ReproRehab

Week 2: Tools for reproducible science II

ReproRehab POD1, Week 2

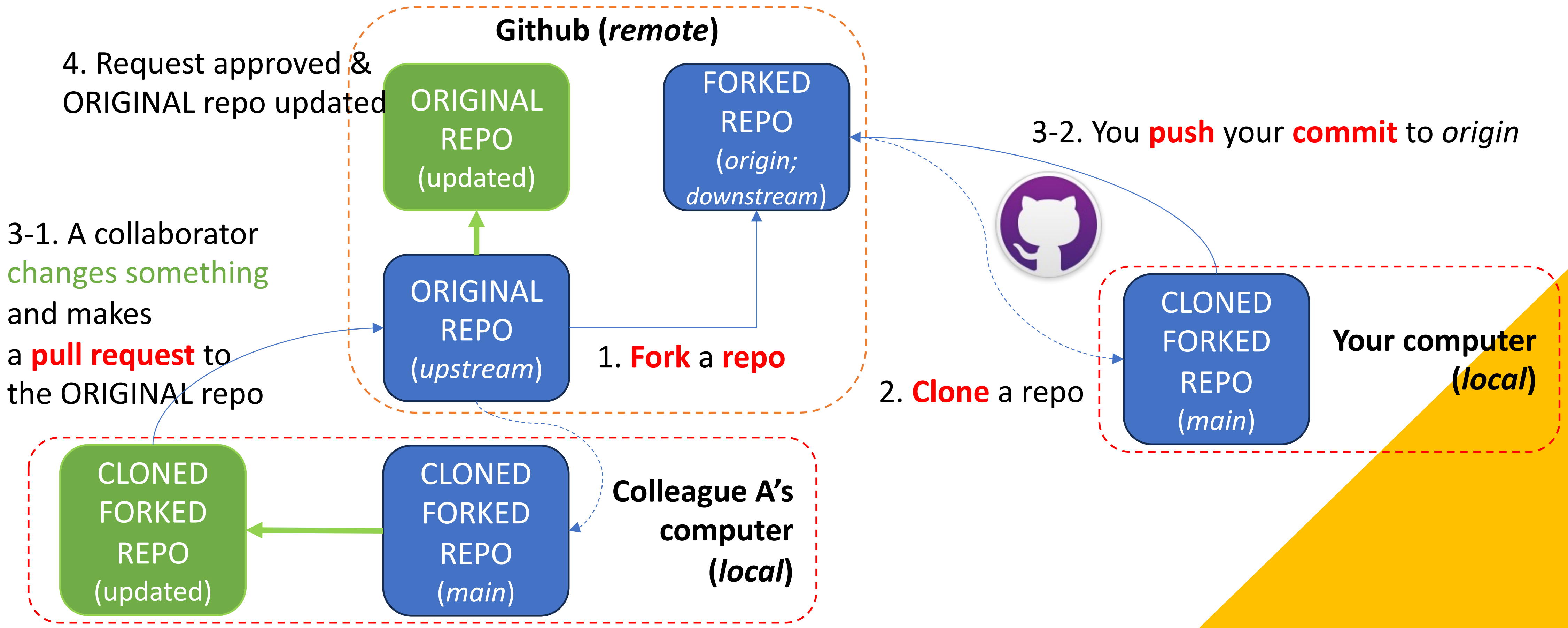
Agenda

- Review: collaborating with Git/Github

- Use Github Desktop to:
  1) clone your forked repository to your computer
  2) upload (*add*, *commit* and *push*) files to your remote repository
  3) make a new repository, make a new branch, and merge branches
     - public vs. private repository
     - inviting collaborators to work together

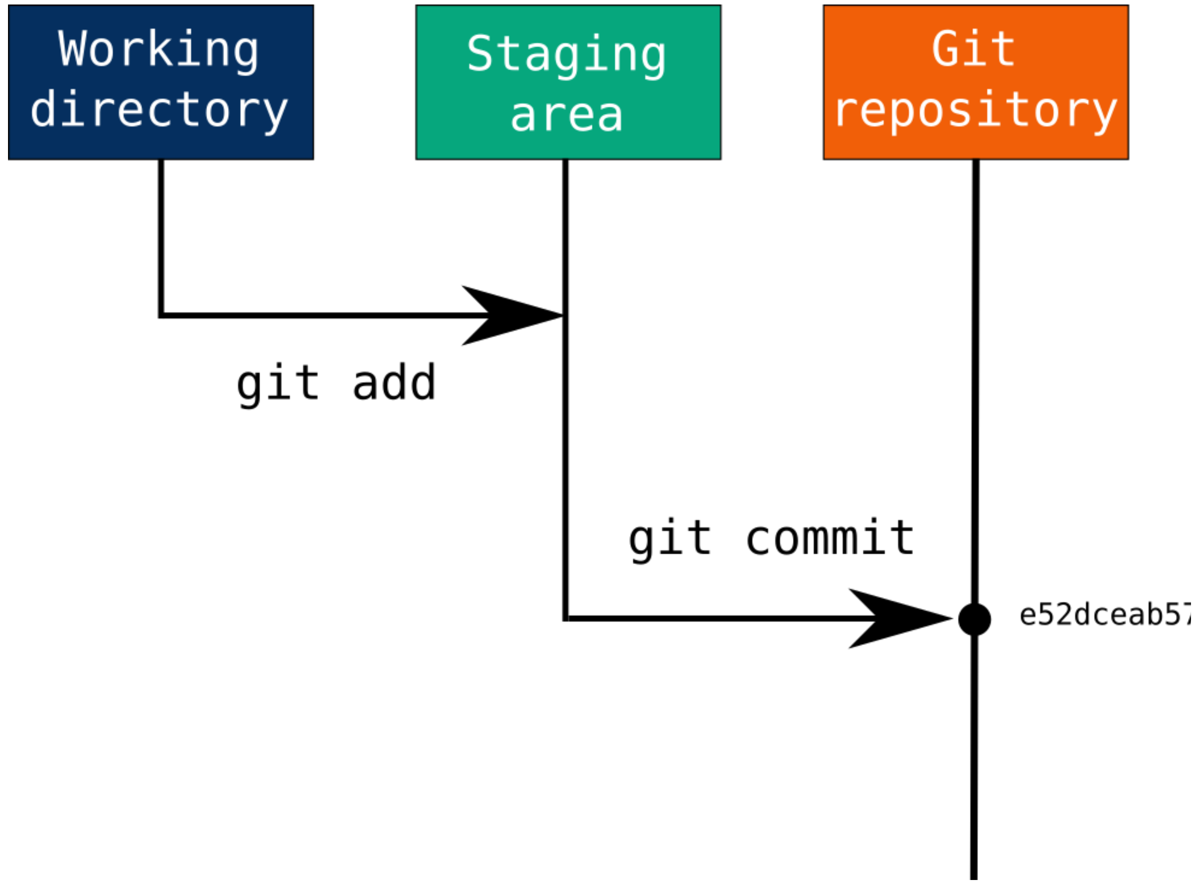
- Data cloning using Datalad / Repos to share data

Collaborating with Git/Github

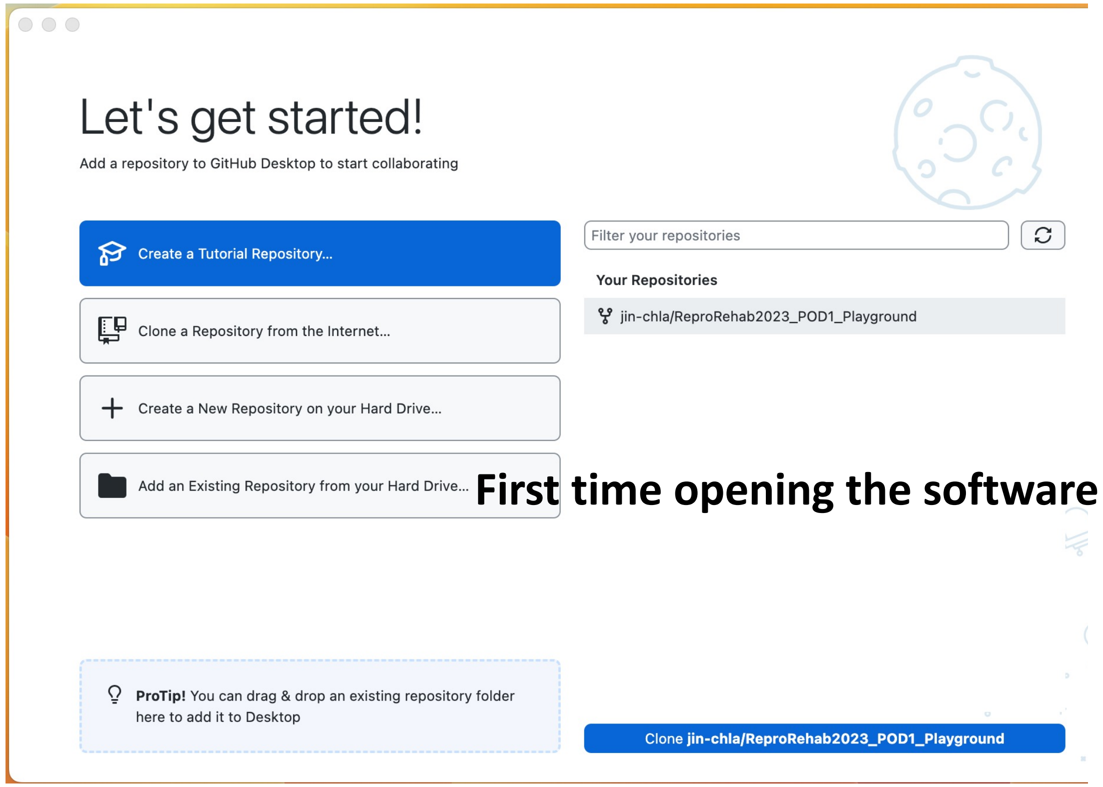

Github (*remote*)
4. Request approved & ORIGINAL repo updated
ORIGINAL REPO (updated)
FORKED REPO (*origin; downstream*)
3-2. You **push** your **commit** to *origin*
3-1. A collaborator changes something and makes a **pull request** to the ORIGINAL repo
ORIGINAL REPO (*upstream*)
1. **Fork** a **repo**
2. **Clone** a repo
CLONED FORKED REPO (*main*)
Your computer (*local*)
CLONED FORKED REPO (updated)
CLONED FORKED REPO (*main*)
Colleague A's computer (*local*)

Add, commit and push changes

- Technically, a repository is a folder named **.git** (This is hidden, so not easy to access)
- Changes you make on your computer is added to the staging area (done automatically in Github Desktop)
- You then commit the staged changes to your local git repository
- Finally you push your commit from the local to remote repository (the one in Github.com)



Cloning a repository



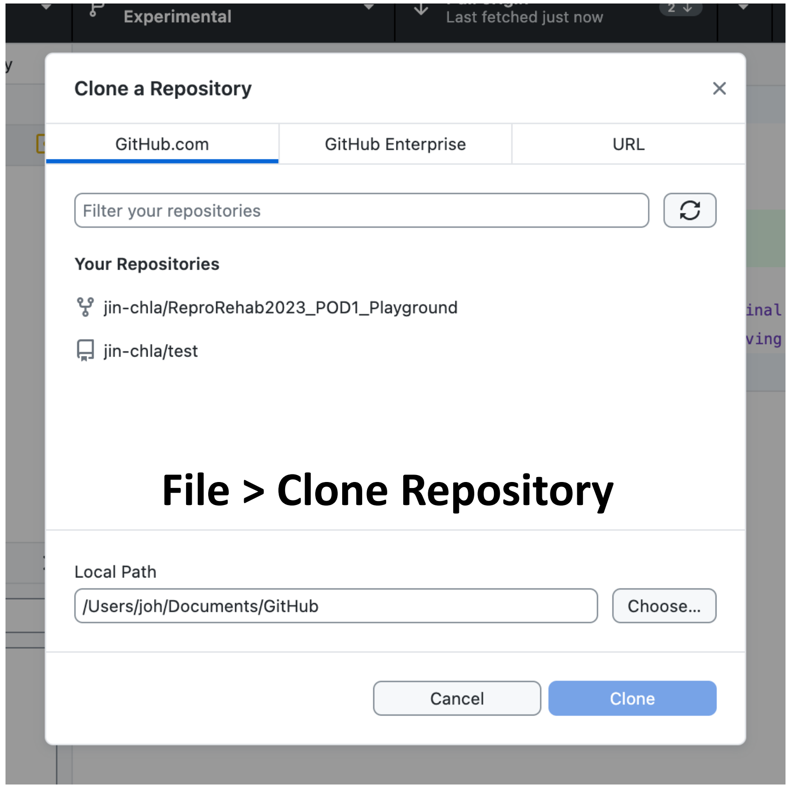
Let's get started!

Add a repository to GitHub Desktop to start collaborating

Create a Tutorial Repository...
Clone a Repository from the Internet...
Create a New Repository on your Hard Drive...
Add an Existing Repository from your Hard Drive...
Filter your repositories
Your Repositories
jin-chla/ReproRehab2023_POD1_Playground
Clone **jin-chla/ReproRehab2023_POD1_Playground**

ProTip! You can drag & drop an existing repository folder here to add it to Desktop

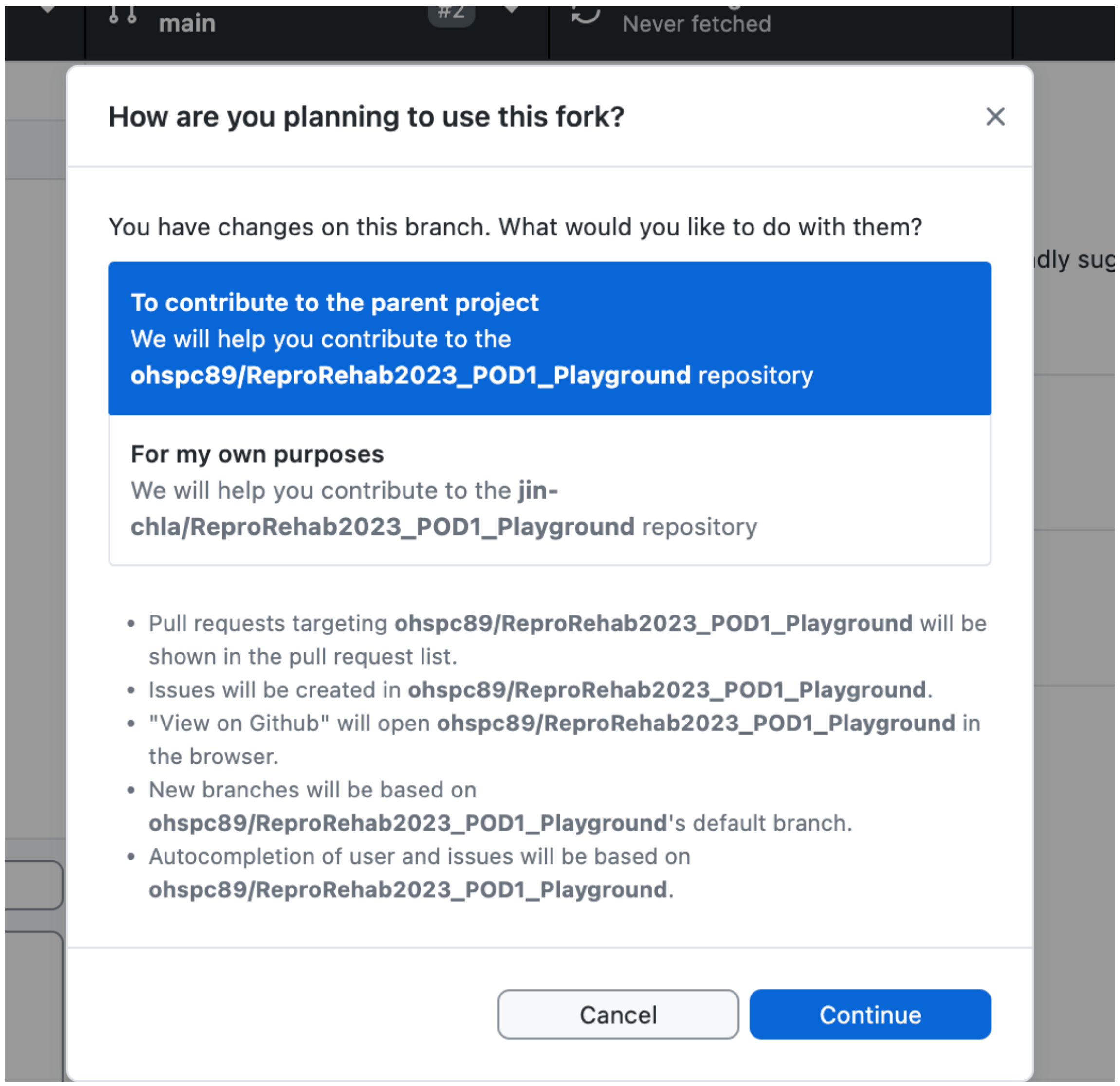
First time opening the software



File > Clone Repository

Cloning a repository

- If you clone a *forked* repository, you will see the screen on the right.
- See the next slide for more details.



How are you planning to use this fork?

You have changes on this branch. What would you like to do with them?

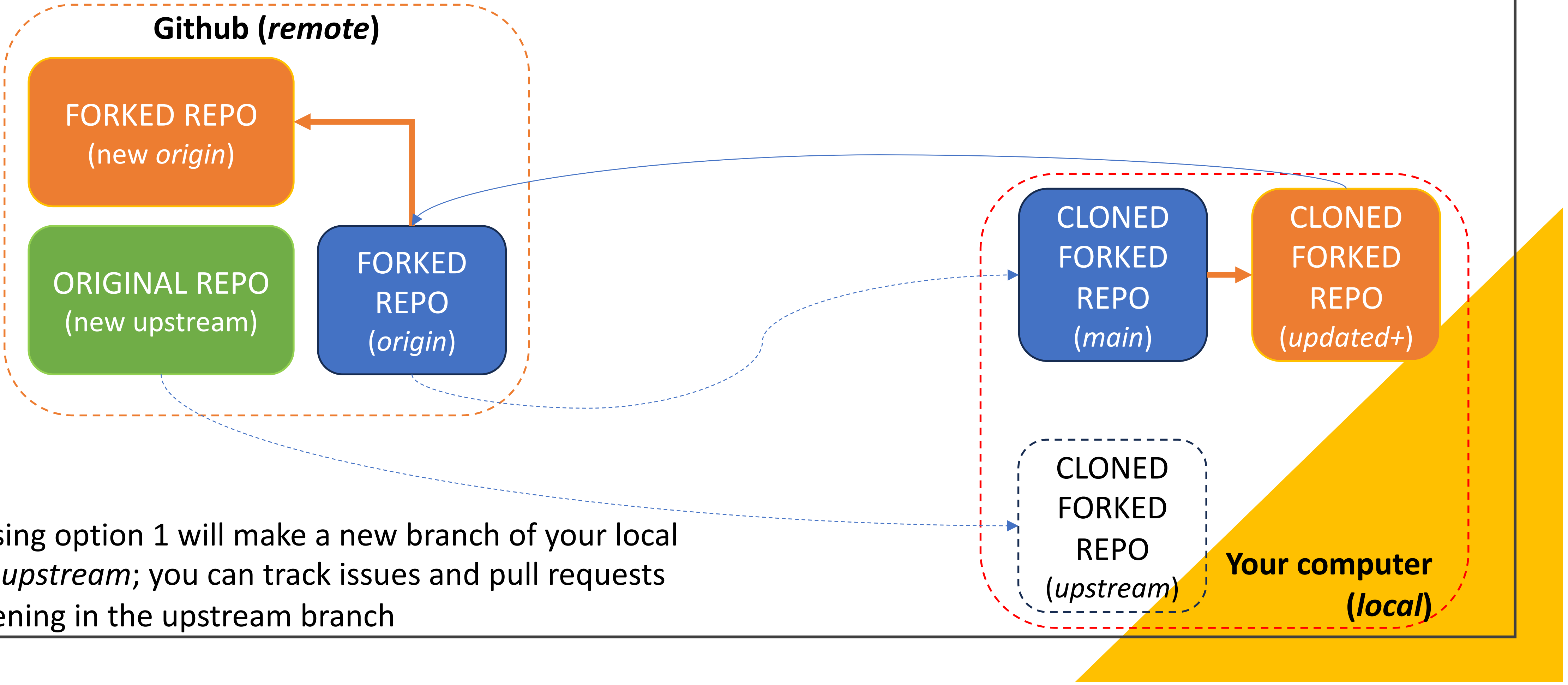
To contribute to the parent project
We will help you contribute to the **ohspc89/ReproRehab2023_POD1_Playground** repository

For my own purposes
We will help you contribute to the **jin-chla/ReproRehab2023_POD1_Playground** repository

- Pull requests targeting **ohspc89/ReproRehab2023_POD1_Playground** will be shown in the pull request list.
- Issues will be created in **ohspc89/ReproRehab2023_POD1_Playground**.
- "View on Github" will open **ohspc89/ReproRehab2023_POD1_Playground** in the browser.
- New branches will be based on **ohspc89/ReproRehab2023_POD1_Playground**'s default branch.
- Autocompletion of user and issues will be based on **ohspc89/ReproRehab2023_POD1_Playground**.

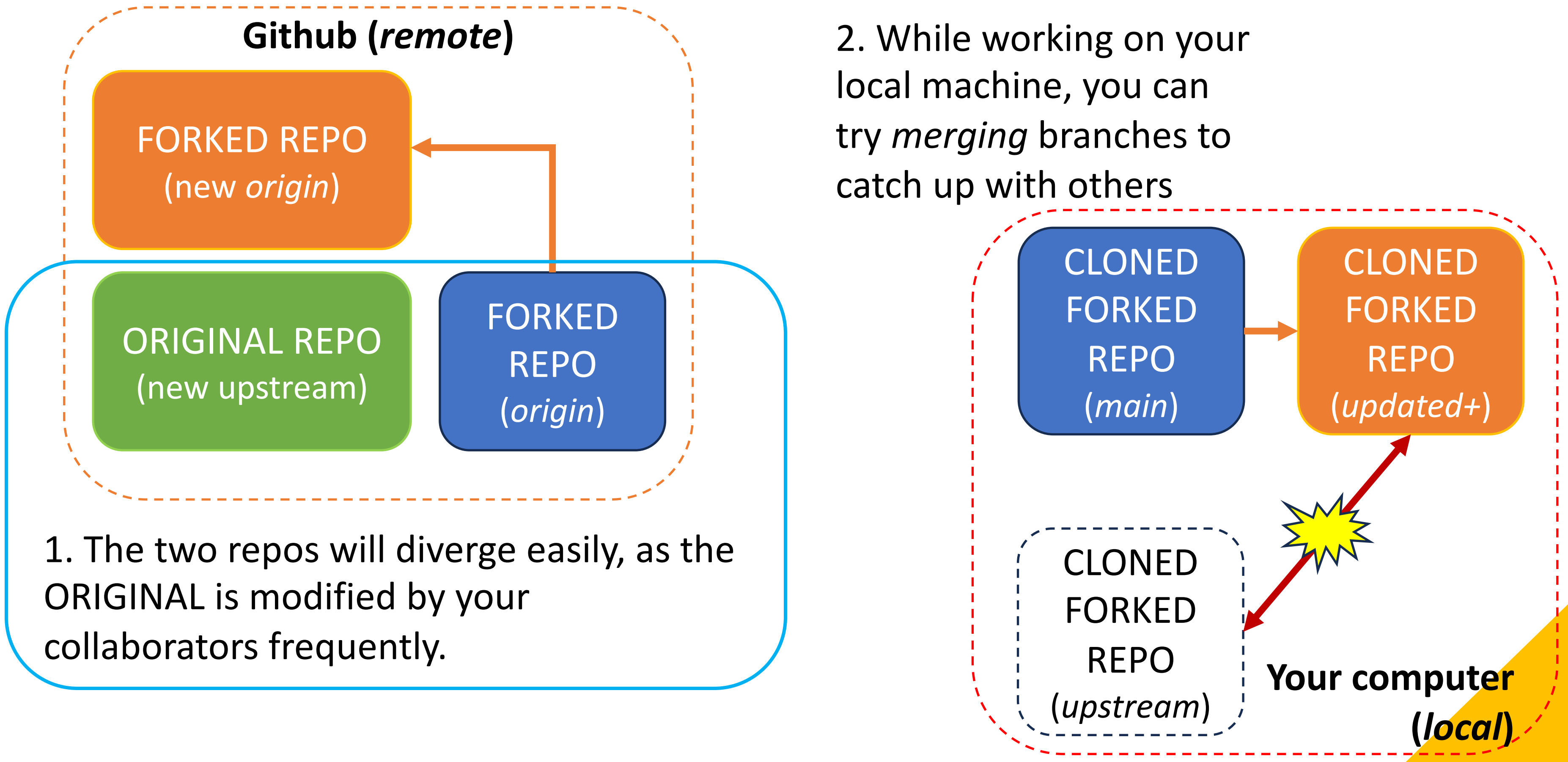
Cancel Continue

Fork option 1: prioritize the parent (upstream)


Github (*remote*)
FORKED REPO (new *origin*)
ORIGINAL REPO (new upstream)
FORKED REPO (*origin*)
CLONED FORKED REPO (*main*)
CLONED FORKED REPO (*updated+*)
CLONED FORKED REPO (*upstream*)
Your computer (*local*)

Choosing option 1 will make a new branch of your local repo: *upstream*; you can track issues and pull requests happening in the upstream branch

Fork option 1: prioritize the parent (upstream)



1. The two repos will diverge easily, as the ORIGINAL is modified by your collaborators frequently.

2. While working on your local machine, you can try *merging* branches to catch up with others

3. You can check for any conflicts (same files modified differently) and resolve.

Example of conflict

- README.md file

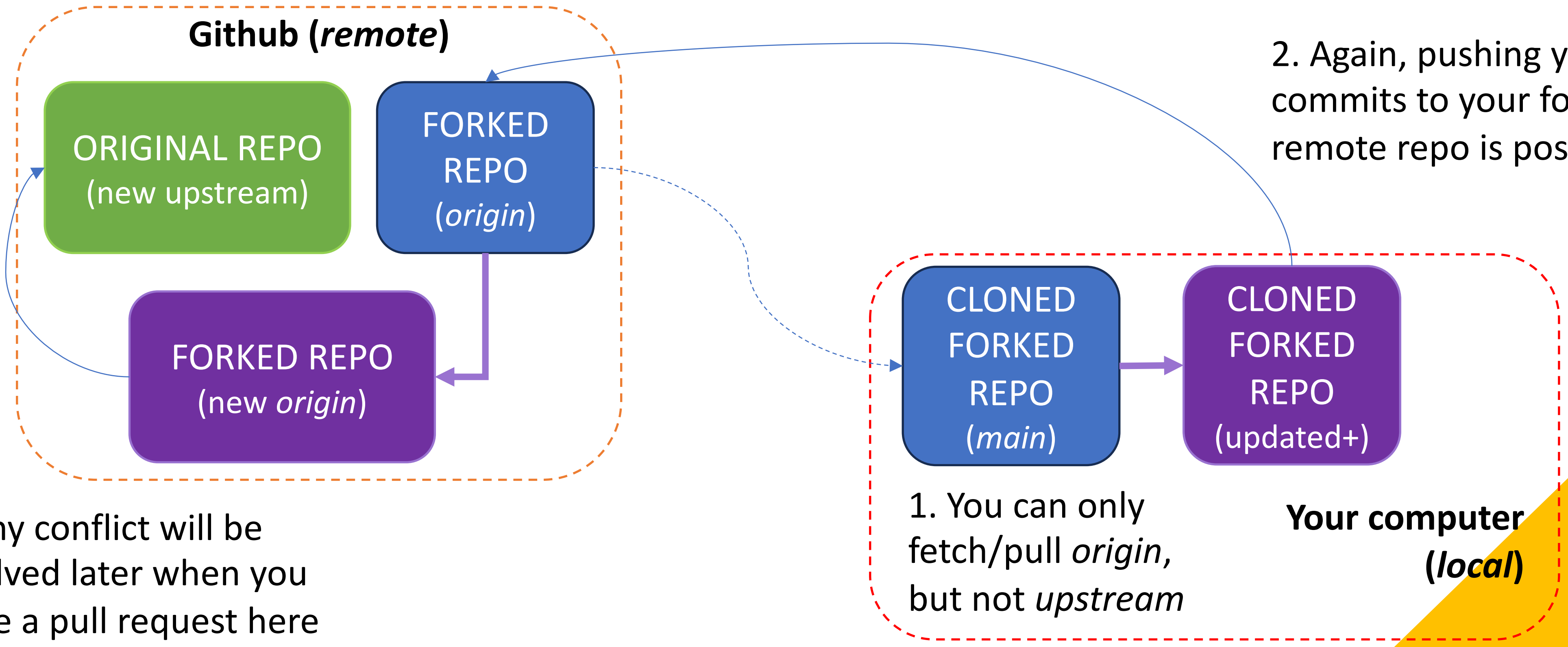
[Original version in upstream]
“This is a repository prepared for ReproRehab2023 POD 1 learners. Please fork this repository, make edits in your branch, and make pull requests!”

[Your colleague’s edit saved on upstream]
“This is a repository prepared for ReproRehab2023 POD 1 learners. ~~Please fork this repository, make edits in your branch, and make pull requests!”~~
I hope you enjoy working with Git/Github!

[Edits you made in your local main branch]
“This is a repository prepared for ReproRehab2023 POD 1 learners. Please fork this repository, make edits in your branch, and make pull requests! \n\n Date added: 10/5/2023 \n\n Adding anotherline here \n\n LALALALALA”

Git: How should I merge the two?

Fork option 2: prioritize the forked repo (origin)


Github (*remote*)
ORIGINAL REPO (new upstream)
FORKED REPO (*origin*)
FORKED REPO (new *origin*)
CLONED FORKED REPO (*main*)
CLONED FORKED REPO (updated+)
Your computer (*local*)

2. Again, pushing your commits to your forked remote repo is possible

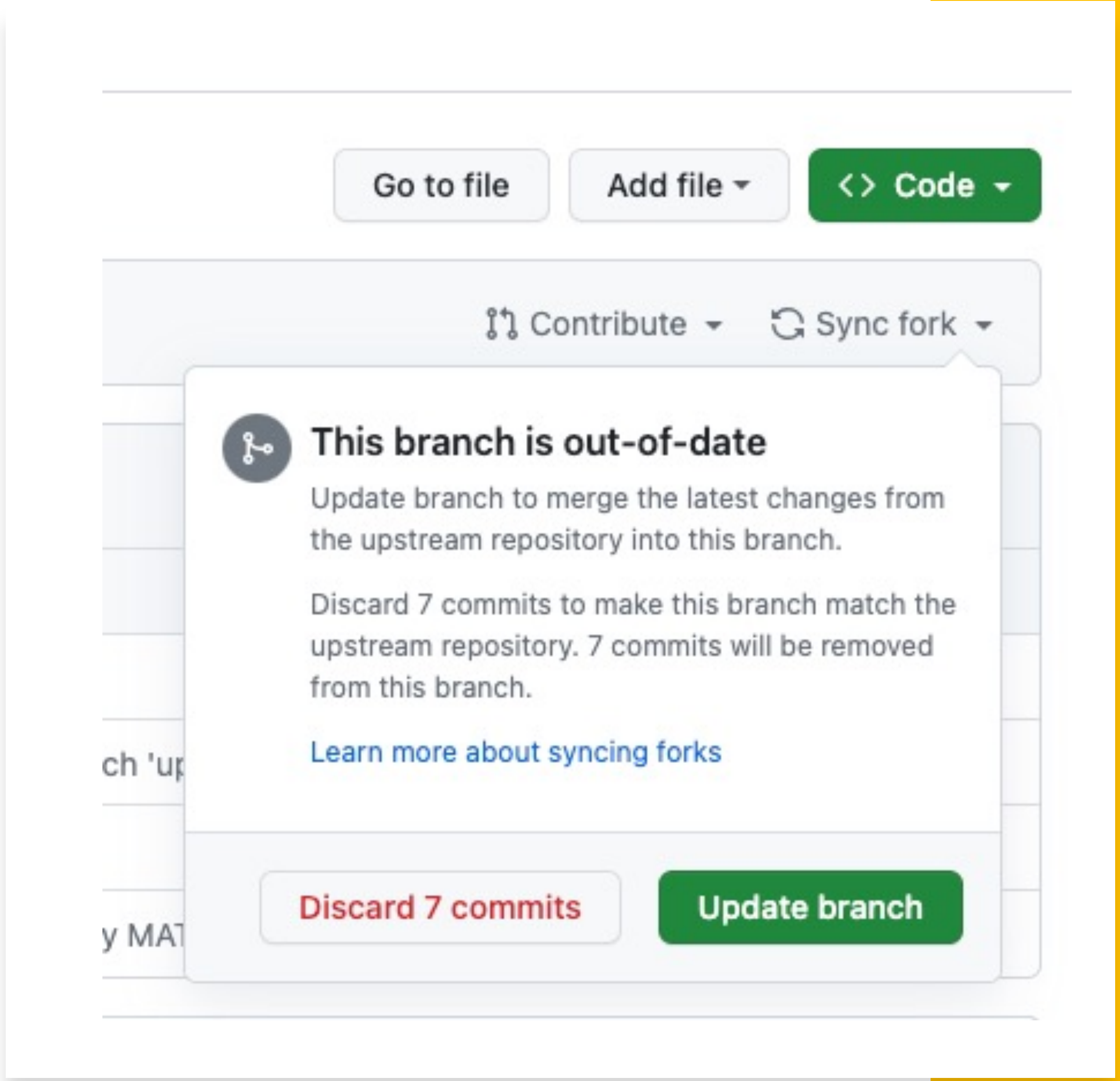
1. You can only fetch/pull *origin*, but not *upstream*

3. Any conflict will be resolved later when you make a pull request here

Synchronizing fork

- With option 2, you can go to your remote repository and click “Sync fork” to synchronize with the upstream repo.

- If you want to “harmonize”, **update branch**. Changes in the upstream repo will be merged to your forked repo.



This branch is out-of-date

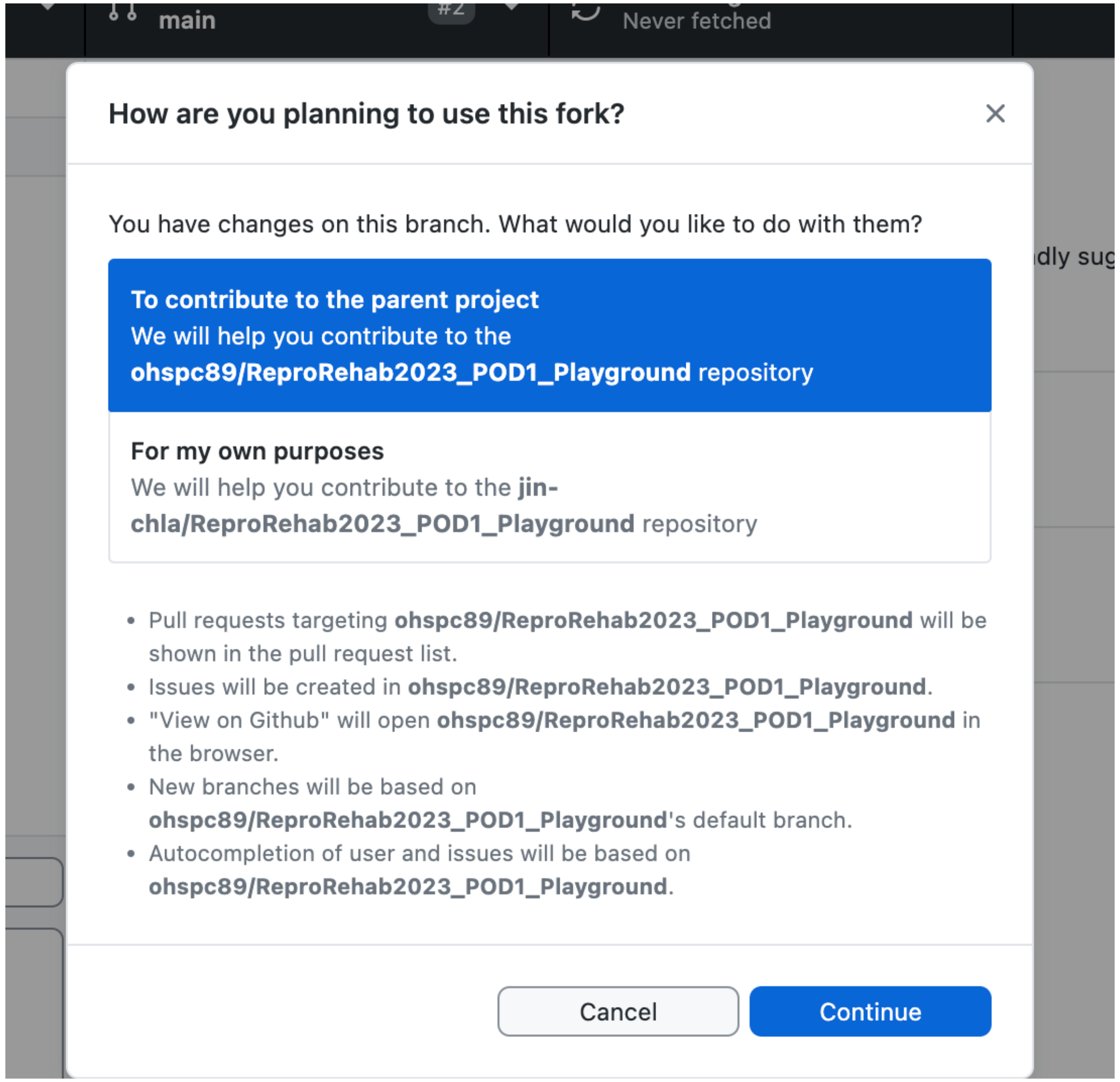
Update branch to merge the latest changes from the upstream repository into this branch.

Discard 7 commits to make this branch match the upstream repository. 7 commits will be removed from this branch.

Learn more about syncing forks

Cloning a repository

- Option 1 allows you to keep the direct communication with the *upstream* repo open. This lets you work in a more *collaborative* fashion.
- Option 2 lets you work on your repo independently and try different features without messing with the *upstream* repo.
- For today's practice, let's go with the first option, as we will continue using this repo.



How are you planning to use this fork?

You have changes on this branch. What would you like to do with them?

To contribute to the parent project
We will help you contribute to the **ohspc89/ReproRehab2023_POD1_Playground** repository

For my own purposes
We will help you contribute to the **jin-chla/ReproRehab2023_POD1_Playground** repository

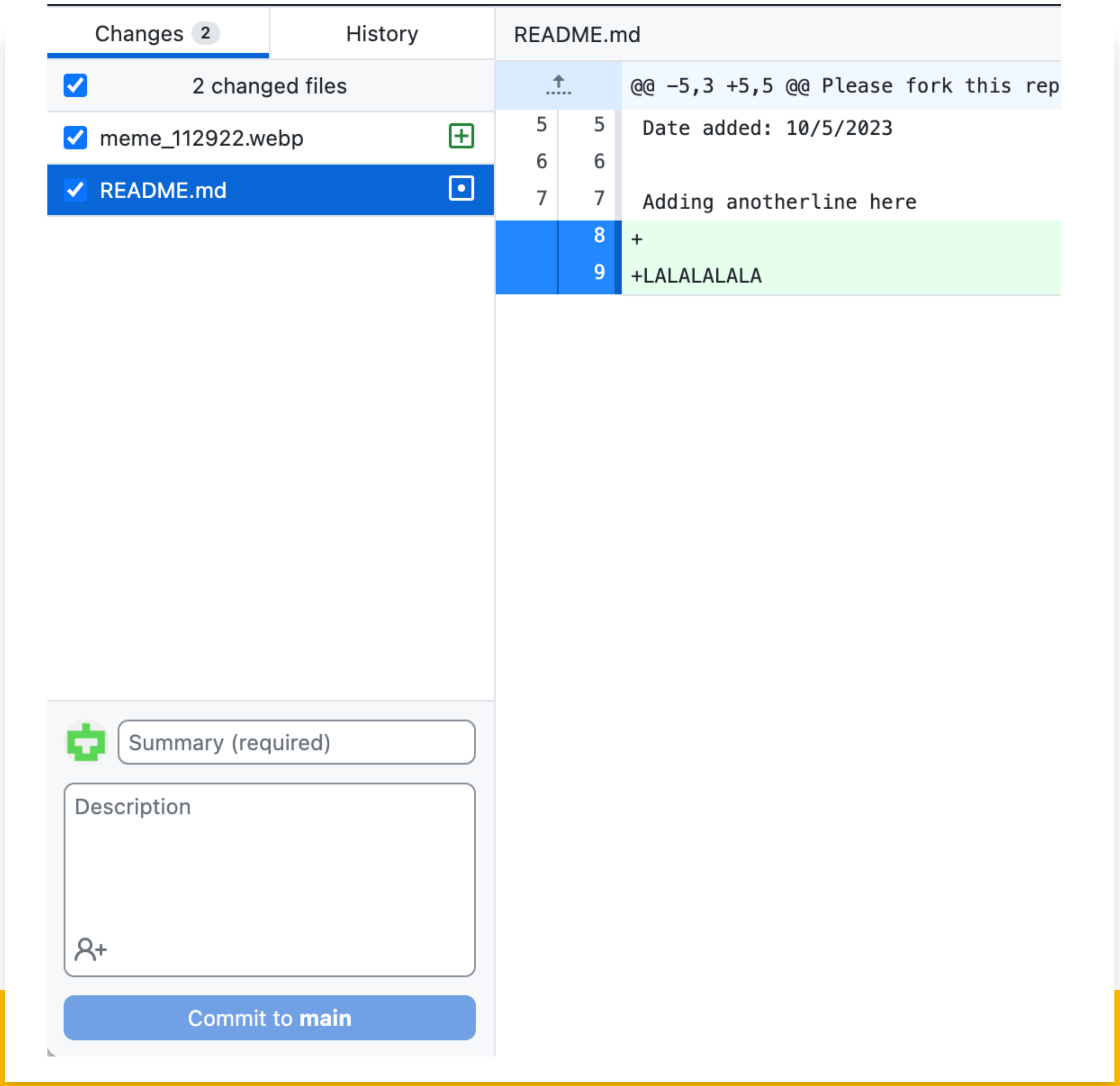
- Pull requests targeting **ohspc89/ReproRehab2023_POD1_Playground** will be shown in the pull request list.
- Issues will be created in **ohspc89/ReproRehab2023_POD1_Playground**.
- "View on Github" will open **ohspc89/ReproRehab2023_POD1_Playground** in the browser.
- New branches will be based on **ohspc89/ReproRehab2023_POD1_Playground**'s default branch.
- Autocompletion of user and issues will be based on **ohspc89/ReproRehab2023_POD1_Playground**.

Working from your local machine

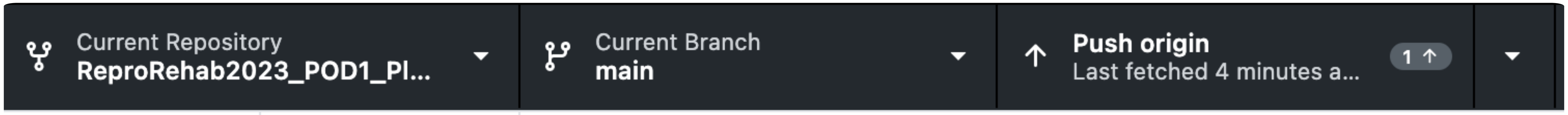
- For the sake of time, let’s do a simple task. Choose Repository > Show in File Explorer (or what is the exact command on windows?)
- Open **README.md** using a text editor and make any changes.
- Copy a *meme* (in .jpg, .png, .webp…) to the opened folder.

(Add) Commit and Push

- If you use Github Desktop, any changes you make to the folder is immediately added (*staged*) for commits.
- Put a short summary about the change you made. Try making it meaningful and memorable. Writing description is optional.
- Then press Commit to **main**



```
@@ -5,3 +5,5 @@ Please fork this rep
5 5   Date added: 10/5/2023
6 6
7 7   Adding anotherline here
  8  +
  9  +LALALALALA
```

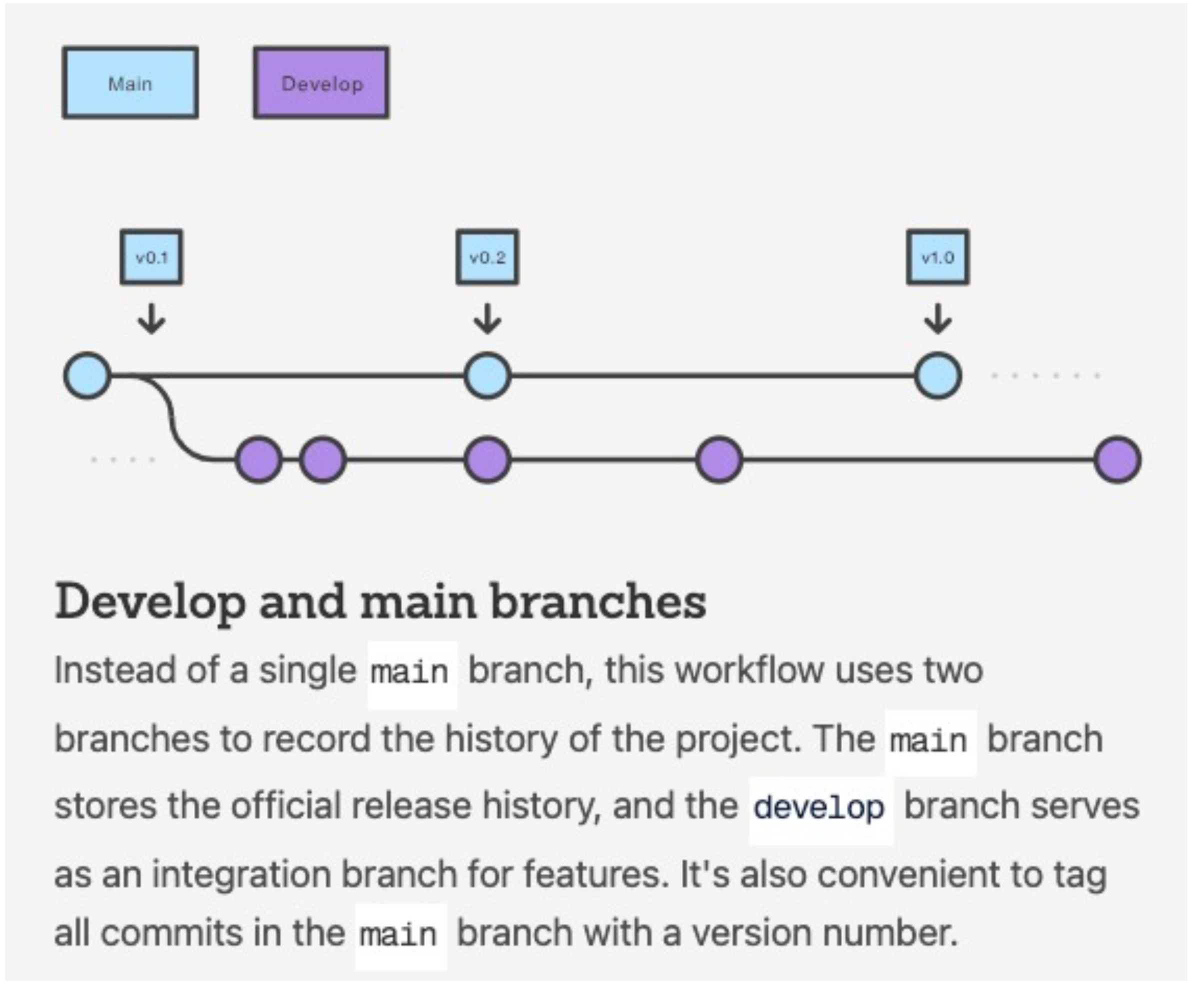


(Add) Commit and Push

- Then you can see the option **Push origin**. If you click that then these edits will be made to your FORKED repo.
- Please go to your online repo and check if all changes are made properly.

Multiple branches – why?

- The legacy workflow of software developers
- You don't want to release any unstable (or in-progress) version of your package.
- Therefore, you always store the clean and working version in the main branch and work *under the hood*.



Develop and main branches

Instead of a single `main` branch, this workflow uses two branches to record the history of the project. The `main` branch stores the official release history, and the `develop` branch serves as an integration branch for features. It's also convenient to tag all commits in the `main` branch with a version number.

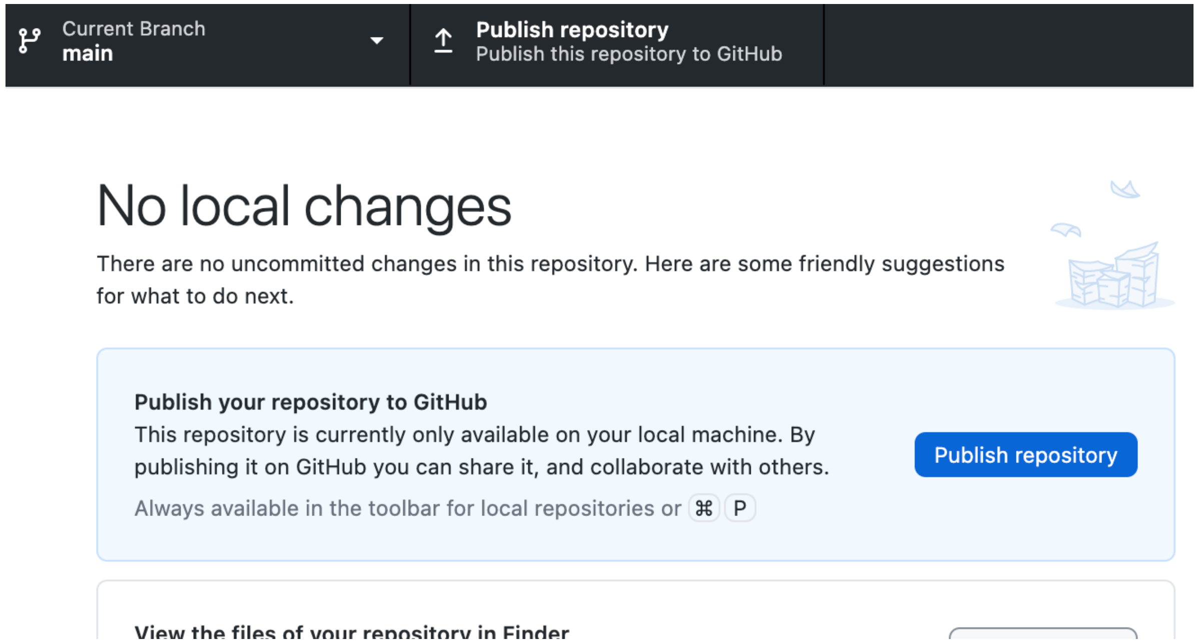
https://www.atlassian.com/git/tutorials/comparing-workflows/gitflow-workflow

Make a new repository

- File > New Repository

  Publish your repository to GitHub

- Yay! You just created your own repository



No local changes

There are no uncommitted changes in this repository. Here are some friendly suggestions for what to do next.

Publish your repository to GitHub

This repository is currently only available on your local machine. By publishing it on GitHub you can share it, and collaborate with others.

Always available in the toolbar for local repositories or ⌘ P

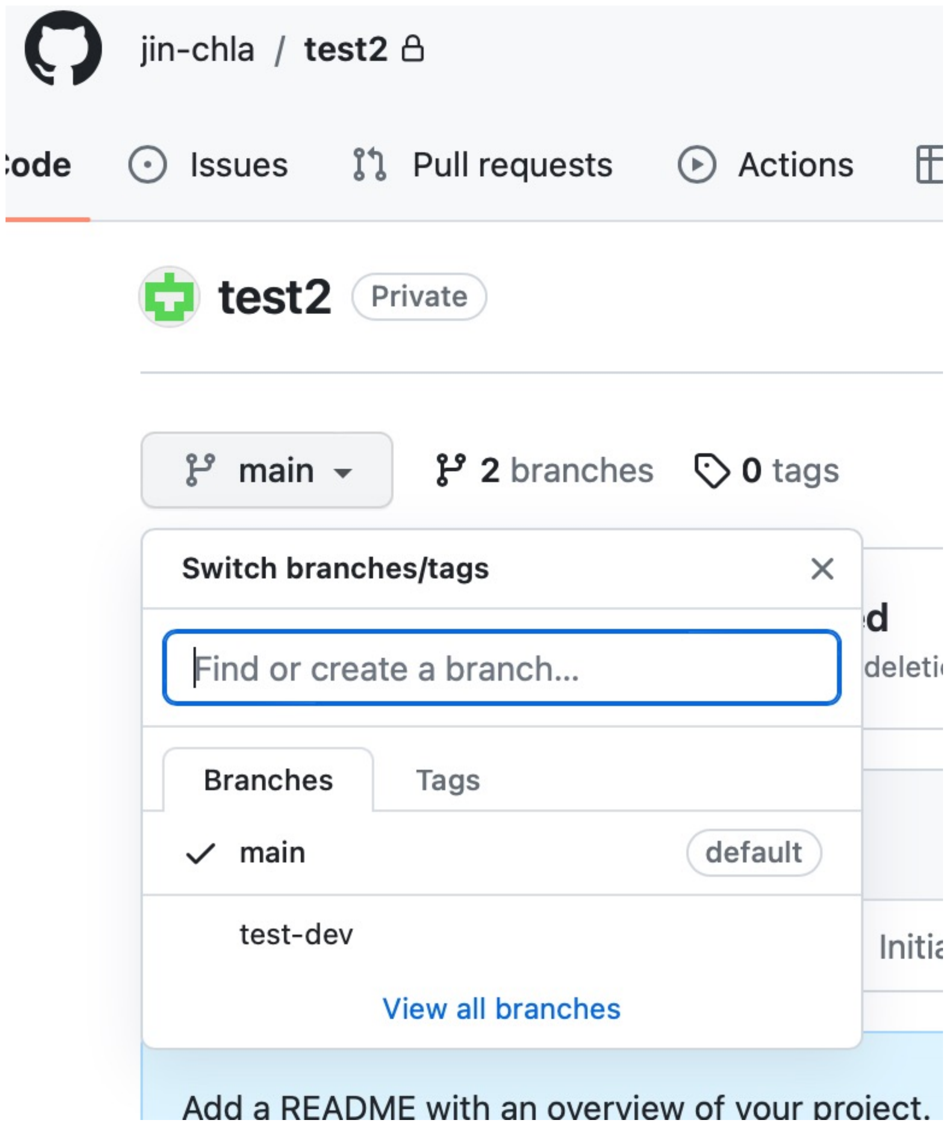
View the files of your repository in Finder

Make a new branch

- Branch > New Branch
- Name your new branch (e.g. test-dev)

  Publish your branch to GitHub

- If you go to the online repository, you can switch between branches.



jin-chla / test2

Code Issues Pull requests Actions

test2 Private

main 2 branches 0 tags

Switch branches/tags

Find or create a branch...

Branches Tags

✓ main default

test-dev

View all branches

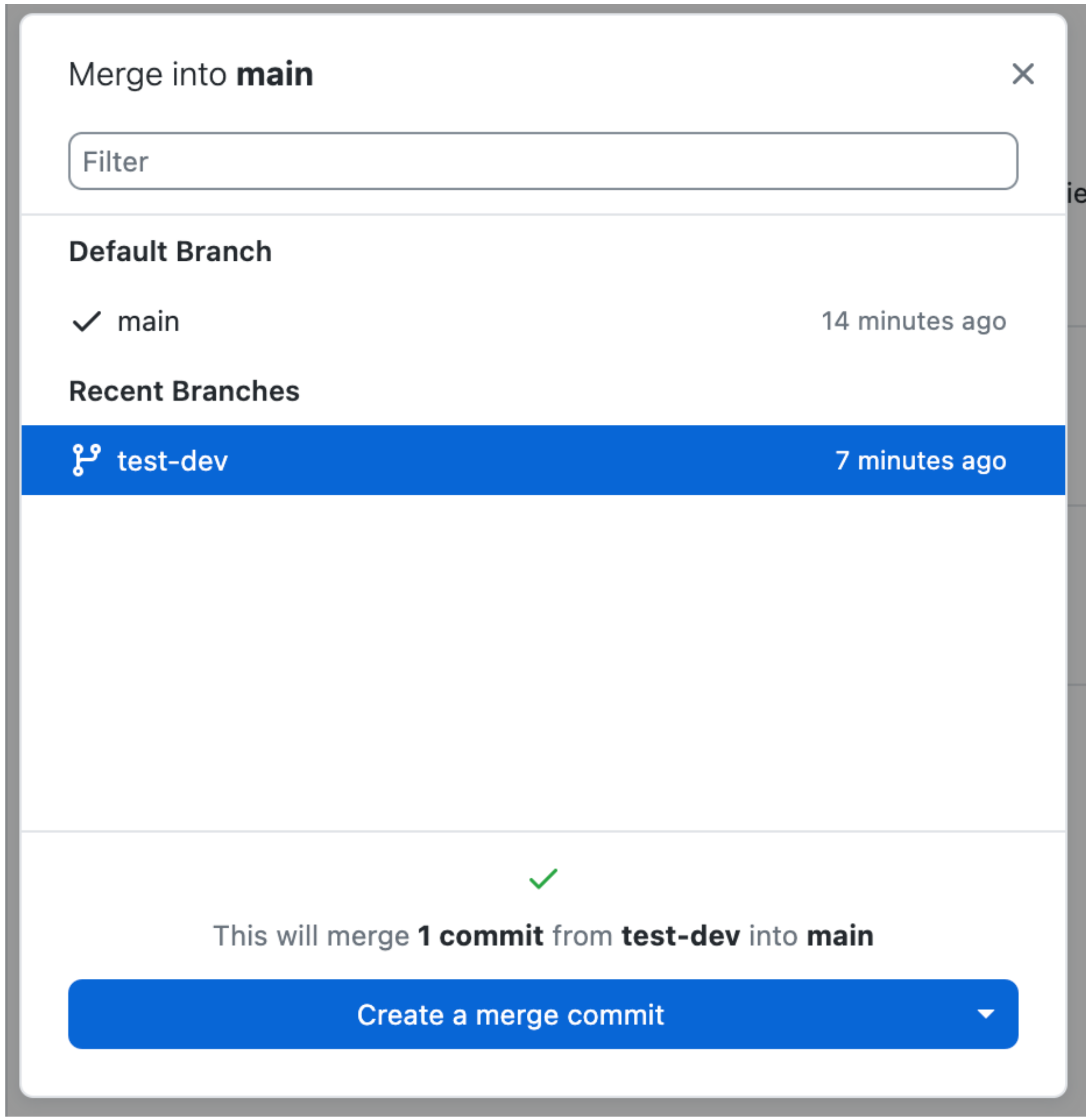
Add a README with an overview of your project.

Merge branches

- Set your current branch as the new branch other than *main*.
- Place any file (ex. a txt file) in the folder. You can open the folder by Repository > Show in File Explorer
- Commit and Push.
- Then switch the current branch to *main*.

Merge branches

- Branch > Merge into current branch
- Select the new branch and create a merge commit.
- Push origin
- Go to your online repository and check branches.


Merge into **main**

Filter

Default Branch

✓ main 14 minutes ago

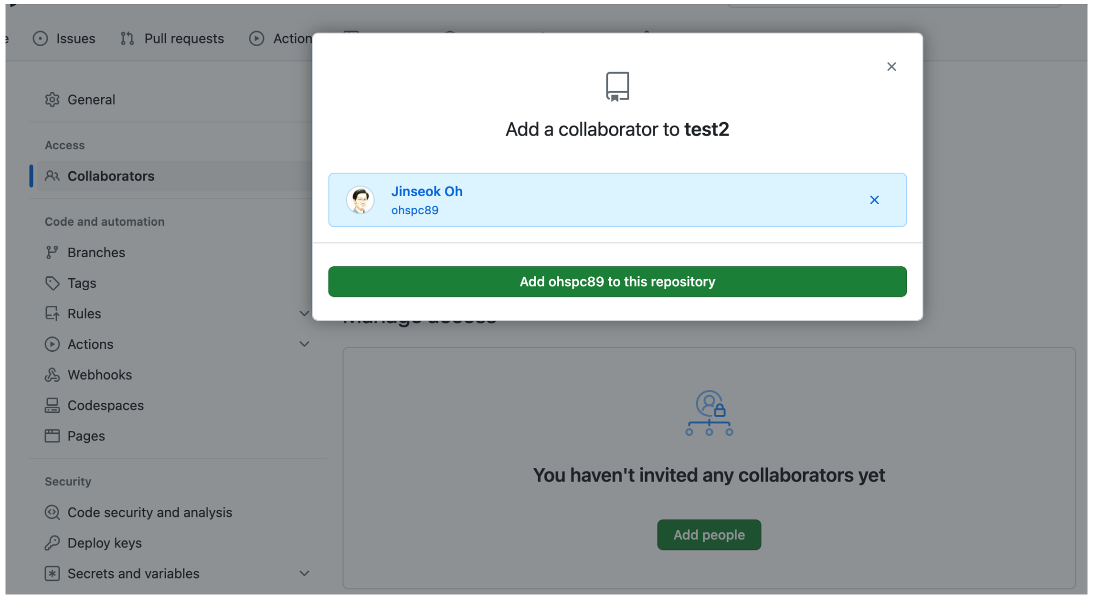
Recent Branches

test-dev 7 minutes ago

✓

This will merge **1 commit** from **test-dev** into **main**

Github: two more things!



Add a collaborator to **test2**

Jinseok Oh
ohspc89

Add ohspc89 to this repository

You haven't invited any collaborators yet

Add people

- Public vs. Private repository (read more here)
- Inviting collaborators (Settings > Collaborators > Add people)
- Collaborators can directly push to remote repository

Datalad

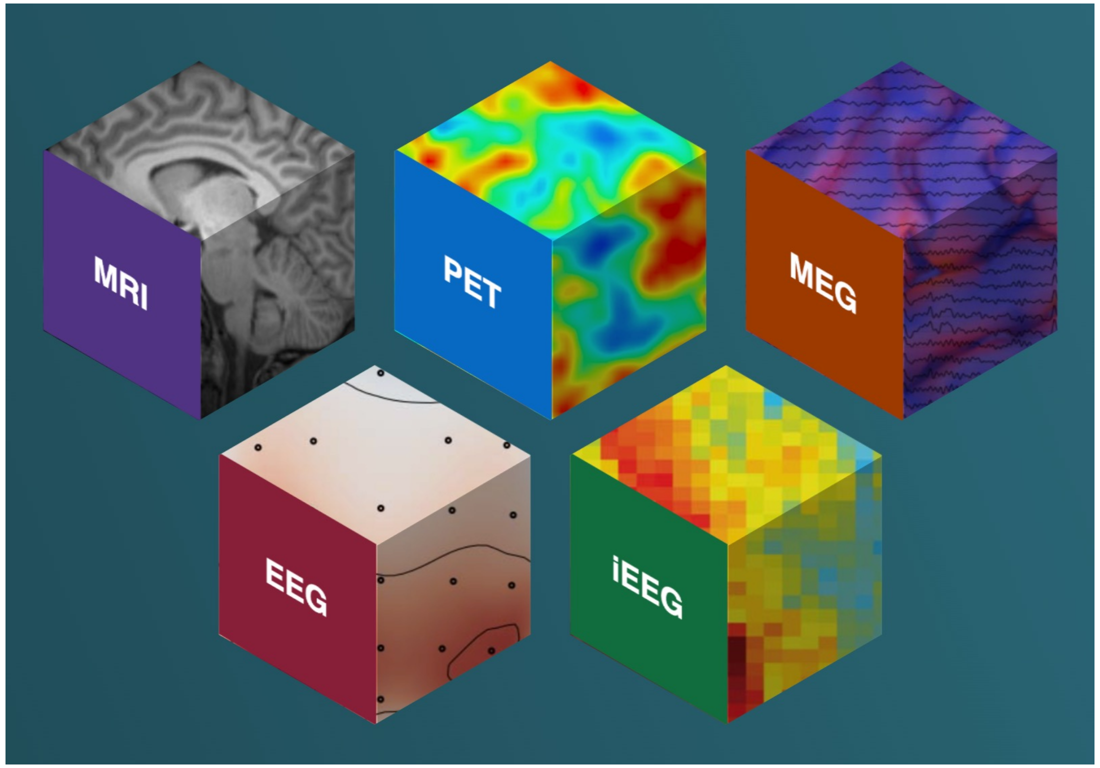
- “DataLad can *clone* a dataset to another location in a different computer.”
- Build on top of Git + α
- Allows version-controlling **data** and software alongside to code

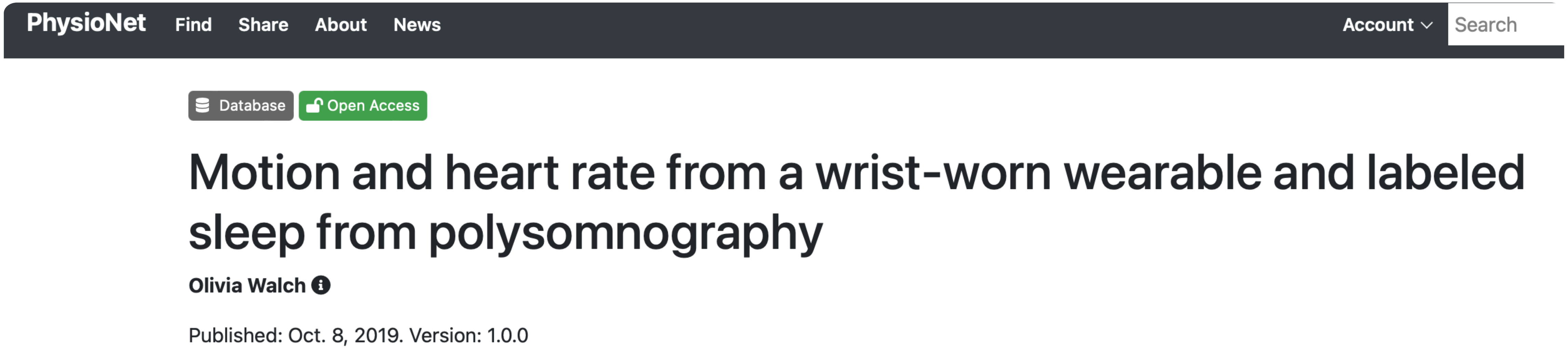
YouTube: What is DataLad?

OpenNeuro

- A platform for sharing *BIDS-compliant* neuroimaging data
  (BIDS: Brain Imaging Data Structure)

- Use datalad to download datasets from OpenNeuro





Motion and heart rate from a wrist-worn wearable and labeled sleep from polysomnography

Olivia Walch

Published: Oct. 8, 2019. Version: 1.0.0

Repositories of motion* data

*: Wearable sensor, EMG, Motion Capture…

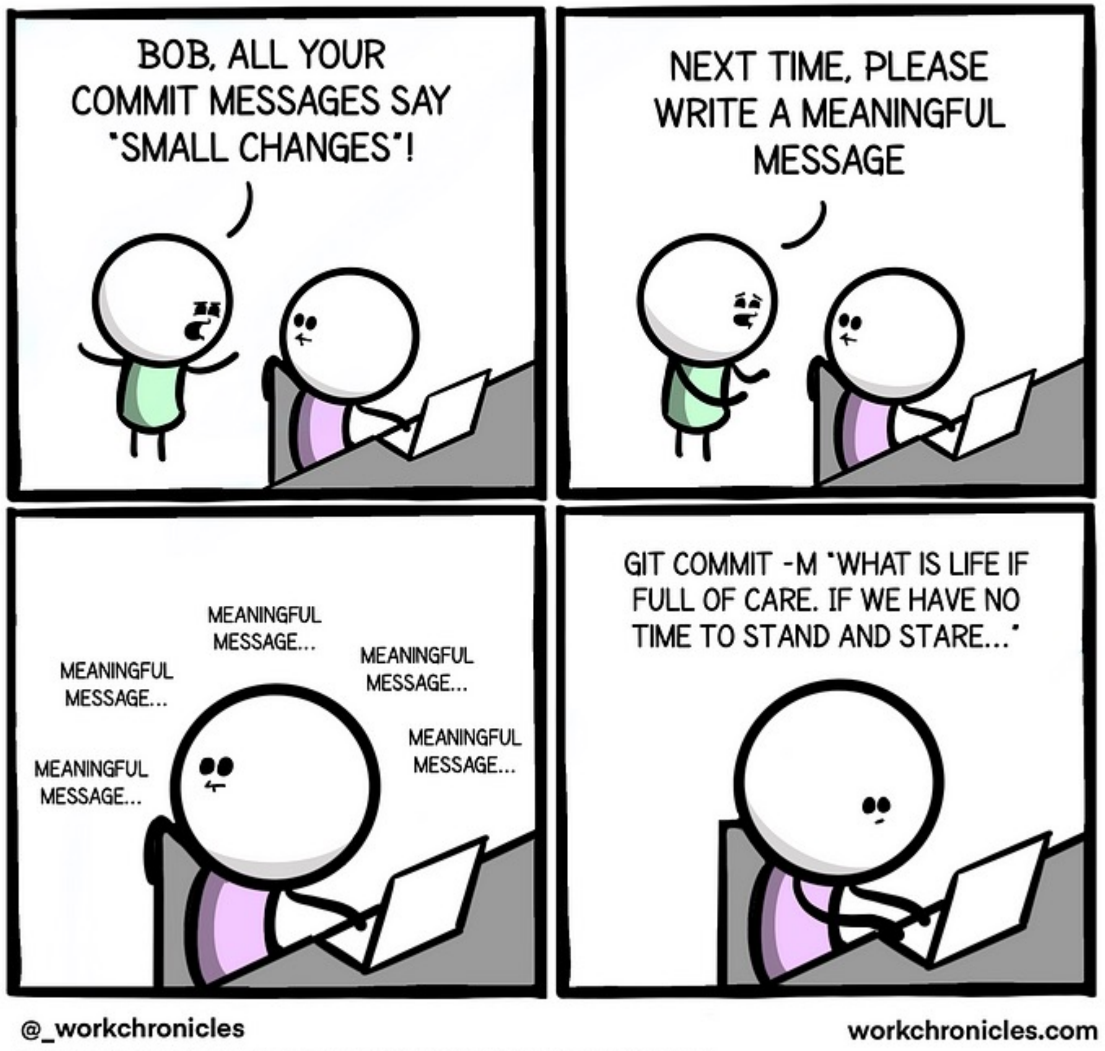
- PhysioNet (physionet.org)
- CMU Graphics Lab Motion Captue Database (http://mocap.cs.cmu.edu)
- UC Irvine Machine Learning Repository (https://archive.ics.uci.edu)

If you want to share your code / data

- MATLAB File Exchange (link)
- Your polished Github repository (read more)
- Try Code Ocean (read more) to share your environment (see example here)
- Check this NIH page to find repositories for sharing scientific data (link)

Summary

- So… why bother learning Git/Github again?
  - Version control: keep track of **who** did **what** on **which** file **when**
  - Even if you never make use of this feature, you still can **clone** different repositories that have processing pipeline codes you may find useful
  - You may at one day wish to suggest a feature to an existing code or even prepare one by yourself and share it with others
  - Other version control software (ex. datalad) are based on Git – useful to know the basics


BOB, ALL YOUR COMMIT MESSAGES SAY "SMALL CHANGES"!
NEXT TIME, PLEASE WRITE A MEANINGFUL MESSAGE
MEANINGFUL MESSAGE...
MEANINGFUL MESSAGE...
MEANINGFUL MESSAGE...
MEANINGFUL MESSAGE...
MEANINGFUL MESSAGE...
GIT COMMIT -M "WHAT IS LIFE IF FULL OF CARE. IF WE HAVE NO TIME TO STAND AND STARE..."
@_workchronicles
workchronicles.com