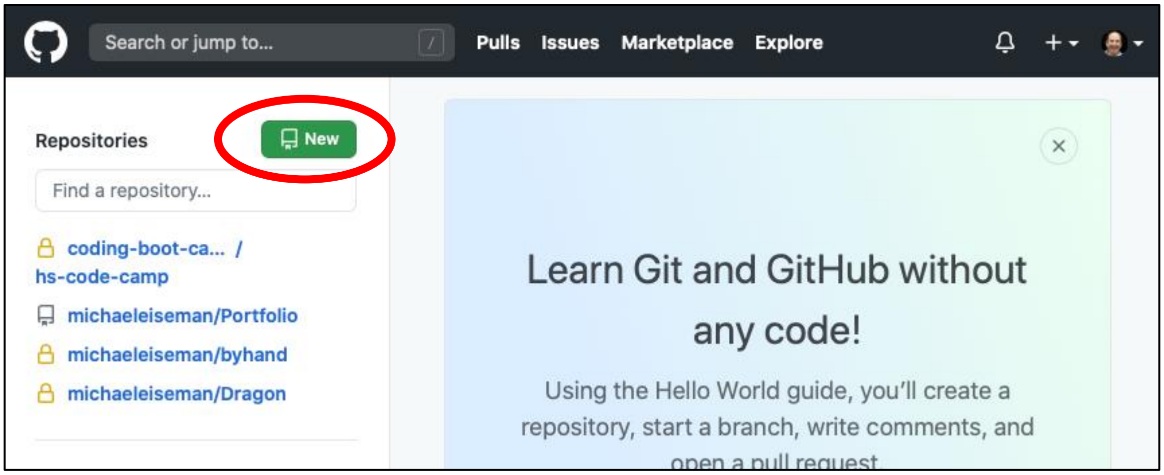
Using GitHub as a Team

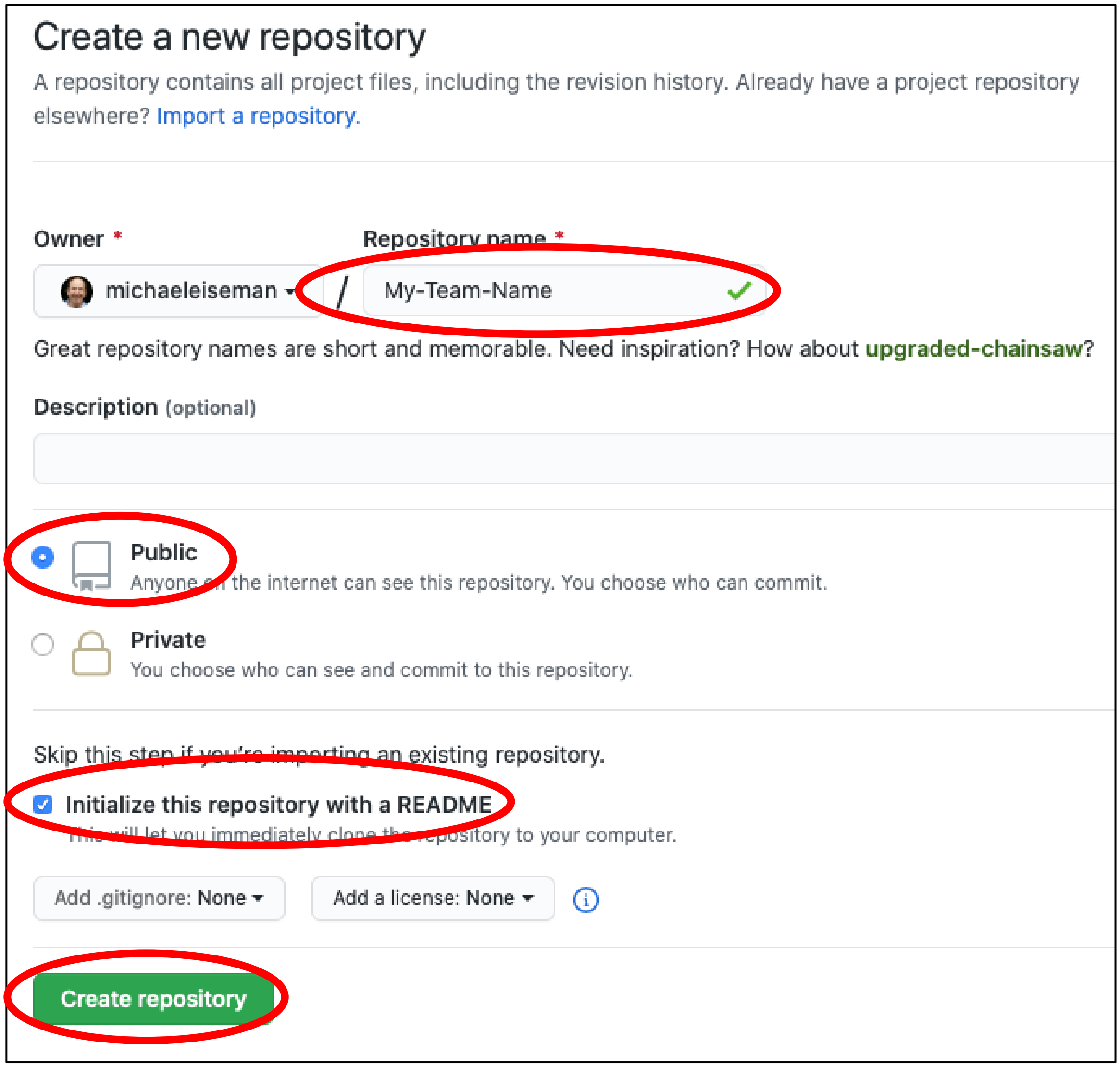
Steps for distributing the repository to all members on the team (you should only need to do this once)



Learn Git and GitHub without any code!

Using the Hello World guide, you'll create a repository, start a branch, write comments, and open a pull request

One team member should log into GitHub and click the green "New" button to create a NEW repository on their account while the other team members should slack their GitHub username to the team member that is creating the new repository.



Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? Import a repository.

Owner * michaeleiseman / Repository name * My-Team-Name ✓

Great repository names are short and memorable. Need inspiration? How about upgraded-chainsaw?

Description (optional)

● Public
Anyone on the internet can see this repository. You choose who can commit.

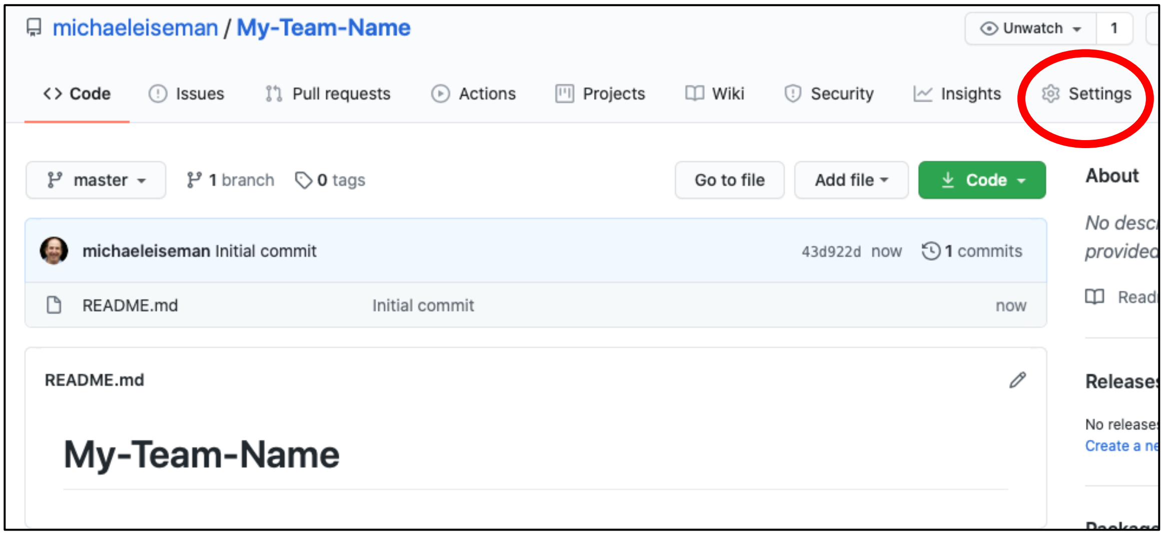
○ Private
You choose who can see and commit to this repository.

Skip this step if you're importing an existing repository.

☑ Initialize this repository with a README
This will let you immediately clone the repository to your computer.

Create the repository by

1. entering in the repository name of your choice (could be your team name),
2. making sure you have designated that you want the repository to be public,
3. selecting on the option to initialize the repository with a README,
4. clicking on the green "Create repository" button.



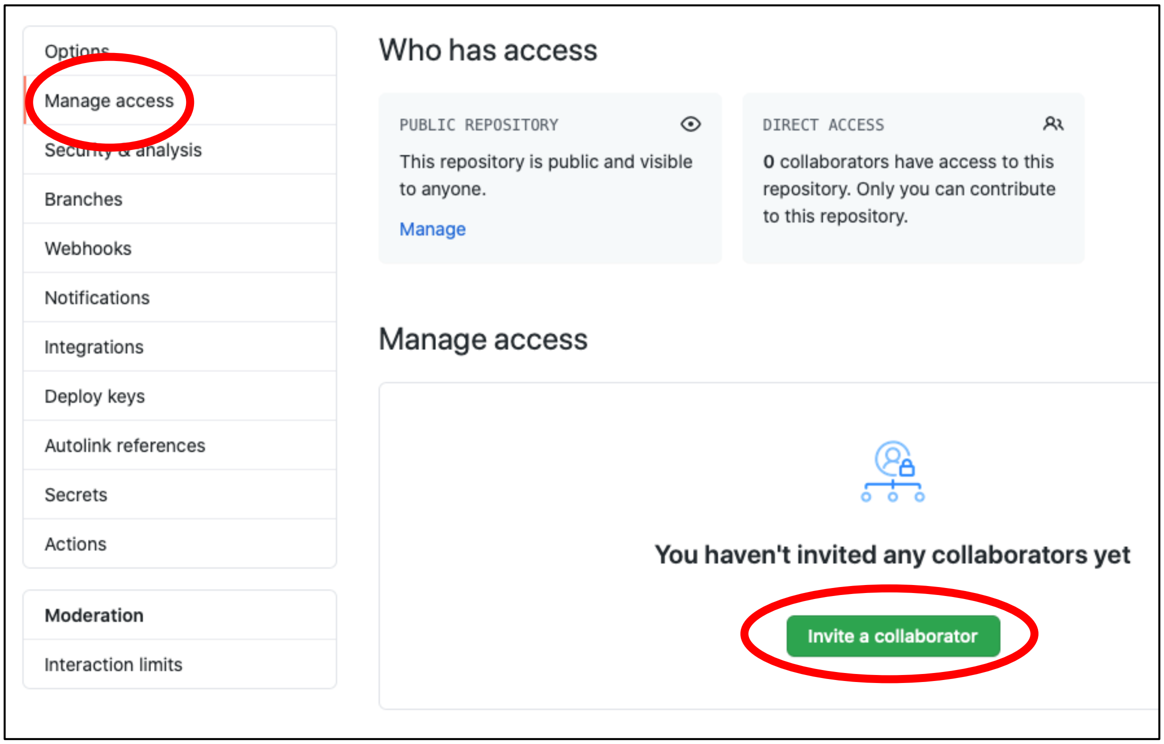
michaeleiseman Initial commit 43d922d now 1 commits

README.md Initial commit now

README.md

My-Team-Name

Next admit members to the repository by first clicking on the Settings gear



Who has access

PUBLIC REPOSITORY

This repository is public and visible to anyone.

DIRECT ACCESS

0 collaborators have access to this repository. Only you can contribute to this repository.

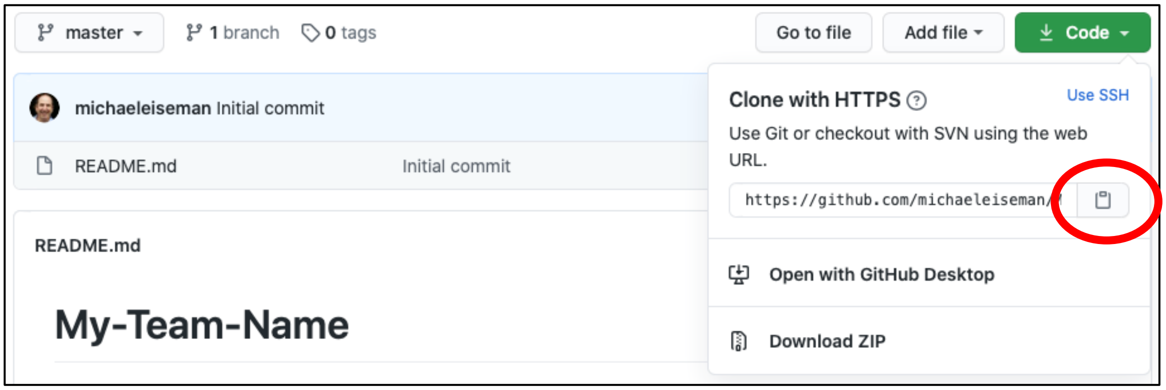
Manage access

You haven't invited any collaborators yet

Click on "Manage access" and then click on the green "Invite a collaborator" button.

Enter in and select each of your team members and add them as team members to your repository.

Each of your team members should get an email with a green View invitation button. They first must log into GitHub and then click on this button in their email. This should take them to a GitHub page with an Accept invitation button. Once they click on this button, they will be brought to the page containing the group's repository.



Use Git or checkout with SVN using the web URL.

```
https://github.com/michaeleiseman/
```

Every team member should click on the green "Code" button and then click on the clipboard icon to copy the web address of the repository to their clipboard.

With the web address on their clipboard, each team member should bring up the command line interface (Terminal for Mac and Git Bash for PC) and change directories to where you want your local copy of the project to reside. Note that it is NOT a good idea to put this project inside your Portfolio folder. You could perhaps make this project a sibling of your Portfolio folder if you like. This means that if you typed "ls", you would see your Portfolio folder listed.

After you have changed directories, type

```
    git clone paste-in-remote-directory-from-clipboard-here
```

This will put a copy of the current master branch for your project.

To start working on your project

First, you should make sure that your local copy of the master branch is up to date. To do this, open your terminal and cd to your local project folder. Type

```
git checkout main
git pull
```

Your local master copy should now be updated

It is best to create your own branch before you modify your team's project. To do this, type

```
git checkout -b type-your-branch-name-here
```

Note that the branch name may NOT include spaces.

Next, to make it easier to push your new branch up to the remote repository, you should type

```
git push --set-upstream origin type-your-branch-name-here
```

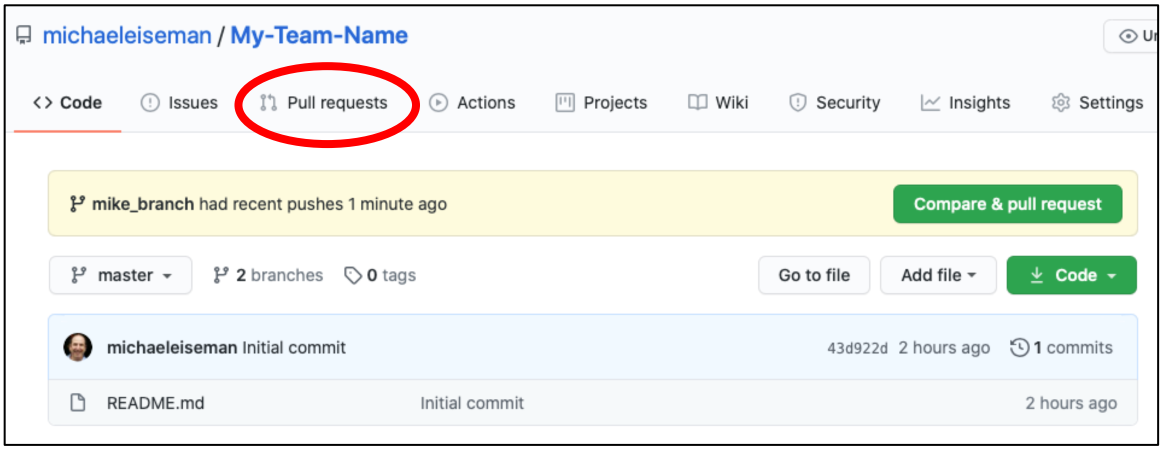
You should only have to do this last step once.

Now you can edit your new branch in Visual Studio Code just as you have been doing in class.

Steps to push your changes to the remote repository

Save all of your changes in Visual Studio Code while on your branch. Open your terminal and cd to your local project folder. Type

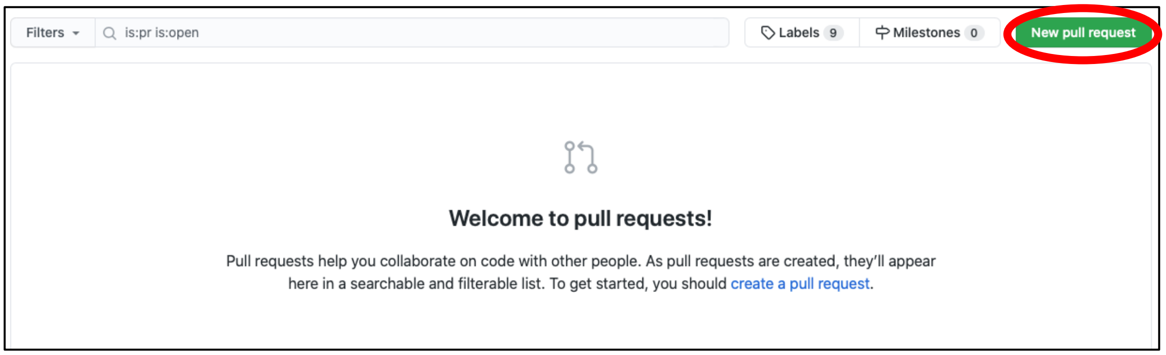
```
    git add .
    git commit -m "indicate changes here"
    git push
```



mike_branch had recent pushes 1 minute ago

michaeleiseman Initial commit

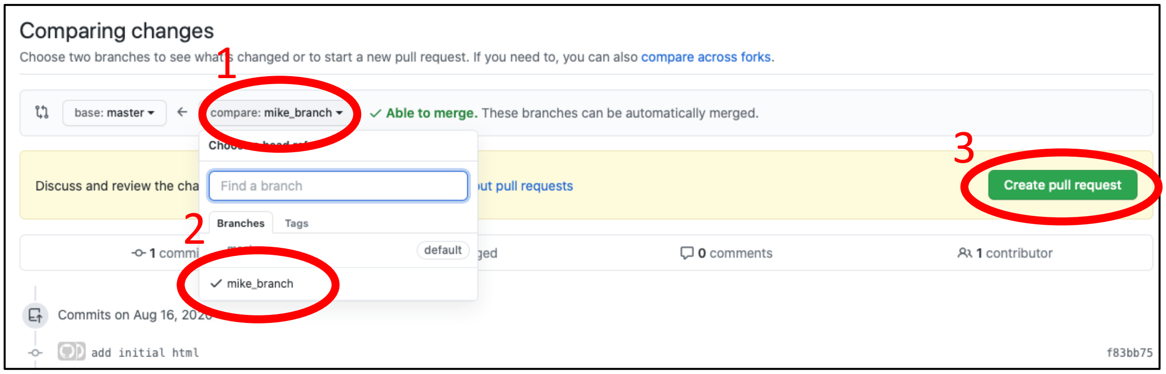
Now log into GitHub and submit a pull request by clicking on the navigation item labeled "Pull requests.



Welcome to pull requests!

Pull requests help you collaborate on code with other people. As pull requests are created, they'll appear here in a searchable and filterable list. To get started, you should create a pull request.

Now click on the green button labeled "New pull request."



Comparing changes

Choose two branches to see what's changed or to start a new pull request. If you need to, you can also compare across forks.

Able to merge. These branches can be automatically merged.

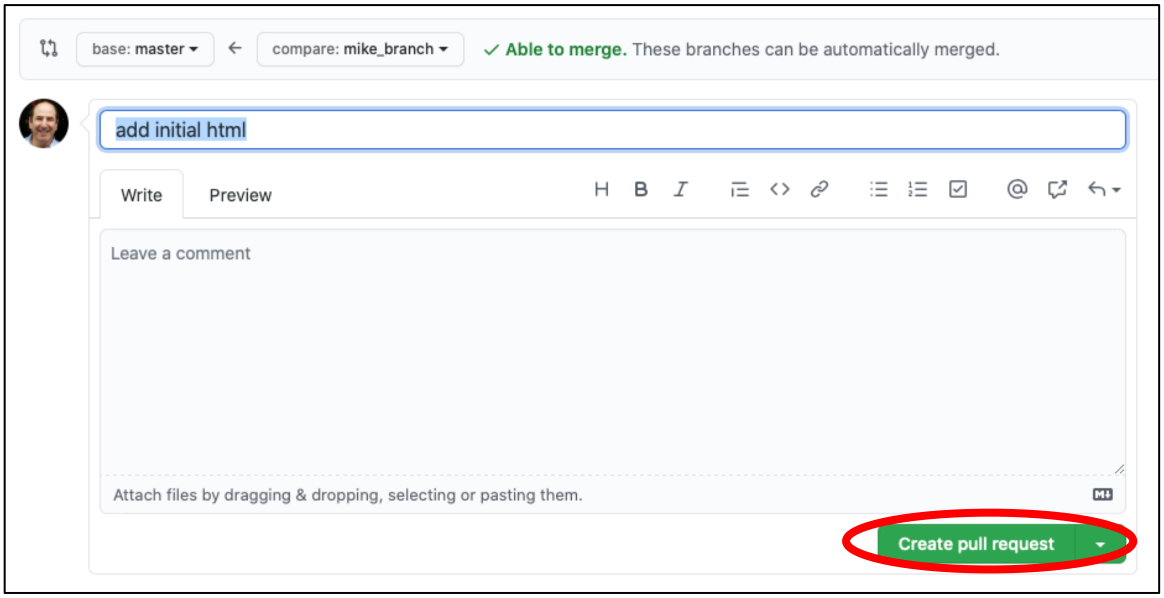
Discuss and review the cha

Commits on Aug 16, 2020

add initial html

Next

1. Click on the compare drop-down button,
2. Select the branch you want merged with the master branch, and
3. Click on the green "Create pull request" button.



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

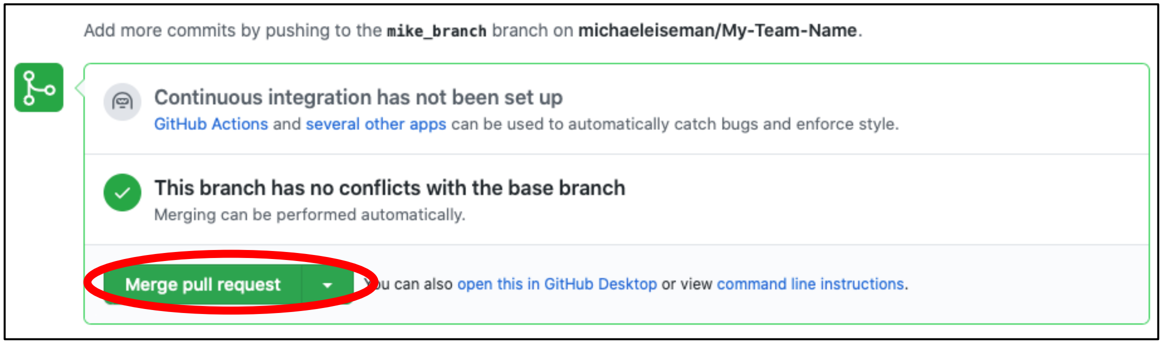
add initial html

Leave a comment

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

Click the green "Create pull request" button.

If there are no conflicts, you will see something like this:



Add more commits by pushing to the **mike_branch** branch on **michaeleiseman/My-Team-Name**.

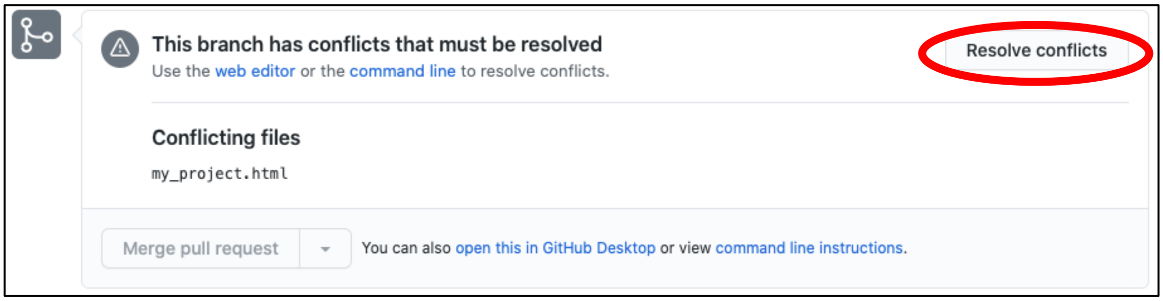
Continuous integration has not been set up
GitHub Actions and several other apps can be used to automatically catch bugs and enforce style.

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.

Click on the green "Merge pull request" button and then click on the **Confirm merge** button.

If there are conflicts, you will see something like this:



This branch has conflicts that must be resolved

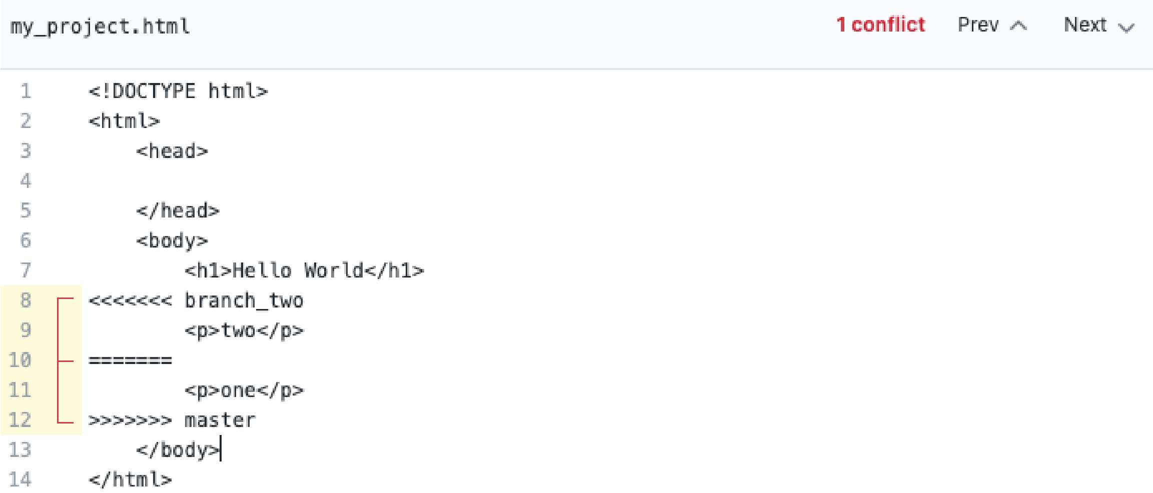
Use the web editor or the command line to resolve conflicts.

Conflicting files

my_project.html

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

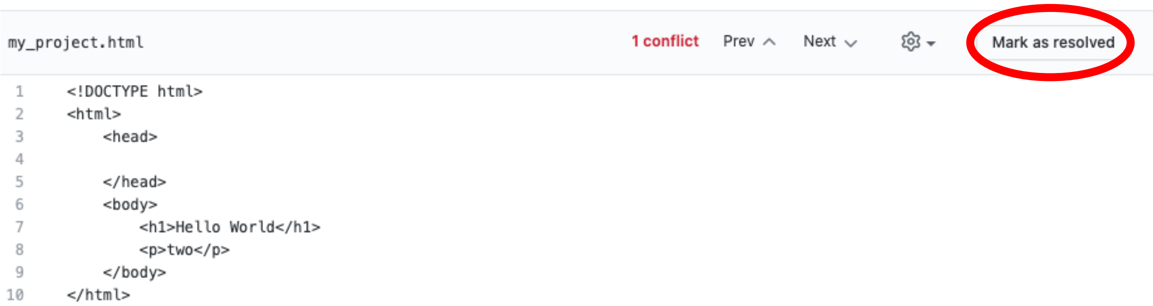
If this happens, click on the "Resolve conflicts" button and edit the code online until it looks the way you want it to look. You should also delete any markers that look like <<<<<<<, >>>>>>, or =======. For example, if your conflict looks like this:



```
<!DOCTYPE html>
<html>
    <head>

    </head>
    <body>
        <h1>Hello World</h1>
<<<<<<< branch_two
        <p>two</p>
=======
        <p>one</p>
>>>>>>> master
    </body>
</html>
```

And you wanted the <p> tag to contain the word "two", then you should edit the code in the web browser to look like this:



```
<!DOCTYPE html>
<html>
    <head>

    </head>
    <body>
        <h1>Hello World</h1>
        <p>two</p>
    </body>
</html>
```

Then click on the "Mark as resolved" button and then click on the green Commit merge button. Now you should be able to click on the green Merge pull request button and the Confirm merge button as shown above.

The master branch on GitHub should now include the new code you added in your branch.

You should now also update your local master branch by typing

```
      git checkout main
      git pull
```

in your terminal.

To see what branch you are currently working on

Open your terminal and cd to your local project folder. Type

```
     git branch
```

This will list all the branches you have locally and place an asterisk by the branch that you are currently on.

Steps to switch to a different local branch.

If you are currently editing a local branch and want to switch to a different local branch, first, make sure to save your work and commit your changes

```
     git add.
     git commit -m "change notes"
```

in your command line. Then type

```
     git checkout different-local-branch-name
```

To force a password request on next push:

```
git remote set-url origin
https://username@github.com/username/repo-name.git
```