Source Code Management

Task 1.1

&

Task 1.2

&

Task 2

(CS181)

Submitted by

Name-Aanchal Sharma Roll No.- 2110990009

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	**Beta**	Group No	G01
Course Coordinator	**Dr. Neeraj Singla**		
Faculty Name	**Mr. Monit Kapoor**		

TASK 1.1

S.no.	Title
1.	Setting up of Git Client
2.	Setting up of GitHub Account
3.	Generate logs
4.	Create and visualize branches
5.	Git lifecycle description

Experiment No. 01

Aim: Setting up of Git Client

Theory:

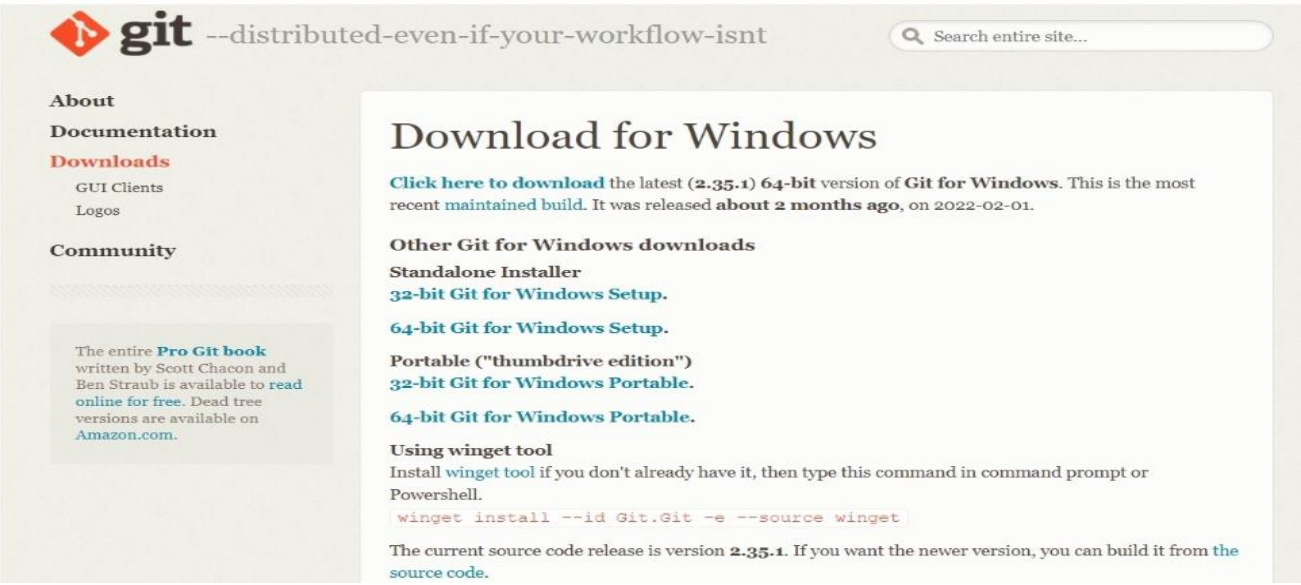
GIT –> It is basically used for pushing and pulling of code. We can use git and git-hub parallelly to work with multiple members or individually. We can make , edit , recreate ,copy or download any code on git hub using git.

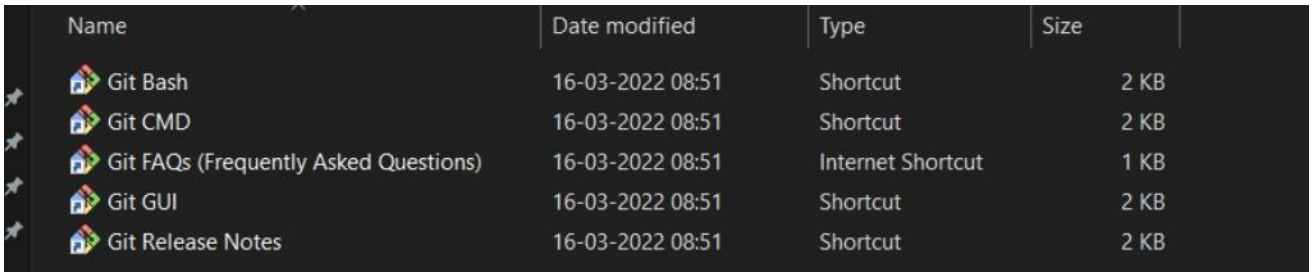
What is GIT ? –> It's a Version Control System(VCS) -> It is a software or we can say a server by which we are able to track all the previous changes in the code.

Advantages of GIT –>

Procedure: We can install Git on Windows, using the most official build which is available for download on the GIT's official website or by just typing (s c m git) on any search engine . We can go on https://git-scm.com/download/win and can select the platform and bit-version to download. And after clicking on your desired bit-version or ios it will start downloading automatically.

Snapshots of download:




Name
Date modified
Type
Size
Git Bash
16-03-2022 08:51
Shortcut
2 KB
Git CMD
16-03-2022 08:51
Shortcut
2 KB
Git FAQs (Frequently Asked Questions)
16-03-2022 08:51
Internet Shortcut
1 KB
Git GUI
16-03-2022 08:51
Shortcut
2 KB
Git Release Notes
16-03-2022 08:51
Shortcut
2 KB


MINGW64:/c/Users/HP
HP@DESKTOP-I9TBD2O MINGW64 ~
$

Experiment No. 02

Aim: Setting up GitHub Account

Theory:

What is GitHub -> GitHub is a website and cloud-based service (client) that helps an individual or a developers to store and manage their code. We can also track as well as control changes to our or public code.

Advantages of GitHub -> GitHub's has a user-friendly interface and is easy to use .We can connect the git-hub and git but using some commands shown below in figure 001. Without GitHub we cannot use Git because it generally requires a host and if we are working for a project we need to share it will our team members, which can only be done by making a repository . Additionally , anyone can sign up and host a public code repository for free, which makes GitHub especially popular with open-source projects.

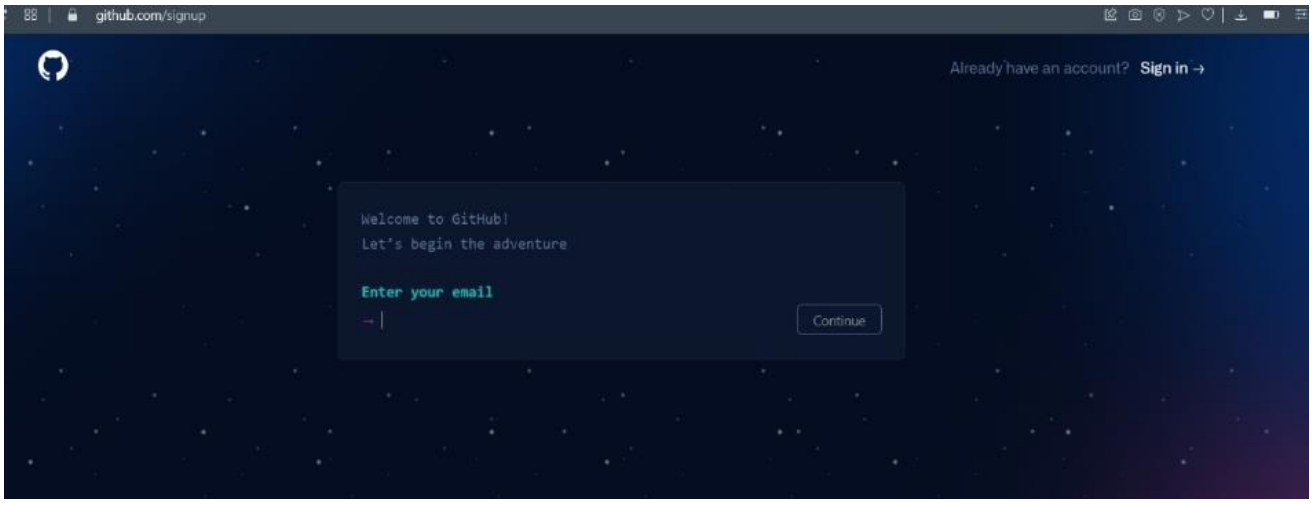
Procedure:-

Step1 :-

Google (any search engine)
Search for git-hub or (https://github.com/signup).

Step2 :-

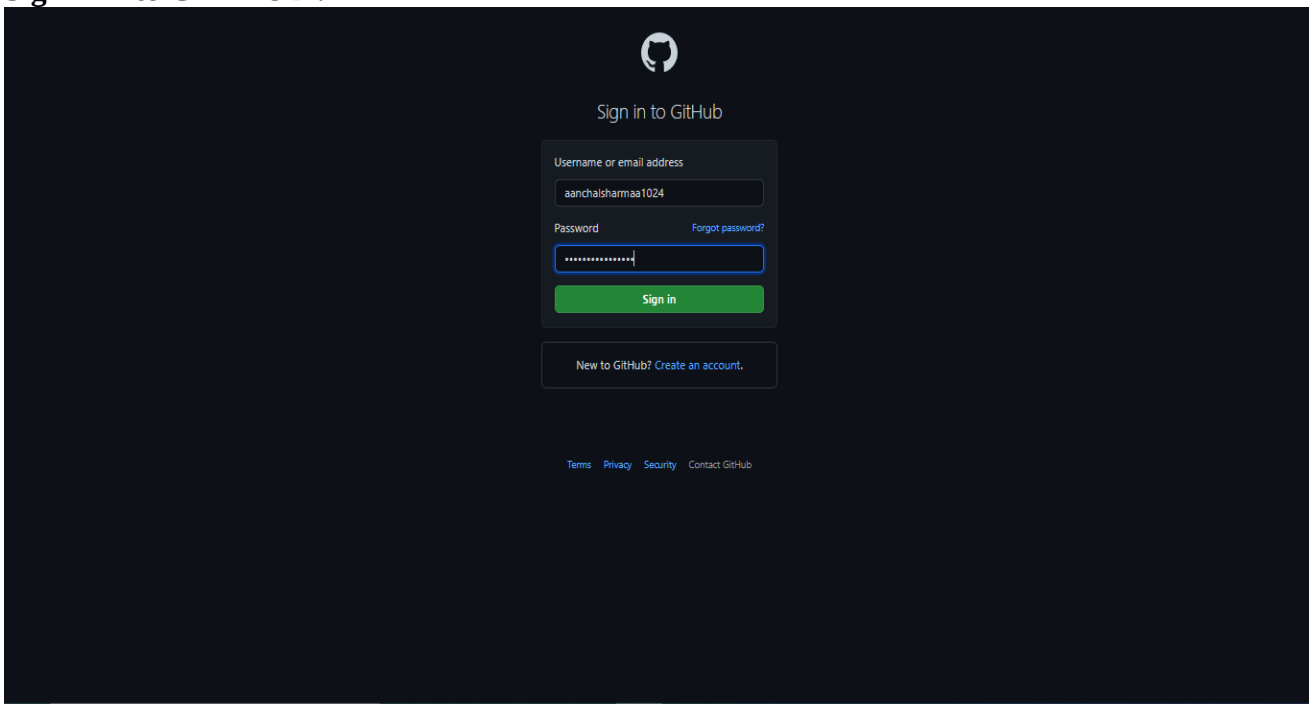
Snapshots –



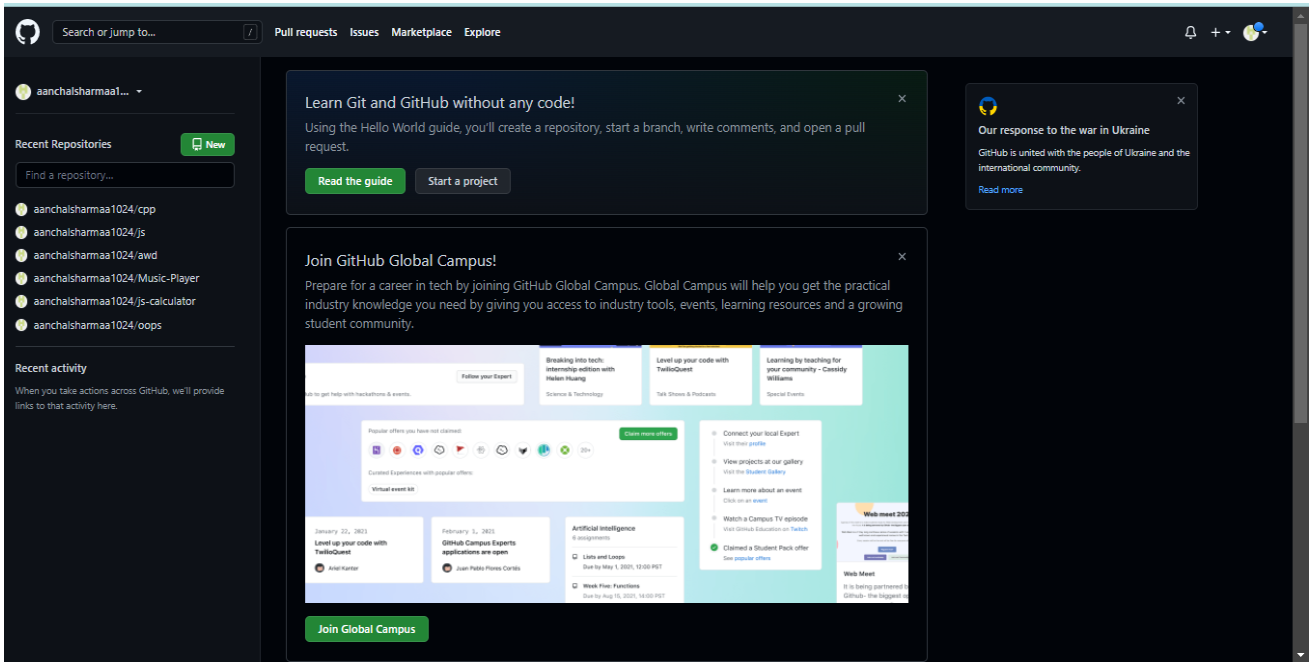
After visiting the link this type of interface will appear, if you already have account you can sign in

and if not you can create.

Sign in into GIT-HUB :-



Interface of GitHub :-



To link GitHub account with Git bash –

For username:-

git config --global user.name “username in git-hub”

For user email:-

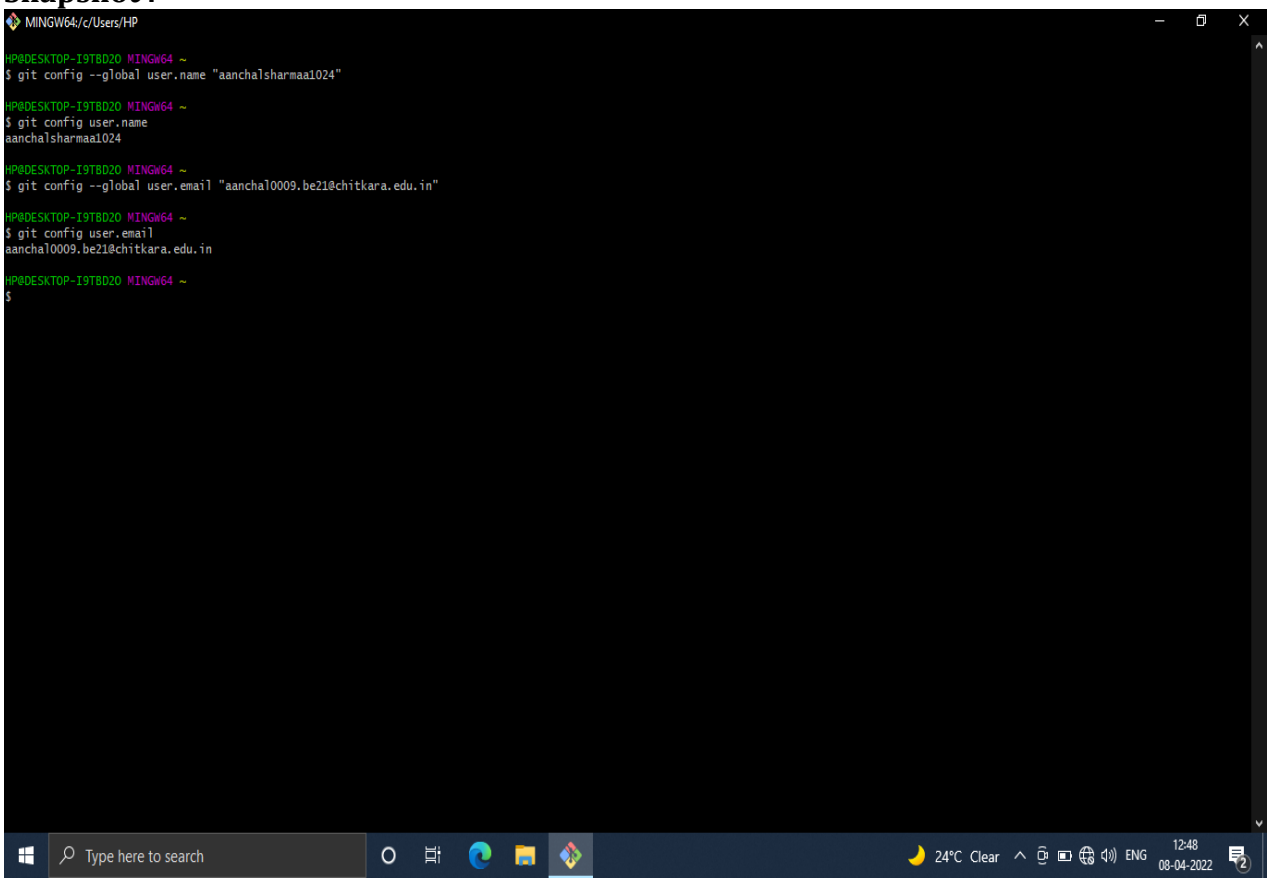
git config --global user.email “your email in git-hub”

To verify:-

git config user.name

git config user.email

Snapshot :-



Experiment No. 03

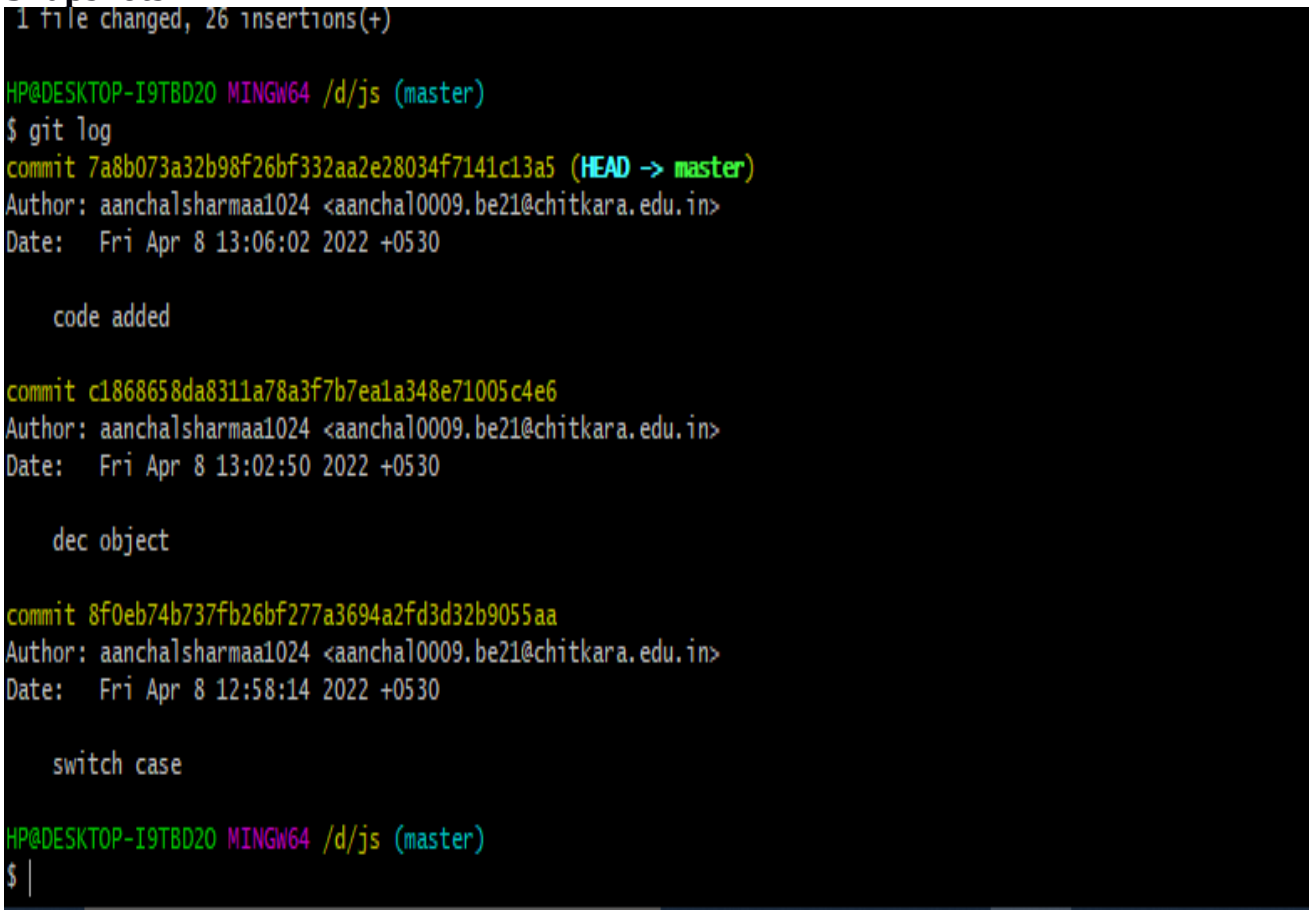
Aim: Program to Generate log

Theory:-

Logs -> Logs are nothing but the history which we can see in git by using the code git log.
It contains all the past commits, insertions and deletions in it which we can see any time.

Why logs -> Logs helps to check that what were the changes in the code or any other file and by whom. It also contains the number of insertions and deletions including at which time it was changed.

Snapshots –



Experiment No. 04

Aim: Create and visualize branches

Create branches :-

The main branch in git is called as master branch. But we can make branches out of this main master branch. All the files present in master can be shown in branch but the file which are created in branch are not shown in master branch. We can also merge both the parent (master) and child (other branches).

Syntax:-

1. For creating a new branch.

git branch name of branch , by default it is master branch

Snapshots –

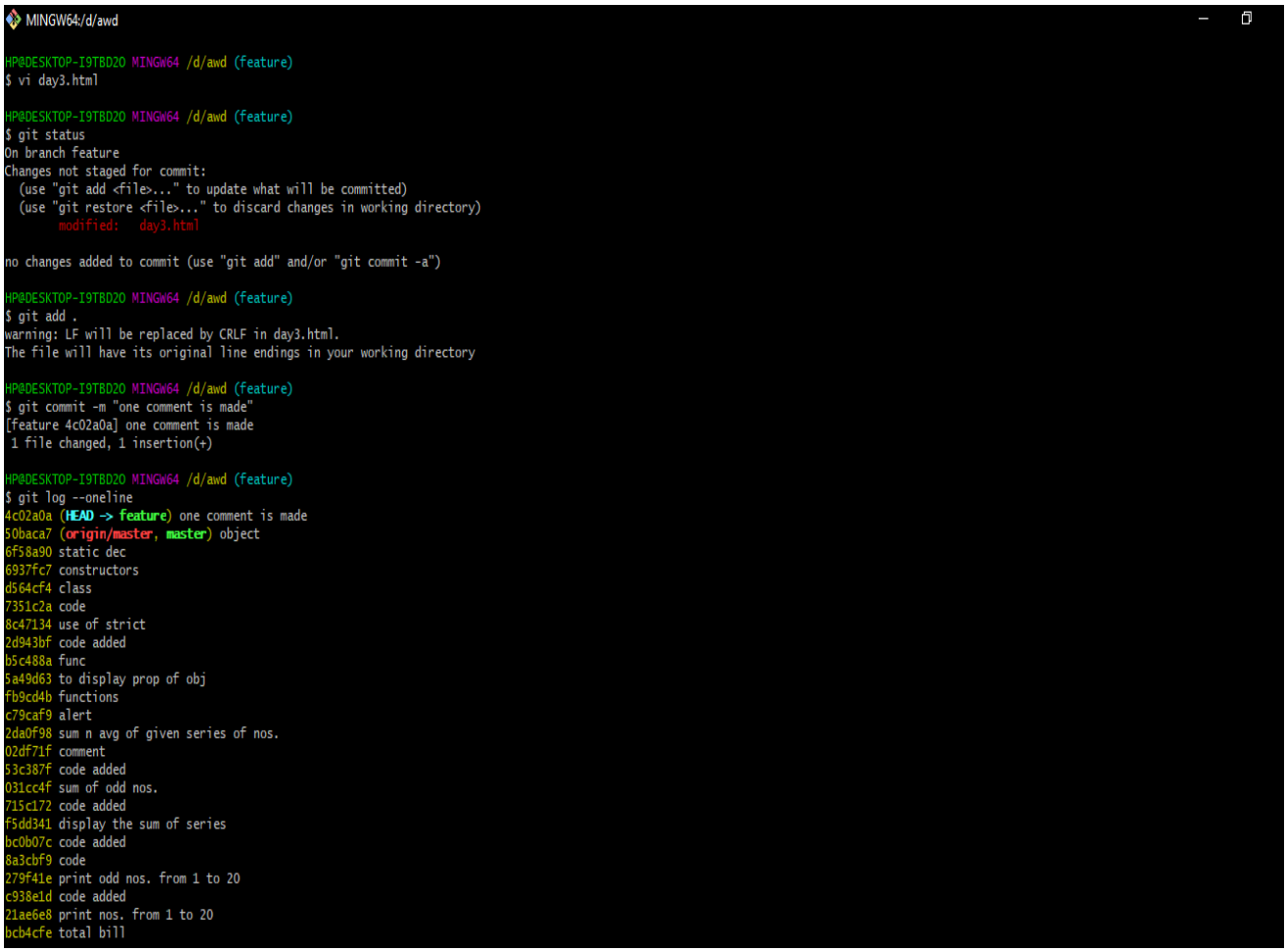
2. To change the present working branch.

git checkout name of branch.

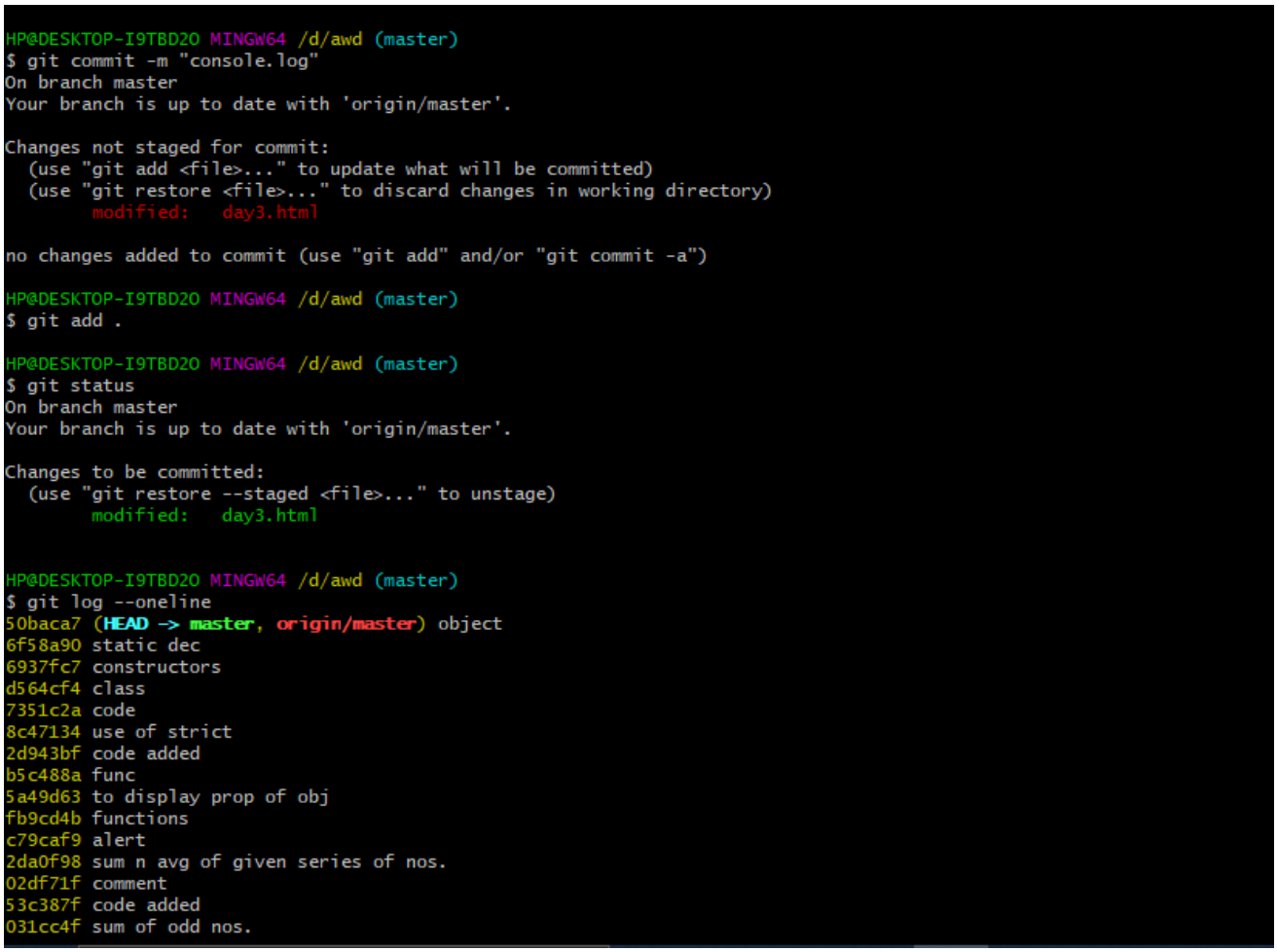
```
HP@DESKTOP-I9TBD2O MINGW64 /d/awd (master)
$ git branch feature

HP@DESKTOP-I9TBD2O MINGW64 /d/awd (master)
$ git branch
  feature
* master

HP@DESKTOP-I9TBD2O MINGW64 /d/awd (master)
$ git checkout feature
Switched to branch 'feature'

HP@DESKTOP-I9TBD2O MINGW64 /d/awd (feature)
$ git branch
* feature
  master
```

Visualizing branches :-

```
HP@DESKTOP-I9TBD2O MINGW64 /d/awd (feature)
$ git log --oneline
50baca7 (HEAD -> feature, origin/master, master) object
6f58a90 static dec
6937fc7 constructors
d564cf4 class
7351c2a code
8c47134 use of strict
2d943bf code added
b5c488a func
5a49d63 to display prop of obj
fb9cd4b functions
c79caf9 alert
2da0f98 sum n avg of given series of nos.
02df71f comment
53c387f code added
031cc4f sum of odd nos.
715c172 code added
f5dd341 display the sum of series
bc0b07c code added
8a3cbf9 code
279f41e print odd nos. from 1 to 20
c938e1d code added
21ae6e8 print nos. from 1 to 20
bcb4cfe total bill
6ec9c64 code added
22e1faa let var
7be5958 index of array
5ee9e36 code added
8e26823 swaping of two nos.
1d2293b swaping two nos.
a5da5be to display grades
f95025d reverse a number

HP@DESKTOP-I9TBD2O MINGW64 /d/awd (feature)
$
```


MINGW64:/d/awd
HP@DESKTOP-I9TBD2O MINGW64 /d/awd (feature)
$ vi day3.html
HP@DESKTOP-I9TBD2O MINGW64 /d/awd (feature)
$ git status
On branch feature
Changes not staged for commit:
(use "git add <file>..." to update what will be committed)
(use "git restore <file>..." to discard changes in working directory)
modified: day3.html
no changes added to commit (use "git add" and/or "git commit -a")
HP@DESKTOP-I9TBD2O MINGW64 /d/awd (feature)
$ git add .
warning: LF will be replaced by CRLF in day3.html.
The file will have its original line endings in your working directory
HP@DESKTOP-I9TBD2O MINGW64 /d/awd (feature)
$ git commit -m "one comment is made"
[feature 4c02a0a] one comment is made
1 file changed, 1 insertion(+)
HP@DESKTOP-I9TBD2O MINGW64 /d/awd (feature)
$ git log --oneline
4c02a0a (HEAD -> feature) one comment is made
50baca7 (origin/master, master) object
6f58a90 static dec
6937fc7 constructors
d564cf4 class
7351c2a code
8c47134 use of strict
2d943bf code added
b5c488a func
5a49d63 to display prop of obj
fb9cd4b functions
c79caf9 alert
2da0f98 sum n avg of given series of nos.
02df71f comment
53c387f code added
031cc4f sum of odd nos.
715c172 code added
f5dd341 display the sum of series
bc0b07c code added
8a3cbf9 code
279f41e print odd nos. from 1 to 20
c938e1d code added
21ae6e8 print nos. from 1 to 20
bcb4cfe total bill

```
HP@DESKTOP-I9TBD2O MINGW64 /d/awd (feature)
$ git checkout master
Switched to branch 'master'
Your branch is up to date with 'origin/master'.

HP@DESKTOP-I9TBD2O MINGW64 /d/awd (master)
$ git log --oneline
50baca7 (HEAD -> master, origin/master) object
6f58a90 static dec
6937fc7 constructors
d564cf4 class
7351c2a code
8c47134 use of strict
2d943bf code added
b5c488a func
5a49d63 to display prop of obj
fb9cd4b functions
c79caf9 alert
2da0f98 sum n avg of given series of nos.
02df71f comment
53c387f code added
031cc4f sum of odd nos.
715c172 code added
f5dd341 display the sum of series
bc0b07c code added
8a3cbf9 code
279f41e print odd nos. from 1 to 20
c938e1d code added
21ae6e8 print nos. from 1 to 20
bcb4cfe total bill
6ec9c64 code added
22e1faa let var
7be5958 index of array
5ee9e36 code added
8e26823 swaping of two nos.
1d2293b swaping two nos.
a5da5be to display grades
f95025d reverse a number

HP@DESKTOP-I9TBD2O MINGW64 /d/awd (master)
$ vi day3.html

HP@DESKTOP-I9TBD2O MINGW64 /d/awd (master)
$ git commit -m "console.log"
On branch master
Your branch is up to date with 'origin/master'.

Changes not staged for commit:
```



Experiment No. 05

Aim: Git lifecycle description

Theory:

Stages in GIT Life Cycle -> Files in a Git project have various stages like Creation, Modification, Refactoring, and Deletion and so on. Irrespective of whether this project is tracked by Git or not, these phases are still prevalent. However, when a project is under Git version control system, they are present in three major Git states in addition to these basic ones. Here are the three Git states:

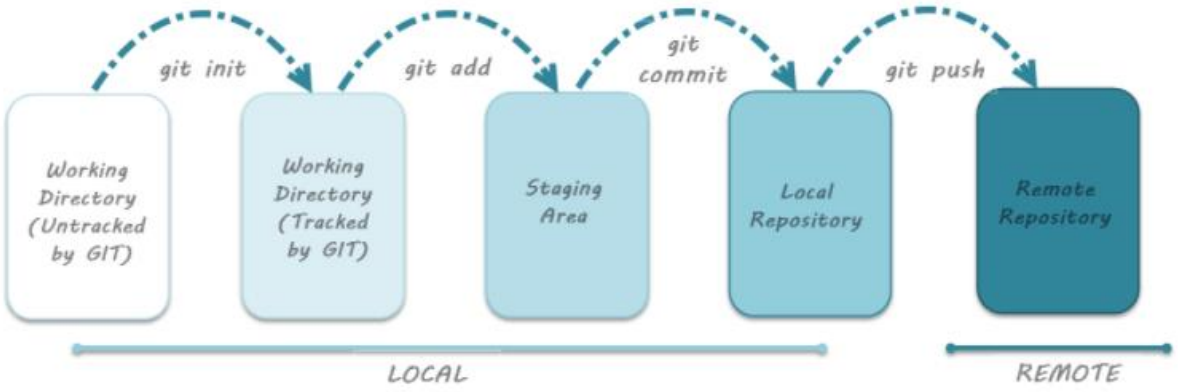
- Working directory
- Staging area
- Git directory

Working Directory ->
Consider a project residing in your local system. This project may or may not be tracked by Git. In either case, this project directory is called your Working directory.

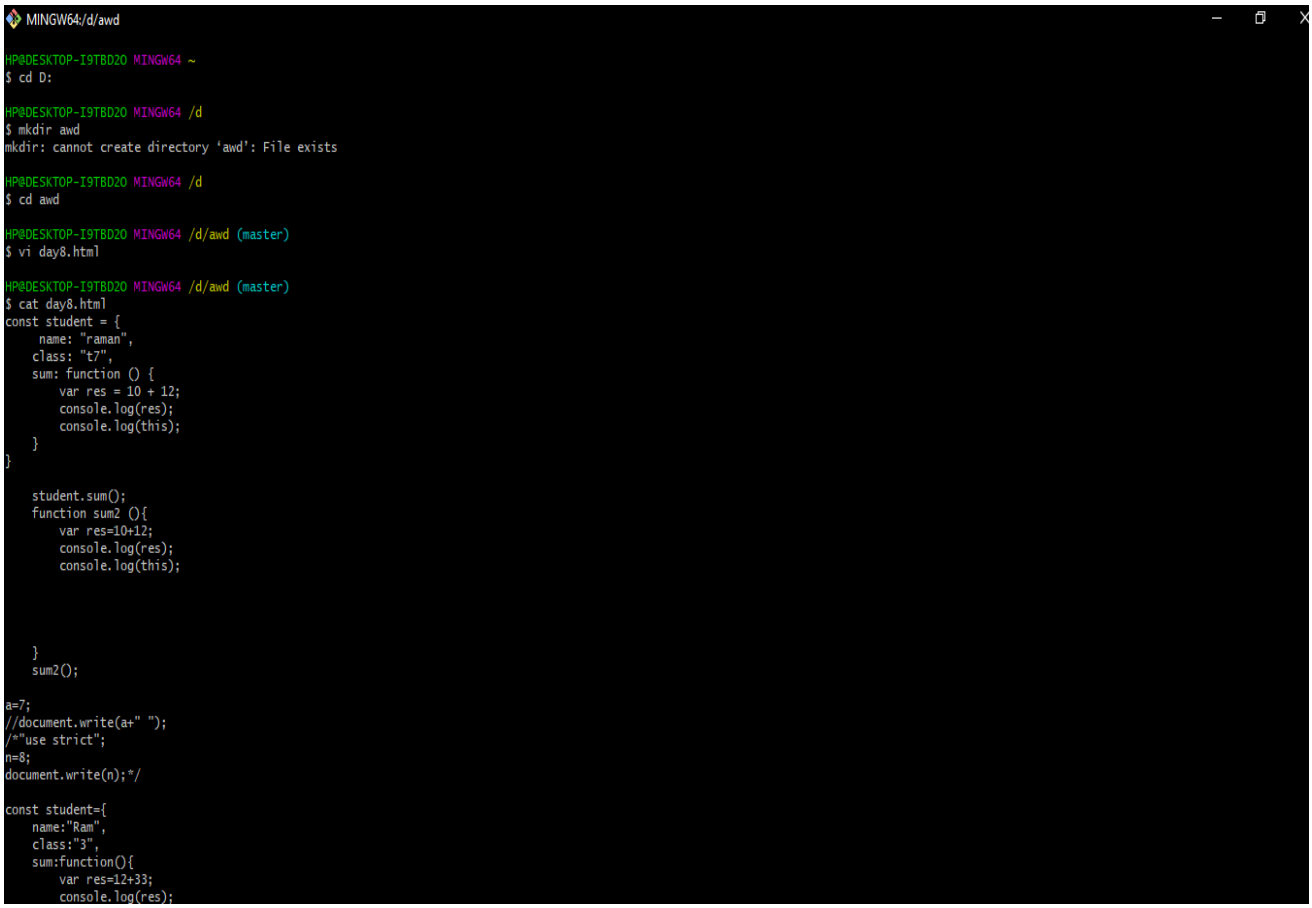
Staging Area ->
Staging area is the playground where you group, add and organize the files to be committed to Git for tracking their versions.

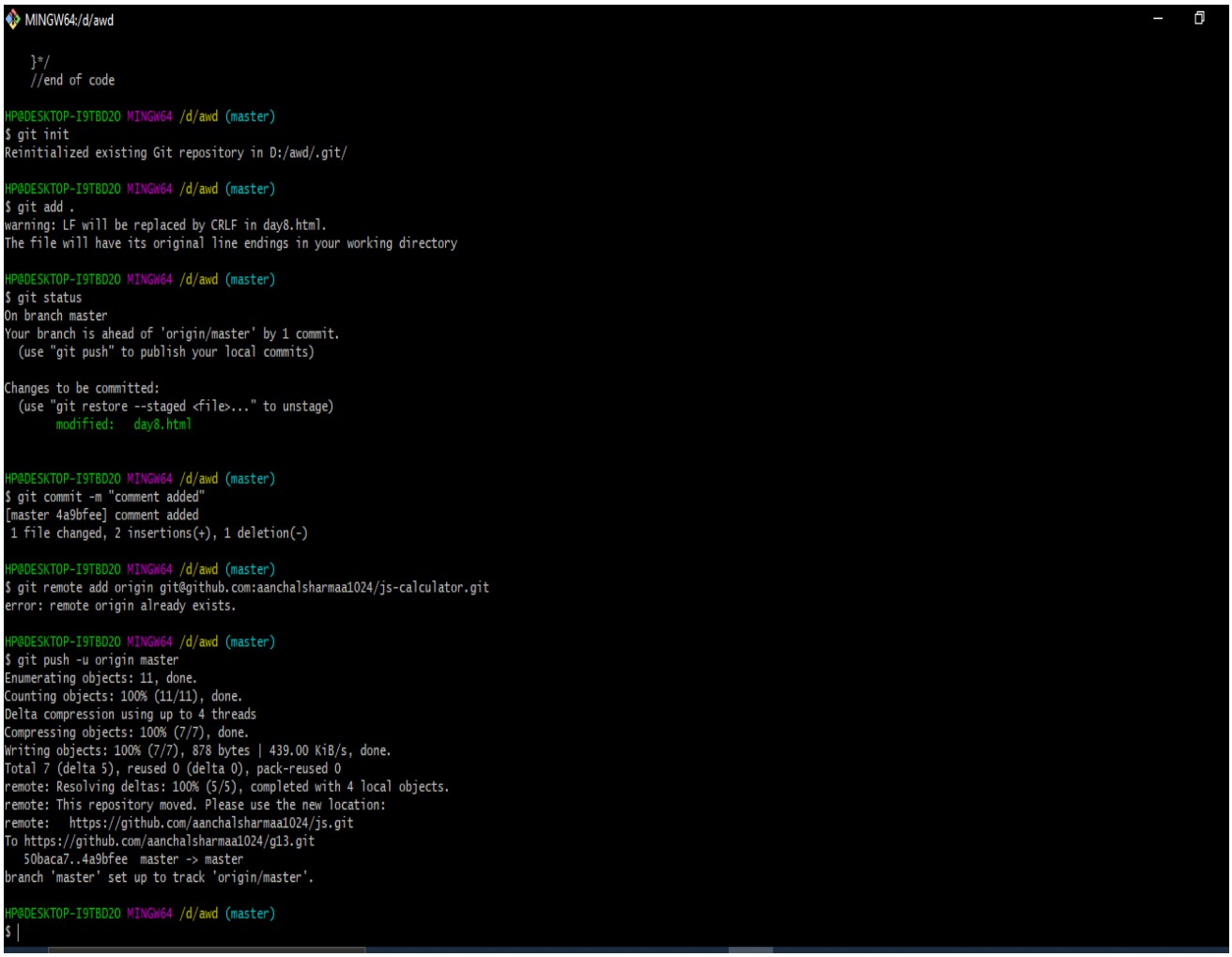
Git Directory ->
Now that the files to be committed are grouped and ready in the staging area, we can commit these files. So, we commit this group of files along with a commit message explaining what is the commit about. Apart from commit message, this step also records the author and time of the commit. Now, a snapshot of the files in the commit is recorded by Git. The information related to this commit is stored in the Git directory.

Remote Repository-> means mirror or clone of the local Git repository in GitHub. And **pushing means uploading the commits from local Git repository to remote repository hosted in GitHub.**



Snapshots –


MINGW64:/d/awd
HP@DESKTOP-I9TBD2O MINGW64 ~
$ cd D:
HP@DESKTOP-I9TBD2O MINGW64 /d
$ mkdir awd
mkdir: cannot create directory 'awd': File exists
HP@DESKTOP-I9TBD2O MINGW64 /d
$ cd awd
HP@DESKTOP-I9TBD2O MINGW64 /d/awd (master)
$ vi day8.html
HP@DESKTOP-I9TBD2O MINGW64 /d/awd (master)
$ cat day8.html
const student = {
name: "raman",
class: "t7",
sum: function () {
var res = 10 + 12;
console.log(res);
console.log(this);
}
}
student.sum();
function sum2 (){
var res=10+12;
console.log(res);
console.log(this);
}
sum2();
a=7;
//document.write(a+" ");
/*"use strict";
n=8;
document.write(n);*/
const student={
name:"Ram",
class:"3",
sum:function(){
var res=12+33;
console.log(res);


MINGW64:/d/awd
}*/
//end of code
HP@DESKTOP-I9TBD2O MINGW64 /d/awd (master)
$ git init
Reinitialized existing Git repository in D:/awd/.git/
HP@DESKTOP-I9TBD2O MINGW64 /d/awd (master)
$ git add .
warning: LF will be replaced by CRLF in day8.html.
The file will have its original line endings in your working directory
HP@DESKTOP-I9TBD2O MINGW64 /d/awd (master)
$ git status
On branch master
Your branch is ahead of 'origin/master' by 1 commit.
(use "git push" to publish your local commits)
Changes to be committed:
(use "git restore --staged <file>..." to unstage)
modified: day8.html
HP@DESKTOP-I9TBD2O MINGW64 /d/awd (master)
$ git commit -m "comment added"
[master 4a9bfee] comment added
1 file changed, 2 insertions(+), 1 deletion(-)
HP@DESKTOP-I9TBD2O MINGW64 /d/awd (master)
$ git remote add origin git@github.com:aanchalsharmaa1024/js-calculator.git
error: remote origin already exists.
HP@DESKTOP-I9TBD2O MINGW64 /d/awd (master)
$ git push -u origin master
Enumerating objects: 11, done.
Counting objects: 100% (11/11), done.
Delta compression using up to 4 threads
Compressing objects: 100% (7/7), done.
Writing objects: 100% (7/7), 878 bytes | 439.00 KiB/s, done.
Total 7 (delta 5), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (5/5), completed with 4 local objects.
remote: This repository moved. Please use the new location:
remote: https://github.com/aanchalsharmaa1024/js.git
To https://github.com/aanchalsharmaa1024/g13.git
50baca7..4a9bfee master -> master
branch 'master' set up to track 'origin/master'.
HP@DESKTOP-I9TBD2O MINGW64 /d/awd (master)
$

TASK 1.2

S. No.	Title
1	Add collaborators on GitHub Repo
2	Fork and Commit
3	Merge and Resolve conflicts created due to own activity and collaborators activity.
4	Reset and Revert

ADD COLLABORATORS ON GITHUB REPO

In GitHub, we can invite other GitHub users to become collaborators to our private repositories (which expires after 7 days if not accepted, restoring any unclaimed licenses). Being a collaborator, of a personal repository you can pull (read) the contents of the repository and push (write) changes to the repository. You can add unlimited collaborators on public and private repositories.

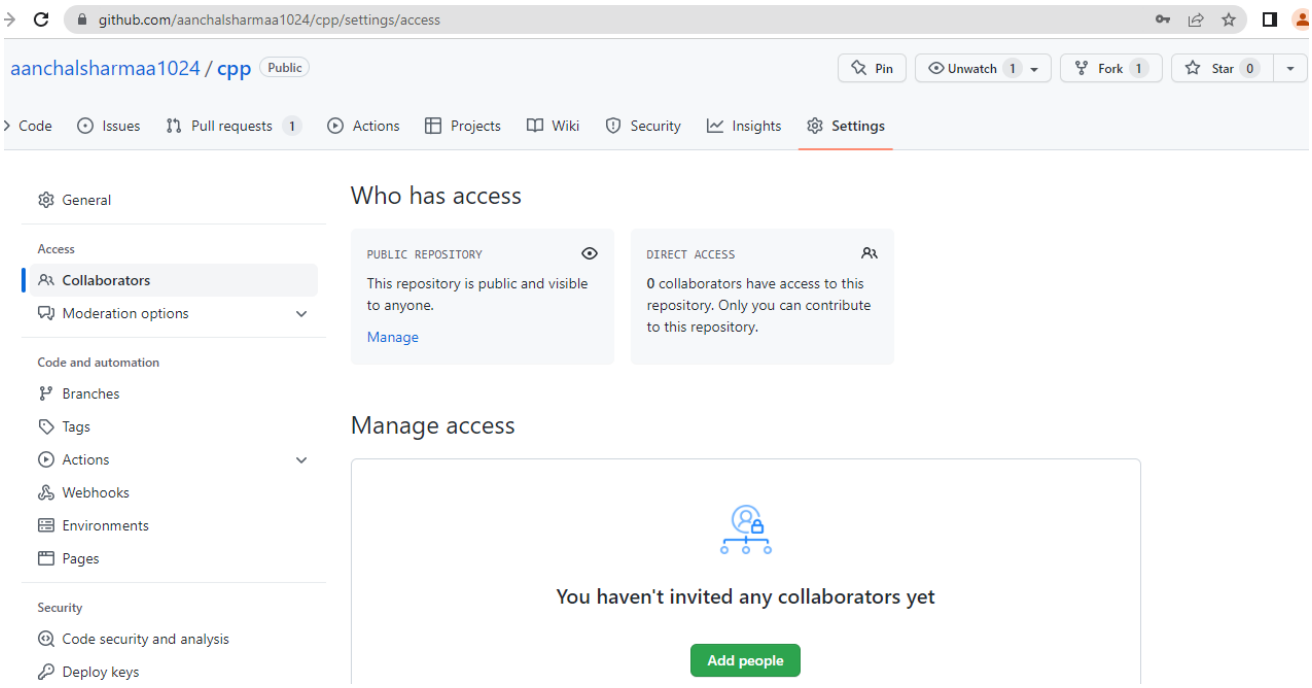
Collaborators can perform a number of actions into someone else's personal repositories, they have gained access to. Some of them are,

1. Create, merge, and close pull requests in the repository
2. Publish, view, install the packages
3. Fork the repositories
4. Make the changes on the repositories as suggested by the Pull requests.
5. Mark issues or pull requests as duplicate
6. Create, edit, and delete any comments on commits, pull requests, and issues in the repository
7. Removing themselves as collaborators on the repositories.
8. Manage releases in the repositories.

STEPS TO ADD COLLABORATORS:

1. Navigate to the repository on Github you wish to share with your collaborator.
2. Click on the "Settings" tab on the right side of the menu at the top of the screen.

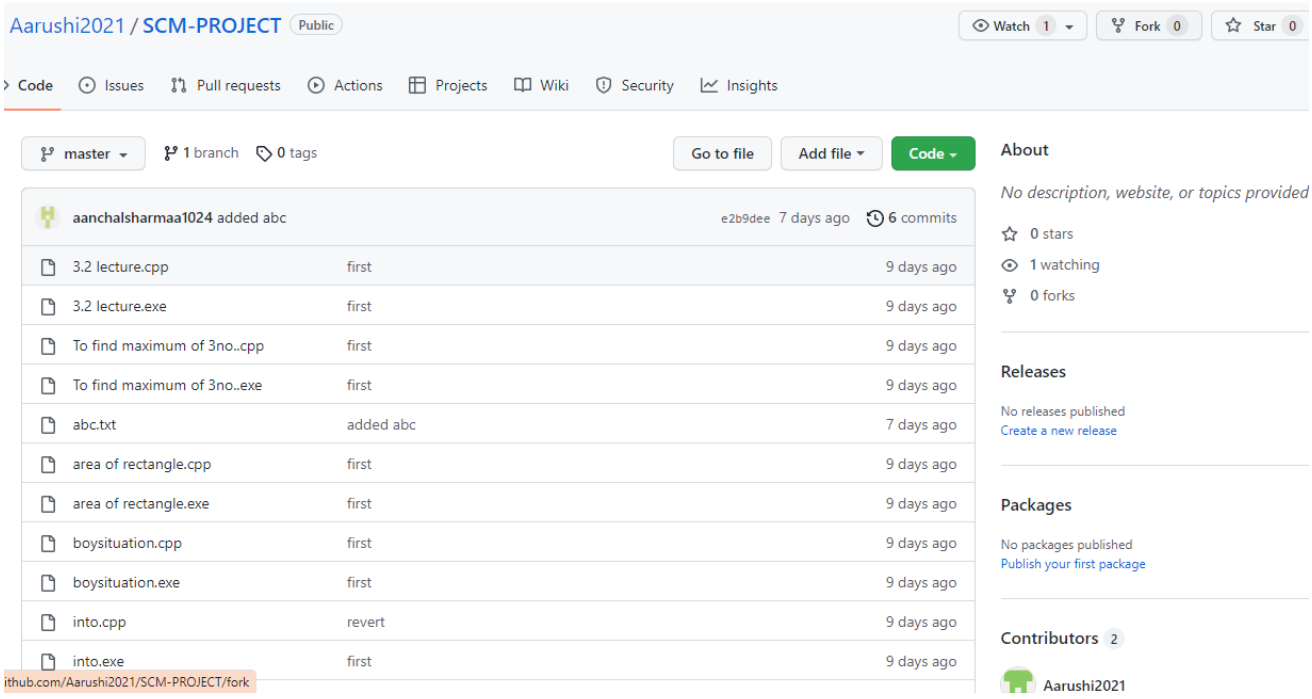
3.On the new page, click the "Collaborators" menu item on the left side of the page.



FORK AND COMMIT

A fork is a copy of a repository. Forking a repository allows you to freely experiment with changes without affecting the original project. Most commonly, forks are used to either propose changes to someone else's project to which you do not have write access, or to use someone else's project as a starting point for your own idea.

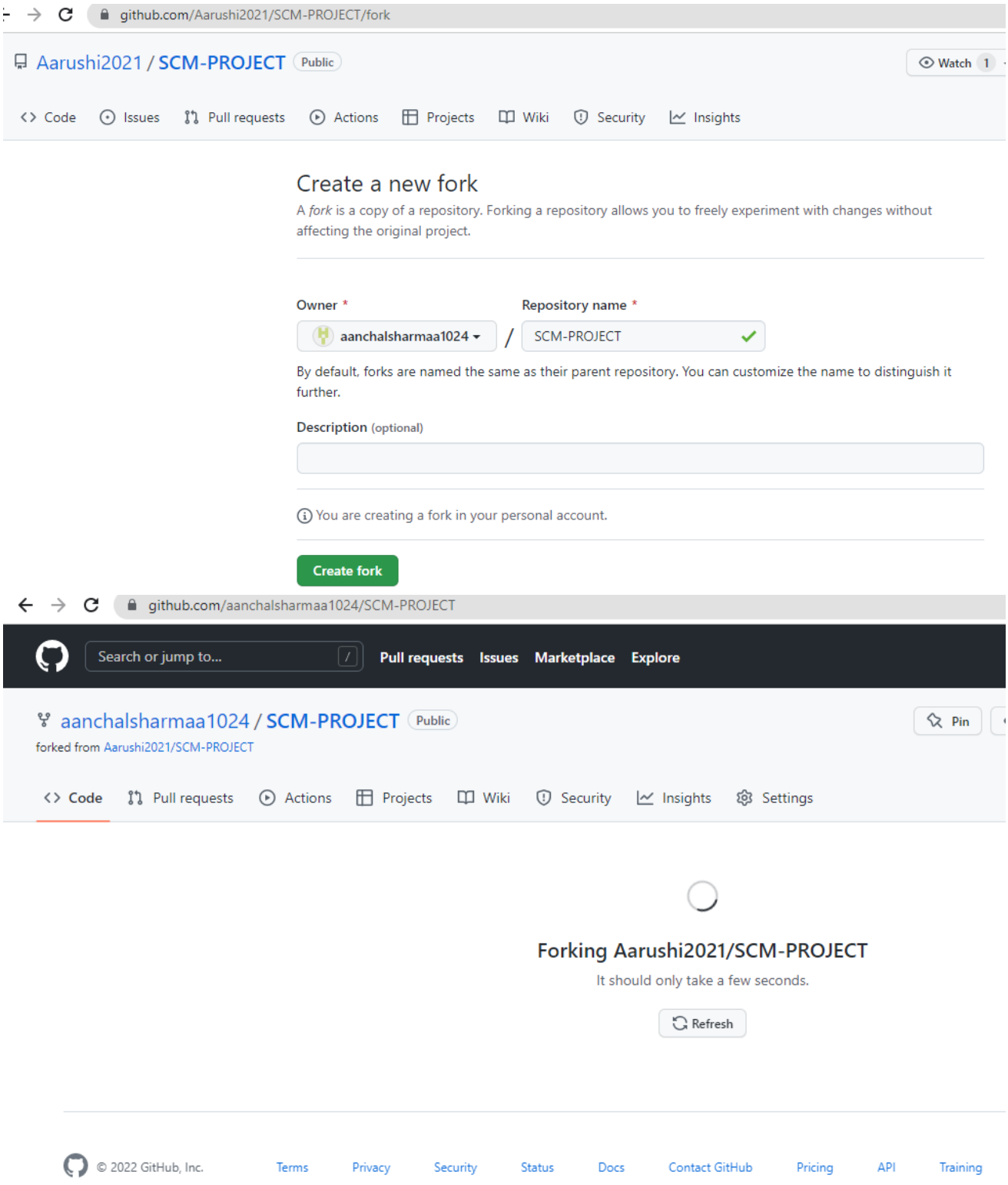
STEPS TO FORK A REPO-



1. Go to the repository that you wish to fork.

2. Click on the option ‘Fork’ in the top right corner.

3. You now have a forked repository.

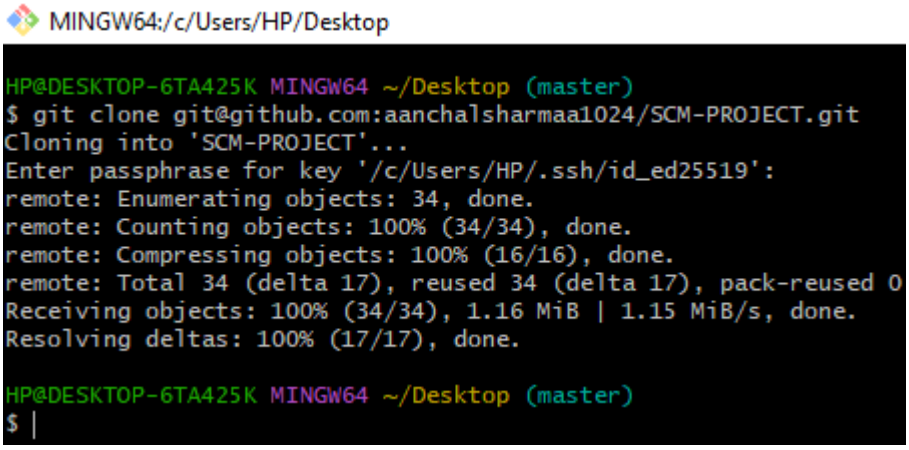

github.com/Aarushi2021/SCM-PROJECT/fork
Aarushi2021 / SCM-PROJECT
Public
Watch 1
Code
Issues
Pull requests
Actions
Projects
Wiki
Security
Insights
Create a new fork
A fork is a copy of a repository. Forking a repository allows you to freely experiment with changes without affecting the original project.
Owner *
Repository name *
aanchalsharmaa1024
SCM-PROJECT
By default, forks are named the same as their parent repository. You can customize the name to distinguish it further.
Description (optional)
You are creating a fork in your personal account.
Create fork
github.com/aanchalsharmaa1024/SCM-PROJECT
Search or jump to...
Pull requests
Issues
Marketplace
Explore
aanchalsharmaa1024 / SCM-PROJECT
Public
forked from Aarushi2021/SCM-PROJECT
Pin
Code
Pull requests
Actions
Projects
Wiki
Security
Insights
Settings
Forking Aarushi2021/SCM-PROJECT
It should only take a few seconds.
Refresh
© 2022 GitHub, Inc.
Terms
Privacy
Security
Status
Docs
Contact GitHub
Pricing
API
Training

CLONING THE REPO INTO YOUR DEVICE

When you create a repository on GitHub.com, it exists as a remote repository. You can clone your repository to create a local copy on your computer and sync between the two locations.

1. Once you have forked the repository, you can clone it into your computer using directly the option given on github or through running git clone command in git bash.
2. Copy the URL of the forked repository

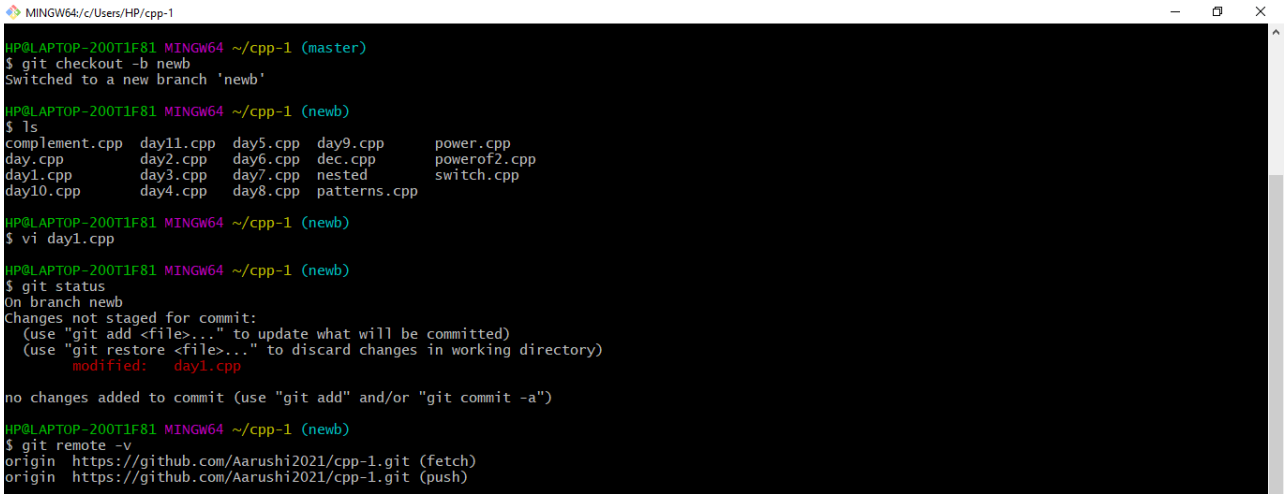
3. Open git bash and type the command " git clone <url of the forked repository>"



COMMITING CHANGES TO THE FORKED REPOSITORY

1. Once you have cloned the repository you can introduce changes to it as per your wish.

2. After changing it you have to stage the file and then commit it.
3. After committing changes push it to your remote repository.



MERGE AND RESOLVE CONFLICTS CREATED DUE TO OWN ACTIVITY AND COLLABORATORS ACTIVITY

Merging and conflicts are a common part of the Git experience. Conflicts generally arise when two people have changed the same lines in a file, or if one developer deleted a file while another developer was modifying it. In these cases, Git cannot automatically determine what is correct. Conflicts only affect the developer conducting the merge, the rest of the team is unaware of the conflict. Git will mark the file as being conflicted and halt the merging process. It is then the developers' responsibility to resolve the conflict.

1.To understand the merging concept of branches, create a branch named “feature” in your repository.

```
HP@LAPTOP-2OOT1F81 MINGW64 ~/cpp-1 (newb)
$ git branch
  master
* newb

HP@LAPTOP-2OOT1F81 MINGW64 ~/cpp-1 (newb)
$ git branch feature

HP@LAPTOP-2OOT1F81 MINGW64 ~/cpp-1 (newb)
$ git checkout feature
Switched to branch 'feature'

HP@LAPTOP-2OOT1F81 MINGW64 ~/cpp-1 (feature)
$ git log --oneline
a036d15 (HEAD -> feature, origin/newb, newb) ADDED
77c9057 (origin/master, origin/HEAD, master) inverted half pyramid
2bae9ae power
f42bbd9 mini calculator
9667d52 switch
0173bd8 power of 2
31452d4 complement
585e953 binary to dec
78370b2 power of a no. using pow() func
e2d7db5 power of a no.
2e7755b LCM using HCF
56a8c6e to find LCM
2b507b7 reverse a no.
```

2.Here, there is a file called ‘day2.cpp’. Make changes to it, add and commit them.

```
HP@LAPTOP-2OOT1F81 MINGW64 ~/cpp-1 (feature)
$ vi day2.cpp

HP@LAPTOP-2OOT1F81 MINGW64 ~/cpp-1 (feature)
$ git add day2.cpp

HP@LAPTOP-2OOT1F81 MINGW64 ~/cpp-1 (feature)
$ git commit -m "addition of two numbers"
[feature 596bc10] addition of two numbers
 1 file changed, 12 insertions(+), 1 deletion(-)

HP@LAPTOP-2OOT1F81 MINGW64 ~/cpp-1 (feature)
$ git checkout master
Switched to branch 'master'
Your branch is up to date with 'origin/master'.

HP@LAPTOP-2OOT1F81 MINGW64 ~/cpp-1 (master)
$ ls
complement.cpp  day1.cpp   day11.cpp  day3.cpp  day5.cpp  day7.cpp  day9.cpp  nested        power.cpp     switch.cpp
day.cpp         day10.cpp  day2.cpp   day4.cpp  day6.cpp  day8.cpp  dec.cpp   patterns.cpp  powerof2.cpp

HP@LAPTOP-2OOT1F81 MINGW64 ~/cpp-1 (master)
$ vi day2.cpp

HP@LAPTOP-2OOT1F81 MINGW64 ~/cpp-1 (master)
$
```

3.Similarly, change the same lines of day2.cpp file in the master branch.

4.If you are not already on the branch that you want the other one to merged in (in this example master branch), then switch to it.

5.Using the command try merging feature branch into master branch using the “git merge <branch name>”

```
HP@LAPTOP-2OOT1F81 MINGW64 ~/cpp-1 (master)
$ git add day2.cpp

HP@LAPTOP-2OOT1F81 MINGW64 ~/cpp-1 (master)
$ git commit -m "addition of two numbers"
[master e9be400] addition of two numbers
 1 file changed, 13 insertions(+), 1 deletion(-)

HP@LAPTOP-2OOT1F81 MINGW64 ~/cpp-1 (master)
$ git merge feature
Auto-merging day2.cpp
CONFLICT (content): Merge conflict in day2.cpp
Automatic merge failed; fix conflicts and then commit the result.

HP@LAPTOP-2OOT1F81 MINGW64 ~/cpp-1 (master|MERGING)
$
```

```
}
//break and continue statement
<<<<<<< HEAD
//sum of two numbers
//a+b
//sum=a+b
=======
# include<iostream>
using namespace std;
int main(){
int a,b;
cin>>a>>b;
int sum;
sum=a+b;
cout<<"sum"<<sum;
cout<<"hello world";
return 0;
>>>>>>> feature
```

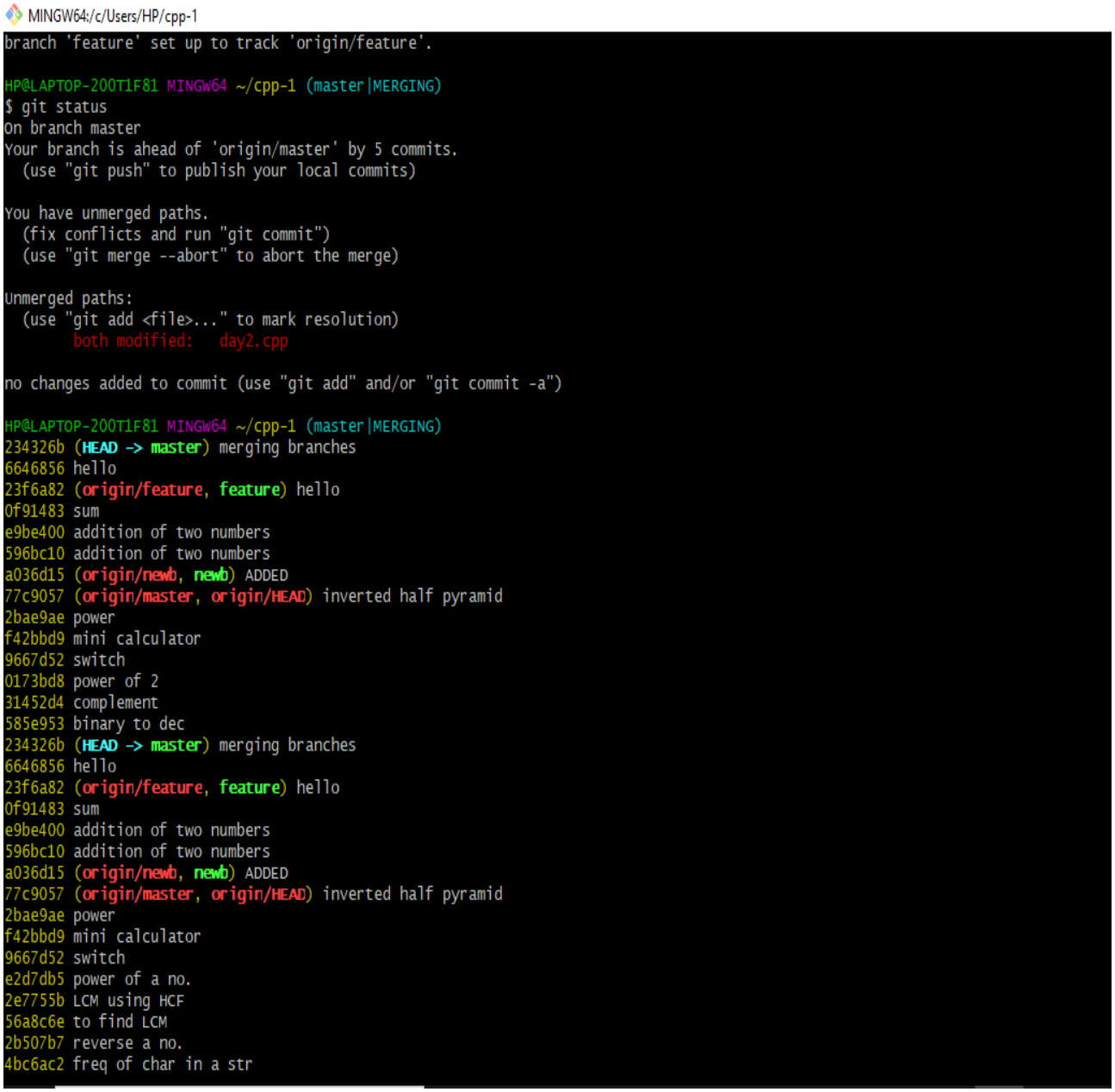
```
day2.cpp [dos] (20:47 22/05/2022)
-- INSERT --
```

```
        cout<<n<<endl;
        cin>>n;
    }while(n>0);
    return 0;
}
//break and continue statement
<<<<<<< HEAD
//sum of two numbers
//a+b
//sum=a+b
=======
# include<iostream>
using namespace std;
int main(){
int a,b;
cin>>a>>b;
int sum;
sum=a+b;
cout<<"sum"<<sum;
cout<<"hello world";
return 0;
>>>>>>> feature
```

```
day2.cpp [dos] (20:48 22/05/2022)                                                    59
"day2.cpp" [dos] 59L, 911B
```

6.Auto merging fails and conflict arises. In order to resolve it we make use of the mergetool by running the command “git mergetool”. The mergetool editor will open.

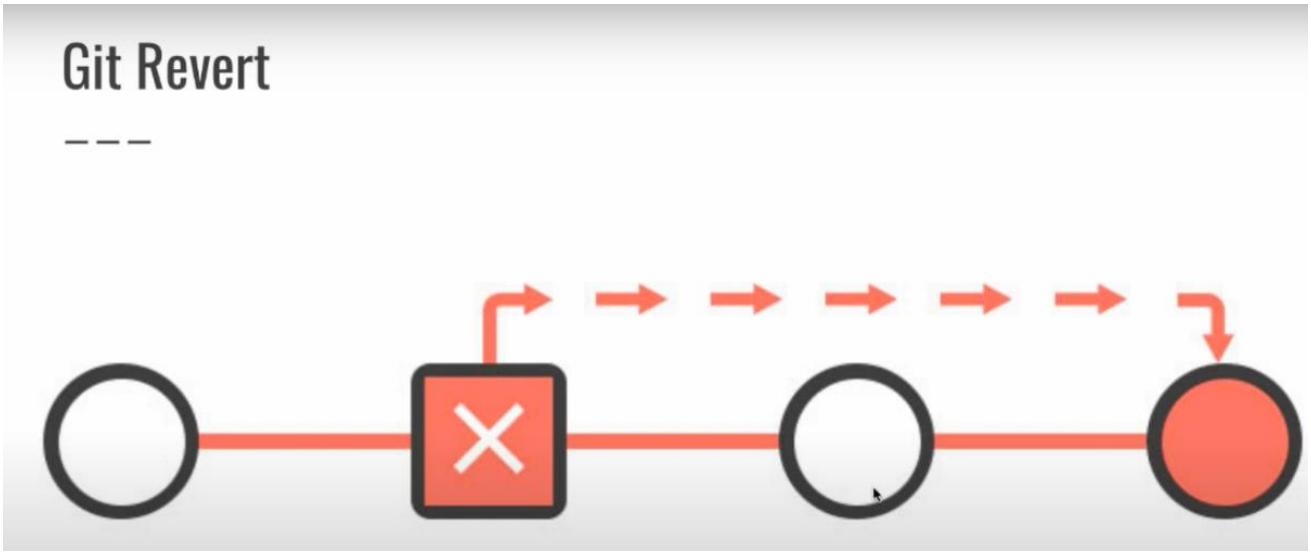
7.Make changes as per requirement in order to resolve the conflicts and exit the editor.



RESET AND REVERT

While Working with Git in certain situations we want to undo changes in the working area or index area, sometimes remove commits locally or remotely and we need to reverse those changes. We can do it by using the git reset, git revert, git checkout commands.

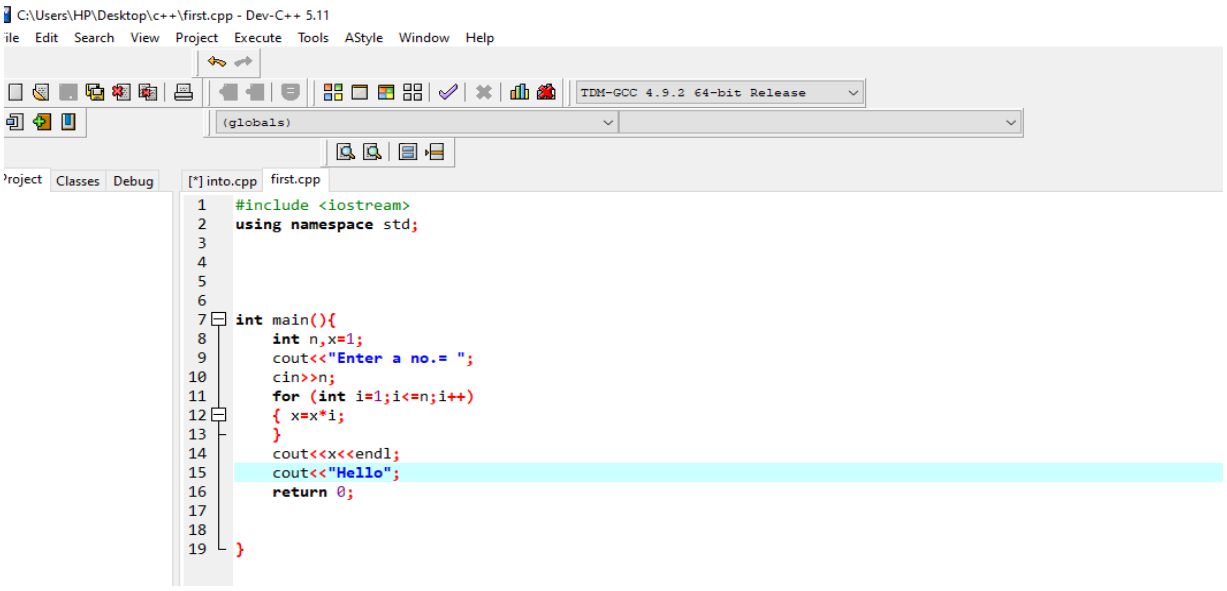
REVERT-
git revert is used to remove the commits from the remote repository. git revert removes the commit that we have done but adds one more commit which tells us that the revert has been done.



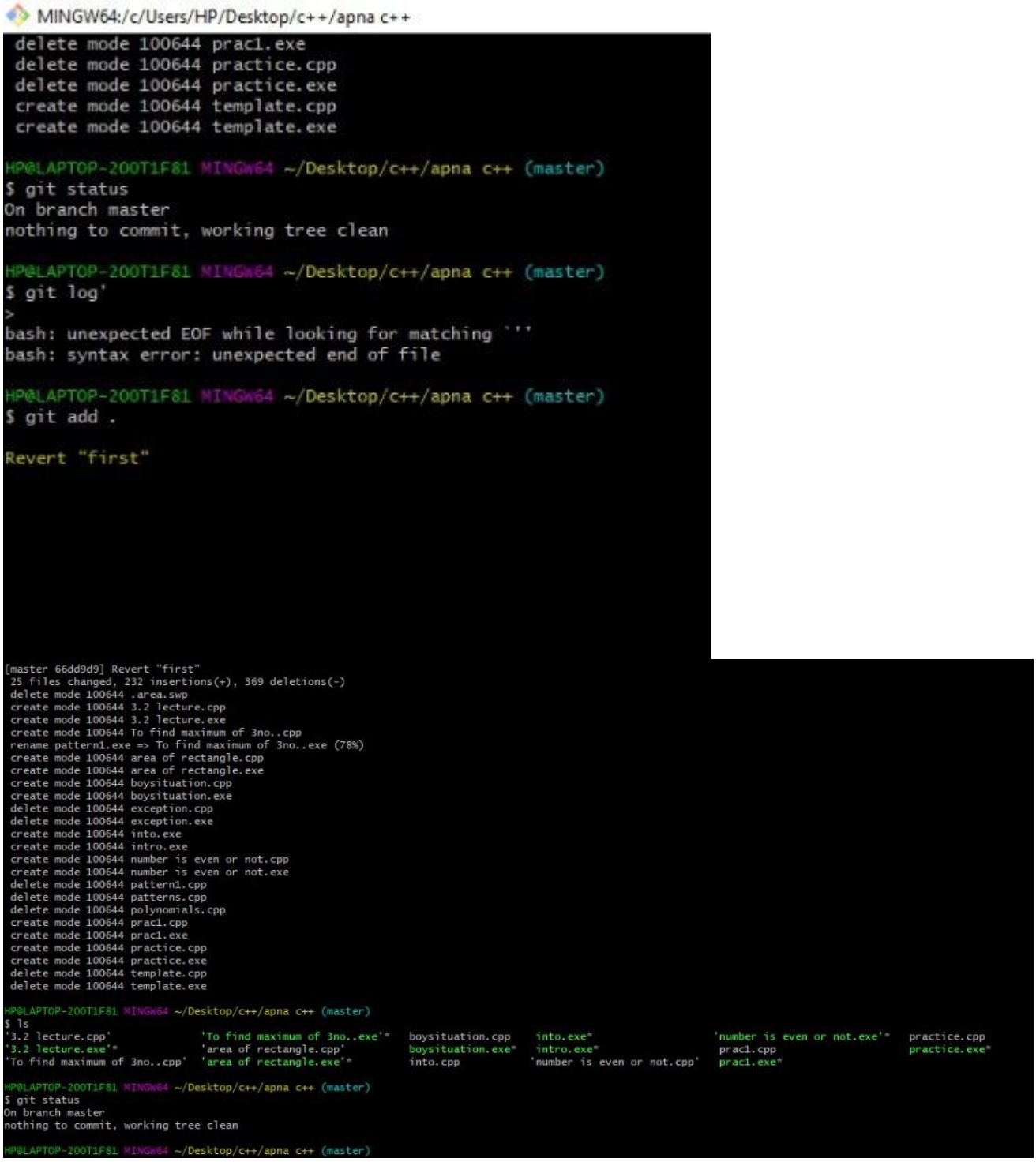
Let's see how to revert a commit, say haved pushed a unwanted commit from your local and now we will revert it.

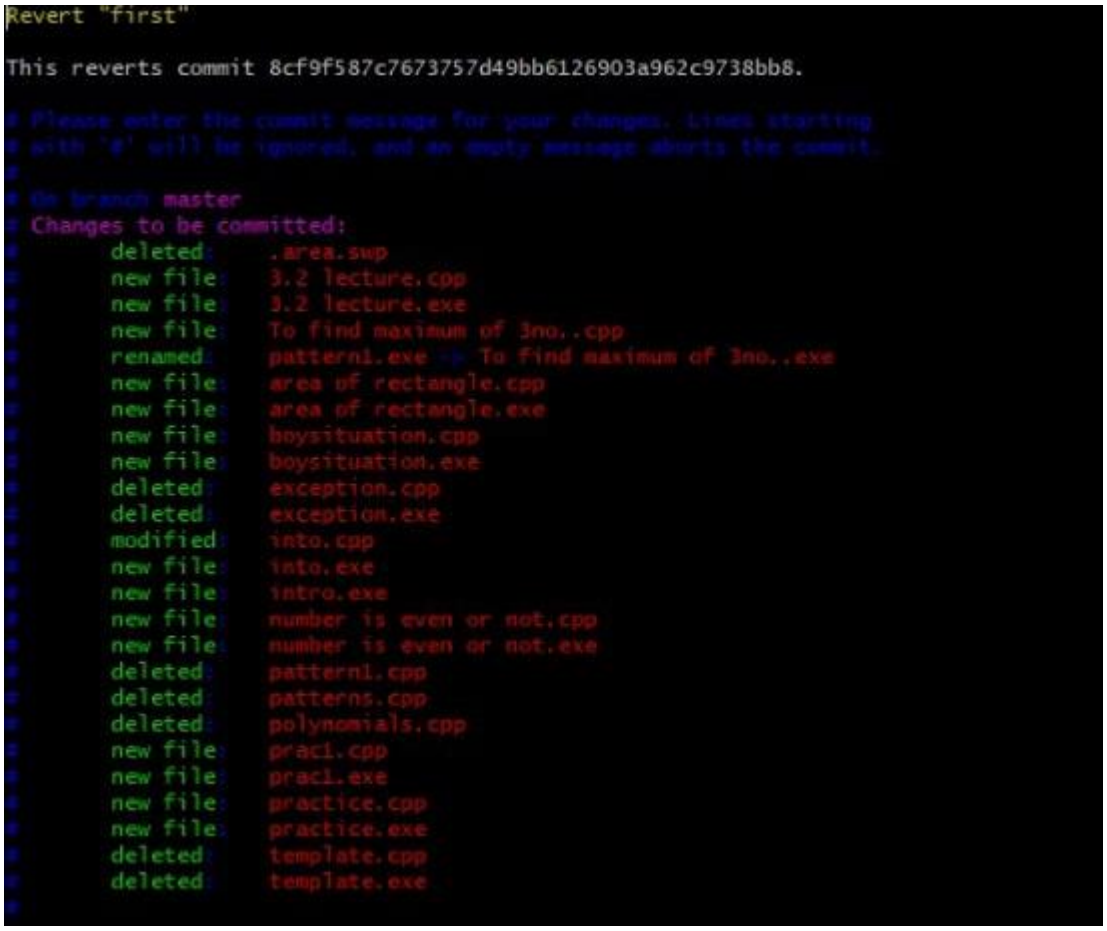
The basic advantage of reverting a commit is that it is not permanentatly deleted.

```
    second

commit bca55151dce372b6032ba0638678c2449d35b401
Author: Aarushi2021 <aarushi0015.be21@chitkara.edu.in>
Date:   Sun May 22 22:52:04 2022 +0530

    first

HP@LAPTOP-2OOT1F81 MINGW64 ~/Desktop/c++ (master)
$ git status first.cpp
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   first.cpp

no changes added to commit (use "git add" and/or "git commit -a")

HP@LAPTOP-2OOT1F81 MINGW64 ~/Desktop/c++ (master)
$ git add .

HP@LAPTOP-2OOT1F81 MINGW64 ~/Desktop/c++ (master)
$ git commit -m "1"
[master 5918be4] 1
 1 file changed, 4 insertions(+), 2 deletions(-)

HP@LAPTOP-2OOT1F81 MINGW64 ~/Desktop/c++ (master)
$ git log
commit 5918be4445dc38b5b1d452e4a3c9f1797a56a652 (HEAD -> master)
Author: Aarushi2021 <aarushi0015.be21@chitkara.edu.in>
Date:   Sun May 22 23:37:45 2022 +0530

    1

commit 6bc79a1240a7eadf5650f2d24e6223c5f9bf0e7a
Author: Aarushi2021 <aarushi0015.be21@chitkara.edu.in>
Date:   Sun May 22 22:57:38 2022 +0530

    second

commit bca55151dce372b6032ba0638678c2449d35b401
Author: Aarushi2021 <aarushi0015.be21@chitkara.edu.in>
Date:   Sun May 22 22:52:04 2022 +0530

    first

HP@LAPTOP-2OOT1F81 MINGW64 ~/Desktop/c++ (master)
$
```



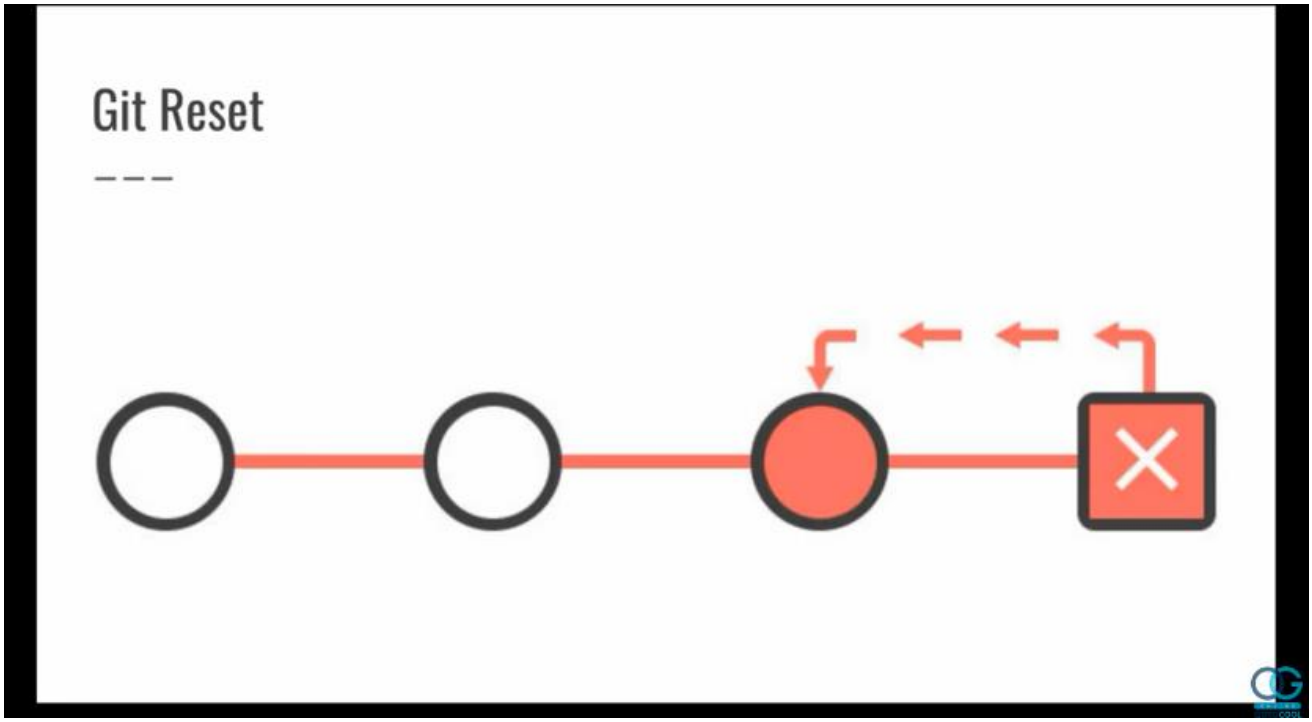
Now to revert the changes made in the commit run the “git revert <commit id>” command.



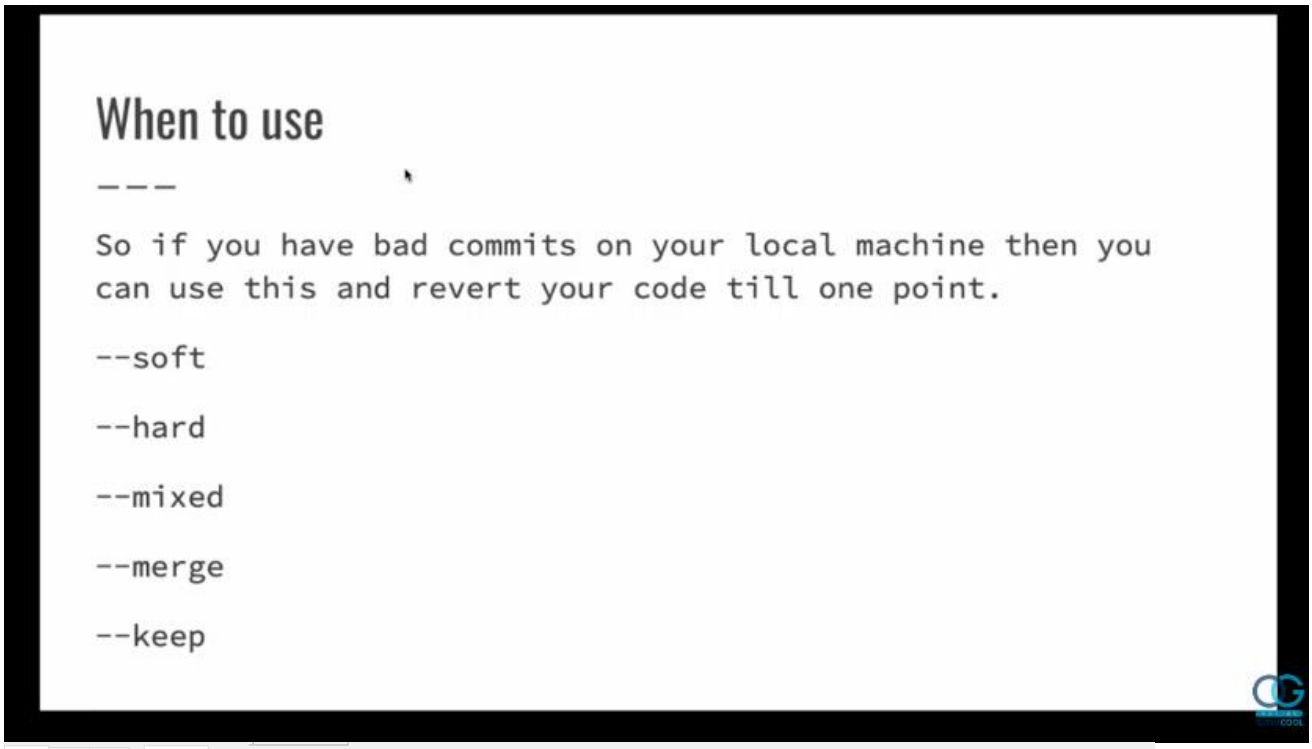

Revert "first"
This reverts commit 8cf9f587c7673757d49bb6126903a962c9738bb8.
Please enter the commit message for your changes. Lines starting
with '#' will be ignored, and an empty message aborts the commit.
#
On branch master
Changes to be committed:
#	deleted:	.area.swp
#	new file:	3.2 lecture.cpp
#	new file:	3.2 lecture.exe
#	new file:	To find maximum of 3no..cpp
#	renamed:	pattern1.exe -> To find maximum of 3no..exe
#	new file:	area of rectangle.cpp
#	new file:	area of rectangle.exe
#	new file:	boysituation.cpp
#	new file:	boysituation.exe
#	deleted:	exception.cpp
#	deleted:	exception.exe
#	modified:	into.cpp
#	new file:	into.exe
#	new file:	intro.exe
#	new file:	number is even or not.cpp
#	new file:	number is even or not.exe
#	deleted:	pattern1.cpp
#	deleted:	patterns.cpp
#	deleted:	polynomials.cpp
#	new file:	prac1.cpp
#	new file:	prac1.exe
#	new file:	practice.cpp
#	new file:	practice.exe
#	deleted:	template.cpp
#	deleted:	template.exe
#

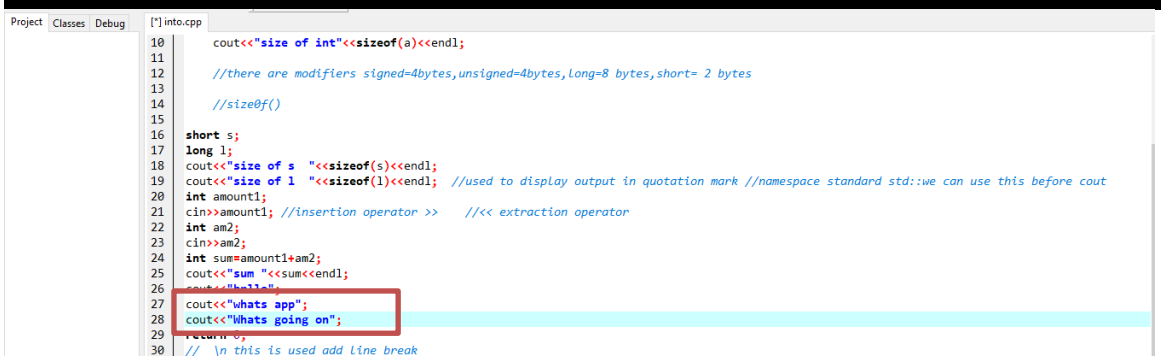
RESET-

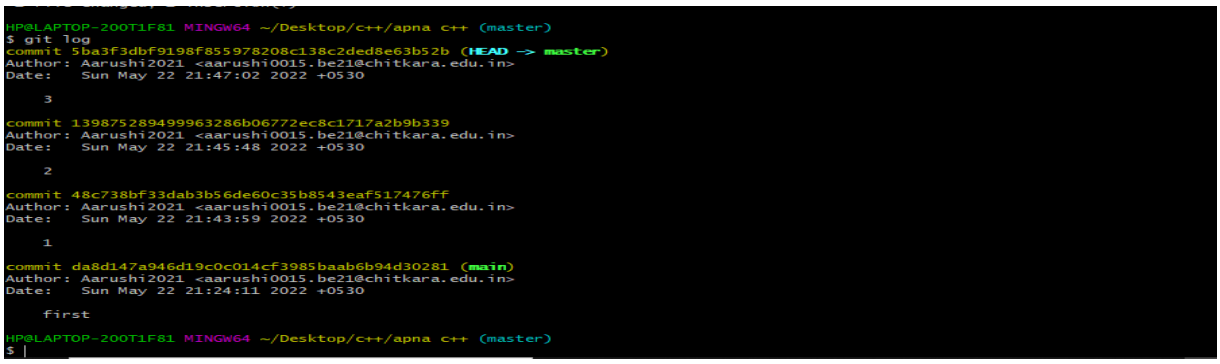
git reset is used when we want to unstage a file and bring our changes back to the working directory. Git reset can also be used to remove commits from the local repository.



Suppose we make edits to a file, stage it and commit it







In order to reset the changes made in the recent commit, run the "git reset --hard HEAD~1" command.
Or a command git "reset commit no."

```
HP@LAPTOP-2OOT1F81 MINGW64 ~/Desktop/c++/apna c++ (master)
$ git reset  48c738bf33dab3b56de60c35b8543eaf517476ff

Unstaged changes after reset:
M       into.cpp

HP@LAPTOP-2OOT1F81 MINGW64 ~/Desktop/c++/apna c++ (master)
$ git log
commit 48c738bf33dab3b56de60c35b8543eaf517476ff (HEAD -> master)
Author: Aarushi2021 <aarushi0015.be21@chitkara.edu.in>
Date:   Sun May 22 21:43:59 2022 +0530

    1

commit da8d147a946d19c0c014cf3985baab6b94d30281 (main)
Author: Aarushi2021 <aarushi0015.be21@chitkara.edu.in>
Date:   Sun May 22 21:24:11 2022 +0530

    first

HP@LAPTOP-2OOT1F81 MINGW64 ~/Desktop/c++/apna c++ (master)
$
```

The HEAD returns to the previous commit and the changes made are reset.

```
14      //sizeOf()
15
16  short s;
17  long l;
18  cout<<"size of s  "<<sizeof(s)<<endl;
19  cout<<"size of l  "<<sizeof(l)<<endl;  //used to display output in quotation mark //namespace standard std::we can use this before cout
20  int amount1;
21  cin>>amount1; //insertion operator >>    //<< extraction operator
22  int am2;
23  cin>>am2;
24  int sum=amount1+am2;
25  cout<<"sum "<<sum<<endl;
26  cout<<"hrllo";
27
28
29  return 0;
```

TASK 2

S. No.	Title
1	Introduction
2	Create a distributed repository and add members in project team
3	Open and Close Pull request
4	Create a pull request on a team member’s repo and close pull requests generated by team members on own repository as a maintainer
5	Network graphs

INTRODUCTION:

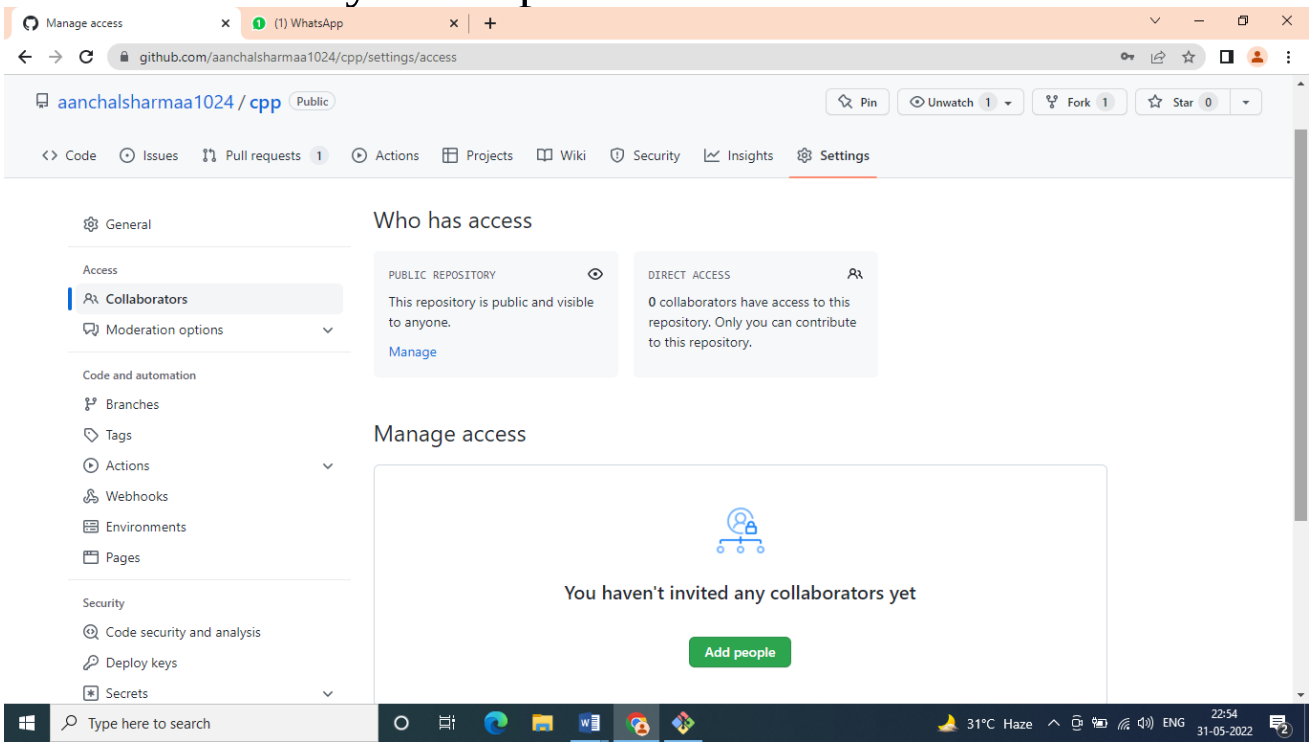
This task is performed in the group of four. Each one of us made it possible to work on this project as if we are doing an open source contribution.

Each one of us create his/her repo and rest of the three contributers in the repo , firstly forked that repo and then clone it in our local machine and then make a new branch and made some changes in the existing file in master branch in the repo and then push it from your local system.

And finally make pull request to the owner of the repo in whose repo we want to make the changes.

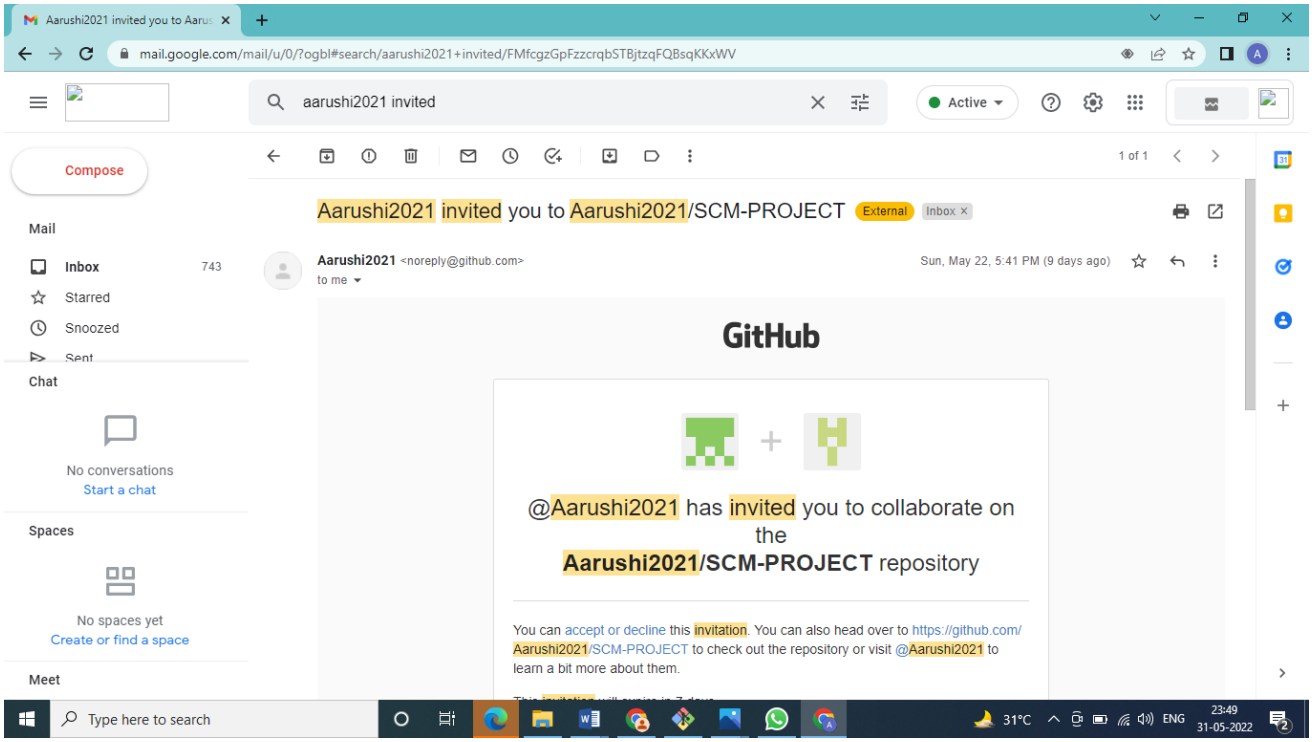
CREATE A DISTRIBUTED REPOSITORY AND ADD MEMBERS IN PROJECT TEAM

1. On the homepage of your GitHub account, click on Repositories option in the menu bar.
2. Click on the ‘New’ button in the top right corner.
3. Enter the Repository name and add the description of the repository.
4. To add members to your repository, open your repository and select settings option in the navigation bar.
5. Click on Collaborators option under the access tab.
6. You can manage access and add/remove team members to your project.
7. To add members, click on the add people, option and search the id of your respective team members.



9. To accept the invitation from your team members, open your email registered with GitHub.

10. You will receive an invitation mail from the repository owner. Open the email and click on accept invitation.



Similarly , you can add more collaborators to your project.

OPEN AND CLOSE PULL REQUEST:

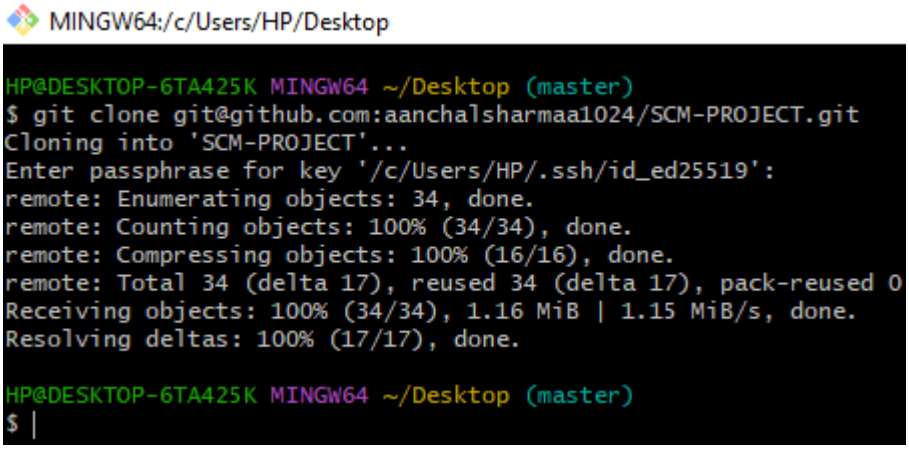
1. First, select a repository of the other person in which you want to make changes and create a pull request.

2. Clone it into your local storage.

3. To open a pull request we first have to make a new branch, by using git checkout -b branch name option.

4. After making new branch we add a file to the branch or make changes in the existing file.

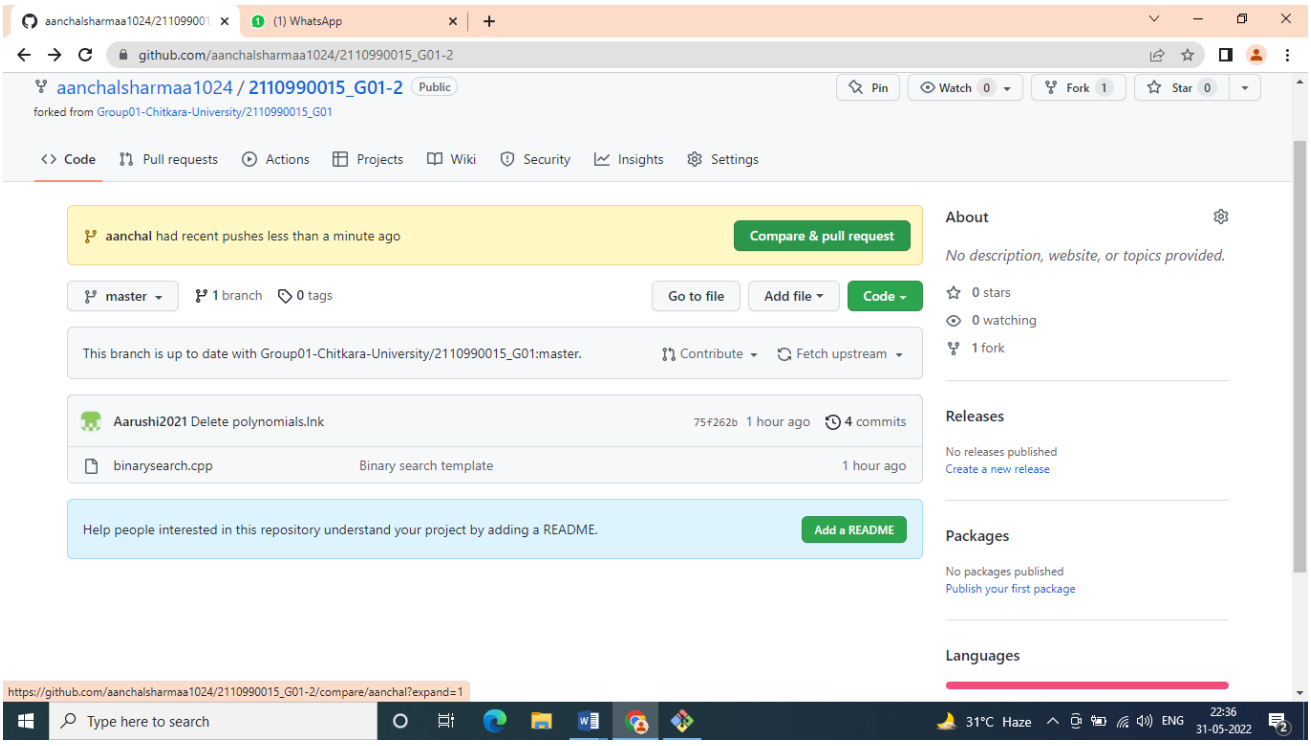
5. Add and commit the changes to the local repository.



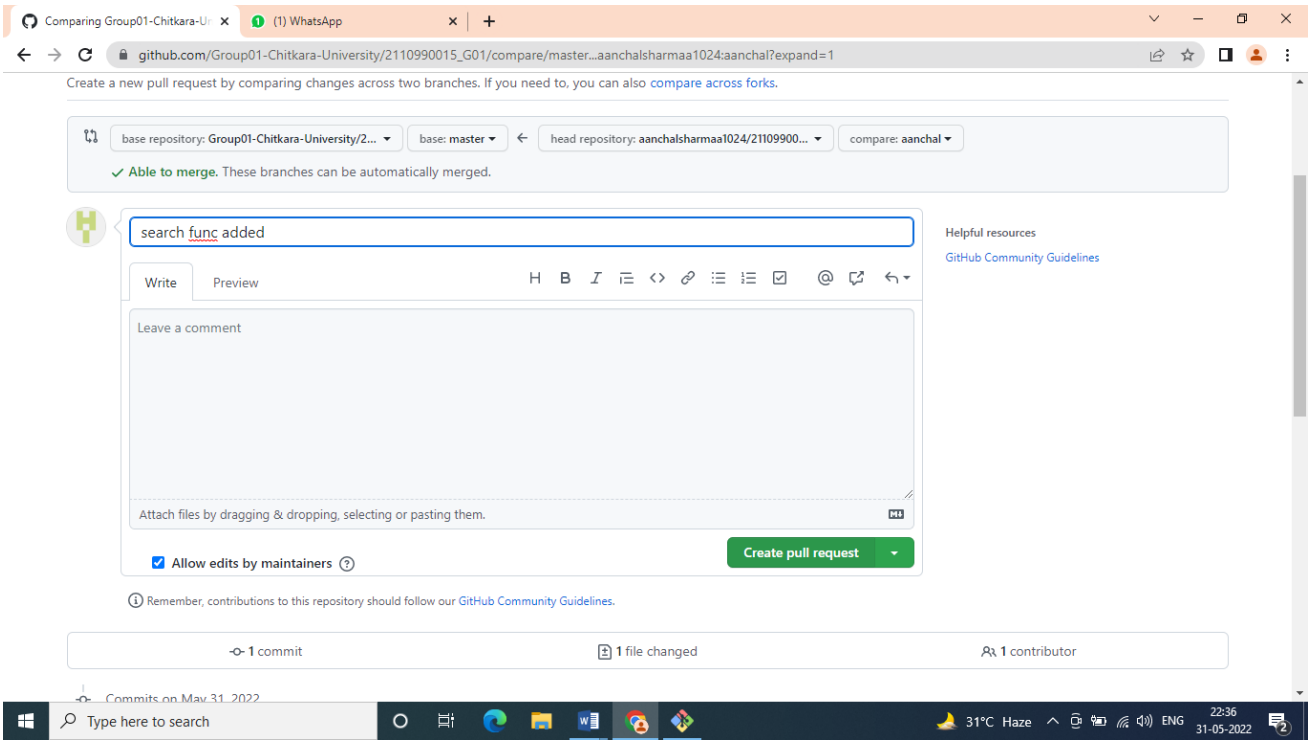
6. Use git push origin branch name option to push the new branch to the main repo

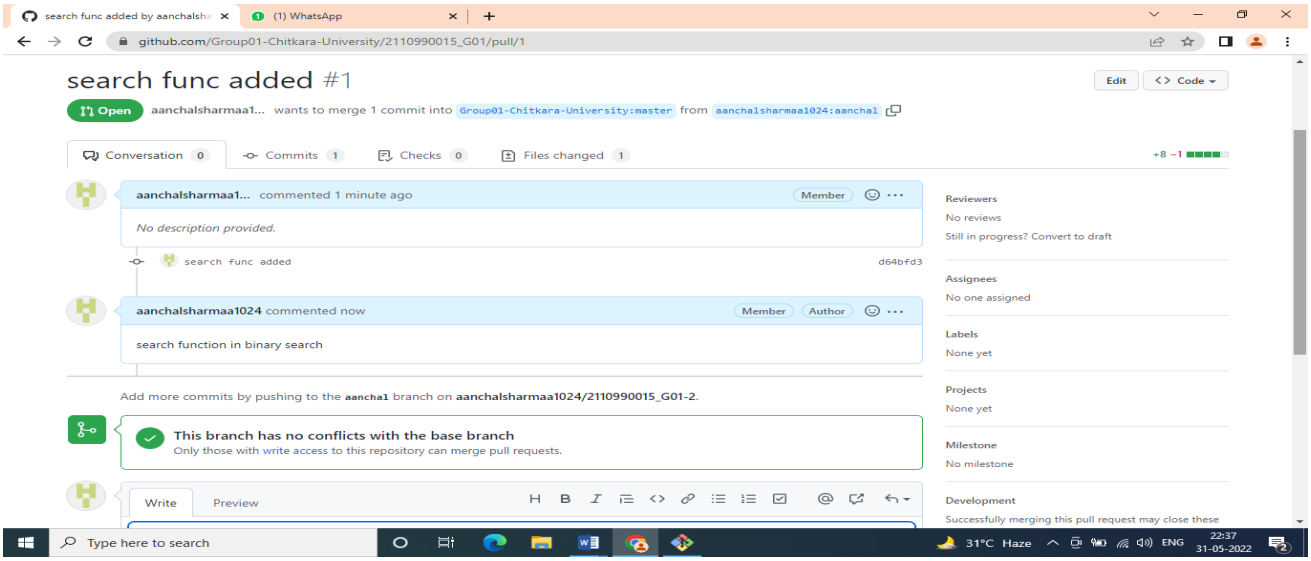
```
HP@DESKTOP-6TA425K MINGW64 ~/2110990015_G01-1 (aanchal)
$ git push -u origin aanchal
Enter passphrase for key '/c/Users/HP/.ssh/id_ed25519':
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 4 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 603 bytes | 603.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
remote:
remote: Create a pull request for 'aanchal' on GitHub by visiting:
remote:      https://github.com/aanchalsharmaa1024/2110990015_G01-1/pull/new/aanchal
remote:
To github.com:aanchalsharmaa1024/2110990015_G01-1.git
 * [new branch]      aanchal -> aanchal
branch 'aanchal' set up to track 'origin/aanchal'.
```

7. After pushing new branch GitHub will either automatically ask you to create a pull request or you can create your own pull request.



8. to create your own pull request ,click on pull request option.


Comparing Group01-Chitkara-U
(1) WhatsApp
github.com/Group01-Chitkara-University/2110990015_G01/compare/master...aanchalsharmaa1024:aanchal?expand=1
Create a new pull request by comparing changes across two branches. If you need to, you can also compare across forks.
base repository: Group01-Chitkara-University/2...
base: master
head repository: aanchalsharmaa1024/21109900...
compare: aanchal
Able to merge. These branches can be automatically merged.
search func added
Write
Preview
Leave a comment
Attach files by dragging & dropping, selecting or pasting them.
Allow edits by maintainers
Create pull request
Helpful resources
GitHub Community Guidelines
Remember, contributions to this repository should follow our GitHub Community Guidelines.
1 commit
1 file changed
1 contributor
Commits on May 31, 2022
Type here to search
31°C Haze
ENG
22:36
31-05-2022


search func added by aanchalsha
(1) WhatsApp
github.com/Group01-Chitkara-University/2110990015_G01/pull/1
search func added #1
Open
aanchalsharmaa1... wants to merge 1 commit into Group01-Chitkara-University:master from aanchalsharmaa1024:aanchal
Edit
Code
Conversation 0
Commits 1
Checks 0
Files changed 1
+8 −1
aanchalsharmaa1... commented 1 minute ago
Member
No description provided.
search func added
d64bfd3
aanchalsharmaa1024 commented now
Member
Author
search function in binary search
Add more commits by pushing to the aancha1 branch on aanchalsharmaa1024/2110990015_G01-2.
This branch has no conflicts with the base branch
Only those with write access to this repository can merge pull requests.
Write
Preview
Reviewers
No reviews
Still in progress? Convert to draft
Assignees
No one assigned
Labels
None yet
Projects
None yet
Milestone
No milestone
Development
Successfully merging this pull request may close these
Type here to search
31°C Haze
ENG
22:37
31-05-2022

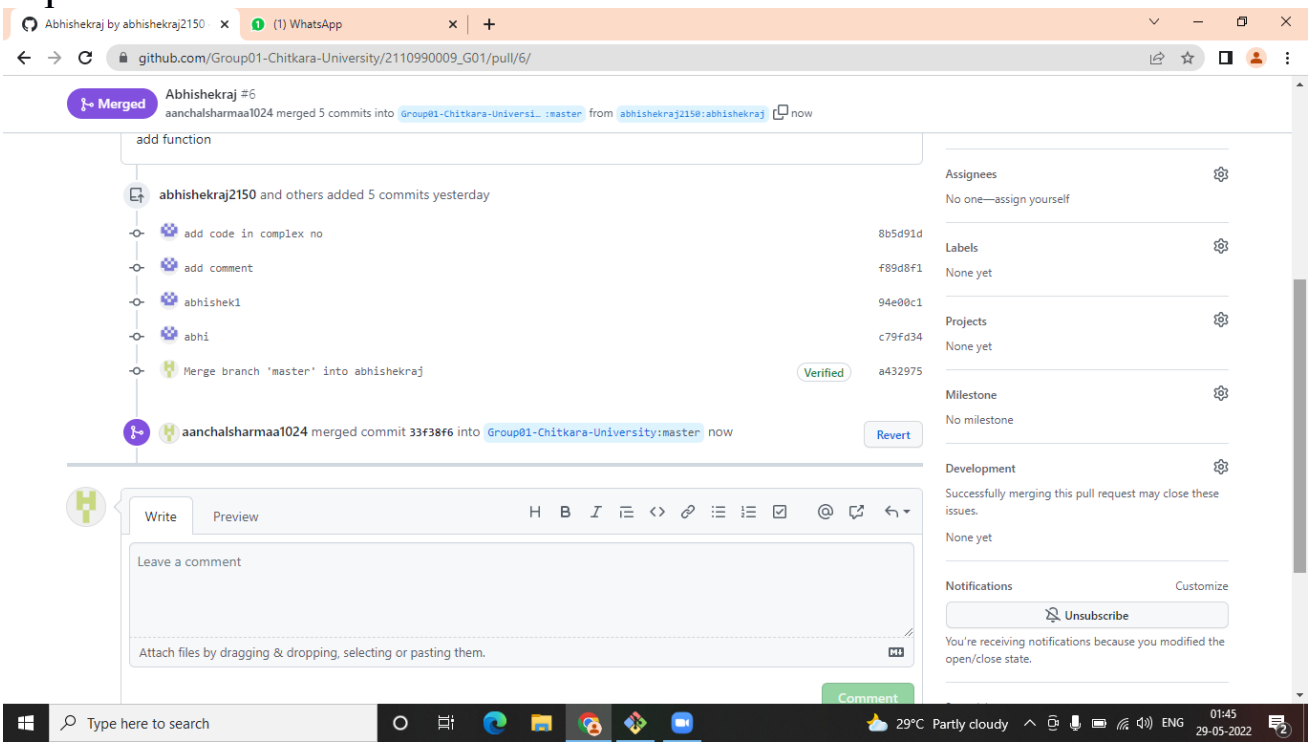
9. GitHub will detect any conflicts and ask you to enter a description of your pull request.

10.After opening a pull request the owner of the original repository will be sent the request if they want to merge or close the request.

11.If the owner chooses not to merge your pull request, they will close it.

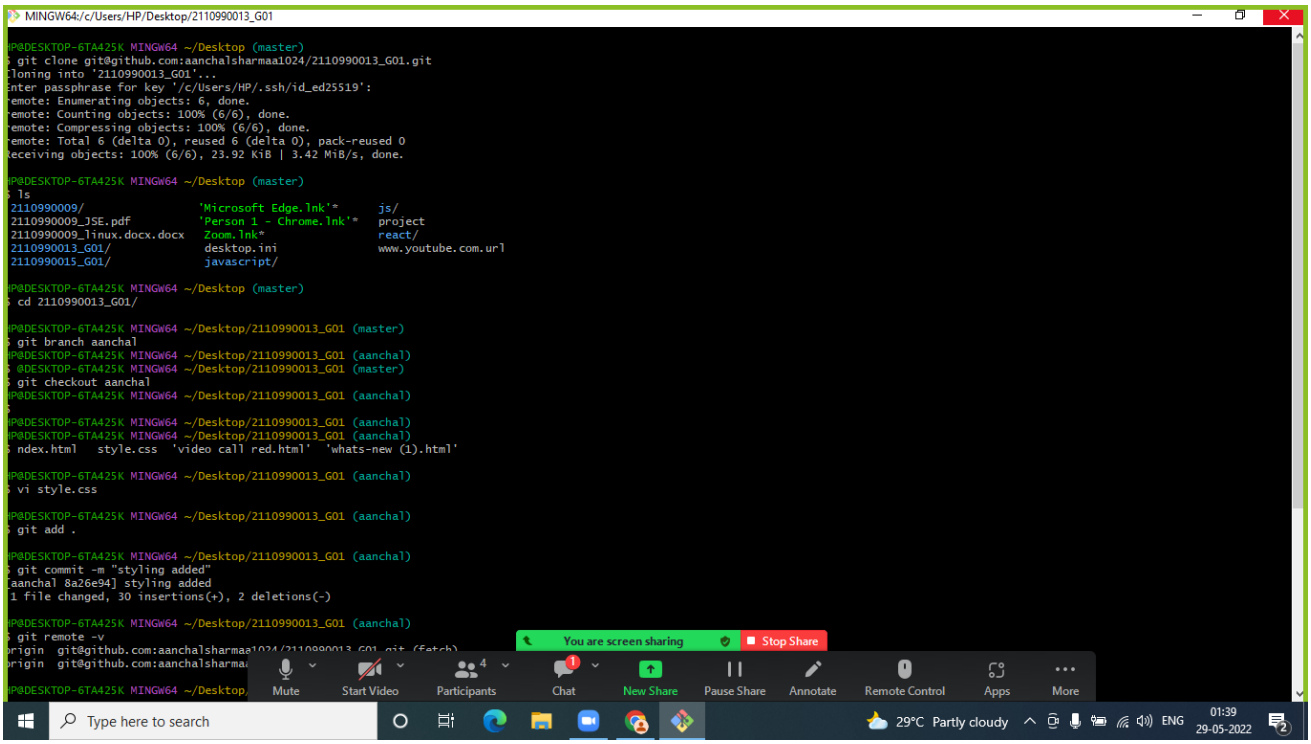
12.To close the pull request simply click on close pull request and add comment/ reason why you closed the pull request.

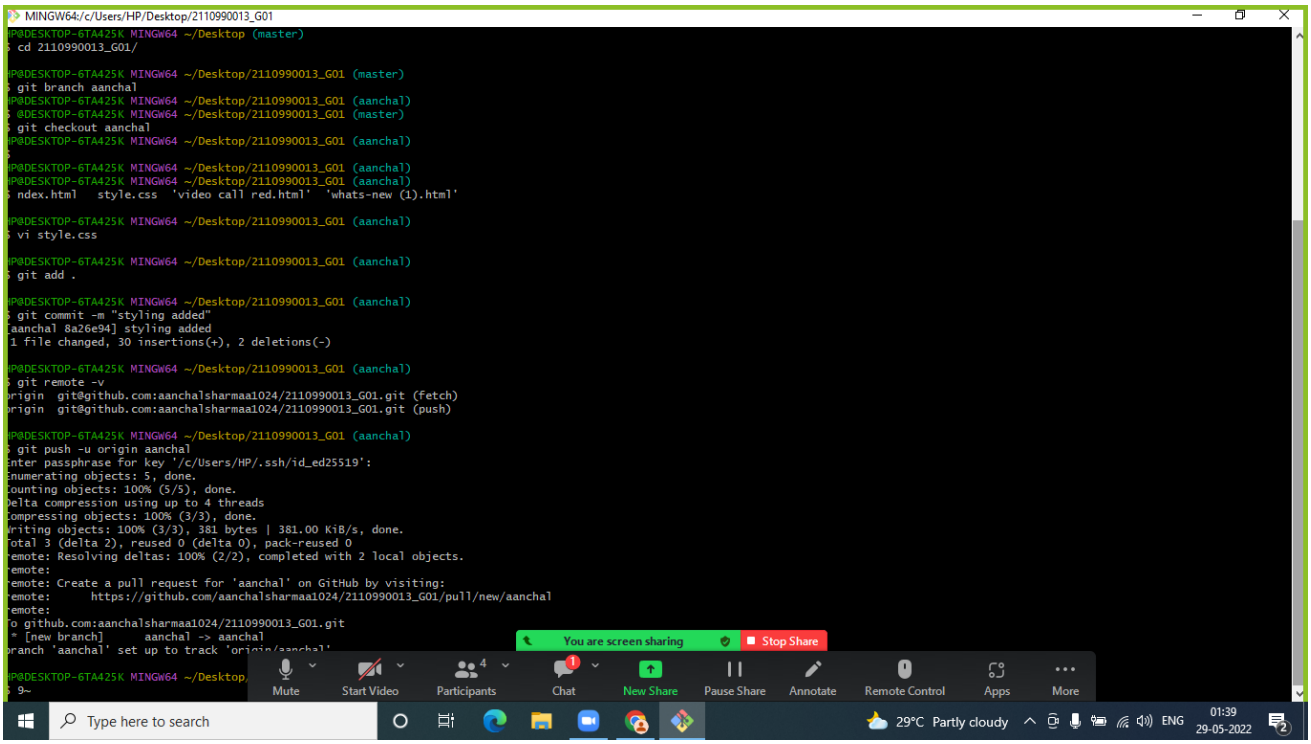
13.If you want to merge it into the original, click on merge pull request.



CREATE A PULL REQUEST ON A TEAM MEMBER'S REPO AND CLOSE PULL REQUESTS GENERATED BY TEAM MEMBERS ON OWN REPOSITORY AS A MAINTAINER

#Creating pull request on a team member's repo

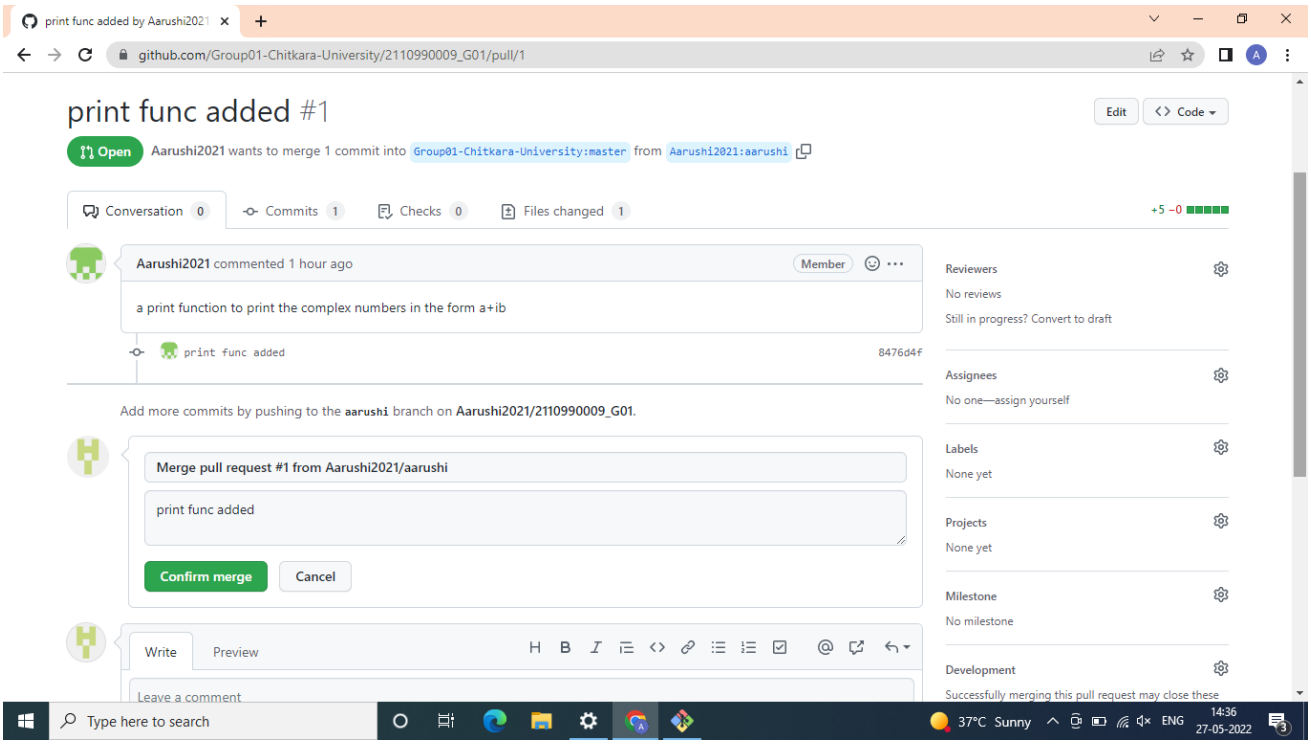


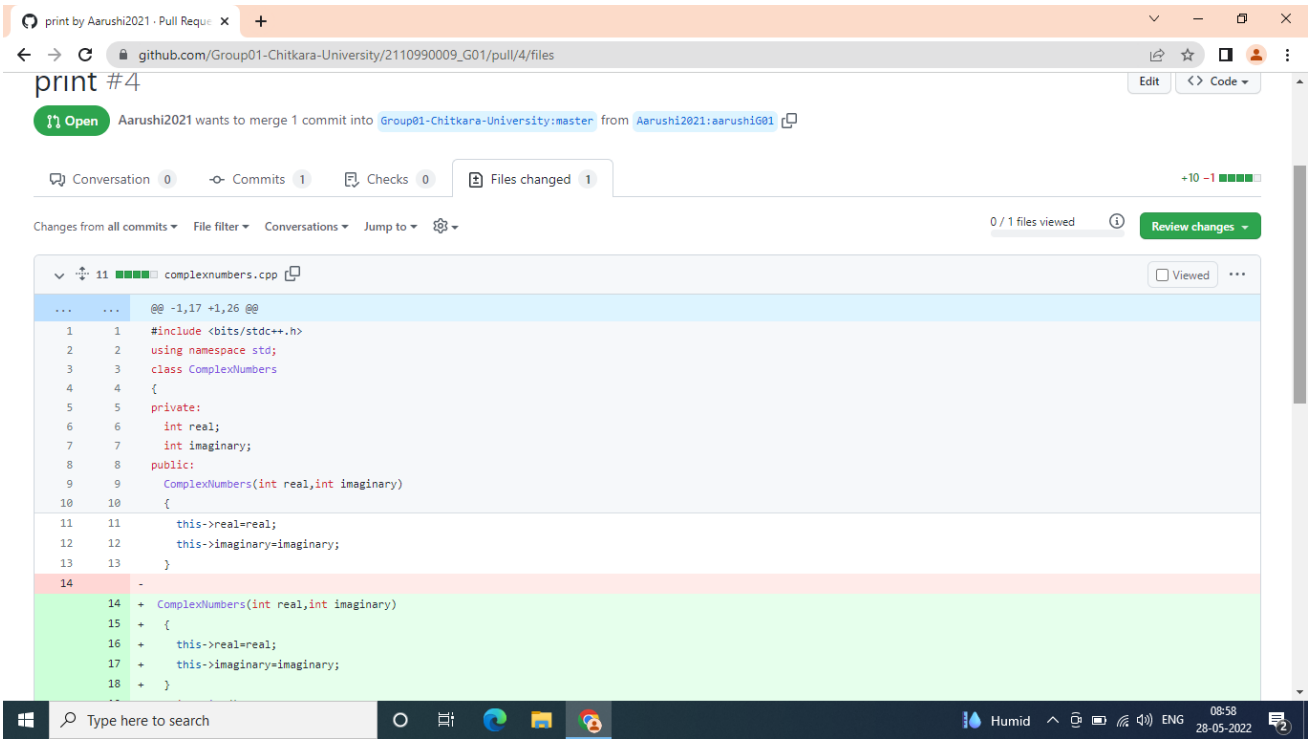


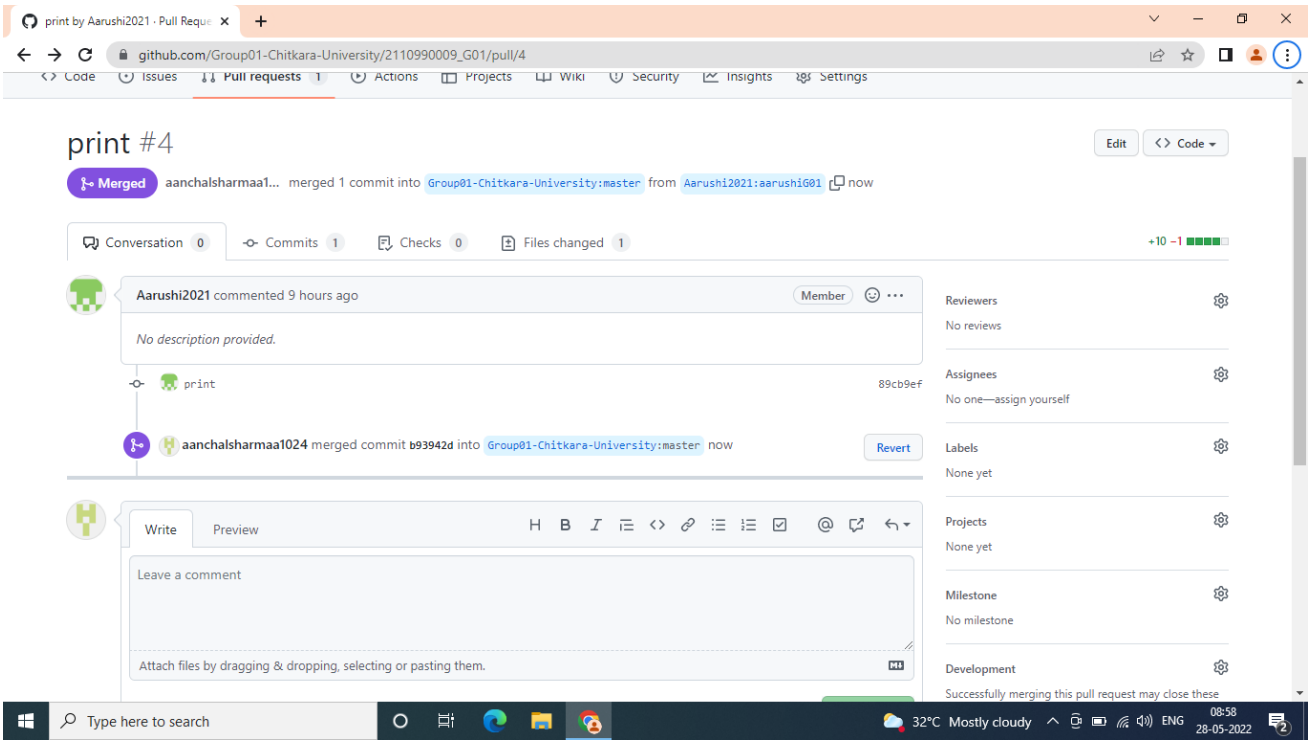
close pull request generated by team members on own repo as a maintainer.

1. Firstly open the pull request , review changes in it , and then merge it ...if there will be a conflict in merging the pull request resolve it .

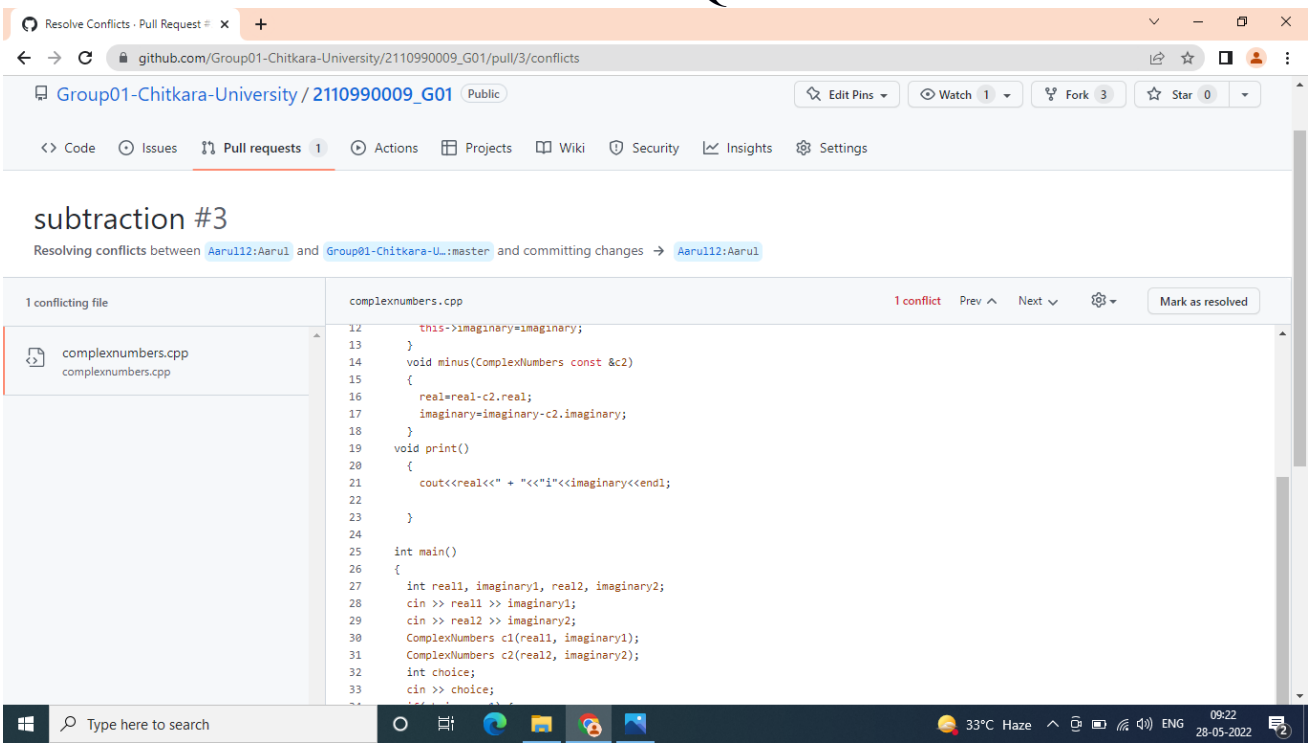
#1 TEAM MEMBER'S PULL REQUEST

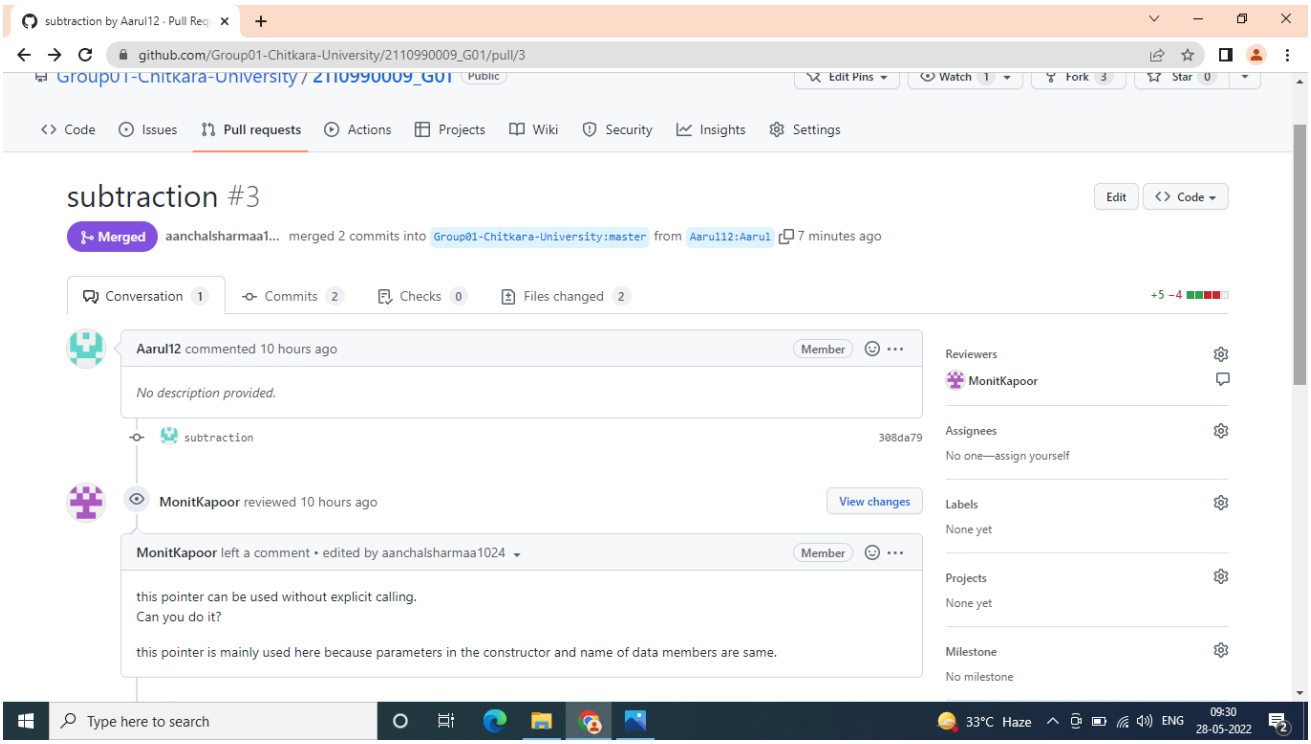

print func added by Aarushi2021
github.com/Group01-Chitkara-University/2110990009_G01/pull/1
print func added #1
Edit
Code
Open
Aarushi2021 wants to merge 1 commit into Group01-Chitkara-University:master from Aarushi2021:aarushi
Conversation 0
Commits 1
Checks 0
Files changed 1
+5 −0
Aarushi2021 commented 1 hour ago
Member
a print function to print the complex numbers in the form a+ib
print func added
8476d4f
Add more commits by pushing to the aarushi branch on Aarushi2021/2110990009_G01.
Merge pull request #1 from Aarushi2021/aarushi
print func added
Confirm merge
Cancel
Write
Preview
Leave a comment
Reviewers
No reviews
Still in progress? Convert to draft
Assignees
No one—assign yourself
Labels
None yet
Projects
None yet
Milestone
No milestone
Development
Successfully merging this pull request may close these
Type here to search
37°C Sunny
ENG
14:36
27-05-2022


print by Aarushi2021 · Pull Requ
github.com/Group01-Chitkara-University/2110990009_G01/pull/4/files
print #4
Edit
Code
Open
Aarushi2021 wants to merge 1 commit into Group01-Chitkara-University:master from Aarushi2021:aarushiG01
Conversation 0
Commits 1
Checks 0
Files changed 1
+10 −1
Changes from all commits
File filter
Conversations
Jump to
0 / 1 files viewed
Review changes
11
complexnumbers.cpp
Viewed
@@ -1,17 +1,26 @@
#include <bits/stdc++.h>
using namespace std;
class ComplexNumbers
{
private:
int real;
int imaginary;
public:
ComplexNumbers(int real,int imaginary)
{
this->real=real;
this->imaginary=imaginary;
}
-
+ ComplexNumbers(int real,int imaginary)
+ {
+ this->real=real;
+ this->imaginary=imaginary;
+ }
Type here to search
Humid
ENG
08:58
28-05-2022

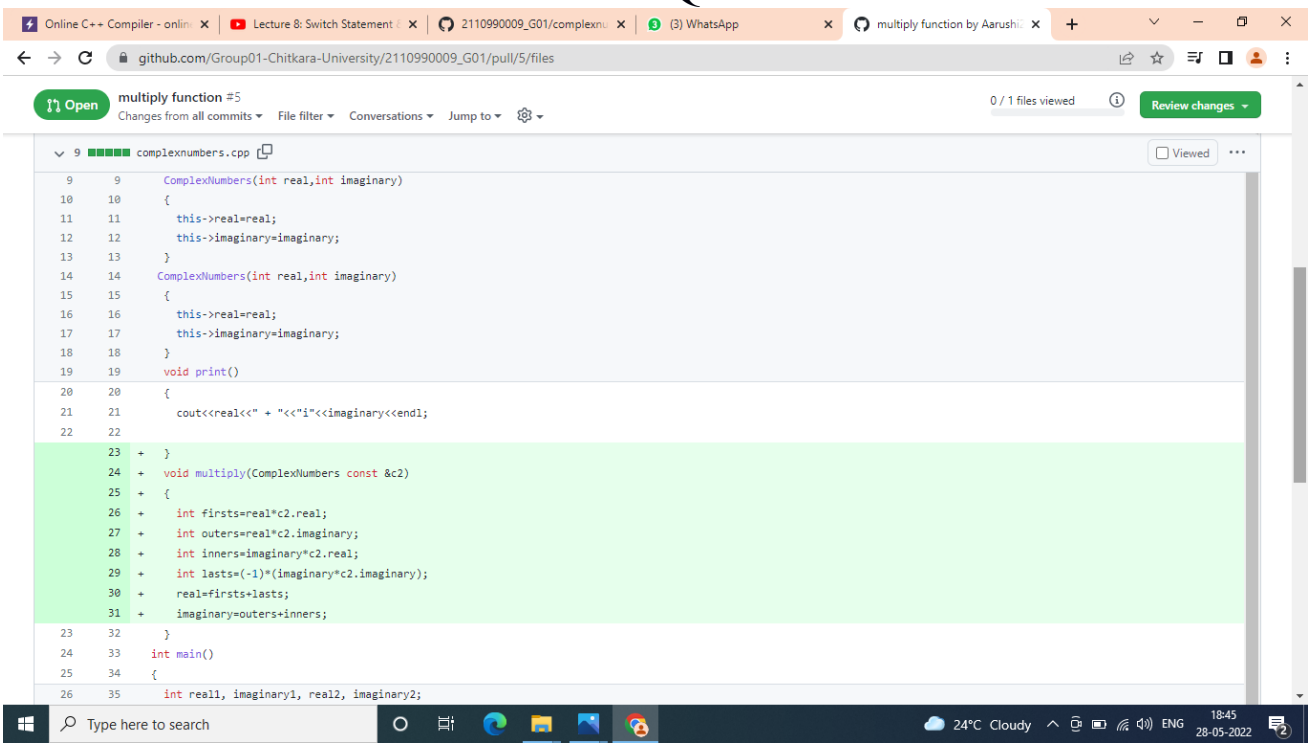


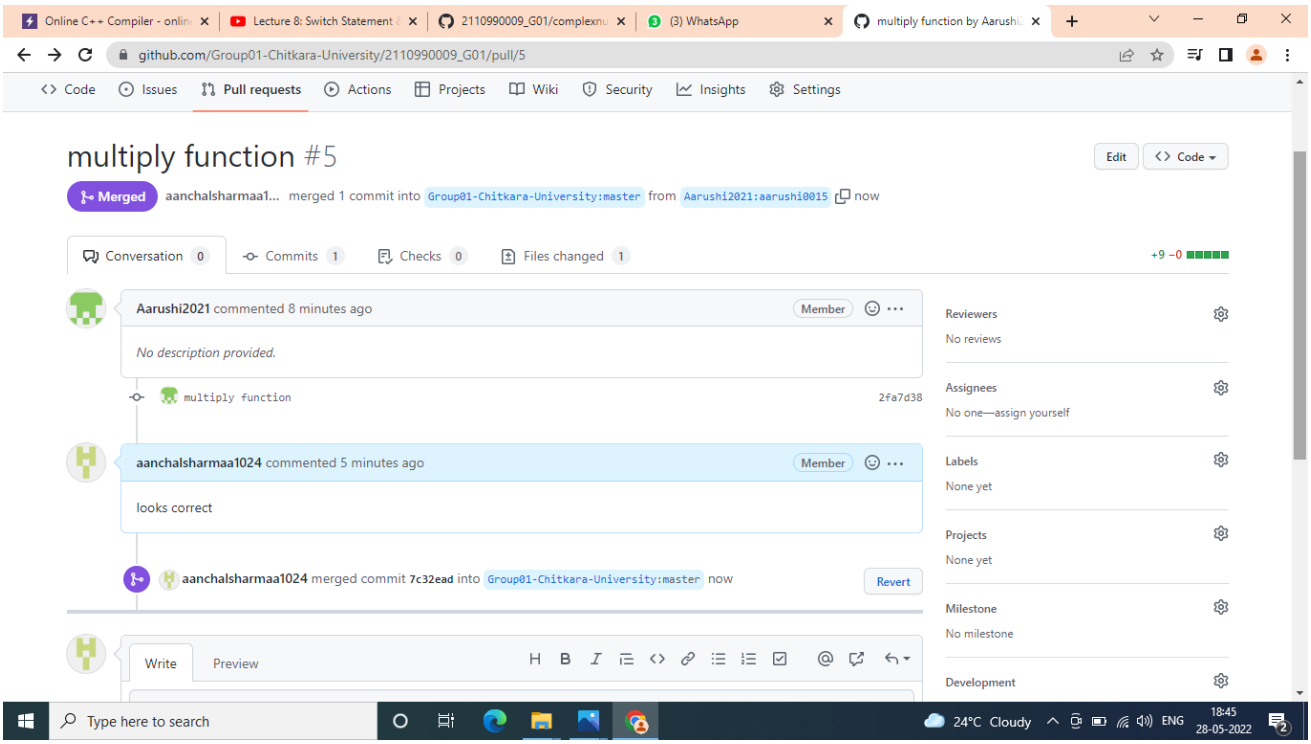
#2 TEAM MEMBERS'S PULL REQUEST





#3 TEAM MEMBER'S PULL REQUEST



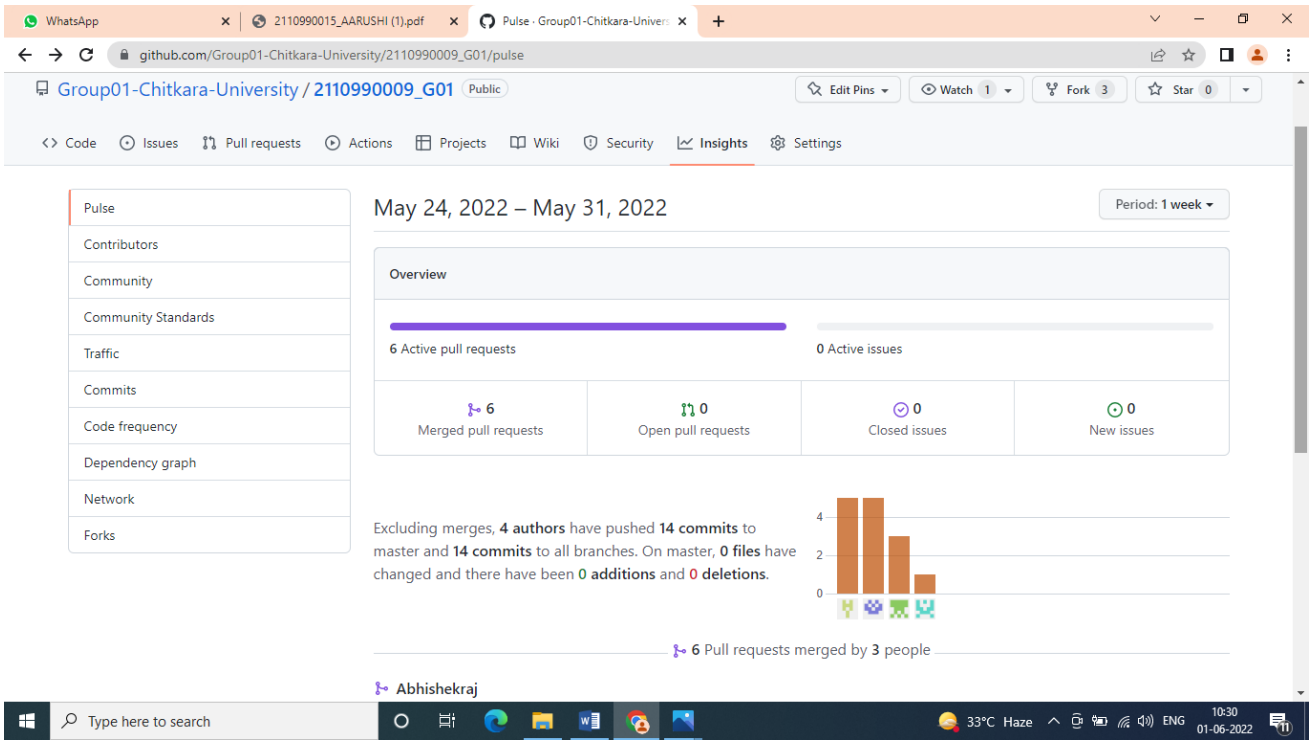


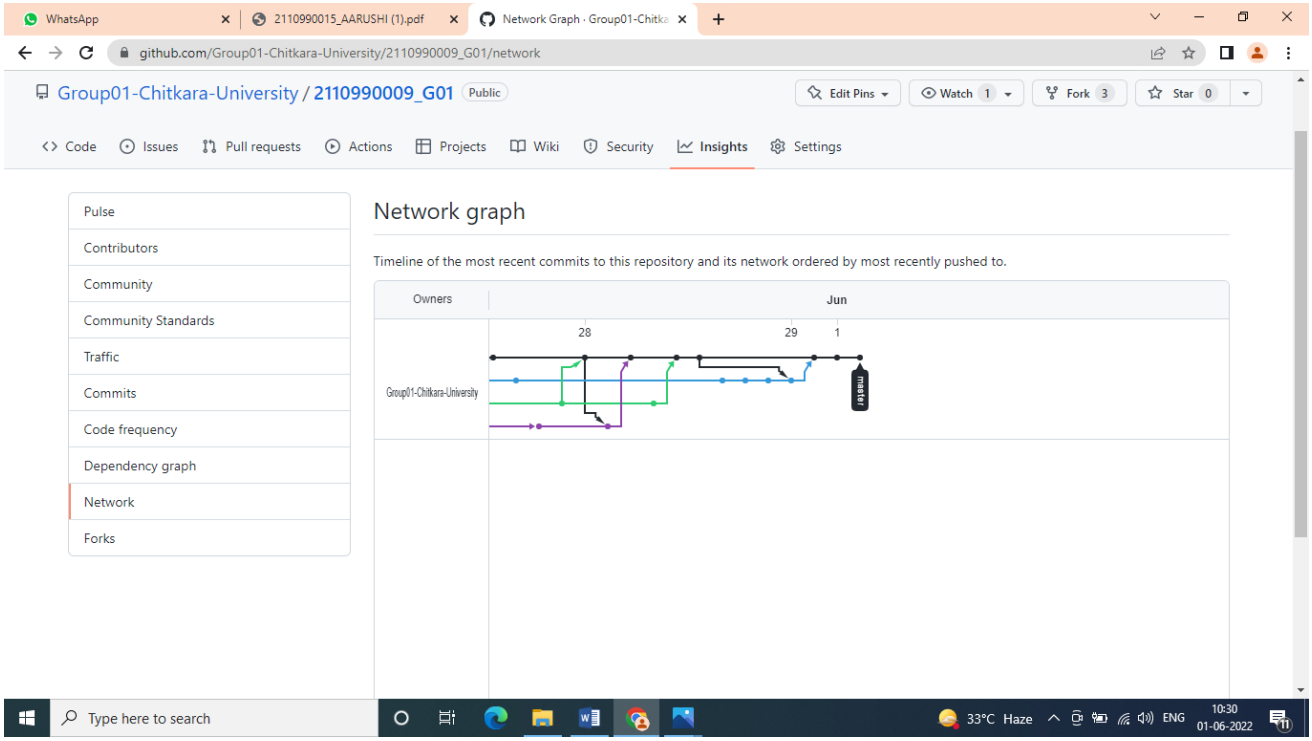
NETWORK GRAPHS

To view the network graphs of your repository, follow the steps:

1. Go to the repository of which you want the graph/details.
2. Click on the ‘Insights’ option it the menu bar.
3. In the right menu list click on network.
4. You can see the network graph there.

It shows the timeline of the most recent commits to this repository and its network ordered by most recently pushed to.





The points in the network graph represents the commits. By hovering over the points, you can see the information about the commit such as author, checksum, message of commit