Collaboration with GitHub and RStudio

Brooke Anderson

February 23, 2016

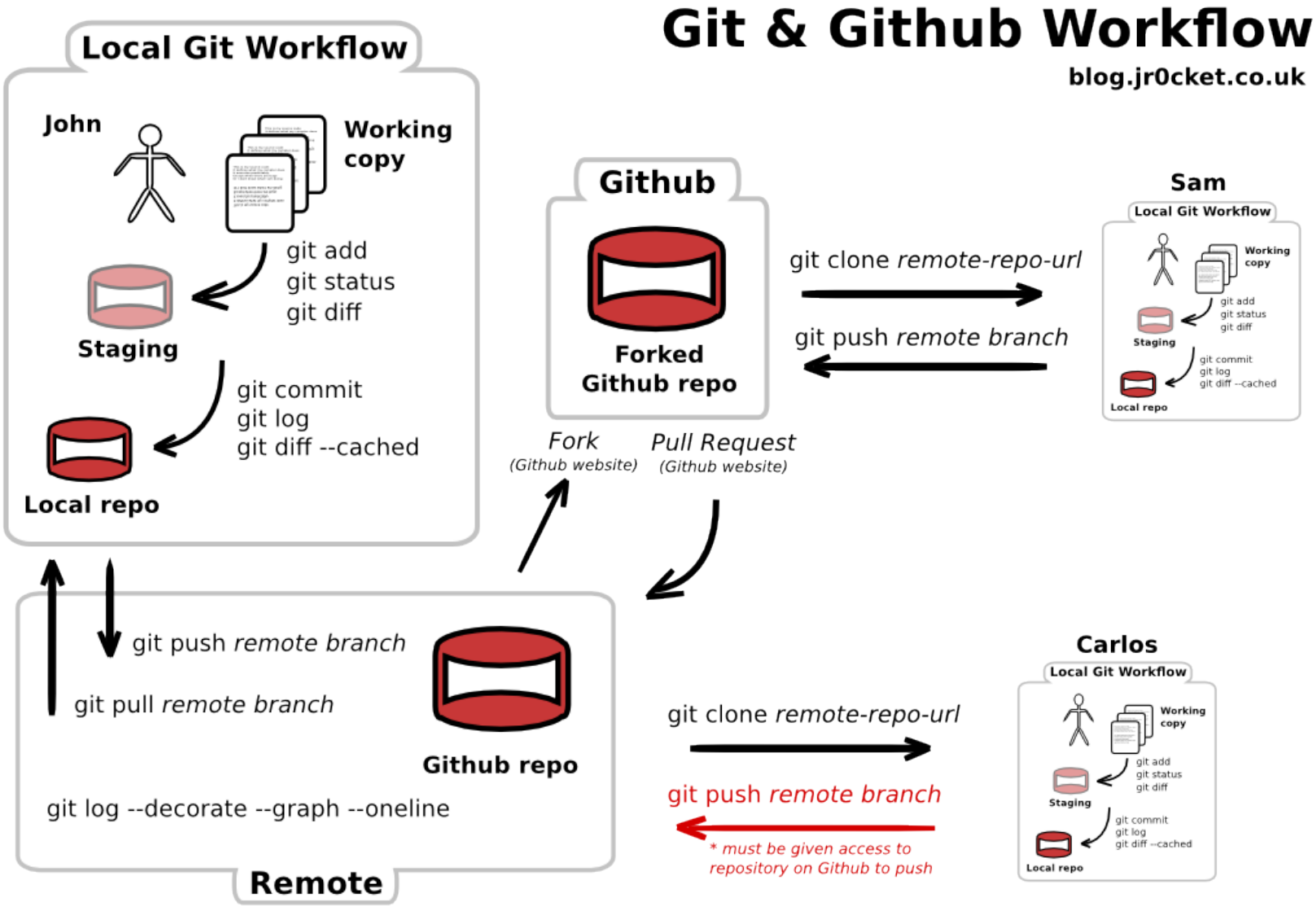
Reading

Chapter in **R Packages** on git and GitHub:
`http://r-pkgs.had.co.nz/git.html`

"Whoah, I've just read this quick tuto about git and oh my god it is cool. I feel now super comfortable using it, and I'm not afraid at all to break something."— said no one ever.

(from blog post by Pierre de Wulf)

GitHub collaboration

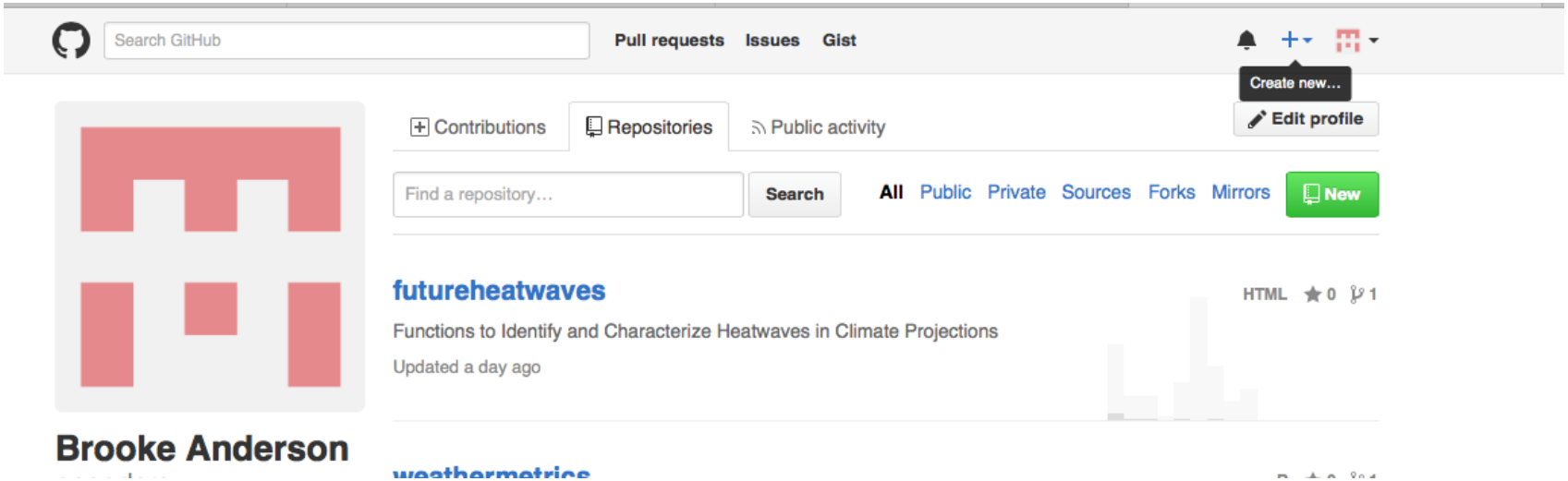


Setting up GitHub collaboration

1. One group member creates a repository.
2. That group member pushes local material to the repository
3. All other group members fork this repository.
4. Everyone creates a local clone of their own GitHub fork of the repository.

1. One group member creates a repository.

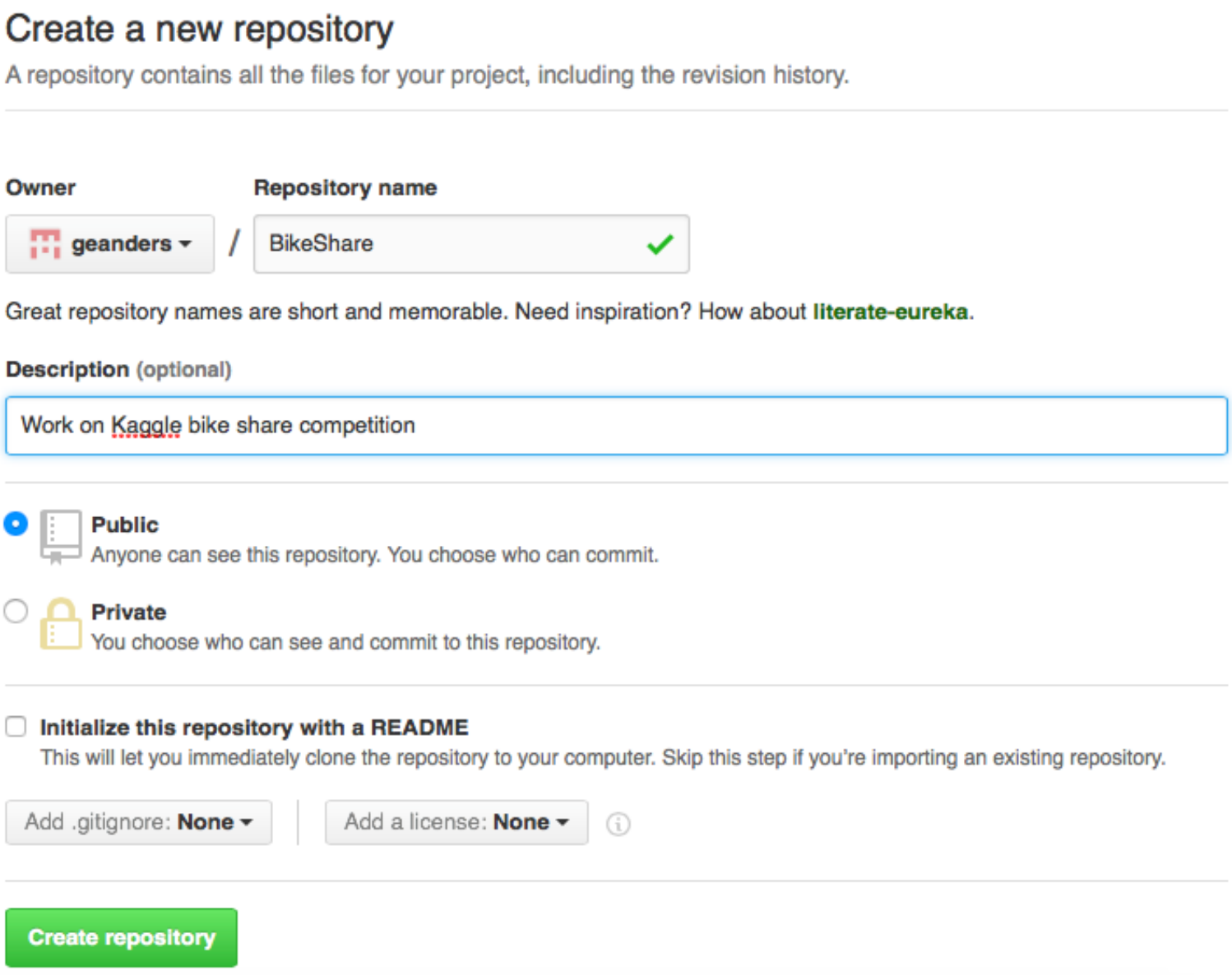
The "+" in the upper right corner of your GitHub account page will let you create a new repository.



1. One group member creates a repository.

I would suggest you create without a README file (you can always add one from RStudio) and use the same name for your repository as the name for your project directory on your own computer.

1. One group member creates a repository.



2. That group member pushes local material to the repository

(Before you do this, make sure that you've made this directory an R project and started git for your local directory using `Tools -> Project Options`.)

First, on your computer, use the command line (**Terminal** with Macs) to move into the directory. You can use the command `cd` to do this.

```
cd Desktop/MachineLearningClass/BikeShare
```

2. That group member pushes local material to the repository

Instruct git that the GitHub repository is a remote version of this directory.

```
git remote add origin git@github.com:geanders/BikeShare.git
```

For RStudio, it's better to use this `git@github.com` syntax rather than what GitHub suggests.

2. That group member pushes local material to the repository

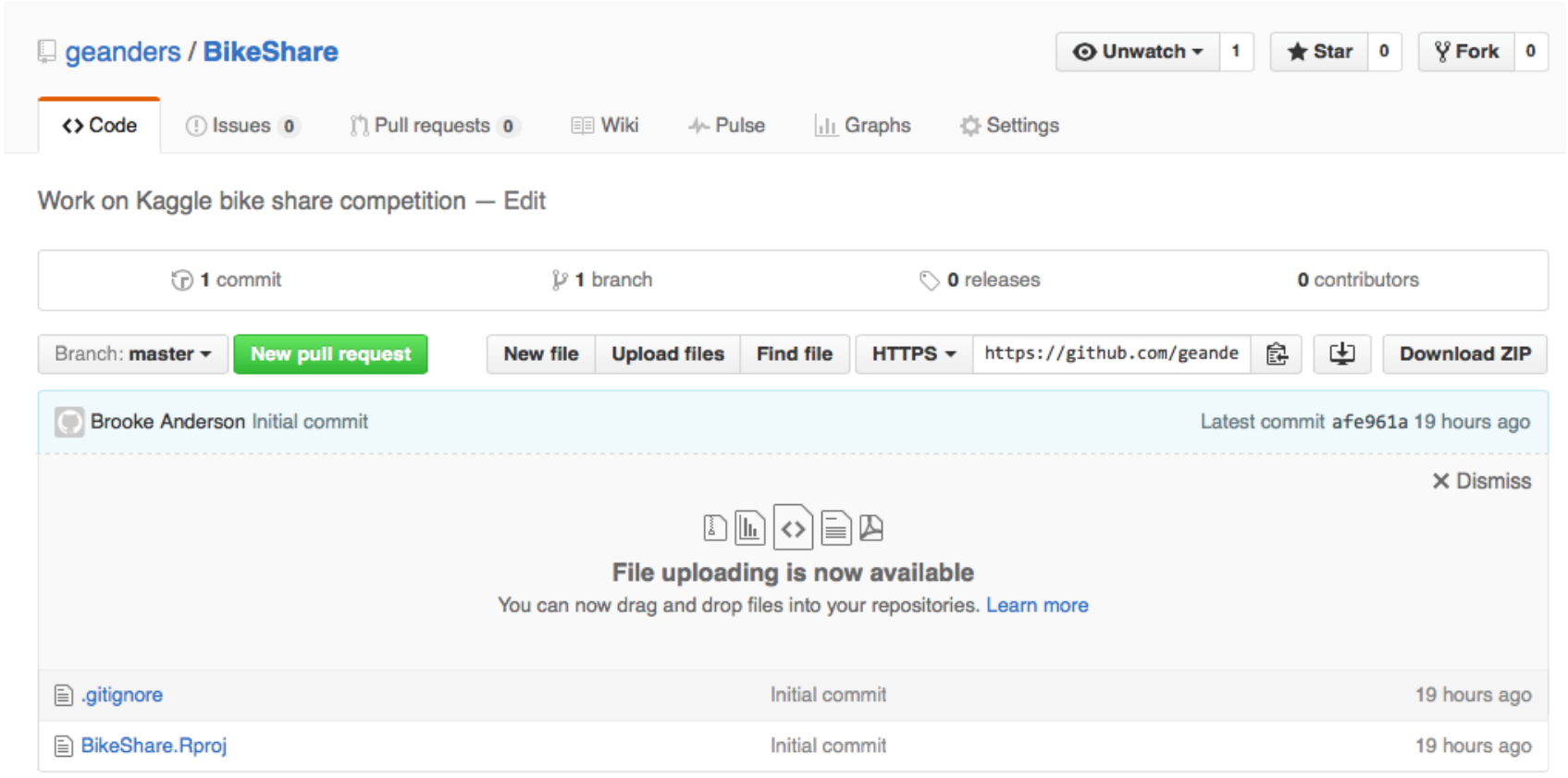
Now push everything from your local directory to the GitHub repository:

```
git push -u origin master
```

Now everything in your directory that wasn't listed in your `.gitignore` file should be on your GitHub repository.

3. All other group members fork this repository.

Next, everyone else in the group needs to fork this repository. They can do this by going to the repository and clicking on the "Fork" button in the top right.



3. All other group members fork this repository.

Once you fork a repository, it will show up as a Repository in your GitHub account.

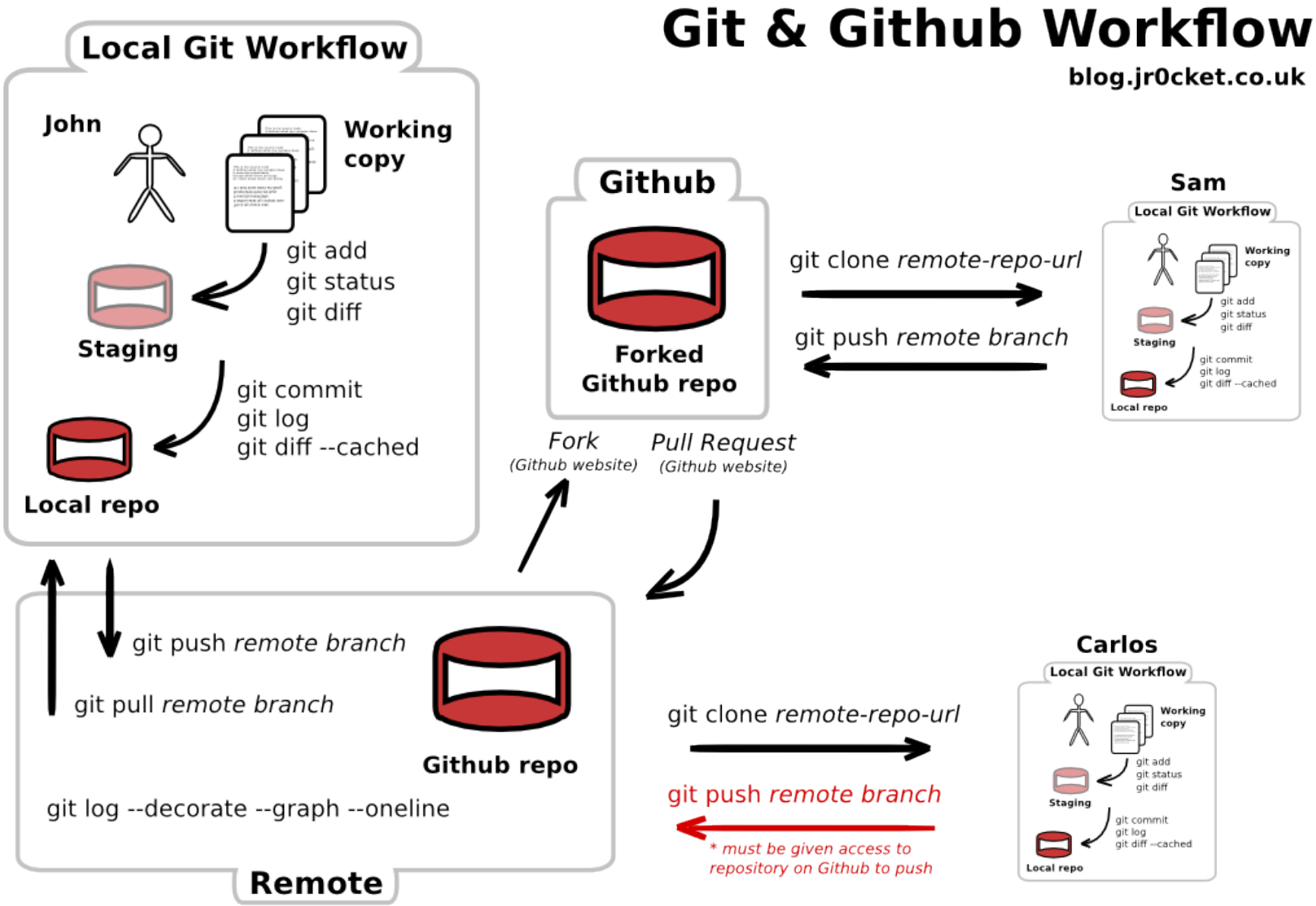
4. Everyone creates a local clone of their fork.

From your command line, change directories into the directory where you want to download the clone of the repository. Then use the following code to clone your fork to a directory in your computer. (Note: This is what you do to pull materials **from** a GitHub directory onto your computer.)

```
git clone git@github.com:group_member/BikeShare.git
```


THIS IS GIT. IT TRACKS COLLABORATIVE WORK ON PROJECTS THROUGH A BEAUTIFUL DISTRIBUTED GRAPH THEORY TREE MODEL.
COOL. HOW DO WE USE IT?
NO IDEA. JUST MEMORIZE THESE SHELL COMMANDS AND TYPE THEM TO SYNC UP. IF YOU GET ERRORS, SAVE YOUR WORK ELSEWHERE, DELETE THE PROJECT, AND DOWNLOAD A FRESH COPY.

GitHub collaboration

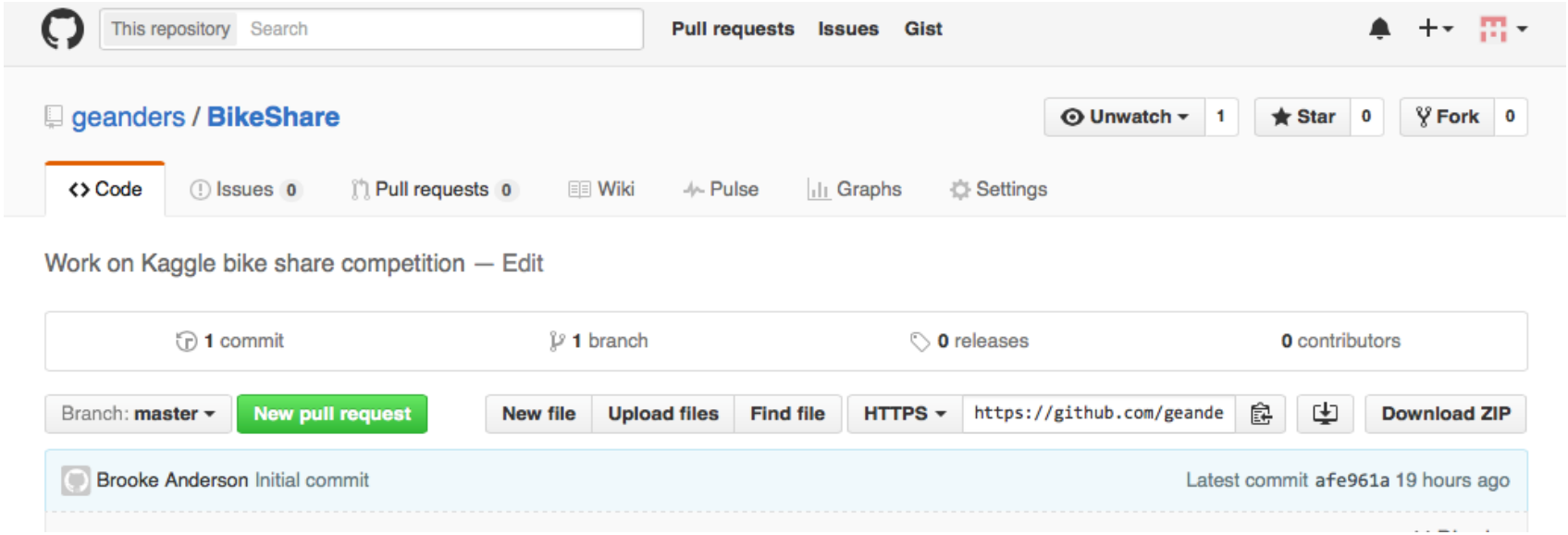


Collaborating

1. Before you start working on your code, make sure you're up-to-date with your group members.
2. Work on your code locally in RStudio.
3. Commit often locally.
4. Push your work to your GitHub fork.
5. When your ready to share with other group members, submit pull requests to them.

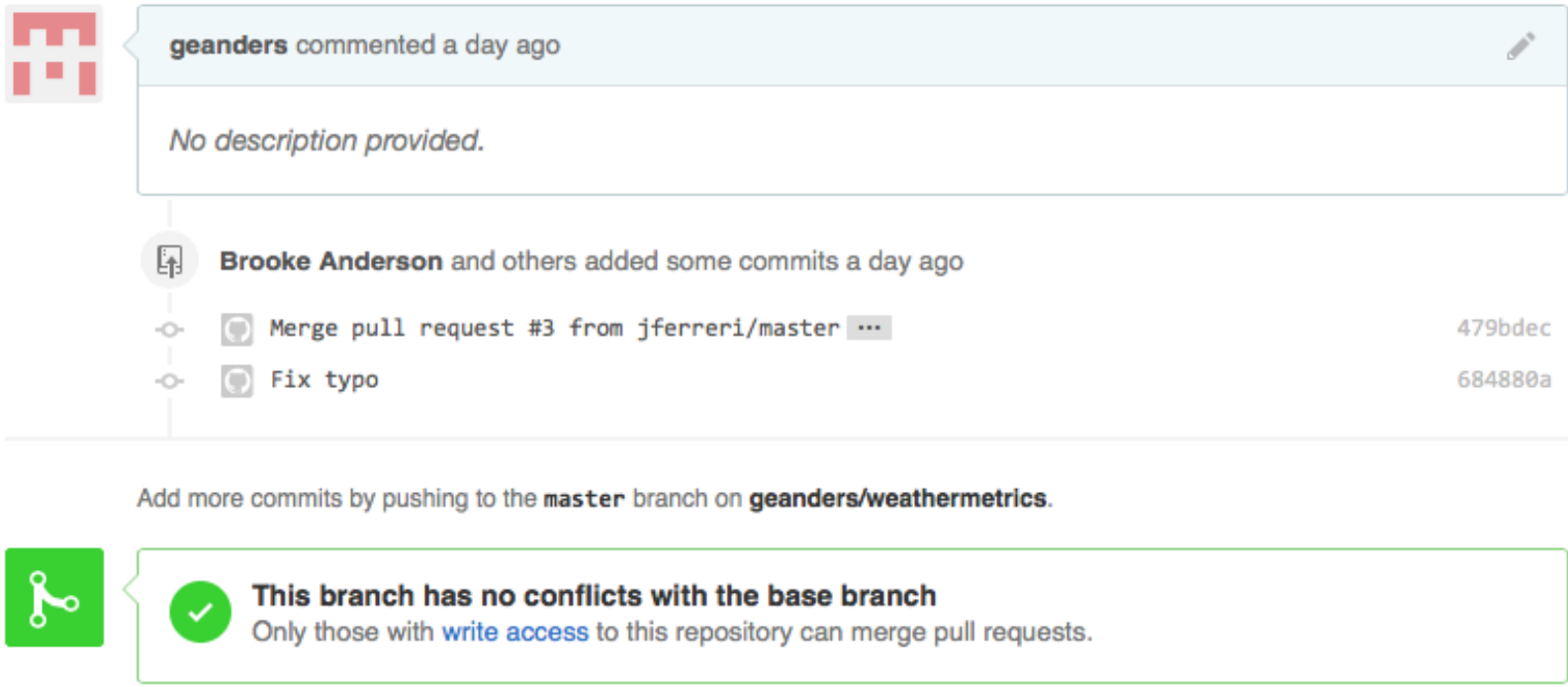
1. Make sure you're up-to-date with your group

I would suggest that you start by trying to work on your repositories at different times, so there won't be conflicts the first few times you try to merge pull requests. If so, just look for any pull requests on your GitHub repository page.



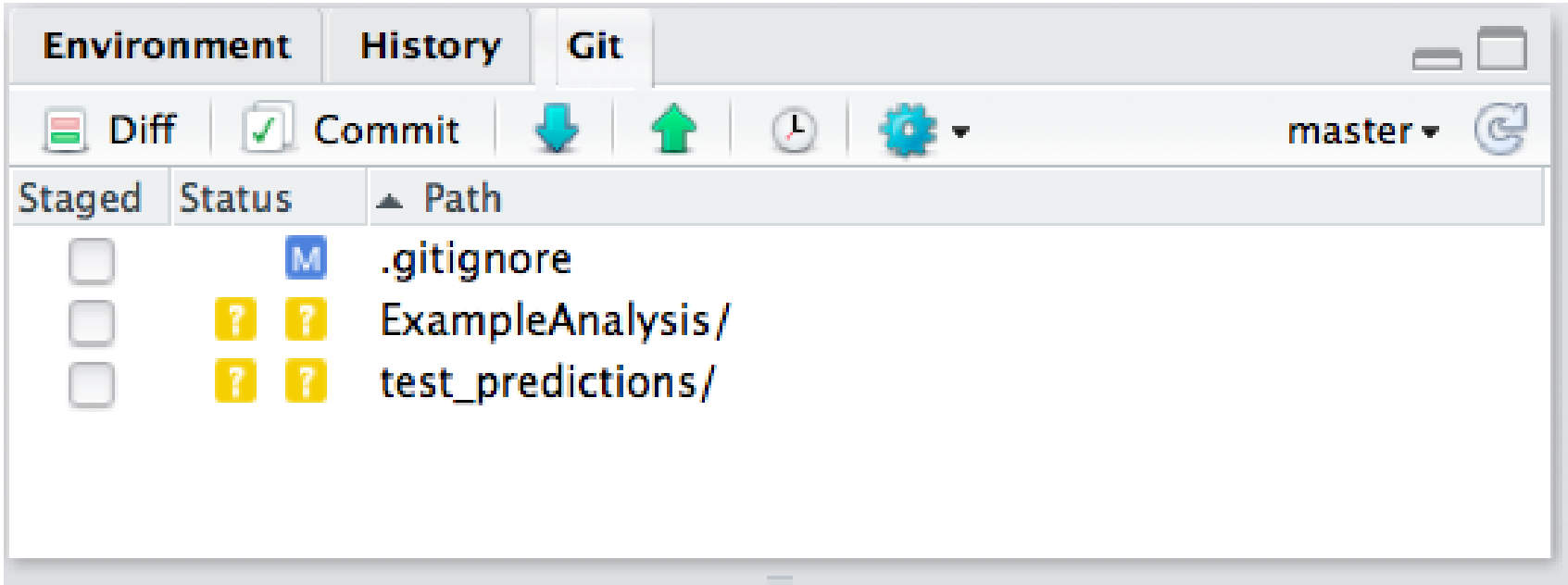
1. Make sure you're up-to-date with your group

If there are no pull requests, you're good to go. If there are, click on the "Pull request" tab. If you're lucky, there will be no merge conflicts. If so, you can merge the pull automatically.



1. Make sure you're up-to-date with your group

To get your local version up-to-date, use the blue down arrow on RStudio's Git pane. This brings in any changes on your GitHub fork to your local directory.

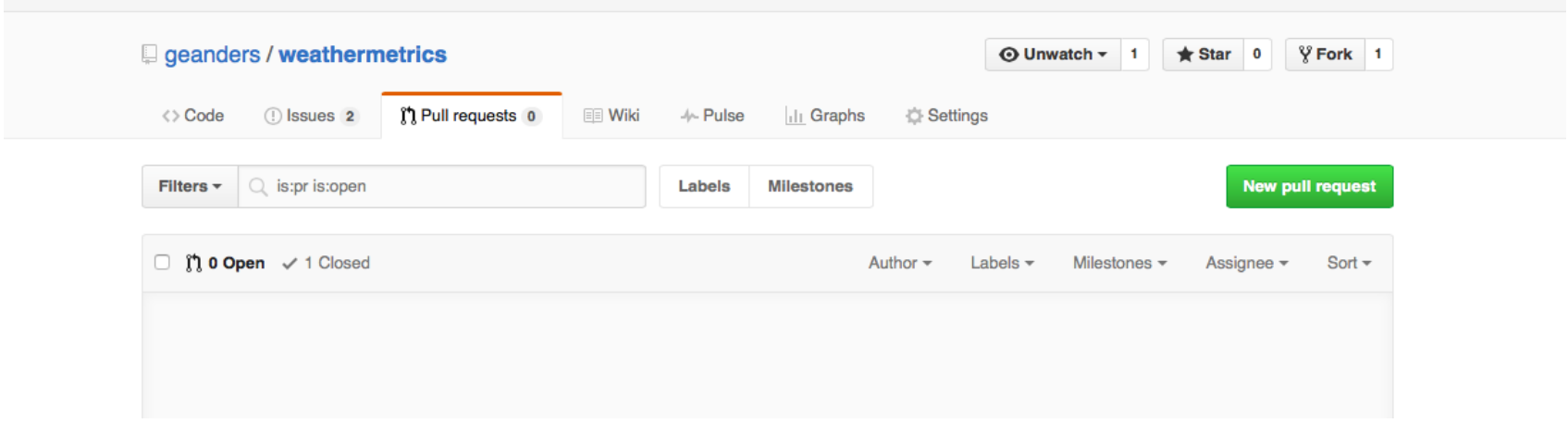


2.-4. Do your own work.

2. Work on your code locally in RStudio.
3. Commit often locally (Use "Commit" window in RStudio.).
4. Push your work to your GitHub fork (Use the green up arrow.).

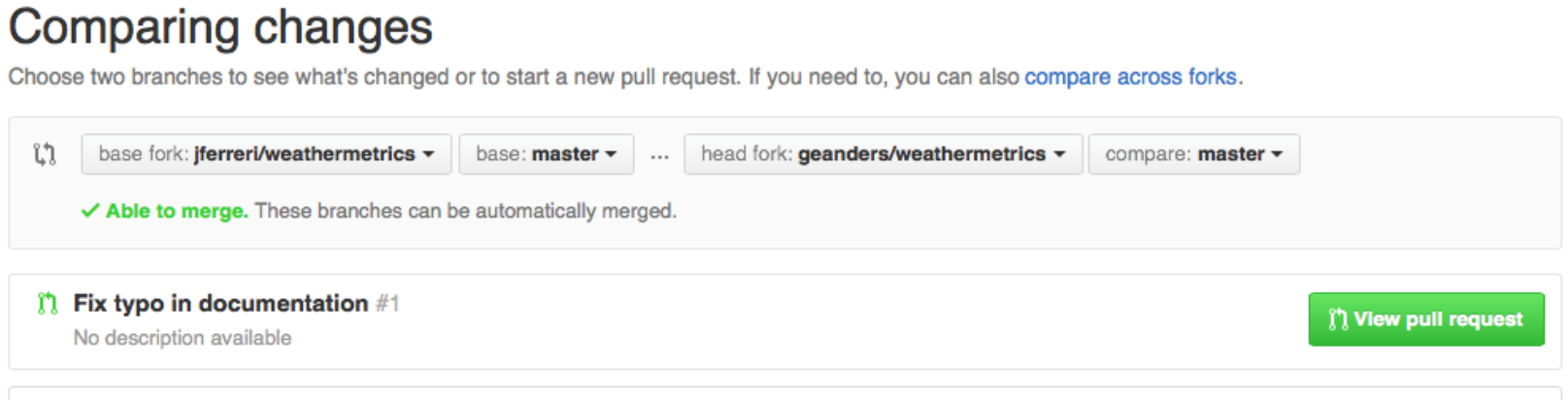
5. Send group members pull requests.

When you're ready for the rest of your group to incorporate your work, send them all pull requests through GitHub. Go to the "Pull requests" tab of your fork of the repository and look for the green "New pull request" button.



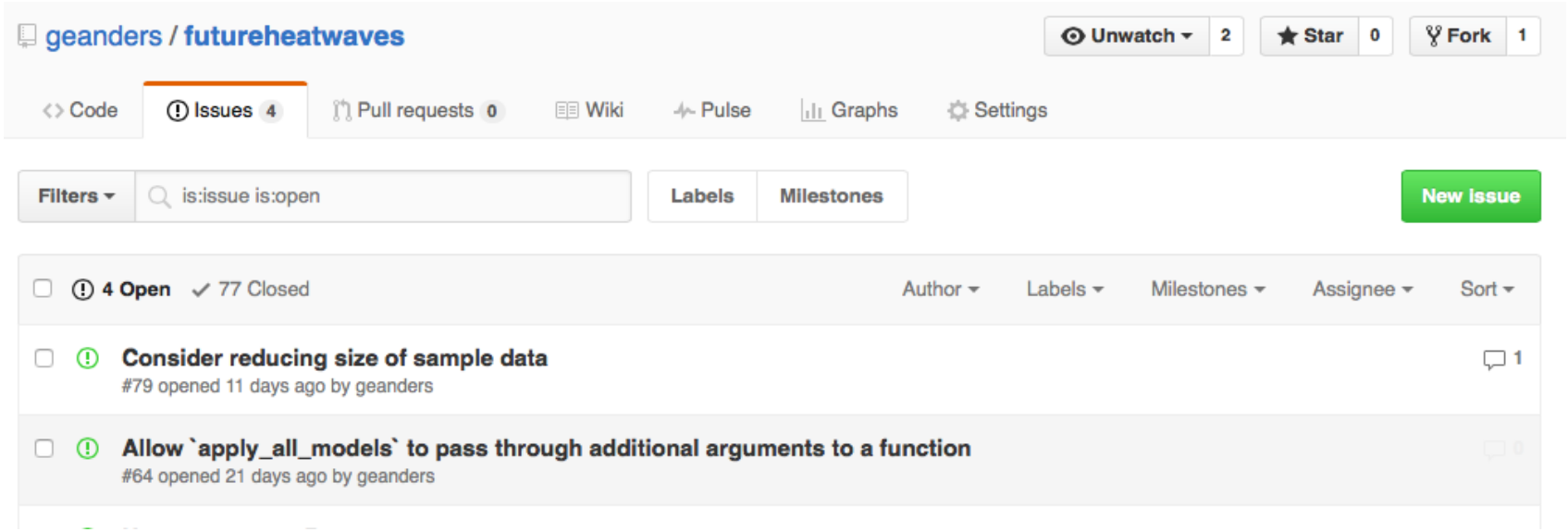
5. Send group members pull requests.

Submit your pull request. You may need to use the "compare across forks" link to set up the right pull direction. Make sure that the base fork is the one you want to send your updates to and the head fork is yours.



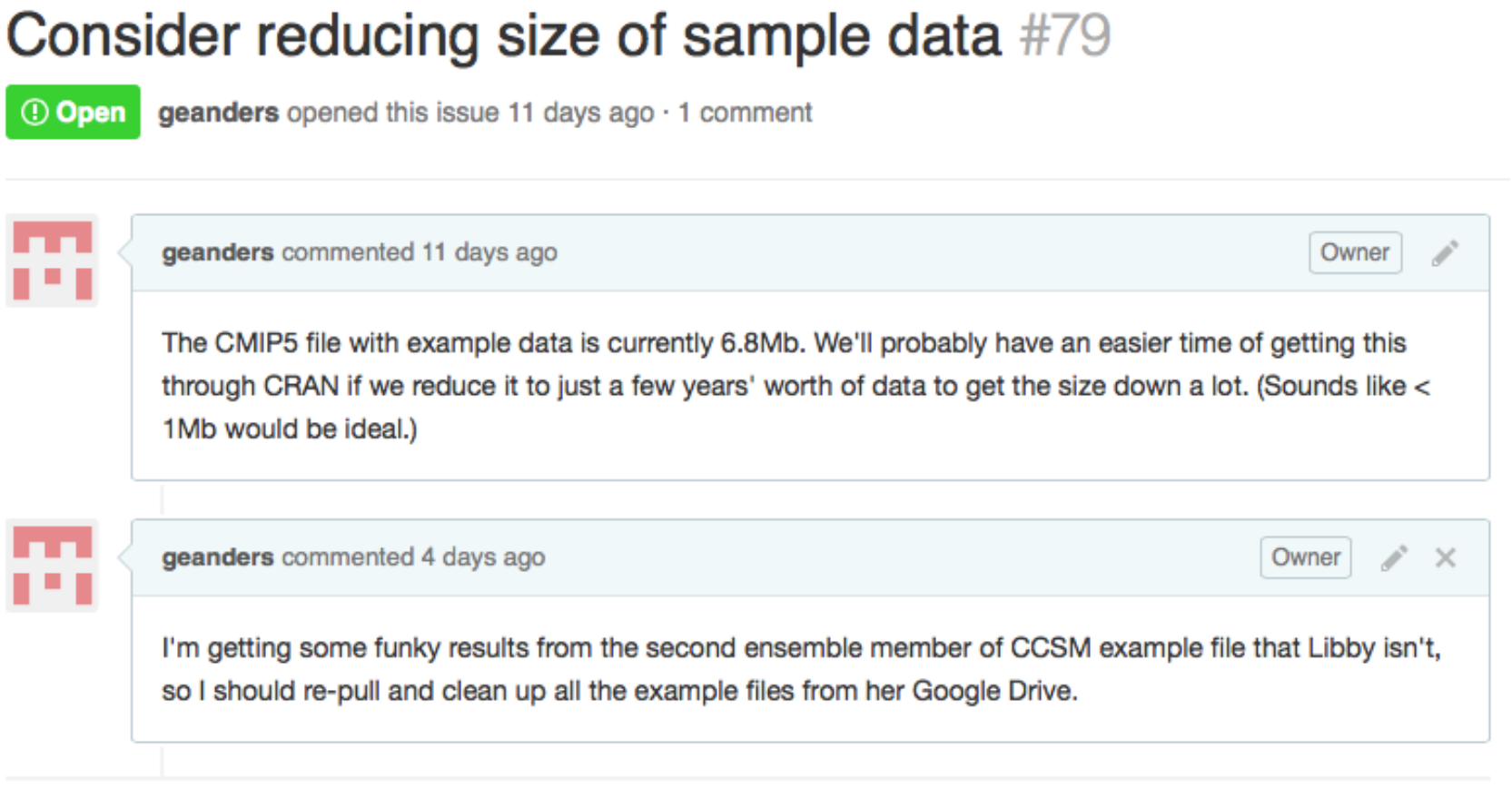
Issues page

The Issues page is also great to use to work together. You can create new issues using the green "New Issue" button.



Issues page

Each issue has a page where you can add notes about the issue.



Issues page

When you've fixed an issue, you can close it. You can do that either from the GitHub page for the issue, or do it with a commit message. For example, if a commit fixes issue #5, you could close the issue by using the commit message:

```
Close #5.
```