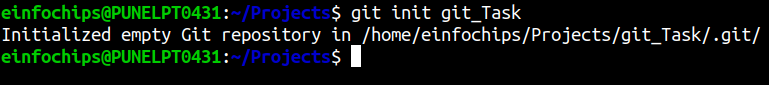
**Steps for REWRITING HISTORY WITH INTERACTIVE REBASE**

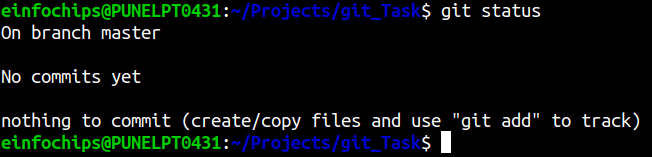
* Firstly, lets create a repository named git\_task using the command:



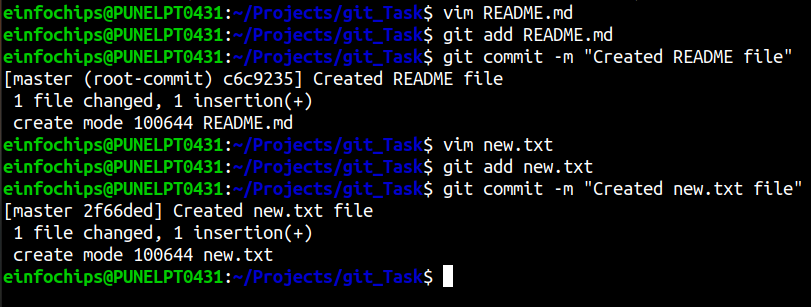
* Then, let’s move to the repository by using command:



* Let’s check the status of our git repository by doing:

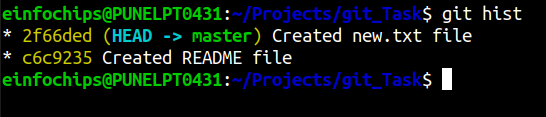


* As we are branch master, let’s create 2 commits by adding and committing 2 files on master branch.

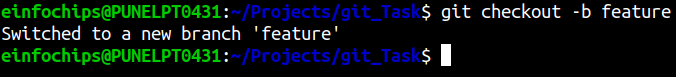


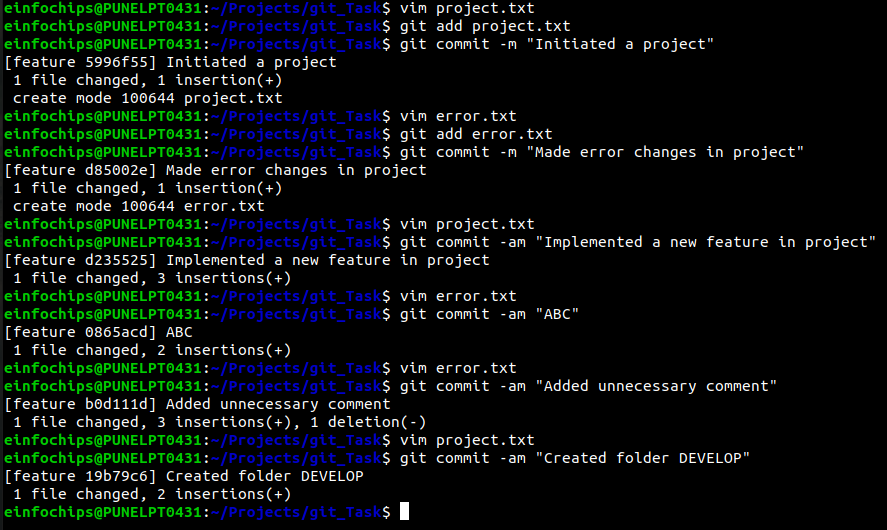
We have created 2 files on the master branch and made the commits.

* We can check the commit history we have done in this repo by doing:



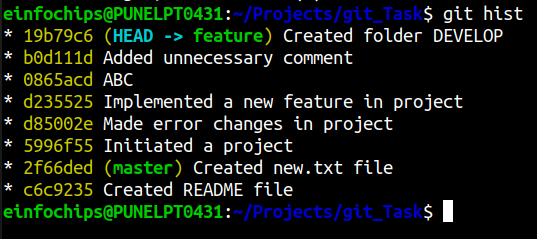
* Let’s create a new branch named ‘feature’ so that we can make commits in that branch and perform interactive rebase.



* Now let’s create 2 files named project.txt and error.txt on this branch.We will add some content in these files.

We made 6 commits by making changes in project.txt and error.txt. Among these commits, some are meaningful and some are irrelevant.

* If we see the commit history, we can see we are in the feature branch and made 6 commits in that branch.

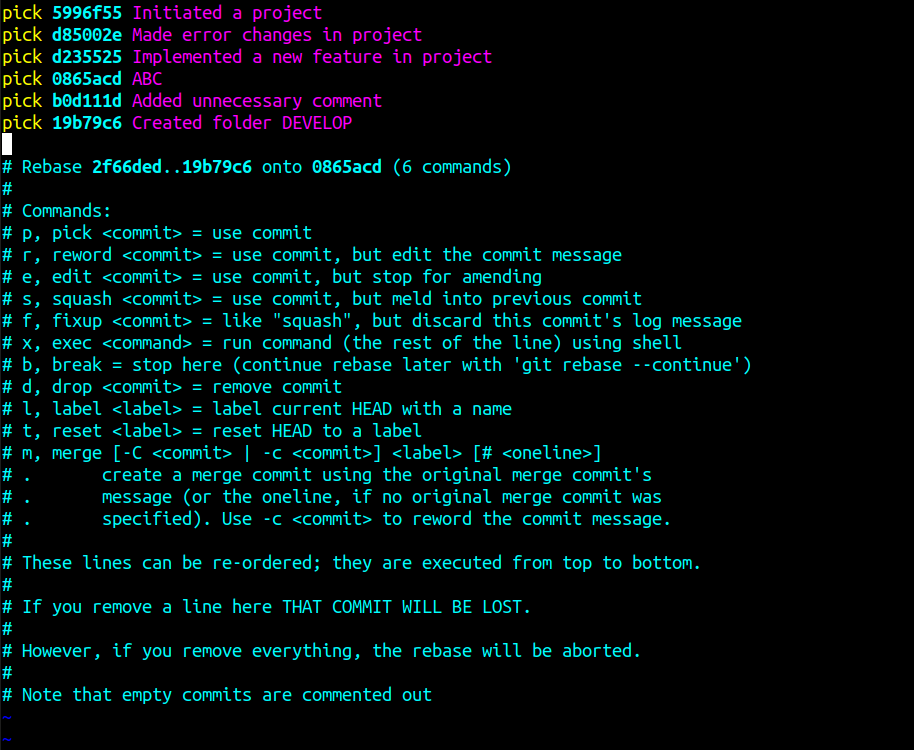


* Now we need to squash irrelevant or unnecessary commits into one, so that our history looks clean. For doing this, we use the command:



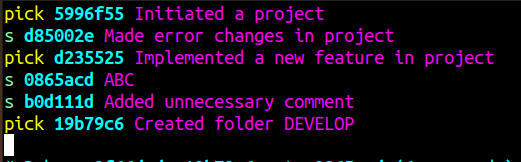
This command opens the default editor and shows us all the commits after the commit we have mentioned i.e. 2f66ded. This commit was made on the master branch, and all the commits after this one were done on the feature branch, which is what we need.

* After running the above command, the following window opens up in the default editor:



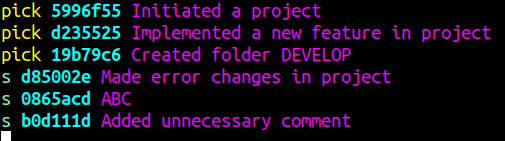
As we can see, all the commits made in the feature branch are present above and now we can squash and reorder commits as we need.

* We need to squash irrelevant/unnecessary commits into one and reorder if necessary. To squash the commits, we need to replace the word ‘pick’ with ‘s’ or ‘squash’, for the commits we need to squash.

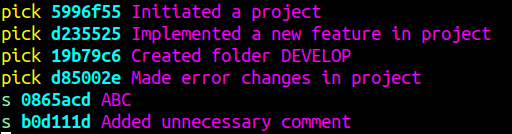


So, we mentioned ‘s’ in front of the commits we want to squash.

* Now, we need to reorder the commits, we can do this as follows:

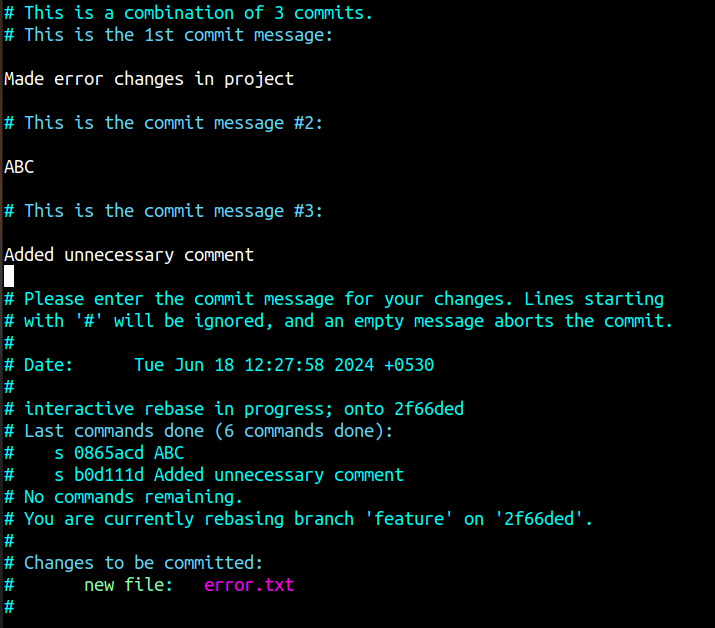


* Squashing squashes all the commits we want to squash in the previous commit. So, we will make one change and add ‘pick’ in front of the commit that is first mentioned to be squashed.



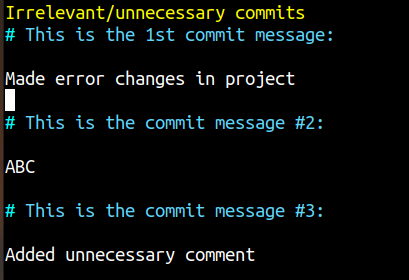
This will squash the bottom 2 commits into the previous commit we gave ‘pick’ i.e. the commit d85002e. Save the changes written and save the file by pressing “:wq”.

* After saving the file, we will be prompted with a new window which will ask us to give the commit message for the squashed commits. The window looks like this:



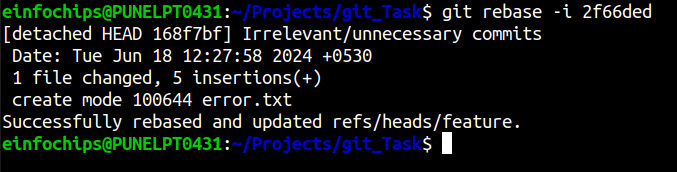
We can see the squashed commits also.

* Now, we will edit the first line by removing the ‘#’ in front of it, to give one commit message to the squashed commits. We can also change the commit message of the individual squashed commits.

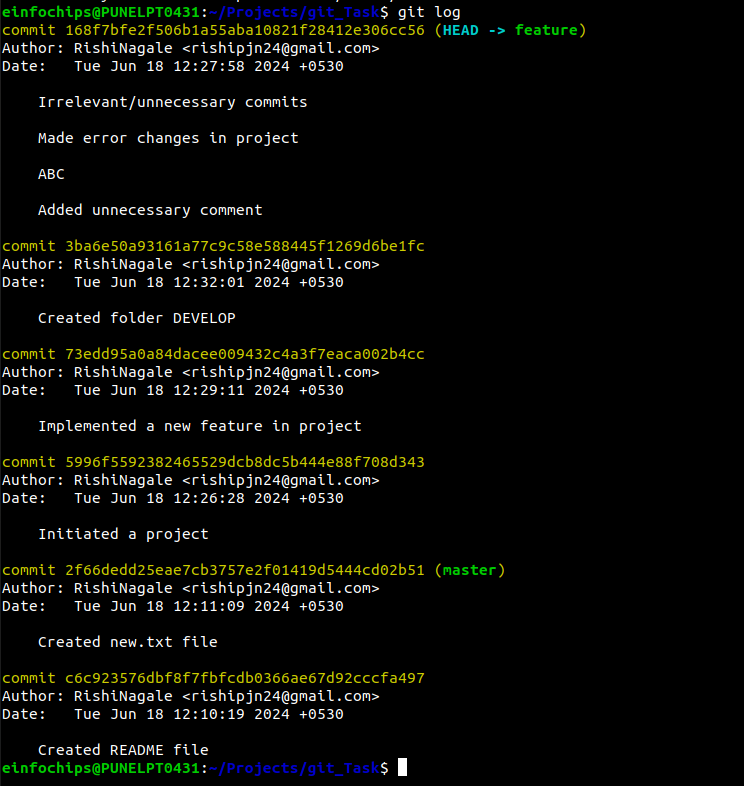


We changed the commit message to the squashed commits as “Irrelevant/unnecessary commits. Save the written changes.

* We get the message in terminal as:



* Now commit history will look this:



We have squashed and rebased and got a clean history.

* 