



Memorial Sloan Kettering
Cancer Center™

Git/GitHub Training Part 2

Working Collaboratively With Git & Github



March 7, 2022

Karissa Whiting, Epi-Bio
Jessica Lavery, Epi-Bio

Daniel D. Sjoberg, Epi-Bio

Caroline Kostrzewska, Epi-Bio

Shannon Pileggi, PCCTC, Epi-Bio

Karolyn Ismay, Strategy & Innovation

Agenda

Review Git & Github Basics

- Git
- Github
- Repositories

New Concepts

- Branching
- Forking
- Pull Requests

Example Collaborative Workflow

- Concepts in Practice

5 Min Break + Breakout Rooms

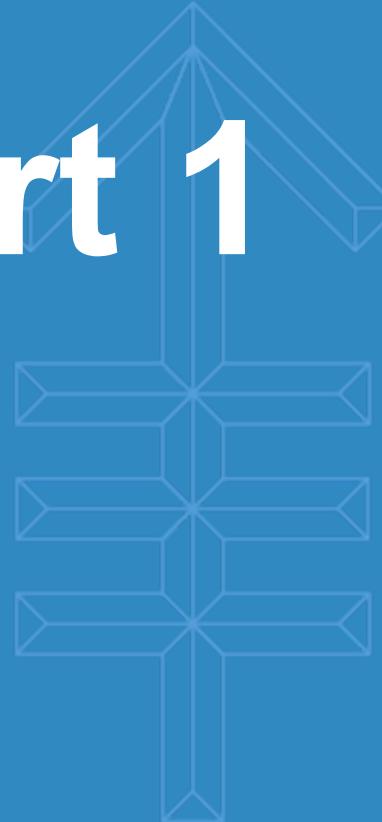
Part 2 Training Goals

- Add new terms to your git vocabulary to facilitate collaborative work:
 - Branching
 - Forking
 - Pull Requests
- **Understand in what contexts these concepts are useful**



Memorial Sloan Kettering
Cancer Center™

Review Part 1



What is Git?



- **Git** is a free, open-source software
- Allows you to set up **version control** systems attached to your projects that can be used to **track changes** across project files and lines of code

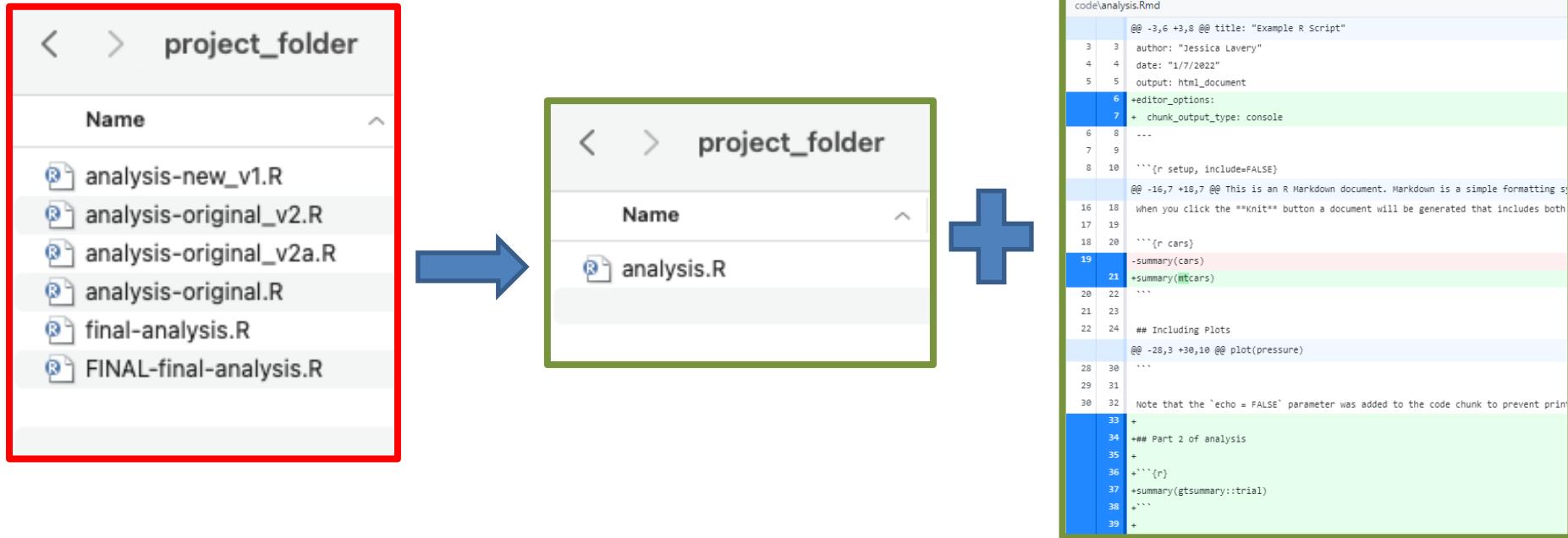
Why Use Git?

- More formal **version control system** to better understand when and why you made specific changes to a project
- **Reproducibility-** revert to old version of the code/project as needed
- **Collaboration** & code review

In every project you have at least one other collaborator;
future-you. You don't want future-you to curse past-you.

- Hadley Wickham

Why Use Git?



What is GitHub?

- **Git (“local”)** - software installed on your local computer to track changes
- **GitHub (“remote”)**- a home for your git-based projects on the internet
 - Similar to Google Drive or Dropbox/Box
 - Work on your project locally on your computer but you Sync them to Drive/Dropbox/Box on the cloud
 - Files can be accessed via those websites and shared with others

What's a Repository?

- A **repository** is essentially a project folder that you have enabled git to track.
- Therefore, it consists of both the project files themselves, as well as records of the files' version histories.

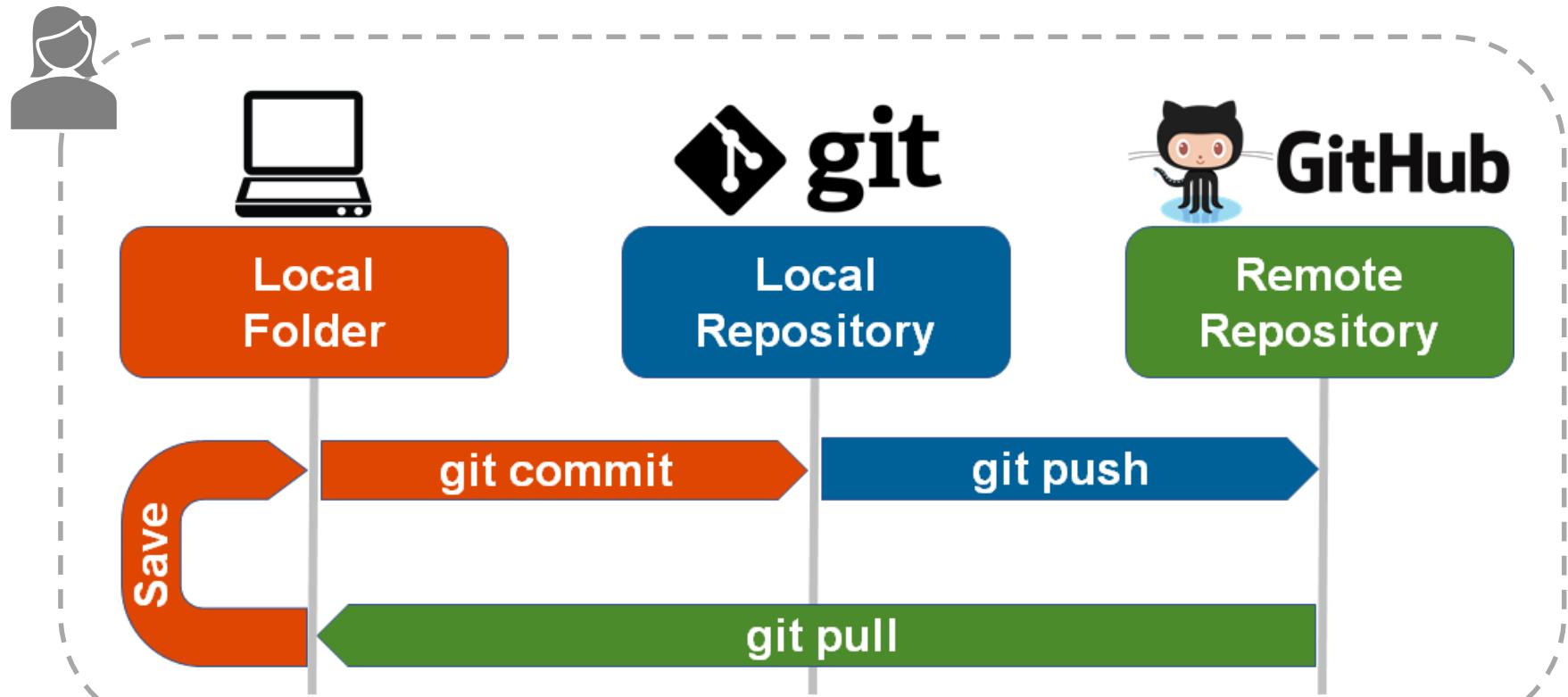


Repo Locations

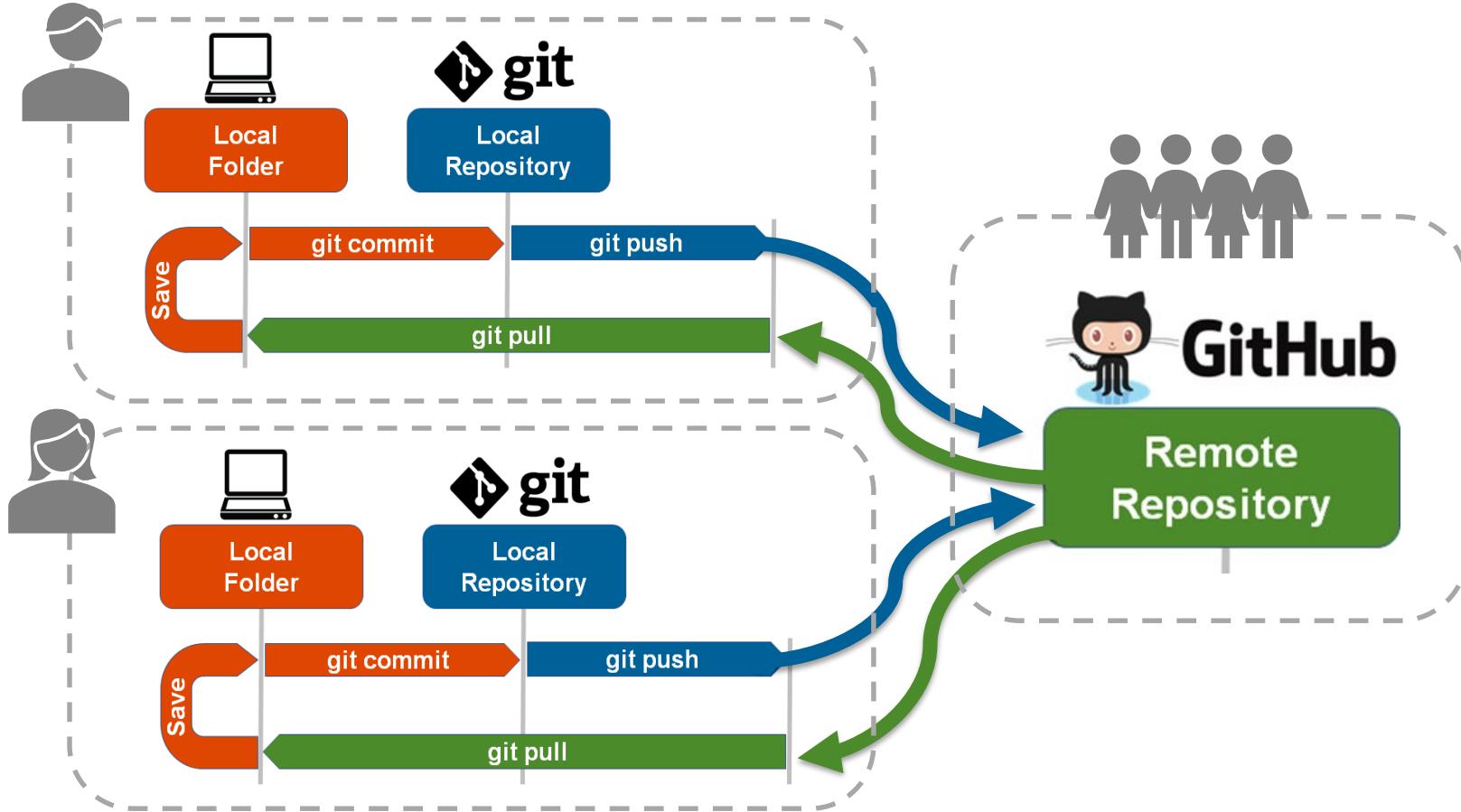
- **Local repository** - Version of your project that lives on your local computer. *This is the version you make edits to.*
- **Remote repository** - Version that lives remotely on GitHub.
 - GitHub offers a suite of features for managing, organizing and searching your project code.



Review: Git & GitHub Workflow



Expanded Git & GitHub Workflow





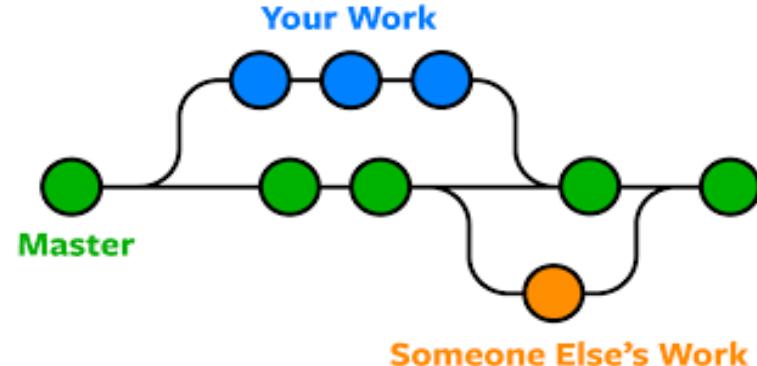
Memorial Sloan Kettering
Cancer Center™

Branching



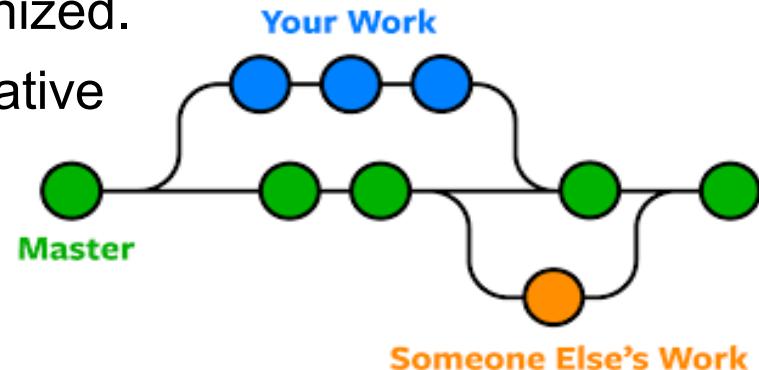
What is a Branch?

- **Branches** are a copy of your code that you keep separate from the main project code
- On a branch, you can experiment with changes safely as these changes are made **in isolation of the main project code**
- This allows you to develop features, fix bugs, and safely experiment with new ideas in a contained area
- It **remains linked** with the main project



Branches are useful for collaborations

- **Problem:** If everyone makes changes directly to the main code of project at the same time, this may create conflicts or and we may break each other's code.
- **Solution:** Each member works on their own branch and each branch is QA-ed before merging into the main project
- **Bonus:** Branches help you keep organized.
Give your branch a unique and informative name.



Example of Branching In Action

A screenshot of a GitHub repository page for `ddsjoberg/gtsummary`. The repository is public, has 15 issues, and 4 pull requests. The main branch is selected. There are 14 other branches listed, including `add_glance_msg_update_for_mice`, `add_n_after_overall`, `as_hux_xlsx`, `cont2_footnotes`, `deprecate_options`, and `get_theme`. There are also 24 tags.

Switch branches/tags

Find or create a branch...

Branches Tags

- ✓ main default
- add_glance_msg_update_for_mice
- add_n_after_overall
- as_hux_xlsx
- cont2_footnotes
- deprecate_options
- get_theme

A screenshot of the `ddsjoberg/gtsummary` GitHub repository page. The repository is public, has 15 issues, 4 pull requests, and 4 active branches. The `main` branch is the default branch, last updated yesterday by `ddsjoberg`.

Overview Yours Active Stale All branches

Default branch

<code>main</code>	Updated yesterday by <code>ddsjoberg</code>	✓	(Default)
-------------------	---	---	-----------

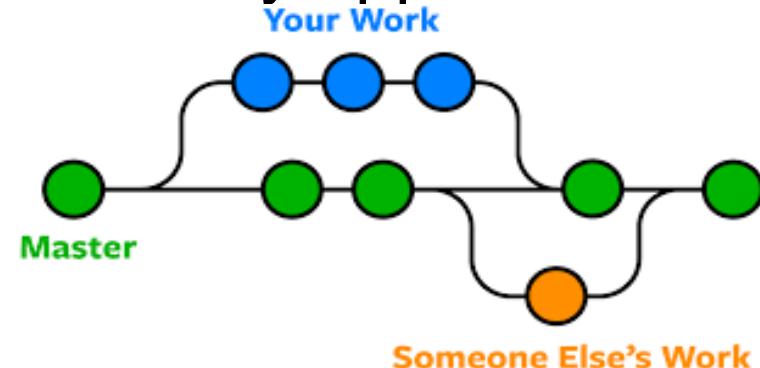
Active branches

Branch	Last Updated	Pull Request	Actions
<code>deprecate_options</code>	Updated 17 minutes ago by <code>Curry</code>	#1180 Open	
<code>gh-pages</code>	Updated yesterday by <code>ddsjoberg</code>		
<code>modify_err_msg</code>	Updated yesterday by <code>ddsjoberg</code>	#1177 Draft	
<code>kable_extra_extra_extra</code>	Updated yesterday by <code>ddsjoberg</code>	#1155 Open	
<code>as_hux_xlsx</code>	Updated yesterday by <code>ddsjoberg</code>	#1179 Open	

[View more active branches >](#)

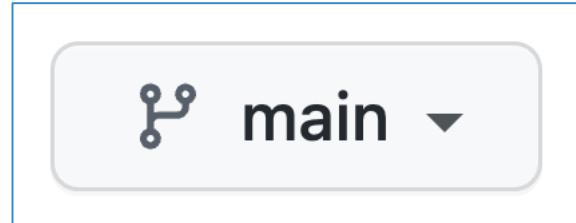
Branches can be useful for individual projects too

- Specific project versions for journal submission/review
- Try a complex new analysis - easy to abandon
- Maintain stable versions of dashboards/automated reports/ shiny apps



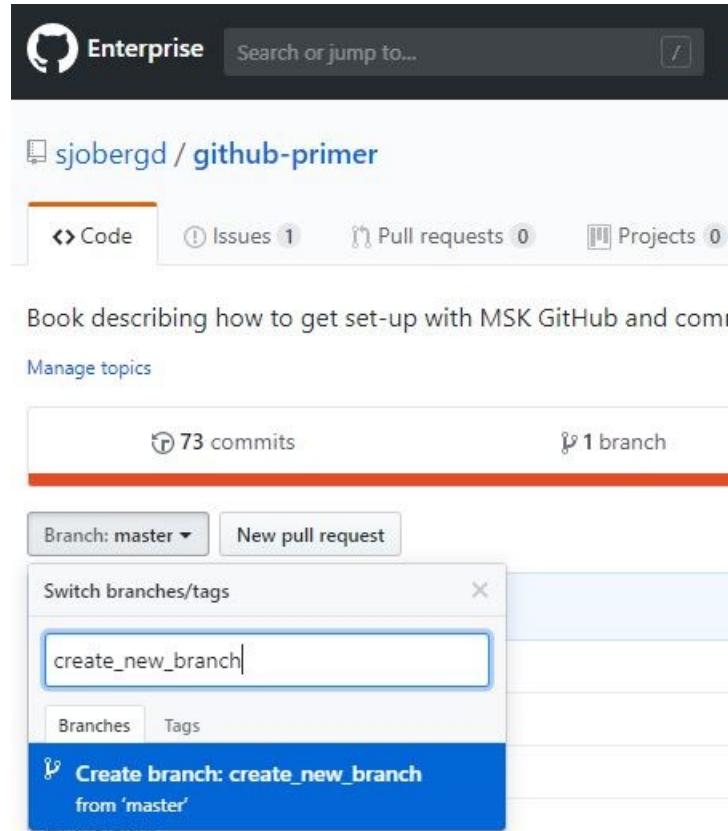
Main (Default) Branch

- You've actually been working with branches already!
- So far, we've been committing and pushing changes to the default branch called **Main** (sometimes also called **Master** branch)
- This is created by default when you create a repository



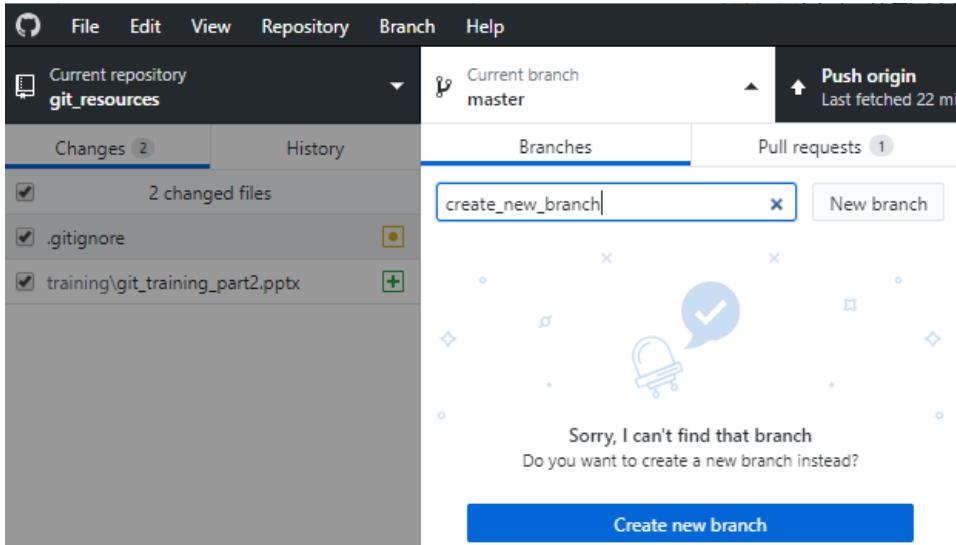
Create New Branch (in GitHub)

1. Go to <https://github.mskcc.org>
2. Navigate to repository
3. Click “Branch: master” button
4. Type name of new branch
5. Click “Create Branch”

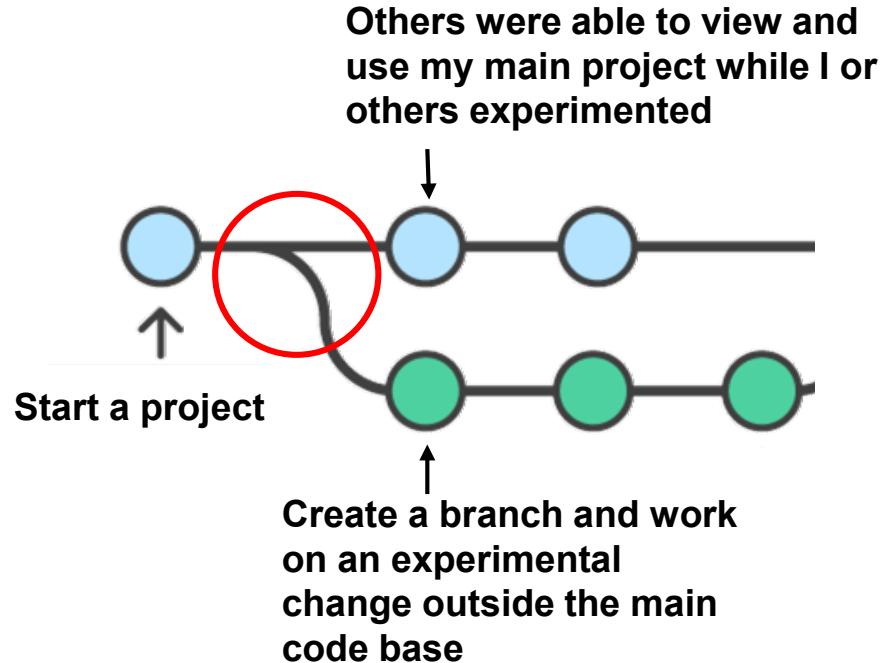


Create New Branch (in GitHub Desktop)

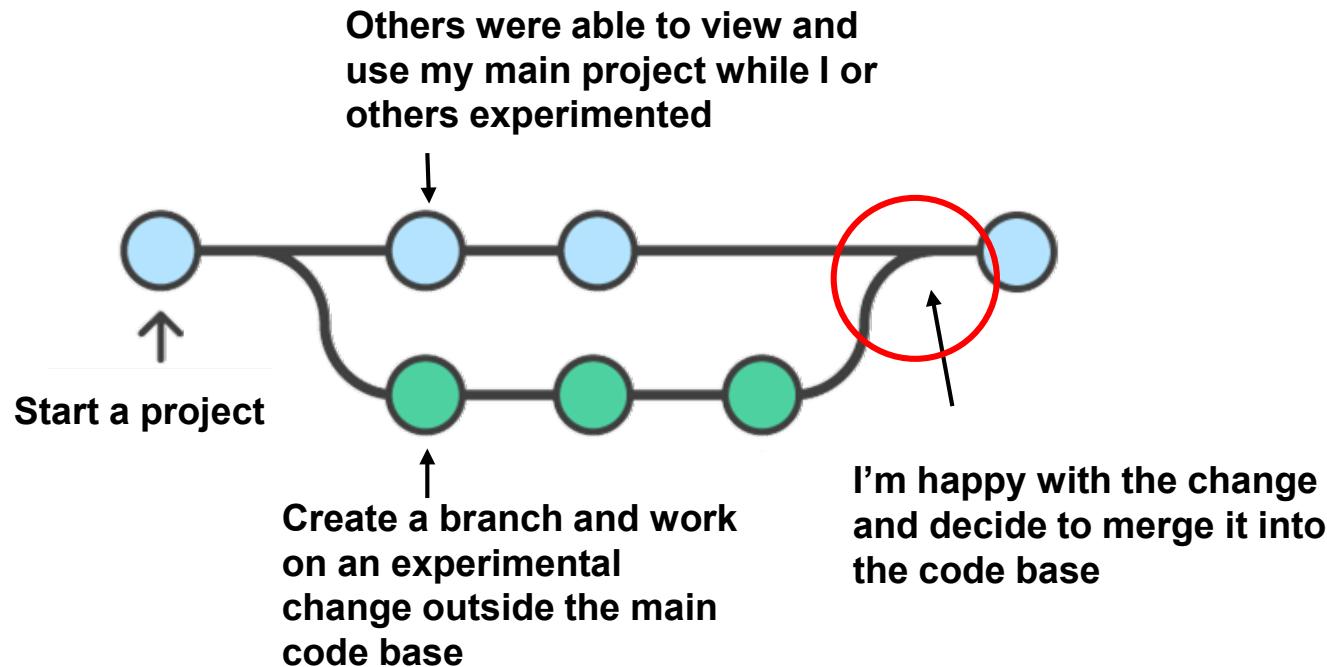
1. Open GitHub Desktop
2. Open repository
3. On the branch selection tab, type name of new branch
4. Click “Create New Branch”



Create New Branch



Merge Changes Into Main Project





Memorial Sloan Kettering
Cancer Center™

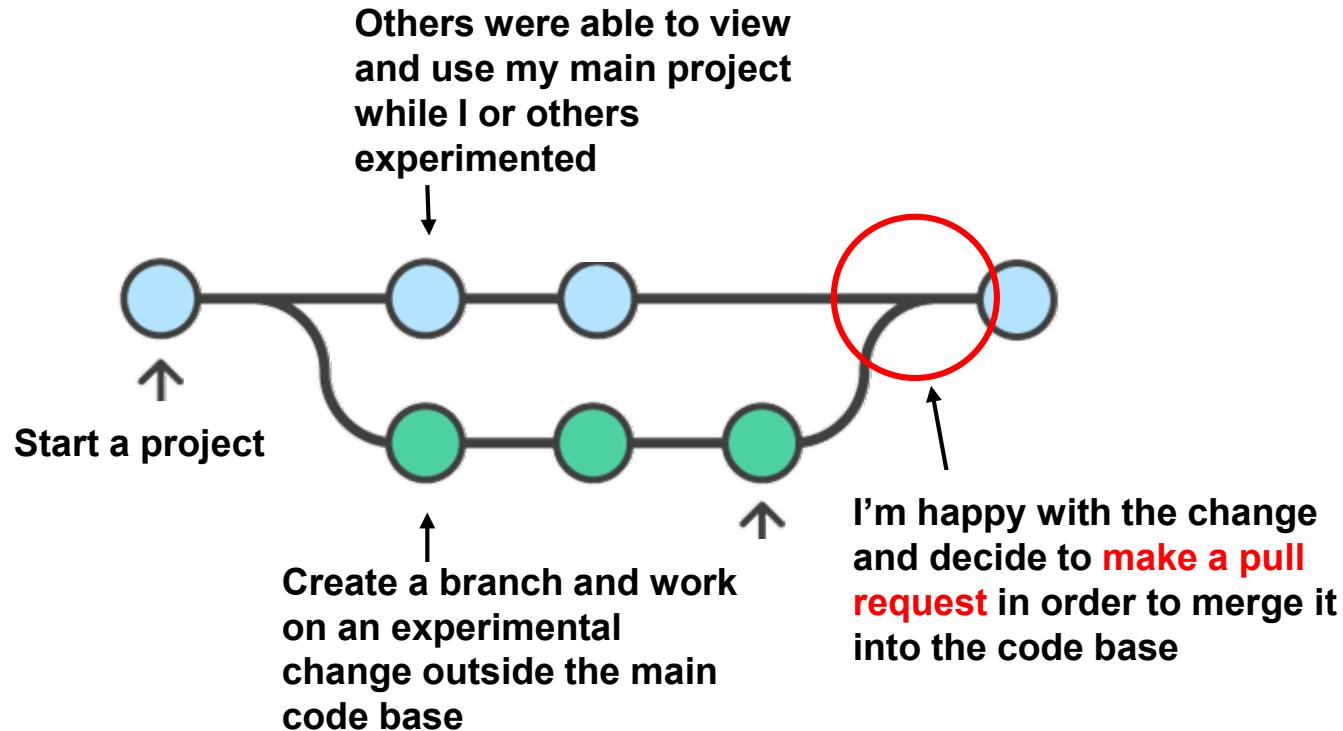
Pull Requests



What is a Pull Request?

- **Pull Request** = A Code Review + A Merge Into Main Code
- Pull Requests are how you get changes from a branch into the main project (“main” branch).

Pull Request



Why “Pull Request” Not “Push Request”

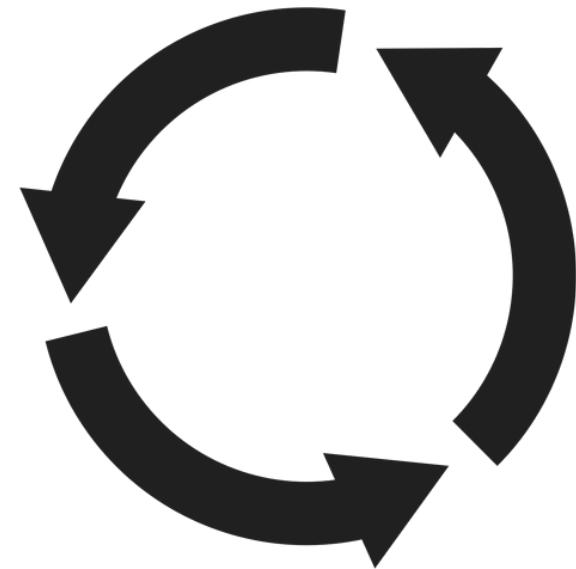
- If you are trying to “push” your new changes into the main repository why is it called a “pull request”?!

Why “Pull Request” Not “Push Request”

- If you are trying to “push” your new changes into the main repository why is it called a “pull request”?!
- Because it’s more polite? 

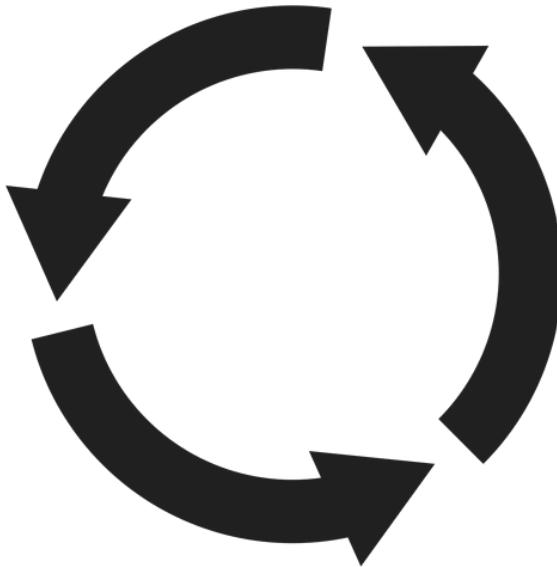
Workflow Steps

1. Create a new GitHub repo or decide to work on an existing one
2. **Clone** repo with GitHub Desktop
3. Work on the project, make changes locally, **save**, and **commit**
4. **Push** local changes to GitHub remote repo
5. Repeat steps 3-4 as you update code



Workflow Steps

1. Create a new GitHub repo or decide to work on an existing one
2. **Clone** repo with GitHub Desktop
3. **Create a branch to house the new changes**
4. Work on the project, make changes locally, save, and commit **to that branch**
5. Push local changes to GitHub remote repo **on that new branch**
6. Repeat steps 3-4 as you update code
7. **Create a pull request to integrate branch into the main project**





Memorial Sloan Kettering
Cancer Center™

Forking



Forking

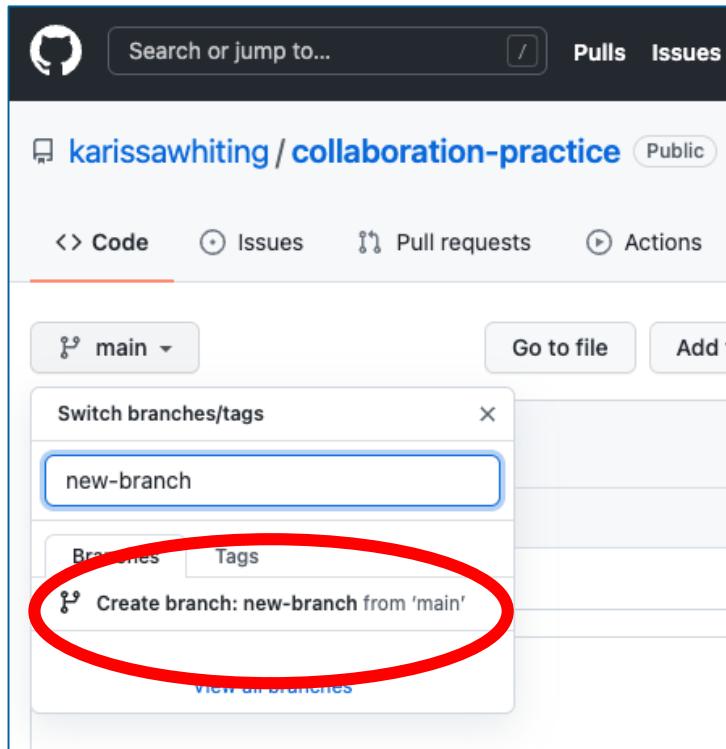
- Forking creates a copy of someone else's Github repository on your Github account
- This gives you ownership of a separate and completely independent copy to edit in whatever way you chose.

Why Create a Fork?

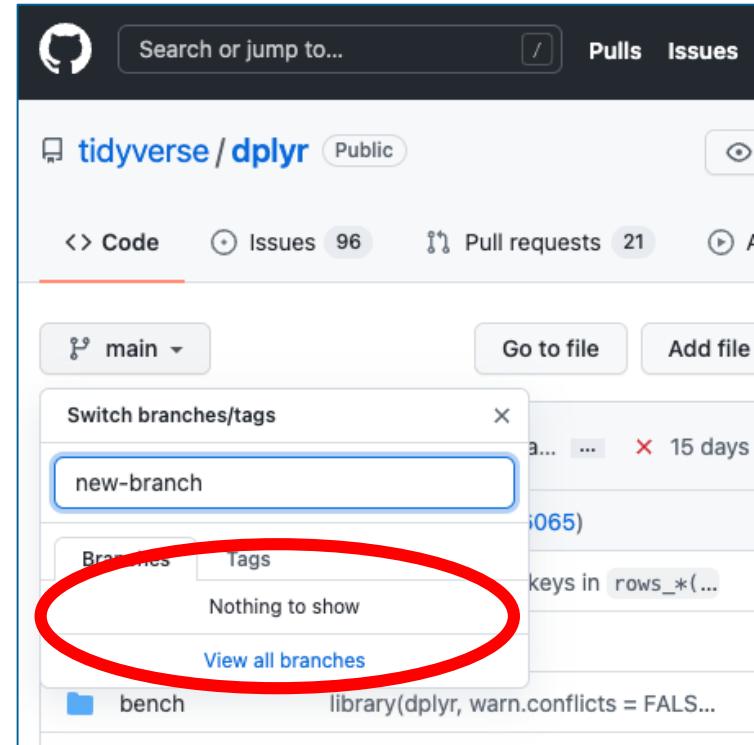
- **Permissions** - Collaborators may not let you branch directly from their main repository. You may need to fork first, then branch.
- **Protects the main code base** in collaborative work. (it's another degree of separation between you and the main repository code)
- Forking and branching often used together in larger collaborative projects



Why Create a Fork?

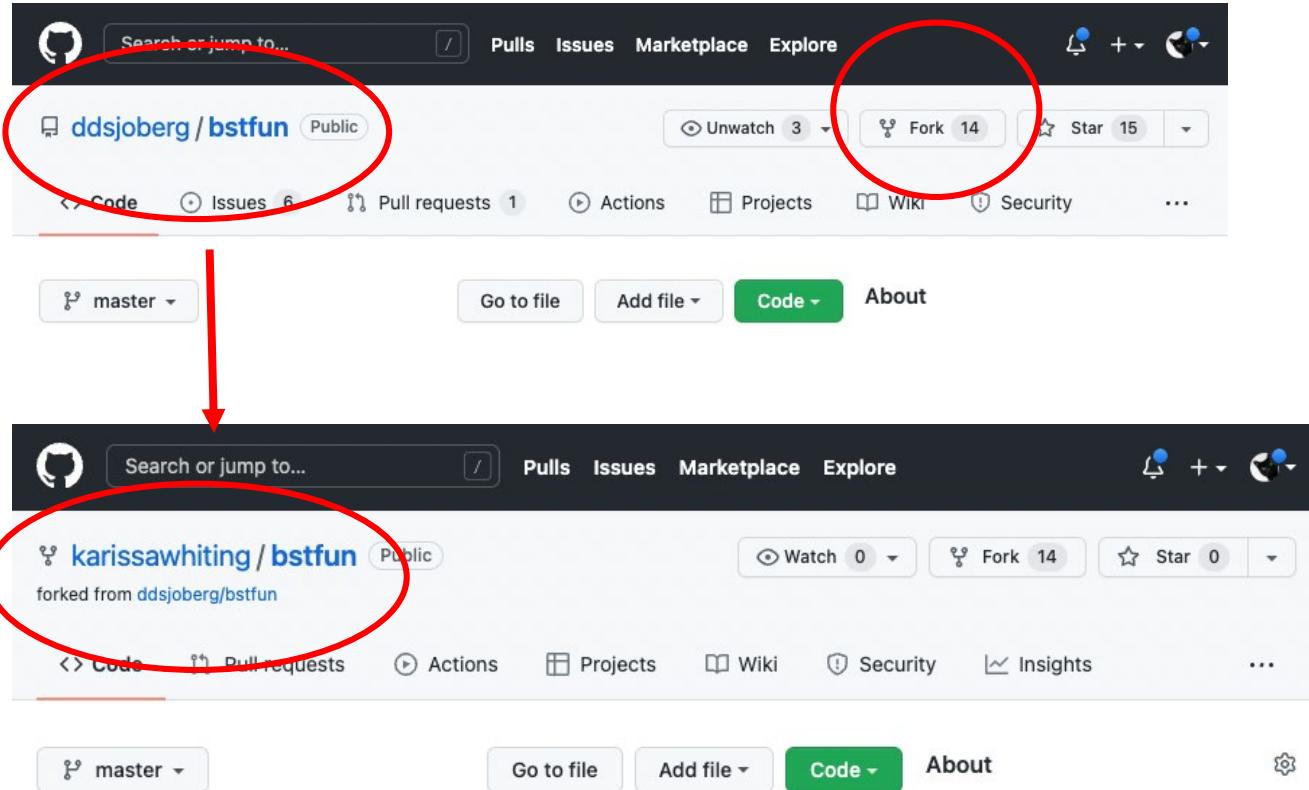


A screenshot of a GitHub fork creation interface. At the top, there's a search bar and navigation links for 'Pulls' and 'Issues'. Below that, the repository name 'karissawhitling / collaboration-practice' is shown as 'Public'. There are tabs for 'Code', 'Issues', 'Pull requests', and 'Actions', with 'Code' being the active tab. A dropdown menu shows 'main' selected. Below it is a 'Switch branches/tags' input field containing 'new-branch', which is highlighted with a blue border. A red oval highlights the button below the input field that says 'Create branch: new-branch from \'main\''. At the bottom of the dropdown, there's a link 'View all branches'.



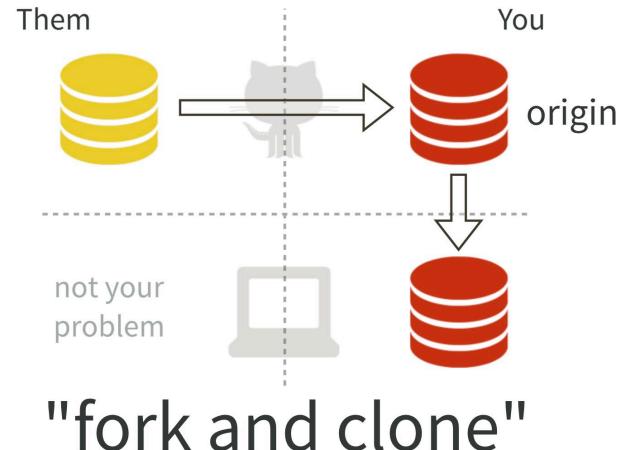
A screenshot of a GitHub repository interface for 'tidyverse / dplyr', which is public. It shows 96 issues and 21 pull requests. The 'Code' tab is active. A dropdown menu shows 'main' selected. Below it is a 'Switch branches/tags' input field containing 'new-branch', which is highlighted with a blue border. A red oval highlights the message 'Nothing to show' in the dropdown, indicating no branches have been created yet. At the bottom of the dropdown, there's a link 'View all branches'.

Create a Fork



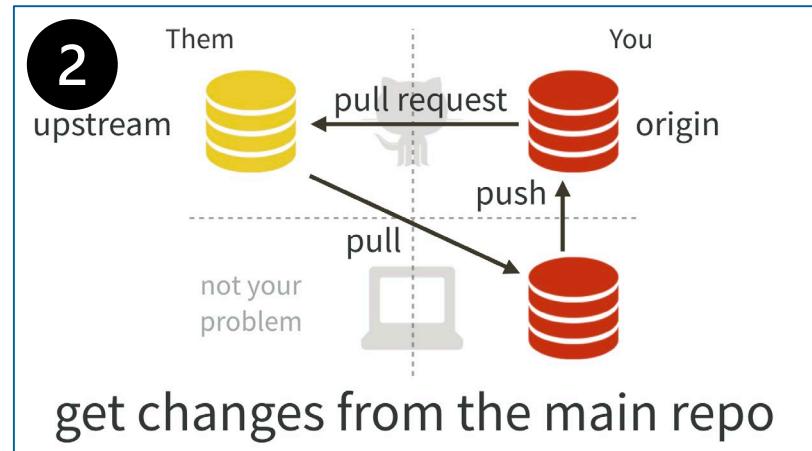
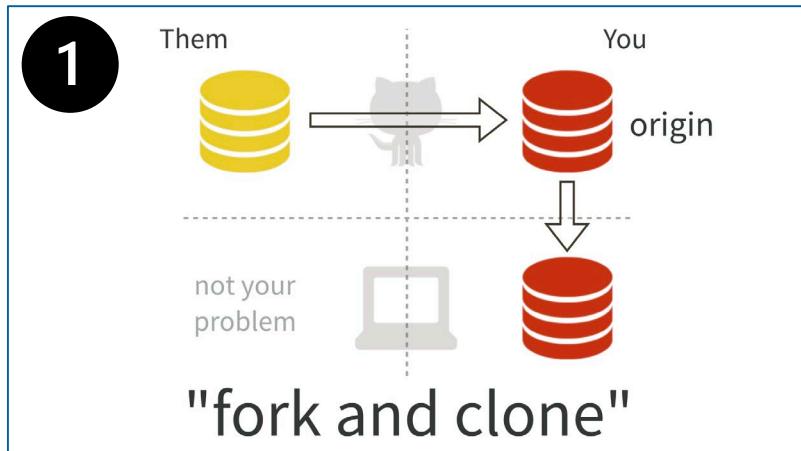
Fork vs. Clone

- Unlike **cloning** (which brings a copy of a remote repo down to your local computer), **forking is entirely on Github** (remote).



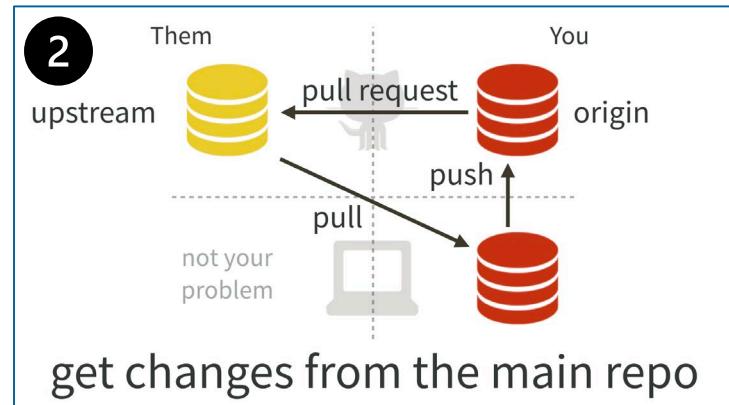
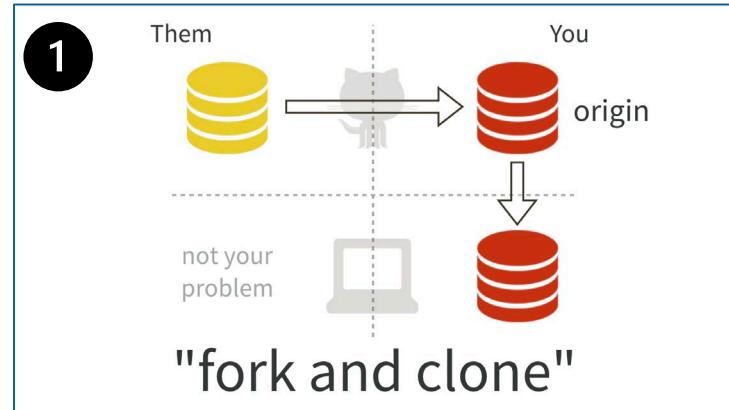
Merging Fork With Main Repo

- **Pull Requests** – Like with branches, you can merge your changes from your fork into the main project with a pull request



Fork + Clone + Branch + Pull Request

- 1) **Fork** the project onto your Github account
- 2) **Clone** your fork onto your local computer
- 3) Make a **branch** to house unique set of changes
- 4) **Commit** and **push** changes to your remote fork on Github
- 5) Eventually make **a pull request** asking repository owner merge your changes into main project.



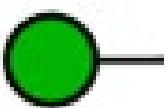


Memorial Sloan Kettering
Cancer Center™

A Tale of Two Contributors

(Inspired by true events)

1



Dan created the
`{bstfun}` R package

(Green = Master/Main
Project Branch)

2



Meier wanted to contribute cumulative incidence risk table functions

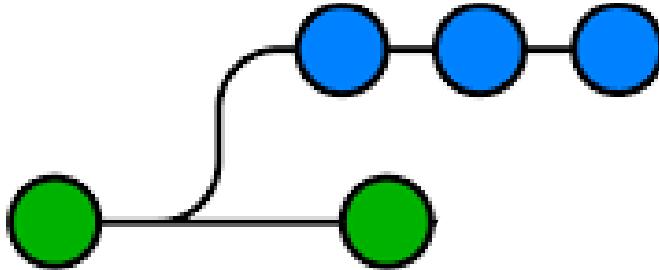
- 1) Forked the repo
- 2) Created a branch called (cuminc-update)
- 3) Added a feature and commit code to branch
- 4) Created a pull request for Dan to review to incorporate changes into the package

1



Dan created the
`{bstfun}` R package

(Green = Master/Main
Project Branch)



2



Meier wanted to contribute cumulative incidence risk table functions

- 1) Forked the repo
- 2) Created a branch called (cuminc-update)
- 3) Added a feature and commit code to branch
- 4) Created a pull request for Dan to review to incorporate changes into the package

1



Dan created the
`{bstfun}` R package

(Green = Master/Main
Project Branch)

3



While Meier worked, Jasme decided to work on project templates in the package. She created a fork and a designated branch for her change

2



Meier wanted to contribute cumulative incidence risk table functions

- 1) Forked the repo
- 2) Created a branch called (cuminc-update)
- 3) Added a feature and commit code to branch
- 4) Created a pull request for Dan to review to incorporate changes into the package

1



Dan created the {bstfun} R package

(Green = Master/Main Project Branch)

3



While Meier worked, Jasme decided to work on project templates in the package. She created a fork and a designated branch for her change

4



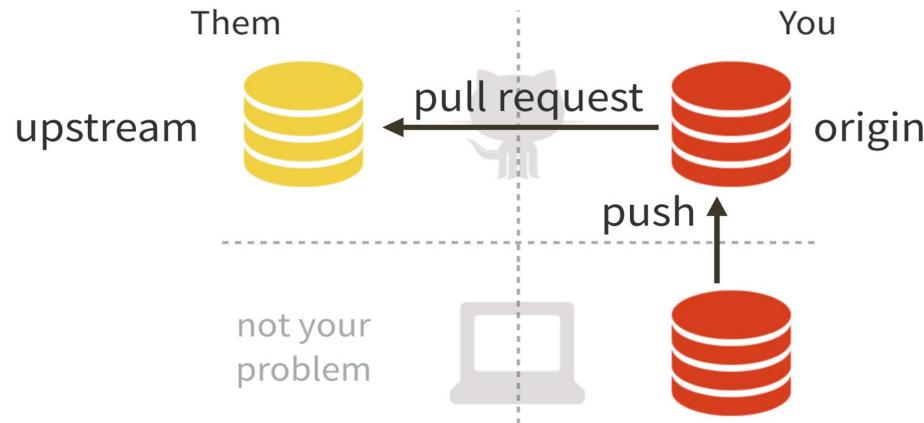
While Jasme worked, Dan reviewed and accepted Meier's pull request. Her contribution is now part of the main branch of the project.



Jasme will likely want to pull these new updates into her branch so she is working off the latest copy

Pulling From “Upstream”

- Jasme will likely want to keep her fork/branch up to date with the main project
- She will have to **pull** from the updated **upstream repository**



Code Conflicts Between Contributors

- What happens to Jasme's code when she pulls in Meier's changes?
- If any of her code conflicts with the changes Meier made, github will ask her to review and decide which version of the code she want to keep

Thank You!



Questions?



Memorial Sloan Kettering
Cancer Center™

5 Min Break + Breakout Rooms



Memorial Sloan Kettering
Cancer Center™

Git/GitHub

Training Part 2

Breakout Rooms



March 21, 2022

Karissa Whiting, Epi-Bio

Jessica Lavery, Epi-Bio

Daniel D. Sjoberg, Epi-Bio

Caroline Kostrzewska, Epi-Bio

Shannon Pileggi, PCCTC

Karolyn Ismay, Strategy & Innovation

Goals

- **Fork A Remote Repo** – Make a copy on your Github Enterprise account of the ‘**group-leader/gh2-collaboration-practice**’ Repository
- **Bring It Local** by cloning your fork into a local repository/folder on your computer
- **Make A Branch** on your local repository using Github Desktop to organize your future changes
- **Make Changes to Project Files** – You will create a new .txt file and name it with your initials, e.g. kaw.txt
- **Commit the Changes Locally** to the branch on your local repository using Github Desktop
- **Publish the Branch and Push the Committed Changes** to your remote fork on Github Enterprise
- **Create a Pull Request** to merge your new file into the main branch of the original repo on Github (**group-leader/ gh2-collaboration-practice**)

Fork Your Group Leader's Remote Repository

REMOTE

LOCAL



GitHub

Someone Else's
Remote Repository

REMOTE

LOCAL



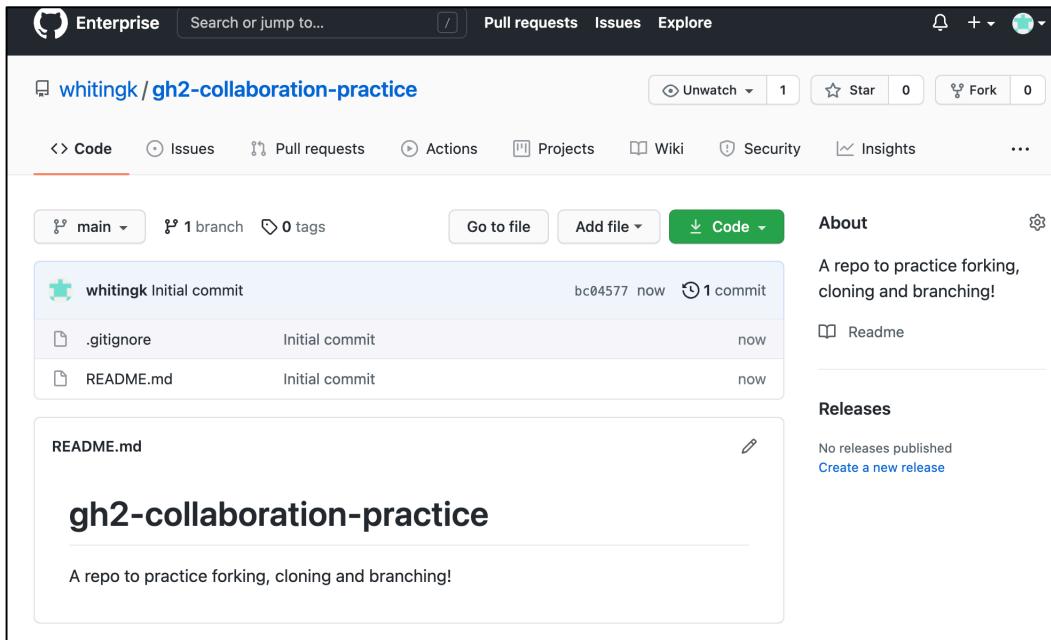
GitHub

Someone Else's
Remote Repository

fork

Your
Remote Repository

Find The Remote Repo To Fork



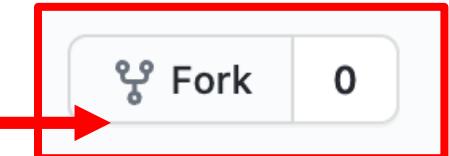
1. Sign into Github Enterprise <https://github.mskcc.org/>

2. Find Your Breakout Room Leader's Repository
<https://github.mskcc.org/<your-leader-username>/gh2-collaboration-practice>

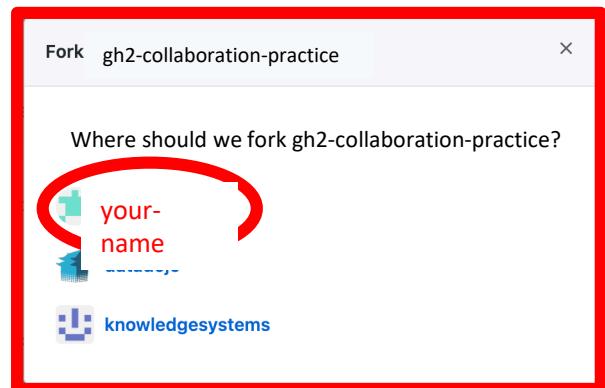
Fork The Repo

The screenshot shows a GitHub repository page. At the top, there's a navigation bar with links for 'Enterprise', 'Search or jump to...', 'Pull requests', 'Issues', and 'Explore'. Below the navigation is the repository name 'whitingk/gh2-collaboration-practice'. The main area displays a commit history with one commit from 'whitingk' labeled 'Initial commit'. There are also files listed: '.gitignore' and 'README.md', both with 'Initial commit' status. On the right side, there's an 'About' section with a brief description: 'A repo to practice forking, cloning and branching!', a 'Readme' link, and a 'Releases' section indicating 'No releases published' and a link to 'Create a new release'. The top right of the page has a 'Fork' button with a count of '0'.

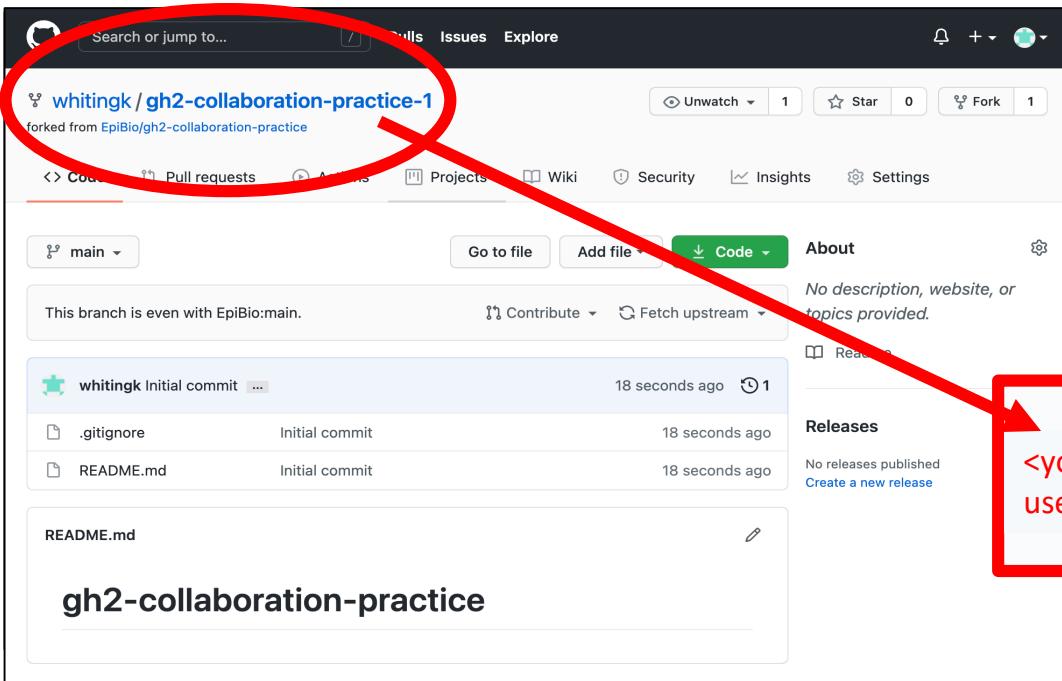
3. Click



You may get a pop-up asking you where to fork. If so, click your username



Fork The Repo



Now you have your very own version of the repo in your github account!

/ gh2-collaboration-practice-

whitingk / gh2-collaboration-practice

Clone Your Fork Onto Your Local Machine

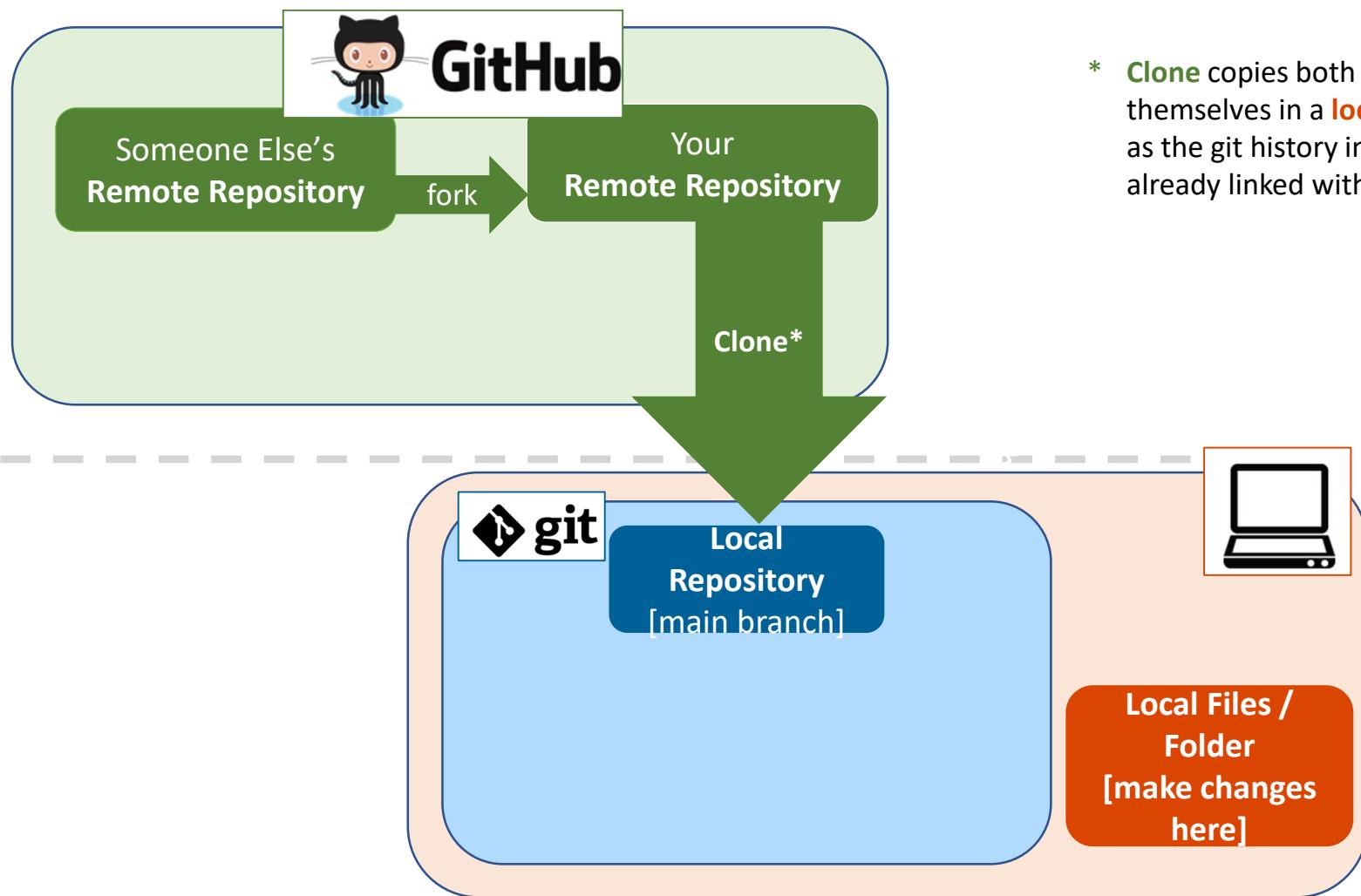


GitHub

Someone Else's
Remote Repository

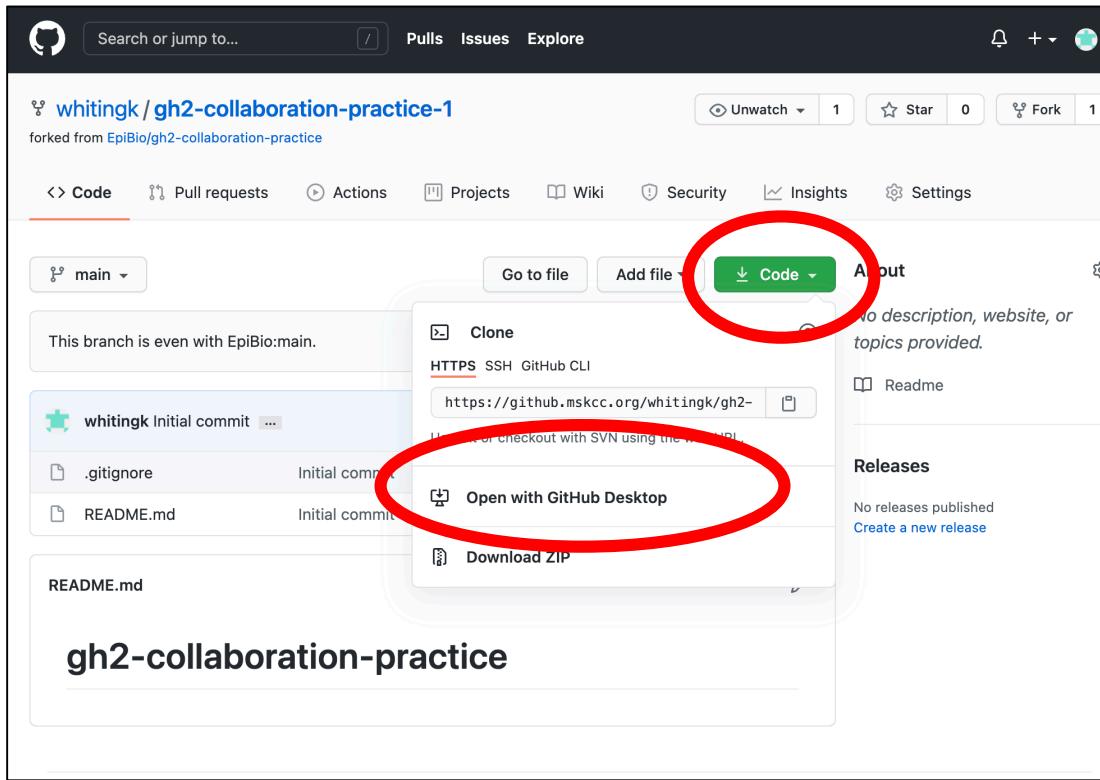
fork

Your
Remote Repository



- * **Clone** copies both the project files themselves in a **local folder** as well as the git history in a **local git repo** already linked with the folder

Clone Your Forked Repo

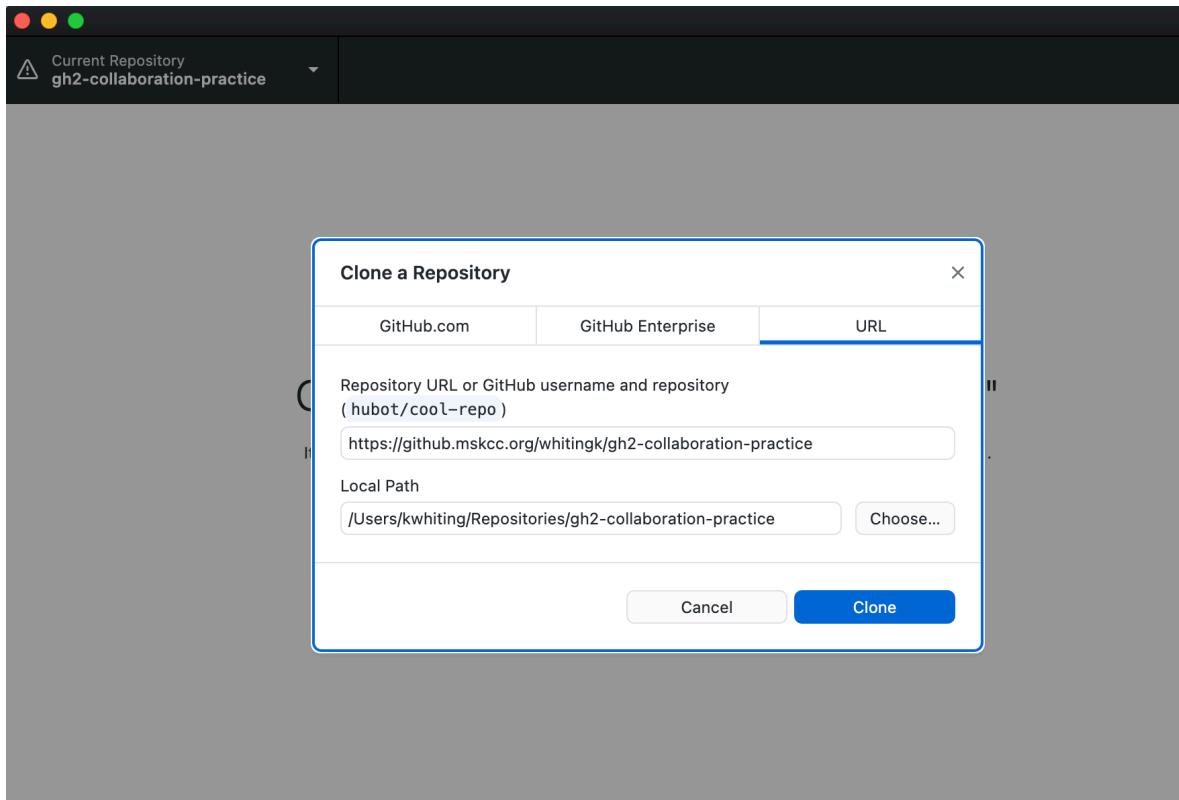


4. Click

Code

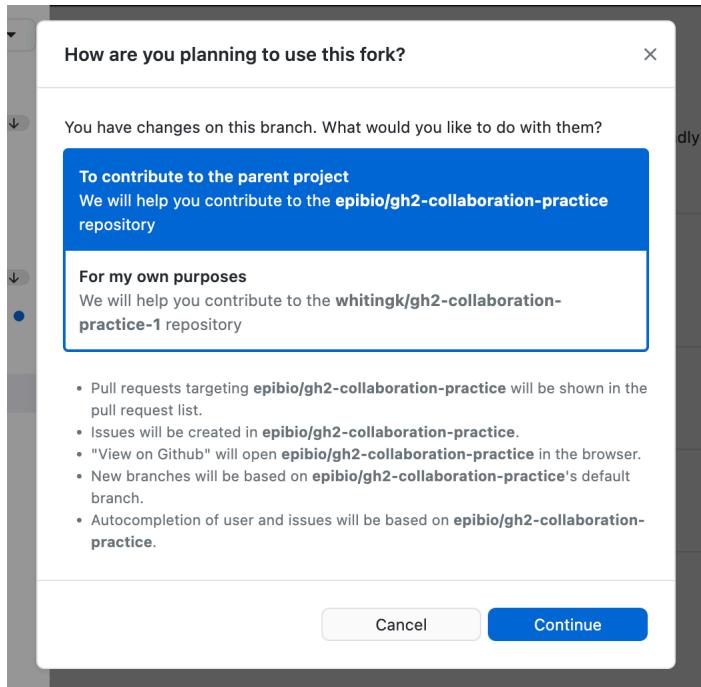
5. Click “Open with Github Desktop”

Clone Your Forked Repo



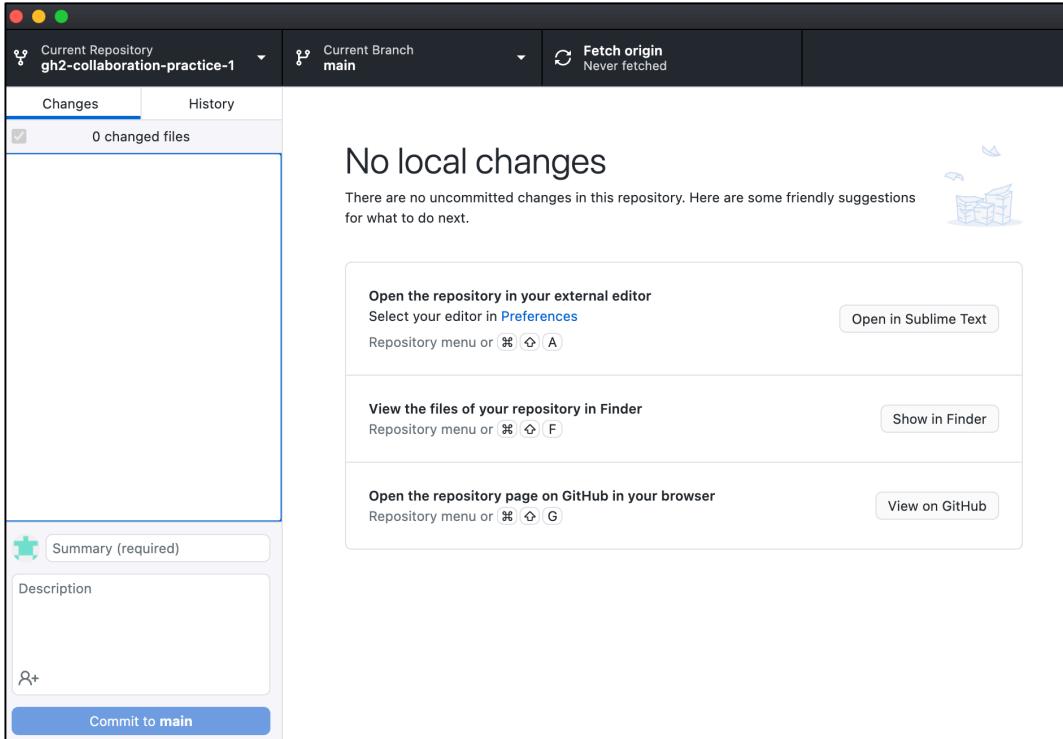
5. Select local folder location where you want to clone the forked repo

Clone Your Forked Repo



6. You will be asked the purpose of your fork. Click “contribute to the parent project”

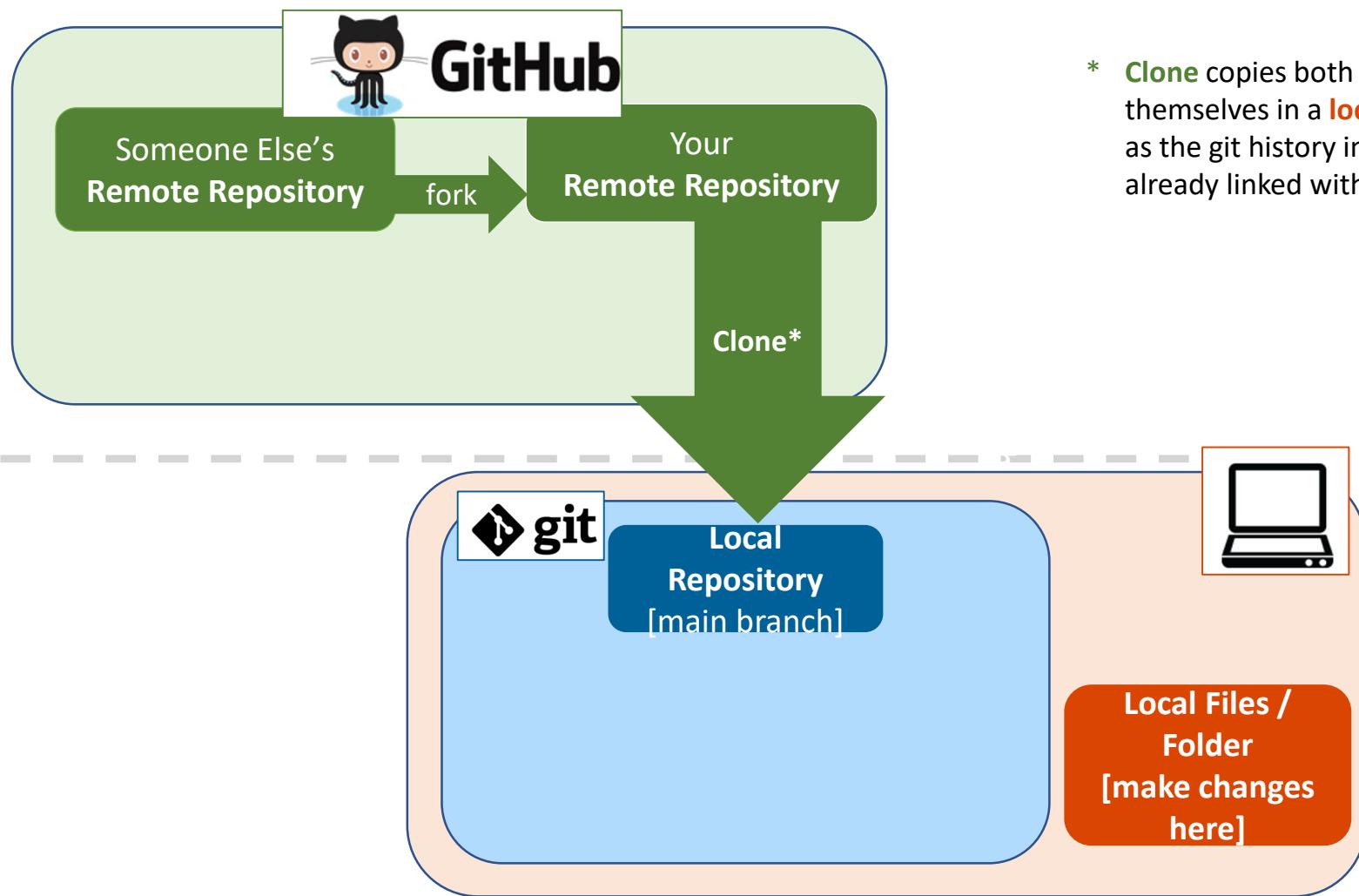
Clone Your Forked Repo



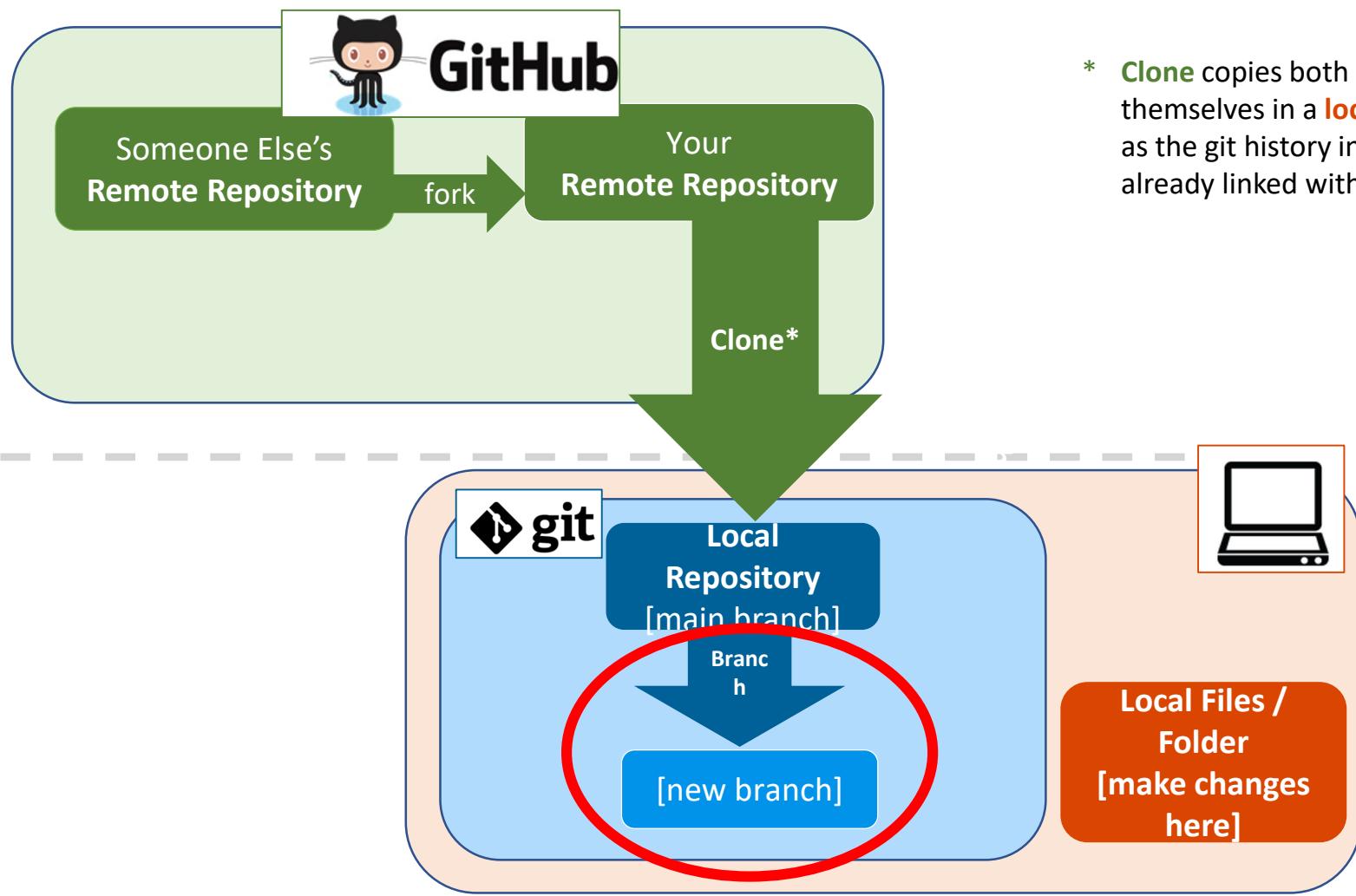
Now you should have your remote fork successfully cloned on your local computer.

Future changes to the project folder will be automatically tracked by git

Make a Branch

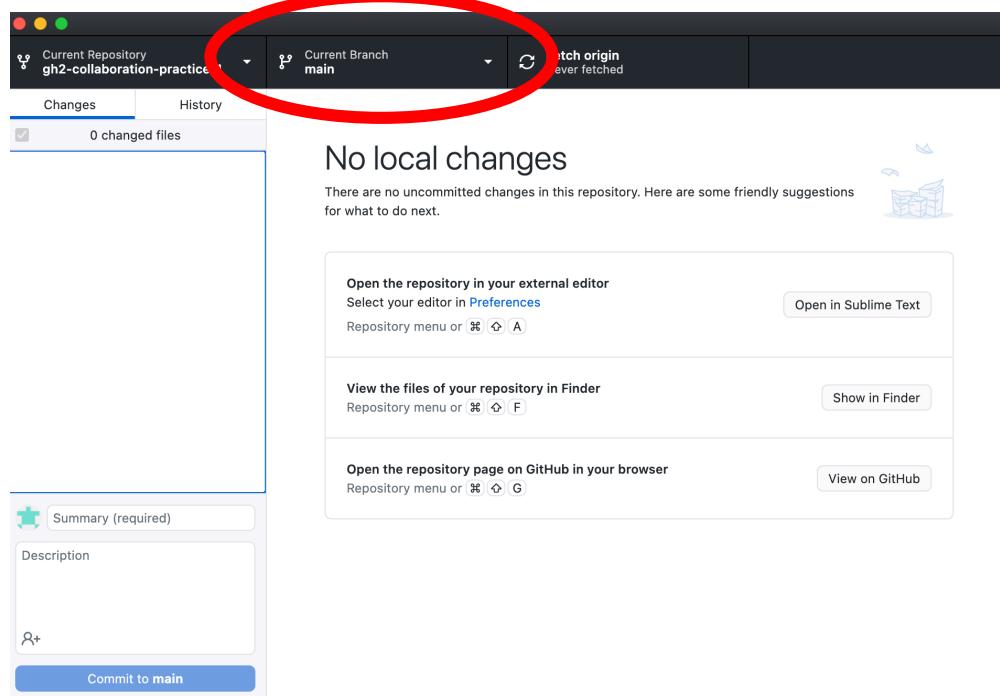


- * **Clone** copies both the project files themselves in a **local folder** as well as the git history in a **local git repo** already linked with the folder

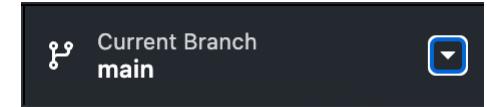


- * **Clone** copies both the project files themselves in a **local folder** as well as the git history in a **local git repo** already linked with the folder

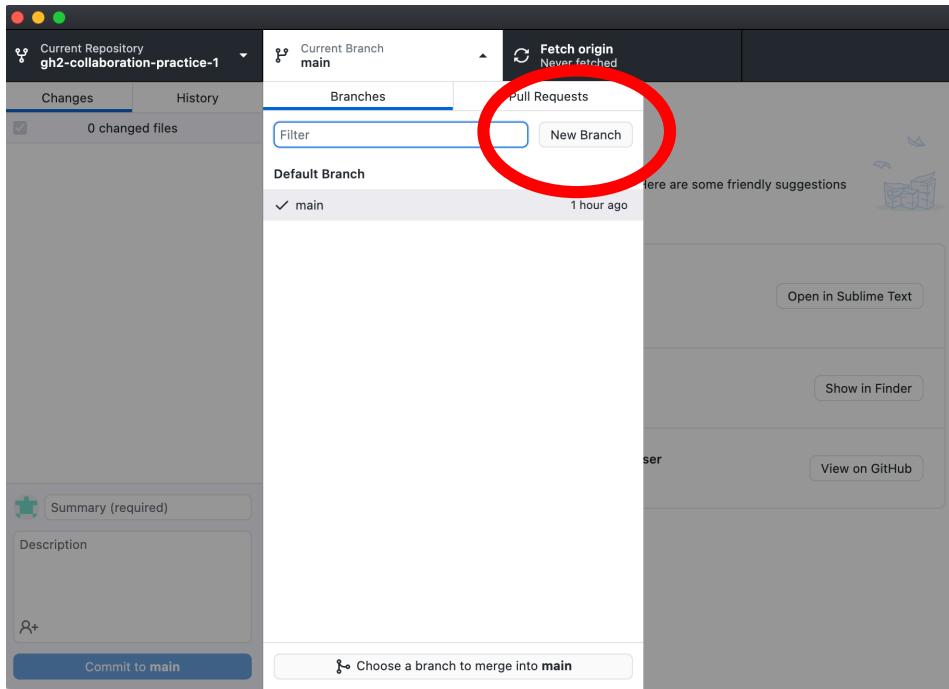
Create a Branch on Your Local Repo using Github Desktop



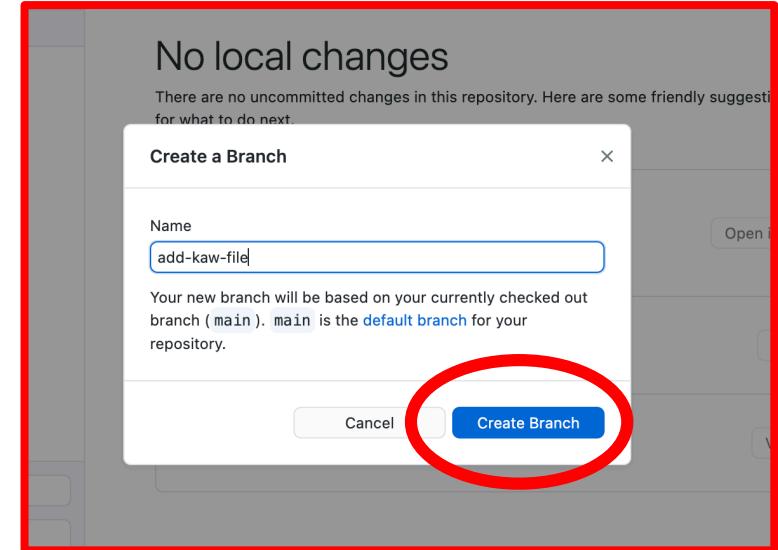
7. Click



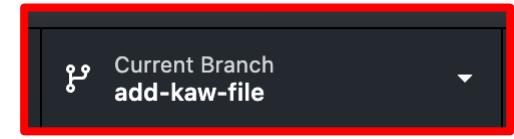
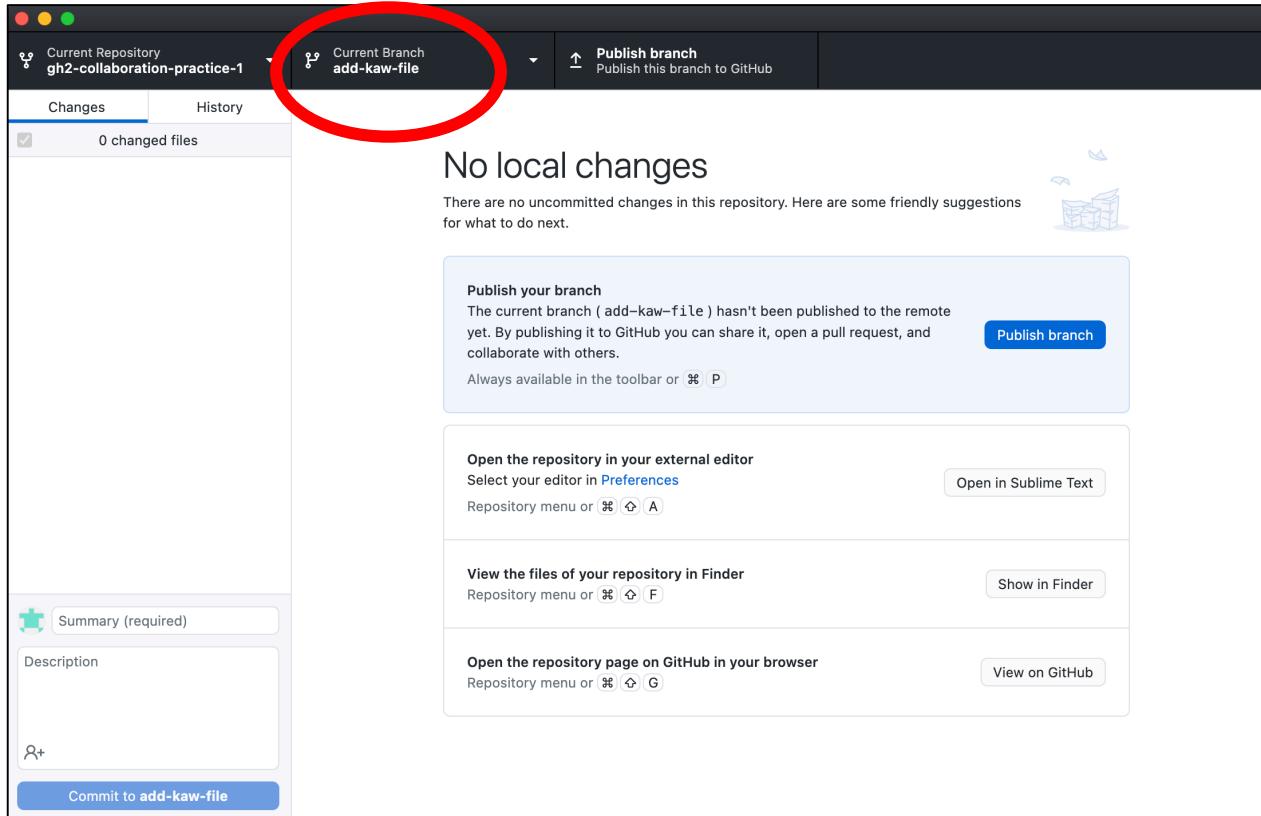
Create a Branch on Your Local Repo Using Github Desktop



8. Click **Create Branch**

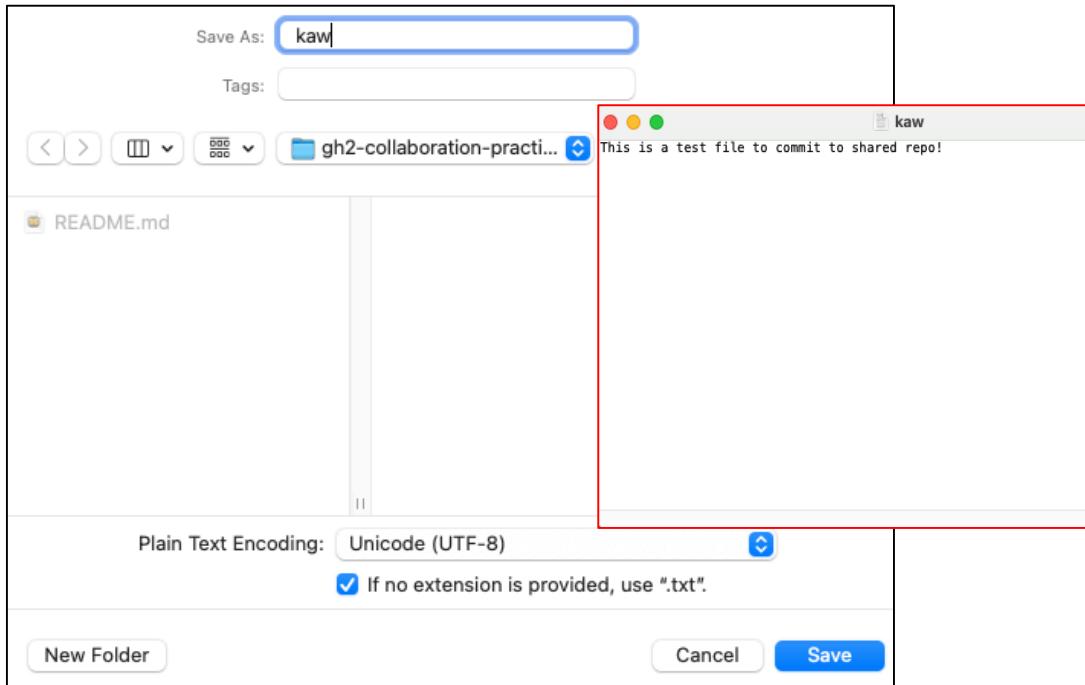


Confirm You Are on The New Branch



Add a New File To Your Project

Create A New File In Any Text Editor and Save



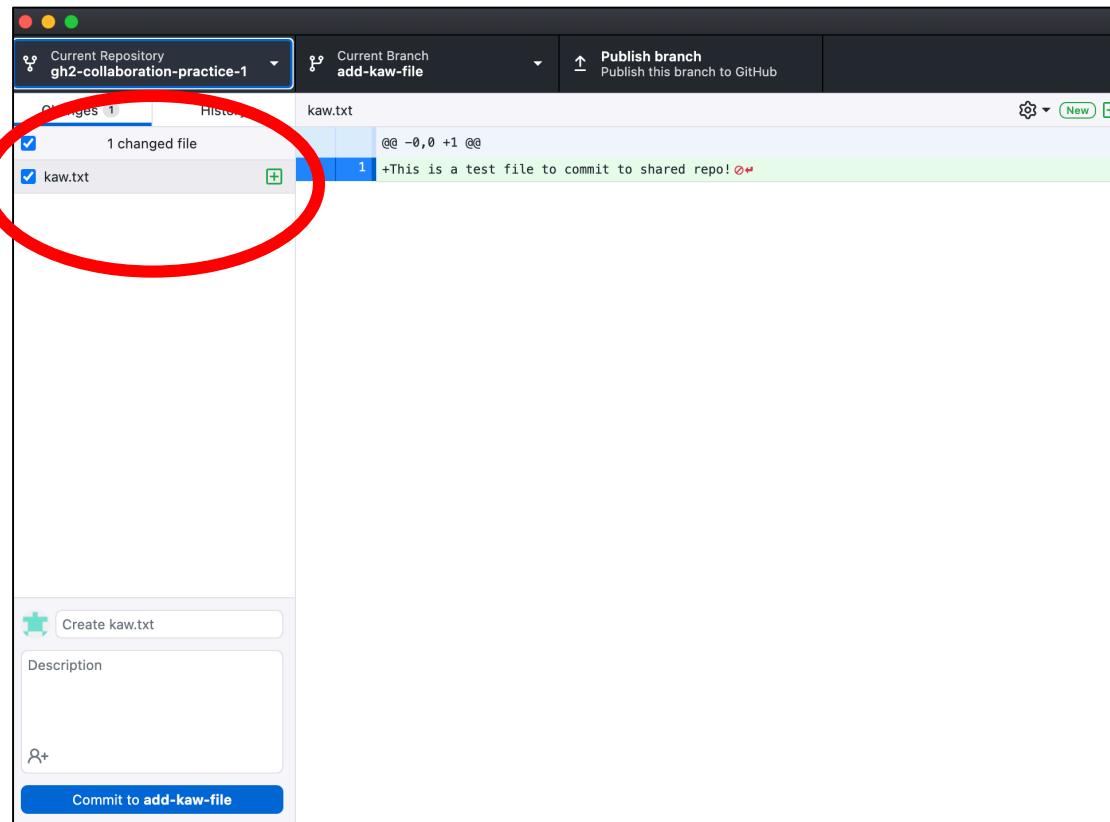
9. Create File in any text editor (e.g.TextEdit or Notepad)

10. Save file in the local folder where your repository is located

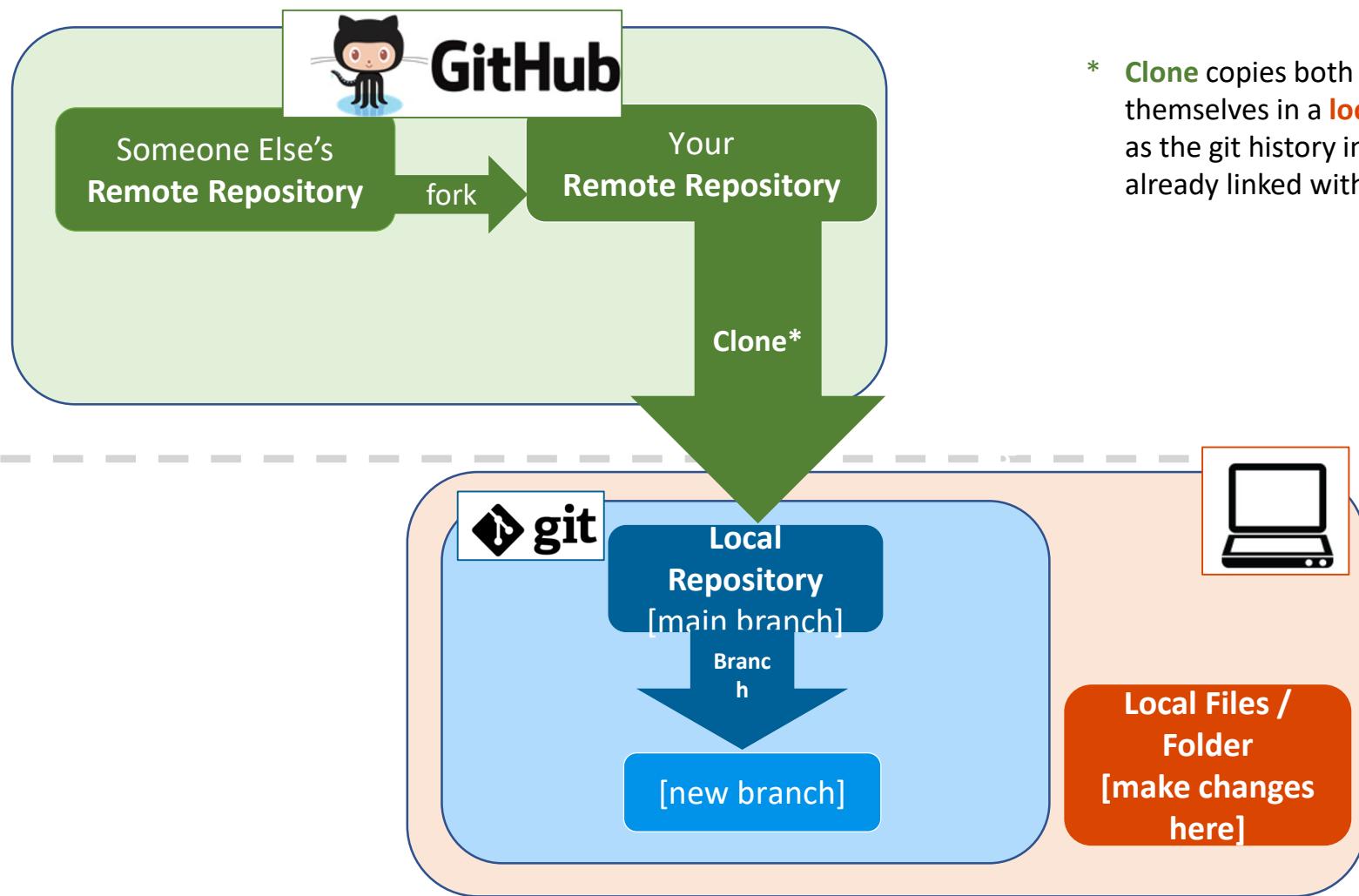
if you forgot where that is you can go to GitHub Desktop and click:

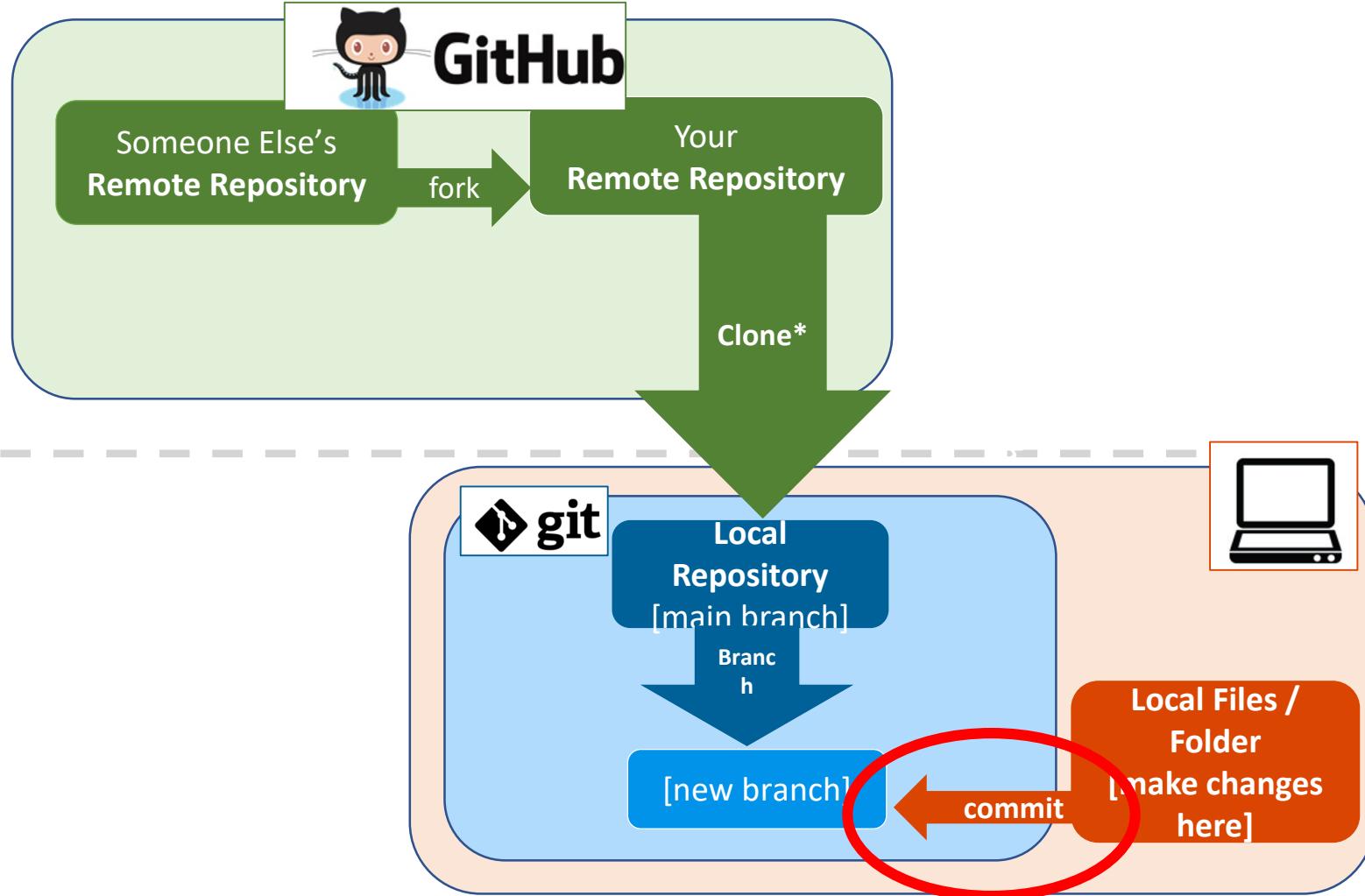
Show in Finder

Change will show in Github Desktop

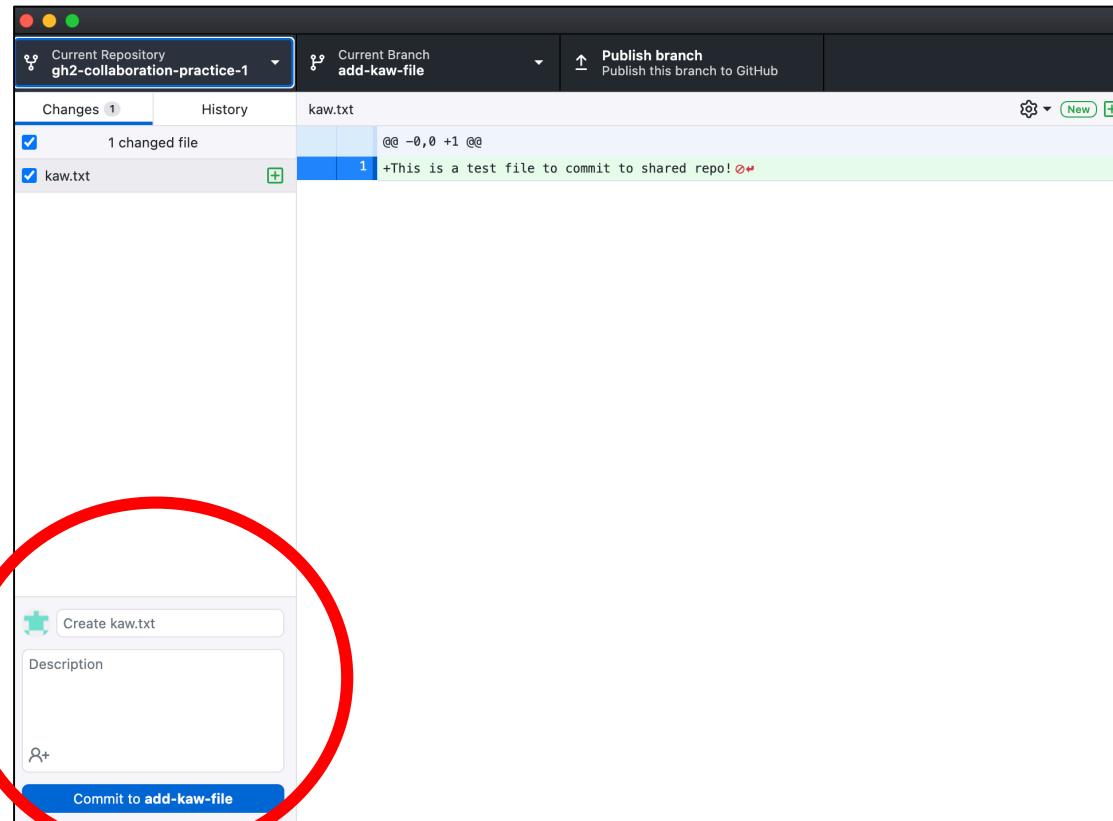


Commit Change we made in
Local Folder
To Local Repository





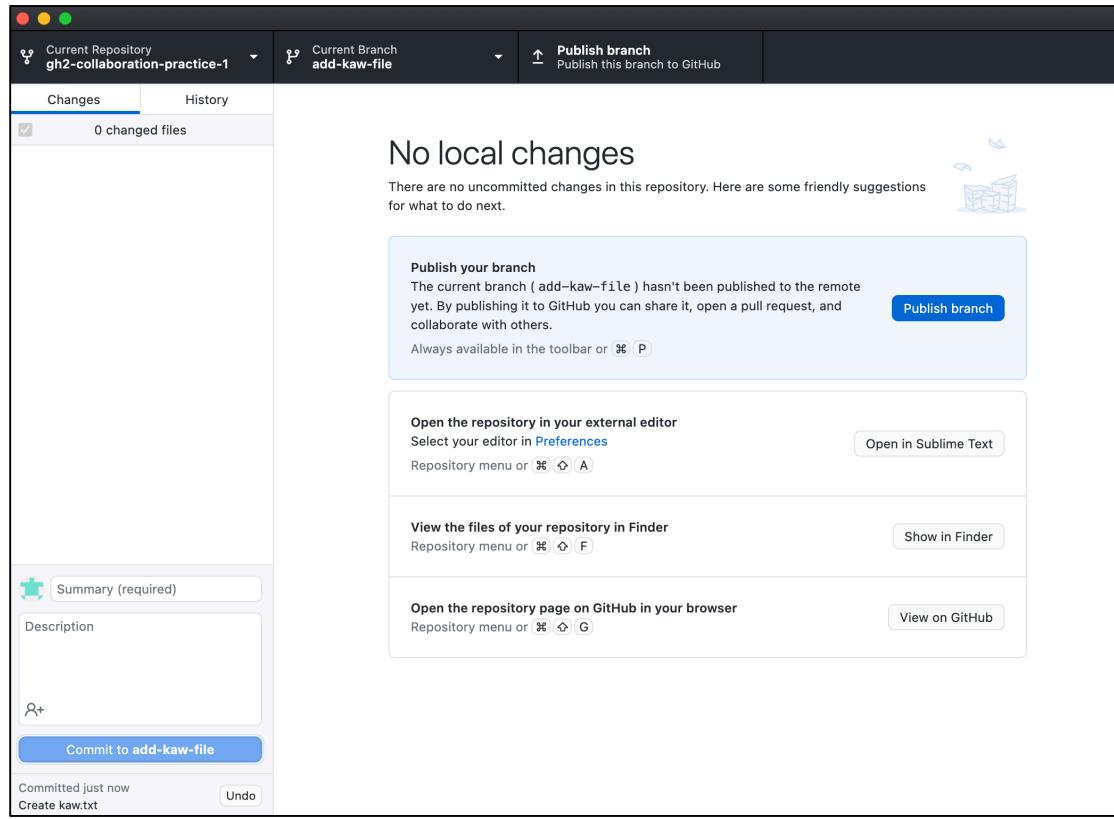
Commit Change (new file) to git



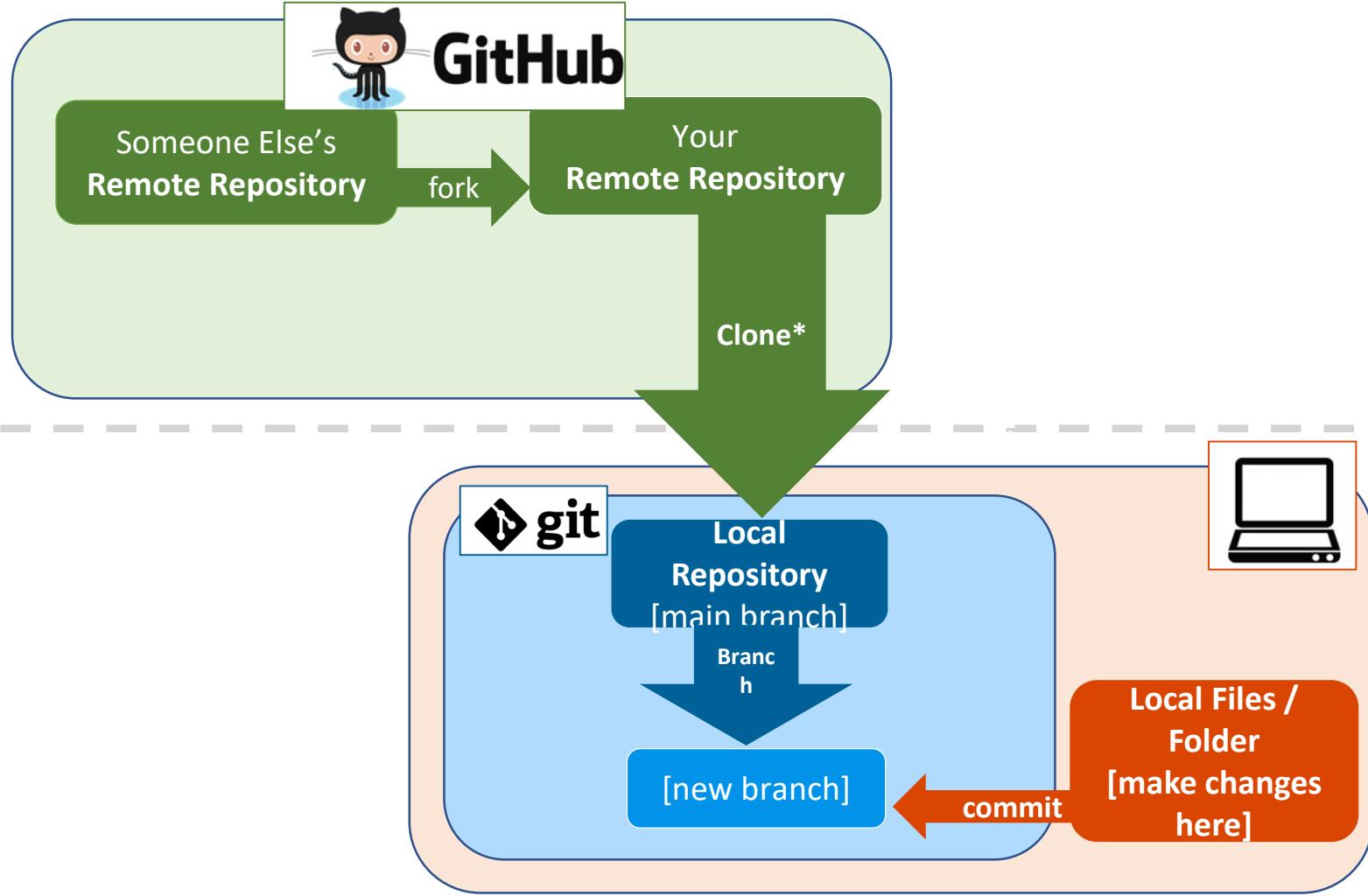
11. Commit change to git
(still local) on our new
branch

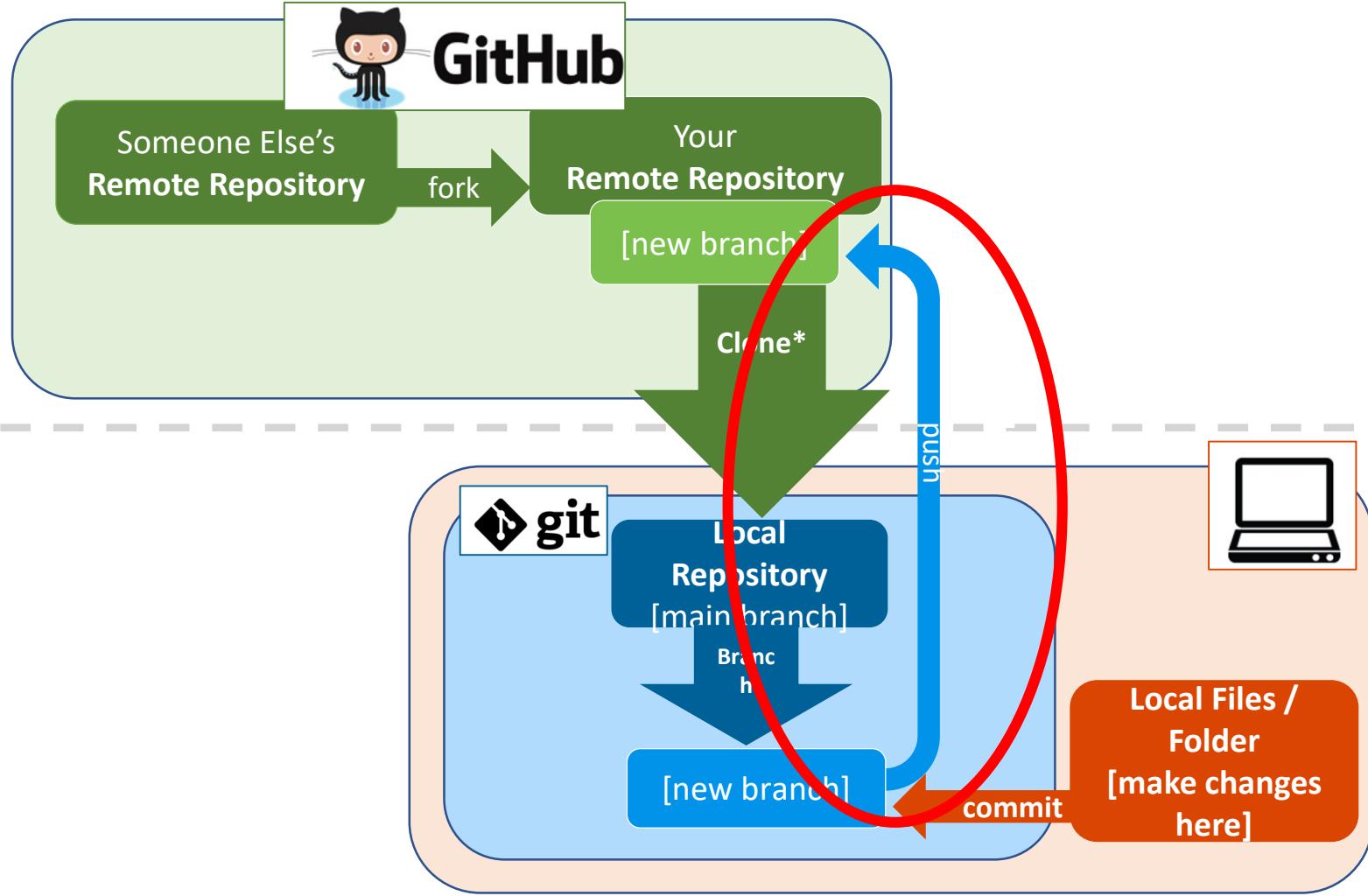
Commit to **add-kaw-file**

Commit Change (new file) to git

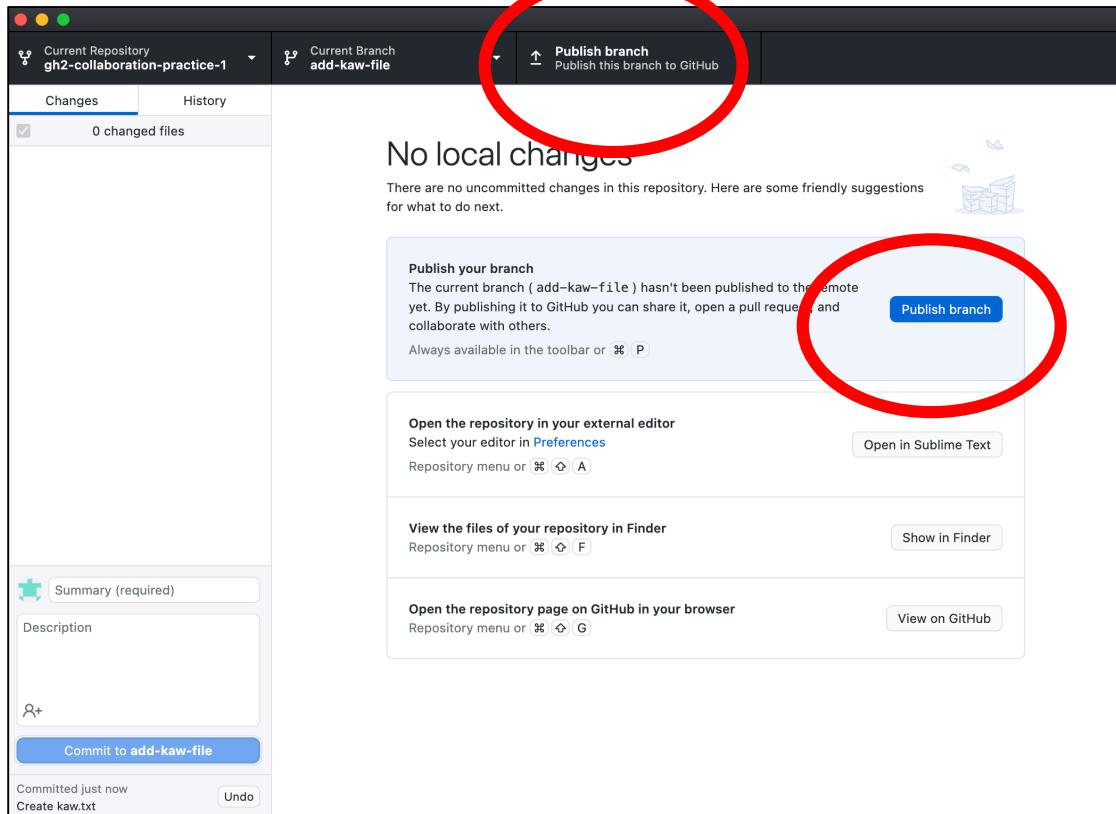


Push Your New Branch to
Your Remote Fork on
Github Enterprise





Push Change & Publish Branch on Remote Fork



12. Publish the new branch on our remote fork along with our pushed changes

Note: We could first publish branch on the remote, then make changes > commit changes > push, but here we are doing it in all one step

Confirm New Changes in Remote Fork on Github

The screenshot shows a GitHub fork page for a repository named 'Your-username/gh2-collaboration-project'. A red circle highlights the repository name in the header. Another red circle highlights the 'add-kaw-file' branch in the main content area, which displays a green banner indicating recent pushes. The commit history for the 'add-kaw-file' branch is shown, with a commit by 'whitingk' creating a file named 'kaw.txt'. Below the commit history, the repository name 'ah2-collaboration-practice' is displayed.

File	Commit Message	Time Ago
.gitignore	Initial commit	14 hours ago
README.md	Initial commit	14 hours ago
kaw.txt	Create kaw.txt	12 minutes ago

Take a minute to click around your fork
([yourname/gh2-collaboration-practice](#))

Click branches and confirm there is both a main branch as well as your newly published branch

Also click around the original repo from which you forked
([your-group-leader/gh2-collaboration-practice](#))

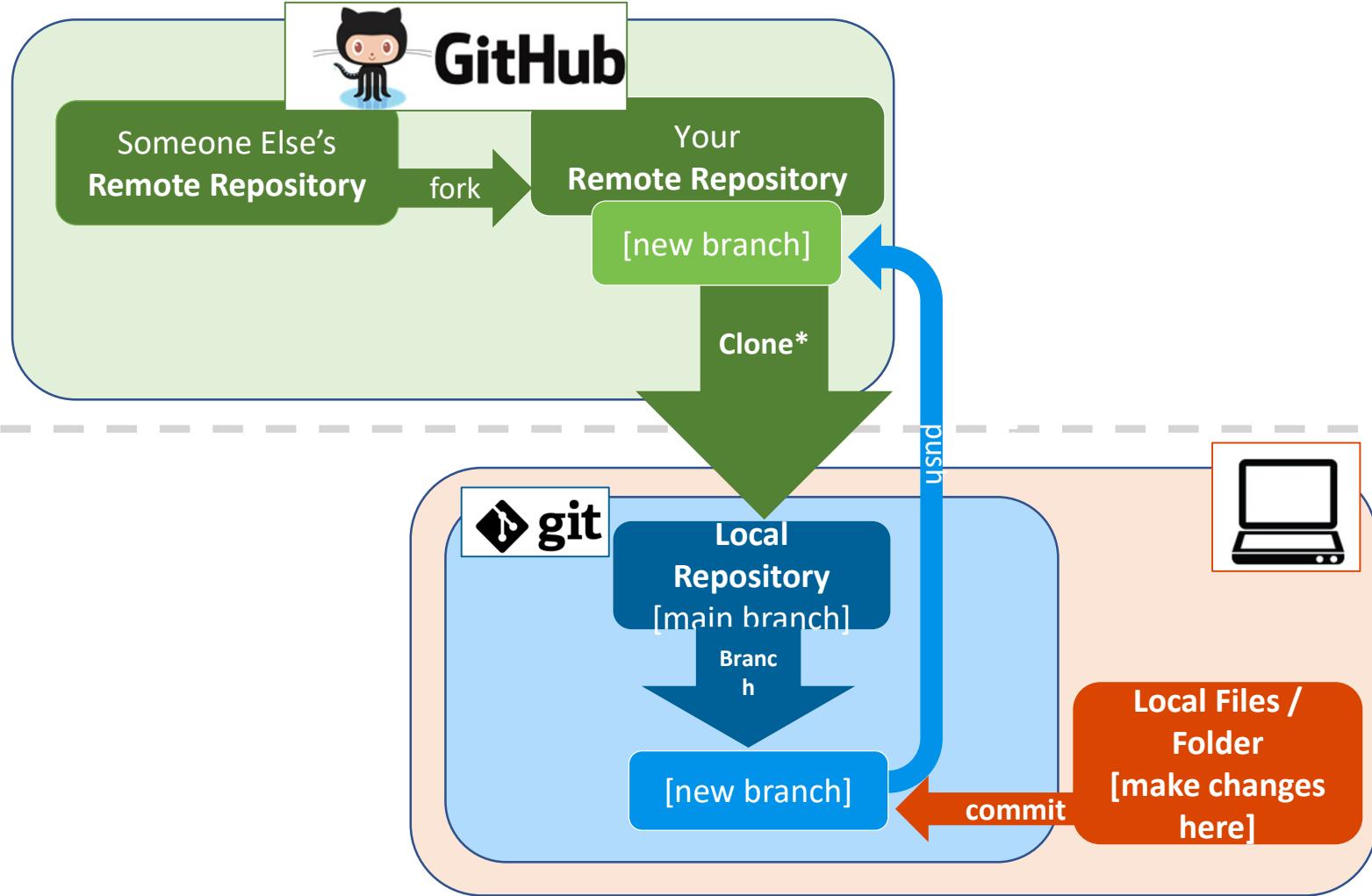
Your branch isn't there yet

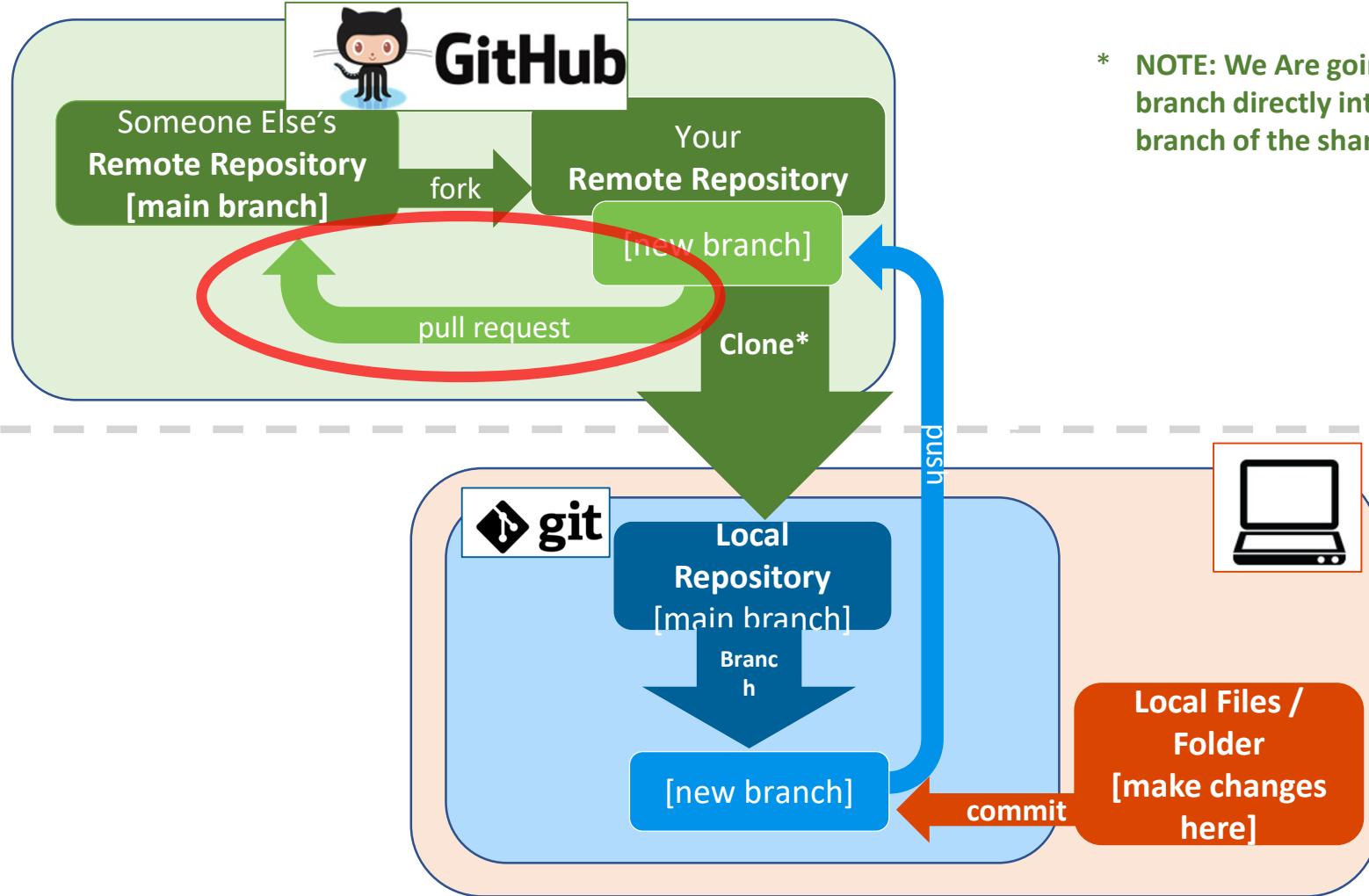
We Are Happy With Our Changes
And Ready To Merge Our
Updates With The Main Project

We Now Need to Request Our
Changes be Reviewed and Pulled
Into Main Shared Repo
(your group leader's repo)

We Now Need to Request Our
Changes be Reviewed and Pulled
Into Main Shared Repo
(your group leader's repo)

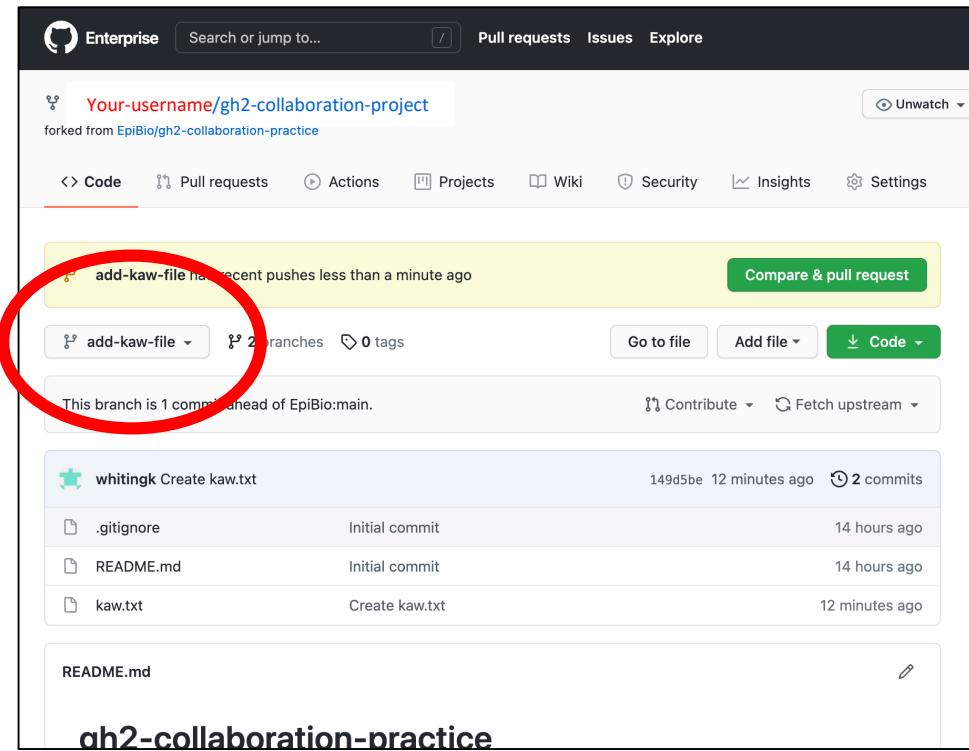
Let's Make a Pull Request





* NOTE: We Are going to merge our branch directly into the [Main] branch of the shared repo.

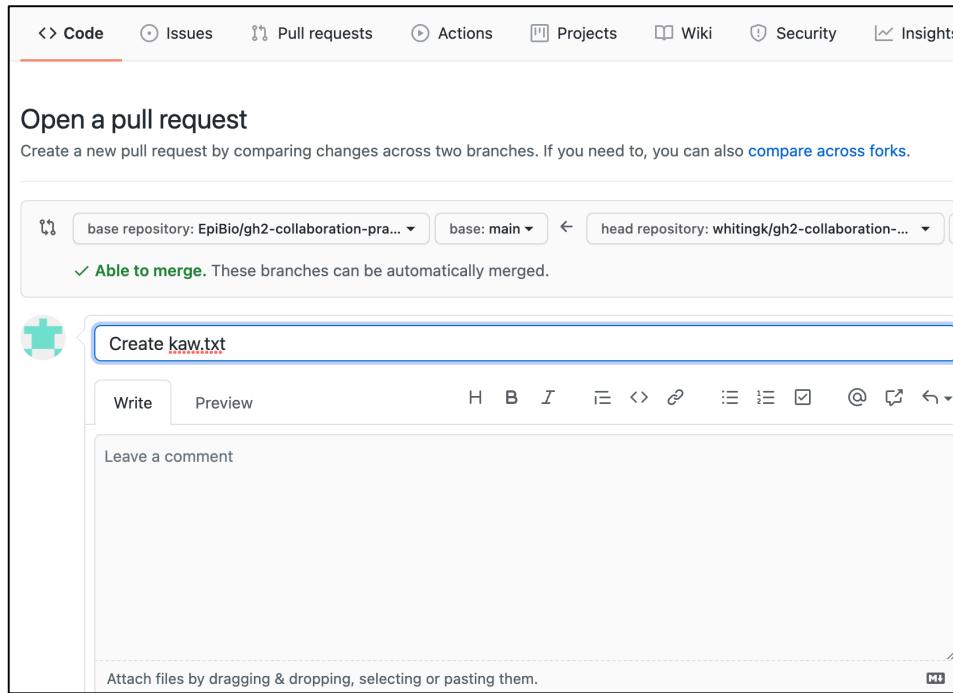
Open a Pull Request on Github



13. From github, navigate to your new branch on your fork and click

Compare & pull request

Open a Pull Request on Github



14. Describe the changes you wish to incorporate

Create pull request ▾

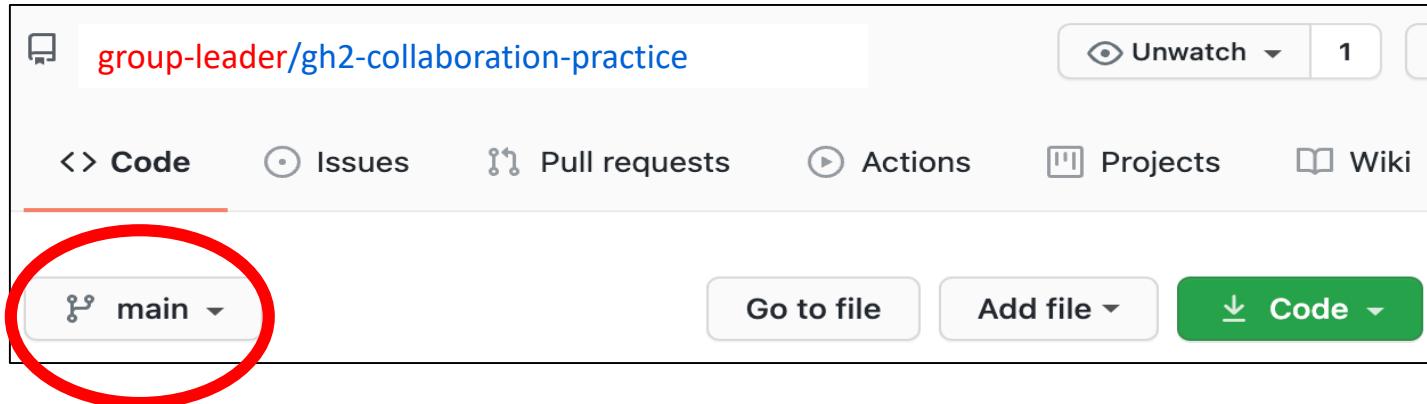
Merge Pull Request

The screenshot shows a GitHub pull request page for the repository 'EpiBio / gh2-collaboration-practice'. The pull request is titled 'Create kaw.txt #1'. It shows a single commit from 'whitingk' merging into 'EpiBio:main' from 'whitingk:add-kaw-file'. A comment from 'whitingk' says 'Adding a file with my name'. Below the commit, there is a note: 'Add more commits by pushing to the [add-kaw-file](#) branch on [whitingk/gh2-collaboration-practice-1](#)'. A green callout box highlights a message: 'This branch has no conflicts with the base branch. Merging can be performed automatically.' At the bottom, there is a green 'Merge pull request' button.

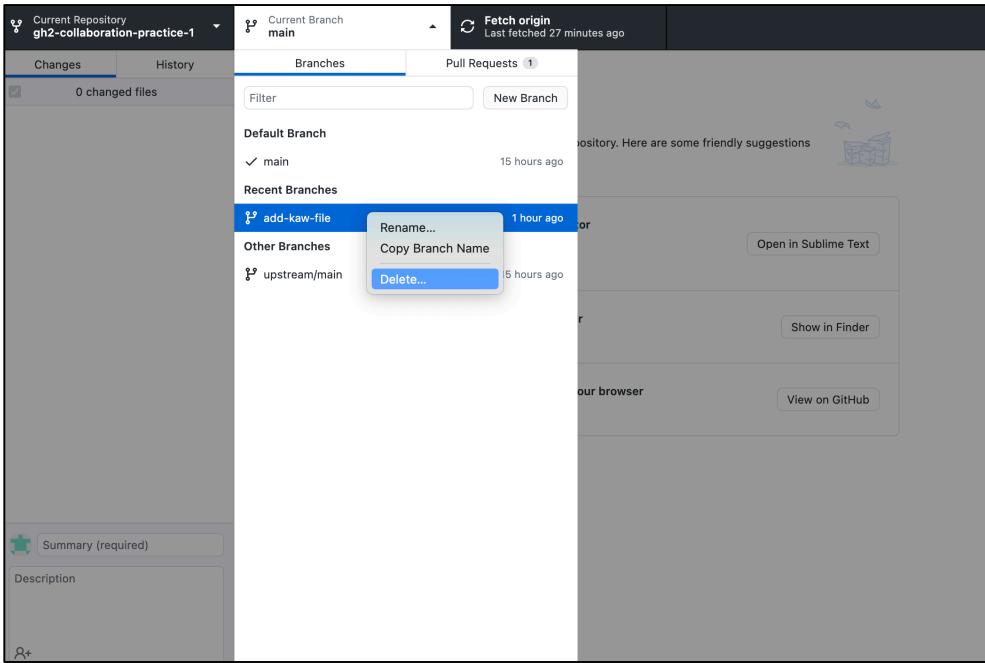
15. Your ‘Code’ will be reviewed and either you or your group leader will merge your changes



Confirm Your Changes on Your Group Leader's Remote Repo

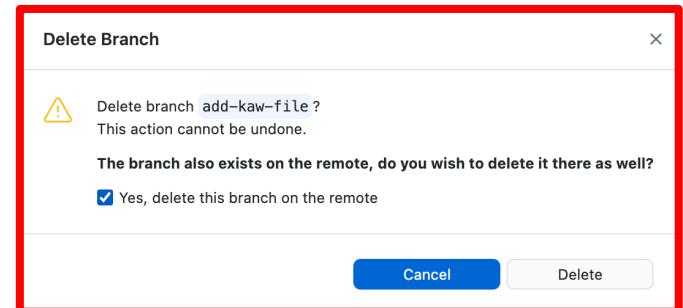


Delete The Branch (optional)



16. Once your changes are merged into the main project, it may be a good idea to delete the branch.

You can do this from Github Desktop

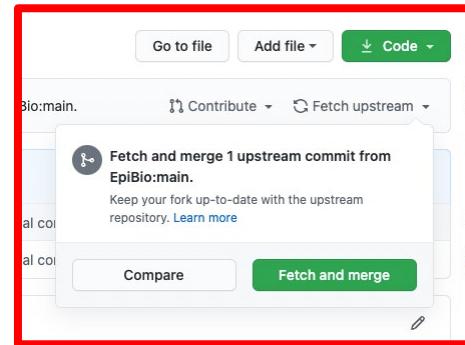


Lastly, Let's Update Our Fork
With Recent Changes Others
Have Made To The Original
Repo (Upstream)

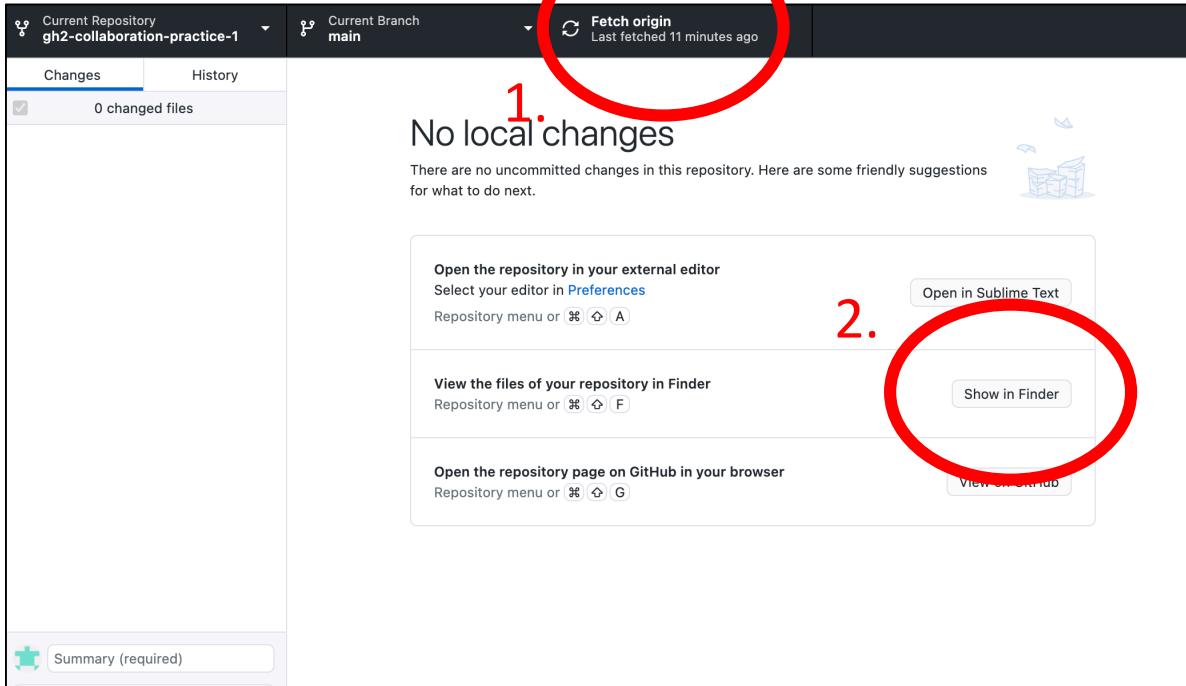
Update Our Remote Fork on Github With Upstream Changes

A screenshot of a GitHub repository page for 'Your-username/gh2-collaboration-project'. The page shows the 'main' branch. A red circle highlights the status bar at the top which says 'This branch is 1 commit behind group-leader:main'. Below the status bar, there's a list of files: '.gitignore' (Initial commit, 15 hours ago), 'README.md' (Initial commit, 15 hours ago). At the bottom, there's a code editor with the file 'README.md' containing the text 'gh2-collaboration-practice'.

16. There are several ways you can do this. Its easy to do on Github by clicking “Fetch Upstream”



Then Update Our Local Repository and Folder With Changes



17. Go to Github Desktop and “Fetch Origin”.

Now you should have everyone else’s changes in in your local folder/local repository!

Questions?

Thank You!

