ID______6388014__ Name __________WarisDamkham_________Group _____Samoyed_______

Lab Exercise #1

Version Control System (Git) and Code Review

Create a copy of this exercise sheet and fill in your answers. After finishing, save as a PDF and submit on MyCourses.

This lab exercise is both individual and team work. Complete the exercise by following the steps below [1]*.

Git is a distributed version control system for tracking changes in source code during software development. It is designed for coordinating work among programmers, but it can be used to track changes in any set of files. Its goals include speed, data integrity, and support for distributed, non-linear workflows. — Wikipedia

0. Preparation (individual):

1. Installing Git on your operating system by downloading the installation file here: https://git-scm.com/download

2. Getting yourself familiar with basic command line terminal and tools.

   Windows

   1) Open Command Prompt
   2) Try the following commands

Command	Description
`dir`	Show items at the current directory
`cd`	Show the current directory
`cd <location>`	Change directory to <location>
`mkdir <name>`	Create a directory

[1] Some of the steps are copied from https://www.atlassian.com/git/tutorials/install-git

3) Create a new directory called “test” for your software project

```
$ cd C:\<Your Preferred Location>
$ mkdir test
$ cd test
```

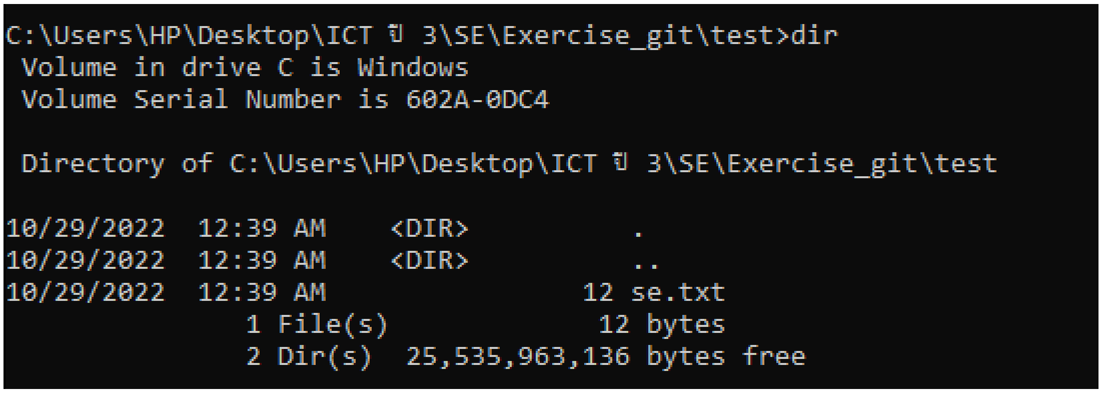
4) Open Vim (text editor) and learn basic functions by following the steps below.

```
$ notepad se.txt
```

5) Type “Hello ITCS371”.
6) Save the file
7) You’ll be back at the prompt again. Check the file(s) in the current directory.

```
$ dir
```

8) Save a screenshot of your screen with the command prompt at step 7 and paste below.



```
C:\Users\HP\Desktop\ICT ปี 3\SE\Exercise_git\test>dir
 Volume in drive C is Windows
 Volume Serial Number is 602A-0DC4

 Directory of C:\Users\HP\Desktop\ICT ปี 3\SE\Exercise_git\test

10/29/2022  12:39 AM    <DIR>          .
10/29/2022  12:39 AM    <DIR>          ..
10/29/2022  12:39 AM                12 se.txt
               1 File(s)             12 bytes
               2 Dir(s)  25,535,963,136 bytes free
```

1. Git Setup (individual)

1) Check Git version

```
$ git --version
```

2) See all Git commands

```
$ git help
```

3) Configure your Git username and email using the following commands, **replacing Emma's name with your own.** These details will be associated with any commits that you create:

```
$ git config --global user.name "Emma Paris"
$ git config --global user.email "eparis@gmail.com"
```

2. Setting up a local repository (individual)

1) Initialize Git in the `test` project directory (i.e., repository) that you created previously.

```
$ cd <location>/<to>/<your>/test
$ git init
```

2) Check the status of files in your Git repository.

```
$ git status
```

3) Add the se.txt file to staging mode.

```
$ git add se.txt
```

4) Check the status of files in your Git repository.

```
$ git status
```

5) Commit the change.

```
$ git commit -m “Added se.txt”
```

6) Check the status of files in your Git repository.

```
$ git status
```

7) See the list of commits

```
$ git log
$ git log --name-status
```

8) Modify the file se.txt further by adding one more line

se.txt

```
Hello ITCS371
Introduction to Software Engineering
```

9) Check the status

```
$ git status
```

10) See the new changes

```
$ git diff se.txt
```

11) Stage the file and commit the change

```
$ git add se.txt
$ git commit -m “Added a modification to se.txt”
```

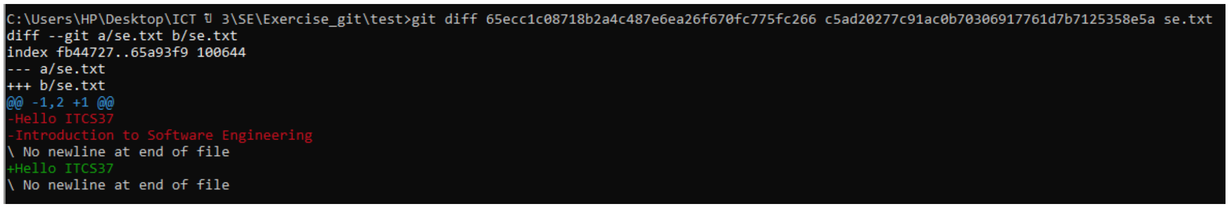
12) See all the commits again

```
$ git log
$ git log --name-status
```

13) Compare the file a.txt between two commits

```
$ git diff <commitID#1 (older)> <commitID#2> se.txt
```

14) Save a screenshot of your screen with the command prompt at step 13 and paste below.



```
C:\Users\HP\Desktop\ICT ปี 3\SE\Exercise_git\test>git diff 65ecc1c08718b2a4c487e6ea26f670fc775fc266 c5ad20277c91ac0b70306917761d7b7125358e5a se.txt
diff --git a/se.txt b/se.txt
index fb44727..65a93f9 100644
--- a/se.txt
+++ b/se.txt
@@ -1,2 +1 @@
-Hello ITCS37
-Introduction to Software Engineering
\ No newline at end of file
+Hello ITCS37
\ No newline at end of file
```

3. Work on a remote repository on GitHub (Team)

Next, we will use Git to work in a team using the team project repository that you already have on GitHub.

1) If you are using Mahidol's Wi-Fi, add the proxy authentication into your machine git configuration (skip this if you're using your machine at home).

```
$ git config --global http.proxy http://<Mahidol Username>:<Mahidol
Password>@proxy-sa.mahidol:8080
```

2) Clone the group project's repository from GitHub by going to github.com, login and navigate to your project, and copy the repo's clone URL.
3) Fill in the URL of your project's repository here:

https://github.com/ICT-Mahidol/2022-ITCS371-1-Samoyed.git

4) Back to your Terminal, clone the repository

```
$ cd <your>/<preferred>/<location>
$ git clone <repo’s URL>
```

5) Go into your repo’s directory and check the status of the repo.

```
$ cd <repo name>
$ git status
```

6) Create a new folder for this exercise. Name it “lab-exercise-1”.

```
$ mkdir lab-exercise-1
$ cd lab-exercise-1
```

7) Create a new Java code file based on your first name called <Yourname>.java using any text editor.
   Add the following content by replacing <Yourname> with your first name.

```
public class <Yourname> {
    public static void main(String[] args) {
        System.out.println("Hello <Yourname>");
    }
}
```

8) Stage the file and commit

```
$ git add .
$ git commit -m “Added <Yourname> class file”
```

9) Push the commit to a remote repository on GitHub

```
$ git push
```

10) Check the status again

```
$ git status
```

11) Wait until all the team members push their changes to GitHub, then compare the status of your local repository to the remote repository

```
$ git fetch
```

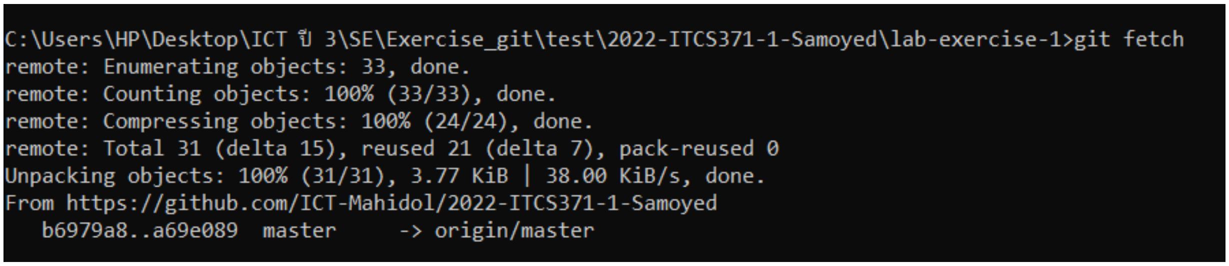
12) Pull the new changes and merge them into your local repository.

```
$ git pull
```

13) Check what is added into the repository and investigate the commit logs.

```
$ ls (Linux) or cd (Windows)
$ git log
```

14) Save a screenshot of your screen with the command prompt at step 11 and paste below.



```
C:\Users\HP\Desktop\ICT ปี 3\SE\Exercise_git\test\2022-ITCS371-1-Samoyed\lab-exercise-1>git fetch
remote: Enumerating objects: 33, done.
remote: Counting objects: 100% (33/33), done.
remote: Compressing objects: 100% (24/24), done.
remote: Total 31 (delta 15), reused 21 (delta 7), pack-reused 0
Unpacking objects: 100% (31/31), 3.77 KiB | 38.00 KiB/s, done.
From https://github.com/ICT-Mahidol/2022-ITCS371-1-Samoyed
   b6979a8..a69e089  master     -> origin/master
```

4. Branching

1) See the current branches

```
$ git branch
```

2) Create a new branch and list all the branches again

```
$ git branch dev_<Yourname>
$ git branch
```

3) Switch to the new branch to make modifications (e.g., adding a new feature). Check that you have already moved to the new branch.

```
$ git checkout dev_<Yourname>
$ git branch
```

4) Create a new file for the newly developed feature using any editor and call it <YournameFeature>.java. Put the following code in.

```
 public class <Yourname>Feature {
      public void feature() {
          // this is a new feature
      }
  }
```

5) Stage the file and commit.

```
$ git add <YournameFeature>.java
$ git commit -m “Added a new feature”
```

6) See the current files in the repository (under the dev_<Yourname> branch)

```
$ ls (Linux) or cd (Windows)
```

7) Switch back to the master branch and list the files again. Compared to the previous result.

```
$ git checkout master
$ ls (Linux) or cd (Windows)
```

8) Push the newly created branch and the changes to GitHub

```
$ git push --set-upstream origin dev_<Yourname>
```

9) Go to GitHub and see the branches.
10) Pull new branches created by your teammates into your local repository

```
$ git fetch
$ git branch -a
```

11) Switch to the new branch

```
$ git checkout -b <branchname> origin/branchname
```

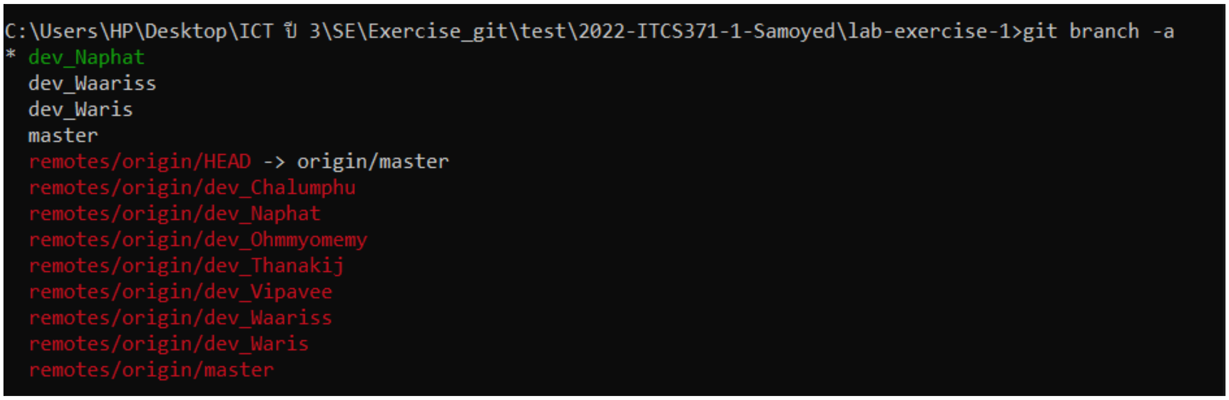
12) Check the files in the branch and compare them to the master branch.

```
ls (Linux) or cd (Windows)
```

13) Show all the branches

```
git branch -a
```

14) Save a screenshot of your screen with the command prompt at step 13 and paste below.

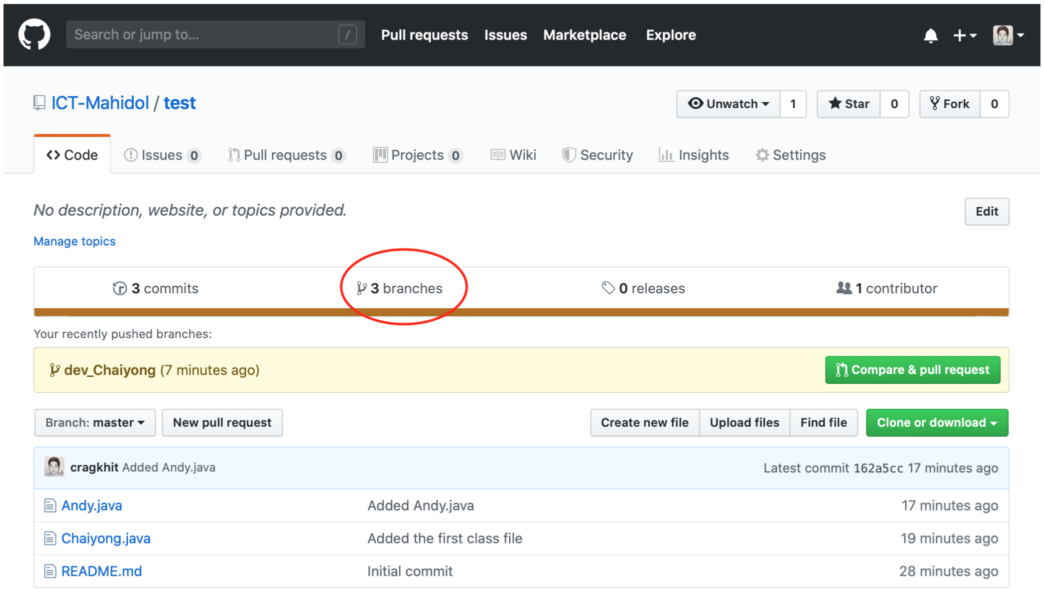


```
C:\Users\HP\Desktop\ICT ปี 3\SE\Exercise_git\test\2022-ITCS371-1-Samoyed\lab-exercise-1>git branch -a
* dev_Naphat
  dev_Waariss
  dev_Waris
  master
  remotes/origin/HEAD -> origin/master
  remotes/origin/dev_Chalumphu
  remotes/origin/dev_Naphat
  remotes/origin/dev_Ohmmyomemy
  remotes/origin/dev_Thanakij
  remotes/origin/dev_Vipavee
  remotes/origin/dev_Waariss
  remotes/origin/dev_Waris
  remotes/origin/master
```

5. Merging and Code Review (Pull Request)

Now we are ready to merge the new feature made by each teammate into the master branch. Before merging the change, we will perform a code review first.

1) Go to your repository on GitHub website. Select “Branches”



No description, website, or topics provided.

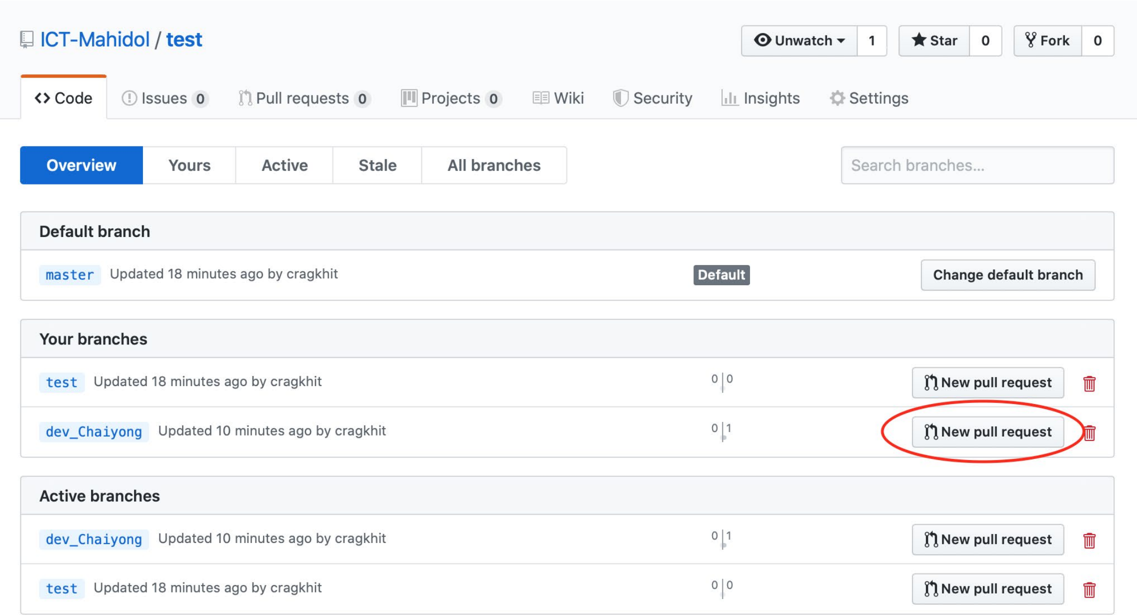
Your recently pushed branches:

dev_Chaiyong (7 minutes ago)

cragkhit Added Andy.java Latest commit 162a5cc 17 minutes ago

Andy.java	Added Andy.java	17 minutes ago
Chaiyong.java	Added the first class file	19 minutes ago
README.md	Initial commit	28 minutes ago

2) Select your branch and select “New pull request”. Every team member must do this for their own branch.



Default branch

master	Updated 18 minutes ago by cragkhit	Default	Change default branch

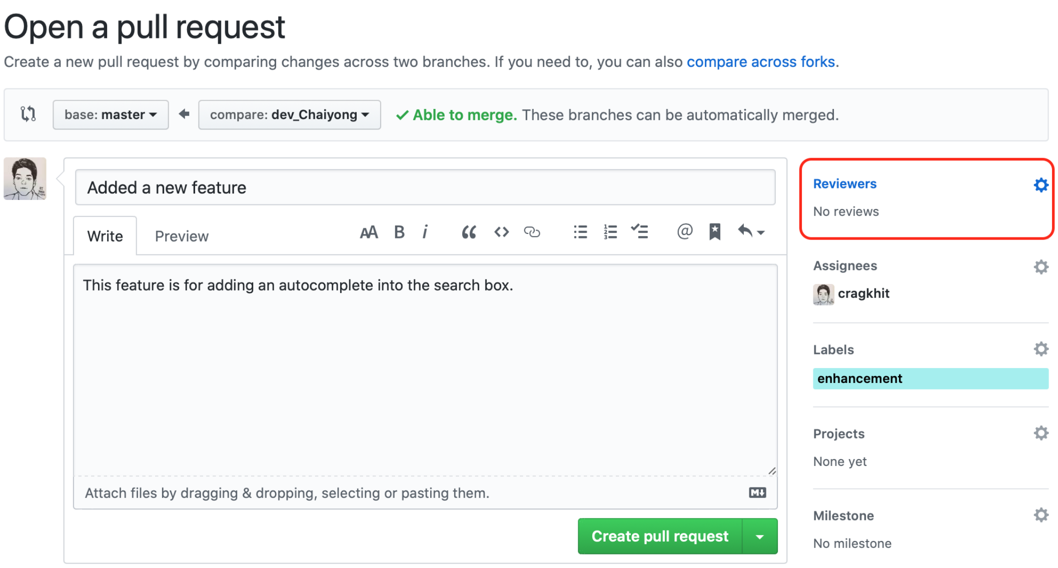
Your branches

test	Updated 18 minutes ago by cragkhit	0 | 0	New pull request
dev_Chaiyong	Updated 10 minutes ago by cragkhit	0 | 1	New pull request

Active branches

dev_Chaiyong	Updated 10 minutes ago by cragkhit	0 | 1	New pull request
test	Updated 18 minutes ago by cragkhit	0 | 0	New pull request

3) In the Pull Request page, fill in the description of the pull request in order to help the reviewer during code review. Then, assign the pull request to all the team members and select other properties for the pull request. Finally, press the “Create pull request” button.



Open a pull request

Create a new pull request by comparing changes across two branches. If you need to, you can also compare across forks.

base: master ← compare: dev_Chaiyong ✓ Able to merge. These branches can be automatically merged.

Added a new feature

Write Preview

This feature is for adding an autocomplete into the search box.

Attach files by dragging & dropping, selecting or pasting them.

Create pull request

Reviewers
No reviews

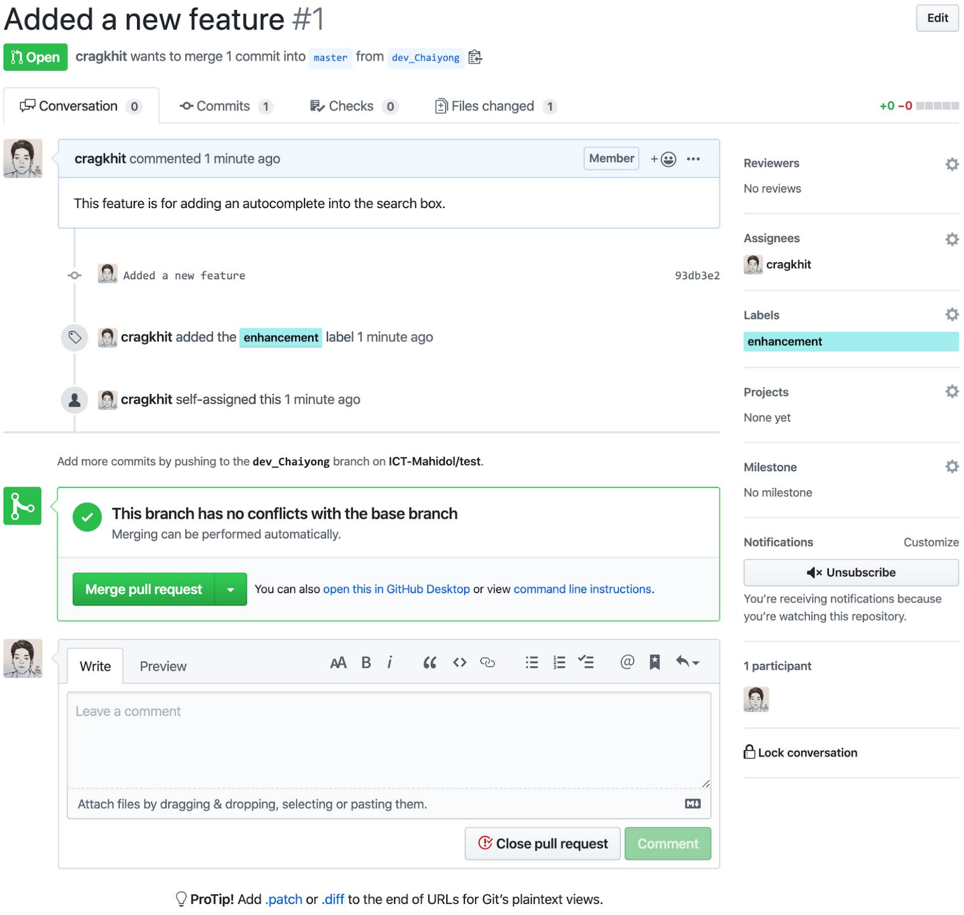
Assignees
cragkhit

Labels
enhancement

Projects
None yet

Milestone
No milestone

4) The new pull request (i.e., code review) will be created.



Added a new feature #1

Edit

Open cragkhit wants to merge 1 commit into `master` from `dev_Chaiyong`

Conversation 0 Commits 1 Checks 0 Files changed 1 +0 −0

cragkhit commented 1 minute ago Member

This feature is for adding an autocomplete into the search box.

`Added a new feature` `93db3e2`

cragkhit added the enhancement label 1 minute ago

cragkhit self-assigned this 1 minute ago

Add more commits by pushing to the **dev_Chaiyong** branch on **ICT-Mahidol/test**.

This branch has no conflicts with the base branch
Merging can be performed automatically.

Merge pull request You can also open this in GitHub Desktop or view command line instructions.

Write Preview

Leave a comment

Attach files by dragging & dropping, selecting or pasting them.

Close pull request Comment

ProTip! Add .patch or .diff to the end of URLs for Git's plaintext views.

Reviewers
No reviews

Assignees
cragkhit

Labels
enhancement

Projects
None yet

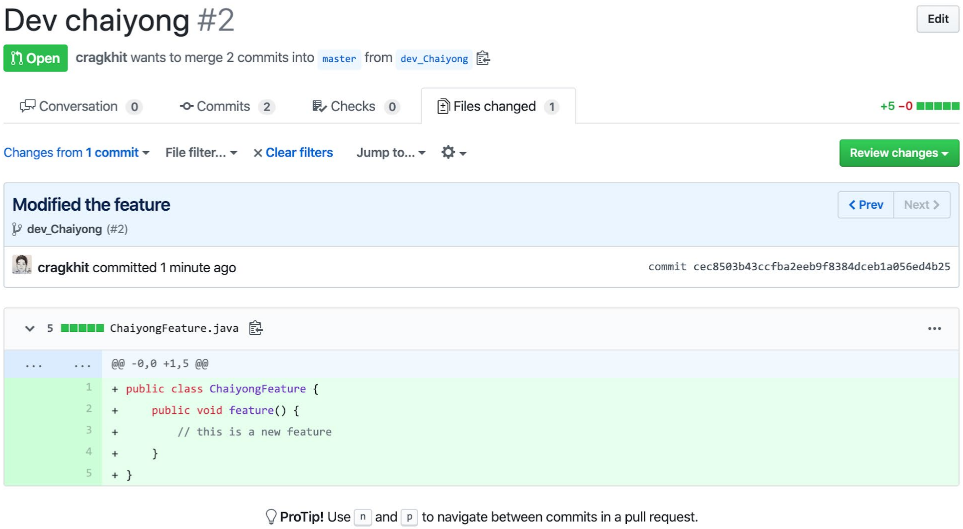
Milestone
No milestone

Notifications Customize
Unsubscribe
You're receiving notifications because you're watching this repository.

1 participant

Lock conversation

5) Click on a commit ID to see the changes and make a review.



Dev chaiyong #2

Open cragkhit wants to merge 2 commits into `master` from `dev_Chaiyong`

Modified the feature

dev_Chaiyong (#2)

cragkhit committed 1 minute ago

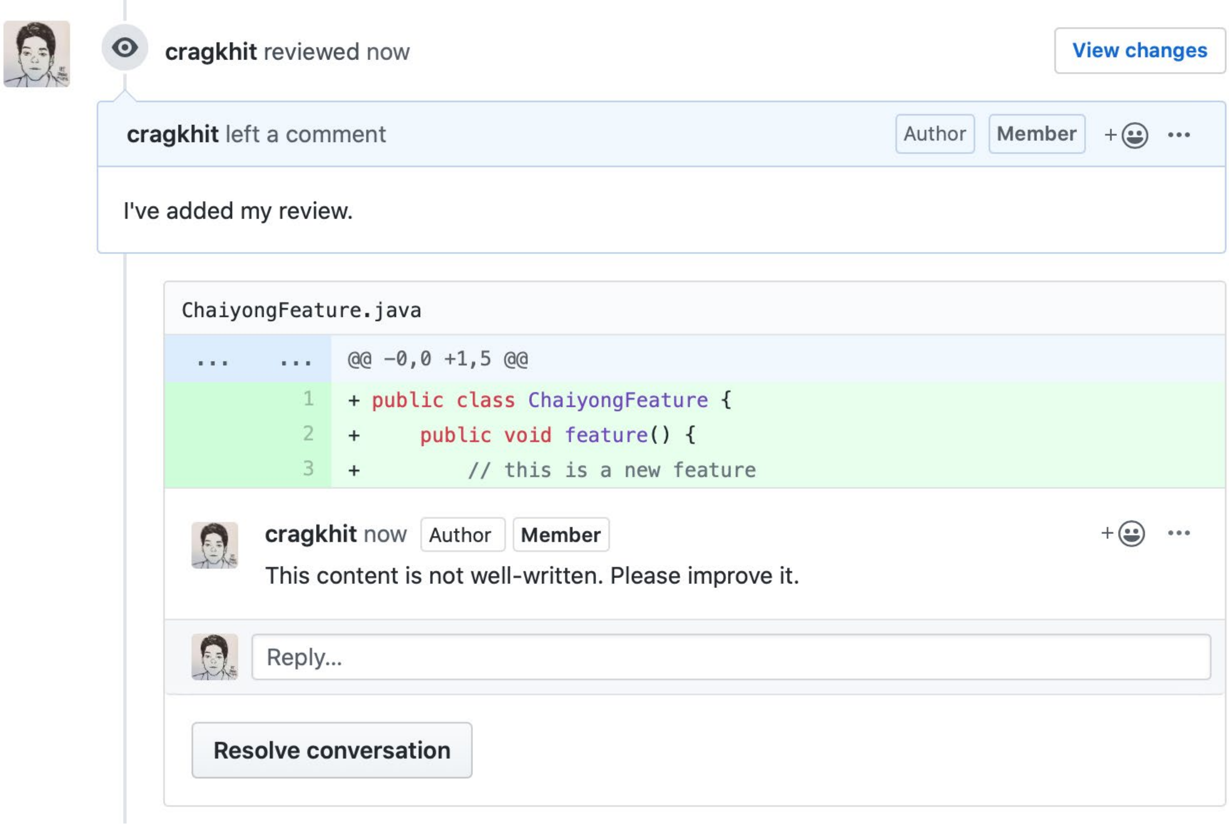
commit cec8503b43ccfba2eeb9f8384dceb1a056ed4b25

5 ChaiyongFeature.java

```
@@ -0,0 +1,5 @@
+ public class ChaiyongFeature {
+     public void feature() {
+         // this is a new feature
+     }
+ }
```

ProTip! Use n and p to navigate between commits in a pull request.

6) Click on the '+' sign in front of any line to add the review comment. Then, submit the review.



cragkhit reviewed now

cragkhit left a comment

I've added my review.

ChaiyongFeature.java

```
@@ -0,0 +1,5 @@
+ public class ChaiyongFeature {
+     public void feature() {
+         // this is a new feature
```

cragkhit now

This content is not well-written. Please improve it.

7) *** Every team member must give one review comment in every pull request. If not, you won’t get a score for this exercise. ***

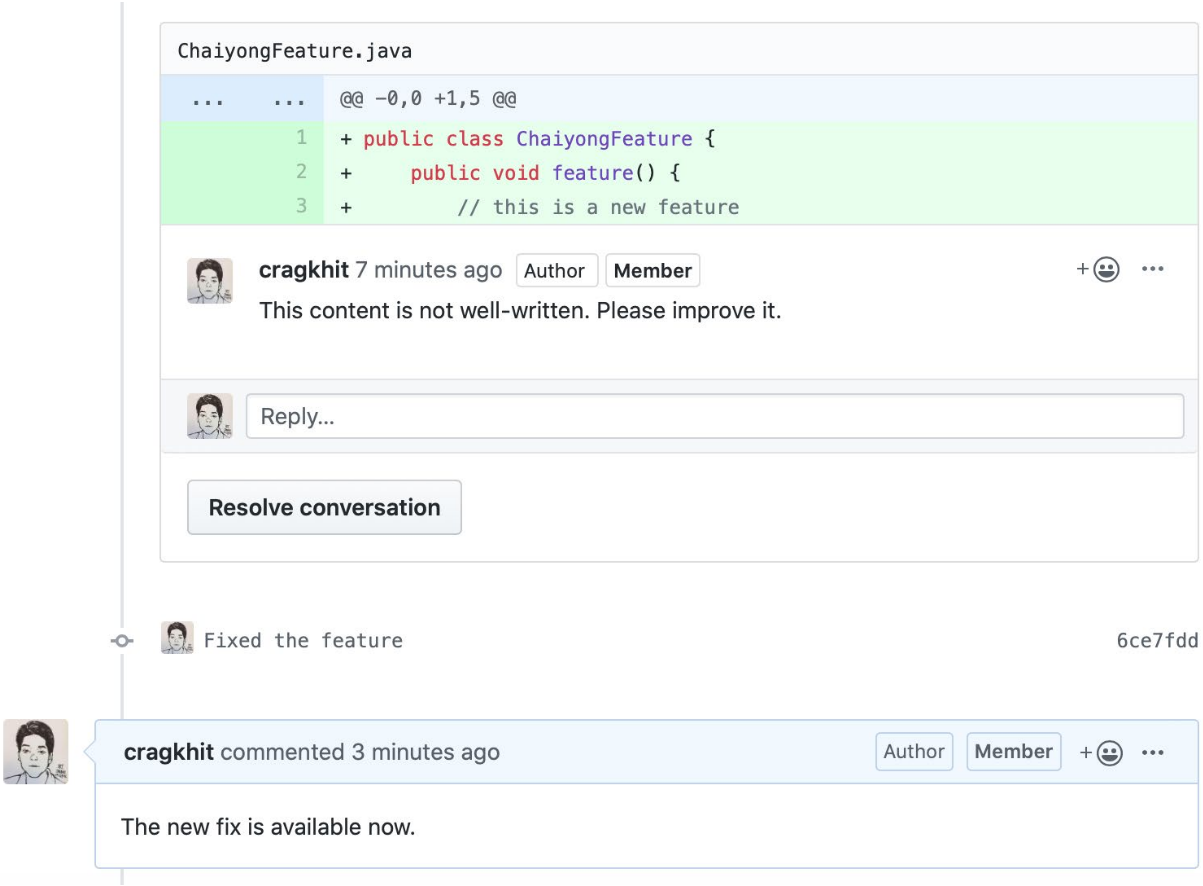
8) The owner of the branch comes back to the local repository and modifies the file as follows.

```
public class <Yourname>Feature {
     public void feature() {
         // this is a new feature
         int x = 0;
         x = x+1;
     }
 }
```

9) Commit and push the change to GitHub.

```
$ git add <Yourname>Feature.java
$ git commit -m “Fixed the feature”
$ git push
```

10) The owner of the branch comes back to GitHub. You’ll see the newly added commit in the pull request. Add a comment to let the reviewer know that the new fix is already available.



ChaiyongFeature.java

```
@@ -0,0 +1,5 @@
+ public class ChaiyongFeature {
+     public void feature() {
+         // this is a new feature
```

cragkhit 7 minutes ago Author Member

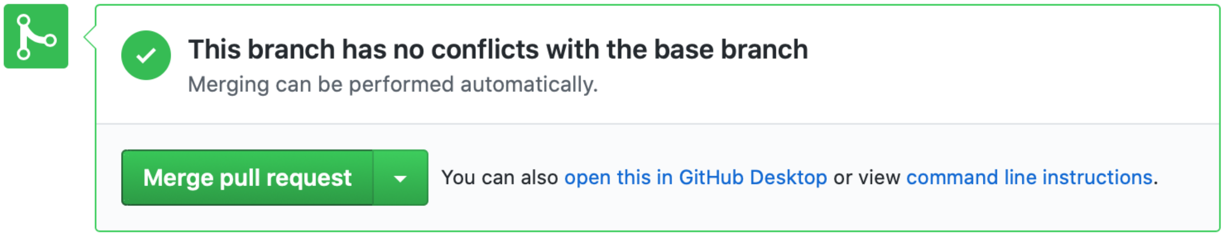
This content is not well-written. Please improve it.

Fixed the feature 6ce7fdd

cragkhit commented 3 minutes ago Author Member

The new fix is available now.

11) Everyone in the team reviews and comments again. If all the reviewers are satisfied with the change, click “Merge pull request” to integrate the new feature into the master branch.

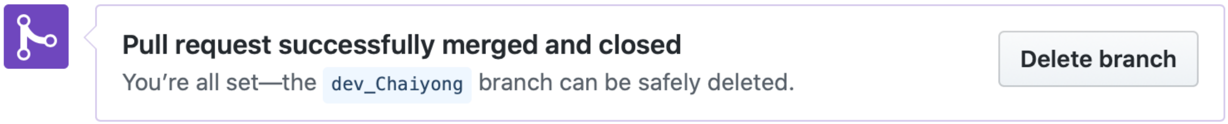


This branch has no conflicts with the base branch

Merging can be performed automatically.

You can also open this in GitHub Desktop or view command line instructions.

12) Delete the development branch after merging



Pull request successfully merged and closed

You’re all set—the `dev_Chaiyong` branch can be safely deleted.

13) Each team member comes back to their local repository and fetches the updates.

```
$ git fetch -p
$ git branch -vv
```

14) You can delete the branch that has already been merged from your local repository.
15) You do not need to upload screenshots for this part. We will check from your GitHub repository.