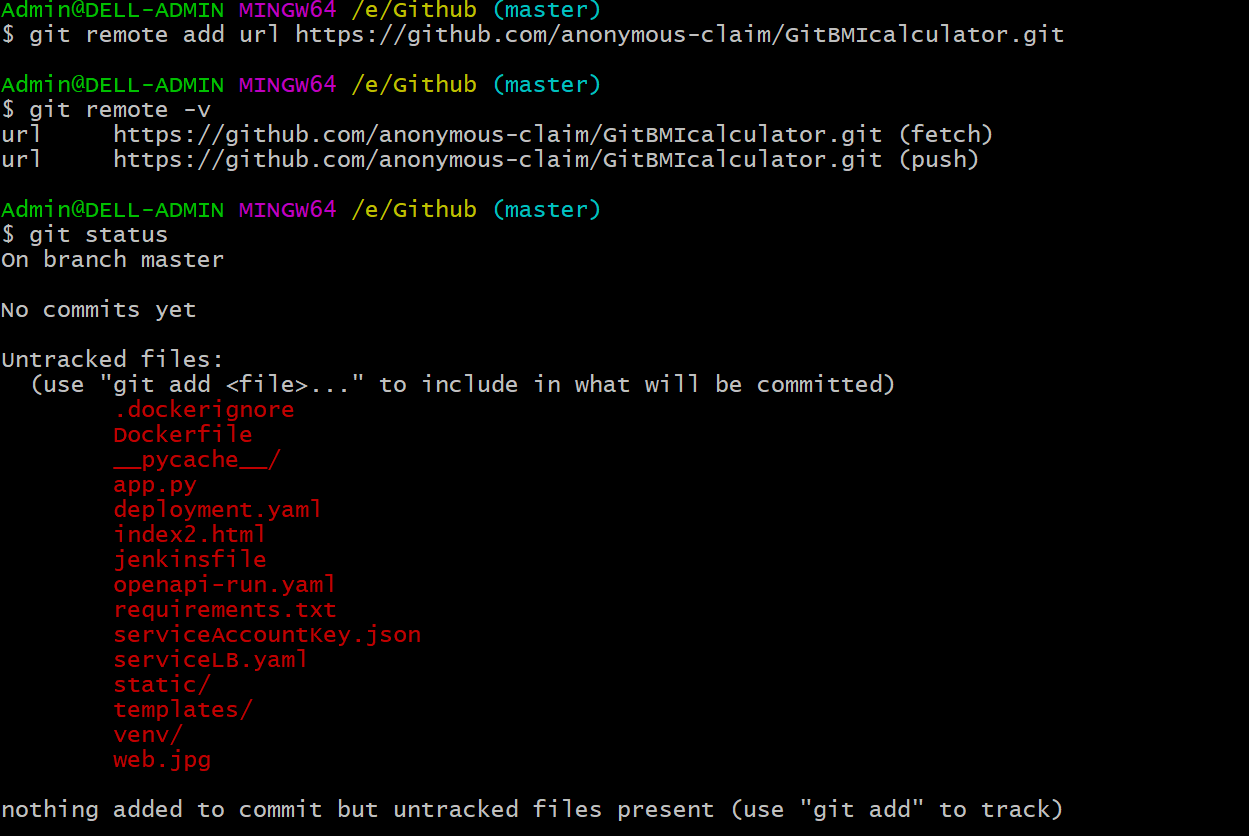
**GIT TASK**

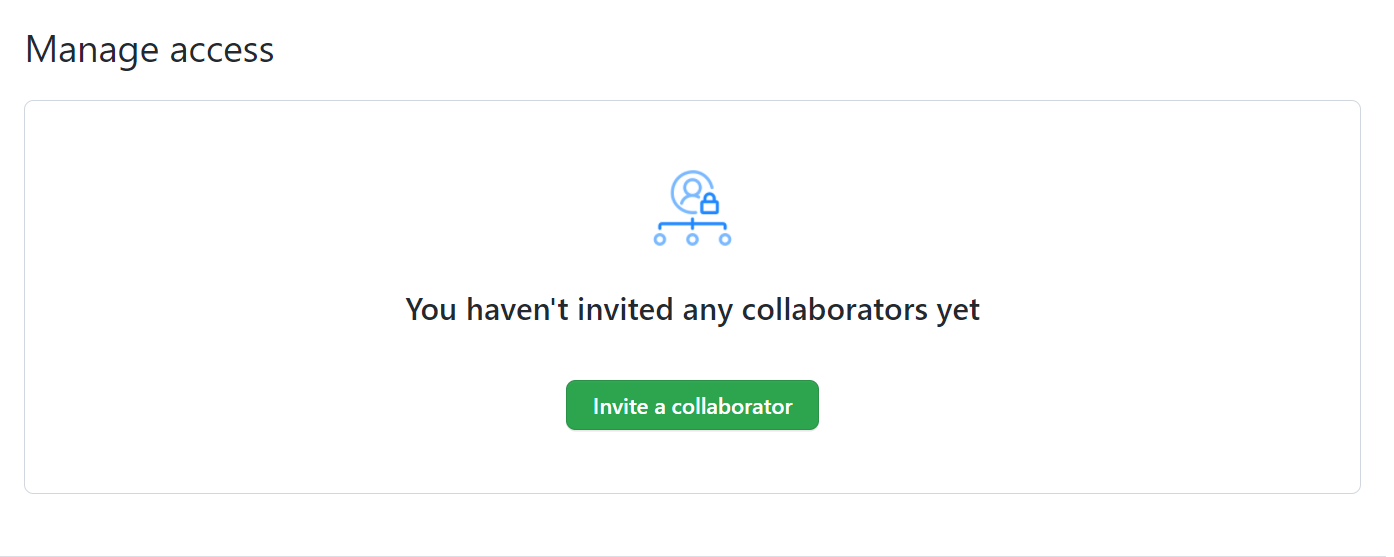
**Steps to be followed: -**

* Github Repository is created, collaborator is invited for enhancement.
* Branches are created namely Master, Develop and Feature
* Any changes are pushed to Feature branch
* A Pull Request (PR) is created for those changes
* Reviewer is assigned to Check the correctness of the PR
* Manager will either approve the PR by merging to develop branch or suggest the feedback
* Force push/commit and revert the changes
* Merge development branch to production branch
* Version release using tags
* Importance of Readme and. gitignore files

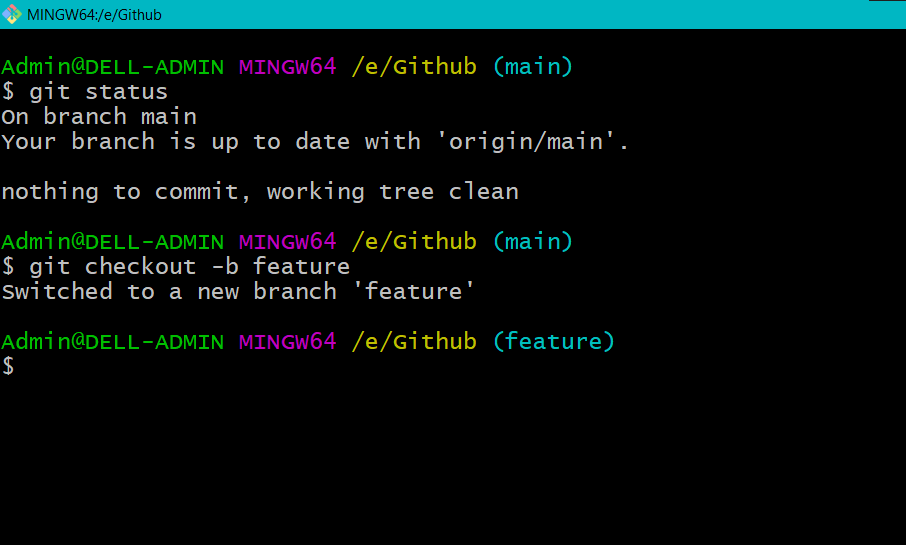
1. **Create a Github repository and push the code from local:**

****

1. **Add Collaborator to your git repo**

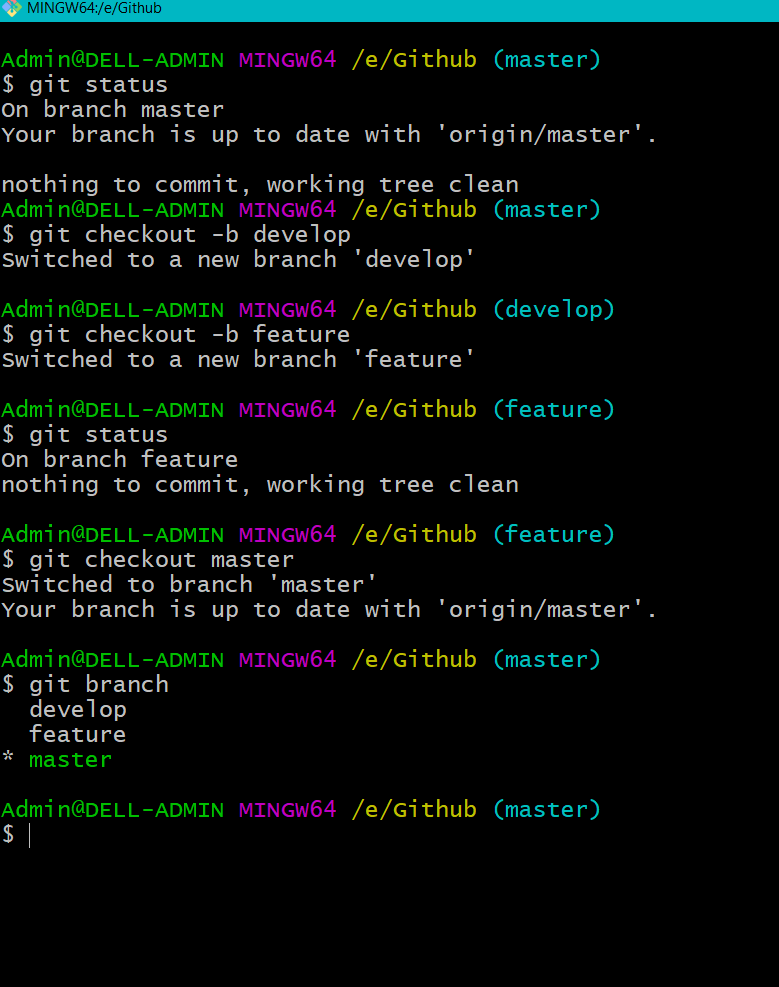
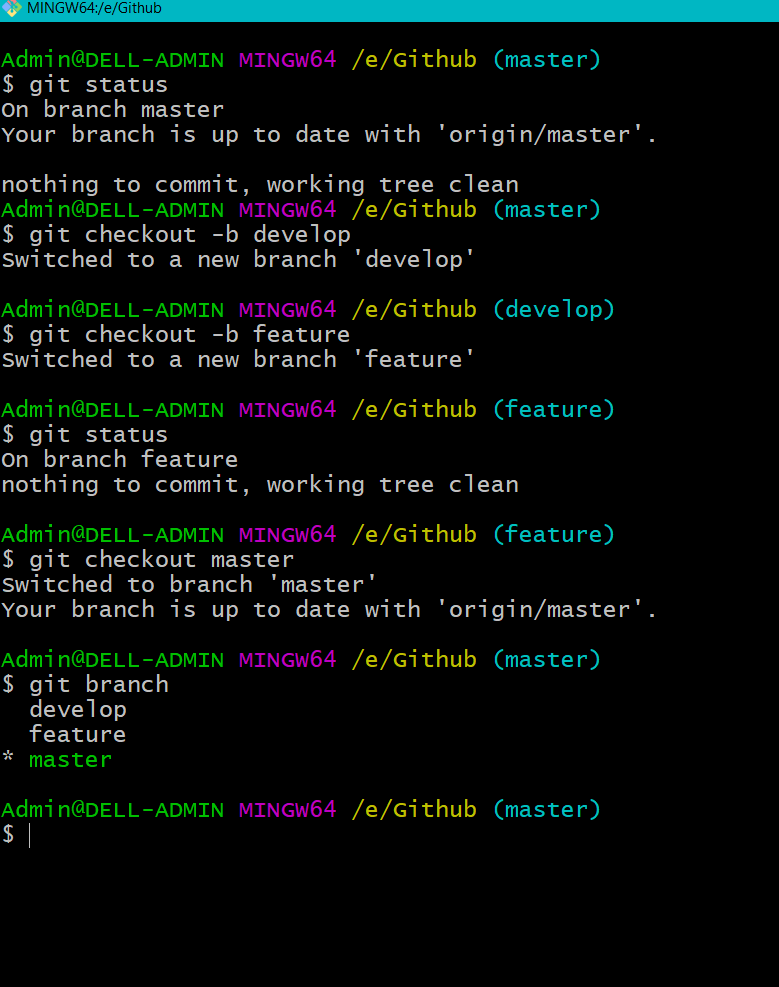


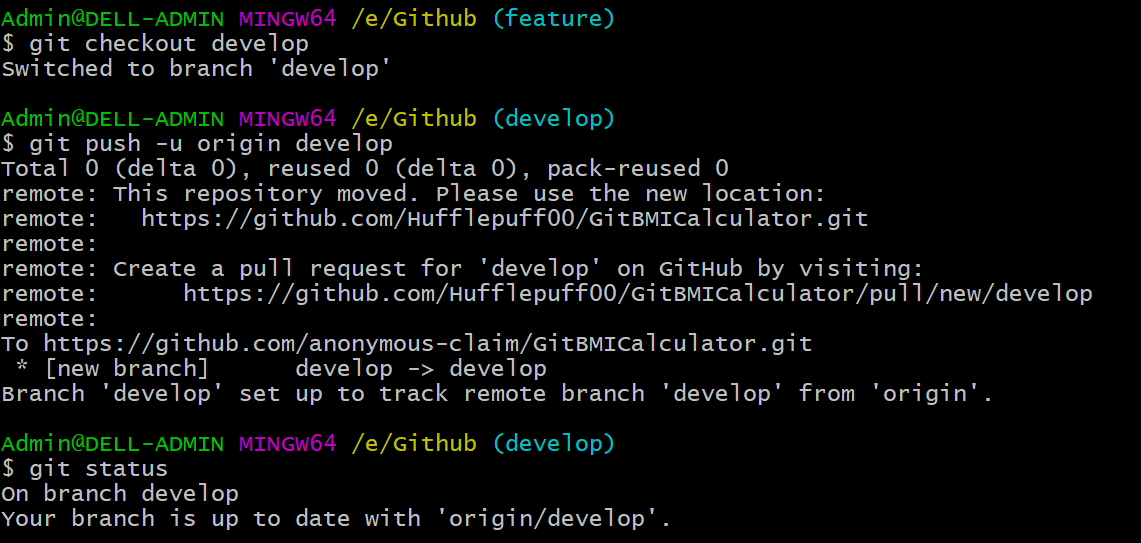
1. **Creating new branches (Develop, Feature)**



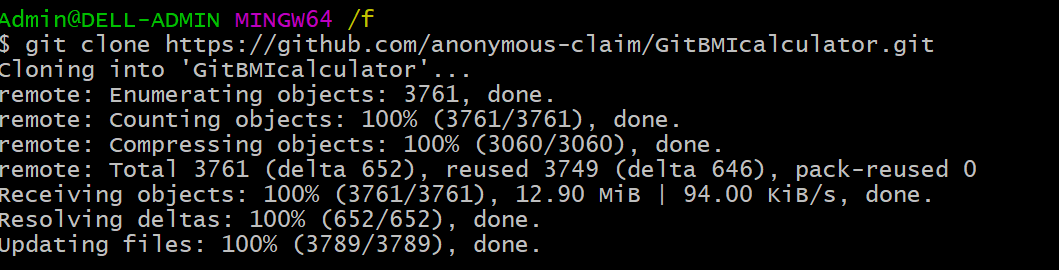
1. **Creation of branches in git:**

*git checkout {branch name}* - Used to make branches for existing repo

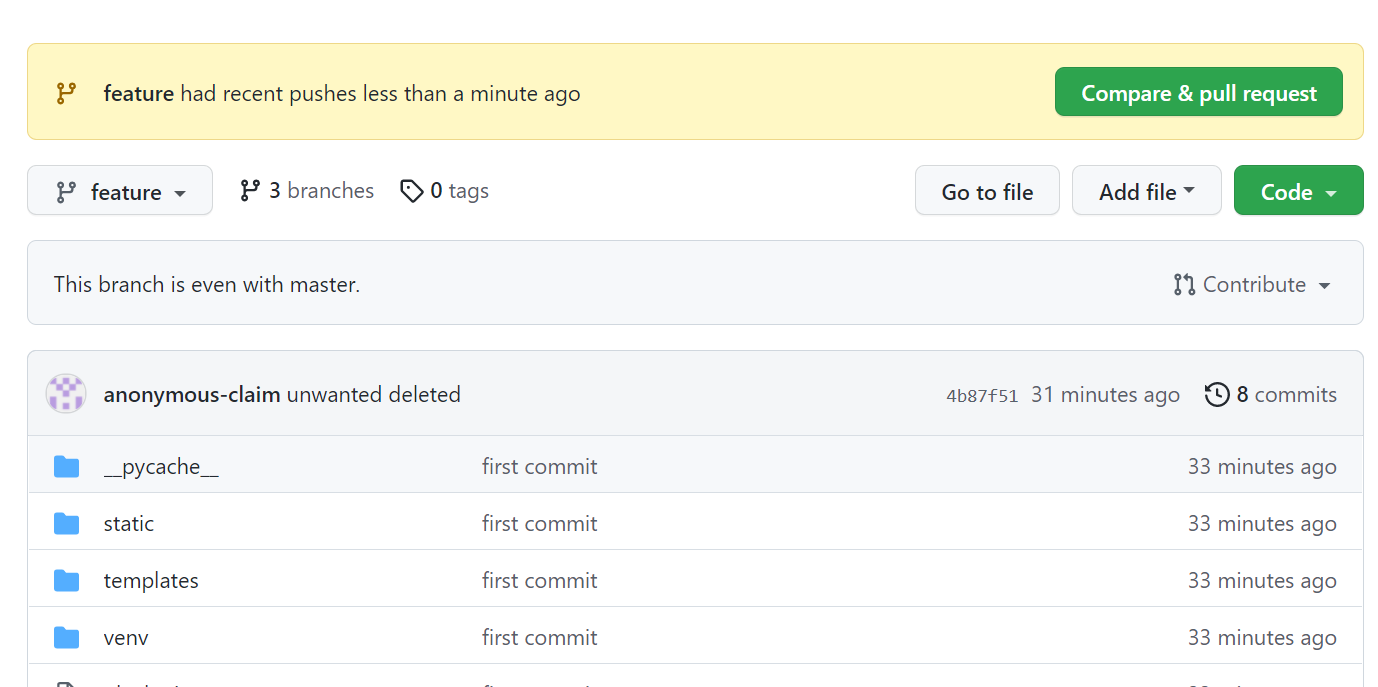


1. **Collaborator cloning repository**

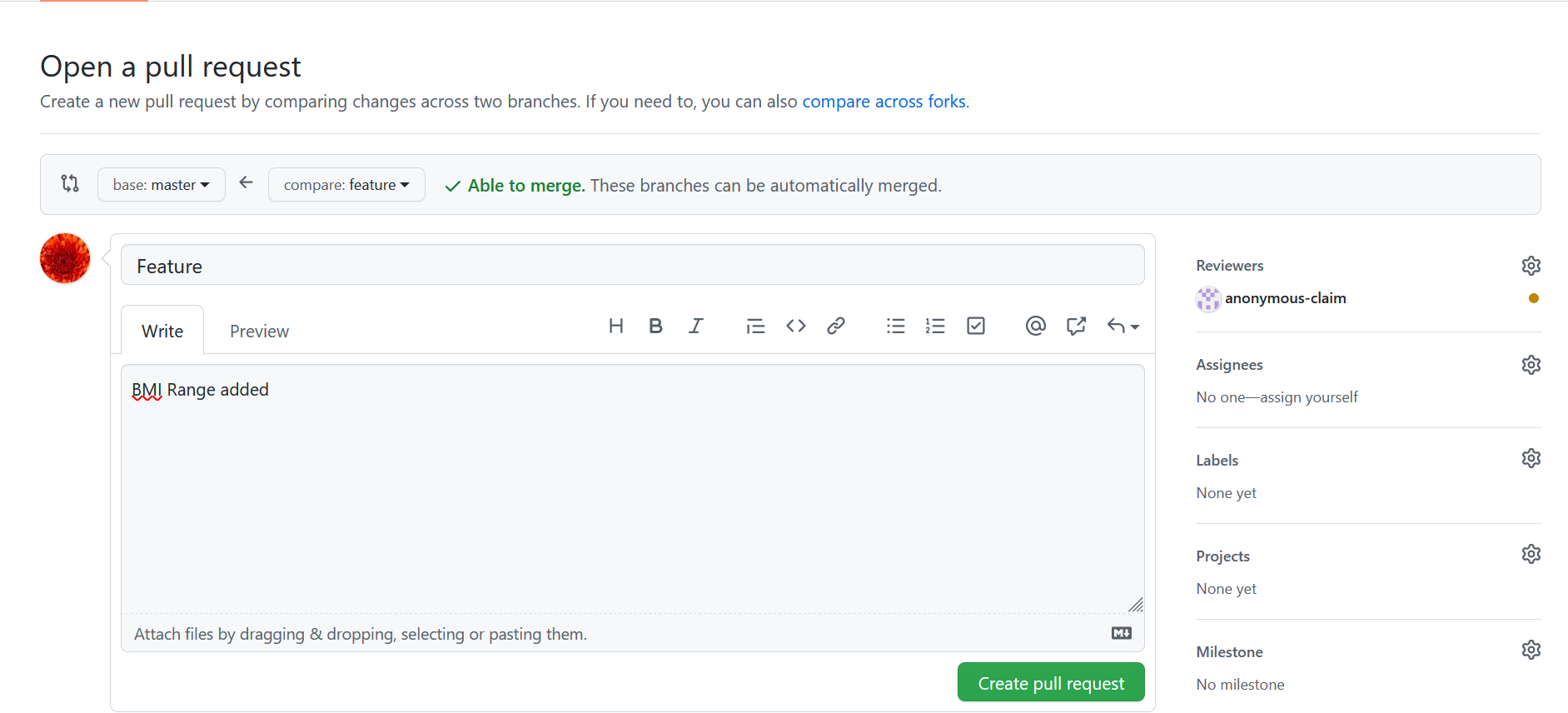


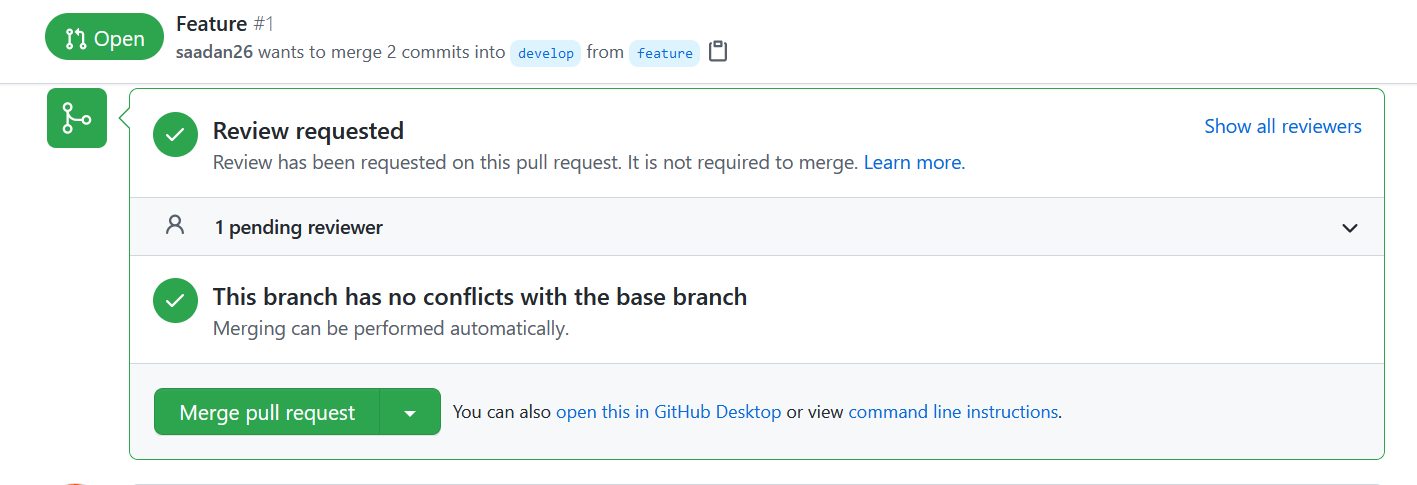
Collaborator can make changes to the repository locally in feature branch. The changes has to be pushed to repo by committing the changes and pushing them.

1. **Changes pushed to repo by collaborator**

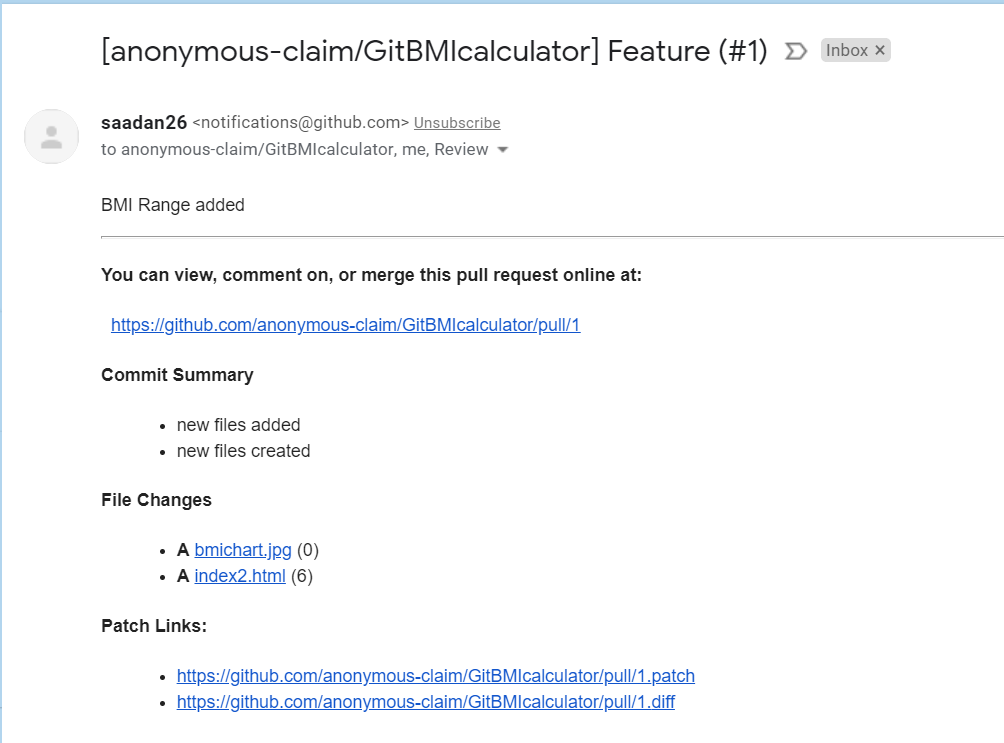


1. **Pull Request is created and assigned to Manager for reviewing**



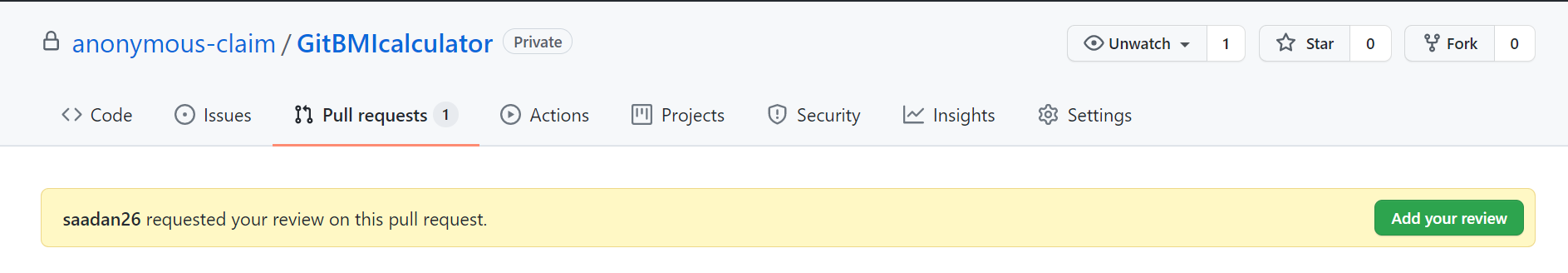


1. **Once the Pull request is created, Reviewer will be notified with an email**.

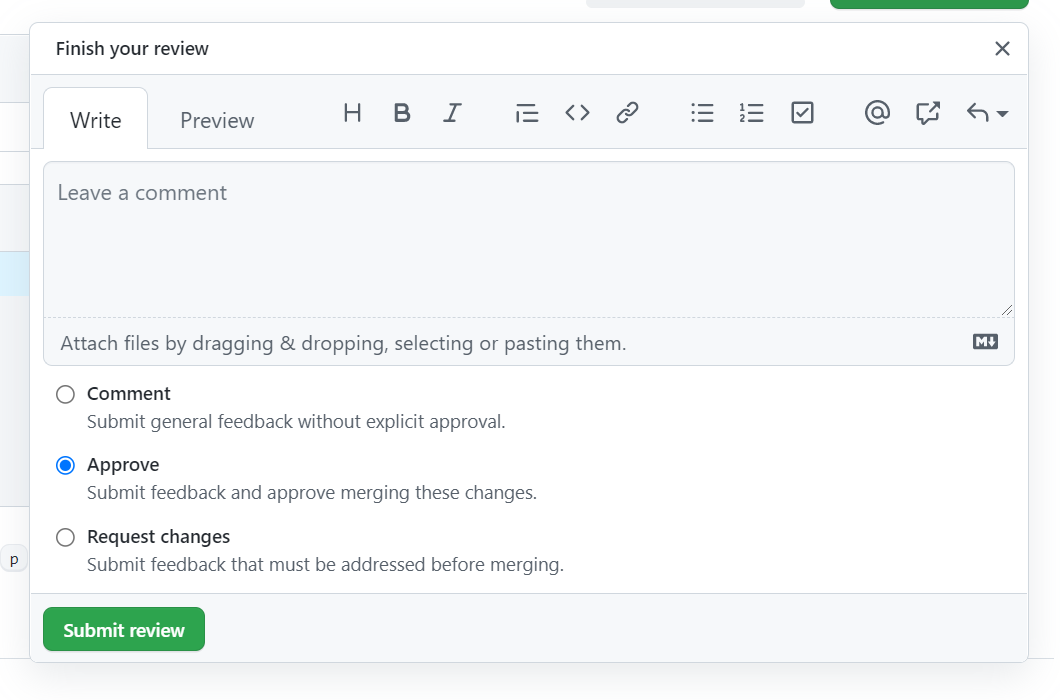


Manager will see the new changes made, can do any one of the following: -

1. Add review by adding comments in between the changes
2. Approve the new feature added
3. Request to make changes

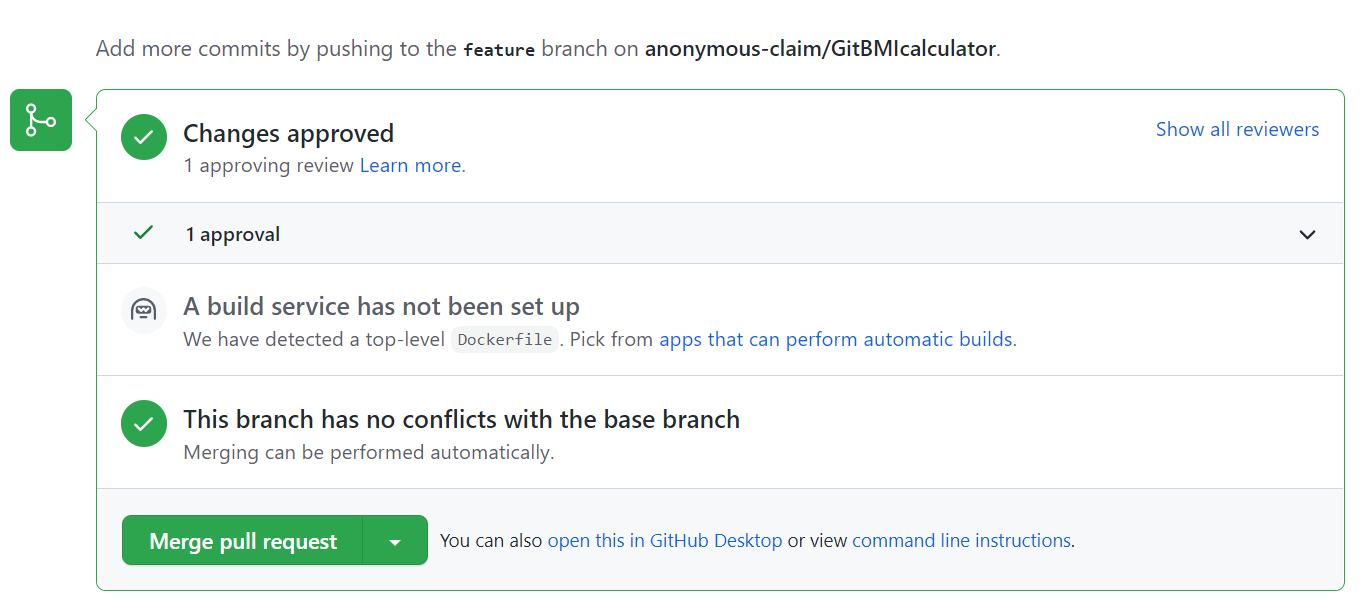


Click **Add your review**, Select the appropriate options also leave a comment if it is required. Click **Submit review**.

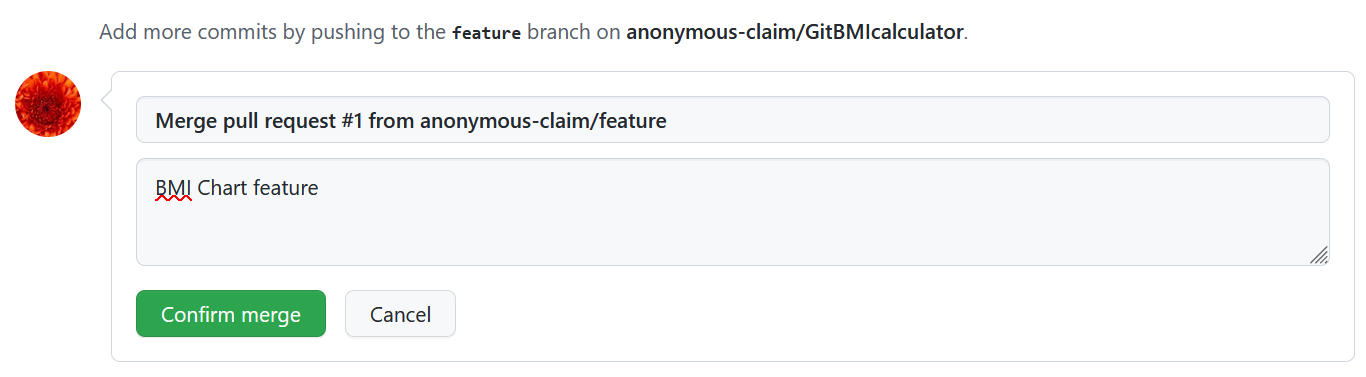


1. **Pull request(PR)** is approved. Now Pull request editor can **Merge pull request** to add feature.

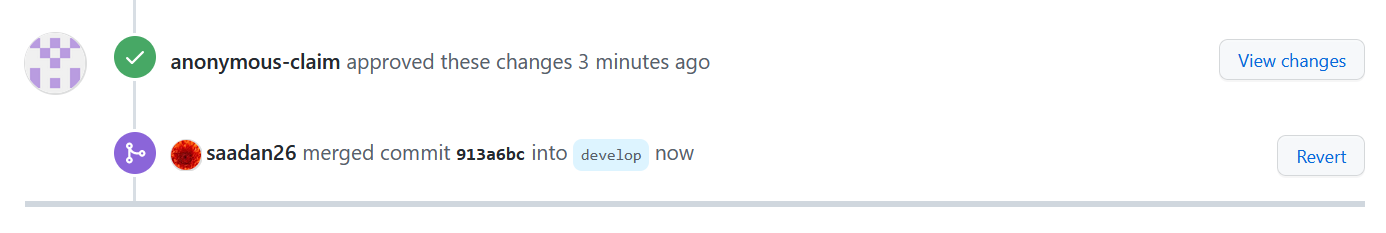
* Merge pull request will merge the approved changes or feature added.



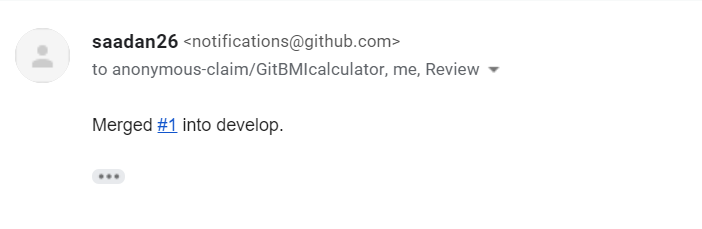
* Collaborator is the PR editor; Changes are pushed to “Develop branch” from “feature branch”.



* Changes are merged into develop branch

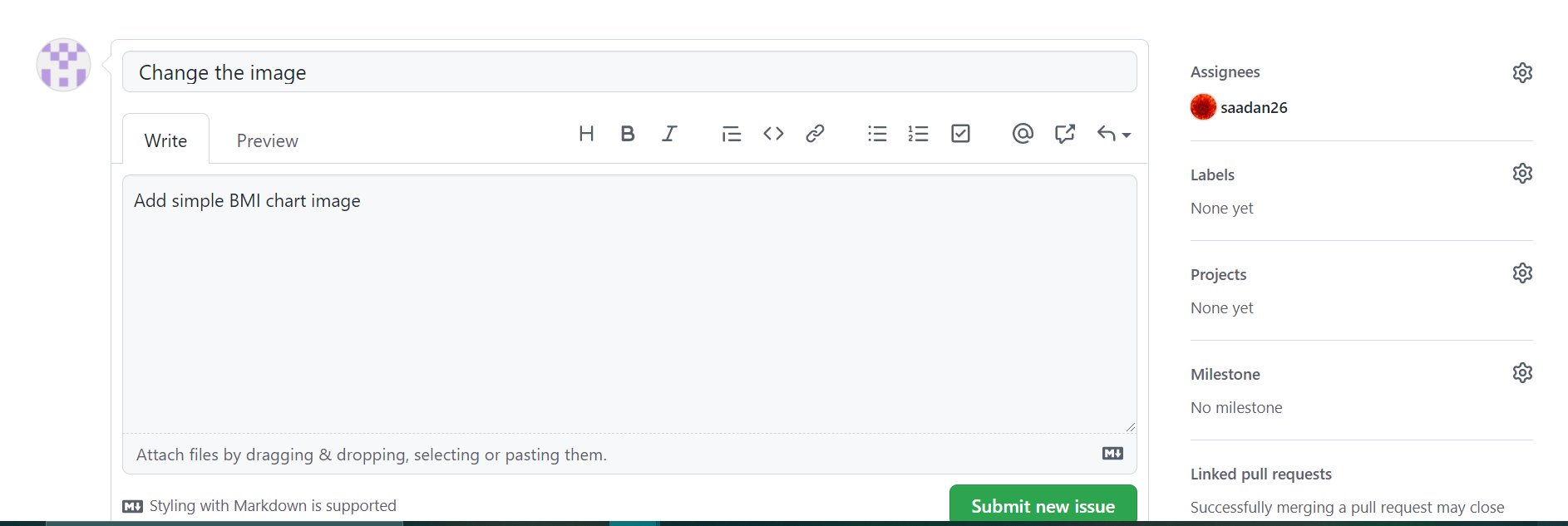


* Email notified on success of merge request.

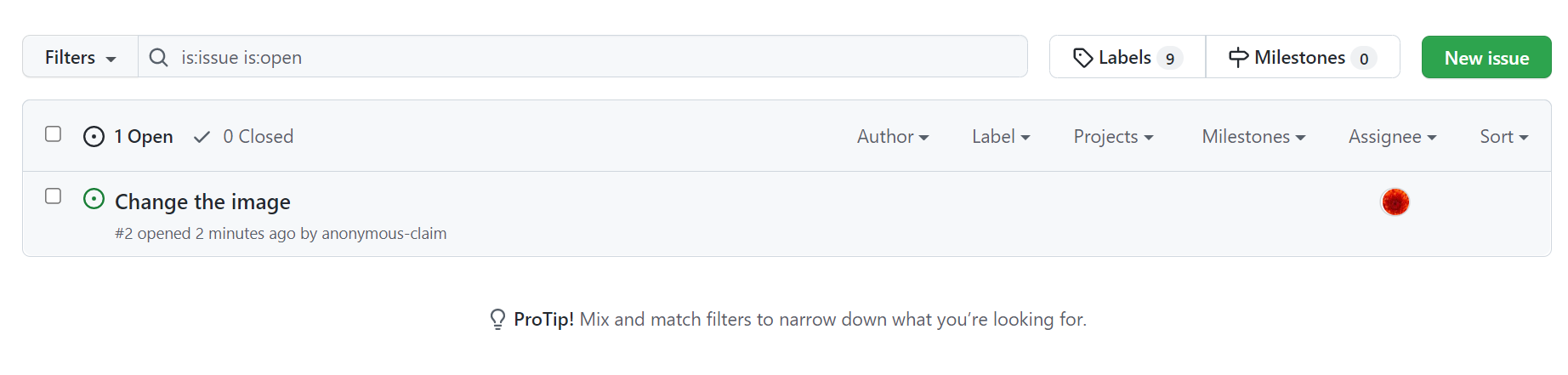


1. **To create an issue in the repo,**

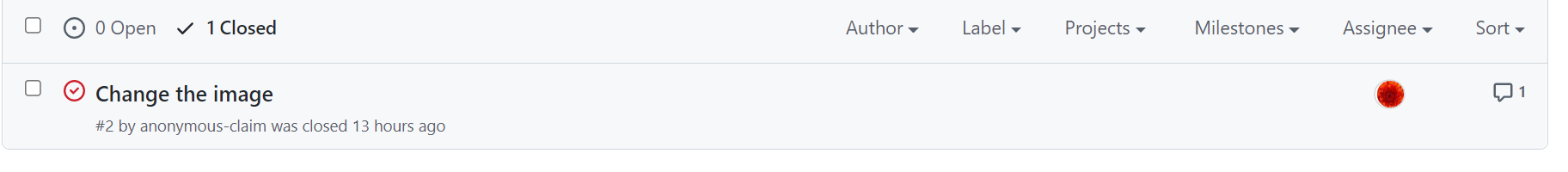
* Navigate to the repository page.
* Click **issue** tab and create new issue.
* Issue can be assigned and taken care of.



* Here, Open issue is created and assigned to collaborator

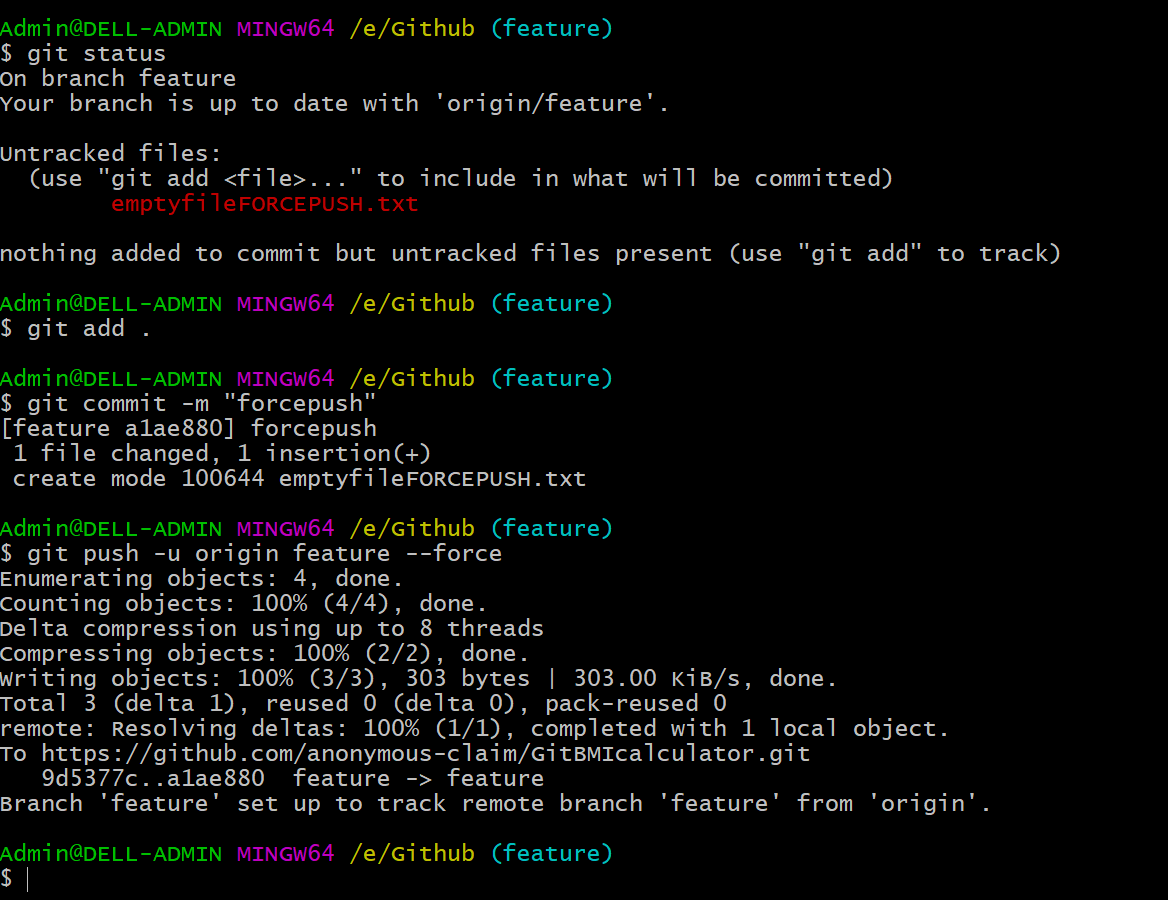


* To **Close an issue,** "Close Issue" button **is to the left of the "comment" button under the comment box**



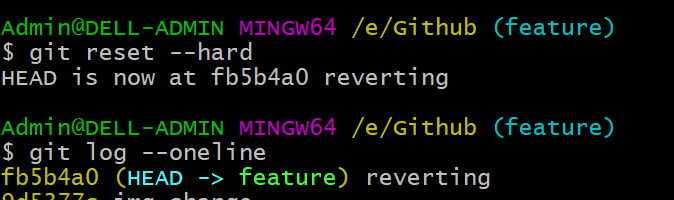
1. **Do a force push/commit from locally**

--force flag is used for force commit/push to repo

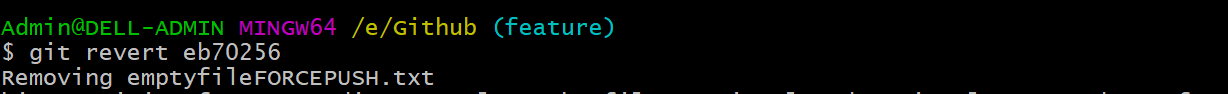


To reset the changes made, using this command

*git reset –hard {hash id}*

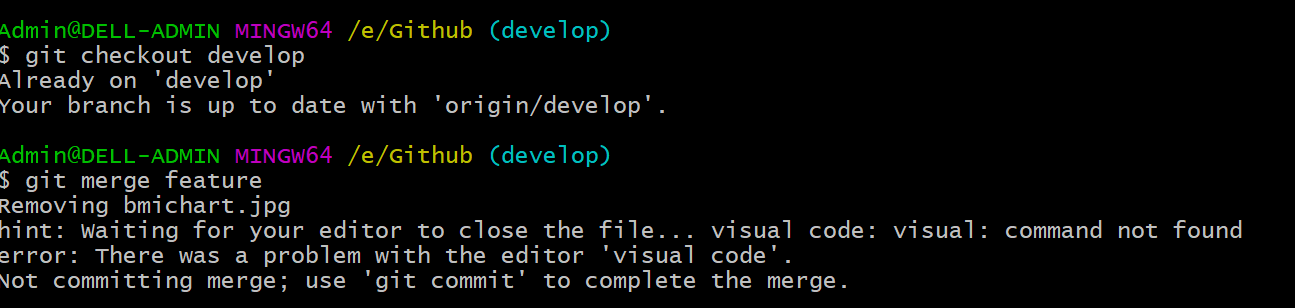


git revert {hashid} - this removes the previously pushed changes to repo

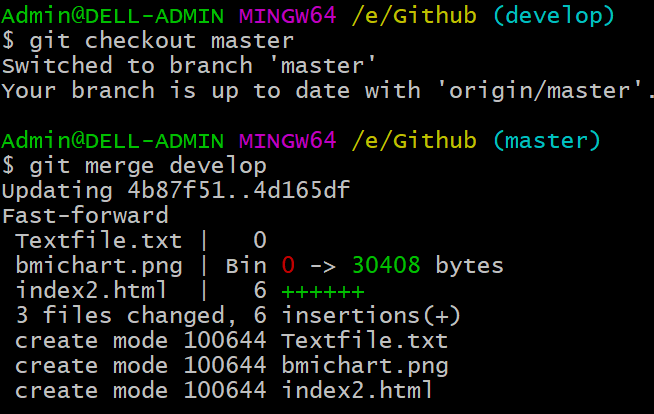


1. **Staging “development branch to production branch”**

You create a feature branch complete the feature, and then merge back into development. This can then be added to the final production version by merging into production

****

1. **Merge development branch to production branch**

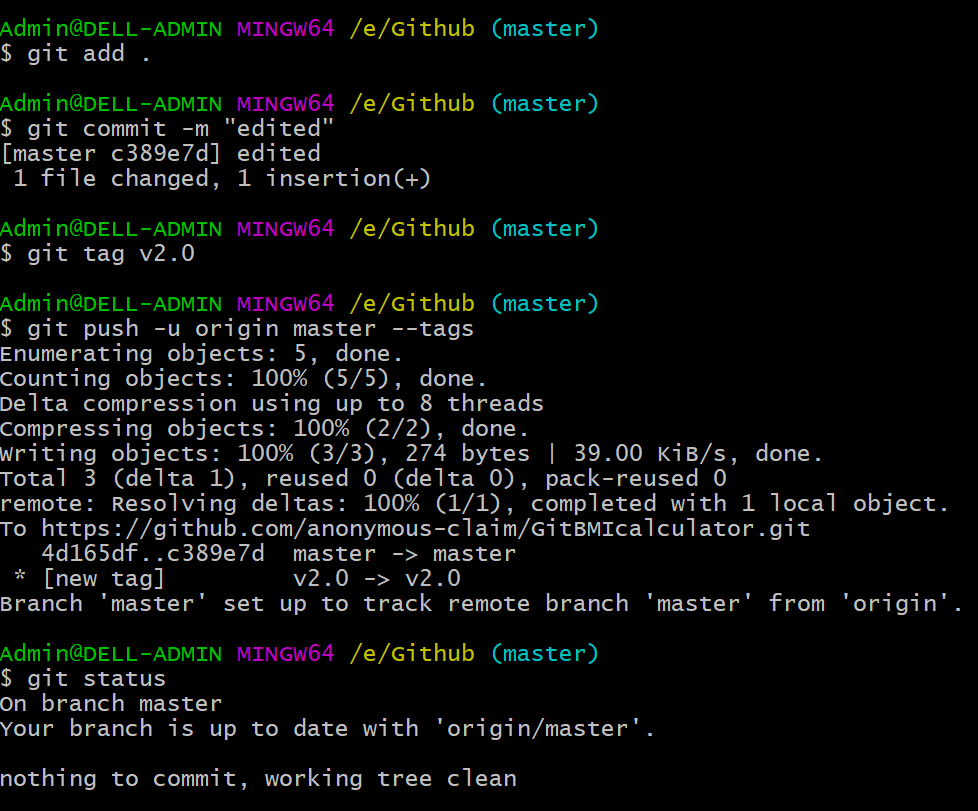
****

1. **Version release using tags**

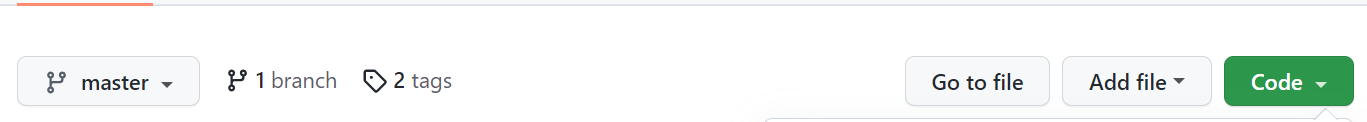
*git tag v2.0*

To push tags in remote repo, using the following command

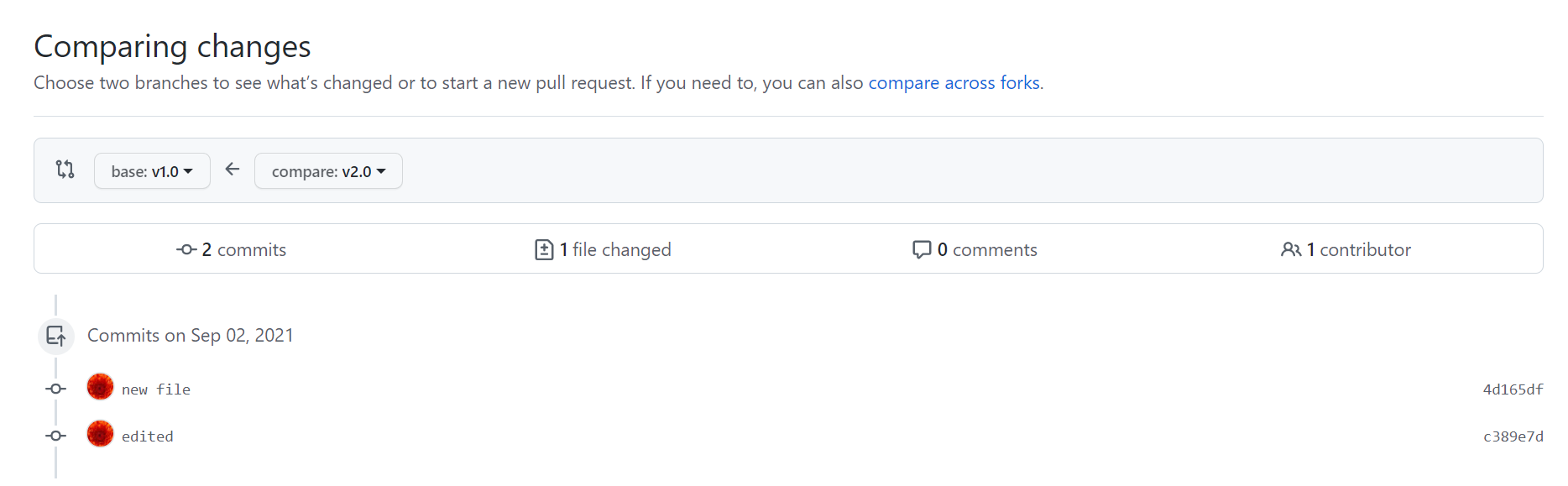
*git push -u origin master --tags*



* After tagging



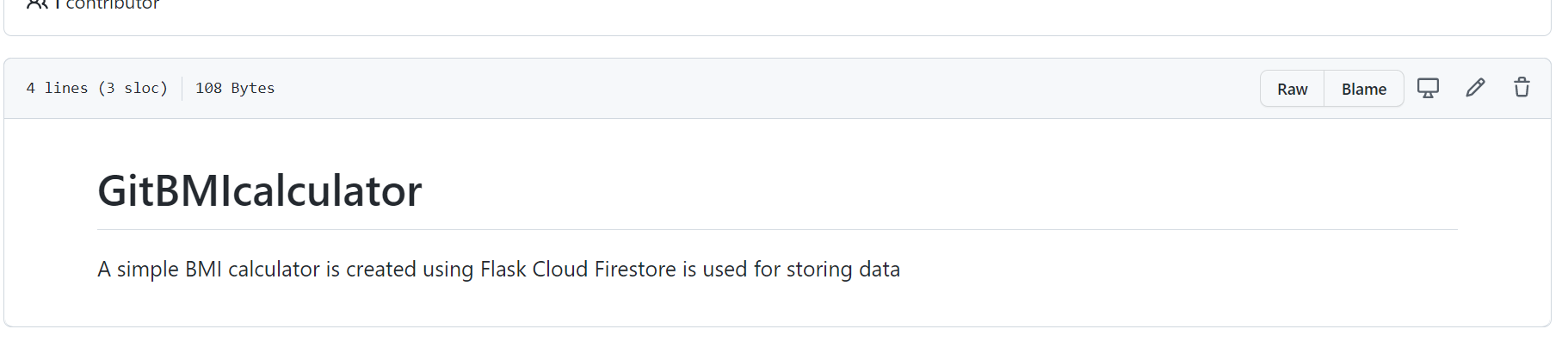
* Comparing the tags, the changes made can be viewed.



1. **README file**

Readme file is an introduction to the files contained in the repository. It contains the following

1. Installation instructions
2. Pointers to the project’s website
3. Information for the user about the software
4. Details about patches or updates.
5. Bug tracker
6. Little about the author



1. **.gitignore**

gitignore is an auto-generated file inside the project folder that ignores/prevents files to get committed to the local and remote repositories. Files that should be ignored are added in the .gitignore are :-

* Log files
* Files with API keys/secrets, credentials, or sensitive information
* Dependencies which can be downloaded from a package manager

