

SOURCE CODE MANAGEMENT FILE

Submitted by:

Name: Savya Sharma
Roll No. 2210990809
Group 28 - 2
Submission of:
Task 1.1

Submitted To:

Dr. Harveen Kaur
Assistant Professor
Department of Computer
Science & Engineering
Chitkara University Institute
of Engineering and
Technology
Rajpura, Punjab



Source Code Management File

Subject Name: **Source Code Management (SCM)**

Subject Code: **CS181**

Submitted By:

Name: **Savya Sharma**

Roll No. **2210990809**

Group: **G28-2**

Task 1.1 Submission (Week 4)

1. Setting up of Git Client
2. Setting up GitHub Account
3. Generate logs
4. Create and visualize branches

Experiment 1

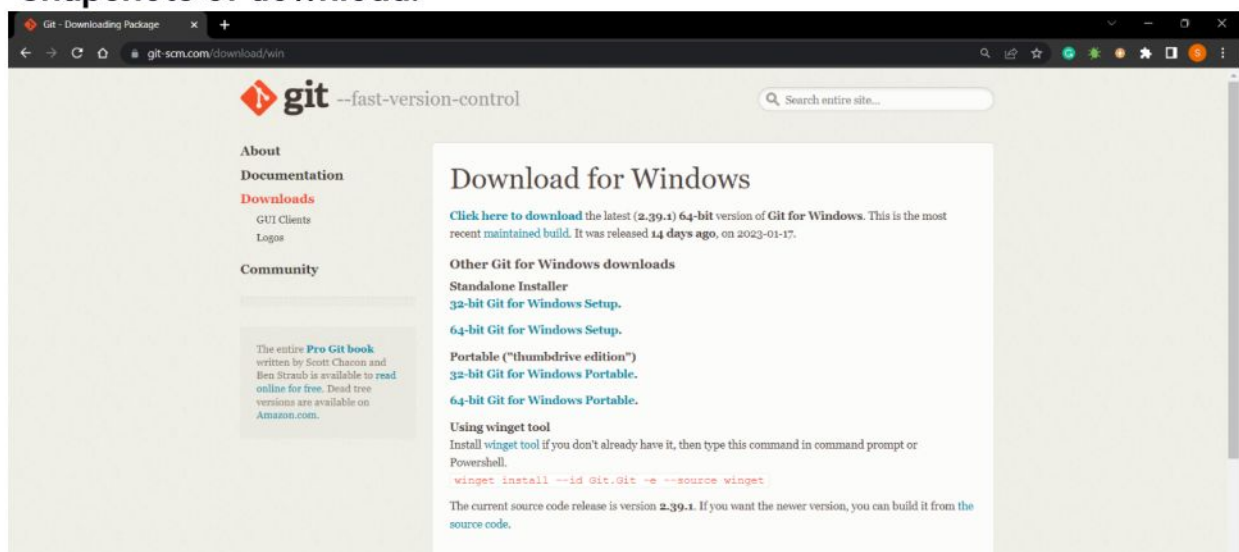
Aim: Setting up of Git Client

Theory:

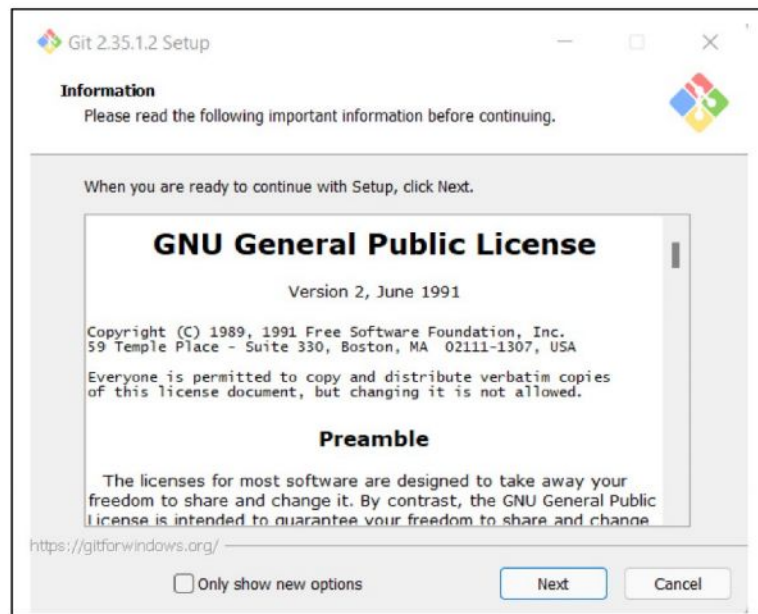
GIT: It's a Version Control System (VCS). It is a software or we can say a server by which we are able to track all the previous changes in the code. It is basically used for pushing and pulling of code. We can use git and git-hub parallelly to work with multiple members or individually. We can make, 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.

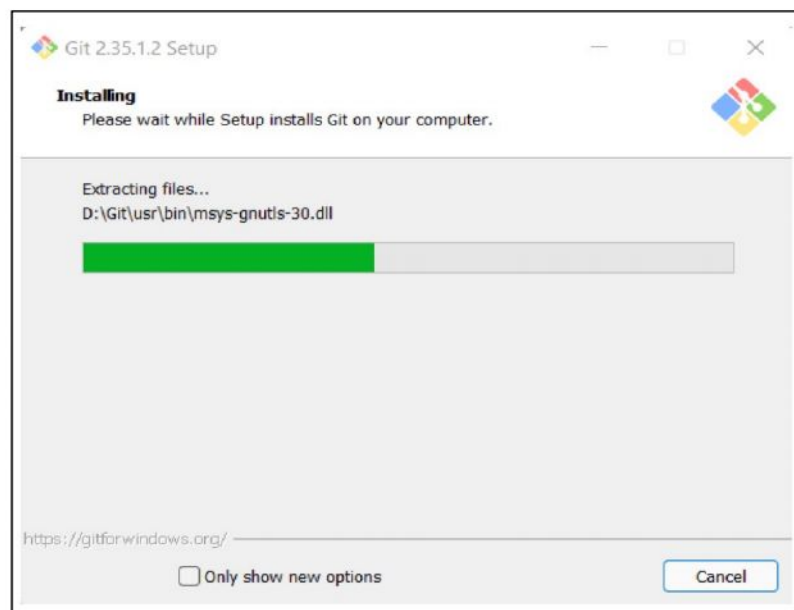
Snapshots of download:



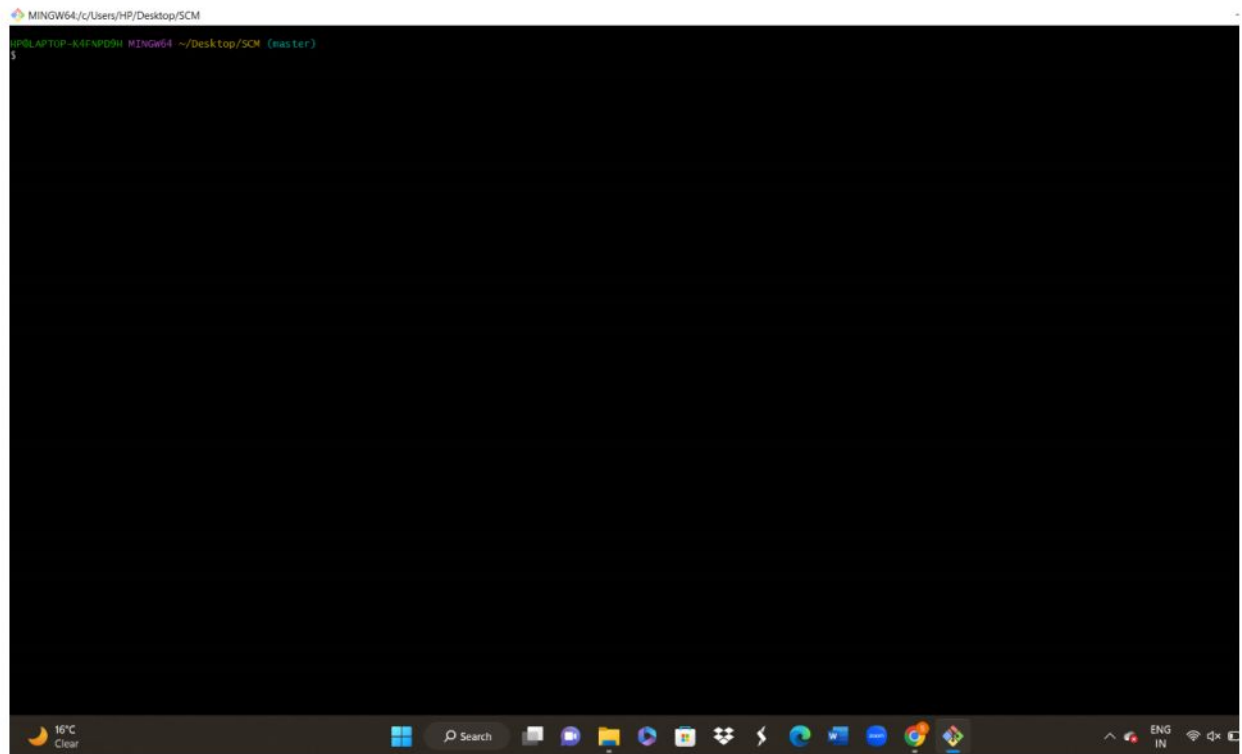
Opted for "64-bit Git for Windows Setup"



Git Setup



Git Installation



Git Bash
launched



Experiment 2

Aim: Setting up GitHub Account

Theory:

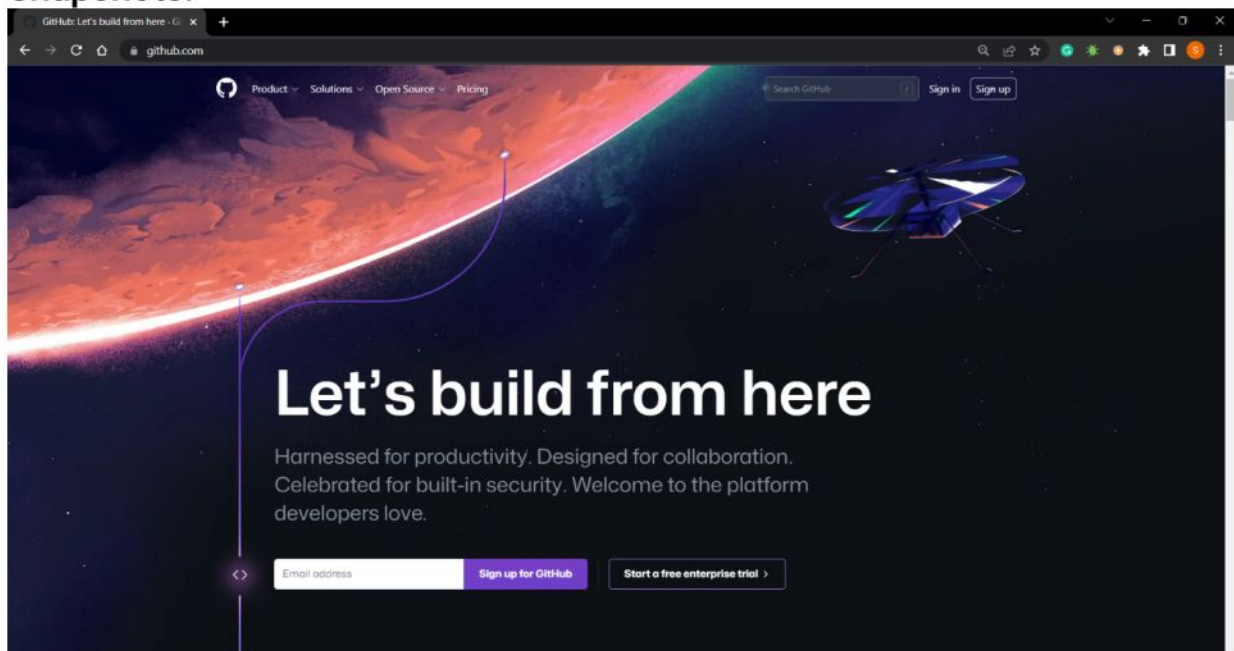
GitHub: GitHub is a website and cloud-based service (client) that helps an individual or developers to store and manage their code. We can also track as well as control changes to our or public code.

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 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 with 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.

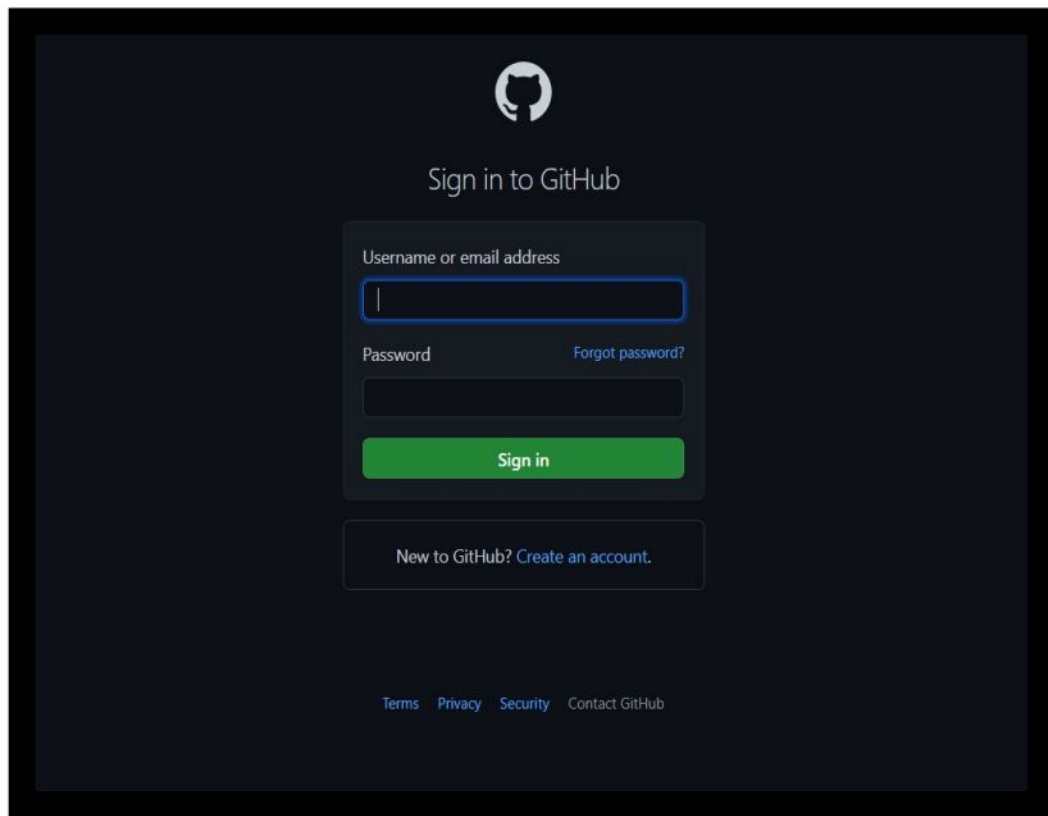
Procedure:

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.

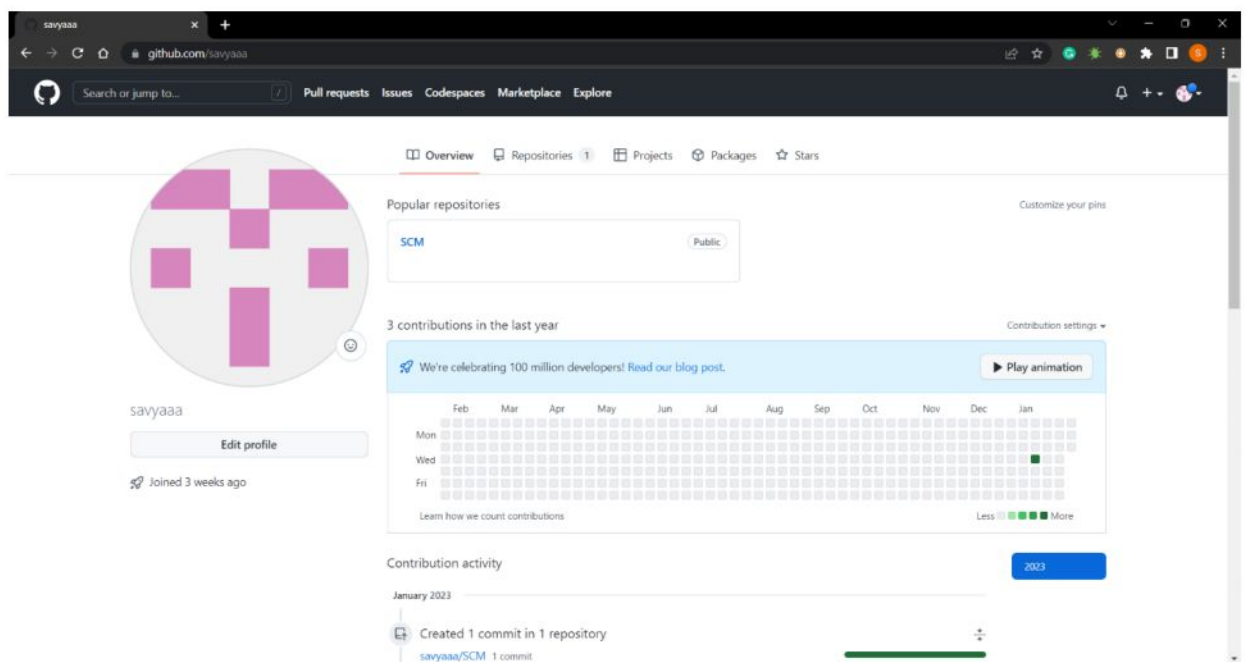
Snapshots:



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.



GitHub Login



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".



```
MINGW64 ~/Desktop/SCM DP
msScorpions@mscorp ~$ git init
Initialized empty Git repository in C:/Users/HP/Desktop/SCM DP/.git/
msScorpions@mscorp ~$
```

When we use GIT for the first time, we have to give the user name and email so that 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"

"git config --global user.email email"

For verifying the user's name and email, we use:

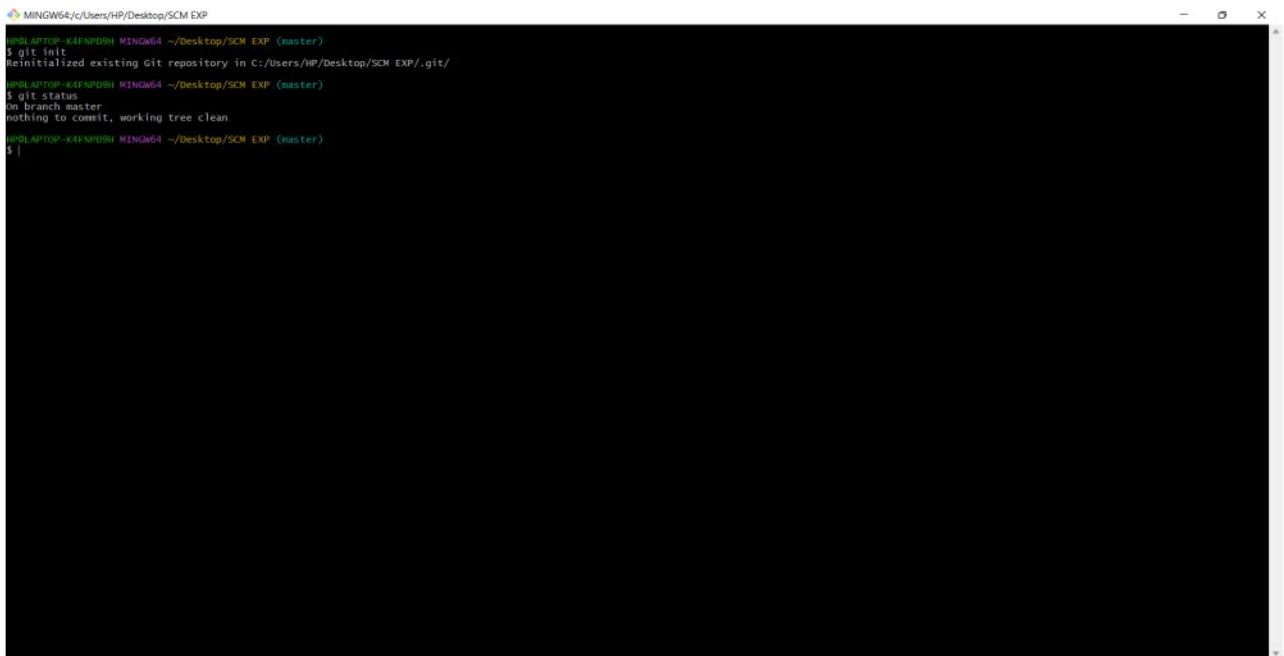
"git config --global user.name"

"git config --global user.email"

Some Important Commands

- ls → It gives the file names in the folder.
- ls -la → 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.

Git Status:



```
MINGW64~/Users/HP/Desktop/SCM EXP
mslaptop-k4fnp08h MINGW64 ~/Desktop/SCM EXP (master)
$ git init
Initialized existing Git repository in C:/Users/HP/Desktop/SCM EXP/.git/
mslaptop-k4fnp08h MINGW64 ~/Desktop/SCM EXP (master)
$ git status
On branch master
nothing to commit, working tree clean
mslaptop-k4fnp08h MINGW64 ~/Desktop/SCM EXP (master)
$
```

The git log command displays a record of the commits in a Git repository. By default, the git log command displays a commit hash, the commit message and other commit metadata.

Git Log:



```
MINGW64~/Users/HP/Desktop/SCM EXP
mslaptop-k4fnp08h MINGW64 ~/Desktop/SCM EXP (master)
$ git init
Initialized existing git repository in C:/Users/HP/Desktop/SCM EXP/.git/
mslaptop-k4fnp08h MINGW64 ~/Desktop/SCM EXP (master)
$ git status
On branch master
no commits yet
nothing to commit (create/copy files and use "git add" to track)
mslaptop-k4fnp08h MINGW64 ~/Desktop/SCM EXP (master)
$ git log
fatal: your current branch "master" does not have any commits yet
mslaptop-k4fnp08h MINGW64 ~/Desktop/SCM EXP (master)
$
```

Experiment 4

Aim: Create and visualize branches

i

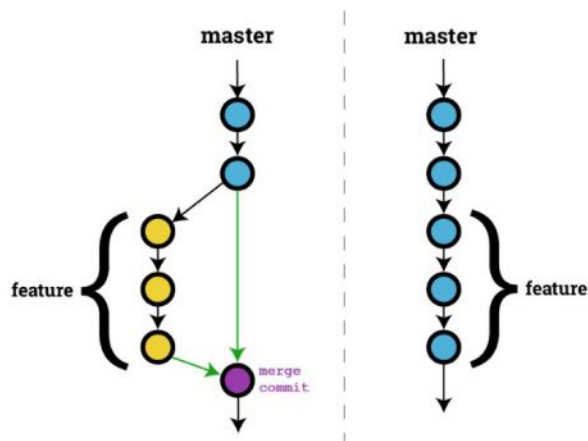
Theory:

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 –

```
MINGW64 ~/Users/HP/Desktop/SCM EXP
git init
$ git branch
* master
git checkout -b feature
$
```

Default branch is master branch

```
MINGW64 ~/Users/JP/Desktop/SCM EXP
mslaptop-kat@mslaptop: ~/Desktop/SCM EXP (master)
$ git branch
* master

mslaptop-kat@mslaptop: ~/Desktop/SCM EXP (master)
$ git branch feature

mslaptop-kat@mslaptop: ~/Desktop/SCM EXP (master)
$ git branch
* master
  feature

mslaptop-kat@mslaptop: ~/Desktop/SCM EXP (master)
$
```

Adding a feature branch

```
MINGW64 ~/Users/JP/Desktop/SCM EXP
mslaptop-kat@mslaptop: ~/Desktop/SCM EXP (master)
$ git branch
* master

mslaptop-kat@mslaptop: ~/Desktop/SCM EXP (master)
$ git branch feature

mslaptop-kat@mslaptop: ~/Desktop/SCM EXP (master)
$ git branch
* master
  feature

mslaptop-kat@mslaptop: ~/Desktop/SCM EXP (master)
$ git checkout feature
Switched to branch 'feature'

mslaptop-kat@mslaptop: ~/Desktop/SCM EXP (feature)
$ git branch
* feature
  master

mslaptop-kat@mslaptop: ~/Desktop/SCM EXP (feature)
$
```

Switching to feature branch

```
MINGW64 ~/Users/JP/Desktop/SCM EXP
mslaptop-kat@mslaptop: ~/Desktop/SCM EXP (master)
$ git branch
* master
  feature

mslaptop-kat@mslaptop: ~/Desktop/SCM EXP (master)
$ git branch feature

mslaptop-kat@mslaptop: ~/Desktop/SCM EXP (master)
$ git branch
* master
  feature

mslaptop-kat@mslaptop: ~/Desktop/SCM EXP (master)
$ git checkout feature
Switched to branch 'feature'

mslaptop-kat@mslaptop: ~/Desktop/SCM EXP (feature)
$ git branch
* feature
  master

mslaptop-kat@mslaptop: ~/Desktop/SCM EXP (feature)
$ git checkout master
Switched to branch 'master'

mslaptop-kat@mslaptop: ~/Desktop/SCM EXP (master)
$ git branch
* master
  feature

mslaptop-kat@mslaptop: ~/Desktop/SCM EXP (master)
$
```

Switching to master branch

```
mslaptop-kat@mslaptop: ~/Desktop/SCM EXP (master)
$ git branch
* master
  feature

mslaptop-kat@mslaptop: ~/Desktop/SCM EXP (master)
$ git tag --online
47465d (HEAD => master, feature) alpha

mslaptop-kat@mslaptop: ~/Desktop/SCM EXP (master)
$
```

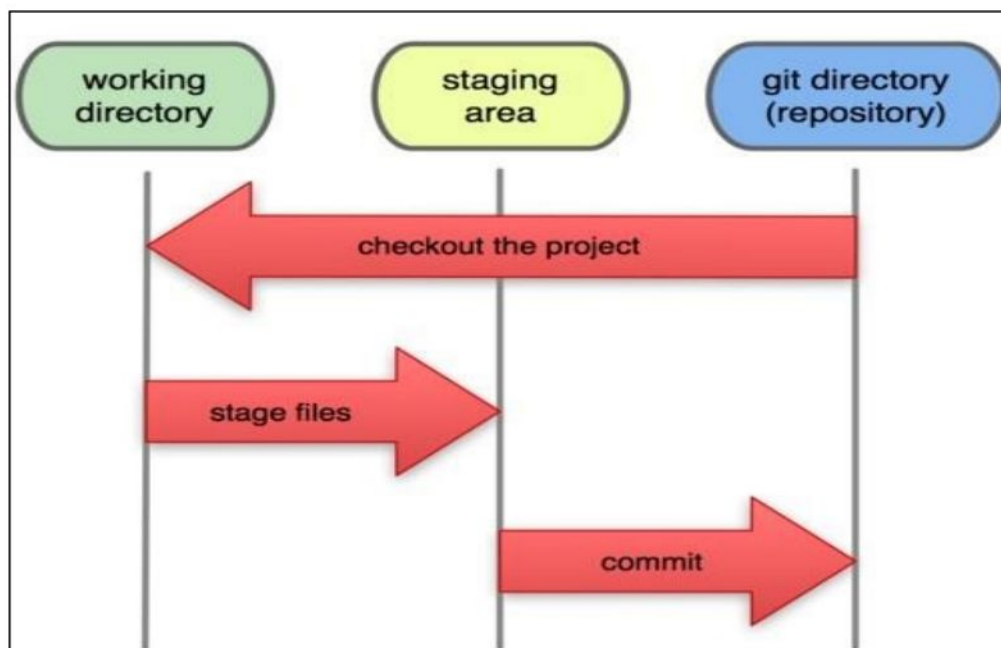
Experiment 5

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.

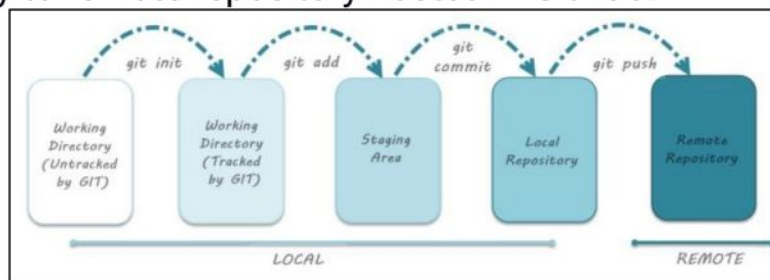
Staging Area:

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.



Subject Name **Source Code Management**

Subject Code **CS181**

Department: **CSE**



CHITKARA
UNIVERSITY

PUNJAB

Submitted By:

Name: Savya Sharma

2210990809

Group 28-2

Submitted To:

Dr. Harveen Kaur

Assistant Professor

Department of Computer

Science & Engineering

Chitkara University Institute of

Engineering and Technology

Rajpura, Punjab