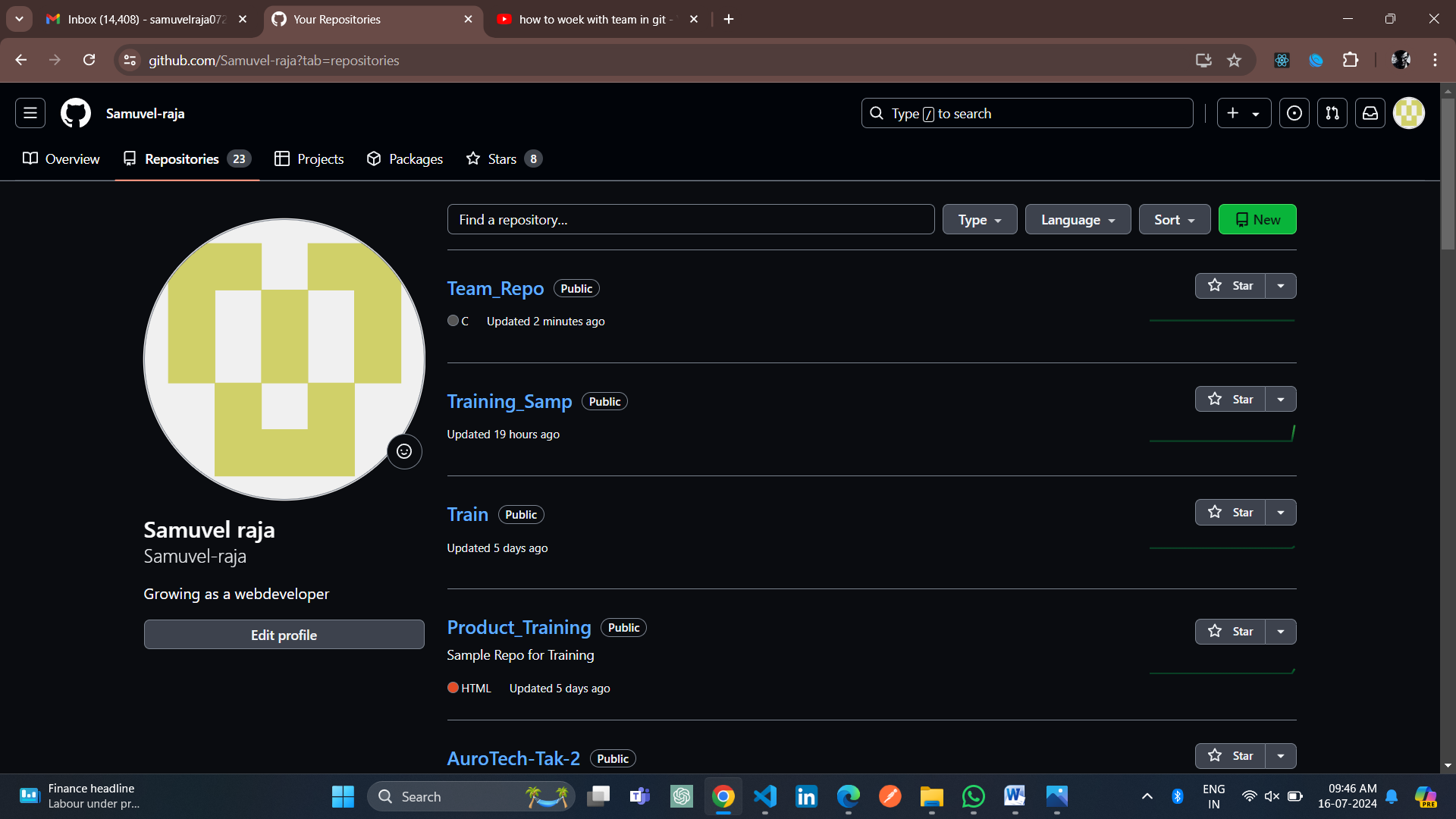
**Product Training Assignment 3**

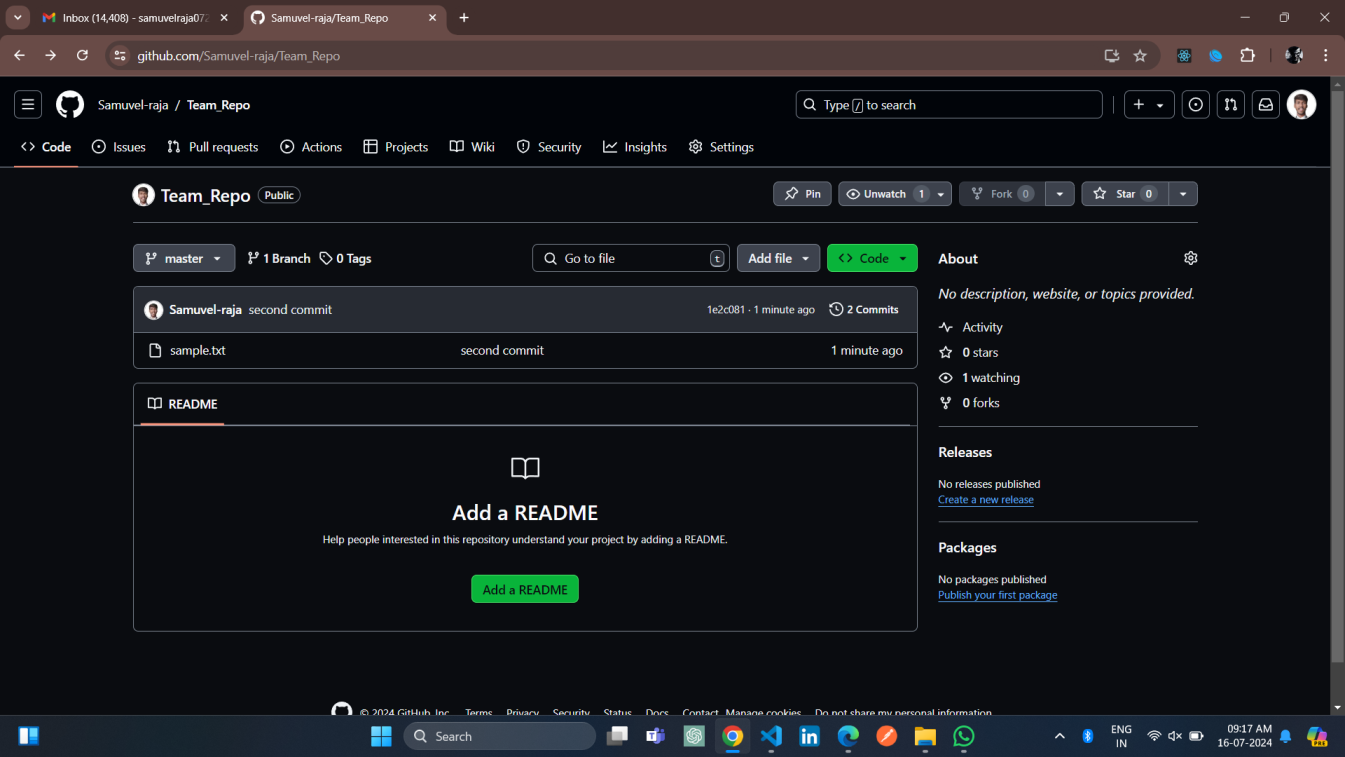
**Step 1:**

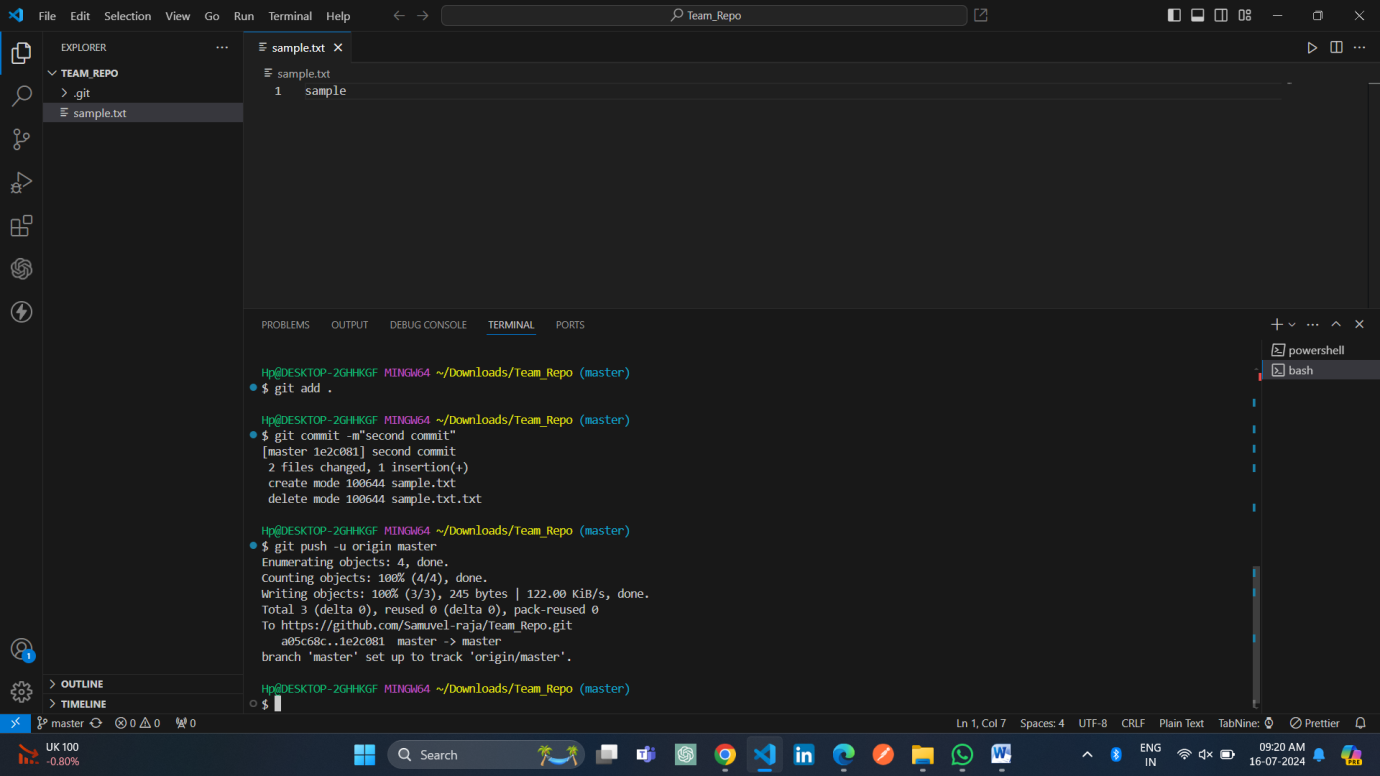
First, one person in the group should create a public repository using their GitHub account.



**Step 2:**

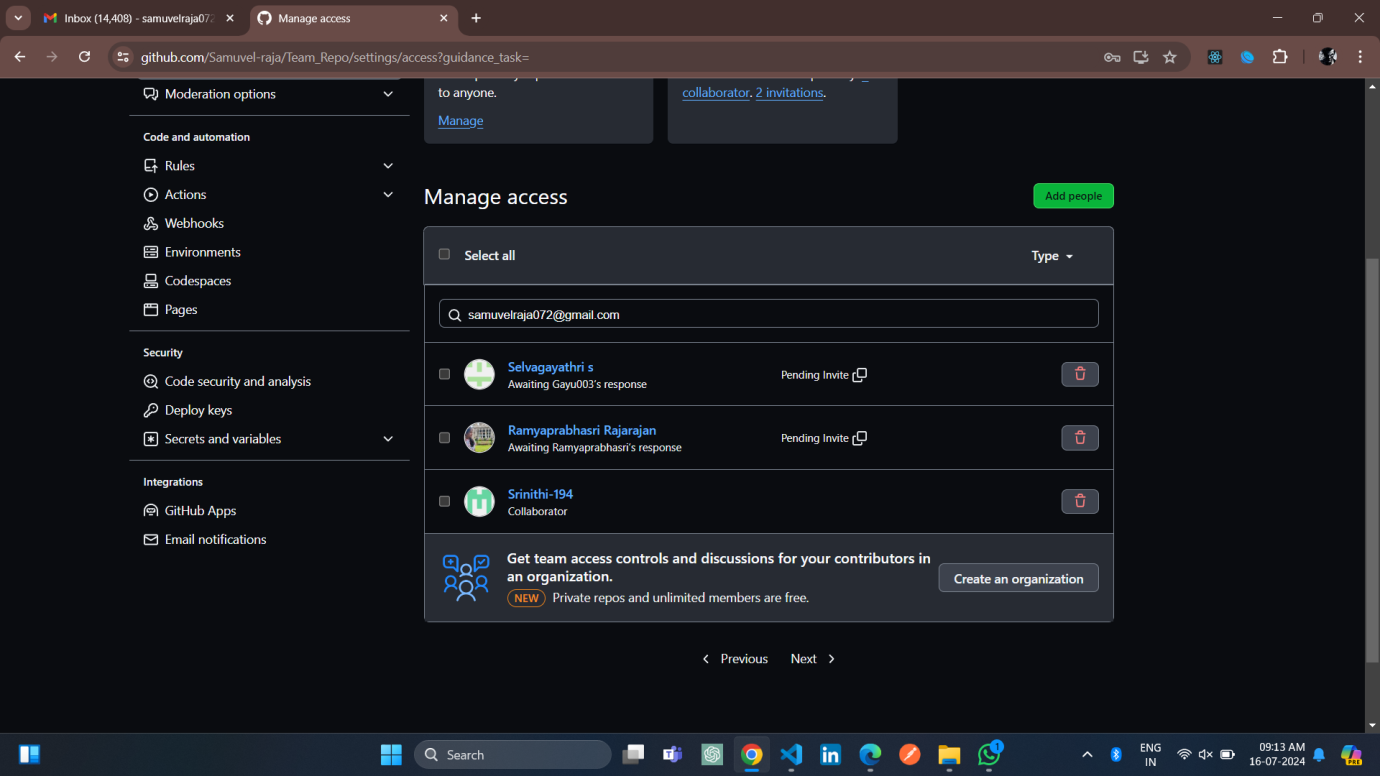
This same person should then follow the instructions from GitHub to add a remote, and then push their repository. Do not forget the –u flag, as suggested by GitHub!

****

****

**Step 3:**

All of the other members of the group should then be added as collaborators, so they can commit to the repository also.

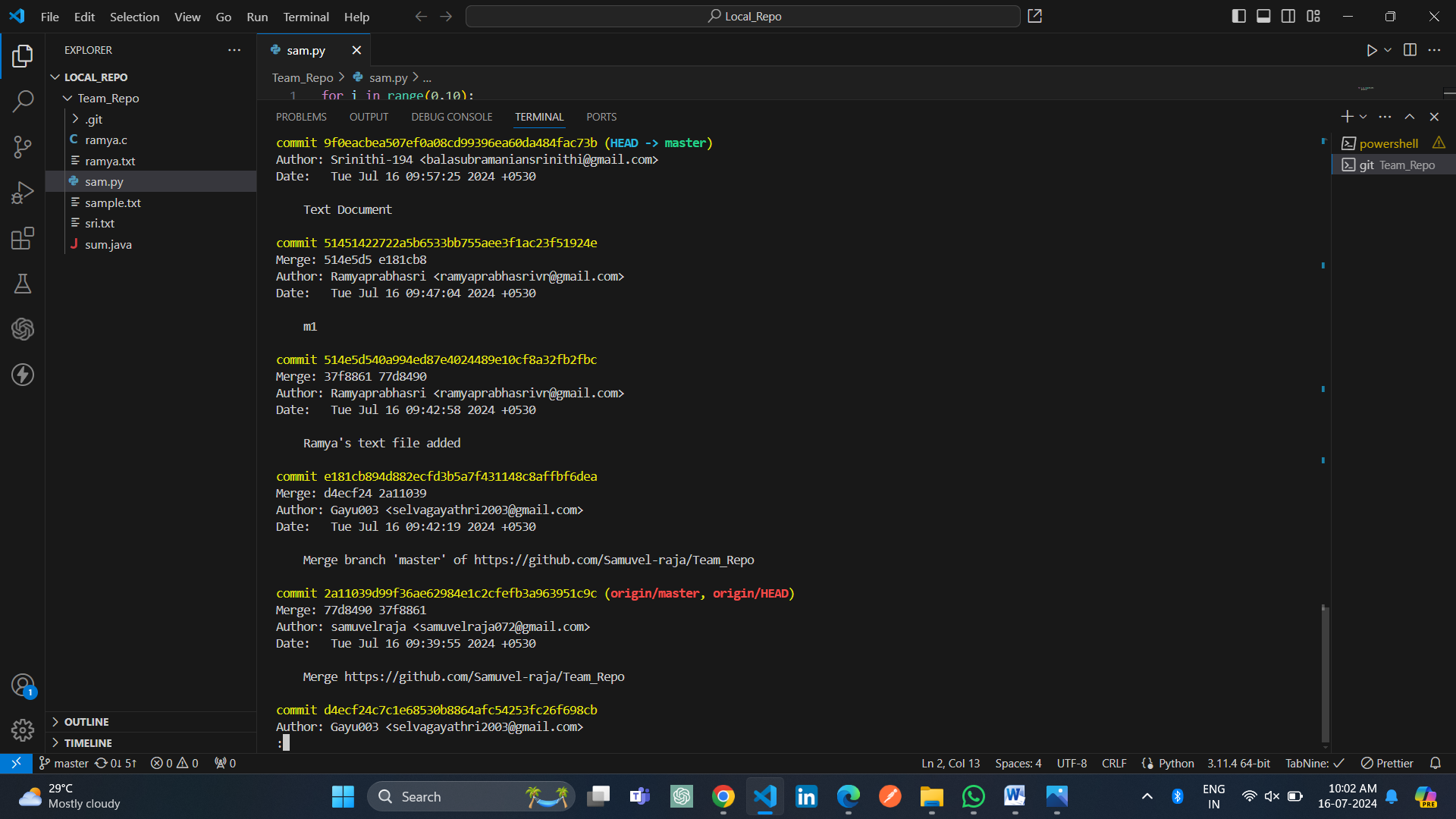
****

**Step 4:**

Next, everyone else in the group should clone the repository from GitHub. Verify that the context of the repository is what is expected. One of the group members who just cloned should now make a local commit, then push it. Everyone should verify that when they pull, that commit is added to their local repository (use git log to check for it). Look at each other’s git log output.

**Qn:** Notice how the SHA-1 is the same for a given commit across every copy of the repository. Why is this important?

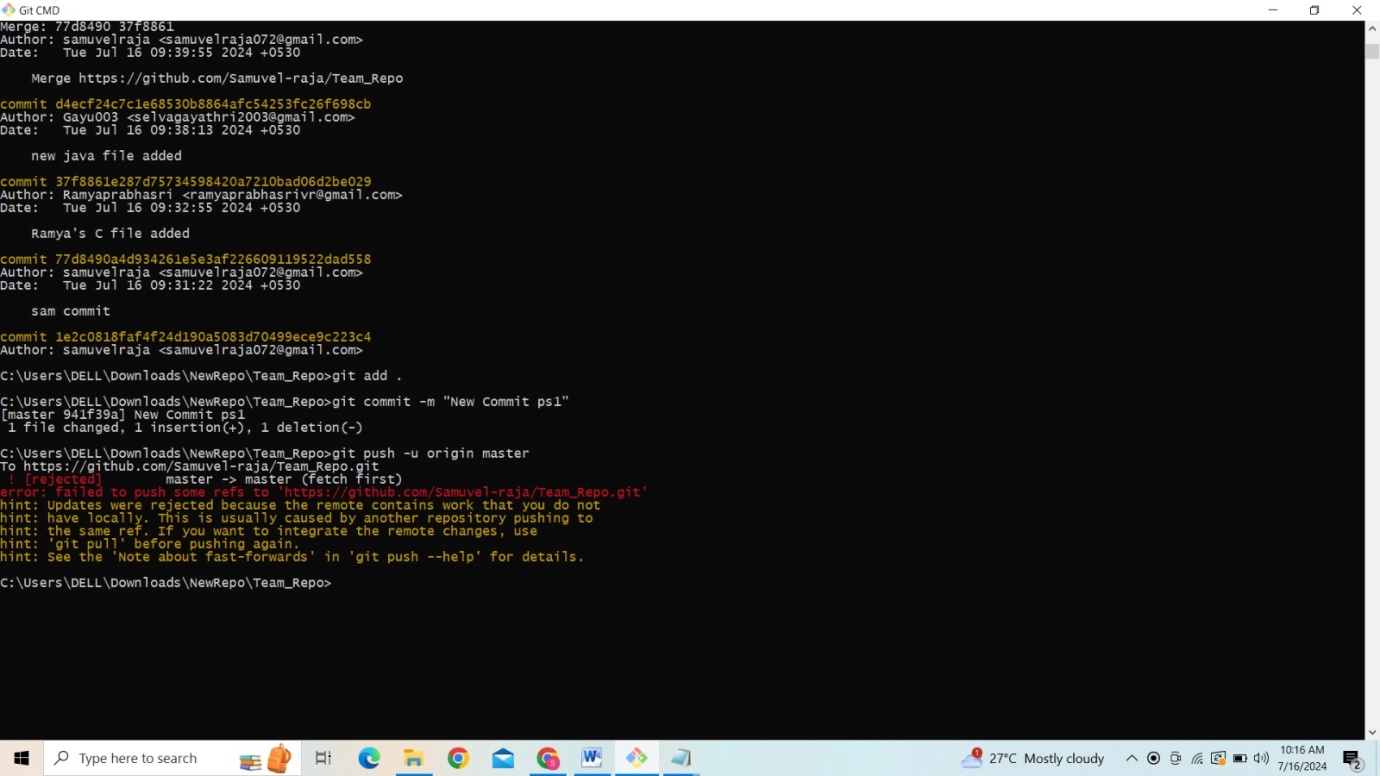
**Ans:** The SHA-1 hash allows for consistency across these distributed copies. Since the hash is based on the commit content, the same commit will have the same hash in every clone of the repository, ensuring that everyone is referring to the exact same data.



**Step 5:**

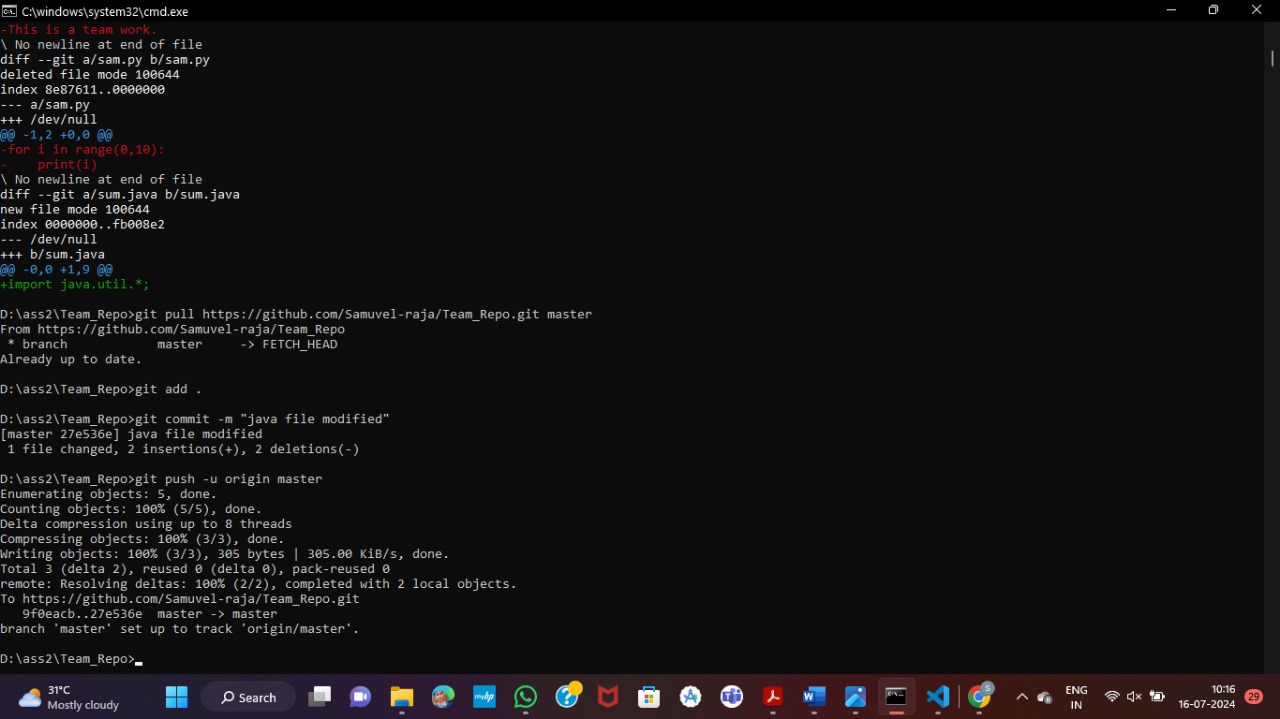
Two members of the group should now make a commit locally, and race to push it. To keep things simple, be sure to edit different files. What happens to the runner-up.

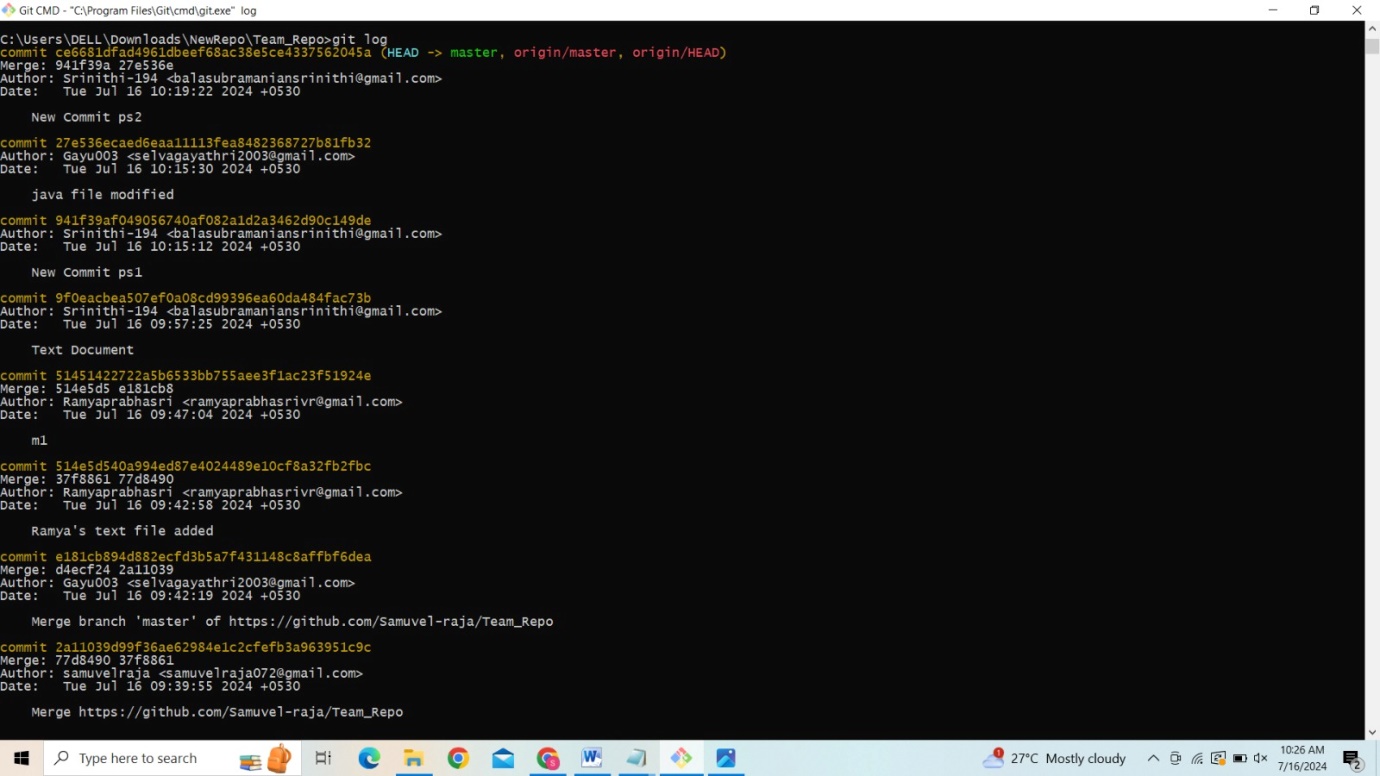
Ans: The runner-up's push will be rejected because their local repository is out of sync with the remote. They will need to pull the latest changes from the remote repository, merge them with their local changes, and then push again.

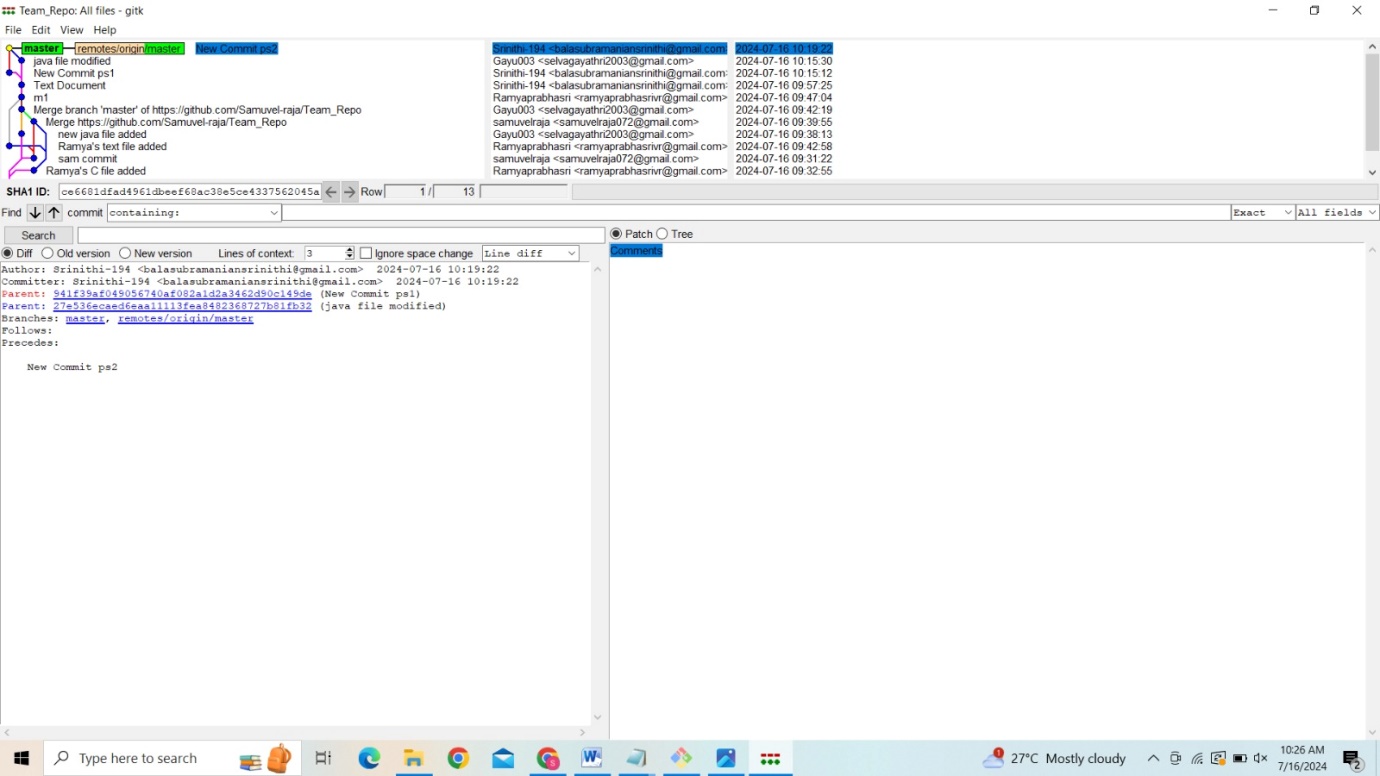


**Step 6:**

The runner-up should now pull. As a group, look at the output of the command. Additionally, look at the git log, and notice that there is a merge commit. You may also wish to view the DAG in gitk.

****

****

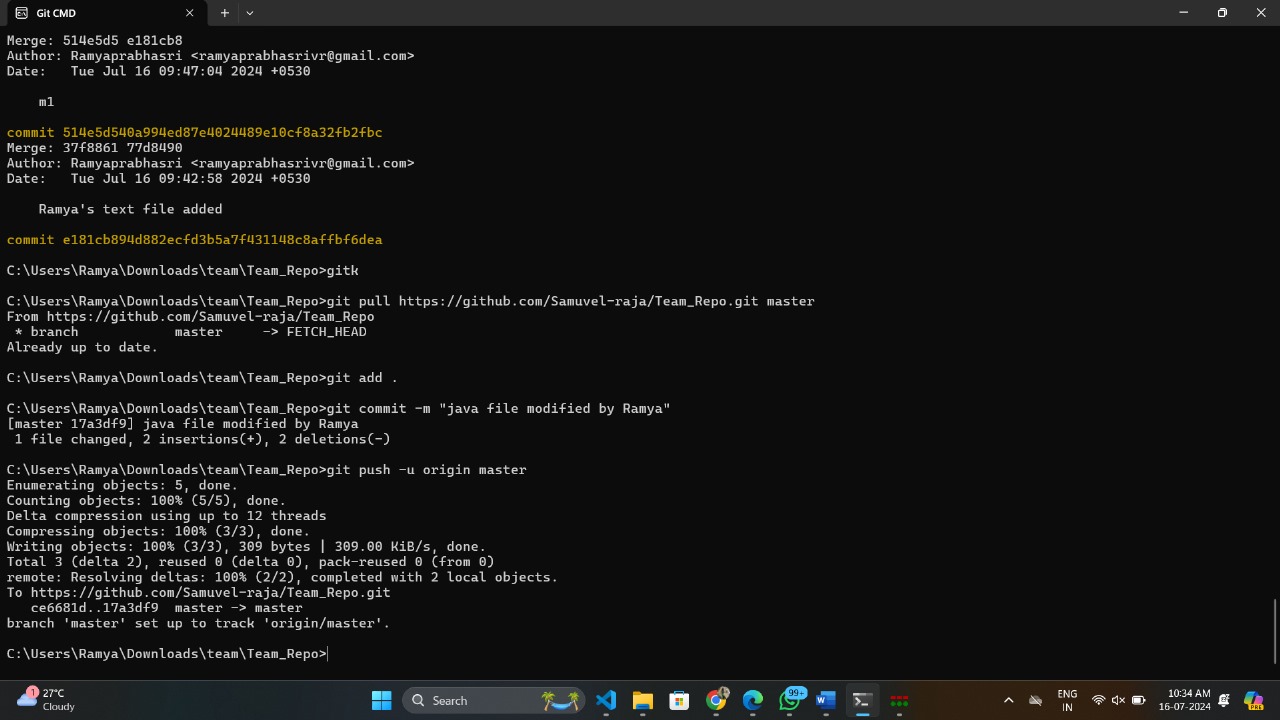
****

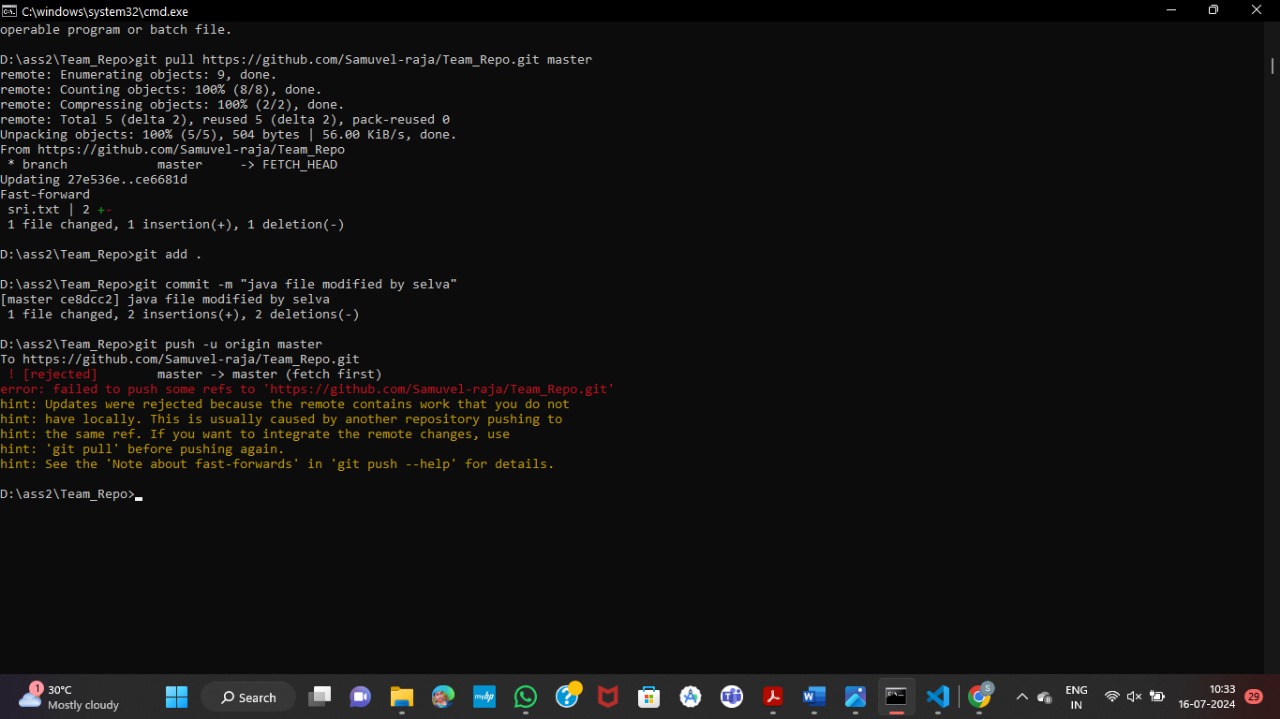
**Stretch Task:**

**Step 1:**

Now create a situation where two group members both edit the same line in the same file and commit it locally. Race to push

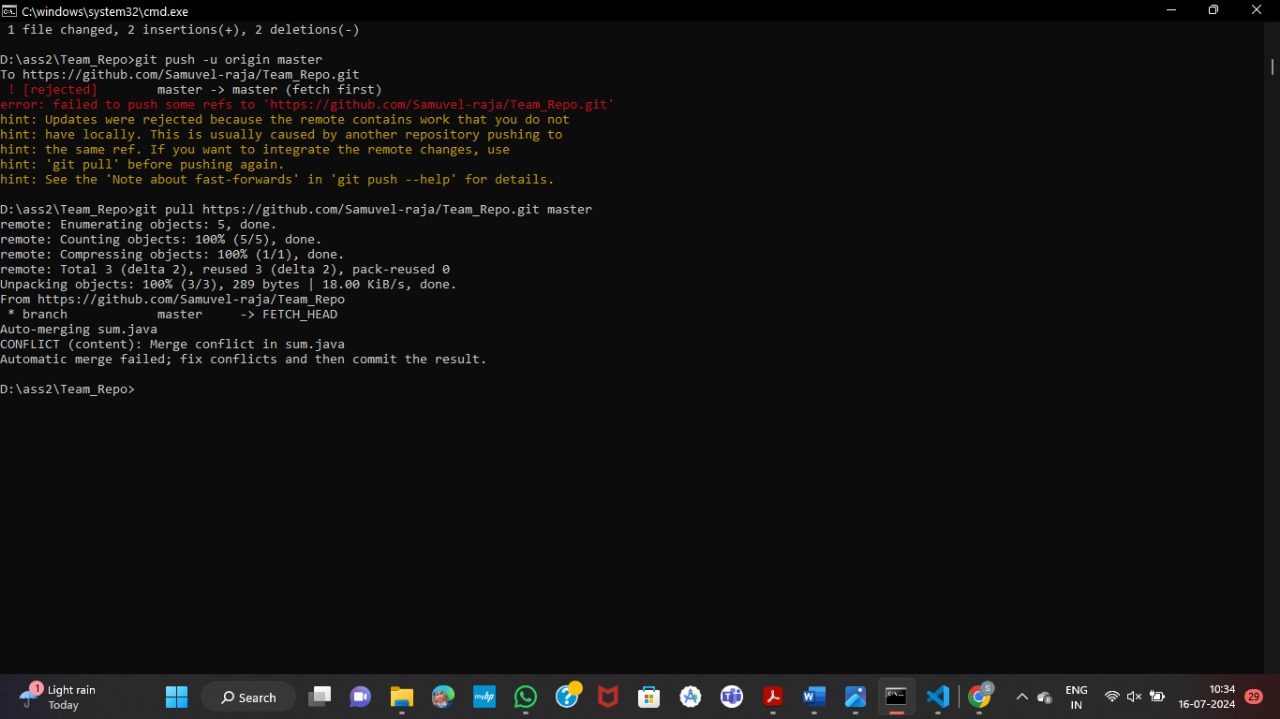
Ans: When two group members edit the same line in the same file and commit it locally, the first member to push their changes will succeed. The second member's push will be rejected because their local repository is out of sync with the remote. The runner-up will need to pull the latest changes from the remote, resolve the merge conflict manually, commit the resolved changes, and then push again.

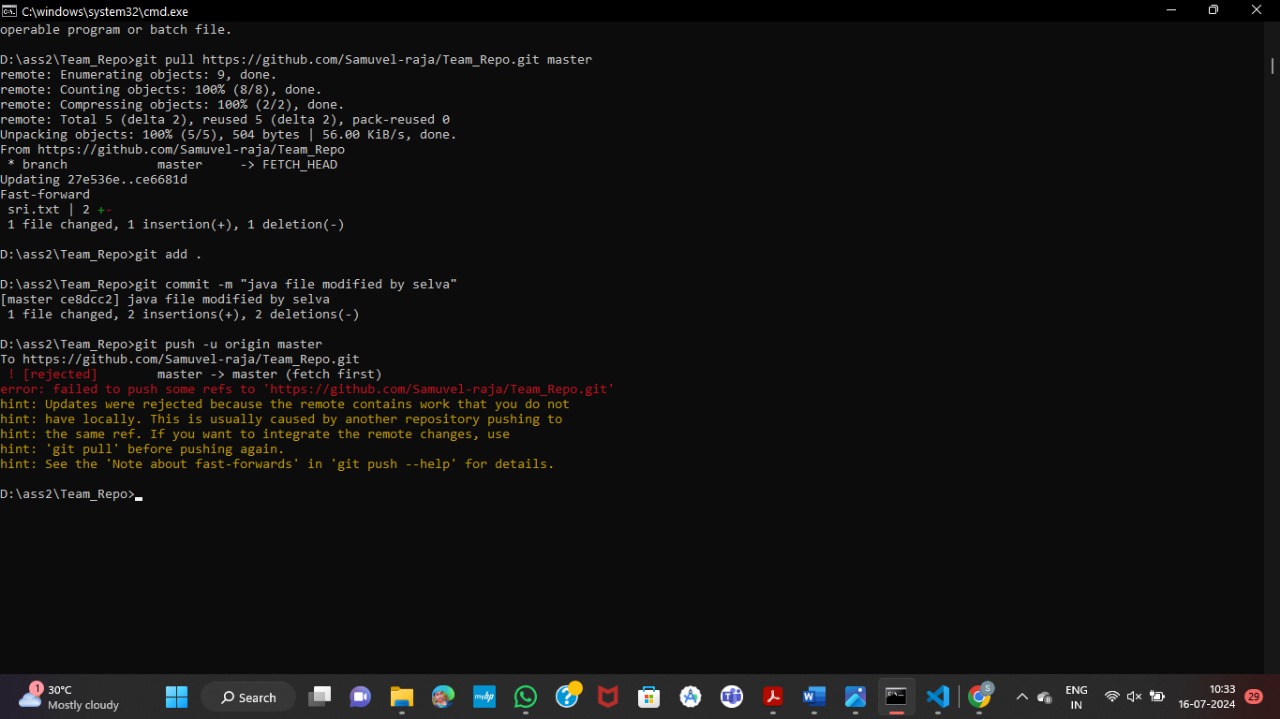




**Step 2:**

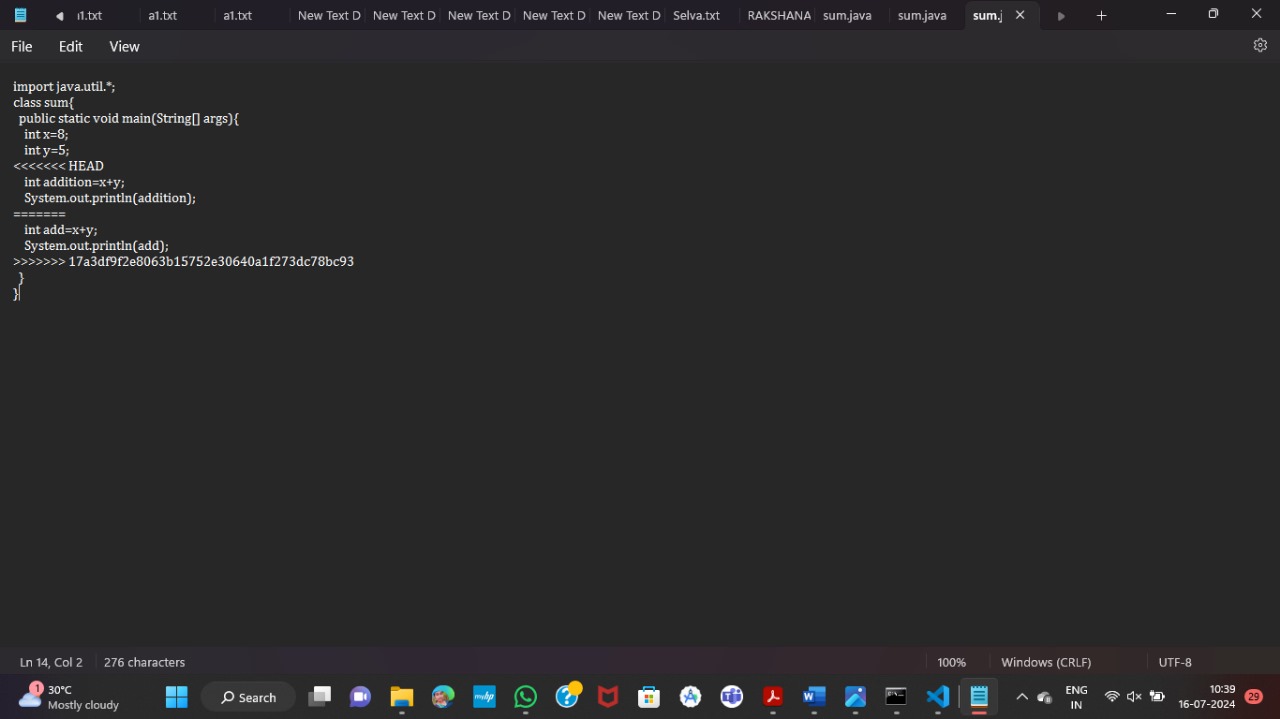
When the runner-up does a pull, they should get a merge conflict.

****

****

**Step 3:**

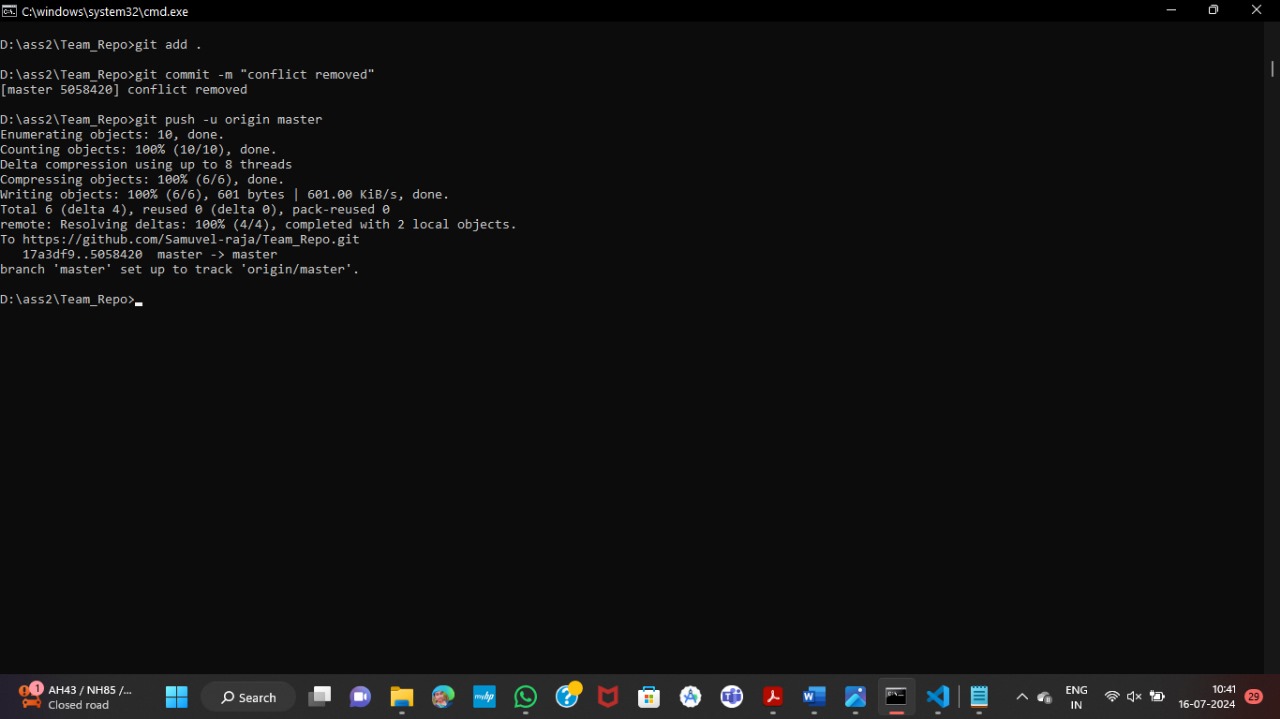
Look as a group at the file in conflict, and resolve it.

****

**Step 4:**

Use the add command to stage the fix, and then use commit to make the merge commit. Notice how this procedure is exactly the one you got used to when resolving conflicts in branches.

****

****