Subject Name: **Source Code Management**

Subject Code: **CS181**

Cluster: **Gamma**

Department: **DCSE**

|  |  |  |
| --- | --- | --- |
| **Submitted By:**    Sajal Nanda  2110991232 |  | **Submitted To:**  Dr. Sarita  Simaiya |



**Department of Computer Science & Engineering**

Chitkara University Institute of Engineering and Technology, Punjab

Jan- June   
(2021-22)

|  |  |  |  |
| --- | --- | --- | --- |
| Institute/School Name | **Chitkara University Institute of Engineering and Technology** | | |
| Department Name | **Department of Computer Science & Engineering** | | |
| Programme Name | **Bachelor of Engineering (B.E.), Computer Science & Engineering** | | |
| Course Name | **Source Code Management** | Session | **2021-22** |
| Course Code | **CS181** | Semester/Batch | **2nd/2021** |
| Vertical Name | **Gamma** | Group No | G15 -B |
| Course Coordinator | **Dr. Neeraj Singla** | | |
| Faculty Name | **Dr. Sarita Simaiya** | | |

Name: Sajal Nanda

Date: April 8, 2022

**Table of Content**

|  |  |  |
| --- | --- | --- |
| **S. No.** | **Title** | **Page No.** |
| 1 | Version control with Git |  |
| 2 | Problem Statement |  |
| 3 | Objective |  |
| 4 | Resources Requirements – Frontend / Backend |  |
| 5 | Concepts and commands |  |
| 6 | Workflow and Discussion |  |
| 7 | Reference |  |

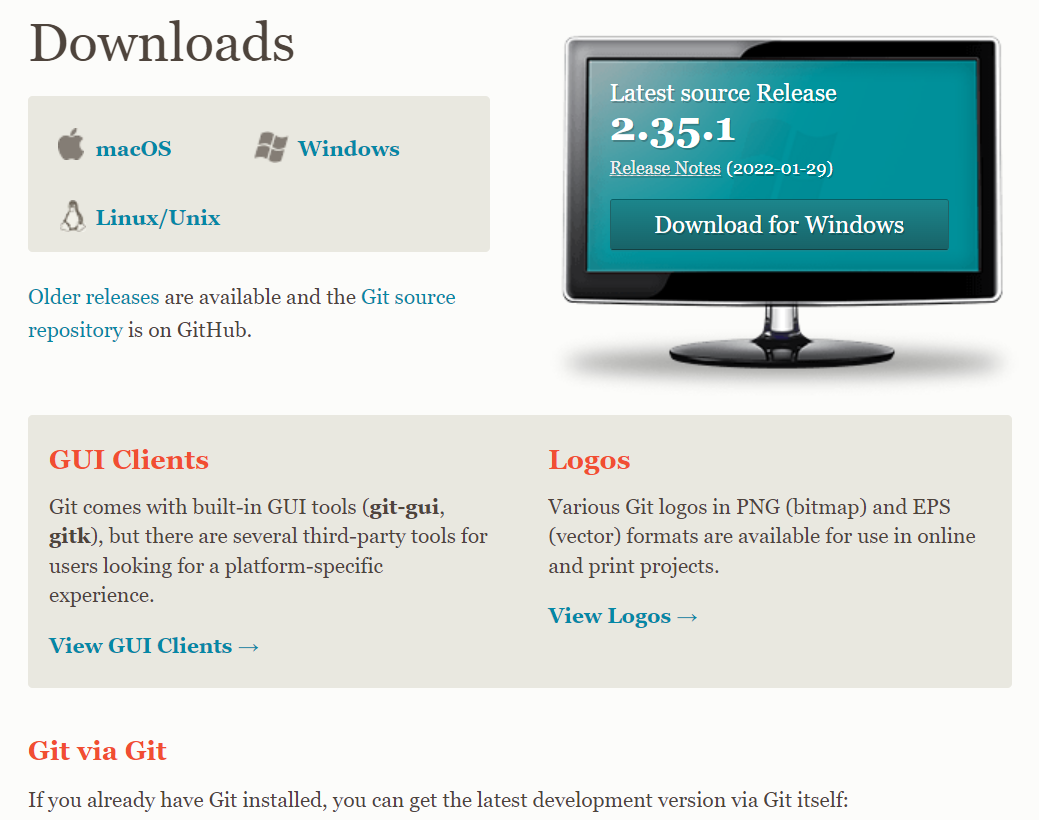
1. Installation of GIT

**Step 1)**

To download the Git installer, visit the Git official site and go to the download page.

The link for the download page is <https://git-scm.com/downloads>

The page looks like as: -



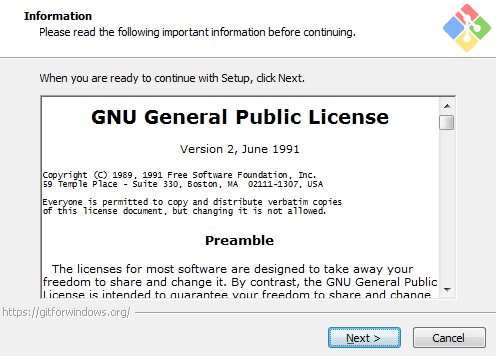
Click on the package given on the page as **download 2.23.0 for windows**. The download will start after selecting the package.

Now, the Git installer package has been downloaded.

**Step 2)**

Click on the download installer file and then click on next.

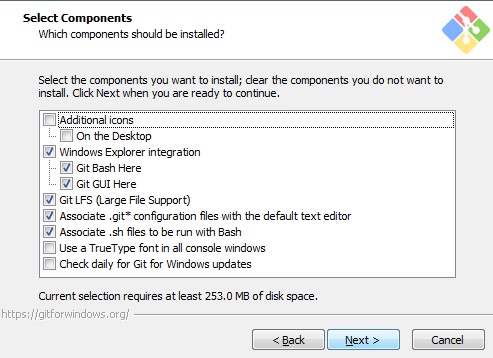
The page looks like as: -



**Step 3)**

Simply click on the next button as it automatically selects the required file.

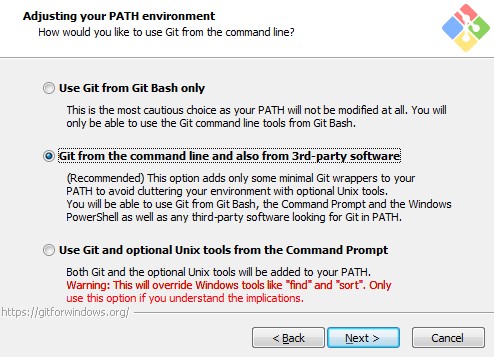
The page looks like as: -



**Step 4)**

You can choose your preferred choice. Click next to continue.

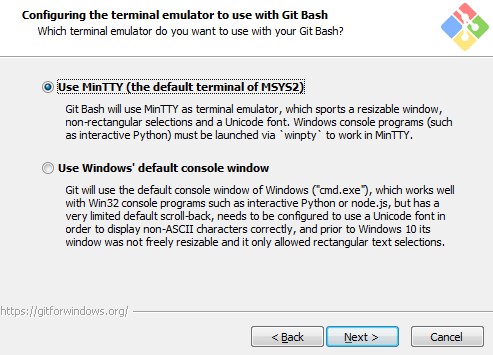
The page looks like as: -



**Step 5)**

***Note*: -** Just simply click on next as it automatically selects the required file.

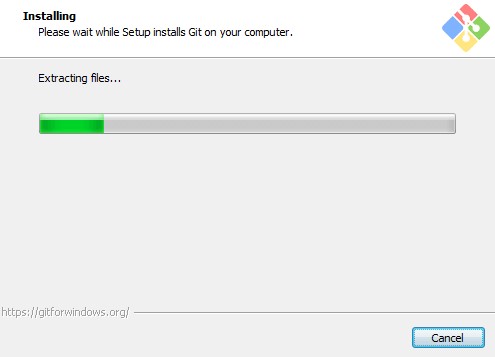
The page looks like as: -

****

**Step6)**

The Git is getting download in your system

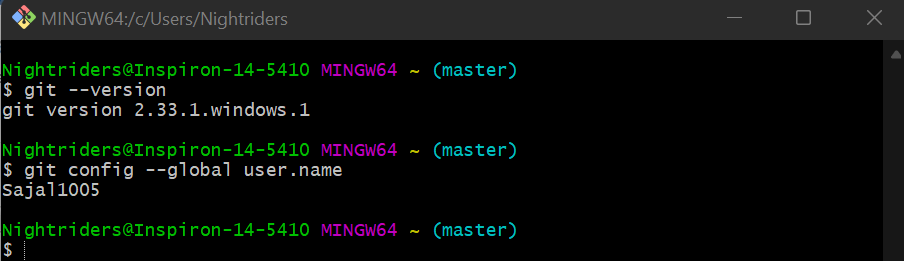
The page looks like as: -



**Step7)**

You can check that Git is install by simply type git - -version in

The page looks like as: -



GIT is finally installed on your desktop.

2. Setting Up GitHub Account

**Step1)**

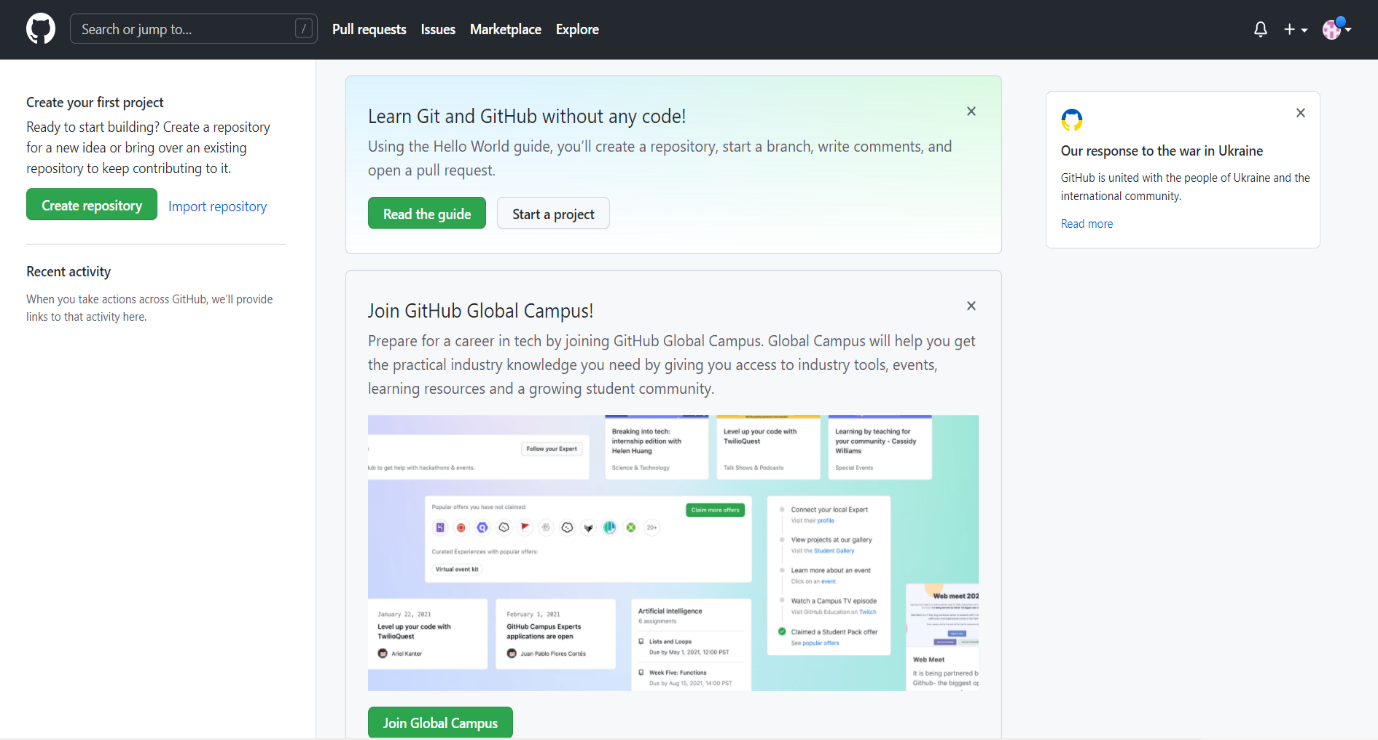
To set up your GitHub account you need to visit <https://github.com/> and click sign-up.

**Step2)**

Enter your email, username and desired password.



Your account is created🚀🚀



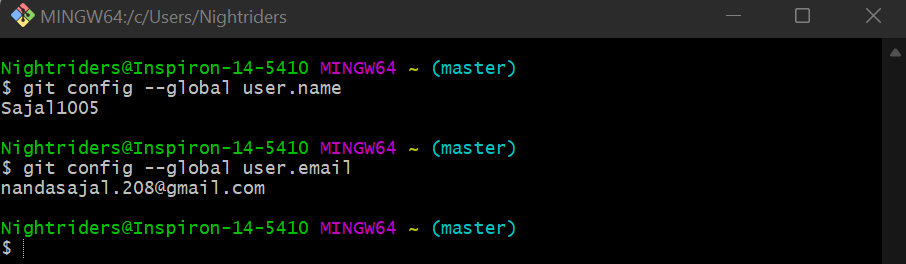
3. Configuration of Git

**Step1)**

You can configure your Git by typing: -

1. Set your username: git config --global user.name "Your Name"
2. Set your email address: git config --global user.email "Your Email

The page looks like as: -

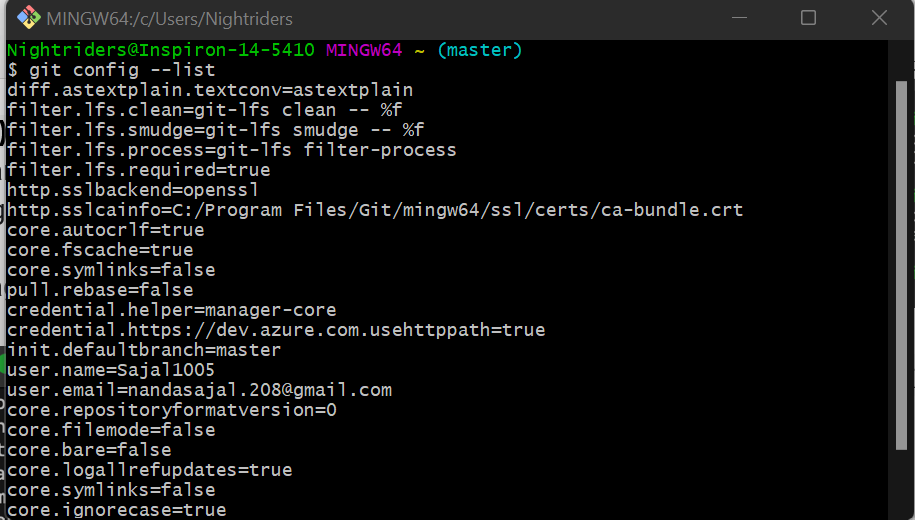


**Step2)**

You can check configuration of Git by typing -

1. git config --list

The page looks like as: -



4. Program to Generate logs

Program to Generate logs:

Advantage of version control systems like git is that it can record changes.

‘Git log’ command is used to display all these changes that were made by the user in the repository. Log is a record of all the previous commits.

To understand Logs we need to get familiar with all the commands that are used in making changes to a repository.

1. Repository:  A repository is a directory that contains all the project-related data.
2. Git init: The git init command is used to create a new blank repository.
3. Git status: We can list all the untracked, modified and deleted files using the git status command.
4. Git add: Adds all the untracked and modified files to the staging area.
5. Git commit: Git commit finalizes the changes in the repository. Every commit is saved in the branch you are working in and can be used to revert back to older versions of the project.

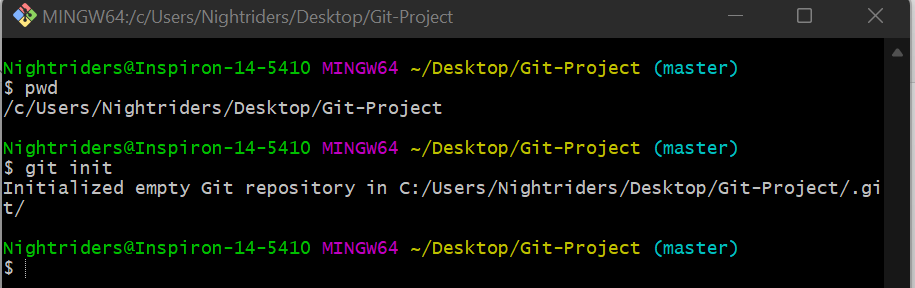
**Making GIT Repository**

**Step1: GIT INIT**

Initializing a new repository, You Can do it by typing: -

1. git init

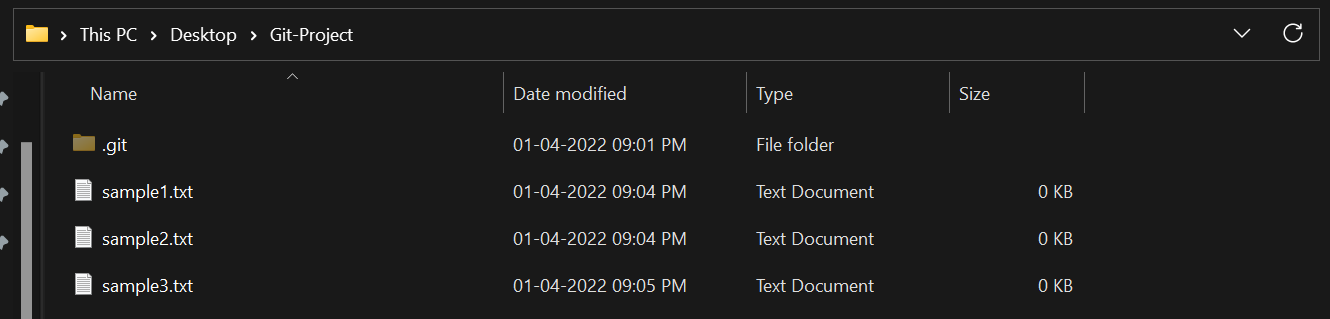
The page looks like as: -



**Step2: ADDING THE FILES TO THE FOLDER**

Just like (Samples...)

The page looks like as: -



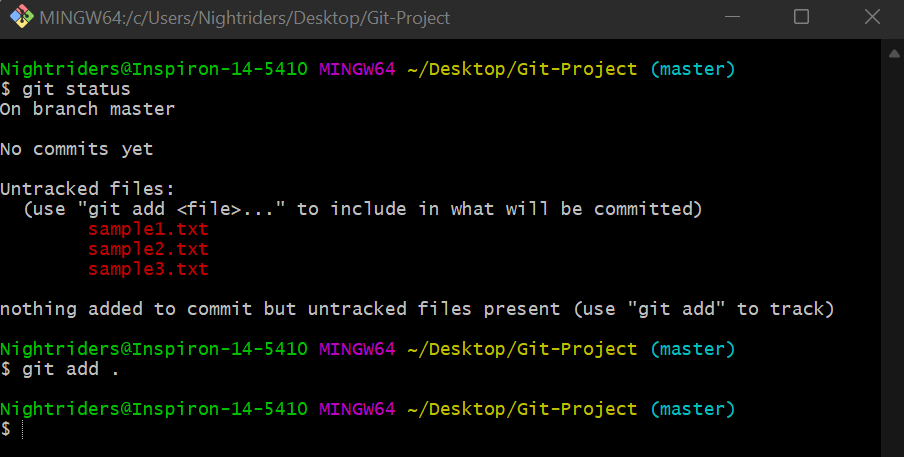
**Step3: GIT ADD**

The git add command adds a change in the working directory to the staging area.

You Can do it by typing: -

1. git add
2. git add (current file name)

The page looks like as: -



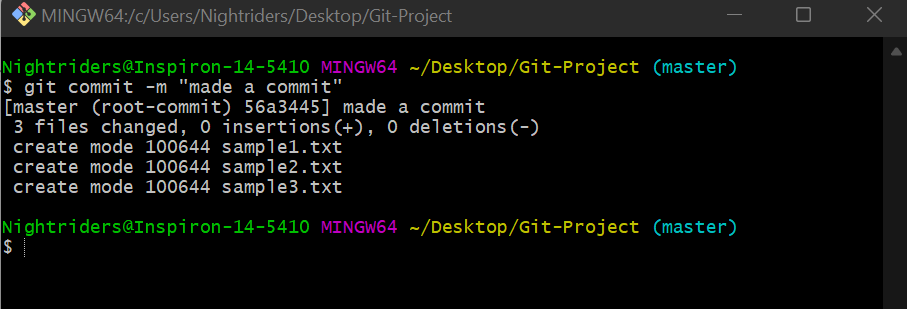
**Step4: GIT COMMIT**

The "commit" command is used to save your changes to the local repository.

You Can do it by typing: -

1. git commit -m”any text”

The page looks like as: -



**Step5: GIT LOG**

Git log will show all the commits made by the author with their time.

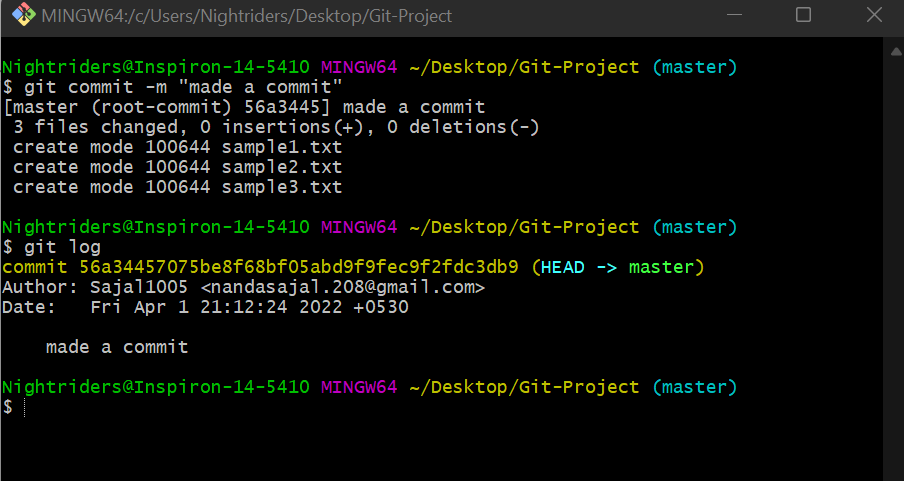
After every commit the checksum value (written In yellow color) of the folder changes.

Checksum is used to verify that the data in that file has not been tampered with or manipulated, possibly by a malicious entity.

You Can do it by typing: -

1. git log

The page looks like as: -



**Step6: Create and Visualize Branches**

A branch in Git is simply a lightweight movable pointer to one of these commits. The default branch name in Git is master.

5. Create and Visualize Branches

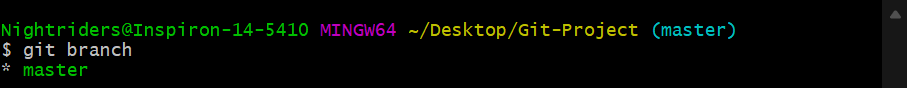
**Step1: CHECKING UP THE BRANCHES**

You can check which branch you are working in by using the command

1. ‘git branch’.

The default branch is always the master branch.

The page looks like as: -



**Step2: CHECKING MULTIPLE BRANCHES**

You Can do it by typing: -

1. git branch (BRANCH NAME)

The page looks like as: -



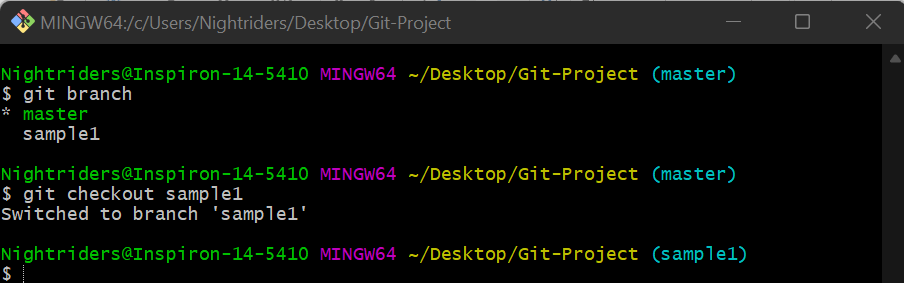
**Step3: CHANGING BRANCHES**

To switch to the other branch

You Can do it by typing: -

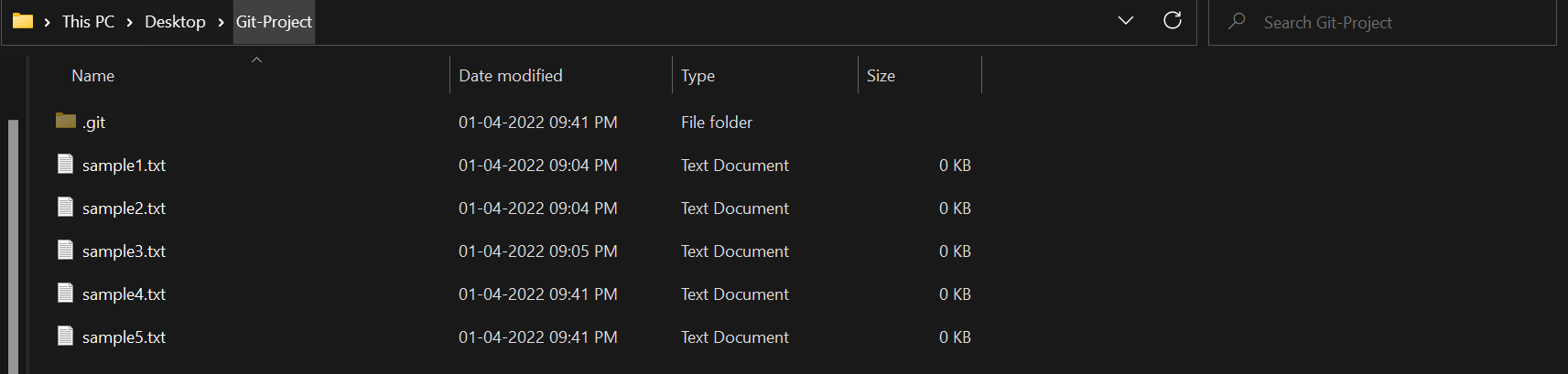
1. git checkout (BRANCH NAME)

The page looks like as: -

****

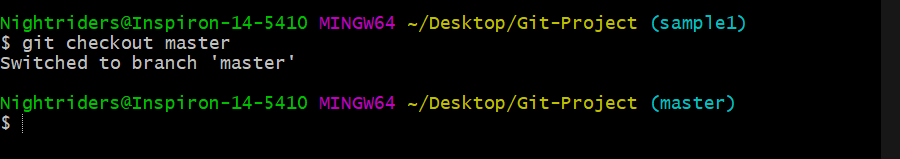
**Step4: NOW ADD FILE TO THE NEW BRANCH AND COMMIT IT**

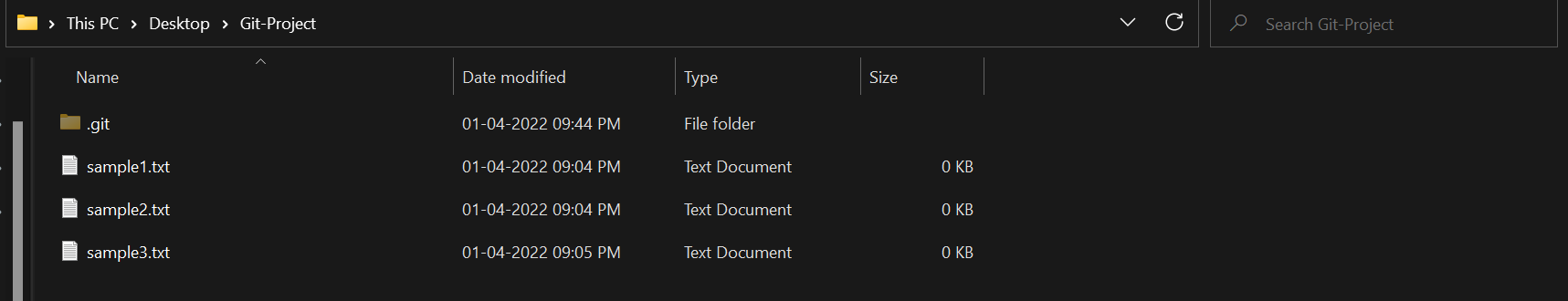
The page looks like as: -

****

**Step5: NOW SWITCH TO BRANCH AND CHECK FILE**

The page looks like as: -

****

****

Now you can see that there is no file named HelloWorld.txt in the master branch because we created the file in the sample1 branch. So, it will be exclusive to the feature1 branch.

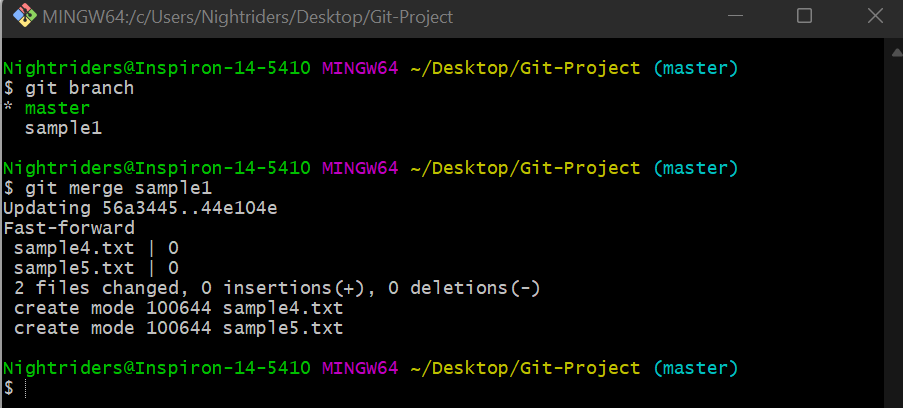
**Step6: GIT MERGING**

Now you can merge two branches by command.

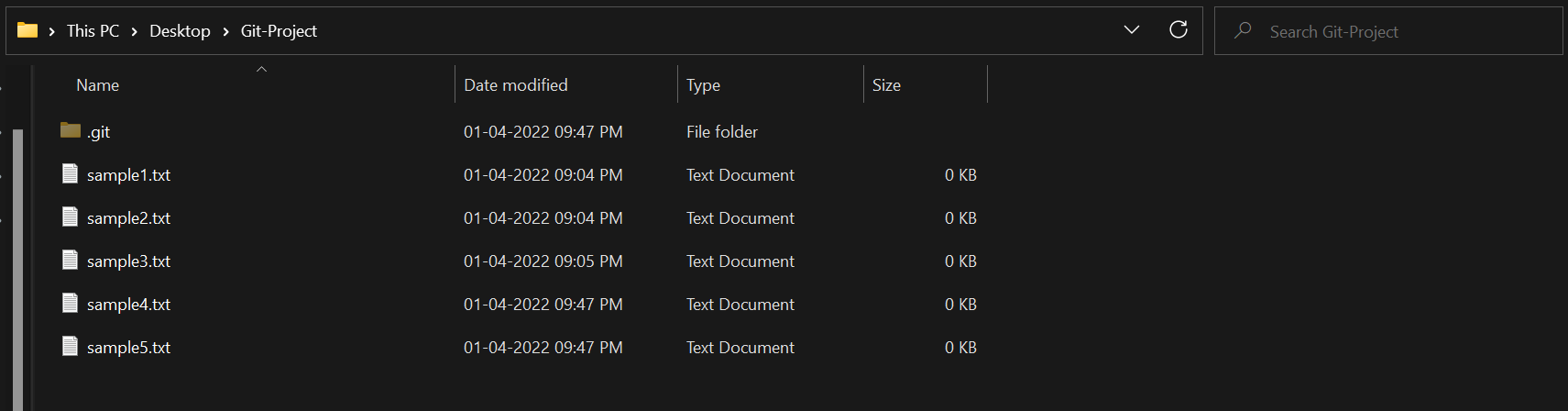
1. git merge (BRANCH NAME)

If you want to merge a new branch in master branch you need to first checkout into the master branch and then run the command.

The page looks like as: -



Now you can check the files in the master branch.

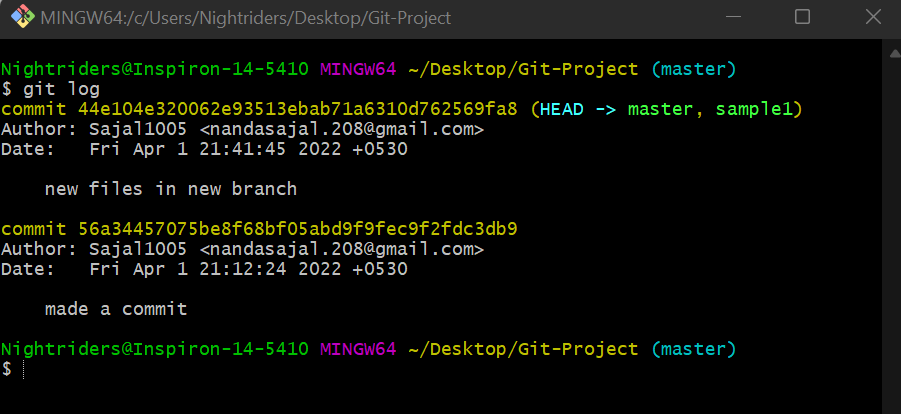


As you can see the sample4.txt and sample5.txt files are added into the master branch.

**Step7: RUNNING GIT LOG**

By running git log command on the master branch you can see all the commits made in master as well as the sample1 branch.

The page looks like as: -



6. Git Lifecycle Description

There are three stages for git lifecycle:

1. Working directory
2. Staging area
3. Git repository

Working Directory:

The working directory is the folder in your local computer where the project files and folders are stored.

The local directory is created by the command ‘git init’ which creates a ‘.git’ named folder which is used to track the files in the directory.

‘.git folder’ is generally hidden but can be tracked enabling hidden files.



Staging area:

The staging area has those files which are supposed to go to the next commit. Only those files which are needed to go to the next commit stay in the staging area.

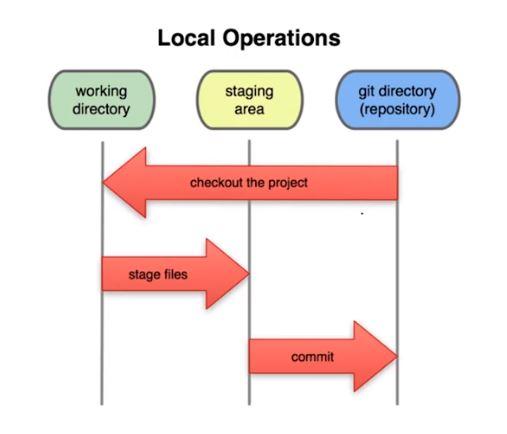
You can shift the files to the git repository by using the command

‘git add --a’.

Git repository:

Now since we have all the files that are to be tracked and are ready in the staging area, we are ready to commit our files using the git commitcommand. Commit helps us in keeping the track of the metadata of the files in our staging area. We specify every commit with a message which tells what the commit is about.

You can commit files by using command ‘git commit -m “message”’



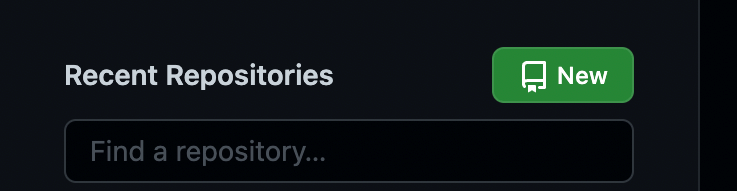
7. UPLOADING DATA ON GITHUB

NOTE-

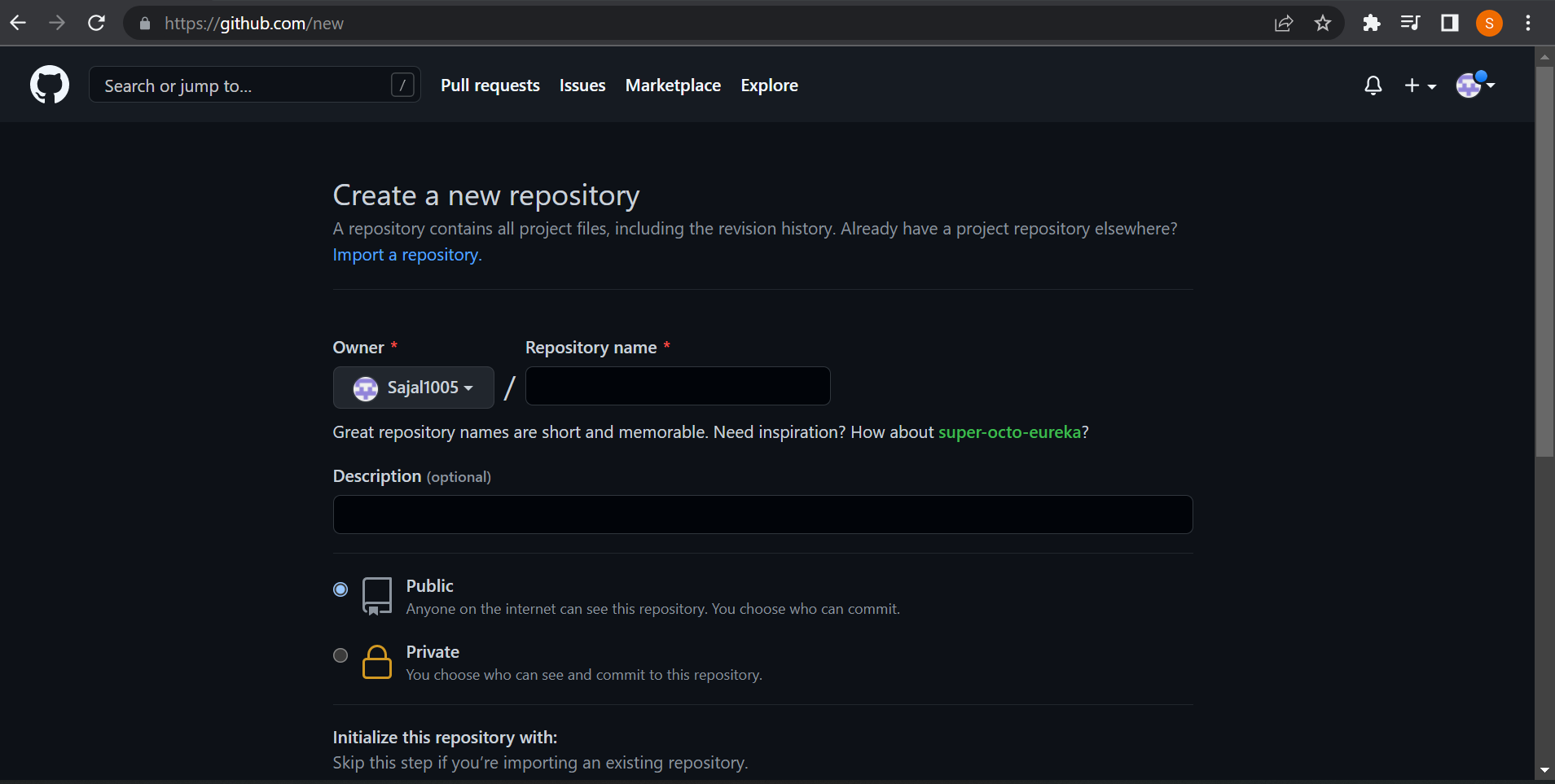
YOU HAVE TO MAKE A REPOSITORY IN GITHUB.

**Step1) CREATING REPOSITORY IN GITHUB**

The page looks like as: -

****

By clicking on new you are able to make a new repository.

****

Write the repository name and click on next.

****

Your GITHUB Repository has been created.

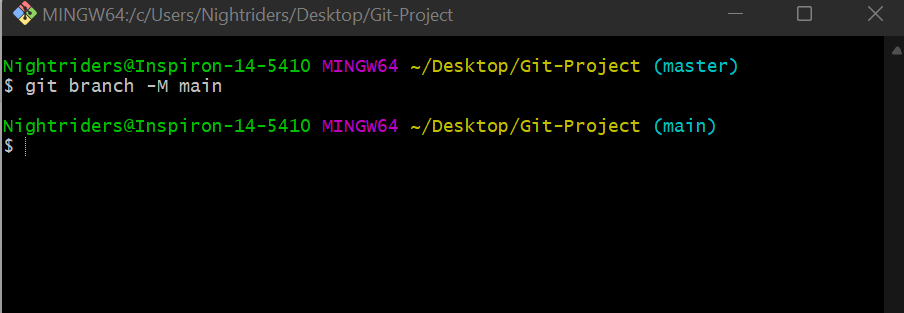
**Step2) GIT ADDING REMOTE BRANCH**

Git stores a branch as a reference to a commit, and a branch represents the tip of a series of commits.

You Can do it by typing: -

1. git branch -M main

The page looks like as: -



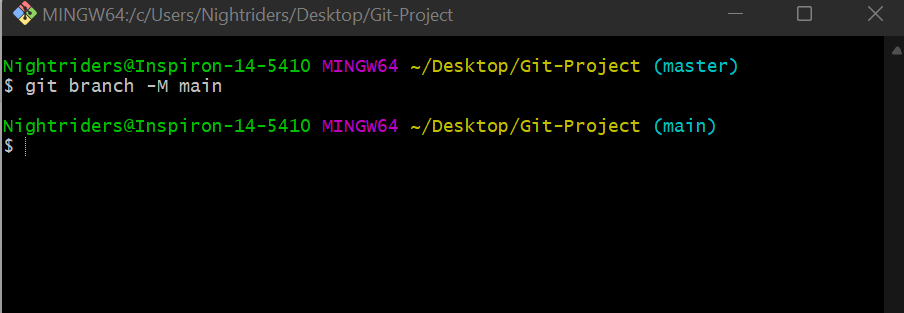
**Step3) GIT ADDING REMOTE ORIGIN**

Is a Git repository that's hosted on the Internet

You Can do it by typing: -

1. git remote add origin (URL)

The page looks like as: -



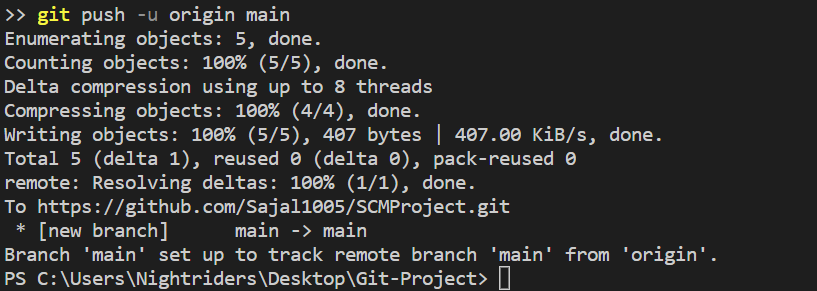
**Step4) GIT PUSHING**

The git push command is used to upload local repository content to a remote repository.

You Can do it by typing: -

1. git push -u origin main

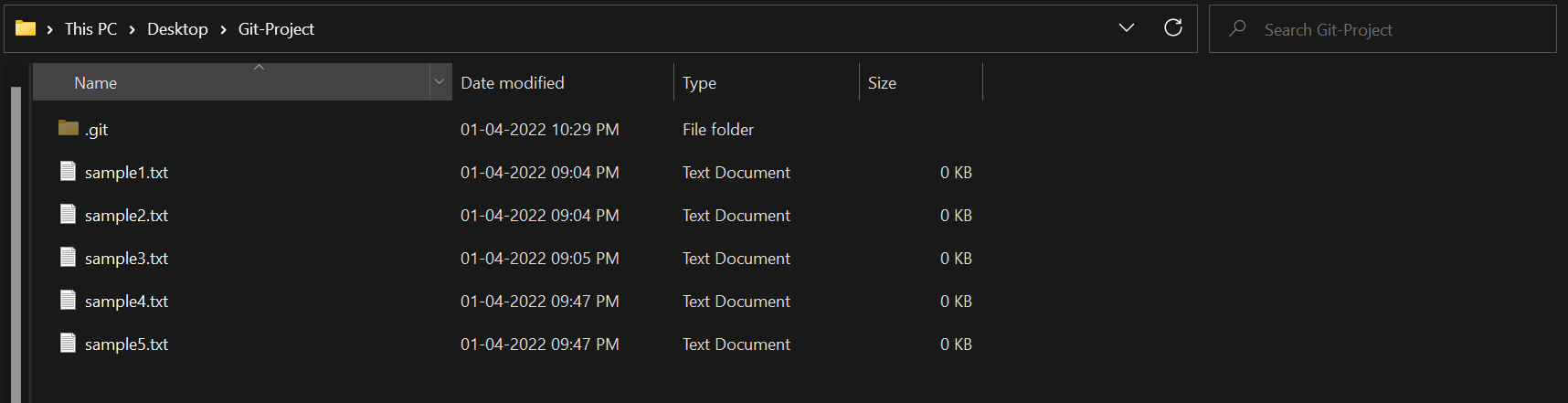
The page looks like as: -

****

**Final Result**

1. Document in your system

The page looks like as: -



1. Document in your GITHUB Repository

The page looks like as: -

