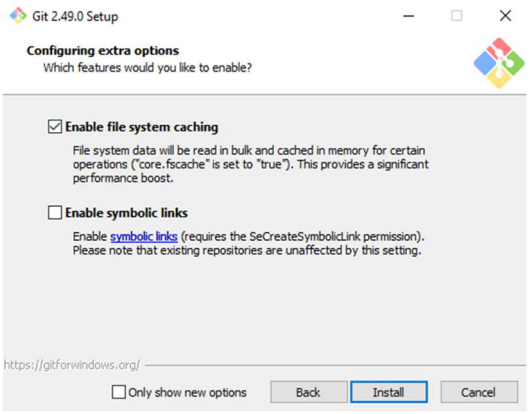

Git 2.49.0 Setup
Choosing HTTPS transport backend
Which SSL/TLS library would you like Git to use for HTTPS connections?
Use the OpenSSL library
Server certificates will be validated using the ca-bundle.crt file.
Use the native Windows Secure Channel library
Server certificates will be validated using 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/
Only show new options
Back
Next
Cancel


Git 2.49.0 Setup
Configuring the line ending conversions
How should Git treat line endings in text files?
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").
https://gitforwindows.org/
Only show new options
Back
Next
Cancel


Git 2.49.0 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/
Only show new options
Back
Next
Cancel

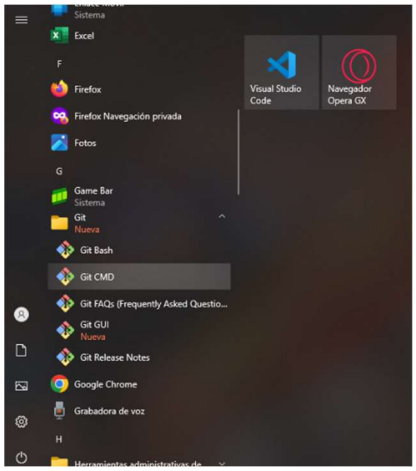

Git 2.49.0 Setup
Choose the default behavior of `git pull`
What should `git pull` do by default?
Fast-forward or merge
Fast-forward the current branch to the fetched branch when possible, otherwise create a merge commit.
Rebase
Rebase the current branch onto the fetched branch. If there are no local 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/
Only show new options
Back
Next
Cancel


Git 2.49.0 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/
Only show new options
Back
Next
Cancel



Aquí solo deberás darle a ‘INSTALL’ y ya se procederá a su instalación

PRÁCTICA

1. 

2.

```
Portatil Kaimane@DESKTOP-D3MTPU4 MINGW64 ~
$ cd desketop
bash: cd: desketop: No such file or directory

Portatil Kaimane@DESKTOP-D3MTPU4 MINGW64 ~
$ cd desktop

Portatil Kaimane@DESKTOP-D3MTPU4 MINGW64 ~/desktop
$ cd git

Portatil Kaimane@DESKTOP-D3MTPU4 MINGW64 ~/desktop/git
$ cd ..

Portatil Kaimane@DESKTOP-D3MTPU4 MINGW64 ~/desktop
$ cd GIT

Portatil Kaimane@DESKTOP-D3MTPU4 MINGW64 ~/desktop/GIT
$ git init
Initialized empty Git repository in C:/Users/Portatil Jorge/Desktop/GIT/.git/

Portatil Kaimane@DESKTOP-D3MTPU4 MINGW64 ~/desktop/GIT (main)
$ git status
On branch main

No commits yet

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

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

Portatil Kaimane@DESKTOP-D3MTPU4 MINGW64 ~/desktop/GIT (main)
$ git add index_prueba.txt

Portatil Kaimane@DESKTOP-D3MTPU4 MINGW64 ~/desktop/GIT (main)
$ git status
On branch main

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   index_prueba.txt


Portatil Kaimane@DESKTOP-D3MTPU4 MINGW64 ~/desktop/GIT (main)
$
```