**What is Git?**

Selecting transcript lines in this section will navigate to timestamp in the video

- [Instructor] Let's take a minute to examine what Git is. Now, Git is what's called a version control or source control system. In essence, it lets you manage changes you've made to files over time. With Git, you're basically a historian with a time machine. You're in charge of documenting the history of your project, with the added benefit of being able to jump back and forth through time. But unlike normal historians, you have the ability to rewrite history. Now, you can save checkpoints which are called commits and leave messages about what happened at each of these different checkpoints. The best part is another feature called branching that lets you create alternate versions of your code. It's like being able to create alternate realities of your project. A branch is a copy of your project that you can work on with or without changing the original. You can then synchronize branches, which is called merging, or go back and forth in between them. In order to use GitHub, you're going to need a few things. First, of course, you need to install Git from the Git website. Now, you may already have Git in your machine, but it's a good idea to make sure that you have the latest version. You can find Git at this URL right here git-scm.com and the SCM means source control management. From there, you can simply download an installer for your machine type and follow the instructions. The defaults are fine for most installations. However, if you're on Windows, one of the options is to install a terminal application called GitBash, which makes it easier to run Linux commands. That's what most people use when working with Git. However, the commands on a Windows machine are pretty similar. You just have to remember that to list files, you use DIR on Windows instead of LS on Linux. You'll also need a terminal to run commands. Now, if you're on a Mac, you can use the terminal application that comes with your machine or another terminal like Hyper which is also available for Windows. And that's what I'll be using in some parts of the course. I'll also be using Visual Studio Code and here's where you can get that. Finally, on most web projects, you'll be using something called Node.js, which handles most of the tools for the web. So you may want to go to the website and download the installer for that. If you're comfortable with those requirements, then let's get started.

**Installing Git and requirements**

Selecting transcript lines in this section will navigate to timestamp in the video

- [Instructor] Let's take care of setting up a project to use Git. Now the first step is to set up some of the configuration variables. Git is designed to be used by more than one person. You have to tell it who you are so that it will give you credit for the changes that you make. You can use the Git Config command and set up the username as well as the user email. If you already have a GitHub account, you may want to use your GitHub email address here. So you'll get credit on GitHub as well. The global option makes sure that every project in this computer will use the same name and email address. Right, so I have a terminal open right here. You can use hyper on a Mac or it git bash on a PC, and I'm going to start off by adding those configuration variables. So I'll say Git config with the global option and type in user.name. And then I'll type in my name next I'll type in my email address. Once you configure Git, the next step is to prepare the folder that's going to hold the project. For this project, I've got a folder with some files that I like to manage with Git, you can use your own files, but if you want to find out how to get these, make sure you watch the video on working with the exercises. Let's open this up in visual studio code. Now visual studio code has a built-in terminal that you can use. You can go to the terminal menu and select new terminal. This terminal can use different flavors of whatever is installed in the operating system. I have seashell installed here on my Mac, so that will work. If you're on a PC, you may still want to use something like git bash. Let's start off by using the git init command. Notice that the colors in my project changed. That means that this project is now being managed with Git. When you initialize Git, it creates an invisible folder called .git in the project folder. Now this is where Git stores all the information about the project. If you're on Linux, you can take a look at it by doing an ls minus la command. We make this a lot bigger. You'll see that there is now a Git folder. We can even switch to that folder by doing a cd .git command, and then we'll do an ls minus la command again. That's going to show us all the files that are in there. Git us going to write these files for us and take care of keeping track of what the project is doing. I'm going to switch back up to the previous directory with cd.. and let's go ahead and issue a clear command. In order to create an entry that we can go back to. We have to add the files to the staging environment. With the add command, staging is a temporary area that we can store files that we want to commit later on, you use the git add and then specified the file name that you want to move to staging. There are some alternative versions of this command. So for example, the minus minus all flag, we'll add all of the files in the project. The shortcut for that is minus capital A. Most of the longer GitHub commands, usually have a shortcut. That's the first letter of the command. If there's more than one command with the same letter, some of them will be capitalized like this one. There's also an even shorter way to write this. The period is a shortcut and Linux for the current directory. So we can use that. You'll see the shortcut used very often. Let's go ahead and add all these files to staging. Notice that the letter right here change to an a, the last step in the process is to commit the files with the commit command. You always need to include a message for this. It can be a short label so that you can remember what you were doing. You issue a git commit and use the minus M flag and then type in something like first commit or anything else that will make you remember what you were doing. This will tell Git that this is one of the checkpoints that we want to track for our project. That way we can come back to it later. Let's go ahead and clear this out. To verify that Git is keeping track of what we've done. We can issue a git log command. You can see the entry that Git has made, which is being tracked and the .git folder. You've also finished your very first commit. So congratulations, we'll see more about what's happening in the next video.