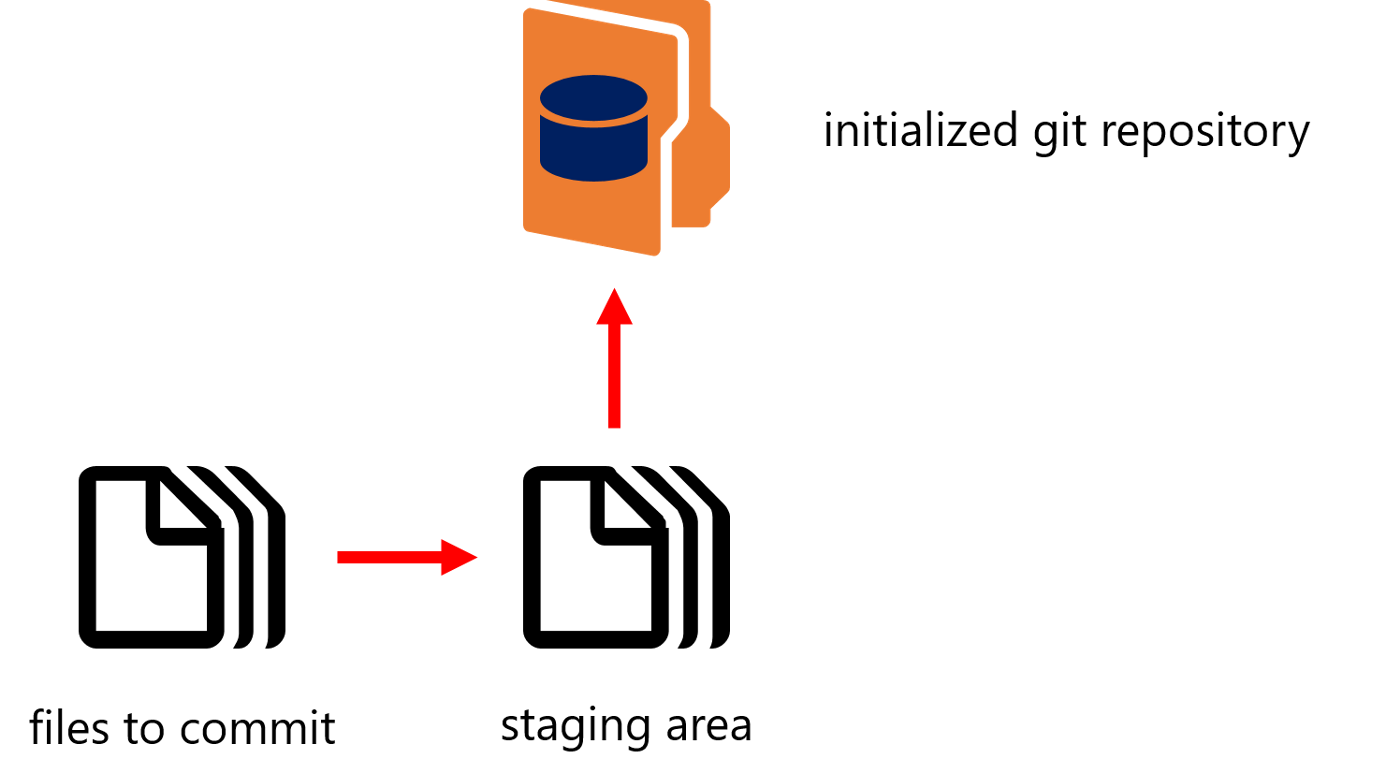
# **EDUNET FOUNDATION-Class Exercise Notebook**

Lab 3 - GitHub Commands

With Git, you record local changes to your code using a *command-line* tool, called the “Git Shell” (you can use Git in other command-line tools — Refer to Git Shell through the following sections). Command-line lets you enter commands to view, change, and manage files and folders in a simple terminal, instead of using a graphical user interface (GUI). If you have not used command-line before, don’t worry, once you get started, it is incredibly straightforward.

****

Essentially, when using Git, you make changes to your code files as you normally would during the development process. When you have completed a coding milestone, or want to snapshot certain changes, you add the files you changed to a staging area and then commit them to the version history of your project (repository) using Git. Below, you’ll learn about the Git commands you use for those steps.

**Terminal Commands**

While using Git on the command line, chances are you will also use some basic terminal commands while going through your project and system files / folders, including:

* **pwd**- check where you are in the current file system
* **ls** - list files in the current directory (folder)
* **cd [directory-name]** - moves to the given directory name or path
* **mkdir [directory-name]** - makes a new directory with the given name

**Creating Repositories**

When you wish to utilize Git for a project, the first command you must do is *git init*, with the name of your project:

**git init [project-name]**

You run this command on the Git Shell command-line in the main *directory*(folder) of your project, which you can navigate to in the Shell using the commands listed above. Once you run this command, Git creates a hidden .git file inside the main directory of your project. This file tracks the version history of your project and is what turns the project into a Git *repository*, enabling you to run Git commands on it.

**Making Changes**

* **git add [file]** or**git add \***

Once you make changes to your files and choose to snapshot them to your project’s version history, you have to add them to the staging area with *git add*, by file name, or by including all of the files in your current folder using *git add \**.

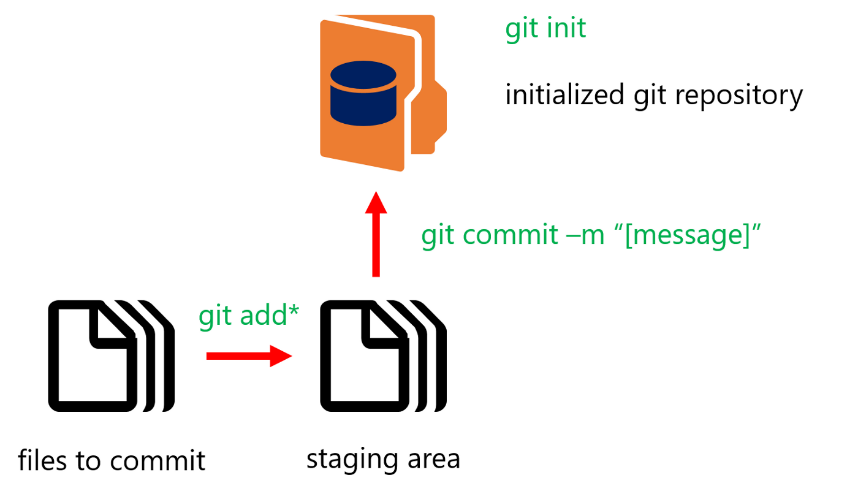
* **git commit -m “[message]”**

To finally commit the changes, you made to your files from the staging area to your repository’s version history, you need to run *git commit* with a descriptive message of what changes you made.

* **git status**

If at any point, you wish to view a summary of the files you have changed and not yet committed, simply run git status in your project’s repository on the Git Shell command-line.

**How It Works**

****

GitHub Workflow

Now, with the basic Git commands in place, you can utilize Git to snapshot the version history of your project. Simply initialize a new repository by running *git init* in your project’s main directory. Using *git add \**, or *git add* with specific file names, you add your changes to the staging area. Finally, using *git commit*, you can add your changes to the repository’s version history.

1. Open Anaconda Command Prompt and install git library

pip install git

1. Create a directory with two random .txt files. Place any piece of text in those files.

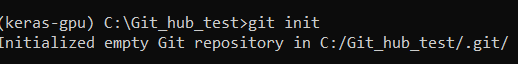
Configure the access of github profile on local github library

git config --global user.name username

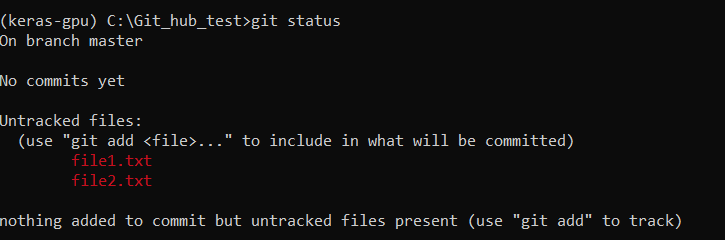
git config --global user.email useremail

\*\* Place your github login username and email information.

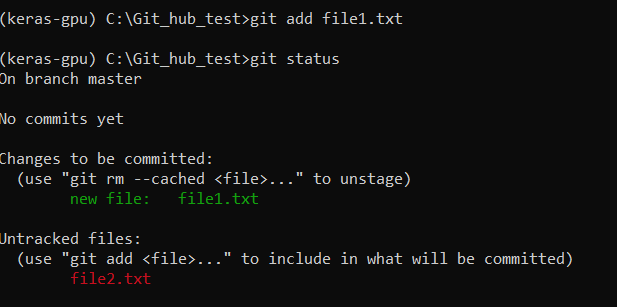
1. Change the folder to as current working directory. And run git init code

****

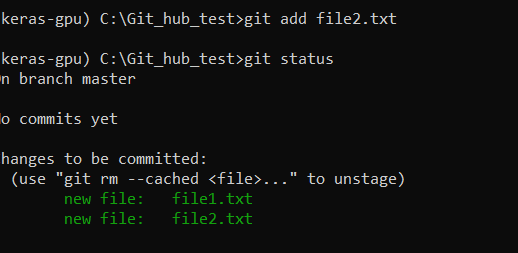
1. Git status shows files are untracked

****

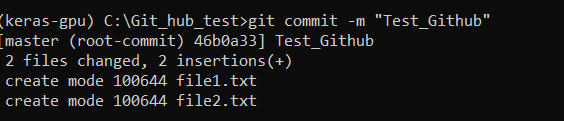
1. Add file1.txt to stage and check git status

****

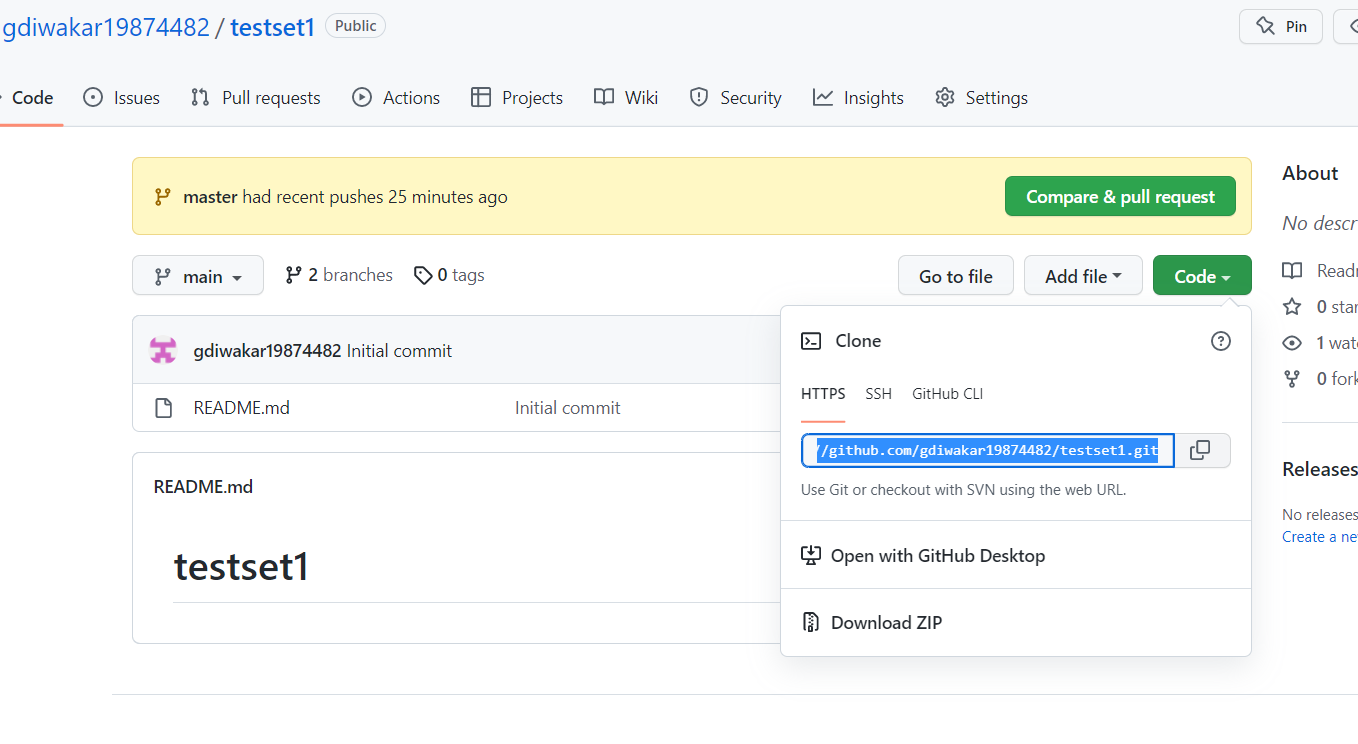
1. Now add another file2.txt to staging stage and check git status

****

1. Commit the traced file using git commit command. You can also mention the commit message.

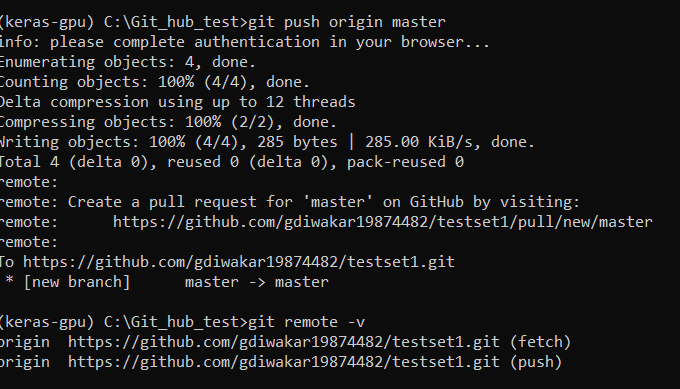
****

1. Add the address of repository to where you test files are to be uploaded.

****

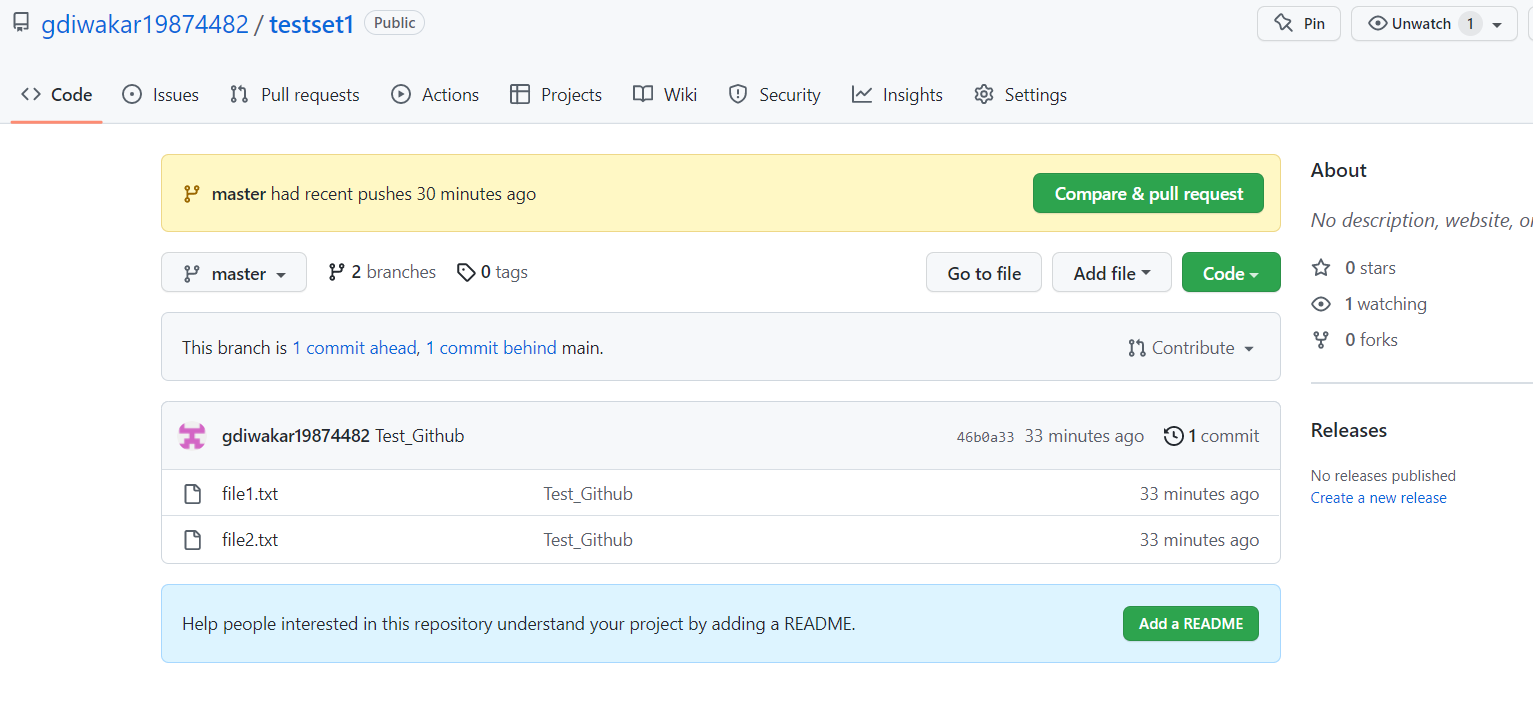
****

1. Upload your local repository to github using git push command

****

\*\*Note: GUI window will request for read/ write access.

1. You will observe and conclude that your local repository gets uploaded to github repository

****