# About the Kit

DIY Home Automation Kit

Key Features:

Remotely control any appliance using an app

No batteries required

Picture of phone showing the app and an appliance plugged into the kit pluged into a wall

What is a smart plug?

What does the smart plug do?

Where can it be used?

How much current can it handle? What can it power?

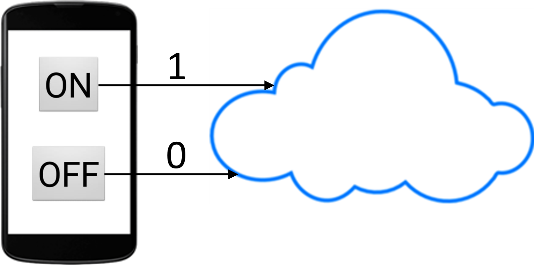
How can it be controlled?

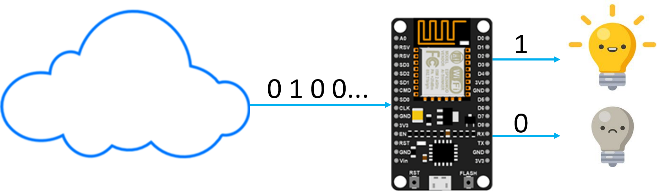
What can the line follower robot do

How long does it last?

What all is involved to build it?

# How does the system work?

When the user presses a button on the app, data is sent by the app and is stored on the cloud. Value 1 is sent if ON button is pressed and value 0 is sent if OFF button is pressed.

NodeMCU on the kit constantly reads data from the cloud. If it reads value 1 from the cloud, it switches ON the relay which turns the appliance ON. If the NodeMCU reads value 0 from the cloud, it switches OFF the relay which turns OFF the appliance.

# List of Components

Very very good looking pictures of actual parts will come here

|  |  |  |
| --- | --- | --- |
|  | Related image | https://images-na.ssl-images-amazon.com/images/I/51n8fVANTIL.jpg |
| Node MCU x1 | Single Relay Module x1 | AC to DC 5V Converter x1 |
|  | Anchor by Panasonic Polycarbonate Roma Multi Socket (White) | https://images-na.ssl-images-amazon.com/images/I/61POUF-j6ML._SL1500_.jpg |
| PCB x1 | Socket x1 | Switch x1 |
|  |  |  |
| Plug x1 | USB Cable x1 | Left Side Plate x1 |
|  |  |  |
| Right Side Plate x1 | Back Plate x1 | Front Plate x1 |
|  |  |  |
| Top Plate x1 | Bottom Plate x1 | Partition x1 |
|  |  |  |
| Nuts and Bolts x80 | Jumper Wires | Wires |

# Building Instructions

The building process involves 2 steps –

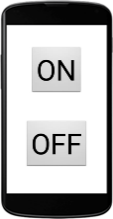
IoT System setup: involves setting up the cloud (ThingSpeak Channel), developing the interface (Android app) and programming the control system (NodeMCU devkit)

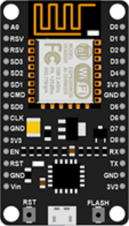
Hardware Setup: Involves assembling the hardware and making connections and assembling the outer box.

## IoT System Setup

The IoT system employed in Smart Plug consists of three basic elements:

Cloud: A ThingSpeak Channel forms the “cloud” part of the IoT system which data will be written on to and read from.

The Interface: An Android App acts as an interface between the User and the IoT System. The user presses a button on the app, which then sends data of a certain value to the cloud.

Control Unit: NodeMCU Devkit forms the control unit. It constantly reads data from the cloud. Based on the value of data, NodeMCU signals the relay module to turn the appliance ON or OFF.

### Setting up the Cloud

ThingSpeak is an IoT analytics platform that allows you to collect, analyze and act on your IoT data.

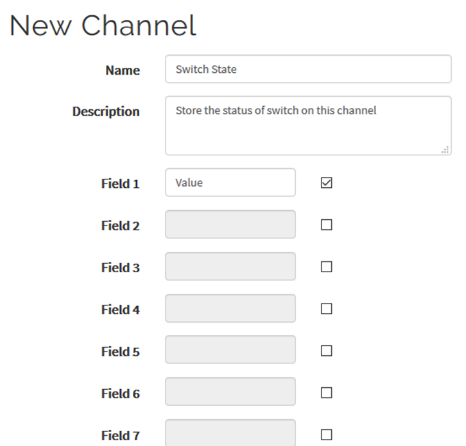
Go to [https://ThingSpeak.com/](https://thingspeak.com/)

Create an account by clicking on sign up and filling up the required details. Once you have a ThingSpeak account, login into your account.

Setting up a new channel:

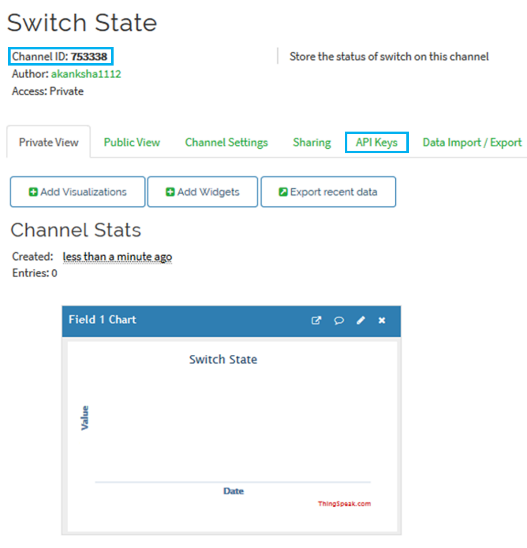
Create a new channel by clicking on the “New Channel” button.

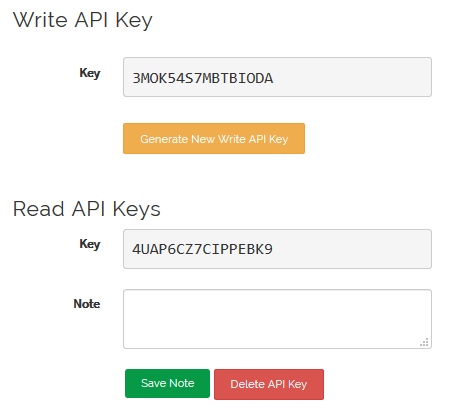
A channel is where you can store your data and retrieve it. A channel can have maximum 8 fields. It means you can store 8 different data to a channel.

Enter basic details of the channel. Here we are creating the channel to store the state of the switch, so we need only one field.

Scroll down and save the channel.

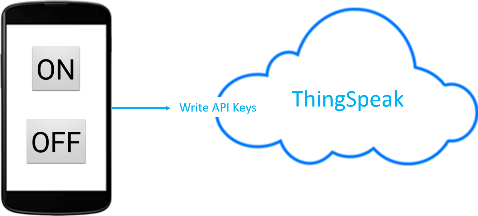
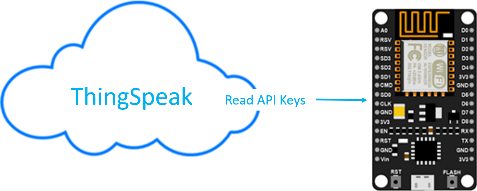
Channel ID:

Channel ID is the identity of your channel.

Click on the API tab to know your API keys.

API Keys:

API (Application Programming Interface) keys are the keys to access to your channel. In simple language you can understand that these are password to access your channel. You can access your channel in two ways-

* 1. To write data to the channel: API Write Key will be used to write data to the channel by the Android App in the IoT System.
  2. To read data from the channel: API Read Key will be used to read data from the channel by the NodeMCU in the IoT System.

### Developing the Interface

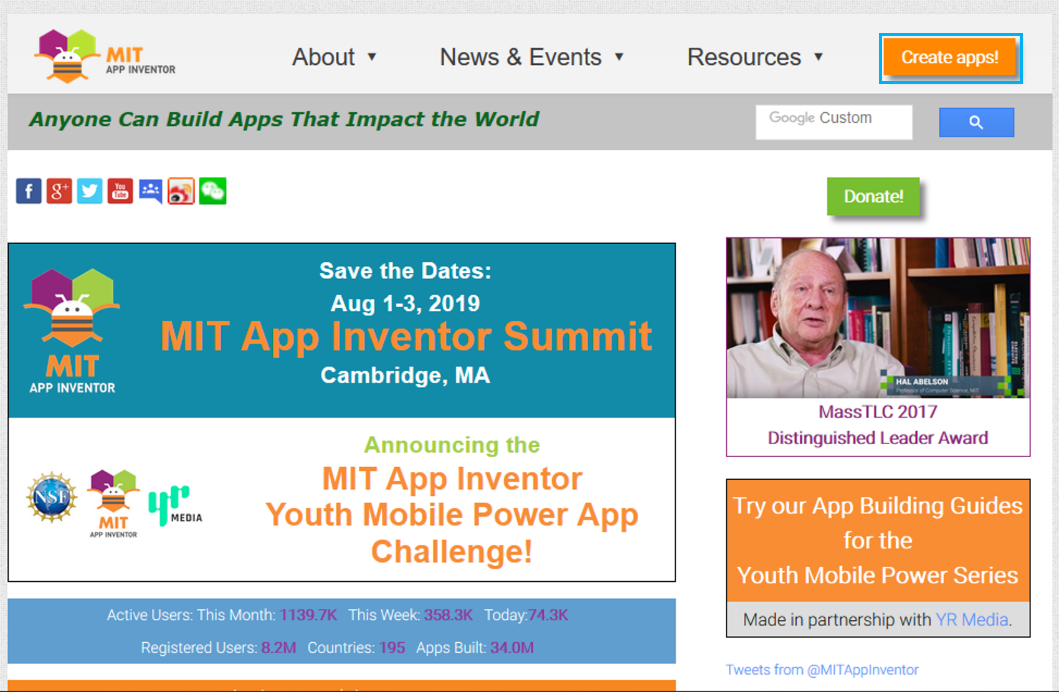
We will be using an Android App as an interface for the IoT system.

To build the app, we will be using a platform called MIT App Inventor.

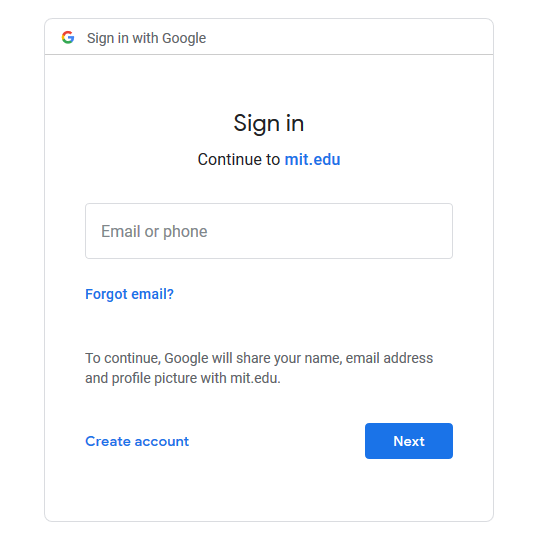
MIT App Inventor is an open-source web application originally provided by Google, and now maintained by the Massachusetts Institute of Technology (MIT). It allows everyone and even children to build fully functional apps for smartphones and tablets for the Android operating system (OS).

Follow the steps to develop the app for the IoT System:

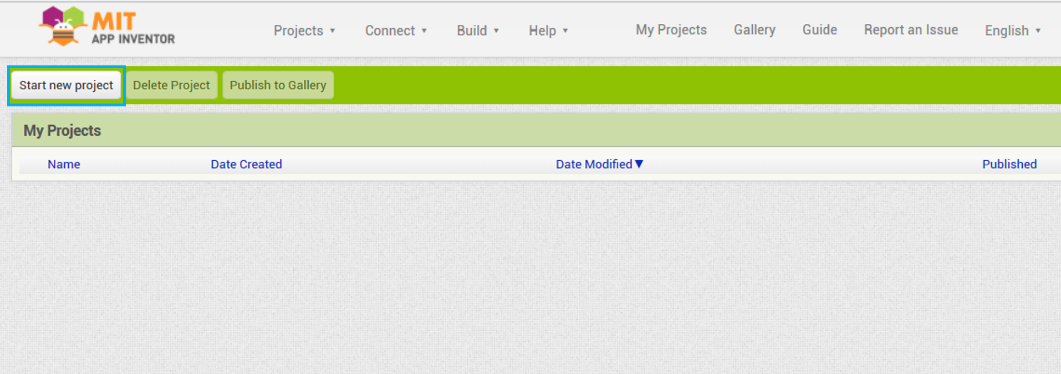
Go to [www.ai2.appinventor.mit.edu](http://www.ai2.appinventor.mit.edu) and click on “Create Apps”



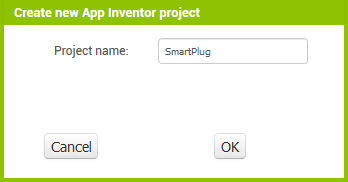
You will need a Google Account to build apps using MIT App Inventor.



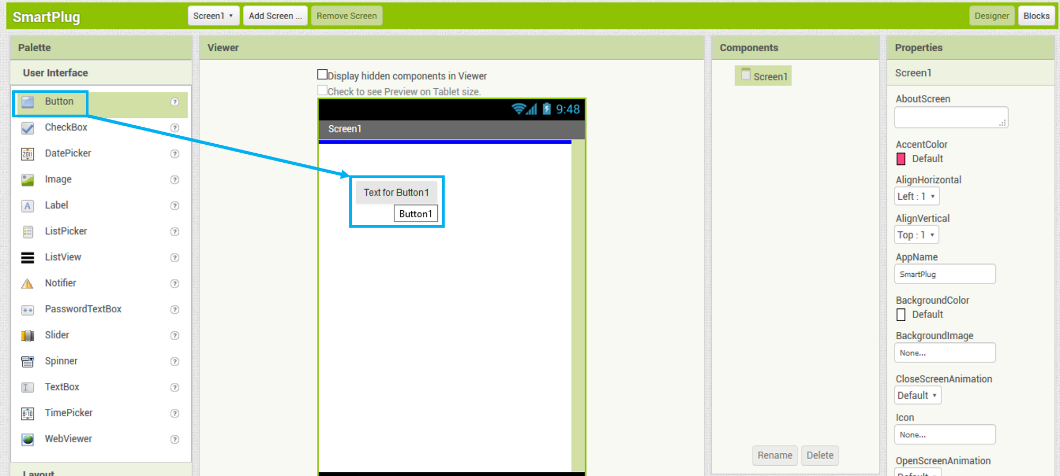
Go to My Projects and click on Start a New project.

****

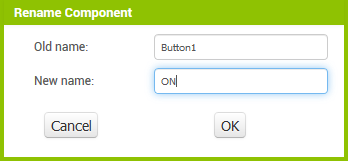
Let’s name the project “SmartPlug”



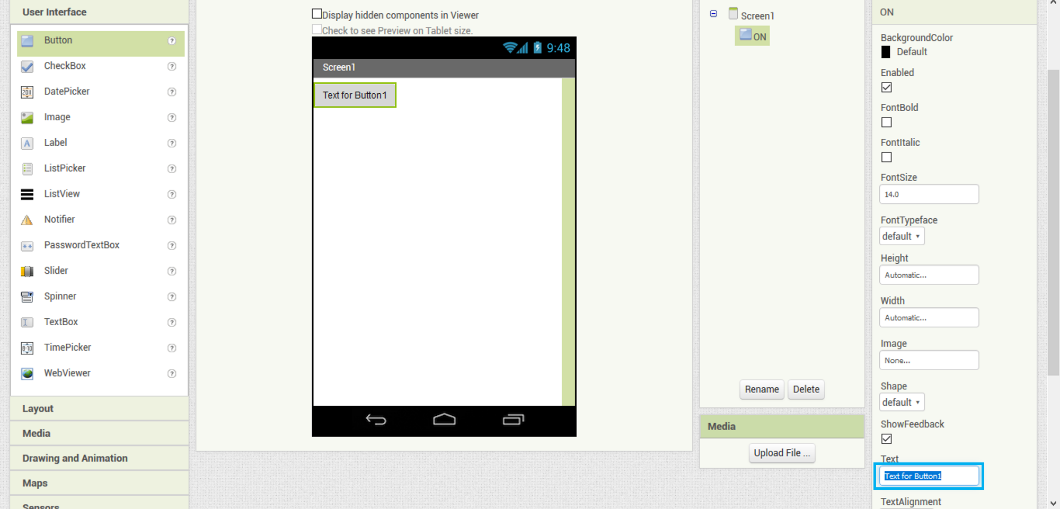
Drag and drop a button from User Interface.

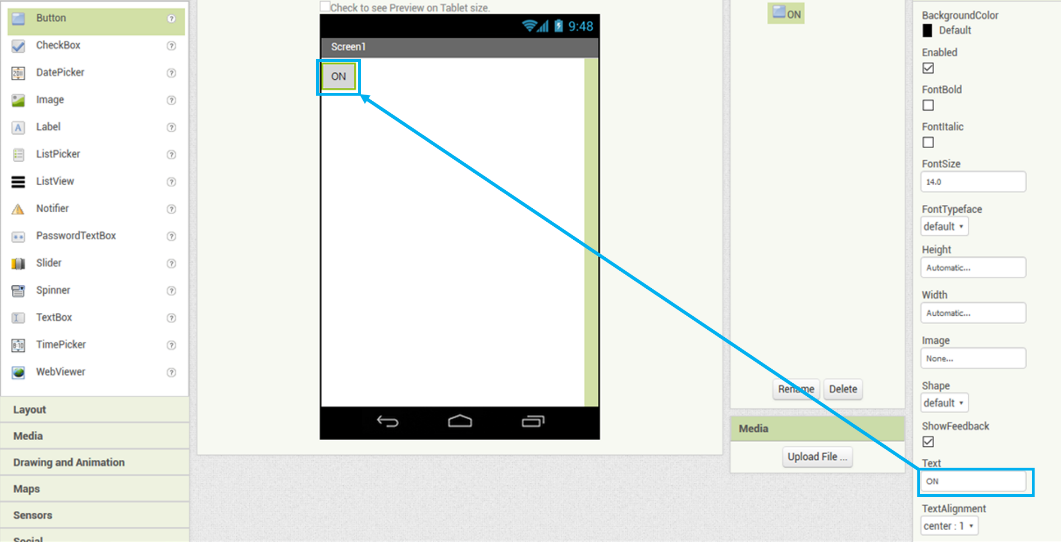


Rename the button to “ON”. This will change their names on the Back End side of the App.

Change the text of the Button from the properties to “ON” to change the button name on the Front End.

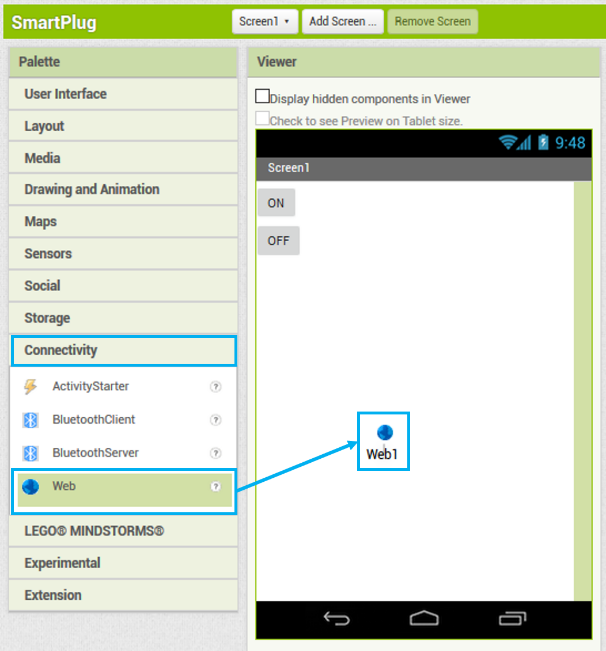




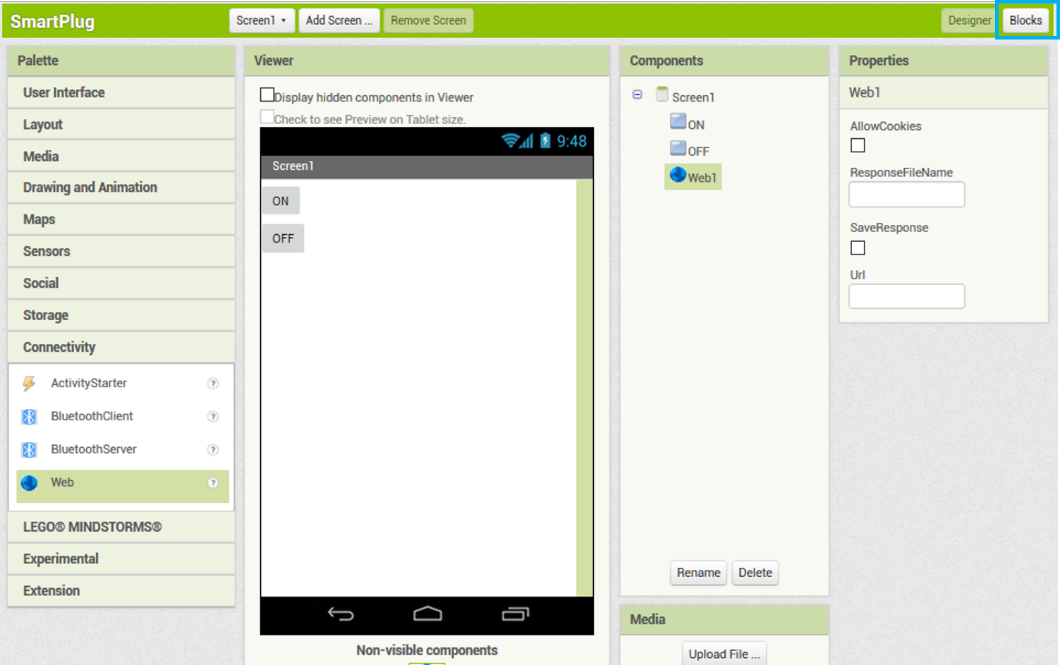
Similarly, add another button and change its name to “OFF” on both Back and Front End.



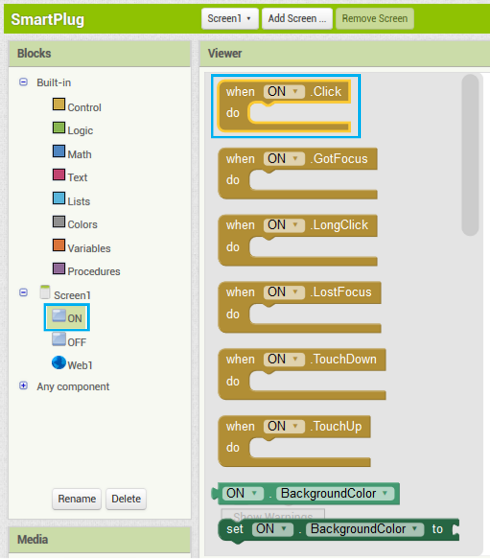
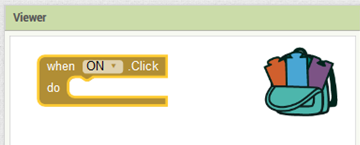
Finally add Web component to the screen from the Connectivity drawer.

****

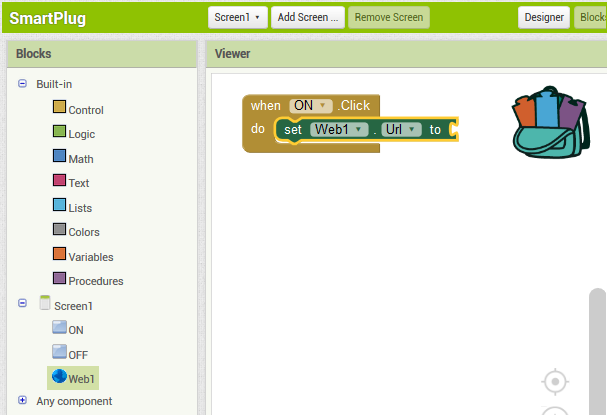
Go to the Block Editor by clicking the blocks button.



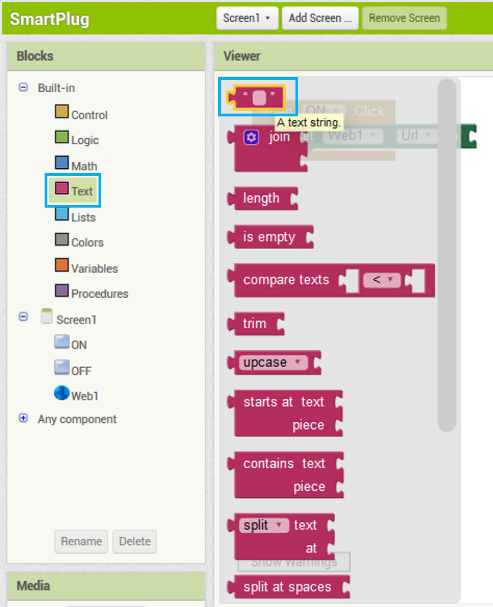
From the ON Button drawer, select **when Button1.Click do** block and drag it in the workspace.

Add a **set Web1.Url** blockand connect it to the **when Button1.Click do** block

Add a **text string** block and connect it to **set Web1.Url** block

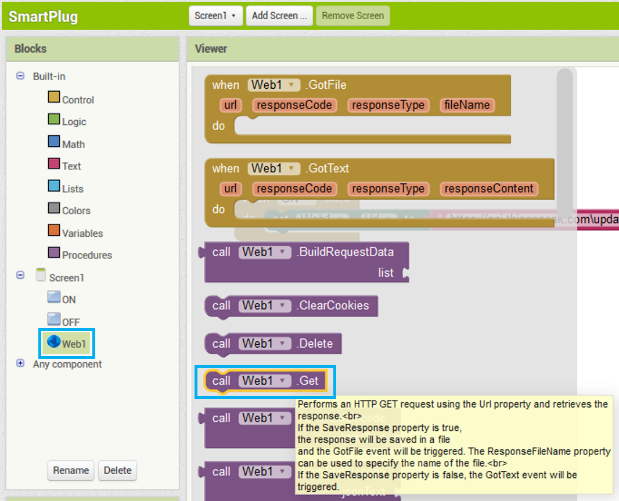


Write the following in the text block:

[https://api.thingspeak.com\update?api\_key=**3MOK54S7MBTBIODA**&field1=1](https://api.thingspeak.com\update?api_key=3MOK54S7MBTBIODA&field1=1)

In place of the word “3MOK54S7MBTBIODA”, write the WRITE API KEY of your own ThingSpeak channel.

Add **call.Web1.get** block to send the http request.





This means that every time you click the “ON” button, the web URL will be set to [https://api.thingspeak.com\update?api\_key=3MOK54S7MBTBIODA &field1=1](https://api.thingspeak.com\update?api_key=KEY&field1=1) and a value of 1 will be sent to the ThingSpeak channel at field1.

Do the same for OFF button. However, a value of 0 will be sent to ThingSpeak instead of 1. The web URL will thus be:

[https://api.thingspeak.com\update?api\_key=**3MOK54S7MBTBIODA**&field1=0](https://api.thingspeak.com\update?api_key=3MOK54S7MBTBIODA&field1=0)

In place of the word “3MOK54S7MBTBIODA”, write the WRITE API KEY of your own ThingSpeak channel.



The app is ready!

### Programming the Control System

We will program NodeMCU using Arduino IDE.

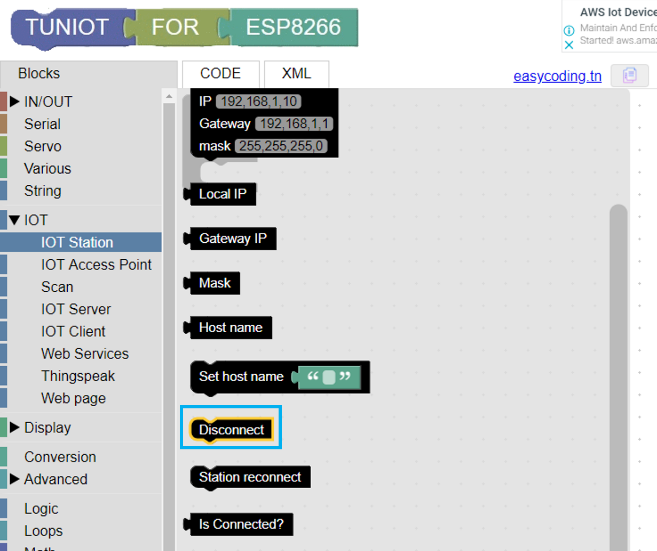
Arduino IDE programs contain 2 types of functions – Setup function and Loop Function.

Setup() is the function that only runs once when the controller (NodeMCU) is reset. Tasks like connecting to wifi, declaring the variables etc will be done here.

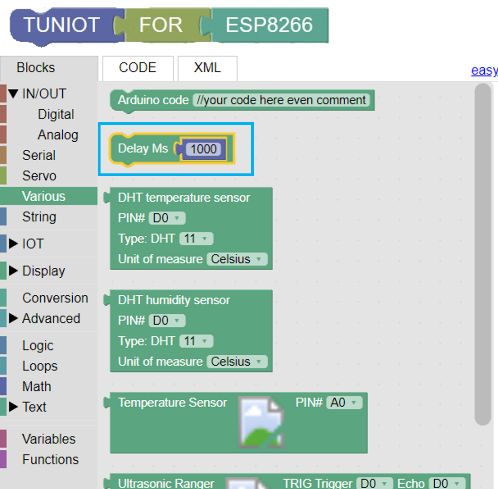
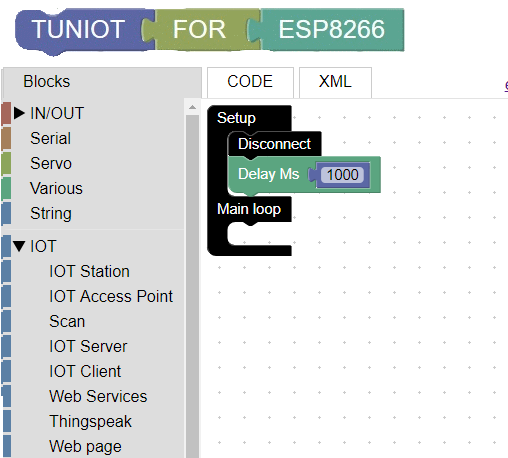
Loop() is the function which runs over and over till NodeMCU continues to receive power. Tasks like reading data from ThingSpeak and signaling the relay will be done here.

For simplicity, the code is build using a block based online application called TUNIOT. Open TUNIOT code generator by following the given link: [http://easycoding.tn/TUNIOT/demos/code/](http://easycoding.tn/tuniot/demos/code/)

From “IOT Station”, click the “Disconnect” block and snap it in the Setup function to disconnect the board from any previous network.

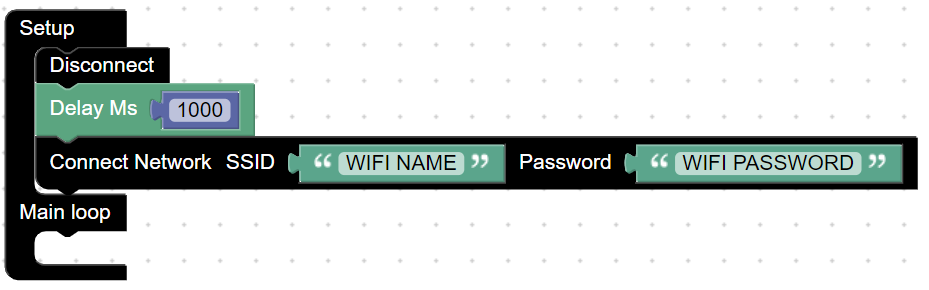
****

From “Various”, click the “Delay Ms” block and snap it in the Setup function under “Disconnect”. This will pause the program execution for 1000 milli seconds (1 second) and allow the board to disconenct from any previous network.

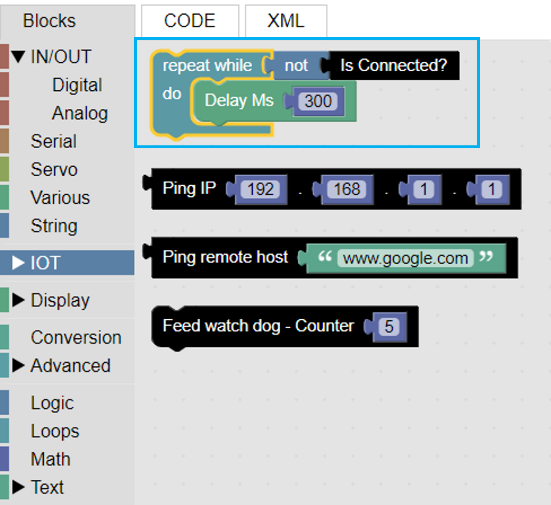
** **

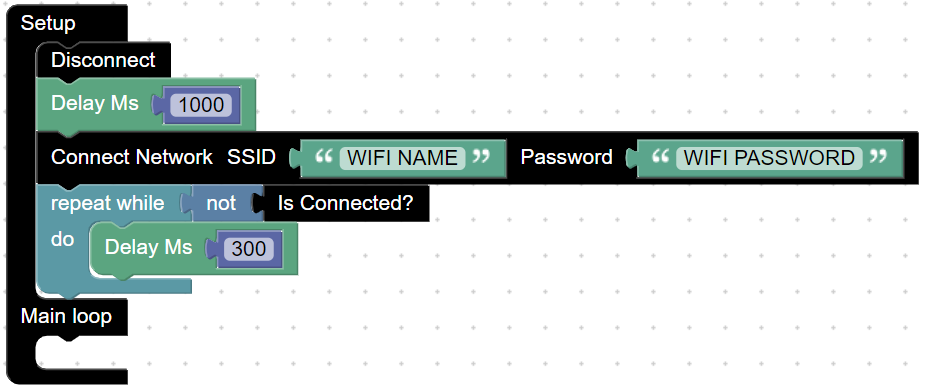
From “IOT Station”, click the “Connect Network” block and snap it in the Setup function to connect the board to your Wi-fi network.   
Enter your Wi-Fi name and password in the SSID and Password textbox respectively. Notice that the names are case sensitive.



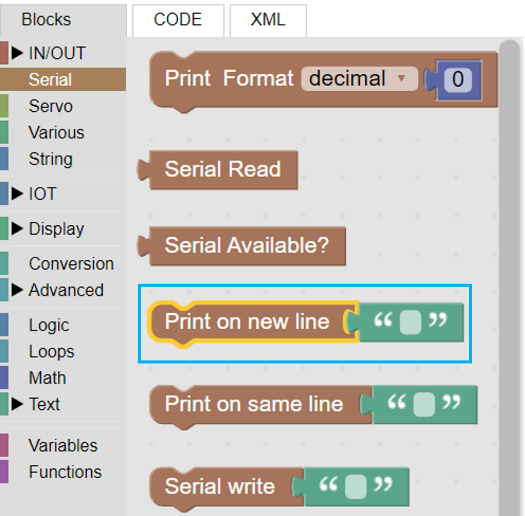


From“IOT station” insert the while loop. This will check if the board is not connected to a network and will wait if the statement is true i.e., it will wait if the board is not connected to the Wi-fi network.

****

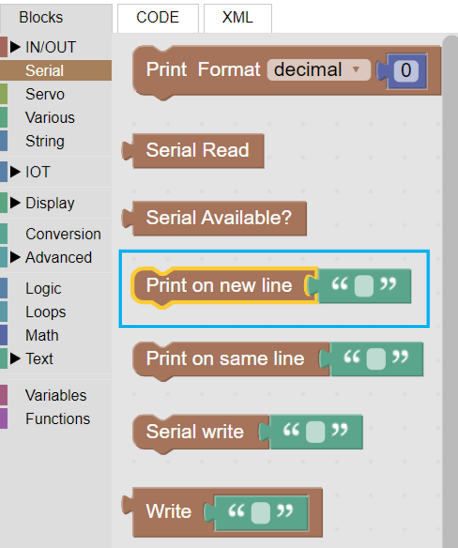


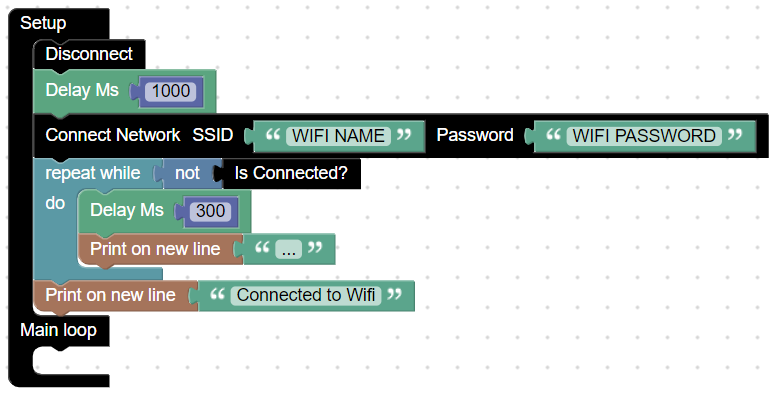
It is better to print something on the Seral Monitor while waiting for NodeMCU to connect to the Wi-fi network. So add “Print on new line” from “IN/OUT Serial” to print dots on the serial monitor while the board connects to Wi-fi.



****

When the board is connected to the network, the loop will be exited. To detect this connection, we will print a response on the serial monitor.





Now that the board is connected to the wifi network, we will read data from the ThingSpeak channel.

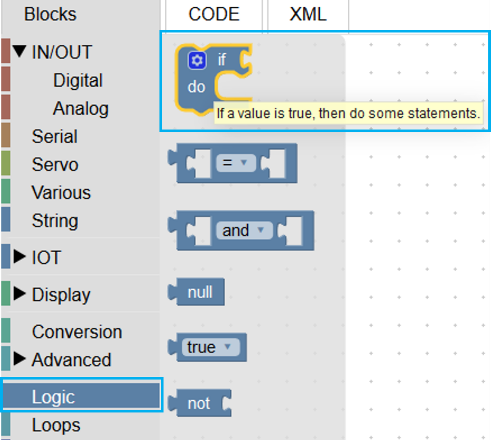
The board will perform this operation continuously so we will add the block in the loop function. Add an “if-do” statement to check if connect NodeMCU can connect to ThingSpeak in the loop function. Delete the “send to ThingSpeak” block.

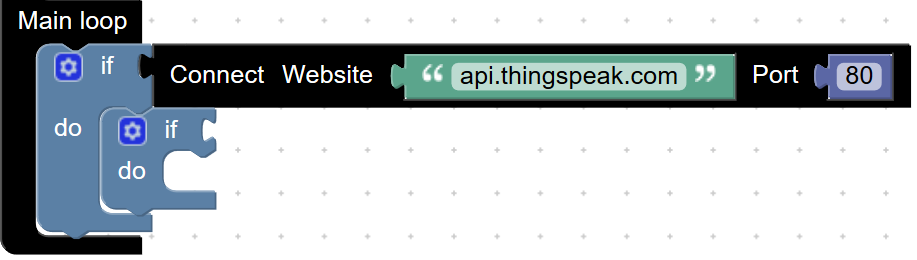
****



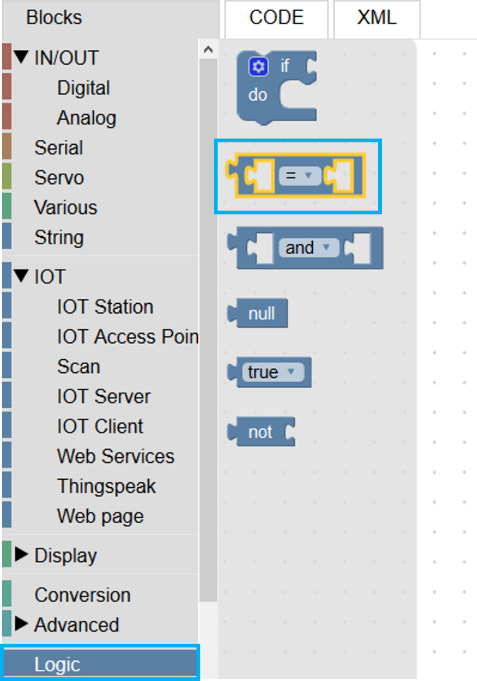
Next we will read data from the ThingSpeak Channel and check it’s value.

Add an “if-do” statement inside the previous “if-do” statement.





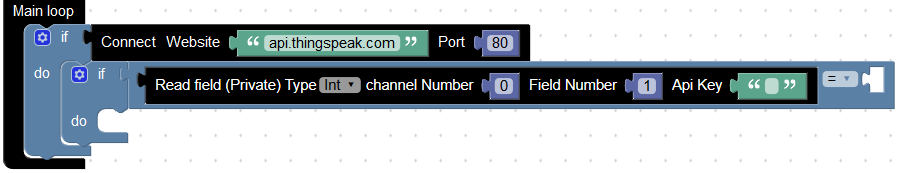
Add a comparison equation from the logic drawer and connect it to the “if-do” statement.



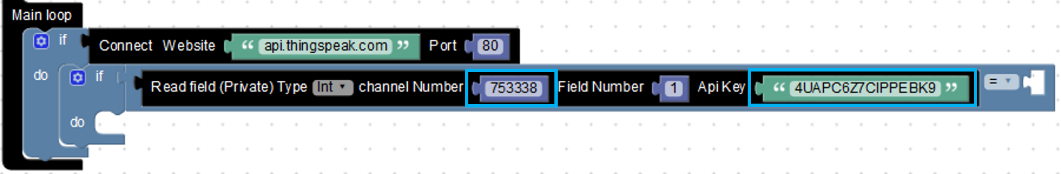


Add a statement to read data from ThingSpeak website from the “IOT – ThingSpeak” drawer and connect it to the equation in the if-do statement.

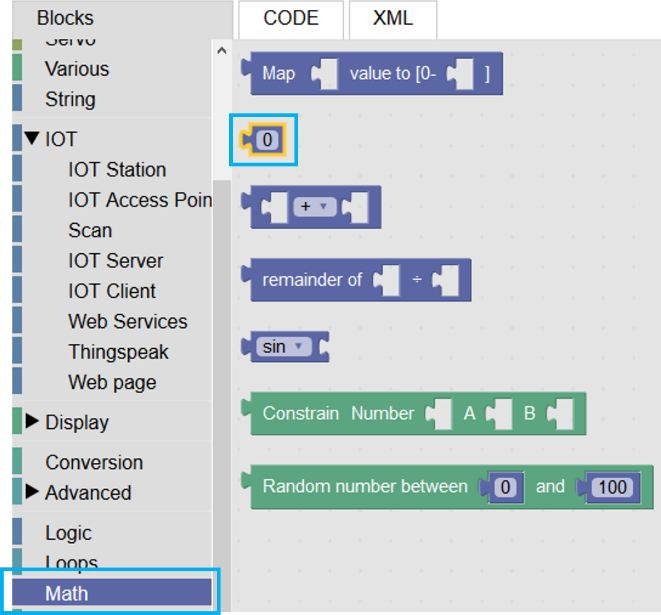
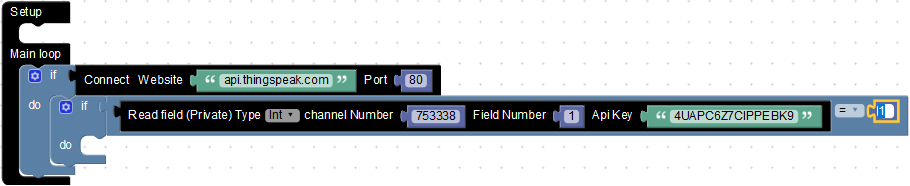




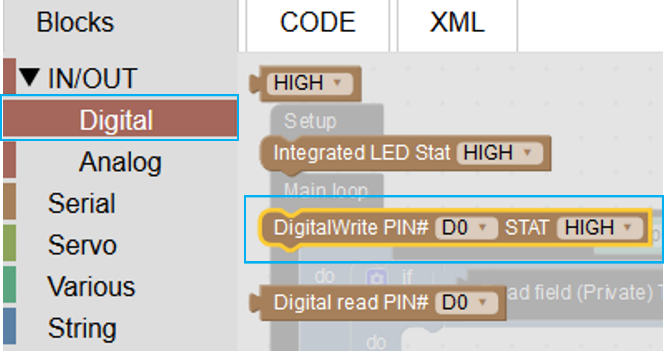
Enter the channel number to read data from, field number to read data from and the Read API key of the channel.

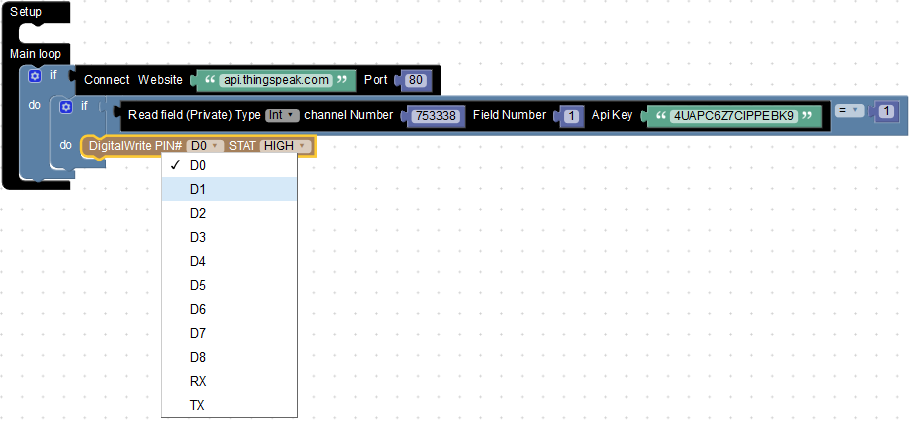


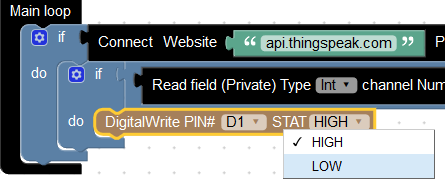
Check if the value of data read from ThingSpeak is 1.

If this data is equal to 1, then set the state of pin number D1 (or any other I/O pin of your preference) as LOW. The relay provided with this kit is a low level trigger relay. This means that when it receives a low signal from NodeMCU, it will switch the circuit between COM and NO on.

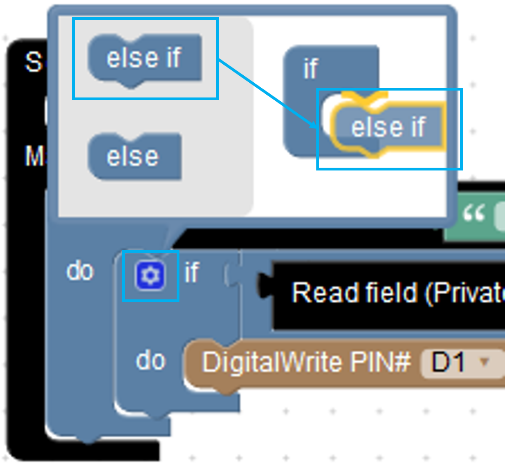


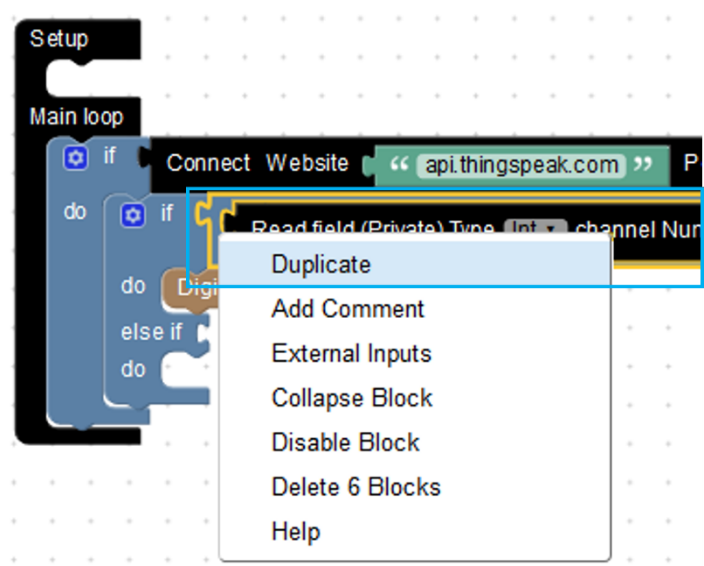


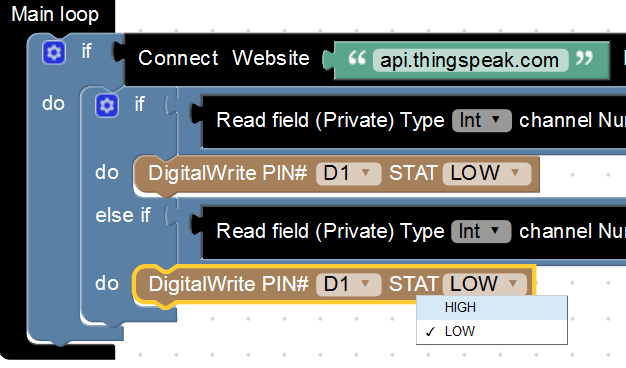


Similarly add another condition to read data from ThingSpeak website. Repeat the channel number to read data from, field number to read data from and the Read API key of the channel. If the data is equal to 0, then set the state of pin number D1 as HIGH. The relay will switch off the circuit between COM and NO on.

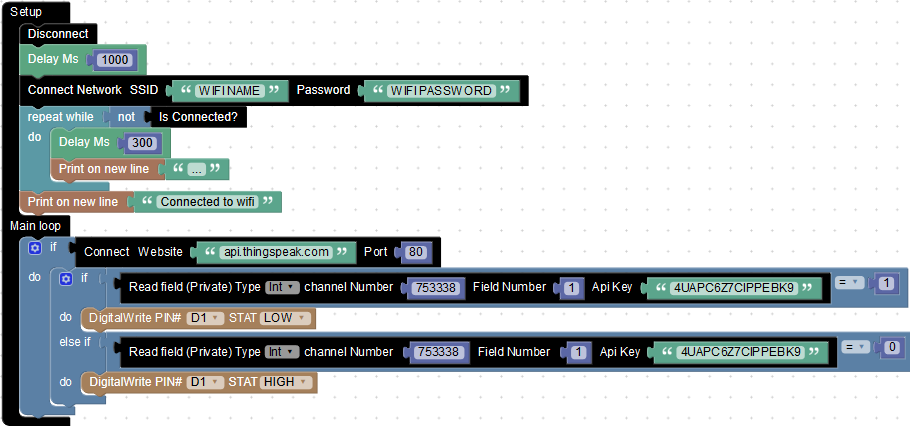
**Step 6:** The logic is complete. Copy the code from TUNIOT by clicking on the copy button and paste it in the Arduino IDE and upload it to NodeMCU board.



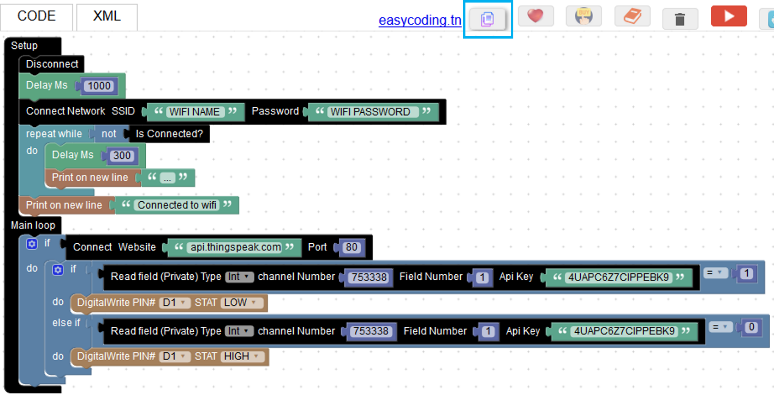


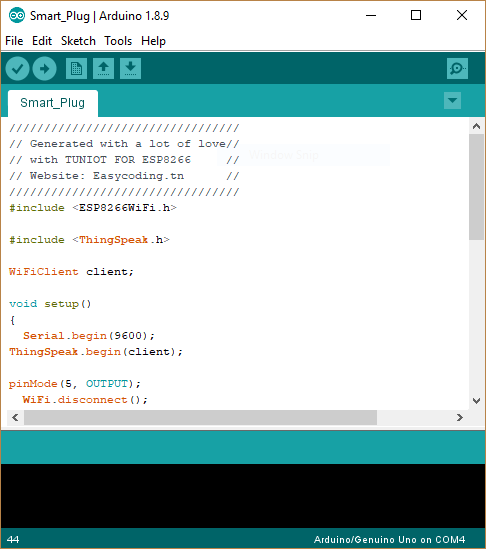


The logic is complete.

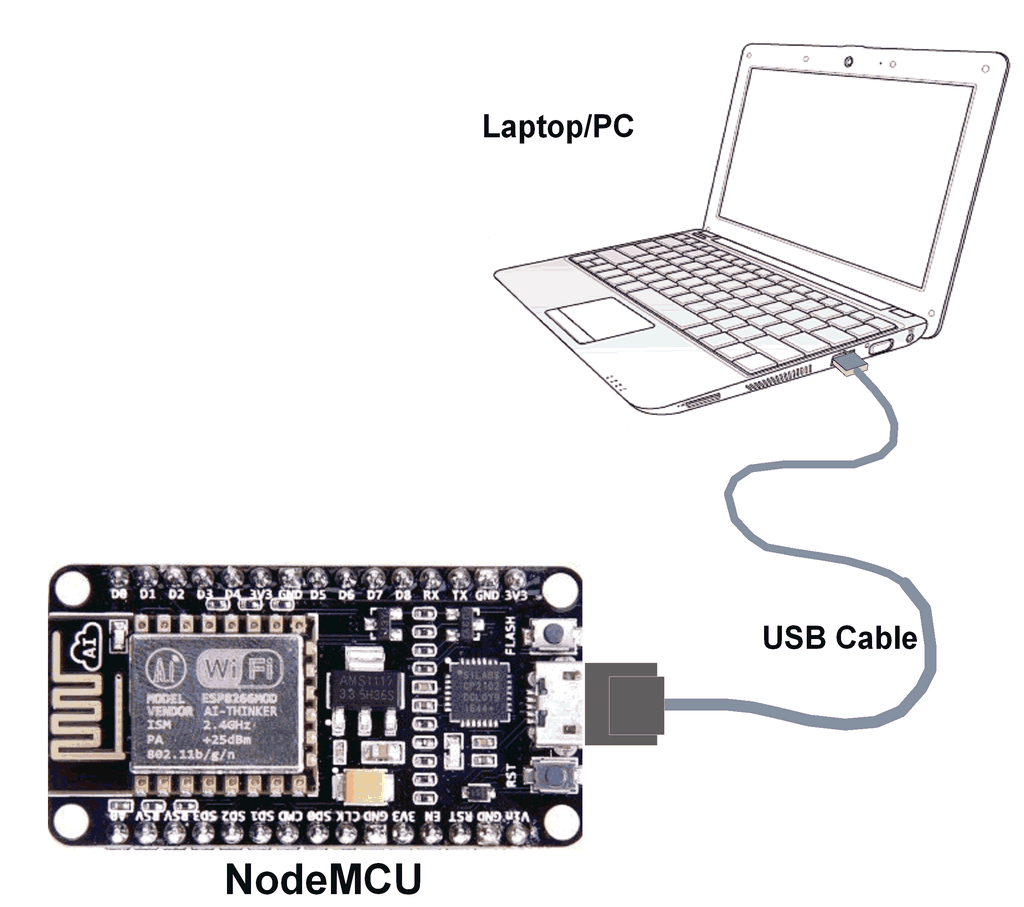


Copy the code from TUNIOT by clicking on the copy button and paste it in the Arduino IDE and upload it to NodeMCU board.





|  |
| --- |
| #include <ESP8266WiFi.h>  #include <**ThingSpeak**.h>  WiFiClient client;  void setup()  {  **Serial**.begin(9600);  **ThingSpeak**.begin(client);   pinMode(16, OUTPUT);  **WiFi**.disconnect();   delay(1000);  **WiFi**.begin("WIFI NAME", "WIFI PASSWORD");   while ((!(**WiFi**.status() == WL\_CONNECTED)))   {     delay(300);  **Serial**.println("...");   }  **Serial**.println("Connected to Wifi");  }  void loop()  {   if (client.connect("api.thingspeak.com", 80)) {     if ((**ThingSpeak**.readIntField(12345, 1, "API KEY")) == 1)     {       digitalWrite(16, LOW);     } else if ((**ThingSpeak**.readIntField(12345, 1, "API KEY")) == 0)     {       digitalWrite(16, HIGH);     }   }  } |

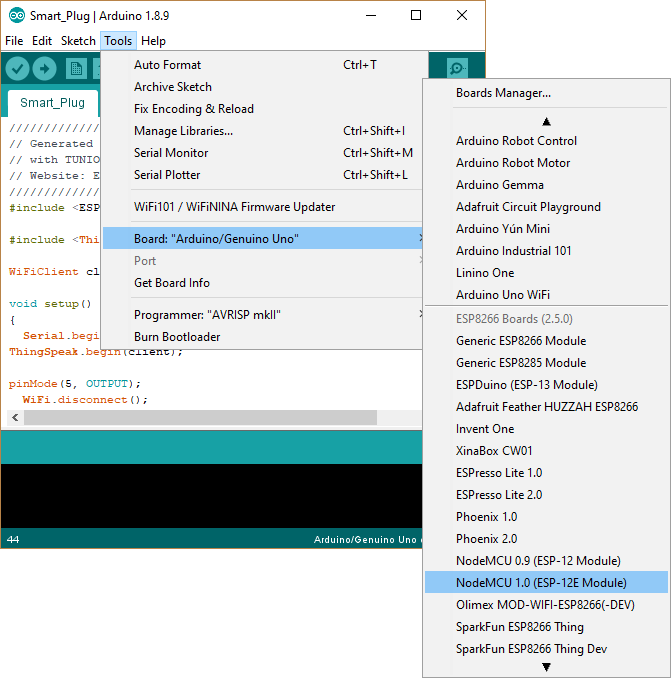


Connect NodeMCU dev Kit with PC as shown.

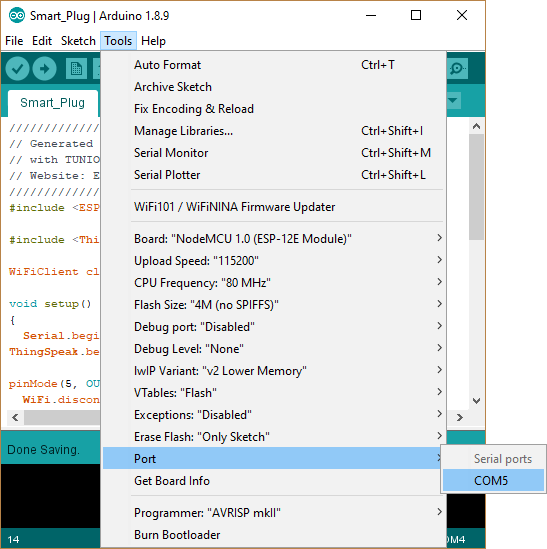
Before you can upload the program to NodeMCU, you will have to add the ESP8266 board to Arduino IDE. To do the same, scan the QR code or follow the link:

<https://www.youtube.com/watch?v=NEo1WsT5T7s>

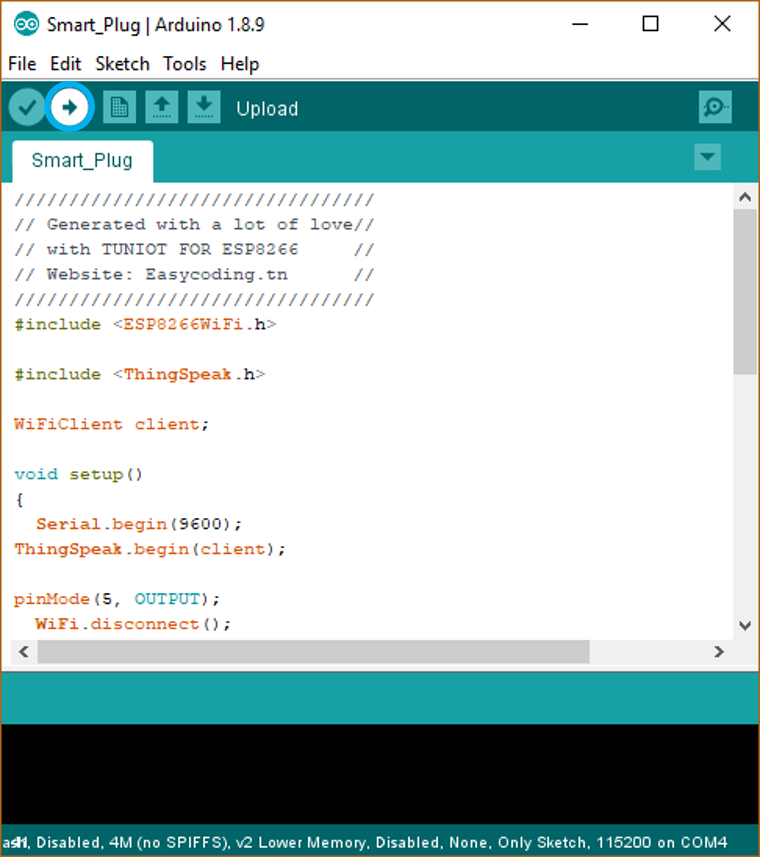
**Once the board has been added to Arduino IDE, open Arduino IDE, go to** Tools->Board->and select NodeMCU 1.0(ESP-12E Module).

****

Make sure that you have selected the appropriate **COM port**. In our system, the NodeMCU is connected at COM5 port. In your PC, the port could be different.

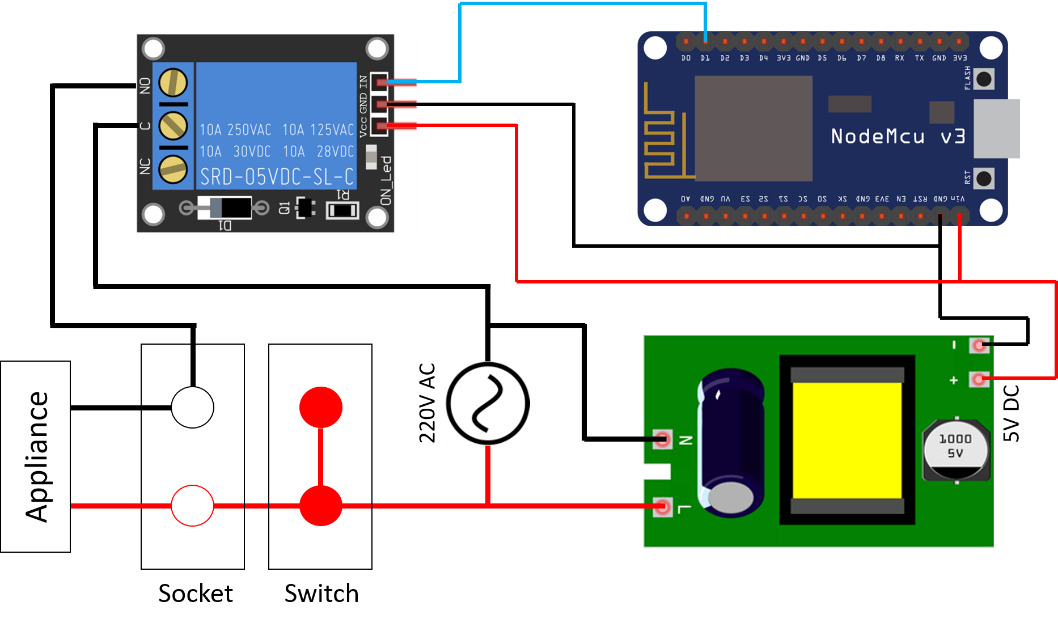
****

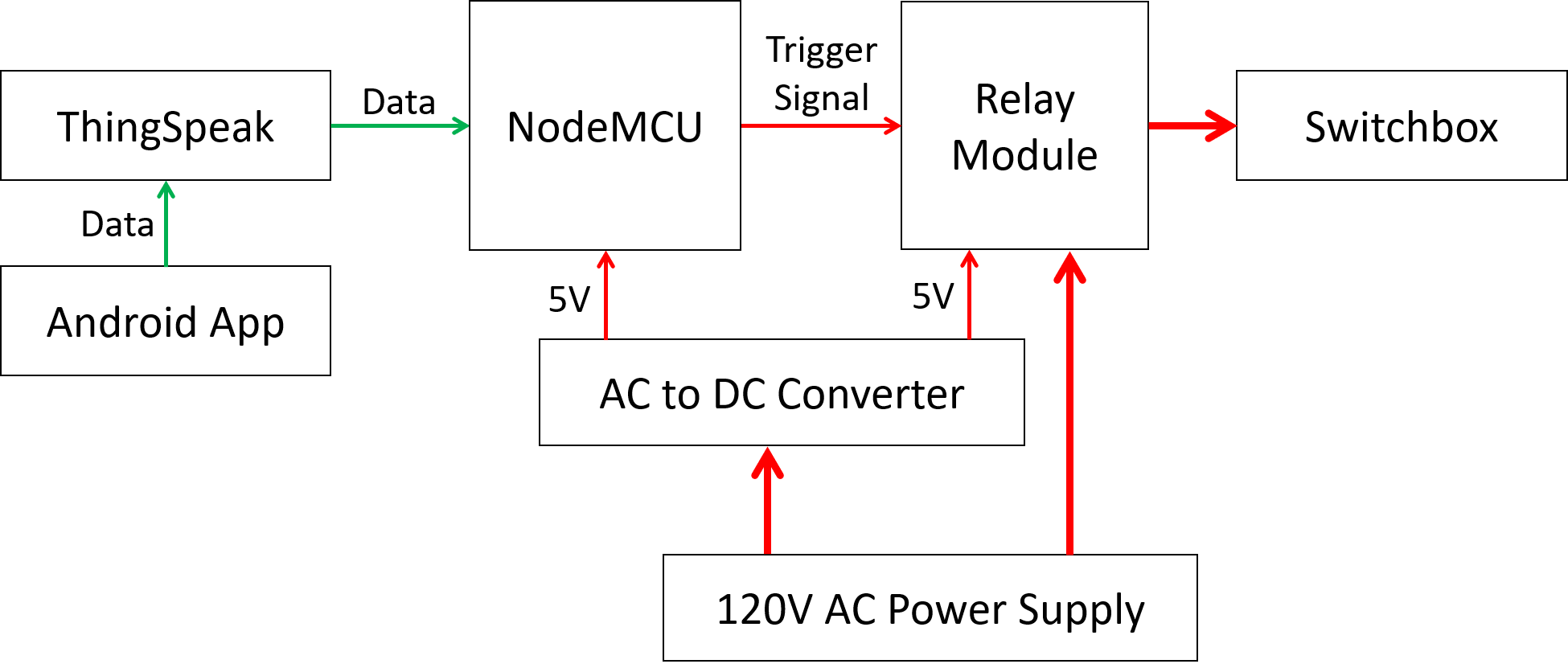
Click the upload button to upload the code to NodeMCU.

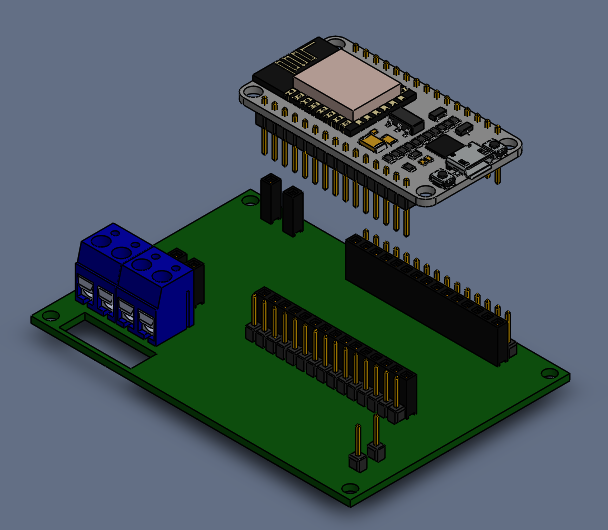
****

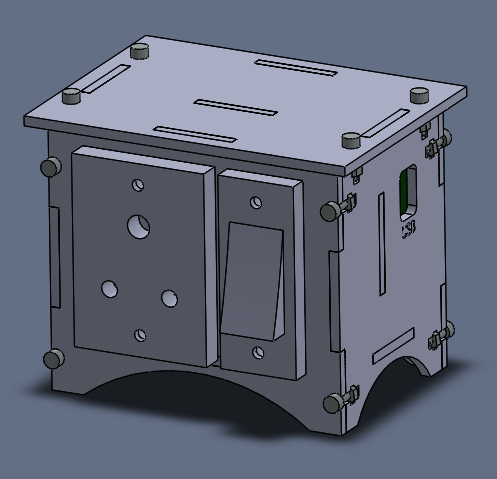
## Hardware Setup: Assembly and Connections

Step 1:

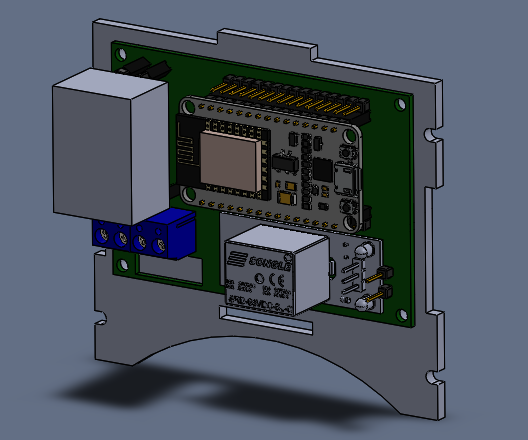








Step 1: Mount the PCB on the plate using fasteners provided



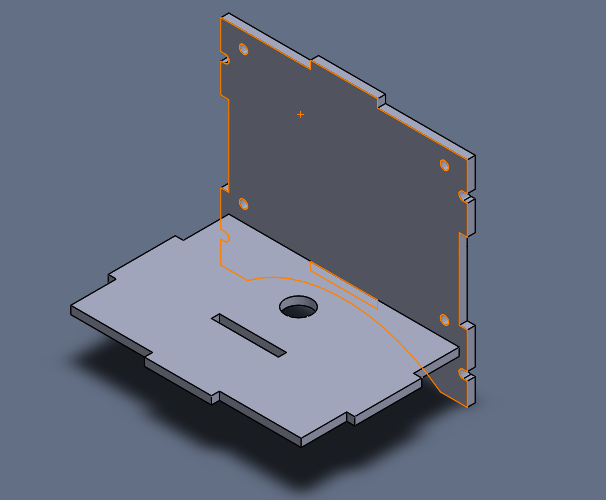
Step 2: Make the connections as shown in the

a) Nodemcu to relay

relay gnd and vcc to relay

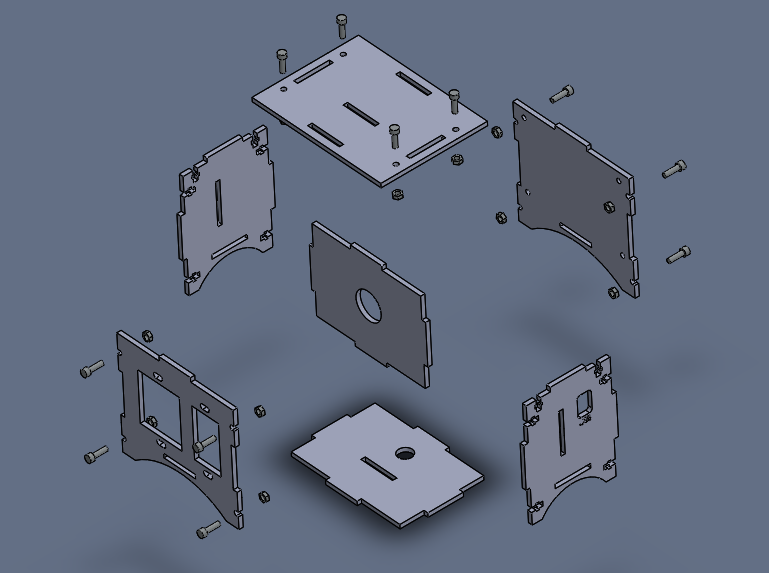
screw terminals to relay module

Step 3: Fix the base of the box to the back panel

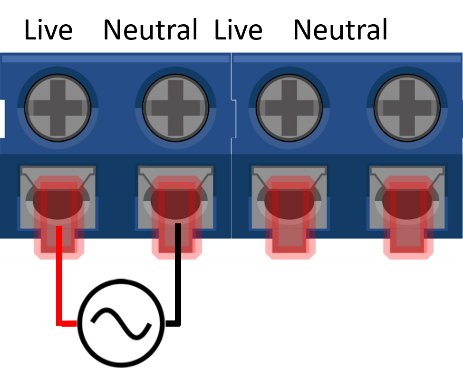


Route the plug and the wire through the hole provided at the base.

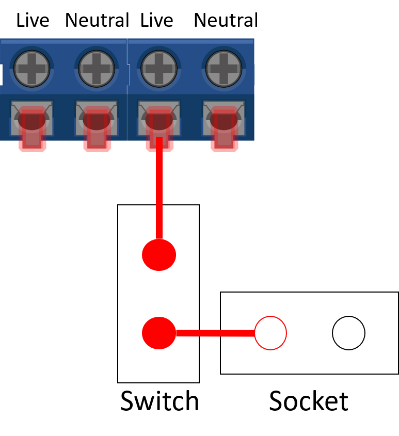
Plug live and neutral to screw terminals



Plug Connections

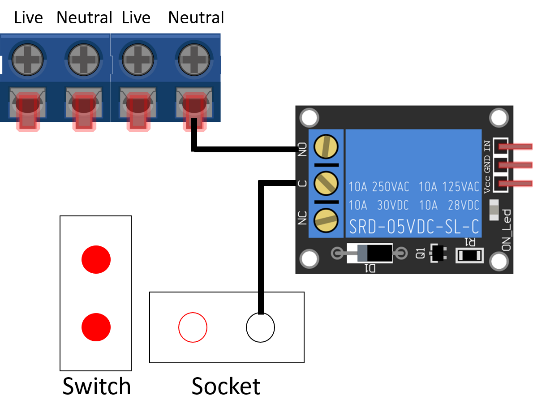
Connect the red and black wires from the plug to the screw terminals marked Live and Neutral.

Switch Connections

Connect one end of the Switch to the screw terminal marked “Live”.

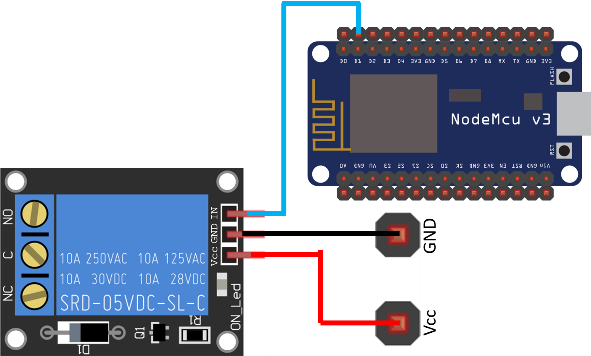
Connect the other end of the screw terminal to the end of the socket marked “L”.

Relay Connections – Output Side

Connect “NO” terminal of the Relay to the screw terminal marked “Neutral”.

Connect “COM” terminal of the Relay to the end of the socket marked “N”.

Relay Connections – Control Side

Connect the “Vcc” pin of Relay to the pin adjacent to it marked “Vcc”.

Connect pin “D1” of NodeMCU to the Relay module pin marked “IN”.

### Box assembly

# Mentornet Resources

Link

Contact us

Articles to read