

Version control and GitHub

Why use version control?

"FINAL".doc



FINAL.doc!



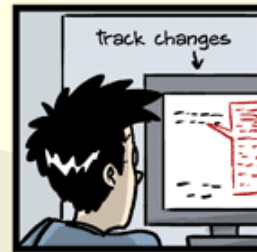
FINAL_rev.2.doc



FINAL_rev.6.COMMENTS.doc



FINAL_rev.8.comments5.
CORRECTIONS.doc



FINAL_rev.18.comments7.
corrections9.MORE.30.doc



FINAL_rev.22.comments49.
corrections.10.#@\$%WHYDID
ICOMETOGRADSCHOOL?????.doc

JORGE CHAM © 2012

Version control and Git

- Main features of a version control system:
 1. Save each new set of changes sequentially
 2. Keep track of different versions of a document/project
 3. Merge changes from multiple versions
- **Git** is a specific version control **system**
 - Think “track changes” in Word + Dropbox, but much more general and powerful
- A whole new system to learn. Is it worth the effort?
 - Maybe not when working alone
 - But critical to avoid disaster when collaborating
- The gold standard in the private sector – used EVERYWHERE

GitHub

- GitHub is a **specific website** that uses Git to host projects in the cloud
- We will use GitHub at a few points in this course
 - Lecture slides
 - Assignment 2
 - Term project
- Why?
 - To start building habits of using version control
 - To get you used to the basic terminology and actions of Git and GitHub
- “Real” developers & data scientists use Git at the command line
 - I’m not going to require you to do that now
 - But I encourage you to learn it on your own

Getting set up with GitHub

1. Create a GitHub account (unless you already have one)
2. Download GitHub Desktop
3. Connect GitHub Desktop to your GitHub account

1. Create a GitHub account

- Go here and fill out the forms: <https://github.com/>
- No need to apply for the GitHub Student benefits (though you can if you want to)

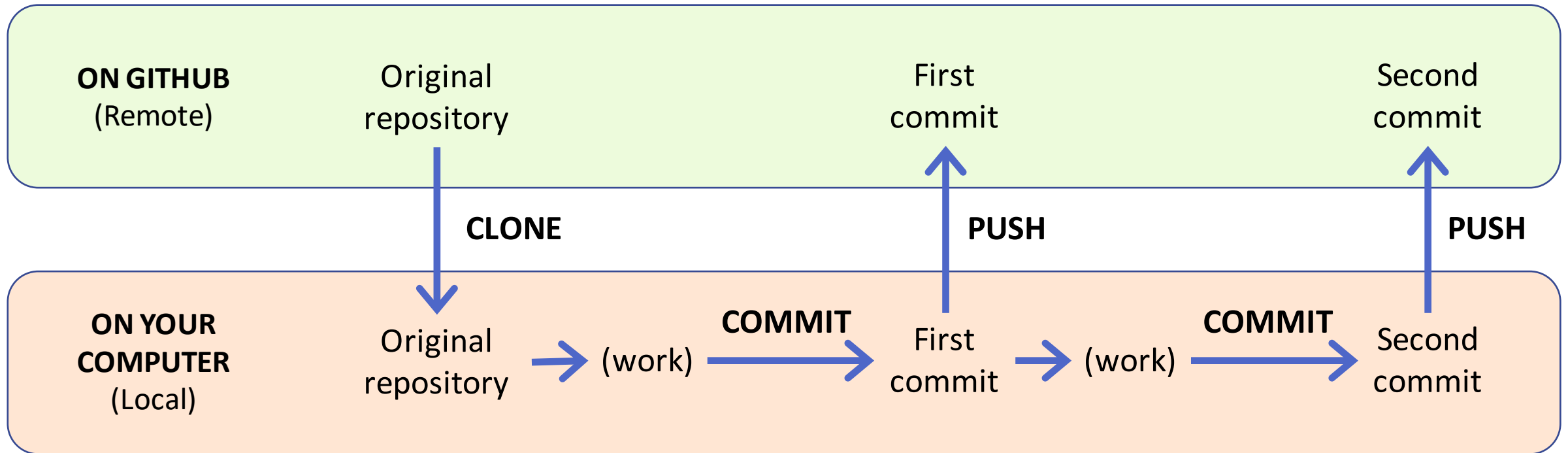
2. Download GitHub Desktop

- Go here: <https://desktop.github.com/>
- GitHub Desktop is a standalone app for using Git and GitHub through a graphical user interface (GUI).
- Recommend but not strictly required
 - You can make changes directly through the GitHub website, but it will end up being harder in many ways
 - You can use Git at the command line (shell) if you already know it or want to learn

3. Connect GitHub Desktop to your GitHub account

- Open GitHub Desktop and go to File -> Options
- If you need help, try this:
<https://docs.github.com/en/desktop/installing-and-configuring-github-desktop/installing-and-authenticating-to-github-desktop/setting-up-github-desktop>

Basic workflow (only 1 contributor)

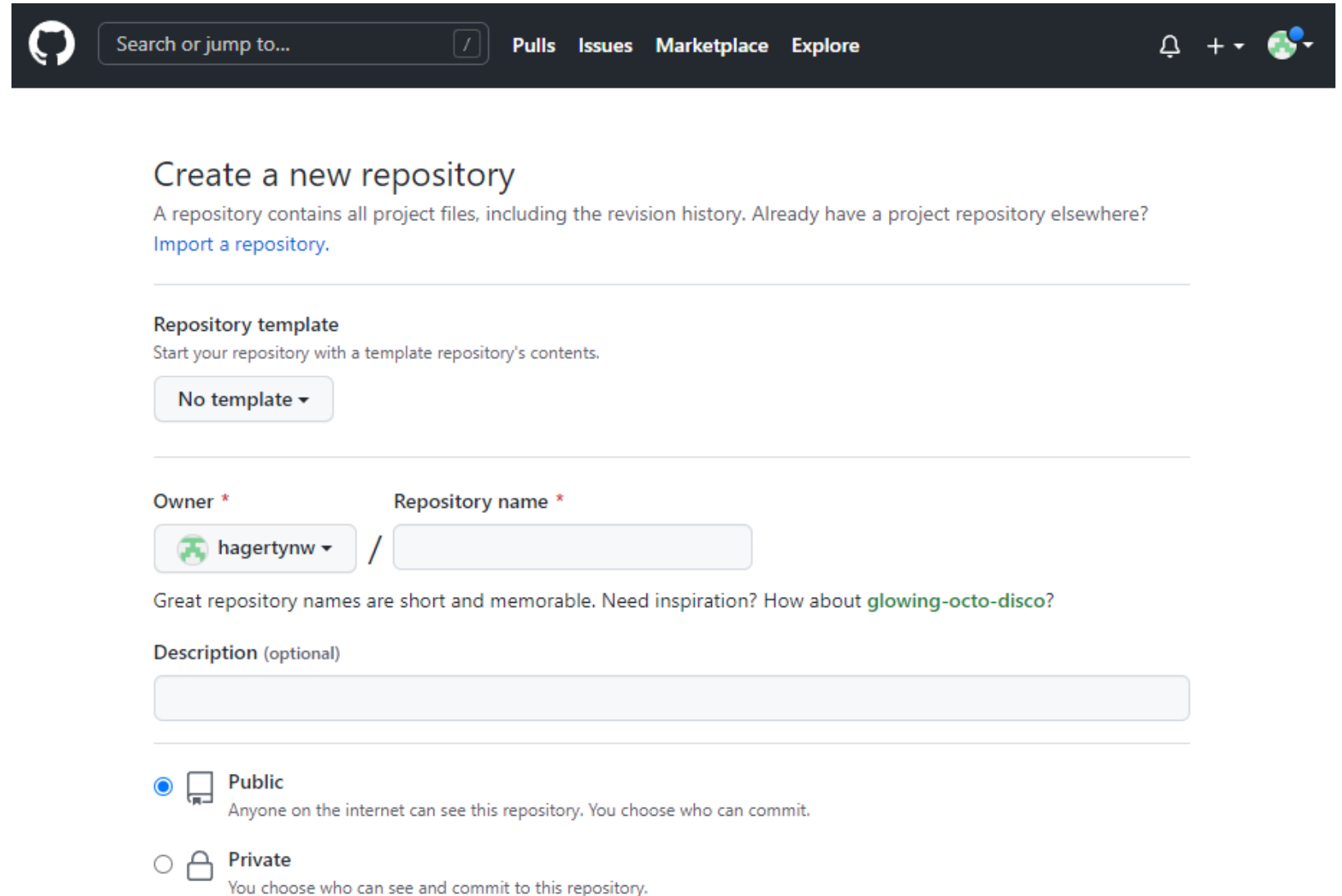


Workflow for your project

1. On GitHub.com, create a new repository
2. **Clone** this repository to your local machine
3. Do some work (edit the repository)
4. **Commit** changes (i.e., save a draft)
5. **Push** your commit to GitHub (back it up to the cloud)

1. Create a new repository

A repository (**repo**) is the full record of a project folder and all its changes ever.



The screenshot shows the GitHub interface for creating a new repository. At the top is a dark navigation bar with the GitHub logo, a search bar, and links for Pulls, Issues, Marketplace, and Explore. Below this is the main heading 'Create a new repository' with a subtext explaining that a repository contains all project files and revision history. There is a link to 'Import a repository.' for existing projects. The 'Repository template' section has a dropdown menu set to 'No template'. The 'Owner' field is set to 'hagertynw' and the 'Repository name' field is empty. A suggestion for repository names is provided: 'glowing-octo-disco?'. The 'Description' field is optional and currently empty. At the bottom, the 'Public' option is selected, indicating that anyone on the internet can see the repository.

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Repository template

Start your repository with a template repository's contents.

No template ▾

Owner * Repository name *

hagertynw ▾ /

Great repository names are short and memorable. Need inspiration? How about [glowing-octo-disco?](#)

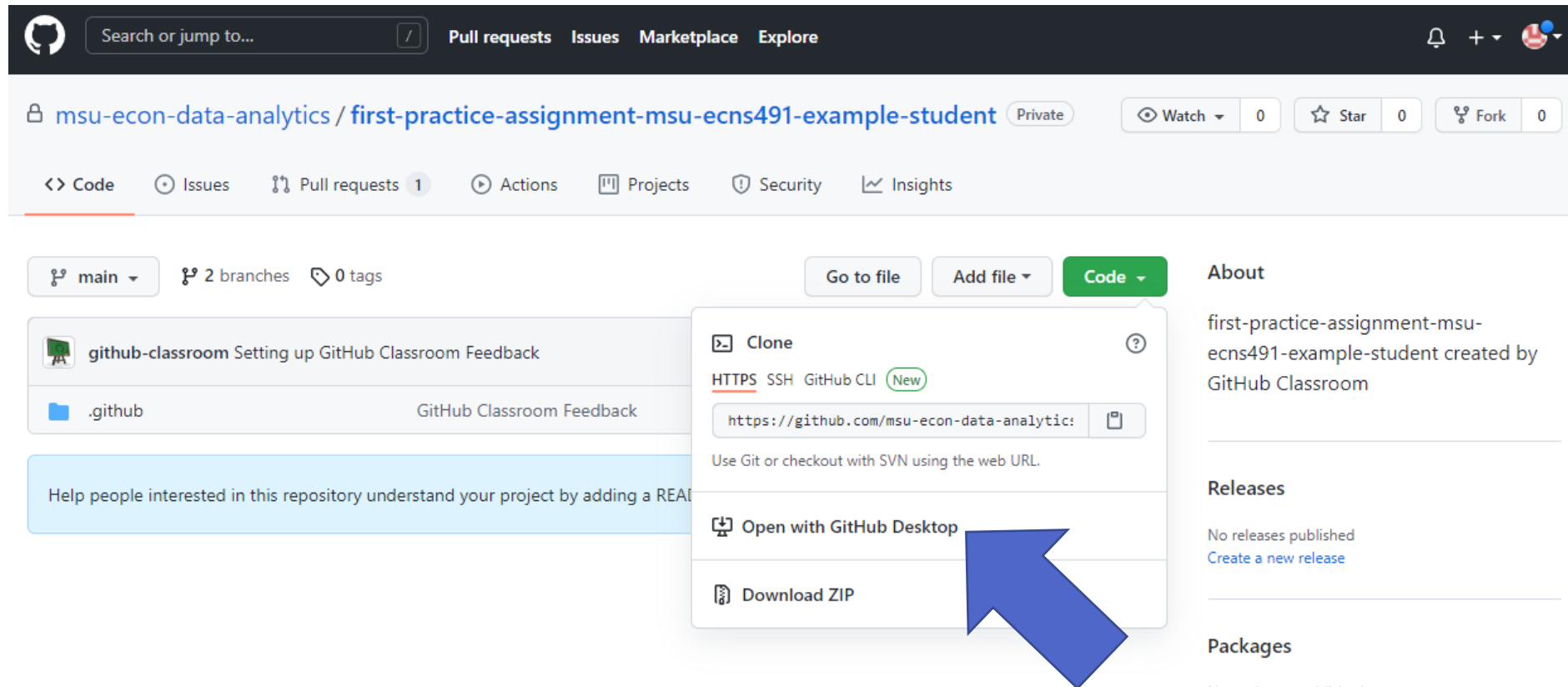
Description (optional)

☒ Public
Anyone on the internet can see this repository. You choose who can commit.

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

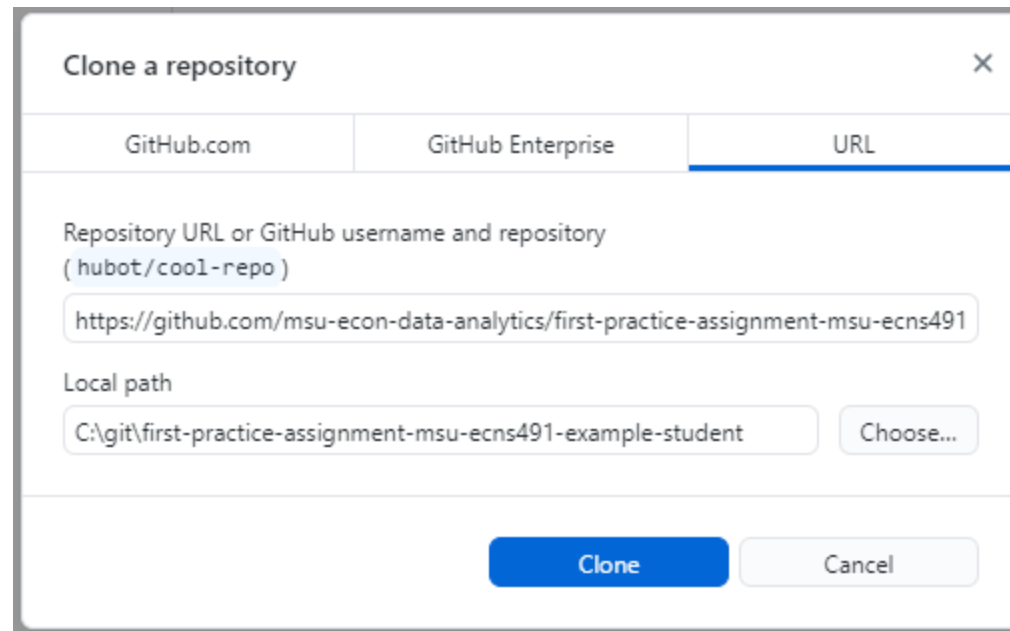
2. Clone the repo to your local machine

- Clone downloads a full copy of the repo from GitHub to file storage on your computer



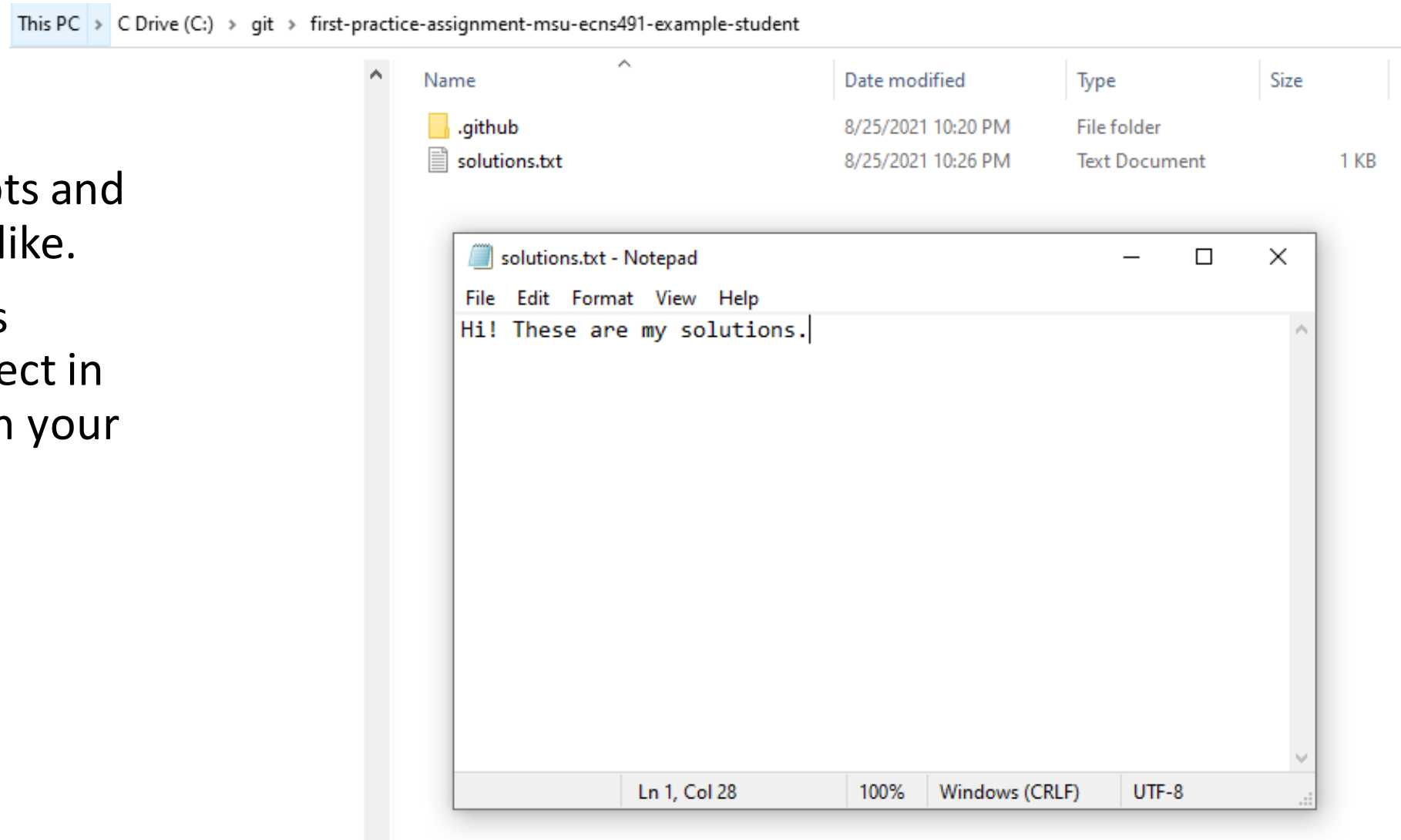
2. Clone the repo to your local machine

- GitHub Desktop should now come up
- Choose where you want to store the repo on your computer (the default location is probably fine)



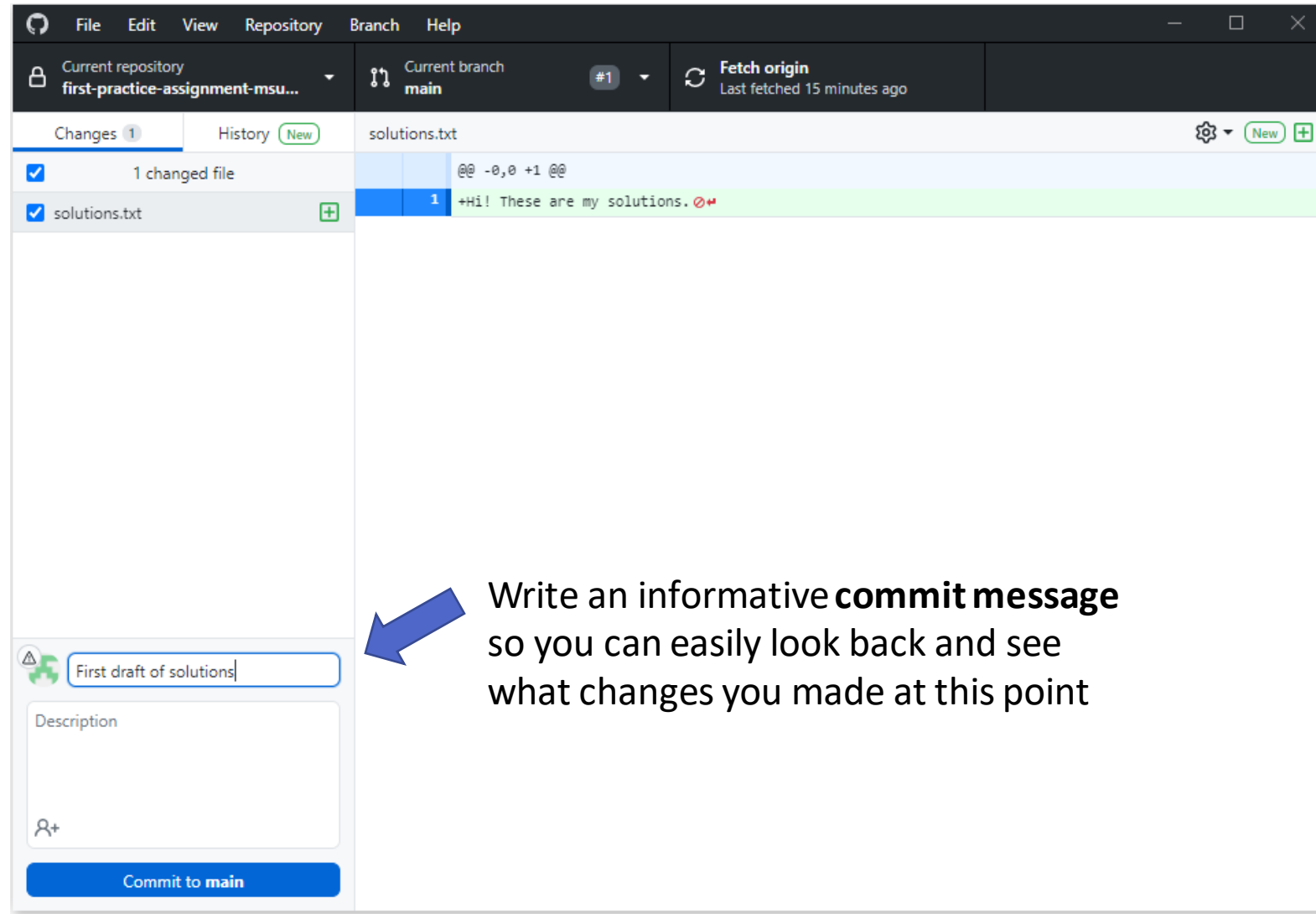
3. Do the assignment (edit the repo)

- Create or edit scripts and documents as you like.
- Save all documents related to this project in the repo's folder on your computer.



4. Commit your changes

- Commit is like Save, but for your whole project
- It records a snapshot of your whole directory at this point
- Unlike Save (but like version history in Google Docs), you can go back to a particular commit later

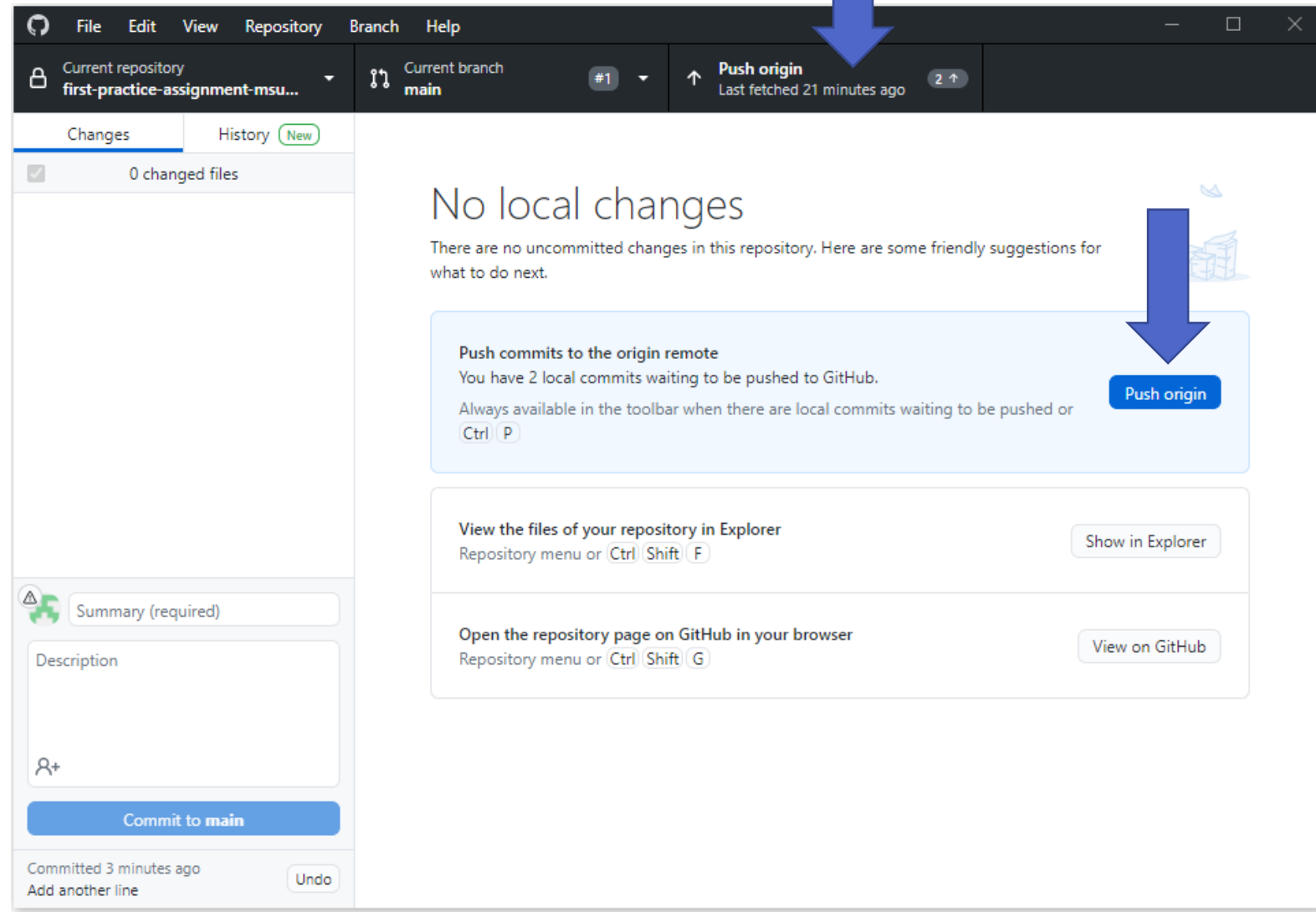


4. Commit your changes

- Commit early and often!
 - Every time you make a major change, or take a break from working
 - If you make a big mistake, you can use GitHub Desktop to roll back to an earlier commit

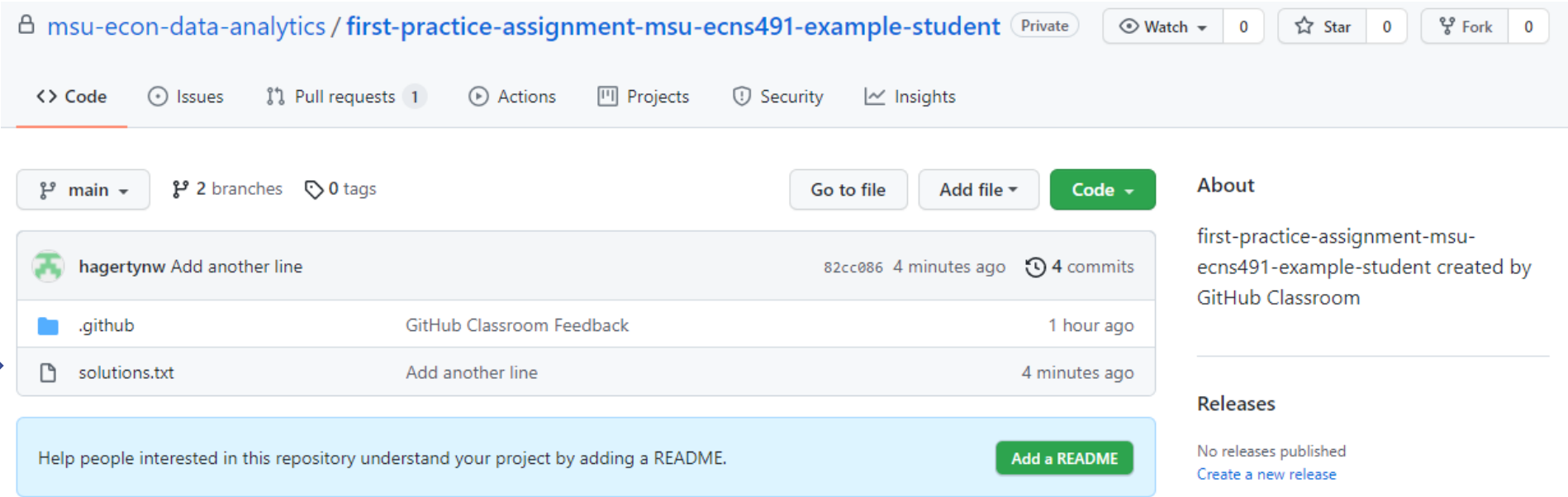
5. Push your commit to GitHub

- Commit is only local (your changes aren't on GitHub yet)
- Now we need to **push** the commit(s) to the remote GitHub repository
- Push uploads your changes to the cloud (GitHub)



5. Push your commit to GitHub

- Now, back on GitHub, you can see the new files you added



The screenshot shows the GitHub interface for a repository named 'first-practice-assignment-msu-ecns491-example-student' under the user 'msu-econ-data-analytics'. The repository is private and has 0 watches, 0 stars, and 0 forks. The 'Code' tab is selected, showing a commit by 'hagertynw' with the message 'Add another line' (commit hash 82cc086, 4 minutes ago, 4 commits total). The file list shows two files: '.github' (GitHub Classroom Feedback, 1 hour ago) and 'solutions.txt' (Add another line, 4 minutes ago). A blue arrow points to the 'solutions.txt' file. Below the file list is a prompt to 'Add a README'. The right sidebar shows 'About' (first-practice-assignment-msu-ecns491-example-student created by GitHub Classroom), 'Releases' (No releases published), and 'Packages' (No packages published).

msu-econ-data-analytics / first-practice-assignment-msu-ecns491-example-student Private Watch 0 Star 0 Fork 0

<> Code Issues Pull requests 1 Actions Projects Security Insights

main 2 branches 0 tags Go to file Add file Code

hagertynw Add another line 82cc086 4 minutes ago 4 commits

.github GitHub Classroom Feedback 1 hour ago

solutions.txt Add another line 4 minutes ago

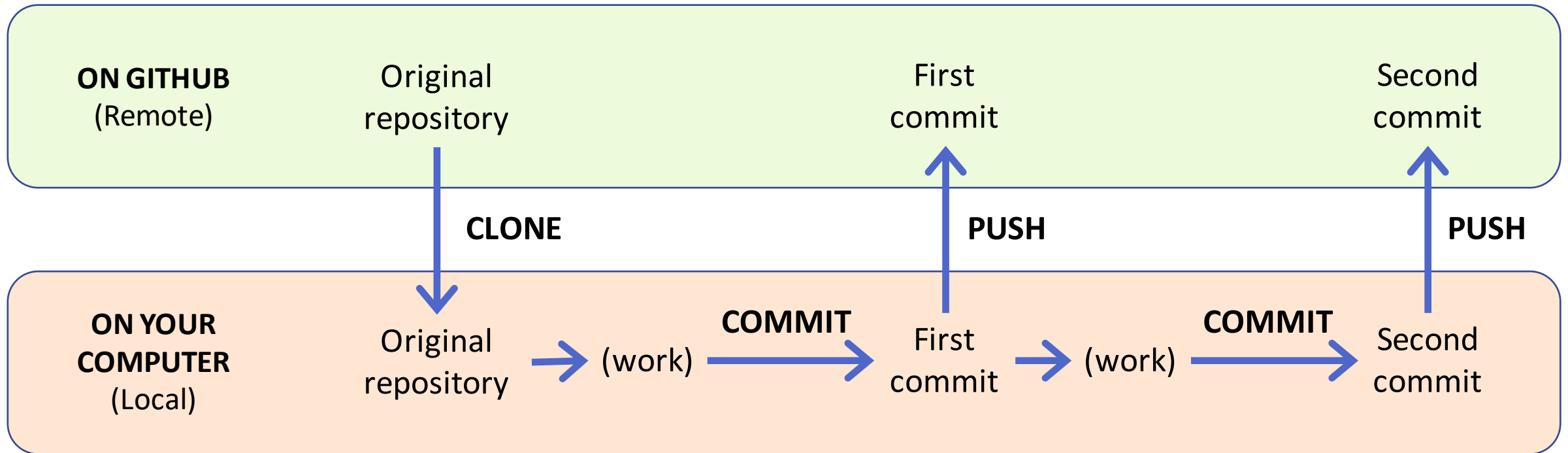
Help people interested in this repository understand your project by adding a README. Add a README

About first-practice-assignment-msu-ecns491-example-student created by GitHub Classroom

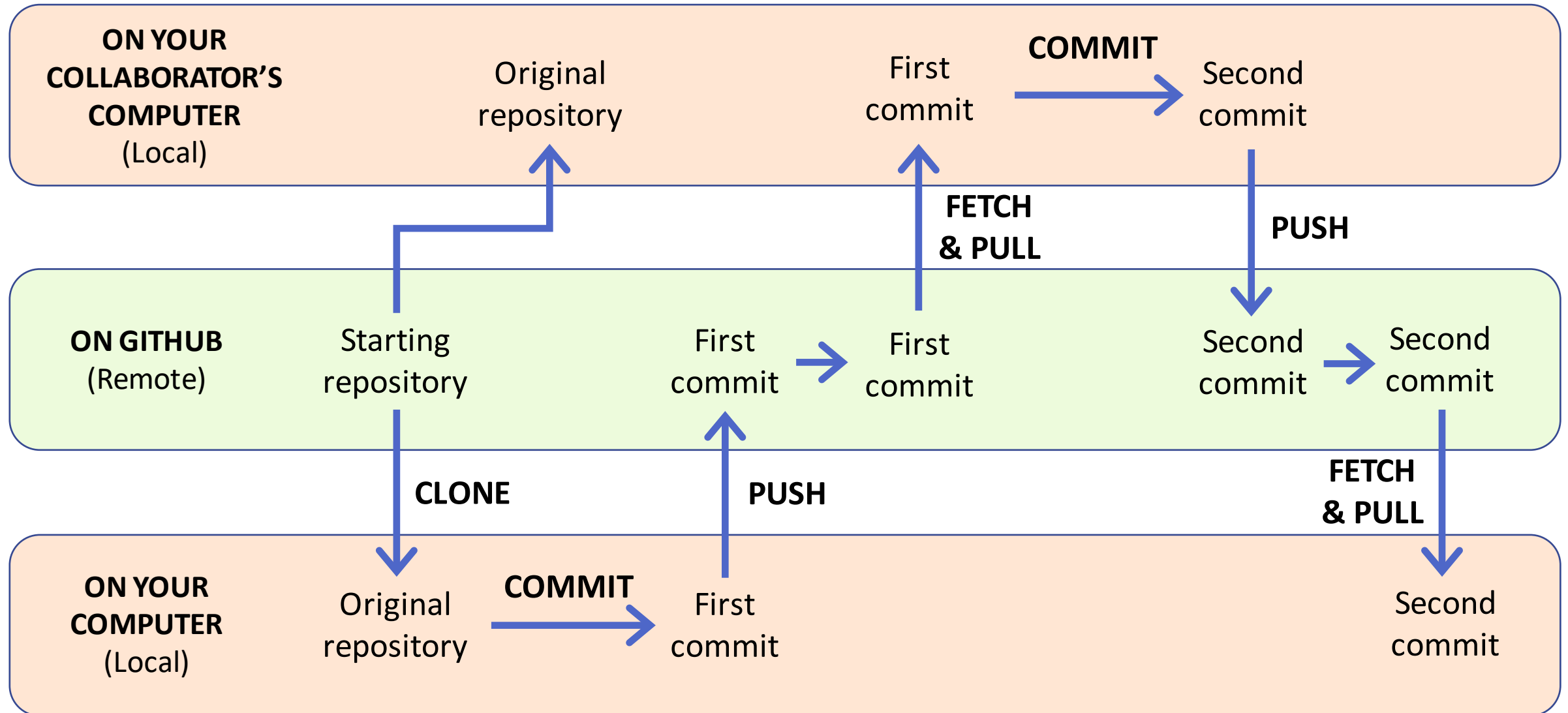
Releases No releases published Create a new release

Packages No packages published Publish your first package

Basic workflow (only 1 contributor)

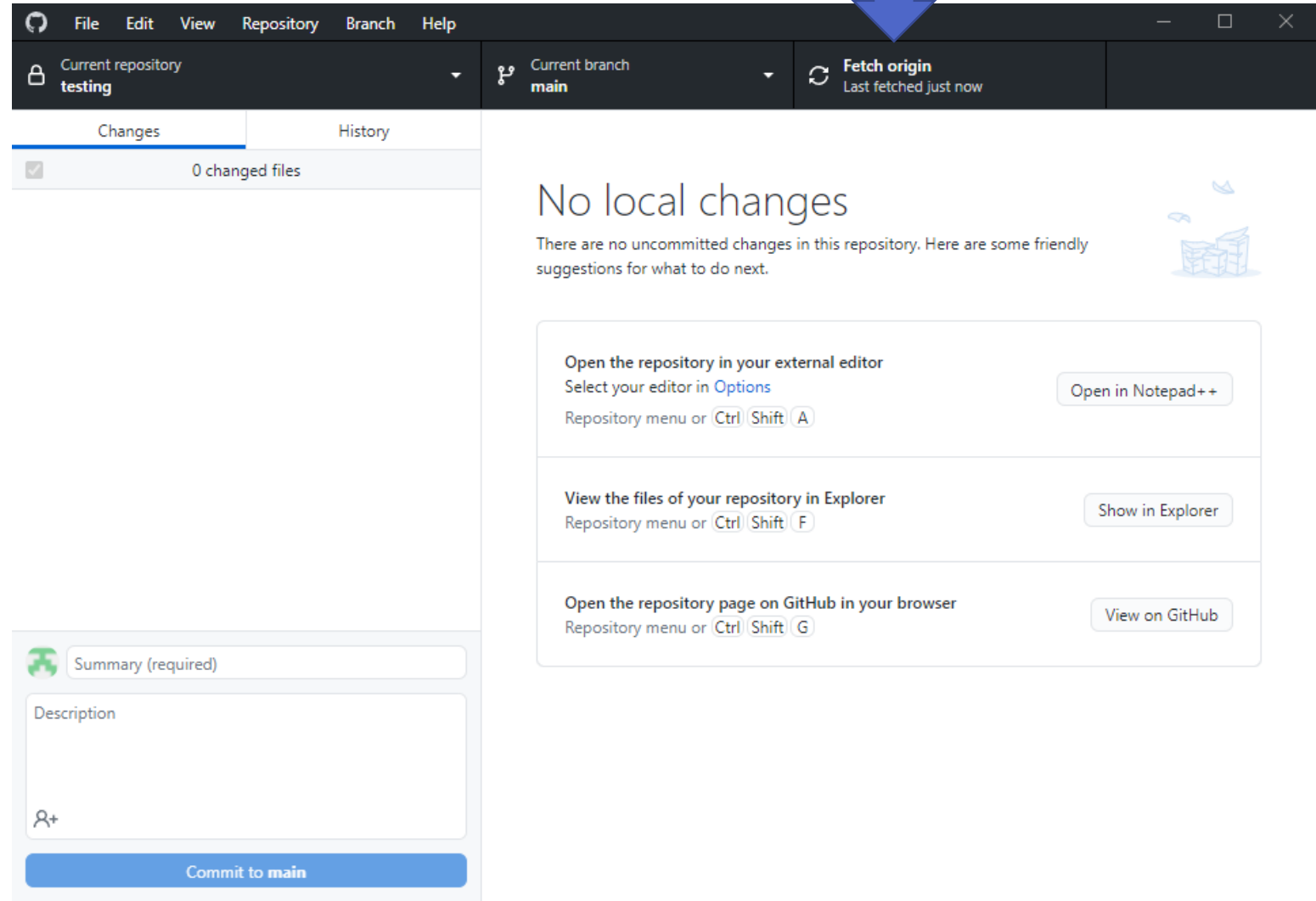


Example collaborative workflow



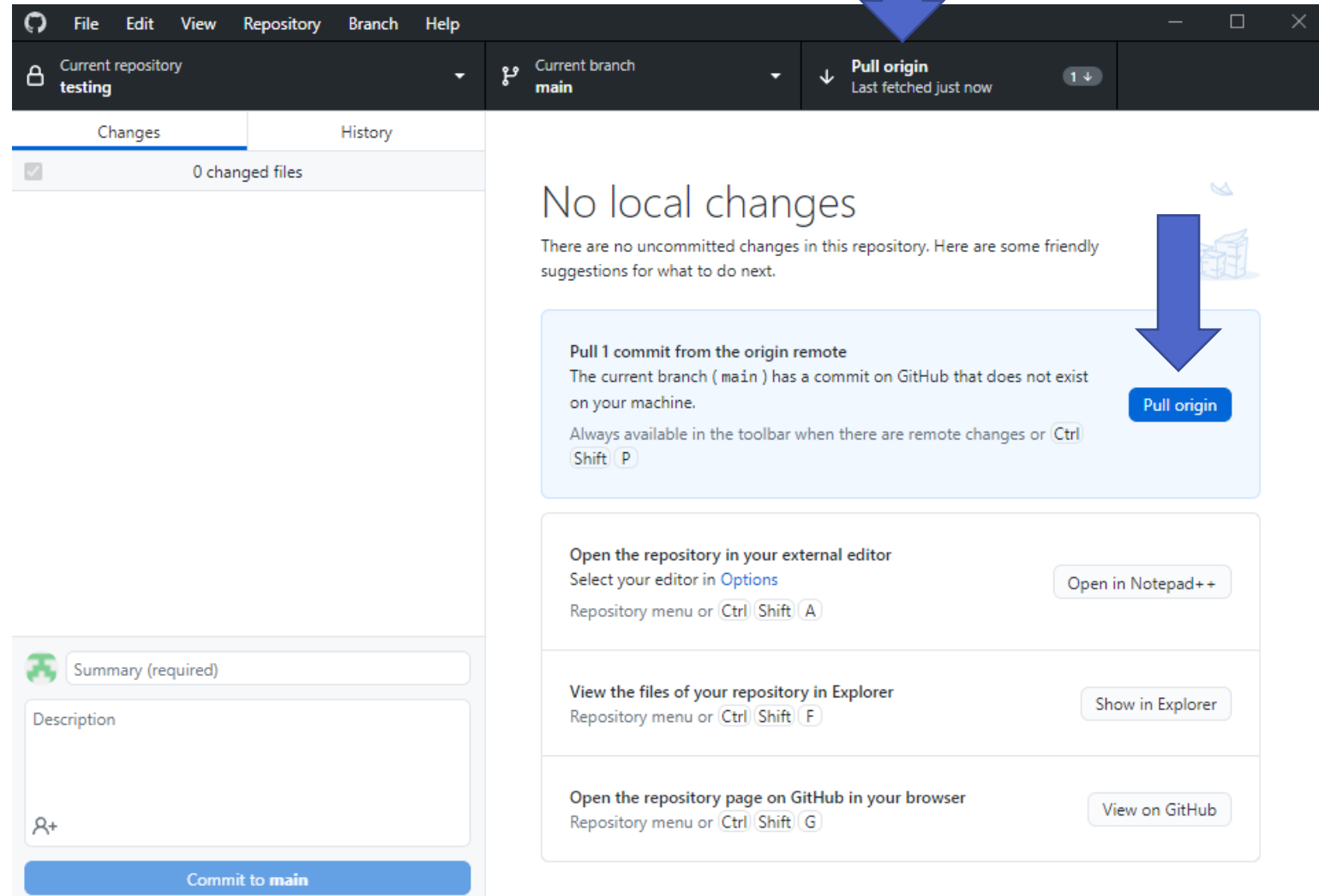
Always **Fetch** and **Pull** before you **Push**

- Your collaborator might have made changes since you last worked on it
- **Fetch** to check for changes



Always **Fetch** and **Pull** before you **Push**

- Your collaborator might have made changes since you last worked on it
- **Fetch** to check for changes
- **Push** to merge their changes with yours
- Resolve any merge conflicts
- Now you can **push**!



Many more features & workflow options

(All optional, but very useful for collaborating)

- Forking and pull requests: <https://guides.github.com/activities/forking/>
- Branches and merges: <https://guides.github.com/activities/hello-world/>
- For much more, see the other “Git and GitHub” resources on the course resource list: <https://github.com/msu-econ-data-analytics/course-materials#git-and-github>