Vishwakarma Institute of Technology, Pune

Internet of Things: CS2221

AY-2021-2022, Sem-2

Name: Bhavin Ratansing Patil

Div. & Roll No.: CS-D-78

GR No.- 12120056

Experiment No: 1

Installation of Node MCU Using Arduino IDE and Basic LED Blinking

Name of the Student: Bhavin Patil

Div.: D

Roll No.: 78

Aim: To install Node MCU using Arduino IDE and to Blink an On-Board LED on Node MCU

Components Required:

1) Node MCU – 1
2) Micro USB Cable – 1
3) PC/Laptop – 1
4) Connecting Wires
5) Bread Board – 1

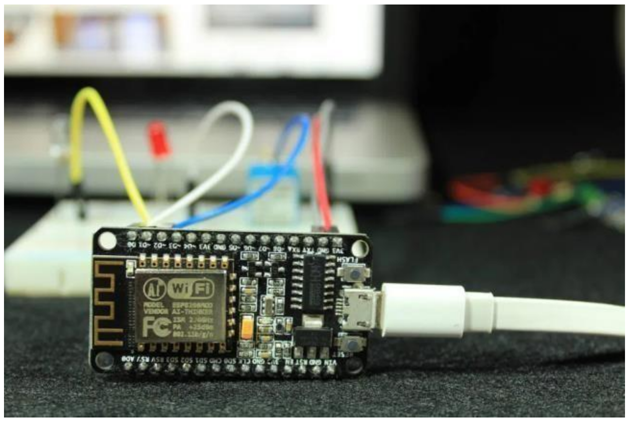
Software Required:

Arduino IDE

Theory:

Today, IOT applications are on the rise, and connecting objects are getting more and more important. There are several ways to connect objects such as Wi-Fi protocol.

Node MCU is an open source platform based on ESP8266 which can connect objects and let data transfer using the Wi-Fi protocol. In addition, by providing some of the most important features of microcontrollers such as GPIO, PWM, ADC, and etc, it can solve many of the project's needs alone.



The general features of this board are as follows:

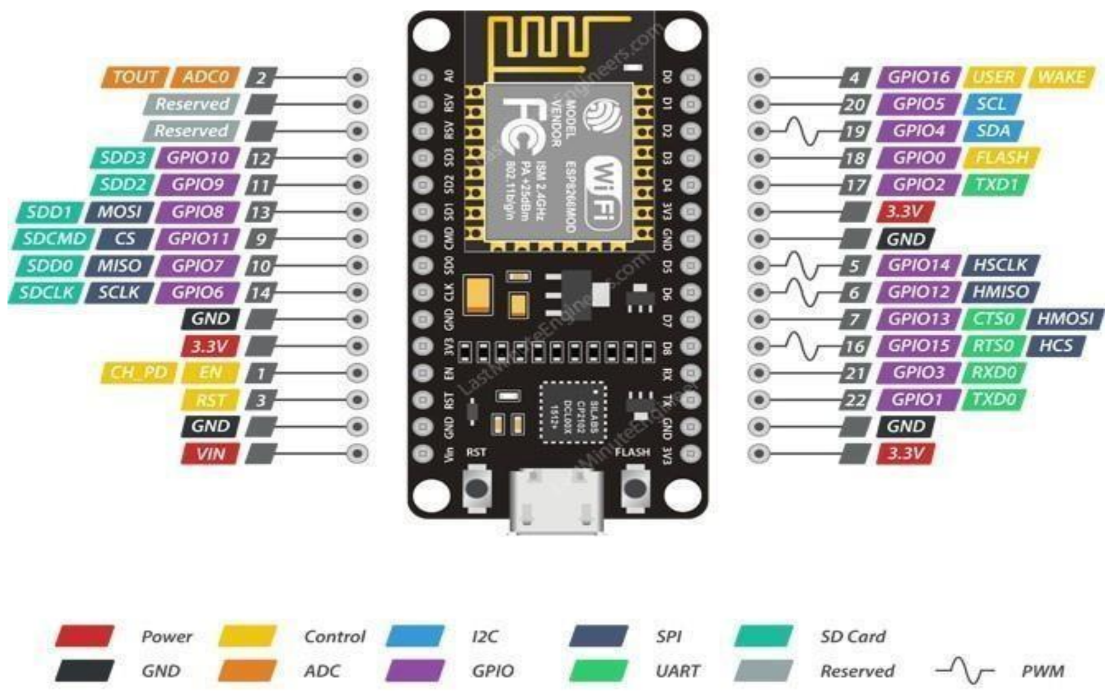
- Easy to use
- Programmability with Arduino IDE or IUA languages
- Available as an access point or station
- Practicable in Event-driven API applications
- Having an internal antenna
- Containing 13 GPIO pins, 10 PWM channels, I2C, SPI, ADC, UART, and 1-Wire

ESP8266 Specifications:

- 11 b/g/n protocol
- Wi-Fi Direct (P2P), soft-AP
- Integrated TCP/IP protocol stack
- Built-in low-power 32-bit CPU
- SDIO 2.0, SPI, UART

ESP 8266 Pin Out:

Pin Description:



Power Pin: There are four power pins viz. one VIN pin & three 3.3V pins. The VIN pin can be used to directly supply the ESP8266 and its peripherals, if you have a regulated 5V voltage source. The 3.3V pins are the output of an on-board voltage regulator. These pins can be used to supply power to external components

GND: GND is a ground pin of ESP8266 Node MCU development board.

I2C Pins: I2C Pins are used to hook up all sorts of I2C sensors and peripherals in your project. Both I2C Master and I2C Slave are supported. I2C interface functionality can be realized programmatically, and the clock frequency is 100 kHz at a maximum. It should be noted that I2C clock frequency should be higher than the slowest clock frequency of the slave device

GPIO Pins: ESP8266 Node MCU has 17 GPIO pins which can be assigned to various functions such as I2C, I2S, UART, PWM, IR Remote Control, LED Light and Button programmatically. Each digital enabled GPIO can be configured to internal pull-up or pull-down, or set to high impedance. When configured as an input, it can also be set to edge-trigger or level-trigger to generate CPU interrupts.

ADC Channel: The Node MCU is embedded with a 10-bit precision SAR ADC. The two functions can be implemented using ADC viz. Testing power supply voltage of VDD3P3 pin and testing input voltage of TOUT pin. However, they cannot be implemented at the same time.

UART Pins: ESP8266 Node MCU has 2 UART interfaces, i.e. UART0 and UART1, which provide asynchronous communication (RS232 and RS485), and can communicate at up to 4.5 Mbps. UART0 (TXD0, RXD0, RST0 & CTS0 pins) can be used for communication. It supports fluid control. However, UART1 (TXD1 pin) features only data transmit signal so, it is usually used for printing log.

SPI Pins: ESP8266 features two SPIs (SPI and HSPI) in slave and master modes. These SPIs also support the following general-purpose SPI features:

- 4 timing modes of the SPI format transfer.
- Up to 80 MHz and the divided clocks of 80 MHz.
- Up to 64-Byte FIFO.

SDIO Pins: ESP8266 features Secure Digital Input/output Interface (SDIO) which is used to directly interface SD cards. 4-bit 25 MHz SDIO v1.1 and 4-bit 50 MHz SDIO v2.0 are supported

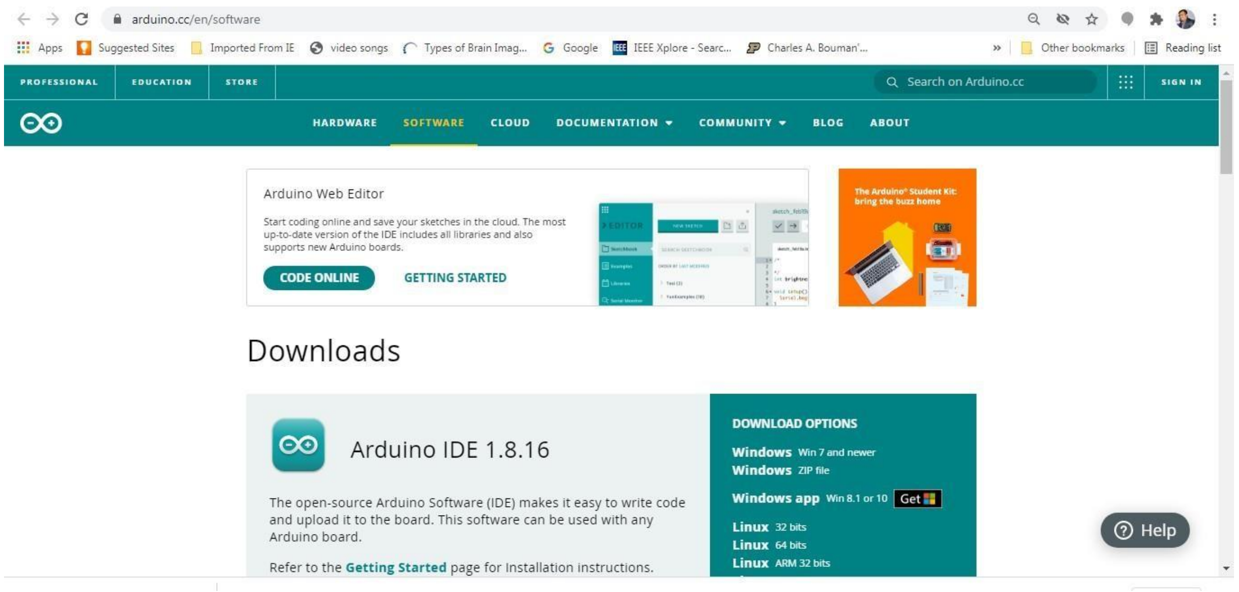
PWM Pins: The board has 4 channels of Pulse Width Modulation (PWM). The PWM output can be implemented programmatically and used for driving digital motors and LEDs. PWM frequency range is adjustable from 1000 μs to 10000 μs, i.e., between 100 Hz and 1 kHz.
Control Pins: Control Pins are used to control ESP8266. These pins include Chip Enable pin (EN), Reset pin (RST) and WAKE pin. EN pin – The ESP8266 chip is enabled when EN pin is pulled HIGH. When pulled LOW the chip works at minimum power. RST pin – RST pin is used to reset the ESP8266 chip. WAKE pin – Wake pin is used to wake the chip from deep-sleep.

Procedure:

1. Installing ESP8266 Board in Arduino IDE:

The ESP8266 community created an add-on for the Arduino IDE that allows you to program the ESP8266 using the Arduino IDE and its programming language.

Step 1: Visit arduino.cc/en/Main/Software for downloading the latest version of Arduino IDE software [Arduino IDE 1.8.16]



Arduino Web Editor

Start coding online and save your sketches in the cloud. The most up-to-date version of the IDE includes all libraries and also supports new Arduino boards.

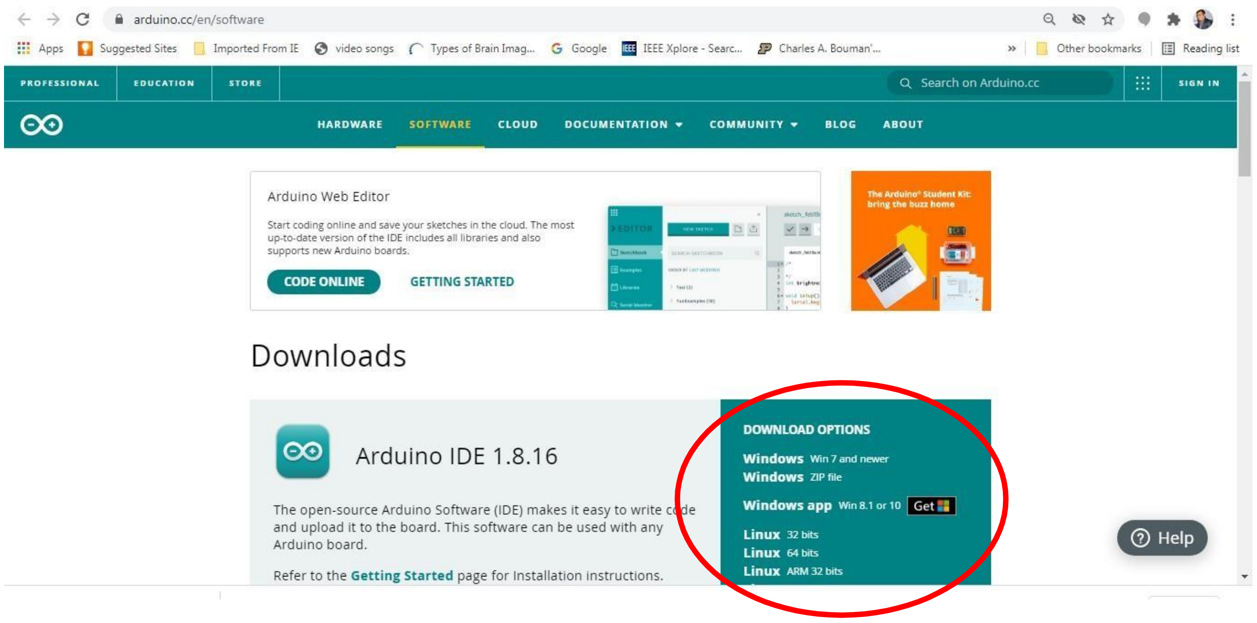
Downloads

Arduino IDE 1.8.16

The open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. This software can be used with any Arduino board.

Refer to the Getting Started page for Installation instructions.

Step 2: Click on appropriate download link as per the operating system installed on your PC / laptop



Arduino Web Editor

Start coding online and save your sketches in the cloud. The most up-to-date version of the IDE includes all libraries and also supports new Arduino boards.

Downloads

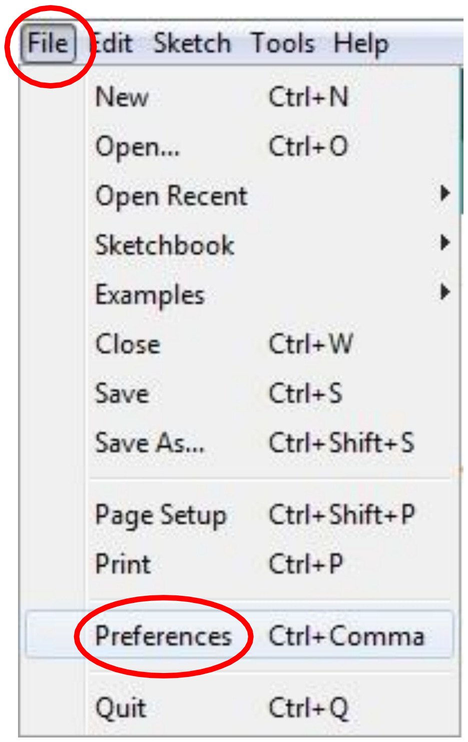
Arduino IDE 1.8.16

The open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. This software can be used with any Arduino board.

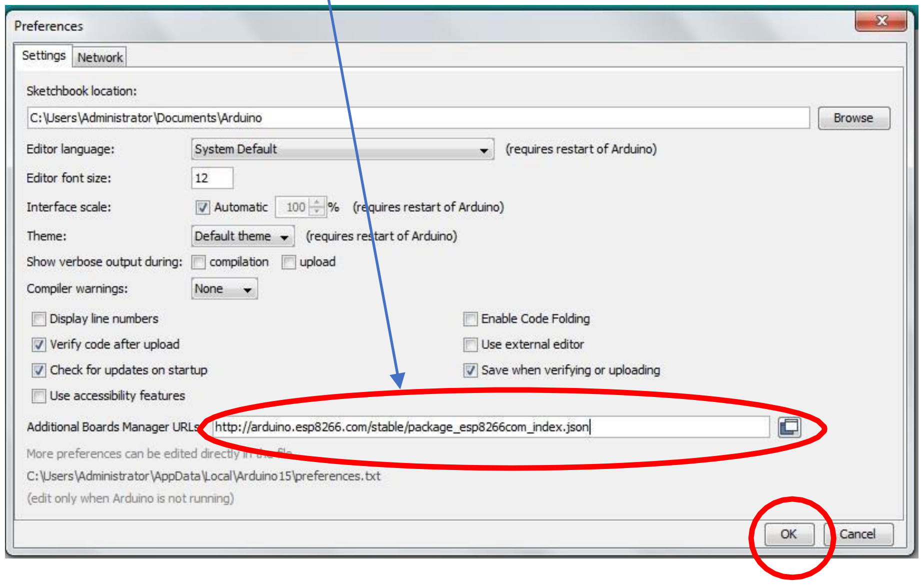
Refer to the Getting Started page for Installation instructions.

Step 3: Download and install it on your PC / Laptop

Step 4: Install ESP 8266 Add on in Arduino IDE. Open Arduino IDE ----> File ---- > Preferences



Step 5: Enter **http://arduino.esp8266.com/stable/package_esp8266com_index.json** into the "Additional Boards Manager URLs" field as shown in the figure below. Then, click the "OK" button



Preferences

Settings | Network

Sketchbook location: C:\Users\Administrator\Documents\Arduino (Browse)

Editor language: System Default (requires restart of Arduino)

Editor font size: 12

Interface scale: ✓ Automatic 100 % (requires restart of Arduino)

Theme: Default theme (requires restart of Arduino)

Show verbose output during: ☐ compilation ☐ upload

Compiler warnings: None

☐ Display line numbers

✓ Verify code after upload

✓ Check for updates on startup

☐ Use accessibility features

☐ Enable Code Folding

☐ Use external editor

✓ Save when verifying or uploading

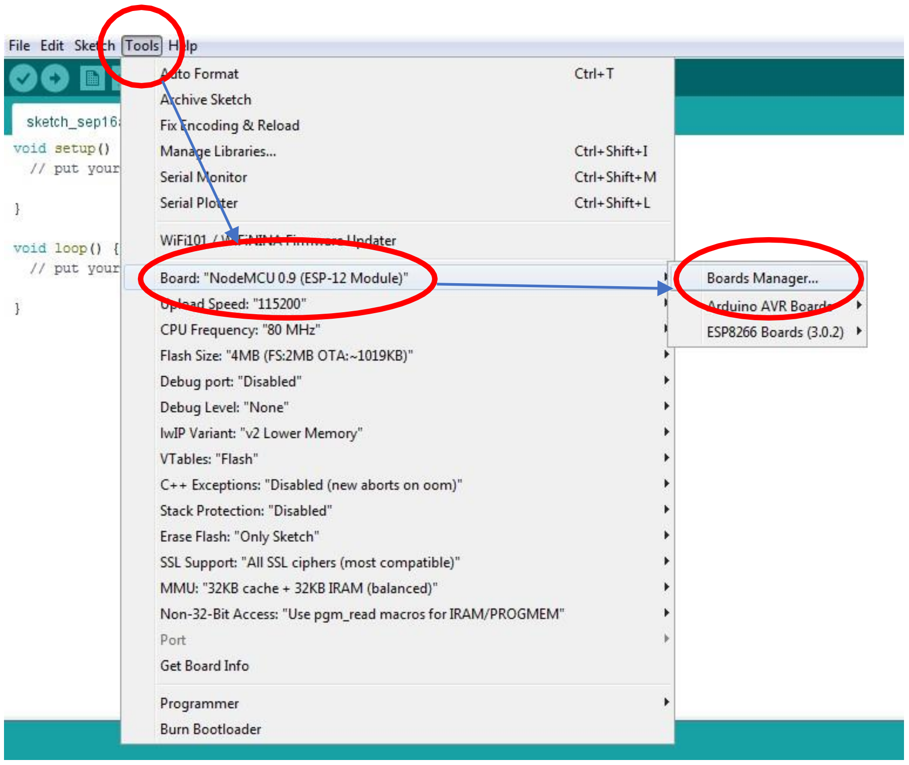
Additional Boards Manager URLs: http://arduino.esp8266.com/stable/package_esp8266com_index.json

More preferences can be edited directly in the file

C:\Users\Administrator\AppData\Local\Arduino15\preferences.txt

(edit only when Arduino is not running)

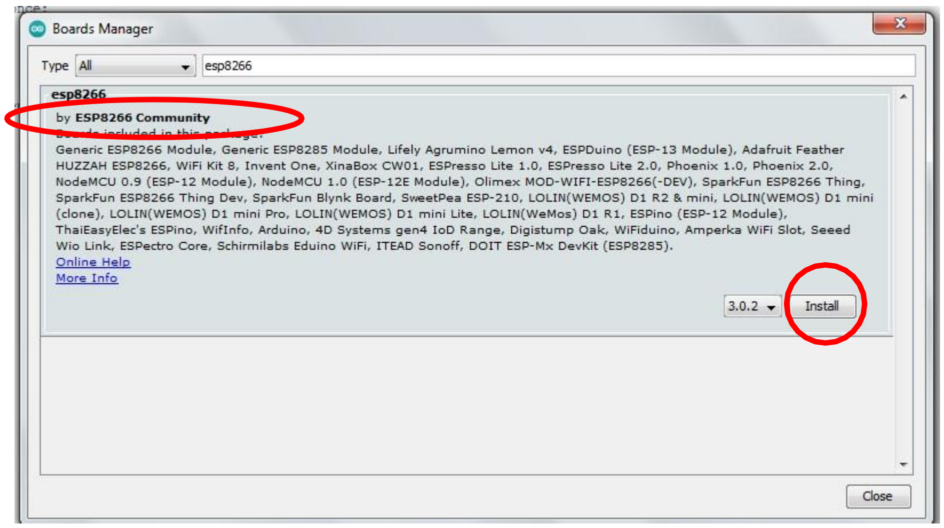
Step 6: Open the Boards Manager. Go to Tools---- > Board ---- > Boards Manager…



```
void setup() {
  // put your

}

void loop() {
  // put your

}
```

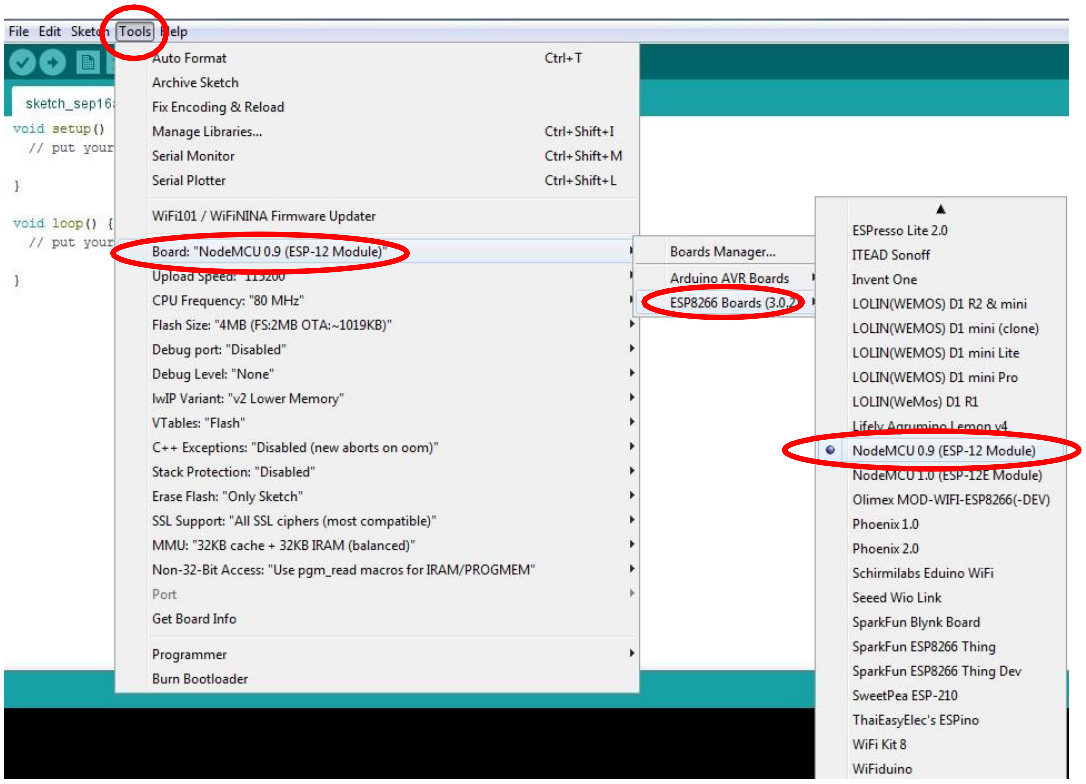
```
Auto Format                          Ctrl+T
Archive Sketch
Fix Encoding & Reload
Manage Libraries...                  Ctrl+Shift+I
Serial Monitor                       Ctrl+Shift+M
Serial Plotter                       Ctrl+Shift+L
WiFi101 / WiFiNINA Firmware Updater
Board: "NodeMCU 0.9 (ESP-12 Module)"
Upload Speed: "115200"
CPU Frequency: "80 MHz"
Flash Size: "4MB (FS:2MB OTA:~1019KB)"
Debug port: "Disabled"
Debug Level: "None"
lwIP Variant: "v2 Lower Memory"
VTables: "Flash"
C++ Exceptions: "Disabled (new aborts on oom)"
Stack Protection: "Disabled"
Erase Flash: "Only Sketch"
SSL Support: "All SSL ciphers (most compatible)"
MMU: "32KB cache + 32KB IRAM (balanced)"
Non-32-Bit Access: "Use pgm_read macros for IRAM/PROGMEM"
Port
Get Board Info
Programmer
Burn Bootloader
```

Step 7: Search for **ESP8266** and press install button for the **ESP8266 by ESP8266 Community**



esp8266
by **ESP8266 Community**
Boards included in this package:
Generic ESP8266 Module, Generic ESP8285 Module, Lifely Agrumino Lemon v4, ESPDuino (ESP-13 Module), Adafruit Feather HUZZAH ESP8266, WiFi Kit 8, Invent One, XinaBox CW01, ESPresso Lite 1.0, ESPresso Lite 2.0, Phoenix 1.0, Phoenix 2.0, NodeMCU 0.9 (ESP-12 Module), NodeMCU 1.0 (ESP-12E Module), Olimex MOD-WIFI-ESP8266(-DEV), SparkFun ESP8266 Thing, SparkFun ESP8266 Thing Dev, SparkFun Blynk Board, SweetPea ESP-210, LOLIN(WEMOS) D1 R2 & mini, LOLIN(WEMOS) D1 mini (clone), LOLIN(WEMOS) D1 mini Pro, LOLIN(WEMOS) D1 mini Lite, LOLIN(WeMos) D1 R1, ESPino (ESP-12 Module), ThaiEasyElec's ESPino, WifInfo, Arduino, 4D Systems gen4 IoD Range, Digistump Oak, WiFiduino, Amperka WiFi Slot, Seeed Wio Link, ESPectro Core, Schirmilabs Eduino WiFi, ITEAD Sonoff, DOIT ESP-Mx DevKit (ESP8285).
Online Help
More Info

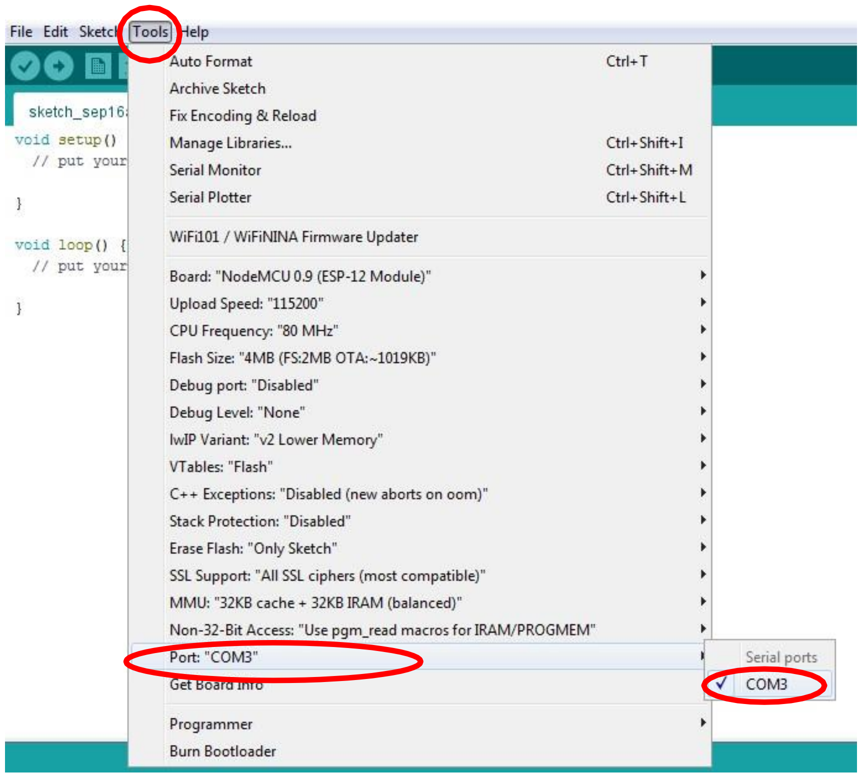
Step 8: Go to Tools ----> Board ----> ESP 8266 Boards (3.0.2) ---- > Select Node MCU 0.9(ESP-12 module)



```
void setup() {
  // put your

}

void loop() {
  // put your

}
```

Board: "NodeMCU 0.9 (ESP-12 Module)"
Upload Speed: "115200"
CPU Frequency: "80 MHz"
Flash Size: "4MB (FS:2MB OTA:~1019KB)"
Debug port: "Disabled"
Debug Level: "None"
IwIP Variant: "v2 Lower Memory"
VTables: "Flash"
C++ Exceptions: "Disabled (new aborts on oom)"
Stack Protection: "Disabled"
Erase Flash: "Only Sketch"
SSL Support: "All SSL ciphers (most compatible)"
MMU: "32KB cache + 32KB IRAM (balanced)"
Non-32-Bit Access: "Use pgm_read macros for IRAM/PROGMEM"

Step 9: Go to Tools ----> Port ----- > Select Serial Port COM3


File Edit Sketch Tools Help
sketch_sep16a
Auto Format Ctrl+T
Archive Sketch
Fix Encoding & Reload
Manage Libraries... Ctrl+Shift+I
Serial Monitor Ctrl+Shift+M
Serial Plotter Ctrl+Shift+L
WiFi101 / WiFiNINA Firmware Updater
Get Board Info
Programmer
Burn Bootloader
Serial ports
✓ COM3

```
void setup() {
  // put your

}

void loop() {
  // put your

}
```

Board: "NodeMCU 0.9 (ESP-12 Module)"
Upload Speed: "115200"
CPU Frequency: "80 MHz"
Flash Size: "4MB (FS:2MB OTA:~1019KB)"
Debug port: "Disabled"
Debug Level: "None"
IwIP Variant: "v2 Lower Memory"
VTables: "Flash"
C++ Exceptions: "Disabled (new aborts on oom)"
Stack Protection: "Disabled"
Erase Flash: "Only Sketch"
SSL Support: "All SSL ciphers (most compatible)"
MMU: "32KB cache + 32KB IRAM (balanced)"
Non-32-Bit Access: "Use pgm_read macros for IRAM/PROGMEM"
Port: "COM3"

2. Testing the Installation [Basic LED Blinking]:

On-board LED

Most of the ESP8266 development boards have two built-in LEDs. These LEDs are usually connected to GPIO D0 and D4.

Step 1: Connect Node MCU to PC / Laptop with the help of micro USB cable

Step 2: Open new Sketch, Go to file ---- > New
Step 3: Write following code in new sketch

```
#define LED_BUILTIN D4

// the setup function runs once when you press reset or power the board void

setup() {

  pinMode(LED_BUILTIN, OUTPUT); // initialize digital pin LED_BUILTIN as an output. }

// the loop function runs over and over again forever void

loop() {

  digitalWrite(LED_BUILTIN, HIGH); // turn the LED on (HIGH is the voltage level)

  delay(1000);                     // wait for a second

  digitalWrite(LED_BUILTIN, LOW);    // turn the LED off by making the voltage LOW

  delay(1000);                     // wait for a second
}
```

Step 4: Save the new sketch by appropriate name in a folder on your PC / Laptop

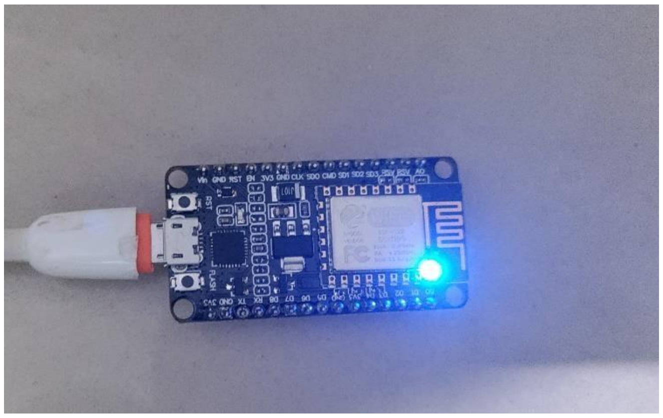
Step 5: Upload the sketch on Node MCU. Go to Sketch ---- > Upload

Step 6: Observe the output [Blinking LED]

Practice:

1. Change the on and off time of LED
2. Upload a sketch for dancing on board LEDs [one on and other off continuously]

Photo:



<u>Conclusion:</u>

The experiment taught us about the various Node MCUs and their configuration. We have successfully alternately lit the inbuilt LED of Node MCU with multiple shifting times & both illuminate at the same time using Pins D0 and D4.

Experiment No: 2

Interfacing External LED and Buzzer

Name of the Student: Bhavin Patil

Div.: D

Roll No.: 78

Aim: To interface external LED and Buzzer to Node MCU **Components Required:**

1) Node MCU – 1
2) Micro USB Cable – 1
3) PC/Laptop – 1
4) Connecting Wires
5) Bread Board – 1
6) LED – 4
7) Buzzer -1
8) Resistor 200 Ohm – 1

Software Required:

Arduino IDE

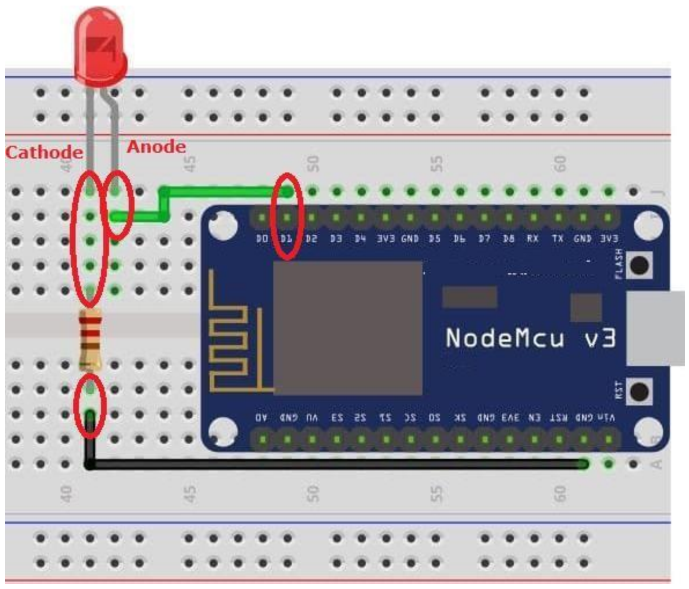
Theory:

We are familiar with blinking LED using Arduino boards as this is the fundamental step towards using a new development board. In this experiment, first we will see how to connect an external LED with **Node MCU** and how to **blink it using GPIO pins** of ESP8266 Node MCU. After that we will see how to connect buzzer with Node MCU.

Procedure:

1. Blinking External LED

Step 1: Make the circuit diagram on bread board according to connection diagram shown below. Anode of the LED [long leg of the LED] is connected to the D1 pin of the Node MCU, the cathode of the LED [short leg of the LED] is connected with the one terminal of the resistor [200 Ohm] and another terminal of the resistor is connected to the ground pin.



Step 2: Connect Node MCU to PC / Laptop with the help of micro USB cable

Step 3: Open new Sketch, Go to file ----> New

Step 4: Write following code in new sketch

```
#define ledPin D1 void setup() {

pinMode(ledPin, OUTPUT);

pinMode(LED_BUILTIN, OUTPUT);


}


void loop() {
  digitalWrite(ledPin,                         LOW);
digitalWrite(LED_BUILTIN, HIGH);
  delay(1000);
  digitalWrite(ledPin,                         HIGH);
digitalWrite(LED_BUILTIN, LOW);
  delay(2000);
}
```

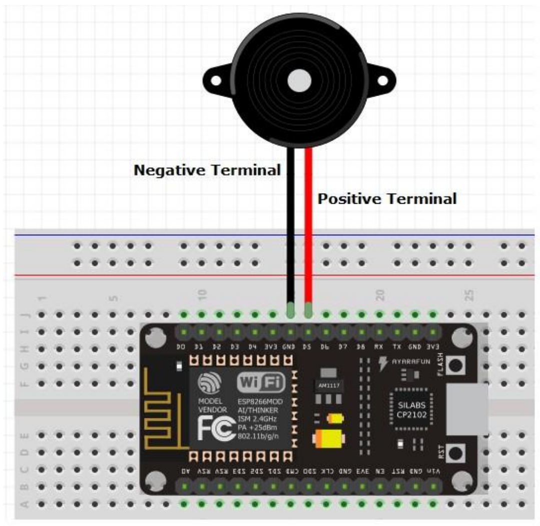
Step 5: Save the new sketch by appropriate name in a folder on your PC / Laptop

Step 6: Upload the sketch on Node MCU. Go to Sketch ----> Upload

Step 7: Observe the output [Blinking External LED]

2. Connecting Buzzer

Step 1: Make the circuit diagram on bread board according to connection diagram shown below. Positive terminal of buzzer [long leg of the buzzer] is connected to the D5 pin of the Node MCU, negative terminal of buzzer [short leg of the buzzer] is connected to the ground pin.



Step 2: Connect Node MCU to PC / Laptop with the help of micro USB cable

Step 3: Open new Sketch, Go to file ----> New

Step 4: Write following code in new sketch void

```
setup()

{  } void
loop()
{     tone(14, 494,
500);
delay(1000);
}
```

The three numbers inside the tone() function represent: the pin we send the sound (D5 or 14 in our case), the frequency of the sound wave we send and the duration of the tone.
You can change the last two parameters and play with the speed of the beeps and the sound of them.

Step 5: Save the new sketch by appropriate name in a folder on your PC / Laptop

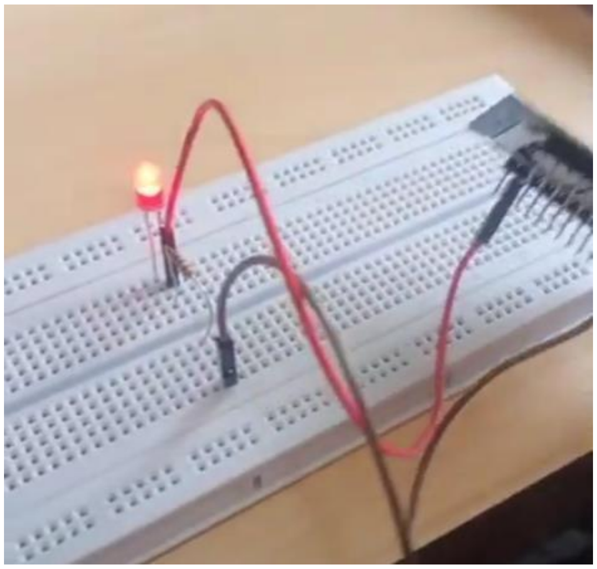
Step 6: Upload the sketch on Node MCU. Go to Sketch ----> Upload

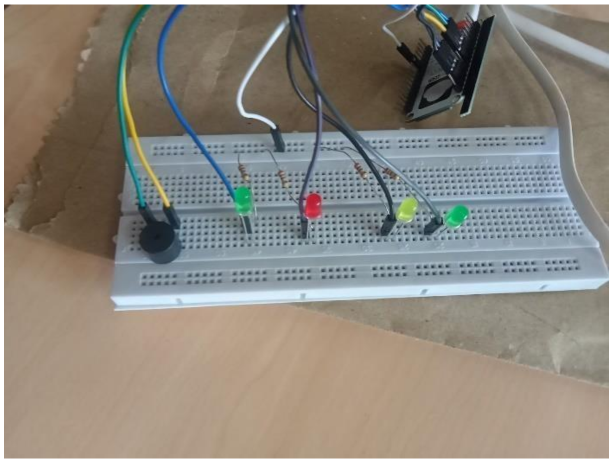
Step 7: Observe the output

Practice:

1. Connect 2/3/4 External LEDs and try different patterns of dancing LEDs
2. In tone function, use different music notes with different time period and run the melody.

Photo:





Conclusion:

Using the Node MCU and Arduino IDE we have successfully implemented the code for connecting the external LED light with buzzer on Bread Board and using the tune and delay function played some tune on buzzer.

<u>Experiment No: 3</u>

<u>Interfacing Temperature Sensor LM35</u>

Name of the Student: Bhavin Patil

Div.: D

Roll No.: 78

Aim: To interface temperature sensor LM35 with Node MCU **Components Required:**

1) Node MCU – 1
2) Micro USB Cable – 1
3) PC/Laptop – 1
4) Connecting Wires
5) Bread Board – 1
6) LM35 – 1

Software Required:

Arduino IDE

Theory:

In this experiment, we will see how to connect temperature sensor LM35 with **Node MCU** and how to measure temperature using analog pin of ESP8266 Node MCU. In general, an LM35 is a temperature sensor which is designed specifically to measure the hotness or coldness of an object.

LM35 is a precision IC temperature sensor with its output proportional to the temperature (in °C).

With LM35, the temperature can be measured more accurately than compared to the thermistor.

In this experiment, LM35 is used to measure the Room Temperature.

LM35 temperature sensor

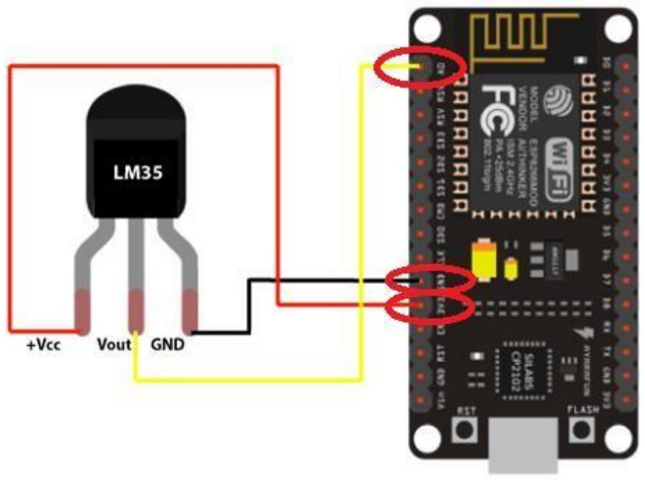
LM35 is a temperature sensor that can measure temperature in the range of -55°C to 150°C.

It is a 3-terminal device that provides an analog voltage proportional to the temperature. The higher the temperature, the higher is the output voltage.

The output analog voltage can be converted to digital form using ADC so that a microcontroller can process it.

Procedure:

Step 1: Make the circuit diagram on bread board according to connection diagram shown below. **Pin 1** of the LM35 goes into **+3.3v[3V3 Pin]** of the Node MCU. **Pin 2** of the LM35 goes into Analog Pin [**A0]** of the Node MCU. **Pin 3** of the LM35 goes into Ground Pin (**GND**) of the Node MCU.



Before getting the Celsius reading of the temperature The analog output voltage from LM35 must first be read from the Vout pin of LM35.This will be the raw value divided by 1024 times 3300. It is divided by 1024 because a span of 1024 occupies 3.3v. Here we get the ratio of the raw value to the full span of 1024 and then multiply it by 3300 to get the millivolt value. Since the output pin can give out a maximum of 3.3 volts (1024), 1024 represents the possible range it can give out.

Step 2: Connect Node MCU to PC / Laptop with the help of micro USB cable

Step 3: Open new Sketch, Go to file ----> New

Step 4: Write following code in new sketch

```
//initializes/defines the output pin of the LM35 temperature sensor int
outputpin = A0;
//this sets the ground pin to LOW and the input voltage pin to high void
setup ()
{
Serial.begin(9600);
//Setting the baud rate for Serial Monitor Communication
}

void loop() //main loop
{ int analogValue = analogRead(outputpin); float millivolts = (analogValue/1024.0) * 3300;
//3300 is the voltage provided by NodeMCU float celsius = millivolts/10; Serial.print("in
DegreeC=   ");
Serial.println(celsius);

//---------- Here is the calculation for Fahrenheit ----------//
float fahrenheit = ((celsius * 9)/5 + 32);
Serial.print(" in Farenheit=   ");
Serial.println(fahrenheit); delay(1000);
}
```

Step 5: Save the new sketch by appropriate name in a folder on your PC / Laptop

Step 6: Upload the sketch on Node MCU. Go to Sketch ----> Upload

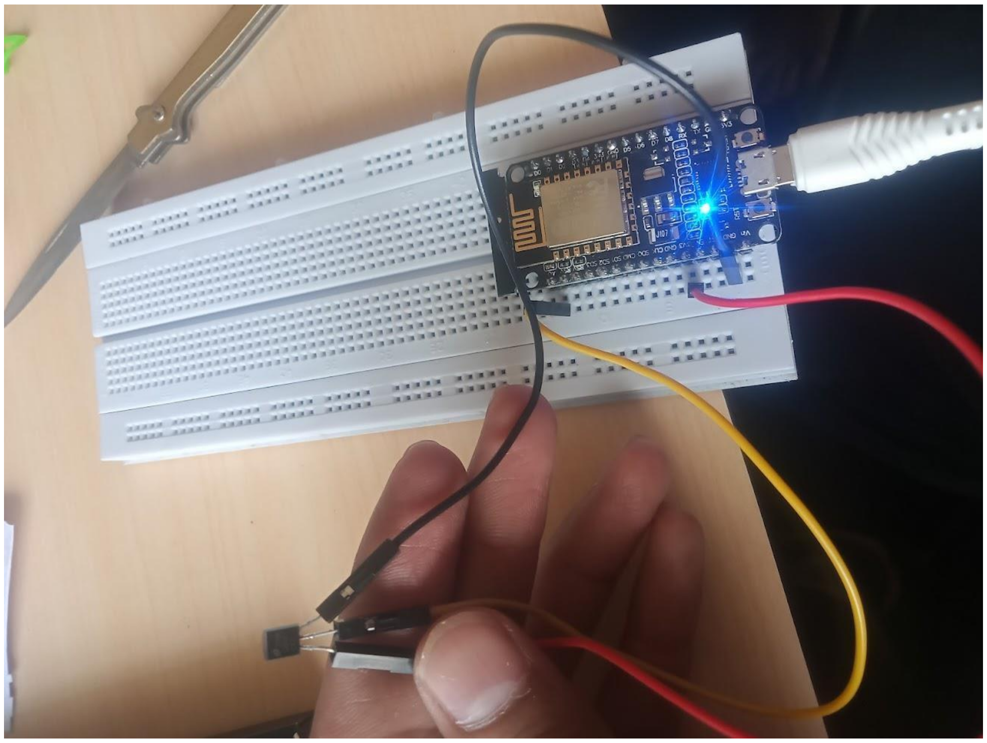
Step 7: Observe the output by clicking on Serial Monitor icon on the upper right corner of Arduino IDE. [Room Temperature]

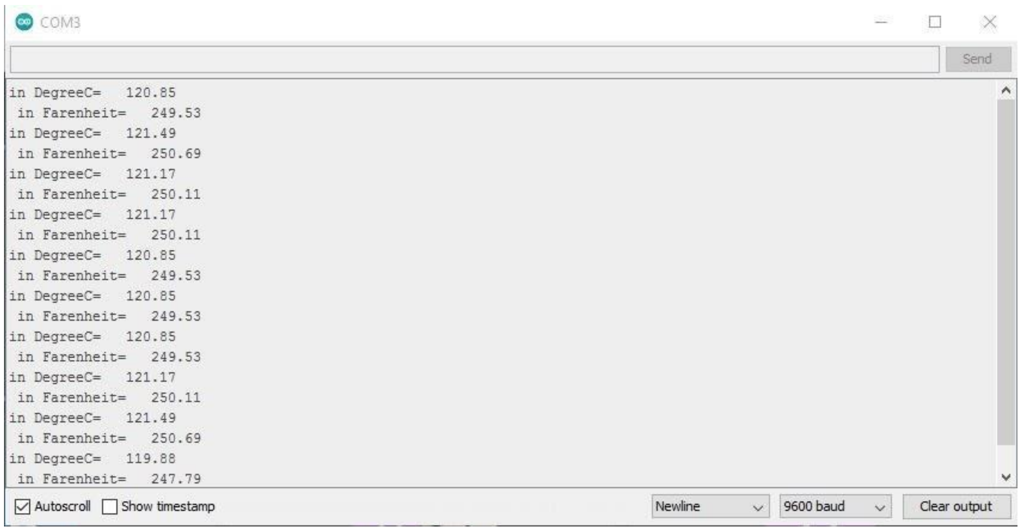
Step 8: Hold LM35 tightly in your hand and observe the change in the temperature reading sensed by it.

Practice:

1. Interface LM35 and Buzzer both, to Node MCU. Modify the sketch such that buzzer should beep after temperature increases or decreases beyond certain value

Photo:





```
in DegreeC=   120.85
 in Farenheit=   249.53
in DegreeC=   121.49
 in Farenheit=   250.69
in DegreeC=   121.17
 in Farenheit=   250.11
in DegreeC=   121.17
 in Farenheit=   250.11
in DegreeC=   120.85
 in Farenheit=   249.53
in DegreeC=   120.85
 in Farenheit=   249.53
in DegreeC=   120.85
 in Farenheit=   249.53
in DegreeC=   121.17
 in Farenheit=   250.11
in DegreeC=   121.49
 in Farenheit=   250.69
in DegreeC=   119.88
 in Farenheit=   247.79
```

☑ Autoscroll ☐ Show timestamp Newline 9600 baud Clear output

Conclusion:

In this experiment we have connected the temperature sensor LM35 with Node MCU and measured the room temperature using analog pin and printed the temperature on serial monitor. Due to faulty LM35 sensor the room temperature showing is wrong as the sensor itself got heated and printed that temperature.

Experiment No: 4

Node MCU /Arduino / Raspberry Pi wireless communication Raspberry Pi as a web server for Traffic Signal Control.

Name of the Student: Bhavin Patil

Div.:- D

Roll No.:- 78

Aim: Node MCU /Arduino / Raspberry Pi wireless communication Raspberry Pi as a web server for Traffic Signal Control.

Components Required:

1) Node MCU – 1
2) Micro USB Cable – 1
3) PC/Laptop – 1
4) Connecting Wires
5) Bread Board – 1
6) Red Yellow Green LED – (each 1)
7) Resistor 200 Ohm – 3

Software Required:

Arduino IDE **Procedure:**

Step 1: Connect Node MCU to PC / Laptop with the help of micro USB cable.

HOW TO CONNECT LED :

Make the circuit diagram on the bread board according to the connection diagram shown below. Positive terminal of LED (RED) long leg of the LED is connected to the one point of the Resistor 200 Ohm and another point is connected to the D5 pin of the Node MCU, negative terminal of LED (RED) [short leg of the LED] is connected to the ground pin.

Positive terminal of LED (YELLOW) long leg of the LED is connected to the one point of the Resistor 200 Ohm and another point is connected to the D6 pin of the Node MCU, negative terminal of LED (YELLOW) [short leg of the LED] is connected to the ground pin.

Positive terminal of LED (GREEN) long leg of the LED is connected to the one point of the Resistor 200 Ohm and another point is connected to the D7 pin of the Node MCU, negative terminal of LED (GREEN) [short leg of the LED]is connected to the ground pin.

Step 2: Open new Sketch, Go to file ----> New

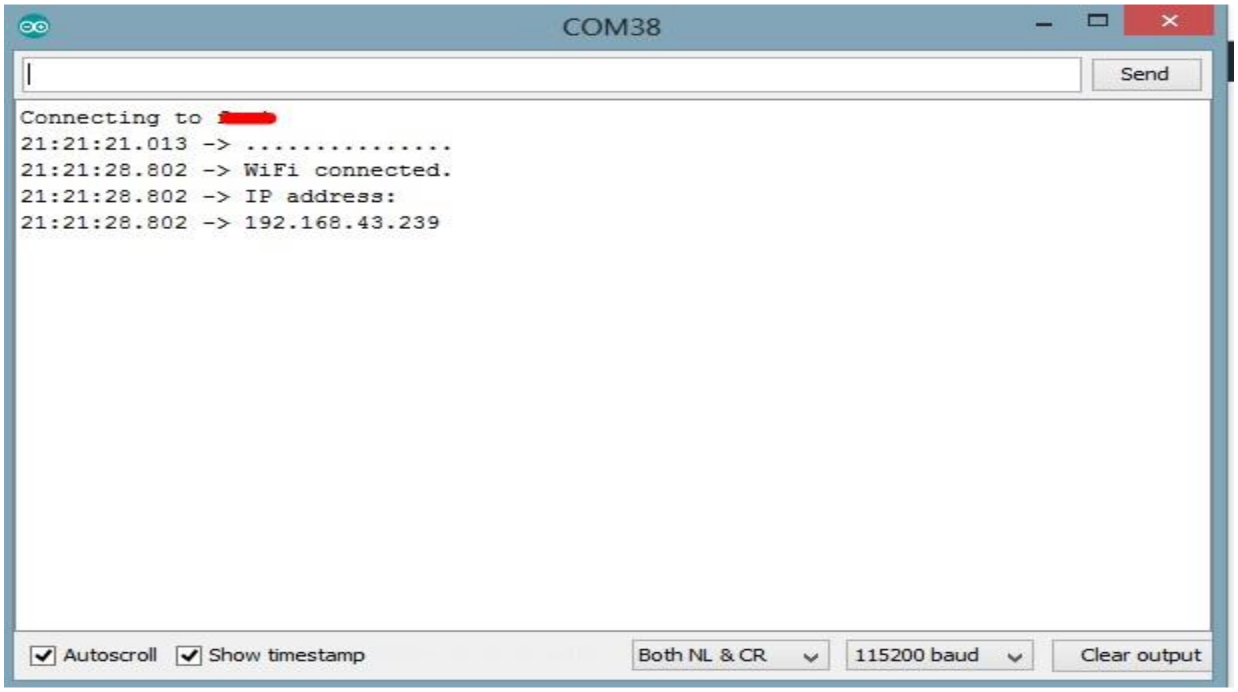
Step 3: Write following code in new sketch

```
#include <ESP8266WiFi.h>

WiFiClient client;
WiFiServer server(80); #
define LED_Red D5
# define LED_Yellow D6
# define LED_Green D7 void
setup() {
  // put your setup code here, to run once:
Serial.begin(9600);
WiFi.begin("Galaxy M317509","tzoh1462");
while(WiFi.status() !=WL_CONNECTED)
{delay(500);
Serial.print(".");
}
Serial.println();
Serial.println("NodeMCU is Connected");
Serial.println(WiFi.localIP());
server.begin();
pinMode(LED_Red, OUTPUT);
pinMode(LED_Yellow, OUTPUT);
pinMode(LED_Green, OUTPUT);
}  void
loop() {
  // put your main code here, to run repeatedly:
client=server.available(); if(client==1)
{
  String request =client.readStringUntil('\n');
  Serial.println(request);
request.trim();
  if(request =="GET /LED_Red HTTP/1.1")
  {
    digitalWrite(LED_Red, HIGH);
digitalWrite(LED_Yellow, LOW);
digitalWrite(LED_Green, LOW);
  }
  if(request =="GET /LED_Yellow HTTP/1.1")
  {
    digitalWrite(LED_Red, LOW);
digitalWrite(LED_Yellow, HIGH);
digitalWrite(LED_Green, LOW);
  }
  if(request =="GET /LED_Green HTTP/1.1")
  {
    digitalWrite(LED_Red, LOW);
digitalWrite(LED_Yellow, LOW);
digitalWrite(LED_Green, HIGH);
  }
  } }
```

Step 5: Save the new sketch by appropriate name in a folder on your PC / Laptop

Step 6: Upload the sketch on Node MCU. Go to Sketch ----> Upload

Ensure everything is connected as described under the schematics section. After uploading the code, you should see the IP address of your web server displayed in the serial monitor as shown below.

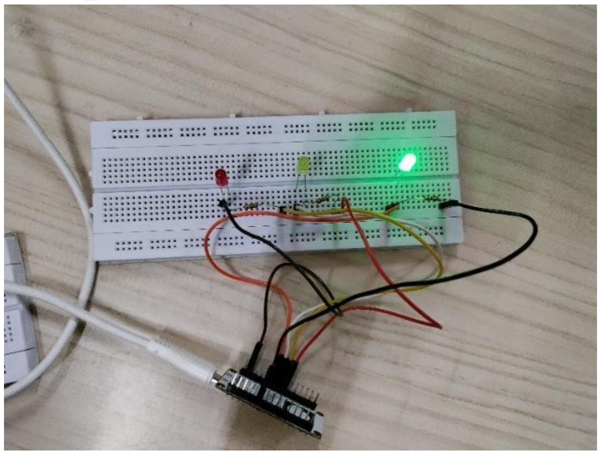


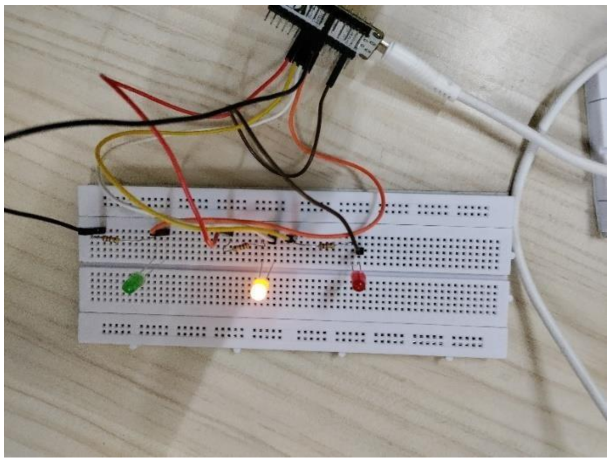
```
Connecting to [redacted]
21:21:21.013 -> ...............
21:21:28.802 -> WiFi connected.
21:21:28.802 -> IP address:
21:21:28.802 -> 192.168.43.239
```

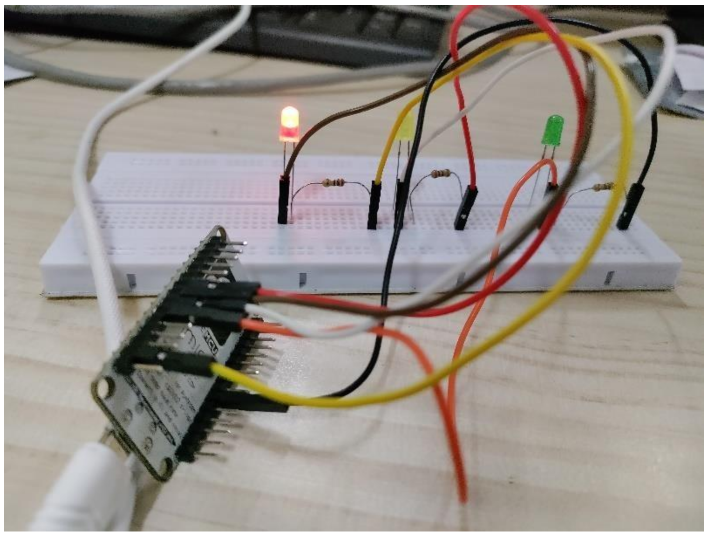
Copy the IP address and paste it in a web browser on any device (Mobile or PC) connected to the same network as the Node MCU. You should see the web page and be able to toggle the connected appliances by clicking the buttons. Copy the IP which occur on serial monitor copy that and next to that write down _ Red then RED led becomes glow. If you are writing Yellow than YELLOW LED becomes glow similarly for Green.

Step 7: Observe the outputs.

Output:







Conclusion:

With the help of the Node MCU, we used ESP8266 and used it as a web server for a traffic light by writing the proper code and putting the circuit together. Using the wifi, we have connected them on same network and writing the LED color after IP address in web browser we have changed the blinking LEDs.

Experiment No: 5

Node MCU Cloud interfacing and programming using Thingspeak.

Name of the Student: Bhavin Patil

Div.: D

Roll No.: 78

Aim: Uploading the data on cloud using Thingspeak.

Components Required:

1) Node MCU – 1
2) Micro USB Cable – 1
3) PC/Laptop – 1
4) Connecting Wires
5) Bread Board – 1
6) Temperature Sensor LM 35

Software Required:

Arduino IDE

Theory: We all are observed LM 35 temperature sensor during practical no. 3.

Procedure:

Step 1: Include Wi-Fi and ThingSpeak directories.

Step 2: Note that we already have wi-fi directory installed. If not, install it.

Step 3: Installing Thingspeak. Goto Sketch---Include Library---Manage Libraries---Write in Library Manage 'Thingspeak'---Install latest version

Step 4: Goto www.google.com. Googlr -thingspeak login---Sign-in

Step 5: Create account if you don't have one. Use vit.edu mailID. Location India. Your name etc--- Continue

Step 6: Goto Channels --- My Channels--- New Channel

Step 7: Write Channel name, description (not mandatory)---Create two fields. Field 1- temp in degrees Celsius. Field 2- temp in Fahrenheit. Save.

Step 8: Copy Channel ID and paste it in the code. long myChannelNumber = 1587542;

Step 9: Goto API keys. Copy API key (Write API Key) and paste it in the code. const char myWriteAPIKey[] = "OMVXC2R3UOKGBNV1";

Step 10: Enter the wifi login and password in the code. (Same as in Expt 4).
WiFi.begin("Login","Password");

Step 11: Write code for reading data from LM35 temperature sensor (Same as in Expt 3).
ThingSpeak.begin(client); ------ Starts thingspeak ThingSpeak.writeField (myChannelNumber, 1, tempc, myWriteAPIKey); ----- Displays temp in the field in thingspeak.

Step 12: Make hardware connections using node MCU and LM35, to sense and measure temperature.

Step 13: Upload sketch. The data (temp) will be displayed in the serial monitor. Also it will be collected and uploaded on cloud and displayed in the two fields.

Step 13: Observe the outputs.

Code:

```
#include <ESP8266WiFi.h>

#include <ThingSpeak.h> WiFiClient client;w long

myChannelNumber = 1715622; const char

myWriteAPIKey[] = "QDULOD8O4YKIDYLF";

const int sensor=A0; // Assigning analog pin A0 to variable 'sensor'
```

```
float tempc;  //variable to store temperature in degree Celsius

float tempf;  //variable to store temperature in Fahreinheit  float

vout;  //temporary variable to hold sensor reading


void setup() {

  // put your setup code here, to run once:

Serial.begin(9600);

WiFi.begin("realme C25","QWERTYOP"); while(WiFi.status()

!=WL_CONNECTED)

{delay(100);

Serial.print(".");

}

Serial.println();

Serial.println("NodeMCU is Connected");

Serial.println(WiFi.localIP());

ThingSpeak.begin(client); pinMode(sensor,INPUT);

}


void loop() {

  // put your main code here, to run repeatedly:

vout=analogRead(sensor);
```

vout=(vout*500)/1024-18; tempc=vout; // Storing

value in Degree Celsius tempf=(vout*1.8)+32; //

Converting to Fahrenheit

  Serial.println("Temperature in C: " + (String) tempc);

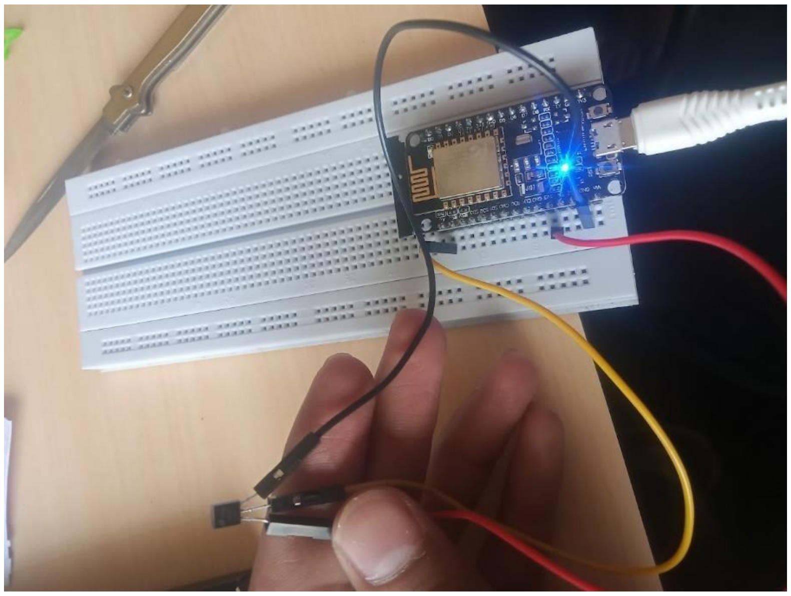
  Serial.println("Temperature in F: " + (String) tempf);

  ThingSpeak.writeField(myChannelNumber, 1, tempc, myWriteAPIKey);

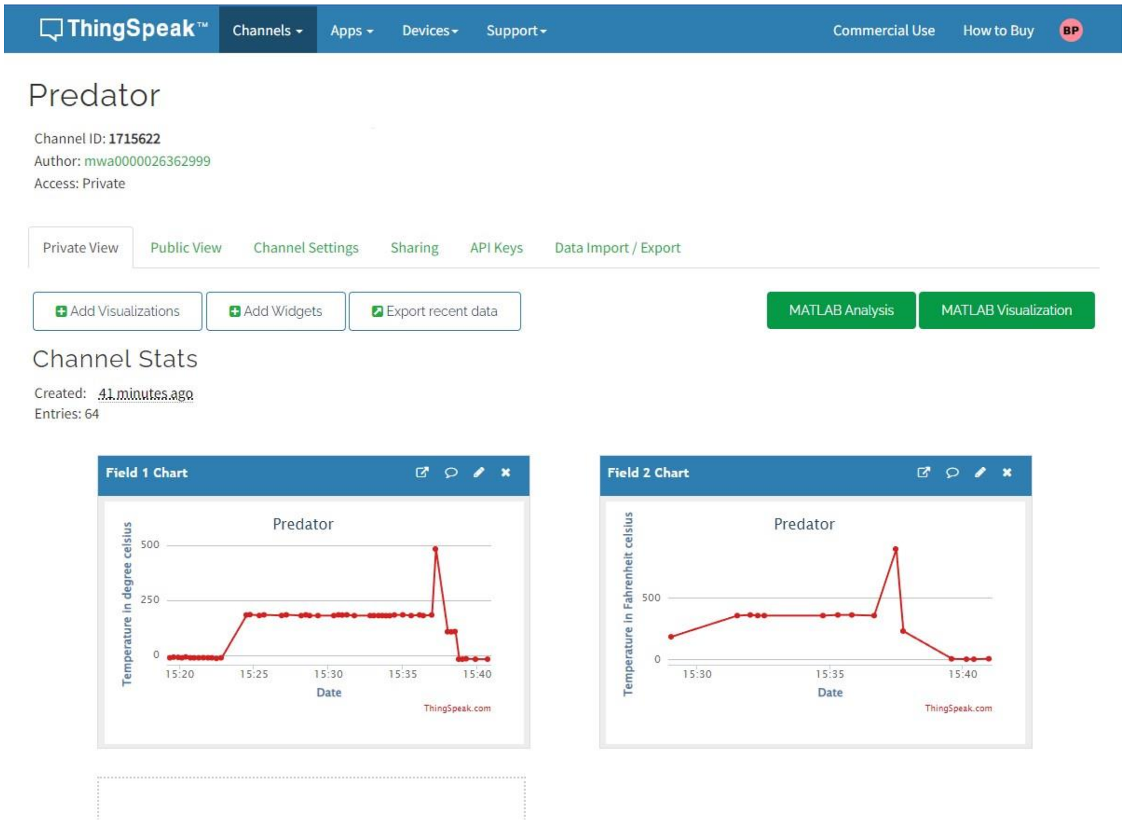
ThingSpeak.writeField(myChannelNumber, 2, tempf, myWriteAPIKey);   delay(2000);

}

Result:



Setup



Predator

Channel ID: 1715622
Author: mwa0000026362999
Access: Private

Channel Stats

Created: 41 minutes ago
Entries: 64

Thingspeak dashboard

Conclusion:

ThingSpeak is a platform that allows us to visualize data from sensors in a graphical format, making it easier to interpret data. This technology can also be utilized to keep an eye on our system from faraway.

Experiment No: 6

Home Automation using Cisco Packet Trace

Name of the Student: Bhavin Patil

Div. D

Roll No. 78

Objectives

Part 1: Explore the Existing Smart Home Network

Part 2: Add Wired IoT Devices to the Smart Home Network

Part 3: Add Wireless IoT Devices to the Smart Home Network

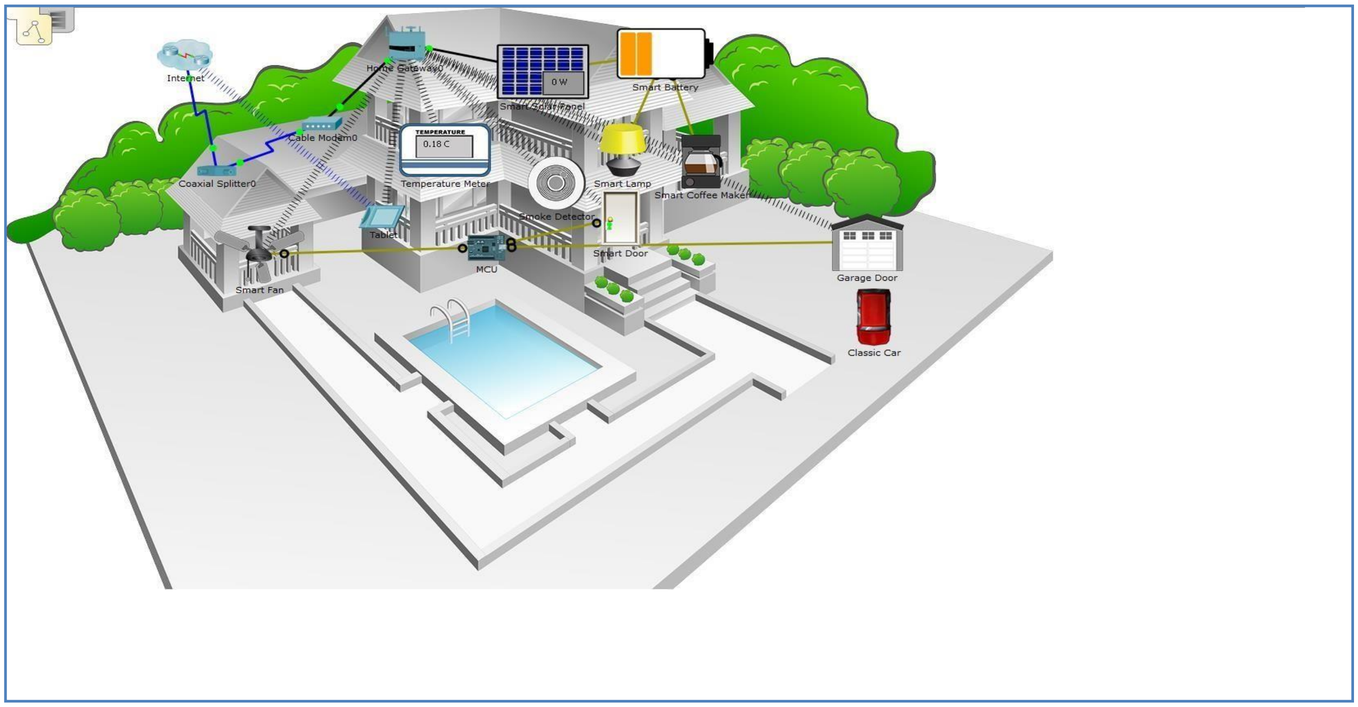
Theory-

We have studied a case study on Home Automation. Let us build a network for Home Automation using Packet Tracer.

In this Lab we will open a Packet Tracer file with an existing home network, explore the devices on the network, and then add additional wired and wireless IoT devices

Packet Tracer – Adding IoT Devices
The Smart Home Network

Part 1: Explore the Existing Smart Home Network



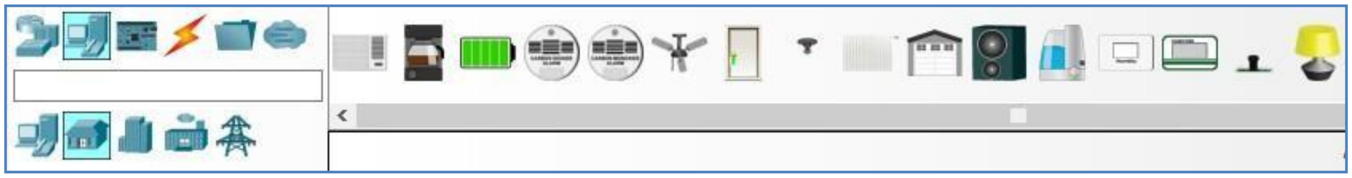
Step 1: Open the Smart_Home_Network.pkt file

- Open the **Smart_Home_Network.pkt** file.
- Save the file to your computer.

Step 2: Explore the Smart Home Network

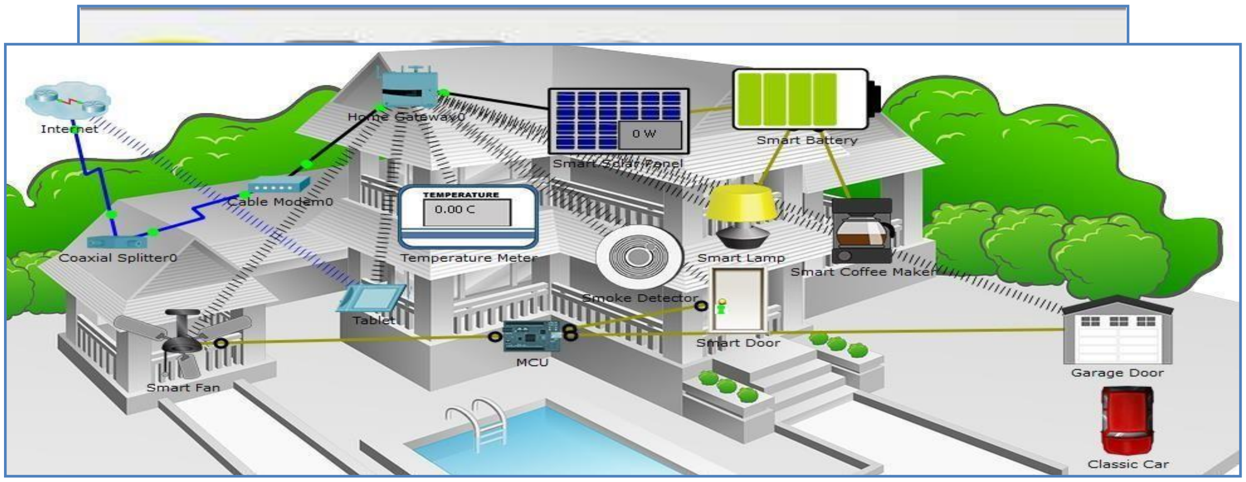
- Explore IoT end devices.

At the bottom left corner of the Packet Tracer window, locate and click the **End Devices** icon in the top row, and the **Home** icon in the bottom row of the **Device-Type Selection** box.



Across the bottom of the Packet Tracer window, the **Device-Specific Selection** box displays the many different Smart Home IoT devices available.

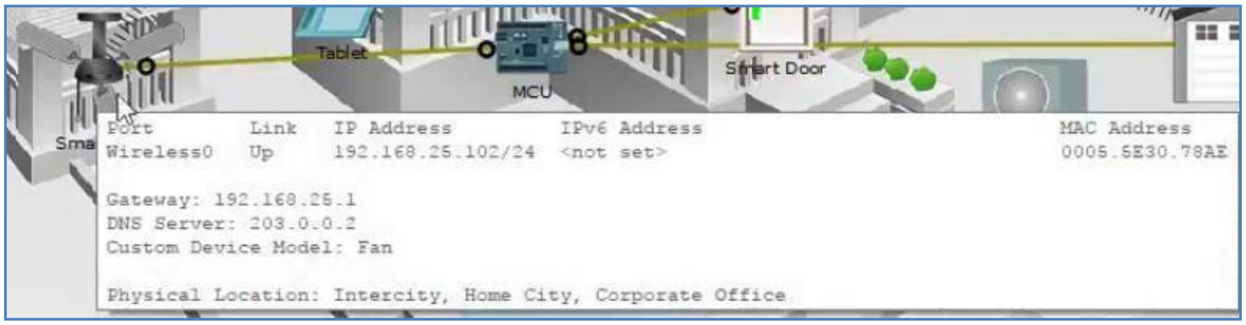
Move the mouse pointer over each device and notice that the descriptive name of the device is displayed at the bottom of the **Device-Specific Selection** box. Take a moment to look at each device type.



- Explore the Smart Home network.

In the **Logical** workspace is a prebuilt smart home network that consists of many wired and wireless IoT devices, and network infrastructure devices.

When we place your curser over a device, such as the Smart Fan, an informational window opens containing basic network information about that device.

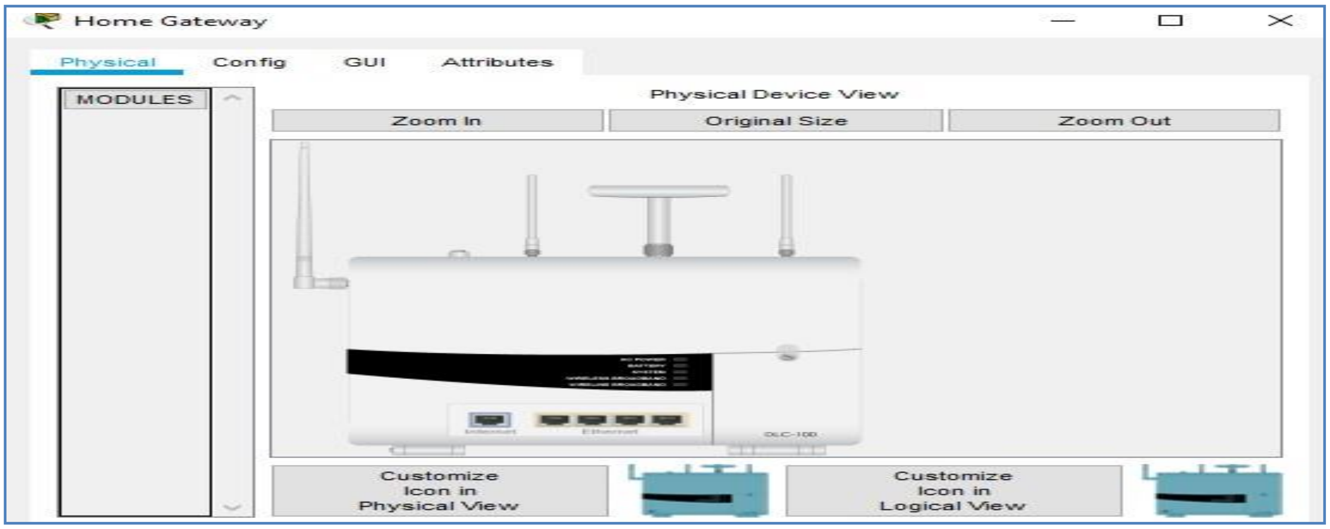


```
Port       Link   IP Address          IPv6 Address                         MAC Address
Wireless0  Up     192.168.25.102/24   <not set>                            0005.5E30.78AE

Gateway: 192.168.25.1
DNS Server: 203.0.0.2
Custom Device Model: Fan

Physical Location: Intercity, Home City, Corporate Office
```

To turn on or activate a device, simply hold down the **Alt** key on the keyboard and then move the cursor over the device. Try this on each of the smart devices to observe what they do.

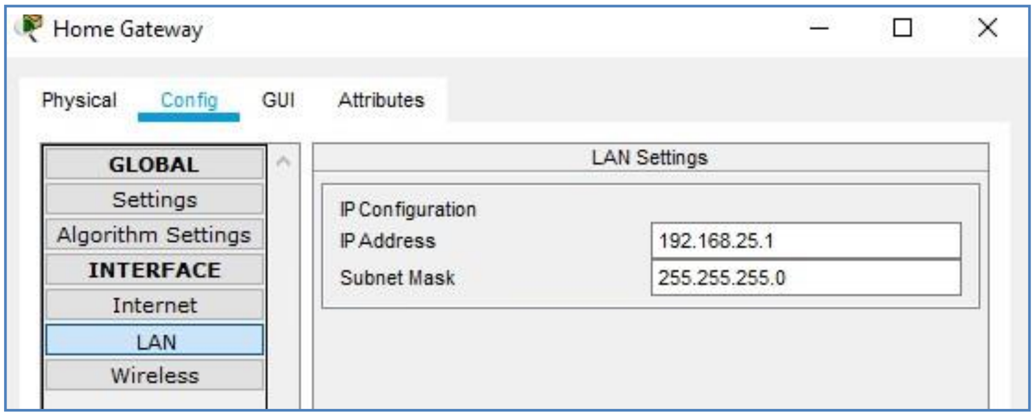
The smart home network also consists of infrastructure devices such as a home gateway.
Click the **Home Gateway** icon to open the **Home Gateway** window.

The **Physical** tab is selected by default and shows a picture of the Home Gateway.


Home Gateway
Physical
Config
GUI
Attributes
MODULES
Physical Device View
Zoom In
Original Size
Zoom Out
Customize Icon in Physical View
Customize Icon in Logical View

Next, click the **Config** tab and then in the left pane click **LAN** to view the LAN Settings of the Home Gateway.

Write down the IP Address of the home network for future reference. _____________

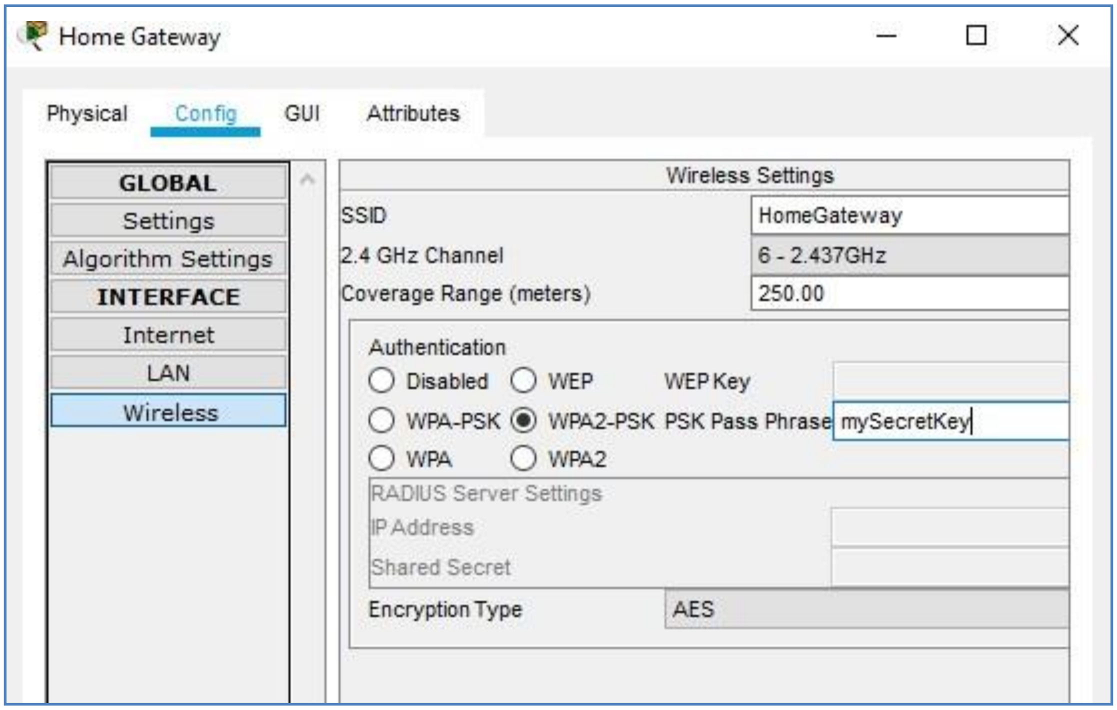


IP Configuration

IP Address	192.168.25.1
Subnet Mask	255.255.255.0

Click **Wireless** in the left pane to view the wireless settings of the Home Gateway.

Write down the SSID of the home network___________and the WPA2-PSK Pass Phrase for future reference.



SSID	HomeGateway
2.4 GHz Channel	6 - 2.437GHz
Coverage Range (meters)	250.00

Authentication

Disabled WEP WEP Key

WPA-PSK [✓ WPA2-PSK] PSK Pass Phrase mySecretKey

WPA WPA2

RADIUS Server Settings

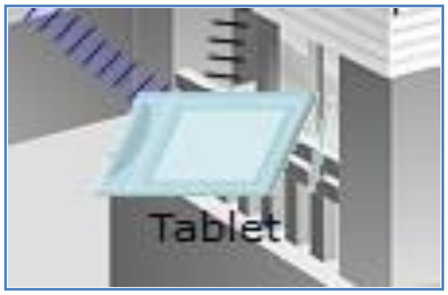
IP Address

Shared Secret

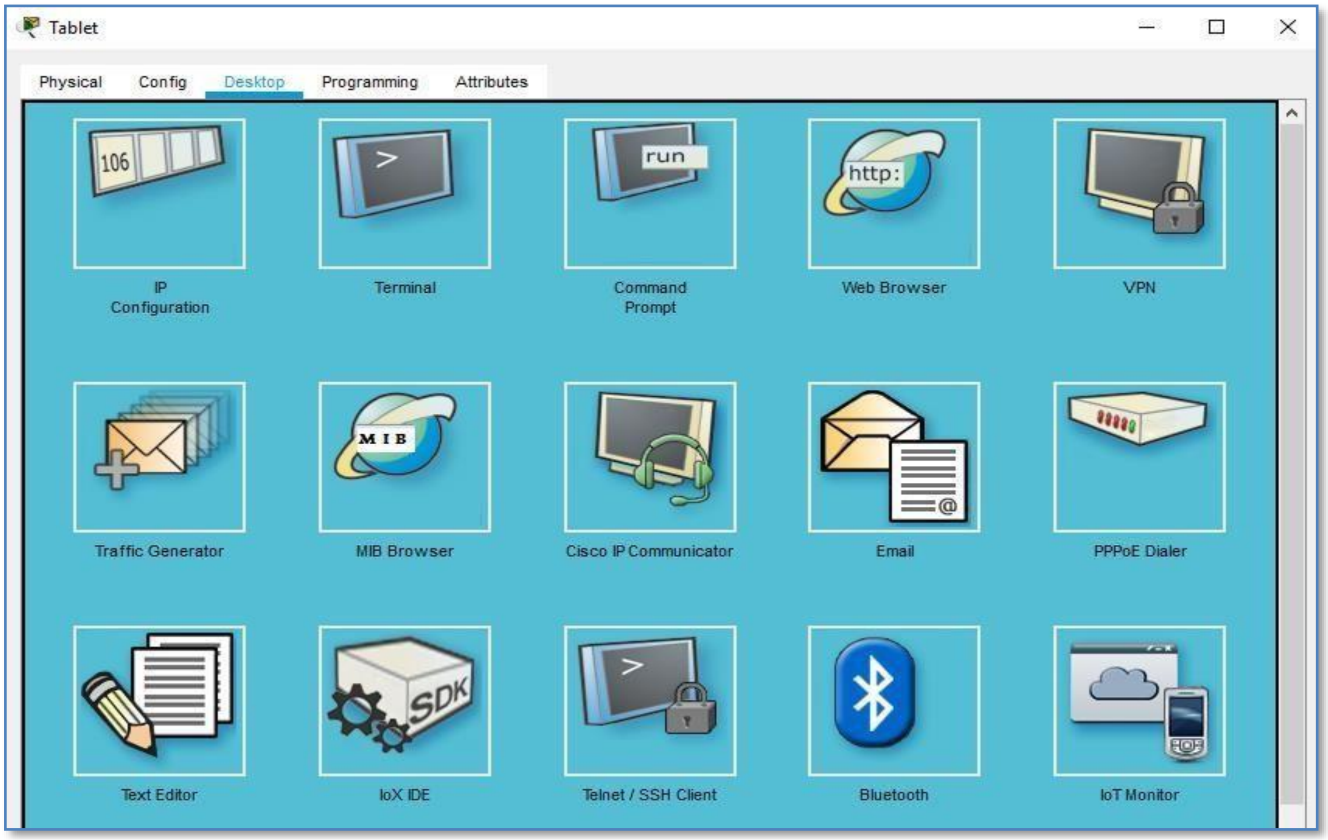
Encryption Type AES

Close the **Home Gateway** window.

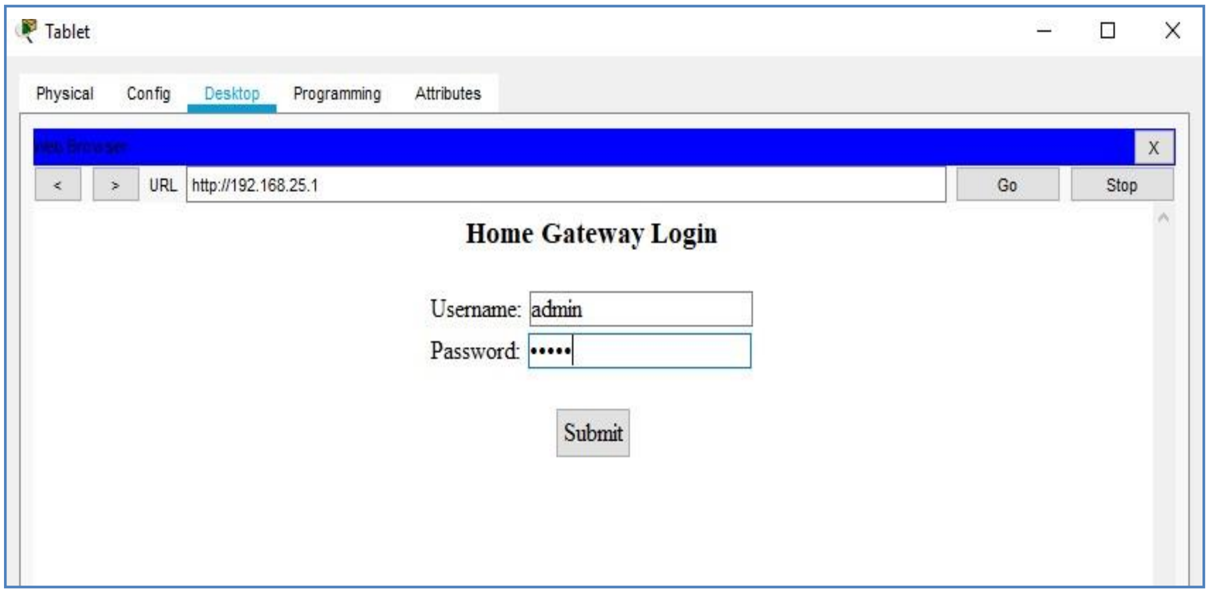
Next, click the **Tablet** device icon to open the **Tablet** window.



In the **Tablet** window, select the **Desktop** tab and then click the **Web Browser** icon.



In the **Web Browser** window, type the IP address of the Home Gateway 192.168.25.1 into the URL box and click **Go**. In the **Home Gateway Login** screen, type admin for both the username and the password and click **Submit**.



Tablet

Physical Config Desktop Programming Attributes

Web Browser

URL http://192.168.25.1

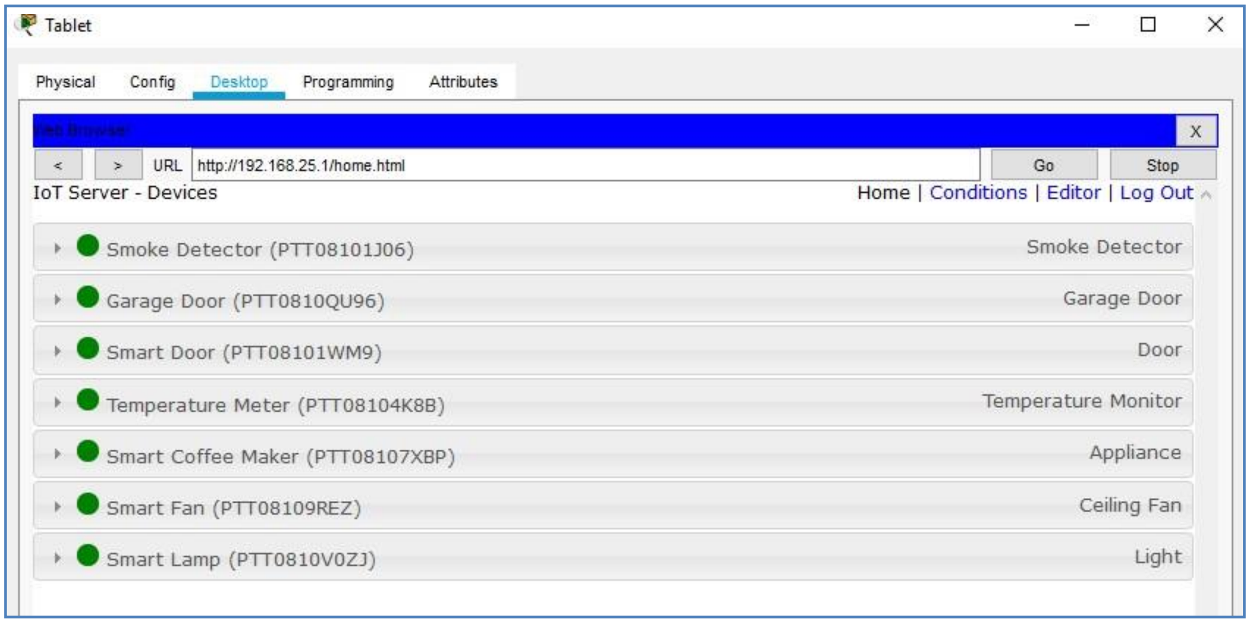
Home Gateway Login

Username: admin

Password: •••••

Submit

After you have connected to the Home Gateway web interface, a list of all the connected IoT devices appears.



Tablet

Physical Config Desktop Programming Attributes

Web Browser

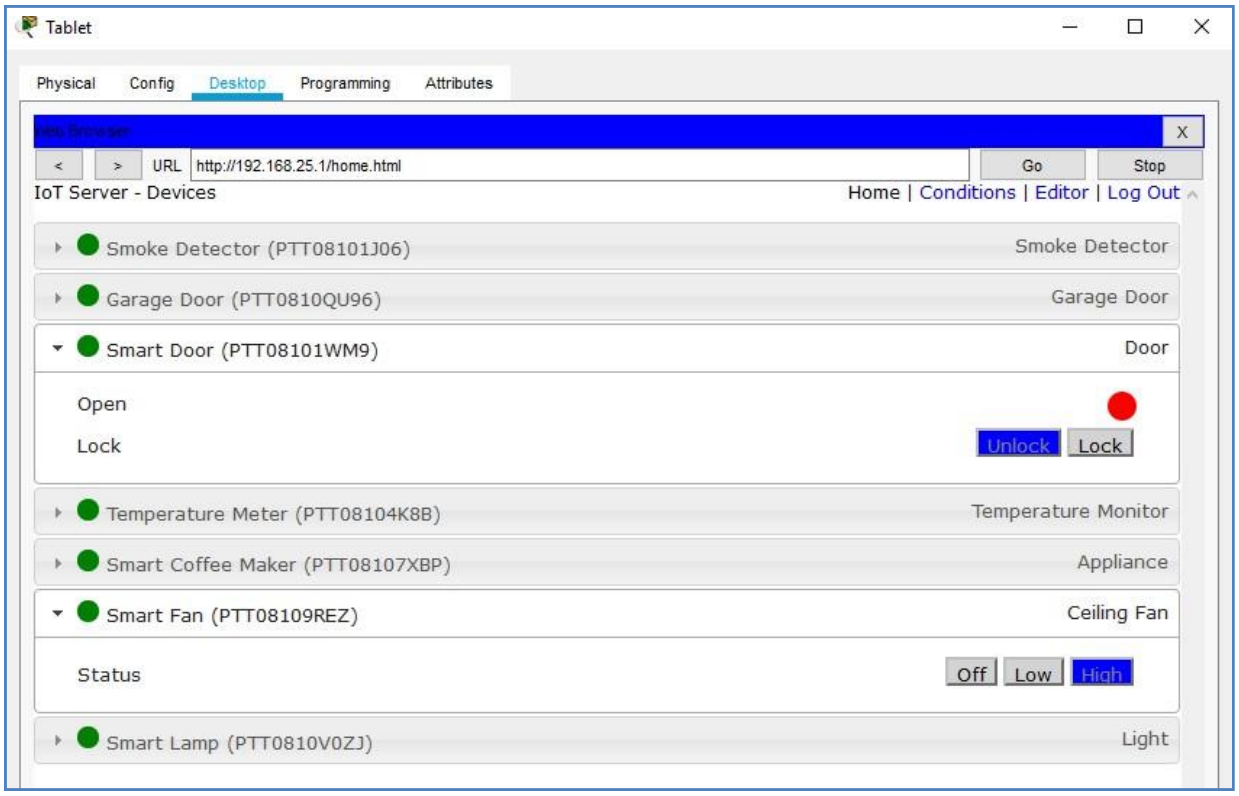
URL http://192.168.25.1/home.html

IoT Server - Devices

Home | Conditions | Editor | Log Out

● Smoke Detector (PTT08101J06)	Smoke Detector
● Garage Door (PTT0810QU96)	Garage Door
● Smart Door (PTT08101WM9)	Door
● Temperature Meter (PTT08104K8B)	Temperature Monitor
● Smart Coffee Maker (PTT08107XBP)	Appliance
● Smart Fan (PTT08109REZ)	Ceiling Fan
● Smart Lamp (PTT0810V0ZJ)	Light

When you click a device in the list, the status and settings of that device is displayed.



URL http://192.168.25.1/home.html

IoT Server - Devices — Home | Conditions | Editor | Log Out

- Smoke Detector (PTT08101J06) — Smoke Detector
- Garage Door (PTT0810QU96) — Garage Door
- Smart Door (PTT08101WM9) — Door
  - Open ●
  - Lock — Unlock | Lock
- Temperature Meter (PTT08104K8B) — Temperature Monitor
- Smart Coffee Maker (PTT08107XBP) — Appliance
- Smart Fan (PTT08109REZ) — Ceiling Fan
  - Status — Off | Low | High
- Smart Lamp (PTT0810V0ZJ) — Light

Close the **Tablet** window.

Part 2: Add Wired IoT Devices to the Smart Home Network

Step 1: Cable a device to the network

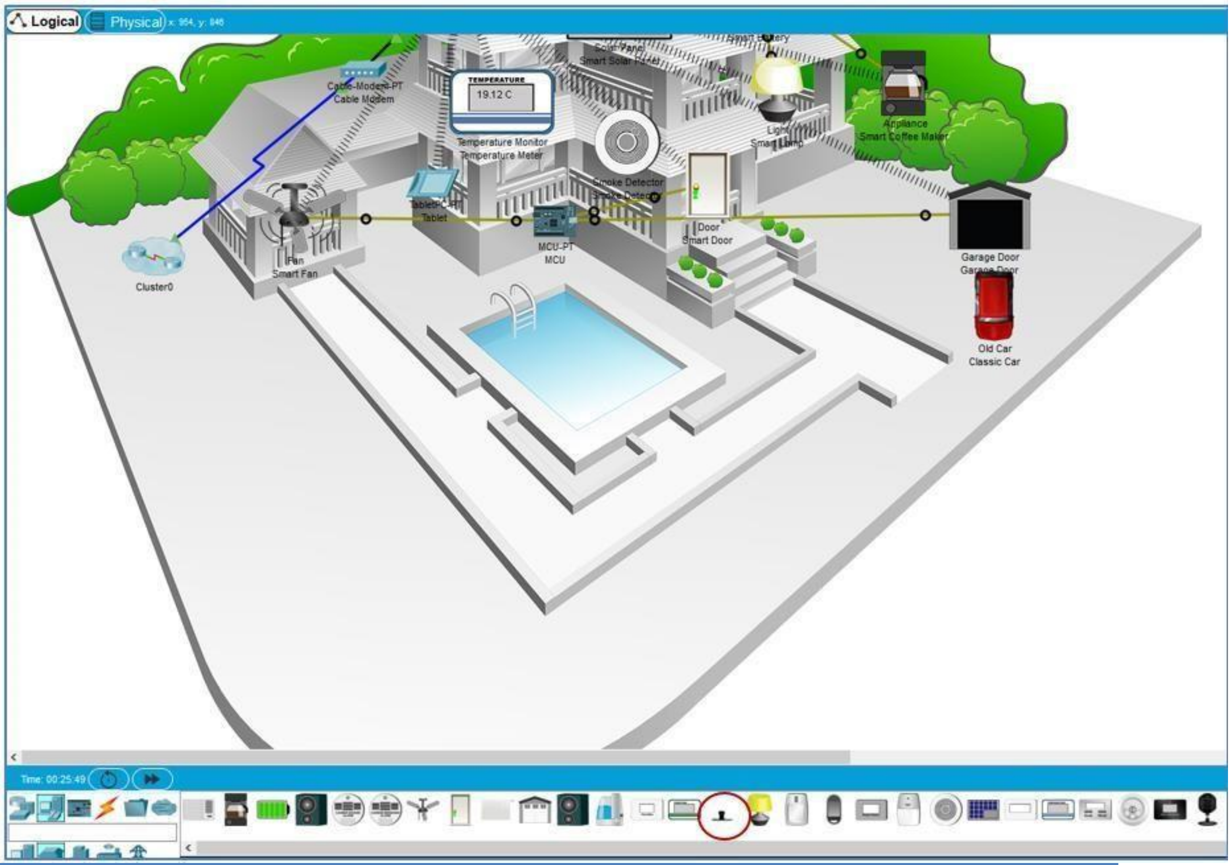
- In the **Device-Specific Selection** box, click the **Lawn Sprinkler** icon and then click in the workspace where you would like to locate the **Lawn Sprinkler**.
- Cable the Lawn Sprinkler to the Home Gateway.

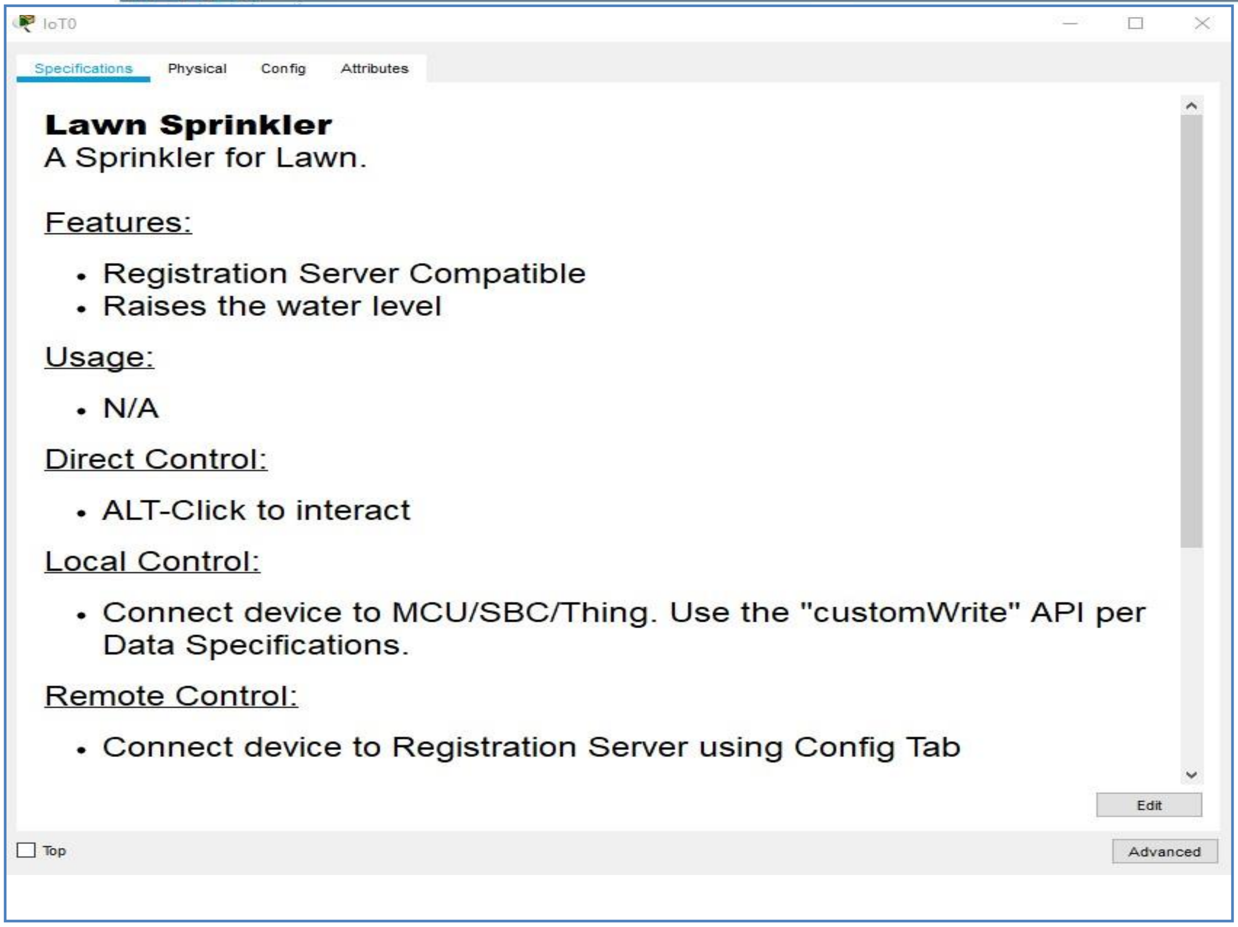
In the **Device-Type Selection** box, click the **Connections** icon (this looks like a lightning bolt). Click the **Copper Straight Through** connector type icon in the **Device-Specific Selection** box.

Then click the **Sprinkler** icon and connect one end of the cable to the Sprinkler's FastEthernet0 interface. Next, click the **Home Gateway** icon and connect the other end of the cable to an available Ethernet interface.

Step 2: Configure the sprinkler for network connectivity

- Click the **Lawn Sprinkler** device icon in the workspace to open the device window. Notice that right now the name of the Lawn Sprinkler is a generic IoT0.
- The device window will open to the **Specification** tab which gives information about the device which can be edited.





Lawn Sprinkler
A Sprinkler for Lawn.

Features:

- Registration Server Compatible
- Raises the water level

Usage:

- N/A

Direct Control:

- ALT-Click to interact

Local Control:

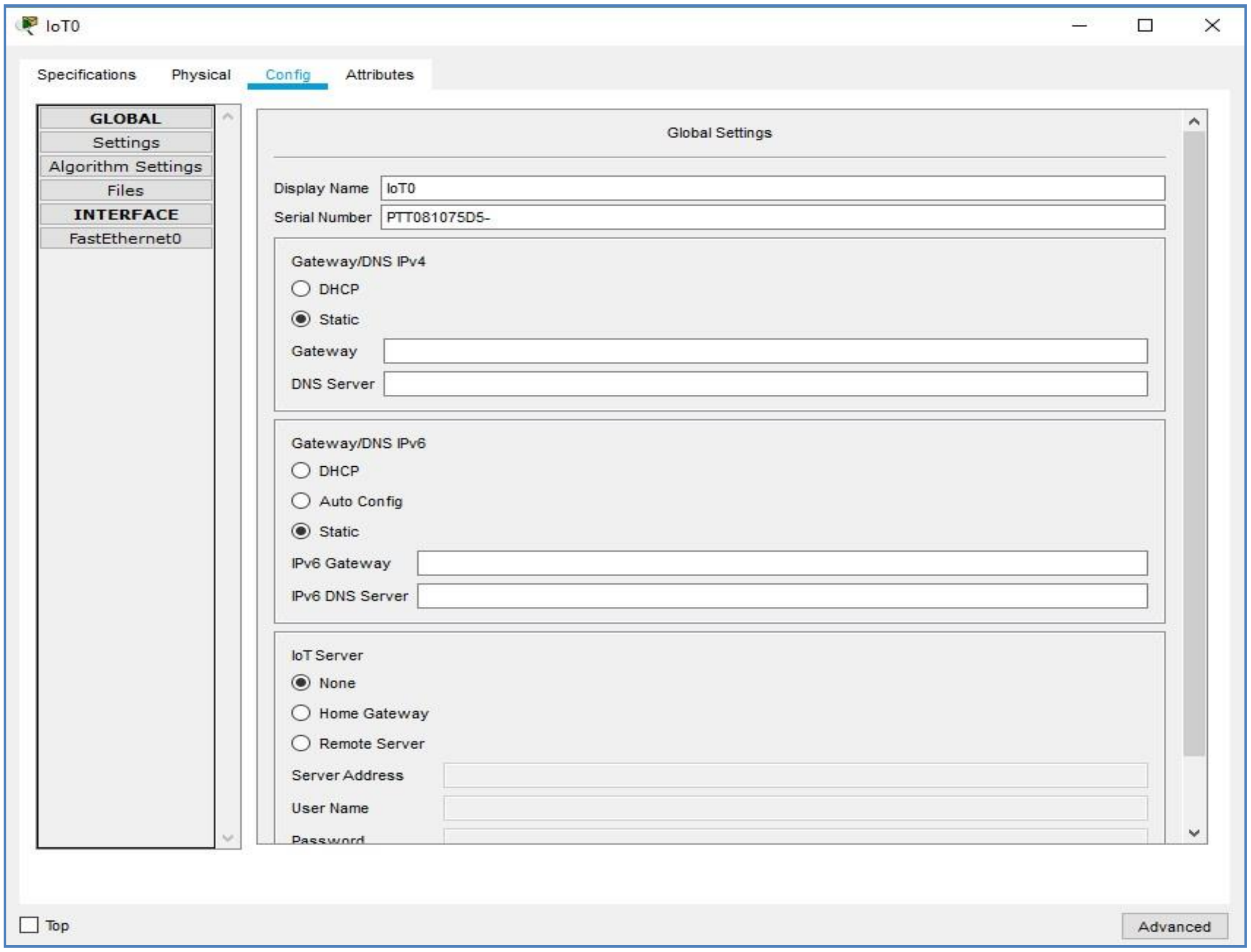
- Connect device to MCU/SBC/Thing. Use the "customWrite" API per Data Specifications.

Remote Control:

- Connect device to Registration Server using Config Tab

Click the Config tab to edit the device configuration settings. In the Config tab, make the following changes to Settings:

- Set the **Display Name** to Sprinkler1 (notice the window name changes to Sprinkler1) • Set the IoT Server to Home Gateway


IoT0

Global Settings

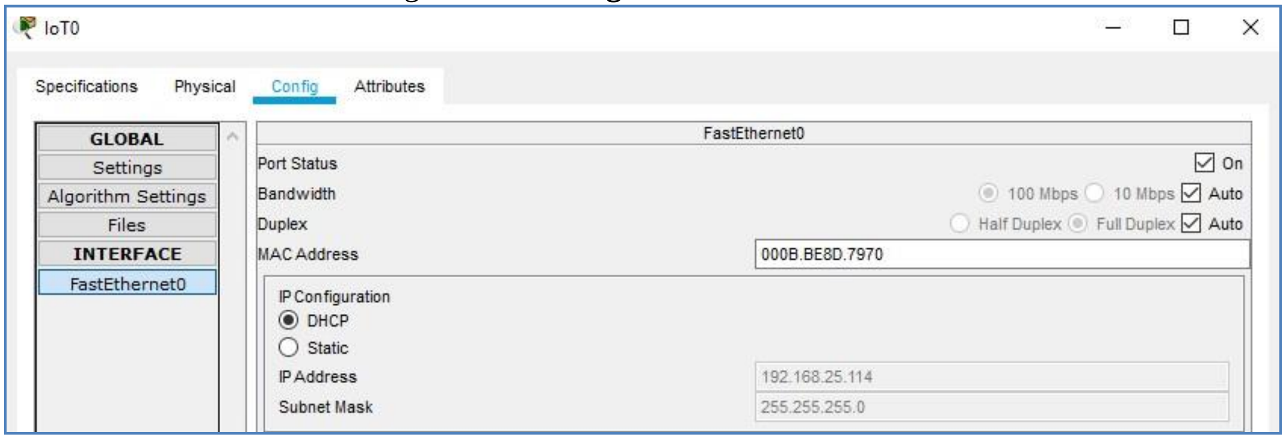
Display Name: IoT0
Serial Number: PTT081075D5-

Gateway/DNS IPv4: ○ DHCP [✓ Static]
Gateway:
DNS Server:

Gateway/DNS IPv6: ○ DHCP ○ Auto Config [✓ Static]
IPv6 Gateway:
IPv6 DNS Server:

IoT Server: [✓ None] ○ Home Gateway ○ Remote Server
Server Address:
User Name:
Password

Click **FastEthernet0** and change the **IP Configuration** to **DHCP**.


IoT0

FastEthernet0

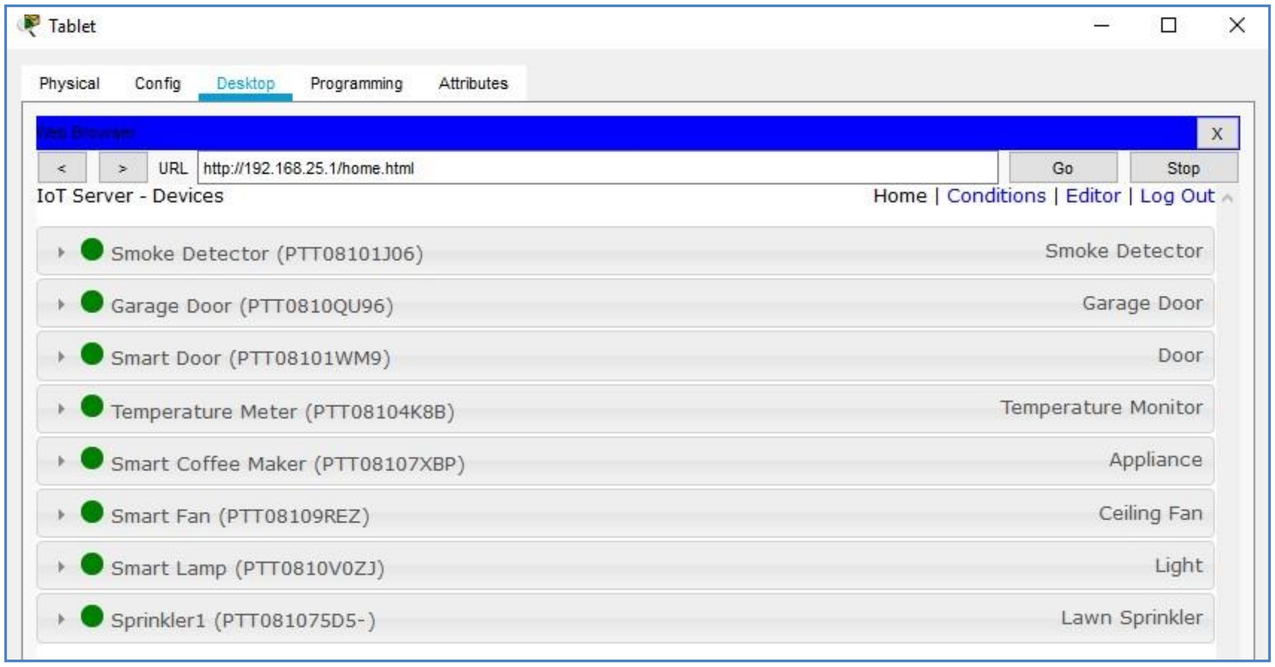
Port Status: ☑ On
Bandwidth: ○ 100 Mbps ○ 10 Mbps ☑ Auto
Duplex: ○ Half Duplex ○ Full Duplex ☑ Auto
MAC Address: 000B.BE8D.7970

IP Configuration: [✓ DHCP] ○ Static
IP Address: 192.168.25.114
Subnet Mask: 255.255.255.0

Close the **Sprinkler1** window.

- Verify that the sprinker is on the network.Log into the **Home Gateway** from the **Tablet.**

The device Sprinkler 1 should now appear in the IoT Server – Devices list.



http://192.168.25.1/home.html

IoT Server - Devices

Smoke Detector (PTT08101J06)	Smoke Detector
Garage Door (PTT0810QU96)	Garage Door
Smart Door (PTT08101WM9)	Door
Temperature Meter (PTT08104K8B)	Temperature Monitor
Smart Coffee Maker (PTT08107XBP)	Appliance
Smart Fan (PTT08109REZ)	Ceiling Fan
Smart Lamp (PTT0810V0ZJ)	Light
Sprinkler1 (PTT081075D5-)	Lawn Sprinkler

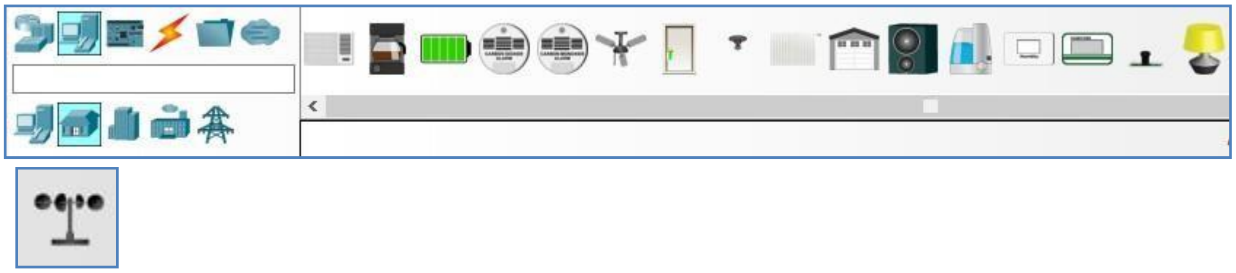
Close the Tablet window.

Step 3: Experiment by adding other types of IoT devices to the smart home network.

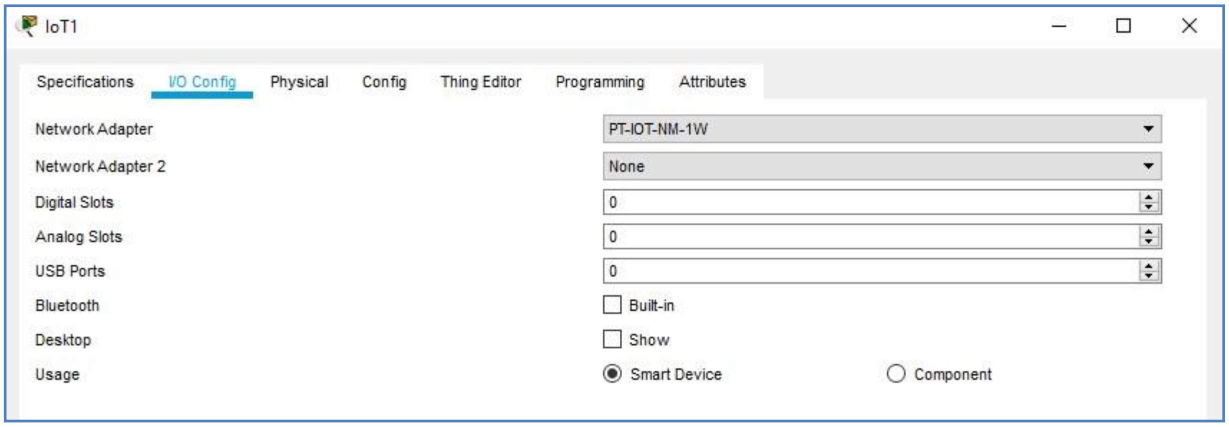
Part 3: Add Wireless IoT Devices to the Smart Home Network

Step 1: Add a wireless device to the network

- In the **Device-Specific Selection** box click the **Wind Detector** icon and then click in the workspace where you would like to locate the **Wind Detector**.



- Add wireless module to the Wind Detector.
  Change the **Network Adapter** drop down list to **PT-IOT-NM-1W**, which is a wireless adapter. Click the **Wind Detector** icon in the workspace to open the IoT device window. In the bottom right corner of the IoT device window, click the **Advanced** button. Notice more tabs become visible at the top of the window. Click the **I/O Config** tab.

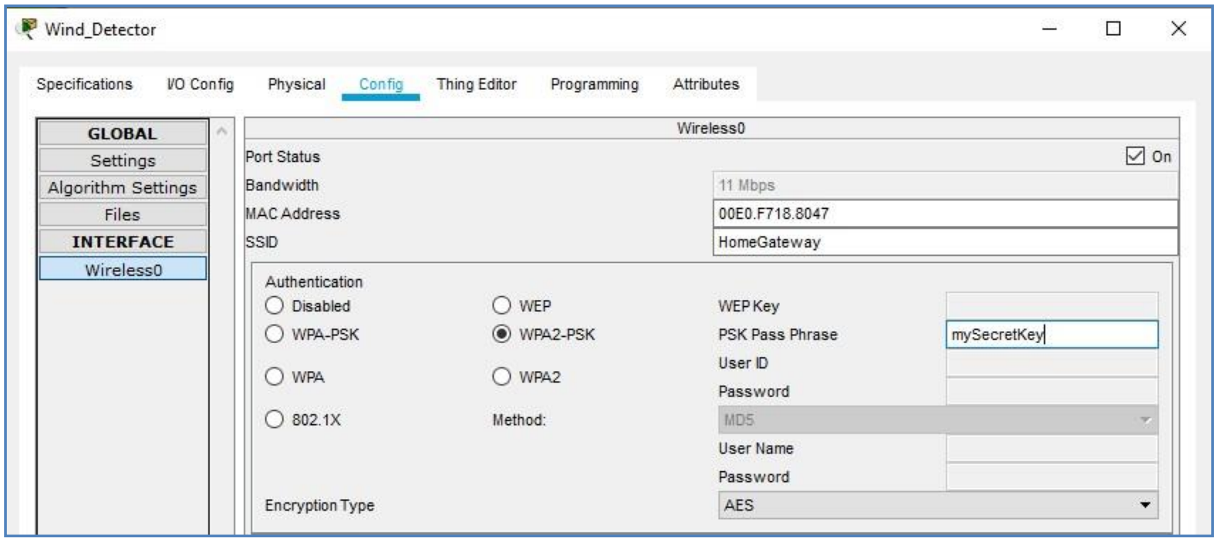

IoT1

Network Adapter	PT-IOT-NM-1W
Network Adapter 2	None
Digital Slots	0
Analog Slots	0
USB Ports	0
Bluetooth	☐ Built-in
Desktop	☐ Show
Usage	[✓ Smart Device] Component

Configure the Wind Detector for the wireless network. Click the **Config** tab.

Change the **Display Name** to **Wind_Detector** and change the **IoT Server** to **Home Gateway**.

Next click **Wireless0** in the left pane. Change the Authentication type to **WPA2-PSK** and in the **PSK Pass Phrase** box type **mySecretKey.** These are the wireless settings from the Home Gateway that you recorded in Part 1.


Wind_Detector

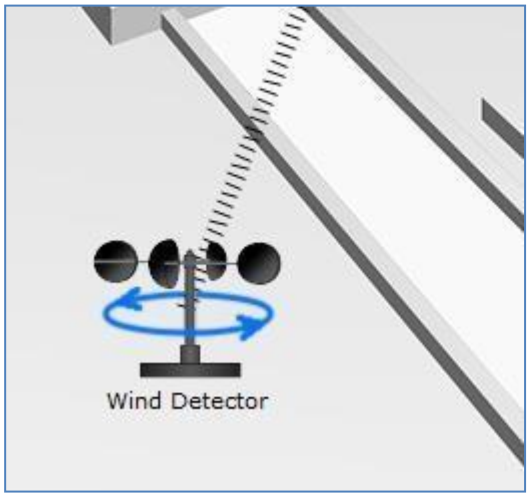
Wireless0

Port Status	✓ On
Bandwidth	11 Mbps
MAC Address	00E0.F718.8047
SSID	HomeGateway

Authentication: Disabled, WEP, WPA-PSK, [✓ WPA2-PSK], WPA, WPA2, 802.1X

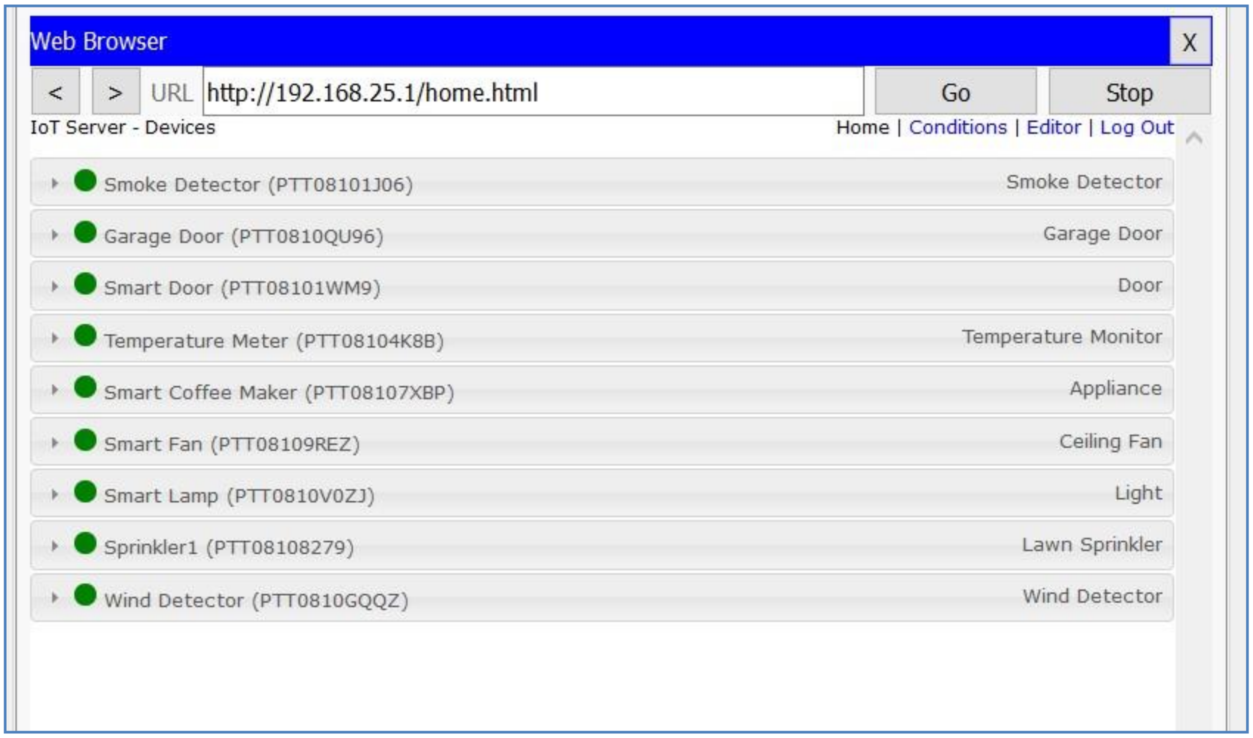
WEP Key	
PSK Pass Phrase	mySecretKey
User ID	
Password	
Method:	MD5
User Name	
Password	
Encryption Type	AES

A wireless connection should be formed between the Wind Detector and the Home Gateway.



Verify the Wind Detector is on the network. Log into the Home Gateway from the Tablet**.**

The device Wind Detector should now appear in the **IoT Server – Devices** list.



Web Browser

URL http://192.168.25.1/home.html

IoT Server - Devices

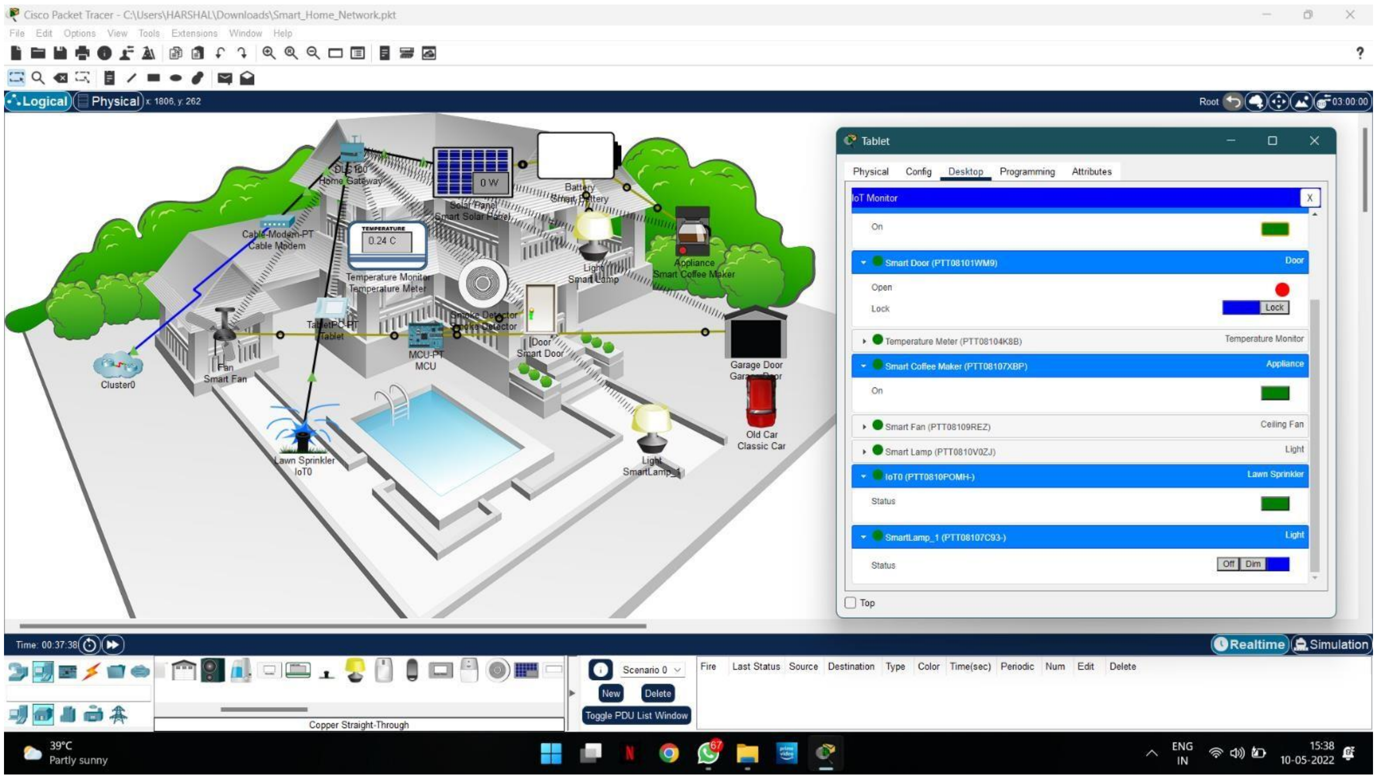
Home | Conditions | Editor | Log Out

● Smoke Detector (PTT08101J06)	Smoke Detector
● Garage Door (PTT0810QU96)	Garage Door
● Smart Door (PTT08101WM9)	Door
● Temperature Meter (PTT08104K8B)	Temperature Monitor
● Smart Coffee Maker (PTT08107XBP)	Appliance
● Smart Fan (PTT08109REZ)	Ceiling Fan
● Smart Lamp (PTT0810V0ZJ)	Light
● Sprinkler1 (PTT08108279)	Lawn Sprinkler
● Wind Detector (PTT0810GQQZ)	Wind Detector

Close the Tablet window.

Step 2: Experiment by adding other types of IoT devices to the smart home wireless network.

Photo:



Tablet

Physical | Config | Desktop | Programming | Attributes

IoT Monitor

On

Smart Door (PTT08101WM9) — Door
Open
Lock — Lock

Temperature Meter (PTT08104K8B) — Temperature Monitor

Smart Coffee Maker (PTT08107XBP) — Appliance
On

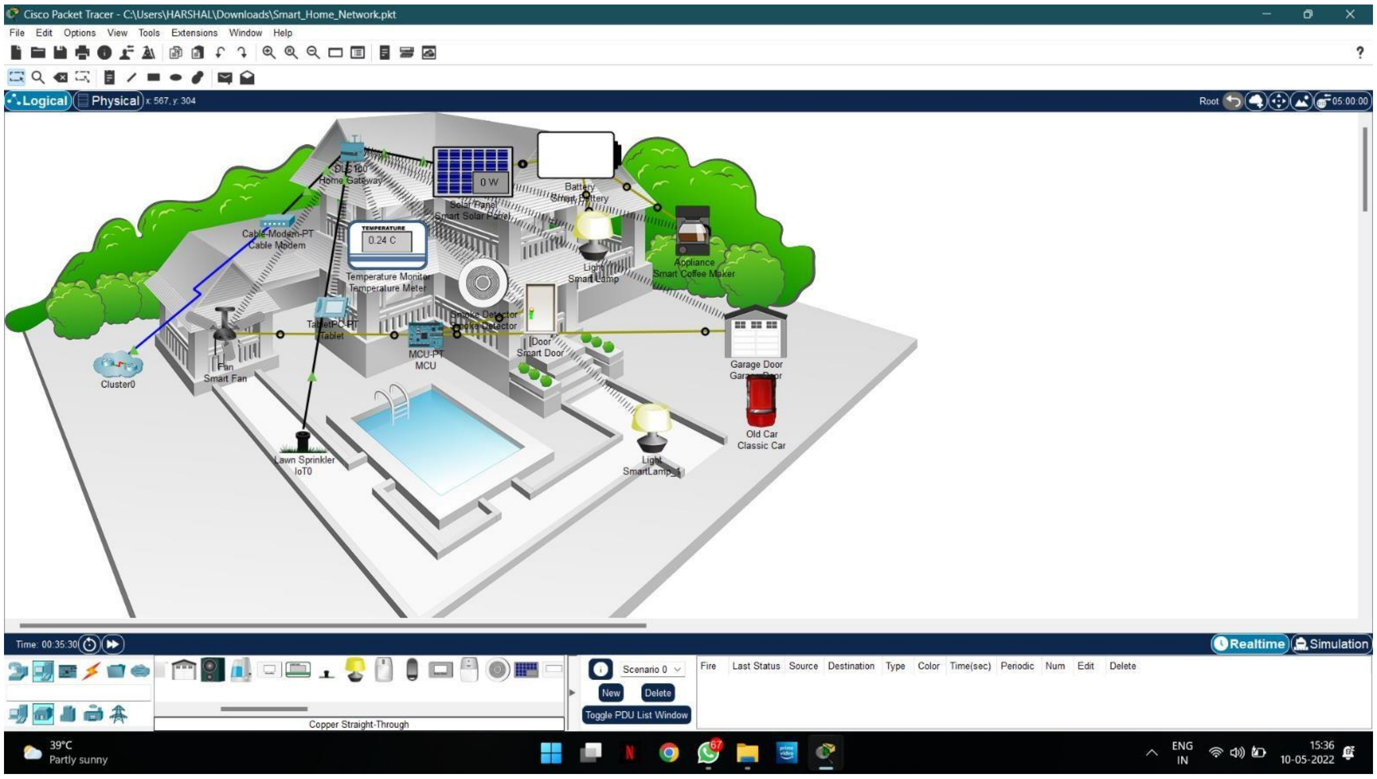
Smart Fan (PTT08109REZ) — Ceiling Fan

Smart Lamp (PTT0810V0ZJ) — Light

IoT0 (PTT0810POMH-) — Lawn Sprinkler
Status

SmartLamp_1 (PTT08107C93-) — Light
Status — Off | Dim

Top



Conclusion :

We were successfully able to create a cisco account and work with the existing home automation system, We verified the working of existing wired and wireless systems. We were also able to control the systems using the tablet. We also successfully added two new devices which were Lawn sprinkler and Smart lamp and also automated their working by making proper connections with the Home gateway.

Internet of Things (IoT) based Automatic Pet Feeder

Manasi Patil, Harshal Sonawane, Shashank Patil, Uma Thakur, Chaitanya Patil, Bhavin Patil

80 89 81

91 79 78

Abstract— **The Internet of Things (IoT) generates an extensive network of devices that frequently exchange data as the world becomes increasingly networked. Automation can be done independently by machines, but it can be augmented with monitoring and regulating capabilities with the help of IoT. While this interconnectedness occurs on a worldwide scale in businesses and organizations, it also occurs in individuals' homes. Consumers are increasingly interested in smart home devices and gadgets, which allow them to connect all of their devices for convenience and ease, comfort, energy efficiency, and, most importantly, personalization, which is one of the project's main goals. With the help of automation and IoT, the user's experience becomes even more individualized.**

Not everyone is a pet specialist, maintaining your pet's food can be difficult and time-consuming. Overeating and obesity are two of the most common health problems in pets. They are usually content with whatever is handed to them, especially when they are younger. Many adult pets are fed in an incorrect manner, which may result in a shorter lifetime. Another issue with feeding pets is that owners may not always be there. Being engaged with personal stuff while still caring for a starved young fellow at home is often a source of stress for owners. This paper presents an IoT enabled pet feeder system as the pet keeping is a time-consuming responsibility and we want to provide convenience to owners by helping them feed their pets easily and smartly.

Keywords—*Pet Feeder, Internet of Things, Automation, Blynk, Servo motor, NodeMCU ESP8266*

I. INTRODUCTION

This document describes The Internet of Things (IoT) based Automatic Pet Feeder.

Automation has recently been a technological revolution in demand on an industrial scale as well as in our daily life gadgets. Customers are more fascinated by automatic devices than anything else, and this is for the purpose of ease of use and time savings. Companies are working to meet these demands, and the automation industry is becoming better and much more developed each day. Automation is a means of automatically controlling and operating procedures using electronics and software that may be programmed and applied using machine learning technology. Automation is not a new concept; the first ATM machine was introduced in the 1960s, and with the help of such technology, the process became much easier, faster, and more convenient for the consumer. [1]

The majority of pet owners nowadays would like to enjoy their pets' company; some pet owners have the time and patience to feed their pets, while others do not. This is where automation and the Internet of Things (IoT) come in helpful to create a system that can satisfy the pet owner while causing no harm to the pets.

The automatic pet feeder can provide a fully personalization where the pet owner can program the feeding schedule, where the food can be dispensed at set times and in specific quantities, where previous research will be taken into account, where it will open up more opportunities to understand and learn from previous experiences, and the system can be upgraded with as many functions as possible
to satisfy the pet owner.[3]

Not every pet owner has the time to feed their animals as per their diet plan. In fact, most pet owners who work or study until late in the evening are not punctual in feeding their pets. In general, people underestimate the severity of this problem, so pet owners often solve the problem by overfilling the food dish with a large quantity of food. Not only busy pet owners do this, but also pet owners who are impatient in putting food for their pets three times a day and doing this. Pets, of course, need special attention and care. Pet care, however, becomes extremely difficult due to the hectic schedule.

In the proposed system, the automatic pet feeder, we are able to feed the pet automatically at the set time by the owner using Blynk application, servo motor and the ESP8266.

II. LITERATURE SURVEY

The goal of this literature review is to provide an overview of current knowledge in the fields of sensors, controllers, and automatic animal feed dispensers. A number of scholarly articles and conference proceedings were read and analyzed for any relevant material on the topic at hand as part of the comprehensive research. The use of technology to remotely control a sequence of processes is becoming increasingly common in today's society. It is possible to entirely automate and remove the feeding process from any personal physical necessity using MCUs and a well-built and designed pet feeder system, resulting in significant labor cost savings on a wider scale.

Current automatic pet feeders on the market meet some of our design goals, but they are extremely expensive. The Petwant SmartFeeder by GemTune has a smartphone app that allows for scheduled feeding and system configuration. When the pet is close by and it is time to feed, the Automatic Pet Feeder from Wireless Whiskers uses RFID to release food for the pet.[5]

In 2016, the paper "Design of Pet Feeder Using Web Server as Internet of Things Application" was published in the 2nd international conference on Electrical Engineering (IconIEE). This system was created to replace manual feeding with automated feeding. The primary flaw we discovered with this device was that it relied on a web application to monitor and feed pets automatically. Because this web application is not a good alternative, our device includes an Blynk App that can be used from anywhere.[6]

"The Study and Application of the IoT in Pet Systems" This paper was published online in January 2013. The smart pet door was the first gadget in the pet monitor system, and it can assist the pet owner in controlling their pet's behavior. The smart pet feeder is the other device. The pet owner might schedule the pet eating bowl time remotely with the help of the technology. They have provided a pet door as a downside, which may not be the best solution, but our equipment has a camera that is used for genuine monitoring.[7]

In 2017, the work "Automatic Pet Monitoring and Feeding System Using IoT" was published in the IJCR. This was a pet feeding system that served the same purpose as ours in terms of feeding the pet. However, the designer attempted to offer a new feature with this device, which was a pet collar that was used to track the whereabouts of the pet. The primary downside is that because it was designed for pets that are normally kept at home, having a tracker makes little sense. [8]

"Automatic Pet Feeder Using Arduino" was published in the International Journal of Innovative Research in Science and Technology (IJIRSET) on March 3rd, 2018. As the name implies, this was a system designed to automatically feed pets as a substitute to manual feeding. But the primary issue was that it was built on Arduino, which isn't necessarily a bad thing, but we prefer the NodeMCU. [9]

Moving away from traditional general feeders, more macro components, such as the mechanism for dispensing food from the container, must be examined.[10] This mechanism must be able to dispense food precisely while also being mechanically sound enough to prevent jamming and binding all along the food's path. In real-world circumstances, several dispensing systems were discovered. The paddle-based dispenser found in cereal dispensers is the first mechanism. These are straightforward, but they frequently bind and lack precise control. The sliding door dispenser is the second mechanism. This is a simple open and close sliding door that would rarely bind, but portions would be quite difficult to handle.

III. **PROPOSED METHODOLOGY**

In the proposed system, we have used the Blynk app, Arduino, servo motor, NodeMCU ESP8266. This system can be used to feed the the pet just by clicking a button on the Blynk application also the remainder can be set so that the pet owner will receive the mail as pop up as well about time to
feed the pet

Blynk Application

Blynk is a comprehensive software suite for developing, deploying, and remotely administering connected devices at any scale, including small IoT initiatives to millions of commercially available linked devices. It may be used to connect hardware to the cloud and create no-code iOS, Android, and web apps to analyze real-time and statistical data from devices, manage them remotely from anywhere on the planet, receive important notifications, and more. The Blynk mobile application will be used to control the servo motor that is attached to the pet feeding setup in this project.

Arduino IDE

The Arduino IDE is an open-source program for writing and uploading code to controller boards. In the Arduino IDE sketching refers to the process of writing a programme or code To upload the sketch written in the Arduino IDE software, we must link the controller boards to the IDE. The '.ino' extension is used to save the sketch.
In this project we are going to use the Arduino IDE on Windows OS. It will connect NodeMCU with the Blynk Application for automating the pet feeding process and to set the time of feeding

NodeMCU ESP8266

NodeMCU is an open-source platform based on the ESP8266 that allows devices to be connected and data to be transferred using the Wi-Fi protocol. We are using a NodeMCU ESP8266 as the main controller in this project to control the servo motor

Servo Motor

Servo motors, or "servos," are electronic devices having rotary or linear actuators that precisely rotate and push elements of a machine. Servos are primarily used to control angular or linear position, velocity, and acceleration. In this project we are using the servo motor to operate the pet feeder.

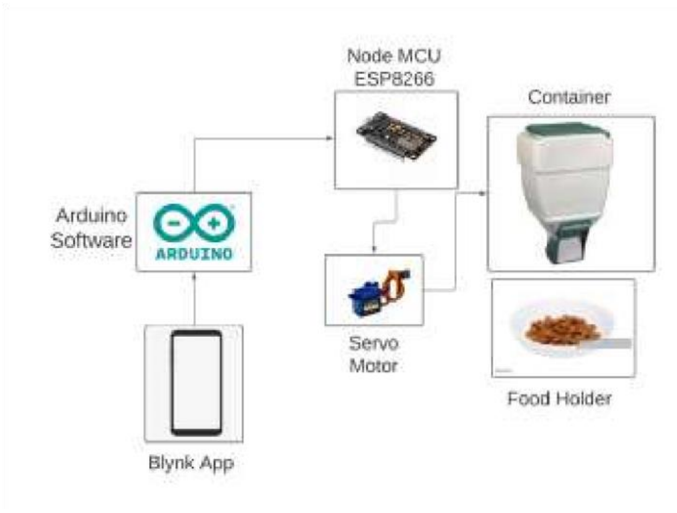
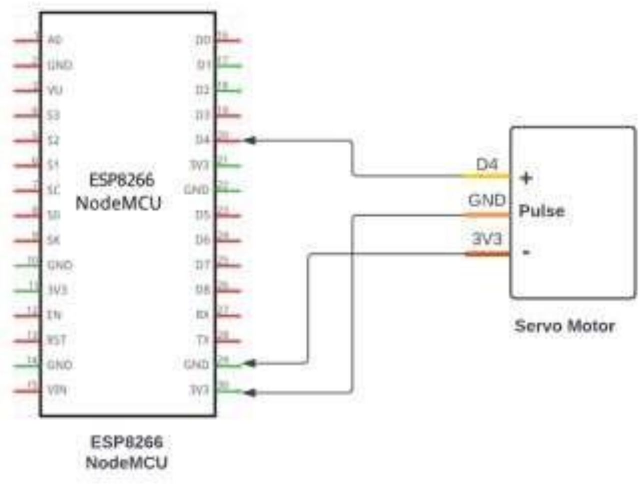


Fig 1: Block Diagram of pet feeder

As shown in the figure 1, we will be using the Blynk application to operate the pet feeder. We have created the template on the Blynk application and then added the device which will be connected to the NodeMCU. Arduino IDE is used to code to begin the Blynk Application and code to drive the motor. Program will be uploaded into the NodeMCU which later on can be provided with the current supply to work without any connectivity with the laptop. Servo motor will be implanted on the container that holds the food. Servo motor is basically used to operate the lid of the container that dispenses the food in the food container. Servo motor will be driven by the NodeMCU with 3.3V supply. Using the Blynk application, pet owners can set the time to feed the dog. Pet owners can set time for 3-4 times of the day. On the set time pet owner will receive the email and pop up about the time to feed the pet. So that pet owners can feed their pet no matter where they are on the time.

Fig 2: Circuit connectivity of NodeMCU ESP8266 with servo motor



Connections are as easy as it seems in figure 2. We have to connect the servo motor with the NodeMCU. Red wire of the servo motor which is the Vcc connects it to the 3.3V pin of the NodeMCU. Brown wire of the servo motor which is the GND connects it to the GND of the NodeMCU. Yellow wire which is the signal wire of the servo motor connect it to the D3 pin of NodeMCU

IV. **RESULTS**

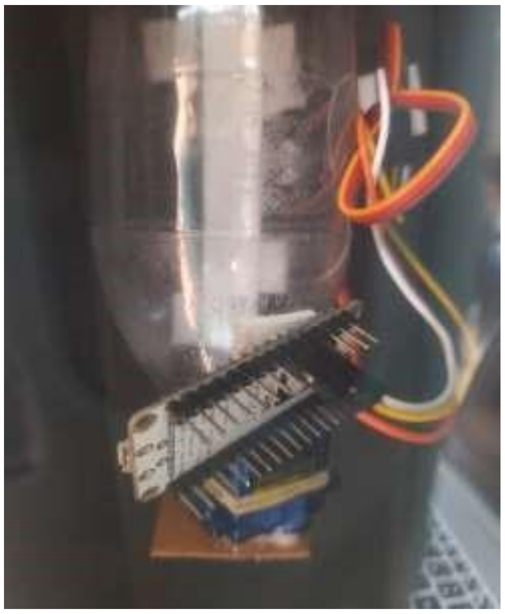
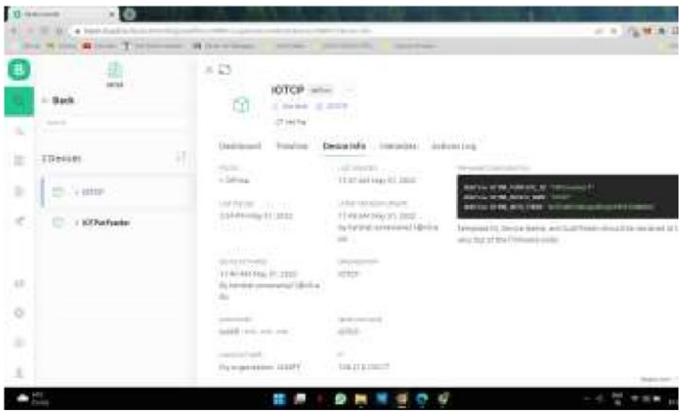


Fig 3: Pet Feeder Assembly



Back

2 Devices

IOTCP

KTRefleeder

IOTCP

Dashboard Timeline Device Info Metadata Actions Log

Offline

May 31, 2022

Template ID, Device Name, and AuthToken should be declared at the very top of the firmware code.

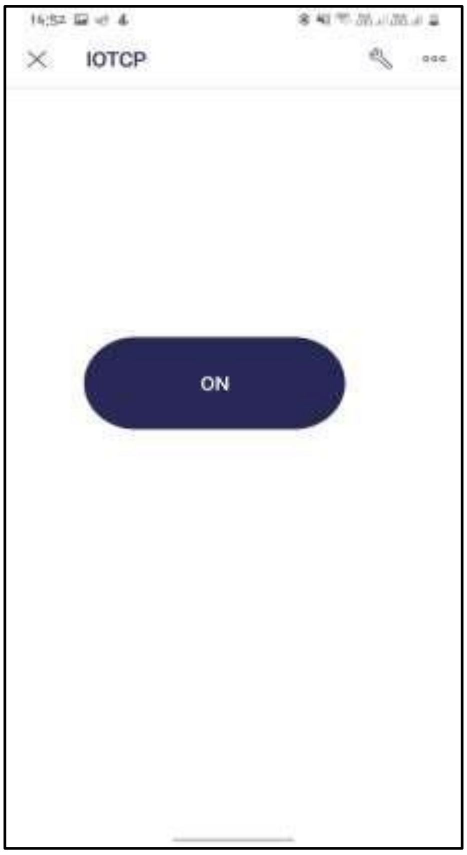
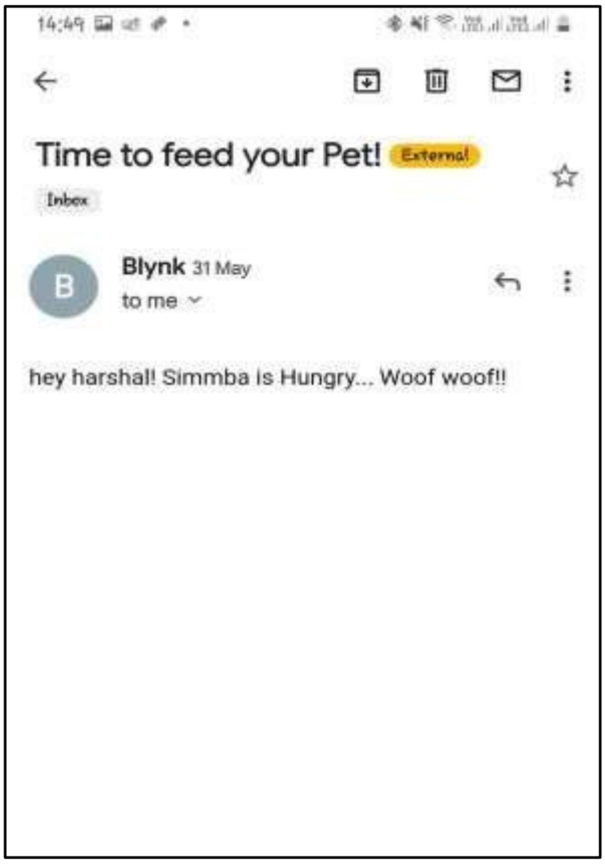


Fig 6: On-Off button



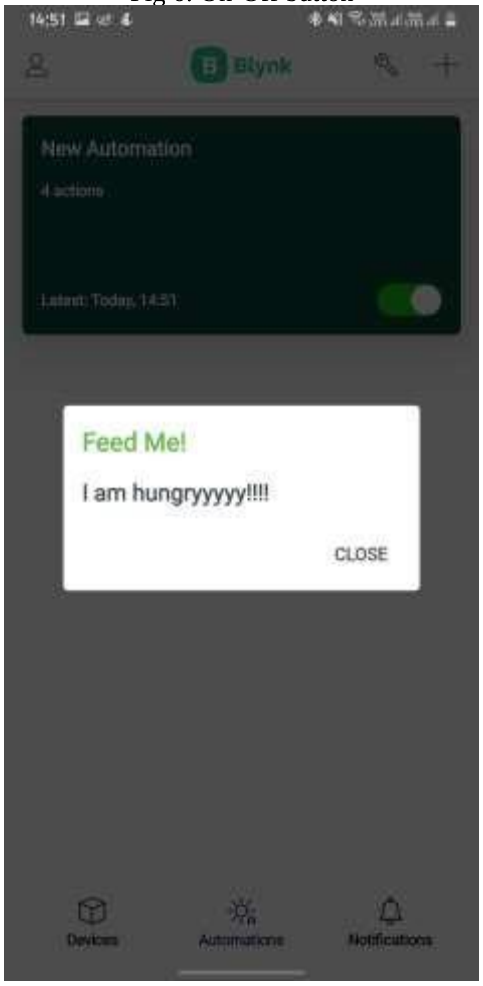
Time to feed your Pet! External

Inbox

Blynk 31 May

to me

hey harshal! Simmba is Hungry... Woof woof!!



Feed Me!

I am hungryyyyy!!!!

CLOSE

Fig 7: Owner receives pop up message

Fig 4: Blynk template

Fig 5: Owner receiving Email

Code is uploaded on the NodeMCU. Upon configuring the Blynk application , a template was created as shown in figure 4. The Blynk template was registered. Pet feeder is connected to the internet. When a button is clicked on the Blynk app the food gets dispensed from the container. Food will be dispensed as long as we hold the button. Apart from that the pet owner can set the reminder in the Blynk app to get us the pop up on the app at that particular time reminding him to feed the dog(figure-7). Also the pet owners receive an email about the same(figure-5).

V. Conclusions and Future Work

Most of the pet owners are not having enough time to feed their pet at the right time each day due to several reasons. The proposed system is to overcome this problem. Using this system one can feed their pet by using Blynk Mobile Application or web dashboard at any time and from anywhere. Just setting the time you can feed your pet. Pet Feeder is simple, efficient and economic.
Future work in this project can be done as the system can automatically detect when the pet is hungry and how much amount of food to be dispensed according to the weight of the pet. This can be achieved with the help of the Internet of Things.

References

[1] Raed Abdulla. 'IOT based Pet Feeder', March - April 2020, Volume 83, ISSN: 0193-4120

[2] Priya Mondal, Dr. Swapnili Karmore, Rajnandnee Parnami, "Design and development of IoT based Smart Pet Feeder", May 2020, Volume 9, pp. 6081-6083

[3] Zhuokai Zhao, Ziyun He, Fan Ling. "Automatic Pet Feeder Project, February". 2016

[4] Ahmed Mandy, Hassan Qazweeni, Mohammed Noureddine, Talal Al-Radhwan, Mohammed El-Abd, "Smart Pet House", 2020

[5] S. Subaashri, et al., "Automatic Pet Monitoring and Feeding System Using IoT", Volume.10 No.14, pp 253-258, 2017

[6] Soumallya Koley, Sneha Srimani, Debanjana Nandy, Pratik Pal, Samriddha Biswas, Dr. Indranath Sarkar, "Smart Pet Feeder". 2021 [7] Matt Smith, Chris Ranc, Shabab Siddiq, "Automated Cat feeder"2018

[8] San Luis Obispo, "Design And Build Of An Automated Animal Feed Dispenser", 2016

[9] Chung-Ming Own, Haw-Yun Shin, Chen-Ya Teng, "The Study and Application of the IoT in Pet Systems" 2013

[10] Andi Adriansyah, Muchd Wibowo, Eko Ihsanto, "Design of Pet Feeder using Web Server as Internet of Things Application", 2016

[11] Mritunjay Tiwari , Sahil Hawal , Nikhil Mhatre, "Automatic Pet Feeder Using Arduino", Vol. 7, Issue 3, March 2018

[12] M. Todica, "Controlling Arduino board with smartphone and Blynk via internet ", 2016

[13] Vaibhav Malav, "Research Paper On Home Automation Using Arduino", April 2019

[14] Abdul Wali Abdul Ali, "A Review on The AC Servo Motor Control Systems", 2020

[15] Joao Mesquita, Diana Guimarãe, Carlos Pereira, Frederico Miguel Santos, "Assessing the ESP8266 WiFi module for the Internet of Things",
2018