

Introduction to GitHub

Welcome! This document serves as a general introduction to making an account on GitHub, joining the Poulin Lab organization and adding code.

Table of Contents

Creating a GitHub Account	2
GitHub Education	2
Joining the Poulin Lab Organization	4
Storing & Sharing Code	6
Creating Repositories	6
Adding Code	8
Sharing Code with Members Outside the Lab	9
Modifying Code and Development.....	12
Creating Branches Within Repositories	12
Committing Changes	15
Pull Requests & Merging Branches	18

While this document focuses primarily on uploading and sharing code, the second half does touch on the basics of development. For further reading on using GitHub as a workspace, resources are available below:

<https://docs.github.com/fr/get-started/start-your-journey/hello-world>

<https://skills.github.com/>

<https://docs.github.com/en/get-started/start-your-journey/about-github-and-git>

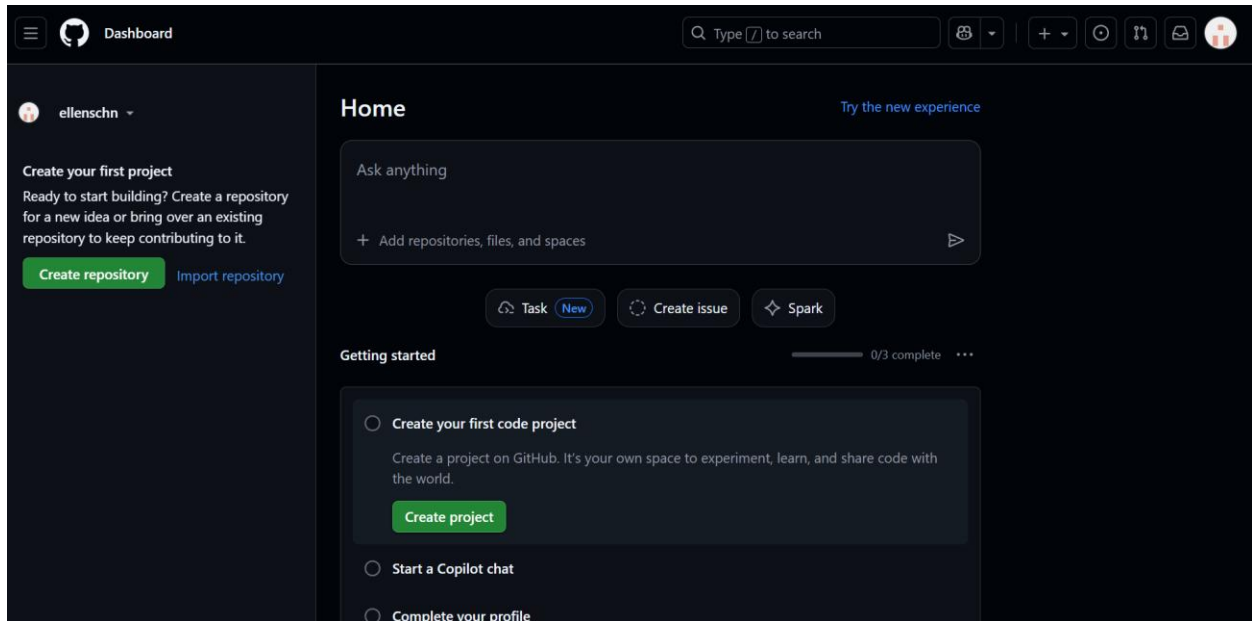
Creating a GitHub Account

To add code and join the lab organization, you should create a GitHub account. Ideally, each member of the lab should have their own GitHub account.

To create a GitHub personal account, navigate to the main GitHub page:

<https://github.com/>. Enter your email (ideally McGill email) and create a password.

Once made, log into GitHub. The main page of the website should look like this:

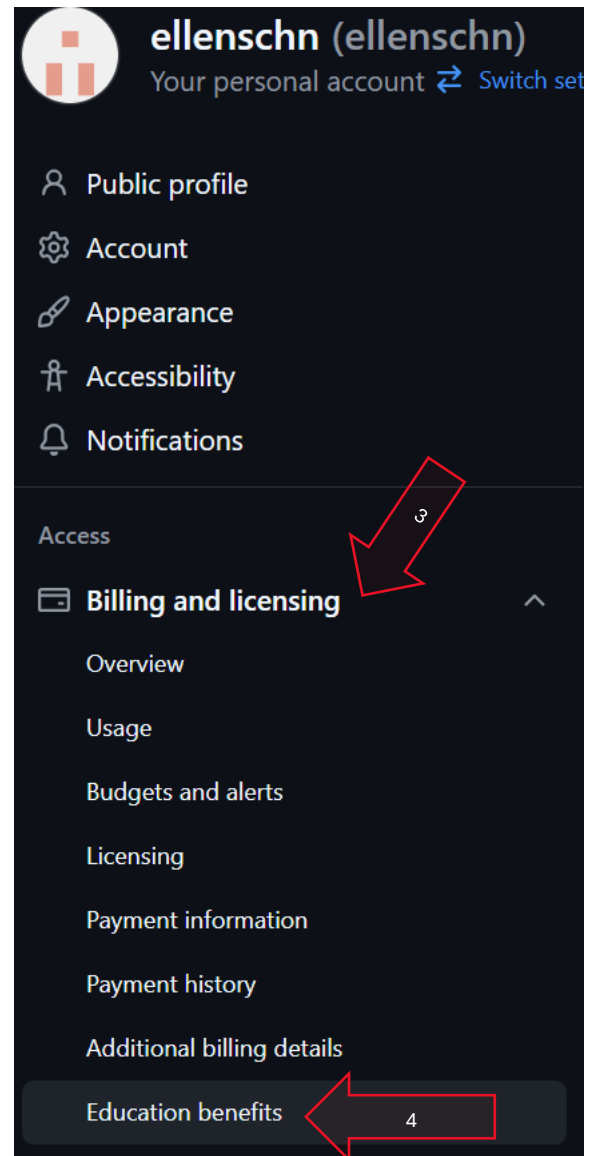
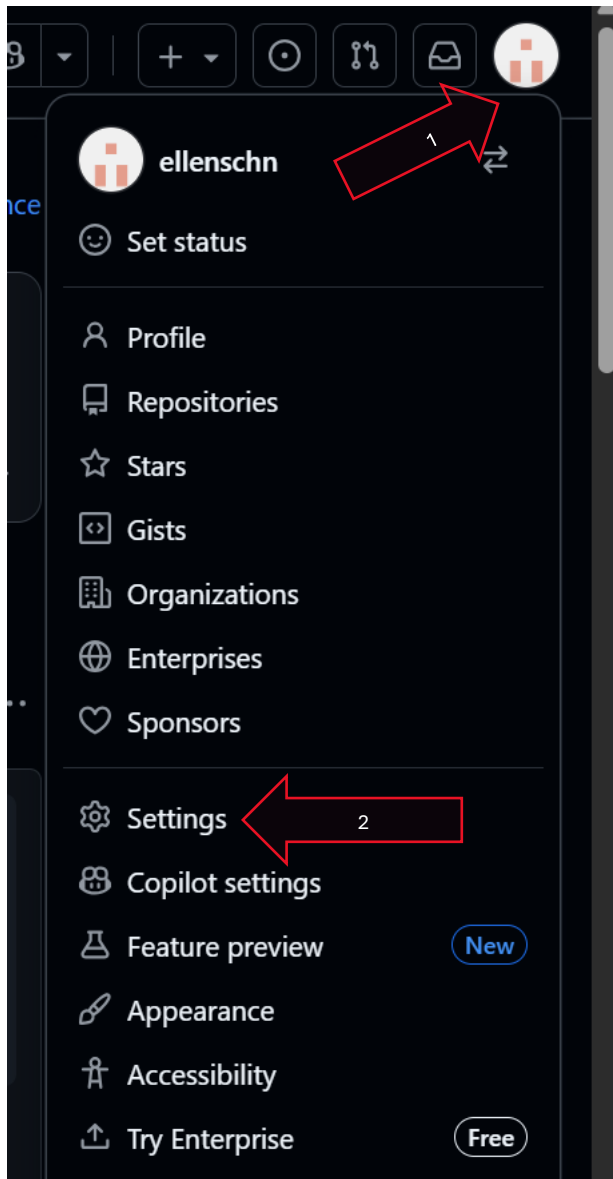


GitHub Education

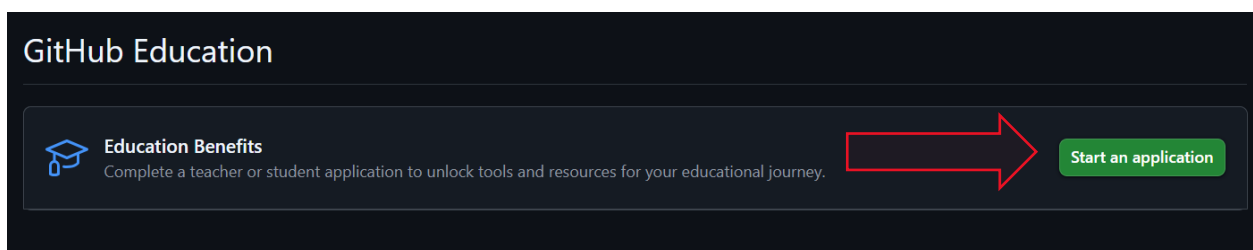
Note: this part isn't necessary, just a nice feature of GitHub.

As a McGill student, you can sign up for GitHub Education for free! You can read on the benefits here: <https://github.com/education>

To sign up for GitHub Education, navigate to your settings. Click Billing and licensing -> Education benefits.



Start an application and follow the prompts. You will need to share your location and upload a proof of enrolment (can be in the form of Student ID, transcripts, enrolment letter, etc.)

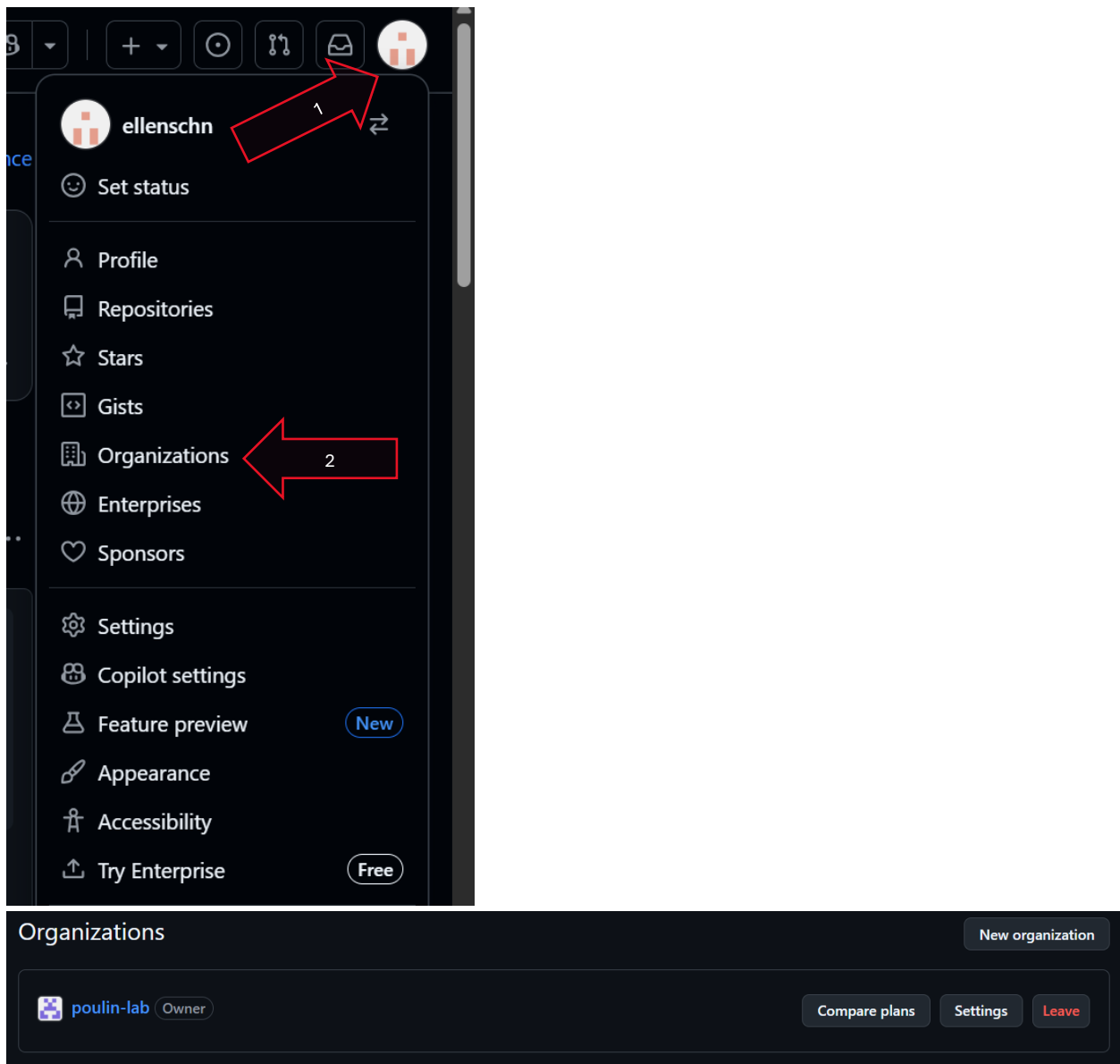


Once your application is approved, you can access all the Education benefits!

Joining the Poulin Lab Organization

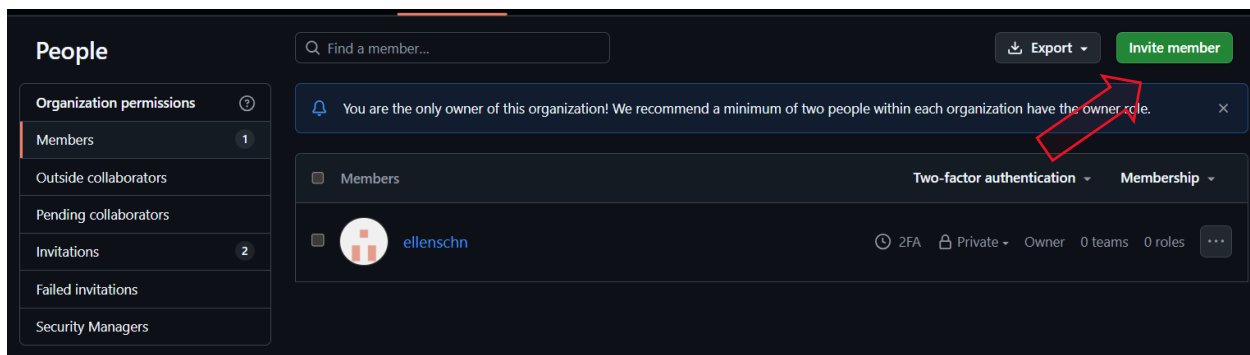
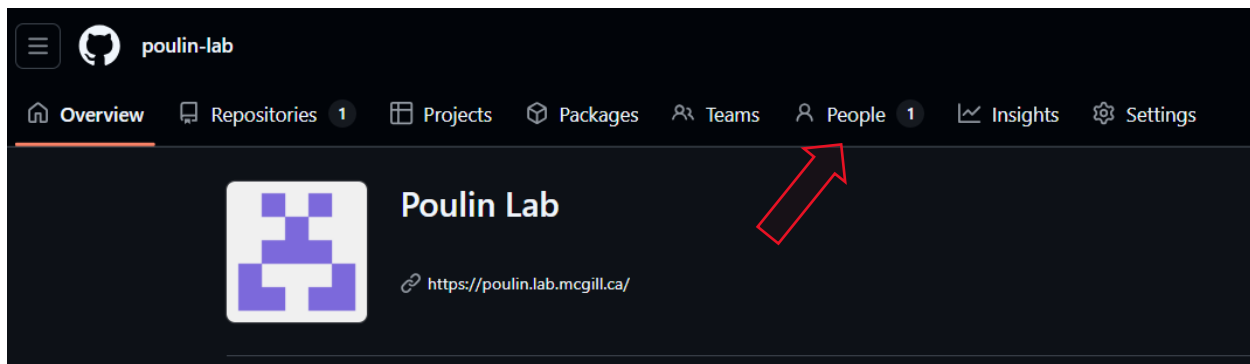
An organization is a shared space between all lab members to add, modify and share their code. To join the Poulin Lab Organization, you must be invited by an organization admin. Admins include JF Poulin and Duncan Spencer. Ensure that you are receiving an invitation to the email you signed up to GitHub with.

To access the organization in GitHub, navigate to the Organizations section, and then click Poulin Lab.

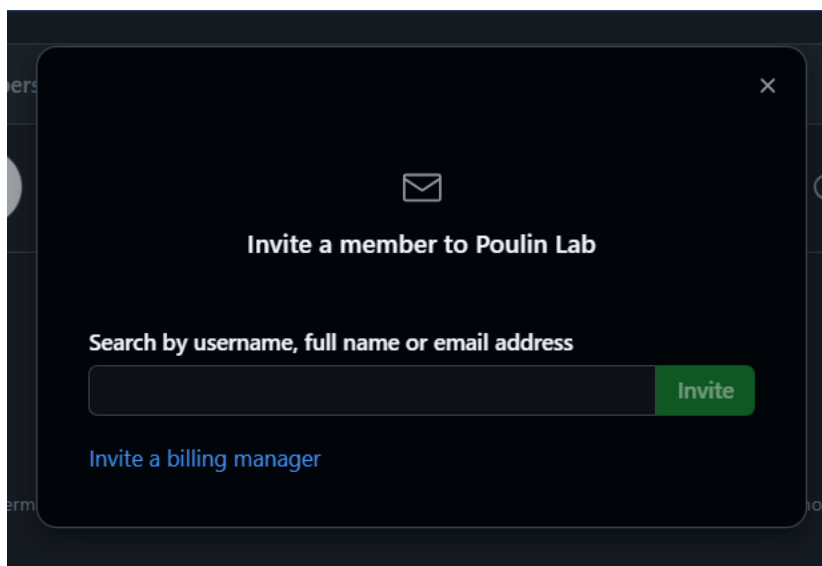


For Admins: How to Invite Members

Start on the main page of the Poulin Lab organization and navigate to the “People” tab.



Simply type in their email address and invite!



Note that it will ask you to give the person Member or Admin benefits. The lab organization is configured so that Members can add code, create and delete repos but **cannot invite other collaborators**. An administrator on the other hand, has full access to everything. You can additionally give “admin access” to a person for a particular repository.

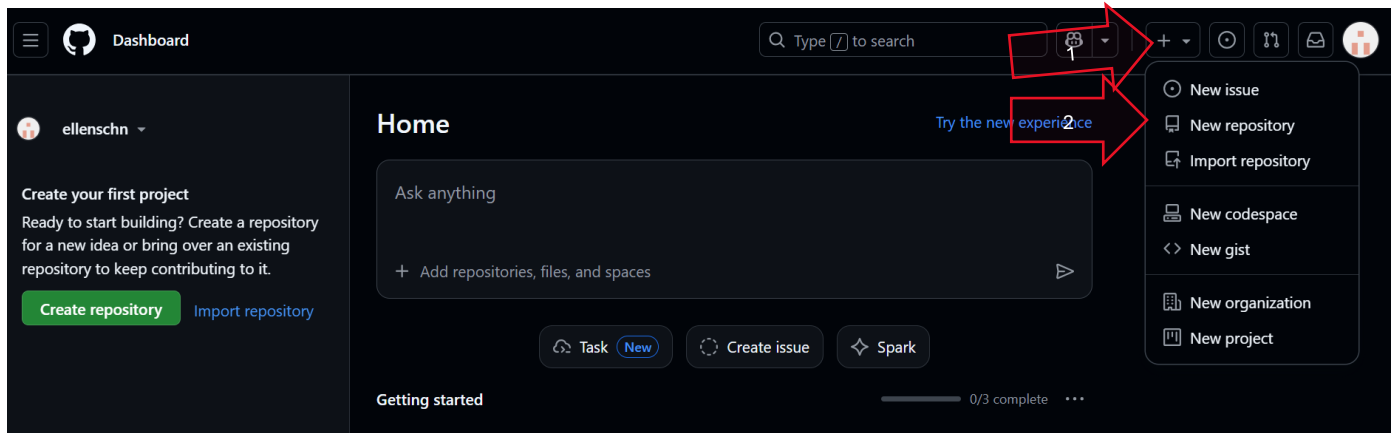
Storing & Sharing Code

All code created and shared should be stored under the lab organization for ease of access and trackability.

Creating Repositories

A repository (or repo) can be thought of like a folder. It will contain related items to a particular experiment or project. Each repository should have a README file to detail information about that particular experiment.

To create a repository, navigate to the “+” button on the right hand side of the screen.



Name the repository accordingly. Ensure that the owner falls under “Poulin-Lab” and not your account so that everyone has access to the repository. Add a description if you like. Visibility should ideally be public, and README should be checked.

Create a new repository

Repositories contain a project's files and version history. Have a project elsewhere? [Import a repository](#).
Required fields are marked with an asterisk (*).

1

General

Owner *

poulin-lab

Repository name *

Great repository names are short and memorable. How about **automatic-octo-spork**?

Description

0 / 350 characters

2

Configuration

Choose visibility *

Choose who can see and commit to this repository

Public

Add README

READMEs can be used as longer descriptions. [About READMEs](#)

On ☒

Add .gitignore

.gitignore tells git which files not to track. [About ignoring files](#)

No .gitignore

Add license

Licenses explain how others can use your code. [About licenses](#)

No license

Create repository

poulin-lab / .github

Type / to search

<> Code

Issues

Pull requests

Actions

Projects

Wiki

Security

Insights

Settings

.github

Public

Edit Pins

Watch 0

Fork 0

Star 0

main 1 Branch 0 Tags

Go to file

Add file

<> Code

ellenschn Create README.md 1b68075 · 1 hour ago 2 Commits

profile Create README.md 1 hour ago

README.md Initial commit 1 hour ago

README

.github

All information needed to get started w/ GitHub, in regards to making repositories, adding code, pulling & changing it, along with sharing.

poulin-lab/.github is a special repository.
The profile/README.md will appear on the organization's profile.

Edit README Visit profile

About

All information needed to get started w/ GitHub, in regards to making repositories, adding code, pulling & changing it, along with sharing.

Readme

Activity

Custom properties

0 stars

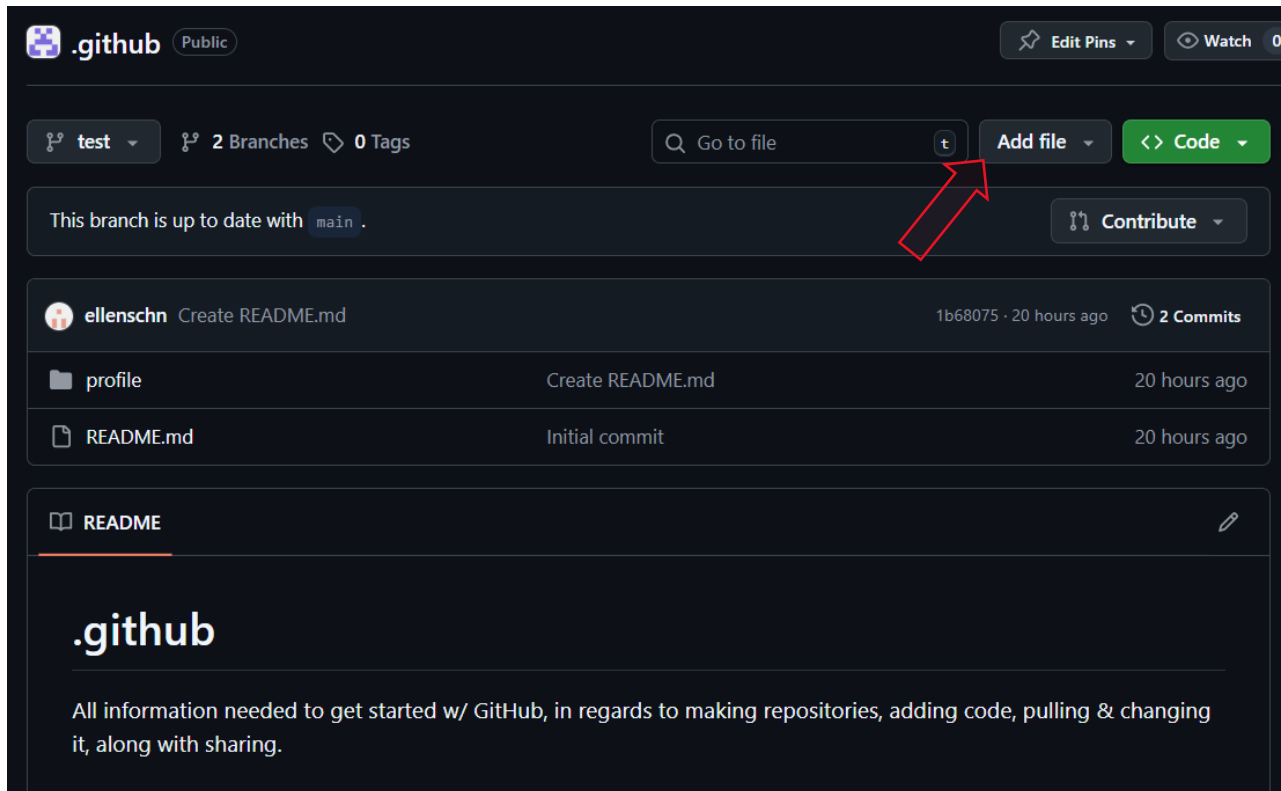
0 watching

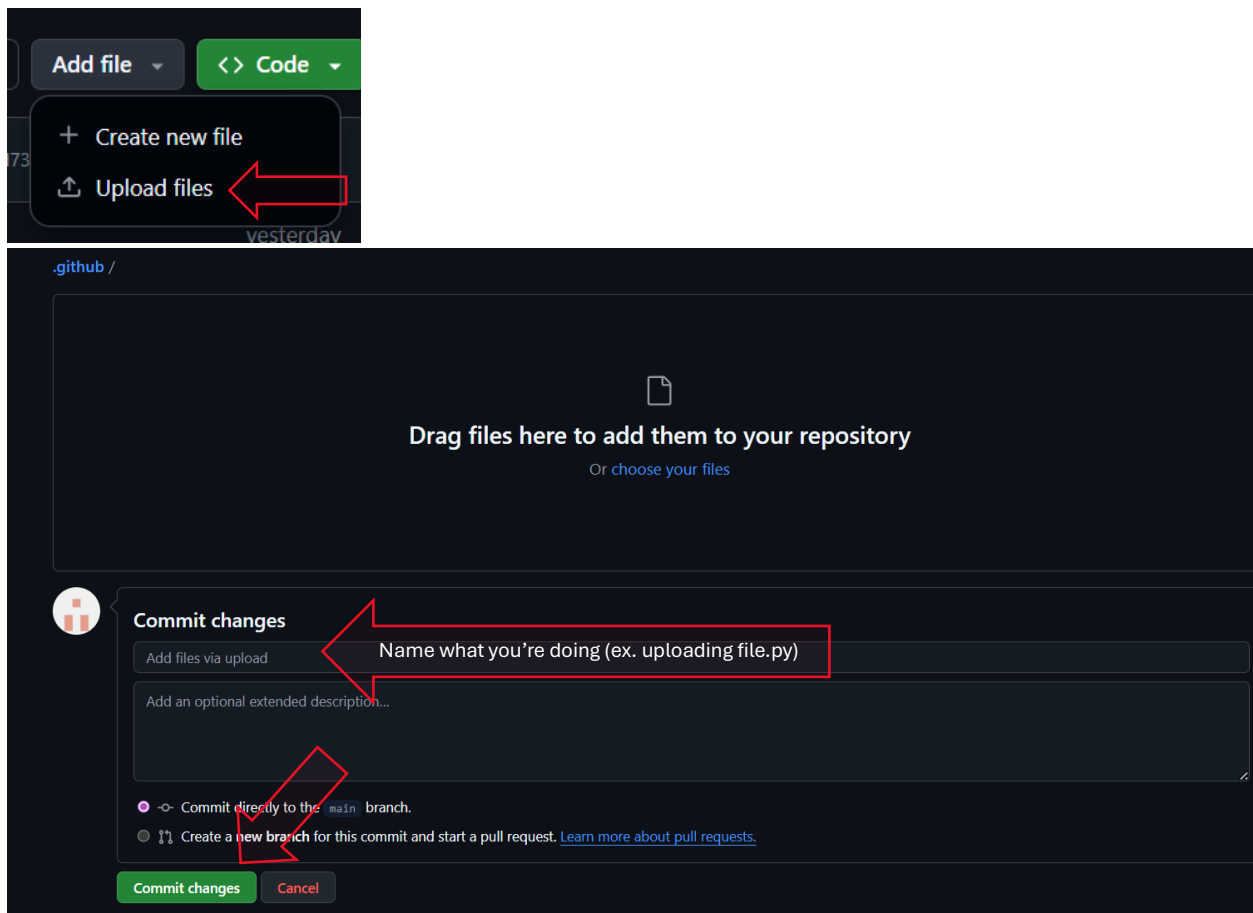
0 forks

Report repository

Adding Code

You can add code to any branch in a repository. For most cases in this lab, we'll keep the final or official version in the main branch.



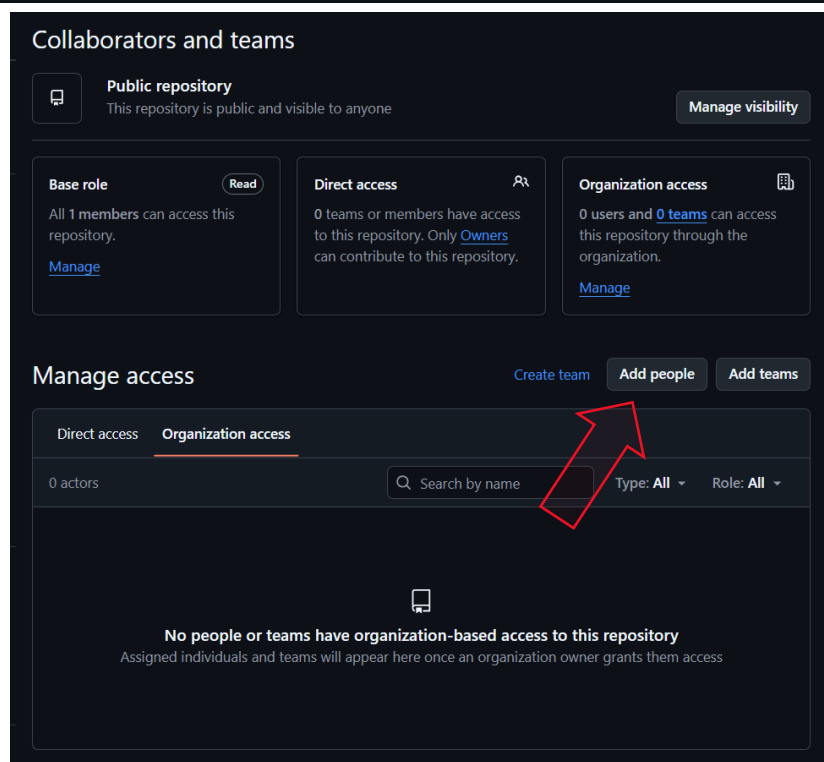
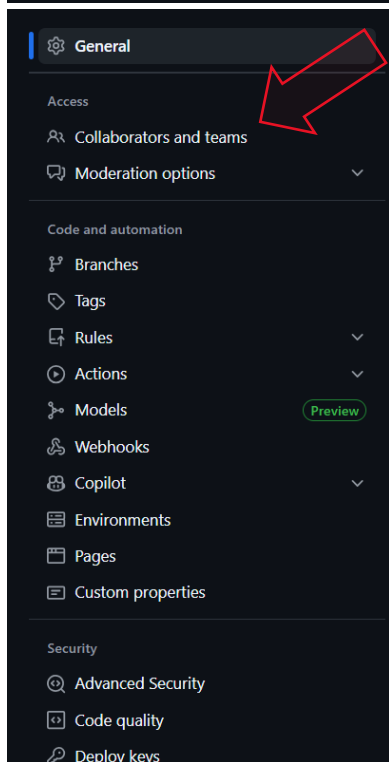
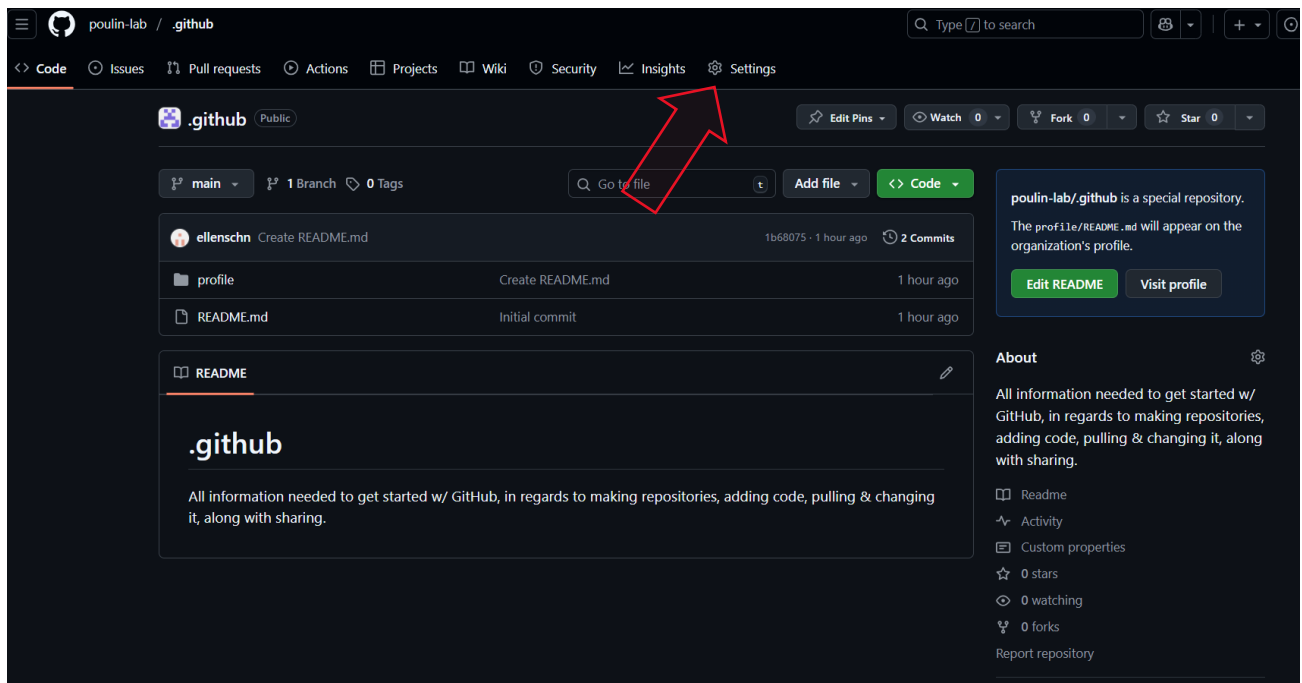


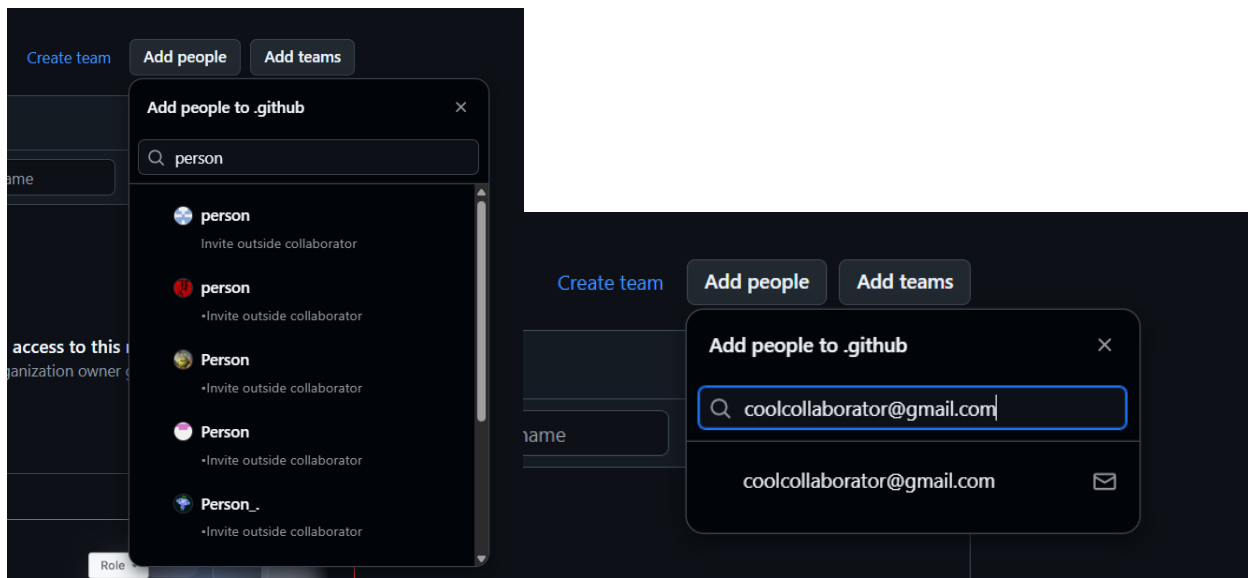
Note: you can upload just about anything (doesn't have to be code)!

Sharing Code with Members Outside the Lab

Note: this is sharing for a particular repository, if you want someone to have full access to the organization, an administrator should invite them as a member of the lab.

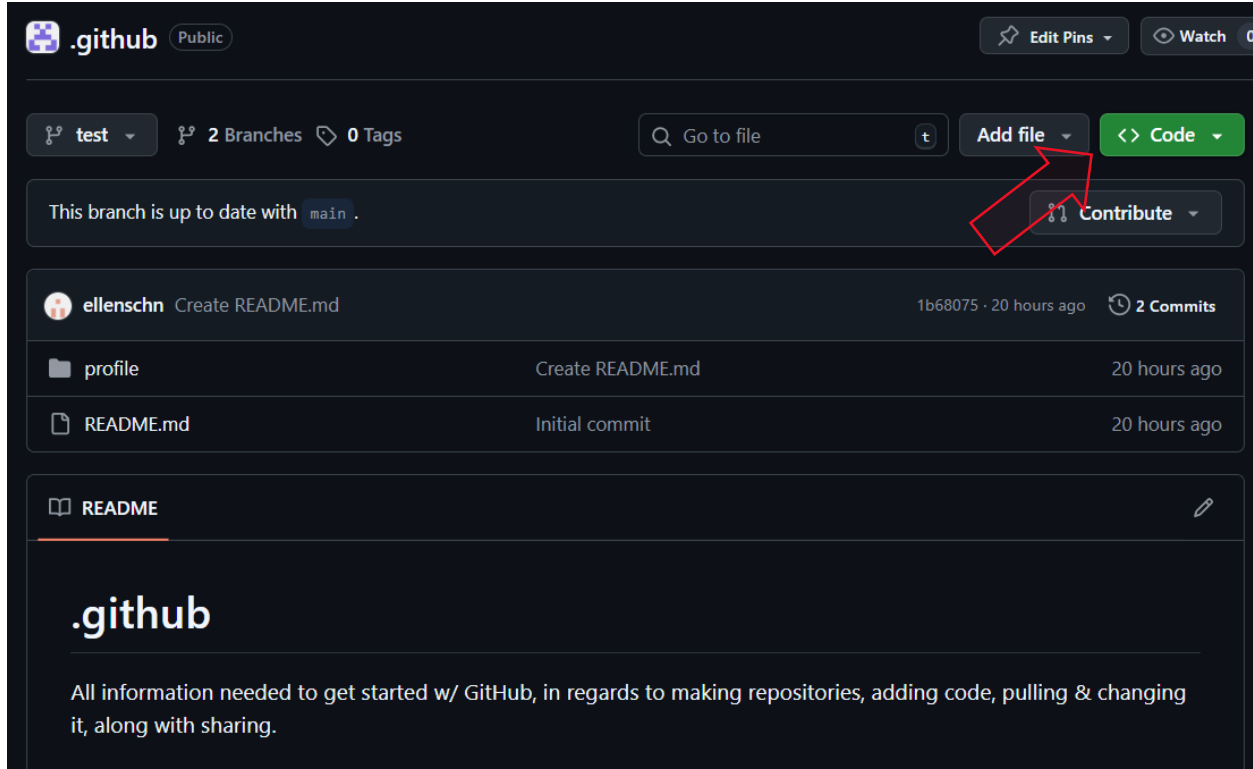
Sharing code can be done to someone's GitHub account or to an email address. To share a repository directly to a person, go to the repository's Settings → Manage Access → Invite collaborators.

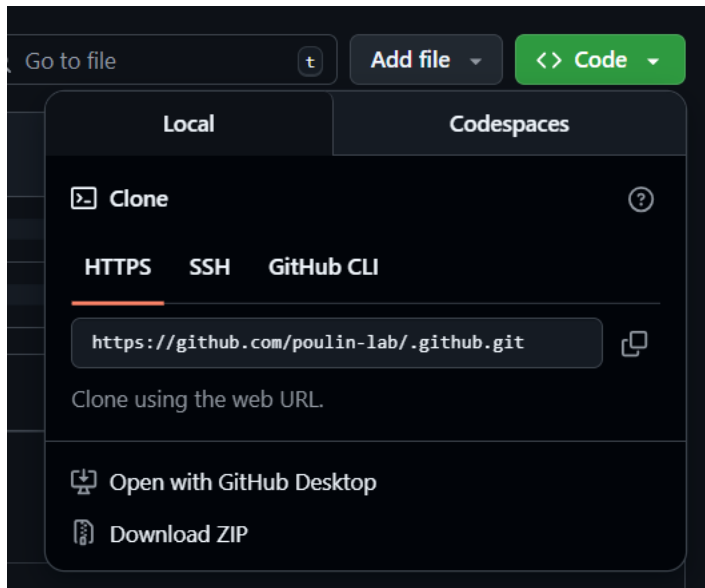




You can search to find people who already have a GitHub account or invite people directly through email. You can additionally modify someone's access under the "Manage Access" box.

If the repository is public, an alternative is through a URL link. Note that this only allows others to see the code, they can not modify or add anything.





Simply copy the URL and anyone will be able to view the repository!

Modifying Code and Development

Code uploaded to GitHub can be modified and changed as needed. The best way to approach this is by creating branches off of main repositories, committing new changes, and sending pull requests to other lab members.

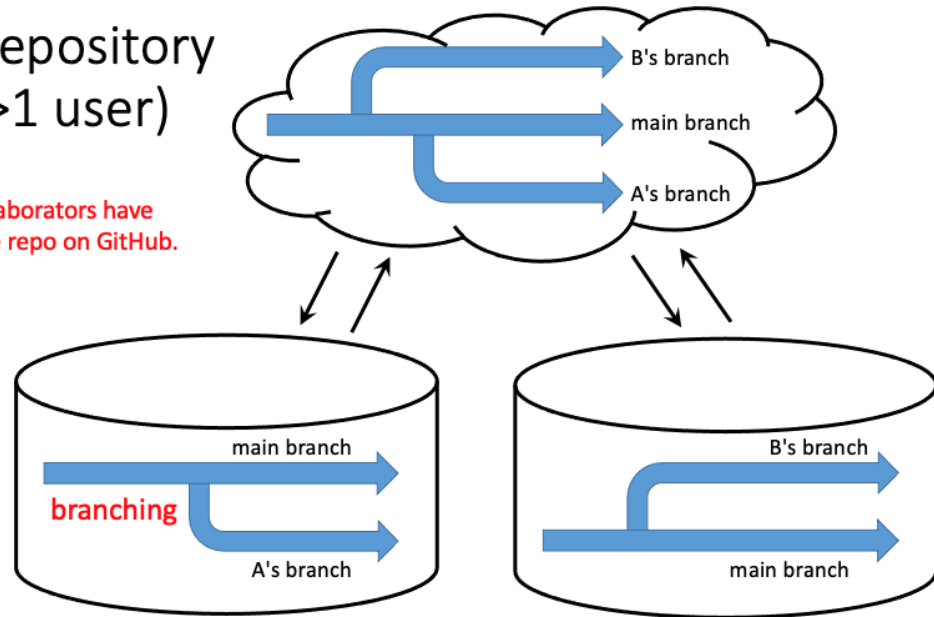
Creating Branches Within Repositories

A branch is a separate version of the code within a repository. It acts like a workspace where a user can make changes without affecting the main version. Each branch can have its own commits and updates, letting you experiment safely.

By default, all repositories have a main branch where the source code should be stored. There is no limit on how many branches a repository can have.

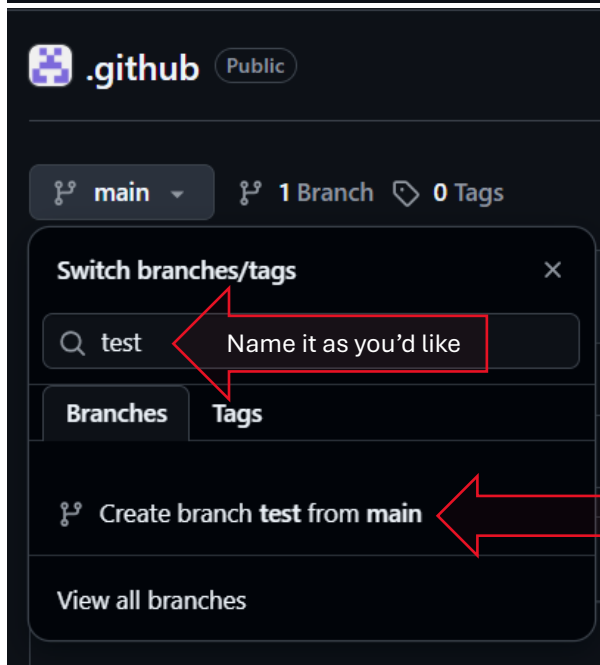
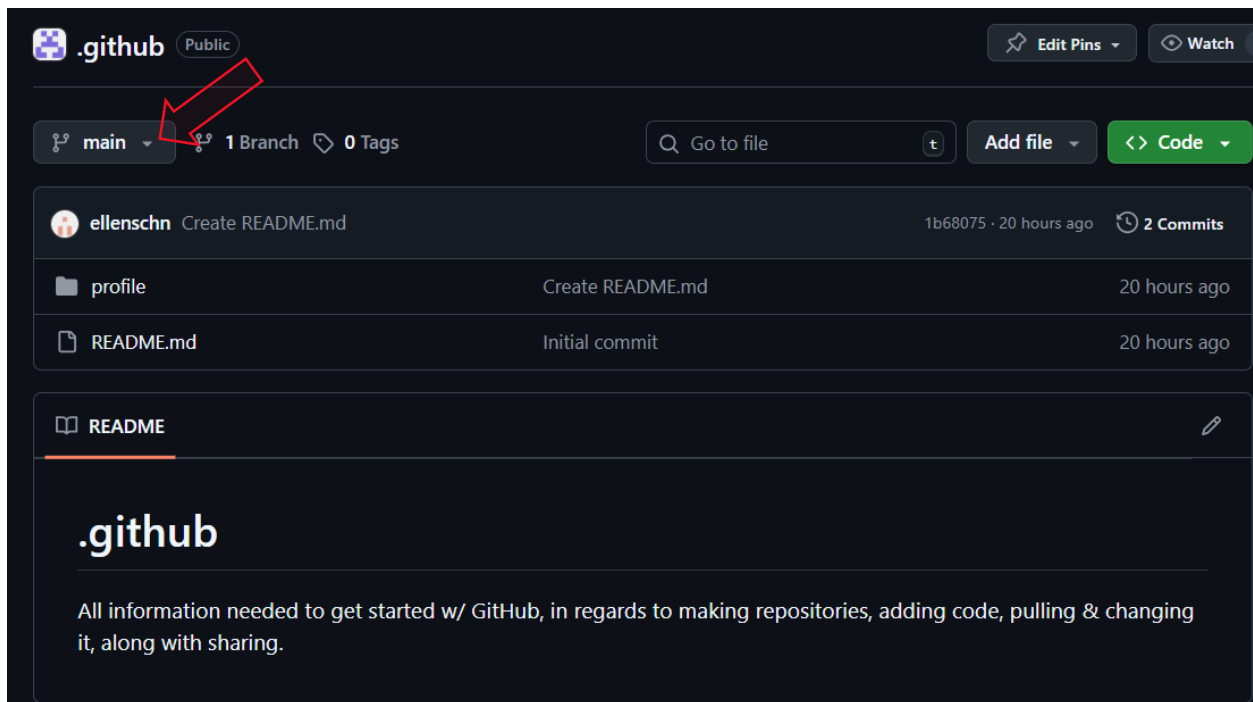
Shared repository model (>1 user)

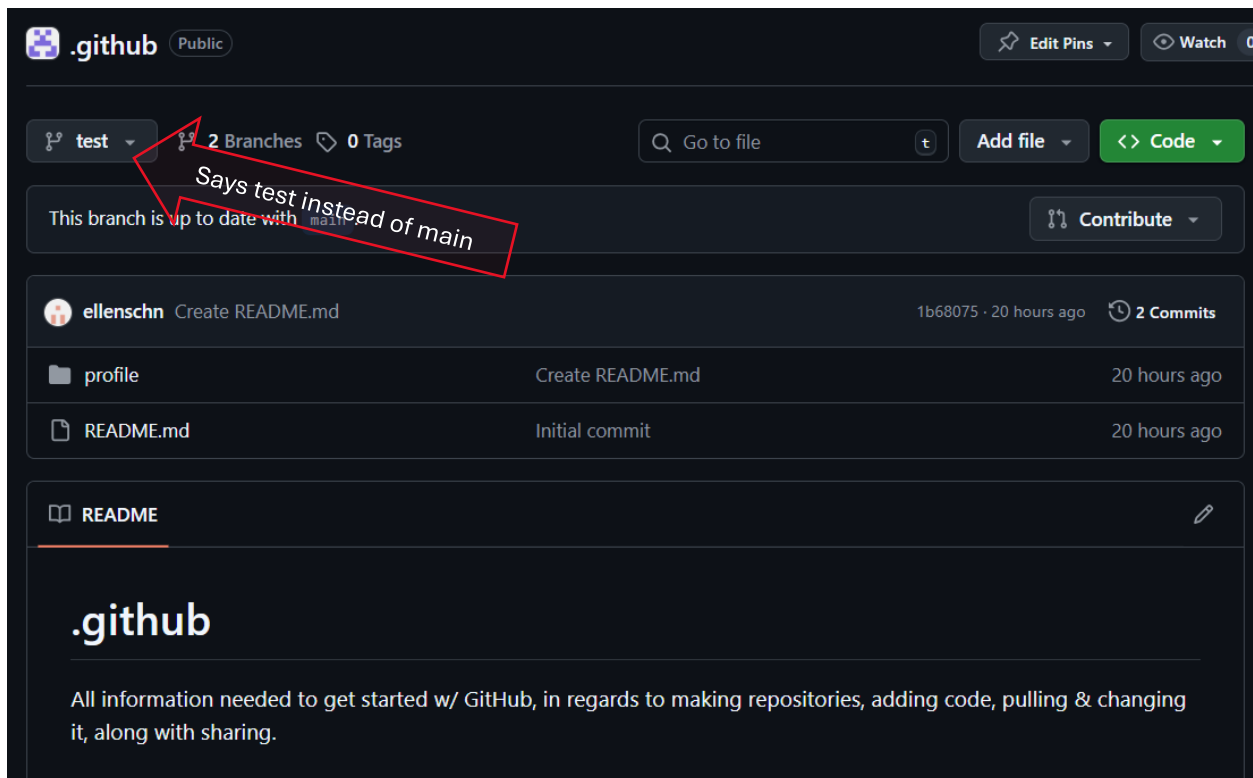
Used when all collaborators have write access to the repo on GitHub.



Version conflicts avoided when working on separate branches

To create a new branch, go to the dropdown menu that says main. Create a name for your branch and click create.






A new branch starts as an exact copy of the main branch, but any changes you make in the new branch won't affect the main branch. If the main branch gets updated after your branch is created, you can pull those updates into your branch as needed. If you want your branch's changes to become part of the main branch, you'll need to merge the branches.

To read more about branches: <https://heardlibrary.github.io/digital-scholarship/manage/control/github/branch/>

Committing Changes

In GitHub, saving any changes is called “committing”. Each commit has a title and an optional description (which is what you likely did just now by uploading code!) This allows for everyone to be aware of what changes are happening in the organization.

To modify any file, simply click on it and change it as you'd like. Once you're done, you must click on “commit changes” and write a commit message that describes what you have done.

 **.github** PublicEdit PinsWatch


test 2 Branches 0 Tags

Go to file

Add file

<> Code

This branch is up to date with `main`. Contribute

 **ellenschn** Create README.md 1b68075 · 20 hours ago 2 Commits

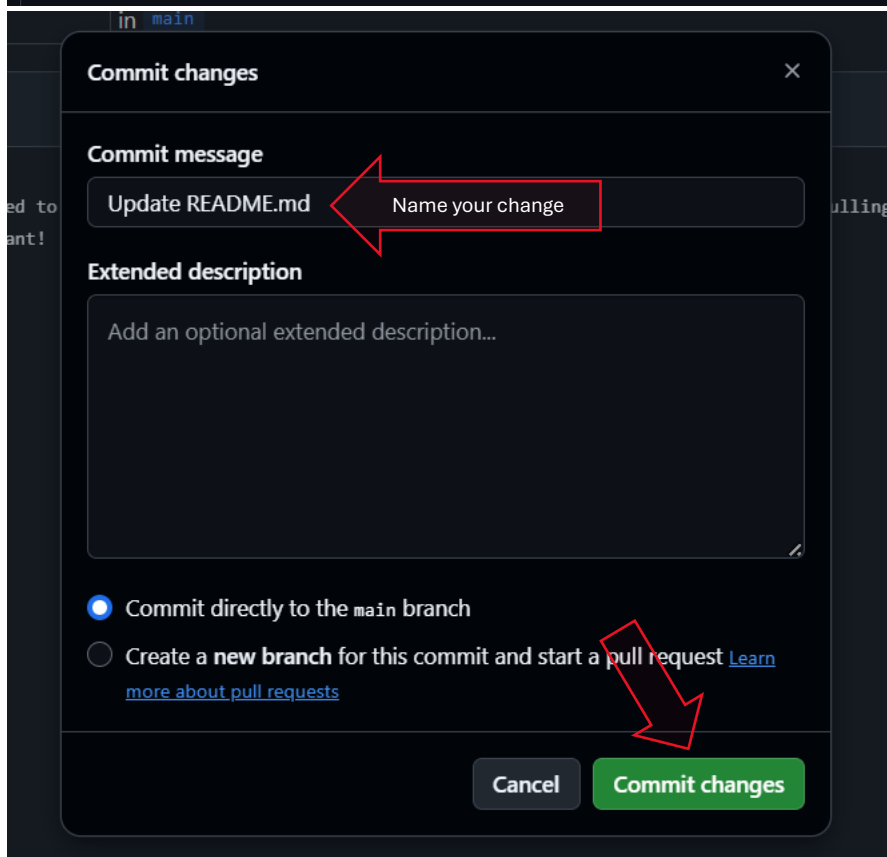
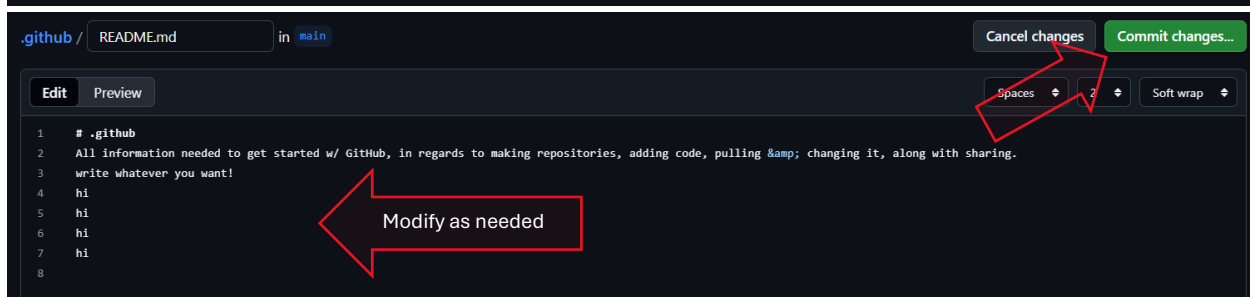
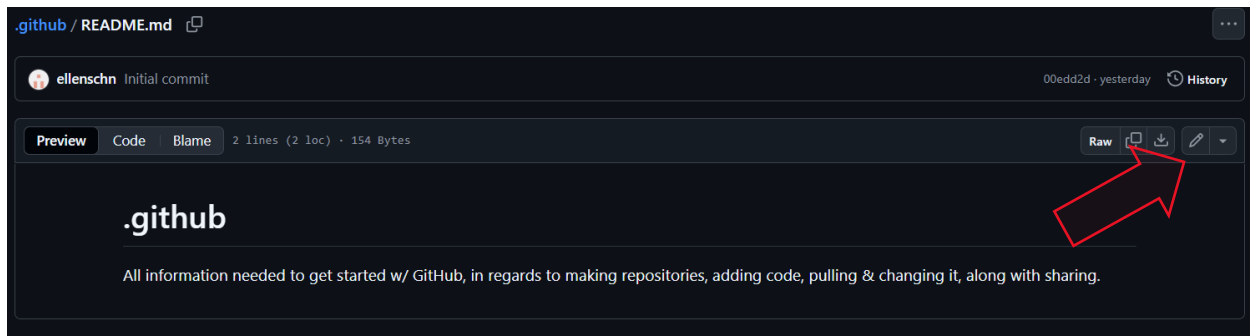
profile Create README.md 20 hours ago

README.md Click on whatever document you want to modify 20 hours ago

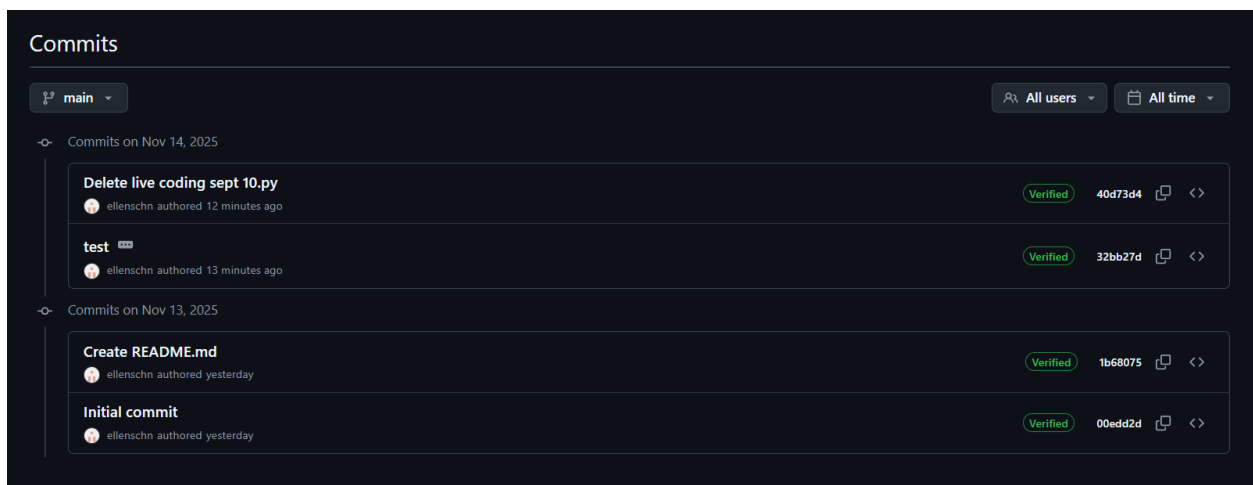
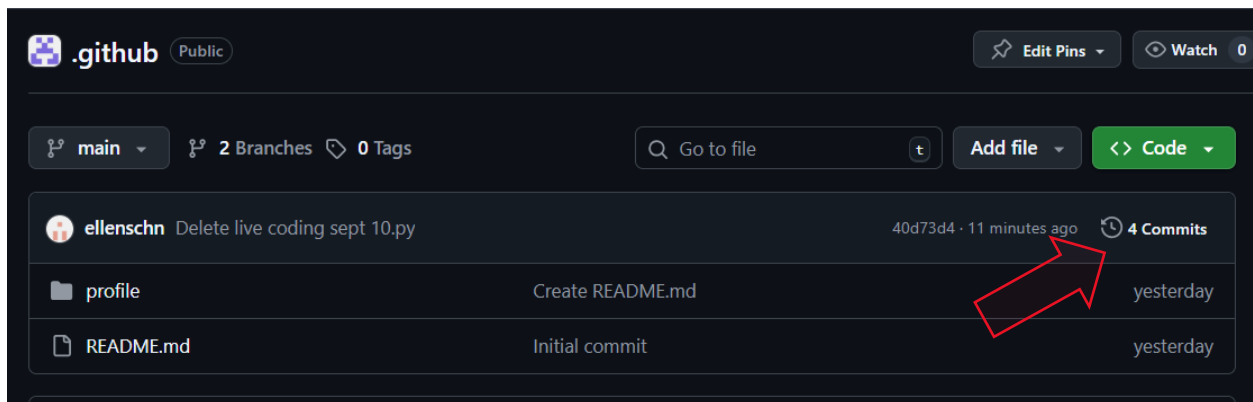
README

.github

All information needed to get started w/ GitHub, in regards to making repositories, adding code, pulling & changing it, along with sharing.

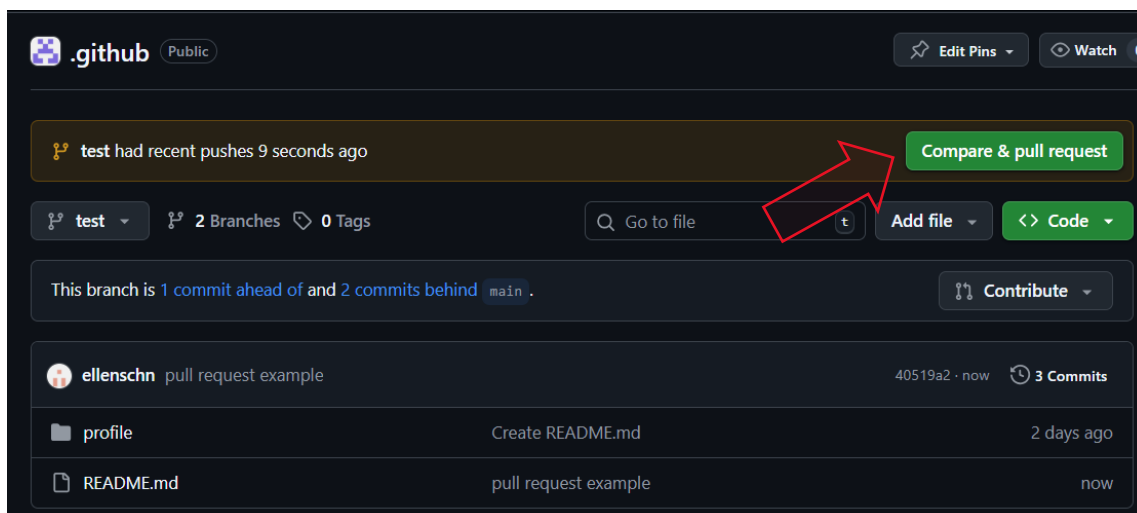


All recent commits are listed at the top of all files. Click on the number of commits to see a full list of changes.



Pull Requests & Merging Branches

Pull requests are for when you want to merge your branch with the main branch. Creating a pull request will signal to other collaborators for their input and display all the commits you are proposing. To create a pull request, first go into the branch that you want to “pull” into the main branch. There should be a button at the top saying “Compare & pull request”.



Open a pull request

Create a new pull request by comparing changes across two branches. If you need to, you can also create across forks. [Learn more about diff comparisons here.](#)

base: `main` ← compare: `test` ✓ **Able to merge.** These branches can be automatically merged.

Add a title

pull request example

Add a description

Write Preview

Add your description here...

Markdown is supported Paste, drop, or click to add files

Reviewers

No reviews

Assignees

No one—[assign yourself](#)

Labels

None yet

Projects

None yet

Milestone

No milestone

Development

Use [Closing keywords](#) in the description to automatically close issues

Helpful resources

[GitHub Community Guidelines](#)

Create pull request

Remember, contributions to this repository should follow our [GitHub Community Guidelines](#).

Annotations:

- Can request reviewers (will be notified to look at the code)
- Title it as you'd like
- Description as needed
- Assignees are those who worked on the code

pull request example #1

Open ellenschn wants to merge 1 commit into `main` from `test`

Conversation 0 Commits 1 Checks 0 Files changed 1

ellenschn commented [now](#)

No description provided.

pull_request example **Verified** 40519a2

ellenschn self-assigned this [now](#)

No conflicts with base branch

Merging can be performed automatically.

Merge pull request You can also merge this with the command line. [View command line instructions.](#)

Still in progress? [Convert to draft](#)

Add a comment


Write Preview


Add your comment here...

Annotations:

- See exactly what gets added
- Look at the commits proposed
- Add to the conversation about the code






**No conflicts with base branch**
Merging can be performed automatically.

Merge pull request

▼

You can also merge this with the command line. [View command line instructions.](#)

Still in progress? [Convert to draft](#)



Commit message
Merge pull request #1 from poulin-lab/test



Extended description
pull request example


This commit will be authored by ellen.s.chen@mail.mcgill.ca.


Confirm merge


Cancel


pull request example #1


 **Merged** ellenschn merged 1 commit into `main` from `test`  now

 Conversation 0


 Commits 1



 Checks 0


 Files changed 1



 ellenschn commented 5 minutes ago Member ...


No description provided.



  `pull request example` Verified `40519a2`

 ellenschn self-assigned this 5 minutes ago

  ellenschn merged commit `42a7c8e` into `main` now Revert



Pull request successfully merged and closed
You're all set — the `test` branch can be safely deleted.

Delete branch

Now the two branches have successfully merged! Note that the “pulled” branch doesn’t get deleted. If you no longer have use for the branch, you will have to manually delete it.

Forking is another way to contribute to workflows. This is a more advanced technique, feel free to read about it here: <https://docs.github.com/en/pull-requests/collaborating-with-pull-requests/working-with-forks/fork-a-repo>