

Source Code Management

Task 1.1

&

Task 1.2

&

Task 2

(CS181)

Submitted by

Name-Aanchal Sharma

Roll No.- 2110990009

CHITKARA
UNIVERSITY



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:

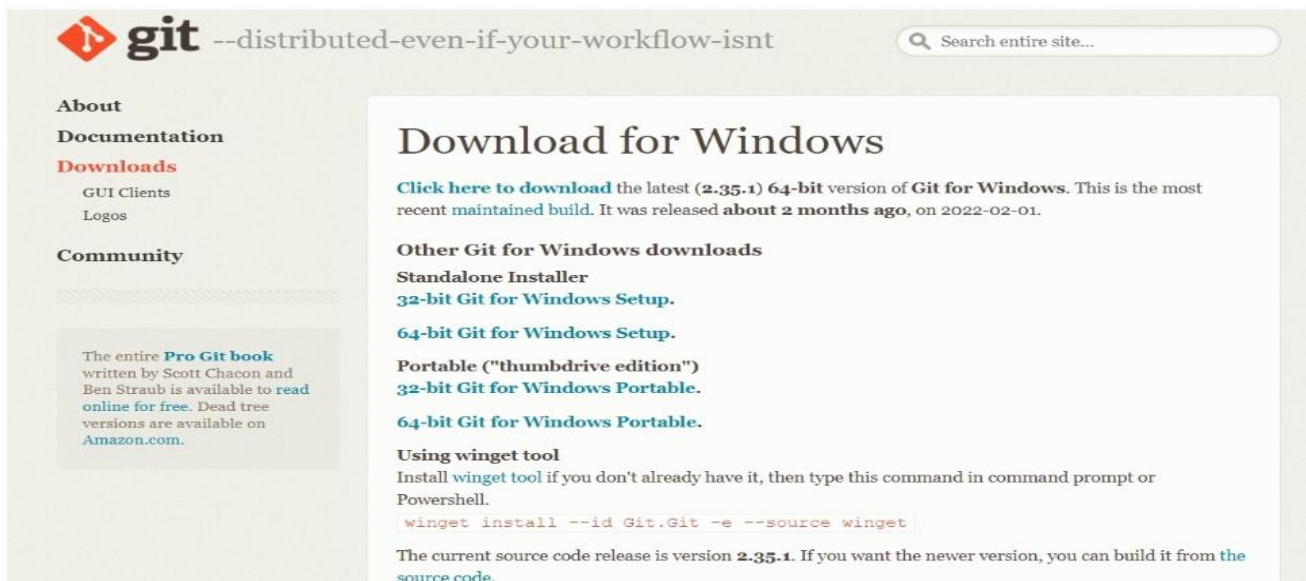
GIT → It is basically used for pushing and pulling of code. We can use git and git-hub parallelly to work with multiple members or individually. We can make , edit , recreate ,copy or download any code on git hub using git.

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:



The screenshot shows the Git website's 'Download for Windows' page. The header features the Git logo and the tagline '--distributed-even-if-your-workflow-isnt'. A search bar is located in the top right corner. The left sidebar contains navigation links: 'About', 'Documentation', 'Downloads' (highlighted), 'GUI Clients', 'Logos', and 'Community'. The main content area is titled 'Download for Windows' and includes a link to download the latest (2.35.1) 64-bit version of Git for Windows. It also lists other download options: 'Standalone Installer', '32-bit Git for Windows Setup.', '64-bit Git for Windows Setup.', 'Portable ("thumbdrive edition")', '32-bit Git for Windows Portable.', and '64-bit Git for Windows Portable.'. A section titled 'Using winget tool' provides instructions on how to install Git using the winget command. The footer mentions the current source code release is version 2.35.1.

git --distributed-even-if-your-workflow-isnt

Search entire site...

About
Documentation
Downloads
GUI Clients
Logos
Community

The entire **Pro Git book** written by Scott Chacon and Ben Straub is available to read online for free. Dead tree versions are available on Amazon.com.

Download for Windows

[Click here to download](#) the latest (2.35.1) 64-bit version of Git for Windows. This is the most recent maintained build. It was released about 2 months ago, on 2022-02-01.

Other Git for Windows downloads

Standalone Installer
[32-bit Git for Windows Setup.](#)
[64-bit Git for Windows Setup.](#)






Portable ("thumbdrive edition")
[32-bit Git for Windows Portable.](#)
[64-bit Git for Windows Portable.](#)

Using winget tool

Install winget tool if you don't already have it, then type this command in command prompt or Powershell.

```
winget install --id Git.Git -e --source winget
```

The current source code release is version 2.35.1. If you want the newer version, you can build it from [the source code](#).

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-I9T8D20 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.

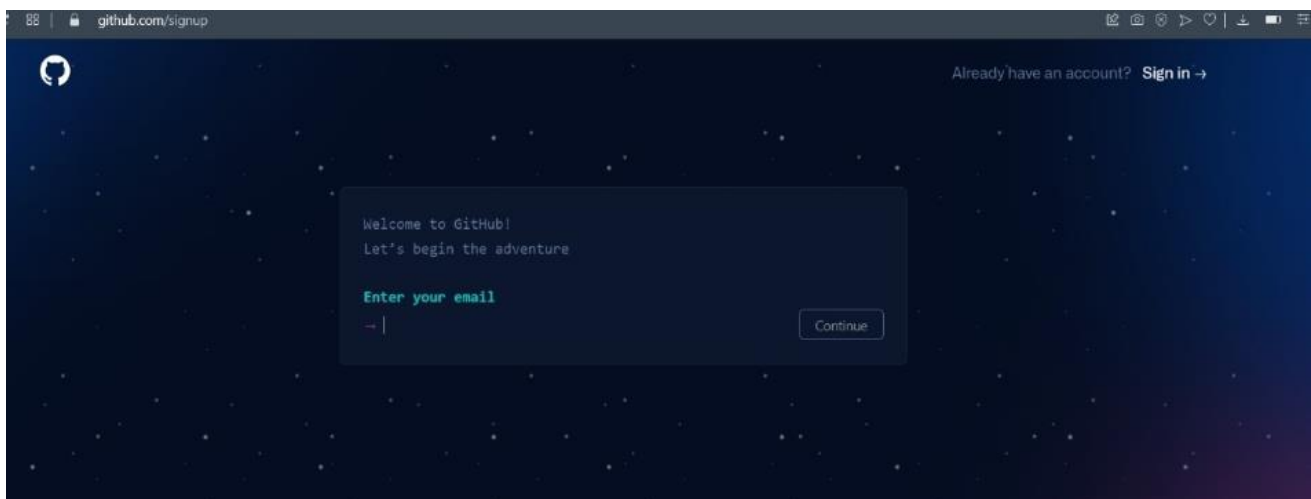
Procedure:-

Step1 :-

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

Step2 :-

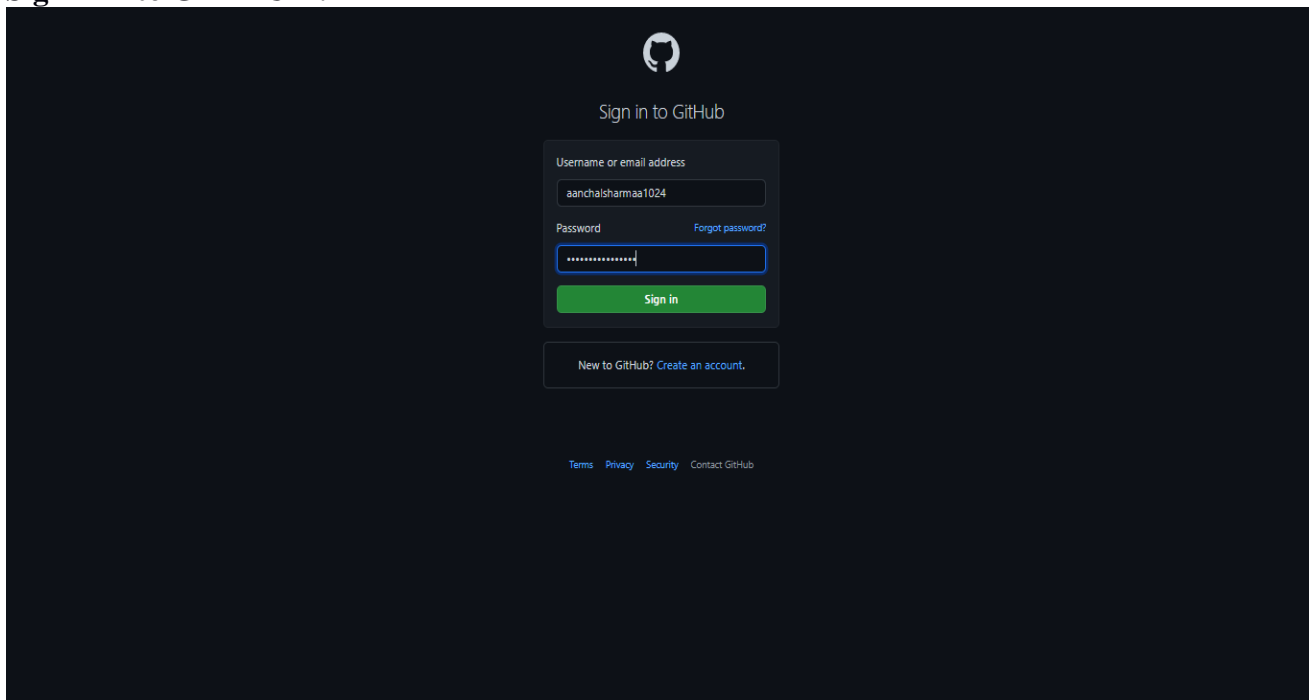
Snapshots –



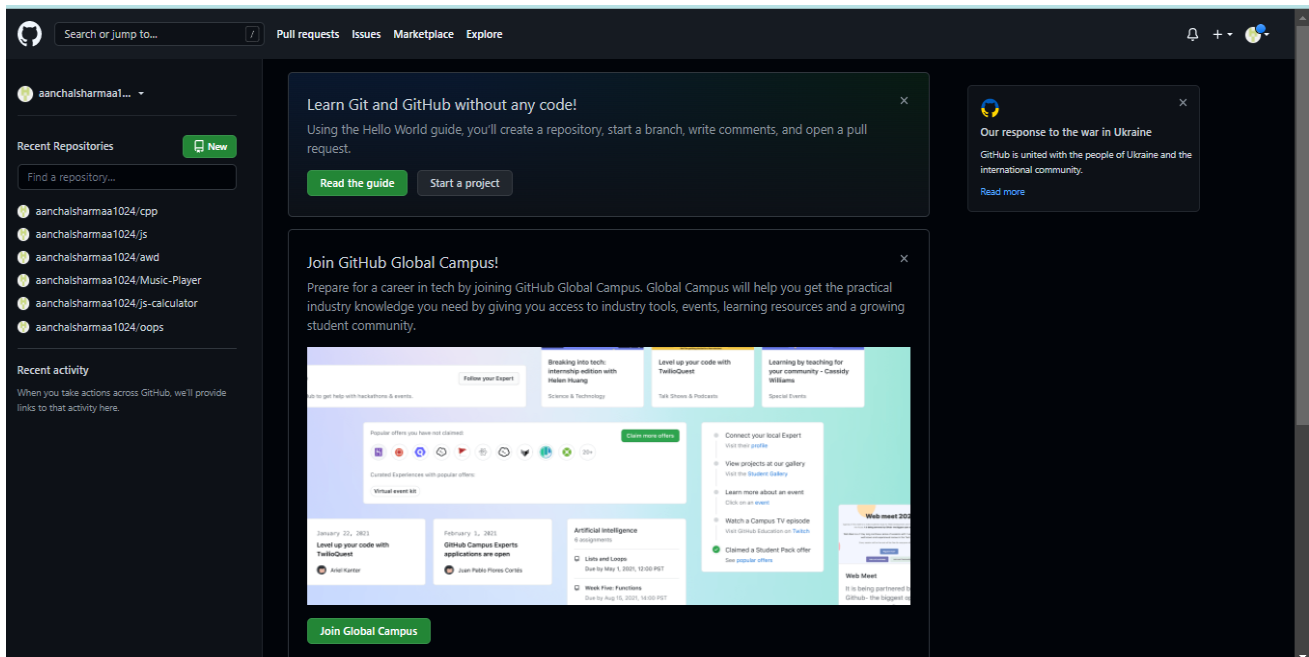
After visiting the link this type of interface will appear, if you already have account you can sign in

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:-

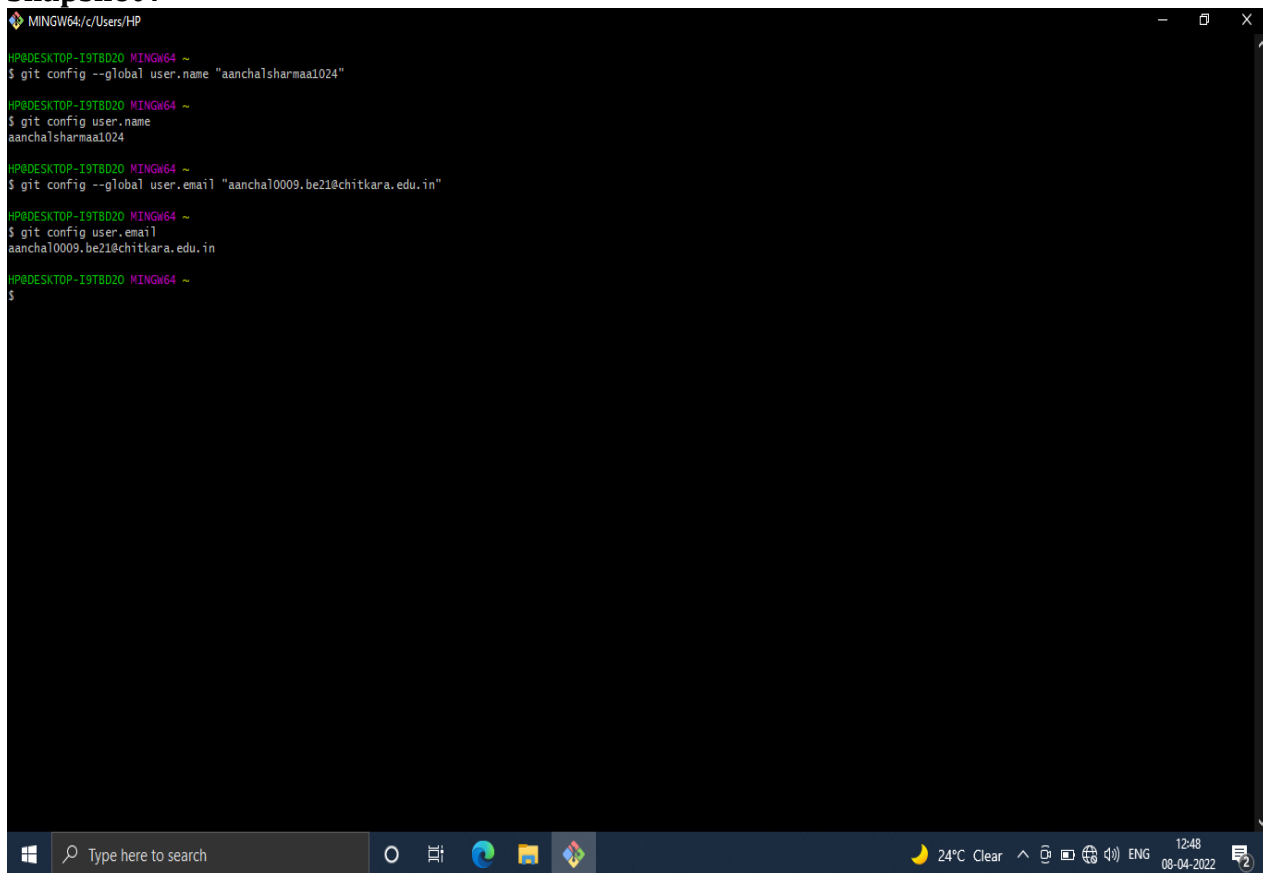
git config --global user.email "your email in git-hub"

To verify:-

git config user.name

git config user.email

Snapshot :-



```
MINGW64/c/Users/HP
HPDESKTOP-I9T8D20 MINGW64 ~
$ git config --global user.name "aanchalsharmaa1024"
HPDESKTOP-I9T8D20 MINGW64 ~
$ git config user.name
aanchalsharmaa1024
HPDESKTOP-I9T8D20 MINGW64 ~
$ git config --global user.email "aanchal0009.be21@chitkara.edu.in"
HPDESKTOP-I9T8D20 MINGW64 ~
$ git config user.email
aanchal0009.be21@chitkara.edu.in
HPDESKTOP-I9T8D20 MINGW64 ~
$
```

The screenshot shows a Windows 10 taskbar at the bottom with the search bar, taskbar icons, and system tray. The command prompt window is titled 'MINGW64/c/Users/HP' and shows the execution of git configuration commands. The output of 'git config user.name' is 'aanchalsharmaa1024' and the output of 'git config user.email' is 'aanchal0009.be21@chitkara.edu.in'.

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 –

```
1 file changed, 26 insertions(+)

HP@DESKTOP-I9TBD20 MINGW64 /d/js (master)
$ git log
commit 7a8b073a32b98f26bf332aa2e28034f7141c13a5 (HEAD -> master)
Author: aanchalsharmaa1024 <aanchal0009.be21@chitkara.edu.in>
Date: Fri Apr 8 13:06:02 2022 +0530

    code added

commit c1868658da8311a78a3f7b7ea1a348e71005c4e6
Author: aanchalsharmaa1024 <aanchal0009.be21@chitkara.edu.in>
Date: Fri Apr 8 13:02:50 2022 +0530

    dec object

commit 8f0eb74b737fb26bf277a3694a2fd3d32b9055aa
Author: aanchalsharmaa1024 <aanchal0009.be21@chitkara.edu.in>
Date: Fri Apr 8 12:58:14 2022 +0530

    switch case

HP@DESKTOP-I9TBD20 MINGW64 /d/js (master)
$ |
```

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 –



```
MINGW64:/d/awd

HP@DESKTOP-I9TBD20 MINGW64 ~
$ cd D:

HP@DESKTOP-I9TBD20 MINGW64 /d
$ mkdir awd
mkdir: cannot create directory 'awd': File exists

HP@DESKTOP-I9TBD20 MINGW64 /d
$ cd awd

HP@DESKTOP-I9TBD20 MINGW64 /d/awd (master)
$ git branch
* master
```

2. To change the present working branch.
git checkout name of branch.

```

HP@DESKTOP-I9TBD20 MINGW64 /d/awd (master)
$ git branch feature

HP@DESKTOP-I9TBD20 MINGW64 /d/awd (master)
$ git branch
  feature
* master

HP@DESKTOP-I9TBD20 MINGW64 /d/awd (master)
$ git checkout feature
Switched to branch 'feature'

HP@DESKTOP-I9TBD20 MINGW64 /d/awd (feature)
$ git branch
* feature
  master

```

Visualizing branches :-

```

HP@DESKTOP-I9TBD20 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-I9TBD20 MINGW64 /d/awd (feature)
$

```

```

MINGW64/d/awd

HP@DESKTOP-I9TBD20 MINGW64 /d/awd (feature)
$ vi day3.html

HP@DESKTOP-I9TBD20 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-I9TBD20 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-I9TBD20 MINGW64 /d/awd (feature)
$ git commit -m "one comment is made"
[feature 4c02a0a] one comment is made
1 file changed, 1 insertion(+)

HP@DESKTOP-I9TBD20 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-I9TBD20 MINGW64 /d/awd (feature)
$ git checkout master
Switched to branch 'master'
Your branch is up to date with 'origin/master'.

HP@DESKTOP-I9TBD20 MINGW64 /d/awd (master)
$ git log --oneline
50bac7 (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-I9TBD20 MINGW64 /d/awd (master)
$ vi day3.html

HP@DESKTOP-I9TBD20 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:
```

```

HP@DESKTOP-I9TBD20 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:
  (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-I9TBD20 MINGW64 /d/awd (master)
$ git add .

HP@DESKTOP-I9TBD20 MINGW64 /d/awd (master)
$ git status
On branch master
Your branch is up to date with 'origin/master'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   day3.html

HP@DESKTOP-I9TBD20 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.

```

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.

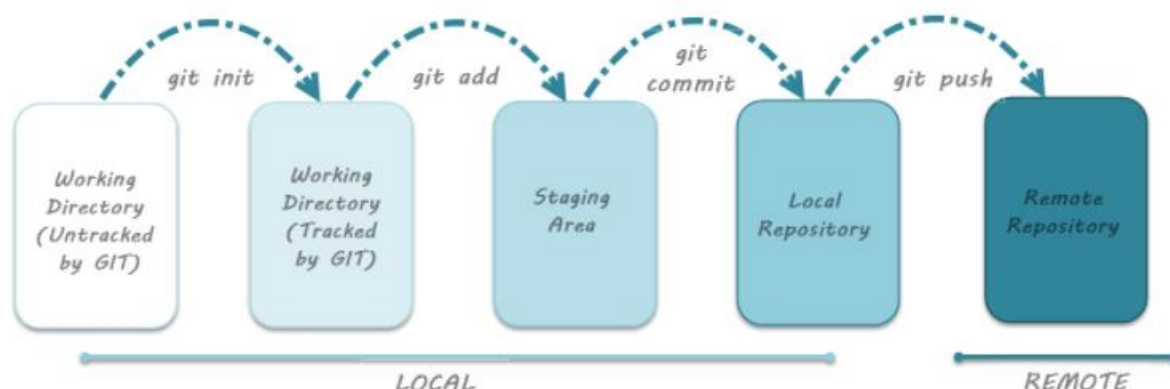
Staging Area ->

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
HP0DESKTOP-I9TBD20 MINGW64 ~
$ cd D:

HP0DESKTOP-I9TBD20 MINGW64 /d
$ mkdir awd
mkdir: cannot create directory 'awd': File exists

HP0DESKTOP-I9TBD20 MINGW64 /d
$ cd awd

HP0DESKTOP-I9TBD20 MINGW64 /d/awd (master)
$ vi day8.html

HP0DESKTOP-I9TBD20 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

HP0DESKTOP-I9TBD20 MINGW64 /d/awd (master)
$ git init
Reinitialized existing Git repository in D:/awd/.git/

HP0DESKTOP-I9TBD20 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

HP0DESKTOP-I9TBD20 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

HP0DESKTOP-I9TBD20 MINGW64 /d/awd (master)
$ git commit -m "comment added"
[master 4a9bfee] comment added
1 file changed, 2 insertions(+), 1 deletion(-)

HP0DESKTOP-I9TBD20 MINGW64 /d/awd (master)
$ git remote add origin git@github.com:aanchalsharmaa1024/js-calculator.git
error: remote origin already exists.

HP0DESKTOP-I9TBD20 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/gl3.git
   50baca7..4a9bfee  master -> master
branch 'master' set up to track 'origin/master'.

HP0DESKTOP-I9TBD20 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

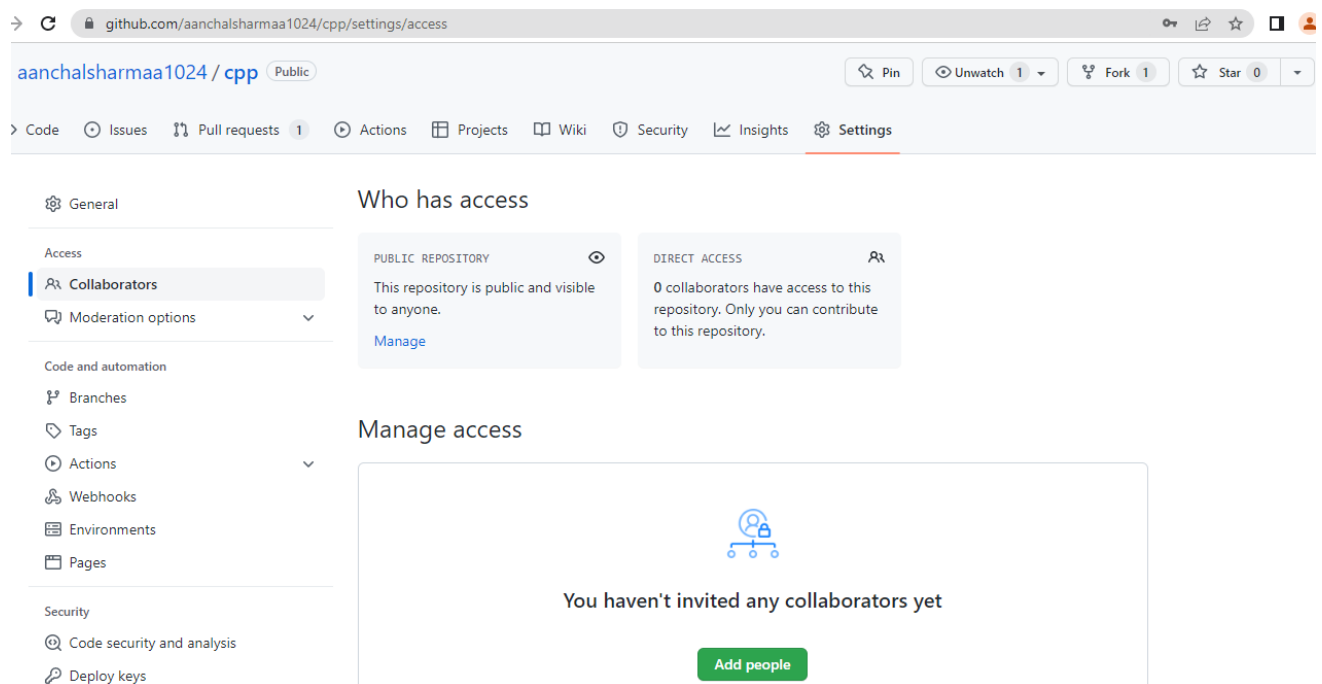
In GitHub, we can invite other GitHub users to become collaborators to our private repositories (which expires after 7 days if not accepted, restoring any unclaimed licenses). Being a collaborator, of a personal repository you can pull (read) the contents of the repository and push (write) changes to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

The screenshot shows the GitHub interface for the repository 'Aarushi2021 / SCM-PROJECT'. The repository is public and has 1 watch, 0 forks, and 0 stars. The main content area displays a list of files and their commit history. The files listed are: 3.2 lecture.cpp, 3.2 lecture.exe, To find maximum of 3no..cpp, To find maximum of 3no..exe, abc.txt, area of rectangle.cpp, area of rectangle.exe, boysituation.cpp, boysituation.exe, into.cpp, and into.exe. The commit history for each file shows the commit message, the commit hash, the time ago, and the number of commits. The 'About' section on the right indicates that there is no description, website, or topics provided for this repository.

File	Commit Message	Commit Hash	Time Ago	Commits
3.2 lecture.cpp	first	e2b9dee	9 days ago	6
3.2 lecture.exe	first	e2b9dee	9 days ago	6
To find maximum of 3no..cpp	first	e2b9dee	9 days ago	6
To find maximum of 3no..exe	first	e2b9dee	9 days ago	6
abc.txt	added abc	e2b9dee	7 days ago	6
area of rectangle.cpp	first	e2b9dee	9 days ago	6
area of rectangle.exe	first	e2b9dee	9 days ago	6
boysituation.cpp	first	e2b9dee	9 days ago	6
boysituation.exe	first	e2b9dee	9 days ago	6
into.cpp	revert	e2b9dee	9 days ago	6
into.exe	first	e2b9dee	9 days ago	6

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.

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 *

aanchalsharmaa1024

Repository name *

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



Forking Aarushi2021/SCM-PROJECT

It should only take a few seconds.

Refresh



© 2022 GitHub, Inc.

Terms

Privacy

Security

Status

Docs

Contact GitHub

Pricing

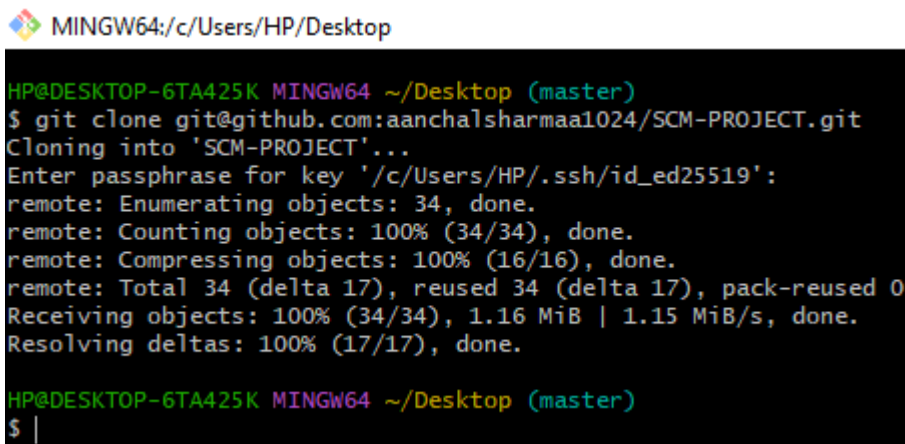
API

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.

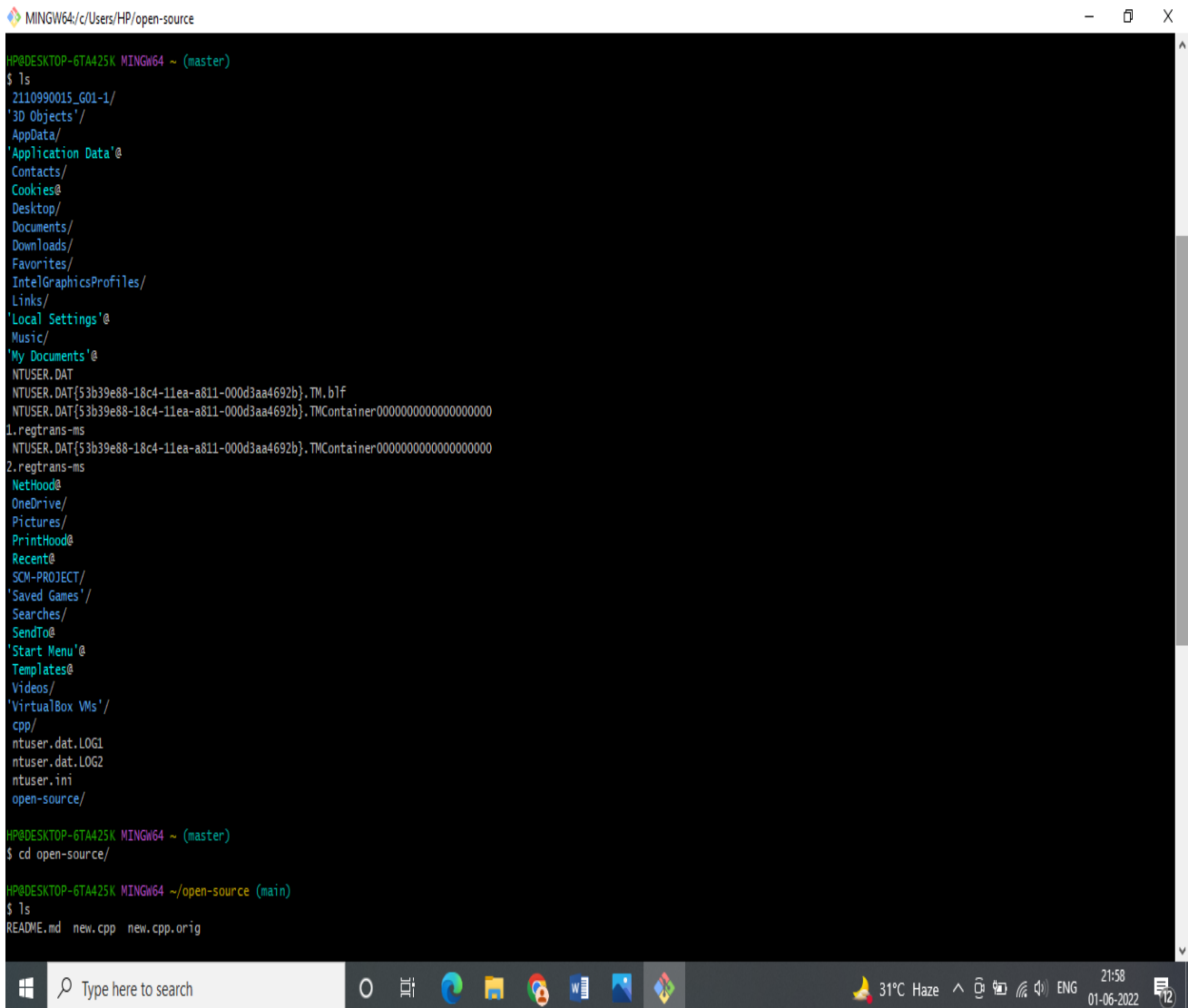
1. Once you have forked the repository, you can clone it into your computer using directly the option given on github or through running git clone command in git bash.
2. Copy the URL of the forked repository
3. Open git bash and type the command “ git clone <url of the forked repository>”

A screenshot of a terminal window with a black background and white text. The window title is 'MINGW64:/c/Users/HP/Desktop'. The prompt is 'HP@DESKTOP-6TA425K MINGW64 ~/Desktop (master)'. The user enters '\$ git clone git@github.com:aanchalsharmaa1024/SCM-PROJECT.git'. The output shows the cloning process: 'Cloning into 'SCM-PROJECT'...', 'Enter passphrase for key '/c/Users/HP/.ssh/id_ed25519':', 'remote: Enumerating objects: 34, done.', 'remote: Counting objects: 100% (34/34), done.', 'remote: Compressing objects: 100% (16/16), done.', 'remote: Total 34 (delta 17), reused 34 (delta 17), pack-reused 0', 'Receiving objects: 100% (34/34), 1.16 MiB | 1.15 MiB/s, done.', 'Resolving deltas: 100% (17/17), done.'. The prompt returns to '\$ |'.

COMMITTING 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.



```
MINGW64~/c/Users/HP/open-source
HP@DESKTOP-6TA425K MINGW64 ~ (master)
$ ls
2110990015_G01-1/
'3D Objects'/
AppData/
'Application Data'@
Contacts/
Cookies@
Desktop/
Documents/
Downloads/
Favorites/
IntelGraphicsProfiles/
Links/
'Local Settings'@
Music/
'My Documents'@
NTUSER.DAT
NTUSER.DAT{$3b39e88-18c4-11ea-a811-000d3aa4692b}.TM.b1f
NTUSER.DAT{$3b39e88-18c4-11ea-a811-000d3aa4692b}.TMContainer000000000000000000
1.regtrans-ms
NTUSER.DAT{$3b39e88-18c4-11ea-a811-000d3aa4692b}.TMContainer000000000000000000
2.regtrans-ms
NetHood@
OneDrive/
Pictures/
PrintHood@
Recent@
SCM-PROJECT/
'Saved Games'/
Searches/
SendTo@
'Start Menu'@
Templates@
Videos/
'VirtualBox VMs'/
cpp/
ntuser.dat.LOG1
ntuser.dat.LOG2
ntuser.ini
open-source/

HP@DESKTOP-6TA425K MINGW64 ~ (master)
$ cd open-source/

HP@DESKTOP-6TA425K MINGW64 ~/open-source (main)
$ ls
README.md new.cpp new.cpp.orig
```



```
MINGW64/c/Users/HP/open-source
VirtualBox VMs/
cpp/
ntuser.dat.LOG1
ntuser.dat.LOG2
ntuser.ini
open-source/

HP@DESKTOP-6TA425K MINGW64 ~ (master)
$ cd open-source/

HP@DESKTOP-6TA425K MINGW64 ~/open-source (main)
$ ls
README.md  new.cpp  new.cpp.orig

HP@DESKTOP-6TA425K MINGW64 ~/open-source (main)
$ vi new.cpp

HP@DESKTOP-6TA425K MINGW64 ~/open-source (main)
$ git remote -v
origin  git@github.com:aanchalsharma1024/open-source.git (fetch)
origin  git@github.com:aanchalsharma1024/open-source.git (push)

HP@DESKTOP-6TA425K MINGW64 ~/open-source (main)
$ git init
Reinitialized existing Git repository in C:/Users/HP/open-source/.git/

HP@DESKTOP-6TA425K MINGW64 ~/open-source (main)
$ git add .

HP@DESKTOP-6TA425K MINGW64 ~/open-source (main)
$ git commit -m "final commit"
[main 1c59e66] final commit
1 file changed, 1 insertion(+)

HP@DESKTOP-6TA425K MINGW64 ~/open-source (main)
$ git push -u origin main
Enter passphrase for key '/c/Users/HP/.ssh/id_ed25519':
Enumerating objects: 14, done.
Counting objects: 100% (14/14), done.
Delta compression using up to 4 threads
Compressing objects: 100% (12/12), done.
Writing objects: 100% (12/12), 1.10 KiB | 374.00 KiB/s, done.
Total 12 (delta 6), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (6/6), completed with 1 local object.
To github.com:aanchalsharma1024/open-source.git
   6904717..1c59e66  main -> main
branch 'main' set up to track 'origin/main'.

HP@DESKTOP-6TA425K MINGW64 ~/open-source (main)
$ 9~
```

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 , merge conflicts Firstly on the master branch
Change ,add and commit the file.
2. And then make a new branch and switch to another branch
Using the command , git branch branch name
git checkout branch name.

MINGW64:/c:/Users/HP/Desktop/new

```
HP@DESKTOP-6TA425K MINGW64 ~/Desktop (master)
$ mkdir new
mkdir: cannot create directory 'new': File exists

HP@DESKTOP-6TA425K MINGW64 ~/Desktop (master)
$ cd new

HP@DESKTOP-6TA425K MINGW64 ~/Desktop/new (master)
$ ls
array.cpp

HP@DESKTOP-6TA425K MINGW64 ~/Desktop/new (master)
$ vi array.cpp

HP@DESKTOP-6TA425K MINGW64 ~/Desktop/new (master)
$ git add .

HP@DESKTOP-6TA425K MINGW64 ~/Desktop/new (master)
$ git commit -m "initial commit"
[master dc627fd] initial commit
1 file changed, 1 insertion(+), 1 deletion(-)

HP@DESKTOP-6TA425K MINGW64 ~/Desktop/new (master)
$ git branch aanchal

HP@DESKTOP-6TA425K MINGW64 ~/Desktop/new (master)

HP@DESKTOP-6TA425K MINGW64 ~/Desktop/new (master)
$ git checkout aanchal
Switched to branch 'aanchal'

HP@DESKTOP-6TA425K MINGW64 ~/Desktop/new (aanchal)
$ vi array.cpp
```

3. Now in the new branch again change the same file .and if your changes are in the same line
4. Then while merging the two branches , merge conflict will arise.

```
MINGW64/c/Users/HP/Desktop/new
Switched to branch 'aanchal'

HP@DESKTOP-6TA425K MINGW64 ~/Desktop/new (aanchal)
$ vi array.cpp

HP@DESKTOP-6TA425K MINGW64 ~/Desktop/new (aanchal)
$ git add .

HP@DESKTOP-6TA425K MINGW64 ~/Desktop/new (aanchal)
$ git commit -m "Final commit"
[aanchal d9da901] final commit
1 file changed, 1 insertion(+)

HP@DESKTOP-6TA425K MINGW64 ~/Desktop/new (aanchal)
$ git checkout master
Switched to branch 'master'
Your branch is ahead of 'origin/master' by 3 commits.
(use "git push" to publish your local commits)

HP@DESKTOP-6TA425K MINGW64 ~/Desktop/new (master)
$ git merge aanchal
Updating d627fd..d9da901
Fast-forward
 array.cpp | 1 +
1 file changed, 1 insertion(+)

HP@DESKTOP-6TA425K MINGW64 ~/Desktop/new (master)
$ vi array.cpp

HP@DESKTOP-6TA425K MINGW64 ~/Desktop/new (master)
$ git add .

HP@DESKTOP-6TA425K MINGW64 ~/Desktop/new (master)
$ git commit -m "class str"
[master d03a63a] class str
1 file changed, 3 insertions(+), 1 deletion(-)

HP@DESKTOP-6TA425K MINGW64 ~/Desktop/new (master)
$ git checkout aanchal
Switched to branch 'aanchal'

HP@DESKTOP-6TA425K MINGW64 ~/Desktop/new (aanchal)
$ vi array.cpp

HP@DESKTOP-6TA425K MINGW64 ~/Desktop/new (aanchal)
$ git add .

HP@DESKTOP-6TA425K MINGW64 ~/Desktop/new (aanchal)
$ git commit -m "polynomial"
[aanchal 6dc0802] polynomial
```

```
HP@DESKTOP-6TA425K MINGW64 ~/Desktop/new (aanchal)
$ git checkout master
Switched to branch 'master'
Your branch is ahead of 'origin/master' by 5 commits.
(use "git push" to publish your local commits)

HP@DESKTOP-6TA425K MINGW64 ~/Desktop/new (master)
$ git merge aanchal
Auto-merging array.cpp
CONFLICT (content): Merge conflict in array.cpp
Automatic merge failed; fix conflicts and then commit the result.

HP@DESKTOP-6TA425K MINGW64 ~/Desktop/new (master|MERGING)
$ git mergetool

This message is displayed because 'merge.tool' is not configured.
See 'git mergetool --tool-help' or 'git help config' for more details.
'git mergetool' will now attempt to use one of the following tools:
opendiff kdiff3 tkdiff xxdiff meld tortoisemerge gvimdiff diffuze diffmerge ecmerge p4merge araxis bc codecompare smerge emerge vimdiff nvimdiff
Merging:
array.cpp

Normal merge conflict for 'array.cpp':
{local}: modified file
{remote}: modified file
Hit return to start merge resolution tool (vimdiff):
4 files to edit
```

5. Now we will resolve the merge conflict using the mergetool command .

```

#include<iostream>
using namespace std;
Hero{
    //structure of a class
}

//this is a simple code of c++

+ -- 11 lines: this line added for reset-----
~
~
~

./array_LOCAL_649.cpp [dos] (22:16 01/06/2022) 1,1 A11
#include<iostream>
using namespace std;
Hero{
    //structure of a class
}

//this is a simple code of c++

+ -- 11 lines: this line added for reset-----
~
~
~

./array_BASE_649.cpp [dos] (22:16 01/06/2022) 1,1 A11
#include<iostream>
using namespace std;
Polynomial{
    //STRUCTURE OF A CLASS
}

//this is a simple code of c++

+ -- 11 lines: this line added for reset-----
~
~
~

./array_REMOTE_649.cpp [dos] (22:16 01/06/2022) 1,1 A11
#include<iostream>
using namespace std;
Hero{
    //structure of a class
}

//this is a simple code of c++

+ -- 11 lines: this line added for reset-----
~
~
~

array.cpp [dos] (22:16 01/06/2022) 4,5 A11
array.cpp [dos] 271, 3198

```

6. After resolving the conflict, add and commit the file .

```

HP@DESKTOP-6TA425K MINGW64 ~/Desktop/new (aanchal)
$ git add .

HP@DESKTOP-6TA425K MINGW64 ~/Desktop/new (aanchal)
$ git commit -m "polynomial"
[aanchal 6dc0802] polynomial

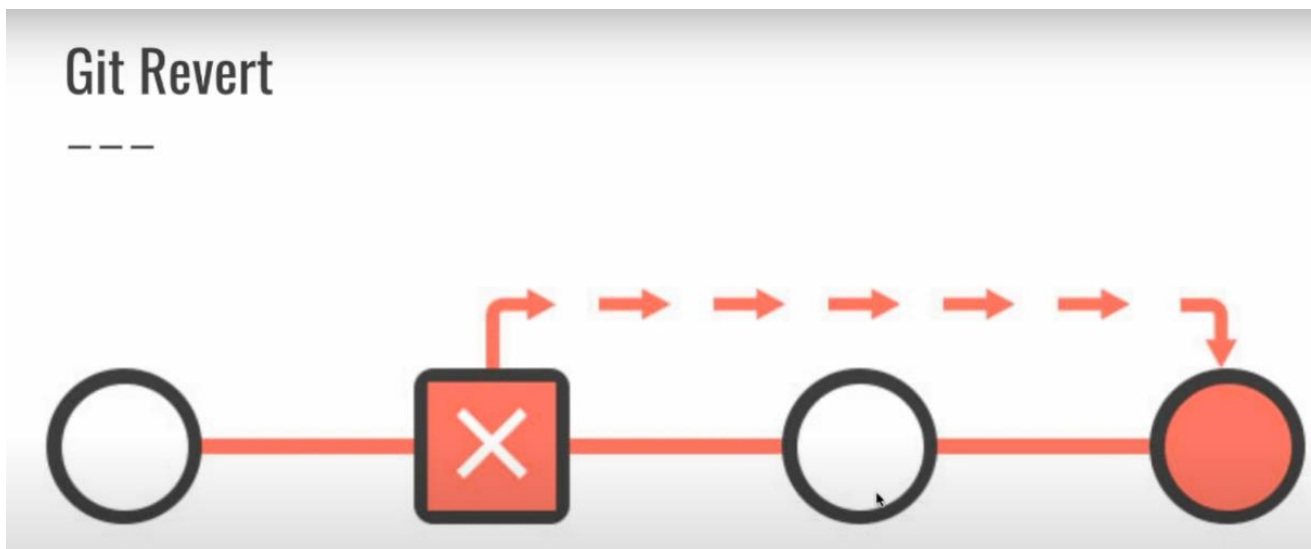
```

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 have pushed a unwanted commit from your local and now we will revert it.

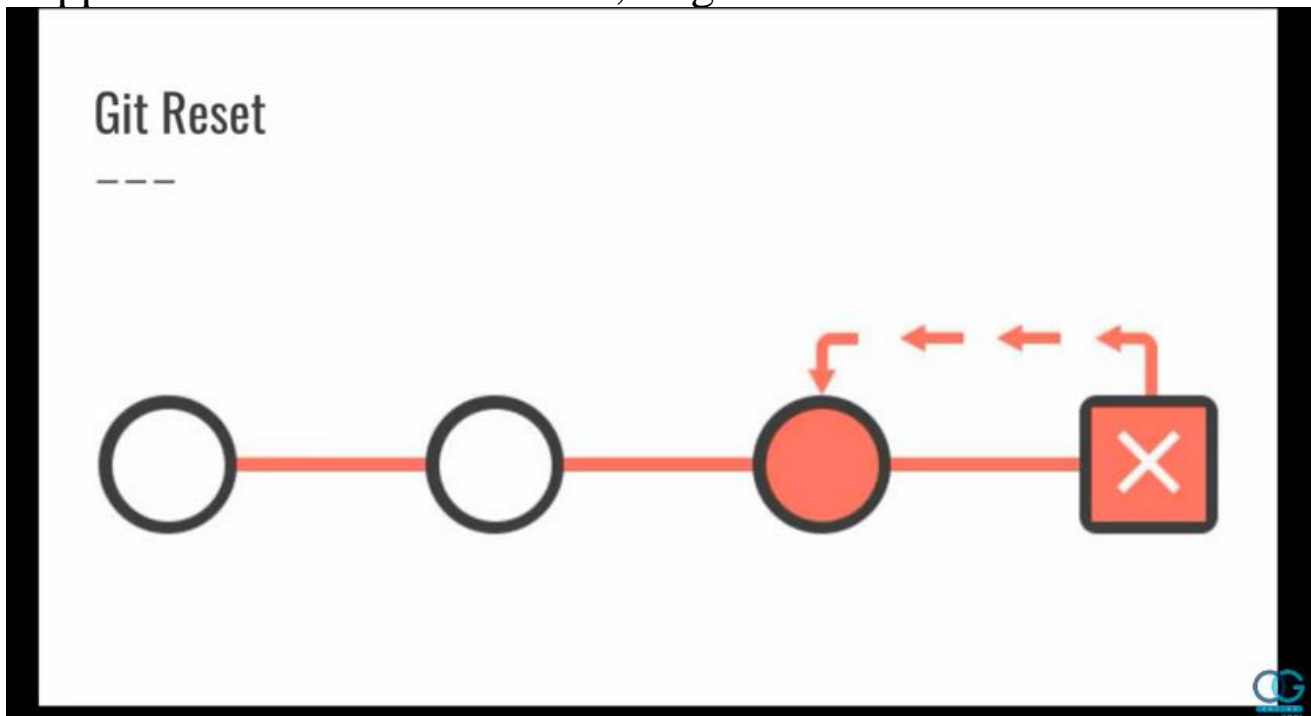
The basic advantage of reverting a commit is that it is not permanently deleted.

```
MINGW64/c/Users/HP/Desktop/new
$ git add .
warning: LF will be replaced by CRLF in array.cpp.
The file will have its original line endings in your working directory
HP@DESKTOP-6TA425K MINGW64 ~/Desktop/new (master|REVERTING)
$ git push -u origin master
Enter passphrase for key '/c/Users/HP/.ssh/id_ed25519':
Everything up-to-date
branch 'master' set up to track 'origin/master'.
HP@DESKTOP-6TA425K MINGW64 ~/Desktop/new (master|REVERTING)
$ git merge --abort
fatal: There is no merge to abort (MERGE_HEAD missing).
HP@DESKTOP-6TA425K MINGW64 ~/Desktop/new (master|REVERTING)
$ git revert --abort
HP@DESKTOP-6TA425K MINGW64 ~/Desktop/new (master)
$ git reset --hard HEAD
HEAD is now at 2ac1e44 final commit
HP@DESKTOP-6TA425K MINGW64 ~/Desktop/new (master)
$ git log --pretty=oneline
fatal: invalid --pretty format: oneline
HP@DESKTOP-6TA425K MINGW64 ~/Desktop/new (master)
$ git log --pretty=oneline
2ac1e4457224bc2a76e0a1d2bb30d62b8973f7a0 (HEAD -> master, origin/master) final c
ommit
5f9de58c17f8119f628d396a59152c078c910607 comment added
91d0b01a42fac06a20a610908444155ea26fec39 first commit
HP@DESKTOP-6TA425K MINGW64 ~/Desktop/new (master)
$ git revert ^C
HP@DESKTOP-6TA425K MINGW64 ~/Desktop/new (master)
$ git revert 5f9de58c17f8119f628d396a59152c078c910607
Auto-merging array.cpp
[master 8f0648e] Revert "comment added"
1 file changed, 1 deletion(-)
HP@DESKTOP-6TA425K MINGW64 ~/Desktop/new (master)
$ git log --pretty=oneline
8f0648e83a69e7b99353b23bae739eb728c1b33f (HEAD -> master) Revert "comment added"
2ac1e4457224bc2a76e0a1d2bb30d62b8973f7a0 (origin/master) final commit
5f9de58c17f8119f628d396a59152c078c910607 comment added
91d0b01a42fac06a20a610908444155ea26fec39 first commit
HP@DESKTOP-6TA425K MINGW64 ~/Desktop/new (master)
$ |
```

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



When to use

--

So if you have bad commits on your local machine then you can use this and revert your code till one point.

--soft

--hard

--mixed

--merge

--keep



```
HP@DESKTOP-6TA425K MINGW64 ~/Desktop/new (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 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:   array.cpp

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

HP@DESKTOP-6TA425K MINGW64 ~/Desktop/new (master)
$ git add .

HP@DESKTOP-6TA425K MINGW64 ~/Desktop/new (master)
$ git commit -m "undo"
bash: git commit: command not found

HP@DESKTOP-6TA425K MINGW64 ~/Desktop/new (master)
$ git commit -m "undo"
[master bb0ae17] undo
1 file changed, 1 insertion(+)

HP@DESKTOP-6TA425K MINGW64 ~/Desktop/new (master)
$ git log --pretty=oneline
bb0ae171fe45c5521995b6a1cf96c68bc78e81d3 (HEAD -> master) undo
8f0648e83a69e7b99353b23bae739eb728c1b33f Revert "comment added"
2ac1e4457224bc2a76e0a1d2bb30d62b8973f7a0 (origin/master) final commit
5f9de58c17f8119f628d396a59152c078c910607 comment added
91d0b01a42fac06a20a610908444155ea26fec39 first commit

HP@DESKTOP-6TA425K MINGW64 ~/Desktop/new (master)
$ ^C

HP@DESKTOP-6TA425K MINGW64 ~/Desktop/new (master)
$ git reset --soft bb0ae171fe45c5521995b6a1cf96c68bc78e81d3

HP@DESKTOP-6TA425K MINGW64 ~/Desktop/new (master)
$ git status
On branch master
Your branch is ahead of 'origin/master' by 2 commits.
  (use "git push" to publish your local commits)

nothing to commit, working tree clean

HP@DESKTOP-6TA425K MINGW64 ~/Desktop/new (master)
$ |
```

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:

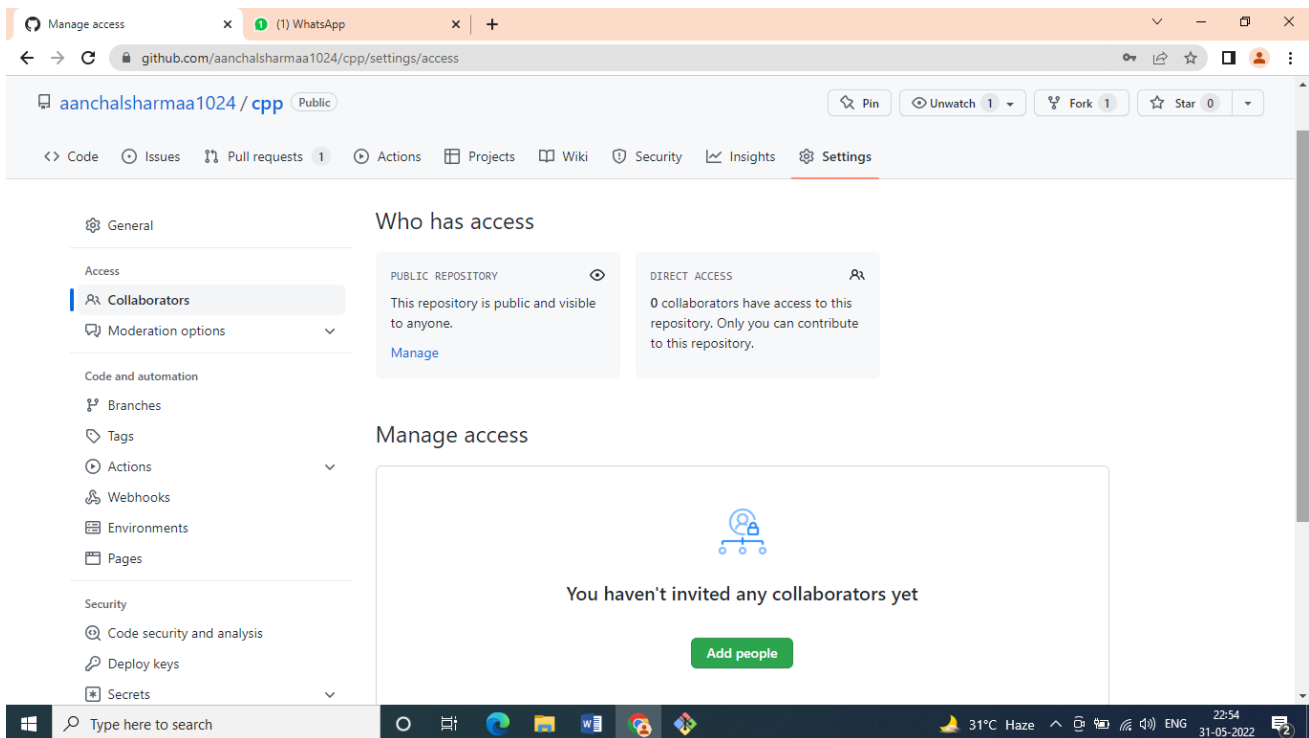
This task is performed in the group of four. Each one of us made it possible to work on this project as if we are doing an open source contribution.

Each one of us create his/her repo and rest of the three contributors 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.

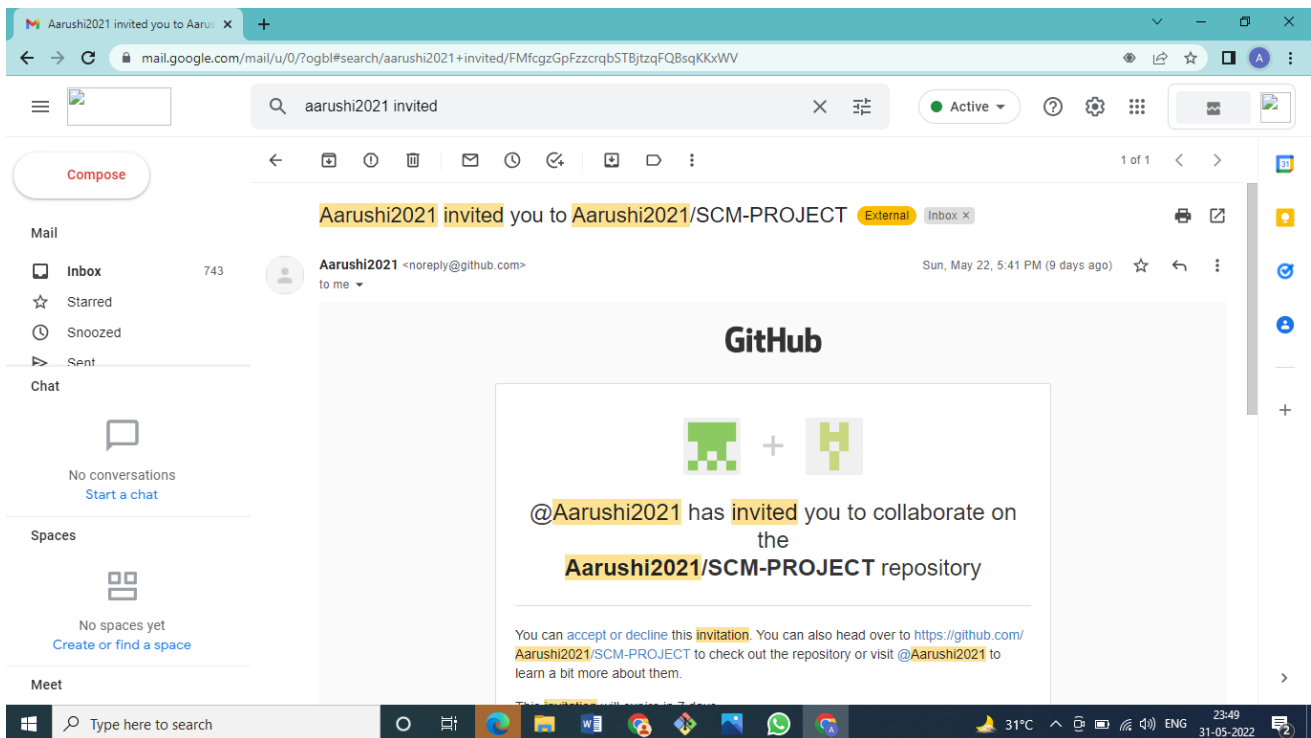
And finally make pull request to the owner of the repo in whose repo we want to make the changes.

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.

MINGW64:/c/Users/HP/Desktop

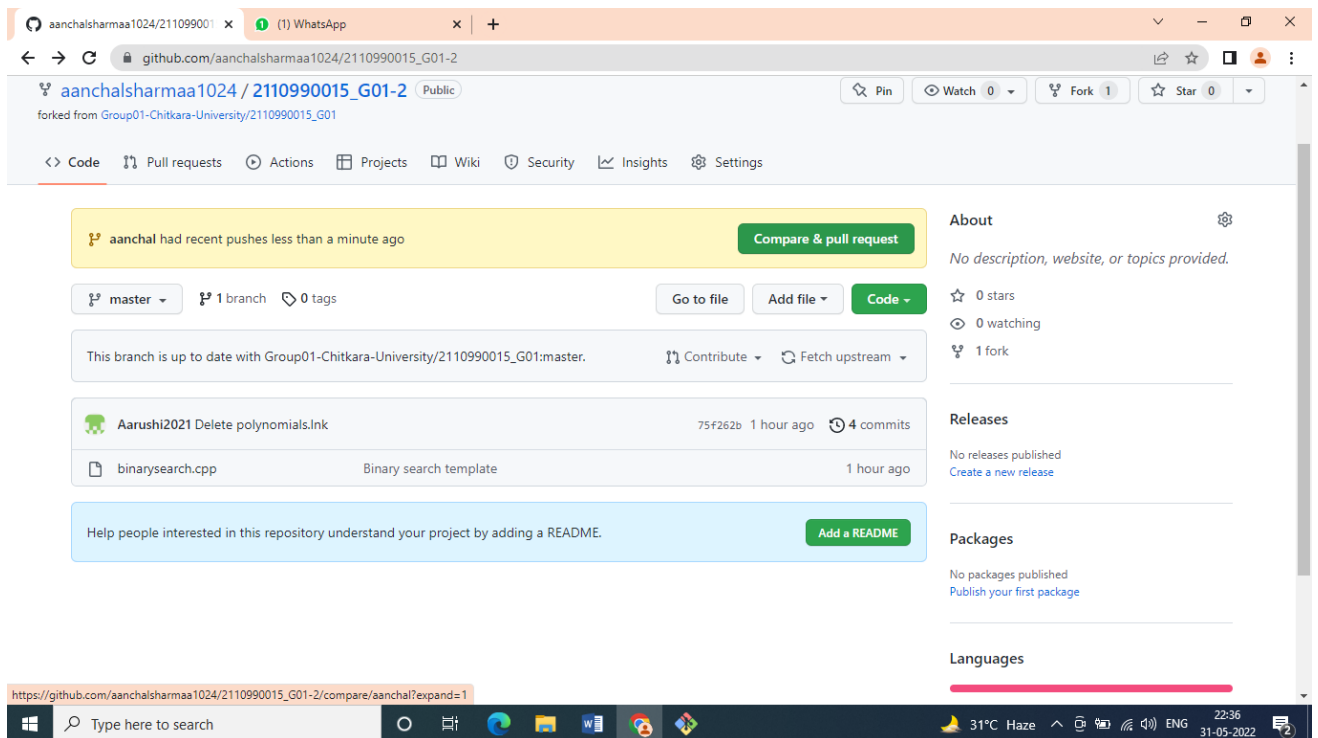
```
HP@DESKTOP-6TA425K MINGW64 ~/Desktop (master)
$ git clone git@github.com:aanchalsharmaa1024/SCM-PROJECT.git
Cloning into 'SCM-PROJECT'...
Enter passphrase for key '/c/Users/HP/.ssh/id_ed25519':
remote: Enumerating objects: 34, done.
remote: Counting objects: 100% (34/34), done.
remote: Compressing objects: 100% (16/16), done.
remote: Total 34 (delta 17), reused 34 (delta 17), pack-reused 0
Receiving objects: 100% (34/34), 1.16 MiB | 1.15 MiB/s, done.
Resolving deltas: 100% (17/17), done.

HP@DESKTOP-6TA425K MINGW64 ~/Desktop (master)
$ |
```

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-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/2110990015_G01 base: master head repository: aanchalsharmaa1024/2110990015_G01 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

Remember, contributions to this repository should follow our [GitHub Community Guidelines](#).

1 commit 1 file changed 1 contributor

Commits on May 31, 2022

search func added by aanchalsharmaa1024

search func added #1

Open aanchalsharmaa1024 wants to merge 1 commit into Group01-Chitkara-University:master from aanchalsharmaa1024:aanchal

Conversation 0 Commits 1 Checks 0 Files changed 1

aanchalsharmaa1024 commented 1 minute ago

No description provided.

search func added d64bf3

aanchalsharmaa1024 commented now

search function in binary search

Add more commits by pushing to the aanchal1 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

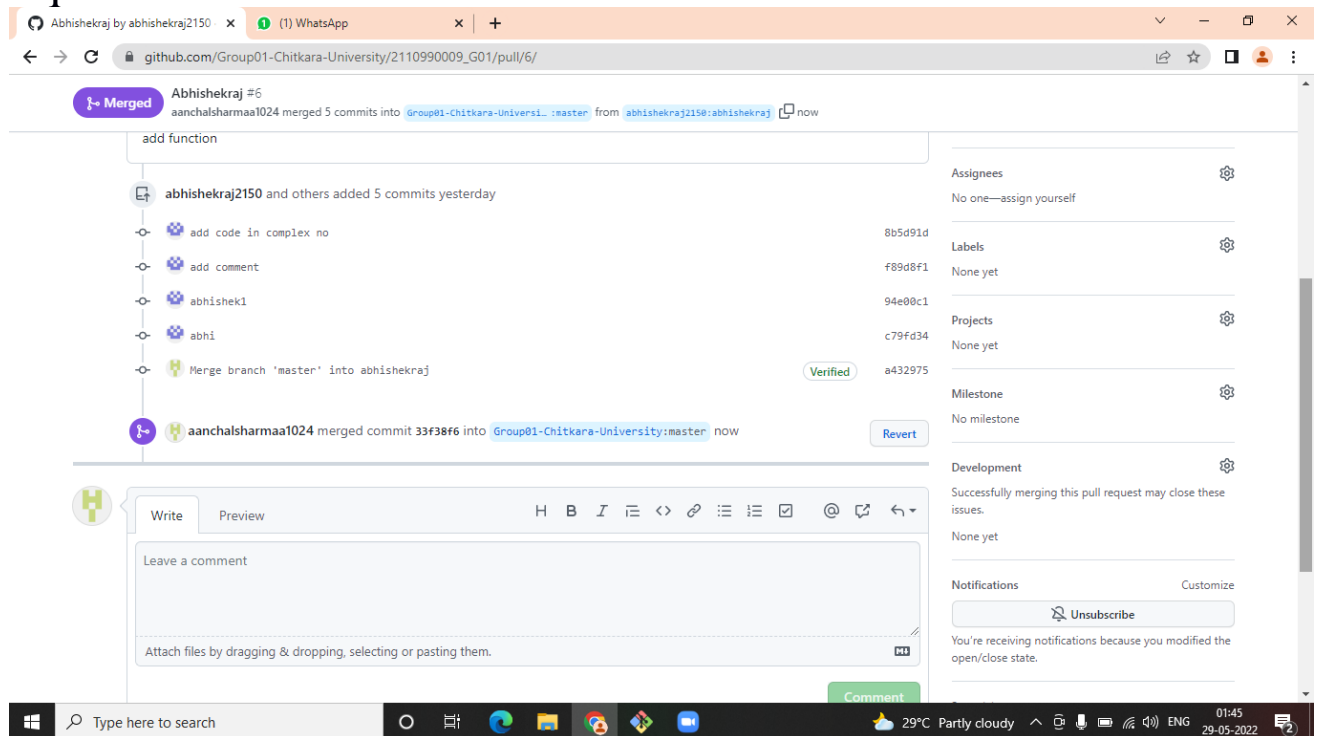
9. GitHub will detect any conflicts and ask you to enter a description of your pull request.

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.

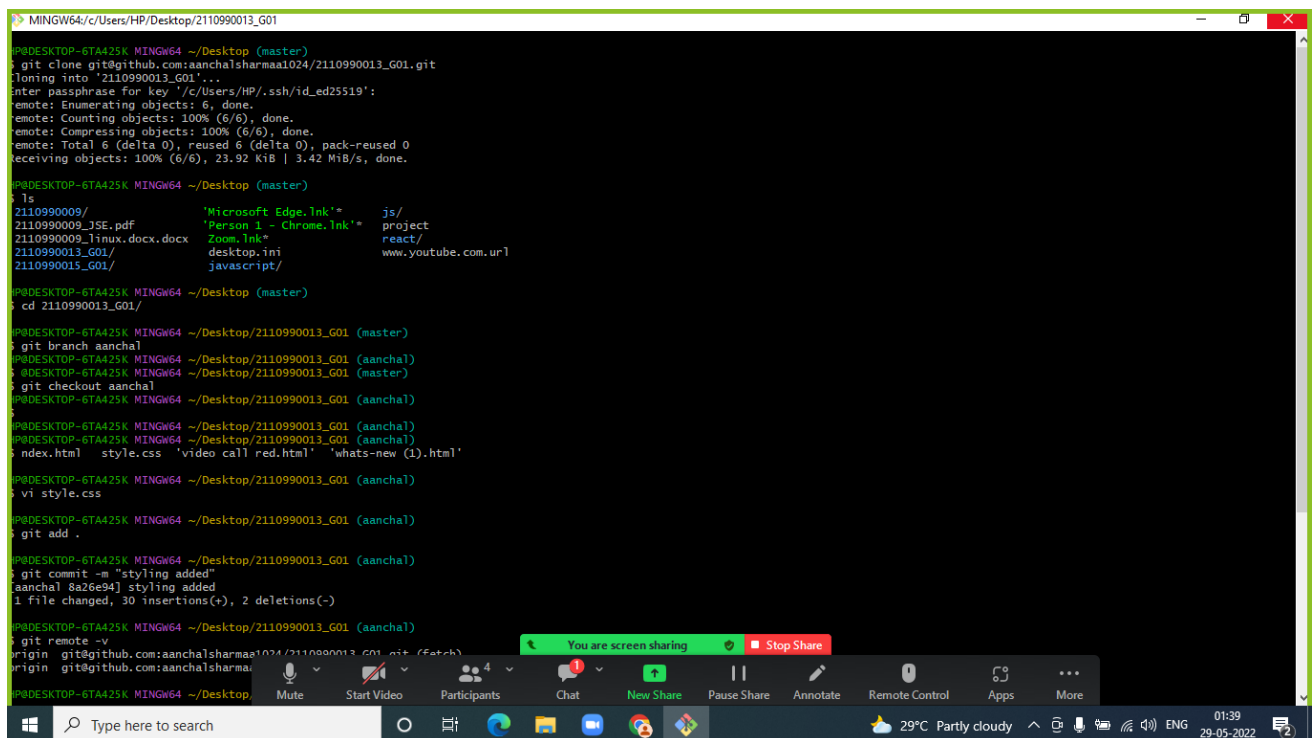
12. To close the pull request simply click on close pull request and add comment/ reason why you closed the pull request.

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



```
MINGW64/c/Users/HP/Desktop/2110990013_G01
IP0DESKTOP-6TA425K MINGW64 ~/Desktop (master)
$ git clone git@github.com:aanchalsharma1024/2110990013_G01.git
Cloning into '2110990013_G01'...
Enter passphrase for key '/c/Users/HP/.ssh/id_ed25519':
remote: Enumerating objects: 6, done.
remote: Counting objects: 100% (6/6), done.
remote: Compressing objects: 100% (6/6), done.
remote: Total 6 (delta 0), reused 6 (delta 0), pack-reused 0
Receiving objects: 100% (6/6), 23.92 KiB | 3.42 MiB/s, done.

IP0DESKTOP-6TA425K MINGW64 ~/Desktop (master)
$ ls
2110990009/      'Microsoft Edge.lnk'*  js/
2110990009_JSE.pdf  'Person 1 - Chrome.lnk'*  project
2110990009_linux.docx.docx  Zoom.lnk*  react/
2110990013_G01/    desktop.ini  www.youtube.com.url
2110990015_G01/    javascript/

IP0DESKTOP-6TA425K MINGW64 ~/Desktop (master)
$ cd 2110990013_G01/

IP0DESKTOP-6TA425K MINGW64 ~/Desktop/2110990013_G01 (master)
$ git branch aanchal
IP0DESKTOP-6TA425K MINGW64 ~/Desktop/2110990013_G01 (aanchal)
$ git checkout aanchal
IP0DESKTOP-6TA425K MINGW64 ~/Desktop/2110990013_G01 (aanchal)
IP0DESKTOP-6TA425K MINGW64 ~/Desktop/2110990013_G01 (aanchal)
$ index.html style.css 'video call red.html' 'whats-new (1).html'
IP0DESKTOP-6TA425K MINGW64 ~/Desktop/2110990013_G01 (aanchal)
$ vi style.css
IP0DESKTOP-6TA425K MINGW64 ~/Desktop/2110990013_G01 (aanchal)
$ git add .
IP0DESKTOP-6TA425K MINGW64 ~/Desktop/2110990013_G01 (aanchal)
$ git commit -m "styling added"
[aanchal 8a26e94] styling added
1 file changed, 30 insertions(+), 2 deletions(-)

IP0DESKTOP-6TA425K MINGW64 ~/Desktop/2110990013_G01 (aanchal)
$ git remote -v
origin git@github.com:aanchalsharma1024/2110990013_G01.git (fetch)
origin git@github.com:aanchalsharma1024/2110990013_G01.git (push)

IP0DESKTOP-6TA425K MINGW64 ~/Desktop (master)
$ git pull origin master
remote: Enumerating objects: 1, done.
remote: Compressing objects: 100% (1/1), done.
remote: Total 1 (delta 0), reused 1 (delta 0), pack-reused 0
Receiving objects: 100% (1/1), 1.00 KiB | 1.00 MiB/s, done.
Unpacking objects: 100% (1/1), 0.00 KiB | 0.00 MiB/s, done.
From git@github.com:aanchalsharma1024/2110990013_G01:
   master       8a26e94 styling added
Updating master (8a26e94..8a26e94): 1 new commit, 0 deletions.
```

The screenshot shows a terminal window with the above commands. A file explorer window is overlaid on the terminal, displaying the contents of the '2110990013_G01' directory. The file explorer shows a list of files and folders, including 'Microsoft Edge.lnk', 'Person 1 - Chrome.lnk', 'js/', 'project', 'Zoom.lnk', 'react/', 'www.youtube.com.url', 'desktop.ini', and 'javascript/'. The terminal window also shows a 'You are screen sharing' notification bar at the bottom.

```
MINGW64/c/Users/HP/Desktop/2110990013_G01
#0DESKTOP-6TA425K MINGW64 ~/Desktop (master)
$ cd 2110990013_G01/

#0DESKTOP-6TA425K MINGW64 ~/Desktop/2110990013_G01 (master)
$ git branch aanchal
#0DESKTOP-6TA425K MINGW64 ~/Desktop/2110990013_G01 (aanchal)
$ git checkout aanchal
#0DESKTOP-6TA425K MINGW64 ~/Desktop/2110990013_G01 (aanchal)
$ index.html style.css 'video call red.html' 'whats-new (1).html'
#0DESKTOP-6TA425K MINGW64 ~/Desktop/2110990013_G01 (aanchal)
$ vi style.css
#0DESKTOP-6TA425K MINGW64 ~/Desktop/2110990013_G01 (aanchal)
$ git add .
#0DESKTOP-6TA425K MINGW64 ~/Desktop/2110990013_G01 (aanchal)
$ git commit -m "styling added"
[aanchal 8a26e94] styling added
1 file changed, 30 insertions(+), 2 deletions(-)
#0DESKTOP-6TA425K MINGW64 ~/Desktop/2110990013_G01 (aanchal)
$ git remote
origin github.com:aanchalsharmaa1024/2110990013_G01.git (fetch)
origin git@github.com:aanchalsharmaa1024/2110990013_G01.git (push)
#0DESKTOP-6TA425K MINGW64 ~/Desktop/2110990013_G01 (aanchal)
$ git push -u origin aanchal
Enter passphrase for key '/c:/Users/HP/.ssh/id_rsa':
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 4 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 381 bytes | 381.00 KiB/s, done.
Total 3 (delta 2), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
remote:
remote: Create a pull request for 'aanchal' on GitHub by visiting:
remote: https://github.com/aanchalsharmaa1024/2110990013_G01/pull/new/aanchal
remote:
To github.com:aanchalsharmaa1024/2110990013_G01.git
   (new branch) aanchal -> aanchal
branch 'aanchal' set up to track 'origin/aanchal'
#0DESKTOP-6TA425K MINGW64 ~/Desktop
```

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 by Aarushi2021 · Pull Request

github.com/Group01-Chitkara-University/2110990009_G01/pull/4

print #4

Merged aanchalsharmaa1... merged 1 commit into Group01-Chitkara-University:master from Aarushi2021:aarushi681

Conversation 0 Commits 1 Checks 0 Files changed 1

Aarushi2021 commented 9 hours ago

No description provided.

print 89cb9ef

aanchalsharmaa1024 merged commit b93942d into Group01-Chitkara-University:master now

Revert

Write Preview

Leave a comment

Attach files by dragging & dropping, selecting or pasting them.

Reviewers: No reviews

Assignees: No one—assign yourself

Labels: None yet

Projects: None yet

Milestone: No milestone

Development

Successfully merging this pull request may close these

#2 TEAM MEMBERS'S PULL REQUEST

Resolve Conflicts · Pull Request #

github.com/Group01-Chitkara-University/2110990009_G01/pull/3/conflicts

Group01-Chitkara-University / 2110990009_G01 Public

Code Issues Pull requests 1 Actions Projects Wiki Security Insights Settings

subtraction #3

Resolving conflicts between Aaru112:Aaru1 and Group01-Chitkara-U...:master and committing changes → Aaru112:Aaru1

1 conflicting file

complexnumbers.cpp

```

12     this->imaginary=imaginary;
13 }
14 void minus(ComplexNumbers const &c2)
15 {
16     real=real-c2.real;
17     imaginary=imaginary-c2.imaginary;
18 }
19 void print()
20 {
21     cout<<real<<" + "<<"i"<<imaginary<<endl;
22 }
23 }
24
25 int main()
26 {
27     int real1, imaginary1, real2, imaginary2;
28     cin >> real1 >> imaginary1;
29     cin >> real2 >> imaginary2;
30     ComplexNumbers c1(real1, imaginary1);
31     ComplexNumbers c2(real2, imaginary2);
32     int choice;
33     cin >> choice;

```

subtraction by Aarul12 · Pull Request

github.com/Group01-Chitkara-University/2110990009_G01/pull/3

Group01-Chitkara-University / 2110990009_G01

<> Code Issues Pull requests Actions Projects Wiki Security Insights Settings

subtraction #3

Merged aanchalsharmaa1... merged 2 commits into Group01-Chitkara-University:master from Aarul12:Aarul 7 minutes ago

Conversation 1 Commits 2 Checks 0 Files changed 2 +5 -4

Aarul12 commented 10 hours ago

No description provided.

subtraction 388da79

MonitKapoor reviewed 10 hours ago

MonitKapoor left a comment • edited by aanchalsharmaa1024

this pointer can be used without explicit calling.
Can you do it?

this pointer is mainly used here because parameters in the constructor and name of data members are same.

Reviewers: MonitKapoor

Assignees: No one—assign yourself

Labels: None yet

Projects: None yet

Milestone: No milestone

#3 TEAM MEMBER'S PULL REQUEST

Online C++ Compiler - online x Lecture 8: Switch Statement x 2110990009_G01/complexnu x (3) WhatsApp x multiply function by Aarushi x

github.com/Group01-Chitkara-University/2110990009_G01/pull/5/files

Open multiply function #5 0 / 1 files viewed Review changes

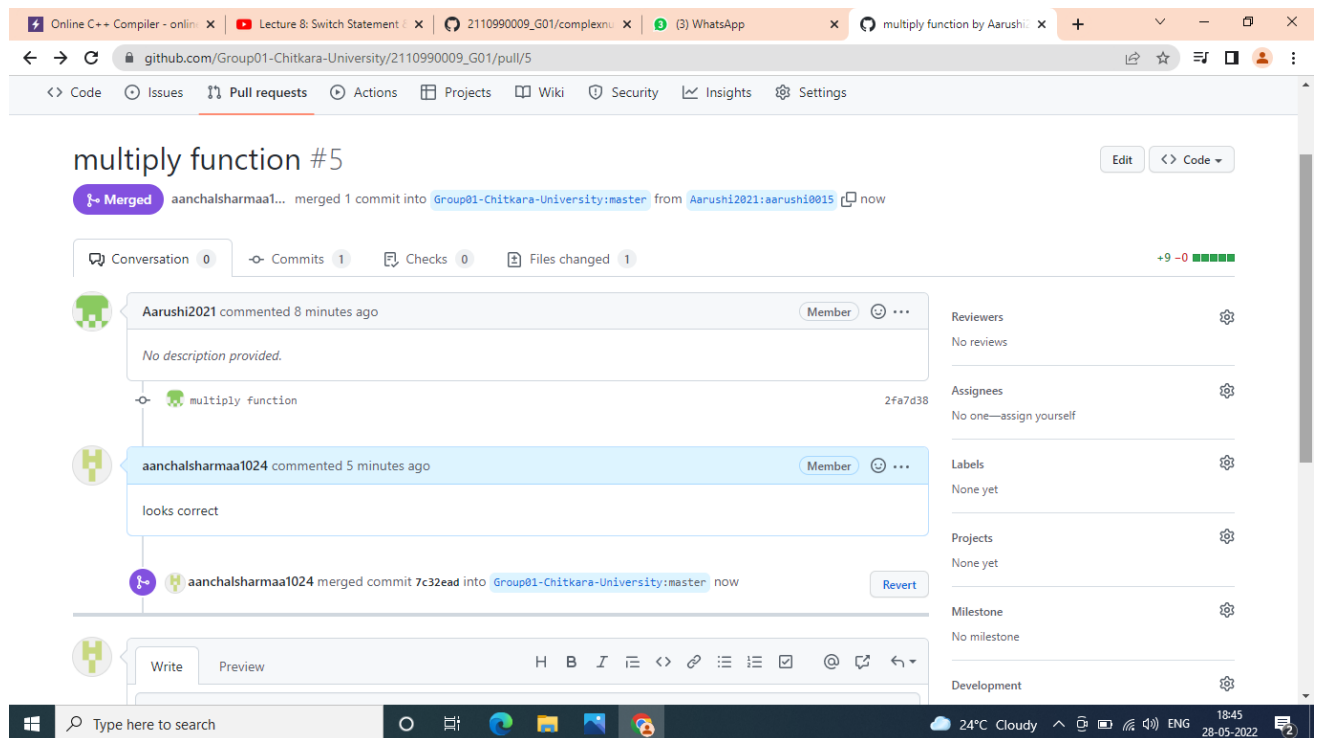
Changes from all commits File filter Conversations Jump to

complexnumbers.cpp

```

9 9      ComplexNumbers(int real,int imaginary)
10 10     {
11 11         this->real=real;
12 12         this->imaginary=imaginary;
13 13     }
14 14     ComplexNumbers(int real,int imaginary)
15 15     {
16 16         this->real=real;
17 17         this->imaginary=imaginary;
18 18     }
19 19     void print()
20 20     {
21 21         cout<<real<<" + "<<"i"<<imaginary<<endl;
22 22     }
23 + }
24 + void multiply(ComplexNumbers const &c2)
25 + {
26 +     int firsts=real*c2.real;
27 +     int outers=real*c2.imaginary;
28 +     int inners=imaginary*c2.real;
29 +     int lasts=(-1)*(imaginary*c2.imaginary);
30 +     real=firsts+lasts;
31 +     imaginary=outers+inners;
23 32 }
24 33 int main()
25 34 {
26 35     int real1, imaginary1, real2, imaginary2;

```

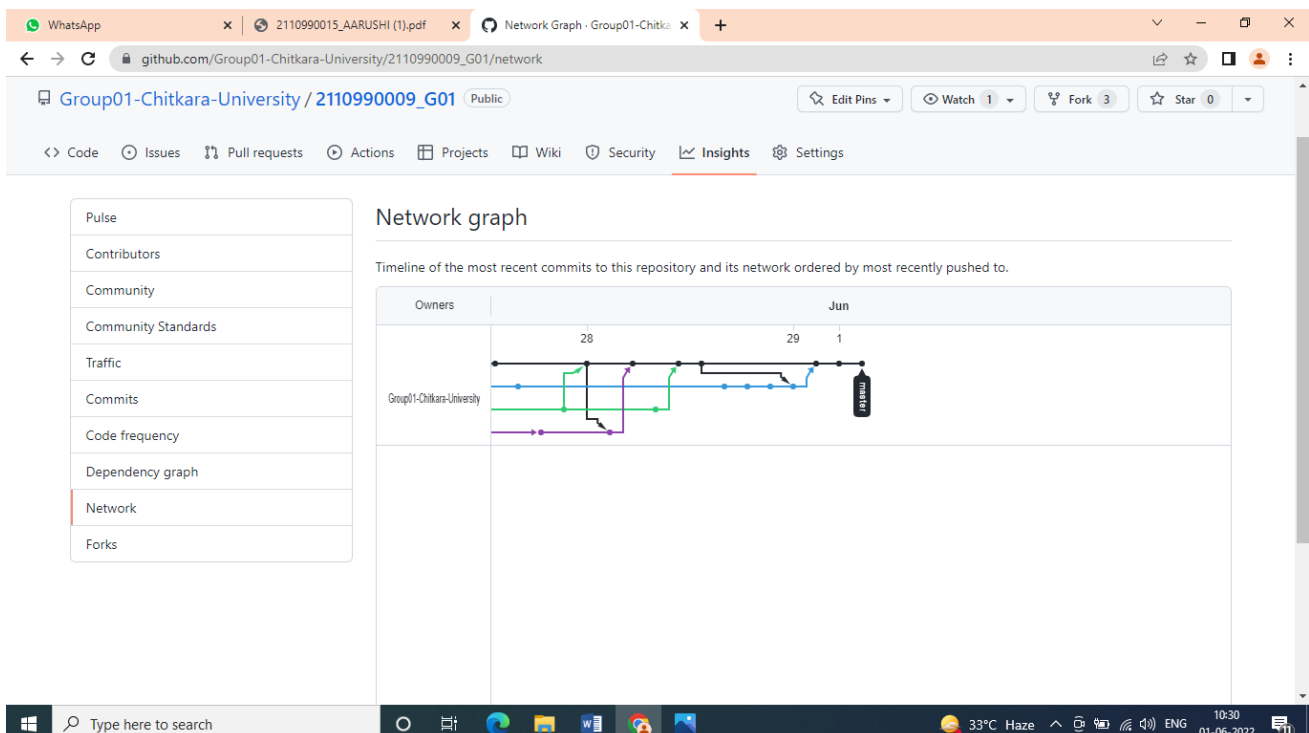
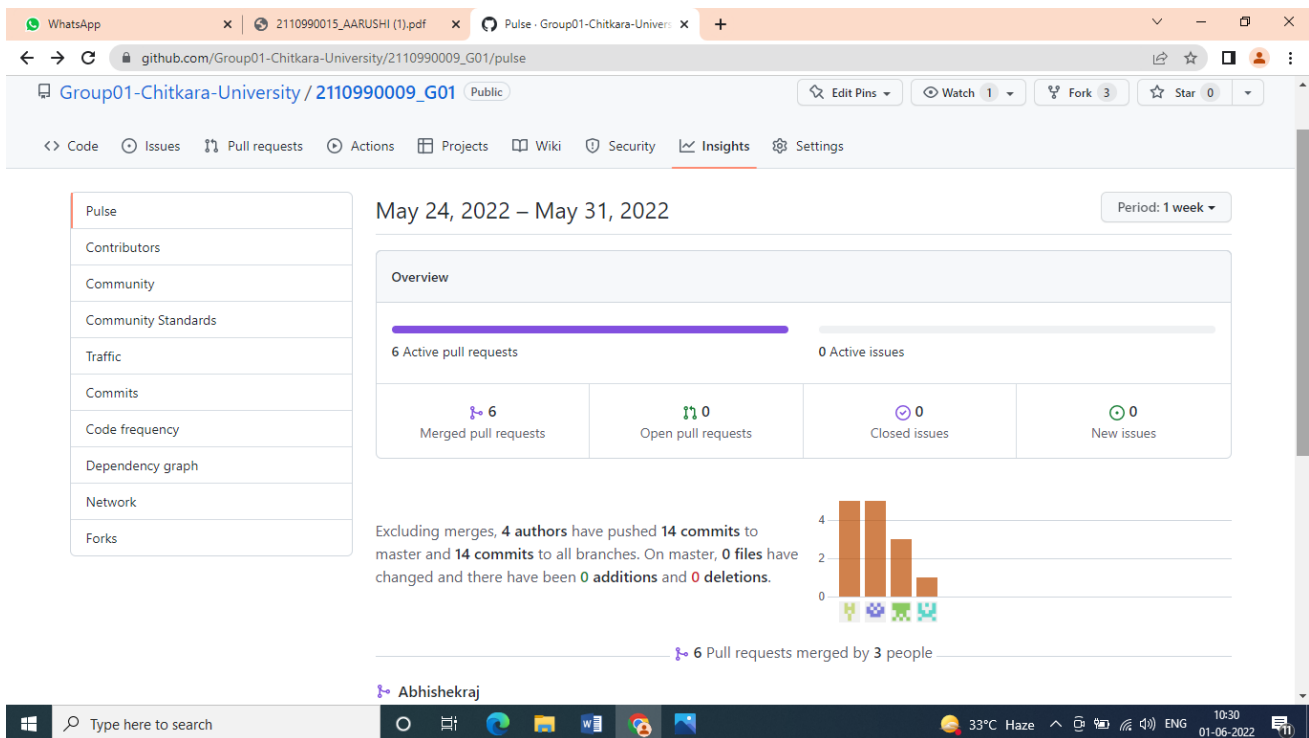



NETWORK GRAPHS

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 in 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

