SOURCE CODE MANAGEMENT FILE

Submitted by:
Muskan
Roll No.: 2310991992
Group: 22-B

Submitted To:
Mr.Sharad
Professor, CSE,
Chitkara University,
Punjab

Submission of: Task1.1
Subject Name: Source Code Management
Sub Code: 22CS003
Department: B.E. CSE

LIST OF PROGRAMS

S. No	Program Title	Page No.
Task 1.1(Week 4)		
1	Setting up of Git Client	3-5
2	Setting up GitHub Account	6-7
3	Generate logs	8-10
4	Create and visualize branches	11-13
5	Git life cycle description	14-15

CHITKARA
UNIVERSITY
PUNJAB

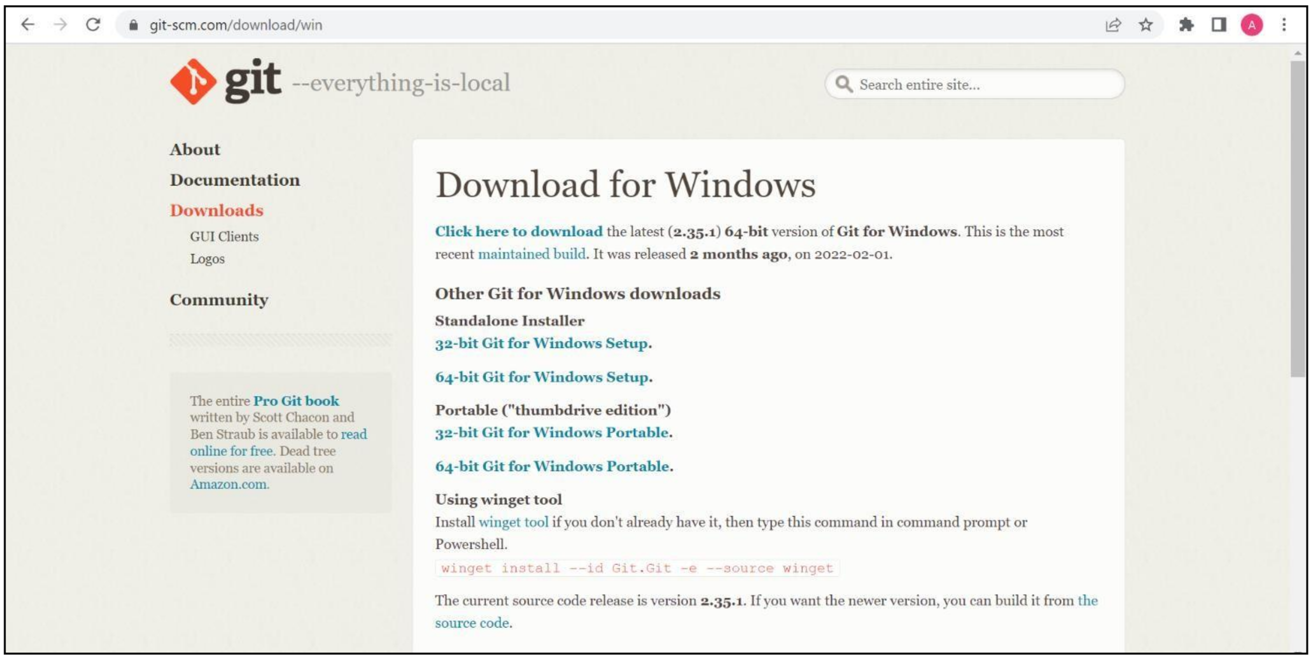
Experiment 1

Aim: Setting up of Git Client

Theory:
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. 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.

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 (scm git) 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:


git-scm.com/download/win
git --everything-is-local
Search entire site...
About
Documentation
Downloads
GUI Clients
Logos
Community
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 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.

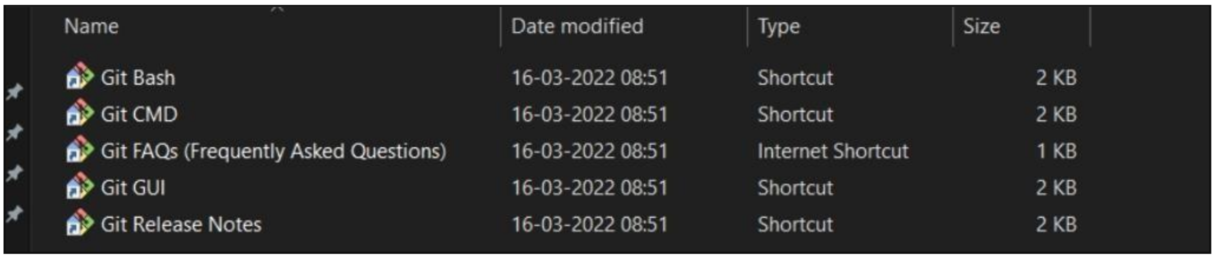
Portable ("thumbdrive edition")
32-bit Git for Windows Portable.
64-bit Git for Windows Portable.

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
```

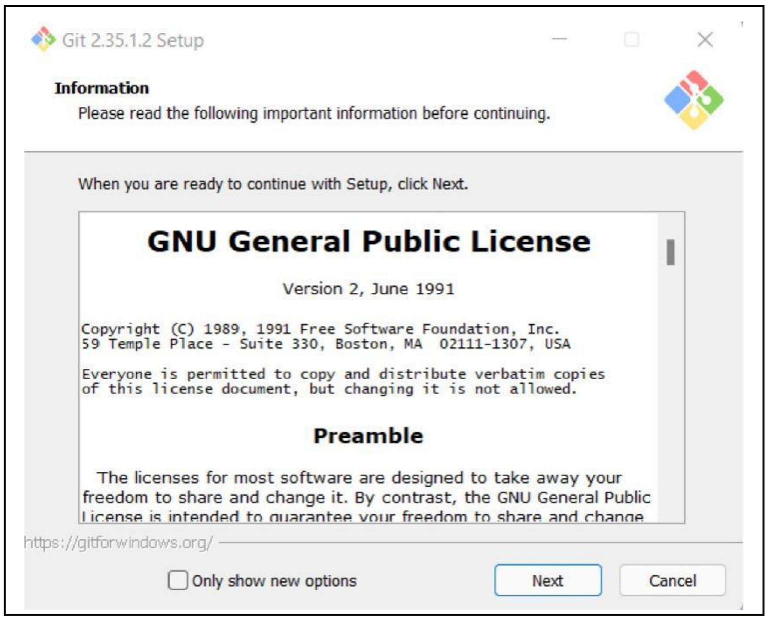
The current source code release is version 2.35.1. If you want the newer version, you can build it from the source code.

Opted for "64-bit Git for Windows Setup"



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

Git and its files in downloads



Git 2.35.1.2 Setup

Information

Please read the following important information before continuing.

When you are ready to continue with Setup, click Next.

GNU General Public License

Version 2, June 1991

```
Copyright (C) 1989, 1991 Free Software Foundation, Inc.
59 Temple Place - Suite 330, Boston, MA  02111-1307, USA

Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.
```

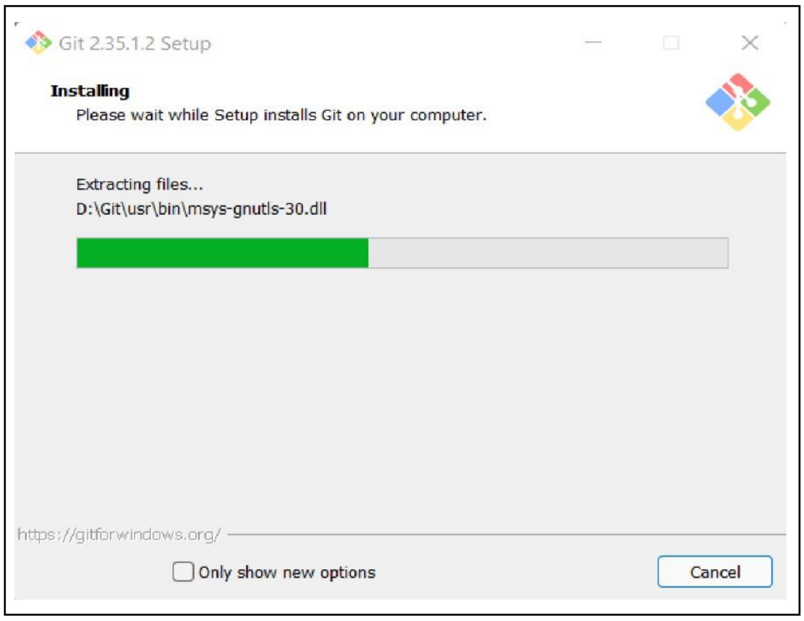
Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change

https://gitforwindows.org/

☐ Only show new options

Git Setup



Git 2.35.1.2 Setup

Installing

Please wait while Setup installs Git on your computer.

Extracting files...
D:\Git\usr\bin\msys-gnutls-30.dll

https://gitforwindows.org/

☐ Only show new options

Git Installation

```
hp@LAPTOP-HI19V135 MINGW64 ~
$
```

Git Bash launched

Experiment 2

Aim: Setting up GitHub Account

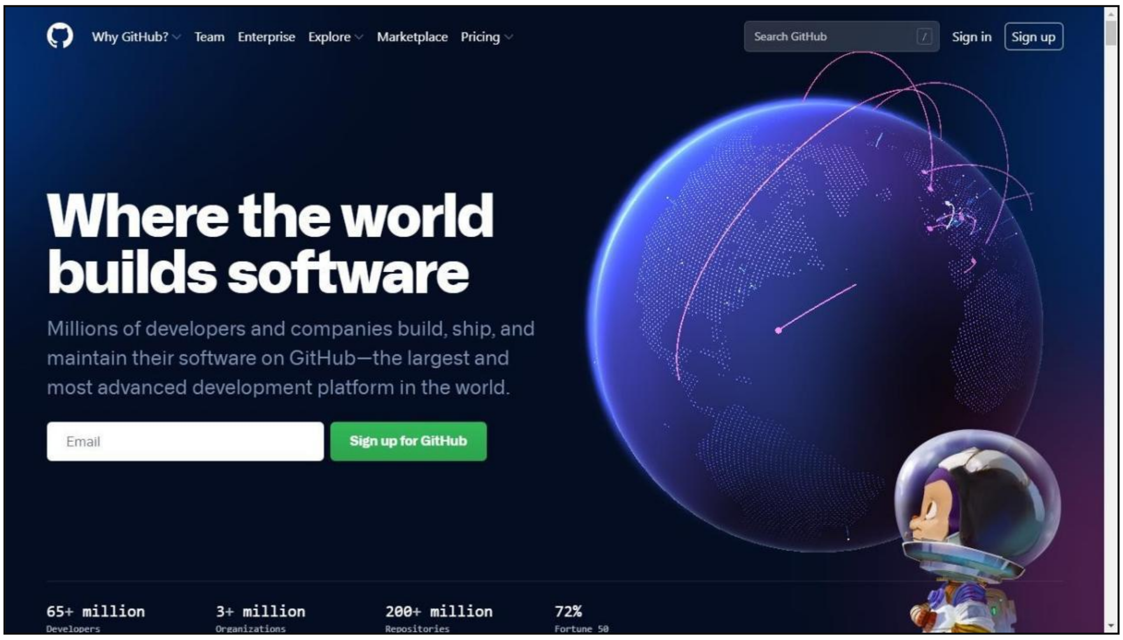
Theory:

GitHub: GitHub is a website and cloud-based service (client) that helps an individual or 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 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 01. 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:
To make an account on GitHub, we search for GitHub on our browser or visit https://github.com/signup. Then, we will enter our mail ID and create a username and password for a GitHub account.

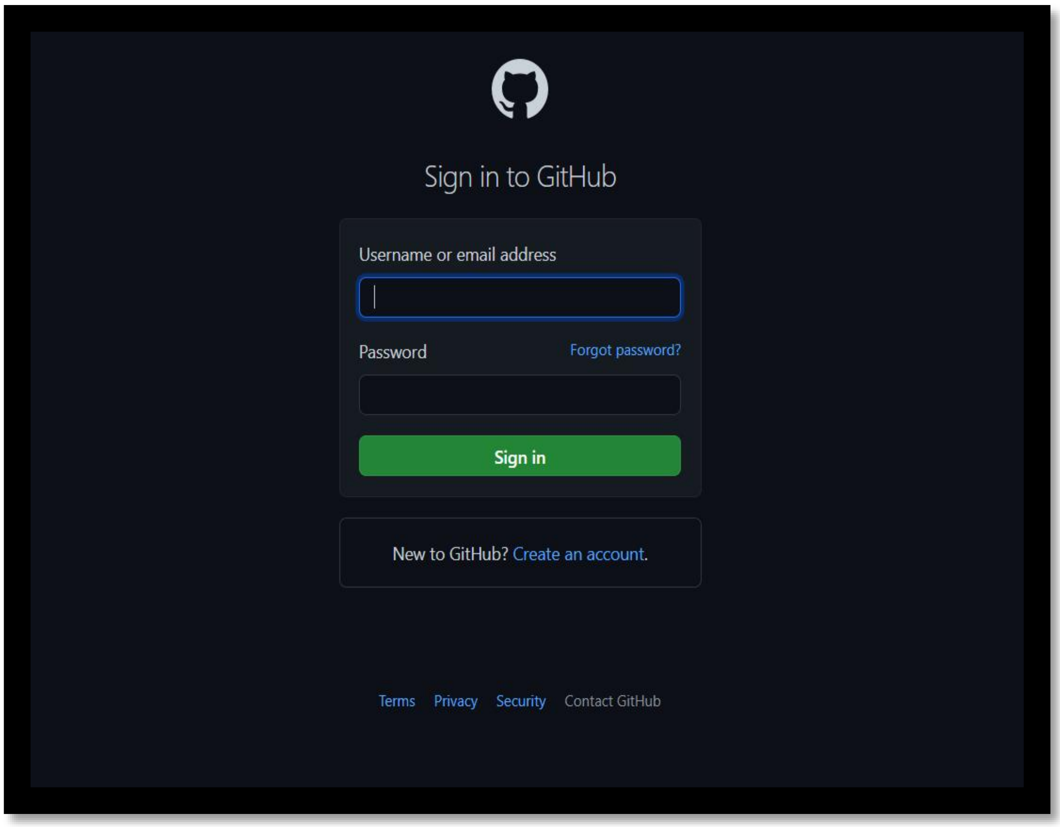
Snapshots –



Where the world builds software

Millions of developers and companies build, ship, and maintain their software on GitHub—the largest and most advanced development platform in the world.

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


Sign in to GitHub

Username or email address

Password

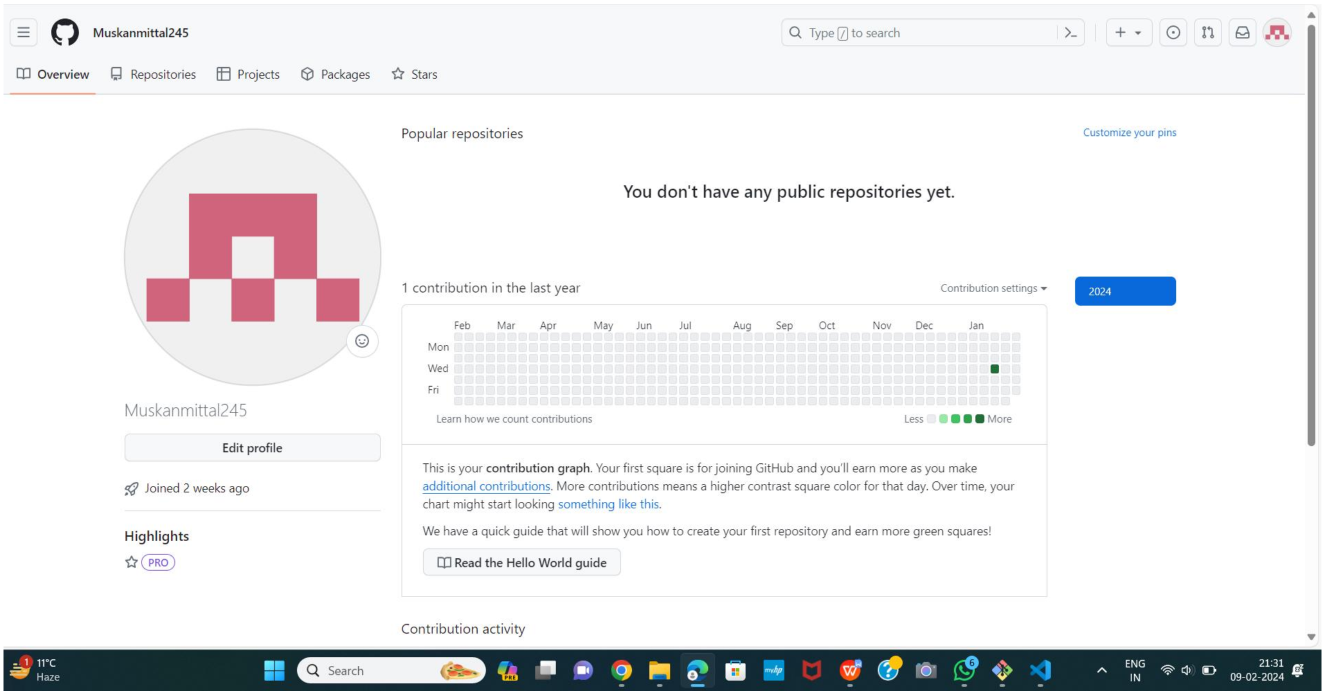
Forgot password?

Sign in

New to GitHub? Create an account.

Terms Privacy Security Contact GitHub

GitHub Login


Muskanmittal245

Type / to search

Overview Repositories Projects Packages Stars

Popular repositories

Customize your pins

You don't have any public repositories yet.

1 contribution in the last year

Contribution settings

2024

Muskanmittal245

Edit profile

Joined 2 weeks ago

Highlights

PRO

Learn how we count contributions

Less More

This is your **contribution graph**. Your first square is for joining GitHub and you'll earn more as you make additional contributions. More contributions means a higher contrast square color for that day. Over time, your chart might start looking something like this.

We have a quick guide that will show you how to create your first repository and earn more green squares!

Read the Hello World guide

Contribution activity

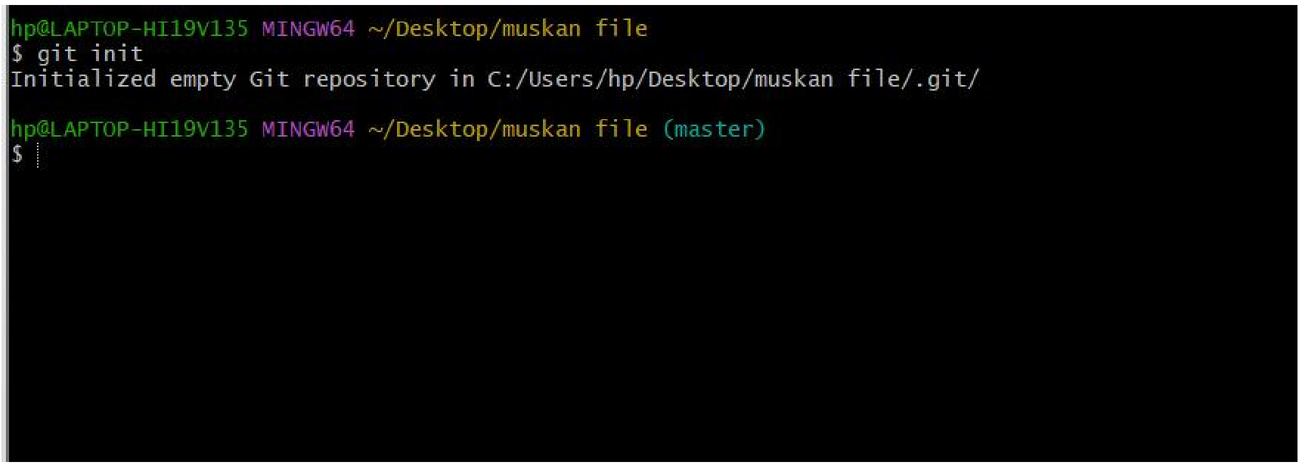
GitHub Interface

Experiment 3

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. 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.

Procedure:
First of all, create a local repository using Git. For this, you have to make a folder in your device, right click and select "Git Bash Here". This opens the Git terminal. To create a new local repository, use the command "git init" and it creates a folder ".git".

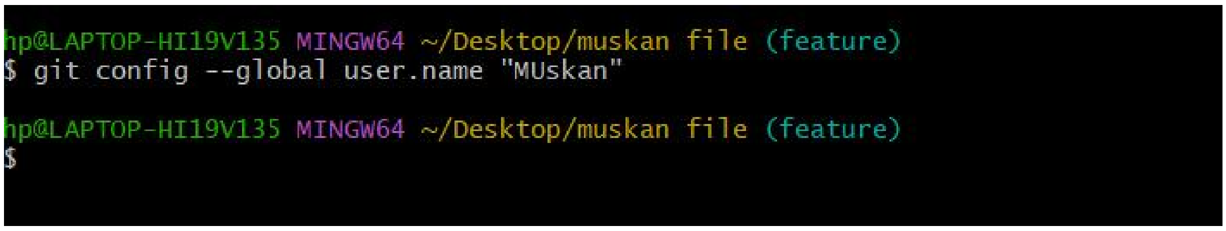


```
hp@LAPTOP-HI19V135 MINGW64 ~/Desktop/muskan file
$ git init
Initialized empty Git repository in C:/Users/hp/Desktop/muskan file/.git/

hp@LAPTOP-HI19V135 MINGW64 ~/Desktop/muskan file (master)
$ 
```

When we use GIT for the first time, we have to give the user name and email sothat if I am going to change in project, it will be visible to all.

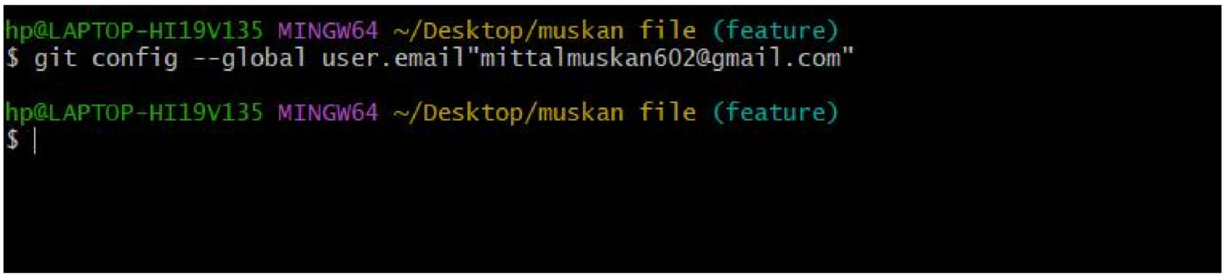
For this, we use command:

"git config --global user.name Name"



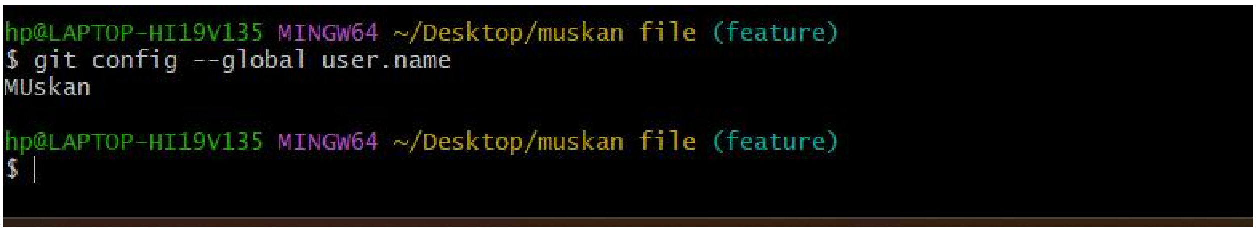
```
hp@LAPTOP-HI19V135 MINGW64 ~/Desktop/muskan file (feature)
$ git config --global user.name "MUskan"

hp@LAPTOP-HI19V135 MINGW64 ~/Desktop/muskan file (feature)
$
```

"git config --global user.email email"



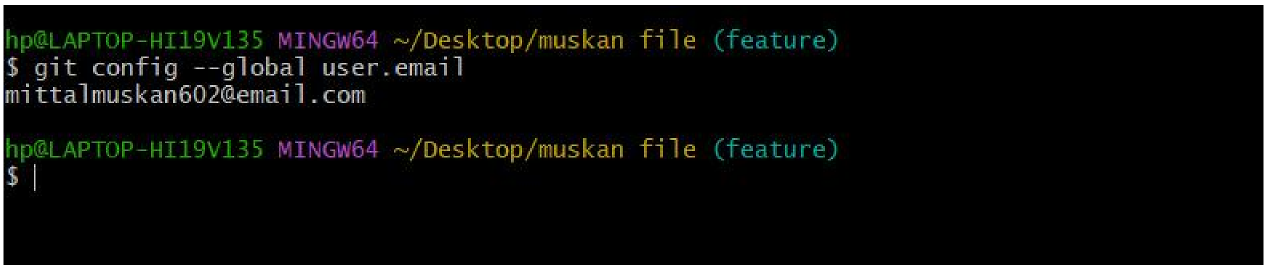
```
hp@LAPTOP-HI19V135 MINGW64 ~/Desktop/muskan file (feature)
$ git config --global user.email"mittalmuskan602@gmail.com"

hp@LAPTOP-HI19V135 MINGW64 ~/Desktop/muskan file (feature)
$ |
```

For verifying the user's name and email, we use:
"git config --global user.name"



```
hp@LAPTOP-HI19V135 MINGW64 ~/Desktop/muskan file (feature)
$ git config --global user.name
MUskan

hp@LAPTOP-HI19V135 MINGW64 ~/Desktop/muskan file (feature)
$ |
```

"git config --global user.email"

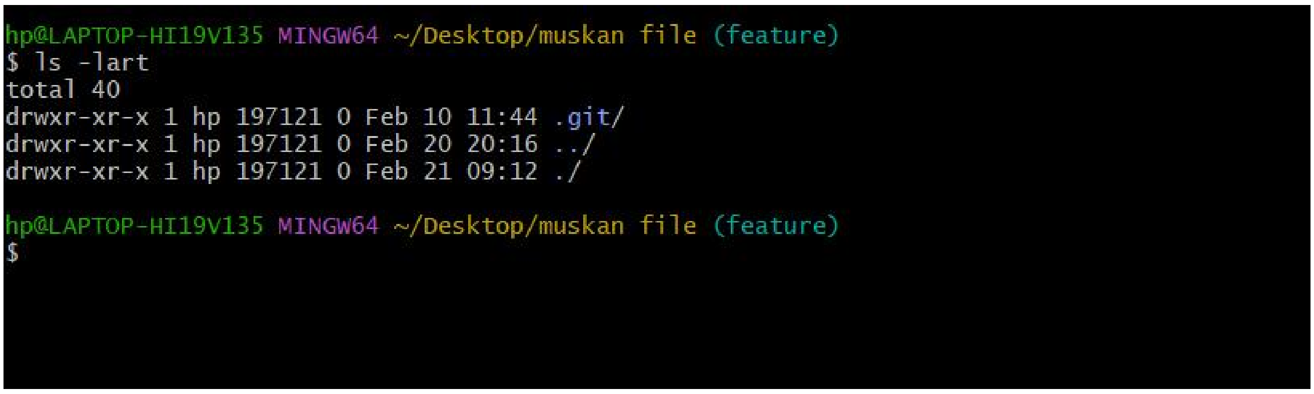


```
hp@LAPTOP-HI19V135 MINGW64 ~/Desktop/muskan file (feature)
$ git config --global user.email
mittalmuskan602@email.com

hp@LAPTOP-HI19V135 MINGW64 ~/Desktop/muskan file (feature)
$ |
```

Some Important Commands

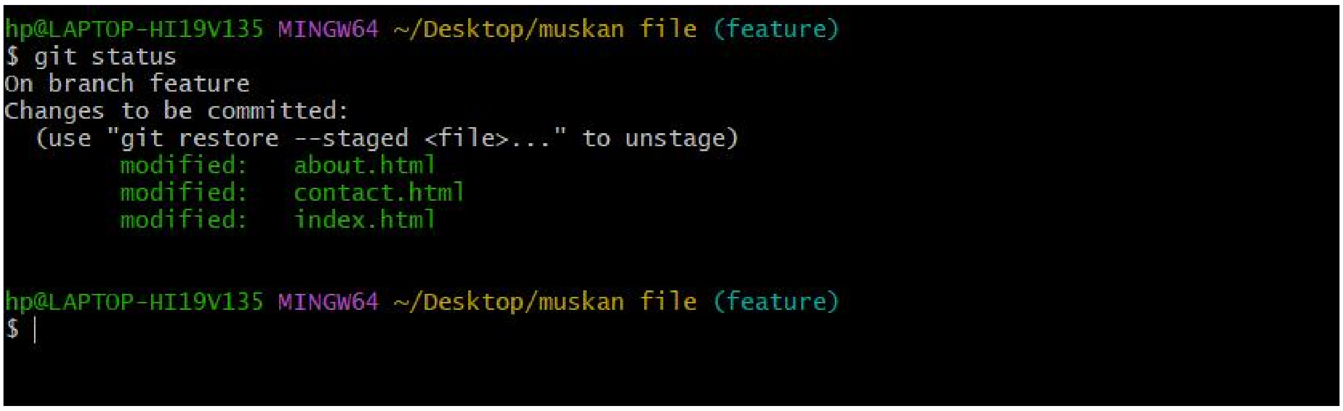
- ls → It gives the file names in the folder.
- ls -lart → Gives the hidden files also.
- git status → Displays the state of the working directory and the staged snapshot.
- touch filename → This command creates a new file in the repository.
- Clear → It clears the terminal.
- rm -rf .git → It removes the repository.
- git log → displays all of the commits in a repository's history
  git diff → It compares my working tree to staging area.

ls -lart:



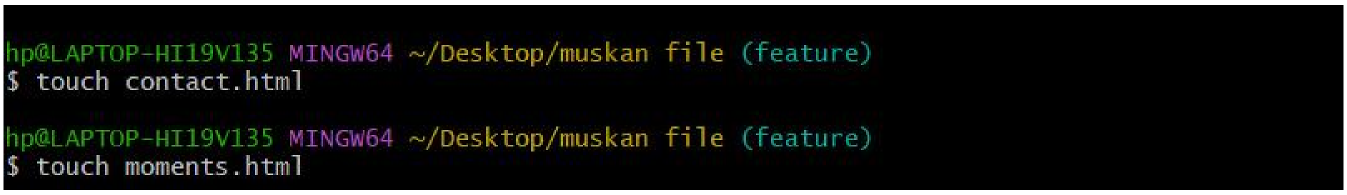
```
hp@LAPTOP-HI19V135 MINGW64 ~/Desktop/muskan file (feature)
$ ls -lart
total 40
drwxr-xr-x 1 hp 197121 0 Feb 10 11:44 .git/
drwxr-xr-x 1 hp 197121 0 Feb 20 20:16 ../
drwxr-xr-x 1 hp 197121 0 Feb 21 09:12 ./

hp@LAPTOP-HI19V135 MINGW64 ~/Desktop/muskan file (feature)
$
```

Git status:



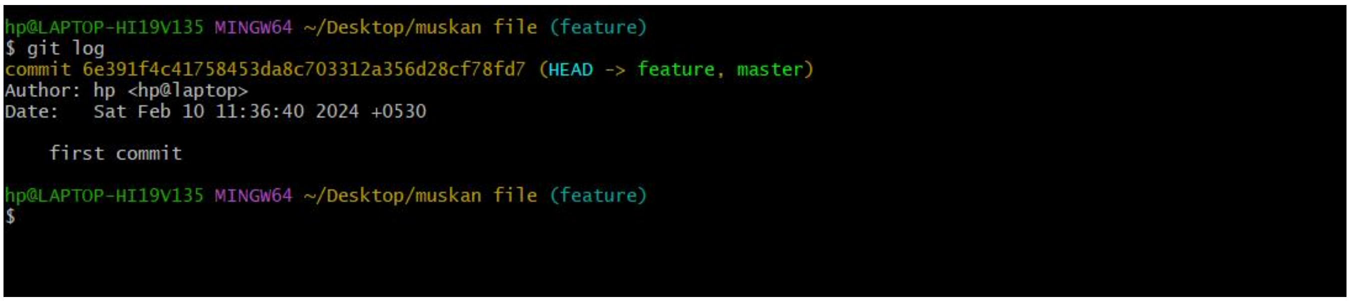
```
hp@LAPTOP-HI19V135 MINGW64 ~/Desktop/muskan file (feature)
$ git status
On branch feature
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   about.html
        modified:   contact.html
        modified:   index.html


hp@LAPTOP-HI19V135 MINGW64 ~/Desktop/muskan file (feature)
$ |
```

Touch filename:



```
hp@LAPTOP-HI19V135 MINGW64 ~/Desktop/muskan file (feature)
$ touch contact.html

hp@LAPTOP-HI19V135 MINGW64 ~/Desktop/muskan file (feature)
$ touch moments.html
```

Git log:



```
hp@LAPTOP-HI19V135 MINGW64 ~/Desktop/muskan file (feature)
$ git log
commit 6e391f4c41758453da8c703312a356d28cf78fd7 (HEAD -> feature, master)
Author: hp <hp@laptop>
Date:   Sat Feb 10 11:36:40 2024 +0530

    first commit

hp@LAPTOP-HI19V135 MINGW64 ~/Desktop/muskan file (feature)
$
```

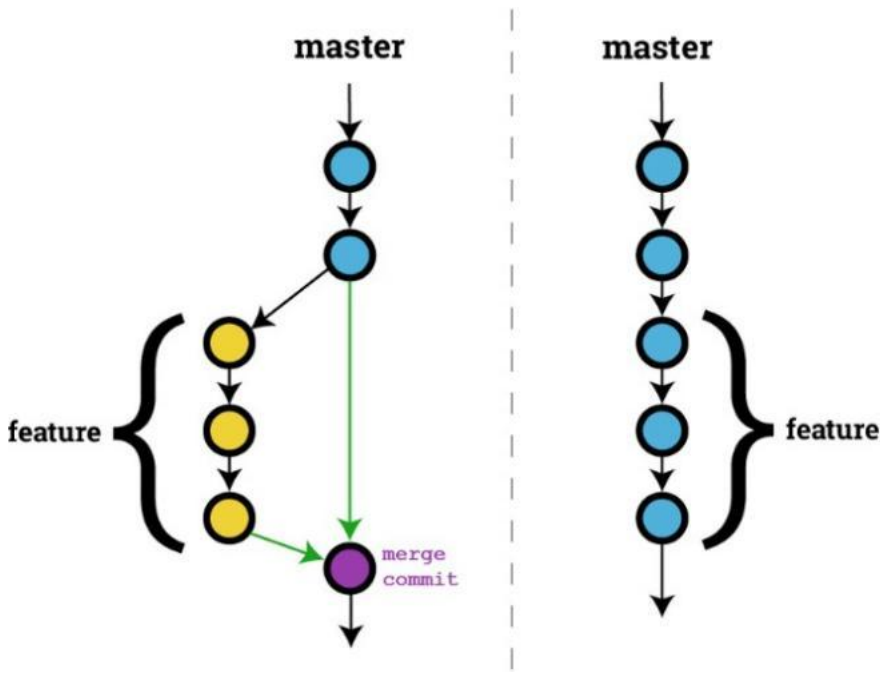
Experiment 4

Aim: Create and visualize branches

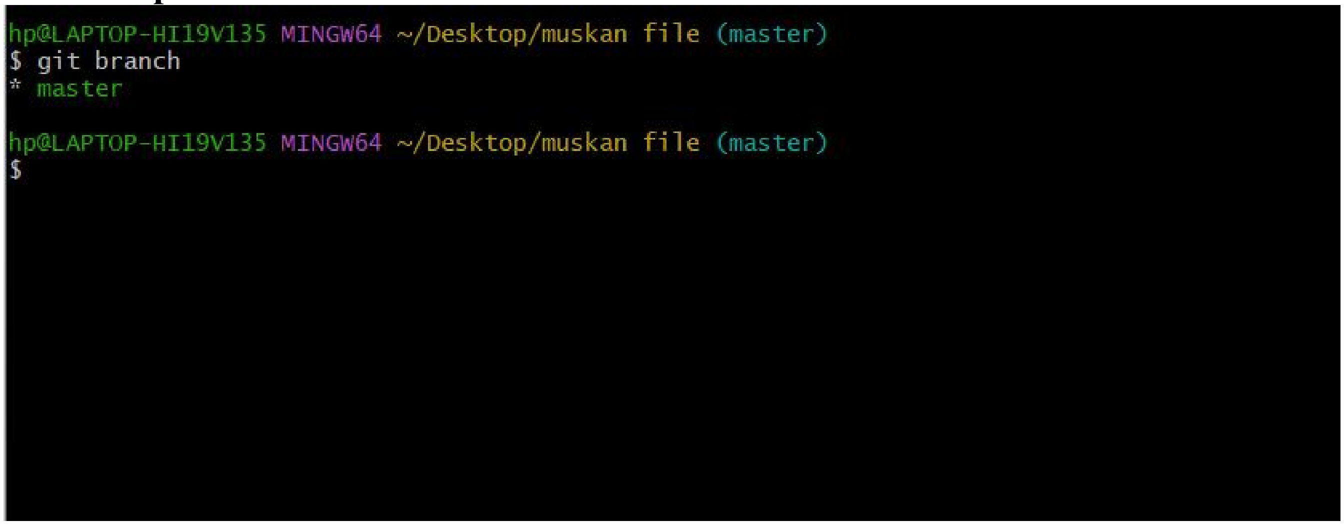
Theory:
Branching: A branch in Git is an independent line of work (a pointer to a specific commit). It allows users to create a branch from the original code (master branch) and isolate their work. Branches allow you to work on different parts of a project without impacting the main branch.

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:

For creating a new branch, git branch name by default is master branch.

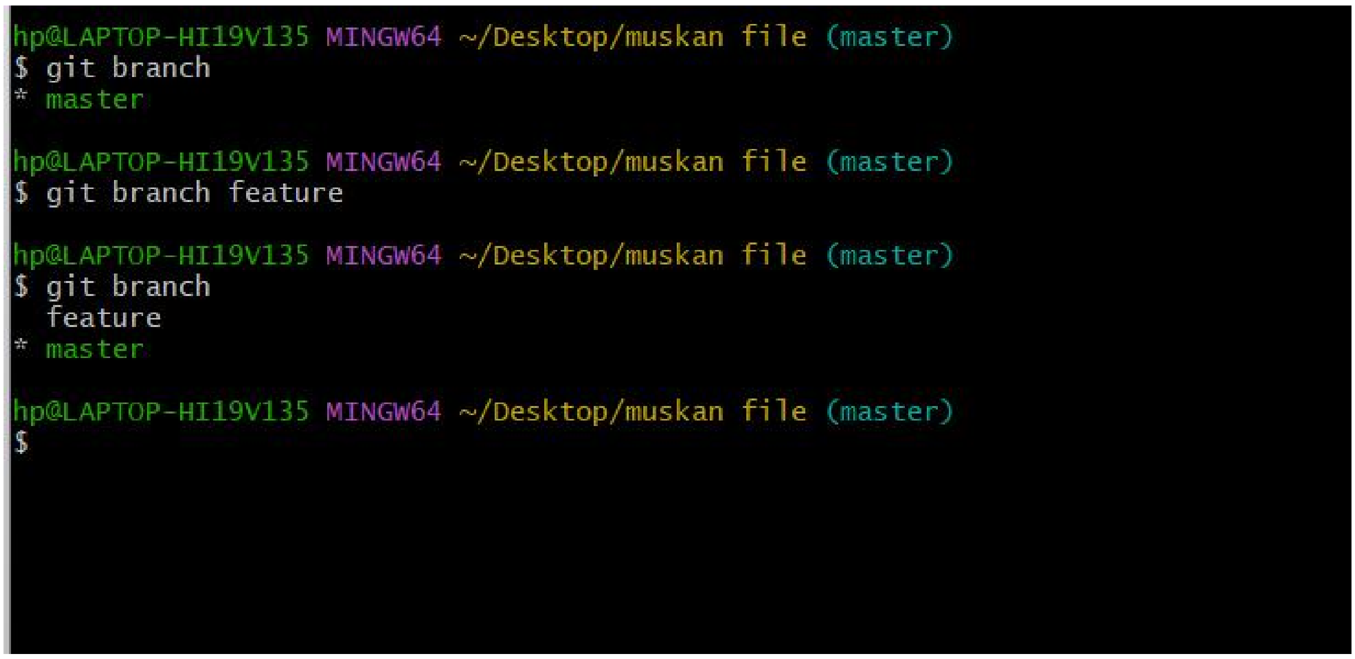


Snapshots –



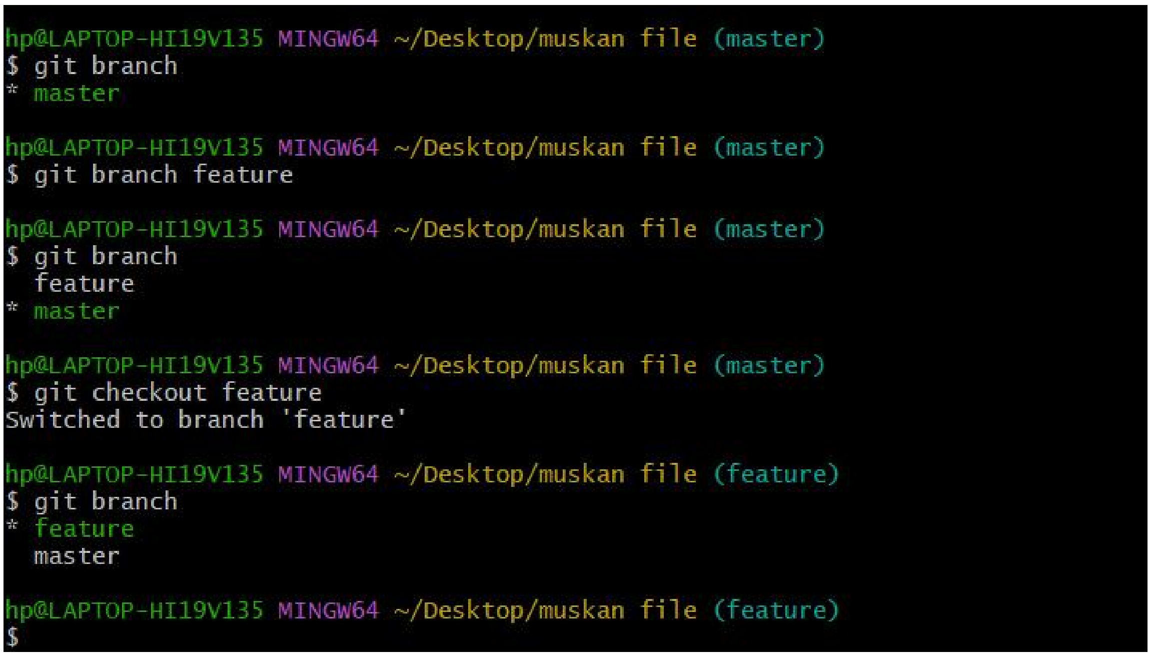
```
hp@LAPTOP-HI19V135 MINGW64 ~/Desktop/muskan file (master)
$ git branch
* master

hp@LAPTOP-HI19V135 MINGW64 ~/Desktop/muskan file (master)
$
```

Default branch is master branch



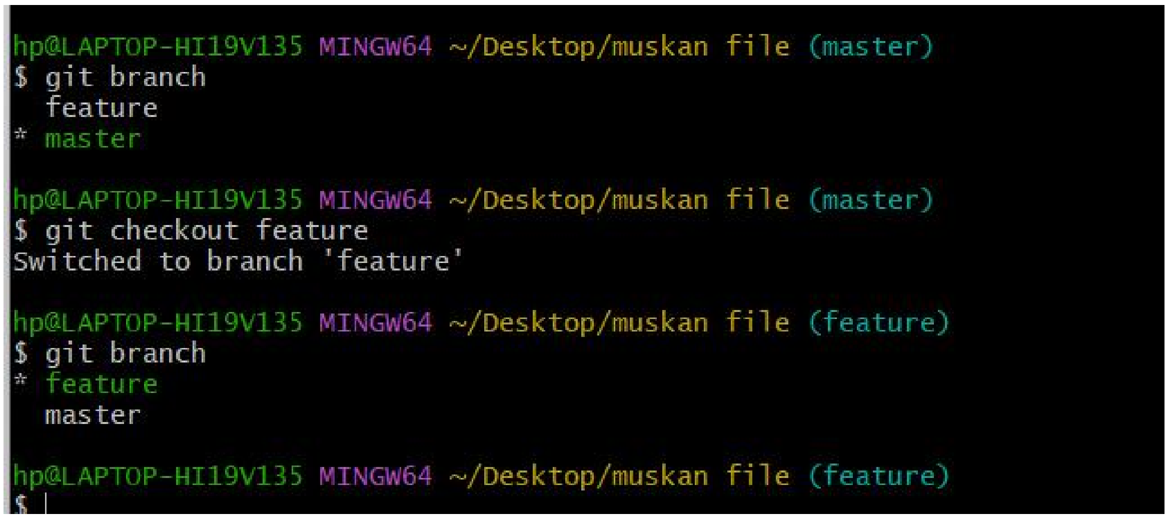
```
hp@LAPTOP-HI19V135 MINGW64 ~/Desktop/muskan file (master)
$ git branch
* master

hp@LAPTOP-HI19V135 MINGW64 ~/Desktop/muskan file (master)
$ git branch feature

hp@LAPTOP-HI19V135 MINGW64 ~/Desktop/muskan file (master)
$ git branch
  feature
* master

hp@LAPTOP-HI19V135 MINGW64 ~/Desktop/muskan file (master)
$
```

Adding a feature branch



```
hp@LAPTOP-HI19V135 MINGW64 ~/Desktop/muskan file (master)
$ git branch
* master

hp@LAPTOP-HI19V135 MINGW64 ~/Desktop/muskan file (master)
$ git branch feature

hp@LAPTOP-HI19V135 MINGW64 ~/Desktop/muskan file (master)
$ git branch
  feature
* master

hp@LAPTOP-HI19V135 MINGW64 ~/Desktop/muskan file (master)
$ git checkout feature
Switched to branch 'feature'

hp@LAPTOP-HI19V135 MINGW64 ~/Desktop/muskan file (feature)
$ git branch
* feature
  master

hp@LAPTOP-HI19V135 MINGW64 ~/Desktop/muskan file (feature)
$
```

Switching to feature branch



```
hp@LAPTOP-HI19V135 MINGW64 ~/Desktop/muskan file (master)
$ git branch
  feature
* master

hp@LAPTOP-HI19V135 MINGW64 ~/Desktop/muskan file (master)
$ git checkout feature
Switched to branch 'feature'

hp@LAPTOP-HI19V135 MINGW64 ~/Desktop/muskan file (feature)
$ git branch
* feature
  master

hp@LAPTOP-HI19V135 MINGW64 ~/Desktop/muskan file (feature)
$ |
```

Switching to master branch

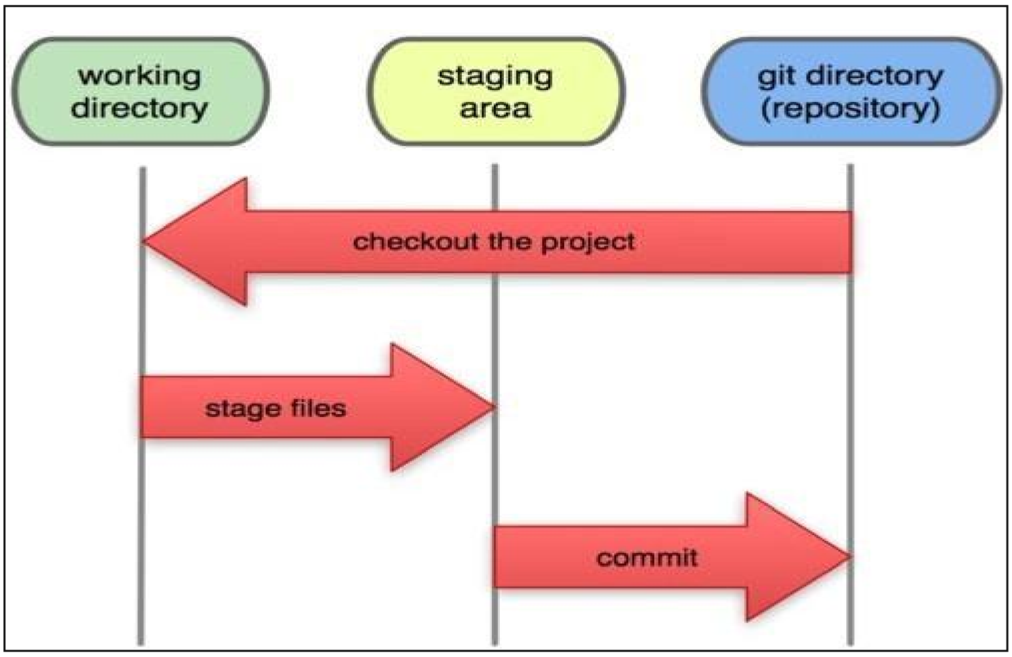
Experiment 5

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

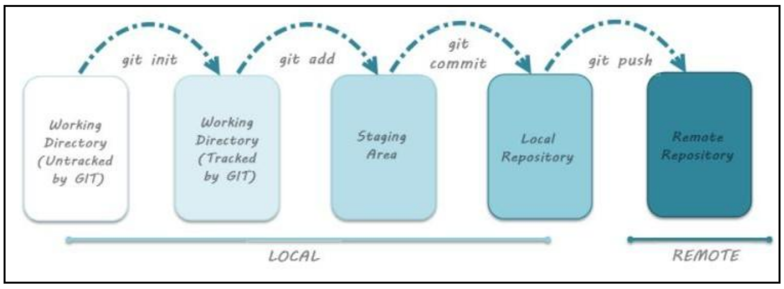


Working Directory:
Consider a project residing in your local system. This project may or may not be tracked by Git. In either case, this project directory is called your Working directory.

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: It 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.


git init
git add
git commit
git push
Working Directory (Untracked by GIT)
Working Directory (Tracked by GIT)
Staging Area
Local Repository
Remote Repository
LOCAL
REMOTE