# Making Decisions Lab

## Instructions

To start, download the exercise content for today.

Once you have done that, use the command line to navigate into the folder and follow the steps from last class to initialize a git repository. As you move through todays lab (and especially once you have completed it), make sure to commit your changes.

Open the lab.js file, and attempt as many problems as you can.

The goal is to get through all of the starting questions, and then as many of the intermediate and advanced questions as possible. If you get through the intermediate questions, challenge yourself with the advanced questions. The advanced questions will cover topics not taught in class. This is intentional. Use resources online to help you figure out the solutions.

**Do not move on to the next section until you have completed the starting and intermediate questions.**

## Backing Up Code to GitHub

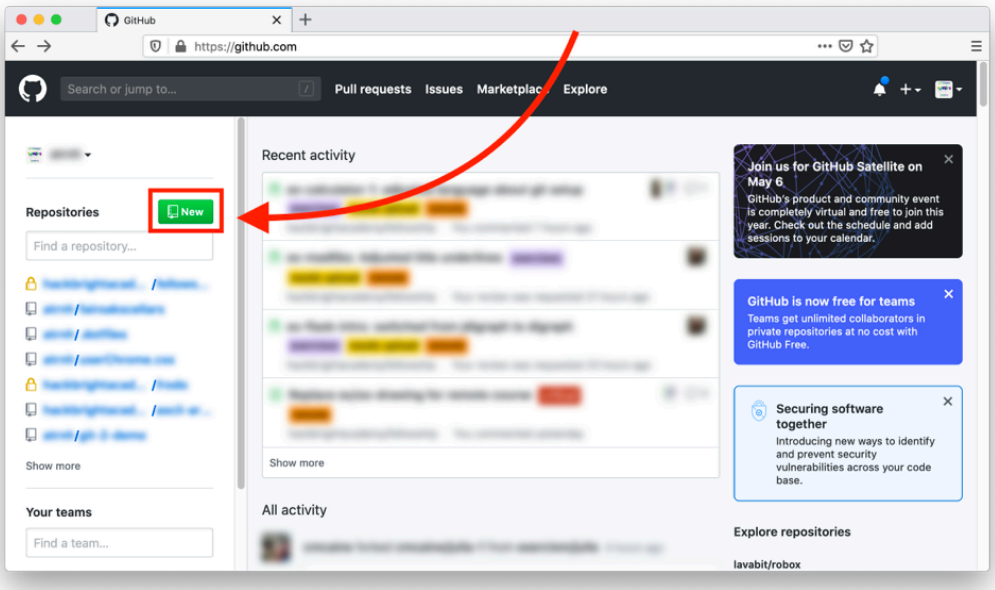
Git is great and all, but your files still just exist on your computer which is nice… as long as your computer works. Enter GitHub!

GitHub is a company that gives developers a place to upload their code, showcase projects in a portfolio, and enable collaboration between developers by taking advantage of Git to sync code between different machines. You don't need to have a GitHub account to use Git but a GitHub account *will* allow you to make your code and yourself more visible in the software engineering community.

**Warning: Do you have a GitHub account?**

If you have not yet created a GitHub account, click here to create one now.

Every time you start a new software project, you should upload it to GitHub. Let's start by uploading your code from this exercise. To get started, log into GitHub and click the green ***New*** button to create a new GitHub repository (see the screenshot below). You'll want to create a new GitHub repository for every Git repository you want to upload.

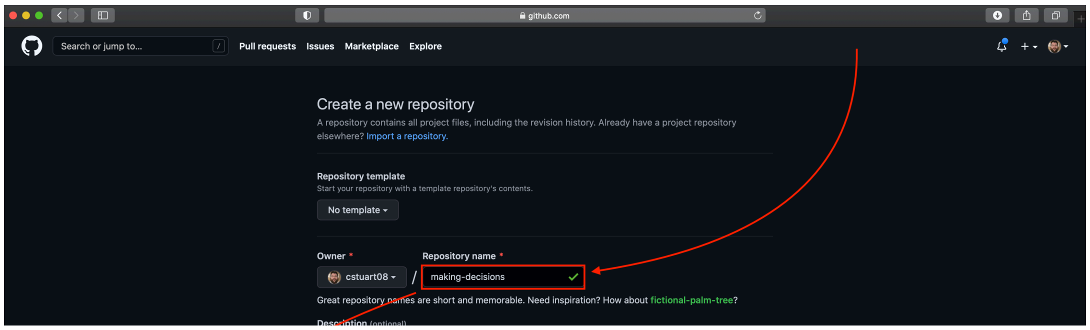


Join us for GitHub Satellite on May 6
GitHub's product and community event is completely virtual and free to join this year. Check out the schedule and add sessions to your calendar.

GitHub is now free for teams
Teams get unlimited collaborators in private repositories at no cost with GitHub Free.

Securing software together
Introducing new ways to identify and prevent security vulnerabilities across your code base.

This will take you to a form to create a GitHub repository:



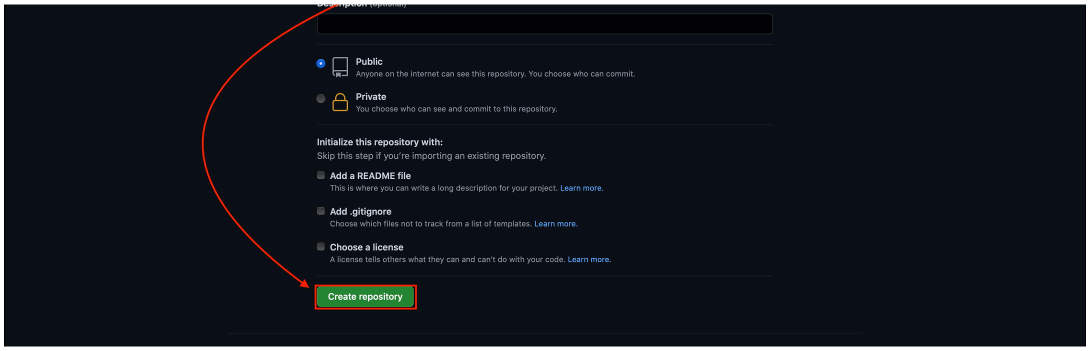
Create a new repository
A repository contains all project files, including the revision history. Already have a project repository elsewhere? Import a repository.

Repository template
Start your repository with a template repository's contents.
No template

Owner * cstuart08 / Repository name * making-decisions

Great repository names are short and memorable. Need inspiration? How about fictional-palm-tree?

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.

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

☐ Add a README file
This is where you can write a long description for your project. Learn more.

☐ Add .gitignore
Choose which files not to track from a list of templates. Learn more.

☐ Choose a license
A license tells others what they can and can't do with your code. Learn more.

Here's how to fill out the form:

- Under ***Repository Name***, enter `making-decisions` (or whatever you'd like call it, but note this affects the URL)
- The description is optional
- Leave it set to ***Public*** — we want your future employers to be able to check out your code
- Leave ***Initialize this repository with a README*** unchecked — you've already initialized the repository on your local machine with ***git init***

Click ***Create Repository*** to create a repository with the above settings. You've just created a ***remote repository*** ("remote repo" for short or just "remote"). It's a remote repository (think like a *remote island*) because it exists far away from your computer. The repository on your computer is a ***local repository*** (or "local repo").

**Note: Industry terms**

"Remote repository" is a mouthful, so most of the time people use the term "remote repo" or, even more informally, "remote" instead. For example:

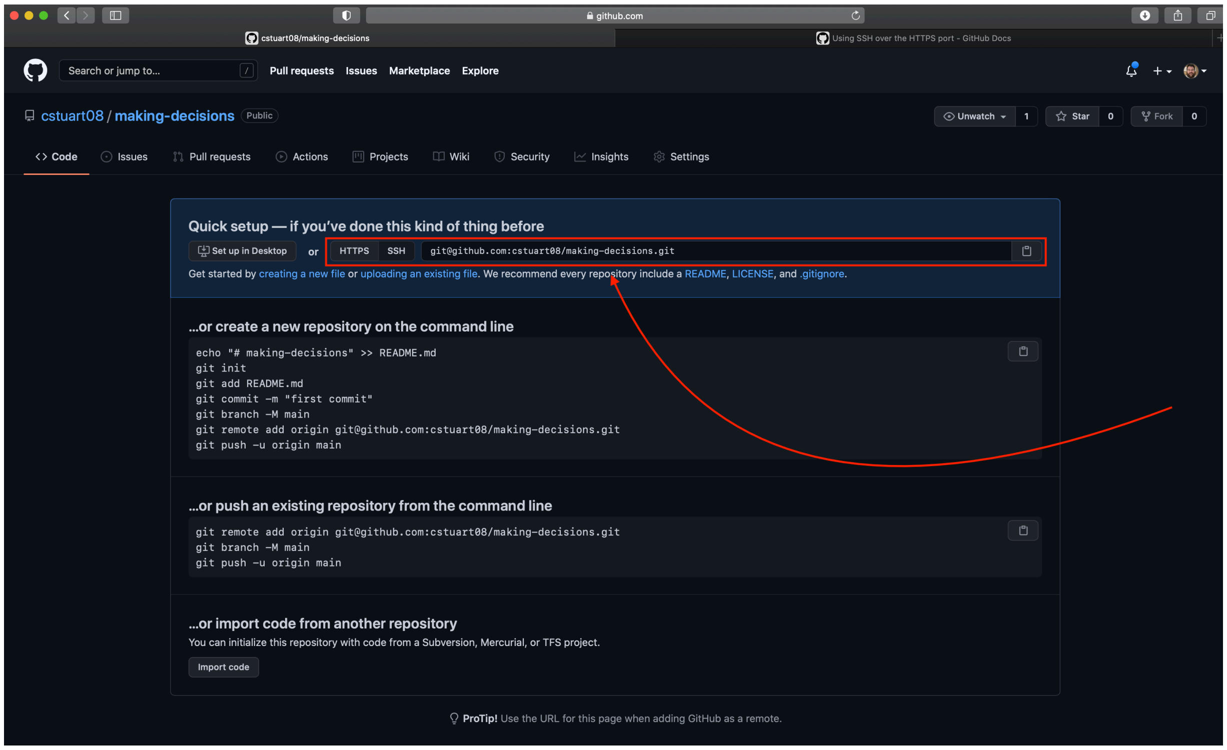
- Did you check what's in the remote?
  - Translation: Did you check to see if the code you're looking for is in GitHub/the remote repository?
- I can't fetch from the remote repo
  - Translation: I can't get the latest code from the GitHub repository.

Similarly, "local repository" is shortened to "local repo" and sometimes it's shortened to "local" but only in certain contexts. For example:

- The updates are only local right now

    - Translation: My latest code exists locally, on my machine (implying that I haven't uploaded it to the remote repo yet)
- I have a local repo with the files you're looking for
    - Translation: I have a copy of the repository on my computer that contains some files that probably went missing, somehow
    - In this case, you wouldn't say, "I have a local with the files you're looking for" — that just sounds weird!

Now you'll need to connect your remote repository to your local repository. To do this, you'll need the address of the remote repo. From your remote repository's homepage, locate the ***Quick Setup*** box and copy the HTTPS URL of your remote repo:



Quick setup — if you've done this kind of thing before

```
git@github.com:cstuart08/making-decisions.git
```

Get started by creating a new file or uploading an existing file. We recommend every repository include a README, LICENSE, and .gitignore.

…or create a new repository on the command line

```
echo "# making-decisions" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin git@github.com:cstuart08/making-decisions.git
git push -u origin main
```

…or push an existing repository from the command line

```
git remote add origin git@github.com:cstuart08/making-decisions.git
git branch -M main
git push -u origin main
```

…or import code from another repository

You can initialize this repository with code from a Subversion, Mercurial, or TFS project.

**ProTip!** Use the URL for this page when adding GitHub as a remote.

With your remote's address in hand, head over to your terminal — next, you'll tell your local repo that it should track your GitHub repo with the Git command, ***git remote add***.

**Warning: Make sure you're in the right place**

Make sure you're in the directory of your Git repo before you execute any of the Git commands below.

***git remote add*** takes in two arguments:

- The name you want to use to refer to your remote
    - The name can be anything you want but `origin` is the name everyone uses by default
    - This allows you to connect your local repo with *multiple* remote repos
- The remote's address

Putting all of the above together, here's the command to connect your GitHub repo and local repo together:

```
$ git remote add origin REPLACE_THIS_WITH_YOUR_URL
```

When we add a remote, much like when we do any other configuration, it only exists for that repository. Outside of the folder our repository is in, none of these settings exist.

Now we have a remote — this is like a portal that goes to whatever URL we point at. So, now that we've established a portal link, what do we do with it? Same thing you'd probably do if you actually opened a portal to the server room at GitHub. Push things through it and see what happens.

```
$ git push REPLACE_WITH_REMOTE_NAME main
```

Replace ***[remote name]*** (no square brackets) with the name you used when you added the remote (it's probably `origin`). You might wonder about what `main` is — that's the name of the branch. For now, we'll just have one branch, so we'll always type `main` there. In the ***Further Study*** section, there's some information about branching.

Once you've ***git push***-ed, Git will ask you for a username. This is your GitHub username. Then it will ask for your password. (When you type a password, you may be used to seeing asterisks (*******)

appear; in this case, nothing will. Don't worry! It's still working. This is just another way of hiding your password.)

After this, head over to your GitHub profile. You should see that your new repo has been created, and that any files you've added have been pushed to the server. Pushing code to GitHub is the simplest way to back up your work to the cloud, share code between teams, and make sure you remember what you did and when you did it (complete with notes to yourself!).

## Congratulations on pushing your first exercise!