GitHub Classroom in the Classroom

David Singletary David.Singletary@fscj.edu
William Money s2735014@students.fscj.edu

This material is based upon work supported by the National Science Foundation under Grant No. 1902524.

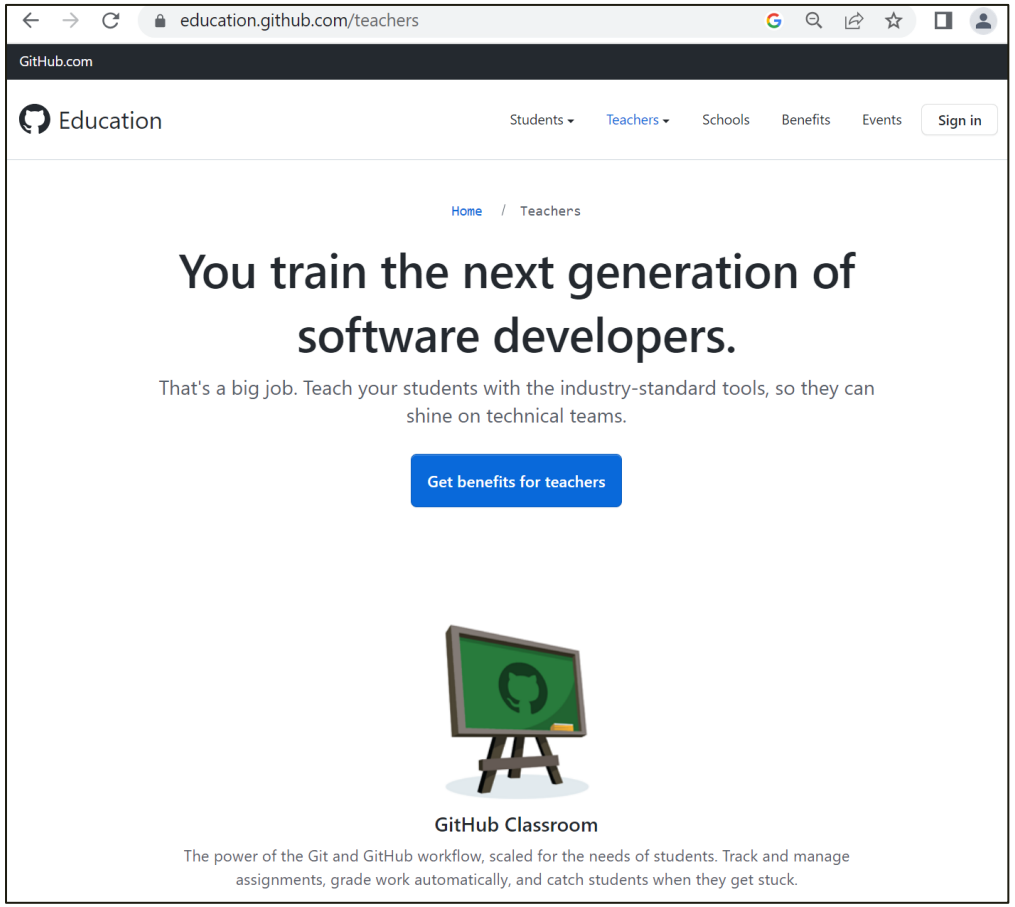
FSCJ Florida State College at Jacksonville

GitHub Classroom

- The presentation provides information for using GitHub Classroom, which allows the creation of individual classrooms and assignments in the context of GitHub
- Why GitHub?
  - GitHub (and Git) are critical components in maintaining versioned repositories of work in data science, software development, and other areas
  - Using these tools is an important job skill that students should be familiar with
- Why GitHub Classroom?
  - GitHub Classroom enables instructors to assign and assess individual work while simultaneously providing students with hands-on experience with GitHub
  - Assignment repositories created by the instructor can be cloned by students and maintained within the classroom
  - Cloned student repositories actually belong to the instructor and are private by default

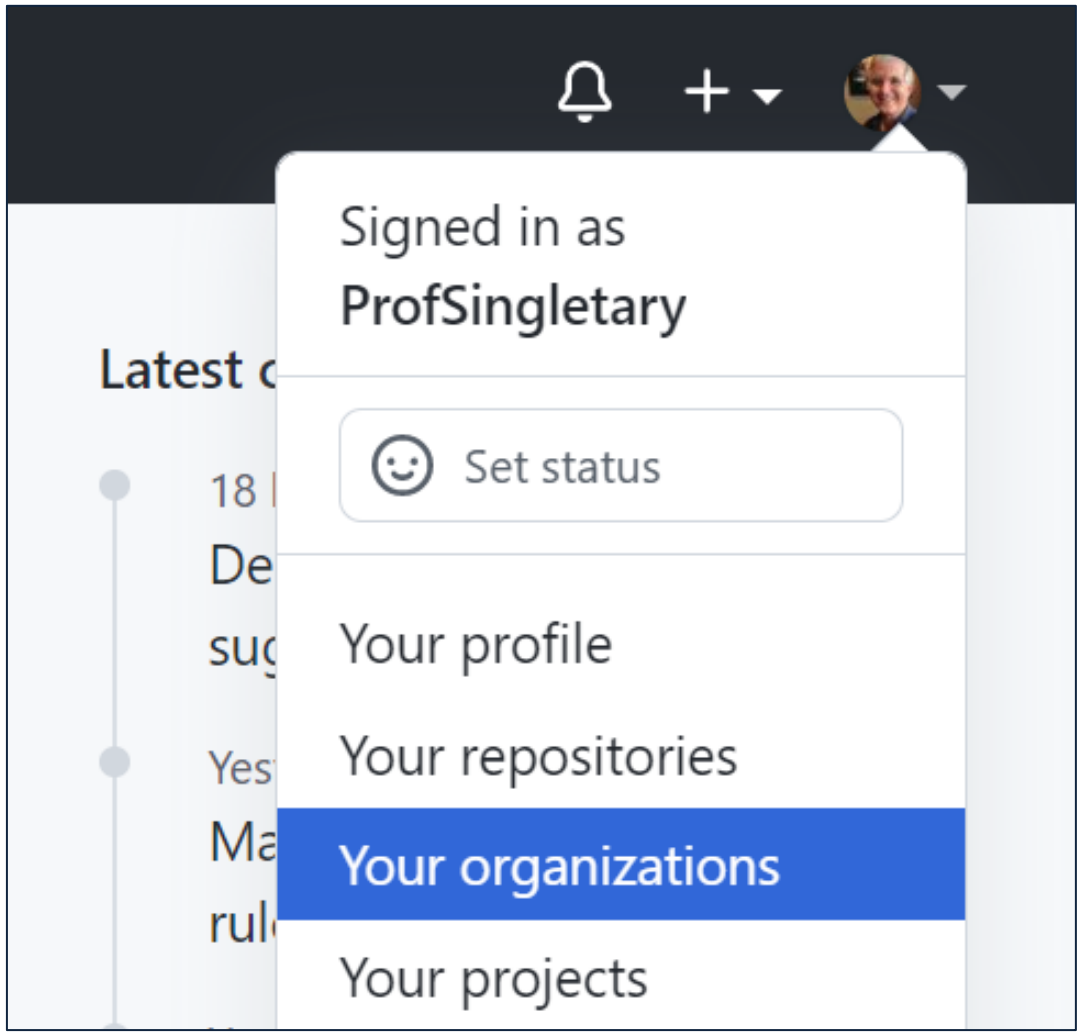
Start with an Instructor Account on GitHub Education

- Apply for an instructor account on GitHub Education
  - (https://education.github.com/teachers)
  - account verification may take a day or two



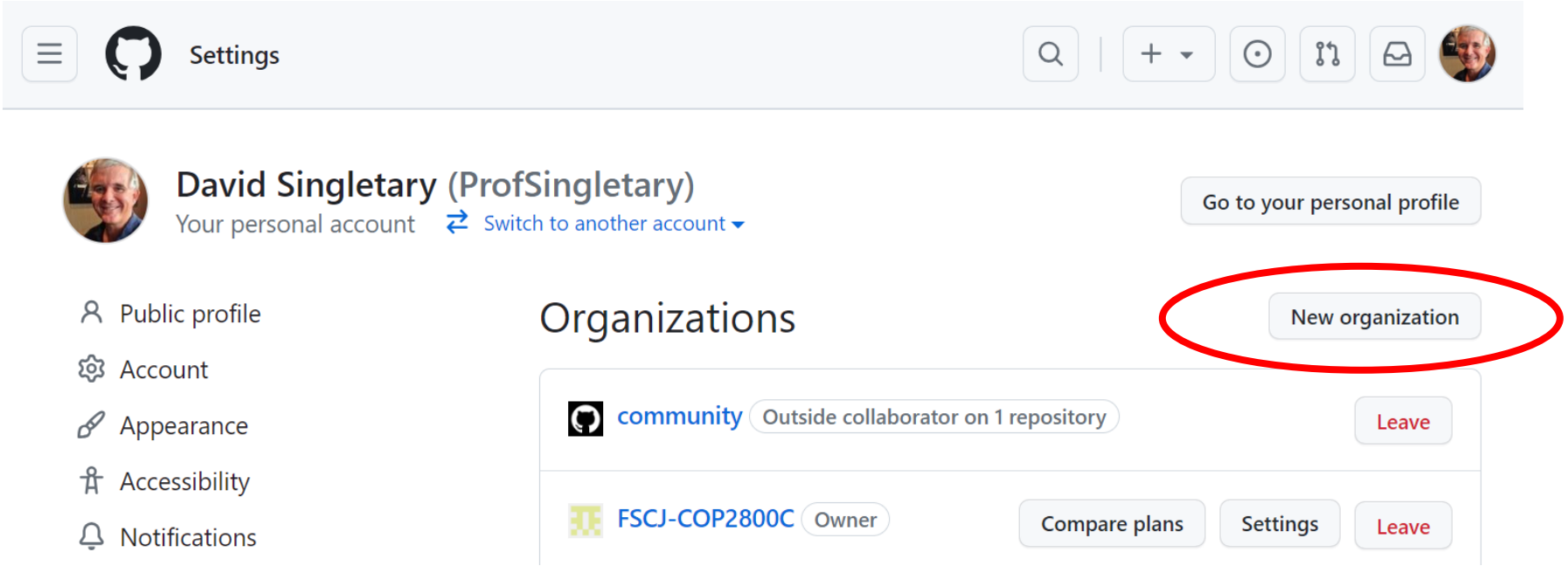
Create an Organization in GitHub

- Organizations can be used to organize your courses (e.g. Programming, Analytics, Networking, etc.
- Select "Your organizations" from your profile menu in GitHub



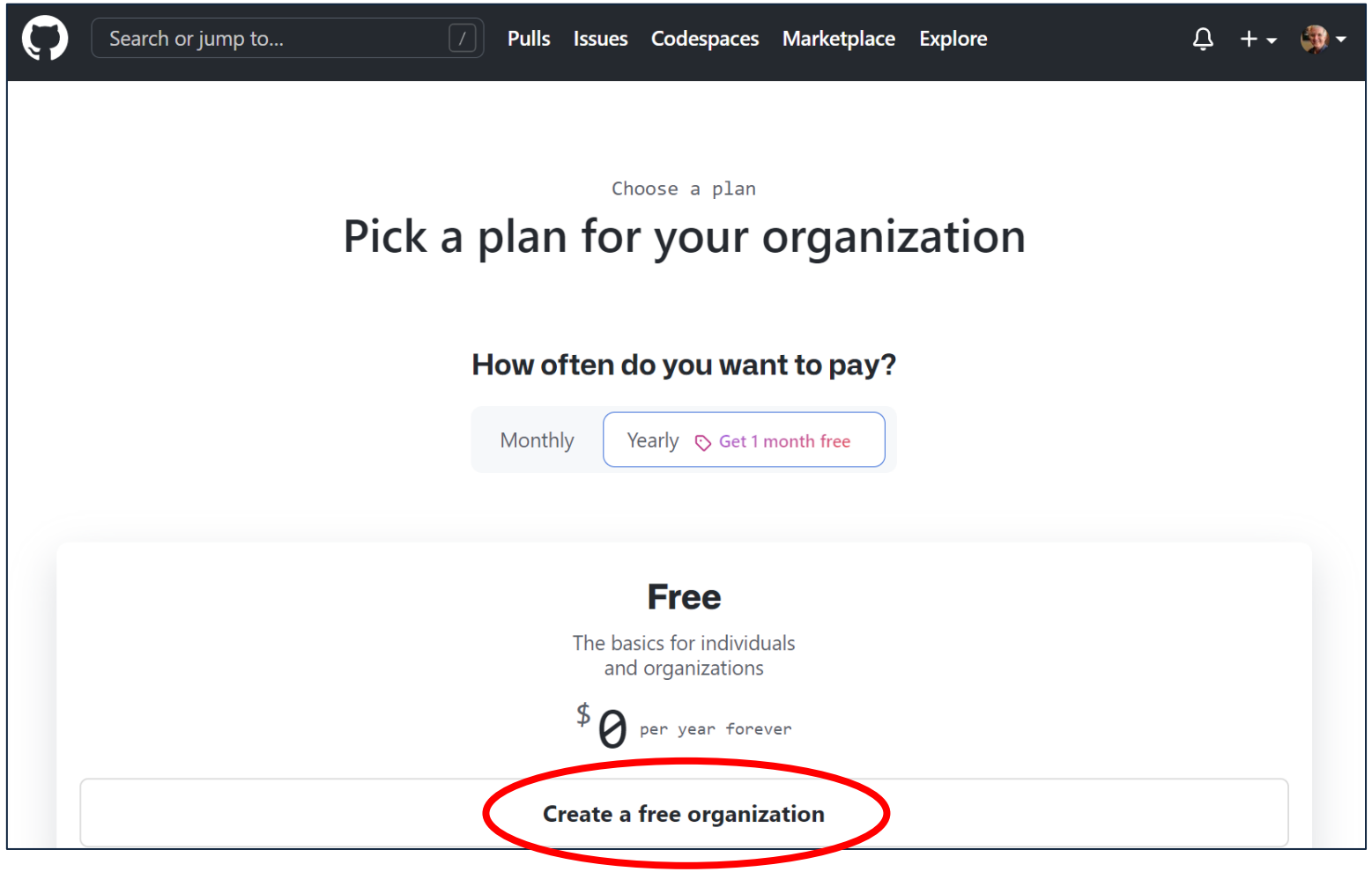
Creating an Organization

- Select "New organization" from the page

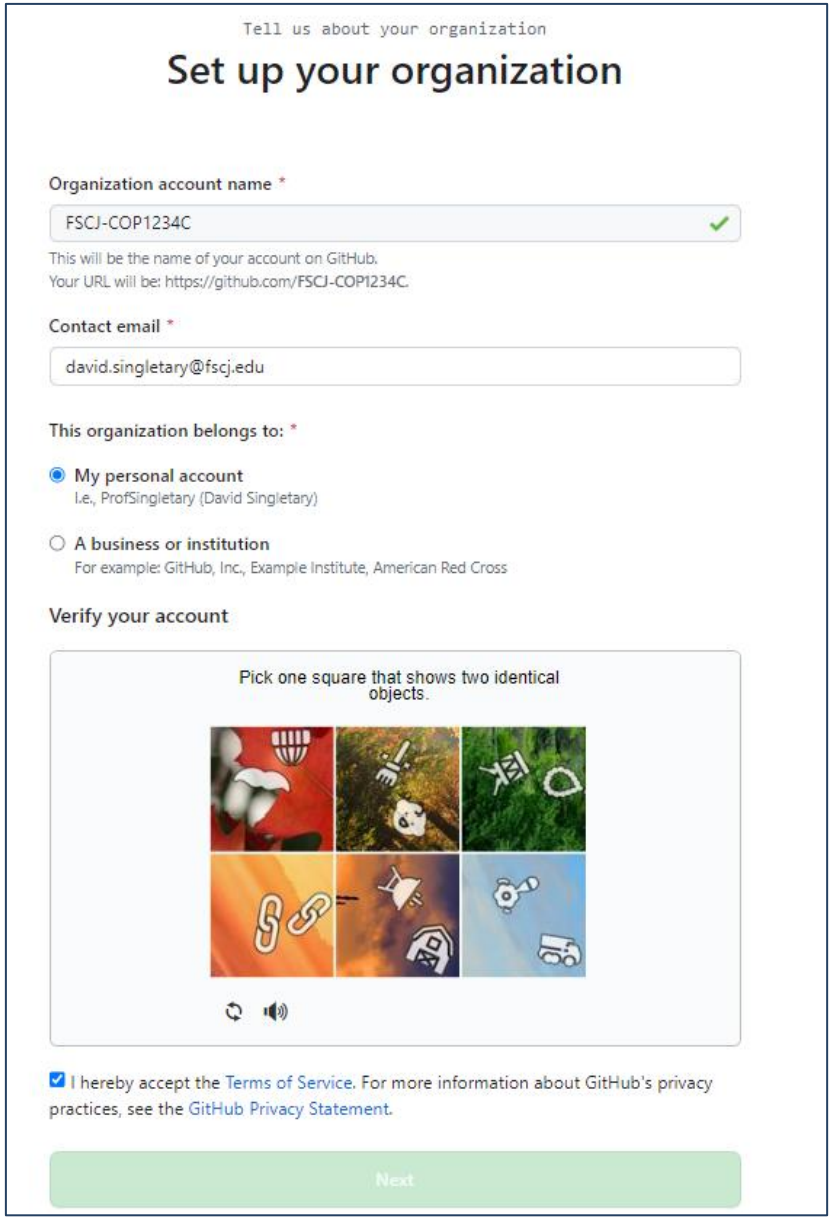


Creating an Organization

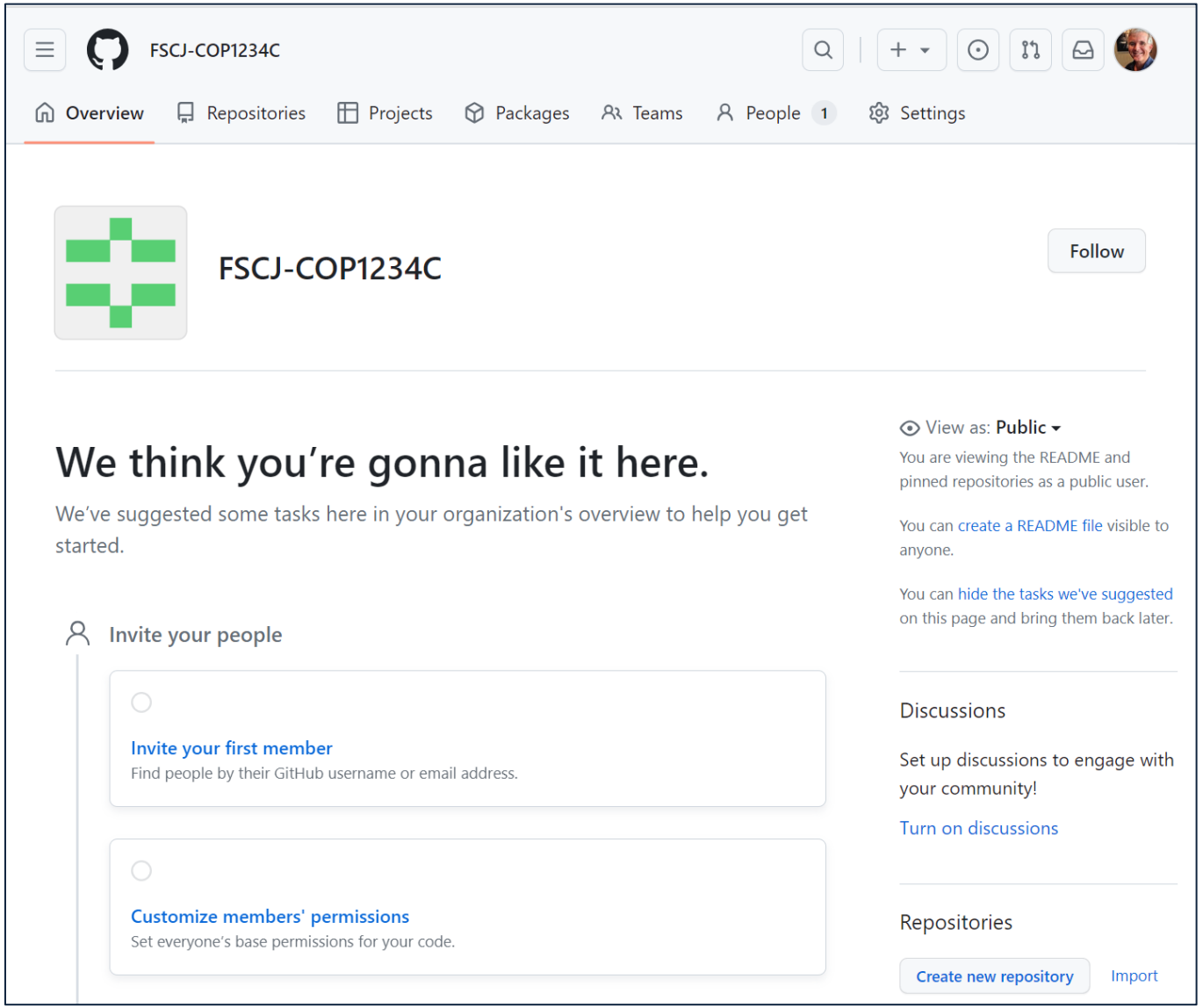
- Choose "Create a free organization"



Creating an Organization

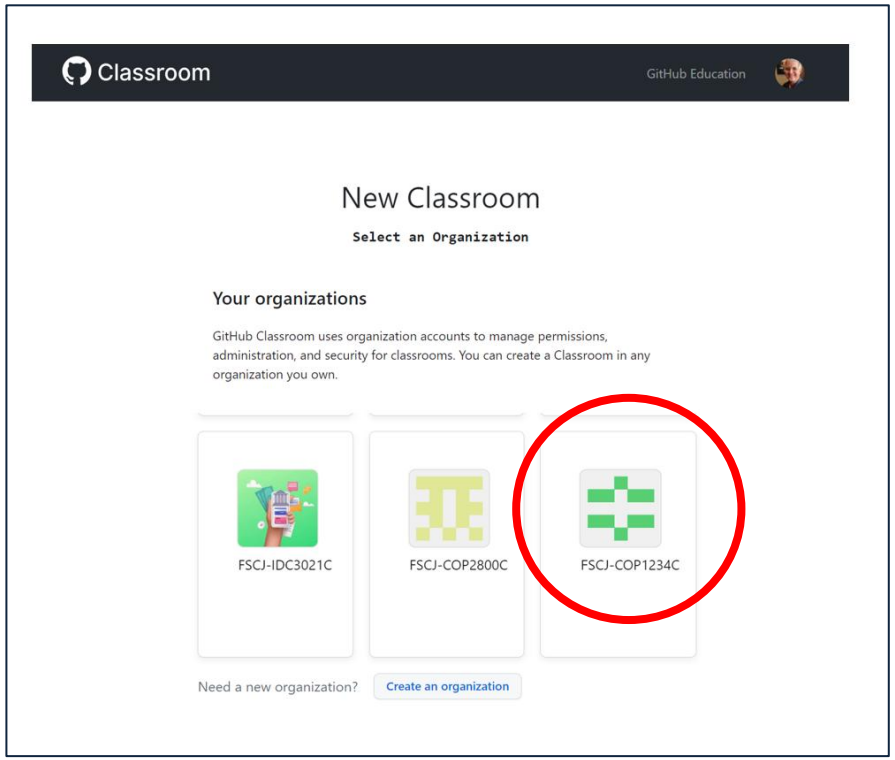


Creating an Organization



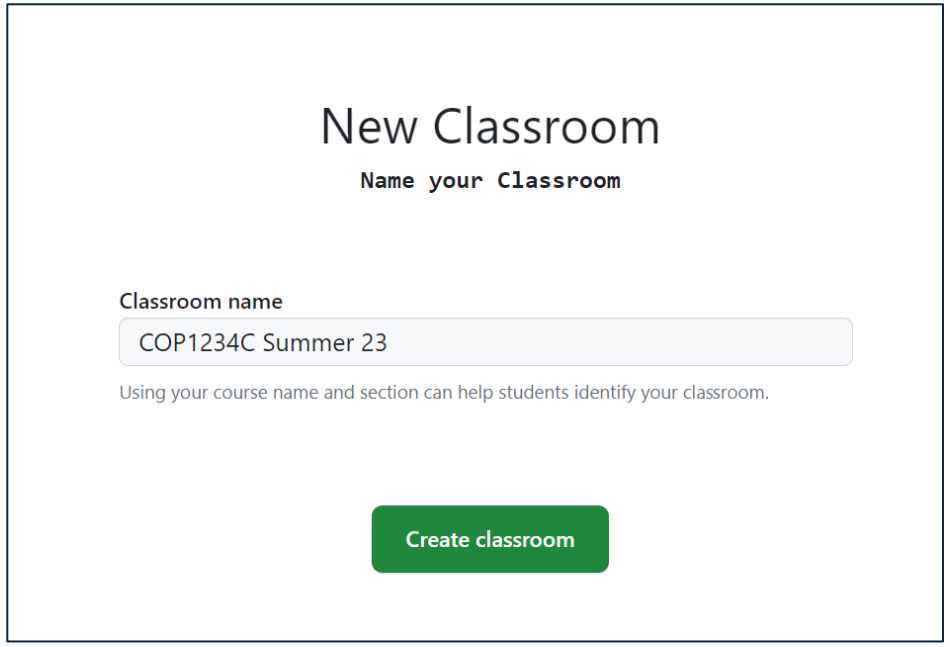
Creating a Classroom

- In GitHub Classroom (https://classroom.github.com), select the desired organization for your classroom



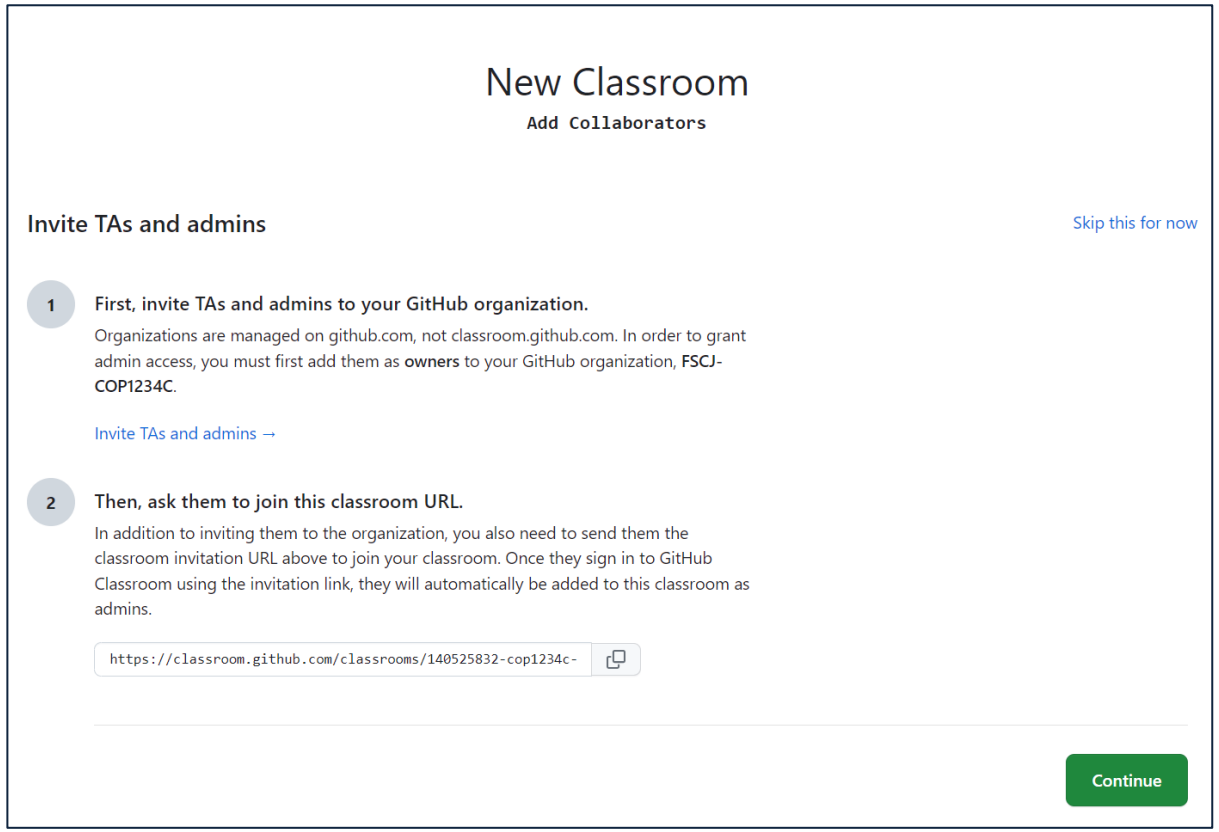
Creating a Classroom

- Name your classroom



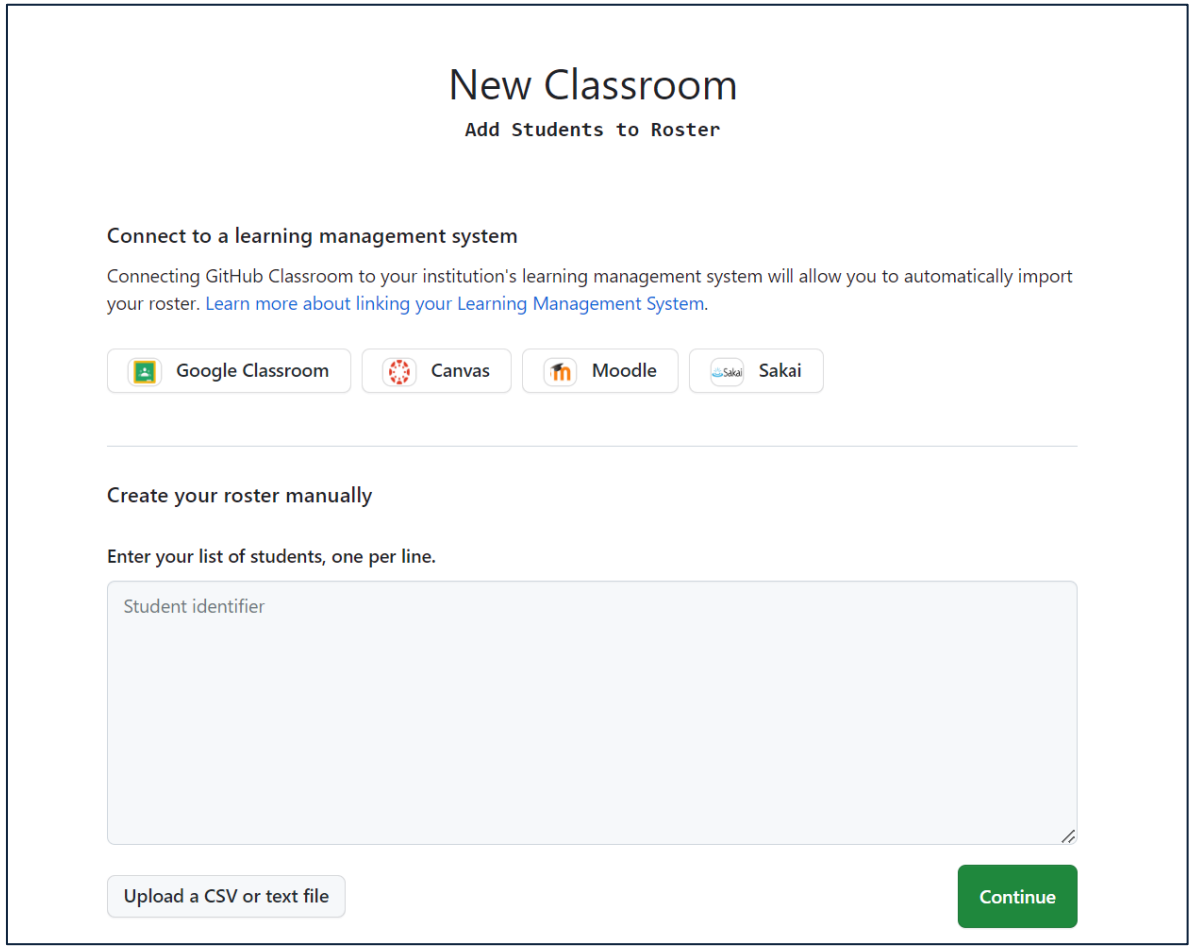
Creating a Classroom

- TAs or Admins?



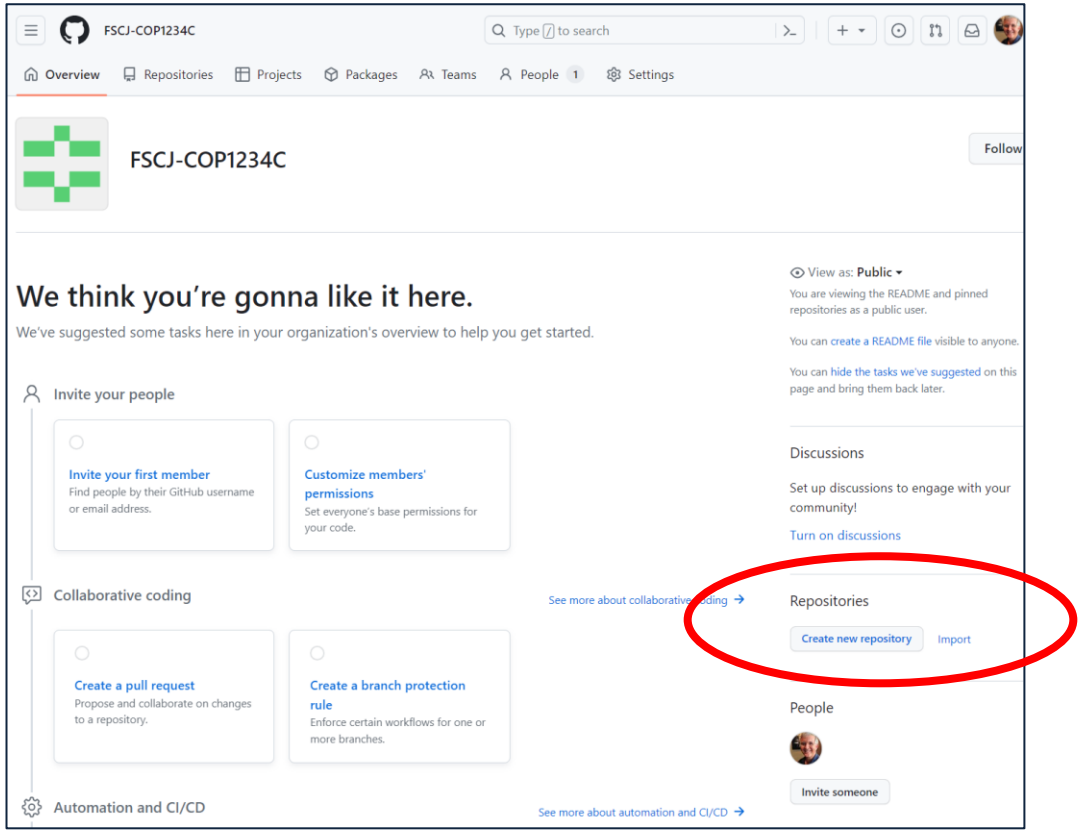
Creating a Classroom

- Add Students
- Note: this step is not required; an alternative approach is to let your students add themselves by accepting your assignment invitations



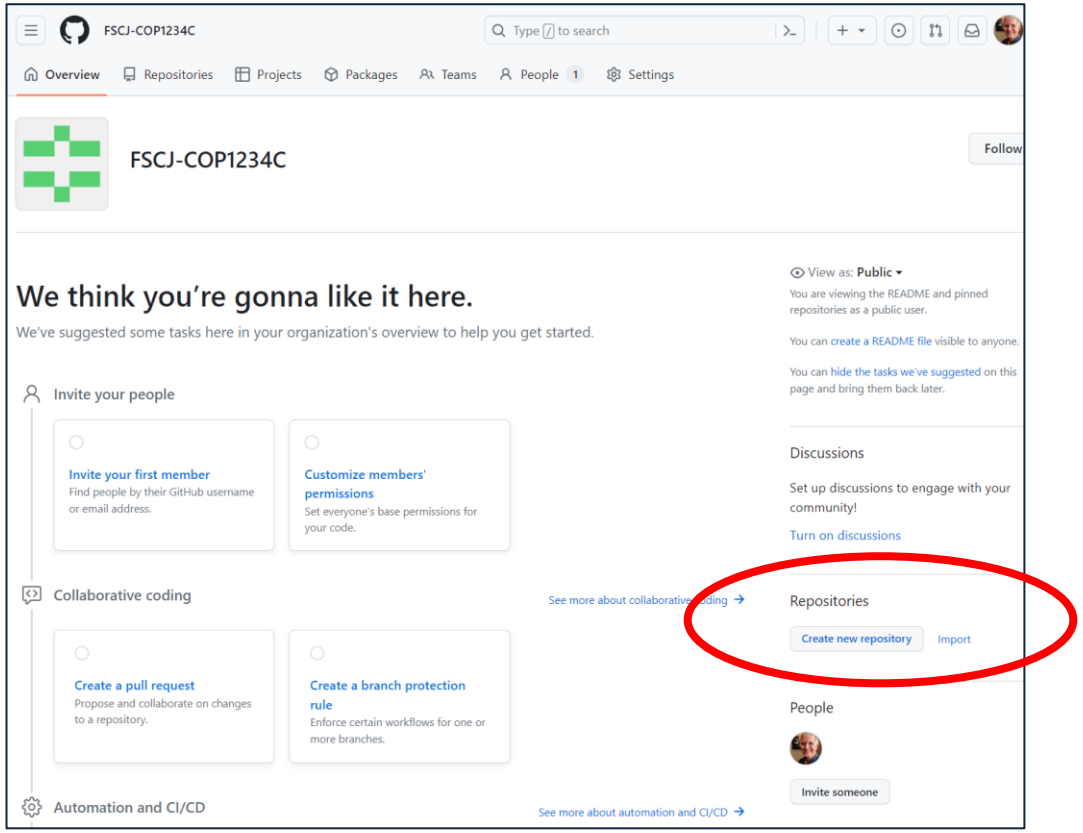
Back in GitHub: Create an Assignment Template

- Working in your organization, create a new repository



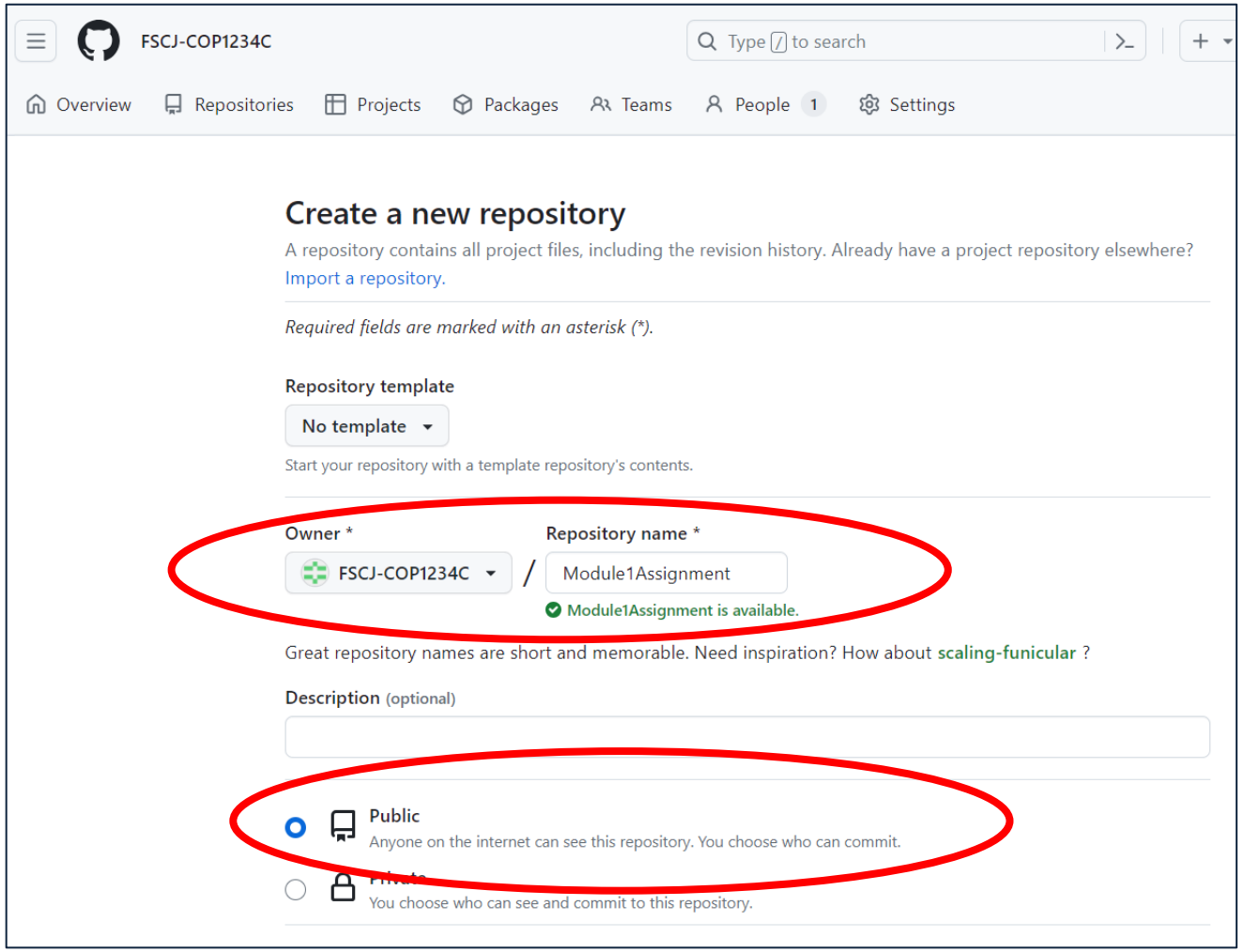
Back in GitHub: Create an Assignment Template

- Working in your organization, create a new repository



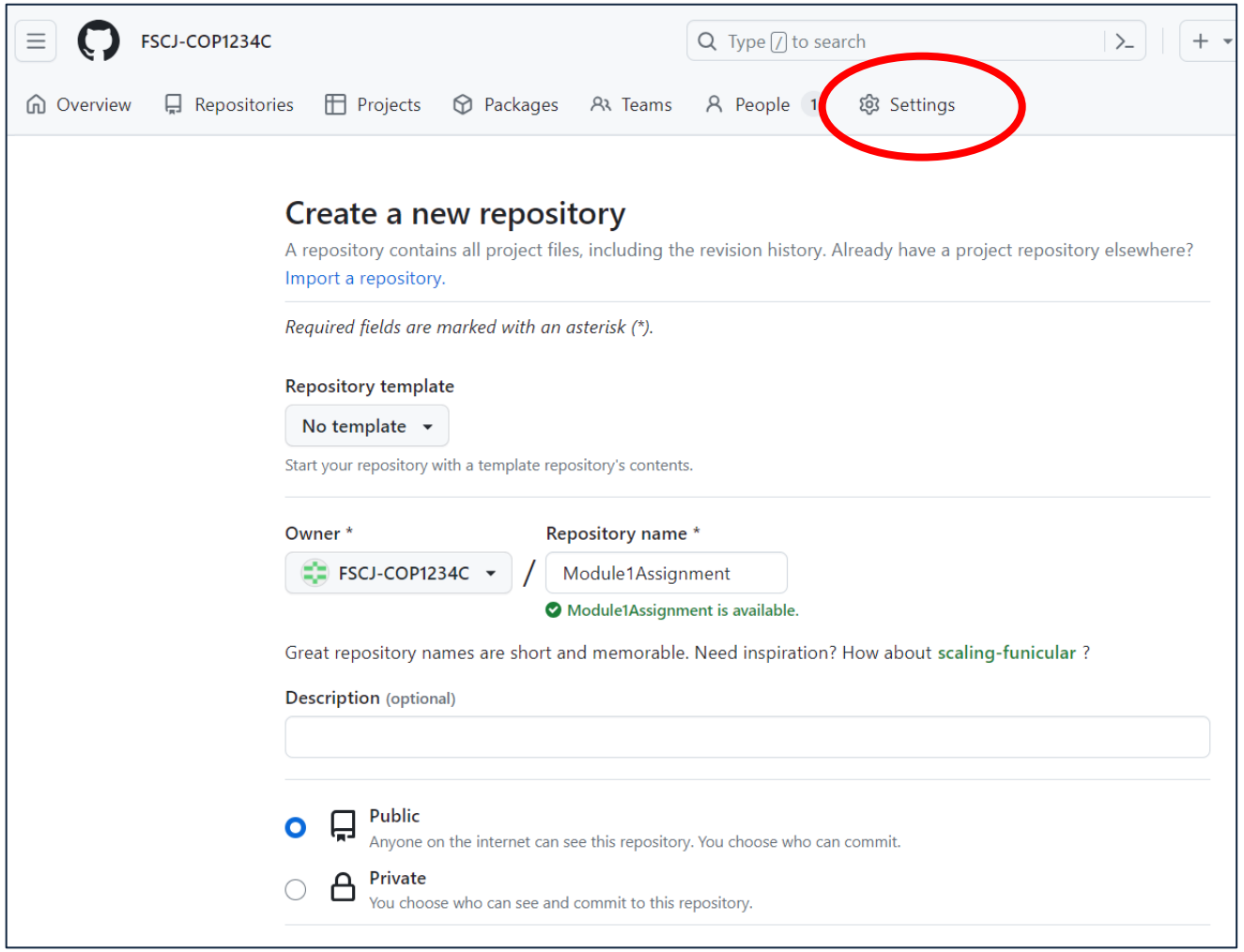
Creating Assignments

- Name the repo and set it to Public



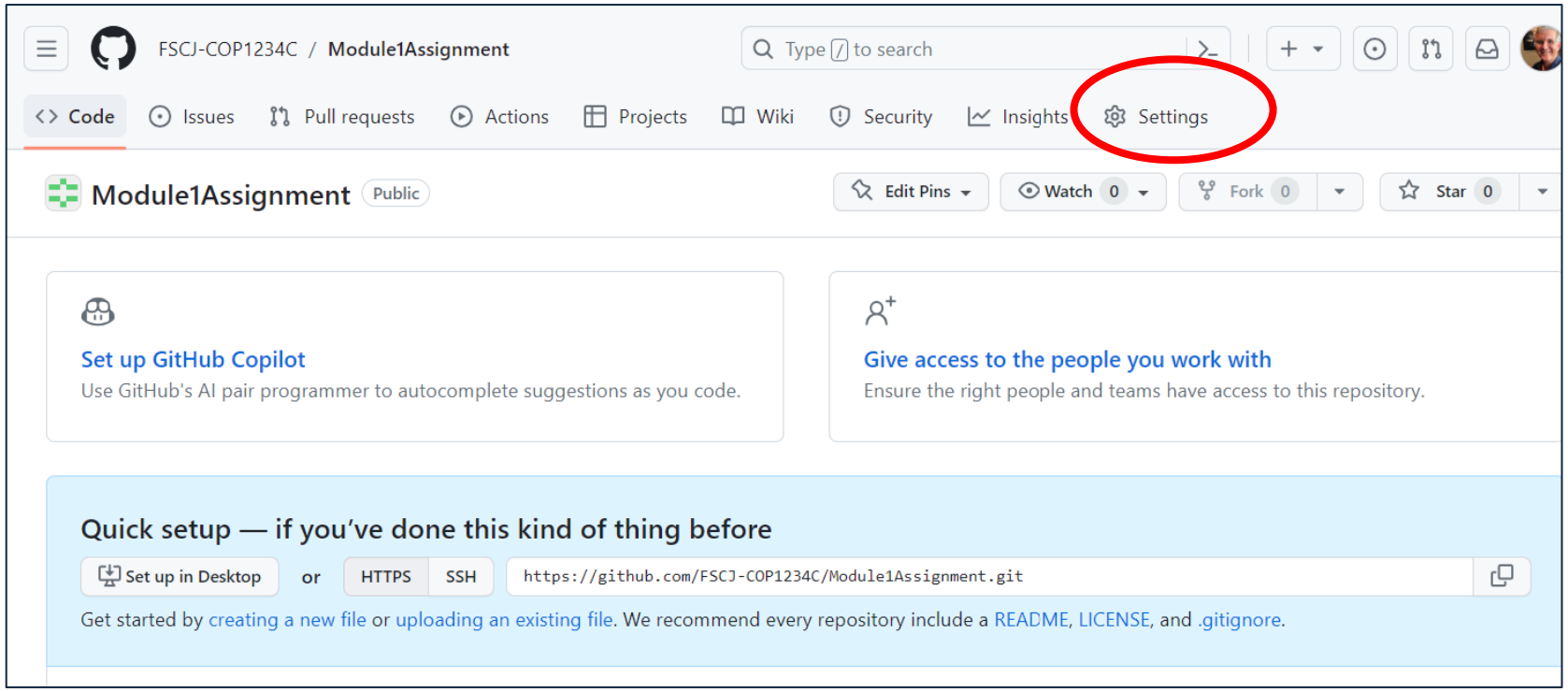
Creating Assignments

- The repo needs to be a template:
- Select the Settings tab



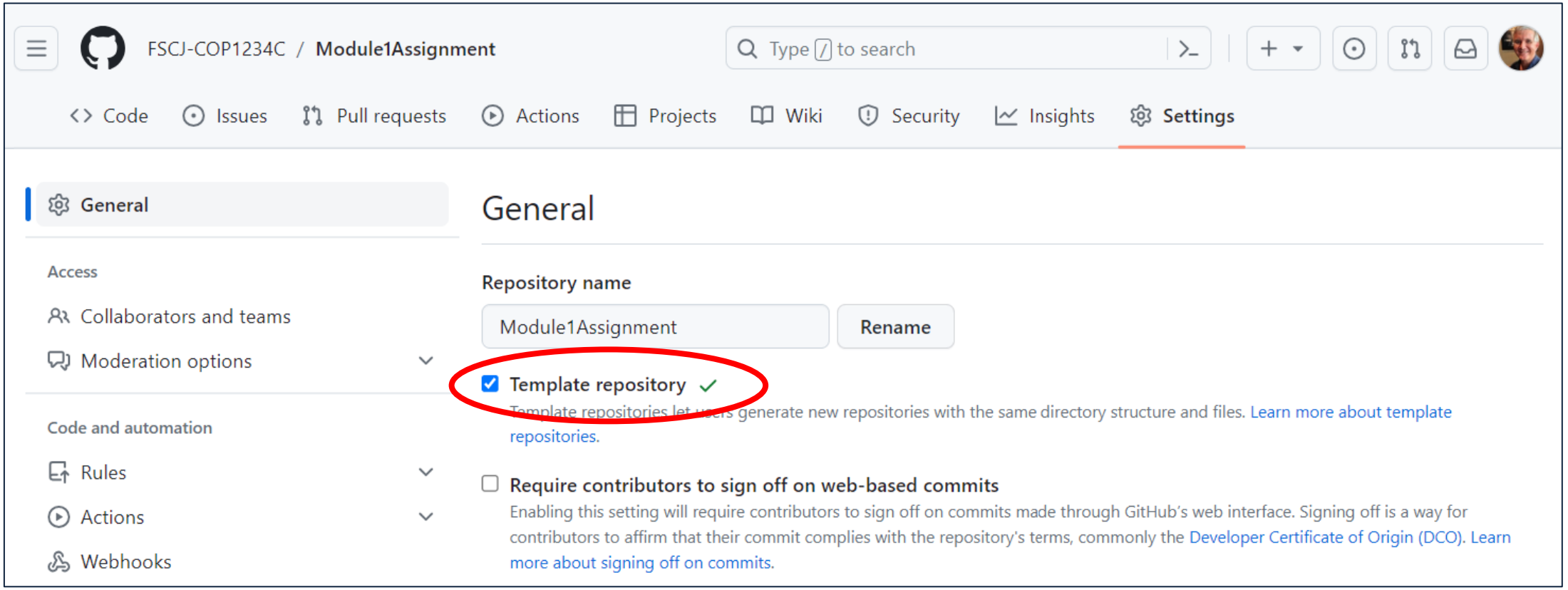
Creating Assignments

- The repo needs to be a template
- After creating the repo, select the Settings tab



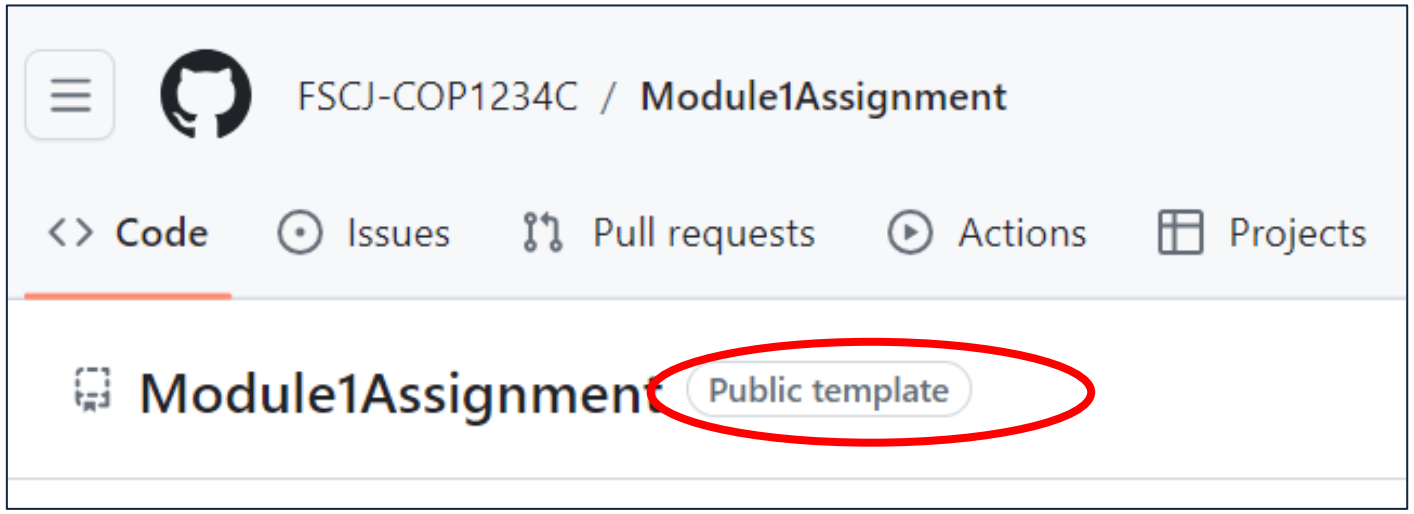
Creating Assignments

- On the Settings page, check the Template repository box



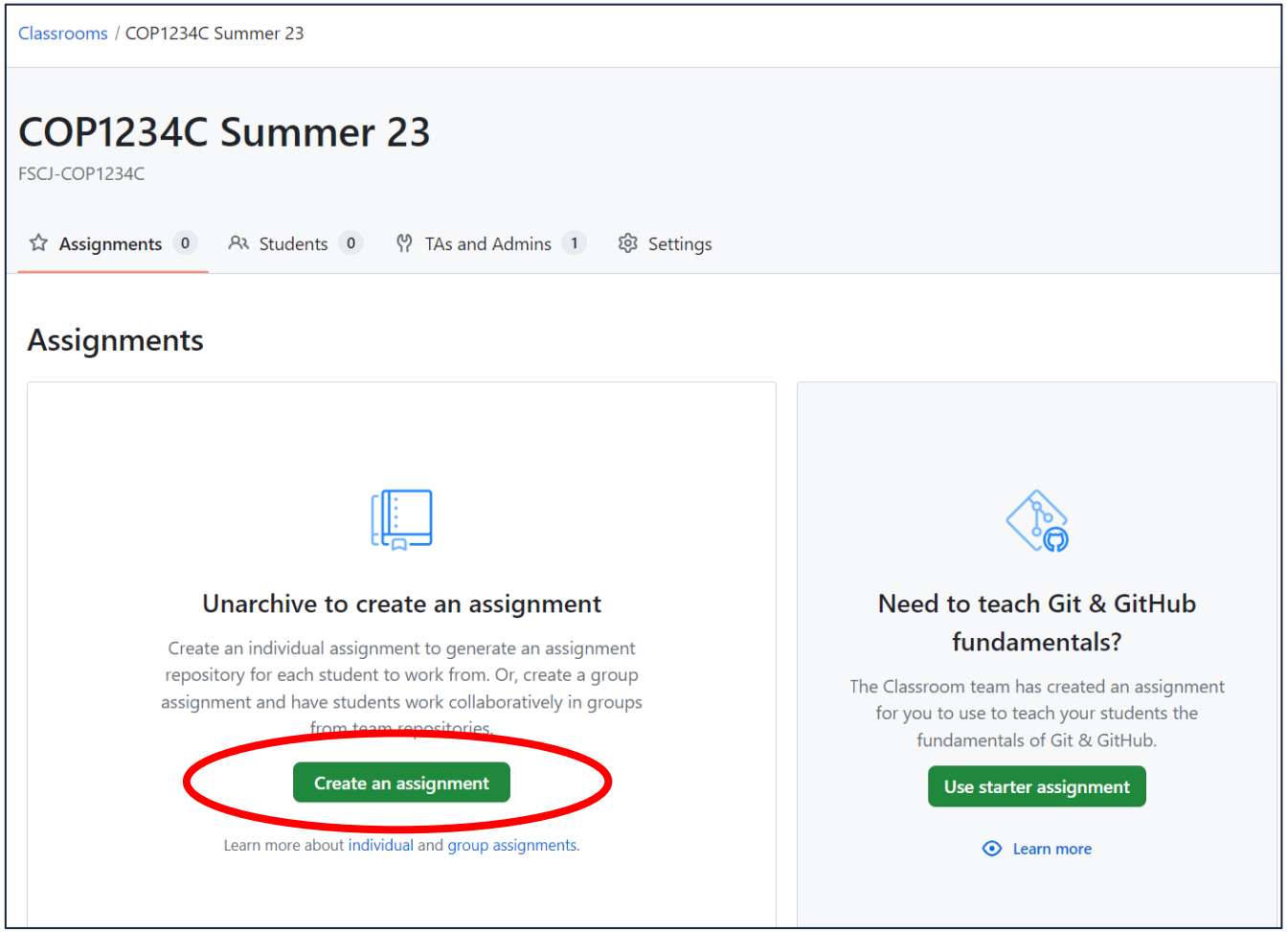
Creating Assignments

- Note: there is no "OK", "Submit", or "Commit" button – just checking the Template repository box commits the setting
- Return to the repo home page and refresh the browser to verify the repo is a "Public template"



Creating Assignments

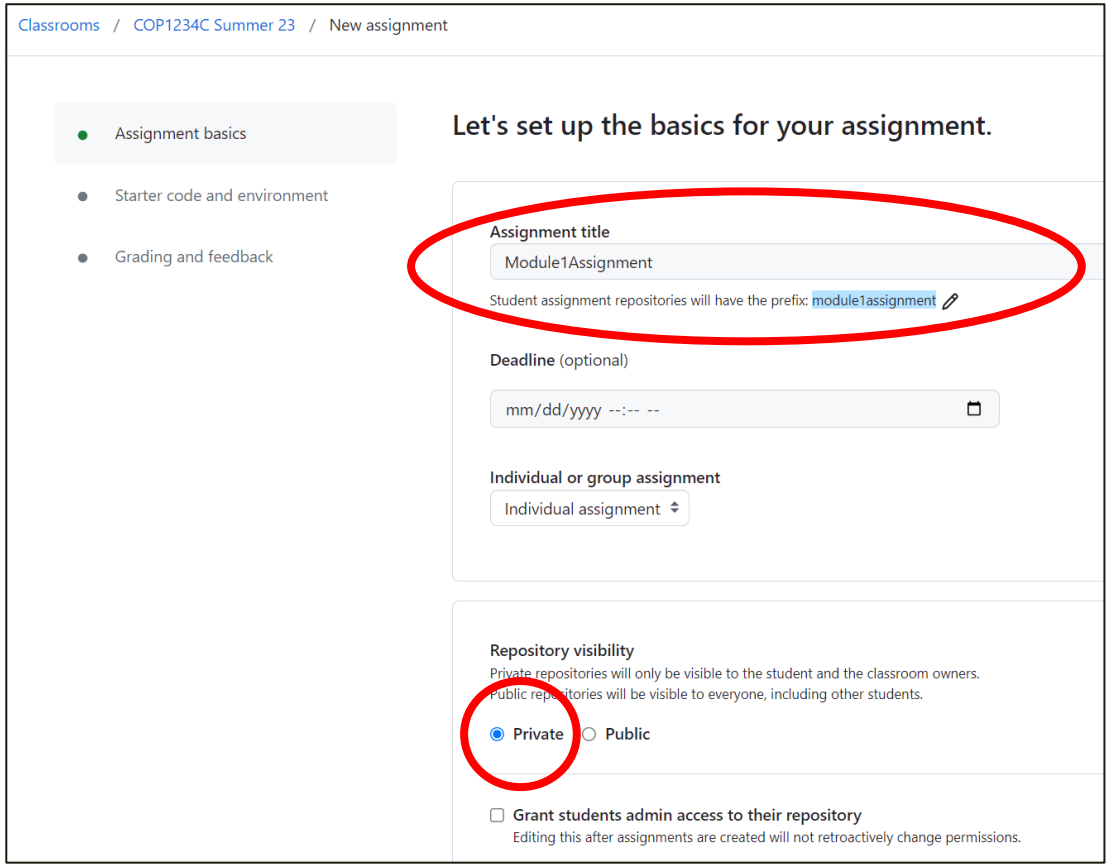
- Back in GitHub Classroom, select your course and create an assignment



Creating Assignments

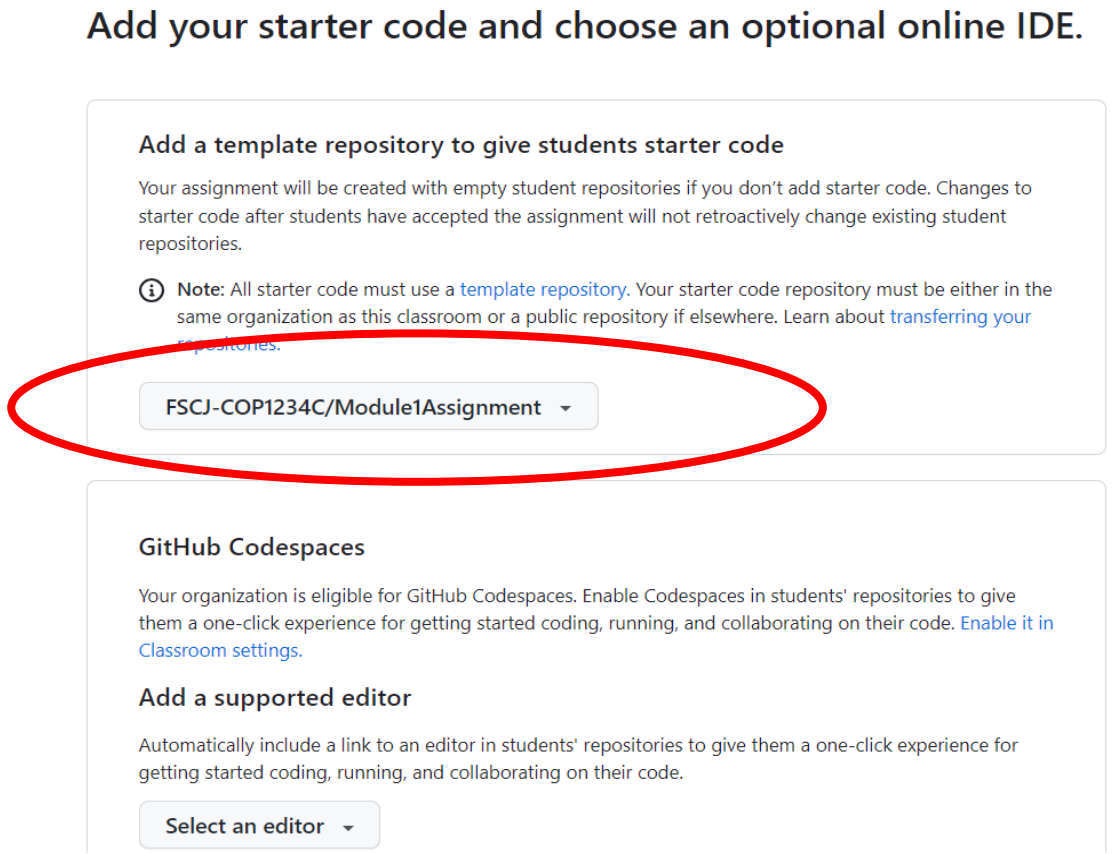
- Back in GitHub Classroom, select your course and create an assignment

- Can use same name as template, or not
- Private visibility for repo is a good thing



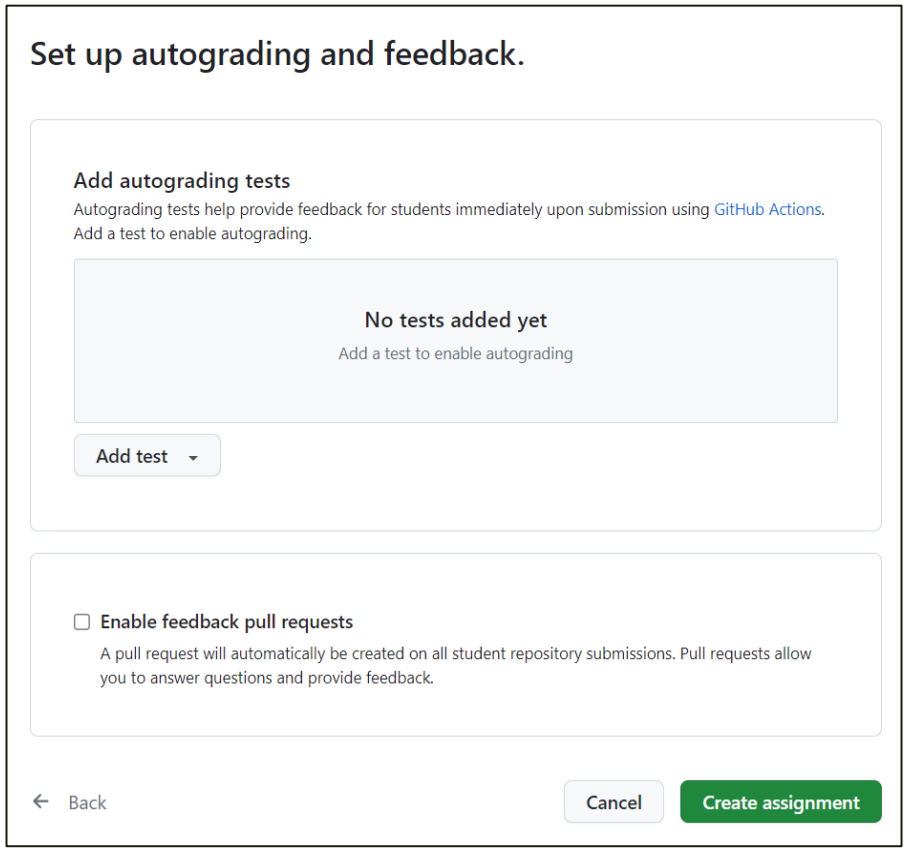
Creating Assignments

- Select your GitHub template repo for the assignment starter code:



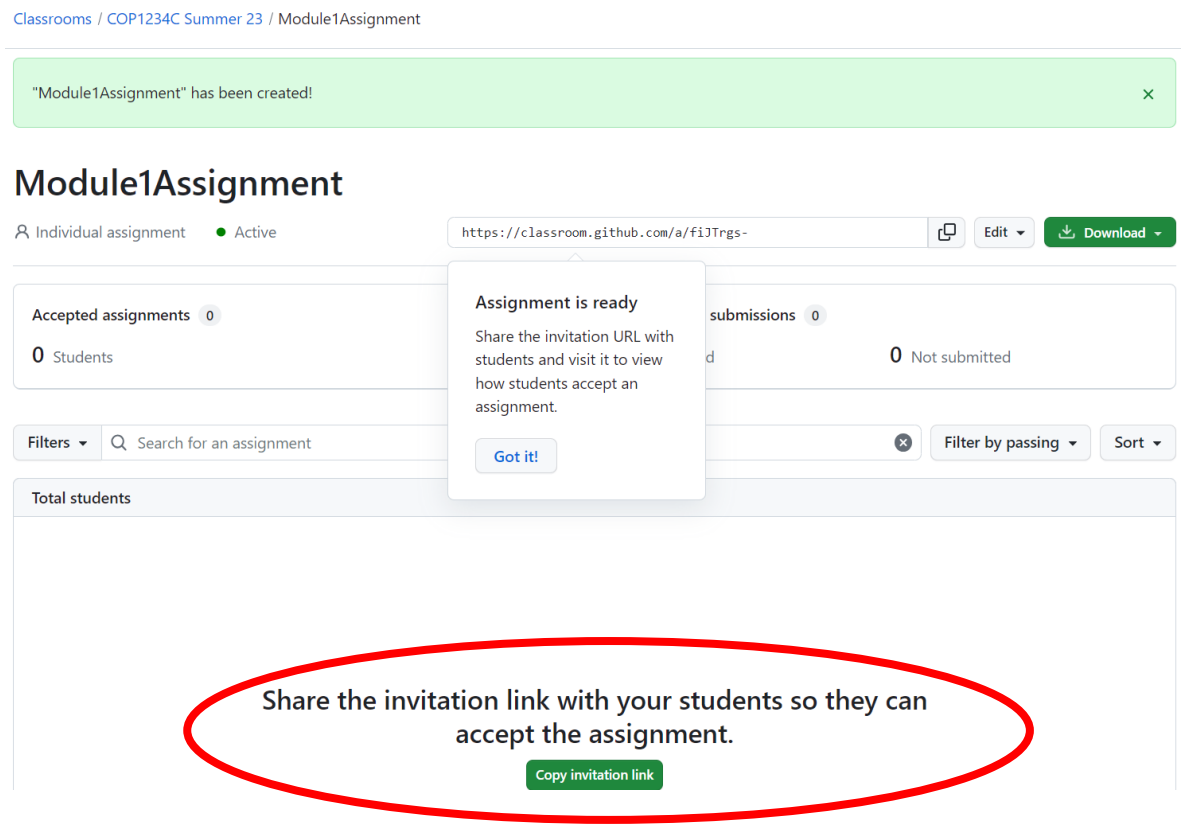
Creating Assignments

- Set up a test for autograding if desired, then "Create assignment"



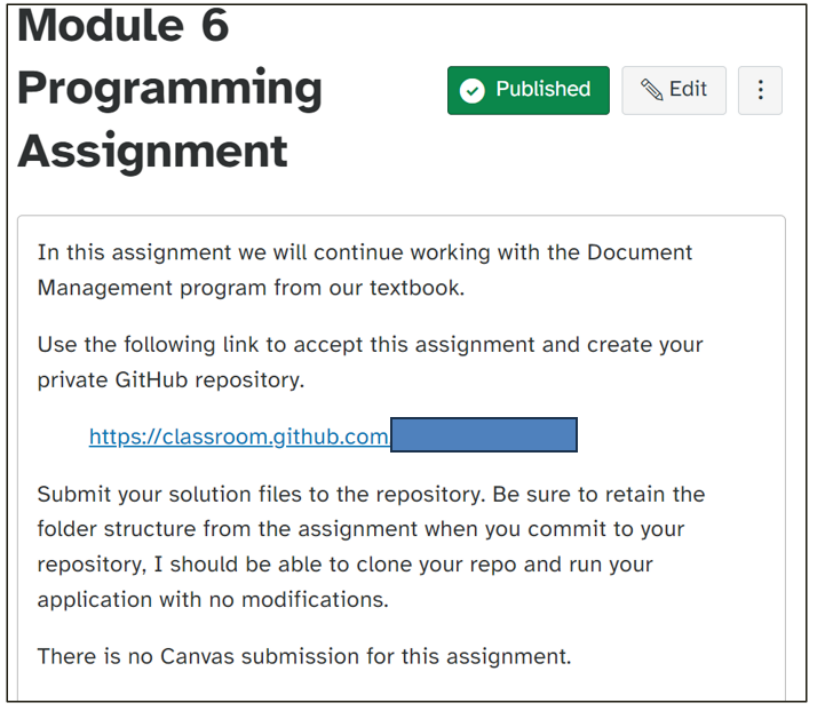
Creating Assignments

- Provide the assignment invitation link to your students



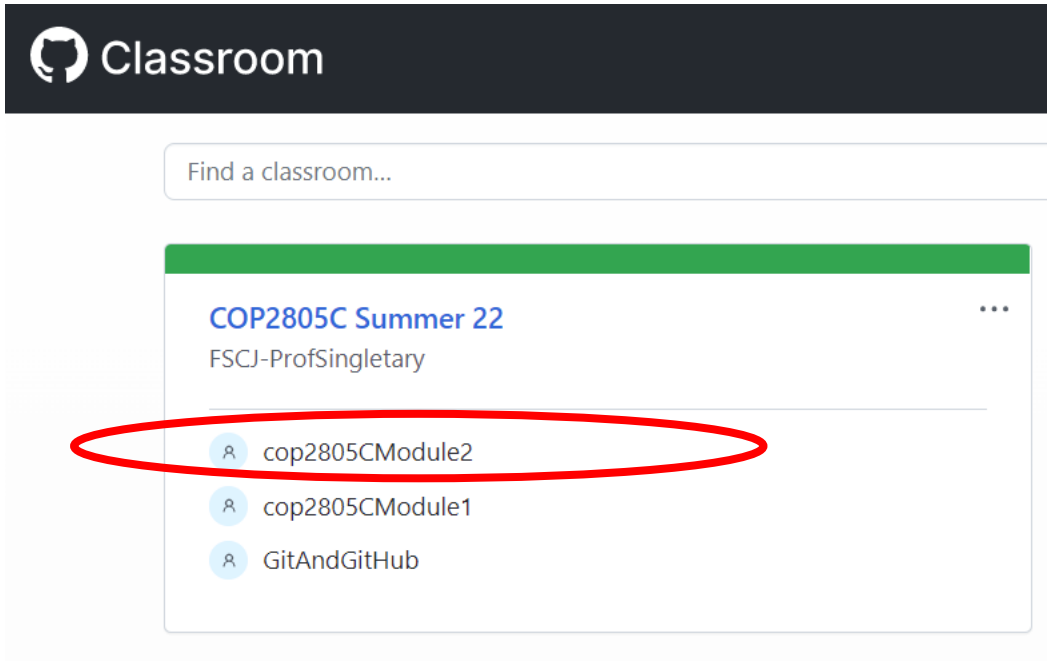
Creating Assignments

- Sample assignment in Canvas:



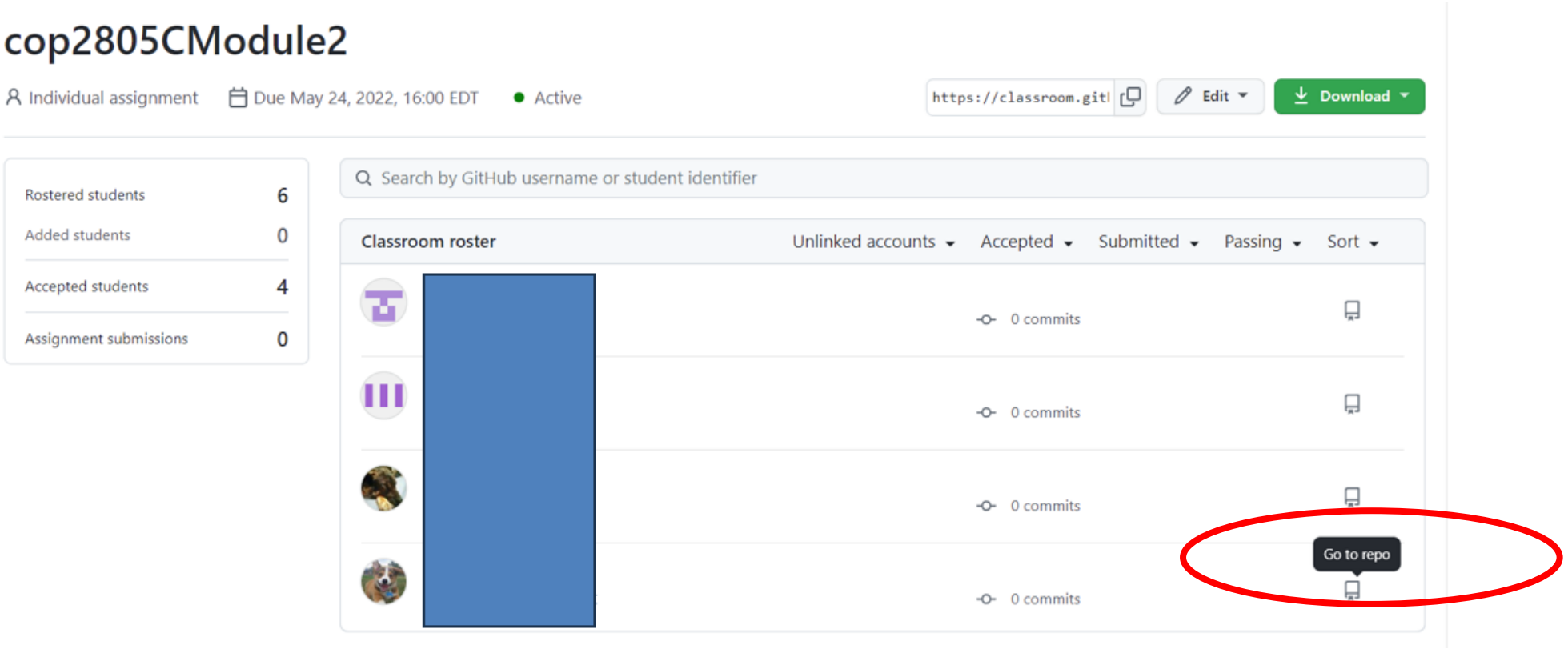
Grading Assignments

- Log in to GitHub Classroom and Select the Course Assignment
  - https://classroom.github.com/classrooms



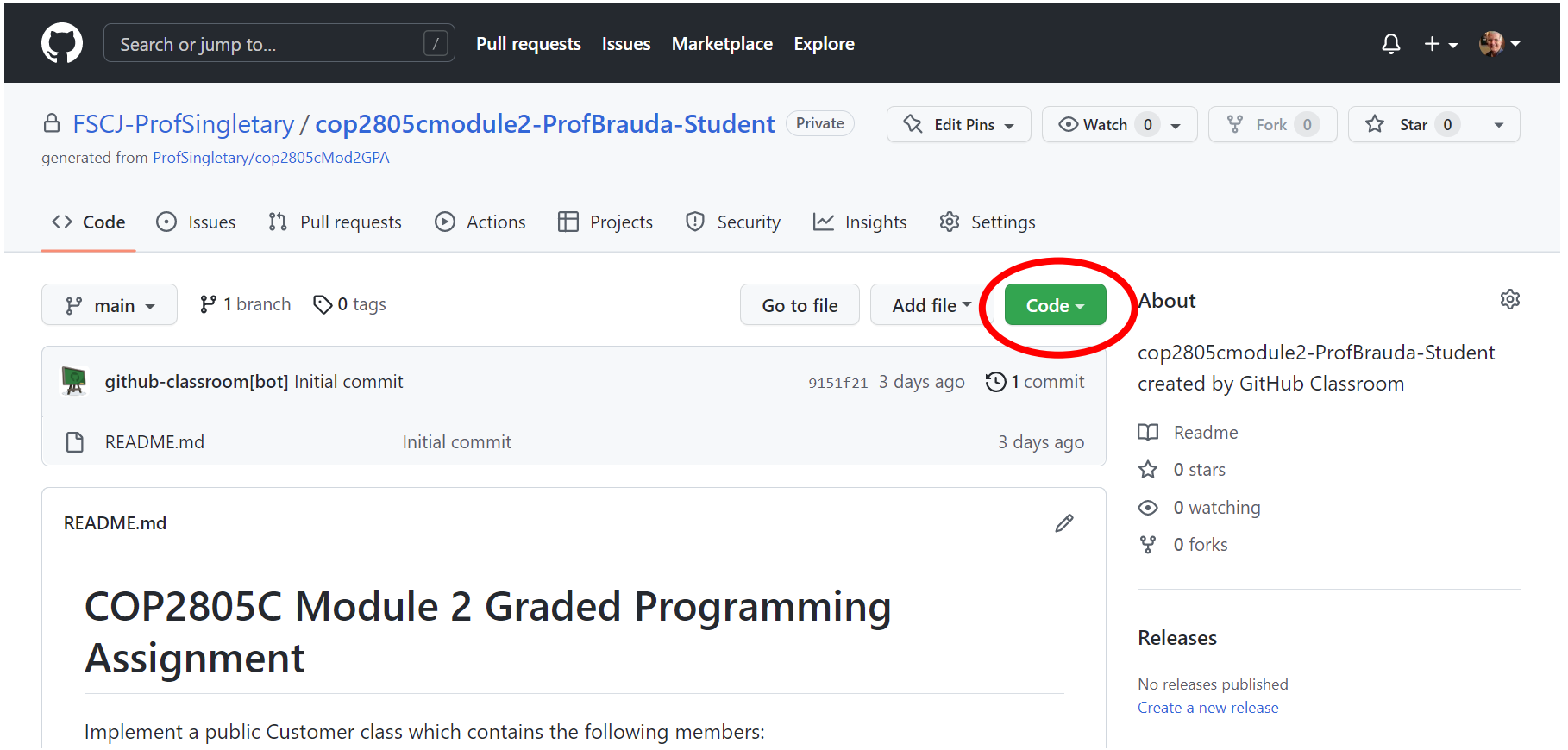
Grading Assignments

- Select the "Go to repo" Icon



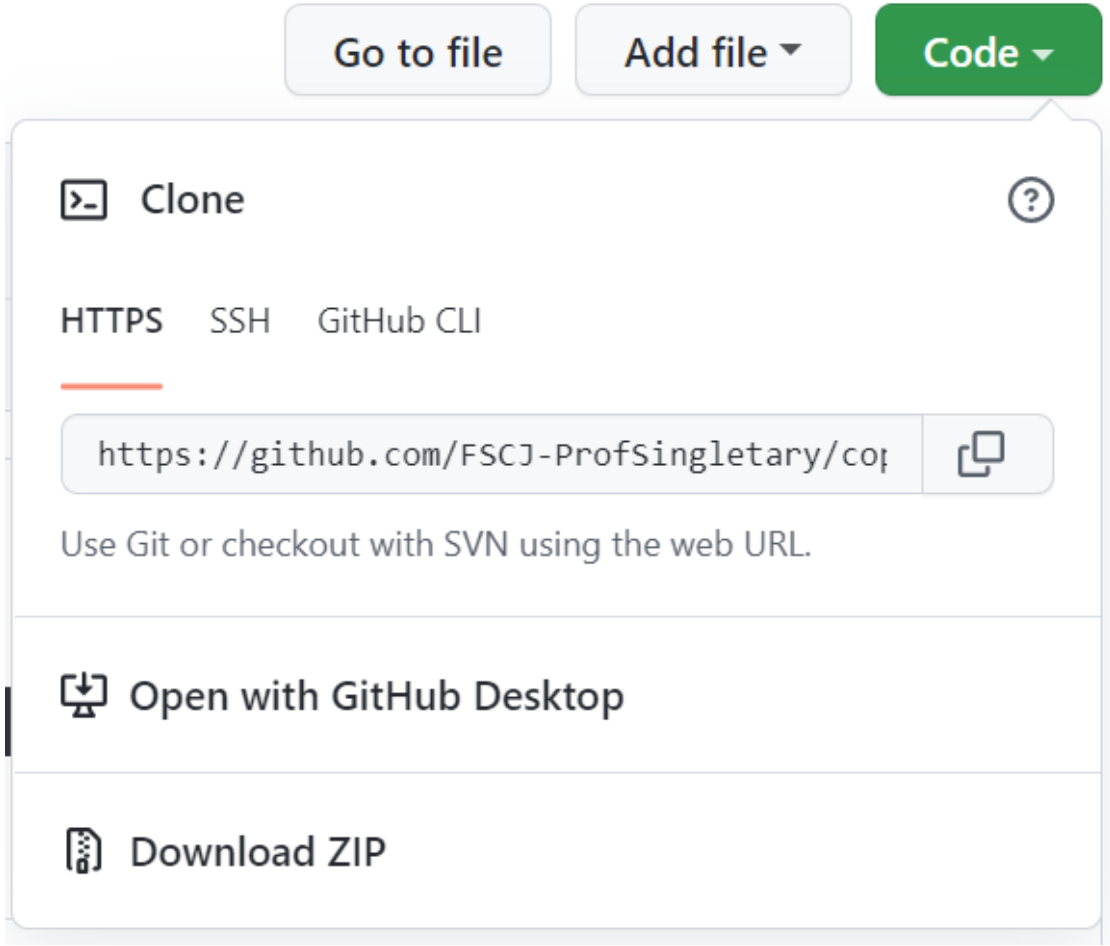
Grading Assignments

- Select "Code" For Download/Access Options



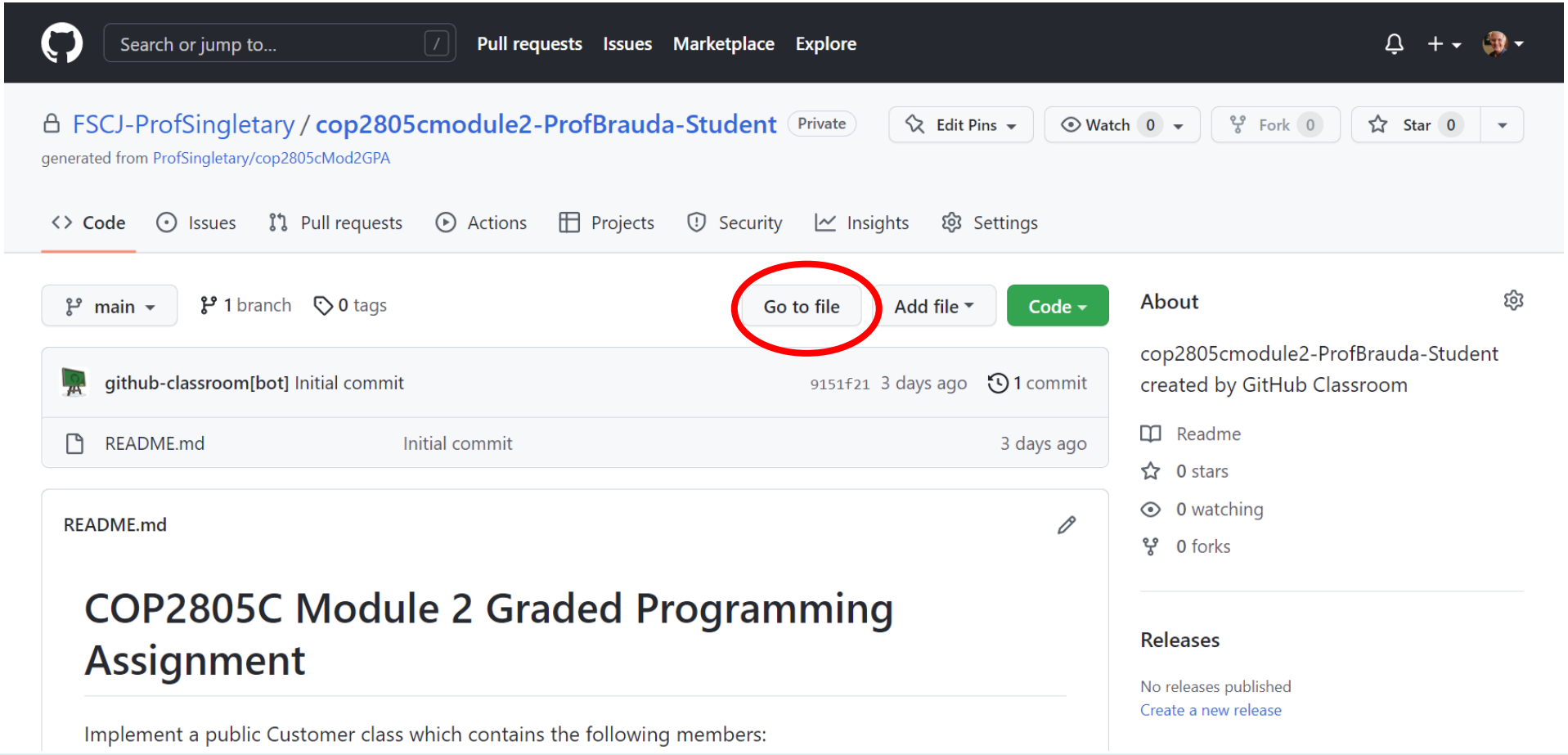
Grading Assignments

- Choose Your Preferred Option
  - Clone the student repo
  - Open with GH Desktop
  - Download Zip



Grading Assignments

- Or Use "Go to file" to Go Directly to Desired File



GitHub Workflows

- A GitHub workflow is a configurable automated process that will run one or more jobs.
- Workflows are defined by a YAML file checked in to your repository and will run when triggered by an event in your repository, or they can be triggered manually, or at a defined schedule.
- Workflows are defined in the .github/workflows directory in a repository, and a repository can have multiple workflows, each of which can perform a different set of tasks.
  - You can have one workflow to build and test pull requests, another workflow to deploy your application every time a release is created, and still another workflow that adds a label every time someone opens a new issue

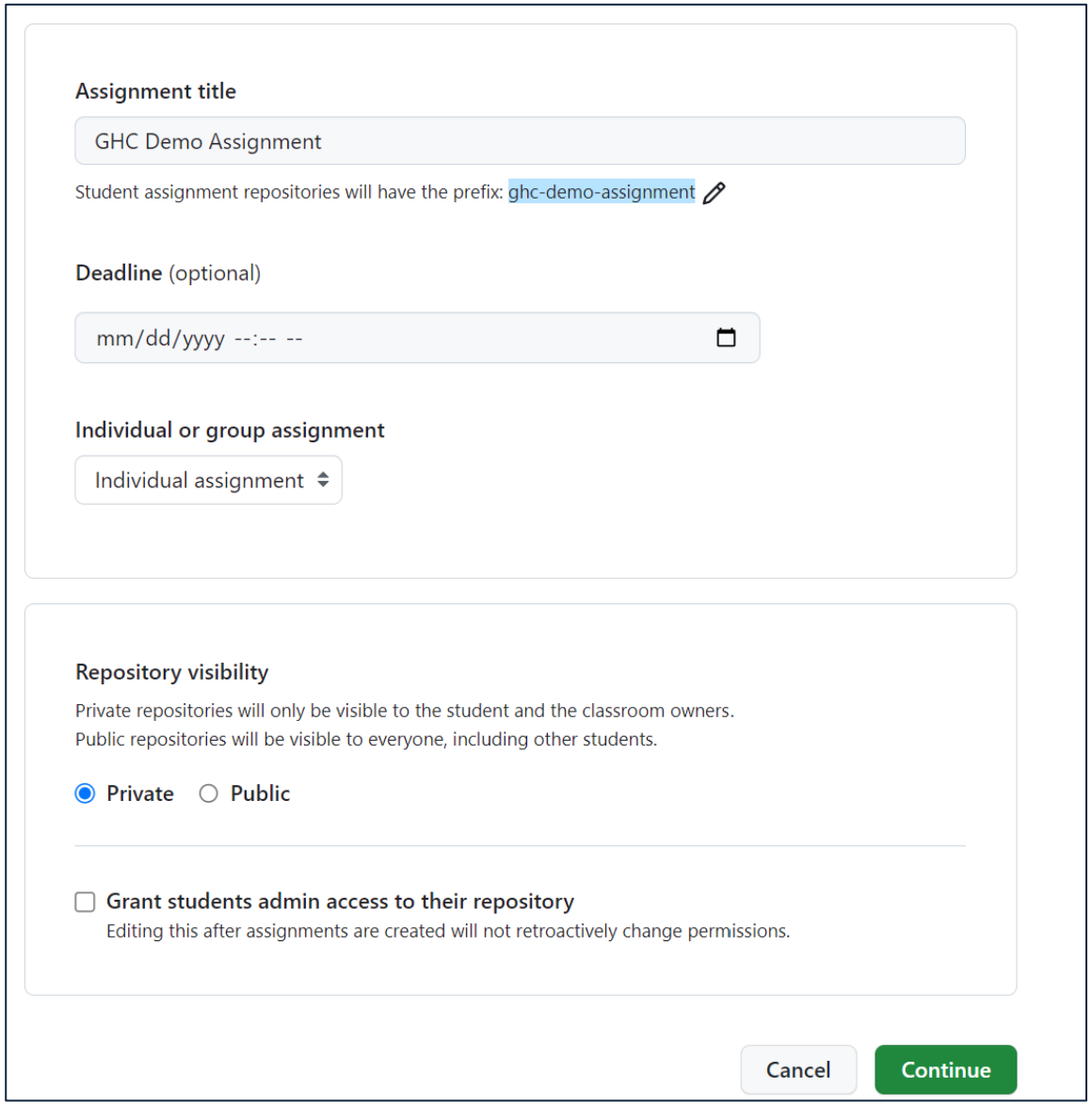
https://docs.github.com/en/actions/using-workflows/about-workflows

Automated Grading using Actions and Workflows

- You can use autograding to automatically check a student's work for an assignment on GitHub Classroom.
- You configure tests for an assignment, and the tests run immediately every time a student pushes to an assignment repository on GitHub.com.
  - The student can view the test results, make changes, and push to see new results.
- After a student accepts an assignment, on every push to the assignment repository, GitHub Actions runs the commands for your autograding test in a Linux environment containing the student's newest code.
- GitHub Classroom creates the necessary workflows for GitHub Actions.
- You can add, edit, or delete autograding tests for an existing assignment.
  - (All changes made via the Classroom UI will be pushed to existing student repositories, so use caution when editing tests)
- https://docs.github.com/en/education/manage-coursework-with-github-classroom/teach-with-github-classroom/use-autograding

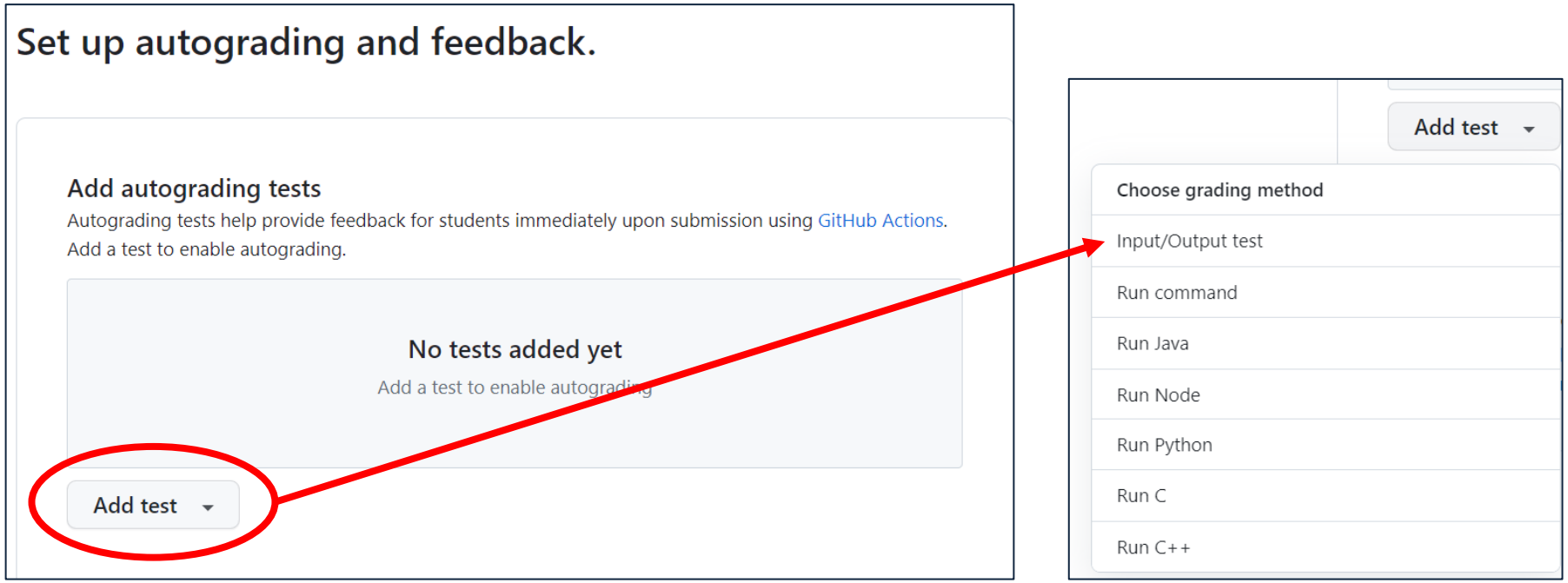
Automated Grading

- Set up the assignment as shown earlier



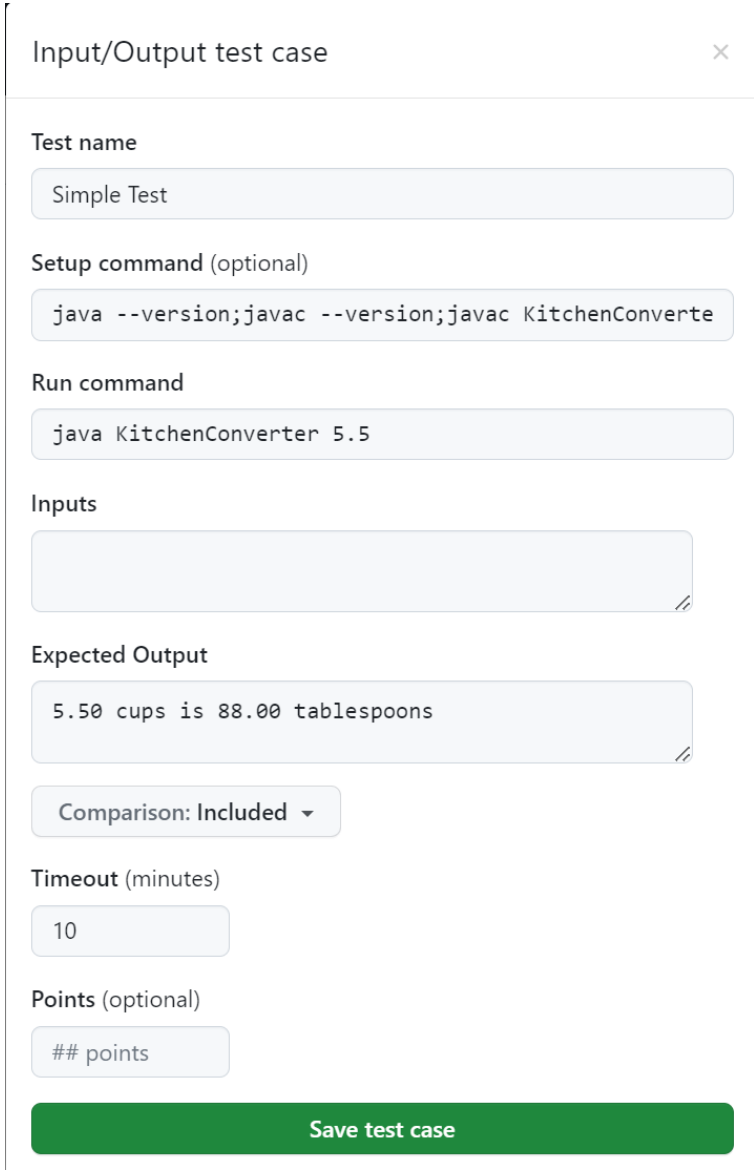
Automated Grading

- Add a test in the "Set up autograding and feedback" section, choose "Input/Output test"



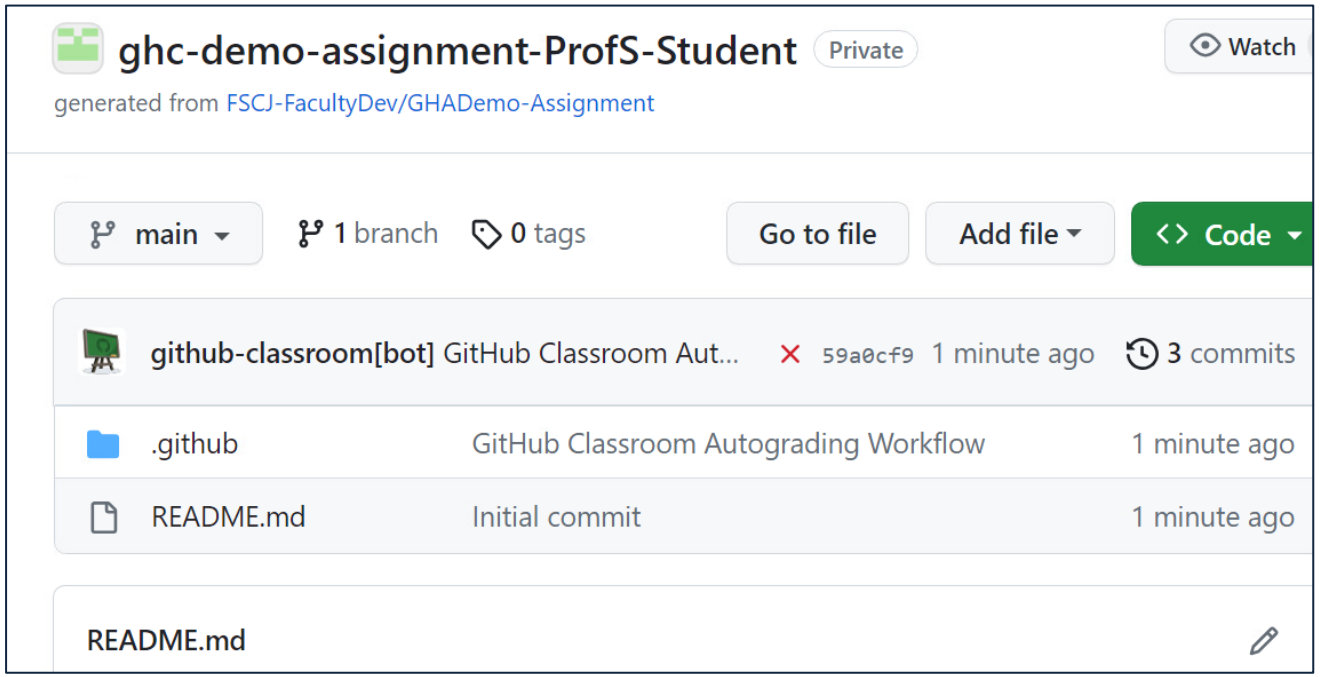
Automated Grading

- I used the following parameters:
  - Test name: Simple Test
  - Setup command: javac KitchenConverter.java
  - (builds the students application)
  - Run command:
  - java KitchenConverter 5.5 (application requires command line input)
  - Expected output: 5.50 cups is 88.00 tablespoons
  - Timeout: 10 minutes (default)
  - Points (optional): not specified



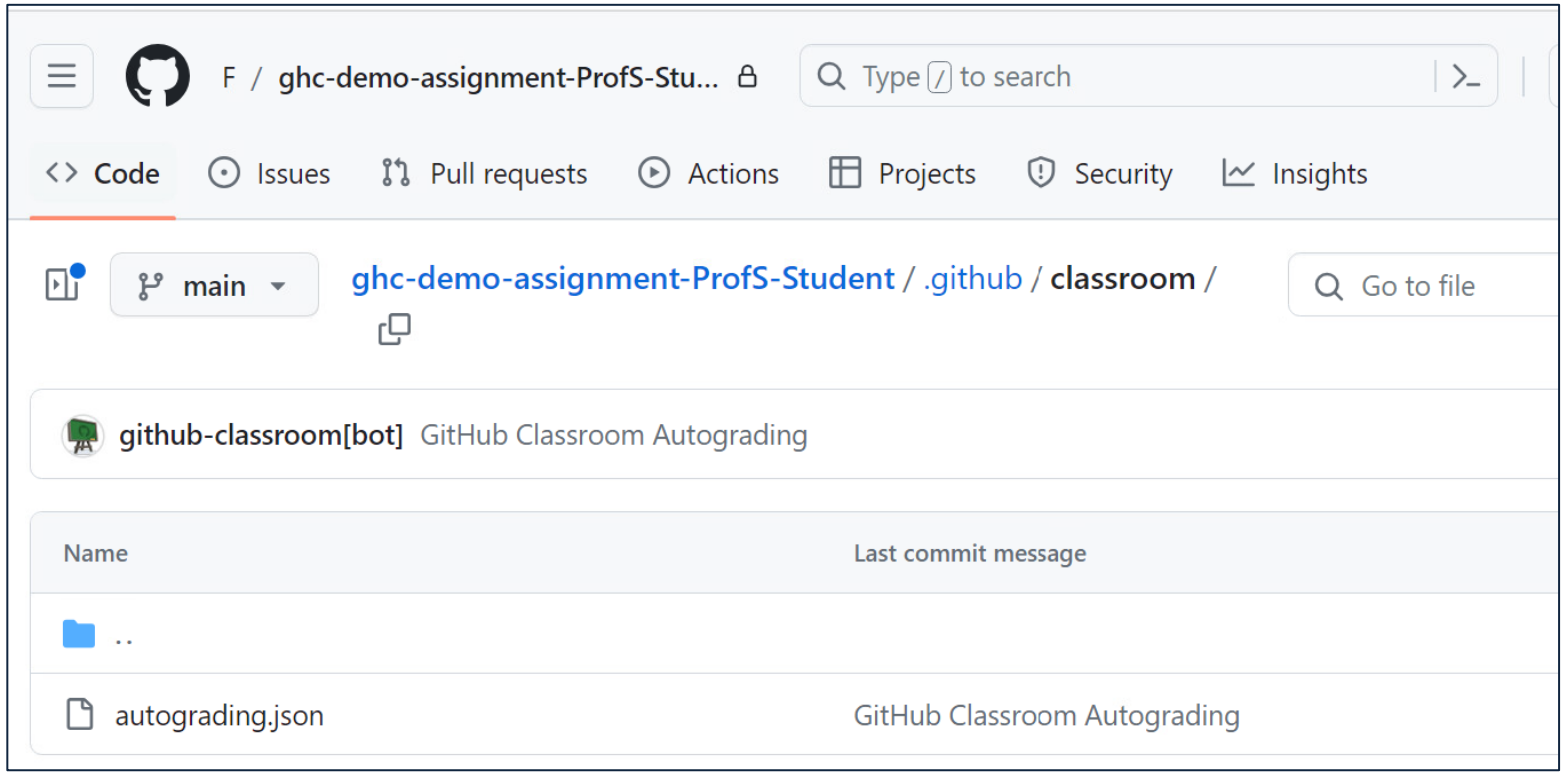
Automated Grading

- The assignment repo is created with two folders: .github/classroom and .github/workflows



Automated Grading using Workflows

- The classroom folder contains a JSON input file



Automated Grading

- JSON input file

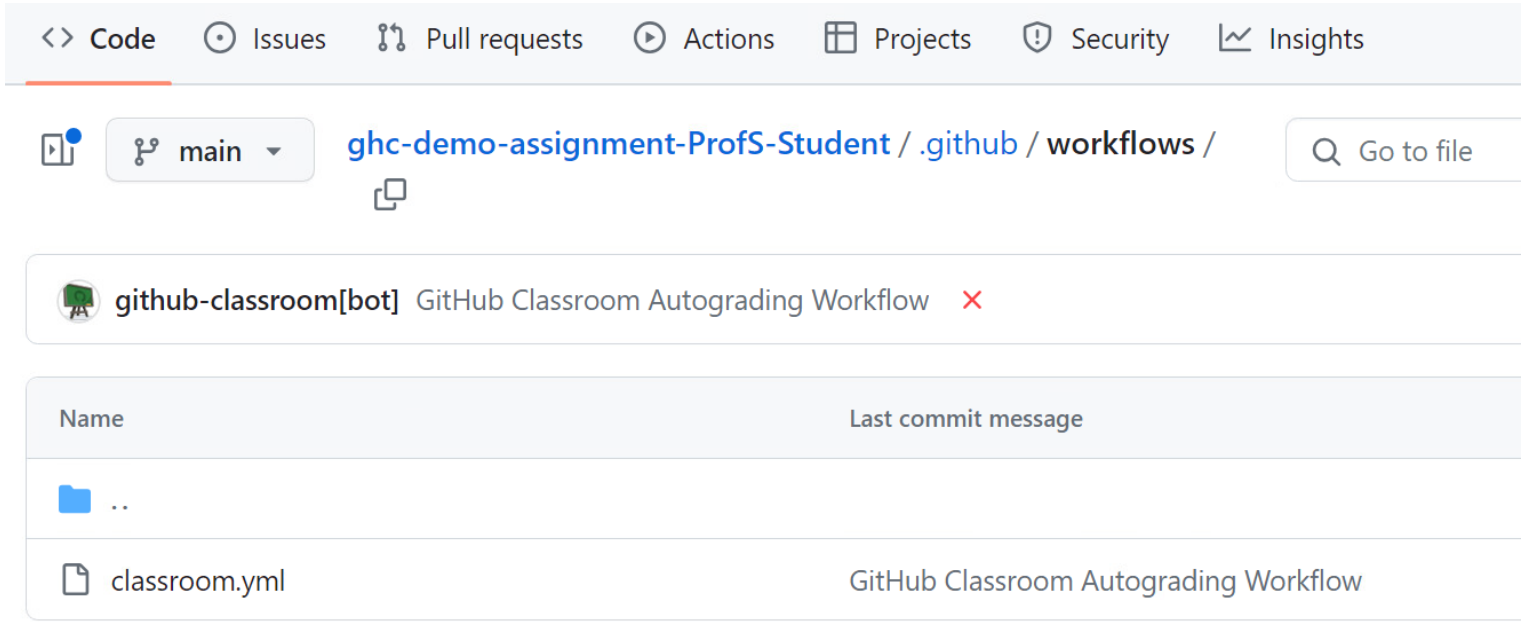
github-classroom[bot] GitHub Classroom Autograding

Code | Blame | 14 lines (14 loc) · 323 Bytes

```
1   {
2     "tests": [
3       {
4         "name": "Simple test",
5         "setup": "java --version;javac --version; javac KitchenConverter.java",
6         "run": "java KitchenConverter",
7         "input": "5.5",
8         "output": "5.50 cups is 88.00 tablespoons",
9         "comparison": "included",
10        "timeout": 10,
11        "points": null
12      }
13    ]
14  }
```

Automated Grading

- The workflows folder contains a YAML workflow file

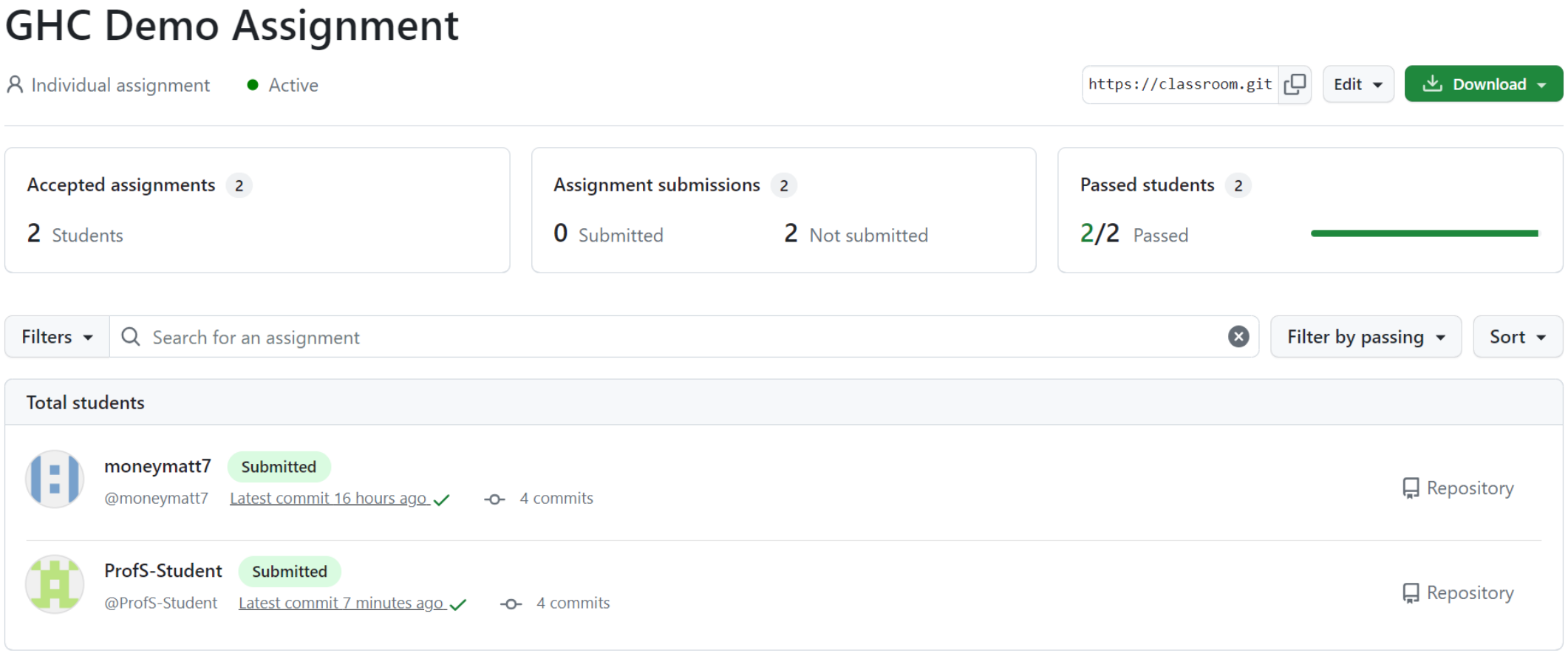


Automated Grading

Code | Blame | 20 lines (20 loc) · 979 Bytes

```
1   name: GitHub Actions Demo
2   run-name: ${{ github.actor }} is testing out GitHub Actions 🚀
3   on: [push]
4   jobs:
5     Explore-GitHub-Actions:
6       runs-on: ubuntu-latest
7       steps:
8         - run: echo "🎉 The job was automatically triggered by a ${{ github.event_name }} event."
9         - run: echo "🐧 This job is now running on a ${{ runner.os }} server hosted by GitHub!"
10        - run: echo "🔎 The name of your branch is ${{ github.ref }} and your repository is ${{ github.repository }}."
11        - name: Check out repository code
12          uses: actions/checkout@v3
13        - run: echo "💡 The ${{ github.repository }} repository has been cloned to the runner."
14        - run: echo "🖥️ The workflow is now ready to test your code on the runner."
15        - name: List files in the repository
16          run: |
17            ls ${{ github.workspace }}
18        - run: echo "🍏 This job's status is ${{ job.status }}."
19        - run: javac KitchenConverter.java
20        - run: java KitchenConverter 5
```

Automated Grading

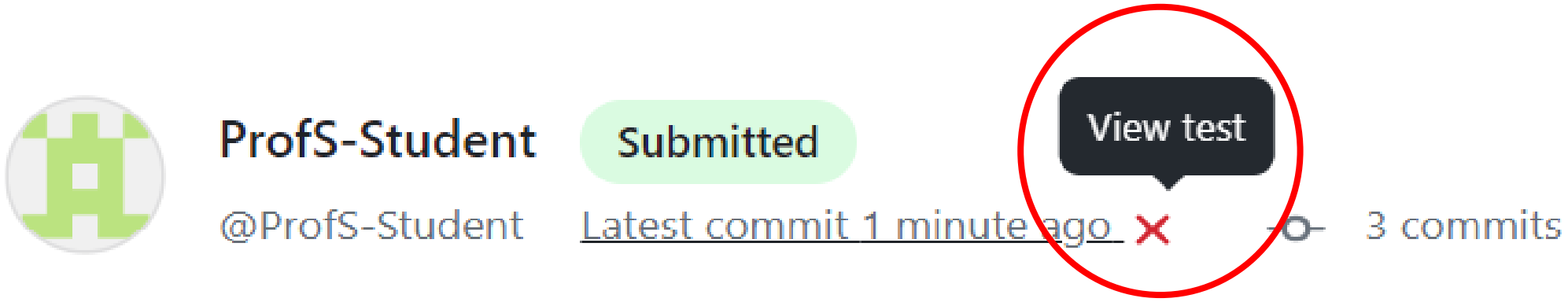


Automated Grading

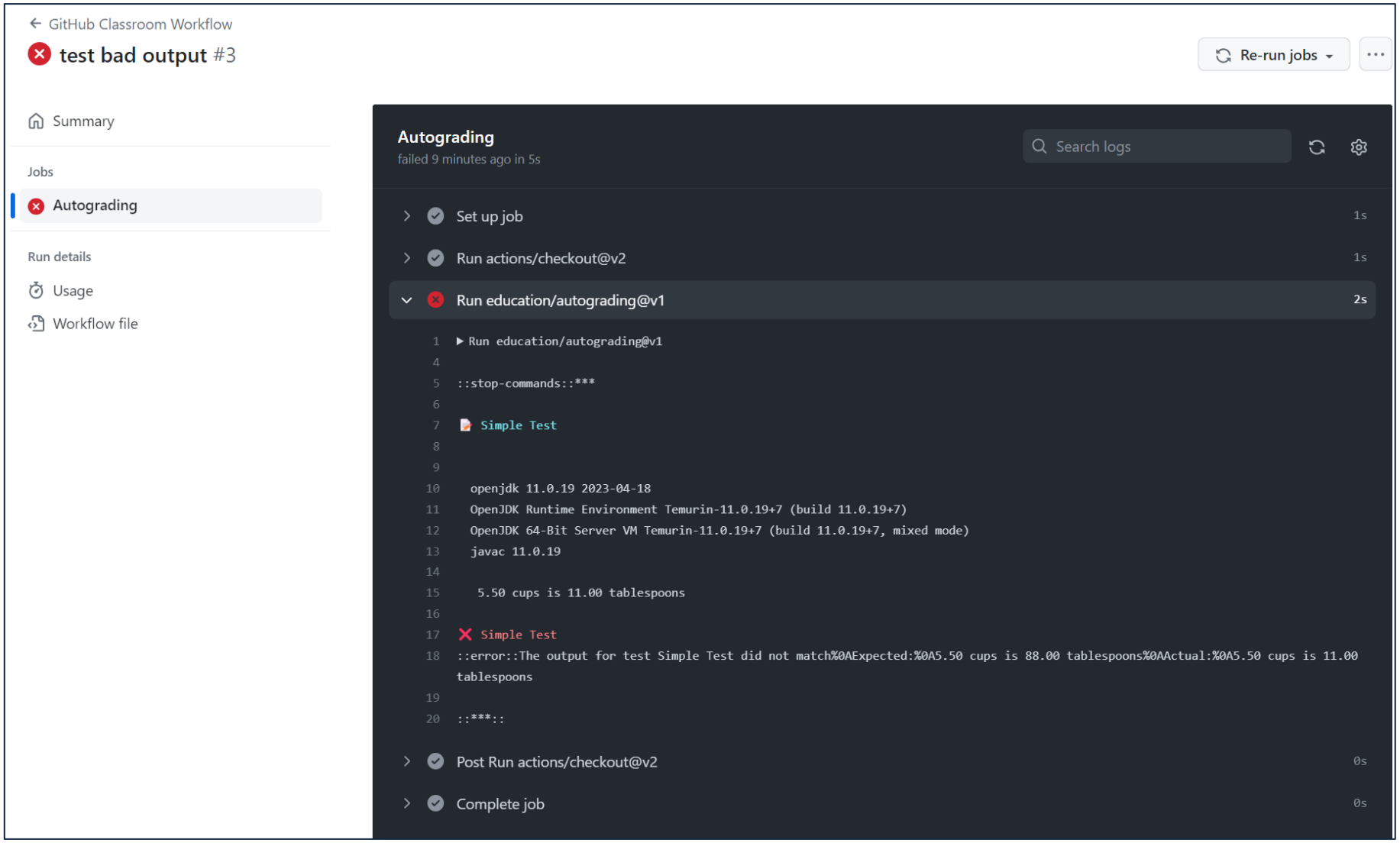
- To see test details, click on the icon.

Automated Grading

- Failed tests are marked with an X, which you can click on

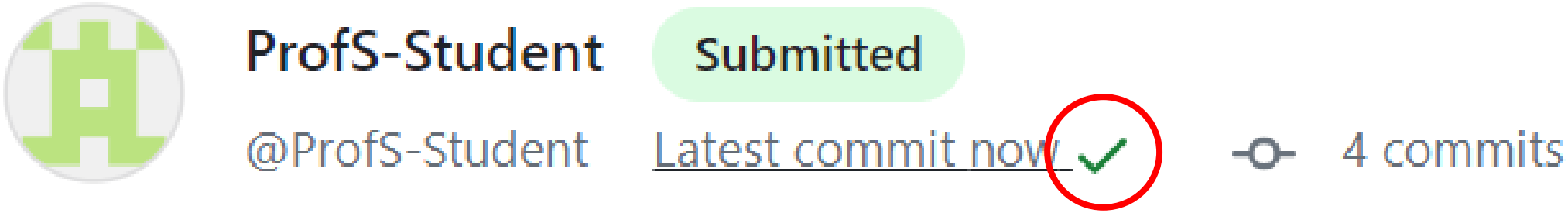


Automated Grading

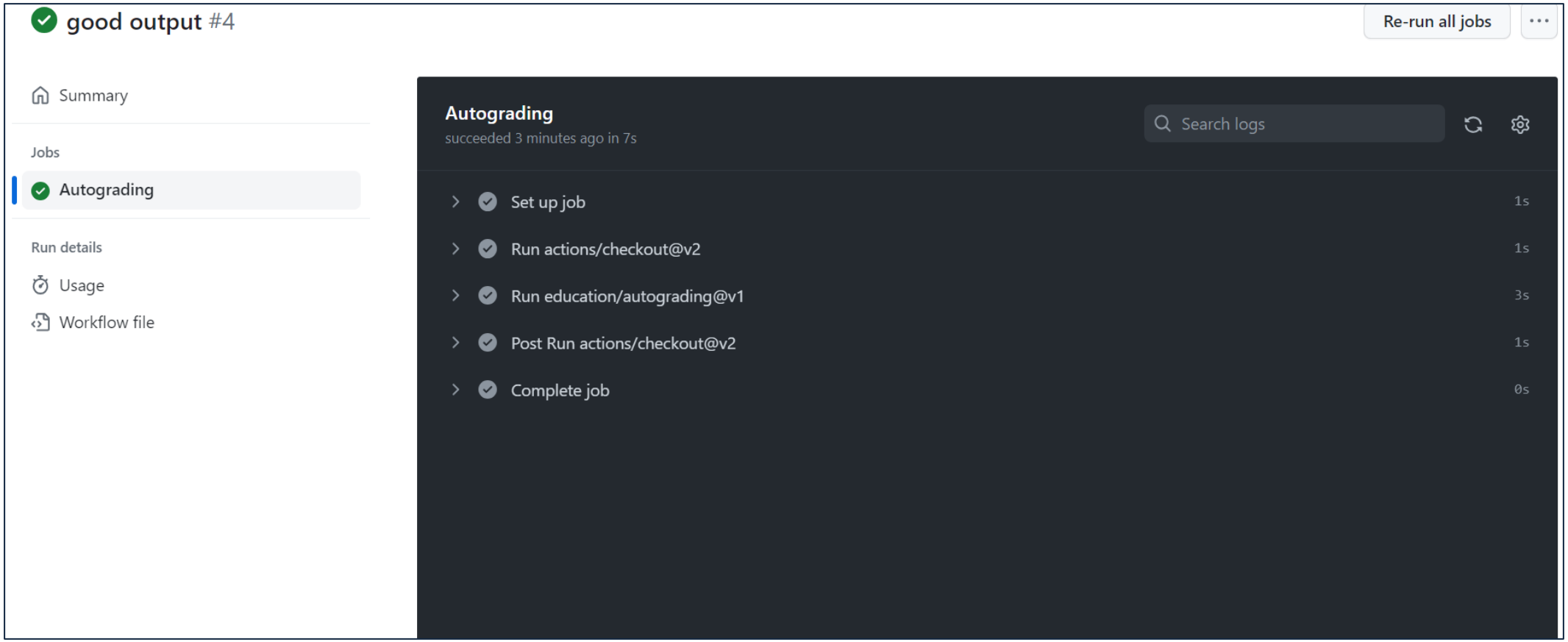


Automated Grading

- Successful tests are marked with a green checkmark, which you can also click on



Automated Grading



This material is based upon work supported by the National Science Foundation under Grant No. 1902524.