Get Crackin' with Git Using GitKraken

MLH localhost

Welcome to:
Get Crackin' with Git using GitKraken

Wifi Network:
[Library WiFi]

Wifi Password:
[11111111]

Facebook Handle:
@ieeeduth
Twitter Handle:
@GitKraken

1

Using your web browser, open this URL & fill out the form:

http://mlhlocal.host/checkin

2

Afterwards, check your email to find:

- Setup Instructions
- An Invite to the MLH Slack
- The Code Samples
- A Workshop FAQ
- These Workshop Slides
- More Learning Resources

This presentation was created in partnership with MLH

MLH
MAJOR LEAGUE HACKING

Whose mission is to empower hackers!

65,000+
HACKERS

12,000+
PROJECTS CREATED

3,000+
SCHOOLS

Why are we here?

- How many people here have had to collaborate on a code project before and weren't sure how to go about it?
- How many of us have used Git/GitHub or other tools to collaborate and felt like it was a really tricky process?
- Maybe you've worked on a project with others using Git and had a hard time figuring out who had done what.
- Or maybe you are comfortable with Git, but you're not sure how to track who's working on what?

Luckily, there are tools to make this process much simpler!

What You Are Going to Learn Today

1. A best practice workflow for collaborating using Git
2. How GitKraken simplifies collaboration
3. How to use Glo Boards to manage tasks

What You Are Going To Build

Together with your teammates, you'll use a simple HTML/CSS template to build a team webpage like the one below. You don't need to know any HTML/CSS for this to work! We'll be using this project to help us learn GitKraken and Glo Boards.

Who We Are

Rosie the Kraken

I'm Rosie the Kraken! In the 1940s, I encouraged women to join the World War II effort in the United States. Since then, I have become a symbol of women participating in all sorts of industries where they are underrepresented. We can do it!

How You Are Going To Do It

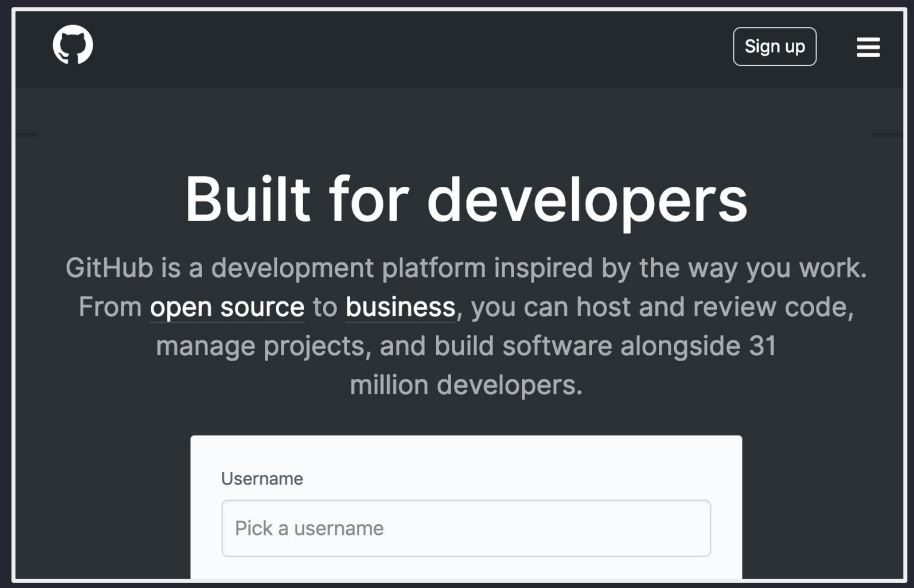
1. We'll discuss the traditional Git workflow and working with Git from the command line.
2. Then, you'll make GitHub and GitKraken accounts.
3. You'll download and install GitKraken.
4. We'll explore GitKraken and Glo Boards.
5. You'll form a team to work with.
6. Your team will create a shared Glo Board to track tasks.
7. You'll fork and clone the project.
8. Your team will collaborate on the project using GitKraken!

Make a GitHub Account

Visit the URL on the left if you're a student or the right if you are not.
Follow the instructions on GitHub.com to make your account!

http://mlhlocal.host/github-edu

http://mlhlocal.host/github-signup



Great! Now that you know what's coming, let's understand Git.

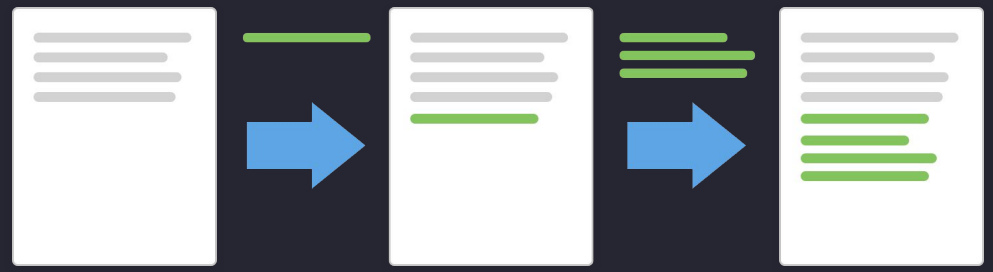
Table of Contents

1. Intro to Git and GitHub
2. Set up GitKraken
3. Collaborate!
4. Review and Quiz
5. Wrap Up and Next Steps

What's Git?

Git is a version control system that helps you collaborate on projects with others. There are two main features that help you achieve this.

1. Log changes in a searchable way, instead of renaming a file for each version.



2. Collaborators can work in parallel and merge their changes automatically, instead of manually comparing the differences between a file

Old-School Git

Some developers use the command line to interact with Git and GitHub by following the best-practice workflow described in the next slides.

1. First, they might **fork** someone else's code. This creates their own copy of someone else's project.

2. Then they would create a **branch**. The branch is a parallel version of their fork, where they can test changes without editing the master copy.

Key Terms

fork: your own copy of someone else's repository.

branch: a parallel version of the master copy of a repo. Making a branch allows you to edit code without accidentally breaking a working version.

Old-School Git

3. They **stage** their changes as they go. When they're happy with them, they **commit** them.
4. To add their working version (branch) to the master branch, they open a **pull request**.
5. Then, they (or the project owner) will **merge** the branch into the master branch.

Key Terms

stage: add to a cohesive group/bundle of revisions

commit: a group of revisions you want to officially add to your branch

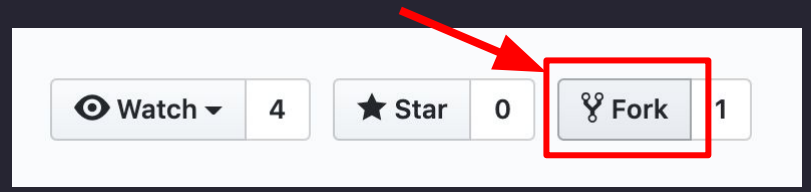
merge: to officially add the changes from your branch into the master branch (or another branch)

The next few slides will explain the process by which developers might interact with Git using their command line. This is just an explanation – these are not steps to follow!

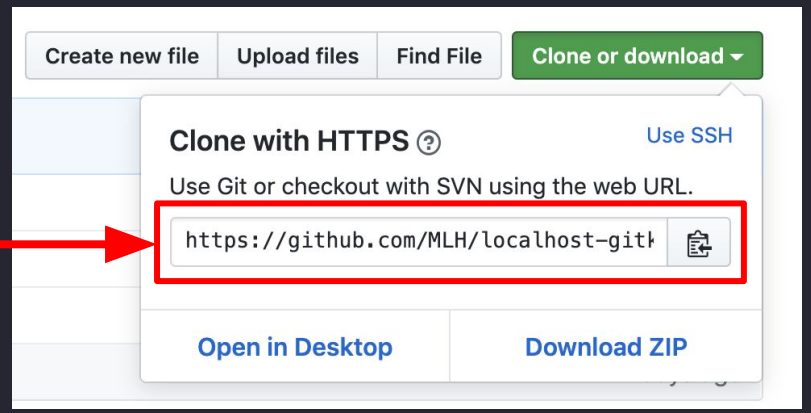
How do they do this?

If a developer is using the command line to complete these steps, their old-school workflow might look something like this:

1. Go to GitHub to find the project they want to work with and click fork.



2. Go to their own GitHub profile to see their fork of the repo, and copy the clone URL.



How do they do this?

3. Open their terminal and enter these commands to clone the repo and create a branch.

```
@localhost > git clone https://github.com/MLH/localhost-gitkraken.git
Cloning into 'localhost-gitkraken'...
remote: Enumerating objects: 7, done.
remote: Counting objects: 100% (7/7), done.
remote: Compressing objects: 100% (6/6), done.
remote: Total 7 (delta 1), reused 3 (delta 0), pack-reused 0
Unpacking objects: 100% (7/7), done.
@localhost > cd localhost-gitkraken/
@localhost > git checkout -b "mlh-add-bio"
Switched to a new branch 'mlh-add-bio'
@localhost >
```

How do they do this?

4. After making changes using their text editor, they would enter these commands to stage their changes and commit them.

```
@localhost > git status
On branch mlh-add-bio
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

no changes added to commit (use "git add" and/or "git commit -a")
@localhost > git add README.md
@localhost > git status
On branch mlh-add-bio
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

        modified:   README.md

@localhost > git commit -m "modify readme"
[mlh-add-bio a60f4e6] modify readme
 1 file changed, 1 insertion(+), 1 deletion(-)
@localhost >
```

How do they do this?

5. Then they would push the changes up to the repository on GitHub.

```
@localhost > git push --set-upstream origin mlh-add-bio
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 12 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 312 bytes | 312.00 KiB/s, done.
Total 3 (delta 2), reused 0 (delta 0)
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
remote:
remote: Create a pull request for 'mlh-add-bio' on GitHub by visiting:
remote:      https://github.com/MLH/localhost-gitkraken/pull/new/mlh-add-bio
remote:
To https://github.com/MLH/localhost-gitkraken.git
 * [new branch]      mlh-add-bio -> mlh-add-bio
Branch 'mlh-add-bio' set up to track remote branch 'mlh-add-bio' from 'origin'.
@localhost >
```

6. Then they might go to GitHub to review and merge the pull request.

Why GitKraken?

Git is powerful, but memorizing commands is time-consuming and how Git works can be tricky to visualize.

GitKraken provides a visual interface enabling the same workflow with only a few clicks.

Now that you have a basic introduction to Git, let's have GitHub and GitKraken do the hard work for us!

Table of Contents

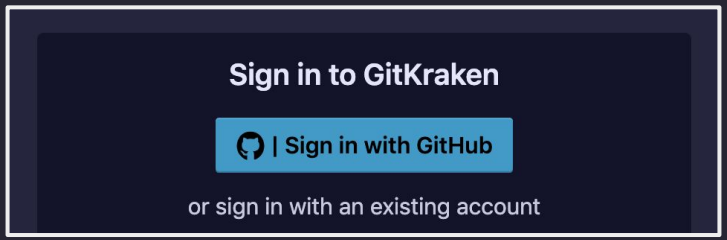
1. Intro to Git and GitHub
2. Set up GitKraken
3. Collaborate!
4. Review and Quiz
5. Wrap Up and Next Steps

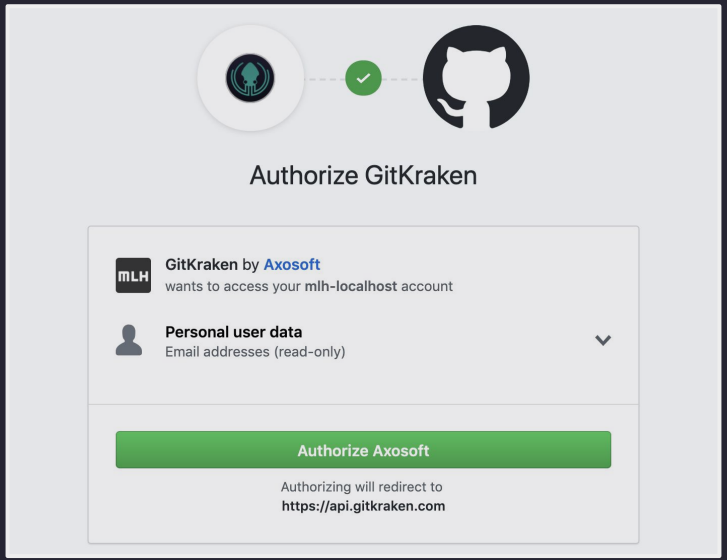
Sign up for GitKraken

Follow the instructions below to create a GitKraken account.

https://www.gitkraken.com/invite/wXaTLK8r

1. Once you have created a GitHub account, visit the URL above.
2. Select **Sign in with GitHub**.
3. Then select **Authorize Axosoft** (the company that makes GitKraken!)





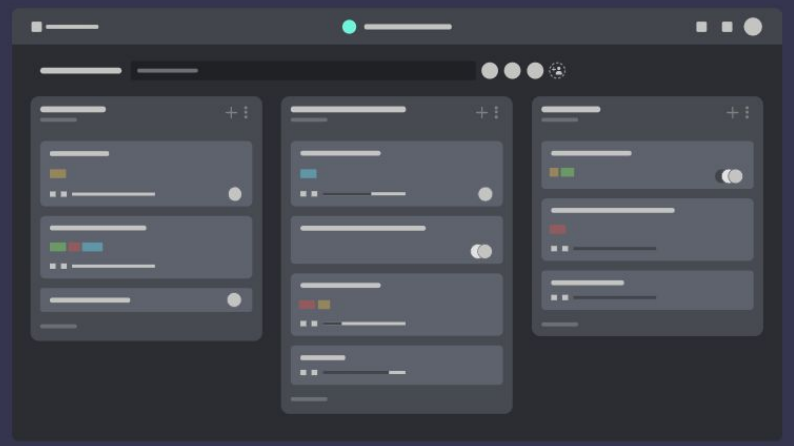
Sign up for GitKraken

Products

Get started with your account by downloading the GitKraken Git Client or visiting Glo Boards.

Download the Git Client

Start Using Glo Boards

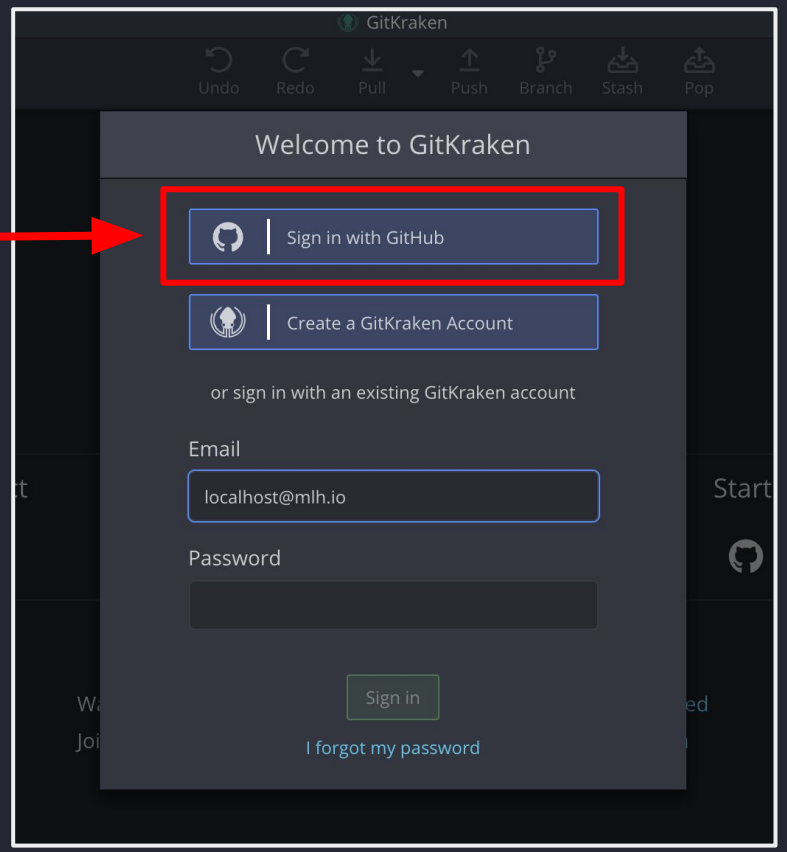


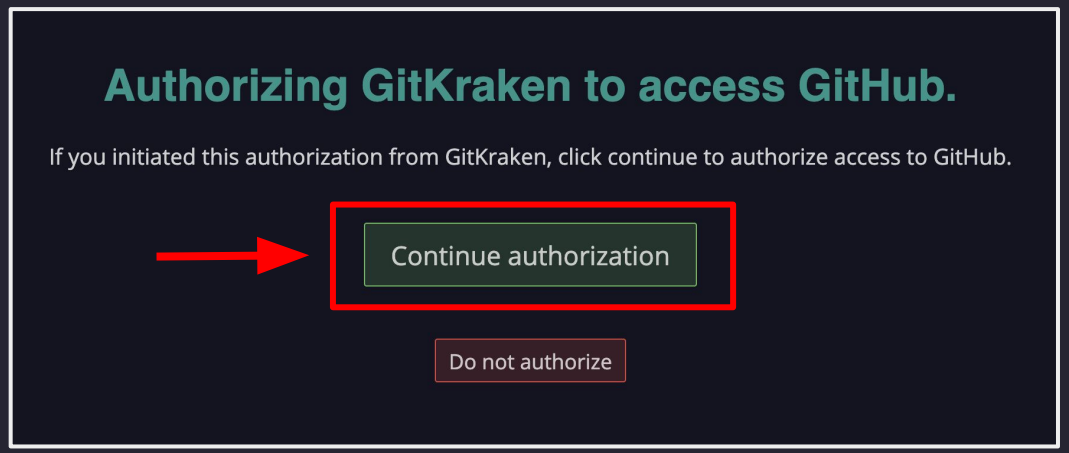
Download the Git Client

6. Download the Git Client for your operating system.
7. Then, follow the installation instructions for your operating system.

Download the Git Client

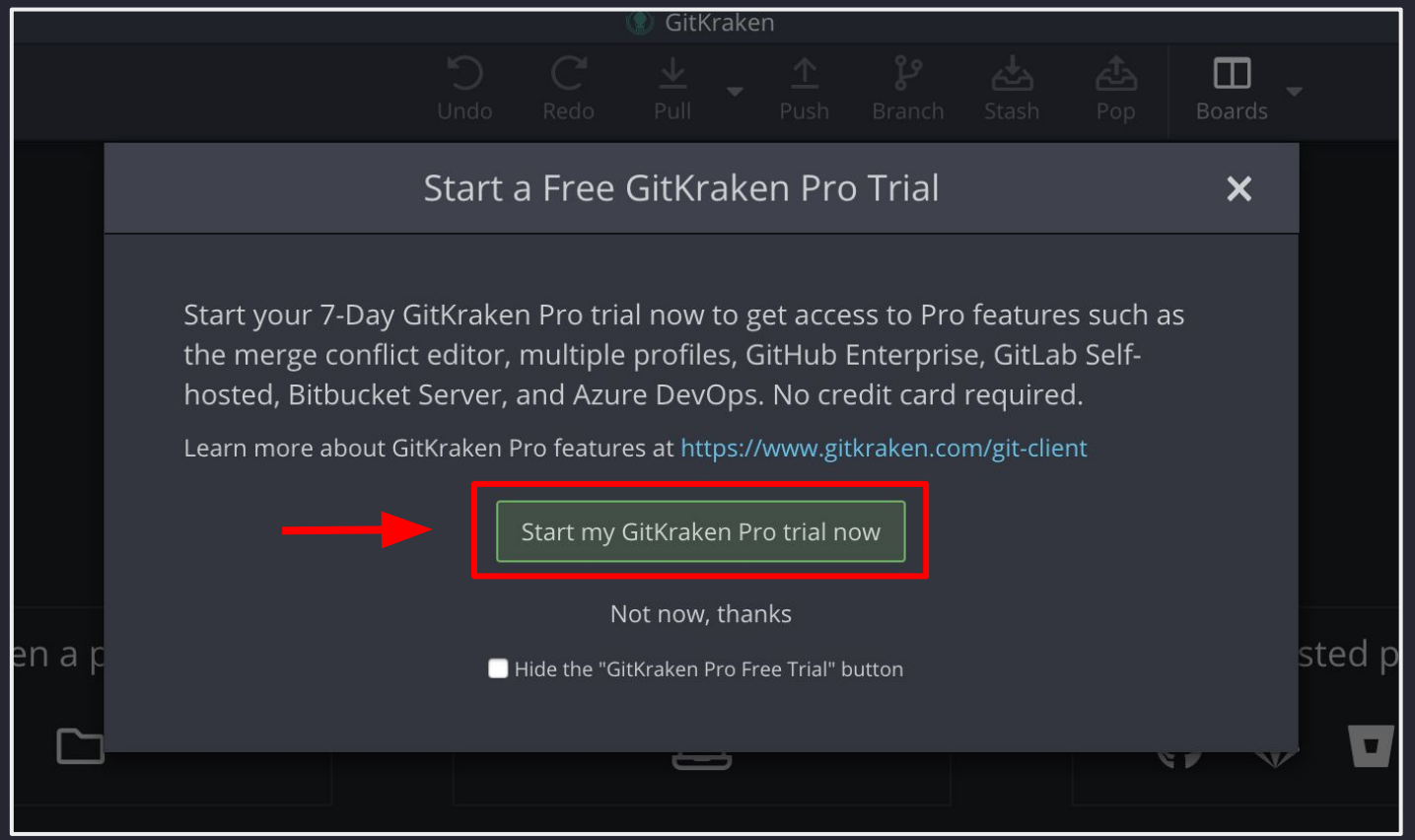
8. Select **Sign in with GitHub** as before, which will open a web browser.
9. Then, select **Continue authorization**.





Download the Git Client

10. If you see this modal, select **Start my GitKraken Pro trial now**. If you don't see this, your GitKraken account was automatically updated to Pro when you made your GitHub Edu account!

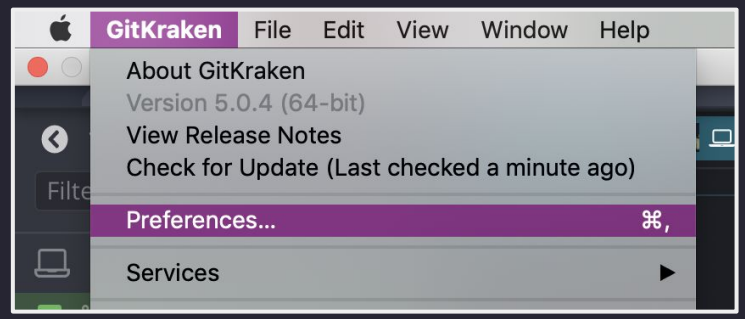


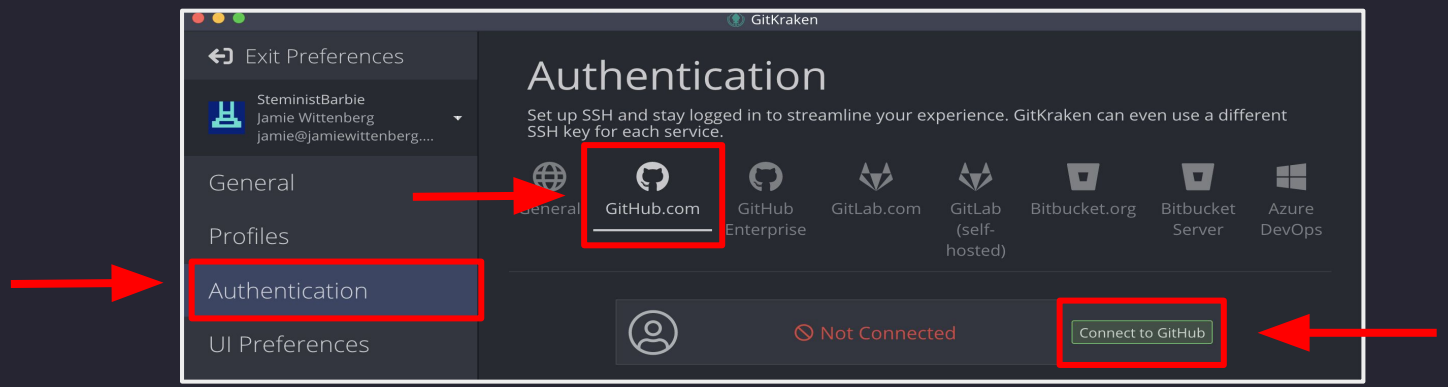
Integrate GitHub

11. To make sure that GitKraken is authorized to work with your GitHub account, open your **Preferences**.

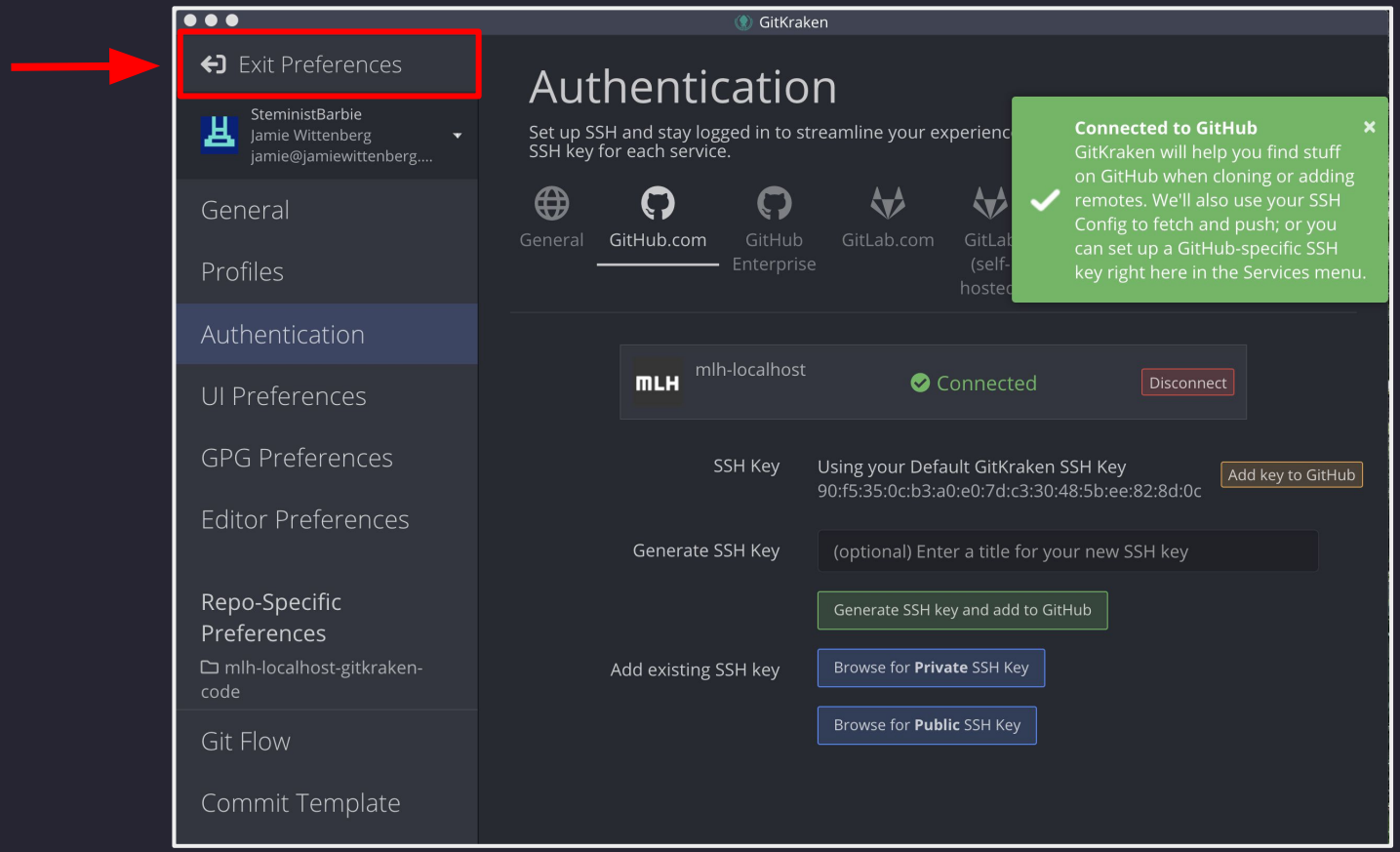
12. Select **Authentication**, then **GitHub.com**.

13. If your GitHub account is not already connected, select **Connect to GitHub**.





Integrate GitHub



Now that you're all set up, let's form a team!

Table of Contents

1. Intro to Git and GitHub
2. Set up GitKraken
3. Collaborate!
4. Review and Quiz
5. Wrap Up and Next Steps

Team Formation!

In order to really learn how to collaborate, you have to work with a team! Teams of 3 – 5 people will work well for this workshop. Group with people around you.

Now that you have your team,
let's get the code.

Get the Source Code

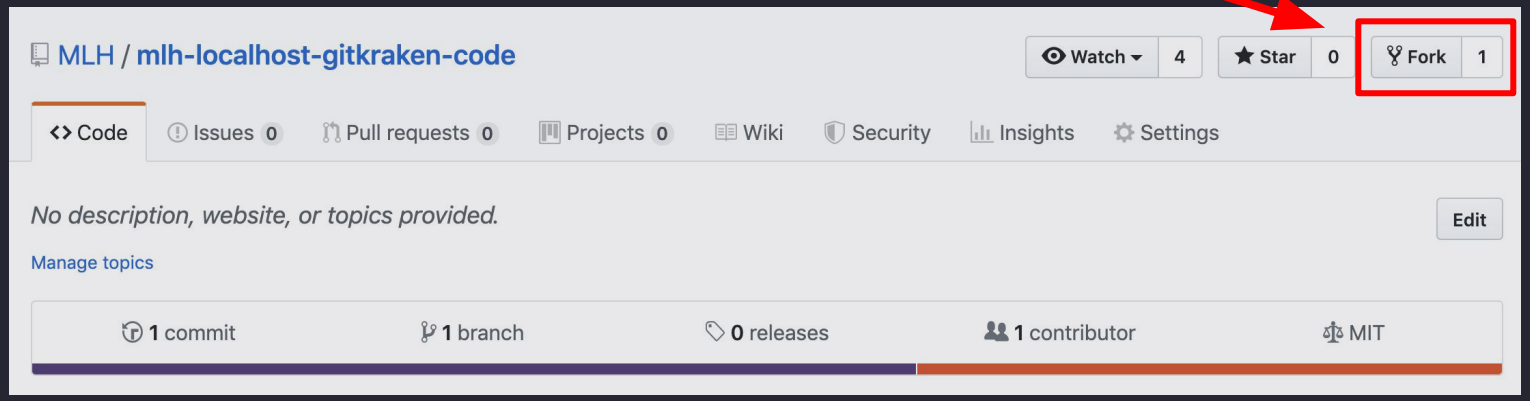
In the next few steps, your team will create a shared repo for your team webpage! Follow these instructions very carefully.

http://mlhlocal.host/gitkraken-code

Key Term

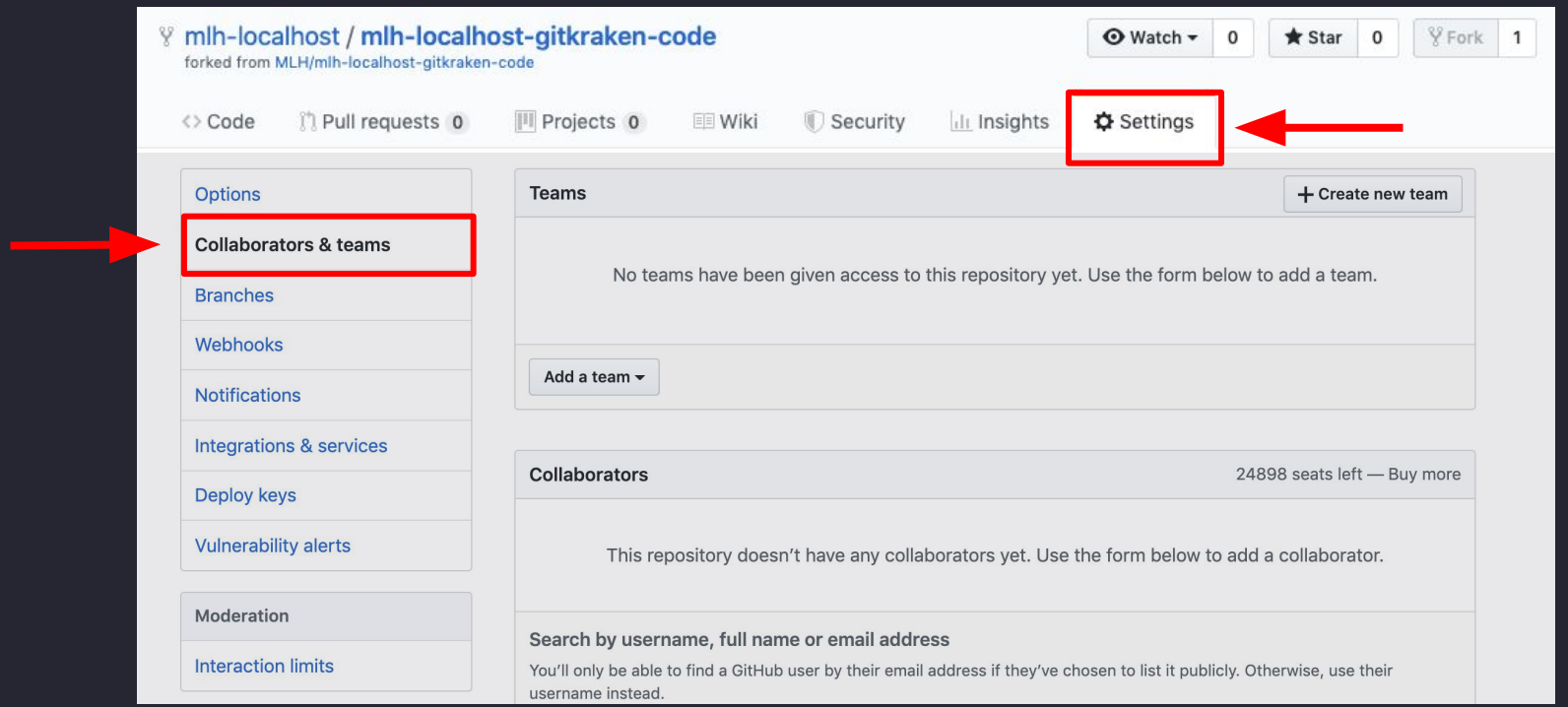
fork: create a copy of a Git repository

1. Pick **one** of your team members to navigate to the URL above.
2. This team member will fork the repository.



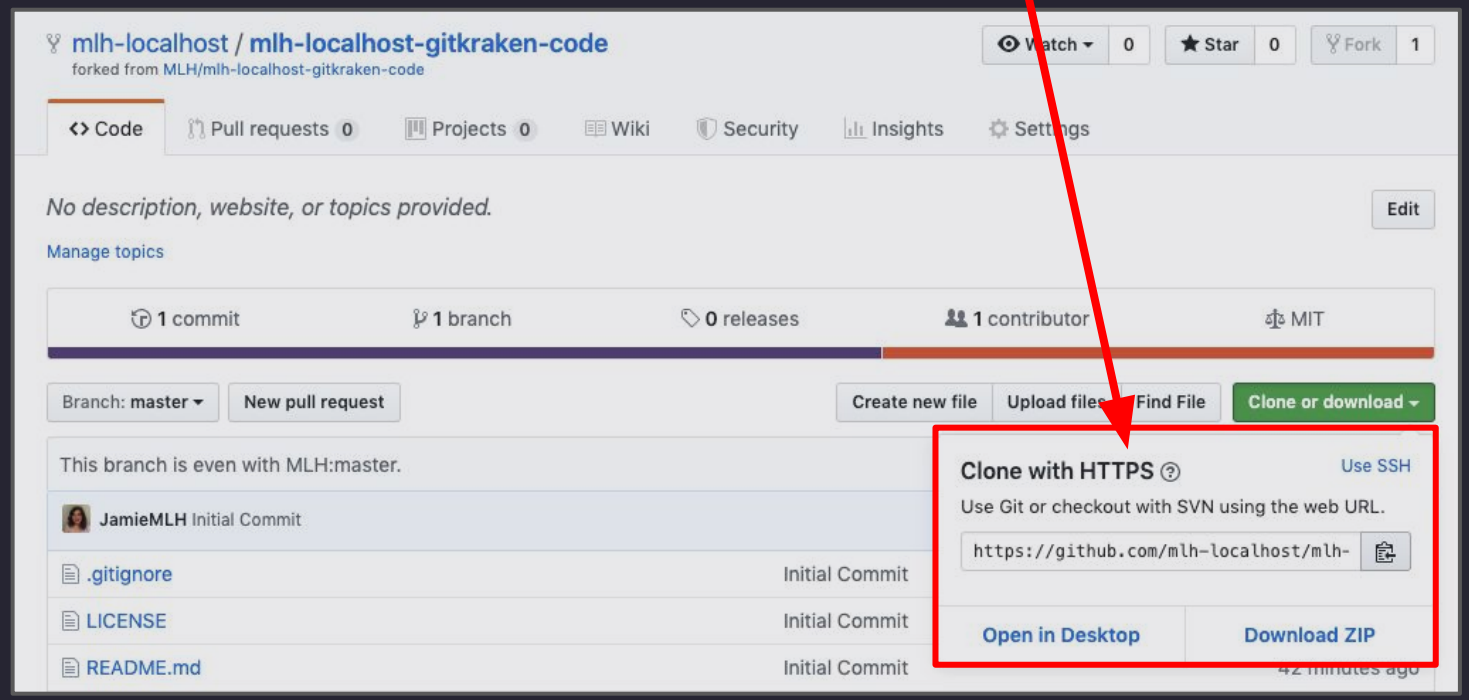
Get the Source Code

3. The same team member needs to add the other team members as collaborators on the repository. Click **Settings** -> **Collaborators** at the top of your fork of the repo.
4. Add each person's GitHub username under Collaborators.



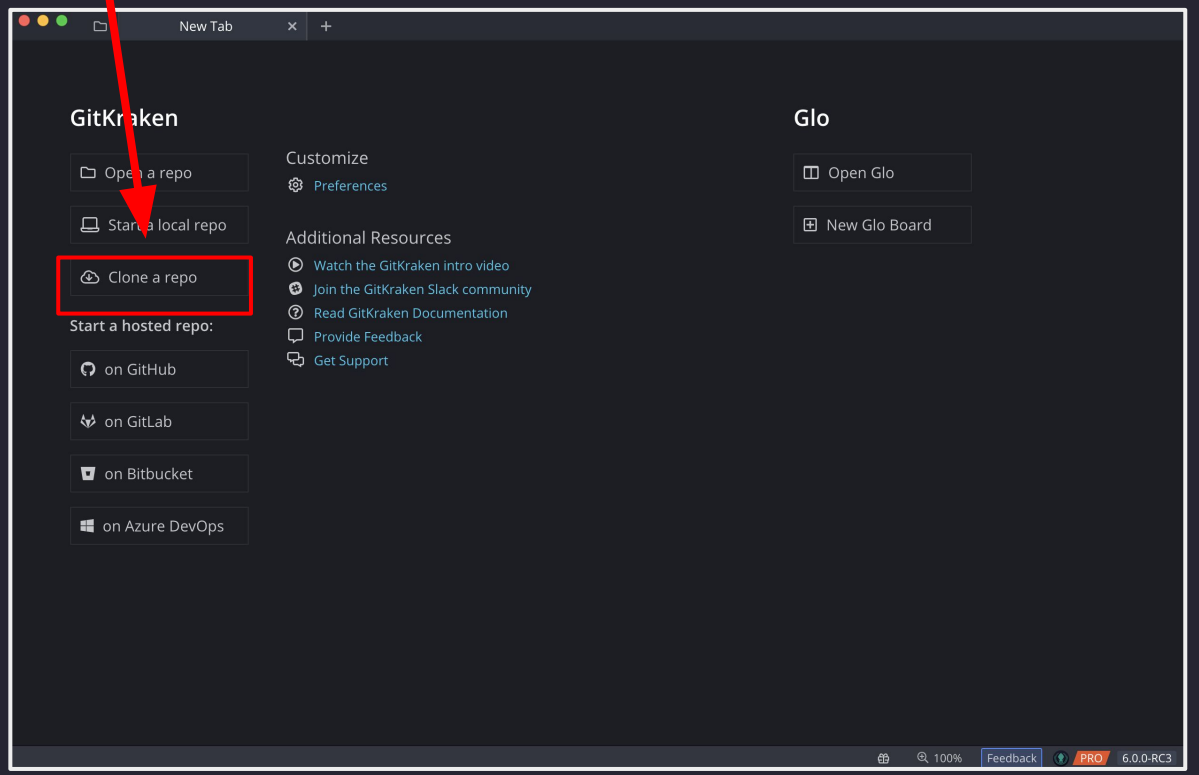
Get the Source Code

5. Then, each person, including the first, should copy their clone URL.



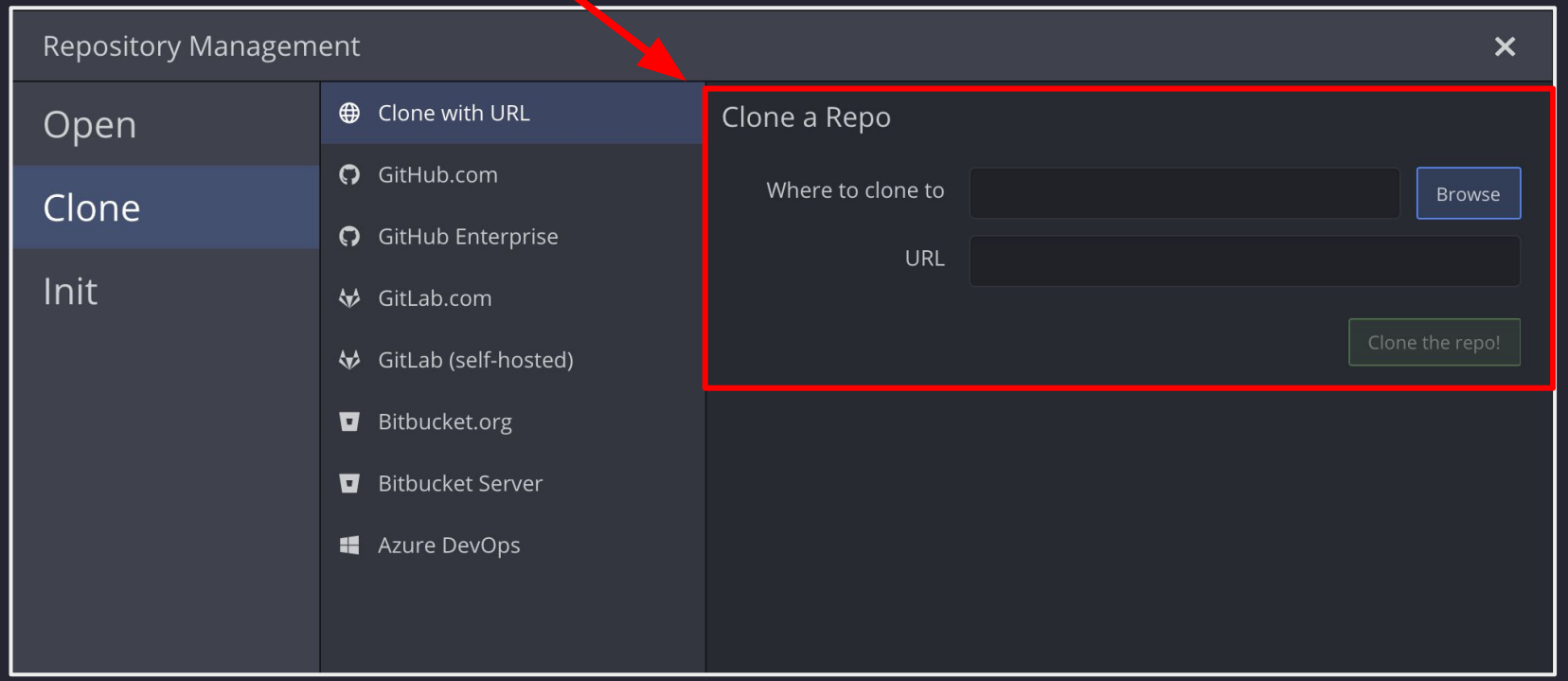
Get the Source Code

6. Each team member should return to GitKraken.
7. Select **Clone a repo**.



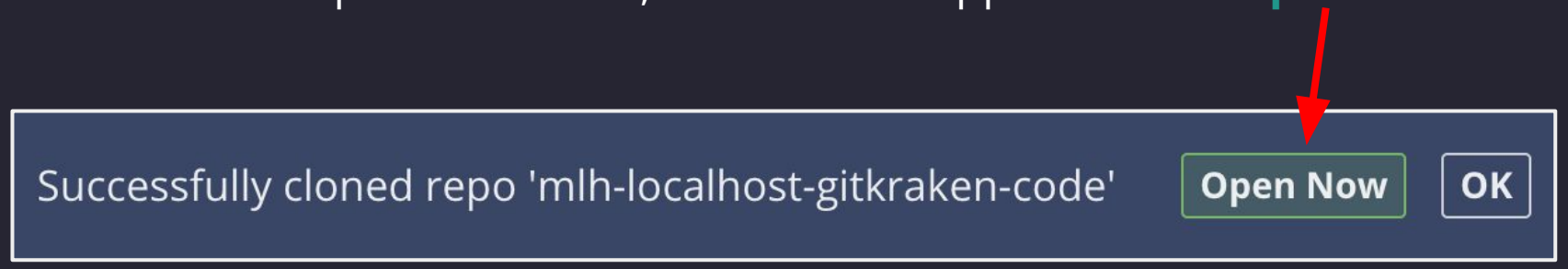
Get the Source Code

8. Browse for where you'd like the project to be stored on your computer.
9. Paste the clone URL you copied into the **URL** field.
10. Click **Clone the repo!**



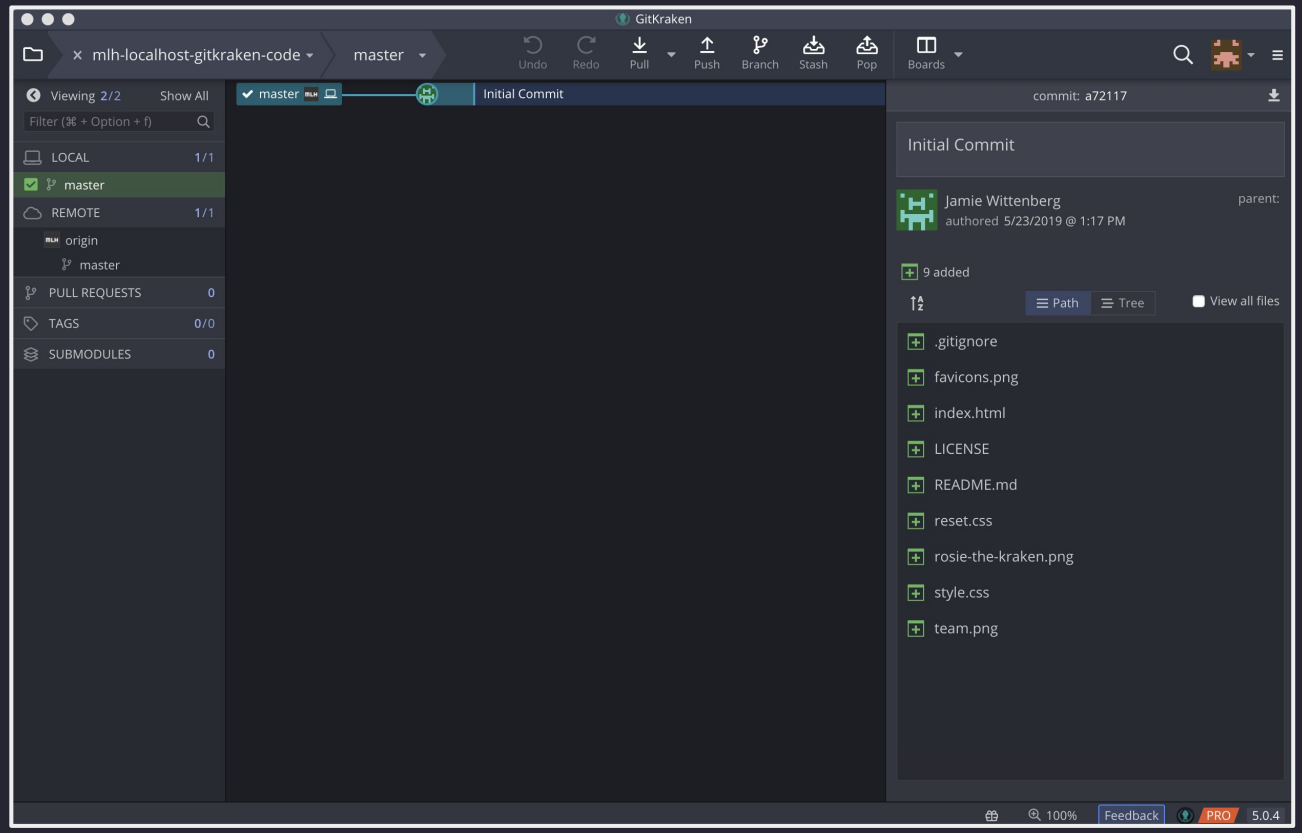
Get the Source Code

11. At the top of the screen, a bar should appear. Select **Open Now**.



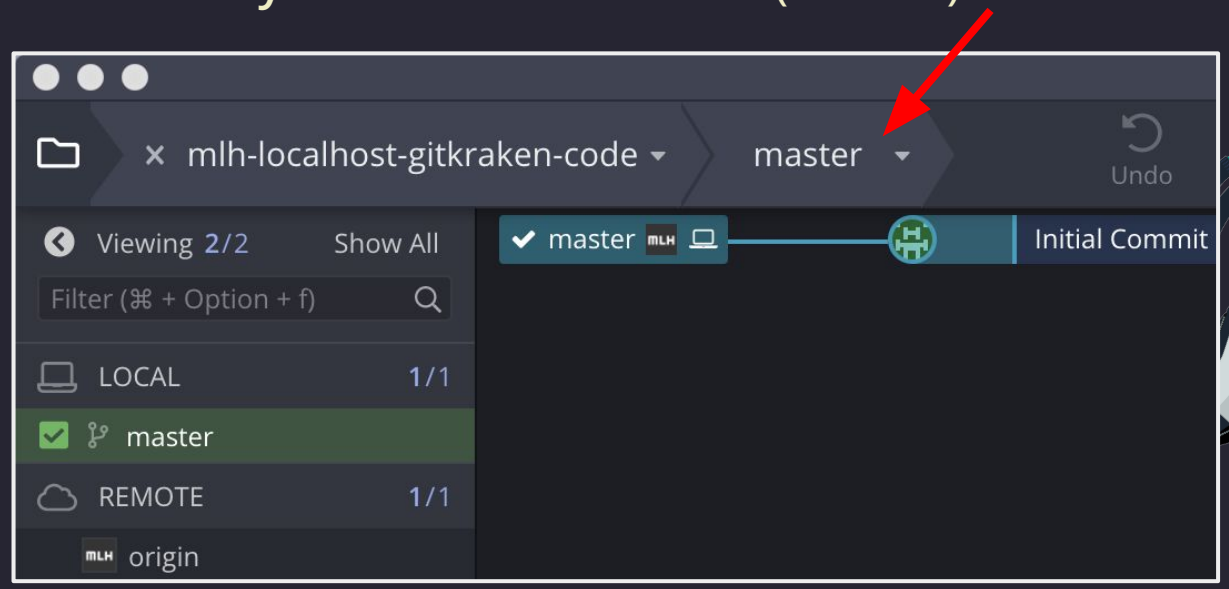
Let's Explore GitKraken!

There are SO MANY awesome features in GitKraken, all of which correspond to parts of the Git workflow. Let's identify a few of them before getting to work.



Let's Explore GitKraken!

In the upper left hand corner, you can see the name of the project you're working on (`mlh-localhost-gitkraken-code`) and the name of the branch you have checked out (`master`).



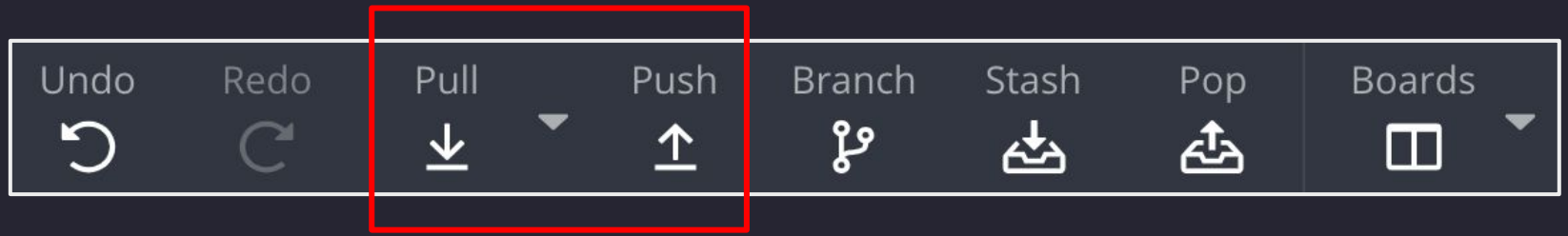
Key Term

branch: a parallel copy of a repository, inside the repository (whereas a fork is a copy of a repository outside of the repo).

Let's Explore GitKraken!

At the top of GitKraken, in the center, you can see a selection of useful options!

- **pull** – to bring changes from the remote down to your local environment. You would use this option if someone else has made changes to the project that you also want in your copy of the project.
- **push** – to add your local changes to the remote.



Key Term

remote: the repository on GitHub (or any other version control platform) that your local repository is associated with

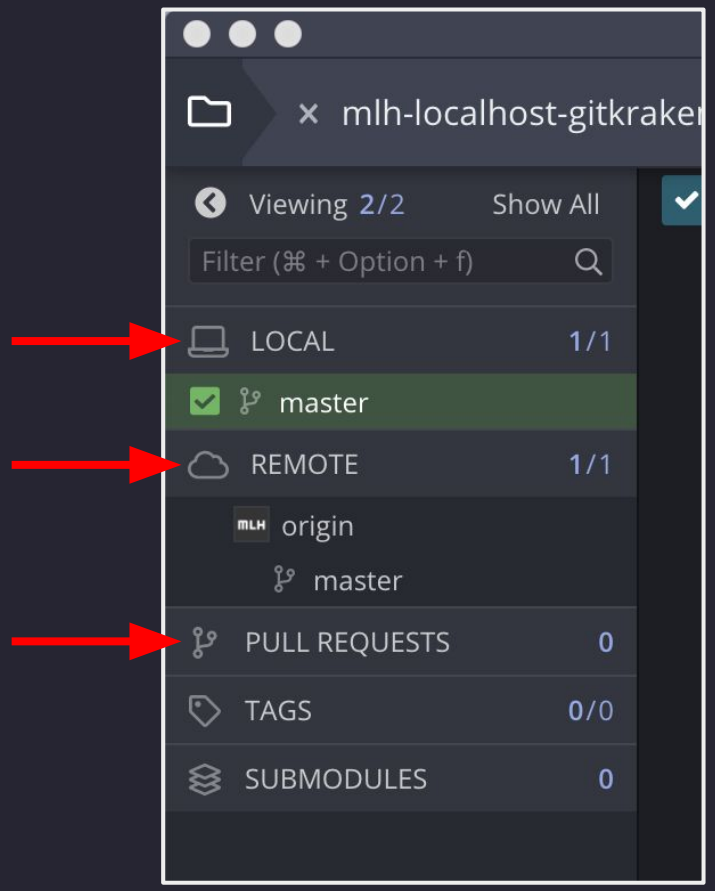
Let's Explore GitKraken!

At the top of GitKraken, in the center, you can see a selection of useful options!

- **branch** – to create a copy of the project inside of the project itself. This allows you to make changes and test them out, without finalizing them until you're satisfied.
- **stash** – when you have changes that you aren't ready to commit, but you want to switch to another branch, you can stash your changes.
- **pop** – to delete your stash from the stack

Let's Explore GitKraken!

On the left side of the window, you can see branches and pull requests.

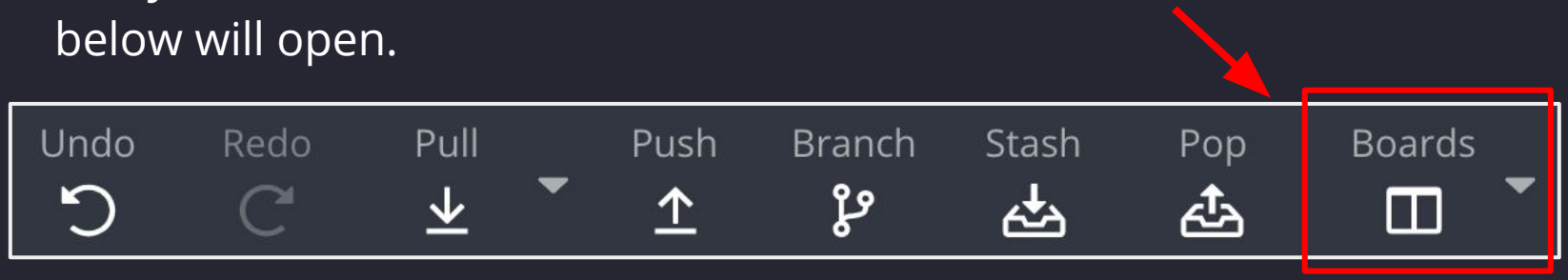


- **LOCAL** – this shows the branches you have in your local environment (on the computer you're using). If you create a branch in GitKraken (or through the command line) that branch will appear here. You can then push it to the remove.
- **REMOTE** – this shows the branches in the remote repository (the hosted repository, like on GitHub). These branches might not always be the same.
- **PULL REQUESTS** – this lists any open pull requests against your repository.

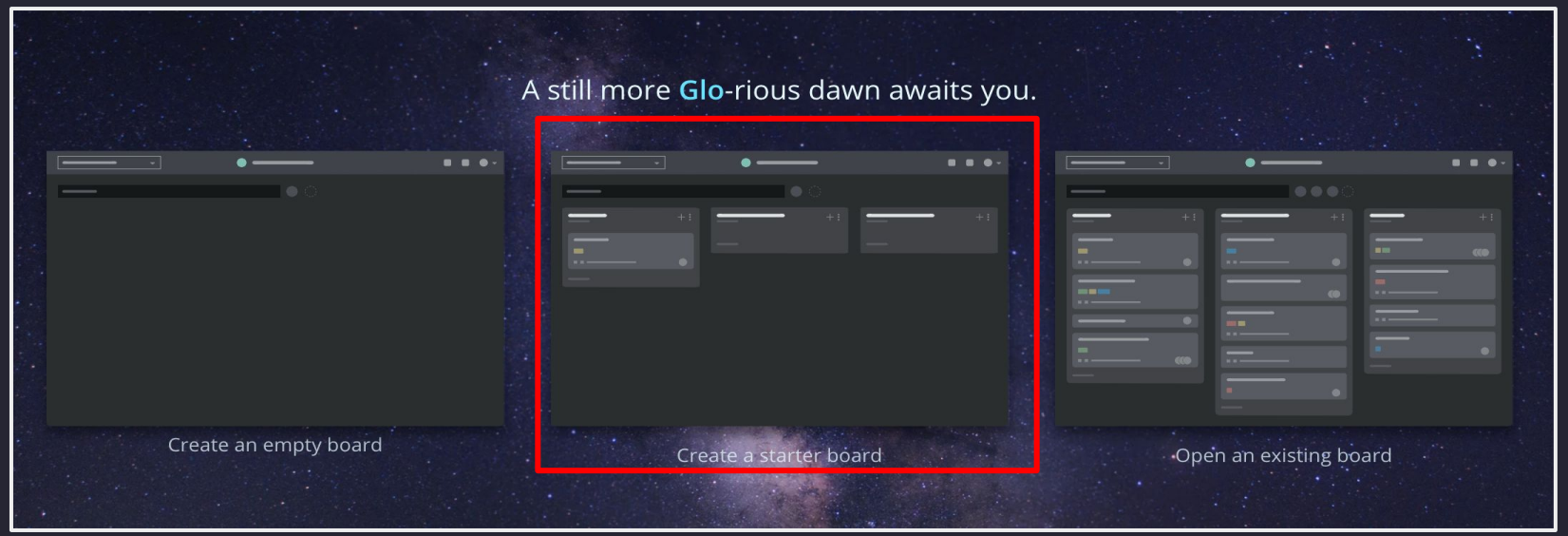
There are a lot of steps coming up! Let's create a team Glo Board to keep track of our progress.

Access Glo Boards

1. Only **one** person on your team should make a board. They will invite everyone else to the board. In the toolbar, select **Boards**. The screen below will open.

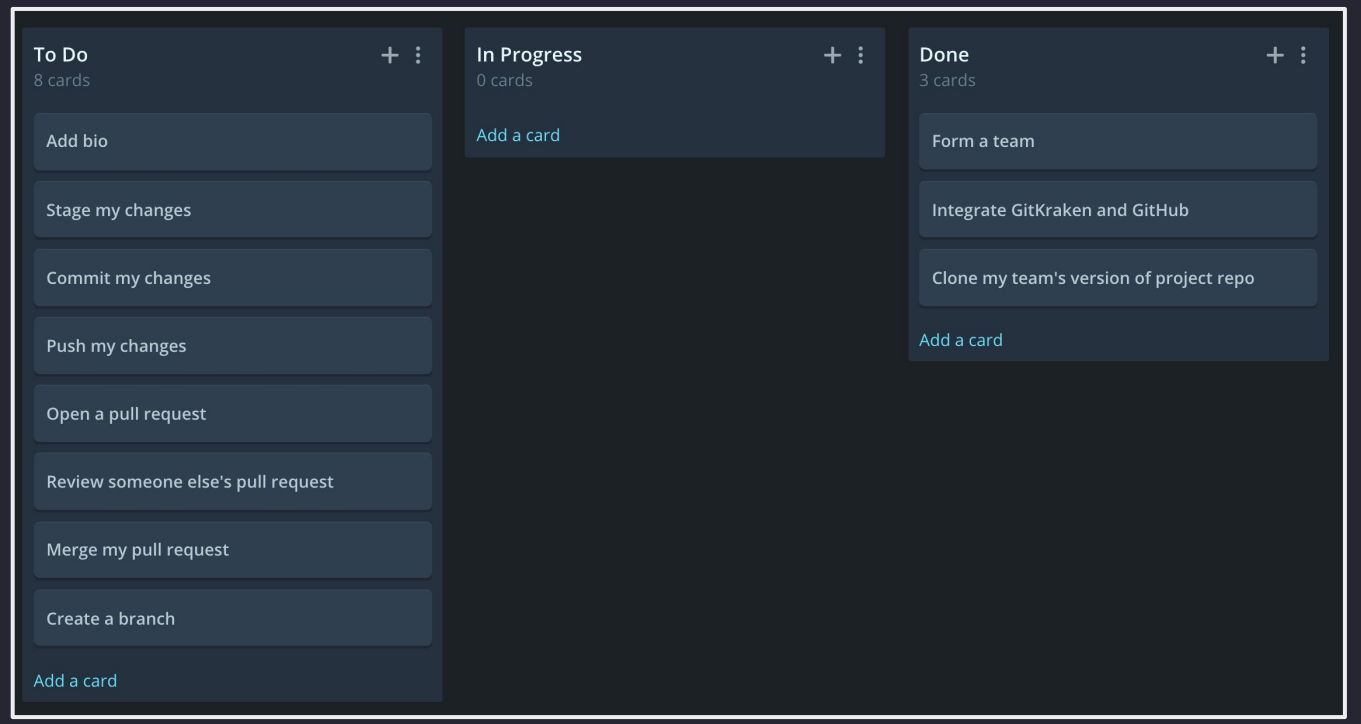


2. Select **Create a starter board**.



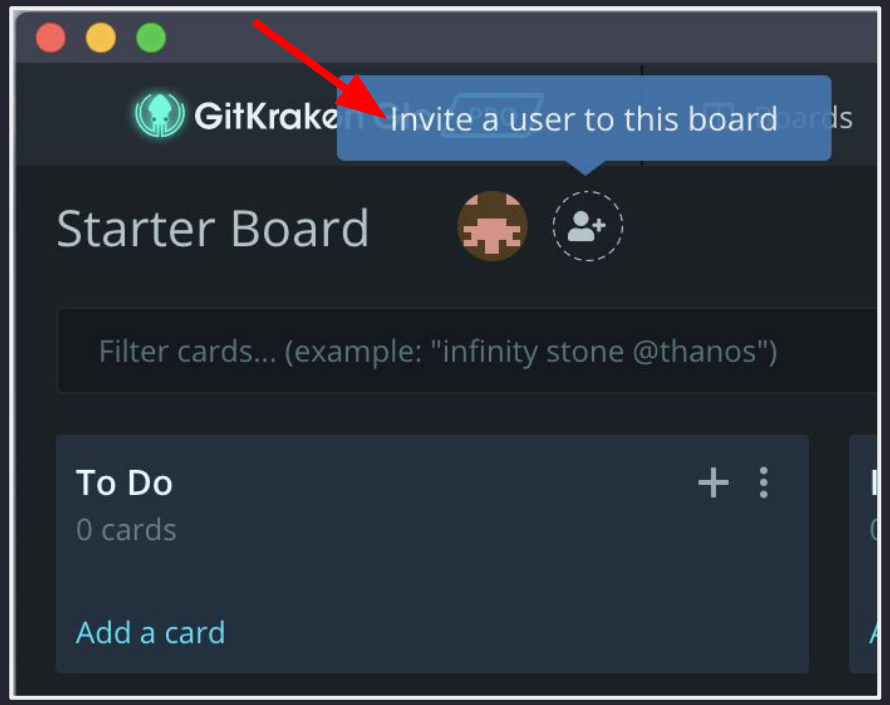
Access Glo Boards

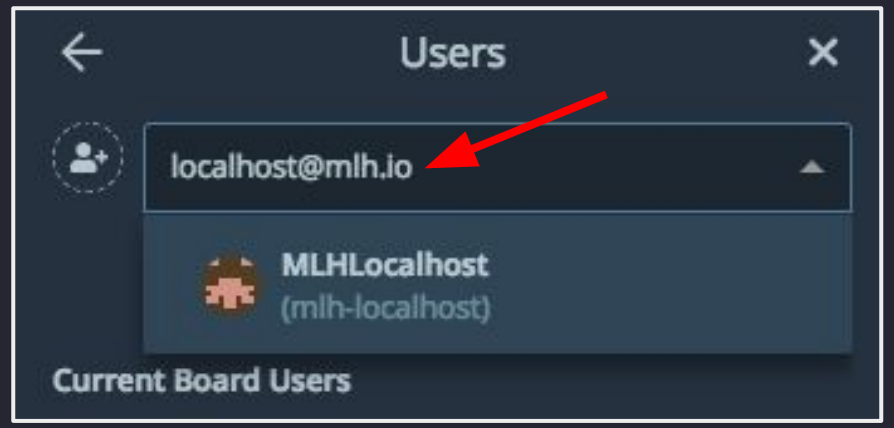
3. Click **Starter Board** to rename the board to something descriptive, like GitKraken workshop, by clicking on the title.
4. Add these cards to your **To Do** list by clicking **+** or **Add a card**
5. Move any steps that you have completed to **Done**, and any steps you're currently working on to **In Progress** by dragging and dropping.

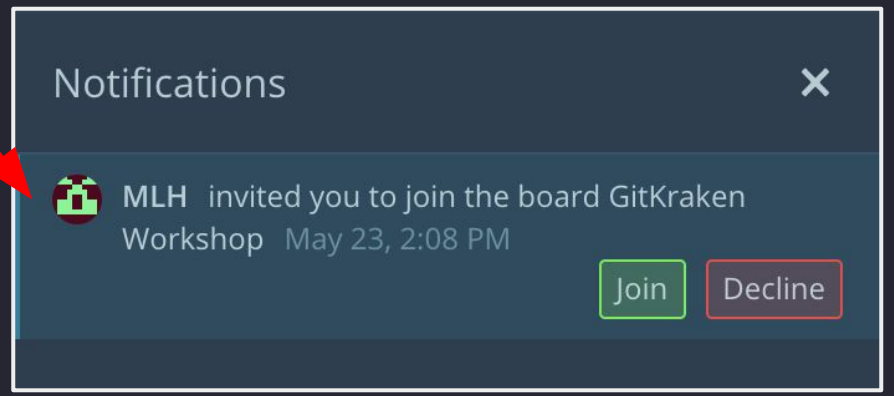


Access Glo Boards

6. Next to the name of the board, you can see who is on the board. Select the icon to add collaborators.
7. Everyone else will get a notification about the invitation to join the board.

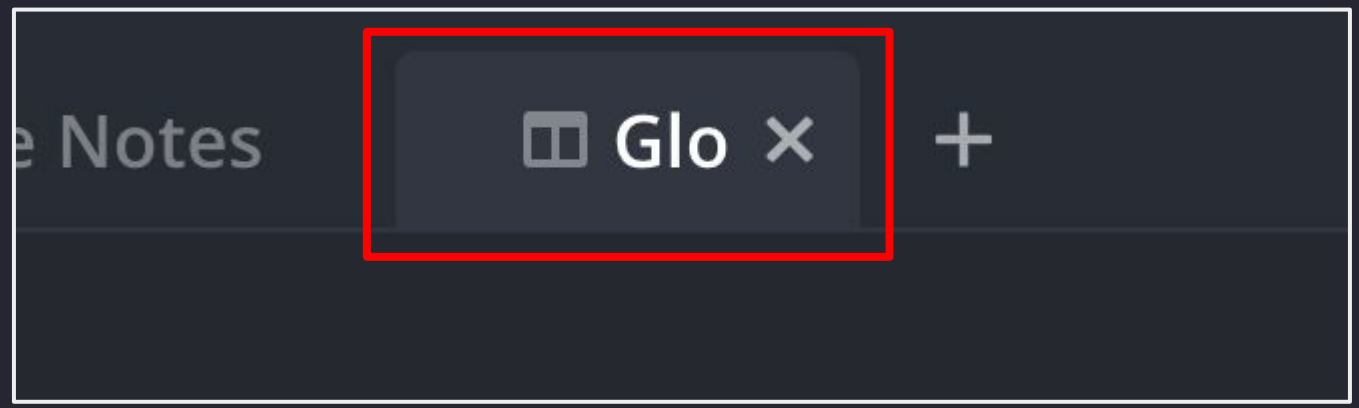






Access Glo Boards

8. Click the Glo tab in GitKraken to refer back to your board as you complete steps!



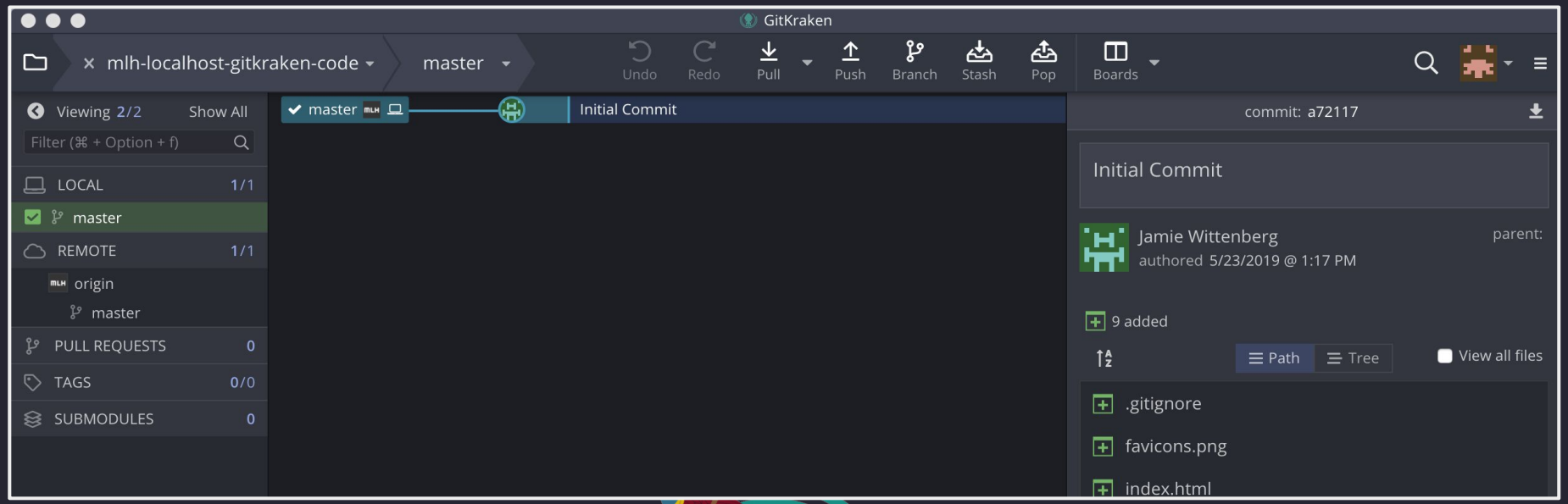
Now that everyone's teams are in place, we're going to clone the source code repo using GitKraken!

Table of Contents

1. Intro to Git and GitHub
2. Set up GitKraken
3. Collaborate!
4. Review and Quiz
5. Wrap Up and Next Steps

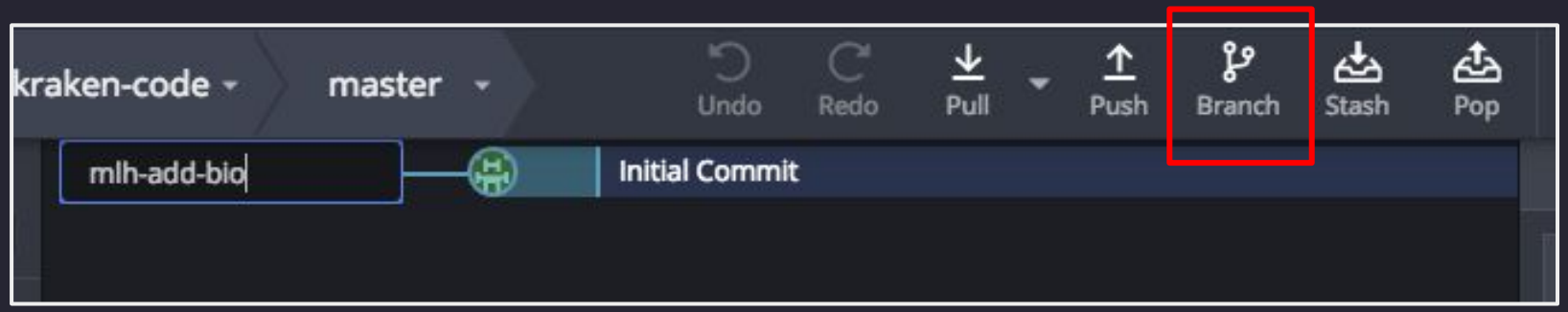
Collaborate

Let's put all these features to use! As you complete each of these steps, you'll switch back and forth between the Git Client and Glo Boards to mark off steps.



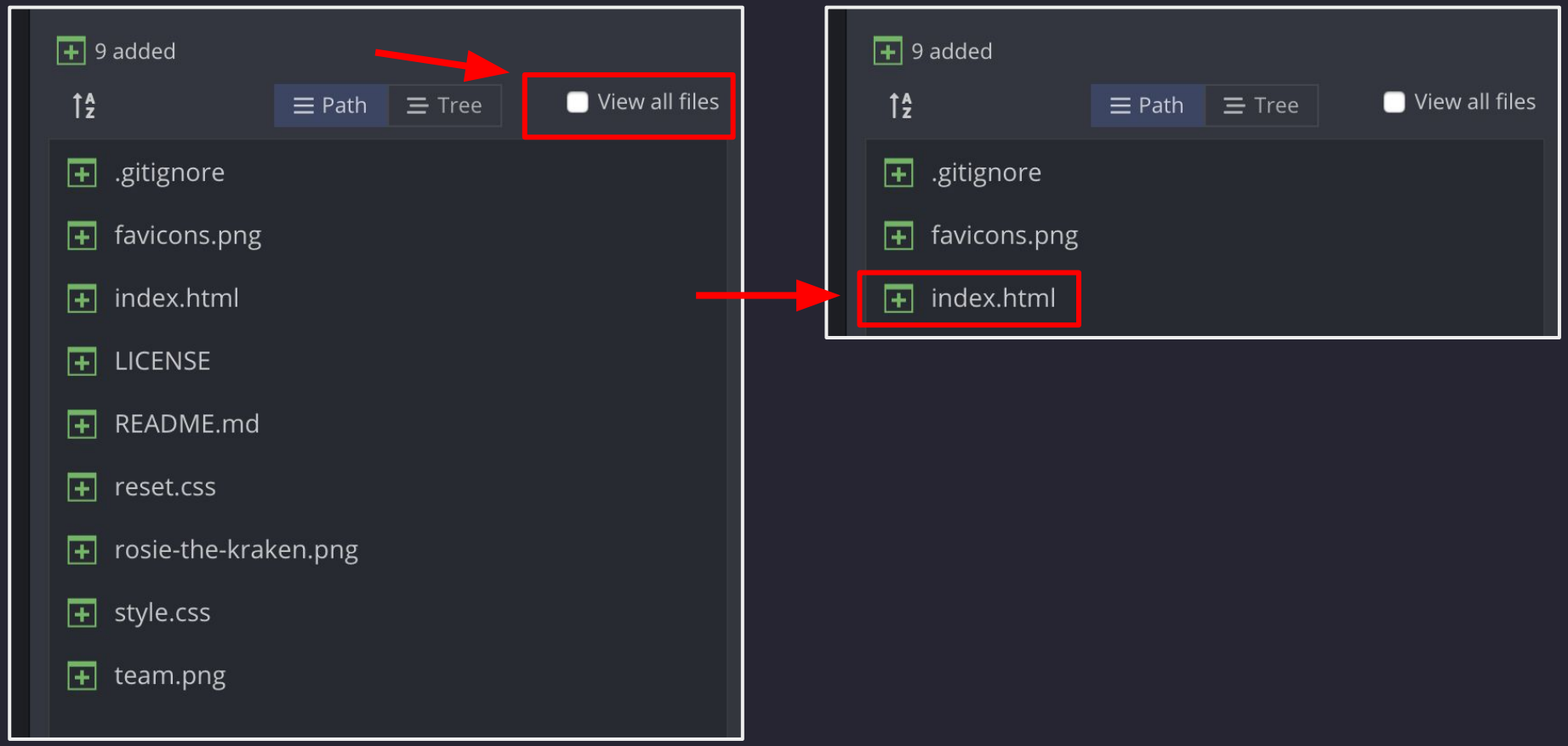
Create Your Own Branch

1. At the top of GitKraken, click **Branch**.
2. Name your branch and hit Enter or Return. You will be switched to your new branch!



Open the Project

1. On the right side of GitKraken, select **View all files**.
2. Then, click **`index.html`** to open it in the GitKraken editor.



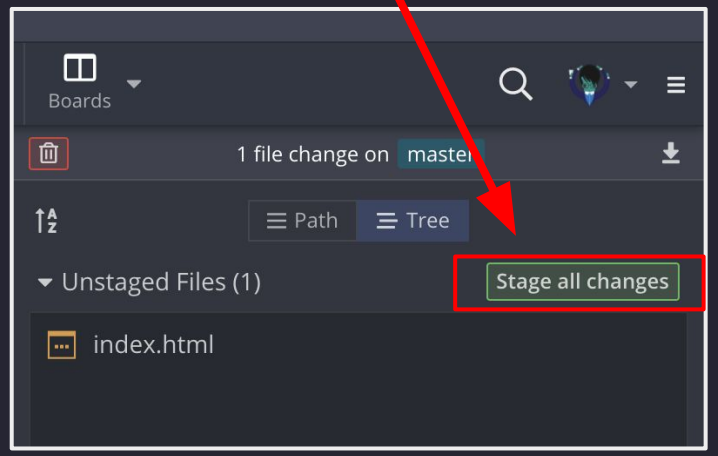
Edit The Project

1. In **`index.html`**, scroll down to line 39.
2. There is a template for adding your personal bio. Copy lines 40 – 48.
3. Paste what you copied directly below line 51.
4. Put your name on line 45. Write a short bio about yourself on line 46. Change the image to be one of yourself, if you wish!
5. Save your work!

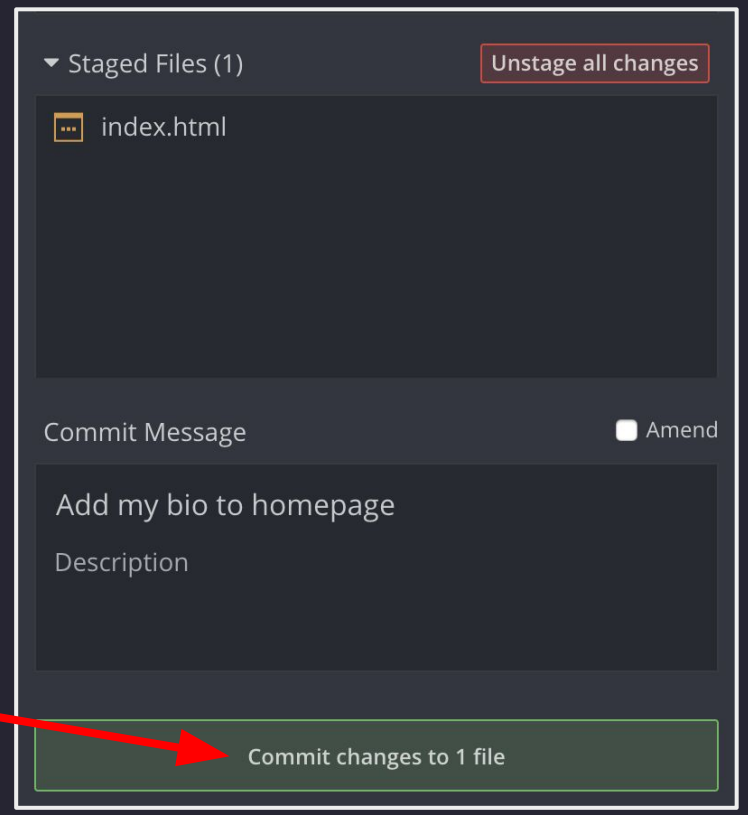
```
39      <!-- copy from here -->
40      <div class="about">
41       <div class="bio-left">
42        <img src="rosie-the-kraken.png"/>
43       </div>
44       <div class="bio-right">
45        <h2>Rosie the Kraken</h2>
46        <p> I'm Rosie the Kraken! In the 1940s, I encouraged women to join the World War II effort in the United Sta
47       </div>
48      </div>
49      <!-- to here  -->
50
51      <!-- paste what you copied directly below here -->
```

Commit Your Changes

1. **`index.html`** is listed under **Unstaged Files**. This means you have changes locally, but have not added them to a commit. Select **Stage all changes**.
2. Your file has moved from **Unstaged Files** to **Staged Files**. Type a descriptive commit message.



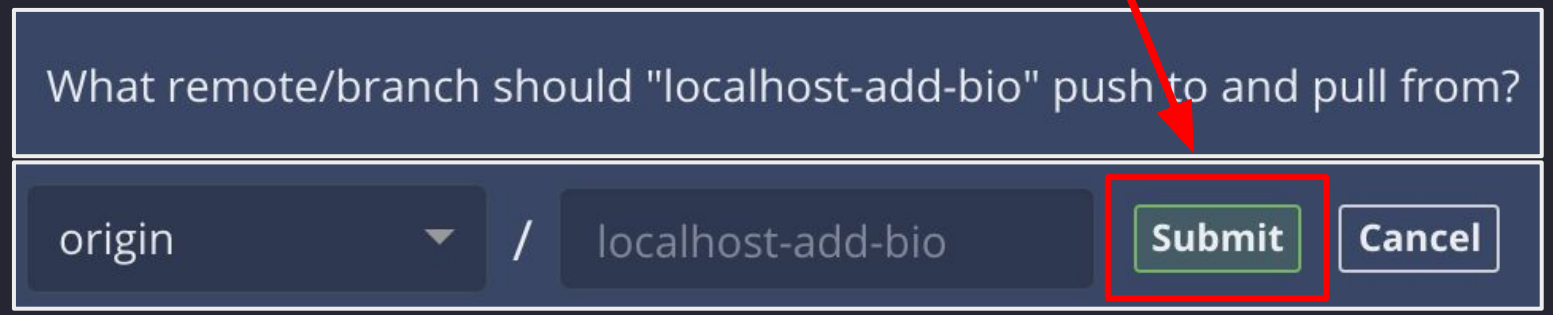
3. Select **Commit Changes to 1 file**.



Commit Your Changes

4. Now you have a new commit in your history!
5. Click **Push** to push your changes. This sends them to GitHub.

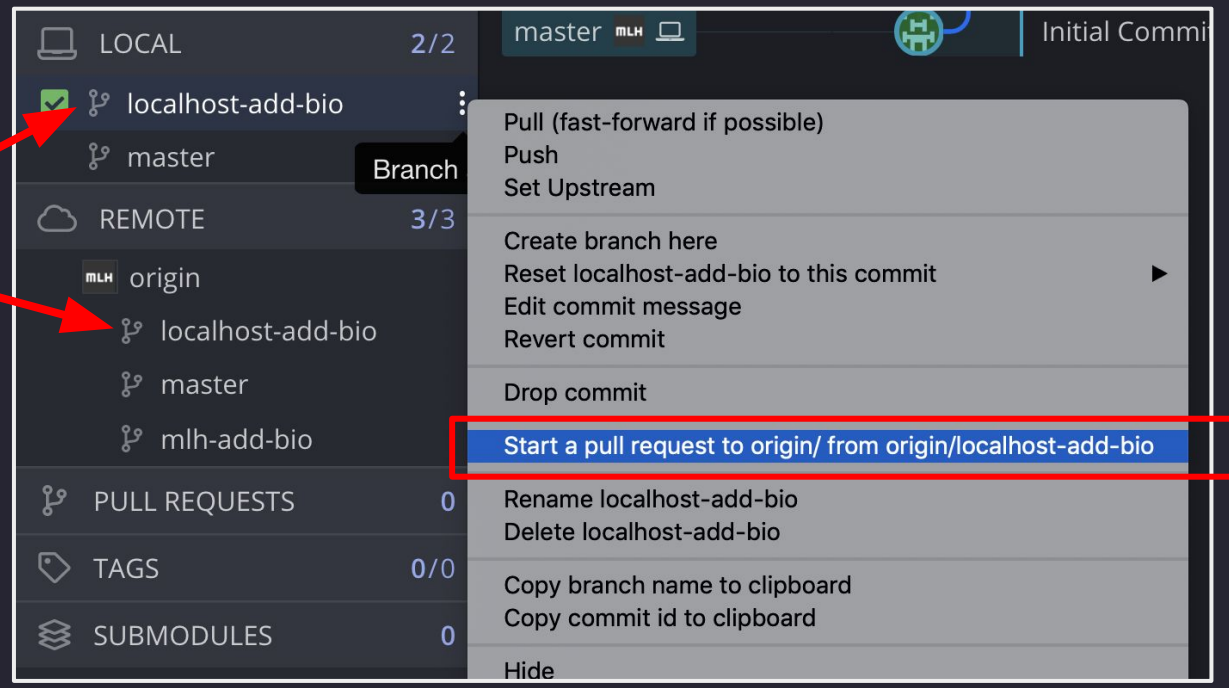
6. The first time you push your changes from a new branch, you might see the notification below at the top of GitKraken. Click **Submit**.



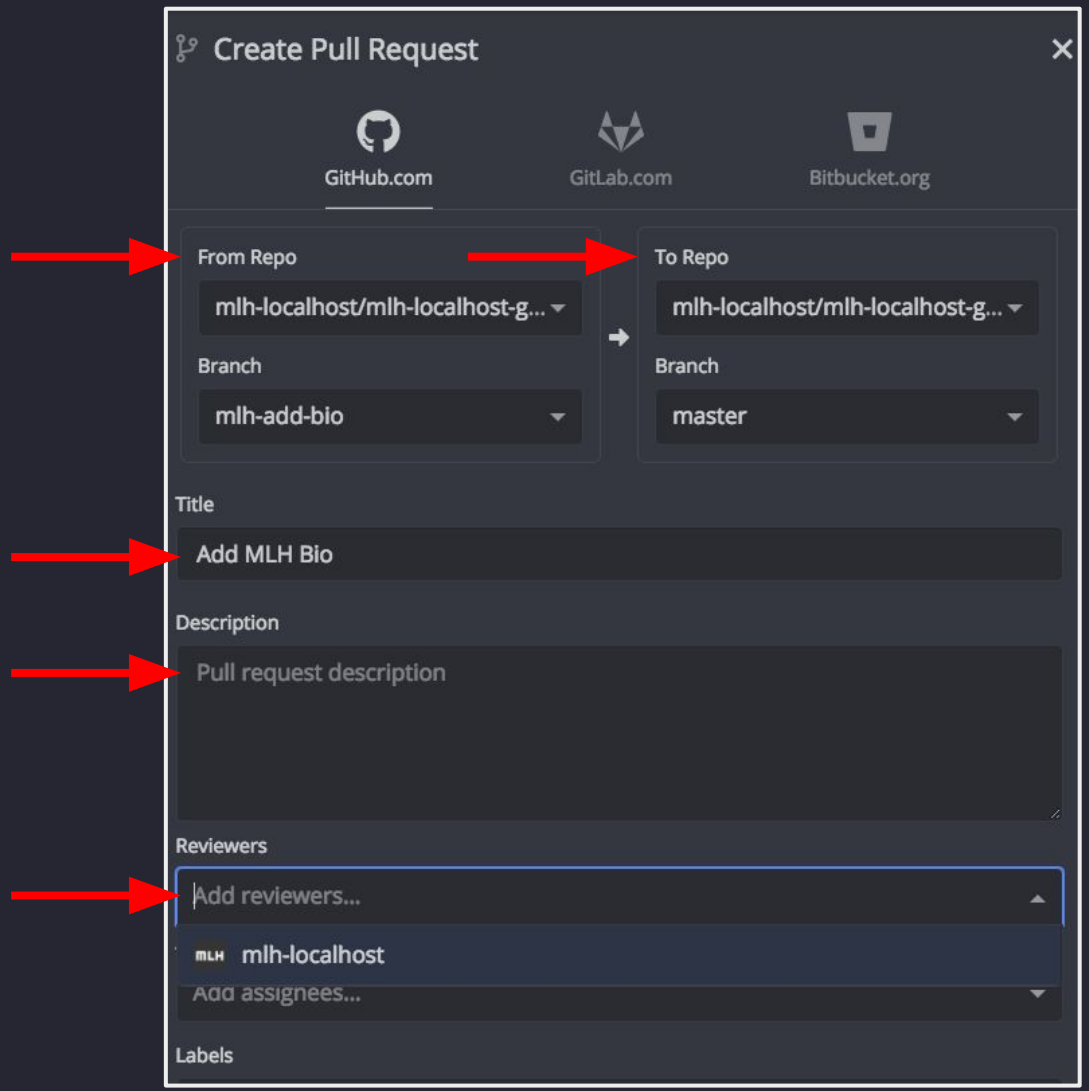
Open a Pull Request

1. Now you're ready to open a pull request to incorporate your changes. Click the name of your branch.
2. Click **Start a pull request** . . .

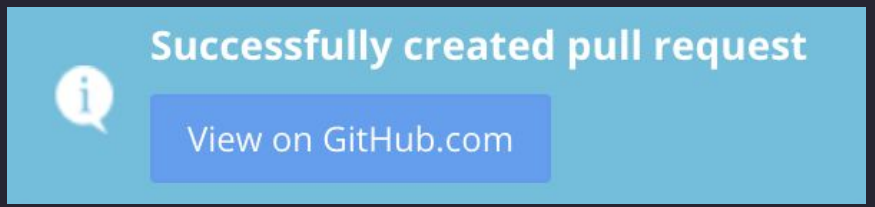
Note: You might notice that the name of your branch is now listed under LOCAL and REMOTE. You might notice some branches on REMOTE that you don't have locally – those are your teammates' branches!



Open a Pull Request

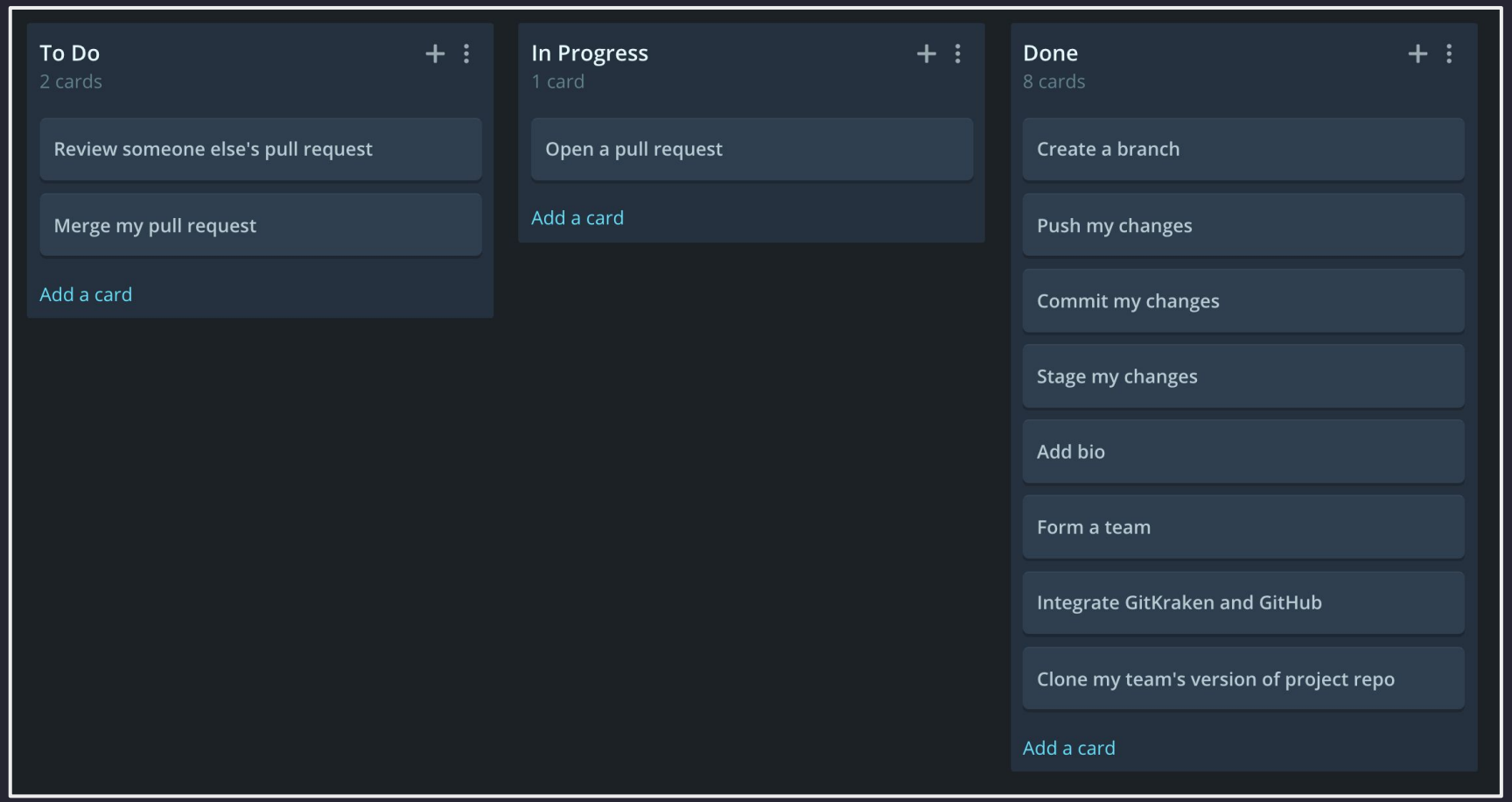


3. The branch under From Repo is the branch you made at the beginning of the workshop. The branch under To Repo is master. You're asking the owner of the repo to pull your changes into master.
4. Add a title for your pull request and a brief description.
5. Assign someone on your team as a reviewer.
6. Select **Create Pull Request** and the pop-up below will appear.



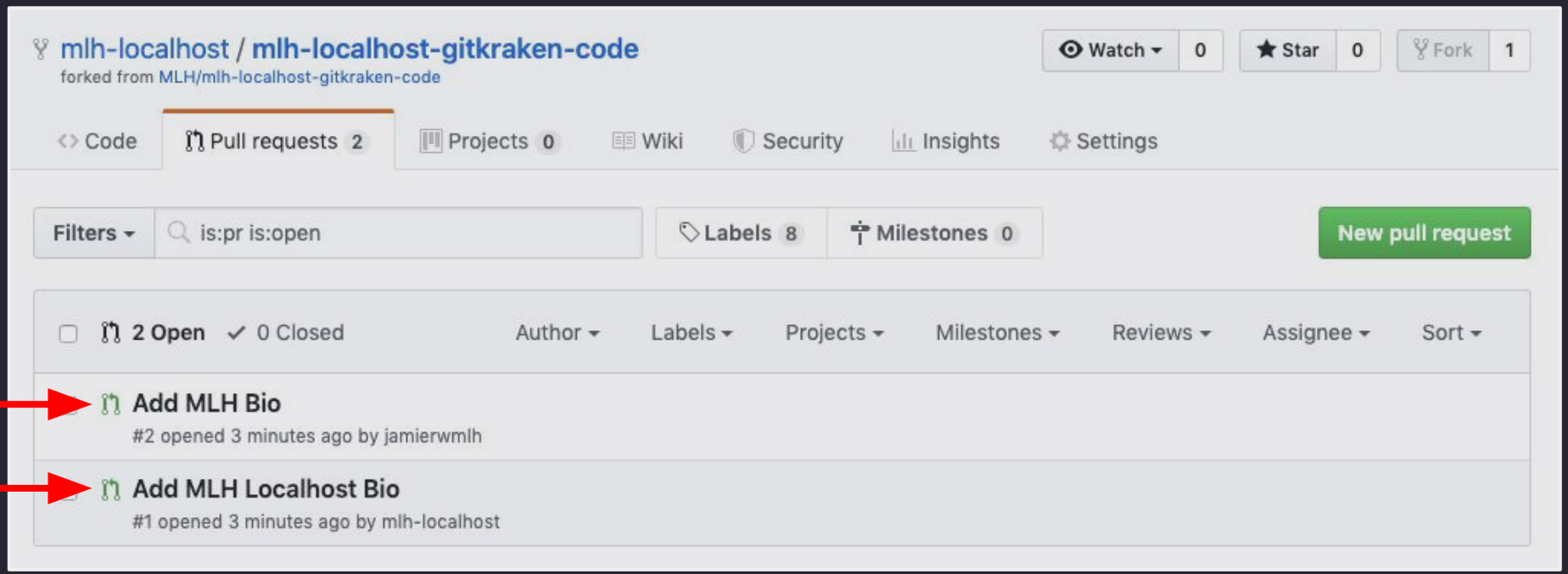
Check Your Progress

1. Take a moment to switch back to your Glo Board and update your tasks.



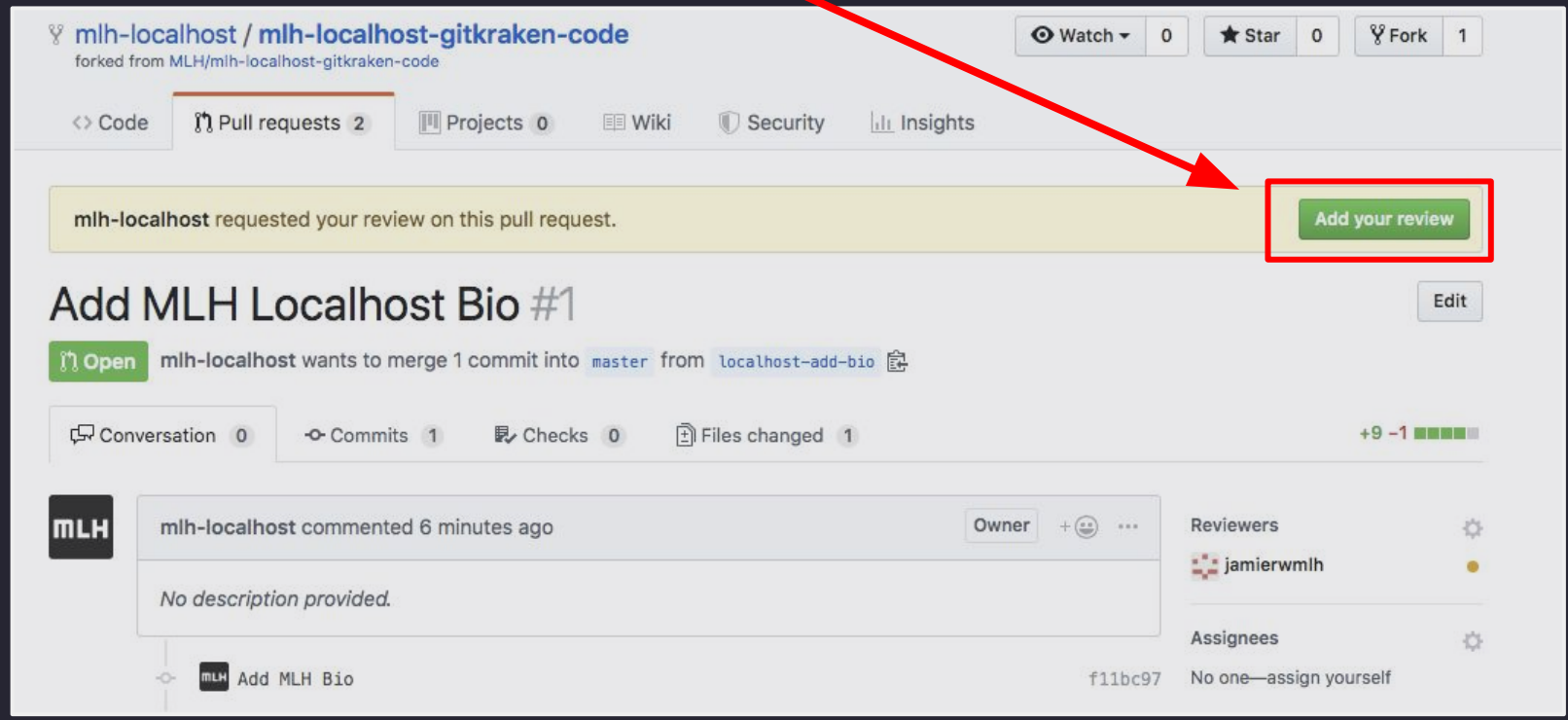
Review Each Other's Pull Requests

If you visit your repo on GitHub and select the Pull Requests tab, you can see your pull request and those of anyone else on your team who has already made their PR.



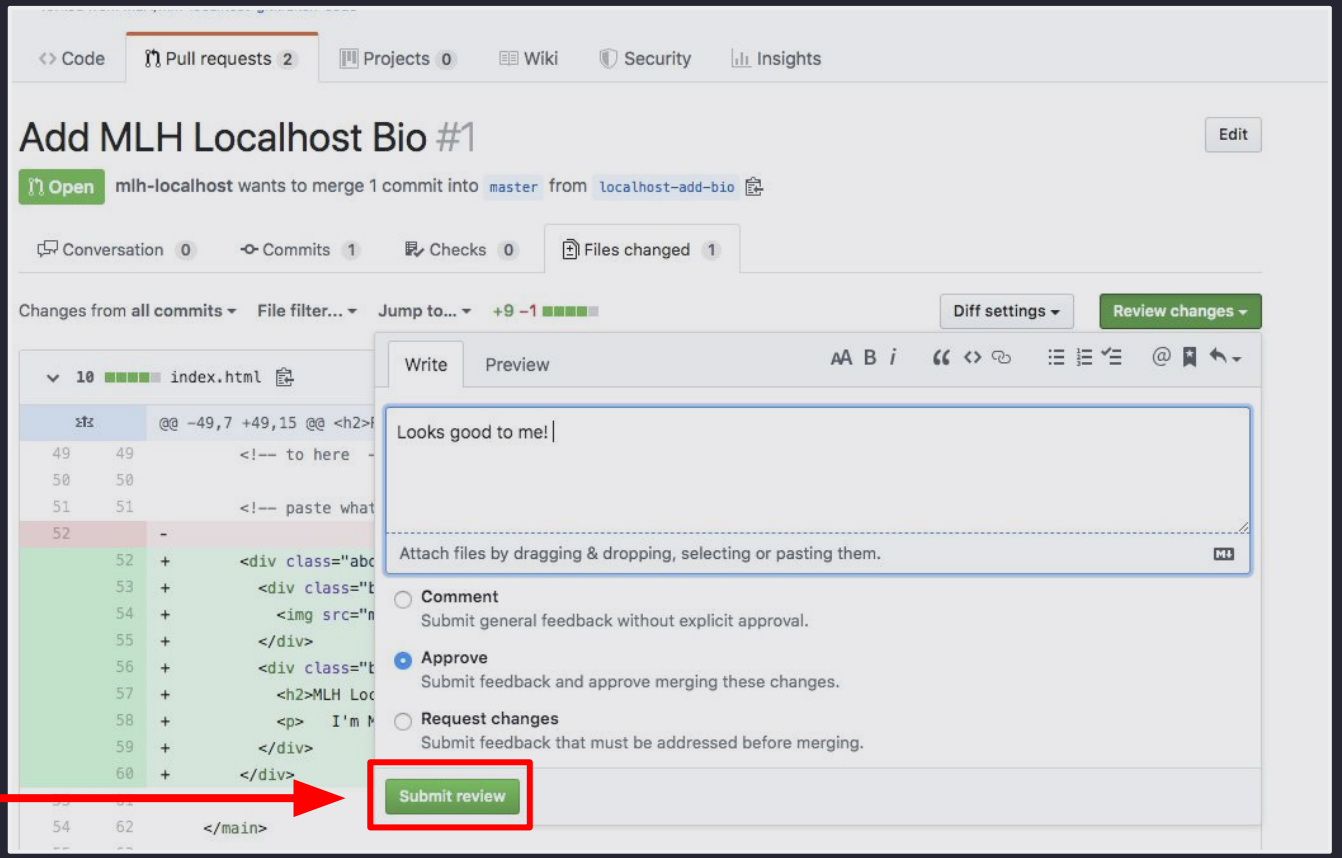
Review Each Other's Pull Requests

1. Select the pull request of your teammate who assigned you as reviewer. You should see this at the top of the screen:
2. Select **Add your review**.



Review Each Other's Pull Requests

3. You have several options. You should read over their changes, leave a comment, and select **Approve** if you want them to merge their changes.
4. Select **Submit** review when you're done.



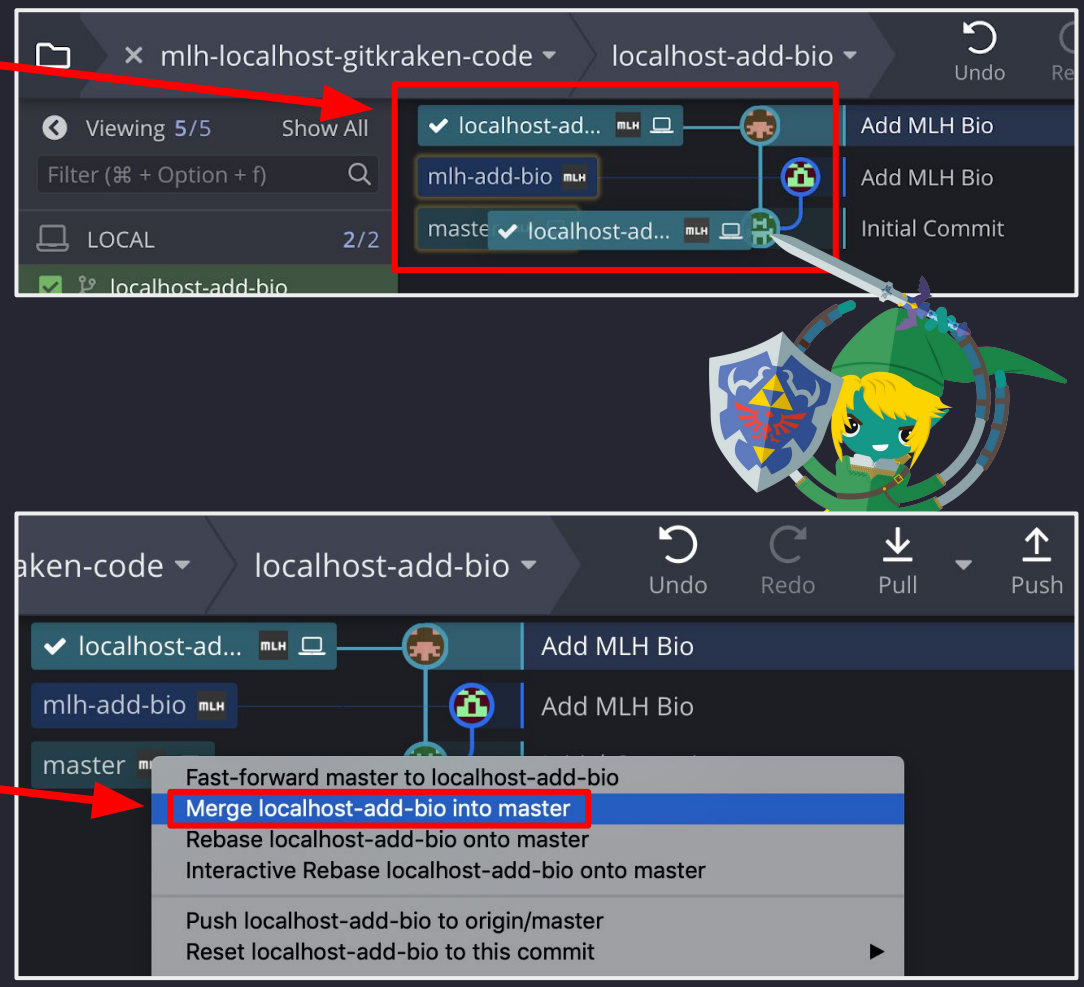
Now that you have reviewed each other's changes, it's time to merge everyone's changes into master!

The next steps are very tricky, so start with one team member.

Merge Your Changes

1. Pick any one of your team members to follow this step. Everyone else should wait for the next few slides. Back in GitKraken, look at the commit tree. Select your branch name and drag it onto master.

2. When the menu opens, select **Merge <your branch name> into master**.



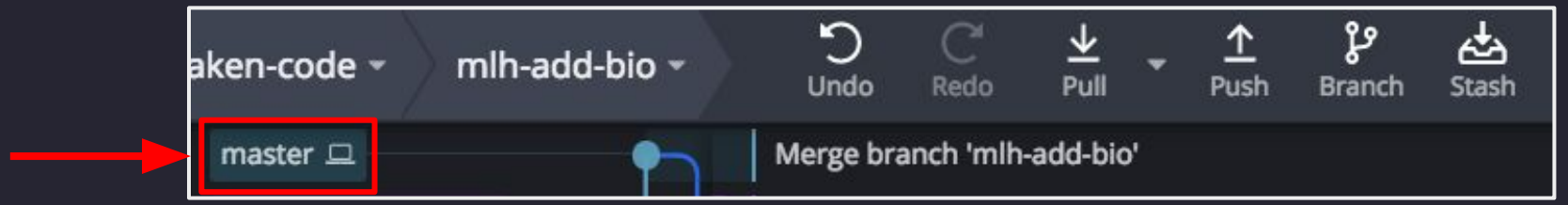
Merge Your Changes

Let's take a look at the commit tree!

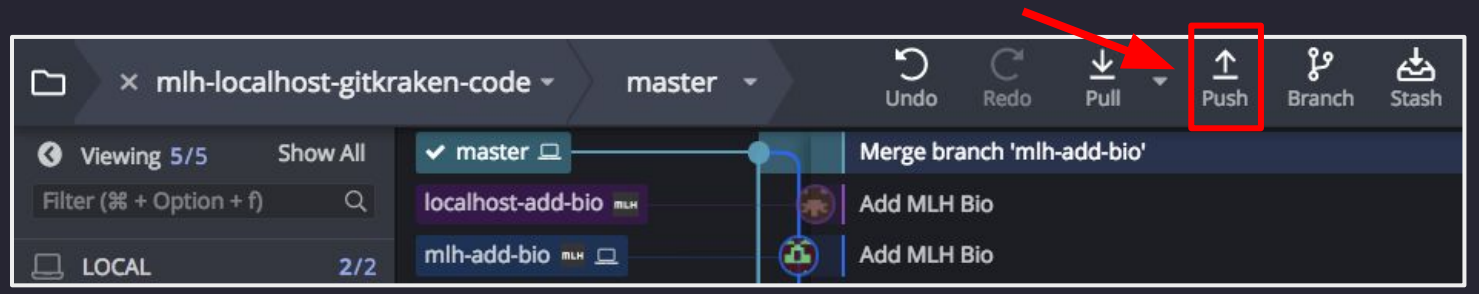
- Notice that at the very bottom, you see master, and you also see master at the top. The higher master has a computer icon, and the lower master has the avatar of the repo's owner.
- Any branch that has the computer icon is your local version of the branch. Any branch that has the repo owner's avatar is on the remote.
- We want our version of master to be on the remote. Let's do that now!

Merge Your Changes

1.Double click the local master branch. You will be switched to that branch.



2.Click **Push** at the top.

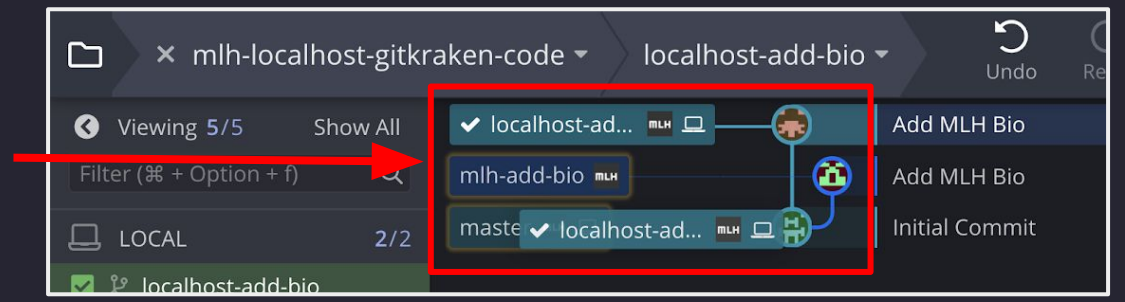


The local version of master disappeared and the remote moved to the top!

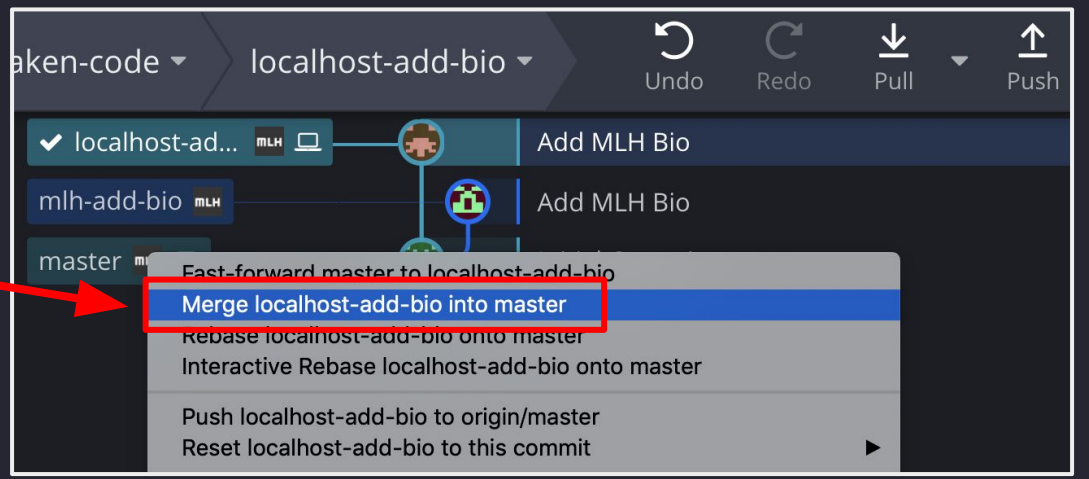
Now, everyone else is going to do the same!
There will be a few extra steps.

Merge Your Changes

1. The steps to merge start the same way. Select your branch name and drag it onto master.

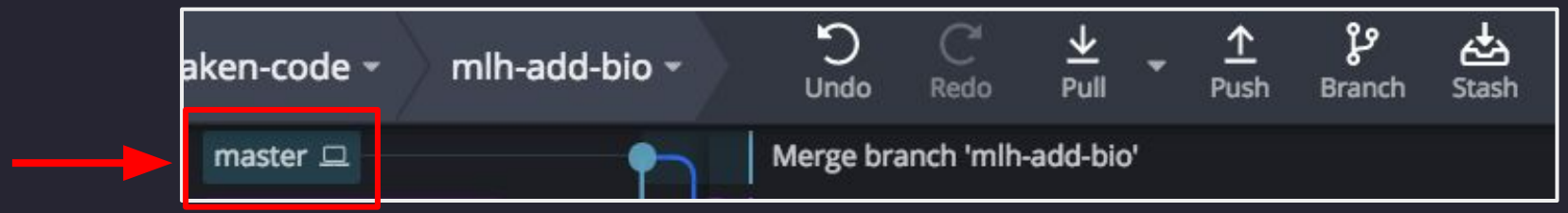


2. When the menu opens, select **Merge <your branch name> into master**.

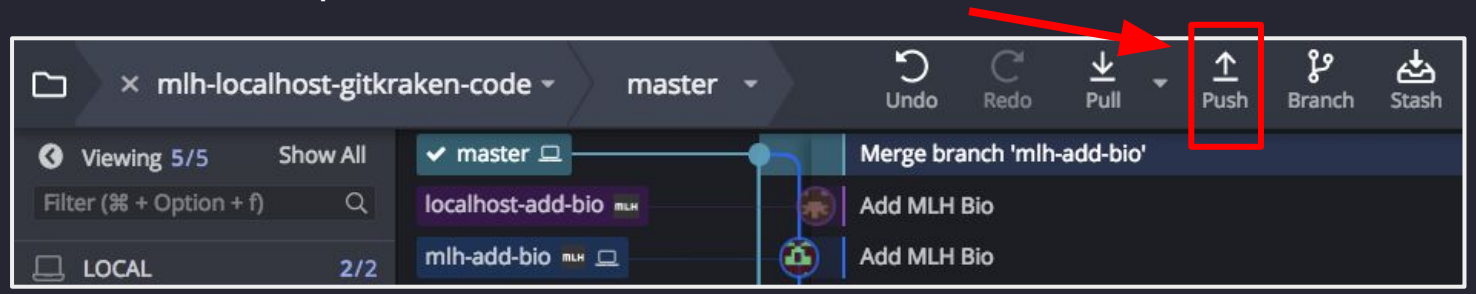


Merge Your Changes

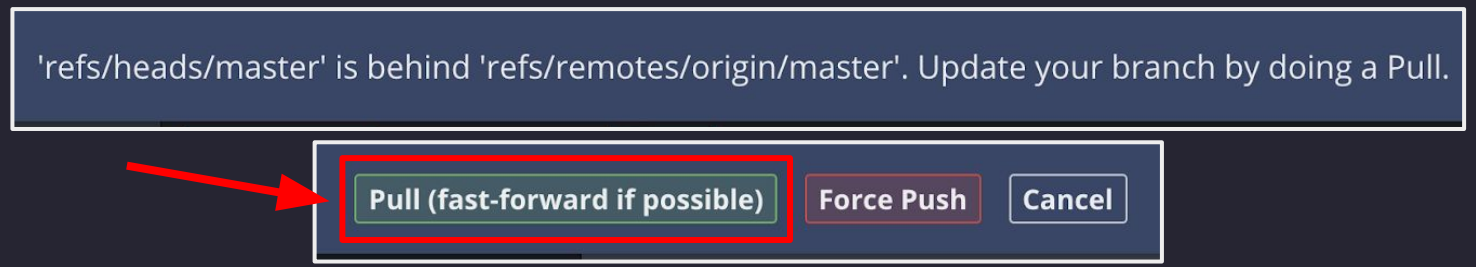
1. Double click the local master branch. You will be switched to that branch.



2. Click **Push** at the top.

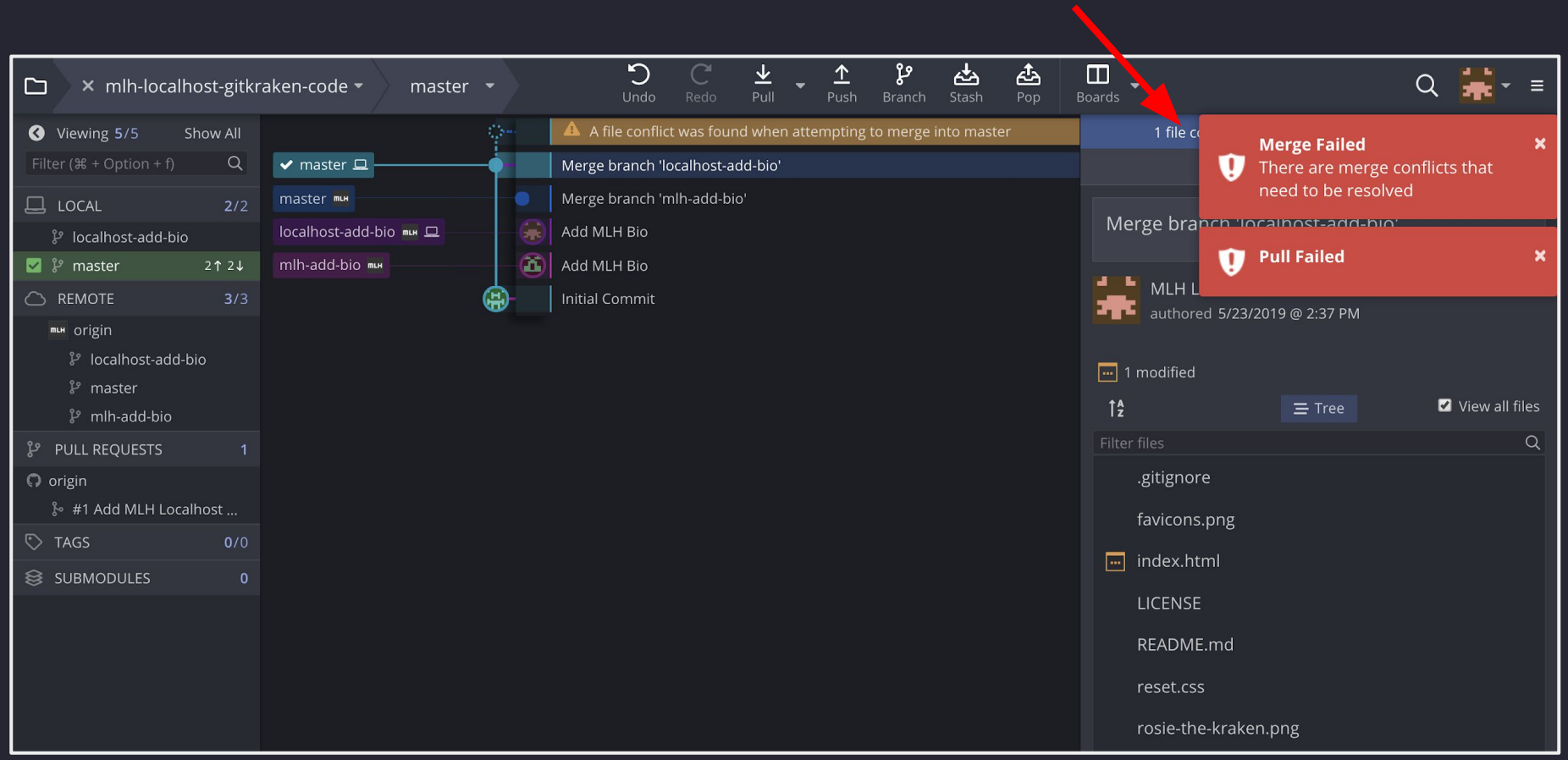


3. You'll get the warning below. This means that master on the remote has new code that yours doesn't. Select **Pull (fast-forward if possible)**.



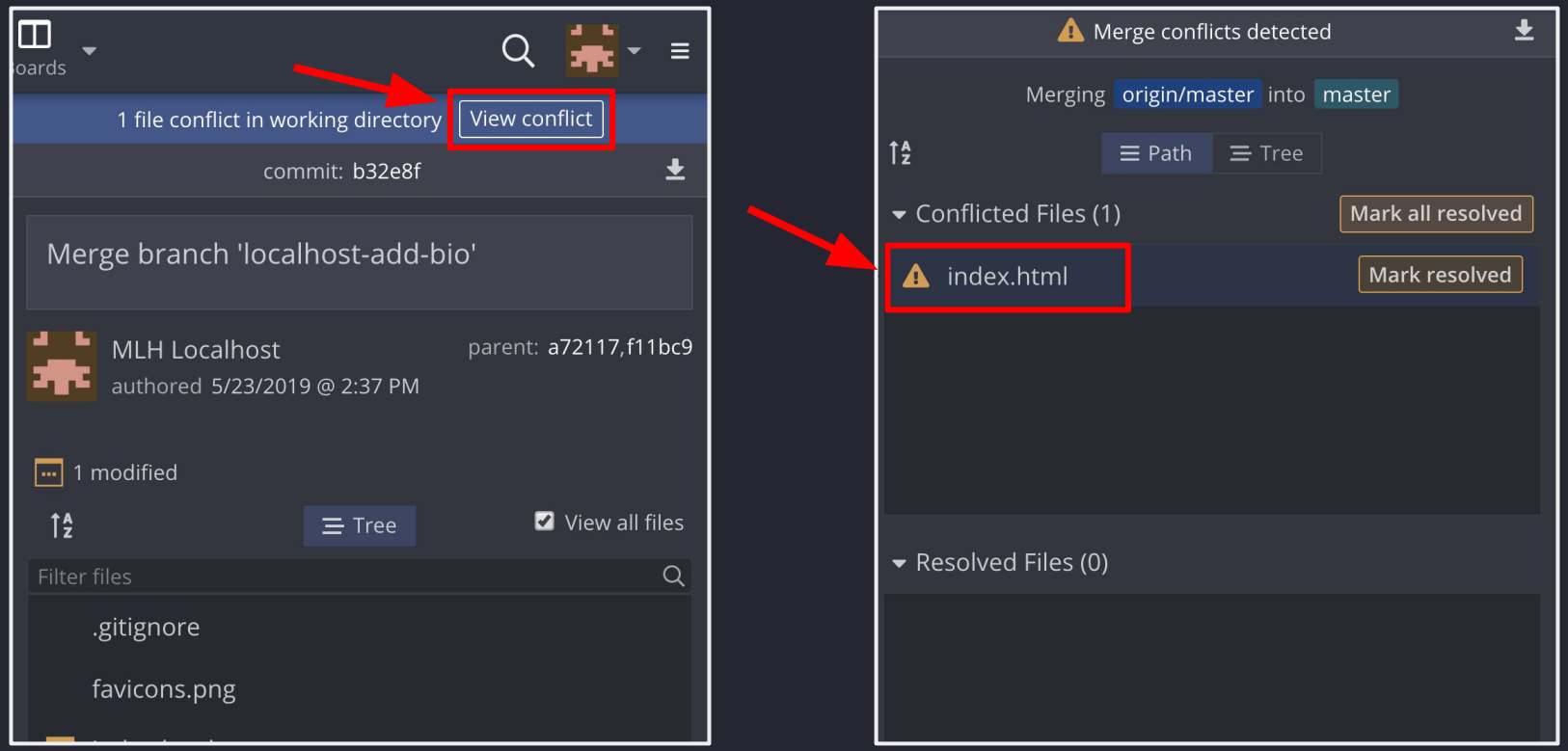
Merge Your Changes

You'll see these errors! This is because the code you have locally doesn't match the code on master, and Git couldn't determine which one was the correct one. X them out so we can fix it!



Resolve your Merge Conflict

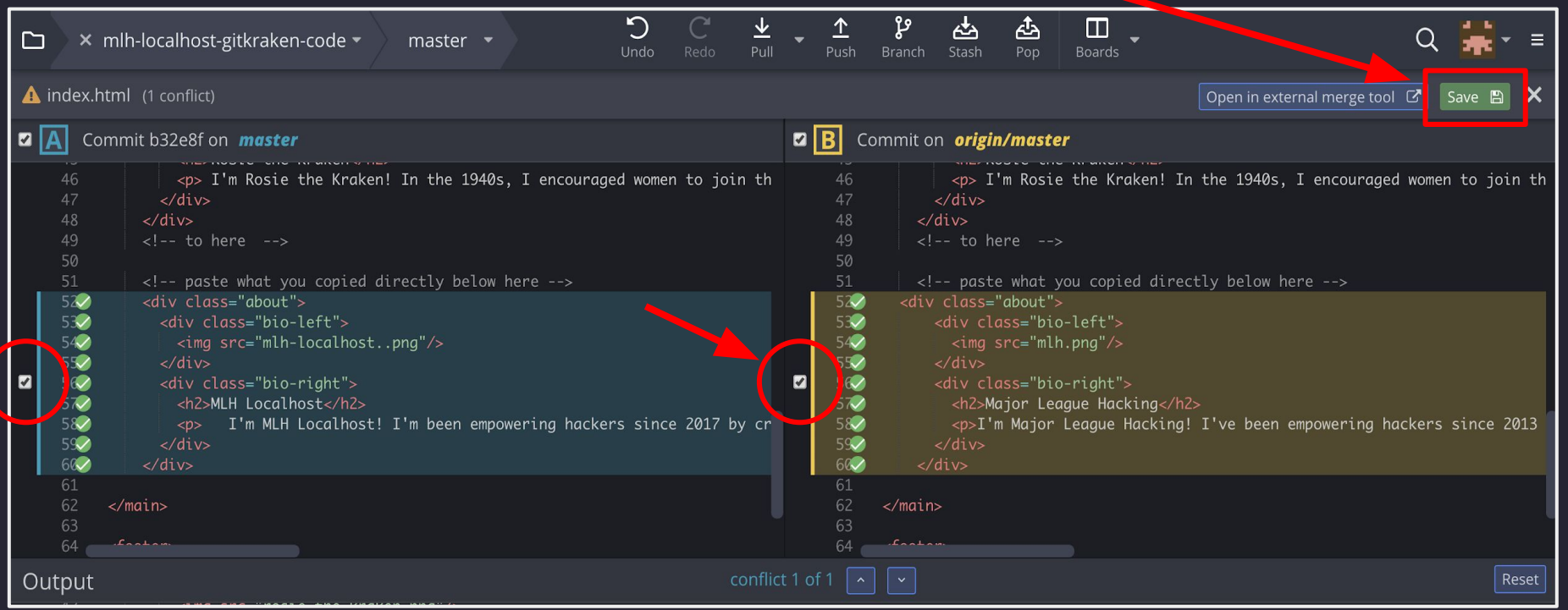
1. After you dismiss the error messages, you will see the option to **view conflict**. Select it.
2. The conflict is in index.html. This is because multiple people edited it at the same time. Click **index.html** to open the Merge Conflict Editor.



Resolve your Merge Conflict

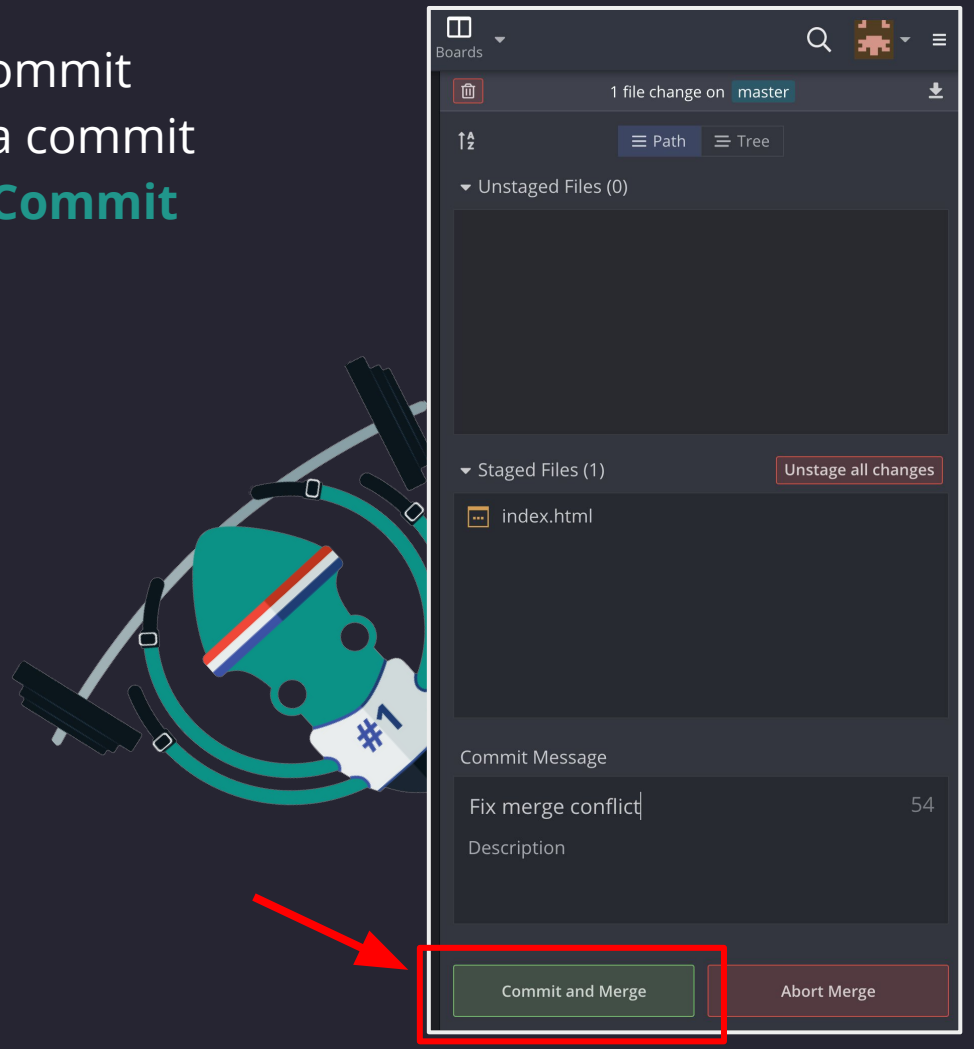
3. The code on the left is what you have locally. The code on the right is what's in the remote repository. We want both bios to appear on our website! Click the checkboxes on both sides.

4. Select **Save** in the upper right hand corner.



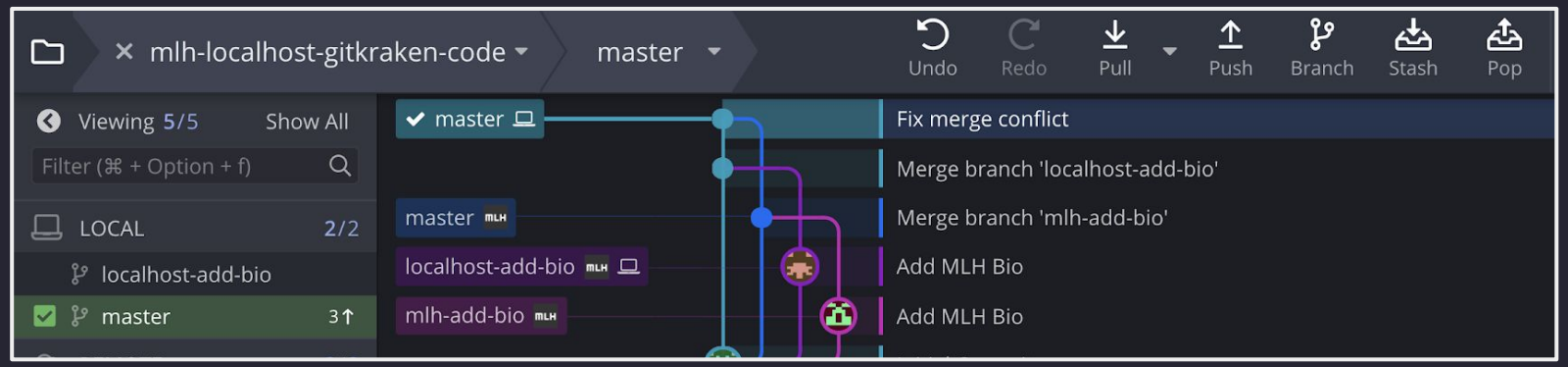
Resolve your Merge Conflict

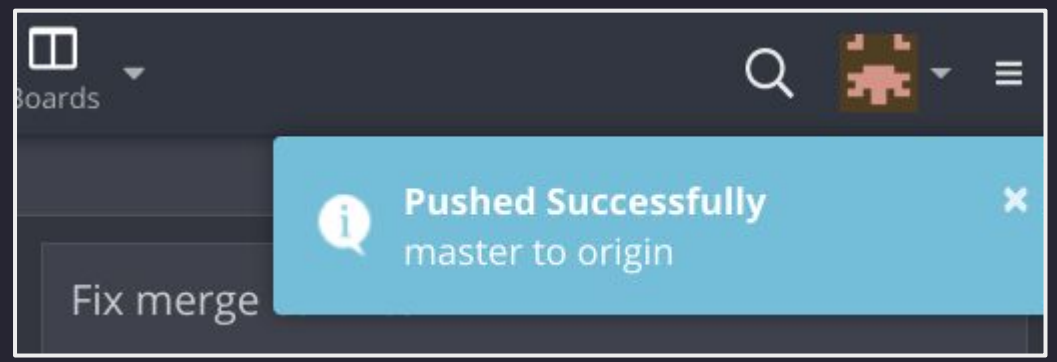
5. Now you'll have to commit these changes. Add a commit message and select **Commit and Merge**.



Resolve your Merge Conflict

6. Notice now that you have a local version of master again. Select Push.
7. If everything worked, you'll see the blue success message below. If not, this is likely because you and someone else tried to resolve your conflicts at the same time. Resolve them one at a time following the previous steps.

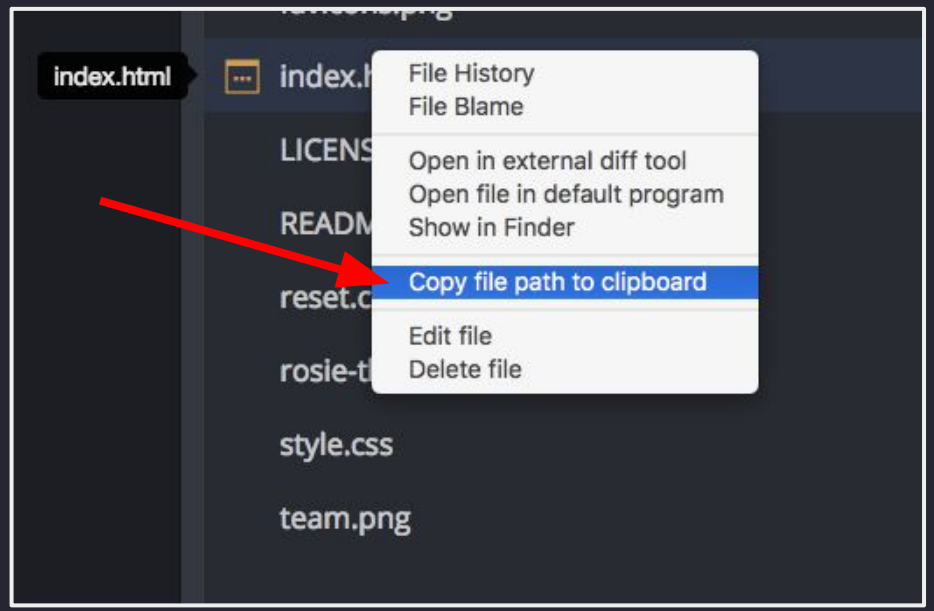




Check out your Team Page!

Want to see what your team page looks like now that it's done?

1. In GitKraken, make sure you have the latest version of master.
2. Right click index.html.
3. Select Copy file path to clipboard.
4. In your browser, paste the file path to view your project!



Awesome! Look what you built!

Table of Contents

1. Intro to Git and GitHub
2. Set up GitKraken
3. Collaborate!
4. Review and Quiz
5. Wrap Up and Next Steps

What You Learned Today

1. A best practice workflow for collaborating using Git
2. How to use GitKraken to make collaboration simple
3. How to use Glo Boards to manage tasks

Review what you learned!

We created a fun quiz to test your knowledge and see what you learned from this workshop.

http://mlhlocal.host/quiz

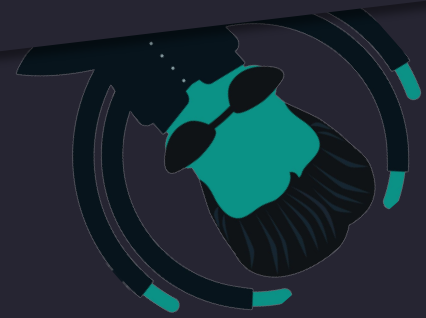
Table of Contents

1. Intro to Git and GitHub
2. Set up GitKraken
3. Collaborate!
4. Review and Quiz
5. Wrap Up and Next Steps

Keep Learning

Extra Practice Problem 1:

Deploy your Team Website using GitHub Pages



Learning shouldn't stop when the workshop ends...

Check your email for access to:

- These workshop slides
- Practice problems to keep learning
- Deeper dives into key topics
- Instructions to join the community
- More opportunities from MLH!

THANK YOU!

www.gitkraken.com

Get Crackin' with Git Using GitKraken

MLH localhost