Maddula Rupa Sri Manohar
maddularupasrimanohar2001@gmail.com

Assignment-5

Assignment 1:Initialize a new Git repository in a directory of your choice. Add a simple text file to the repository and make the first commit.

Answer:

Open Terminal

Navigate to your desired directory: Use the cd command to move to the location where you wantyour Git repository.

```
$ mkdir Myfile
```

```
$ cd Myfile
```

Initialize the Git repository: Run the command git init. This creates a hidden folder named .git in yourcurrent directory, which holds the Git repository data.

```
$ git init
```

Create a text file

```
Test.txt
```

Stage the file for commit

```
$ git add .
```

Commit the changes

```
$ git commit -m " First commit"
```

```
C:\Sample>git init
Initialized empty Git repository in C:/Sample/.git/

C:\Sample>git remote add origin https://github.com/Manohar-M-233/Mano_repo.git

C:\Sample>git add.
git: 'add.' is not a git command. See 'git --help'.

The most similar command is
        add

C:\Sample>git add .

C:\Sample>git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   hello.txt


C:\Sample>git commit -m "hello file"
[master (root-commit) f0b1f52] hello file
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 hello.txt
```

Assignment 2: Branch Creation and Switching

Create a new branch named 'feature' and switch to it. Make changes in the 'feature' branch and commit them.

Answer:

Create a new branch

$ git branch feature

Switch to feature branch

$ git checkout feature

Create some file/folder add,commit,push to the feature branch

Ex: ProductFile.txt

This will add UserData.txt file to local repo

$ git add .

Status command will show the list of files or folders added

$ git status

This will save ProductFile.txt in local repo

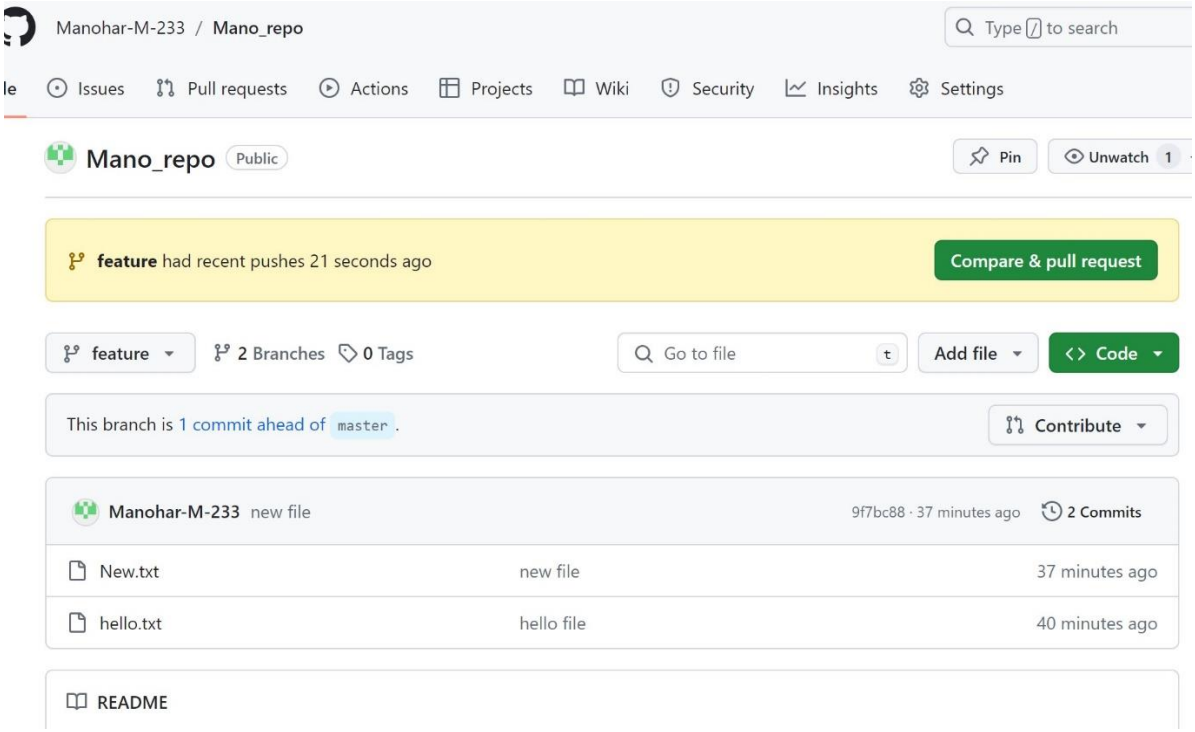
$ git commit -m “Product file added”

This will add ProductFile.txt file to the feature branch only in local repo

$ git push origin feature

Note: This UserData.txt file will not be appear in master branch untill we do merge.

```
C:\Sample>git branch feature

C:\Sample>git branch
  feature
* master

C:\Sample>git checkout feature
Switched to branch 'feature'

C:\Sample>git add .

C:\Sample>git status
On branch feature
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   New.txt


C:\Sample>git commit -m "new file"
[feature 9f7bc88] new file
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 New.txt

C:\Sample>git push origin feature
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Delta compression using up to 16 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (2/2), 252 bytes | 252.00 KiB/s, done.
Total 2 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
remote:
remote: Create a pull request for 'feature' on GitHub by visiting:
remote:      https://github.com/Manohar-M-233/Mano_repo/pull/new/feature
remote:
To https://github.com/Manohar-M-233/Mano_repo.git
 * [new branch]      feature -> feature
```



Assignment 3: Feature Branches and Hotfixes

Create a 'hotfix' branch to fix an issue in the main code. Merge the 'hotfix' branch into 'main' ensuring that the issue is resolved.

Answer:

Create a new branch

$ git branch branch_1

Switch to branch_1 branch

$ git checkout branch_1

Create some file/folder add,commit,push to the branch_1

Ex: UserData.txt

This will add UserData.txt file to local repo

$ git add .

Status command will show the list of files or folders added

$ git status

This will save UserData.txt in local repo

$ git commit -m “UserData file added”

This will add UserData.txt file to the feature branch only in local repo

$ git push origin feature

Note: This UserData.txt file will not be appear in master branch untill we do merge.

Then switch to master branch

$ git checkout master

Merge branch_1 to master

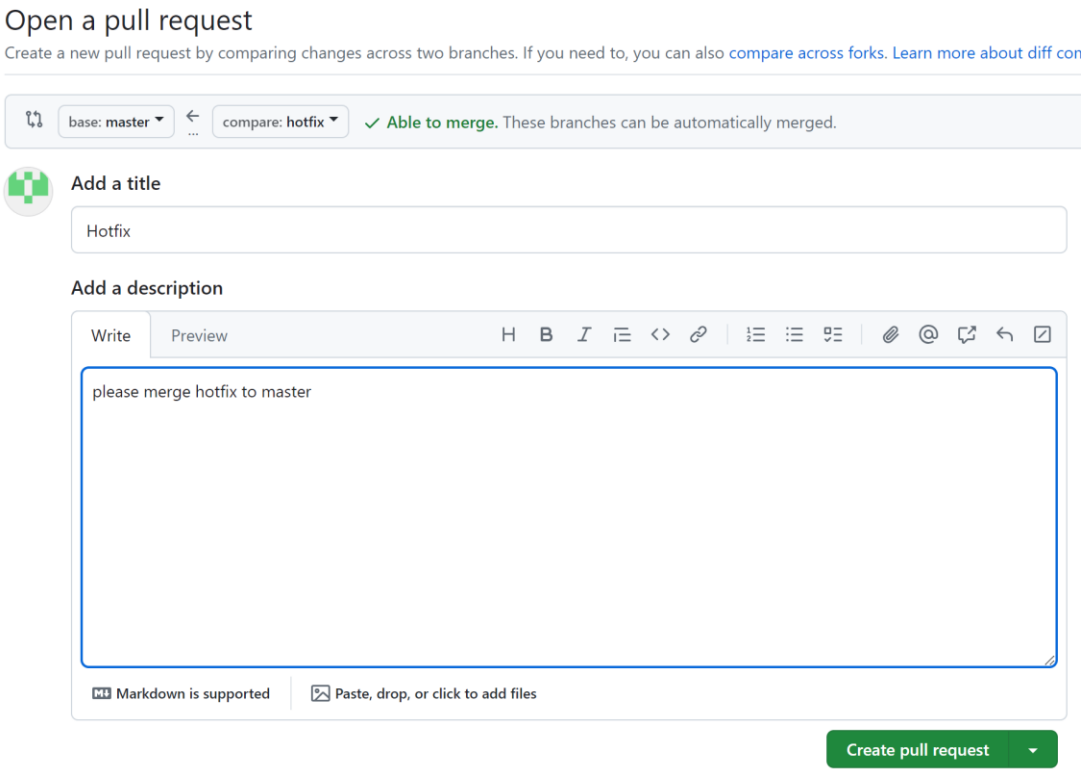
$ git merge branch _1

```
C:\Sample>git branch hotfix

C:\Sample>git checkout hotfix
Switched to branch 'hotfix'

C:\Sample>git add .

C:\Sample>git status
On branch hotfix
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   File.txt


C:\Sample>git commit -m "New File added"
[hotfix 7604819] New File added
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 File.txt

C:\Sample>git push origin hotfix
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Delta compression using up to 16 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (2/2), 264 bytes | 264.00 KiB/s, done.
Total 2 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
remote:
remote: Create a pull request for 'hotfix' on GitHub by visiting:
remote:      https://github.com/Manohar-M-233/Mano_repo/pull/new/hotfix
remote:
To https://github.com/Manohar-M-233/Mano_repo.git
 * [new branch]      hotfix -> hotfix

C:\Sample>git checkout master
Switched to branch 'master'
```

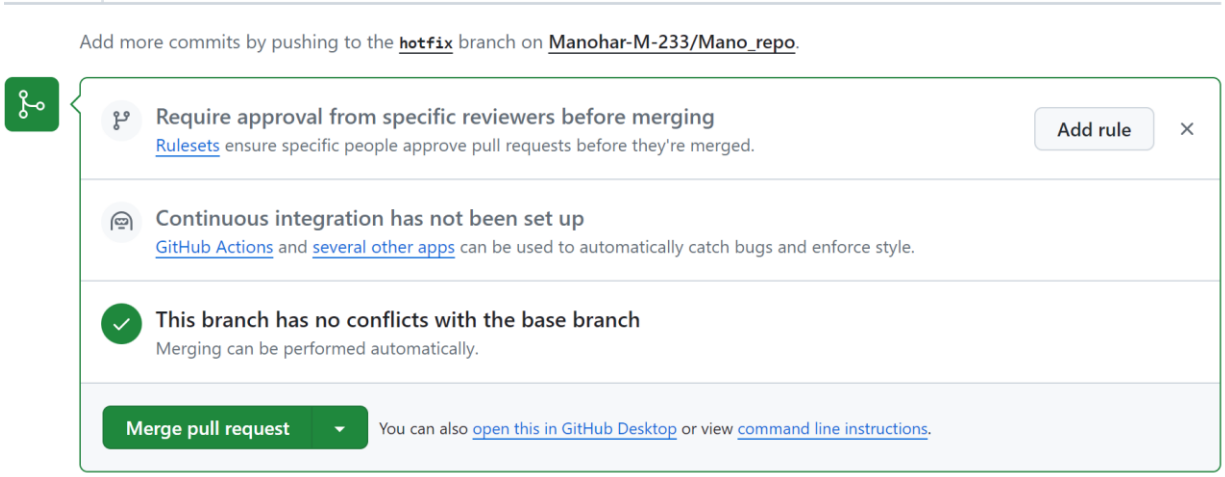
```
C:\Sample>git checkout master
Switched to branch 'master'

C:\Sample>git merge hotfix
Updating f0b1f52..7604819
Fast-forward
 File.txt | 0
 New.txt  | 0
 2 files changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 File.txt
 create mode 100644 New.txt

C:\Sample>
```

Now , when ever we do merge , it will not reflect to the origin/remote until and unless

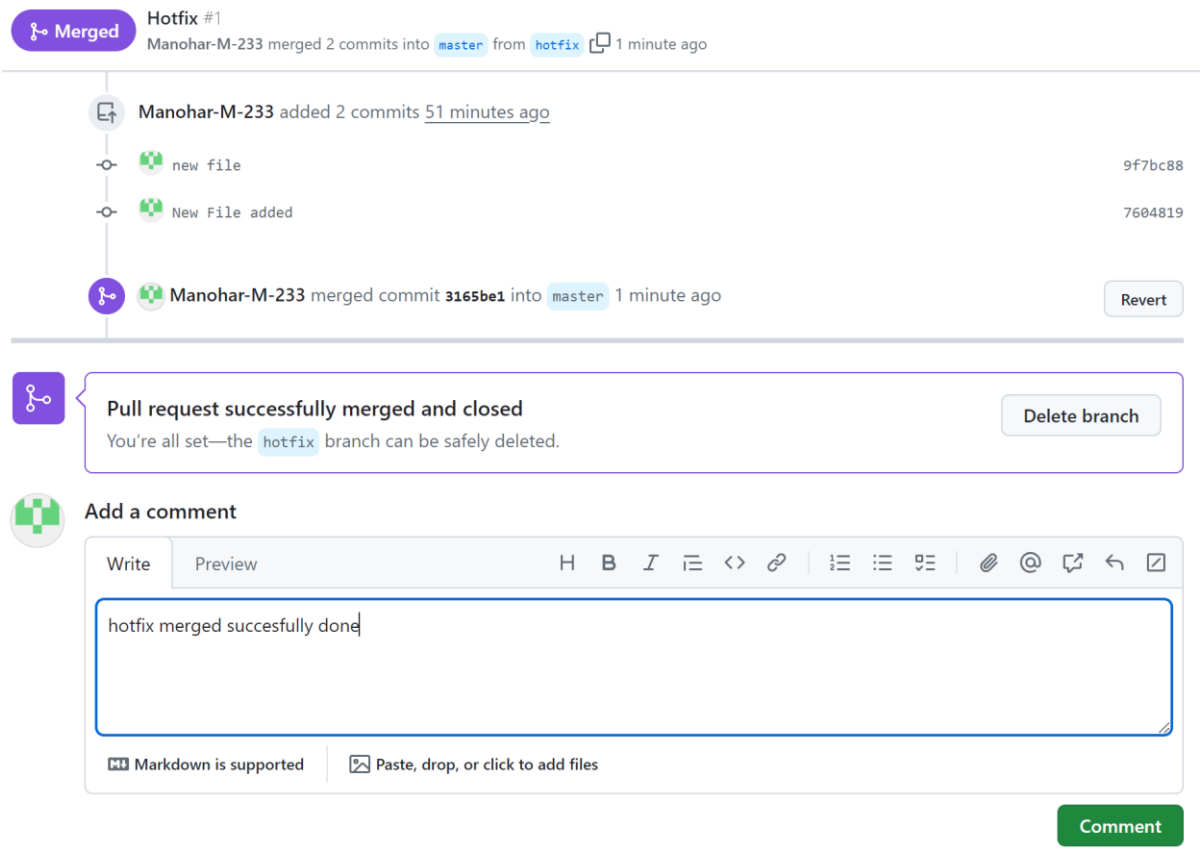
i. pull request is rise (any developer or Team Lead can do this)



ii.merge request is confirmed (Repository owner/Product owner / senior person)



iii.comment is done (its mandatory)



Then the file which are in the hotfix will reflect in master branch.

Here File.txt is the file in hotfix