Contents

[***Order items to start with Raspberry pi*** 2](#_Toc471065674)

[***Raspbian OS & VNC Downloads and other important informaion*** 4](#_Toc471065675)

[***Install the Raspbian OS in the SD Card*** 4](#_Toc471065676)

[***Boot-up the Raspberry Pi for the first time*** 5](#_Toc471065677)

[***A few more housekeeping*** 8](#_Toc471065678)

[***Experiment# 1: Lighting a LED from Raspberry Pi*** 11](#_Toc471065679)

[***Experiment# 2: Blinking a LED programmatically from Raspberry Pi*** 14](#_Toc471065680)

[***Install LAMP in Raspberry Pi*** 15](#_Toc471065681)

[***Experiment# 3: Lighting LEDs over Internet*** 17](#_Toc471065682)

[***Experiment# 4: Collecting Temperature and Humidity Data*** 21](#_Toc471065683)

# Order items to start with Raspberry pi

It took me hours and days to figure out what exact items do I need to start with Raspberry Pi and IOT experiments.

For your convenience, here are the items I bought -

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **SL#** | **Item Name** | **Description** | **Qty** | **From** |
| 1 | Raspberry Pi 3 Model B | Raspberry Pi Motherboard - heart of the project Other variants of Raspberry Pi is also available, but I found this one as most suitable | 1 | Amazon.com |
| 2 | Raspberry Pi 3 Power Supply | Found in internet that Galaxy S5 charger could also be used, but at owner's risk Hence, thought of buing this one, the cheapest I could find | 1 | Amazon.com |
| 3 | Micro SD Card | Since Raspberry Pi doesn’t have any Hard Drive , it needs an external memory I have bought a 32 GB SD Card | 1 | Buy.com |
| 4 | Micro SD Card Reader | Since we need to install Raspbian OS in the SD card from our PC, needed a SD Card reader | 1 | Buy.com |
| 5 | Breadboard | This is needed to start with experimentation, like blinking LED etc. It came with several Male-to-male jumper wires | 1 | Amazon.com |
| 6 | Jumper wires | Even though my breadboard came along jumper wires, I needed Male to female jumper wires to connect with Raspberry Pi GPIO pins Also, having two different kind of jumpers might come just handy … | 1 | Amazon.com |
| 7 | LEDs | This is the "Guinea pig". It will prove whether our experimentation is successful or not  Since the LEDs are cheap and gets damaged easily, bought a pack of LEDs | NA | Amazon.com |
| 8 | Resistors | In order to experiment with LEDs, resistors are a must We need to connect LEDs & resistors in serial in order to avoid damaging LEDs | NA | Amazon.com |
| 9 | Raspberry Pi Case - OPTIONAL | Case is required to protect your Raspberry Pi | 1 | Ebay.com |
| 10 | Heat Sinks - OPTIONAL | Heat sinks are required to control the temperature of CPU and RAM  This purchase is completely optional | 2/3 | Ebay.com |
| 11 | Motion Sensor |  | 1 | Ebay.com |
| 12 | Light Sensor |  | 1 | Ebay.com |
| 13 | Heat and humidity sensor |  | 1 | Ebay.com |

While you can also order HDMI cable , Monitor , Keyboard etc, I am planning to run the Raspberry Pi in “Headless” mode (i.e, without any monitor & via my Laptop). So, I didn’t order them.

Additionally, you will need at least 1 LAN cable. So, keep it handy.

# Raspbian OS & VNC Downloads and other important informaion

While the hardware is on the way, it’s time to install Win32 Disk Imager and Raspbian OS

* **Win32 Disk Imager Download Link** : <http://sourceforge.net/projects/win32diskimager/files/latest/download>
* **Raspbian OS Download Link :** <https://www.raspberrypi.org/downloads/raspbian/> . Once the Raspbian Image zip is downloaded, you can unzip it and keep it ready for later use
* **Real VNC Viewer Download Link :** <https://www.realvnc.com/download/file/viewer.files/VNC-Viewer-5.3.2-Windows-64bit.exe>
* **Also, we will be needing Putty and WinScp.** So, please download and install them – if you don’t have them already
* **Keep the Raspberry Pi 3 Pin Diagram handy :** <http://pinout.xyz/pinout/pin1_3v3_power>

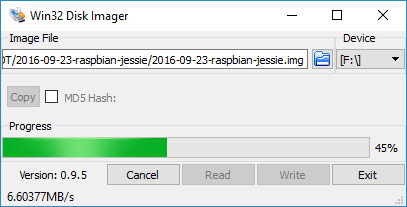
# Install the Raspbian OS in the SD Card

Now, since my packages finally arrived , it’s time to start the game …

First thing first, we need to install the Raspbian OS in the micro SD Card.

For that, do these steps :

* Insert the SD card into the SD Card reader slot
* Plug the SD Card reader into the USB port of your PC
* Open Win32 Disk Manager and select the already downloaded Raspbian OS image and the SD Card. It should look like below –



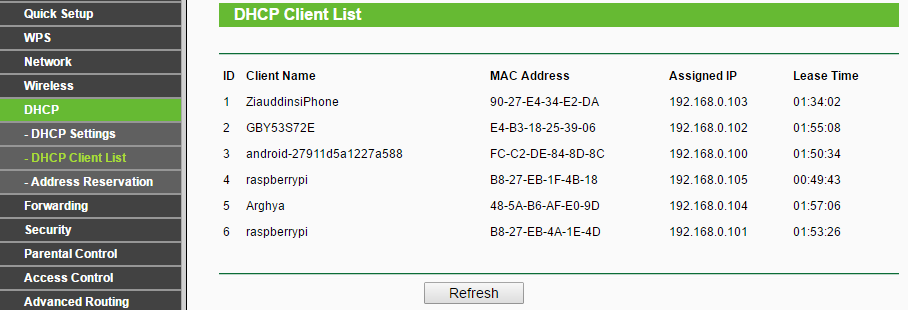
* Click on the [Write] button and it will start the write process. There will be a pop-up warning that the disk might get corrupted, but we can just ignore that
* It might take several minutes for the write to get completed. So, be patient
* It should display “Write Successful” message at the end of the write
* Now, gracefully “Eject” your SD card (don’t just pull the SD card reader/writer off our USB slot)

# Boot-up the Raspberry Pi for the first time

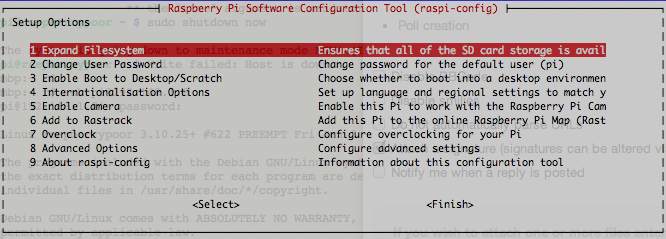
Well done !!! Now it’s time to boot-up our tiny Raspberry Pi …

For that, follow below steps –

* Insert the micro SD card into the memory card slot of the Raspberry Pi
* Connect the charger to the Raspberry Pi. A little red light will start blinking as soon as you do so
* Connect the Raspberry Pi with the Router via a LAN cable
* Now login to your Router. The login URL and credentials should be found at the back of your Router or the help guide that should have shipped with your Router. Usually the Router IP will be 192.168.0.1
* Once logged-in t the Router, find out the list of devices connected with the Router. For my TP-Link Router, the information is located in : DHCP 🡪 DHCP Client List . Here is a snapshot from my Router (IP address of my Raspberry Pi is highlighted) –



* Now open your putty. Enter “Host Name” as the IP address of the Raspberry Pi 🡪 Click on [Open]
* When asked, enter **login name as “pi”** and **password as “raspberry”**
* It should allow you to login
* Now, type “sudo raspi-config”& it will open-up the configuration page



* Select “Expand Filesystem” 🡪 use tab key to select <Select> 🡪 Hit Enter
* Once the operation is successful , use tabs to go to <Finish> 🡪 It will ask whether to reboot the Raspberry Pi 🡪 Select <Yes>
* Once the Raspberry Pi is reboot , re-connect to it via putty
* Now, lets’ update and upgrade the Raspbian OS using following two commands –

Sudo apt-get update

Sudo apt-get upgrade

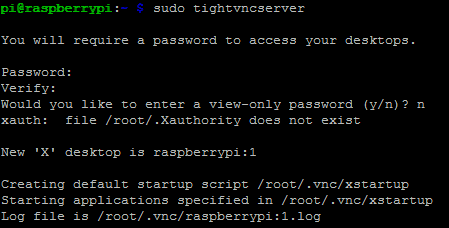
Note : It might take quite some time to perform the operation (30 to 40 mins. or even more ). So, be patient

* Now, it’s time to install Tight NNC Server, so that we can work on the Raspberry Pi in GUI Mode (even in headless mode). For that, execute below command –

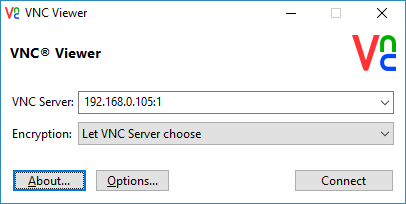
Sudo apt-get install tightvncserver

* Once the installation is done , run the Tight VNC Server using following command : sudo tightvncserver

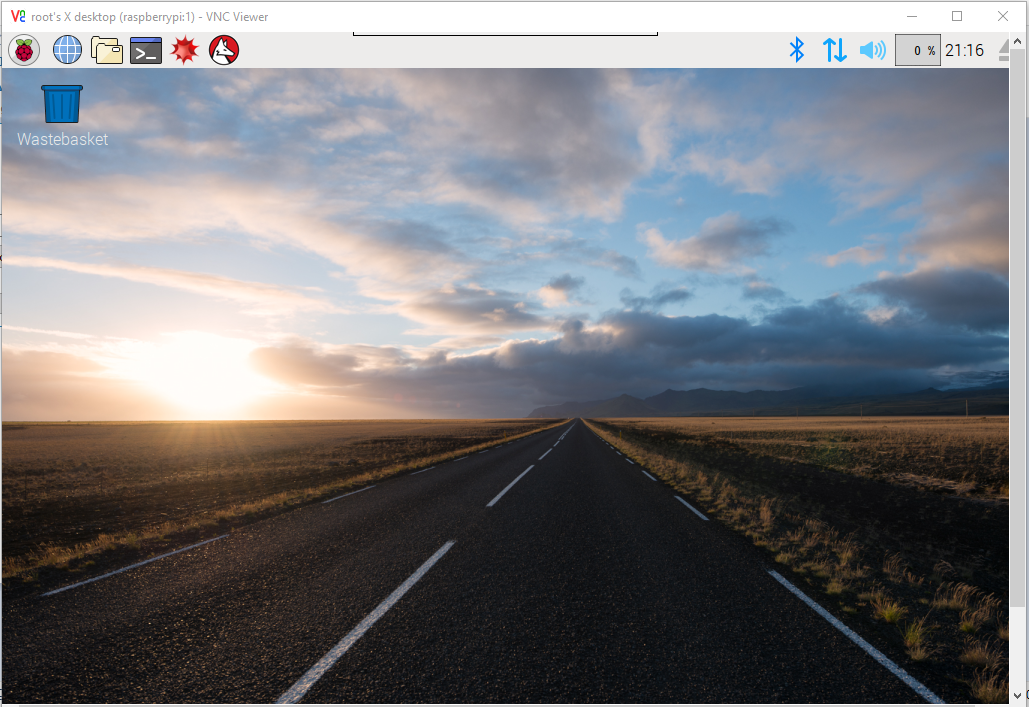
It will yield something like below –



* Cool, now let’s start the VNC Viewer we installed in our PC
* In the “VNS Server” field, enter the IP address of the Raspberry Pi , followed by a colon and then the number appeared after starting the VNC server (1, in my case, as shown in above snapshot) –



* Now, click on [Connect]. When asked for password, enter the one you provided while starting the VNC server. It should bring-up the Raspberry Pi screen like below –

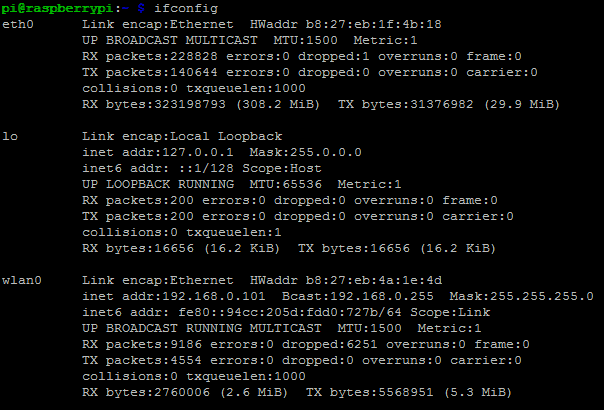


* VOILA !!!!! You are all set-up with your new Raspberry Pi …… Now sky is the only limit for you ☺

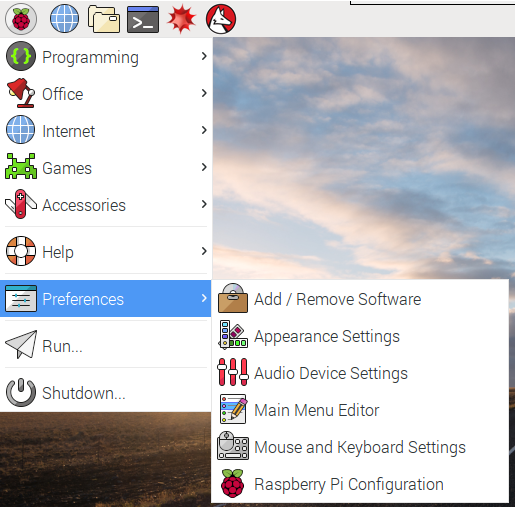
# A few more housekeeping

It’s not mandatory to do the following tasks, but it’s good for our future …

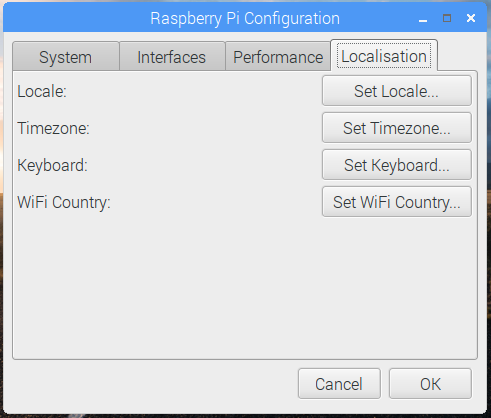
* Once you login to Raspberry Pi using VNC viewer, connect the Raspberry Pi with your Wi-fi. To do so, click on the  symbol at the top right corner of the screen 🡪 select the wi-fi 🡪 enter password 🡪 connect
* Once Pi is connected to Wi-fi, let’s check it’s IP address again . Enter ifconfg command in your putty & look-out for wlan0 address –



* Now, you can disconnect your Raspberry Pi from LAN cable and you can access it via this IP address (Putty , Winscp and VNC Viewer)
* Now, we also need to set the time zone settings in Raspberry Pi. In order to do so, login to Raspberry Pi using VNC Viewer and : Main Menu (the Raspberry symbol at the top left corner of the screen) 🡪 Preferences 🡪 Raspberry Pi Configuration



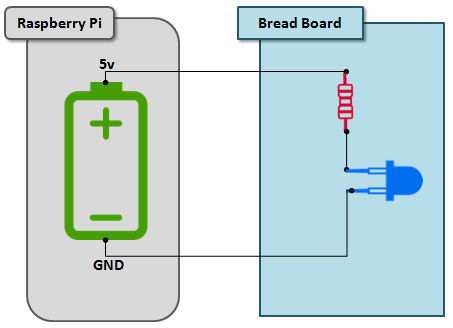
* Now, go to Localisation tab and set all the appropriate values –



* Now, let’s reboot the Raspberry Pi. Enter following command in putty : sudo reboot
* After reboot, if we want to connect to VNC Server, we need to login to putty enter following command : sudo tightvncserver. Then we can connect to Raspberry Pi using our VNC Viewer.

# Experiment# 1: Lighting a LED from Raspberry Pi

* **Objective** : Objective of this simple experiment is to check whether all our tools / devices are functioning perfectly
* **Experiment Target :** Enlightening a LED with the help of Raspberry Pi
* **Required Tools / Devices :**
  + Raspberry Pi (with all the above steps performed on it) . It is assumed that we have already connected the Pi with charger
  + Bread-board
  + Male to male jumpers
  + Male to female jumpers
  + LED
* **Experiment Steps:** In this experiment, we will be trying to create following circuit :



* Connect one end of a Male to Female jumper to Pin# 2 (5V) of Raspberry Pi. The other end need to be connected to +ve lane of the bread-board
* Connect one end of another Male to Female jumper to Pin# 6 (GND) of raspberry Pi. The other end needs to be connected to –ve lane of the bead board
* Take a Male to Male jumper , connect one end of it to the +ve lane of the bead-board and the other end to j23 point of the bread-board
* Take a resistor and connect one end of it at h23 point of bread-board and the other end at h19
* Take a LED and connect +ve end of it (the longer leg) with f19 point of the bead-board and the –ve end (the shorter leg) at f13
* Take another Male to Male jumper and connect one end of it with j13 and the other end with the –ve lane of Bead-board
* This completes the circuit and you should be able to see the LED blinking
* **Evidences :** After completion of my experiment, I have been able to capture following evidences –

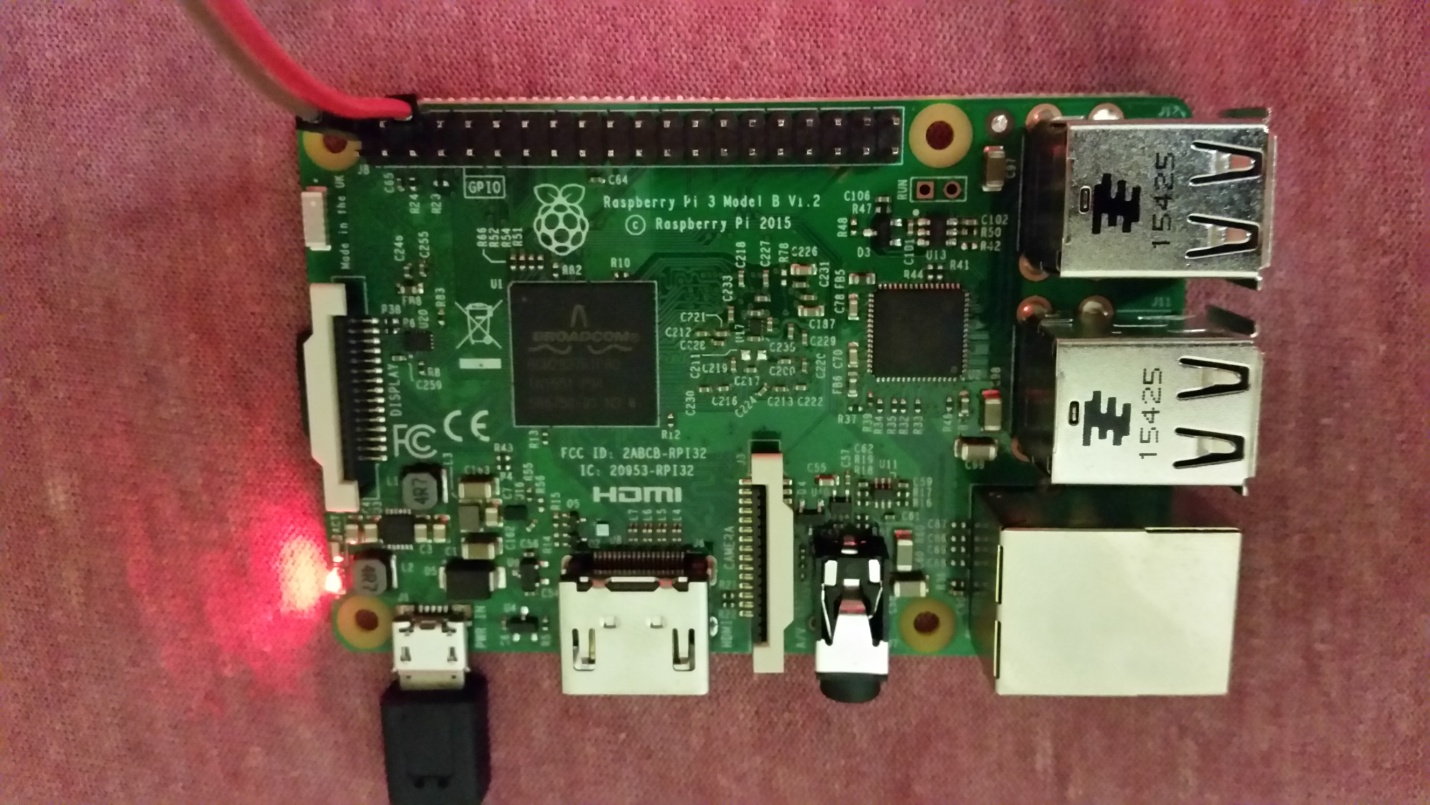


Figure 1: The Raspberry Pi

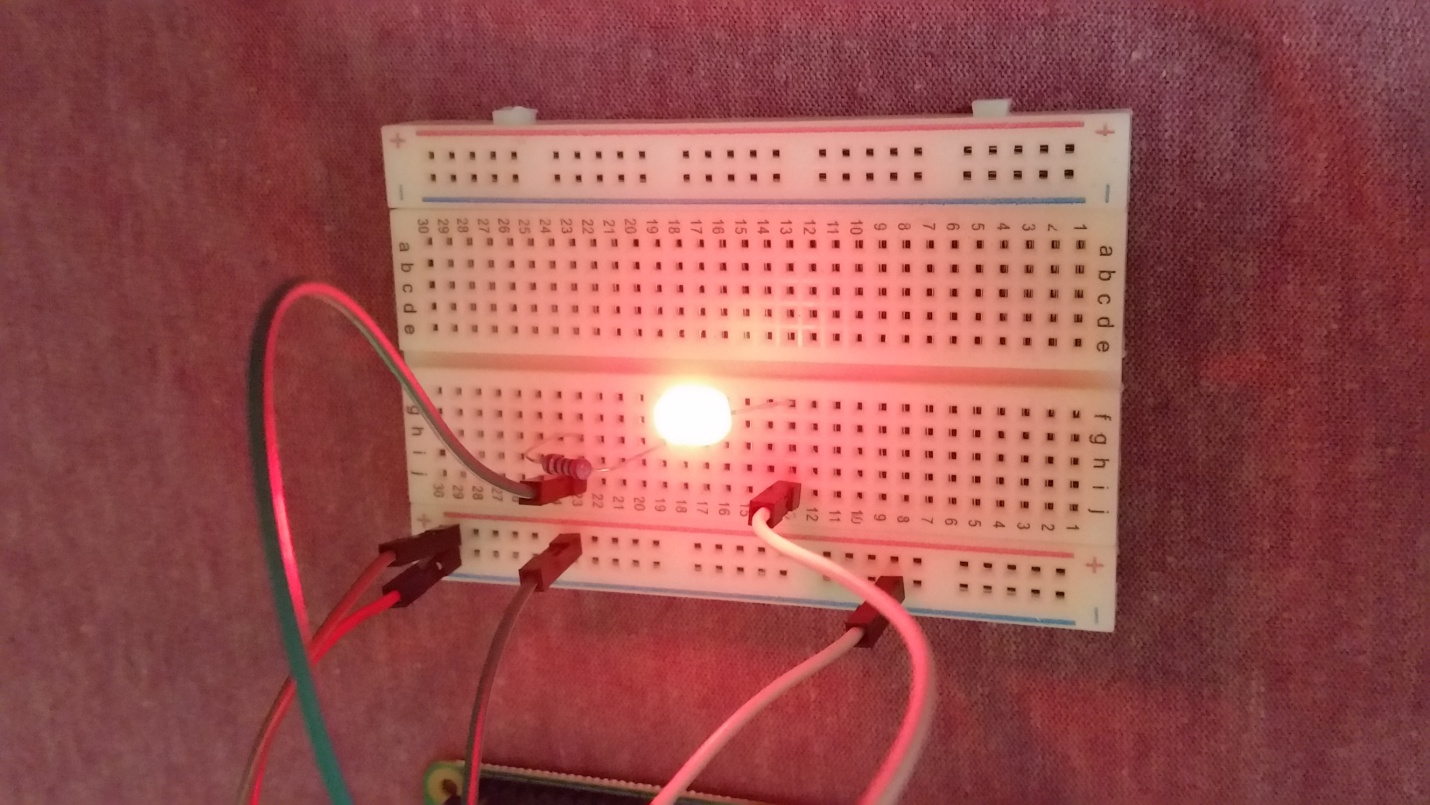


Figure 2: The Bread-board

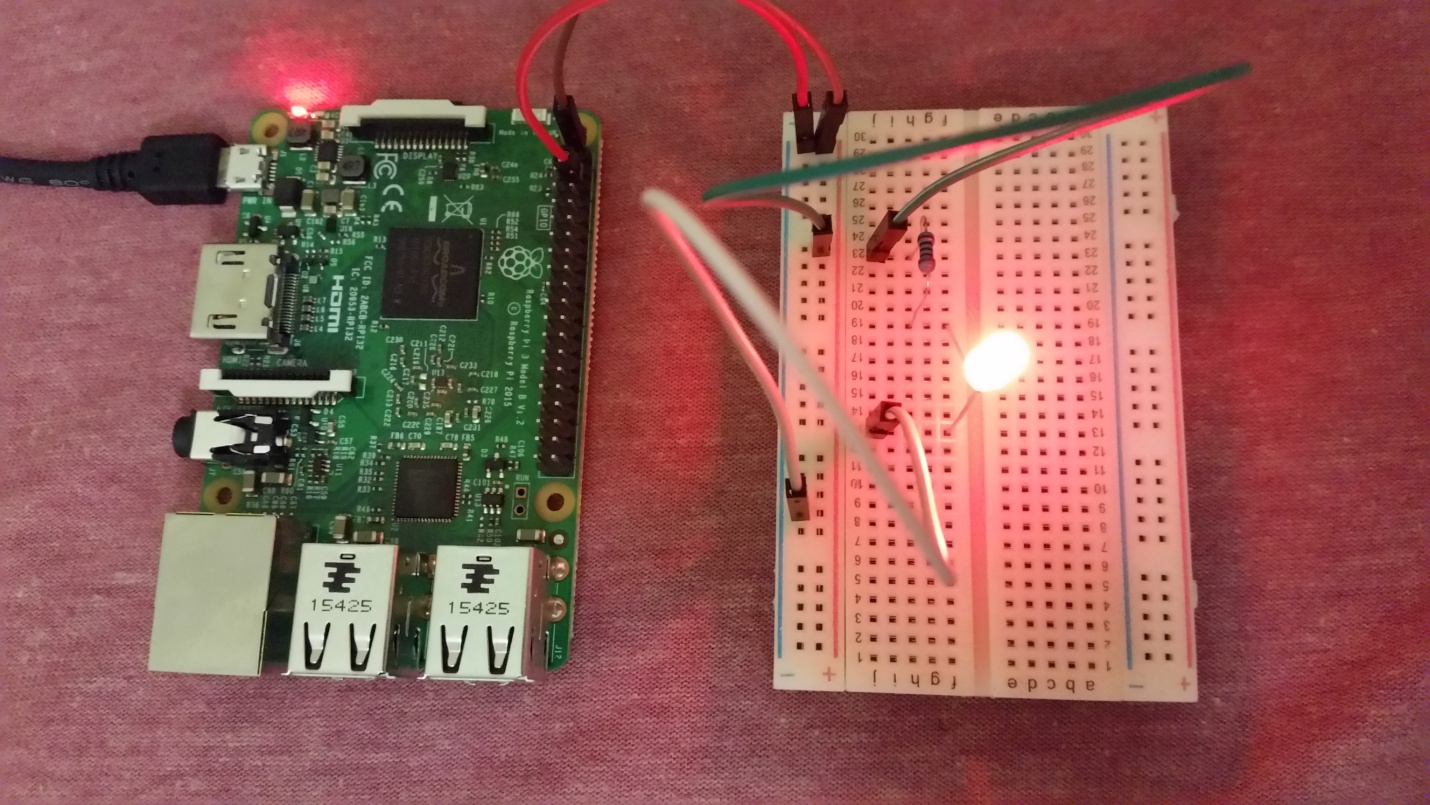


Figure 3: The Complete Set-up

* **Conclusion :** Our set-up is all good and we are ready to roll….

# Experiment# 2: Blinking a LED programmatically from Raspberry Pi

Now it’s time for us to make hands dirty ….

* **Objective** : Making a LED blink via a python script, running on Raspberry Pi
* **Experiment Target** : Without directly powering a LED from Raspberry Pi, we will programmatically light it on and off. This experiment will also help us if we just want to turn of the LED or turn t off programmatically
* **Required Tools / Devices :** 
  + Raspberry Pi (with all the above steps performed on it) . It is assumed that we have already connected the Pi with charger
  + Bread-board
  + Male to male jumpers
  + Male to female jumpers
  + LED
  + Laptop / PC
* **Experiment Steps :** For this experiment, the circuit diagram will remain same as the previous one. But we won’t be connecting the LED directly with +ve pin of the Raspberr Pi. We will rather connect it with one of the GPIO pins and will be sending on/off signals via it. Here are the steps to perform :
  + Connect one end of a Male to Female jumper to Pin# 11 of Raspberry Pi. The other end need to be connected to +ve lane of the bread-board
  + Connect one end of another Male to Female jumper to Pin# 6 (GND) of raspberry Pi. The other end needs to be connected to –ve lane of the bead board
  + Take a Male to Male jumper , connect one end of it to the +ve lane of the bead-board and the other end to j23 point of the bread-board
  + Take a resistor and connect one end of it at h23 point of bread-board and the other end at h19
  + Take a LED and connect +ve end of it (the longer leg) with f19 point of the bead-board and the –ve end (the shorter leg) at f13
  + Take another Male to Male jumper and connect one end of it with j13 and the other end with the –ve lane of Bead-board
  + Now , download the python script to use from Github : <https://github.com/ArghyaChakraborty/RaspberryPi_Project/blob/master/ledBlink.py>
  + Transfer the file to Raspberry Pi via Winscp
  + Now , in Putty, traverse to the location where the python script is placed
  + Execute below command to execute the python script : python ledBlink.py
  + It will start making the LED blink
* **Evidences :** After conducting the experiment, I had been able to capture following video –

<https://youtu.be/4tFQDRNtiCk>

* **Conclusion :** We have been able to programmatically interact with the Raspberry Pi

# Install LAMP in Raspberry Pi

Now, let’s step back from experimentation from a while and get some other important work done, namely installation of LAMP.

LAMP stands for : Linux + Apache (Server) + MySQL + PHP. This will help us in further experimentation.

Follow below steps :

* Install Apache , PHP and MySQL : sudo apt-get install apache2 php5 php5-mysql mysql-server
* During installation , it will ask for MySQL root password. Please set it and remember the same
* Post installation , a directory /var/www/html will be created. This is the place where all our PHP codes need to be placed
* PHP5 , Apache2 and MySQL will get installed in /etc directory
* Now, let’s MySQL to test whether it has been installed properly. Enter following commands –
  + Mysql –u root –p 🡪 This will ask for the root password & once entered, will give mysql prompt
  + Show databases; 🡪 This should provide a list of existing databases
  + Use <pick one of the database names displayed>; 🡪 This should prompt that the database is changed
  + Show databases; 🡪 This should list all the tables in that database
  + Describe <pick a table name>; 🡪 This should list the columns associated with that table
  + Quit; 🡪 This should log you out of MySQL
  + This confirms that MySQL installation is proper
* Before we move ahead, we need to change some file and directory permissions. Here are the set of commands that we need to execute. What those commands are for, the reason is stated beside each command –
  + sudo chown –R www-data:www-data /var/www 🡪 Change the user and group of /var/www directory to www-data user from root user
  + sudo chmod 775 /var/www 🡪 Change the access mode of /var/www directory
  + sudo usermod -a -G www-data pi 🡪 Assign pi user to www-data group
  + Modify sudoers list & add www-data user in it (this step is important if we want to execute Python scripts from PHP code)

sudo nano /etc/sudoers

When the file opens, add following line at the end and save it :

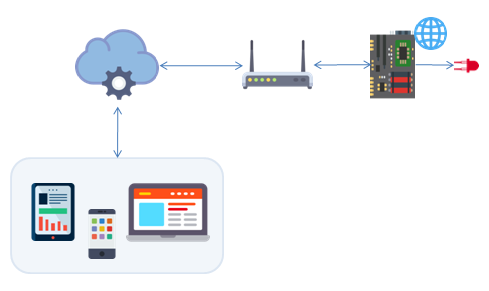
www-data ALL=(ALL) NOPASSWD: ALL

* + sudo reboot 🡪 Reboot the Raspberry Pi for the change to take effect

# Experiment# 3: Lighting LEDs over Internet

Now since we have been able to light the LEDs via python code, we will try to do the same from web, i.e. via a web application.

* **Objective :** Being able to light LEDs from anywhere around the world. If we can be successful, we can (at least theoretically) be able to control our house appliances from a remote location
* **Experiment Target :** Lighting LEDs via a web application
* **Required Tools / Devices :** 
  + Raspberry Pi (with all the above steps performed on it)
  + Bread-board
  + Male to male jumpers
  + Male to female jumpers
  + LED
  + Laptop / PC
* **Experiment Steps :** For this experiment, we will follow below architecture diagram –



1. LED will be connected to Raspberry Pi
2. A python script will be placed in Raspberry Pi to switch on/off the LED
3. A Web Application will be deployed in Raspberry Pi (Apache Server) , which will accept user input via web and invoke the python script to switch on/off the LED
4. The Router, with which the Raspberry Pi is connected, will be configured with Port Forwarding. With this, users will be able to access the web application from anywhere around the world
5. Will register a domain for the Router, so that users of the web application doesn’t have to remember Router’s IP and Port# for invoking the same

Now, we will discuss each step in detail –

1. The Raspberry Pi and LED has to be connected by following the steps in [Experiment# 2](#_Experiment#_2:_Blinking)
2. The python script that has to be written is similar to the one developed in Experiment# 2. In later steps, will provide Github link to download the complete code and place in Raspberry Pi directory
3. The Web Application is developed based on HTML 5, CSS, Jquery , Angular JS and PHP. It’s developed in the form of a Dashboard. The code is placed in Github. You can download the complete code via : <https://github.com/ArghyaChakraborty/RaspberryPi_WebApp.git> . After downloading the code, place it under /var/www/html directory of Raspberry Pi
4. For configuring “Port Forwarding”, follow below steps –
5. Configure Static IP for Raspberry Pi : This is important, because we want the Raspberry Pi to retain the same IP address every time it boots-up. If the IP address changes every time, then the purpose of Port Forwarding will fail. To do so –

* Login to Raspberry Pi via Putty
* Execute following command : sudo nano /etc/dhcpcd.conf
* Once the file gets opened, scroll all the way to the bottom and then copy/paste following lines –

interface eth0

static ip\_address=192.168.0.10/24

static routers=192.168.0.1

static domain\_name\_servers=192.168.0.1

interface wlan0

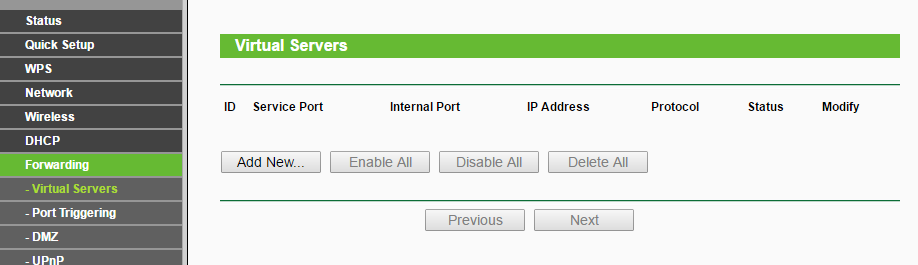
static ip\_address=192.168.0.200/24

static routers=192.168.0.1

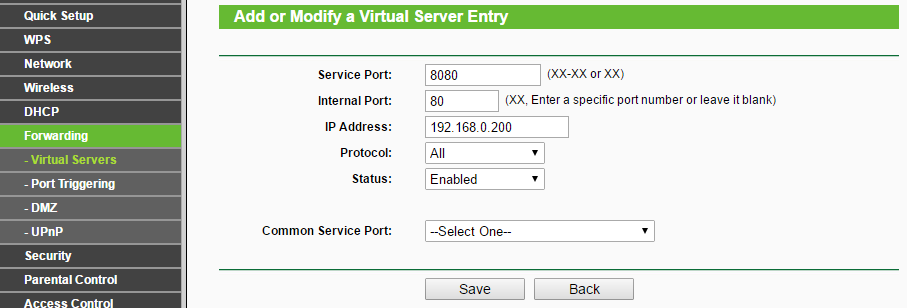
static domain\_name\_servers=192.168.0.1

* Save the file and exit
* Reboot the Raspberry Pi : sudo reboot
* Once the Pi get's ready, check the new IP address of it by logging into your Router's admin page. It should reflect the IP address we have just configured

1. Enabling Port Forwarding :
   * Login to your Router's admin page
   * Go to the Port Forwarding / Forwarding section. For my TP-Link router, it is on : Forwarding tab

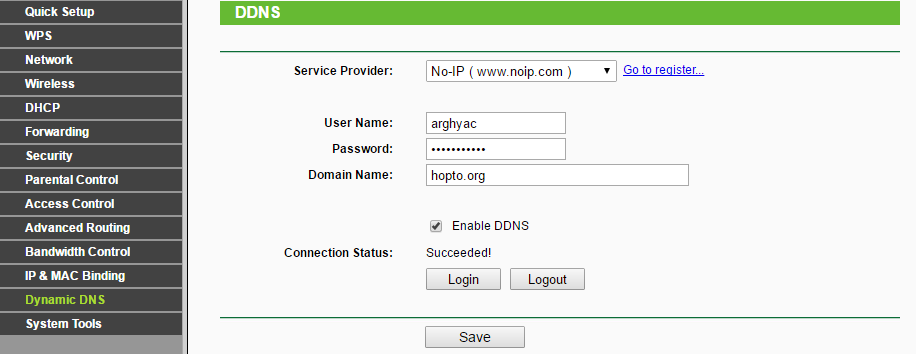


* + Click on [Add New] button
  + Fill the entries as shown in below screenshot. The IP address will be the IP address of your Raspberry Pi, as you just set-up and click on [Save]



1. With the steps being done and + Web Application and Python script being placed + Raspberry Pi and LED connection established, we should be able to invoke the Web Application from Internet by entering URL : http://<Router\_IP>:8080/RASPi
2. Since it’s hard for regular users to remember Router’s IP address and Web Application Service Port, it’s time for us to register for a “FREE” domain and associate our Router with the domain. Follow these steps –

* Go to <http://www.noip.com> , sign-up and register a domain for free
* Now, login to your Router's admin page --> go to Dynamic DNS configuration page --> Enter the details as shown in below screenshot --> click on [Login] --> It should show "successful" message --> click on [Save] button



* + With this setup in place, we should be able to access our web application via http://<No-IP Domain Name>:8080/RASPi
* **Evidences :** After completion of my project, I have been able to collect this evidence :

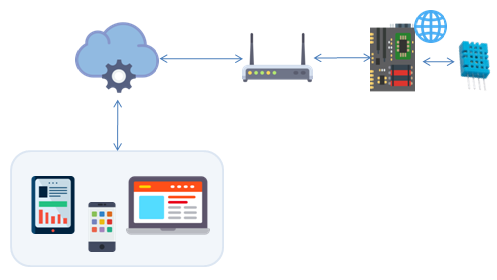
<https://youtu.be/Et8filGHLpk>

* **Conclusion :** We can successfully cotrol electronic appliances from anywhere around the world by using a Web Application deployed in Raspberry Pi

# Experiment# 4: Collecting Temperature and Humidity Data

So far so good. We will now try to collect Temperature and Humidity data via DHT11 sensor.

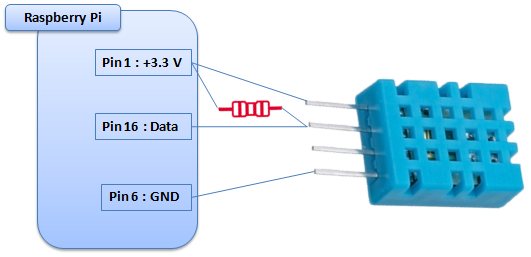
* **Objective :** Being able to collect room temperature and humidity data and project the same ina Time Series chart
* **Experiment Target :**
  + Collecting temperature and humidity data via DHT11 sensor
  + Projecting the data as Time Series in a web UI
  + Controlling DHT11 via web application, i.e. collecting data from it and stop collection of data
* **Required Tools / Devices :** 
  + Raspberry Pi (with all the above steps performed on it)
  + Bread-board
  + Male to male jumpers
  + Male to female jumpers
  + Resistor
  + DHT11 Temperature and Humidity sensor
  + Laptop / PC
* **Experiment Steps :** For this experiment, we will follow below architecture diagram –

****

1. Complete the connectivity between Raspberry Pi and DHT11 Sensor
2. Download required packages for connecting and collecting data from DHT11 sensor
3. Meeting additional prerequisites
4. Download code for connecting and collecting data from DHT11 sensor and deploy the same in Raspberry Pi

Now we will explain each step in detail –

1. The circuit diagram will be as follows –



* Connect Raspberry Pi Pin# 1 to Breadboard +ve Power Lane via a Female to Male jumper
* Connect Raspberry Pi Pin# 6 to Breadboard –ve Power Lane via a Female to Male jumper
* Put the DHT11 sensor pins in Breadboard (say across Lane A)
* Connect Pin# 1 of DHT11 sensor with +ve lane of Breadboard via a Male to Male Jumper
* Connect Pin# 4 of DHT11 sensor with –ve lane of Breadboard via a Male to Male jumper
* Connect a resistor between Pin# 2 of DHT11 sensor and +ve lane of breadboard
* Connect Pin# 2 of DHT11 sensor and Pin# 16 of Raspberry Pi via a Male to Female jumper
* Once the circuitry is complete, it should look like below –

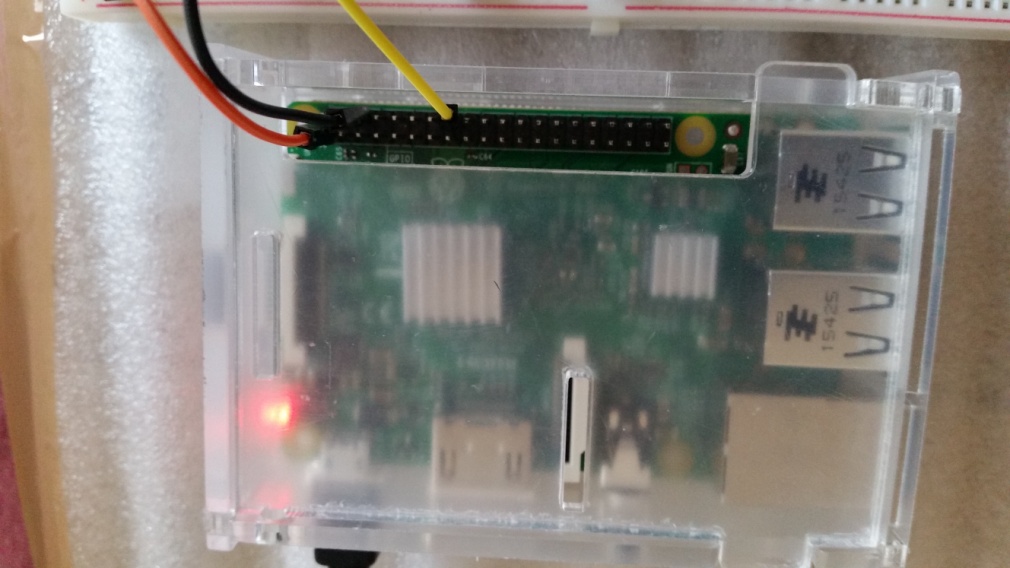


Figure 4 : Raspberry Pi connected to DHT11

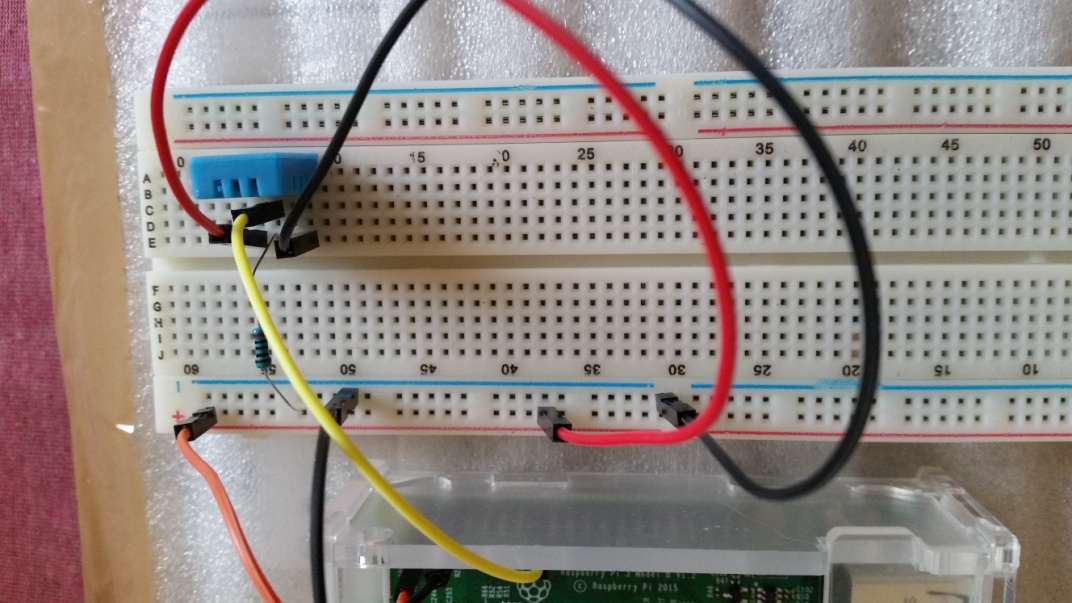


Figure 5 : Breadboard and DHT11 Connected to Raspberry Pi

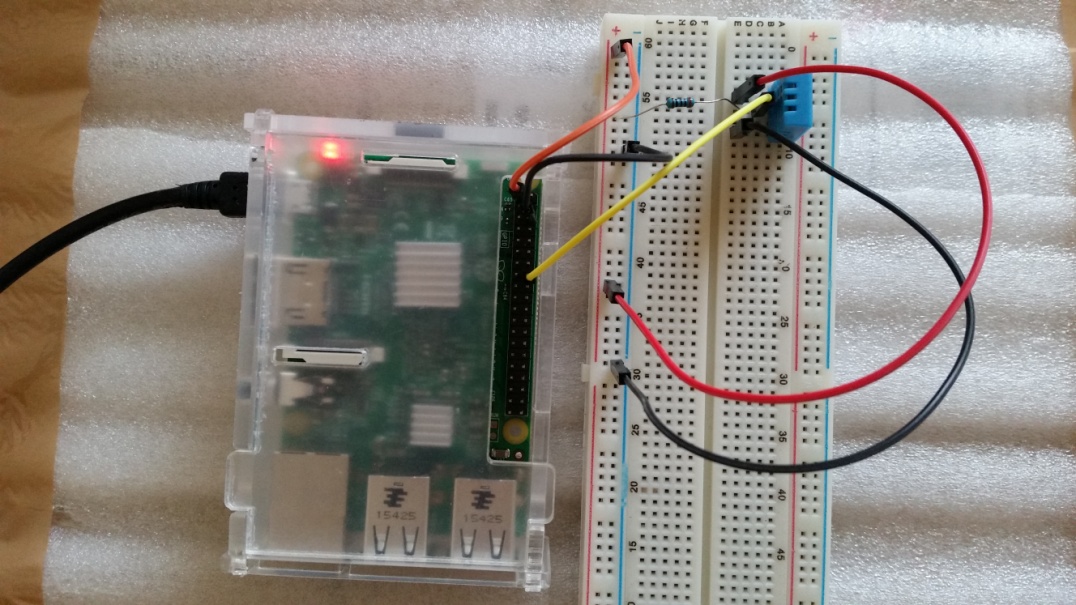


Figure 6 : Raspberry Pi and DHT11 - Complete setup

1. In order for our application to work, we need some Python modules to be downloaded / installed in Raspberry Pi. Following table will detail the commands and their purpose –

|  |  |  |
| --- | --- | --- |
| SL# | Command | Purpose |
| 1 | sudo apt-get update | Update OS |
| 2 | sudo apt-get install python-dev | Package on which Rpi.GPIO depends on. THIS MIGHT ALREADY BE INSTALLED |
| 3 | sudo apt-get install python-rpi.gpio | Install Rpi.GPIO. THIS MIGHT ALEADY BE INSTALLED |
| 4 | git clone https://github.com/adafruit/Adafruit\_Python\_DHT.git | Package which helps in reading data from DHT11 Sensor |
| 5 | cd Adafruit\_Python\_DHT |  |
| 6 | sudo apt-get install build-essential python-dev python-openssl | Packages in which installation of Adafruit\_DHT depends on. MOST OF THE PACKAGES MIGHT ALREADY BE INSTALLED |
| 7 | sudo python setup.py install | Install Adafruit\_DHT Module |
| 8 | sudo pip install pubnub | Download Pubnub Py module. We will use Pubnub’s messaging service |
| 9 | sudo apt-get install python-pip python-dev libmysqlclient-dev | Download MySQL dependencies |
| 10 | sudo pip install MySQL-python | Install MySQL python |

1. We also need to create few MySQL tables and a PubNub account-
   * + Creation of Pubnub Account :
       1. Open <https://www.pubnub.com/>
       2. Click on [Get Started] button
       3. Register for a new account
       4. Upon creation of the account. Login to Pubnub
       5. Click on [NEW APP+] button
       6. Enter new App’s name 🡪 Click on [CREATE] button
       7. Once the App is created, click on it
       8. You should be able to see “Demo Keyset”. You should also be able to create new keyset (if you wish)
       9. Note down the Publish and Subscribe keys
     + Creation of MySQL Tables :
       1. Open a ssh connection (via putty for example) to Raspberry Pi
       2. Type following to login to MySQL:

Mysql –u root –p

It will prompt for password. Please enter the same to proceed

* + - 1. Type below commands to create a new Database & enter into the same:

Create database raspi;

Use raspi;

* + - 1. Create an Audit Table which will contain log of all actions-

CREATE TABLE audit\_table (id INT NOT NULL PRIMARY KEY AUTO\_INCREMENT, date TIMESTAMP NOT NULL DEFAULT CURRENT\_TIMESTAMP, ip VARCHAR(200) NOT NULL , action VARCHAR(500) NOT NULL , note VARCHAR(500) DEFAULT 'NA');

* + - 1. Create a status table which will contain status of all components connected to Raspberry Pi –

CREATE TABLE COMPONENT\_STATUS (Component varchar(500) NOT NULL PRIMARY KEY , Status varchar(500) NOT NULL);

* + - 1. Initialize COMPONENT\_STATUS table with DHT11 status as ‘disconnected’ –

insert into COMPONENT\_STATUS values ('DHT11','disconnected');

* + - 1. Create a table which will keep readings from DHT11 sensor –

CREATE TABLE TEMP\_HUMID\_DATA (id INT NOT NULL PRIMARY KEY AUTO\_INCREMENT, datetime TIMESTAMP NOT NULL DEFAULT CURRENT\_TIMESTAMP, temparature FLOAT NOT NULL , humidity FLOAT NOT NULL);

1. Now, it’s time to download the code from Github and deploy the same in Raspberry Pi. This step assumes that you have already installed PHP , Apache and MySQL Server in Raspberry Pi. If not, please follow the steps provided in “[Install LAMP in Raspberry Pi](#_Install_LAMP_in)” section.

* Copy the zip file @ <https://github.com/ArghyaChakraborty/RaspberryPi_WebApp/blob/master/RASPiUI_V2.zip>
* Login to Raspberry Pi via Putty / Winscp and create a directory ‘RASPi’ under /var/www/html
* Unzip the file and place the contents under /var/www/html/RASPi directory of Raspberry Pi
* Edit config.ini file (sudo nano config.ini) and change following key values –
  + DB\_PASSWORD
  + PUBNUB\_PUBLISH\_KEY
  + PUBNUB\_SUBSCRIBE\_KEY

At this stage, we should be good to go. You can check the web application via http://<your-domain-name>:8080/RASPi. This assumes that we have already configured our router for port forwarding and created a domain in No-IP. If not, please follow the steps 4 & 5 of “[Experiment# 3: Lighting LEDs over Internet](#_Experiment#_3:_Lighting)”

* **Evidences :** After completing this project, I have been able to collect an evidence, which can be viewed @ <https://youtu.be/RaZ_PQZXl3s>
* **Conclusion :** With this experiment, we have been able to read Temperature & Humidity data from DHT11 sensor and project the same as Time series chart. We have also been able to control components connected to Raspberry Pi via a web application