Creating new projects

Starting from scratch

EPI 590R final project

Your goal will be to create an analysis that

- I can reproduce on my own computer
- Is easy to rerun if I tell you, for example, to remove the 12th row of your dataset

We'll start this in class!

New projects

We cloned our first project from GitHub; now we are going to start a new project from scratch

1. File > New Project > New Directory > New Project

- If you ever want to convert an existing folder that holds an analysis into an R project, you can choose “Existing Directory”
- You’ll also see other options besides “New Project” – an R package, a Shiny app, etc.
  - These will get you set up with some initial files for these types of projects
  - You can also make a template of your own!

New projects

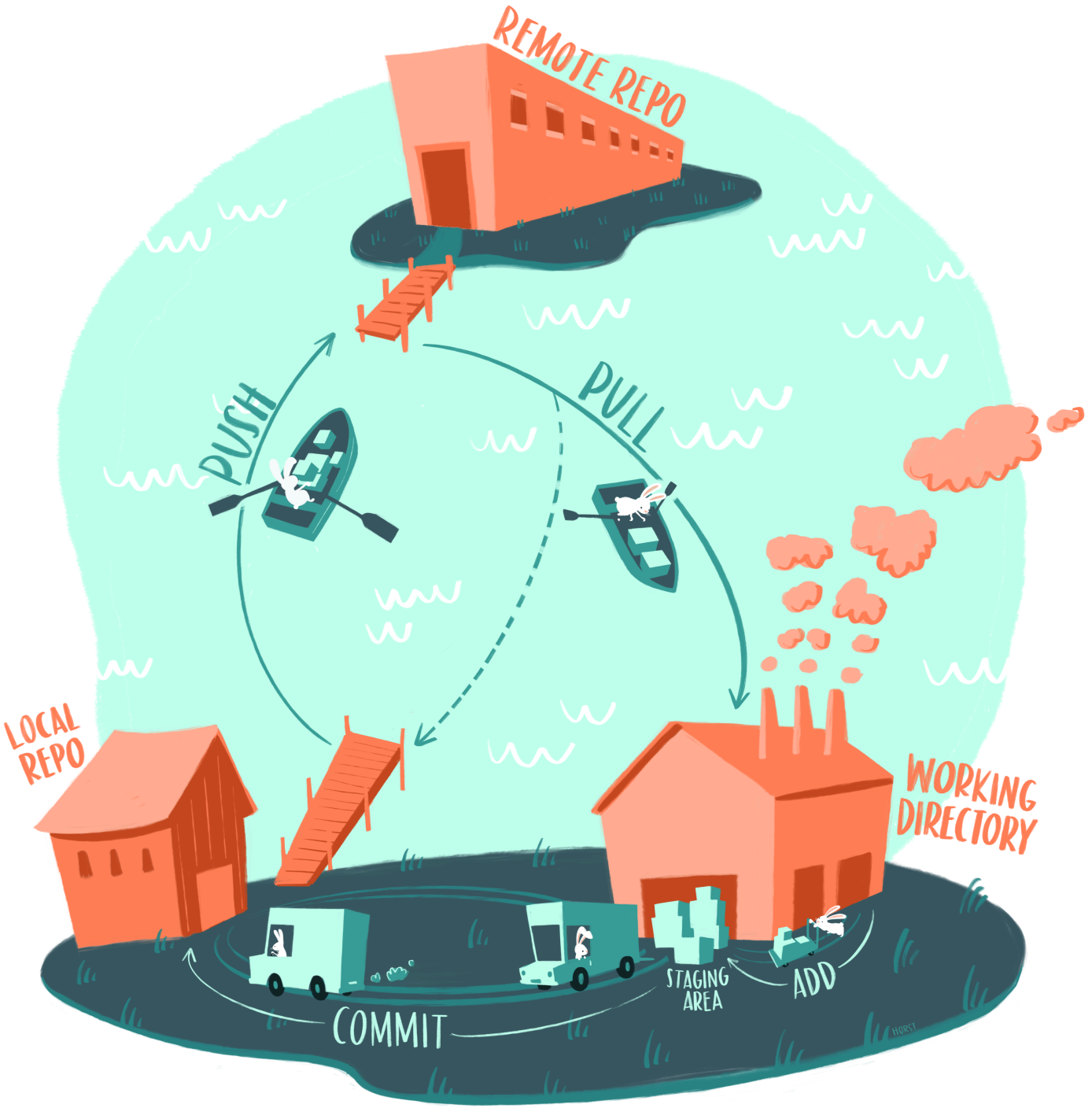
2. Choose a name for your new project and where to store it on your computer

- Check “Create a git repository”
  - This gets you all set to connect to GitHub and creates a `.gitignore` file
- You can leave “Use renv with this project” unchecked (we’ll be introducing the `{renv}` package later!)

Initial Git commit

3. Stage and commit the files

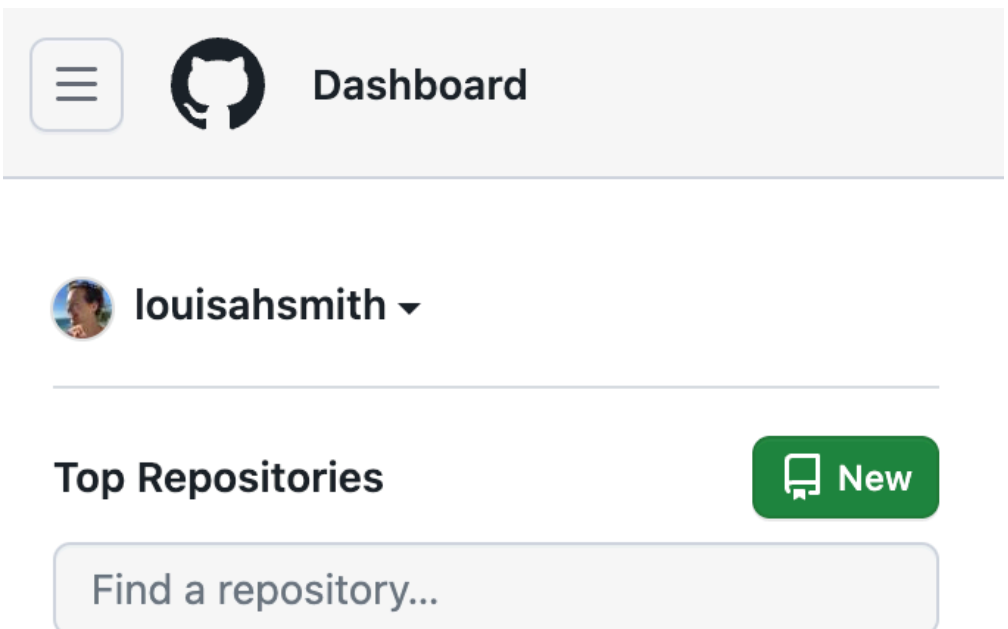
- I usually use “initial commit” as my first commit message since I haven’t don’t anything yet!
- We can’t yet push because we haven’t connected to a remote repository


REMOTE REPO
PUSH
PULL
LOCAL REPO
WORKING DIRECTORY
STAGING AREA
ADD
COMMIT

Creating a new repo on GitHub

4. Open up your web browser to GitHub and make a new repository

From github.com | From github.com/username



Repository options

5. Choose a name (preferably one that matches the name you gave your R project).

- You can choose to make it private, if you wish
  - Private repos have fewer features *unless* you have GitHub Pro (which you can get for free as a student with the GitHub student developer pack!)
- You don't need to click anything else

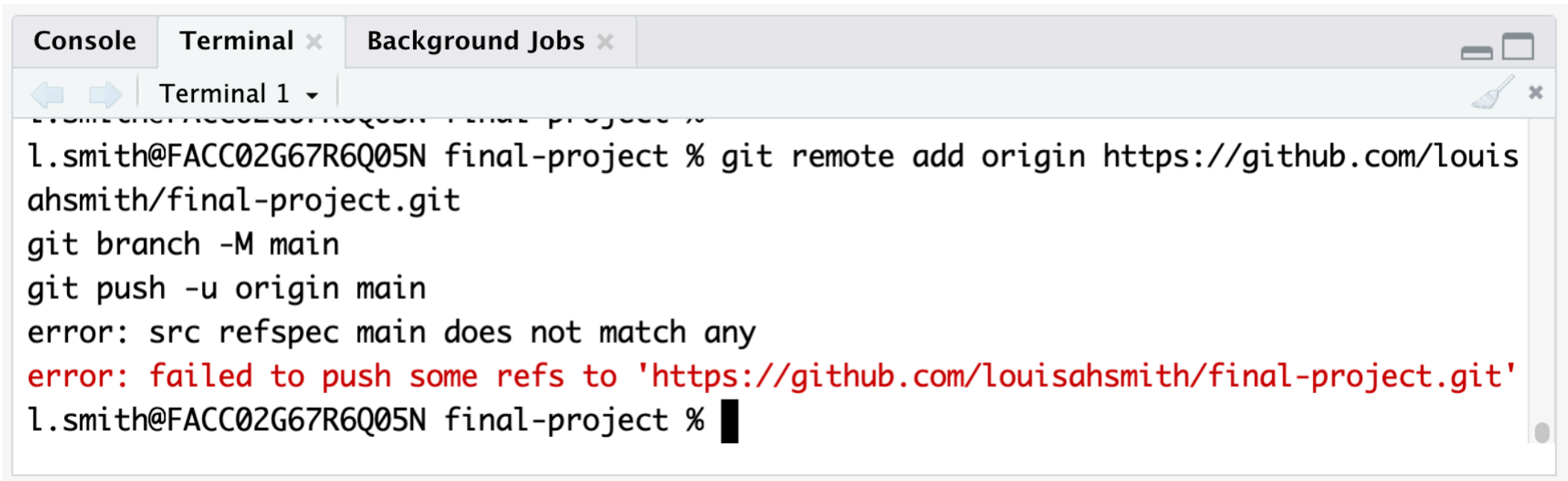
Connect the local to the remote

- You created your local repo with RStudio in a directory you chose
- Now you need to connect it to the remote repo on GitHub

6. Copy the code from the second section: “push an existing repository from the command line” in the *terminal* within RStudio.

Connect the local to the remote

7. Run the three lines of code *one at a time*, then refresh your GitHub page!



`.gitignore`

You likely don't want to push everything to GitHub, even if you have a private repository

- Be especially careful about data and passwords
- You also can't push very large files (>100 mb)

A `.gitignore` is a special text file that tells Git not to track certain files

- RStudio starts you off with a few entries, including `.Rhistory` since no one needs to see everything you've run in R!

`.gitignore` exercises

8. Create a new file called `secrets.txt` within this new repo

- Write down your deepest, darkest secrets and save (e.g., I am scared of spiders)

9. Open `.gitignore` via the RStudio filepane

- Add “secrets.txt” below the files that RStudio helpfully ignored for you
- Save

Keep your eye on the Git pane!

Starting the final project

10. Set up your folders how you'd like in your repo (you can always change this)

- Find some data, download it, and store it in your repo
- Commit and push to GitHub!

For your final project, your data must be something that can be stored online and accessed by me.

Some fun options for data are:

- https://data.fivethirtyeight.com/
- https://github.com/rfordatascience/tidytuesday#datasets
- https://github.com/higgi13425/medicaldata/tree/master/data (descriptions: https://higgi13425.github.io/medicaldata/#list-of-datasets)

Exercises

Get started making a new project and GitHub repo for your final project, editing the `.gitignore` file, and finding some fun data

You can always change anything you want later, and even delete the whole thing and start fresh!

15:00