**By Aradhana Dubey**

**Question 1**

Step 1: Create a new Git repository.

Step 2: Create a file and commit changes.

Step 3: View the commit history of your repository.

Step 4: Open the file you created earlier and make some changes to it.

Step 5: Check the file you modified is now marked as "modified" and unstaged.

Hint (git status)

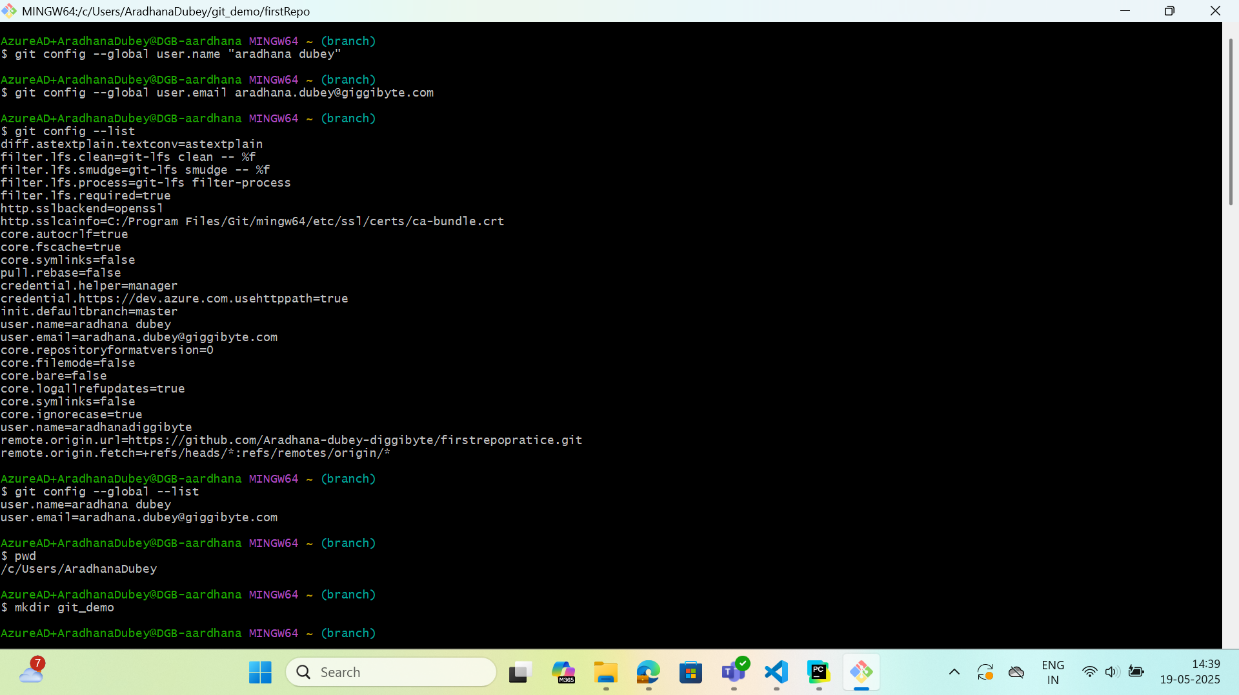
Step 6: Stage the changes you made to the file and commit the changes to the repository.

Step 7: Clone the repository you have created in GitHub.

Step 8: Fetch the changes, navigate into the cloned repository using the command line, and use the command git fetch to fetch any changes that have been made to the original repository since you cloned it.

Step 9: Pull changes, merge the changes you just fetched into your local copy of the repository, and use the command git pull.

**Answer**

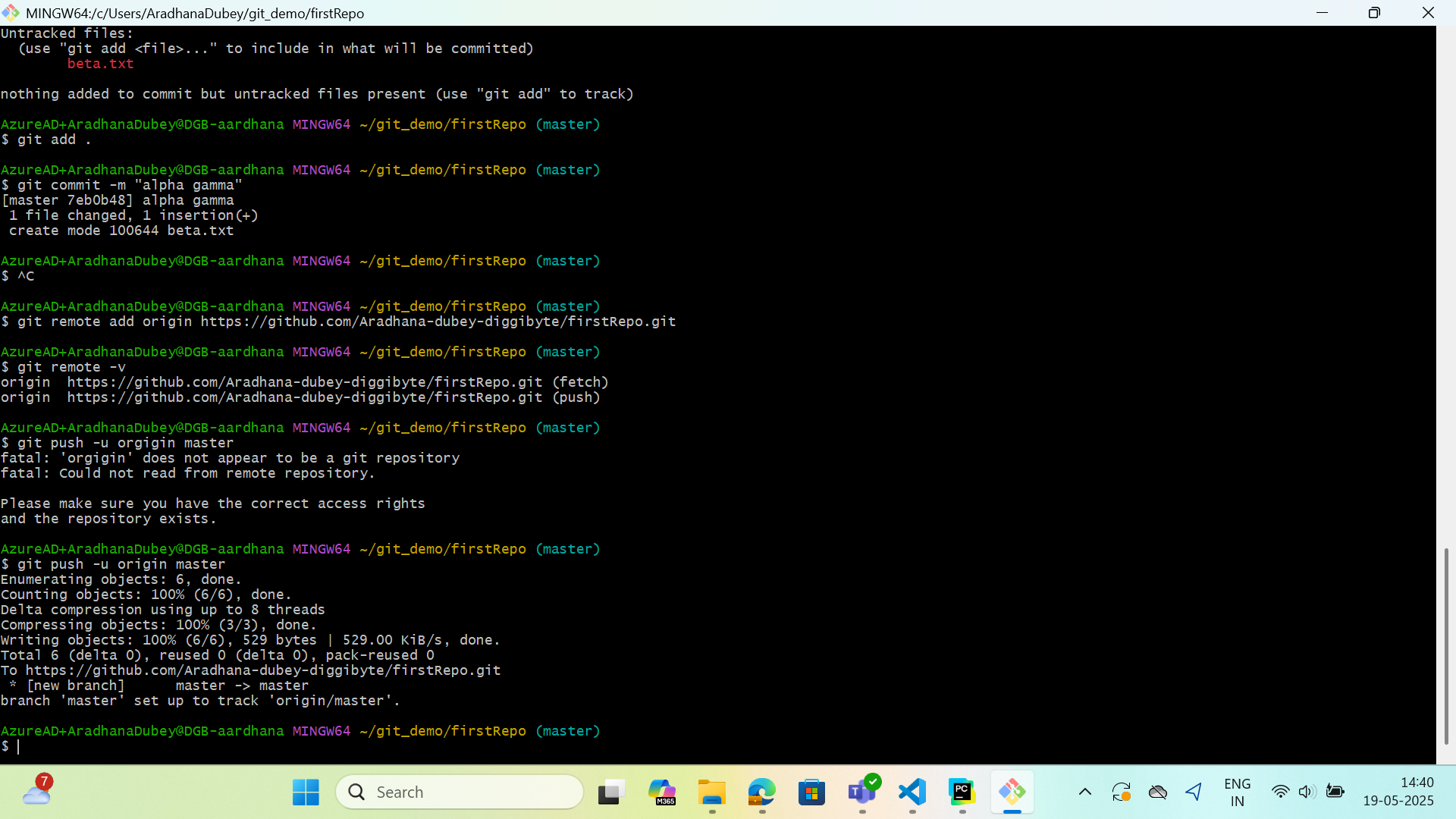
****

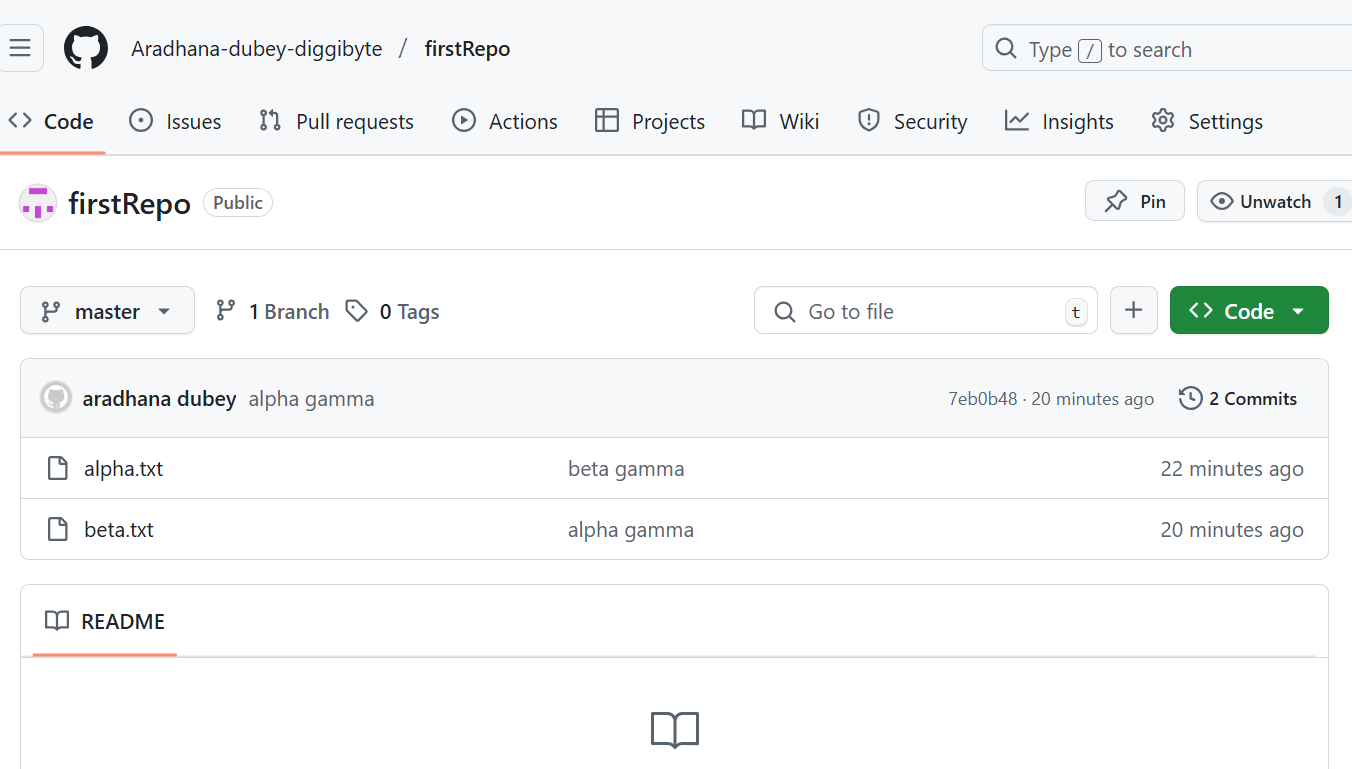
**A screenshot of a computer

AI-generated content may be incorrect.**

**A screenshot of a computer

AI-generated content may be incorrect.**

****

****

**Question 2**

Step 1: Clone the repository you have created in GitHub.

Step 2: Create a new branch using the command.

Step 3: Switch to the new branch.

Step 4: Make some changes to the code in your local copy of the repository.

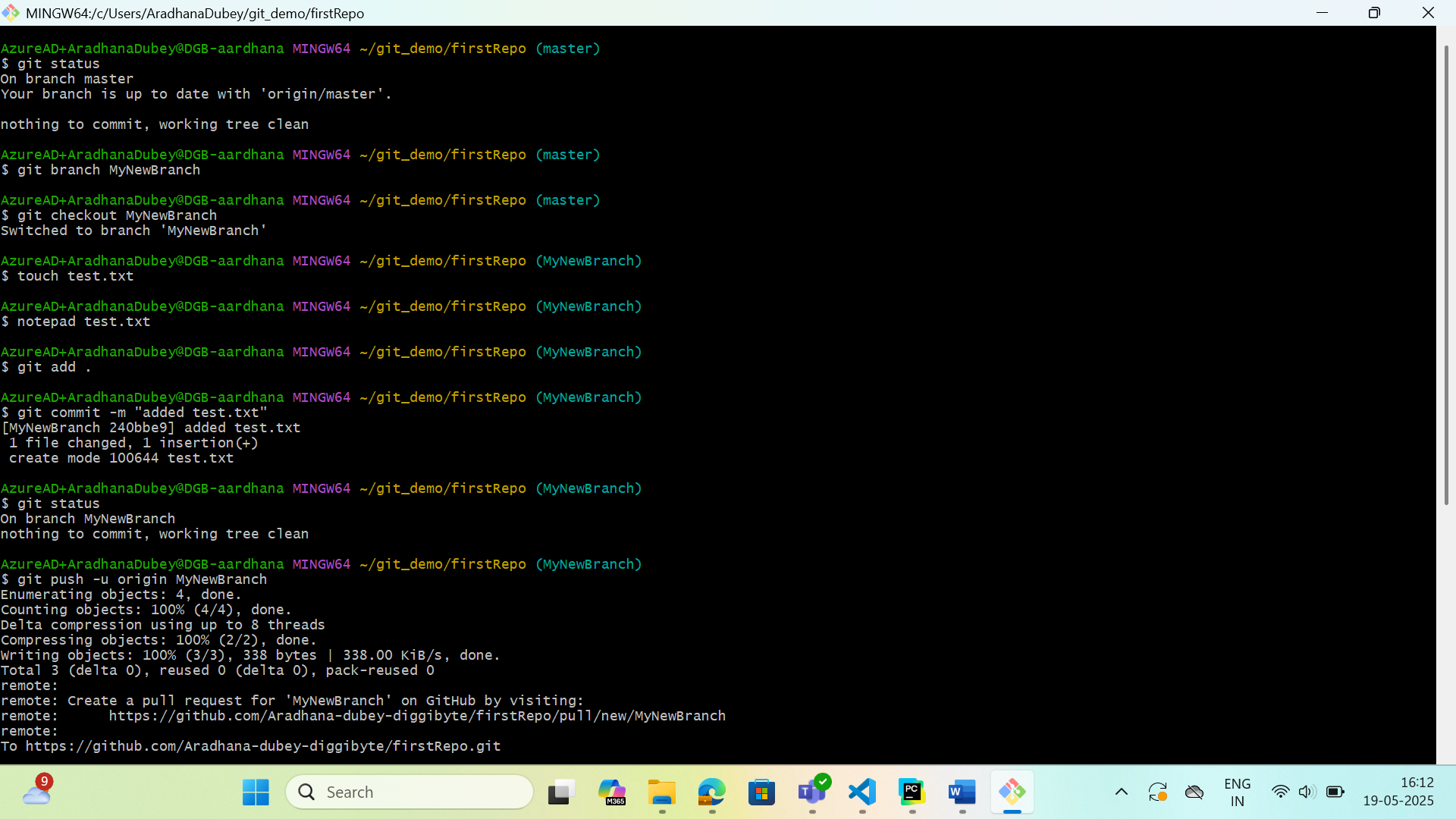
Step 5: Commit changes to the new branch.

Step 6: Switch back to the original branch

Step 7: Merge the new branch.

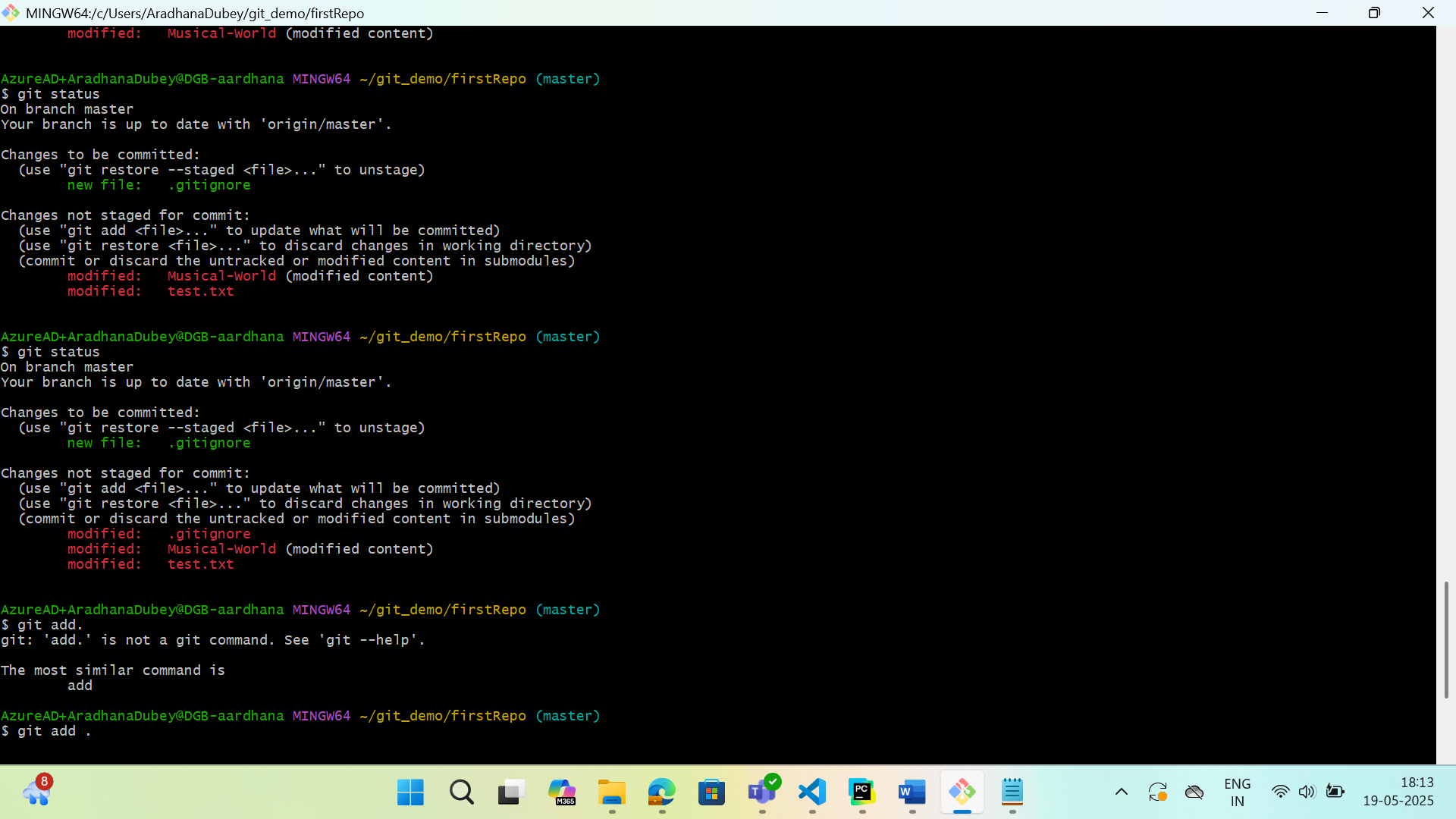
Step 8: Push changes to the original branch

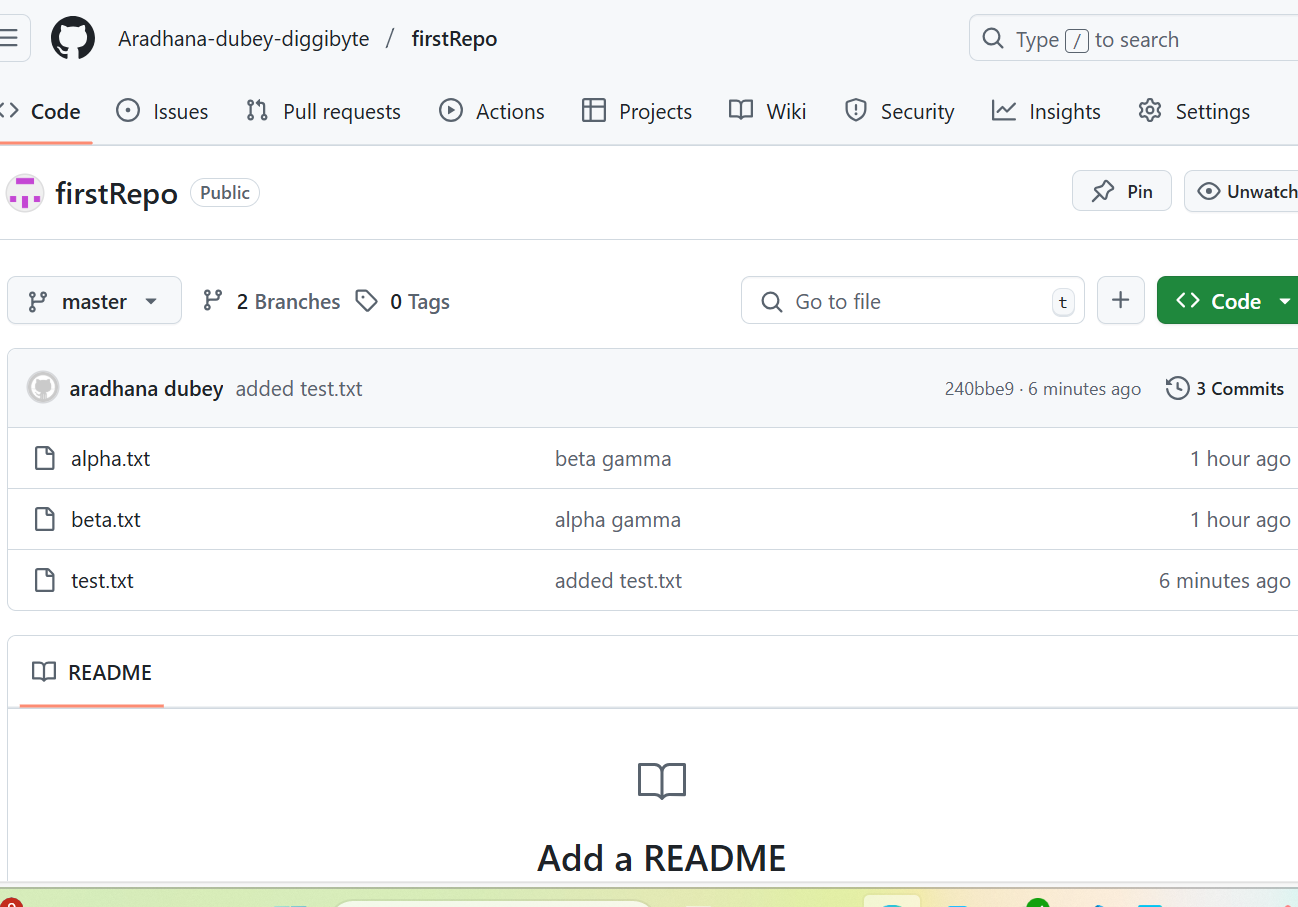
**Answer**

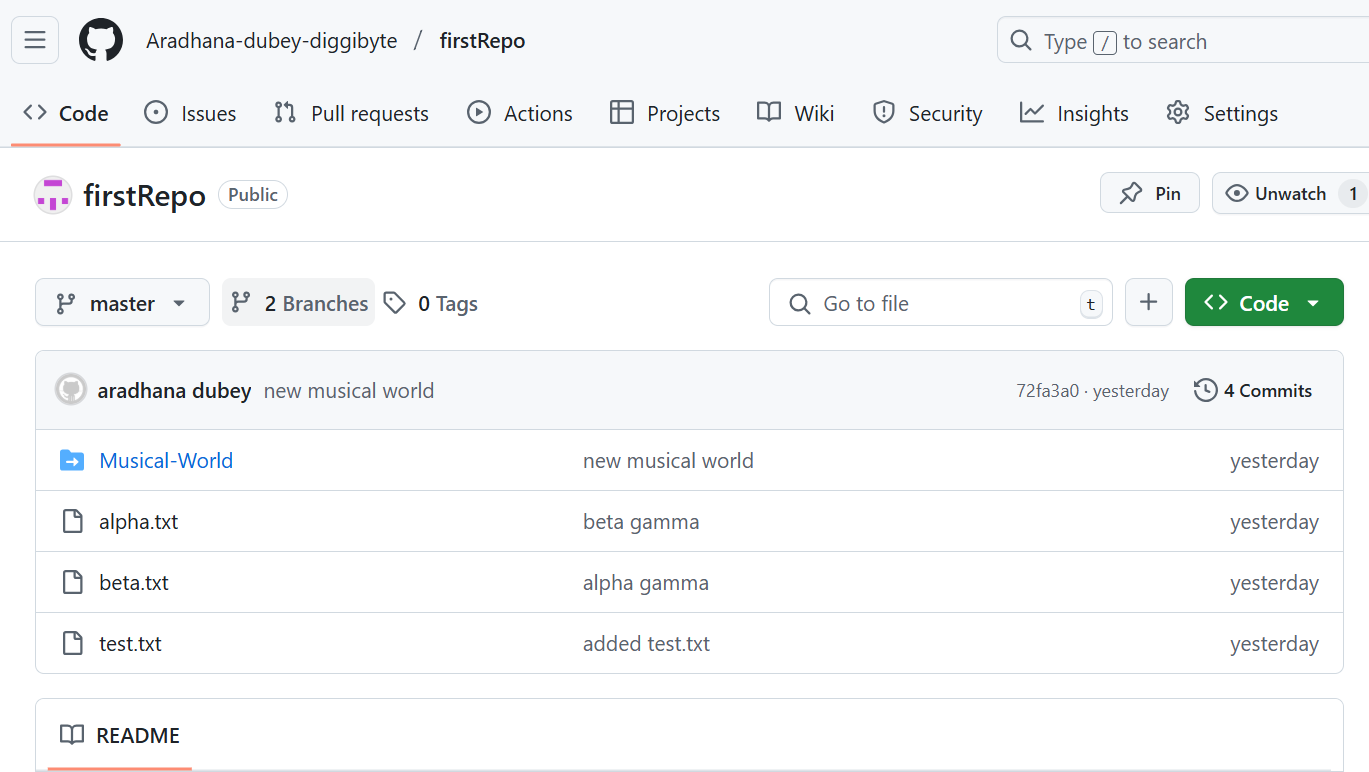
****

**A computer screen shot of a black screen

AI-generated content may be incorrect.**

****

****

****

**Question 3**

Step 1: Create a feature branch.

Step 2: Switch to the new branch.

Step 3: open the file and make some changes to it.

Step 4: Add and commit the changes to the new branch.

Step 5: Push the changes to the new feature branch.

Step 6: Create a pull request.

Step 6: As another user in the master branch make some changes to the same file.

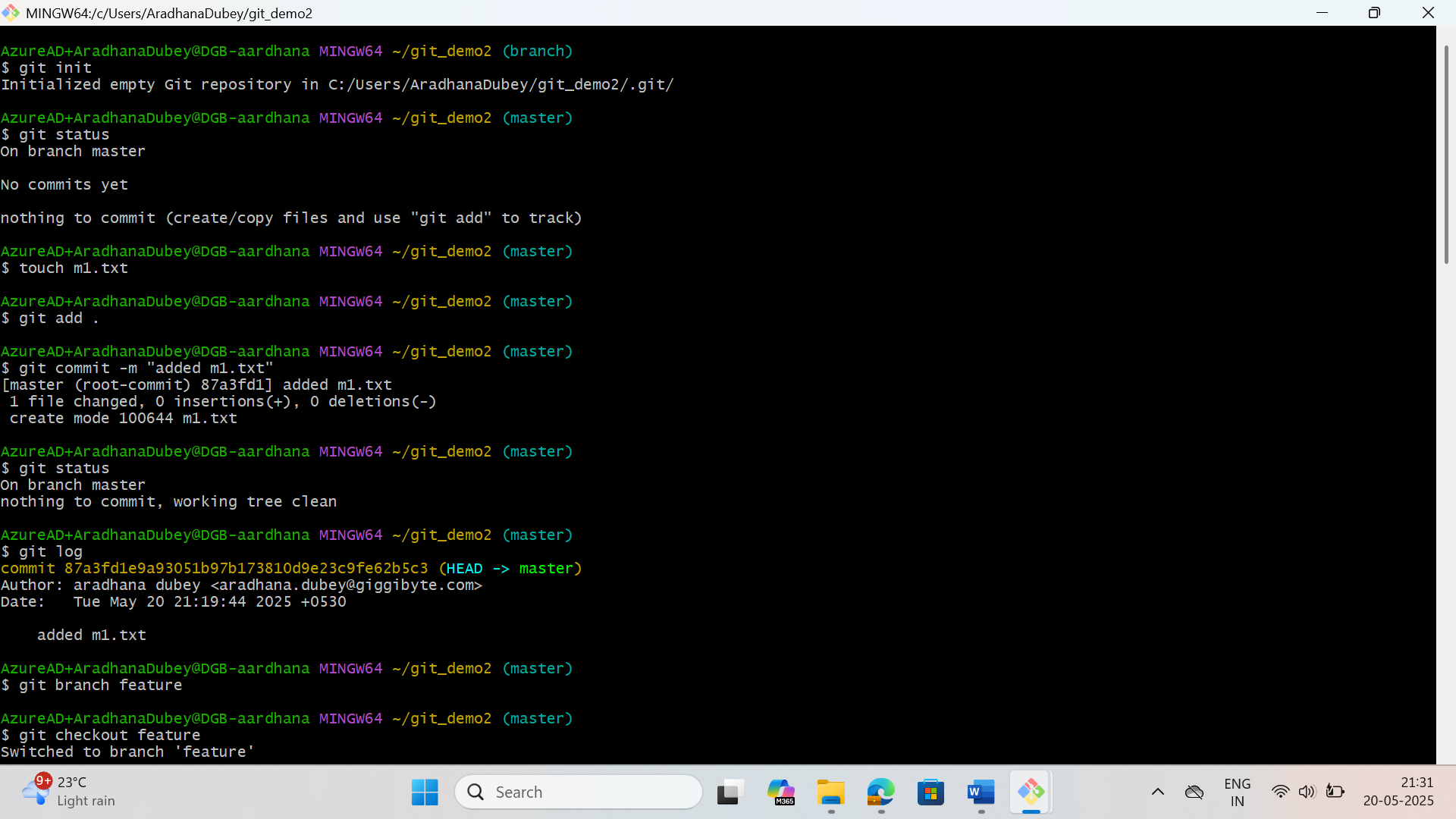
Step 7: Add and commit the changes to the master branch.

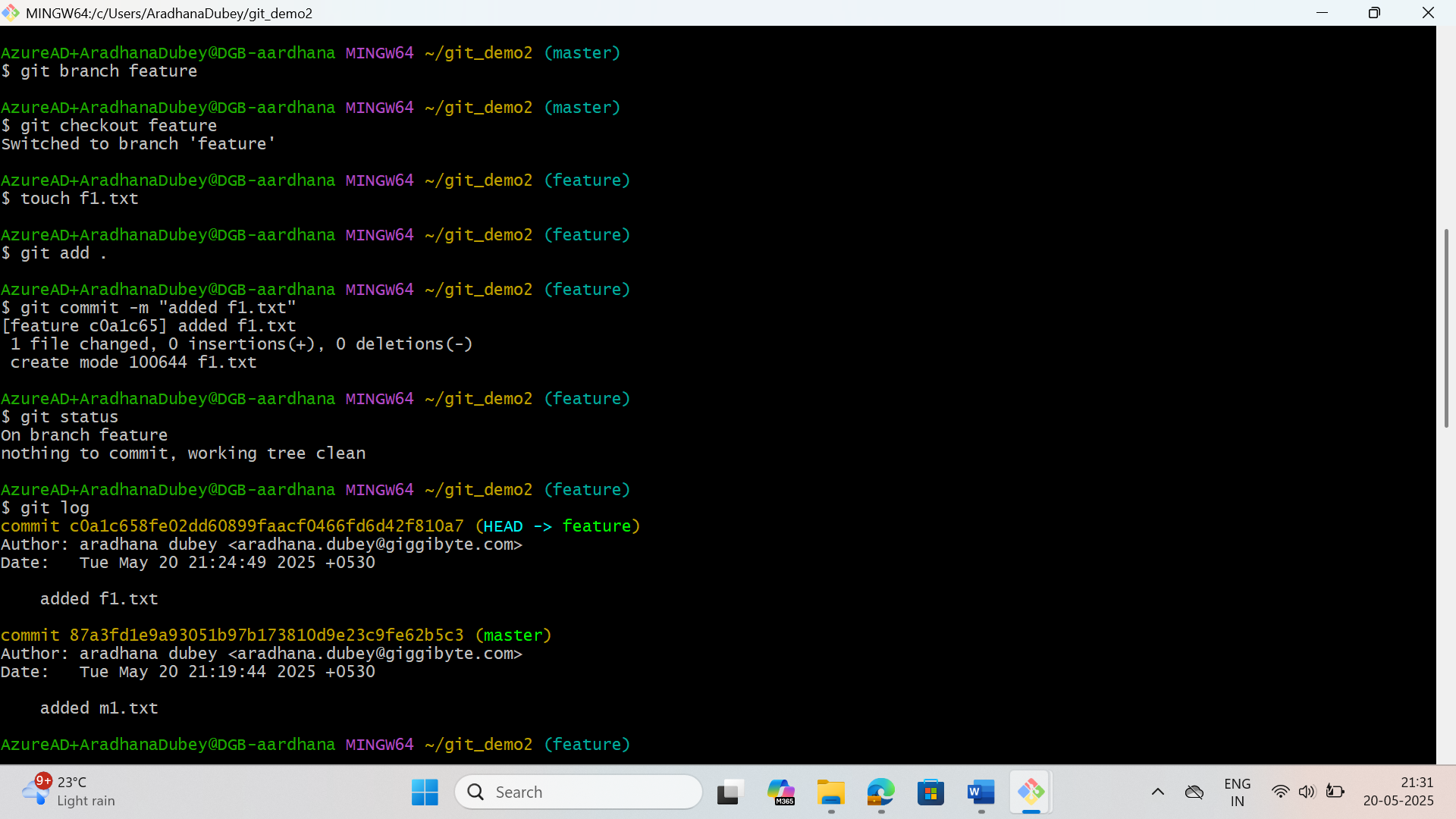
Step 8: Push the changes to the master branch.

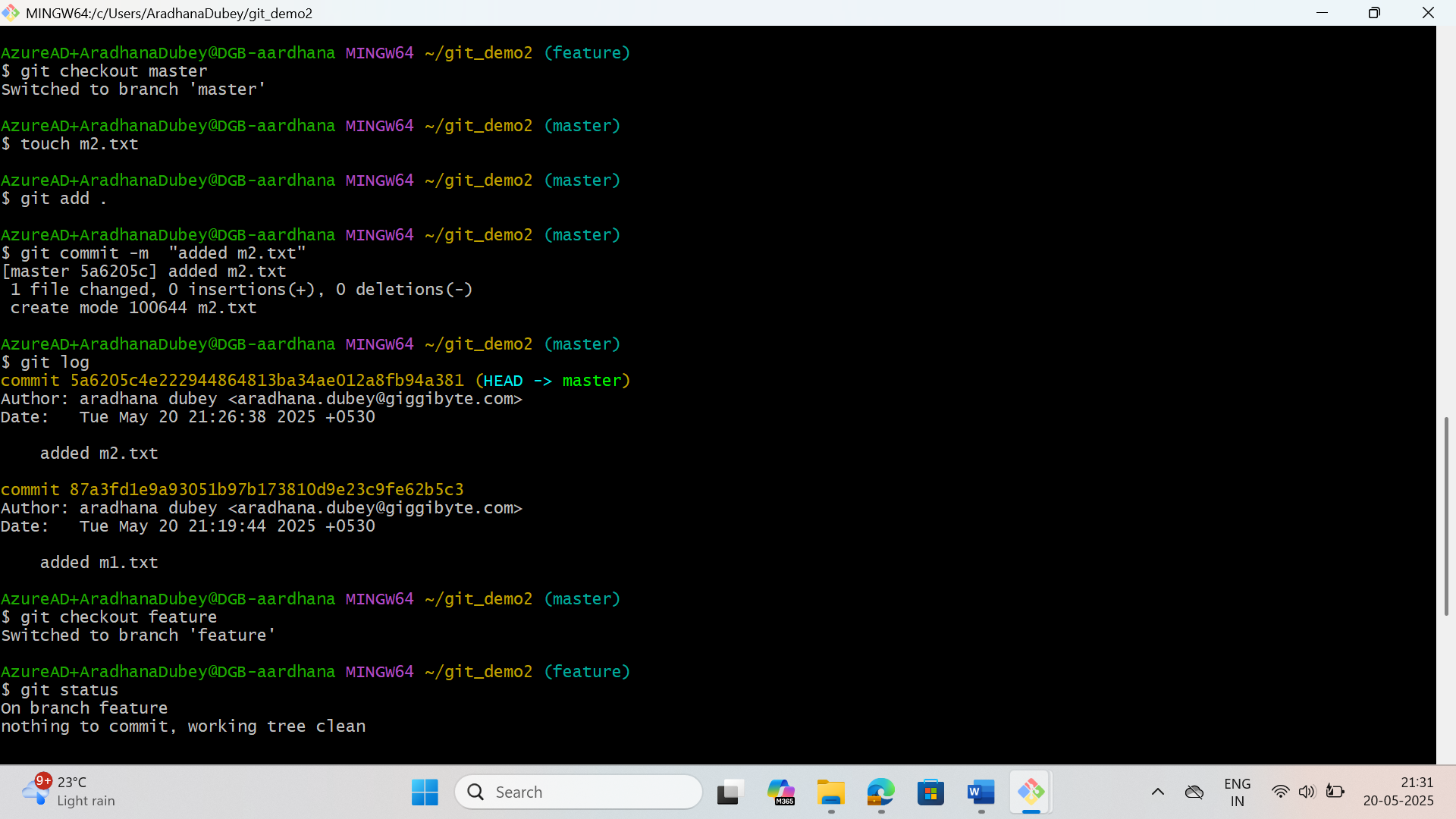
Note: There will be a conflict in the pull request, how do we resolve it??

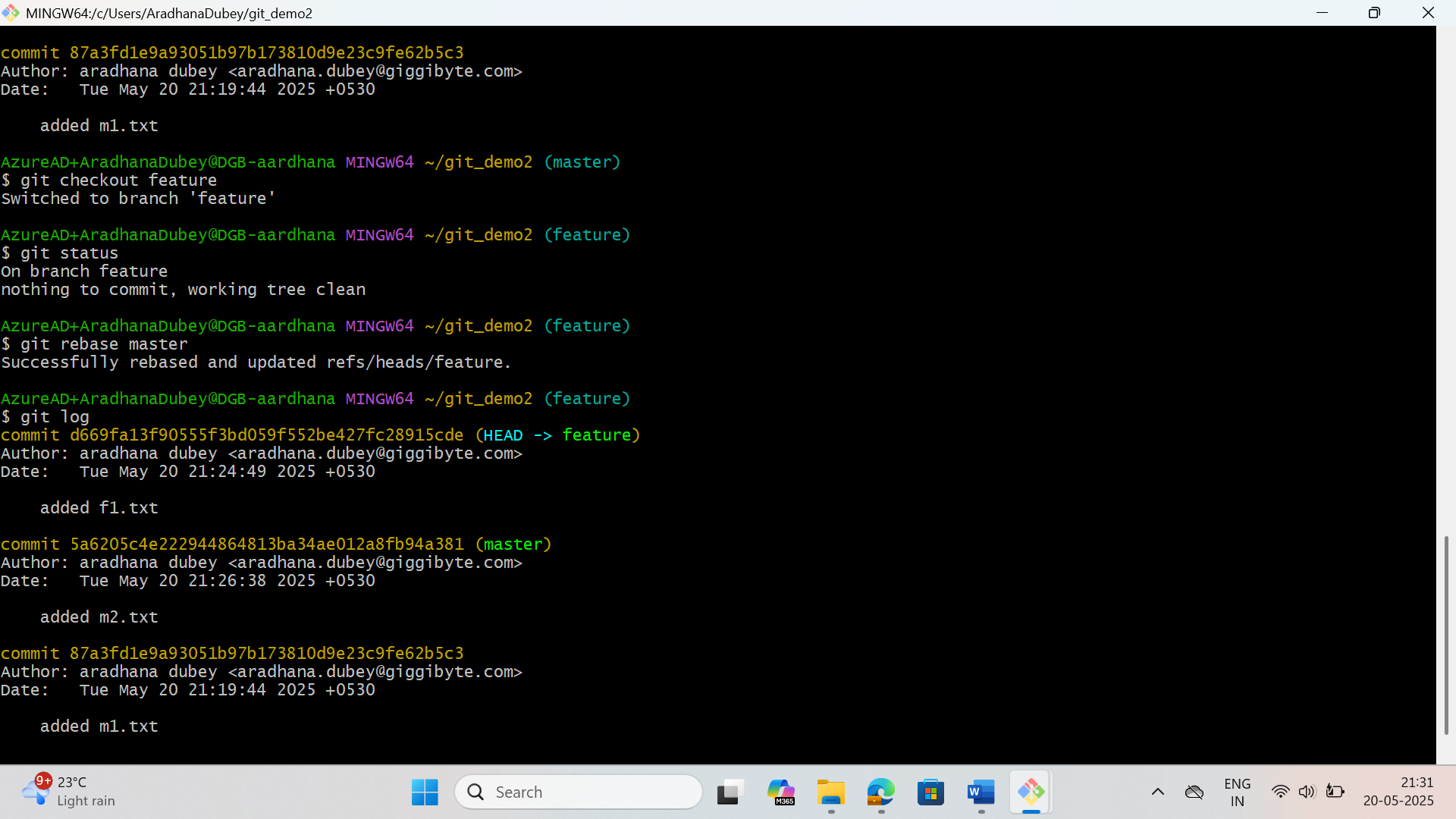
Hint: git rebase

**Answer**

****

****

****

****

**Question 4**

Step 1: Step 1: Create a feature branch.

Step 2: Switch to the new branch.

open the file and make some changes to it.

Add and commit the changes to the new branch.

open the same file and make some changes to it.

Add and commit the changes to the new branch.

open the same file and make some changes to it.

Add and commit the changes to the new branch.

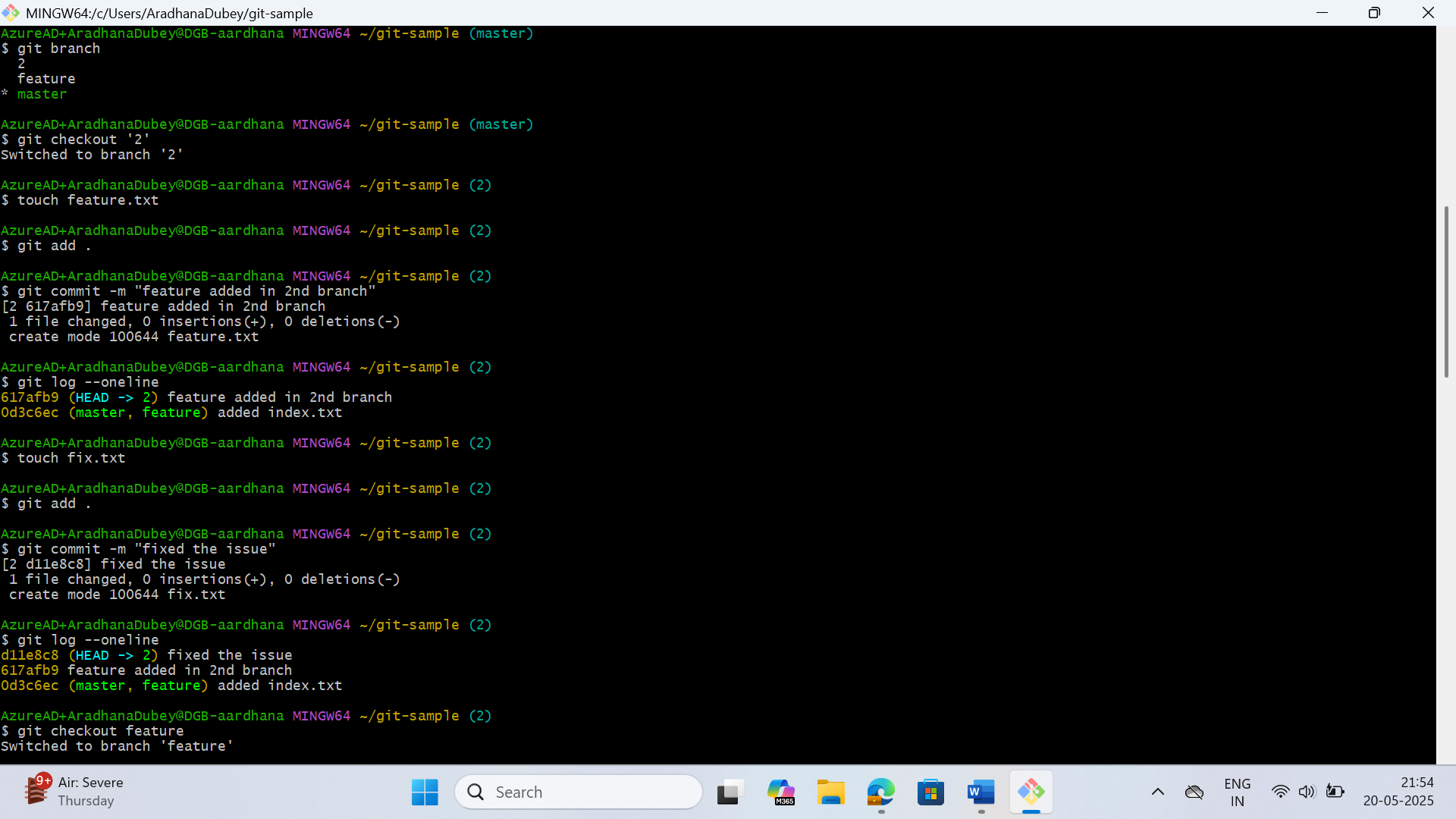
Step 3: Identify the commit or commits that you want to "cherry-pick"(Note the hash of the commit or commits that you want to "cherry-pick")

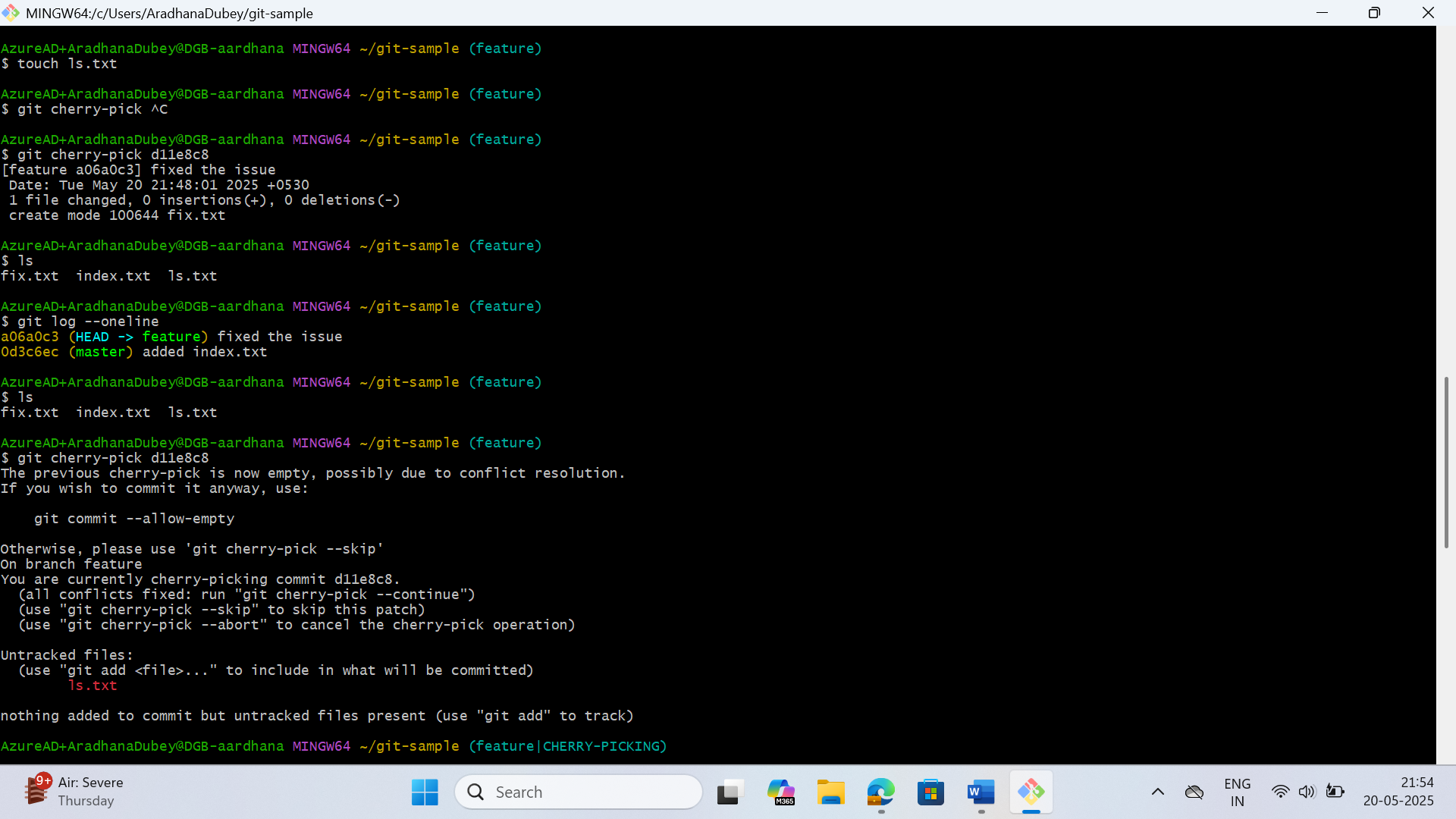
Step 4: Use the "git checkout" command to switch to the branch where you want to apply the changes.

Step 5: Use the "git cherry-pick" command followed by the commit hash(es) that you want to apply

**Answer**

****

****

****

****

**Question 5**

Step 1: Step 1: Create a feature branch.

Step 2: Switch to the new branch.

open the file and make some changes to it.

Add and commit the changes to the new branch.

open the same file and make some changes to it.

Add and commit the changes to the new branch.

open the same file and make some changes to it.

Add and commit the changes to the new branch.

Step 3: Use the "git log" command to view the commit history and identify the commit to which you want to reset.

Step 4: Use the "git reset" command followed by the desired reset type and the commit hash

Step 5: Verify that the reset was successful by using the "git log" command again.

Step 6: Use the "git log" command to view the commit history and identify the commit that you want to reverse.

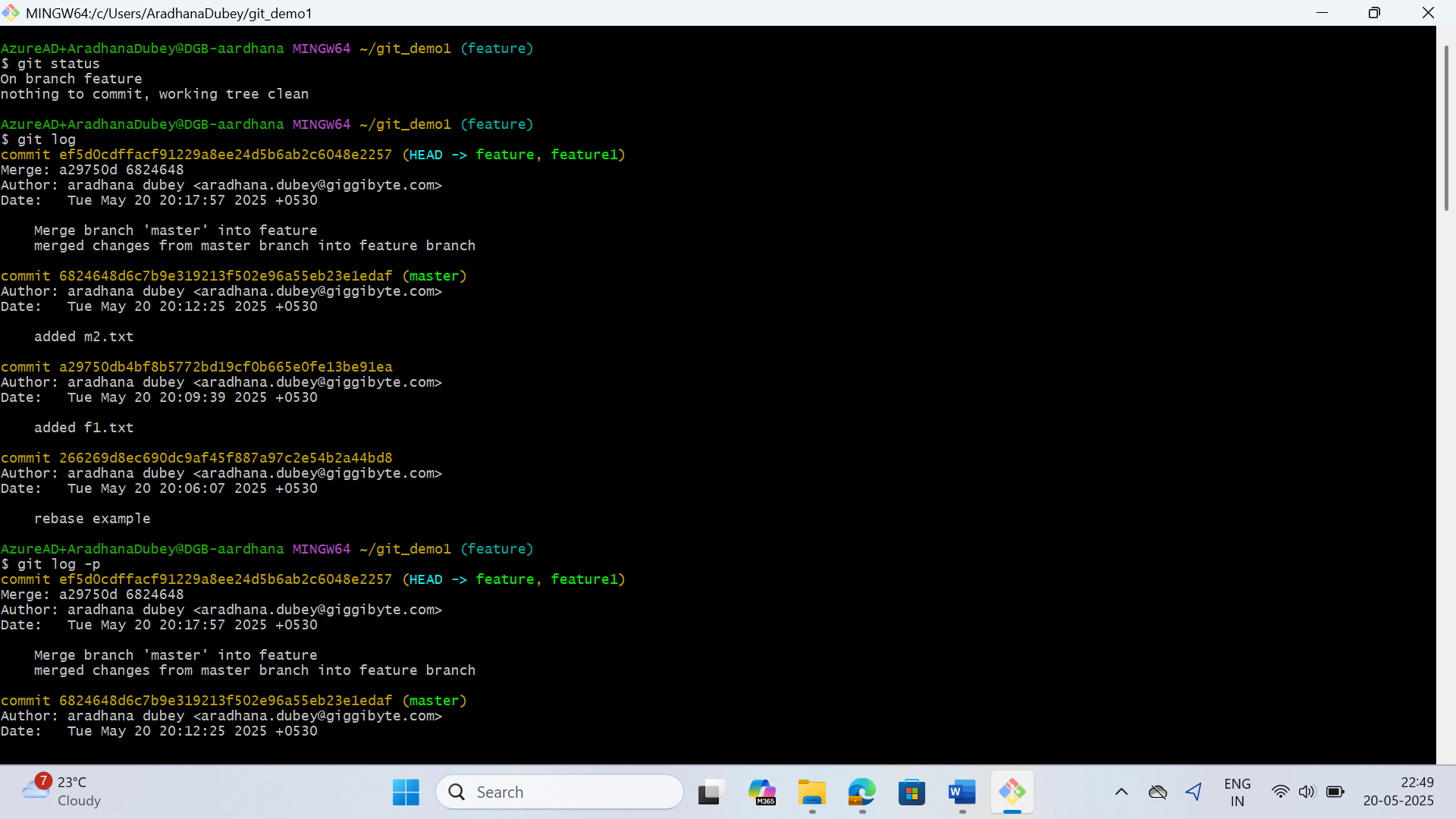
Step 7: Use the "git revert" command followed by the commit hash or reference to which you want to revert. (Hint: git revert <commit hash>)

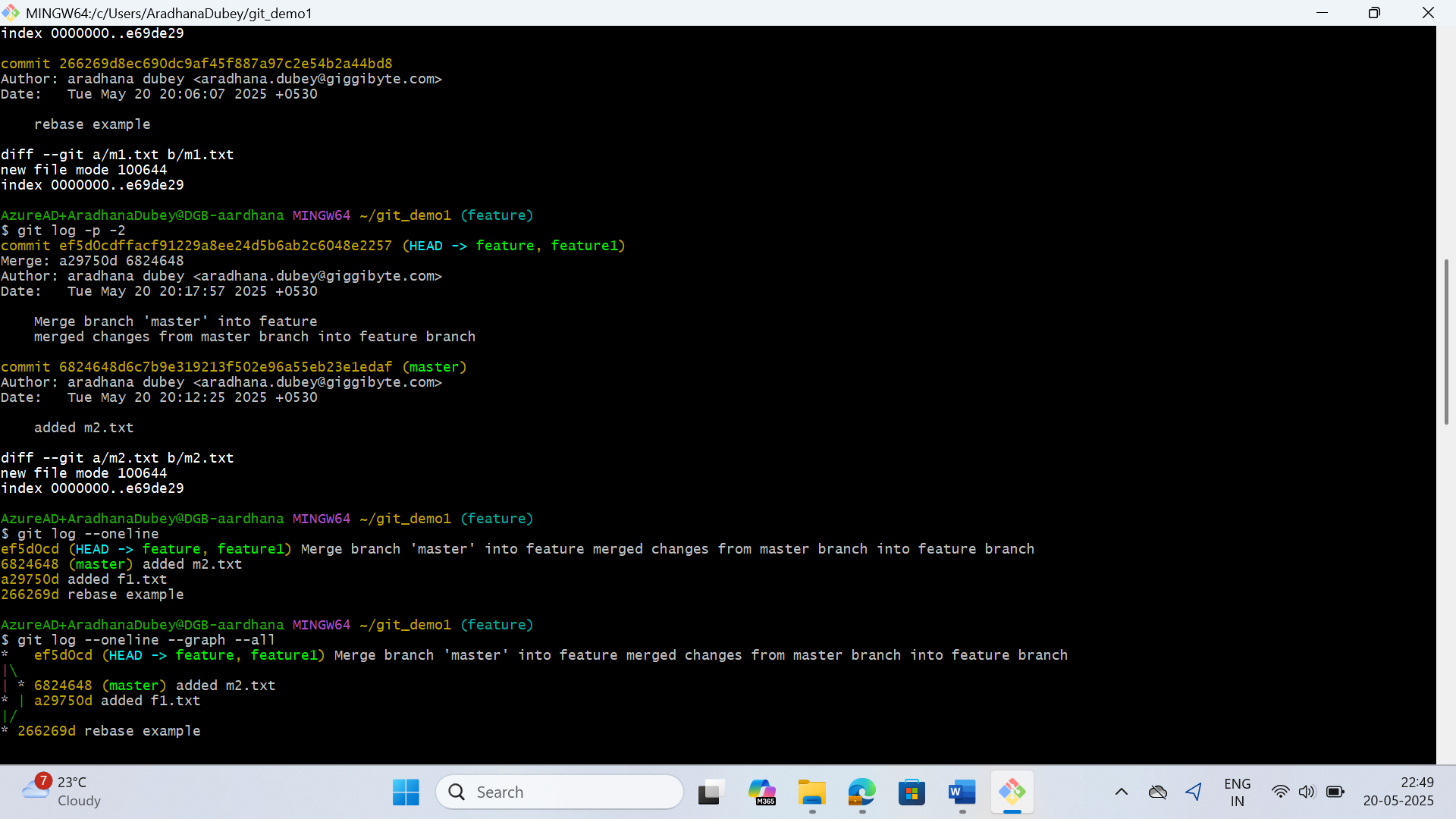
Step 8: Verify that the revert was successful by using the "git log" command again.

Note: Identify the difference between git log after git reset and git r evert.

**Answer**

**Git logs commands :**

****

****

**A screenshot of a computer screen

AI-generated content may be incorrect.**

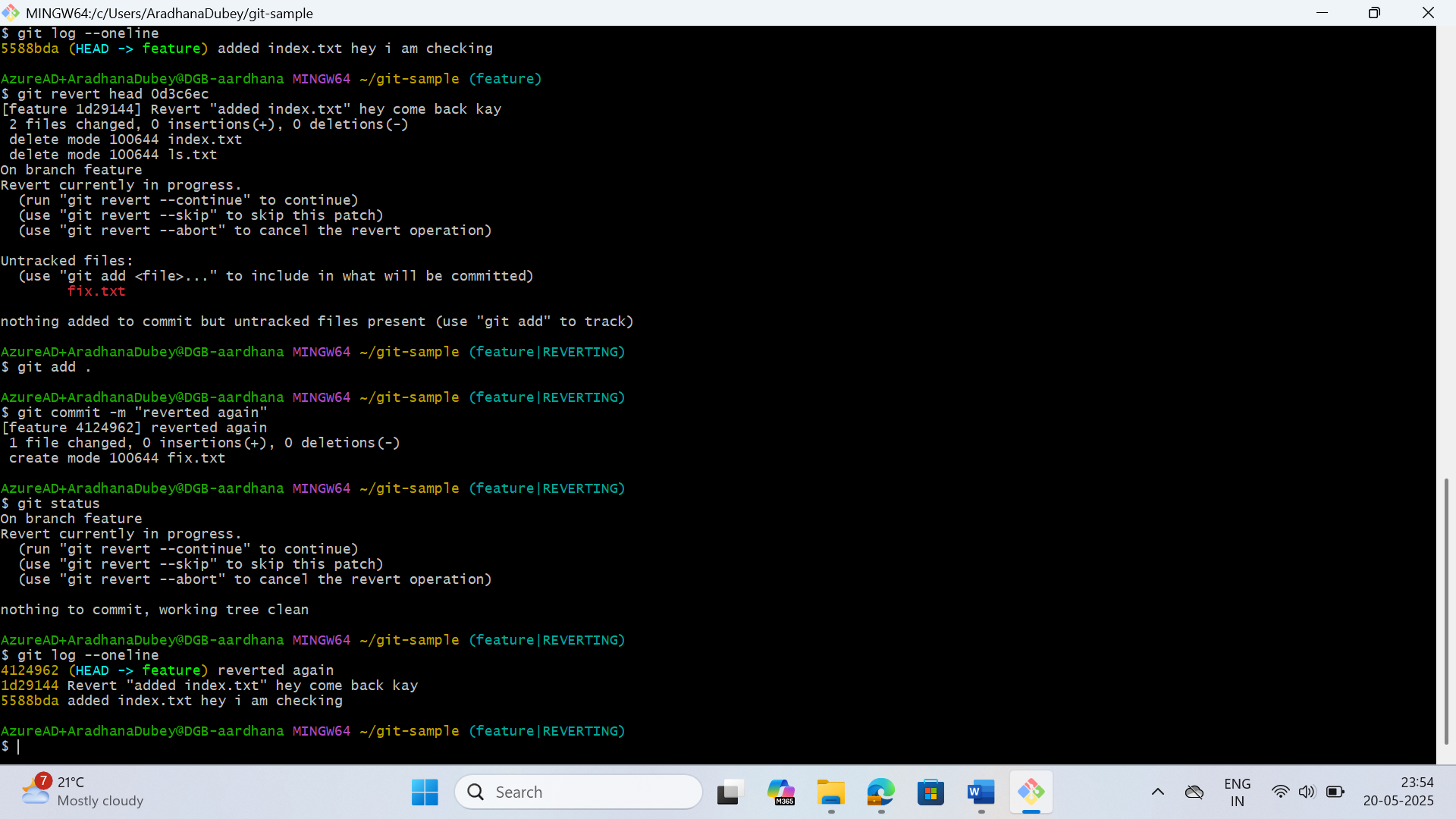
**Git reset:**

****

**A screenshot of a computer screen

AI-generated content may be incorrect.**

**Git revert:**

****