and

Python Training for NOAA GSL Interns

Summer 2022

What is  git ?

…a version control system

It lets you manage changes you've made to files over time.

You can jump back and forth through different **versions** of code

Save checkpoints, or **commits**, with notes, or **messages**, about that version

Enables a handy project-level "undo" button.

And you can see how/when things change over time.

Save multiple versions of your code side-by-side in different **branches**

This is especially helpful when you want to make your own sandbox in a repo that was shared with you via GitHub

All the git docs: https://git-scm.com/docs

A few more helpful terms…

A **clone** is local copy of the entire repository

A git **hash** is a 40 character pseudo-random hexadecimal string that uniquely identifies each commit

A **remote** repository refers to a repository stored elsewhere, maybe on GitHub

A **fork** is a copy of a a GitHub repository that you save under your own GitHub account

An **authoritative repository** is a repository that a fork is based on

Install Git

```
$ conda activate <my_env>
$ conda install -c anaconda git
```

Reminder: See all your conda environments with

```
$ conda env list
```

Setting up Git

Tell Git who you are

```
$ git config --global user.name “Your Name”
$ git config –global user.email “your.email@...”
```

Tell git which editor you’d like to use:

```
$ git config --global core.editor <editor name>
```

for Notepad on Windows:

```
$ git config --global core.editor \
"'C:/Program Files/Notepad++/notepad++.exe’ \
-multiInst -notabbar -nosession -noPlugin"
```

Popular editors include:

- emacs
- vim
- nano
- notepad

Start a repository in a folder

```
$ mkdir my_work
$ cd my_work
$ git init
$ git branch –m main
```

Staging Files

This step tells git that you'd like to *track a file*, or modifications to a file, in the repository

Linux/Mac

```
$ touch README
$ git add README
```

Windows

```
$ type nul > README
$ git add README
```

Current Repo Status

```
$ git status
On branch main

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will
be committed)
     README

nothing added to commit but untracked files present
(use "git add" to track)
```

Git’s view of reality

Files look like they did during last commit

After making a change, stage a file using the `git add` command

Add a new version of a file to the repo and update working state. Generates a new hash in history.

File States

Tracked

Git knows about a file and it exists in a snapshot of the repo

Cached

File has been staged for commit with "`git add`"

Untracked

File exists on disk, but git doesn't have information about its history

Tracked files may be in one of 3 states:

- Unmodified
- Modified
- Staged

Adding a new file

Puts a file into the *staging*, or *cached* state

```
$ git add README
```

Now what does `git status` look like?

```
$ git status
On branch main


No commits yet


Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
new file:   README
```

Seeing modifications to files

Open and edit the README file to add some content.
Now check the status again.

```
$ git status
On branch main
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
      modified:   README


no changes added to commit (use "git add" and/or "git commit -a")
```

How are the files modified?

```
$ git diff
diff --git a/README b/README
index e69de29..24192d8 100644
--- a/README
+++ b/README
@@ -0,0 +1,2 @@
+# Description
+
```

If you've already staged a file, use the `--cached` flag with `git diff` to see differences

```
$ git diff --cached
```

Committing Files

This step saves the file in the history of the repository

```
$ git commit -m "A message about these changes"
```

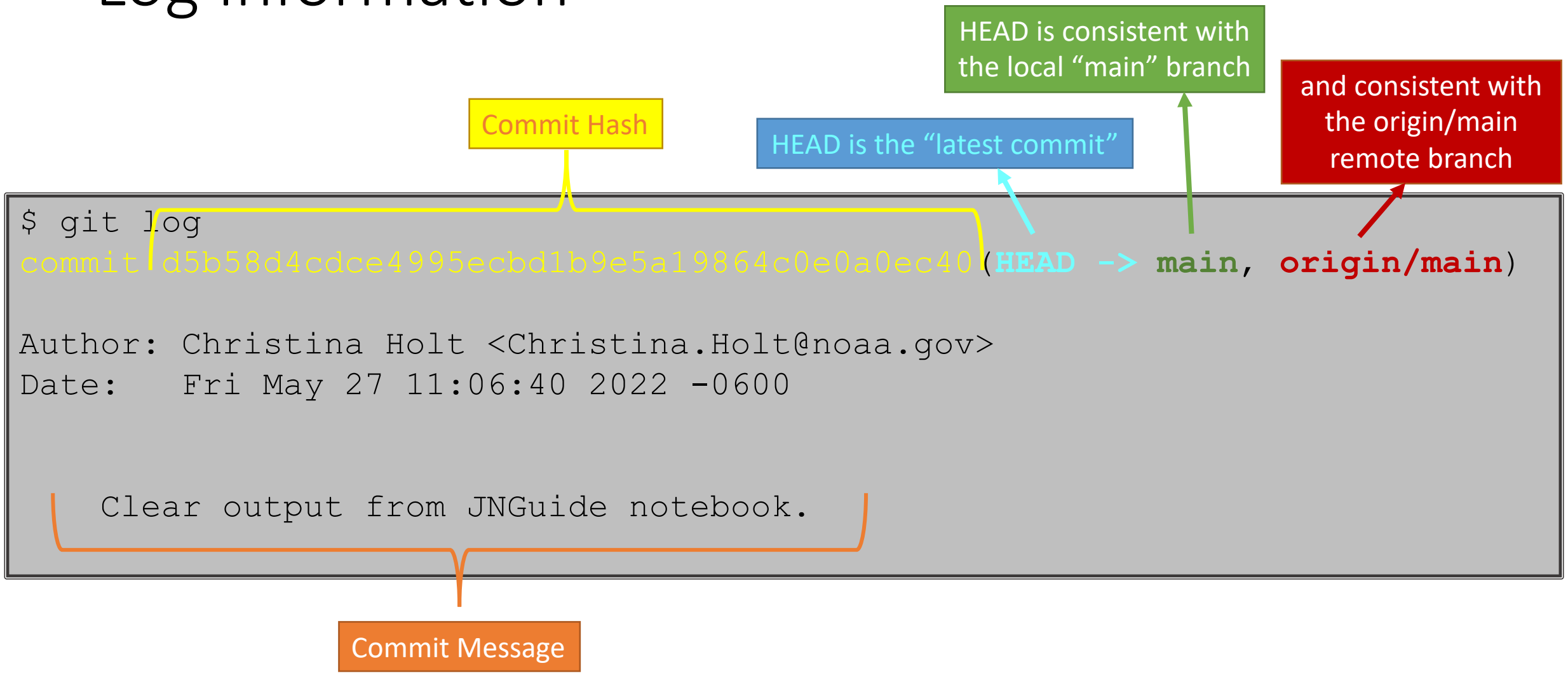
Without the `-m` flag, git will open your editor for you to add a message.

Repository history

```
$ git log
$ git log -p      # Shows the diffs in each commit
$ git log --stat # Shows the file changes in each commit
```

Output explained on next slide….

Log Information



Git Branches

Show all branches in a repository and all remote repositories

```
$ git branch -a
```

Create a new branch that is an exact replica of the branch you are currently on

```
$ git checkout -b <new_branch_name>
```

Newer versions of git use “switch” command instead

```
$ git switch -c <new_branch_name>
```

Compare your branch with another branch

```
$ git diff <other branch name>
```

A **cloud-based storage** service with an intuitive UI

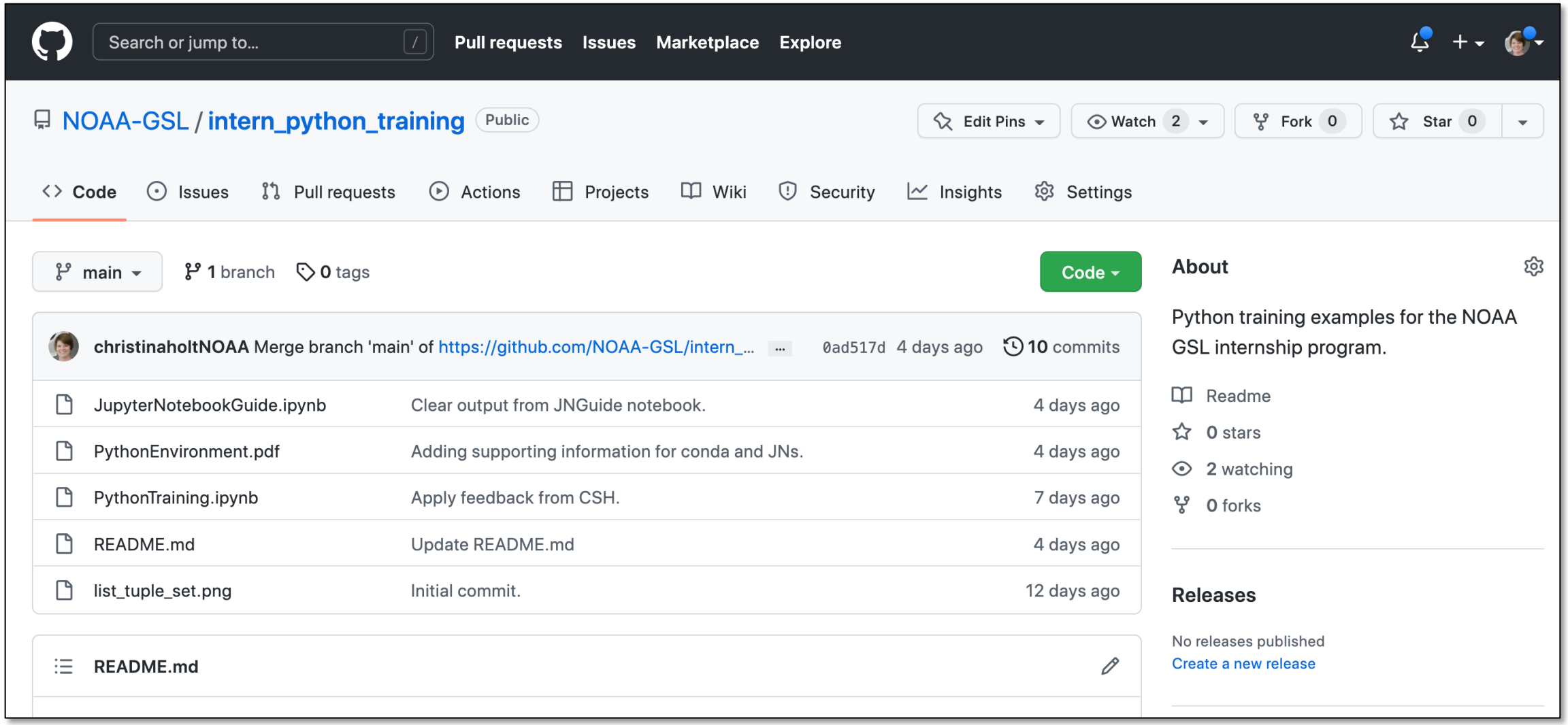
Use it as free backup for your code!

A **collaborative** service that enables teamwork

Share code with others

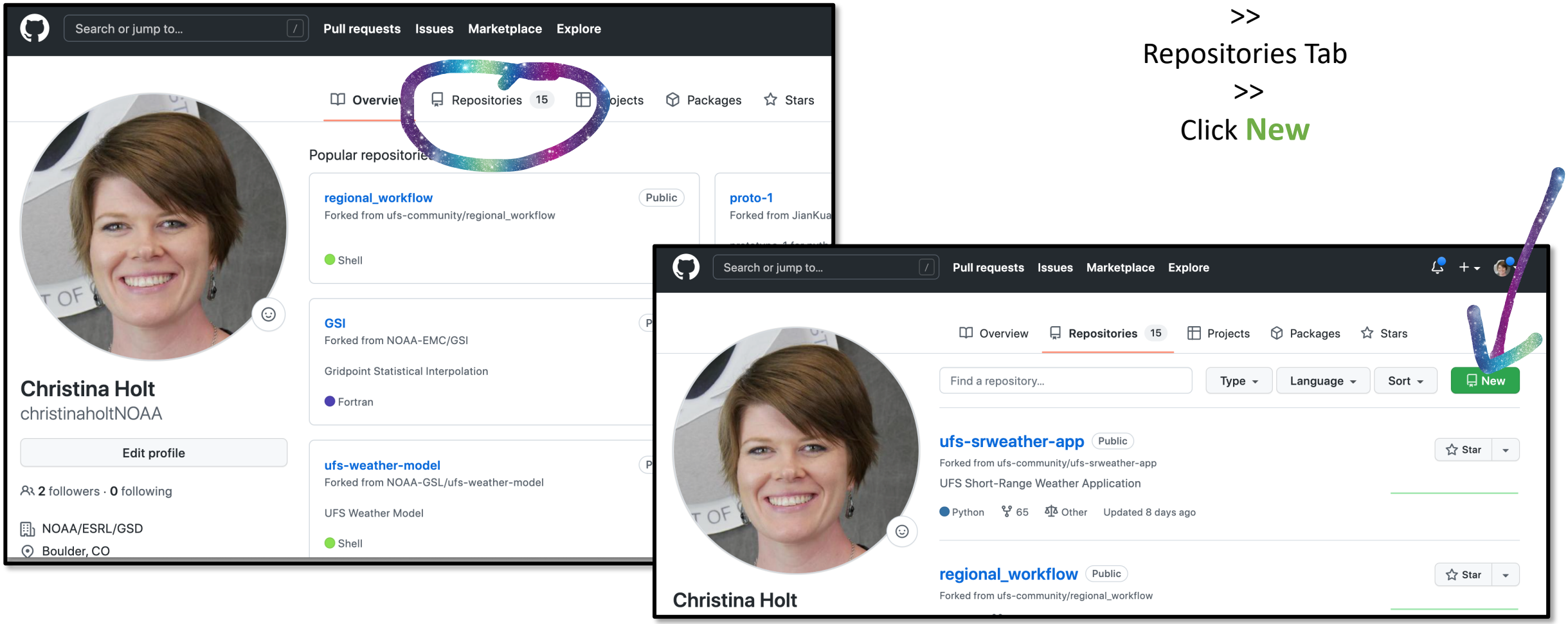
There are tons of other collaborative features and project management tools that we won't discuss here…

Let’s take a look at the training repo
https://github.com/NOAA-GSL/intern_python_training



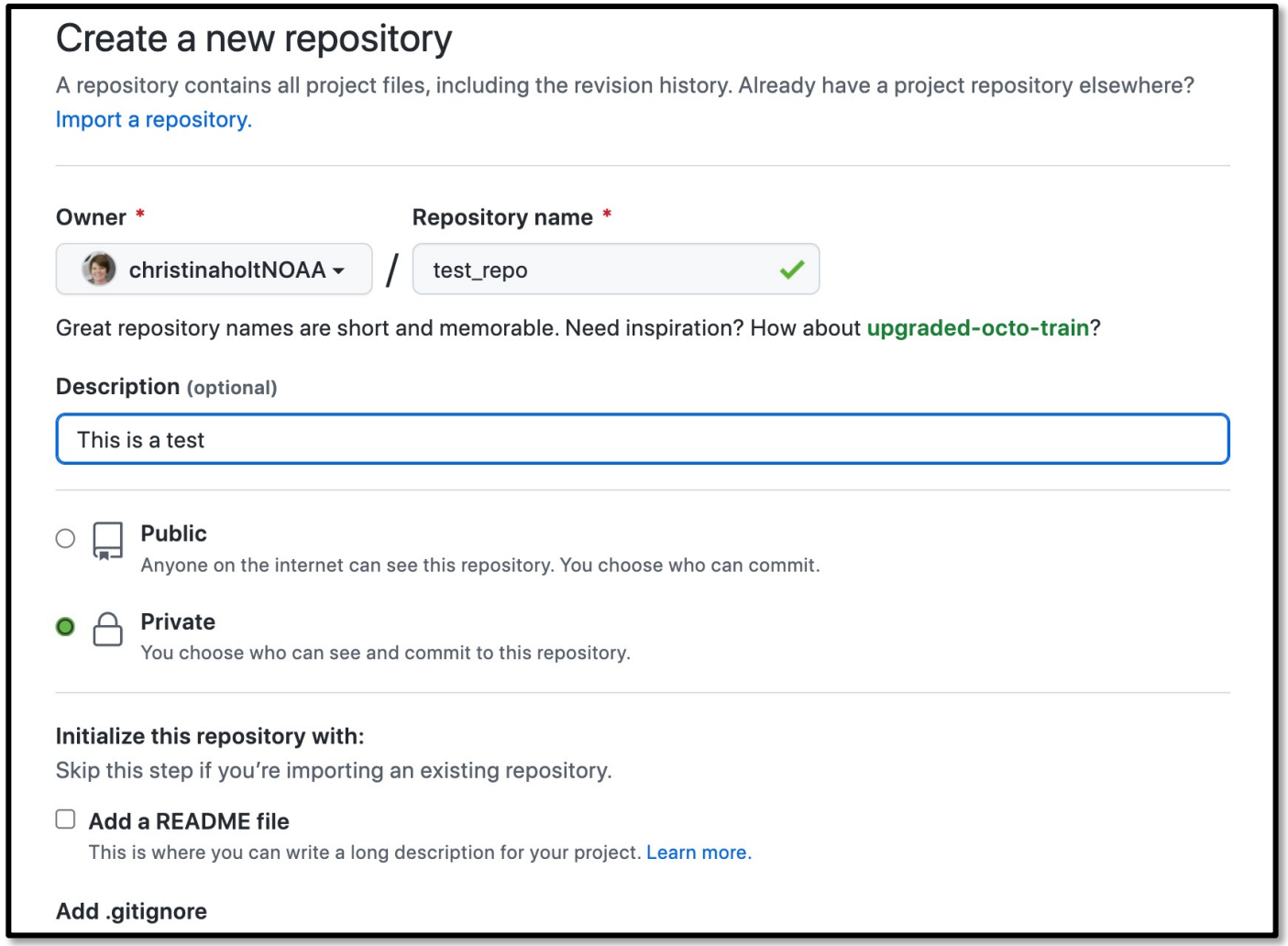
Create a new empty repository

Navigate to your GitHub Profile
>>
Repositories Tab
>>
Click **New**



Fill in the form

Create a private repo for the tutorial walk-through



Click **Create Repository** and follow the directions

Connecting with GitHub

Connect your local repository on disk with GitHub by adding a **remote** repository

```
$ git remote add origin <url>
$ git remote -v     # To see the labels and urls
```

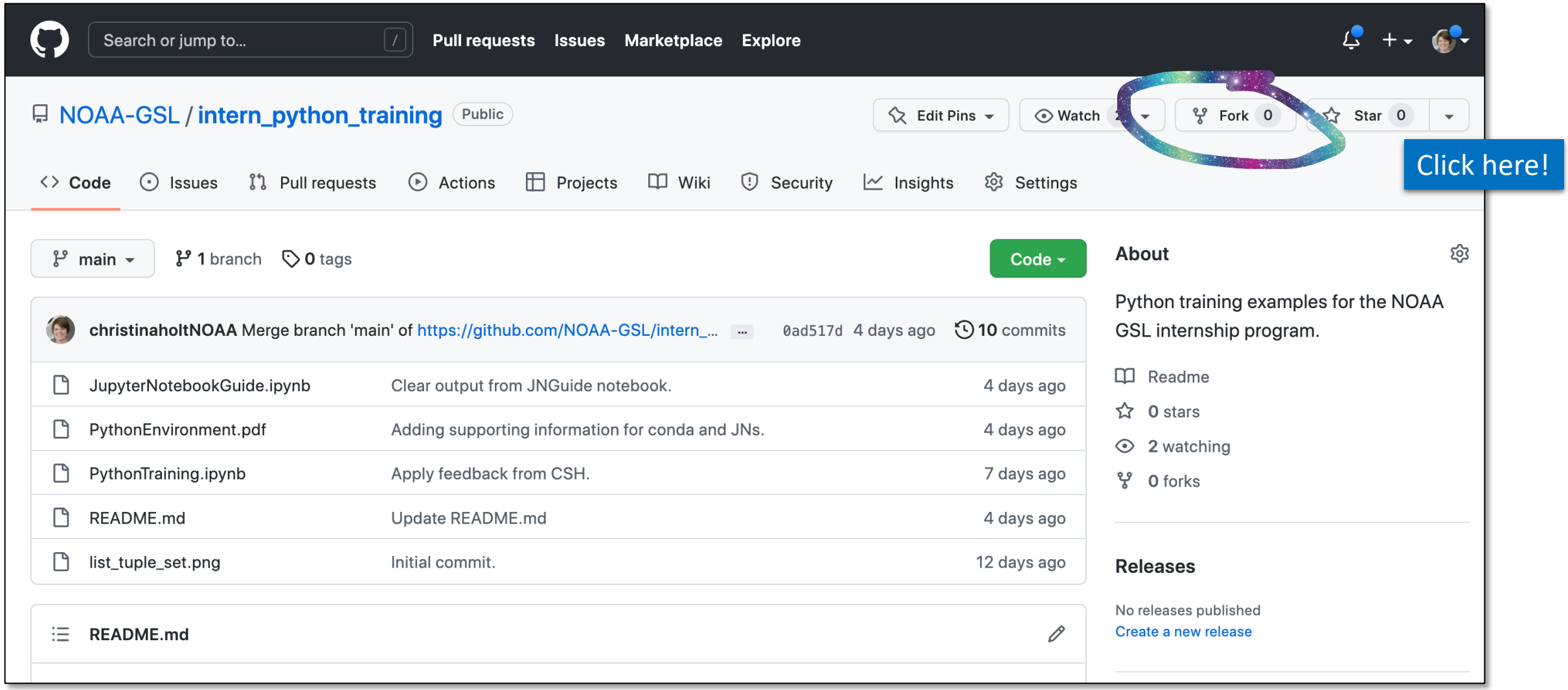
Push your local changes to GitHub

```
$ git push
$ git push -set-upstream-to origin/<branch-name>
```

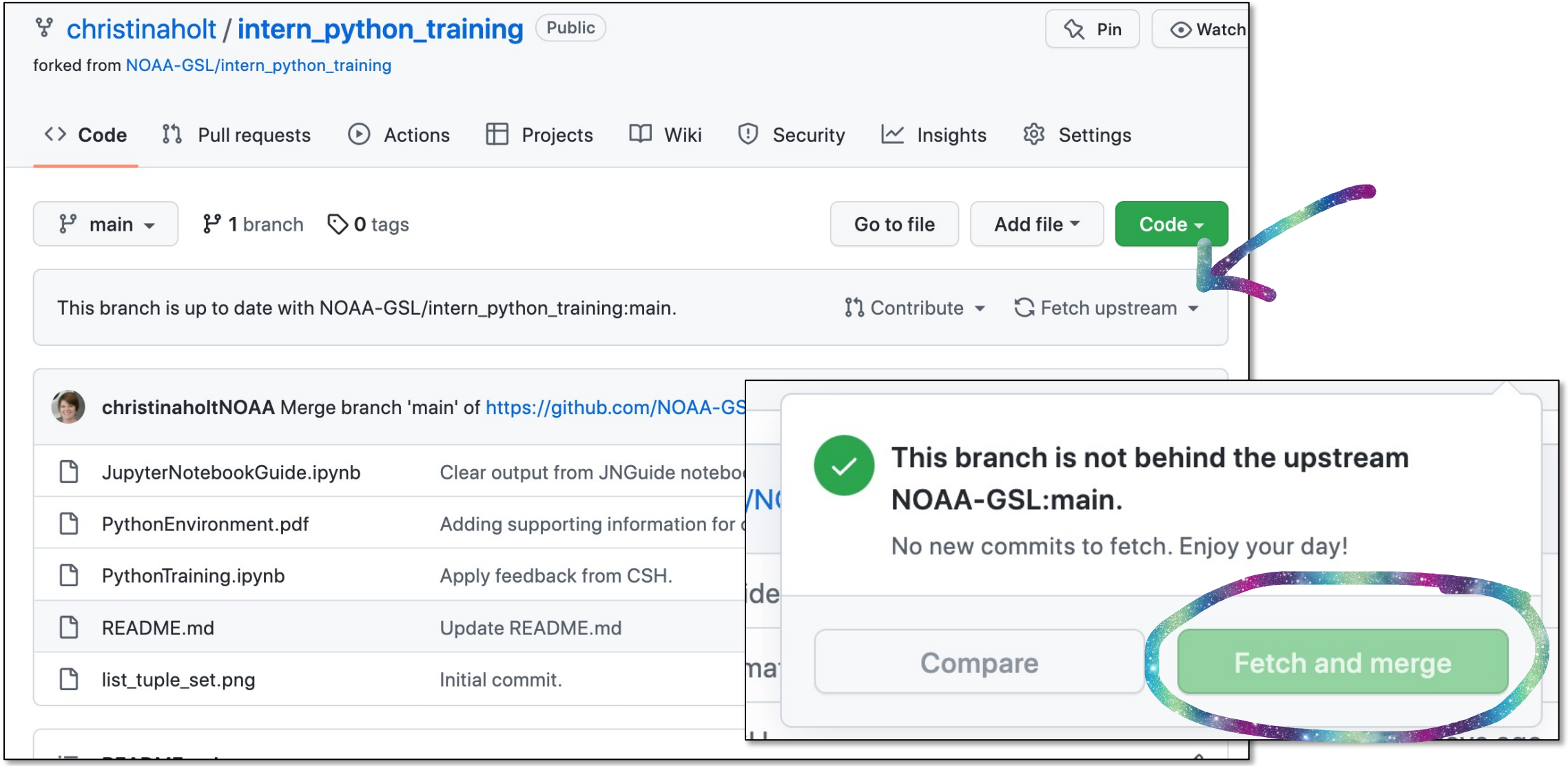
Fetch/Pull changes from GitHub

```
$ git fetch   # Retrieve changes, but don't
                apply them to the local code
$ git pull    # Retrieve changes and apply them
                to the local code
```

Fork a repository to have a copy of your own!



Pull in changes from your authoritative repo



Some helpful Git and GitHub Resources

- The official git tutorial: https://git-scm.com/docs/gittutorial
- An more thorough explanation of how git and GitHub work: https://www.youtube.com/watch?v=DVRQoVRzMIY
- GitHub Learning Lab Introduction to GitHub
- GitHub Getting Started