*For Internal Usage of Binus Only. Made by Web Service Team.*

**Tutorial Git 101: Simple Comprehensive Guide**

**Logo, icon

Description automatically generated**

Contents

[What is Git? 1](#_Toc65614459)

[Why use Git? 2](#_Toc65614460)

[How Git Works 3](#_Toc65614461)

[Installing Git 3](#_Toc65614462)

[Common Problems 5](#_Toc65614463)

[1. Git Initialize (Git Init) 7](#_Toc65614464)

[2. Git Commit 9](#_Toc65614465)

[3. Git Push 10](#_Toc65614466)

[4. Git Clone 12](#_Toc65614467)

[5. Git Pull 12](#_Toc65614468)

[6. Git Fetch 13](#_Toc65614469)

[7. Git Branch 13](#_Toc65614470)

[8. Git Merge 16](#_Toc65614471)

[9. Pull Request 19](#_Toc65614472)

## What is Git?

Git is a software used for tracking changes in files.

## Why use Git?

A group of people having a discussion

Description automatically generated with medium confidence

Imagine you worked in a small team, your friend, Andy in Front-End, pushed an update for File A, then your other friend, Billy, decided to update File A too. Your friend, Charlie, does not know what is happening. All of you fight against each other to decide which file to keep.

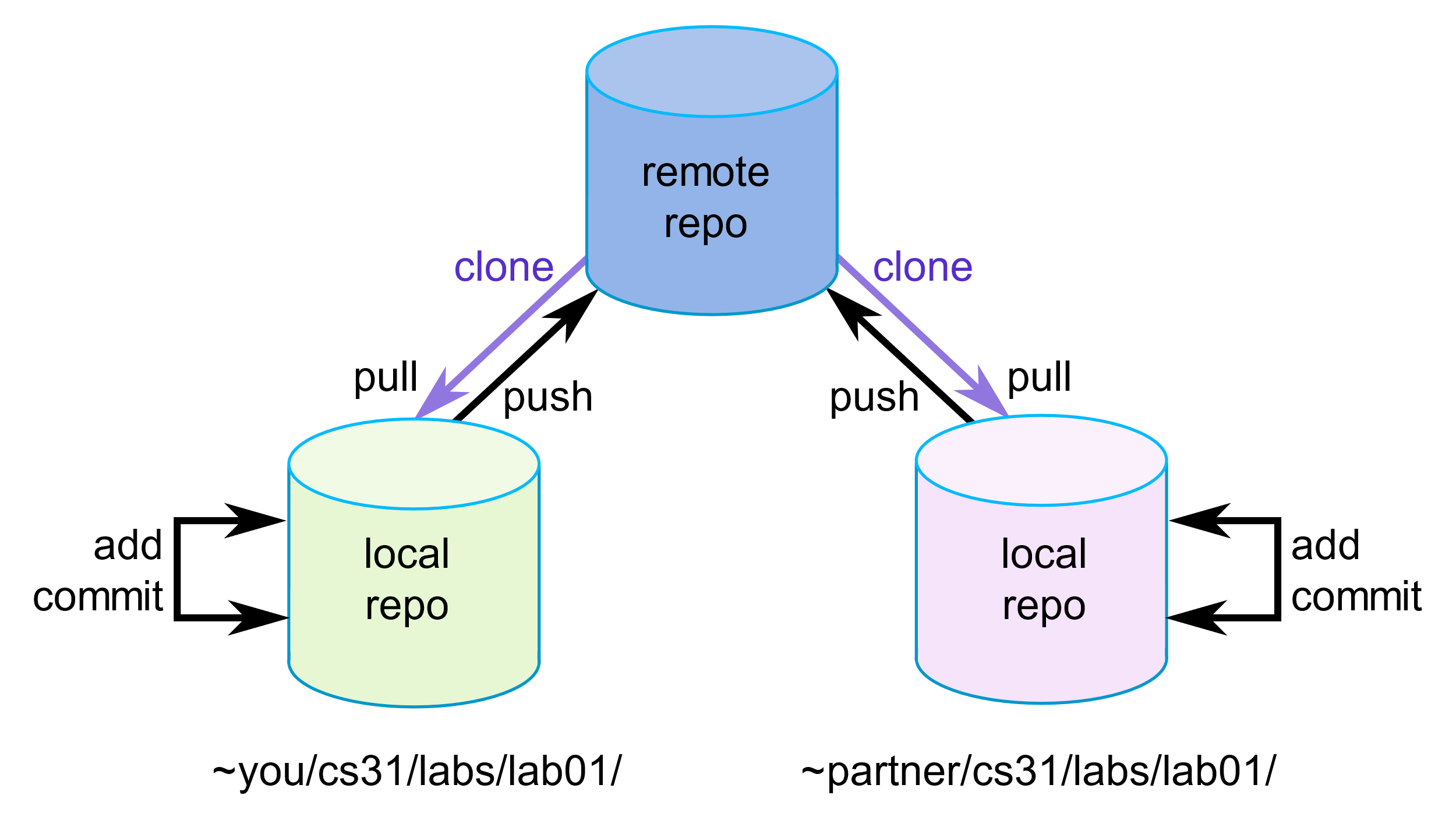
Diagram

Description automatically generated

This is only just for one file – what if your project is in middle of late development, with hundreds and thousands of different files, and your best friend has deleted some “unused” files, how do you know which one to keep? This is where **Version Controlling** comes in.

## How Git Works

Git works by saving and tracking all changes that happened on each file in a project – so every time you finished in a file, you can save it and tell Git to save the changes so your other teammates can “know” and “get” all updates from you.



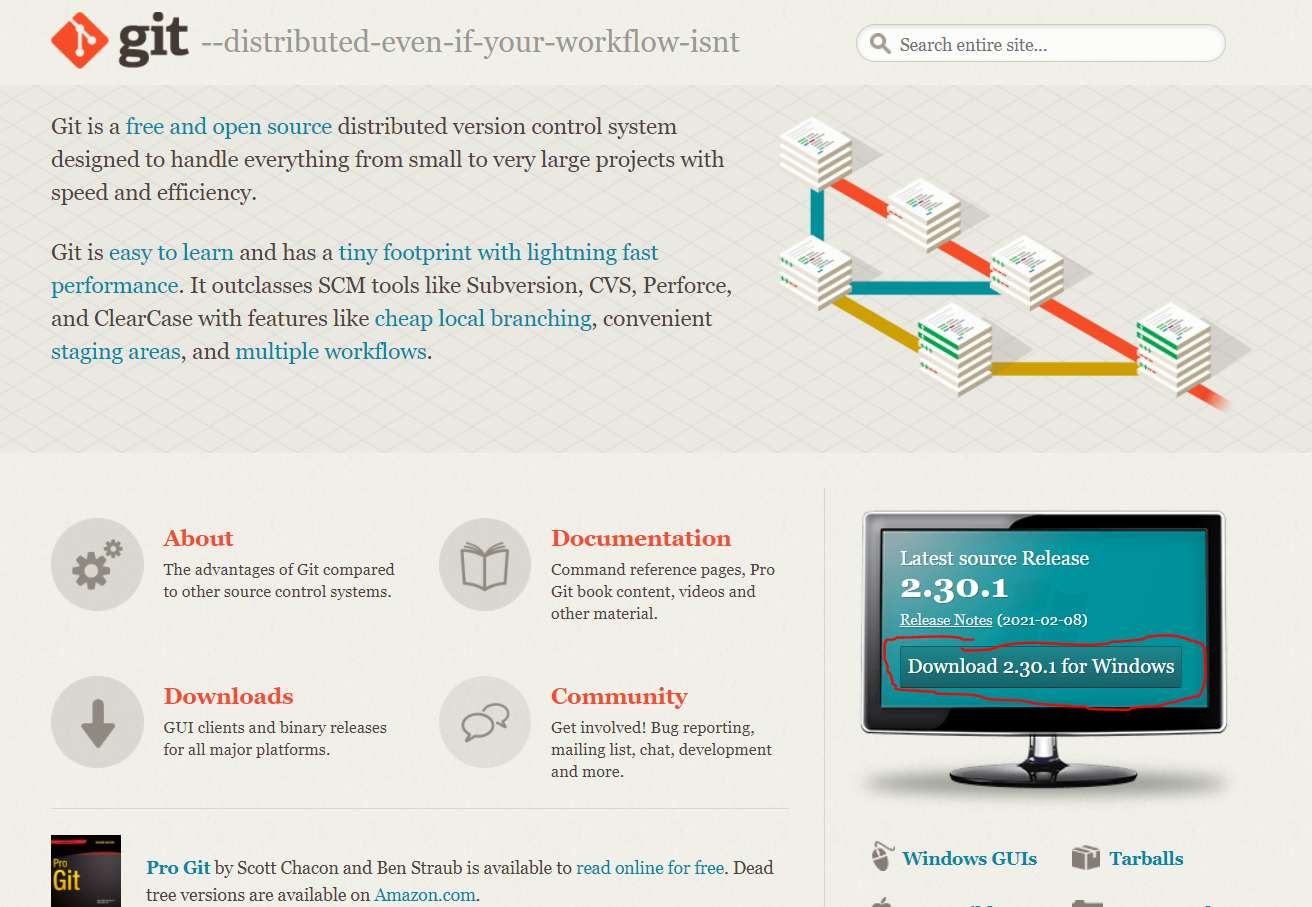
To use git effectively, you need a shared place called “**Remote Repository**” (Think Repository as a bucket, where you hold everything there), which is a server that holds all your files on the internet (Do note that this optional, but is necessary for team projects). All the project files in your computer are called “**Local Repository**”.

Every change that you do will be saved in your **Local Repository** (as in, your computer), and you can tell Git to “push” all updates from your Local Repository to the Remote (Shared) Repository.

Because of that, your other teammates will then be able to “pull” any updates from the Remote (Shared) Repository to their own Repository. And vice versa.

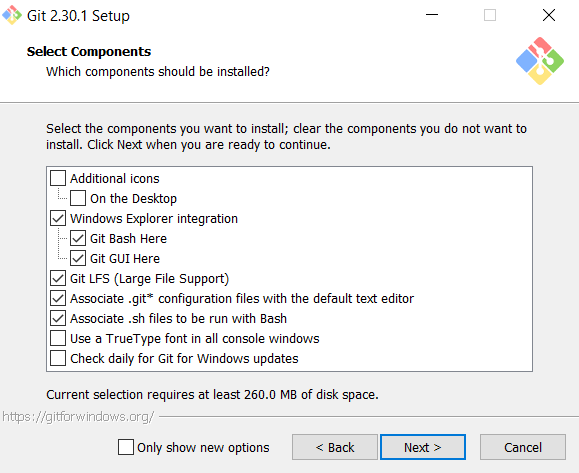
## Installing Git

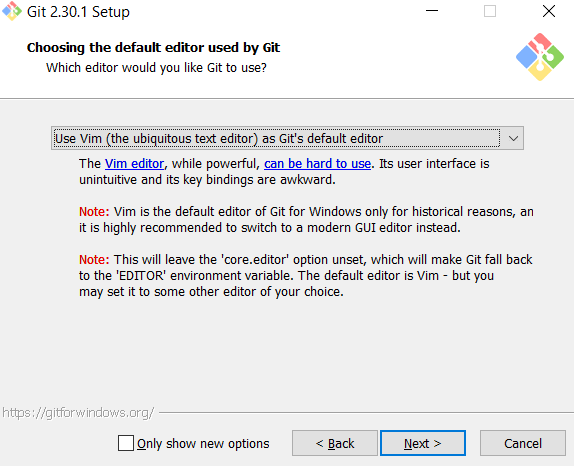
To Install Git, first, go to <https://git-scm.com/> using your favorite Browser.



After that, press the Download <Latest Version> for Windows (if you use MacOS, press the “Mac Build” just below the button).

Your browser will then download Git, run the installer. Check the following boxes:





Use your favorite Text Editor as Git default editor, it is not required to use Vim, if you feel better using VS Code, use that instead.

If you are using Git 2.30.1, in order, after selecting Next from above:

Let Git Decide for the default branch name. And then, use the Recommended setting (Git from the command line and also from 3rd-party software), Use the OpenSSL Library, Checkout Windows-Style, commit Unix-style line endings, Use MinTTY, Set the default behavior of git pull with Default (fast-forward or merge), For Credential helper, use Git Credential Manager Core, and Enable File System Caching. Do not enable Experimental support for pseudo consoles.

Then Install. To ensure that this thing works, go to Command Prompt, type “git --version”, if this appears:

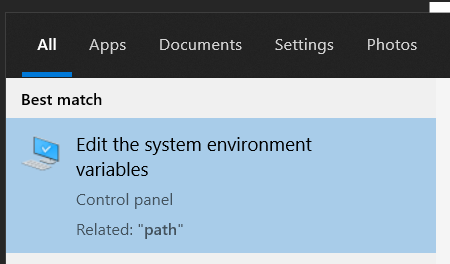


Then it is already installed. However, if you get an error, there is several common problems:

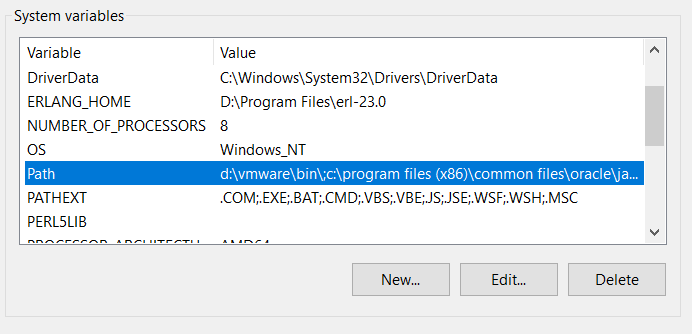
## Common Problems

1. Check Path

Check the Environment path of Git by typing “Path” on your computer.



Then, go to your “System Variables”, press “Edit…” on the “Path” Variable.



Make sure C:\<where you installed Git>\Git\cmd exists:



If not, just add the path where your Git is located, by clicking “New” on the right ribbon.

(Answer from: [windows - 'git' is not recognized as an internal or external command - Stack Overflow](https://stackoverflow.com/questions/4492979/git-is-not-recognized-as-an-internal-or-external-command))

2. Incompatible Git Build

Make sure you installed the right Git build (Mac build for Mac, Windows build for Windows), and the latest version available. (Recommended)

In Git, there is **always two options to do anything in it**, either using the Command Line Prompt, or using Git GUI (such as using Studio 2019, or the Git website itself). Using the GUI is usually always easier, but the Command Line Prompt is faster if you are expert in using Git.

There are several terminologies that is frequently used in Git.

## 1. Git Initialize (Git Init)

Git initialize is basically telling Git you are creating a **new Local Repository.**

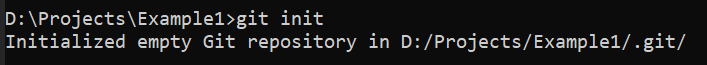
Using Git init, you basically has done the following:

Diagram

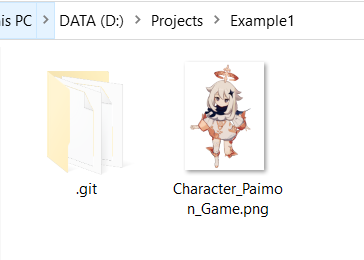
Description automatically generated with medium confidence

This means that you created a Local Repository but **is not connected to any Remote Repository.**

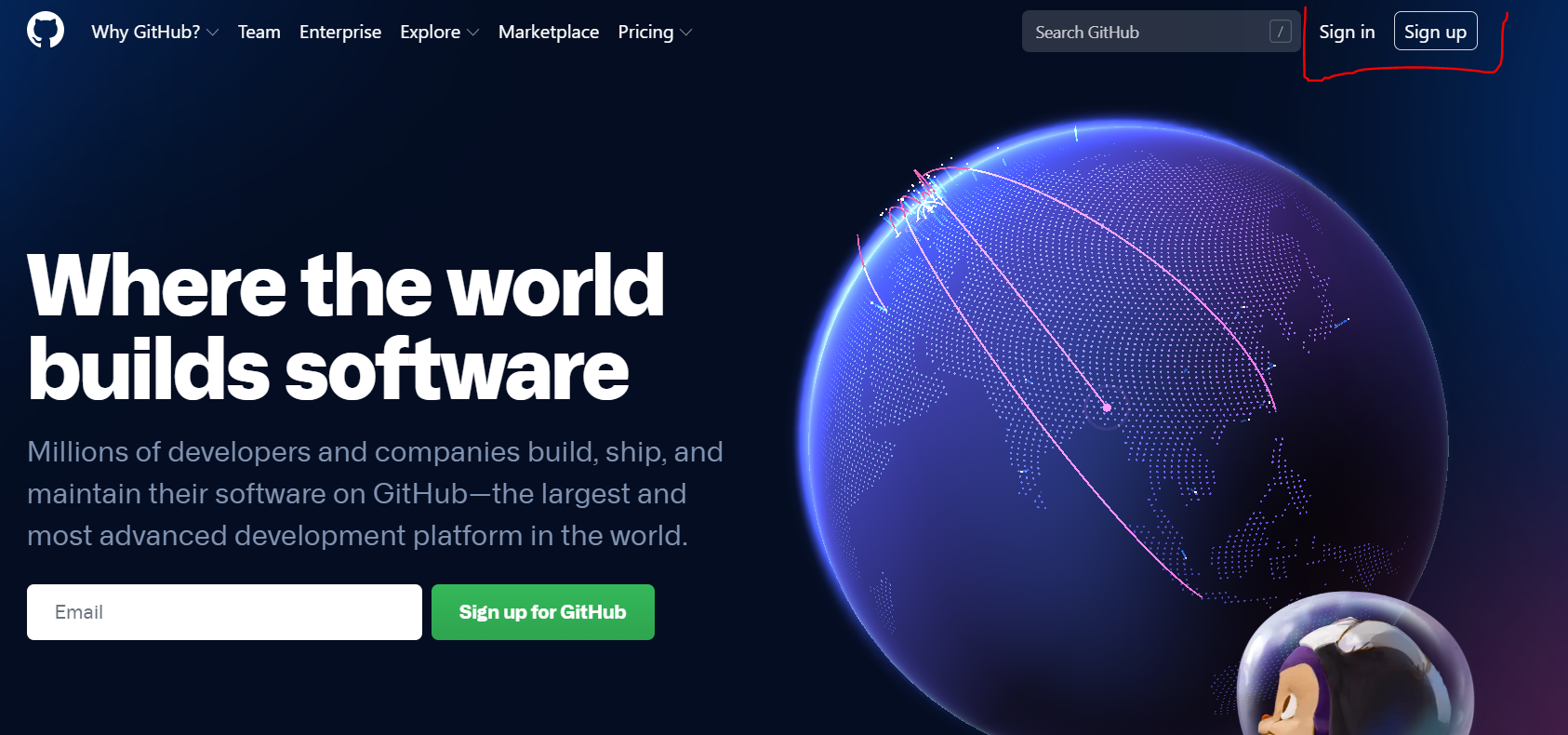
To practice, first, open Command Prompt, and then go to your desired directory that will be used as Repository (or in Git term, “tracked”, **Make sure that directory hasn’t been used for git previously**.), you can create a new folder for this example. Then type “git init”



And then, move any file you want **inside** the directory, for example, I moved this picture inside the directory.



After Initializing, we usually create the Remote Repository that will be shared among teammates. For practice, we can use one of the largest Git providers – GitHub.

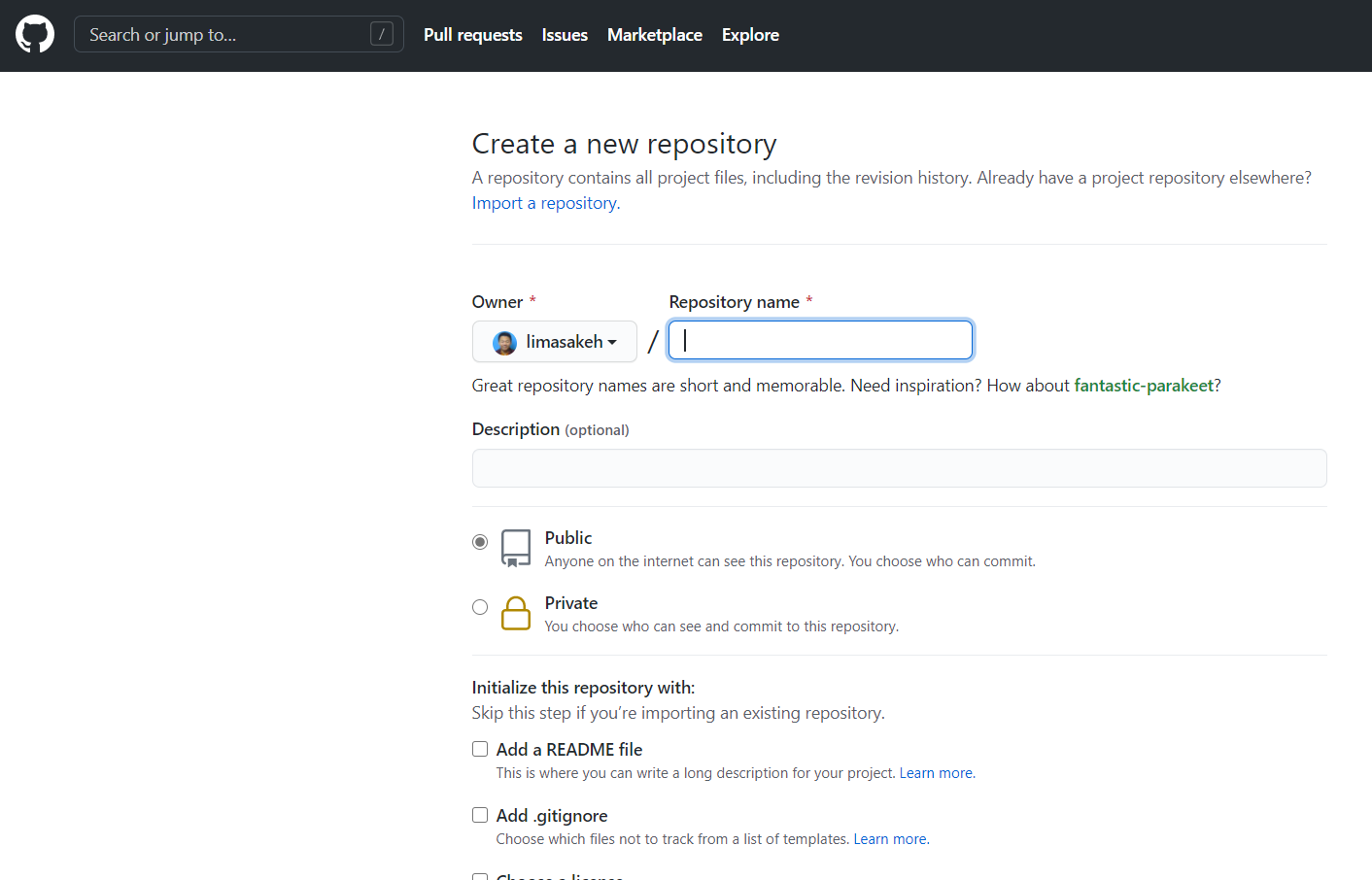


First, Login to your Github account. If you don’t have a github account – Press Sign Up instead.

After that, inside your account, on the top-right corner, press the Plus button.

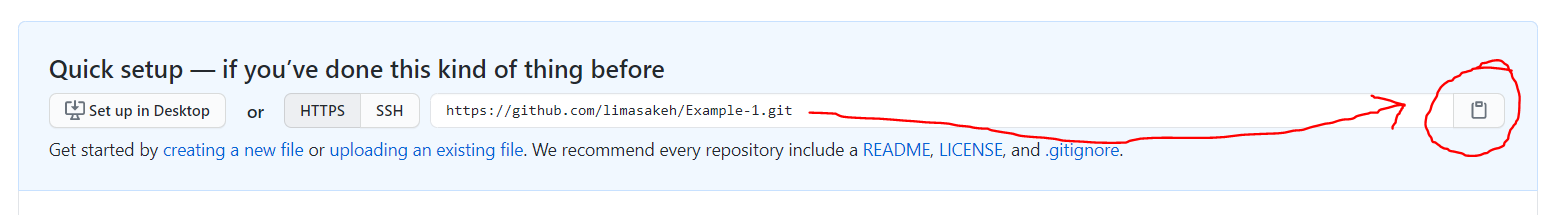


And press [New Repository].



On this page, type your Repository Name, you can use anything but for the sake of example, the Repository Name is [Example-1], and **make sure to set the Repository as Private.** (Since this is only for practice purpose).

After that, Copy the link that ends with .git provided from the web.



Go back to your command prompt, then type “git remote add origin <copied link>”



Doing the above command will link your Local Repository toward that Remote Repository, shown by the following diagram:

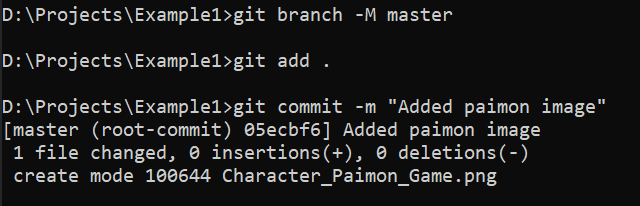
Graphical user interface

Description automatically generated with medium confidence

## 2. Git Commit

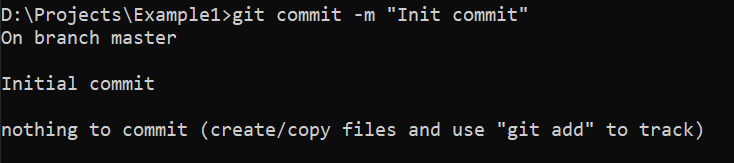
Git commit means you are sure about your update and want to **update your Local Repository** (update to your computer). Just think of it as “I like to **commit** that this update **is final.**”

Type “git branch -M master”, and then, type “git add .” (this adds all updated files), after that, type “git commit -m <your comment>” to finalize your update.



If the above appears, you had committed your update, but your Local Repository and Remote Repository is not synced yet because you had not pushed the update. Now you can now push the update, see [Git Push](#_2._Git_Push) for the next step.

However, if the below appears instead:

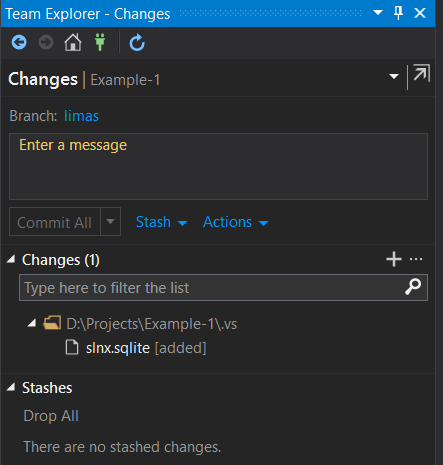


You might put the file in the wrong directory, try checking the file in the directory, and **make sure the Repository is in the correct Directory.**

Second option of committing is by using Visual Studio 2019, open Visual Studio 2019, and look at the bottom-right corner. Click the pen icon.



After that, write the Commit Message which details what you updated; after that, click “Commit All.”



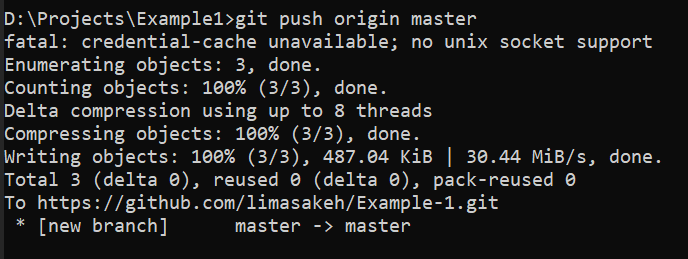
This commits all changes and updates your Local Repository.

## 3. Git Push

Git push means “please **push** everything that I **committed**” to **the Remote Repository.** Usually, you need to **commit** first.

Following from the git init tutorial, you **had committed your update in your Local Repository,** however, the **Remote Repository does not know this.** So, you need to “push the update”.

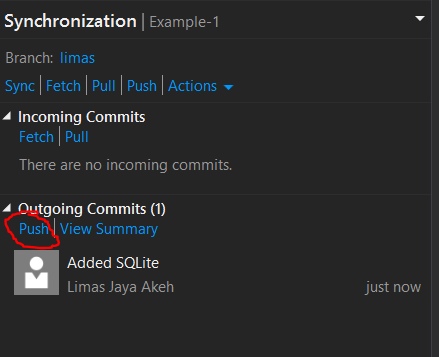
To do git push is extremely simple, just type “git push <branch destination>”



Or you can use Visual Studio 2019 to push the update, click the Arrow Icon.



On the right side, click the small “Push” button



After you do that, you just did the following:

Graphical user interface, application

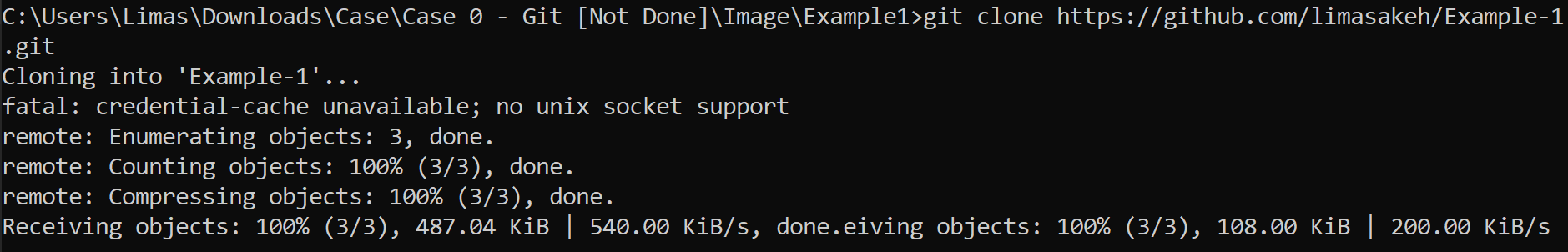
Description automatically generated

## 4. Git Clone

Will most likely be used. Git Clone is a command to **create a Local Repository that is linked to the Remote Repository.** This means that if your teammate has created a Remote Repository, you can use this command to directly initialize a local repository based from the Remote Repository.

For practicing Git Clone, you can try cloning this repository: <https://github.com/limasakeh/Example-1.git>

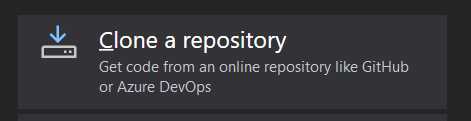
First, create a new folder named “Example1”, and then, open Command Prompt and go to the newly created folder. Type “git clone <the git link>”



Or you can use Visual Studio 2019.

- Open Visual Studio 2019

- Clone the project using the “Clone a Repository”



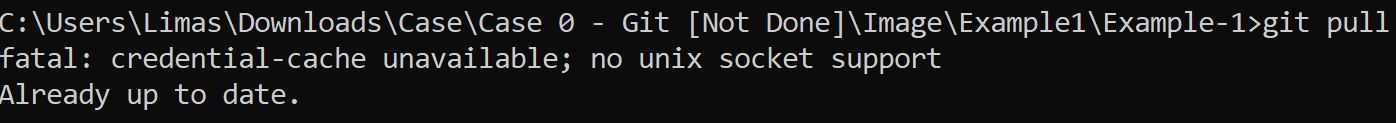
And you’re done, you can code right away.

## 5. Git Pull

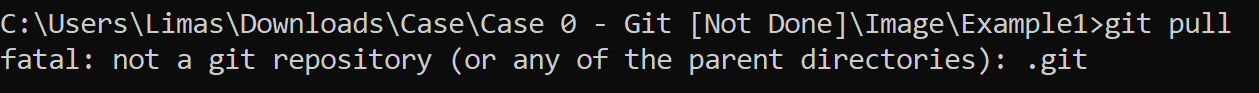
Git pull means “please pull any new updates” from the Remote Repository to your Local Repository. This means that any update **from the Remote Repository** will be given **to Local Repository**.

To practice Git pull, you need to have a repository that is managed by others, so you will practice it alongside others.

These messages may appear:



If you already have the latest updates from everyone – then it will say “Already up to date”.



If you are **not** in a git repository folder, make sure the folder is correct.

## 6. Git Fetch

Git fetch is like Git pull, except that it **will not affect your Local Repository.** Useful if you want to check if there is any update. Think of it as “Hey, I like to fetch updates, any news?”

Doing git fetch is like git pull, just type “git fetch” to know the latest updates.

## 7. Git Branch

Graphical user interface

Description automatically generated with low confidence

In Git, there is one important feature, which is called “Branch”.

Think of branch as “alternative timeline”. Anything you do in the branch, will only happen on that branch. For example, in the diagram above, we have two branches in our Local Repository, called “Master” and “Limas”, these branches will also be reflected on the Remote Repository. If you update the “Limas” branch, the “Master” branch timeline will stay the same without any changes, so branches is useful **if you want to create, update, or any sort of changes toward the project.**

In a huge project, Branches can be useful, look at the following diagram:

Diagram

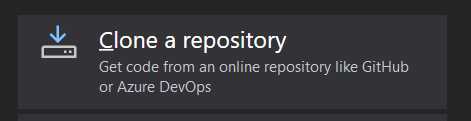
Description automatically generated with medium confidence

There are many branches, for example, the branches are named “Limas”, “Timotius” in the Front End, and “Jevon”, “Hansel” in the Back End. Each programmer develops and finishes their own task. The result – the “Master” branch, will be where everyone merges and connect all progress into a single product that will be sent to end users. The process of merging and connecting all different things can be done using Git Merge.

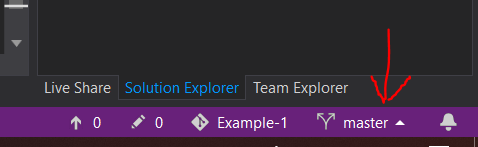
To create a new Branch easily, follow these steps:

- Open Visual Studio 2019

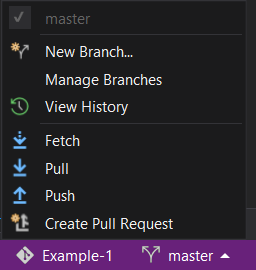
- Clone the project using the “Clone a Repository” (this will help you git clone the project easily)



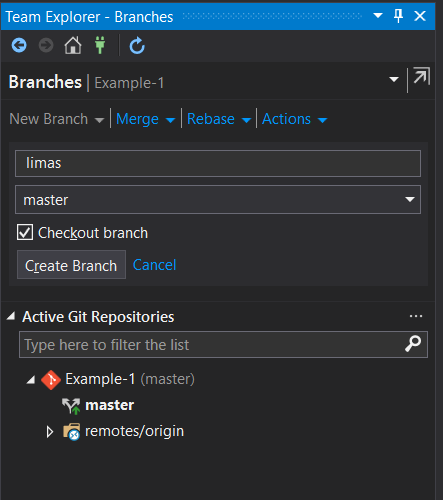
- On the newly cloned project, on the right-bottom ribbon, click the “Master” text.



- Click “New Branch…”



- On the right side tab, on the Text field “Enter the Branch Name <required>”, type the name of branch you want to make, then click “Create Branch”



- At this point, you have created a branch **in your Local Repository.** We need **to tell this update to the Remote Repository**. Which command do we use?

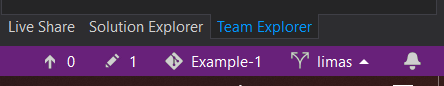
A. Git Push

B. Git Pull

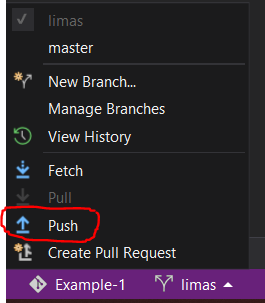
C. Git Update

If you answered A, you are correct. But if you answered B or C, go to page 1.

- Now, we can push the update to the Remote Repository. Click the branch name on the bottom ribbon.



- Click “Push” to finally tell the update to Git.



Congratulations! You have created a new branch.

## 8. Git Merge

Git merge is useful to **merge all your updates in your current branch** into **another branch.** This is useful because in a multiple programmer environment, there is usually a single branch that is used as the final branch where everything is put together into place.

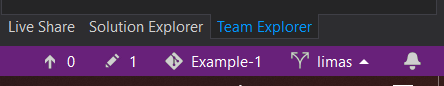
There are two merges that usually happens:

- Merge Upwards: Merging your updates to master branch (usually uses [Pull Requests](#_9._Pull_Request))

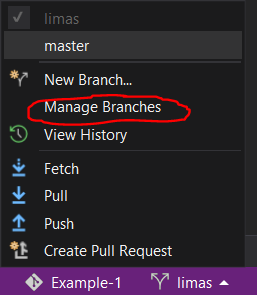
- Merge Downwards: Merging the master branch updates to your own branch (using Git Merge)

To do Git Merge, **make sure you have pushed all of your updates to your own branch in the Remote Repository** (that is, committing and pushing it to Remote Repository)and do the following step:

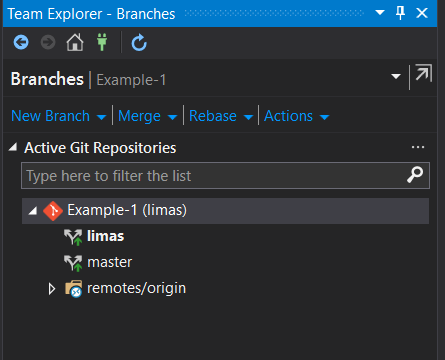
- First, click the branch name on the bottom ribbon.



- Click “Manage Branches”

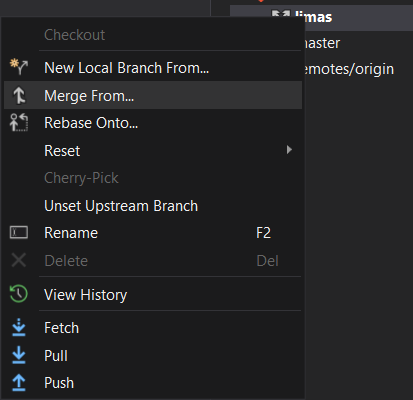


- The following will appear on the right side:

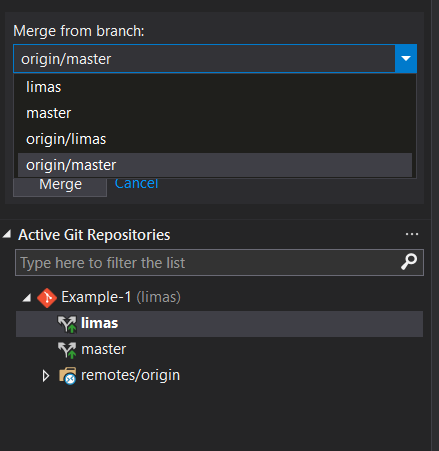


- As the best practice, it is **advisable to merge upwards using Pull Request**, but we are going to do merge downwards for this example.

- Now, right-click on your unique branch (for this example, the branch name is “limas”) and click “Merge From…”



- Click the branch you wish to merge from, do note that branches start with “origin” is usually the **Remote Branches.** Try clicking “origin/master”



- And then click “Merge” button. By doing that, we just did the following:

A picture containing text

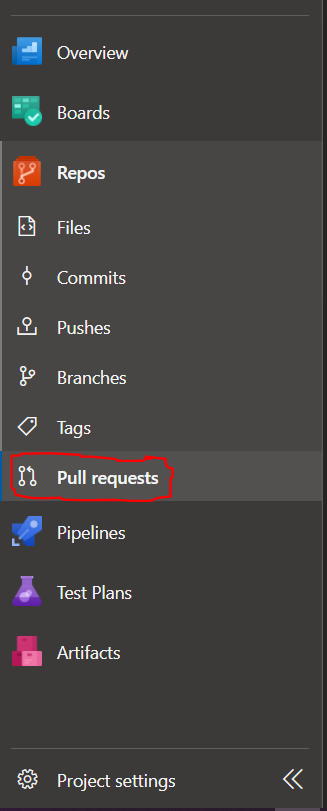
Description automatically generated

This will make sure that the “Limas” branch **on the Local Repository** is up to date with the “Master” branch on the **Remote Repository**.

## 9. Pull Request

Pull Request Is like Git Merge, except that it is more “formal” since other people needs to review and accept your request to merge the update.

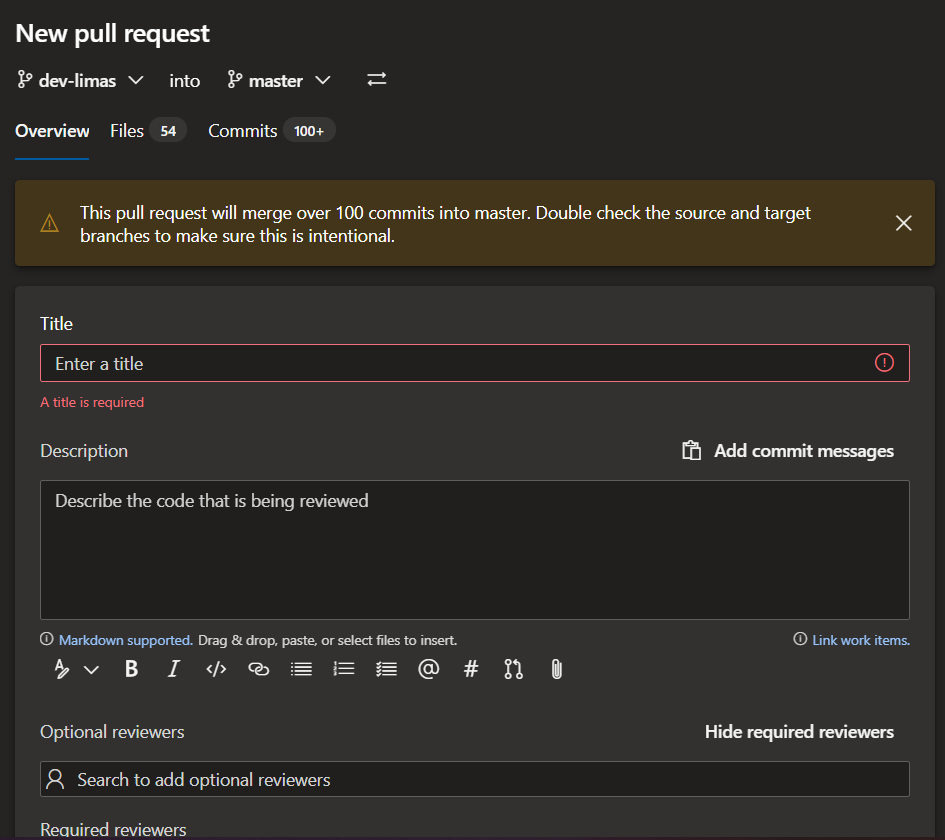
We will use Azure DevOps for Pull Requests, first, go to “Repos” > “Pull Requests”.

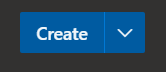


- Click the “New Pull Request” button on the top-right corner.



- Fill in the required fields such as Title, which branch to merge, descriptions, etc.

- After that, click the “Create” button



- You have created a Pull Request, now, your senior or supervisor will check updates that you had committed toward the project.