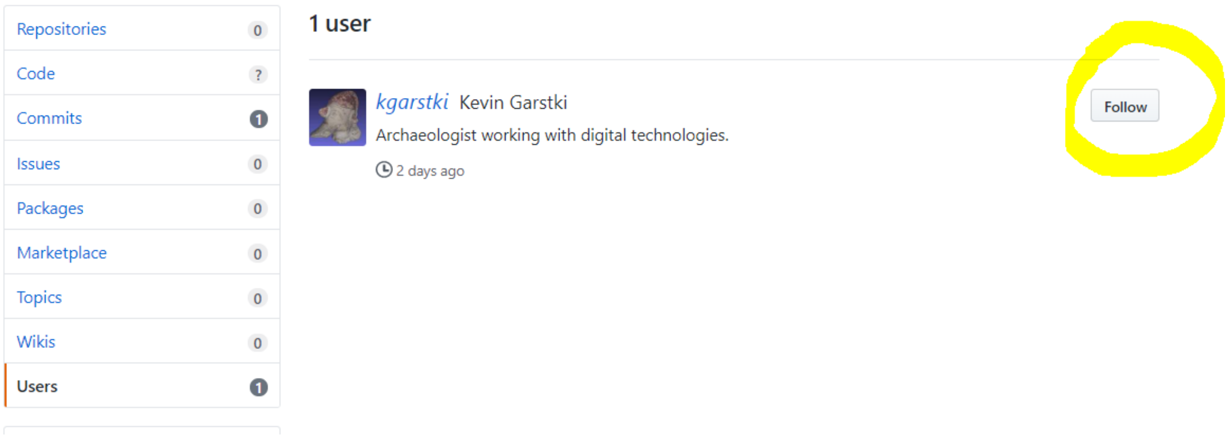
Week 2: Exercise 1 - GitHub

This activity was based on the instructions found here. The concepts and take-aways are aligned with those found in ODAT chapter 1.3, so make sure you read the accompanying text.

Getting started with GitHub

1. Go here and follow the instructions to start a GitHub account
2. When you have an account, search for "kgarstki" and follow me

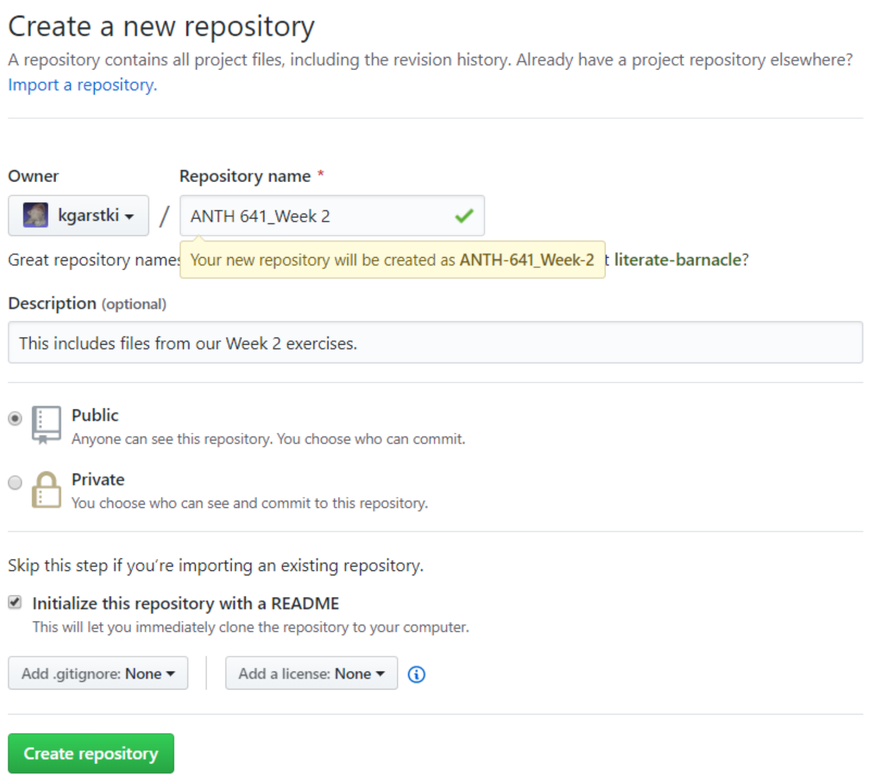


Archaeologist working with digital technologies.

Create a Repository in GitHub.

A repository is usually used to organize a single project. Repositories can contain folders and files, images, videos, spreadsheets, and data sets – anything your project needs. I recommend including a README, or a file with information about your project. GitHub makes it easy to add one at the same time you create your new repository.

1. When you're on your GitHub page, go to the upper right corner, next to your avatar or identicon, click + and then select **New repository**.
2. Name your repository "ANTH 641_Week 2"
3. Write a short description
4. Select **Initialize this repository with a README**.



Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? Import a repository.

Owner: kgarstki / Repository name *: ANTH 641_Week 2

Great repository name Your new repository will be created as ANTH-641_Week-2 t literate-barnacle?

Description (optional): This includes files from our Week 2 exercises.

[✓ Public] Anyone can see this repository. You choose who can commit.

Private You choose who can see and commit to this repository.

Skip this step if you're importing an existing repository.

[✓] Initialize this repository with a README
This will let you immediately clone the repository to your computer.

Add .gitignore: None | Add a license: None

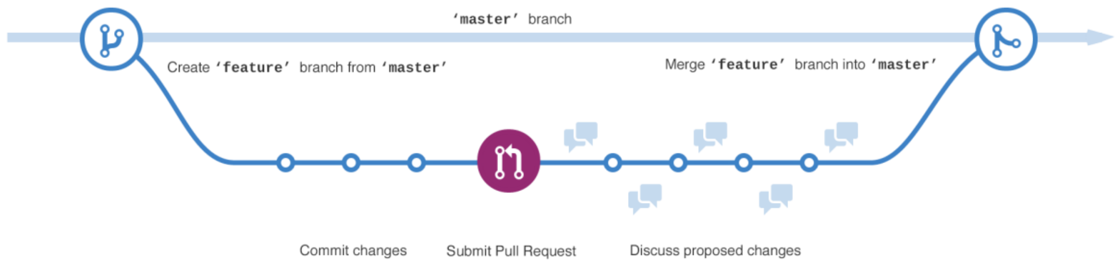
5. Click **Create repository.**

Create a Branch. **Branching** is the way to work on different versions of a repository at one time. By default your repository has one branch named master which is considered to be the definitive branch. We use branches to experiment and make edits before committing them to master.

When you create a branch off the master branch, you're making a copy, or snapshot, of master as it was at that point in time. If someone else made changes to the master branch while you were working on your branch, you could pull in those updates.

This diagram shows:

- The master branch
- A new branch called feature (because we're doing 'feature work' on this branch)
- The journey that feature takes before it's merged into master


'master' branch
Create 'feature' branch from 'master'
Merge 'feature' branch into 'master'
Commit changes
Submit Pull Request
Discuss proposed changes

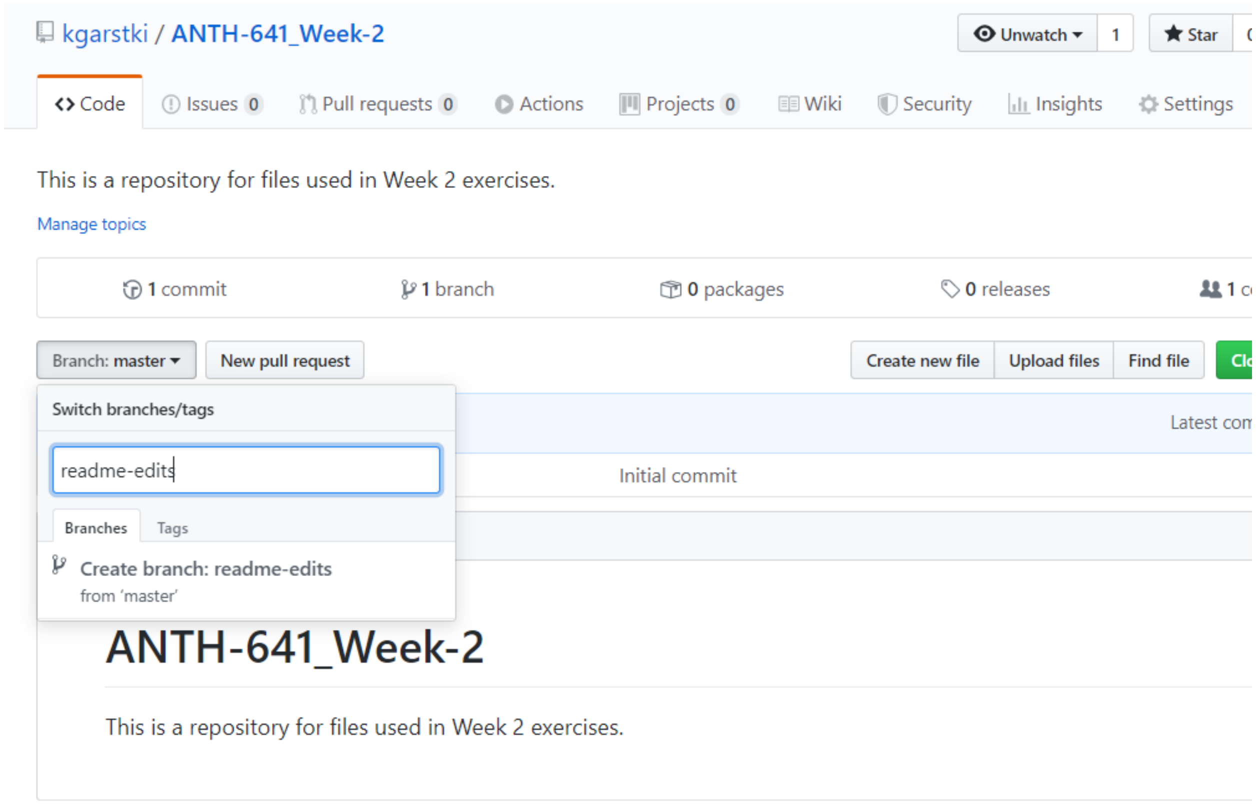
Have you ever saved different versions of a file? Something like:

- story.txt
- story-joe-edit.txt
- story-joe-edit-reviewed.txt

Branches accomplish similar goals in GitHub repositories.

To Create a New Branch:

1. Go to your repository "ANTH 641_Week 2"
2. Click the drop down at the top of the file list that says **branch: master**
3. Type a branch name, "readme-edits", into the new branch text box.
4. Select the blue Create branch box or hit "Enter" on your keyboard.



This is a repository for files used in Week 2 exercises.

ANTH-641_Week-2

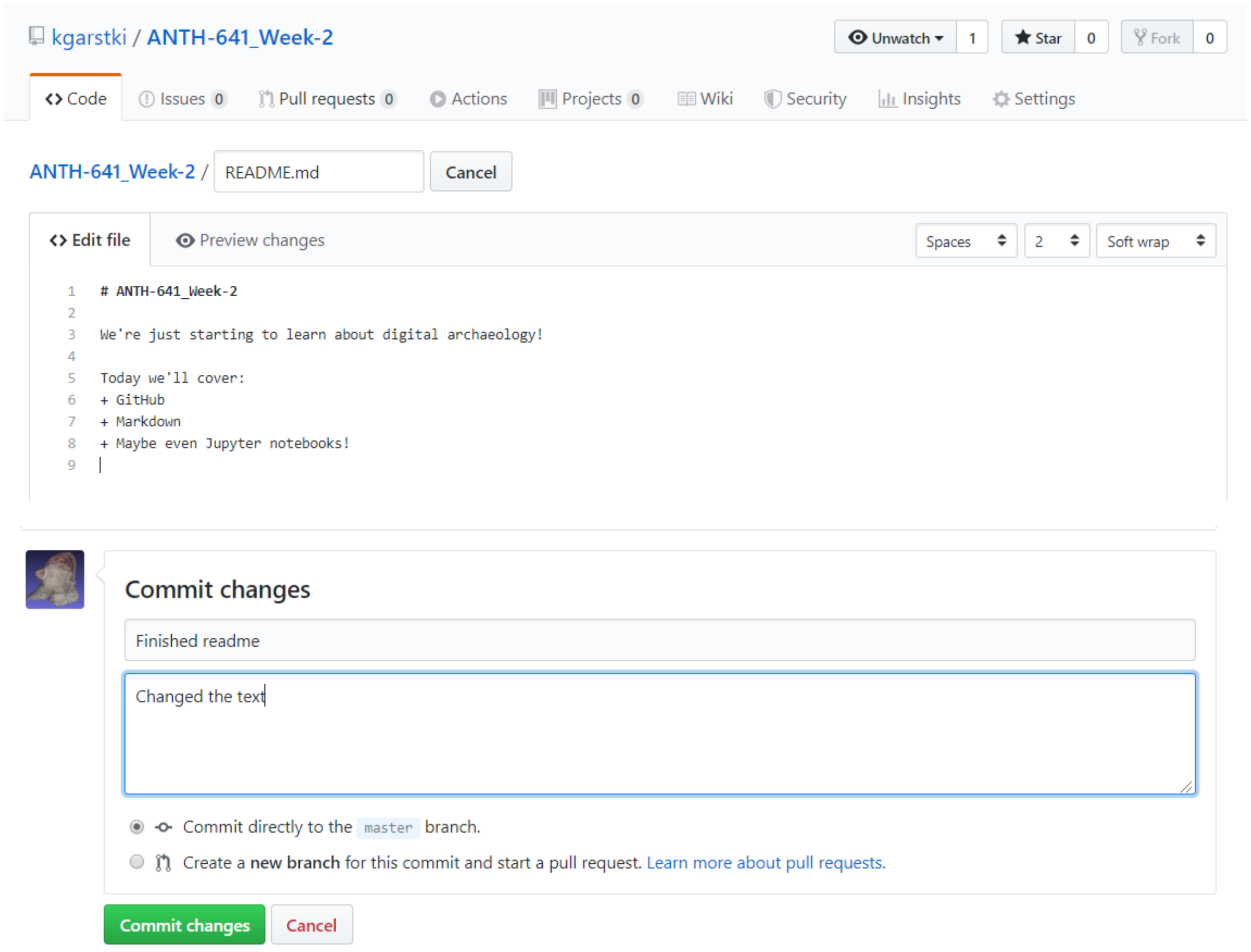
This is a repository for files used in Week 2 exercises.

Now you have two branches, “master” and “readme-edits”. They look exactly the same, but not for long! Next we’ll add our changes to the new branch.

Make and commit changes. Now, you’re on the code view for your `readme-edits` branch, which is a copy of `master`. Let’s make some edits.

On GitHub, saved changes are called *commits*. Each commit has an associated *commit message*, which is a description explaining why a particular change was made. Commit messages capture the history of your changes, so other contributors can understand what you’ve done and why.

1. Click the `README.md` file.
2. Click the pencil icon in the upper right corner of the file view to edit.
3. In the editor, write a bit about yourself.
4. Write a commit message that describes your changes.
5. Click **Commit changes** button.



Spaces: 2, Soft wrap

```
# ANTH-641_Week-2

We're just starting to learn about digital archaeology!

Today we'll cover:
+ GitHub
+ Markdown
+ Maybe even Jupyter notebooks!

```

Finished readme

Changed the text

[✓ Commit directly to the `master` branch.]

Create a **new branch** for this commit and start a pull request.

These changes will be made to just the README file on your `readme-edits` branch, so now this branch contains content that’s different from `master`.

Open a Pull Request

Now that you have changes in a branch off of master, you can open a pull request. Pull Requests are the heart of collaboration on GitHub. When you open a pull request, you're proposing your changes and requesting that someone review and pull in your contribution and merge them into their branch. Pull requests show diffs, or differences, of the content from both branches. The changes, additions, and subtractions are shown in green and red.

As soon as you make a commit, you can open a pull request and start a discussion, even before the code is finished. You can even open pull requests in your own repository and merge them yourself. It's a great way to learn the GitHub flow before working on larger projects.

1. Click the **Pull Request** tab, then from the Pull Request page, click the green **New pull request** button.
2. In the **Example Comparisons** box, select the branch you made, `readme-edits`, to compare with `master` (the original).
3. Look over your changes in the diffs on the Compare page, make sure they're what you want to submit.
4. When you're satisfied that these are the changes you want to submit, click the big green **Create Pull Request** button.
5. Give your pull request a title and write a brief description of your changes. Then click on the green **Create pull request** button.

Merge your Pull Request

In this final step, it's time to bring your changes together – merging your `readme-edits` branch into the `master` branch.

1. Click the green **Merge pull request** button to merge the changes into `master`.
2. Click **Confirm merge**.
3. Go ahead and delete the branch, since its changes have been incorporated, with the **Delete branch** button in the purple box.

You did it!! Review the terms you learned

- repository: a single folder that holds all of the files and subfolders of your project
- commit: this means, 'take a snapshot of the current state of my repository'
- publish: take my folder on my computer, and copy it and its contents to the web as a repository at github.com/myusername/repositoryname
- sync: update the web repository with the latest commit from my local folder
- branch: make a copy of my repository with a 'working name'
- merge: fold the changes I have made on a branch into another branch
- fork: to make a copy of someone else's repo
- clone: to copy an online repo onto your own computer

- pull request: to ask the original maker of a repo to ‘pull’ your changes into their master, original, repository
- push: to move your changes from your computer to the online repo
- conflict: when two commits describe different changes to the same part of a file