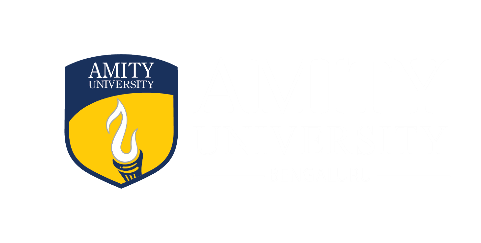
****

Report By:

**Nikhil M**

A866185924003

B.Tech CSE(DA)

Source Code

Management

Dr. Monit Kapoor

**Source Code Management**

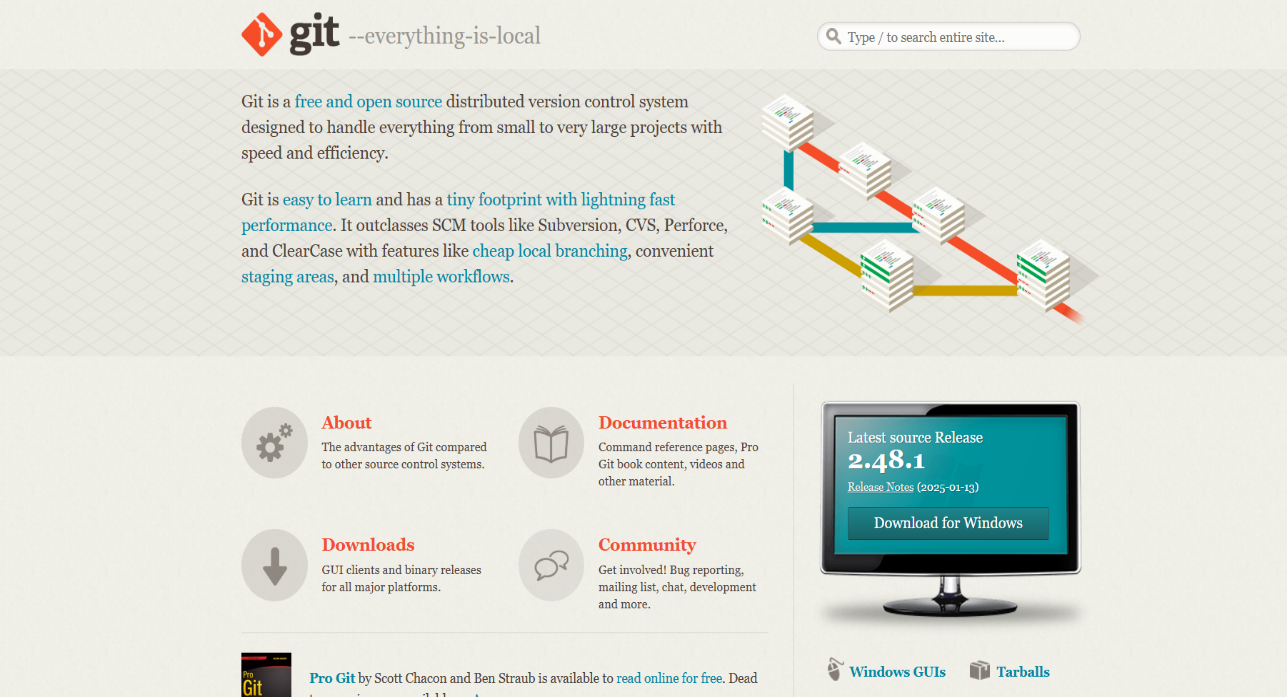
**LAB REPORT**

**Overview of Git:**

Git  is a [distributed version control](https://en.wikipedia.org/wiki/Distributed_version_control) [system](https://en.wikipedia.org/wiki/Software_system) that tracks versions of [files](https://en.wikipedia.org/wiki/Computer_file). It is often used to control [source code](https://en.wikipedia.org/wiki/Source_code) by [programmers](https://en.wikipedia.org/wiki/Programmer) who are developing software collaboratively. Design goals of Git include speed, data integrity, and support for distributed, non-linear workflows — thousands of parallel branches running on different computers.

**Step 1: Downloading Git**

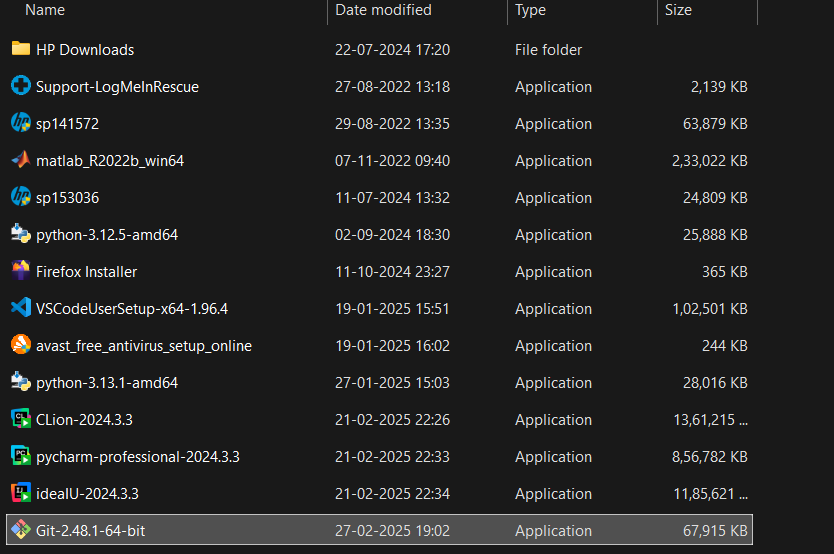
1. Open your web browser and navigate to the official Git website: <https://git-scm.com>.
2. On the homepage, you will see a "**Download**" button that automatically detects your OS. Click on the "Download" button to download the appropriate installer for your operating system (Windows, macOS, or Linux).
3. Alternatively, you can manually select your OS from the website to download a specific version.



**Figure - 1**

**Step 2: Running the Git Installer**

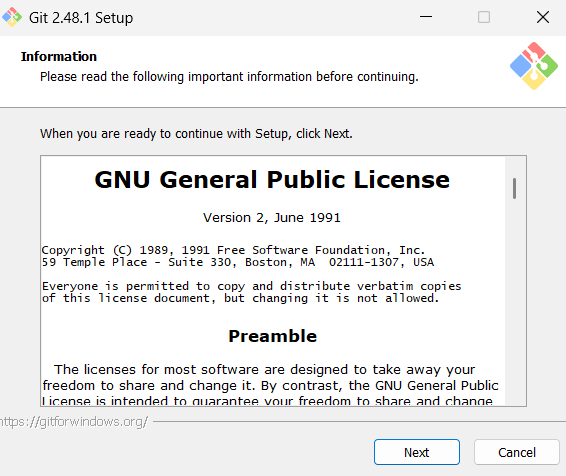
Locate the downloaded **Git.exe** file and double-click to run it.



**Figure – 2**

**Step 3: License (Terms and Conditions)**

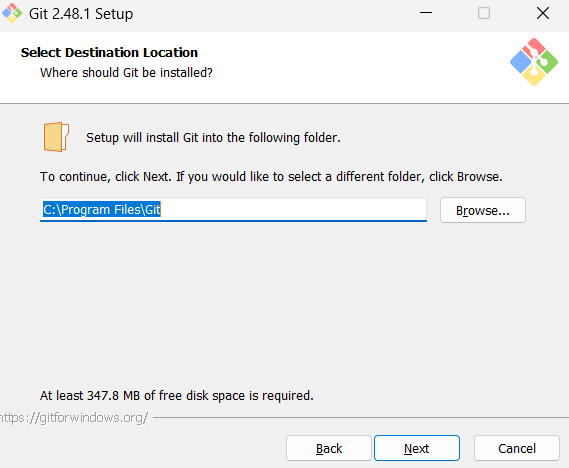
Read the **GNU** General Public License’s terms and conditions and click on **Next**.



**Figure - 3**

**Step 4: Choose Installation Location**

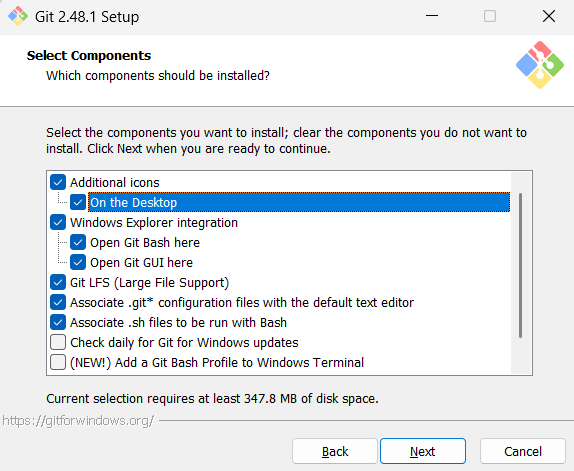
Choose the installation location (default is **C:\Program Files\Git**) and click **Next**.

****

**Figure – 4**

**Step 5: Select the Components**

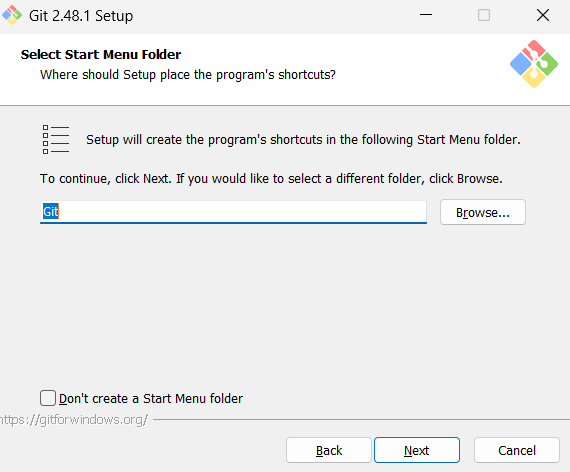
Select the components you want (default options are fine) and click **Next**.



**Figure – 5**

**Step 6: Select Start Menu Folder**

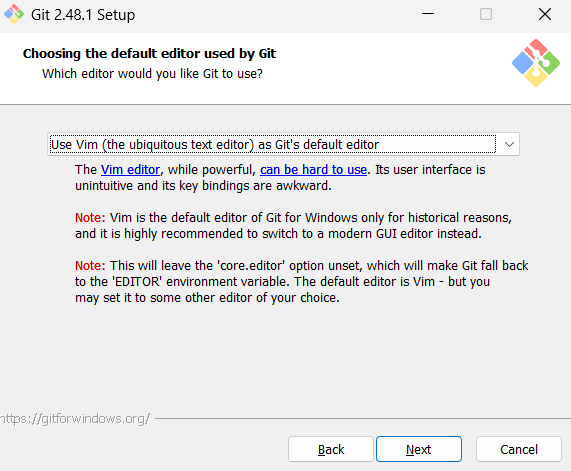
Choose the Start Menu folder where Git shortcuts will be placed. By default, the folder is named **"Git"**. Keep the default name and click **Next** to Proceed.



**Figure – 6**

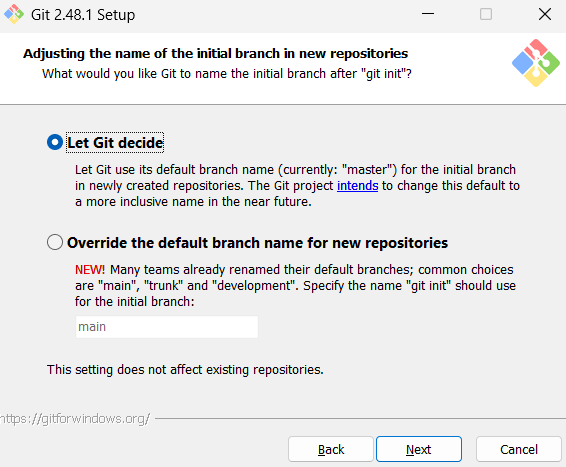
**Step 7: Choose the Text Editor**

Choose a default text editor (select **Vim**) and Click **Next**.

**Figure – 7**

**Step 8: Adjusting Initial Branch Name**

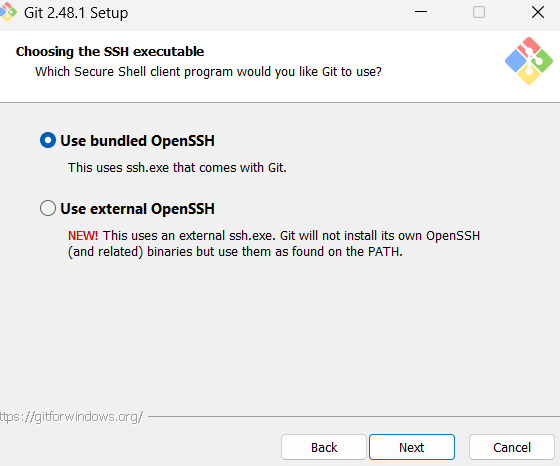
Choose the default name for the first branch when initializing a new Git repository. Go with **‘Let Git Decide’** option setting the branch as **Master** branch and proceed with **Next**.



**Figure – 8**

**Step 9: Choosing the SSH Executable**

Select "**Use bundled OpenSSH**" for better compatibility and Click on **Next**.



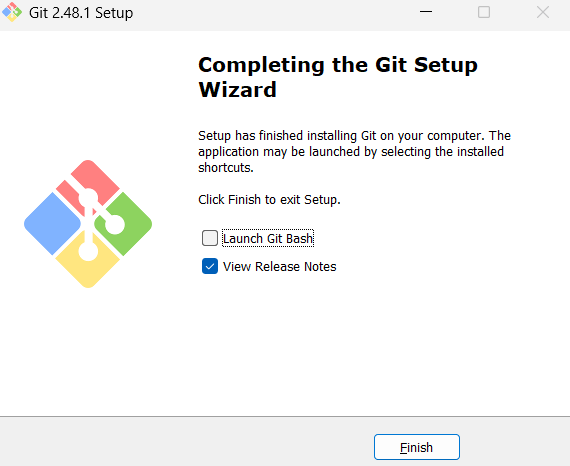
**Figure - 9**

**Step 10: Continue the installation with recommended settings**

Once clicking next, continue the installation with recommended settings for smother operation of the application.

**Step 11: Completing the Git Set - Up Wizard**

Once the installation is complete, **"Completing the Git Setup Wizard"** screen appears. Check the ‘**Launch Git bash’** option and Click on **Finish**.

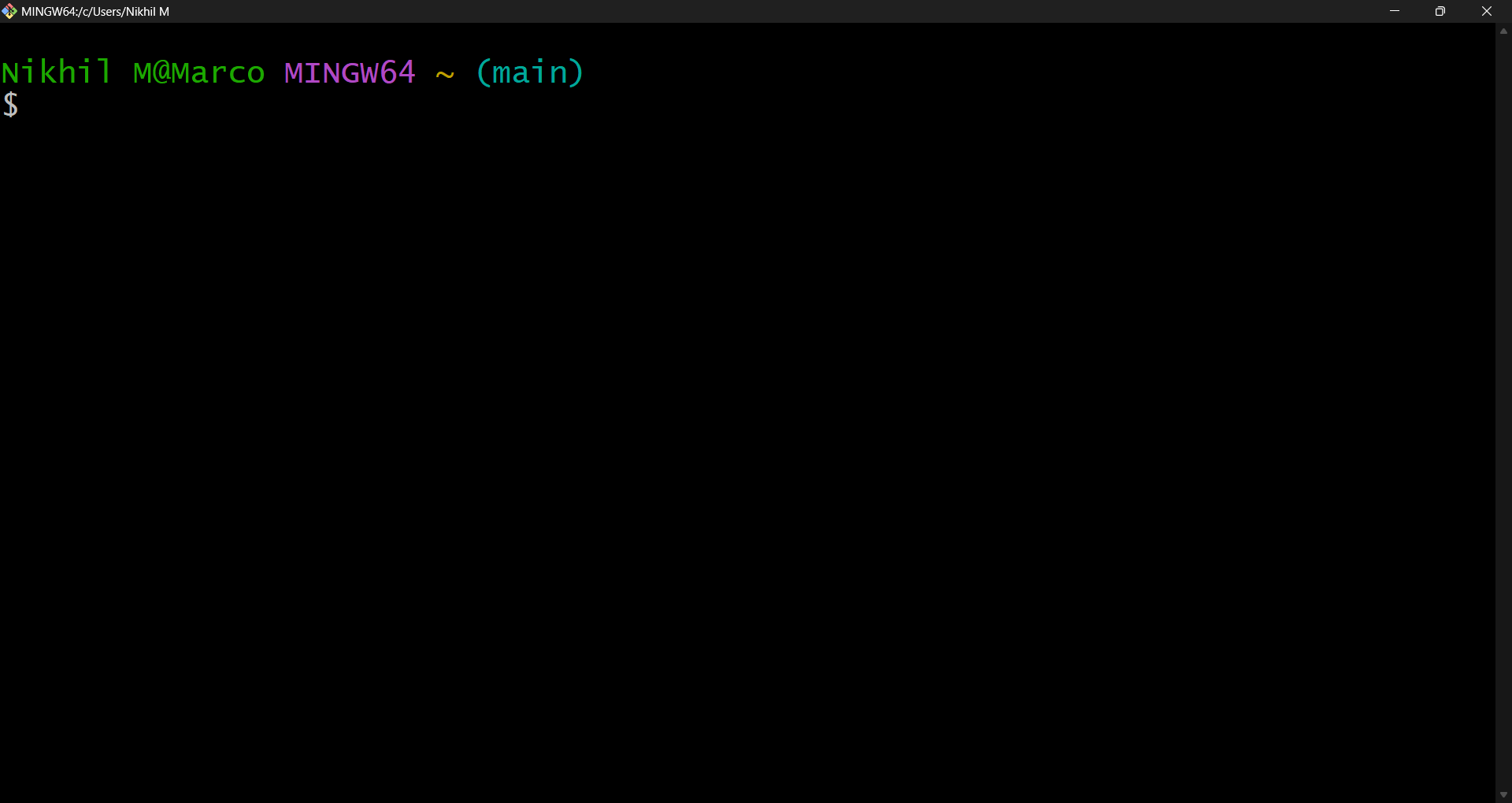


**Figure – 11**

**LAB SESSION 2**

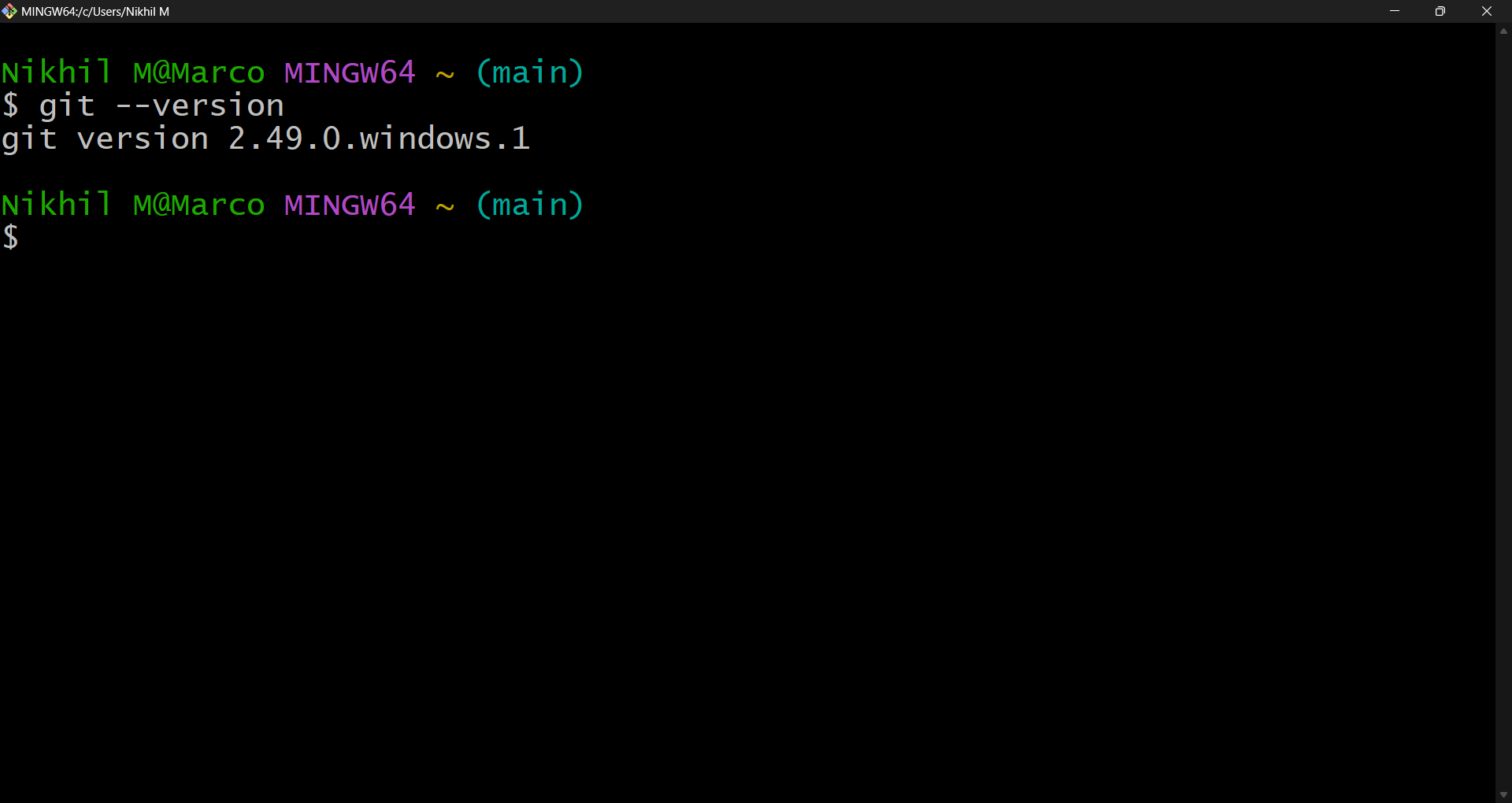
**Step 1: Open Git Bash**

Open **Git Bash** from the Start menu or by searching for it.



**Figure – 1**

**Step 2: Check Git Version**



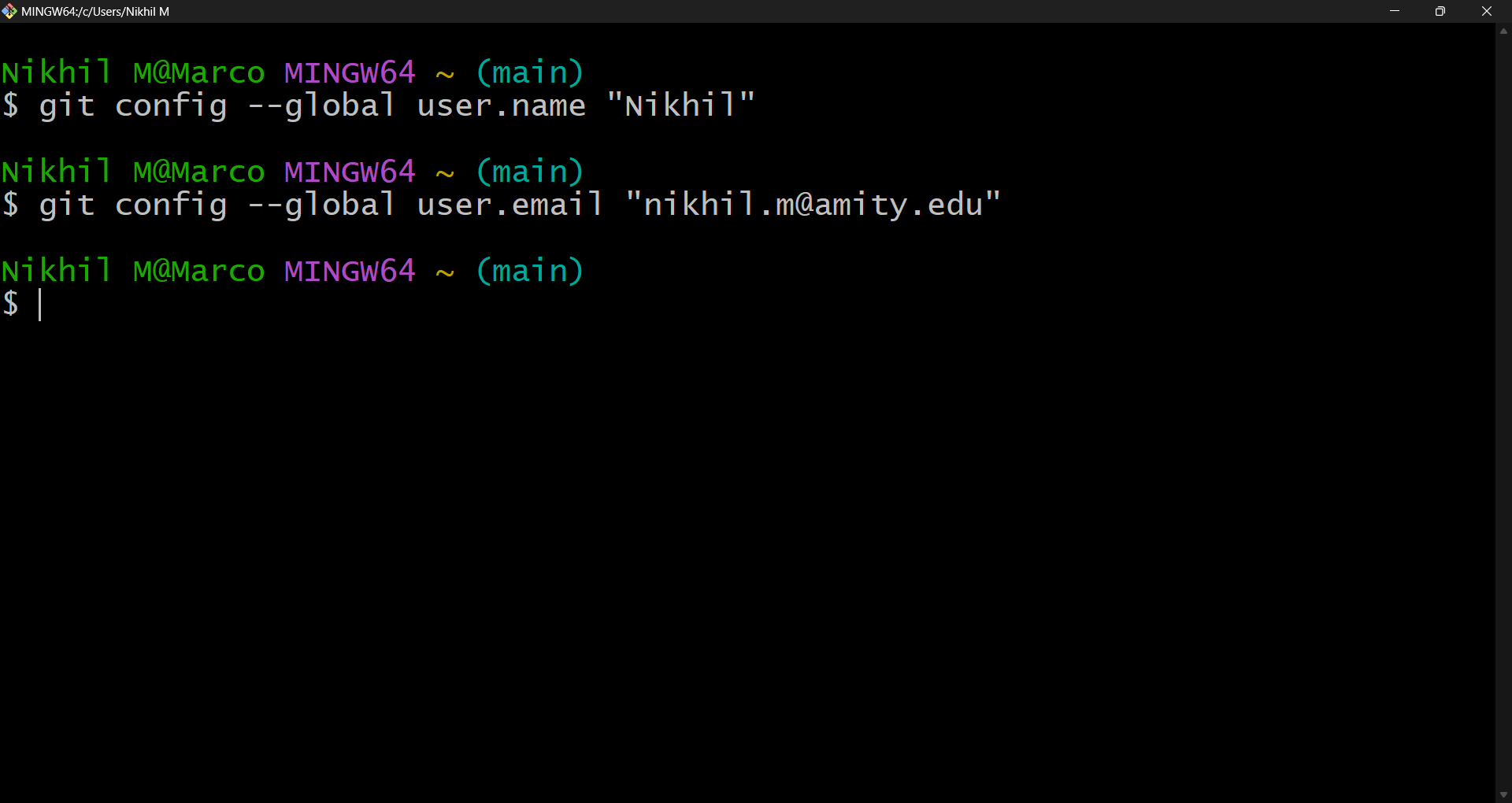
To verify that Git is installed correctly, run: **git –version**

**Figure – 2**

**Step 3: Configure Git**

Set up your Git username and email (required for commits):

* **git config --global user.name "Your Name"**
* **git config --global user.email “**[**your-email@example.com**](mailto:your-email@example.com)**”**

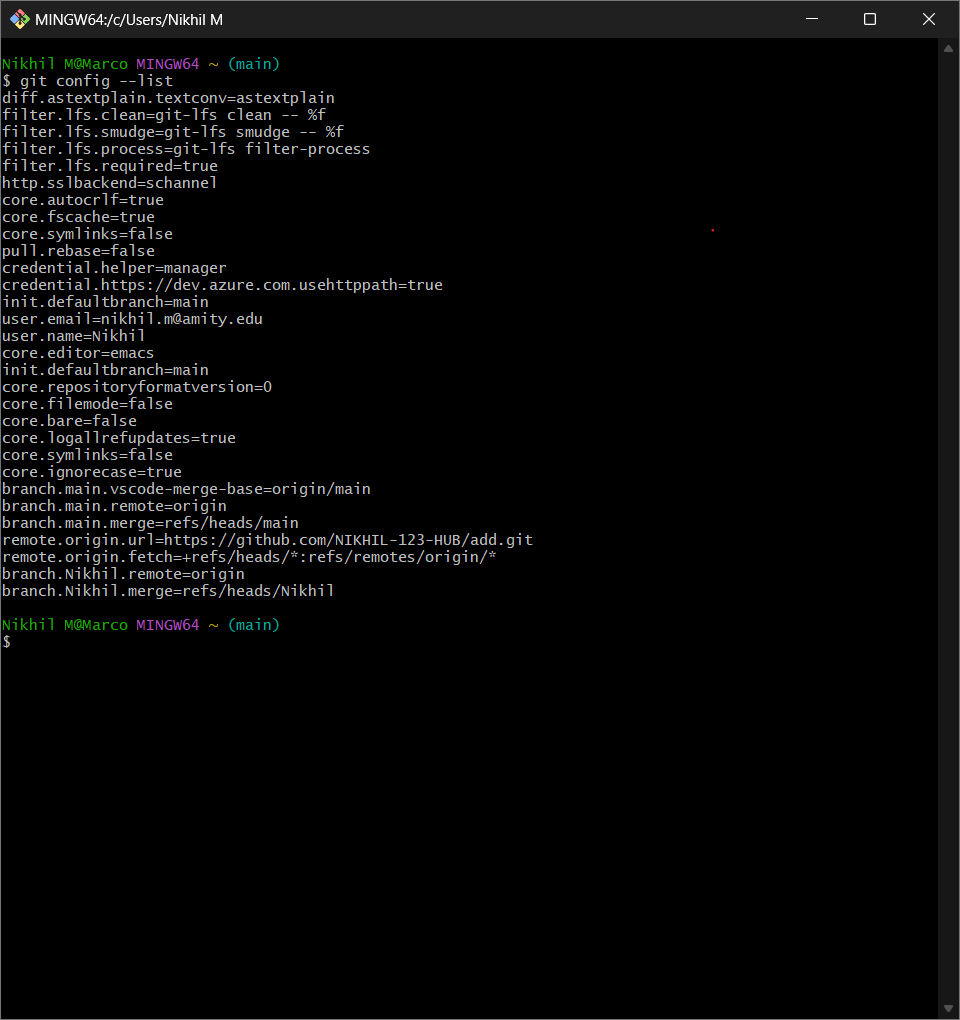
****

**Figure – 3**

**Step 4: Verify Git Configurations**

To check if the configurations were set correctly, run:

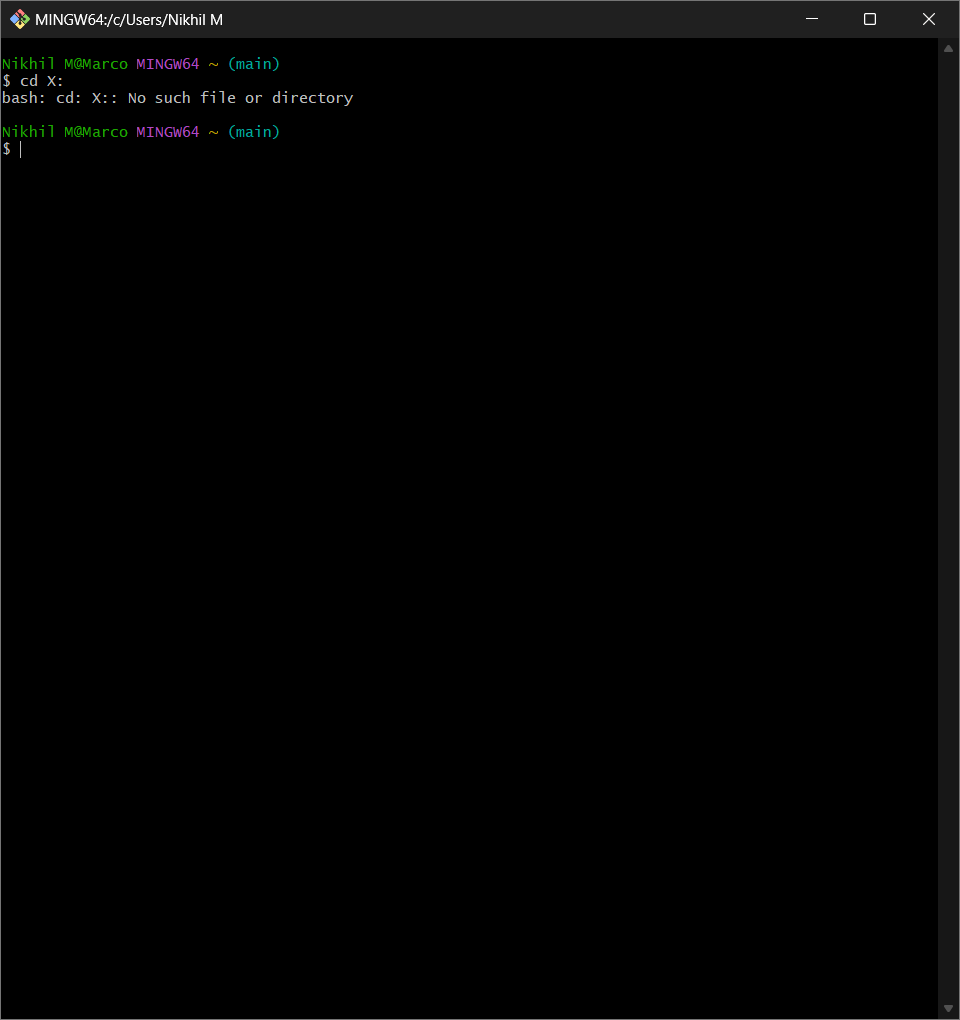
* **git config --list**

****

**Figure – 4**

**Step 5: Change Directory**

Change directory (**cd**) to your preferred location using the **‘cd’** command.



**Figure – 5**

**Step 6: Print the Current Directory**

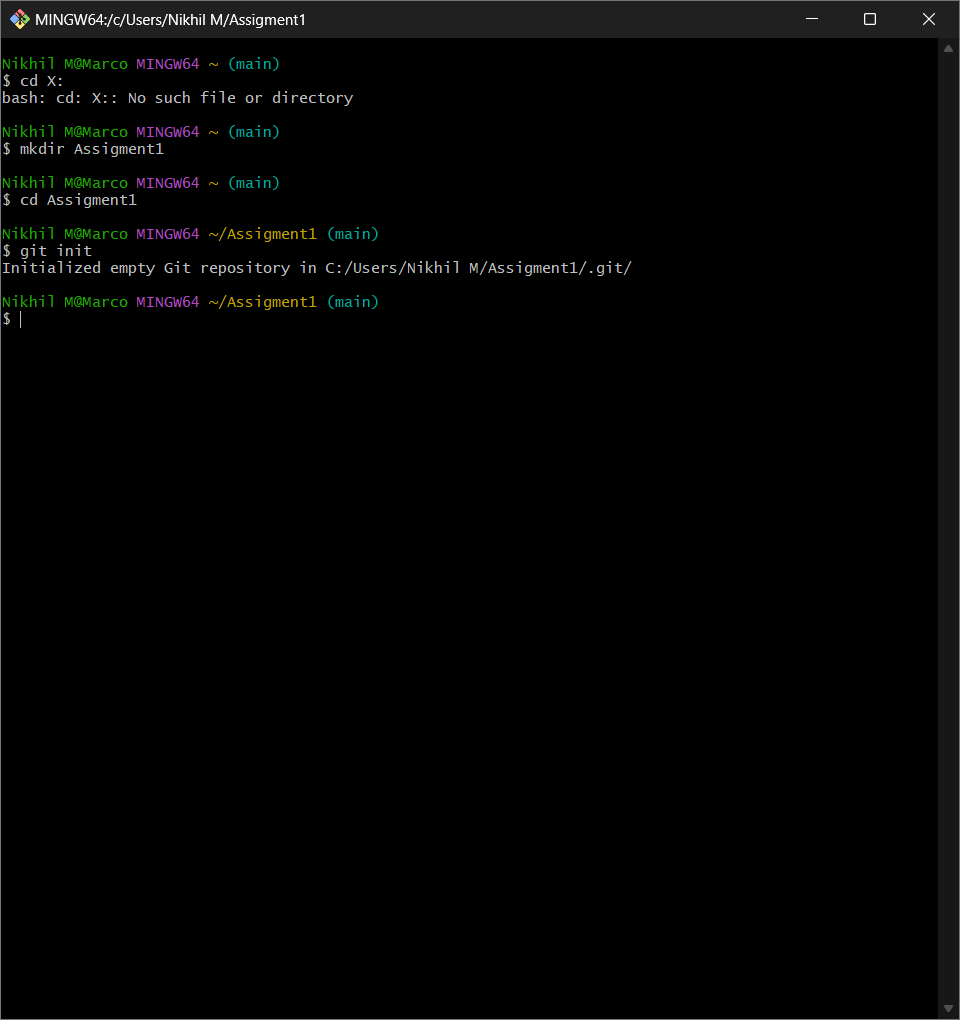
To print the full path of your current Directory use the ‘**pwd’** command.



**Figure – 6**

**Step 7: Create a New Folder**

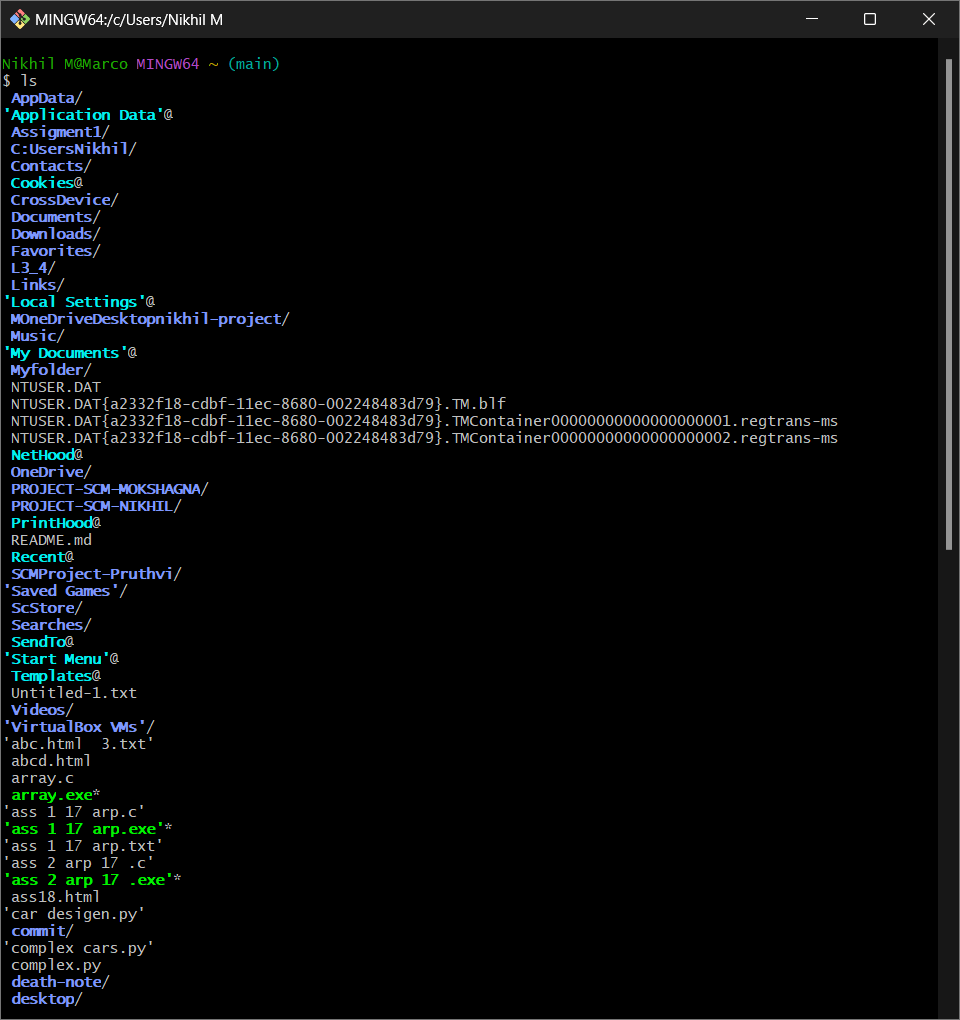
To Create a new folder in the Directory, use the command: **mkdir** folder-name.

****

**Figure – 7**

**Step 8: Listing the Files and Folders**

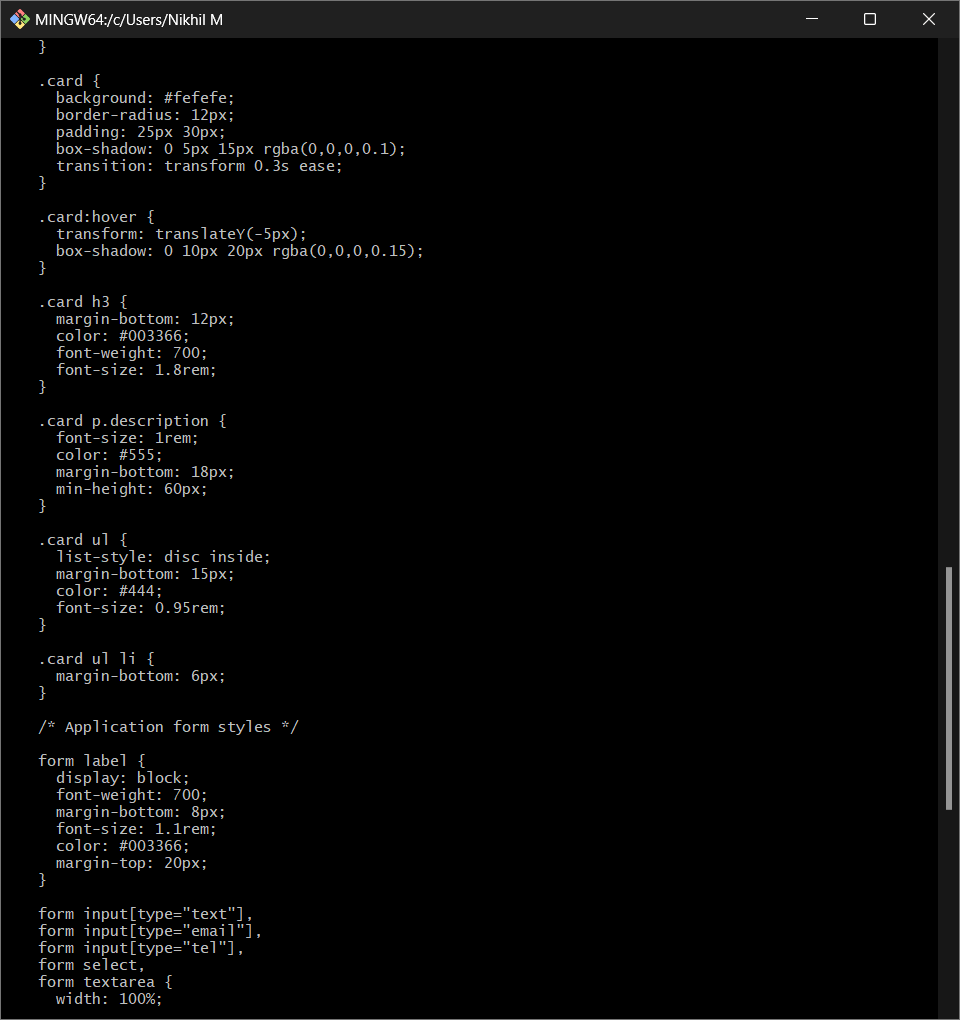
To Display the list of all files and folders in the current directory use the **‘ls’** command.

****

**Figure – 8**

**Step 9: Inside the VI Editor**

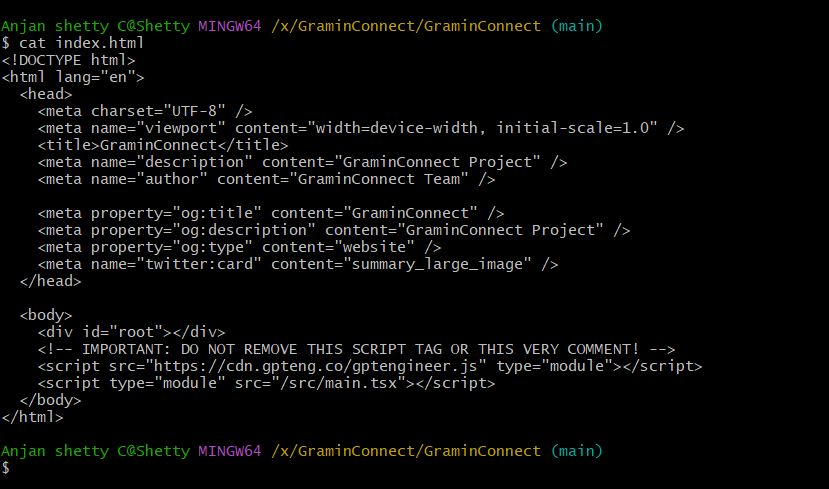
Once typed Git opens the **‘vi’** editor to create or edit a file named **Hello.cpp.** Press **i** to enter **INSERT** mode. Now start typing your code in the **vi** Editor.



**Figure – 9**

**Step 10: Display File Contents**

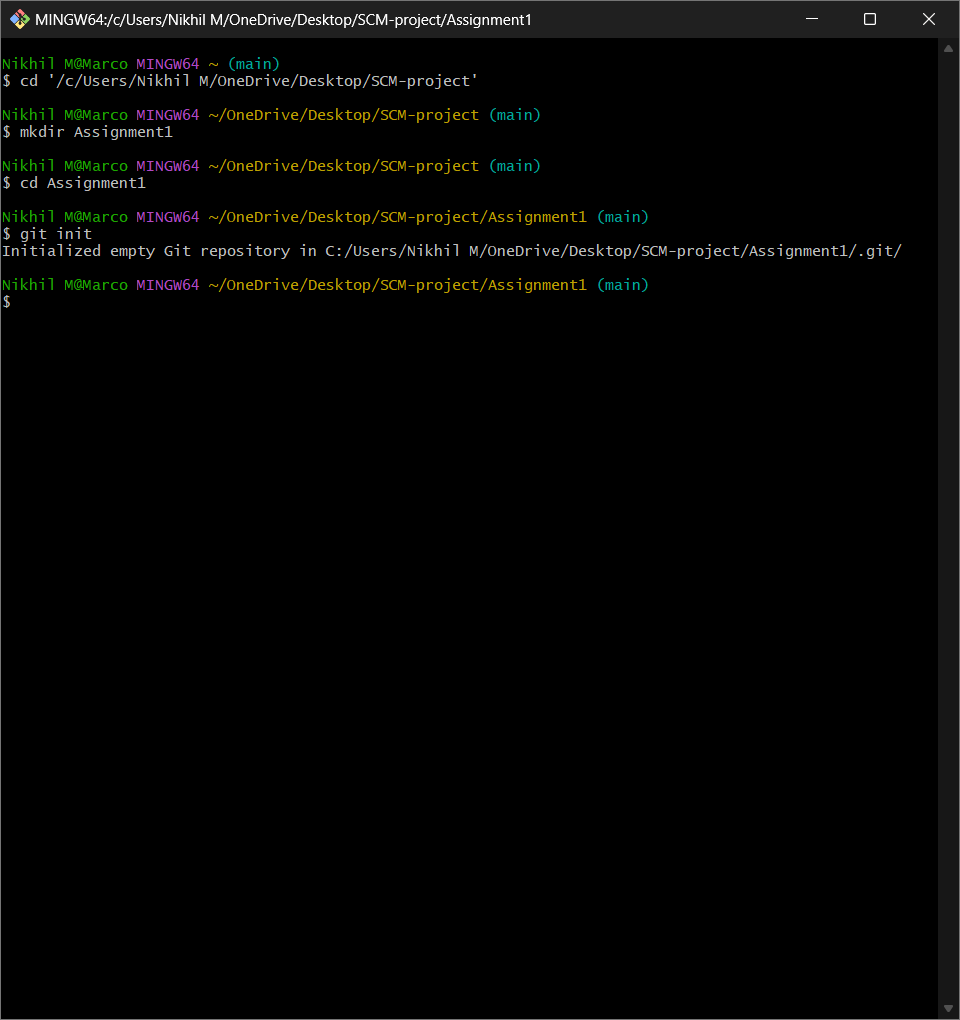
To Display the contents of the CPP File use the **cat** Command as: **cat** filename.extension.

****

**Figure – 10**

**Step 11: Initialize Git in Directory**

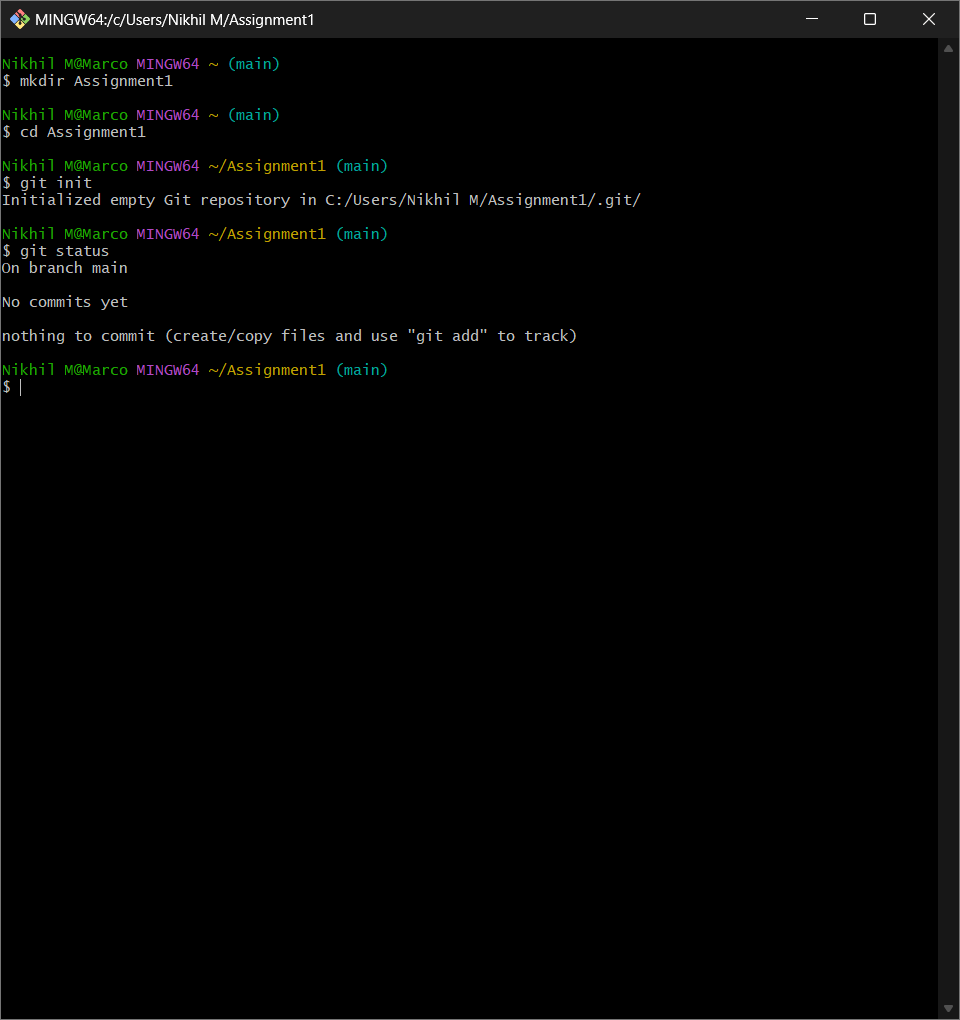
To turn the directory into a Git repository, run: **git init**



**Figure – 11**

**Step 12: Check Git Status**

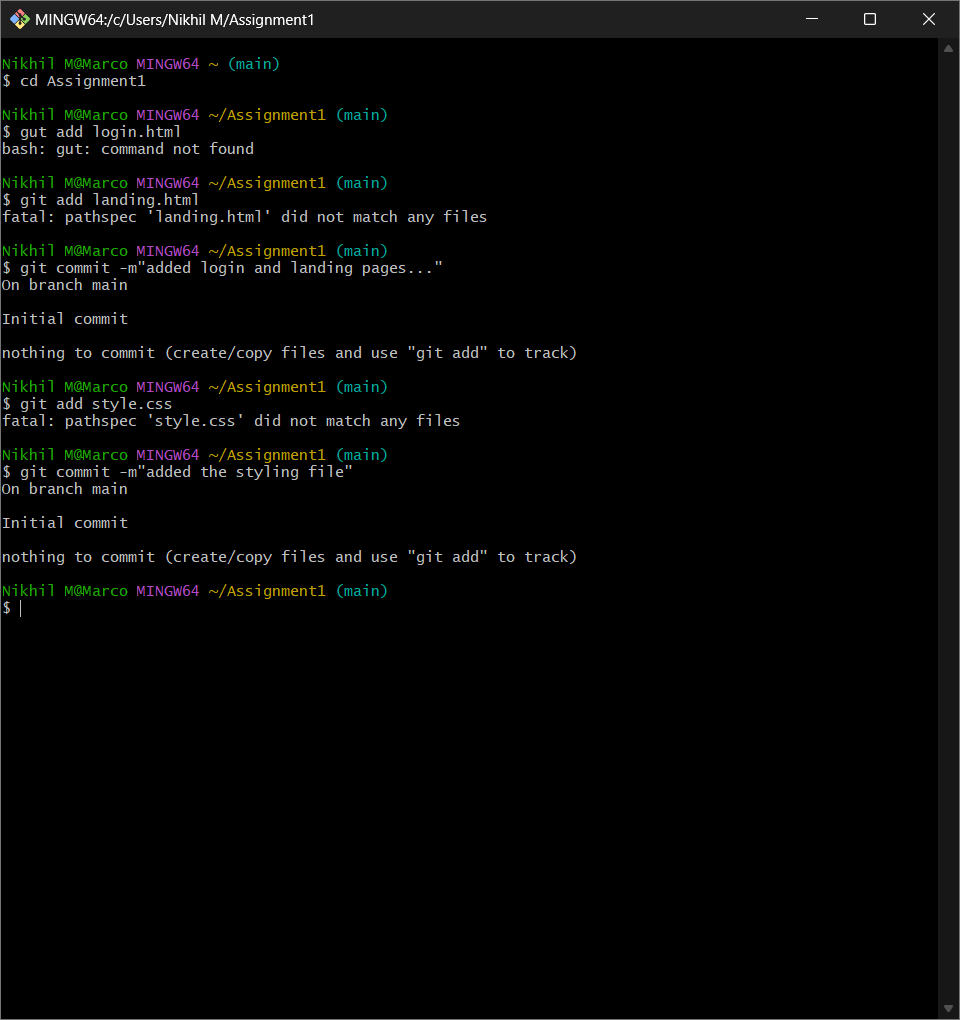
The **git status** command is used to check for **untracked files**, along with other changes in the repository.



**Figure – 12**

**Step 13: Commit the File**

To save the changes in Git, commit the file with a message**: git commit -m "Added the styling file"**

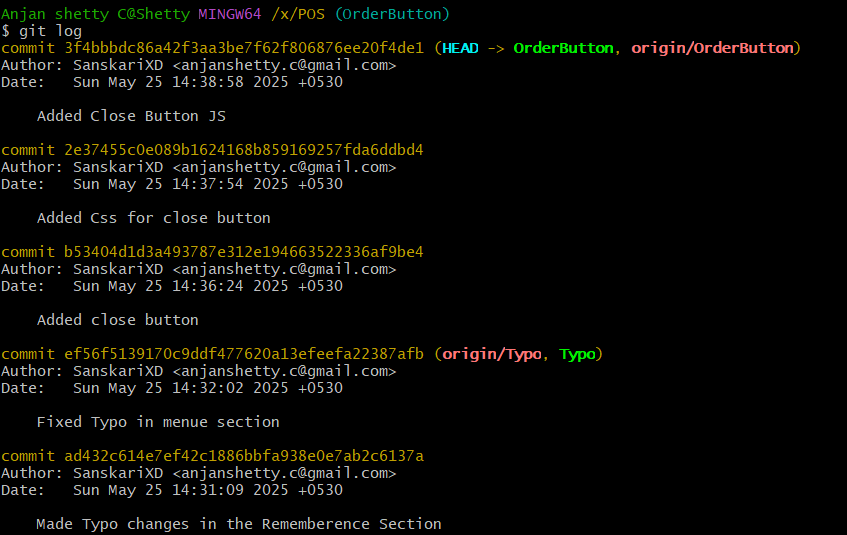
****

**Figure – 13**

**LAB SESSION 3**

**Step 1: Check Git Commit History**

 The **git log** command displays the commit history in detail.

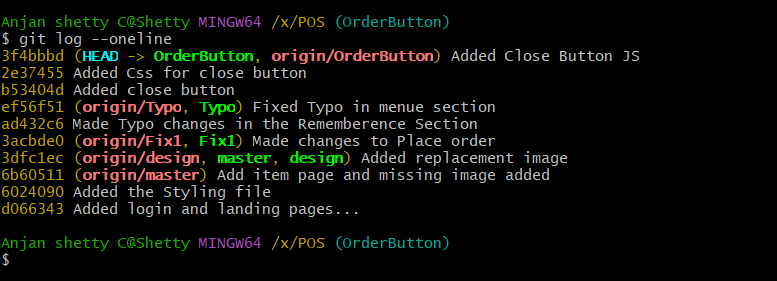
 It shows the commit hash, author, date, and commit message.

**Figure – 1**

**Step 2: View Git Log in One Line Format**

 The **git log --oneline** command displays a compact version of the commit history.

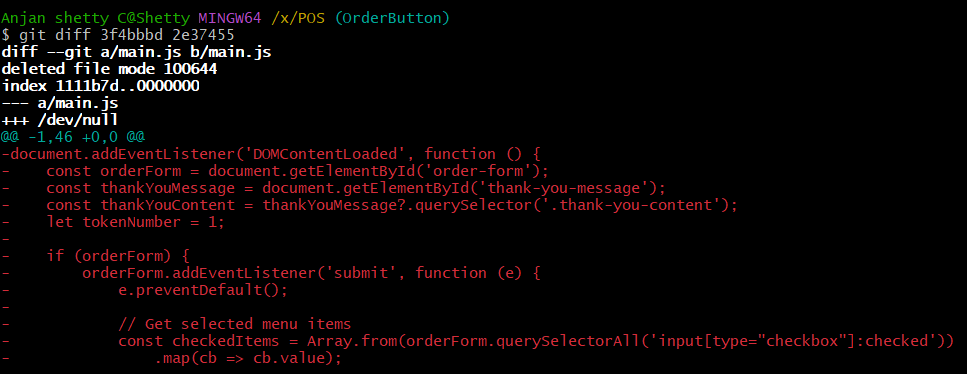
 It only shows the commit hash and the commit message.



**Figure – 2**

**Step 3: View Differences Between Commits**

The **git diff** command shows the exact lines changed between each commits. You can compare between multiple commits.

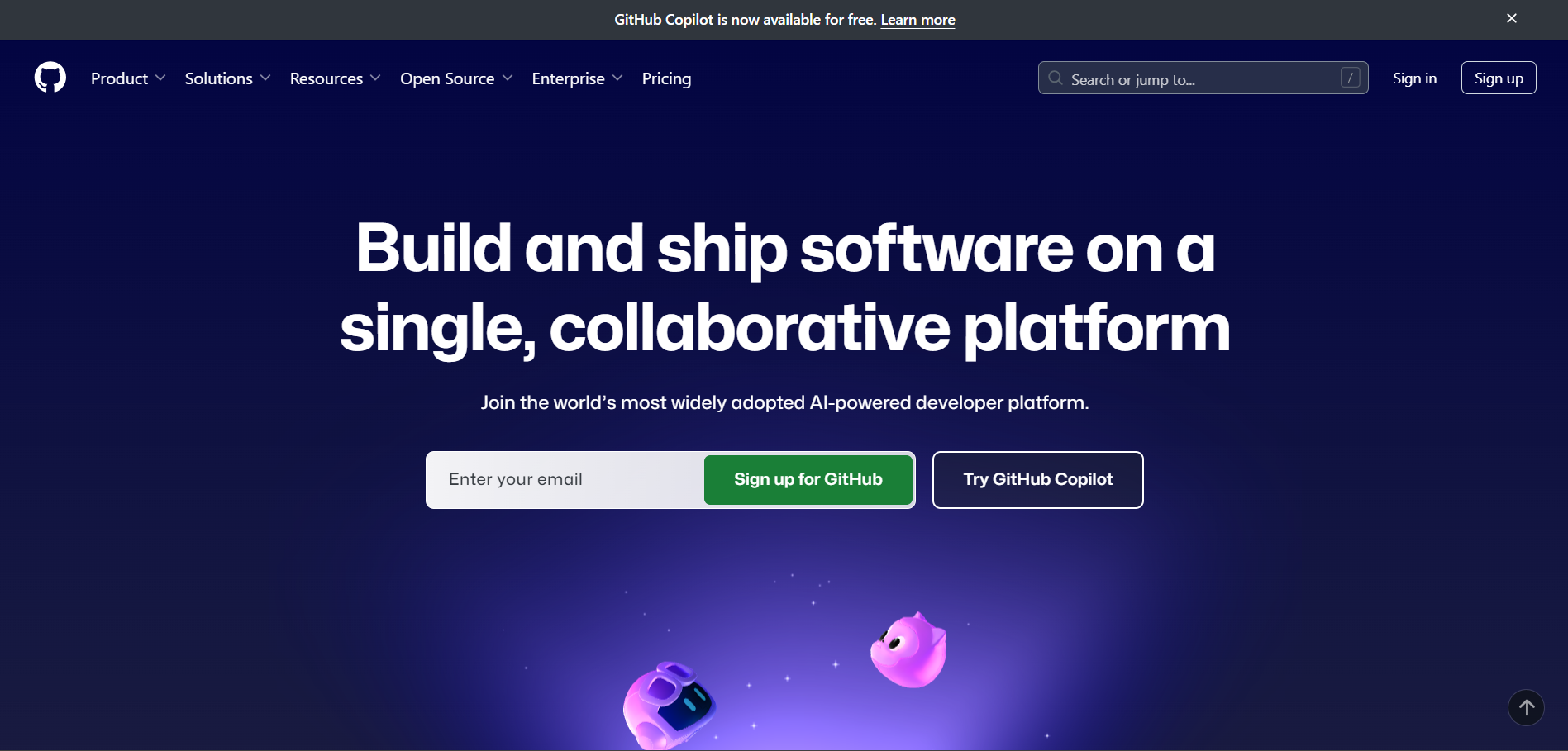
This shows changes between the 3f4bbbd and 2e37455

**Figure – 3**

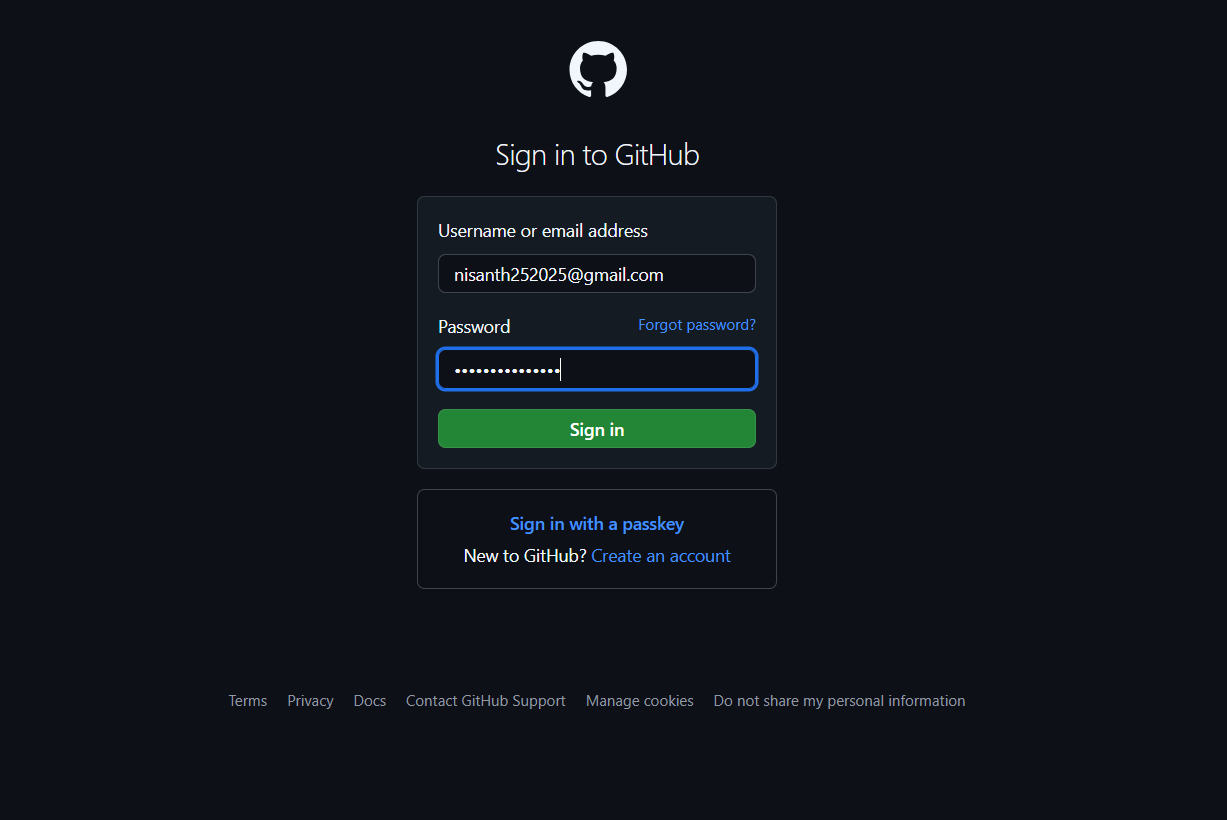
**LAB SESSION 4**

**Step 1: Sign in to GitHub**

Open a web browser and go to [github.com](https://github.com/)



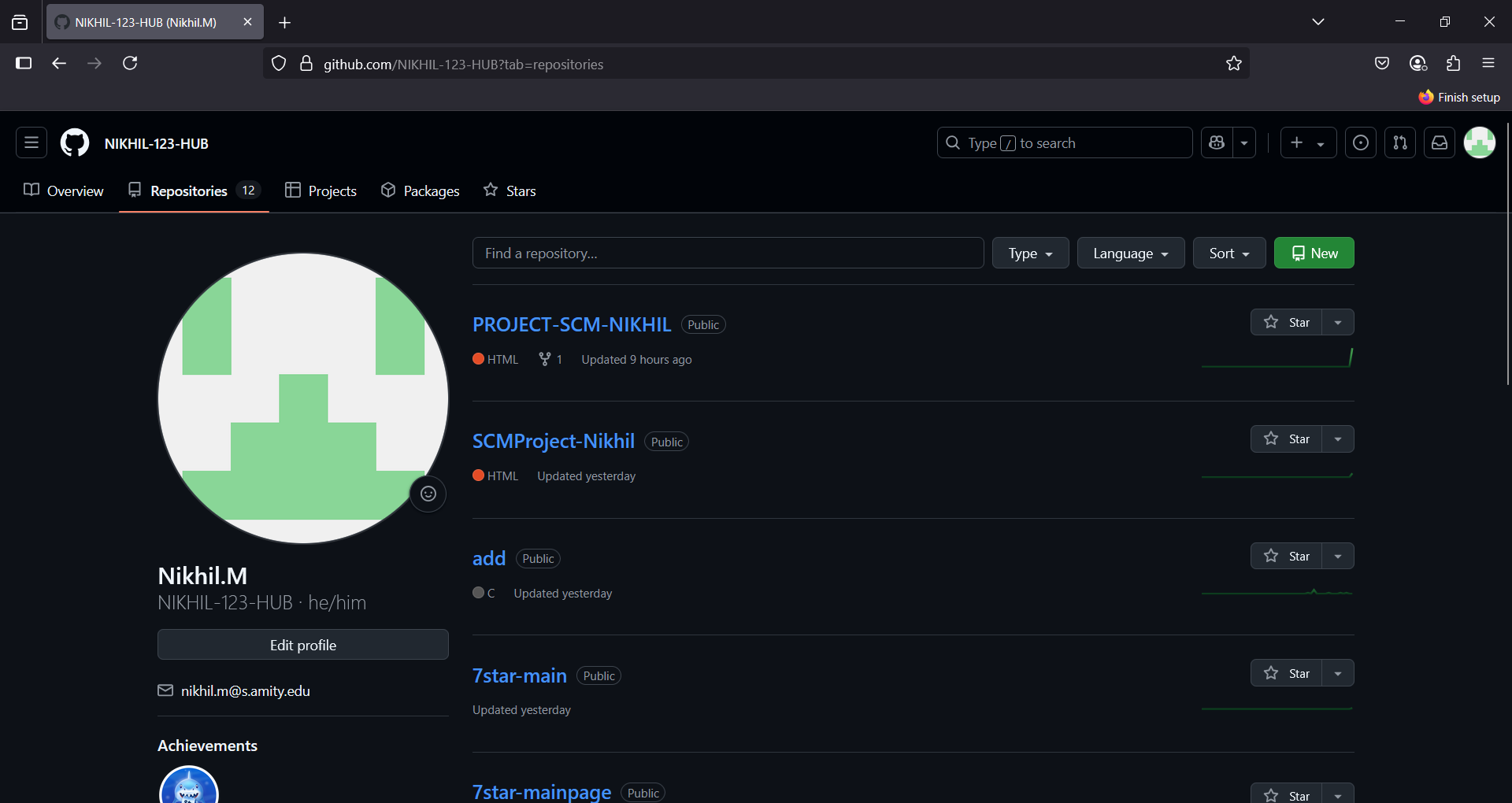
**Figure – 1**

Click Sign in and enter your credentials.

**Figure – 2**

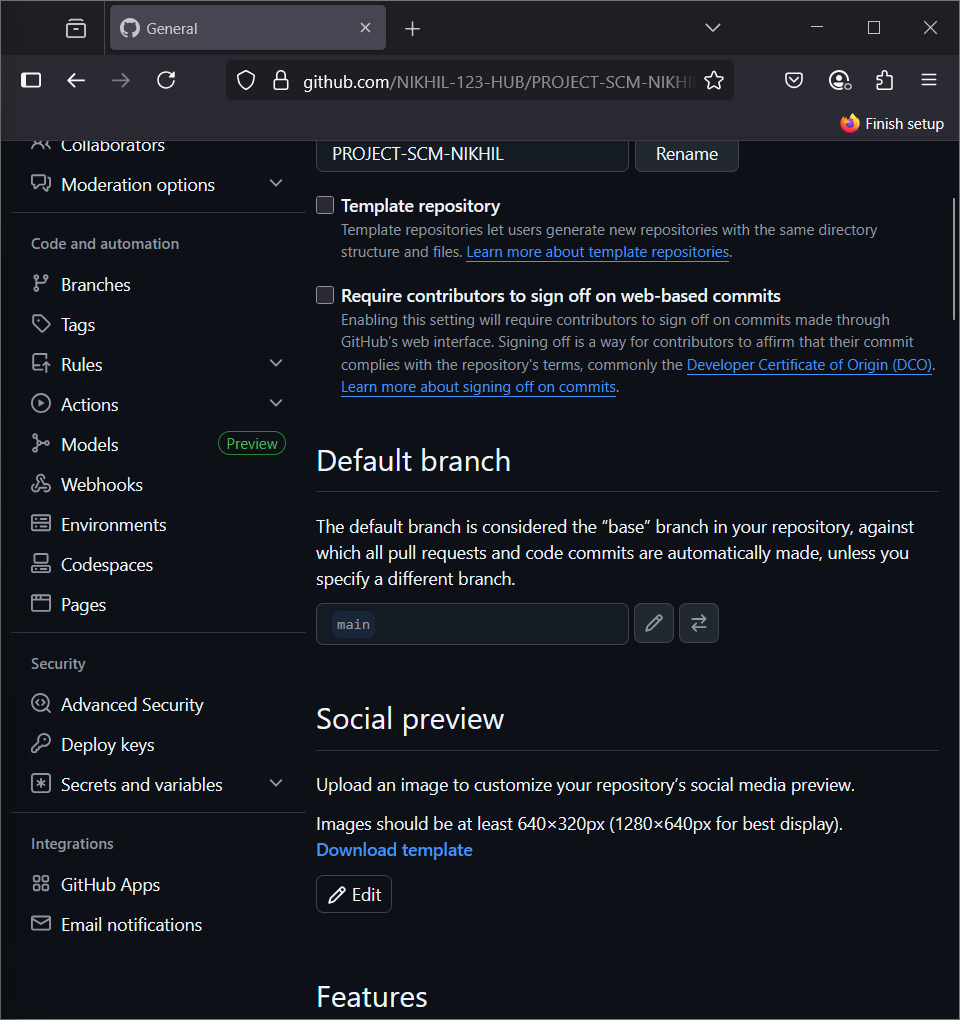
**Step 2: Creating a Repository**

Click on the **NEW** icon (top-right corner).



**Figure – 3**

In the **Repository name** field, enter the same name as your local folder.   
Click **Create repository**.

****

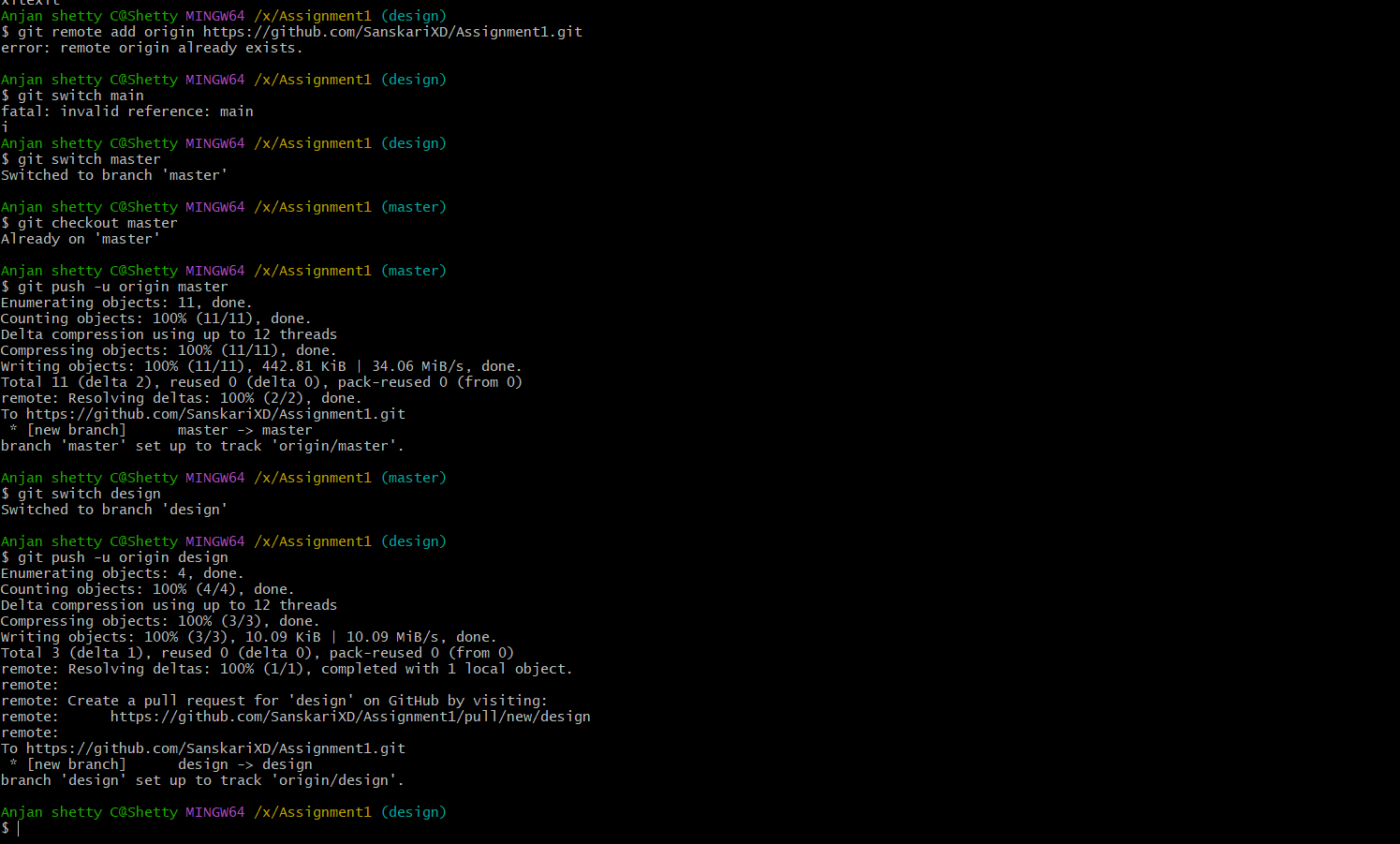
**Figure – 4**

**Step 3: Connect Local Repository to GitHub**

On the next page, copy the **HTTPS URL** under **"Quick setup"** it looks like (**https://github.com/yourusername/repositoryname.git**).

Add the GitHub repository as a remote:

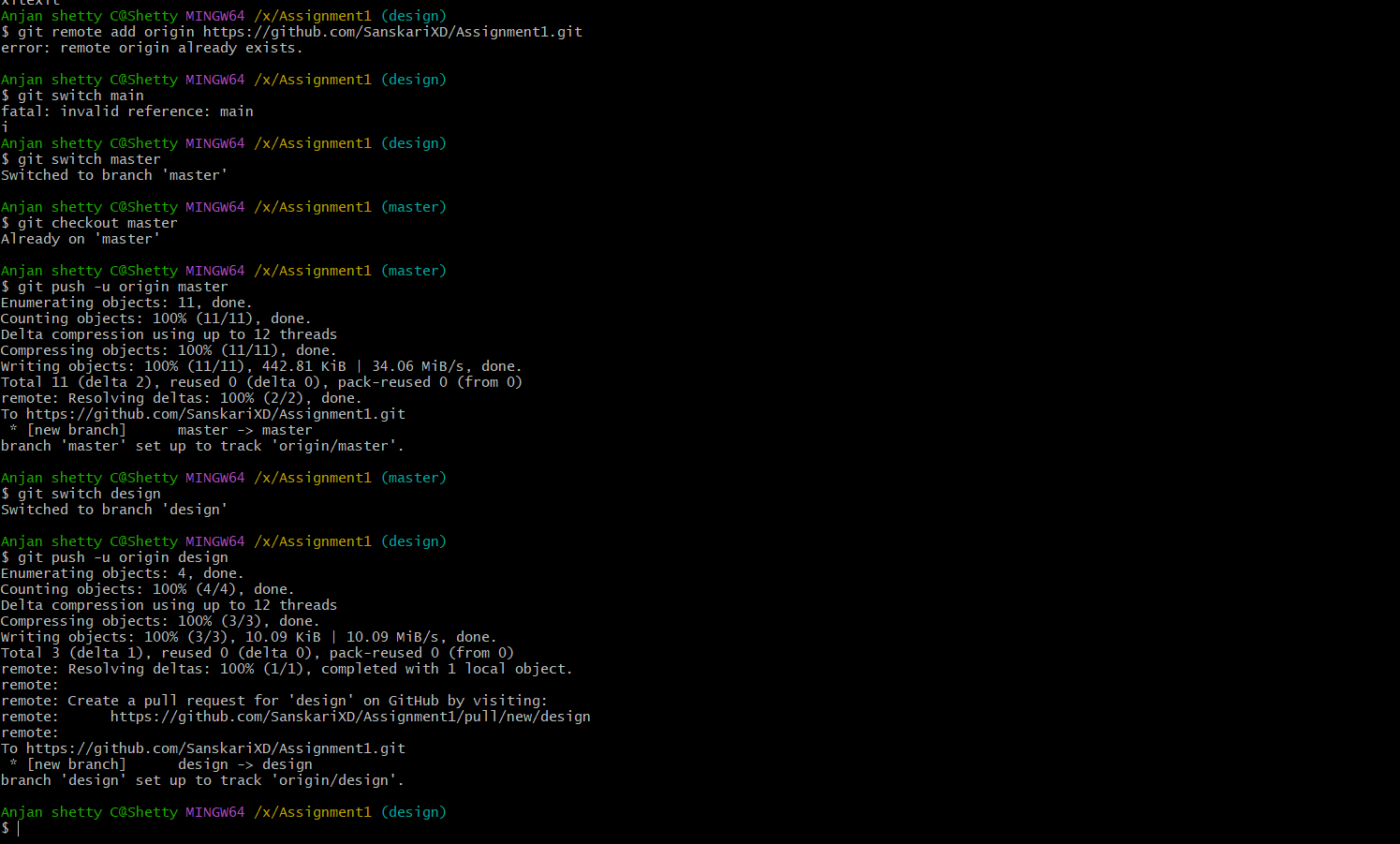
* **git remote**
* **git remote add origin <repository-URL>**



**Figure – 5**

**Step 4: Push Code To GitHub**

Push the committed files to GitHub using the command: **git push -u origin master**

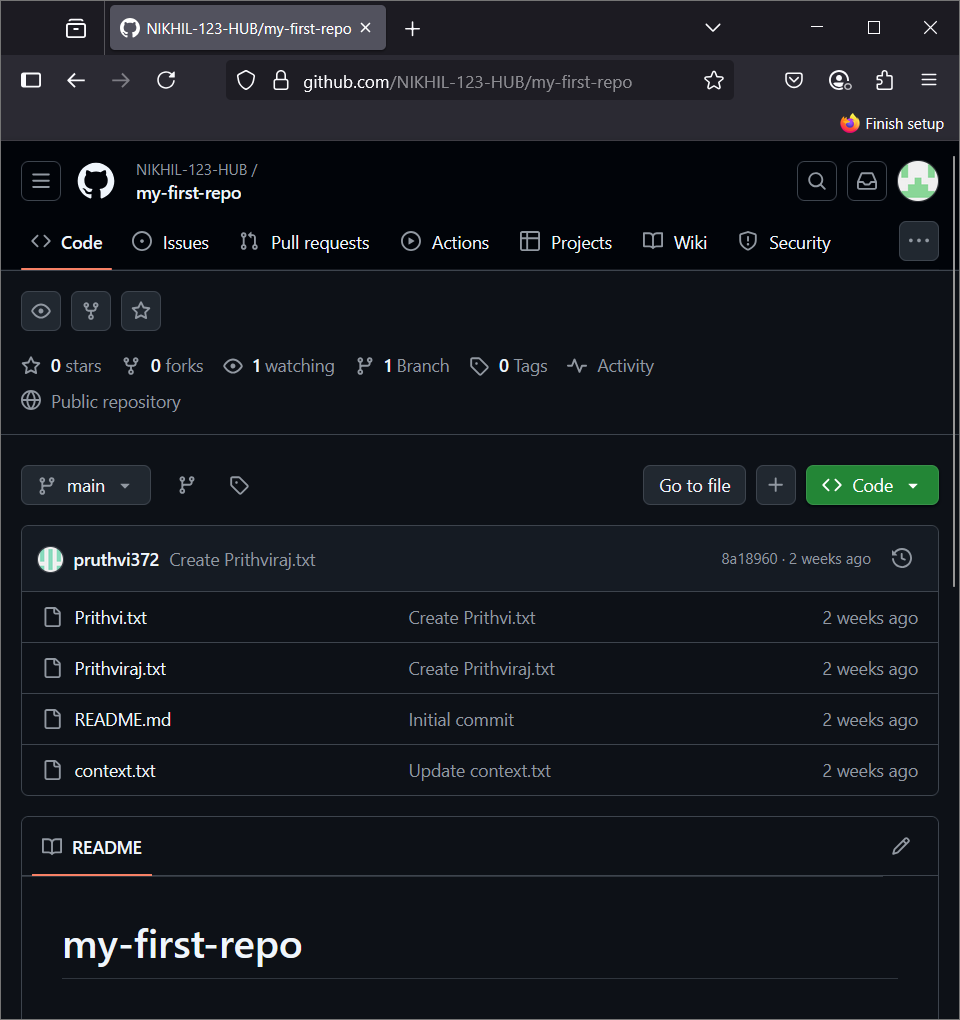


**Figure – 6**

**Step 5: Verify Changes on GitHub**

1. Open **GitHub** in your browser.

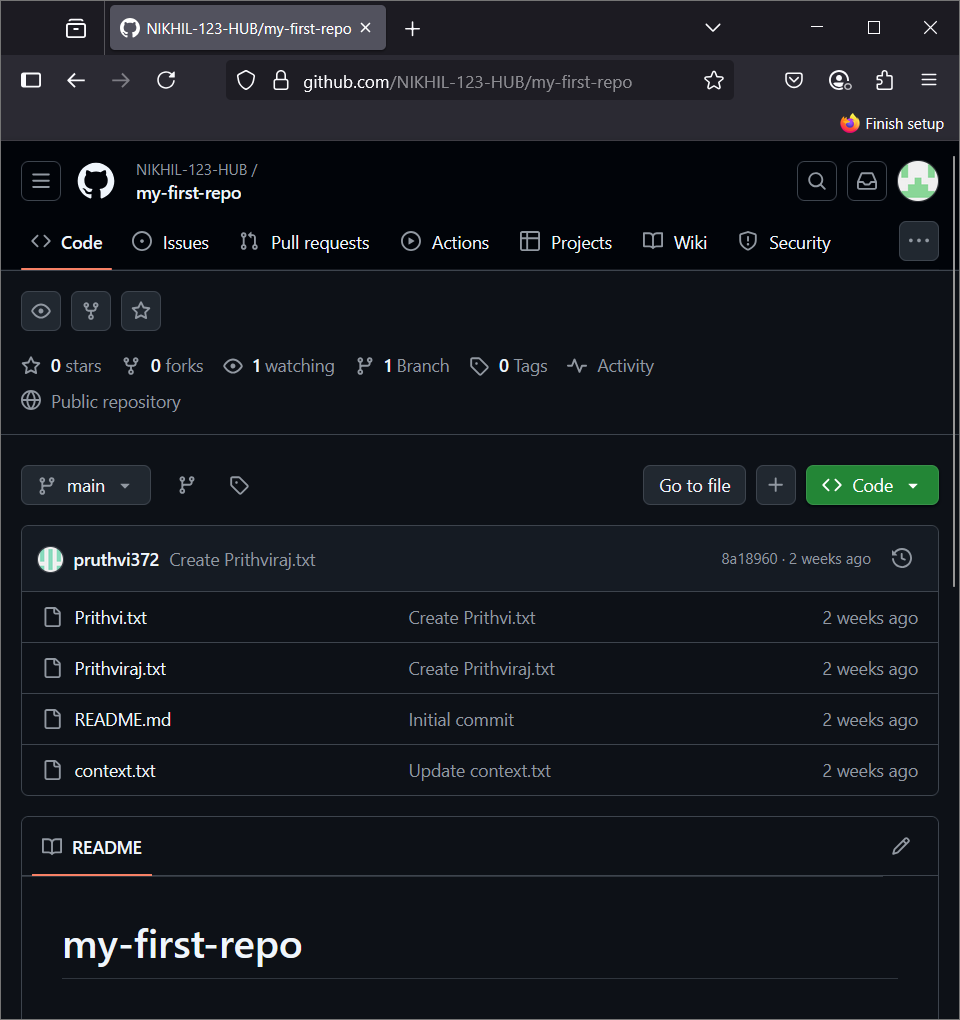
2. Go to your repository.

3. Refresh the page – your files should be visible in the repository.

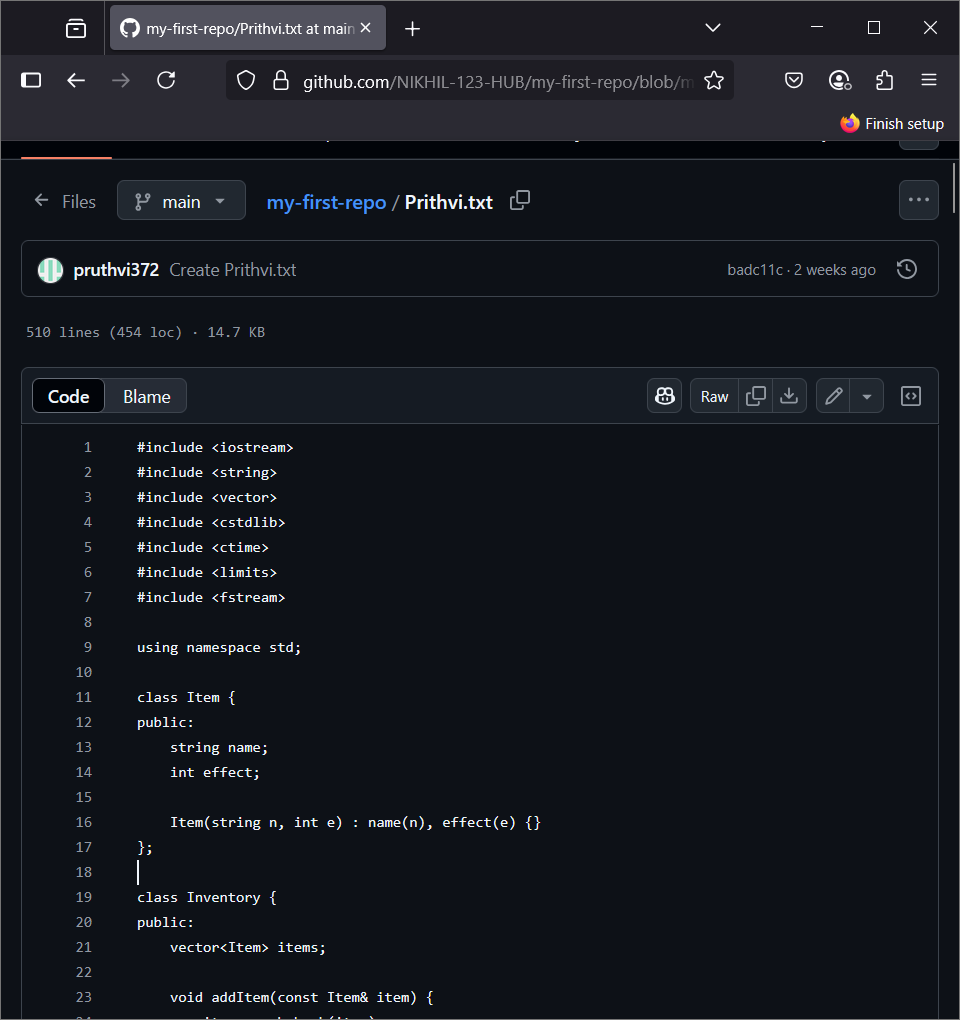
**Figure – 7**

**Step 6: Edit the File Directly on GitHub**

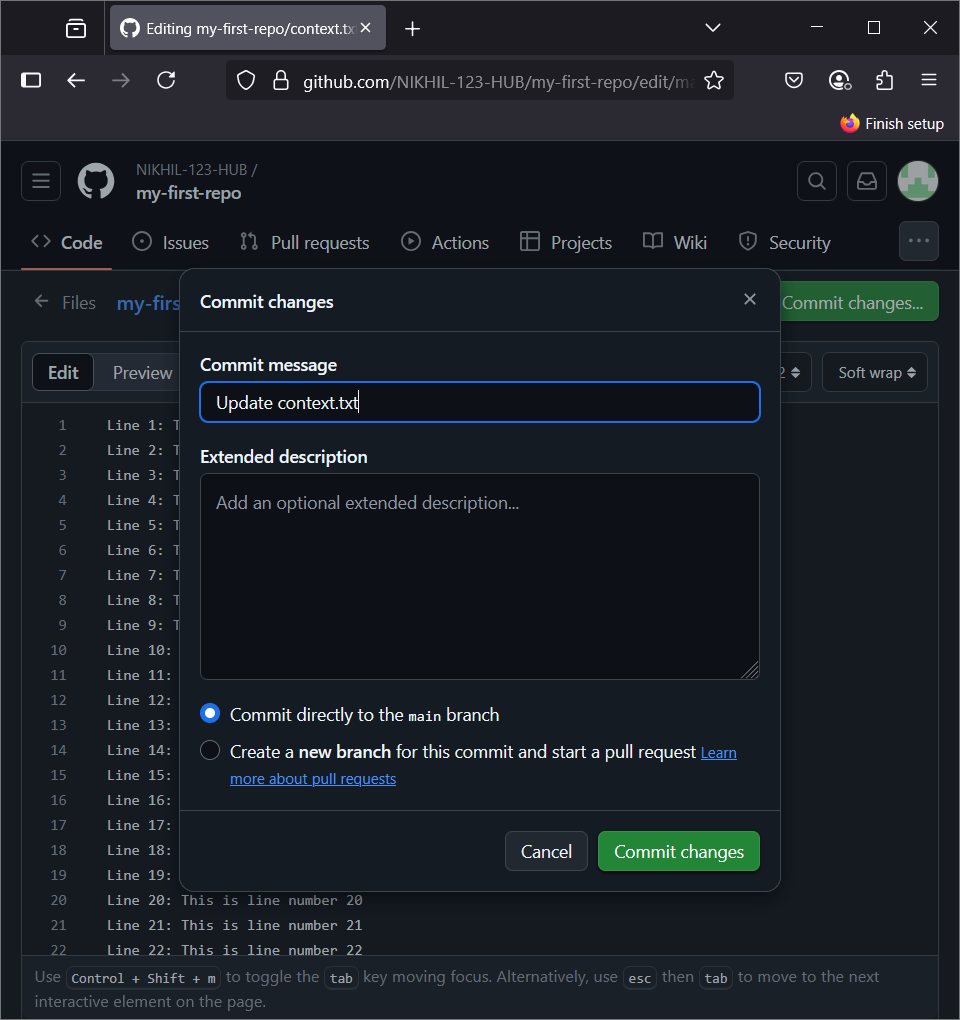
1. Open any file in your GitHub repository.

****

**Figure – 8**



**Figure – 9**

3. Make some changes to the file, scroll down, enter a commit message, and click Commit changes.

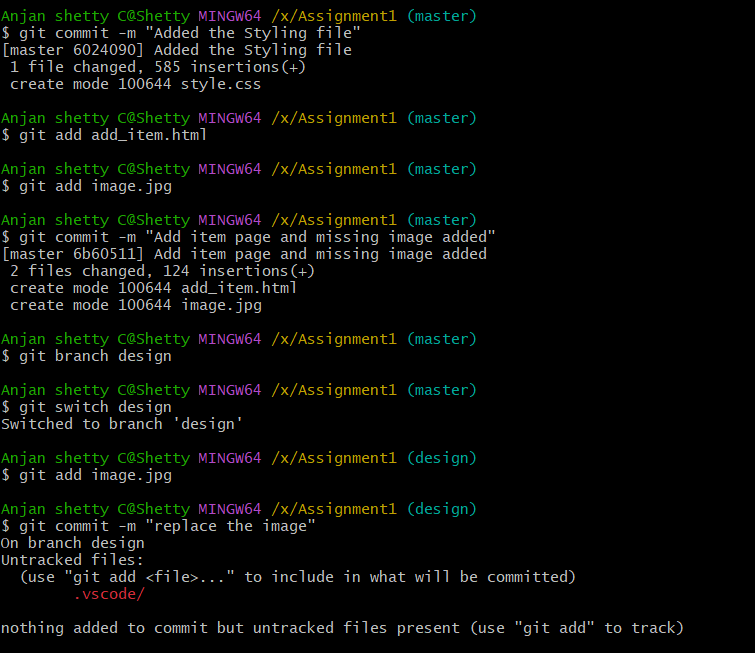
**Figure – 10**

**LAB SESSION 5**

**Step 1: Create a New Branch**

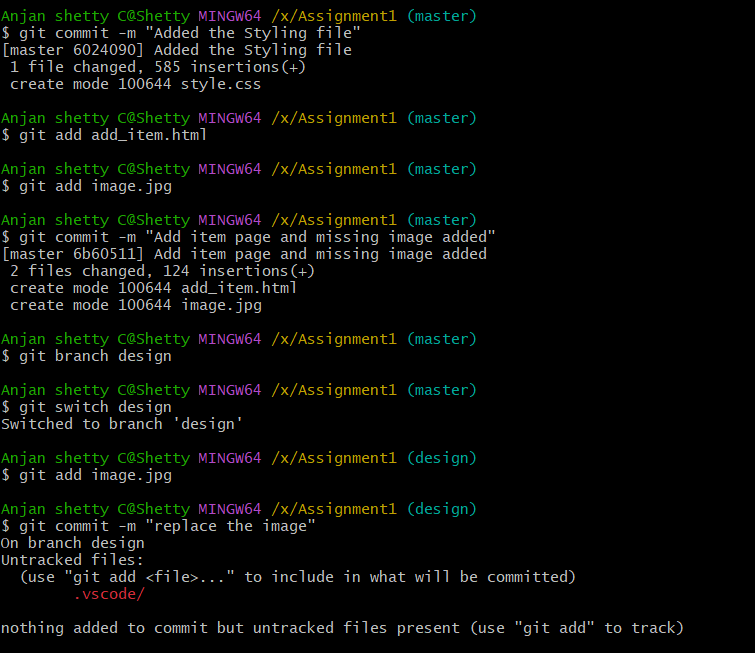
Use the following command to create a new branch named **design** and switch to it:

**git branch design**



**Figure - 1**

**Step 2: Make Changes in the design Branch and commit**

Switch to design branch and add image file then commit with message.

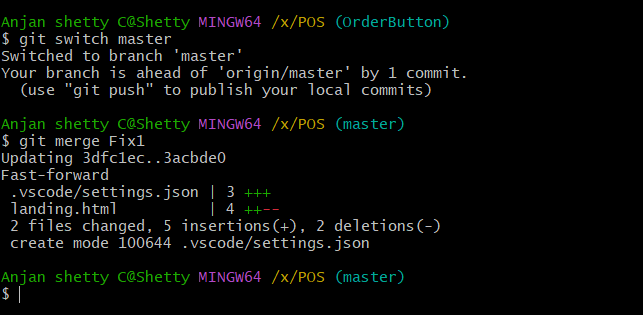
**Figure – 2**

**Figure – 3**

**Step 3: Switch Back to master Branch and merge a branch**

**git switch master**

If there are no conflicts, this will merge the changes from the Fix1 branch into master.

****

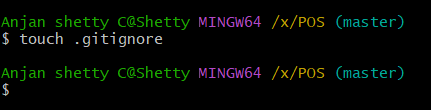
Nikhil m@Marco

**Figure – 3**

**Step 4: Creating a .gitignore File**

The **.gitignore** file tells Git to ignore specific files or directories that do not need to be tracked, such as log files, build directories, or system files.

**touch .gitignore**



**Figure – 4**

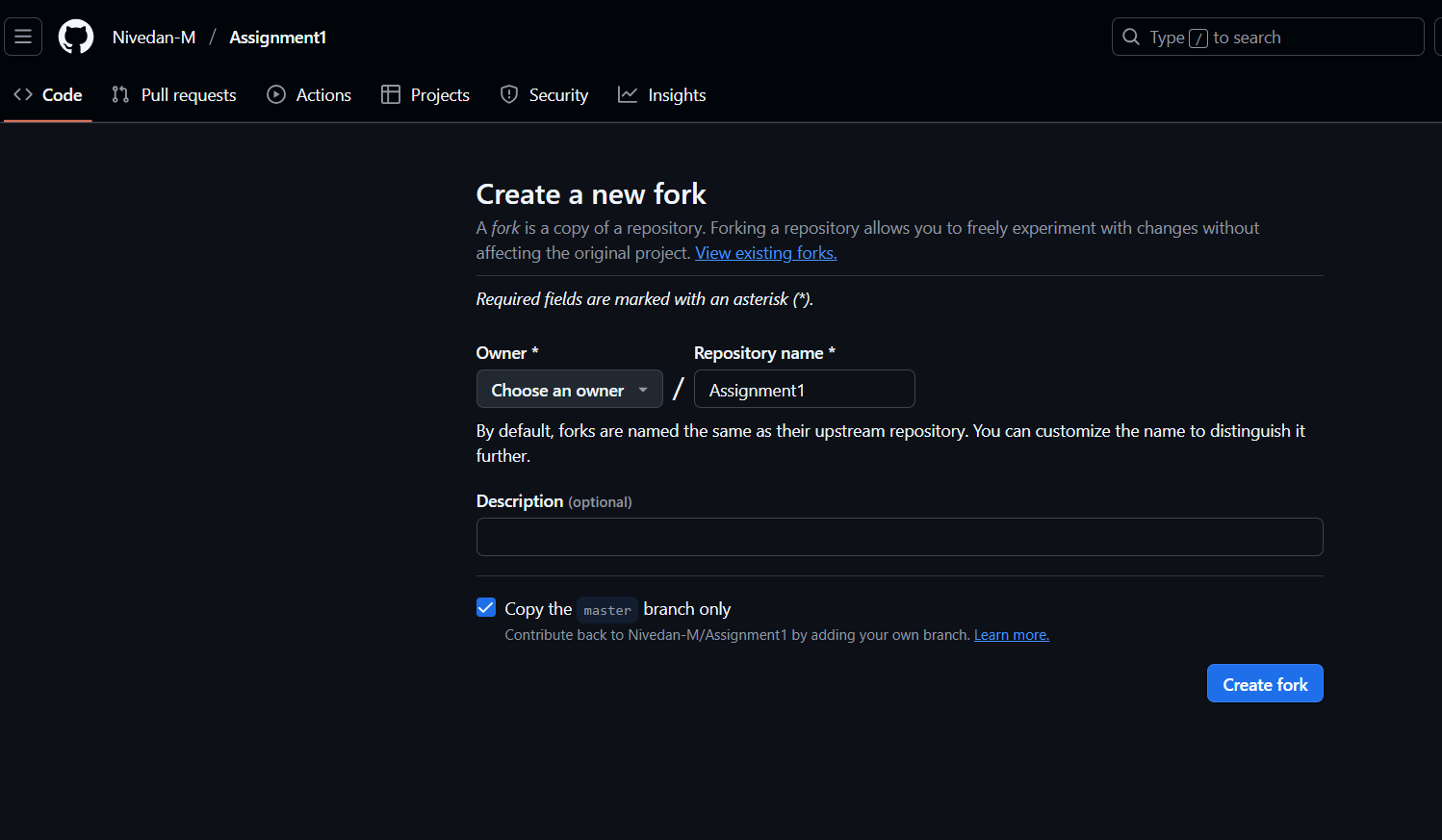
**LAB SESSION 6**

**Step 1: Fork a Repository on GitHub**

 Go to any public repository on GitHub (e.g., github.com/Nivedan-M/Assignment1)

 Click on the **"Fork"** button (top right corner).

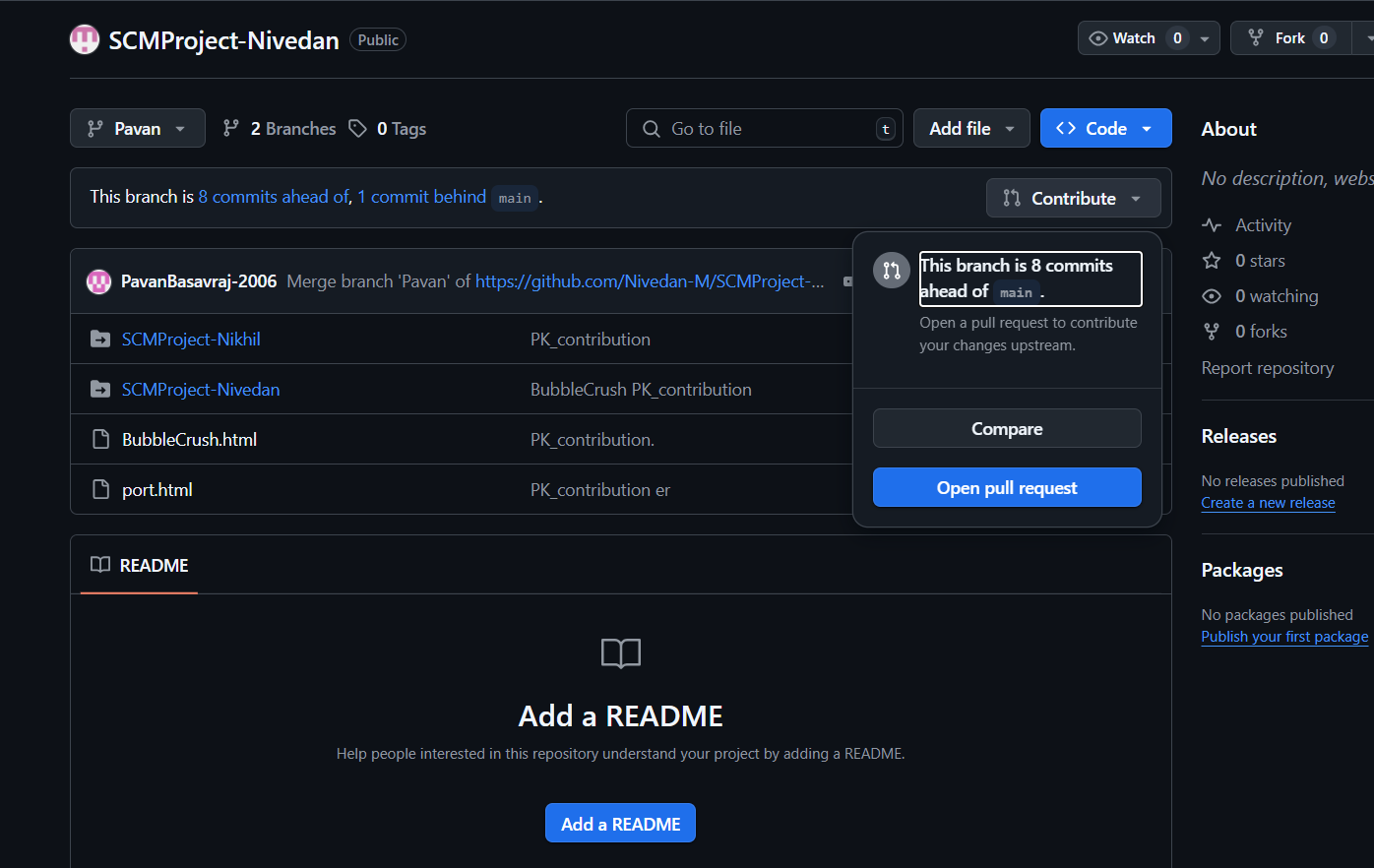
 This creates a copy of the repository under **your GitHub account**.

**Figure - 1**

**Step 2: Create a Pull Request**

If you want your changes to be added to the original repository:

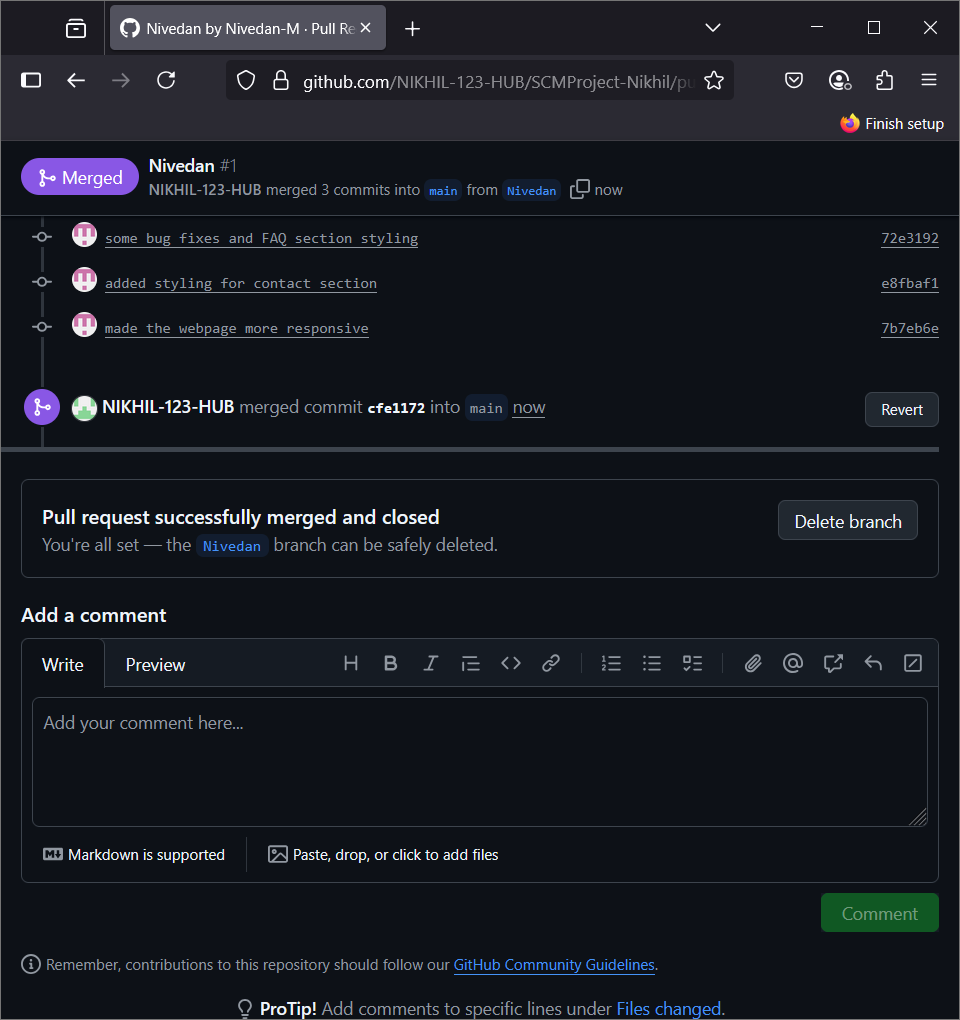
1. Go to a repo on GitHub.
2. Click **"Contribute" > "Open Pull Request"**.
3. Submit your pull request for review.



**Figure - 2**

**Step 3: Merge the pull request**

If you want your changes done by forked user to be added to your original repository:

1. Open Pull request tab in your Repo
2. Scroll down and verify the code and check if there are not conflict
3. Click on Merge.
4. 

**Figure - 3**