REDDY PRAKASH

Assignment GIT and GITHUB -3

1. L1 - Create Local git repository and demonstrate all git reset options and revert. Compare the differences

1) launch the local terminal

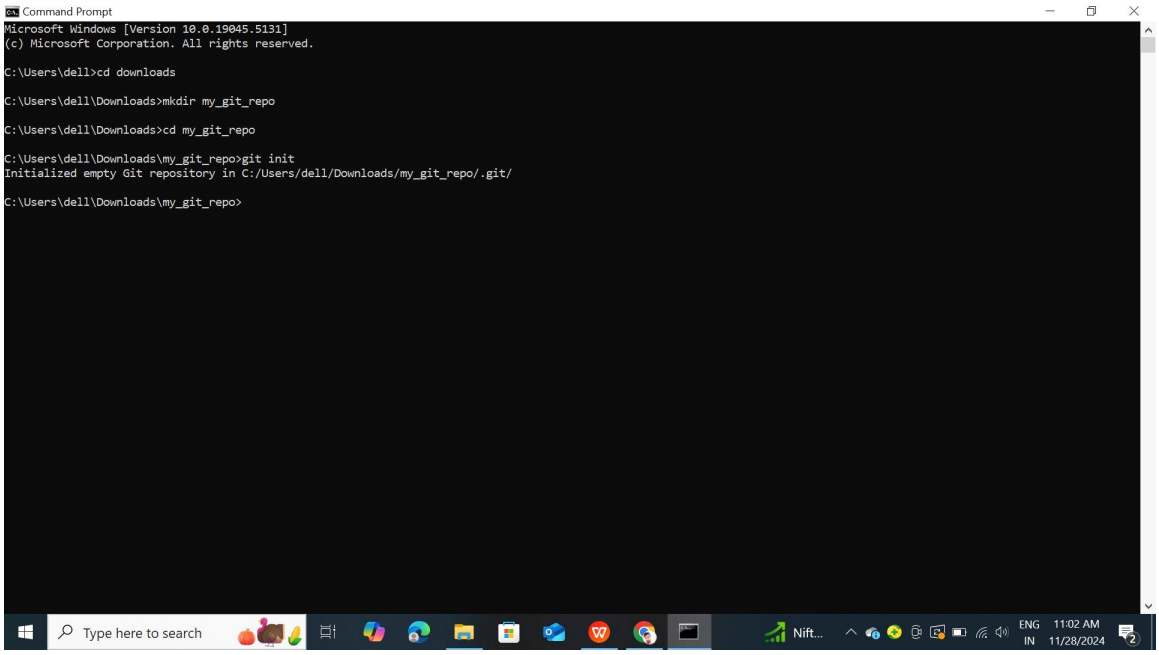


fig:1.0

- Navigate to a directory downloads in local machine to create the Git repository
- create repository name my_git_repo and navigate to that repo use git init to initializated empty git repo

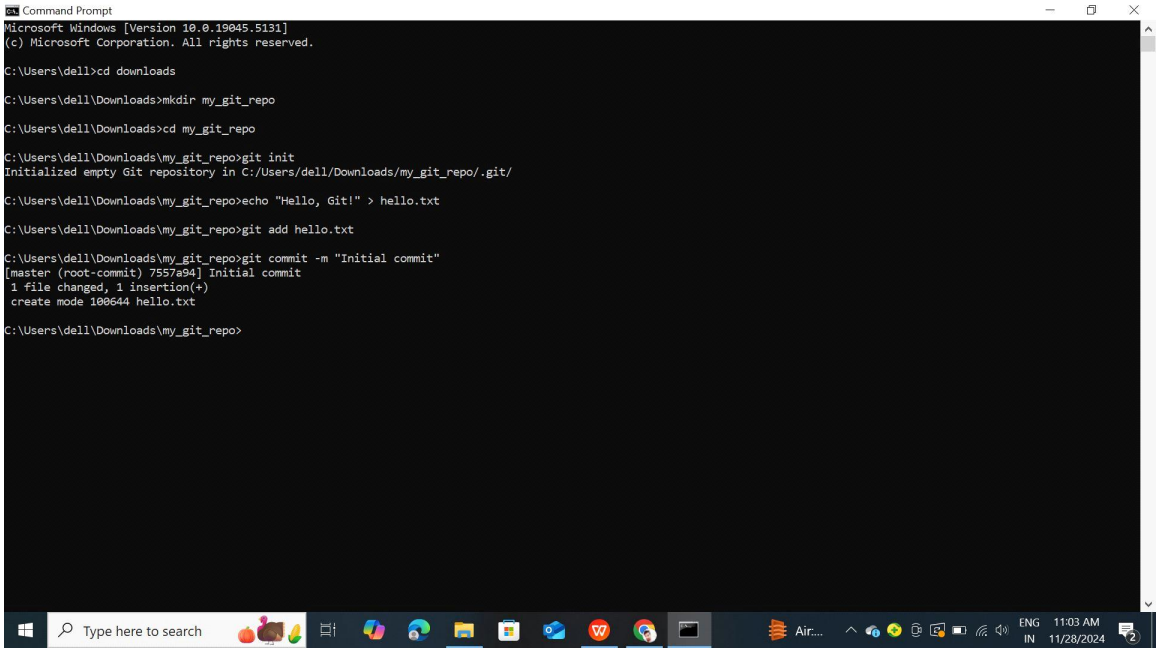


fig:1.1

- Create a file called hello.txt and add some text to it
- Add the file to Git's staging area
- Commit the staged file
- sew the output in above fig :1.1

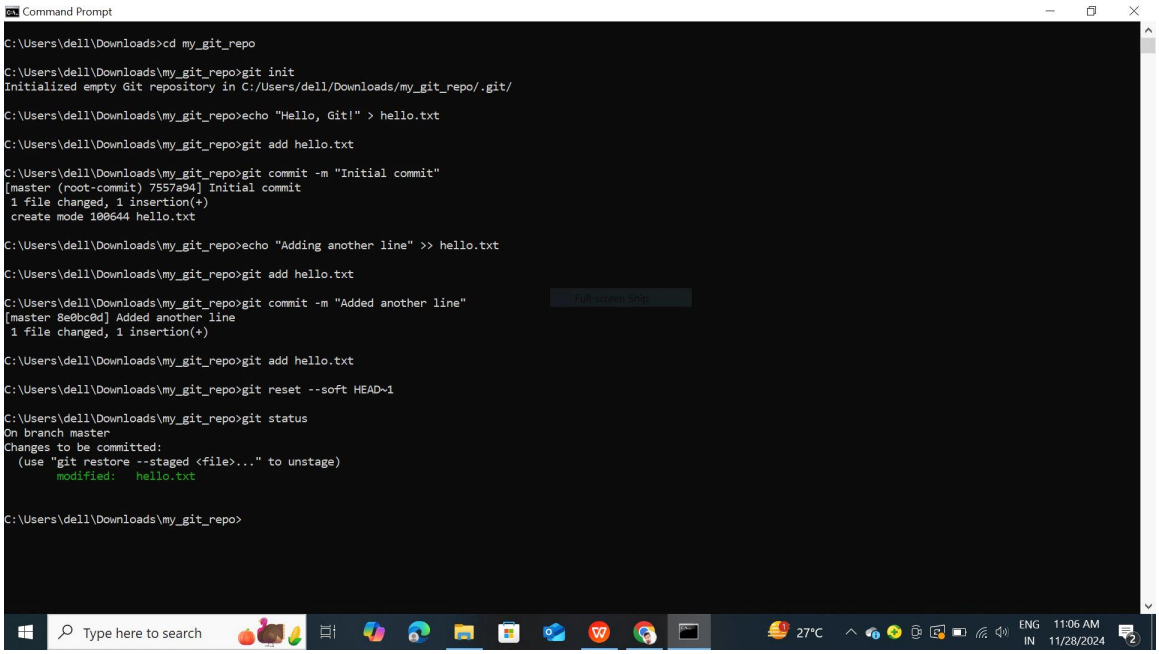


fig :1.2

Demonstrate git reset Options:

1) **git reset --soft**

- git reset --soft resets the HEAD to a previous commit but keeps changes in the staging area
- steps we follow:
- Make Another Change to hello.txt
- Stage and Commit the Change
- Reset with git reset --soft This command moves HEAD back to the previous commit, but keeps the changes staged for the next commit.
- **git status** should we run this command ,You should see that the changeis still staged for commit
- every step out show in above fig :1.2

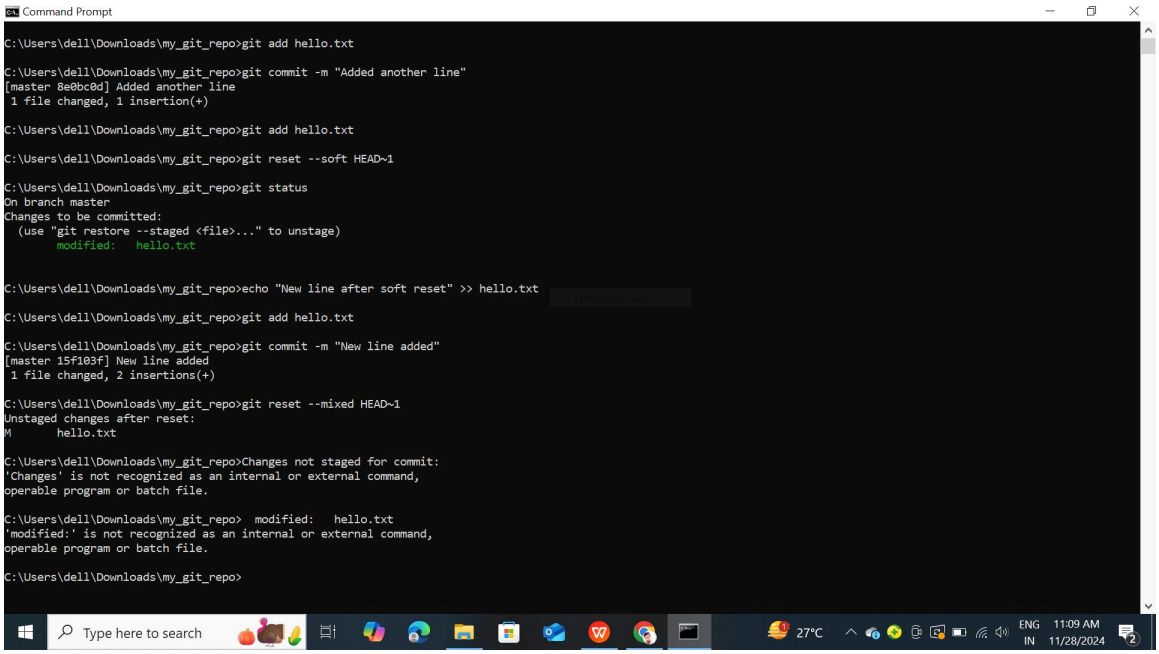


fig :1.3

2) git reset --mixed

- git reset --mixed resets the HEAD and the staging area to a previous commit, but leaves changes in the working directory
- Make another changes of existing file
- after that stage and commit the change in file
- git reset --mixed HEAD~1 use this command This will unstage the changes (move HEAD back), but leave the file unchanged in the working directory.
- then run the git status command we see the out above fig :1.3

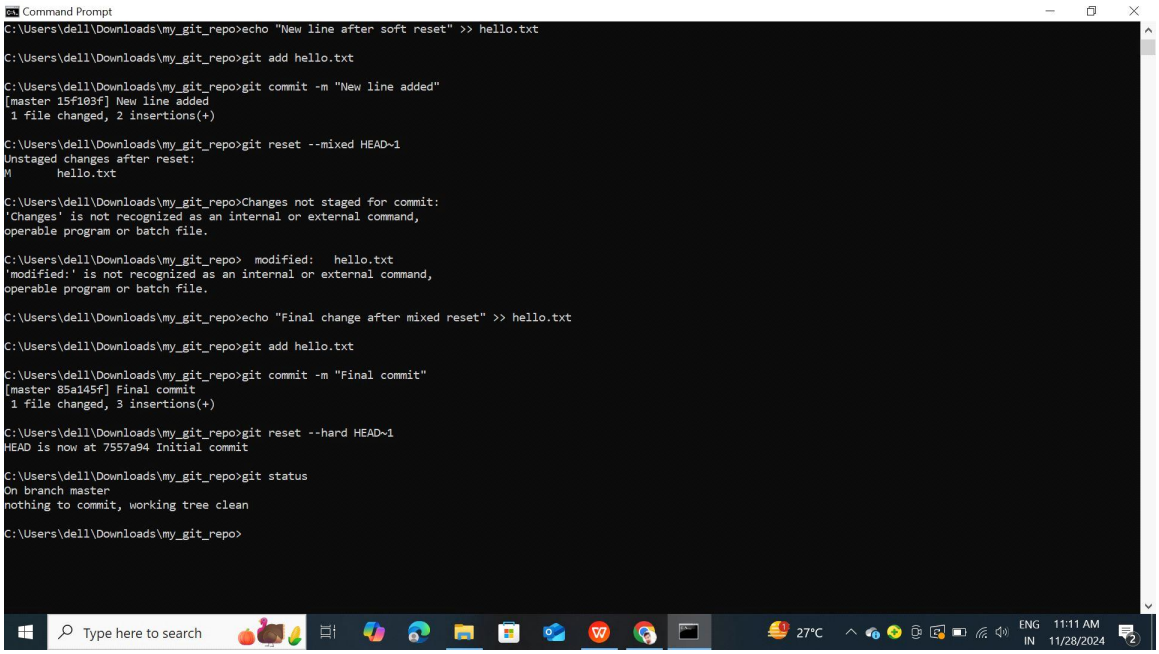


fig :1.4

3) git reset --hard

- same steps we follow above command
- then run git reset --hard HEAD~1 This command will completely discard the last commit and any uncommitted changes.
- the see the output in above fig : 1.4

2) Demonstrate git revert

- same steps we follow what we use in git reset but command changes
- Use git revert to create a new commit that undoes the last commit

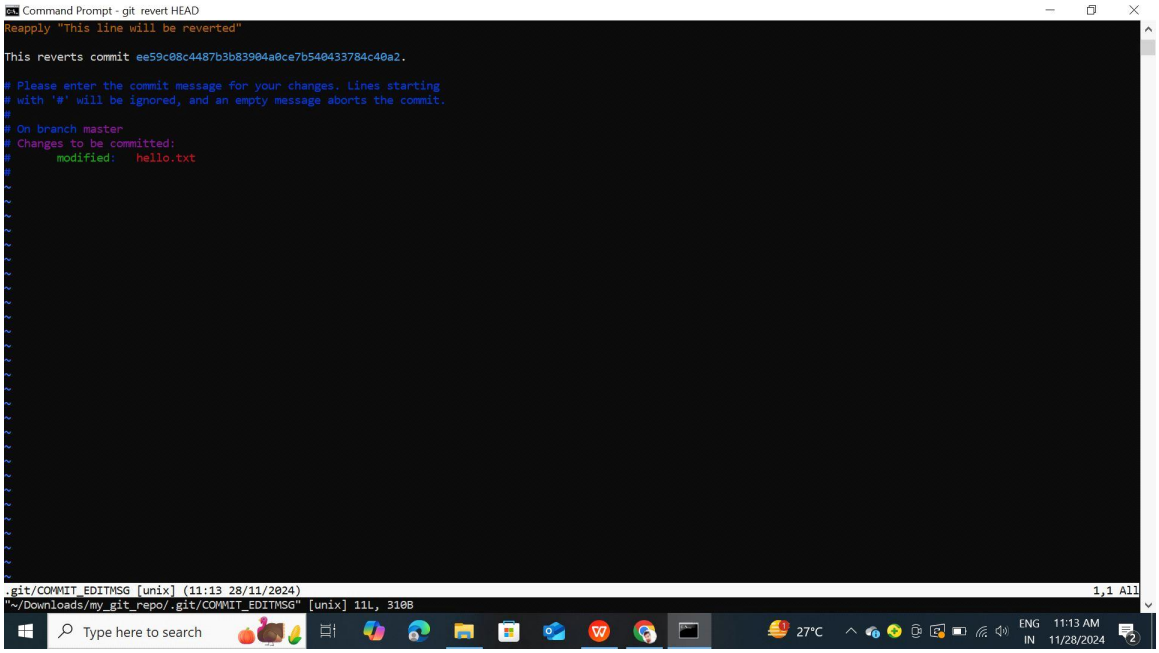


fig : 1.5

- git revert head commad we use see the output in above fig : 1.5

```
C:\Users\dell\Downloads\my_git_repo>echo "This line will be reverted." >> hello.txt

C:\Users\dell\Downloads\my_git_repo>git add hello.txt

C:\Users\dell\Downloads\my_git_repo>git commit -m "This line will be reverted"
[master 2aa44ec] This line will be reverted
 1 file changed, 1 insertion(+)

C:\Users\dell\Downloads\my_git_repo>git revert HEAD
[master ee59c08] Revert "This line will be reverted"
 1 file changed, 1 deletion(-)

C:\Users\dell\Downloads\my_git_repo>git revert HEAD
[master aed3ec1] Reapply "This line will be reverted"
 1 file changed, 1 insertion(+)

C:\Users\dell\Downloads\my_git_repo>
```

- show the output above fig :1.6

```
C:\Users\dell\Downloads\my_git_repo>git revert HEAD
[master aed3ec1] Reapply "This line will be reverted"
 1 file changed, 1 insertion(+)

C:\Users\dell\Downloads\my_git_repo>git log --oneline
aed3ec1 (HEAD -> master) Reapply "This line will be reverted"
ee59c08 Revert "This line will be reverted"
2aa44ec This line will be reverted
7557a94 Initial commit

C:\Users\dell\Downloads\my_git_repo>
```

- git log --oneline View the commit history to confirm the revert commit was added

2. L2 - Create Local git repository and demonstrate git merge and Merge Conflicts with the steps to resolve merge conflicts

- Open local machine terminal

```
Command Prompt
Microsoft Windows [Version 10.0.19045.5131]
(c) Microsoft Corporation. All rights reserved.

C:\Users\dell>cd downloads

C:\Users\dell\Downloads>mkdir my_git_repo

C:\Users\dell\Downloads>cd my_git_repo

C:\Users\dell\Downloads\my_git_repo>git init
Initialized empty Git repository in C:/Users/dell/Downloads/my_git_repo/.git/

C:\Users\dell\Downloads\my_git_repo>
```

fig :1.7

- Navigate to the folder name desktop create your Git repository
- Create a folder called merge_demo

- Move into your newly created folder
- Initialize Git in the current directory
- after above step we saw the output in above fig :1.7

2) Create and Commit Files in the Master Branch

```
:\Users\dell\Desktop\merge_demo>echo "This is the master branch." > file1.txt

:\Users\dell\Desktop\merge_demo>git add file1.txt

:\Users\dell\Desktop\merge_demo>git commit -m "Initial commit in master branch"
[master (root-commit) a5e0f2a] Initial commit in master branch
 1 file changed, 1 insertion(+)
 create mode 100644 file1.txt

:\Users\dell\Desktop\merge_demo>
```

fig:1.8

- Create a file name is file1.txt
- Add the file to Git's staging area
- Commit the file to the repository
- after that we say the output in above fig:1.8

3) Create a New Branch

```
:\Users\dell\Desktop\merge_demo>git checkout -b feature-branch
witched to a new branch 'feature-branch'

:\Users\dell\Desktop\merge_demo>echo "This change is in the feature branch." >> file1.txt

:\Users\dell\Desktop\merge_demo>git add file1.txt

:\Users\dell\Desktop\merge_demo>git commit -m "Changes in feature branch"
feature-branch 9eea8e1] Changes in feature branch
1 file changed, 1 insertion(+)

:\Users\dell\Desktop\merge_demo>
```

fig:1.9

- Create and Switch to a New Branch
- Create a new branch called feature-branch
- Let's modify the file1.txt in the feature-branch

- add file for changes in git stages and also commit the chamges
- we saw the output in above fig:1.9

```
C:\Users\dell\Desktop\merge_demo>git checkout master
Switched to branch 'master'

C:\Users\dell\Desktop\merge_demo>
```

- Switch Back to the master Branch and Modify the Same File

```
C:\Users\dell\Desktop\merge_demo>git checkout master
Switched to branch 'master'

C:\Users\dell\Desktop\merge_demo>echo "This change is in the master branch." >> file1.txt

C:\Users\dell\Desktop\merge_demo>git add file1.txt

C:\Users\dell\Desktop\merge_demo>git commit -m "Changes in master branch"
[master b63f97d] Changes in master branch
 1 file changed, 1 insertion(+)

C:\Users\dell\Desktop\merge_demo>
```

fig:2.0

- Modify the file1.txt in the master branch
- Stage the change and commit the changes
- showen in above fig :2.0

4) Merge feature-branch into master and Create a Merge Conflict

```
C:\Users\dell\Desktop\merge_demo>git merge feature-branch
Auto-merging file1.txt
CONFLICT (content): Merge conflict in file1.txt
Automatic merge failed; fix conflicts and then commit the result.

C:\Users\dell\Desktop\merge_demo>git status
On branch master
You have unmerged paths.
  (fix conflicts and run "git commit")
  (use "git merge --abort" to abort the merge)
```

fig:2.1

- Now, let’s try to merge feature-branch into master
- show error message like automatic merge failed

5) Resolve the Merge Conflict

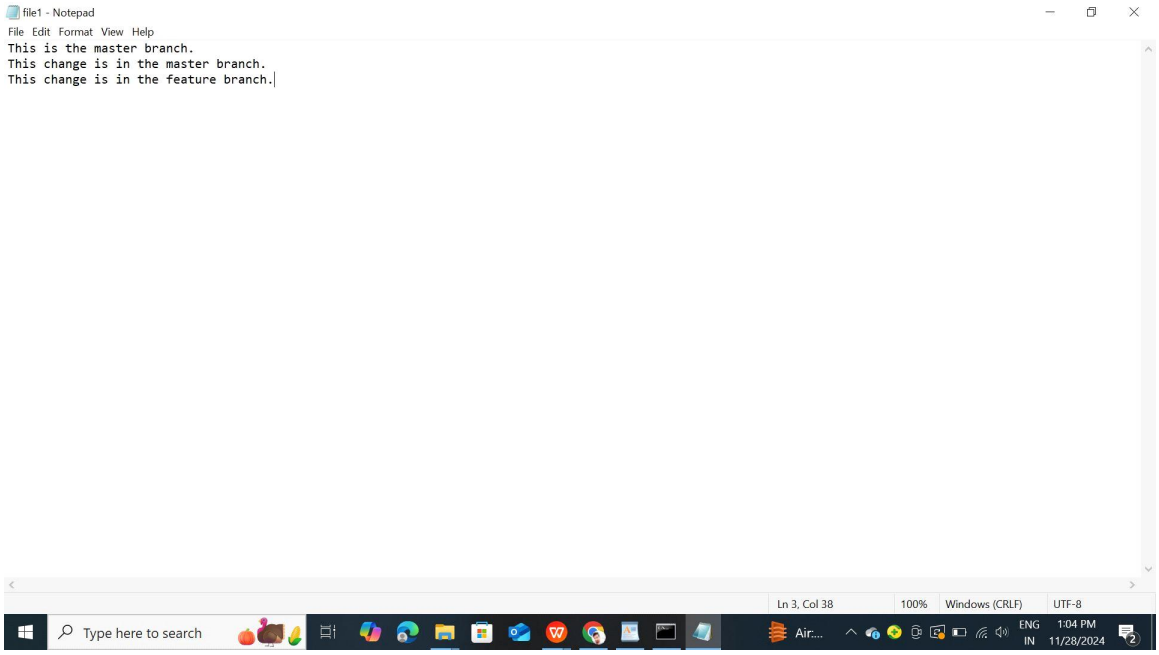


fig:2.2

- Open file1.txt in a text editor. You'll see something like this
- make change show in above fig:2.2

```
C:\Users\dell\Desktop\merge_demo>git add file1.txt

C:\Users\dell\Desktop\merge_demo>git status
On branch master
All conflicts fixed but you are still merging.
  (use "git commit" to conclude merge)

Changes to be committed:
        modified:   file1.txt


C:\Users\dell\Desktop\merge_demo>git commit -m "Resolved merge conflict between master and feature-branch"
[master 47cf801] Resolved merge conflict between master and feature-branch

C:\Users\dell\Desktop\merge_demo>git status
On branch master
nothing to commit, working tree clean
```

fig:2.2

- After resolving the conflict, stage the file and commit merges

6) View the Final File

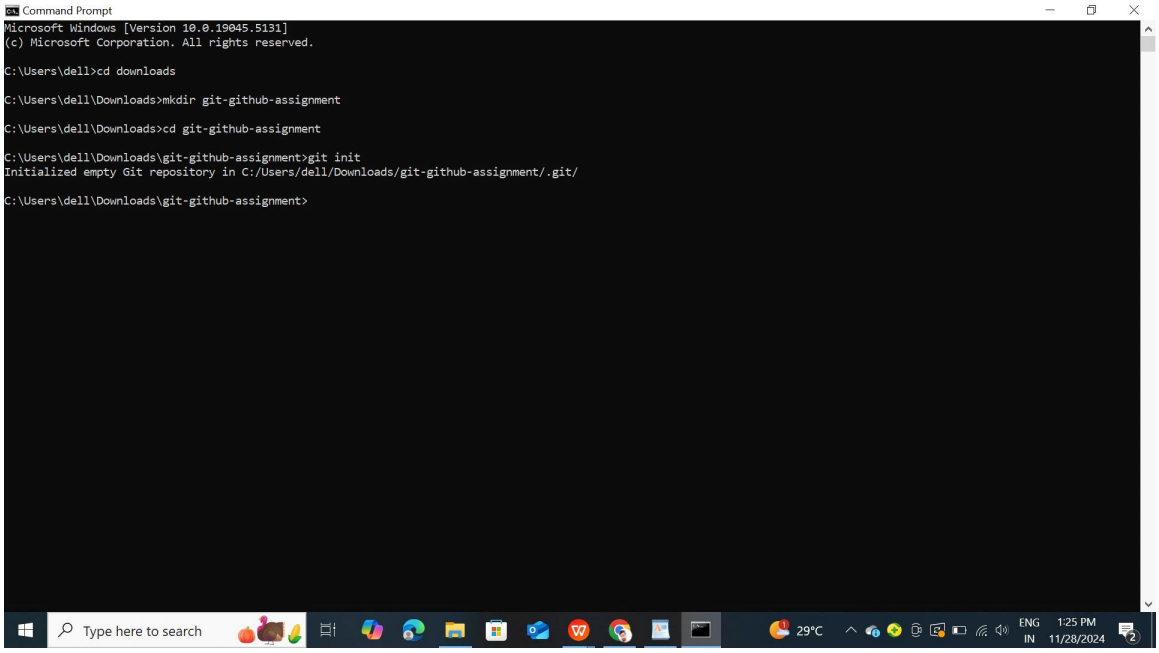
```
C:\Users\dell\Desktop\merge_demo>type file1.txt
"This is the master branch."
<<<<<<< HEAD
"This change is in the master branch."
=======
"This change is in the feature branch."
>>>>>>> feature-branch

C:\Users\dell\Desktop\merge_demo>git add file1.txt

C:\Users\dell\Desktop\merge_demo>git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   file1.txt


C:\Users\dell\Desktop\merge_demo>git commit -m "Resolved merge conflict between master and feature-branch"
[master ce5a107] Resolved merge conflict between master and feature-branch
 1 file changed, 3 insertions(+), 6 deletions(-)

C:\Users\dell\Desktop\merge_demo>type file1.txt
This is the master branch.
This change is in the master branch.
This change is in the feature branch.
```

fig:2.4

- You can check the contents of file1.txt to ensure the conflict was resolved correctly
- The output should show the merged content above fig :2.4

3. L3 - Using Local and Remote git repositories demonstrate git pull and git fetch. Compare the differences

1) Create Local Git Repository

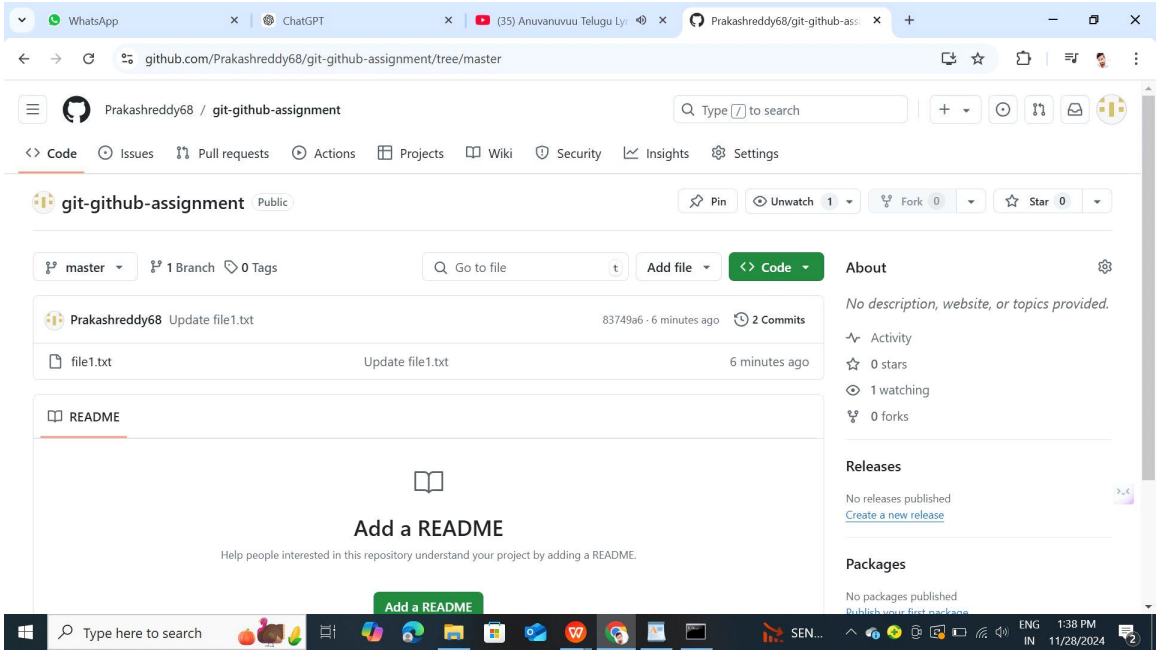


- Create a directory and navigate into it
- Initialize the Git repository

2) Create and Commit a File

- create file name is file1
- after that add file2 for stage area
- finally commit tha changes

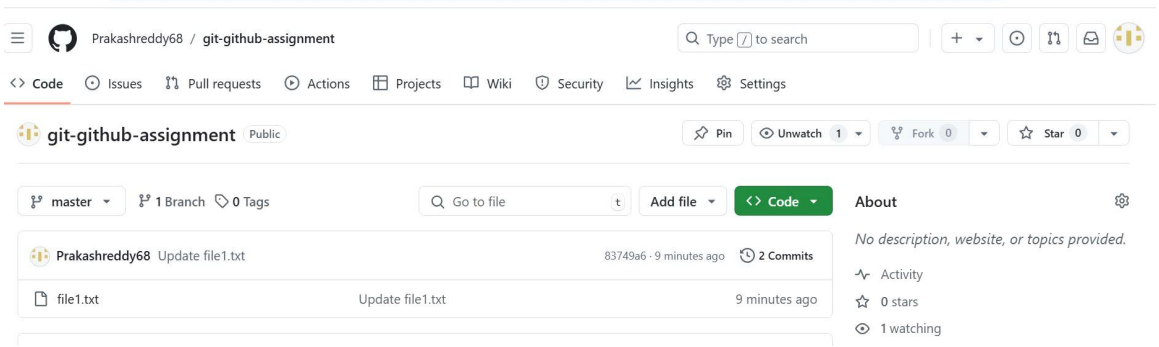
3) Create a Remote Repository on GitHub



- Create a remote repository on GitHub named git-github-assignment.

```
C:\Users\dell\Downloads\git-github-assignment>git remote add origin https://github.com/Prakashreddy68/git-github-assignment.git

C:\Users\dell\Downloads\git-github-assignment>git push -u origin master
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Writing objects: 100% (3/3), 242 bytes | 242.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/Prakashreddy68/git-github-assignment.git
 * [new branch]      master -> master
branch 'master' set up to track 'origin/master'.

C:\Users\dell\Downloads\git-github-assignment>
```

- Link your local repo to the GitHub repository
- Push your local commit to GitHub

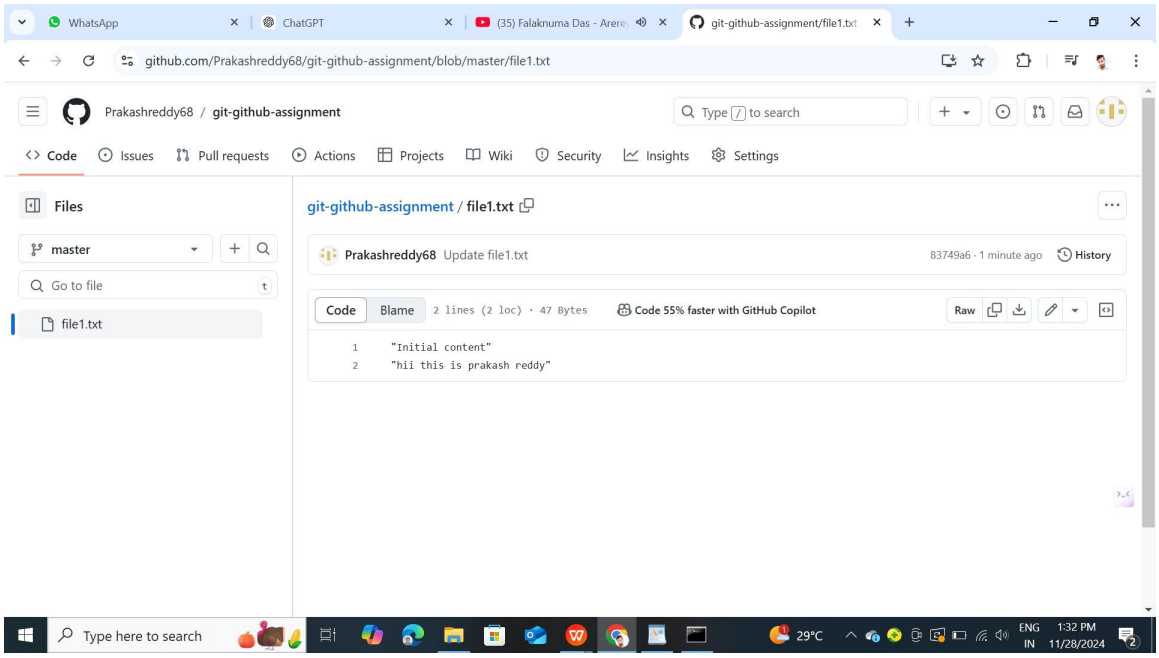


- successfully file1 push to github- repo

4) Simulate Changes in Local and Remote Repositories

```
C:\Users\dell\Downloads\git-github-assignment>echo "New change locally" >> file1.txt

C:\Users\dell\Downloads\git-github-assignment>git add file1.txt

C:\Users\dell\Downloads\git-github-assignment>git commit -m "Local change"
[master a85083a] Local change
 1 file changed, 1 insertion(+)

C:\Users\dell\Downloads\git-github-assignment>
```

- Make a local change in the file
- Stage and commit the local change



- successfully make change in github-repo

5) Fetch Changes from Remote Repository

```
C:\Users\dell\Downloads\git-github-assignment>git fetch origin
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
Unpacking objects: 100% (3/3), 939 bytes | 46.00 KiB/s, done.
From https://github.com/Prakashreddy68/git-github-assignment
   c02f083..83749a6  master     -> origin/master
```

- Fetch changes from the remote

6) Pull Changes from Remote Repository

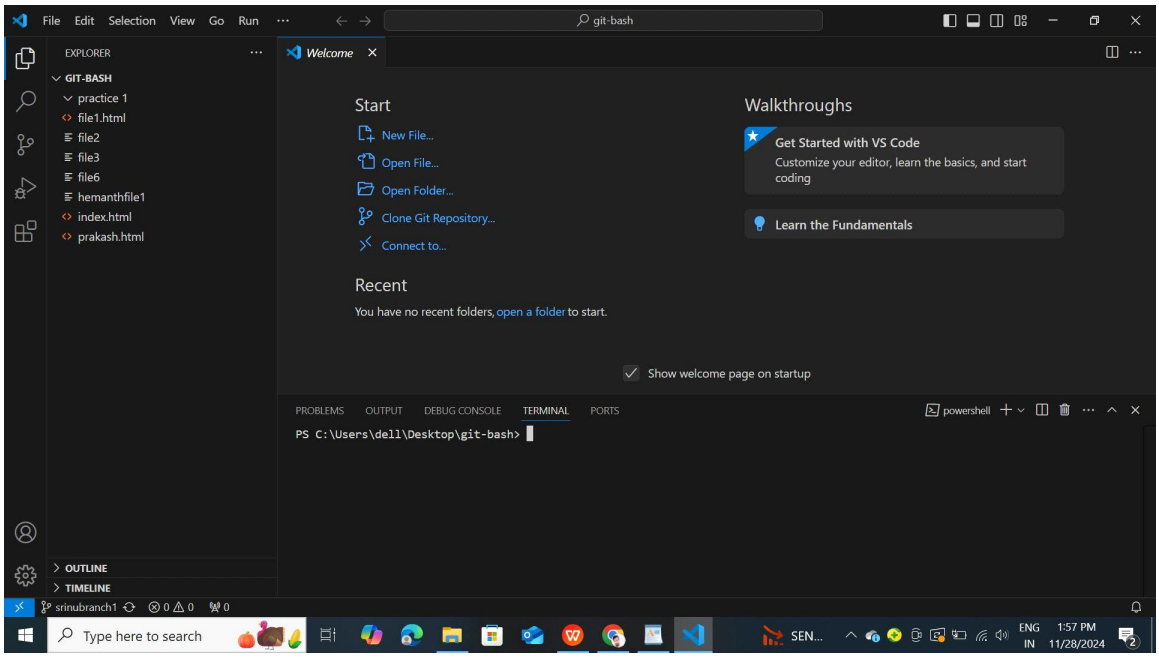
```
C:\Users\dell\Downloads\git-github-assignment>git pull origin master
From https://github.com/Prakashreddy68/git-github-assignment
 * branch            master     -> FETCH_HEAD
Auto-merging file1.txt
CONFLICT (content): Merge conflict in file1.txt
Automatic merge failed; fix conflicts and then commit the result.

C:\Users\dell\Downloads\git-github-assignment>
```

- Pull the changes from the remote repository

4. L4 - Clone GitHub repository using Visual Studio Code IDE

1) Install Visual Studio Code

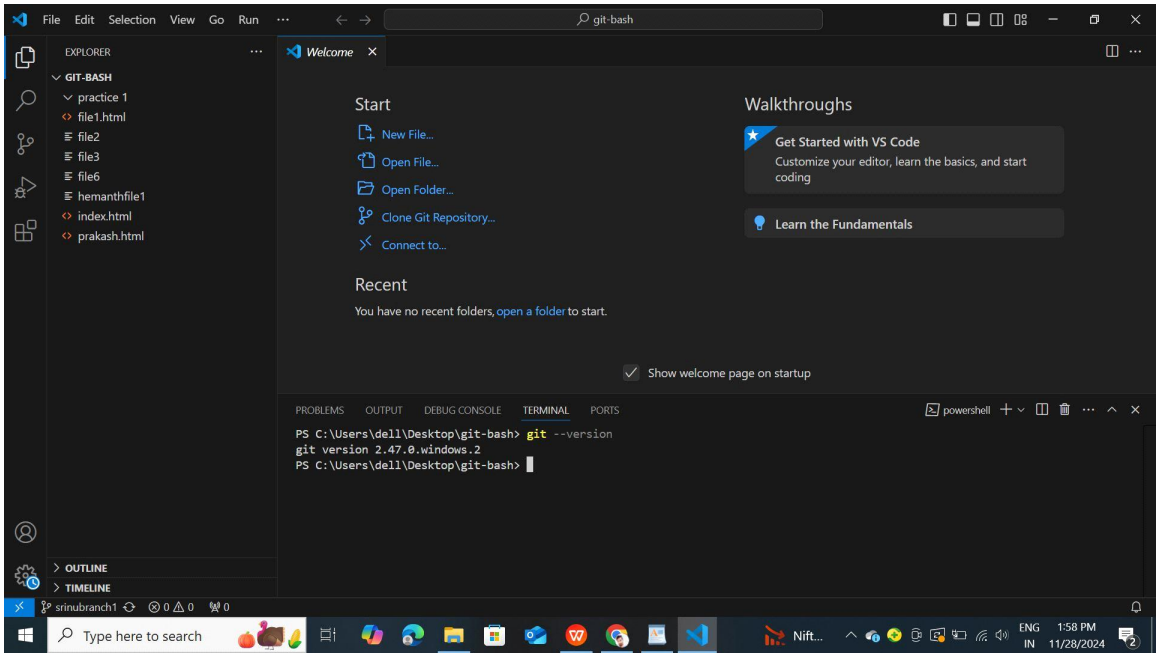
Download and Install Visual Studio Code



- Once installed, open Visual Studio Code

2) Install Git

Download Git

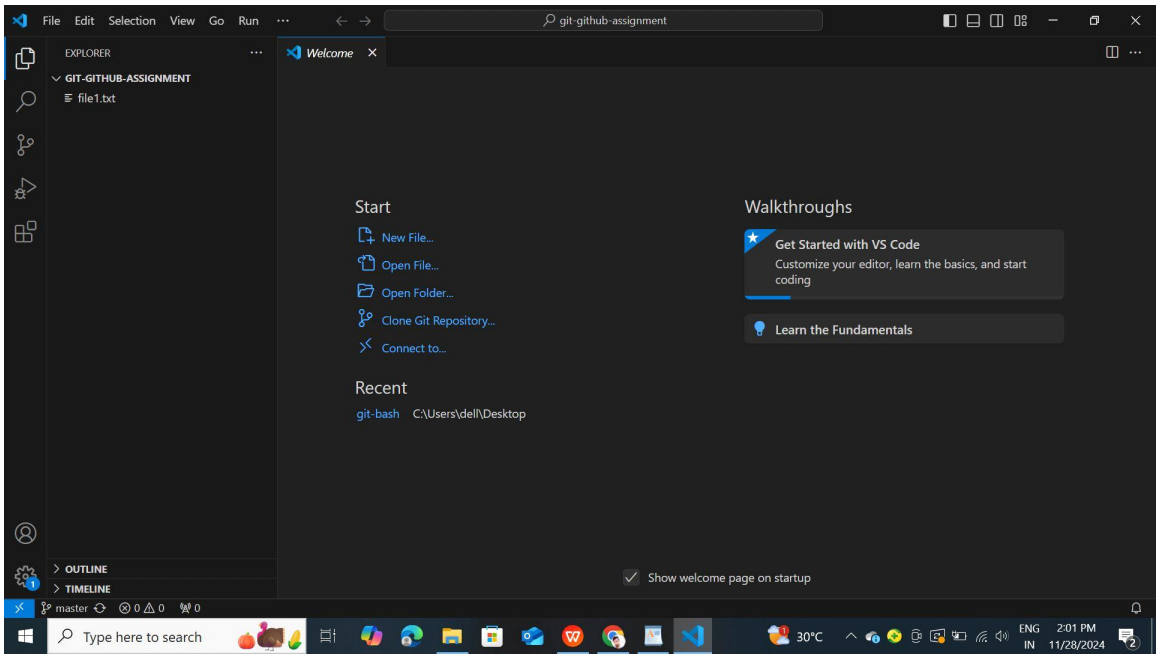


- After installation, verify it by opening the terminal in VS Code

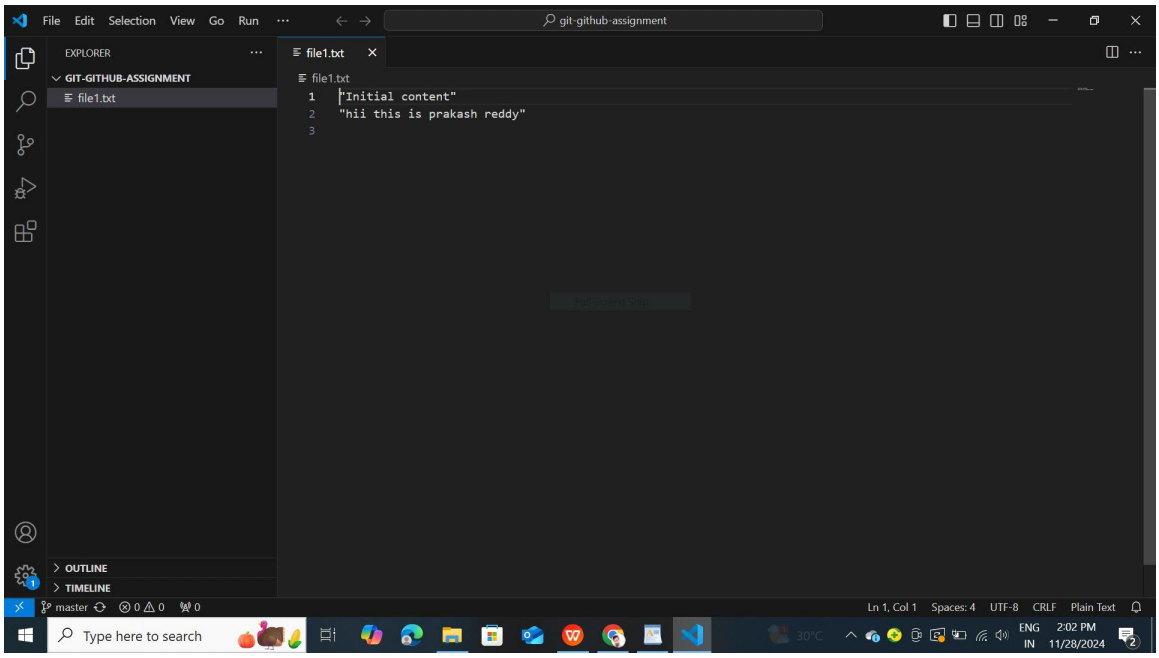
- git --version use this command to saw the version and conformation of git installation

3) Open the Command Palette

- Press Ctrl+Shift+P
- In the Command Palette, type Git: Clone and select it.
- Paste the GitHub https://github.com/Prakashreddy68/git-github-assignment.git



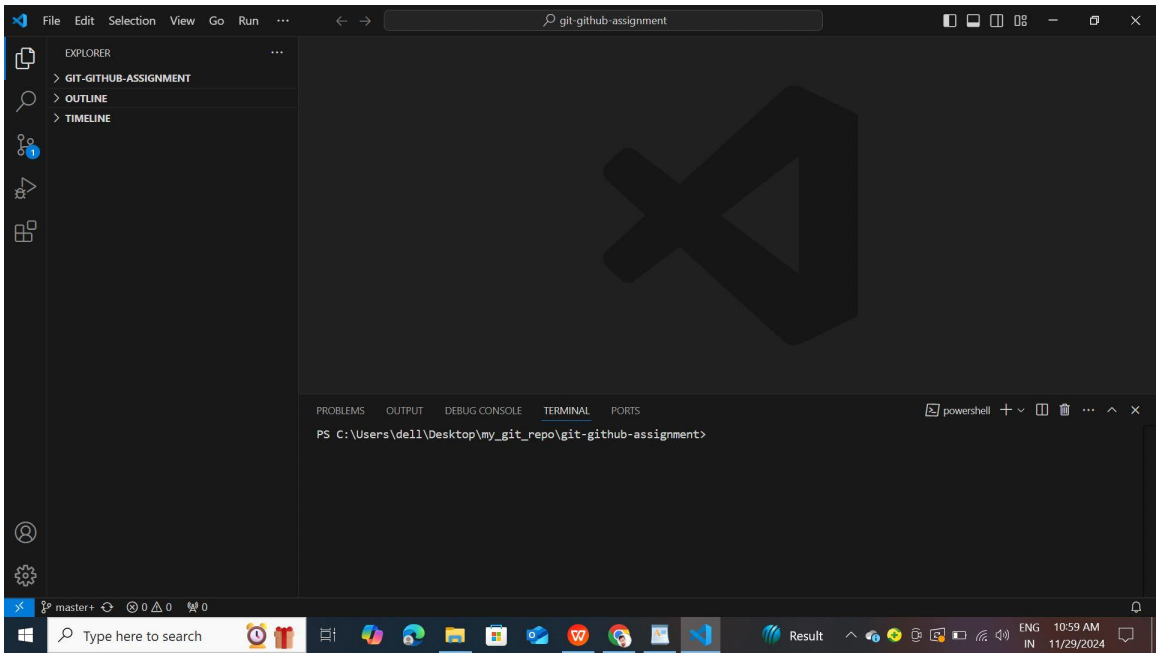
- After the cloning finishes, VS Code will ask if you want to open the repository.



- Once the repository is open, than we will see the project files in VS Code.
- now start editing files, commit changes, or interact with the repository
- show in above screenshot

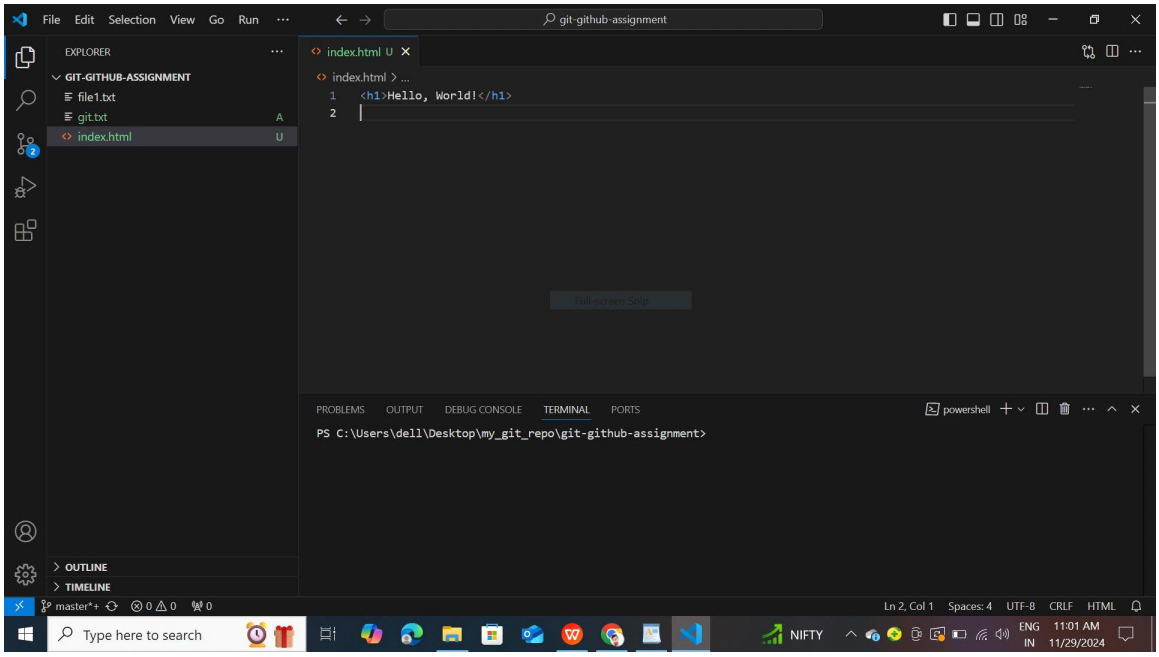
5. L5 - Push the incremental changes to GitHub Repository through Visual Studio Code IDE

1) open VS code terminal



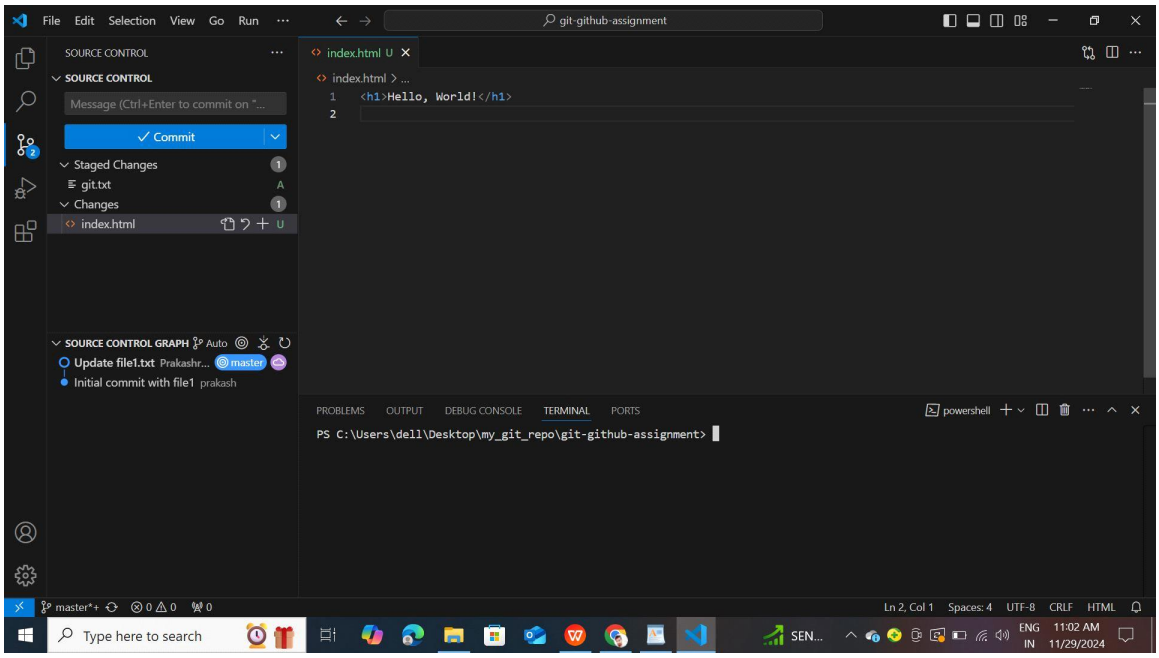
- Successfully download VS code and install git

2) create file in github repository



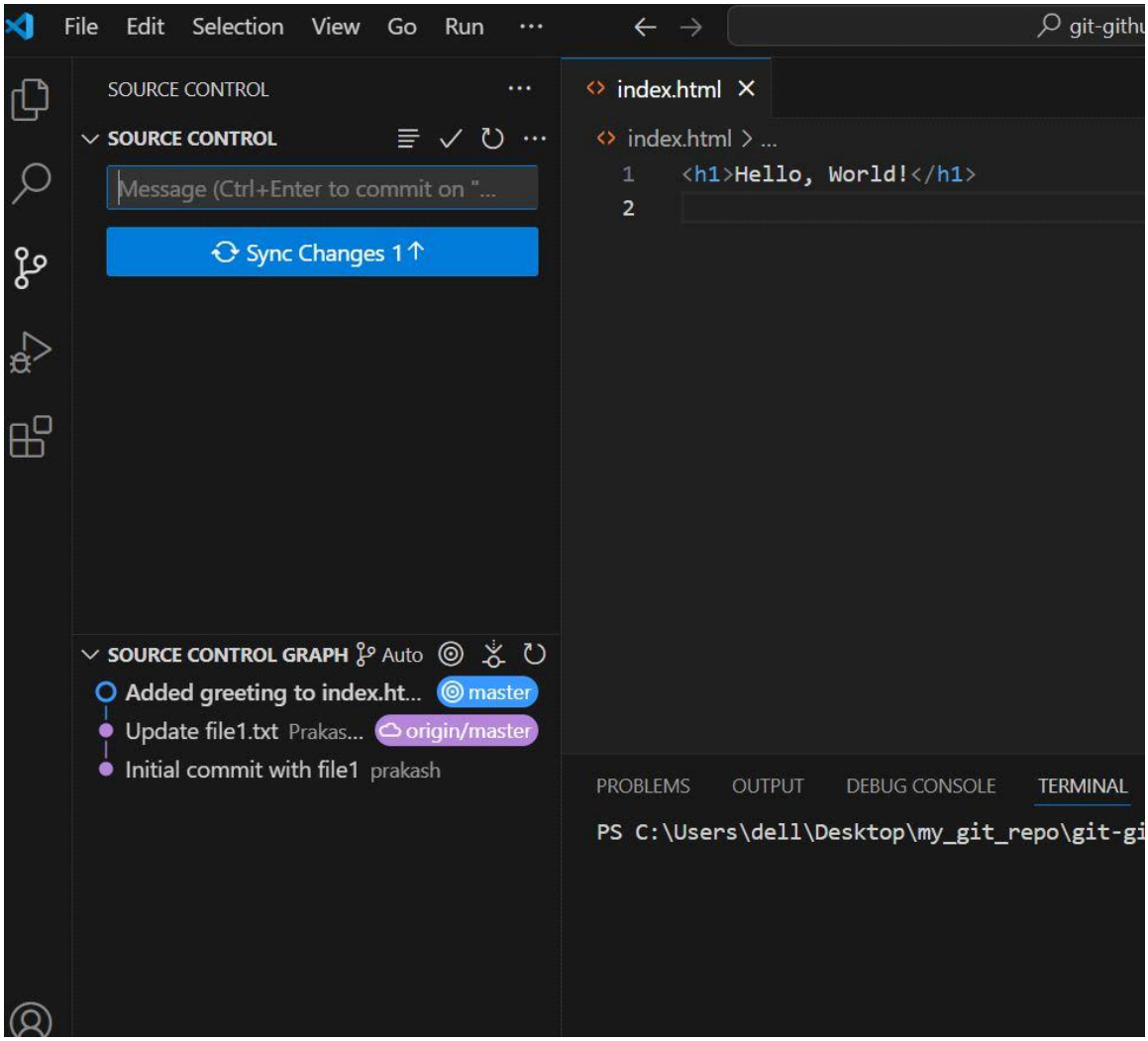
- create file name is index.html and add some content

3) make stages of create file in repo



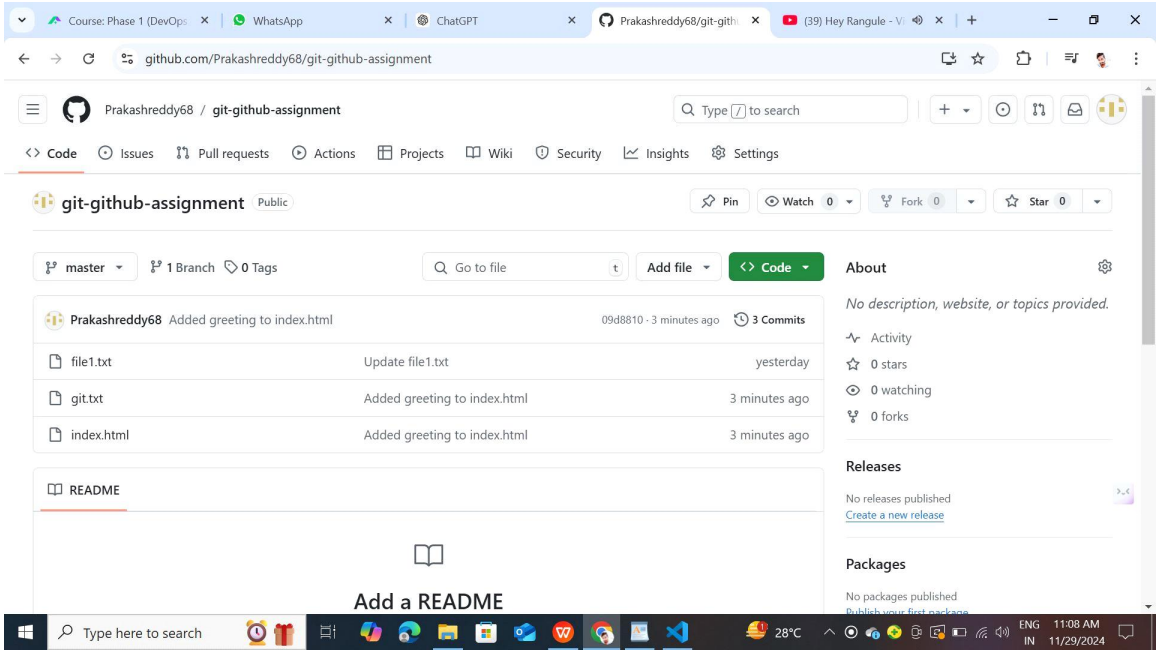
- click the pluse icon of index.html file and alternatively click pluse icon top of source panel

4) commit the file with send message



- successfully commit the chanages

5) GITHUB repo check after the file push



- successfully push the changes with VS code to github repo shown in above figure

THANK YOU