Things needed today:
`python_practice.py` [last class]
Open github.com & log in

Coding Collaboratively with GitHub

Dr. Groves

Dr. Peirce-Cottler

Module 0 Lecture 2

Learning objective: I know how to upload my code to GitHub, pull code from my partner, comment my code for future use, and write

The Restaurant: The Big Picture



Anaconda

How could we share recipes with other restaurants?

Code



Local Folder



Terminal



Packages



Python









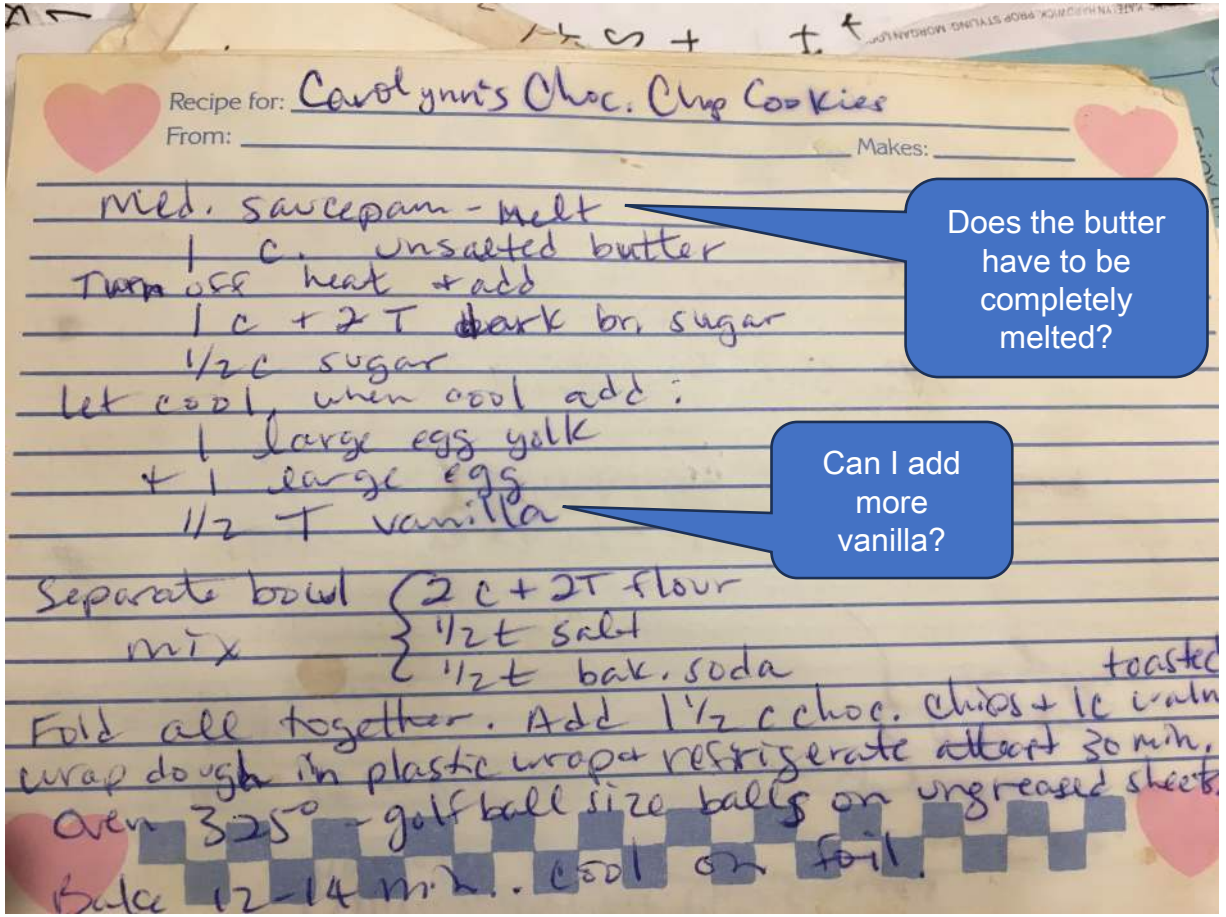


The Restaurant (VS Code/IDE)

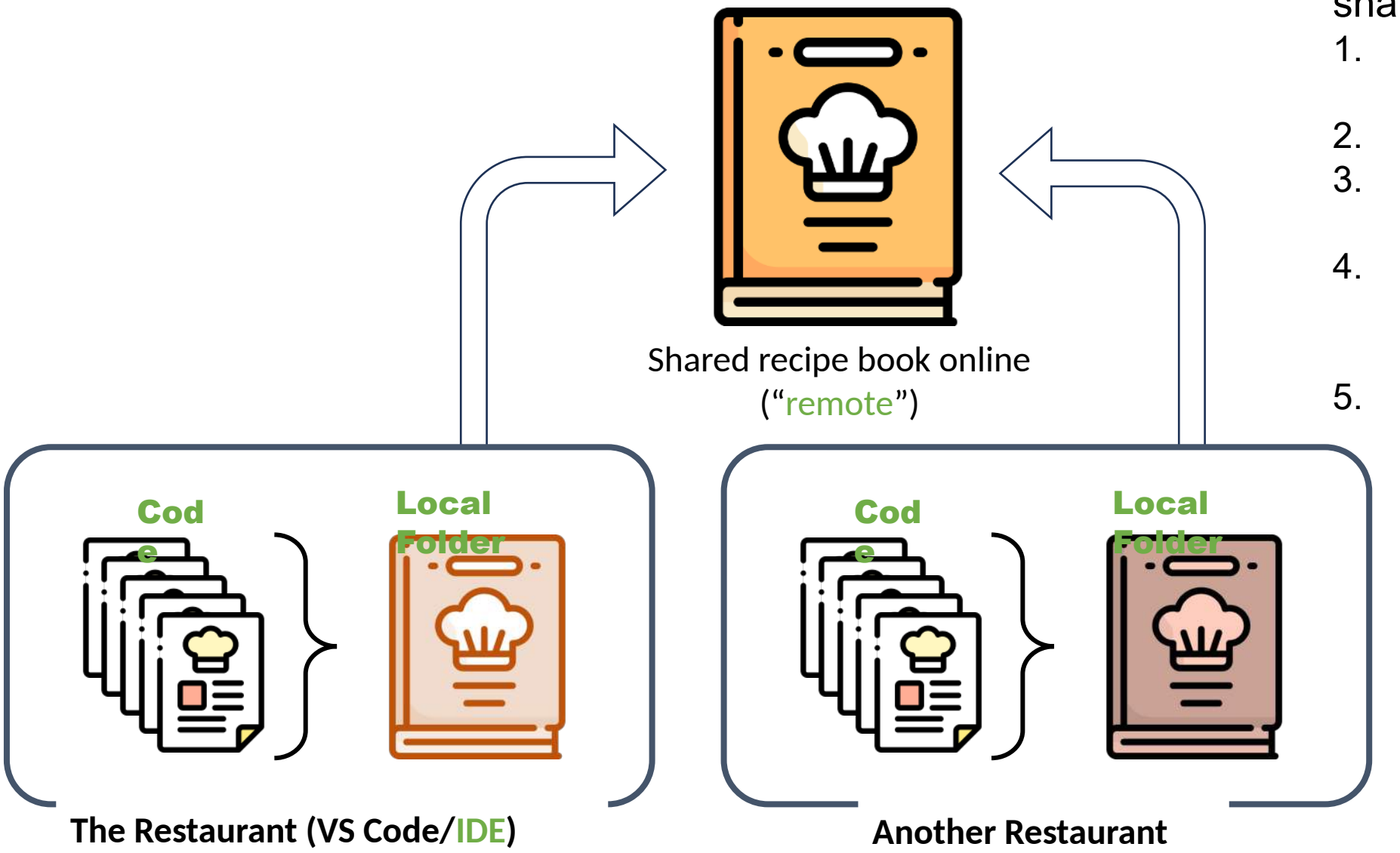
Conda

We could share recipes by…

- Telling them face to face
  - Good when you don't want to share a secret recipe
  - Bad for coding
- Handing them a copy or a picture
  - What if it's hard to read?
  - What if they want to update the recipe and send it back?
  - What if they want to see multiple iterations of the recipe (did they recipe-test?)
- Typing up the recipe and sending them the most recent version and past versions with notes about why things changed



Outside the restaurant: sharing recipes



How do we contribute to the shared recipe book online?

1. Get a local copy of the shared recipe book (“clone”)
2. Make changes (“commit”)
3. Add these changes back to the remote book (“push”)
4. Regularly check if anyone else made changes to the shared book (“pull”)
5. If someone made conflicting changes, figure out how to combine them (“merge”)

This is how GitHub works

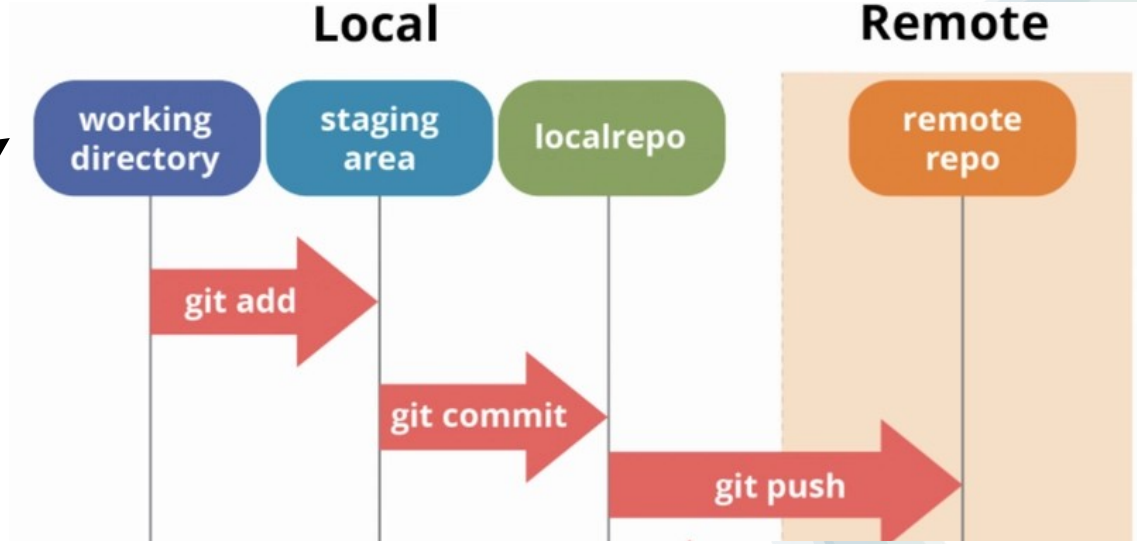
- Github hosts your code, git is the version control system
- In this class, we will use GitHub to **share code** (with partners and instructors), code **collaboratively**, and track **changes**

Remote, shared

GitHub

Local, individual

The “commit” process



https://livablesoftware.com/development-process-in-github-basic-infographic/

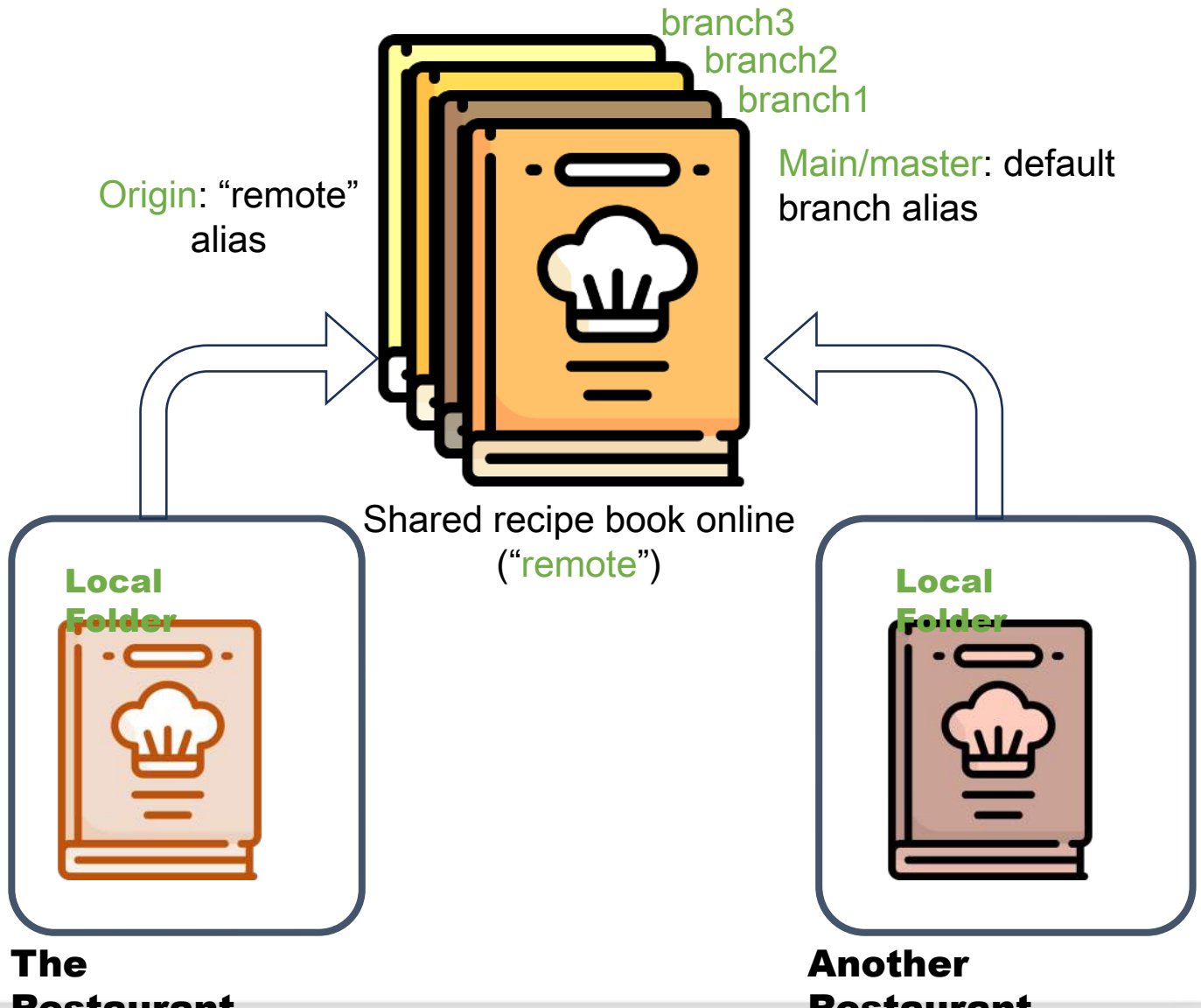
GitHub Repositories

- Entirely different recipe books
- In this class, we will use (shared) repositories for each project

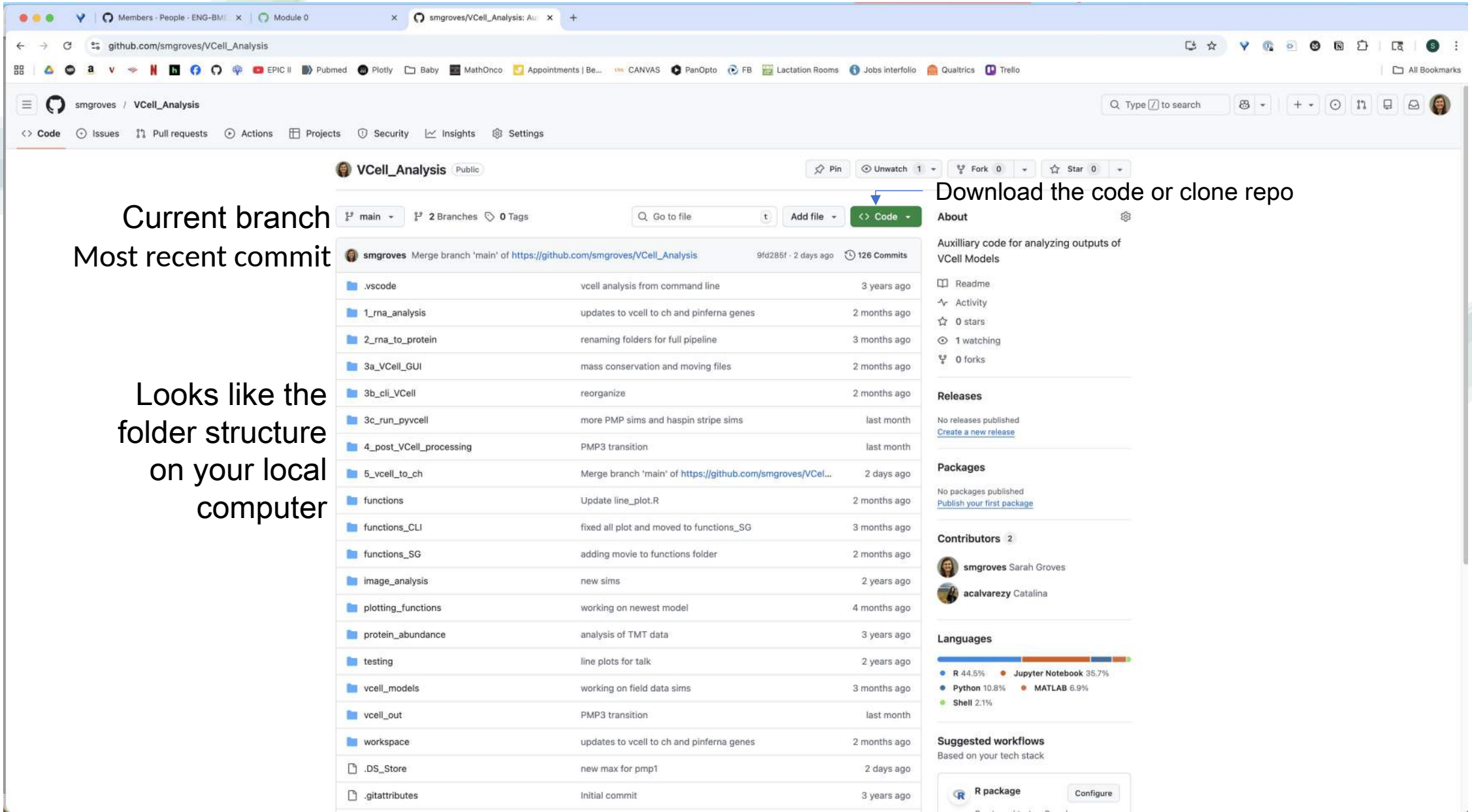


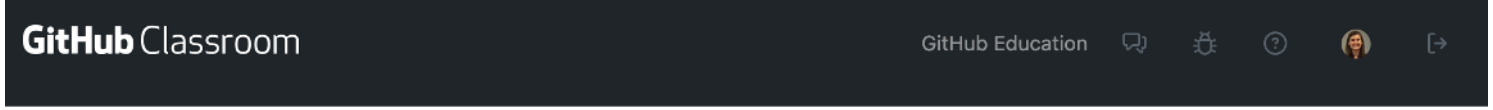
GitHub Branches

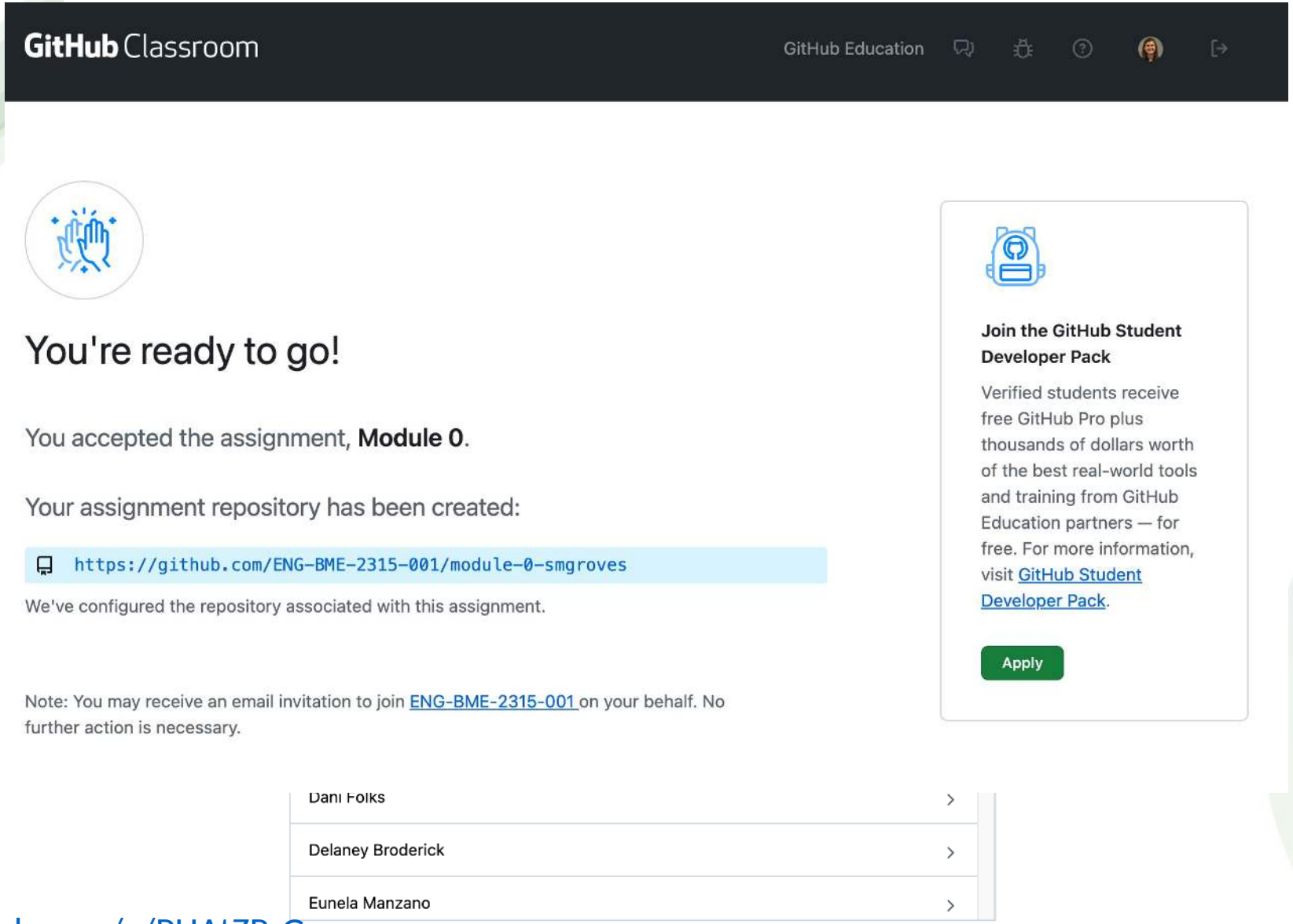
- Different remote versions of the same shared book
- In this class, we will use branches for individual work within a shared project (starting with Module 1)



GitHub (remote, online)





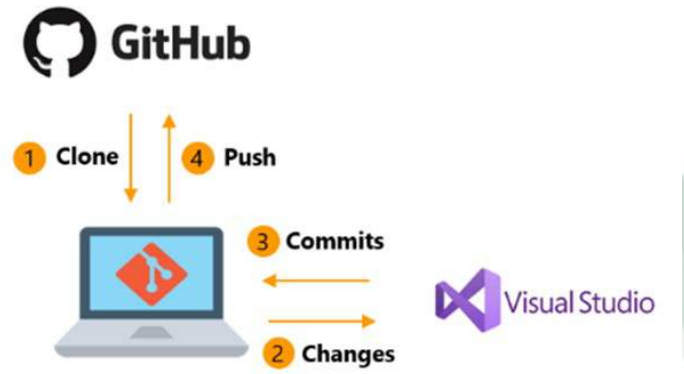


https://classroom.github.com/a/PHAtZR-G

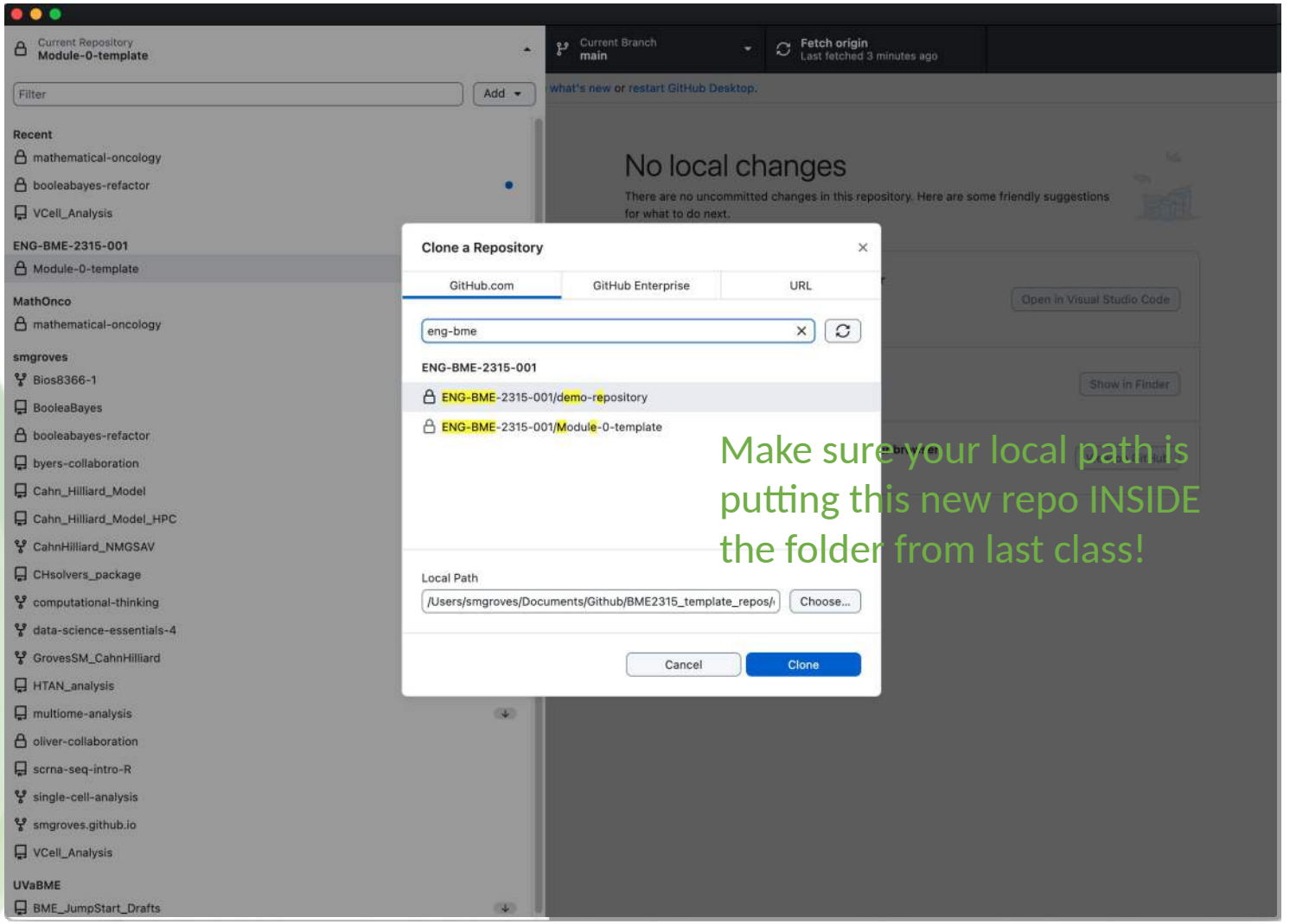
LET'S PRACTICE! Getting set up with GitHub Classroom

1. You should have a github account.
2. Join ENG-BME-2315-001 organization (from email). This will show up on your page under "Organizations"
3. Go to https://classroom.github.com/a/PHAtZR-G & find your name in the roster.
4. Accept Module_0 assignment (it should add your github username)
5. Find your repository by going to the ENG-BME-2315-001 organization (on your homepage) and click on the repository named "Module_0_ YourUserName" corresponding to your name.
6. Clone the repo to your local computer
7. Fetch origin/main
8. Open in VS Code
9. Update README, save, and push changes to the main branch

Stand your name card on its edge when you are done.
Help your neighbors!
If you need a TA/instructor, raise your hand!

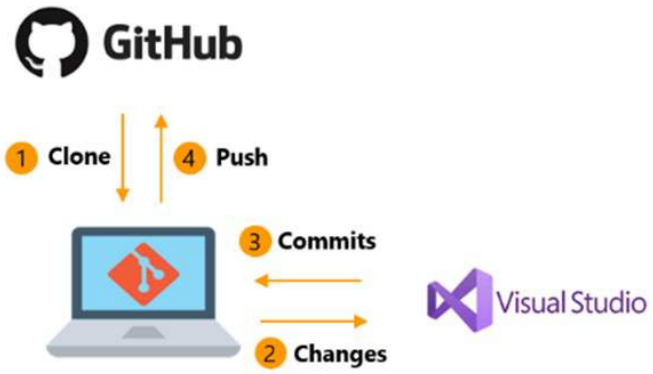


Getting set up with GitHub Desktop & git on command line

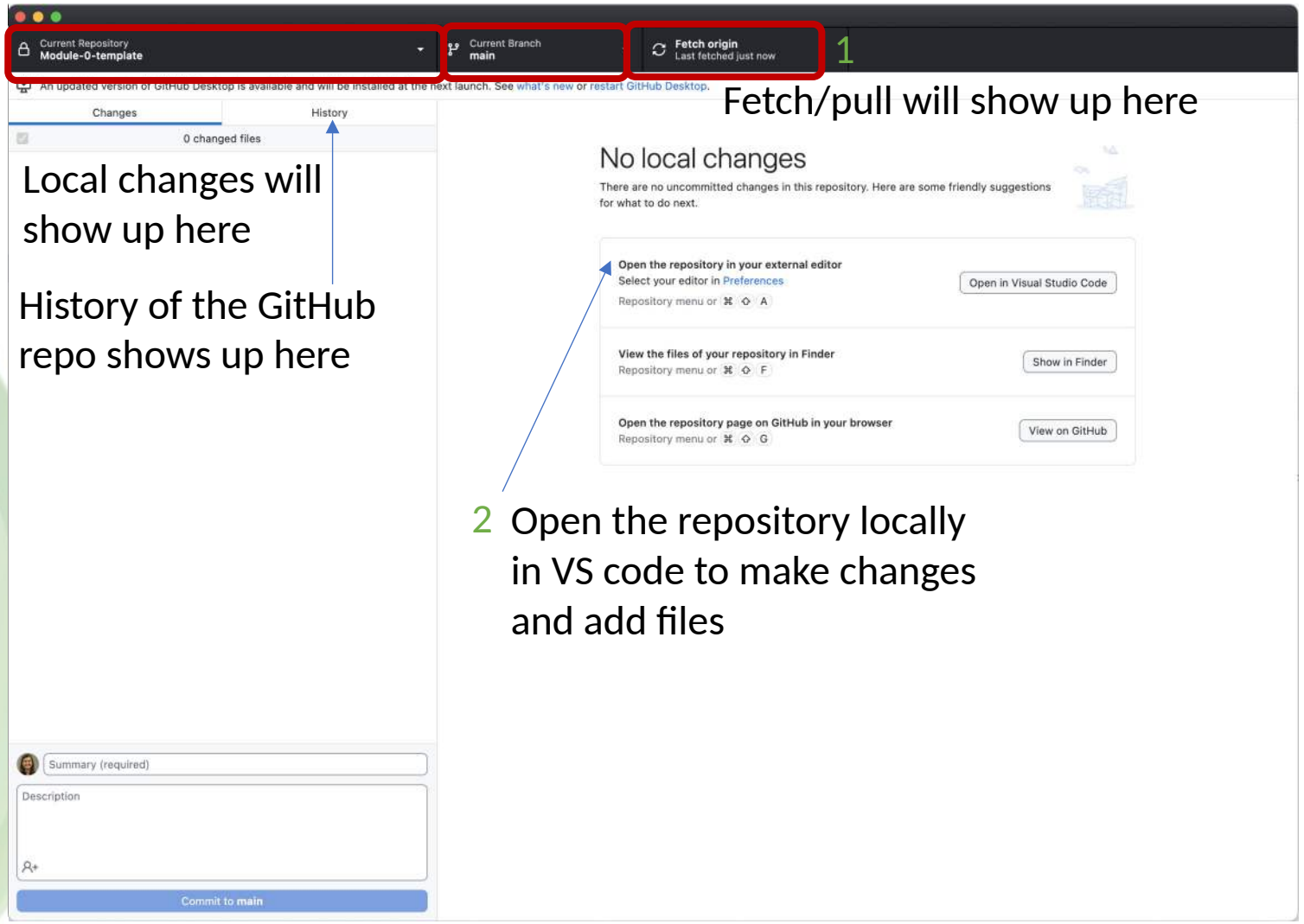


Alternative in terminal:

```
cd /Your/folder/for/this/class
git clone https://github.com/ENG-BME-2315-001/Module-0-LastName.git
git pull origin main
```

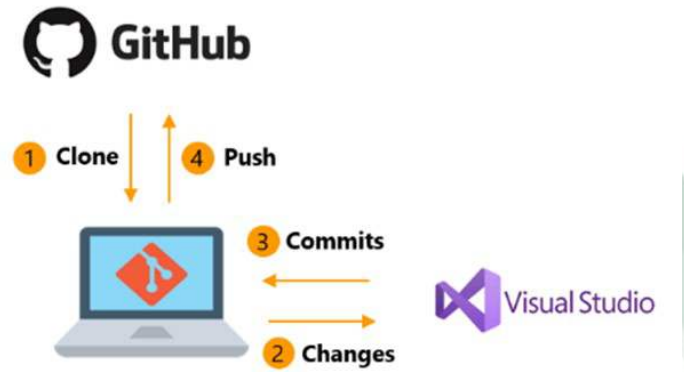


Getting set up with GitHub Desktop & git on command line

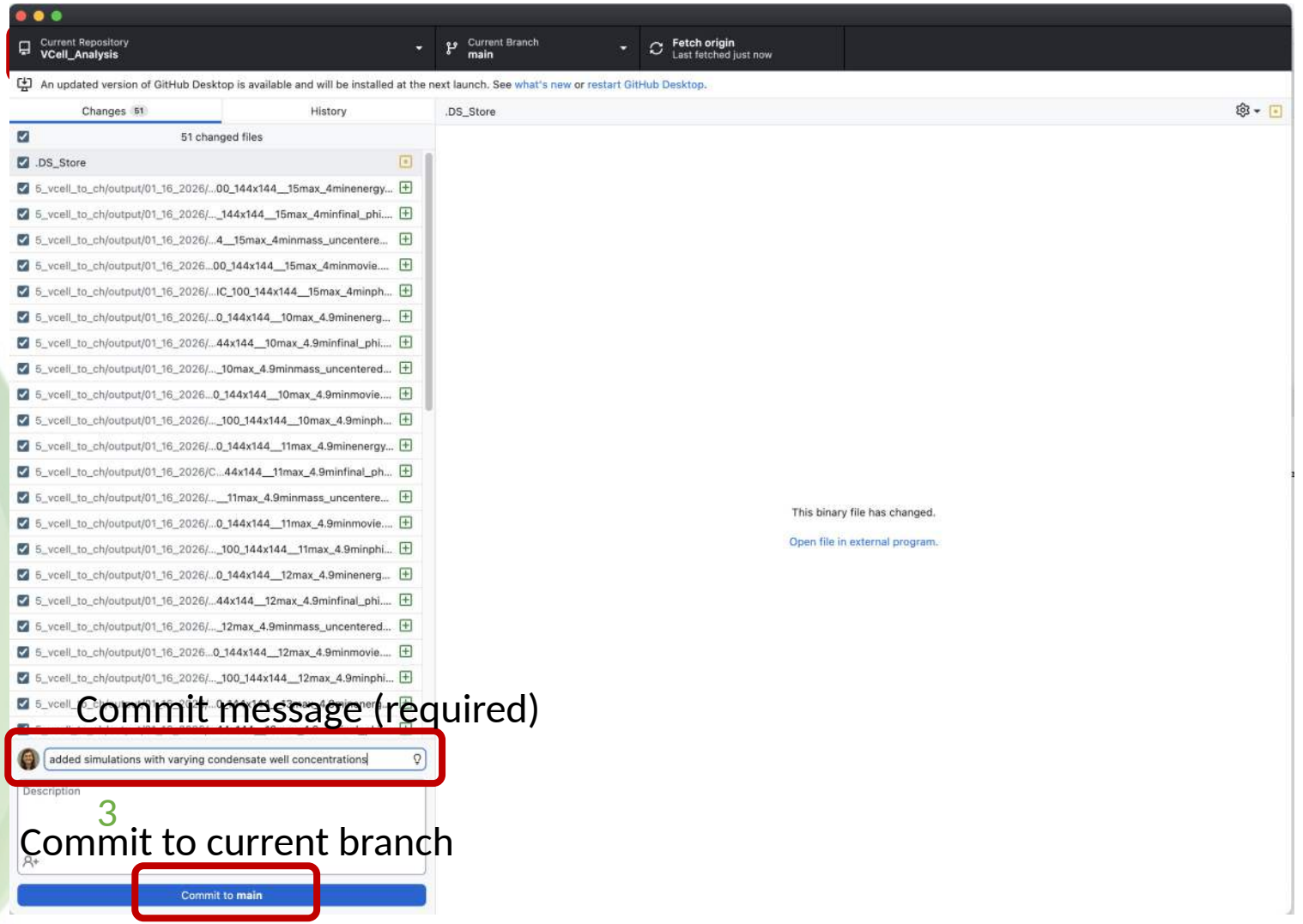


Alternative in terminal:

```
cd /Your/folder/for/this/class
git clone https://github.com/ENG-BME-2315-001/Module-0-LastName.git
git pull origin main
```



Getting set up with GitHub Desktop & git on command line



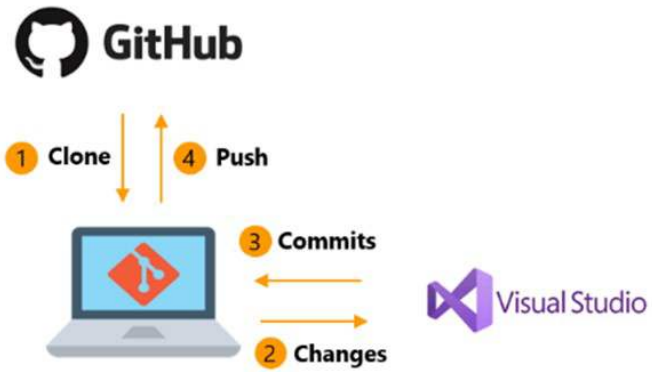
Commit message (required)

3

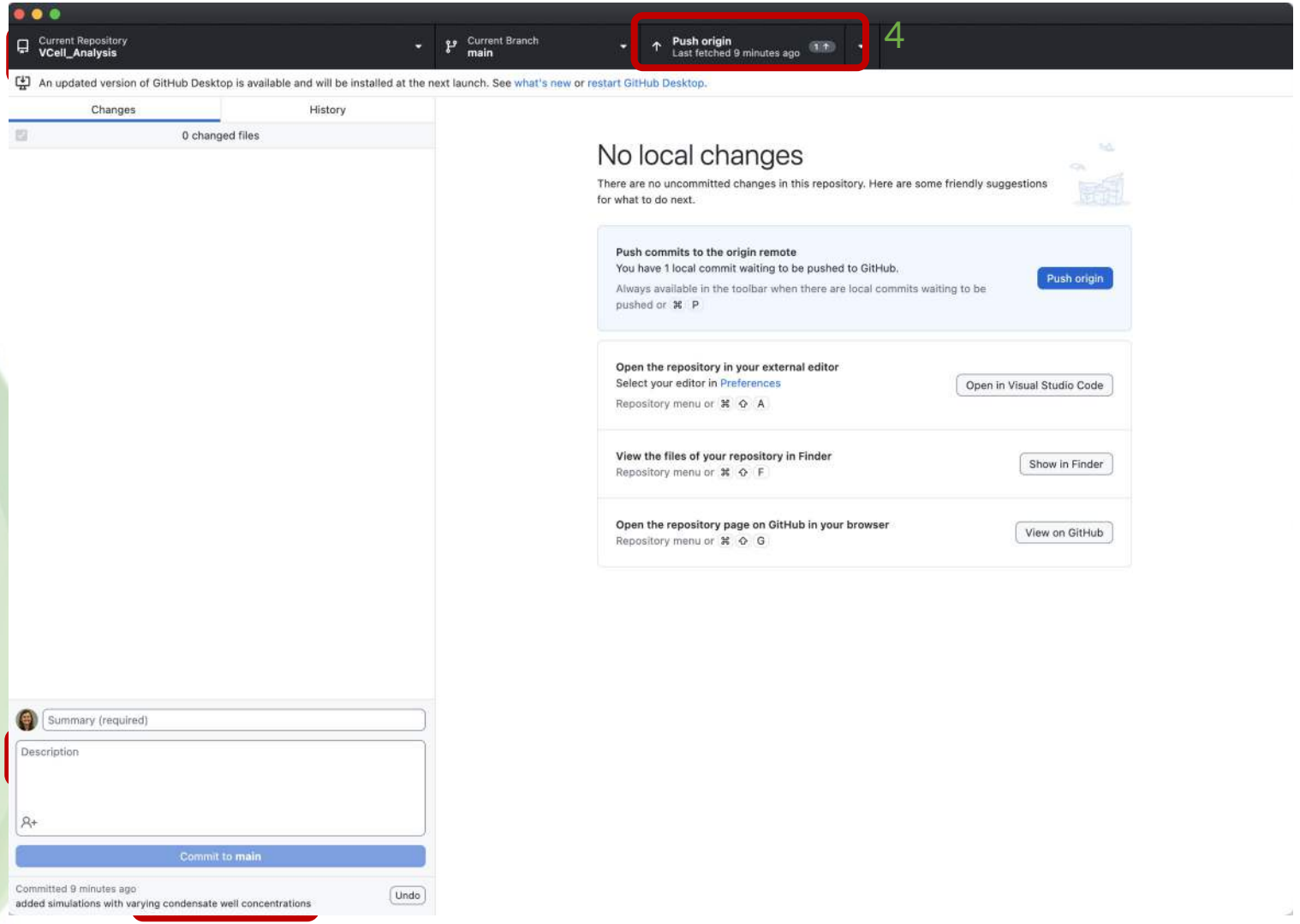
Commit to current branch

Alternative in terminal:

```
cd /Your/folder/for/this/class
git clone https://github.com/ENG-BME-2315-001/Module-0-LastName.git
git pull origin main
```



Getting set up with GitHub Desktop & git on command line



Alternative in terminal:

```
cd /Your/folder/for/this/class
git clone https://github.com/ENG-BME-2315-001/Module-0-LastName.git
git pull origin main
```

Change your code in VS code & save

```
git add .
git commit –m “message for commit”
git push origin main
```

Things needed today:
`python_practice.py` [last class]
Open github.com & log in

Thursday January 22

Coding Collaboratively with GitHub

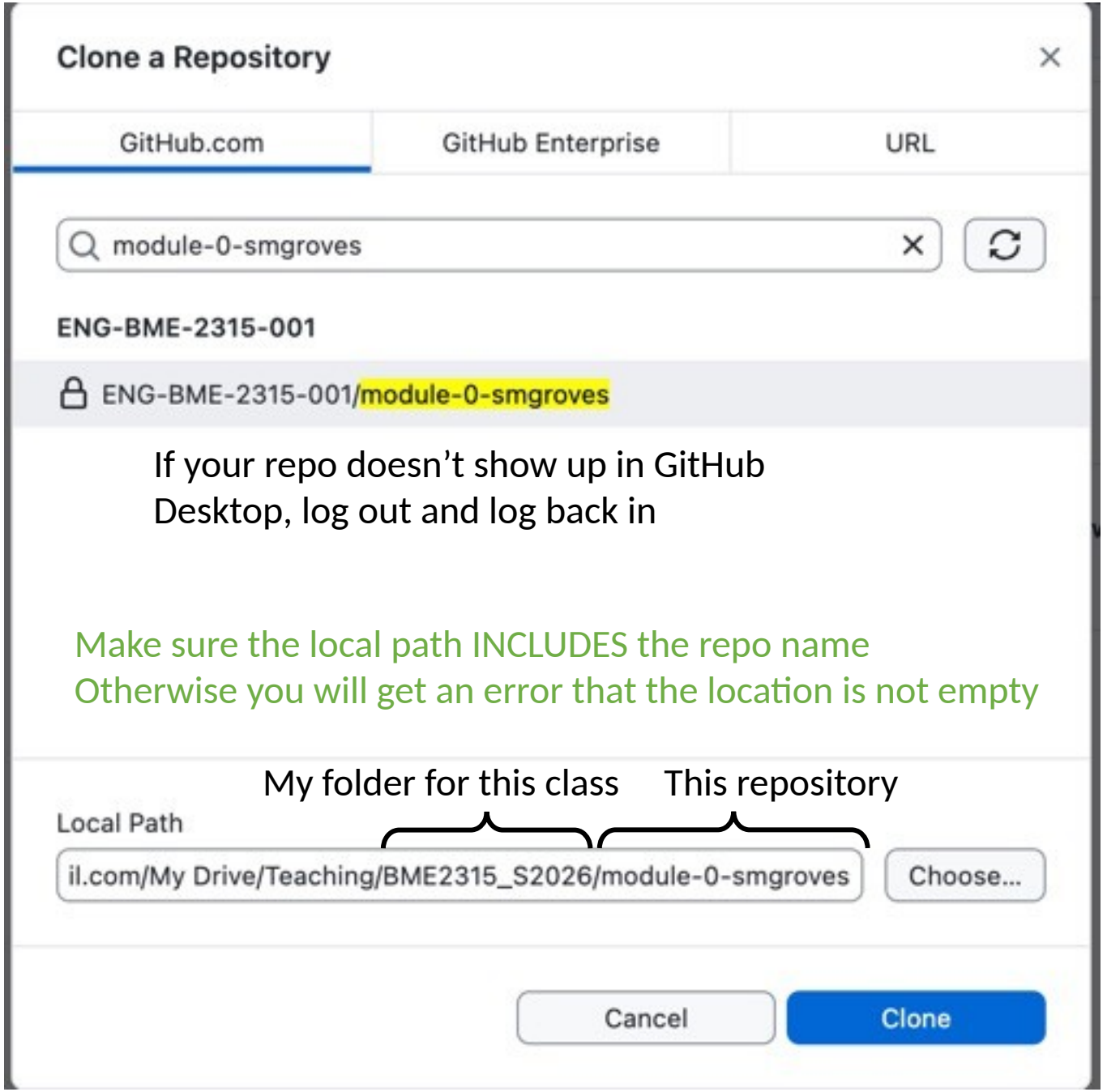
Dr. Groves

Dr. Peirce-Cottler

Module 0 Lecture 2

Learning objective: I know how to upload my code to GitHub, pull code from my partner, comment my code for future use, and write

Notes from last class



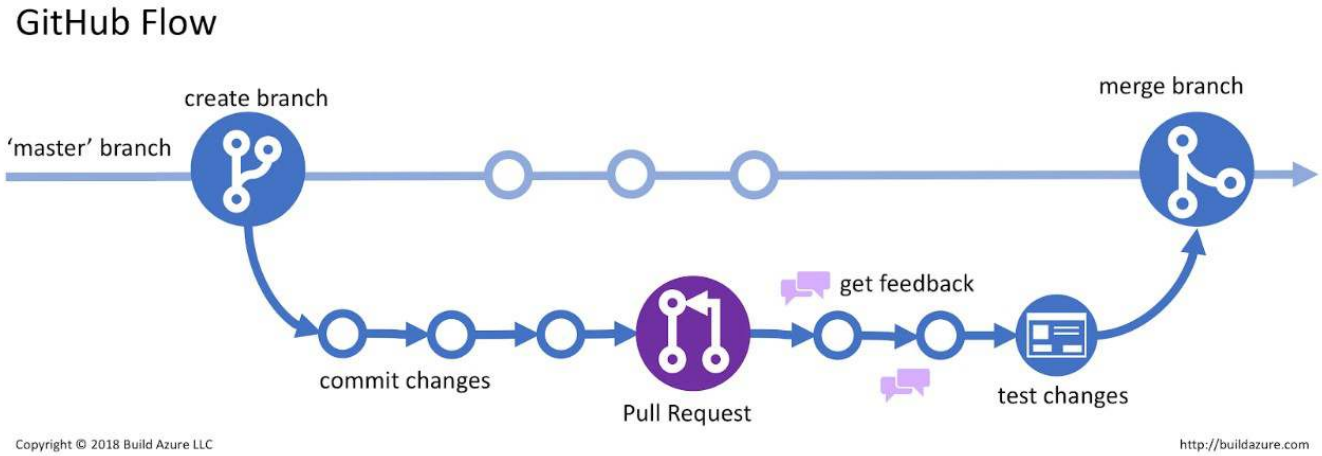
LET'S PRACTICE! Getting set up with GitHub Classroom

1. You should have a github account.
2. Join ENG-BME-2315-001 organization (from email). This will show up on your page under "Organizations"
3. Go to https://classroom.github.com/a/PHAtZR-G & find your name in the roster.
4. Accept Module_0 assignment (it should add your github username)
5. Find your repository by going to the ENG-BME-2315-001 organization (on your homepage) and click on the repository named "Module_0_ YourUserName" corresponding to your name.
6. Clone the repo to your local computer
7. Fetch origin/main
8. Open in VS Code
9. Update README, save, and push changes to the main branch

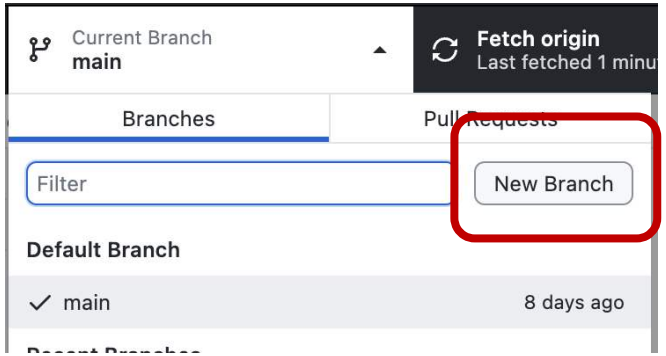
OR
- Add README.MD with your name/date in VS code & save
- Commit changes
- Publish branch

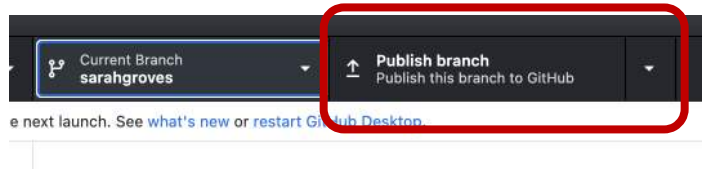
Stand your name card on its edge when you are done.
Help your neighbors!
If you need a TA/instructor, raise your hand!

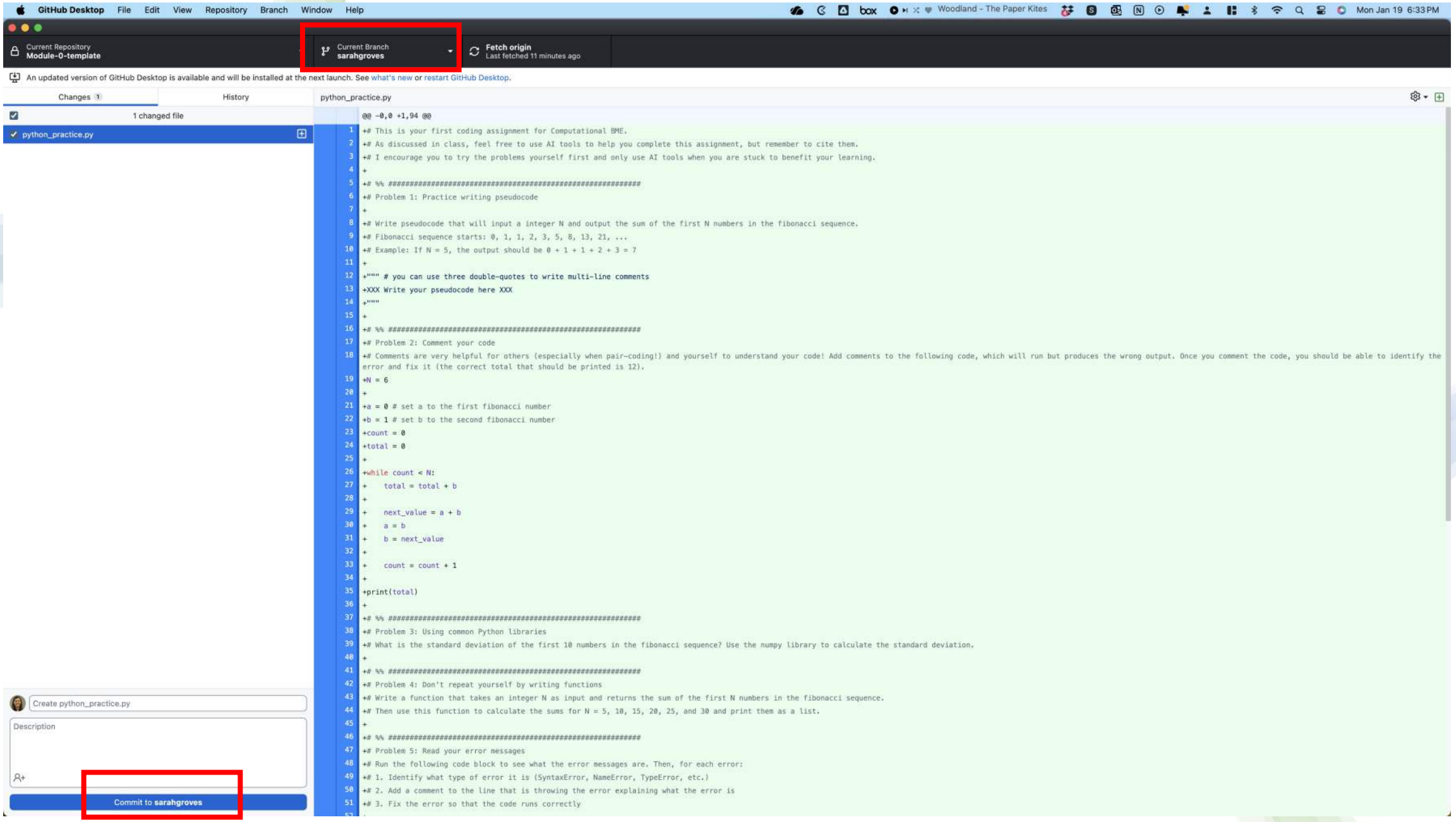
Pull Requests (PRs) and Merging: Recipe testing on a different branch



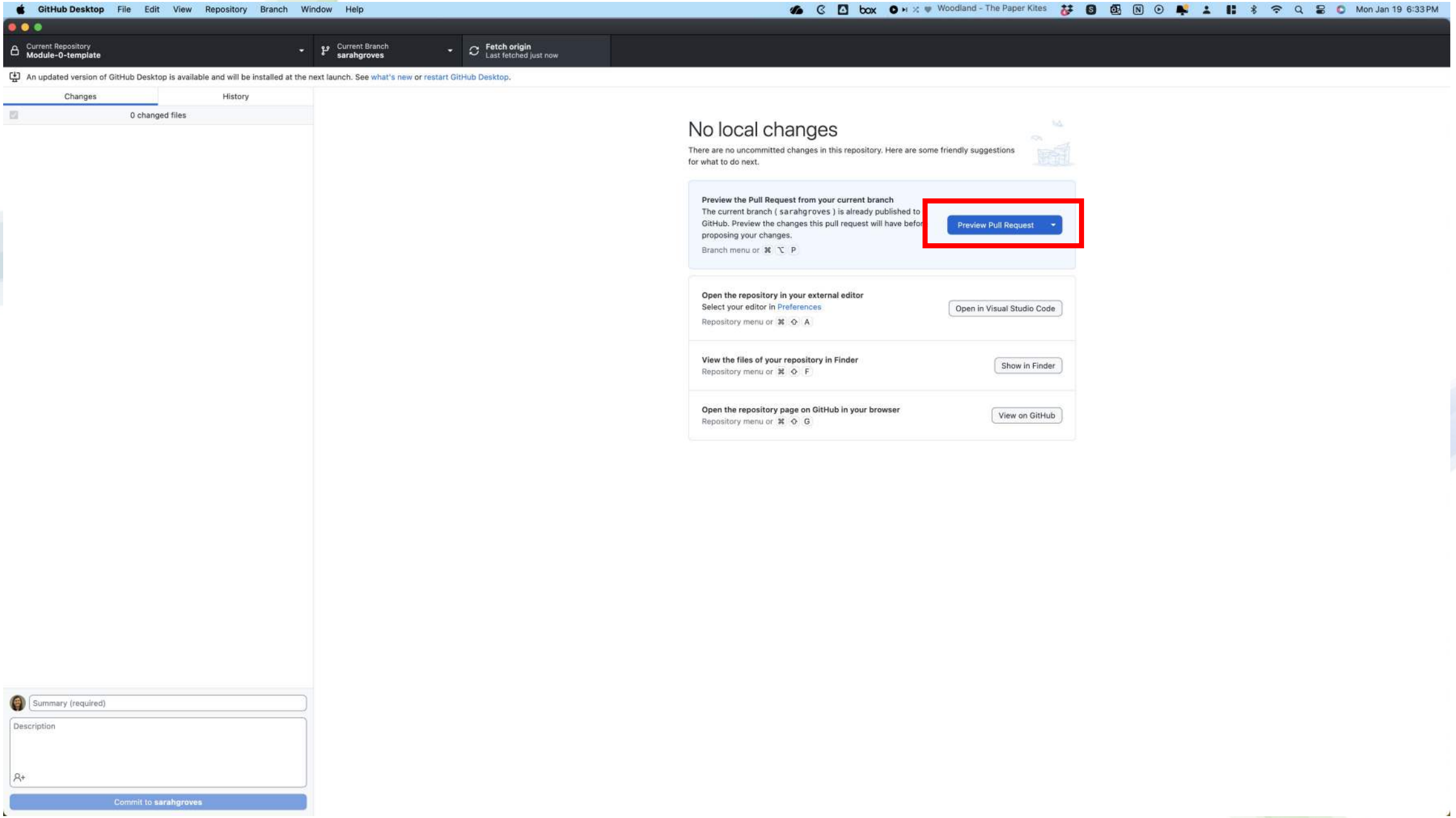
- Whichever branch is "checked out" (showing in GitHub Desktop) is where your edited code will go
- **Once done pushing to this branch**, you will make a "pull request" to merge with the main branch
  - Only merge when code is "done"
  - Don't delete your personal branch!



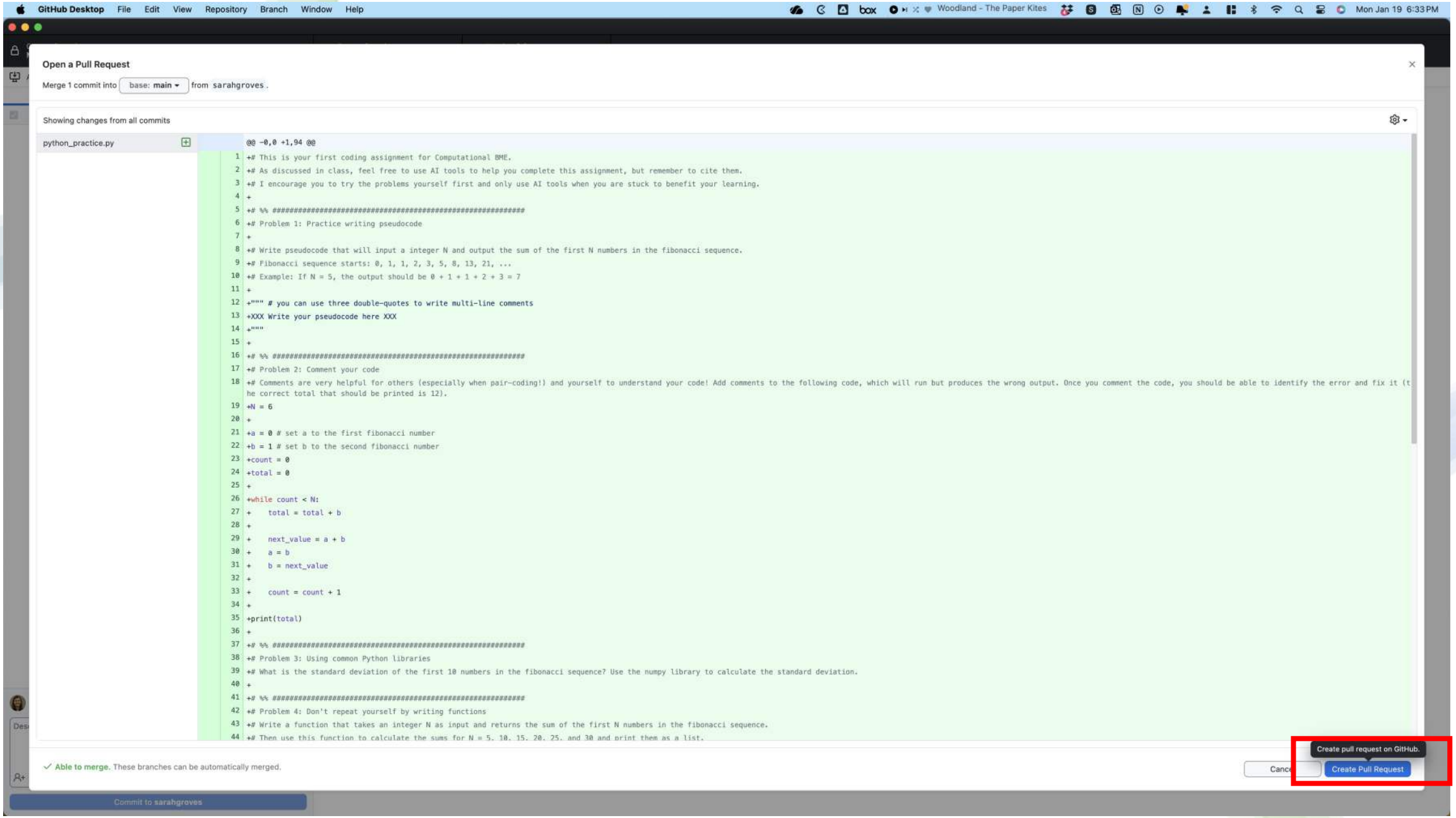




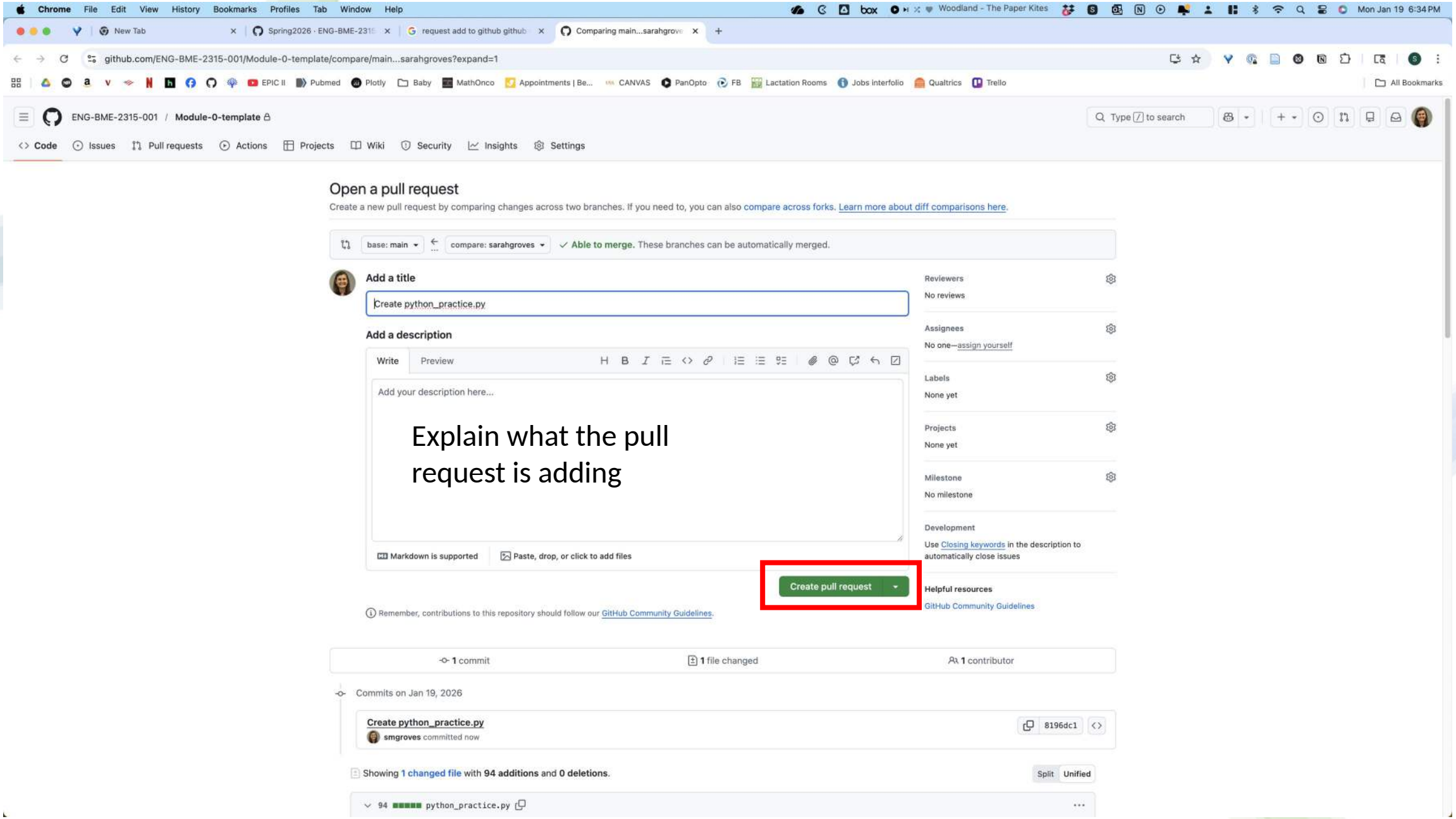
DO THIS (PULL REQUEST) WHEN CODE IS “DONE”



DO THIS (PULL REQUEST) WHEN CODE IS “DONE”

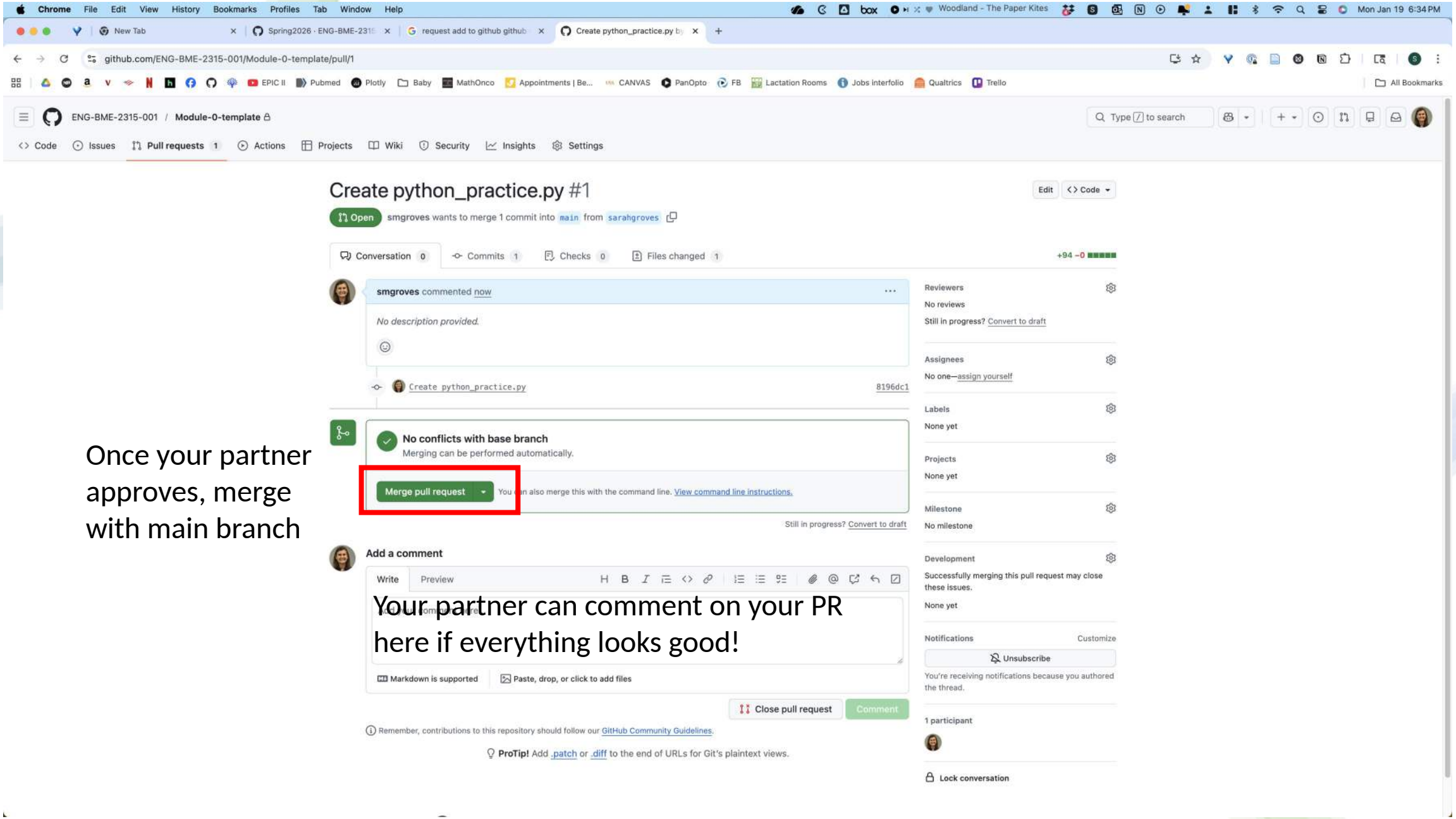


DO THIS (PULL REQUEST) WHEN CODE IS “DONE”



DO THIS (PULL REQUEST) WHEN CODE IS “DONE”

Once your partner approves, merge with main branch



Your partner can comment on your PR here if everything looks good!

DO THIS (PULL REQUEST) WHEN CODE IS “DONE”

How will we use GitHub in this class?

- Each partner should code in their own python files (.py) that are pushed to GitHub regularly.
- Jupyter Notebooks for final report can be worked on throughout the Module, but only final code should go there.
  - Notebooks should import functions from .py files
- GitHub repository structure:
  - One repo for each project
  - Shared with your partner for that module
  - Each partner should have their own branch and regularly make pull requests that merge the code in their branch with the main branch

Let’s Practice!

Make a new branch in GitHub Desktop and switch to this branch.

Add your file from last class to your repo folder (locally) on your own branch.

Then commit & push, make a pull request, and merge changes to the main branch.

Stand your name card on its edge when you are done. Help your neighbors! If you need a TA/instructor, raise your hand!

General protocol for EVERY coding session

PULL FROM REMOTE

OPEN FOLDER LOCALLY

CODE & SAVE

COMMIT CHANGES, PUSH TO REMOTE

When you're happy with your code, open a PULL

When can GitHub be problematic?

Problem	Advice
When you work on the same code in the same file at the same time as another person	- As a general rule, do not work on the same file at the same time - Alternatively, always work on different files and push and pull back and forth to share code - TALK TO YOUR PARTNER WHEN YOU/THEY ARE CODING
When you delete a file some one else is working on	- Don't delete files without talking to your partner first - Don't move files around without talking to your partner first
When you add a file with the same name that someone else has, and try to push them both to the remote branch	- Include your first or last name in your own file names - For shared files (like your final notebook), do not try to make duplicates - Add clear indications to the file name for when code should be used or if it is test code or unused
When you forget to ignore certain files that you really don't care about	- Add things to .gitignore to automatically "lose track" of them
When you are trying to commit a file > GitHub's file size limit (100Mb)	- These files must remain locally on each partner's computer to be used. You can share them through box. - Add these files to .gitignore so you never accidentally push them.
When you test code that doesn't work, but now you can't get back to what was working	- Work on the outside of the github repo then copy/paste working code into the repo - Make a brand-new file (e.g. test_lastname.py) and only change code in that file, then copy/paste when it's working into the repo - Make a branch in your repository for test code - Worst case scenario: git reset (advanced, ask for help if you haven't done this

Criteria Builder

100 Points Possible

1. **Repo exists** — 5 pts
   You should have a repo set up through GitHub classroom named "Module-0-YourUserName."
   > Rating Scale: 2

2. **README** — 5 pts
   The README has been updated.
   > Rating Scale: 2

3. **Problem 1 Completed** — 10 pts
   Problem 1 in python_practice.py has been completed. If AI was used, it has been cited
   > Rating Scale: 3

4. **Problem 2 Completed** — 10 pts
   Problem 2 in python_practice.py has been completed. If AI was used, it has been cited
   > Rating Scale: 3

5. **Problem 3 Completed** — 10 pts
   Problem 3 in python_practice.py has been completed. If AI was used, it has been cited
   > Rating Scale: 3

6. **Problem 4 Completed** — 10 pts
   Problem 4 in python_practice.py has been completed. If AI was used, it has been cited
   > Rating Scale: 3

7. **Problem 5 Completed** — 10 pts
   Problem 5 in python_practice.py has been completed. If AI was used, it has been cited

8. **Problem 6 Completed** — 10 pts

Problem 6 in python_practice.py has been completed. If AI was used, it has been cited

> Rating Scale: 3

9. **Jupyter notebook** — 10 pts

Jupyter notebook has been added to the repository. The student has edited the notebook such that it looks like the "good notebook" from class, with clear comments and notes in Markdown, separate code blocks, plots, and organization using headers. The notebook should not print out excessive output that isn't user to the reader.

> Rating Scale: 5

10. **Pushing to your branch** — 10 pts

A GitHub branch has been created (besides main). Regular pushing to this branch with appropriate comments is evident (rather than a single push of all the code right before the due date).

> Rating Scale: 3

11. **Pull request & merge with main branch** — 10 pts

Student has merged their branch into the main branch once complete. (Student should not delete their secondary branch). The pull request has been commented.

> Rating Scale: 3