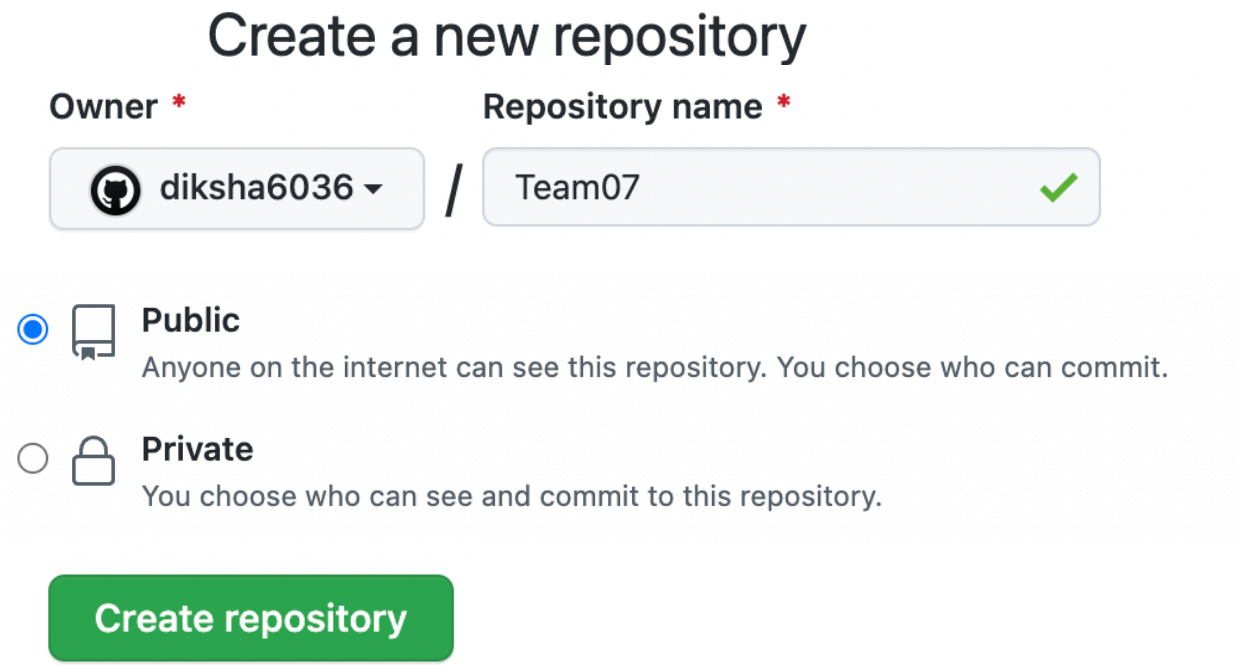
Task 1.2

➡ Add collaborators on GitHub Repo :-

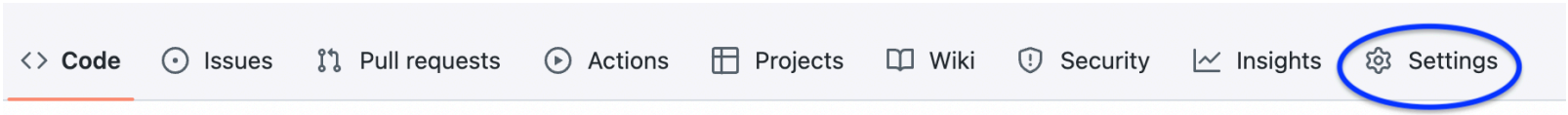
Step1 :

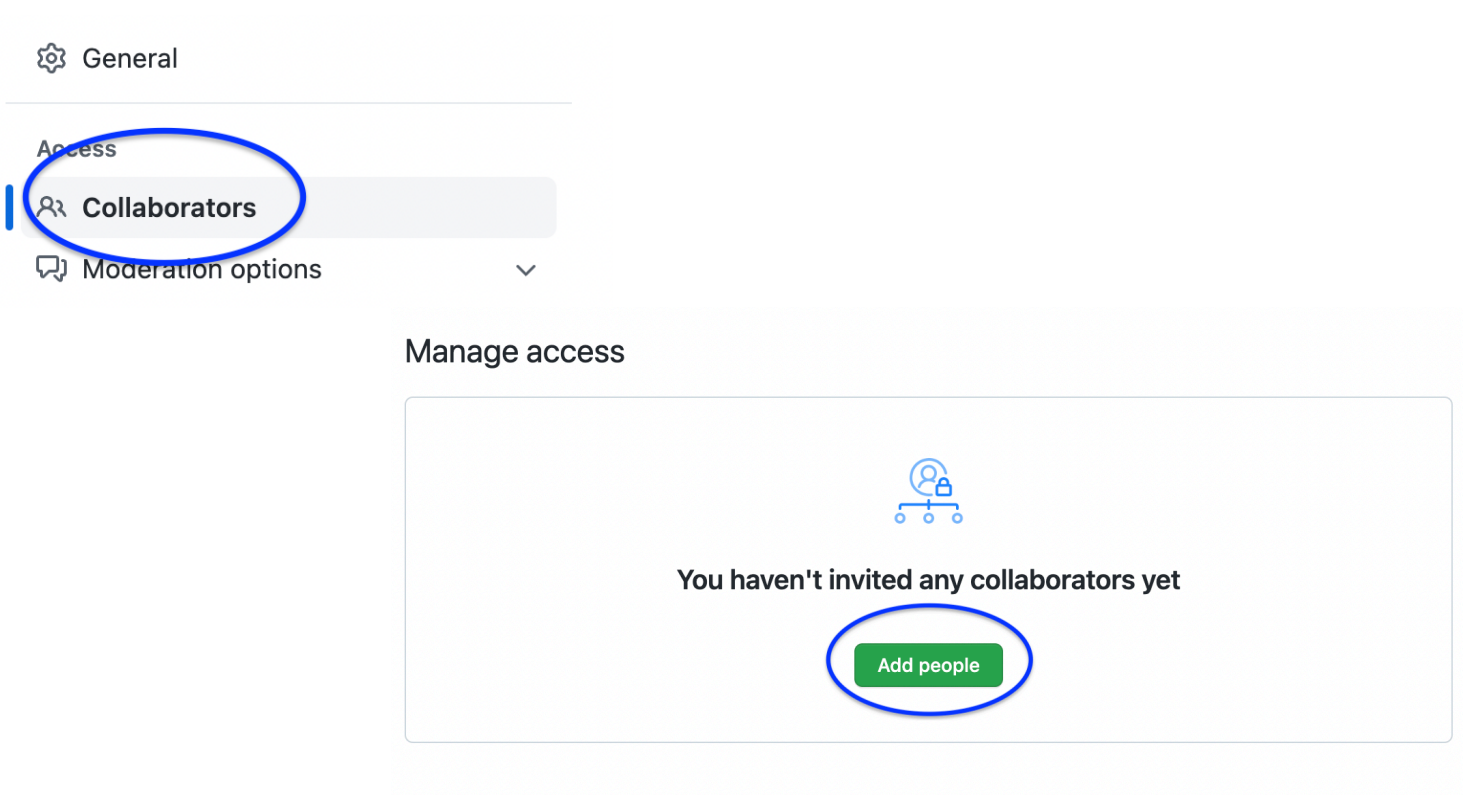
Create a new repository and name it .



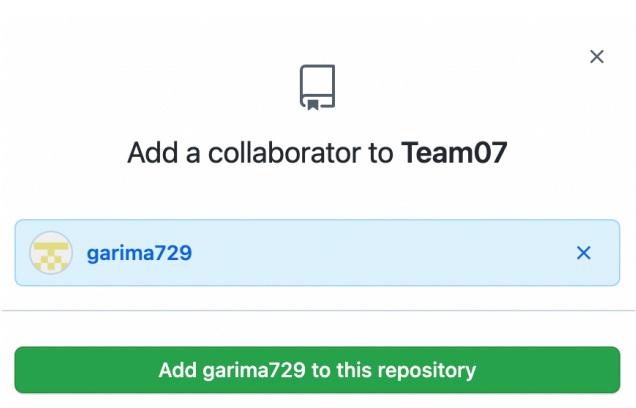
Step 2:

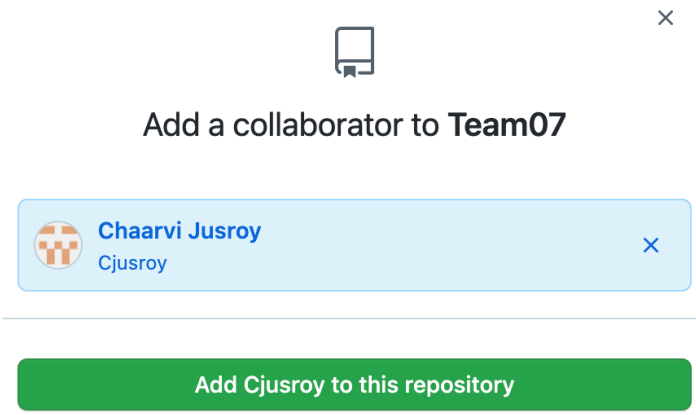
Now go to settings and proceed to add collaborators .

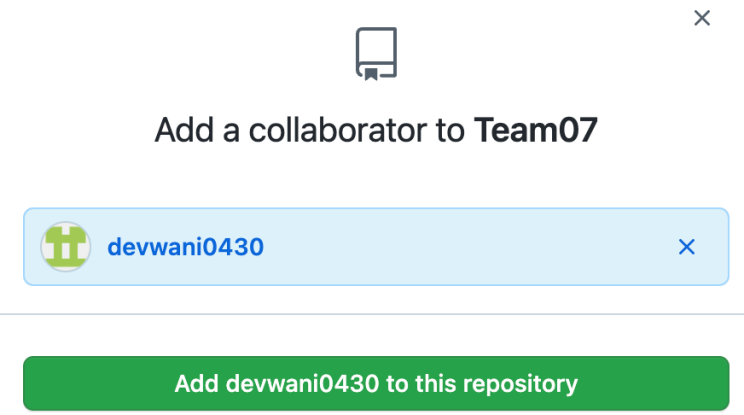




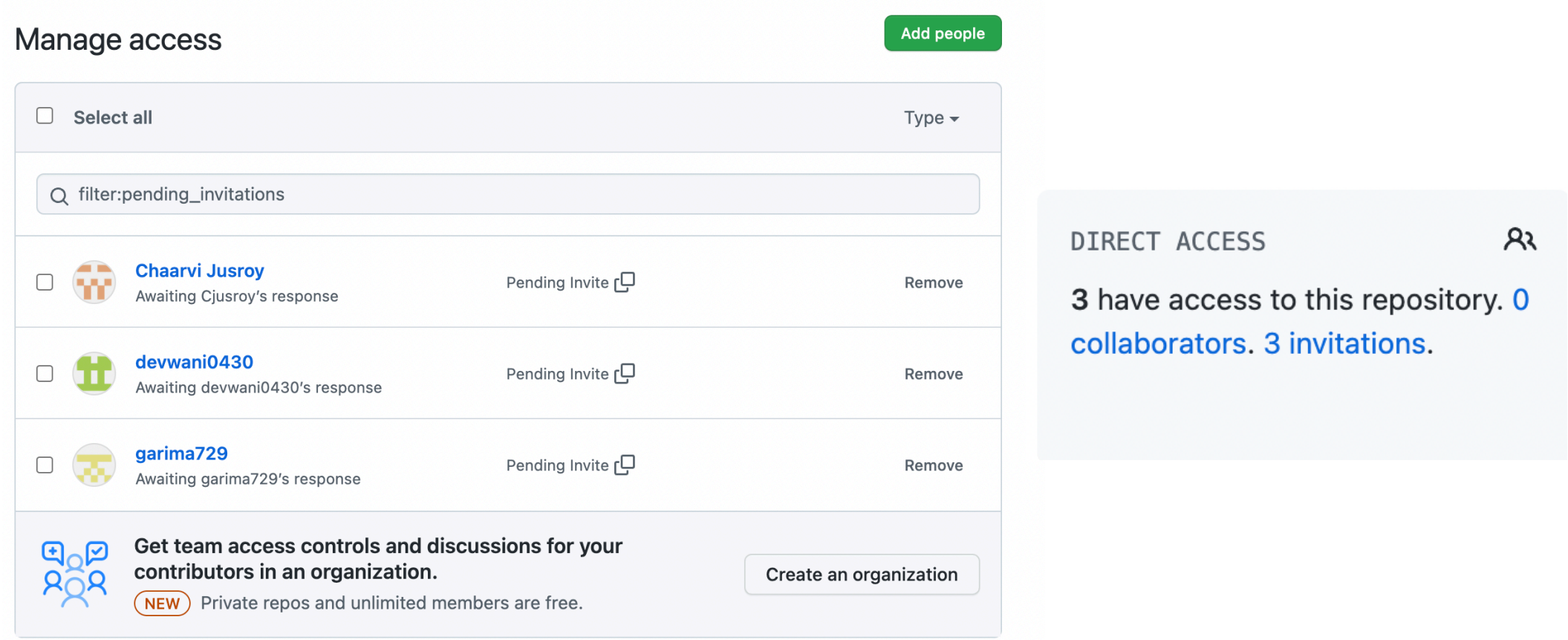
After clicking add people you have to search the people you want to add as a collaborator to your repo.



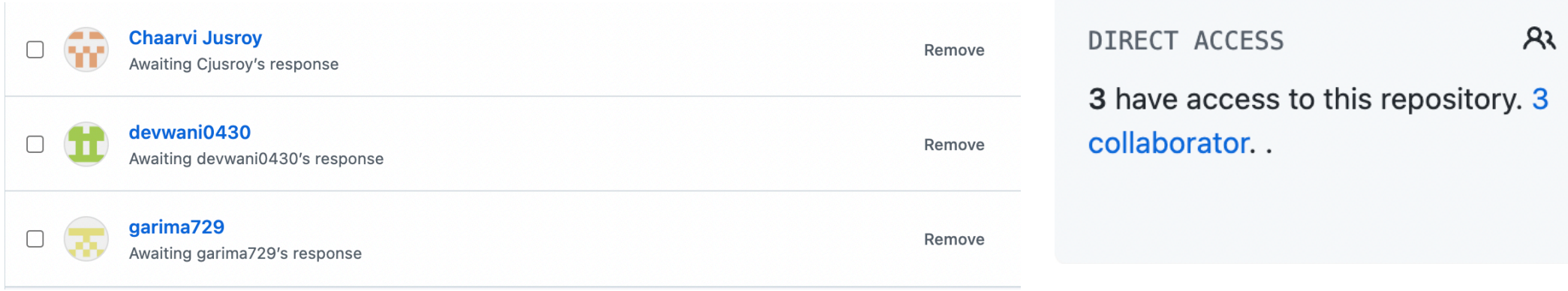




An invitation will be send to all the collaborators that whether they want to be a part os this repo or not.



After Accepting the invitation it will look like :



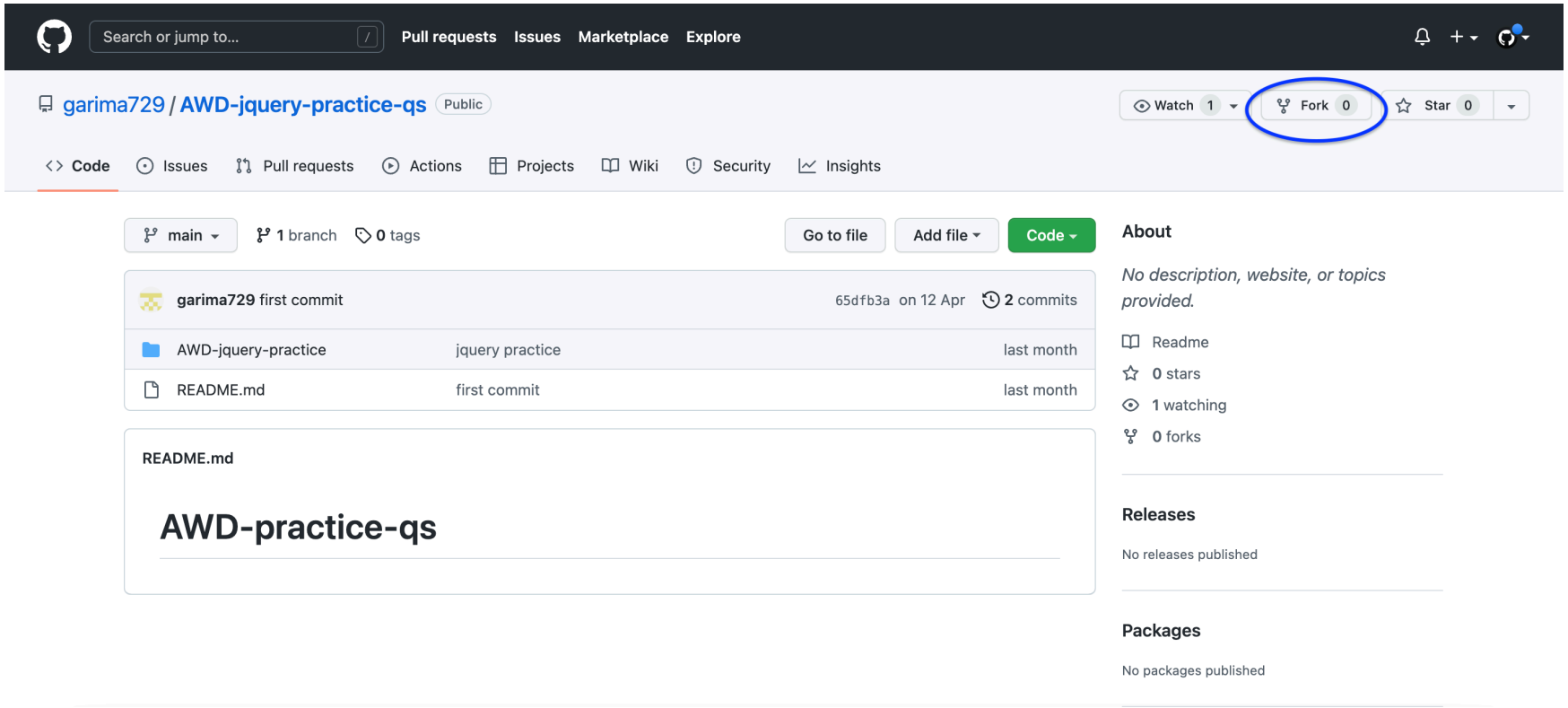
Fork and Commit :-

Fork: A fork is a copy of a repository that you manage. Forks let you make changes to a project without affecting the original repository. You can fetch updates from or submit changes to the original repository with pull requests.

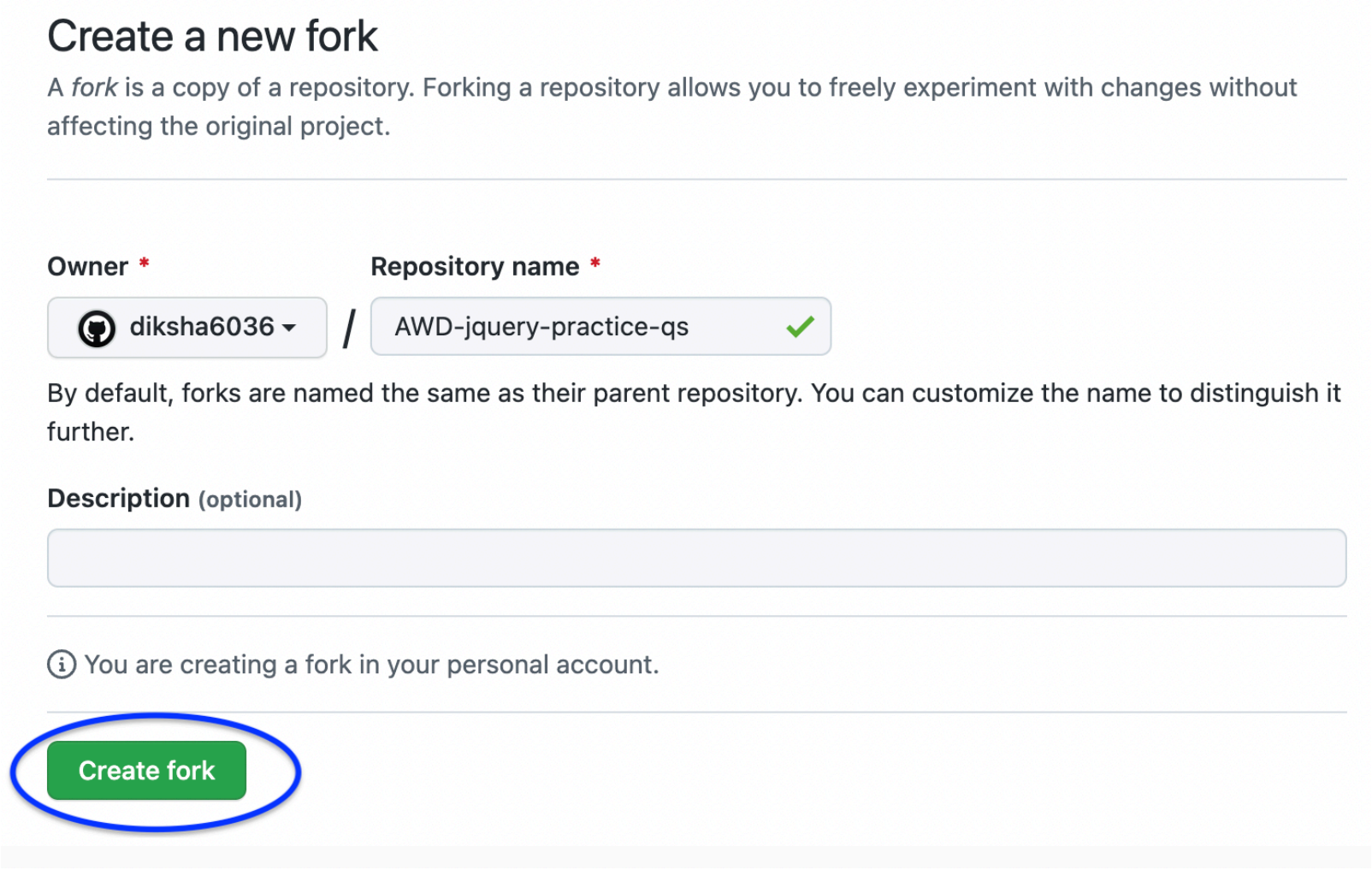
Steps to fork someone's repository:

Step 1:

Go to the repository which you want to fork and then click on fork.



and it will take a while to fork that repository into your own account and you can also rename that repo.



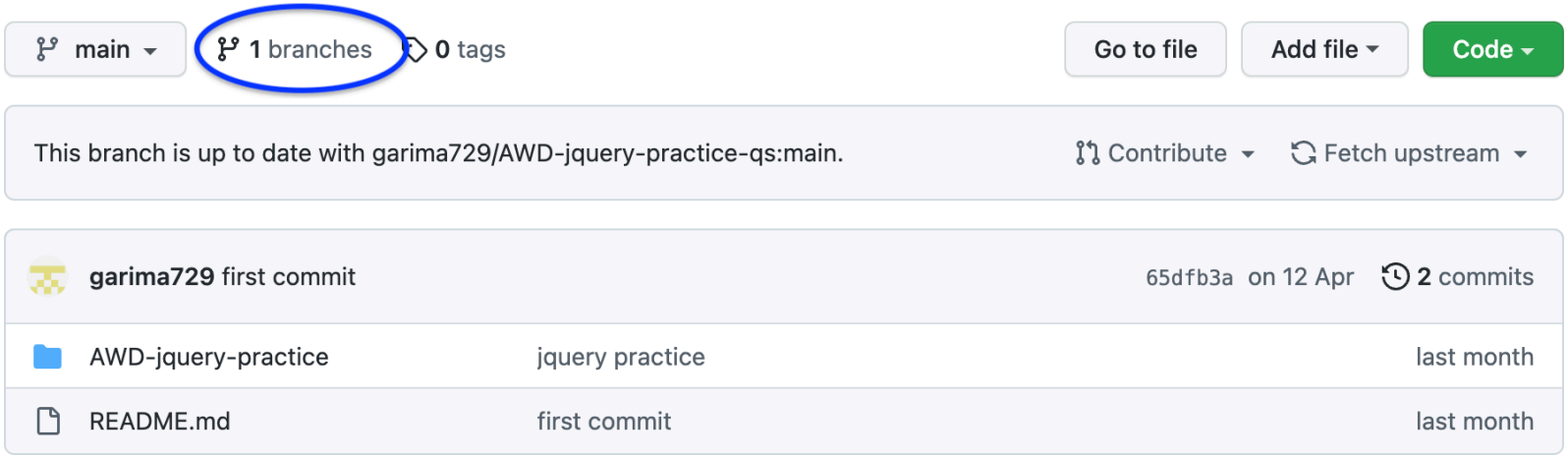
Now we have to clone that repo to make changes in that and to commit and make a pull request.

```
dikshamalik@Dikshas-Air AWD-jquery-practice % git branch -a
* main
  remotes/origin/HEAD -> origin/main
  remotes/origin/main
dikshamalik@Dikshas-Air AWD-jquery-practice % git checkout feature
error: pathspec 'feature' did not match any file(s) known to git.
dikshamalik@Dikshas-Air AWD-jquery-practice % git checkout -b feature
Switched to a new branch 'feature'
dikshamalik@Dikshas-Air AWD-jquery-practice % git log --oneline
cb2e867 (HEAD -> feature, main) commit after forking
65dfb3a (origin/main, origin/HEAD) first commit
ff8616b jquery practice
```

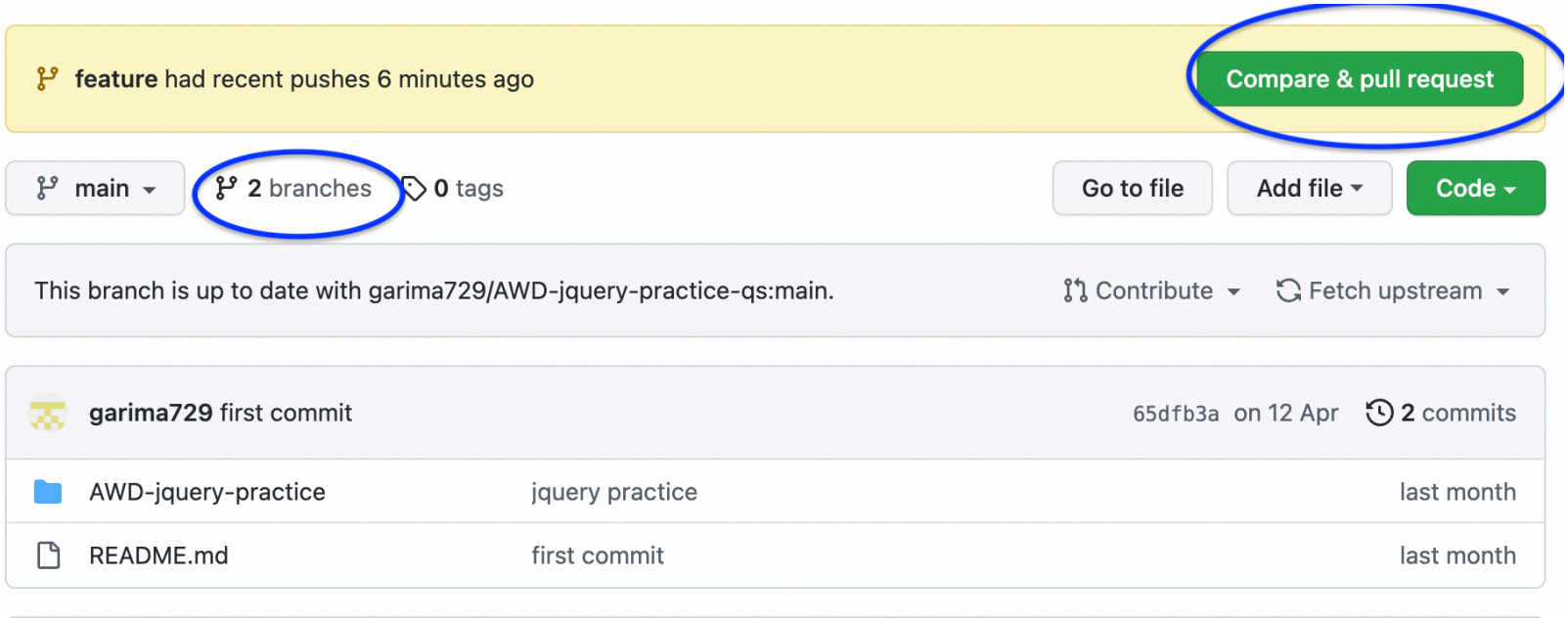
Now create a new branch and make changes in it and them commit it then push it to the Remote repository .

```
dikshamalik@Dikshas-Air AWD-jquery-practice % git branch
* feature
  main
dikshamalik@Dikshas-Air AWD-jquery-practice % vi file.txt
dikshamalik@Dikshas-Air AWD-jquery-practice % git add .
dikshamalik@Dikshas-Air AWD-jquery-practice % git commit -m" modified "
[feature 0a8a79c]  modified
 1 file changed, 1 insertion(+)
 create mode 100644 AWD-jquery-practice/file.txt
dikshamalik@Dikshas-Air AWD-jquery-practice % git push -u origin feature
Counting objects: 8, done.
Delta compression using up to 8 threads.
Compressing objects: 100% (7/7), done.
Writing objects: 100% (8/8), 767 bytes | 767.00 KiB/s, done.
Total 8 (delta 2), reused 0 (delta 0)
remote: Resolving deltas: 100% (2/2), completed with 1 local object.
remote:
remote: Create a pull request for 'feature' on GitHub by visiting:
remote:      https://github.com/diksha6036/AWD-jquery-practice-qs/pull/new/feature
remote:
To https://github.com/diksha6036/AWD-jquery-practice-qs.git
 * [new branch]      feature -> feature
Branch 'feature' set up to track remote branch 'feature' from 'origin'.
dikshamalik@Dikshas-Air AWD-jquery-practice %
```

And now the forked repost which was having only 1 branch now have 2 branches .



Now to make a pull request go on compare & pull request .



Now proceed to make a pull request .

base repository: garima729/AWD-jquery-prac... | base: main | head repository: diksha6036/AWD-jquery-pra... | compare: feature

✓ **Able to merge.** These branches can be automatically merged.

Feature

Write | Preview

Leave a comment

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

Allow edits by maintainers

Create pull request

Helpful resources

GitHub Community Guidelines

Remember, contributions to this repository should follow our GitHub Community Guidelines.

Commits on May 18, 2022

commit after forking
diksha6036 committed 28 minutes ago — cb2e867

modified
diksha6036 committed 18 minutes ago — 0a8a79c

Showing 2 changed files with **3 additions** and **0 deletions**.

Split | Unified

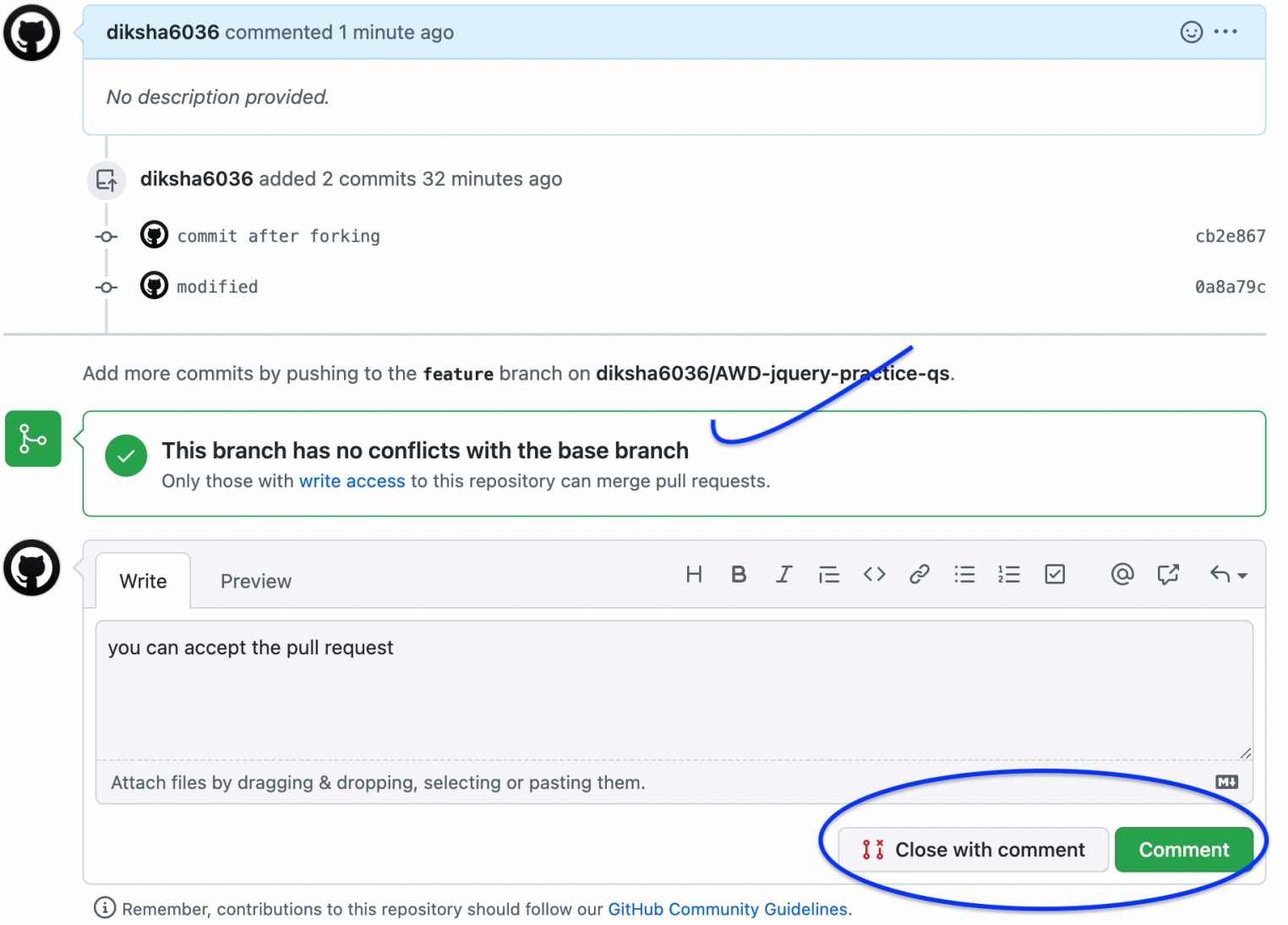
1 AWD-jquery-practice/file.txt

```
@@ -0,0 +1 @@
+       Just triying to modifing and the commit
```

2 AWD-jquery-practice/new.html

```
@@ -0,0 +1,2 @@
+ <!-- this is just a demo for commiting in forked repo -->
+
```

Then you'll be having a message that whether this branch have conflict with base branch or not or you can comment as well .



If the repo owner will accept the merge request our changes will be published in the main repo .

➡ Merge and Resolve conflicts created due to own activity and collaborators activity:-

In a repo if we and our collaborators make some changes . And if 2 or more collaborators make the changes in same lines of codes so here a conflict will be created which have to resolve . They have to finalised the best version of that code and select that and the selected one will be the final and committed.

Steps to get a merge conflict :

Step 1:

Create a file . For eg - index.html
And push that to your Github repo.

```
dikshamalik@Dikshas-Air Team07_project % touch index.html
```

Step 2:

Create a new branch , make some changes in it and commit and push this as well. Now your repo will be having 2 branches.

Below you can see the whole process from creating a branch to pushing on GitHub:

```
dikshamalik@Dikshas-Air Team07_project % git checkout -b feature
Switched to a new branch 'feature'
dikshamalik@Dikshas-Air Team07_project % git branch
* feature
  master
dikshamalik@Dikshas-Air Team07_project % git status
On branch feature
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   index.html

no changes added to commit (use "git add" and/or "git commit -a")
dikshamalik@Dikshas-Air Team07_project % git add .
dikshamalik@Dikshas-Air Team07_project % git commit -m"added Engineer"
[feature d055c49] added Engineer
 1 file changed, 1 insertion(+), 1 deletion(-)
dikshamalik@Dikshas-Air Team07_project % git push origin feature
Counting objects: 3, done.
Delta compression using up to 8 threads.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 290 bytes | 290.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
remote:
remote: Create a pull request for 'feature' on GitHub by visiting:
remote:      https://github.com/diksha6036/Team07/pull/new/feature
remote:
To https://github.com/diksha6036/Team07.git
 * [new branch]      feature -> feature
```

Step 3:

Now again switch to master branch and the most important thing to remember is to create a conflict make change in the same lines of code you did in another branch (feature in my case). And push the change .

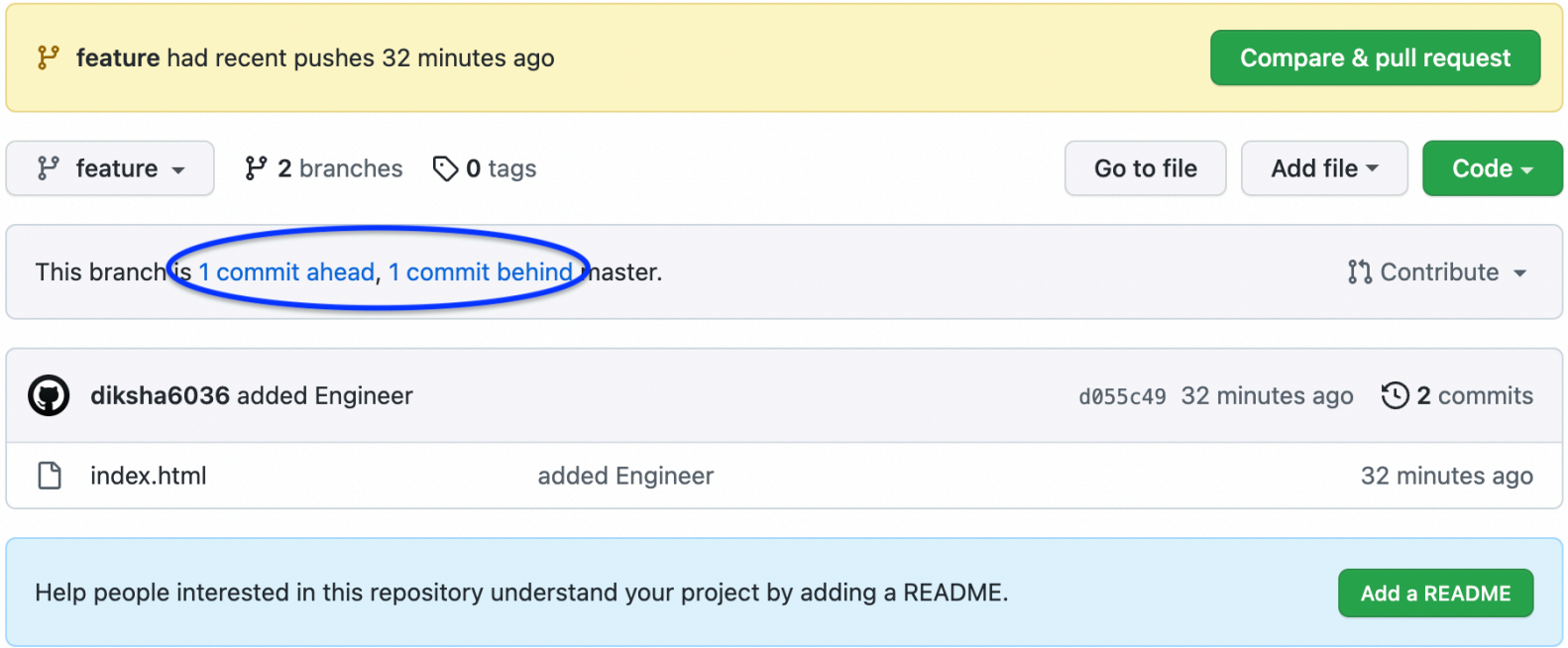
```
dikshamalik@Dikshas-Air Team07_project % git checkout master
Switched to branch 'master'
Your branch is up to date with 'origin/master'.
dikshamalik@Dikshas-Air Team07_project % git status
On branch master
Your branch is up to date with 'origin/master'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   index.html

no changes added to commit (use "git add" and/or "git commit -a")
dikshamalik@Dikshas-Air Team07_project % git add .
dikshamalik@Dikshas-Air Team07_project % git commit -m"added software"
[master 69cb931] added software
 1 file changed, 1 insertion(+), 1 deletion(-)
dikshamalik@Dikshas-Air Team07_project % git push origin master
Counting objects: 3, done.
Delta compression using up to 8 threads.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 301 bytes | 301.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/diksha6036/Team07.git
   ee30ede..69cb931  master -> master
```

On your Github desktop repo you'll be able to see a the changes you made after switching to feature branch .



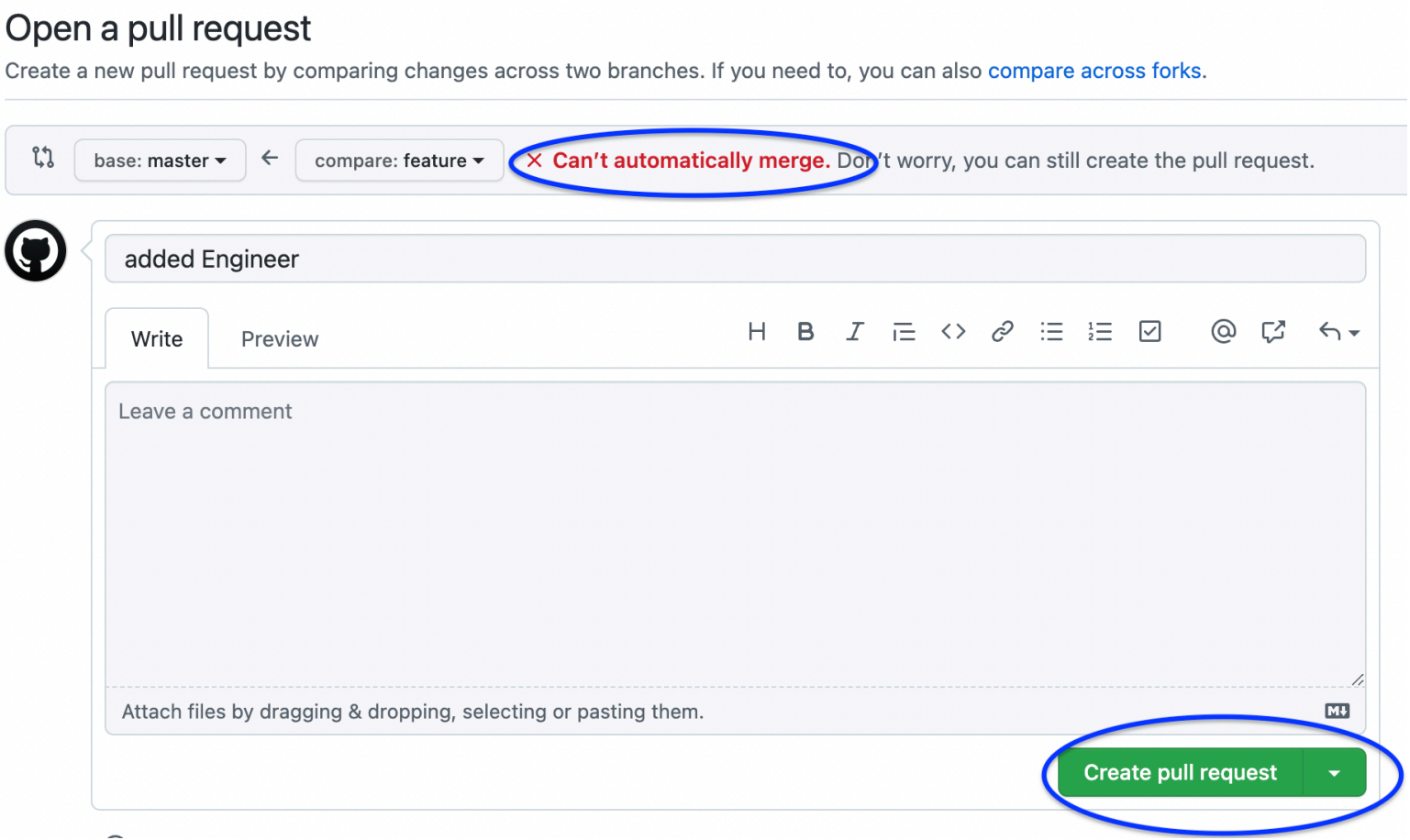
After clicking the highlighted part you can see the changes you made .

2 index.html

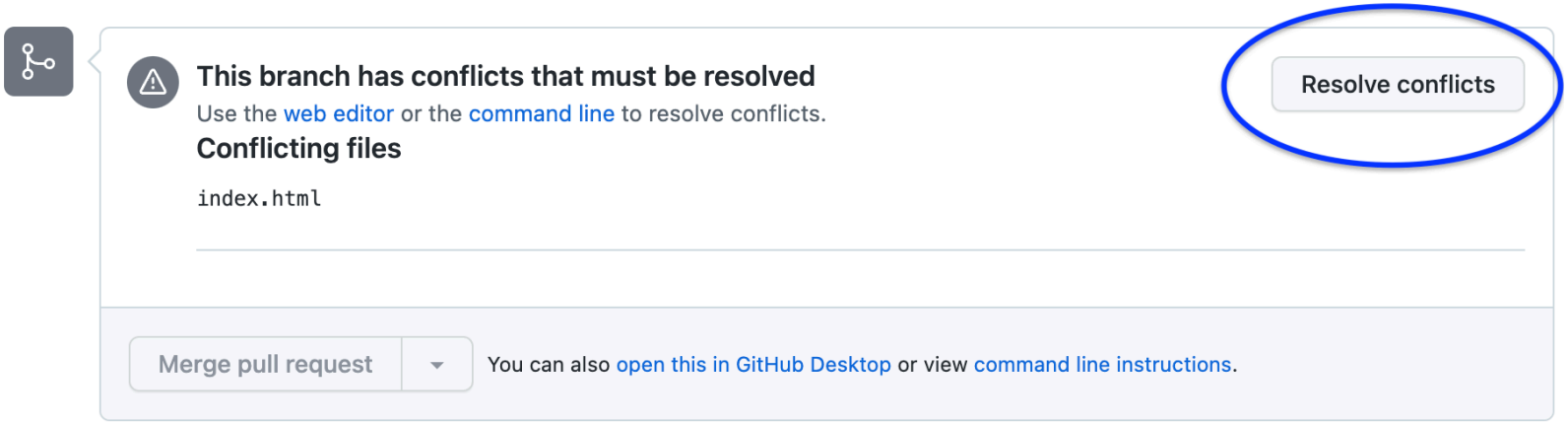
```
@@ -9,7 +9,7 @@
</head>

<body>
-    <h2>APPLY FOR JOB</h2>
+    <h2>APPLY FOR Software Engineers Jobs</h2>
     <form>
         <lable for="name">
             First Name:   <input type="text" name="name" id="name" placeholder="Your Name" required></lable>
```

After clicking the pull request option you'll able to see -



So now you can see the notification that our branch is having a conflict that must be resolved



After clicking resolve conflict it will show a page on which we have to finalise that which code we want to have .

index.html — 1 conflict — Prev — Next — Mark as resolved

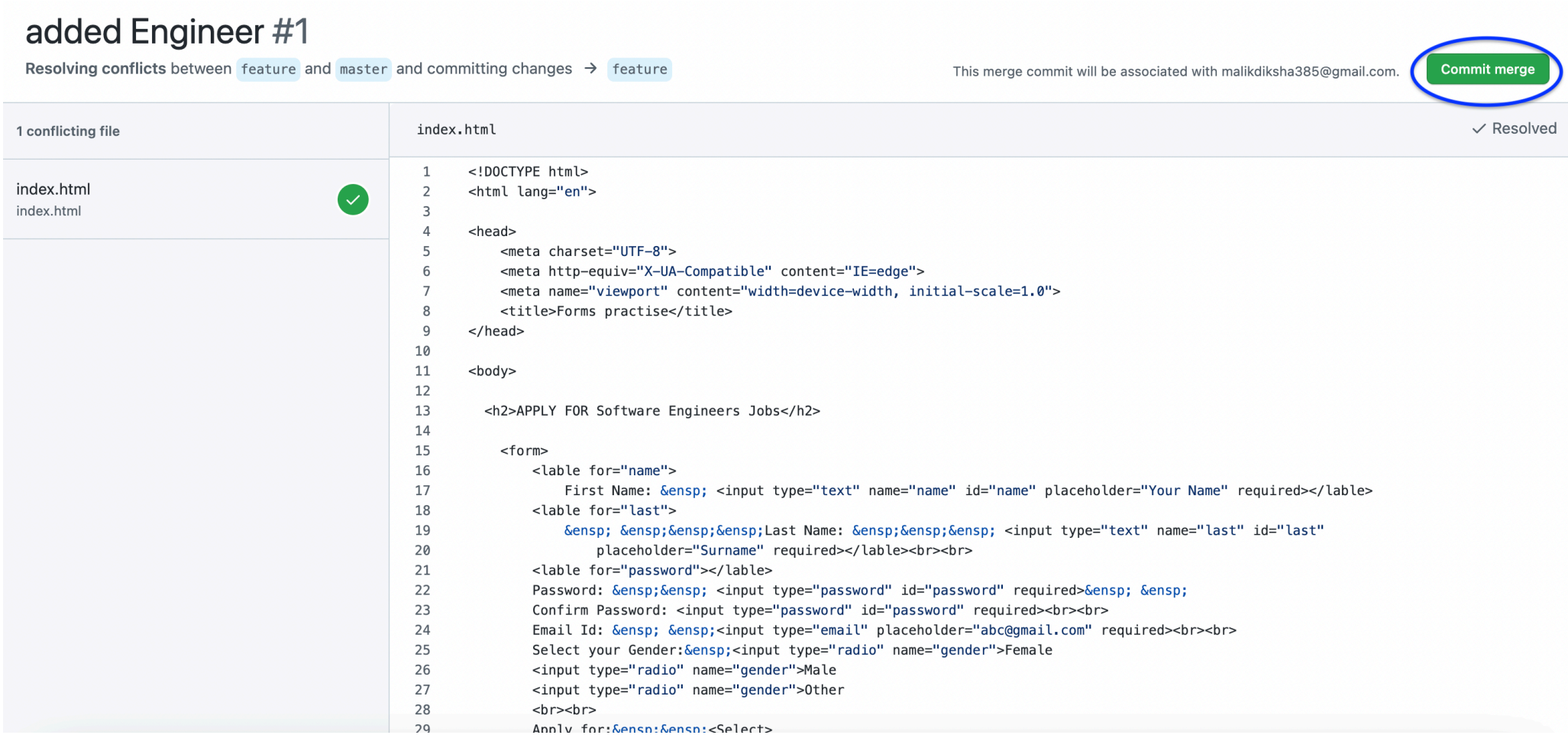
```
1   <!DOCTYPE html>
2   <html lang="en">
3
4   <head>
5       <meta charset="UTF-8">
6       <meta http-equiv="X-UA-Compatible" content="IE=edge">
7       <meta name="viewport" content="width=device-width, initial-scale=1.0">
8       <title>Forms practise</title>
9   </head>
10
11  <body>
12  <<<<<<< feature
13      <h2>APPLY FOR Engineering JOB</h2>
14  =======
15      <h2>APPLY FOR Software Engineers Jobs</h2>
16  >>>>>>> master
17      <form>
```

So here I want to keep the change that I've made in master branch so I'll delete the conflict part of feature branch like this -

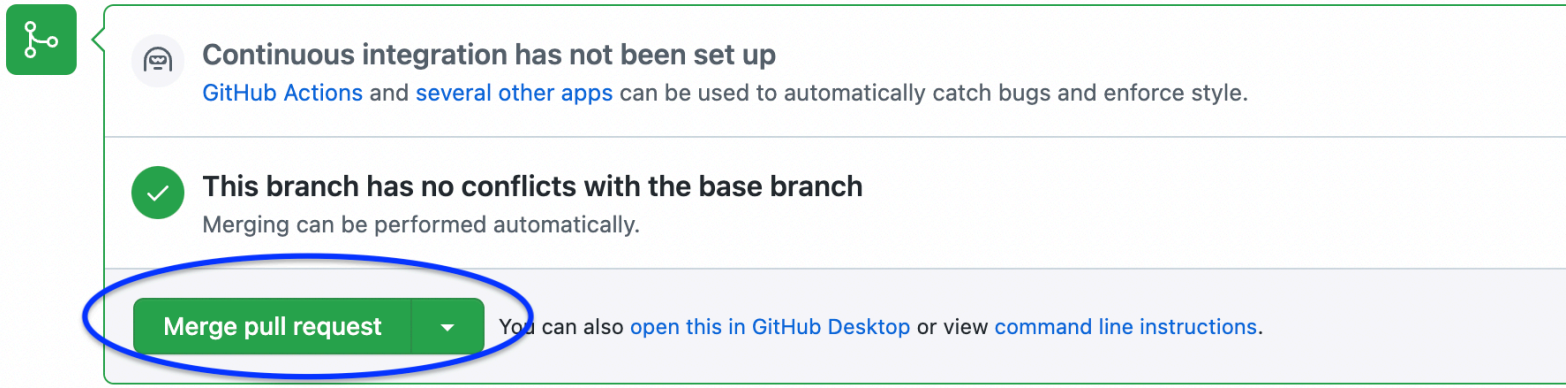
```
11      <body>
12  ┌── <<<<<<< feature
13  │       <h2>APPLY FOR Engineering JOB</h2>
14  ├── =======
15  │       <h2>APPLY FOR Software Engineers Jobs</h2>
16  └── >>>>>>> master
17          <form>
```

After removing the unwanted part our conflict get resolved and you can freely click on mark as resolved.

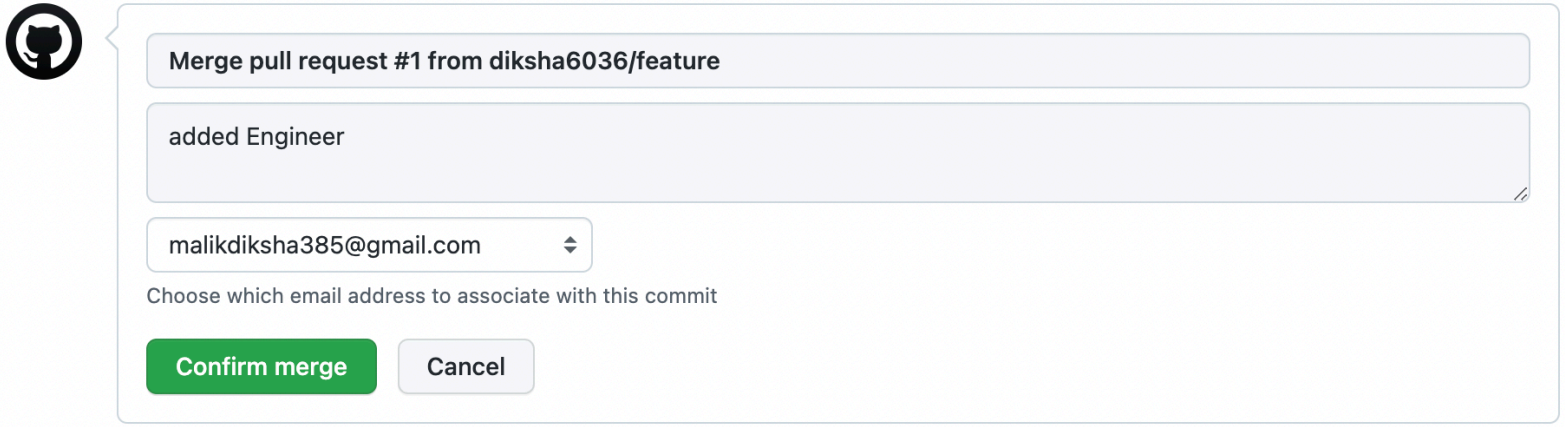
So now the commit you finalise have to merge :



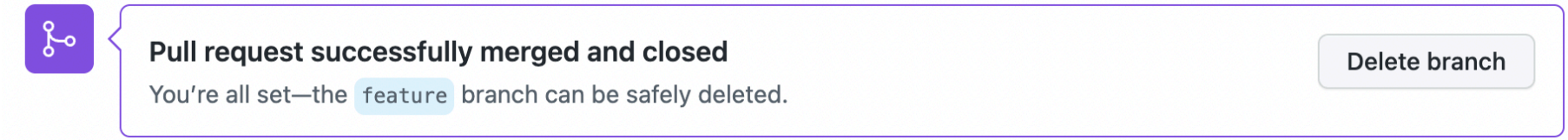
After merging you have to create a merge pull request :



Confirm your merge request :



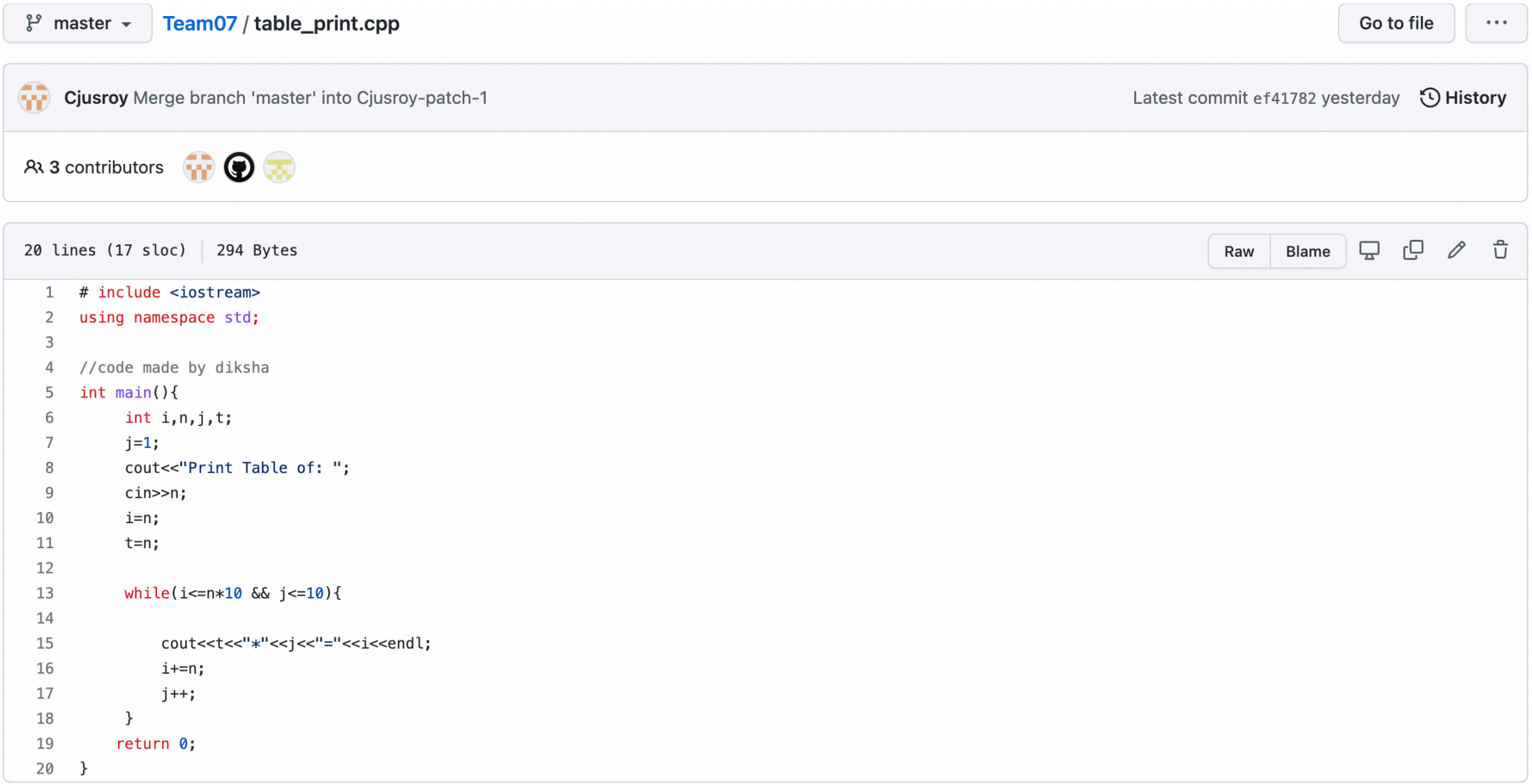
And yes you have successfully resolved a merge conflict .
Now if you want to delete the branch because the changes have been merged you can freely delete that branch .



Its not necessary to solve the conflict in this way there are many other ways in which you can solve a merge conflict .
On the CLI if you'll try to delete a branch without merging it, it will show an error .

Now the collaborators I've added before in my repository created some merge conflicts in the Repository .

1. Initially I've pushed a cpp code in my repo :



master Team07 / table_print.cpp Go to file ···

Cjusroy Merge branch 'master' into Cjusroy-patch-1 Latest commit ef41782 yesterday History

3 contributors

20 lines (17 sloc) | 294 Bytes Raw Blame

```
1  # include <iostream>
2  using namespace std;
3
4  //code made by diksha
5  int main(){
6       int i,n,j,t;
7       j=1;
8       cout<<"Print Table of: ";
9       cin>>n;
10      i=n;
11      t=n;
12
13      while(i<=n*10 && j<=10){
14
15          cout<<t<<"*"<<j<<"="<<i<<endl;
16          i+=n;
17          j++;
18      }
19     return 0;
20 }
```

2. The collaborators made some changes in the same lines of code :

```
1  # include <iostream>
2  using namespace std;
3
4  //code made by diksha
5  int main(){
6       int i,n,j,t;
7       j=1;
8       cout<<"Print Table of: ";
9       cin>>n;
10      i=n;
11      t=n;
12
13      while(i<=n*12 && j<=12){
14
15          cout<<t<<"*"<<j<<"="<<i<<endl;
16          i+=n;
17          j++;
18      }
19     return 0;
20 }
```

```
1  # include <iostream>
2  using namespace std;
3
4  //code made by diksha
5  int main(){
6       int i,n,j,t;
7       j=1;
8       cout<<"Print Table of: ";
9       cin>>n;
10      i=n;
11      t=n;
12
13      while(i<=n*14 && j<=14){
14
15          cout<<t<<"*"<<j<<"="<<i<<endl;
16          i+=n;
17          j++;
18      }
19     return 0;
20 }
```

3. I got an email as well as a notification on GitHub that there is a conflict that must be resolved .

CJ

Chaarvi Jusroy

Fwd: [Cjusroy/MyCodez] Feature (PR #1)

To: diksha0446.be21@chitkara.edu.in

---------- Forwarded message ---------
From: **Diksha** <notifications@github.com>
Date: Sat, May 28, 2022 at 1:43 PM
Subject: [Cjusroy/MyCodez] Feature (PR #1)
To: Cjusroy/MyCodez <MyCodez@noreply.github.com>
Cc: Subscribed <subscribed@noreply.github.com>

You can view, comment on, or merge this pull request online at:

https://github.com/Cjusroy/MyCodez/pull/1

Commit Summary

- 5c565c5 added comment
- 07eea50 editted again

File Changes (2 files)

- **A** Fibonacci (0)
- **M** Reverse of a number (2)

Patch Links:

- https://github.com/Cjusroy/MyCodez/pull/1.patch
- https://github.com/Cjusroy/MyCodez/pull/1.diff

—
Reply to this email directly, view it on GitHub, or unsubscribe.
You are receiving this because you are subscribed to this thread.

G

garima729

[diksha6036/Team07] Update table_print.cpp (PR #3)

To: diksha6036/Team07, Cc: Subscribed,

Reply-To: diksha6036/Team07

This message is from a mailing list.

You can view, comment on, or merge this pull request online at:

https://github.com/diksha6036/Team07/pull/3

Commit Summary

- ef83ff3 Update table_print.cpp

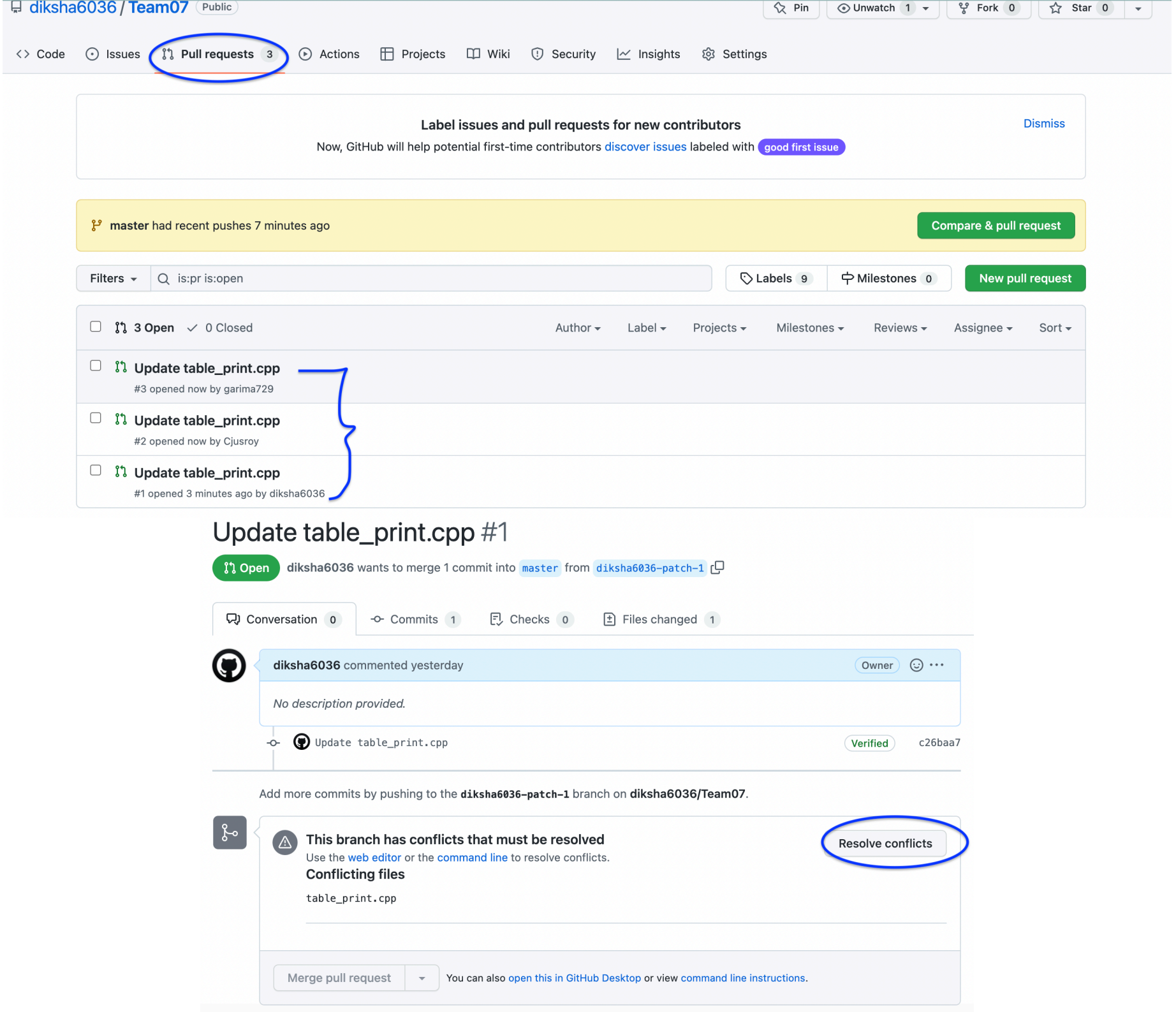
File Changes (1 file)

- **M** table_print.cpp (2)

Patch Links:

- https://github.com/diksha6036/Team07/pull/3.patch
- https://github.com/diksha6036/Team07/pull/3.diff

—
Reply to this email directly, view it on GitHub, or unsubscribe.
You are receiving this because you are subscribed to this thread.


diksha6036 / Team07
Public
Pin
Unwatch 1
Fork 0
Star 0
Code
Issues
Pull requests 3
Actions
Projects
Wiki
Security
Insights
Settings
Label issues and pull requests for new contributors
Now, GitHub will help potential first-time contributors discover issues labeled with good first issue
Dismiss
master had recent pushes 7 minutes ago
Compare & pull request
Filters
is:pr is:open
Labels 9
Milestones 0
New pull request
3 Open
0 Closed
Author
Label
Projects
Milestones
Reviews
Assignee
Sort
Update table_print.cpp
#3 opened now by garima729
Update table_print.cpp
#2 opened now by Cjusroy
Update table_print.cpp
#1 opened 3 minutes ago by diksha6036
Update table_print.cpp #1
Open
diksha6036 wants to merge 1 commit into master from diksha6036-patch-1
Conversation 0
Commits 1
Checks 0
Files changed 1
diksha6036 commented yesterday
Owner
No description provided.
Update table_print.cpp
Verified
c26baa7
Add more commits by pushing to the diksha6036-patch-1 branch on diksha6036/Team07.
This branch has conflicts that must be resolved
Use the web editor or the command line to resolve conflicts.
Resolve conflicts
Conflicting files
table_print.cpp
Merge pull request
You can also open this in GitHub Desktop or view command line instructions.

After clicking on resolve conflict I'll be able to see the conflicts created by the collaborators and I've to finalise the code which I want to have in my repo .

Update table_print.cpp #1

Resolving conflicts between `diksha6036-patch…` and `master` and committing changes → `diksha6036-patch…`

1 conflicting file

table_print.cpp
table_print.cpp

table_print.cpp — 1 conflict — Prev — Next — Mark as resolved

```
1   # include <iostream>
2   using namespace std;
3
4   //code made by diksha
5   int main(){
6        int i,n,j,t;
7        j=1;
8        cout<<"Print Table of: ";
9        cin>>n;
10       i=n;
11       t=n;
12
13  <<<<<<< Cjusroy-patch-1
14       while(i<=n*12 && j<=12){
15  =======
16       while(i<=n*13 && j<=13){
17
18  >>>>>>> master
19            cout<<t<<"*"<<j<<"="<<i<<endl;
20            i+=n;
21            j++;
22       }
23      return 0;
24  }
25  |
```

Simply resolve the conflict by removing the lines you don’t to be a part of the code and marks the Conflict as resolved

table_print.cpp

```
1    # include <iostream>
2    using namespace std;
3
4    //code made by diksha
5    int main(){
6         int i,n,j,t;
7         j=1;
8         cout<<"Print Table of: ";
9         cin>>n;
10        i=n;
11        t=n;
12
13   <<<<<<< diksha6036-patch-1
14        while(i<=n*10 && j<=10){
15   =======
16        while(i<=n*14 && j<=14){
17
18   >>>>>>> master
19             cout<<t<<"*"<<j<<"="<<i<<endl;
20             i+=n;
21             j++;
22        }
23       return 0;
24   }
25
```

table_print.cpp

```
1    # include <iostream>
2    using namespace std;
3
4    //code made by diksha
5    int main(){
6         int i,n,j,t;
7         j=1;
8         cout<<"Print Table of: ";
9         cin>>n;
10        i=n;
11        t=n;
12
13   <<<<<<< Cjusroy-patch-1
14        while(i<=n*12 && j<=12){
15   =======
16        while(i<=n*13 && j<=13){
17
18   >>>>>>> master
19             cout<<t<<"*"<<j<<"="<<i<<endl;
20             i+=n;
21             j++;
22        }
23       return 0;
24   }
25
```

Update table_print.cpp #1

Resolving conflicts between `diksha6036-patch…` and `master` and committing changes → `diksha6036-patch…`

1 conflicting file

table_print.cpp
table_print.cpp

table_print.cpp — 1 conflict — Prev — Next — Mark as resolved

```
1   # include <iostream>
2   using namespace std;
3
4   //code made by diksha
5   int main(){
6        int i,n,j,t;
7        j=1;
8        cout<<"Print Table of: ";
9        cin>>n;
10       i=n;
11       t=n;
12
13
14       while(i<=n*10 && j<=10){
15           cout<<t<<"*"<<j<<"="<<i<<endl;
16           i+=n;
17           j++;
18       }
19      return 0;
20  }
21
```

Update table_print.cpp #1

Resolving conflicts between `diksha6036-patch…` and `master` and committing changes → `diksha6036-patch…`

This merge commit will be associated with malikdiksha385@gmail.com. Commit merge

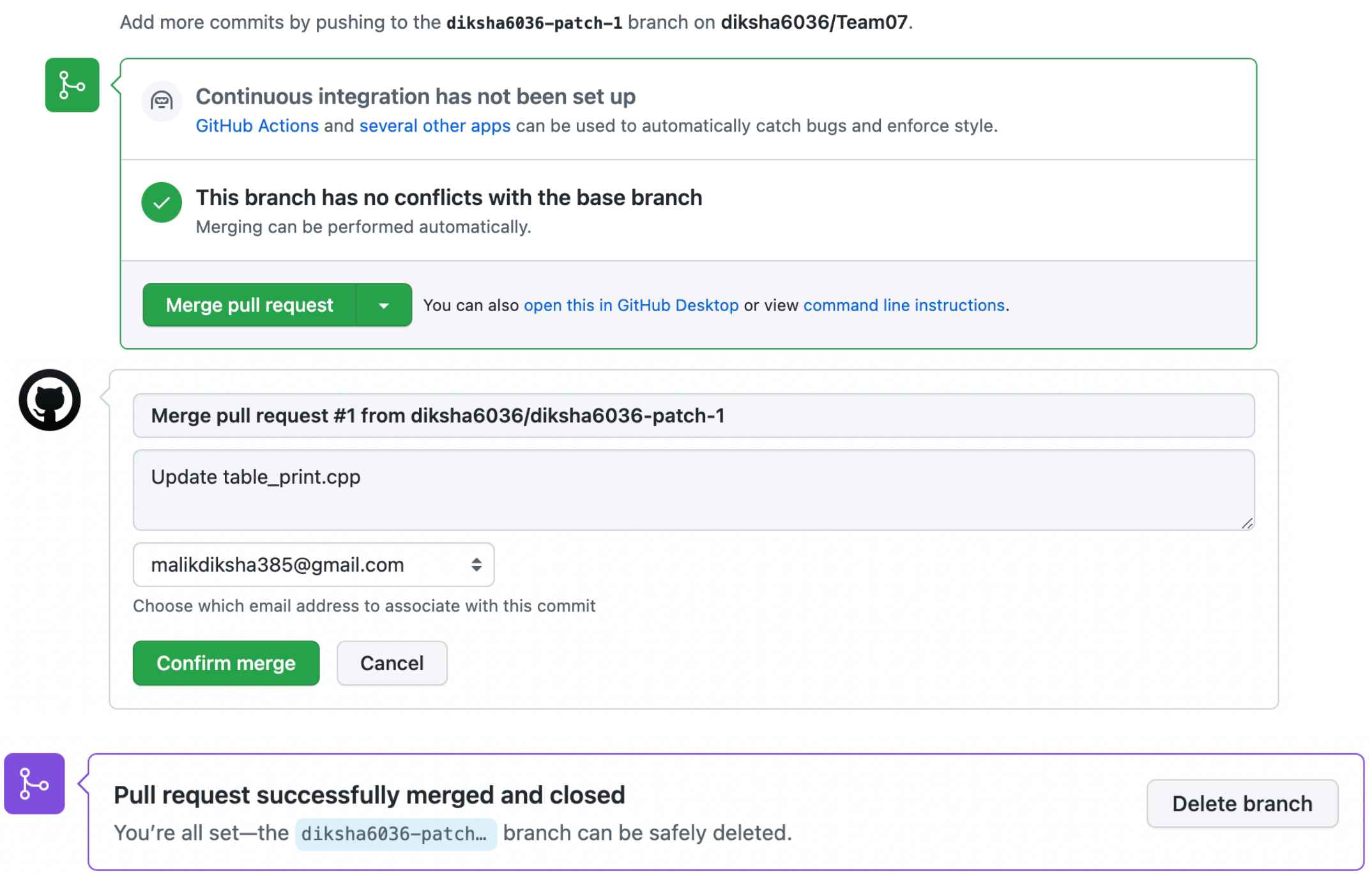
1 conflicting file

table_print.cpp
table_print.cpp

table_print.cpp — ✓ Resolved

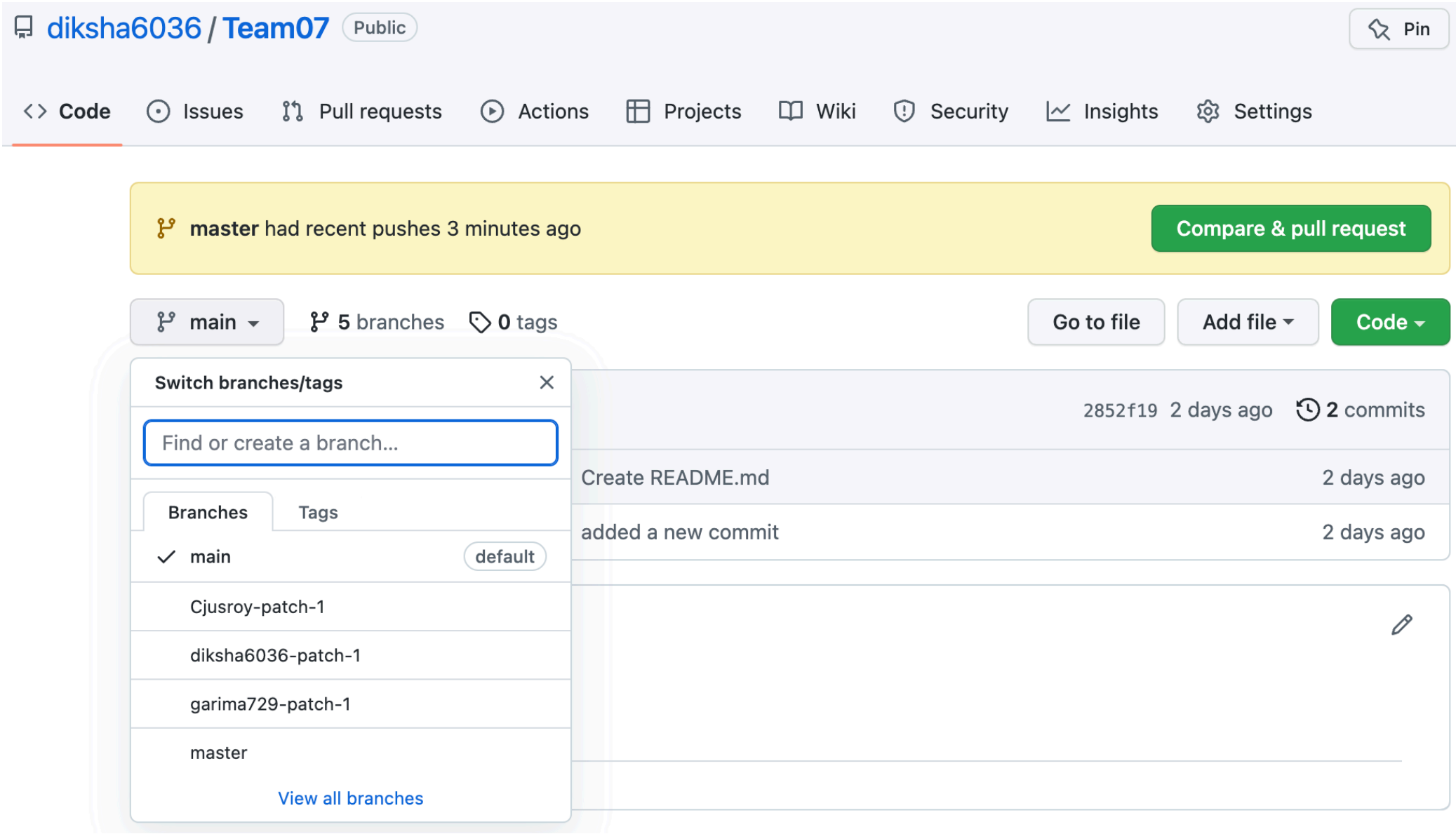
```
1   # include <iostream>
2   using namespace std;
3
4   //code made by diksha
5   int main(){
6        int i,n,j,t;
7        j=1;
8        cout<<"Print Table of: ";
9        cin>>n;
10       i=n;
11       t=n;
12
13
14       while(i<=n*10 && j<=10){
15
16           cout<<t<<"*"<<j<<"="<<i<<endl;
17           i+=n;
18           j++;
19       }
20      return 0;
21  }
22
```

After clicking on commit merge we've to create a merge pull request



If you want to delete the merged branch you can freely delete that and if you want to keep you can .

After creating the merge conflicts our repo will be having more branches as a result of conflict so that if you want to see any particular change you can go to that change directly .



➡ Reset and Revert :-

Git reset is a powerful command that is used to undo local changes to the state of a Git repo.
Git reset operates on "The Three Trees of Git".
These trees are the Commit History (HEAD), the Staging Index, and the Working Directory.

Git reset can be used to unstage a file -

```
dikshamalik@Dikshas-Air reset_revet_demo % git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)

        diksha.txt

nothing added to commit but untracked files present (use "git add" to track)
dikshamalik@Dikshas-Air reset_revet_demo % git add diksha.txt
dikshamalik@Dikshas-Air reset_revet_demo % git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)

        new file:   diksha.txt

dikshamalik@Dikshas-Air reset_revet_demo % git reset .
dikshamalik@Dikshas-Air reset_revet_demo % git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)

        diksha.txt

nothing added to commit but untracked files present (use "git add" to track)
```

Git reset can be used to uncommit the changes but using git reset we can only reset the recents commits and only when the files are present in our local repository .
Removing specific commit is not possible with “git reset” -

```
dikshamalik@Dikshas-Air reset_revet_demo % git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)

        new file:   diksha.txt

dikshamalik@Dikshas-Air reset_revet_demo % git commit -m"1st commit"
[master (root-commit) eaf3697] 1st commit
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 diksha.txt
dikshamalik@Dikshas-Air reset_revet_demo % git status
On branch master
nothing to commit, working tree clean
dikshamalik@Dikshas-Air reset_revet_demo % git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   diksha.txt

no changes added to commit (use "git add" and/or "git commit -a")
dikshamalik@Dikshas-Air reset_revet_demo % git commit -a -m"2nd commit"
[master b25a8e0] 2nd commit
 1 file changed, 1 insertion(+)
dikshamalik@Dikshas-Air reset_revet_demo % git commit -a -m"3rd commit"
[master 5662557] 3rd commit
 1 file changed, 10 insertions(+), 1 deletion(-)
dikshamalik@Dikshas-Air reset_revet_demo % git log --oneline
5662557 (HEAD -> master) 3rd commit
b25a8e0 2nd commit
eaf3697 1st commit
```

Here we've created 3 commits So using Git reset command we'll uncommit that 3 changes.

Git reset have 3 options :

1. git reset --soft : When we run git reset with soft option the last one change is kept in staging area .

```
dikshamalik@Dikshas-Air reset_revet_demo % git reset --soft Head~1
dikshamalik@Dikshas-Air reset_revet_demo % git status
On branch master
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

        modified:   diksha.txt
```

2. git reset -- mixed : This is also a default option with git . This command removes the recent commit .

```
dikshamalik@Dikshas-Air reset_revet_demo % git reset --mixed Head~1
Unstaged changes after reset:
M       diksha.txt
dikshamalik@Dikshas-Air reset_revet_demo % git log --oneline
b25a8e0 (HEAD -> master) 2nd commit
eaf3697 1st commit
```

3. git reset -- hard: This commands removes the recently changes and it won't keep those changes in index or working area .

```
dikshamalik@Dikshas-Air reset_revet_demo % git reset --hard Head~1
HEAD is now at b25a8e0 2nd commit
dikshamalik@Dikshas-Air reset_revet_demo % git log --oneline
b25a8e0 (HEAD -> master) 2nd commit
eaf3697 1st commit
```

After using git reset -- hard we'll not be able to get that changes back if we want.

Git revert command is used for undoing changes after pushing to the remote repo.

So now we have one commit in our repo :

```
dikshamalik@Dikshas-Air reset_revet_demo % git log --oneline
eaf3697 (HEAD -> master) 1st commit
```

Make an another commit :

```
dikshamalik@Dikshas-Air reset_revet_demo % git log --oneline
061cf36 (HEAD -> master) 2nd commit
eaf3697 1st commit
```

Now push that commit to the remote :

~ If we use reset command here for undoing changes it will undo the change only from the local repository not from the remote repo so that's why we've to use the revert command.

~ And also using revert command the limit is not upto only recently commits or only upto 3 or 4 commits we can undo the commits as many we want to make.

Revert command remove the changes in commit but don't remove the commit from the commit history and make a fresh commit to accommodate that reverts -

```
dikshamalik@Dikshats-Air reset_revet_demo % git log --oneline
061cf36 (HEAD -> master, origin/master) 2nd commit
eaf3697 1st commit
dikshamalik@Dikshas-Air reset_revet_demo % git revert 061cf36
[master ba9f6d7] Revert "2nd commit"
 1 file changed, 2 deletions(-)
```

After using revert command it will display a screen in which we can change the name of our fresh commit -

```
Revert "2nd commit"

This reverts commit 061cf36c77c358e4592bb6f87ca892f38e3c51fd.

# Please enter the commit message for your changes. Lines starting
# with '#' will be ignored, and an empty message aborts the commit.
#
# On branch master
# Changes to be committed:
#       modified:   diksha.txt
#
~
~
~
~
~
~
~
~
~
~
~
~
~
"~/Downloads/reset_revet_demo/.git/COMMIT_EDITMSG" 11L, 294B
```

Now check for commit history :

```
dikshamalik@Dikshas-Air reset_revet_demo % git log --oneline
ba9f6d7 (HEAD -> master) Revert "2nd commit"
061cf36 (origin/master) 2nd commit
eaf3697 1st commit
```