

CART 263 **Creative Computation 2**

Email: l.wilkins@concordia.ca

Office Hours: Tuesday 12-1

Course Github: <https://github.com/LeeCyborg/CART263-W-23>

TA: tricia.enns@gmail.com

Welcome back :)

I'm lee, and our TA this semester is Tricia.

- Name / Pronouns
- What you're excited for this semester (doesn't have to be this class!)
- What you did over break

Learning Objectives

- Learn how to use object Oriented Programming to make more advanced and complex work
- Learn how to document, share, and collaborate on code
- Explore advanced applications for creative coding

What we'll be doing today

- Syllabus overview
- Getting set up on Github and VSCode
- Review to markdown and ReadMe files
- Exercise 1: getting back into code

Syllabus

Getting Back Into Code (10%)

The purpose of this short project is to get students back into code after the break. This project should be a small piece of code, experiment, or iteration from a project last semester.

Project 1: Particles With Personalities (20%)

In this project every student will receive a a specification to create a unique particle system using object oriented programming.

Project 2: Data Visualization (20%) In this project students will create a data visualization of any data set.

Project 3: Playful Interaction (40%)

This project is a playful interaction, that can be either a game, experience, or other interactive work.

Participation (10%)

Participation includes: attending class, completing coursework, asking questions, contributing to discussions, sharing ideas, and coming to office hours, emailing the instructor with updates on projects (even if you do not need help, make sure to touch base with the instructor regularly in some capacity). Maintaining an up to date github is part of participation for this class. Due: Assessed at the end of the course

Schedule

- Week 1: Getting back into code: Setting up Git and VSCode.
- Week 2: **Getting back into code due.**
- Week 3: Intro to Object Oriented Programming.
- Week 4: Live code session & OOP continued.
- Week 5: **Particles with Personalities due.**
- Week 6: Data visualization techniques.
- Week 7: Using APIs in P5js.
- Week 8: **Data vis project due.**
- Week 9: Debugging & Using mobile devices.
- Week 10: Digital fabrication with p5js.
- Week 11: IoT.
- Week 12: Studio Session.
- Week 13: **Playful Interaction due.**

Handing in assignments

All assignments should be on **GitHub** and **Moodle**. Every assignment should have a **README.md** page explaining your work.

All code should be **commented** clearly, with reasonable and clear **variable and function names**.

Any code taken from, or modified from somewhere on the internet, a class mate or Stack Overflow should be **clearly labeled and cited**. It's okay to use other people's code, just cite it! If you fail to cite your code, it can result in a 0 in the assignment or even failing the course.

Late work

All **projects** should be handed in at **midnight the day it is due, unless otherwise stated**. This usually means you have opportunity after class to fix any mistakes or address any critique.

All **exercises** should be handed in **before class**, so we can work out the solution together.

Any late work will loose a letter grade per day (B+ work will be C+) unless otherwise negotiated.

Accommodations and support

We want to support you! But its not possible if we don't know theres a problem. Please talk to Lee or Tricia if theres an issue getting your work in on time or if you need an extension. **Do this as soon as possible not the night before it is due!**

If you need additional help:
Access centre for students with disabilities <https://www.concordia.ca/students/accessibility.html>

Student accommodations <https://www.concordia.ca/students/accessibility/accommodations.html>

If you need help....

Don't hesitate to email us if you need help or don't understand. Its important to ask questions, everyone learns differently.

Lee: l.wilkins@concordia.ca
Tricia: tricia.enns@gmail.com

We want you to succeed!

Resources

- P5JS Reference - <https://p5js.org/reference/>
- P5JS Discord server - <https://discord.com/invite/SHQ8dH25r9>
- The Coding Train - <https://thecodingtrain.com/tracks/code-programming-with-p5-js>
- The Nature Of Code - <https://thecodingtrain.com/tracks/the-nature-of-code-2>
- Google it! (Seriously)
- Last semester, CART 253 course notes on Github - <https://github.com/LeeCyborg/CART253-F-22>
- Peers and classmates

Template

We are using the same template as last semester.
Find it on Moodle or Github, [here](#).

GitHub

Github helps you keep code on your computer (local) and code inside the repository (repo) synced.

Github desktop is a tool that helps that process. There's also a command line version some people prefer.

Github is also a **version control software** that lets you go back to a older version, so it's good to update your code when you have a version you're happy with! You can create multiple versions of the same code too, or make changes to someone else's code.

It's also a good tool to help with **team work**. Multiple people can contribute code to the same project easily.

GitHub

We will be using GitHub this semester. Its an important tool to learn how to maintain your code and collaborate with your peers!

- Create a Github account on [GitHub.com](https://github.com)
- Follow our class repository [here](#)
- Add your GitHub link [here](#)
- Download Github desktop [here](#)

GitHub Vocabulary

Repository: A place to keep all code and assets relating to a project. A repository lives on Github and can be added to by others.

Local: Code that is on your computer, but not necessarily on Github.

Commit: The act of publishing new code, or updating old in your local repository.

Push: To update the GitHub repository with your code.

Pull: To take the code from the repository and bring it to your local folder on your computer.

Pull Request: If another user changes your code, they can submit a pull request to add it to your repository.

Merge: To bring two pieces of code together. Sometimes this involves looking at both pieces of code.

Fork: To make another version of an old piece of code.

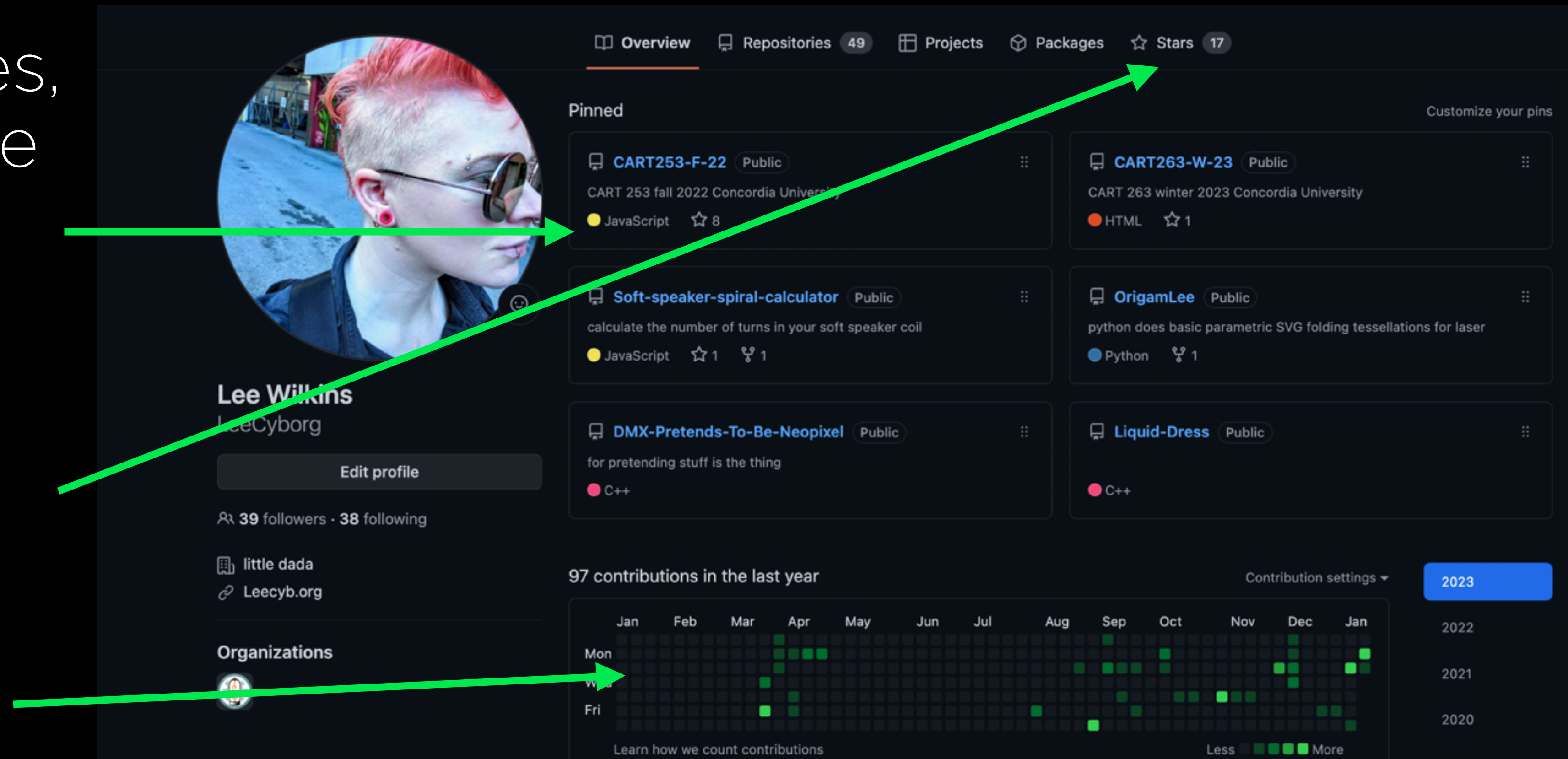
Markdown: A simple set of commands used to format files

Lets look at GitHub account

These are repositories, or collections of code and files that surround a project.

These are favoured repositories

This represents commits, or code contributions.



Lets look at a repository

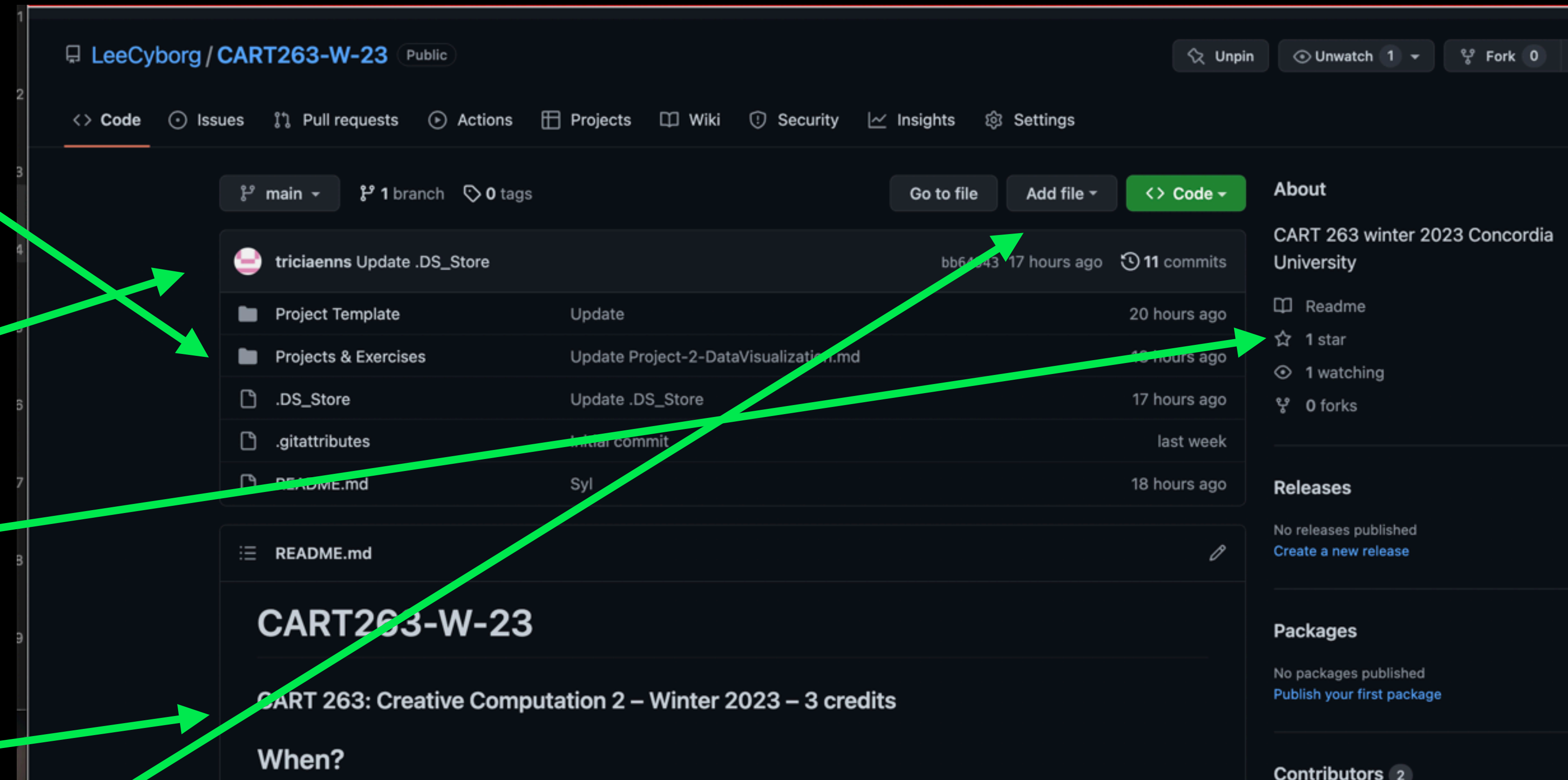
Here you can see all the files associated with the repository

You can see the last person to contribute to the repository

You can see who is watching, following, or who has forked your code.

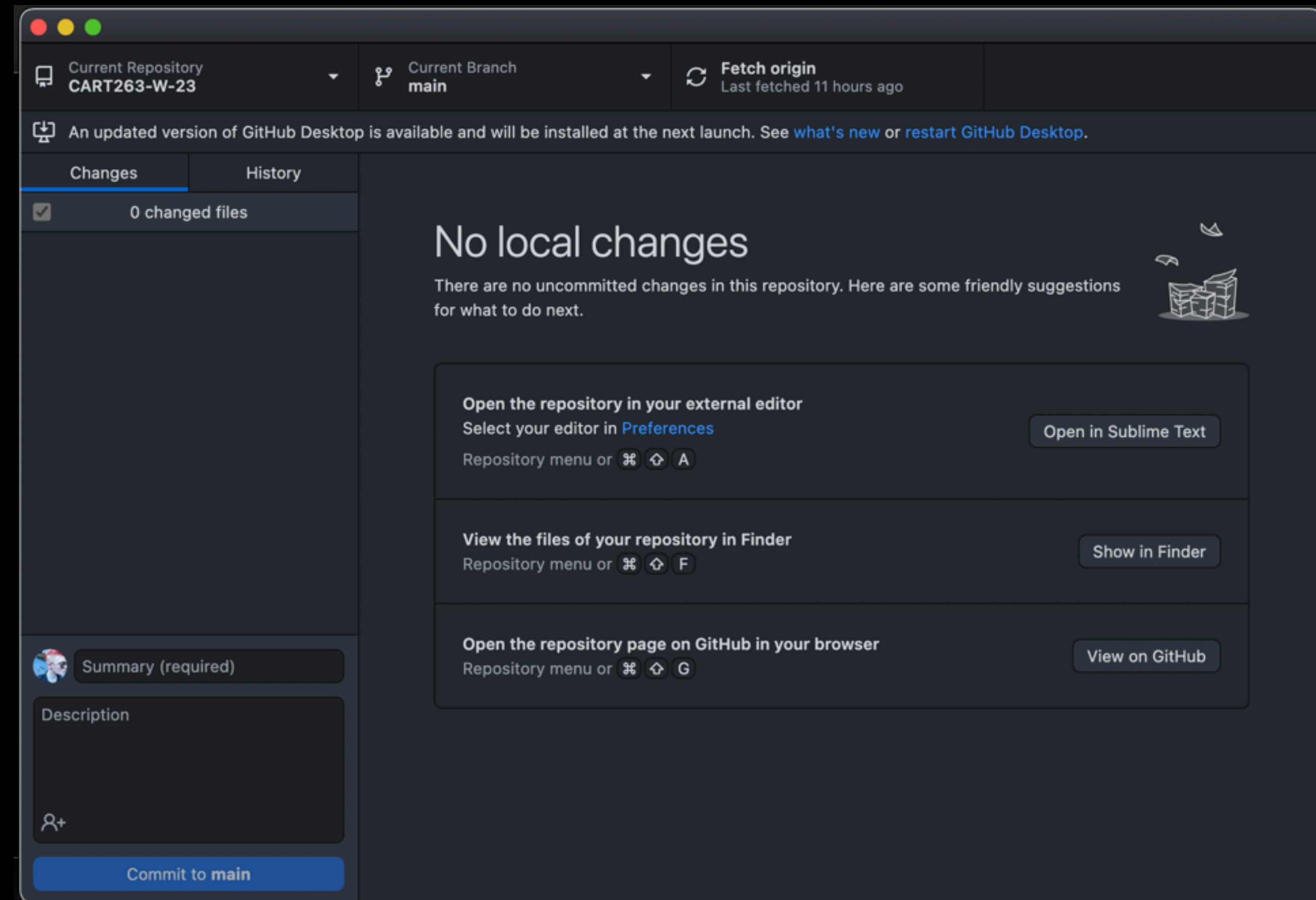
You can see the contents of a readme.md

You can add files manually through the web interface



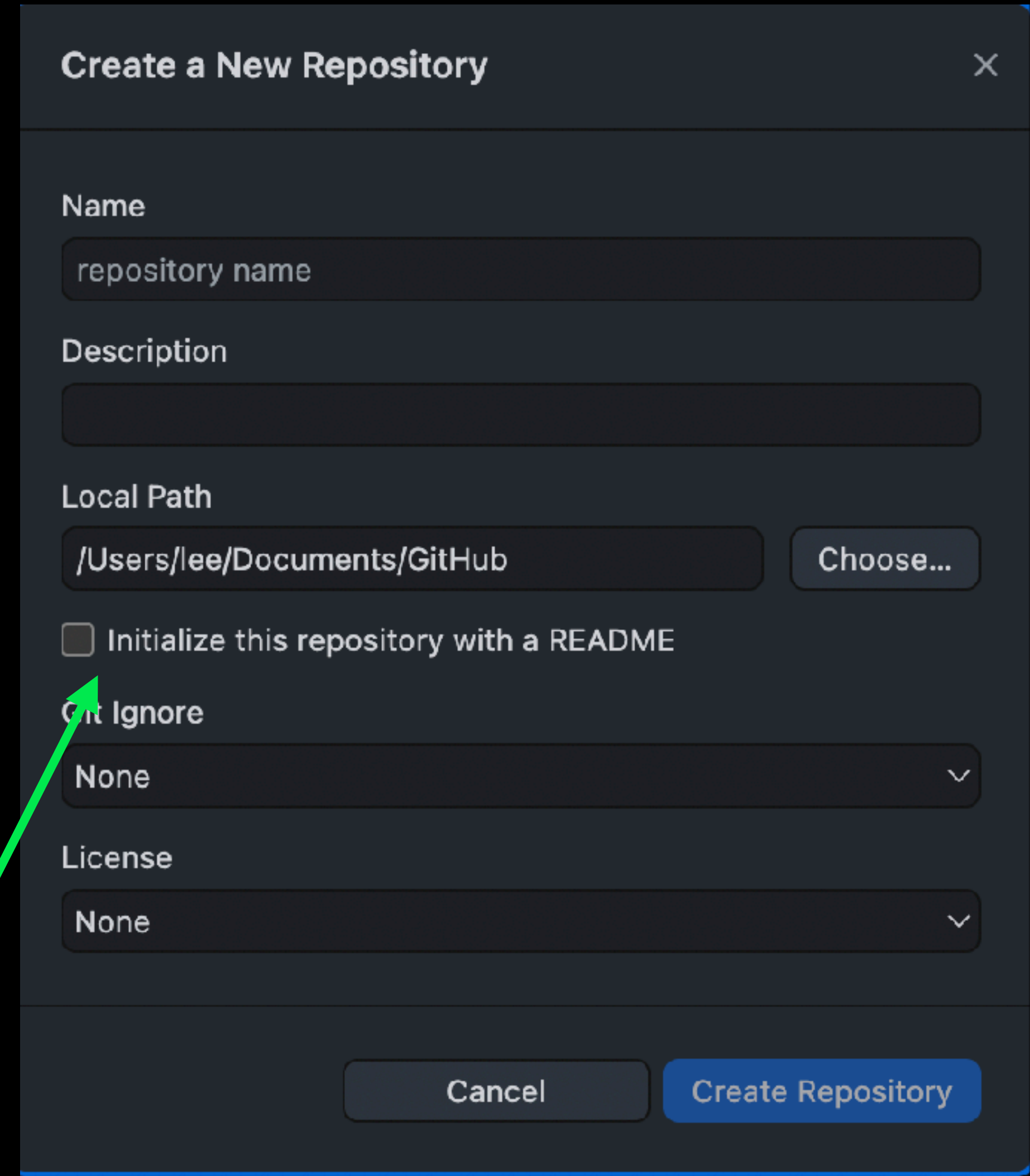
GitHub Desktop

This is a tool for syncing your local folder with your repository folder. Sign into your GitHub account on desktop.



GitHub Desktop

- You can make a repo for the class, or each project. Its up to you! Name it something clear. I suggest making one called CART263 to hold all your projects. File > New Repository.
- Local file path refers to where your files are on your computer. You'll want to keep your GitHub repository and local files synced when you're done working.
- You can automatically add your readme file by clicking this button, lets leave it **UNCHECKED** for now



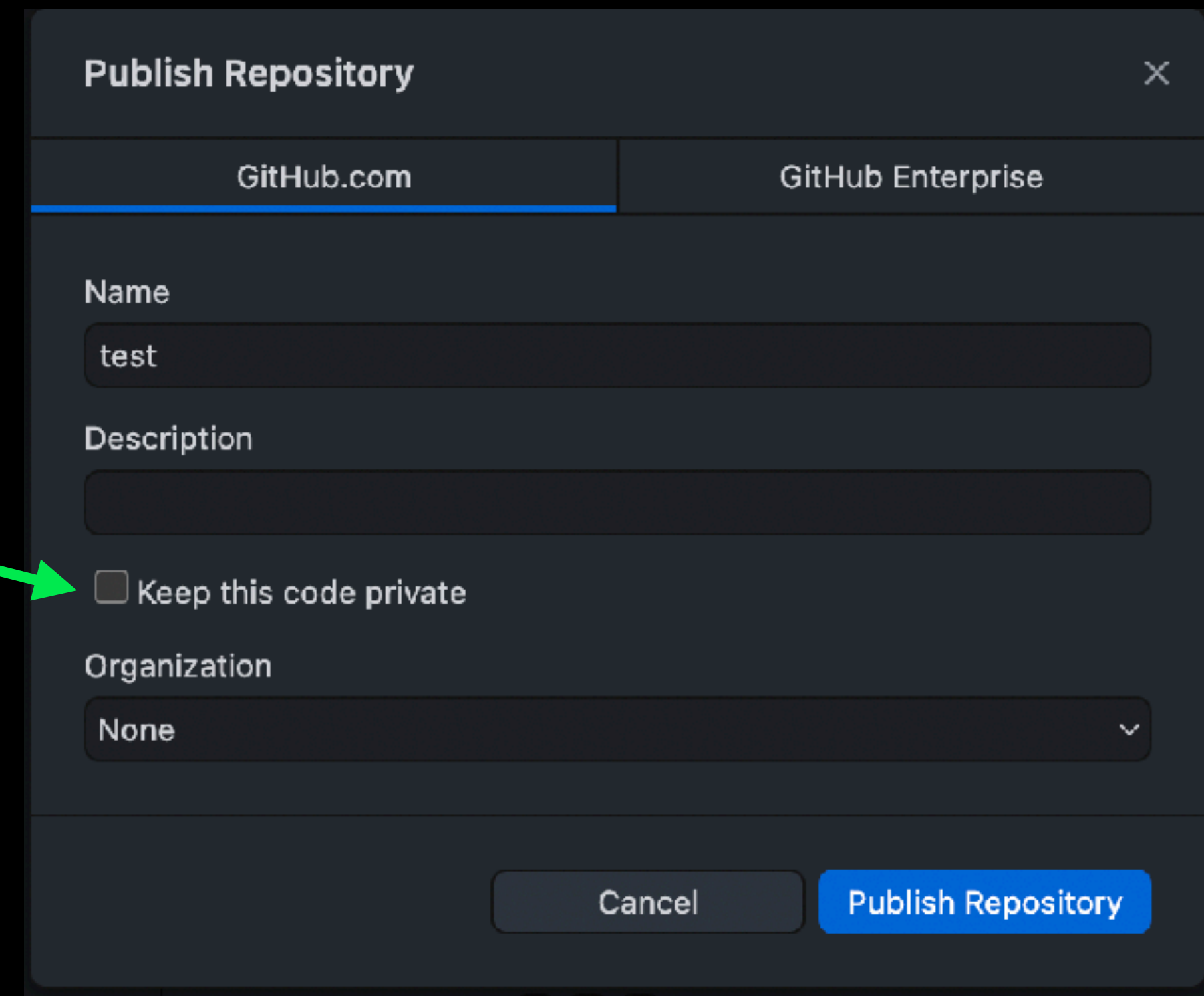
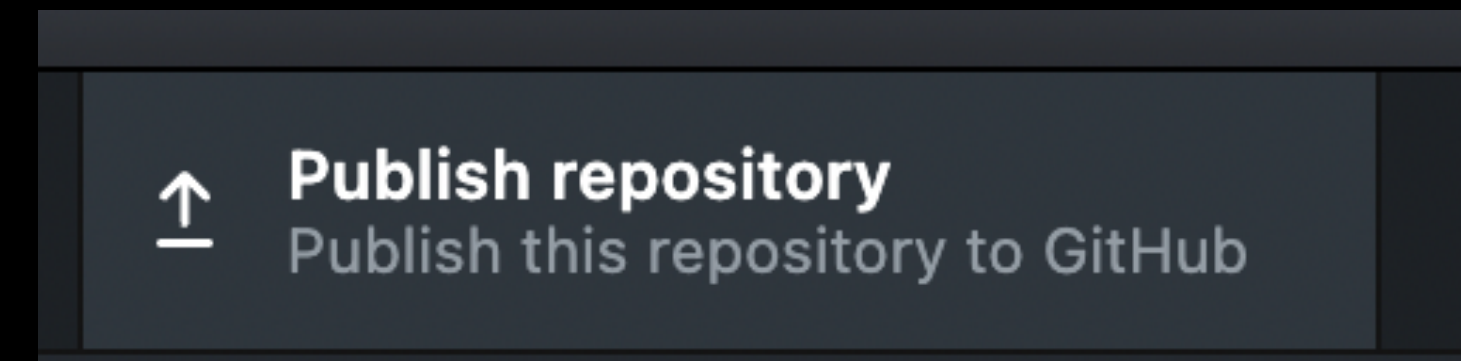
The screenshot shows the 'Create a New Repository' dialog box in GitHub Desktop. The dialog has a title bar with a close button (X). The fields are as follows:

- Name:** A text input field containing 'repository name'.
- Description:** An empty text input field.
- Local Path:** A text input field containing '/Users/lee/Documents/GitHub' and a 'Choose...' button.
- Initialize this repository with a README:** An unchecked checkbox.
- Git Ignore:** A dropdown menu showing 'None'.
- License:** A dropdown menu showing 'None'.

At the bottom, there are two buttons: 'Cancel' and 'Create Repository'.

GitHub Desktop

When you're ready, publish your repository. Make sure your repo isn't private.

A "Publish Repository" dialog box with a dark gray background. It has a title bar with "Publish Repository" and a close button. Below the title bar are two tabs: "GitHub.com" (selected) and "GitHub Enterprise". The form contains the following fields: "Name" with the value "test"; "Description" with an empty text area; a checkbox labeled "Keep this code private" which is unchecked; and "Organization" with a dropdown menu showing "None". At the bottom are two buttons: "Cancel" and "Publish Repository". A green arrow points from the text "Make sure your repo isn't private." to the "Keep this code private" checkbox.

Publish Repository ×

GitHub.com GitHub Enterprise

Name
test

Description

☐ Keep this code private

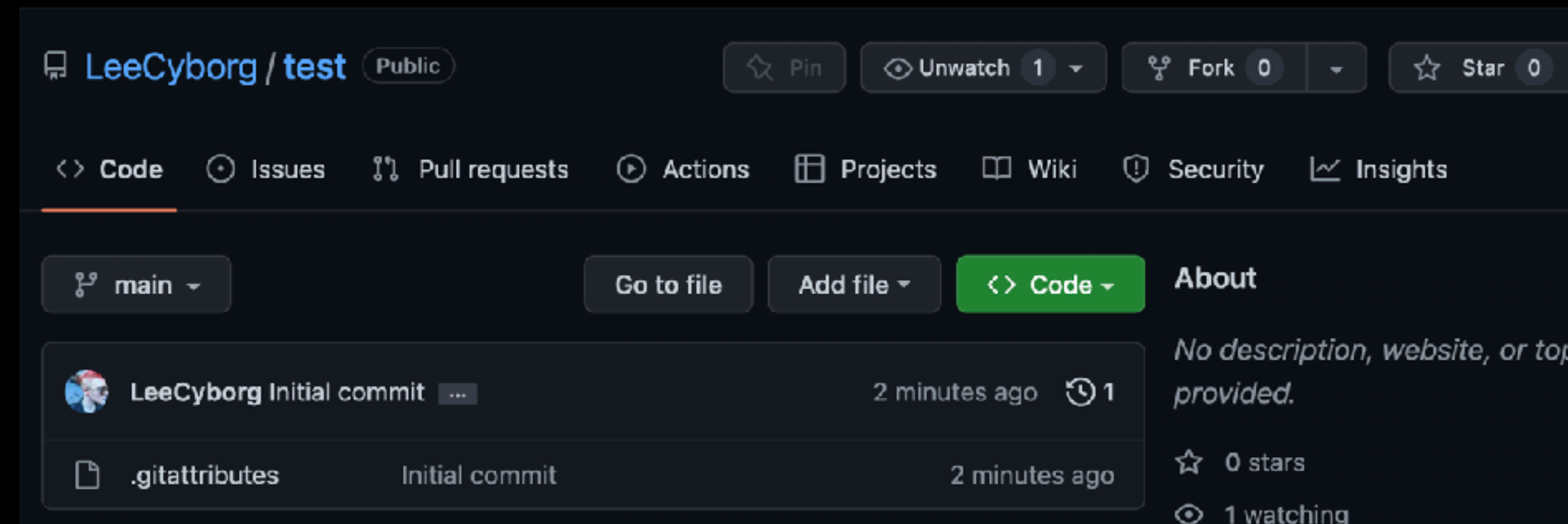
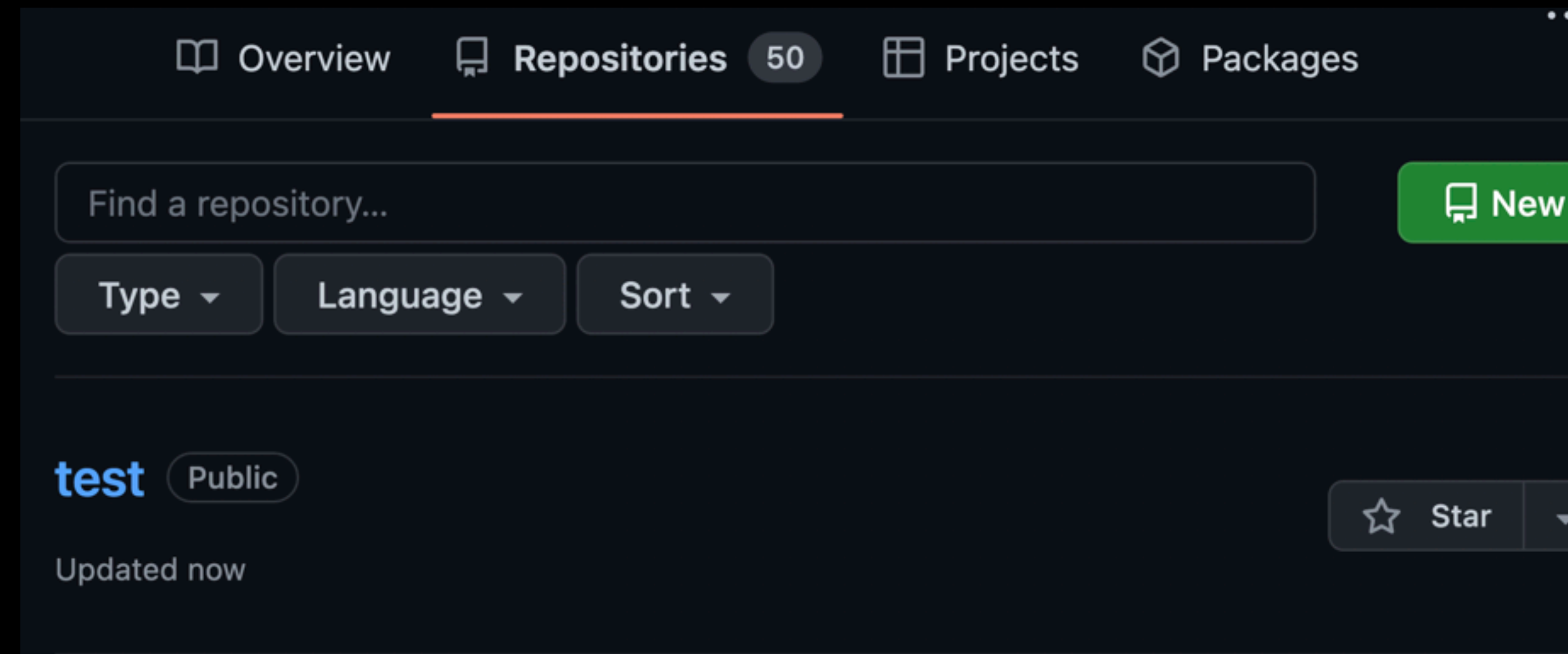
Organization
None ▾

Cancel Publish Repository

GitHub Desktop

On your Github page, you can now see this repository! You can add files manually if you want.

You now have a corresponding folder on your local computer which Github Desktop keeps track of.



Keeping in sync

No matter where you're updating your repo, you want to make sure the code is synced. When you update your local code, you should PUSH it to the website. When you update on the website, you should PULL your changes to your local computer. This way you're always editing the same code no matter where you are.

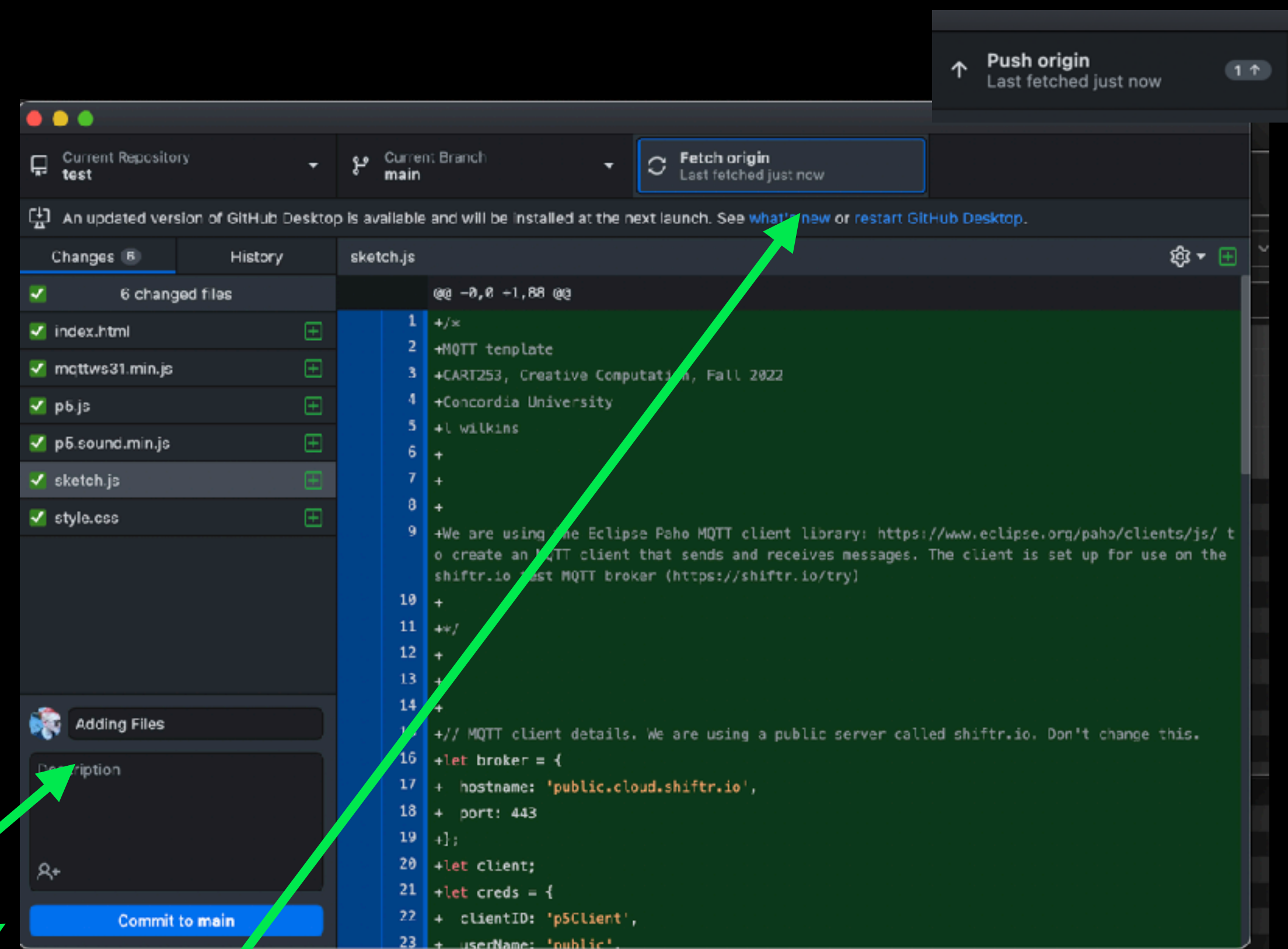
GitHub Desktop

By adding files to this folder on my computer, I can then commit and push them to my GitHub repository.

These are all the files that were changed or added, GitHub desktop checks for you!

When I am ready, I can write a COMMIT message and click “commit to main”

After its been committed, I can then PUSH it to the repository to update it.



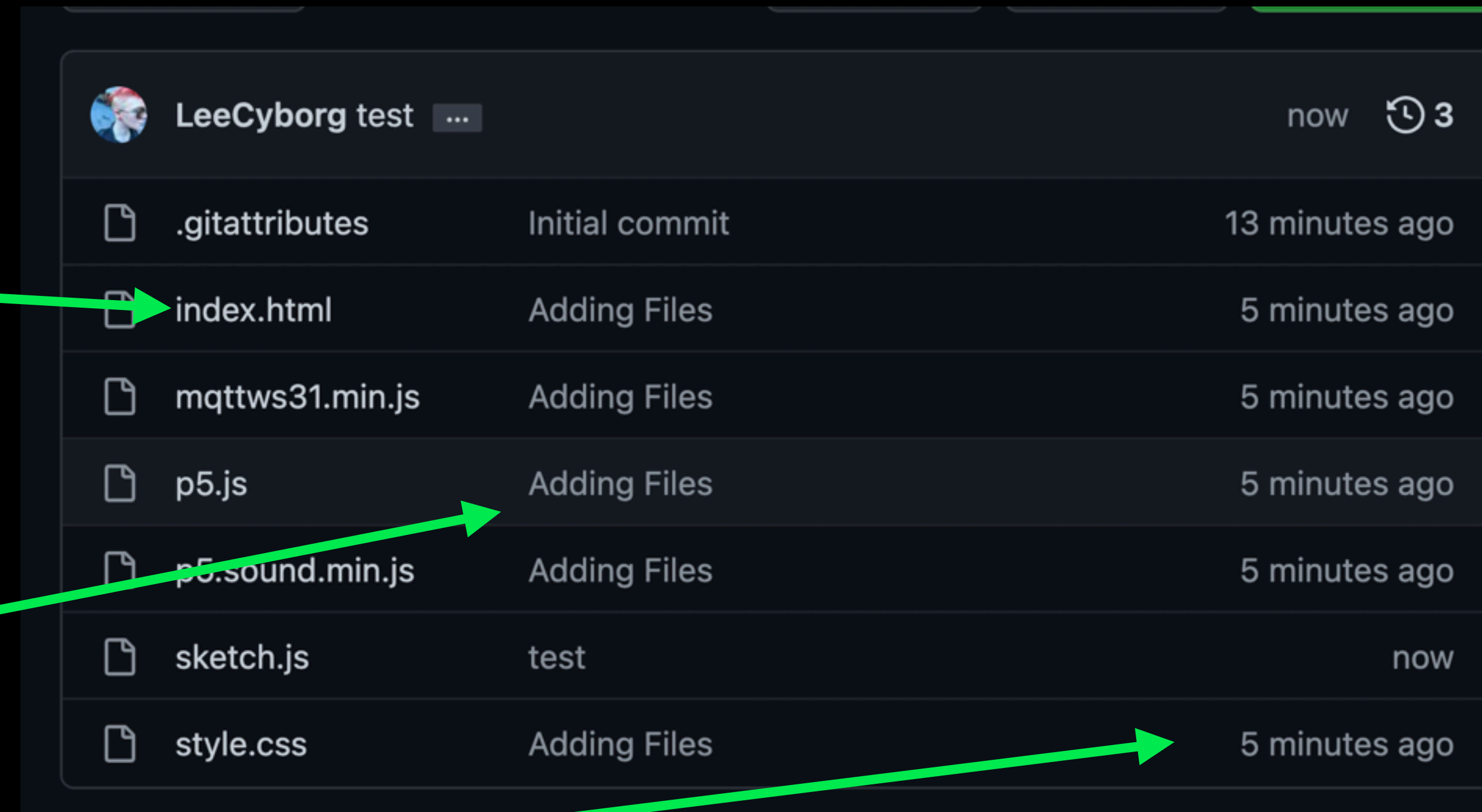
GitHub Desktop

Always check on the GitHub website to be sure your files are updated!

You'll see all the files listed here

You'll see your commit message here

You'll see when it was updated here.



Readme Files

README files are important part of documenting your code. If you make a file called README.md it will automatically appear immediately below your repository files, like in our course outline! The same for each folder inside your main folder.



Markdown

README files have an extension “.md” which stands for markdown. Its a way of formatting text. You can write markdown in Github or in any code editor.

Here is a great cheat sheet
[https://
www.markdownguide.org/
cheat-sheet/](https://www.markdownguide.org/cheat-sheet/)

CART263-W-23

CART 263: Creative Computation 2
– Winter 2023 – 3 credits

When?

January 9 – April 23, 2023
Section A: Tuesday, 13:30 – 17:30 in
EV 5.635.
Section B: Thursday, 13:30 – 17:30
in EV 5.815

Where?

Tuesdays – EV 5.635.
Thursdays – [image.pngimage.png](#).

Who?

Lee Wilkins
Department of Design and Computation
Arts
l.wilkins@concordia.ca
www.leecyb.org

TA: Tricia Enns
tricia.enns@gmail.com

CART 263: Creative Computation 2 – Winter 2023 – 3 credits

When?

January 9 – April 23, 2023
Section A: Tuesday, 13:30 – 17:30 in EV
5.635.
Section B: Thursday, 13:30 – 17:30 in EV
5.815

Where?

Tuesdays - EV 5.635.
Thursdays - [image.pngimage.png](#).

Markdown

This is a title

This is a smaller title

* This is a bullet point

****This is bold****

[This is a link](and the URL) (example: [Click Here]([google.com](https://www.google.com)))

CART263-W-23

CART 263: Creative Computation 2
– Winter 2023 – 3 credits

When?

January 9 – April 23, 2023

Section A: Tuesday, 13:30 – 17:30 in
EV 5.635.

Section B: Thursday, 13:30 – 17:30
in EV 5.815

Where?

Tuesdays – EV 5.635.

Thursdays – [image.pngimage.png](#).

Who?

Lee Wilkins

Department of Design and Computation
Arts

l.wilkins@concordia.ca

www.leecyb.org

TA: Tricia Enns

tricia.enns@gmail.com

CART 263: Creative Computation 2 – Winter 2023 – 3 credits

When?

January 9 – April 23, 2023

Section A: Tuesday, 13:30 – 17:30 in EV
5.635.

Section B: Thursday, 13:30 – 17:30 in EV
5.815

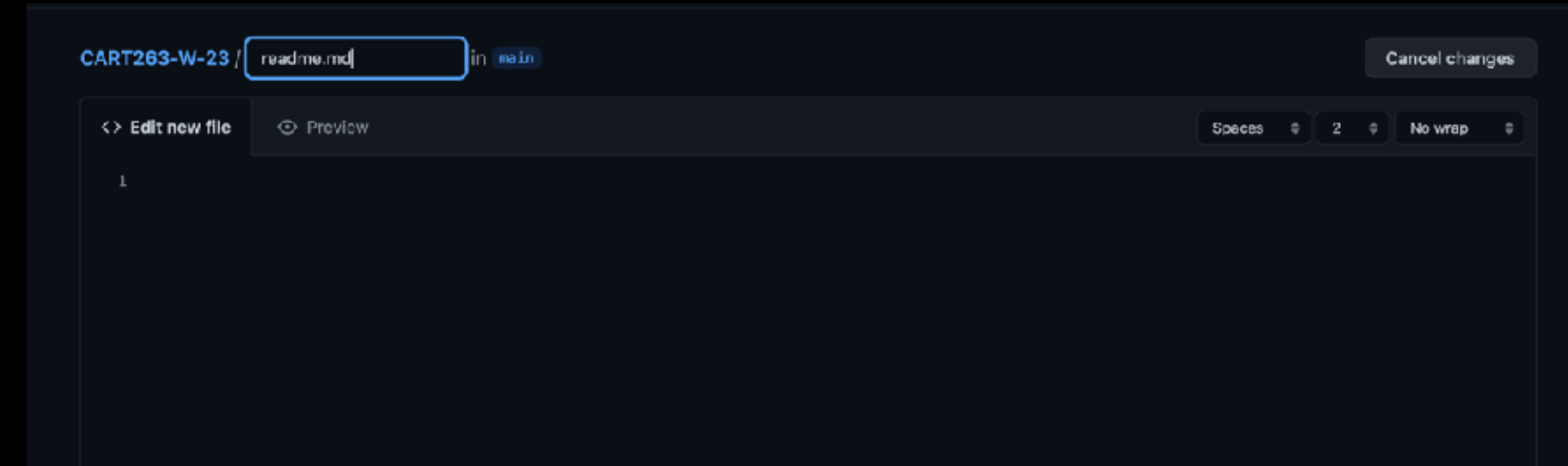
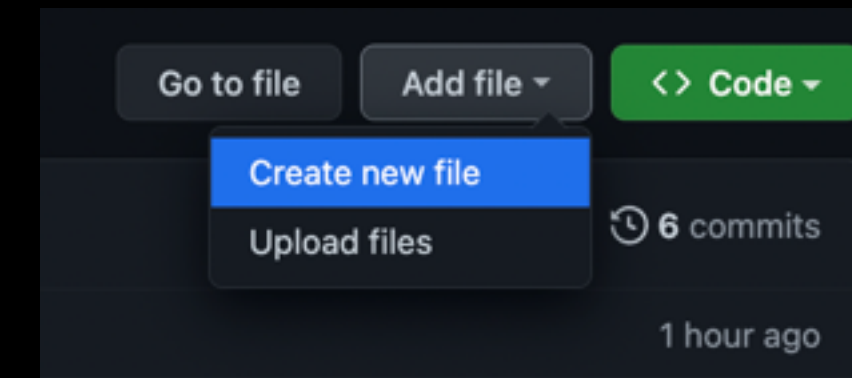
Where?

Tuesdays - EV 5.635.

Thursdays - [image.pngimage.png](#).

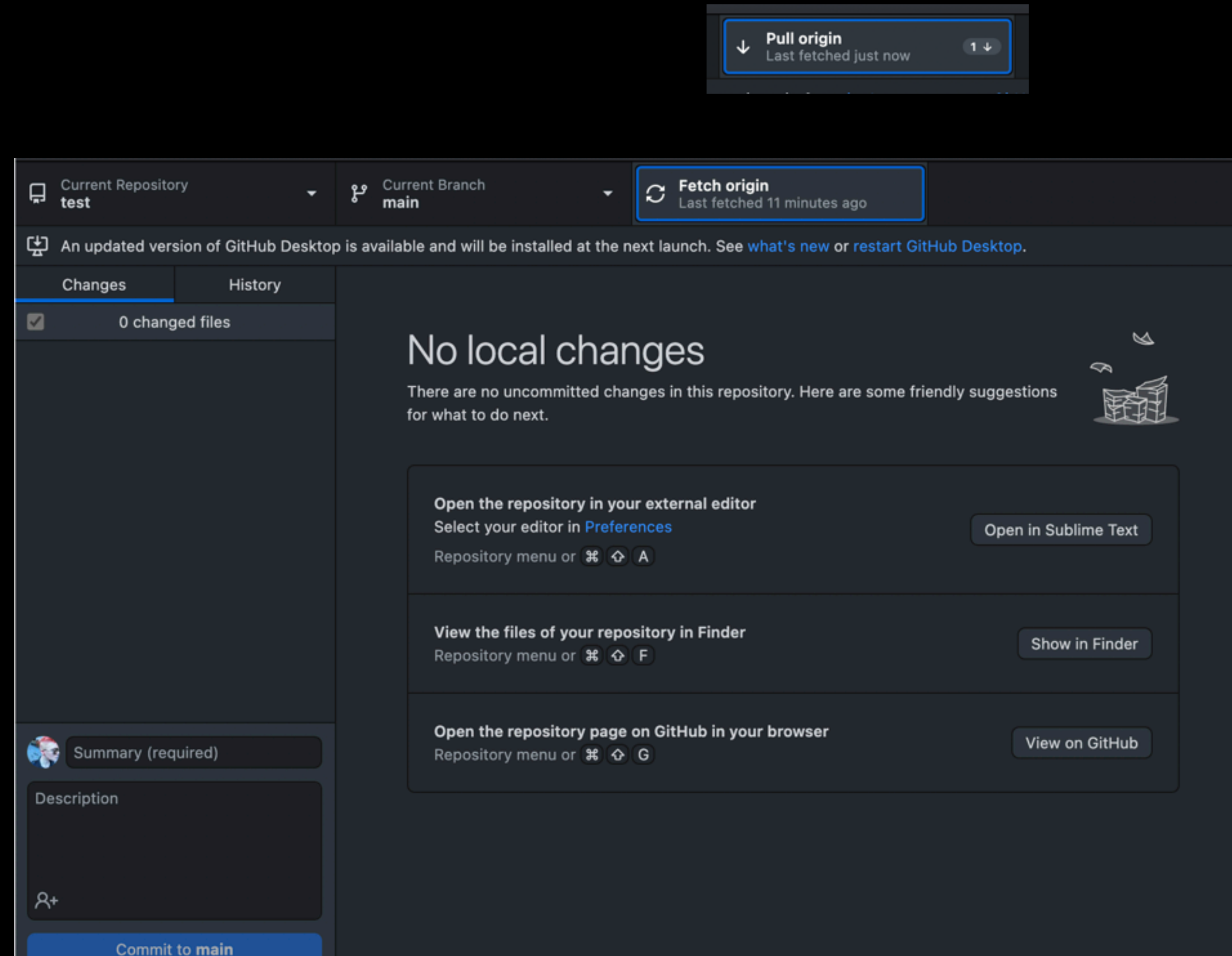
Markdown

We can make a new file on GitHub and call it `readme.md`. You can now write and preview your markdown! Save your changes when they are done.



Fetch and pull

If I've made changes online, and I want to update my local repo, I need to “fetch” and “pull” changes using GitHub desktop. Its good practice to do this every time you start up GitHub to keep your folders synced. This downloaded the new readme.md file we created.

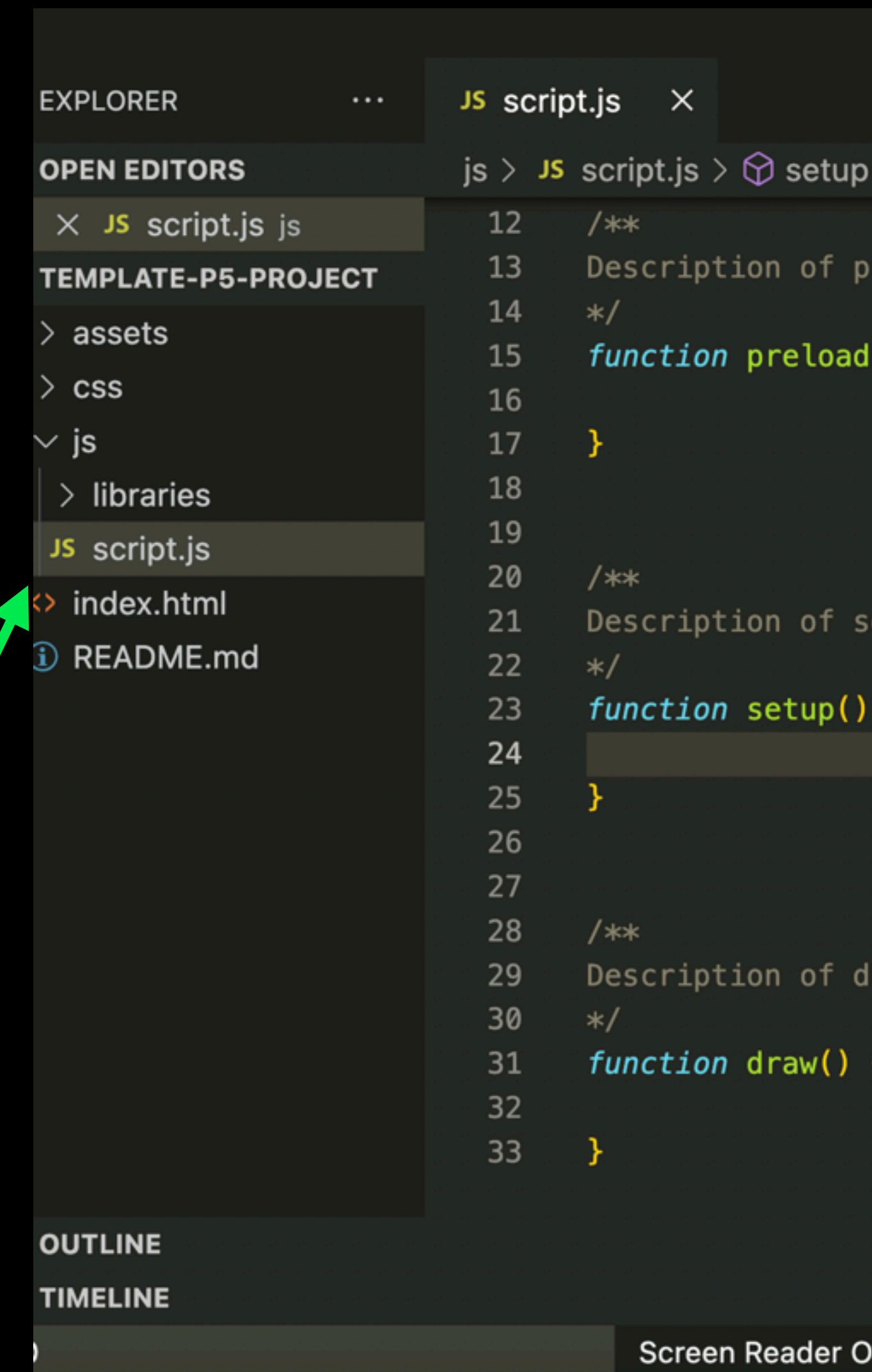


~~ATOM~~ VSCode

Because Atom will no longer be updated, we're going to us VSCode. You can download it [here](#), please use this going forward!

Getting set up

- Download the project template from Github or Moodle.
- **Unzip** it, and put the **contents** inside your new **repository folder**. Name the folder appropriately (Exercise-1)
- Open VS code, and open the folder you just created (File > Open Folder > Select your folder) You should see it here just like in Atom.

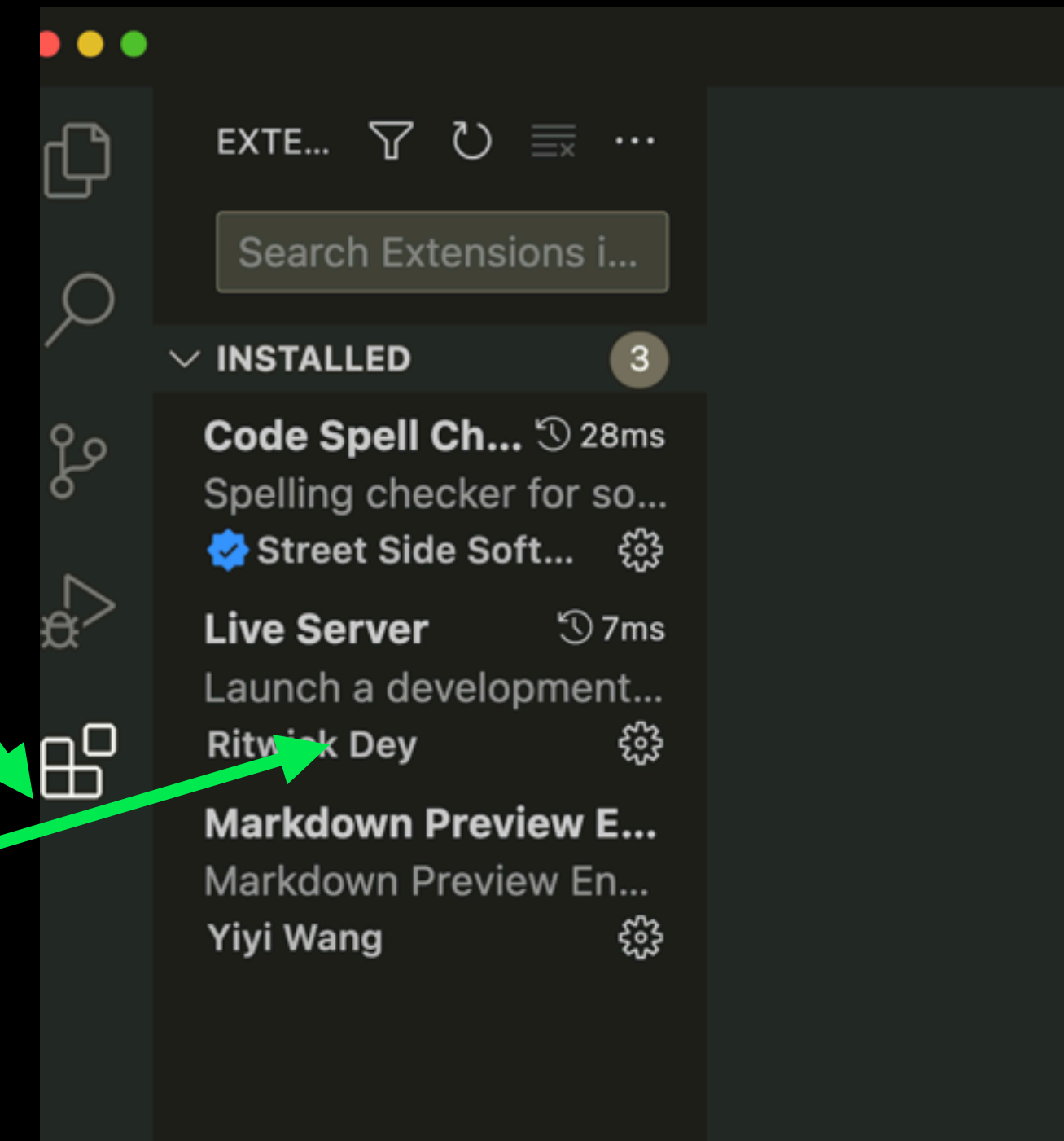


Getting set up

Install the live server extension:

- Click the Extension button on the side
- Search “Live Server” and click install

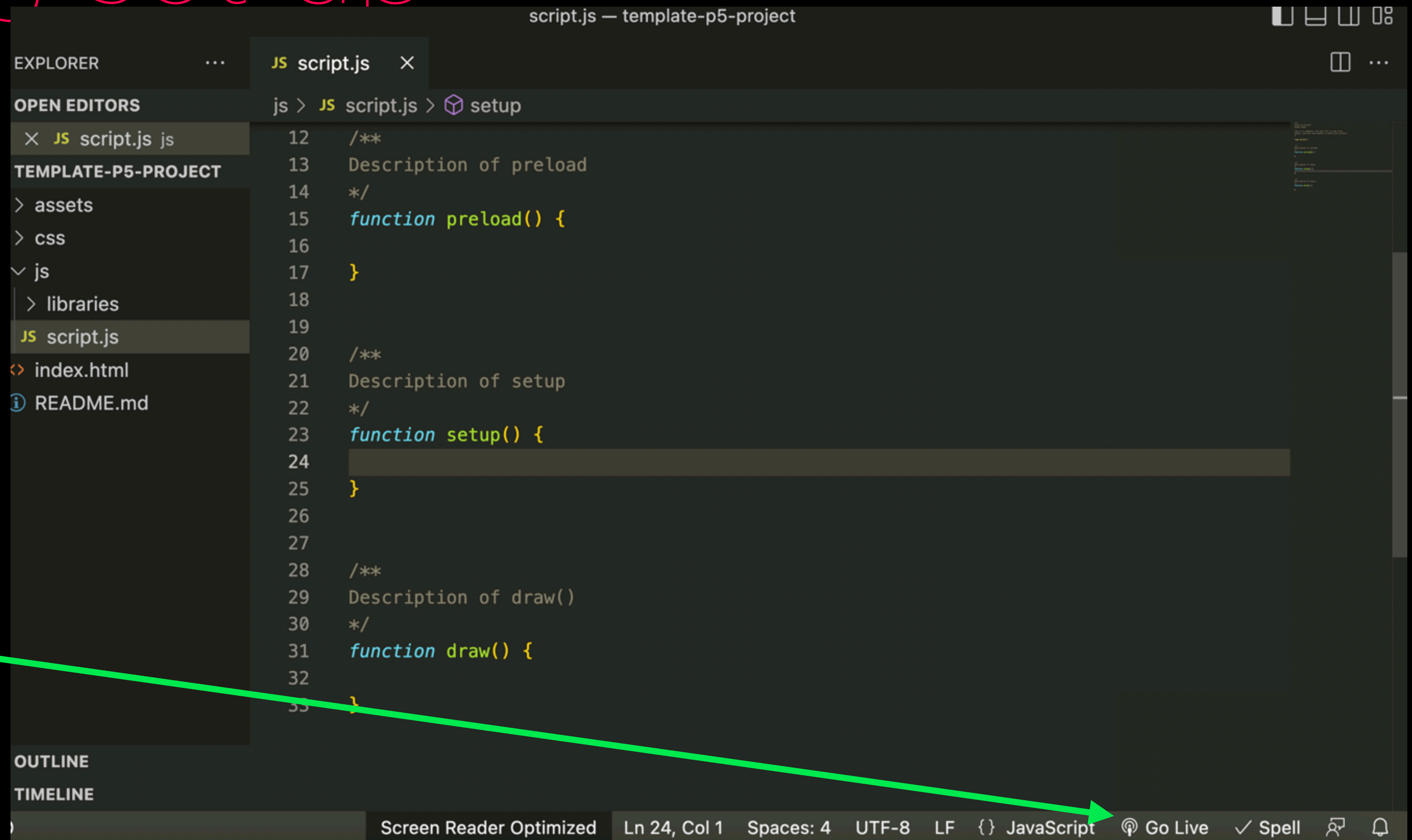
I also have a few other extensions, like markdown preview and spell check. Explore and download what you think is useful!



Getting set up

To run your code, click GO LIVE on the bottom corner of your screen. It should open your web browser just like Atom did.

You can also navigate to localhost:5500 in your browser if the server is on.



script.js — template-p5-project

EXPLORER

OPEN EDITORS

JS script.js js

TEMPLATE-P5-PROJECT

- > assets
- > css
- > js
- > libraries
- JS script.js
- <> index.html
- i README.md

```
12  /**
13  Description of preload
14  */
15  function preload() {
16
17  }
18
19
20  /**
21  Description of setup
22  */
23  function setup() {
24
25  }
26
27
28  /**
29  Description of draw()
30  */
31  function draw() {
32
33  }
```

OUTLINE

TIMELINE

Screen Reader Optimized Ln 24, Col 1 Spaces: 4 UTF-8 LF {} JavaScript Go Live ✓ Spell

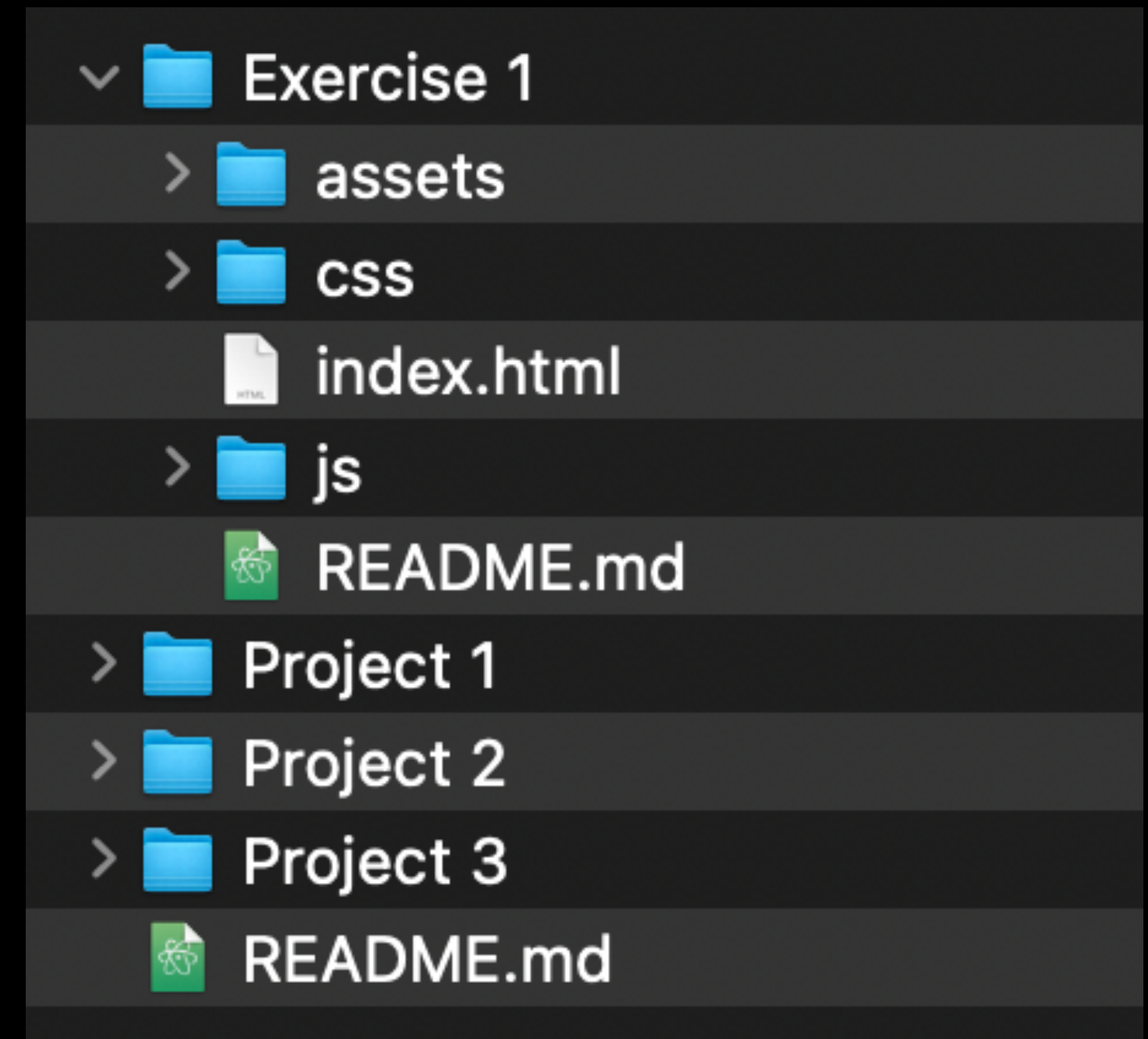
Make your first commit

Edit your code, and save it, and commit and push it!

- Make the canvas 500 by 500 pixels
- Make the background green
- Save your file
- Test your code
- Open up GitHub, fetch origin, write a commit message, and commit your code!
- Review it on [GitHub.com](https://github.com) and make sure your work is saved!

Structure your repository

By the end of the semester, your GitHub repository should look like this. There is a readme file for the whole repo, and individual folders with the p5js template, and their own readme files.



Exercise 1: Getting back into code

Due: Week 2.BEFORE CLASS

This project is an opportunity to quickly get back into practice in coding after the holidays. You can pick between one of the following classic programming problems to solve using P5js.

For this project, you should not use pre-existing code. Feel free to use the reference, or look up concepts, but you must code this all yourself. **Its okay if your code doesn't work fully, you're encouraged to explore and get as far as you can and get as far as you're able to.**

All code must be submitted via Github repository as well as on Moodle. You will be graded on:

- 75% Code is well commented and uploaded to Github and Moodle. Code comments should explain clearly what each part of the code does, functions and variables should be clearly named. You should have a clear readme.md file that explains how far you got and how it works.
- 25% functionality of code. The goal of the project is less about creating fully functional code, and more about exercising your brain after the break!

Exercise 1: Getting back into code

Option 1: Tic Tac Toe (https://www.exploratorium.edu/brain_explorer/tictactoe.html)

Create a TicTacToe game that is able to keep score and determine if there is a winner. It should be fully playable and have:

- 9 squares, 3x3 grid X and O should appear in the grid as each player takes a turn clicking
- When X or O has 3 in row, horizontally, vertically, or diagonally, the game displays a "win" screen and then clears the board
- After each game, X or O's score increases on the bottom of the screen.
- The players should be able to play an unlimited number of games.

Option 2: Pong (<https://www.ponggame.org/>)

Create a Pong game where 2 players bounce a ball back and forth horizontally to score points. Your game should have:

- 2 paddles that move vertically using Up & Down arrows, and W & S.
- A ball should bounce between the two paddles
- A point is scored when the ball passes the opponents's paddle without being hit.
- The ball should hit the walls of the screen and bounce on an angle.
- The first player to 10 points win

Exercise 1: Getting back into code

Its okay if your code doesn't work, most of your grade is about experimenting and learning. We will make a working game next class together, but come with your tips and tricks. Do not use online code. You can use the reference, but don't pull code from other games

Upload your code to GitHub AND Moodle **before** class next week