
Introduction to Git & GitHub

—— **Distributed Version Control System** ——

Introduction to Git

- Git is a Distributed Version Control tool that supports distributed non-linear workflows by providing data assurance for developing quality software.
- Primarily used to manage project, comprising a set of code/text files that may change.
- There are two types of VCS:
 - Centralized Version Control System (CVCS)
 - Distributed Version Control System (DVCS)

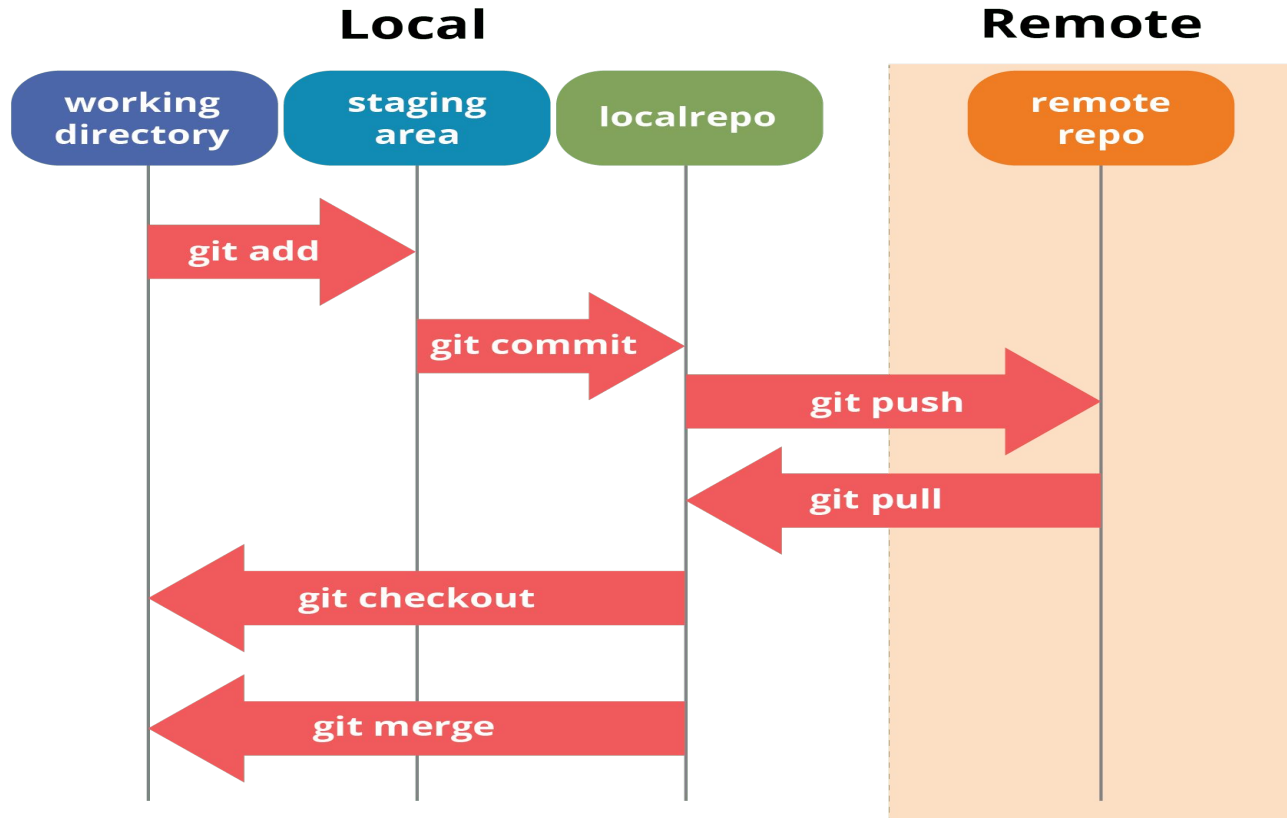
Features of Git

- **Free and open source:** No need to purchase Git. You can modify the source code as per your requirement.
- **Scalable:** As and when the number of collaborators increases, the Git can easily handle this change.
- **Reliable:** Since every contributor has its own local repository, on the events of a system crash, the lost data can be recovered from any of the local repositories.
- **Secure:** Git uses the SHA1 (Secure Hash Function) to name and identify objects within its repository.
- **Supports non-linear development:** Git supports rapid branching and merging.
- **Easy Branching:** Easy to create, delete, and merge branches. Feature branches provide an isolated environment for every change to your codebase.

Git Tutorial

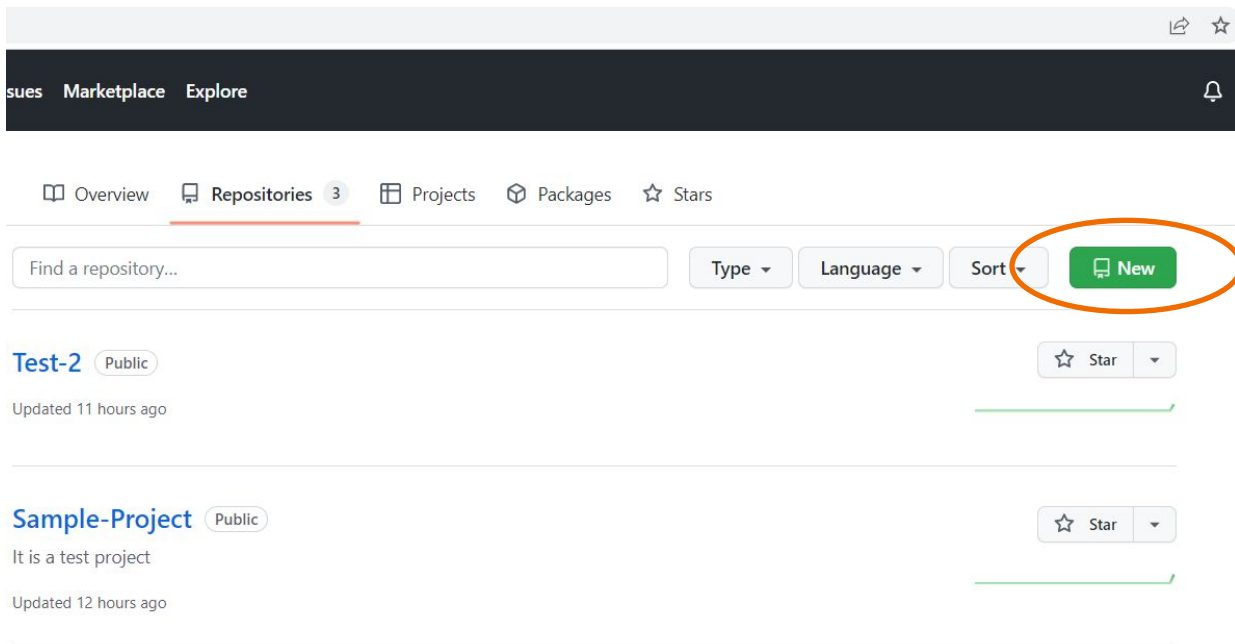
- Some of the basic operations in Git are:
 - Initialize
 - Add
 - Commit
 - Pull
 - Push
- Some advanced Git operations are:
 - Branching
 - Merging
 - Rebasing

Git Tutorial



Remote Repository in GitHub

- Create a new repository in GitHub.



Remote Repository in GitHub

- Add an unique Repository name.
- Select README for any info on the project.
- Choose other settings as per the requirements.
- Click on Create Repository

The screenshot shows the GitHub 'Create new repository' form. The 'Owner' dropdown is set to 'pramodsunagar'. The 'Repository name' field contains 'Demo123' and is circled in orange. Below this, a hint suggests repository names should be short and memorable, with 'urban-rotary-phone' as an example. The 'Description (optional)' field is empty. Under 'Visibility', the 'Public' option is selected. The 'Initialize this repository with:' section has the 'Add a README file' option checked and circled in orange. Below this are sections for 'Add .gitignore' (set to 'None'), 'Choose a license' (set to 'None'), and a note about the default branch 'main'. At the bottom, a note states 'You are creating a public repository in your personal account.' and the 'Create repository' button is circled in orange.

Owner * / Repository name *

pramodsunagar / Demo123 ✓

Great repository names are short and memorable. Need inspiration? How about [urban-rotary-phone](#)?

Description (optional)

☒ Public
Anyone on the internet can see this repository. You choose who can commit.

☐ Private
You choose who can see and commit to this repository.

Initialize this repository with:
Skip this step if you're importing an existing repository.

☒ Add a README file
This is where you can write a long description for your project. [Learn more.](#)

Add .gitignore
Choose which files not to track from a list of templates. [Learn more.](#)

.gitignore template: None ▾

Choose a license
A license tells others what they can and can't do with your code. [Learn more.](#)

License: None ▾

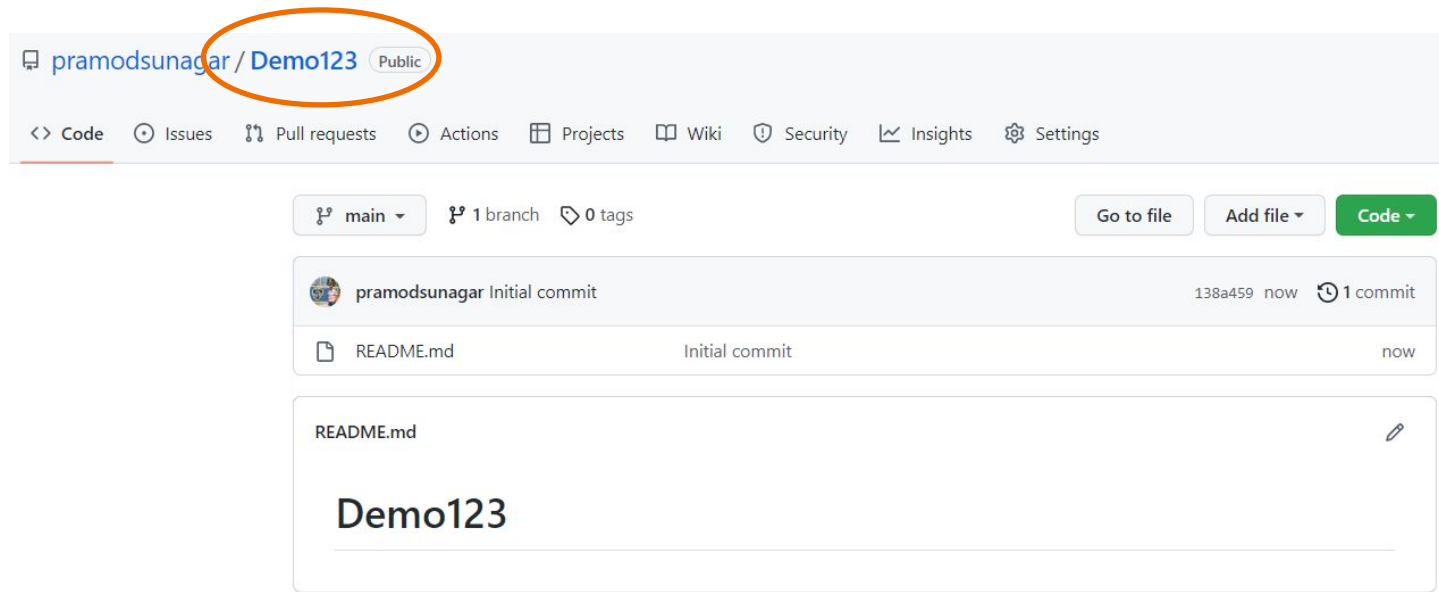
This will set `main` as the default branch. Change the default name in your [settings](#).

① You are creating a public repository in your personal account.

Create repository

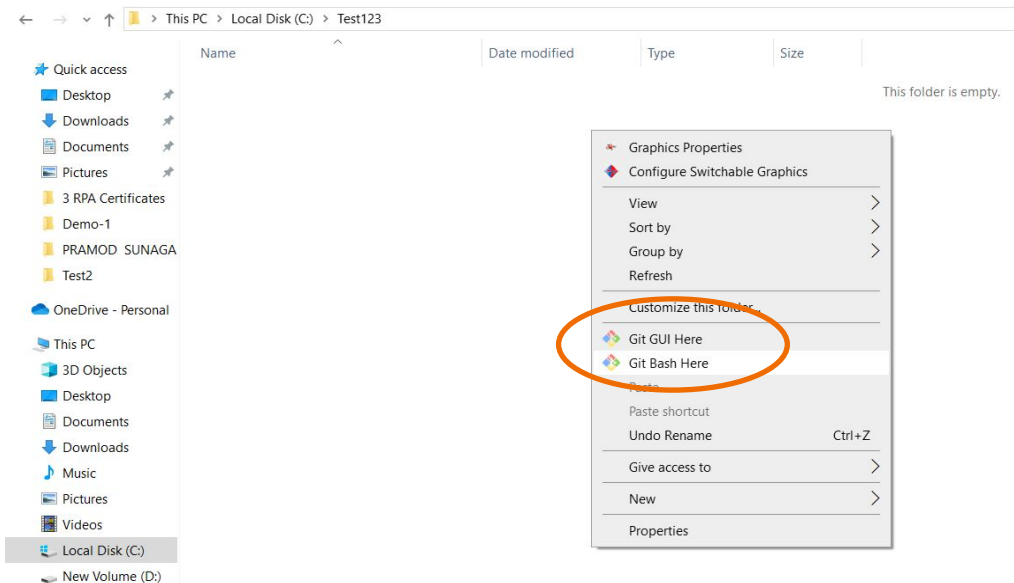
Remote Repository in GitHub

- Remote repository will be created in GitHub.



Git Commands

- Download and install the Git from <https://gitforwindows.org/>
- Create a folder in one drive and right click and select **Git Bash Here**.



Git Commands

- Initialize: The `git init` creates an empty Git repository or re-initializes an existing one. It basically creates a `.git` directory with sub directories and template files.

A screenshot of a Windows terminal window titled 'MINGW64:/c/Test123'. The terminal shows the execution of the 'git init' command. The prompt is 'PS@DESKTOP-6K1ULK0 MINGW64 /c/Test123'. The command '\$ git init' is entered, and the output is 'Initialized empty Git repository in c:/Test123/.git/'. The prompt then changes to 'PS@DESKTOP-6K1ULK0 MINGW64 /c/Test123 (master)' and a new line '\$ |' is shown, indicating the user is ready to enter another command.

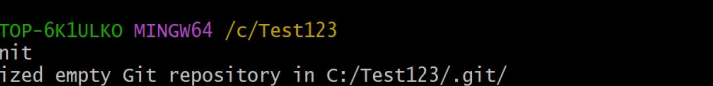
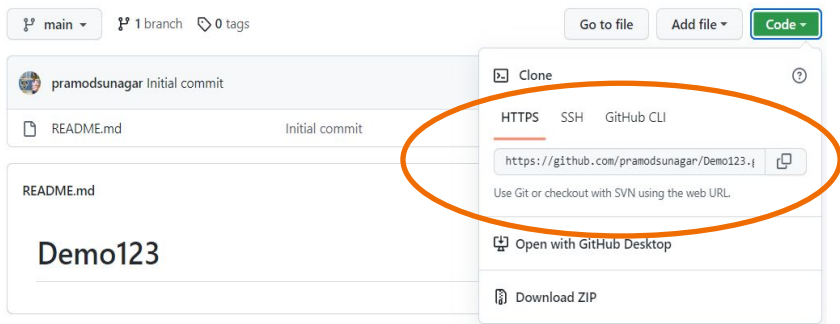
```
PS@DESKTOP-6K1ULK0 MINGW64 /c/Test123
$ git init
Initialized empty Git repository in c:/Test123/.git/

PS@DESKTOP-6K1ULK0 MINGW64 /c/Test123 (master)
$ |
```

Git Commands

- Link to remote repo:
- Copy the web URL of the remote repo to link to local repo.

- git remote add origin "https://github.com/pramodsunagar/Demo123.git"



The screenshot shows a Windows command prompt window with the title bar 'MINGW64/c:/Test123'. The prompt is 'PS@DESKTOP-6K1ULK0 MINGW64 /c:/Test123'. The user enters '\$ git init', and the output is 'Initialized empty Git repository in C:/Test123/.git/'. The user then enters '\$ git remote add origin "https://github.com/pramodsunagar/Demo123.git"', and the prompt returns to '\$ |'.

```
PS@DESKTOP-6K1ULK0 MINGW64 /c:/Test123
$ git init
Initialized empty Git repository in C:/Test123/.git/

PS@DESKTOP-6K1ULK0 MINGW64 /c:/Test123 (master)
$ git remote add origin "https://github.com/pramodsunagar/Demo123.git"

PS@DESKTOP-6K1ULK0 MINGW64 /c:/Test123 (master)
$ |
```

Git Commands

- git pull: The **git pull origin main** command syncs all the files from remote repo to local repo.

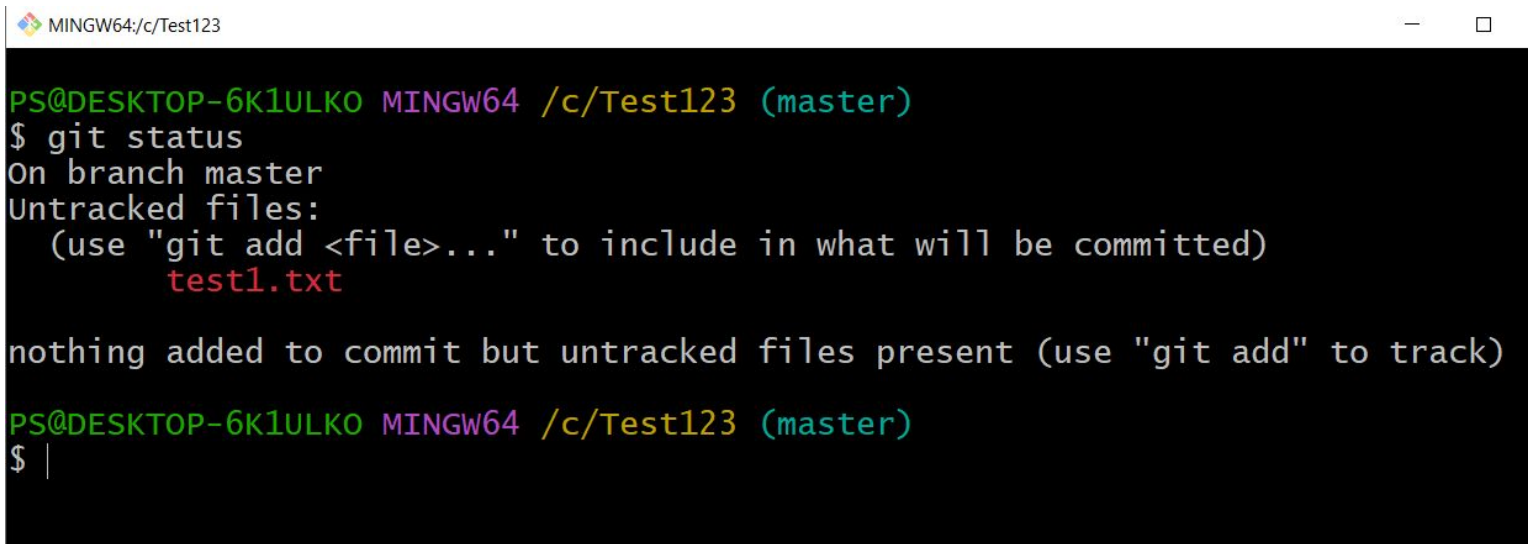
MINGW64:/c/Test123

```
PS@DESKTOP-6K1ULK0 MINGW64 /c/Test123 (master)
$ git pull origin main
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), 596 bytes | 3.00 KiB/s, done.
From https://github.com/pramodsunagar/Demo123
* branch          main          -> FETCH_HEAD
* [new branch]     main          -> origin/main

PS@DESKTOP-6K1ULK0 MINGW64 /c/Test123 (master)
$ |
```

Git Commands

- git status: The git status command lists all the modified files which are ready to be added to the local repository.

A terminal window with a black background and white text. The title bar at the top shows a multi-colored icon followed by the text 'MINGW64:/c/Test123'. The terminal content shows a user at a prompt '\$' running the command 'git status'. The output indicates the current branch is 'master' and lists 'test1.txt' as an untracked file. It also provides instructions on how to add files and track them. The prompt '\$' is followed by a vertical bar '|' indicating the cursor is ready for the next command.

```
MINGW64:/c/Test123  
PS@DESKTOP-6K1ULK0 MINGW64 /c/Test123 (master)  
$ git status  
On branch master  
Untracked files:  
  (use "git add <file>..." to include in what will be committed)  
    test1.txt  
  
nothing added to commit but untracked files present (use "git add" to track)  
PS@DESKTOP-6K1ULK0 MINGW64 /c/Test123 (master)  
$ |
```

Git Commands

- git add: This command updates the index using the current content found in the working tree and then prepares the content in the staging area for the next commit.

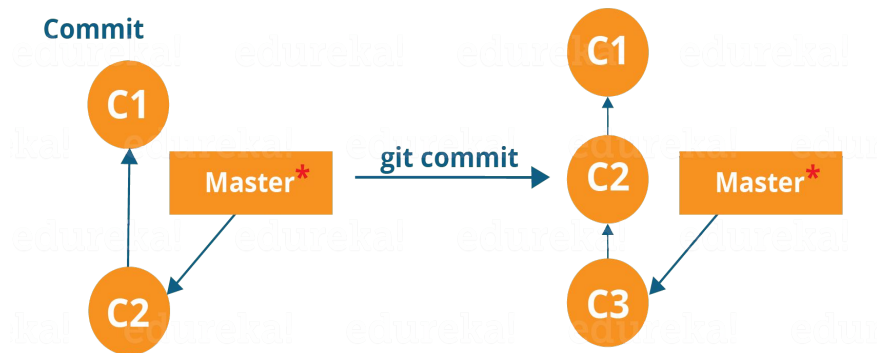
```
MINGW64:/c/Test123
PS@DESKTOP-6K1ULK0 MINGW64 /c/Test123 (master)
$ git add test1.txt

PS@DESKTOP-6K1ULK0 MINGW64 /c/Test123 (master)
$ git add -A

PS@DESKTOP-6K1ULK0 MINGW64 /c/Test123 (master)
$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   test1.txt
        new file:   test2.txt
        new file:   test3.txt
```

Git Commands

- `git commit`: This will commit the staged snapshot.
- `git commit -a -m "Message"`



```
MINGW64:/c/Test123
PS@DESKTOP-6K1ULK0 MINGW64 /c/Test123 (master)
$ git commit -a -m "First Commit"
[master 234143b] First Commit
3 files changed, 3 insertions(+)
create mode 100644 test1.txt
create mode 100644 test2.txt
create mode 100644 test3.txt
PS@DESKTOP-6K1ULK0 MINGW64 /c/Test123 (master)
$ |
```

Git Commands

- git log

MINGW64:/c/Test123

```
PS@DESKTOP-6K1ULK0 MINGW64 /c/Test123 (master)
$ git log
commit 234143bcbe7fb62f9f5339abf465515f781aef59 (HEAD -> master)
Author: pramodsunagar <pramod.sunagar@gmail.com>
Date: Mon Aug 22 11:46:05 2022 +0530

    First Commit

commit 138a459da3e1c3d0ae38cf80d1243b787309babe (origin/main)
Author: Pramod Sunagar <71660046+pramodsunagar@users.noreply.github.com>
Date: Mon Aug 22 11:05:22 2022 +0530

    Initial commit
```


Git Commands

- To create a new branch
- `git branch firstbranch`
- To move to another branch
- `git checkout firstbranch`

 MINGW64:/c/Test123

```
PS@DESKTOP-6K1ULK0 MINGW64 /c/Test123 (master)
$ git branch firstbranch
```

```
PS@DESKTOP-6K1ULK0 MINGW64 /c/Test123 (master)
$ git checkout firstbranch
Switched to branch 'firstbranch'
```

```
PS@DESKTOP-6K1ULK0 MINGW64 /c/Test123 (firstbranch)
$ |
```

Git Commands

- To show merging
- Create a file in new branch and commit

```
MINGW64:/c/Test123
nothing added to commit but untracked files present (use "git add" to track)

PS@DESKTOP-6K1ULK0 MINGW64 /c/Test123 (firstbranch)
$ git add new1.txt

PS@DESKTOP-6K1ULK0 MINGW64 /c/Test123 (firstbranch)
$ git status
On branch firstbranch
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   new1.txt

PS@DESKTOP-6K1ULK0 MINGW64 /c/Test123 (firstbranch)
$ git commit -m "Adding the new file to the first branch"
[firstbranch f376282] Adding the new file to the first branch
1 file changed, 1 insertion(+)
create mode 100644 new1.txt
```

Git Commands

- To show merging
- Go to main branch
- git merge firstbranch

```
MINGW64:/c/Test123
PS@DESKTOP-6K1ULK0 MINGW64 /c/Test123 (firstbranch)
$ git checkout main
Switched to a new branch 'main'
branch 'main' set up to track 'origin/main'.

PS@DESKTOP-6K1ULK0 MINGW64 /c/Test123 (main)
$ git merge firstbranch
Updating 138a459..f376282
Fast-forward
 new1.txt | 1 +
 test1.txt | 1 +
 test2.txt | 1 +
 test3.txt | 1 +
 4 files changed, 4 insertions(+)
 create mode 100644 new1.txt
 create mode 100644 test1.txt
 create mode 100644 test2.txt
 create mode 100644 test3.txt

PS@DESKTOP-6K1ULK0 MINGW64 /c/Test123 (main)
$ |
```

Pushing the files to Github repository

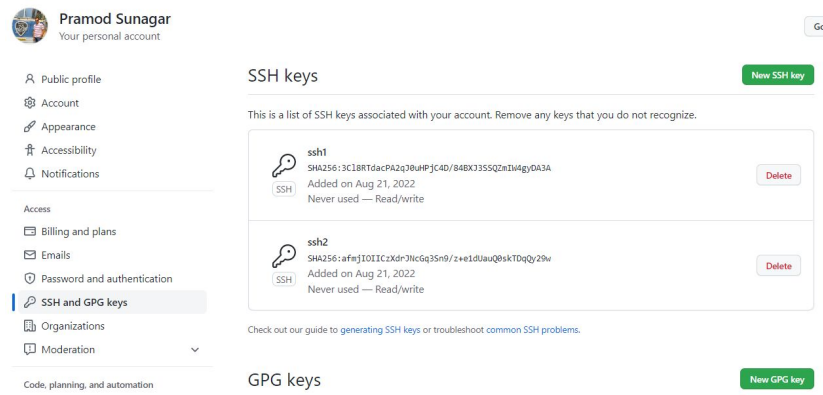
- To sync the files of local repo to remote repo, create a ssh key.
- ssh-keygen

 MINGW64:/c/Test123

```
PS@DESKTOP-6K1ULKU MINGW64 /c/Test123 (main)
$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/c/Users/PS/.ssh/id_rsa):
/c/Users/PS/.ssh/id_rsa already exists.
Overwrite (y/n)? y
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /c/Users/PS/.ssh/id_rsa
Your public key has been saved in /c/Users/PS/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:3THGnoY2luXvRR0byIRKHa+lExpktUEQHuD+KxKOJJo PS@DESKTOP-6K1ULKU
The key's randomart image is:
+---[RSA 3072]-----+
|
|      .o=
|     =..+*
|    .+.o*= o
|   .ooo=+  =
|  + SoB+* ..o
| E . .o o.. .
| . o . . . .
| . o . . . .
| . o . . . .
|  .o.
+----[SHA256]-----+
```

Pushing the files to Github repository

- To add the ssh key go to github account.
- go to setting
- go to SSH and GPG Keys
- Click on new ssh key
- Add title and copy the ssh keys from the command prompt to the ssh key and save.



```
PS@DESKTOP-6K1ULK0 MINGW64 /c/Test123 (main)
$ cat /c/Users/PS/.ssh/id_rsa.pub
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQGDQDN3EpF40vdmivhk5ZvUT7y21g4Fjv4wMF8a1z/wpX
k5czqqARicCPQV3Tp4IF7NomwKcPG1gkKDhde+H6TbvYW4KgQ1ZwbLDQBBqyuYkMdUcMOFHCId9ko/D
79Vsif9ZeygQ0A71ONJ+OXX+/LkbxqH80ejuBZYSg8Pok9myPTbwjDa0uw/aTTRUC1MEqhHdi1wdTqm
hB0sPTDB+FTU8owakTY7GAGGCJzcIftBbZQh+PiPuk90jUZOVKTgme5WdpwrFm70a0zb+EfMcmLmH2N
Hanv5cenm4JBM1gRYIwjPSmjF/ju4FStKnxrZsaHk2s2LegpwZGbhJ+x85Zn0kgYsprWqvXqoCzdB3s
jeXst0NajbHJco8ozOyZ3QuSSBzdR64iY710qWND9r/rz6SoslEYvABY+1XT+JE5Hu3ZPWUf62N6JX
64vm2zV083ts6Ef4yHyJAr12BP6VzXtlnjlpXFkUe1M3w2xwhk0sic0rke7SuMsbvmh5F3Mxe1Cr0=
PS@DESKTOP-6K1ULK0
```

Pushing the files to Github repository

- To authenticate the SSH Key type
- `ssh -T git@github.com`

A terminal window titled 'MINGW64:/c/Test123' with standard window controls. The terminal shows a user at a PS@DESKTOP-6K1ULK0 prompt in a MINGW64 environment at /c/Test123 (main). They run 'ssh -T git@github.com'. The output is 'Hi pramodsunagar! You've successfully authenticated, but GitHub does not provide shell access.' The prompt returns to the user, who has entered a dollar sign '\$' and a vertical bar '|' on the next line.


```
MINGW64:/c/Test123
PS@DESKTOP-6K1ULK0 MINGW64 /c/Test123 (main)
$ ssh -T git@github.com
Hi pramodsunagar! You've successfully authenticated, but GitHub does not provide shell access.
PS@DESKTOP-6K1ULK0 MINGW64 /c/Test123 (main)
$ |
```




Pushing the files to Github repository

- Push all the files
- `git push origin firstbranch`
- `git push origin main`


MINGW64/c/Test123






```
PS@DESKTOP-6K1ULK0 MINGW64 /c/Test123 (main)
$ git push origin firstbranch
Enumerating objects: 9, done.
Counting objects: 100% (9/9), done.
Delta compression using up to 4 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (8/8), 660 bytes | 82.00 KiB/s, done.
Total 8 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), done.
remote:
remote: Create a pull request for 'firstbranch' on GitHub by visiting:
remote:   https://github.com/pramodsunagar/Demo123/pull/new/firstbranch
remote:
To https://github.com/pramodsunagar/Demo123.git
 * [new branch]      firstbranch -> firstbranch
```

 firstbranch had recent pushes less than a minute ago [Compare & pull request](#)

 firstbranch  2 branches  0 tags [Go to file](#) [Add file](#) [Code](#)

This branch is 2 commits ahead of main. [Contribute](#)

 pramodsunagar Adding the new file to the first branch f376282 18 minutes ago 3 commits

 README.md	Initial commit	1 hour ago
 new1.txt	Adding the new file to the first branch	18 minutes ago
 test1.txt	First Commit	26 minutes ago
 test2.txt	First Commit	26 minutes ago
 test3.txt	First Commit	26 minutes ago

Thank You