APPRENTICESHIP TRAINING CLASS PRE-WORK ASSIGNMENT

A s part of the application process for the Techtonic Apprenticeship Program, you are required to complete a series of online classes and exercises. This material might be a review for some applicants, or the first time you have been exposed to these technologies. Either way, you must complete the pre-work before you will move forward in the application process.

This pre-work will shed some light on the concepts and knowledge required to be a software developer. It will also help us assess your ability to absorb this material. Depending upon your current knowledge of software development, the pre-work will take around 40 hours to complete.

Step 1 - Basic Internet Terminology and Understanding

Before diving into the technical and practical aspects of coding, you'll need to have a basic understanding of how the underlying *environment* works. This introduction is meant to provide a quick overview of both the Internet and the Web. Please read sections 1.1 to 1.4.

Step 2 - Commonly Used Terms

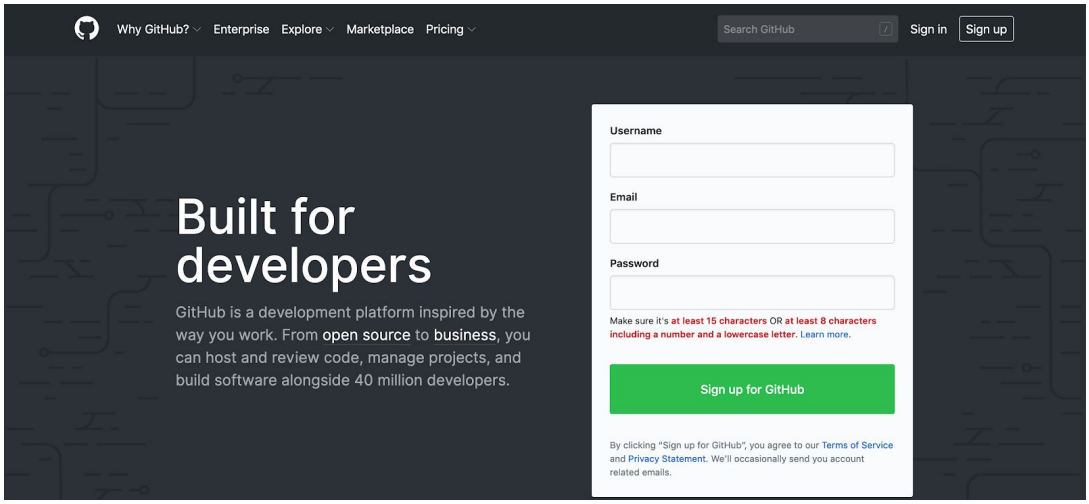
You will need to conduct some independent research to help ensure you have a decent level of understanding of basic industry jargon. To help get you started, here's a helpful site that provides some of the commonly used terms you'll encounter in web development.

Step 3 - GitHub Account Set Up

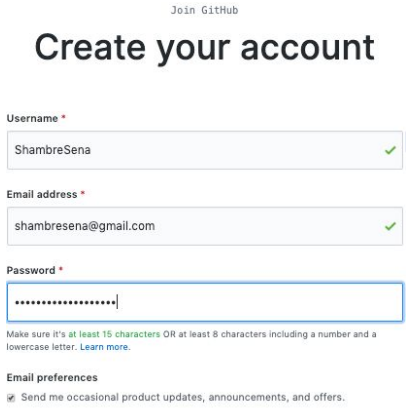
GitHub is a way for us to manage code (locally, *meaning on your machine*, and remotely, *meaning on a cloud server*). It provides a platform for developers to collaborate on the same code base remotely.

Step 3a - Create a GitHub Account

Visit GitHub. Create a personal GitHub account by clicking the **Sign up** button in the top right corner, preferably using your first and last name. Be sure to write down your login credentials, as you will need them again.



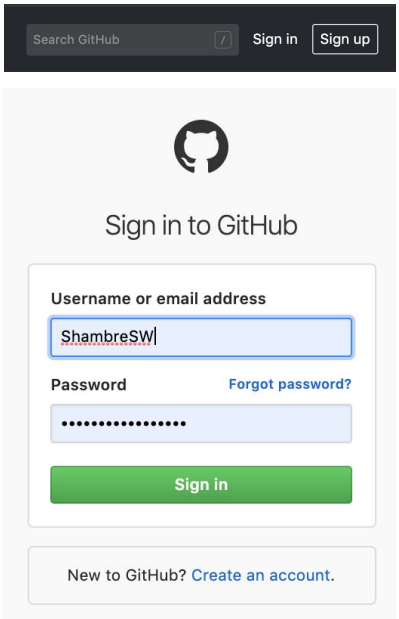
APPRENTICESHIP TRAINING CLASS PRE-WORK ASSIGNMENT



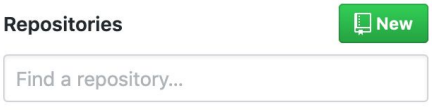
You will have to complete a puzzle to verify your account and select the free plan option. You will also have to verify your email address by going to your email and clicking the given link to complete your set up. Now you're ready to begin!

Step 3b - Create a Repo to House Your Work in GitHub

Sign in with the account credentials you set up in Step 3a by clicking **Sign in** in the upper right-hand corner of the GitHub website and inputting your username and password (yours should look the same as the one above, however the image below is from another dummy account).



Once you have signed in, click on the **New** button on the top left corner of your GitHub window.

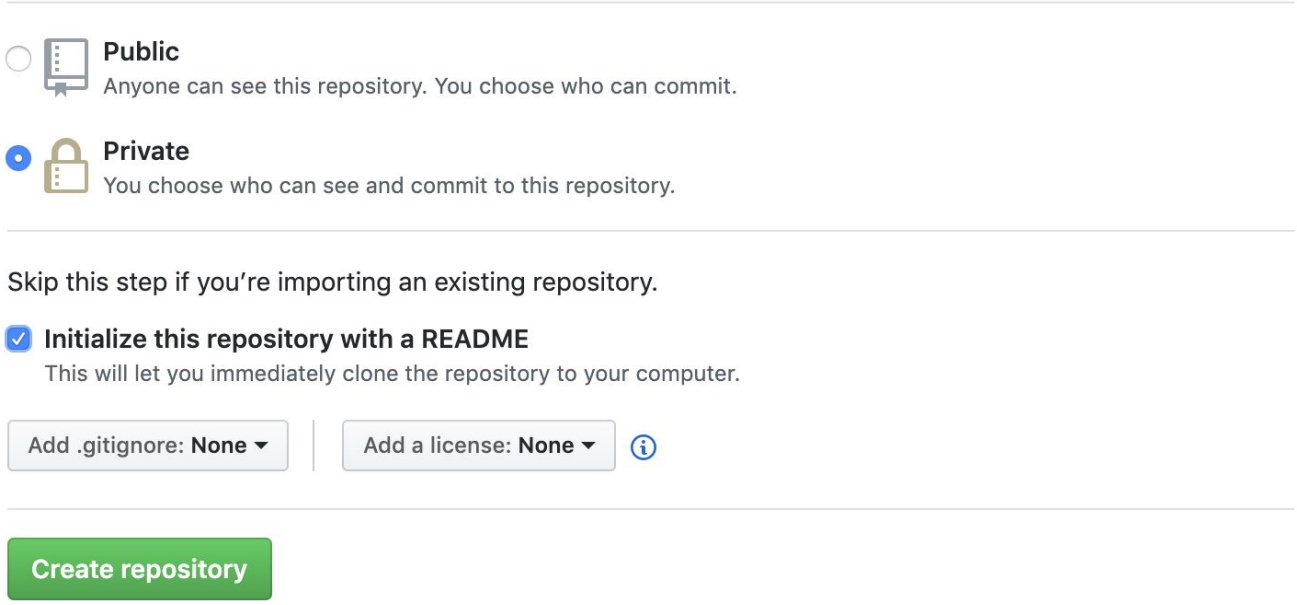


APPRENTICESHIP TRAINING CLASS PRE-WORK ASSIGNMENT

Under **Repository Name**, copy and paste "techtonic-apprenticeship-initial-pre-work" (without the quotation marks) in as the repository's name.

Select the following items:

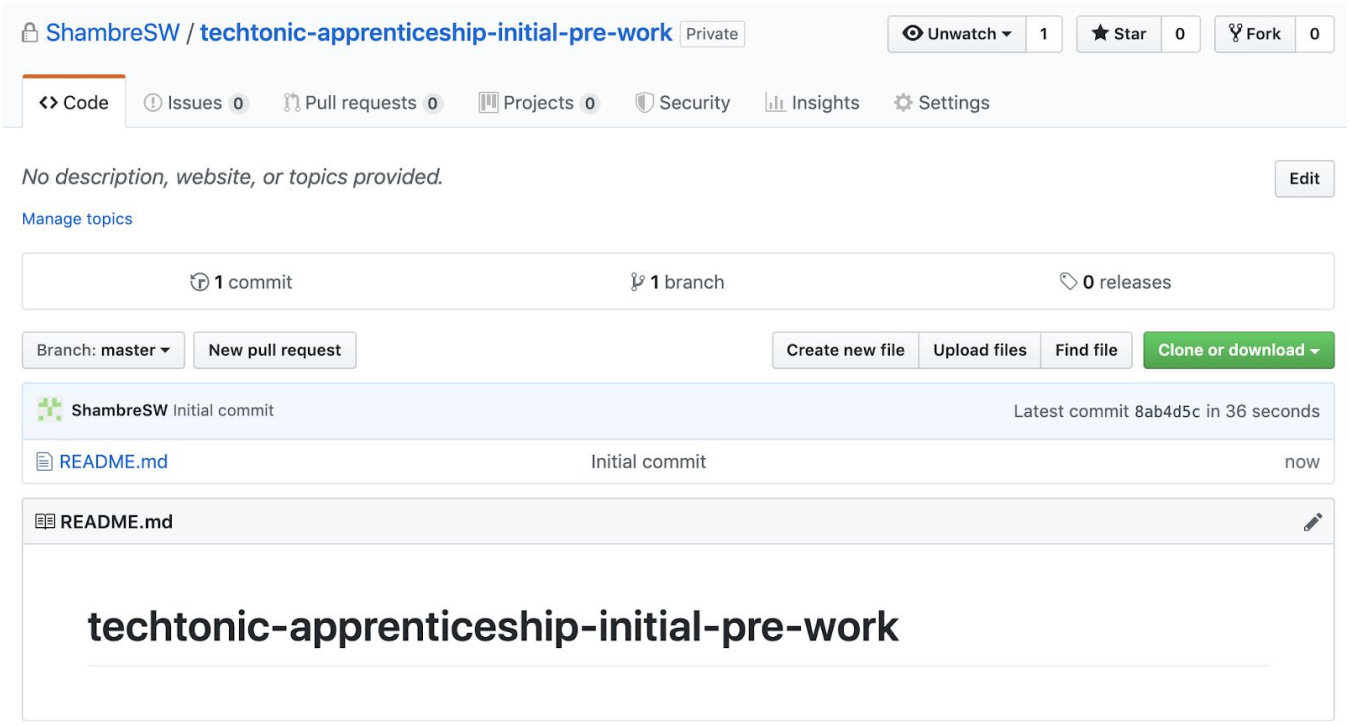
- **PRIVATE** (You choose who can see and commit to this repository)
- **Initialize this repository with a README file** (This is where you will record your progress through the pre-work exercises).
- Then click **Create Repository.**



You will make changes to the README file to help us understand your current proficiency with each concept once you begin doing the freeCodeCamp lessons in Step 5. Leave it blank for now.

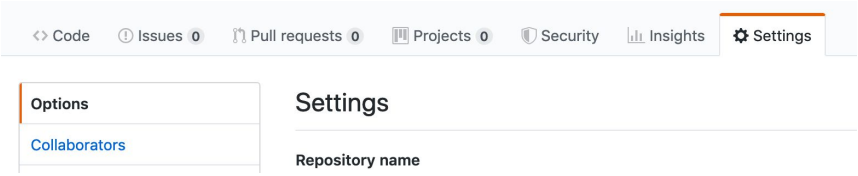
This is what your GitHub should look like after all of this is completed:

APPRENTICESHIP TRAINING CLASS PRE-WORK ASSIGNMENT

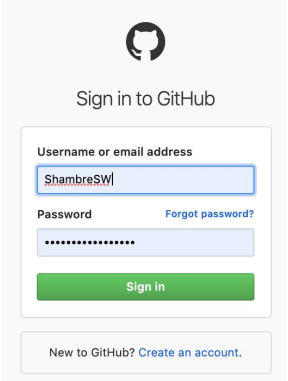


Step 3c - Share your Repository with Us

Once you get to your repo page, go to **Settings** on the right hand side of your screen and then **Collaborators** on the left hand menu.

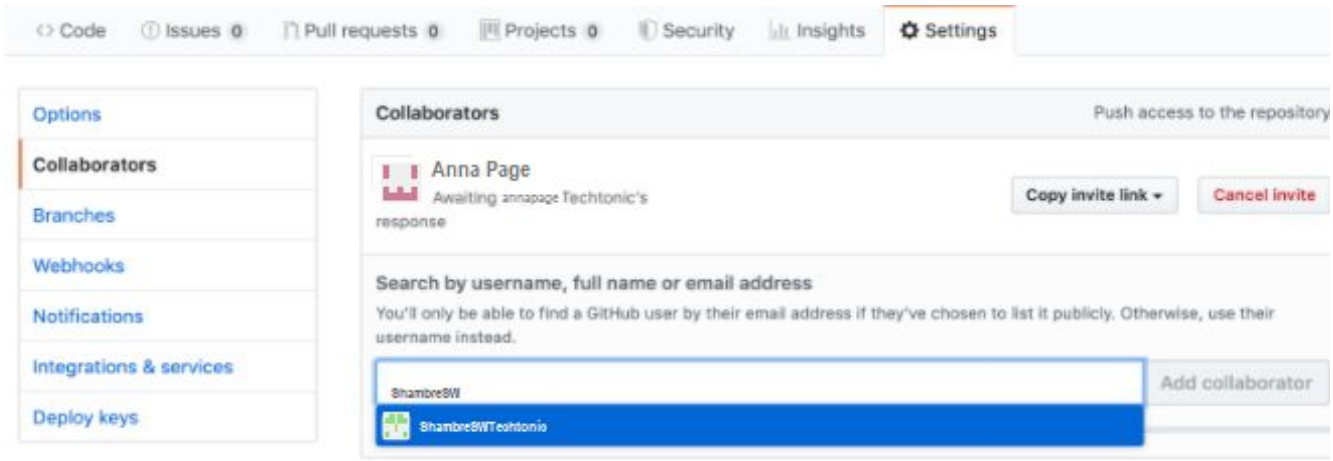


Insert your password again when prompted (the same you created in Step 3a above).



APPRENTICESHIP TRAINING CLASS PRE-WORK ASSIGNMENT

Add our team as collaborators by inserting our GitHub account handles into the text field one at a time - **annapageTechtonic, shambreswTechtonic, scullenTechtonic**. Select our names and click **Add Collaborator**.



By adding us as collaborators, we will be able to see that you have begun working on your pre-work in your repository.

Step 3d - Take a Break from GitHub to Install Git for your Operating System

It is very important to note that **Git and GitHub are different**. Git is a version control system that records changes to a file or set of files over time so that you can recall specific versions later. GitHub is a code hosting platform for version control and collaboration. It lets you and others work together on projects from anywhere and keep track of changes made by all collaborators over time using Git for shared version control.

To install Git command line tools for your computer's operating system click, you will need to work from within your computer's command prompt (Windows) or terminal (Mac) program. You should now have two windows open on your machine (GitHub in one and your command prompt/terminal in the other).

*If you've never worked with the command prompt/terminal it can feel intimidating. See this command prompt/terminal tutorial first for some self-teaching on how to navigate through your computer using various commands. **You do not need to buy anything, simply use the free resources there**. The arrows at the bottom right and left will allow you to navigate through the lessons for free.

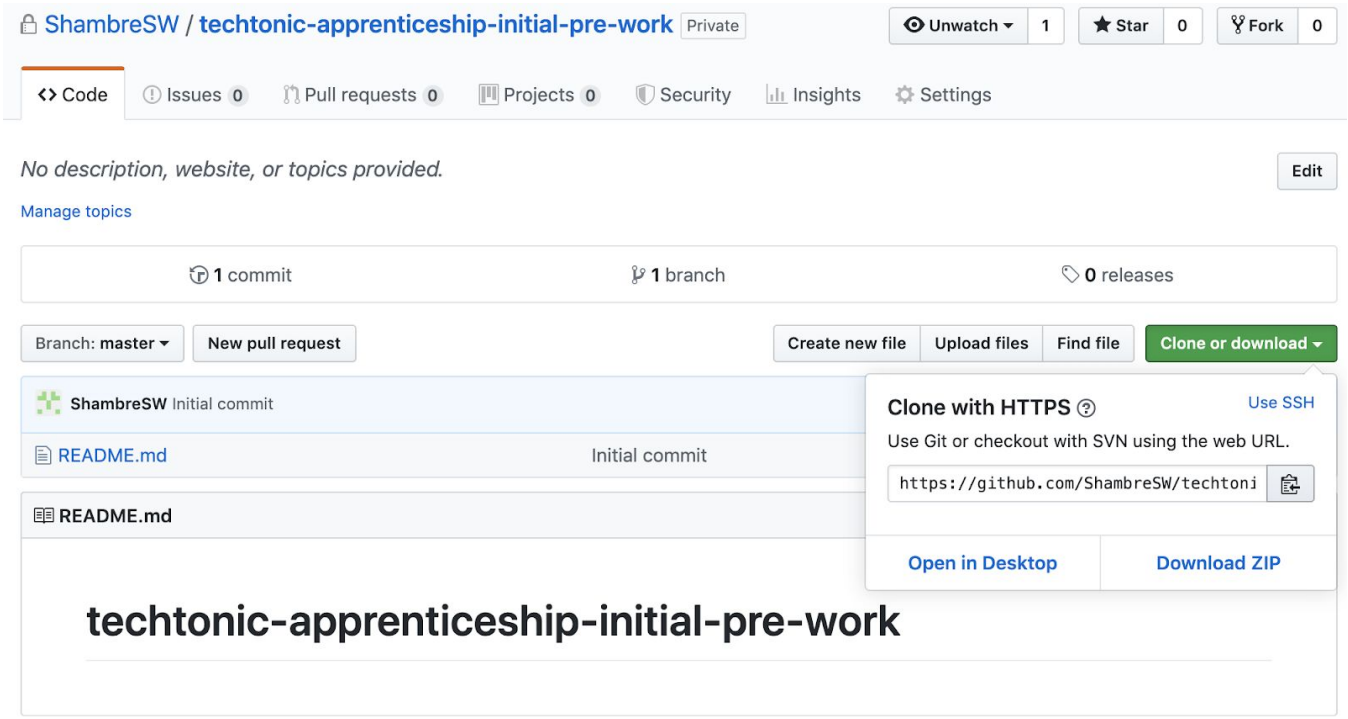
Once you feel ready to tackle the installation, click on the link below that matches your operating system and follow the instructions therein.

Mac OS X
Windows
Linux

APPRENTICESHIP TRAINING CLASS PRE-WORK ASSIGNMENT

Step 3e - Go Back to GitHub

Assuming continuity from Step 3c above, once you have added collaborators in GitHub, you will click on the "<> **Code"** tab in the top left corner of the screen. Next, go to the "**Clone or download**" tab. Hit the arrow to access the drop down. Click the clipboard icon to copy the URL for your remote repository (the one that lives on GitHub).



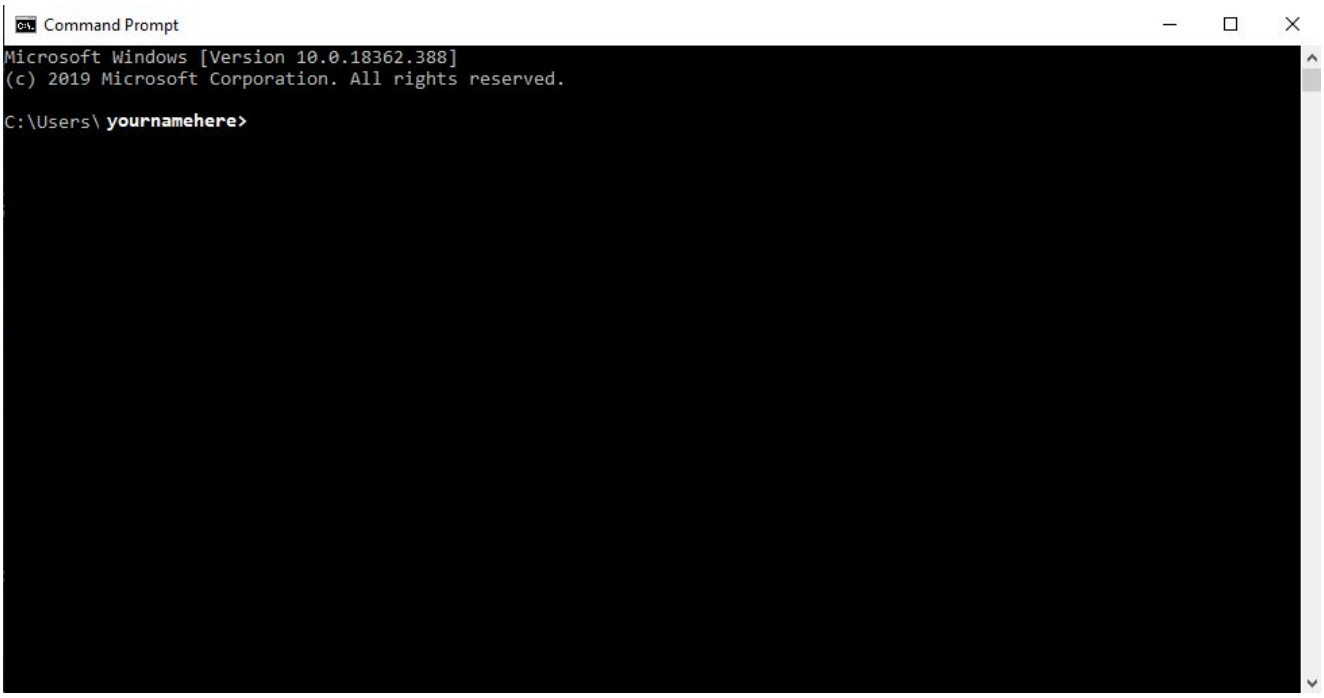
You will use this URL for the git clone command to "clone" (download) the project to your local machine. This will be the copy that you add your own work into on your machine.

Step 3f - Go Back to the Command Prompt/Terminal to Clone the Remote Repository to your Local Machine

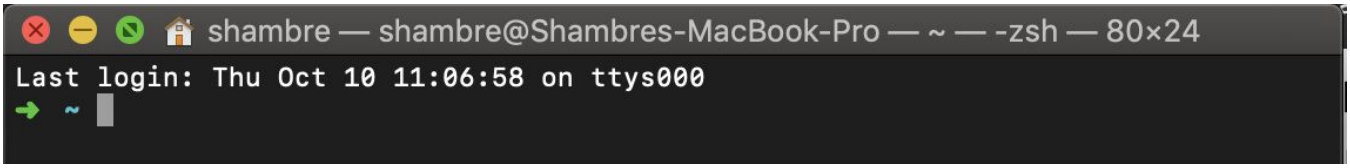
APPRENTICESHIP TRAINING CLASS PRE-WORK ASSIGNMENT

Open up the Command Prompt (Windows) or Terminal (Mac).

Windows:



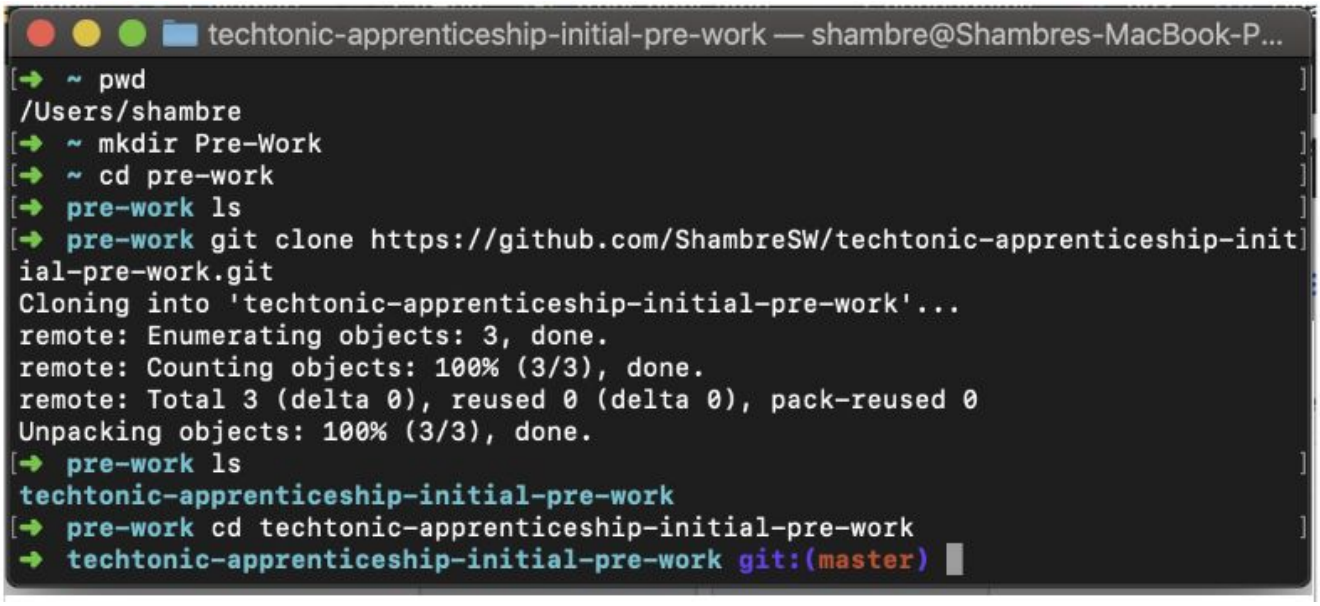
Mac:



Next you will want to create the folder that you would like to work on the pre-work in (reference the same tutorial listed above for help doing this - command prompt/terminal tutorial).

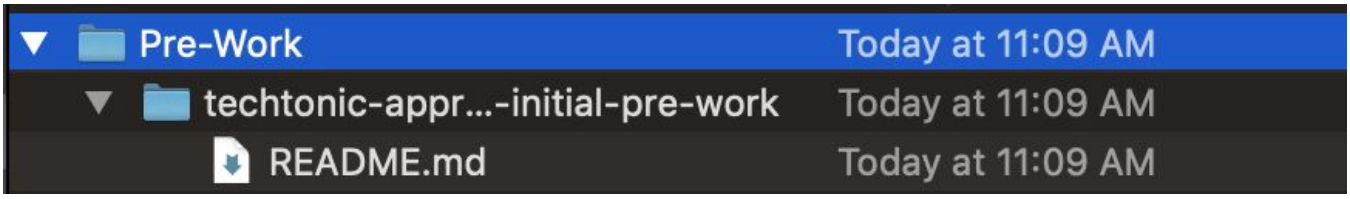
We recommend naming the folder Pre-Work. Once you've created this folder, navigate to it using your command prompt/terminal) and run the git command below with the URL you copied to your clipboard in Step 3e from GitHub:

git clone <insert clone with https url here>



This will run the git command to download the pre-work repository to your computer. The folder will be created by default and it will be named after the git repository called techtonic-apprenticeship-initial-pre-work. This step initializes this folder with a .git file that allows tracking of everything that happens inside of this particular folder.

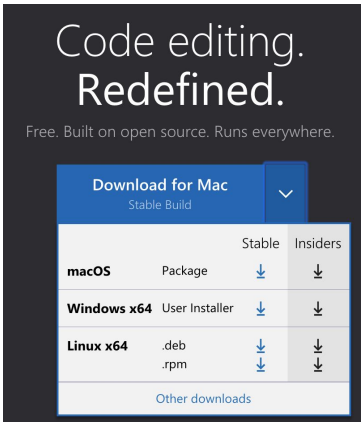
Your folder structure (found inside of your Finder on a Mac, or inside of your My Computer menu on a Windows OS) will look something like this:



We will come back to do more work on this repository soon, but first, you will need to install an Integrated Development Environment to work in.

Step 4 - Install the Visual Studio (VS) Code Integrated Development Environment (IDE)
Got to Visual Studio and choose the **STABLE BUILD** release to download for your operating system.

APPRENTICESHIP TRAINING CLASS PRE-WORK ASSIGNMENT

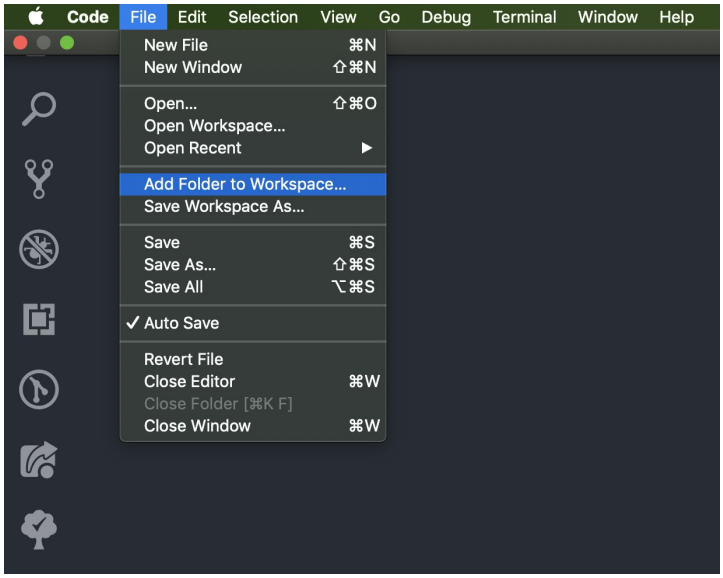


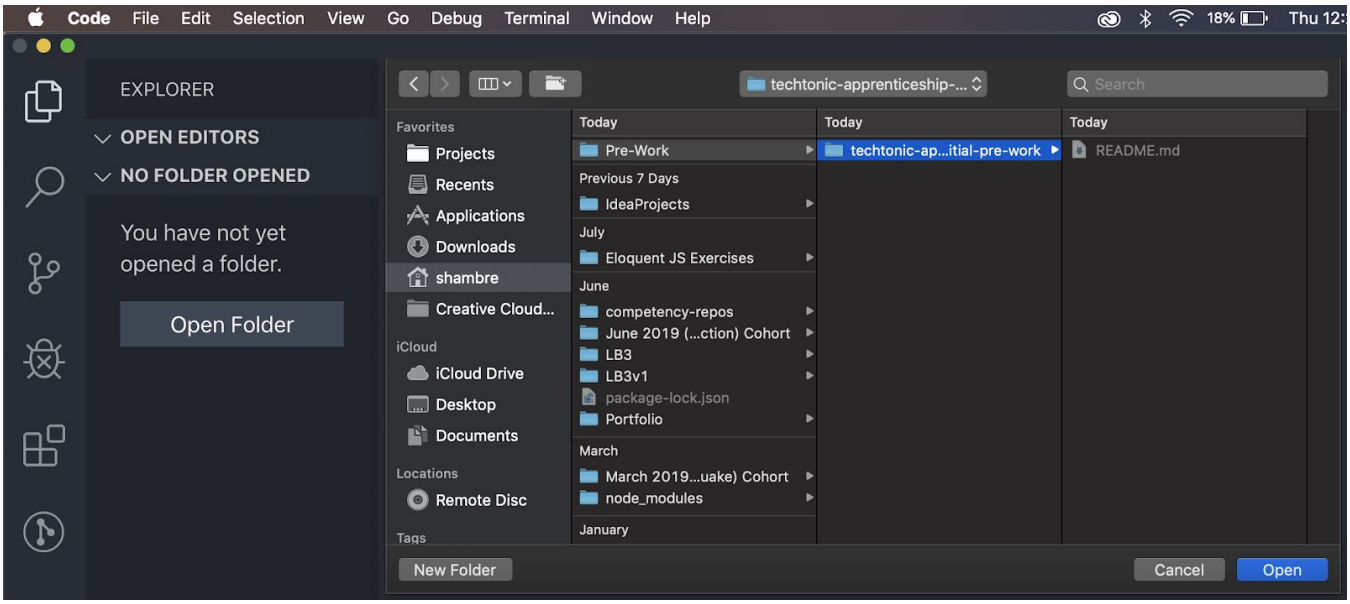
Step 4a - Open VS Code

Once the VS Code IDE is installed, open it. You will now have three windows open on your screen (GitHub, your command prompt/terminal where you've navigated to right inside the techtonic-apprenticeship-intial-pre-work folder that was cloned earlier, and your VS Code application).

Step 4b - Access your Pre-Work Directory in VS Code

Click **File → Add Folder To Workspace** in the top left of your VS Code window and then select the folder that you created with the git clone command (techtonic-apprenticeship-initial-pre-work). This will open up this directory in your workspace in VS Code so you can modify your code base and save your changes.

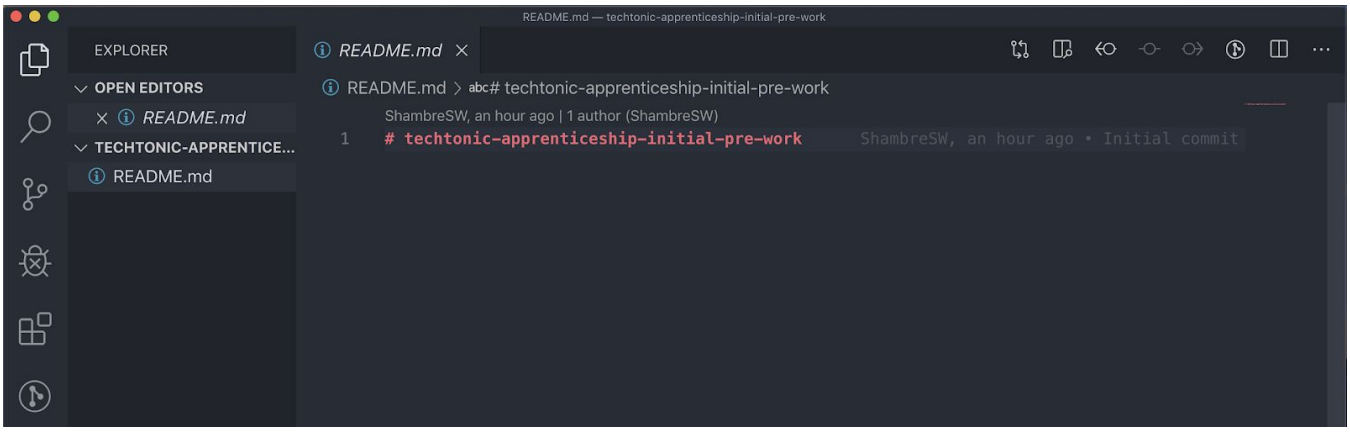




Since you have initialized Git in this folder structure, when you save your changes in VS Code, they will be tracked behind the scenes. You can use your command prompt/terminal to gain insight into these changes using various commands. Certain commands, like add and commit will allow you to take changes you make locally on your machine and commit them to your remote repository on GitHub. However, until you push all changes to your remote repository, they will only be visible to you (not your collaborators).

Step 4c - Practice Integrating VS Code, Your Command Prompt/Terminal, and GitHub
To get a sense of how this process will work, let's try integrating all three tools to make some file updates.

Your README file will look like this once you have opened it in VS Code:



In this file, we have used a special syntax called Markdown (the file extension at the end of README.md stands for MarkDown). The hashtag in front of techtonic-apprenticeship-initial-pre-work simply signifies that those words should be displayed in the README on GitHub's main page as a level 1 heading.

APPRENTICESHIP TRAINING CLASS PRE-WORK ASSIGNMENT

The resources below explain what Markdown is and give a guide to how it works. We are providing a template you can use repeatedly inside of this file, so that you can simply change a few module and lesson titles and substitute your own notes in. **You will follow the same procedure outlined below for adding notes on each lesson that was difficult for you to your README file.**

This will help us gauge the areas you are confident in and the areas you will most likely need more support around developing. In case you are interested, you can read more about Markdown below, but if you follow the highlighted directions underneath those resources, you shouldn't need to delve too deeply into Markdown at this time:

What is Markdown and Why Should I Use It?
Markdown Cheat Sheet

Step 4d - Change Your README File and Save It

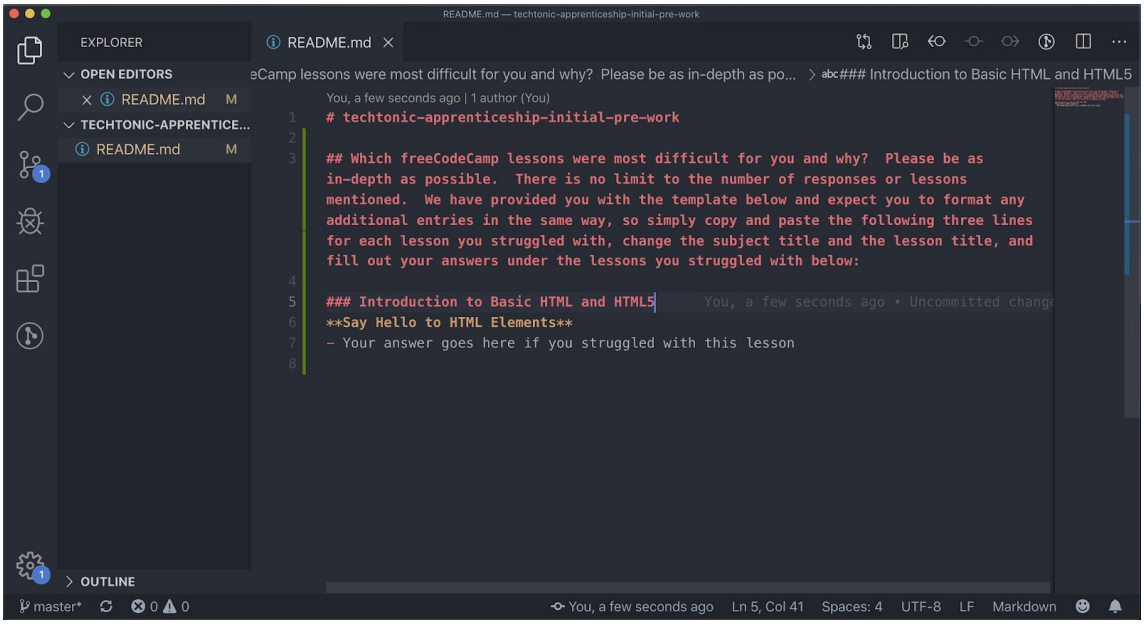
Copy and paste the following paragraph and markdown file template (the whole yellow section) into your README file.

Which freeCodeCamp lessons were most difficult for you and why? Please be as in-depth as possible. There is no limit to the number of responses or lessons mentioned. We have provided you with the template below and expect you to format any additional entries in the same way, so simply copy and paste the following three lines for each lesson you struggled with, change the subject title and the lesson title, and fill out your answers under the lessons you struggled with below:

Introduction to Basic HTML and HTML5
Say Hello to HTML Elements
- Your answer goes here if you struggled with this lesson

APPRENTICESHIP TRAINING CLASS PRE-WORK ASSIGNMENT

Your VS Code file should look like this now:

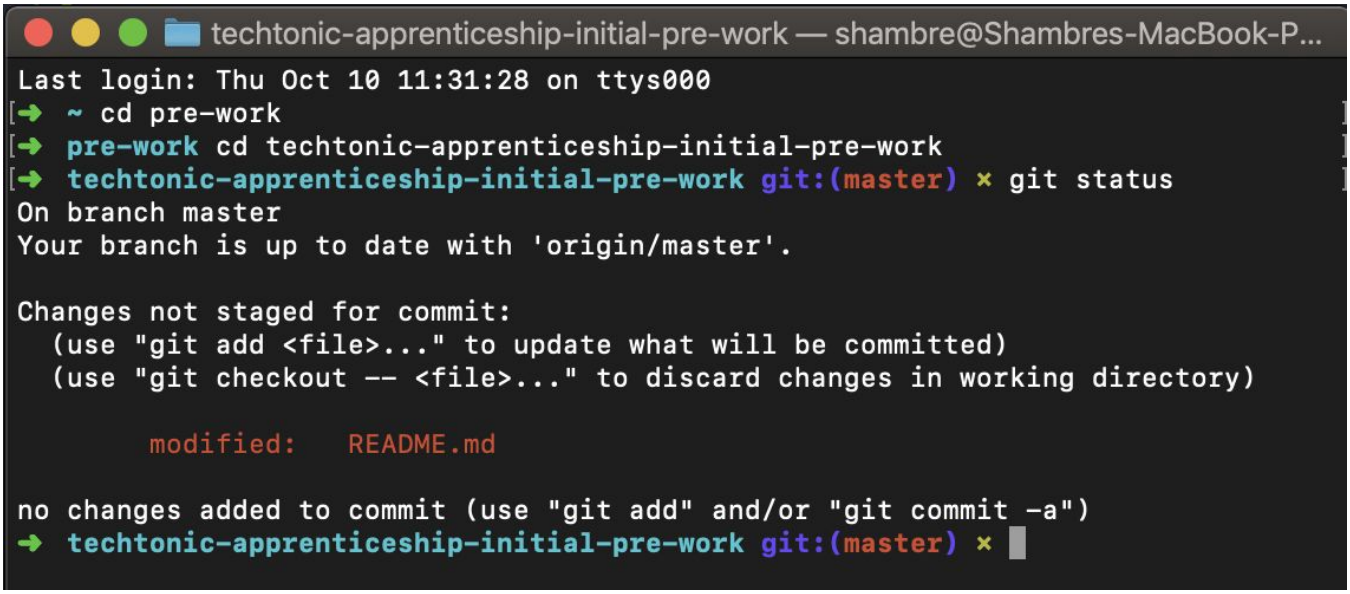


You've just modified your first file in VS Code! Yay YOU! In order for us to see this modification and any more you will make later, you will need to commit your work to the repository you created in Step 3. Let's go over how to do that together now. Before we do, **be sure to SAVE the change you just made in VS Code** using the File menu at the top left, or the appropriate shortcuts for Windows (Ctrl + S) or Macs (Command + S).

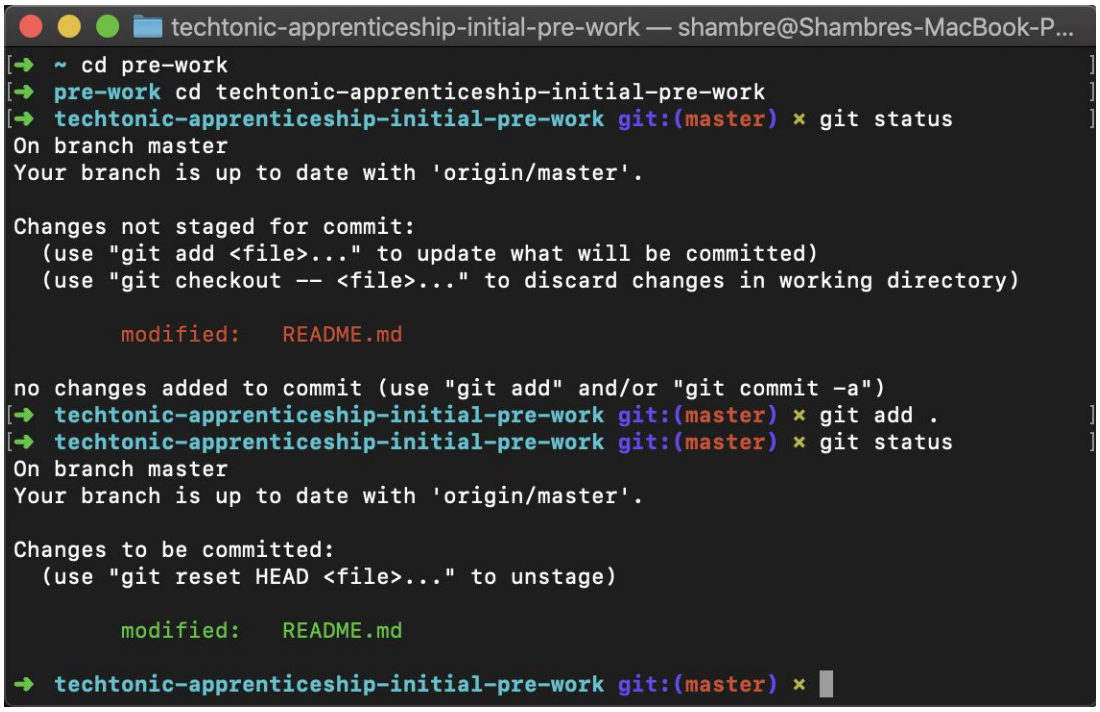
Step 4e - Use Common Git Commands to Track Your Changes

Let's start by getting an idea of what files may have been modified by running **git status in your command prompt/terminal**. This will show us if any files have been modified so far. There have been, because we pasted some code into our file and saved it. Therefore, Git shows us that it is tracking the changes in this repo by showing the modified files in red, as seen below:

APPRENTICESHIP TRAINING CLASS PRE-WORK ASSIGNMENT

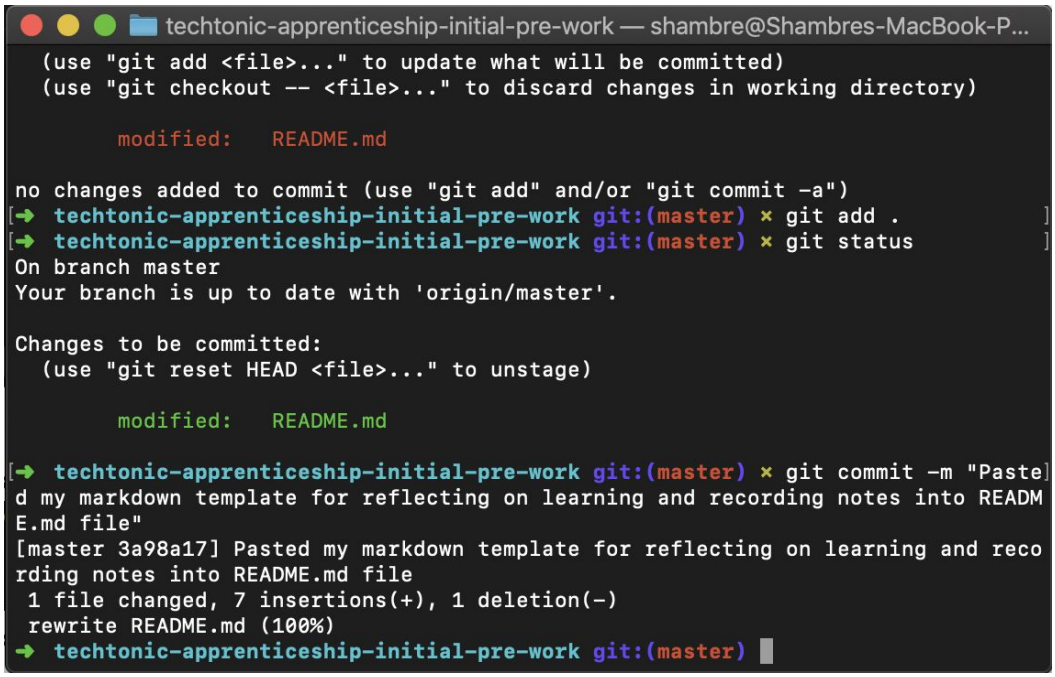


Next you will need to stage the changes to be committed. To do this, run the command **git add . (the period is part of the command, so be sure to include it).** After doing this, you can run **git status** again and you should now see your changes staged and visible in green, as seen below:



Next you will want to commit the staged changes using **git commit -m "<your message>"**

The example message reads “Pasted my markdown template for reflecting on learning and recording notes into README.md file.”



Your number of insertions will likely differ from the screenshot above but that is ok. Git is just letting you know how many changes were committed.

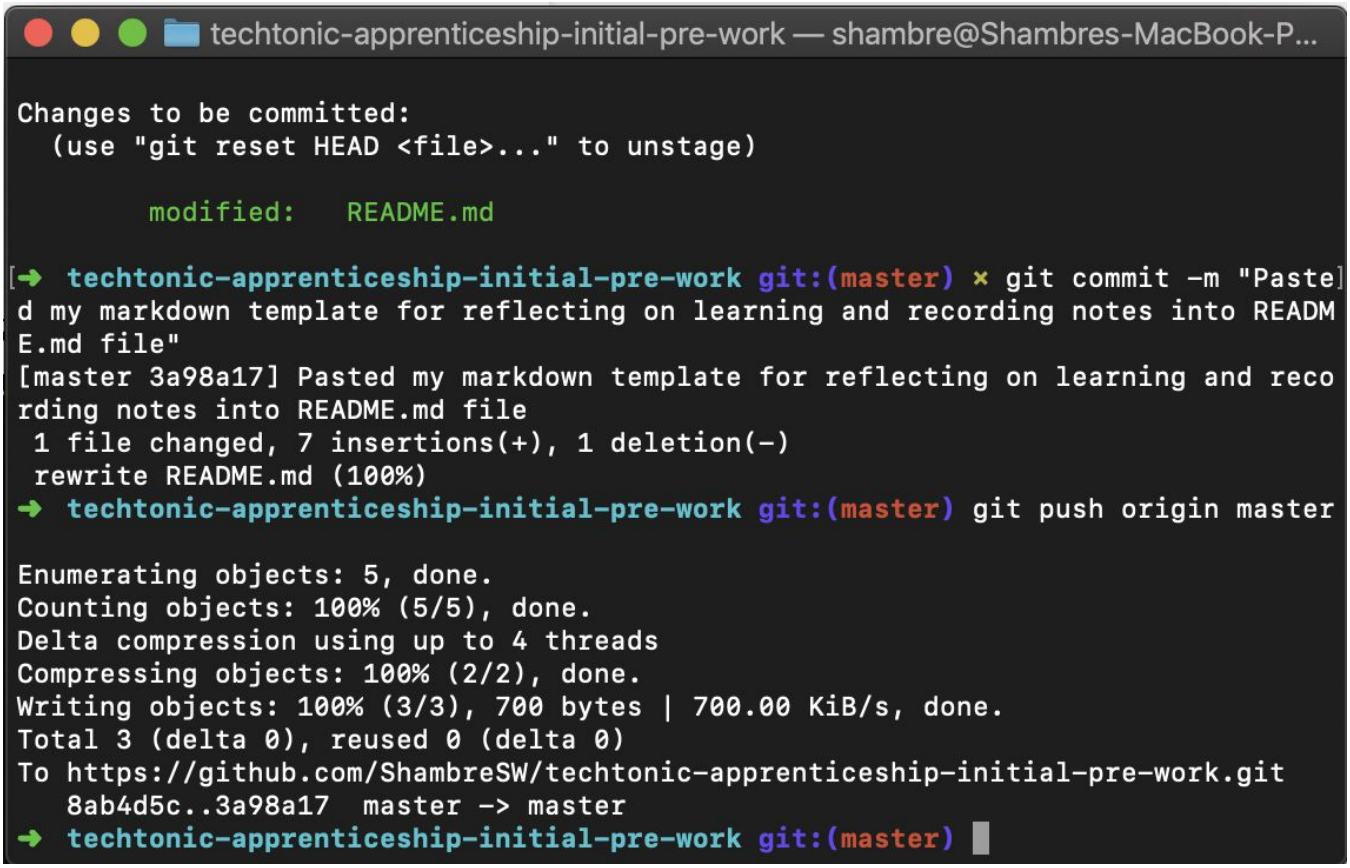
There are very specific methods and conventions for writing high-quality commit messages. For more information on these, check out this article.

After committing your changes, it’s time to push them to the remote repository so that they will be available on the internet for our team to review.

Step 4f - Push Your Changes to Your Remote Repository
To do this run the command **git push origin master** in your command prompt/terminal window.

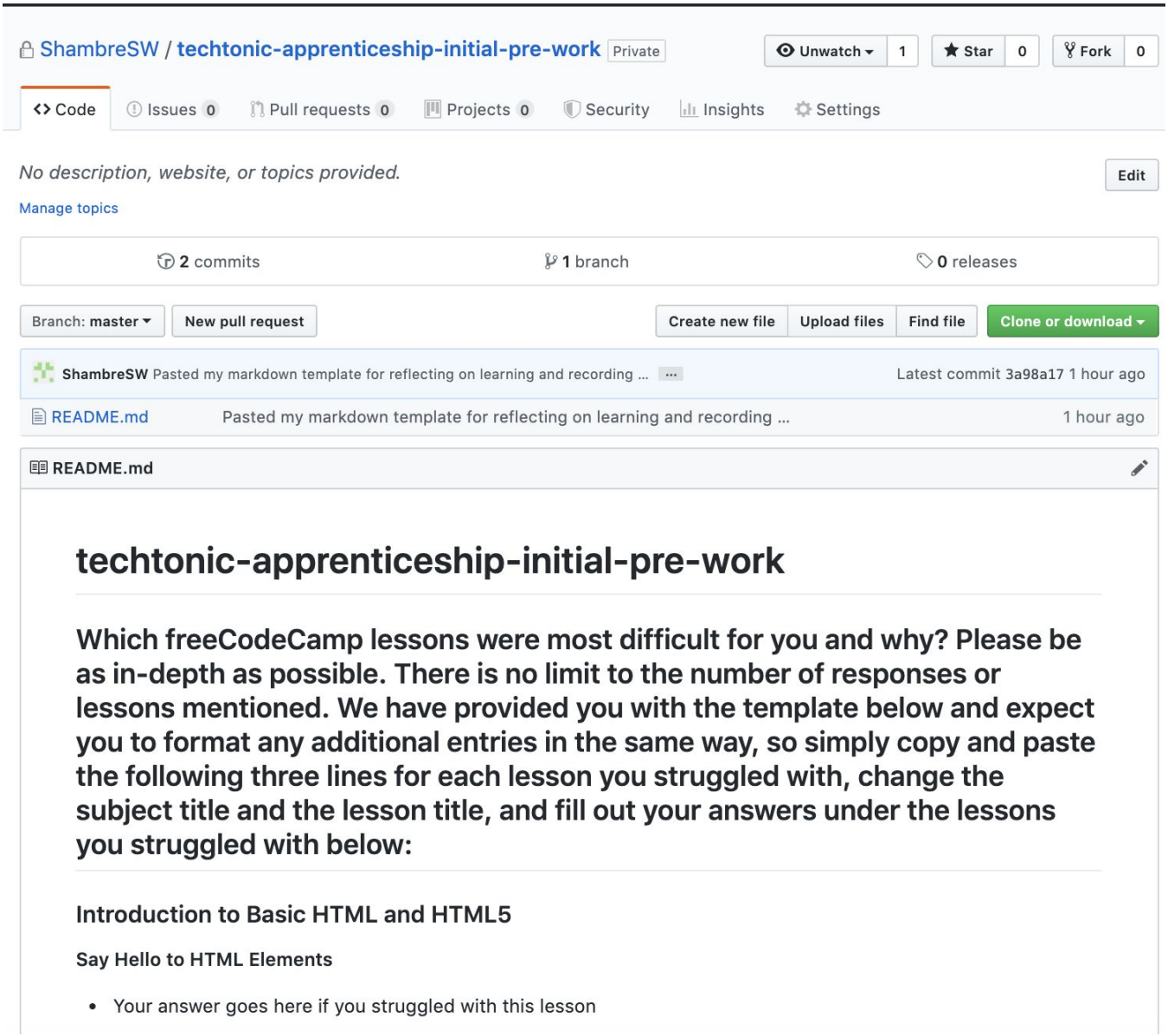
This tells the computer to push (send) the commit (your updated code) to GitHub to be held online in your remote repository. Now, we can see what you’ve added! Screenshot example below:

APPRENTICESHIP TRAINING CLASS PRE-WORK ASSIGNMENT



***If you've set everything up appropriately and followed the steps, you should NOT need to change the remote origin.** If you are being prompted to do so, you have made a misstep. Please start over.

Now, if you go to your GitHub account, your GitHub page should look like this:

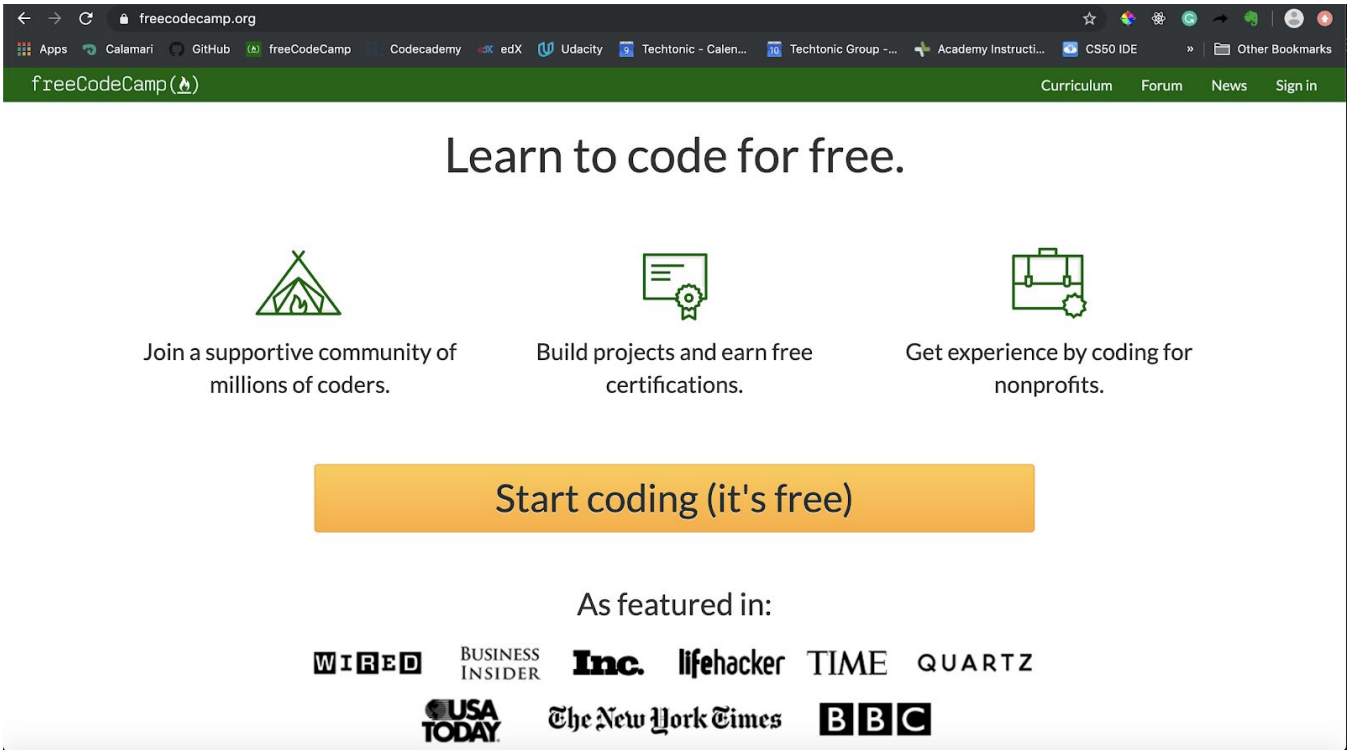


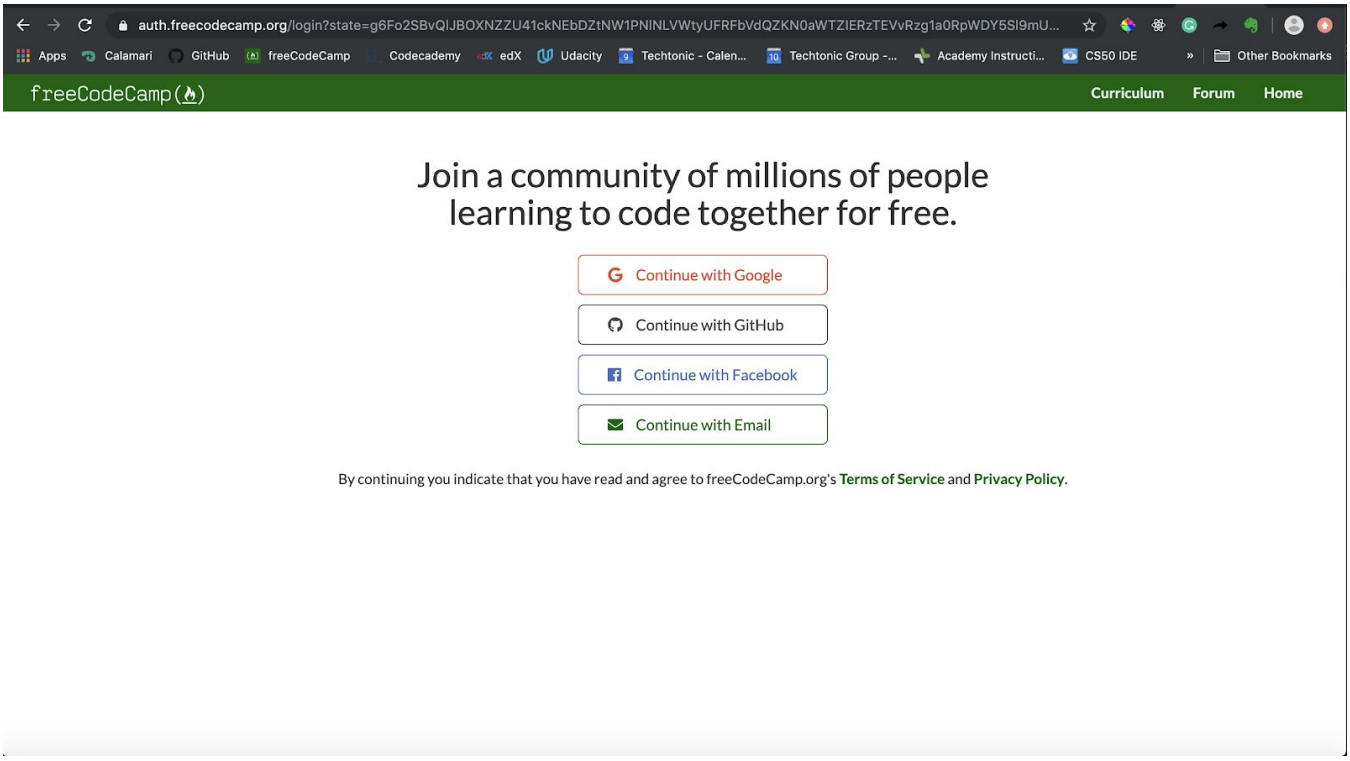
You've done it! You've successfully changed and shared your file with us!

Step 5 - Create a freeCodeCamp Account
Go to freeCodeCamp.

Click "**Start coding (it's free!)**" and choose whichever option of the four available that you prefer for signing up to create an account. **Hold on to your credentials.**

APPRENTICESHIP TRAINING CLASS PRE-WORK ASSIGNMENT





APPRENTICESHIP TRAINING CLASS PRE-WORK ASSIGNMENT

Once you've created your account, go to the Curriculum tab at the top right of the window to proceed to the actual learning modules and start at the beginning with the Basic HTML and HTML5 module.

Step 5a - Complete Each freeCodeCamp Exercise and Reflect and Record Notes on Your Learning by Updating Your README file

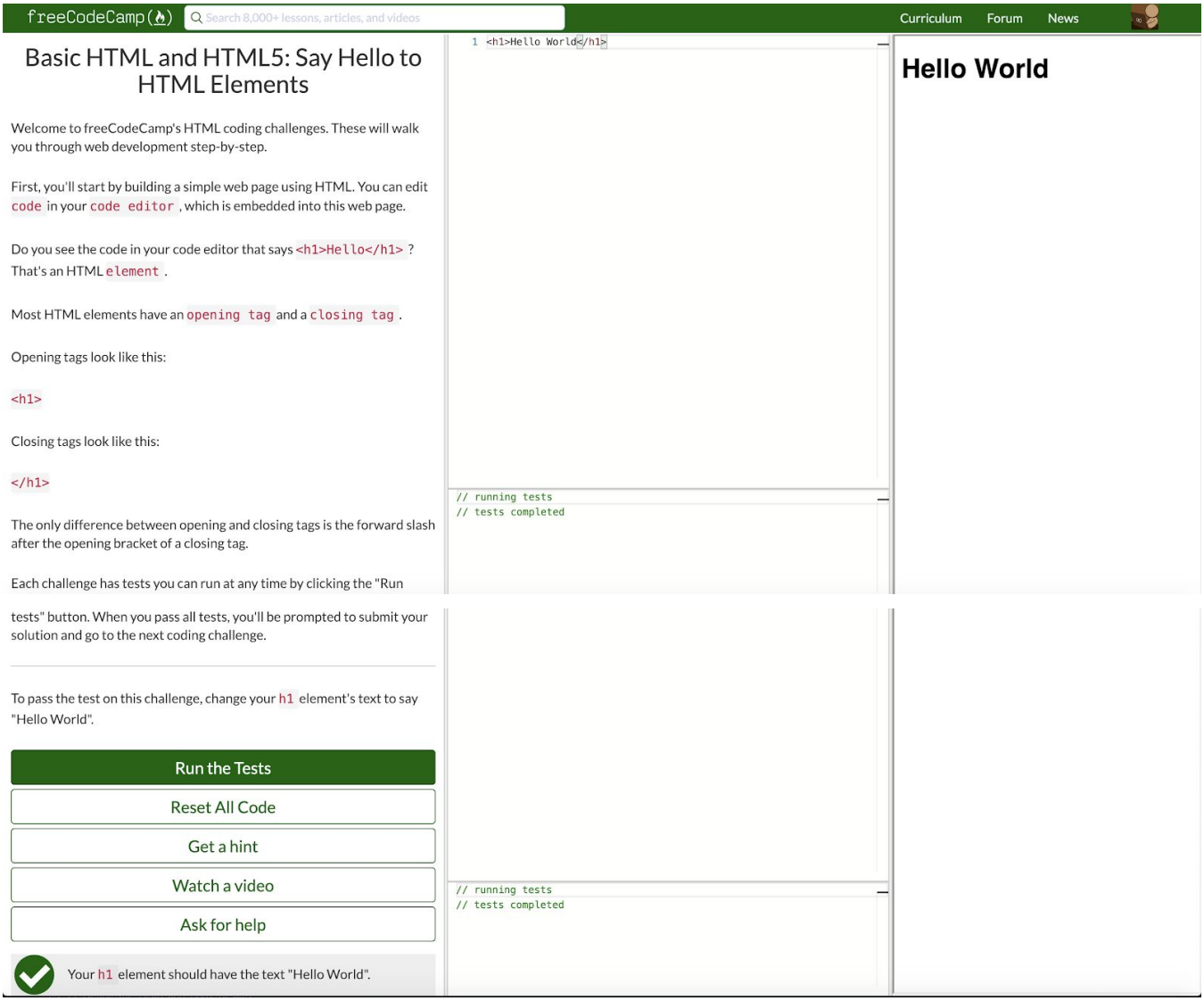
Now that you have all the tools installed and one successful commit under your belt, you can start the freeCodeCamp exercises! You will go through **six** different modules with several lessons in each section. Please make sure to complete each module (in blue below) and lesson (the total number is listed in black beside the module name) in their entirety. This will help you gain an understanding of front-end technologies taught during the classroom training. We highly recommend taking notes, quizzing yourself, and talking to friends/attending meetups because it will force you to speak about programming with others. Become confident in the "language" of web development if you have time to do so. It will greatly benefit you with your face-to-face interview!

Class modules to complete in freeCodeCamp:

1. Introduction to Basic HTML and HTML5 (27 activities)
2. Introduction to Basic CSS (44 activities)
3. Introduction to the CSS Flexbox Challenges (17 activities)
4. Introduction to the CSS Grid Challenges (22 activities)
5. Introduction to JavaScript (107 activities)
6. Introduction to the ES6 Challenges (26 activities)

Your freeCodeCamp account will look like this when you open up the first lesson of the first module:

APPRENTICESHIP TRAINING CLASS PRE-WORK ASSIGNMENT

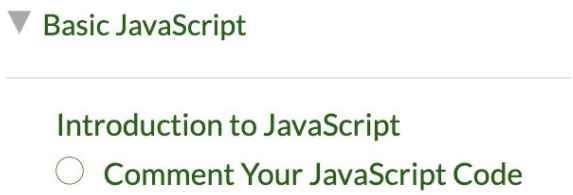


Follow the directions to attempt and submit code for each lesson of freeCodeCamp for the modules outlined above. Then, ensure that you save your work and **follow the steps from 4d - 4f** above (summarized briefly below) to update your README file by answering the question posed in the README template for **lessons that you struggle with**.

Reuse this template as often as needed (changing the name of the module and lesson to reflect the one you're leaving notes for each time):

Introduction to Basic HTML and HTML
Say Hello to HTML Elements
- Your answer goes here if you struggled with this lesson

For instance, let's say that the first area you get stuck on is the first lesson of the first JavaScript module.



Then, here's how the above template (in blue) would look with modifications in your README file:

Introduction to ***JavaScript*** ← The module title has been changed
Comment Your JavaScript Code ← The lesson title has been changed
*- I struggled with this lesson because it didn't make sense to me that I could tell the browser to completely ignore parts of the code I worked so hard to include. However, I see the usefulness, because comments allow me to communicate with the next person maintaining the project, so now it makes sense why this is an option for me to use.** ←* THIS IS AN EXAMPLE OF AN ANSWER - DO NOT USE

Now, repeat this process until all the exercises are completed in freeCodeCamp and all your exercise notes are completed, pushed, and committed to GitHub! We only require notes on areas **you struggled with**, however, if you'd like to add notes for all lessons so that you can easily refer to them later, we will accept them. Just be sure to make it easy for us to EASILY tell which notes pertain to the parts that tripped you up. We will use this information to help us gauge where you may need the most support.

git add .
git commit
git push origin master

If you have any trouble with the Git commands above READ THE ERROR MESSAGE. Git is very helpful in pointing out what went wrong if it's given an incorrect command and it will usually provide text describing exactly what you need to do for the command to run successfully. If you're still having trouble please visit the Git tutorials from the resource section below.

Step 6 - Finalize Your Pre-Work Submission
Send an email to apprenticeship@techtonic.com informing us that you have completed all of the pre-work and are ready for it to be reviewed. Please also **provide the name of your Github account/handle** in the email.

After your pre-work has been sent to apprenticeship@techtonic.com to be reviewed, someone on our team will be in contact with you. If you don't hear from us after 5 business days, please reach out to apprenticeship@techtonic.com.

APPRENTICESHIP TRAINING CLASS PRE-WORK ASSIGNMENT

HELPFUL RESOURCES:

These are <u>not</u> required, but all may prove VERY FRUITFUL if you have the time to explore them. We <u>STRONGLY RECOMMEND</u> all of them.

- Comparison of Windows and Mac Terms
- Markdown Cheat Sheet
- MDN Teach Yourself Web Development
- Additional Web Tutorials on HTML/CSS and JavaScript.
- CS50's Introduction to Computer Science (sign up for an account, but it's free)
- Eloquent JavaScript
- Flexbox Zombies (Try out Flexbox Zombies to help practice flexbox)
- Google "How to Think Like a Programmer" and download a pdf of V. Anton Spraul's book to read. Then, attempt the problems presented in the book!
- GitHub Tutorials - The more tutorials you look into, the better off you will be. There are a variety of resources here that will help you, so take the time to delve deeper for maximum benefit.
- Git GitFlow Workflow Explanation/Interactive Tutorial - It's imperative that you have a good understanding of the Git Gitflow Workflow, which you can get by studying this tutorial. This is one workflow (standard process) that Git users utilize frequently at Techtonic. Many of our projects here incorporate it, so it's worth spending time on and getting a good overview. We WILL use it in class.