Lesson 1 - Setting up our IDE (VS Code) and Source Repository (git)

Note: This semester, I’ll be creating mini lessons in Word documents similar to this, so that they are accessible to all (as opposed to videos). Please do not hesitate to contact me if you need additional help or if you get stuck on a step provided in these lessons. Note that everything we do in this class is very standard, so there are literally thousands of videos and tutorials that can help you along the way. I have also provided a discussion board where you can share resources you find helpful with each other.

Slow and accurate is the goal for this step-by-step tutorial!

Contents

[Step 1 – Install Visual Studio Code 1](#_Toc503196013)

[Step 2 – Setup your Git Account 1](#_Toc503196014)

[Step 3 – Verify your account 3](#_Toc503196015)

[Step 4 – Sign Up for your GitHub Student Developer Pack 3](#_Toc503196016)

[Step 5 – Install git to your Computer 4](#_Toc503196017)

[Step 6 – Setup you Repository in GitHub 4](#_Toc503196018)

[Create the Repository 4](#_Toc503196019)

[Step 7 – Connect VS Code to GitHub 5](#_Toc503196020)

[Step 8 – Push Files to GitHub 6](#_Toc503196021)

[Step 9 – Push files to github.com 7](#_Toc503196022)

# Step 1 – Install Visual Studio Code

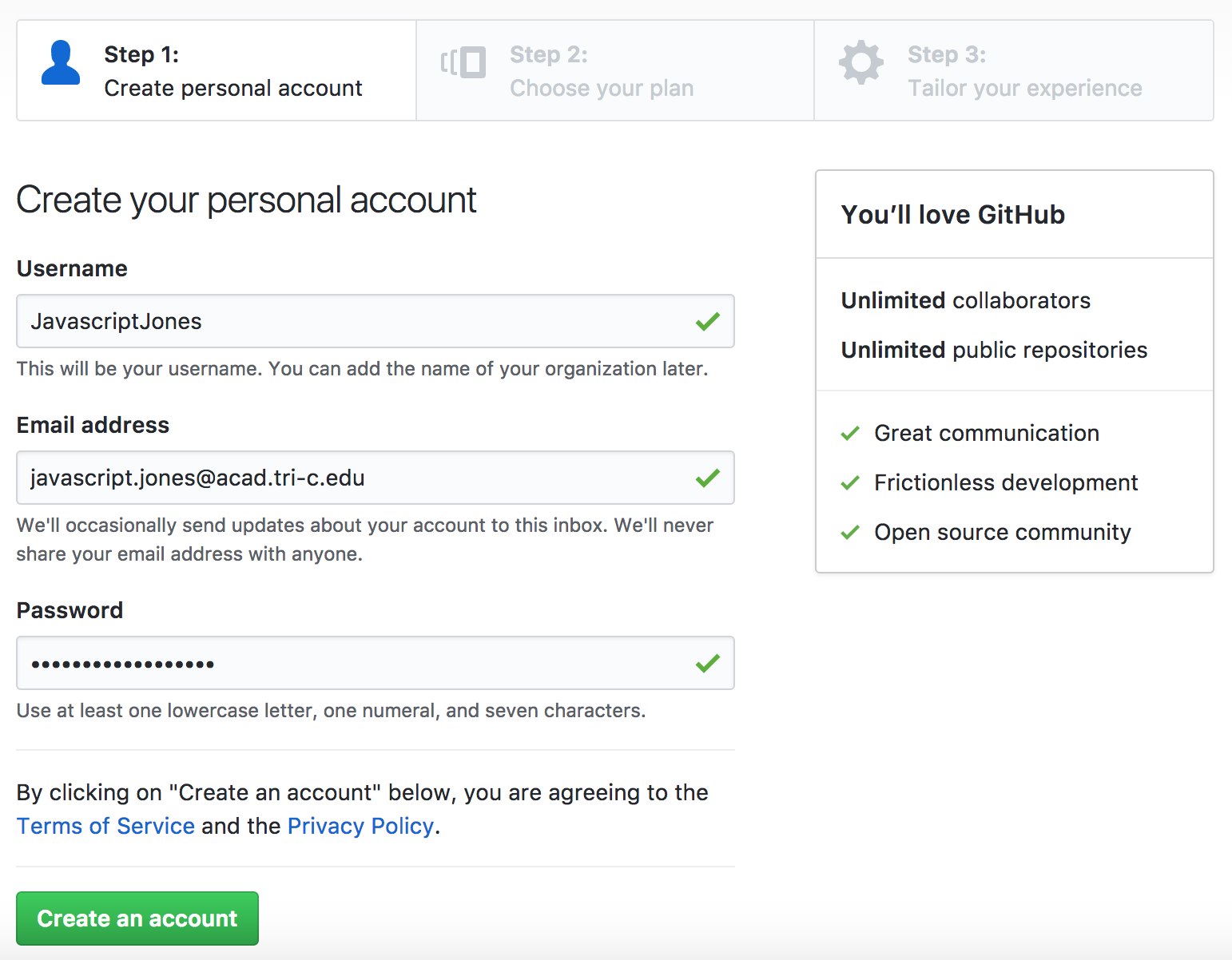
Visual Studio Code is the IDE that we’ll use to develop our web pages including JavaScript. It is a lightweight IDE with built-in support for web application development. In addition, we will be able to hook it up to our version control tool git.

Download and Install Visual Studio Code: <https://code.visualstudio.com/download>.

# Step 2 – Setup your Git Account

Note: skip this part if you already have an account for your @acad.tri-c.edu email address

Visit the github.com sign-up page - <https://github.com/join?source=header-home>. Fill in the following information for your name and email address.



* Use your full name in upper camel case, so that your instructors can easily recognize that the account is yours. Example: instructor James Leasure would use the username JamesLeasure. You might need to add a short number at the end if your username is already taken. Example: JamesLeasure9
* Use your Tri-C email address for the Email Address.
* The password is not connected to your Tri-C account. You’ll need to remember your password!

Click “Create an account”.

On the next screen, select unlimited repositories for free and click Continue. Then for “Tailor your experience” fill-in the information as you see fit and click “Submit”.

Congratulations, you’re now signed in to your git account!

# Step 3 – Verify your account

Log into your Tri-C email account and find an email sent to you from github. Using the email sent from GitHub, verify your account.

Note: This is an important requirement for the next step, so don’t skip it!

# Step 4 – Sign Up for your GitHub Student Developer Pack

Note: skip this part if you already have a student developer account for your @acad.tri-c.edu GitHub account

While still logged into github.com (don’t worry if you logged out, just go to github.com and click Sign-In), enter the following url into the address bar of your browser - <https://education.github.com/> and click Enter.



Note: you have to be logged in before going to this URL!

Click “Get the pack”. The next screen will list all of the great tools you’ll have access to with the developer pack. The reason that we’re requesting the student developer pack is so that you can make your repository private. That’s a requirement of the class. Without the student developer pack option, this costs $7 per month, so we’re definitely going to take advantage of the free option through the student developer pack!

Now, github wants to verify that you’re a student. Click “Yes, I’m a student”.

Now, there are two screens that might appear. If you have not verified your account, it will ask you to upload a picture of your student id. Make sure you’ve verified you’re account and you should be able to easily step through this without anything complicated.

Note: Enter “Cuyahoga Community College” for you School name.

When you have entered all of the information, click “Submit Request”.

# Step 5 – Install git to your Computer

Download and install the git command line tool: <https://git-scm.com/download>

You don't have to do anything with the command line tool. It just has to be installed, so that it works with Visual Studio Code.

# Step 6 – Setup you Repository in GitHub

A git repository is a remote folder that stores a copy of our source code. It can be used in several useful ways: working on source code with a team, keeping a version history, creating several branches (separate copies of the course code for different purposes), and so on. The two important reasons we’ll use a repository are to 1) share you source code with your instructor and 2) keep a version history of changes that you make to your code.

## Create the Repository

1. Log into you github account at github.com
2. Click “New repository” or click on the drop down arrow next to the plus sign + in the upper-right corner and select “New repository”.
3. Enter repository name as follows:

CourseID-FullName-Semester

For the course IT2320 in Spring 2018 for user James Leasure, the repository name would be: IT2320-JamesLeasure-Spring2018

Notice: There are no spaces, camel case is used. And the name is full name starting with first name. Your repository name must be accurate! It will be one of a hundred for your instructor and will be difficult to find if you name it incorrectly.

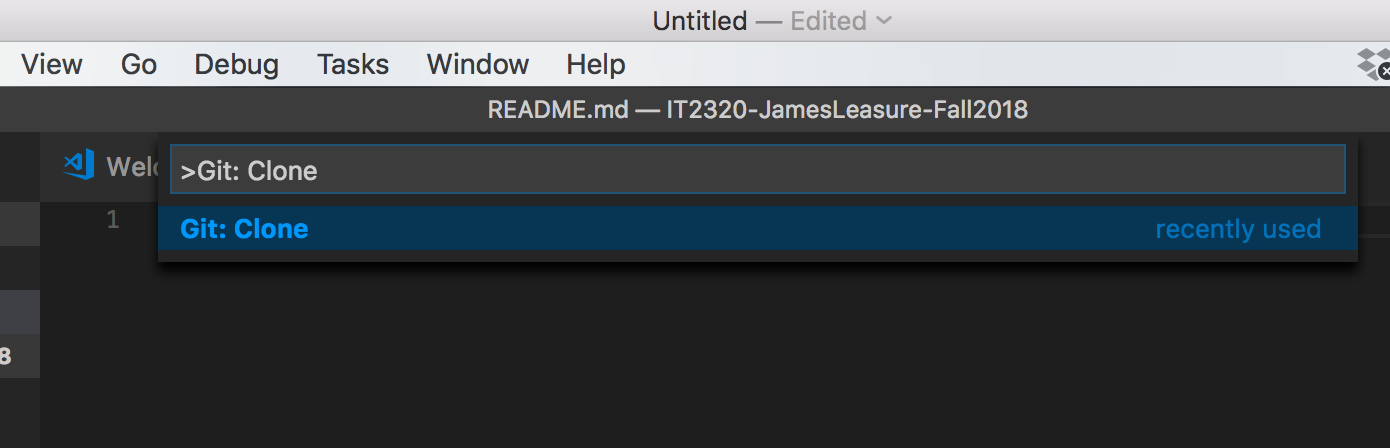
1. Select Private. This will only work if you have finished setup for your student developer account. Otherwise, just make it public for now and contact your instructor for instructions on how to make it private after the fact. Note that your instructor can tell if it’s private or public. And, it must be private.
2. Select “Initialize this repository with a README”. This will put a single README.md document into your new repository.
3. Click “Create repository”.

Congratulations, you just completed the setup for you git repository!

Note the URL in the address bar right now. This is the path to your source code repository for this class. You’ll use this URL to submit your assignments for the class in Blackboard.

# Step 7 – Connect VS Code to GitHub

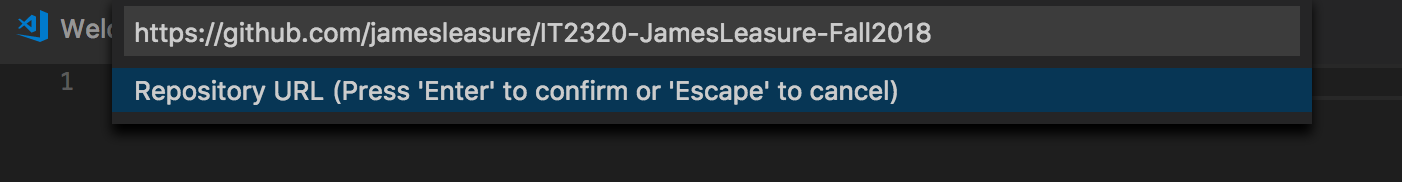
1. In Visual Studio, click View -> Command Palette. Once the Command Palette appears, type in **Git: Clone** and press the Enter key.



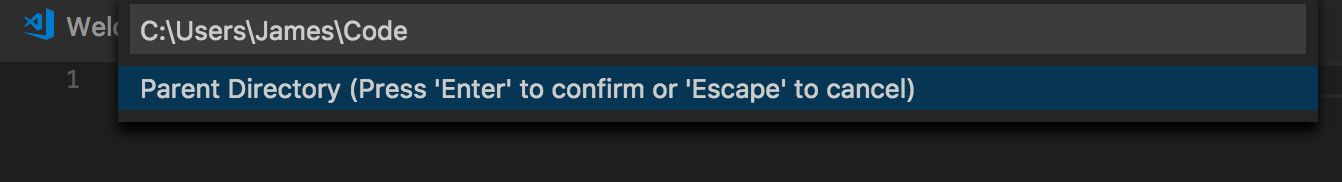
1. Now, you’ll be asked for the URL to your repository and the path to a folder on your computer where you want to store the files on your computer.

Note: this will create a folder with the same name as your repository in github in the folder that you enter here. For example: if I put it in C:\code, it would create C:\code\IT2320-JamesLeasure-Spring2018 where I will eventually put all of my labs. Something to avoid is to create a new folder with the same name as your repository. You don’t want a path like C:\code\IT2320-JamesLeasure-Spring2018\IT2320-JamesLeasure-Spring2018.

1. Next, enter the URL to your repository. This is the URL that we copied from the address bar earlier just after creating our repo in git. You can easily find this by logging into github.com and then clicking on your repository (found under Repositories on the right-side of the screen).



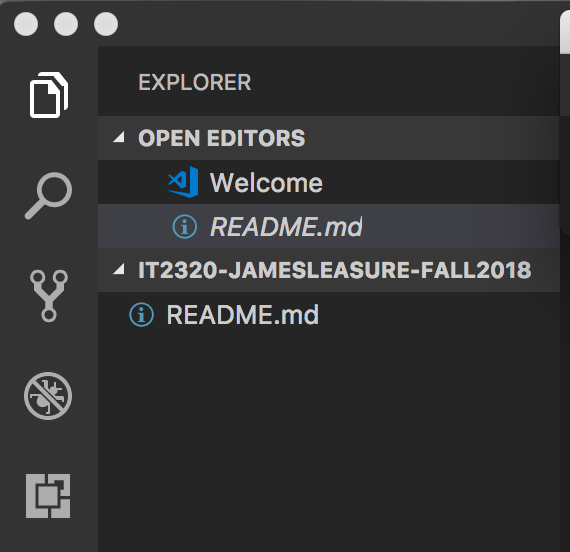
1. Next enter the path to a folder on your computer where you want to store the source code. Remember from the note above: it should be a short path. You might want to copy/paste it from the file explorer or finder.



1. That’s it! You now have the repository open. And, whenever you open that folder from Visual Studio Code, it should automatically open your local source repository.

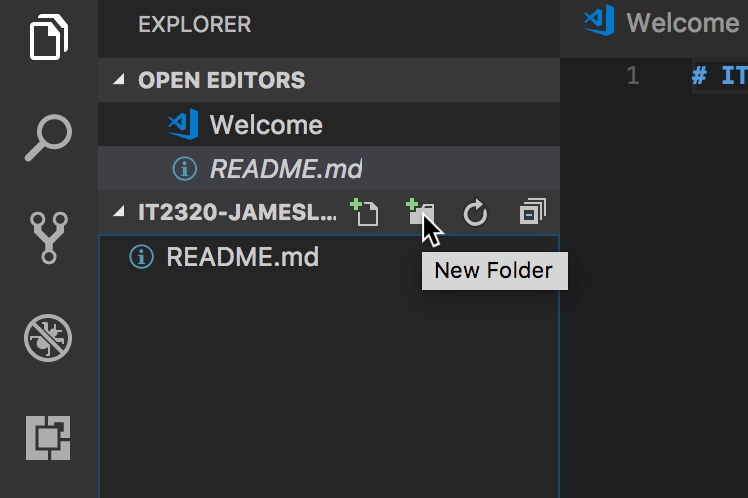
# Step 8 – Push Files to GitHub

Step and check your work! With your local source code repository open, you should see the README.md file that was created along with your remote (the one on github.com’s servers) git repository. It’s a good sign if you see the file. If not, you might want to contact your instructor asap to work through any setup issues prior to this step.



Now, we’re going to create our first Lab Assignment folder. So, again make sure your local repository folder is open! You should see that README.md file.

Notice in the image above, there is a heading for the folder or workspace that is currently open. In the example image it says IT2320-JAMESLEASURE-FALL2018. If you hover your mouse over your folder/workspace title, it should give you a few options: New File, New Folder and so on. Click New Folder.



Enter a name of Lab1 and click Enter.

You should now see your new folder Lab1 listed under the local repository in Visual Studio Code. At the end of the semester, we’ll have folders for each Lab assignment.

Next, click New File. Enter index.html and Enter. Create a basic html file with body content:

|  |
| --- |
| <html>  <head>  <title>Lab 1</title>  <script src="script.js"></script>  </head>  <body onload="sayHelloJS()">  <h1>Hello from HTML!</h1>  <h1 id="js"></h1>  </body>  </html> |

Next, click New File. Enter script.js as the file name. Add the following JavaScript to script.js:

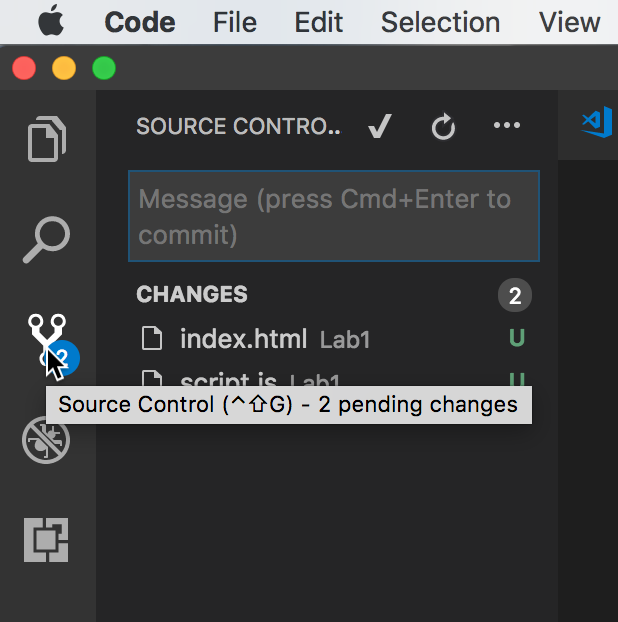
|  |
| --- |
| function sayHelloJS() {  document.getElementById("js").innerHTML = "Hello from JavaScript!";  } |

You can open index.html from file explorer or finder to see the output in a browser: 

Now that we’ve created the folder Lab1 and files required by the Lab example, we’ll now push the files (aka upload the files) to the github.com server.

# Step 9 – Push files to github.com

Click on the “Source Control” icon in Visual Studio Code.



We should pause at this point and talk about how we save changes to github.com. So, we have a few concepts to cover. First is the idea of a local and remote repository. Basically, we have a folder on our computer (local repository) and a folder on github.com’s server (remote repository). We want to keep these folders in sync. So, we work on files in our local repository. At some point, we want to copy the files from our local repository up to the server. This concept is called a push.

Local Repository Folder

C:\Code\IT2320

Remote Repository

https://github.com/James/IT2320

Commit + push

Note: the names are abbreviated ;)

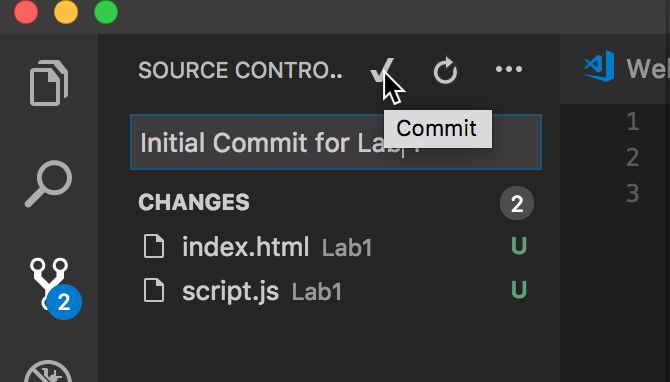
Commit– marks files for changes. In addition, we add a comment, so that we can view a history of commits in github.

Push – sends the files to our remote repository where current files are overwritten with the changes that exist in our local repository.

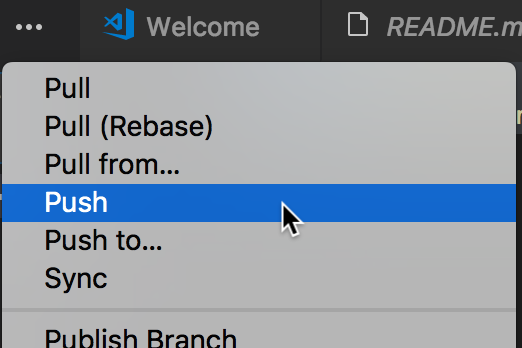
Now, back to the exercise!

Enter a Commit message. Remember this is a note for you (and me) to tell what happened during this commit. An appropriate commit this time might be “Initial Commit for Lab 1”

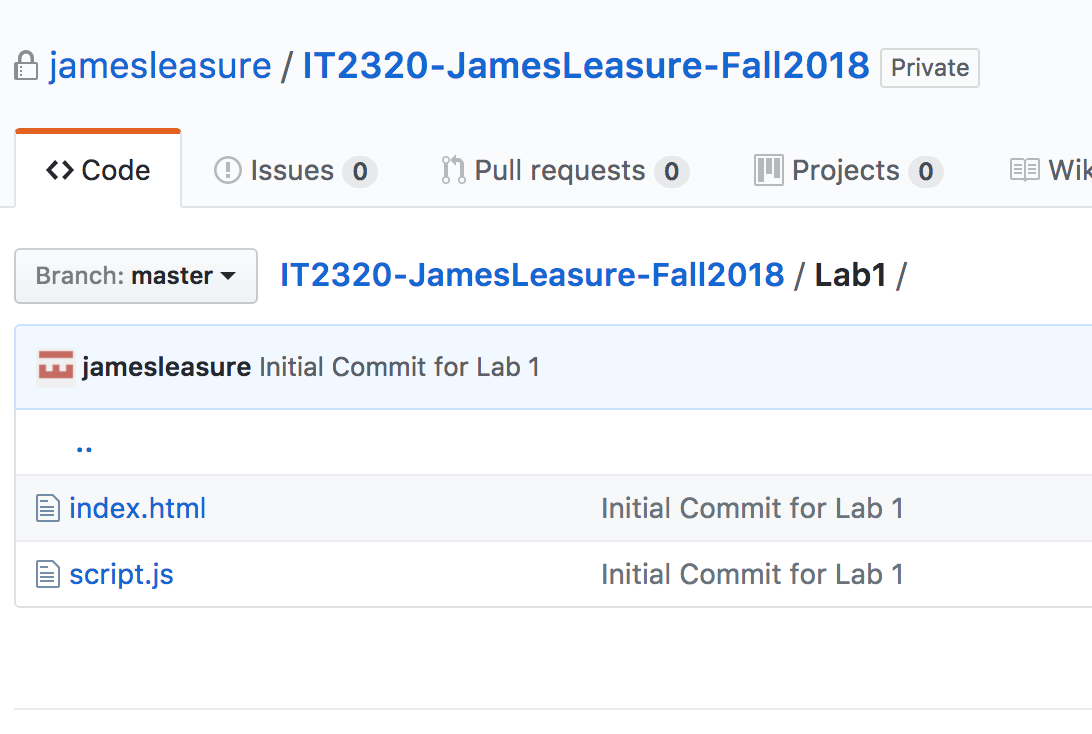
Next click the checkmark to Commit.



Now, click the … button and select “Push”



After pushing the files, you can log into your remote repository on github.com and you should be able to find the files for Lab1.



That’s it! Now, when you submit a lab in Blackboard, all you’re going to include is the URL to your Lab folder.

Oh yeah, make sure that you create all of your Labs under the same repository folder on your computer. That way you can simply push all of your labs to a single repository in github. I’ll know where the files are and you won’t have to go through the process of setting up a new repository each week.

Example:

IT2320-JamesLeasure-Spring2018

* Lab1
* Lab2
* Lab3, etc.