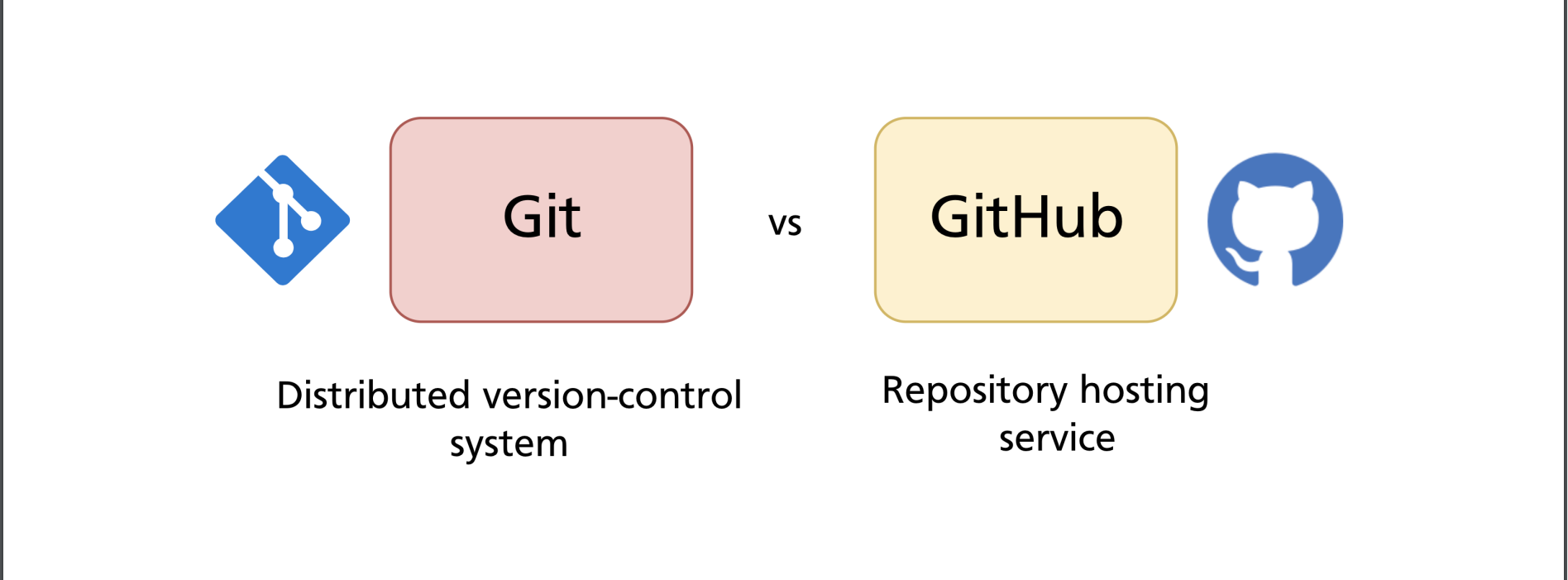
1. **GIT & GITHUB:**

* What’s the difference



1. Git

Git is a versioning tool, and from the word it stores all historical versions of a project e.g., V1, V2, …

Tracking different versions of projects i.e changes and history

One is able to get to any version of the project

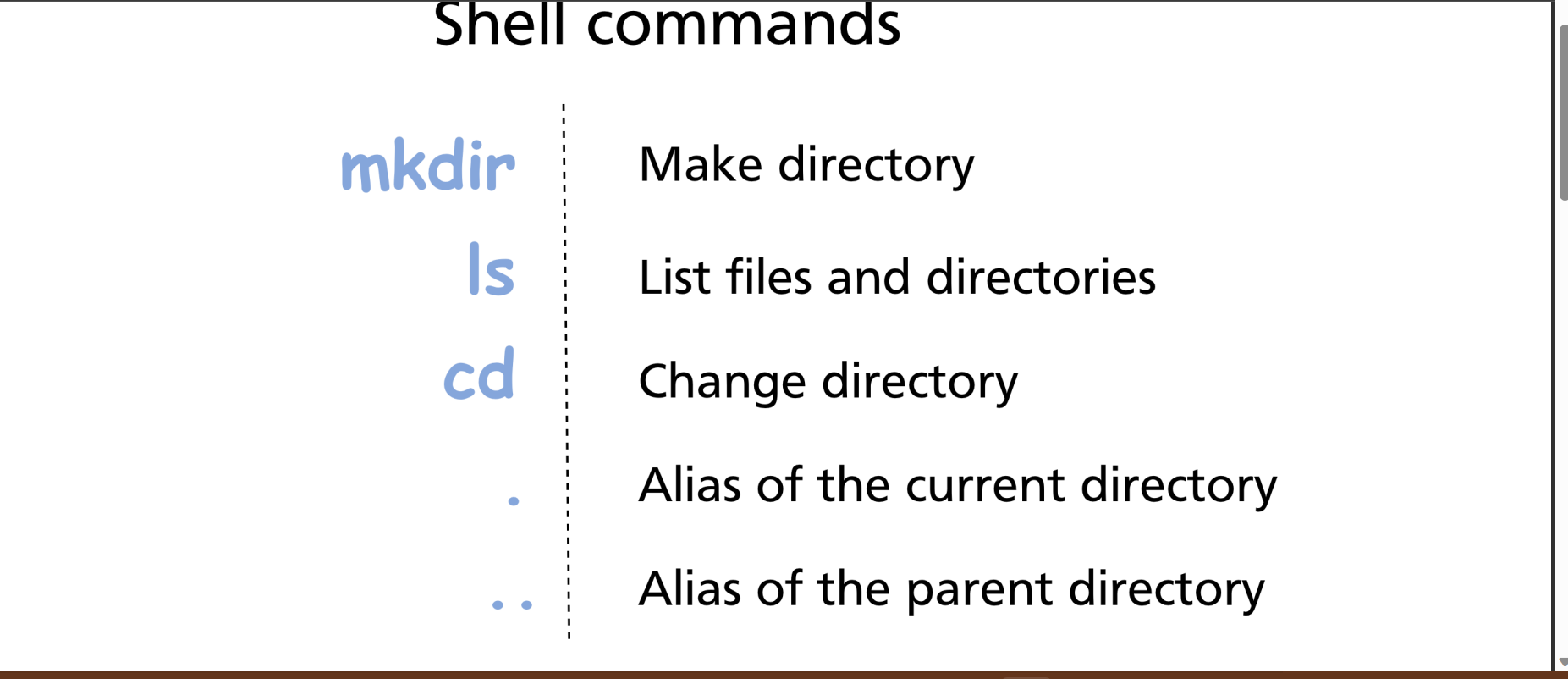
1. **Github:**

* Repository hosting service
* utilizes git for collaboration with other people
* used as a backup for local repositories

Download Git->bash teminal (allows you to run linux commands in windows)

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

1. **Basic Shell commands:**

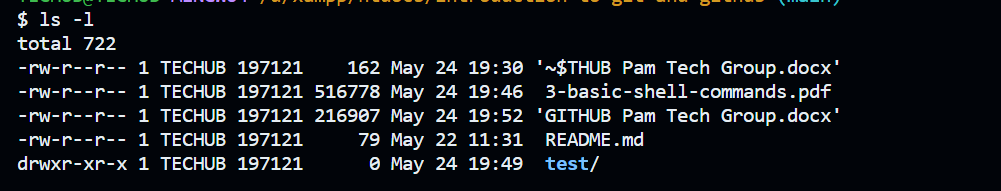


\*An Important Additional command is pwd which outputs the current working directory (where are you at)

. Mkdir



Eg ls -l -> list with more details:



Ls -la (list + hidden files)

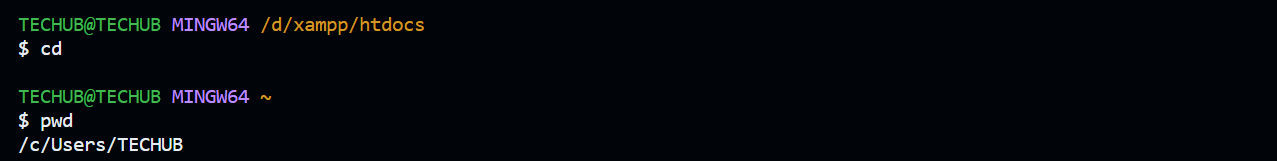


* Notice the .git/ directory that’s where tracking and versioning files/information about our project are stored

Cd and PWD:

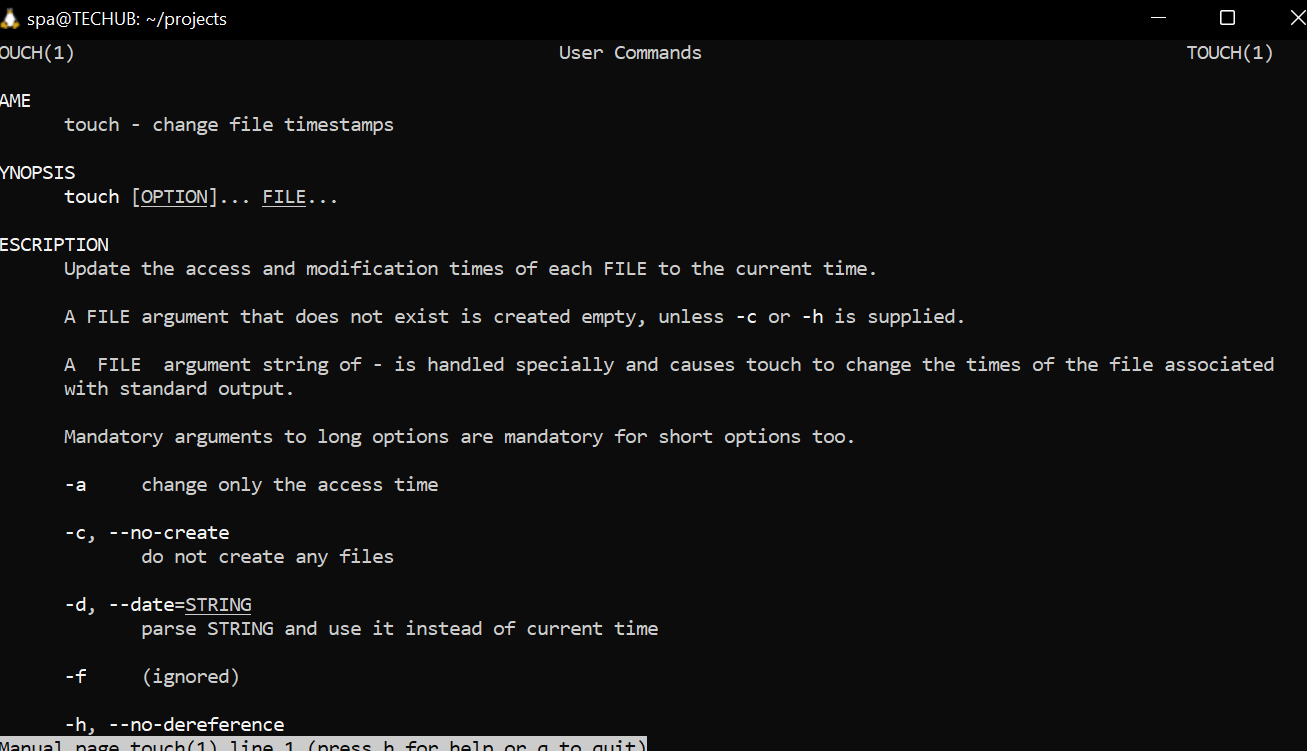


* Cd without any option moves you to your root folder

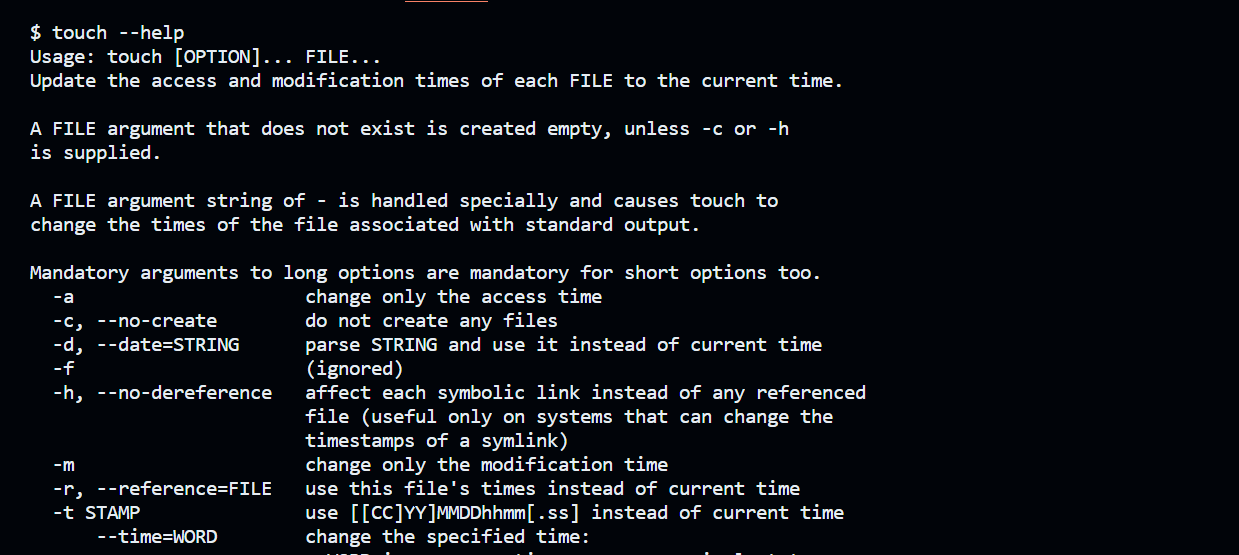


. Man command (For windows is help)

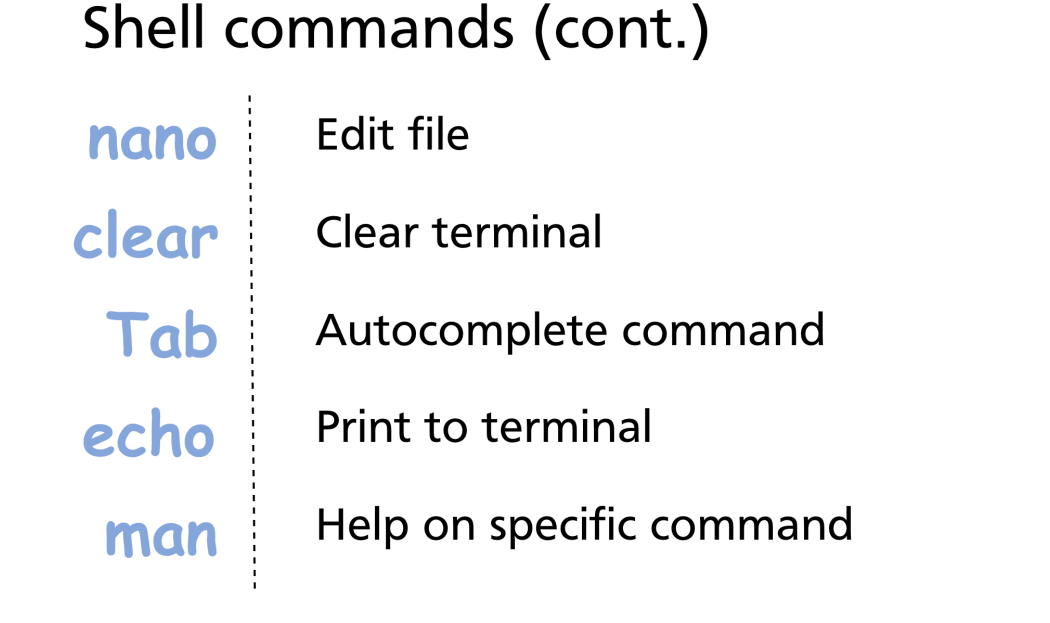
-linux



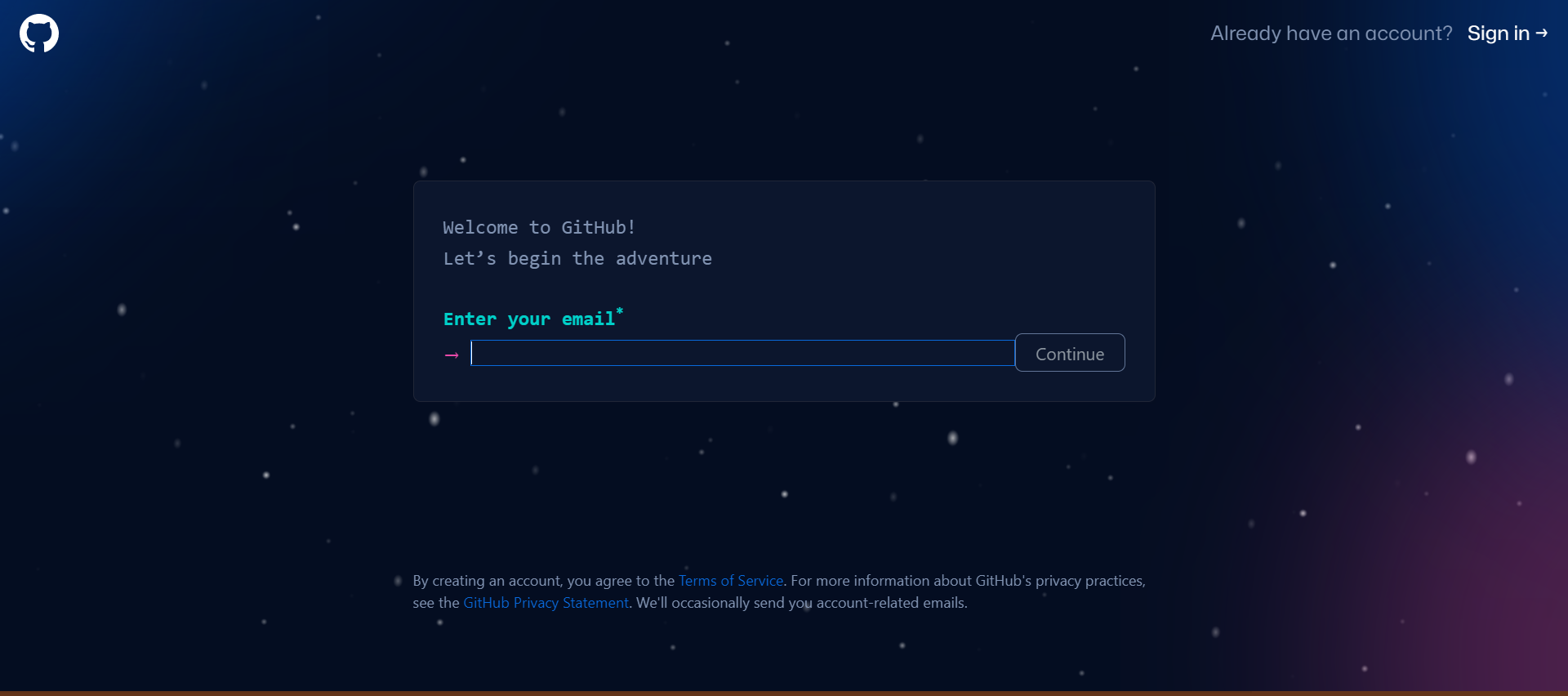
-windows

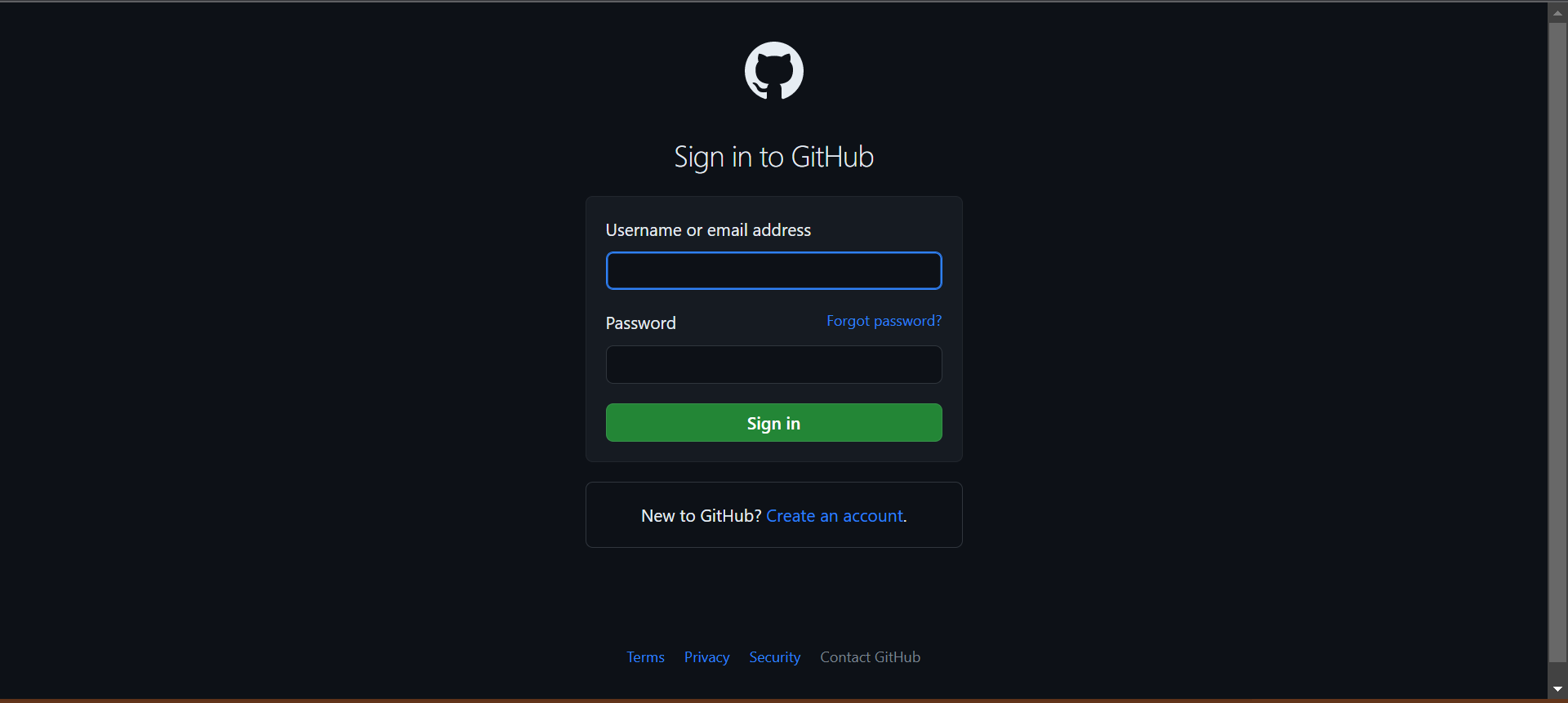


More shell commands:

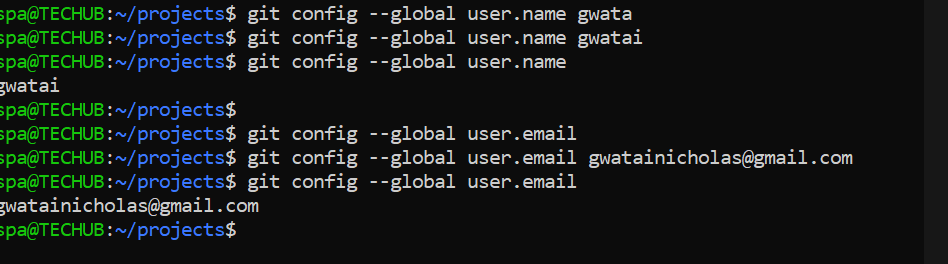


Nano command is used to edit files, so lest fitst create the files using echo, touch , > and >>

1. Creating a Github account: 
2. Signin in:



Basic Configuration:

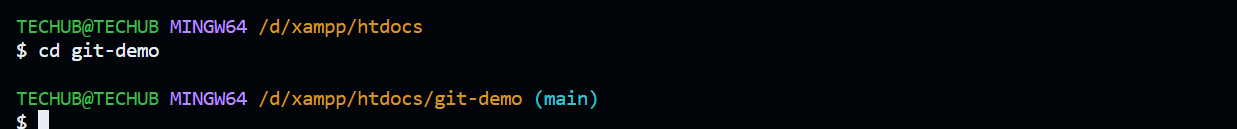


1. Creating a project:

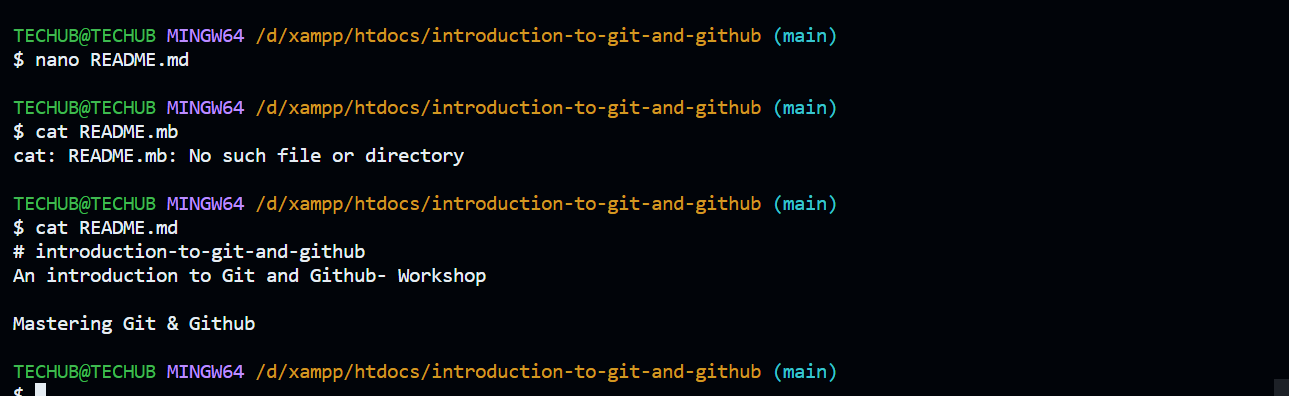
**New repo locally**

* Git init





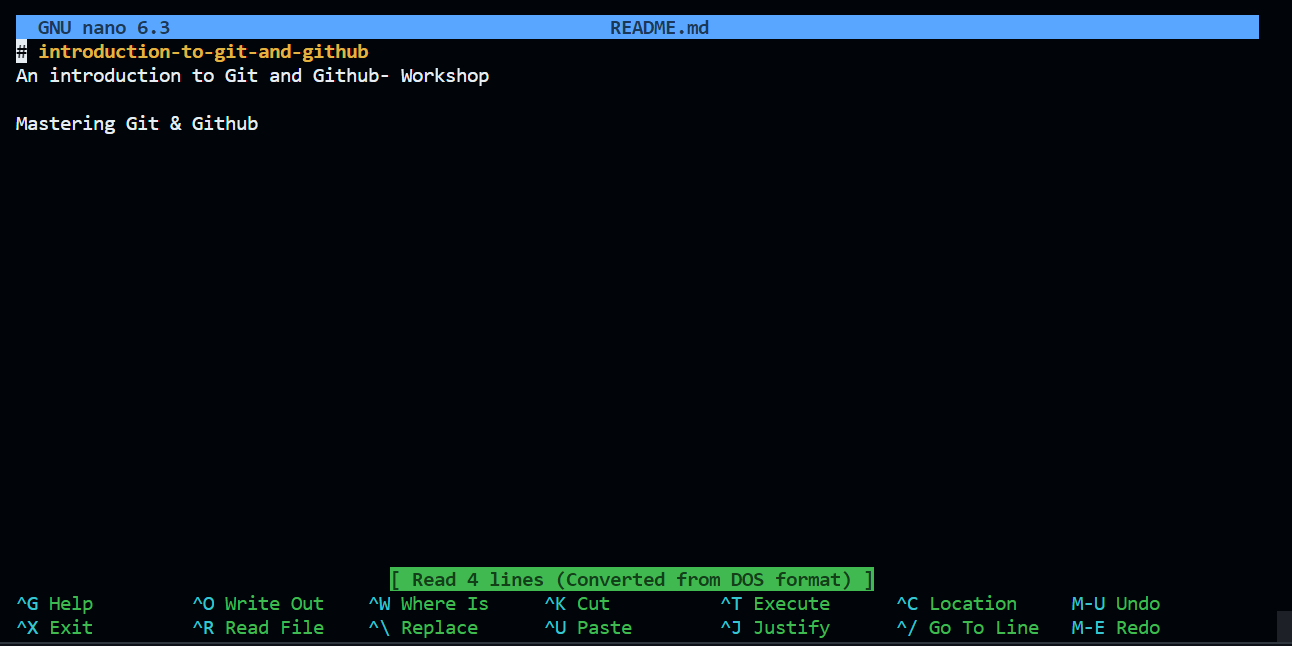
Editing files and making commits:



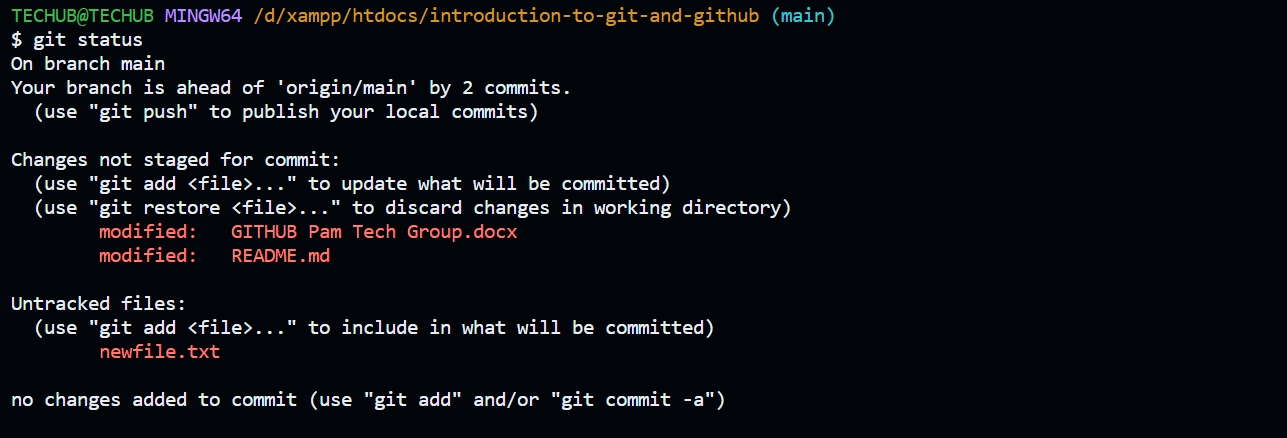
Editing using notepad:



Using nano:

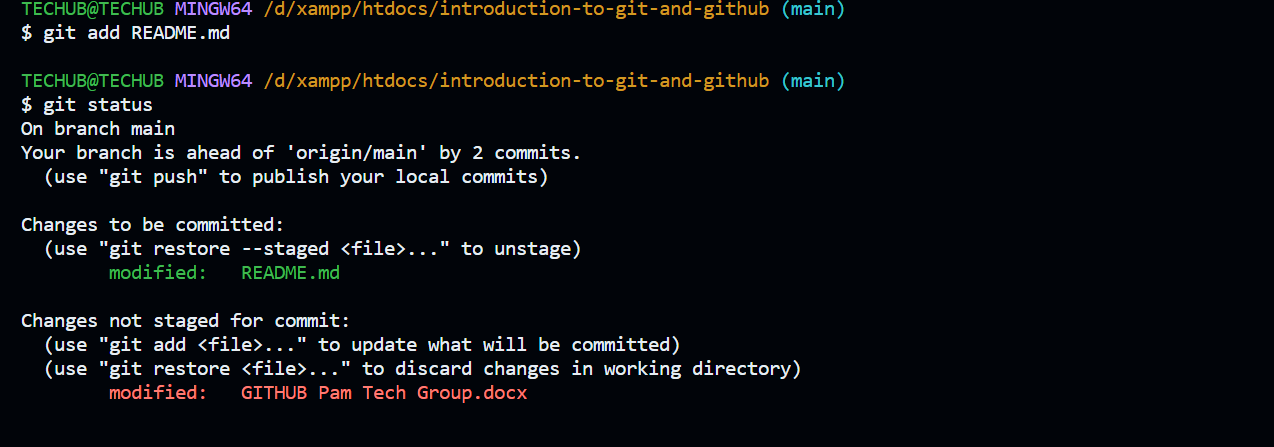


Git status:



* Git alerts us that the files have not been added to the staging area, and Git isn’t tracking them and the changes not yet committed

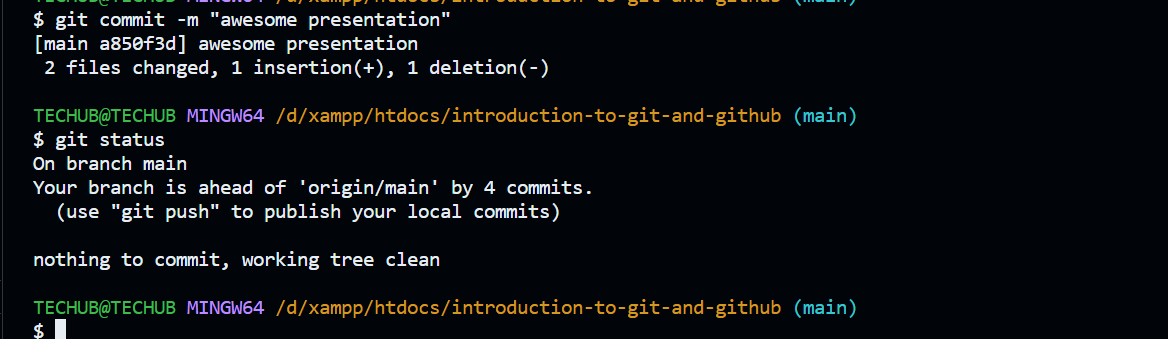
Stage changes (command is git add ) :



Git add . (Stage all project files) :

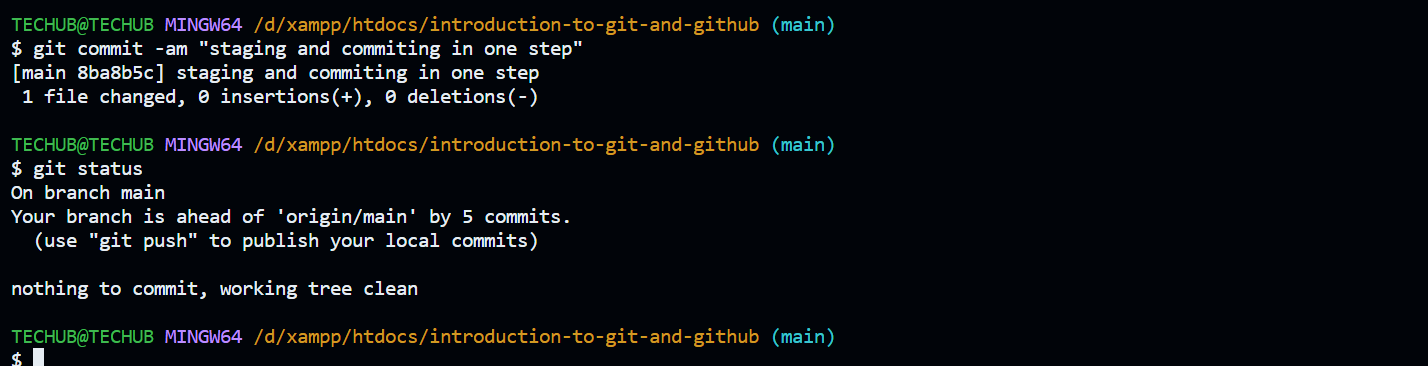


Committing :



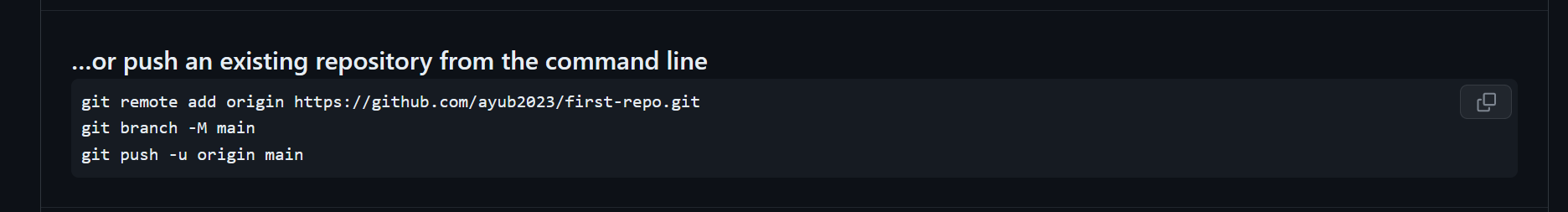
! TIP:

Adding and committing in one step:



More

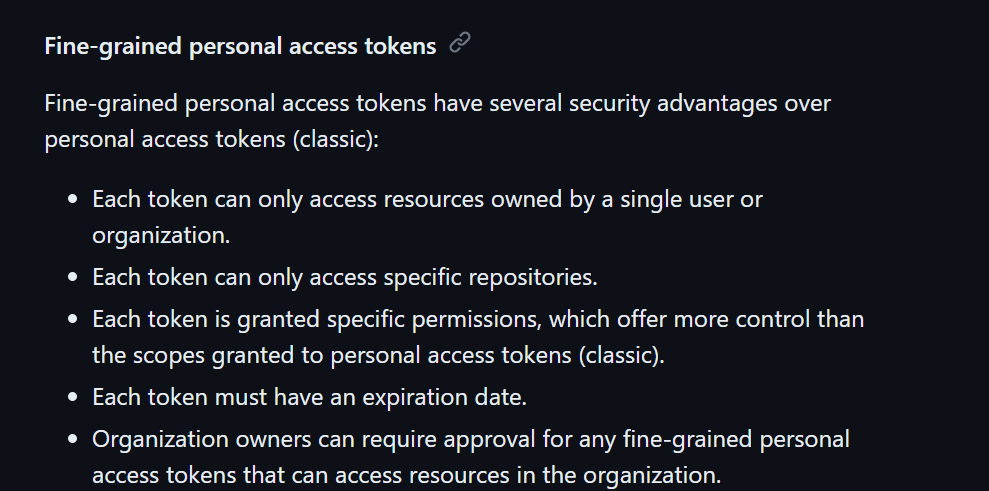
Pushing our local repo to a Remote Repo:



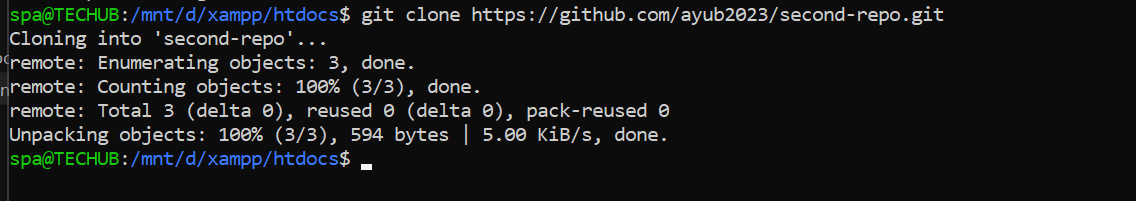
\*You have to create a repo at GITHUB

1. Cloning a project:

Authentication using access tokens:

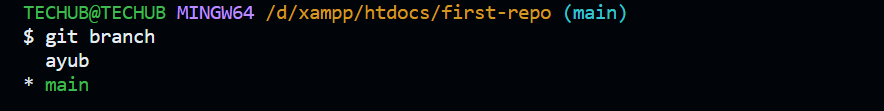


1. cloning our own repo:

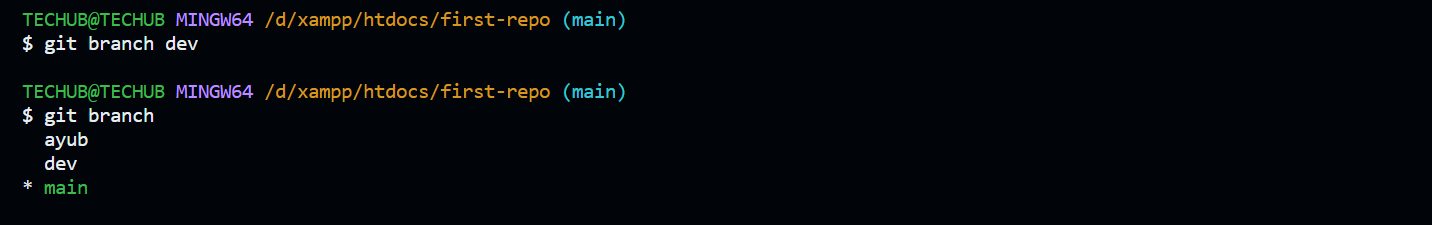


* Create a new branch :

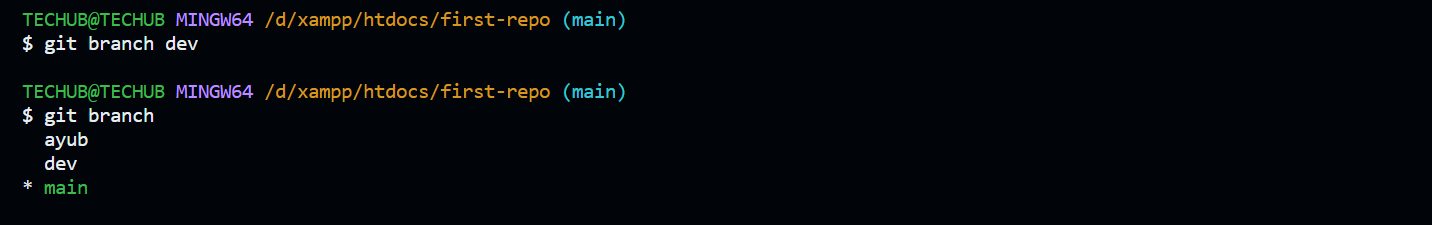
1st check current branch



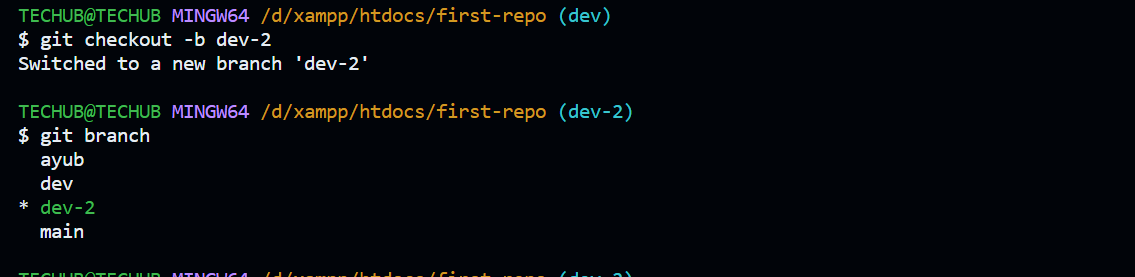
* Create a new branch

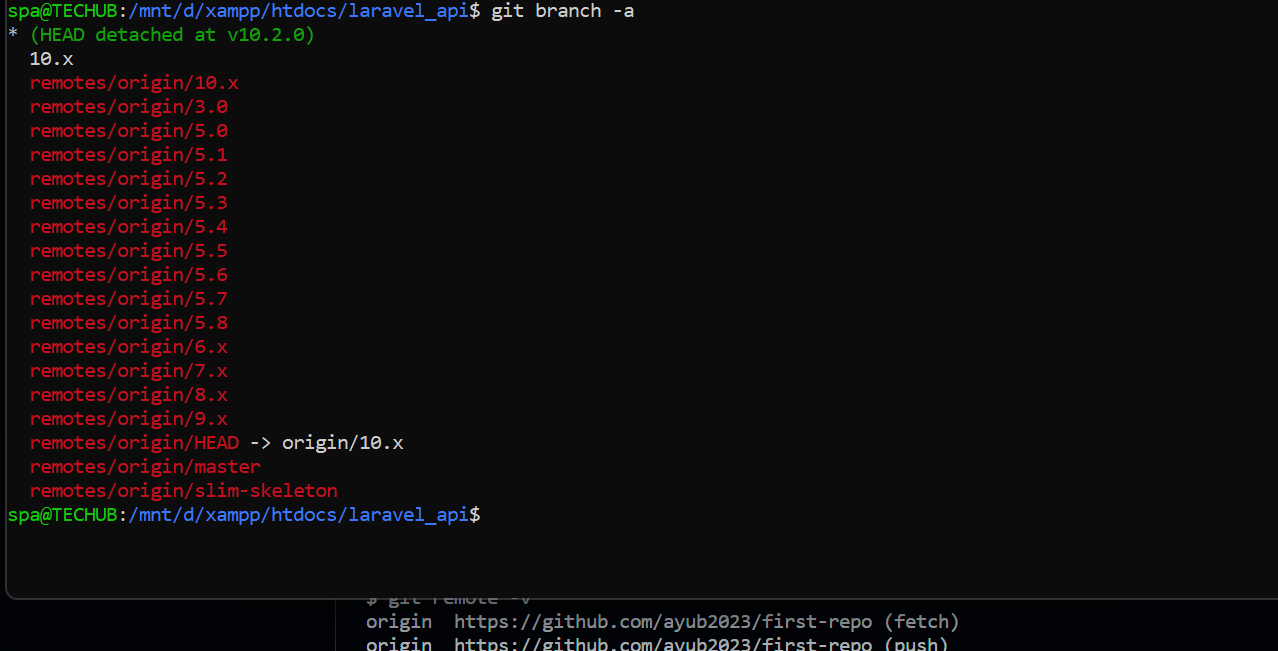


* Enter the new branch:



\*combo -> create new branch and enter the new branch at once:



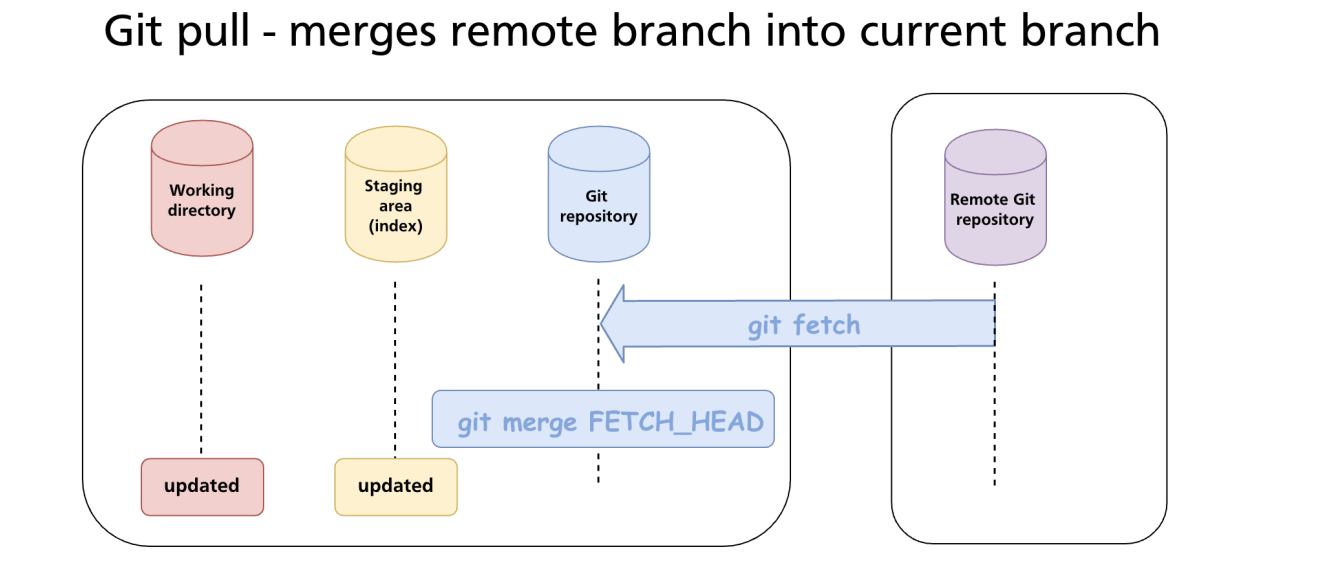
\*listing all branches

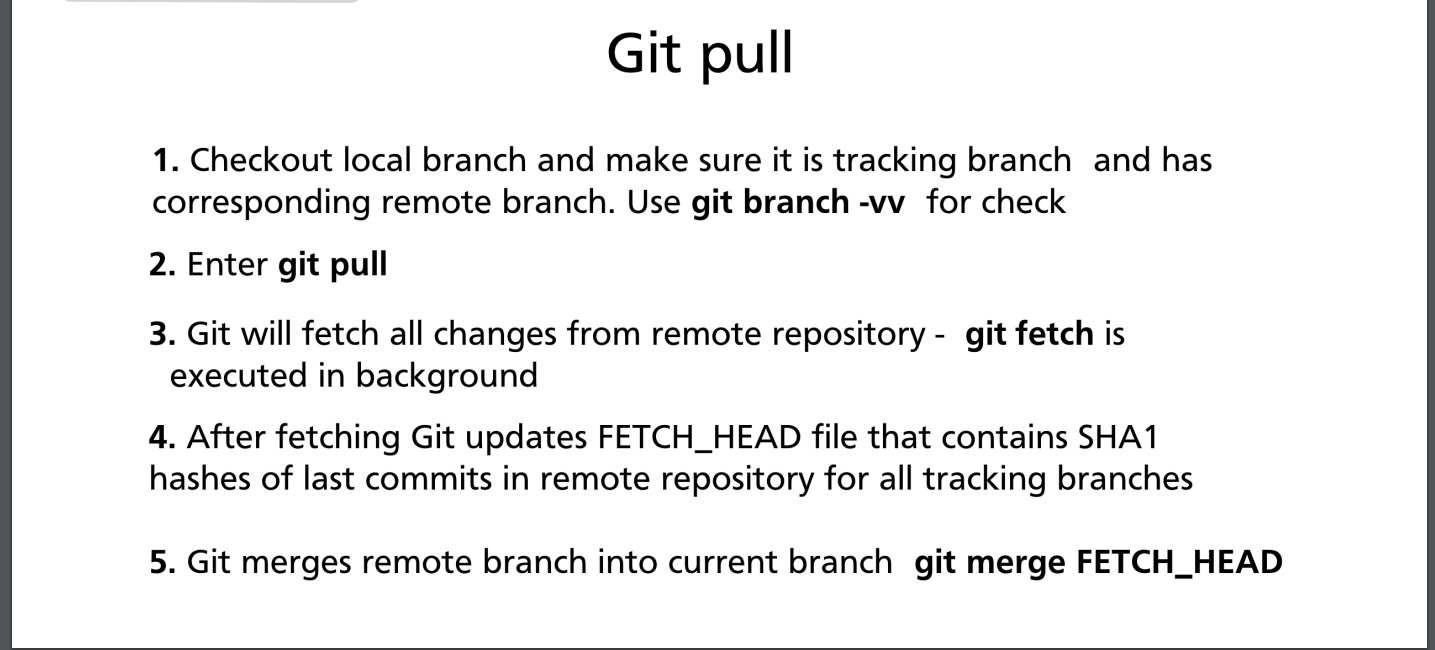
1. Git Fetch, pull & Push

* Push development branch for pull request



Both Fetch and Pull fetch updates from the remote repo, the difference is that with fetch it doesn’t affect you local commits whereas merges completely with the remote updated branch





* Pushing local changes:
* Contributing to a Project
* Clone
* Push
* Pull changes

1. Pull requests

Getting contribution from a project

* Adding contributors
* Pulling a request & merging into master/ main

Git branches & Head

* How to switch between different branches

Cloning Public repositories:

Forking and contributing to public repositories:

Creating pull request from the forked repositories

Merging and rebasing:   
merging is mainly for intergrating features worked on to main or master

Git tags versioning projects:

Rebasing:

Comparison with merging

Git ignore:

Ignoring certain files or even folders

Detach Head:

* Making experimental commits

Advanced Github:

1. Github pages hosting static web sites

* Attach domains

Git hooks:

* You can check commit messages
* Running automated tests
* Verifying index of the code