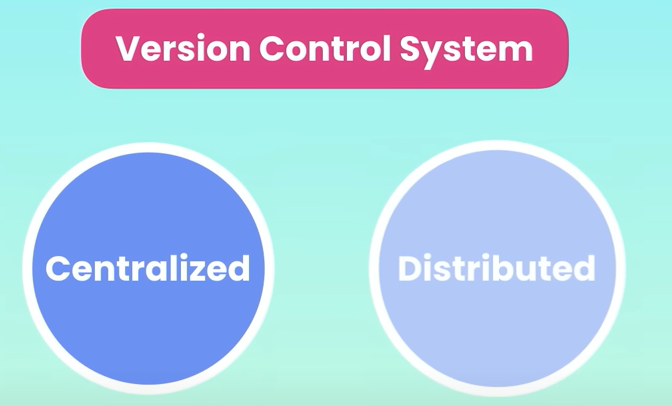
Git is the most popular and widely used version control system in the world and it is free and open source. Version control system is used to keep track of the changes that we have made to code in a special database called “repository”.

We can see the project history, who has made particular changes and why did he make changes like that. If we have made a mistake in check in then we can easily revert back our changes into earlier state.

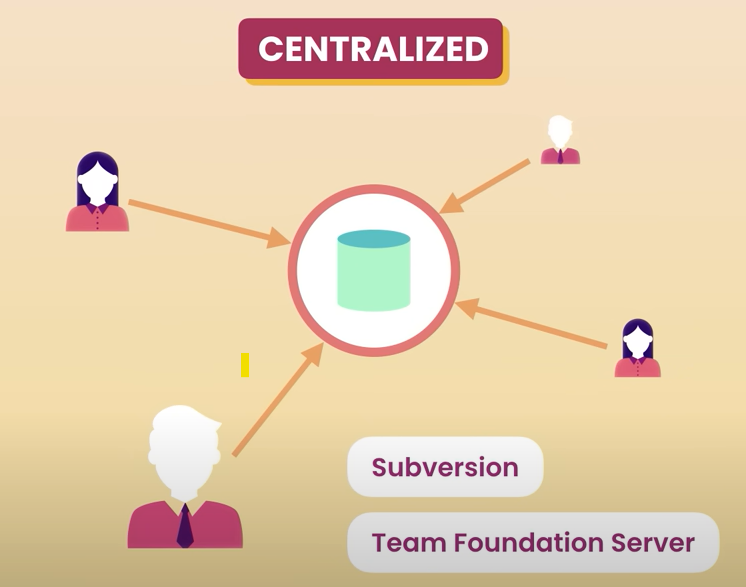


The version control system falls into two categories:

1. Centralised
2. Distributed

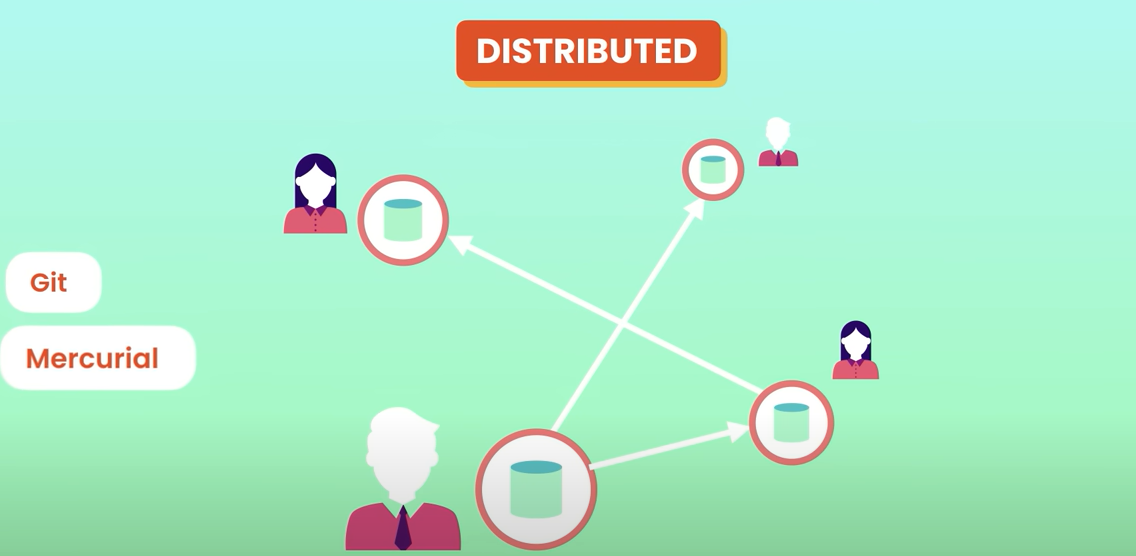


In a **centralized** system all the team members will be connected to a centralised server to get the latest copy of the code and to let everyone know about the changes in the code.



The problem with the centralized architecture is that if the server goes offline we can not merge our code or take a snapshot or get the latest project till the server comes back online.

In the distributed architecture every team member will be having a copy of the project with the history on their machine, so we do not have this problem. If the server is offline we can synchronize the changes directly with the teammates.

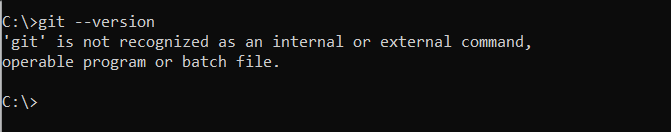


Git and Mercurial are the examples of the distributed version control system however the Git is the most popular version control system in the world because of the following reasons:

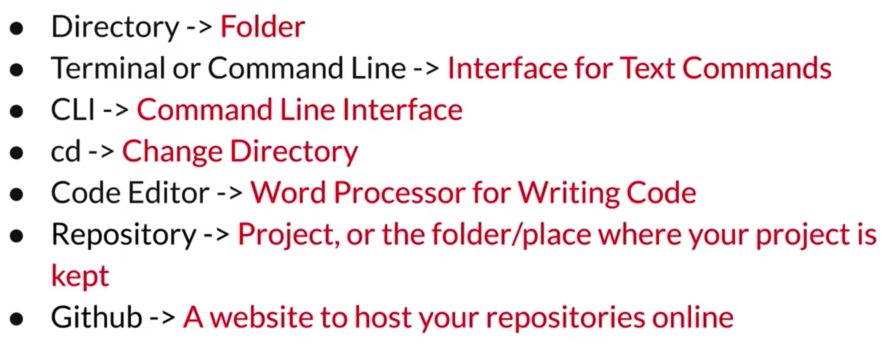
1. It is free to use.
2. It is open source.
3. It is super fast.
4. It is Scalable.

How to know whether the Git is installed on our machine ?

Open a command prompt and type in **git –version**. If it is not installed then we will get a error message like below



Some of the terms that we should be familiar with while using the Git.



* **Github** - Frequent confusion arises between Git and GitHub. To clarify, Git is a version control tool utilised to monitor code changes over time, effectively managing the development process. On the other hand, GitHub functions as a web-based platform where we can host and manage our Git repositories, providing a collaborative and centralized environment for teams to work together on projects.

Frequently utilized terminologies and Git commands include:



1. **Clone** - This command is employed when a repository exists on GitHub but not on the local machine. It allows users to retrieve the repository and its contents to their local environment, enabling them to make changes and updates.

2. **Add** - When modifications are made to files, or new files are created or deleted, the "add" command informs Git to track these changes. It prepares the changes to be included in the next commit.

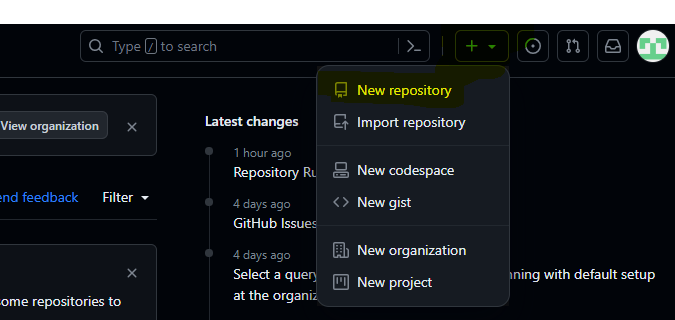
3. **Commit** - As mentioned earlier, Git's purpose is to record changes made to the codebase. The "commit" command is used to create a snapshot of the changes made since the last commit. This action effectively saves the changes to the local repository.

4. **Push** - Once changes have been committed locally and are ready to be uploaded to a remote location, such as GitHub, the "push" command is employed. It transmits the committed changes to the remote repository, making them available for others to access.

5. **Pull** - When changes have been made to the remote repository, for example, by a teammate, and these changes need to be brought into the local repository, the "pull" command is used. This action fetches and merges the changes from the remote repository into the local workspace.

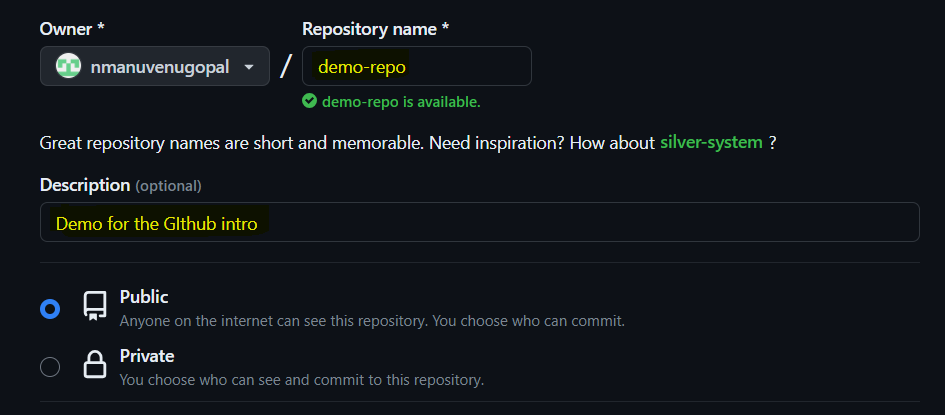
The next thing we need to do is to create a repository. We can create a directory in Github using following ways:

First way is to use the “+” symbol located near to the user profile picture.



Second way is to use the “New” button on the top left corner.

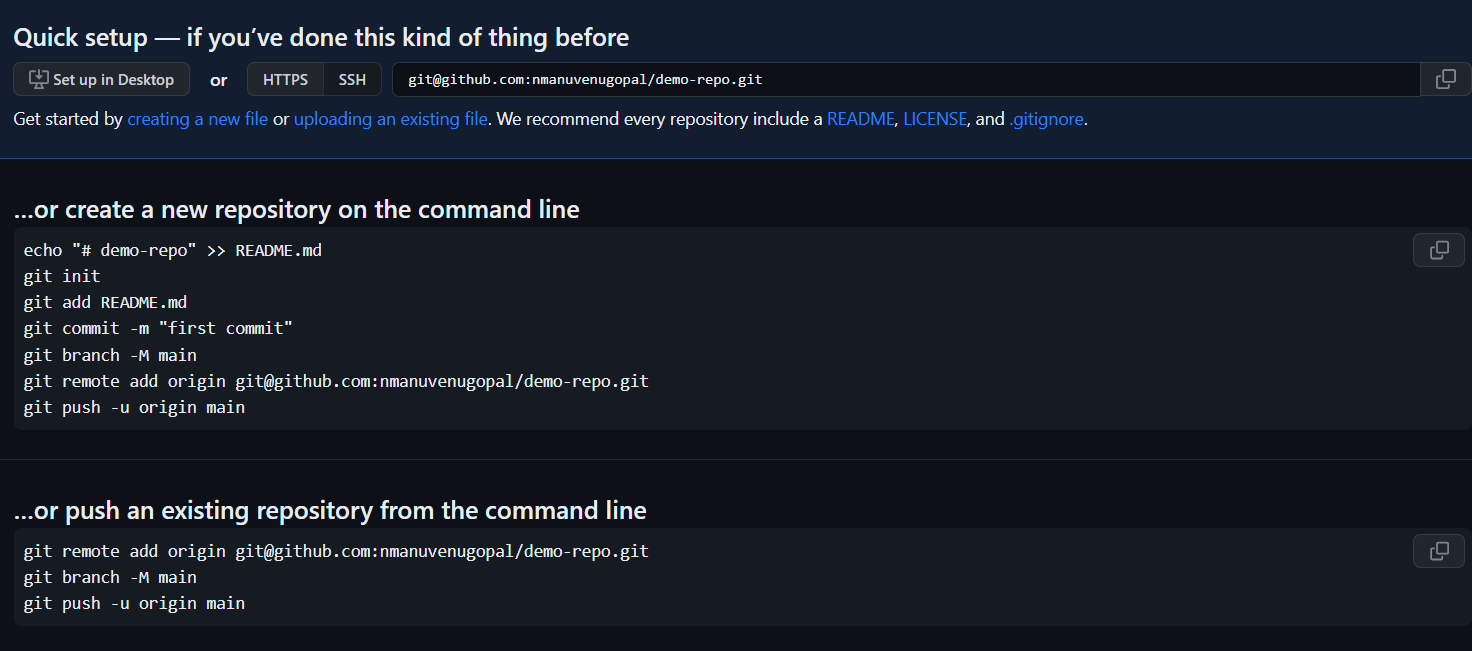


Click on the add button, fill in the details and click “Create repository” button. 

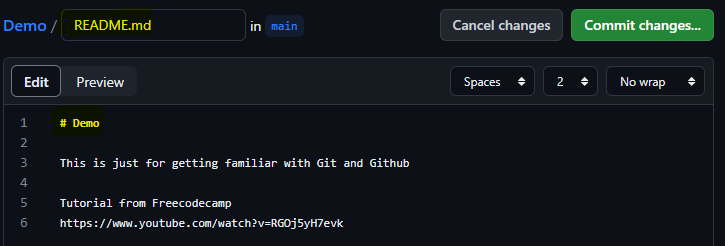
Repository is our project, it contains all the files and subfolders that project utilises to work properly.

We can create the files and folders for the repository locally on our machine or we can create it through the online editors on the Github website.

After we click on the create repository button, I am going to create a basic readme markdown file. Readme.md is the most basic file that contains the basic information about the project, what it does or any other specific information or details.

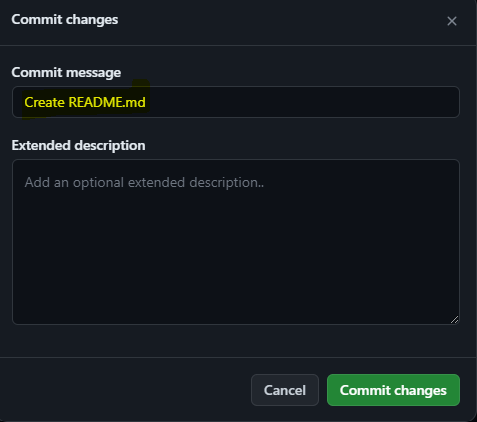


I have clicked on creating a new file and I need to name it as README.md file. Markdown file have many shortcuts like “#” for the main header.

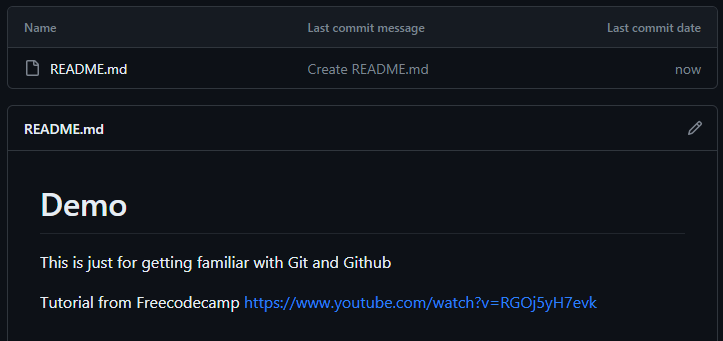


Once I enter the details about the project in the README.md file, the next thing will be committing the changes we have made. For that we need to click on the commit change button.

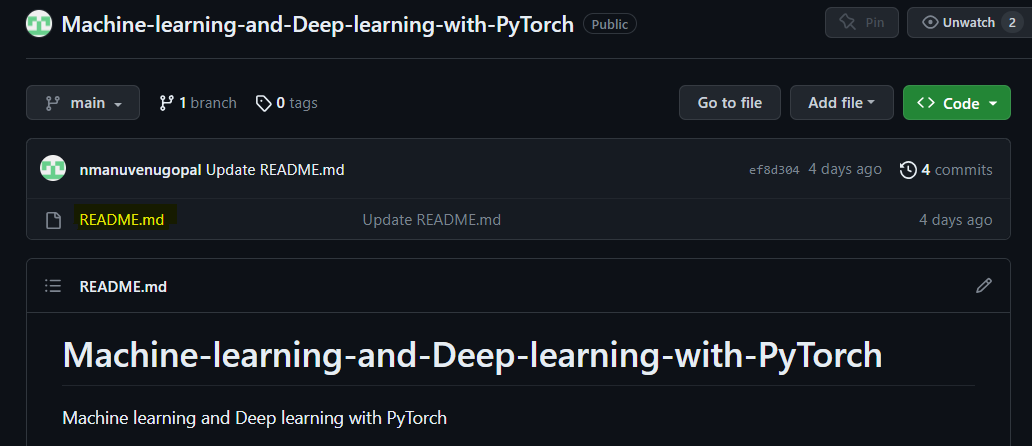
Upon clicking the commit changes button, a dialog box will appear:



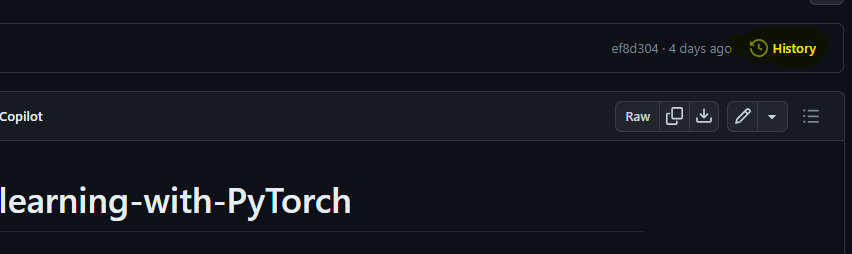
We can add or update the details and click on the commit changes button for saving our changes.



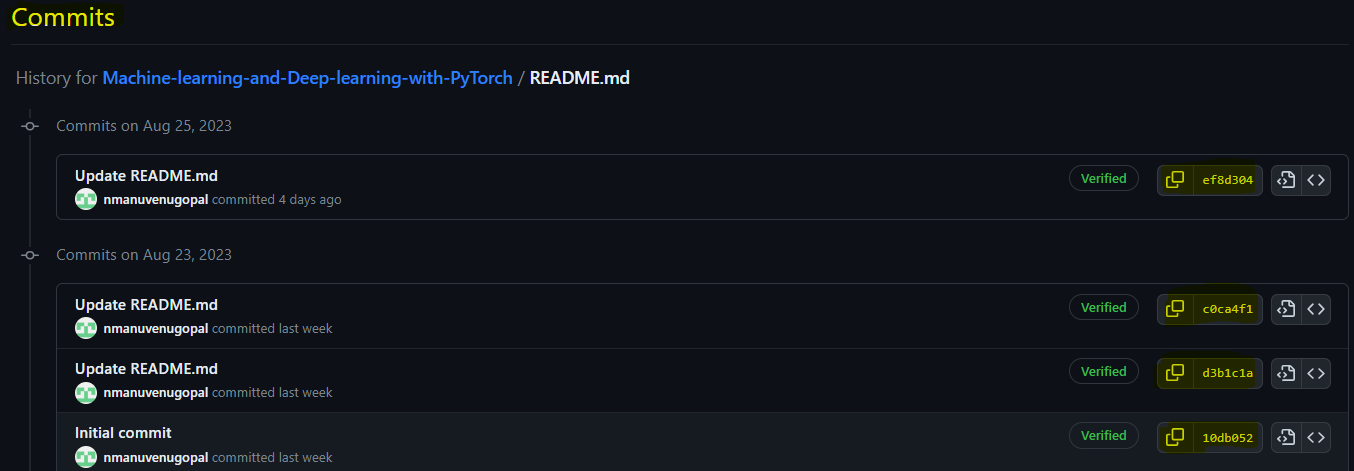
Suppose if we want to see the changes made to the README file then



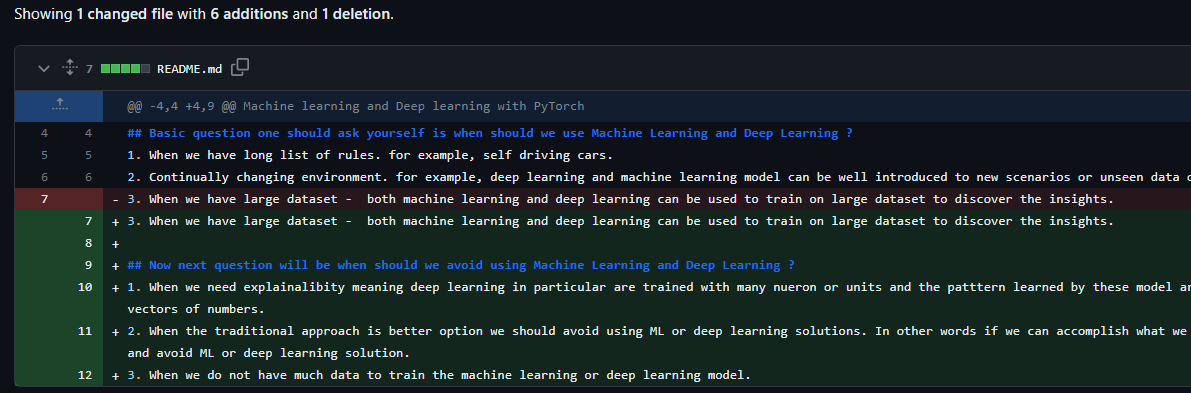
1. Click on the file for which we need to see the history of changes.



1. On the top right corner, we can see the history option, then we can see the list of changes and unique id associated with every change.



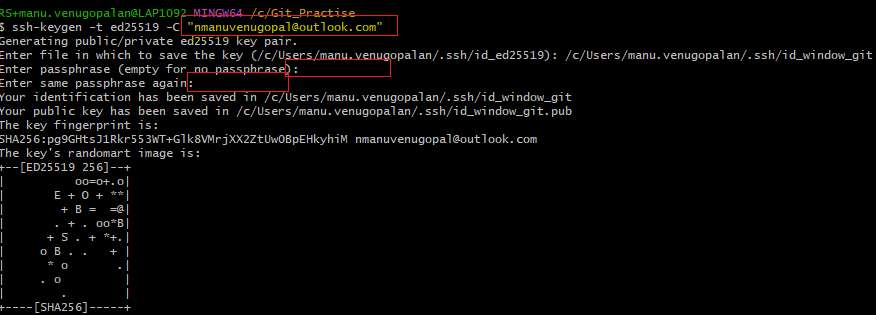
1. If we click on a item above, then we can see the changes corresponding to that commit

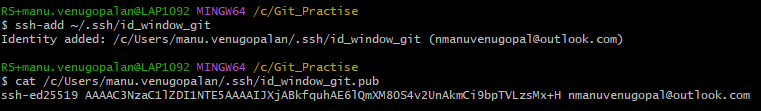


We need to create SSH and KEY for performing the various operations in Git. For setting up an SSH key kindly follow the steps in below link:

<https://docs.github.com/en/authentication/connecting-to-github-with-ssh/generating-a-new-ssh-key-and-adding-it-to-the-ssh-agent?platform=windows>

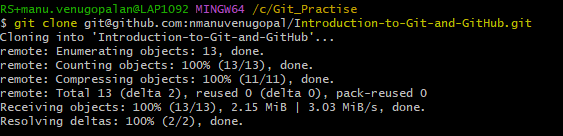
I have done this through the Git Bash(if using windows it is recommended)



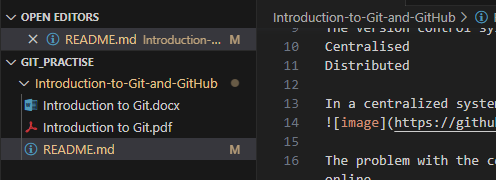


For getting familiar with Git we need to have experience or hands-on experience.

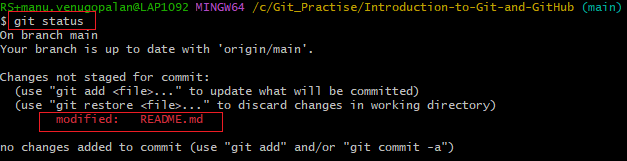
1. I have created an empty folder in my local drive and opened it in VS Code.
2. After opening the empty folder “Git\_Practise”, now a new terminal window in VS code.
3. Now we can use the git clone command to import the code from the github to the local.



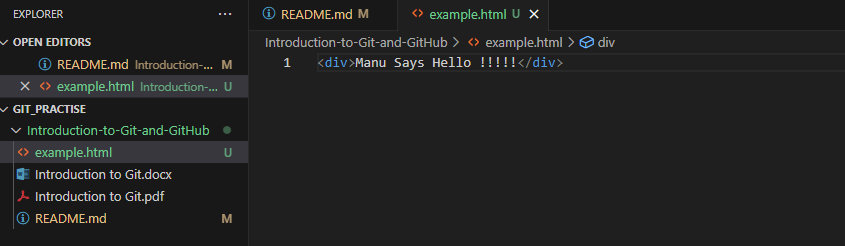
1. After these steps, if we go back to the VS Code then we can see that the files in the corresponding repo are downloaded to our practice folder.

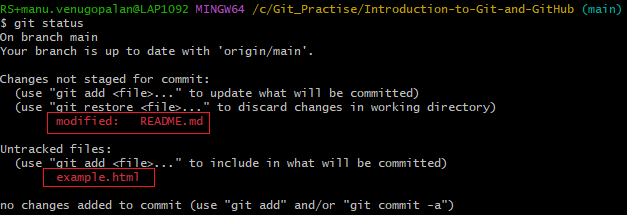


1. I have made the changes to the readme file via VS Code and need to know the details of the files I changed, for this we can use the “status” command.

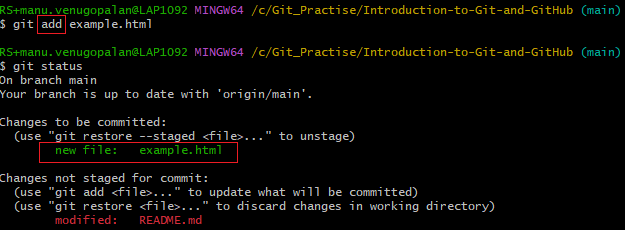


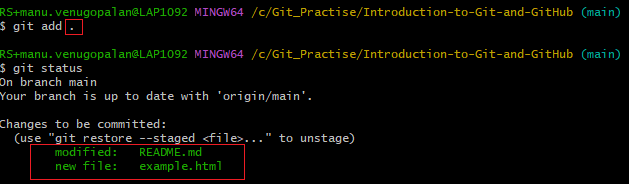
1. What happens if I add a new file, let it be an example.html file to the repository via VS Code then the “git status” returns the following. Status says that example.html is not tracked, that means it is a new file (Git doesn’t know about the file yet) and I have tried to modify the readme file too.





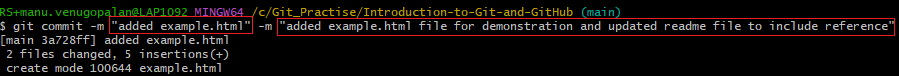
1. For adding the files we can use the “add” command and there are two ways in which we can use the add method. Add with the specific file name or Add with period (‘.’).



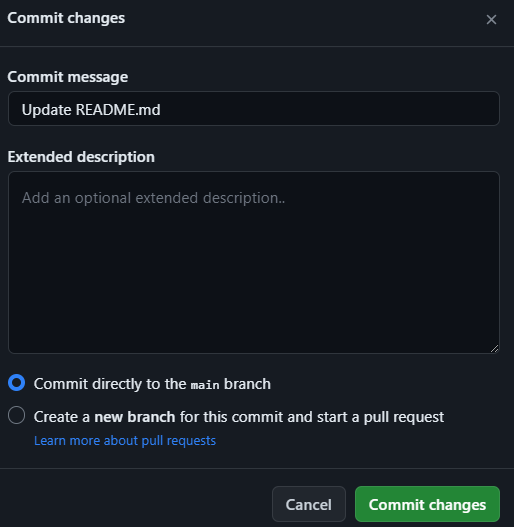


If we are using period (.) then we are saying the git to track all of the newly added and updated records or files. In our case if we add a period then both readme and example.html will be staged with git. But in order to experiment i have used both add with specific file name and add with period too.

1. Now we can see that all of the updated and untracked files are now staged to git and they are ready to be committed. For committing the code we use the commit command.



First description is for the commit message in the commit command and the second description is for extended description.



1. We have still saved our code locally, but the commit isn’t live in GitHub yet. For making it live in GitHub we use another command “push”. Push command will push the changes to a remote repository where my project is hosted.

Reference

Git and Github for beginners - freeCodeCamp

<https://www.youtube.com/watch?v=RGOj5yH7evk>

Git Tutorial for Beginners: Learn Git in 1 Hour

<https://www.youtube.com/watch?v=8JJ101D3knE&t=880s>