

Git, GitHub, and Jupyter

In-class exercise

Ty Janoski

City College of New York, CUNY

**Let's set up a workflow for your
in-class project.**

Creating a GitHub repository

Have *one* member of your group create a repository.

This person will be the owner.*

* The owner doesn't have any special responsibilities, so don't worry.

Overview

Repositories 12

Projects

Packages

Stars 2



Ty Janoski

tyfolino · he/him

Pinned

 **climkern** Public

Python package for easily computing climate kernels.

Python 1

 **CESM_timing** Public

store timing files for CESM simulations

132 contributions in the last year

Q Type ¹ to search


>_

+ ▾

🕒

🔗

📁



📦 Packages


☆ Stars **2**

Find a repository...

Type ▾

Language ▾

Sort ▾

 **New**

spr24-susc-lecture Public

This repository holds the lectures notes and setup instructions for the 2/7 guest lecture.

● Jupyter Notebook

Updated 3 hours ago

☆ Star ▾

ida-precip Public

● Python

Updated last week

☆ Star ▾

- Pick a name for your repository, like `SUSC`.
- Give it a short description.
- Leave it public.
- You can skip the `README`, `.gitignore`, and license for now.



Public

Anyone on the internet can see this repository. You choose who can commit.



Private

You choose who can see and commit to this repository.

Initialize this repository with:



Add a README file

This is where you can write a long description for your project. [Learn more about READMEs.](#)

Add .gitignore

.gitignore template: None ▼

Choose which files not to track from a list of templates. [Learn more about ignoring files.](#)

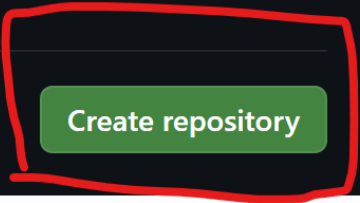

Choose a license

License: None ▼

A license tells others what they can and can't do with your code. [Learn more about licenses.](#)

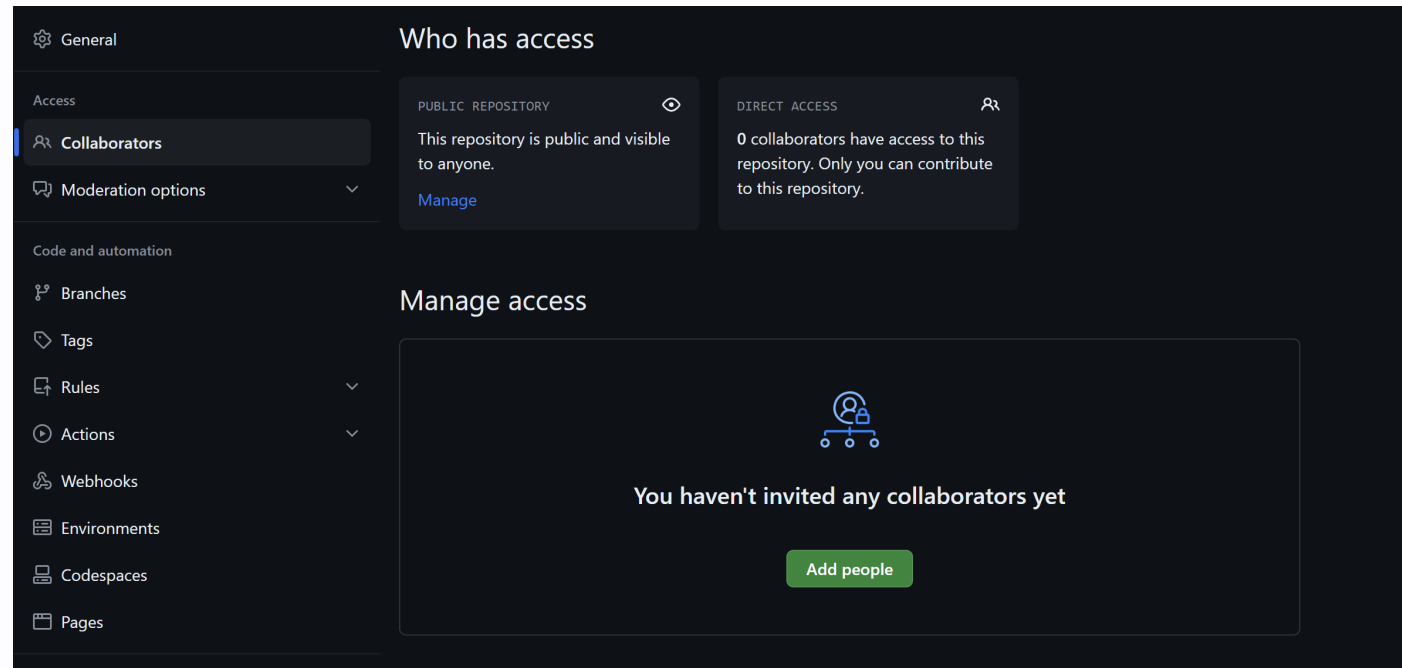


You are creating a public repository in your personal account.



Create repository

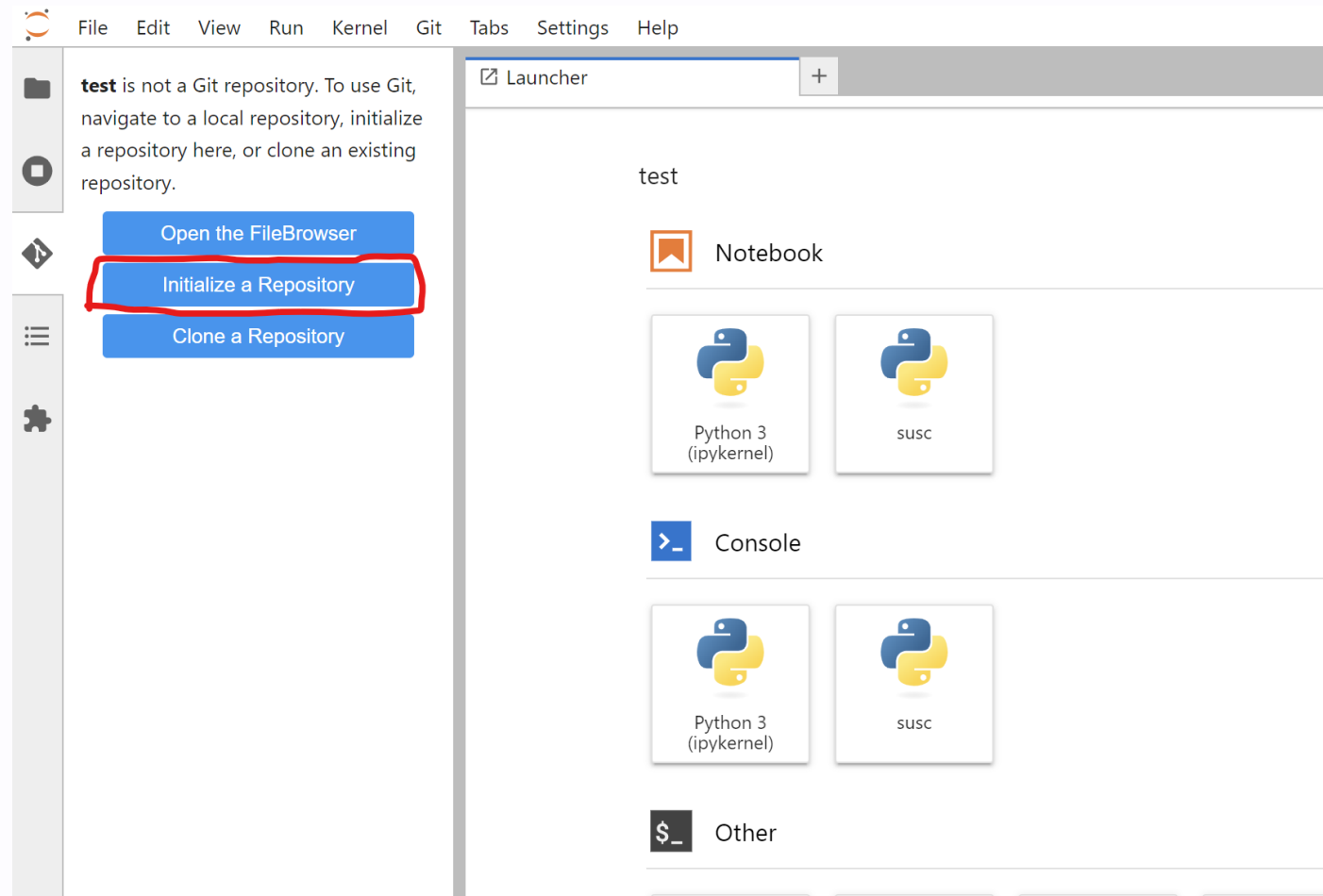
Invite your group members as collaborators



Create a new folder on your computer where your Jupyter Notebook for this class will live.

Make sure you can navigate to it using JupyterLab!

Initialize a new repository



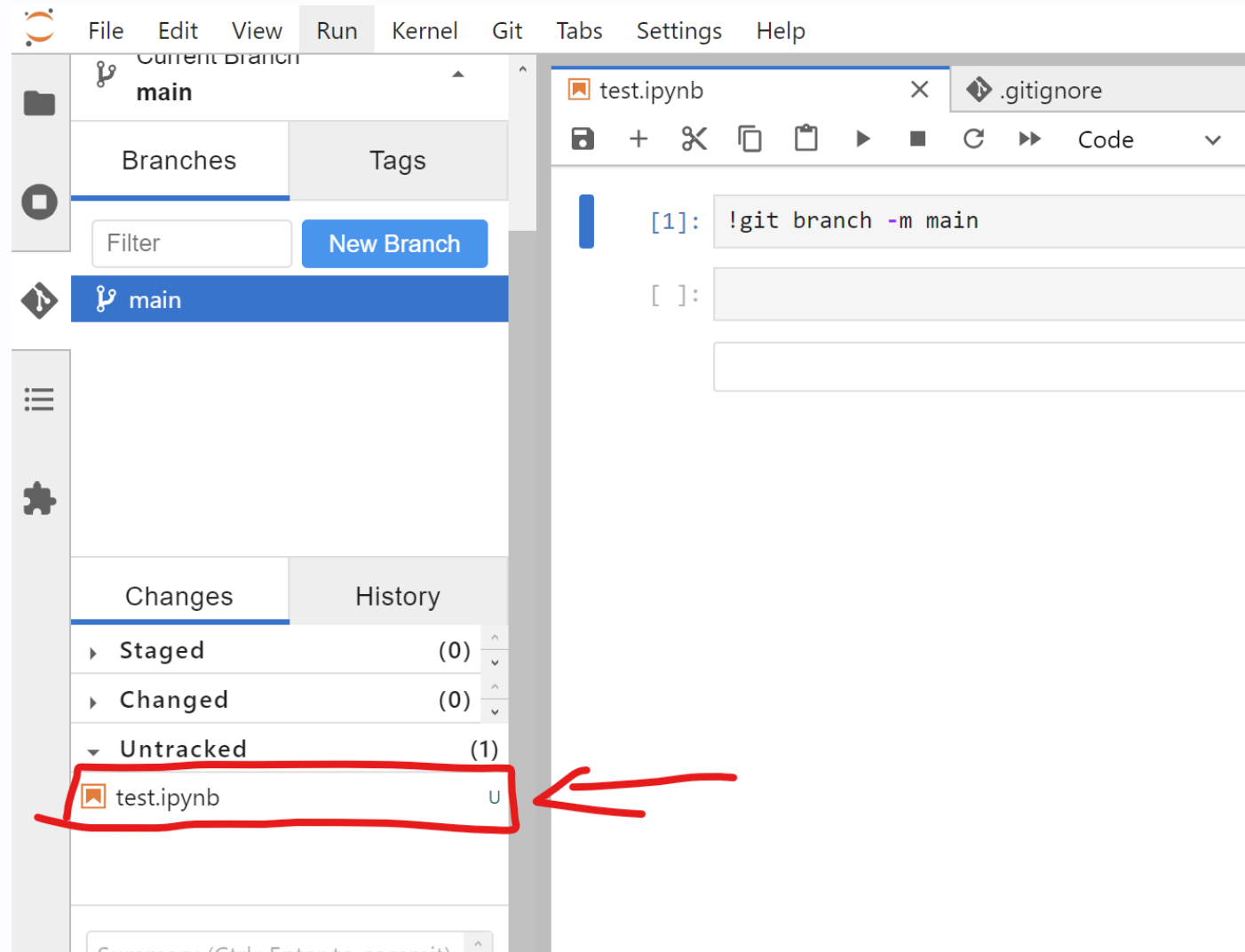
Create a new notebook

- When selecting the kernel, make sure it is `susc`
- In your first cell, run this code:

```
!git branch -m main  
!git config --global user.name "First Last"  
!git config --global user.email "your-github-  
email@example.com"
```

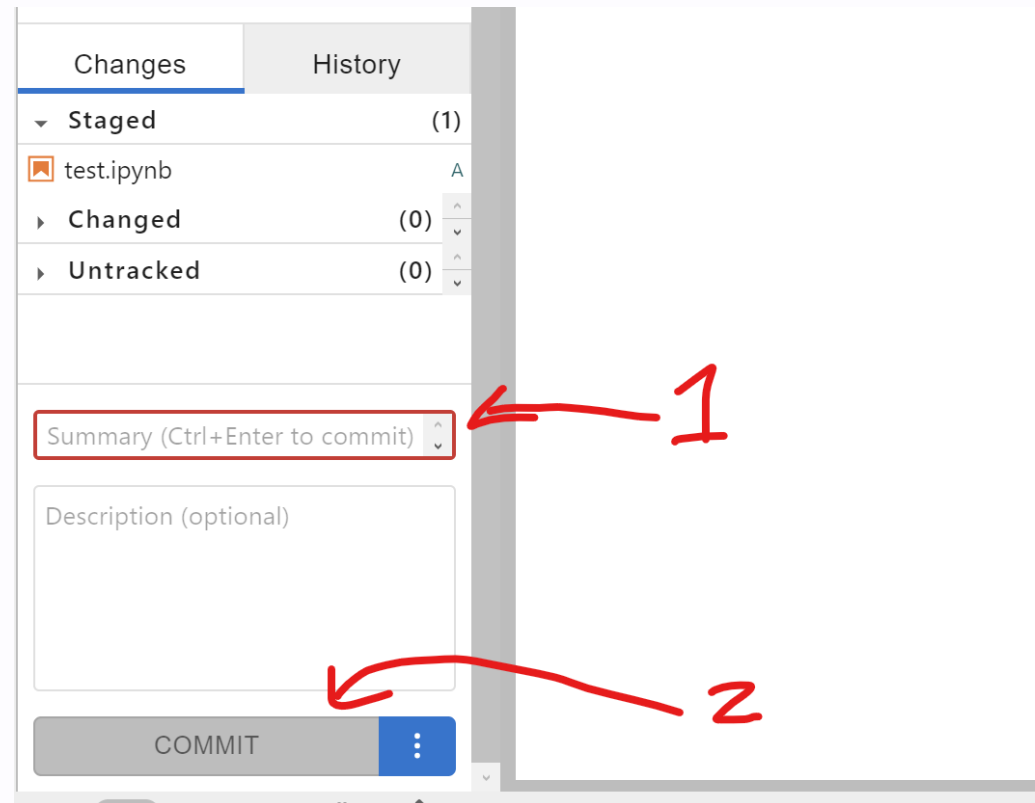
Add your notebook

Hit the plus sign.



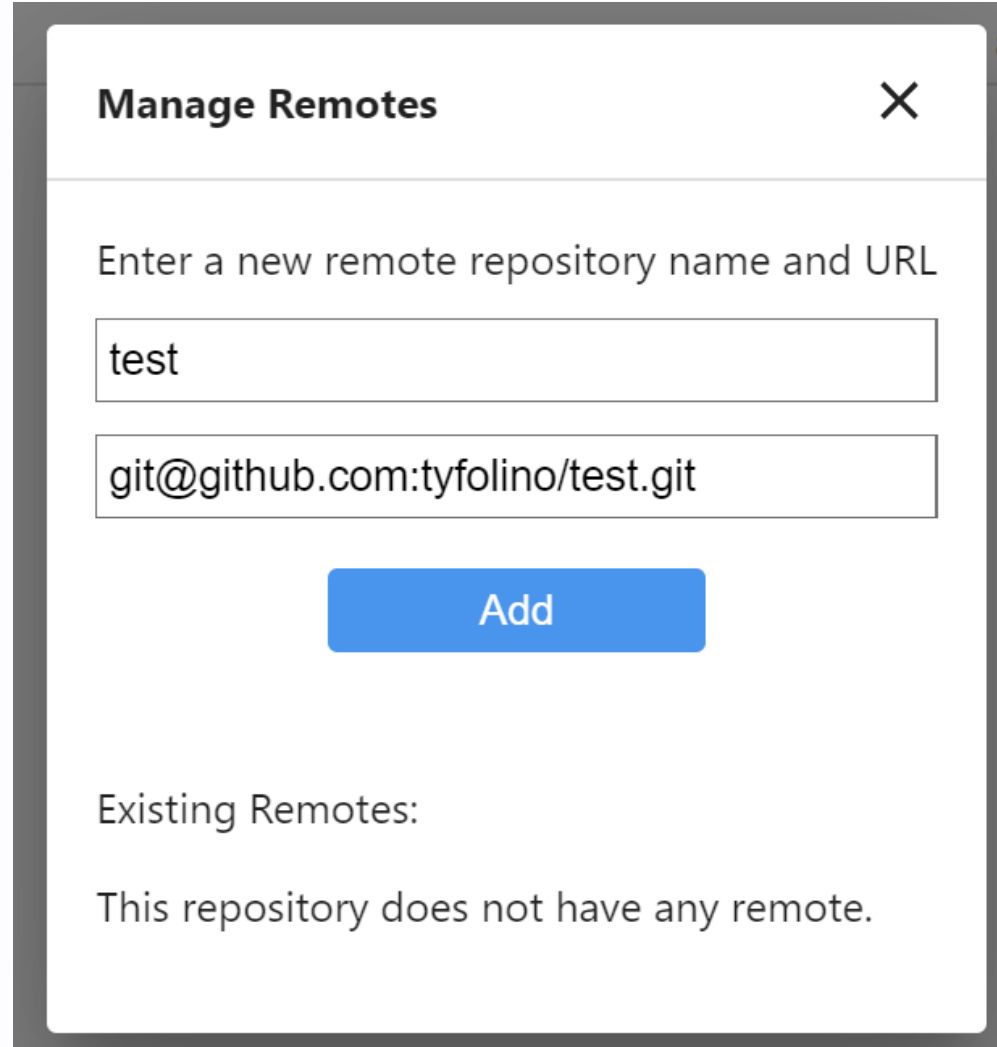
Commit the changes

An example summary is " added my first file "



Connect to remote repository

In the top menu bar, git -> manage remote



The image shows a 'Manage Remotes' dialog box with a close button (X) in the top right corner. The dialog contains a text input field for the repository name, which has 'test' entered, and another text input field for the repository URL, which has 'git@github.com:tyfolino/test.git' entered. Below these fields is a blue 'Add' button. At the bottom of the dialog, it says 'Existing Remotes:' followed by the message 'This repository does not have any remote.'

Manage Remotes X

Enter a new remote repository name and URL

test

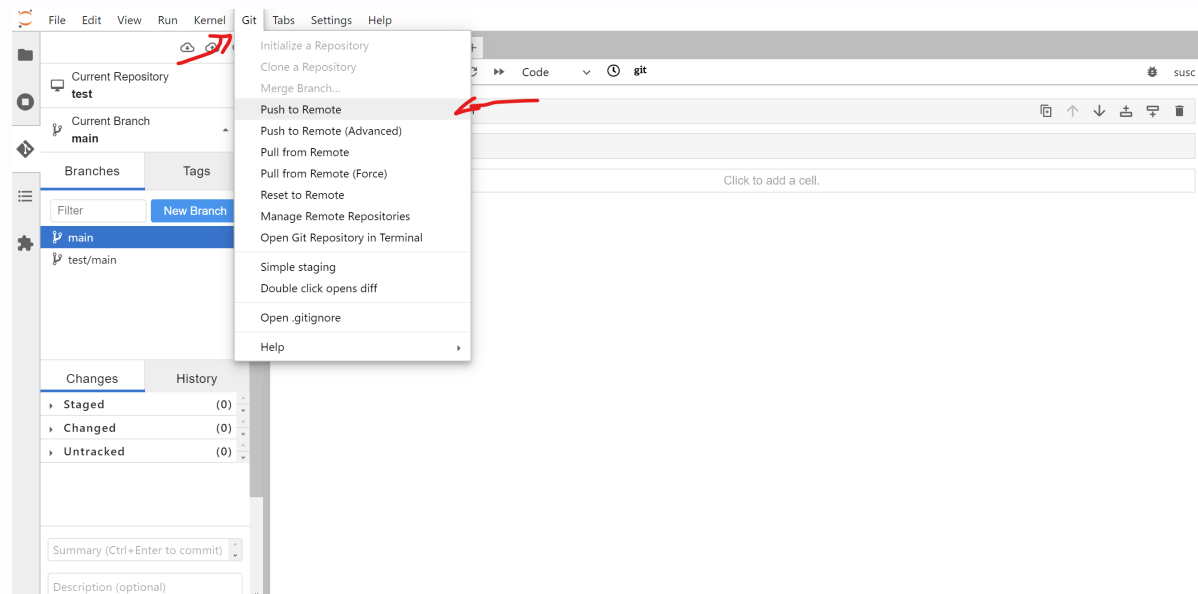
git@github.com:tyfolino/test.git

Add

Existing Remotes:

This repository does not have any remote.

Push to GitHub



**In case people are having
authentication issues...**

Authentication to remote repository hosts

If you are seeing errors similar to `[E yyyy-mm-dd hh:mm:ss ServerApp] 500 POST /git/<clone|push|pull|status>` on the console which is running the JupyterLab server, you probably need to set up a credentials store for your local Git repository. This extension tries to handle credentials for HTTP(S) connections (if you don't have set up a credential manager). But not for other SSH connections.

A note for windows users.

“ For Windows users, it is recommended to install [git for windows](#). It will automatically set up a credential manager.

In order to connect to a remote host, it is recommended to use SSH.

”

SSH protocol

Here are the steps to follow to set up SSH authentication (skip any that is already accomplished for your project):

1. Create a SSH key
2. Register the public part of it to your Git server:
 - GitHub

3. Tell your local Git repository to **connect to remote via ssh**

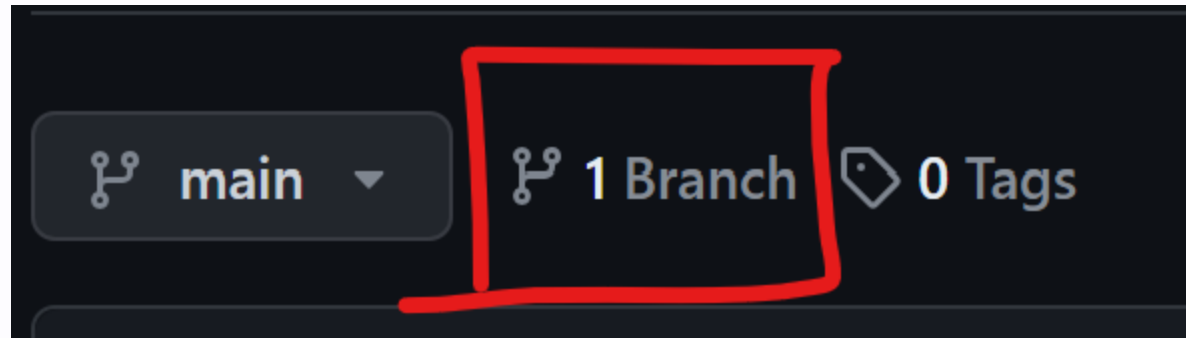
You should now be able to pull and push committed changes to and from your remote repository using the respective buttons on the top of the extension's panel.

If it worked, refresh your GitHub repository page and see your file there now!

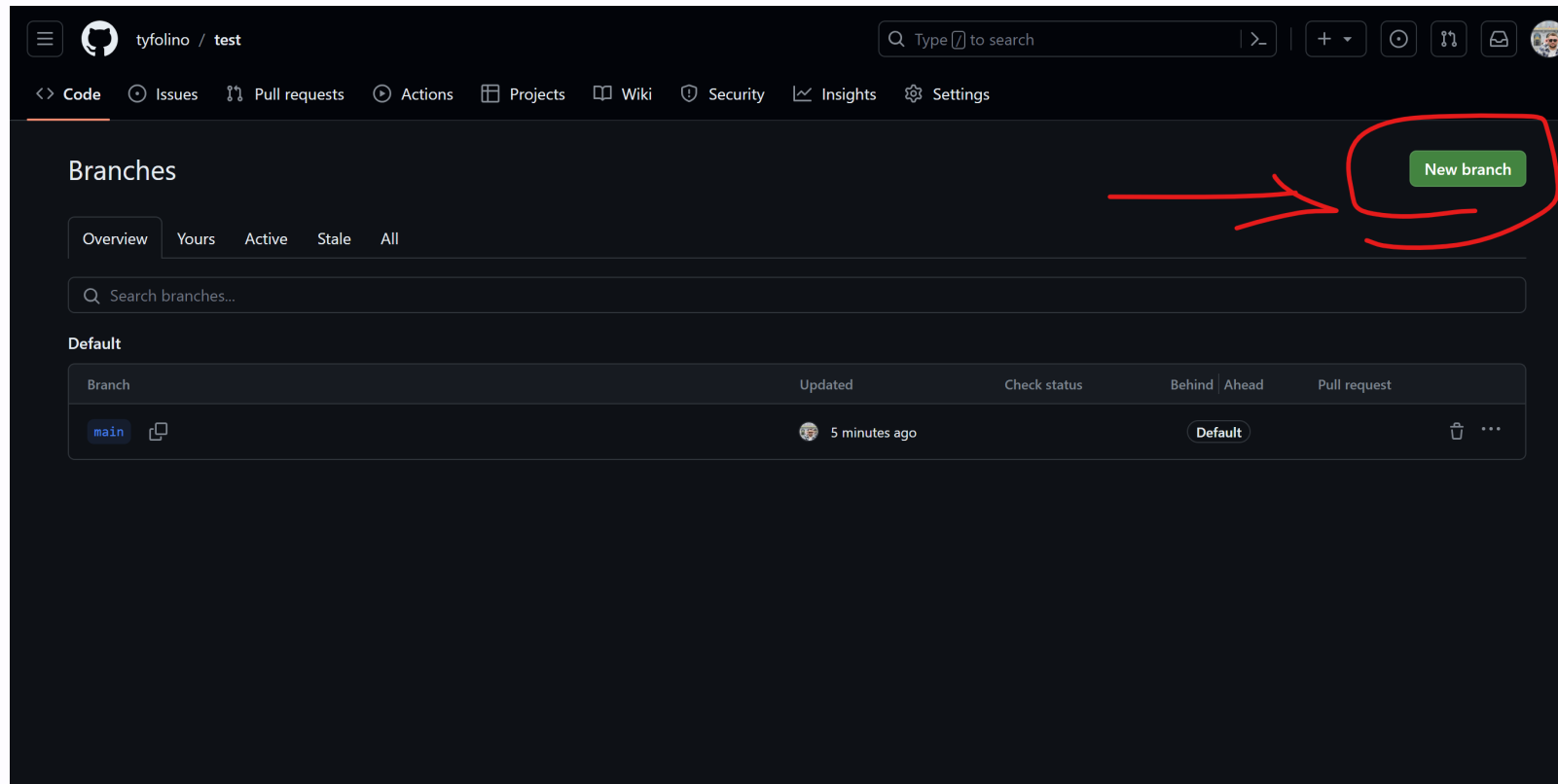
Now it's time for branches.

Create a new branch for each group member.










Click on branches.












Make a new branch.

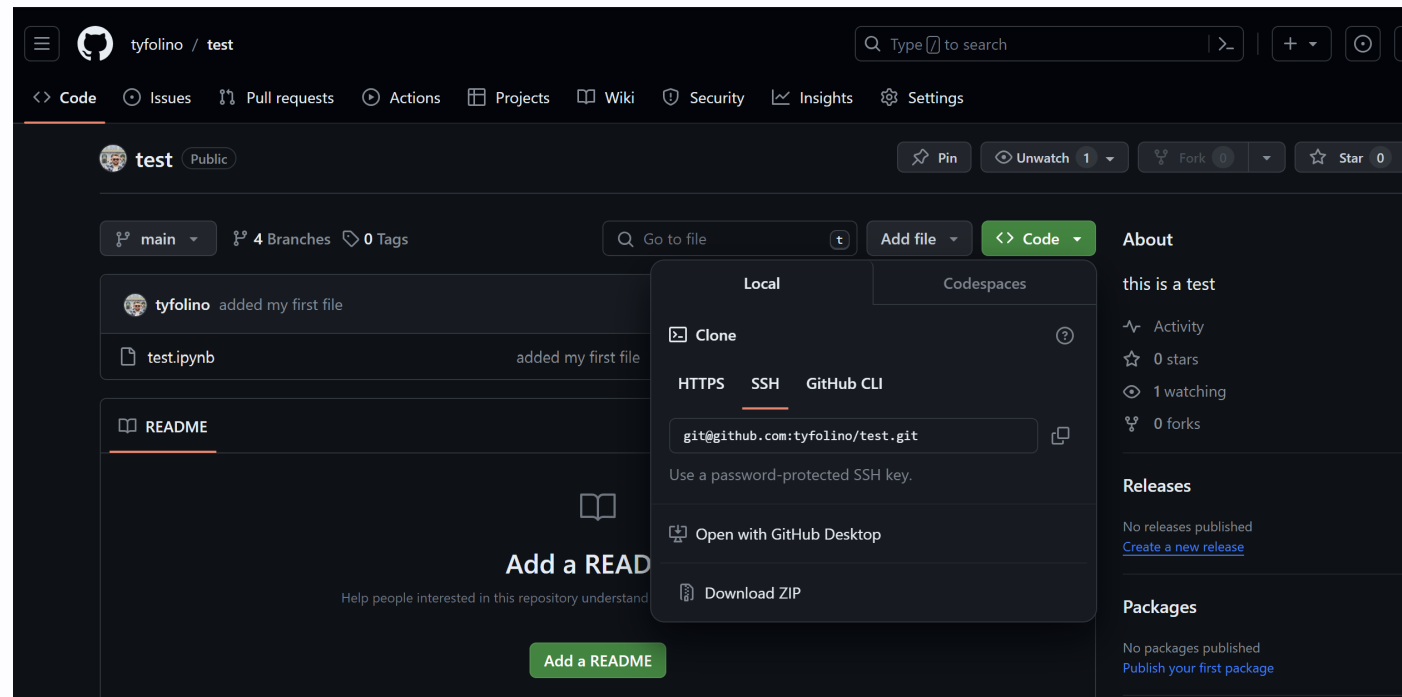


Create branches for each group member including the owner!

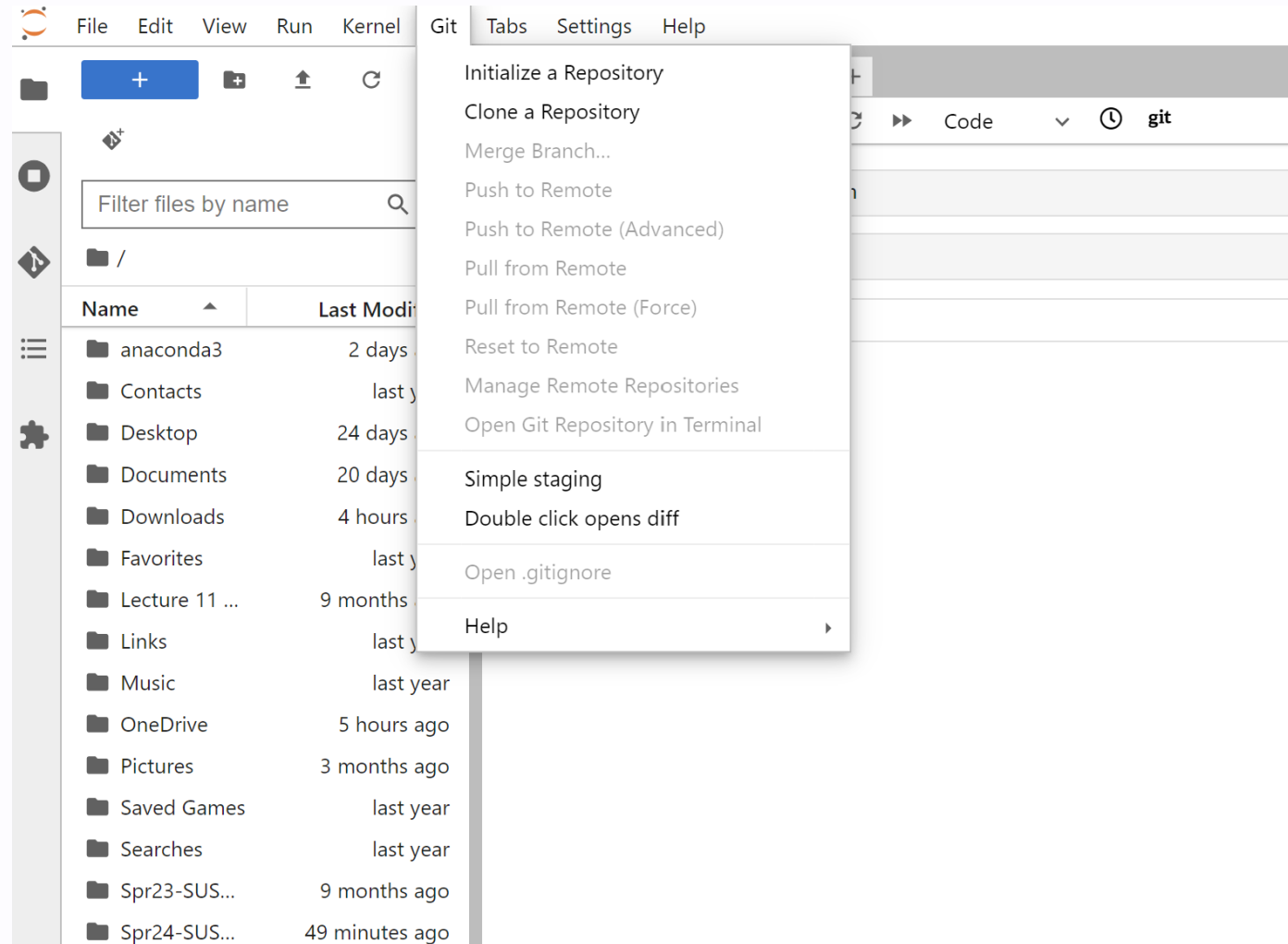
Your branches						
Branch		Updated	Check status	Behind	Ahead	Pull request
Student-C		 now		0	0	 ...
Student-B		 now		0	0	 ...
Student-A		 now		0	0	 ...

Active branches						
Branch		Updated	Check status	Behind	Ahead	Pull request
Student-C		 now		0	0	 ...
Student-B		 now		0	0	 ...
Student-A		 now		0	0	 ...

Now, the other members can clone the repository.

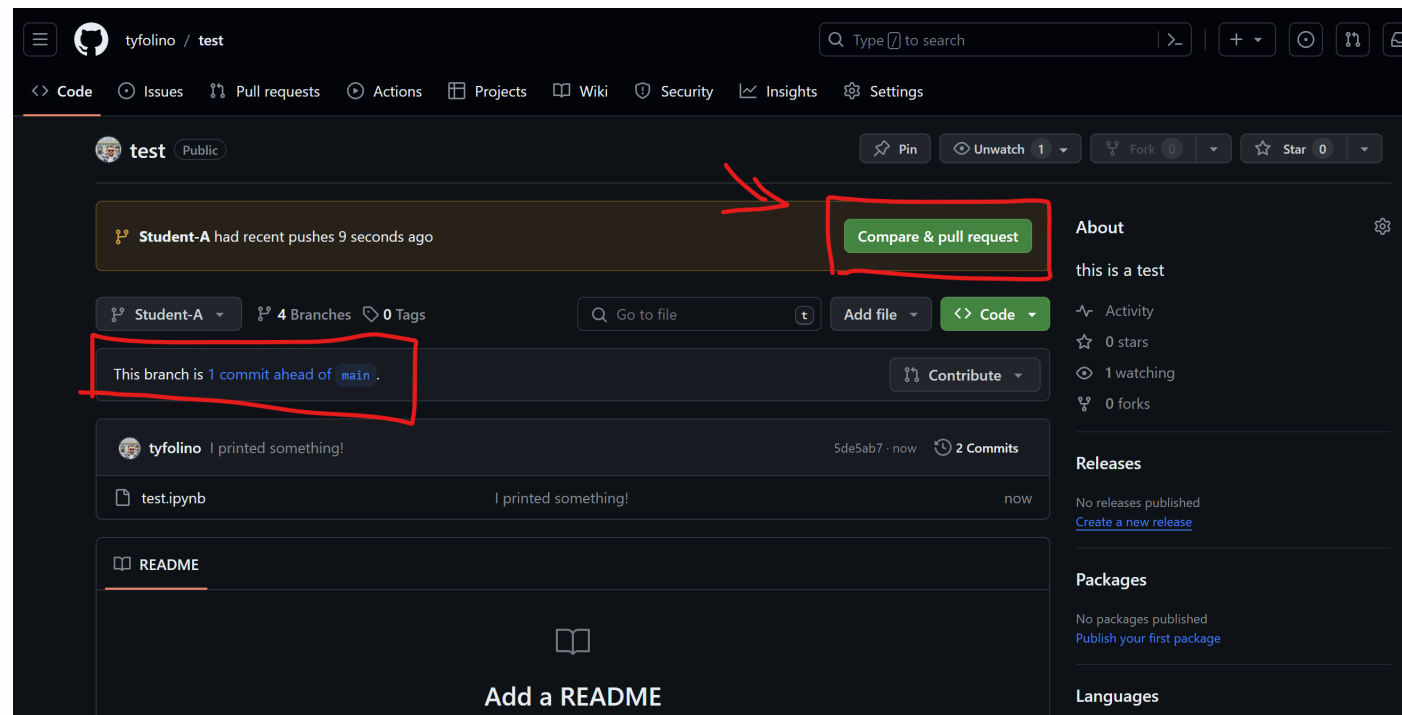


Paste the git link into the `git clone` dialog box



**When you work on this group project,
make sure you are in *your* branch!**


When you are ready to merge your changes into the `main` branch, make a pull request.

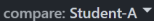


Open a pull request


Open a pull request

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

base: main














compare: Student-A

✓ **Able to merge**. These branches can be automatically merged.


**Add a title**


Add a description


WritePreview

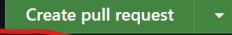
H B I             


I added a line of code where I printed something. It's really great.





 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


Merge pull request

Conversation 0

Commits 1

Checks 0


Files changed 1





tyfolino commented 1 minute ago

Owner ...

I added a line of code where I printed something. It's really great.







 I printed something!


5de5ab7

Add more commits by pushing to the [Student-A](#) branch on [tyfolino/test](#).



 Continuous integration has not been set up

[GitHub Actions](#) and [several other apps](#) can be used to automatically catch bugs and enforce style.

 This branch has no conflicts with the base branch

Merging can be performed automatically.

Merge pull request

You can also [open this in GitHub Desktop](#) or view [command line instructions](#).