Term Project Report

B. Tech.

SEM. VI (EC)

Team Members:

Mandviya Janki Vipulbhai (Batch: B1)

Ujeniya Pushti Bharatbhai (Batch: B3)

Faculty Mentors:

Prof. Vasim A Vohra

Prof. Manish k Patel

Department of Electronics & Communication
Faculty of Technology
Dharmsinh Desai University
Nadiad

Dharmsinh Desai University

Faculty of Technology

College Road, Nadiad – 387001. (Gujarat)

Certificate

This is to certify that the practical / term work carried out in the subject of **Term Project** and recorded in this journal is the bona fide work of Miss **MANDVIYA JANKI VIPULBHAI** Roll No: **EC058** Identity No: **20ECUBS016** of B. Tech. Semester-**VI** in the branch of **Electronics & Communication** during the academic year 2022 - 2023.

Staff In-Charge

Date:

Head of the Department

Date:

Dharmsinh Desai University

Faculty of Technology

College Road, Nadiad – 387001. (Gujarat)

Certificate

This is to certify that the practical / term work carried out in the subject of **Term Project** and recorded in this journal is the bona fide work of Miss **UJENIYA PUSHTI BHARATBHAI** Roll No: **EC103** Identity No: **20ECUBS033** of B. Tech. Semester-**VI** in the branch of **Electronics & Communication** during the academic year 2022 - 2023.

Staff In-Charge

Date:

Head of the Department

Date:

Table of Contents

Sr No.	Title	Page No.
1)	Abstract	**1**
2)	block diagram & Working of the project.	**2**
3)	Introduction to NodeMCU (ESP8266) and creating Blynk dashboard	**4**
Section 1: Monitoring Unit		
4)	Air Temperature and humidity measurement	13
5)	Soil moisture measurement	20
6)	Rain detection	24
Section 2: Irrigation Unit		
7)	Water pump control	28
8)	Water level measurement using ultrasonic sensor	33
9)	Complete circuit diagram, programming code and working explanation	38
10)	Advantages and future aspects, limitations	43
11)	Conclusion	44
12)	References	45

IoT based agriculture monitoring and Irrigation

Abstract:

In earlier days farmers used to calculate the readiness of soil. They didn't consider the stickiness, level of water and particularly atmosphere condition which was difficult to a Farmer. We often come across the news of losses happening to farmers due to unexpected climate change. Every crop needs different soil moisture level, temperature, amount of water etc. The biggest problem is that farmers are unable to detect exact measure of these parameters, it results into improper environmental condition for crops to grow. In this proposed system we are going to solve some of these problems. This project lets the farmer know the current conditions of the farm like soil moisture level, air humidity, air temperature and raining conditions by just few clicks on the mobile. An app named “Blynk IoT” lets the farmer see the data in pictorial representation on the app dashboard, which makes it easy to understand. In addition to that the system also waters the farm according to weather condition and soil conditions. Hence the farmer does not need to continuously monitor the farm and water the soil by turning on the pump manually when needed. It reduces the human effort to many extents and the less man-power is required.

Chapter 1: block diagram & Working of the project

In this chapter we are going to see the block diagram and working of the project in brief. We will also discuss the components used in the project.

❖ Block Diagram:

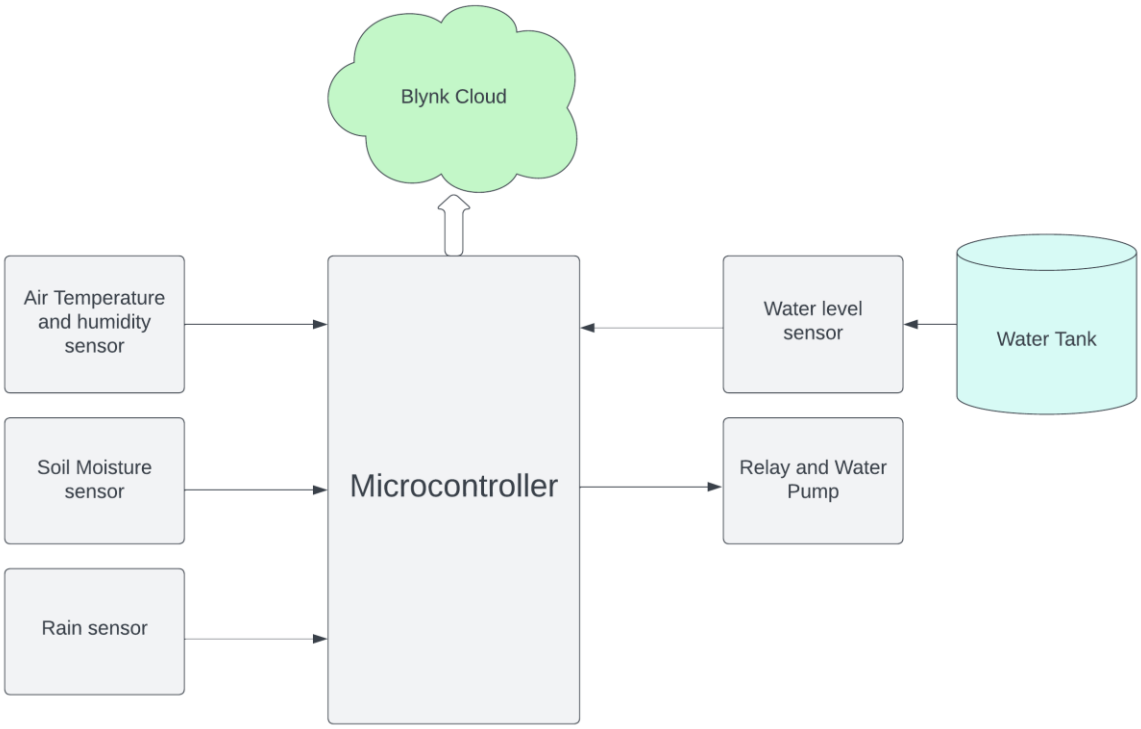


Fig 1.1 Block diagram

❖ Components:

Node MCU(ESP8266)

DHT11

Soil moisture sensor

Rain/Water sensor

Ultrasonic sensor(HC-SR04)

5V 10A relay

Water Pump

Breadboard

Connecting wires

5V battery

LEDs

Resistors

❖ Working of block diagram:

Microcontroller:

This is the heart of the system. All sensors and actuators are connected to the microcontroller. We are going to use ESP8266 as microcontroller which contains ESP 12E microcontroller in the module. ESP8266 will fetch data from various sensors and send it to Blynk cloud. According to the sensors data and water level of the tank it gives output to the relay which is connected to water pump.

Air temperature and humidity sensor:

DHT11 is used as the air temperature and humidity sensor. DHT11 measures the temperature and humidity of air, and sends the data to the microcontroller.

Soil moisture sensor:

This sensor measures the moisture level of soil in steps which is later on converted to percentage. The data is sent to the microcontroller.

Rain/Water sensor:

The rain sensor detects the presence of rain. When the water is detected on the surface of the sensor output HIGH (voltage level 1) is given to the microcontroller.

The Blynk Cloud:

The Blynk cloud is the IoT platform which shows the data in graphical format. The above parameters are shown on the Blynk dashboard. This cloud service also sends notification to the users mobile when the rainfall occurs and also notifies when the water pump is turned ON or OFF.

Water level sensor:

The ultrasonic sensor measures the level of water from the bottom of the tank. When the water level is below the given measure the pump automatically turns off. This unit is included because if the tank is empty, the motor may burn out due to running in dry state.

Relay and water pump:

Relay acts as an electromechanical swich which is connected to the water pump. When the microcontroller sends the high voltage to relay, it gets in contact with normally closed pin. When relay is activated, the water pump turns on and waters the farm.

Chapter 2: Introduction to NodeMCU (ESP8266) and creating Blynk dashboard

What is NodeMCU?

The NodeMCU (Node **m**icro**c**ontroller **U**nit) is an open-source software and hardware development environment built around an inexpensive System-on-a-Chip (soc) called the ESP8266. The ESP8266, designed and manufactured by Espressif Systems, contains the crucial elements of a computer: CPU, RAM, networking (Wi-Fi), and even a modern operating system and SDK. That makes it an excellent choice for Internet of Things (IoT) projects of all kinds

The advantage of NodeMCU over Arduino UNO

- It has better processor and memory of 4MB of flash memory
- The main advantage of NodeMCU over Arduino uno is that it contains Wi-Fi connection and can connect to the internet through Wi-Fi therefore it is best suited for IoT application
- NodeMCU is cheaper than Arduino uno also the NodeMCU is breadboard friendly and compact it can easily inserted into breadboard and test various circuit design but Arduino uno cannot be fit into the breadboard

The disadvantage of NodeMCU over Arduino UNO

- NodeMCU has only one analog input pins but the Arduino uno contains 6 analog pins
- As NodeMCU has lower voltage level so it cannot be compatible with the other modules in rare case

NodeMCU specifications

The NodeMCU is available in various package styles. Common to all the designs is the base ESP8266 core. The most common models of the NodeMCU are the Amica (based on the standard narrow pin-spacing) and the lolin which has the wider pin spacing and larger board.

The Amica NodeMCU is approximately 25% smaller in size than a closely compatible lolin style NodeMCU

Official Amica NodeMCU	Official Amica NodeMCU on Carrier Board	Lolin NodeMCU
		

The below table shows the NodeMCU technical specification

	Official NodeMCU	**NodeMCU Carrier Board**	**Lolin NodeMCU**
Microcontroller	ESP-8266 32-bit	ESP-8266 32-bit	ESP-8266 32-bit
NodeMCU Model	Amica	Amica	Clone lolin
NodeMCU Size	49mm x 26mm	49mm x 26mm	58mm x 32mm
Carrier Board Size	N/a	102mm x 51mm	N/a
Pin Spacing	**0.9" (22.86mm)**	**0.9" (22.86mm)**	1.1" (27.94mm)
Clock Speed	80 MHz	80 MHz	80 MHz
USB **to Serial**	CP2102	CP2102	CH340G
Operating Voltage	3.3V	3.3V	3.3V
Input Voltage	4.5V-10V	4.5V-10V	4.5V-10V
Flash Memory/SRAM	4 MB / 64 KB	4 MB / 64 KB	4 MB / 64 KB
Digital I/O Pins	11	11	11
Analog In Pins	1	1	1
ADC Range	0-3.3V	0-3.3V	0-3.3V
UART/SPI/I2C	1 / 1 / 1	1 / 1 / 1	1 / 1 / 1
Wi-Fi Built-In	802.11 b/g/n	802.11 b/g/n	802.11 b/g/n
Temperature Range	-40C - 125C	-40C - 125C	-40C - 125C
Product Link		NodeMCU	NodeMCU

NodeMCU Pinout and Functions Explained:

Power pins

There are four power pins. **VIN** pin and three **3.3V** pins.

- **VIN** can be used to directly supply the NodeMCU/ESP8266 and its peripherals. Power delivered on **VIN** is regulated through the onboard regulator on the NodeMCU module – you can also supply 5V regulated to the **VIN** pin
- **3.3V** pins are the output of the onboard voltage regulator and can be used to supply power to external components.

GND pin

Are the ground pins of NodeMCU/ESP8266

I2C pins

Are used to connect I2C sensors and peripherals. 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

NodeMCU/ESP8266 has 17 GPIO pins which can be assigned to 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 NodeMCU is embedded with a 10-bit precision SAR ADC. The two functions can be implemented using ADC. 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

NodeMCU/ESP8266 has 2 UART interfaces (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. However, UART1 (TXD1 pin) features only data transmit signal so, it is usually used for printing log.

SPI pins

NodeMCU/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

NodeMCU/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

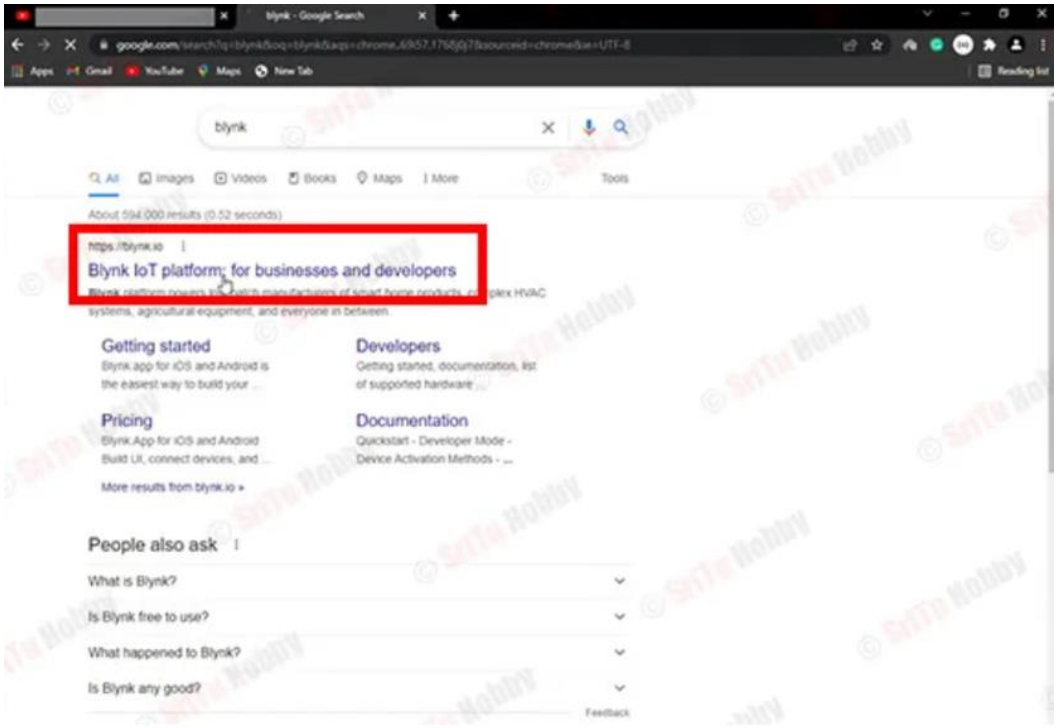
The board has 4 channels of Pulse Width Modulation (PWM). The PWM output can be implemented programmatically and used for driving digital motors and LEDs. PWM frequency range is adjustable from 1000 μs to 10000 μs (100 Hz and 1 KHz).

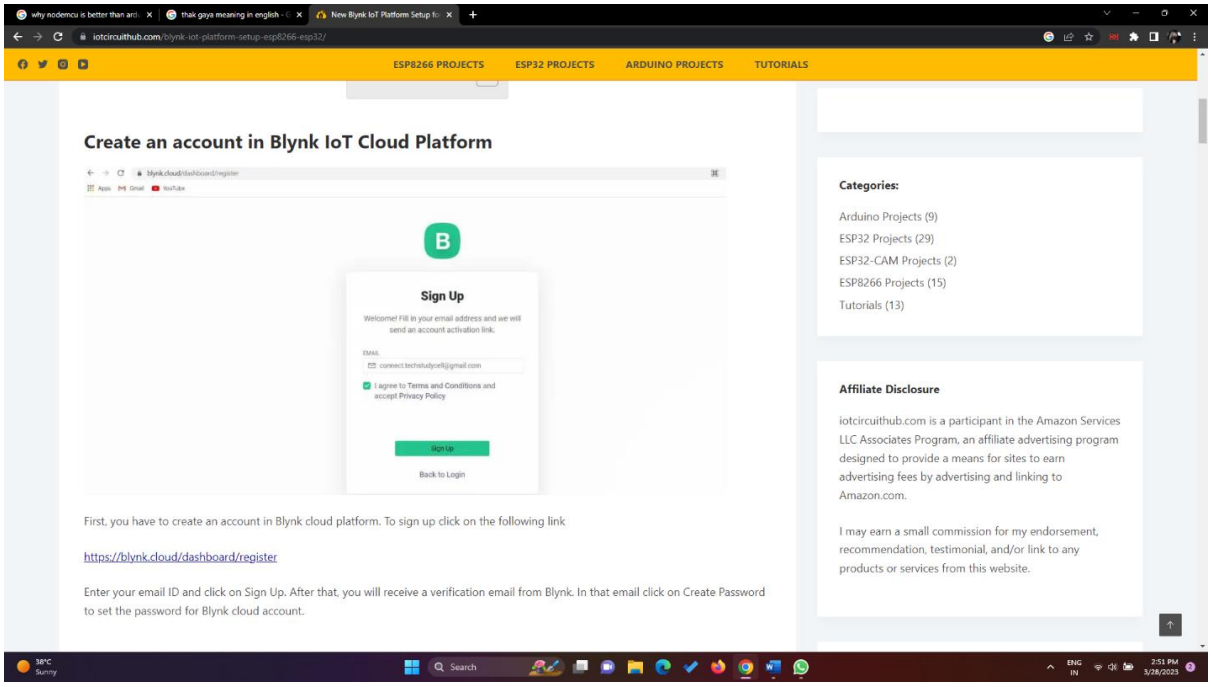
Control pins

Are used to control the NodeMCU/ESP8266. These pins include Chip Enable pin (EN), Reset pin (RST) and WAKE pin.

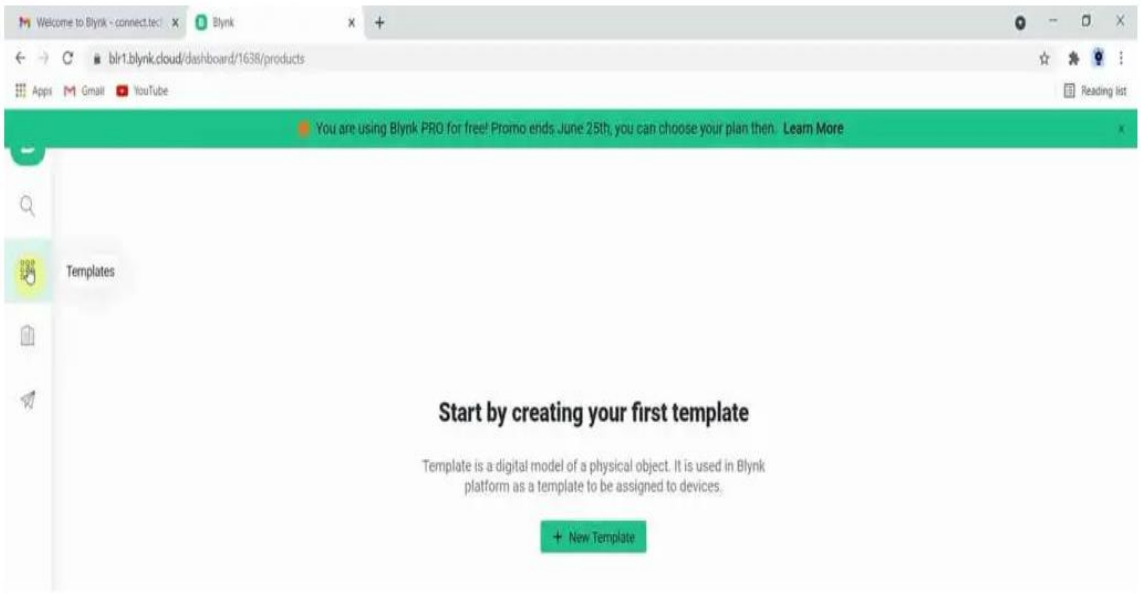
- **EN:** The ESP8266 chip is enabled when EN pin is pulled HIGH. When pulled LOW the chip works at minimum power.
- **RST:** RST pin is used to reset the ESP8266 chip.
- **WAKE:** Wake pin is used to wake the chip from deep-sleep.

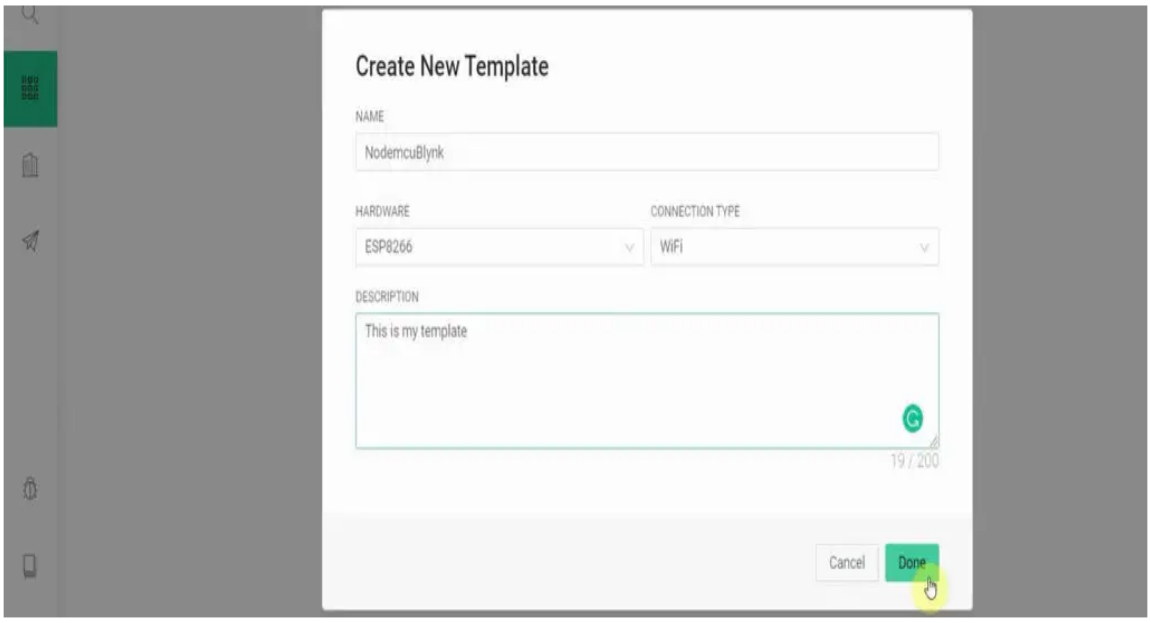
Steps for creating Blynk dashboard



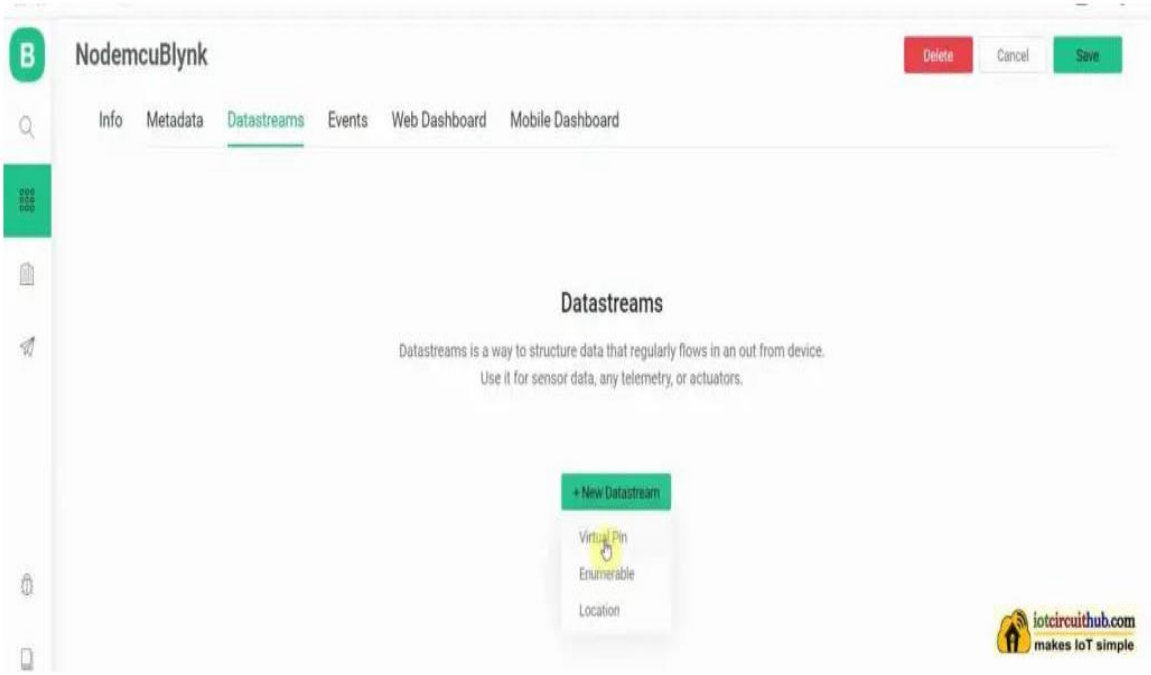


Creating new template

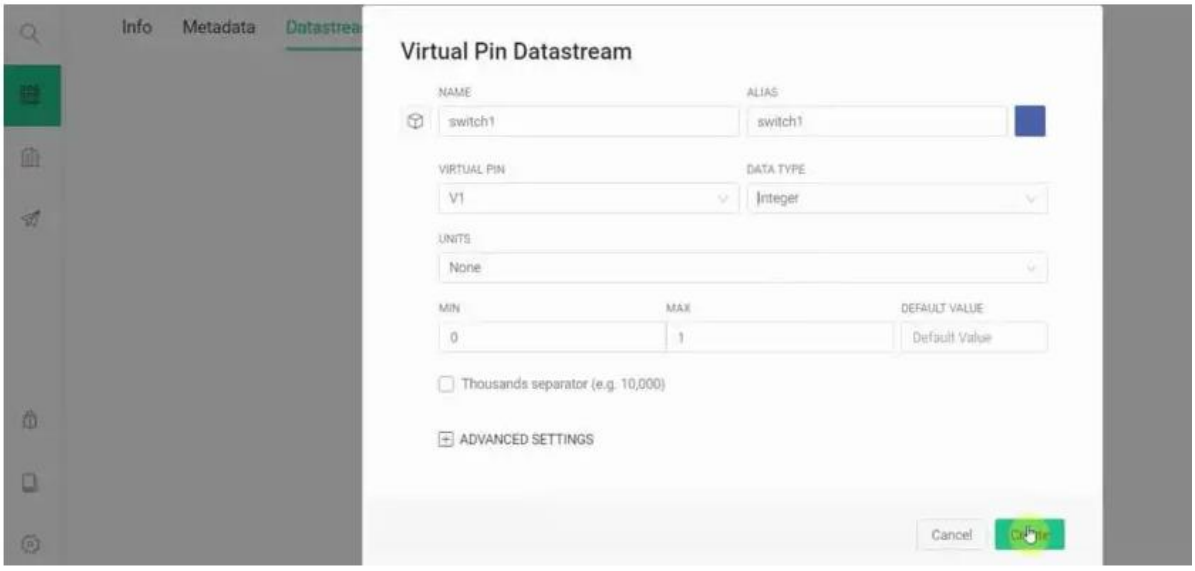


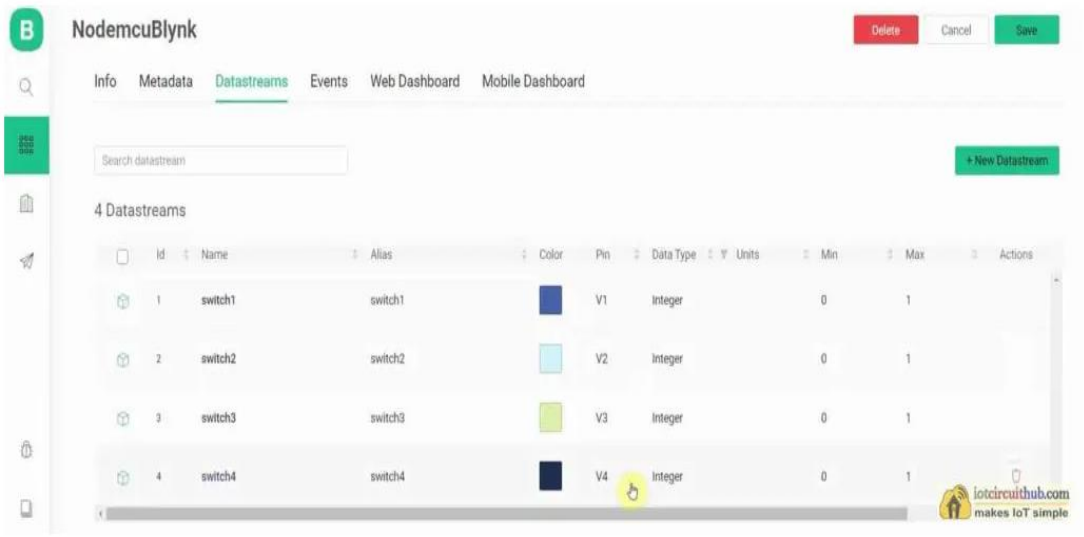


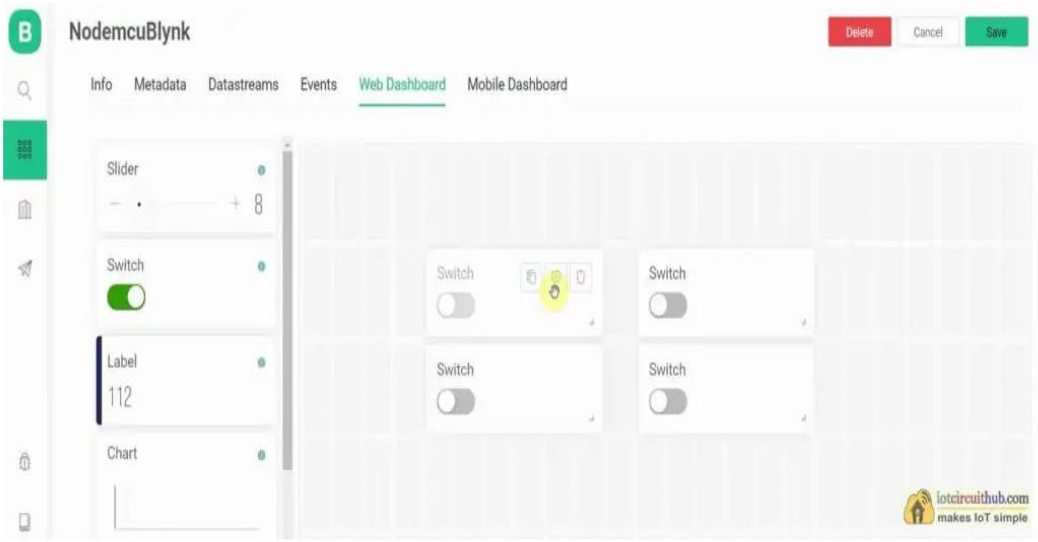
Create data streams

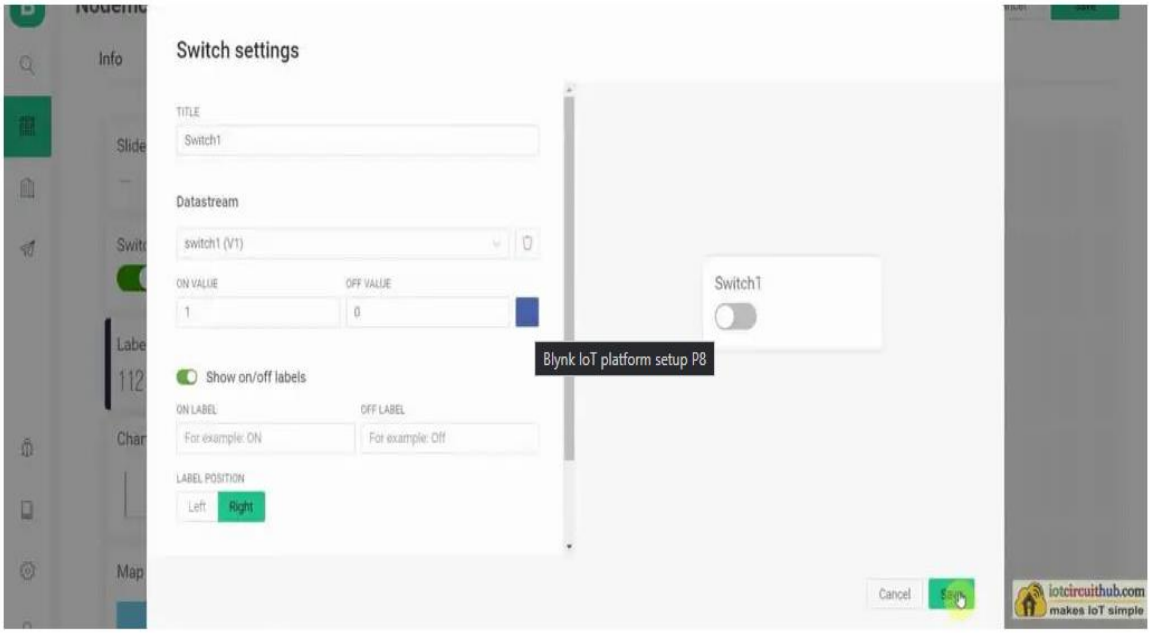


Select data stream and create virtual pin

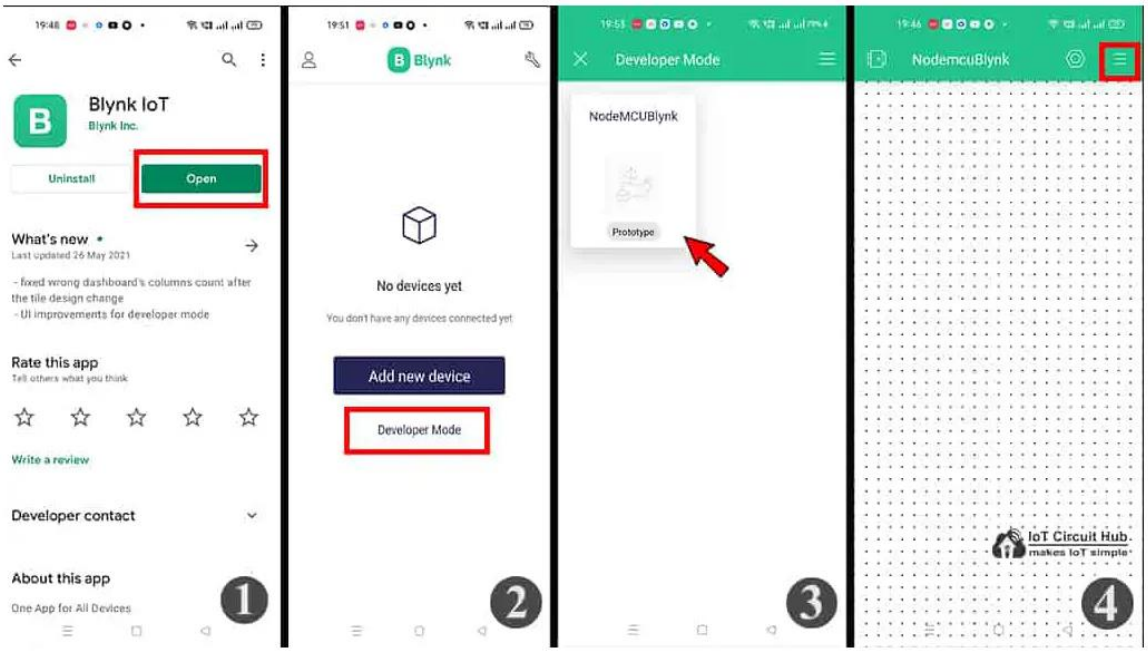


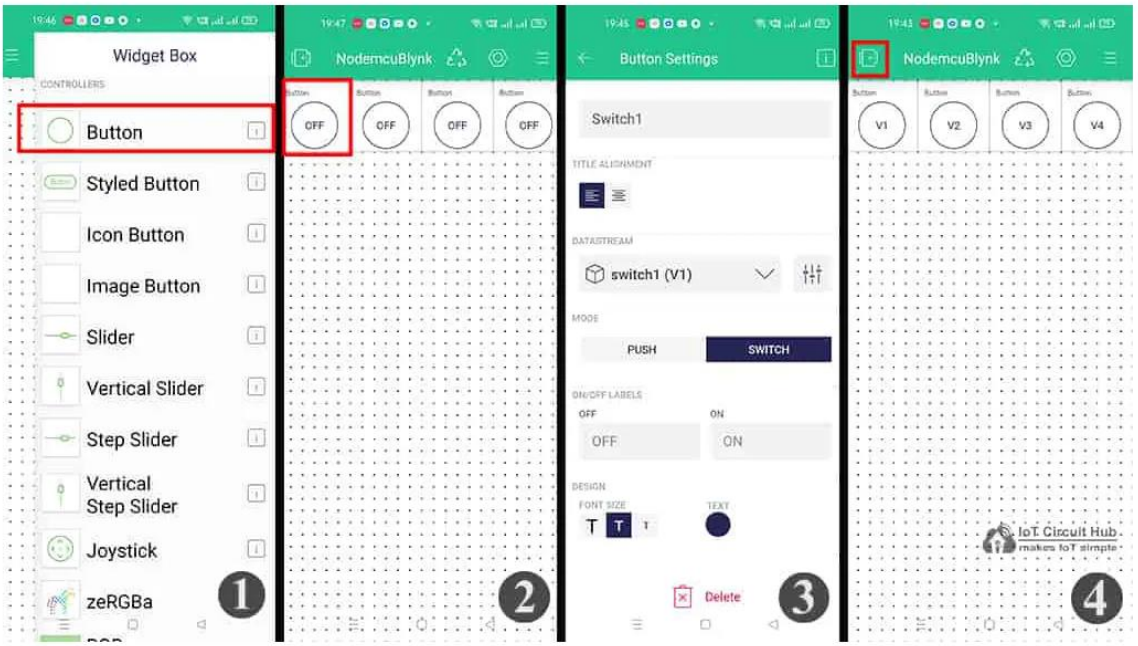

NodemcuBlynk
Delete
Cancel
Save
Info
Metadata
Datastreams
Events
Web Dashboard
Mobile Dashboard
Search datastream
+ New Datastream
4 Datastreams
Id
Name
Alias
Color
Pin
Data Type
Units
Min
Max
Actions
1 switch1 switch1 V1 Integer 0 1
2 switch2 switch2 V2 Integer 0 1
3 switch3 switch3 V3 Integer 0 1
4 switch4 switch4 V4 Integer 0 1
iotcircuithub.com
makes IoT simple


NodemcuBlynk
Delete
Cancel
Save
Info
Metadata
Datastreams
Events
Web Dashboard
Mobile Dashboard
Slider
8
Switch
Label
112
Chart
Switch
Switch
Switch
Switch
iotcircuithub.com
makes IoT simple


Info
Switch settings
TITLE
Switch1
Datastream
switch1 (V1)
ON VALUE
1
OFF VALUE
0
Show on/off labels
ON LABEL
For example: ON
OFF LABEL
For example: Off
LABEL POSITION
Left
Right
Switch1
Blynk IoT platform setup P8
Cancel
Save
iotcircuithub.com
makes IoT simple

Install Blynk IoT app to set up the mobile dashboard





SECTION 1: MONITORING UNIT

Chapter 3: Air temperature and humidity measurement

Introduction to DHT sensors:

Fig 3.1 DHT11-temperature and humidity sensor and its pinout

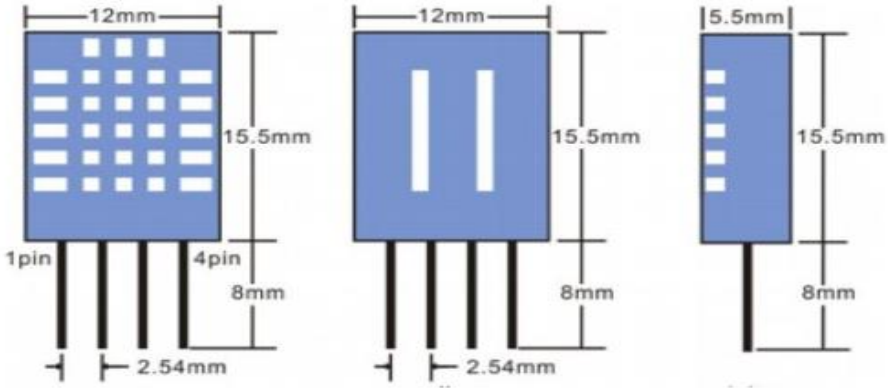


Fig 3.2 2D model and Dimensions of DHT11

There are Two versions of the DHT sensor, they look a bit similar and have the same pinout, but have different characteristics and specifications:

DHT11

- Ultra-low cost
- 3 to 5V power and I/O
- 2.5mA max current use during conversion (while requesting data)
- Good for 20-80% humidity readings with 5% accuracy
- Good for 0-50°C temperature readings ±2°C accuracy
- No more than 1 Hz sampling rate (once every second)

- Body size 15.5mm x 12mm x 5.5mm
- 4 pins with 0.1" spacing

DHT22

- Low cost
- 3 to 5V power and I/O
- 2.5mA max current use during conversion (while requesting data)
- Good for 0-100% humidity readings with 2-5% accuracy
- Good for -40 to 125°C temperature readings ±0.5°C accuracy
- No more than 0.5 Hz sampling rate (once every 2 seconds)
- Body size 15.1mm x 25mm x 7.7mm
- 4 pins with 0.1" spacing

The **DHT11** is a commonly used **Temperature and humidity sensor that** comes with a dedicated NTC to measure temperature and an 8-bit microcontroller to output the values of temperature and humidity as serial data. The sensor is also factory calibrated and hence easy to interface with other microcontrollers. The sensor can measure temperature from 0°C to 50°C and humidity from 20% to 90% with an accuracy of ±1°C and ±1%.

Difference between DHT11 Sensor and Module:

The **DHT11 sensor** can either be purchased as a sensor or as a module. Either way, the performance of the sensor is same. The sensor will come as a 4-pin package out of which only three pins will be used whereas the module will come with three pins as shown above.

The only difference between the sensor and module is that the module will have a filtering capacitor and pull-up resistor inbuilt, and for the sensor, you have to use them externally if required.

For DHT11 Sensor		
1	Vcc	Power supply 3.5V to 5.5V
2	Data	Outputs both Temperature and Humidity through serial Data
3	NC	No Connection and hence not used
4	Ground	Connected to the ground of the circuit

For DHT11 Sensor module		
1	Vcc	Power supply 3.5V to 5.5V
2	Data	Outputs both Temperature and Humidity through serial Data
3	Ground	Connected to the ground of the circuit

Communication with microcontroller:

- DHT11 uses only one wire for communication. The voltage levels with certain time value defines the logic one or logic zero on this pin.
- The communication process is divided in three steps, first is to send request to DHT11 sensor then sensor will send response pulse and then it starts sending data of total 40 bits to the microcontroller.

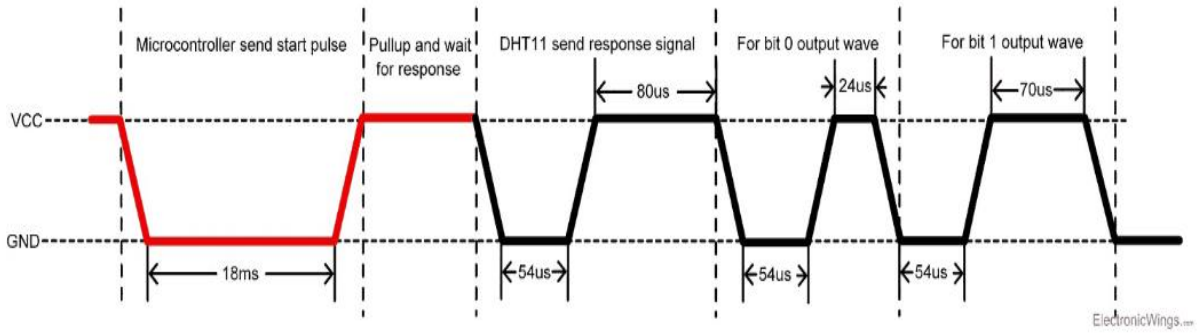


Fig 3.3 communication process of DHT11 with the microcontroller

Start pulse (request)

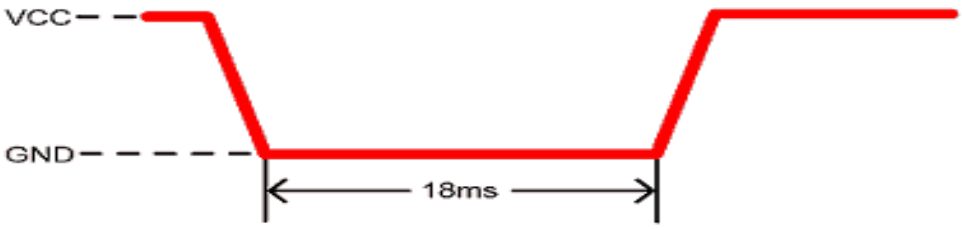


Fig 3.4 start pulse

- To start communication with DHT11, first we should send the start pulse to the DHT11 sensor.
- To provide start pulse, pull down (low) the data pin minimum 18ms and then pull up, as shown in diagram

Response

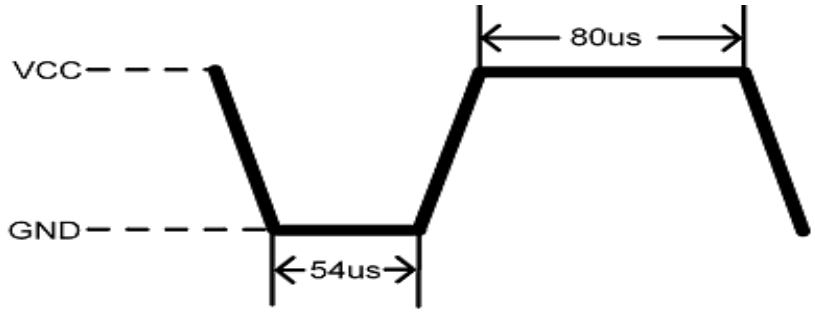


Fig 3.5 response of staring pulse

- After getting start pulse from, DHT11 sensor sends the response pulse which indicates that DHT11 received start pulse.
- The response pulse is low for 54us and then goes high for 80us.

Data

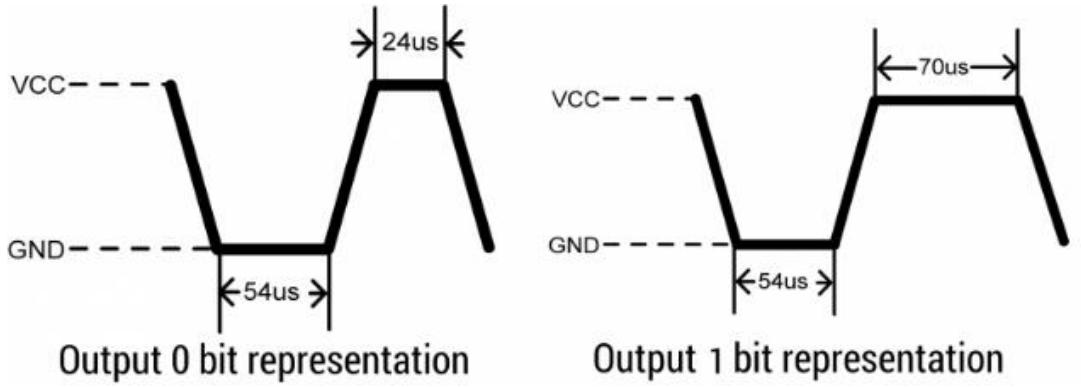


Fig 3.6 bit representation

After sending the response pulse, DHT11 sensor sends the data, which contains humidity and temperature value along with checksum.

- The data frame is of total 40 bits long, it contains 5 segments (byte) and each segment is 8-bit long.
- In these 5 segments, first two segments contain humidity value in decimal integer form. This value gives us Relative Percentage Humidity. 1st 8-bits are integer part and next 8 bits are fractional part.

- Next two segments contain temperature value in decimal integer form. This value gives us temperature in Celsius form.
- Last segment is the checksum which holds checksum of first four segments.
- Here checksum byte is direct addition of humidity and temperature value. And we can verify it, whether it is same as checksum value or not. If it is not equal, then there is some error in the received data.
- Once data received, DHT11 pin goes in low power consumption mode till next start pulse.

End of frame

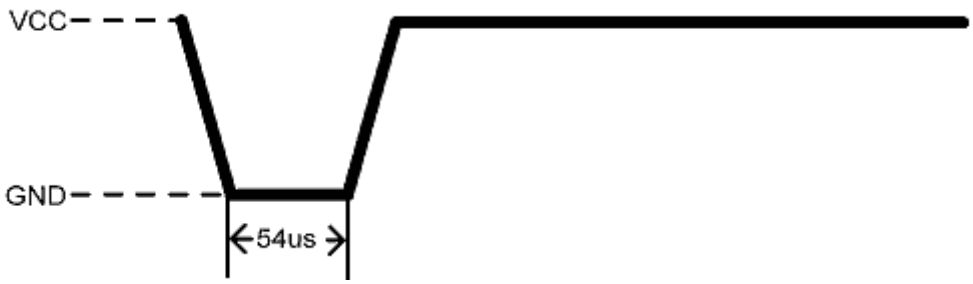


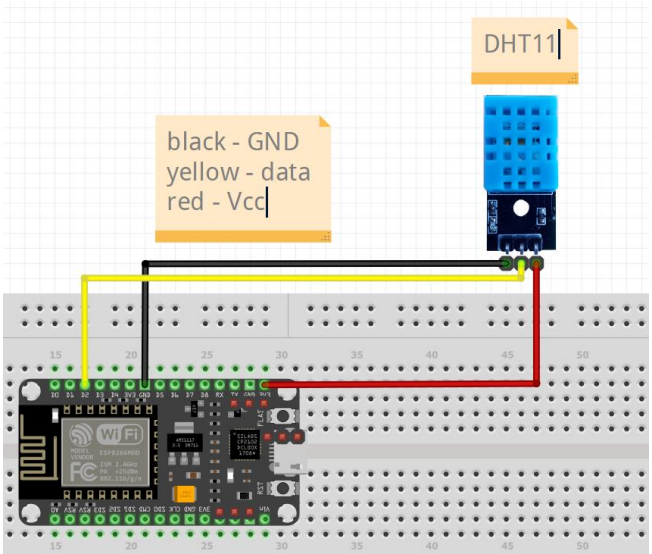
Fig 3.7 end of frame

- After sending 40-bit data, DHT11 sensor sends 54us low level and then goes high. After this DHT11 goes in sleep mode.

Applications of DHT sensors:

- Measure temperature and humidity
- Local Weather station
- Automatic climate control
- Environment monitoring

Connection diagram of DHT11 with NodeMCU:



Programming code for interfacing DHT11:

```
#define BLYNK_TEMPLATE_ID "TMPL0V3dtmm5" //copy from server
#define BLYNK_DEVICE_NAME "Agriculture and Irrigation system"
#define BLYNK_AUTH_TOKEN "qJSUzdH0WAAWlzzstmLNgmnbhTv3caSz"//copy from server

#define DHTTYPE DHT11 //configure dht type module
#define DHTPIN D2 // data pin of module to digital pin of node in d2

#include<ESP8266WiFi.h> // node library
#include<DHT.h> //dht library
#include <BlynkSimpleEsp8266.h> //Blynk library


char auth[]= BLYNK_AUTH_TOKEN; // declare one character
char ssid[]="Jalsa Karone Jentilal"; //declare  character variable called as ssid for wifi name
char pass[]="qwertyuiop";// declare character variable for Wi-Fi password


DHT dht(DHTPIN,DHTTYPE); //declare object of dht library
void  airdata() //declare void function called as airdata
{

  float t=dht.readTemperature(); // temerature data get and  store in t variable
  float h=dht.readHumidity(); //humidity data get and store in h variable
if(isnan(t)  ||  isnan(h))  //  if  data  is  not  coming  then  condition  gets  true otherwise  if data  is  obtained then go to else part
{

Serial.println("fail to get the data from DHT11");
return;

}

Blynk.virtualWrite(V1,t); //v1 for temperature and  value of t given to v1
Blynk.virtualWrite(V2,h);//v2 for humidity given to v2


Serial.print("humidity:");
Serial.println(h);
Serial.print("temperature(celcius):");
Serial.println(t);
}


void setup() {
  // put your setup code here, to run once:
```

```
Serial.begin(115200);
pinMode(DHTPIN, INPUT);
//pinMode(DHTPIN,OUTPUT);
dht.begin(); //to start the dht
delay(700);
Blynk.begin(auth,ssid,pass); //to  configure the blynk platform

}

void loop() {
  // put your main code here, to run repeatedly:
delay(3000);
airdata(); //calling the  function


}
```

Chapter 4: Soil moisture measurement

Introduction to soil moisture sensor:

Fig 4.1 soil moisture sensor

There are two types of sensors

- Volumetric water content sensor

Directly measure the amount of water in the soil.

Due to the technology used, volumetric sensors are the most expensive soil moisture sensors to purchase

They are incredibly accurate and provide instant data to growers. They are commonly used in research settings and in high-value crops where speed and accuracy justify the higher equipment cost.

- Soil tension sensor

Soil particles hold water through either tension or adhesion. Tensiometers are soil moisture sensors that measure this tension between soil particles and water molecules. In order for plants to access this water they must overcome the tension to draw water molecules away from the soil particles and into their roots.

Tensiometers are relatively simple and inexpensive equipment

Soil moisture is basically the content of water present in the soil. This can be measured using a soil moisture sensor which consists of two conducting probes that act as a probe. It can measure the moisture content in the soil based on the change in resistance between the two conducting plates.

The resistance between the two conducting plates varies in an inverse manner with the amount of moisture present in the soil.

Here, the analog output of the soil moisture sensor is processed using ADC. The moisture content in terms of percentage is displayed on the serial monitor.

The output of the soil moisture sensor changes in the range of ADC values from 0 to 1023.

NodeMCU ADC can be used to measure analog voltage from the soil moisture sensor

Pin configuration

Sensor probe	
+	Voltage supply
-	Negative pin
LM393 comparator	
VCC	Power supply
Gnd	Common GND
A0	Analog output
D0	Digital output

Working of soil moisture sensor:

This sensor uses capacitance to measure soil moisture content. And the functionality of this sensor can be performed by inserting this sensor into the soil

This soil moisture sensor is integrated with LM393 comparator chip which is an IC it consists of two integrated operational amplifiers. They are used in devise that measure analog signal and act as ADC also each comparator accepts two inputs for comparison. A comparator compares these two input voltage measures which input voltage is greater and provides an output

this module has built in potentiometer to set the sensitivity of the digital output. And we can also set the threshold with potentiometer. So, when the moisture level exceeds the threshold, the module will output low. Otherwise, the output is high.

There are two LED indicator in the module the one is red led for power indicator and the another one is for green for output indicator

Specification of Soil Moisture Sensor:

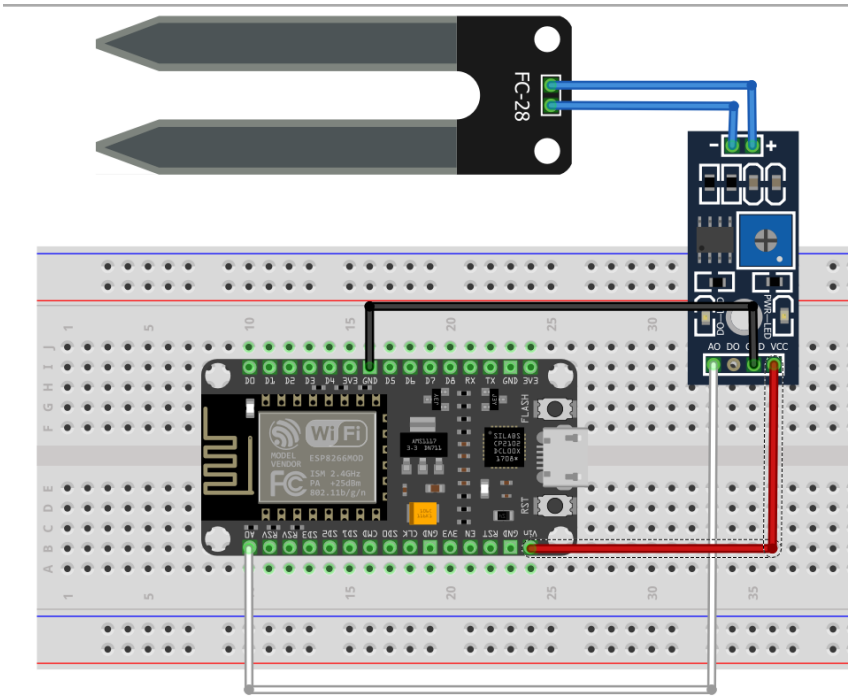
- **Operating Voltage:** The voltage range is 3.3V to 5V DC.
- **Operating Current:** 15mA
- **Output Type:** The type of output that the sensor generates, such as analog voltage or digital signals.
- **Connectivity:** The type of connector or interface that the sensor uses to connect to the controller or computer.
- **IC Used:** LM393 IC is used as a comparator.
- **Indicator:** Two LED Indicators are used one for **Power** and one for **Output**

- **Cable length:** 20 cm
- **Weight:** 50 gms
- **Soil probe dimension:**6 cm x 3 cm

Application:

- Agriculture
- Landscape irrigation
- Research
- Simple sensors for gardeners

Circuit diagram of soil moisture sensor with NodeMCU:



Programming code for interfacing soil moisture sensor with NodeMCU:

```
//soil moisture  sensor
#include <ESP8266WiFi.h>
#include <BlynkSimpleEsp8266.h>


#define BLYNK_TEMPLATE_ID "TMPL0V3dtmm5"
#define BLYNK_DEVICE_NAME "Agriculture and Irrigation system"
#define BLYNK_AUTH_TOKEN "qJSUzdH0WAAWlzzstmLNgmnbhTv3caSz"//copy from server
```

```
#define SOILPIN A0  //soil moisture pin in analog pin A0 of nodemcu

char ssid[] = "Jalsa Karone Jentilal";
char pass[] = "qwertyuiop";
char auth[] = BLYNK_AUTH_TOKEN;

float moisturepercentage;  //variable to calculate soil moisture level in percentage

int presentstate = 0;  //checks the present data of  soil moisture

void  soilmoisturefunc() //declare void function called as soilmoisturefunc
{

presentstate = analogRead(SOILPIN);
moisturepercentage = (100.0 - ((presentstate / 1024.0) * 100.0)) ; //calculate percentage of
moisture
Blynk.virtualWrite(V4, moisturepercentage);
Serial.print ("moisture of soil(percentage):");
Serial.println (moisturepercentage,2);  //2  shows the decimal  point upto 2

}


void setup() {

  pinMode(SOILPIN,OUTPUT);
  Serial.begin(9600);
  Blynk.begin(auth, ssid, pass);
}

void loop() {

  soilmoisturefunc(); //calling the soilmoisture function
  delay(5000);
  Blynk.run(); //for keeping connection with blynk like sending and receiving data

  }
```

Chapter 5: Rain detection

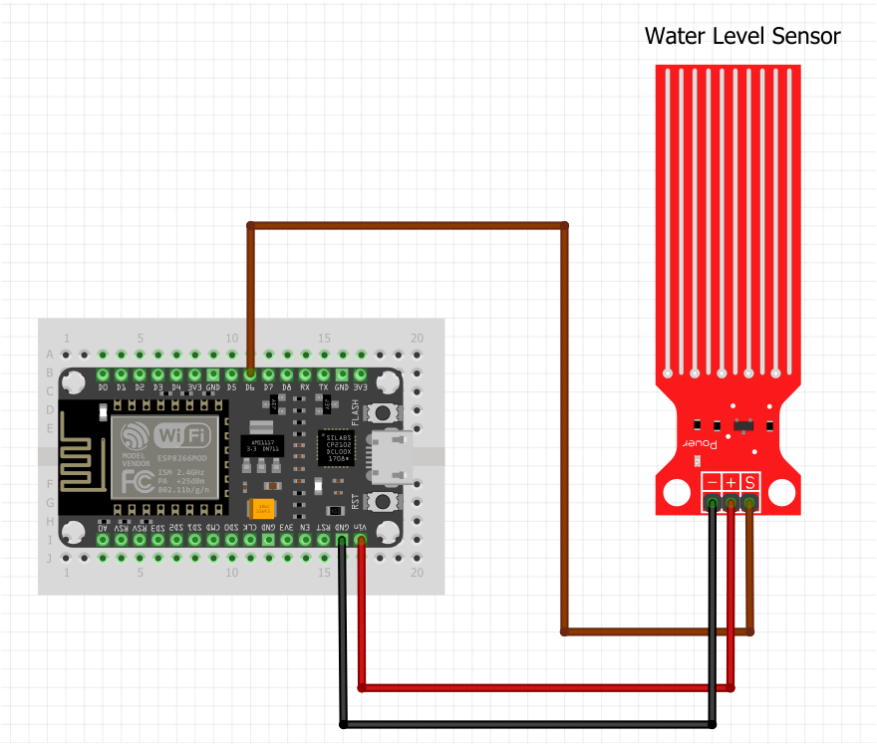
Introduction to water sensor:

Fig 5.1 water sensor

A rain sensor is one kind of switching device which is used to detect the rainfall. It works like a switch and the working principle of this sensor is, whenever there is rain, the switch will be normally closed.

This water level sensor module has a series of parallel exposed traces to measure droplets/water volume in order to determine the water level. Very Easy to monitor water level as the output to analog signal is directly proportional to the water level. This output analog values can be directly read via ADC and can also be connected directly to ESP8266 analog input pins.

Circuit diagram of water sensor with NodeMCU:



Programming code for interfacing water sensor with NodeMCU:

```
//rain sensor

#define BLYNK_TEMPLATE_ID "TMPL0V3dtmm5"
#define BLYNK_DEVICE_NAME "Agriculture and Irrigation system"
#define BLYNK_AUTH_TOKEN "qJSUzdH0WAAWlzzstmLNgmnbhTv3caSz"

#include<ESP8266WiFi.h> //nodemcu library
#include<BlynkSimpleEsp8266.h> //blynk library

char auth[]= BLYNK_AUTH_TOKEN; // declare one character
char ssid[]="Jalsa Karone Jentilal"; //declare  character variable called as ssid for wifi name
char pass[]="qwertyuiop";// declare character variable for wifi password

#define RAINPIN D6 // output pin of rain sensor connected to digital pin D6
int rainval = 0; // define a variable rainval initial value of 0


void raindata() {

 //rain part

 rainval = digitalRead (RAINPIN); //read the digital value to the variable rainval

 if (rainval == 1) //if digital value of rainval is 1, the notification is sent

 {

Serial.println(" it is raining");
Blynk.logEvent("rain_alert","raining outside"); //event name and notification message as
raining outside
 }

else //if digital value of rainval is 0, execute this block

 {

  Serial.println(" it is not raining");

 }



  Blynk.virtualWrite(V3,rainval);// virtual pin of dashborad as V3
  Serial.println (rainval); // print the rainval value on serial monitor


}  //ending of the rain function
```

```
void setup ()
{

Serial.begin (9600); // set the baud rate to 9600
Blynk.begin(auth,ssid,pass);

}


void loop ()
{
raindata(); //calling the rain function
delay (5000); //delay  between  2 samples
Blynk.run(); // for keeping connection with blynk like sending data and receiving  data
}
```

SECTION 2: IRRIGATION UNIT

Chapter 6: Water pump control

Introduction to relay and water pump:

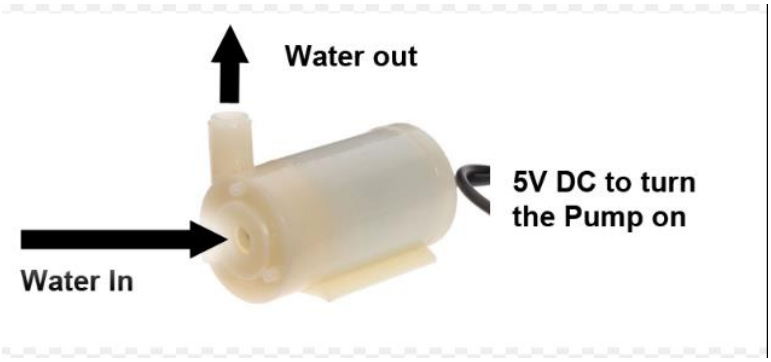


Fig 6.1 water pump

This submersible mini water pump is connected with 5 DC volt relay. When the rain occurs the water pump doesn't start, otherwise it gets turned on and we have set one particular moisture level of soil if moisture is below that level then it gets turn on the water pump. Basically, three conditions are checked to decide whether to turn on the pump or not.

1) Soil moisture level
2) Rain status
3) Water level in tank

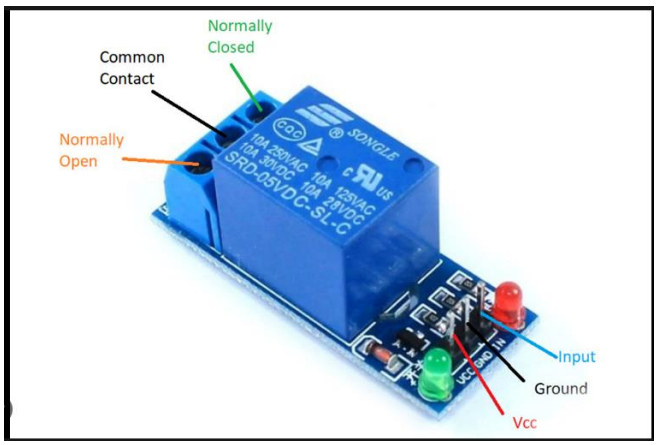


Fig 6.2 5V single channel relay module

So, here relay acts as a switch that controller can turn on to provide power to the pump.

Relay is one kind of electromechanical component that functions as a switch.

This module is mainly designed to interface through different microcontrollers like PIC, Arduino, etc.

It is single channel module which can handle one load and dual channel module can handle 2 load independently.

There are 4 pins which is used for single circuit whereas the 5 pin relay is used to switch the power between 2 circuits

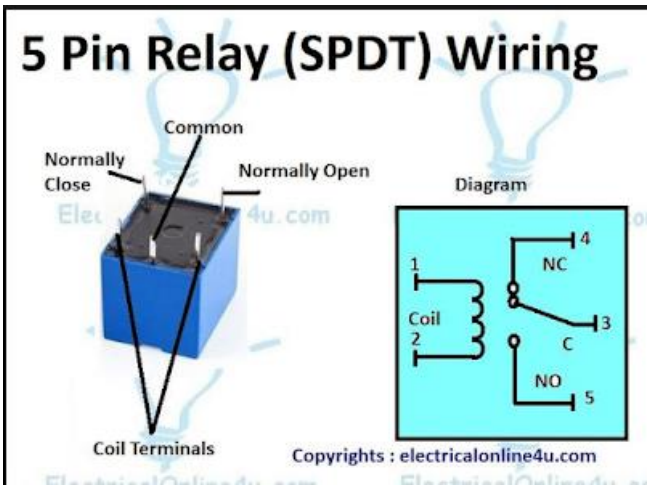


Fig 6.3 relay connection diagram and pinout

A single channel 5V relay module generally includes a coil, and two contacts like normally open (NO) and normally closed (NC). The relay module consists of screws that are used to connect wires & cables Thick main cables are mainly used whenever high voltage & current load is used. This 5V relay is coated with blue colour plastic material. For both AC & DC loads, it operates with 5-volt DC.

Pin detail of 5-volt relay:

Relay module consists of six pins such as normally open pin, normally closed, common, signal, Vcc and ground pins.

Signal pin

It is used to control the relay. This pin can be active low or active high. In case of active low, the relay will activate when we apply an active low signal to the signal pin. On the contrary, in the case of an active high, the relay will activate when we apply an active high signal to the signal pin. But usually, these modules work on an active high signal. This signal will energize the relay coil to make contact with the common terminal with the normally open terminal.

VCC pin

As its name suggests, it is a 5V relay. That means it requires 5V DC to operate. Hence, connect the 5v DC power supply to this pin.

Ground pin

Connect it with the ground terminal of 5V power supply. Furthermore, if you are driving a relay module with a microcontroller, also connect this pin with the ground terminal of the microcontroller.

Common pin

This terminal is connected with the load that we want to switch with the relay module.

Normally close pin

As the name of the normally close terminal suggests, it is normally connected with the COM pin and forms a closed circuit. But this normally closed connection breaks when the relay is activated by applying an active high.

Normally open pin
This pin is normally open unless we apply an activation signal to the signal pin of the 5V single channel relay module. In this case, the COM pin breaks its connection with the NC pin and makes a connection with the NO pin.

Status LED
Status LED is SMD LED which is connected through current limiting resistor and it is available on top right corner of the module. It shows the status of the relay. In other words, the status LED turns on when the relay is active and the coil is energized through a signal input pin. The DC current passes through a relay coil.

Power LED
Power LED is also a SMD type and it shows the status of power source connected with the 5V single channel relay module. Do not connect more than 5V source to Vcc and GND pins of the module. Otherwise, higher voltage may damage the status and power LEDs.

Working of 5 volt relay:

The relay uses the current supply for opening or closing switch contacts. Usually, this can be done through a coil to magnetize the switch contacts & drags them jointly once activated. A spring drives them separately once the coil is not strengthened.

Specifications of relay:

- Normal Voltage is 5V DC
- Normal Current is 70mA
- AC load current Max is 10A at 250VAC or 125V AC
- DC load current Max is 10A at 30V DC or 28V DC
- It includes 5-pins & designed with plastic material
- Operating time is 10msec
- Release time is 5msec
- Maximum switching is 300 operating per minute

Advantages:

The **advantages of the relay module** include the following.

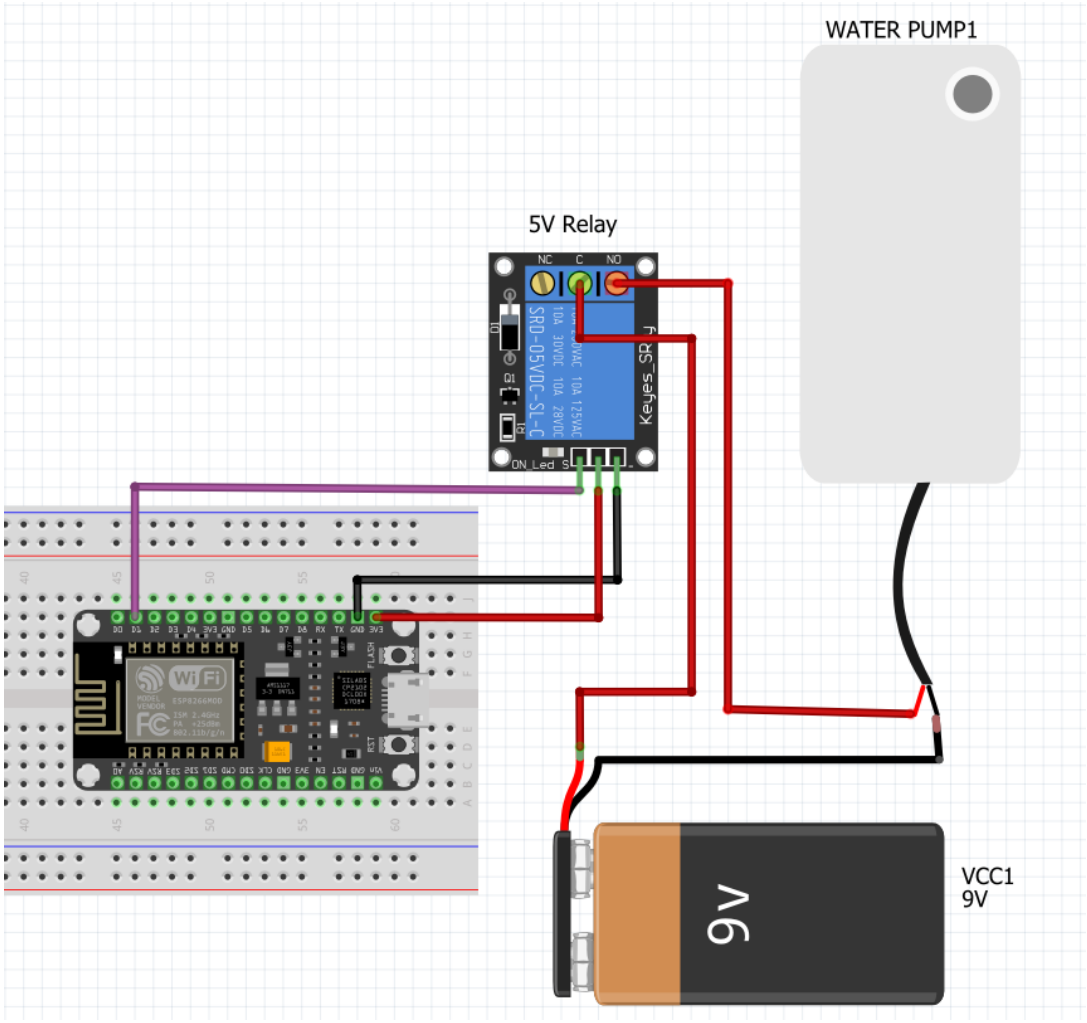
- A remote device can be controlled easily
- It is triggered with less current but it can also trigger high power machines
- Easily contacts can be changed
- At a time, several contacts can be controlled using a single signal
- Activating part can be isolated
- It can switch AC or DC
- At high temperatures, it works very well

Disadvantages:

The **disadvantages of the relay module** include the following.

- When contacts of relay modules are used overtime then they may damage
- Noise can be generated through the opening & closing of the contacts.
- Time taken for switching is High.

Circuit diagram of relay and water pump with NodeMCU:



Programming code for interfacing relay and pump with NodeMCU:

```
#include<ESP8266WiFi.h>
#define RELAYPIN D1

void setup() {
Serial.begin(9600);
```

```
}

void loop()
 {
//turn the pump on
pinMode(D1,OUTPUT);
digitalWrite(D1,LOW);
Serial.println("relay is on");
delay(1000);

//turn the pump off
pinMode(D1,INPUT);
digitalWrite(D1,HIGH);
Serial.println("relay is OFF");
delay(1000);
}
```

Chapter 7: Water level measurement using ultrasonic sensor

Introduction to ultrasonic sensor:

Fig 7.1 Ultrasonic sensor (HC-SR04)

Ultrasonic sensors are electronic devices that calculate the target's distance by emission of ultrasonic sound waves and convert those waves into electrical signals. The speed of emitted ultrasonic waves traveling speed is faster than the audible sound.

There are mainly two essential elements which are the transmitter and receiver. Using the piezoelectric crystals, the transmitter generates sound, and from there it travels to the target and gets back to the receiver component.

To know the distance between the target and the sensor, the sensor calculates the amount of time required for sound emission to travel from transmitter to receiver.

Ultrasonic Sensor Specifications:

- The sensing range lies between 40 cm to 300 cm.
- The response time is between 50 milliseconds to 200 milliseconds.
- The Beam angle is around $5^{0.}$
- It operates within the voltage range of 20 VDC to 30 VDC
- Preciseness is ±5%
- The frequency of the ultrasound wave is 120 kHz
- Resolution is 1mm
- The voltage of sensor output is between 0 VDC – 10 VDC
- The ultrasonic sensor weight nearly 150 grams
- Ambient temperature is -25^0C to $+70^0$C
- The target dimensions to measure maximum distance is 5 cm × 5 cm

Pin discreption of HC-SR04:

Vcc – This pin has to be connected to a power supply +5V.

TRIG – This pin is used to receive controlling signals from the Arduino board. This is the triggering input pin of the sensor

ECHO – This pin is used for sending signals to the Arduino board where the Arduino calculates the pulse duration to know the distance. This pin is the ECHO output of the sensor.

GND – This pin has to be connected to the ground.

Advantages:

- These devices are not impacted by the target's colour.
- The device shows flexibility in its distance measurement range where it holds the capability of measuring in the range of a few centimetres to five meters.
- It provides consistent outcomes and shows high reliability.
- High precision device.
- The measurements can be made every second thus showing rapid refresh rates.

Disadvantages:

Even though ultrasonic sensors employ versatile technology, there are a few limitations to be considered and those are:

- As sound speed is based on humidity and temperature, environmental circumstances might show an impact on the accuracy while measuring the distance.
- For minimal and embedded projects, ultrasonic sensors seem to be a not good option because these devices are large to integrate with small projects.
- These sensors will not function in a vacuum.
- The sensors will get dirt, wet and frozen which results in errors while measuring or the functionality gets impacted.

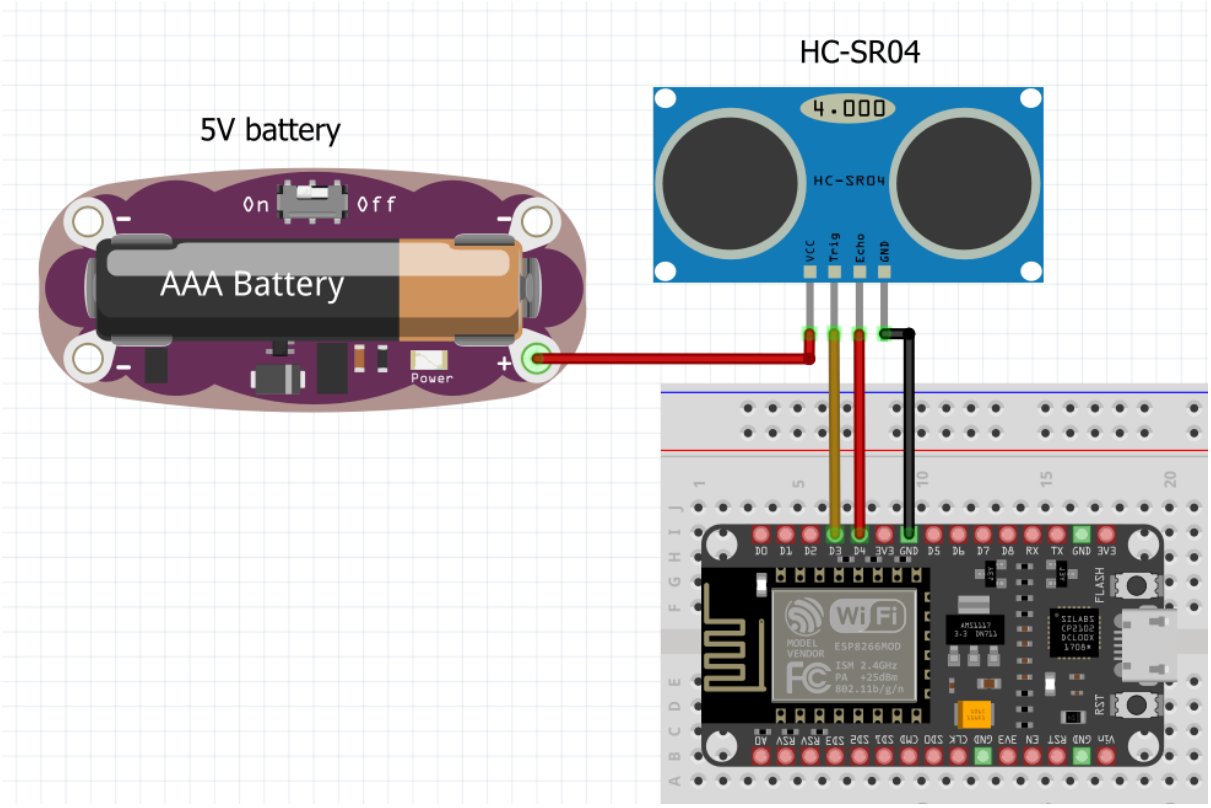
Applications:

The applications of ultrasonic sensors are:

1. Used in robotic sensing for positioning of robotic arms.
2. Employed in washdown design for constantly noticing the filling level of objects on a conveyor belt.
3. Used to detect objects.
4. The diameter of the coil/roll can be known by ultrasonic sensors.
5. Used to avoid a collision.
6. Proximity detection.

What is the range of the ultrasonic sensors?
The operating frequency range of ultrasonic sensors is between 30 kHz – 500 kHz.

Circuit diagram of Ultrasonic sensor with NodeMCU:



Programming code for interfacing HC-SR04 with NodeMCU:

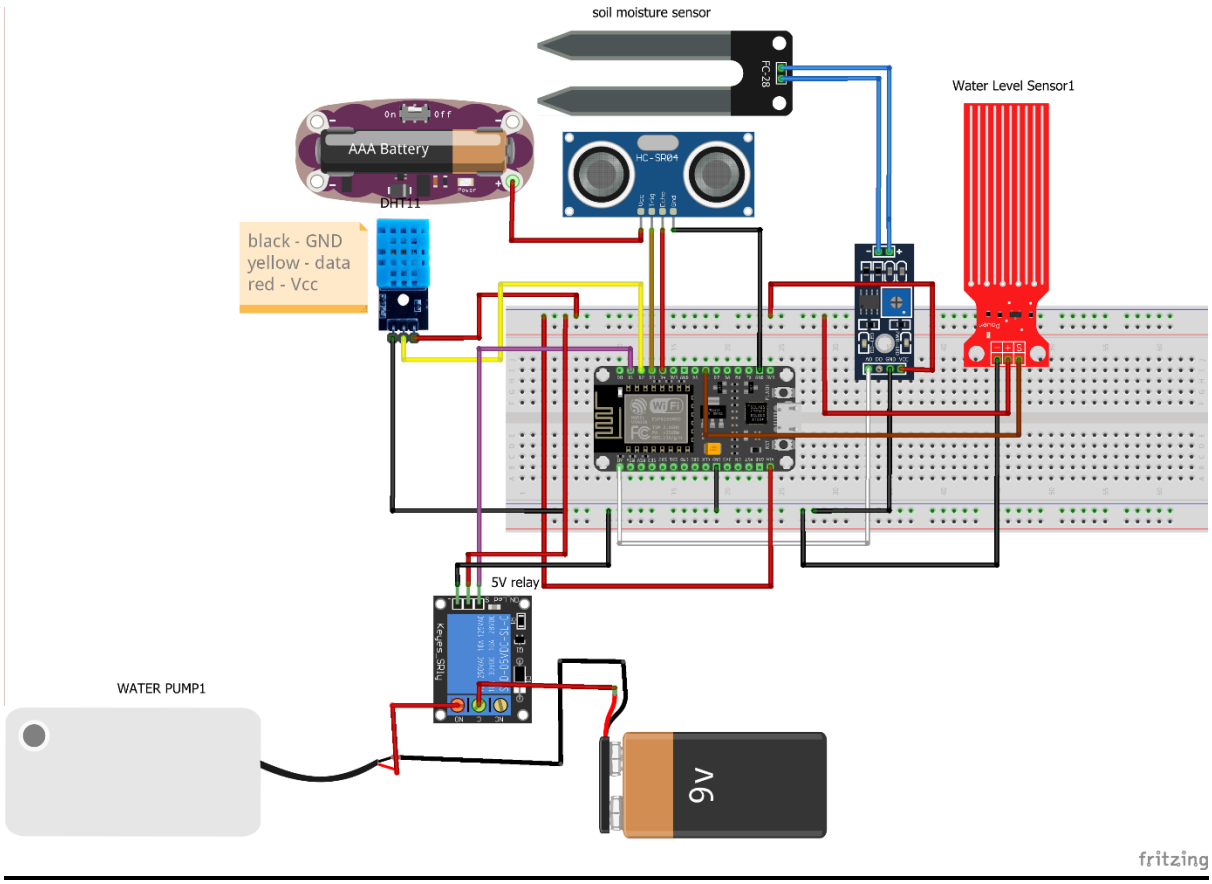
```
#include <ESP8266WiFi.h>  // node library
#define TRIGPIN D3  //digital pin  3 of node connected to trig pin of ultrasonic
#define ECHOPIN D4   // digital pin 4 of node connected to  echo pin of ultrasonic
#define PUMPPIN  D1

int empty_tank=10; //15-10= 5 cm for empty tank (level of water from ground = 5cm)
int full_tank=3;  //15-3= 12cm for full tank (level of water from ground = 12cm)

void setup()
{
pinMode(TRIGPIN,OUTPUT);
pinMode(ECHOPIN,INPUT);
```

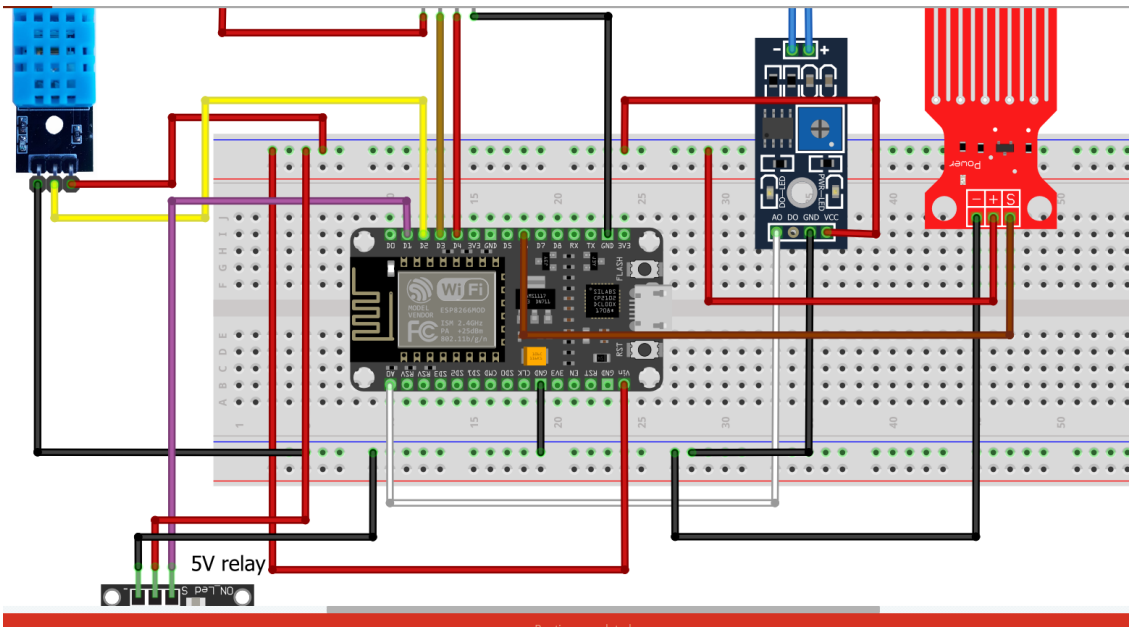
```
Serial.begin(9600);
}

void loop(){
float distance, duration;
/* in ultrasonic sensor we have to set the trig pin high for 10 micro seconds
   so first we will put trig pin low for 2 micro seconds and then put it high for 10 micro seconds*/

digitalWrite(TRIGPIN,LOW);
delayMicroseconds(2);
digitalWrite(TRIGPIN,HIGH);
delayMicroseconds(10);
digitalWrite(TRIGPIN,LOW);
delayMicroseconds(2);

duration=pulseIn(ECHOPIN,HIGH);  //it means the pulse function will get total duration time of echo pin
// speed of sound on room temp=343.5 m/s
//speed of temp at room temp=0.03435 cm/microsec
// total distance=total duration*0.03435  this is the total distance of sound wave from going to object and then return back to sensor
// so to find only distance of object from sensor we have to  divide the duration by 2
// distance from sensor to object=(total duration/2)*0.03435
//distance=duration*340/20000;

distance=(duration/2)*0.03435;  //now this is only distance from object to sensor
Serial.println(distance);

if(distance>=empty_tank )
{
 digitalWrite(PUMPPIN, HIGH); //PUMP  GETS TURN OFF
```

```
 Serial.println("PUMP IS TURNED OFF ");
}

else
{
digitalWrite(PUMPPIN,LOW);  //pump gets turn on
Serial.println("PUMP  IS TURNED ON ");
}

}
```

Chapter 8: Complete circuit diagram and programming code

Complete circuit diagram:





Complete programming code:

```
#define BLYNK_TEMPLATE_ID "TMPL0V3dtmm5"                    //copy from server
#define BLYNK_DEVICE_NAME "Agriculture and Irrigation system"  //copy from server
#define BLYNK_AUTH_TOKEN "qJSUzdH0WAAWlzzstmLNgmnbhTv3caSz"    //copy from
server


#define DHTTYPE DHT11  //configure dht type module


#include <ESP8266WiFi.h>  // node library
#include <DHT.h>              //dht library
#include <BlynkSimpleEsp8266.h>  //blynk library


#define DHTPIN D2   // data pin  of  module to digital pin  of node in d2
#define SOILPIN A0  //soil moisture pin in analog  pin of a0
#define RAINPIN D6  //output pin of rain sensor connected to d6 of node
#define PUMPPIN D1 //pump pin connected to digital pin D1 of  node
float moisturepercentage;


int presentstate = 0;  //checks the present data of  soil moisture


char auth[] = BLYNK_AUTH_TOKEN;        // declare one character
char ssid[] = "Jalsa Karone Jentilal";  //declare  character variable called as ssid for wifi name
char pass[] = "qwertyuiop";           // declare character variable for wifi password


int rainval = 0;  // define a variable val initial value of 0 of rain sensor


//dht part
DHT dht = DHT(DHTPIN, DHTTYPE, 6);  //declare object of dht library
void airdata()  //declare void function called as airdata
{
  float t = dht.readTemperature();  // temerature data get and  store in t variable
```

```
 float h = dht.readHumidity();    //humidity data get and store in h variable
 if (isnan(t) || isnan(h))  // if data is not come then condition gets true  otherwise if data  is obtained then go to else part
 {
  Serial.println("fail to get the data from DHT11");
  return;
 }
 Blynk.virtualWrite(V1, t);  //v1 for temperature and  value of t given to v1
 Blynk.virtualWrite(V2, h);  //v2 for humidity given to v2
 Serial.print("humidity:");
 Serial.println(h, 1);
 Serial.print("temperature(celcius):");
 Serial.println(t, 1);
} // ending the airdata  dht function

void raindata() {
 //rain part
 rainval = digitalRead(RAINPIN);  // read the analog value to the variable rainval
 if (rainval == 1)     //if analog value is greater than the 700 then it shows the notification
 {
  Serial.println(" it is raining");
  Blynk.logEvent("rain_alert", "raining outside");  //event name and  notification message as raining outside
 }
else  //if rainval==0 it means if it  is not raining then go  in this condition
 {
  Serial.println(" it is not raining");
 }
 Blynk.virtualWrite(V3, rainval);  // virtual pin of dashborad as V3
 Serial.println(rainval);        //print the rainval value on serial monitor
} //ending the rain function
```

```
void soilmoisturefunc() {
  //soil moisture part
  presentstate = analogRead(SOILPIN);
  moisturepercentage = (100.0 - ((presentstate / 1024.0) * 100.0));
  Blynk.virtualWrite(V4, moisturepercentage);
  Serial.print("moisture of soil(percentage):");
  Serial.println(moisturepercentage, 2);  //2 means shows the decimal  point upto 2

  //1 means soil is dry and 0 means soil is moist enough
  if (moisturepercentage < 40)  //if it is greater than 500 level of moisture then it needs water
  {
    if (rainval == 0)  //when it is not raining the pump gets turn off
    {
      //water level of tank condition to be added here
      pinMode(D1,OUTPUT);
      digitalWrite(D1,LOW);
      Serial.println("relay is on");
     // digitalWrite(PUMPPIN, HIGH);  //pump led turn on connected to anode
      Serial.println(" water pump is on");
      Blynk.logEvent("pump_alert", "farm is being watered");  //event code= pump_alert and
notification message= farm is being  watered
    }

    else {
      pinMode(D1,INPUT);
      digitalWrite(D1,HIGH);
      Serial.println("relay is OFF");
      //digitalWrite(PUMPPIN, LOW);  //pump gets turn off when soil moisture level is greater
than 40
      Serial.println("water pump is off");
      Blynk.logEvent("off_alert","water pump is off");  //event code for pump off= off alert
```

```
    }

    Serial.println("need water");
  }

  else {
     pinMode(D1,INPUT);
     digitalWrite(D1,HIGH);
     Serial.println("relay is OFF");
    //digitalWrite(PUMPPIN, LOW);  //PUMP LED TURN OFF
    Serial.println("no need of water ");
  }
} //ending the soilmoisturefunc

//function for ultrasonic sensor to be added here
void setup() {
  Serial.begin(9600);
  dht.begin();                //to start the dht
  Blynk.begin(auth, ssid, pass);  //to  configure the blynk platform
  //pinMode(PUMPLED, OUTPUT);
  pinMode(SOILPIN, OUTPUT);  // data soil pin configure to analog pin  A0 of node
}

void loop() {
  airdata();          //calling the  function of dht parameter
  raindata();         //calling the rain function
  soilmoisturefunc();  //calling the soilmoisturefunction
  //function for ultrasonic sensor to be called here
  delay(5000);
  Blynk.run();  // for keeping connection with blynk like sending data and receiving  data
}
```

Chapter 9: Advantages and future aspects, limitations

Advantages:

- The user can monitor the farm 24/7
- Accurate measure of amount of moisture in the soil
- Reduces the human effort
- Less man-power is required
- When the weather is rainy, pump is turned off automatically which reduces the power consumption.
- All the data is sent to IoT platform; hence user can get data of previous hour, last day average of all time.

Limitations:

- It is little bit difficult to implement this prototype for the farms having huge area.
- NodeMCU is not sufficient for controlling too many sensors and actuator because of less number of GPIO pins and low clock speed.
- This system can work out fine for small area farms or gardens, for e.g. Plants, terrace garden. For large area a greater number of sensors will be required and it may increase the cost of product.

Future aspects:

- In future instead of using Blynk IoT, we can build our own mobile application which lets the user set the soil moisture level, temperature, humidity and amount of rain for different crops and plants, as per provided data the system works.
- The mobile app also lets the user set local Wi-Fi password and SSID.
- Since the number of data streams and API request for free account user is limited, the mobile application can set us free to connect as many sensors as we want.
- In future, as per the level of implementation, the NodeMCU may be replaced by controllers used by industry such as PLC, DCS etc. they have more number of pinouts.

Conclusion and learnings

In this project, we have learnt following things:

- Sending the sensor data to Blynk IoT platform.
- Sending the notification to the registered mobile using Blynk.
- Debugging of circuit.
- Interfacing different sensors to ESP8266.
- Logic building for irrigation unit.

References

https://theiotprojects.com/iot-smart-agriculture-automatic-irrigation-system-with-esp8266/

https://www.watelectronics.com/ultrasonic-sensor/

https://components101.com/sensors/dht11-temperature-sensor

https://components101.com/switches/5v-single-channel-relay-module-pinout-features-applications-working-datasheet