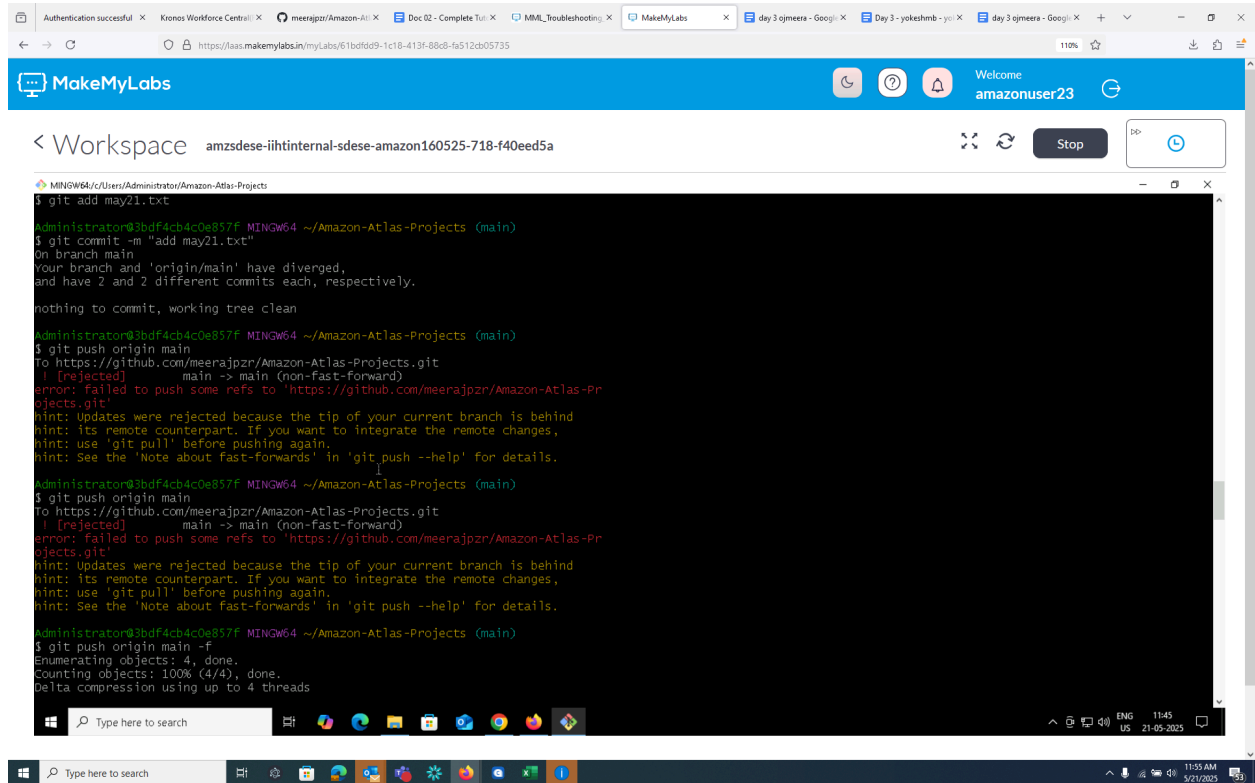


Task 1:

Recap of Last session:

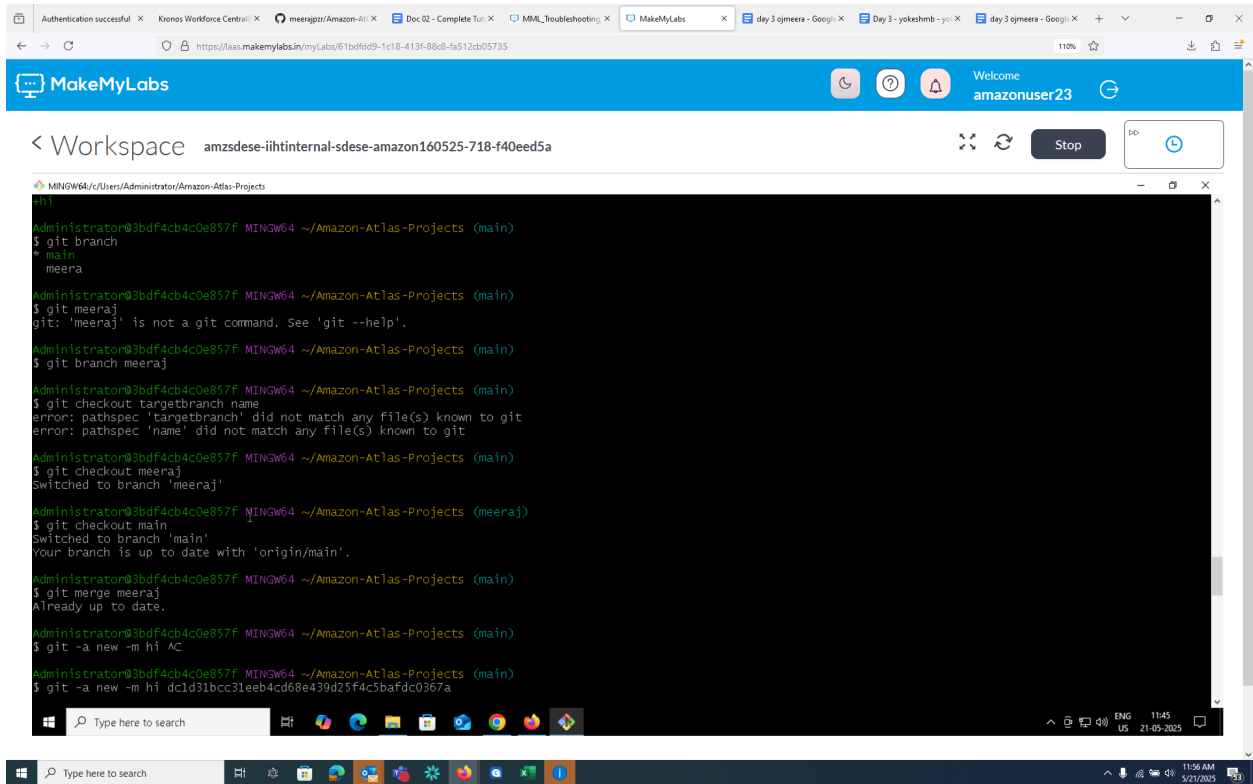
Create a file names 21st May.txt and push it to your git hub.



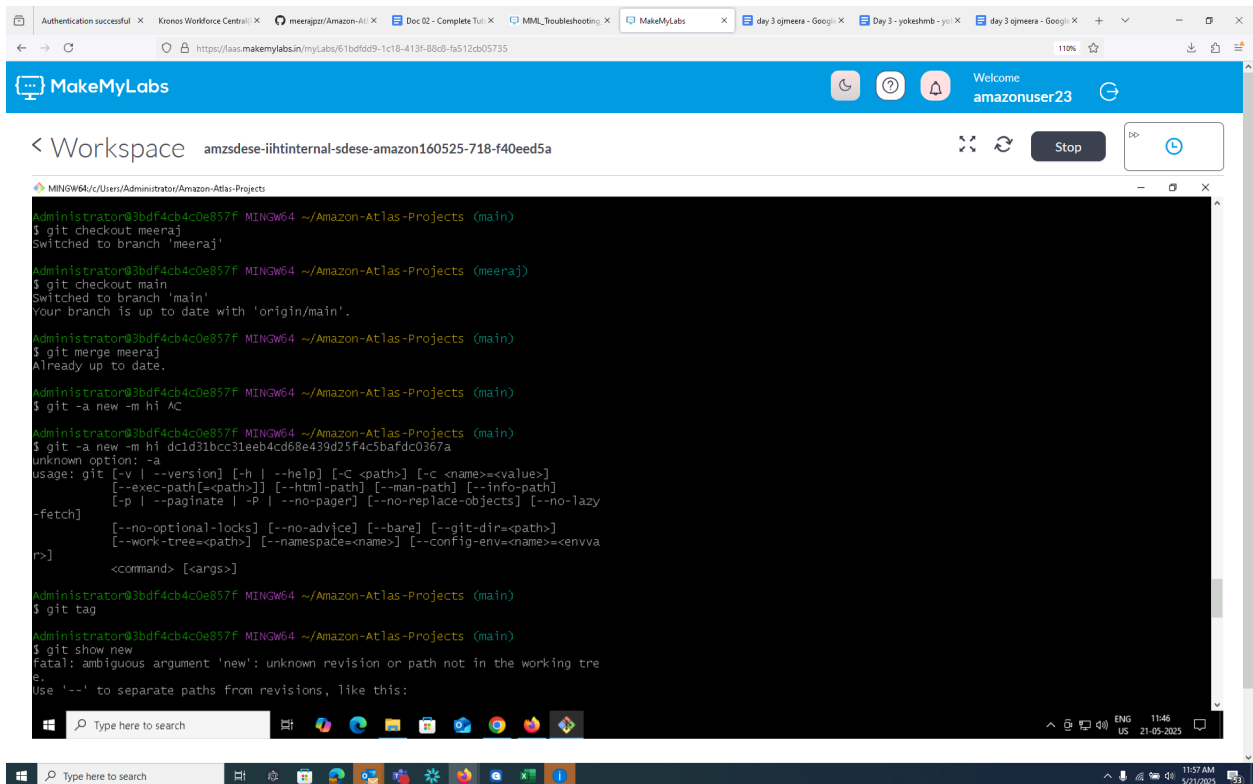
The screenshot shows a web browser window with the MakeMyLabs interface. The workspace is titled "amzsdese-iihtinternal-sdese-amazon160525-718-f40eed5a". A terminal window is open, displaying the following commands and output:

```
$ git add may21.txt
Administrator@3bdf4cb4c0e857f MINGW64 ~/Amazon-Atlas-Projects (main)
$ git commit -m "add may21.txt"
On branch main
Your branch and 'origin/main' have diverged,
and have 2 and 2 different commits each, respectively.
nothing to commit, working tree clean
Administrator@3bdf4cb4c0e857f MINGW64 ~/Amazon-Atlas-Projects (main)
$ git push origin main
To https://github.com/meerajpzn/Amazon-Atlas-Projects.git
! [rejected]        main -> main (non-fast-forward)
error: failed to push some refs to 'https://github.com/meerajpzn/Amazon-Atlas-Projects.git'
hint: updates were rejected because the tip of your current branch is behind
hint: its remote counterpart. If you want to integrate the remote changes,
hint: use 'git pull' before pushing again.
hint: See the 'Note about fast-forwards' in 'git push --help' for details.
Administrator@3bdf4cb4c0e857f MINGW64 ~/Amazon-Atlas-Projects (main)
$ git push origin main
To https://github.com/meerajpzn/Amazon-Atlas-Projects.git
! [rejected]        main -> main (non-fast-forward)
error: failed to push some refs to 'https://github.com/meerajpzn/Amazon-Atlas-Projects.git'
hint: Updates were rejected because the tip of your current branch is behind
hint: its remote counterpart. If you want to integrate the remote changes,
hint: use 'git pull' before pushing again.
hint: See the 'Note about fast-forwards' in 'git push --help' for details.
Administrator@3bdf4cb4c0e857f MINGW64 ~/Amazon-Atlas-Projects (main)
$ git push origin main -f
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 4 threads
```

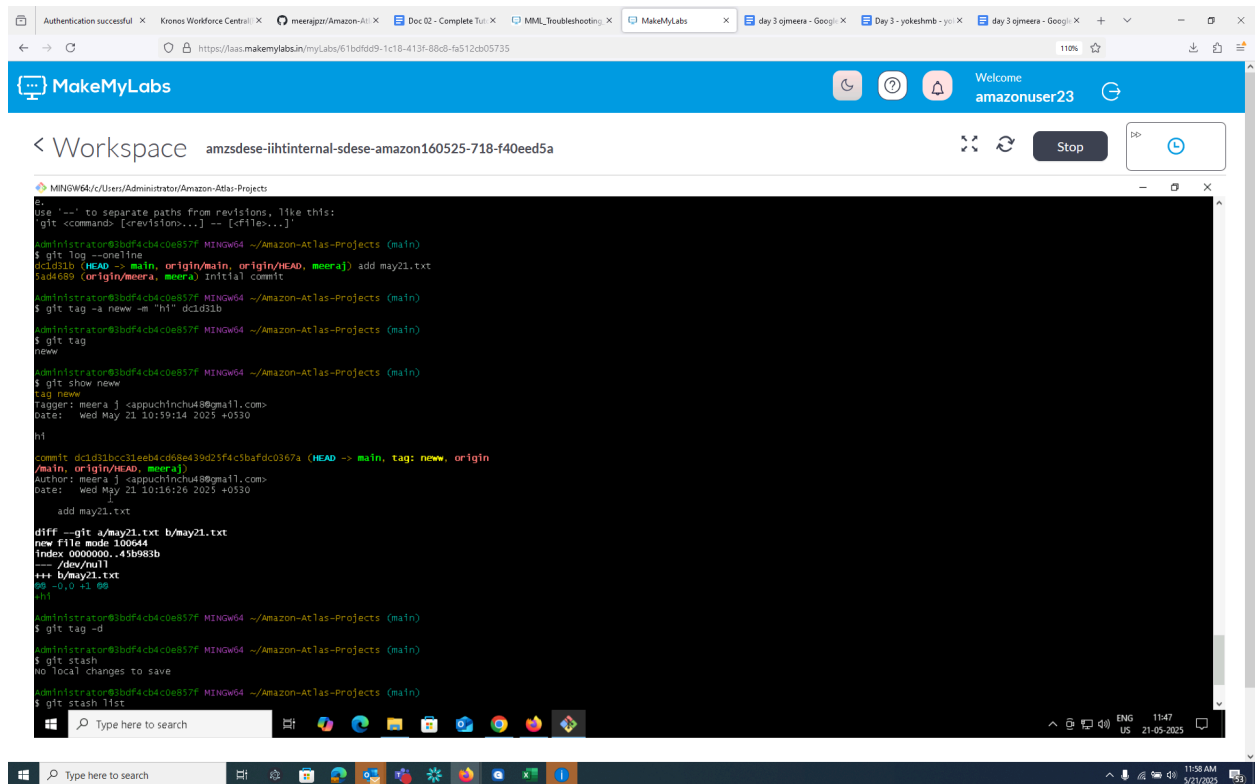
Task 2 : Branch



Branch merge



Task 3 : GIT tag



Task 4 : Acid Propties

Atomicity:

This property ensures that a transaction is treated as a single, indivisible unit. It either completes entirely or does not complete at all. If any part of the transaction fails, the entire transaction is rolled back, leaving the database in a consistent state.

Consistency:

This guarantees that a transaction will bring the database from one valid state to another. It ensures that any data written to the database must be valid according to all defined rules, such as constraints, triggers, and other integrity checks. If a transaction violates any of these rules, it will be rolled back.

Isolation:

Isolation ensures that the operations of one transaction are not visible to other transactions until it is complete. This prevents conflicts between concurrent transactions

and ensures that they do not interfere with each other. Different levels of isolation (like Read Committed, Repeatable Read, and Serializable) determine the degree of visibility.

Durability:

Once a transaction is committed, its changes are permanent, even in the event of a system crash or failure. The data is safely stored and will survive any kind of failure, ensuring that the database reflects the most recent committed transaction.