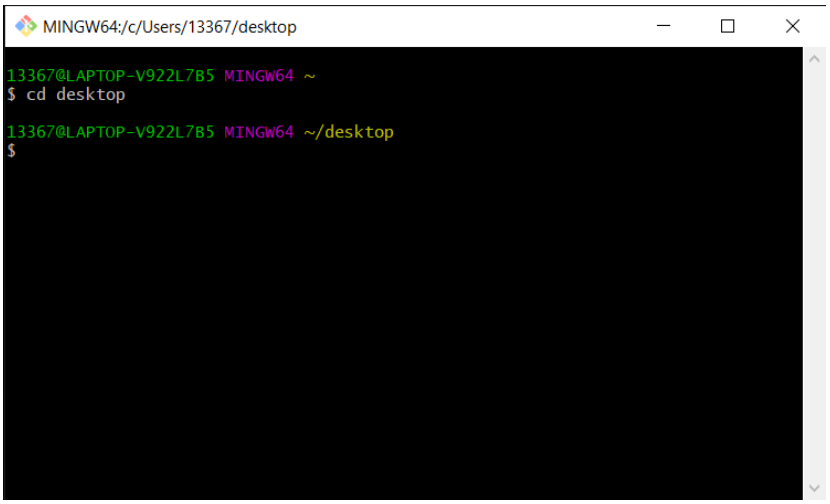
Git Bash Tutorial

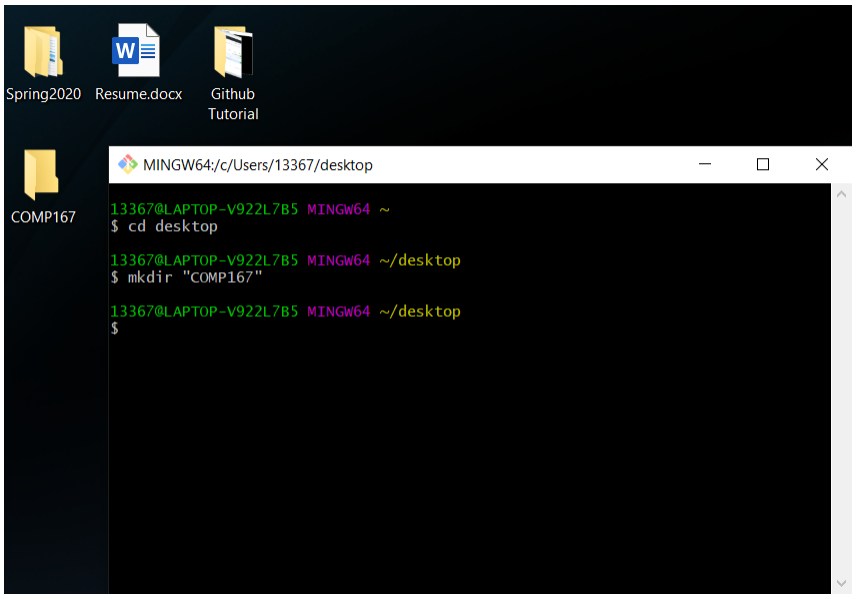
The following steps outline how to push your lab work to github using git bash.

Step 1: Create a place to store your work and navigate to it.

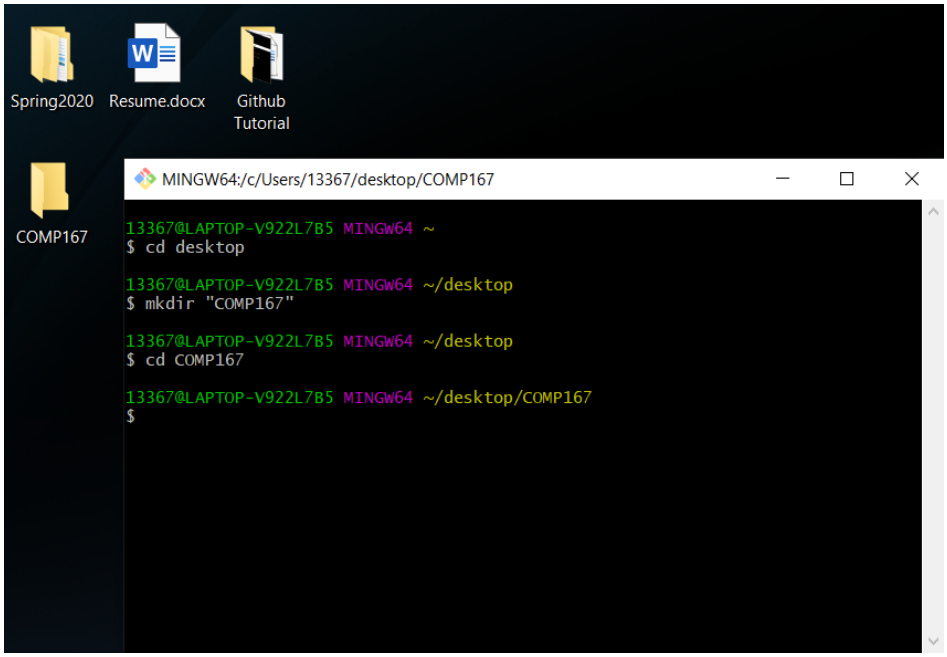
Navigate to where you want to store the folder that will hold your lab work. Do this by typing “cd [FILENAME]” without the quotations. I’ll store mine on my desktop. To navigate to my desktop, I’ll type “cd desktop” into git bash.



See the “**~/desktop**” in yellow? This means we are currently operating within the desktop and can now make our file here. To make the file, type: “mkdir “[FILENAME]”” with the file name in quotations. I suggest naming the file COMP167. I’ll do that now.



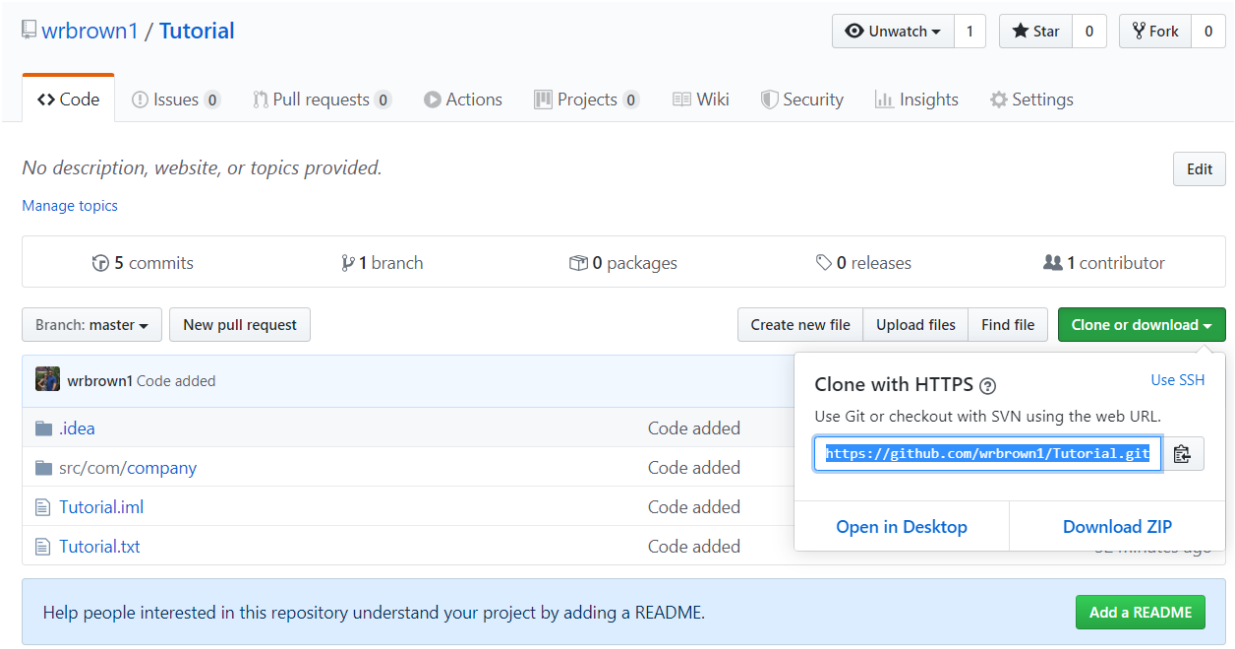
Notice the COMP167 folder to the left that we just made (“mkdir” stands for “make directory”). Now, we need to navigate into the newly made directory the same way we navigated to the desktop.



Again, notice the "**~/desktop/COMP167**". We are now operating in the folder named COMP167.

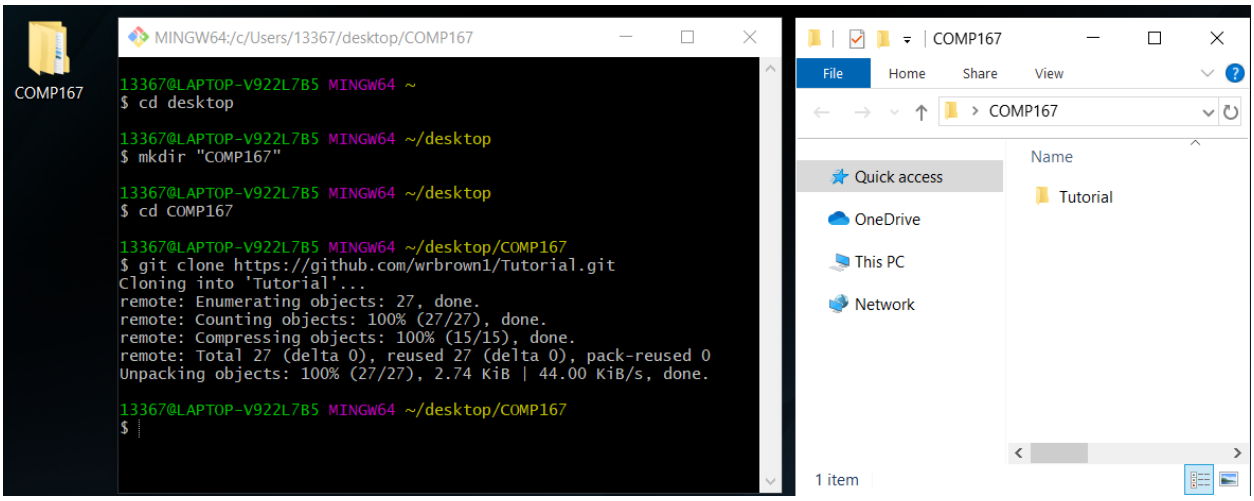
Step 2: Clone the lab from github into your new folder.

Find your lab on blackboard and notice the "Clone or Download" button near the top right.



For the purpose of this tutorial, I created my own repository. Yours should be called something like "lab-1-[YOURNAME]."

Copy the link that is shown after clicking the “Clone or Download” button. Then, in git bash, type “git clone [LINK].” You’ll need to paste it by using right click -> paste. Ctrl+V won’t work in git bash.



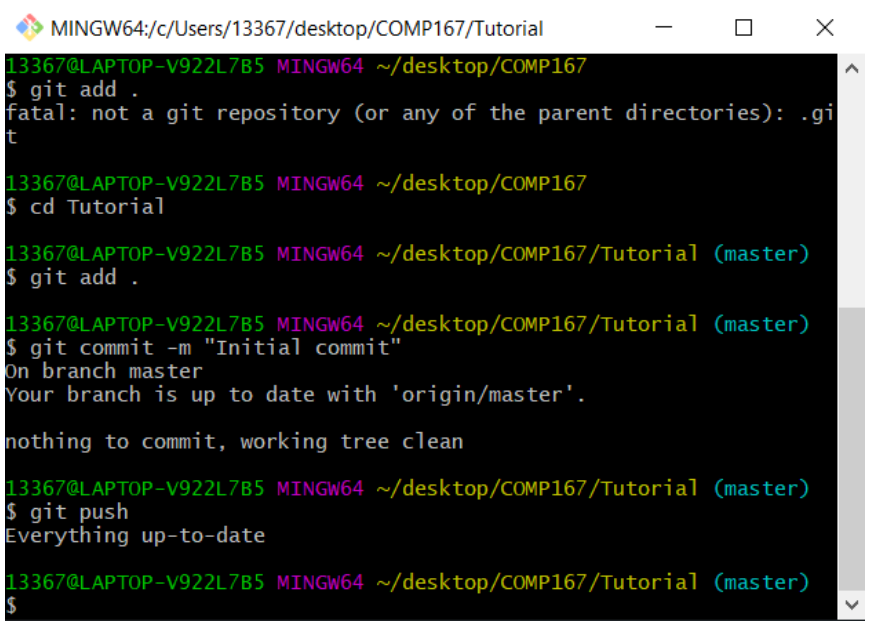
Now those files have been cloned and put into the folder we made. Again, yours will be called “lab-1-[YOURNAME].” Be sure to navigate into it using “cd lab-1-[YOURNAME].”

Step 3: Pushing your work to github.

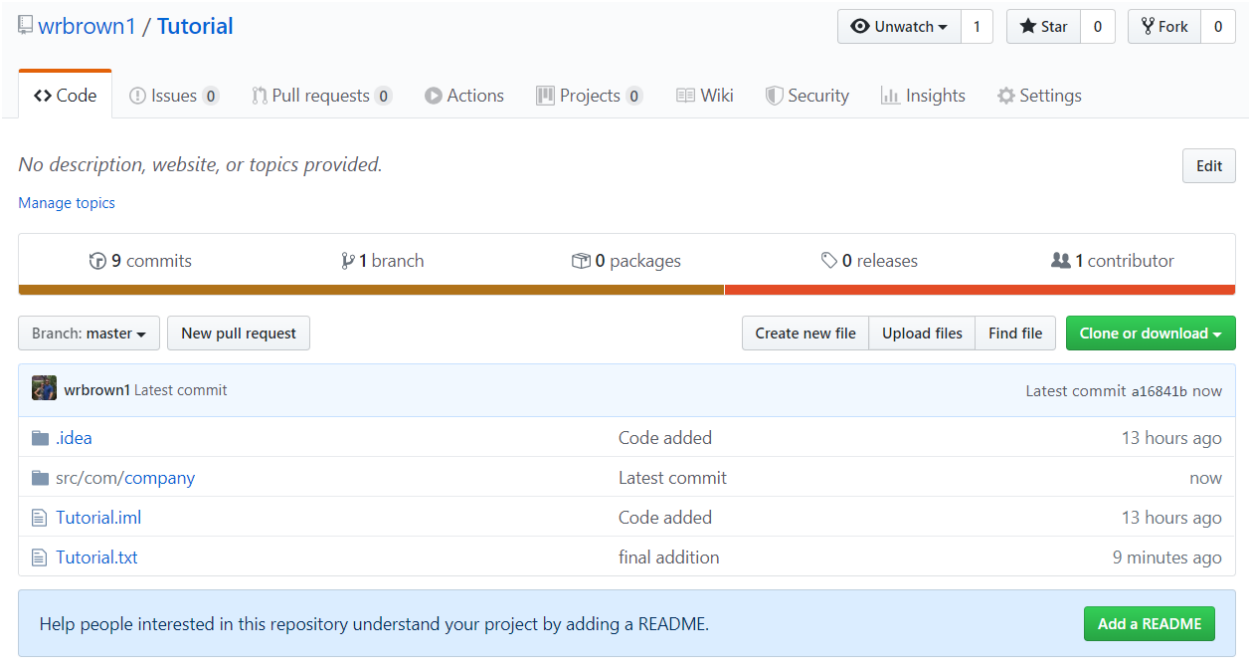
We’ll be using three commands in git bash to frequently push our code to github. Those commands are:

git add . **git commit -m “[COMMENT]”** **git push**

It’s important to make a relevant comment when you commit your code to the repository so that others know how you modified the code on each commit. Something such as “fixed a null pointer” or “improved readability” or “Initial commit.”



Now we need to check github to make sure we pushed properly.



Notice github displays the time of the commits. If we pushed properly, we should have a very recent commit. Mine says “now” because it was very recent. Keep in mind that if you push a file with no change github WILL NOT update. Try making a small arbitrary change to your code before pushing to make sure github has something to update.

You’re done! Any time you make a change to your code simply repeat step 3.