

Task 1.1

Made By: Abhay Aggarwal

Roll No: 2110990034

Submitted To: Dr. Monit Kapoor

Subject Name: **Source Code Management**

Subject Code: **CS181**

Cluster: **Beta**

Department: **CSE**

INDEX

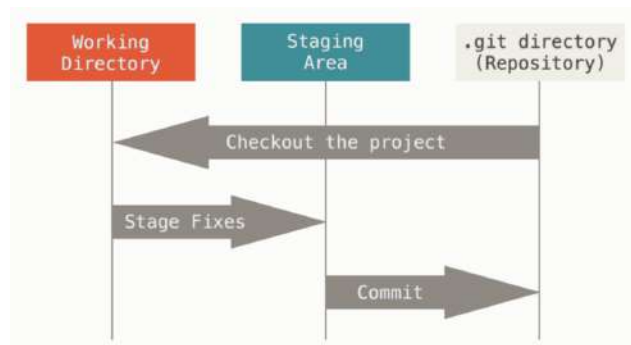
Sr. No.	Title
1.	Setting up of Git Client
2.	Setting up GitHub Account
3.	Generate Logs
4.	Create and visualize branches
5.	Git Life-Cycle description

Task 1.1 – Experiment No. 01

AIM: Setting up of Git Client

Theory:

- **Git:** Git is an example of Version Control Software. Unlike other VCSs every action on Git has its integrity as it is assigned a **SHA-1** hash. It is basically a 40-Character string that stores all the information about the file at that particular version
- **Advantages of Git:**
 - Git uses snapshots to save data
 - Every operation is local
 - It has integrity
 - Every file has in Git three states:
 - `modified` => File changed but not committed
 - `staged` => marked a modified file in its current version to go into the next commit snapshot
 - `committed` => the data is safely stored on the local database



Procedure:

- On Mac [running OS X or greater] git is pre installed.
- Check by running **git --version** in the terminal.

```
abhay ~ - ssh - 80x24
Last login: Fri Apr 8 22:04:46 on ttys000
(base) abhay@Abhays-MacBook-Air ~ % git --version
git version 2.35.1
(base) abhay@Abhays-MacBook-Air ~ %
```

Task 1.1 - Experiment No. 02

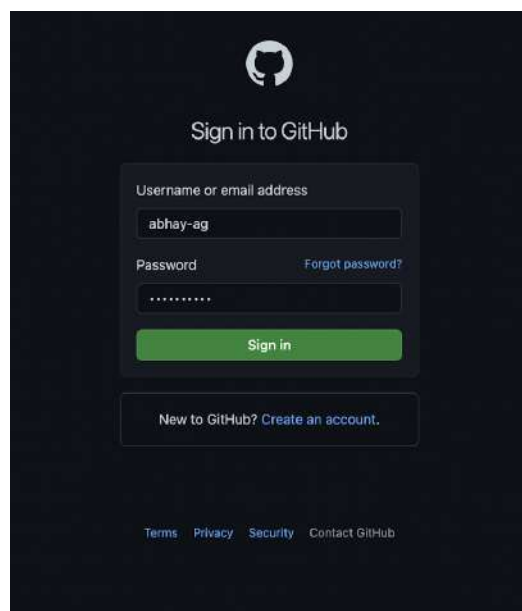
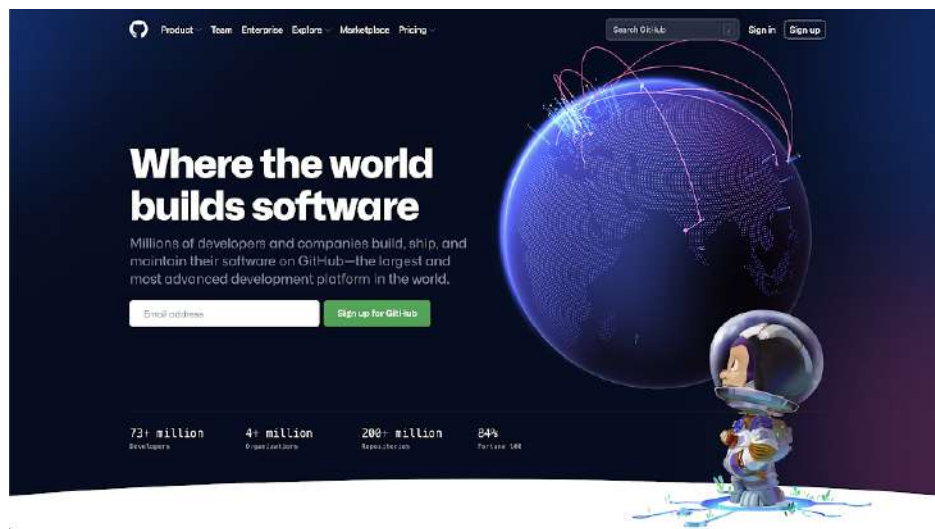
AIM : Setting up GitHub Account

Theory:

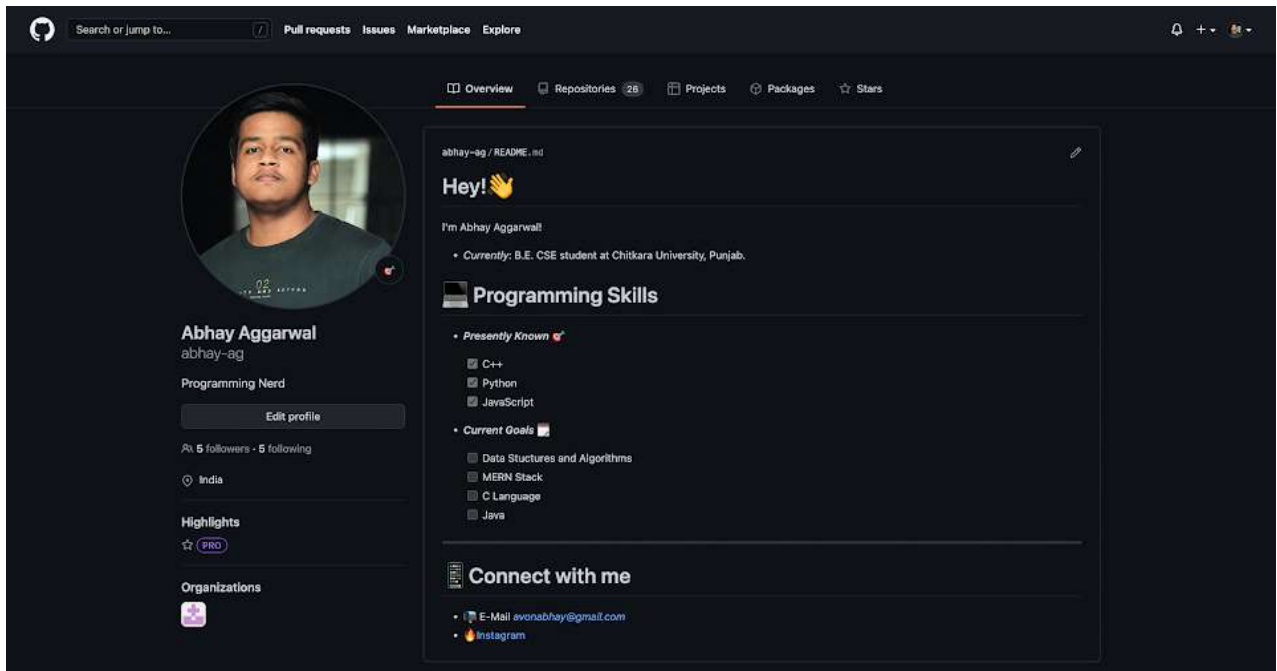
- **GitHub:** GitHub is the web implementation of Git. From storing data on local DataBases, now with the help of GitHub, users can push the code to the cloud based servers and perform various actions on it like: hosting, test running, deployment, etc.

Procedure:

1. Firstly search for GitHub on any search engine.
2. If already have an account then click on **Sign In** else click on **Sign Up**
3. Snapshots of the same are attached below
 - a. First Picture -> Landing Page
 - b. Second Picture -> Login Page



4. Interface of GitHub:



5. Commands to link GitHub with Git Bash:

For Username:

`git config --global user.name "<GitHub user.name>"`

For E-Mail

`git config --global user.email "<GitHub user.email>"`

6. For verification: Run the same commands without the string. It will return the username and email of the current user.

```
Last login: Fri Apr  8 22:16:53 on ttys000
(base) abhay@Abhays-MacBook-Air ~ % git config --global user.name
abhay-ag
(base) abhay@Abhays-MacBook-Air ~ % git config --global user.email
abhay0034.be21@chitkara.edu.in
(base) abhay@Abhays-MacBook-Air ~ %
```

Task 1.1 - Experiment No. 03

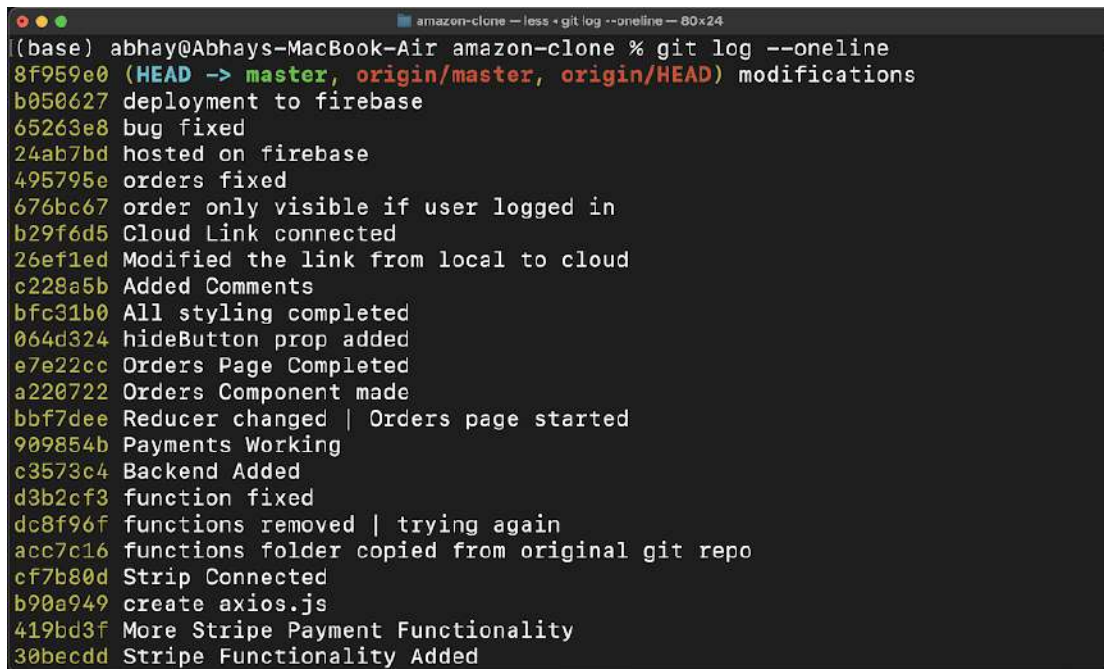
AIM: Generate Logs

Theory:

- **Logs:** Logs are history of the changes made in the file at each version. It stores the SHA-1 Checksum, username, branch name, date and time, etc.

Procedure:

1. Initialize a repo on your local machine.
2. Do some changes and commit at each step.
3. Type **git log** in the terminal



```
amazon-clone -- less + git log --oneline -- 80x24
(base) abhay@Abhays-MacBook-Air amazon-clone % git log --oneline
8f959e0 (HEAD -> master, origin/master, origin/HEAD) modifications
b050627 deployment to firebase
65263e8 bug fixed
24ab7bd hosted on firebase
495795e orders fixed
676bc67 order only visible if user logged in
b29f6d5 Cloud Link connected
26ef1ed Modified the link from local to cloud
c228a5b Added Comments
bfc31b0 All styling completed
064d324 hideButton prop added
e7e22cc Orders Page Completed
a220722 Orders Component made
bbf7dee Reducer changed | Orders page started
909854b Payments Working
c3573c4 Backend Added
d3b2cf3 function fixed
dc8f96f functions removed | trying again
acc7c16 functions folder copied from original git repo
cf7b80d Strip Connected
b90a949 create axios.js
419bd3f More Stripe Payment Functionality
30becdd Stripe Functionality Added
```

Task 1.1 - Experiment No. 04

AIM: Create and Visualize branches.

Theory:

- By default, the branch that is created automatically is called **master** or **main**.

```
((base) abhay@Abhays-MacBook-Air amazon-clone % git branch
* master
```

Procedure:

1. For creating a new branch -> git branch <branch name>
2. For checking all the branches -> git branch

```
((base) abhay@Abhays-MacBook-Air amazon-clone % git branch feature
((base) abhay@Abhays-MacBook-Air amazon-clone % git branch
feature
* master
```

3. To change the present working branch -> git checkout <branch name>

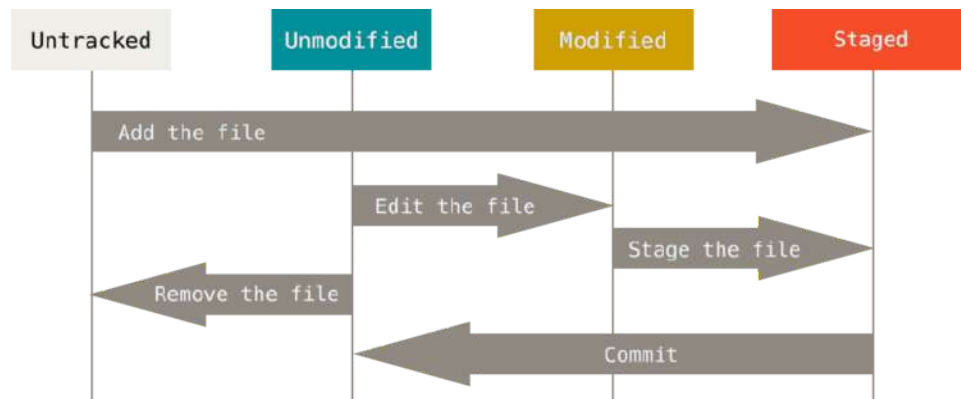
```
((base) abhay@Abhays-MacBook-Air amazon-clone % git checkout feature
Switched to branch 'feature'
((base) abhay@Abhays-MacBook-Air amazon-clone % git branch
* feature
master
```

Task 1.1 - Experiment No. 05

AIM: Git life-cycle description

Theory:

- The lifecycle of the status of our files:
 - untracked: newly added files after the last snapshot of the data
 - unmodified: when an untracked file is staged in the latest snapshot it returns to the unmodified state.
 - modified: when an unmodified file is edited it moves to the modified state and has to be staged again
 - untracked: if the file is removed after the commit



Procedure:

1. Firstly make a directory on your local machine.

```
(base) abhay@Abhays-MacBook-Air ~ % cd Desktop
(base) abhay@Abhays-MacBook-Air Desktop % mkdir SCM_p
(base) abhay@Abhays-MacBook-Air Desktop % cd SCM_p
(base) abhay@Abhays-MacBook-Air SCM_p % vi first.cpp
(base) abhay@Abhays-MacBook-Air SCM_p % git status
fatal: not a git repository (or any of the parent directories): .git
(base) abhay@Abhays-MacBook-Air SCM_p %
```

2. Initialize a repository in the same directory.

```
(base) abhay@Abhays-MacBook-Air SCM_p % git init
hint: Using 'master' as the name for the initial branch. This default branch name
hint: is subject to change. To configure the initial branch name to use in all
hint: of your new repositories, which will suppress this warning, call:
hint:
hint:   git config --global init.defaultBranch <name>
hint:
hint: Names commonly chosen instead of 'master' are 'main', 'trunk' and
hint: 'development'. The just-created branch can be renamed via this command:
hint:
hint:   git branch -m <name>
```


3. Check the status of all the files.

```
(base) abhay@Abhays-MacBook-Air SCM_p % git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    first.cpp

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

4. Here 'first.cpp' is an unstaged file. Add the file, then check the status.

```
(base) abhay@Abhays-MacBook-Air SCM_p % git add .
(base) abhay@Abhays-MacBook-Air SCM_p % git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
    new file:   first.cpp
```

5. 'first.cpp' is now staged and ready to commit. Commit and then check the status.

```
(base) abhay@Abhays-MacBook-Air SCM_p % git commit -m "First Commit"
[master (root-commit) 50960bc] First Commit
 1 file changed, 6 insertions(+)
 create mode 100644 first.cpp
(base) abhay@Abhays-MacBook-Air SCM_p % git status
On branch master
nothing to commit, working tree clean
```

6. The file is now committed and returns to unmodified state. Now to push it to GitHub we use. Repo Link [https://github.com/abhay-ag/file_repo.git]

```
(base) abhay@Abhays-MacBook-Air SCM_p % git push -u origin master
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Delta compression using up to 8 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 300 bytes | 300.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/abhay-ag/file_repo.git
 * [new branch]      master -> master
branch 'master' set up to track 'origin/master'.
```

Task 1.2

Made By: Abhay Aggarwal

Roll No: 2110990034

Submitted To: Dr. Monit Kapoor

Subject Name: Source Code Management

Subject Code: CS181

Cluster: Beta

Department: CSE

INDEX

Sr. No.	Title
1.	Add collaborators on a GitHub repo
2.	Fork and Commit
3.	Merge and resolve conflicts created due to own activity and collaborators activity
4.	Reset and Revert

Task 1.2 - Experiment No. 01

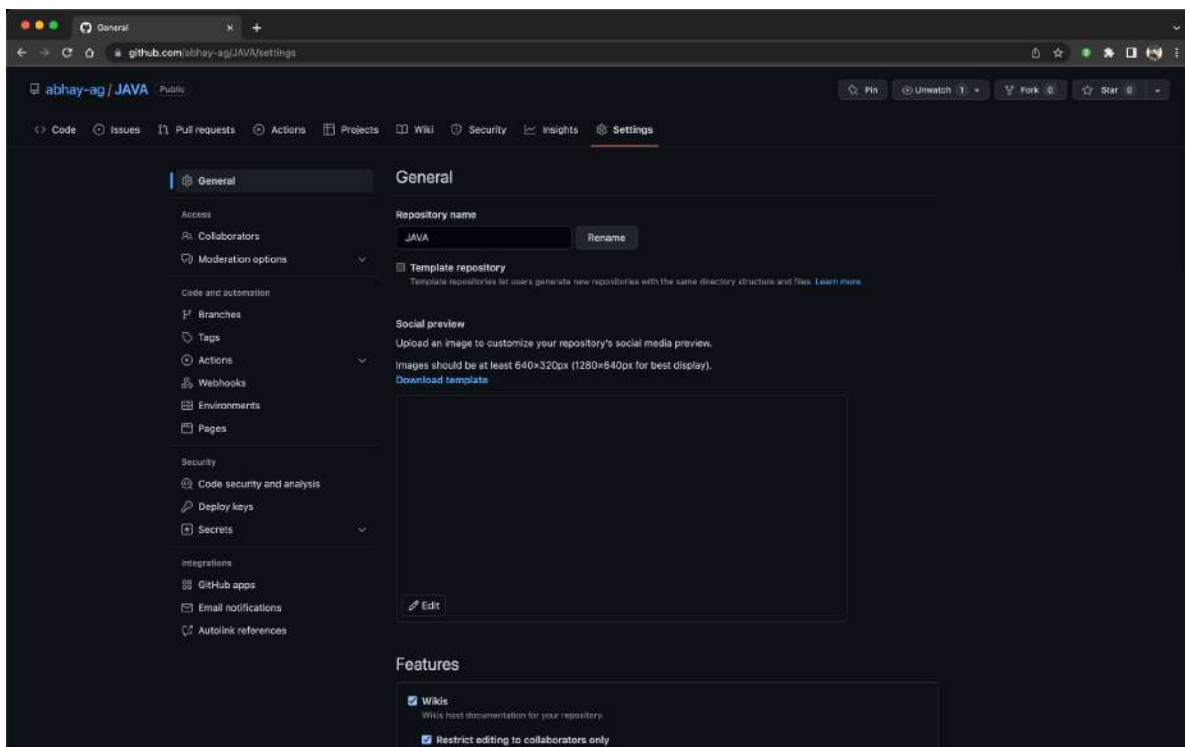
AIM: Add collaborators on a GitHub repo

Theory:

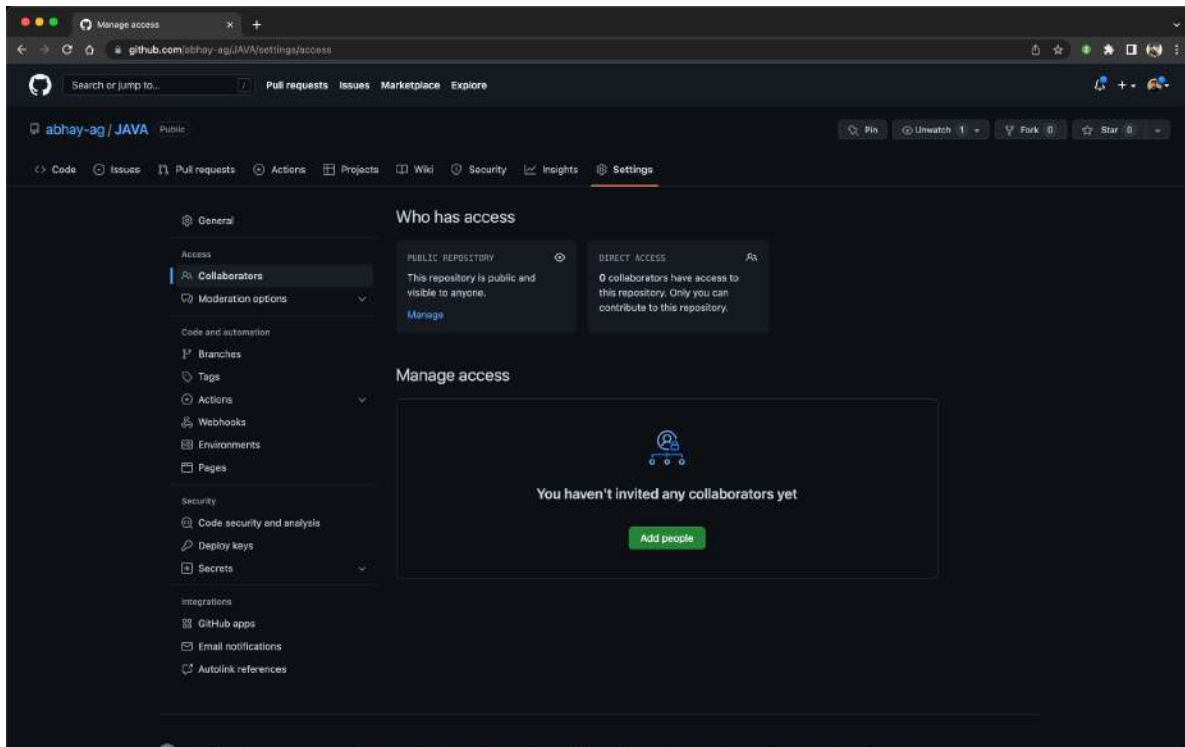
- **Collaborators:** People/developers who can be trusted with the code put up in the repo can be added as a collaborator in the repo. They can either be of the same organization or be at a remote location. They have access to various Git operations on the repo like:
 - Create, merge and close pull requests.
 - Make changes in the code in the repo.
 - Delete any comments on commits, pull requests, and issues in the repository.
 - Remove themselves as a collaborator from the repository.
- **NOTE:** The invitation sent to the collaborator if not accepted will expire in seven (7) days.

Procedure:

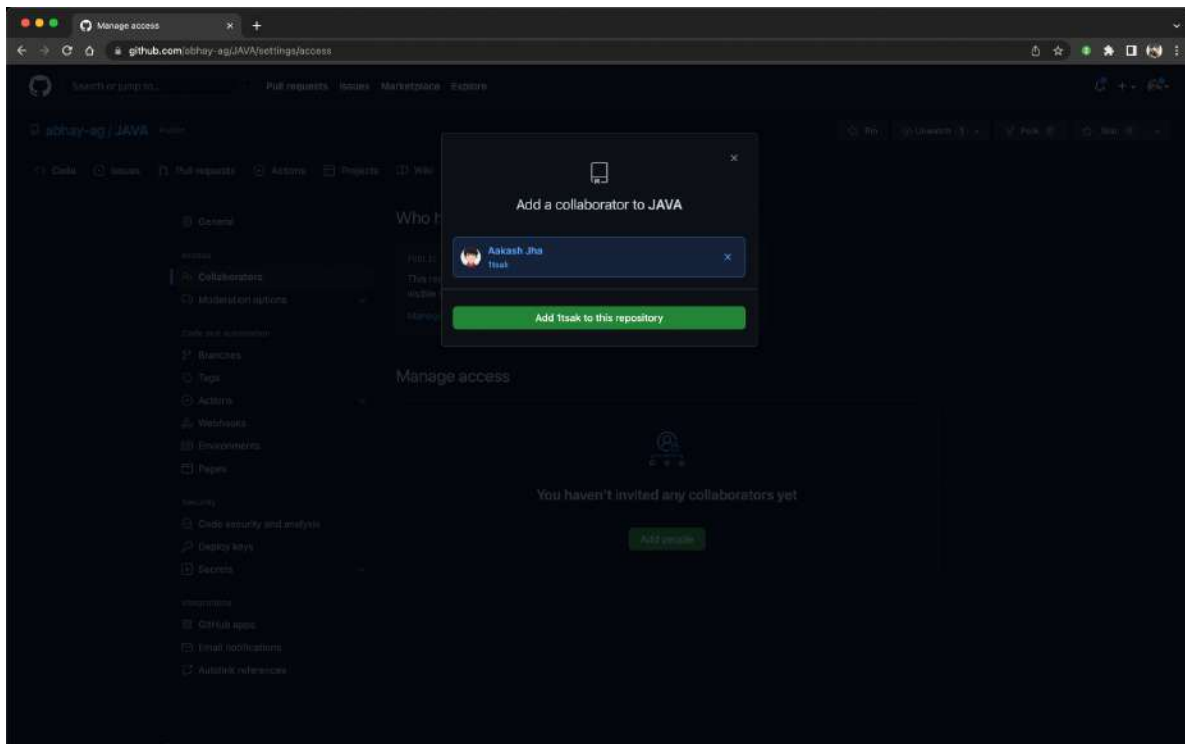
- To add a collaborator into a repo go to the setting tab of the particular repo.



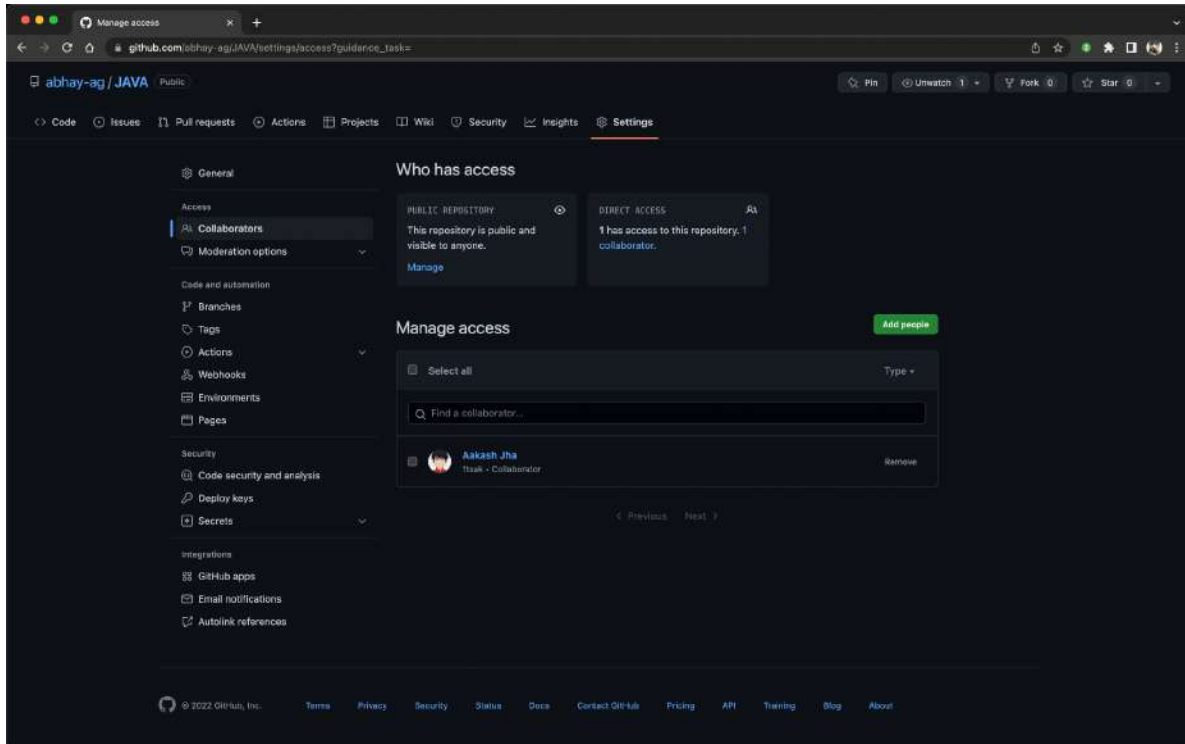
- Navigate to the **COLLABORATORS** section of the navigation panel.



- Click on the add people button and type in the username of the person you want to add as a collaborator.



- Click the button to send an invite to the specified person as a collaborator. When the user specified accepts the invite then he/she is added as a collaborator to the repo.



- The user is now added as a collaborator and can now access the specified functionality of GitHub.

Task 1.2 - Experiment No. 02

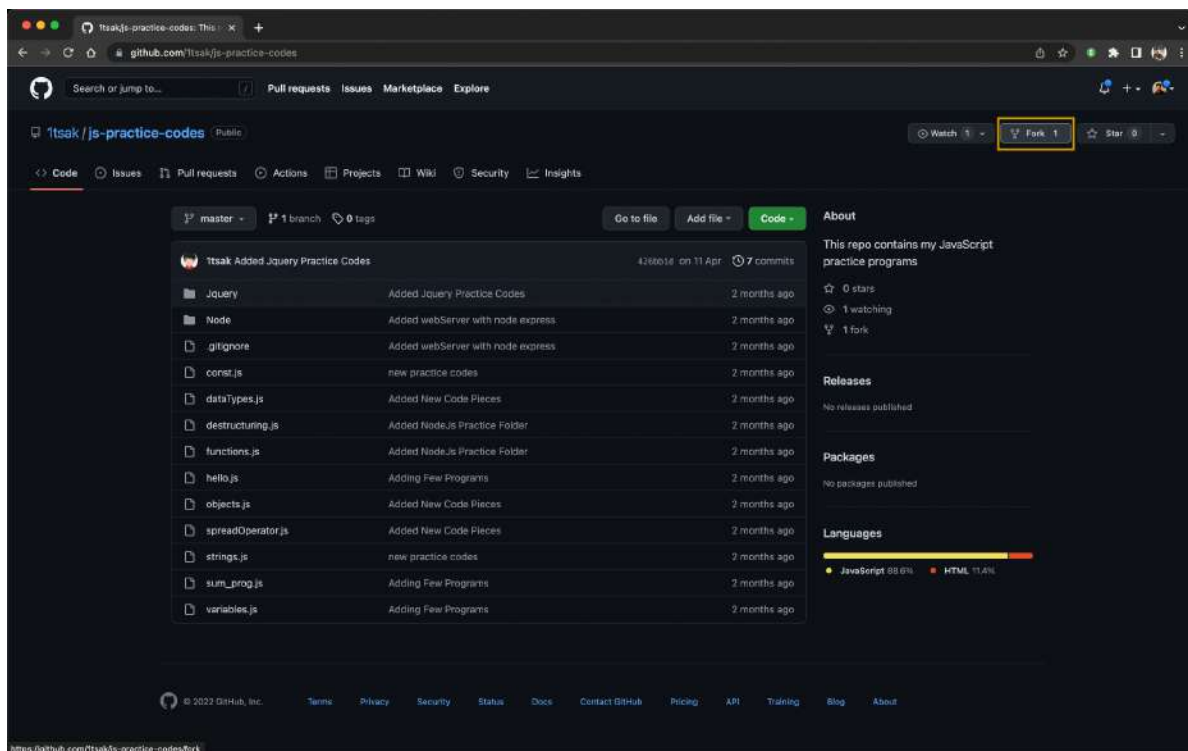
AIM : Fork and Commit

Theory:

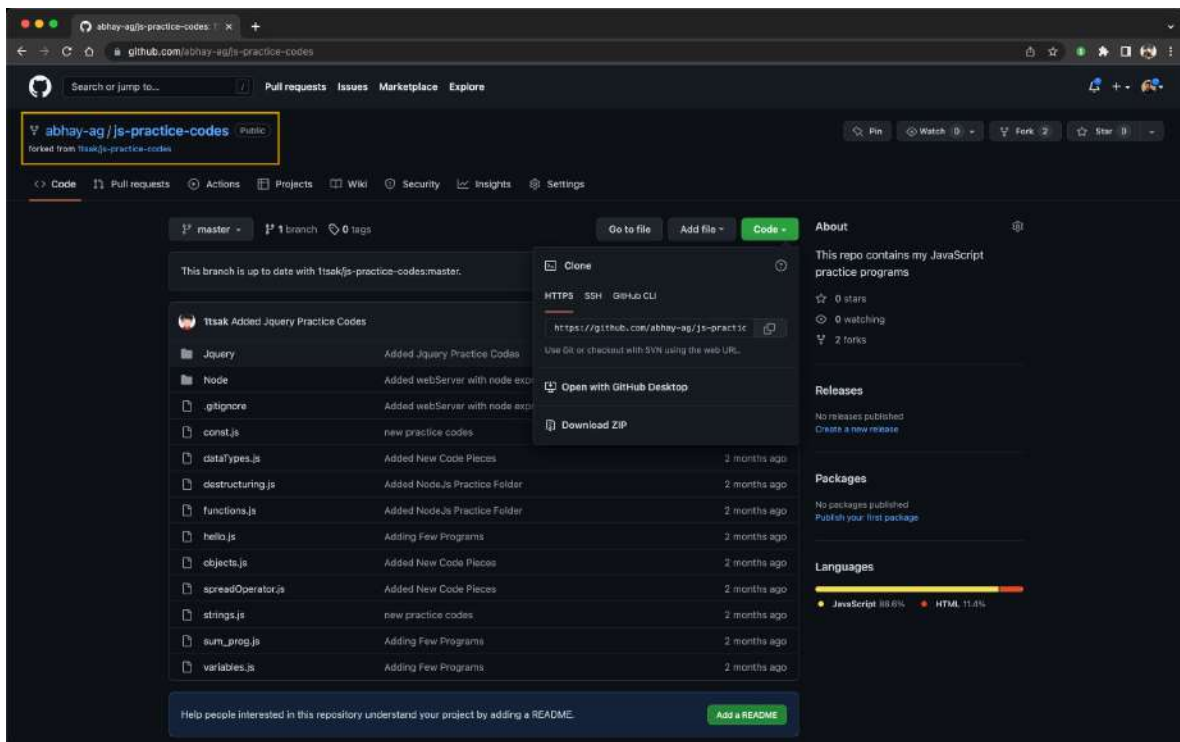
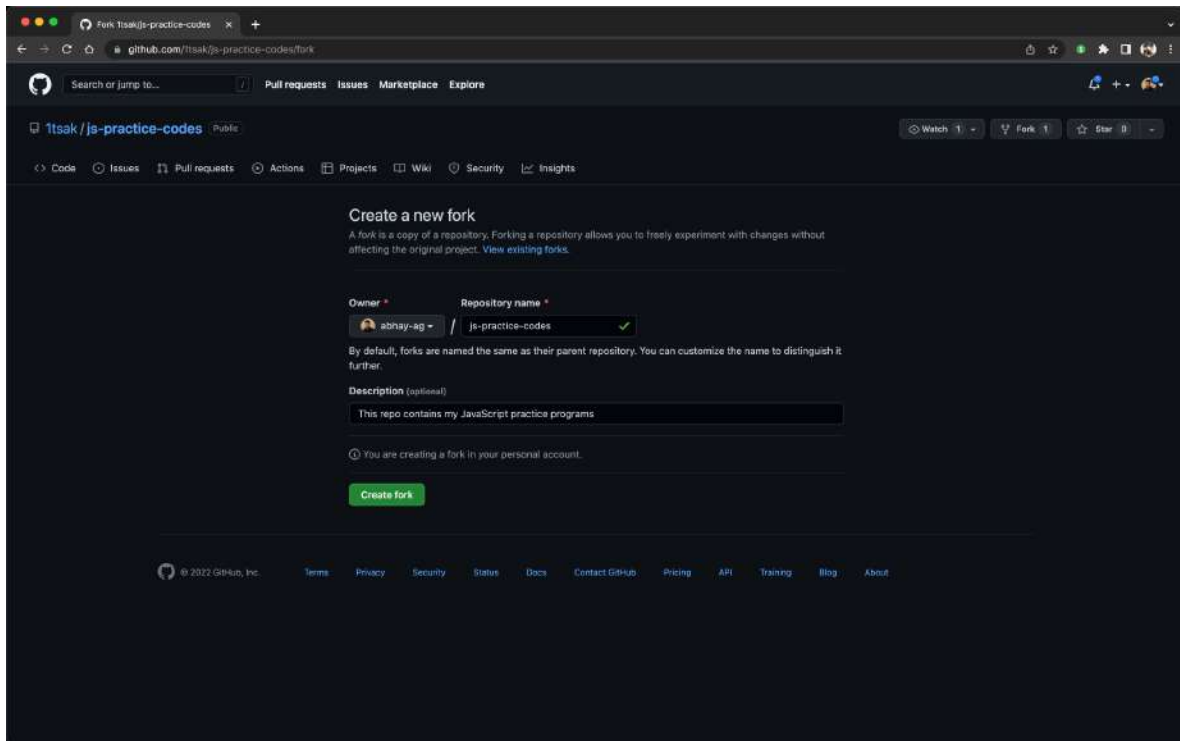
- **Fork:** Forking a repo creates a copy of a repository in the GitHub account of the user who forks the repo. Now the ownership of the repository shifts to the user instead of the person from whom the repo is forked, and the user can freely commit, set up branches, open pull requests etc. on the repo.
- **Why fork?**
 - Users can make changes to the code after finding issues and if the test cases satisfy the requirements then, can open a pull request to merge the modified code in the original repo.
 - Without forking a user can't push to the original repo on GitHub [unless added as a collaborator]. To do this; one has to fork open pull requests and wait for them to be merged into the code.

Procedure:

1. Click on the fork button in the repository you want to fork.



2. After forking [if part of organizations select the owner] the copy of the repo will be created in the user's account.



- The fork is now created in the user account and can now be cloned onto the local machine and commits can now be added without disturbing the code in the original code.

```
Last login: Mon May 30 10:57:36 on console
(base) abhay@Abhays-MacBook-Air ~ % cd Desktop
(base) abhay@Abhays-MacBook-Air Desktop % git clone https://github.com/abhay-ag/js-practice-codes.git
Cloning into 'js-practice-codes'...
remote: Enumerating objects: 41, done.
remote: Counting objects: 100% (41/41), done.
remote: Compressing objects: 100% (30/30), done.
remote: Total 41 (delta 9), reused 37 (delta 5), pack-reused 0
Receiving objects: 100% (41/41), 42.40 KiB | 487.00 KiB/s, done.
Resolving deltas: 100% (9/9), done.
(base) abhay@Abhays-MacBook-Air Desktop %
```

```
(base) abhay@Abhays-MacBook-Air js-practice-codes % git reset --hard
HEAD is now at 8a93763 Made Changes to the comments on the code || spelling fixes
(base) abhay@Abhays-MacBook-Air js-practice-codes % git log --oneline
8a93763 (HEAD -> master) Made Changes to the comments on the code || spelling fixes
426bb1d (origin/master, origin/HEAD) Added JQuery Practice Codes
07ae0eb Added webServer with node express
02076a6 Added NodeJs Practice Folder
ad4547a added destructuring concept
f9c08a0 Added New Code Pieces
fad8a84 new practice codes
e3de23b Adding Few Programs
(base) abhay@Abhays-MacBook-Air js-practice-codes % git reset --hard 426bb1d
HEAD is now at 426bb1d Added JQuery Practice Codes
(base) abhay@Abhays-MacBook-Air js-practice-codes %
```

```
(base) abhay@Abhays-MacBook-Air js-practice-codes % git status
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:   const.js

no changes added to commit (use "git add" and/or "git commit -a")
(base) abhay@Abhays-MacBook-Air js-practice-codes % git commit -m "Spelling fixes || added description"
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:   const.js

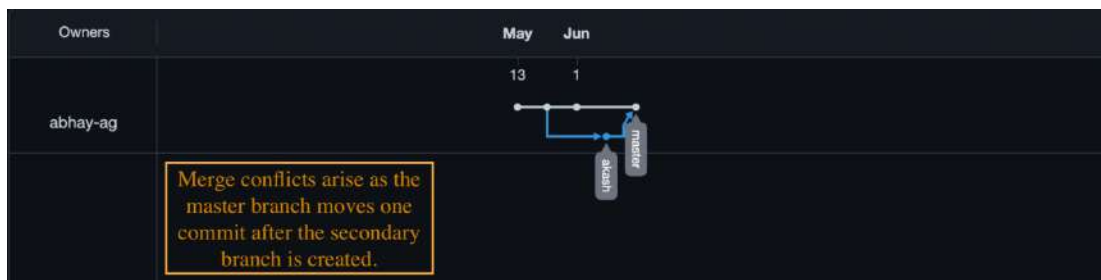
no changes added to commit (use "git add" and/or "git commit -a")
(base) abhay@Abhays-MacBook-Air js-practice-codes % git add .
(base) abhay@Abhays-MacBook-Air js-practice-codes % git commit -m "Spelling fixes || added description"
[master 1629d94] Spelling fixes || added description
 1 file changed, 2 insertions(+), 2 deletions(-)
(base) abhay@Abhays-MacBook-Air js-practice-codes % git push -u origin master
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 8 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 363 bytes | 363.00 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.
To https://github.com/abhay-ag/js-practice-codes.git
  426bb1d..1629d94 master -> master
branch 'master' set up to track 'origin/master'.
(base) abhay@Abhays-MacBook-Air js-practice-codes %
```

Task 1.2 - Experiment No. 03

AIM: Merge and resolve conflicts created due to own activity and collaborators activity

Theory:

- **Merge Conflicts:** Merge conflicts can arise due to our own activity or collaborators activity. They arise when on the same line in two different branches there is a different piece of code and both are committed [the network graph is attached below]. If branch on which another branch has to be merged doesn't have any commits after the other branch is created then there are no merge conflicts and the branches merge without any conflicts [network graph also attached below]



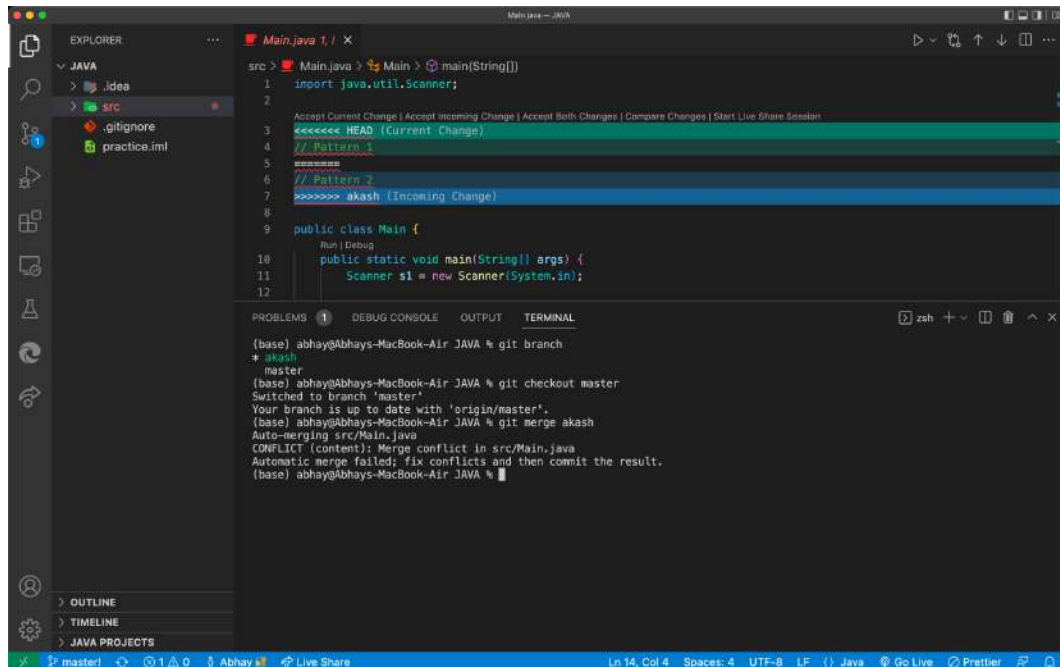
Procedure:

1. Clone a repo onto your machine and create a branch. Make some changes on the same line in the branch you created as well as the default branch.

```
(base) abhay@Abhays-MacBook-Air js-practice-codes % git commit -a -m "Changes on master branch"
[master 01f3037] Changes on master branch
1 file changed, 1 insertion(+), 1 deletion(-)
(base) abhay@Abhays-MacBook-Air js-practice-codes % git checkout abhay
Switched to branch 'abhay'
(base) abhay@Abhays-MacBook-Air js-practice-codes % git commit -a -m "Changes on abhay branch"
[abhay a99fef8] Changes on abhay branch
1 file changed, 1 insertion(+), 1 deletion(-)
(base) abhay@Abhays-MacBook-Air js-practice-codes % git checkout master
Switched to branch 'master'
Your branch is ahead of 'origin/master' by 2 commits.
(use "git push" to publish your local commits)
(base) abhay@Abhays-MacBook-Air js-practice-codes % git merge abhay
Auto-merging const.js
CONFLICT (content): Merge conflict in const.js
Automatic merge failed; fix conflicts and then commit the result.
(base) abhay@Abhays-MacBook-Air js-practice-codes %
```

- This will create merge conflicts as shown above, and will tell us to fix the conflicts and then try merging. If using git through vs code then the errors will be shown and can be resolved there only but I'll demonstrate using **vim** editor.

THIS DEMONSTRATES THE ERRORS IN VS CODE [made by collaborator activity]

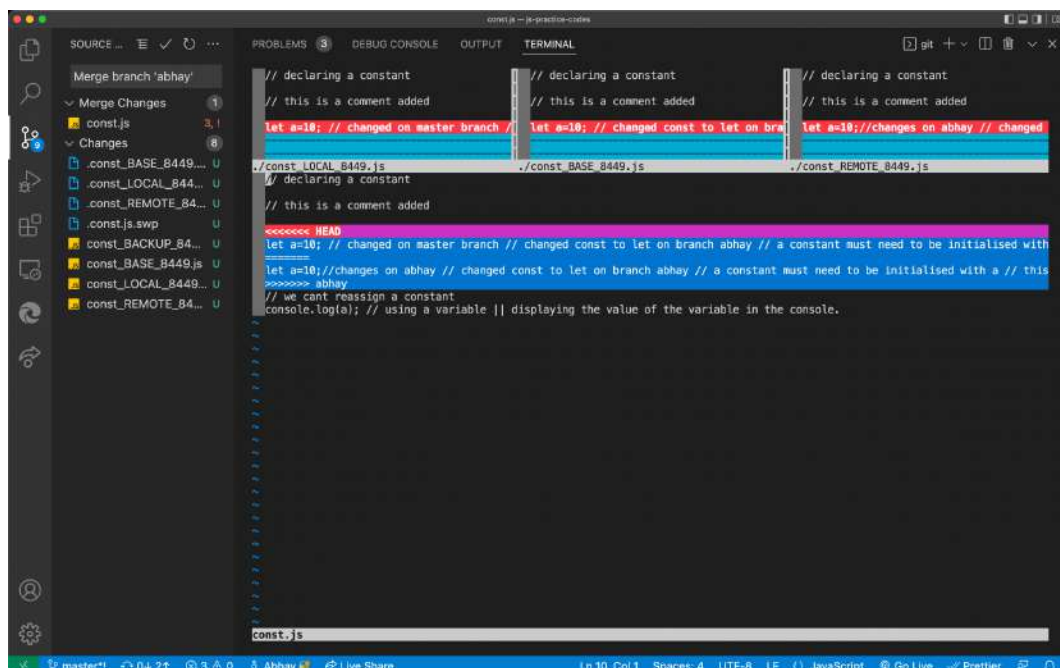


```
src > Main.java > Main > main(String[])
1  import java.util.Scanner;
2
3  Accept Current Change | Accept Incoming Change | Accept Both Changes | Compare Changes | Start Live Share Session
4  <<<<<< HEAD (Current Change)
5  // Pattern 1
6  // Pattern 2
7  >>>>>> akash (Incoming Change)
8
9  public class Main {
10     public static void main(String[] args) {
11         Scanner s1 = new Scanner(System.in);
12     }
13 }
```

PROBLEMS 1 DEBUG CONSOLE OUTPUT TERMINAL

```
(base) abhay@Abhays-MacBook-Air JAVA % git branch
* akash
master
(base) abhay@Abhays-MacBook-Air JAVA % git checkout master
Switched to branch 'master'
Your branch is up to date with 'origin/master'.
(base) abhay@Abhays-MacBook-Air JAVA % git merge akash
Auto-merging src/Main.java
CONFLICT (content): Merge conflict in src/Main.java
Automatic merge failed; fix conflicts and then commit the result.
(base) abhay@Abhays-MacBook-Air JAVA %
```

THIS DEMONSTRATES THE ERROR IN VIM EDITOR [made by my activity]



```
SOURCE ... Merge Changes 1
const.js 3, 1
Changes 8
./const_BASE_8449... U
./const_LOCAL_844... U
./const_REMOTE_84... U
./const.js.swp U
const_BACKUP_84... U
const_BASE_8449.js U
const_LOCAL_8449... U
const_REMOTE_84... U
```

```
// declaring a constant
// this is a comment added
let a=10; // changed on master branch
// declaring a constant
// this is a comment added
let a=10; // changed const to let on brw
// declaring a constant
// this is a comment added
let a=10; // changes on abhay // changed
// declaring a constant
// this is a comment added
let a=10; // changes on abhay // changed const to let on branch abhay // a constant must need to be initialised with
// we cant reassign a constant
console.log(a); // using a variable || displaying the value of the variable in the console.
```

PROBLEMS 3 DEBUG CONSOLE OUTPUT TERMINAL

```
const.js
```

3. Here look for the changes you want to keep or discard and then after pressing **esc** key type **:wq** in the main window. And you can close the other small windows by pressing **:q**.
4. The changes will be made and the branches will be merged successfully. The code can then be pushed onto GitHub.

```
(base) abhay@Abhays-MacBook-Air js-practice-codes % git checkout master
Switched to branch 'master'
Your branch is ahead of 'origin/master' by 2 commits.
  (use "git push" to publish your local commits)
(base) abhay@Abhays-MacBook-Air js-practice-codes % git merge abhay
Auto-merging const.js
CONFLICT (content): Merge conflict in const.js
Automatic merge failed; fix conflicts and then commit the result.
(base) abhay@Abhays-MacBook-Air js-practice-codes % 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:
tortoisemerge emerge vimdiff nvimdiff
Merging:
const.js

Normal merge conflict for 'const.js':
  {local}: modified file
  {remote}: modified file
Hit return to start merge resolution tool (vimdiff):
4 files to edit
(base) abhay@Abhays-MacBook-Air js-practice-codes % git add .
(base) abhay@Abhays-MacBook-Air js-practice-codes % git commit -m "Merge conflicts resolved"
[master e654eb8] Merge conflicts resolved
(base) abhay@Abhays-MacBook-Air js-practice-codes % git push -u origin master
Enumerating objects: 14, done.
Counting objects: 100% (14/14), done.
Delta compression using up to 8 threads
Compressing objects: 100% (12/12), done.
Writing objects: 100% (12/12), 1.24 KiB | 1.24 MiB/s, done.
Total 12 (delta 8), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (8/8), completed with 1 local object.
To https://github.com/abhay-ag/js-practice-codes.git
  1629d94..e654eb8  master -> master
branch 'master' set up to track 'origin/master'.
(base) abhay@Abhays-MacBook-Air js-practice-codes %
```

Task 1.2 - Experiment No. 04

AIM: Reset and Revert

Theory:

- While working on code with Git we might want to undo some changes, delete some commits locally as well as remotely, and remove some buggy code completely, etc. this is where reset and revert come in handy. Reset and revert both can be used to drop commits locally as well as remotely.
- **DIFFERENCE between RESET and REVERT**
 - Reset drops the commits up to the specified commit-id and also removes the changes made to the file(s) made during those commits. Of three types:
 - **--hard:** changes the state of the code completely to the commit specified.
 - **--soft:** changes the state of the code to the commit specified but also keeps the changes, staging area also remains the same.
 - **--mixed:** changes the state of the code to the commit specified but keeps the changes, and the staging area is also changed.
 - Revert undos the changes in the specific commit in a public branch and adds a new commit after undoing the commit with a message and the commit-id. If the commit after the specific commit is dependent on the commit to be reverted then the revert is not possible.

Procedure:

RESET

1. Make changes to a file. Stage and commit the file.
2. Now use the command **git reset --hard <commit id>**. This will reset the state of the working directory, staging area, commit history to the specific commit.

```
(base) abhay@Abhays-MacBook-Air spotify-clone % git log --oneline
0b6660b (HEAD -> master, origin/master, origin/HEAD) Playlists Broken
50d8087 Sidebar Done | Features Added
ad20023 Player Skeleton Made
58ad5b2 Player Building
989f9f6 Player Making Start
19208ef Reducer Working
2f45718 Reducer Added
67a2674 DataLayer Made
92a1287 Player Component Made | React Context API Starts
986f6a1 Spotify API Linked
9f74b42 default readme changed
aee50ba spotify login connected
7c454ab spotify linking falied
1230194 Landing Page Made
a636193 De cluttered
fb79cec Initialize project using Create React App
(base) abhay@Abhays-MacBook-Air spotify-clone % git reset --hard 50d8087
HEAD is now at 50d8087 Sidebar Done | Features Added
(base) abhay@Abhays-MacBook-Air spotify-clone % git log --oneline
50d8087 (HEAD -> master) Sidebar Done | Features Added
ad20023 Player Skeleton Made
```


3. In the above example the commit is dropped and the state is changed back to the commit specified.

REVERT

1. Make changes to a file. Stage and commit the file.
2. Now use the command **git revert <commit id>**. If the commit after the specific commit doesn't depend on the commit to be reverted then it revert will be successful else we get errors [example shown below].

```
(base) abhay@Abhays-MacBook-Air spotify-clone % git log --oneline
50d8087 (HEAD -> master) Sidebar Done | Features Added
ad20023 Player Skeleton Made
58ad5b2 Player Building
989f9f6 Player Making Start
19208ef Reducer Working
2f45718 Reducer Added
67a2674 DataLayer Made
92a1287 Player Component Made | React Context API Starts
986f6a1 Spotify API Linked
9f74b42 default readme changed
aee50ba spotify login connected
7c454ab spotify linking falied
1230194 Landing Page Made
a636193 De cluttered
fb79cec Initialize project using Create React App
(base) abhay@Abhays-MacBook-Air spotify-clone % git revert ad20023
Auto-merging src/Body.css
CONFLICT (content): Merge conflict in src/Body.css
Auto-merging src/Sidebar.css
CONFLICT (content): Merge conflict in src/Sidebar.css
Auto-merging src/Sidebar.js
CONFLICT (content): Merge conflict in src/Sidebar.js
error: could not revert ad20023... Player Skeleton Made
hint: After resolving the conflicts, mark them with
hint: "git add/rm <paths>", then run
hint: "git revert --continue".
hint: You can instead skip this commit with "git revert --skip".
hint: To abort and get back to the state before "git revert",
hint: run "git revert --abort".
(base) abhay@Abhays-MacBook-Air spotify-clone %
```

3. If there are no errors then after typing in the command we will get a **vim** editor screen asking for a revert message.

```
scm - vi - git revert 6dabcc6 - 80x24
Revert "Second Commit"

This reverts commit 6dabcc675bccbc9bb2f8d5a7b1f8c9c55901837e.

# 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:
#   modified:   first.cpp
#
~
~
~
~
~
~
~
~
~
~
~/Desktop/scm/.git/COMMIT_EDITMSG 11L, 296B
```

4. Type the message in the string in the first line by replacing the default text.
5. The revert is successful, the changes are undone and the commit is added depicting the message of reverting.

```
(base) abhay@Abhays-MacBook-Air scm % vi first.cpp
(base) abhay@Abhays-MacBook-Air scm % cat first.cpp
// added a comment
// added another comment
(base) abhay@Abhays-MacBook-Air scm % git log --oneline
f76c295 (HEAD -> master) First commit
(base) abhay@Abhays-MacBook-Air scm % git add .
(base) abhay@Abhays-MacBook-Air scm % git commit -m "Second Commit"
[master 6dabcc6] Second Commit
 1 file changed, 1 insertion(+)
(base) abhay@Abhays-MacBook-Air scm % git log --oneline
6dabcc6 (HEAD -> master) Second Commit
f76c295 First commit
(base) abhay@Abhays-MacBook-Air scm % git revert 6dabcc6
[master 22b87ff] Revert "Second Commit"
 1 file changed, 1 deletion(-)
(base) abhay@Abhays-MacBook-Air scm % git log --oneline
22b87ff (HEAD -> master) Revert "Second Commit"
6dabcc6 Second Commit
f76c295 First commit
(base) abhay@Abhays-MacBook-Air scm % cat first.cpp
// added a comment
(base) abhay@Abhays-MacBook-Air scm %
```

Task 2

Made By: Abhay Aggarwal

Roll No: 2110990034

Submitted To: Dr. Monit Kapoor

Team No: 01

Subject Name: Source Code Management

Subject Code: CS181

Cluster: Beta

Department: CSE

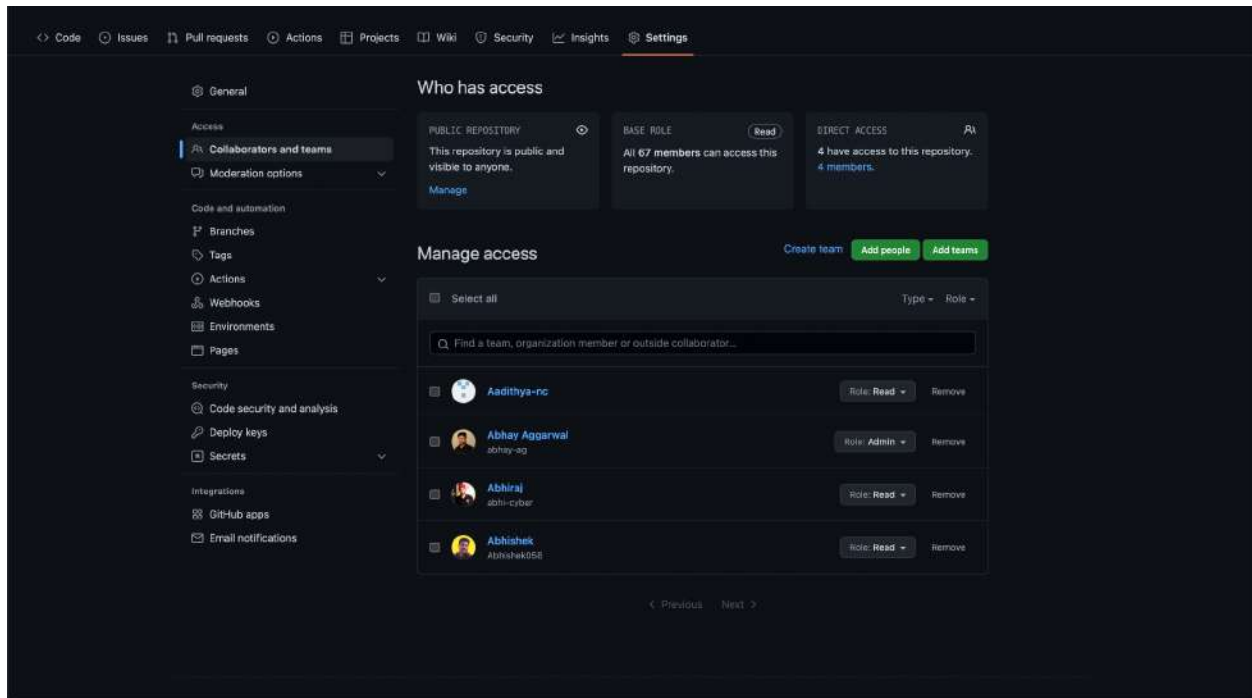
INDEX

Sr. No.	Title
1.	Creating a Distributed Repository and add members to project team
2.	Open and close Pull Request
3.	Publish and Print Network Graphs
4.	Pull Request from each member of team

TODO: Create a distributed repository and add team members.

STEPS:

1. We created a repository in the organization by the naming convention **<Roll No>**.
2. Then from the **collaborator** section of the **settings** panel I added my other teammates as collaborators in the repository.

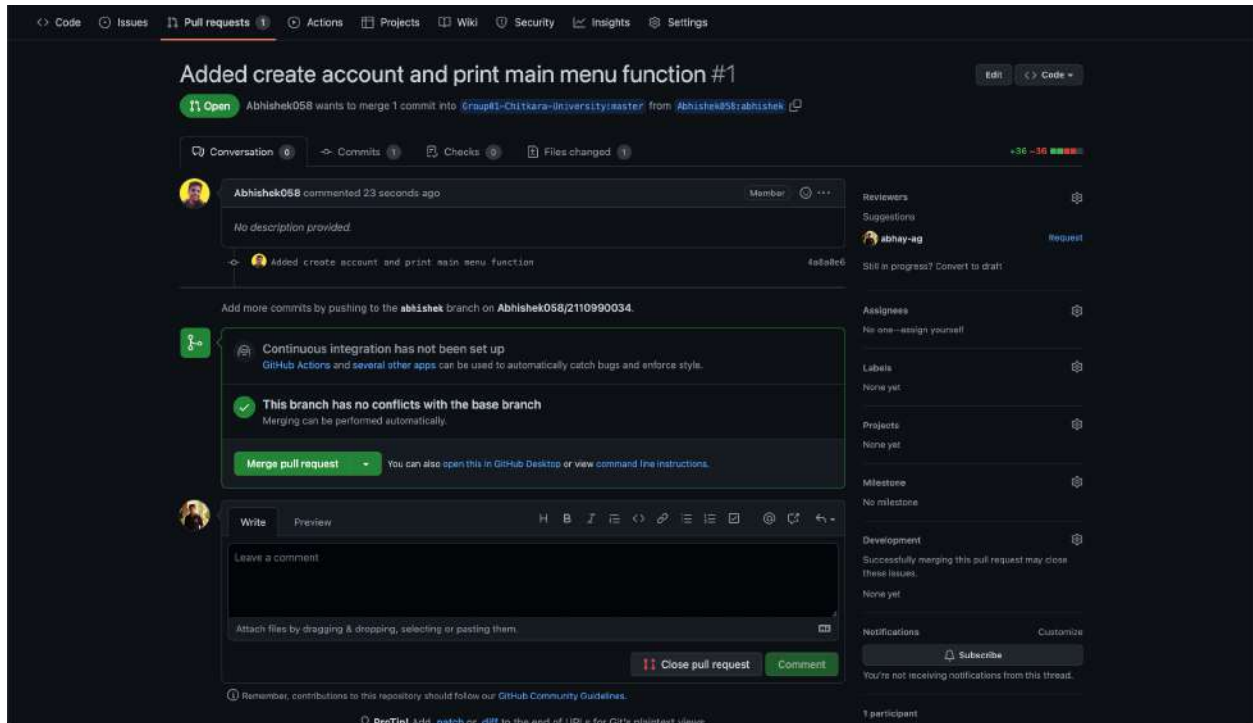


3. The distributed repository is now created, but if the members were added as collaborators then we could not generate any pull request except the ones which involved branches.
4. To avoid this issue I removed them as a collaborator. We then decided to **fork and commit** to each other's repo.
5. We then forked each other's repo and started working on the given code.
6. In my repository the code is in **Python** language and is a basic **ATM Project**. My teammates started working on the code by creating their own branches and then pushing and testing the code.

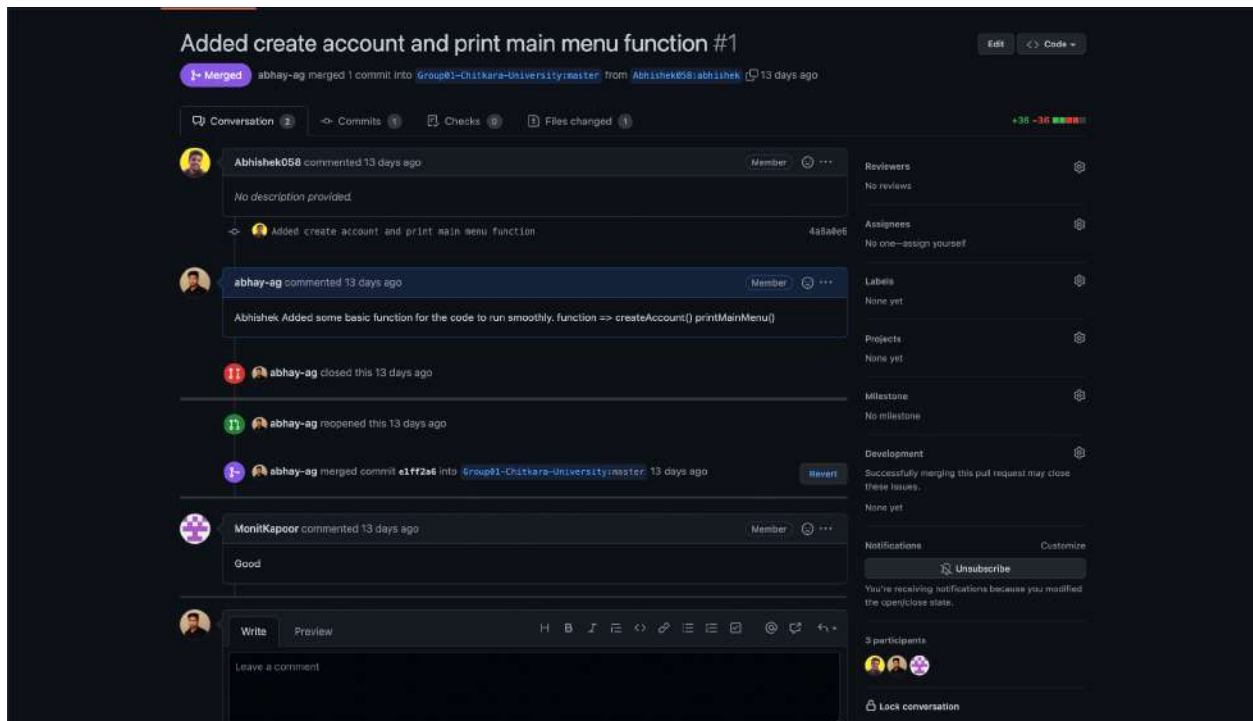
TODO: Open and Close Pull Requests.

STEPS:

1. My teammates started pushing code in their own branches and then opened pull requests in my repository by **Contributing** to my code.

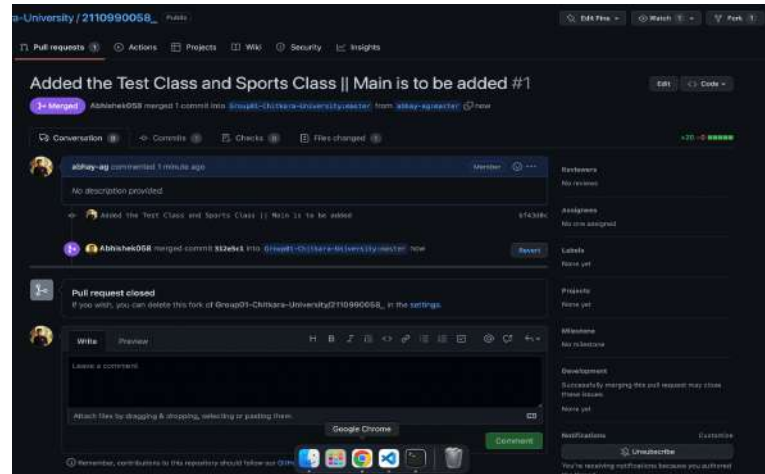
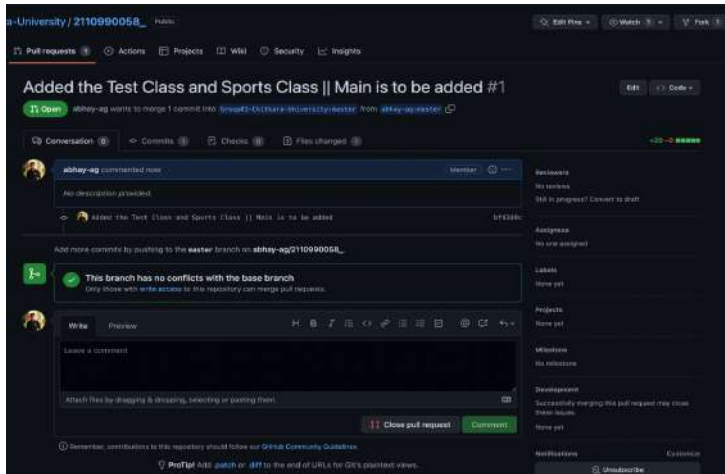


2. I closed them simultaneously as a maintainer of the repo.

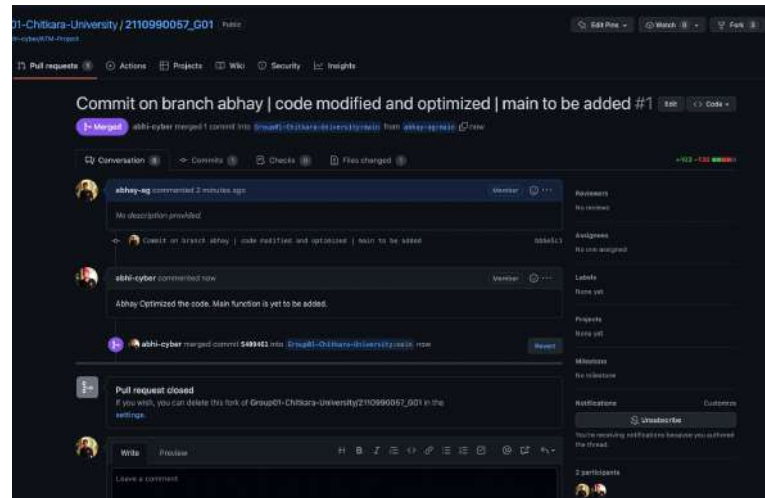
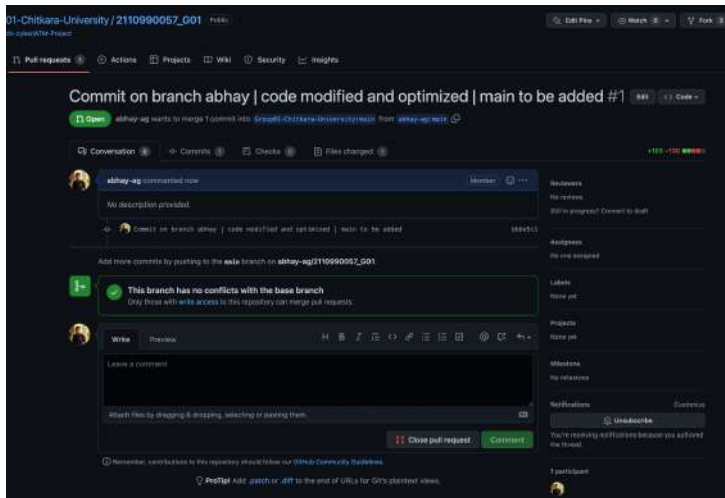


3. As a member of the team and the requirements of the project work I also had to open and close pull requests in my teammates repositories. Below are the screenshots of the same.

First two screenshots: Contribution in Abhishek058



Second two screenshots: Contribution in abhi-cyber



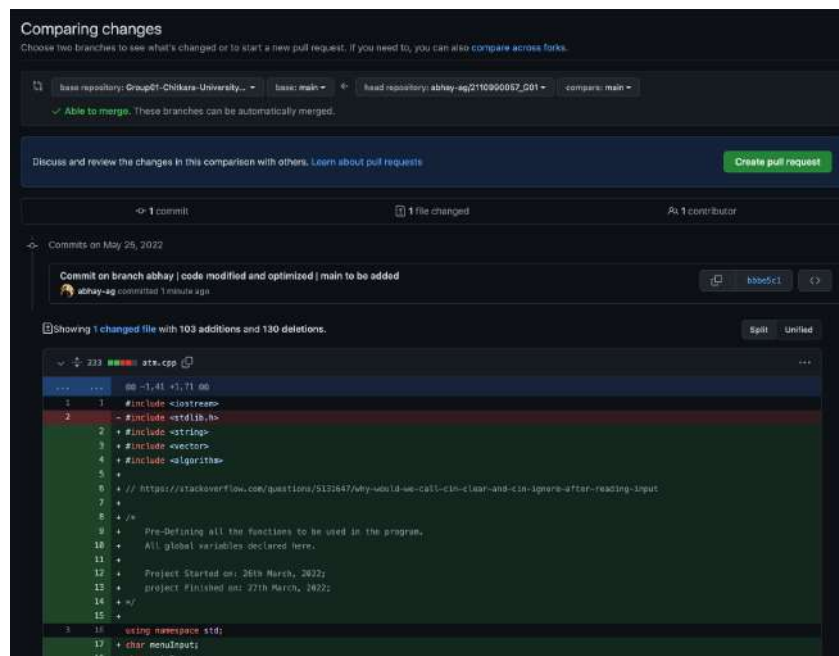
4. I worked with **abhi-cyber**, **Aadithya-nc**, **Abhishek058** on their repositories and modified their code, added new functionalities and tested their code.
5. To achieve this first I had to fork the repo into my account and then I cloned them onto my local machine.

6. After cloning I reviewed the code, made changes on my own branch and then committed to the forked repo. From where I then open a pull request and it's reflected on the repo owner's pull requests tab. [Screenshots attached for the same]. The changes made by me are also visible to the owner.

a. Screenshots for **abhi-cyber**

```
Last login: Wed May 18 14:01:57 on ttys000
(base) abhay@Abhays-MacBook-Air ~ % cd Desktop
(base) abhay@Abhays-MacBook-Air Desktop % git clone https://github.com/abhay-ag/2110990057_G01.git
Cloning into '2110990057_G01'...
remote: Enumerating objects: 42, done.
remote: Counting objects: 100% (42/42), done.
remote: Compressing objects: 100% (30/30), done.
remote: Total 42 (delta 12), reused 41 (delta 11), pack-reused 0
Receiving objects: 100% (42/42), 74.04 KiB | 606.00 KiB/s, done.
Resolving deltas: 100% (12/12), done.
(base) abhay@Abhays-MacBook-Air Desktop %
```

```
(base) abhay@Abhays-MacBook-Air 2110990057_G01 % git commit -a -m "Commit on branch abhay | code modified and optimized | main to be added"
[abhay bbb5c1] Commit on branch abhay | code modified and optimized | main to be added
1 file changed, 137 insertions(+), 154 deletions(-)
rewrite atm.cpp (82%)
(base) abhay@Abhays-MacBook-Air 2110990057_G01 % git push -u origin abhay
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 8 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 1.70 KiB | 1.70 MiB/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 'abhay' on GitHub by visiting:
remote:   https://github.com/abhay-ag/2110990057_G01/pull/new/abhay
remote:
To https://github.com/abhay-ag/2110990057_G01.git
 * [new branch]      abhay -> abhay
branch 'abhay' set up to track 'origin/abhay'.
(base) abhay@Abhays-MacBook-Air 2110990057_G01 % git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.
(base) abhay@Abhays-MacBook-Air 2110990057_G01 % git merge abhay
Updating 2d0fbd0..bbb5c1
Fast-forward
 atm.cpp | 233 +++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
 1 file changed, 103 insertions(+), 130 deletions(-)
(base) abhay@Abhays-MacBook-Air 2110990057_G01 % git push -u origin main
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/abhay-ag/2110990057_G01.git
 2d0fbd0..bbb5c1  main -> main
branch 'main' set up to track 'origin/main'.
(base) abhay@Abhays-MacBook-Air 2110990057_G01 %
```



b. Screenshots for **Abhishek058**

```
(base) abhay@Abhays-MacBook-Air 2110990058_ % git branch
* abhay
  master
(base) abhay@Abhays-MacBook-Air 2110990058_ % git commit -a -m "Added the Test Class and Sports Class || Main is to be added"
[abhay bf43d0c] Added the Test Class and Sports Class || Main is to be added
 1 file changed, 20 insertions(+)
(base) abhay@Abhays-MacBook-Air 2110990058_ % git push -u origin abhay
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 8 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 512 bytes | 512.00 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 'abhay' on GitHub by visiting:
remote:   https://github.com/abhay-ag/2110990058_/pull/new/abhay
remote:
To https://github.com/abhay-ag/2110990058_.git
 * [new branch]   abhay -> abhay
branch 'abhay' set up to track 'origin/abhay'.
(base) abhay@Abhays-MacBook-Air 2110990058_ % git checkout master
Switched to branch 'master'
Your branch is up to date with 'origin/master'.
(base) abhay@Abhays-MacBook-Air 2110990058_ % git merge abhay
Updating f7f520a..bf43d0c
Fast-forward
 res_class.cpp | 20 ++++++++++++++++++++++
 1 file changed, 20 insertions(+)
(base) abhay@Abhays-MacBook-Air 2110990058_ %
```

Comparing changes

Choose two branches to see what's changed or to start a new pull request. If you need to, you can also [compare across forks](#).

base repository: Group01-Chitkara-University... base: master ← head repository: abhay-ag/2110990058_ compare: master

✓ Able to merge. These branches can be automatically merged.

Discuss and review the changes in this comparison with others. [Learn about pull requests](#) [Create pull request](#)

↔ 1 commit 1 file changed R1 contributor

Commits on May 25, 2022

Added the Test Class and Sports Class || Main is to be added
abhay-ag committed 1 minute ago [b743d0c](#) <>

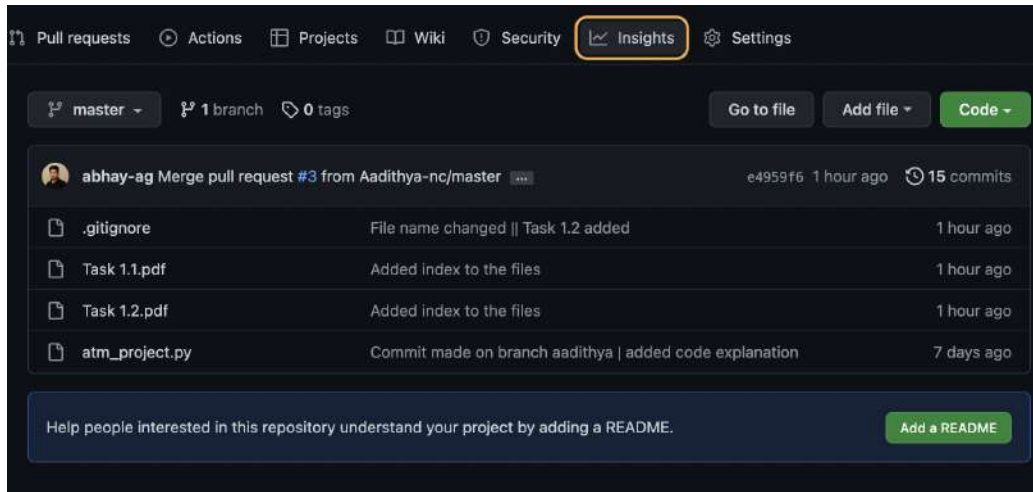
Showing 1 changed file with 20 additions and 0 deletions. [Split](#) [Unified](#)

```
res_class.cpp
11 11         rfile = i;
12 12         name = nj;
13 13     }
14 + };
15 + class Sports{
16 +     protected:
17 +         int score;
18 +
19 +     public:
20 +         Sports(int sc = 0){
21 +             score = sc;
22 +         }
23 + };
24 +
25 + class Testipublic Student{
26 +     protected:
```

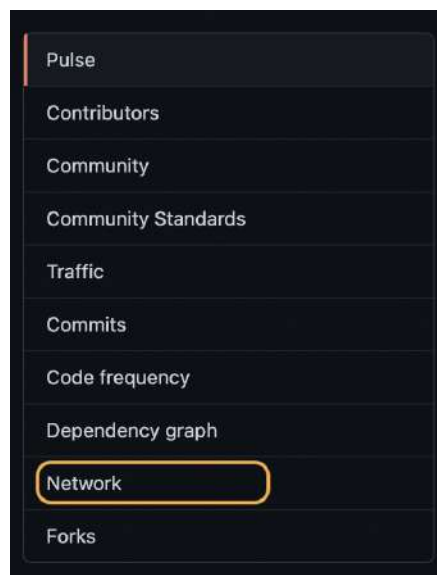

TODO: Publish and Print Network Graphs

STEPS:

1. Goto the **Insights** tab of the Repository.



2. Click on the **Network** panel in the navigation panel.



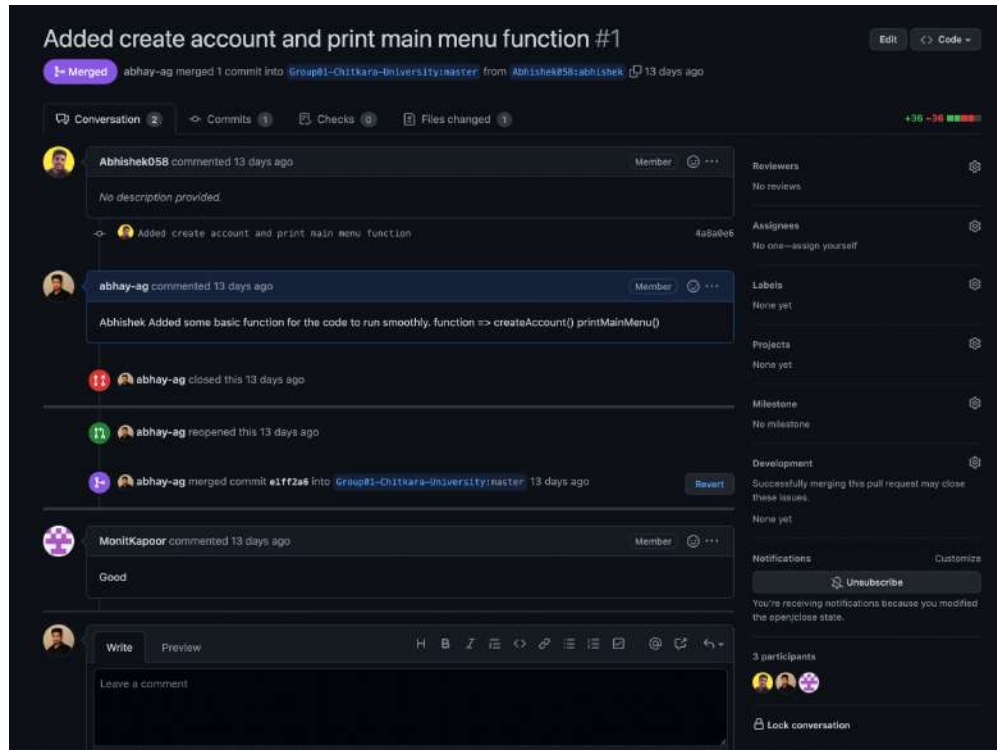
3. We can now visualize the network graph of our repository. Mine is attached below.



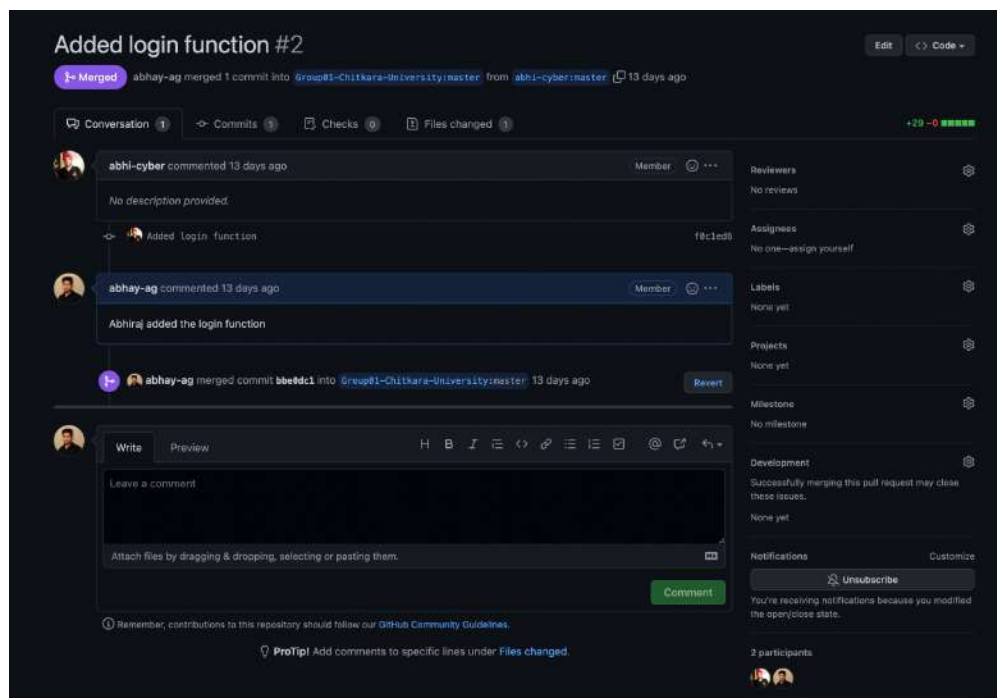
4. Through the network graph we can visualize the creating and merging of all the branches from the master branch and into the master branch. This network graph is very simple, they can be quite complex in bigger repositories.

TODO: Pull request from each member of the team.

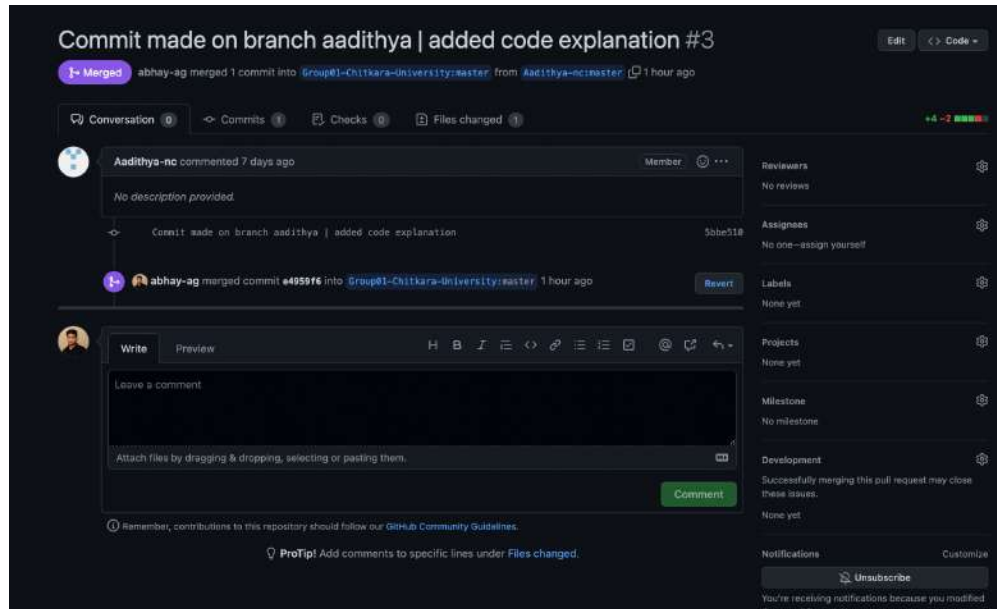
- **Member #1**



- **Member #2**



- **Member #3**



- **Screenshots of the total pull requests and contributors in my repository.**

