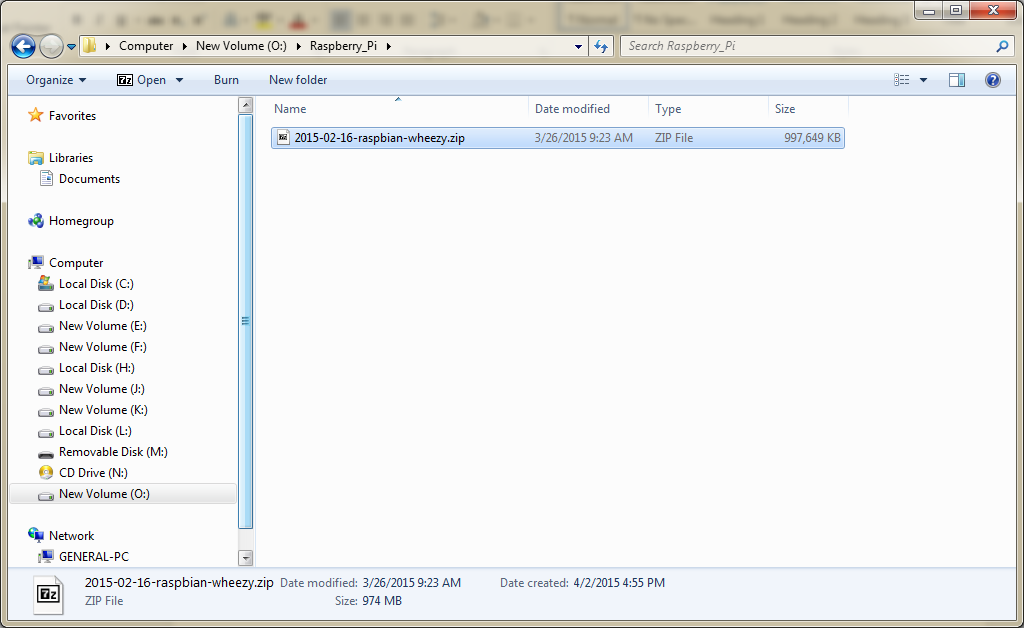
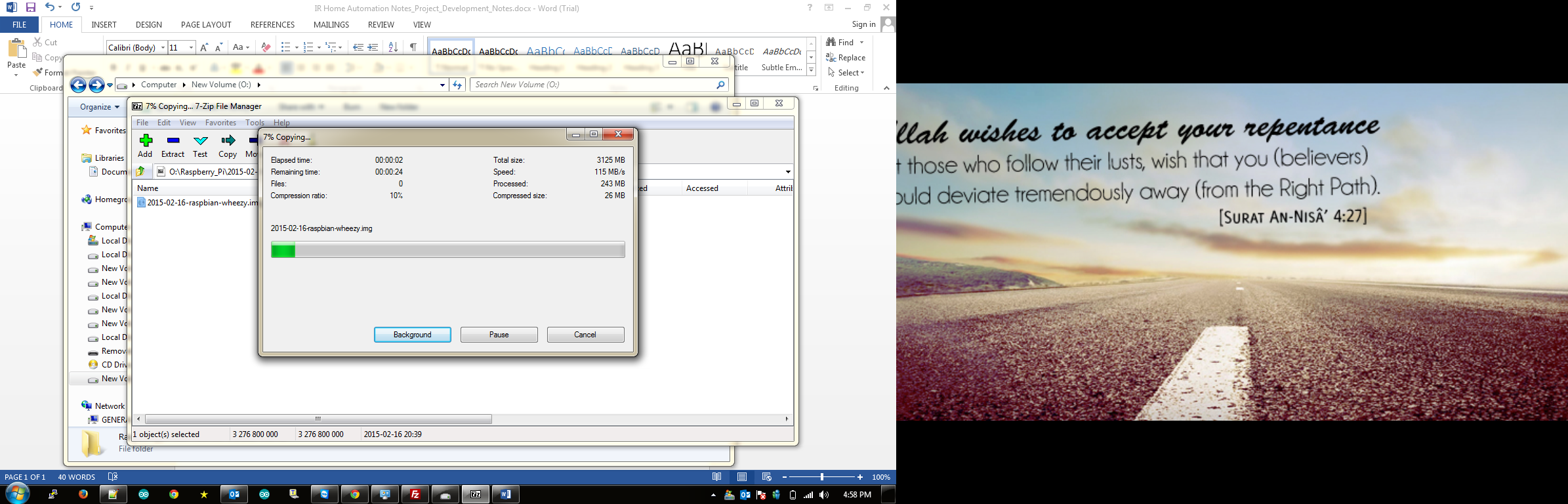
# Getting started with Raspberry Pi (and WebIoPi framework)

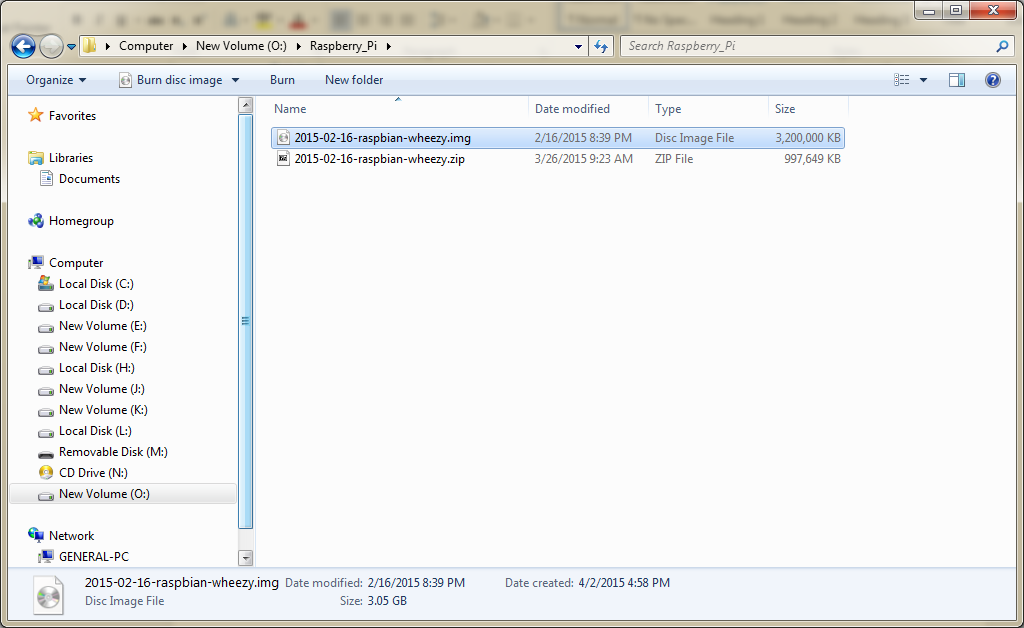
## Installing the OS on the Raspberry Pi

Download the image file from the Raspberry Pi website. It’ll be a zip file as shown below:



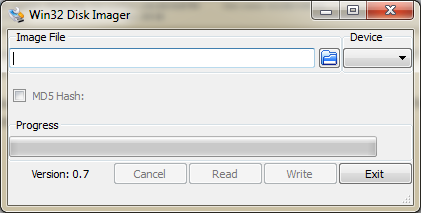
Unzip the file to reveal the Image file with .img extension

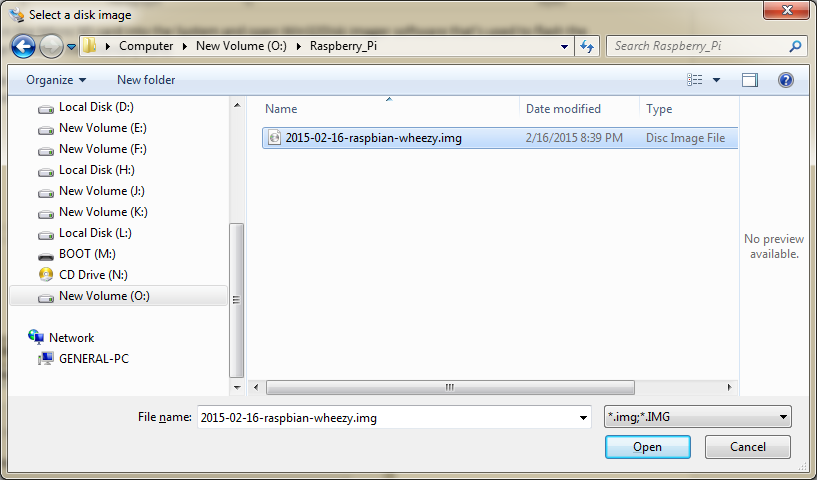


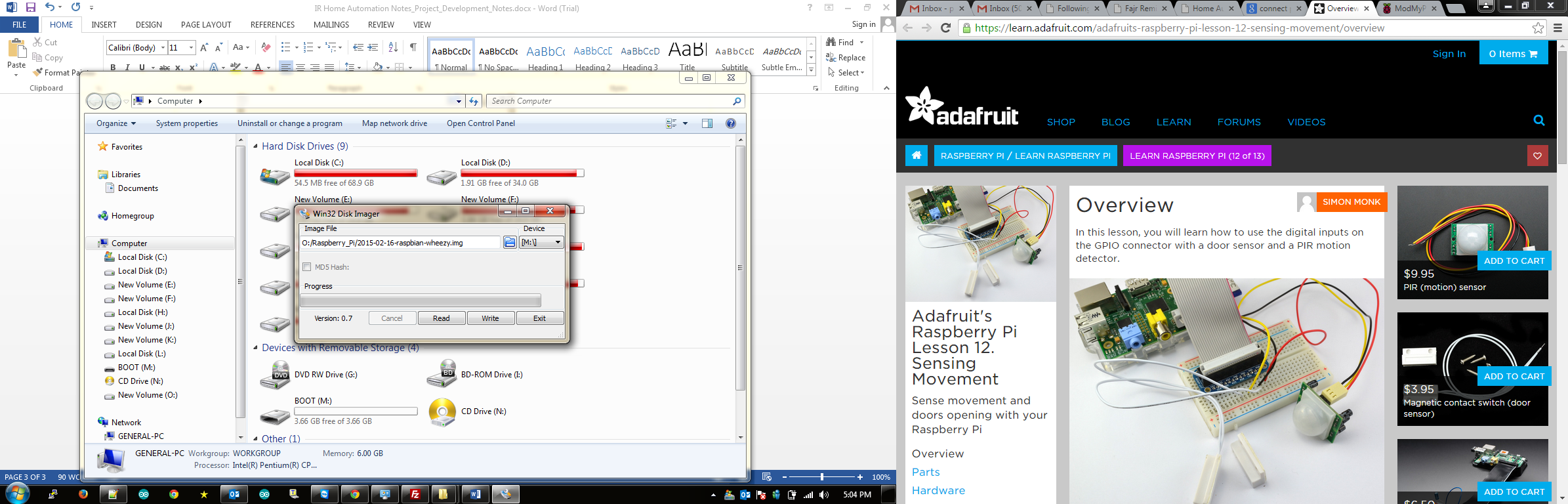


Plug in the Micro SD card into the System and open Win32Disk imager software that’s used to flash the Image file onto the Memory card.

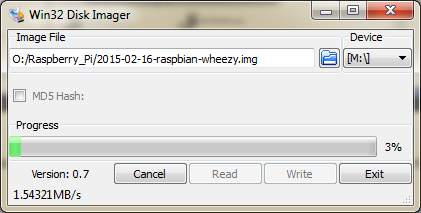
Inside the application, select the Destination drive as the SD card and in the source file, select the recently extracted .IMG file as shown below:



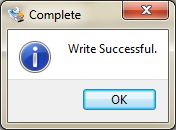


4

Now, hit the **Write** button. The Application will begin writing the OS onto the Memory Card as shown below:



Once done, you’ll see this confirmation dialog box:



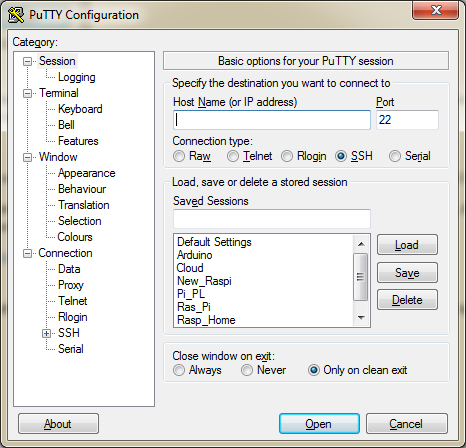
Now, plug in the SD card back to the Raspberry Pi. Use a LAN cable to be able to SSH into the Raspberry Pi if a display device is not connected to the Raspberry Pi.

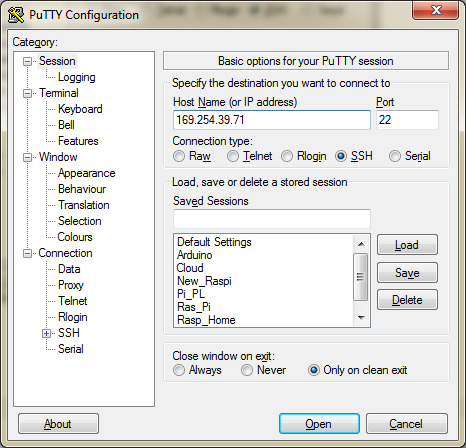
The IP address assigned to my Raspberry Pi after connecting it to the Network is 169.254.39.71.

In order to access the remote terminal of the Raspberry Pi, we need to use PuTTY software, details of which are shown below.

## SSHing into the Raspberry Pi

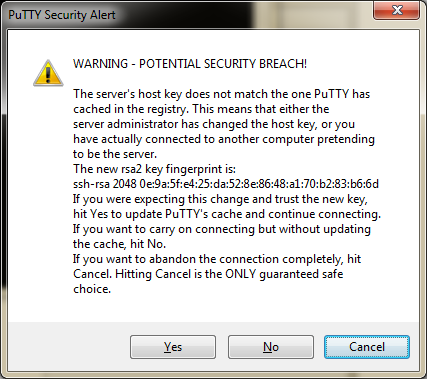
Open PuTTY software and key in the IP address of the Raspberry Pi as determined from the Network:

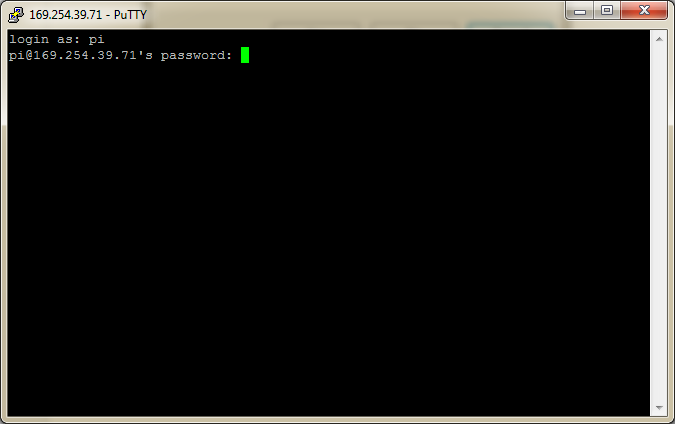




Hit Open

If you get this warning, please select the Yes button.



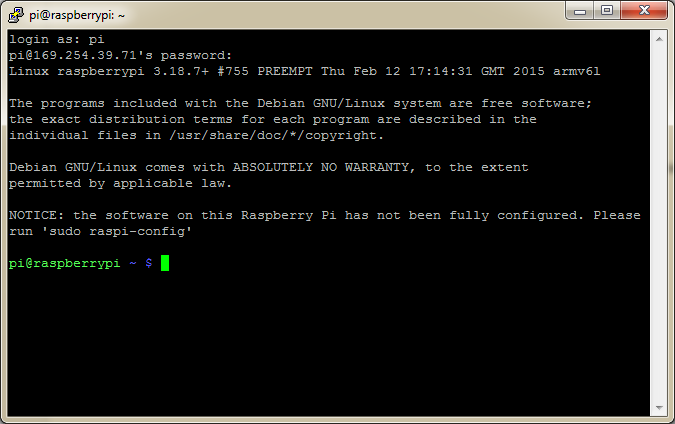


Login with the following default credentials:

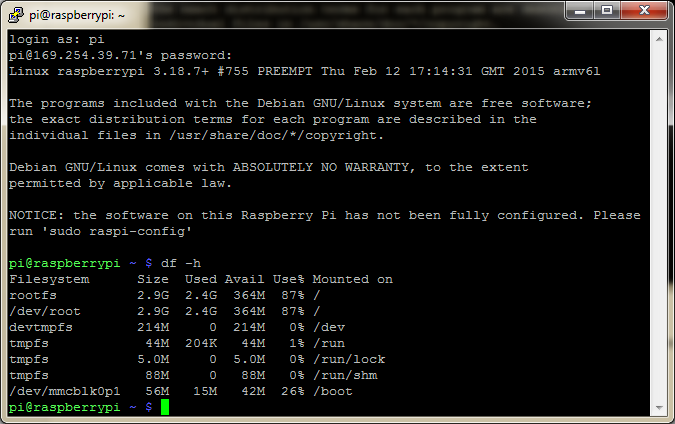
Username: **pi**

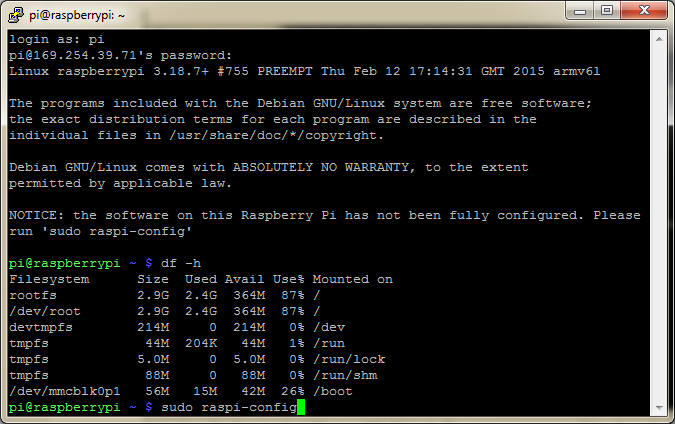
Password: **raspberry**

If you correctly provided the credentials, you should be logged in and should be able to see shell prompt as shown below:

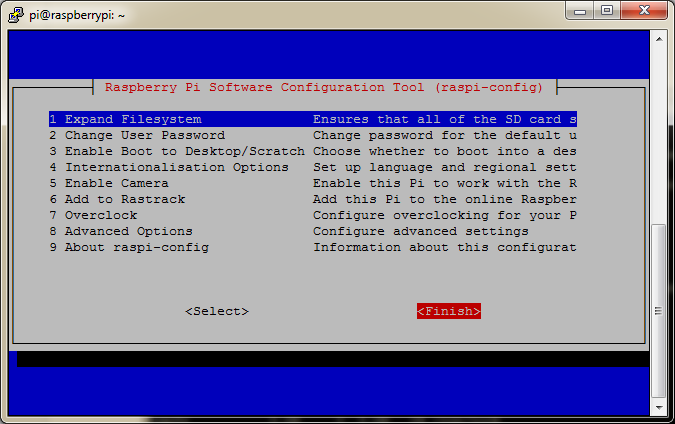


If this is a first installation, we need to run the **raspi-config** command that’ll configure the Raspberry Pi for first use and also expand the root file system.

(Note: how the Size is only 2.9G on a 4GB SD Card. The remaining space will be claimed after running the above command)



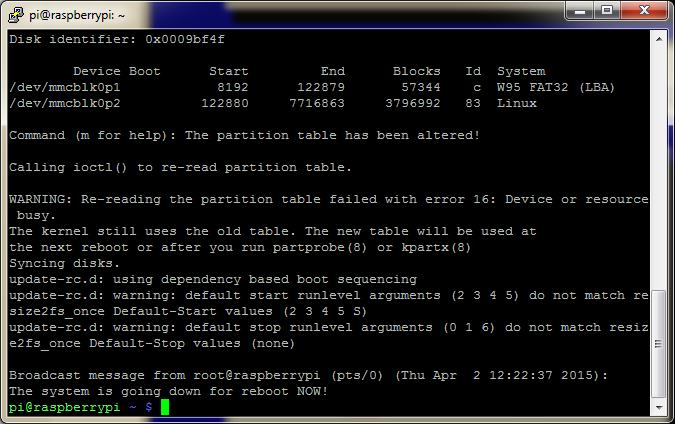
Select Expand rootfs and hit Enter

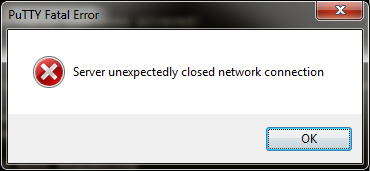




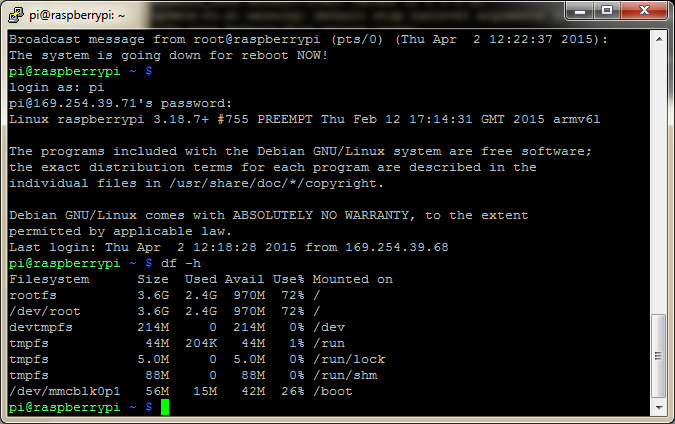
Now, hit Finish. The Raspberry Pi will now reboot for the changes to take effect.







Once the device Reboots, login via SSH again and test for free space:



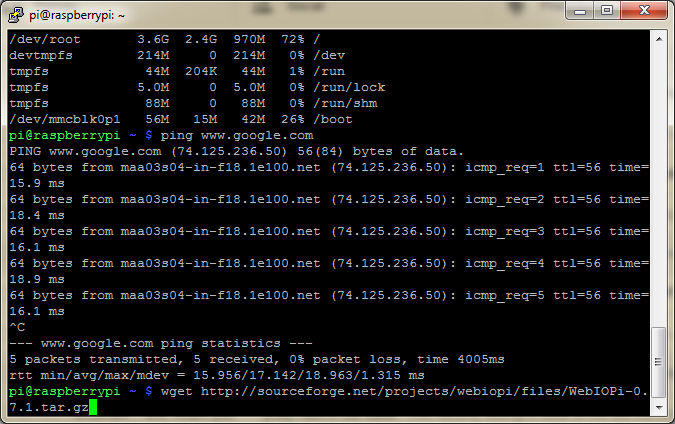
It should now show as **3.6G**

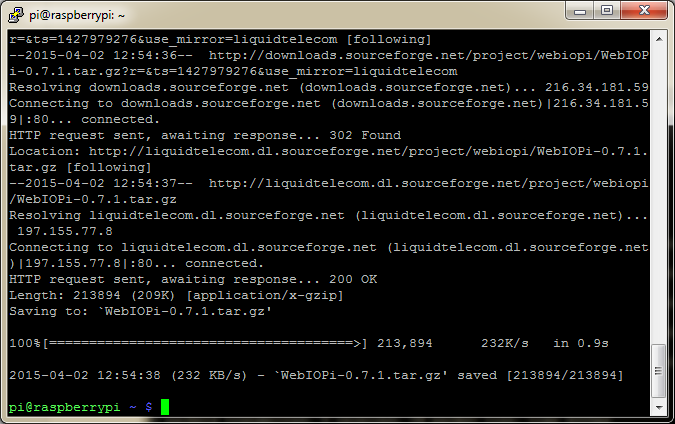
## Installing WebIoPi framework on Raspberry Pi

Navigate to the Downloads page on WebIoPi website: <https://code.google.com/p/webiopi/wiki/DOWNLOADS>

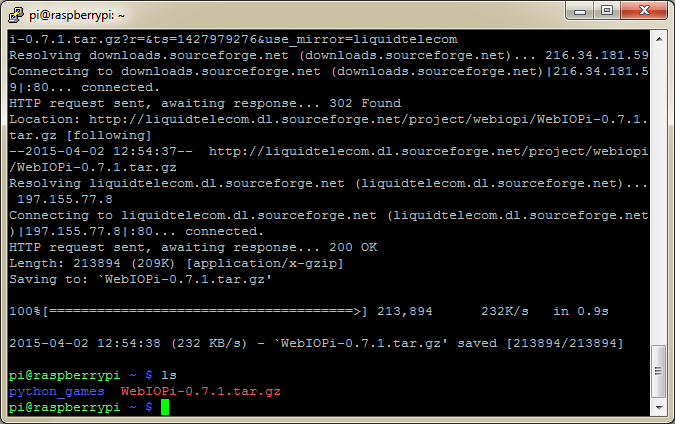
Select the WebIoPi 0.7.1 tar file from this link: <http://sourceforge.net/projects/webiopi/files/>

<http://sourceforge.net/projects/webiopi/files/WebIOPi-0.7.1.tar.gz>

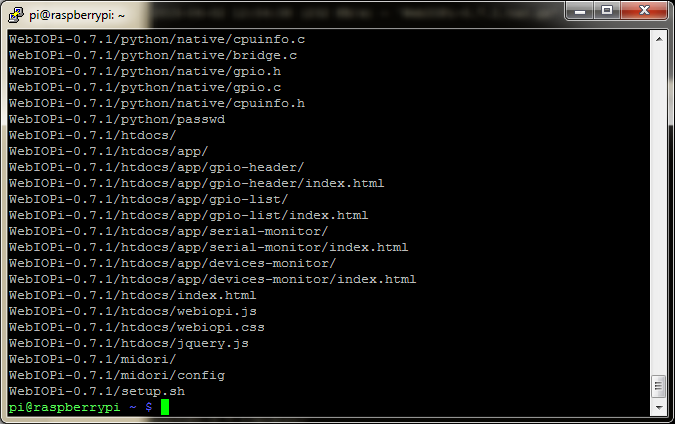
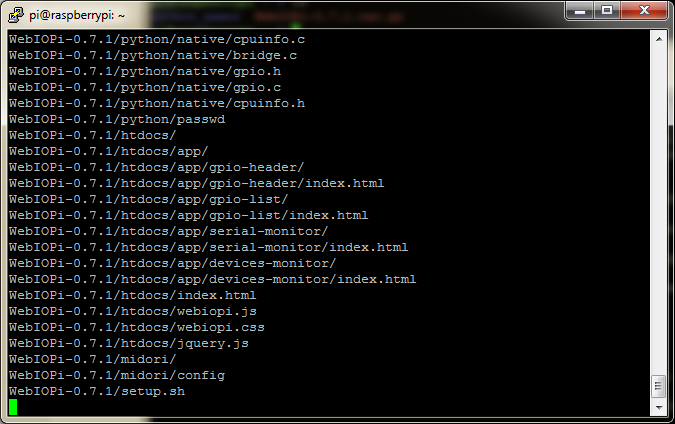
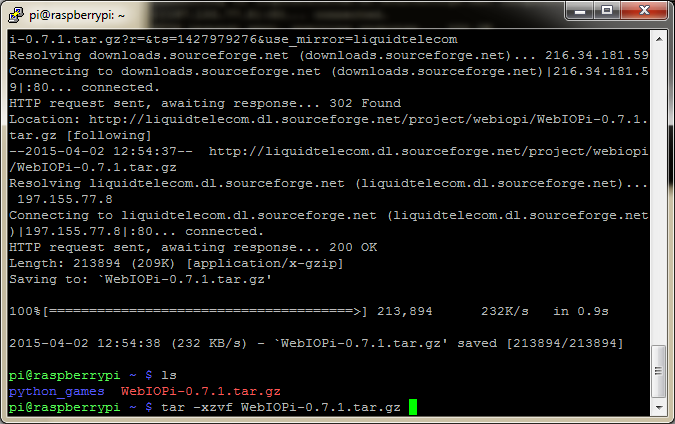




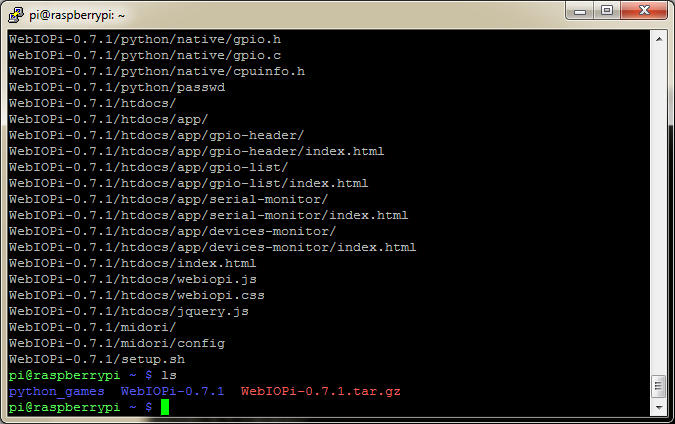
Make sure the file has been downloaded:

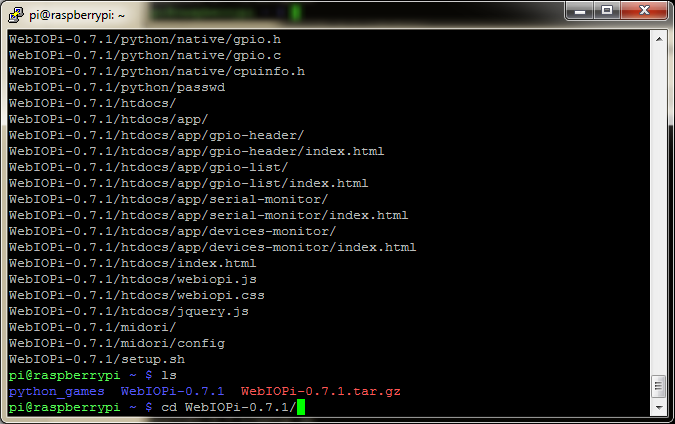


Once done, we need to unzip the file. Use the following command to unzip it:

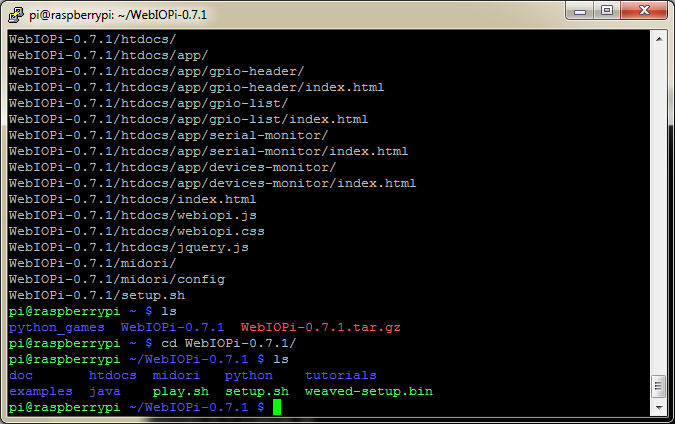


Once it’s been successfully extracted as shown below, we need to change our working directory to the extracted folder:

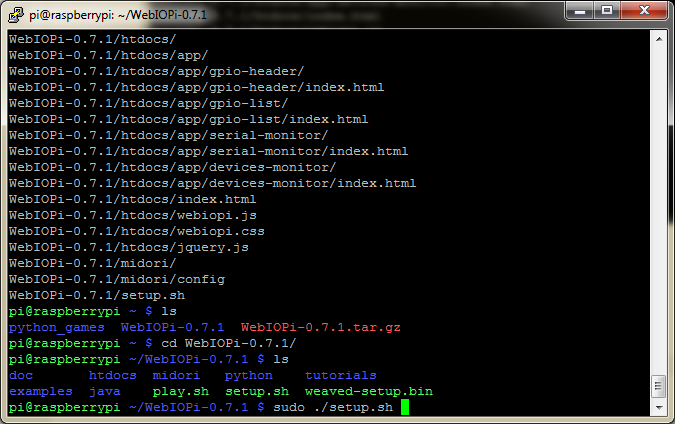




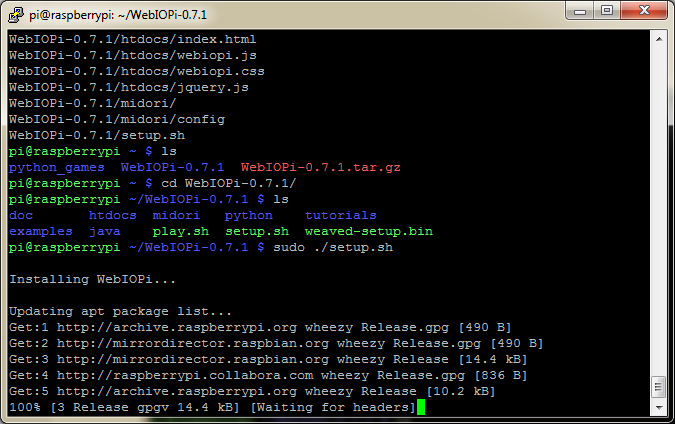
This folder contains the following files as shown below:

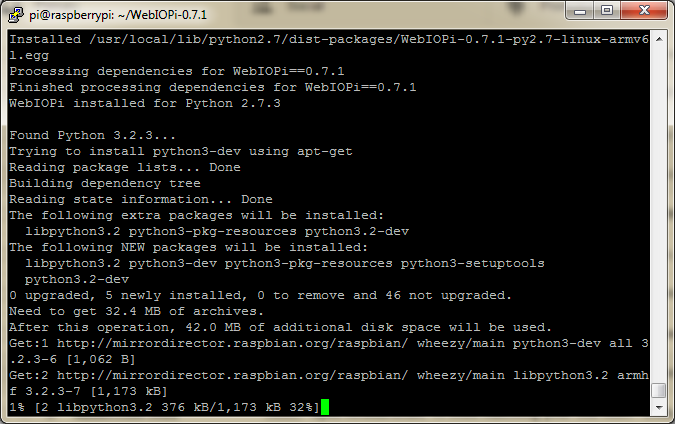


Run the setup file by typing in the following command:

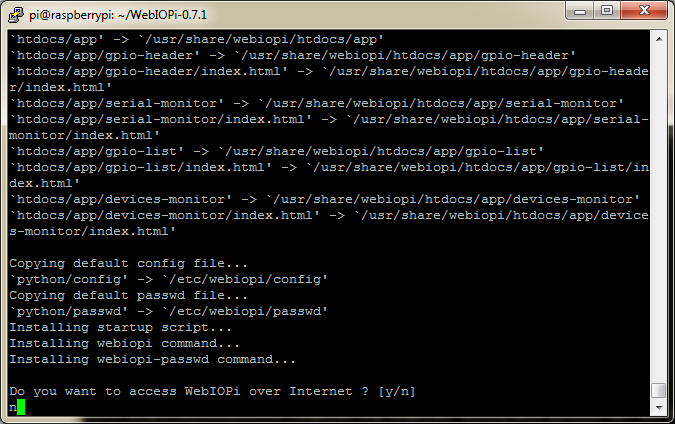


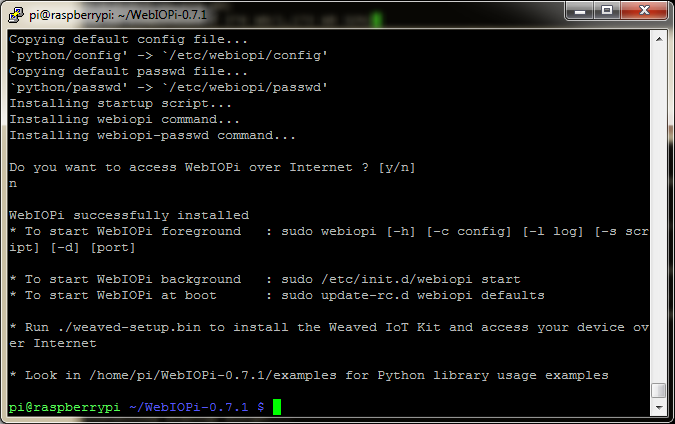
Once done, it’ll start installing and Configuring WebIoPi on the Raspberry Pi device:





If





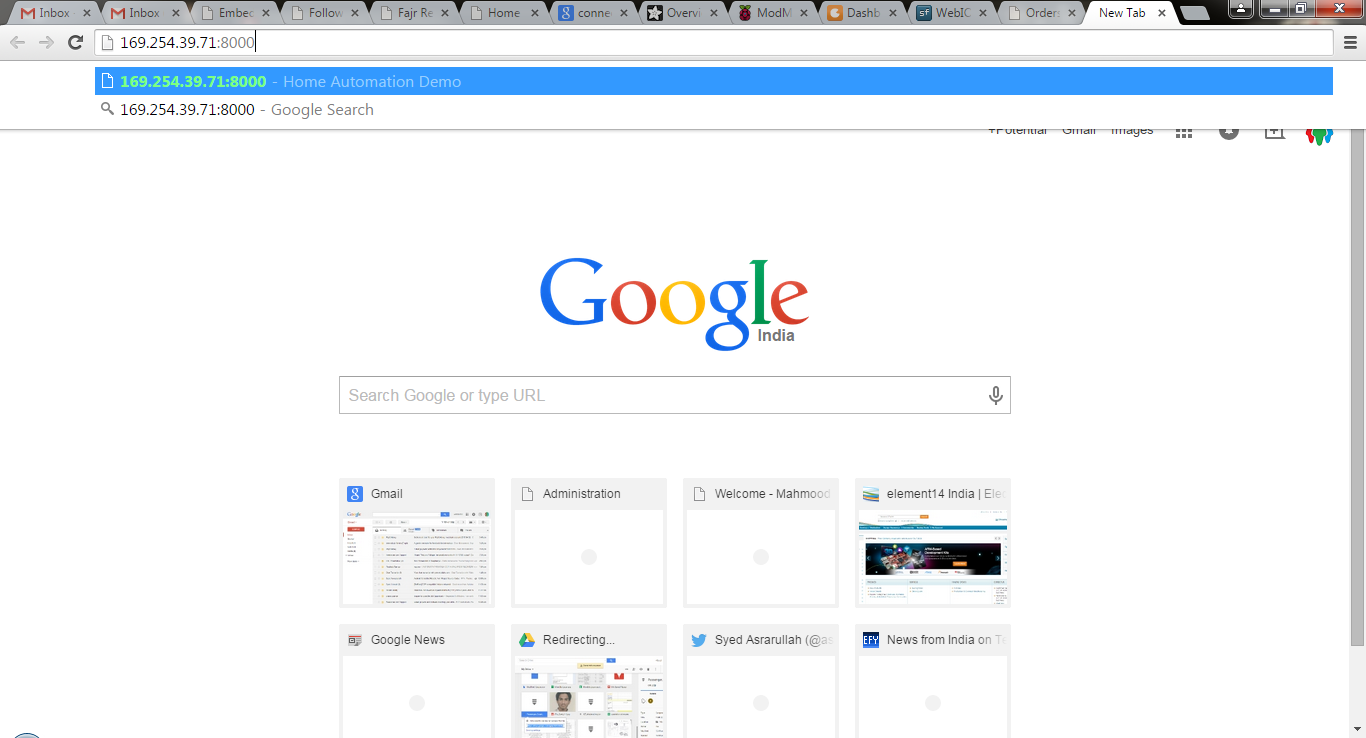
Once done, in order to run WebIoPi, type in the following command:



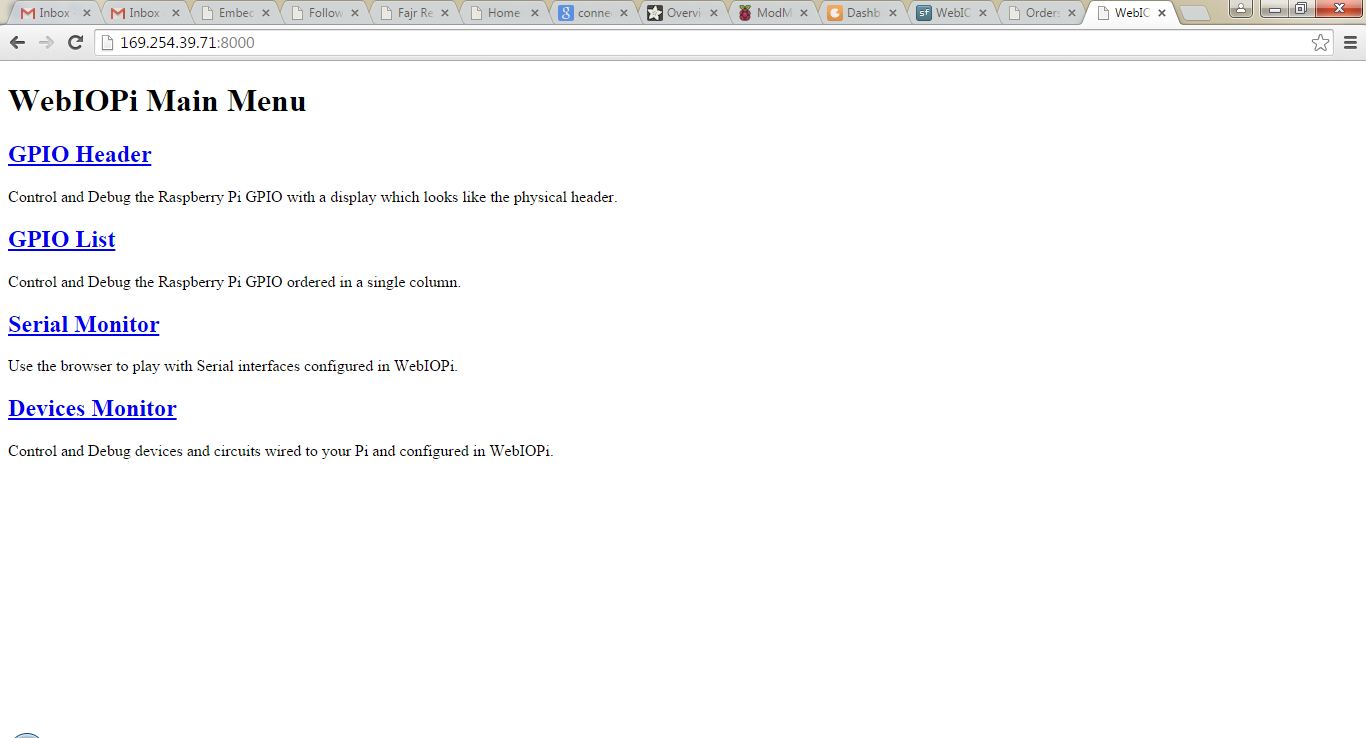
You should see these server messages.

Now, open any browser and hit in the IP address of the Raspberry Pi followed by port 8000 with a colon as shown below:

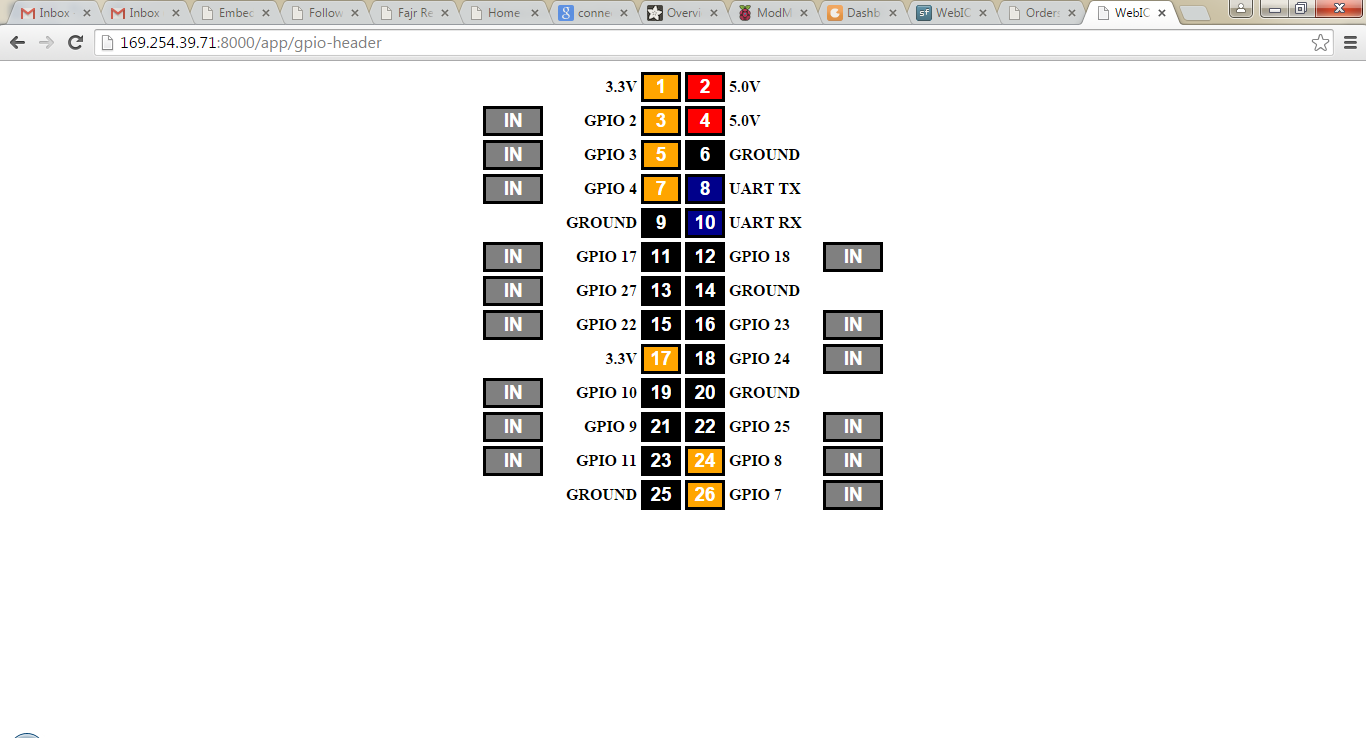
[http://IP:8000](http://IP:8000/)



If you see this page, you’ve successfully configured WebIoPi:



Select the GPIO header to toggle the GPIO pins:



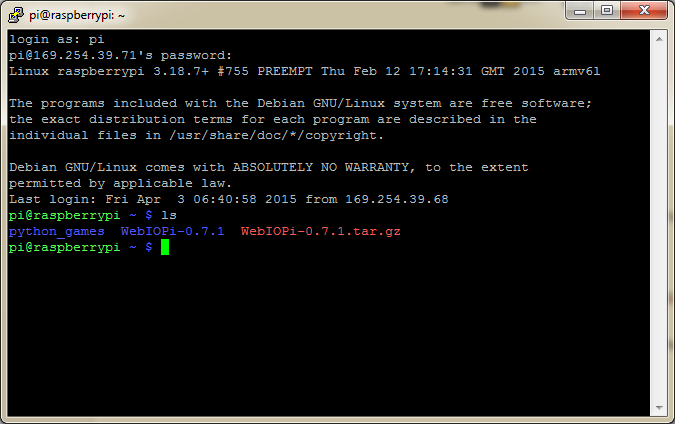
## Creating our Custom Application based on the WebIOPi framework:

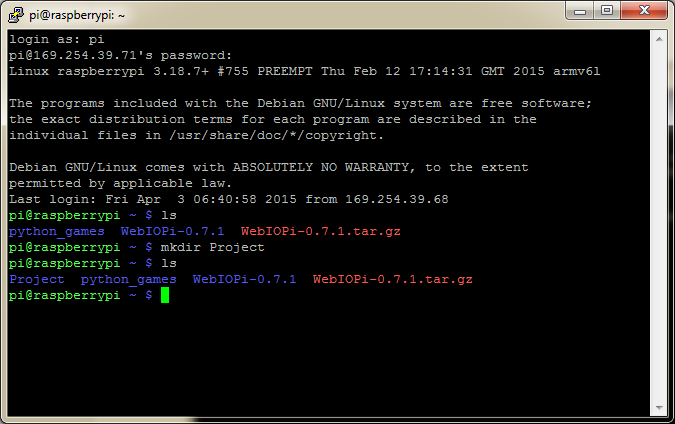
We’ll now look at creating our own Web App that controls Hardware connected to the GPIO of the Raspberry Pi.

The main elements of the App (like any Web App) are:

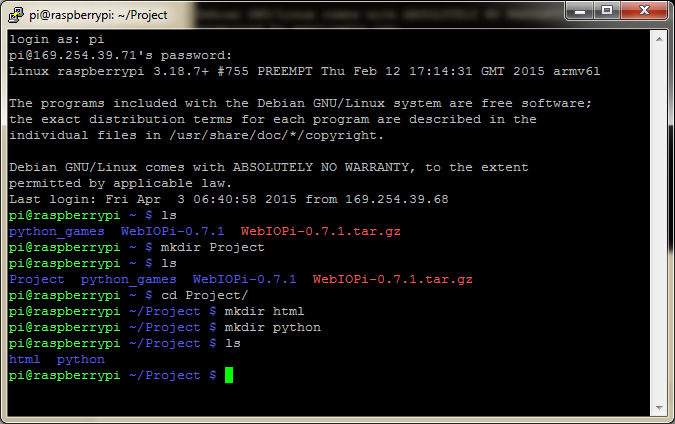
1. Frontend – Which is an HTML page and uses HTML, JavaScript and jQuery
2. Backend – Which is written in Python and works with the GPIO

Firstly, we’d need to create a custom directory within our working directory so that we can later direct the WebIOPi server to load this app instead of the default page.



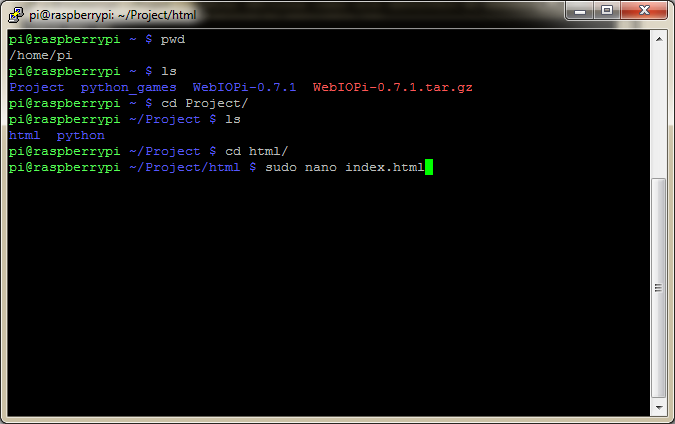


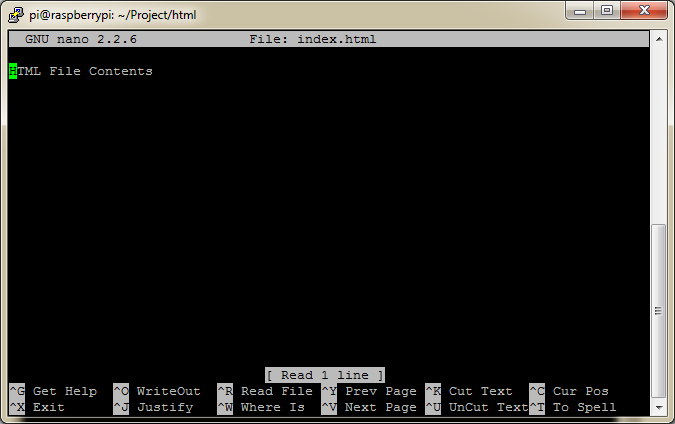
Once created, let’s navigate to that folder and create two subfolders **html** and **python** within it.

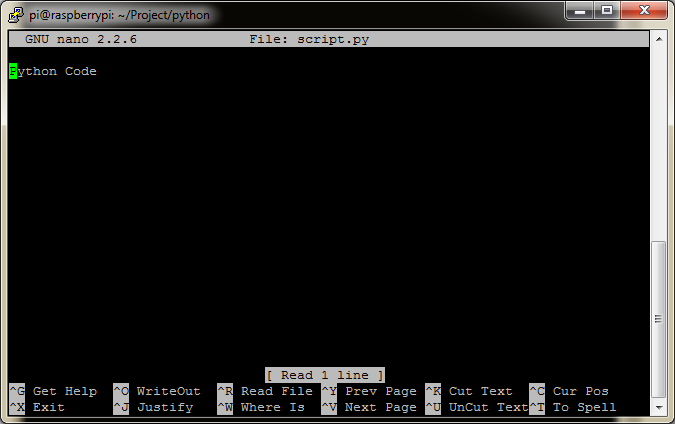


We now have two folders within the main **Project** folder.

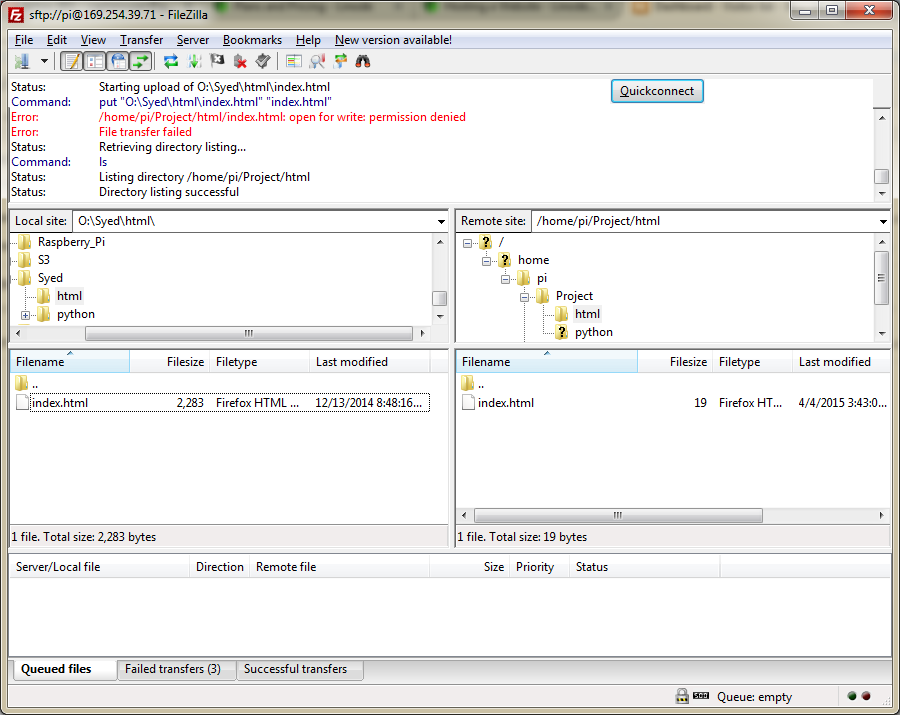
We now need to create a HTML file called as **index.html** within the **html** folder and a **script.py** file within the **python** folder.

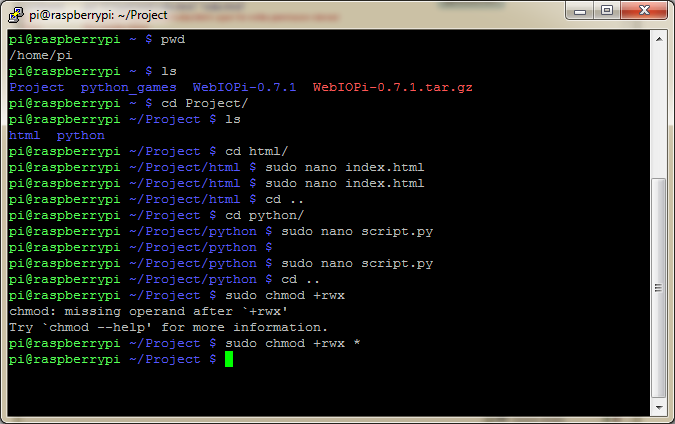




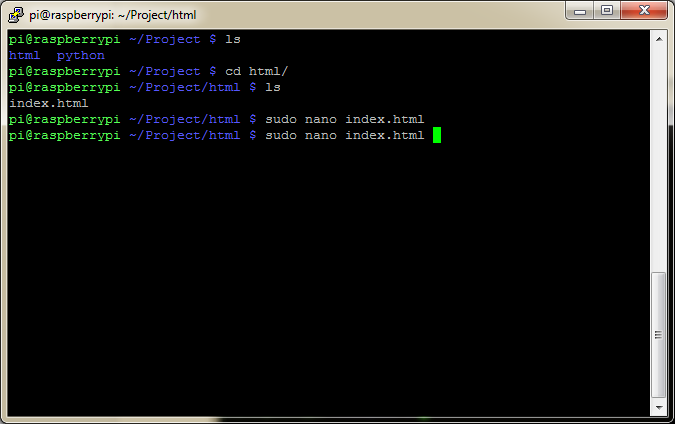


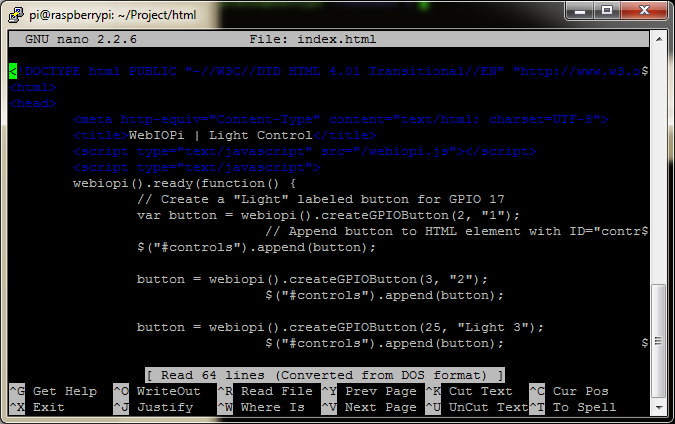
Now, I’ve saved the files locally in my system. We’ll use FileZilla to transfer these files from our System to the Raspberry Pi.

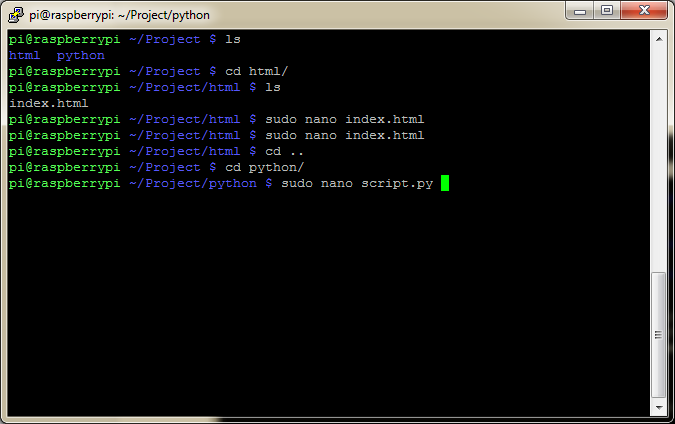
If you get an error like this: 

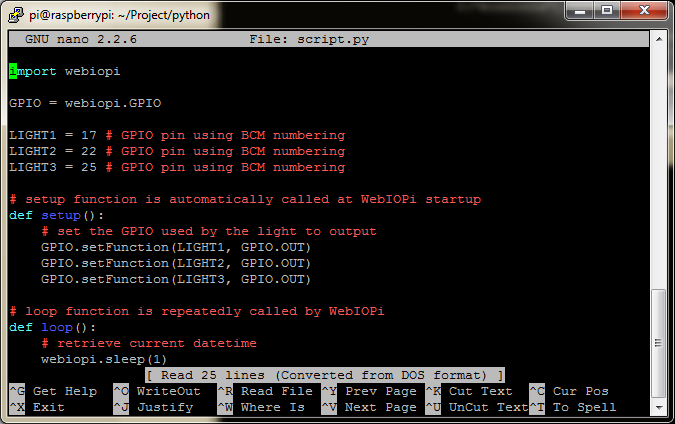


If the error still persists, delete the local file from the Raspberry Pi and copy over from the FTP client.









Once this has been done, we now need to edit the config file of WebIoPi so that it loads our custom Web App.



