GitHub

MLH Localhost

Facilitator's Guide

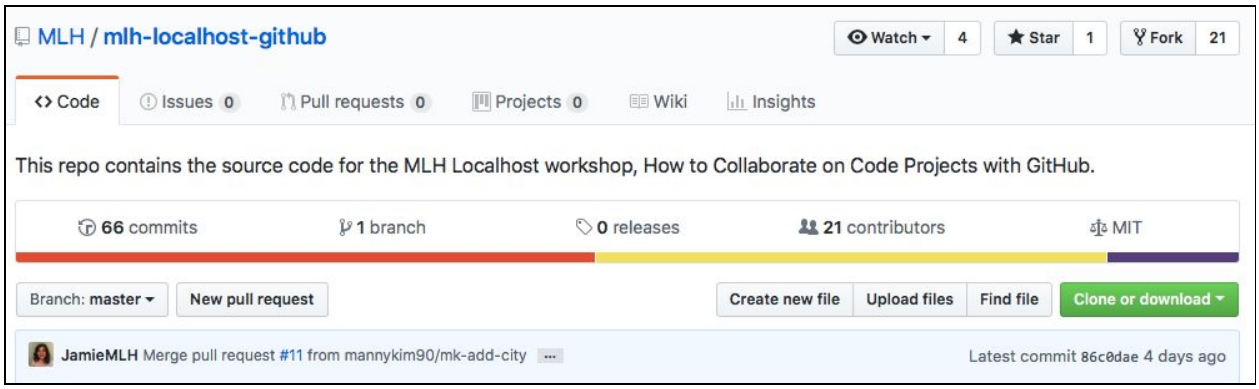
This is the facilitator's guide for the GitHub workshop. In this workshop we will build a map of hometowns that participants from Localhost events come from. While doing so, we will learn the basics of Git and GitHub, the GitHub workflow, and how to work with GitHub through the command line.

STEP 1: Get Ready!

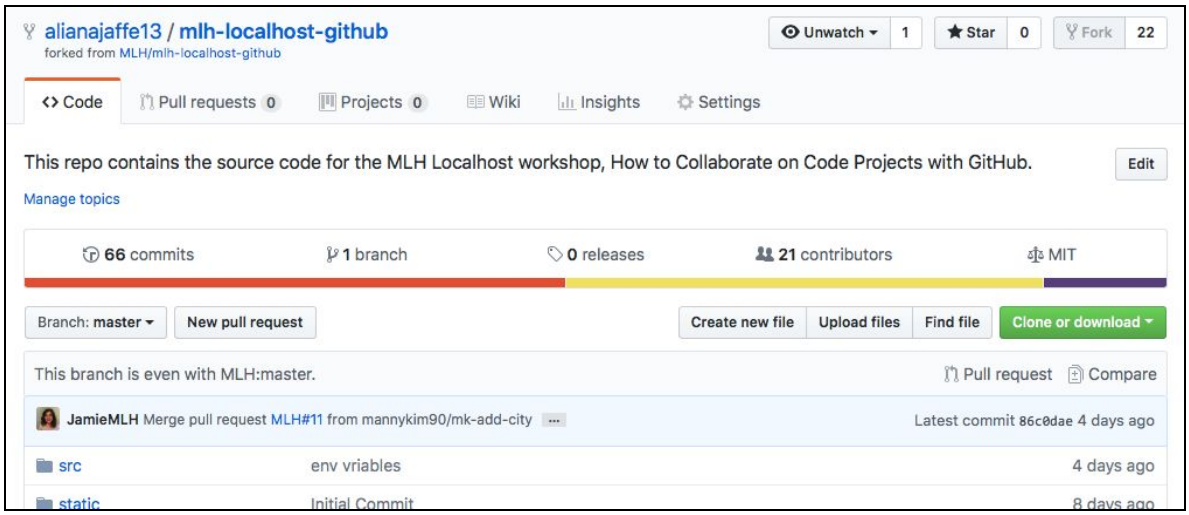
- Navigate to GitHub and make an account: https://github.com
- Navigate to the repository that we'll be working on: http://mlhlocal.host/github-code

STEP 2: Fork Code

- Click on "Fork" in the Top Right corner.

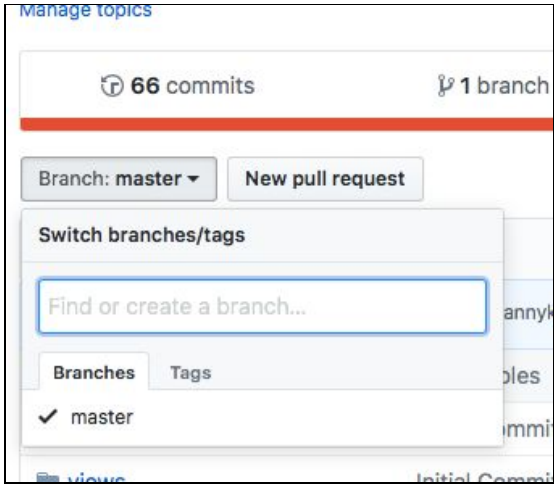


- A new page will load with the forked code. You will know if the fork was successful if you now see your user name at the top of the Repo page.

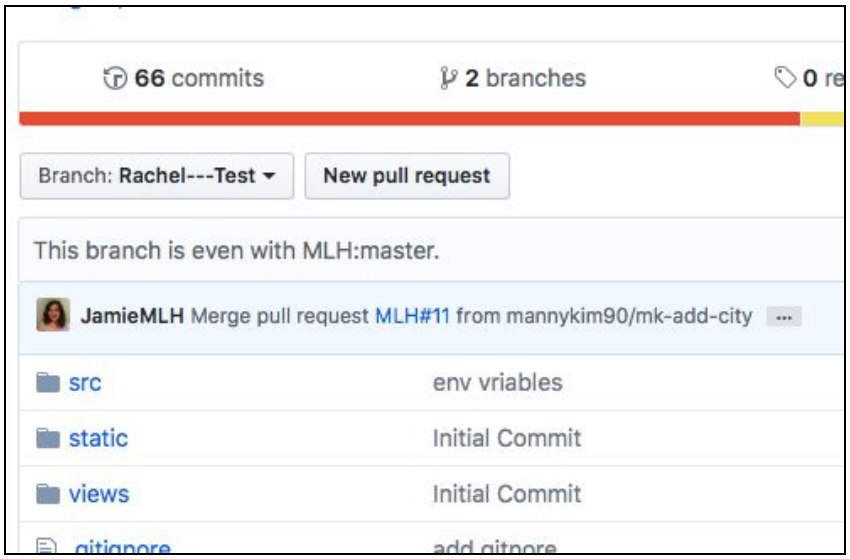


STEP 3: Create a Branch

- Navigate to the button that says **Master.** To create a new branch, put in your name, and what you plan to do in this branch, and then **Select** the new branch to confirm.

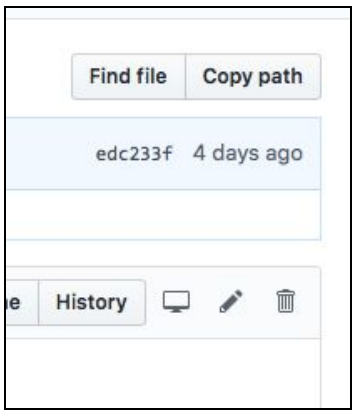


- You should now see a new page with your name and descriptor above.

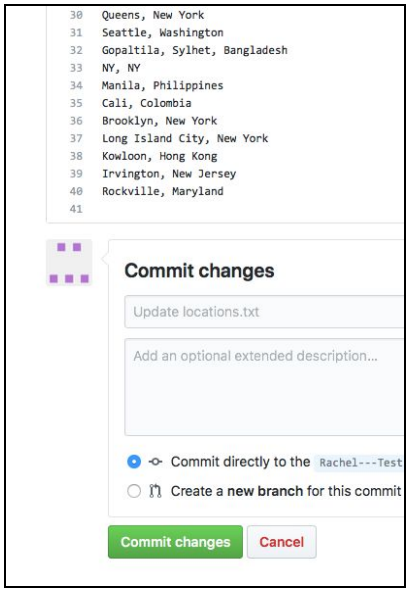


STEP 4: Developer Workflow

- Open `locations.txt` and then Select **Edit** (it's an icon of a pencil).

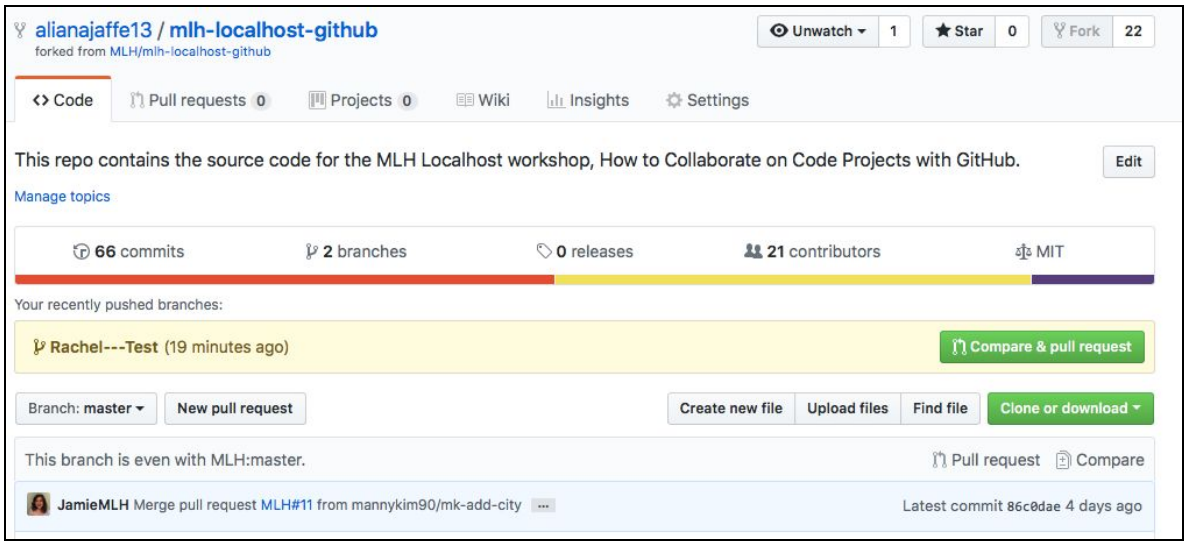


- There should be a list of cities and countries. This list has been constructed from other people that have done this workshop and have had their edits committed to the main branch. Scroll down the list and add your hometown and country to the list! Make sure to add your hometown to the bottom of the list. This will help your workshop leader easily see and approve of changes to the code base.
- Then press the **Commit** button. Make sure that you are committing directly to your main branch instead of making a new branch for this commit.

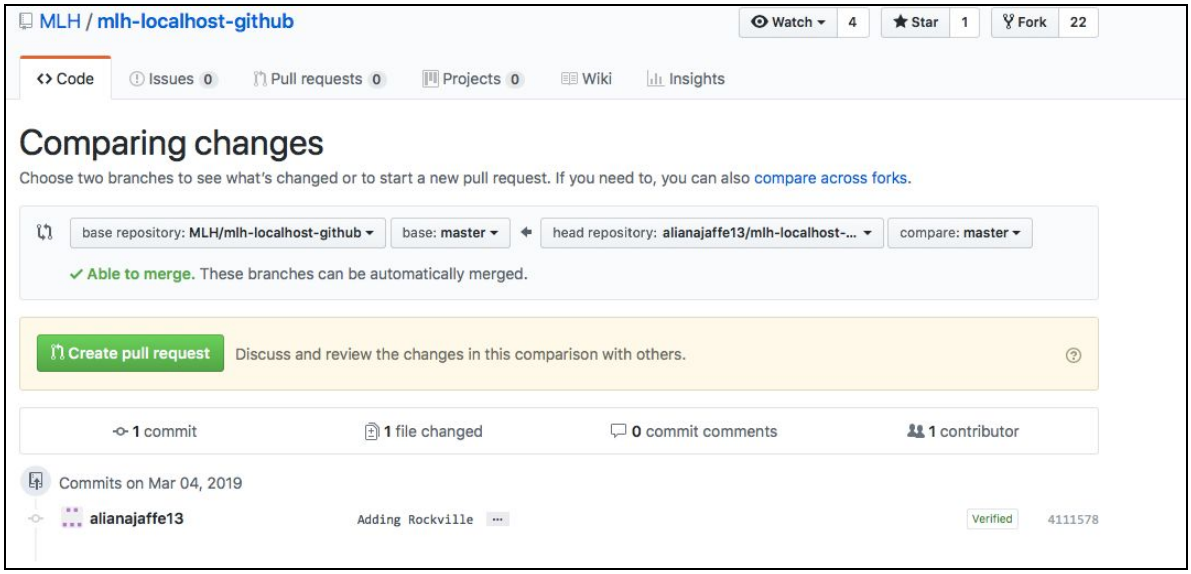


STEP 5: Pull Requests

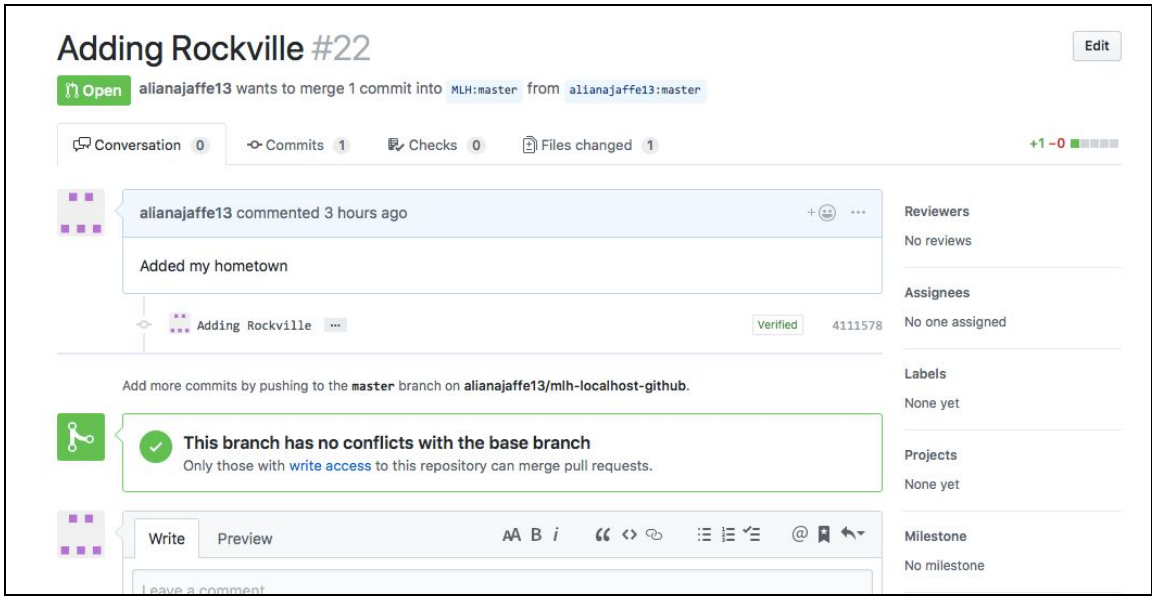
- Navigate back to your home page (**Not** the Pull Requests button on the Header) then Select **Pull Request** in grey. *Don't* Select **Compare & Pull Request.** This will bring you to the wrong page.



- You should get a screen that says "Welcome to Pull Requests!". Select **New Pull Request.** You will be brought to a page for "Comparing Changes". If you have done the previous steps correctly you will see "Able to Merge" in Green.



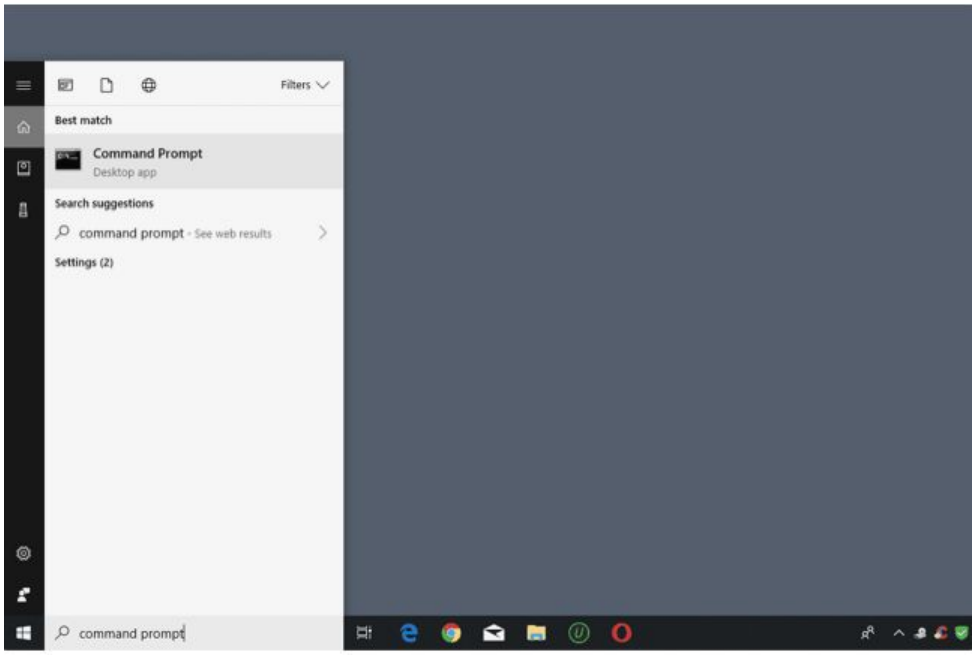
- After you Select **Create Pull Request** you should get a screen that says there are no conflicts with the base branch.



- If you do have a conflict, navigate to your **Pull Requests** page. Within this page, Select the pull request with the merge conflict.
- At the bottom of the page, Select **Resolve Conflicts**.

Step 5: GitHub from the Command Line

- First you will need to navigate to your command line. This is done on Windows by going to the Command Prompt shortcut in the Start window or the Apps screen.



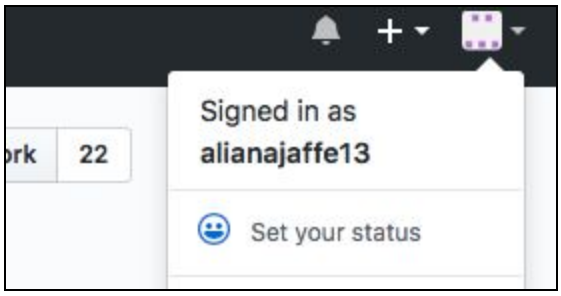
- On Mac, the command line is accessed via **Applications > Utilities**. Within **Utilities**, select **Terminal**.

- Once in your Terminal, type `--git version`.

- If your Terminal does not display a current git version, you will need to install it: **http://mlhlocal.host/github-code**
- Make sure to Select your correct operating system, or the downloaded git version will not run.
- For both Windows and Mac, open the installer and keep the default options selected. Then restart your Terminal, and run the `--git version` command again.

STEP 6: Configure Git

- To configure Git, go to your Top right corner on GitHub and copy your username.



- Then in your Terminal, type:

```
git config --global user.name “your user name”
git config --global user.email “your email address”
```

- It is not immediately apparent that when you configured your Git username and email whether it worked or not. To make sure that you have correctly configured your username and email, type in your Terminal:

```
git config user.name
git config user.email
```

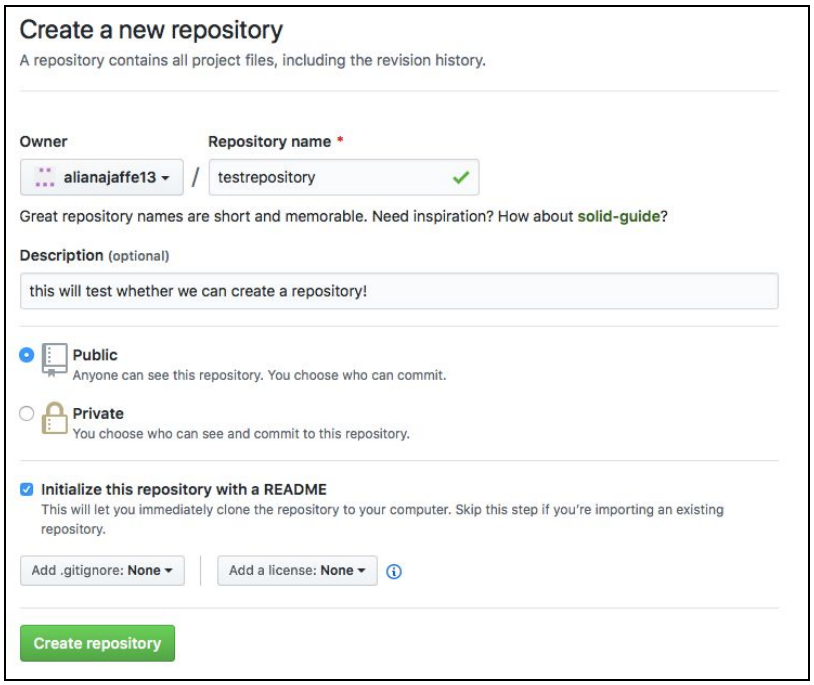
- The final Terminal window should appear as follows:

```
Rachels-MacBook-Air-2:~ racheljaffe$ git config --global user.name "alianajaffe1
3"
Rachels-MacBook-Air-2:~ racheljaffe$ git config --global user.email "aliana@umic
h.edu"
Rachels-MacBook-Air-2:~ racheljaffe$ git config user.name
alianajaffe13
Rachels-MacBook-Air-2:~ racheljaffe$ git config user.email
aliana@umich.edu
Rachels-MacBook-Air-2:~ racheljaffe$
```

- If your Terminal looks different, double check your spelling of the commands and your password.

STEP 7: Create a Repo

- Navigate to Github and Select **New** on your profile or homepage. On this page, add a name and description. Make sure **Initialize this repository with a README** and **Public** is Selected. You will also want to make sure that whatever you name your repository, it is either one word, or connected via dashes.



- Make sure to clone your directory and copy the address to your repo.
- Type git clone `https://github.com/yourgithubname/yourrepositoryname.git` into your Terminal.

```
Rachels-MacBook-Air-2:~ racheljaffe$ git clone https://github.com/alianajaffe13/
testrepository.git
Cloning into 'testrepository'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), done.
Rachels-MacBook-Air-2:~ racheljaffe$
```

STEP 8: Change Directory

- To make a change to the directory, the first thing needed is to change into the directory that we need. This is done by typing the command `cd` and then the name of the directory that you want to navigate into.
- Then we need to create a new branch. This is done by the command `git checkout -b "nameofyourbranch"`. This is where your computer might have trouble finding the repo if you used two words to name it.
- Open Atom, or your text editor of choice, and make some alteration to the `README.md` document that you generated when you created your repository. Make sure to **Save** these changes.
- Navigate back to git and enter the command `git status`. It might look like there is an error, but if you see a red error message that means that your changes to your README have been changed and your repository is communicating correctly with GitHub.

```
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:   README.md

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

        atom.testrepository
```

- Type `git add REAME.md`
- Then, when you check your git status again, there will now be a green message that says modified: README.md.

```
Rachels-MacBook-Air-2:testrepository racheljaffe$ git add README.md
Rachels-MacBook-Air-2:testrepository racheljaffe$ git status
On branch testrepository
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

        modified:   README.md
```

- To commit these changes, enter the command `git commit -m “first commit.”` The message in quotation marks can be anything, but it is standard for the first commit to a repository to be labelled “first commit.”
- Finally, type `git push`.

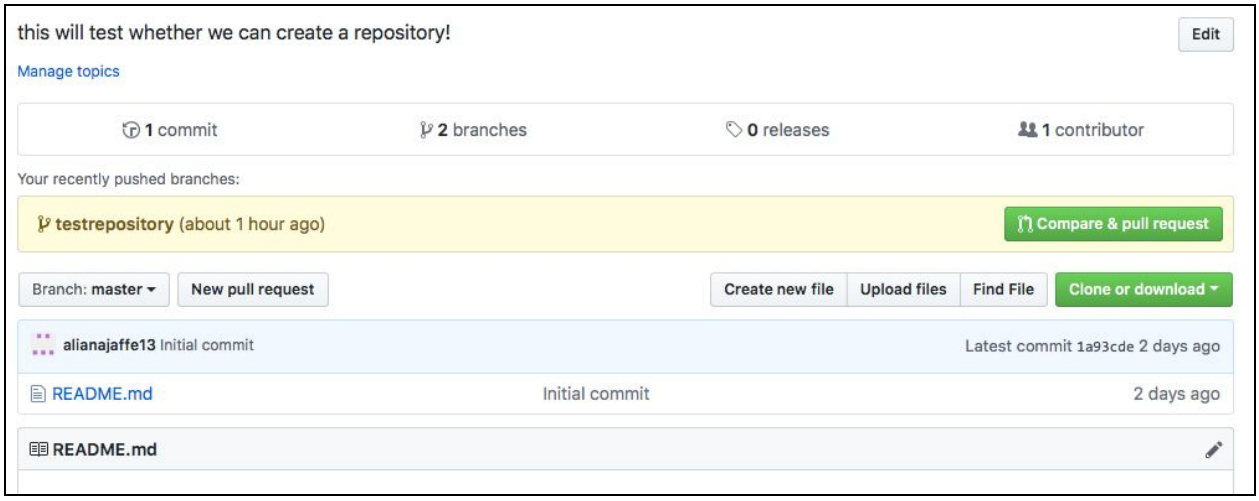
```
Rachels-MacBook-Air-2:testrepository racheljaffe$ git commit -m "first commit"
[testrepository c11f418] first commit
 1 file changed, 2 insertions(+)
Rachels-MacBook-Air-2:testrepository racheljaffe$ git push
fatal: The current branch testrepository has no upstream branch.
To push the current branch and set the remote as upstream, use

    git push --set-upstream origin testrepository
```

- Git will tell you the next command you need to insert. It should look close to this though:

  `Git push --set-upstream origin testrepository`

- The code above is an example. Your terminal will use the name of whatever you named your repository.
- You will be prompted to enter your username and password for GitHub. Do not be alarmed when your password is not visible, it is a security measure.
- You will get a print out in your terminal of a URL address for “Create a pull request.” Copy this URL and put it in your browser. Make sure you copy the first web address that is output to your Terminal. If you copy the second web address, it will take you to the wrong page. If your web page does not have a **Compare & pull request button,** go back to your Terminal and Select the other hyperlink.



- Select **Compare & pull request**, name your pull, and create it.
- There should be no conflicts with the base branch. If you do have a conflict, you will see a list of files with conflicting changes above the Merge pull request button. This button will be deactivated until you have resolved all conflicts. You should not need to deal with this for the workshop, but it is useful to know when working with teams.
- Enter `git pull` into your Terminal. This pulls the changes that you made on the master branch on GitHub onto your computer.
- These changes should be reflected in your browser. If not, check your network connection and refresh the page.