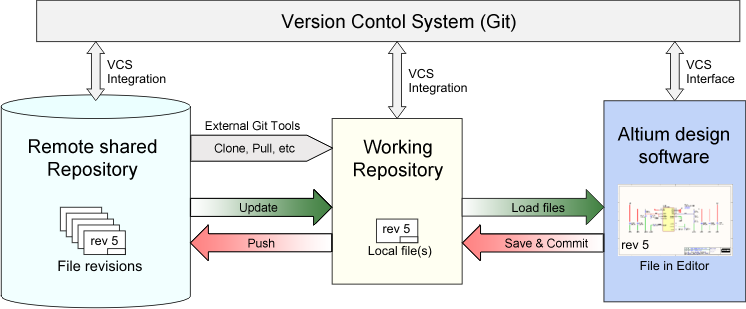
**Introduction to Git and GitHub:**

GitHub, Inc. is an American multinational corporation that provides hosting for software development and version control using Git. It offers the distributed version control and source code management functionality of Git, plus its own features.

At the heart of GitHub is Git, an open source project started by Linux creater. Matthew McCullough, a trainer at GitHub, explains that Git, like other version control systems, manages and stores revisions of projects. Although it’s mostly used for code, McCullough says Git could be used to manage any other type of file, such as Word documents or Final Cut projects. Think of it as a filing system for every draft of a document.

Some of Git’s predecessors, such as CVS and Subversion, have a central “repository” of all the files associated with a project. McCullough explains that when a developer makes changes, those changes are made directly to the central repository. With distributed version control systems like Git, if you want to make a change to a project you copy the whole repository to your own system. You make your changes on your local copy, then you “check in” the changes to the central server. McCullough says this encourages the sharing of more granular changes since you don’t have to connect to the server every time you make a change.

Shows how Git works

**Git makes it easy to contribute to open source projects**

Nearly every open-source project uses GitHub to manage their projects. Using GitHub is free if your project is open source, and it includes a wiki and issue tracker that makes it easy to include more in-depth documentation and get feedback about your project.

If you want to contribute, you just fork (get a copy of) a project, make your changes, and then send the project a pull request using GitHub's web interface. This pull request is your way of telling the project you're ready for them to review your changes.

**Documentation**

By using GitHub, you make it easier to get excellent documentation. Their help section and guides have articles for nearly any topic related to Git that you can think of.

**Integration options**

GitHub can integrate with common platforms such as Amazon and Google Cloud, with services such as Code Climate to track your feedback and can highlight syntax in over 200 different programming languages.

**Track changes in your code across versions**

When multiple people collaborate on a project, it’s hard to keep track of revisions — who changed what, when, and where those files are stored.

GitHub takes care of this problem by keeping track of all the changes that have been pushed to the repository.

Much like using Microsoft Word or Google Drive, you can have a version history of your code so that previous versions are not lost with every iteration. It’s easy to come back to the previous version and contribute your work.

**Showcase your work**

Are you a developer who wishes to attract recruiters? GitHub is the best tool you can rely on for this.

Today, when searching for new recruits for their projects, most companies look at GitHub profiles. If your profile is available, you will have a higher chance of being recruited even if you are not from a great university or college.

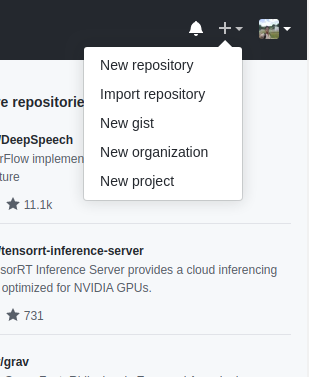
**Working with GitHub projects**

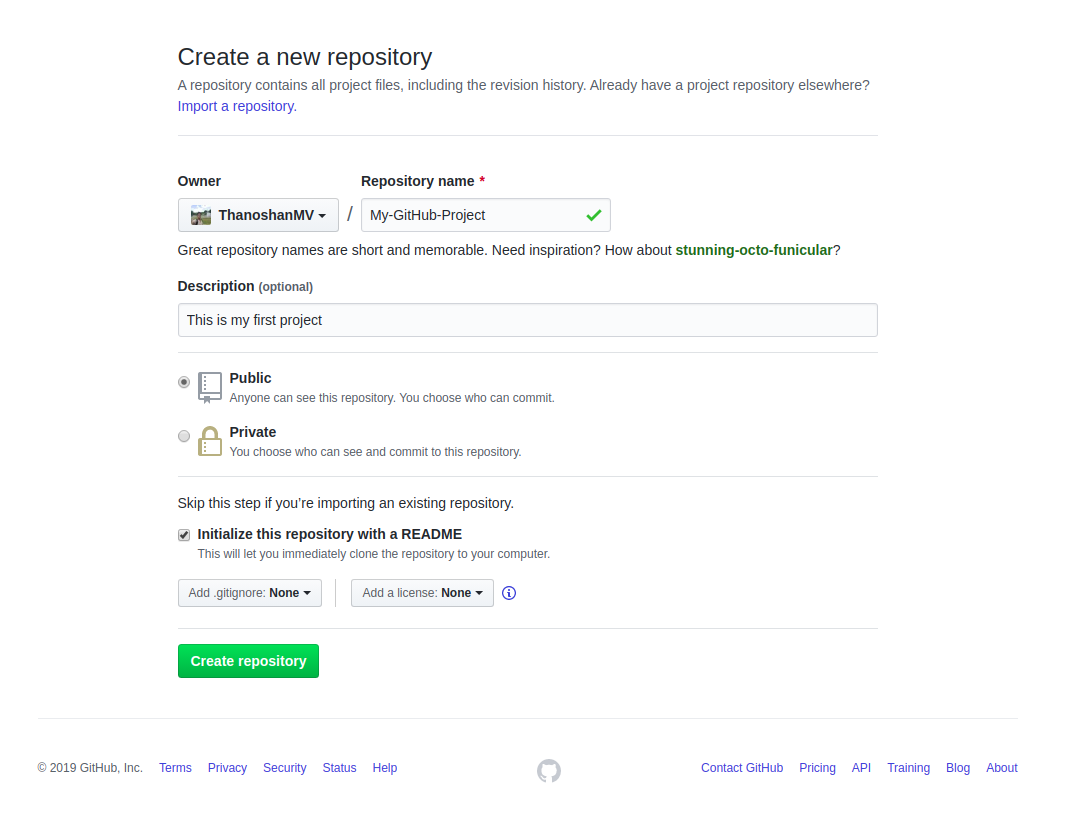
We’ll work with GitHub projects in two ways.

**Type 1: Create the repository, clone it to your PC, and work on it.(Recommended)**

Type 1 involves creating a totally fresh repository on GitHub, cloning it to our computer, working on our project, and pushing it back.

Create a new repository by clicking the “new repository” button on the GitHub web page.

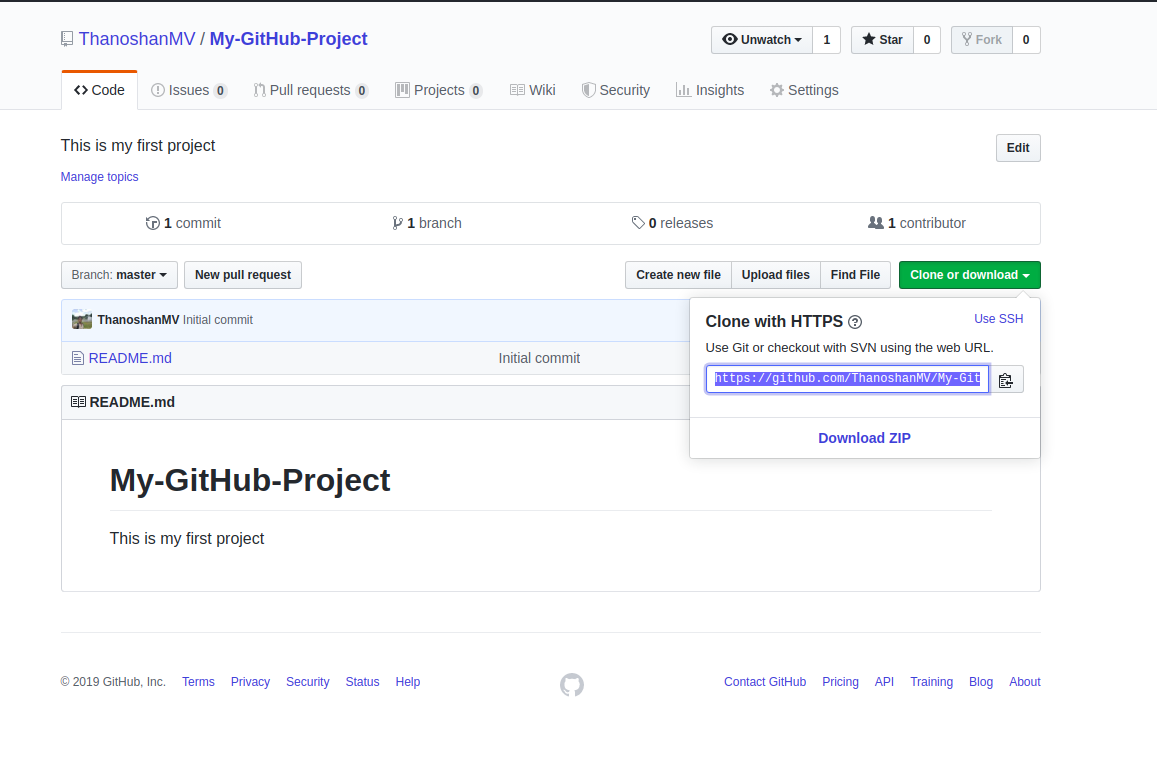


Pick a name for your first repository, add a small description, check the ‘Initialize this repository with a README’ box, and click on the “Create repository” button .

Well done! Your first GitHub repository is created.

Your first mission is to get a copy of the repository on your computer. To do that, you need to **“clone”** the repository on your computer.

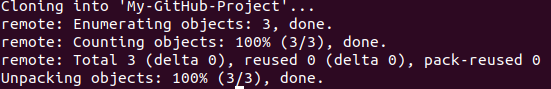
To clone a repository means that you are taking a repository that’s on the server and cloning it to your computer – just like downloading it. On the repository page, you need to get the “HTTPS” address.



Once you have the address of the repository, you need to use your terminal. Use the following command on your terminal. When you’re ready you can enter this:

git clone [HTTPS ADDRESS]

This command will make a local copy of the repository hosted at the given address.



Output message of **“git clone”** command

Now, your repository is on your computer. You need to move in it with the following command.

**cd [NAME OF REPOSITORY]**



As you can see in the above picture, my repository name is “My-GitHub-Project” and this command made me go to that specific directory.

**NOTE:**When you clone, Git will create a repository on your computer. If you want, you can access your project with the computer user interface instead using the above ‘cd’ command on the terminal.

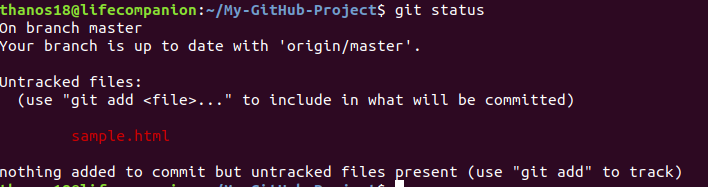
Now, in that folder we can create files, work on them, and save them locally. To save them in a remote place — like GitHub – we have done a process called a “commit”. To do this, get back to your terminal. If you closed it, like I previously stated, use the ‘cd’ command.

cd [NAME OF REPOSITORY]

Now, in the terminal, you are in your repository directory. There are 4 steps in a commit: ‘status’ , ‘add’ , ‘commit’ and ‘push’. All the following steps must be performed within your project. Let's go through them one by one.

1. **“status”:** The first thing you need to do is to check the files you have modified. To do this, you can type the following command to make a list of changes appear.

git status



1. “add”: With the help of the change list, you can add all files you want to upload with the following command,

**git add [FILENAME] [FILENAME] [...]**

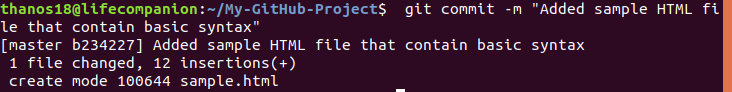
In our case, we will add a simple HTML file.

git add sample.html



3**. “commit”:** Now that we have added the files of our choice, we need to write a message to explain what we have done. This message may be useful later if we want to check the change history. Here is an example of what we can put in our case.

**git commit -m** "Added sample HTML file that contain basic syntax"



4. **“push”:** Now we can put our work on GitHub. To do that we have to ‘push’ our files to Remote. Remote is a duplicate instance of our repository that lives somewhere else on a remote server. To do this, we must know the remote’s name (Mostly remote is named origin). To figure out that name, type the following command.

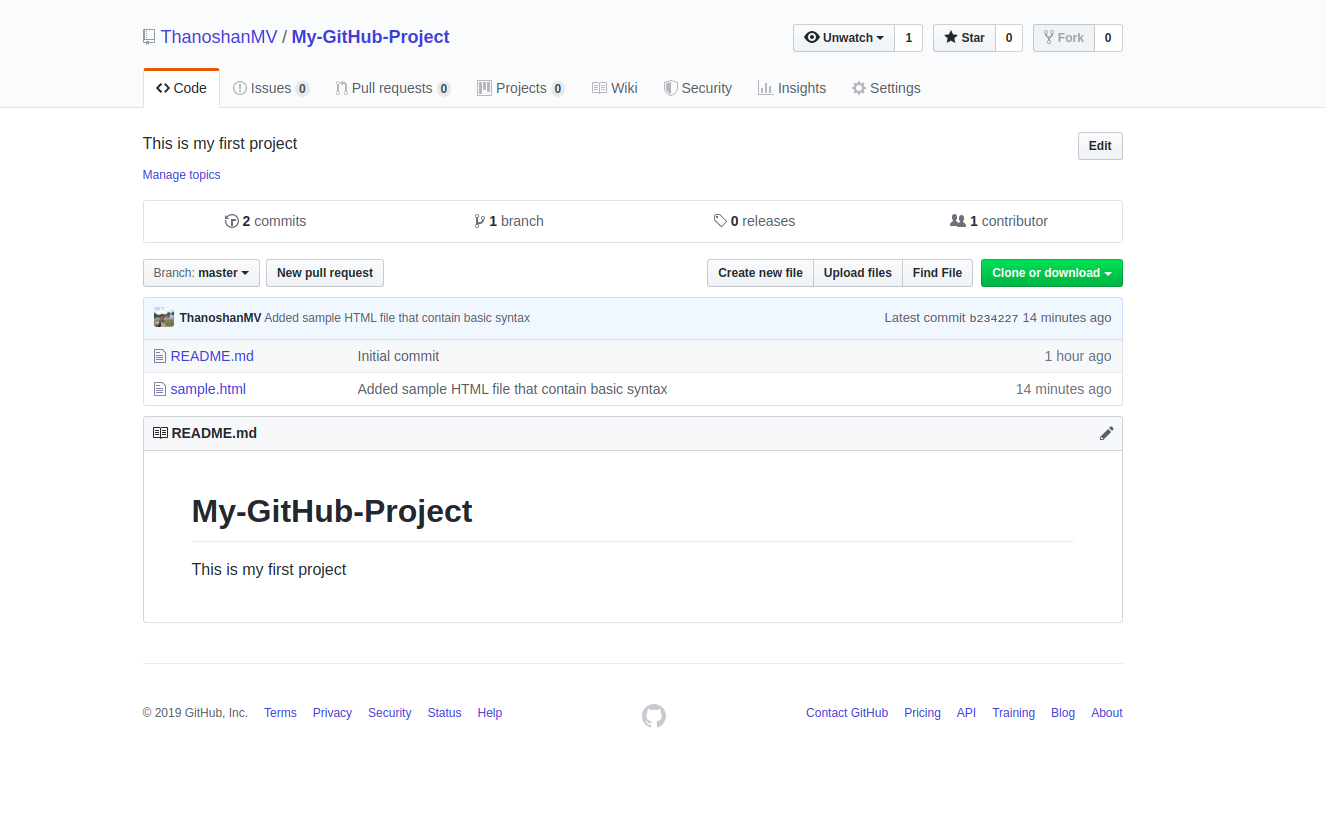
**git remote**



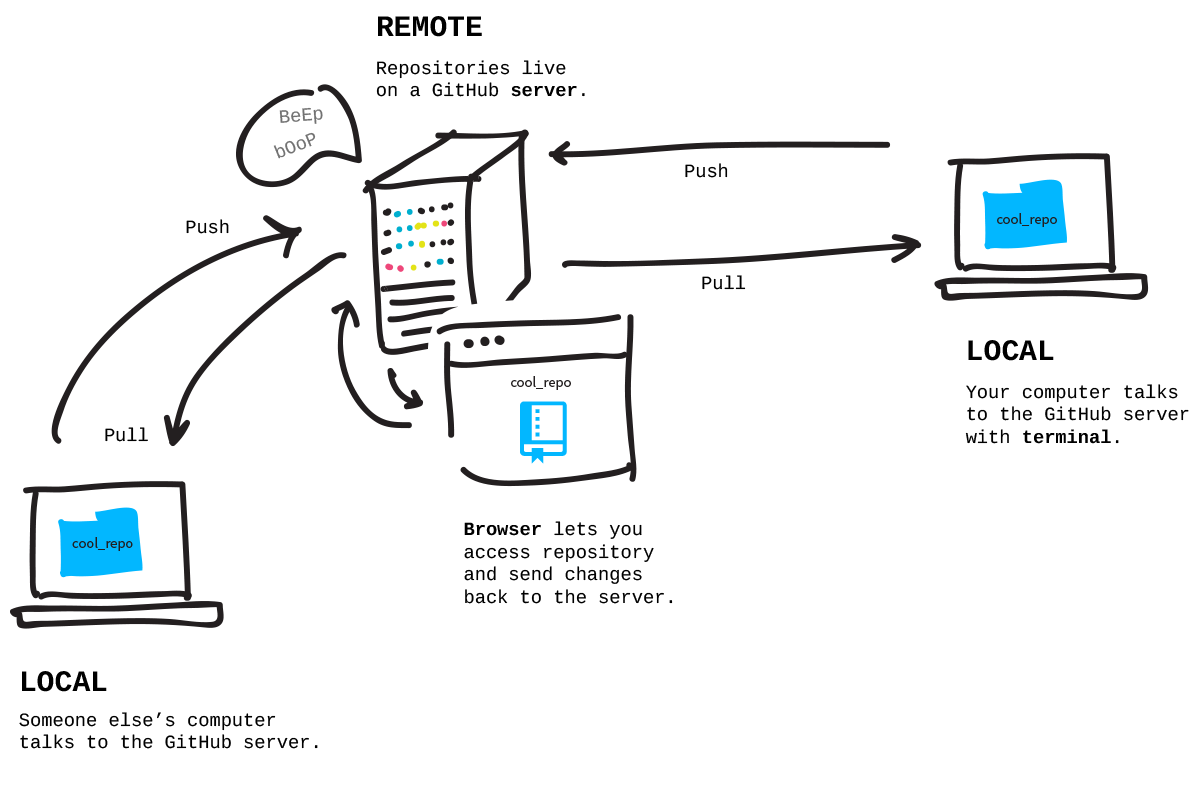
As you can see in the above image, it says that our remote’s name is origin. Now we can safely ‘push’ our work by the following command.

**git push origin master**

Now, if we go to our repository on the GitHub web page, we can see the sample.html file that we’ve pushed to remote — GitHub!



**NOTE**: Sometimes when you’re using Git commands in the terminal, it can lead you to the VIM text editor (a CLI based text-editor). So to get rid of it, you have to type :q and ENTER.

Describes how pull & push work

Pulling is the act of receiving from GitHub.

Pushing is the act of sending to GitHub.

**Type 2: Work on your project locally then create the repository on GitHub and push it to remote.**

Type 2 lets you make a fresh repository from an existing folder on our computer and send that to GitHub. In a lot of cases you might have already made something on your computer that you want to suddenly turn into a repository on GitHub.

I will explain this to you with a Survey form web project that I made earlier that wasn’t added to GitHub.

As I already mentioned, when executing any Git commands, we have to make sure that we are in the correct directory in the terminal.



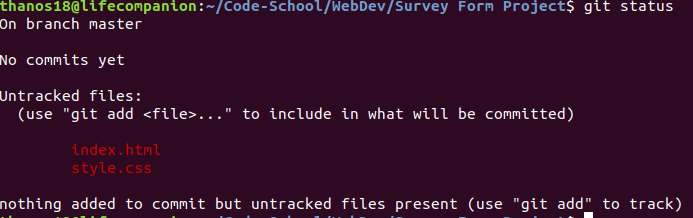
By default, any directory on our computer is not a Git repository – but we can turn it into a Git repository by executing the following command in the terminal.

**git init**



After converting our directory to a Git repository, the first thing we need to do is to check the files we have by using the following command.

**git status**



So there are two files in that directory that we need to “add” to our Repo.

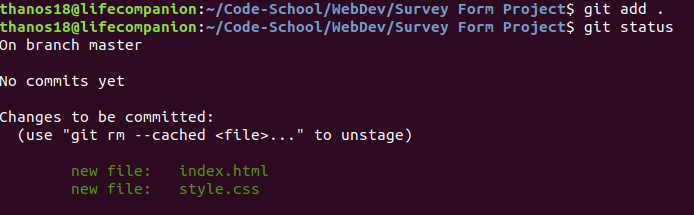
**git add** [FILENAME] [FILENAME] [...]

**NOTE**: To “add” all of the files in our Repository we can use the following command:

**git add .**

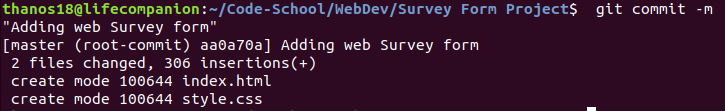
After the staging area (the add process) is complete, we can check whether the files are successfully added or not by executing the git status

If those particular files are in green like the below picture, you’ve done your work!



Then we have to “commit” with a description in it.

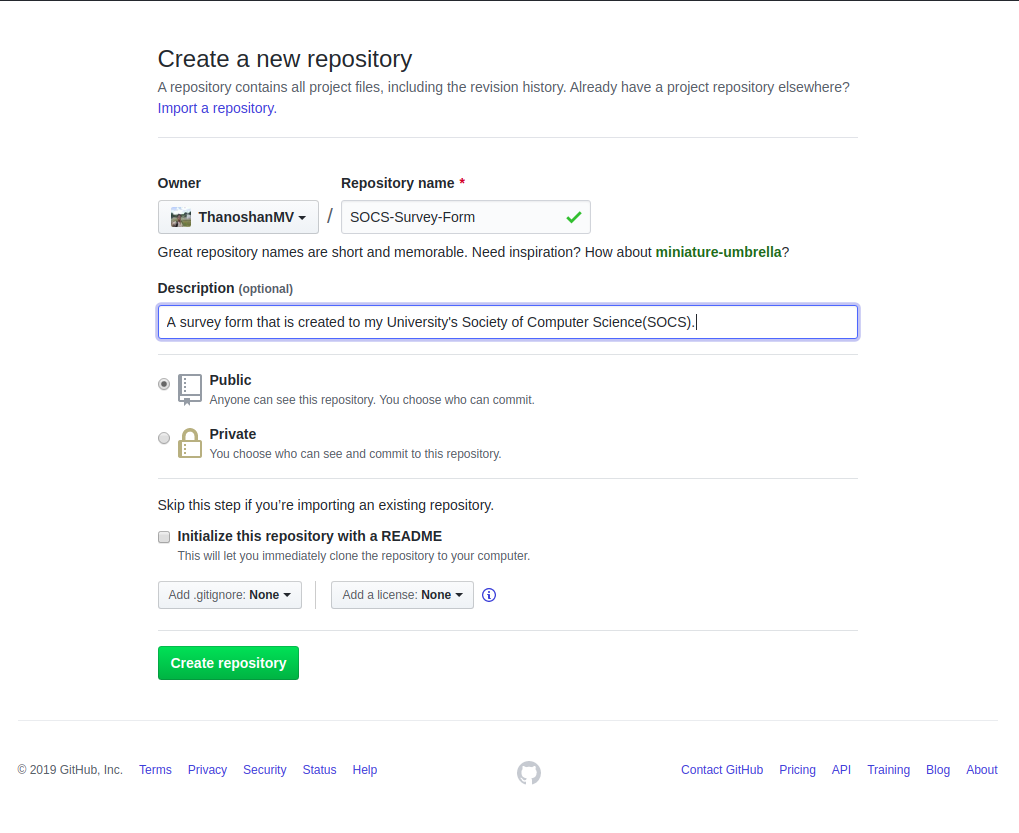
**git commit -m** "Adding web Survey form"



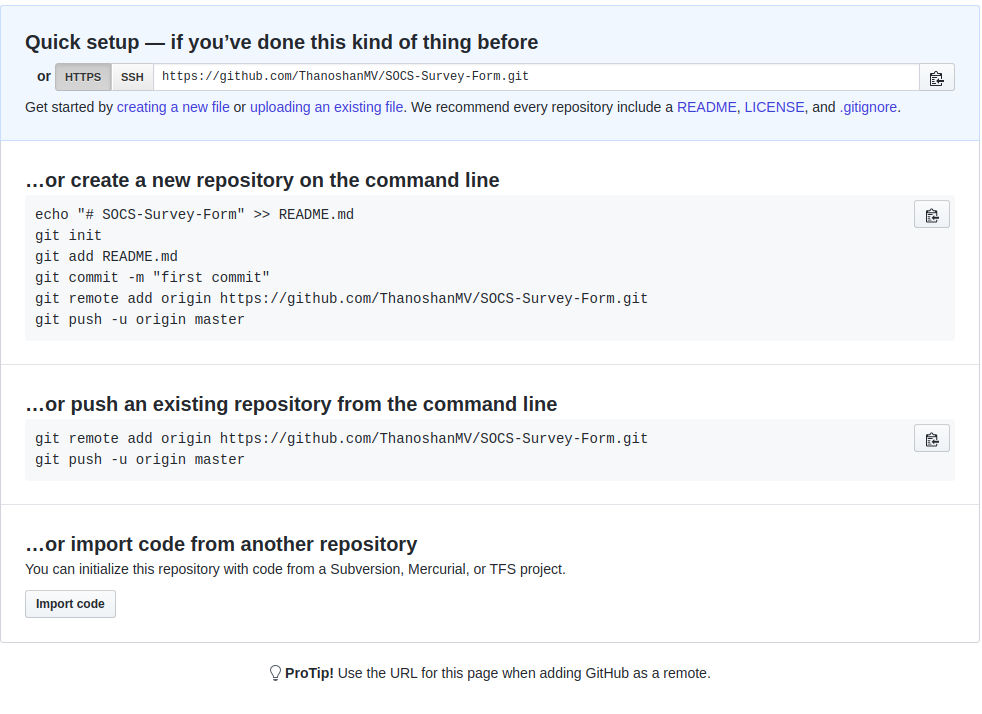
If my repository started on GitHub and I brought it down to my computer, a remote is already going to be attached to it (Type 1). But if I’m starting my repository on my computer, it doesn’t have a remote associated with it, so I need to add that remote (Type 2).

So to add that remote, we have to go to GitHub first. Create a new repository and name it whatever you want to store it in GitHub. Then click the “Create repository” button.

**NOTE**: In Type 2, Please do not initialize the repository with a README file when creating a new repository on the GitHub web page.



After clicking the “Create repository” button you will find the below image as a web page.



Copy the HTTPS address. Now we’ll create the remote for our repository.

**git remote add origin [HTTPS ADDRESS]**

After executing this command, we can check whether we have successfully added the remote or not by the following command

**git remote**

And if it outputs “origin” you have added the remote to your project.

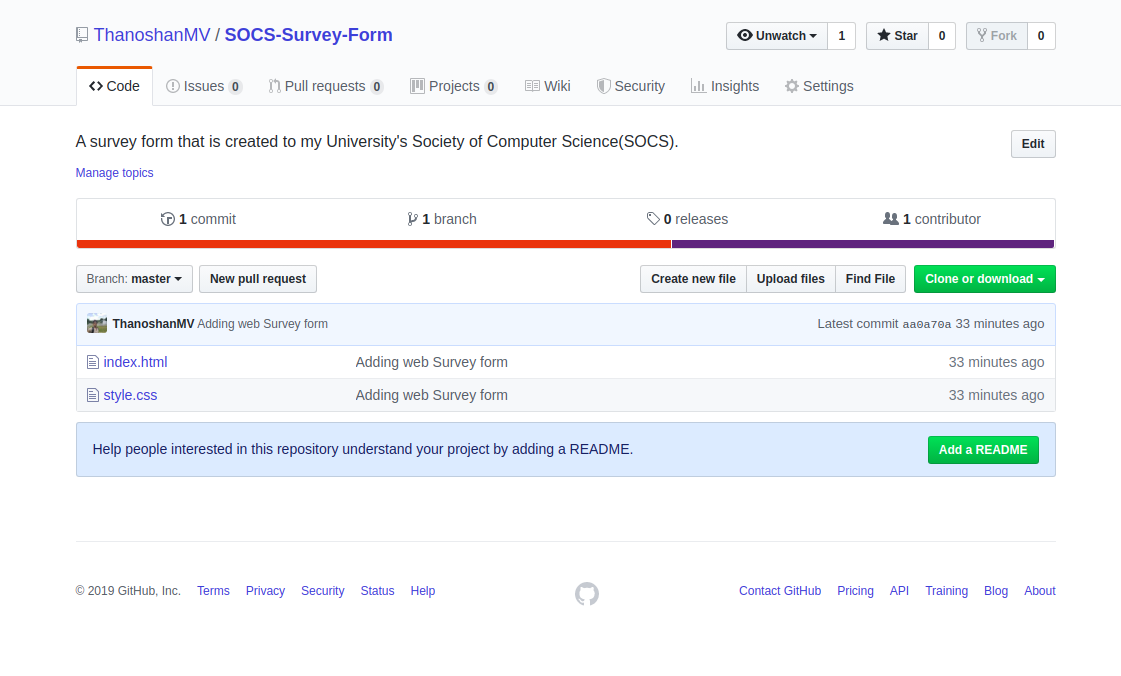
**NOTE**: Just remember we can state any name for the remote by changing the name “origin”. For example:

**git remote add** [REMOTE NAME] [HTTPS ADDRESS]

Now, we can push our project to GitHub without any problems!

**git push origin master**

After completing these steps one by one, if you go to GitHub you can find your repository with the files!



**References:**

<https://www.freecodecamp.org/news/the-beginners-guide-to-git-github/>

<https://en.wikipedia.org/wiki/GitHub>

<https://g.co/kgs/FRwg9F>

<https://techcrunch.com/2012/07/14/what-exactly-is-github-anyway/>