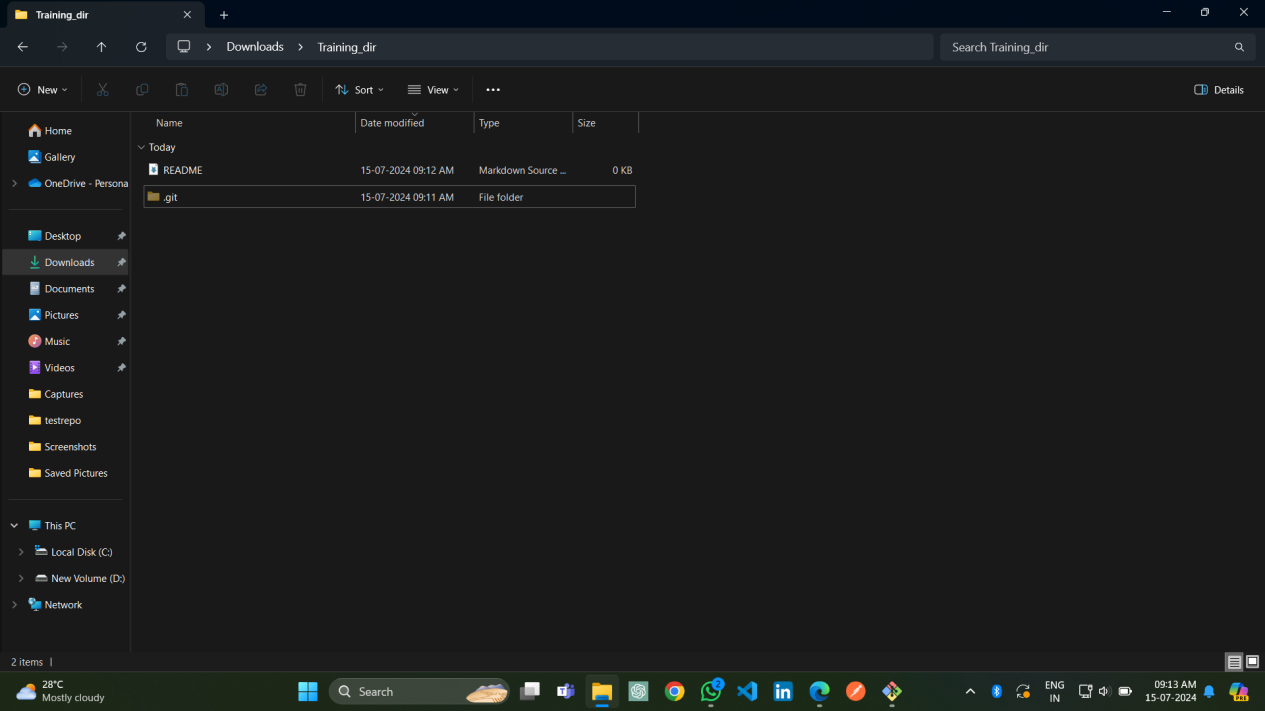
**Product Training Assignment 2 Exercise 1**

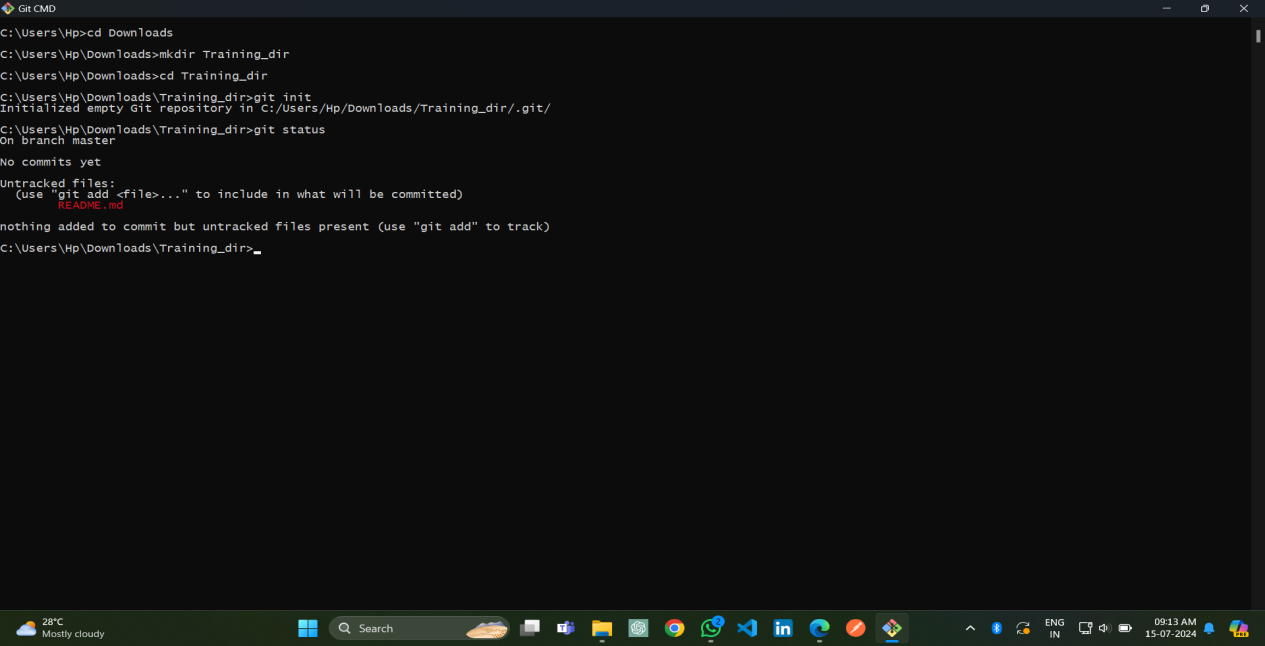
**Step 1:**

Create a new directory and change into it. Use the init command to create a Git repository in that directory. Observe that there is now a .git directory. Create a README file.

****

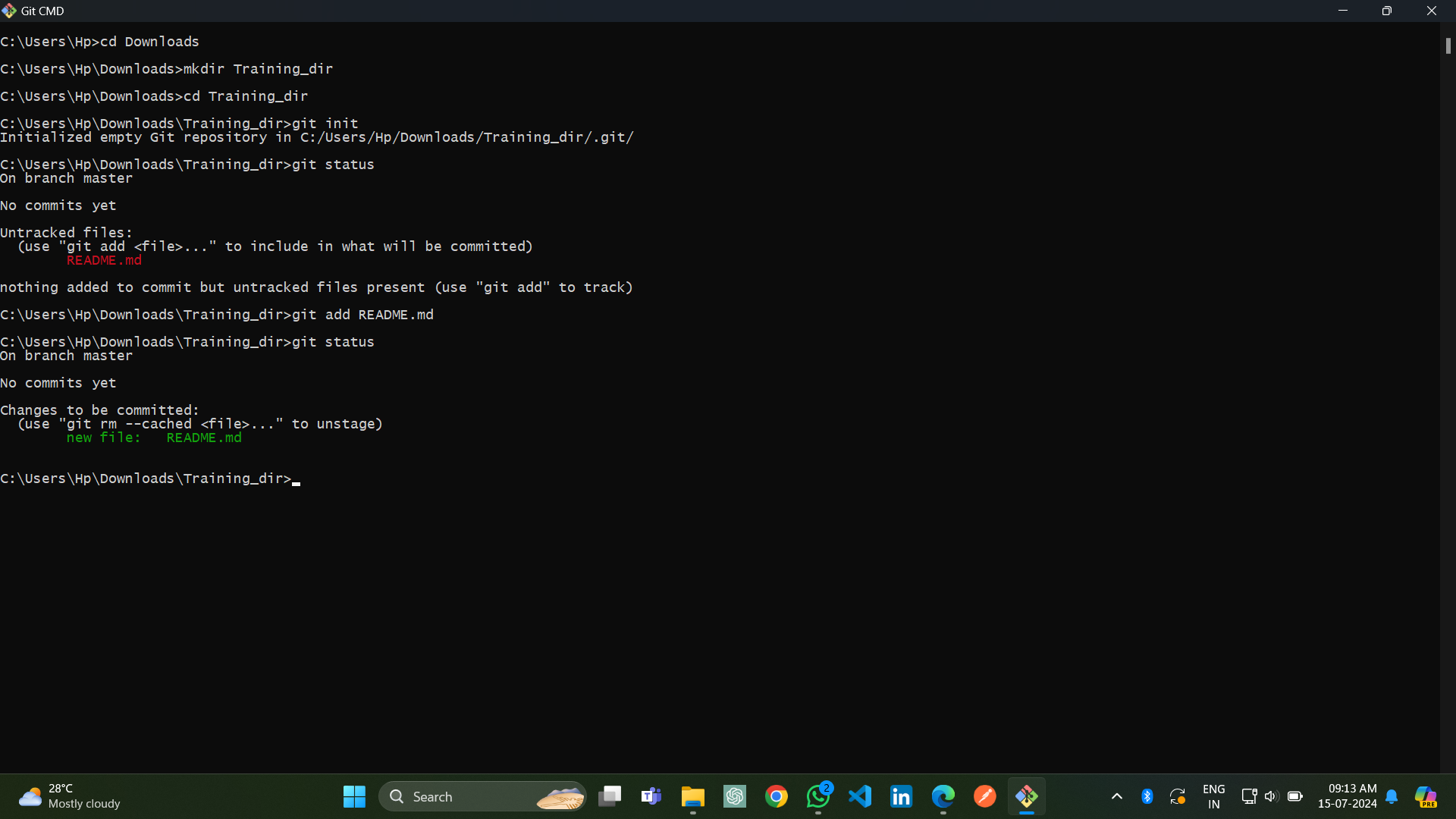
**Step 2:**

Look at the output of the status command; the README you created should appear as an untracked file.



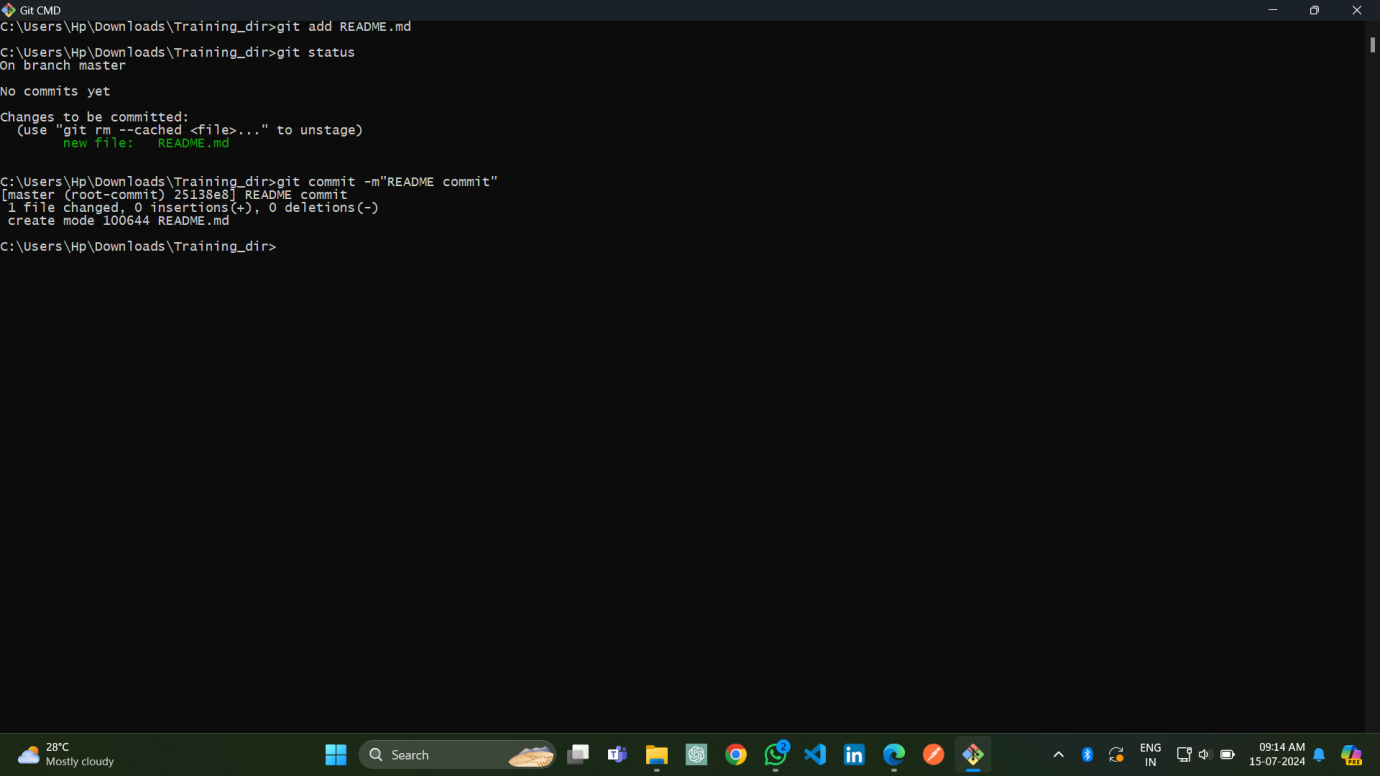
**Step 3:**

Use the add command to add the new file to the staging area. Again, look at the output of the status command.

****

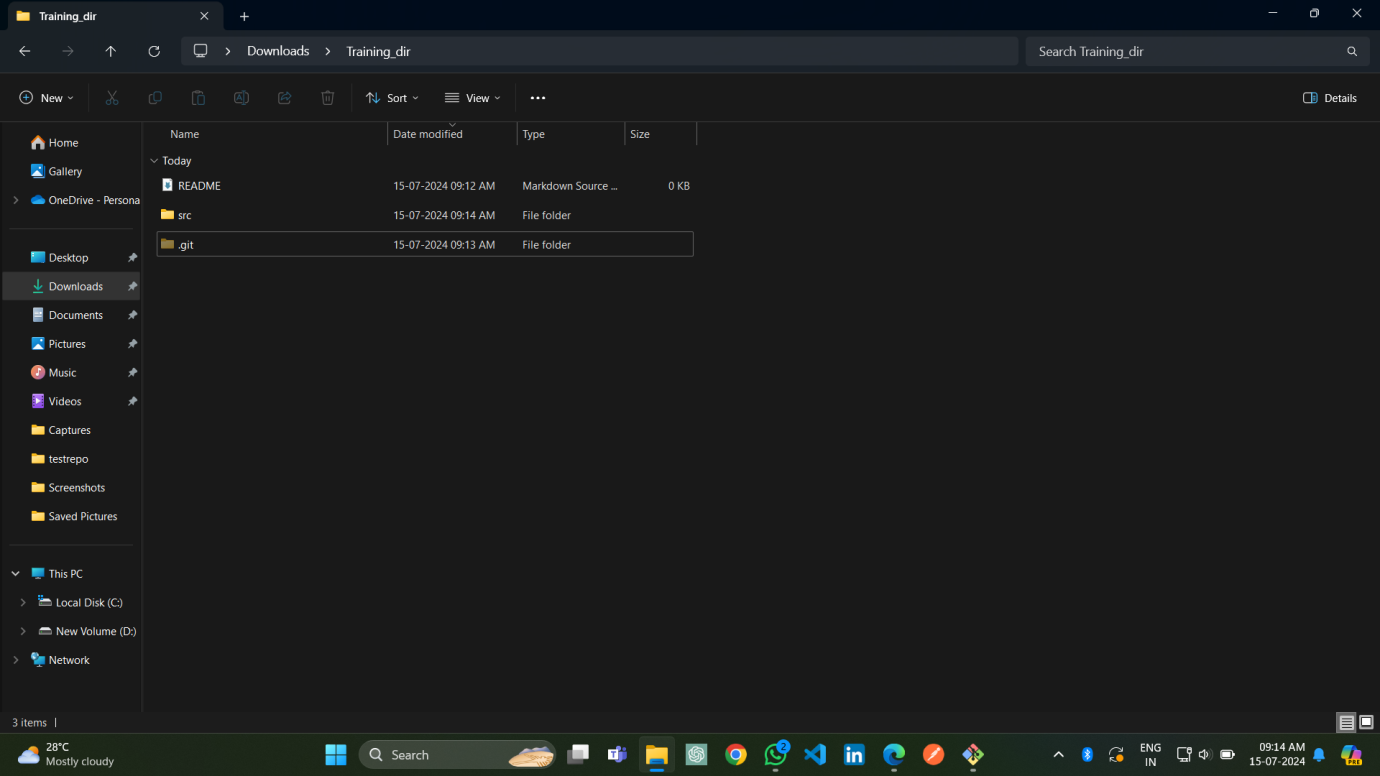
**Step 4:**

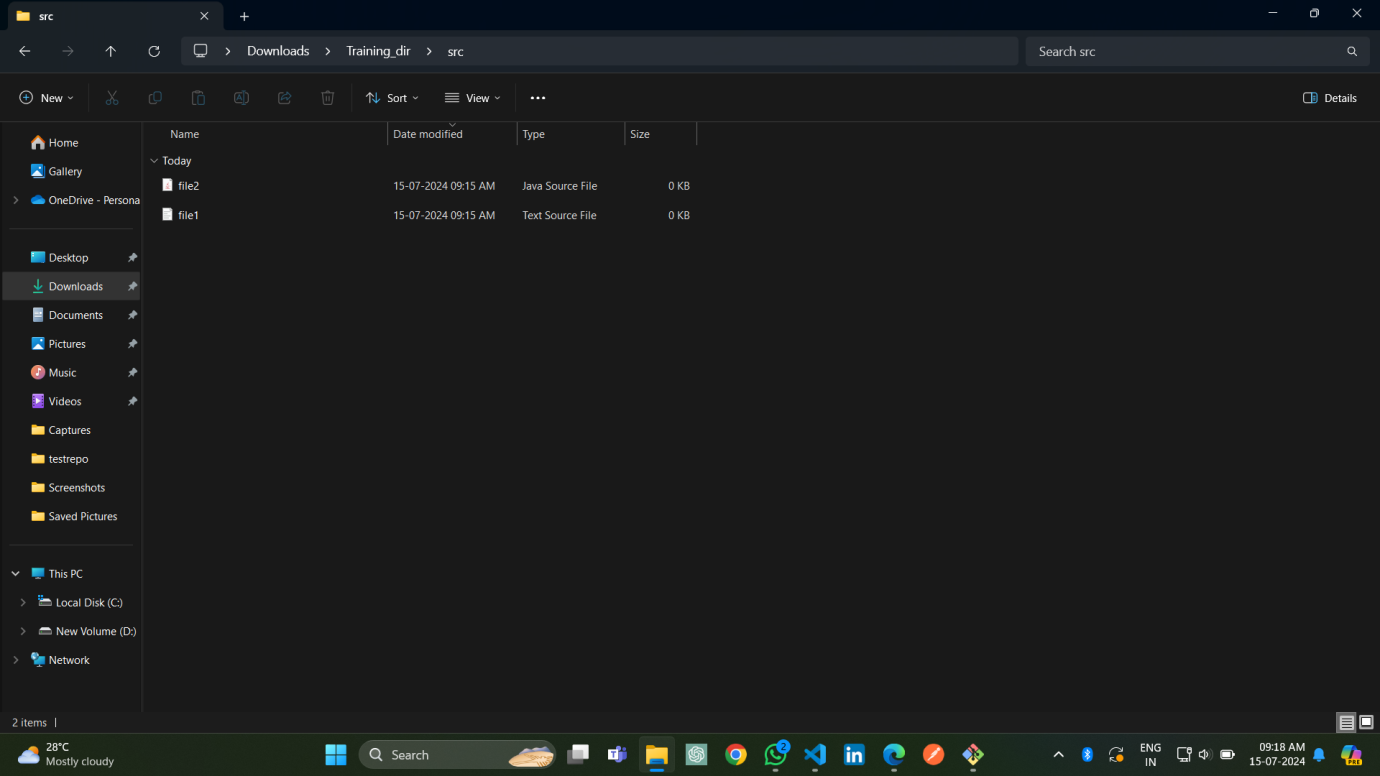
Now use the commit command to commit the contents of the staging area.



**Step 5:**

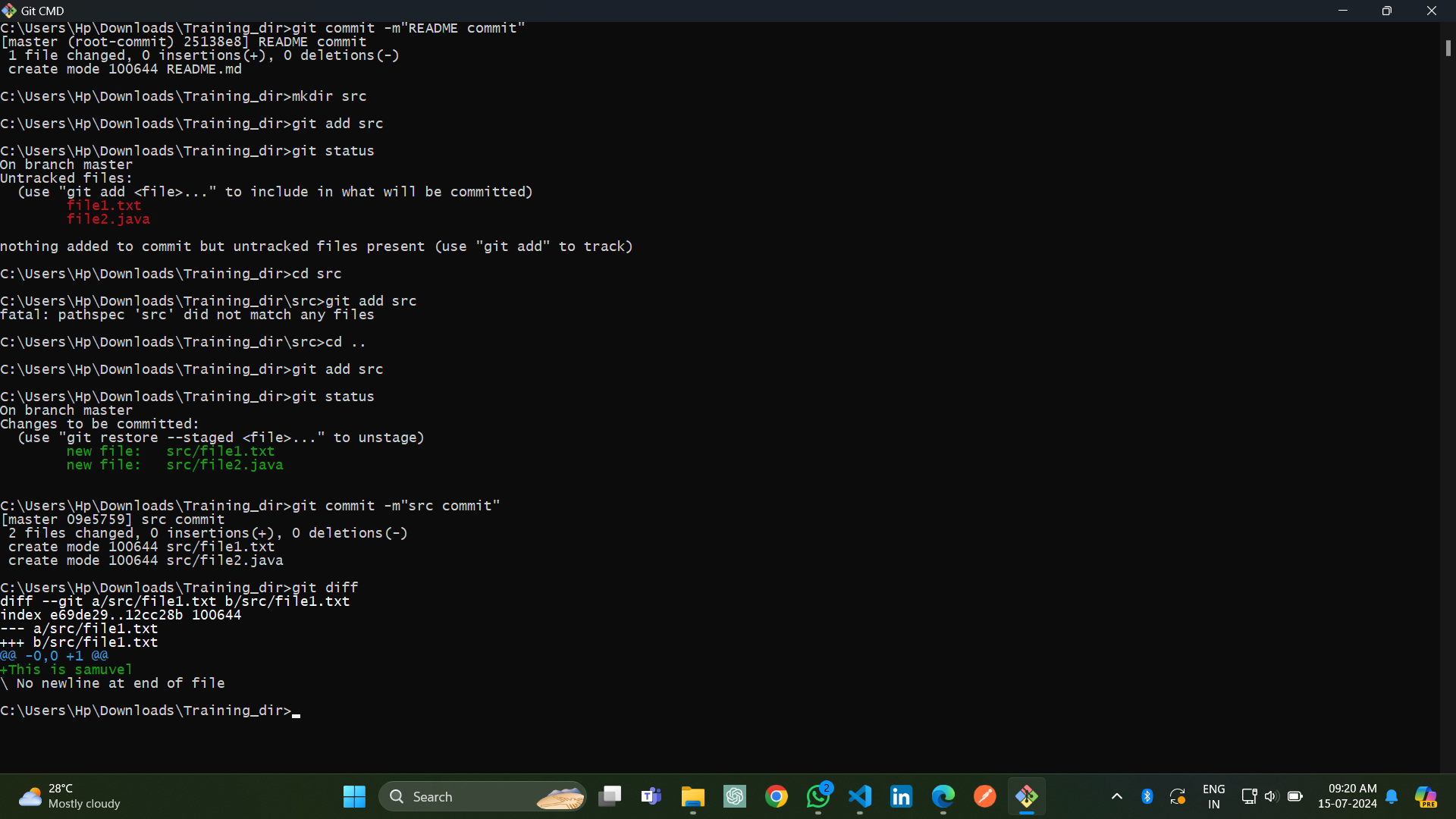
Create a src directory and add a couple of files to it.

****

****

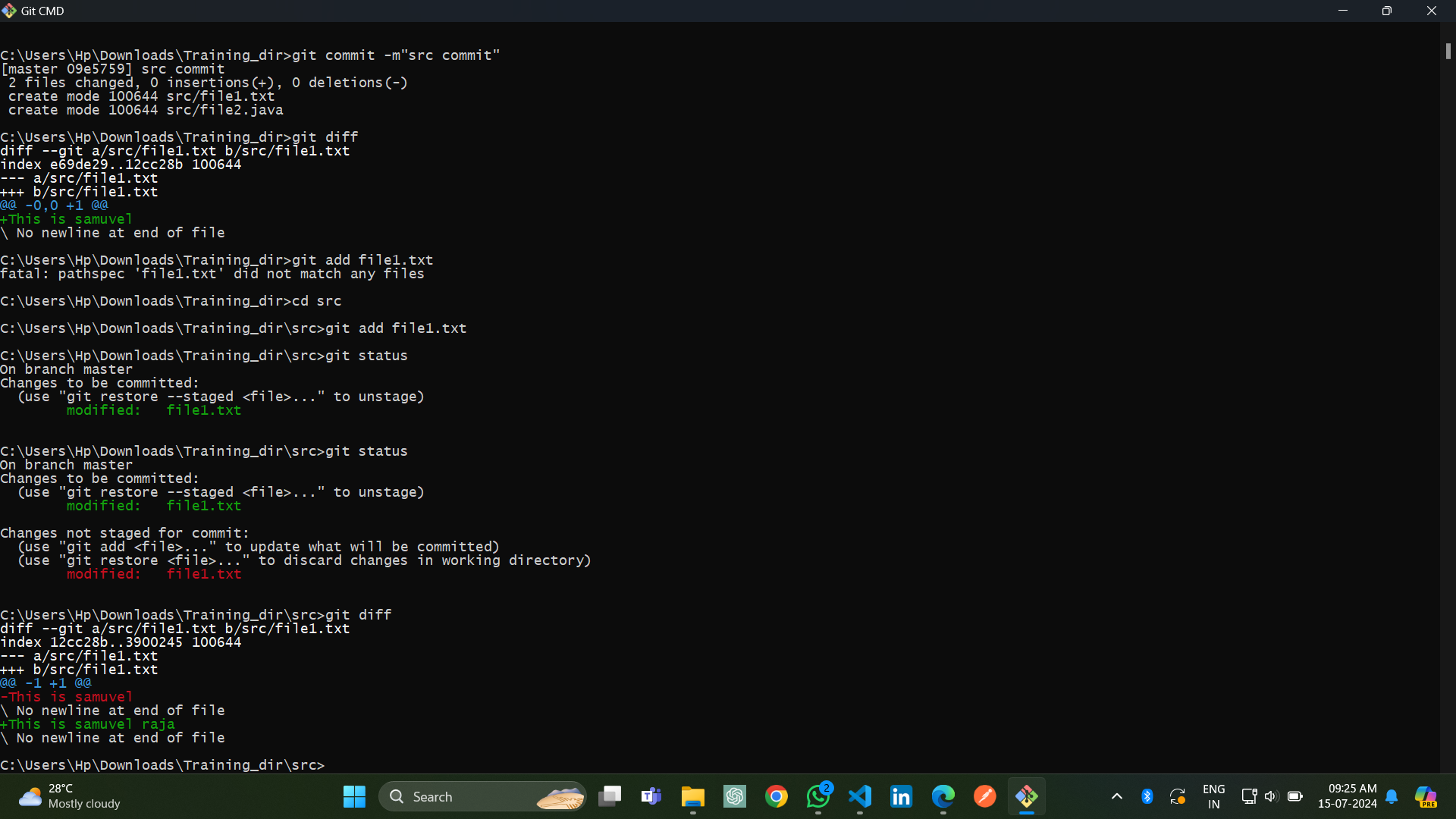
**Step 6:**

Use the add command, but name the directory, not the individual files. Use the status command. See how both files have been staged. Commit them.

****

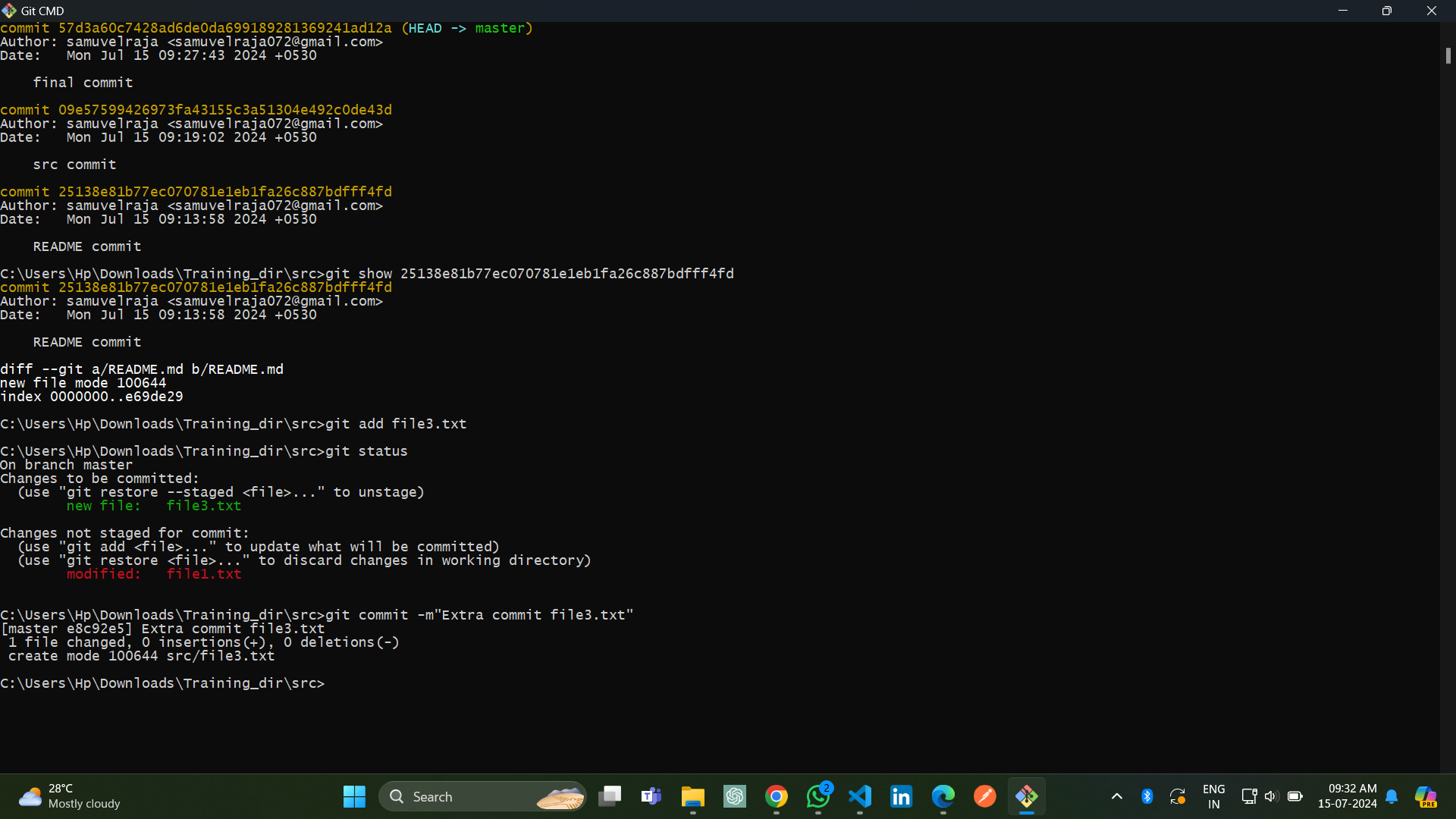
**Step 7:**

Make a change to one of the files. Use the diff command to view the details of the change.

****

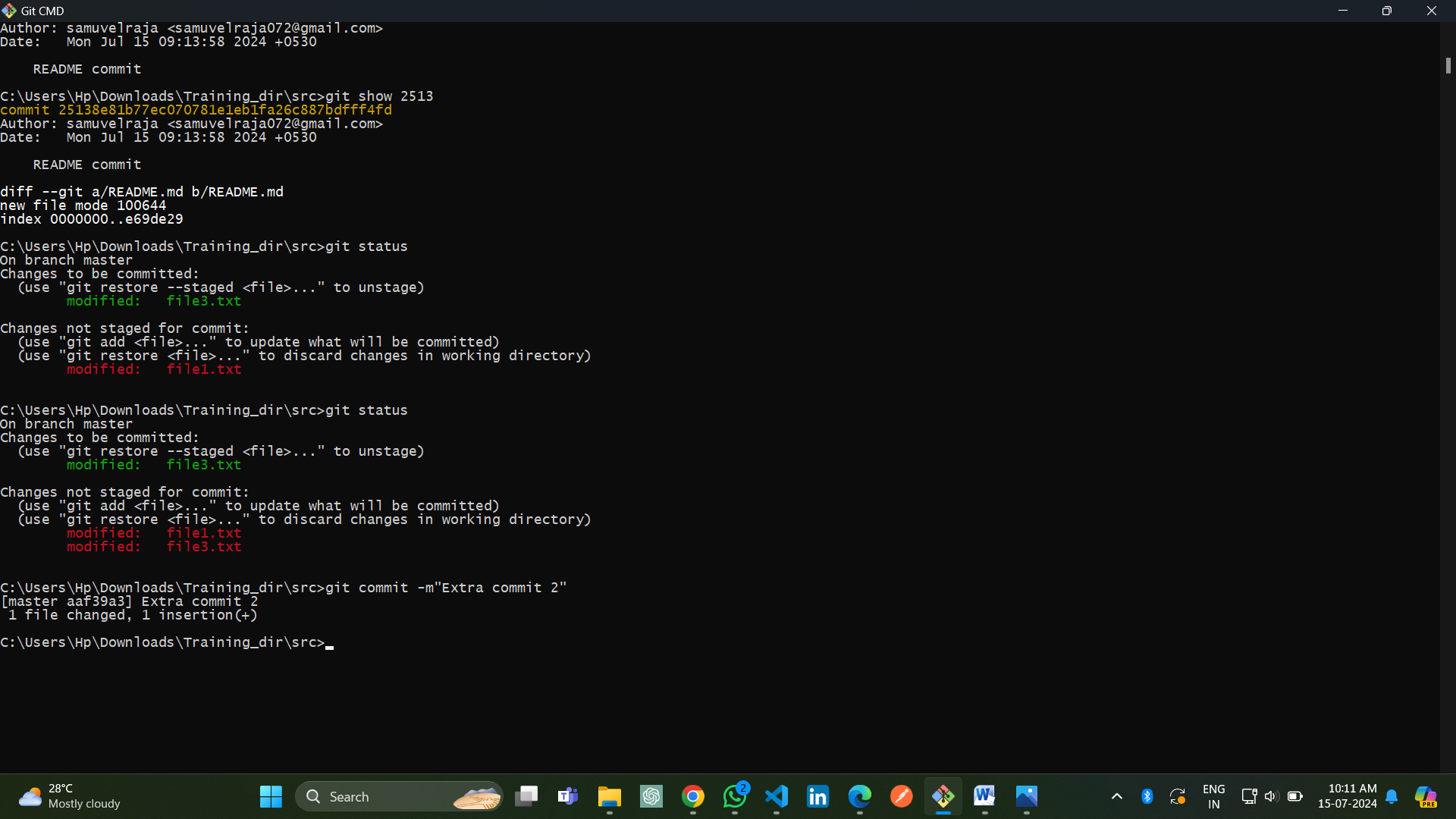
**Step 8:**

Next, add the changed file, and notice how it moves to the staging area in the status output.

****

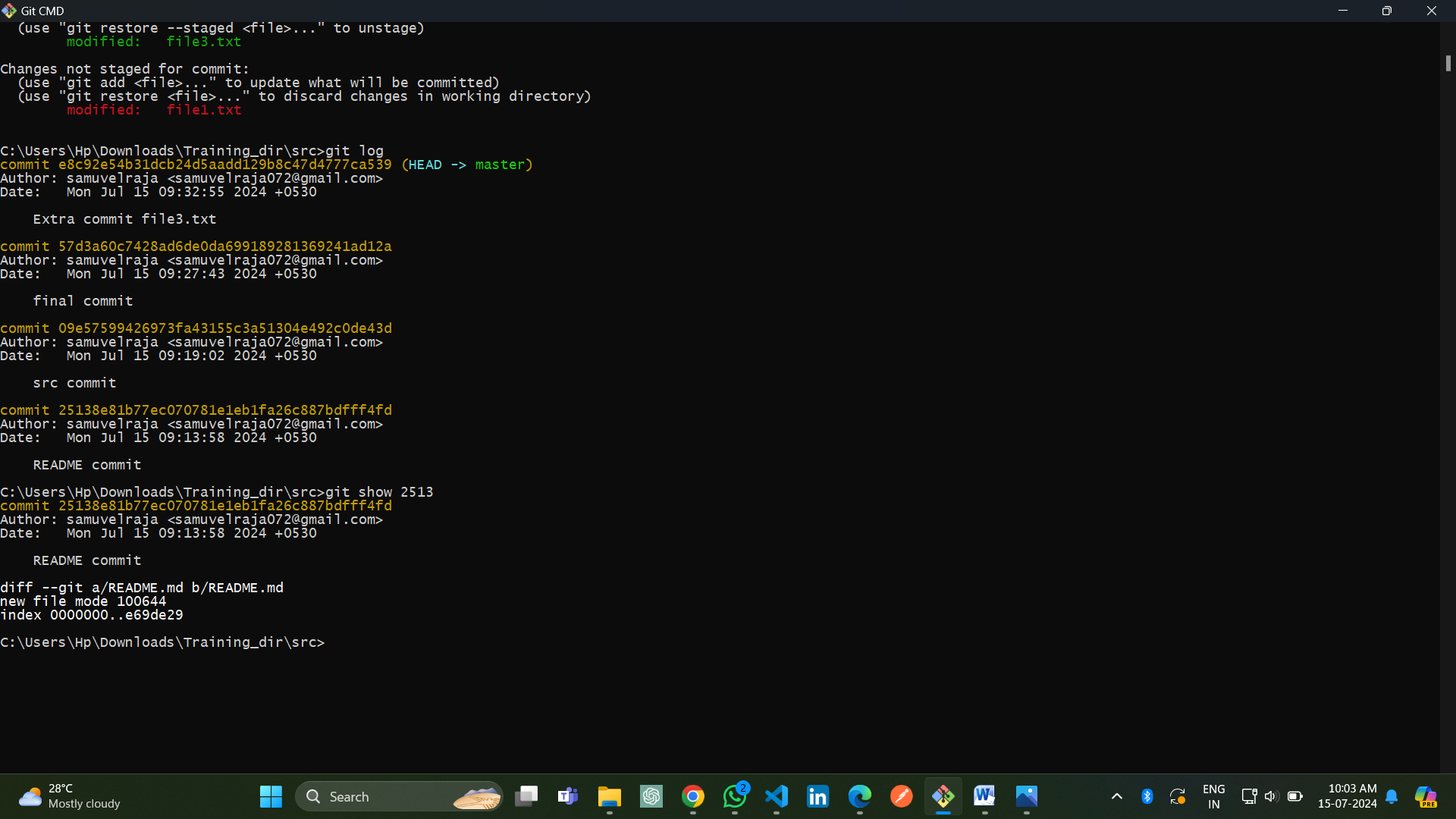
**Step 9:**

Now – without committing – make another change to the same file you changed in step Look at the status output, and the diff output. Notice how you can have both staged and unstaged changes, even when you’re talking about a single file. Observe the difference when you use the add command to stage the latest round of changes. Finally, commit them.



**Step 10:**

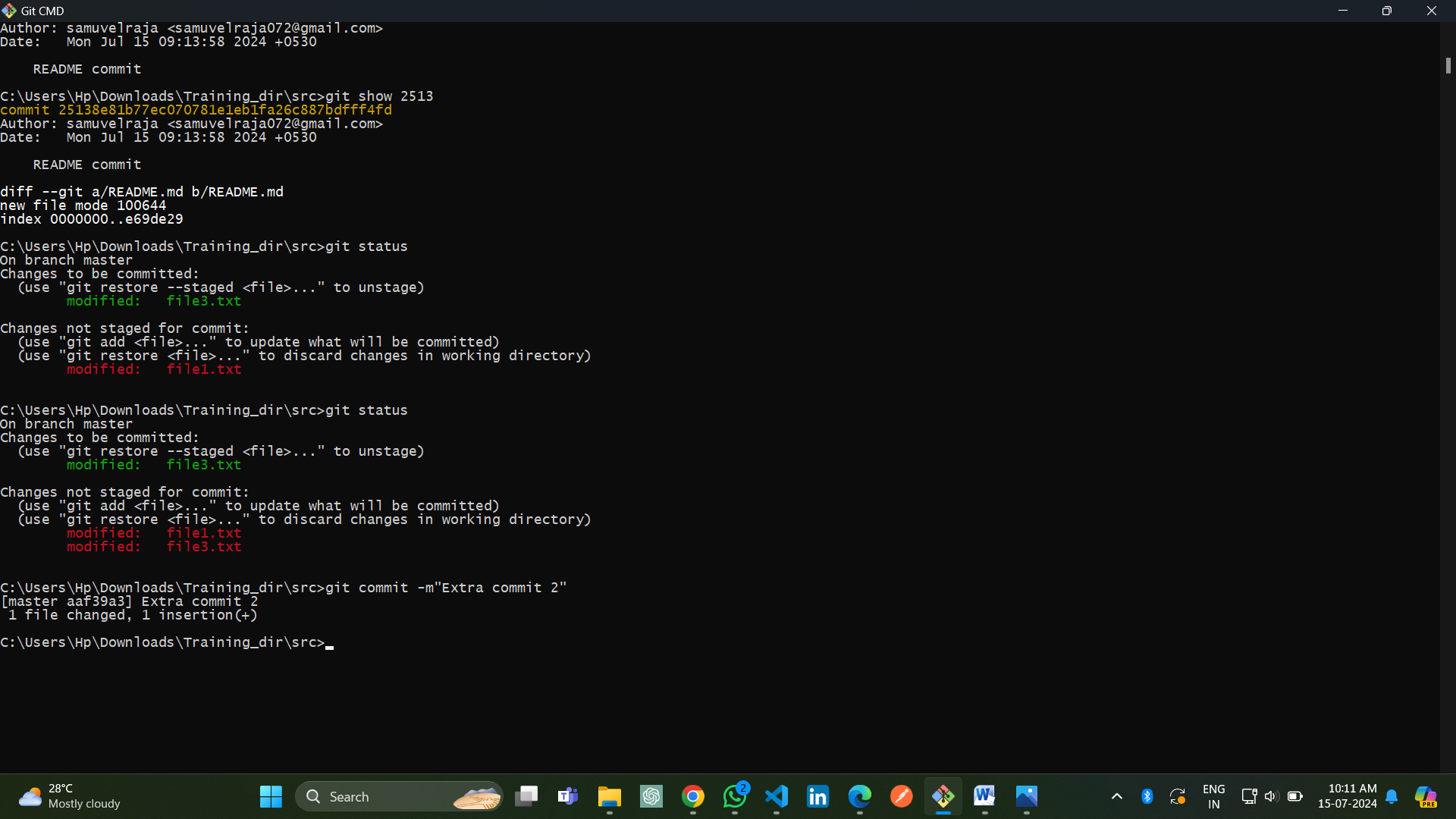
Use the log command in order to see all of the commits you made so far.

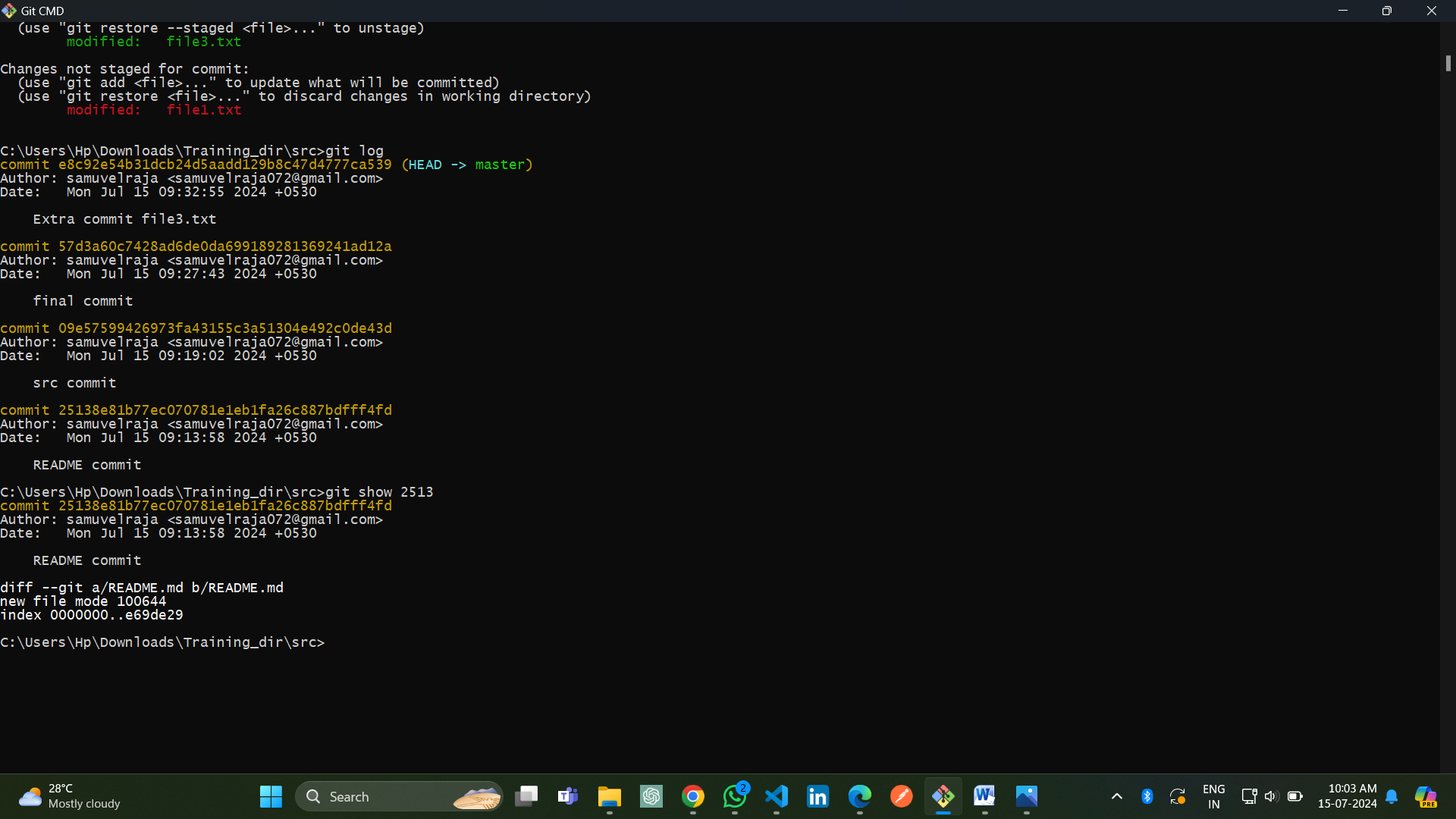
****

**Step 11:**

Use the show command to look at an individual commit. How many characters of the commit identifier can you get away with typing at a minimum?

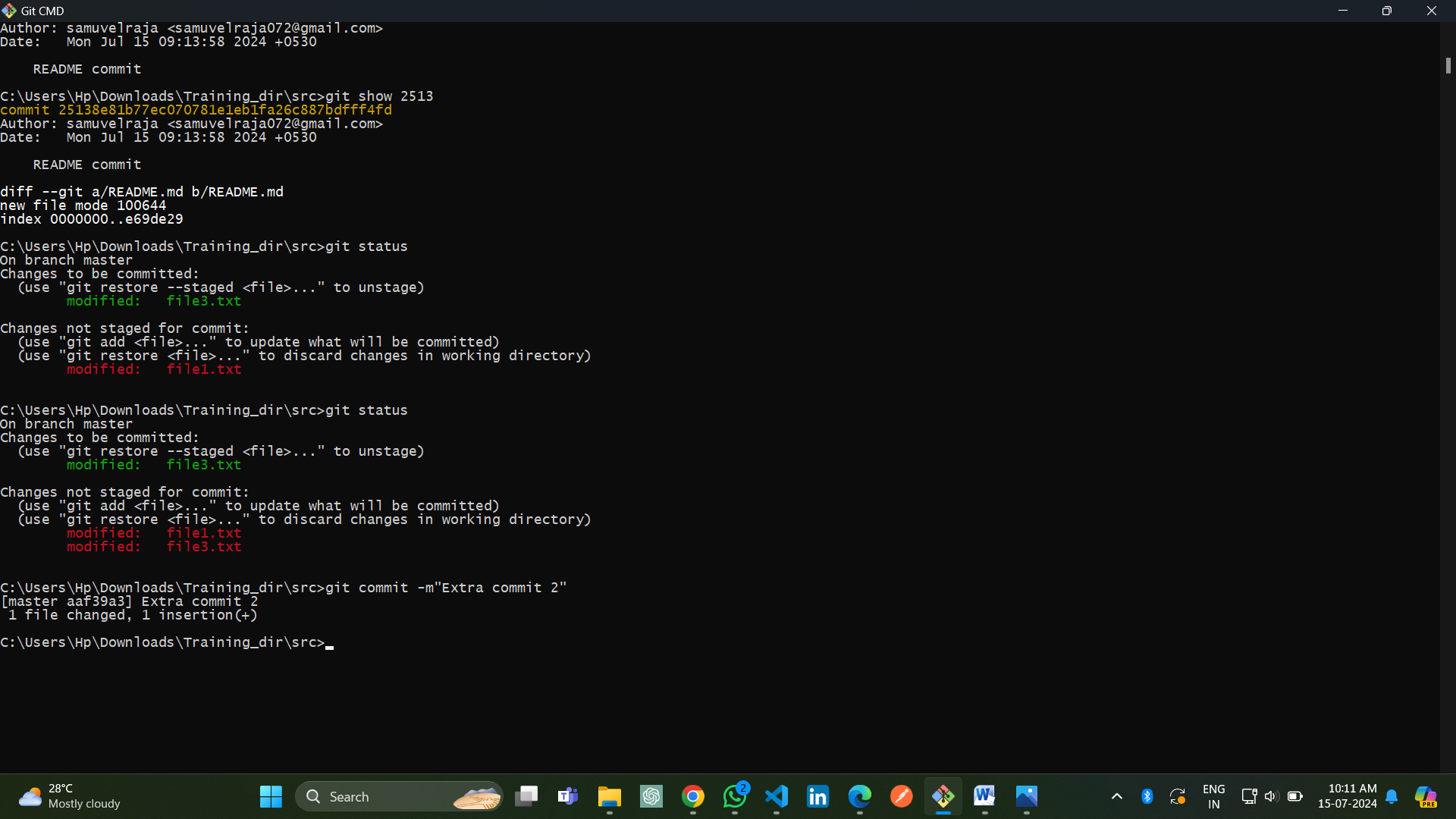
Ans: Minimum 4 characters is required to look at an individual commit.

****

****

**Step 12:**

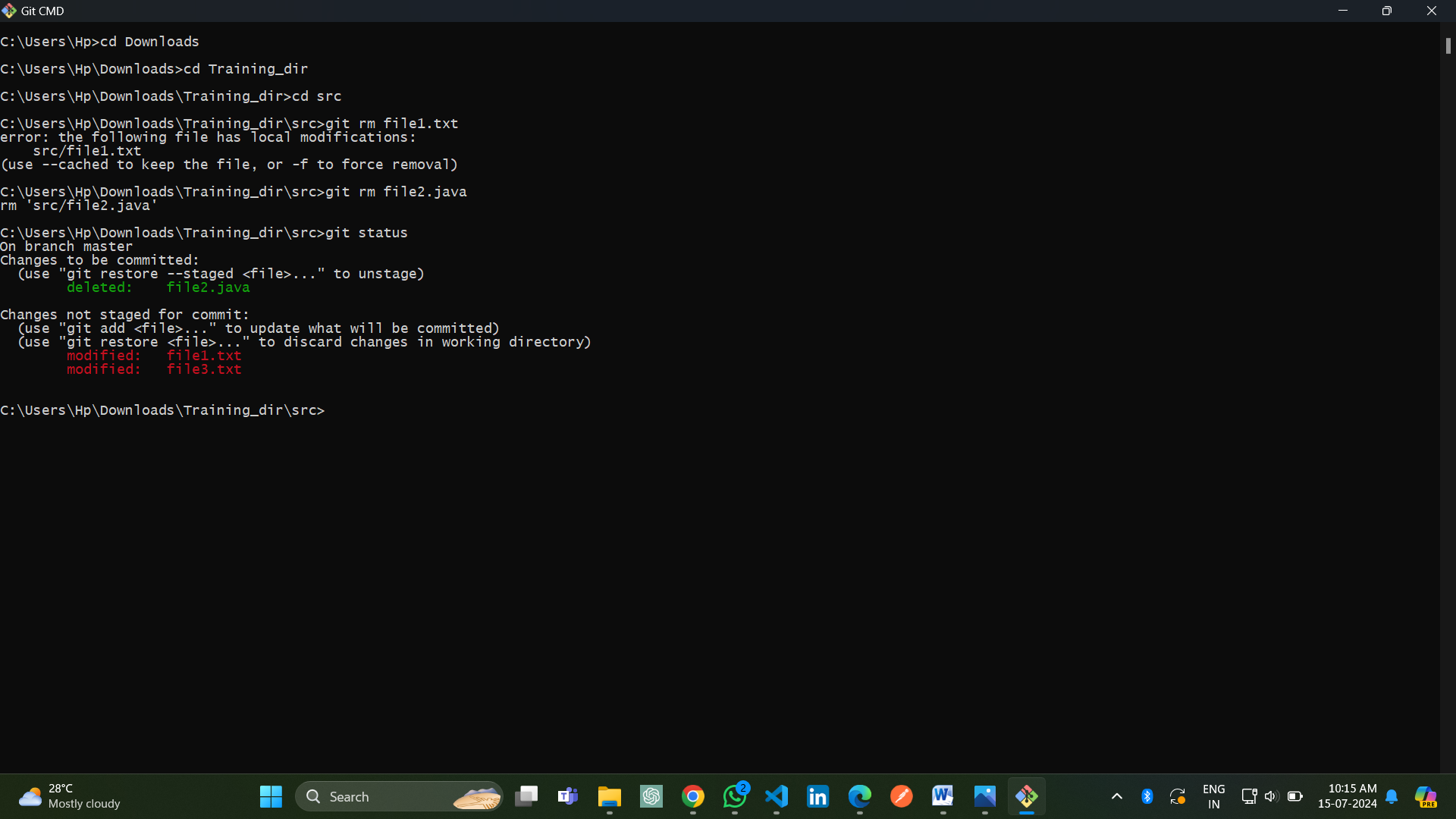
Make a couple more commits, at least one of which should add an extra file.

****

**Stretch Task:**

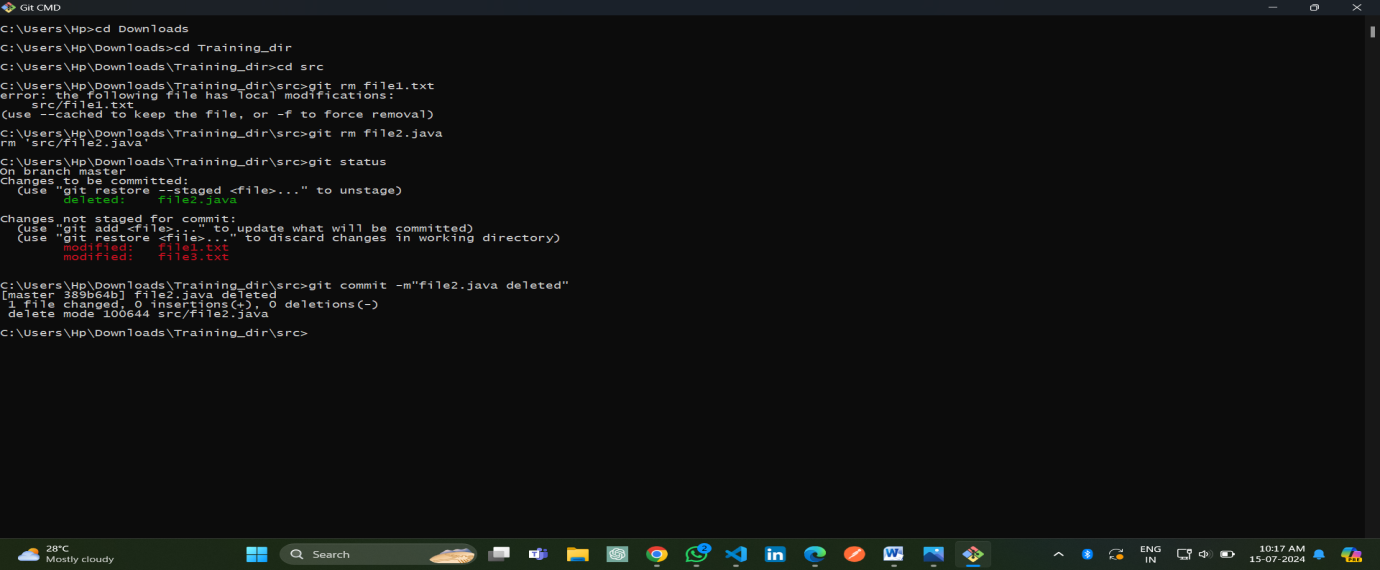
**Step 1:**

Use the Git rm command to remove a file. Look at the status afterwards. Now commit the deletion.

****

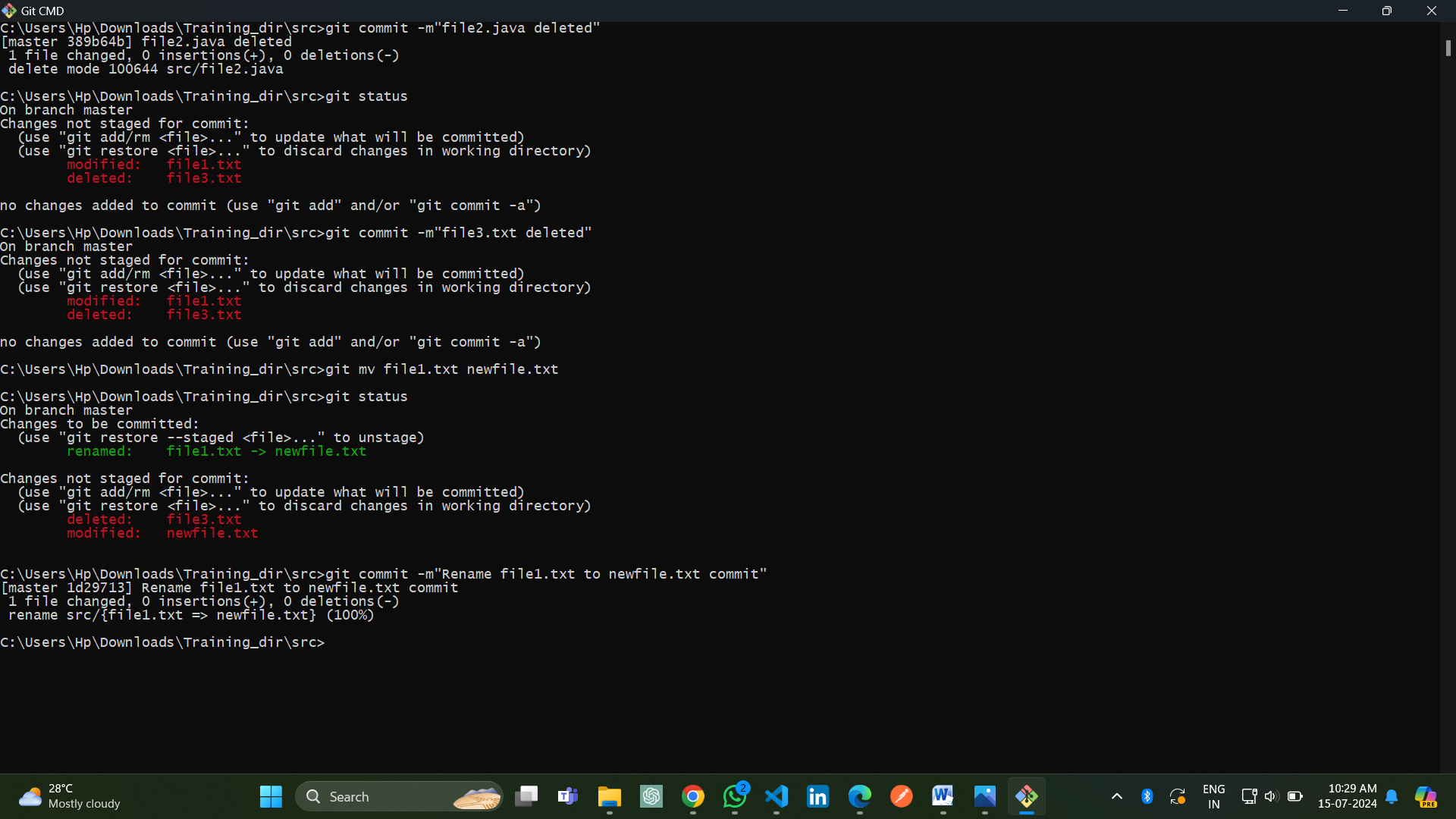
**Step 2:**

Delete another file, but this time do not use Git to do it; e.g. if you are on Linux, just use the normal (non-Git) rm command; on Windows use del and look at the status.



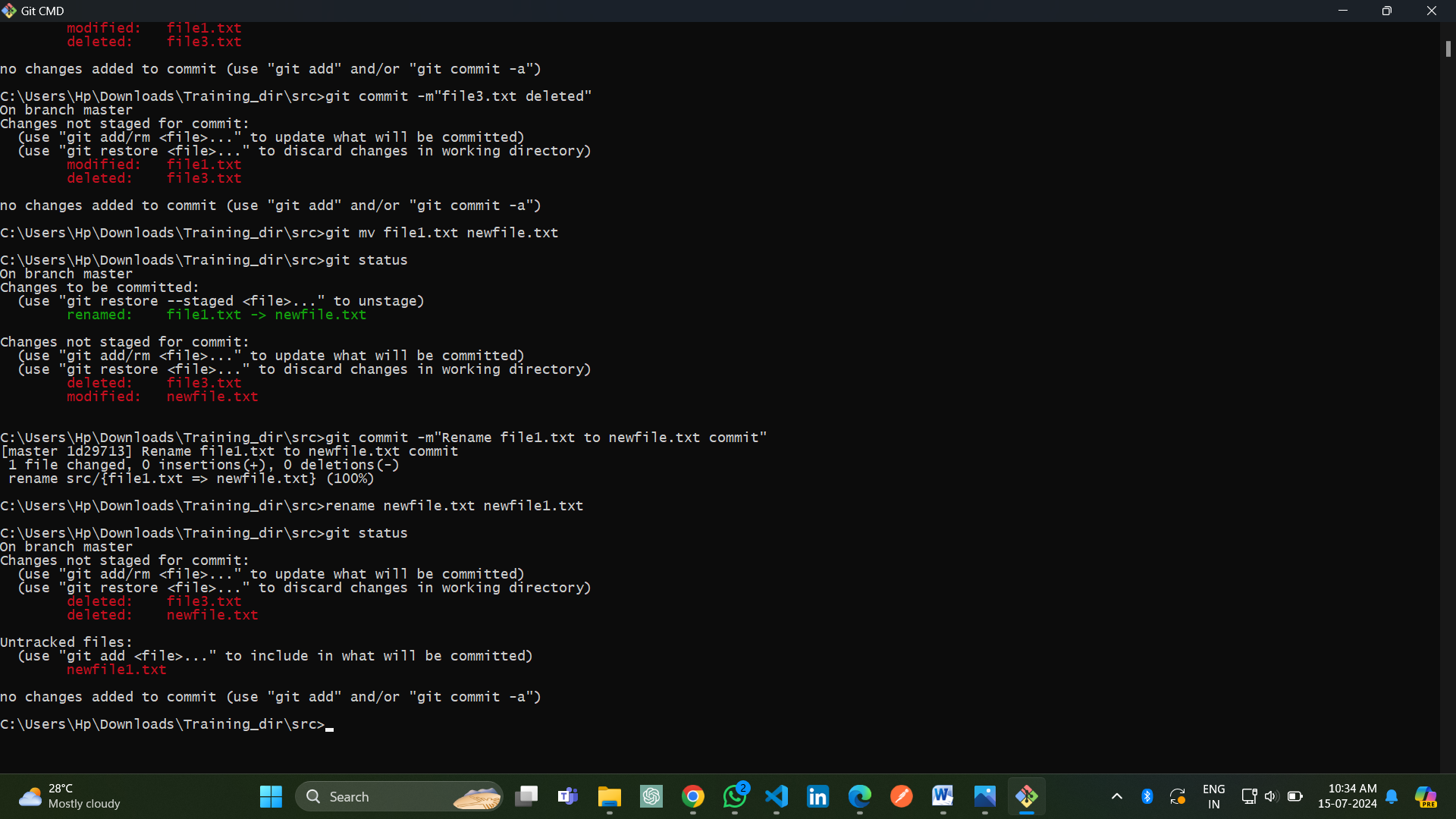
**Step 3:**

Use the Git mv command to move or rename a file; for example, rename README to README.txt. Look at the status. Commit the change.

****

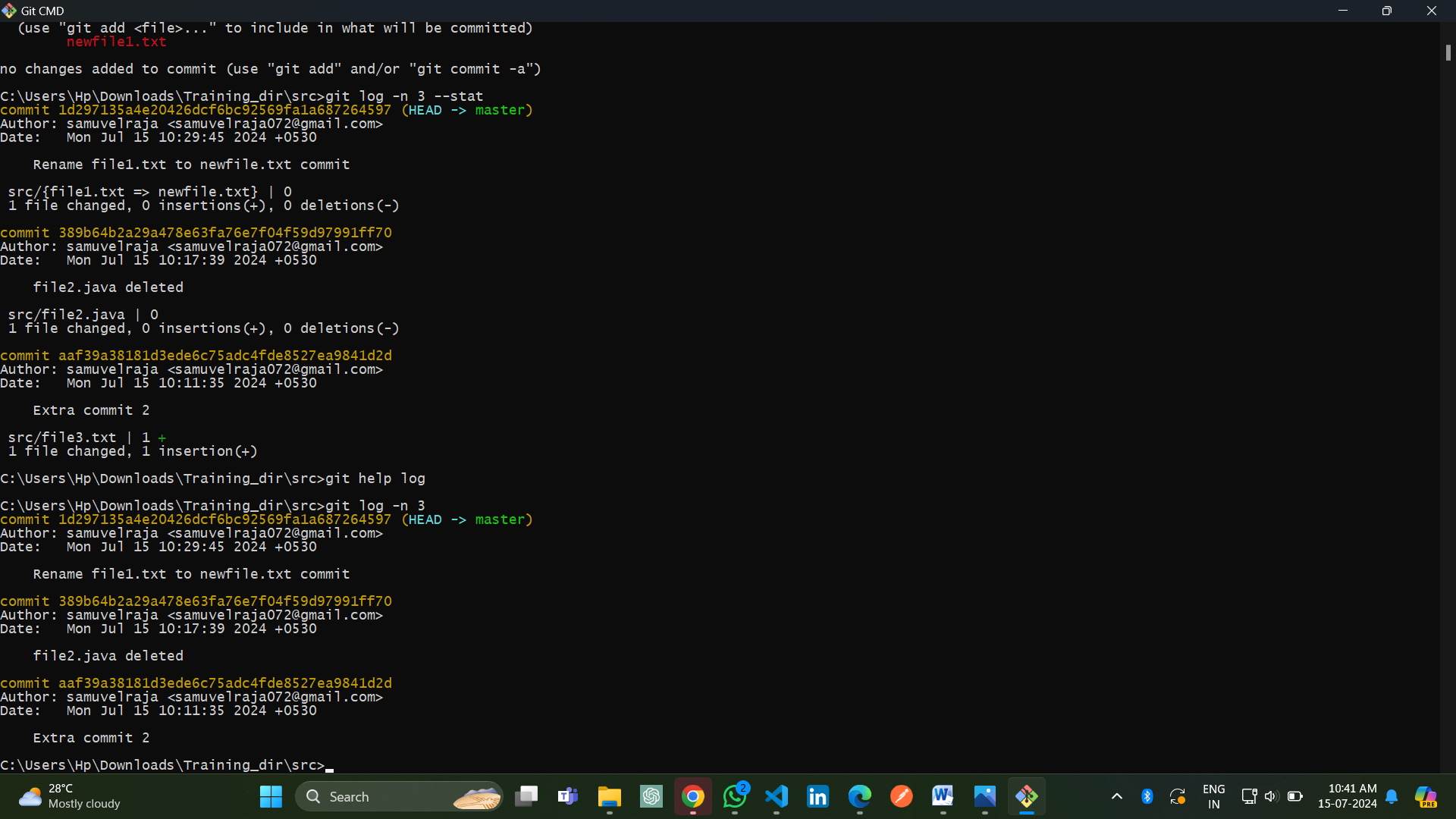
**Step 4:**

Now do another rename, but this time using the operating system’s command to do so.

****

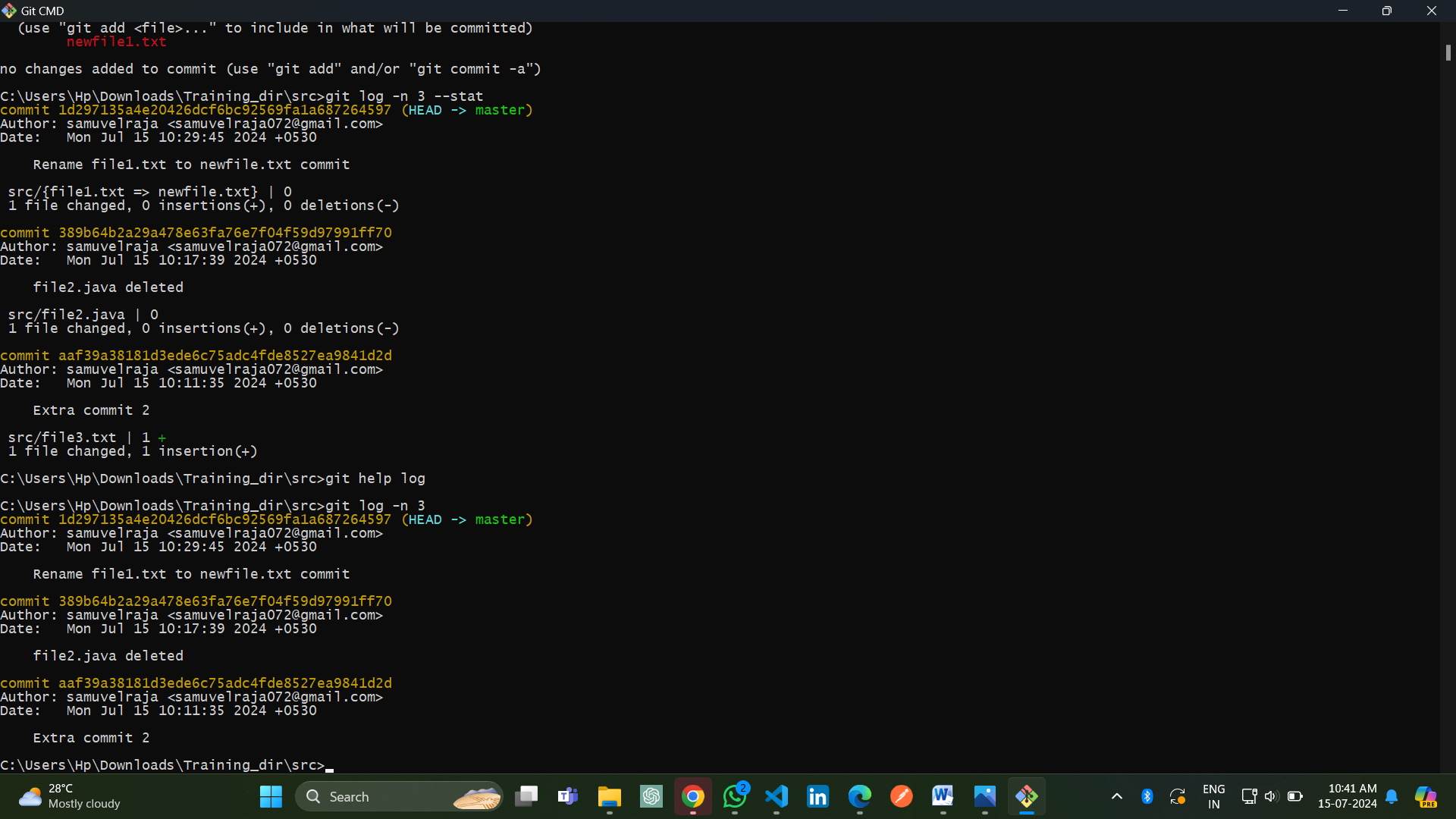
**Step 5:**

Use git help log to find out how to get Git to display just the most recent 3 commits.

****

**Step 6:**

If you don’t remember, look back in the slides to see what the --stat option did on the diff command. Find out if this also works with the show command. How about the log command?

****

**Step 7:**

Imagine you want to see a diff that summarizes all that happened between two commit identifiers. Use the diff command, specifying two commit identifiers joined by two dots (that is, something like abc123..def456). Check the output is what you expect.

