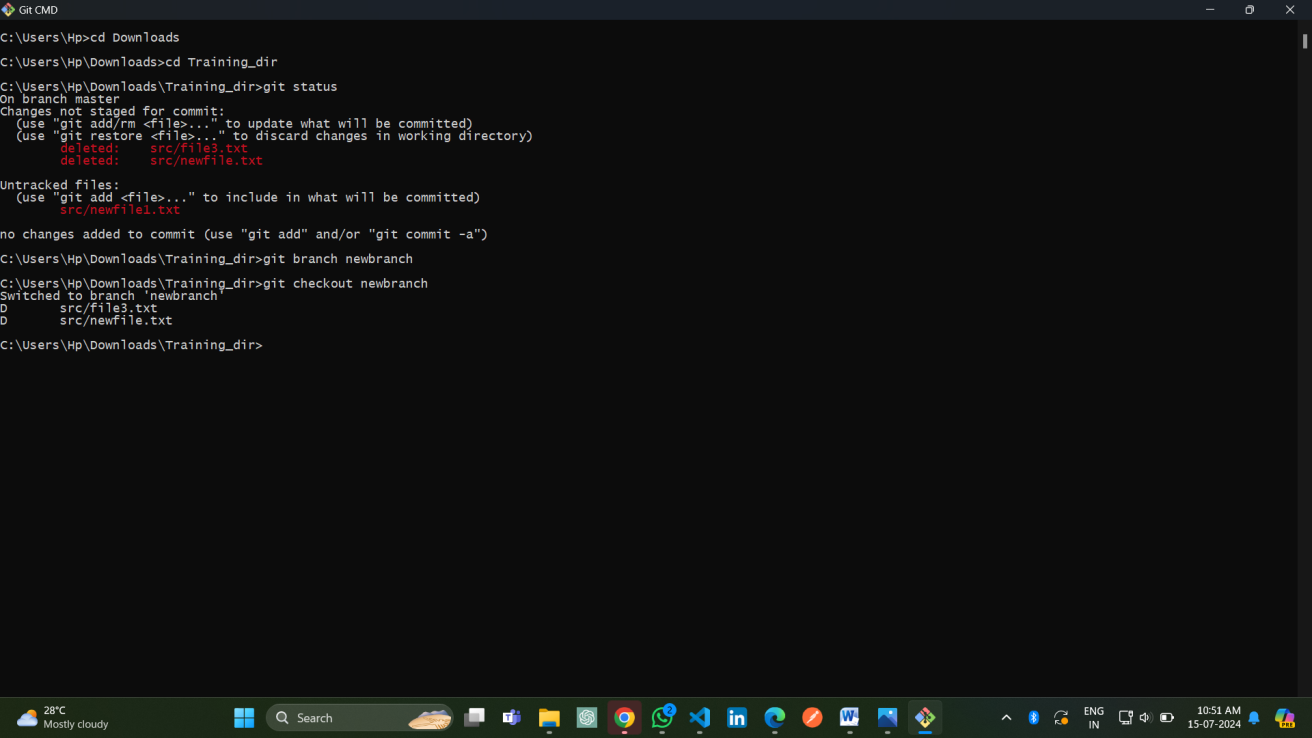
**Product Training Assignment 2**

**Exercise 2**

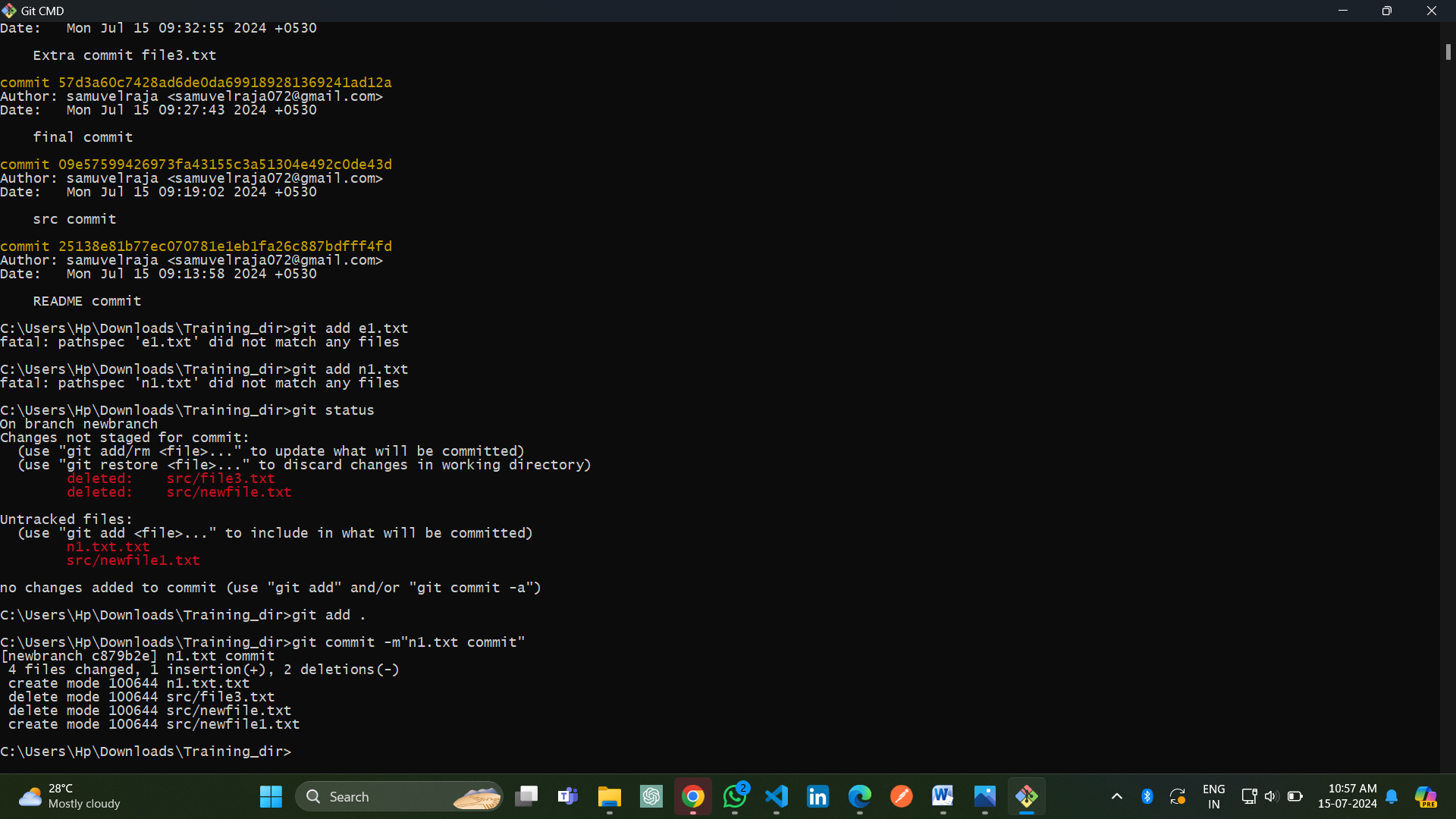
**Step 1:**

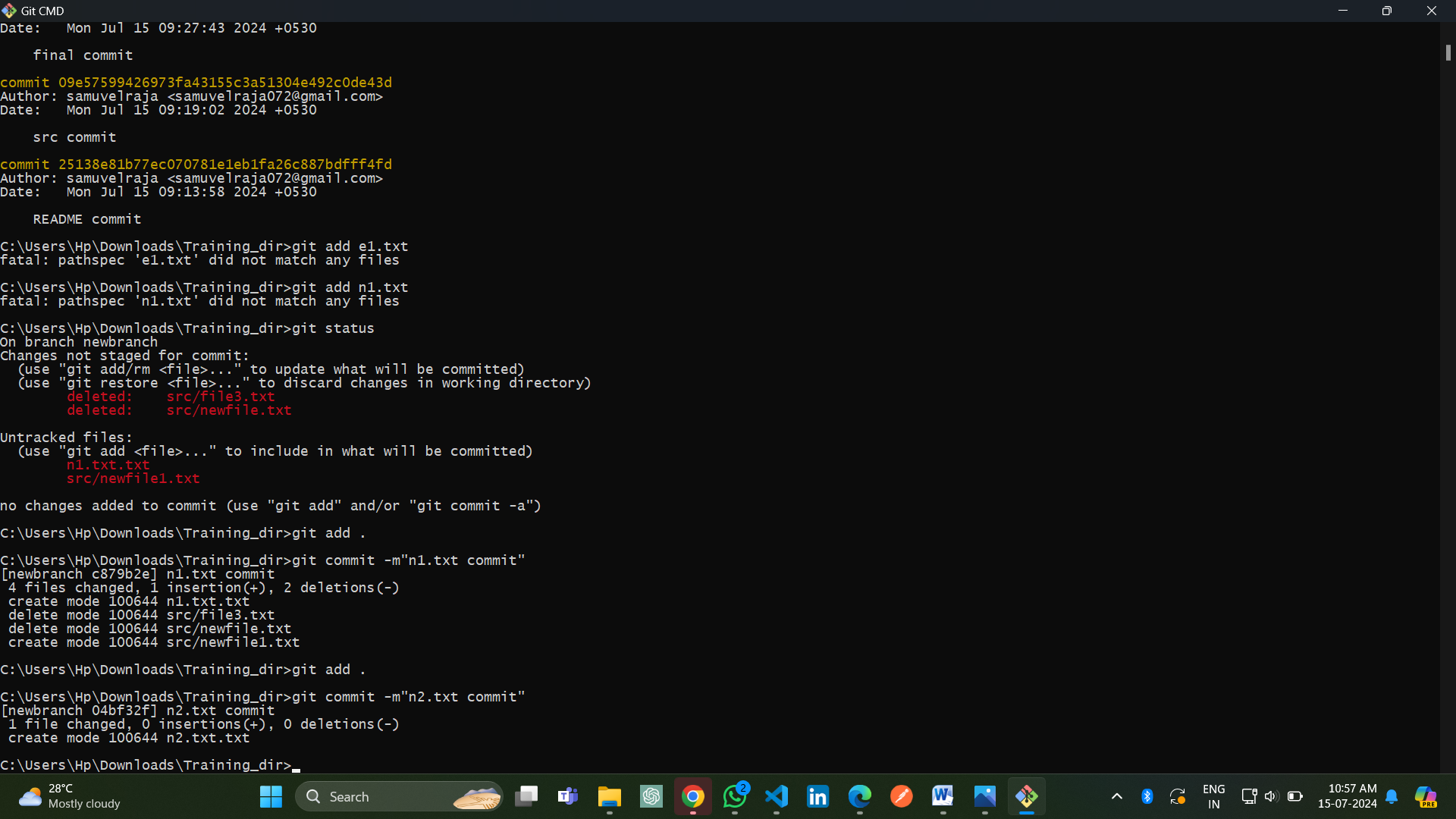
Run the status command. Notice how it tells you what branch you are in. Use the branch command to create a new branch. Use the checkout command to switch to it.

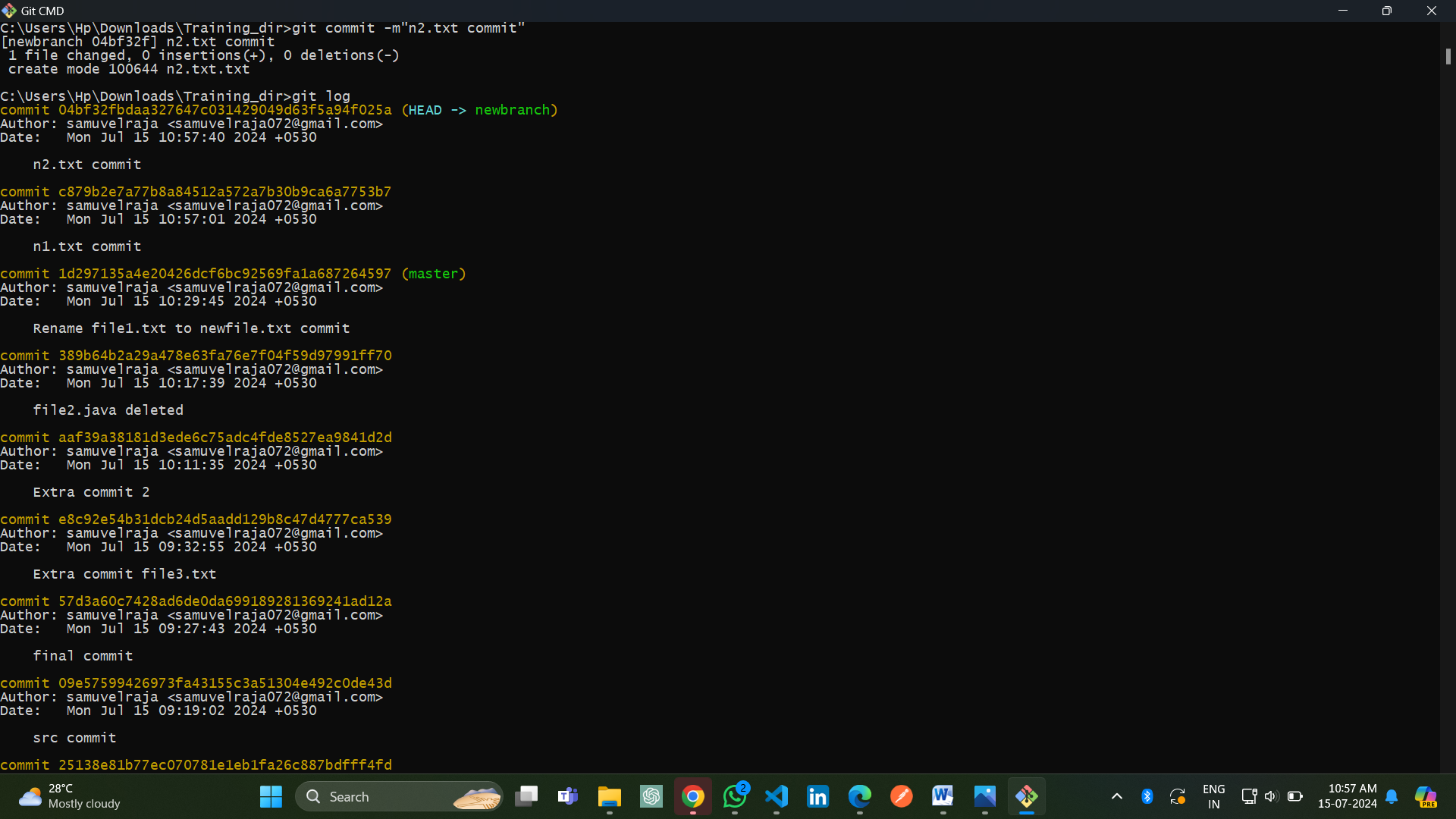
****

**Step 2:**

Make a couple of commits in the branch – perhaps adding a new file and/or editing existing ones. Use the log command to see the latest commits. The two you just made should be at the top of the list.

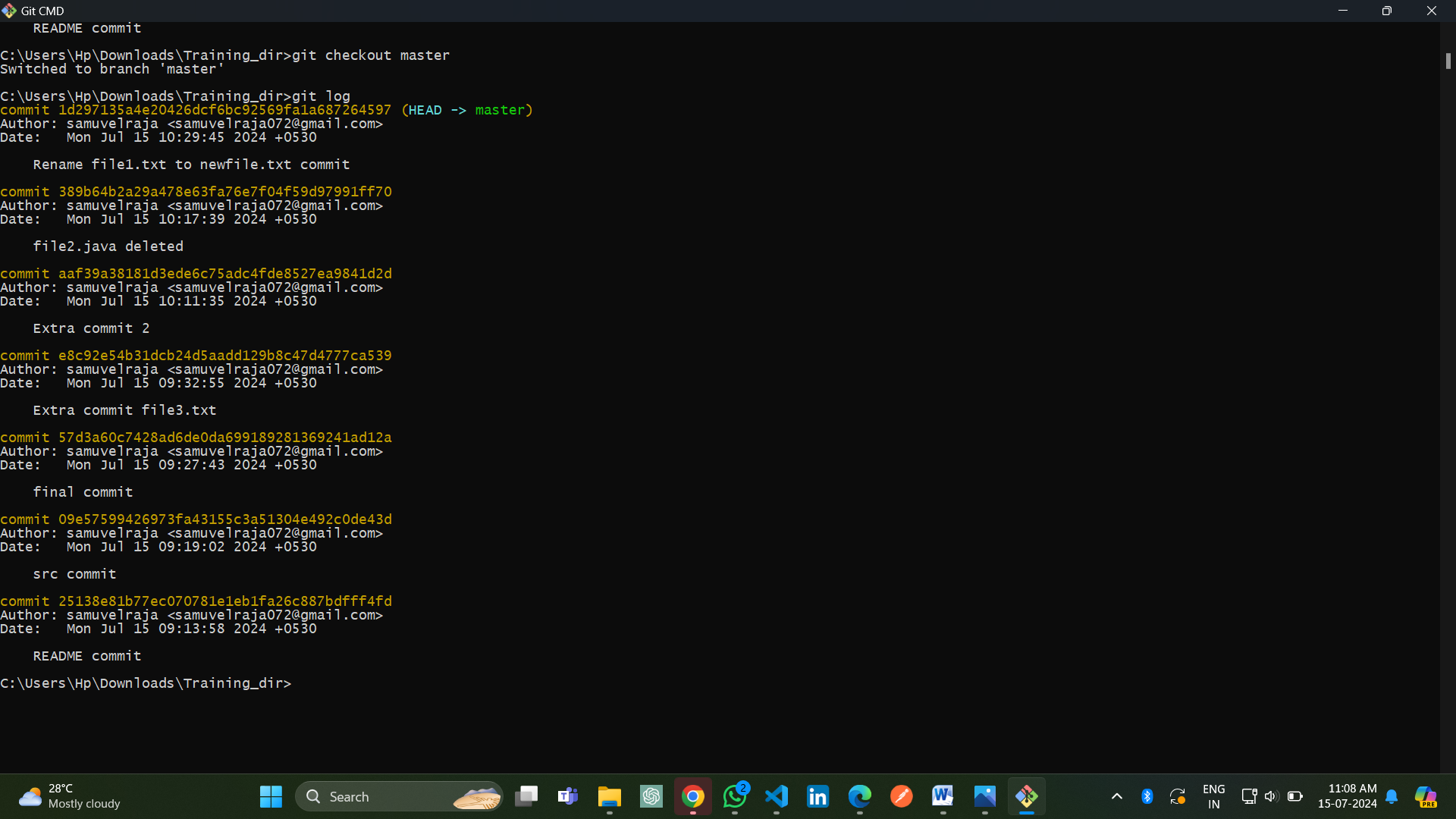






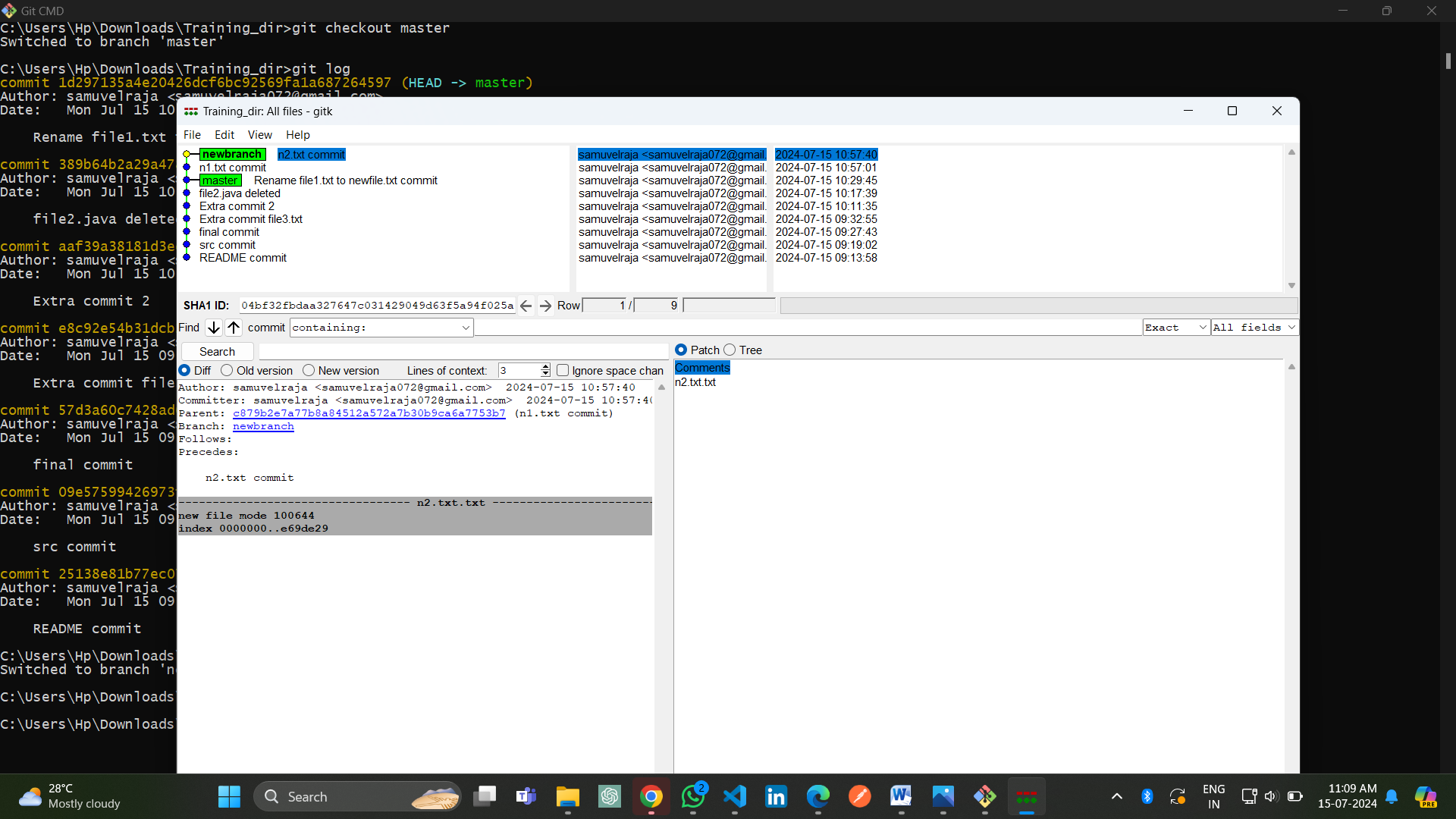
**Step 3:**

Use the checkout command to switch back to the master branch. Run log again. Notice your commits don’t show up now. Check the files also – they should have their original contents.



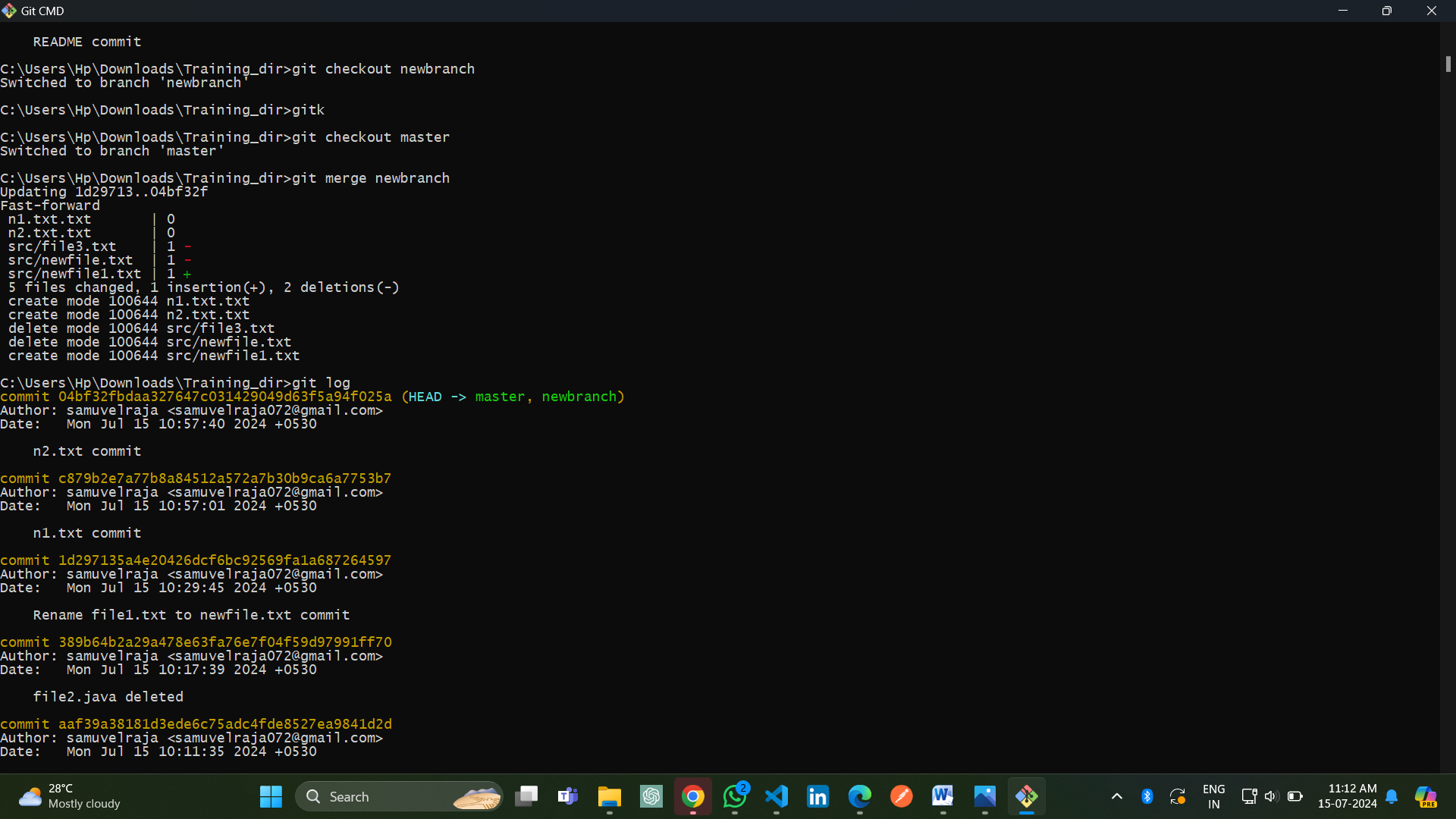
**Step 4:**

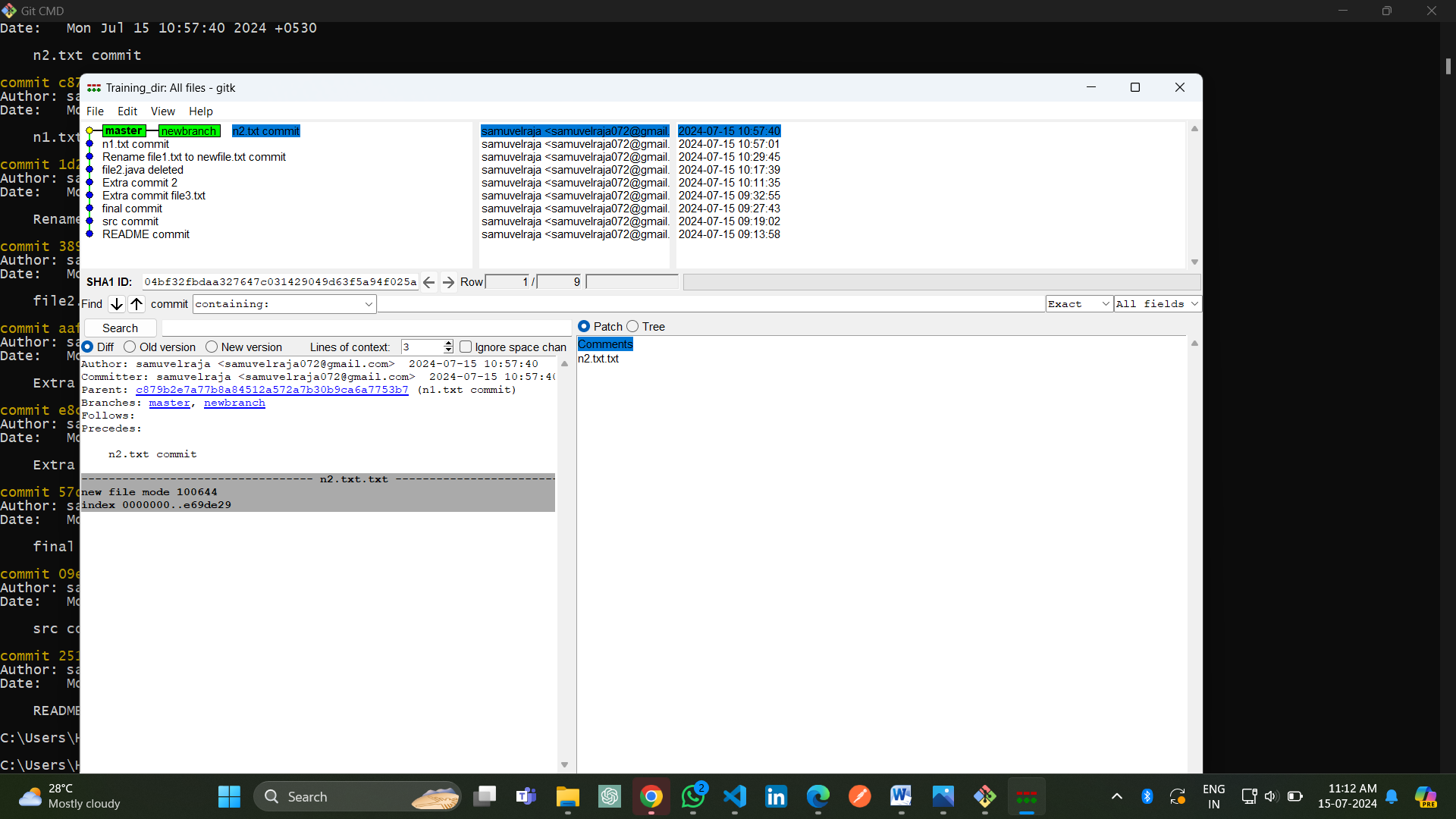
Use the checkout command to switch back to your branch. Use gitk to take a look at the commit graph; notice it’s linear.

****

**Step 5:**

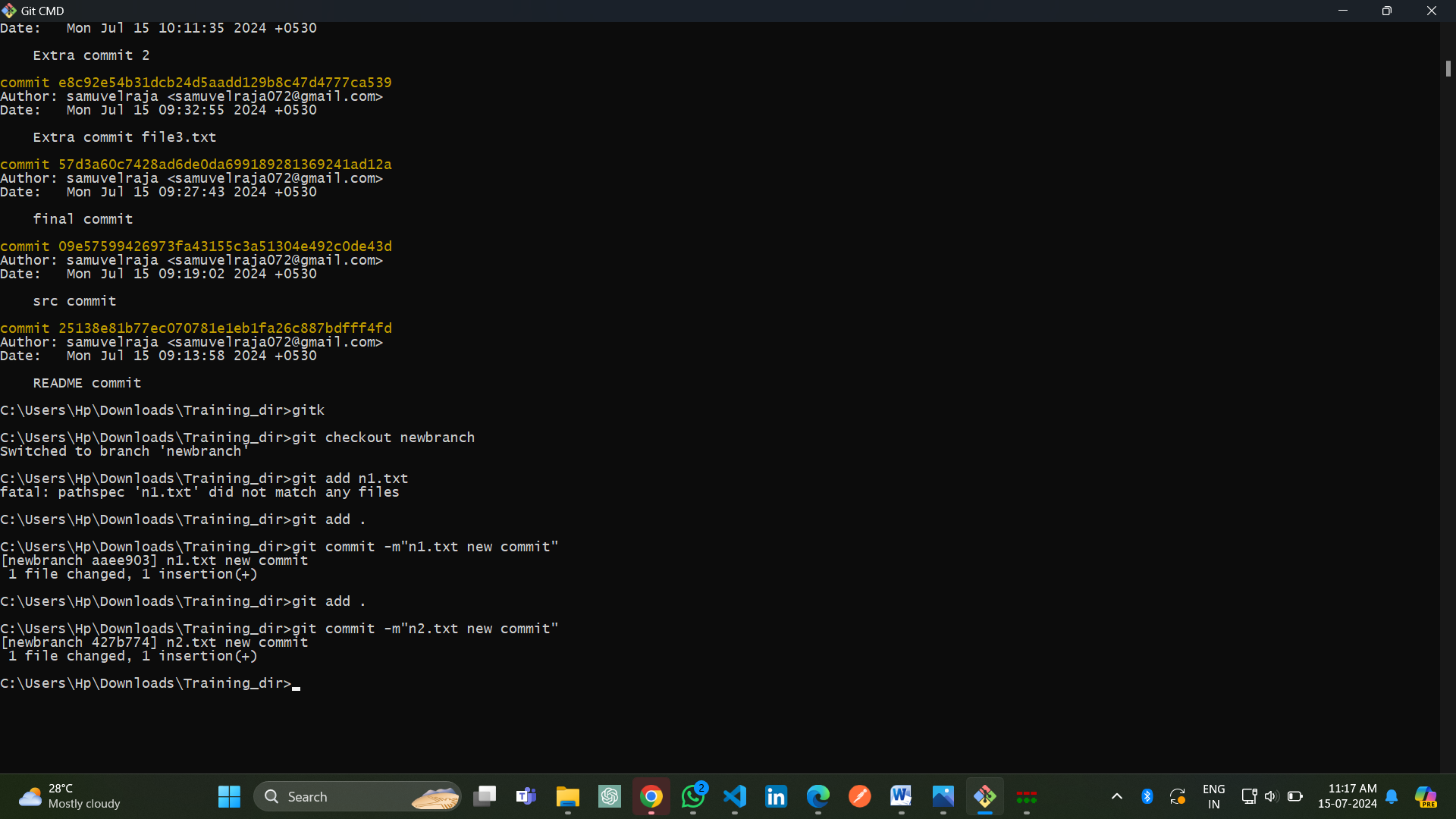
Now checkout the master branch again. Use the merge command to merge your branch in to it. Look for information about it having been a fast-forward merge. Look at git log, and see that there is no merge commit. Take a look in gitk and see how the DAG is linear.

****

****

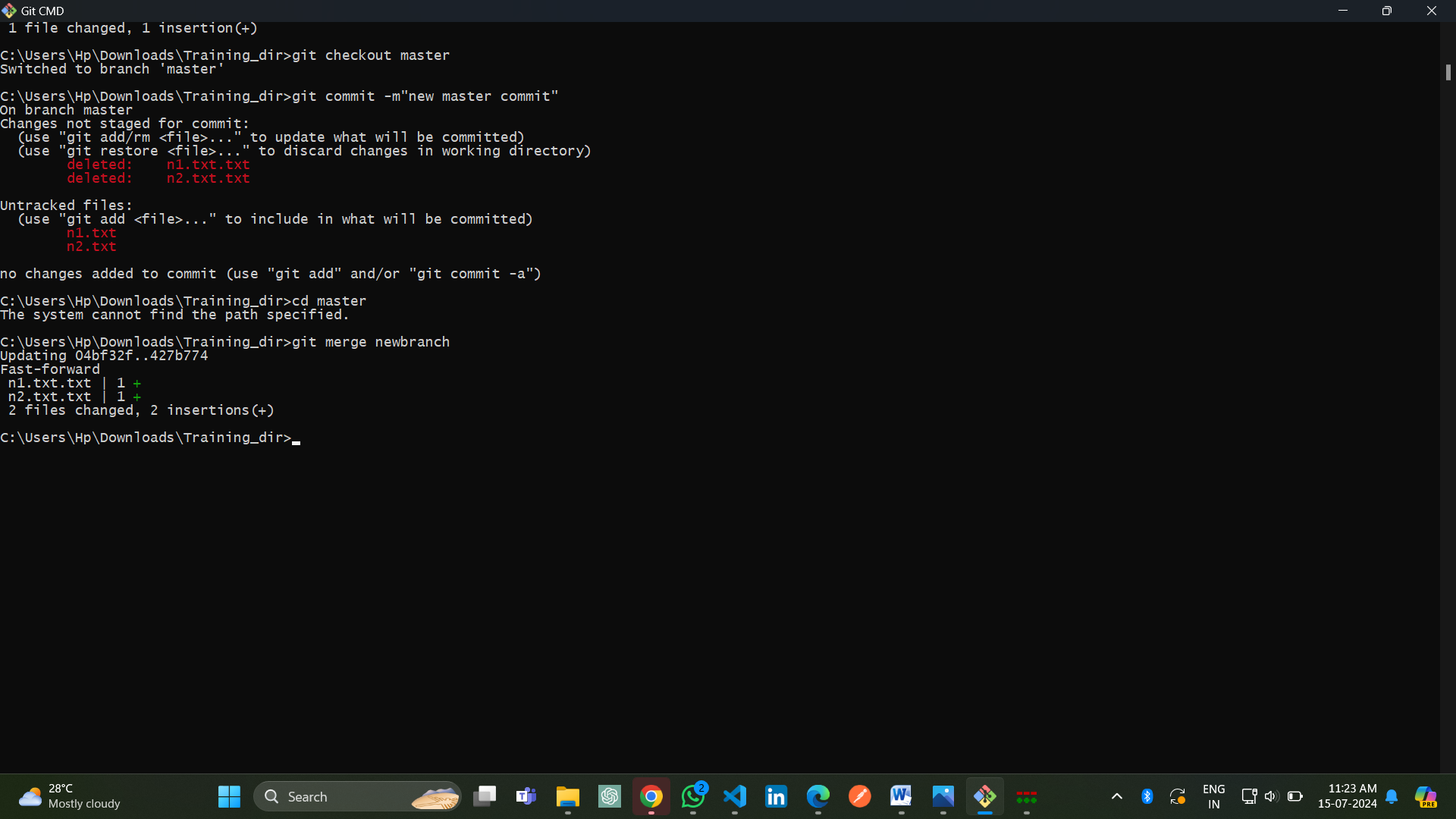
**Step 6:**

Switch back to your branch. Make a couple more commits.

****

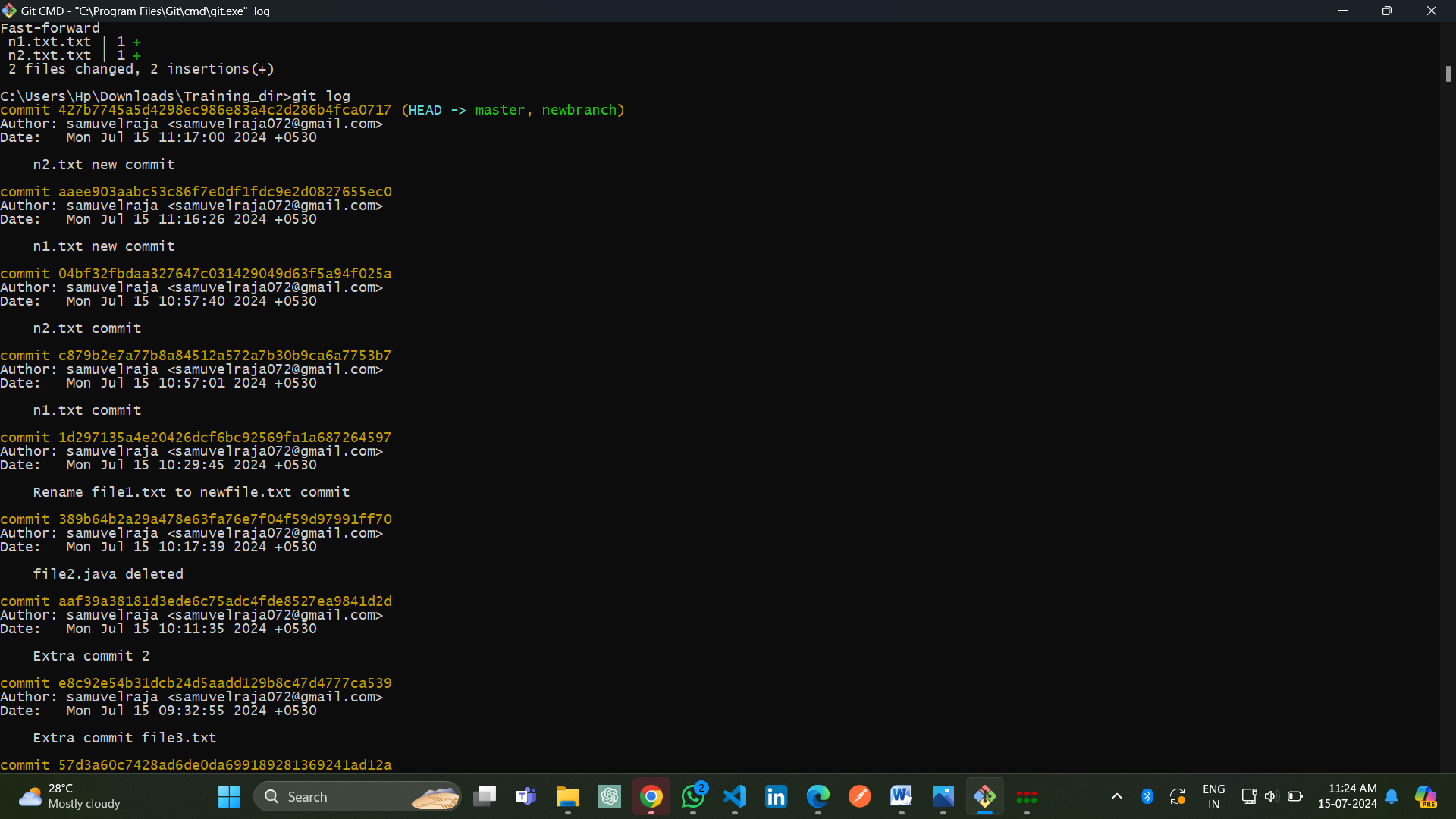
**Step 7:**

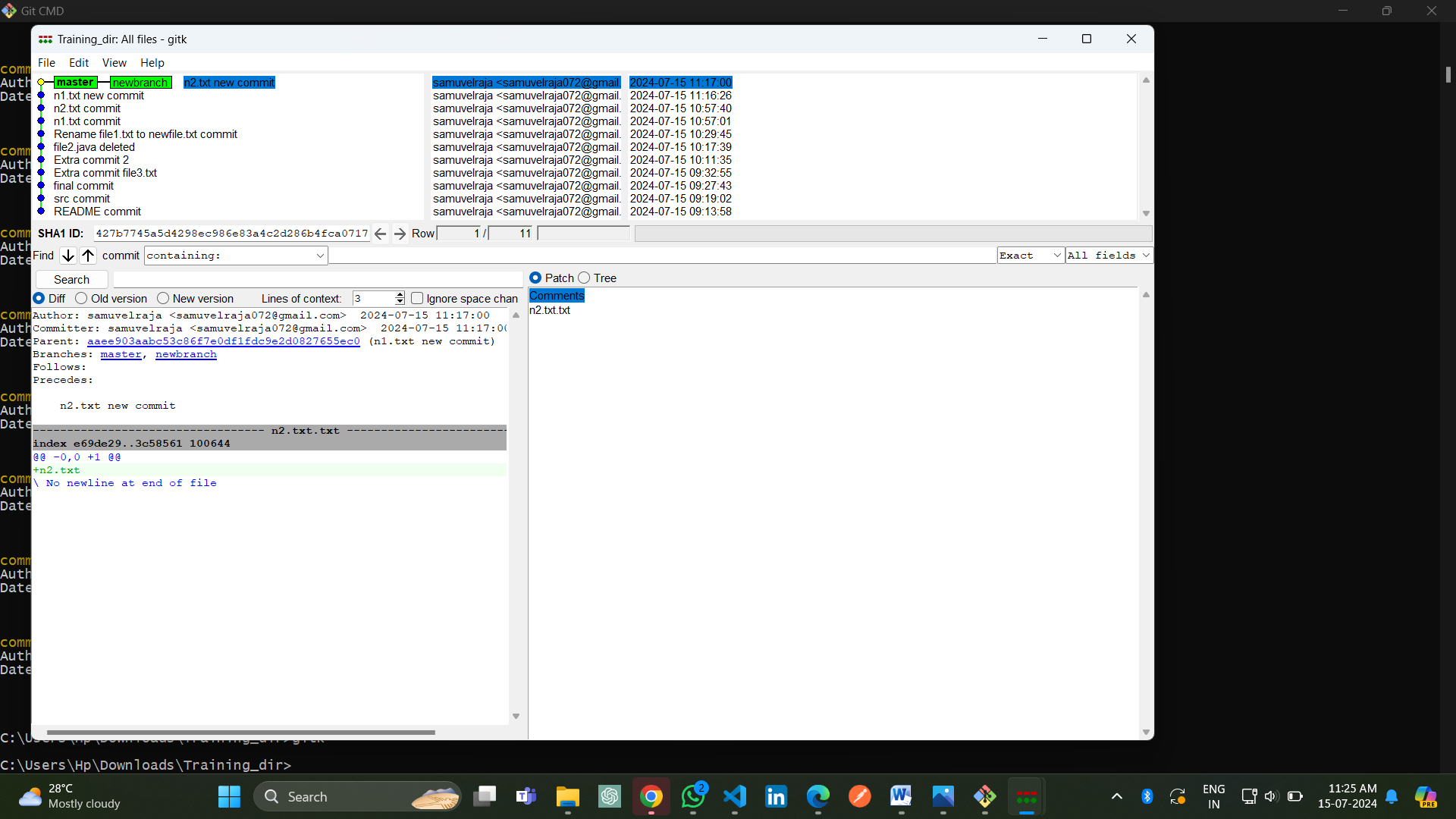
Switch back to master. Make a commit there, which should edit a different file from the ones you touched in your branch – to be sure there is no conflict.Now merge your branch again.



Step 8:

Look at git log. Notice that there is a merge commit. Also look in gitk. Notice the DAG now shows how things forked, and then were joined up again by a merge commit.

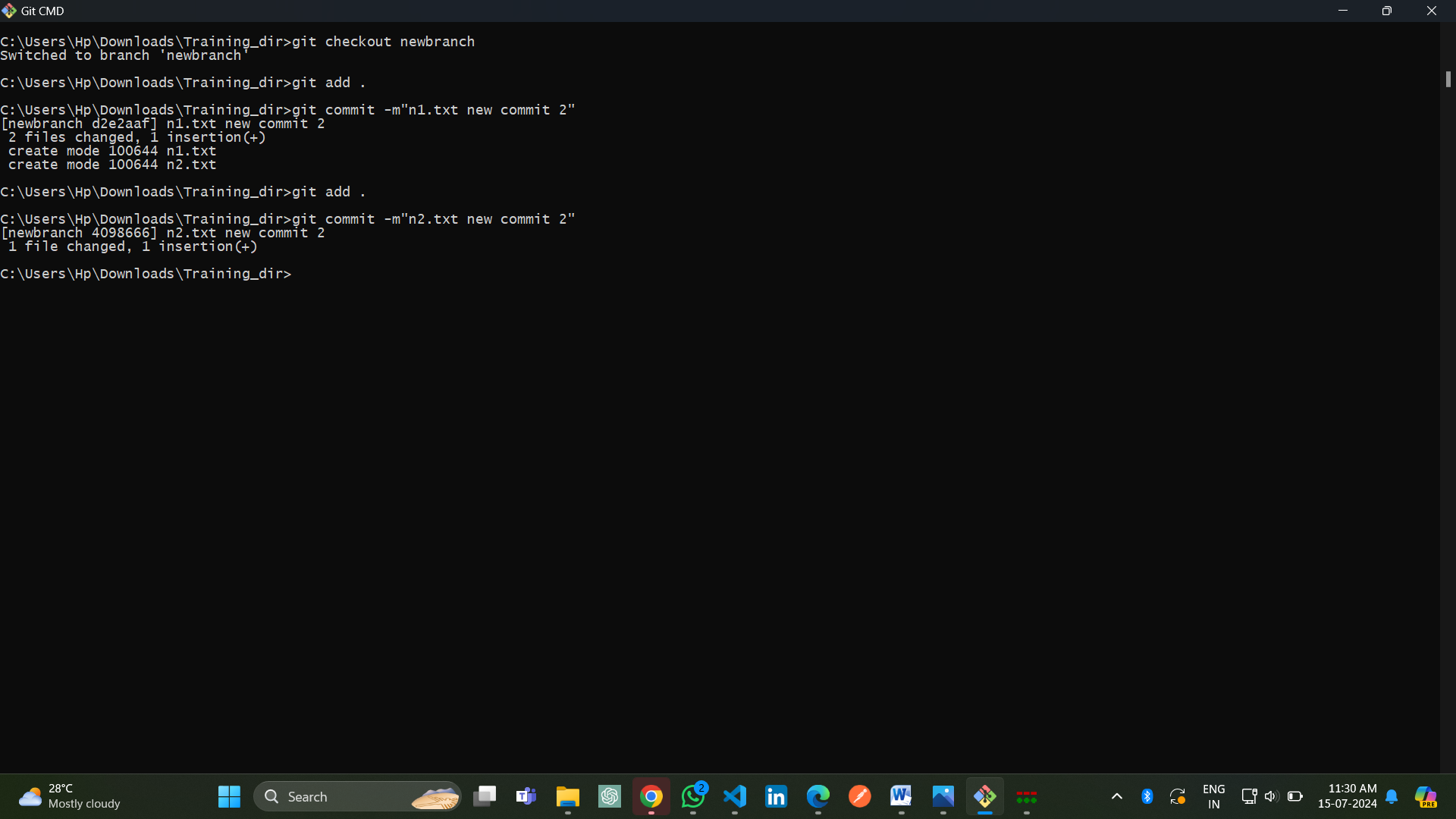




**Stretch Task:**

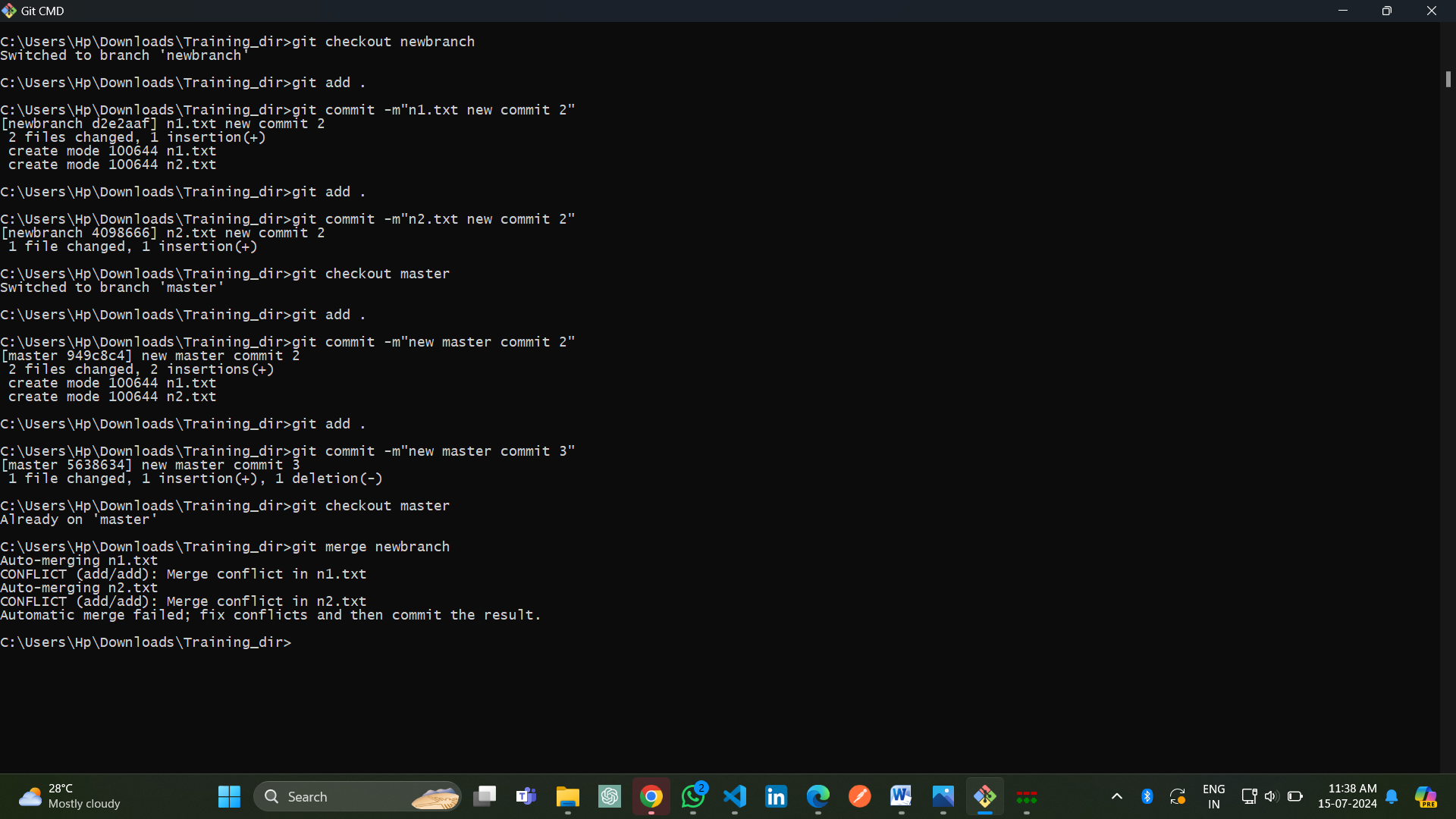
**Step 1:**

Once again, checkout your branch. Make a couple of commits.



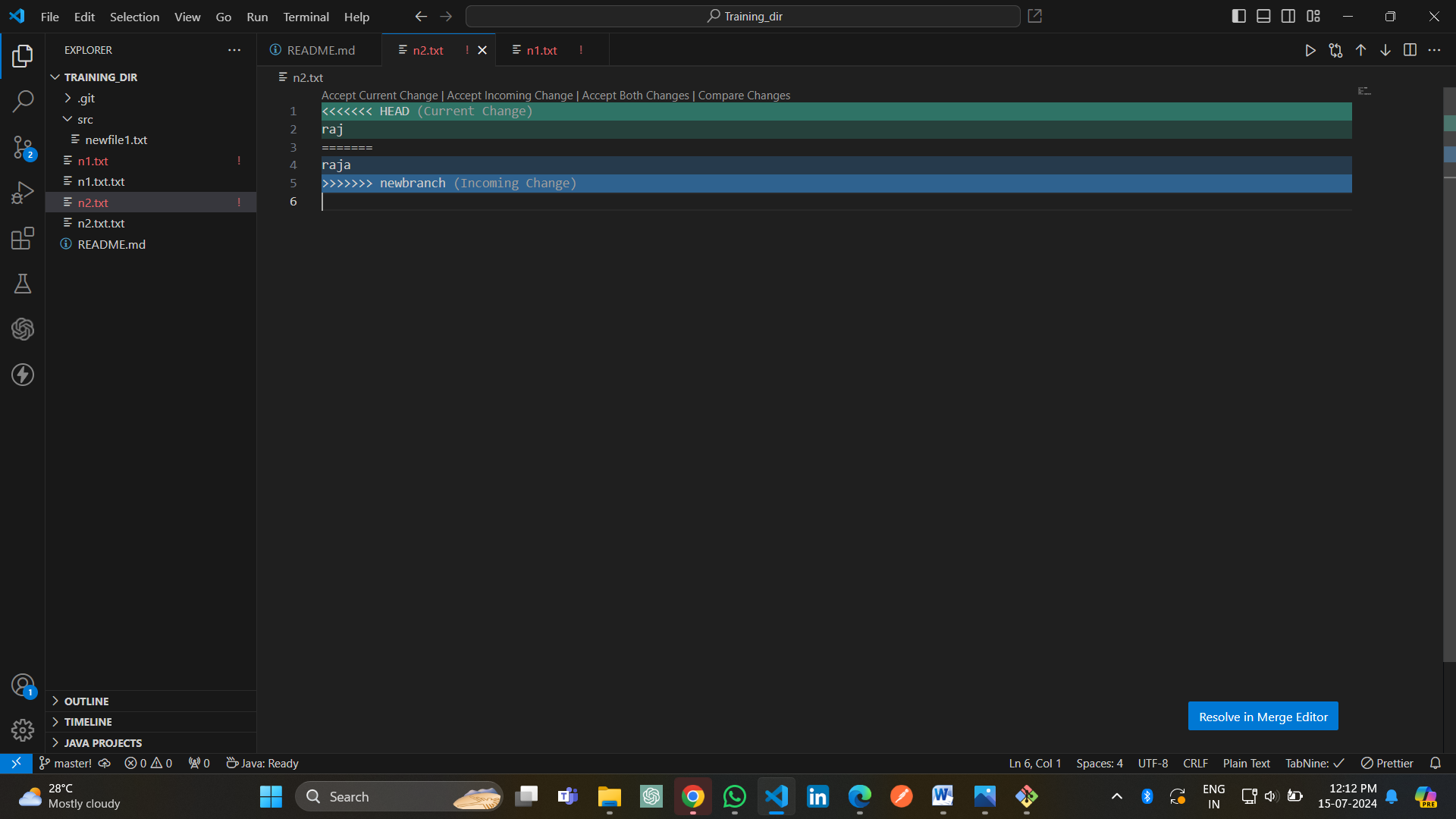
**Step 2:**

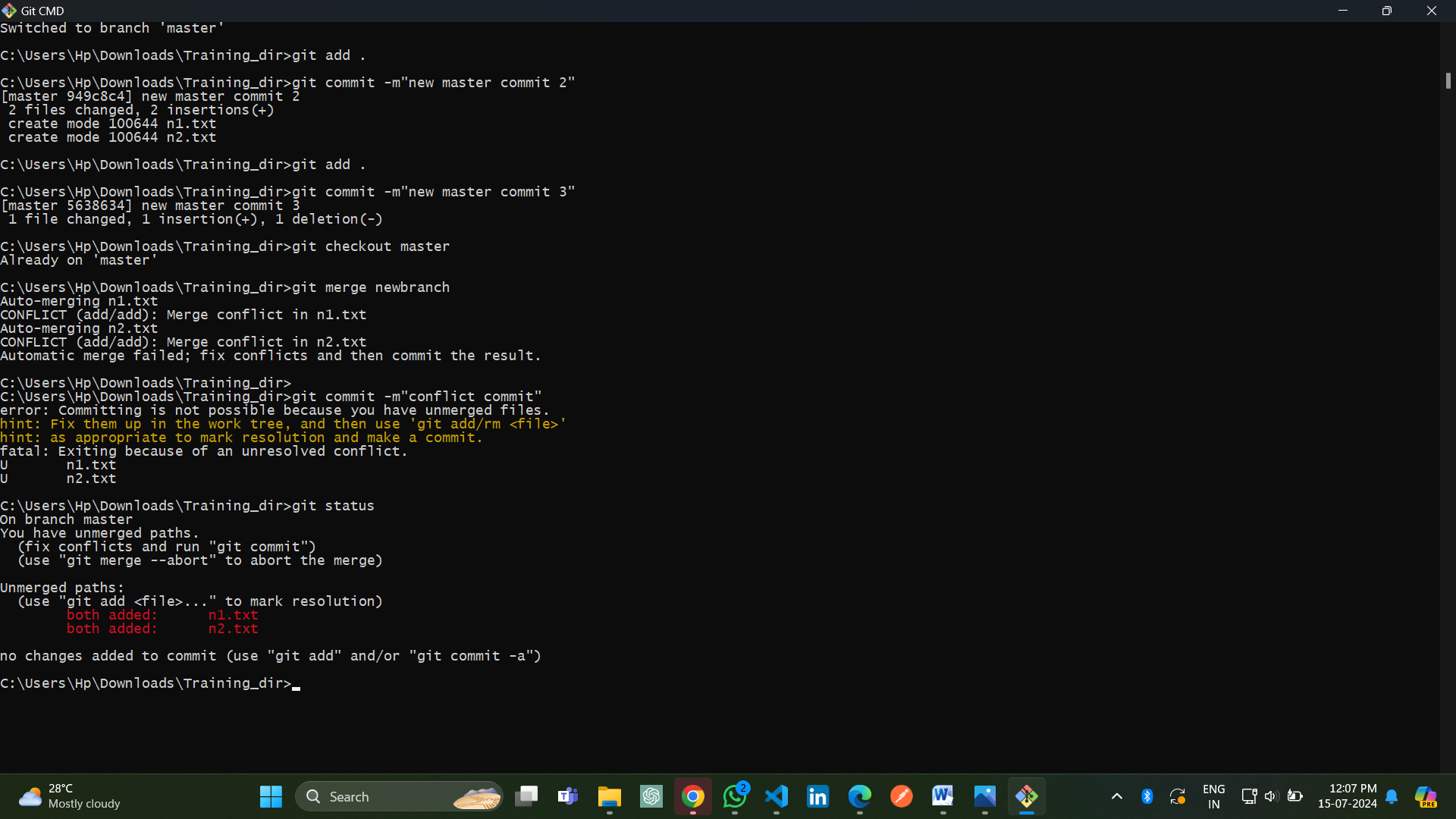
Return to your master branch. Make a commit there that changes the exact same line, or lines, as commits in your branch did. Now try to merge your branch. You should get a conflict.



**Step 3:**

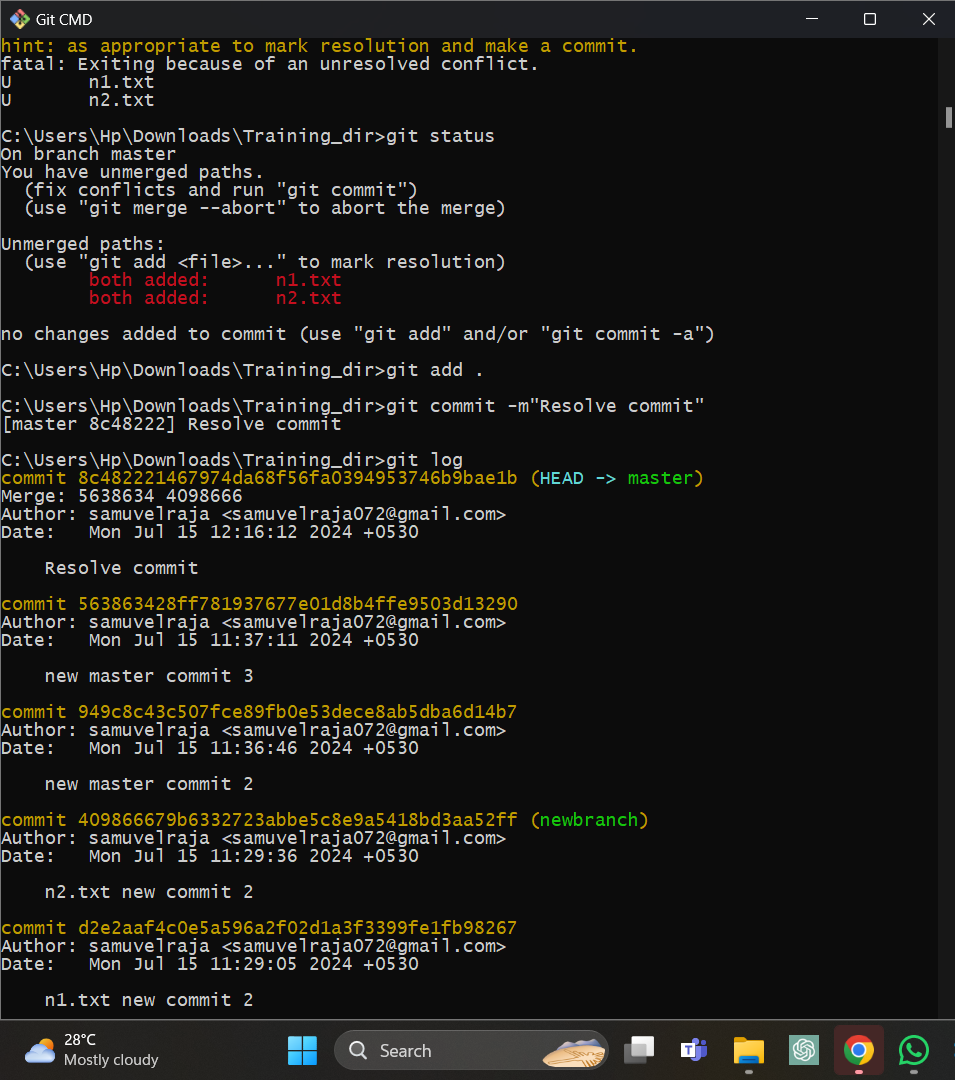
Open the file(s) that is in conflict. Search for the conflict marker. Edit the file to remove the conflict markers and resolve the conflict.Now try to commit. Notice that Git will not allow you to do this when you still have potentially unresolved conflicts. Look at the output of status too.

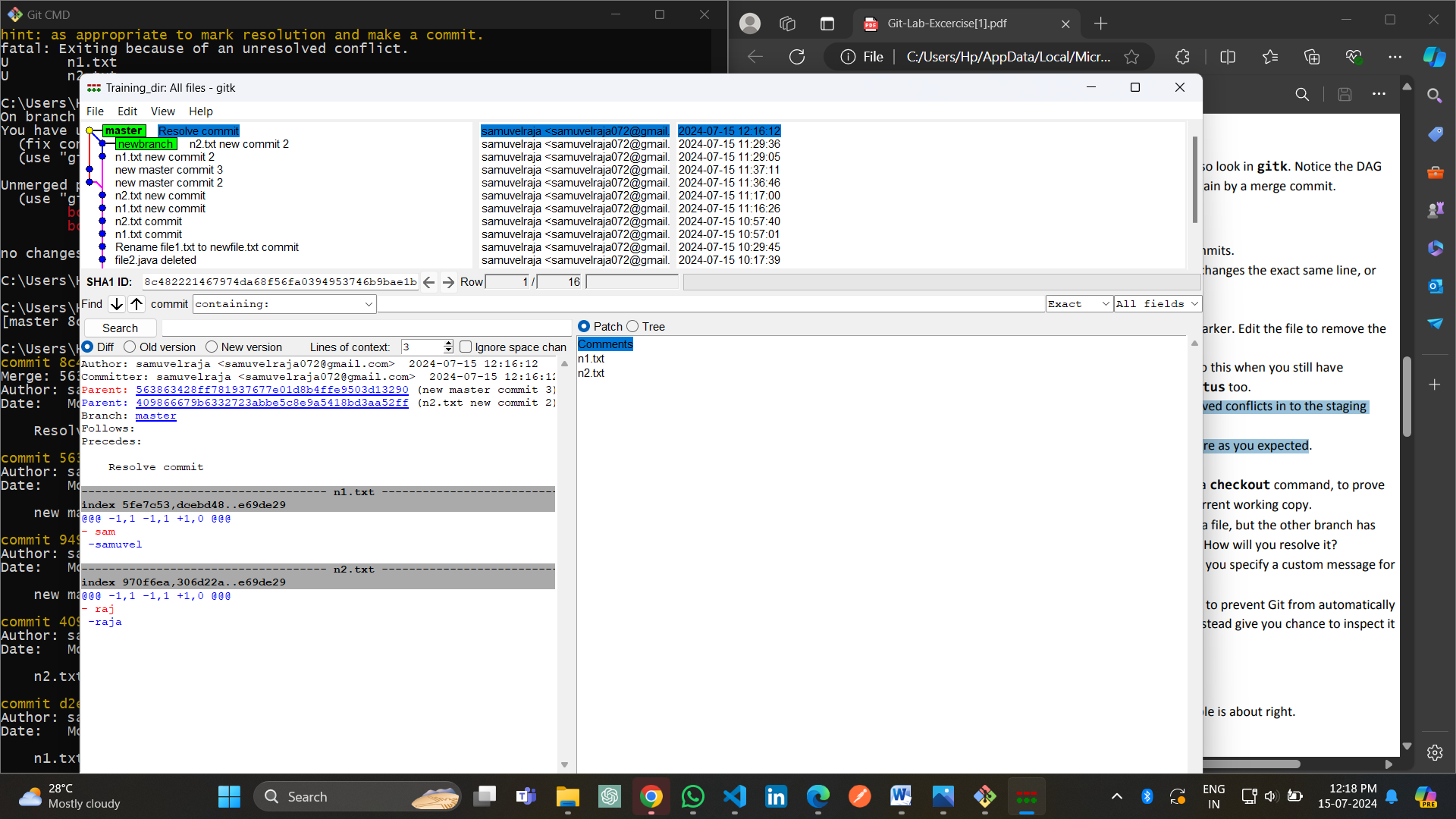




**Step 4:**

Use the add command to add the files that you have resolved conflicts in to the staging area. Then use commit to commit the merge commit.Take a look at git log and gitk, and make sure things are as you expected.

****

****