

# **NITTE MEENAKSHI INSTITUTE OF TECHNOLOGY**

(AN AUTONOMOUS INSTITUTION, AFFILIATED TO VISVESVARAYA TECHNOLOGICAL  
UNIVERSITY, BELGAUM, APPROVED BY AICTE & GOVT.OF KARNATAKA)



## **Internet of Things Lab Record**

*Submitted in partial fulfilment of the requirement for the award of  
Degree of*

*Bachelor of Engineering*

*in*

*Artificial Intelligence and Machine Learning*

**Submitted by:**

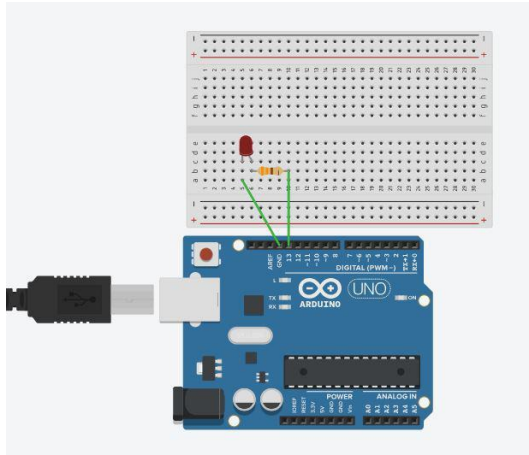
**Swathika Kannan 1NT21AI061**

# Contents

1. Onboard and External LED Blink using Arduino uno.....	1
2. Ultrasonic sensor using Arduino UNO.....	3
3. Traffic light using Arduino UNO.....	5
4. Onboard and External LED blink using ESP8266.....	9
5. Humidity and Temperature using ThingsSpeak, DHT11 and NodeMCU.....	11
6. Liquid Crystal LCD with ESP8266.....	17
7. External LED Blink using Blynk App and ESP8266.....	19
8. Water Level monitoring using NodeMCU and Ultrasonic.....	23
9. LoRa using NodeMCU(ESP8266).....	25

## Lab Experiment 1

### Onboard and External LED Blink using Arduino uno



#### Onboard Code

```
void setup() {  
    pinMode(LED_BUILTIN, OUTPUT);  
}  
  
void loop() {  
    digitalWrite(LED_BUILTIN, HIGH);  
    delay(1000);  
    digitalWrite(LED_BUILTIN, LOW);  
    delay(1000);  
}
```

#### External LED

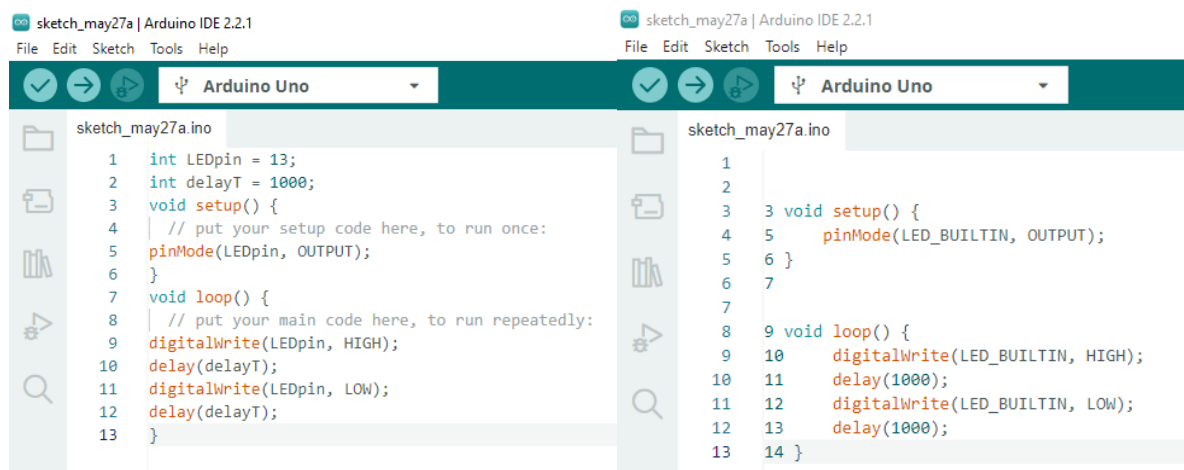
```
int LEDpin = 13;  
int delayT = 1000;  
void setup() {  
    // put your setup code here, to run once:  
    pinMode(LEDpin, OUTPUT);
```

```

}

void loop() {
    // put your main code here, to run repeatedly:
    digitalWrite(LEDpin, HIGH);
    delay(delayT);
    digitalWrite(LEDpin, LOW);
    delay(delayT);
}

```



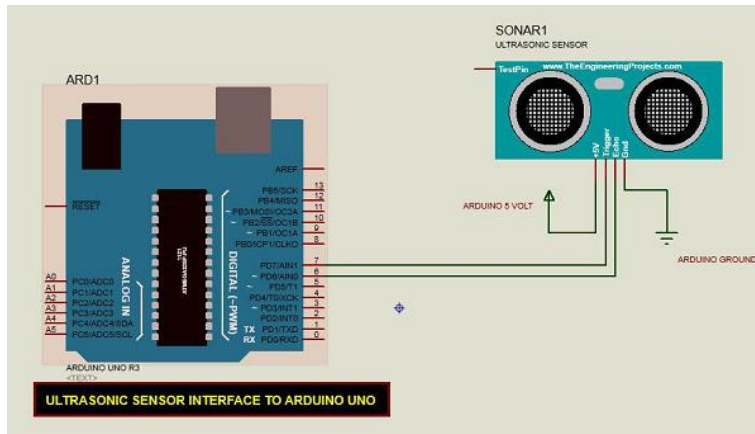
## OUTPUT:

For Onboard the led blinking at regular intervals of 1s.

For external LED, the light blinking at regular interval of 1s.

## Lab Experiment 2

### Ultrasonic sensor using Arduino UNO



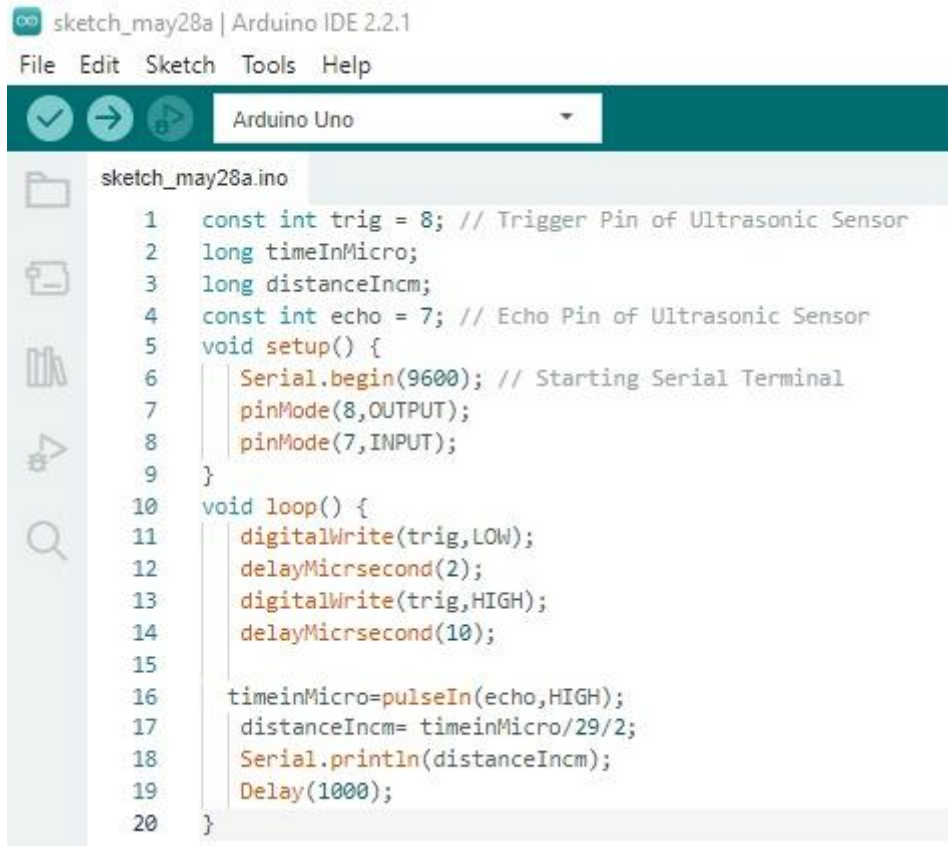
#### CODE:

```
const int trig = 8; // Trigger Pin of Ultrasonic Sensor
long timeInMicro;
long distanceIncm;
const int echo = 7; // Echo Pin of Ultrasonic Sensor
void setup() {
  Serial.begin(9600); // Starting Serial Terminal
  pinMode(8,OUTPUT);
  pinMode(7,INPUT);
}
void loop() {
  digitalWrite(trig,LOW);
  delayMicrosecond(2);
  digitalWrite(trig,HIGH);
  delayMicrosecond(10);
```

```

timeinMicro=pulseIn(echo,HIGH);
distanceIncm= timeinMicro/29/2;
Serial.println(distanceIncm);
Delay(1000);
}

```



```

sketch_may28a.ino
1  const int trig = 8; // Trigger Pin of Ultrasonic Sensor
2  long timeInMicro;
3  long distanceIncm;
4  const int echo = 7; // Echo Pin of Ultrasonic Sensor
5  void setup() {
6      Serial.begin(9600); // Starting Serial Terminal
7      pinMode(8,OUTPUT);
8      pinMode(7,INPUT);
9  }
10 void loop() {
11     digitalWrite(trig,LOW);
12     delayMicrosecond(2);
13     digitalWrite(trig,HIGH);
14     delayMicrosecond(10);
15
16     timeinMicro=pulseIn(echo,HIGH);
17     distanceIncm= timeinMicro/29/2;
18     Serial.println(distanceIncm);
19     Delay(1000);
20 }

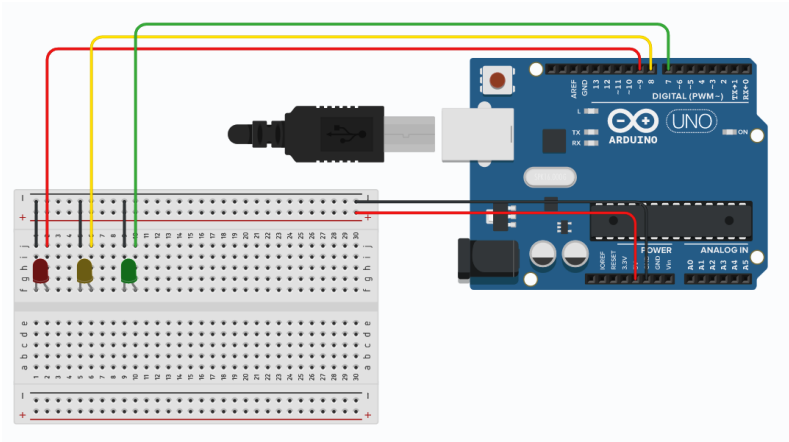
```

## OUTPUT:

The distance measured by sensor in inches and cm on Arduino serial monitor.

## Lab Experiment 3

### Traffic light using Arduino UNO



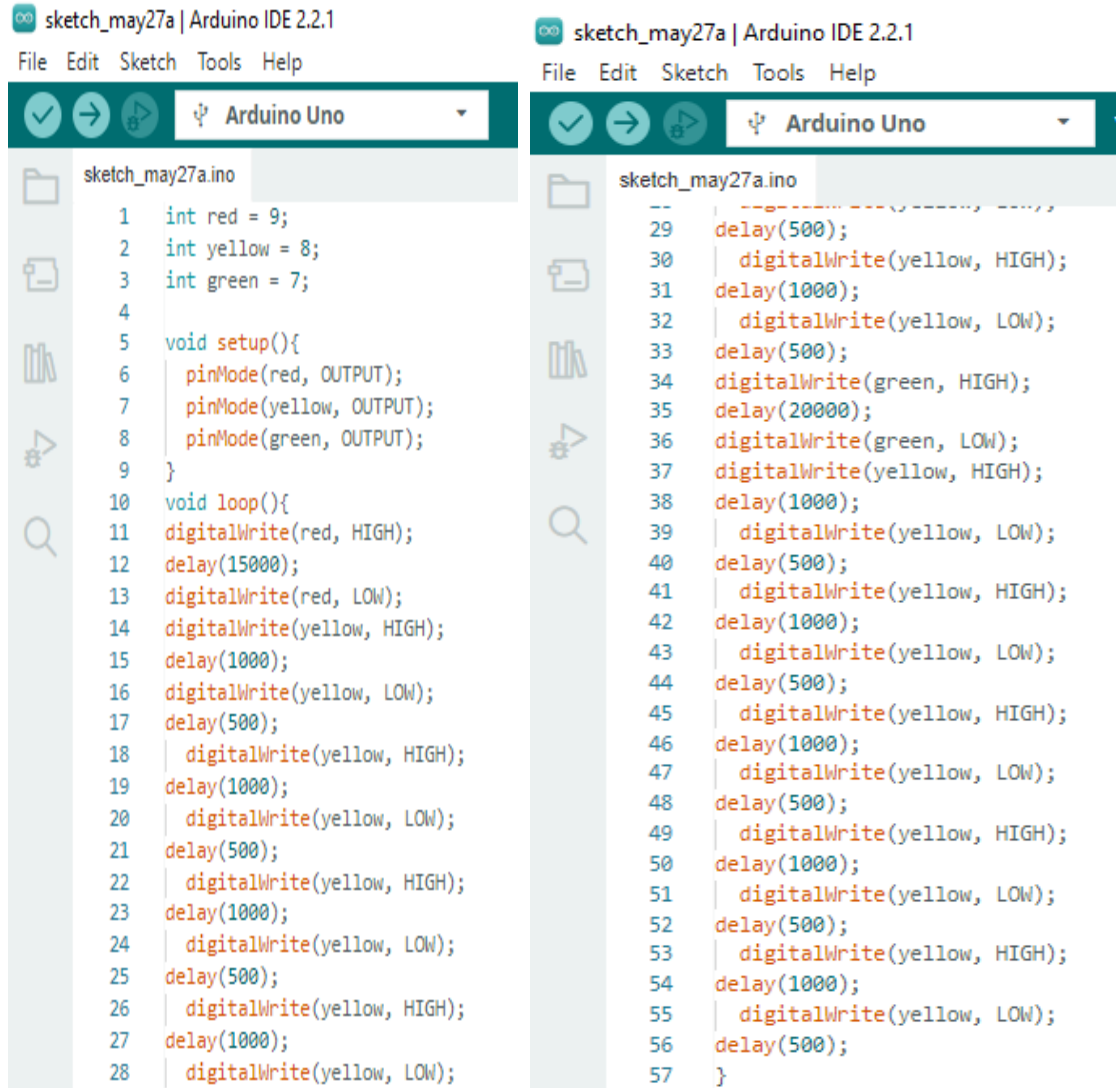
#### CODE:

```
int red = 9;
int yellow = 8;
int green = 7;
void setup(){
  pinMode(red, OUTPUT);
  pinMode(yellow, OUTPUT);
  pinMode(green, OUTPUT);
}
void loop(){
  digitalWrite(red, HIGH);
  delay(15000);
  digitalWrite(red, LOW);
  digitalWrite(yellow, HIGH);
  delay(1000);
  digitalWrite(yellow, LOW);
  delay(500);
```

```
digitalWrite(yellow, HIGH);  
delay(1000);  
digitalWrite(yellow, LOW);  
delay(500);  
digitalWrite(yellow, HIGH);  
delay(1000);  
digitalWrite(yellow, LOW);  
delay(500);  
digitalWrite(yellow, HIGH);  
delay(1000);  
digitalWrite(yellow, LOW);  
delay(500);  
digitalWrite(yellow, HIGH);  
delay(1000);  
digitalWrite(yellow, LOW);  
delay(500);  
digitalWrite(green, HIGH);  
delay(20000);  
digitalWrite(green, LOW);  
digitalWrite(yellow, HIGH);  
delay(1000);  
digitalWrite(yellow, LOW);  
delay(500);  
digitalWrite(yellow, HIGH);  
delay(1000);  
digitalWrite(yellow, LOW);  
delay(500);
```



```
    digitalWrite(yellow, HIGH);  
    delay(1000);  
    digitalWrite(yellow, LOW);  
    delay(500);  
    digitalWrite(yellow, HIGH);  
    delay(1000);  
    digitalWrite(yellow, LOW);  
    delay(500);  
    digitalWrite(yellow, HIGH);  
    delay(1000);  
    digitalWrite(yellow, LOW);  
    delay(500);  
}
```



## OUTPUT:

The red LED is initially on for 15s then it turns off, followed by the yellow LED which is on for 1s and turns off. The yellow LED is turned on for 0.5s for 5 times and turns off and finally the green LED which glows for 20s. This repeats on loop and variations of time delay can be made.

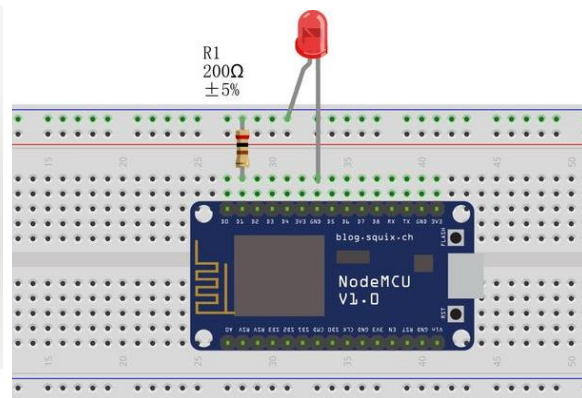
## Lab Experiment 4

### Onboard and External LED blink using ESP8266

Onboard



External



#### External:

```
#define LED D0
```

```
void setup()
```

```
{
```

```
  pinMode(LED, OUTPUT); //LED pin as output
```

```
}
```

```
void loop()
```

```
{
```

```
  digitalWrite(LED, HIGH); //turn the led off
```

```
  delay(1000); //wait for 1 sec
```

```
  digitalWrite(LED, LOW); //turn the led on
```

```
  delay(1000); //wait for 1 sec
```

```
}
```

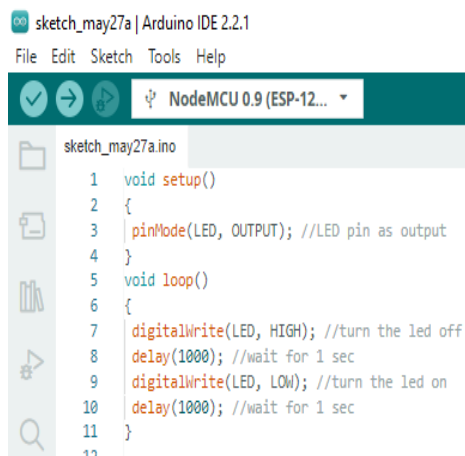
**Onboard:**

```

void setup()
{
  pinMode(LED_BUILTIN, OUTPUT); //LED_BUILTIN pin as an output
}

void loop()
{
  digitalWrite(LED_BUILTIN, LOW); // Turn the LED on
  delay(1000);                    // Wait for a second
  digitalWrite(LED_BUILTIN, HIGH); // Turn the LED off
  delay(2000);                    // Wait for two seconds
}

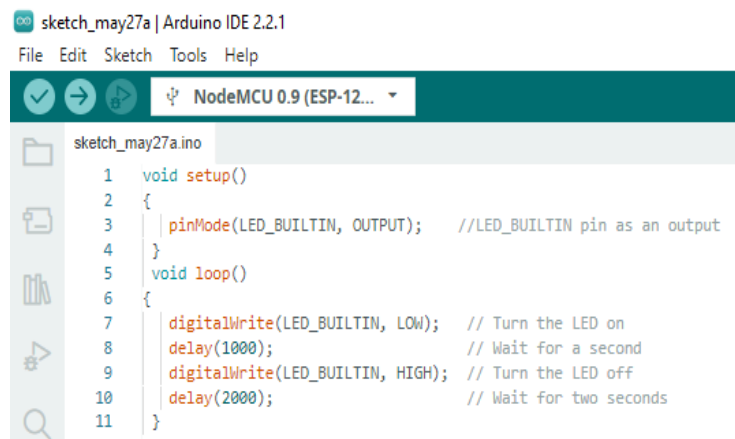
```

**External**


```

sketch_may27a | Arduino IDE 2.2.1
File Edit Sketch Tools Help
NodeMCU 0.9 (ESP-12...
sketch_may27a.ino
1 void setup()
2 {
3   pinMode(LED, OUTPUT); //LED pin as output
4 }
5 void loop()
6 {
7   digitalWrite(LED, HIGH); //turn the led off
8   delay(1000); //wait for 1 sec
9   digitalWrite(LED, LOW); //turn the led on
10  delay(1000); //wait for 1 sec
11 }
12

```

**Onboard**


```

sketch_may27a | Arduino IDE 2.2.1
File Edit Sketch Tools Help
NodeMCU 0.9 (ESP-12...
sketch_may27a.ino
1 void setup()
2 {
3   pinMode(LED_BUILTIN, OUTPUT); //LED_BUILTIN pin as an output
4 }
5 void loop()
6 {
7   digitalWrite(LED_BUILTIN, LOW); // Turn the LED on
8   delay(1000); // Wait for a second
9   digitalWrite(LED_BUILTIN, HIGH); // Turn the LED off
10  delay(2000); // Wait for two seconds
11 }
12

```

**OUTPUT:**

For Onboard the led blinking at regular intervals of 2s and is switched on for 1s.

For external LED, the light blinking at regular interval of 1s and is switched on for 1s.

## Lab Experiment 5

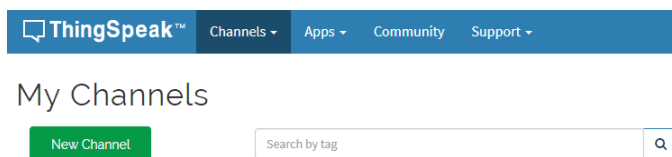
### Humidity and Temperature measurement using ThingsSpeak, DHT11 and NodeMCU

ThingsSpeak:

ThingSpeak is used for temperature and humidity IoT projects due to its user-friendly interface, real-time data visualization, and cloud-based data storage. It integrates seamlessly with MATLAB for advanced data analysis. The platform provides robust APIs for flexible data handling and has strong community support and documentation. It offers a free tier and scalable options for various project sizes.

Steps for setting up the ThinkSpeak App:

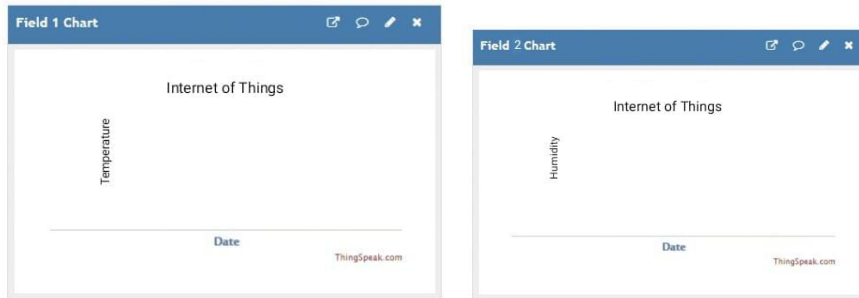
1. Create a ThingSpeak account.
2. Create a new channel:



3. You can store up to 8 fields on 1 channel. We will store 1 parameter

 A screenshot of the 'New Channel' form in the ThingSpeak interface. The form has a title 'New Channel'. It includes a 'Name' field with the text 'Temp and Humidity', a 'Description' field with a green plus icon, and eight 'Field' entries. 'Field 1' is set to 'Temperature' with a checked checkbox. 'Field 2' is set to 'Humidity' with an unchecked checkbox. Fields 3 through 8 are empty with unchecked checkboxes. At the bottom, there is a 'Metadata' field.

4. When a new channel is created, you can see four graphs for each parameter:



5. Get the Channel ID of your Channel (To be used while connecting to the channel):

Channel ID: 733269  
 Author: pve1111111111  
 Access: Private

6. Get the Read and Write API from API tab:

Write API Key

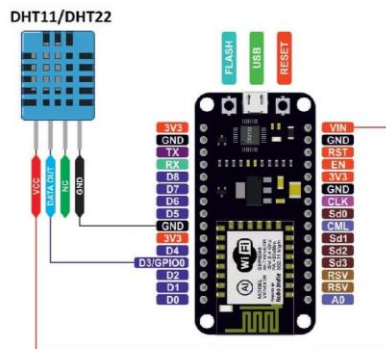
Key: Y1Q18HZFWHNQUW1R

Generate New Write API Key

Read API Keys

Key: VVFUIELNOW0IOVXB

## Circuit Diagram:



**CODE:**

```

#include <DHT.h> // Including library for dht
#include <ESP8266WiFi.h>

String apiKey = "H38TEGNC0XKW43BB"; //Enter your Write API key from
ThingSpeak

const char *ssid = "how2electronics"; // replace with your wifi ssid and wpa2 key
const char *pass = "alhabibi";
const char* server = "api.thingspeak.com";
#define DHTPIN 0 //pin where the dht11 is connected
DHT dht(DHTPIN, DHT11);
WiFiClient client;

void setup()
{
    Serial.begin(115200);
    delay(10);
    dht.begin();
    Serial.println("Connecting to ");
    Serial.println(ssid);
    WiFi.begin(ssid, pass);
    while (WiFi.status() != WL_CONNECTED)
    {
        delay(500);
        Serial.print(".");
    }
    Serial.println("");
    Serial.println("WiFi connected");
}

```

```

void loop()
{
    float h = dht.readHumidity();
    float t = dht.readTemperature();
    if (isnan(h) || isnan(t))
    {
        Serial.println("Failed to read from DHT sensor!");
        return;
    }

    if (client.connect(server,80)) // "184.106.153.149" or
api.thingspeak.com
    {
        String postStr = apiKey;
        postStr += "&field1=";
        postStr += String(t);
        postStr += "&field2=";
        postStr += String(h);
        postStr += "\r\n\r\n";
        client.print("POST /update HTTP/1.1\n");
        client.print("Host: api.thingspeak.com\n");
        client.print("Connection: close\n");
        client.print("X-THINGSPEAKAPIKEY: "+apiKey+"\n");
        client.print("Content-Type: application/x-www-form-
urlencoded\n");
        client.print("Content-Length: ");
        client.print(postStr.length());
        client.print("\n\n");
    }
}

```



```

    client.print(postStr);

    Serial.print("Temperature: ");

    Serial.print(t);

    Serial.print(" degrees Celcius, Humidity: ");

    Serial.print(h);

    Serial.println("%. Send to Thingspeak.");

}

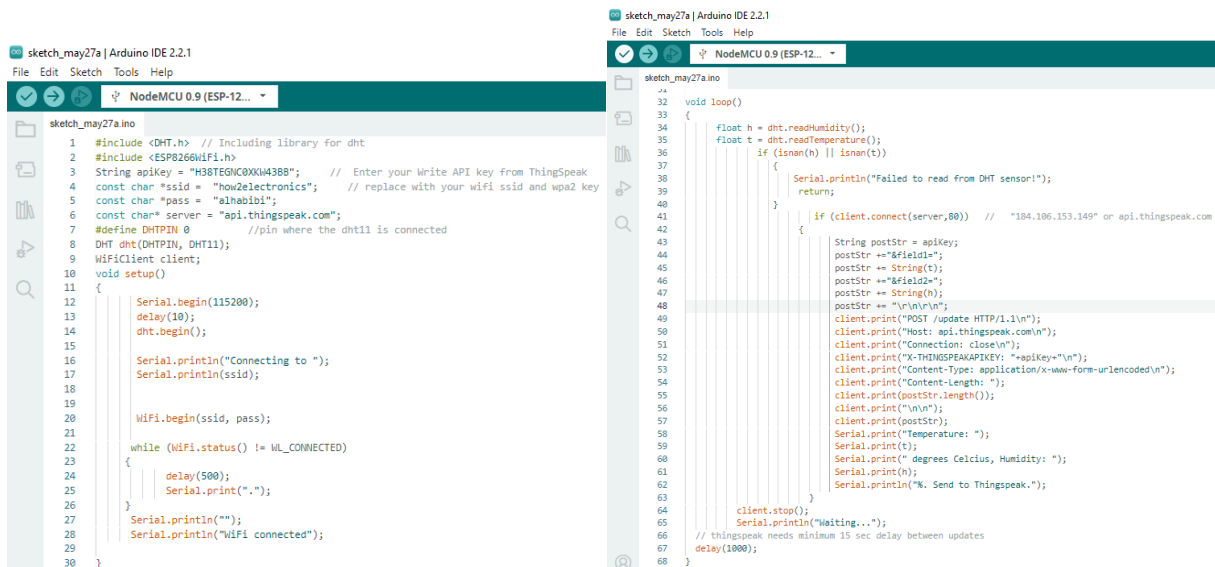
client.stop();

Serial.println("Waiting...");

// thingspeak needs minimum 15 sec delay between updates
delay(1000);

}

```



## OUTPUT:

```

DHT11_Using_Thingspeak

#include <DHT.h> // Including library for dht
#include <ESP8266WiFi.h>

String apiKey = "H38TEGHC0XKW438B"; // Enter your Write API key from Thingspeak

const char *ssid = "how2electronics"; // replace with your wifi ssid and wpa2 key
const char *pass = "alhabibi";
const char* server = "api.thingspeak.com";

#define DHTPIN 0 //pin where the dht11 is connected

DHT dht(DHTPIN, DHT11);

WiFiClient client;

void setup()
{
  Serial.begin(115200);
  delay(10);
  dht.begin();

  Serial.println("Connecting to ");
  Serial.println(ssid);

  WiFi.begin(ssid, pass);

  while (WiFi.status() != WL_CONNECTED)
  {
    delay(500);
    Serial.print(".");
  }
  Serial.println("\nWiFi connected");
}

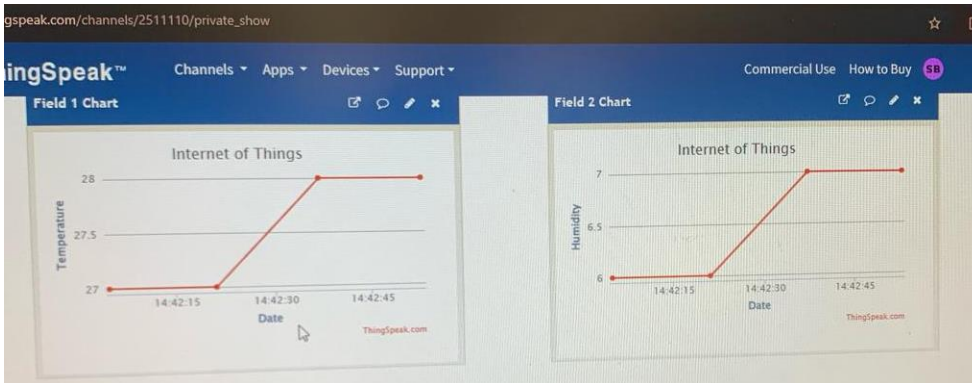
void loop()
{
  float temperature = dht.temperature();
  float humidity = dht.humidity();

  String url = "http://api.thingspeak.com/update?apikey=" + apiKey + "&field1=" + temperature + "&field2=" + humidity;

  HTTPClient http;
  http.begin(url);
  http.POST("");

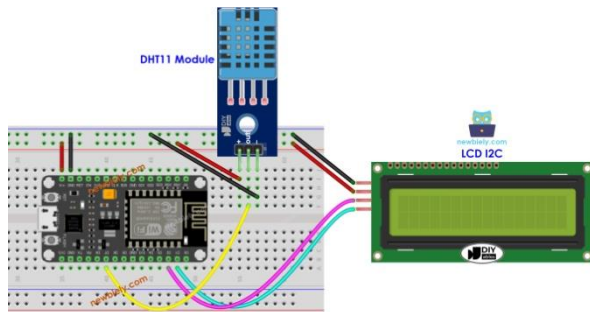
  Serial.println("Temperature: " + String(temperature) + " degrees Celcius, Humidity: " + String(humidity) + "% Send to Thingspeak.");
  Serial.println("Waiting...");
  delay(10000);
}

```



## Lab Experiment 6

### Liquid Crystal LCD with ESP8266



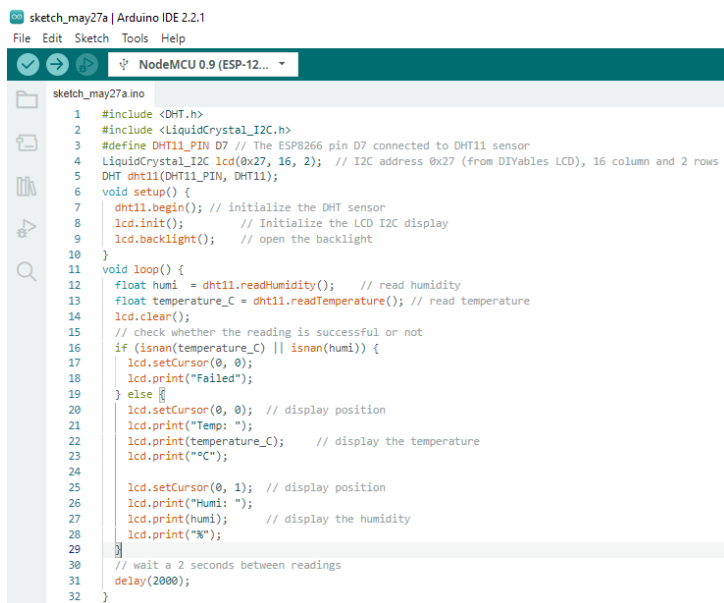
#### CODE:

```
#include <DHT.h>
#include <LiquidCrystal_I2C.h>
#define DHT11_PIN D7 // The ESP8266 pin D7 connected to DHT11 sensor
LiquidCrystal_I2C lcd(0x27, 16, 2); // I2C address 0x27 (from DIYables LCD),
16 column and 2 rows
DHT dht11(DHT11_PIN, DHT11);
void setup() {
    dht11.begin(); // initialize the DHT sensor
    lcd.init();    // Initialize the LCD I2C display
    lcd.backlight(); // open the backlight
}
void loop() {
    float humi = dht11.readHumidity(); // read humidity
    float temperature_C = dht11.readTemperature(); // read temperature
    lcd.clear();
    // check whether the reading is successful or not
    if (isnan(temperature_C) || isnan(humi)) {
        lcd.setCursor(0, 0);
```

```

    lcd.print("Failed");
} else {
    lcd.setCursor(0, 0); // display position
    lcd.print("Temp: ");
    lcd.print(temperature_C); // display the temperature
    lcd.print("°C");
    lcd.setCursor(0, 1); // display position
    lcd.print("Humi: ");
    lcd.print(humi); // display the humidity
    lcd.print("%");
}
// wait a 2 seconds between readings
delay(2000);
}

```



```

1 #include <DHT.h>
2 #include <LiquidCrystal_I2C.h>
3 #define DHT11_PIN D7 // The ESP8266 pin D7 connected to DHT11 sensor
4 LiquidCrystal_I2C lcd(0x27, 16, 2); // I2C address 0x27 (from DIYables LCD), 16 column and 2 rows
5 DHT dht11(DHT11_PIN, DHT11);
6 void setup() {
7   dht11.begin(); // initialize the DHT sensor
8   lcd.init(); // Initialize the LCD I2C display
9   lcd.backlight(); // open the backlight
10 }
11 void loop() {
12   float humi = dht11.readHumidity(); // read humidity
13   float temperature_C = dht11.readTemperature(); // read temperature
14   lcd.clear();
15   // check whether the reading is successful or not
16   if (isnan(temperature_C) || isnan(humi)) {
17     lcd.setCursor(0, 0);
18     lcd.print("Failed");
19   } else {
20     lcd.setCursor(0, 0); // display position
21     lcd.print("Temp: ");
22     lcd.print(temperature_C); // display the temperature
23     lcd.print("°C");
24
25     lcd.setCursor(0, 1); // display position
26     lcd.print("Humi: ");
27     lcd.print(humi); // display the humidity
28     lcd.print("%");
29   }
30   // wait a 2 seconds between readings
31   delay(2000);
32 }

```

## OUTPUT:

Displays the humidity and temperature on the LCD display and is monitored every 2 seconds.

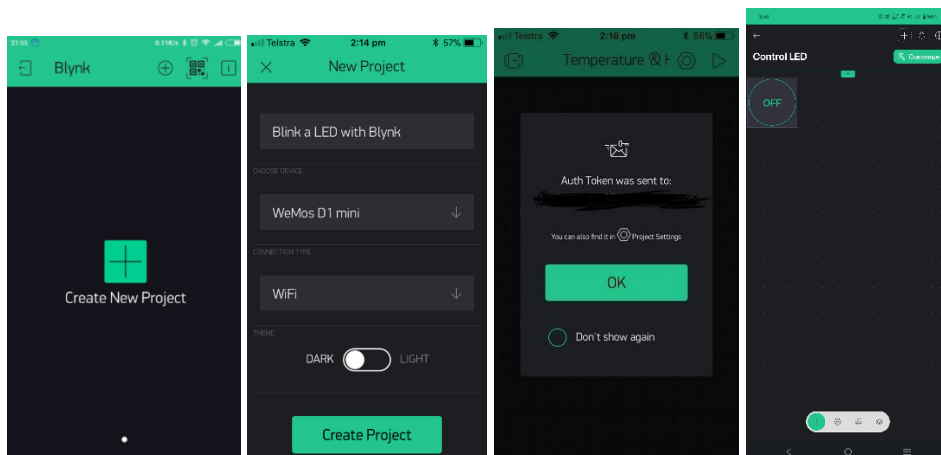
## Lab Experiment 7

### External LED Blink using Blynk App and ESP8266

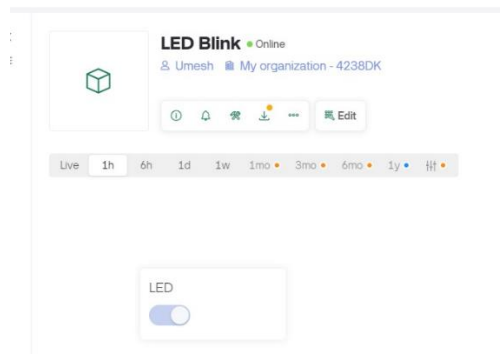


Configuring Blynk App:

1. Download and install the Blynk app from the Google Play Store, then create an account using your Gmail.
2. Create a new project named "Control LED," select ESP32 Dev Board, choose Wi-Fi as the connection type, and click Create.



3. Check your email for the authentication token sent by Blynk.
4. On the canvas window, tap to open the widget box and add a Button widget.
5. Set the Button widget to control GPIO2 in switch mode, then press Play to interact with the hardware.

**CODE:**

```
#define BLYNK_TEMPLATE_ID "TMPL3iOcDda6Z"
#define BLYNK_DEVICE_NAME "Control LED"
#define BLYNK_PRINT Serial
#include <ESP8266WiFi.h>
#include <BlynkSimpleEsp8266.h>
#define BLYNK_AUTH_TOKEN "kYSJcSlRqpHsPMkZgbaHbBnYms2nleaQ"
char auth[] = BLYNK_AUTH_TOKEN;
char ssid[] = "Nokia G21";//Enter your WIFI name
char pass[] = "bandi12345";//Enter your WIFI password
int ledpin = D4;
//Get the button value
BLYNK_WRITE(V0) {
    digitalWrite(ledpin, param.asInt());
}
void setup() {
    pinMode(ledpin, OUTPUT);
    Blynk.begin(auth, ssid, pass, "blynk.cloud", 80);
    WiFi.begin(ssid, pass);
    Blynk.config(BLYNK_AUTH_TOKEN);
}
```

```

void loop() {
  //Run the Blynk library
  Blynk.run();
}

```

sketch\_may27a | Arduino IDE 2.2.1

File Edit Sketch Tools Help

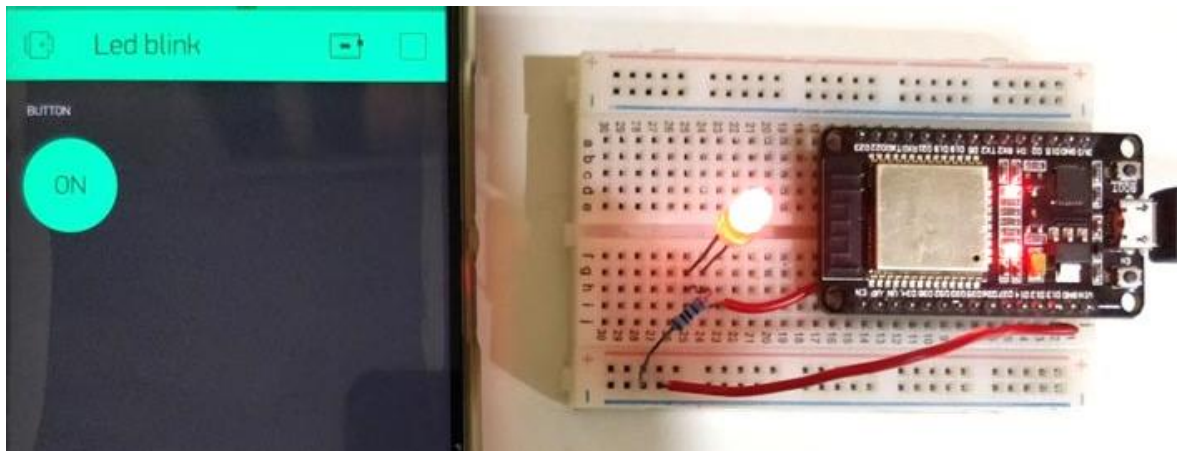
NodeMCU 0.9 (ESP-12...

sketch\_may27a.ino

```

1  #define BLYNK_TEMPLATE_ID "TMPL3i0cDda6Z"
2  #define BLYNK_DEVICE_NAME "Control LED"
3  #define BLYNK_PRINT Serial
4  #include <ESP8266WiFi.h>
5  #include <BlynkSimpleEsp8266.h>
6  #define BLYNK_AUTH_TOKEN "kYSJcS1RqpHsPMkZgbaHb8nYms2nleaQ" //Enter your blynk auth token
7  char auth[] = BLYNK_AUTH_TOKEN;
8  char ssid[] = "Nokia G21";//Enter your WIFI name
9  char pass[] = "bandi12345";//Enter your WIFI password
10 int ledpin = D4;
11 //Get the button value
12 BLYNK_WRITE(V0) {
13   digitalWrite(ledpin, param.asInt());
14 }
15
16 void setup() {
17   //Set the LED pin as an output pin
18   pinMode(ledpin, OUTPUT);
19   //Initialize the Blynk library
20   Blynk.begin(auth, ssid, pass, "blynk.cloud", 80);
21   WiFi.begin(ssid, pass);
22   Blynk.config(BLYNK_AUTH_TOKEN);
23 }
24
25 void loop() {
26   //Run the Blynk library
27   Blynk.run();
28 }

```



```

ledblinkmarch.ino
24

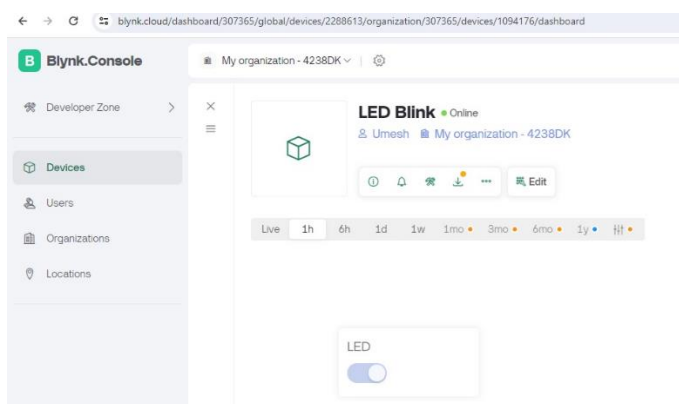
Output
Features: WiFi
Crystal is 26MHz
MAC: cc:50:e3:70:dd:1e
Uploading stub...
Running stub...
Stub running...
Configuring flash size...
Auto-detected Flash size: 4MB
Compressed 286592 bytes to 209972...
Writing at 0x00000000... (7 %)
Writing at 0x00004000... (15 %)
Writing at 0x00008000... (23 %)
Writing at 0x0000c000... (30 %)
Writing at 0x00010000... (38 %)
Writing at 0x00014000... (46 %)
Writing at 0x00018000... (53 %)
Writing at 0x0001c000... (61 %)
Writing at 0x00020000... (69 %)
Writing at 0x00024000... (76 %)
Writing at 0x00028000... (84 %)
Writing at 0x0002c000... (92 %)
Writing at 0x00030000... (100 %)
Wrote 286592 bytes (209972 compressed) at 0x00000000 in 18.7 seconds (effective 122.5 kbit/s)...
Hash of data verified.

Leaving...
Hard resetting via RTS pin...

```

## OUTPUT:

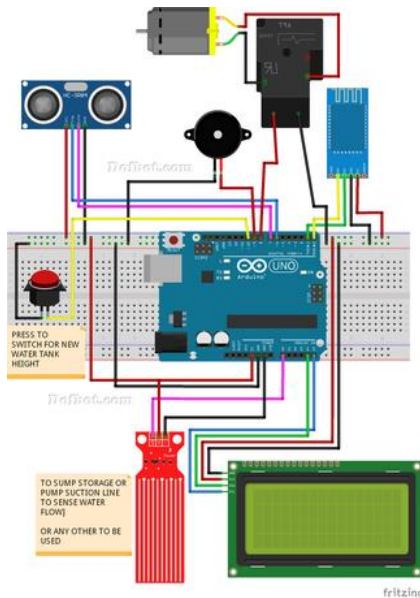
The LED turns on and off based on the switch of the blynk app and is also displayed on the interface.





## Lab Experiment 8

### Water Level monitoring using NodeMCU and Ultrasonic



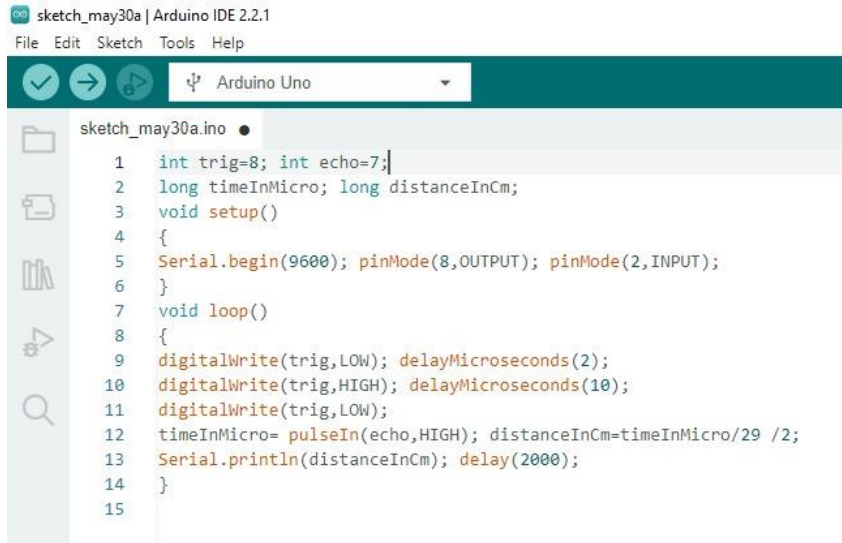
#### CODE:

```
int trig=8; int echo=7;

long timeInMicro; long distanceInCm;

void setup(){
  Serial.begin(9600); pinMode(8,OUTPUT); pinMode(2,INPUT);
}

void loop(){
  digitalWrite(trig,LOW); delayMicroseconds(2);
  digitalWrite(trig,HIGH); delayMicroseconds(10);
  digitalWrite(trig,LOW);
  timeInMicro= pulseIn(echo,HIGH); distanceInCm=timeInMicro/29 /2;
  Serial.println(distanceInCm); delay(2000);
}
```



```

sketch_may30a.ino
1  int trig=8; int echo=7;
2  long timeInMicro; long distanceInCm;
3  void setup()
4  {
5  Serial.begin(9600); pinMode(8,OUTPUT); pinMode(2,INPUT);
6  }
7  void loop()
8  {
9  digitalWrite(trig,LOW); delayMicroseconds(2);
10 digitalWrite(trig,HIGH); delayMicroseconds(10);
11 digitalWrite(trig,LOW);
12 timeInMicro= pulseIn(echo,HIGH); distanceInCm=timeInMicro/29 /2;
13 Serial.println(distanceInCm); delay(2000);
14 }
15

```

## OUTPUT

The distance between the water level and the place of the sensor is calculated and displayed.



```

sketch_jun6a.ino
3  long timeInMicro; long distanceInCm;

```

Output Serial Monitor ×

Not connected. Select a board and a port to connect automatically.

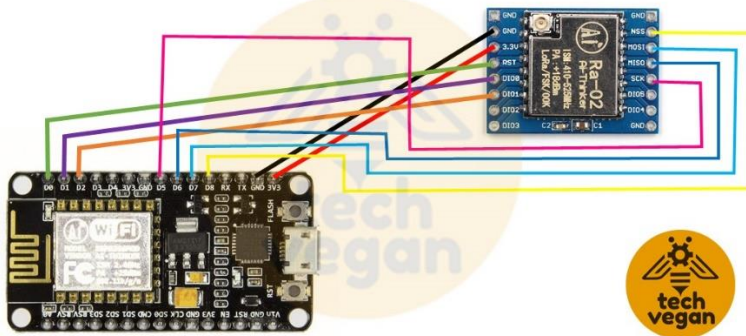
```

175
10
4
4
3
3
4
2
3
3
4
13
202
5
176
21
21
22
17
17
18
18

```

## Lab Experiment 9

### LoRa using NodeMCU(ESP8266)



#### CODE:

##### Sender

```
#include <LoRa.h>
#include <DHT.h>

#define DHTPIN 0      //pin where the dht11 is connected
DHT dht(DHTPIN, DHT11);

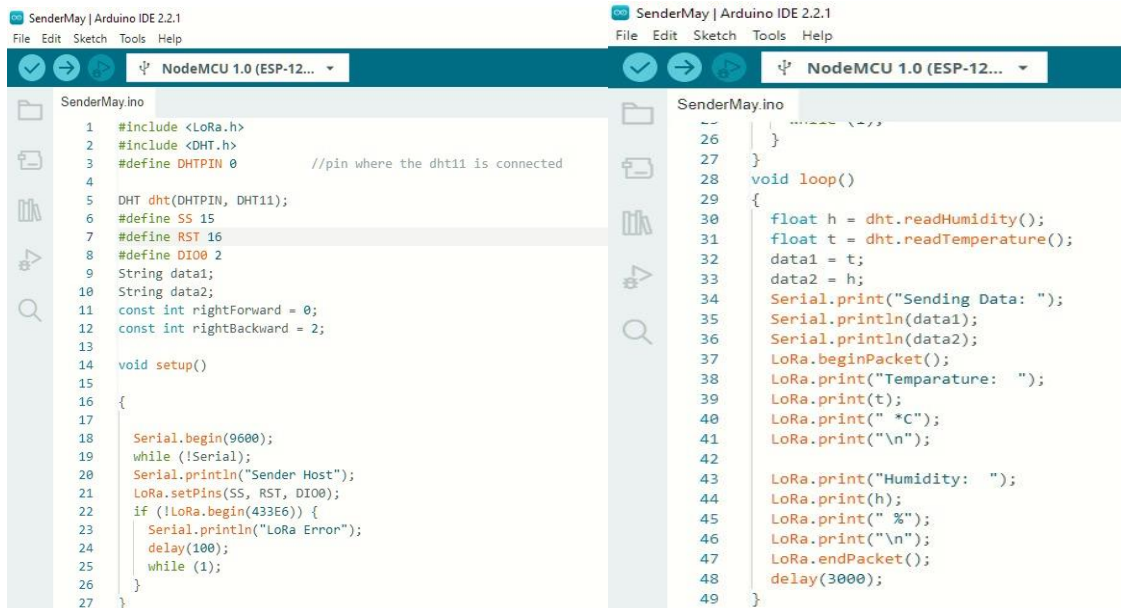
#define SS 15
#define RST 16
#define DIO0 2

String data1;
String data2;

const int rightForward = 0;
const int rightBackward = 2;

void setup(){
  Serial.begin(9600);
  while (!Serial);
  Serial.println("Sender Host");
  LoRa.setPins(SS, RST, DIO0);
```

```
if (!LoRa.begin(433E6)) {  
    Serial.println("LoRa Error");  
    delay(100);  
    while (1);  
}  
}  
void loop()  
{  
    float h = dht.readHumidity();  
    float t = dht.readTemperature();  
    data1 = t;  
    data2 = h;  
    Serial.print("Sending Data: ");  
    Serial.println(data1);  
    Serial.println(data2);  
    LoRa.beginPacket();  
    LoRa.print("Temperature: ");  
    LoRa.print(t);  
    LoRa.print(" *C");  
    LoRa.print("\n");  
    LoRa.print("Humidity: ");  
    LoRa.print(h);  
    LoRa.print(" %");  
    LoRa.print("\n");  
    LoRa.endPacket();  
    delay(3000);  
}
```



Receiver:

```
#include <ESP8266WiFi.h>
```

```
#include <LoRa.h>
```

```
String apiKey = "MXKN78RD40818DX2";// Enter your Write API key from  
ThingSpeak
```

```
const char *ssid = "vivo 1904"; // replace with your wifi ssid and wpa2 key
```

```
const char *pass = "1e544b2103a4";
```

```
const char* server = "api.thingspeak.com";
```

```
#define SS D8
```

```
#define RST D0
```

```
#define DIO0 D1
```

```
WiFiClient client;
```

```
void setup() {
```

```
    //digitalWrite(rightForward,HIGH);
```

```
    //digitalWrite(rightBackward,LOW);
```

```
    Serial.begin(9600);
```

```

while (!Serial);
Serial.println("Receiver Host");
LoRa.setPins(SS, RST, DIO0);
delay(1000);
if (!LoRa.begin(433E6)) {
    Serial.println("LoRa Error");
    delay(100);
    while (1);
}
Serial.println("Connecting to ");
Serial.println(ssid);
WiFi.begin(ssid, pass);
while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
}
Serial.println("");
Serial.println("WiFi connected");
}

void loop() {
    int packetSize = LoRa.parsePacket();
    if (packetSize) {
        Serial.println("Receiving Data: ");
        while (LoRa.available()) {
            String data = LoRa.readString();
            Serial.println(data);
            String temp = data.substring(18, 14);

```

```
Serial.println(temp);
client.connect(server,80); // "184.106.153.149" or api.thingspeak.com
String postStr = apiKey;
postStr += "&field1=";
postStr += String(temp);
postStr += "\r\n\r\n";
client.print("POST /update HTTP/1.1\n");
client.print("Host: api.thingspeak.com\n");
client.print("Connection: close\n");
client.print("X-THINGSPEAKAPIKEY: "+apiKey+"\n");
client.print("Content-Type: application/x-www-form-urlencoded\n");
client.print("Content-Length: ");
client.print(postStr.length());
client.print("\n\n");
client.print(postStr);
    client.stop();
}}
```

Receiver | Arduino 1.8.10
File Edit Sketch Tools Help

Receiver \$

```

#include <ESP8266WiFi.h>
#include <LoRa.h>

String apiKey = "M5G0N718D40811D0E2"; // Enter your Write API key from ThingSpeak

const char *ssid = "rivo 1904"; // replace with your wifi ssid and wpa2 key
const char *pass = "le54t6b10ia4";
const char *server = "api.thingspeak.com";

#define SS D8
#define RST D0
#define DIO0 D1

WiFiClient client;

void setup() {
  //digitalWrite(rightForward, HIGH);
  //digitalWrite(rightBackward, LOW);

  Serial.begin(9600);
  while (!Serial);
  Serial.println("Receiver Host");
  LoRa.setPins(SS, RST, DIO0);

  delay(1000);
  if (!LoRa.begin(433E6)) {
    Serial.println("LoRa Error");
    delay(100);
    while (1);
  }

  Serial.println("Connecting to ");
  Serial.println(ssid);
  WiFi.begin(ssid, pass);

  while (WiFi.status() != WL_CONNECTED)
  {
    delay(500);
  }

  int packetSize = LoRa.parsePacket();
  if (packetSize) {
    Serial.println("Receiving Data: ");
    while (LoRa.available()) {
      String data = LoRa.readString();
      Serial.println(data);

      String temp = data.substring(18, 14);
      Serial.println(temp);

      client.connect(server, 80); // "184.106.153.145" or api.thingspeak.com
      String postStr = apiKey;
      postStr += "&field1=";
      postStr += String(temp);
      postStr += "\n\n";

      client.print("POST /update HTTP/1.1\n");
      client.print("Host: api.thingspeak.com\n");
      client.print("Connection: close\n");
      client.print("X-THINGSPEAKAPIKEY: "+apiKey+"\n");
      client.print("Content-Type: application/x-www-form-urlencoded\n");
      client.print("Content-Length: ");
      client.print(postStr.length());
      client.print("\n\n");
      client.print(postStr);
    }
  }
}

void loop() {
}

```

Receiver | Arduino 1.8.10
File Edit Sketch Tools Help

Receiver \$

```

}

void loop() {

  int packetSize = LoRa.parsePacket();
  if (packetSize) {
    Serial.println("Receiving Data: ");
    while (LoRa.available()) {
      String data = LoRa.readString();
      Serial.println(data);

      String temp = data.substring(18, 14);
      Serial.println(temp);

      client.connect(server, 80); // "184.106.153.145" or api.thingspeak.com
      String postStr = apiKey;
      postStr += "&field1=";
      postStr += String(temp);
      postStr += "\n\n";

      client.print("POST /update HTTP/1.1\n");
      client.print("Host: api.thingspeak.com\n");
      client.print("Connection: close\n");
      client.print("X-THINGSPEAKAPIKEY: "+apiKey+"\n");
      client.print("Content-Type: application/x-www-form-urlencoded\n");
      client.print("Content-Length: ");
      client.print(postStr.length());
      client.print("\n\n");
      client.print(postStr);

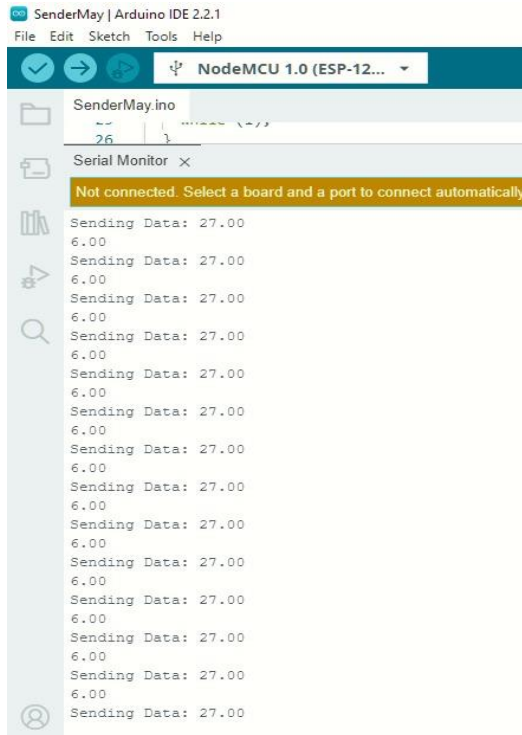
      client.stop();
    }
  }
}

```



## OUTPUT:

Sender:

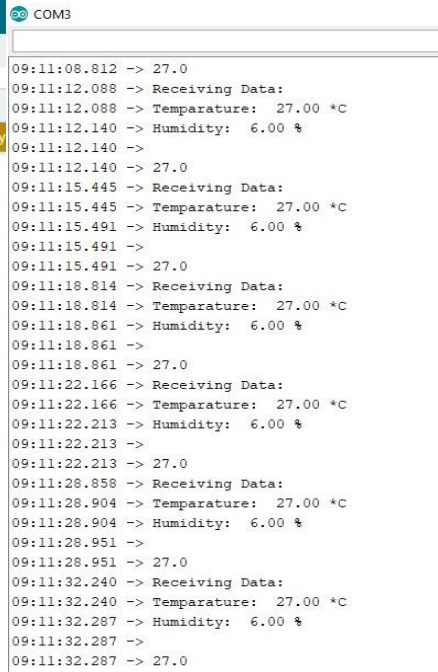


```

SenderMay.ino
26
Serial Monitor x
Not connected. Select a board and a port to connect automatically.
Sending Data: 27.00
6.00
Sending Data: 27.00
6.00
Sending Data: 27.00
6.00
Sending Data: 27.00
6.00
Sending Data: 27.00
6.00
Sending Data: 27.00
6.00
Sending Data: 27.00
6.00
Sending Data: 27.00
6.00
Sending Data: 27.00
6.00
Sending Data: 27.00
6.00
Sending Data: 27.00
6.00
Sending Data: 27.00
6.00

```

Receiver:



```

COM3
09:11:08.812 -> 27.0
09:11:12.088 -> Receiving Data:
09:11:12.088 -> Temperature: 27.00 *C
09:11:12.140 -> Humidity: 6.00 %
09:11:12.140 ->
09:11:12.140 -> 27.0
09:11:15.445 -> Receiving Data:
09:11:15.445 -> Temperature: 27.00 *C
09:11:15.491 -> Humidity: 6.00 %
09:11:15.491 ->
09:11:15.491 -> 27.0
09:11:18.814 -> Receiving Data:
09:11:18.814 -> Temperature: 27.00 *C
09:11:18.861 -> Humidity: 6.00 %
09:11:18.861 ->
09:11:18.861 -> 27.0
09:11:22.166 -> Receiving Data:
09:11:22.166 -> Temperature: 27.00 *C
09:11:22.213 -> Humidity: 6.00 %
09:11:22.213 ->
09:11:22.213 -> 27.0
09:11:28.858 -> Receiving Data:
09:11:28.904 -> Temperature: 27.00 *C
09:11:28.904 -> Humidity: 6.00 %
09:11:28.951 ->
09:11:28.951 -> 27.0
09:11:32.240 -> Receiving Data:
09:11:32.240 -> Temperature: 27.00 *C
09:11:32.287 -> Humidity: 6.00 %
09:11:32.287 ->
09:11:32.287 -> 27.0

```

## On the ThinkSpeak

