**INSTALLING GIT AND CREATING A KEY**

1. Installation of Git

First we will download the Git application, this will allow us to connect the files and folders from our computer to the repository being kept on Github. We will talk more about repositories and their uses another time, but the long and short of it is that the version being stored on the repository is the official “master” version, which allows the rest of us to play around with the code to our heart’s extent without having to worry about ruining it for everyone else! (or at least you need to jump through a lot of hoops to ruin it for everyone else.)

So to download the git software to allow your local files to “speak” with the repository you will need to go to <https://git-scm.com/downloads> and download the installer for the appropriate operating system. When installing there are a lot of options, I usually use the defaults but you are free to decide on all of them on your own. The one I will speak of is the text processor that git defaults to, the standard choice is Vim which – while I’ve gotten used to it – is finicky. I think Notepad++ is probably a better option but I am not sure. The install will be pretty quick, and then you are ready to \*Git\* going!

Graphical user interface, application, website

Description automatically generated

.

1. GITHUB VERIFICATION KEYS

I haven’t done this on this machine yet, and is one of the biggest roadblocks to using Github so I’ll just type this up as I do it for myself! The general concept of this is that you need to get set up with a SSH key to allow the Git program on your computer and Github on the internet to be able to communicate with each other and to know that it is you who is trying to download, edit, or upload the code to the repository. So the first thing to do is to create an account on Github, then in the top right corner there will be a drop-down menu with your icon, under which you can select settings. In order to get the key process going we will select the “SSH and GPG keys” option under “Access” in our account menu.

A screenshot of a computer

Description automatically generated with low confidence

Figure : This is where you go in your account settings to generate and add an SSH key

The good thing here is that Github provides us with a “generating SSH keys” link right on this page! I am going to walk through this so I’ll trust you call can follow those directions but if I feel its necessary I’ll post any additional thoughts.

* After starting the keygen in Git Bash it will ask you where to save the key and to set up a passphrase, I’ve always found that the passphrase is more of a burden than a help and if you just hit enter and have a null passphrase it doesn’t even both you with it.
* Note where it puts the SSH file, you will need to access it shortly
* I guess you can also install Github Desktop … I got no experience with that but have at you.
* The guide is really concise and straightforward about how you add the SSH key to your Github

Congratulations! Now Github knows that you are you!

1. CLONING A REPOSITORY

From here on we are going to be edging up against how to properly use Git and Github together, which is probably best left as a different document for some “best practices” and advice to do proper Configuration Management. It is just kind of hard to do right now since we don’t really have any files to work with now. But at the very least we can set up the directory that we will be under configuration control.

First, we are going to direct Git Bash to that directory and then have Github download the current items from the repository there. I’ll note here that Git Bash operates like a command line Linux system, so navigating files can be pretty hard to get used to, but luckily you will only have to do it a handful of time, the actual use of files is just like your normal OS. For example, here is me navigating to my directory. The first thing to note is the command “cd” which stands for “current directory”, which acts similarly to putting a file location in the address bar of your file explorer. Text

Description automatically generated

Figure :There are a few things of interest to see here. First the orange text on each line tells us where the Operating System is currently located. You can alternatively find that out by entering pwd on the command line. Secondly, and most importantly, there is the light blue text that reads “(master)” this indicates to us that we are in a directory that is under Git management and the current we are looking at is the master branch, you won’t see that immediately but once the repository is downloaded to your machine it should show. (Ed. Note: I guess in recent days the default branch name has been changed to “main”)

Once you have directed Git Bash to the right directory that you want the repository to be located (just note it will create a directory). Now it is time to clone the repository, which basically means “download all the files in the repository and make sure Git knows how to keep track of them”. To do this we’ll go to the repository’s page on GitHub, there will be a button with the word “< > Code” which we can click on to create a drop-down menu. The default choice is to clone using SSH, which provides us with a line of code and a handy copy button! Now we go back to the Git Bash window and type in “git clone ” and then copy those lines from the browser (in Linux paste is middle click.) See below for an image of my cloning. Text

Description automatically generated

Figure : I cloned the repository! All is good and ready now! We'll talk about what to do with this another time.

Okay, now we’ve got our machines set up with Git and connected it with GitHub to have a shared repository. Trust me, the startup and learning curve seems a little unnecessary at first, but in the long run this stuff can save a lot of headaches and keep us more organized and working together! … There’s also the added benefit that you could quickly and easily share your code with people (employers or collaborators etc.) in the future.