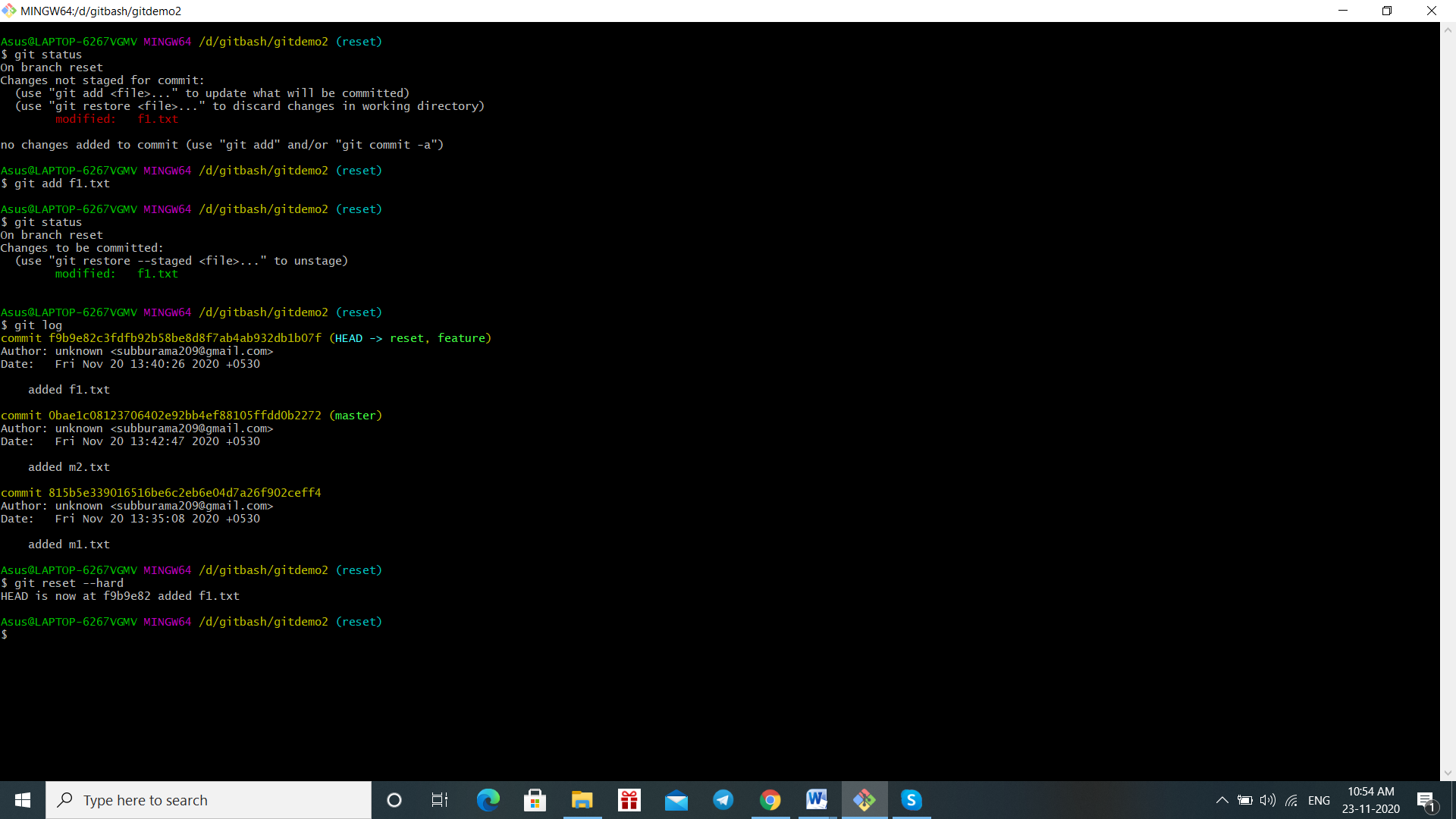
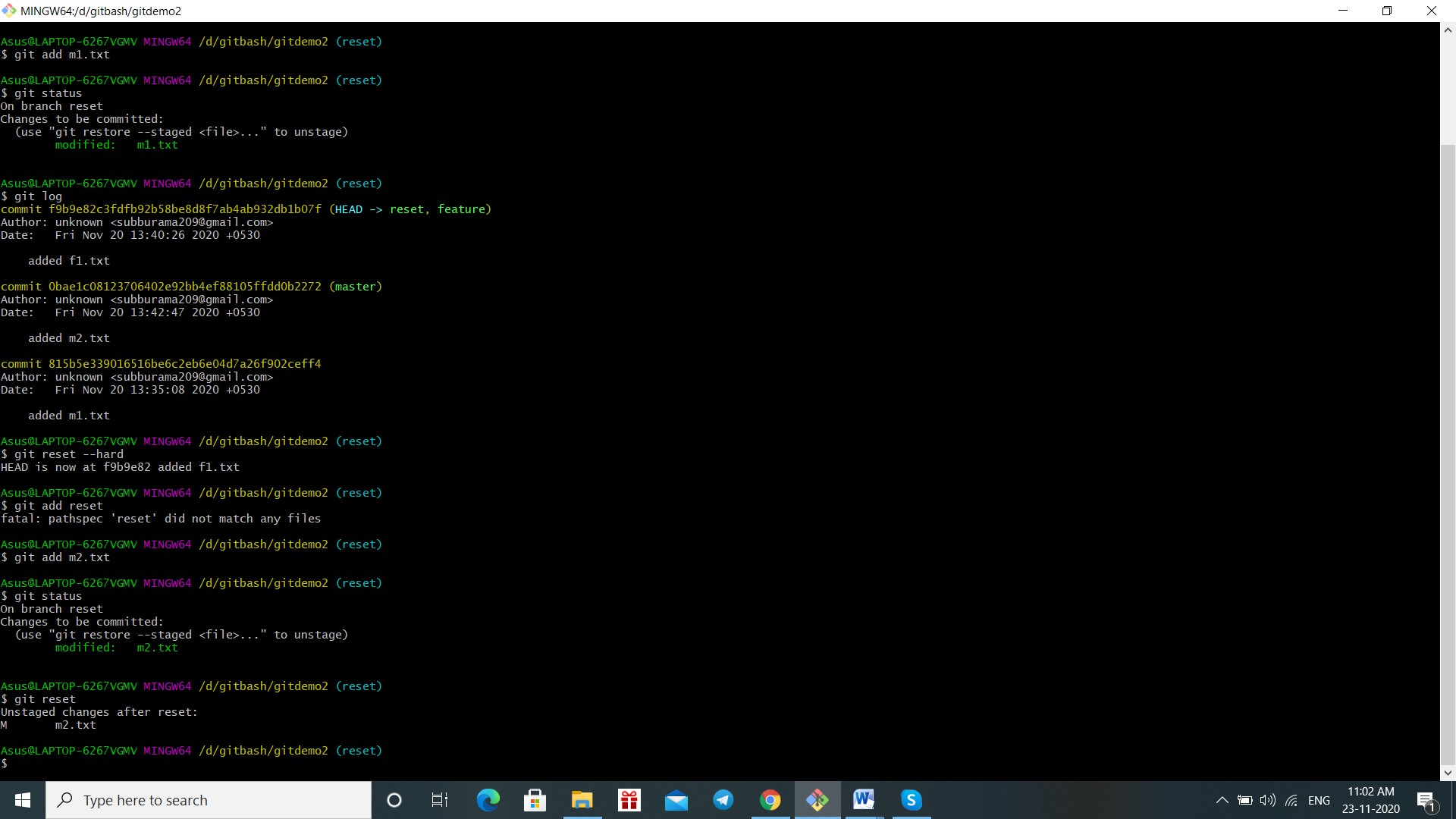
**GIT RESET HARD:**

* The --hard option changes the Commit History
* Any previously pending commits to the Staging Index and the Working Directory gets reset to match Commit Tree.
* It means any expected work will be lost

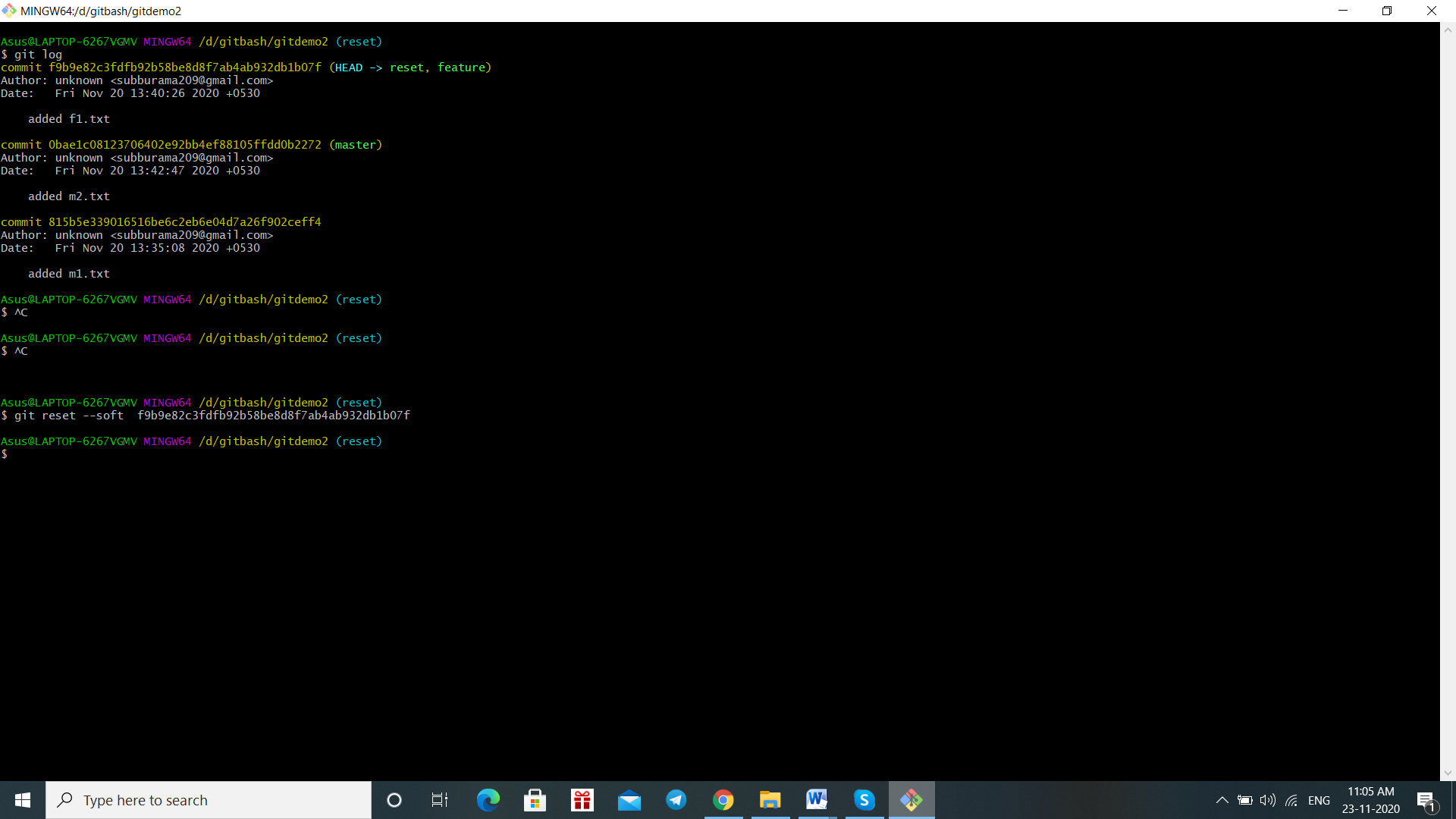


**GIT RESET MIXED:**



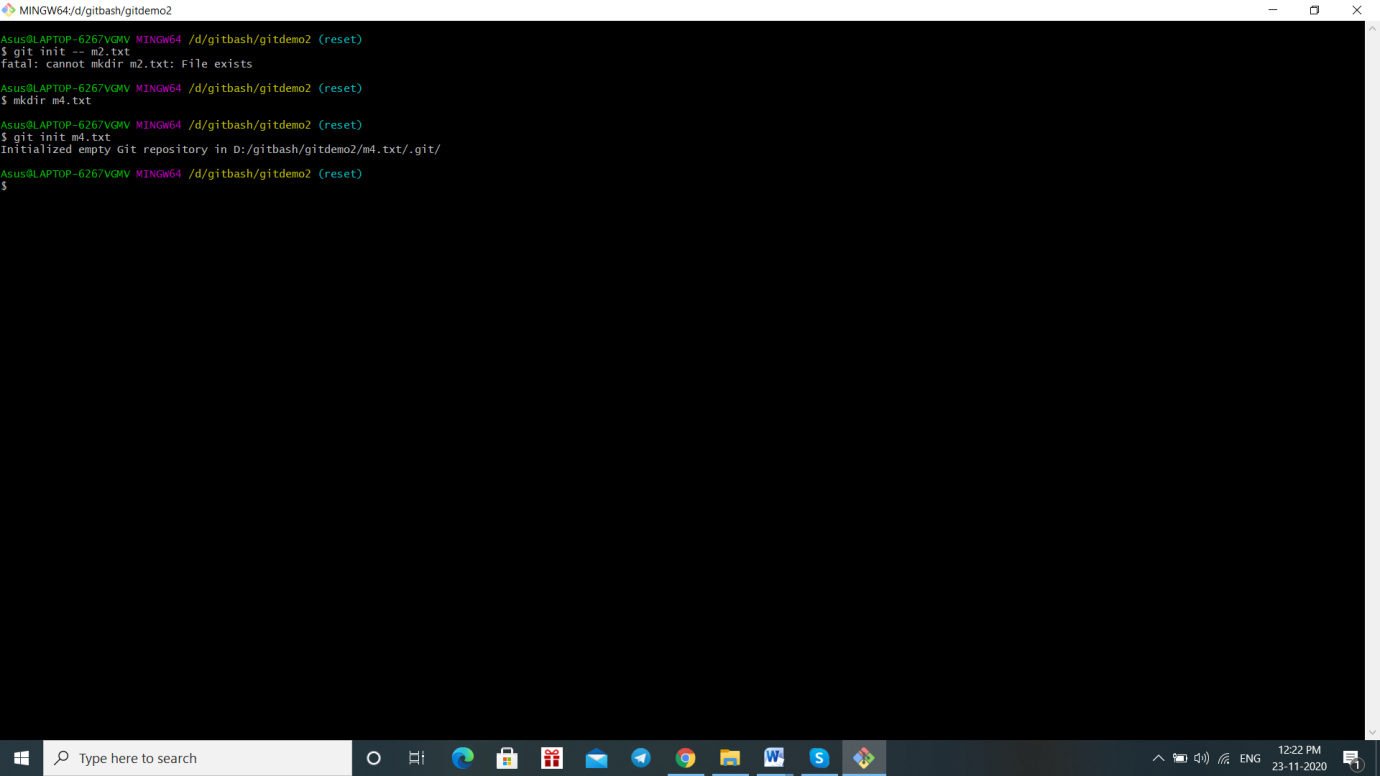
**GIT RESET SOFT:**

It does not touch the index file and reset the head options.so the changes occur in head.



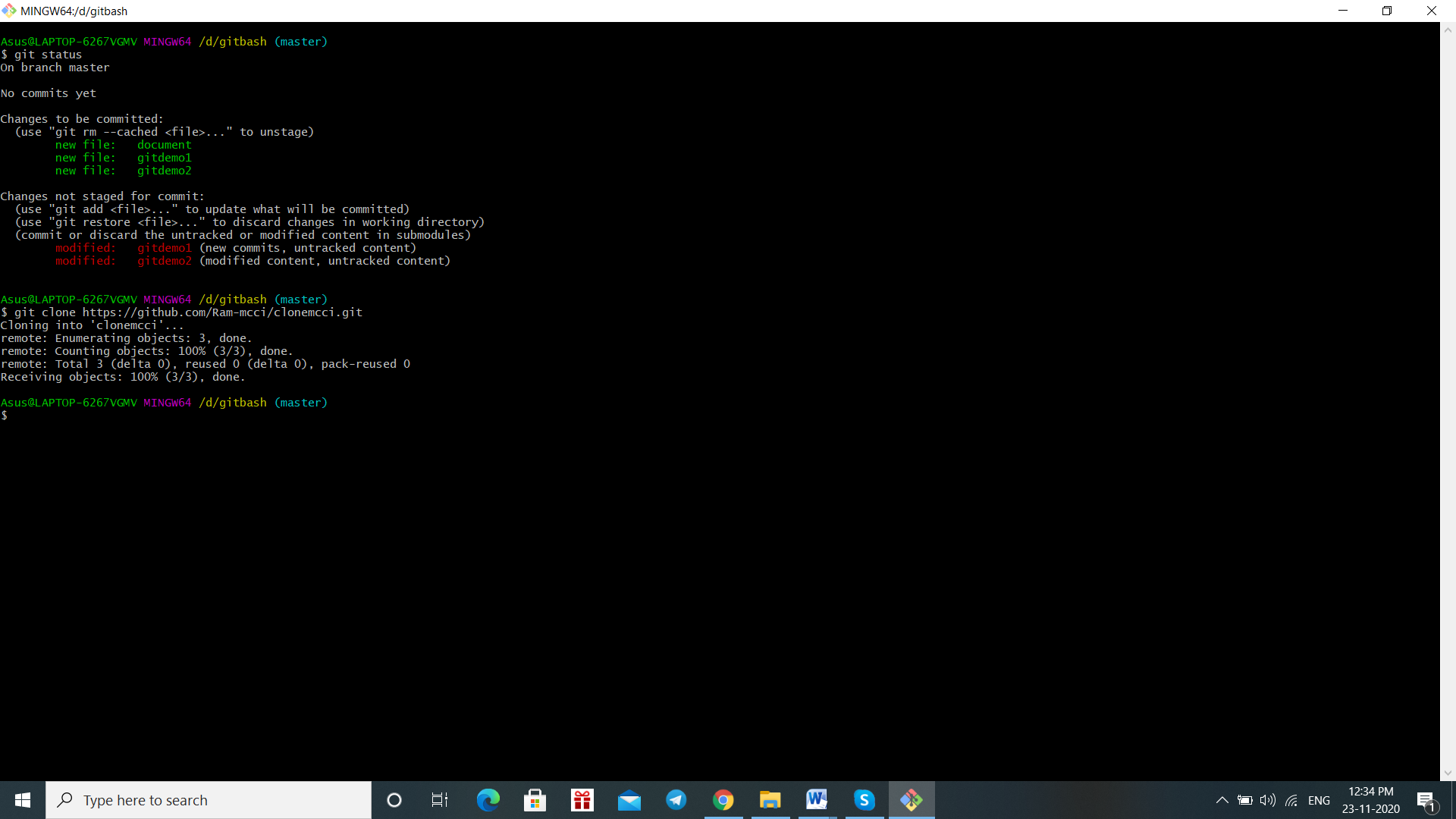
**GIT INIT:**

* Git init is a command creates a new git repository and also other command will not obtain outside the init
* It is also used to convert the existing project to git repository



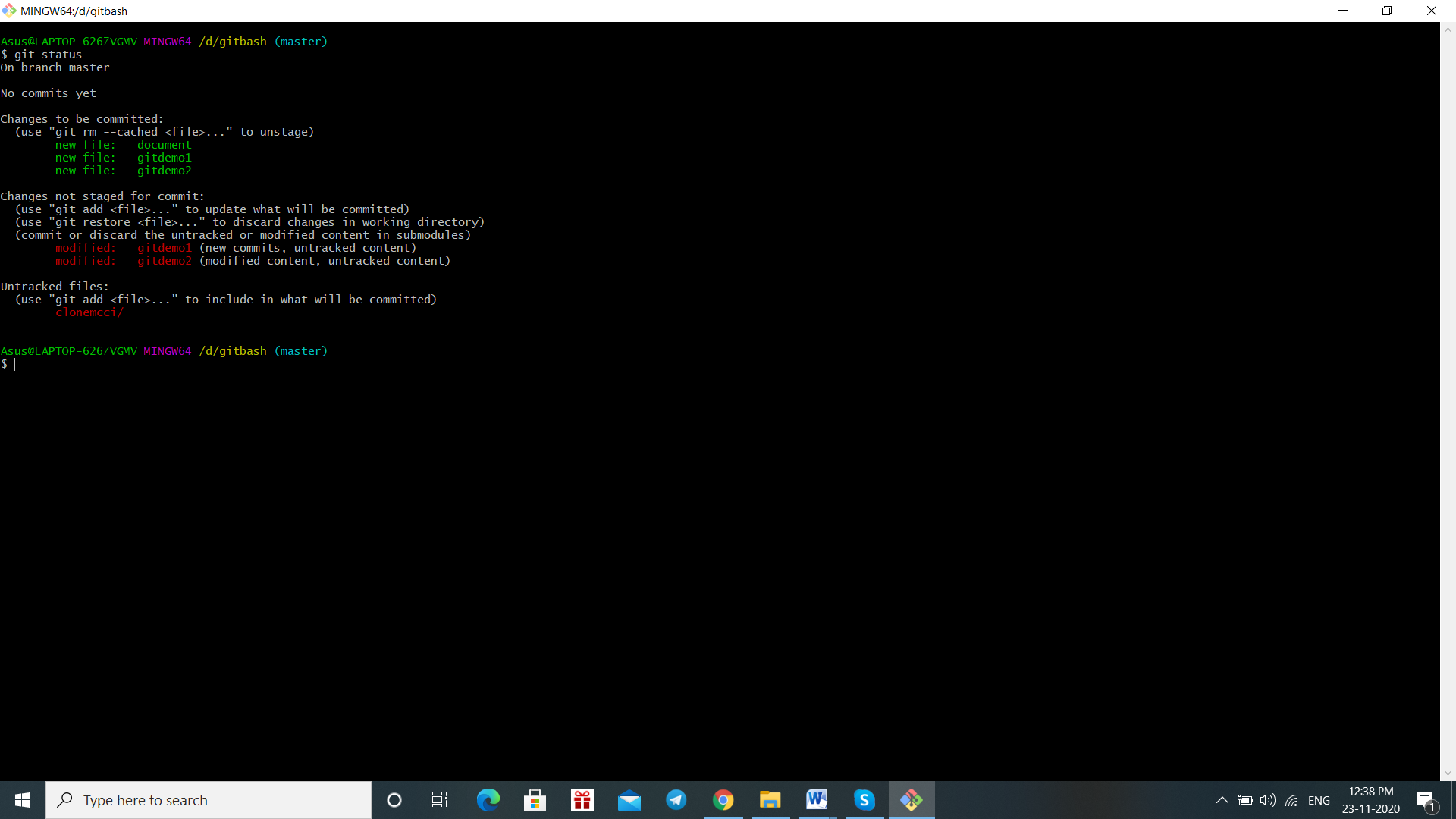
**GIT CLONE:**

* It is used to initiate a new repository and it is dependent on git init
* Generally git clone is used to create a copy of the existing repository



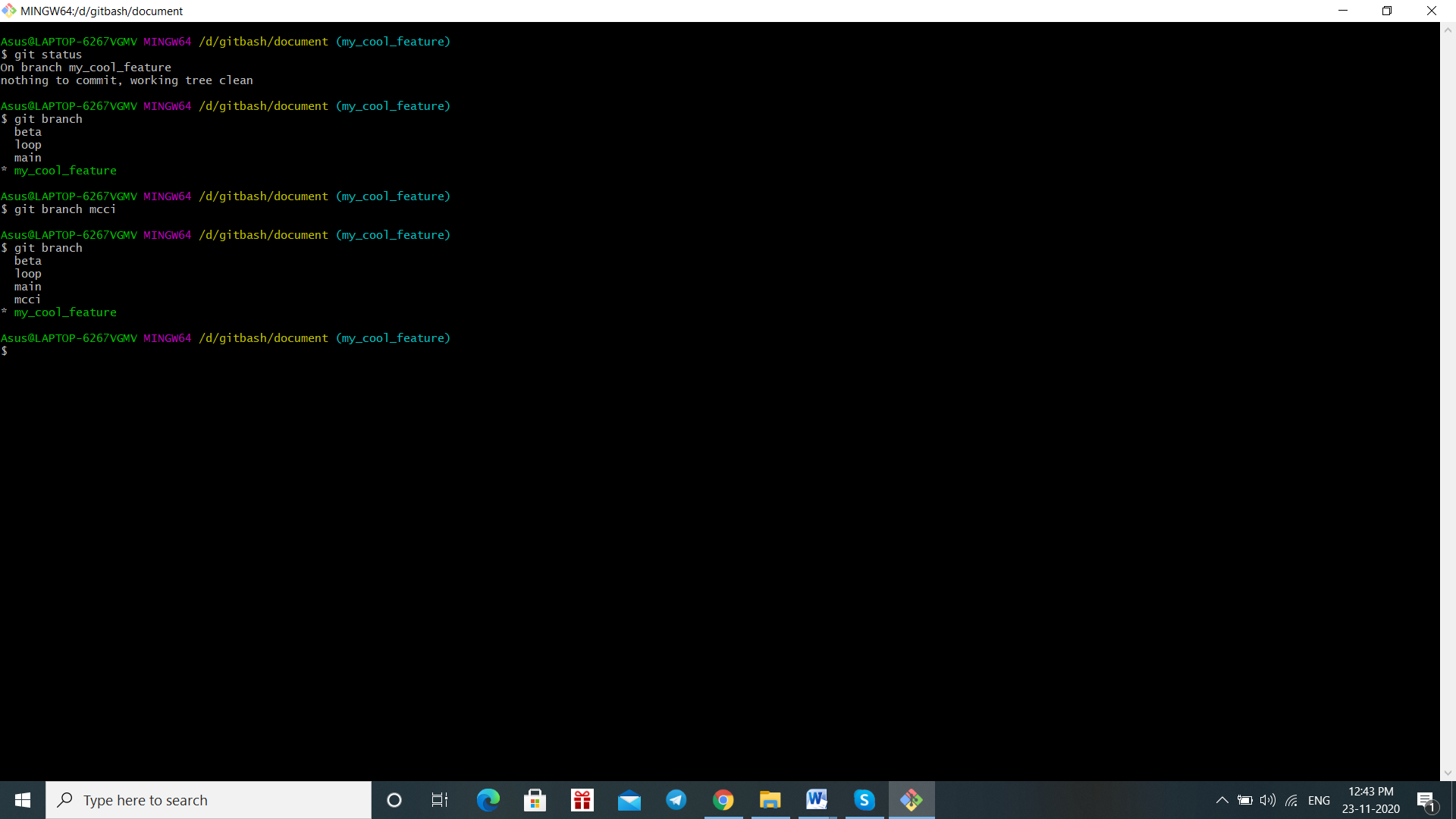
**GIT STATUS:**

* It is nothing but it shows the state of working directory and staging area.



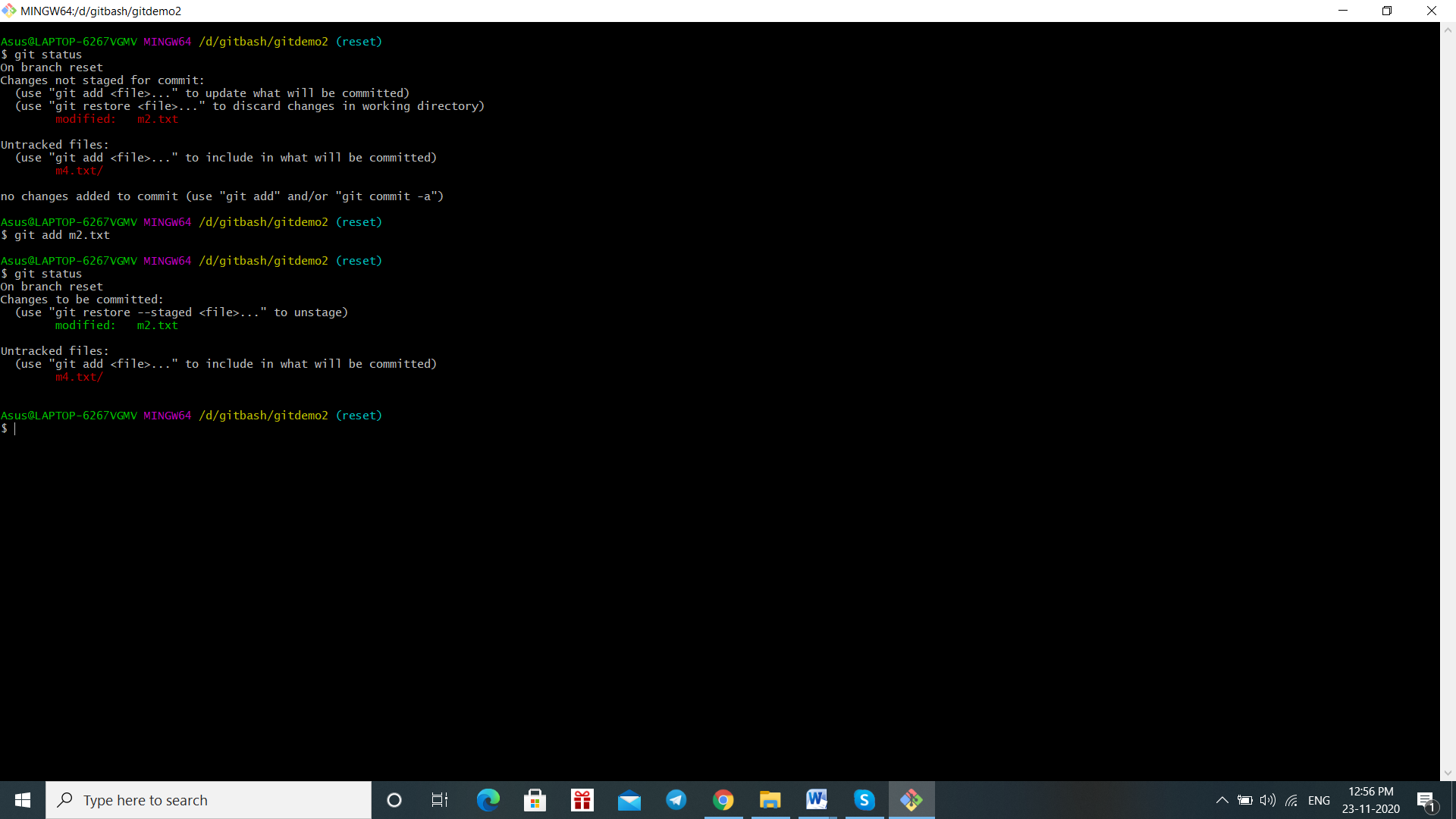
**GIT BRANCH:**

* Branches are used to test the changes the code without affecting the main code



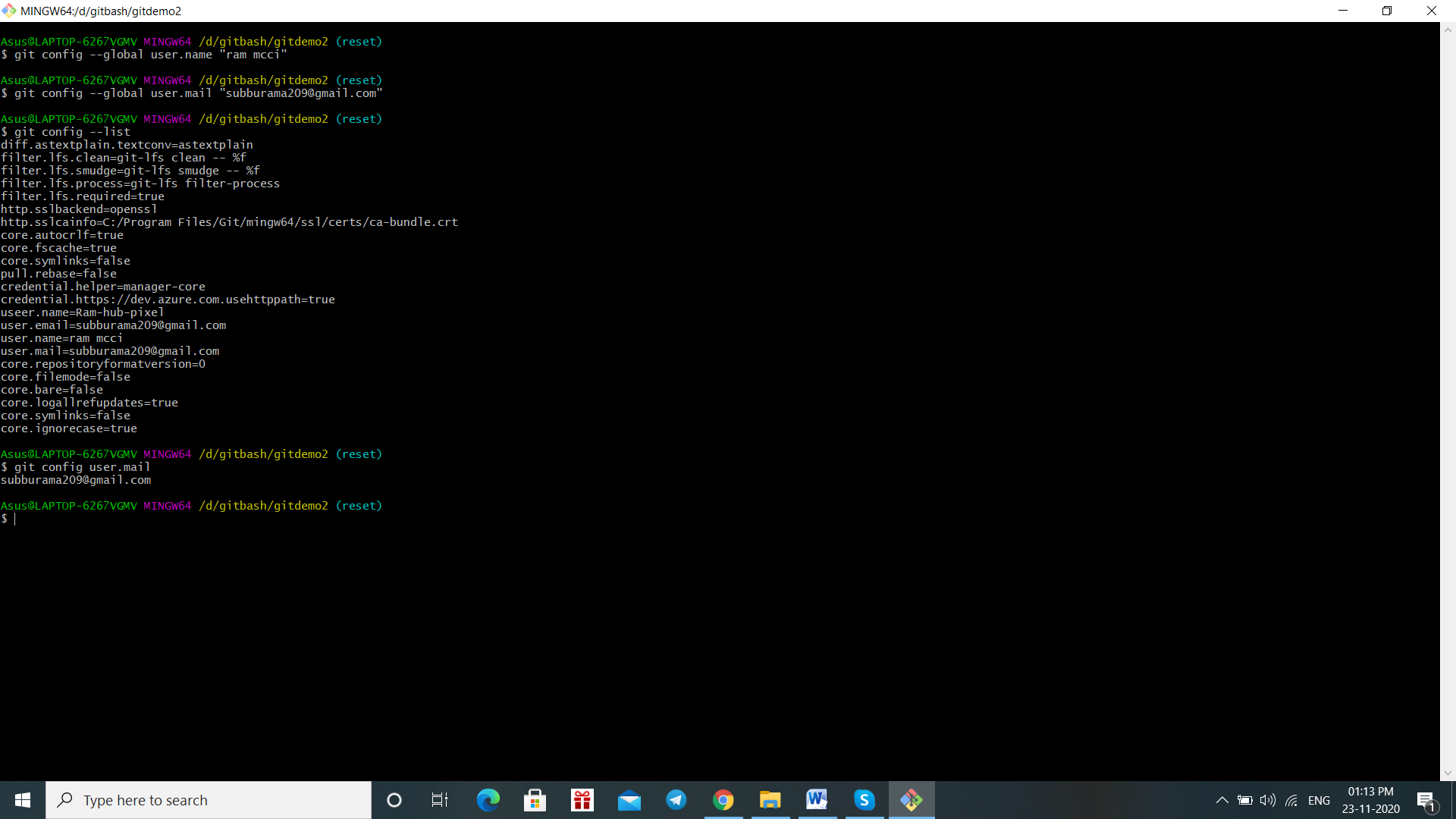
**GIT ADD:**

* It is used to add change in working to the statging area
* It tells git we want to add update in next commit and also it does not affect repository



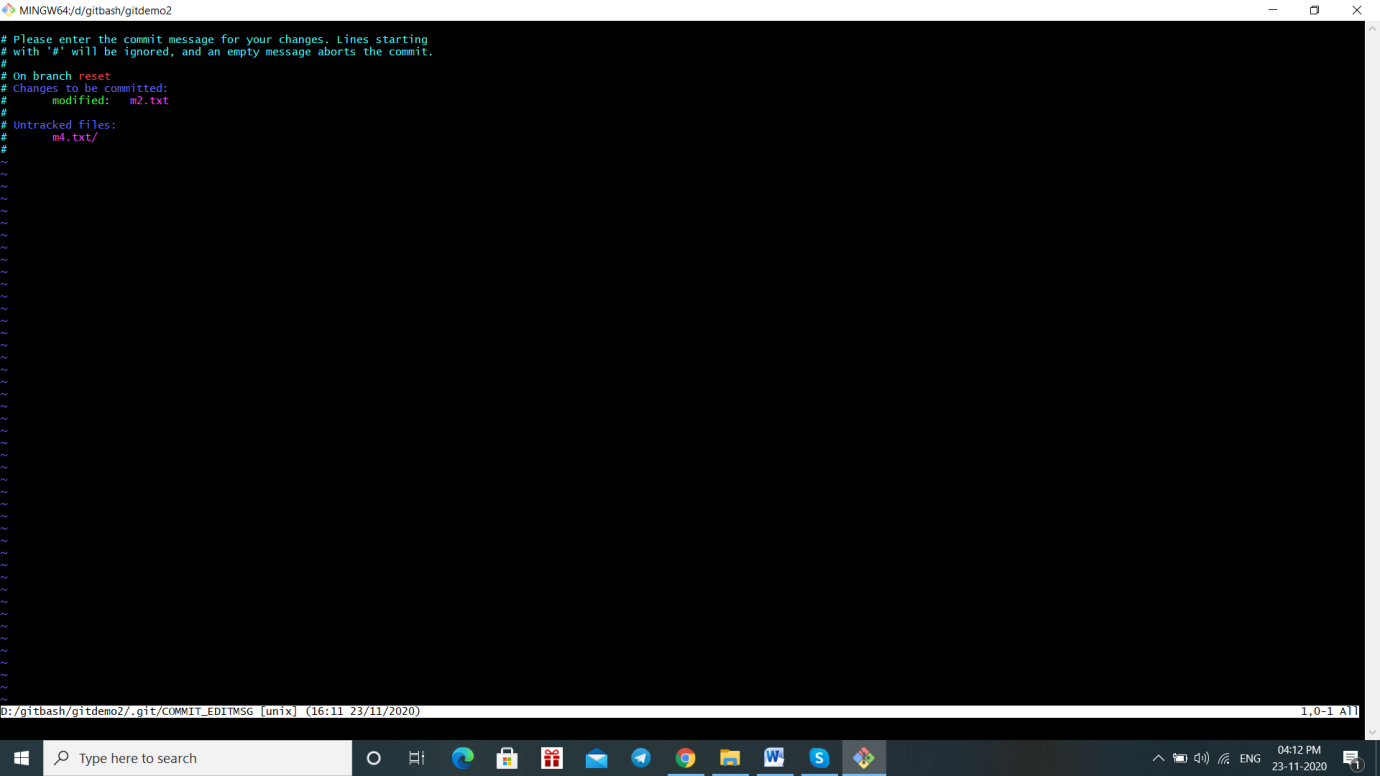
**GIT CONFIG:**

* It is used to set git configuration values on a local or global projectlevel.
* It is used to call username and mail which is display in the screen.
* For example user.mail



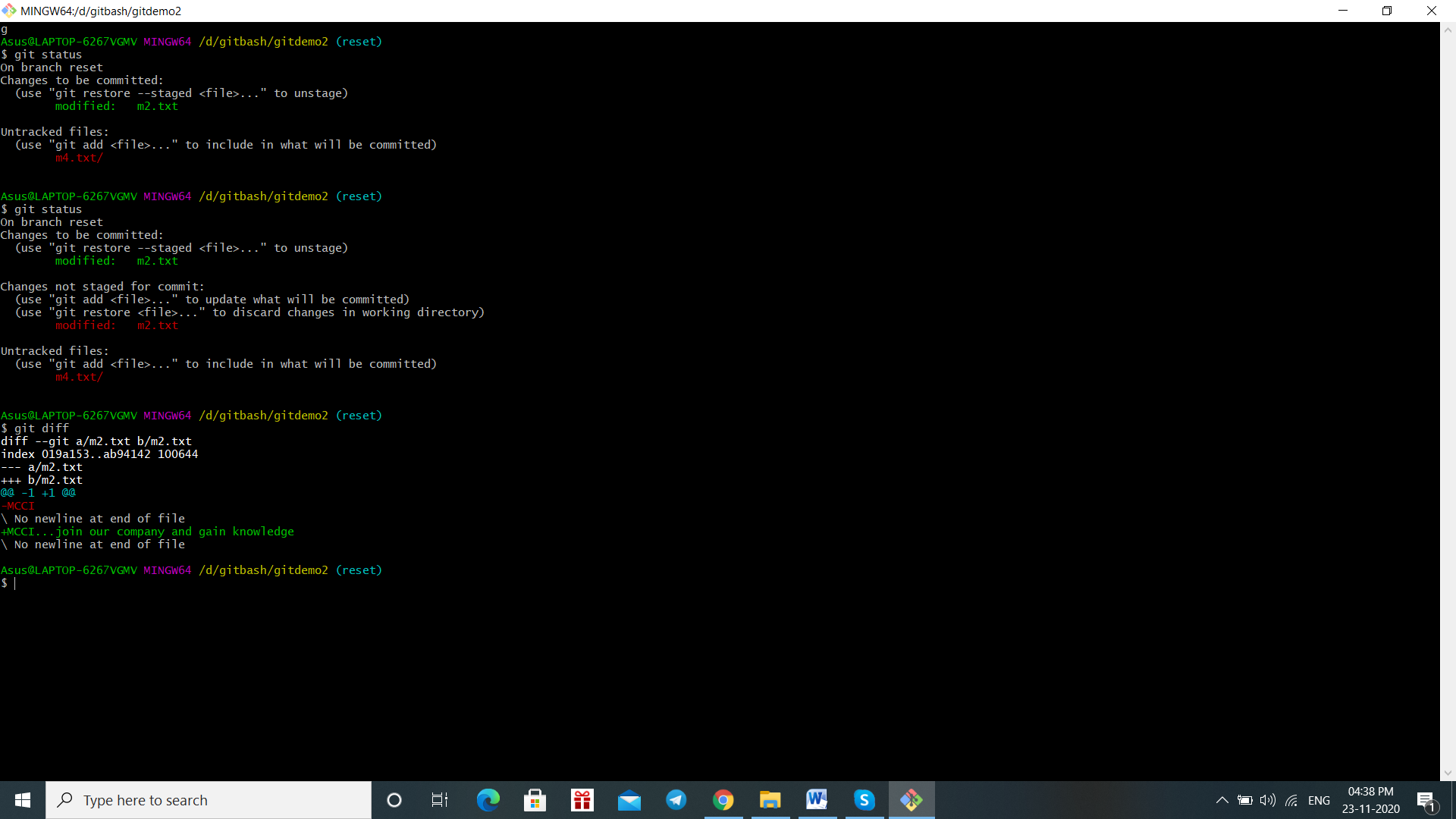
**GIT COMMIT:**

* It is a created by a git command to capture state of project at any time.
* We have to add a file and commit and this will appear



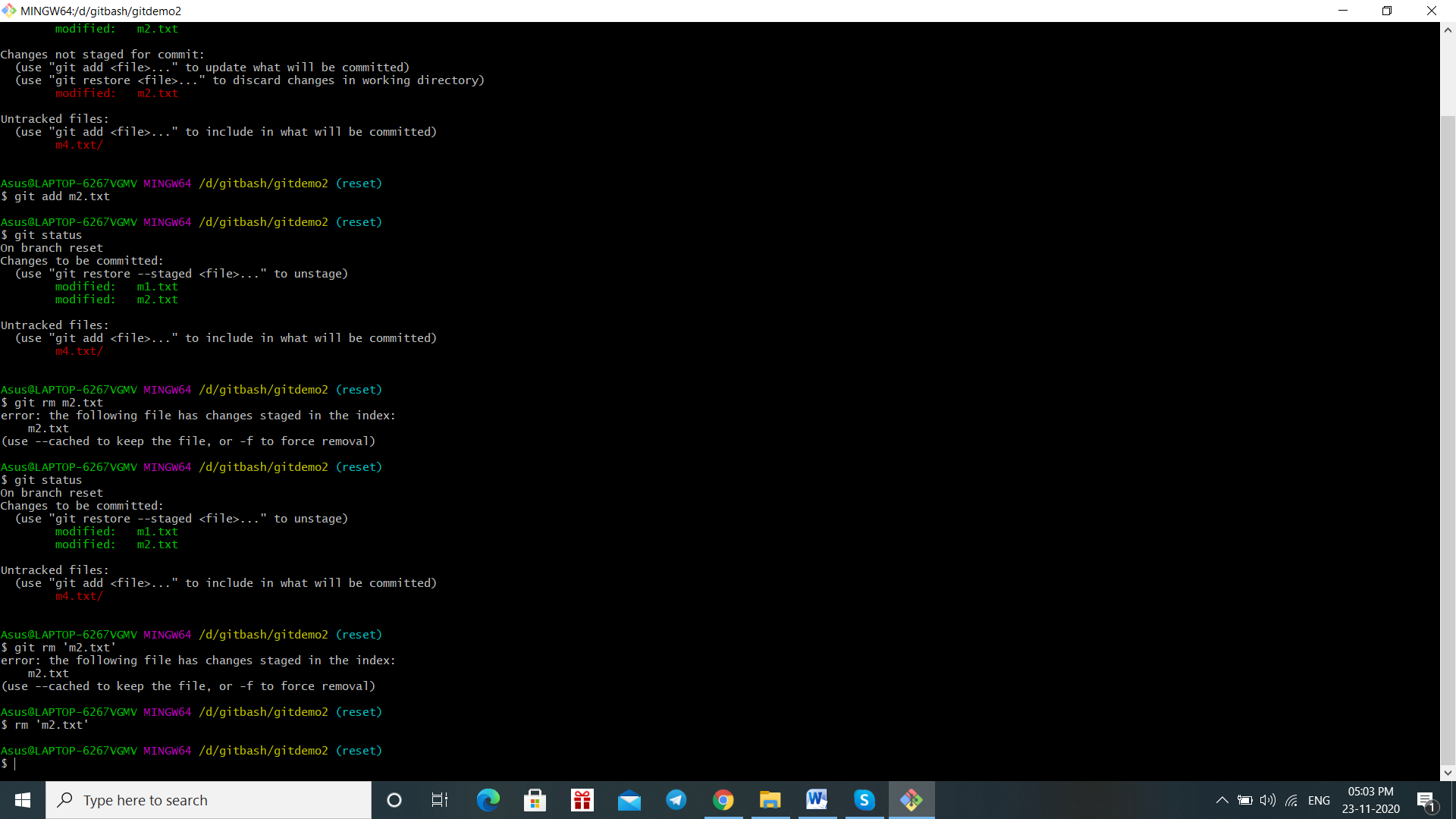
**GITDIFF:**

* **It is used to track the difference between the changes Made on file**



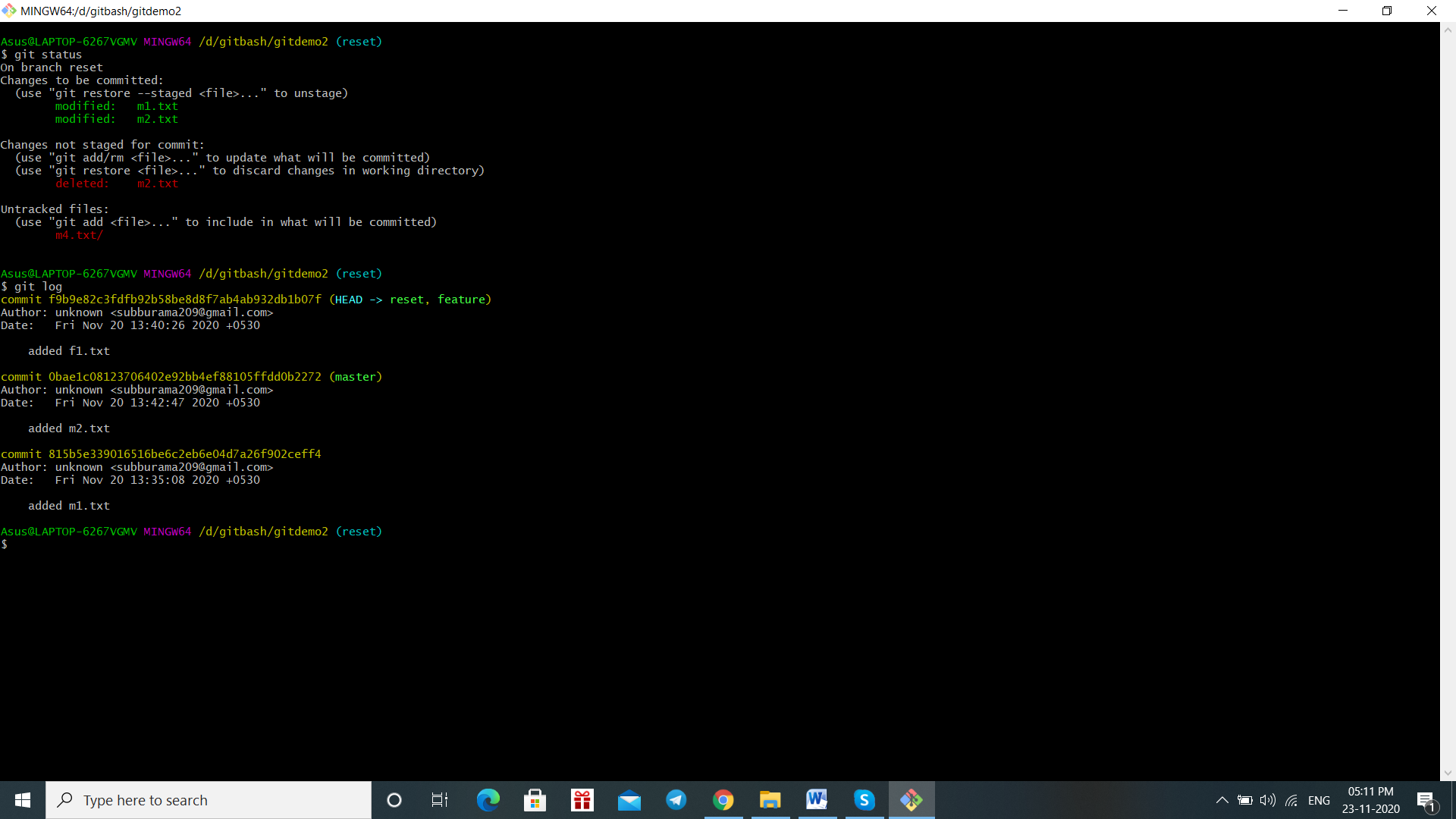
**GIT RM:**

* Git rm is used to remove files in your repository
* It is the inverse of git add



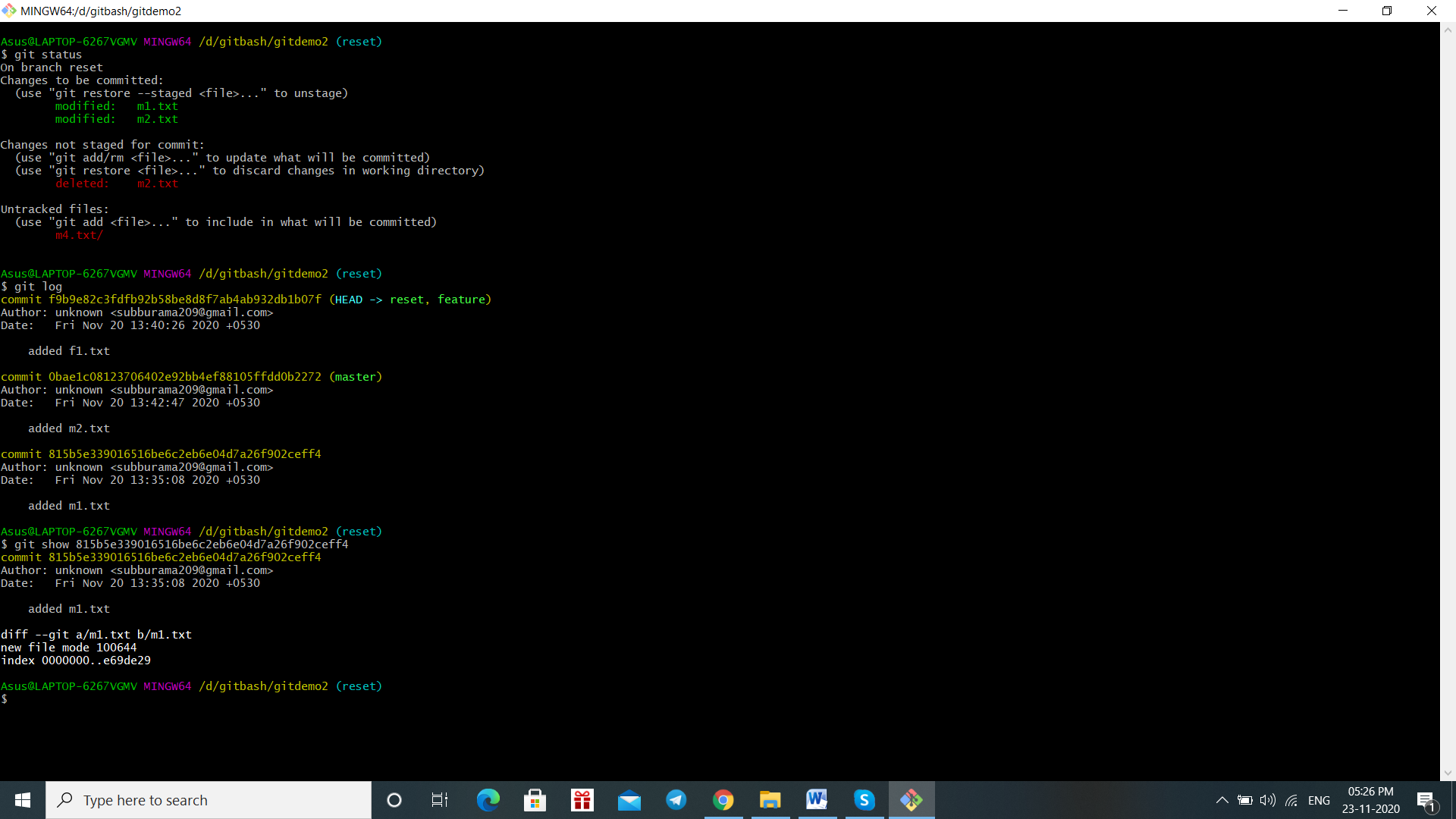
**GIT LOG:**

* It is nothing but it allows you to view information about previous commit that we are doing a project



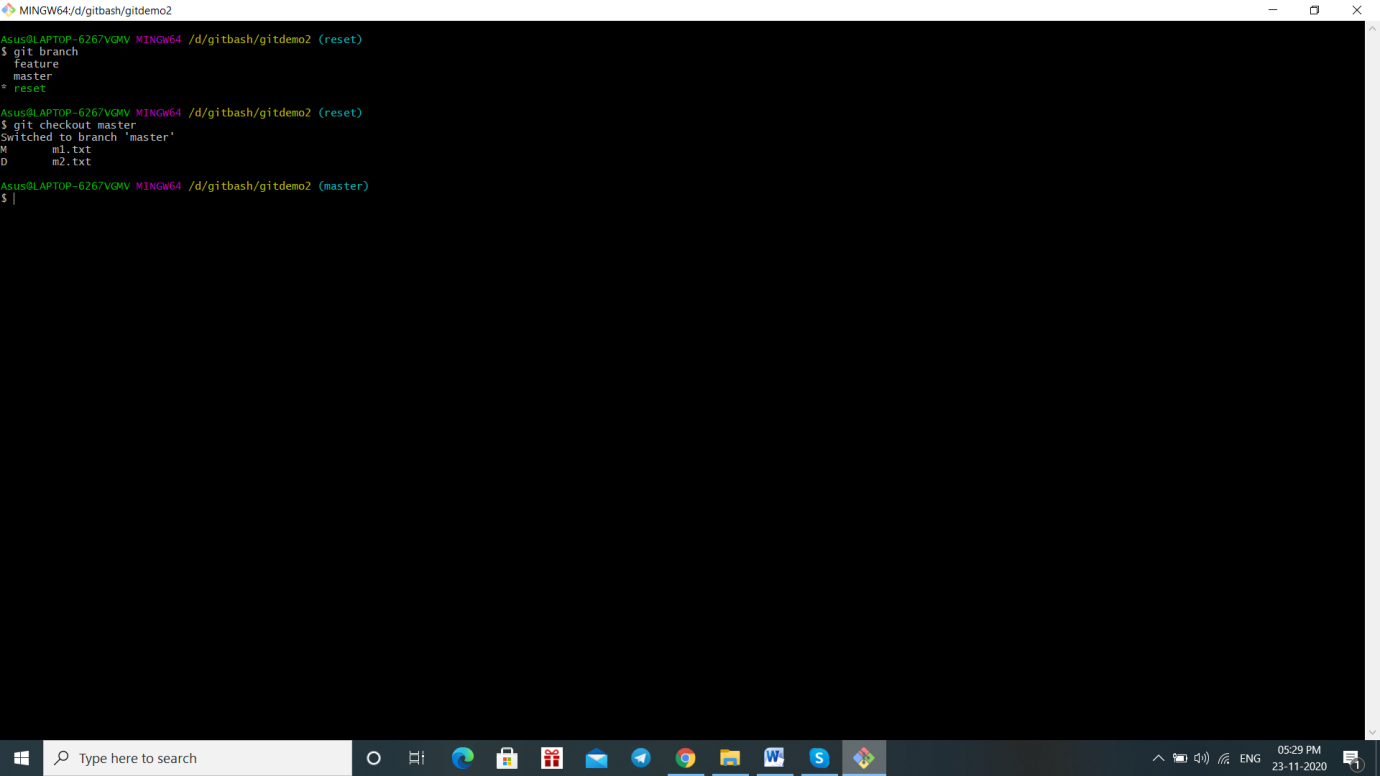
**GIT SHOW:**

It is used to view details on git objects like trees, tags and commits



**GIT CHECKOUT:**

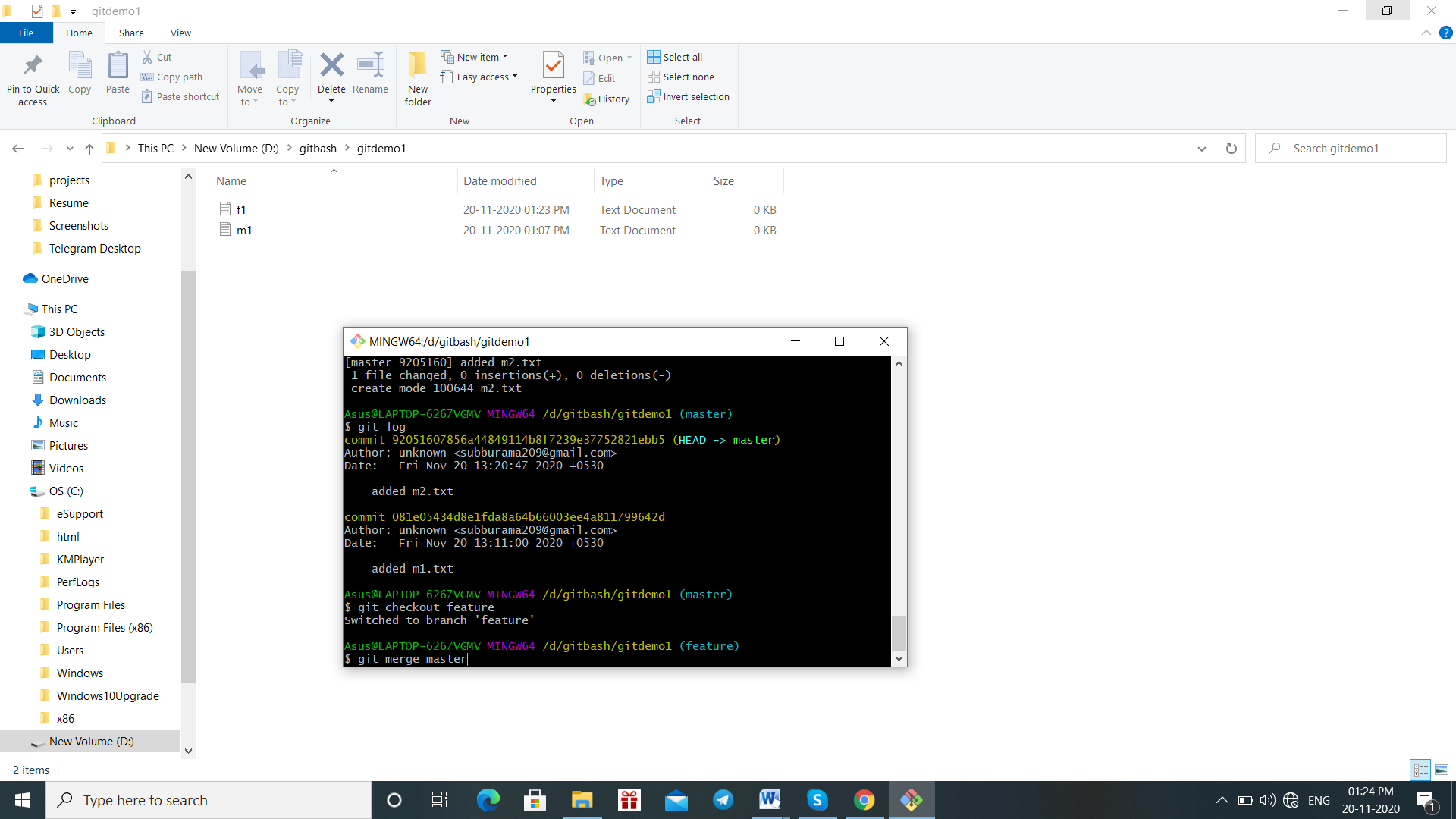
It is simple ,used to move between branches



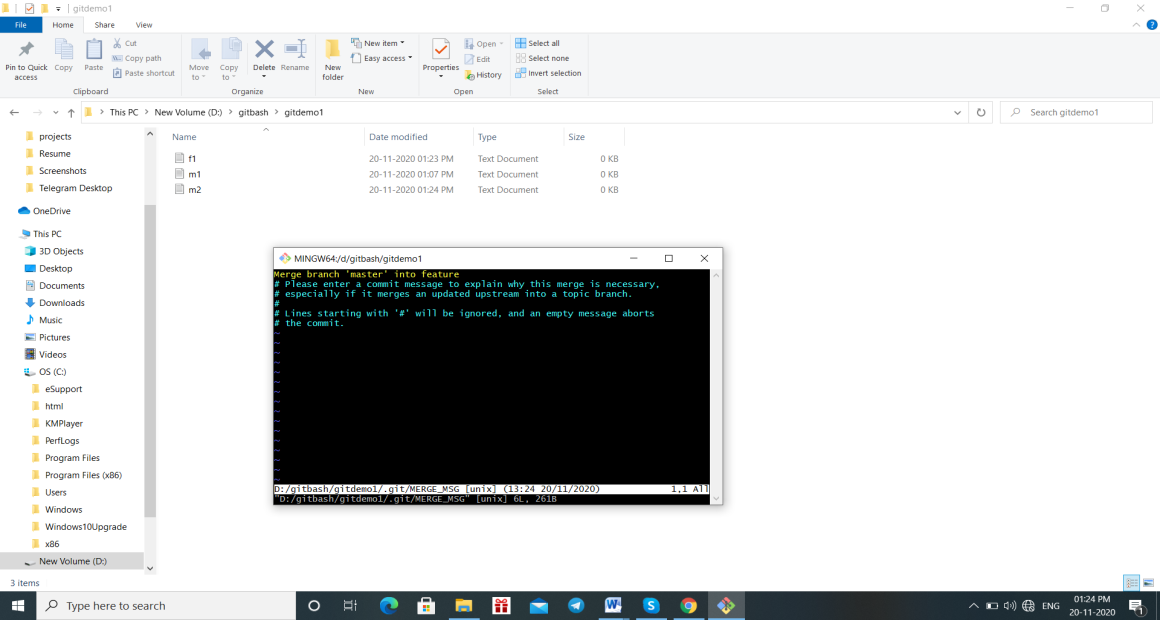
**GIT MERGE:**

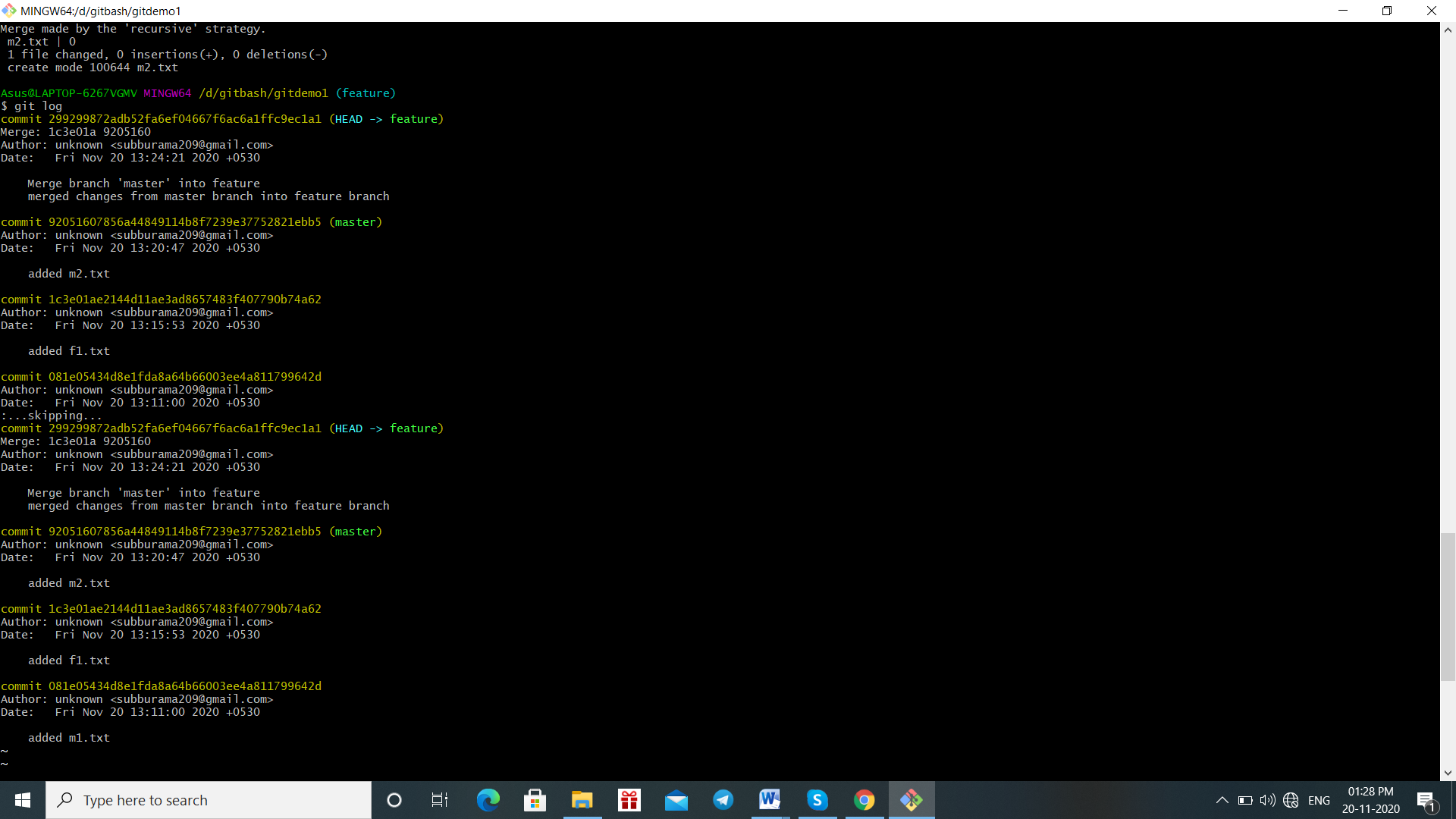
1.It is used to merge branches from git.

2. Merges takes all changes in one branch and merges them into another branch in one commit



* Create a text file m1 and commit it and create feature branch and checkout feature branch
* Switch to master so feature f1 text file gone …because master file holding only its source file not a feature file

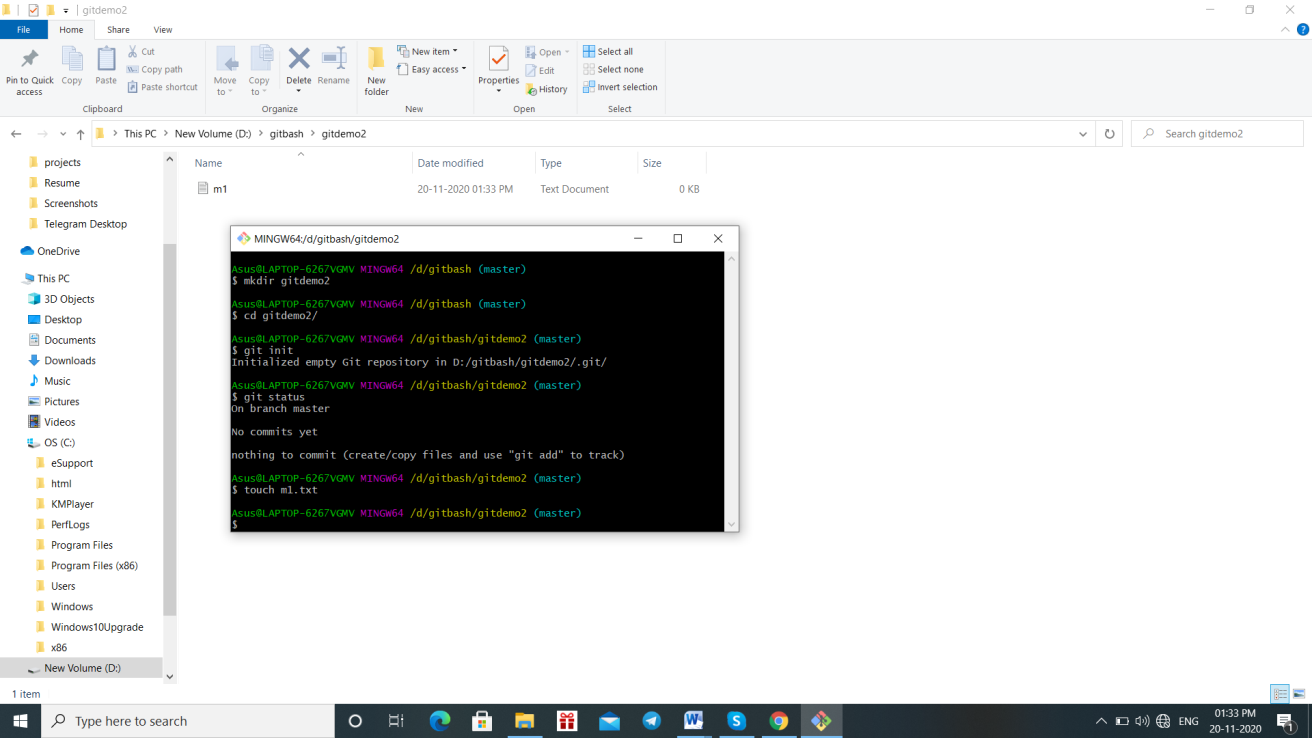


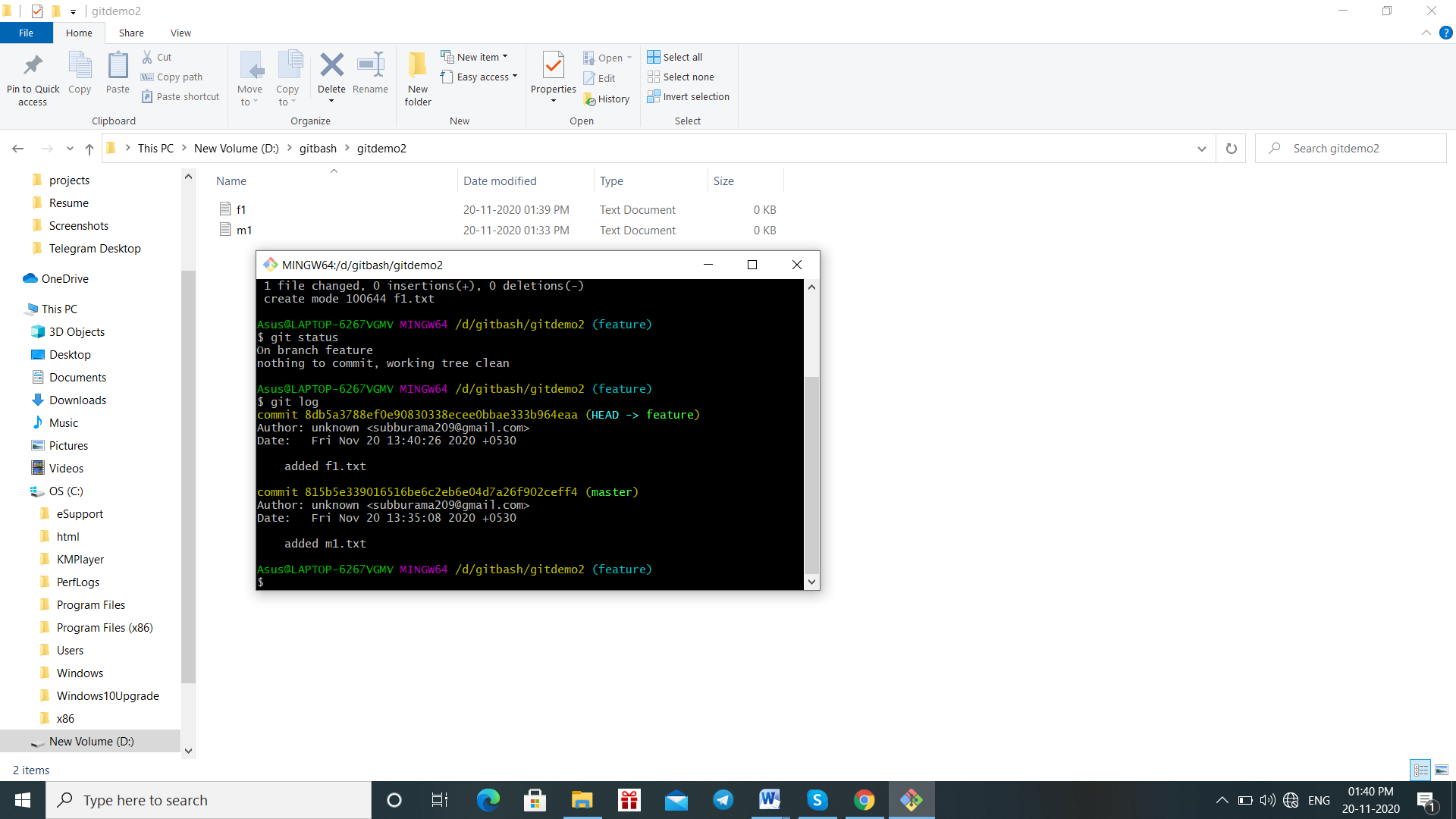


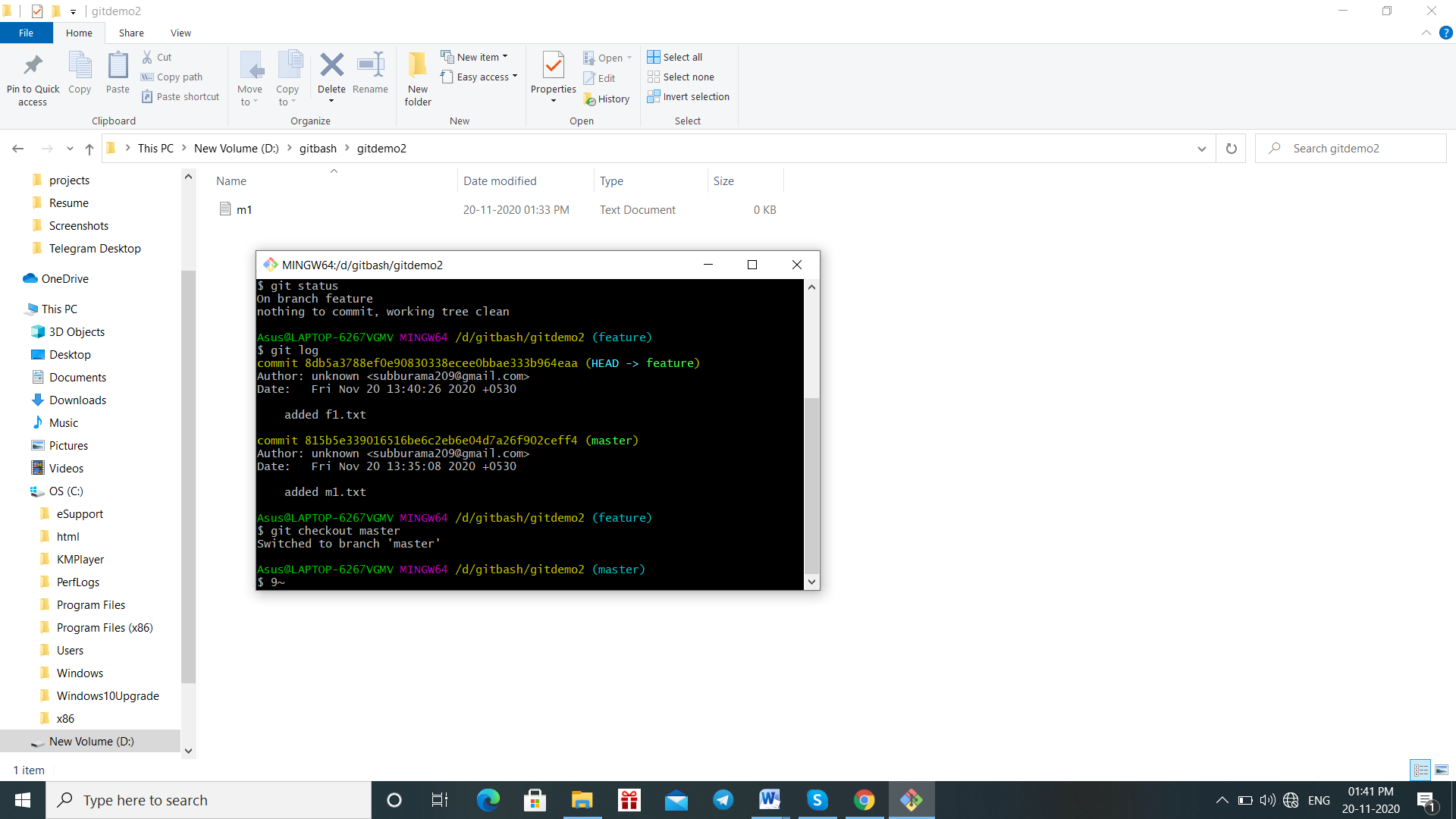
* Now the master text file merged into feature text file

**GIT REBASE:**

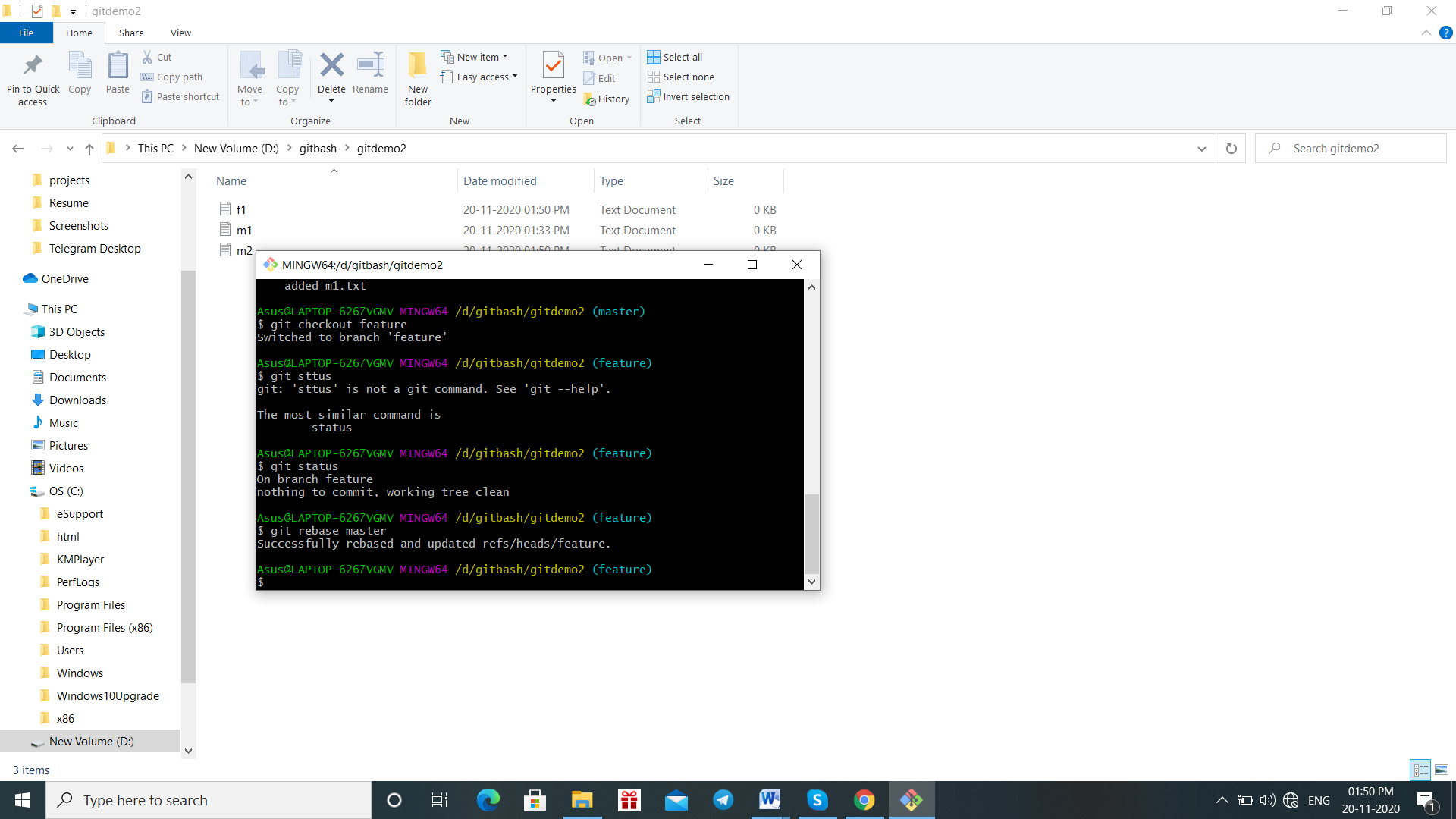
1.Git rebase moves a feature branch into a master

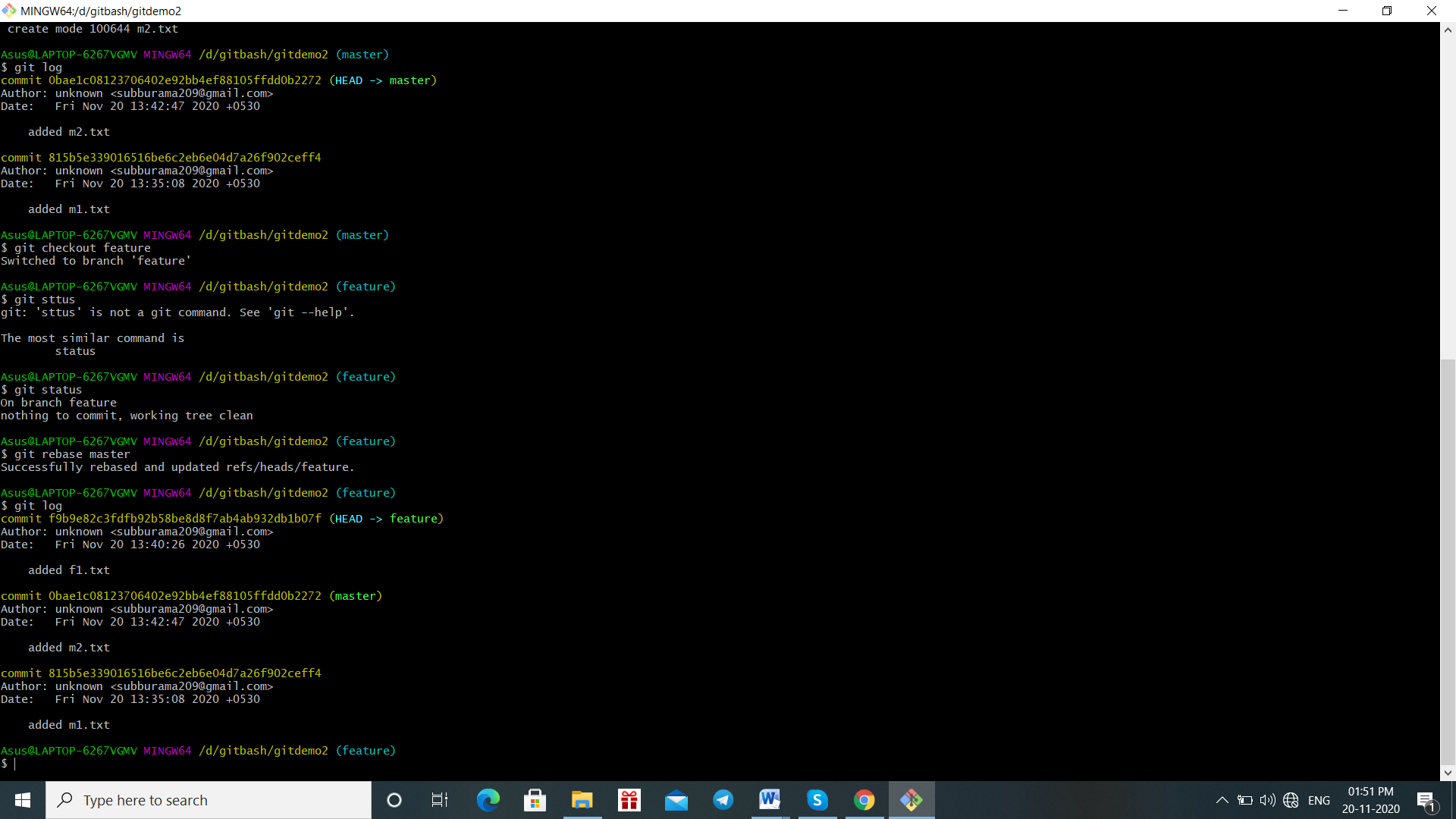






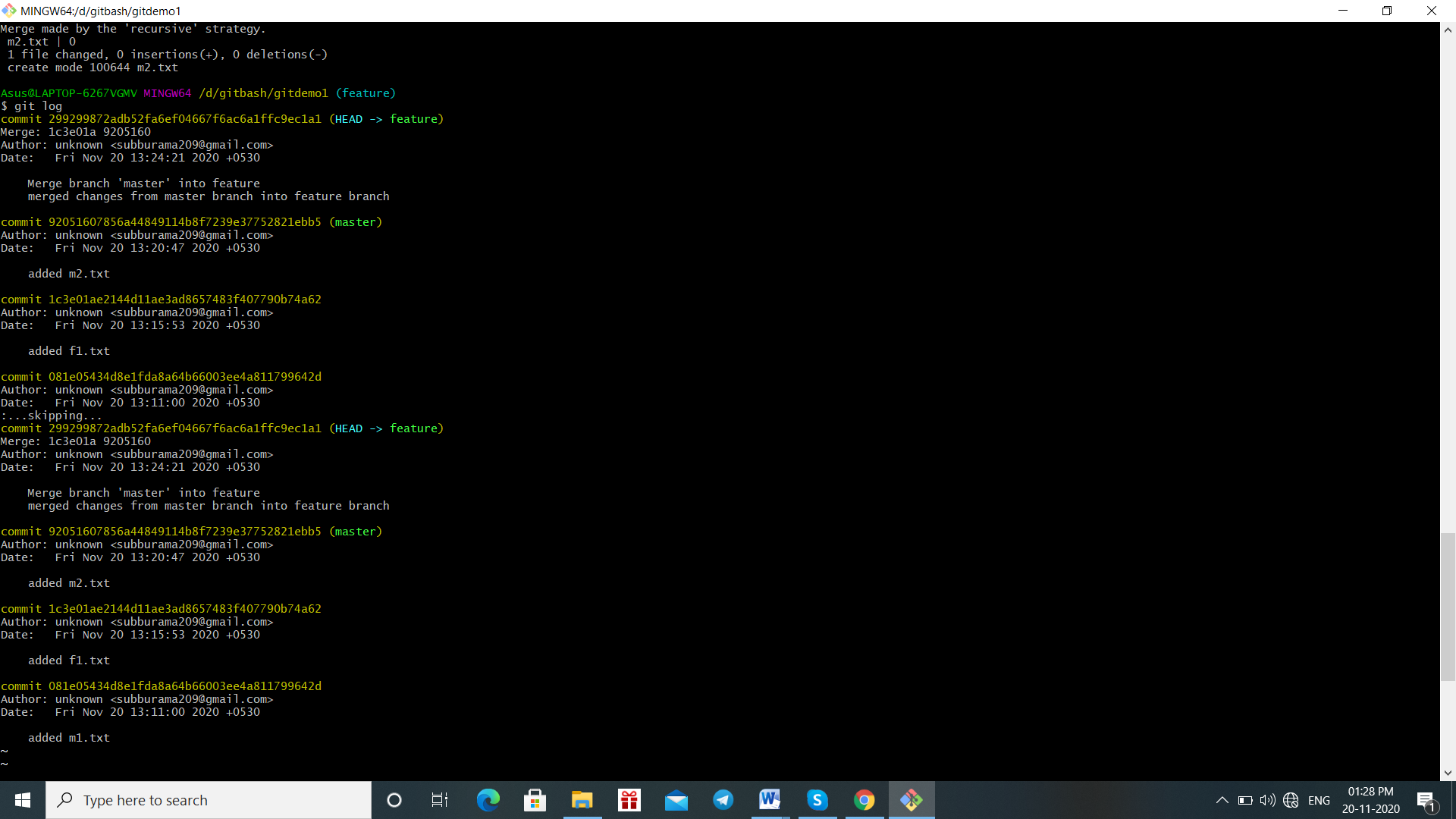




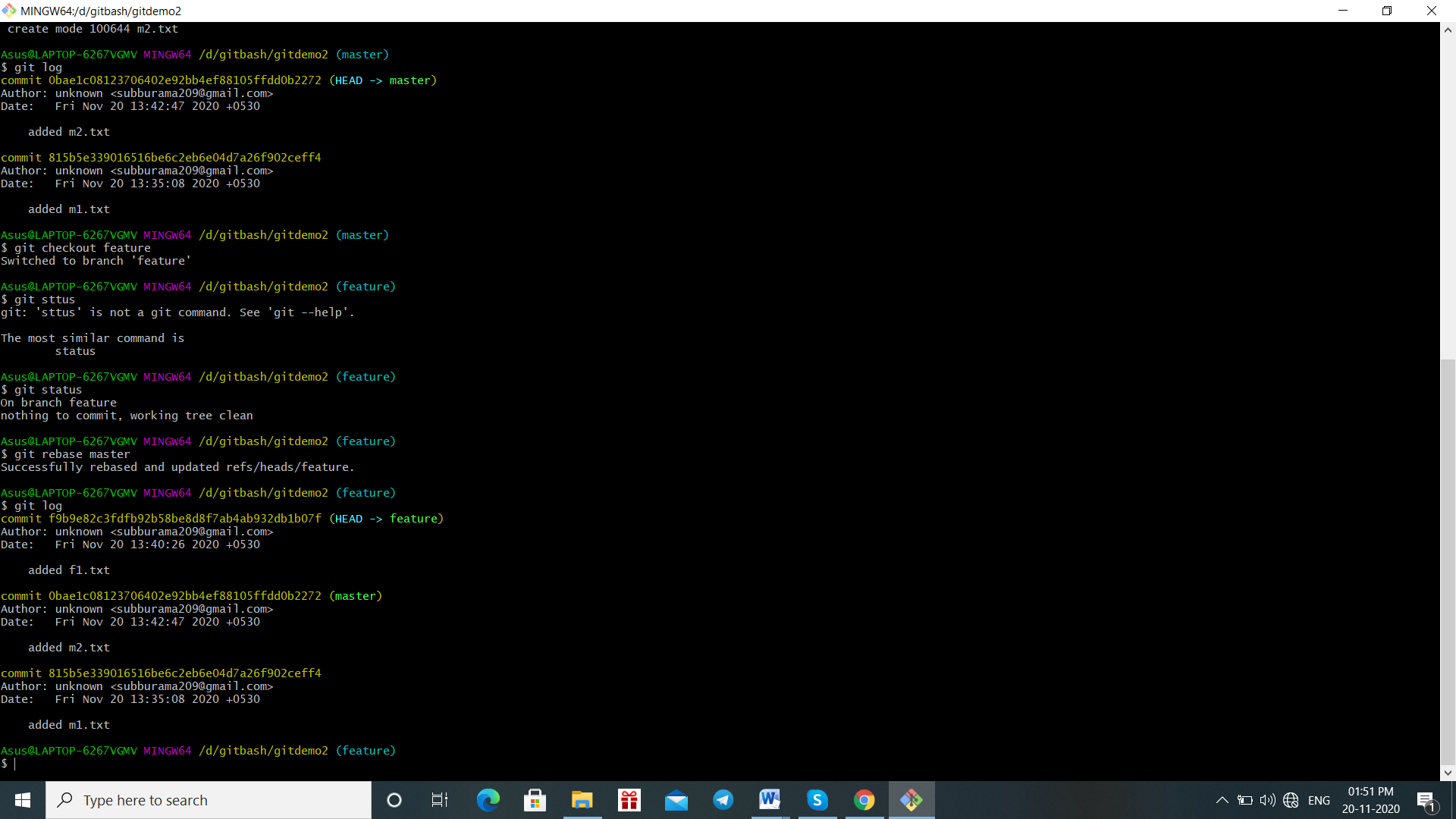


**DIFFERENCE BETWEEN REBASE AND MERGE:**

* **Both are similarly same, they give us the same result that applying changes from one branch into another**
* **But techniques are different**

****

* **Above one is merge**

****

**It is rebase technique**

**DIFFERENCES:**

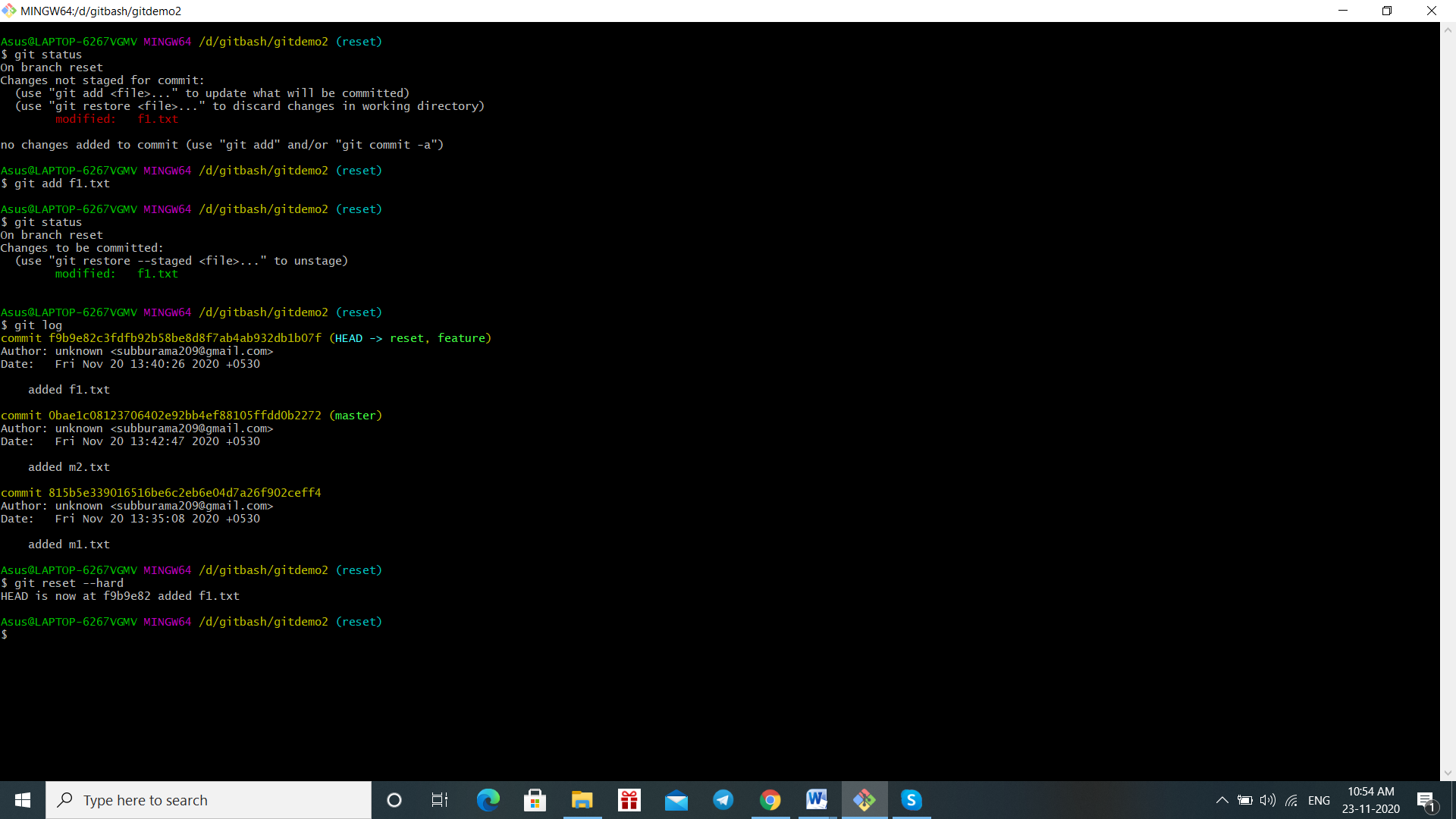
**IN MERGING:**

* **When go to git merge, In git log and in the history the commit we did on master and it also came to feature branch and we did some changes in feature branch and m2.txt coming after merge and also we have a separate merge commit.so it shows separate merge commit**
* **IN REBASE:**

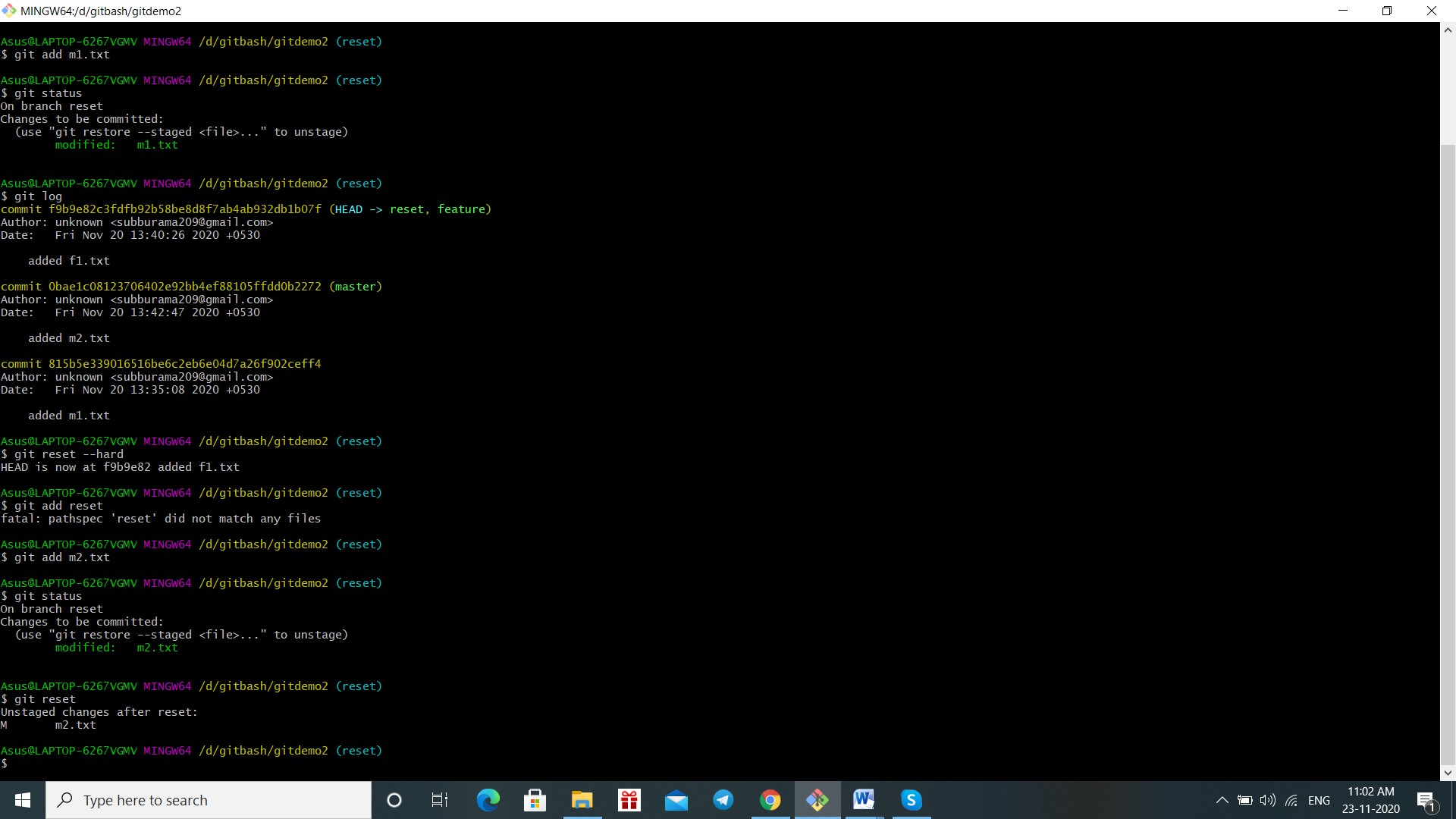
**If we see in git rebase it is very much linear it shows m1.txt first commit and f1.txt and it does not show a separate merge commit.**

**GIT RESET HARD:**

* The --hard option changes the Commit History
* Any previously pending commits to the Staging Index and the Working Directory gets reset to match Commit Tree.
* It means any expected work will be lost

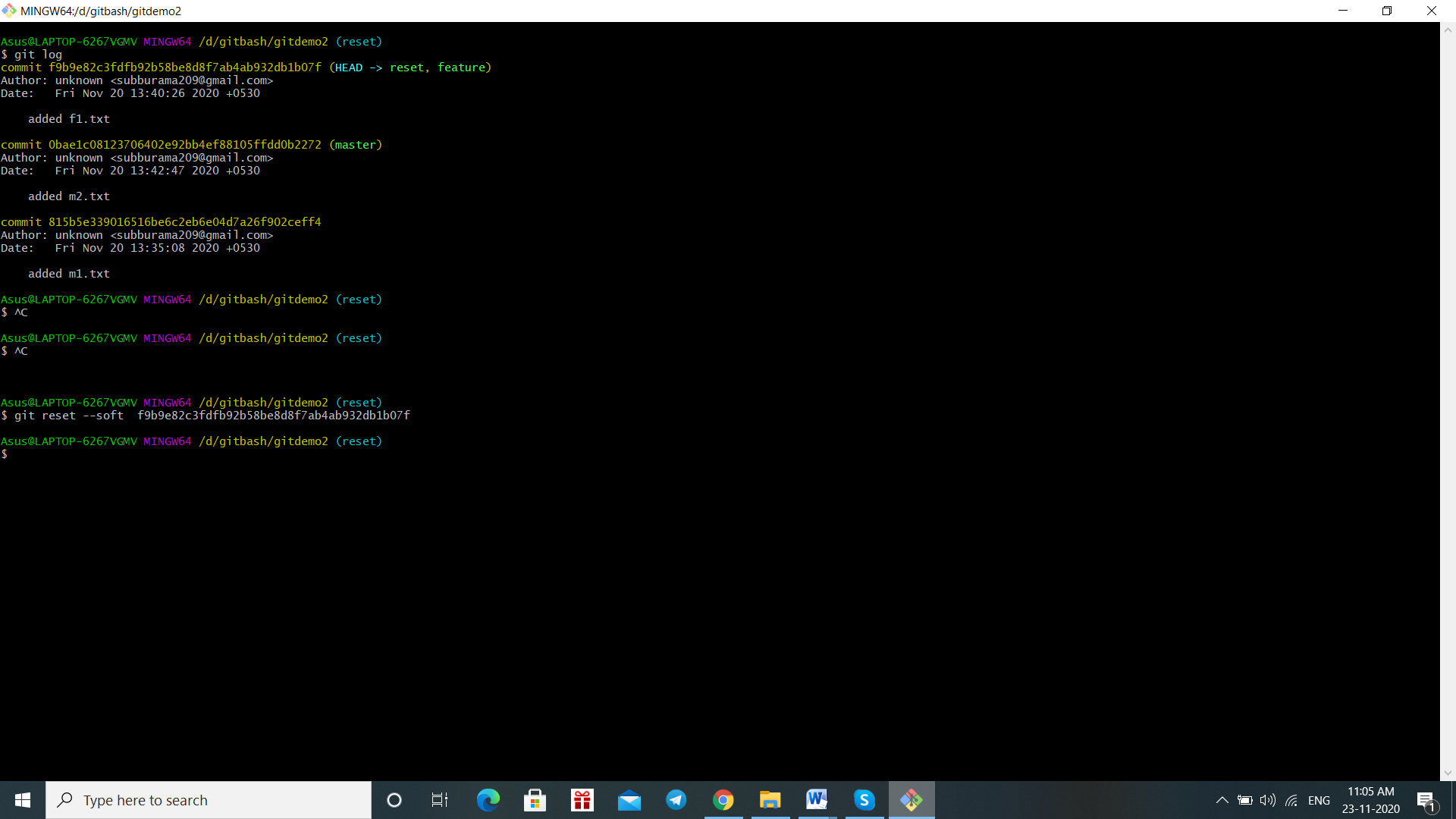


**GIT RESET MIXED:**



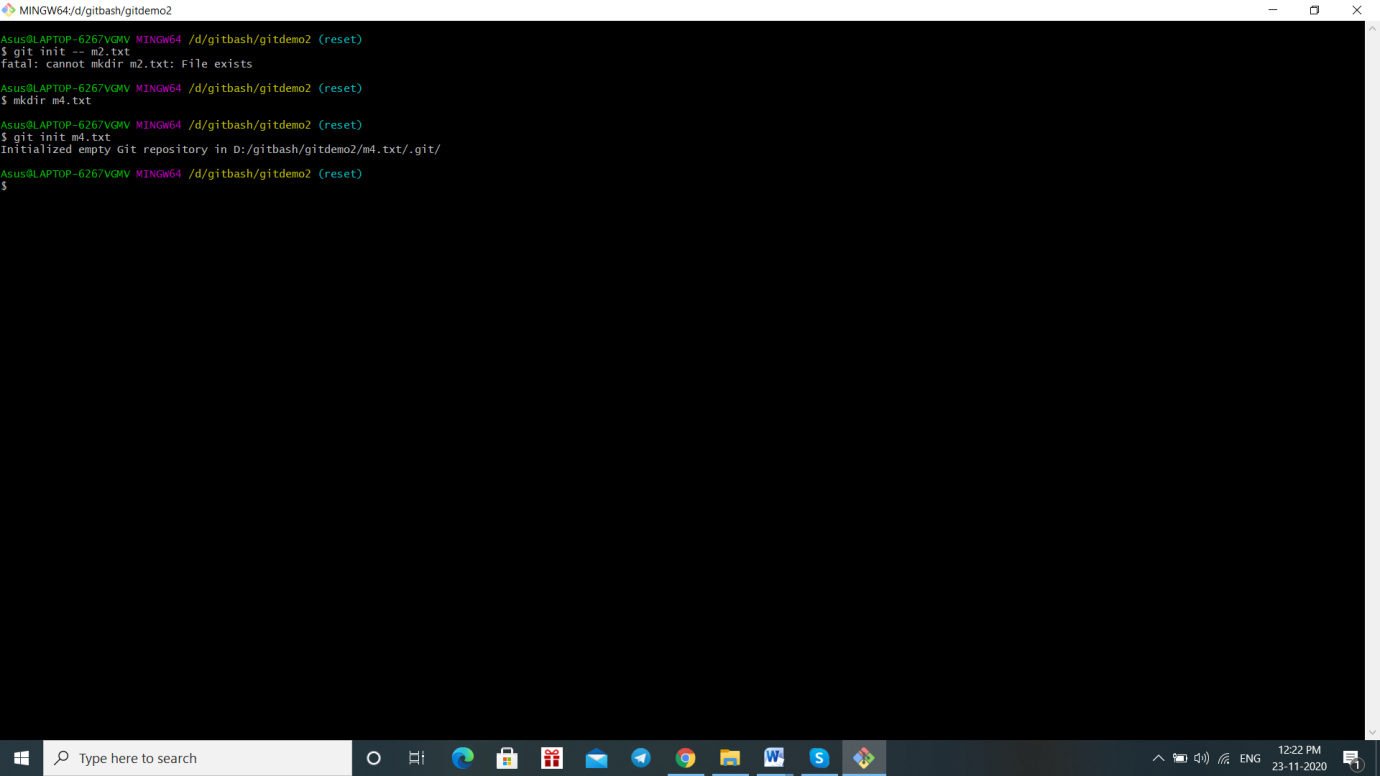
**GIT RESET SOFT:**

It does not touch the index file and reset the head options.so the changes occur in head.



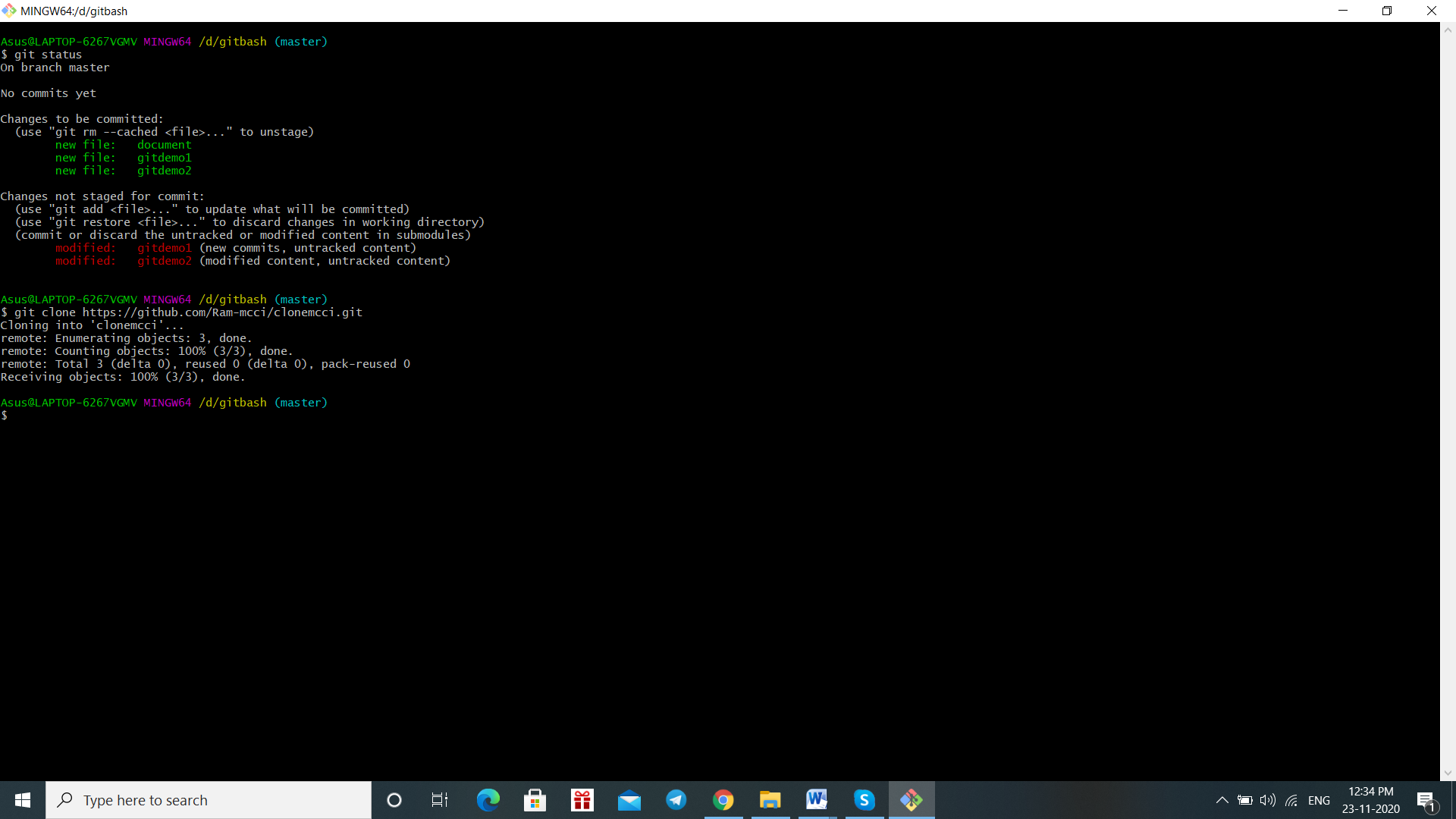
**GIT INIT:**

* Git init is a command creates a new git repository and also other command will not obtain outside the init
* It is also used to convert the existing project to git repository



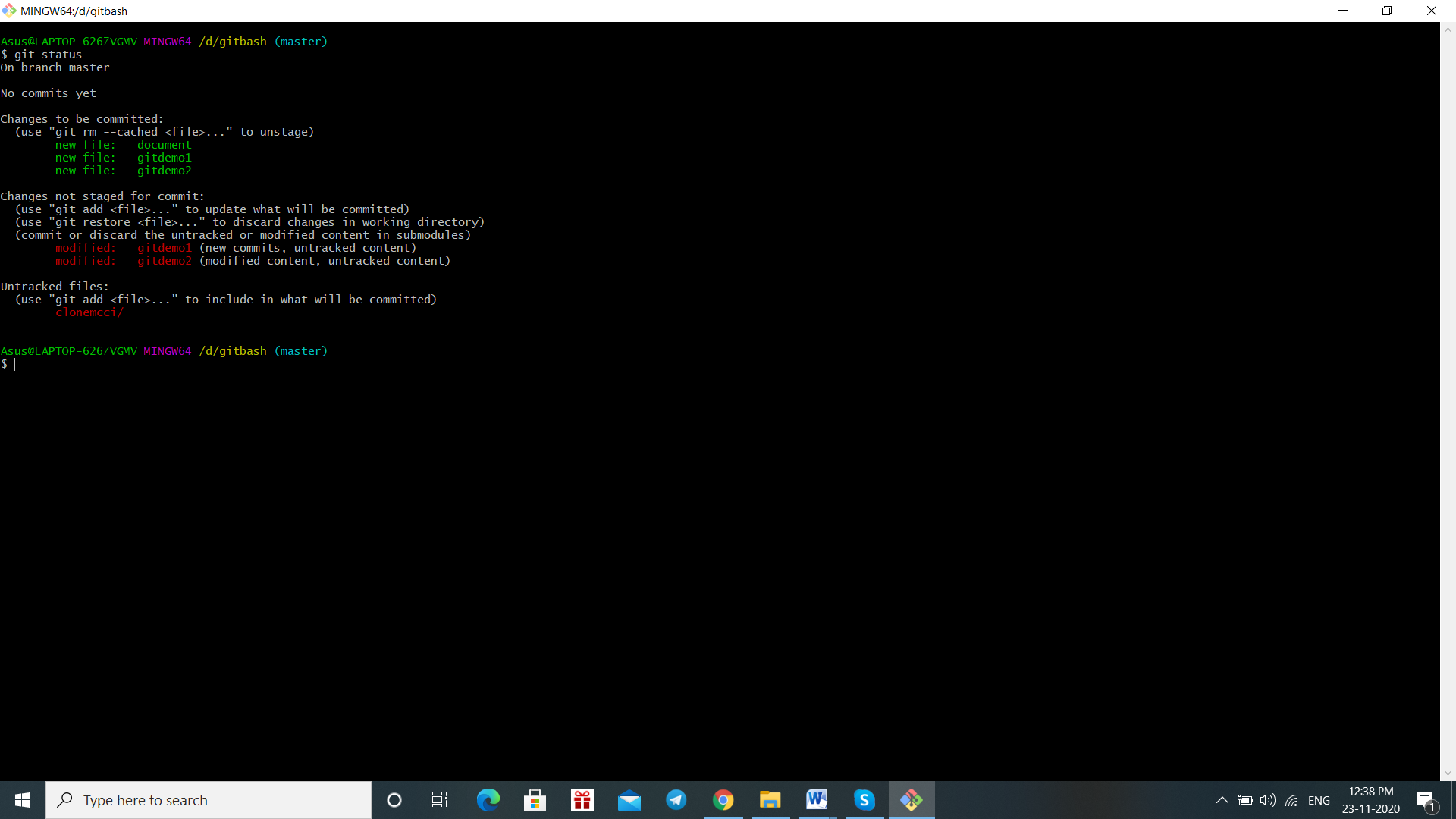
**GIT CLONE:**

* It is used to initiate a new repository and it is dependent on git init
* Generally git clone is used to create a copy of the existing repository



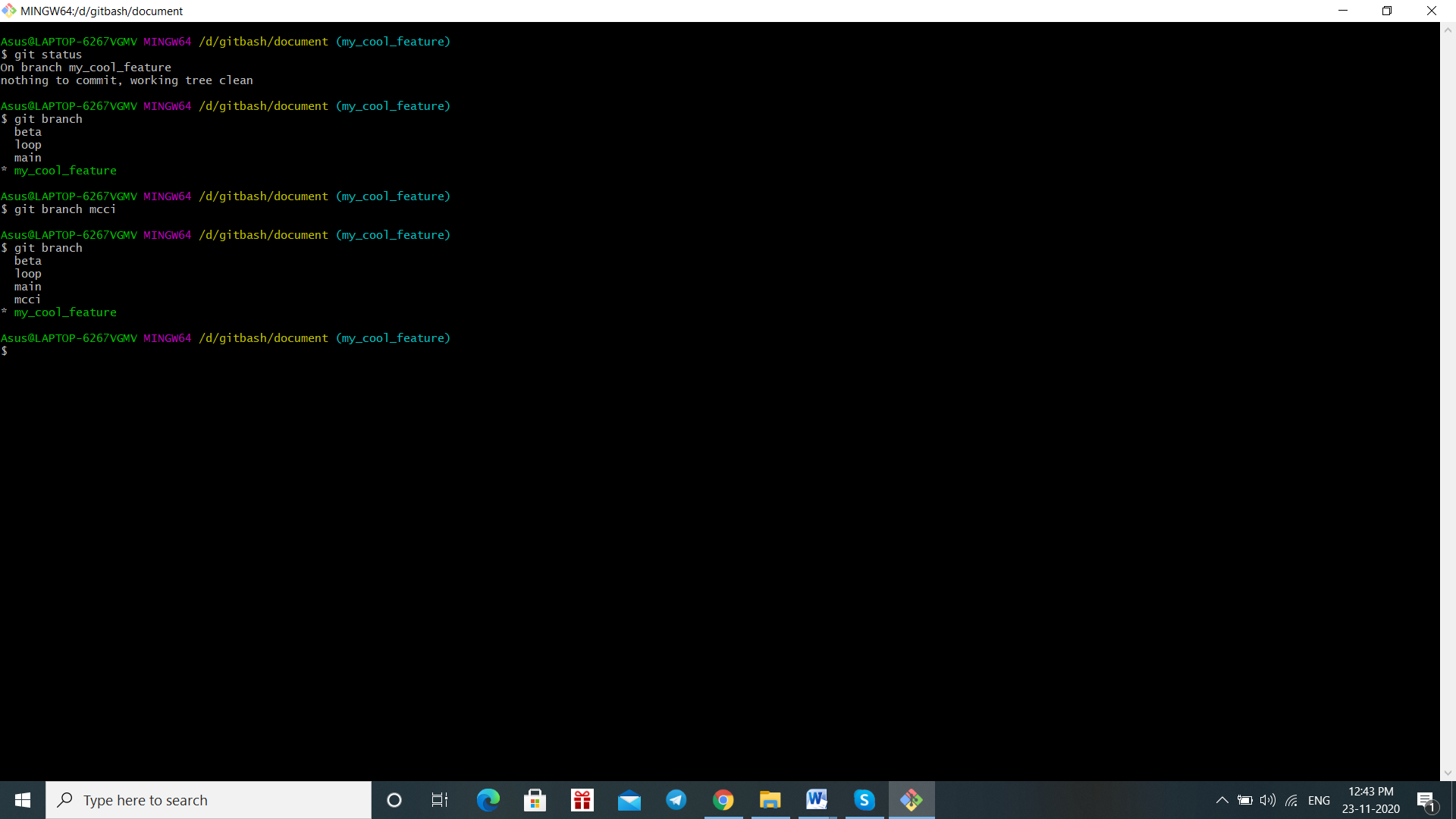
**GIT STATUS:**

* It is nothing but it shows the state of working directory and staging area.



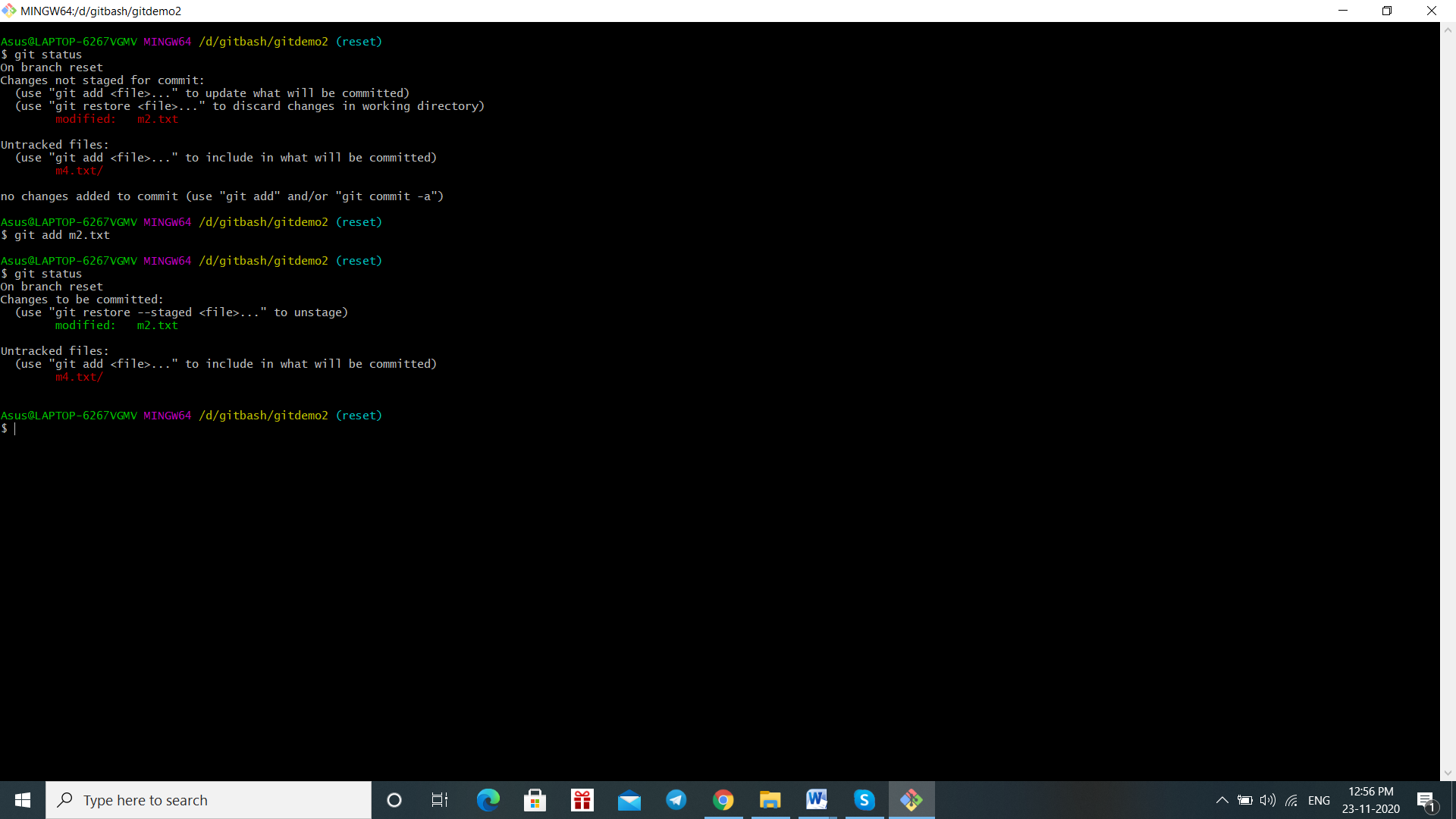
**GIT BRANCH:**

* Branches are used to test the changes the code without affecting the main code



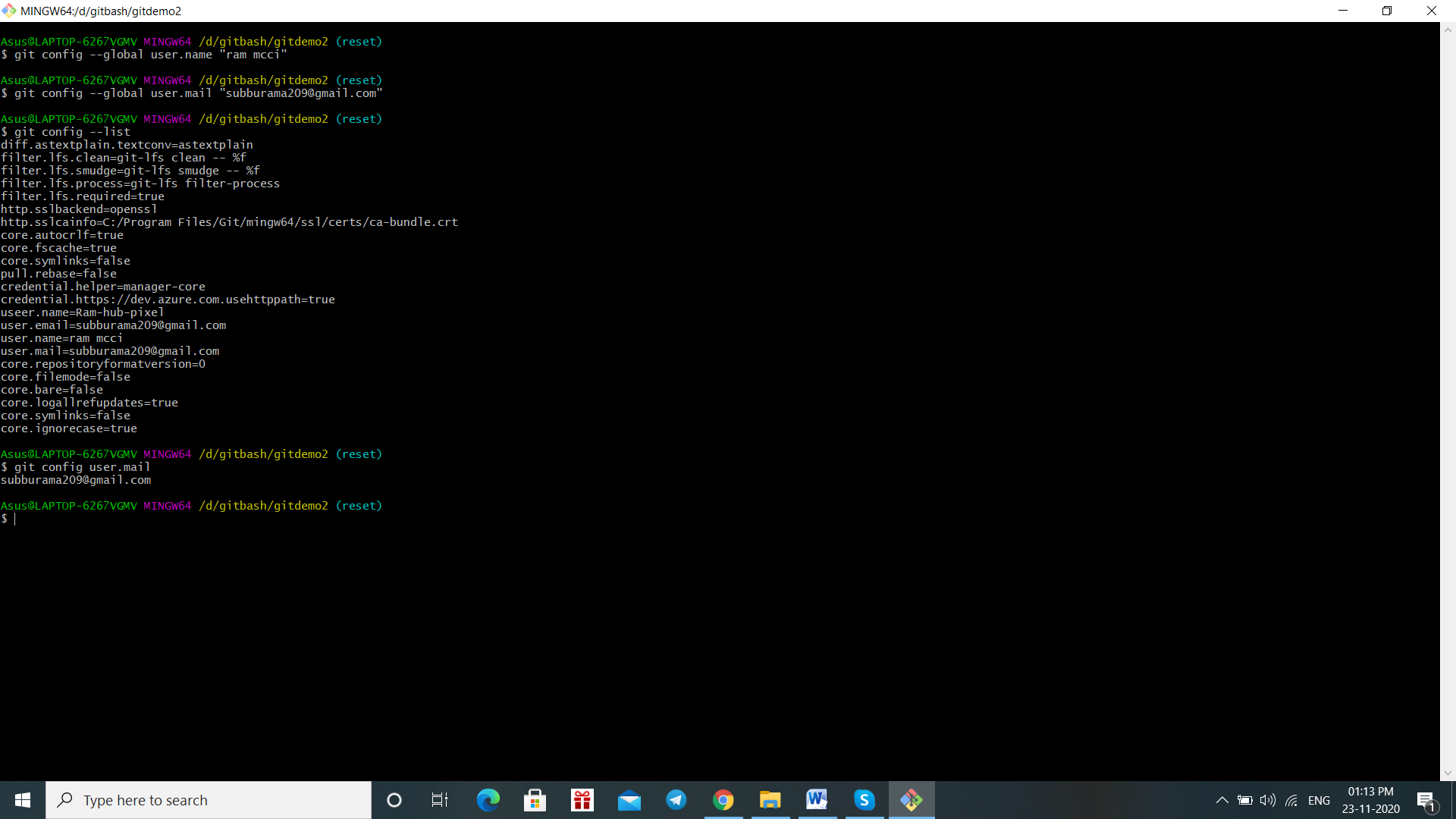
**GIT ADD:**

* It is used to add change in working to the statging area
* It tells git we want to add update in next commit and also it does not affect repository



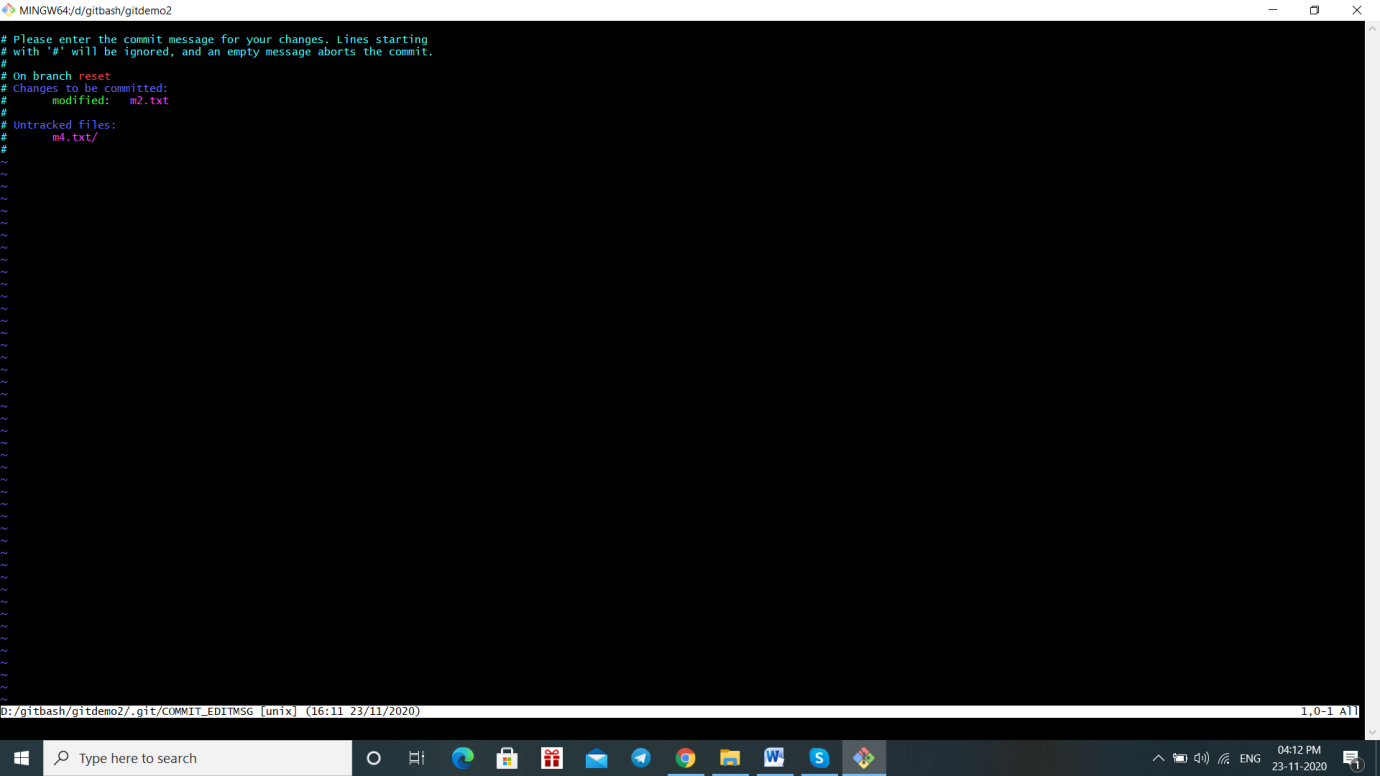
**GIT CONFIG:**

* It is used to set git configuration values on a local or global projectlevel.
* It is used to call username and mail which is display in the screen.
* For example user.mail



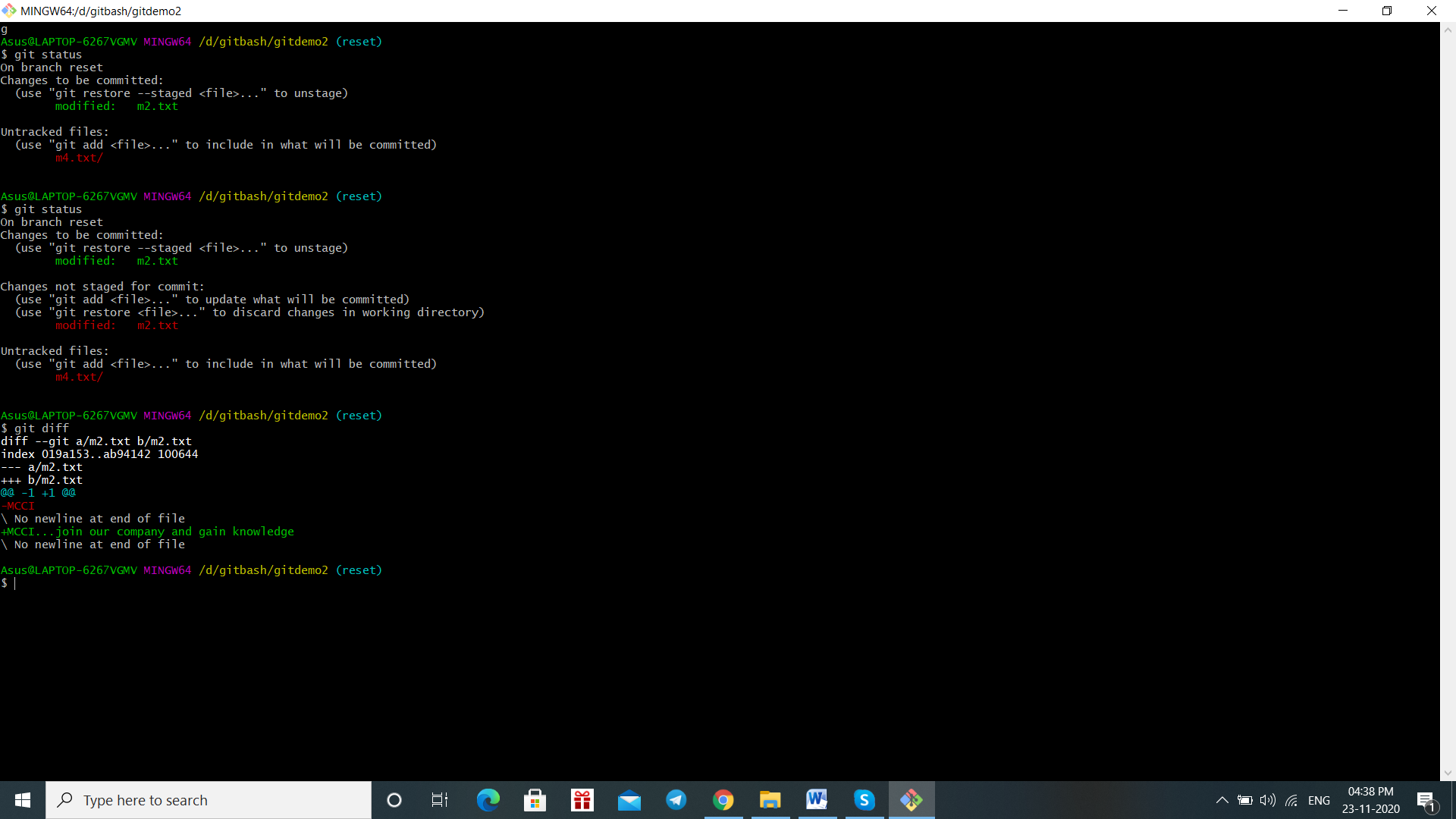
**GIT COMMIT:**

* It is a created by a git command to capture state of project at any time.
* We have to add a file and commit and this will appear



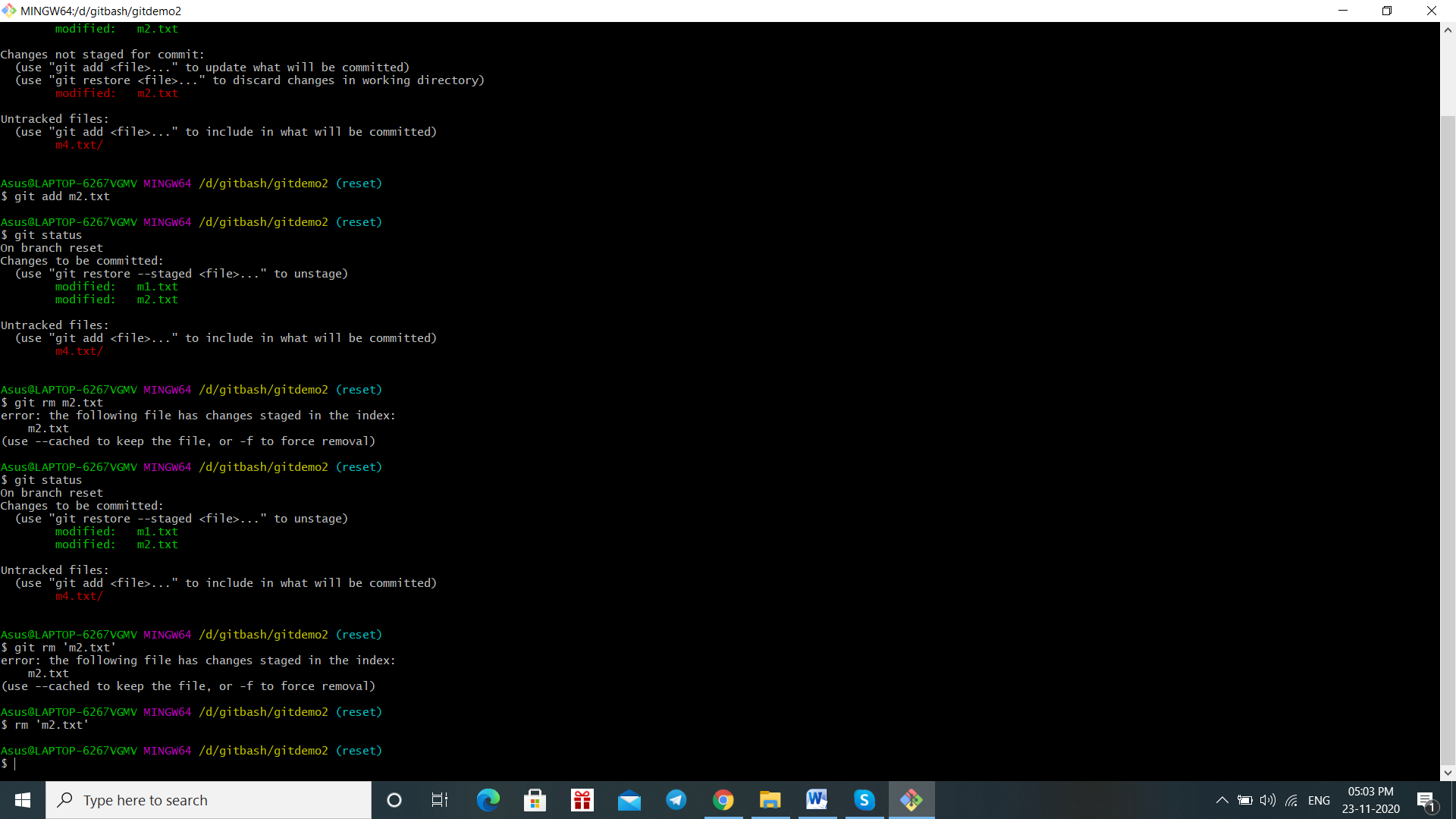
**GITDIFF:**

* **It is used to track the difference between the changes Made on file**



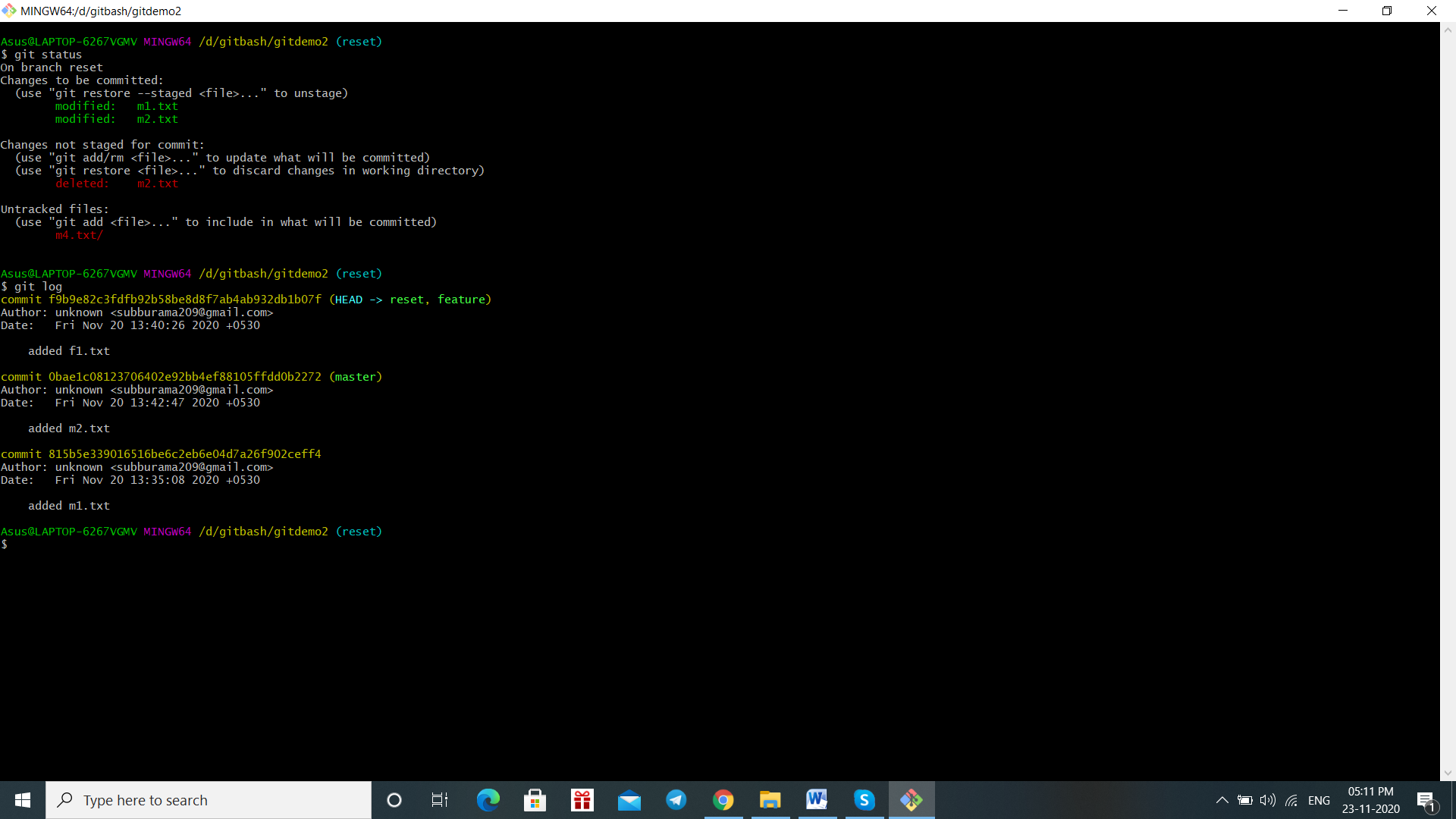
**GIT RM:**

* Git rm is used to remove files in your repository
* It is the inverse of git add



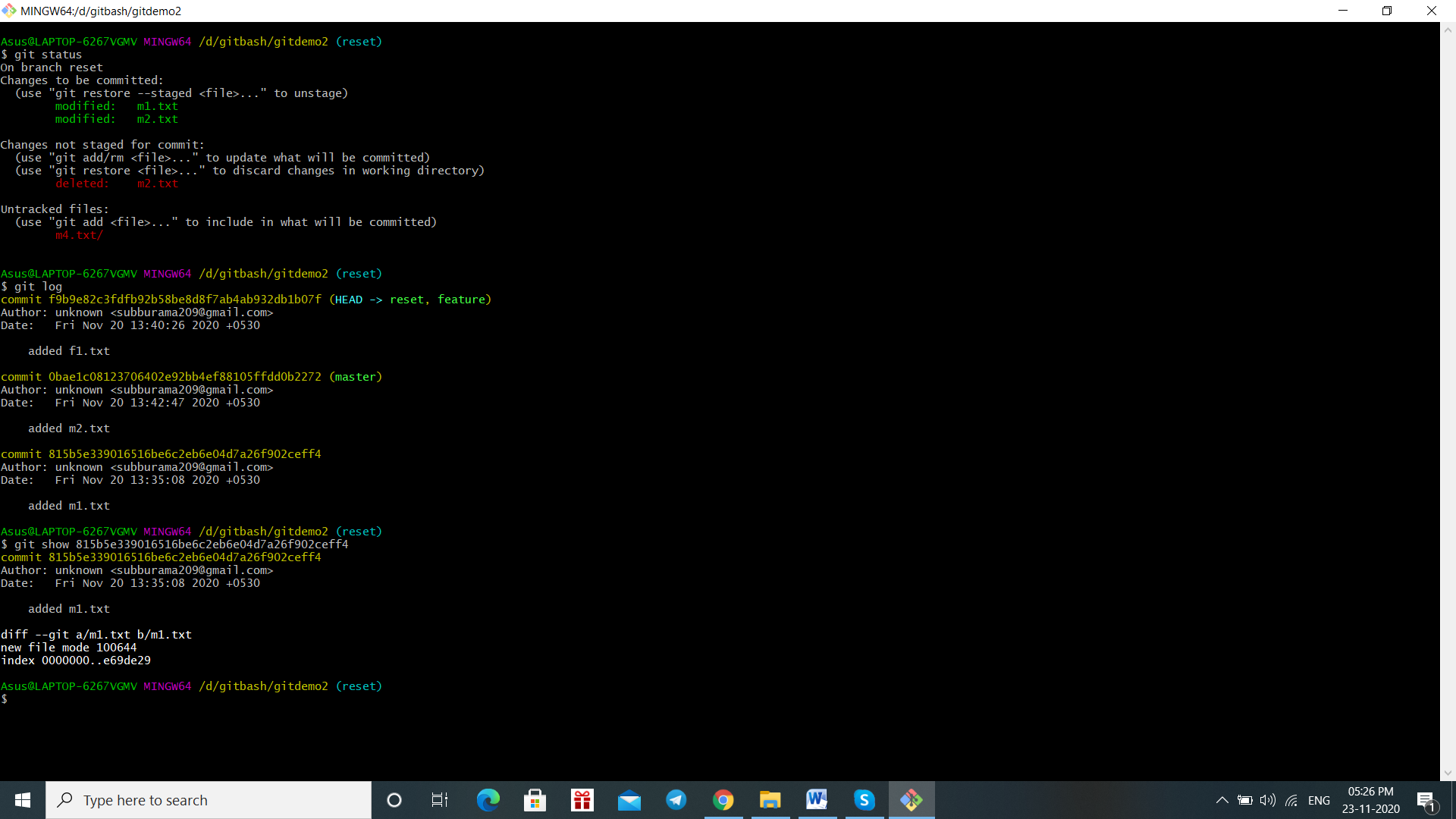
**GIT LOG:**

* It is nothing but it allows you to view information about previous commit that we are doing a project



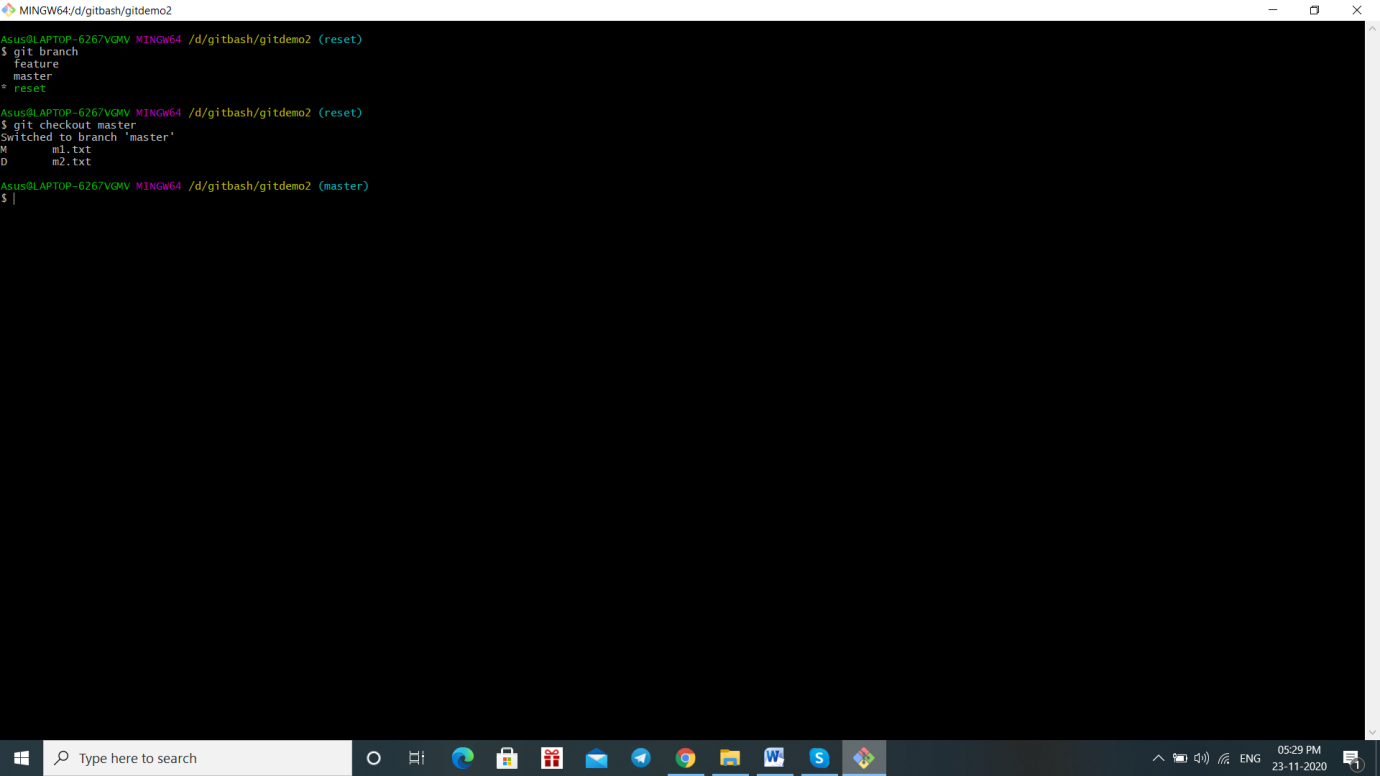
**GIT SHOW:**

It is used to view details on git objects like trees, tags and commits



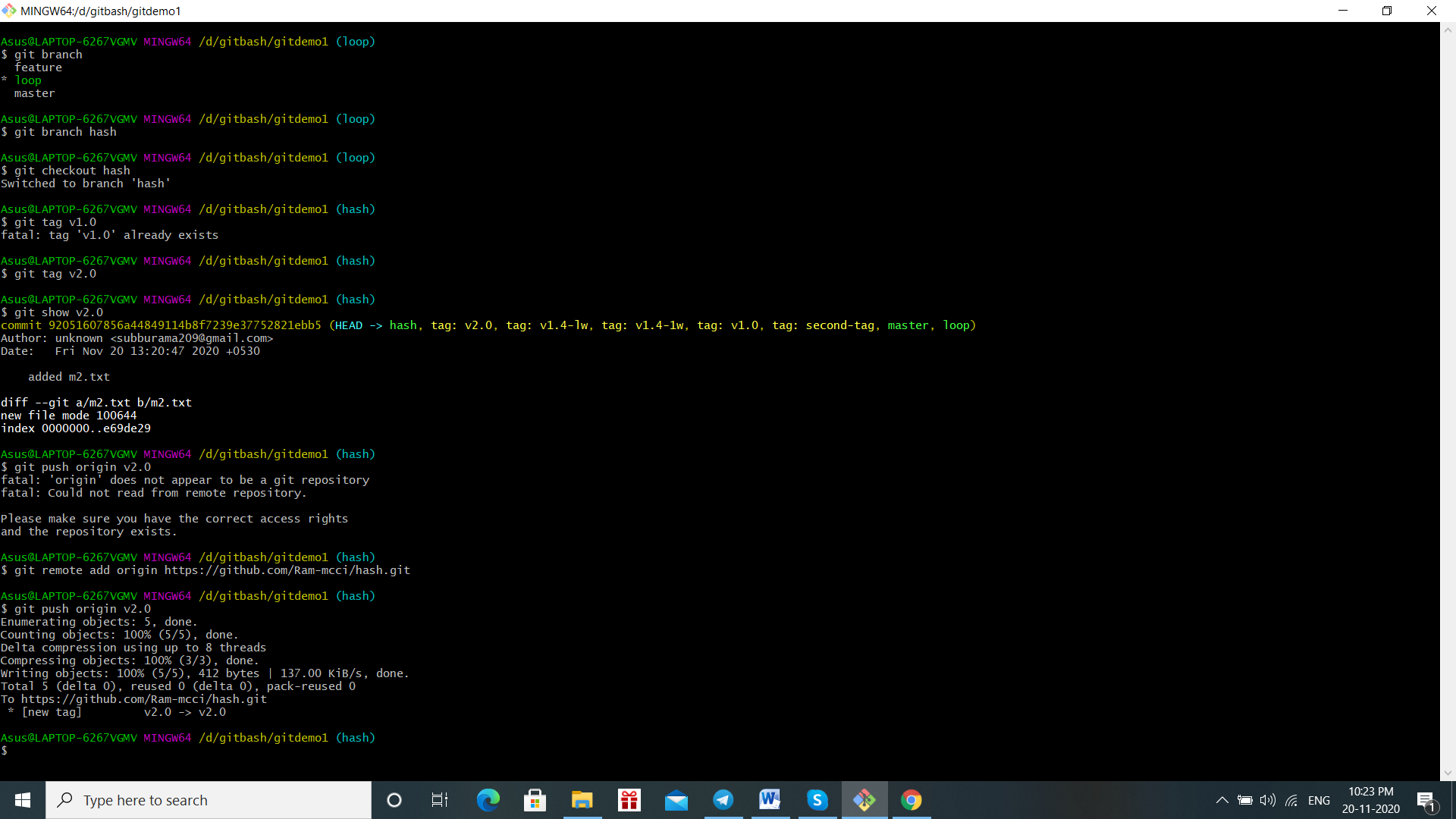
**GIT CHECKOUT:**

It is simple ,used to move between branches



**GIT TAG:**

A tag is like a branch and doesn’t change .after it is created it has no history of commits. It is used to capture a point in history that is used for marked version release.



GIT REMOTE

It is a common repository that al members used to exchange their changes

