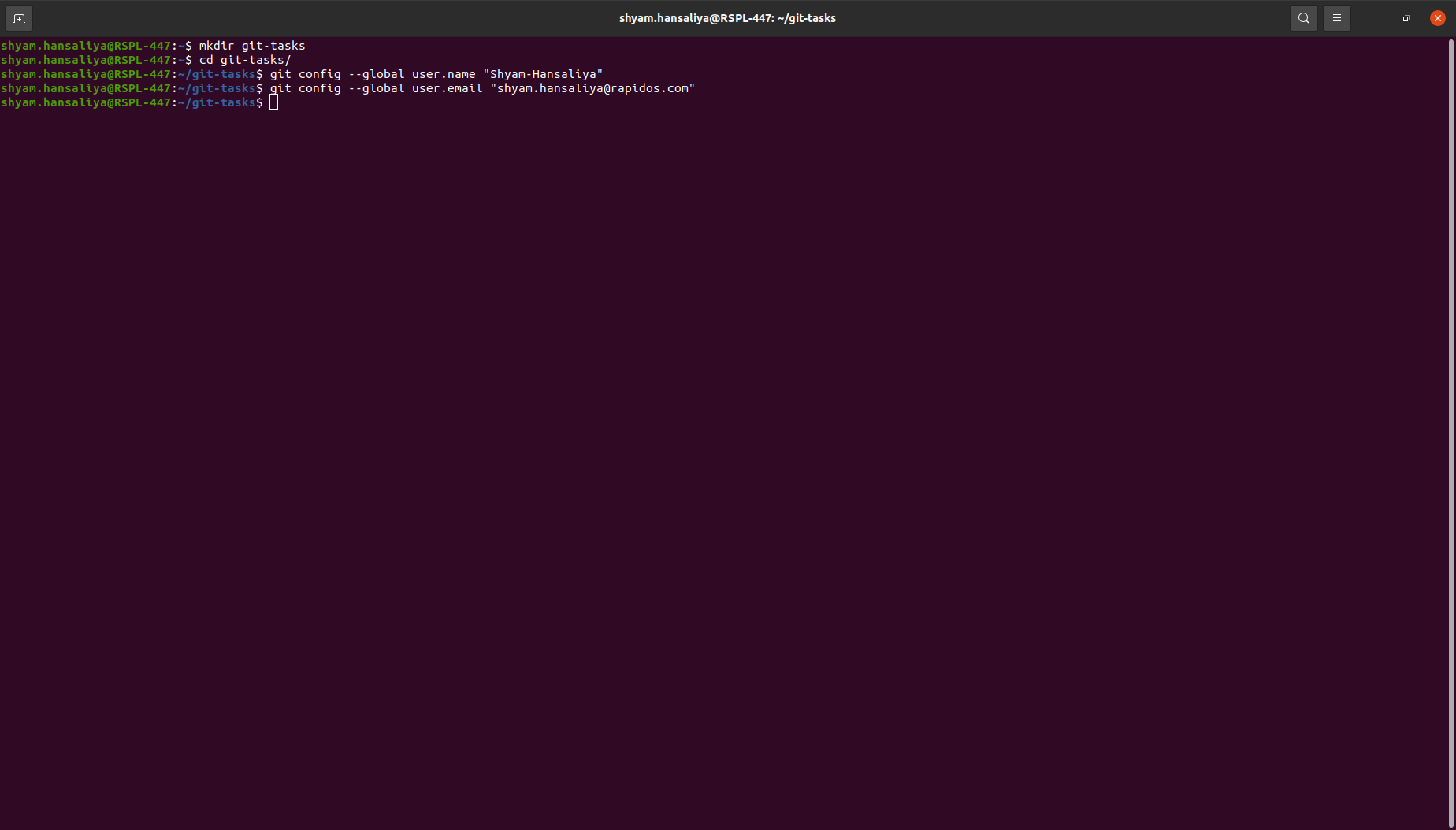
1. Configure your user name and email.

Mkdir git-tasks

Cd git-tasks/

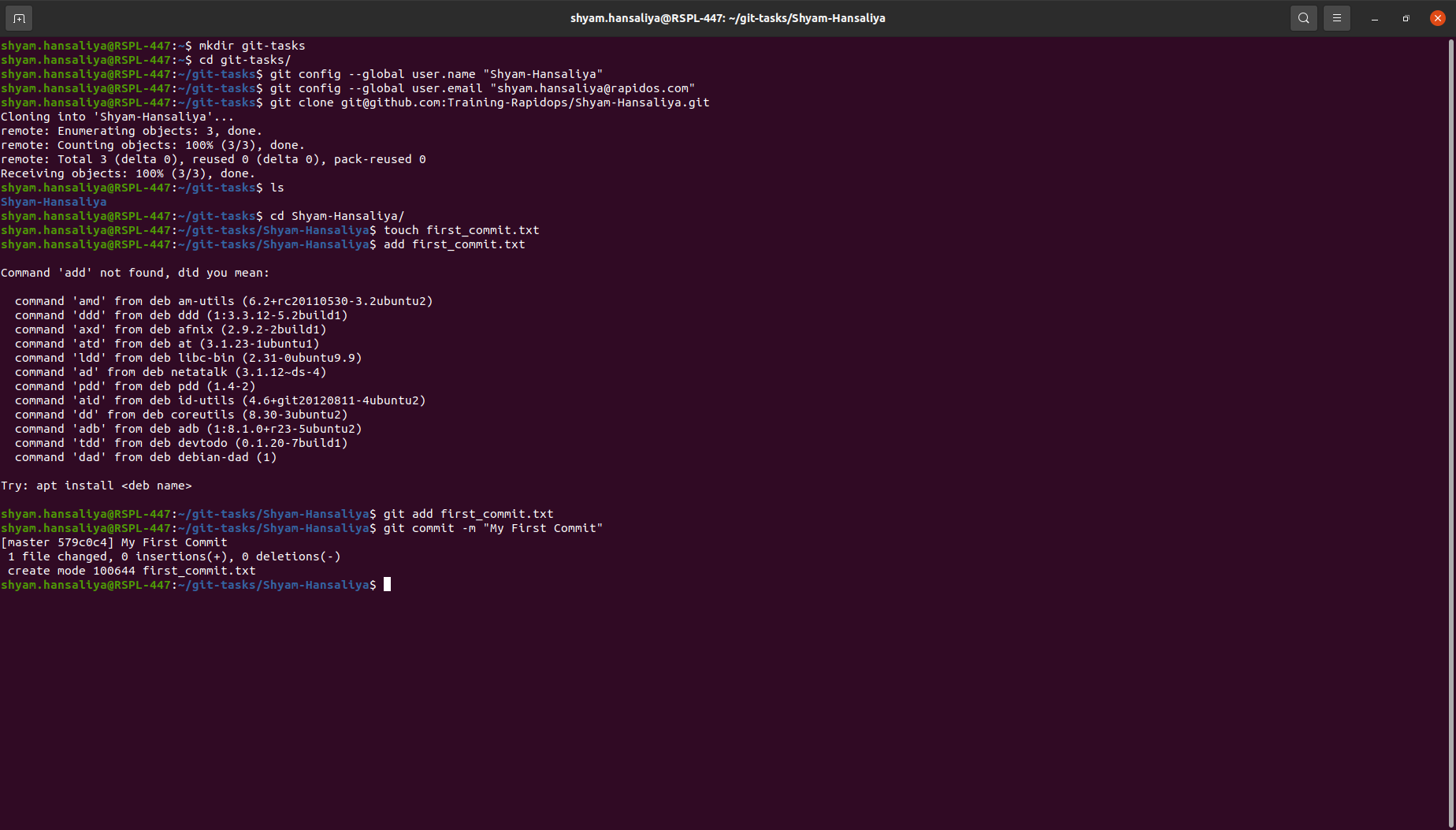
Git config –global user.name “Shyam-Hansaliya”

Git config –global user.email “shyam.hansaliya@rapidops.com”



1. Clone repo of your name from GitHub to the local system.

Git clone git@github.com:Training-Rapidops/Shyam-Hansaliya.git

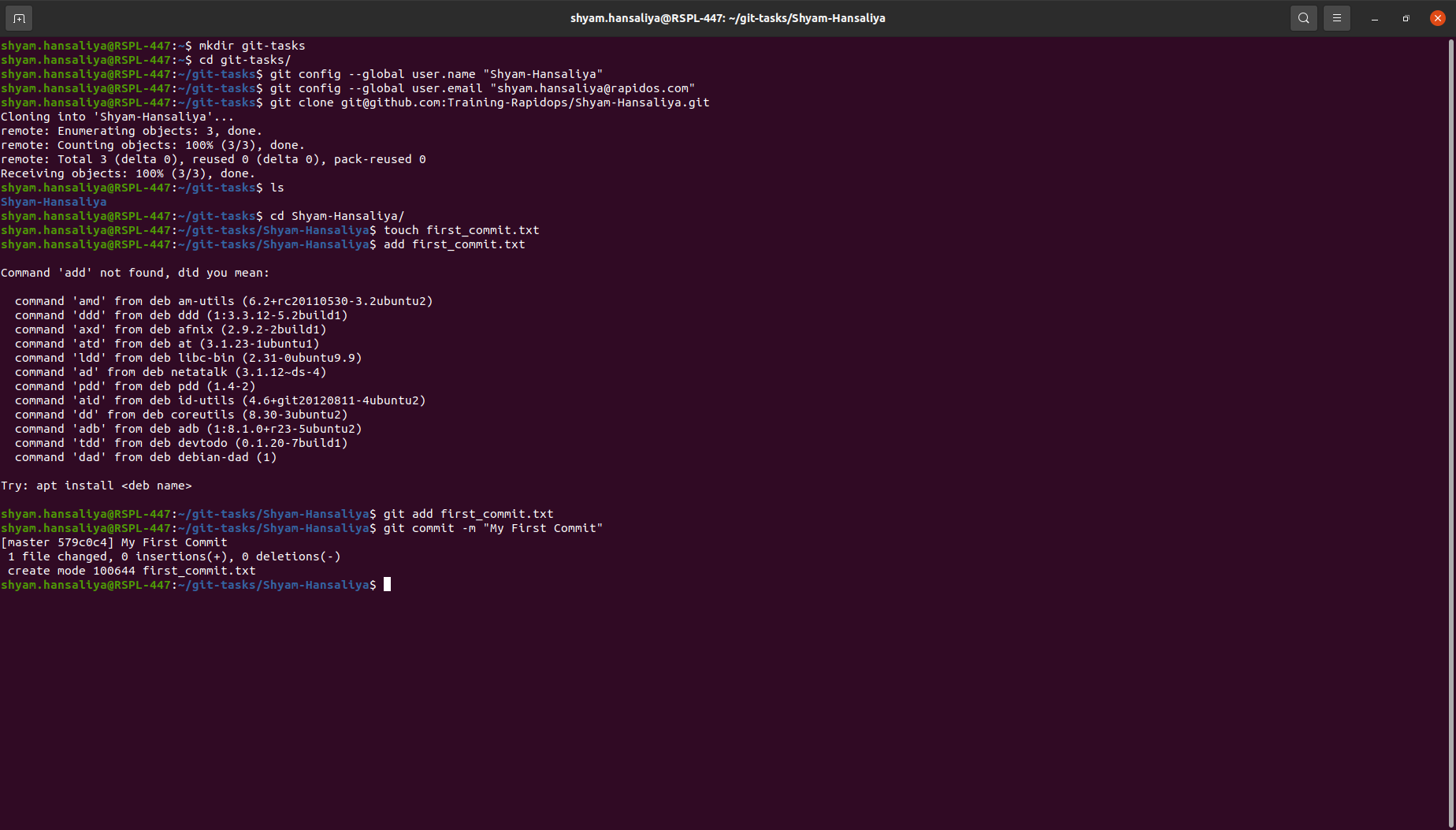


1. Create a file inside the repo, and make your first commit "My First Commit".

Touch first\_commit.txt

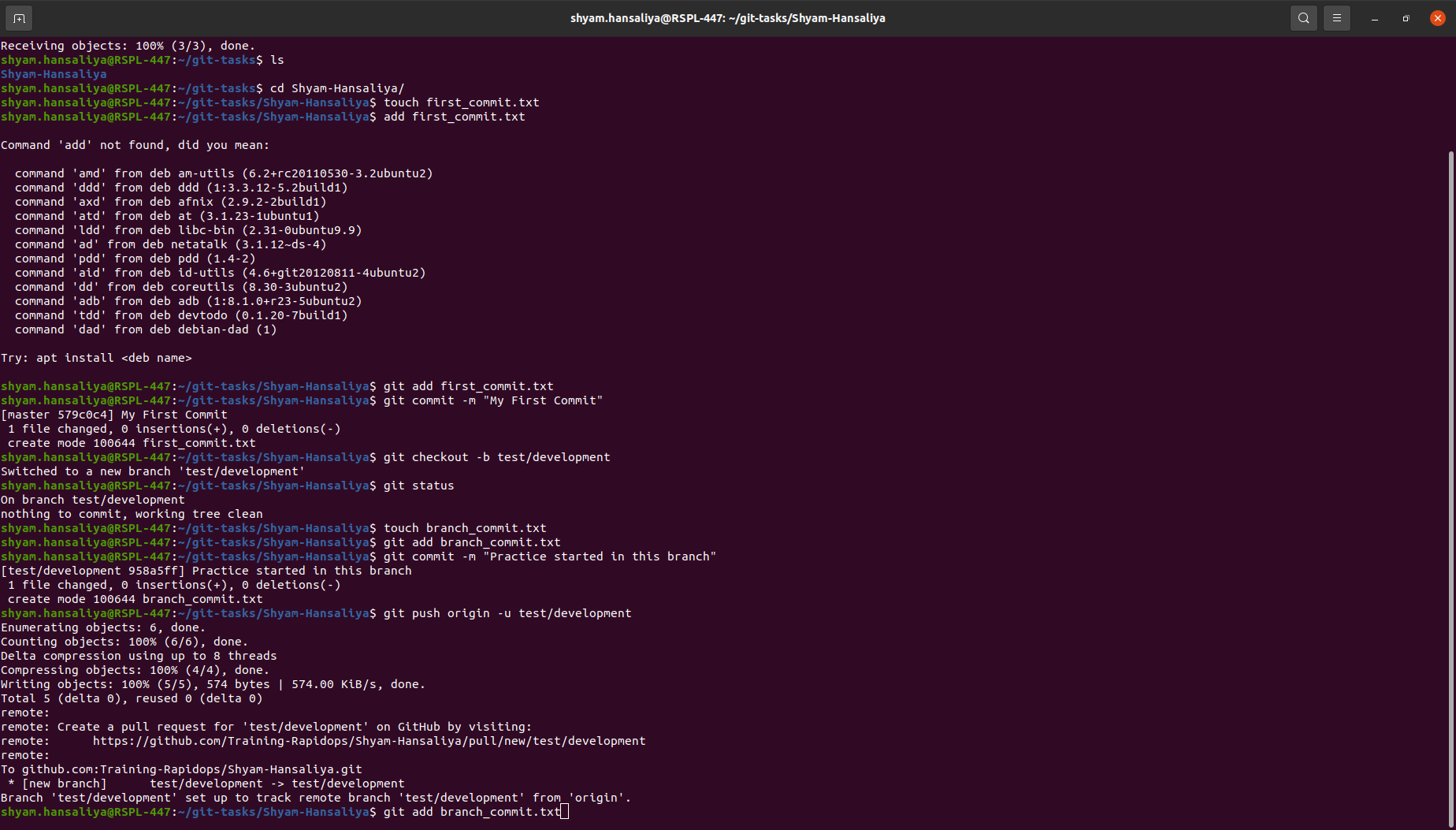
git Add first\_commit.txt

Git commit –m “My First commit”



1. Create and switch to the branch 'test/development' (create from the master branch and it should be from origin).

Git checkout –b test/development

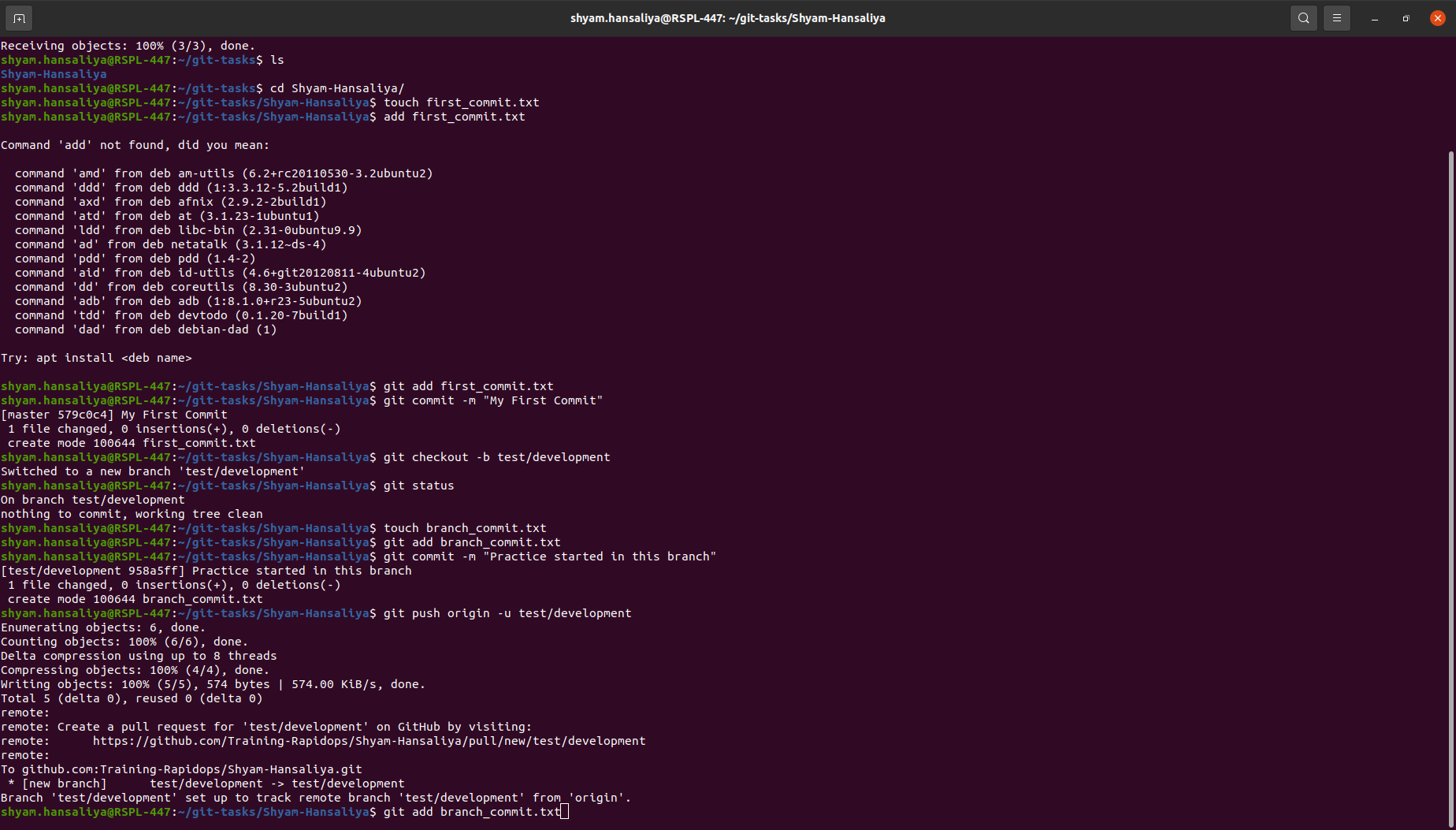


1. Add a file in this branch and commit your changes with the message "Practice started in test branch".

Touch branch\_commit.txt

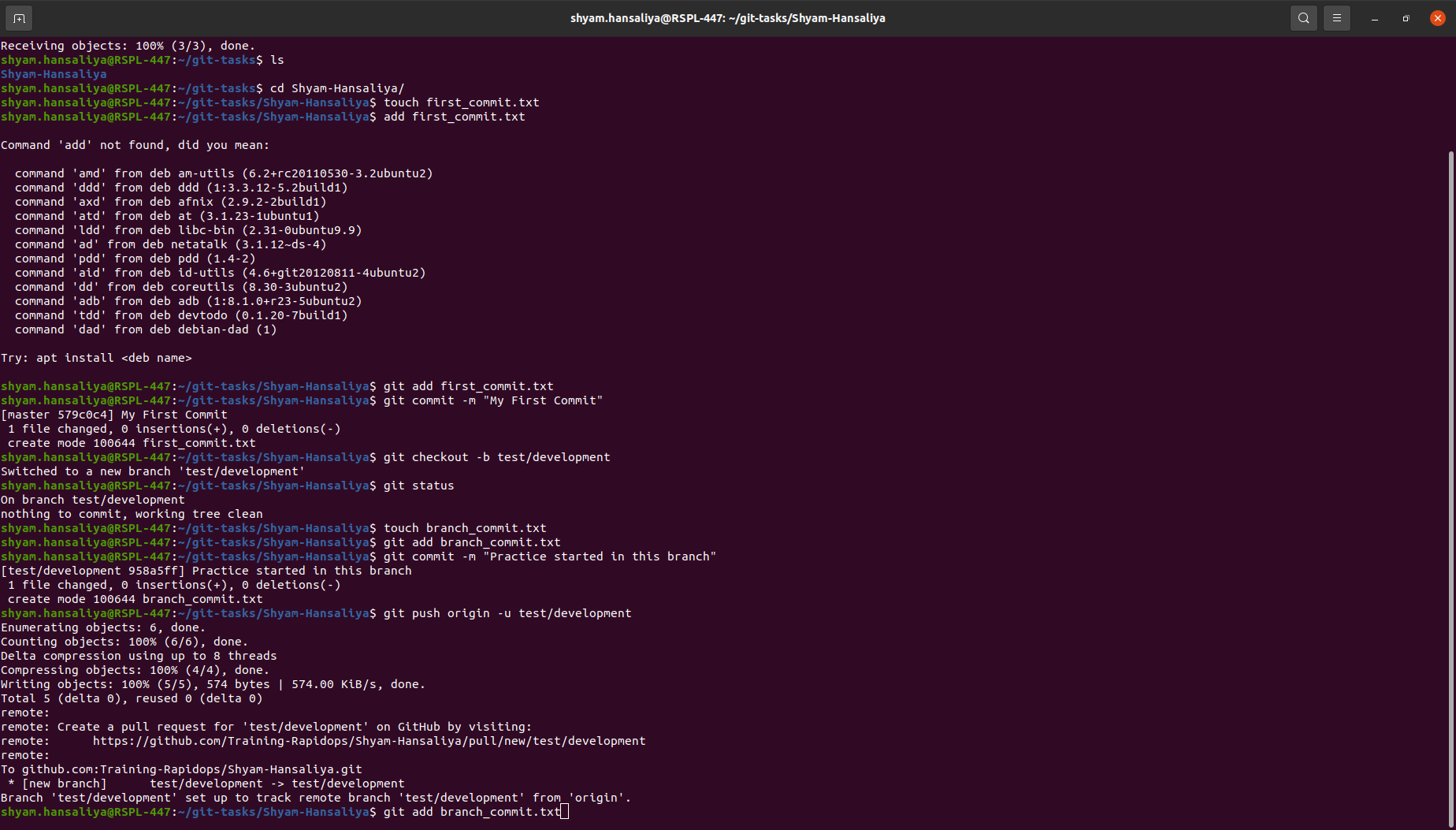
Git add branch\_commit.txt

Git commit –m “Practice started in this branch”

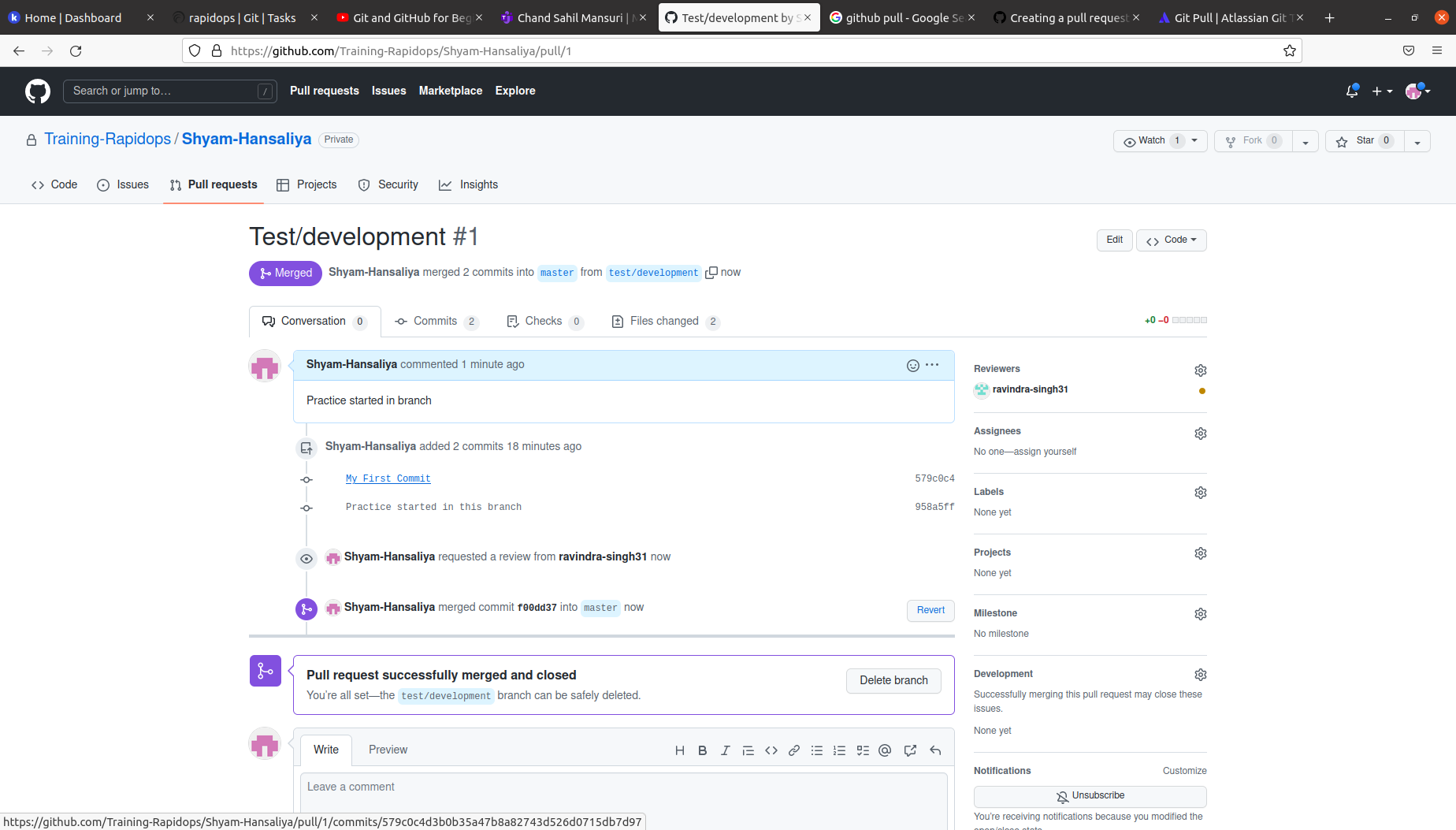


1. Now push your changes and this branch to the remote.

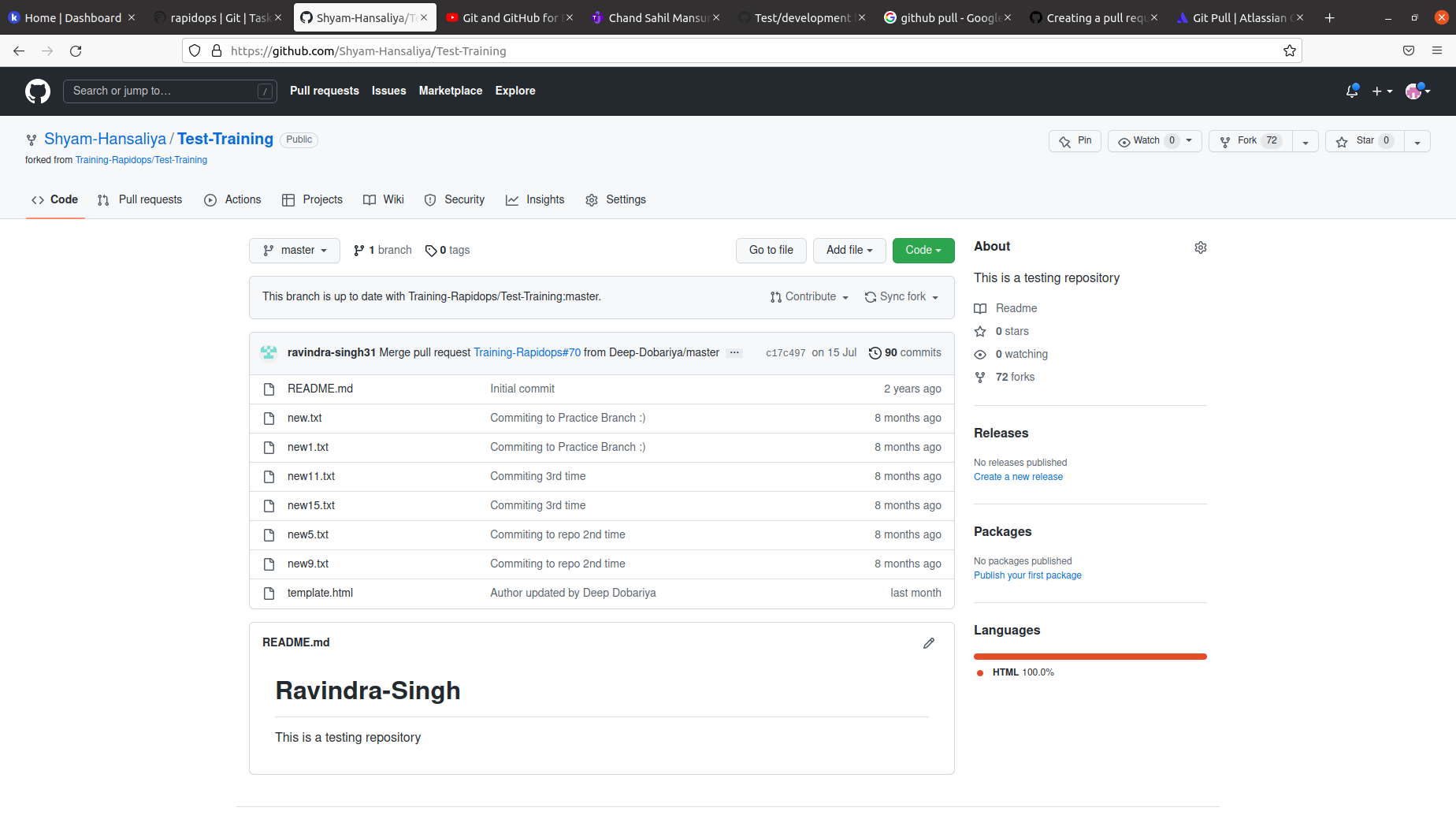
git push origin –u test/development



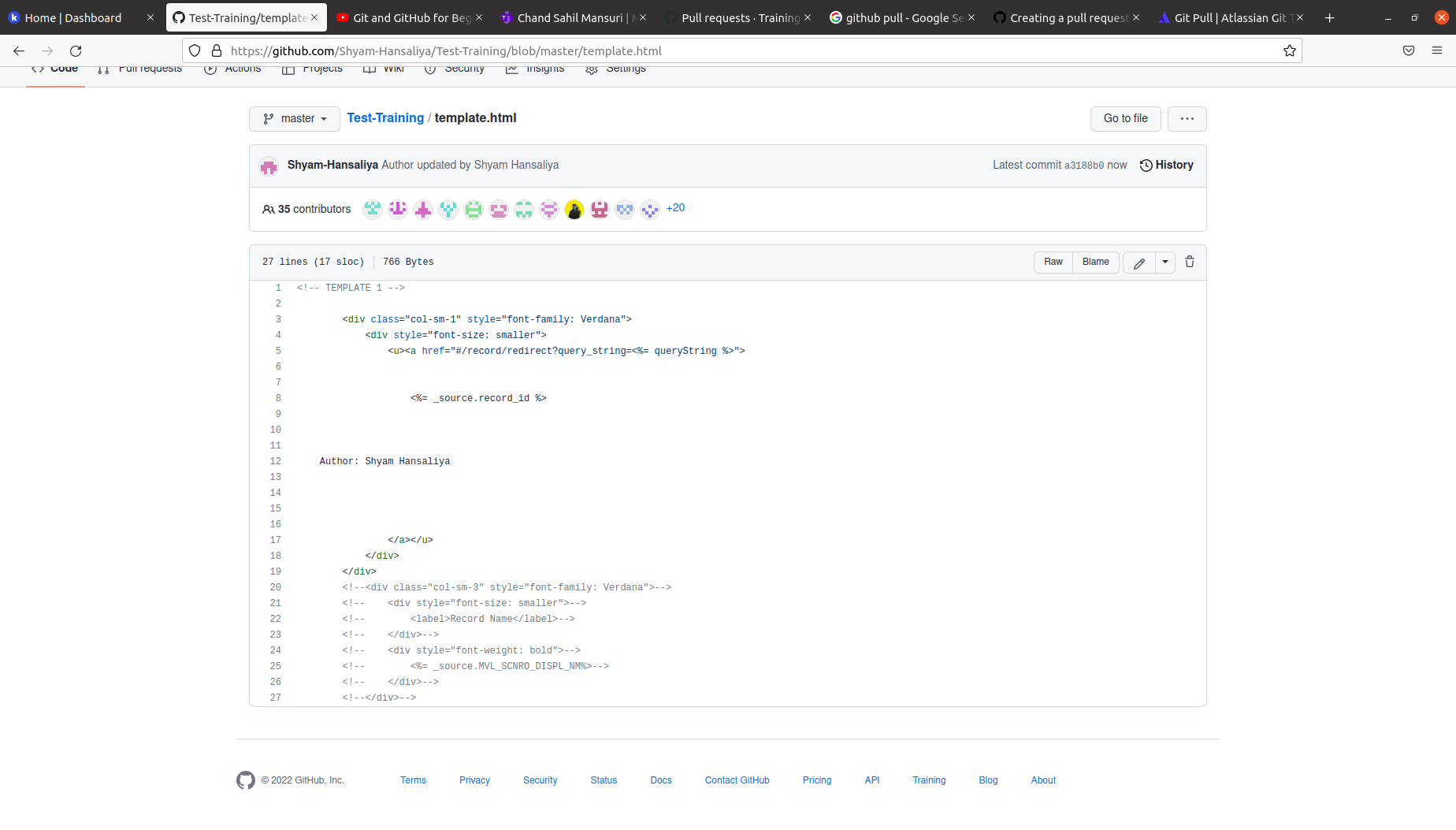
1. Go to your GitHub repository and create a pull request to merge this branch in master. Also, add Ravindra & your mentor as reviewers.



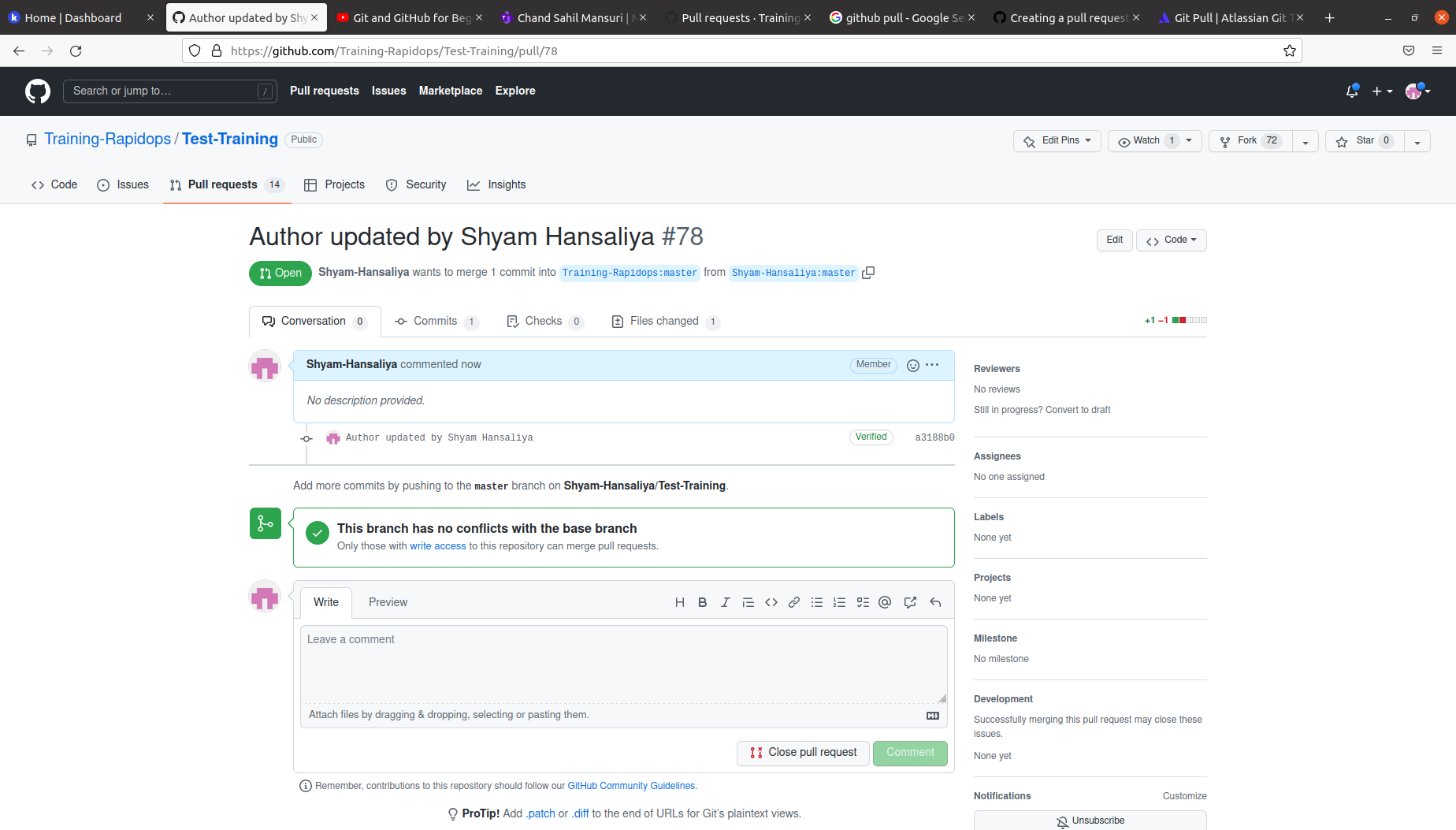
1. Fork a public repository [**Test-Training**](https://github.com/Training-Rapidops/Test-Training)



1. Make a change in the template.html file by adding 'author: {your-name}' and add commit 'Author updated by {your-name}'.



1. Give PR to merge your forked repo in the **Test-Training** repo.



1. Come back to your repo, where you create a branch 'practice' and add 3 commits then rebase it with the master.

touch file1.txt

add file1.txt

git add file1.txt

git commit -m "commit 1"

gedit file1.txt

git add file1.txt

git commit -m "second commit"

gedit file1.txt

git add file1.txt

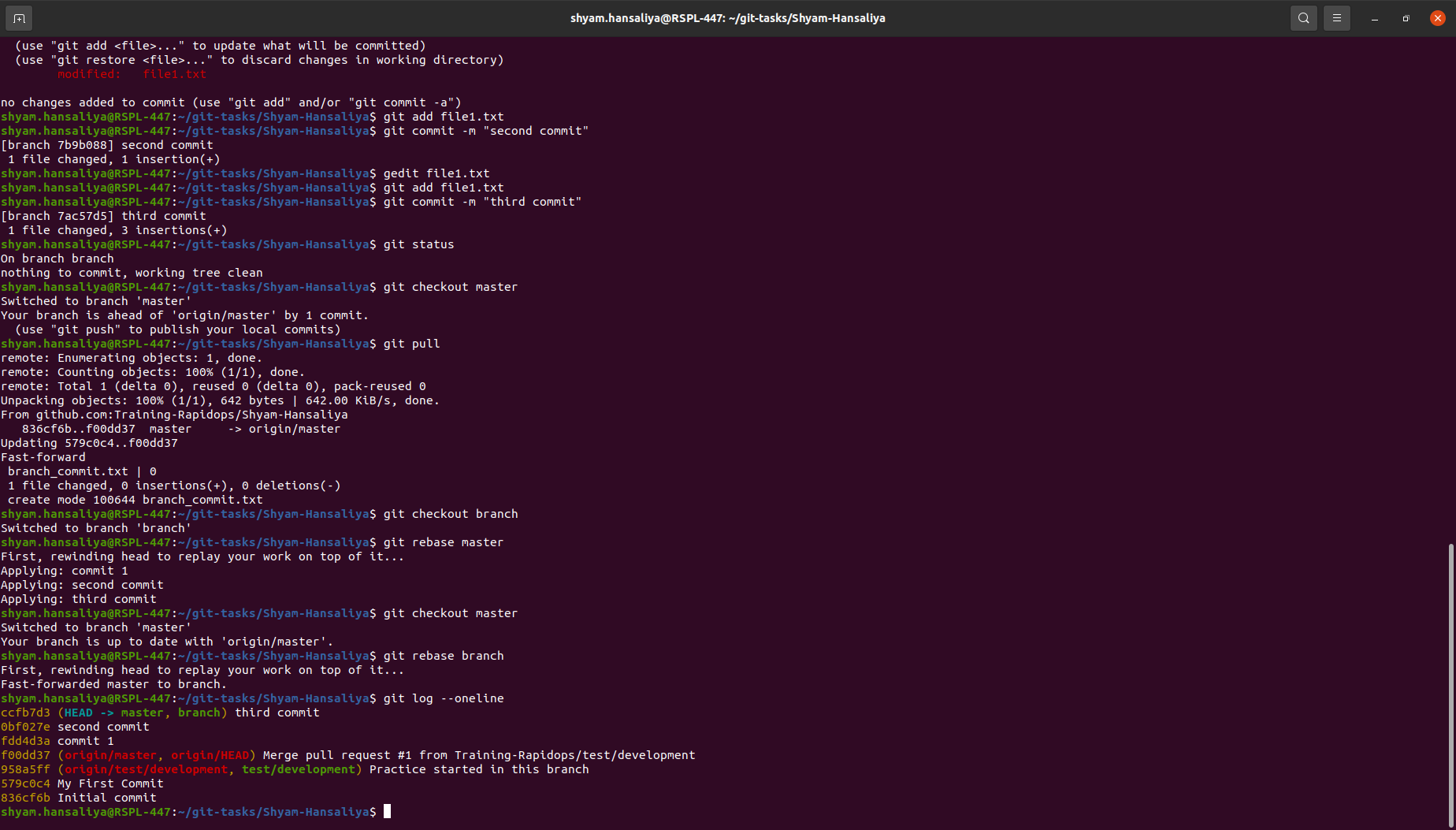
git commit -m "third commit"

git checkout branch

git rebase master

git checkout master

git rebase branch



1. Push your changes to remote, then Add 3 commits again and squash them into the first commit by keeping the message "Rebase squash done".

Git push

ls

touch revert.txt

gedit revert.txt

git add revert.txt

git commit -m "first commit of revert"

gedit revert.txt

git add revert.txt

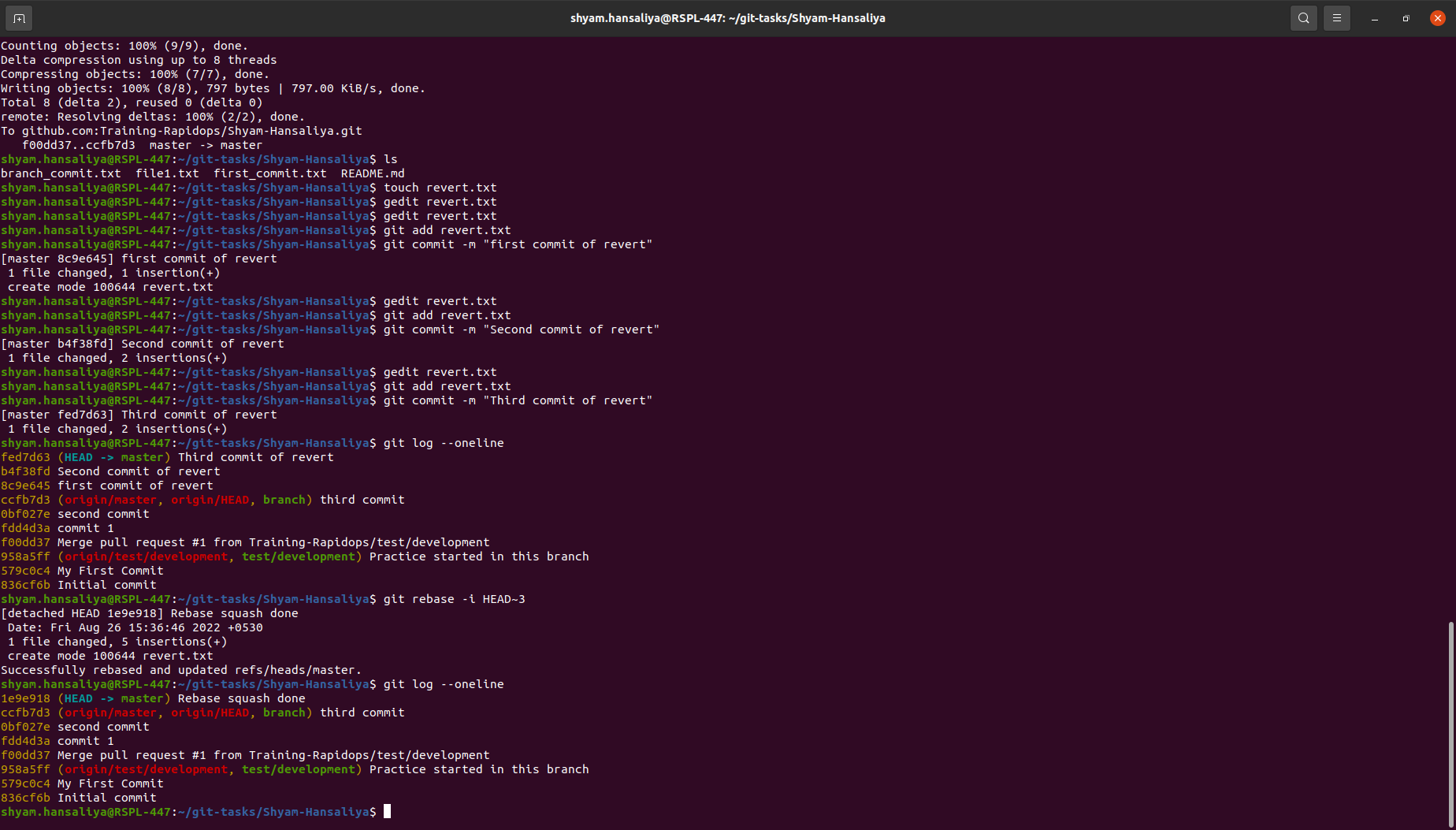
git commit -m "Second commit of revert"

gedit revert.txt

git add revert.txt

git commit -m "Third commit of revert"

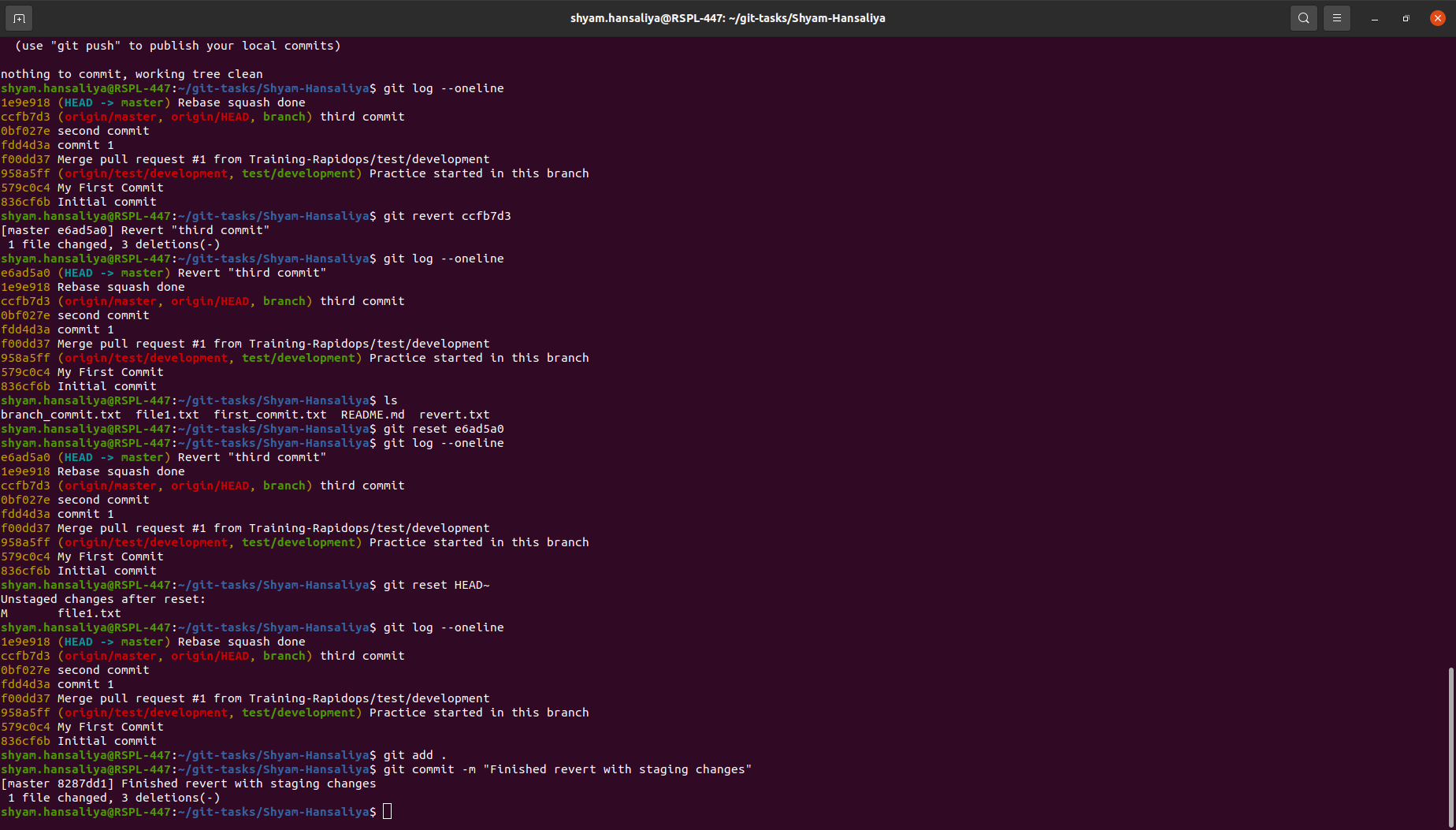
git rebase -I HEAD~3



1. Now revert these changes but note that the changes must be retained in the commit history. (use default revert commit message)

git log --oneline

git revert ccfb7d3

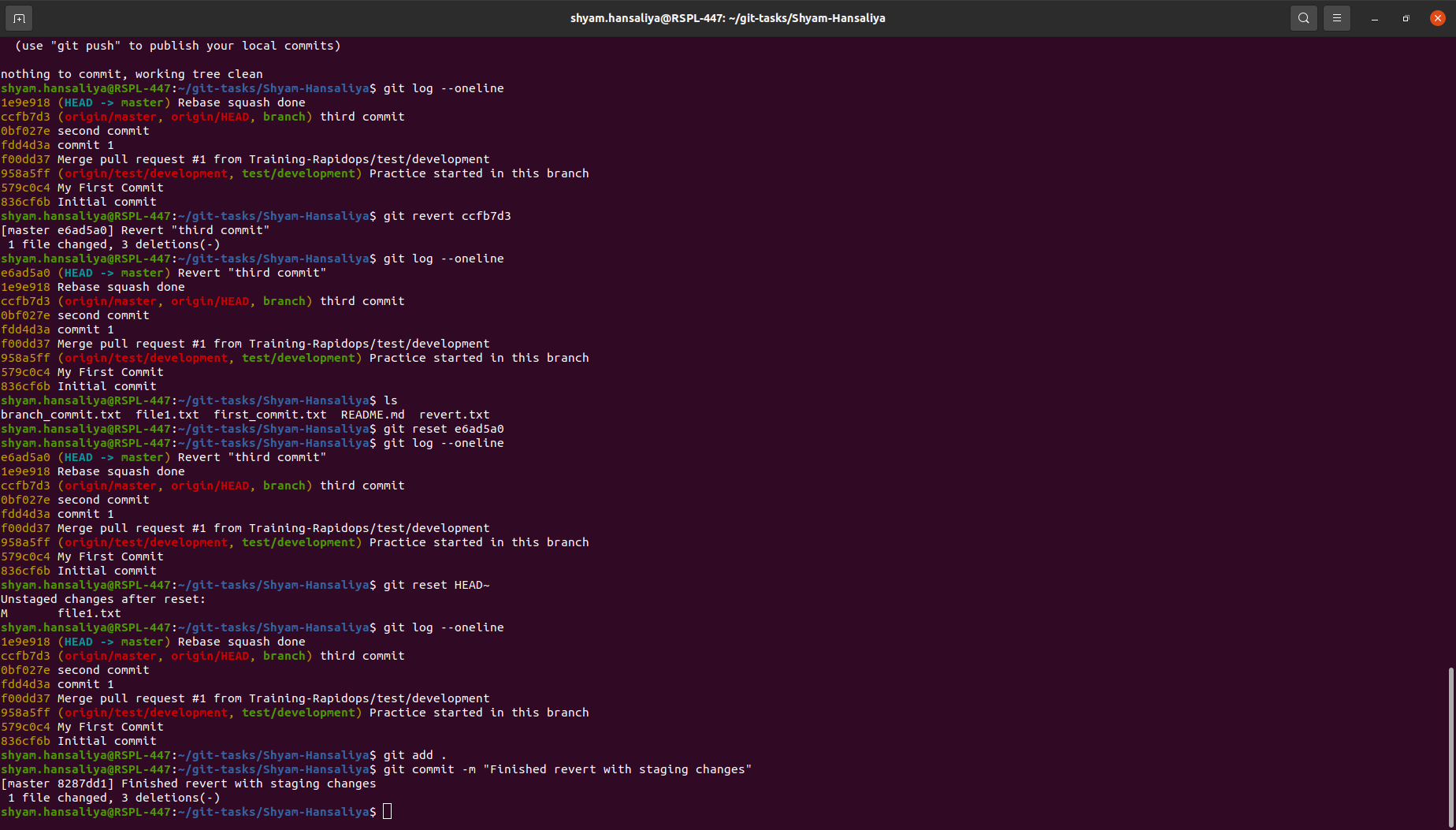


1. Reset your last commit without losing the changes and then commit with the message "Finished revert with staging changes".

git reset HEAD~

Git add .

git commit -m "Finished revert with staging changes"

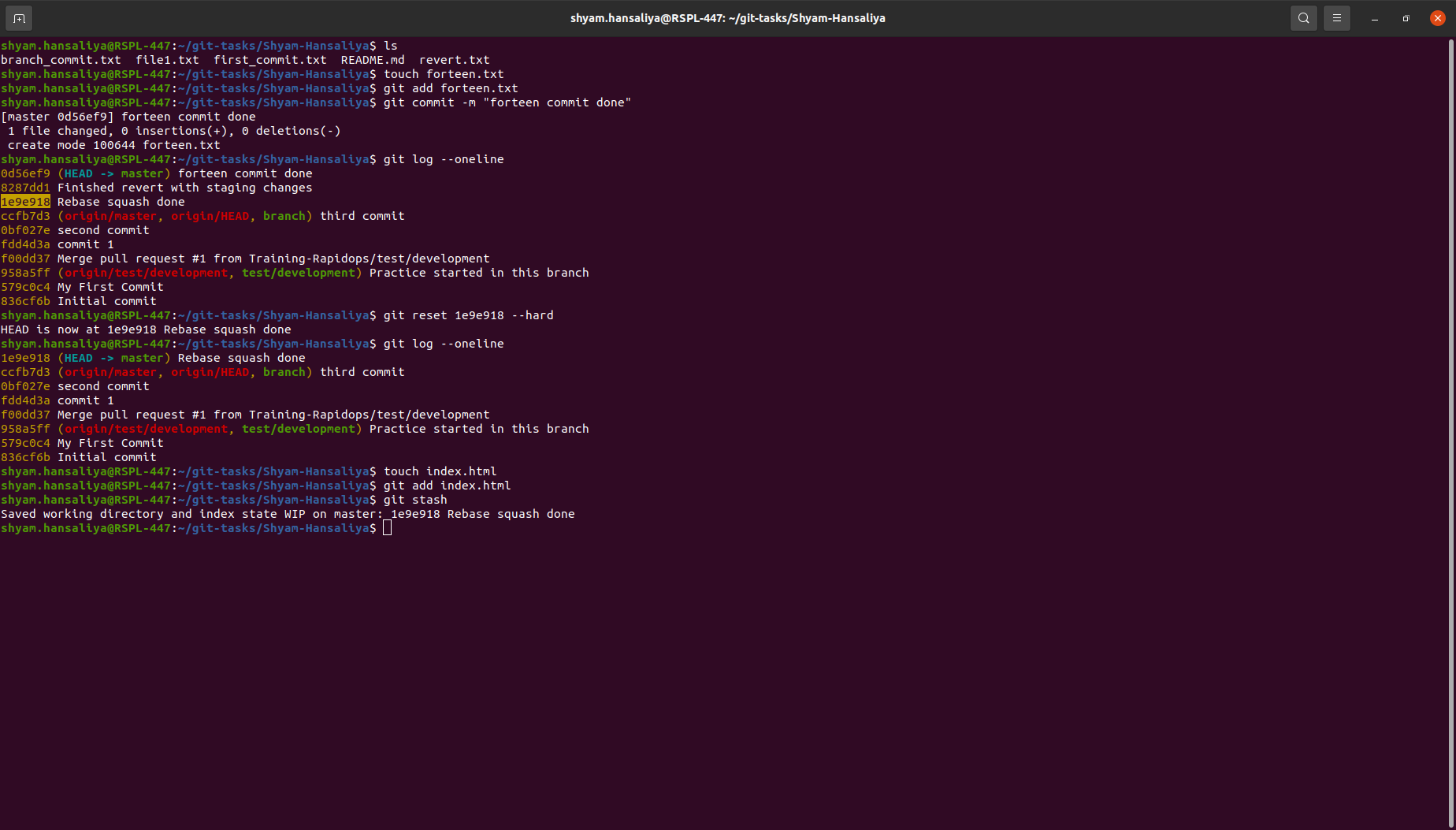


1. Create one commit then perform a hard reset such that you're back to the commit with the message "Rebase squash done".

git commit -m "forteen commit done"

git log –oneline

git reset 1e9e918 --hard

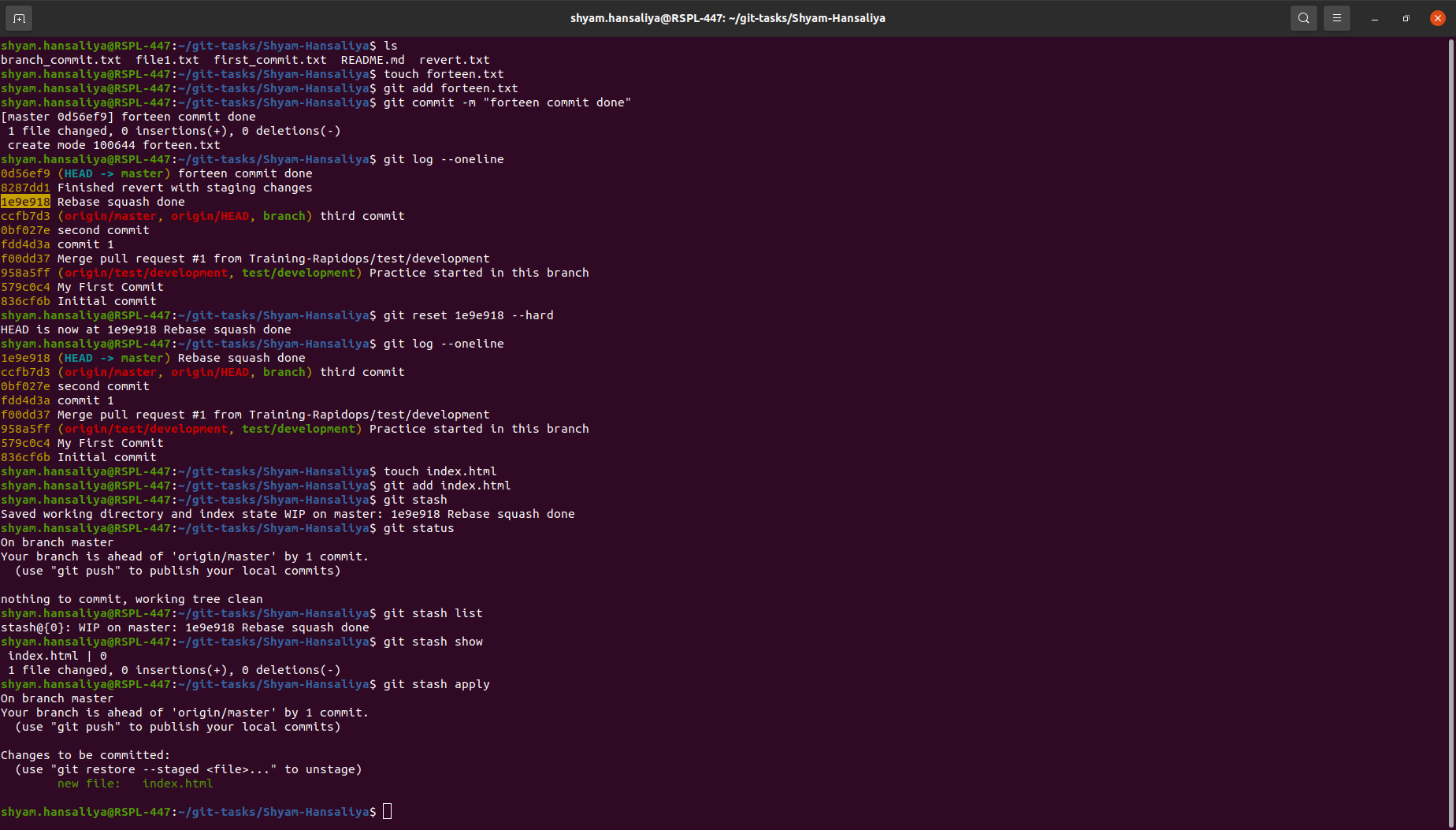


1. Create a file index.html, and add it to the staging index then stash it.

touch index.html

git add index.html

git stash

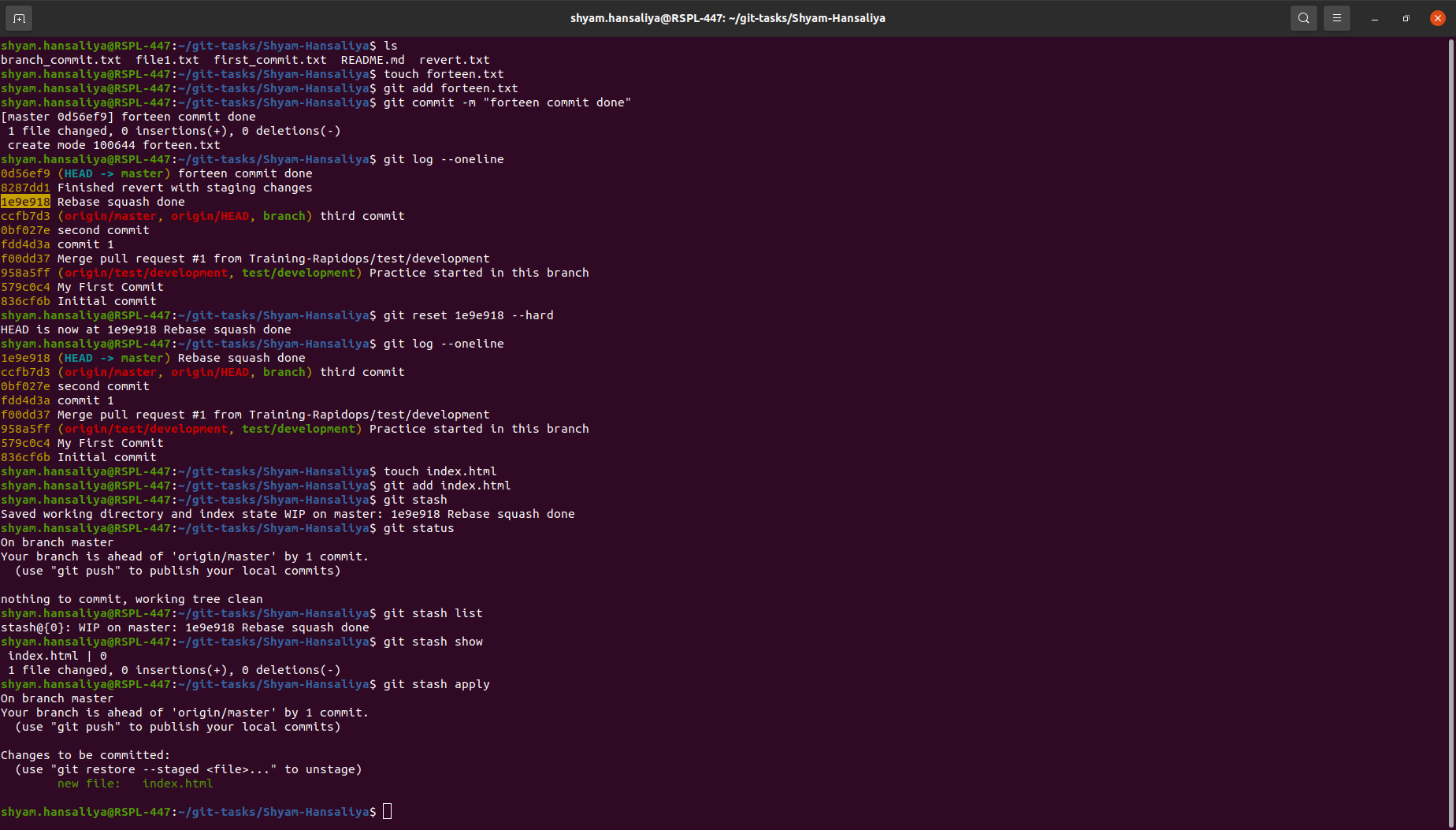


1. Check the list of stash, what changes are there in the stash, then bring your changes from stash.

git stash list

git stash show

git stash apply



1. Commit with a message "Revert, Reset & Stash done".

git commit -m "Revert, Reset & Stash done"

