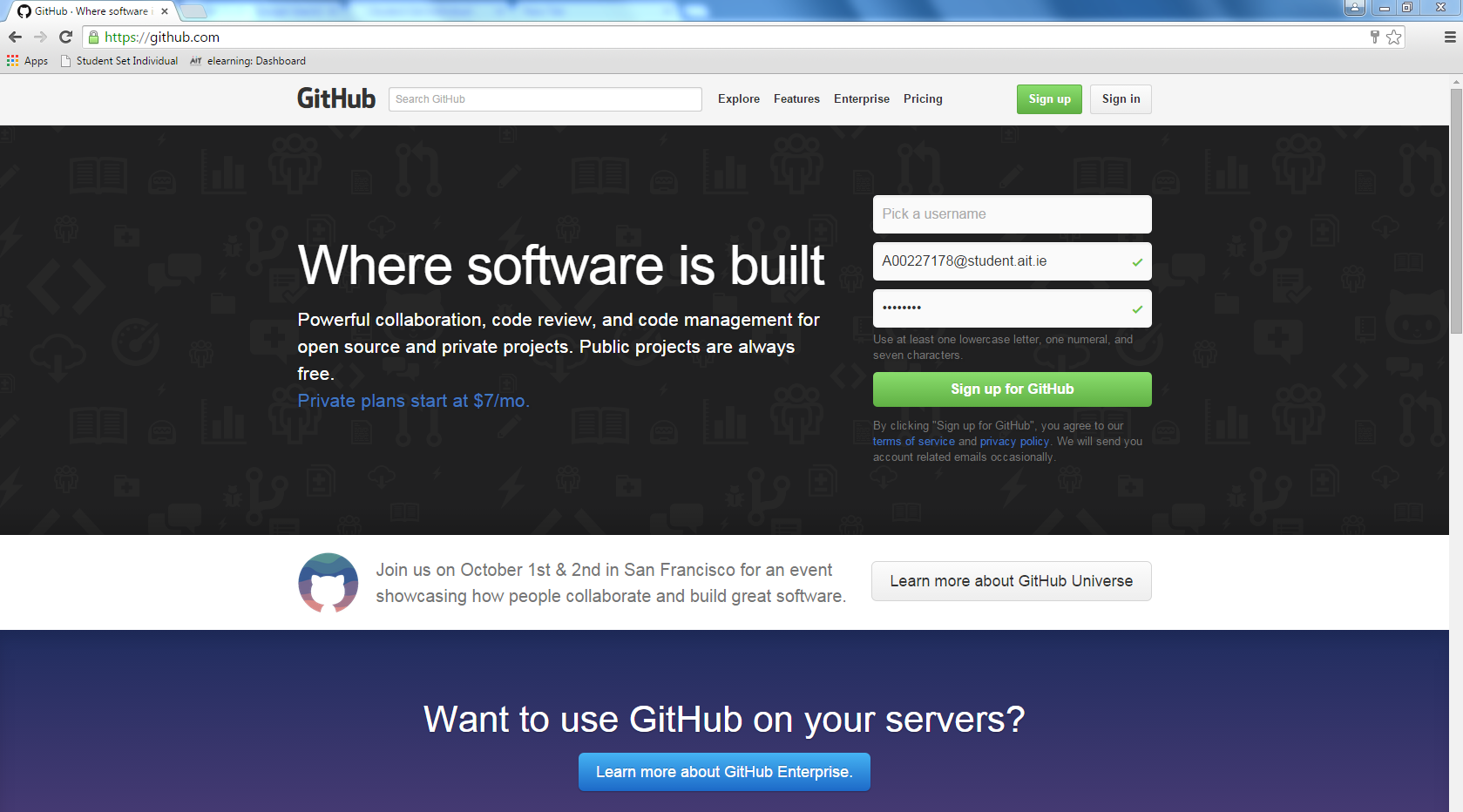
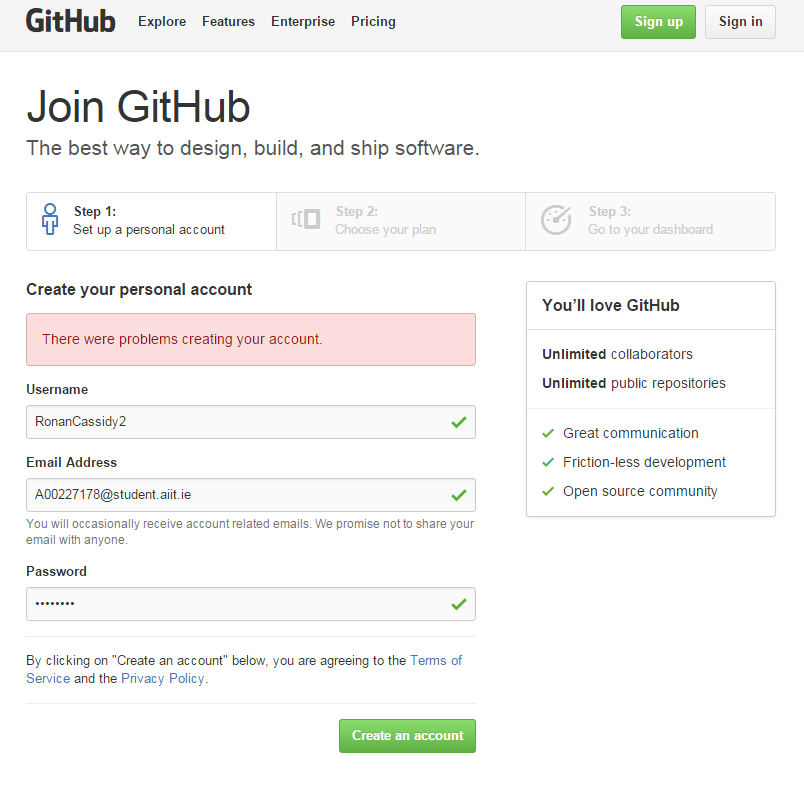
# How to set up Github

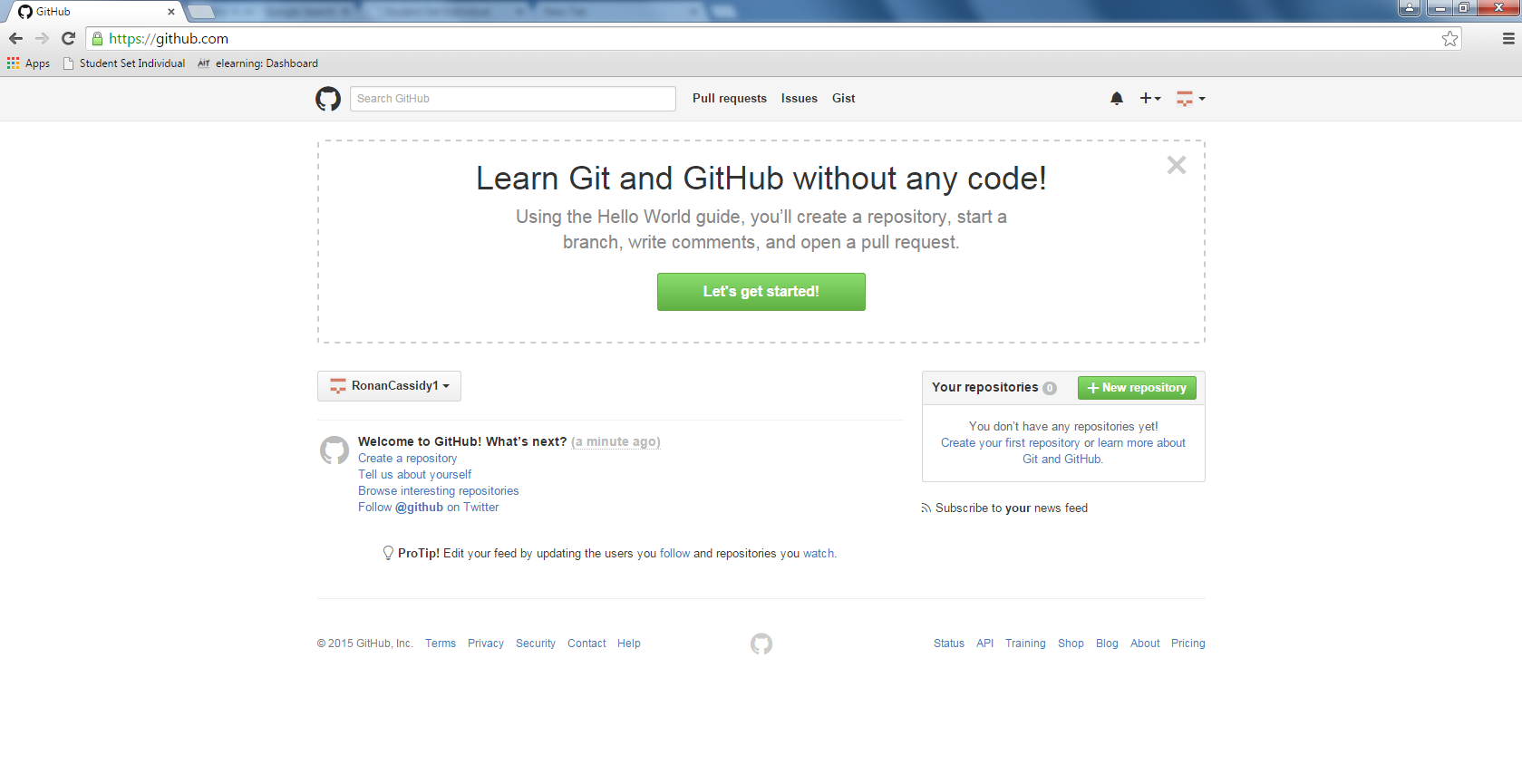
## Step 1: Creating Your Account

Before doing anything you must set up your Github account on the Github website at [www.github.com](http://www.github.com) . This will lead you to this page

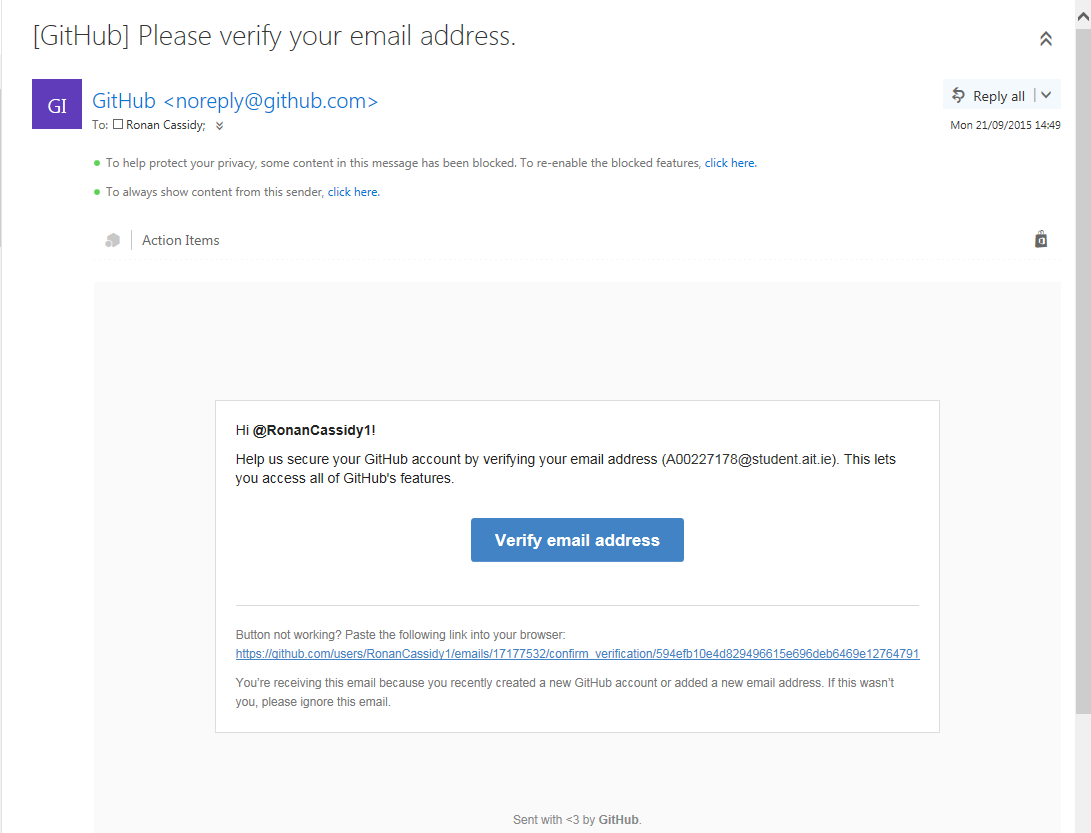
Now just press the button that says “Sign up for Github” or the “Sign up” button at the top of the page and that will bring you to this page.



As you can see you will be asked for your details and once these are entered in correctly just press the “Create an account” button.

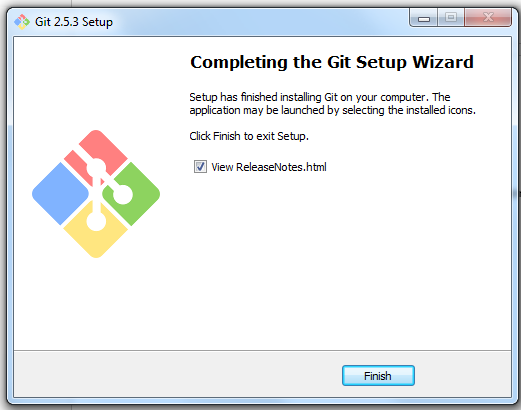
Next if all is correct you will be directed to this page

You should then go to your email and Github will have sent you a verification email.

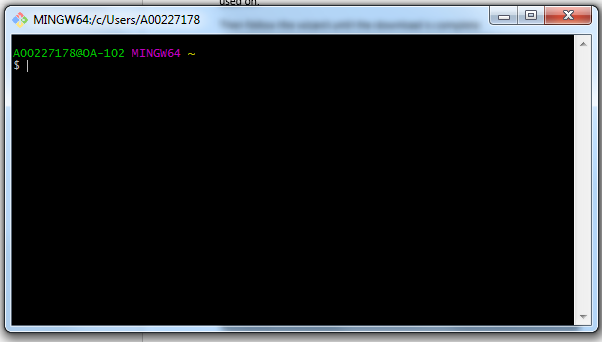


Now that you’re verified the setting up of your Github account is complete.

## Step 2: Installing GitBash

Now go to this website <https://git-scm.com/downloads> and download the terminal for github to be used on. Then follow the wizard until the download is complete

Once this complete you should launch “GitBash” from your start menu. You should see something like this



## Step 3: Setting up GitBash

Now that your gitbash is downloaded you can now begin setting it up. You first need to set up the username that you created your account with and link it to your guitbash this is done like this.

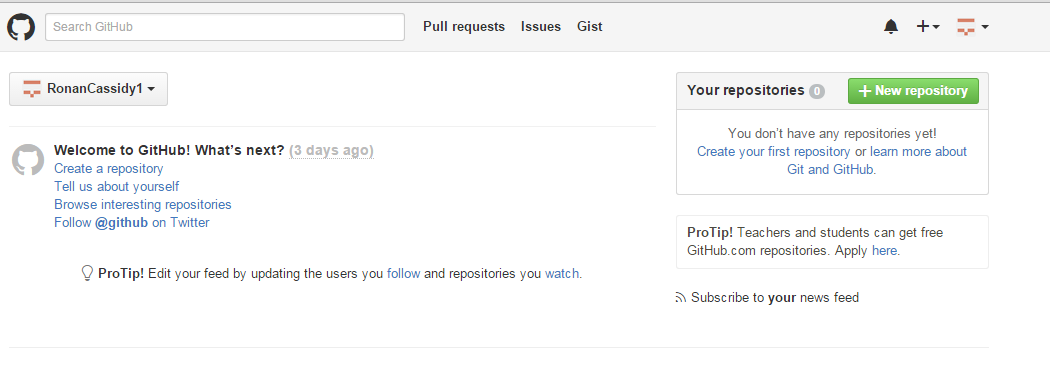
Press enter to execute the statement. Your username should replace “YOUR NAME” in this statement.

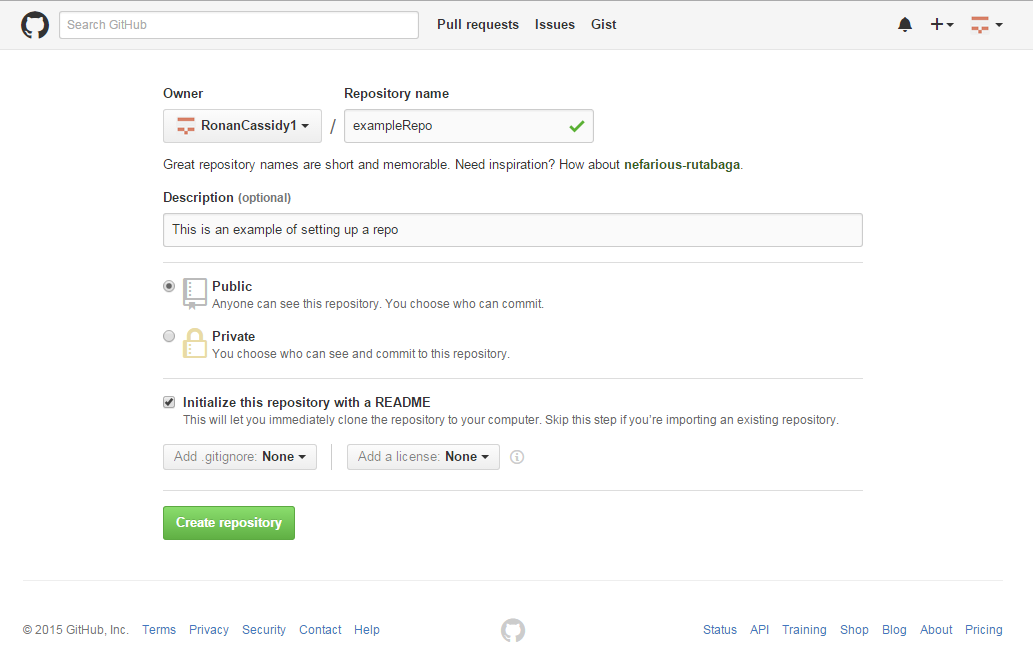
Next you will have to tell git your email this is done like so.

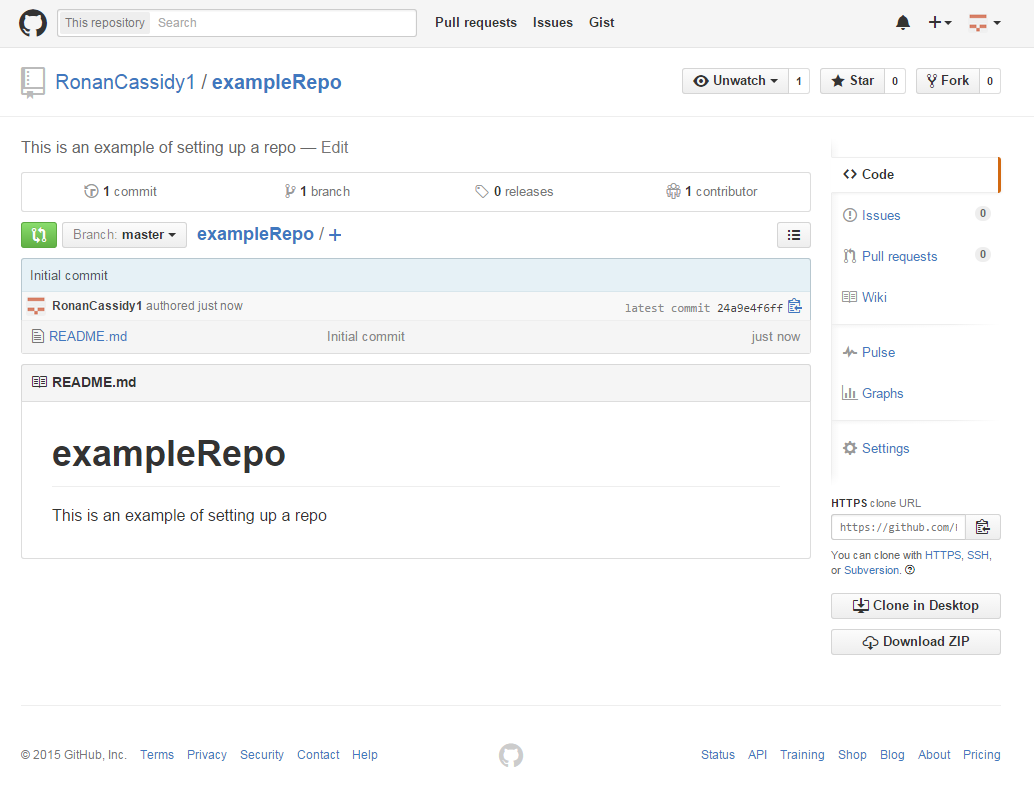


Again where you see “YOUR EMAIL ADDRESS” this is where you put your email that is associated to your account.

## Step 4: Creating Repo

First thing when creating a repo you must go back to the website and log in. Once in you will see this screen

By clicking on the “Create a repository” button or the “+New repository” button this will begin the process of creating a repo. Clicking on one of those will bring you to this page.

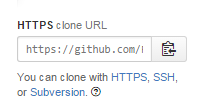
Here you will provide the Repo name, description and other details. Once these are finalized you can click the “Create repository” button to finish.

Now that is complete you have your repo set up and ready to work with.

## Step 5: Cloning repos

Next you will need to clone the repo onto your desktop this is easily done like so. Go back onto the website once again and go to your new repo. Once here there will be two different ways to get the correct URL for cloning. First is the URL bar 

Secondly is along the right side of the page

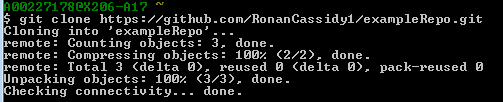
By clicking the button on the right of the link it will copy it for you so you can paste it into gitbash

**Quick Tip:** Ctrl + V doesn’t work in GitBash but right clicking the top of the program and going to “Edit -> Paste” it will do the same.

Now you have the link you need to type git clone the your link like so

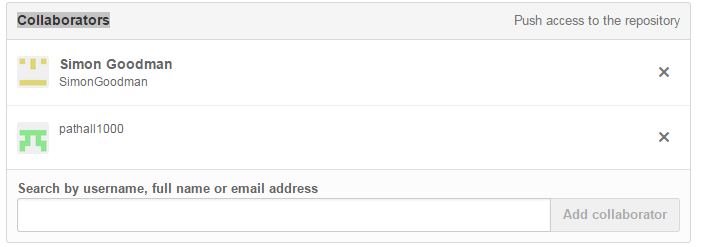


If successful you should read this



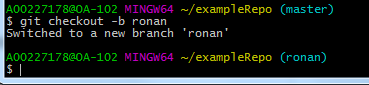
Congrats you have successfully cloned your repo. If you go your C: drive and your personal user folder your repo should be present with the “README” file there. This is where your project will lie and alterations here will have to be pushed to appear online.

## Step 6: Adding people to repo

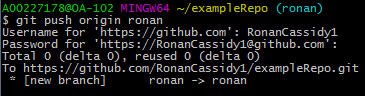
To add people to your repo you need to go to the website log in and go to your repo. You will need either the persons email or username or both if you can just in case one doesn’t work. When in your project go to the **Settings** option on the right hand side of the page. Then click on **Collaborators**. 

Here you can add them by username or email.

## Step 7: Creating branches

For projects with multiple people on it the best practice is to break up each person to have their own branch. This means the master branch stays clean and someone can continually push their own code to their own branch without having to merge with other peoples. Saves the project from getting messy. To do this you need to open the Gitbash window and type the following.

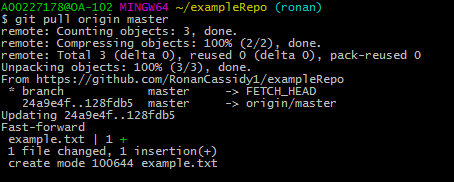
Don’t worry the name of the branch does not have to be “ronan”. As you can see once you checkout it will switch you to your branch as of now this is only a local change in order to properly create this branch you will need to type “git push origin ronan” substitute ronan for the name of your branch. You should also be asked for your username and password. If it worked you should see this.



## Step 8: Pulling, Committing and Pulling

### Pulling

Now that your repo and branches are set up. Since you cloned from the master there may have been some changes to the repo you may need so to do this you need to execute the following code in GitBash. “git pull origin master”

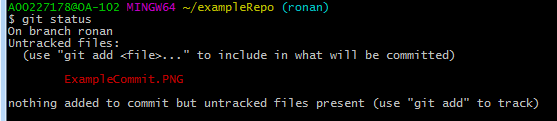


When this is executed couple of lines will appear these are telling you there was new stuff on the master that you did not have and now you have it so you are up to date with the master branch.

**Quick Tip:** Typing in “git status” will tell you what your status is up to date or if you need to commit etc

### Committing

So our branch is now up to date time to make some changes. Add a text file or a picture or anything to the folder that you cloned.



Here I show using “git status” that I have uncommitted changes. Yours should look somewhat similar. Okay now you what your changes to be up on github. There are 3 things you must do first you need to add changes then commit the changes with a message finally you need to push your changes to your branch. So to add simply do this



This adds all changes. You can add specific files if you want using “git add ExampleCommit.PNG” Next do this



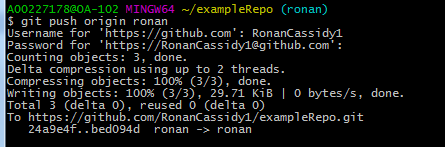
This commits the changes and you add a commit message inside the brackets

### Pushing

Finally you need to push this to your branch. This is done by saying



It should ask for your username and password again but if all of this is successful then it should look like this



If you now go onto the website you will see the changes on your branch and also there will be a Compare & Pull Request on the master branch. This compares both the master and the branch you committed too and checks if they are okay to merge.

## Helpful Hints and Tips:

### What files are in folder using GitBash

When in a directory or folder in GitBash a helpful tool is just typing “ls” this will show you what is in the current directory you are in. This can be helpful when navigating through the directory’s.

### Using TAB to autofill files or directory’s

When typing in GitBash you don’t have to type out all of the file or directory. If you type out the first couple of letters then press TAB it will autofill for you. So if you’re moving through directory’s this can be a time saver.