Working with git &GitHub

Working with Git Commands and sharing the code with others using GitHub.

**In this lab , you will learn how to do the below :**

1.How to configure Git Global Parameters (Username and Email address)

2.How to initialize Git and create project folder with some sample code

3.How to push the code from Git to GitHub Public repository

4.How to clone, pull and fetch from GitHub Public repository

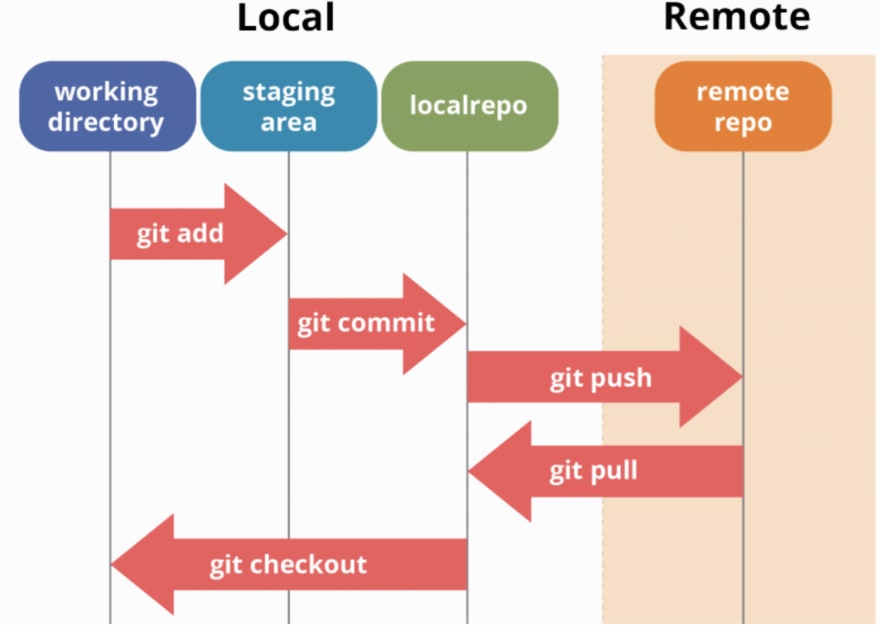
5.How to use git diff and git reset commands in staging area

6.How to check the logs using git log

7.How to create branches and merge the branches with main branch in Git

8.How to revert changes in Git

9.How to create branches and Pull Requests in Git Hub



**Help:**

git command –help

chait@LAPTOP-QHKO7HQH MINGW64 ~ (main)

$ git init --help

**Update Global Configuration parameters:**

git config --global user.name "Name"

git config --global user.email "emailid"

git config –list

chait@LAPTOP-QHKO7HQH MINGW64 ~ (main)

$ git config --global user.name "user1"

chait@LAPTOP-QHKO7HQH MINGW64 ~ (main)

$ git config --global user.email "user1@x.com"

chait@LAPTOP-QHKO7HQH MINGW64 ~ (main)

$ git config --list

diff.astextplain.textconv=astextplain

filter.lfs.clean=git-lfs clean -- %f

filter.lfs.smudge=git-lfs smudge -- %f

filter.lfs.process=git-lfs filter-process

filter.lfs.required=true

http.sslbackend=openssl

http.sslcainfo=C:/Program Files/Git/mingw64/ssl/certs/ca-bundle.crt

core.autocrlf=true

core.fscache=true

core.symlinks=false

credential.helper=manager

pull.rebase=false

user.email=user1@x.com

user.name=user1

core.repositoryformatversion=0

core.filemode=false

core.bare=false

core.logallrefupdates=true

core.symlinks=false

core.ignorecase=true

remote.origin.url=https://github.com/chaitanyagaajula/repo2610.git

remote.origin.fetch=+refs/heads/\*:refs/remotes/origin/\*

branch.main.remote=origin

**Create and log into Project Directory (Working Directory)**

mkdir project1

cd project1

chait@LAPTOP-QHKO7HQH MINGW64 ~ (main)

$ mkdir project1

chait@LAPTOP-QHKO7HQH MINGW64 ~ (main)

$ cd project1

**Initialize git:**

**git init (Creates .git repository under your users folder)**

chait@LAPTOP-QHKO7HQH MINGW64 ~/project1 (main)

$ git init

Initialized empty Git repository in C:/Users/chait/project1/.git/

**Create files**

touch filename

chait@LAPTOP-QHKO7HQH MINGW64 ~/project1 (master)

$ touch example.txt

vi filename and add contents to it

chait@LAPTOP-QHKO7HQH MINGW64 ~/project1 (master)

$ vi example.txt

first line

second line

third line

~

~

~

~

~

~

~

~

~

~

~

~

~

~

~

~

~

~

~

example.txt [unix] (11:12 27/12/2

:wq

**Check if file contents have been saved**

chait@LAPTOP-QHKO7HQH MINGW64 ~/project1 (master)

$ cat example.txt

first line

second line

third line

**Add files to Staging area and check if the file moved to Staging area**

git add filename

ls

git status

chait@LAPTOP-QHKO7HQH MINGW64 ~/project1 (master)

$ git add example.txt

warning: LF will be replaced by CRLF in example.txt.

The file will have its original line endings in your working directory

chait@LAPTOP-QHKO7HQH MINGW64 ~/project1 (master)

$ git status example.txt

On branch master

No commits yet

Changes to be committed:

(use "git rm --cached <file>..." to unstage)

new file: example.txt

**Commit files to local repository**

git commit -m "first commit"

chait@LAPTOP-QHKO7HQH MINGW64 ~/project1 (master)

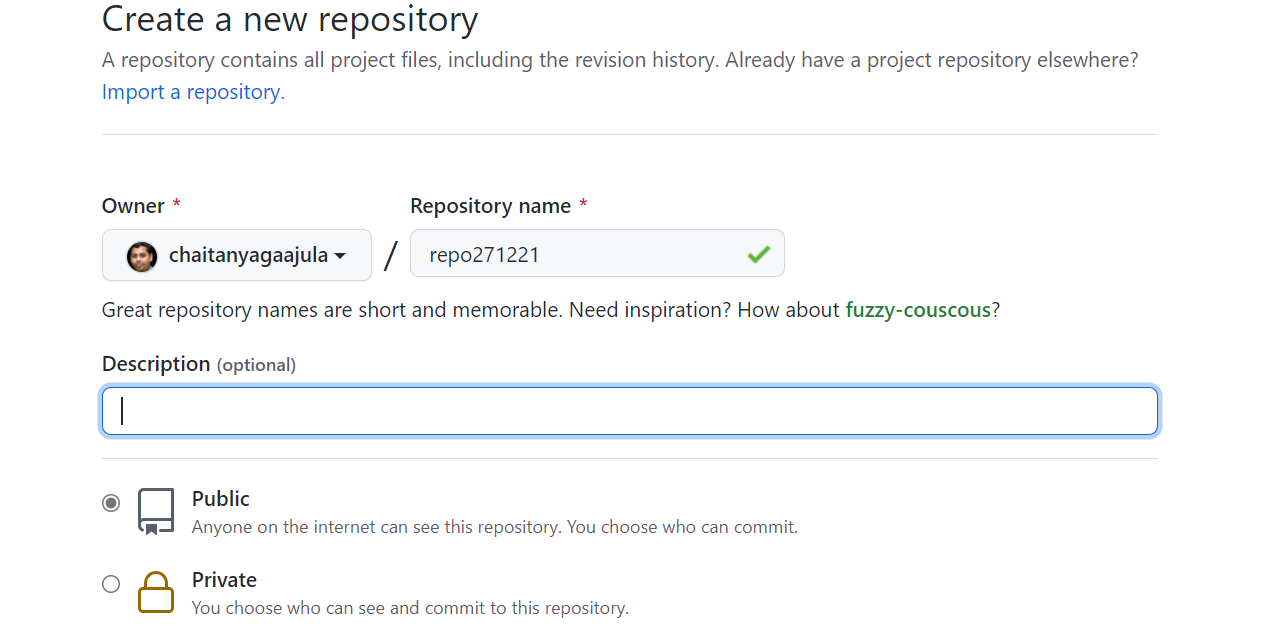
$ git commit -m "first commit"

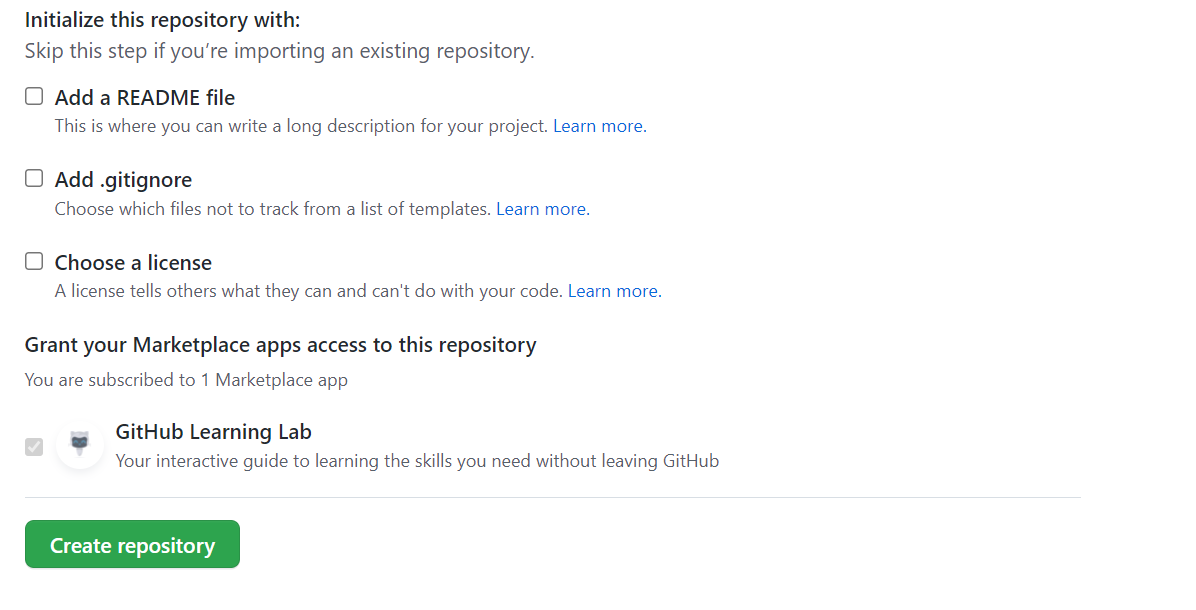
[master (root-commit) ffa7042] first commit

1 file changed, 3 insertions(+)

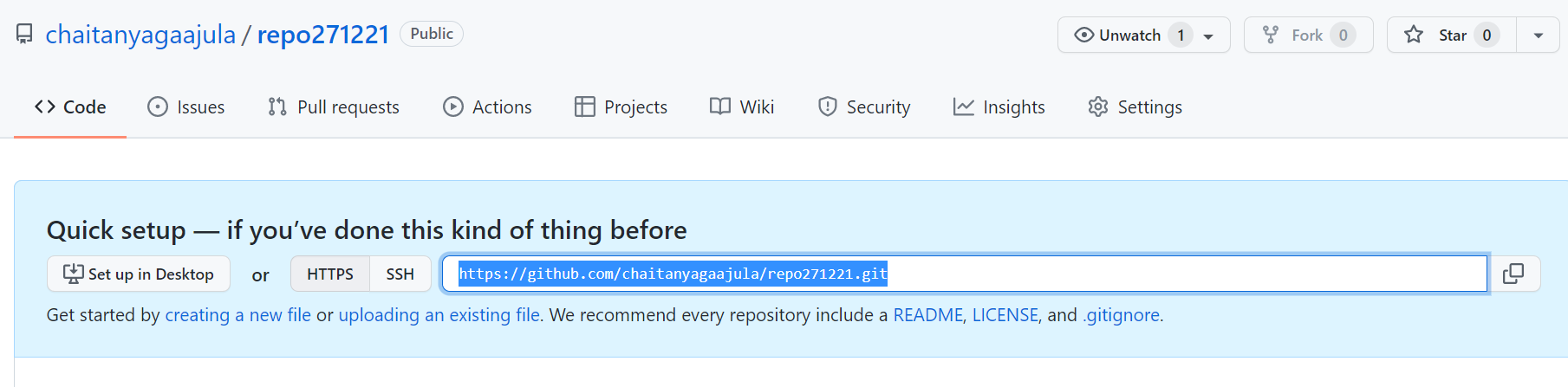
create mode 100644 example.txt

**Create a Public Repository in GitHub(Remote)**





Check if repository has been created and copy the GitHub repository(HTTPS) url



**In git ,add default remote GitHub URL in local**

git remote add origin GitHubRemoteURL

chait@LAPTOP-QHKO7HQH MINGW64 ~/project1 (master)

$ git remote add origin <https://github.com/chaitanyagaajula/repo271221.git>

**Check if the default origin has been added**

git remote -v

chait@LAPTOP-QHKO7HQH MINGW64 ~/project1 (master)

$ git remote -v

origin https://github.com/chaitanyagaajula/repo271221.git (fetch)

origin https://github.com/chaitanyagaajula/repo271221.git (push)

**Push changes from local(git) to remote (GitHub)**

git push -u origin master

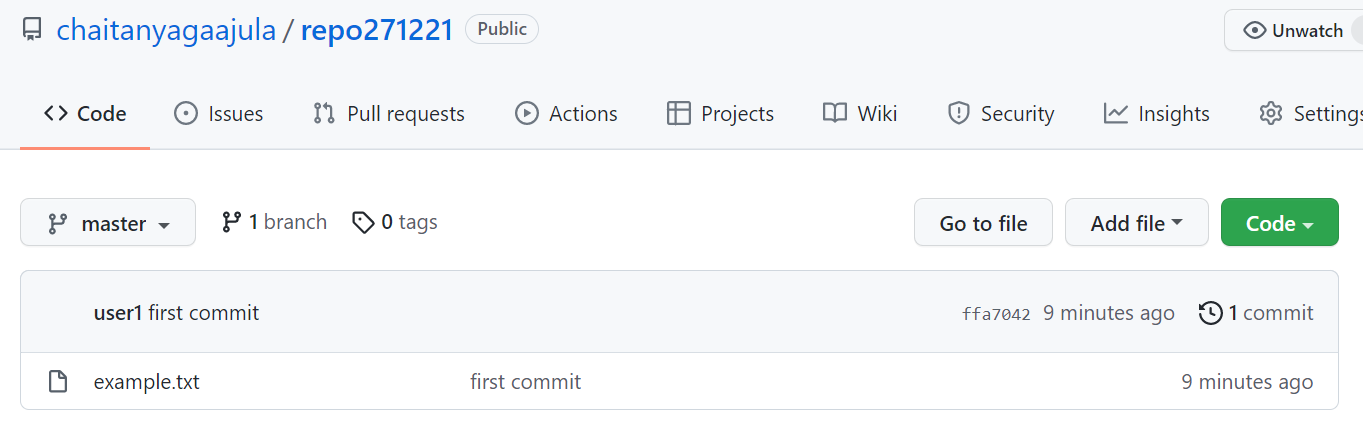
chait@LAPTOP-QHKO7HQH MINGW64 ~/project1 (master)

$ git push -u origin master

Everything up-to-date

Branch 'master' set up to track remote branch 'master' from 'origin'.

**Check if the file named example.txt has been ushed from git to Remote repository**



**Additional Git Commands**

**HEAD is a reference to the most recent commit in the current branch**

git show HEAD

chait@LAPTOP-QHKO7HQH MINGW64 ~/project1 (master)

$ git show HEAD

commit ffa70425b98e85ae974c782eda5000b342f3c6e6 (HEAD -> master, origin/master)

Author: user1 <user1@x.com>

Date: Mon Dec 27 11:18:43 2021 +0530

first commit

**diff --git a/example.txt b/example.txt**

**new file mode 100644**

**index 0000000..20aeba2**

**--- /dev/null**

**+++ b/example.txt**

@@ -0,0 +1,3 @@

+first line

+second line

+third line

**Exit from the directory**

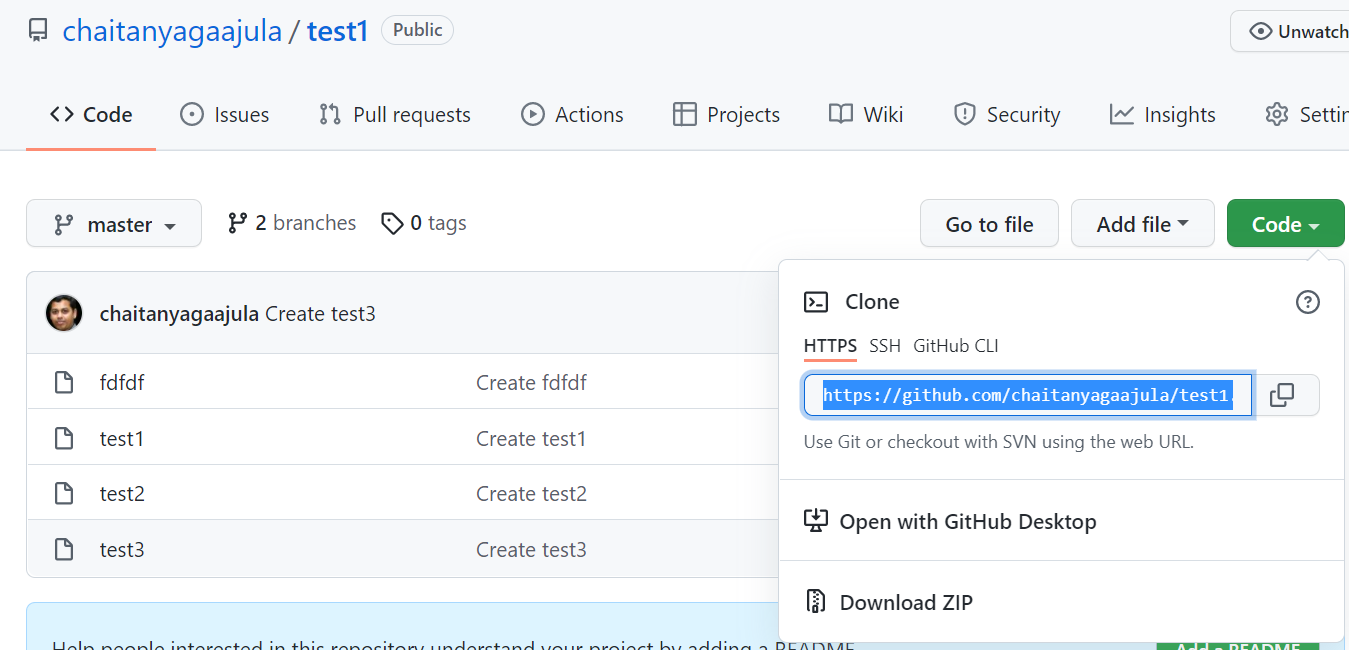
chait@LAPTOP-QHKO7HQH MINGW64 ~/project1 (master)

$ exit

**git clone is a Git command line utility which is used to target an existing repository and create a clone, or copy of the target repository**

git clone remote url

Access the remote repository which you want to clone on local , click on **Code** button and copy the remote URL



**In local Git , perform the below operations**

git clone remoteurl

chait@LAPTOP-QHKO7HQH MINGW64 ~ (main)

$ git clone https://github.com/chaitanyagaajula/test1.git

Cloning into 'test1'...

remote: Enumerating objects: 12, done.

remote: Counting objects: 100% (12/12), done.

remote: Compressing objects: 100% (7/7), done.

remote: Total 12 (delta 2), reused 0 (delta 0), pack-reused 0

Unpacking objects: 100% (12/12), 2.36 KiB | 43.00 KiB/s, done.

**Check if the contents have been downloaded**

chait@LAPTOP-QHKO7HQH MINGW64 ~ (main)

$ cd test1

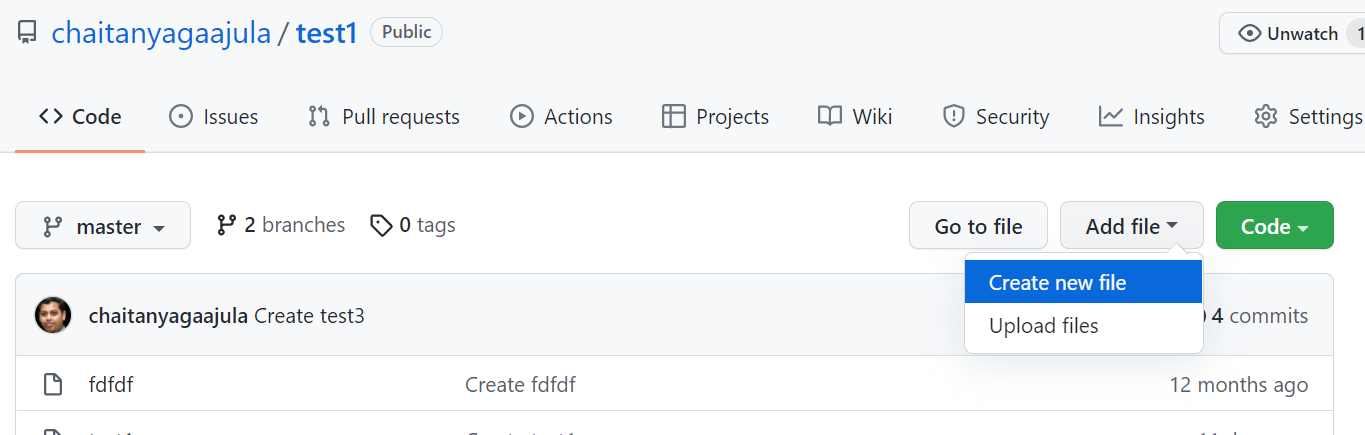
chait@LAPTOP-QHKO7HQH MINGW64 ~/test1 (master)

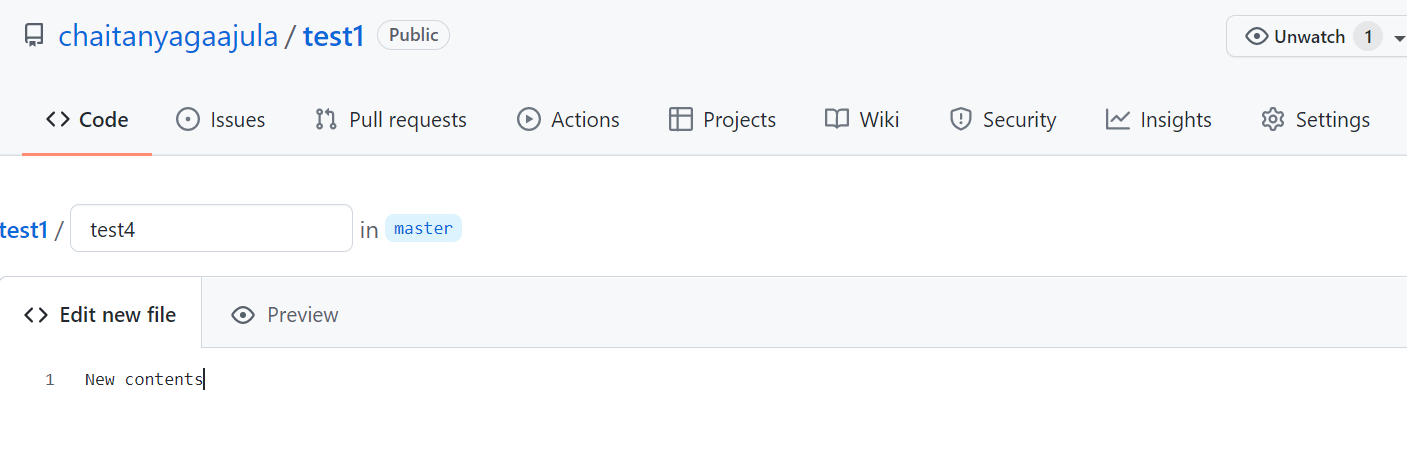
$ ls

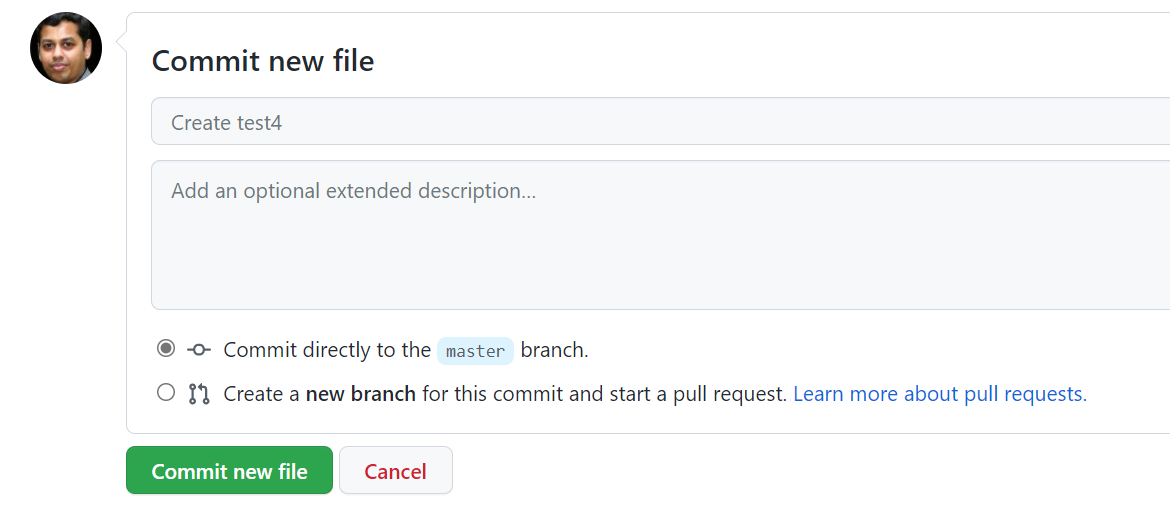
fdfdf test1 test2 test3

**Working with git Pull**

Create a new file in test remote repository and commit file







**git pull, in contrast, is used with a different goal in mind: to update your current HEAD branch with the latest changes from the remote server. This means that pull not only downloads new data; it also directly integrates it into your current working copy files. This has a couple of consequences:**

git pull remote url

chait@LAPTOP-QHKO7HQH MINGW64 ~/test1 (master)

$ git pull https://github.com/chaitanyagaajula/test1.git

remote: Enumerating objects: 4, done.

remote: Counting objects: 100% (4/4), done.

remote: Compressing objects: 100% (2/2), done.

remote: Total 3 (delta 1), reused 0 (delta 0), pack-reused 0

Unpacking objects: 100% (3/3), 648 bytes | 38.00 KiB/s, done.

From https://github.com/chaitanyagaajula/test1

\* branch HEAD -> FETCH\_HEAD

Updating 17ae2dd..9771564

Fast-forward

test4 | 1 +

1 file changed, 1 insertion(+)

create mode 100644 test4

chait@LAPTOP-QHKO7HQH MINGW64 ~/test1 (master)

$ ls -l

total 5

-rw-r--r-- 1 chait 197609 8 Dec 27 11:42 fdfdf

-rw-r--r-- 1 chait 197609 18 Dec 27 11:42 test1

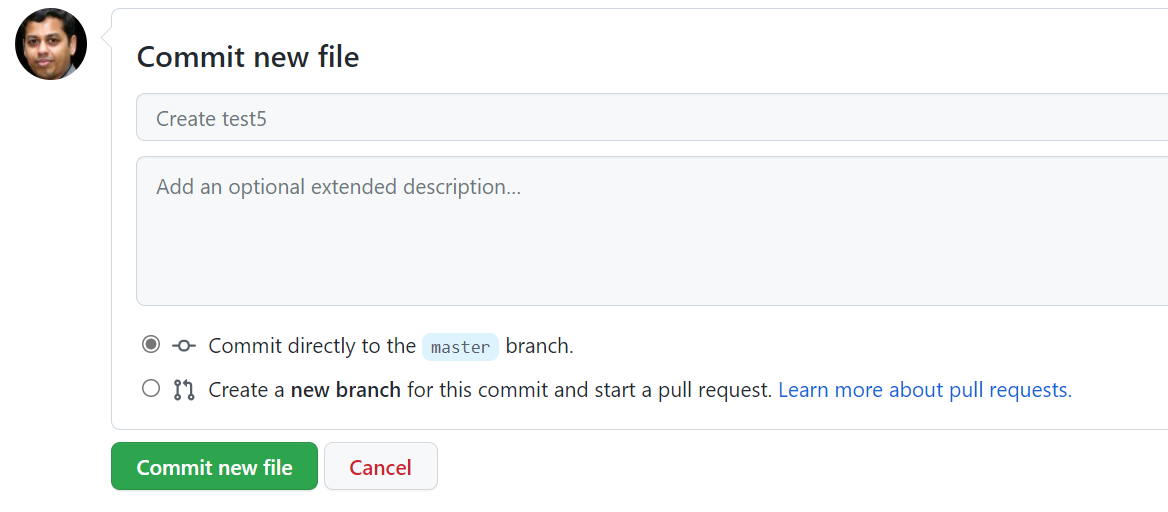
-rw-r--r-- 1 chait 197609 19 Dec 27 11:42 test2

-rw-r--r-- 1 chait 197609 8 Dec 27 11:42 test3

-rw-r--r-- 1 chait 197609 14 Dec 27 11:47 test4

**Note: We observed now that only test4 file has been pulled and previous file contents has not been modified**

**git fetch** really only downloads new data from a remote repository - but it doesn't integrate any of this new data into your working files. Fetch is great for getting a fresh view on all the things that happened in a remote repository.



Due to it's "harmless" nature, you can rest assured: fetch will never manipulate, destroy, or screw up anything. This means you can never fetch often enough.

git fetch remote url

chait@LAPTOP-QHKO7HQH MINGW64 ~/test1 (master)

$ git fetch https://github.com/chaitanyagaajula/test1.git

remote: Enumerating objects: 4, done.

remote: Counting objects: 100% (4/4), done.

remote: Compressing objects: 100% (2/2), done.

remote: Total 3 (delta 1), reused 0 (delta 0), pack-reused 0

Unpacking objects: 100% (3/3), 645 bytes | 20.00 KiB/s, done.

From https://github.com/chaitanyagaajula/test1

\* branch HEAD -> FETCH\_HEAD

chait@LAPTOP-QHKO7HQH MINGW64 ~/test1 (master)

$ ls -l

total 5

-rw-r--r-- 1 chait 197609 8 Dec 27 11:42 fdfdf

-rw-r--r-- 1 chait 197609 18 Dec 27 11:42 test1

-rw-r--r-- 1 chait 197609 19 Dec 27 11:42 test2

-rw-r--r-- 1 chait 197609 8 Dec 27 11:42 test3

-rw-r--r-- 1 chait 197609 14 Dec 27 11:47 test4

**Note: Observe the fetch command does not modify the directory contents**

**Shows file differences after staging**

git diff filename

Create a newfile and add contents into it

vi test7.txt

new1

new2

new3

new4

new5

chait@LAPTOP-QHKO7HQH MINGW64 ~/test1 (master)

$ cat test7.txt

new1

new2

new3

new4

new5

**Add the file to staging area**

chait@LAPTOP-QHKO7HQH MINGW64 ~/test1 (master)

$ git add test7.txt

Make changes in the file

Remove the line named as new5 and save

chait@LAPTOP-QHKO7HQH MINGW64 ~/test1 (master)

$ git diff

warning: LF will be replaced by CRLF in test7.txt.

The file will have its original line endings in your working directory

**diff --git a/test7.txt b/test7.txt**

**index 06d8d5e..c0aa3c9 100644**

**--- a/test7.txt**

**+++ b/test7.txt**

@@ -2,6 +2,6 @@ new1

new2

new3

new4

-new5

+

**To reset the files which are in staging back to working directory**

git reset

chait@LAPTOP-QHKO7HQH MINGW64 ~/test1 (master)

$ git reset test7.txt

chait@LAPTOP-QHKO7HQH MINGW64 ~/test1 (master)

$ git status

On branch master

Your branch is ahead of 'origin/master' by 1 commit.

(use "git push" to publish your local commits)

Untracked files:

(use "git add <file>..." to include in what will be committed)

test6.txt

test7.txt

nothing added to commit but untracked files present (use "git add" to track)

**Shows all commit details”**

git log

$ git log

commit 9771564c42268a34f2f28aa639ce1a94781f7f4c (HEAD -> master)

Author: Chaitanya R Gaajula <chaitanya@gaajula.com>

Date: Mon Dec 27 11:46:00 2021 +0530

Create test4

commit 17ae2dd8f411ed9b7eeda43d89e78522ff418973 (origin/master, origin/HEAD)

Author: Chaitanya R Gaajula <chaitanya@gaajula.com>

Date: Thu Dec 16 23:31:00 2021 +0530

Create test3

commit 4e6160e3ff486a68c08a1209b23a97f7bf9060f0

Author: Chaitanya R Gaajula <chaitanya@gaajula.com>

Date: Thu Dec 16 23:29:22 2021 +0530

Create test2

commit 88c73a2b1d4ee64d0ab88ea8d701841e96fed531

Author: Chaitanya R Gaajula <chaitanya@gaajula.com>

Date: Thu Dec 16 23:27:10 2021 +0530

**Working with Branches:**

**Listing Branches:**

git branch

chait@LAPTOP-QHKO7HQH MINGW64 ~ (branch1)

$ cd project1

chait@LAPTOP-QHKO7HQH MINGW64 ~/project1 (master)

$ git branch

\* master

**Creating branch to add a feature under project1**

git branch branchname

chait@LAPTOP-QHKO7HQH MINGW64 ~/project1 (master)

$ git branch feature1

chait@LAPTOP-QHKO7HQH MINGW64 ~/project1 (master)

$ git branch

feature1

\* master

**Accessing/Logging into a branch:**

git checkout branchname

chait@LAPTOP-QHKO7HQH MINGW64 ~/project1 (master)

$ git checkout feature1

Switched to branch 'feature1'

chait@LAPTOP-QHKO7HQH MINGW64 ~/project1 (feature1)

$ git branch

\* feature1

master

**Adding files in branch and then commit:**

touch filename

git add filename

git commit -m "feature branch changes"

chait@LAPTOP-QHKO7HQH MINGW64 ~/project1 (feature1)

$ ls

example.txt

chait@LAPTOP-QHKO7HQH MINGW64 ~/project1 (feature1)

$ touch new.txt

chait@LAPTOP-QHKO7HQH MINGW64 ~/project1 (feature1)

$ git add new.txt

chait@LAPTOP-QHKO7HQH MINGW64 ~/project1 (feature1)

$ git commit -m "feature1 changes"

[feature1 2c9c259] feature1 changes

1 file changed, 0 insertions(+), 0 deletions(-)

create mode 100644 new.txt

**Merging branches:**

git checkout master

git merge branchname

chait@LAPTOP-QHKO7HQH MINGW64 ~/project1 (feature1)

$ git checkout master

Switched to branch 'master'

Your branch is up to date with 'origin/master'.

chait@LAPTOP-QHKO7HQH MINGW64 ~/project1 (master)

$ git merge feature1

Updating ffa7042..2c9c259

Fast-forward

new.txt | 0

1 file changed, 0 insertions(+), 0 deletions(-)

create mode 100644 new.txt

chait@LAPTOP-QHKO7HQH MINGW64 ~/project1 (master)

$ ls

example.txt new.txt

**Note: We observe that changes has been pushed from branch named feature1 to master**

**Check o/p using git log**

chait@LAPTOP-QHKO7HQH MINGW64 ~/project1 (master)

$ git log

commit 2c9c2599b1850f774a8a5a9d17cc90062dfa9765 (HEAD -> master, feature1)

Author: user1 <user1@x.com>

Date: Mon Dec 27 12:22:38 2021 +0530

feature1 changes

commit ffa70425b98e85ae974c782eda5000b342f3c6e6 (origin/master)

Author: user1 <user1@x.com>

Date: Mon Dec 27 11:18:43 2021 +0530

first commit

**Delete branch:**

git branch -d branchname

chait@LAPTOP-QHKO7HQH MINGW64 ~/project1 (master)

$ git branch -d feature1

Deleted branch feature1 (was 2c9c259).

chait@LAPTOP-QHKO7HQH MINGW64 ~/project1 (master)

$ git branch

\* master

**Revert the latest commit:**

git revert commitid

git revert ffa70425b98e85ae974c782eda5000b342f3c6e6

It will open up a new screen , confirm the changes by doing esc:wq

$ git revert ffa70425b98e85ae974c782eda5000b342f3c6e6

Removing example.txt

[master 315354e] Revert "first commit"

1 file changed, 3 deletions(-)

delete mode 100644 example.txt

chait@LAPTOP-QHKO7HQH MINGW64 ~/project1 (master)

$ ls

new.txt

chait@LAPTOP-QHKO7HQH MINGW64 ~/project1 (master)

$ git log

commit 315354ebaaab3cd31eb1ff434da22822412d43cb (HEAD -> master)

Author: user1 <user1@x.com>

Date: Mon Dec 27 12:26:06 2021 +0530

Revert "first commit"

This reverts commit ffa70425b98e85ae974c782eda5000b342f3c6e6.

commit 2c9c2599b1850f774a8a5a9d17cc90062dfa9765

Author: user1 <user1@x.com>

Date: Mon Dec 27 12:22:38 2021 +0530

feature1 changes

commit ffa70425b98e85ae974c782eda5000b342f3c6e6 (origin/master)

Author: user1 <user1@x.com>

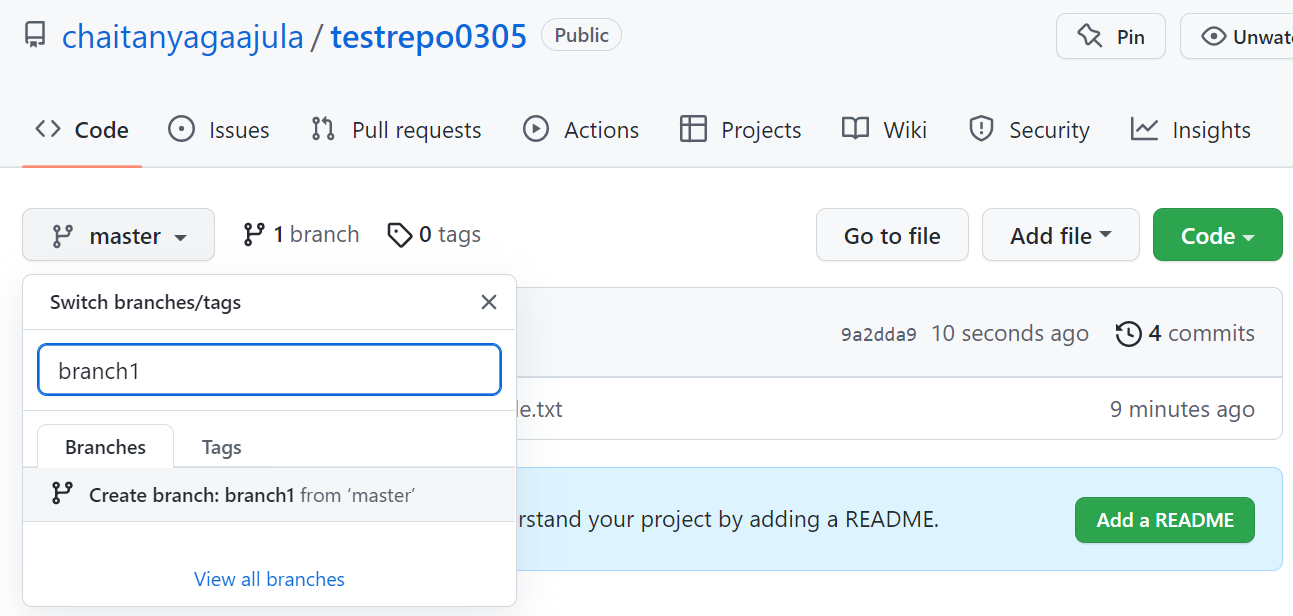
Date: Mon Dec 27 11:18:43 2021 +0530

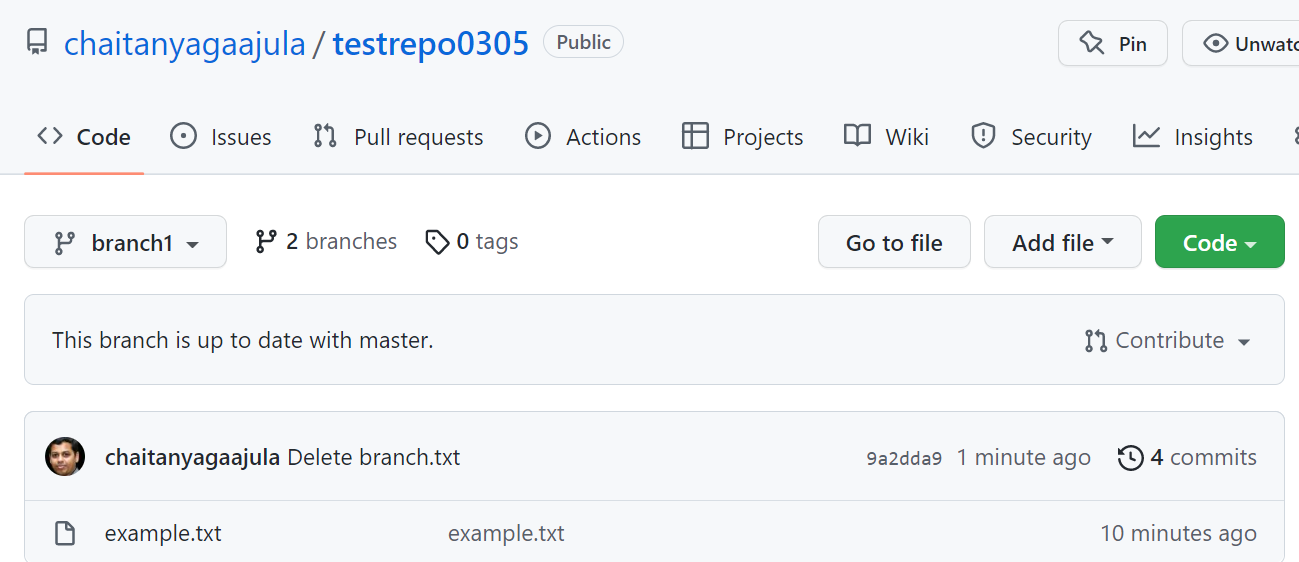
first commit

**Note: Since we reverted back to the previous commit , example.file is not being shown as current HEAD , it has reverted back to the previous commit**

**Working with GitHub Branches**

1.Create a branch named branch1 in existing repo .Observe all files from master branch has been copied to branch1 .





**On git perform the below operations**

**Check the remote origin details**

chait@LAPTOP-QHKO7HQH MINGW64 ~/project0305 (master)

$ git remote -v

origin https://github.com/chaitanyagaajula/testrepo0305.git (fetch)

origin https://github.com/chaitanyagaajula/testrepo0305.git (push)

**Pull the recent changes**

chait@LAPTOP-QHKO7HQH MINGW64 ~/project0305 (master)

$ git pull origin master

From https://github.com/chaitanyagaajula/testrepo0305

\* branch master -> FETCH\_HEAD

Already up to date.

**Create a new file named branch and commit to local repository**

chait@LAPTOP-QHKO7HQH MINGW64 ~/project0305 (master)

$ touch branch.txt

chait@LAPTOP-QHKO7HQH MINGW64 ~/project0305 (master)

$ git add branch.txt

chait@LAPTOP-QHKO7HQH MINGW64 ~/project0305 (master)

$ git commit -m "Need to push to branch in GitHub"

[master a2ecba9] Need to push to branch in GitHub

1 file changed, 0 insertions(+), 0 deletions(-)

create mode 100644 branch.txt

**Let us now push the new file to branch1 in GitHub**

chait@LAPTOP-QHKO7HQH MINGW64 ~/project0305 (master)

$ git push -u origin master:branch1

Enumerating objects: 7, done.

Counting objects: 100% (7/7), done.

Delta compression using up to 8 threads

Compressing objects: 100% (5/5), done.

Writing objects: 100% (5/5), 629 bytes | 209.00 KiB/s, done.

Total 5 (delta 1), reused 0 (delta 0), pack-reused 0

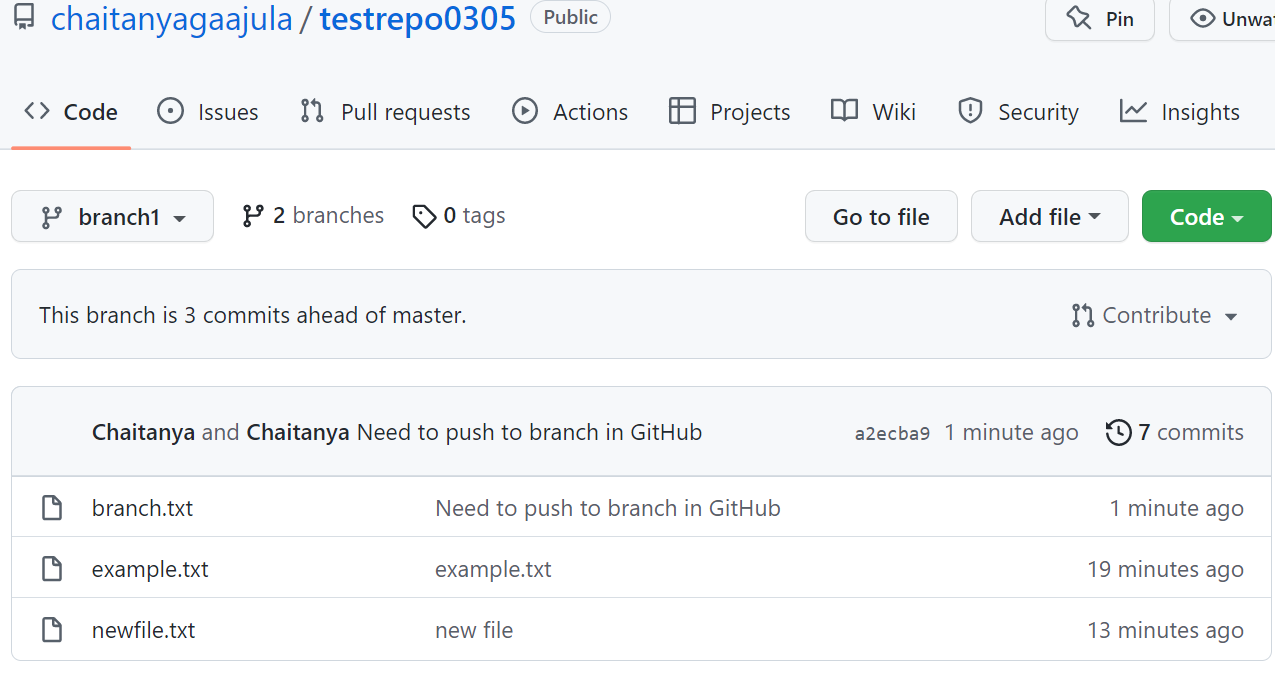
remote: Resolving deltas: 100% (1/1), done.

To https://github.com/chaitanyagaajula/testrepo0305.git

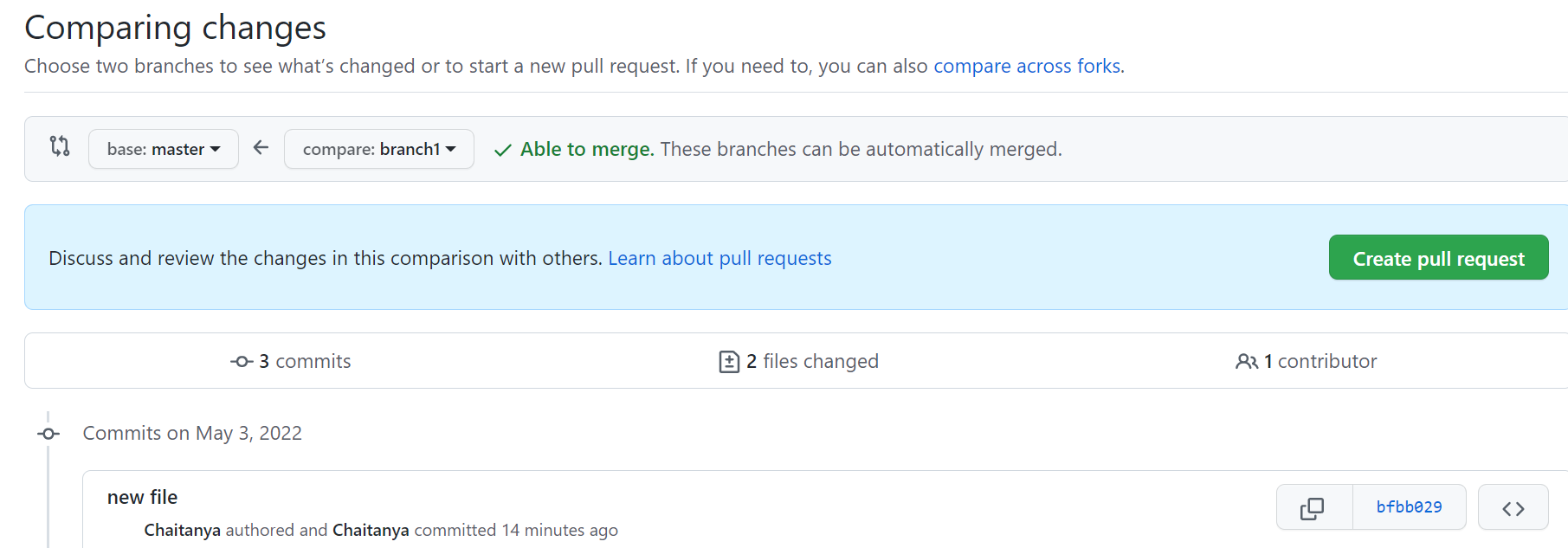
9a2dda9..a2ecba9 master -> branch1

branch 'master' set up to track 'origin/branch1'.

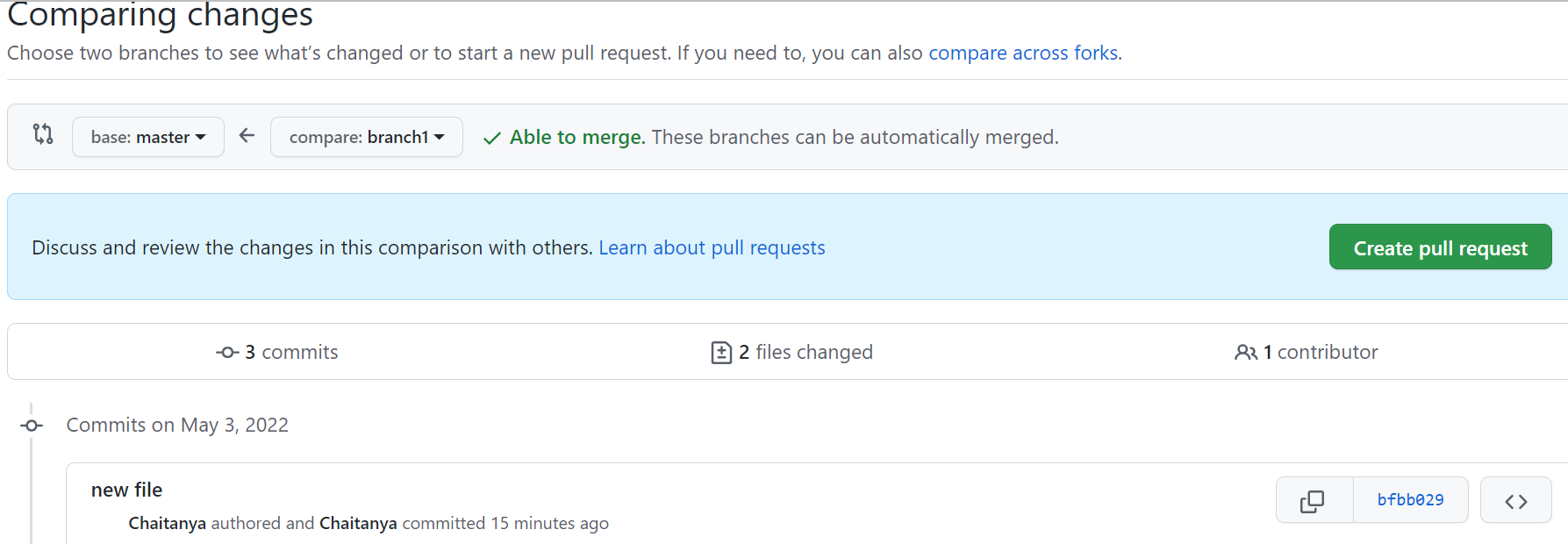
**Observe the new file is showing in branch1 in GitHub**



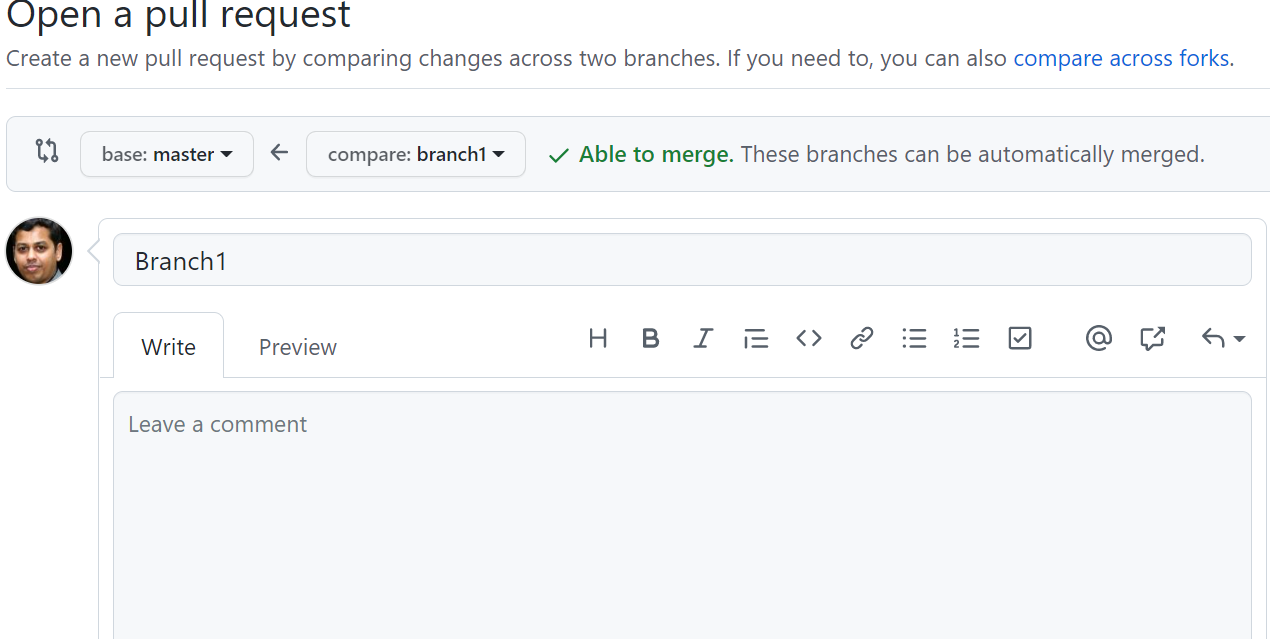
* **Create Pull Request, Click on Pull Request and under base choose master and branch 1 under compare**
* **Click on Create Pull Request button**

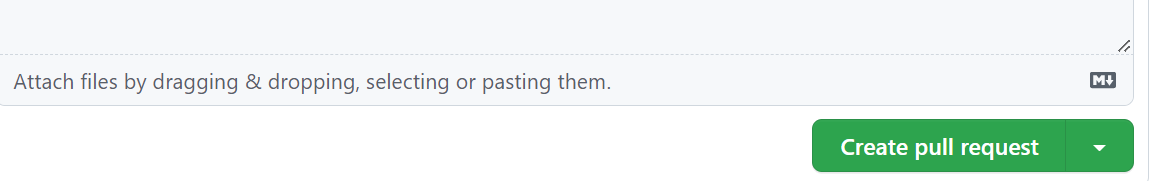


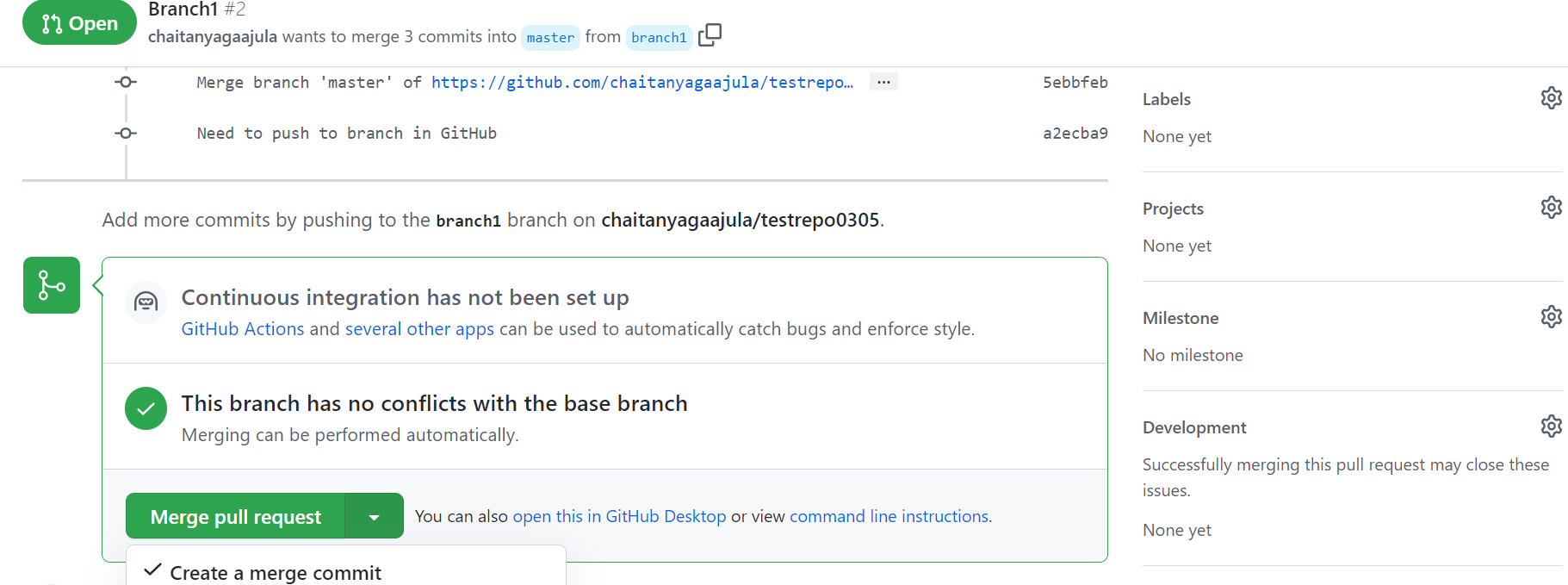
**Compare the changes by Clicking on Create Pull Request**

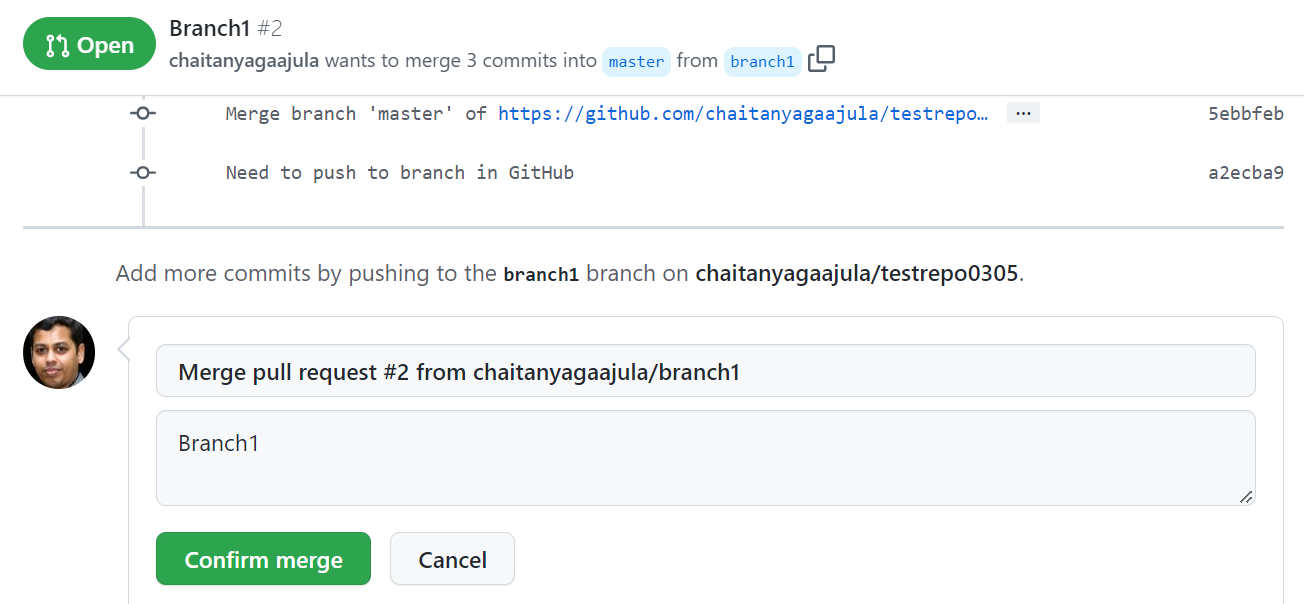


**Open a pull request and merge the changes to Master**









**Observe if the files has been merged on Master**

