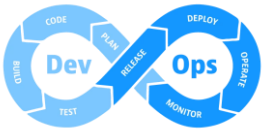


Complete Git & GitHub Workflow for an IoT-Based ECE Project

CCS342-DevOps

Submitted by,

SUBADEESH D

420722106048

ECE FINAL YEAR

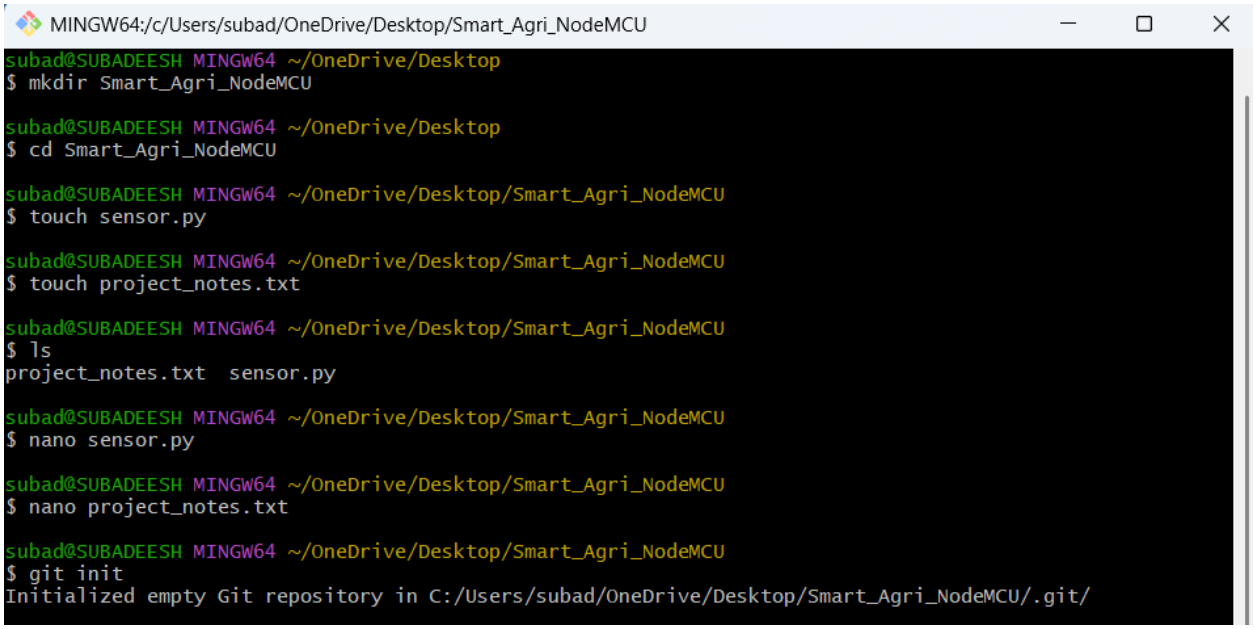
Repository Setup & Commit History

Step 1. Create a new project folder using the mkdir command, then navigate into it using cd.

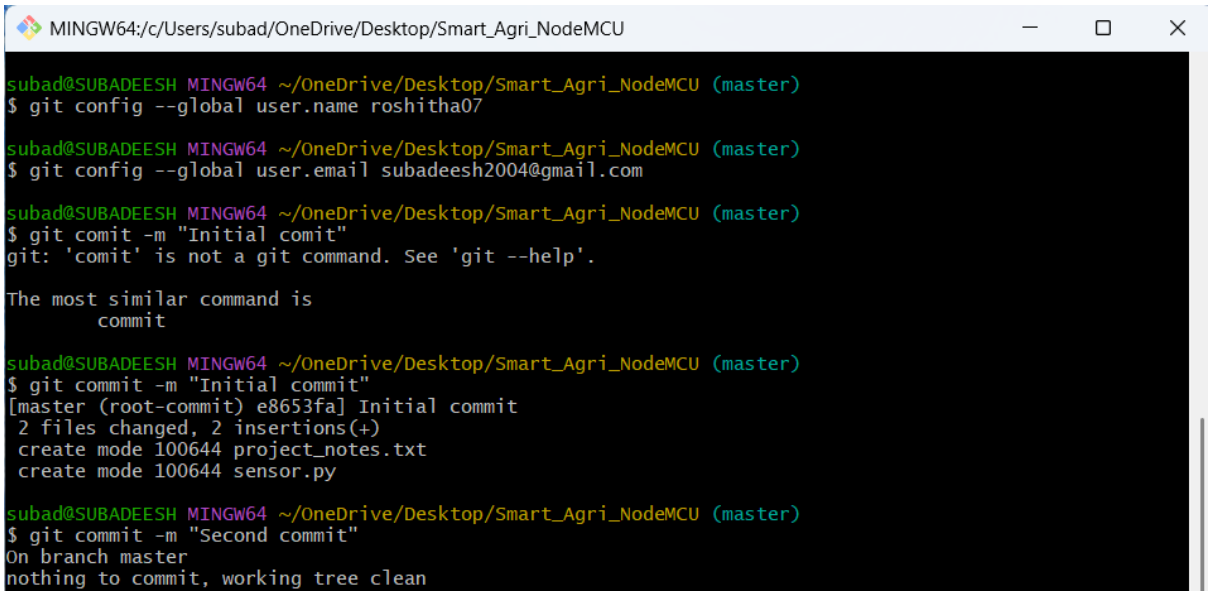
Step 2. Generate two files—sensor.py and project_notes.txt—using the touch command. Confirm their creation by listing the directory contents with ls.

Step 3. Open and edit both files using the nano editor to add initial content relevant to the IoT project.

Step 4. Initialize a Git repository with git init, and verify the repository status using git status.

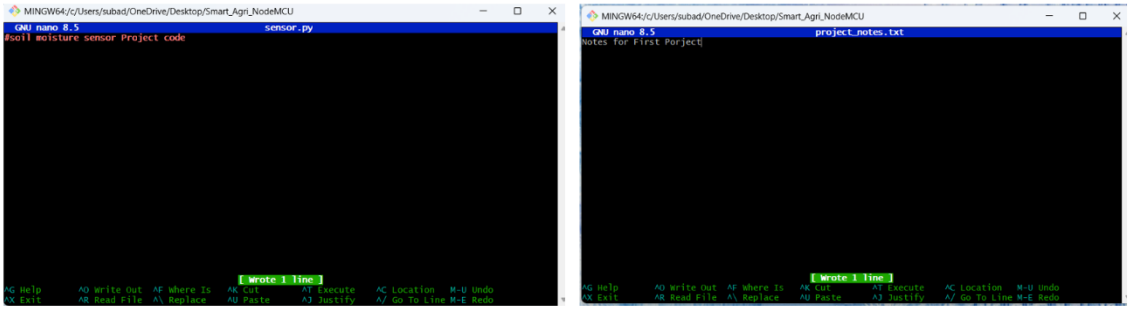


Step 5. Configure Git with your username and email using git config. Stage the files and perform the first commit using git commit -m.

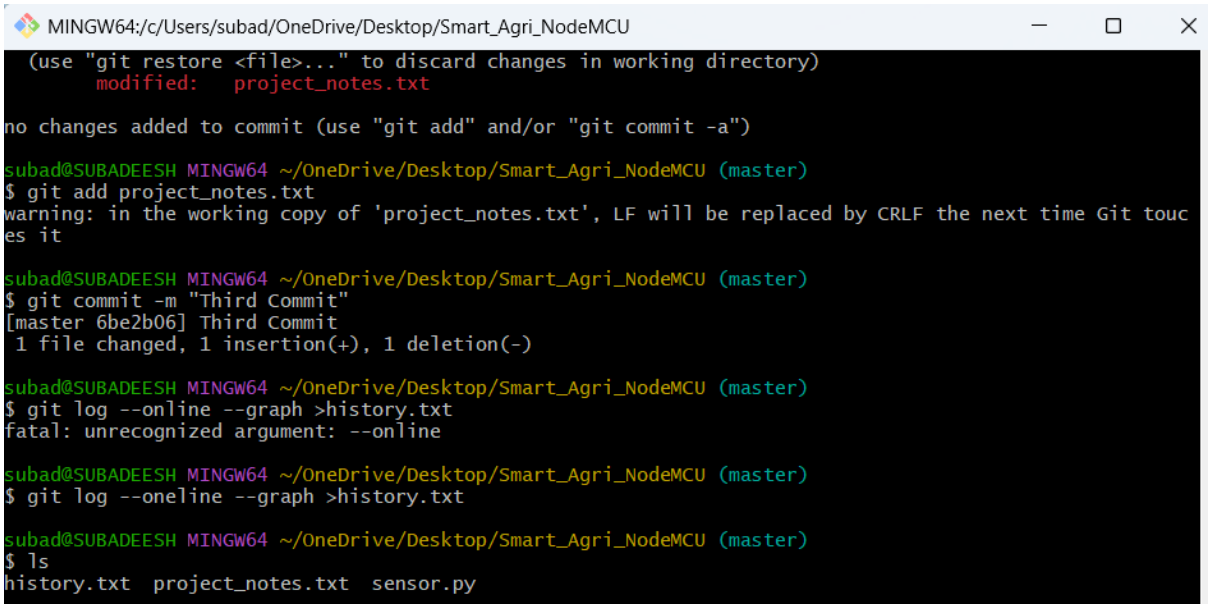


Step 6. Make edits to sensor.py, stage the changes, and execute the second commit to capture the update.

Step 7. Modify project_notes.txt, add the changes to the staging area, and perform the third commit.



Step 8. Export a concise visual log of the commit history using git log --oneline --graph > history.txt. Use ls to confirm that history.txt has been successfully created.



Branching, Merging & Conflict Resolution

Step 9. Create two new branches—feature_sensor and feature_cloud—to manage separate development tasks related to sensor data and cloud integration.

```
subad@SUBADEESH MINGW64 ~/OneDrive/Desktop/Smart_Agri_NodeMCU (master)
$ git branch origin

subad@SUBADEESH MINGW64 ~/OneDrive/Desktop/Smart_Agri_NodeMCU (master)
$ git branch feature_sensor

subad@SUBADEESH MINGW64 ~/OneDrive/Desktop/Smart_Agri_NodeMCU (master)
$ git branch feature_cloud
```

Step 10. Switch to the feature_sensor branch and update the sensor.py file by adding a print statement for reading soil moisture sensor data. Stage the modified file for commit.

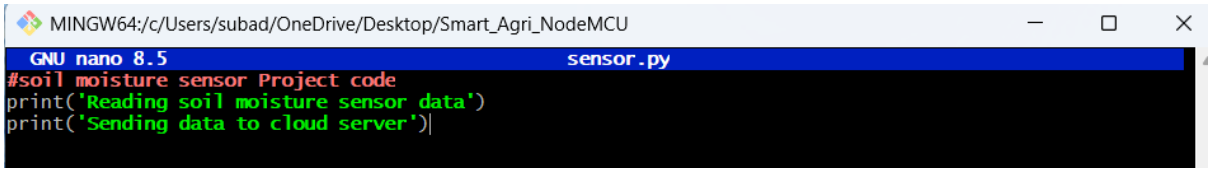
```
subad@SUBADEESH MINGW64 ~/OneDrive/Desktop/Smart_Agri_NodeMCU (master)
$ git checkout feature_sensor
Switched to branch 'feature_sensor'

subad@SUBADEESH MINGW64 ~/OneDrive/Desktop/Smart_Agri_NodeMCU (feature_sensor)
$ nano sensor.py

subad@SUBADEESH MINGW64 ~/OneDrive/Desktop/Smart_Agri_NodeMCU (feature_sensor)
$ git add sensor.py
warning: in the working copy of 'sensor.py', LF will be replaced by CRLF the next time Git touches it

subad@SUBADEESH MINGW64 ~/OneDrive/Desktop/Smart_Agri_NodeMCU (feature_sensor)
$ git commit -m "Added Moisture Reading"
[feature_sensor 80bde6f] Added Moisture Reading
 1 file changed, 1 insertion(+)
```

Step 11. Switch to the feature_cloud branch and modify the sensor.py file to include a print statement for sending data to the cloud server. Stage the updated file for commit.

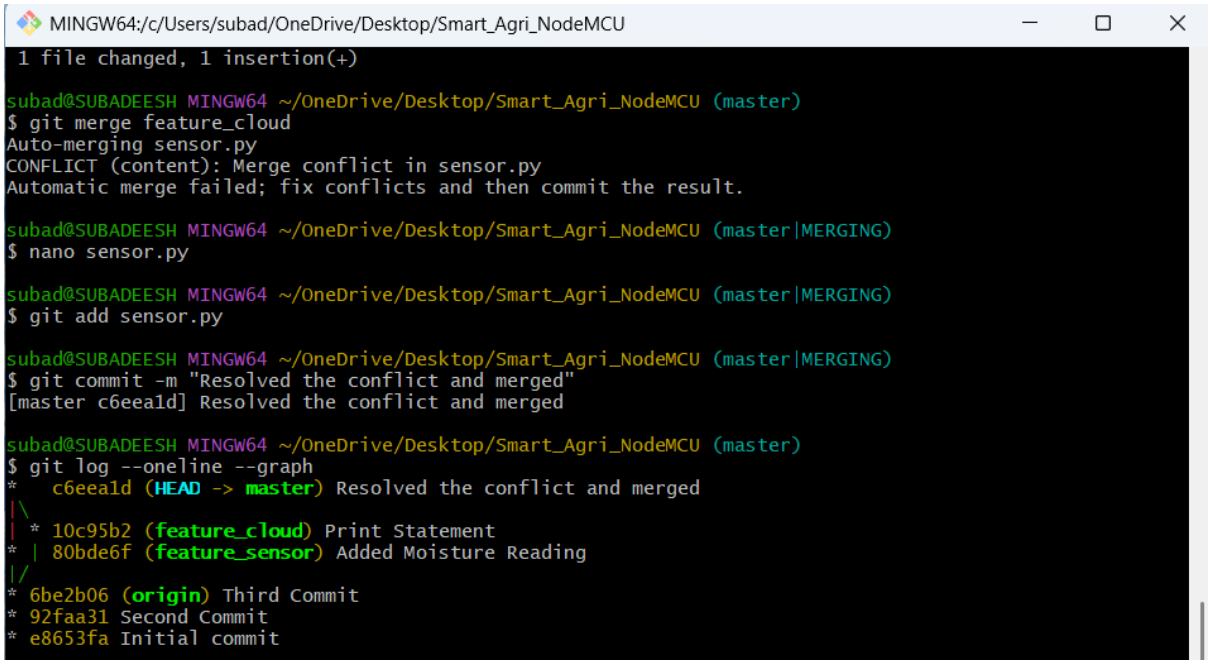
```
subad@SUBADEESH MINGW64 ~/OneDrive/Desktop/Smart_Agri_NodeMCU (feature_sensor)
$ git checkout feature_cloud
Switched to branch 'feature_cloud'

subad@SUBADEESH MINGW64 ~/OneDrive/Desktop/Smart_Agri_NodeMCU (feature_cloud)
$ nano sensor.py

subad@SUBADEESH MINGW64 ~/OneDrive/Desktop/Smart_Agri_NodeMCU (feature_cloud)
$ nano sensor.py

subad@SUBADEESH MINGW64 ~/OneDrive/Desktop/Smart_Agri_NodeMCU (feature_cloud)
$ git add sensor.py
```

Step 12. Merge both branches into the main branch. During the merge process, a conflict arises in sensor.py, which is manually resolved by retaining both print statements. The resolved file is committed to finalize the merge.

```
subad@SUBADEESH MINGW64 ~/OneDrive/Desktop/Smart_Agri_NodeMCU (master)
$ git merge feature_cloud
Auto-merging sensor.py
CONFLICT (content): Merge conflict in sensor.py
Automatic merge failed; fix conflicts and then commit the result.

subad@SUBADEESH MINGW64 ~/OneDrive/Desktop/Smart_Agri_NodeMCU (master|MERGING)
$ nano sensor.py
```

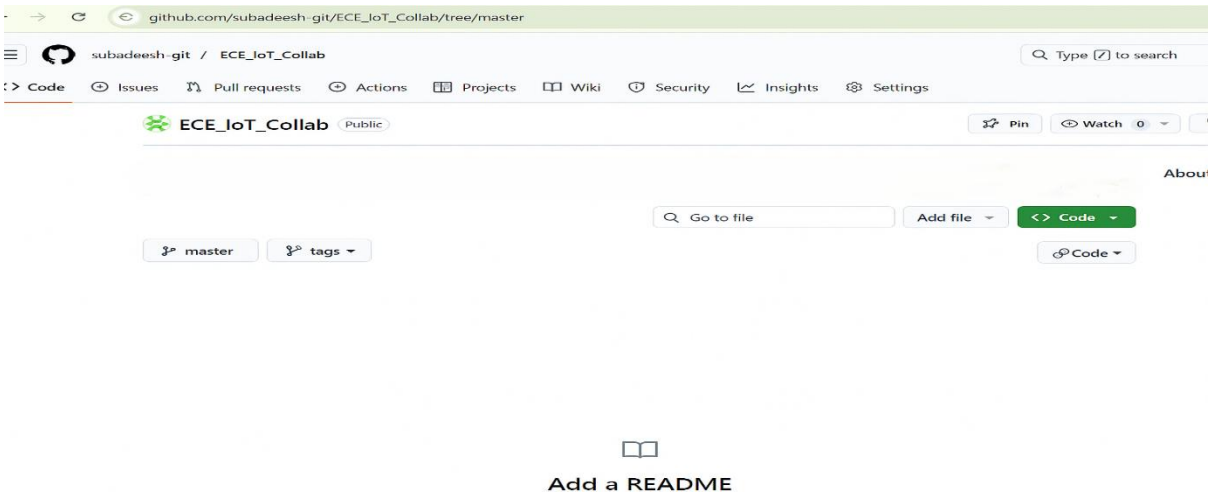


Step 13. Review the final commit history using a visual log to confirm that changes from both branches have been successfully integrated into the main branch.



GitHub Remote & Collaboration

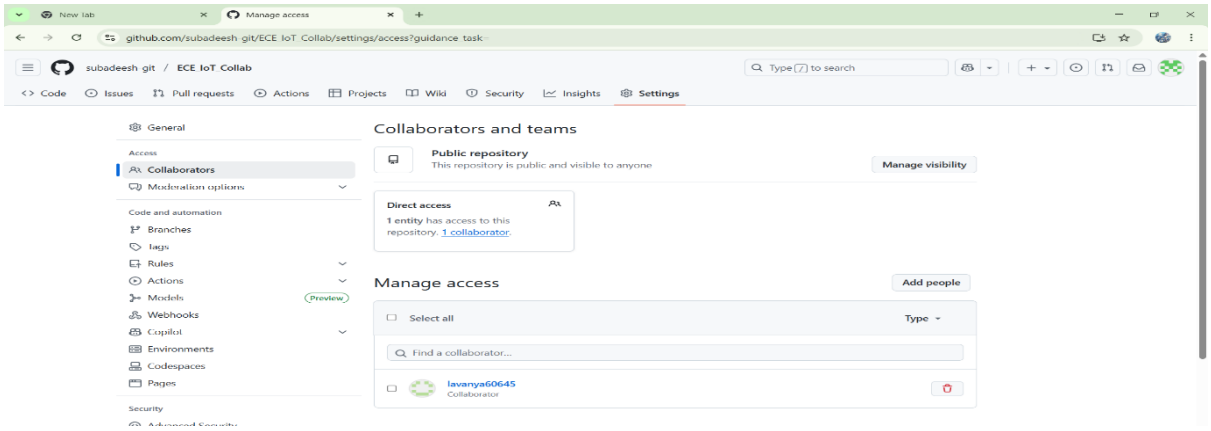
Step 14. A new remote repository named ECE_IoT_Collab was created on GitHub to host the local project and enable team collaboration.



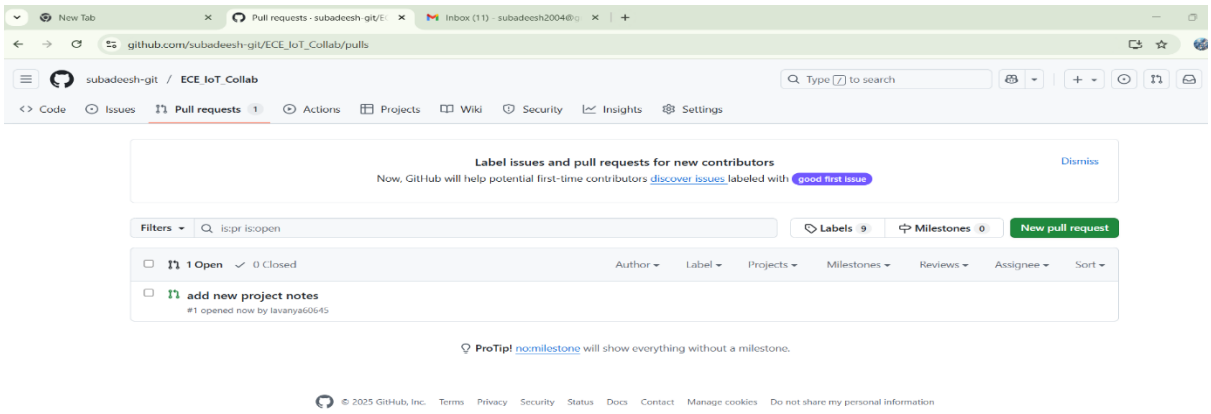
Step 15. The local Git repository was successfully linked to the remote GitHub repository, and all project files were pushed to the remote using Git.

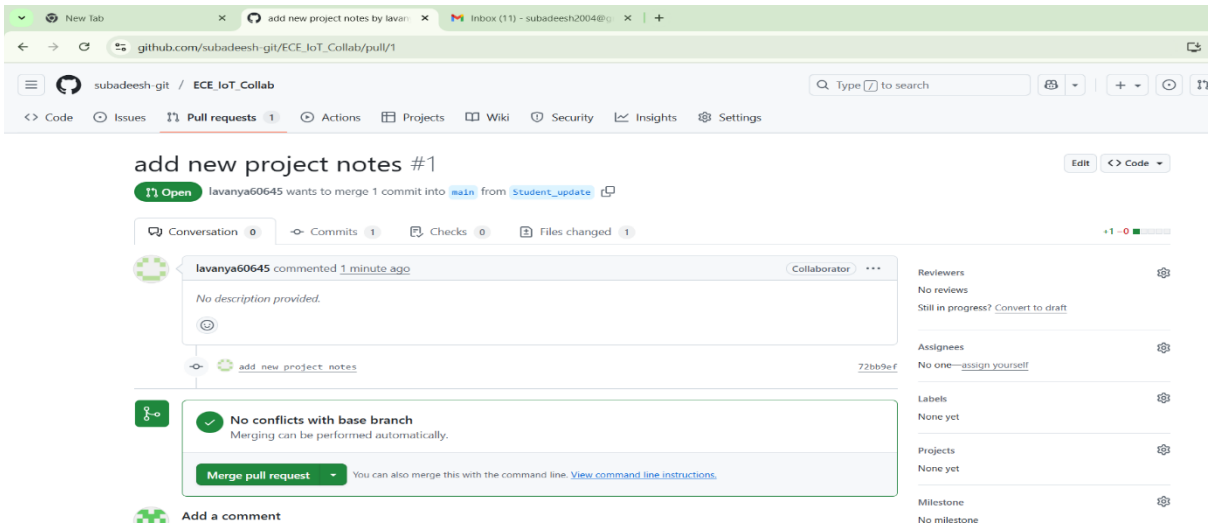
```
subad@SUBADEESH MINGW64 ~/OneDrive/Desktop/Smart_Agri_NodeMCU (master)
$ git remote add origin https://github.com/subadeesh-git/ECE_IoT_Collab.git

subad@SUBADEESH MINGW64 ~/OneDrive/Desktop/Smart_Agri_NodeMCU (master)
$ git push origin master
info: please complete authentication in your browser...
Enumerating objects: 19, done.
Counting objects: 100% (19/19), done.
Delta compression using up to 16 threads
Compressing objects: 100% (15/15), done.
Writing objects: 100% (19/19), 1.70 KiB | 347.00 KiB/s, done.
Total 19 (delta 3), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (3/3), done.
remote:
remote: Create a pull request for 'master' on GitHub by visiting:
remote:      https://github.com/subadeesh-git/ECE_IoT_Collab/pull/new/master
remote:
To https://github.com/subadeesh-git/ECE_IoT_Collab.git
 * [new branch]      master -> master
```

Step 16. A collaborator was added to the GitHub repository to contribute to the project through a dedicated feature branch.

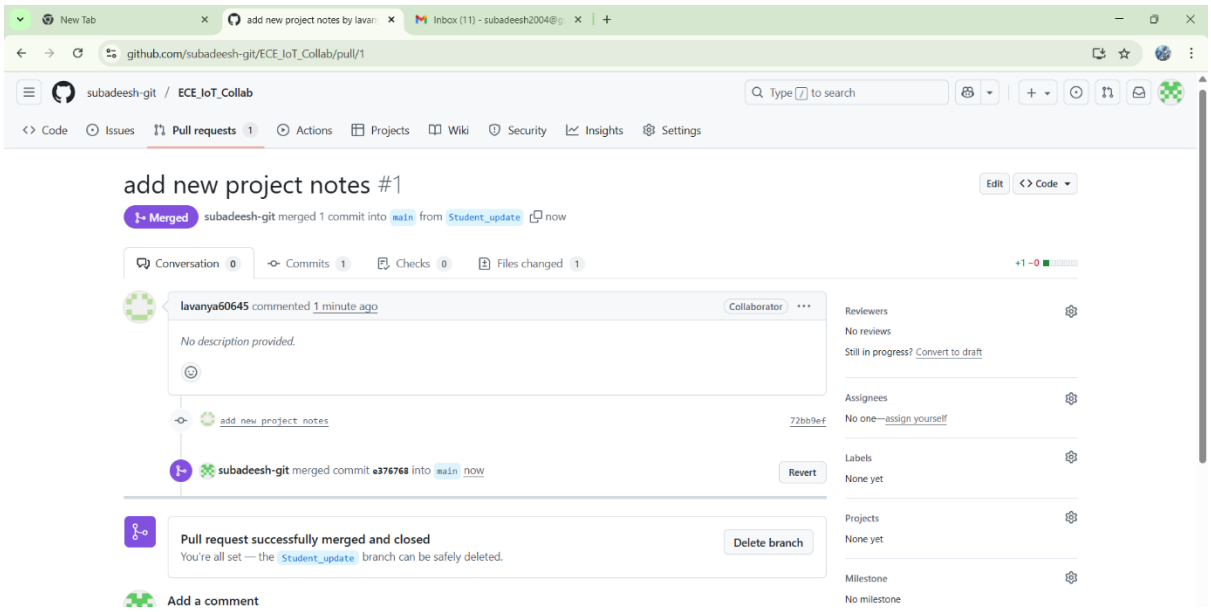


Step 17. The collaborator created a new branch named student_update, made edits to the project_notes.txt file, and submitted a Pull Request for review.





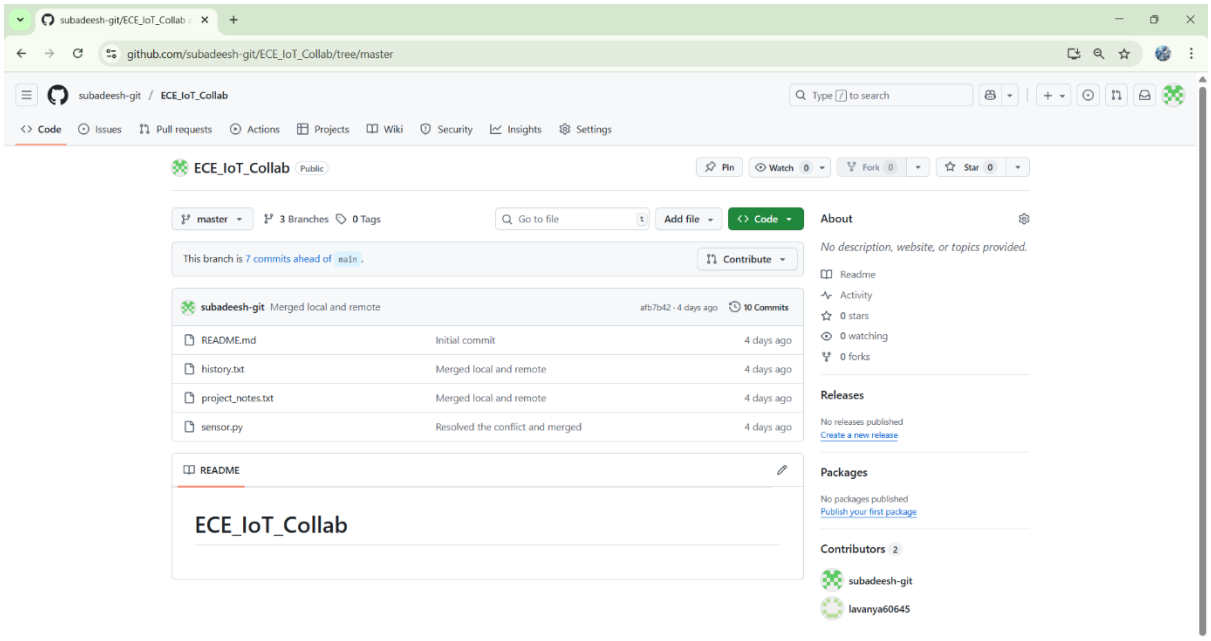
Step 18. The Pull Request was reviewed and merged into the main branch on GitHub, integrating the collaborator's changes into the project.



Step 19. The local repository was synchronized with the updated remote repository using the pull command to reflect the latest changes.

```
subad@SUBADEESH MINGW64 ~/OneDrive/Desktop/Smart_Agri_NodeMCU (master)
$ git push origin master
Enumerating objects: 8, done.
Counting objects: 100% (8/8), done.
Delta compression using up to 16 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (4/4), 561 bytes | 561.00 KiB/s, done.
Total 4 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/subadeesh-git/ECE_IoT_Collab.git
   c6eea1d..afb7b42  master -> master

subad@SUBADEESH MINGW64 ~/OneDrive/Desktop/Smart_Agri_NodeMCU (master)
$ git pull origin master
From https://github.com/subadeesh-git/ECE_IoT_Collab
 * branch            master     -> FETCH_HEAD
Already up to date.

subad@SUBADEESH MINGW64 ~/OneDrive/Desktop/Smart_Agri_NodeMCU (master)
$ git pull origin main
From https://github.com/subadeesh-git/ECE_IoT_Collab
 * branch            main       -> FETCH_HEAD
Already up to date.
```

Conclusion:



This mini project has successfully demonstrated the practical application of Git-based DevOps workflows in an IoT development context. The structured approach ensures reproducibility, collaboration, and long-term maintainability of the project assets.

Repository Access: The complete report, along with all relevant files and documentation, has been uploaded to the project's GitHub repository. For reference and further review, the repository can be accessed via the following link:

https://github.com/subadeesh-git/ECE_IoT_Collab.git