# Version History

|  |  |  |
| --- | --- | --- |
| Version | Date | Release Note |
| V1.0 | 10-Mar-2020 | Initial Draft |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

index

1 Version History 2

2 Exercise 1 4

2.1 Create new repository 4

2.2 Chekcout from central repository 8

2.3 Daily Operations 11

2.4 Merge conflict 11

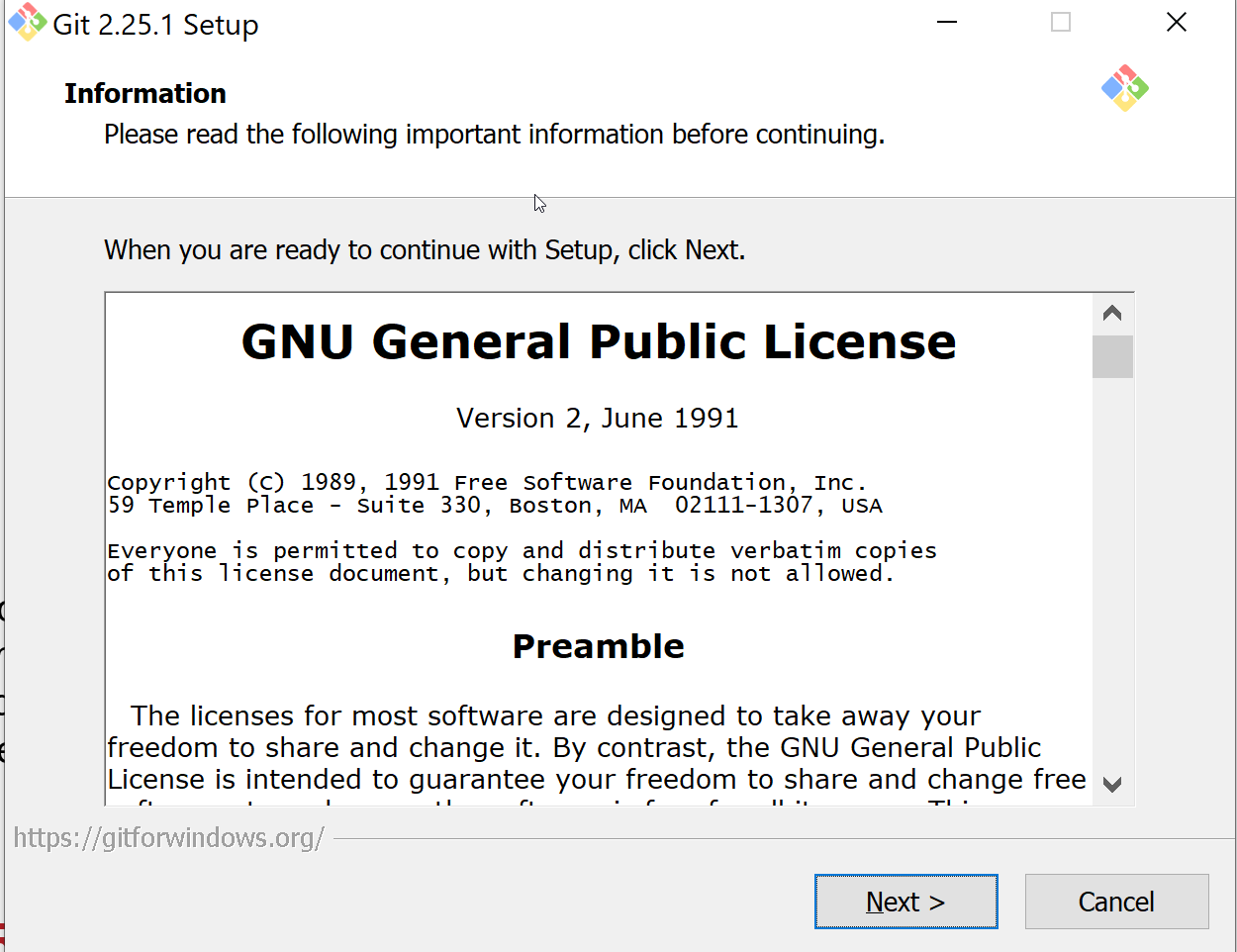
2.5 Raise pull request 11

2.6 Merge Branch 11

# Exercise 1

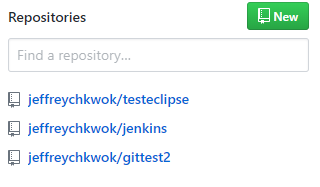
The exercise will guide you how to perform source code management through the git

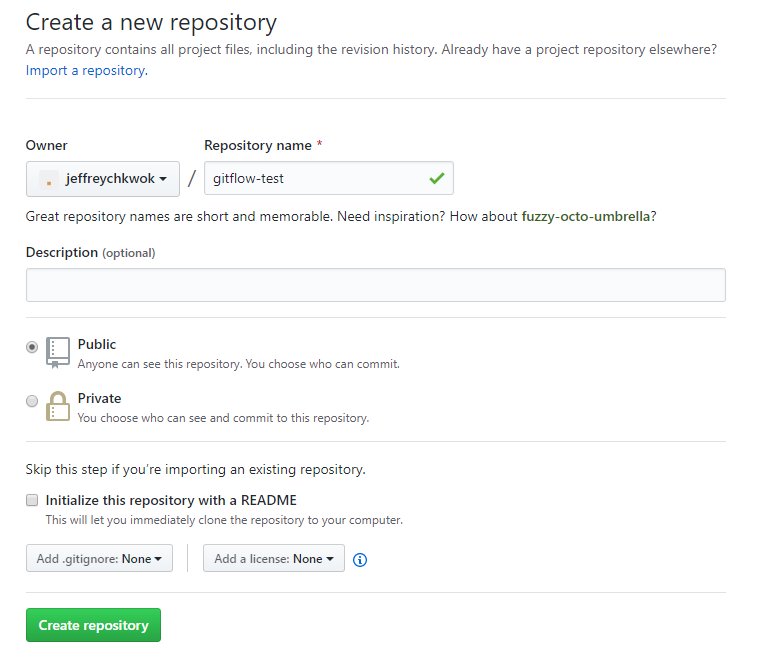
Prerequisite

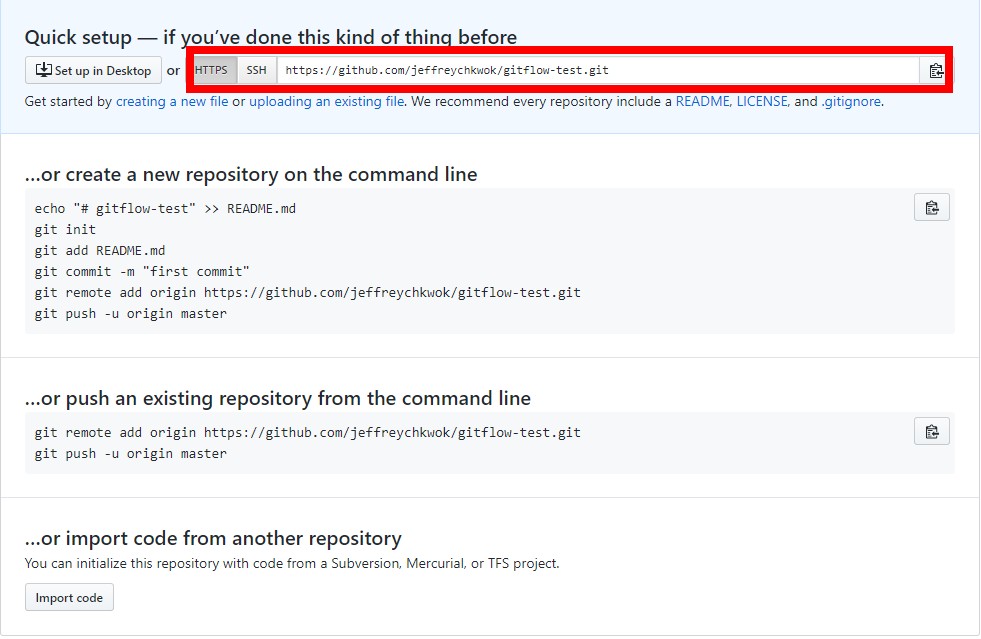
1. Register a new account in <https://github.com/>
2. Download a git command and install it
   1. Download git command from <https://git-scm.com/downloads>
   2. Follow the instructions to install the Git with default options

## Create new repository

### Create new respository in GitHub

1. Click **New** to new a repository
2. Enter repository name gitflow-test and click Create respository



1. ****The http/ssh url in the result screen is the remote url we used to access the git repository
2. On your local machine, you can create a new directory to maintain your git repository

mkdir c:\local-git

cd c:\local-git

1. Follow the steps Create a new repository on the command line

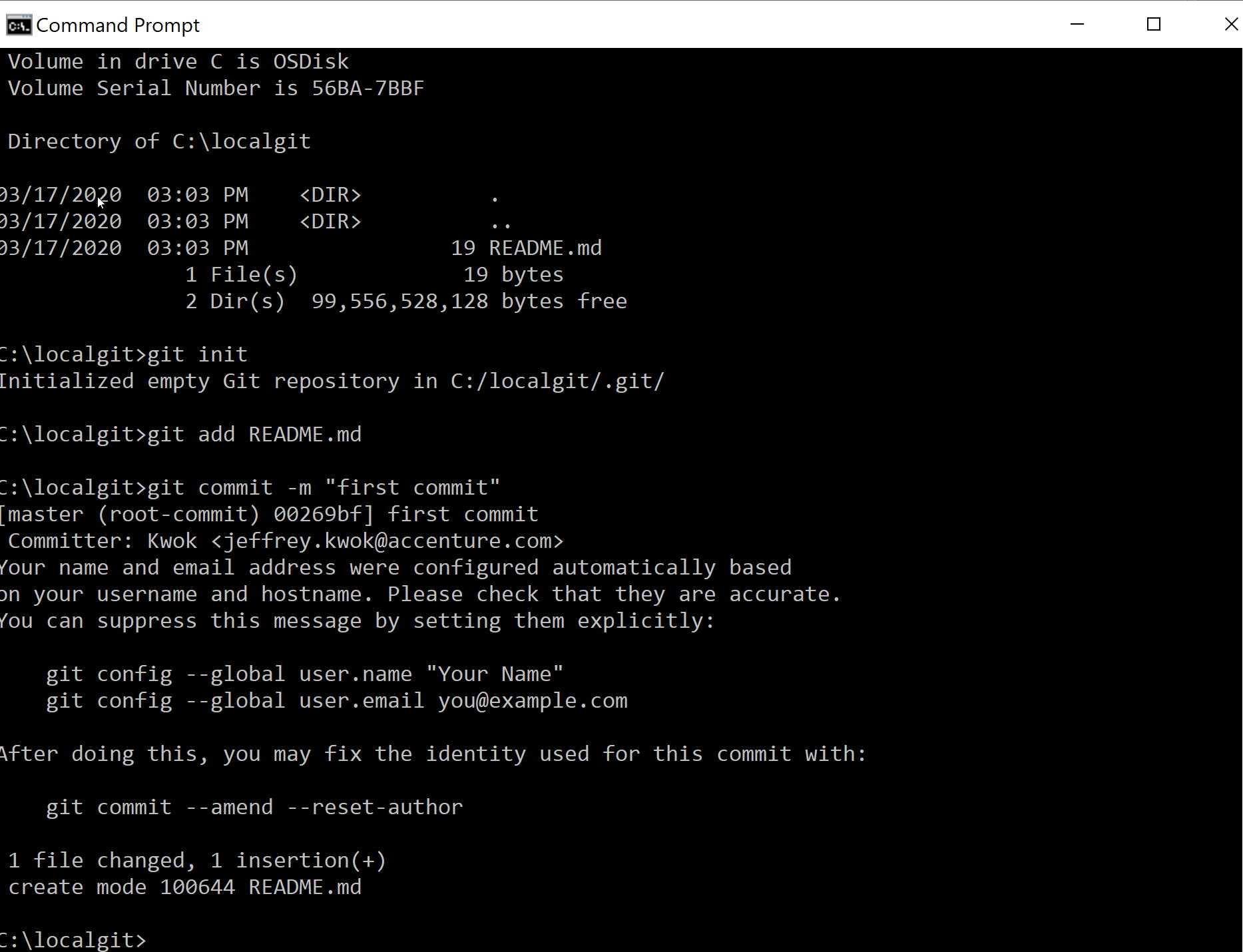
echo "# gitflow-test" >> README.md

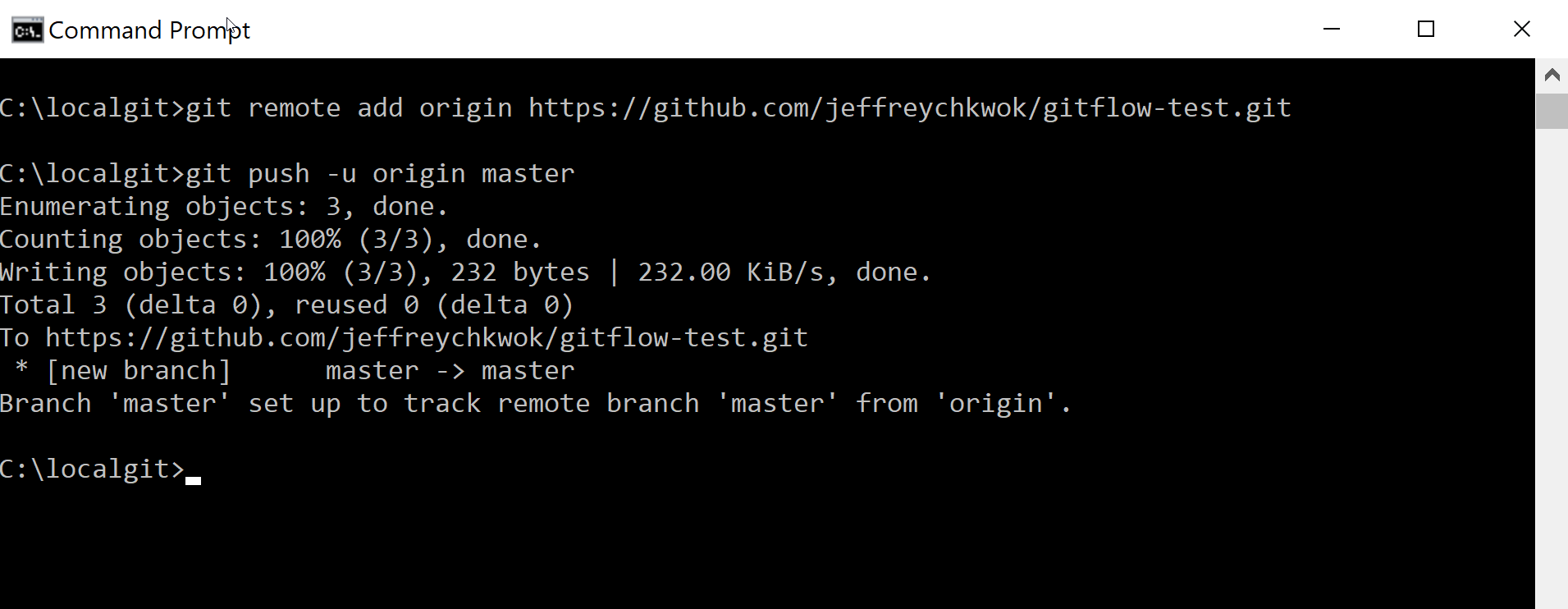
git init

git add README.md

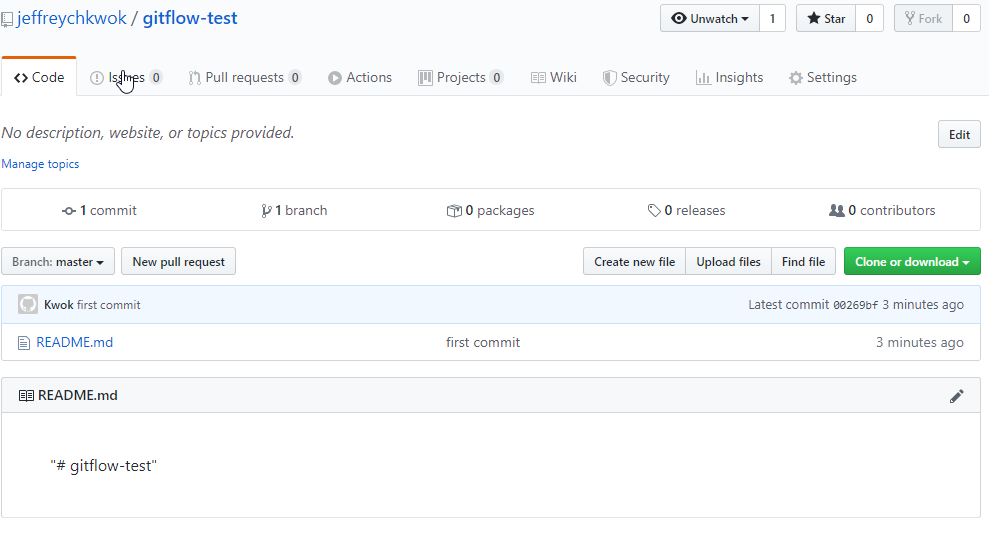
git commit -m "first commit"

git remote add origin <your git url>

git push -u origin master

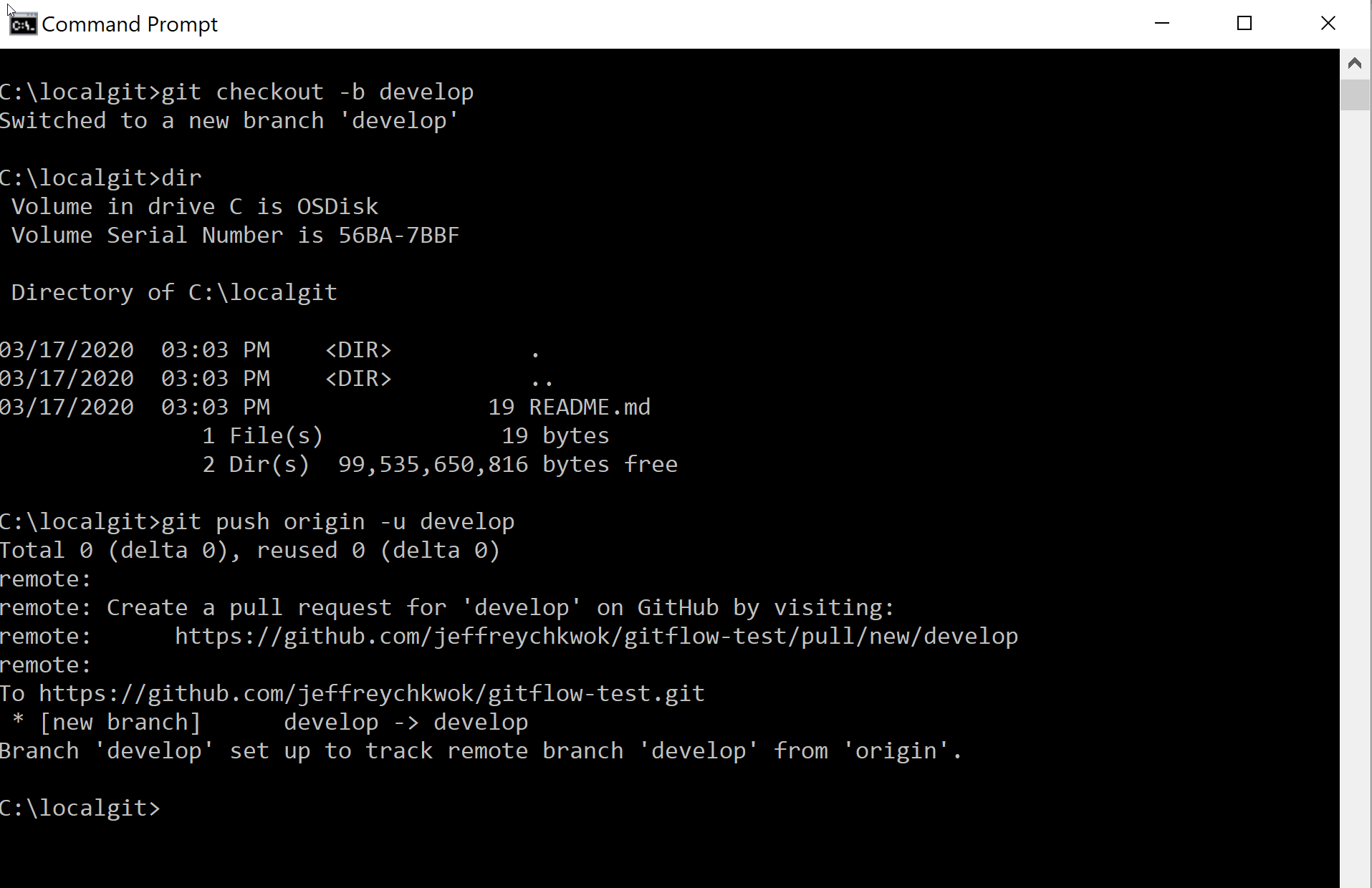
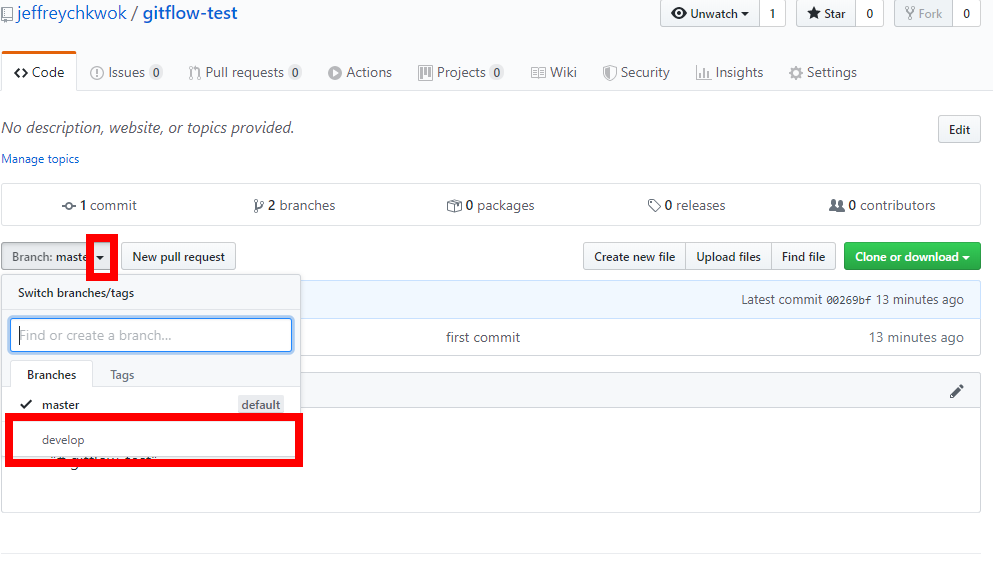


1. Browse your Github page and refresh it, your files and repository should be there

A master branch is created already.

### Create develop branch

Apart from the master branch, we should create develop branch also

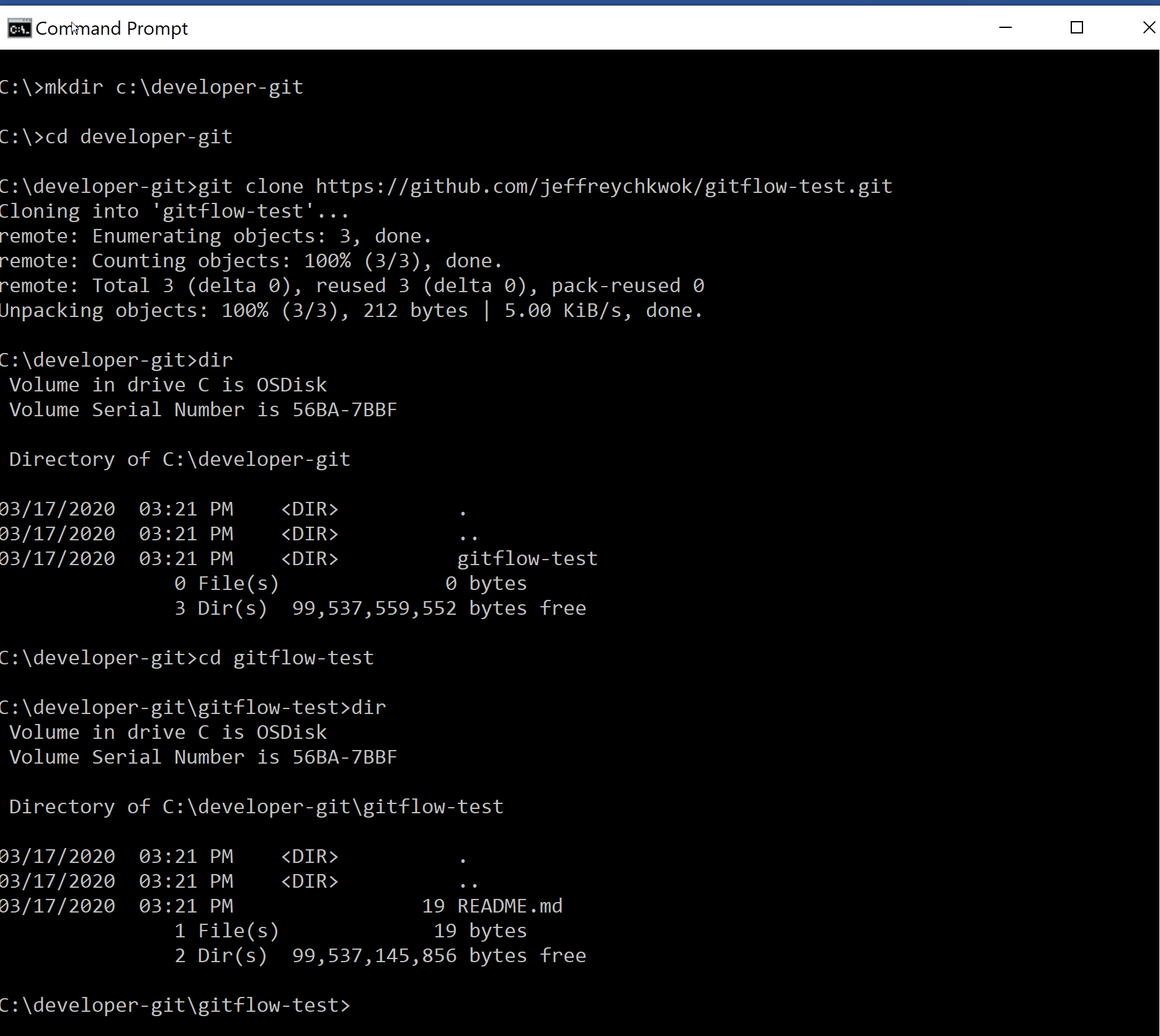
1. In your command pompt, enter git checkout -b develop and git push -u origin develop
2. Refresh your Github page and you should found your develop branch
3. You can click **develop** to switch to view develop branch files

## Chekcout from central repository

### Clone the respository

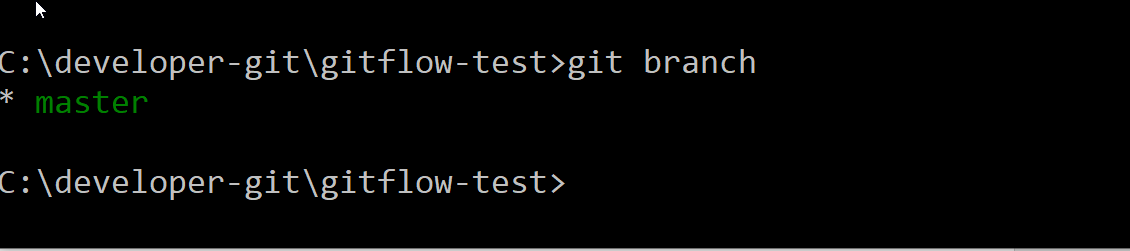
Assume you are another developer now and you want to checkout the code from the code repository. You can clone the repository from Github

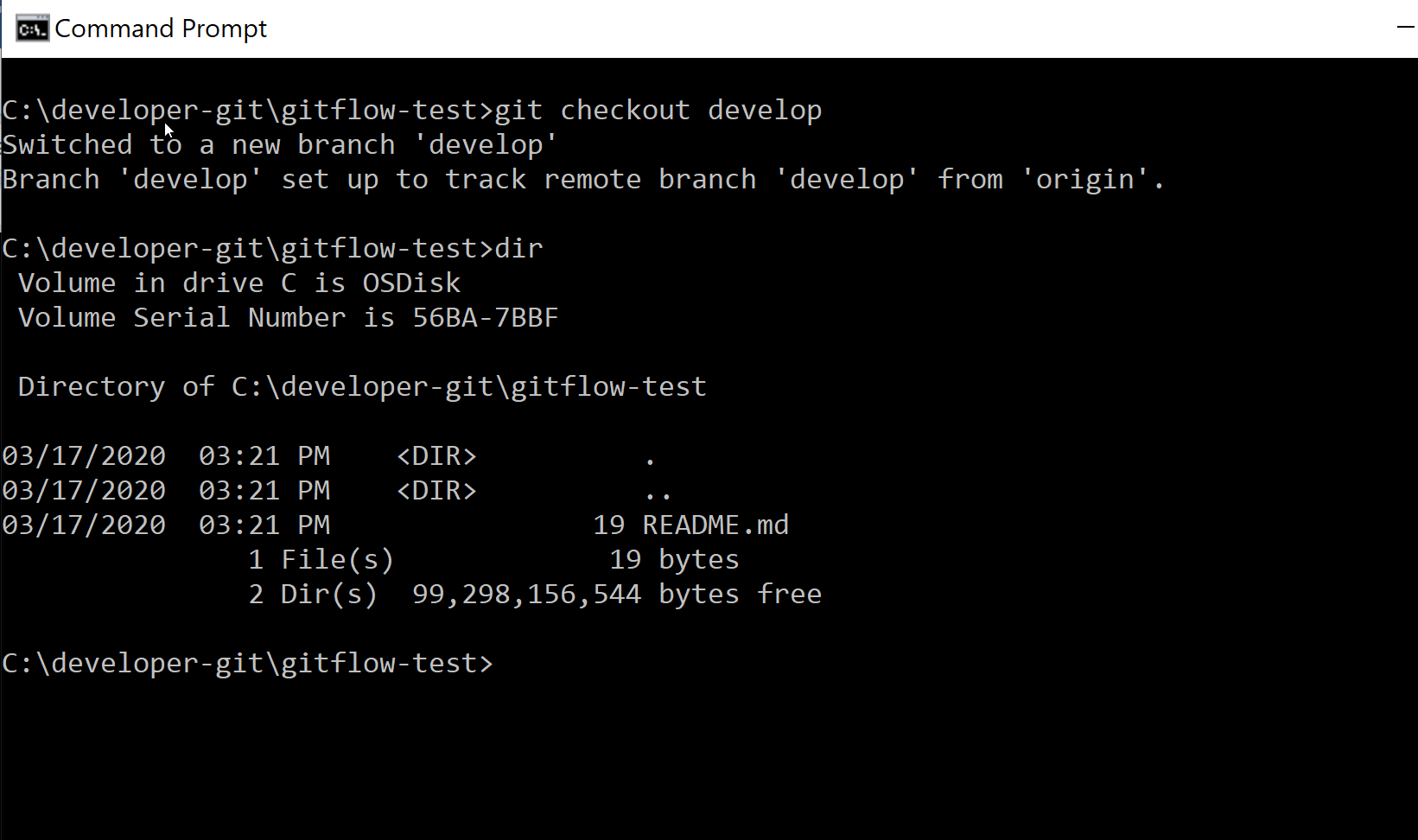
1. Create a new folder and enter **git clone <git url>**



The respository folder will be created automatically. In this folder, you will found your code there

But, what is the branch?

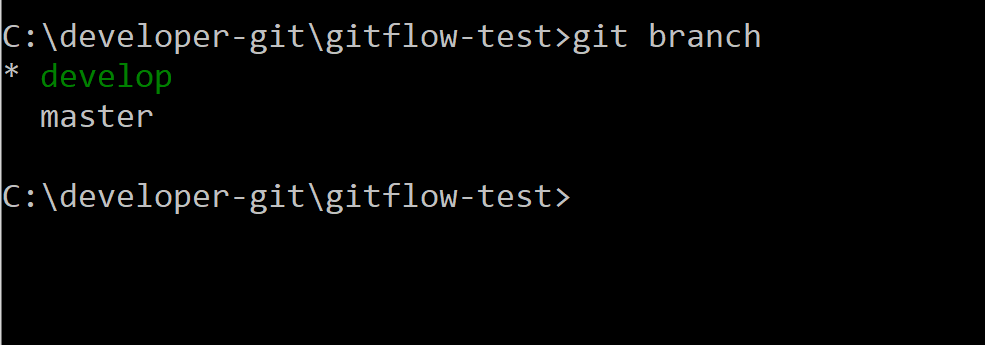
Enter **git branch** to view what is the branch you are viewing

1. Enter git checkout develop

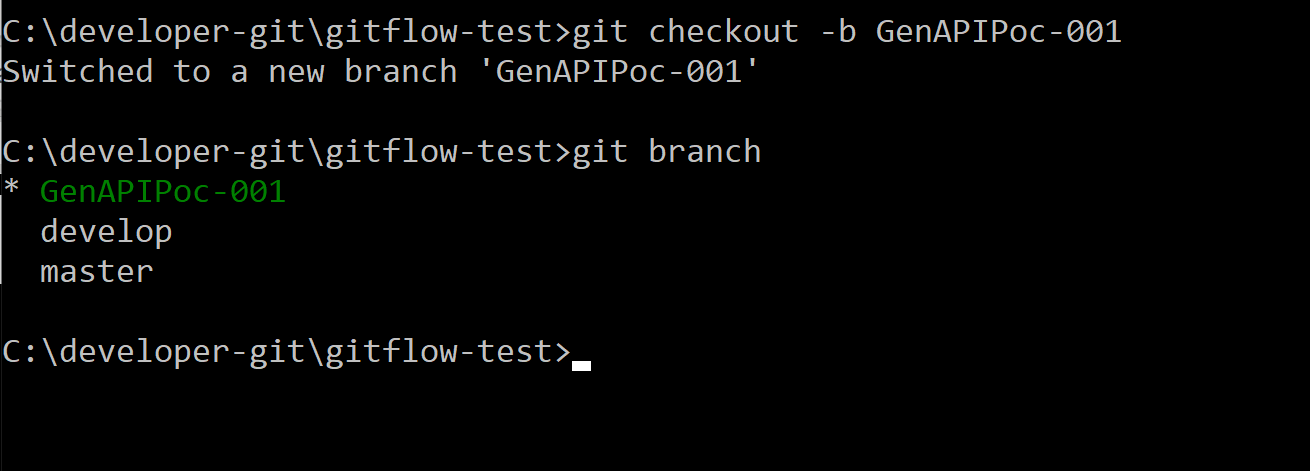
### Create new feature branch

Now, you want to start the development. You needed to create a new feature branch based on the develop branch

1. Make sure you checkout the develop branch successfully and you are in the develop branch

Enter **git branch** to view the branch you are working now

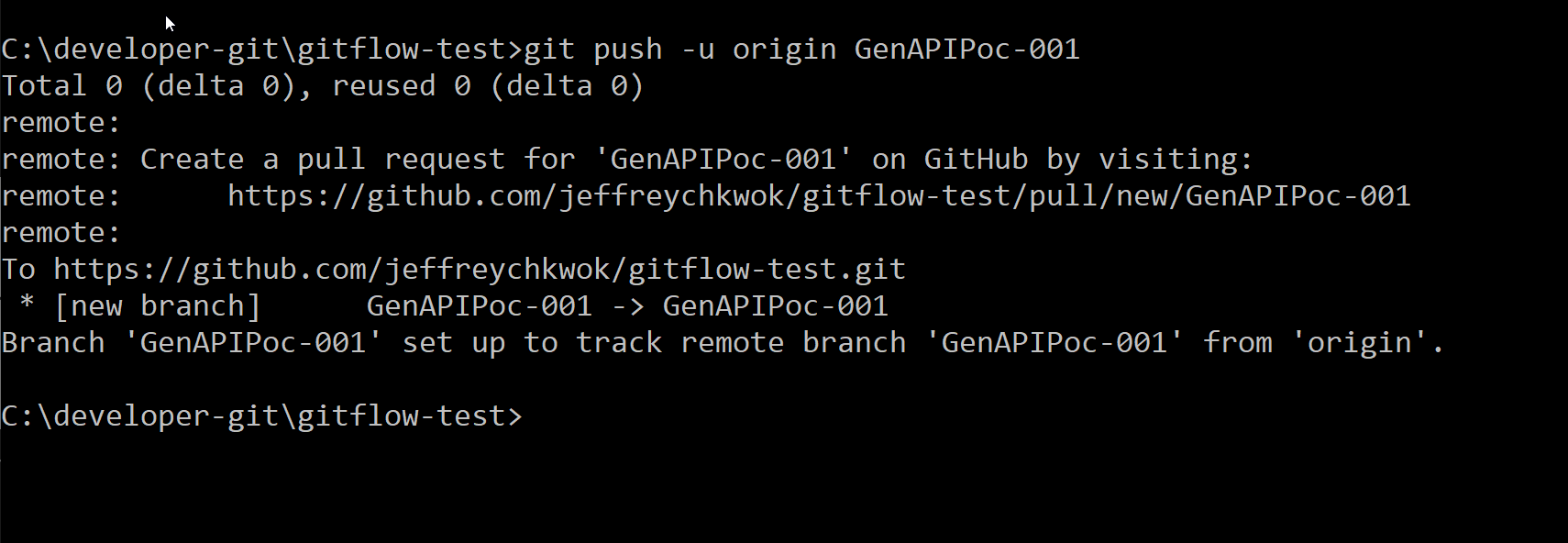
1. The feature branch name should be meaningful and easy to identify. In best practice, we will use the jira number as the branch name. e.g We have a story in Jira with number **GenAPIPoc-001** and we are developing new features for this Jira

Enter git checkout -b GenAPIPoc-001

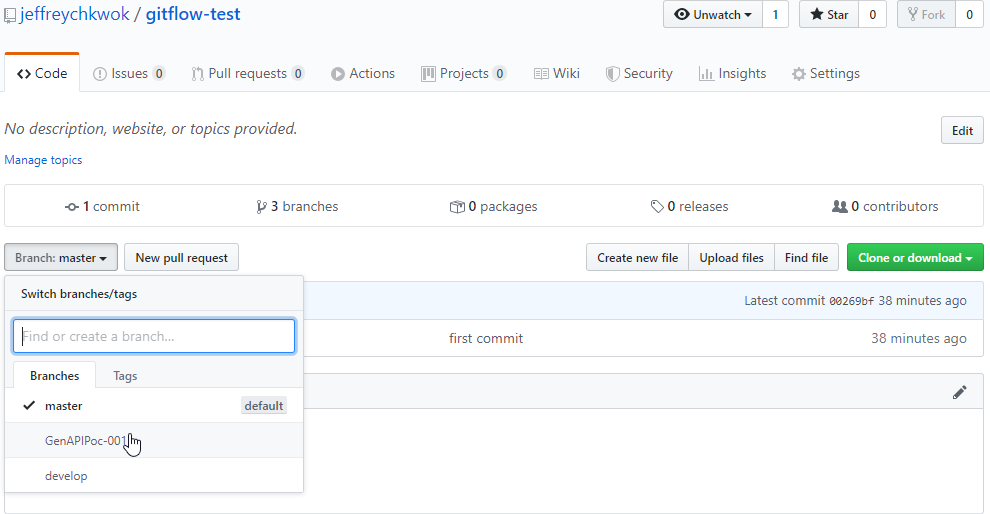
A new branch is created locally and you can start your development tasks now.

\*\*The new feature branch is not yet created in central repository

1. Push the code back to central repository

Enter git push -u origin GenAPIPoc-001

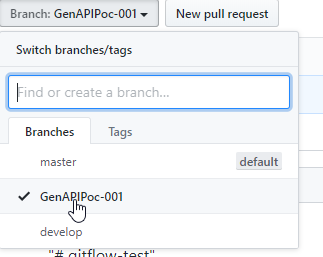
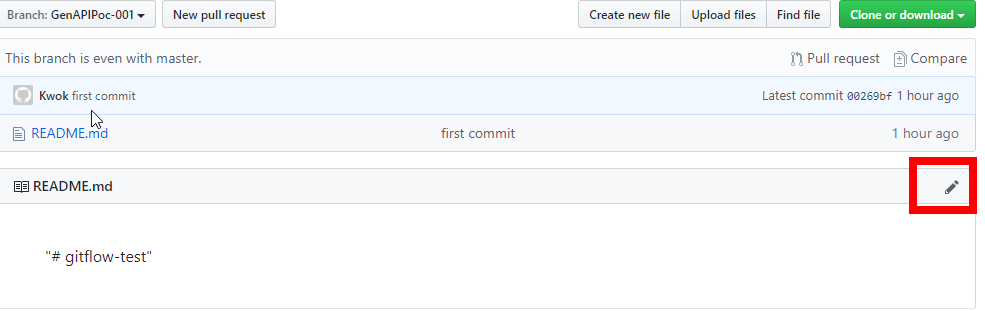
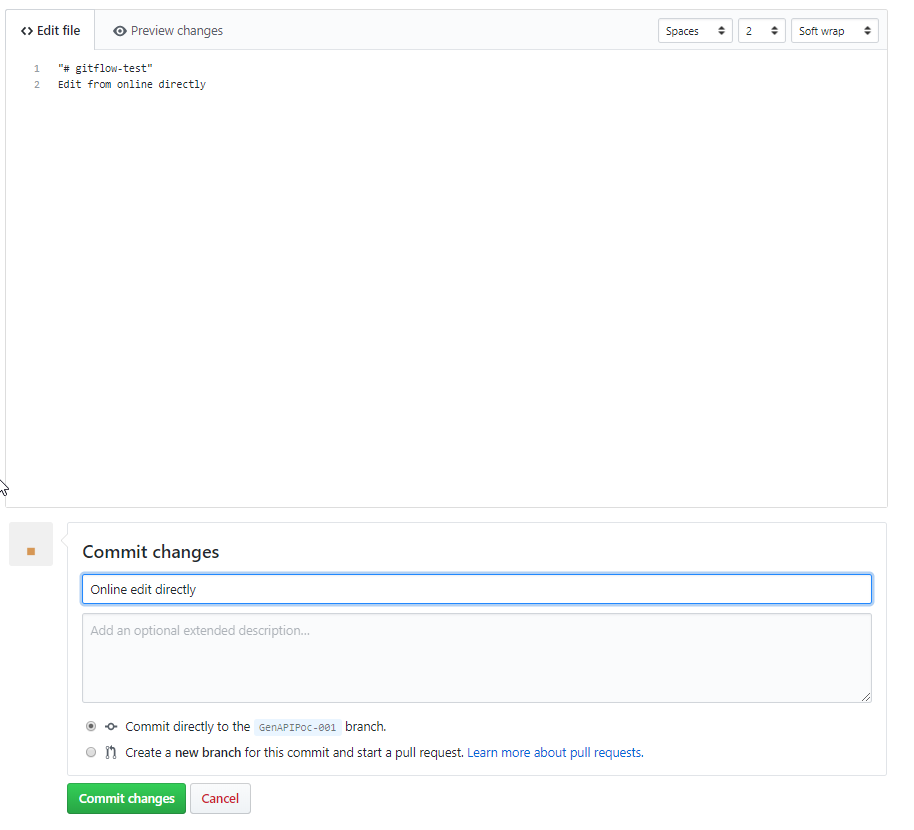
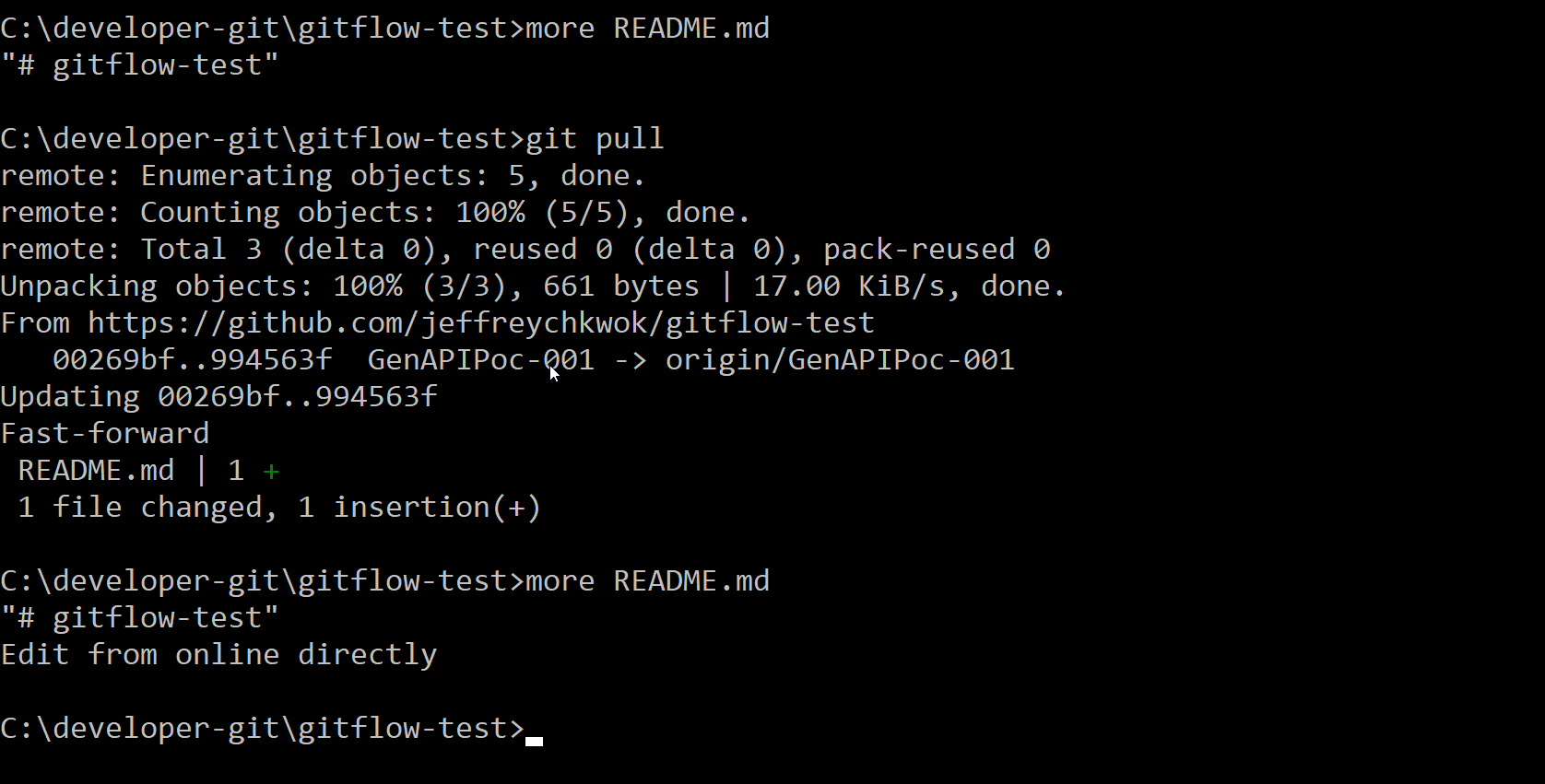
You can view the new branch from GitHub page



## Daily Operations

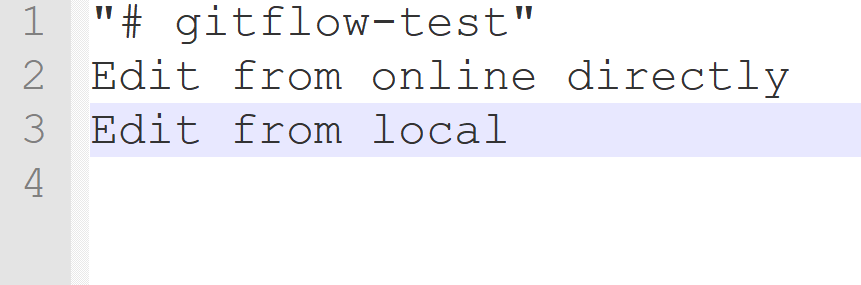
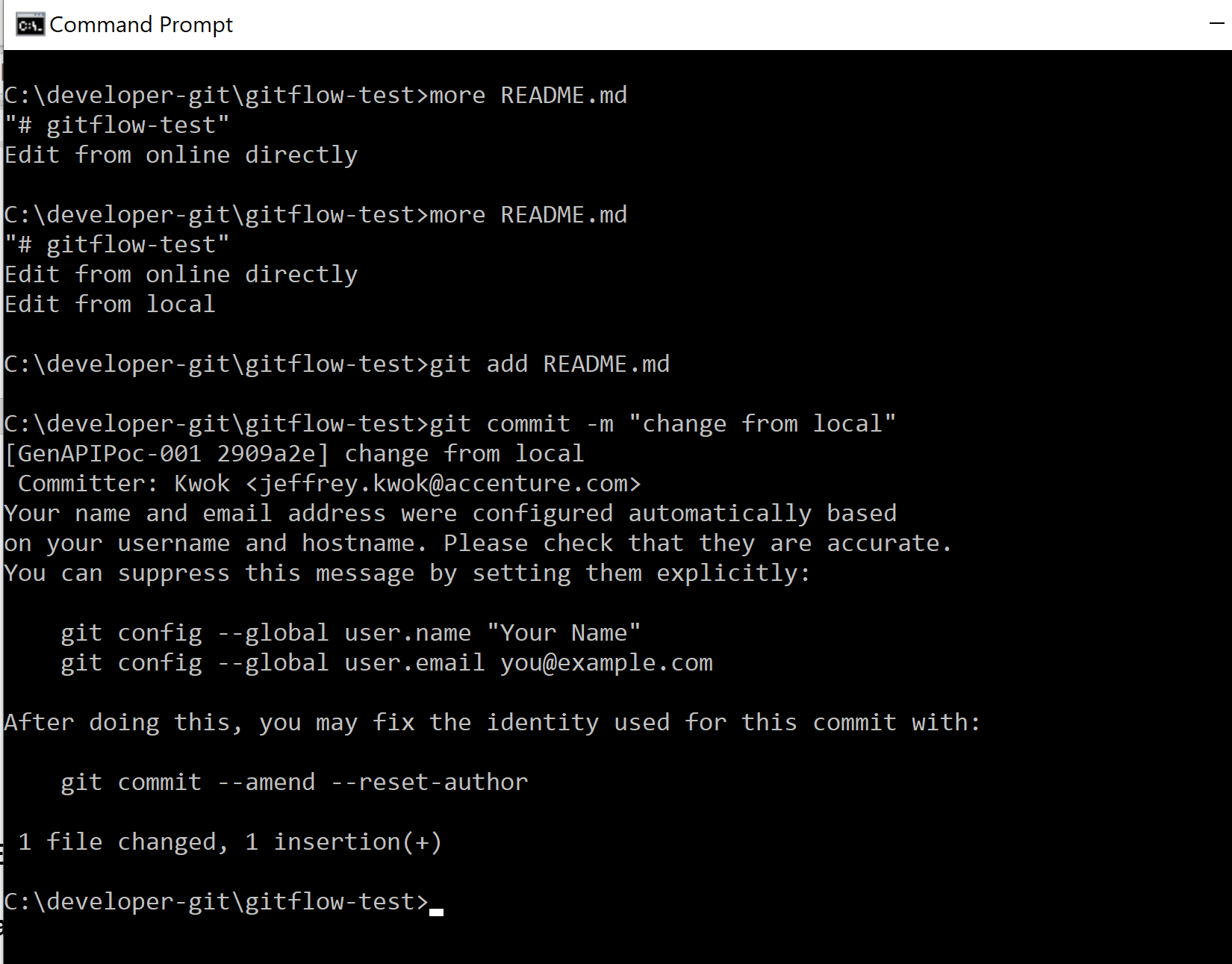
### Pull the code

Developer should pull the code from feature branch regularly to synchronize the code checked in by other developers

1. On the GitHub, switch to the feature branch **GenAPIPoc-001**
2. Click the **Edit** to edit the **README.md** file content
3. Click Commit Changes
4. In local machine, enter **git pull** to get lastest changes from central repository

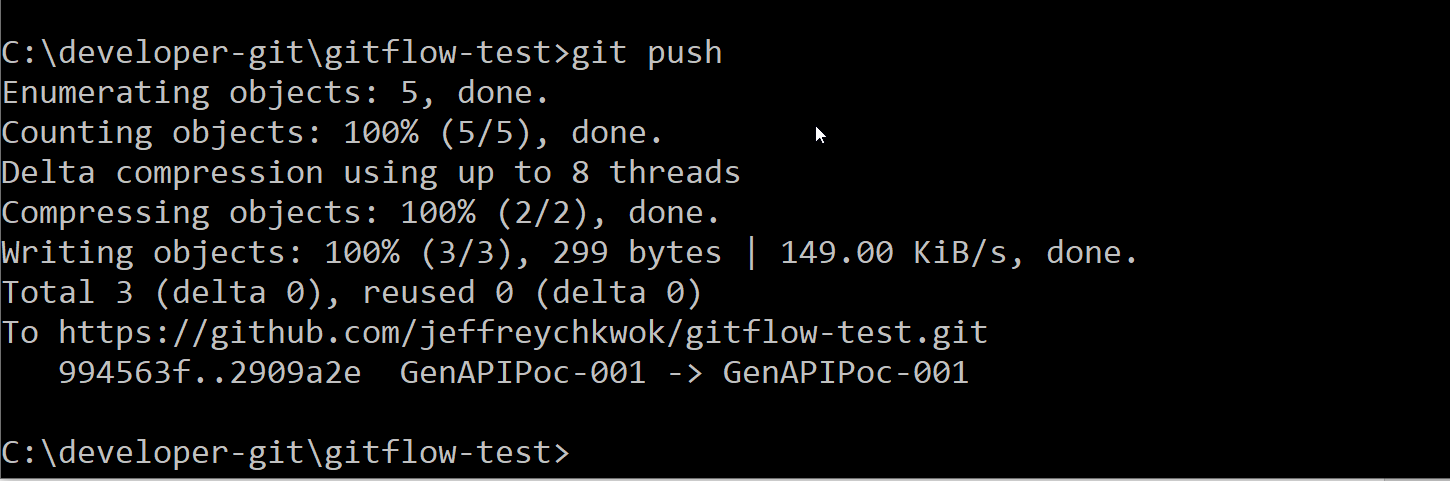
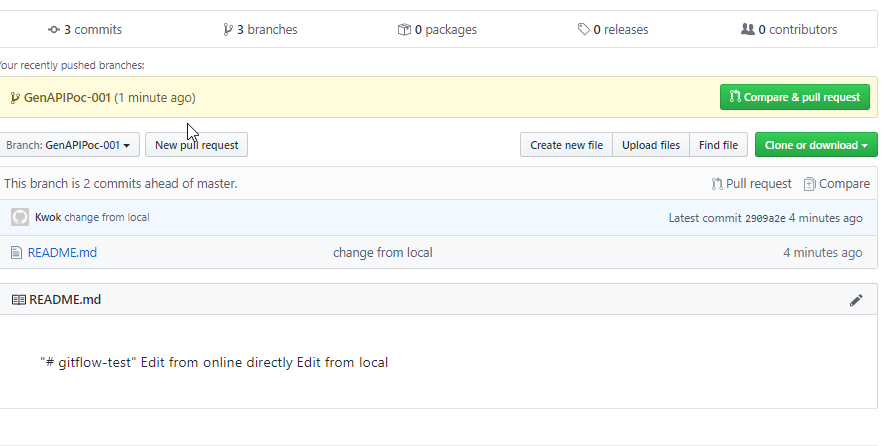
When you view the file content, latest changes should be downloaded.

### Commit the changes

1. Edit the README.md file and save the file
2. Add the file to stage. Enter **git add README.md**
3. Commit the file changes git commit -m “change from dev”

### Push the code

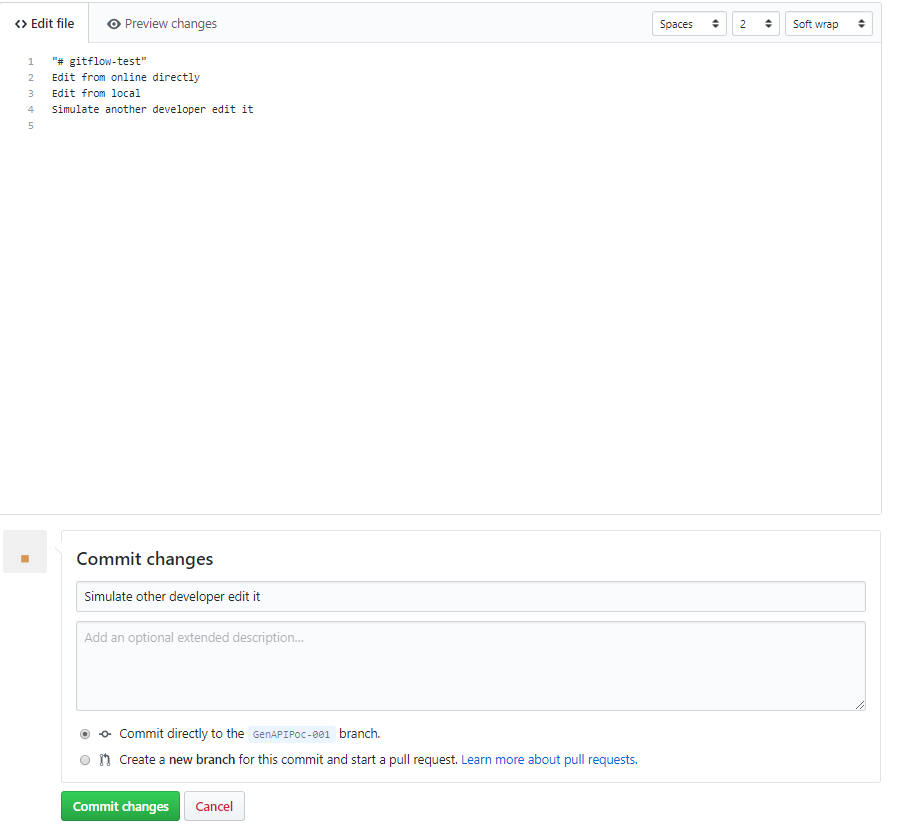
Upload the changes to central repository

1. Enter **git push**
2. Local changes is uploaded to central repository

## Merge conflict

Different team members may working on the same file which introduce code conflict. Developers should fix it when commit the code the repository

### Modify the file from remote repository

1. Edit the **README.md** again online directly to simulate another developer is updated it

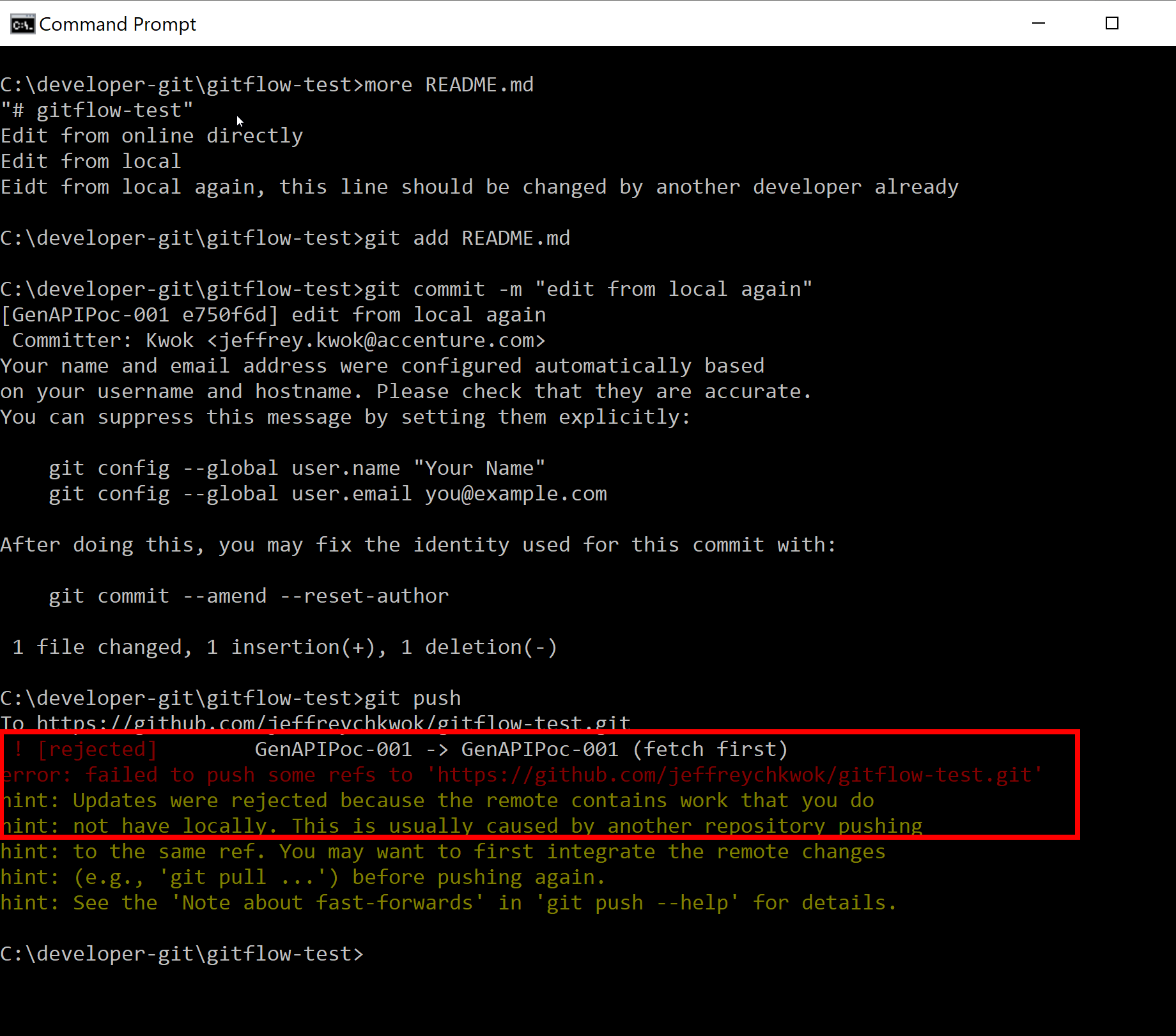
Save and **Commit Changes**

### Modify the file from local

We edit the 4th line from local and this line should be updated in the central repository already

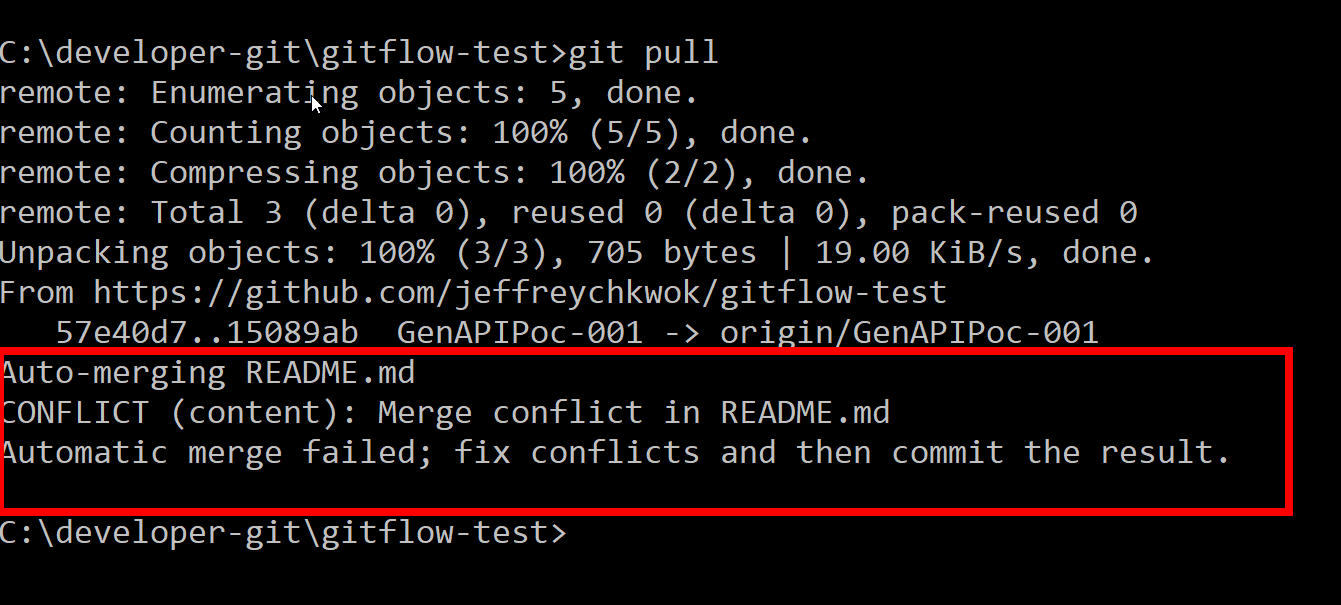
### Push the code and merge conflict

1. Enter git add README.md
2. Enter git commit -m “edit from local again”
3. Enter git push

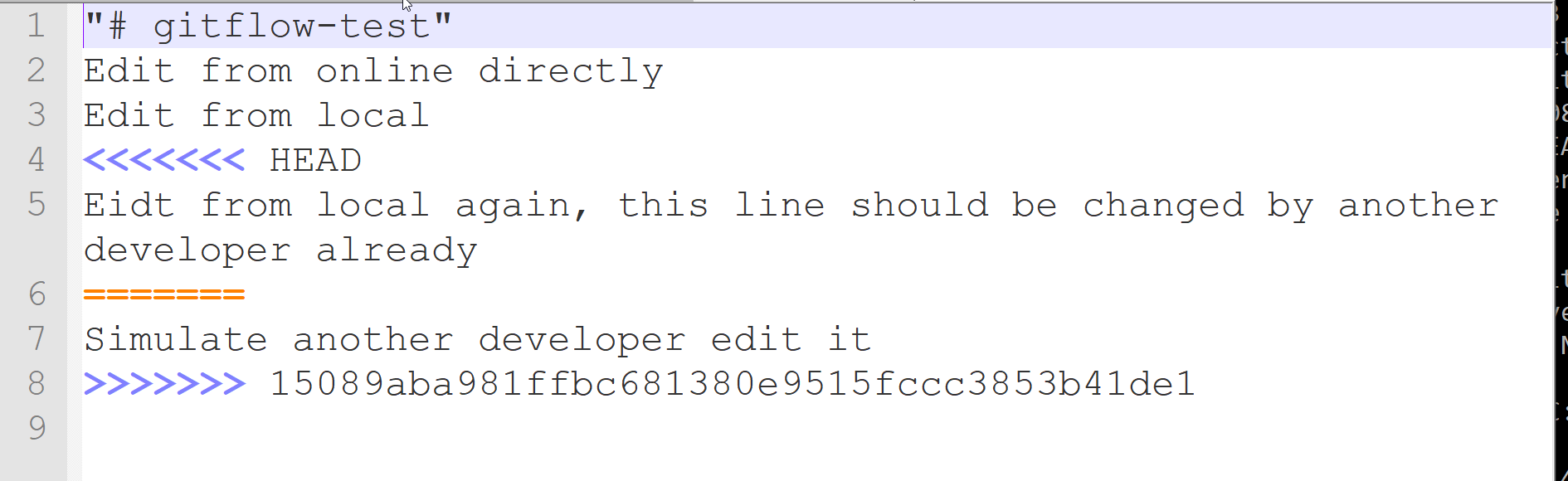


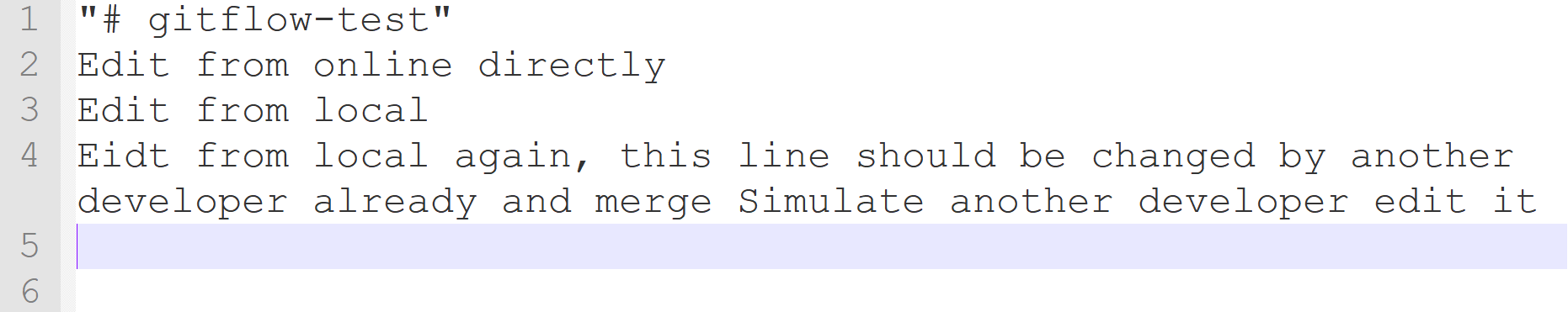
The push is **rejected** as the local changes conflict with the one in the central repository. Git does not know which version is correct. We needed to resolve the conflict.

1. We pull the latest code from repository. Enter **git pull**



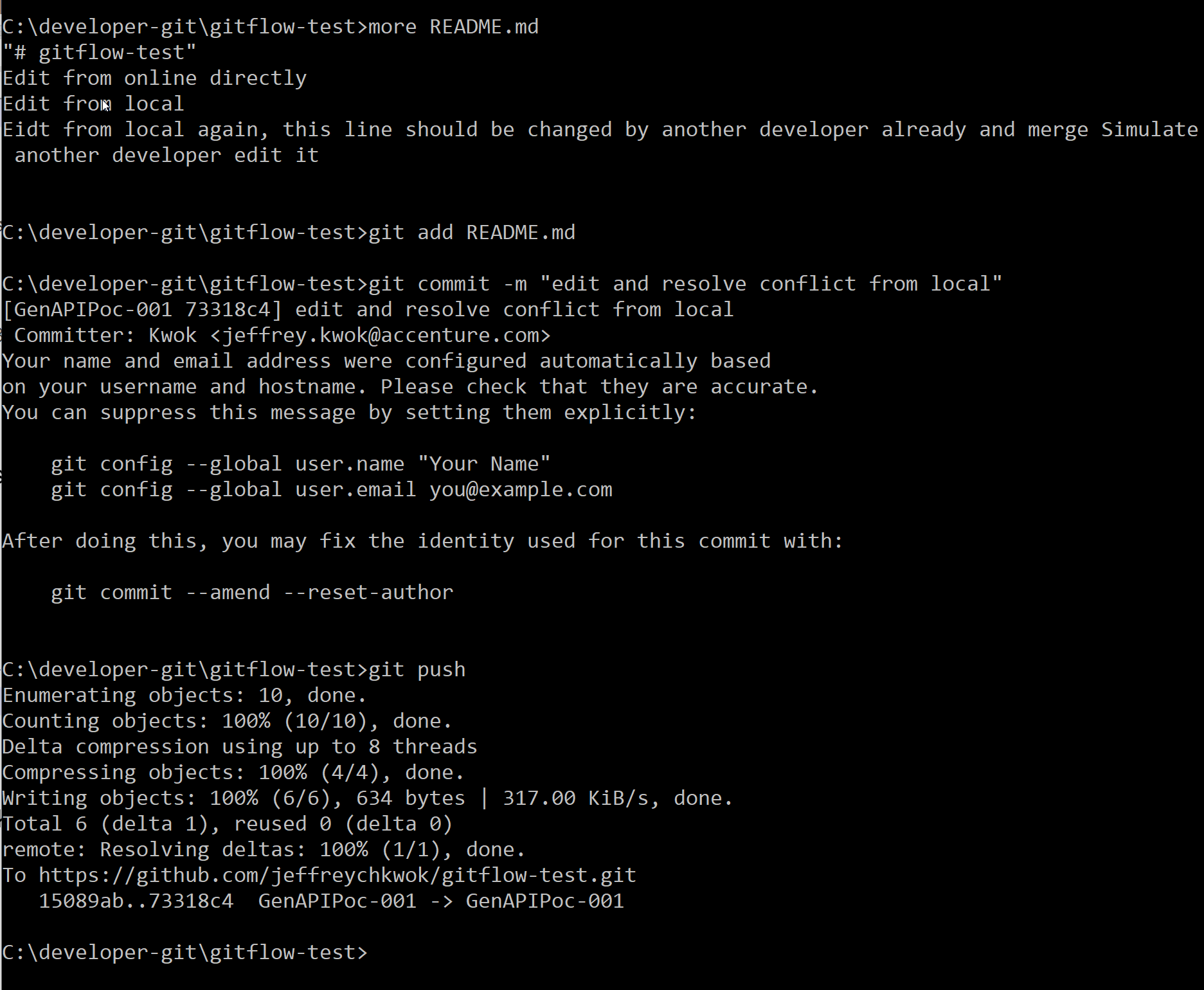
Git will try to merge the code for us. However, the automatic merge is fail as git have not idea which version should be kept when both versions working on the same line.

1. Now, we open the README.md file again and handle the change manually. Conflict lines are indicated in the file
2. We can edit the file directly to let git knows what should be the content.



\*\*There are different code merge tools in the market to help us handle the conflict easier. Such as Eclipse Merge Tool,

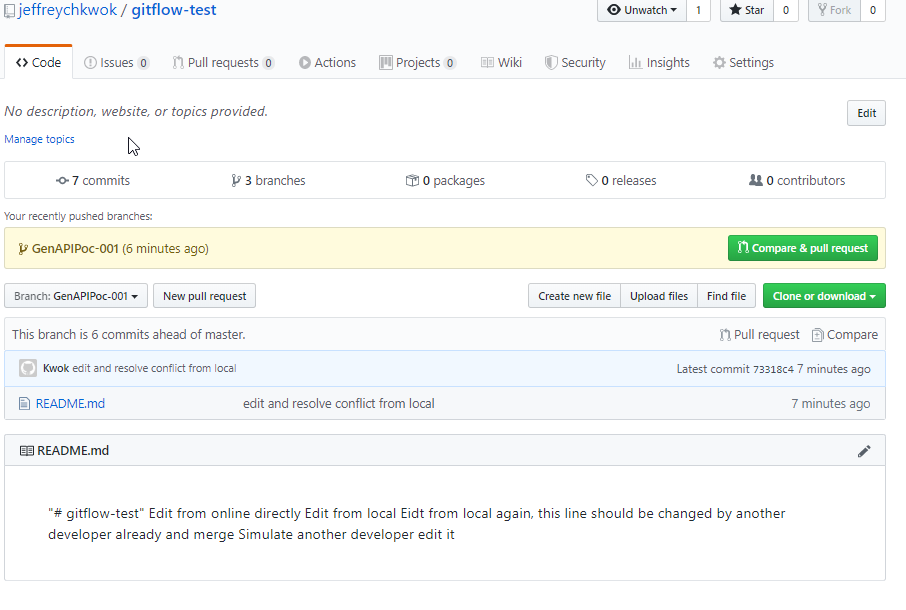
1. Commit the file again and push to repository

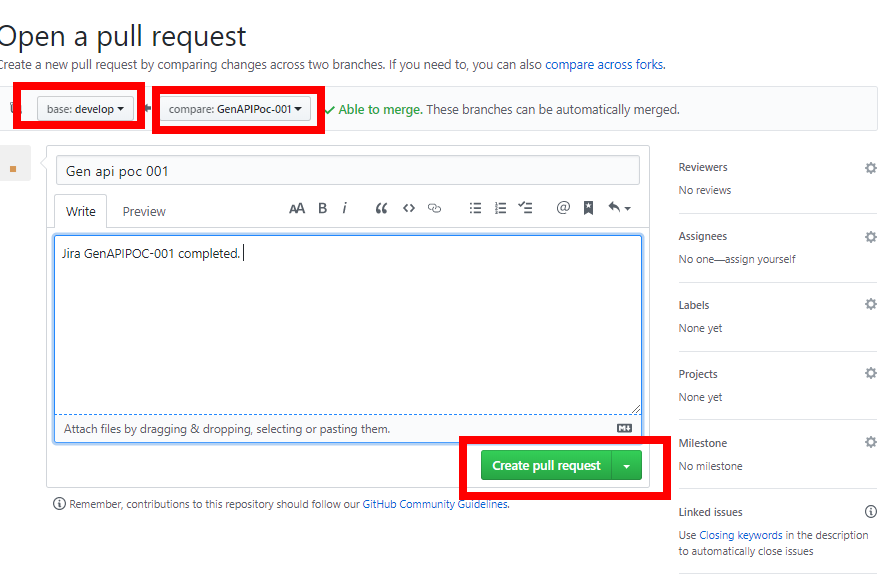


## Raise pull request

Once the feature development is completed, we should merge the code from feature branch back to develop branch. Before we merge the code, we should ask the owner of the develop branch to review the changes in the feature branch and approve the merge request.

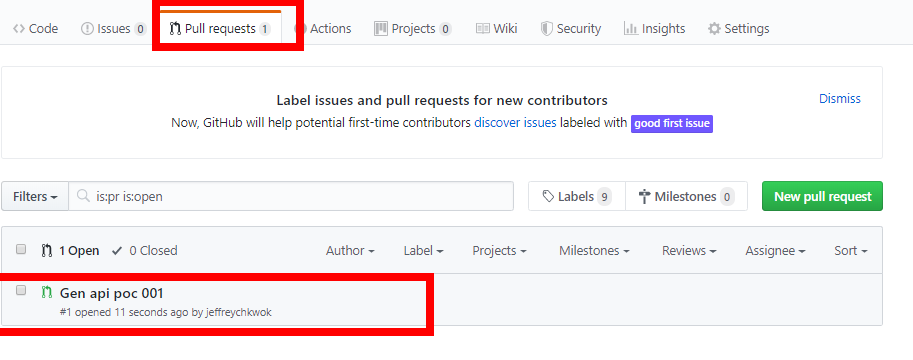
1. Raise the Pull Request to notify team members to review the code

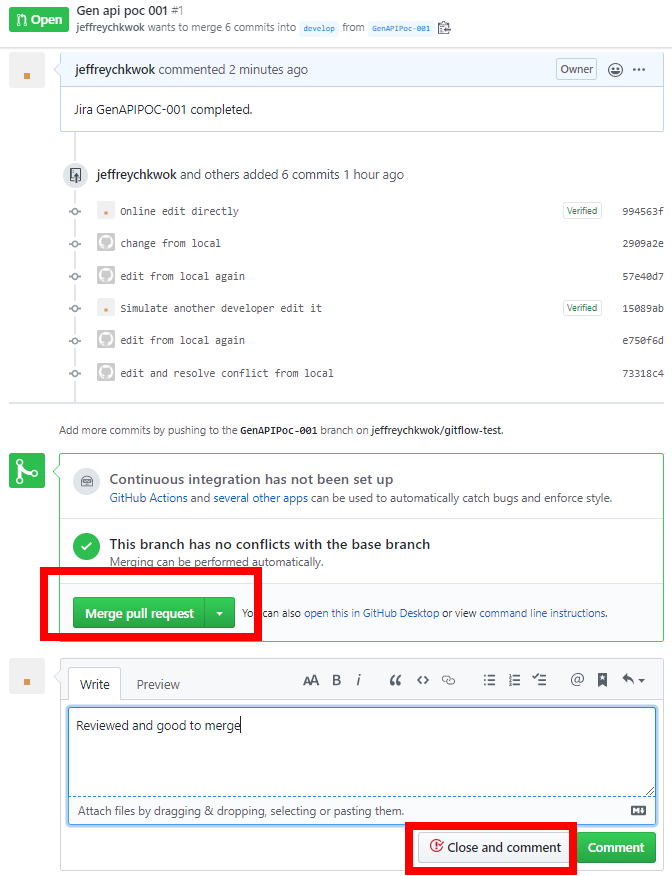
Click Compare & pull request.

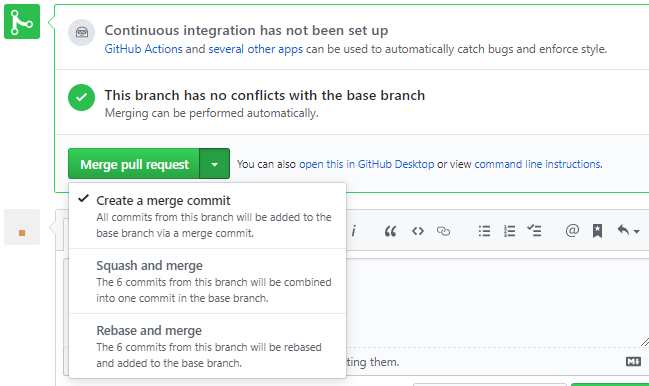
1. We want to merge from **GenAPIPoc-001** to **develop**
2. Click Create pull request

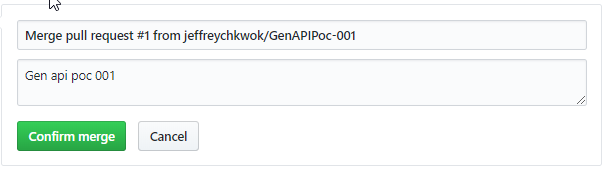
## Merge Branch

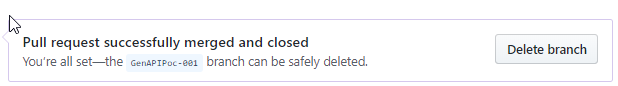
1. When other developers/ project maintainer login github, they can find the pull requests there.

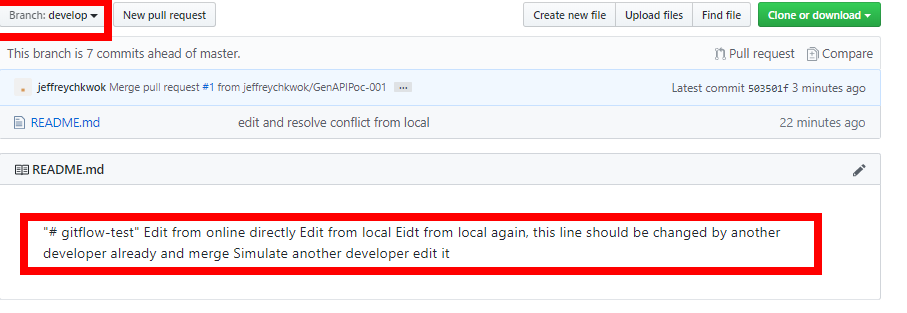


1. They can reivew the changes and approve/reject the request.
2. Click **Merge pull request** to merge feature branch changes to develop branch





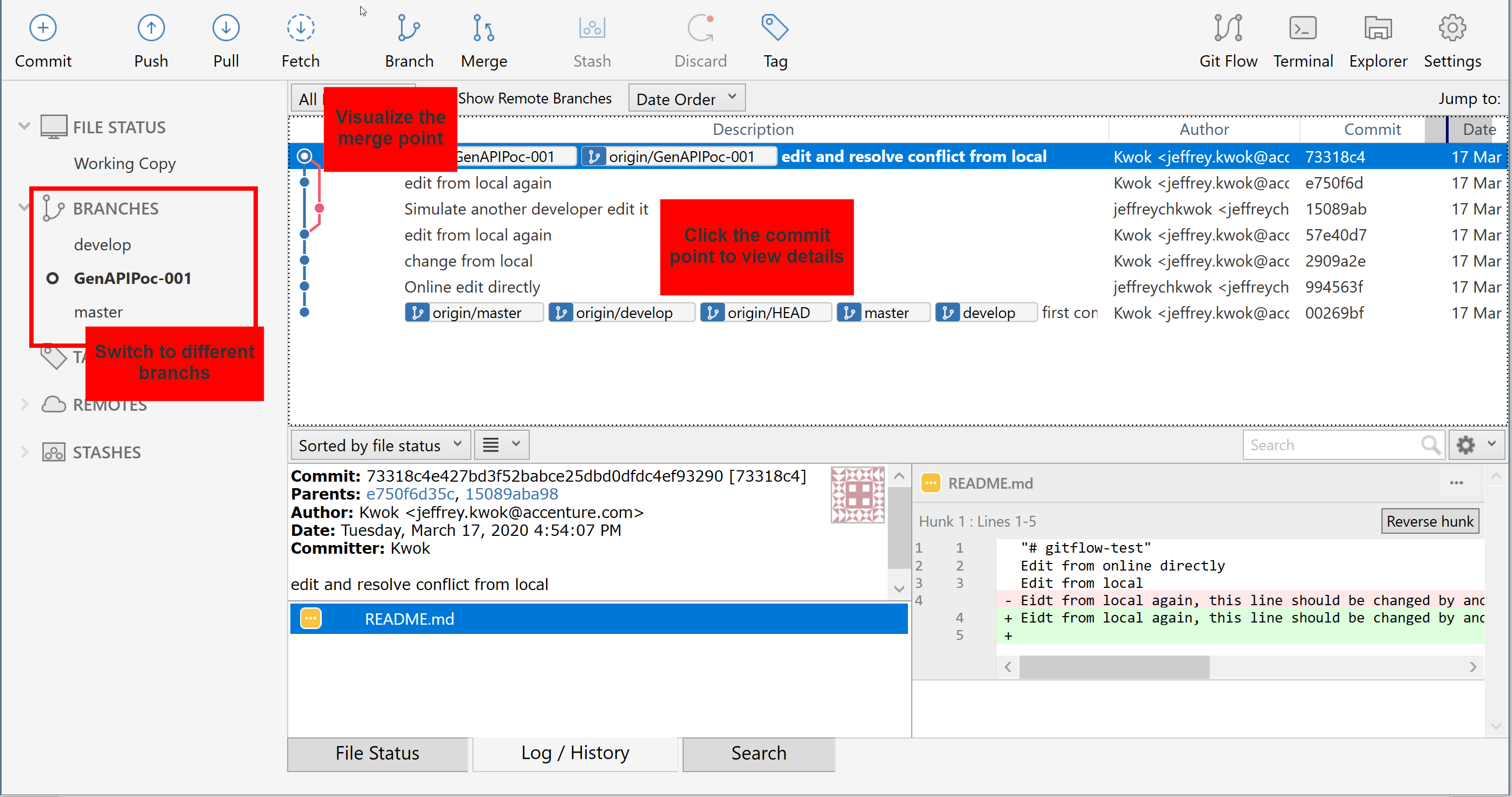


1. Changes are merged to develop branches

# SOURCETREE (Optional)

Sourcetree is a free Git GUI client for Windows/Mac manage your git repositories. It can let us focus on coding rather than git management

<https://www.sourcetreeapp.com/>



END of Document