-

Subject Name: **Source Code Management**

Subject Code: **CS181**

Cluster: **Beta**

Department: **DCSE**

CHITKARA UNIVERSITY

Submitted By:
Aarush Khanna
2110990014
G01-A
Cluster-Beta

Submitted To:
DR. MONIT
KAPOOR

Experiment No. 01

Aim: Setting up of Git Client

Theory:

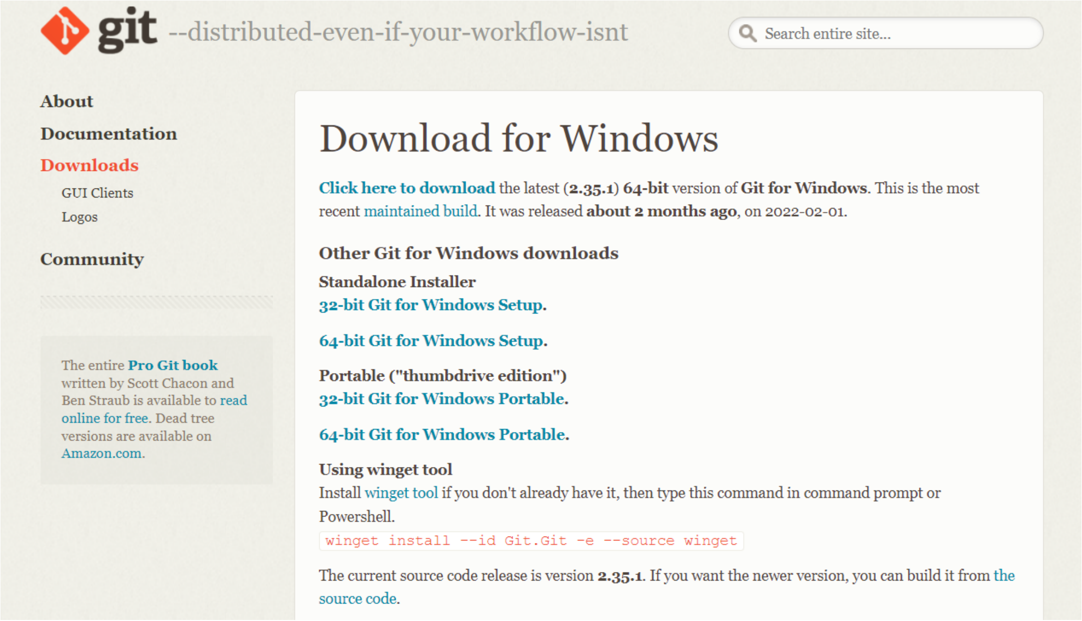
GIT –> It is basically used for pushing and pulling of code. We can use git and git-hub parallelly to work with multiple members or individually. We can make , edit , recreate ,copy or download any code on git hub using git.

What is GIT ? –> It's a Version Control System(VCS) -> It is a software or we can say a server by which we are able to track all the previous changes in the code.

Advantages of GIT –>

Procedure: We can install Git on Windows, using the most official build which is available for download on the GIT's official website or by just typing (scmgit) on any search engine. We can go on https://git-scm.com/download/win and can select the platform and bit-version to download. And after clicking on your desired bit-version or ios it will start downloading automatically.

Snapshots of download:



The entire Pro Git book written by Scott Chacon and Ben Straub is available to read online for free. Dead tree versions are available on Amazon.com.

Download for Windows

Click here to download the latest (2.35.1) 64-bit version of Git for Windows. This is the most recent maintained build. It was released about 2 months ago, on 2022-02-01.

Other Git for Windows downloads

Standalone Installer

32-bit Git for Windows Setup.

64-bit Git for Windows Setup.

Portable ("thumbdrive edition")

32-bit Git for Windows Portable.

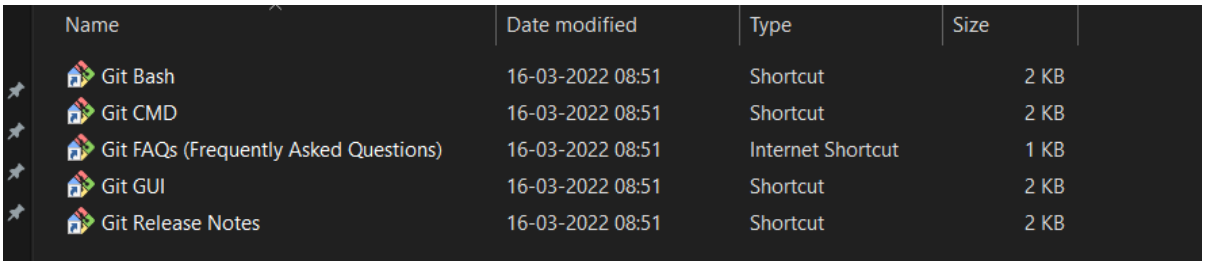
64-bit Git for Windows Portable.

Using winget tool

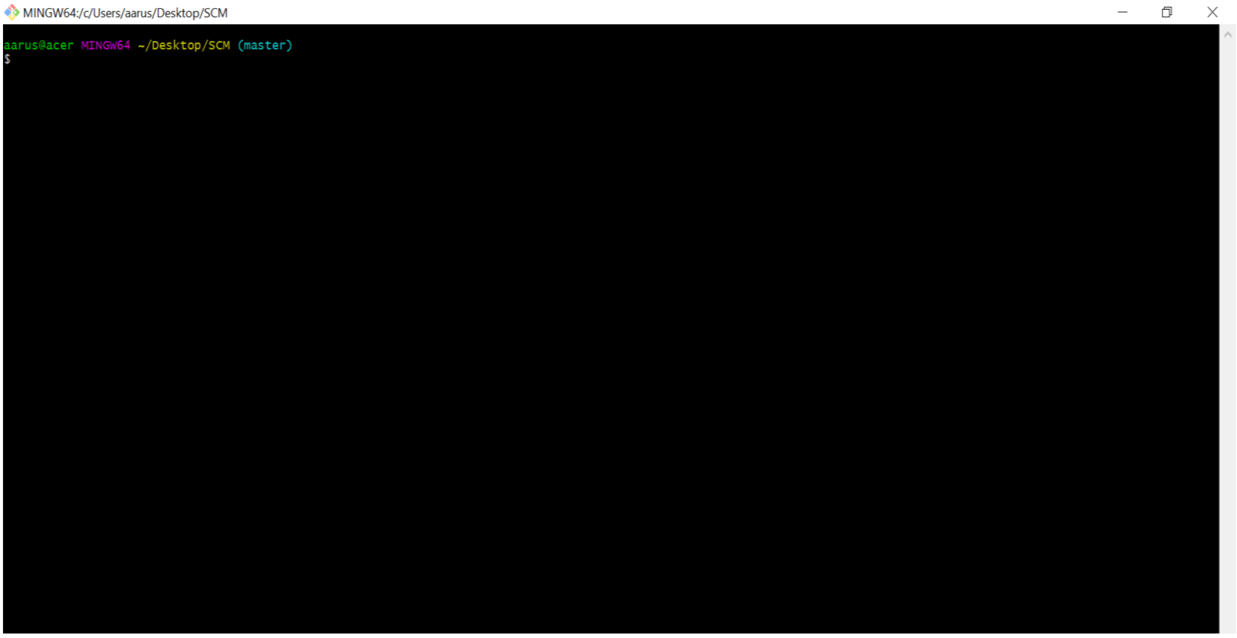
Install winget tool if you don't already have it, then type this command in command prompt or Powershell.

```
winget install --id Git.Git -e --source winget
```

The current source code release is version 2.35.1. If you want the newer version, you can build it from the source code.



Name	Date modified	Type	Size
Git Bash	16-03-2022 08:51	Shortcut	2 KB
Git CMD	16-03-2022 08:51	Shortcut	2 KB
Git FAQs (Frequently Asked Questions)	16-03-2022 08:51	Internet Shortcut	1 KB
Git GUI	16-03-2022 08:51	Shortcut	2 KB
Git Release Notes	16-03-2022 08:51	Shortcut	2 KB



```
aarus@acer MINGW64 ~/Desktop/SCM (master)
$
```

Experiment No. 02

Aim: Setting up GitHub Account

Theory:

What is GitHub -> GitHub is a website and cloud-based service (client) that helps an individual or a developers to store and manage their code. We can also track as well as control changes to our or public code.

Advantages of GitHub -> GitHub's has a user-friendly interface and is easy to use .We can connect the git-hub and git but using some commands shown below in figure 001. Without GitHub we cannot use Git because it generally requires a host and if we are working for a project we need to share it will our team members, which can only be done by making a repository . Additionally , anyone can sign up and host a public code repository for free, which makes GitHub especially popular with open-source projects.

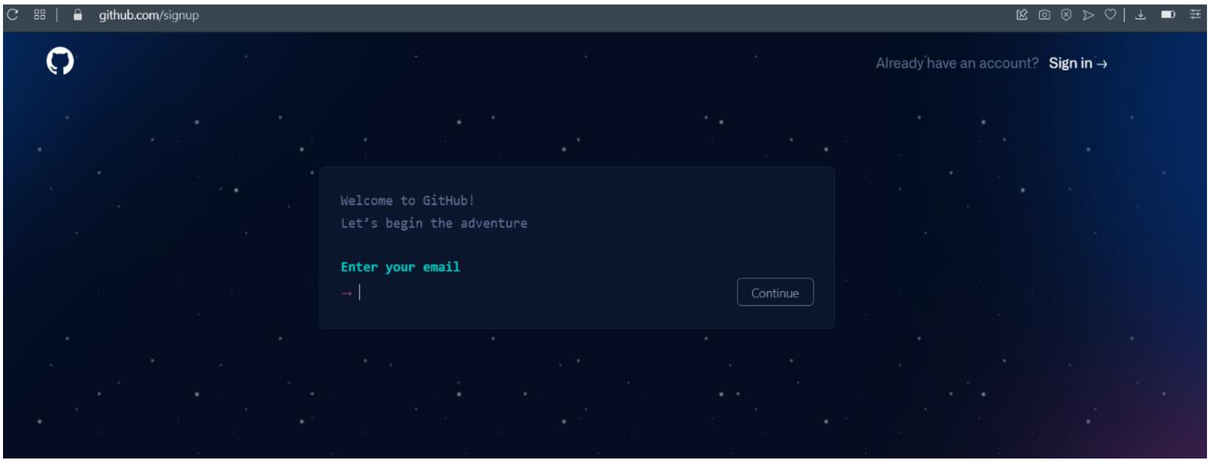
Procedure:-

Step1 :-

Google (any search engine)
Search for git-hub or (https://github.com/signup).

Step2 :-

Snapshots –

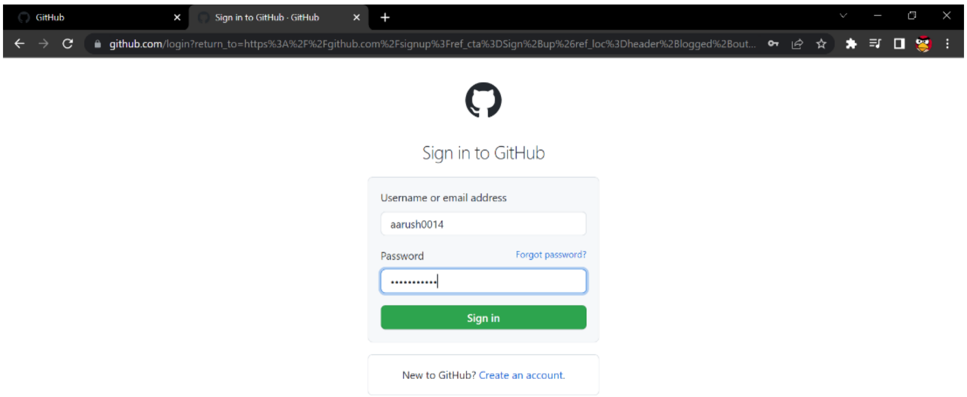


Welcome to GitHub!
Let's begin the adventure

Enter your email
→ |

After visiting the link this type of interface will appear, if you already have account you can sign in and if not you can create.

Sign in into GitHub :-

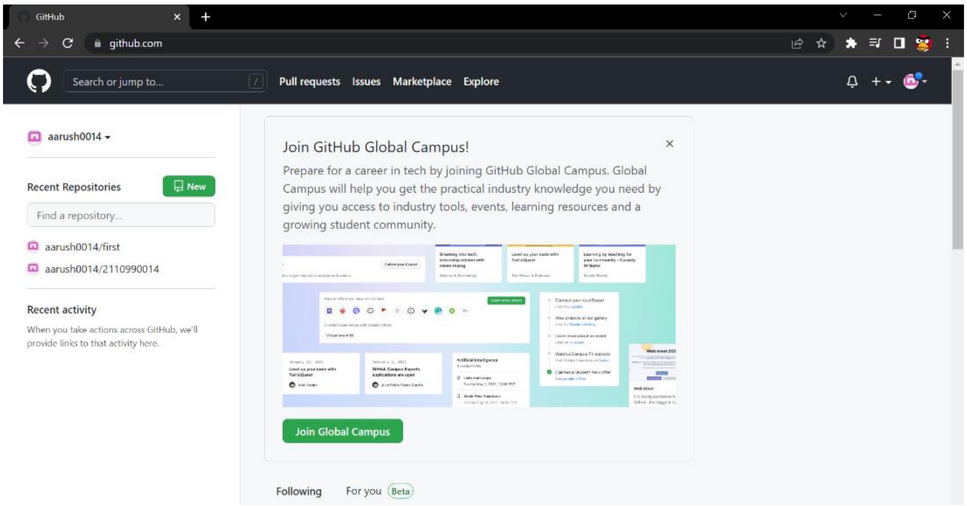

GitHub
Sign in to GitHub · GitHub
github.com/login?return_to=https%3A%2F%2Fgithub.com%2Fsignup%3Fref_cta%3DSign%2Bup%26ref_loc%3Dheader%2Blogged%2Bout...
Forgot password?
Sign in
New to GitHub? Create an account.

Sign in to GitHub

Username or email address: aarush0014

Password: ••••••••••

Interface of GitHub :-



aarush0014/first

aarush0014/2110990014

Recent activity

When you take actions across GitHub, we'll provide links to that activity here.

Join GitHub Global Campus!

Prepare for a career in tech by joining GitHub Global Campus. Global Campus will help you get the practical industry knowledge you need by giving you access to industry tools, events, learning resources and a growing student community.

To link GitHub account with Git bash –

<u>For username</u>:-

```
git config --global user.name “username in git-hub”
```

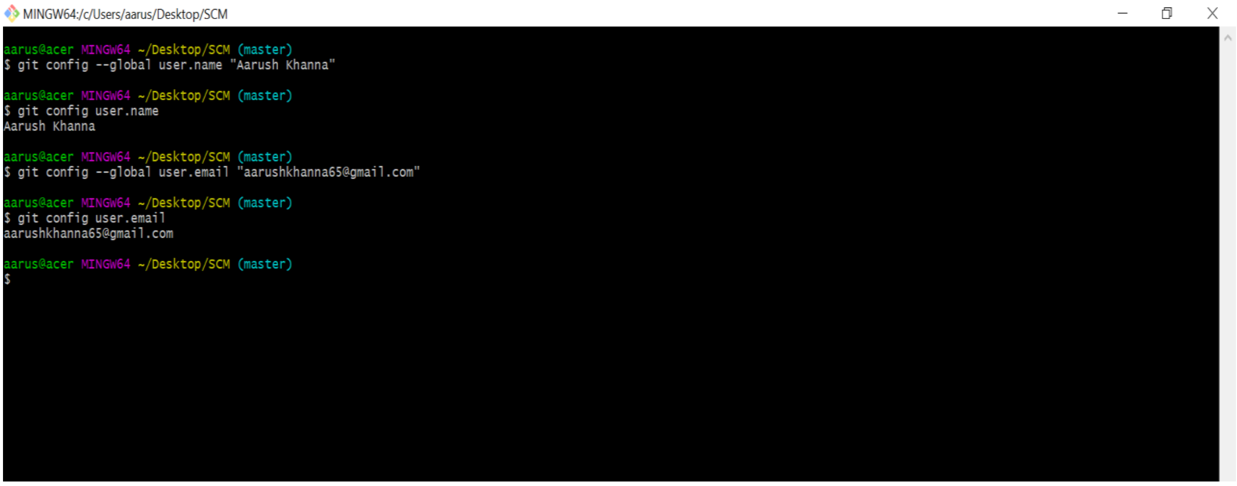
<u>For user email</u>:-

```
git config --global user.email “your email in git-hub”
```

<u>To verify</u>:-

```
git config user.name

git config user.email
```

Snapshot :-



```
aarus@acer MINGW64 ~/Desktop/SCM (master)
$ git config --global user.name "Aarush Khanna"

aarus@acer MINGW64 ~/Desktop/SCM (master)
$ git config user.name
Aarush Khanna

aarus@acer MINGW64 ~/Desktop/SCM (master)
$ git config --global user.email "aarushkhanna65@gmail.com"

aarus@acer MINGW64 ~/Desktop/SCM (master)
$ git config user.email
aarushkhanna65@gmail.com

aarus@acer MINGW64 ~/Desktop/SCM (master)
$
```

Experiment No. 03

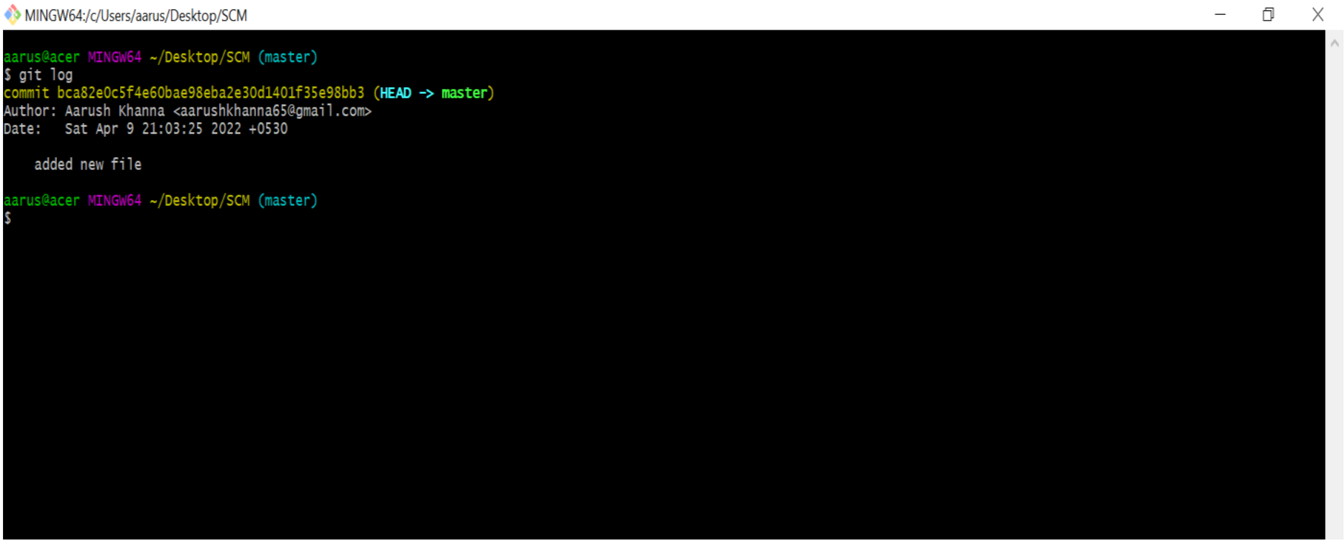
Aim: Program to Generate log

Theory:-

Logs -> Logs are nothing but the history which we can see in git by using the code git log. It contains all the past commits, insertions and deletions in it which we can see any time.

Why logs -> Logs helps to check that what were the changes in the code or any other file and by whom. It also contains the number of insertions and deletions including at which time it was changed.

Snapshots –



```
aarus@acer MINGW64 ~/Desktop/SCM (master)
$ git log
commit bca82e0c5f4e60bae98eba2e30d1401f35e98bb3 (HEAD -> master)
Author: Aarush Khanna <aarushkhanna65@gmail.com>
Date:   Sat Apr 9 21:03:25 2022 +0530

    added new file

aarus@acer MINGW64 ~/Desktop/SCM (master)
$
```

Experiment No. 04

Aim: Create and visualize branches

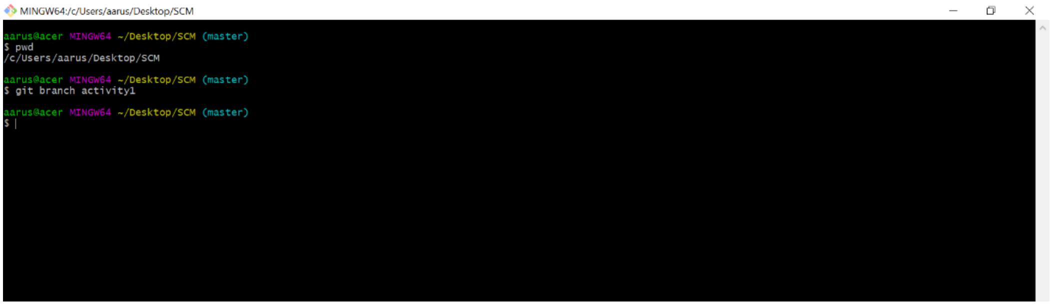
Create branches :-

The main branch in git is called as master branch. But we can make branches out of this main master branch. All the files present in master can be shown in branch but the file which are created in branch are not shown in master branch. We can also merge both the parent (master) and child (other branches).

Syntax:-

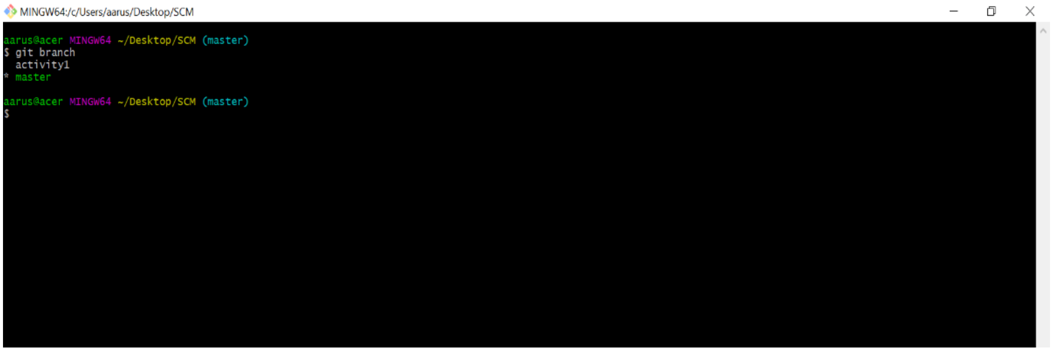
1. For creating a new branch.
   git branch name of branch

Snapshots –



```
aarus@acer MINGW64 ~/Desktop/SCM (master)
$ pwd
/c/Users/aarus/Desktop/SCM

aarus@acer MINGW64 ~/Desktop/SCM (master)
$ git branch activity1

aarus@acer MINGW64 ~/Desktop/SCM (master)
$ |
```

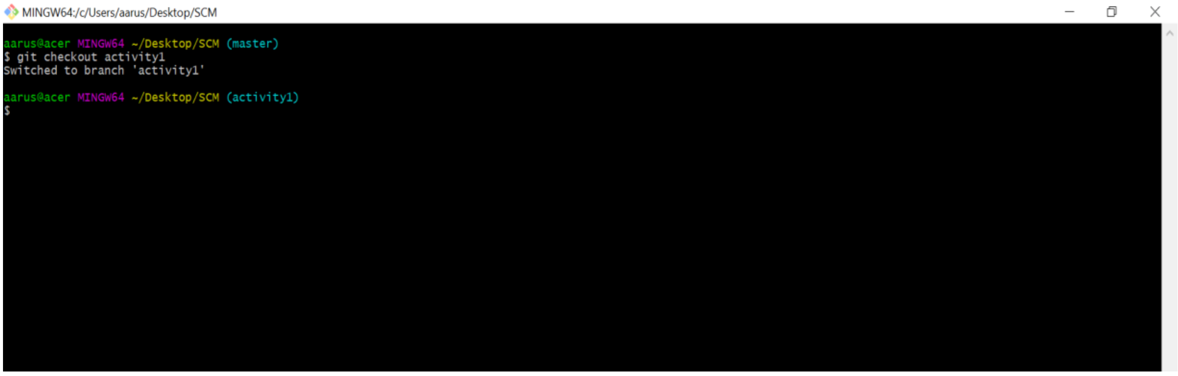
2. We can also check how many branches we have.
git branch



```
aarus@acer MINGW64 ~/Desktop/SCM (master)
$ git branch
  activity1
* master

aarus@acer MINGW64 ~/Desktop/SCM (master)
$
```

3. To change the present working branch.
git checkout name of branch.

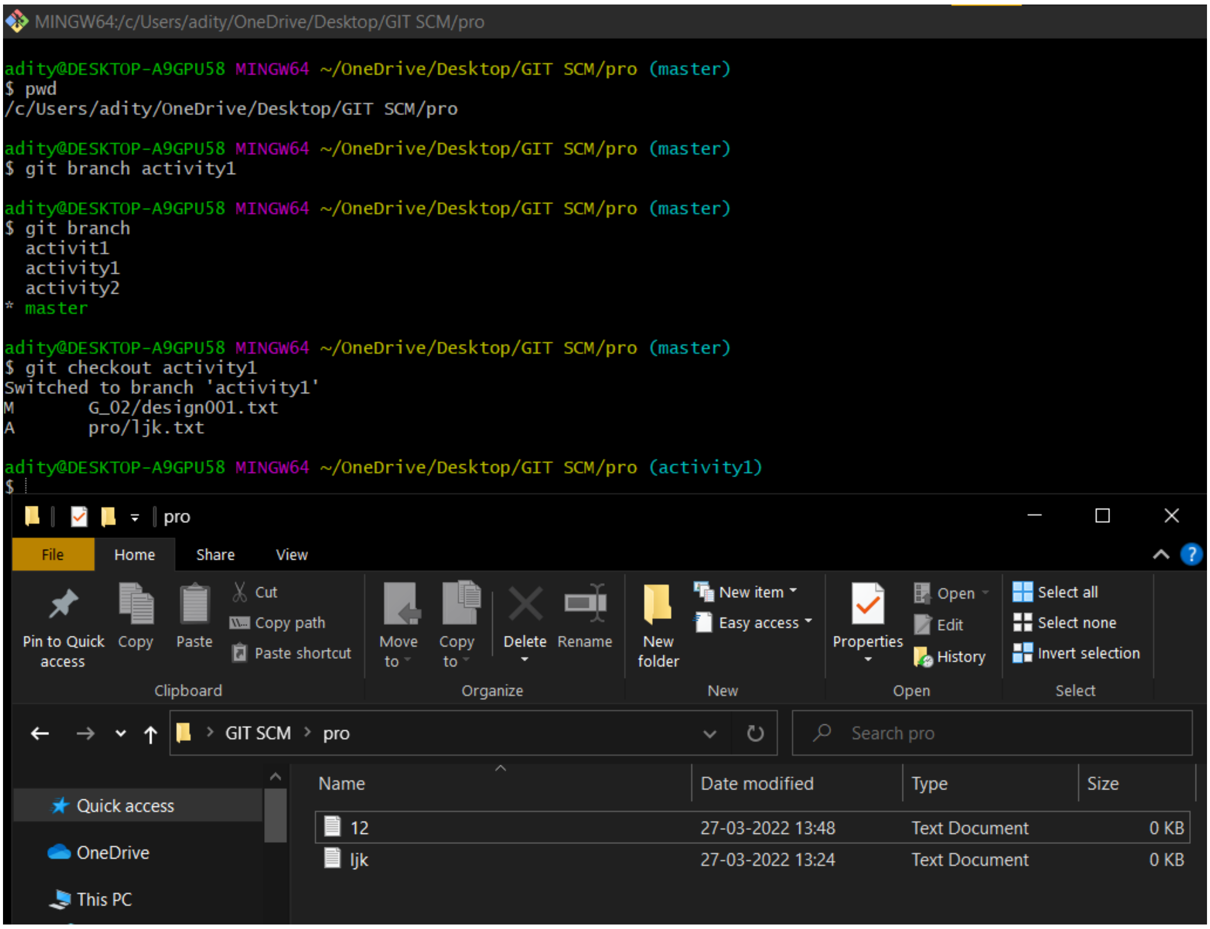
Snapshots –



```
aarus@acer MINGW64 ~/Desktop/SCM (master)
$ git checkout activity1
Switched to branch 'activity1'

aarus@acer MINGW64 ~/Desktop/SCM (activity1)
$
```

Visualizing branches :-

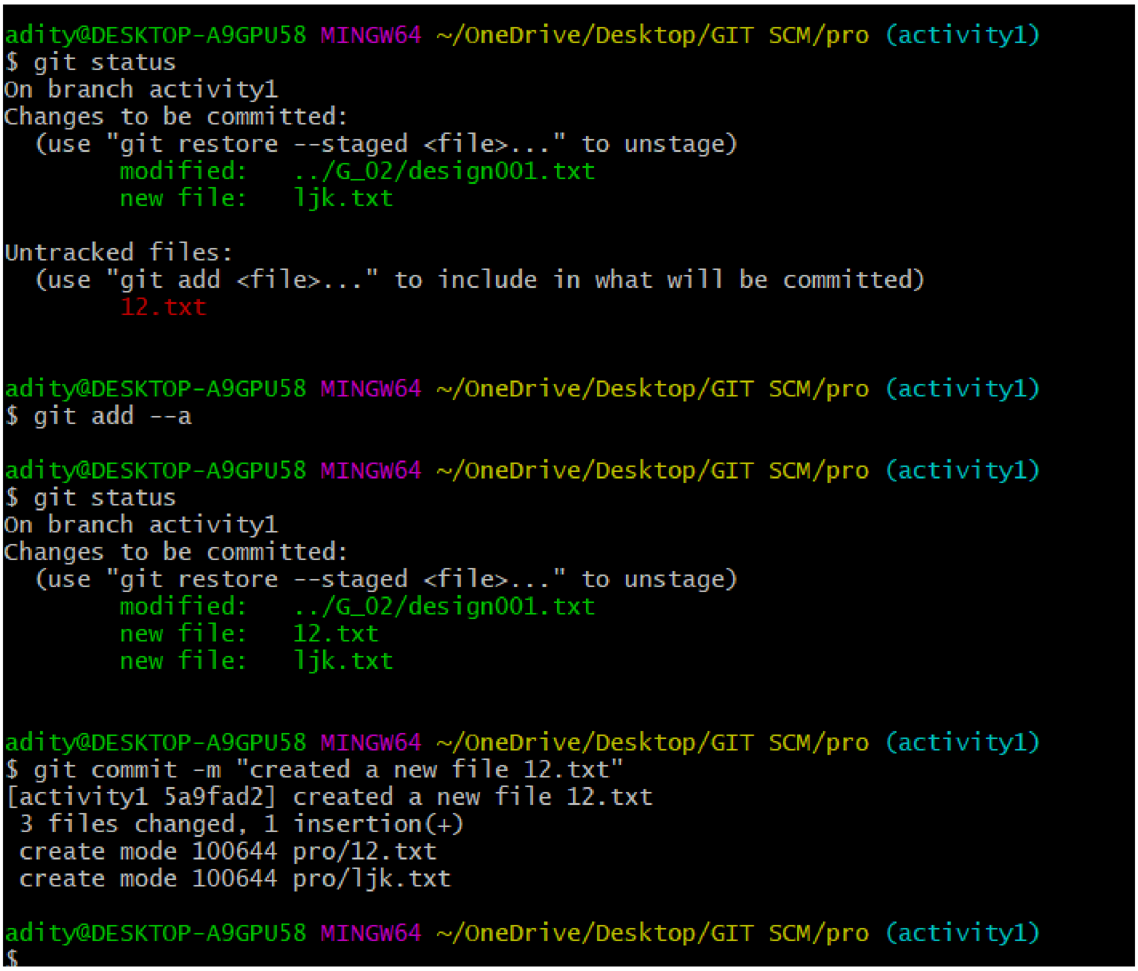
To visualize I have created a new file in a new branch activity 1 instead of master branch.



```
adity@DESKTOP-A9GPU58 MINGW64 ~/OneDrive/Desktop/GIT SCM/pro (master)
$ pwd
/c/Users/adity/OneDrive/Desktop/GIT SCM/pro

adity@DESKTOP-A9GPU58 MINGW64 ~/OneDrive/Desktop/GIT SCM/pro (master)
$ git branch activity1

adity@DESKTOP-A9GPU58 MINGW64 ~/OneDrive/Desktop/GIT SCM/pro (master)
$ git branch
  activit1
  activity1
  activity2
* master

adity@DESKTOP-A9GPU58 MINGW64 ~/OneDrive/Desktop/GIT SCM/pro (master)
$ git checkout activity1
Switched to branch 'activity1'
M       G_02/design001.txt
A       pro/ljk.txt

adity@DESKTOP-A9GPU58 MINGW64 ~/OneDrive/Desktop/GIT SCM/pro (activity1)
$
```

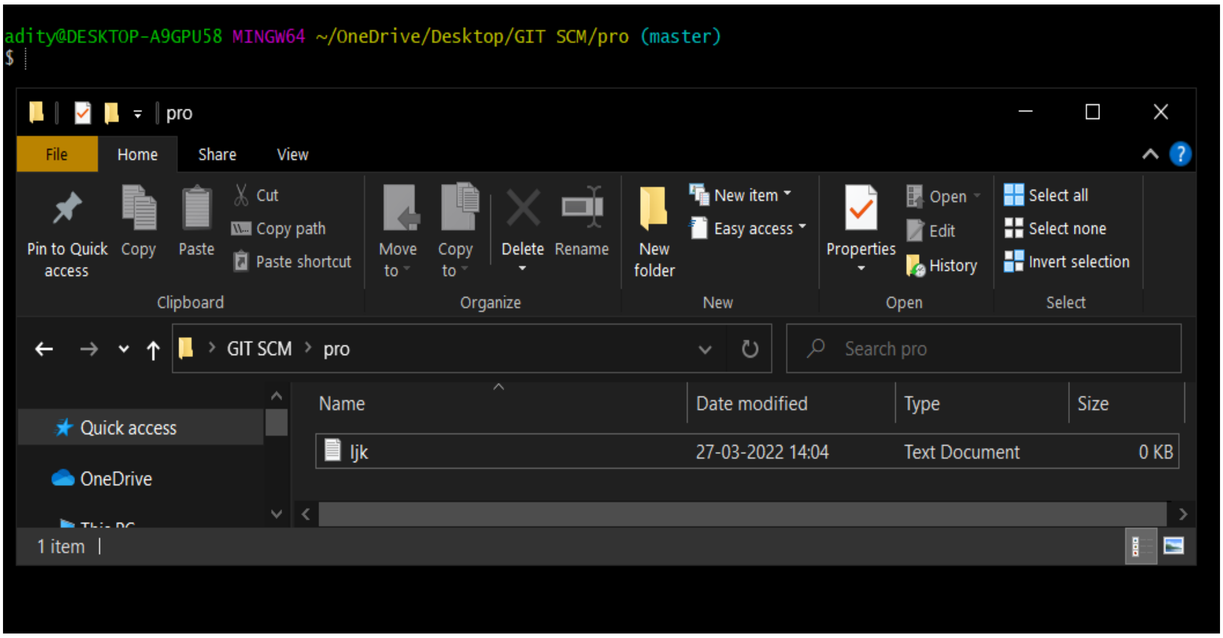
Name	Date modified	Type	Size
12	27-03-2022 13:48	Text Document	0 KB
ljk	27-03-2022 13:24	Text Document	0 KB

After this I have done the 3 step architecture which is tracking the file , send it to stagging area and finally we can role back to any previously saved version of this file.



```
adity@DESKTOP-A9GPU58 MINGW64 ~/OneDrive/Desktop/GIT SCM/pro (activity1)
$ git status
On branch activity1
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   ../G_02/design001.txt
        new file:   ljk.txt

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        12.txt


adity@DESKTOP-A9GPU58 MINGW64 ~/OneDrive/Desktop/GIT SCM/pro (activity1)
$ git add --a

adity@DESKTOP-A9GPU58 MINGW64 ~/OneDrive/Desktop/GIT SCM/pro (activity1)
$ git status
On branch activity1
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   ../G_02/design001.txt
        new file:   12.txt
        new file:   ljk.txt


adity@DESKTOP-A9GPU58 MINGW64 ~/OneDrive/Desktop/GIT SCM/pro (activity1)
$ git commit -m "created a new file 12.txt"
[activity1 5a9fad2] created a new file 12.txt
 3 files changed, 1 insertion(+)
 create mode 100644 pro/12.txt
 create mode 100644 pro/ljk.txt

adity@DESKTOP-A9GPU58 MINGW64 ~/OneDrive/Desktop/GIT SCM/pro (activity1)
$
```

After this we will change the branch from activity1 to master, but when I will switch to the master branch there will not be the same file in the master , it will not show the new file in the master branch.



```
adity@DESKTOP-A9GPU58 MINGW64 ~/OneDrive/Desktop/GIT SCM/pro (master)
$
```

Name	Date modified	Type	Size
ljk	27-03-2022 14:04	Text Document	0 KB

In this way we can create and change different branches . We can also merge the branches by using git merge command.

Aim: Git lifecycle description

Theory:

Stages in GIT Life Cycle -> Files in a Git project have various stages like Creation, Modification, Refactoring, and Deletion and so on. Irrespective of whether this project is tracked by Git or not, these phases are still prevalent. However, when a project is under Git version control system, they are present in three major Git states in addition to these basic ones. Here are the three Git states:

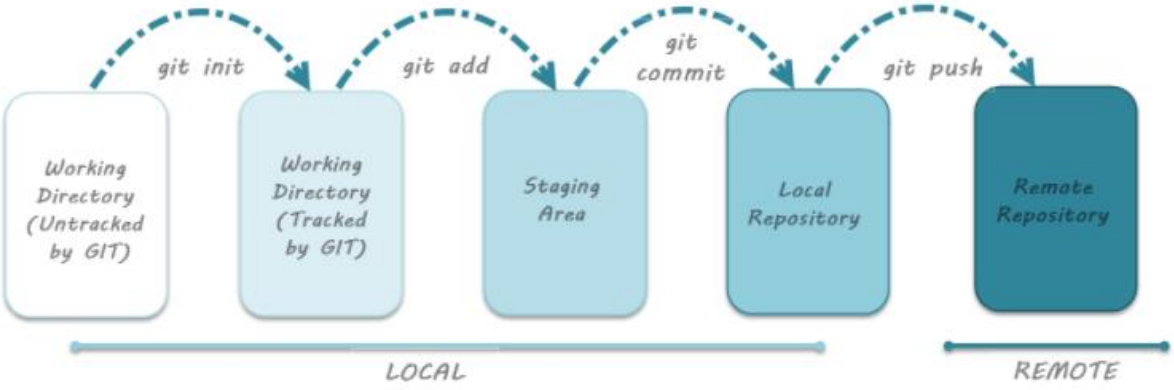
- Working directory
- Staging area
- Git directory

Working Directory ->
Consider a project residing in your local system. This project may or may not be tracked by Git. In either case, this project directory is called your Working directory.

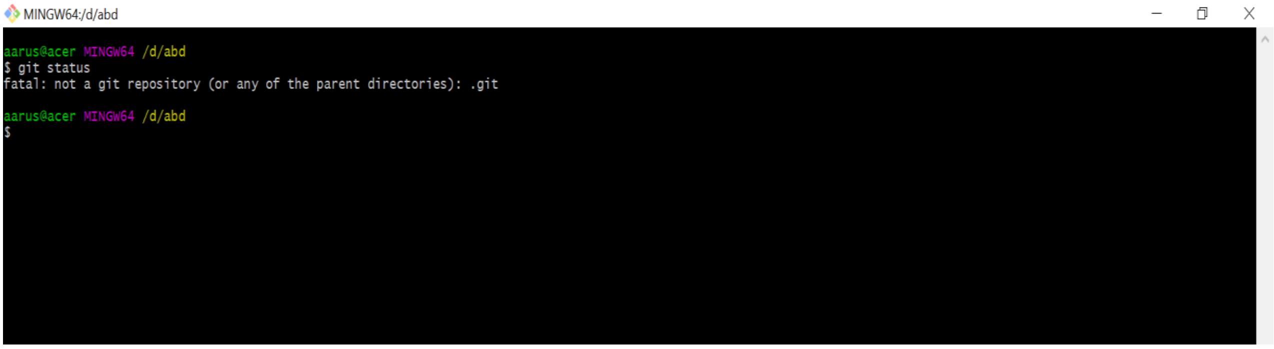
Staging Area ->
Staging area is the playground where you group, add and organize the files to be committed to Git for tracking their versions.

Git Directory ->
Now that the files to be committed are grouped and ready in the staging area, we can commit these files. So, we commit this group of files along with a commit message explaining what is the commit about. Apart from commit message, this step also records the author and time of the commit. Now, a snapshot of the files in the commit is recorded by Git. The information related to this commit is stored in the Git directory.

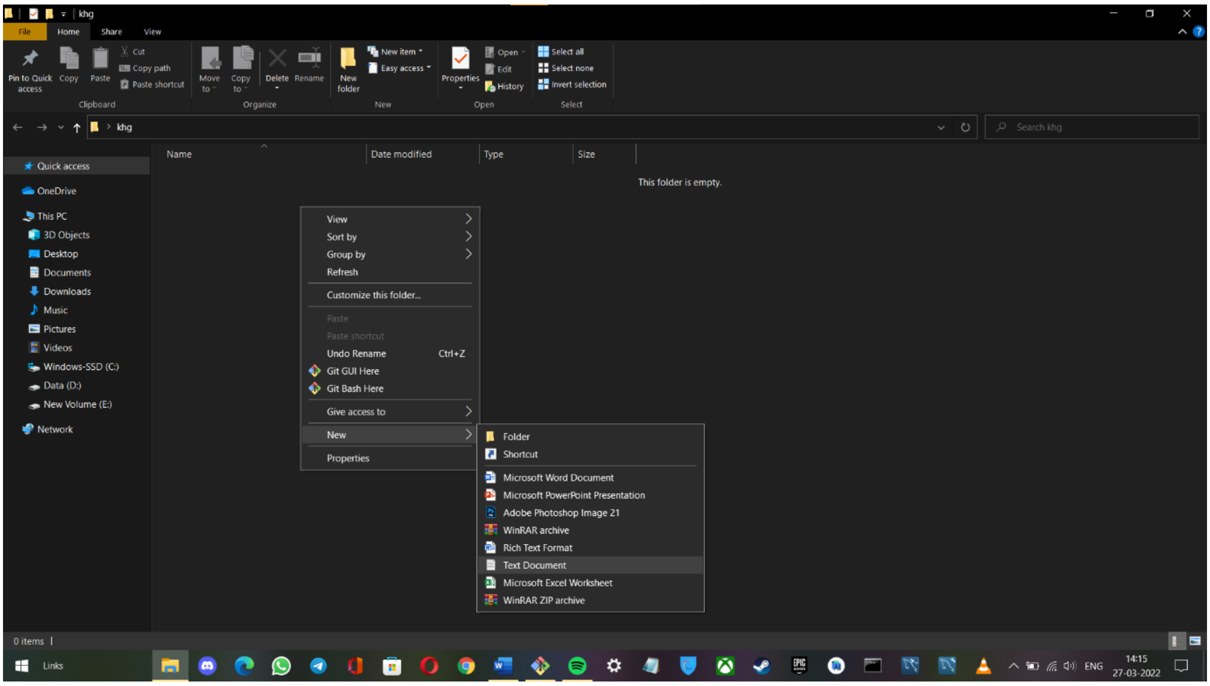
Remote Repository-> means mirror or clone of the local Git repository in GitHub. And pushing means uploading the commits from local Git repository to remote repository hosted in GitHub.



Snapshots –

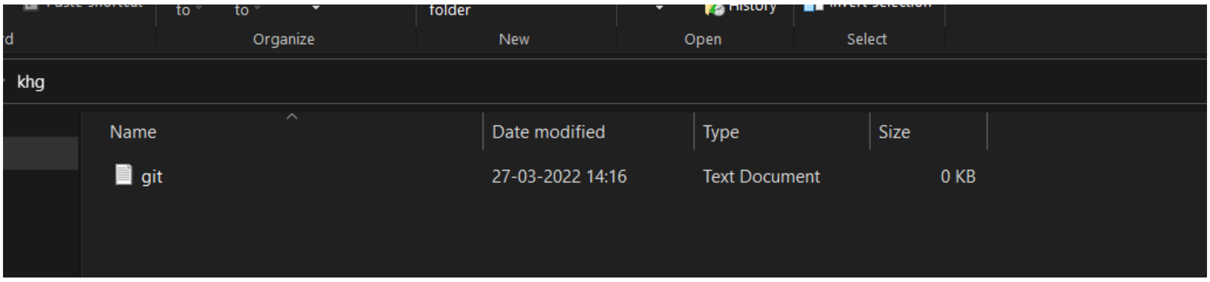


```
aarus@acer MINGW64 /d/abd
$ git status
fatal: not a git repository (or any of the parent directories): .git

aarus@acer MINGW64 /d/abd
$
```

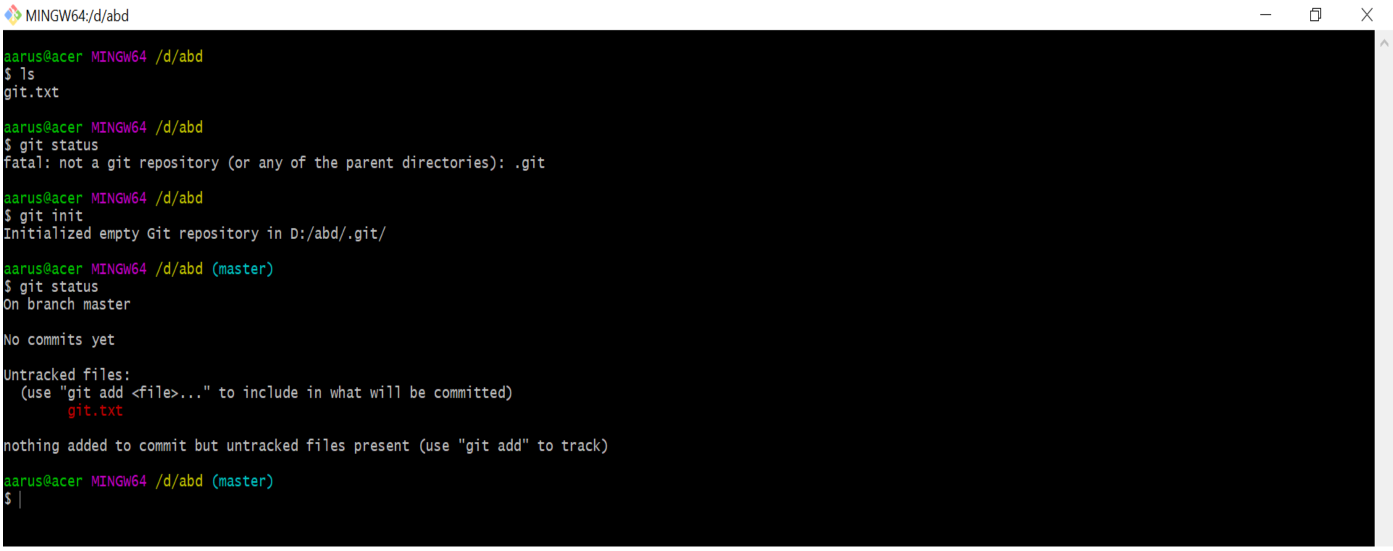


This folder is empty.

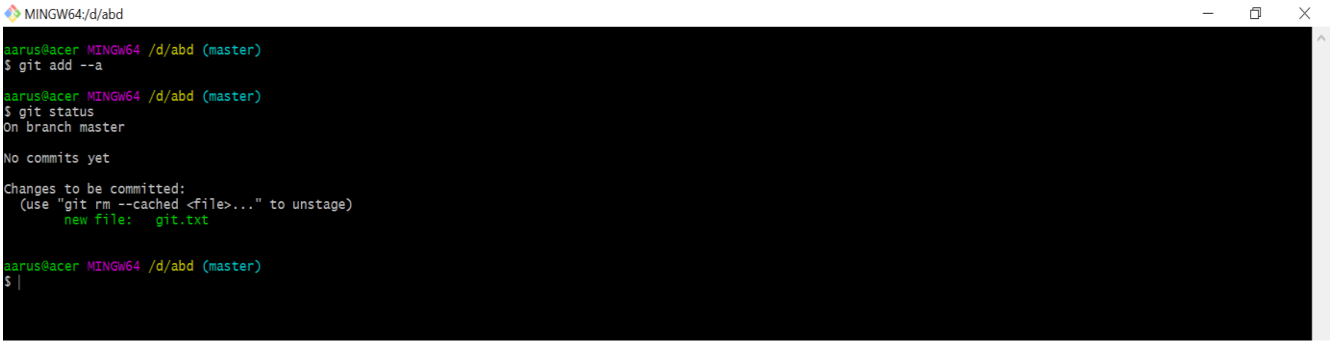
0 items



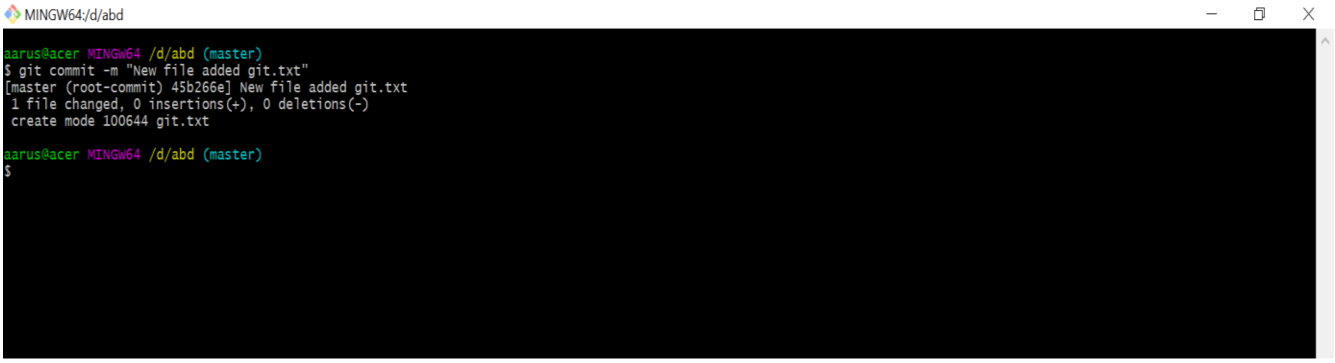
Name	Date modified	Type	Size
git	27-03-2022 14:16	Text Document	0 KB



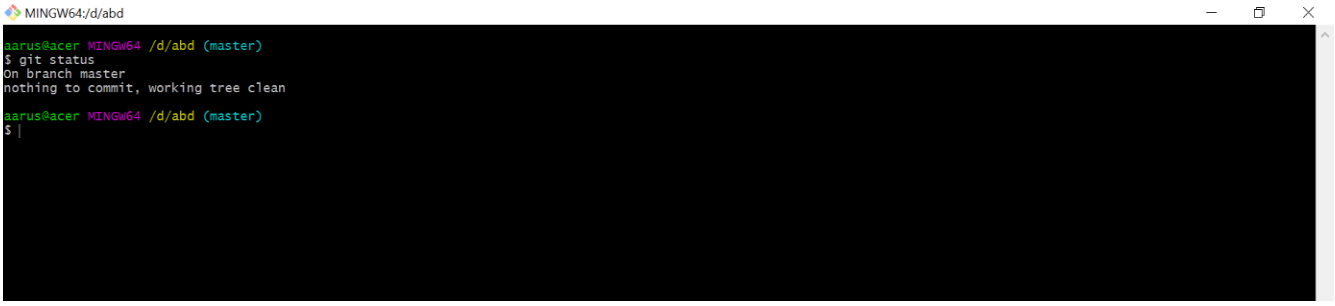
```
aarus@acer MINGW64 /d/abd
$ ls
git.txt

aarus@acer MINGW64 /d/abd
$ git status
fatal: not a git repository (or any of the parent directories): .git

aarus@acer MINGW64 /d/abd
$ git init
Initialized empty Git repository in D:/abd/.git/

aarus@acer MINGW64 /d/abd (master)
$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        git.txt

nothing added to commit but untracked files present (use "git add" to track)

aarus@acer MINGW64 /d/abd (master)
$ |
```



```
aarus@acer MINGW64 /d/abd (master)
$ git add --a

aarus@acer MINGW64 /d/abd (master)
$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   git.txt


aarus@acer MINGW64 /d/abd (master)
$ |
```



```
aarus@acer MINGW64 /d/abd (master)
$ git commit -m "New file added git.txt"
[master (root-commit) 45b266e] New file added git.txt
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 git.txt

aarus@acer MINGW64 /d/abd (master)
$
```



```
aarus@acer MINGW64 /d/abd (master)
$ git status
On branch master
nothing to commit, working tree clean

aarus@acer MINGW64 /d/abd (master)
$ |
```