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**

Task 1.2 - Experiment No. 01

AIM: Add collaborators on a GitHub repo

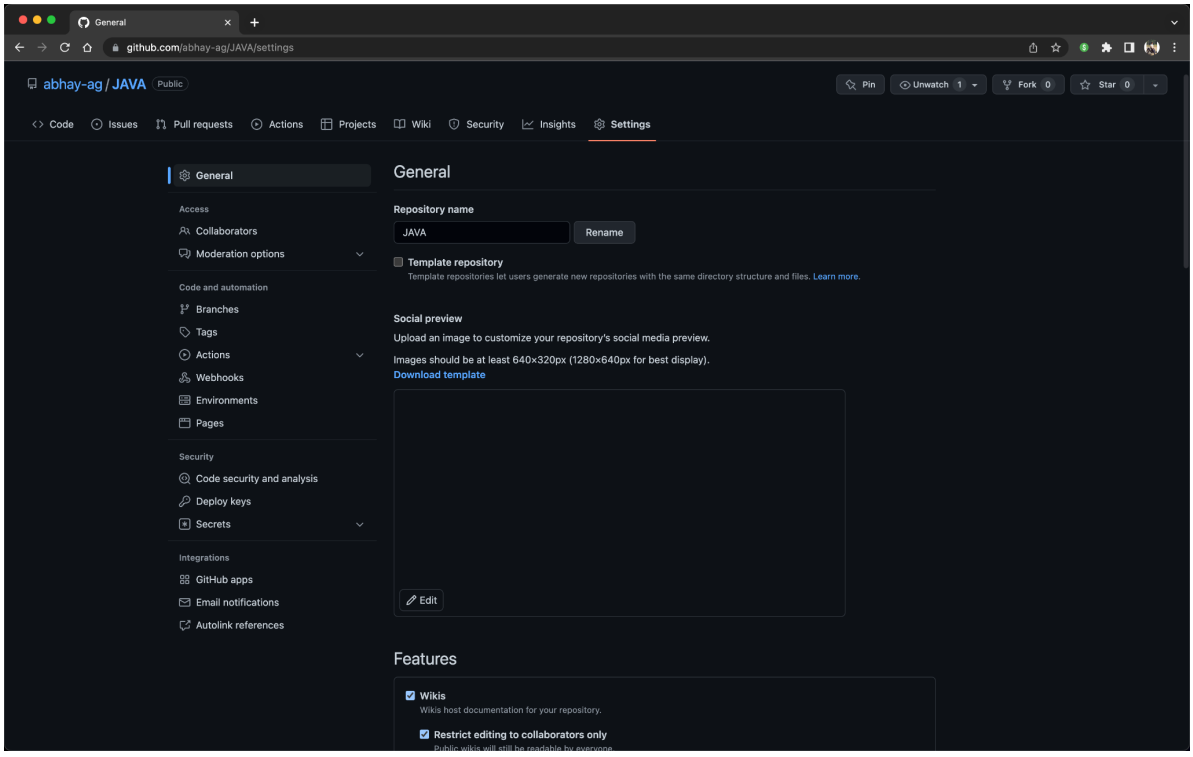
Theory:

- **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.

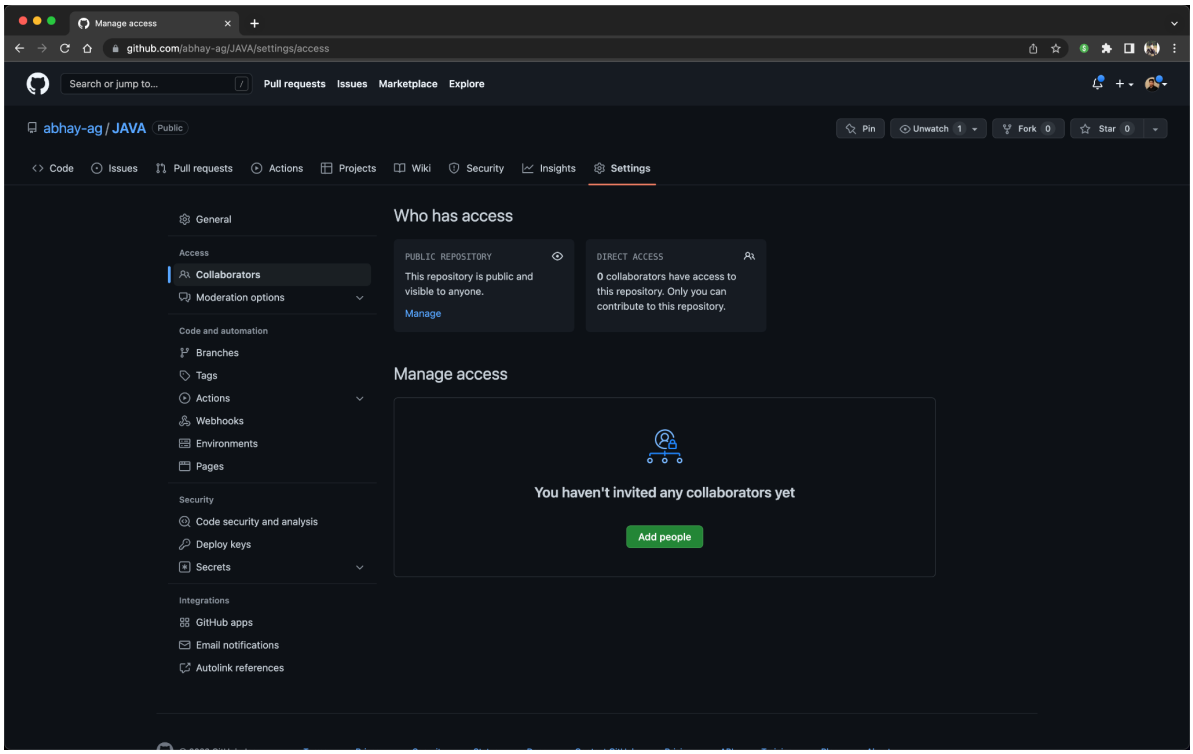
- **NOTE:** The invitation sent to the collaborator if not accepted will expire in seven (7) days.

Procedure:

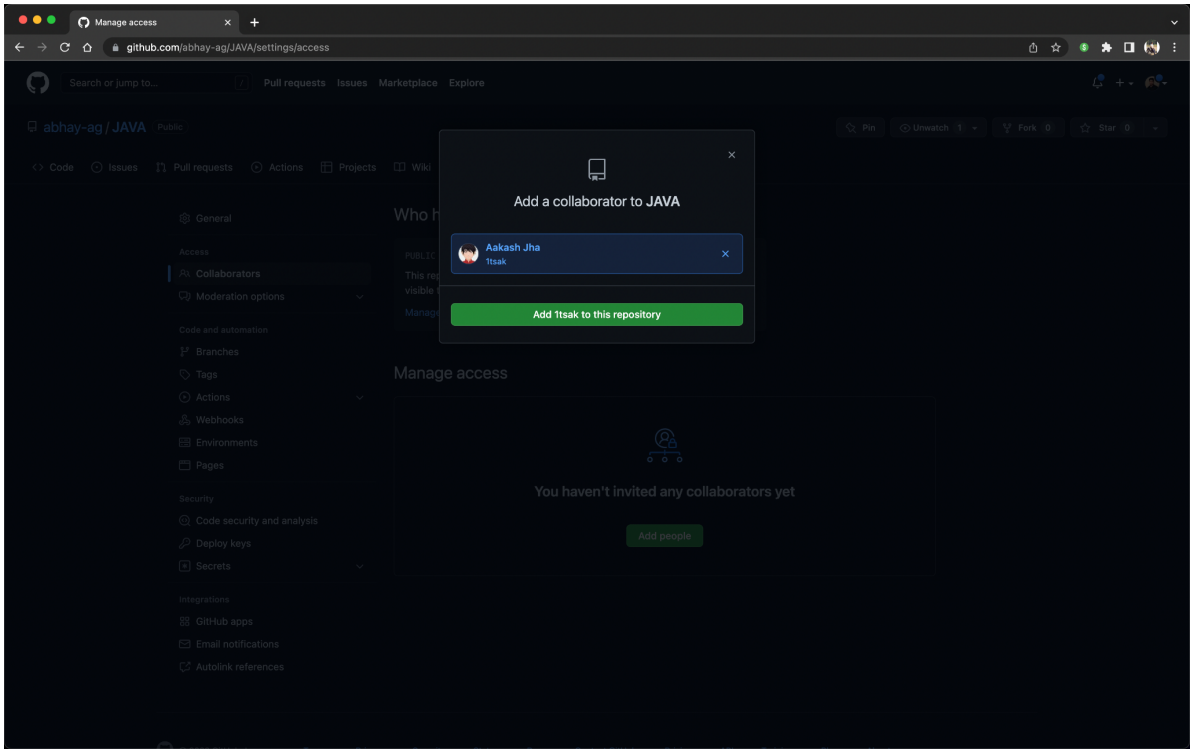
- To add a collaborator into a repo go to the setting tab of the particular repo.



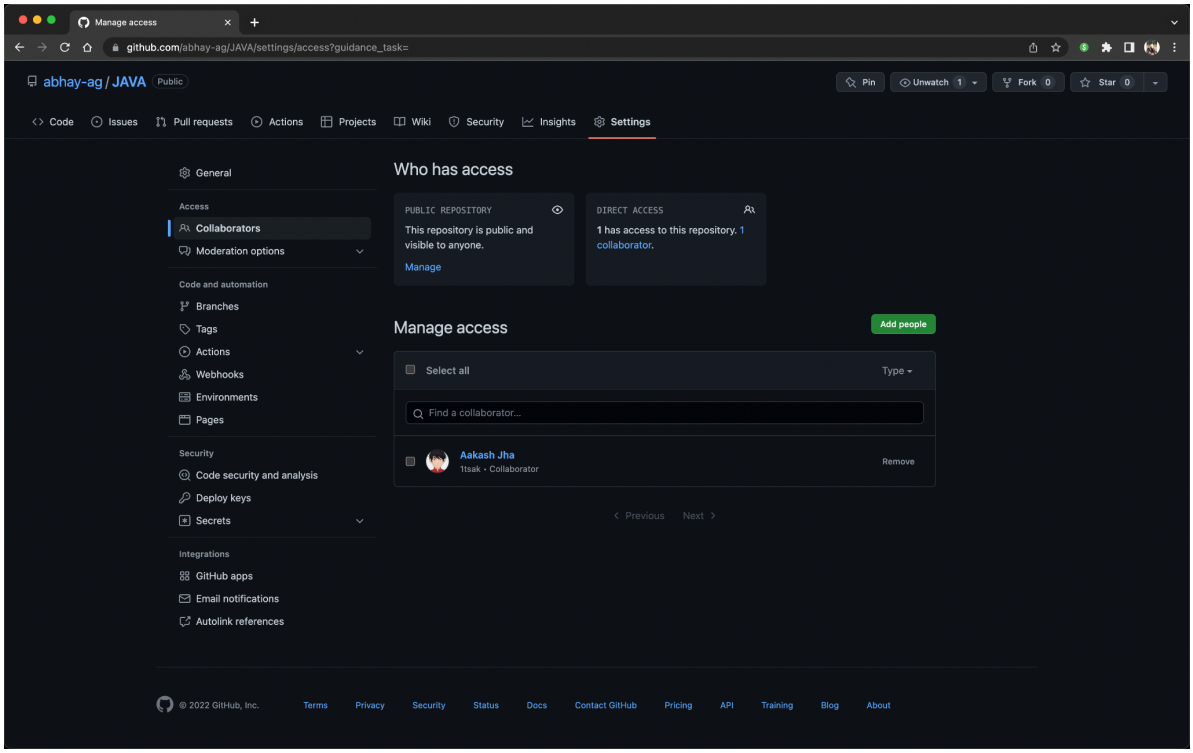
- Navigate to the **COLLABORATORS** section of the navigation panel.



- 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

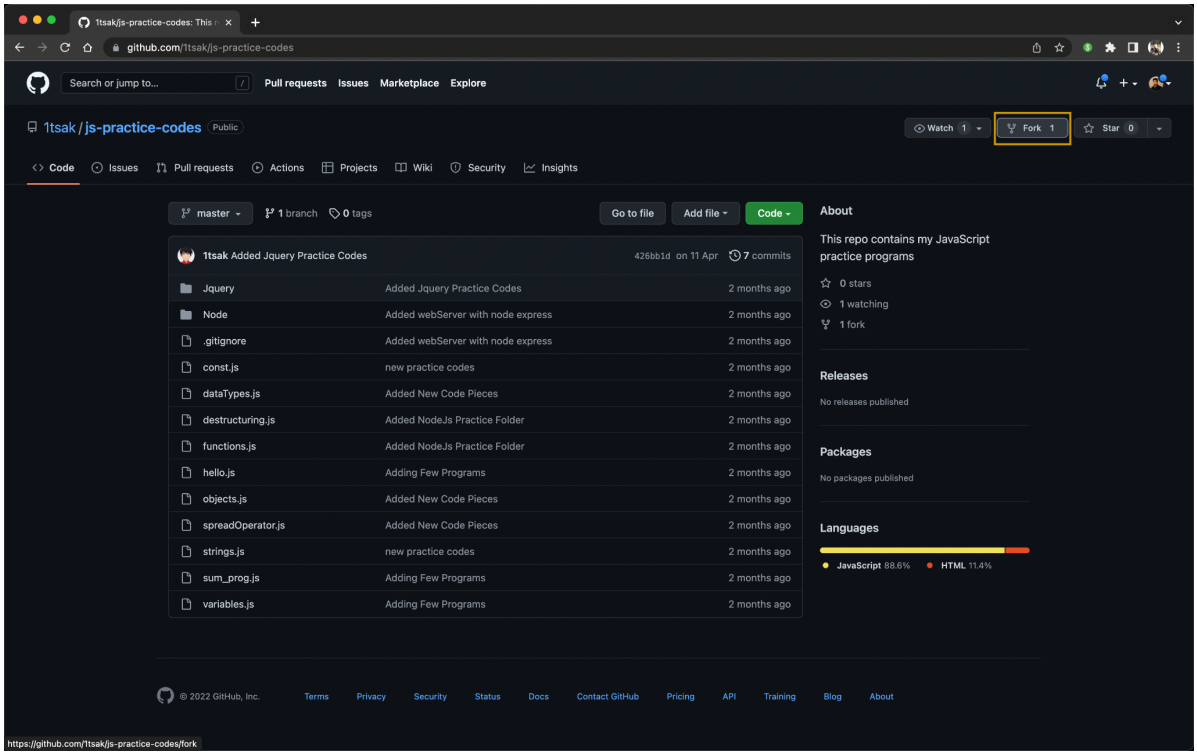
AIM : Fork and Commit

Theory:

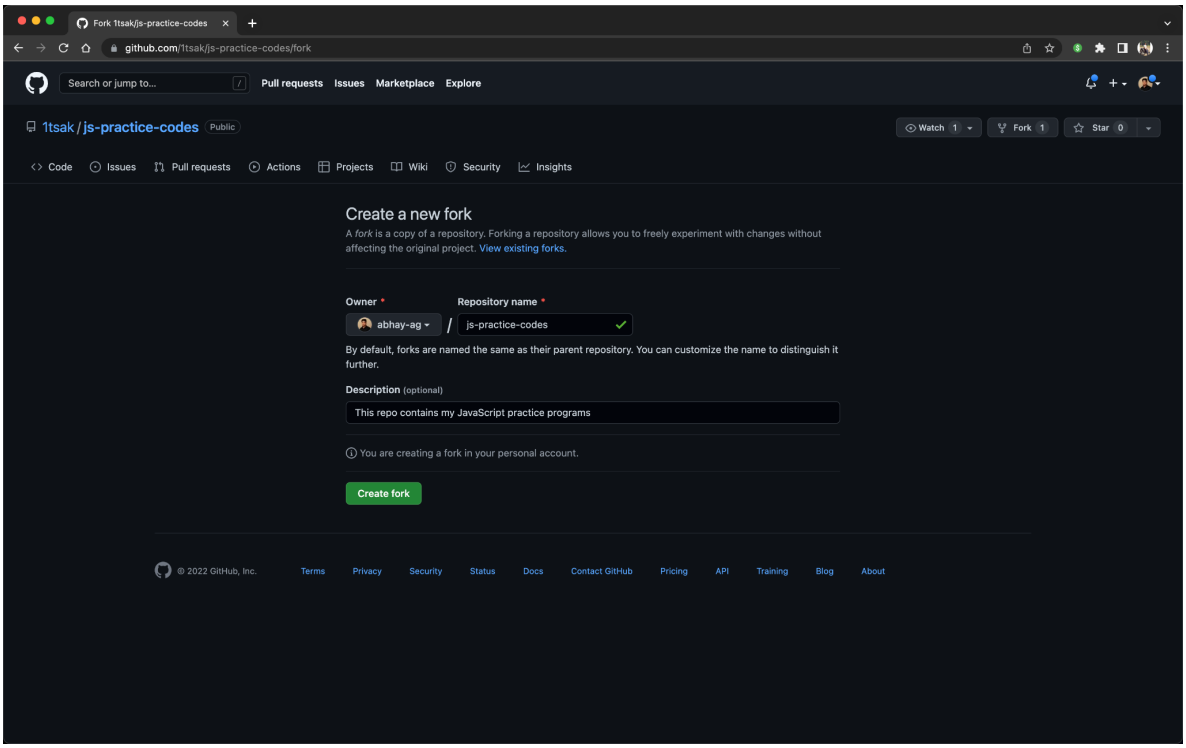
- **Fork:** Forking a repo creates a copy of a repository in the GitHub account of the user who forks the repo. Now the ownership of the repository shifts to the user instead of the person from whom the repo is forked, and the user can freely commit, set up branches, open pull requests etc. on the repo.
- **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.

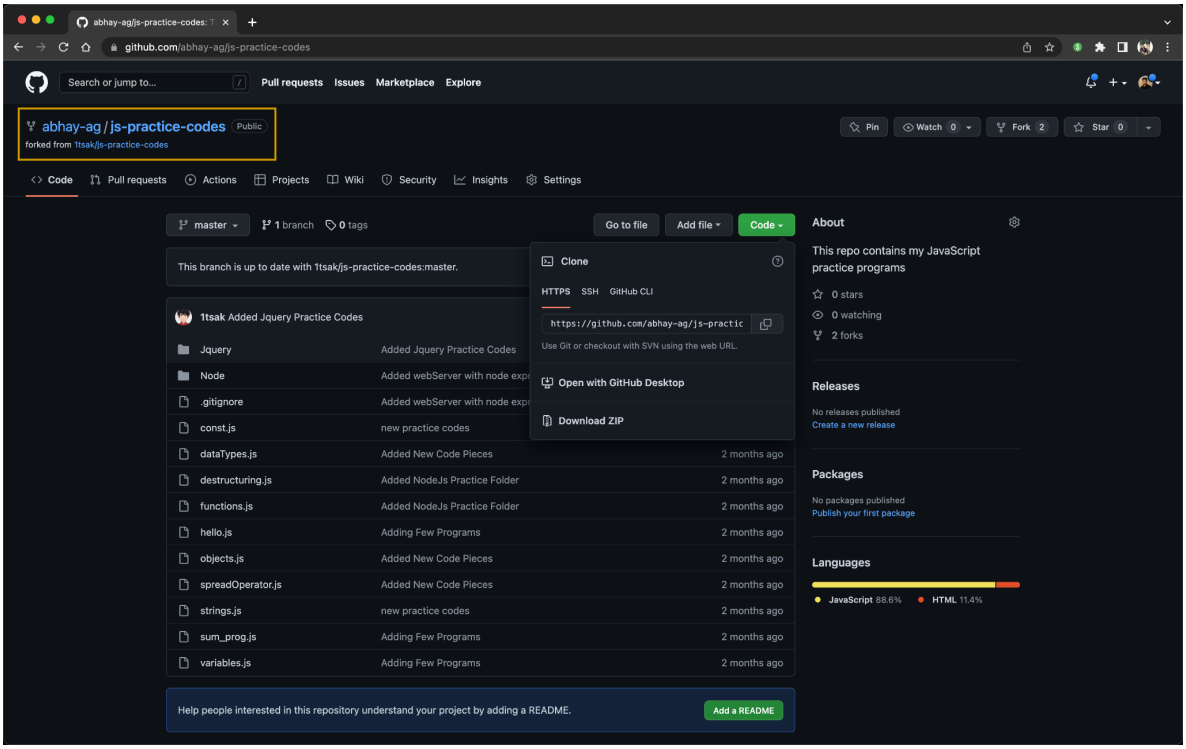
Procedure:

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

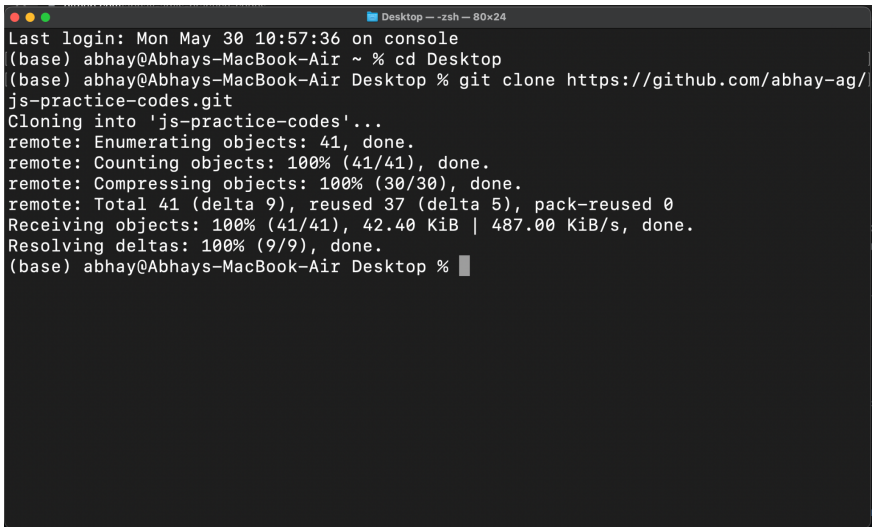


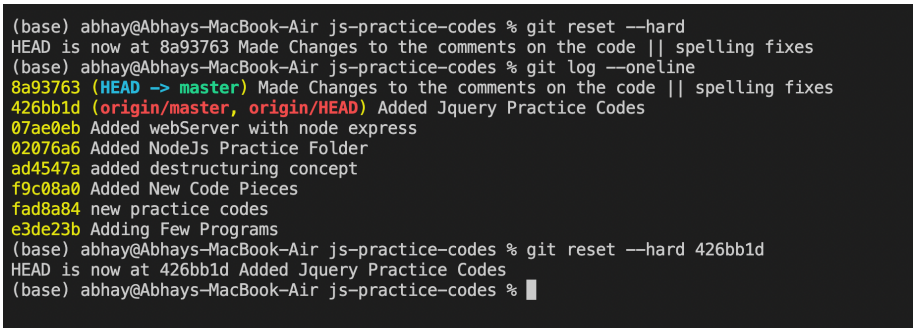
2. After forking [if part of organizations select the owner] the copy of the repo will be created in the user's account.

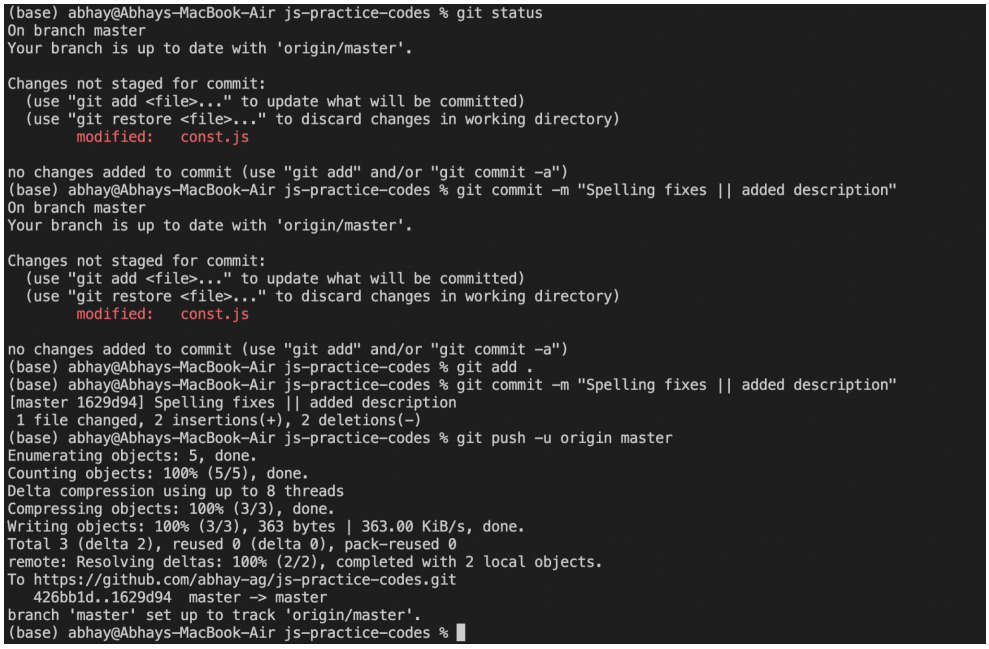




3. The fork is now created in the user account and can now be cloned onto the local machine and commits can now be added without disturbing the code in the original code.





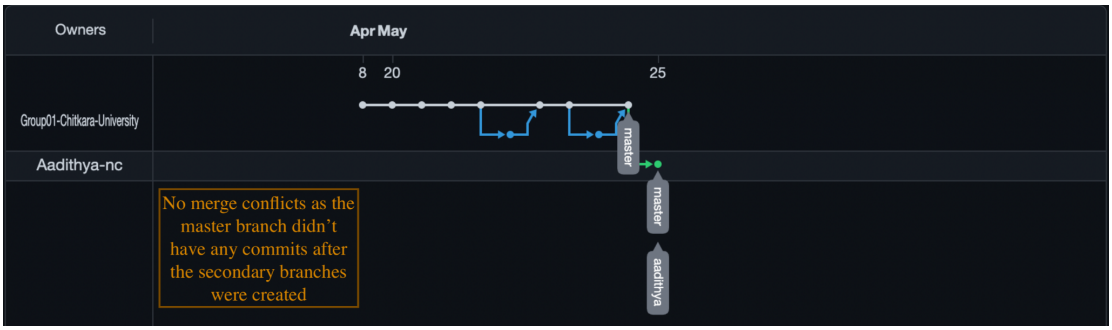


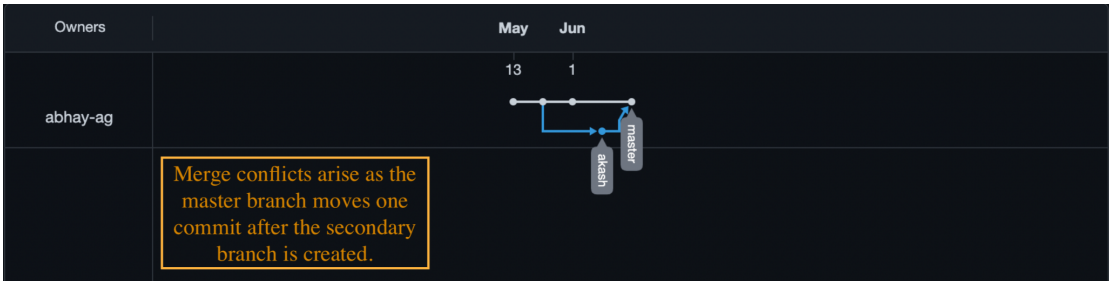
Task 1.2 - Experiment No. 03

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]





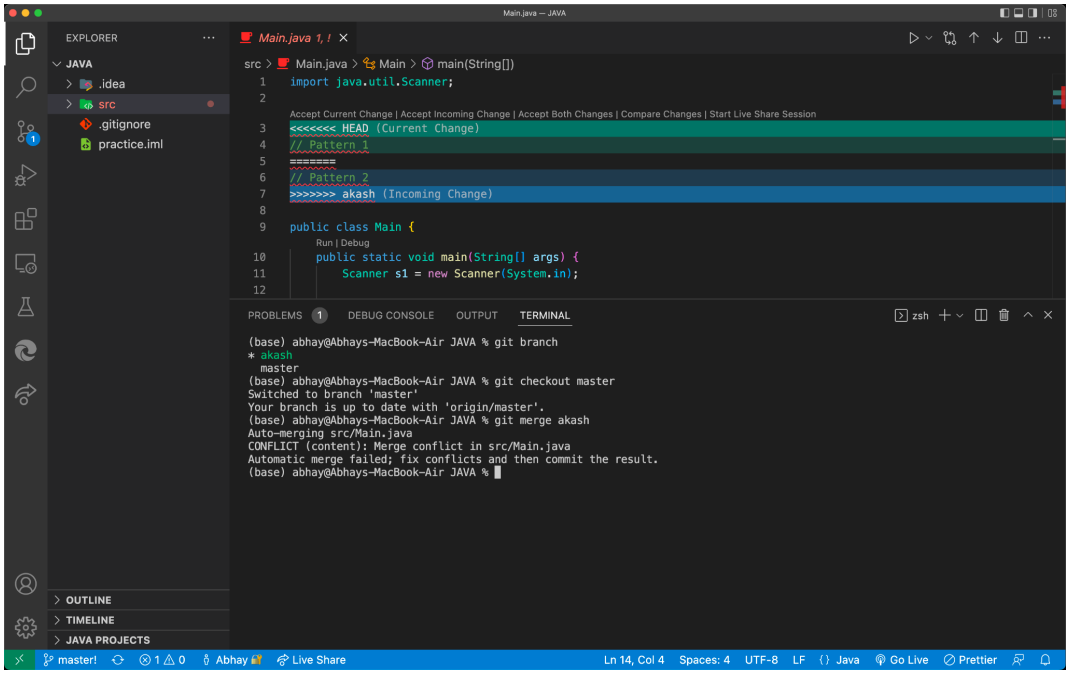
Procedure:

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.

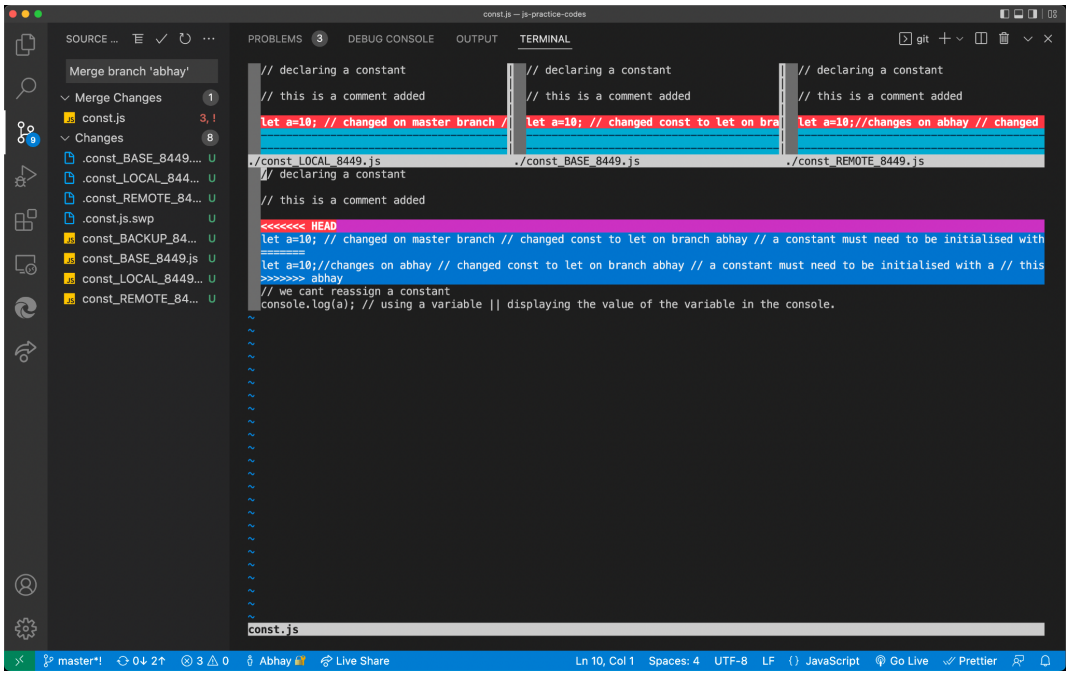
```
(base) abhay@Abhays-MacBook-Air js-practice-codes % git commit -a -m "Changes on master branch"
[master 01f3037] Changes on master branch
 1 file changed, 1 insertion(+), 1 deletion(-)
(base) abhay@Abhays-MacBook-Air js-practice-codes % git checkout abhay
Switched to branch 'abhay'
(base) abhay@Abhays-MacBook-Air 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 %
```

2. 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.

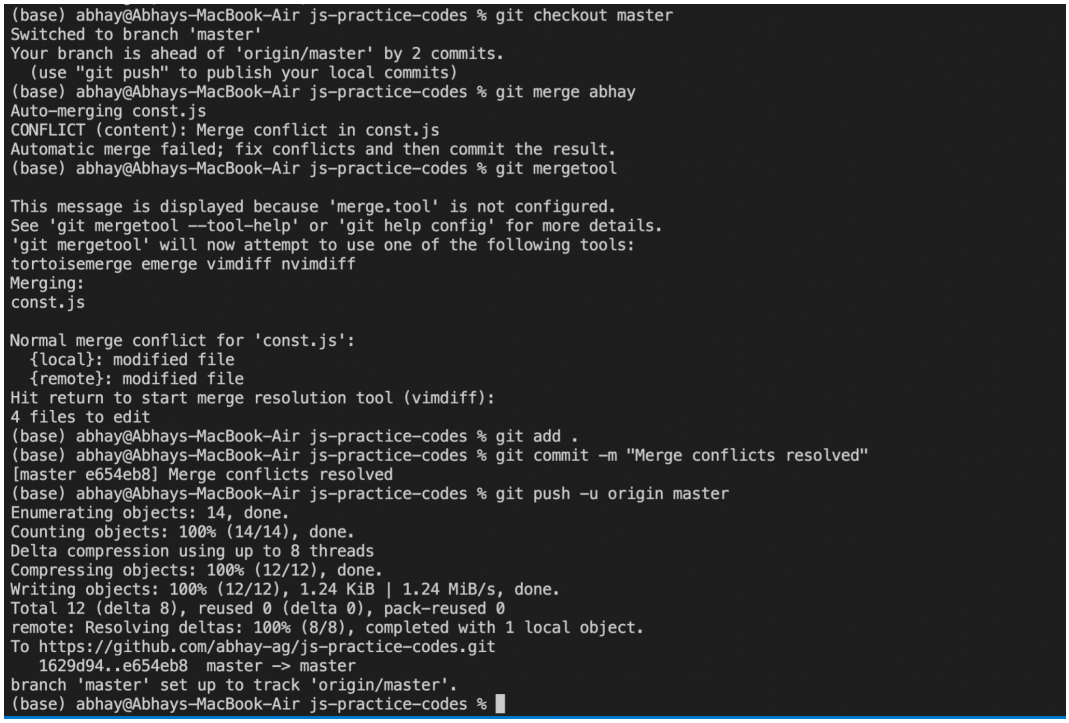
THIS DEMONSTRATES THE ERRORS IN VS CODE [made by collaborator activity]



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



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.



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
```

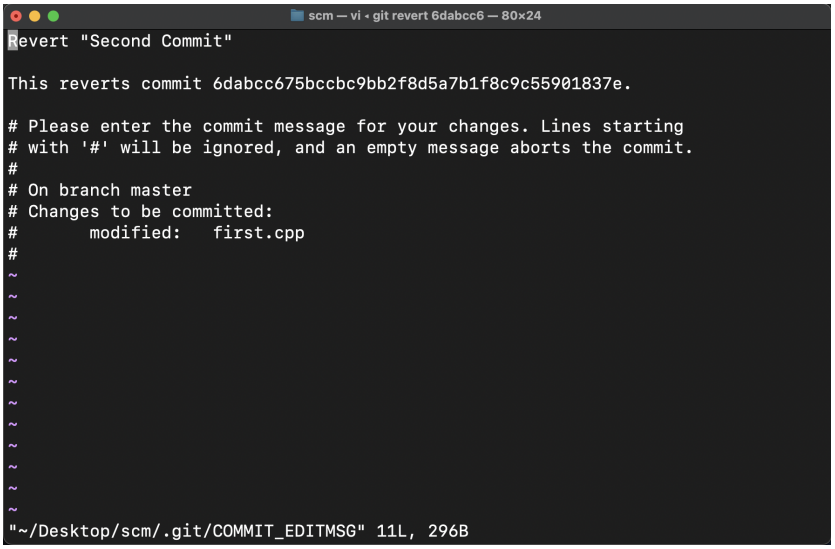
3. In the above example the commit is dropped and the state is changed back to the commit specified.

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 <pathspec>", 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.



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.