ADITYA COLLEGE OF ENGINEERING

(Affiliated to J.N.T.U.A, Anantapur & Approved by A.I.C.T.E, New Delhi).
PUNGANUR ROAD, MADANAPALLE-517325

DEPARTMENT OF ELECTRONICS & COMMUNICATION ENGINEERING

INTERNET OF THINGS LAB MANUAL

Name:_____

H.T.No: _____

Year/Semester:_____

Vision of the institute

To impart quality in engineering education to meet the technological advances and industrial requirements with global standards.

Mission of the institute

Mission_1: Provide quality technical education through skill-based trainings and promote research and development, and consultancy services.

Mission_2: Offer state-of-the-art infrastructure for supporting technological advances.

Mission_3: Develop disciplined, creative and globally competent engineers.

Mission_4: Equip and empower the faculty at all levels to promote innovations and technical advancements in various domains of engineering.

Vision of the department

To design Centre for excellence in the field of revolutionary electronic developments and the future communication to create quality engineers through innovation and impact professional ethics with a good teamwork to fulfil the social needs.

Mission of the department

Mission_1: To creating graduates with technical expertise, professional, attitude and ethical values by establishing top notch learning environment.

Mission_2: Inculcating creative thoughts through innovative and team work-based methods to develop entrepreneurship skills and employability among professionals.

Mission_3: Strengthening soft skills to rural students through cocurricular and extracurricular activities to face industrial challenges.

Programme Educational Objectives (PEOs)

PEO 1: To be equipped with skills for solving complex real- world problems related to VLSI, Embedded Systems, Signal/Image processing, and Digital and Wireless Communication.

PEO-2: To communicate effectively, work collaboratively and exhibit high levels of professionalism, moral and ethical responsibility.

PEO-3: To develop professional skills that will equip them to succeed in their careers and encourage lifelong learning in advanced areas of Electronics and communications and related fields.

PEO-4: To develop the ability to understand and analyze engineering issues in a broader perspective with ethical responsibility towards sustainable development

Programme Outcomes(Pos)

PO_1	**Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
PO_2	**Problem analysis:** Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
PO_3	**Design/development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
PO_4	**Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
PO_5	**Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.
PO_6	**The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
PO_7	**Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
PO_8	**Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
PO_9	**Individual and team work:** Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
PO_10	**Communication:** Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
PO_11	**Project management and finance:** Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
PO_12	**Life-long learning:** Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

Programme Specific Outcome(PSOs)

PSO_1	Able to identify, formulate, analyze, and solve engineering problems and apply them to analog and digital electronics, communication, signal processing, VLSI systems, embedded systems, IoT, and other multidisciplinary domains.
PSO_2	Ability to understand electronics and Communication Engineering changes and future trends and apply them to electronic system design using hardware, software, and electronic design automation tools.

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY ANANTAPUR

B.Tech (CSE)– III-II Sem

L	T	P	C
0	0	3	1.5

(20A05603P) INTERNET OF THINGS LAB

Course Objectives:

- To introduce components such as WiFi, Bluetooth, Temperature, Moisture sensors
- To know the Micro controller such as Arduino
- To know the System on Chip (SOC) / Single Board Computer such as Raspberry Pi
- To understand HTTP IoT protocols and perform Experiments for data transmission
- To understand UAV/Drones and Internet of Drones Experiments

Course Outcomes:

After completion of the course, students will be able to

- Know the various IoT sensors and understand the functionality
- Design and analyze IoT experiments and transfer the data to IoT Clouds
- Design the IoT systems for real time applications
- Understand Drones and Perform Internet of Drones Experiments

List of Experiments:

Experiments using ESP32

1. Serial Monitor, LED, Servo Motor - Controlling

- **Experiment1:**
  Controlling actuators through Serial Monitor. Creating different led patterns and controlling them using push button switches. Controlling servo motor with the help of joystick.

2. **Distance Measurement of an object**

- **Experiment 2:**
  Calculatethe distance to an object with the help of an ultrasonic sensor and display it on an LCD.

3, **LDR Sensor, Alarm and temperature, humidity measurement**

Experiment 3:

- Controlling relay state based on ambient light levels using LDR sensor.
- Basic Burglar alarm security system with the help of PIR sensor and buzzer.
- Displaying humidity and temperature values on LCD

4. Experiments using Raspberry Pi

Experiment 4:

- Controlling relay state based on input from IR sensors
- Interfacing stepper motor with R-Pi
- Advanced burglar alarm security system with the help of PIR sensor, buzzer and keypad. (Alarm gets disabled if correct keypad password is entered)
- 5. Automated LED light control based on input from PIR (to detect if people are present) and LDR(ambient light level)

5. IOT Framework

Experiment 5:

Upload humidity & temperature data to ThingSpeak, periodically logging ambient light level to ThingSpeak

Experiment 6:

Controlling LEDs, relay & buzzer using Blynk app

6. HTTP Based

Experiment 7:

- Introduction to HTTP. Hosting a basic server from the ESP32 to control various digital based actuators (led, buzzer, relay) from a simple web page.

Experiment 8:

- Displaying various sensor readings on a simple web page hosted on the ESP32.

7. MQTT Based

Experiment 9:

Controlling LEDs/Motors from an Android/Web app, Controlling AC Appliances from an android/web app with the help of relay.

Experiment 10:

Displaying humidity and temperature data on a web-based application

8. UAV/Drone:

Experiment 11:

- Demonstration of UAV elements, Flight Controller
- Mission Planner flight planning design

Experiment 12:

- Python program to read GPS coordinates from Flight Controller

Reference:

1. Adrian McEwen, Hakim Cassimally - Designing the Internet of Things, Wiley Publications, 2012.
2. Alexander Osterwalder, and Yves Pigneur – Business Model Generation – Wiley, 2011
3. ArshdeepBahga, Vijay Madisetti - Internet of Things: A Hands-On Approach, Universities Press, 2014.
4. The Internet of Things, Enabling technologies and use cases – Pethuru Raj, Anupama C. Raman, CRC Press.

Online Learning Resources/Virtual Labs:

https://www.arduino.cc/
https://www.raspberrypi.org/

(20A05603P) INTERNET OF THINGS LAB

List of Experiments to be conducted

1. Serial Monitor, LED, Servo Motor - Controlling

- **Experiment1:**
  Controlling actuators through Serial Monitor. Creating different led patterns and controlling them using push button switches. Controlling servo motor with the help of joystick.

2. Distance Measurement of an object

- **Experiment 2:**
  Calculatethe distance to an object with the help of an ultrasonic sensor and display it on an LCD.

3, **LDR Sensor, Alarm and temperature, humidity measurement**

Experiment 3:

- Controlling relay state based on ambient light levels using LDR sensor.
- Basic Burglar alarm security system with the help of PIR sensor and buzzer.
- Displaying humidity and temperature values on LCD

4. Experiments using Raspberry Pi

Experiment 4:

- Controlling relay state based on input from IR sensors
- Interfacing stepper motor with R-Pi
- Advanced burglar alarm security system with the help of PIR sensor, buzzer and keypad. (Alarm gets disabled if correct keypad password is entered)
- 5. Automated LED light control based on input from PIR (to detect if people are present) and LDR(ambient light level)

5. IOT Framework

Experiment 5:

Upload humidity & temperature data to ThingSpeak, periodically logging ambient light level to ThingSpeak

Experiment 6:

Controlling LEDs, relay & buzzer using Blynk app

6. HTTP Based

Experiment 7:

- Introduction to HTTP. Hosting a basic server from the ESP32 to control various digital based actuators (led, buzzer, relay) from a simple web page.

Experiment 8:

- Displaying various sensor readings on a simple web page hosted on the ESP32.

7. MQTT Based

Experiment 9:

Controlling LEDs/Motors from an Android/Web app, Controlling AC Appliances from an android/web app with the help of relay.

Experiment 10:

Displaying humidity and temperature data on a web-based application

8. UAV/Drone:

Experiment 11:

- Demonstration of UAV elements, Flight Controller
- Mission Planner flight planning design

Experiment 12:

- Python program to read GPS coordinates from Flight Controller

ADDITIONAL EXPERIMENTS

1. Design an IoT based air pollution control system which monitors the air pollution by measuring carbon monoxide, ammonia, etc and gives alarm or sends message when the pollution level is more than permitted range.

2. Design an IoT based system which measures the physical and chemical properties of the water and displays the measured values.

CONTENTS

DOS & DONTS IN LABORATORY

DO's

1. Students should be punctual and regular to the laboratory.
2. Students should come to the lab in-time with proper dress code.
3. Students should maintain discipline all the time and obey the instructions.
4. Students should carry observation and record completed in all aspects.
5. Students should be at their concerned experiment table, unnecessary moment is restricted.
6. Students should follow the indent procedure to receive and deposit the components from lab technician.
7. While doing the experiments any failure/malfunction must be reported to the faculty.
8. Students should check the connections of circuit properly before switch ON the power supply.
9. Students should verify the reading with the help of the lab instructor after completion of experiment.
10. Students must endure that all switches are in the lab OFF position, all the connections are removed.
11. At the end of practical class the apparatus should be returned to the lab technician and take back the indent slip.
12. After completing your lab session SHUTDOWN the systems, TURNOFF the power switches and arrange the chairs properly.
13. Each experiment should be written in the record note book only after getting signaturefrom the lab in charge in the observation notebook.

DON'Ts

1. Don't eat and drink in the laboratory.
2. Don't touch electric wires.
3. Don't turn ON the circuit unless it is completed.
4. Avoid making loose connections.
5. Don't leave the lab without permission.
6. Don't bring mobiles into laboratory.
7. Do not open any irrelevant sites on computer.
8. Don't use a flash drive on computers.

SCHEME OF EVALUVATION

S No	Date	Name Of The Experiment	Marks Awarded			Sign.
			Observation (10M)	Viva voce (10M)	Total (20M)	
1.		Controlling actuators through Serial Monitor. Creating different led patterns and controlling them using push button switches. Controlling servo motor with the help of joystick.				
2.		Calculate the distance to an object with the help of an ultrasonic sensor and display it on an LCD.				
3.		• Controlling relay state based on ambient light levels using LDR sensor. • Basic Burglar alarm security system with the help of PIR sensor and buzzer. • Displaying humidity and temperature values on LCD				
4		Controlling relay state based on input from IR sensors • Interfacing stepper motor with R-Pi • Advanced burglar alarm security system with the help of PIR sensor, buzzer and keypad. (Alarm gets disabled if correct keypad password is entered) • Automated LED light control based on input from PIR (to detect if people are present) and LDR(ambient light level)				
5		Upload humidity & temperature data to ThingSpeak, periodically logging ambient light level to ThingSpeak				
6		Controlling LEDs, relay & buzzer using Blynk app				
7		Introduction to HTTP. Hosting a basic server from the ESP32 to control various digital based actuators (led, buzzer, relay) from a simple web page.				
8		Displaying various sensor readings on a simple web page hosted on the ESP32.				

9		Controlling LEDs/Motors from an Android/Web app, Controlling AC Appliances from an android/web app with the help of relay.				
10		Displaying humidity and temperature data on a web-based application				
11		• Demonstration of UAV elements, Flight Controller • Mission Planner flight planning design				
12		Python program to read GPS coordinates from Flight Controller				
ADDITIONAL EXPERIMENTS						
1		Design an IoT based air pollution control system which monitors the air pollution by measuring carbon monoxide, ammonia, etc and gives alarm or sends message when the pollution level is more than permitted range.				
2		Design an IoT based system which measures the physical and chemical properties of the water and displays the measured values.				

Signature of Lab In-charge

EXPERIMENT 1
Serial Monitor, LED, Servo Motor - controlling

AIM:

(a) To write a program that controls actuators through serial monitor.

(b) To write a program that controls LED through push button.

(c) To write a program that controls servo motor with the help of joystick.

COMPONENT REQUIRED:

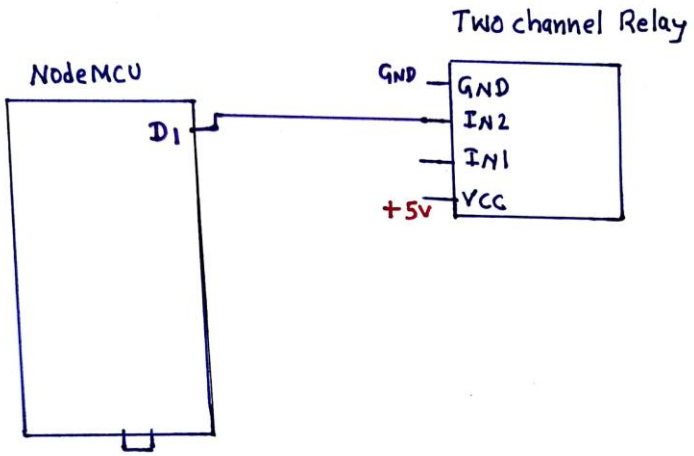
1. NodeMCU,
2. Arduino uno
3. Micro USB cable
4. LED
5. Resistors (470Ω,10kΩ)
6. Push button
7. Servo motors
8. Joystick
9. Jumper wires

Procedure:

1. Open Arduino IDE
2. Open new sketch and type the program.
3. Compile the program.
4. Connect the NODEMCU board with USB cable to the computer.
5. Select tools -> select board ->NodeMCU 1.0
6. Select tools -> select port.
7. Upload the program.
8. Connect the circuit as per the circuit diagram.
9. Observe the output.

(a) Turn on Turn off relay connected to D1 pin of nodeMCU through Serial Monitor

Circuit diagram

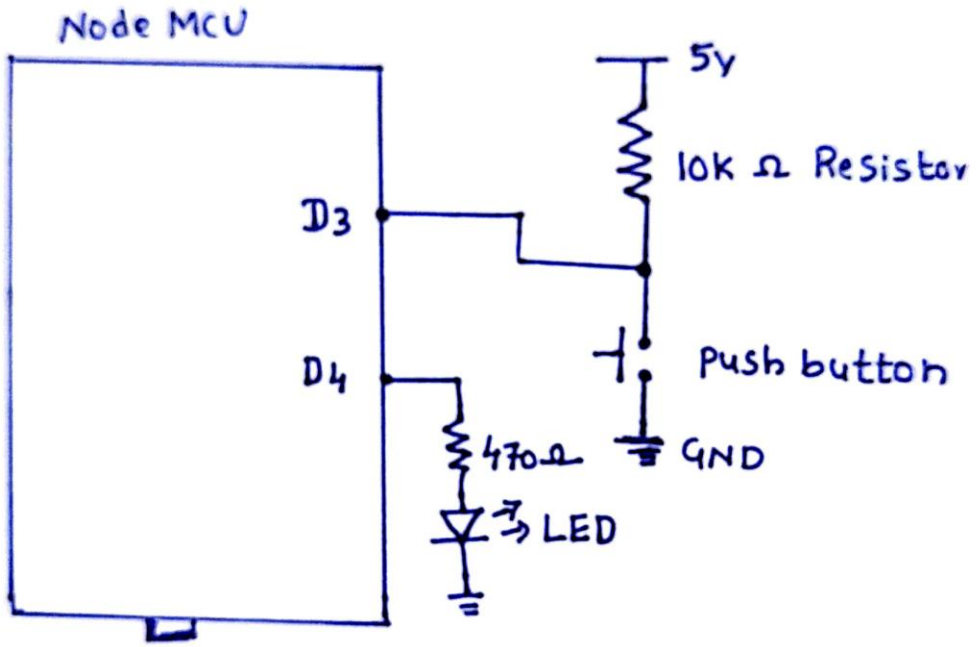


CODE:

```
String voice = "";
void setup()
{
  Serial.begin(9600);
  pinMode(D1,OUTPUT);
}
void loop()
{
  while(Serial.available())
  {
    char c = Serial.read();
    voice = voice + c;
  }
 Serial.println(voice);
 if(voice == "light on")
 {
  digitalWrite(D1,LOW);// Relay module needs a low to TURN ON
 }
 if(voice == "light off")
 {
  digitalWrite(D1,HIGH);// Relay module needs a HIGH to TURN OFF
 }
 voice = "";
 delay(1000);
}
```

(b) Creating different led patterns and controlling them using push button switches

Circuit diagram



CODE:

```
void setup() {
pinMode(D3,INPUT);
pinMode(D4,OUTPUT);
}
void loop() {
int button=digitalRead(D3);
if(button==1){
digitalWrite(D4,LOW);
delay(100);
}
else{
digitalWrite(D4,HIGH);
delay(100);
}
}
```

(c) Controlling servo motor with the help of joystick

Connect the circuit as per the figure below

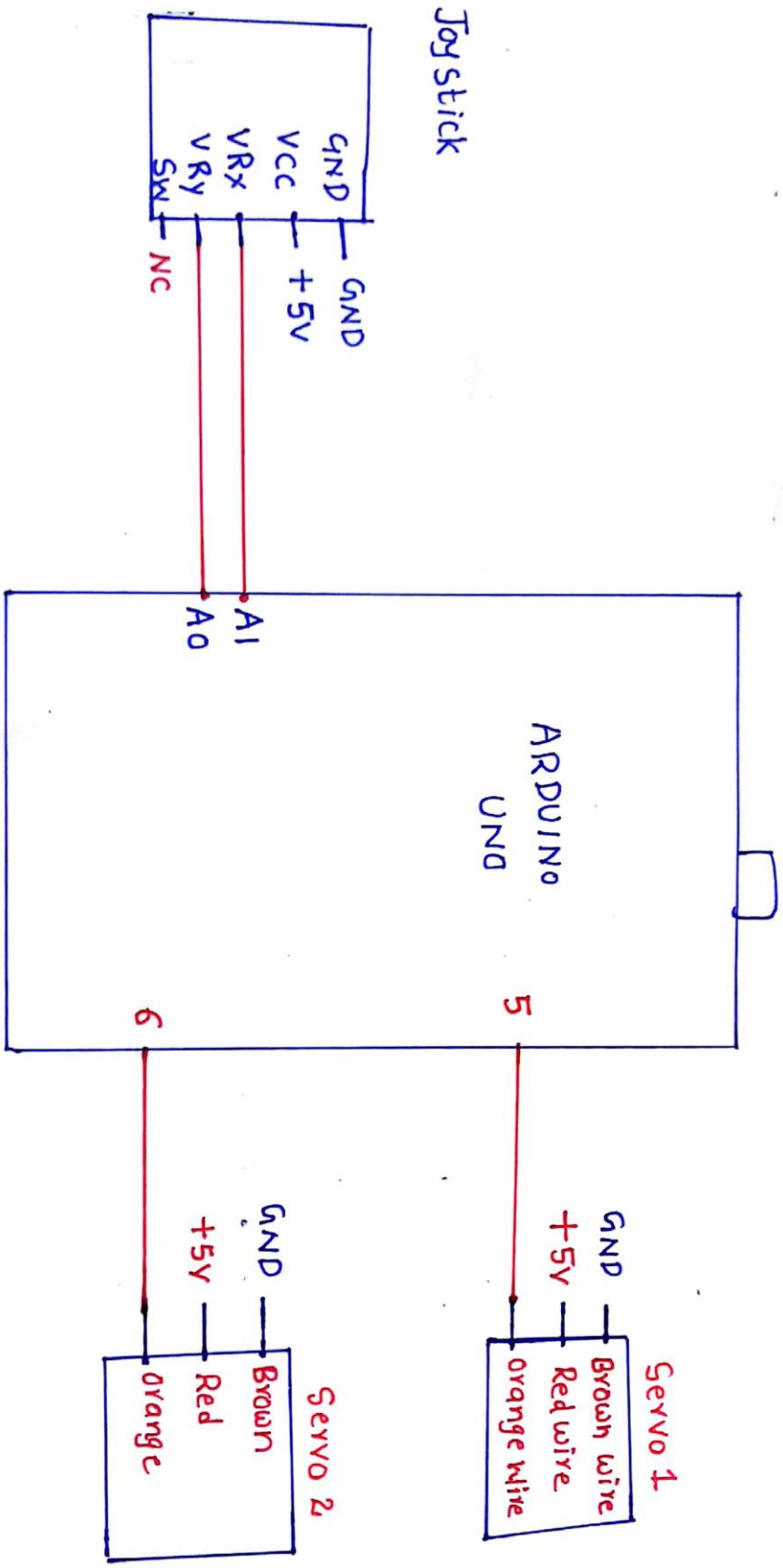
The connections for the joystick module and the Arduino are as follows:

- Connect the VCC on the joystick module with the 5V pin on the Arduino
- Connect the GND pin on the joystick module with the GND on the Arduino
- Connect the VRy pin on the joystick module with the A0 on the Arduino
- Connect the VRx pin on the joystick module with the A1 on the Arduino

After that, connect the servo motors with the Arduino. The connections for servo motors with Arduino are as follows:

- Connect the black wire on both the servo motors with the GND on the Arduino
- Connect the red wire on both the servo motors with the 5V pin on the Arduino
- Connect the yellow/orange wire on the first motor with pin 5 on the Arduino
- Connect the yellow/orange wire on the second motor with pin 6 on the Arduino

Circuit diagram



Code:

```
#include <Servo.h>
Servo MyServo1;
Servo MyServo2;
int Servo1_Position;
int Servo2_Position;

void setup() {
  Serial.begin(9600);
  MyServo1.attach(5);
  MyServo2.attach(6);
}

void loop() {
    Serial.print("Y:");
    Serial.println(analogRead(A0));
    Serial.print("X:");
    Serial.println(analogRead(A1));
   //Controlling Servo 1
    Servo1_Position = analogRead(A0);  //Reading Joystick Y position
(Verical)
    Servo1_Position = map(Servo1_Position, 0, 1023, 0, 180);
    MyServo1.write(Servo1_Position);
    delay(15);

    //Controlling Servo 2
    Servo2_Position = analogRead(A1);  //Reading Joystick X position
(Horizontal)
    Servo2_Position = map(Servo2_Position, 0, 1023, 0, 180);
    MyServo2.write(Servo2_Position);
    delay(15);
}
```

Viva Questions:

1. **What is Serial Monitor?**

2. **What is servo motor?**

3. **What is Joystick?**

4. **What is push button?**

5. **What is NoeMCU?**

6. **What is GPIO?**

7. **What is analog pin?**

Result:

Actuator is controlled through Serial monitor. LEDs are controlled through pushbutton. Servo motor is controlled through joystick

EXPERIMENT 2
Distance Measurement of an object

AIM:

To write a program that calculates the distance to an object with the help of an ultrasonic sensor and displays it on an LCD.

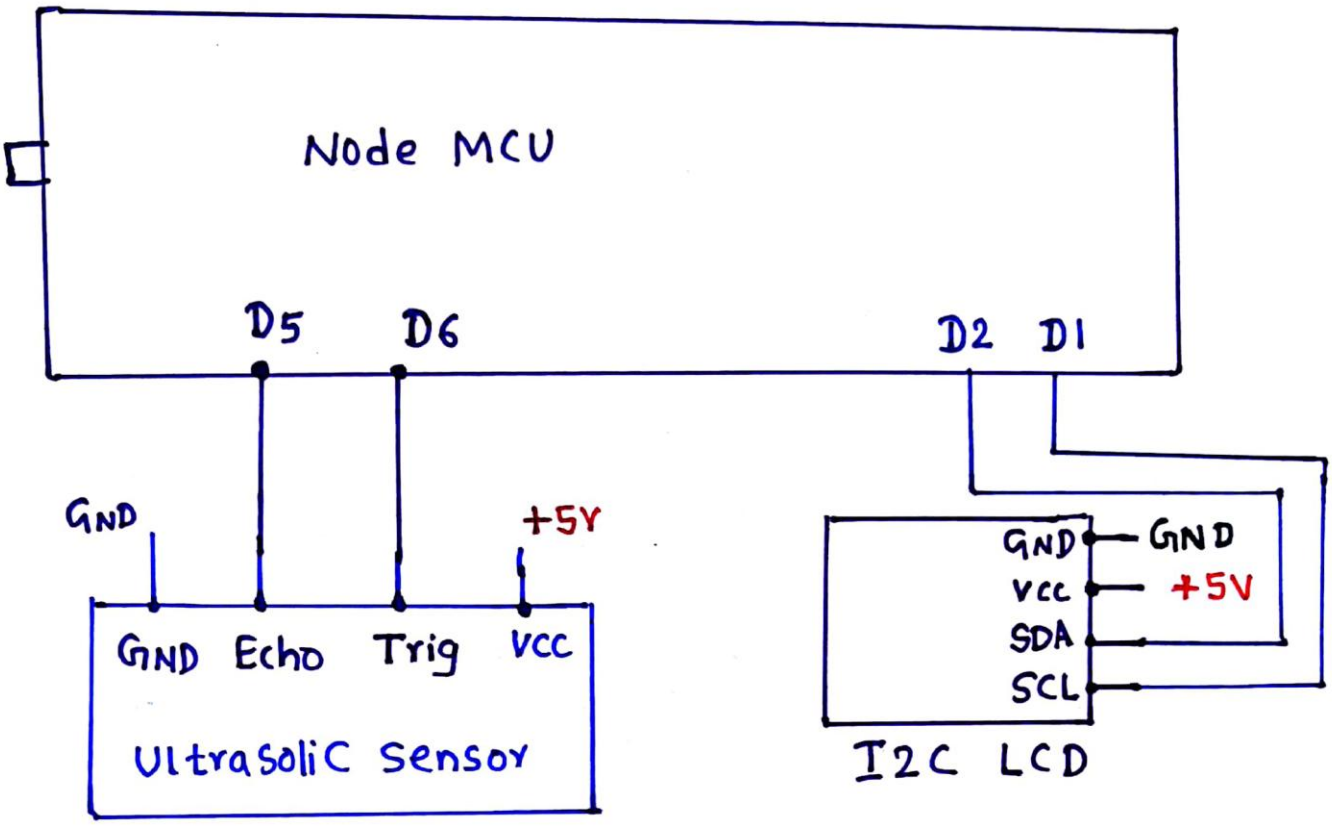
COMPONENT REQUIRED:

1. NodeMCU,
2. Micro USB cable
3. I2C LCD
4. Ultrasonic sensor
5. Jumper wires

Procedure:

1. Open Arduino IDE
2. Open new sketch and type the program.
3. Compile the program.
4. Connect the NODEMCU board with USB cable to the computer.
5. Select tools -> select board ->NodeMCU 1.0
6. Select tools -> select port.
7. Upload the program.
8. Connect the circuit as per the circuit diagram.
9. Observe the output.

Circuit diagram



CODE:

```
#define trigger_pin 12   //D6 pin of Nodemcu
#define Echo_pin 14      //D5 pin of Nodemcu

#include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd(0x27,16,2);

/* two variables to store duraion and distance value */
long duration;
int distance;

/* configure D5 and D6 as digital input and output respectively */
void setup() {
  pinMode(trigger_pin, OUTPUT); // configure the trigger_pin(D9) as an
Output
  pinMode(Echo_pin, INPUT); // configure the Echo_pin(D11) as an Input
  Serial.begin(9600); // Enable the serial with 9600 baud rate
  lcd.init();
  lcd.backlight();
}

void loop() {

digitalWrite(trigger_pin, LOW); //set trigger signal low for 2us
delayMicroseconds(2);

/*send 10 microsecond pulse to trigger pin of HC-SR04 */
digitalWrite(trigger_pin, HIGH);  // make trigger pin active high
delayMicroseconds(10);            // wait for 10 microseconds
digitalWrite(trigger_pin, LOW);   // make trigger pin active low

/*Measure the Echo output signal duration or pulss width */
```

```
duration = pulseIn(Echo_pin, HIGH); // save time duration value in
"duration variable

distance= duration*0.034/2; //Convert pulse duration into distance


// print measured distance value on Arduino serial monitor

Serial.print("Distance: ");

Serial.print(distance);

Serial.println(" cm");


// print measured distance value on LCD display

lcd.setCursor(0,0);

lcd.print("Distance=");

lcd.print(distance);

delay(500);

lcd.clear();

}
```

Viva Questions

1. **What is Ultrasonic sensor?**

2. **What is normal LCD display?**

3. **What is I2C LCD display?**

4. **What is I2C communication?**

5. **What is UART communication?**

Result:

Object distance is measured using ultrasonic sensor, and it is displayed on LCD.

EXPERIMENT 3
LDR Sensor, Alarm and temperature, humidity measurement

AIM:

a) To write a program that controls relay state based on ambient light levels using LDR sensor.
b) To write a program for Basic Burglar alarm security system with the help of PIR sensor and buzzer.
c) To write a program Displaying humidity and temperature values on LCD.

COMPONENTS REQUIRED:

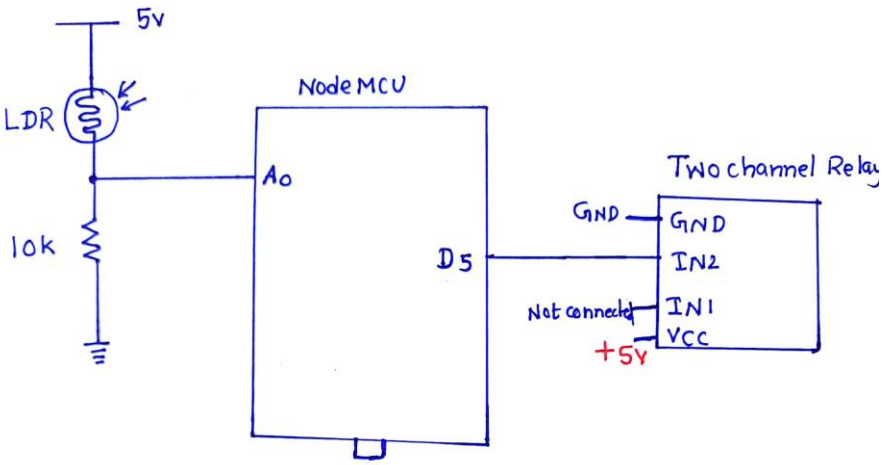
1. NodeMCU
2. Micro USB cable
3. LDR
4. 10k Ohm Resistor
5. 5V Relay
6. PIR sensor
7. Buzzer
8. DHT11 Sensor
9. I2C LCD
10. Jumper wires

Procedure:

1. Open Arduino IDE
2. Open new sketch and type the program.
3. Compile the program.
4. Connect the NODEMCU board with USB cable to the computer.
5. Select tools -> select board ->NodeMCU 1.0
6. Select tools -> select port.
7. Upload the program.
8. Connect the circuit as per the circuit diagram.
9. Open serial monitor.
10. Observe the output.

(a) Controlling relay state based on ambient light levels using LDR sensor

Circuit diagram

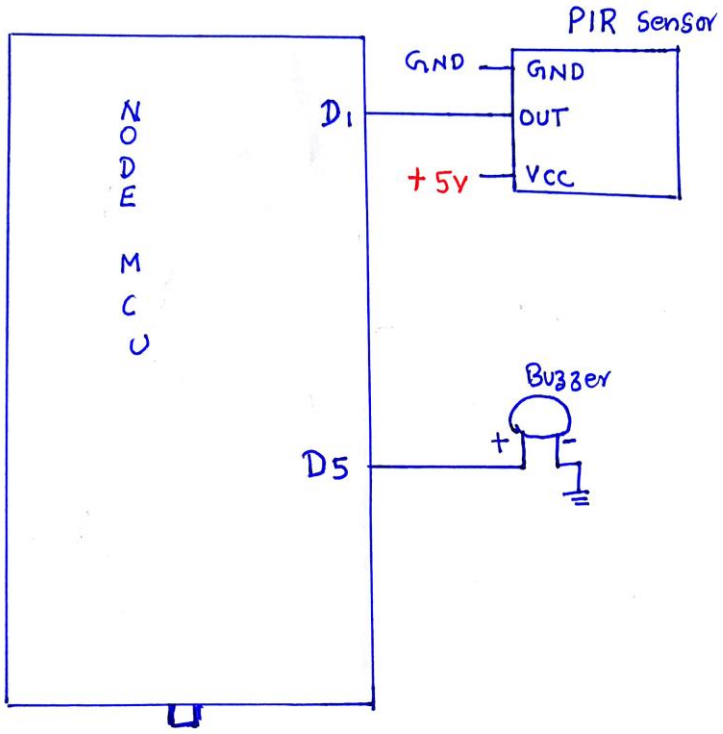


Code:

```
const int ledPin = 5;
const int ldrPin = A0;
void setup()
{
      Serial.begin(9600);
      pinMode(ledPin, OUTPUT);
      pinMode(ldrPin, INPUT);
}
void loop()
{
      int ldrStatus = analogRead(ldrPin);
      if (ldrStatus <=300)
      {
         digitalWrite(ledPin, HIGH);
         Serial.print(ldrStatus);
         Serial.println("LDR is DARK, Relay is ON");
      }
      else
      {
         digitalWrite(ledPin, LOW);
         Serial.println("Relay is OFF");
      }
}
```

(b) Basic Burglar alarm security system with the help of PIR sensor and buzzer

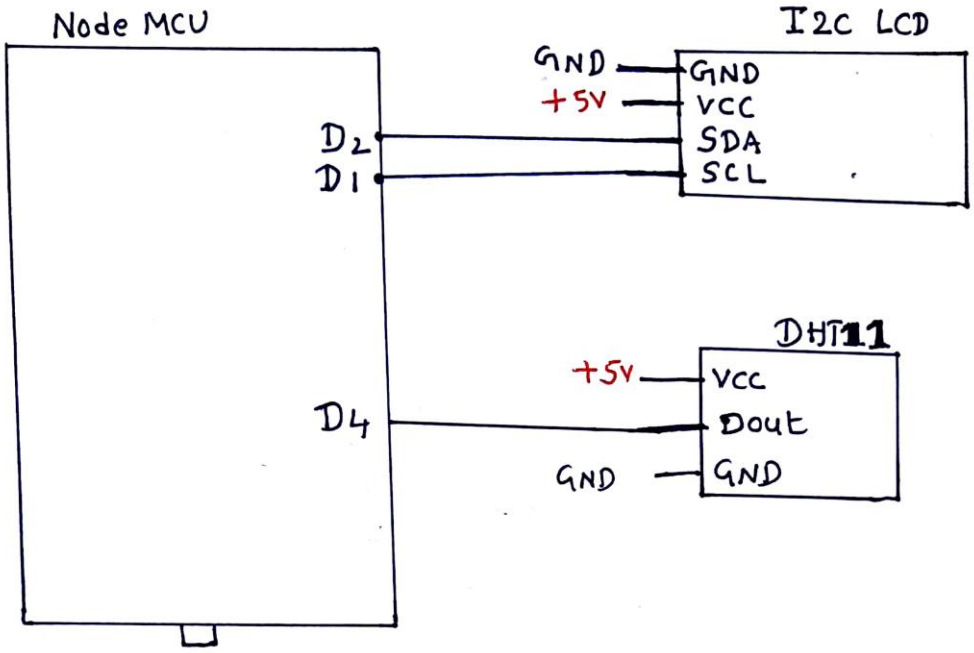
Circuit diagram



Code:

```
int buzzer = D5;   //Buzzer  alarm  connected to D5 of nodemcu
int PIRsensor = D1; //PIR sensor output connected to D1 of nodemcu

void setup() {
  Serial.begin(9600);
  pinMode(PIRsensor, INPUT); // PIR sensor as input
  pinMode(buzzer, OUTPUT);   // Buzzer alaram as output
  digitalWrite (buzzer, LOW);// Initially buzzer off
}

void loop(){
  int state = digitalRead(PIRsensor); //Continuously check the state of PIR
sensor
  delay(500);   //Check state of PIR after every half second

    if(state == HIGH){
      digitalWrite (buzzer, HIGH); //If intrusion detected ring the buzzer
      delay(5000);
      delay(5000);
      delay(5000);              //Ring buzzer for 15 seconds
      Serial.println("Motion detected!");
    }
    else {
      digitalWrite (buzzer, LOW);     //No intrusion Buzzer off
      Serial.println("Motion absent!");
      }
}
```

(C) Displaying humidity and temperature values on LCD



Code:

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd(0x27,16,2);
#include "DHT.h"
#define DHTTYPE DHT11
#define dht_dpin D4
DHT dht(dht_dpin, DHTTYPE);
void setup(void)
{
  dht.begin();
  lcd.init();
  lcd.backlight();
  Serial.begin(9600);
}
void loop() {
    float h = dht.readHumidity();
    float t = dht.readTemperature();
    Serial.print("Current humidity = ");
    Serial.print(h);
    Serial.println("%  ");
    Serial.print("temperature = ");
    Serial.print(t);
    Serial.println("C  ");
    lcd.setCursor(0,0);
    lcd.print("humidity = ");
    lcd.print(h);
    lcd.print("%");
    lcd.setCursor(0,1);
    lcd.print("temperature = ");
    lcd.print(t);
    lcd.print("C  ");
    delay(800);
}
```

Viva Questions

1. **What is LDR sensor?**

2. **What is relay?**

3. **What is PIR sensor?**

4. **What is buzzer?**

5. **What is temperature sensor and give some examples?**

Result:

Relay is controlled based on ambient light using LDR sensor. Basic Burglar alarm security system is made with the help of PIR sensor and buzzer. Displayed humidity and temperature values on LCD.

EXPERIMENT 4
Experiments using Raspberry Pi

AIM:

(a) To Control relay state based on input from IR sensors
(b) To Interface stepper motor with R-Pi
(c) To create Advanced burglar alarm security system with the help of PIR sensor, buzzer and keypad. (Alarm gets disabled if correct keypad password is entered)
(d) To create Automated LED light control based on input from PIR (to detect if people are present) and LDR (ambient light level)

COMPONENT REQUIRED:

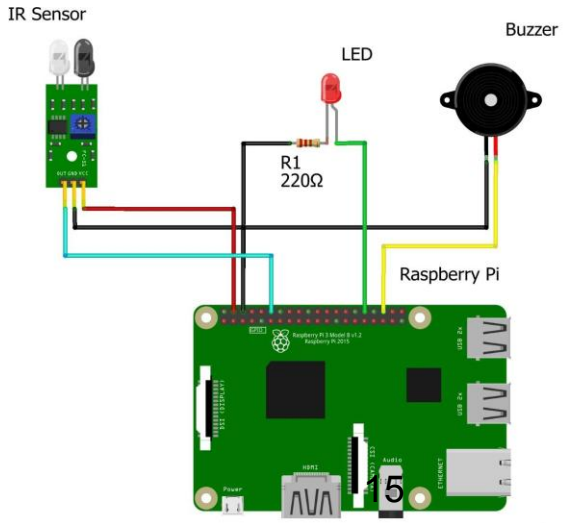
1. Rasberry pi
2. IR Sensor
3. PIR Sensor
4. Relay
5. Buzzer
6. Stepper motor
7. keypad
8. ULN2003
9. Micro USB cable
10. LED
11. Resistors (470Ω,10kΩ)
12. Jumper wires

Procedure:

1. Connect the raspberry pi board with monitor and turn on power supply.
2. Open python IDE
3. Open new file and type the program.
4. Save the program.
5. Connect the circuit as per the circuit diagram.
6. Observe the output.

(a) Control relay state based on input from IR sensors

Circuit diagram

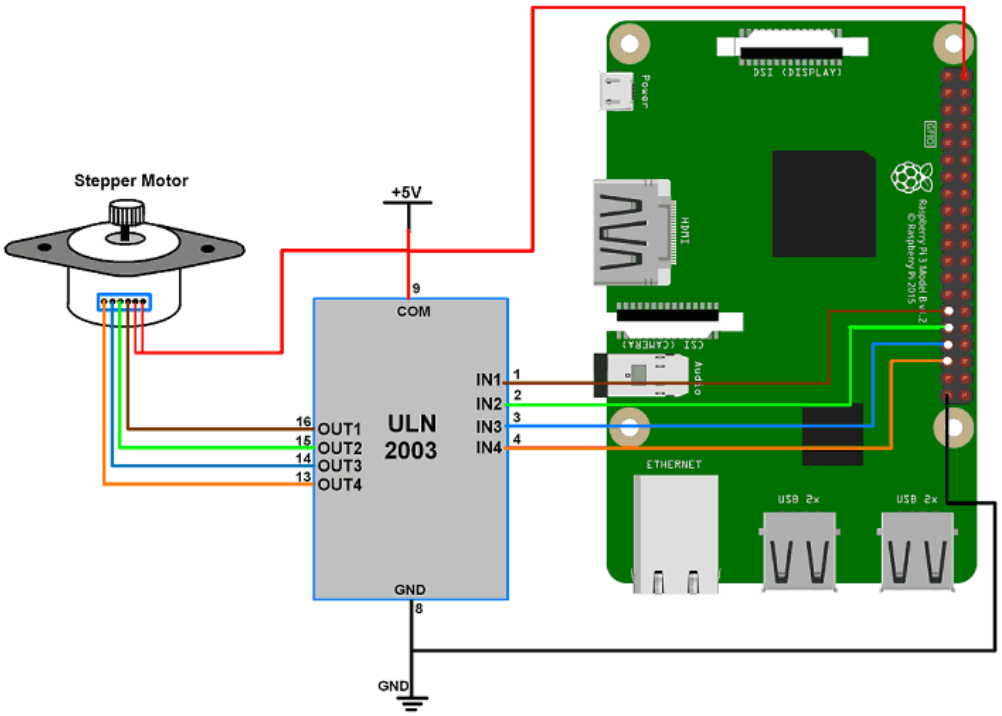


CODE:

```
import RPi.GPIO as gp
 from time import sleep
 gp.setmode(gp.BOARD)
 gp.setup(12,gp.IN)
 gp.setup(32,gp.OUT)
 gp.setup(36,gp.OUT)
 while True:
   print(not gp.input(12))
   gp.output(32,not gp.input(12))
   gp.output(36,not gp.input(12))
```

(b) Interface stepper motor with R-Pi

Circuit diagram



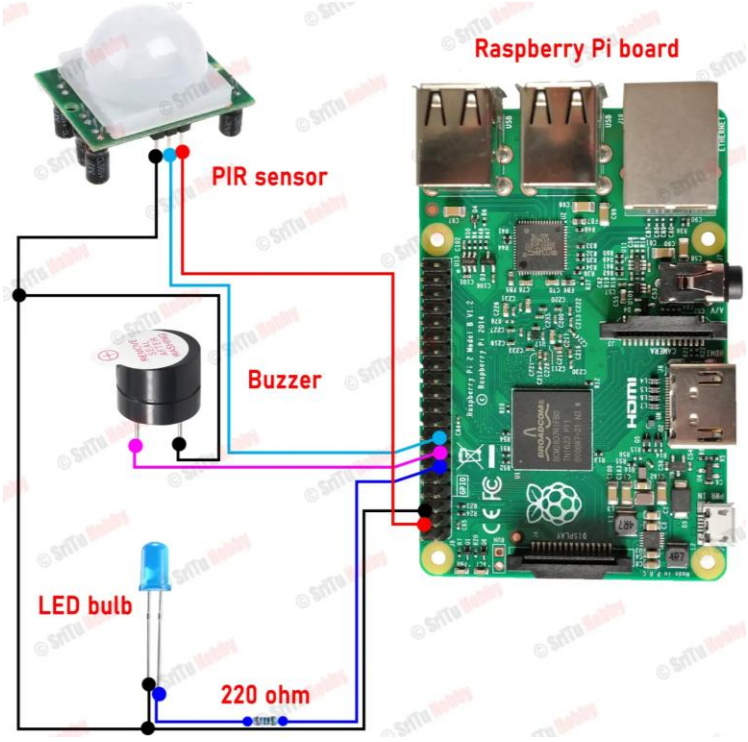
CODE:

```
import RPi.GPIO as GPIO
from time import sleep
import sys

#assign GPIO pins for motor
motor_channel = (29,31,33,35)
GPIO.setwarnings(False)
GPIO.setmode(GPIO.BOARD)
#for defining more than 1 GPIO channel as input/output use
GPIO.setup(motor_channel, GPIO.OUT)

motor_direction = input('select motor direction a=anticlockwise, c=clockwise: ')

while True:
    try:
```

```
        if(motor_direction == 'c'):
            print('motor running clockwise\n')
            GPIO.output(motor_channel, (GPIO.HIGH,GPIO.LOW,GPIO.LOW,GPIO.HIGH))
            sleep(0.02)
            GPIO.output(motor_channel, (GPIO.HIGH,GPIO.HIGH,GPIO.LOW,GPIO.LOW))
            sleep(0.02)
            GPIO.output(motor_channel, (GPIO.LOW,GPIO.HIGH,GPIO.HIGH,GPIO.LOW))
            sleep(0.02)
            GPIO.output(motor_channel, (GPIO.LOW,GPIO.LOW,GPIO.HIGH,GPIO.HIGH))
            sleep(0.02)

        elif(motor_direction == 'a'):
            print('motor running anti-clockwise\n')
            GPIO.output(motor_channel, (GPIO.HIGH,GPIO.LOW,GPIO.LOW,GPIO.HIGH))
            sleep(0.02)
            GPIO.output(motor_channel, (GPIO.LOW,GPIO.LOW,GPIO.HIGH,GPIO.HIGH))
            sleep(0.02)
            GPIO.output(motor_channel, (GPIO.LOW,GPIO.HIGH,GPIO.HIGH,GPIO.LOW))
            sleep(0.02)
            GPIO.output(motor_channel, (GPIO.HIGH,GPIO.HIGH,GPIO.LOW,GPIO.LOW))
            sleep(0.02)


    #press ctrl+c for keyboard interrupt
    except KeyboardInterrupt:
        #query for setting motor direction or exit
        motor_direction = input('select motor direction a=anticlockwise, c=clockwise or q=exit: ')
        #check for exit
        if(motor_direction == 'q'):
            print('motor stopped')
            sys.exit(0)
```

(c) Advanced burglar alarm security system

Circuit diagram



Connect the circuit as per the figure above

Code :

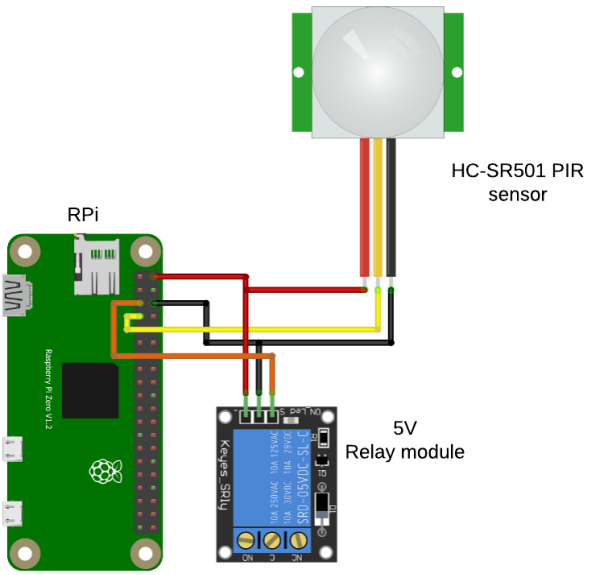
```
#include the library
from gpiozero import MotionSensor,Buzzer,LED

# include the PIR sensor pin
pirSensor = MotionSensor(22)
# Include the buzzer pin
buzzer = Buzzer(27)
# Include the LED bulb pin
led = LED(17)


while True:
    pirSensor.wait_for_motion()# Detect the motion
    buzzer.on()# Turn on the buzzer
    led.on() # Turn on the LED bulb
    print("Motion")
    pirSensor.wait_for_no_motion()# Detect the no motion
    print("No motion")
    buzzer.off()#Turn off the buzzer
    led.off()# Turn off the LED bulb
```

(d) Automated LED light control based on input from PIR

Circuit diagram

Code:

```
from gpiozero import MotionSensor, LED
from time import sleep

pir = MotionSensor(4)
relay = LED(17)

while True:
  pir.when_motion = relay.on
  pir.when_no_motion = relay.off
```



Viva Questions:

1. **What is raspberry pi pico?**

2. **What is servo motor?**

3. **What is PIR sensor?**

4. **What is LDR sensor?**

5. **Write all GPIO pins in raspberry pi?**

6. **What is raspberry pi?**

Result:

- Relay is controlled based on input from IR sensors
- Stepper motor is interfaced with R-Pi
- Burglar alarm security system is created with the help of PIR sensor
- Automated LED light is made based on input from PIR

EXPERIMENT 5

Upload humidity & temperature data to ThingSpeak

AIM:

To write a program to read data from DHT11 sensor and send the data to Blynk IoT cloud.

COMPONENTS REQUIRED:

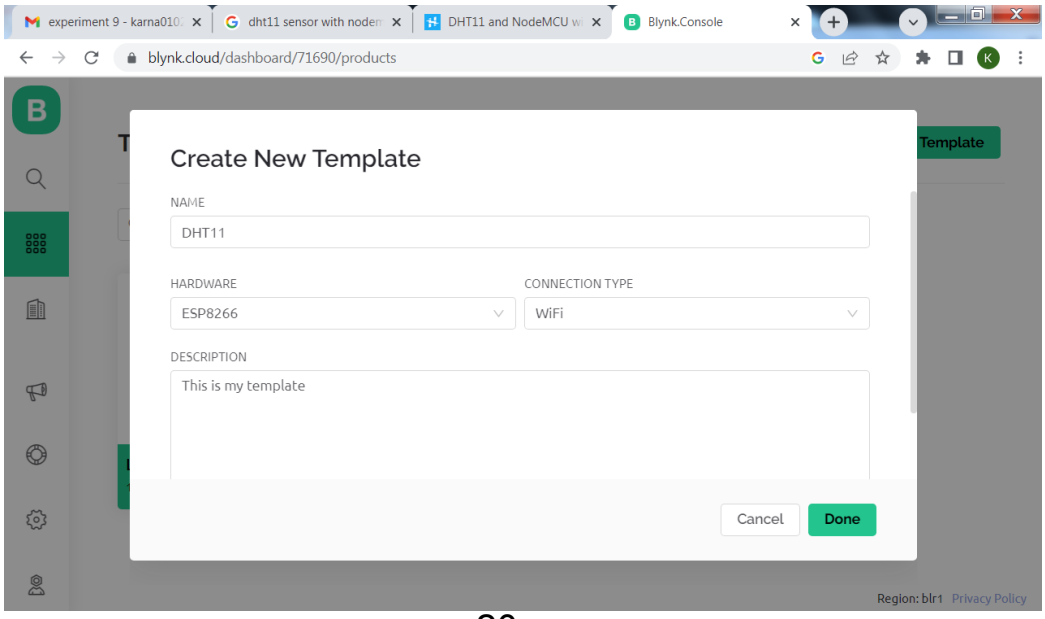
1. NodeMCU
2. Micro USB cable
3. DHT11 Sensor
4. Jumper wires

Procedure:

1. Open Arduino IDE
2. Open new sketch and type the program.
3. Compile the program.
4. Connect the NODEMCU board with USB cable to the computer.
5. Select tools -> select board ->NodeMCU 1.0
6. Select tools -> select port.
7. Upload the program.
8. Connect the circuit as per the circuit diagram.
9. Open web dashboard in blynk.
10. Observe the output.

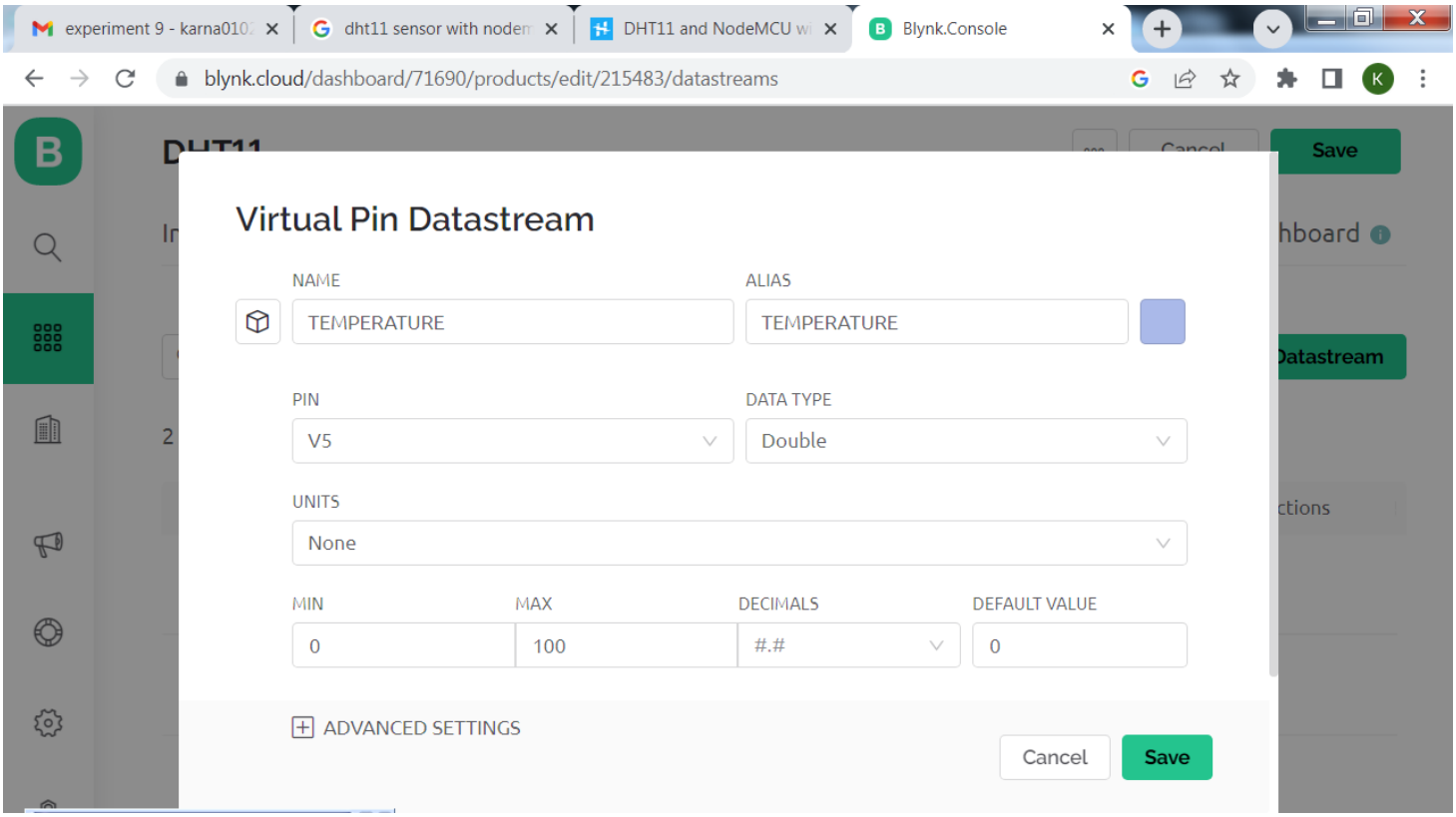
Step 2: Creating template

1. Open https://blynk.io
2. Login into blynk
3. Click on the templates menu in the left side.
4. Click on + New Template. Create New Template window opens.
5. In the NAME field type DHT11(or any appropriate name)
6. In the HARDWARE select ESP8266.
7. In the CONNECTION TYPE WiFi.
8. Click on Done

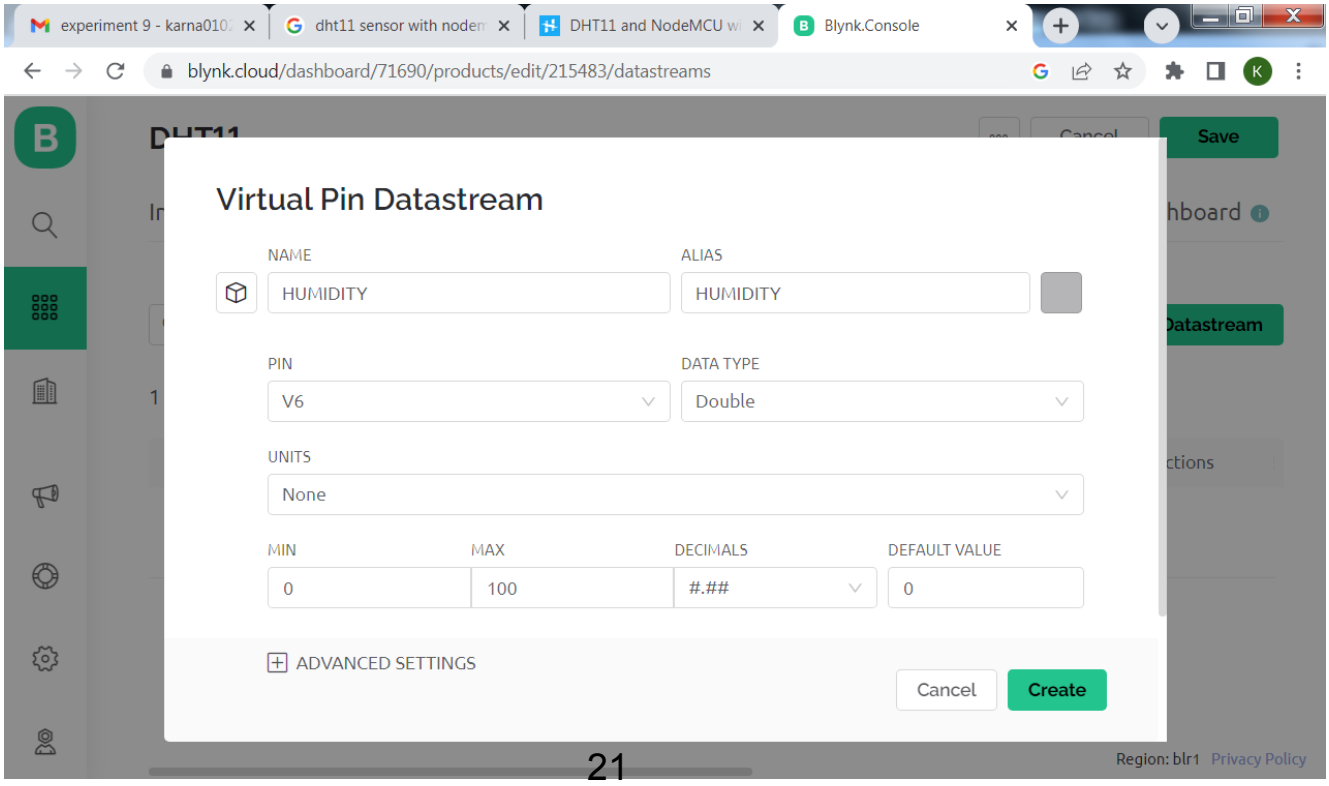


Step 3: Adding Datastreams

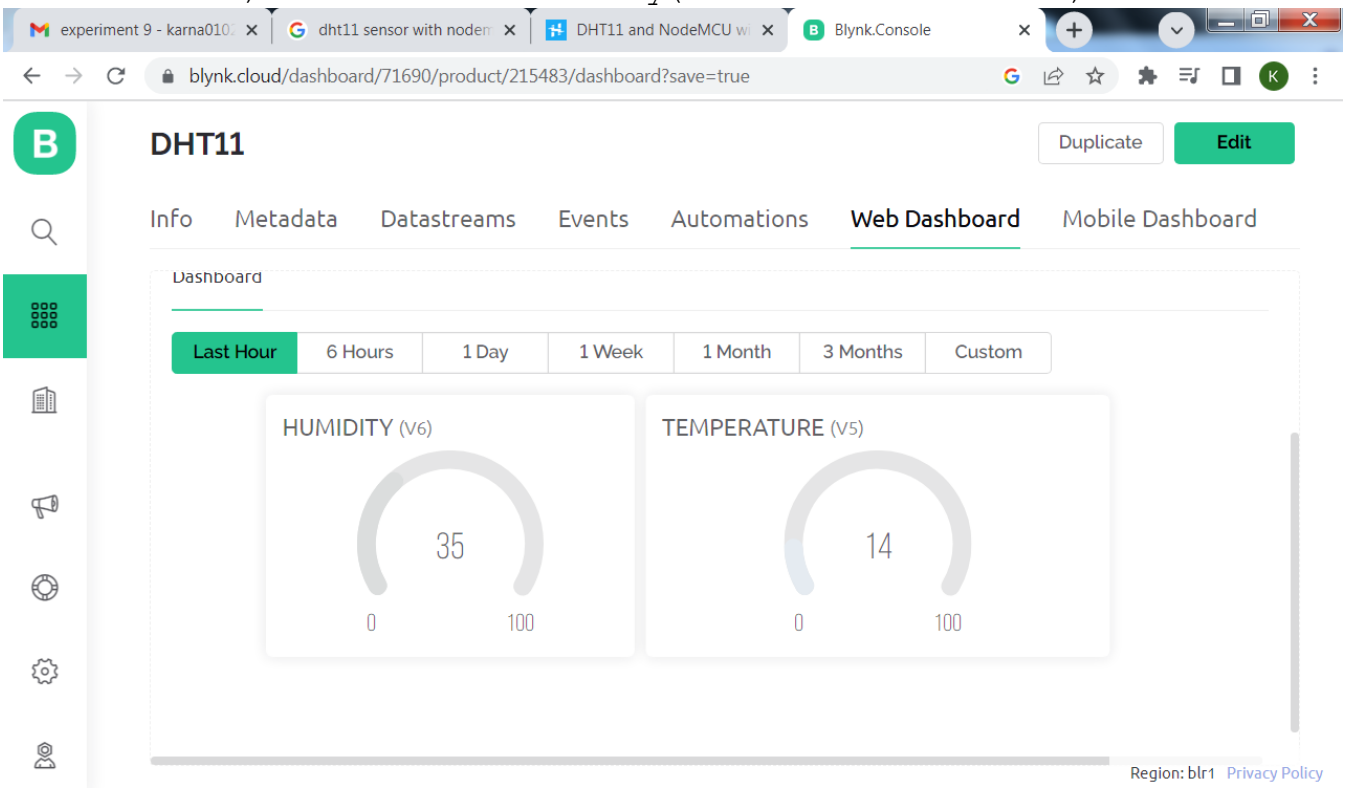
1. In the templates menu open the template DHT11
2. Go to Datastreams
3. Click on +New Datastream, select virtual Pin. Virtual Pin Datastream window opens.
4. Give the name as TEMPERATURE,pin as V5, DATA TYPE as Double, MIN to 0, MAX to 100 DECIMALS as #.## , DEFAULT VALUE to 0. Click on create.



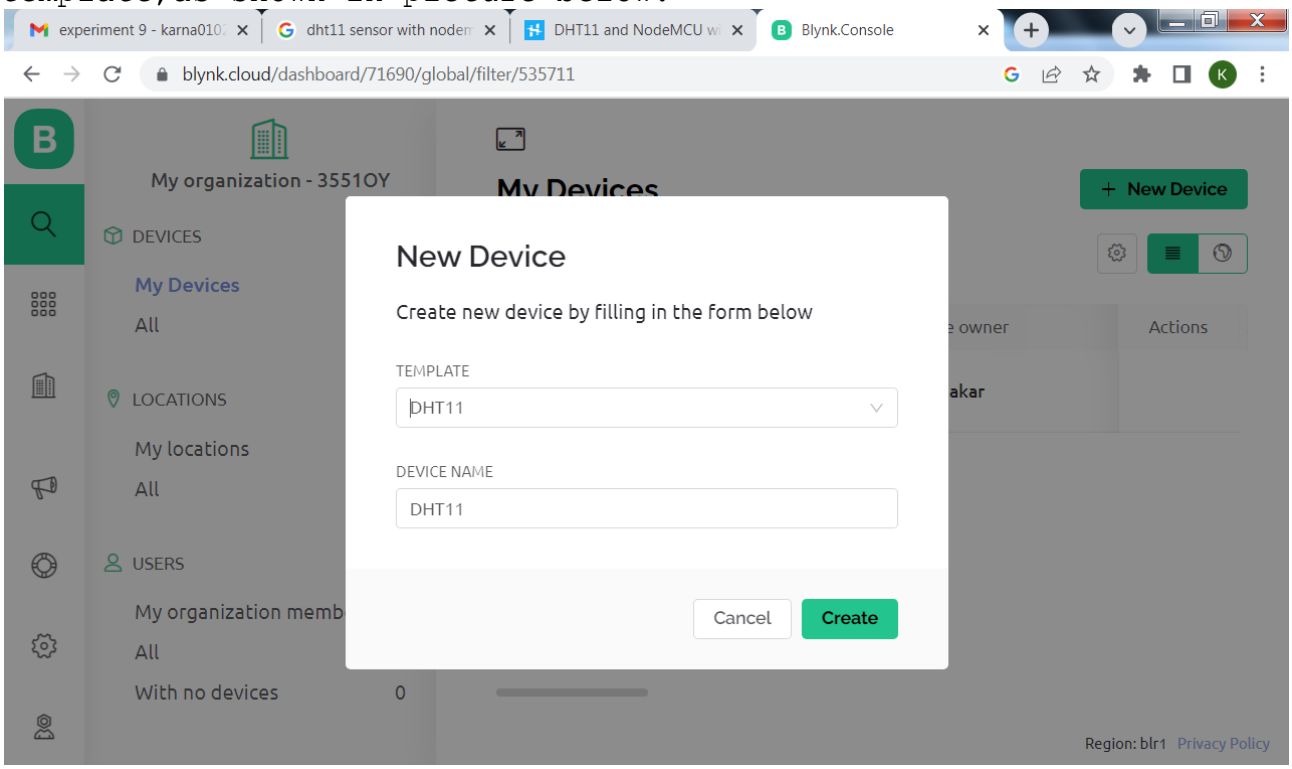
5. Similarly create another datastream for HUMIDITY as shown in picture below.



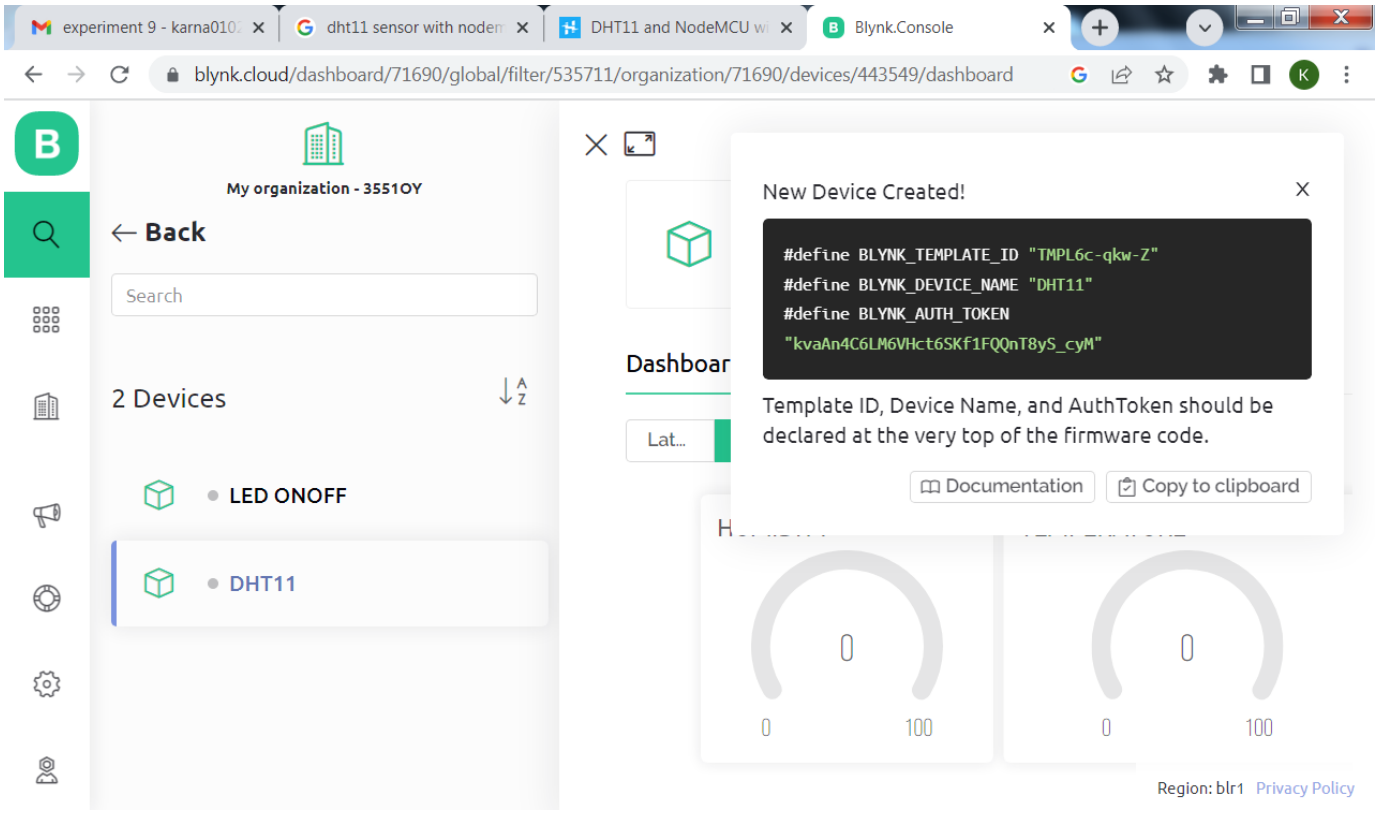
6. Click on web dashboard add two gauges, one for temperature(choose datastream V5) another for humidity(choose datastream V6)



7. Click on search symbol to create new device. Create device selecting the template DHT11 and let the device name be the same name as template,as shown in picture below.



8. Now the device is created with the name DHT11 as shown in picture below.

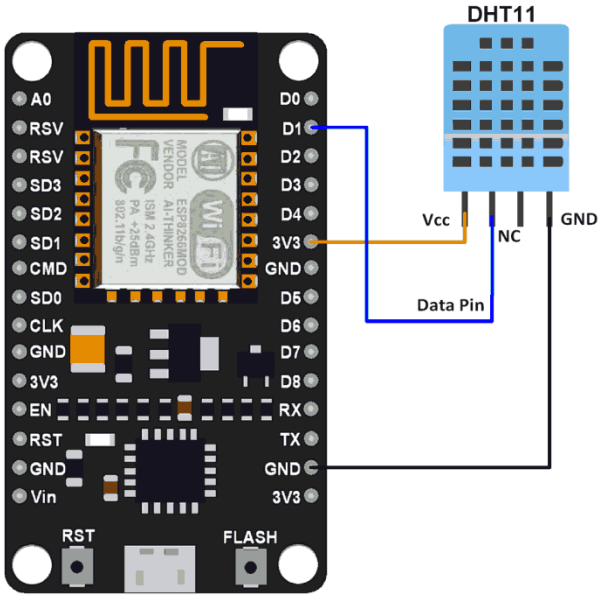


EXP 6: DHT11 sensor

```
#define BLYNK_TEMPLATE_ID "TMPL6c-qkw-Z"
#define BLYNK_DEVICE_NAME "DHT11"
#define BLYNK_AUTH_TOKEN "kvaAn4C6LM6VHct6SKf1FQQnT8yS_cyM"

#include "DHT.h"
#define DHTTYPE DHT11
#define dht_dpin D1
DHT dht(dht_dpin, DHTTYPE);

#define BLYNK_PRINT Serial

#include <ESP8266WiFi.h>
#include <BlynkSimpleEsp8266.h>

BlynkTimer timer;

char auth[] = BLYNK_AUTH_TOKEN;

char ssid[] = "happyfy";
char pass[] = "asdfghjkl";

void sendSensor()
{
  float h = dht.readHumidity();
  float t = dht.readTemperature(); // or dht.readTemperature(true) for
//Fahrenheit
```

```
  Blynk.virtualWrite(V5, t);
  Blynk.virtualWrite(V6, h);
}
void setup()
{
  Serial.begin(9600);
  Blynk.begin(auth, ssid, pass);
  dht.begin();
  // Setup a function to be called every second
  timer.setInterval(1000L, sendSensor);
}

void loop()
{
  Blynk.run();
  timer.run();
}
```

Circuit diagram



Circuit connections

DHT11	NodeMCU
+	5V supply
out	D1
-	GND

Viva Questions

1. **Name some IoT platforms for building IoT applications**

2. **Give examples of Microcontrollers with built in WiFi**

3. **What is Thingspeak platform?**

4. **What is blynk platform?**

5. **What is data stream?**

Result:

DHT11 Sensor data is shown on web dashboard and mobile dashboard.

EXPERIMENT 6
Controlling LEDs, relay & buzzer using Blynk app

AIM:

To Control LEDs, relay & buzzer using Blynk app.

COMPONENTS REQUIRED:

1. Computer with internet connection
2. NodeMCU,
3. Micro USB cable
4. LED
5. Relay
6. Buzzer
7. Resistors (470Ω)
8. Jumper wires

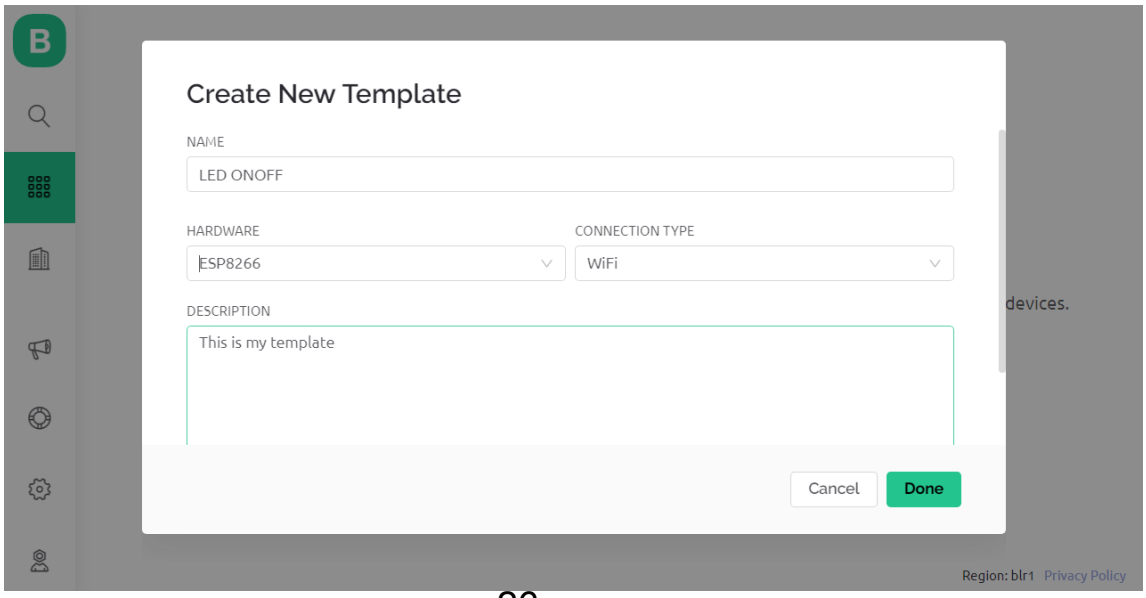
Procedure:

Step1: Creating account

1. Open https://blynk.io
2. Click on LOG IN
3. Click on create new account
4. Provide email in the EMAIL field
5. Click on Sign Up
6. A confirmation is sent to mail.
7. Open the mail and create the password.

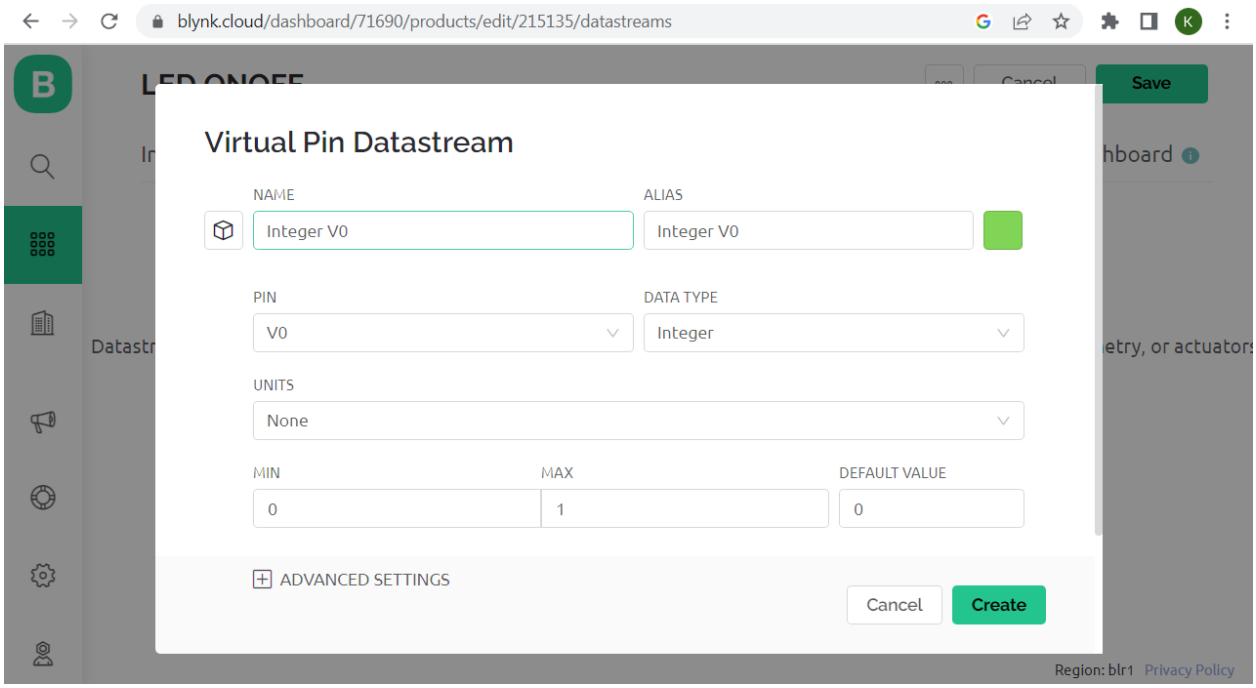
Step 2: Creating template

1. Click on the templates menu in the left side.
2. Click on + New Template. Create New Template window opens.
3. In the NAME field type LED ONOFF(or any appropriate name)
4. In the HARDWARE select ESP8266.
5. In the CONNECTION TYPE WiFi.
6. Click on Done

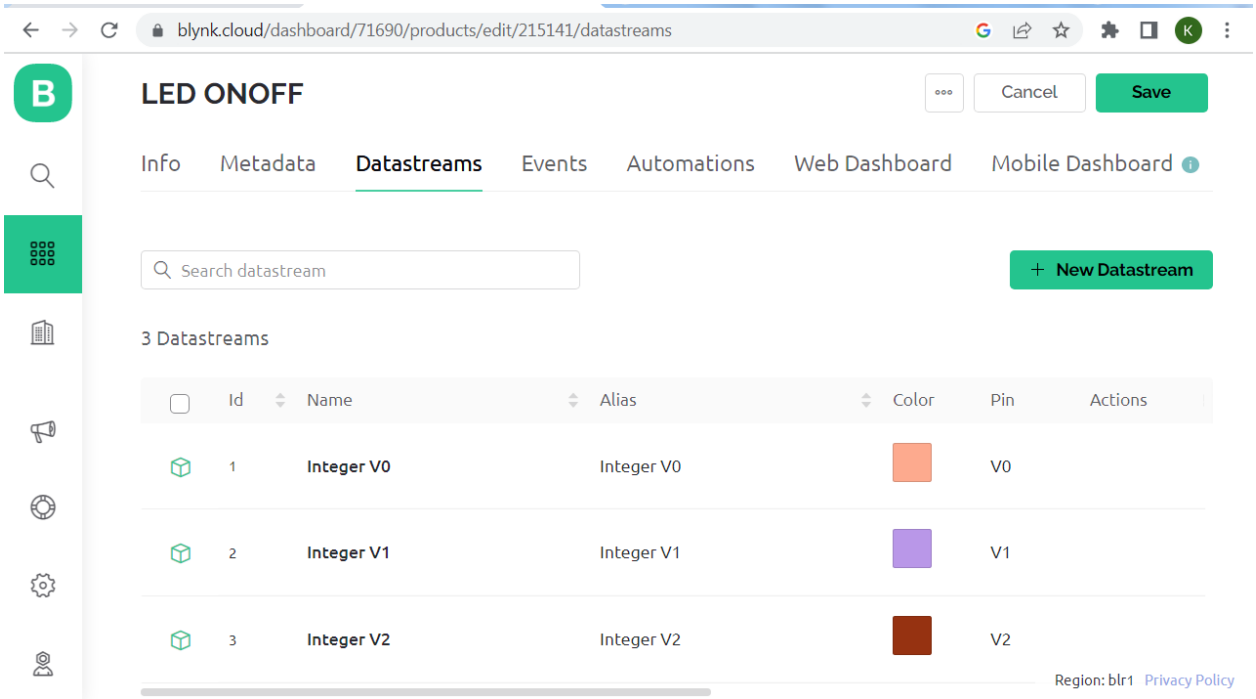


Step 3: Adding Datastreams

1. In the templates menu open the template LED ONOFF
2. Go to Datastreams
3. Click on +New Datastream, select virtual Pin. Virtual Pin Datastream window opens.
4. With all the default values as shown in fig click on Create.

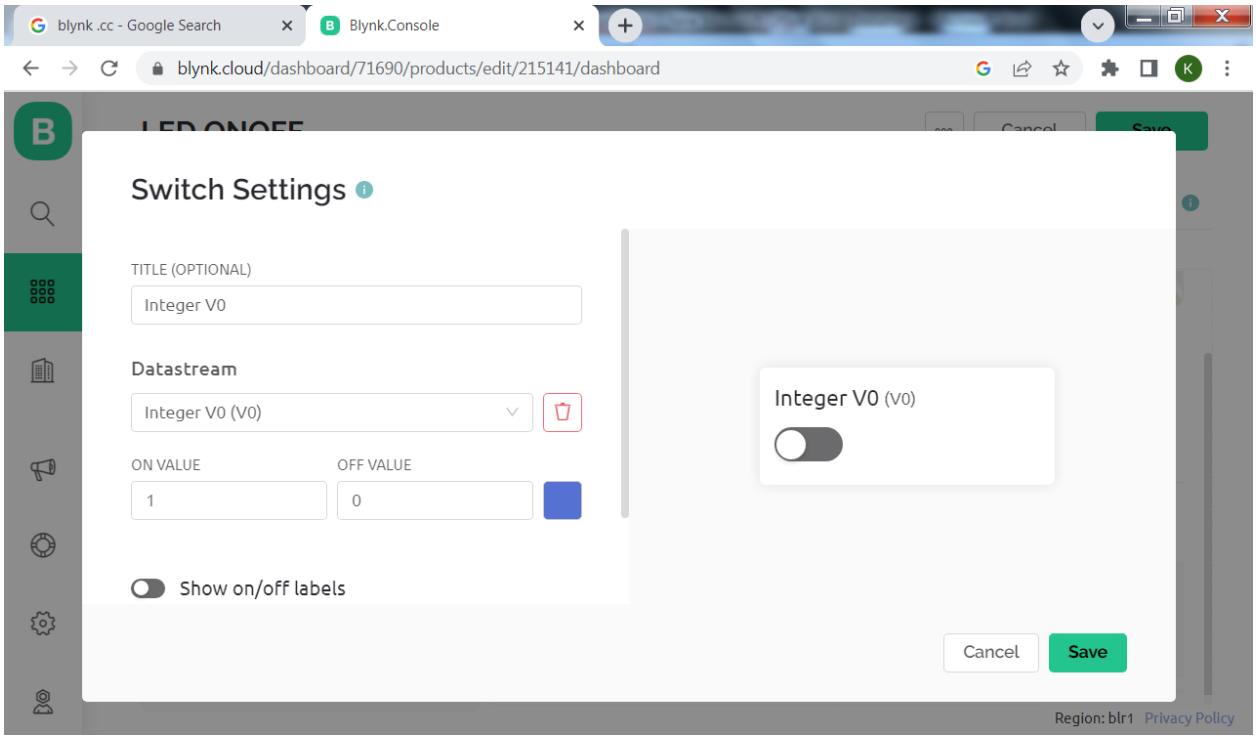


5. Similarly repeat the above step two more times to create totally three datastreams. And click on Save.

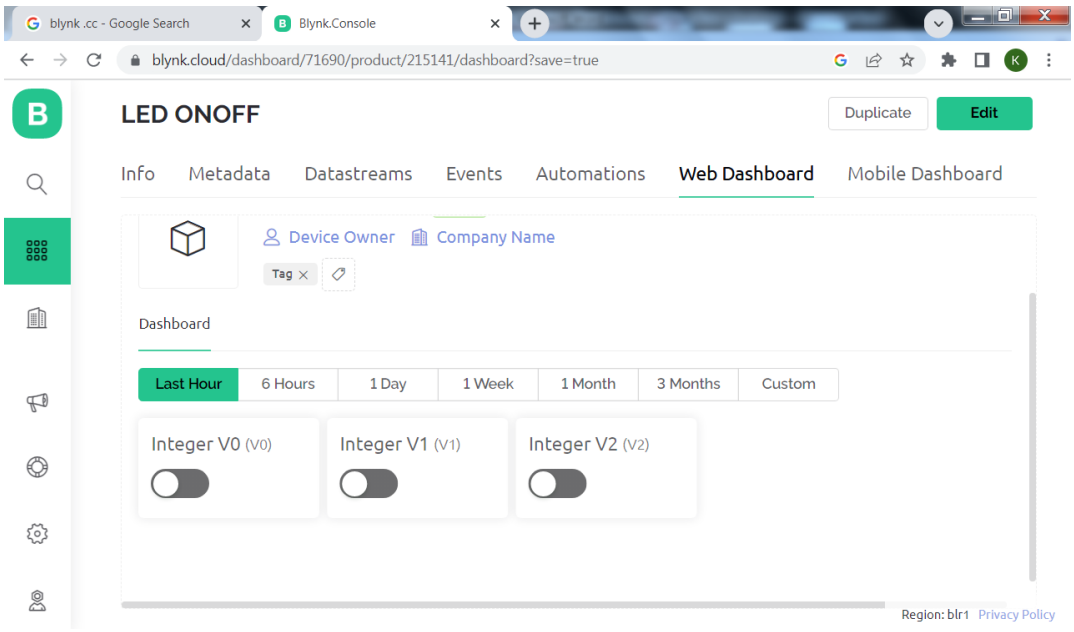


Step 4: Creating Web Dashboard

1. Click on Web Dashboard.
2. In the left we see Widget Box. Double click on Switch to bring it on to the dashboard.
3. Repeat above step two more times to haves three switches in the dashboard
4. Move the mouse cursor to swith, we see gear symbol, click on gear symbol.
5. The Switch Settings window opens. Select Datastream as Integer V0(v0).
6. Turn on the Show on/off labels button.
7. Click on Save.



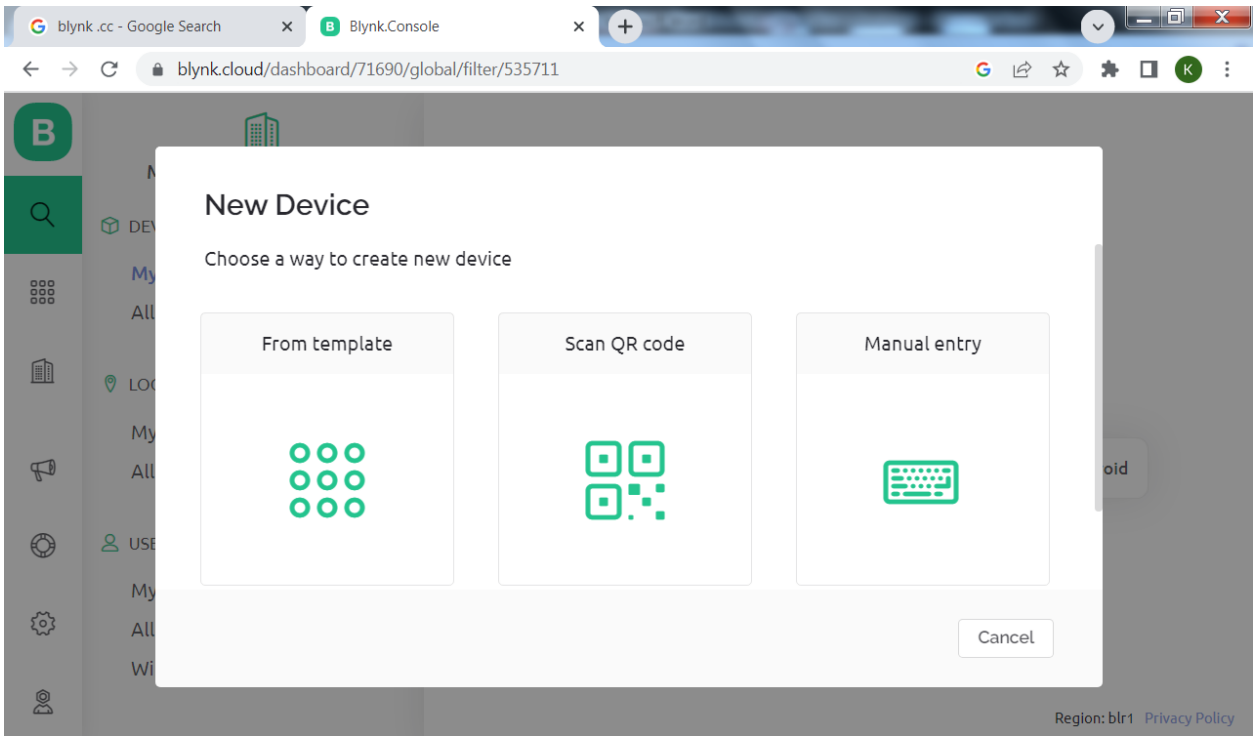
8. Repeat above steps for all the switches except that choose different dataststreams.



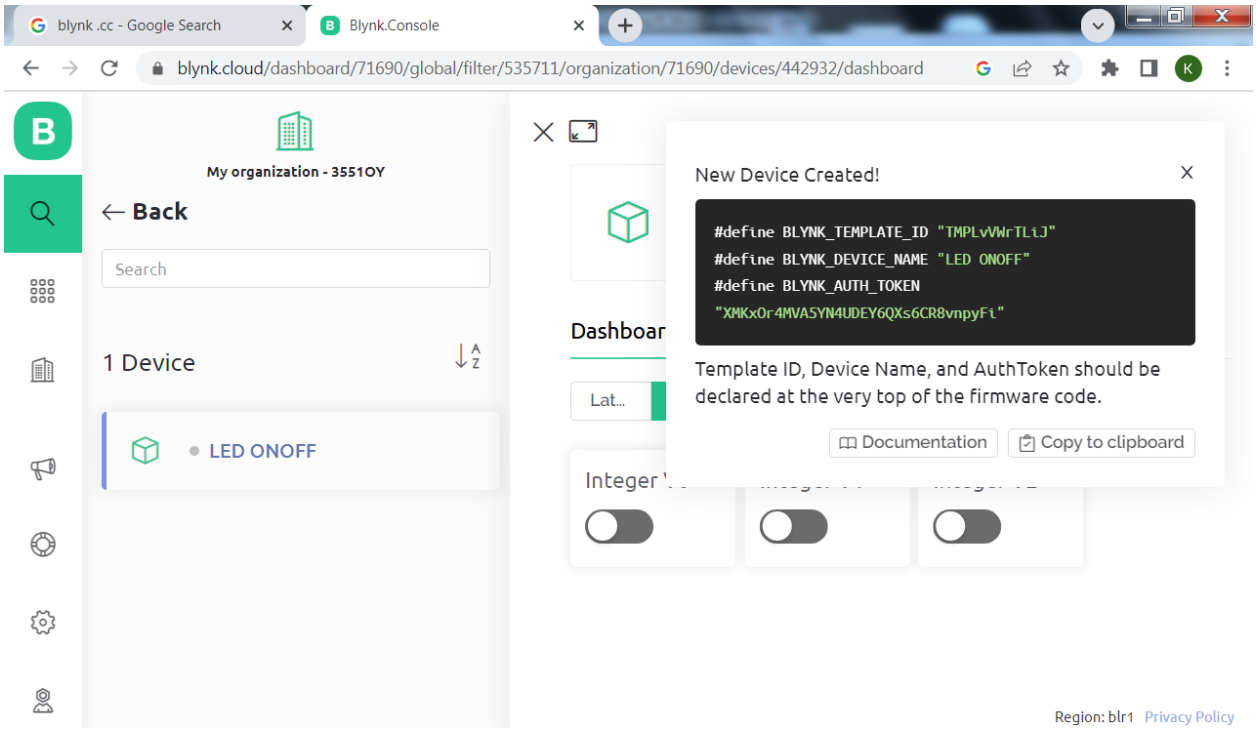
9. Click on Save on top right corner to save the whole web dashboard.

Step 5: Creating Device

1. Click on Search symbol in the left side and click on + New Device
2. The New Device window opens. Create a new device from template by clicking on From template option.



3. New Device window opens. Select the template LED ONOFF and device name LED ONOFF(you can give any name here).
4. Click on Create.
5. In the templates menu open the template LED ONOFF
6. Go to Datastreams
7. Click on +New Datastream, select virtual Pin. Virtual Pin Datastream window opens.
8. With all the default values as shown in fig click on Create.
9. Similarly repeat the above step two more times to create totally three datastreams. And click on Save

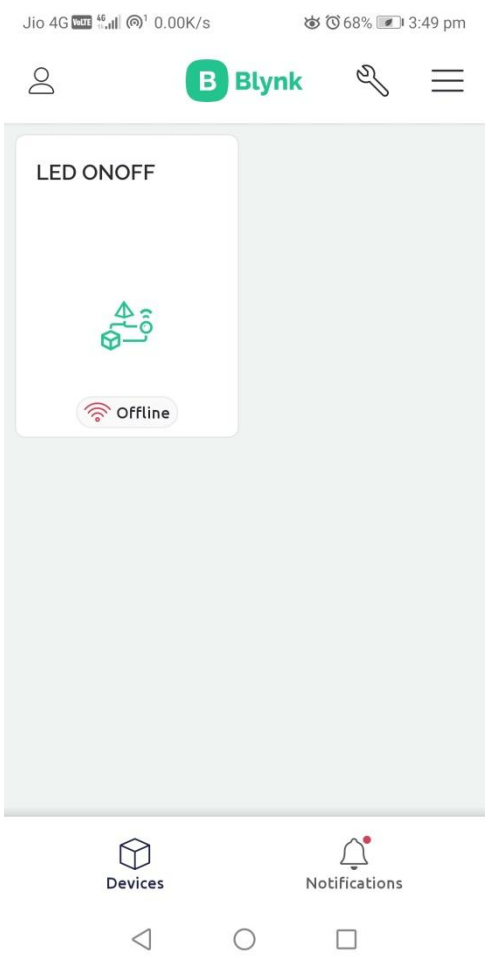


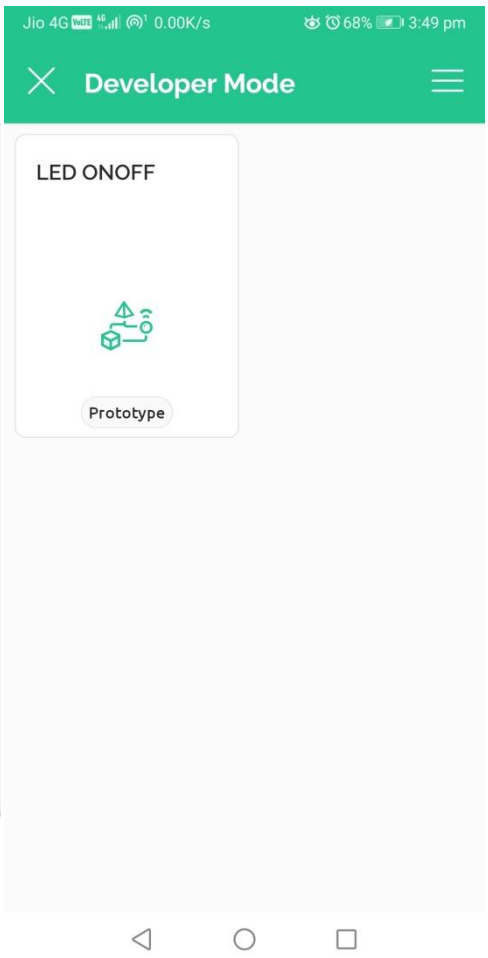
Procedure to set up mobile dashboard

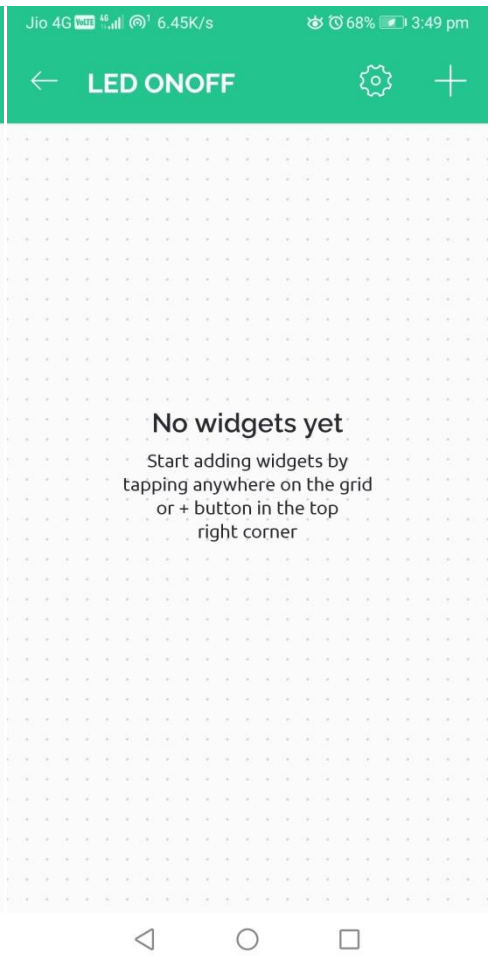
1. Open Blynk IoT app on phone
2. You will see LED ONOFF device on the screen
3. Click on wrench symbol to enter into Developer Mode
4. Tap LED ONOFF device
5. You will see screen showing No widgets yet
6. Click on + on top right to add widgets
7. Now you will see Widget Box showing many widgets
8. Tap on Button widget, that will be added to our dashboard
9. Similarly add two more buttons
10. Tap on first button to change the settings
11. Choose the DATASTREAM as Integer V0(V0)
12. Change the mode to Switch
13. Repeat the same thing for all the buttons
14. Go back to the device screen, now by tapping on the buttons you Will be able to control LEDs connected to D0, D1 and D2.

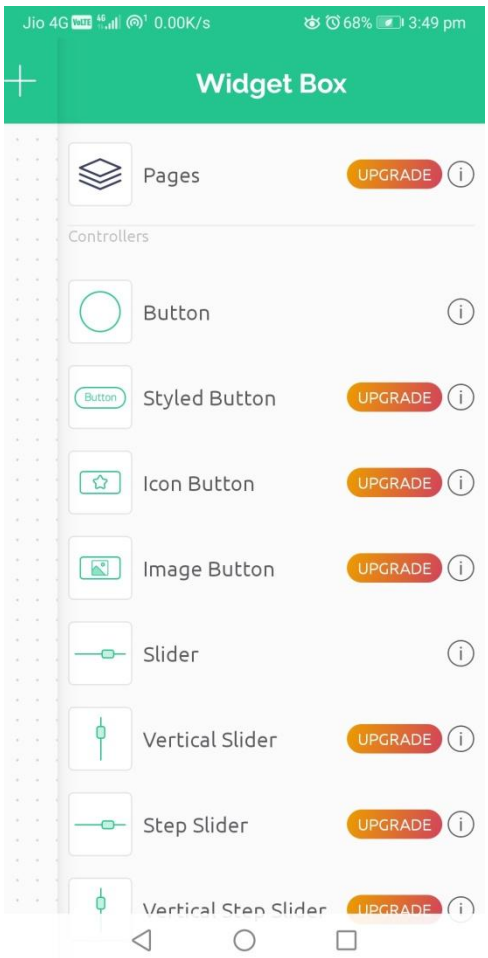
The above steps are shown in the pictures.

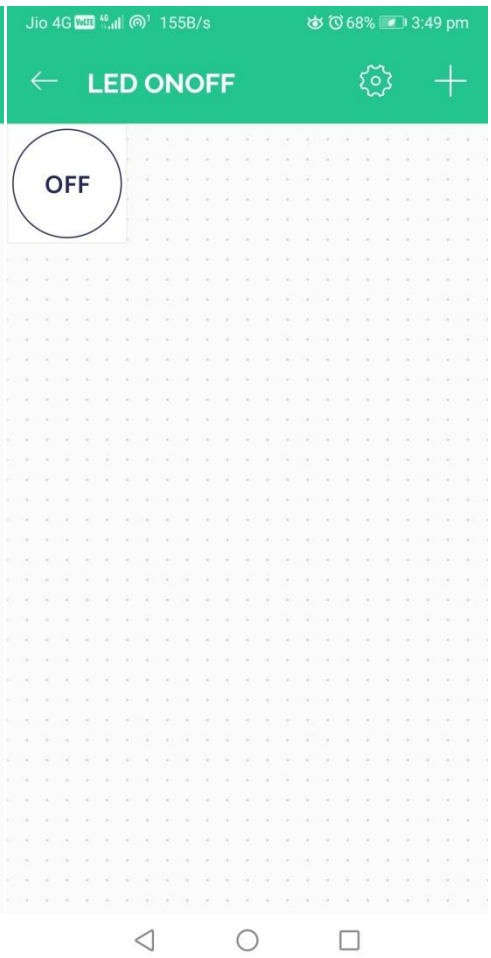
NOTE: TURN ON THE HOTSPOT WHOSE USER NAME AND PASSWORD IS WRITTEN IN THE PROGRAM

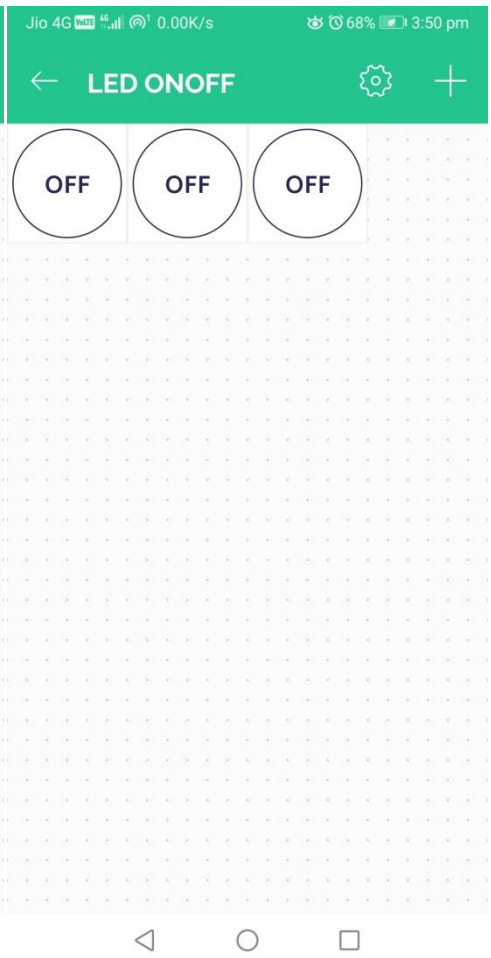

Jio 4G 0.00K/s
68% 3:49 pm
Blynk
LED ONOFF
Offline
Devices
Notifications

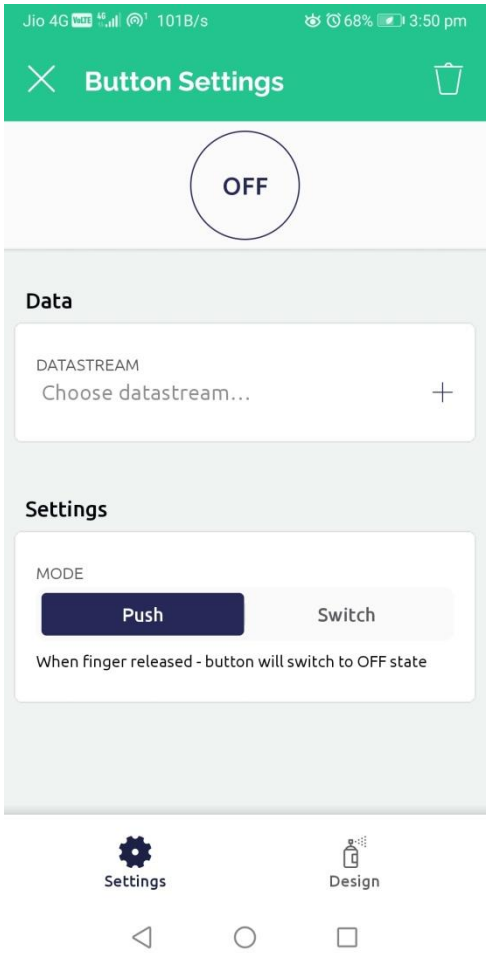

Jio 4G 0.00K/s
68% 3:49 pm
Developer Mode
LED ONOFF
Prototype

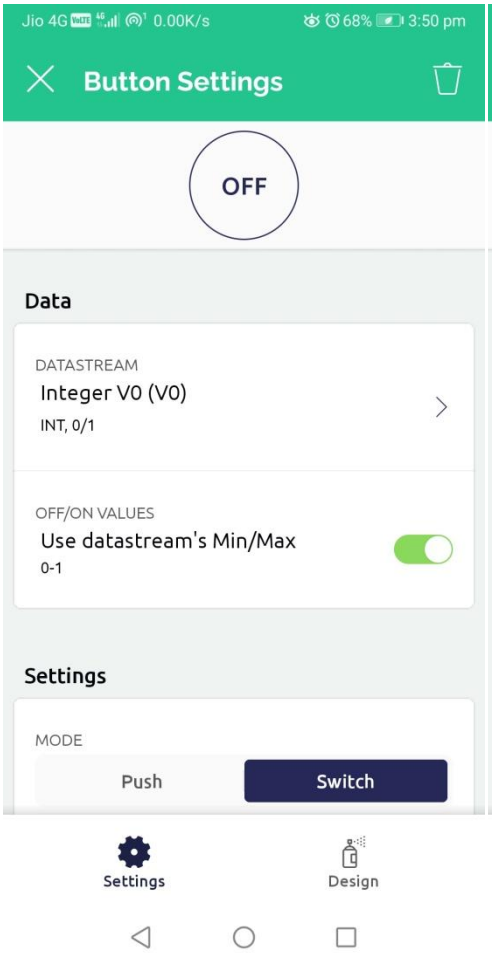

Jio 4G 6.45K/s
68% 3:49 pm
LED ONOFF
No widgets yet
Start adding widgets by tapping anywhere on the grid or + button in the top right corner

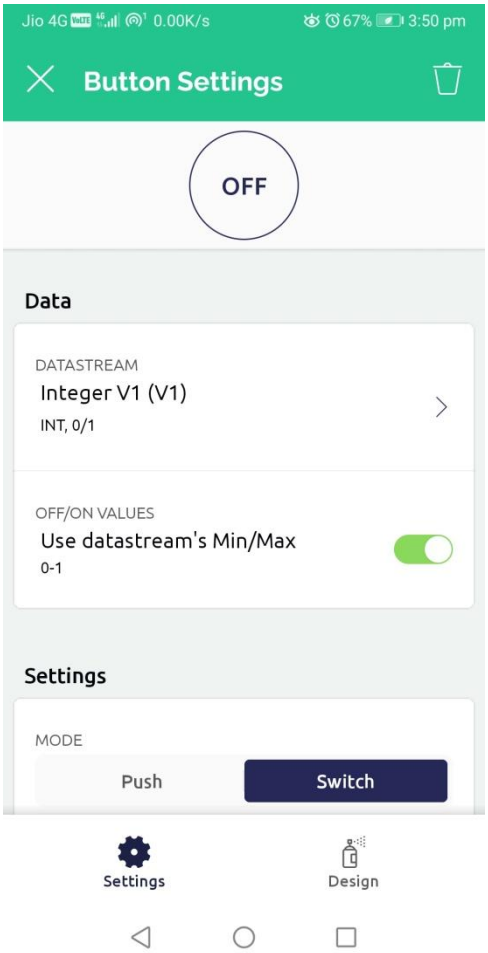

Jio 4G 0.00K/s
68% 3:49 pm
Widget Box
Pages
UPGRADE
Controllers
Button
Styled Button
Icon Button
Image Button
Slider
Vertical Slider
Step Slider

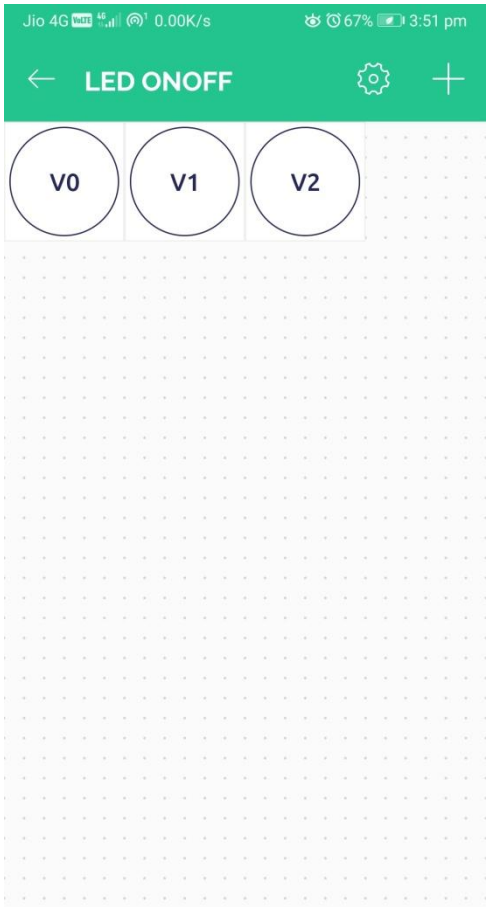

Jio 4G 155B/s
68% 3:49 pm
LED ONOFF
OFF

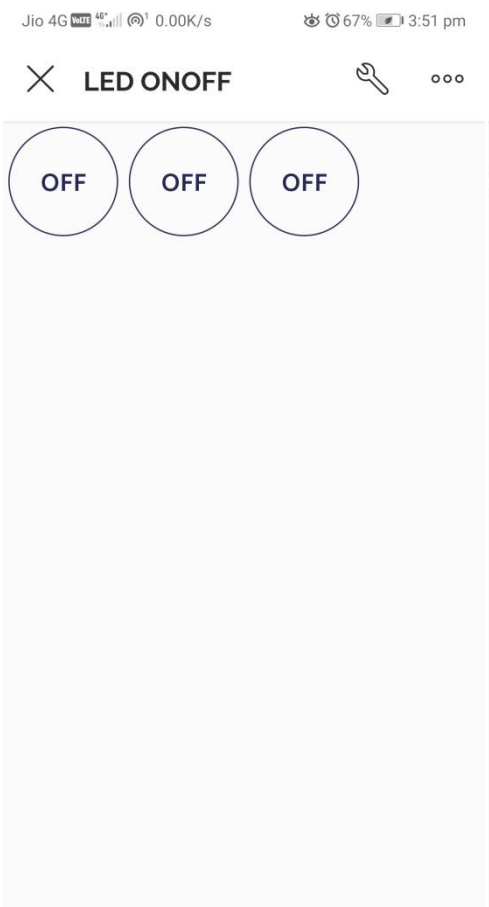

Jio 4G 0.00K/s
68% 3:50 pm
LED ONOFF
OFF
OFF
OFF

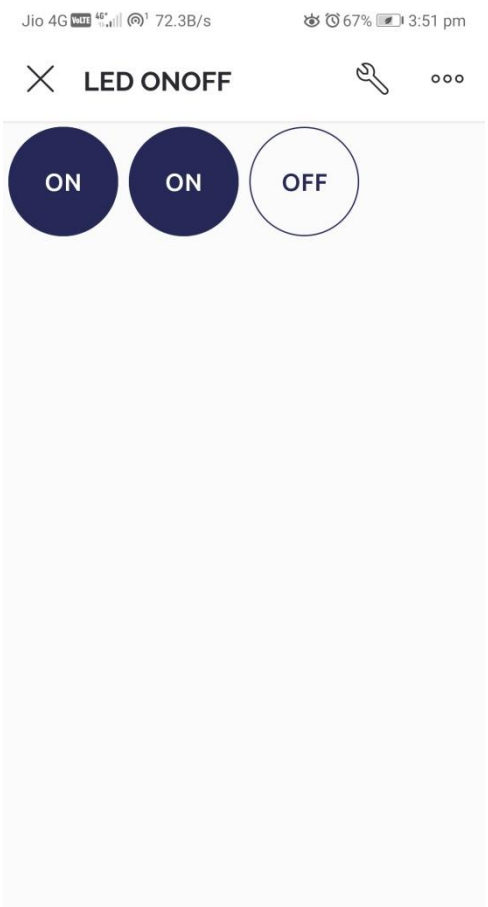

Jio 4G 101B/s 68% 3:50 pm
Button Settings
OFF
Data
DATASTREAM
Choose datastream...
Settings
MODE
Push
Switch
When finger released - button will switch to OFF state
Settings
Design


Jio 4G 0.00K/s 68% 3:50 pm
Button Settings
OFF
Data
DATASTREAM
Integer V0 (V0)
INT, 0/1
OFF/ON VALUES
Use datastream's Min/Max
0-1
Settings
MODE
Push
Switch
Settings
Design


Jio 4G 0.00K/s 67% 3:50 pm
Button Settings
OFF
Data
DATASTREAM
Integer V1 (V1)
INT, 0/1
OFF/ON VALUES
Use datastream's Min/Max
0-1
Settings
MODE
Push
Switch
Settings
Design


Jio 4G 0.00K/s 67% 3:51 pm
LED ONOFF
V0
V1
V2

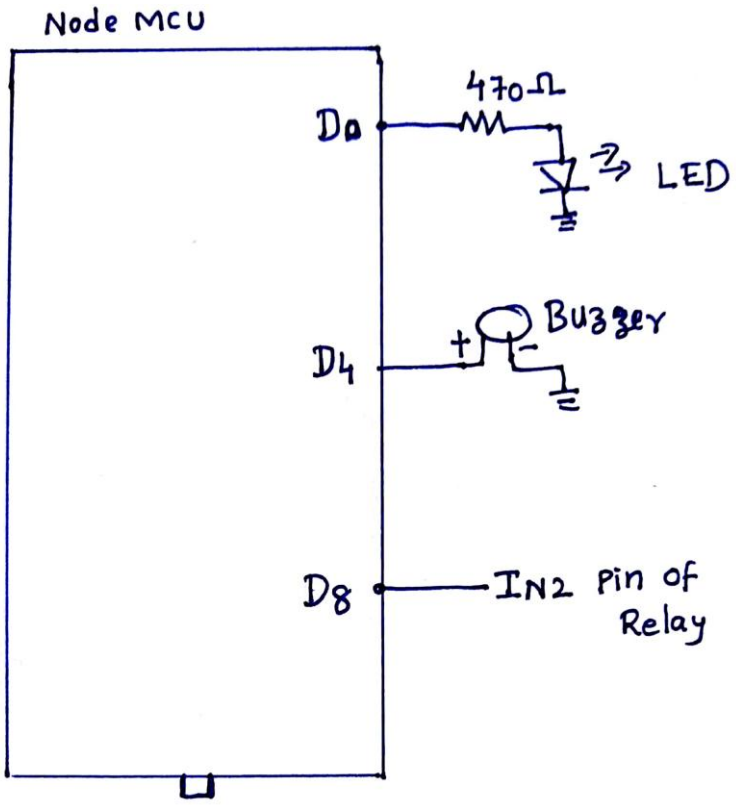

Jio 4G 0.00K/s 67% 3:51 pm
LED ONOFF
OFF
OFF
OFF


Jio 4G 72.3B/s 67% 3:51 pm
LED ONOFF
ON
ON
OFF

EXP 7: Turn on Turn off D0, D4, D8 pins through blynk IoT application

```
#define BLYNK_TEMPLATE_ID "TMPLeSZ6uL3B"
#define BLYNK_DEVICE_NAME "helloworld device"
#define BLYNK_AUTH_TOKEN "hiIaGur_T54_fwCCbvaYEwuprpCc_nqo"
//    Replace    above    three    lines    with    your    BLYNK_TEMPLATE_ID,
//BLYNK_DEVICE_NAME, BLYNK_AUTH_TOKEN

#define BLYNK_PRINT Serial

#include <ESP8266WiFi.h>
#include <BlynkSimpleEsp8266.h>

char auth[] = BLYNK_AUTH_TOKEN;

char ssid[] = "happyfy"; //Replace happyfy with your hotspot name
char pass[] = "asdfghjkl"; // Replace asdfghjkl with your hotspot password

// This function is called every time the Virtual Pin 0 state changes
BLYNK_WRITE(V0)
{
  int value = param.asInt();
  value ? digitalWrite(D0, HIGH) : digitalWrite(D0, LOW);
}



BLYNK_WRITE(V1)
{
   int value = param.asInt();
   value ? digitalWrite(D4, HIGH) : digitalWrite(D4, LOW);
}

BLYNK_WRITE(V2)
{
   int value = param.asInt();
   value ? digitalWrite(D8, HIGH) : digitalWrite(D8, LOW);
}

void setup()
{
  pinMode(D0, OUTPUT);
  pinMode(D4, OUTPUT);
  pinMode(D8, OUTPUT);
  Serial.begin(115200);
  Blynk.begin(auth, ssid, pass);
}

void loop()
{
  Blynk.run();
}
```

Circuit connections



Viva Questions

1. **What is template?**

2. **What is dashboard?**

3. **What are widgets?**

4. **Which pins of nodeMCU have built in LEDs?**

5. **What is the use of Reset pin?**

Result:

LED, relay & buzzer are controlled using Blynk app

EXPERIMENT 7
Controlling LEDs, relay & buzzer from a simple web page

AIM:

To Hosting a basic server from the ESP32 to control various digital based actuators (led, buzzer, relay) from a simple web page.

COMPONENTS REQUIRED:

1. Computer with internet connection
2. NodeMCU,
3. Micro USB cable
4. LED
5. Relay
6. Buzzer
7. Resistors (470Ω)
8. Jumper wires

Procedure:

1. Open Arduino IDE
2. Open new sketch and type the program.
3. Compile the program.
4. Connect the NODEMCU board with USB cable to the computer.
5. Select tools -> select board ->NodeMCU 1.0
6. Select tools -> select port.
7. Upload the program.
8. Connect the circuit as per the circuit diagram.
9. Paste the URL found in Serial monitor (http://192.168.43.249) in either phone or computer browser to control LED, Buzzer, Relay.

EXP 7: Turn on Turn off D0, D4, D8 pins from a simple web page

```
#include <ESP8266WiFi.h>

#define gpio16LEDPin 16 /* One LED connected to GPIO4 - D0 */
#define gpio2LEDPin 2 /* One LED connected to GPIO5 - D4 */
#define gpio15LEDPin 15 /* One LED connected to GPIO5 - D8 */

const char* ssid = "happyfy"; /* Add your router's SSID */
const char* password = "asdfghjkl"; /*Add the password */

int gpio16Value;
int gpio2Value;
int gpio15Value;
WiFiServer espServer(80); /* Instance of WiFiServer with port number 80 */
/* 80 is the Port Number for HTTP Web Server */

void setup()
{
```

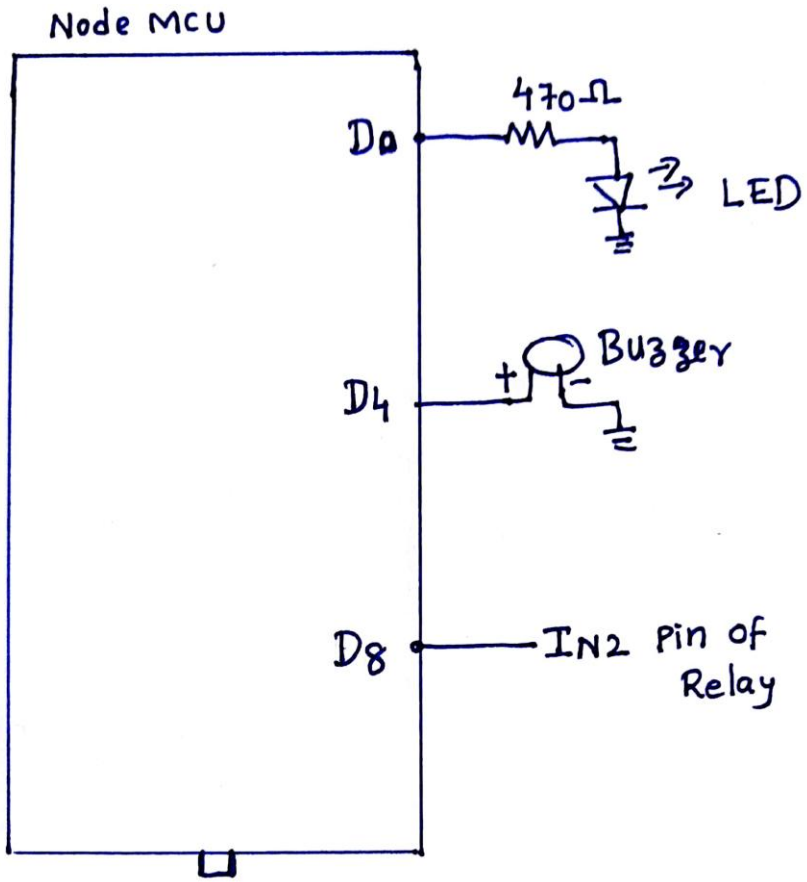
```
  Serial.begin(115200); /* Begin Serial Communication with 115200 Baud Rate
*/
  /* Configure GPIO4 and GPIO5 Pins as OUTPUTs */
  pinMode(gpio16LEDPin, OUTPUT);
  pinMode(gpio2LEDPin, OUTPUT);
  pinMode(gpio15LEDPin, OUTPUT);
  /* Set the initial values of GPIO4 and GPIO5 as LOW*/
  /* Both the LEDs are initially OFF */
  digitalWrite(gpio16LEDPin, LOW);
  digitalWrite(gpio2LEDPin, LOW);
  digitalWrite(gpio15LEDPin, LOW);

  Serial.print("\n");
  Serial.print("Connecting to: ");
  Serial.println(ssid);
  WiFi.mode(WIFI_STA); /* Configure ESP8266 in STA Mode */
  WiFi.begin(ssid, password); /* Connect to Wi-Fi based on above SSID and
Password */
  while(WiFi.status() != WL_CONNECTED)
  {
    Serial.print("*");
    delay(500);
  }
  Serial.print("\n");
  Serial.print("Connected to Wi-Fi: ");
  Serial.println(WiFi.SSID());
  delay(100);

  Serial.print("\n");
  Serial.println("Starting ESP8266 Web Server...");
  espServer.begin(); /* Start the HTTP web Server */
  Serial.println("ESP8266 Web Server Started");
  Serial.print("\n");
  Serial.print("The URL of ESP8266 Web Server is: ");
  Serial.print("http://");
  Serial.println(WiFi.localIP());
  Serial.print("\n");
  Serial.println("Use the above URL in your Browser to access ESP8266 Web
Server\n");
}

void loop()
{
  WiFiClient client = espServer.available(); /* Check if a client is
available */
  if(!client)
  {
    return;
  }

  Serial.println("New Client!!!");

  String request = client.readStringUntil('\r'); /* Read the first line of
the request from client */
  Serial.println(request); /* Print the request on the Serial monitor */
```

```
/* The request is in the form of HTTP GET Method */
client.flush();

/* Extract the URL of the request */
/* We have four URLs. If IP Address is 192.168.1.6 (for example),
 * then URLs are:
 * 192.168.1.6/GPIO4ON and its request is GET /GPIO4ON HTTP/1.1
 * 192.168.1.6/GPIO4OFF and its request is GET /GPIO4OFF HTTP/1.1
 * 192.168.1.6/GPIO5ON and its request is GET /GPIO5ON HTTP/1.1
 * 192.168.1.6/GPIO4OFF and its request is GET /GPIO5OFF HTTP/1.1
 */
 /* Based on the URL from the request, turn the LEDs ON or OFF */
if (request.indexOf("/GPIO16ON") != -1)
{
  Serial.println("GPIO16 LED is ON");
  digitalWrite(gpio16LEDPin, HIGH);
  gpio16Value = HIGH;
}
if (request.indexOf("/GPIO16OFF") != -1)
{
  Serial.println("GPIO16 LED is OFF");
  digitalWrite(gpio16LEDPin, LOW);
  gpio16Value = LOW;
}
if (request.indexOf("/GPIO2ON") != -1)
{
  Serial.println("GPIO2 LED is ON");
  digitalWrite(gpio2LEDPin, HIGH);
  gpio2Value = HIGH;
}
if (request.indexOf("/GPIO2OFF") != -1)
{
  Serial.println("GPIO2 LED is OFF");
  digitalWrite(gpio2LEDPin, LOW);
  gpio2Value = LOW;
}


if (request.indexOf("/GPIO15ON") != -1)
{
  Serial.println("GPIO15 LED is ON");
  digitalWrite(gpio15LEDPin, HIGH);
  gpio15Value = HIGH;
}
if (request.indexOf("/GPIO15OFF") != -1)
{
  Serial.println("GPIO15 LED is OFF");
  digitalWrite(gpio15LEDPin, LOW);
  gpio15Value = LOW;
}



/* HTTP Response in the form of HTML Web Page */
client.println("HTTP/1.1 200 OK");
```

```
  client.println("Content-Type: text/html");
  client.println(); //  IMPORTANT
  client.println("<!DOCTYPE HTML>");
  client.println("<html>");
  client.println("<head>");
  client.println("<meta name=\"viewport\" content=\"width=device-width,
initial-scale=1\">");
  client.println("<link rel=\"icon\" href=\"data:,\">");
  /* CSS Styling for Buttons and Web Page */
  client.println("<style>");
  client.println("html { font-family: Courier New; display: inline-block;
margin: 0px auto; text-align: center;}");
  client.println(".button {border: none; color: white; padding: 10px 20px;
text-align: center;");
  client.println("text-decoration: none; font-size: 25px; margin: 2px;
cursor: pointer;}");
  client.println(".button1 {background-color: #13B3F0;}");
  client.println(".button2 {background-color: #3342FF;}");
  client.println("</style>");
  client.println("</head>");

  /* The main body of the Web Page */
  client.println("<body>");
  client.println("<h2>ESP8266 Web Server</h2>");

  if(gpio16Value == LOW)
  {
    client.println("<p>GPIO16 LED Status: OFF</p>");
    client.print("<p><a href=\"/GPIO16ON\"><button class=\"button
button1\">Click to turn ON</button></a></p>");
  }
  else
  {
    client.println("<p>GPIO16 LED Status: ON</p>");
    client.print("<p><a href=\"/GPIO16OFF\"><button class=\"button
button2\">Click to turn OFF</button></a></p>");
  }

  if(gpio2Value == LOW)
  {
    client.println("<p>GPIO2 LED Status: OFF</p>");
    client.print("<p><a href=\"/GPIO2ON\"><button class=\"button
button1\">Click to turn ON</button></a></p>");
  }
  else
  {
    client.println("<p>GPIO2 LED Status: ON</p>");
    client.print("<p><a href=\"/GPIO2OFF\"><button class=\"button
button2\">Click to turn OFF</button></a></p>");
  }

  if(gpio15Value == LOW)
  {
    client.println("<p>GPIO15 LED Status: OFF</p>");
```

```
    client.print("<p><a href=\"/GPIO15ON\"><button class=\"button
button1\">Click to turn ON</button></a></p>");
  }
  else
  {
    client.println("<p>GPIO15 LED Status: ON</p>");
    client.print("<p><a href=\"/GPIO15OFF\"><button class=\"button
button2\">Click to turn OFF</button></a></p>");
  }
  client.println("</body>");
  client.println("</html>");
  client.print("\n");

  delay(1);
  /* Close the connection */
  client.stop();
  Serial.println("Client disconnected");
  Serial.print("\n");
}
```

Circuit connections



Viva Questions

1. **What is web server**

2. **What is simple webpage**

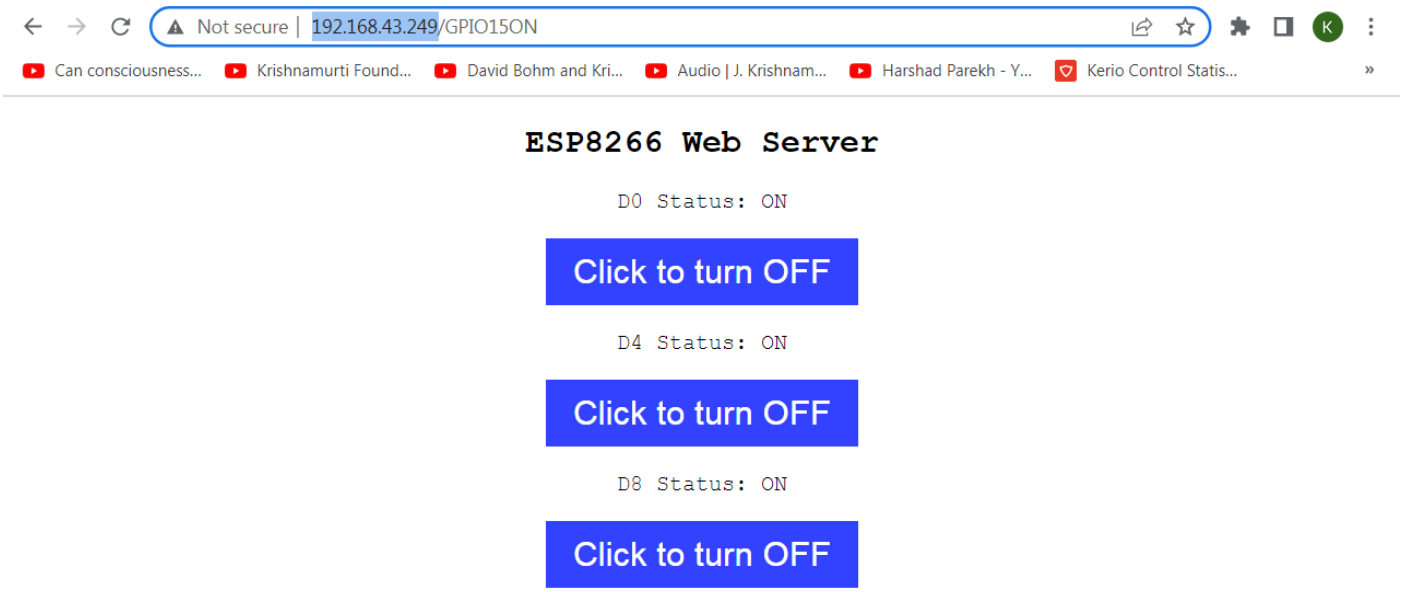
3. **How many GPIO pins are present in nodeMCU, name them?**

4. **What are the pins present in UART communications?**

5. **What is load ?**

Result:

Screenshot of web page can be seen below:



LED, relay & buzzer are controlled from a simple web page.

EXPERIMENT 8

Displaying various sensor readings on a simple web page hosted on the ESP32

AIM:

To write a arduino program that displays DHT11 sensor data on a simple web page hosted on the ESP32

COMPONENTS REQUIRED:

1. Computer
2. NodeMCU,
3. Micro USB cable
4. DHT11 sensor
5. Jumper wires

Procedure:

1. Open Arduino IDE
2. Open new sketch and type the program.
3. Compile the program.
4. Connect the NODEMCU board with USB cable to the computer.
5. Select tools -> select board ->NodeMCU 1.0
6. Select tools -> select port.
7. Upload the program.
8. Connect the circuit as per the circuit diagram.
9. Paste the URL found in Serial monitor (http://192.168.43.249) in either phone or computer browser to see DHT11 sensor values.

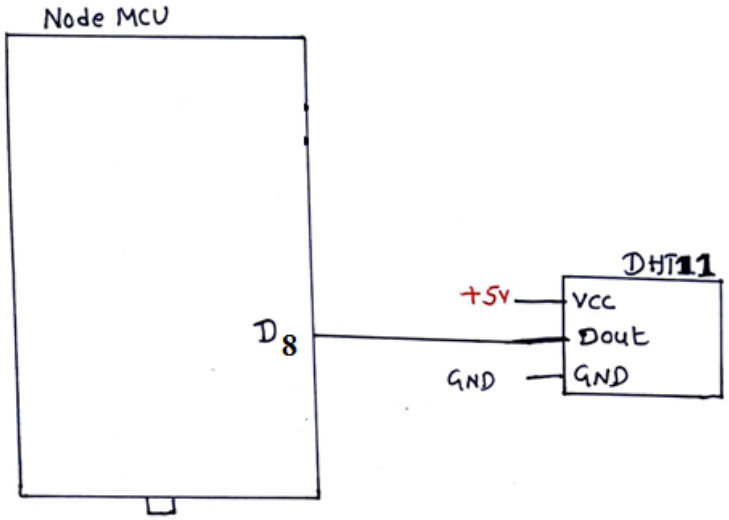
PROGRAM

```
#include <ESP8266WiFi.h>
#include <ESP8266WebServer.h>
#include "DHT.h"

// Uncomment one of the lines below for whatever DHT sensor type you're
using!
#define DHTTYPE DHT11   // DHT 11
//#define DHTTYPE DHT21   // DHT 21 (AM2301)
//#define DHTTYPE DHT22   // DHT 22  (AM2302), AM2321

/*Put your SSID & Password*/
const char* ssid = "happyfy";  // Enter SSID here
const char* password = "asdfghjkl";  //Enter Password here

ESP8266WebServer server(80);

// DHT Sensor
uint8_t DHTPin = D8;

// Initialize DHT sensor.
DHT dht(DHTPin, DHTTYPE);
```

```
float Temperature;
float Humidity;

void setup() {
  Serial.begin(115200);
  delay(100);

  pinMode(DHTPin, INPUT);

  dht.begin();

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

  //connect to your local wi-fi network
  WiFi.begin(ssid, password);

  //check wi-fi is connected to wi-fi network
  while (WiFi.status() != WL_CONNECTED) {
  delay(1000);
  Serial.print(".");
  }
  Serial.println("");
  Serial.println("WiFi connected..!");
  Serial.print("Got IP: ");  Serial.println(WiFi.localIP());

  server.on("/", handle_OnConnect);
  server.onNotFound(handle_NotFound);

  server.begin();
  Serial.println("HTTP server started");

}
void loop() {

  server.handleClient();

}

void handle_OnConnect() {

 Temperature = dht.readTemperature(); // Gets the values of the temperature
  Humidity = dht.readHumidity(); // Gets the values of the humidity
  server.send(200, "text/html", SendHTML(Temperature,Humidity));
}

void handle_NotFound(){
  server.send(404, "text/plain", "Not found");
}

String SendHTML(float Temperaturestat,float Humiditystat){
  String ptr = "<!DOCTYPE html> <html>\n";
  ptr +="<head><meta name=\"viewport\" content=\"width=device-width,
initial-scale=1.0, user-scalable=no\">\n";
```

```
  ptr +="<title>ESP8266 Weather Report</title>\n";
  ptr +="<style>html { font-family: Helvetica; display: inline-block;
margin: 0px auto; text-align: center;}\n";
  ptr +="body{margin-top: 50px;} h1 {color: #444444;margin: 50px auto
30px;}\n";
  ptr +="p {font-size: 24px;color: #444444;margin-bottom: 10px;}\n";
  ptr +="</style>\n";
  ptr +="</head>\n";
  ptr +="<body>\n";
  ptr +="<div id=\"webpage\">\n";
  ptr +="<h1>ESP8266 NodeMCU Weather Report</h1>\n";

  ptr +="<p>Temperature: ";
  ptr +=(int)Temperaturestat;
  ptr +="°C</p>";
  ptr +="<p>Humidity: ";
  ptr +=(int)Humiditystat;
  ptr +="%</p>";

  ptr +="</div>\n";
  ptr +="</body>\n";
  ptr +="</html>\n";
  return ptr;
}
```

Circuit connections



Viva Questions

1. **What is DHT11 sensor?**

2. **What is DHT22 sensor?**

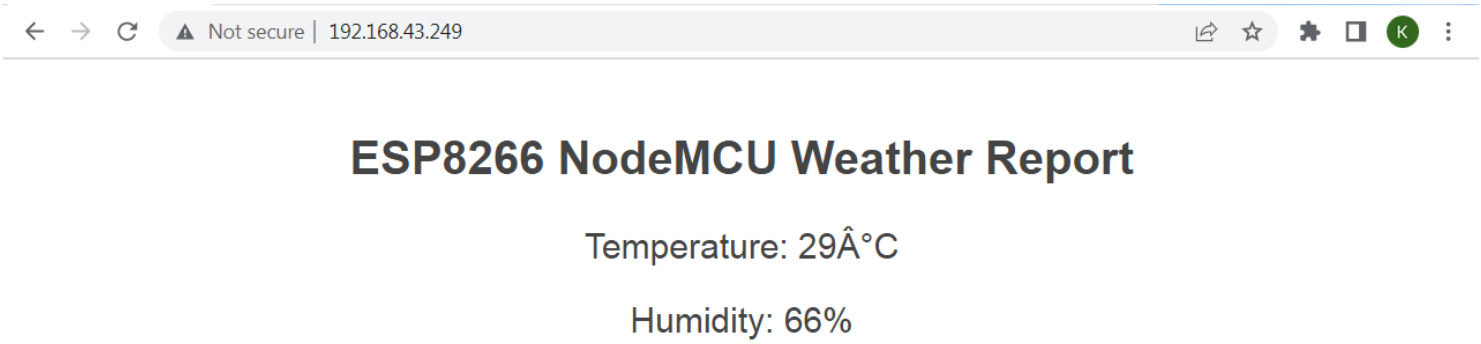
3. **What is arduino board?**

4. **What is arduino mega board ?**

5. **What is ESP32?**

Result:

Screenshot of web page can be seen below:



Temperature and Humidity are displayed on the web pages hosted by ESP32.

EXPERIMENT 9
Controlling LEDs/Motors/Relays from an Android app

AIM:

To control LED/Motor/Relay from Android app.

COMPONENTS REQUIRED:

1. NodeMCU
2. Micro USB cable
3. Relay module
4. Jumper wires

Procedure:

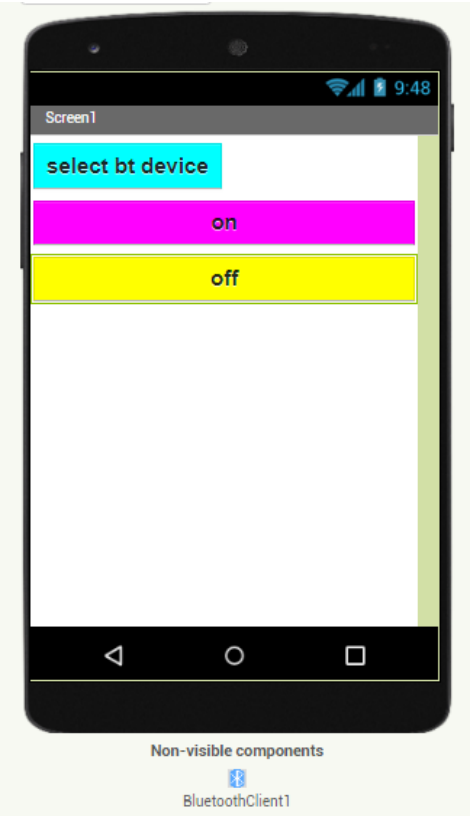
1. Open MIT APPINVNTOR
2. Go to My projects -> start new project -> give project name.
3. Add list picker, two buttons and Bluetooth client to the designer section as shown in the picture below.
4. You may change the properties of the above user interface controls in properties section.
5. Go to blocks section and add blocks as shown in the picture below.
6. Go to build and click on Android Apk(.apk)
7. Download MIT AI2 companion app from playstore(if it is not there in your phone).
8. Download the app just designed by using scan QR code.
9. Open Arduino IDE
10. Open new sketch and type the program.
11. Compile the program.
12. Connect the NODEMCU board with USB cable to the computer.
13. Select tools -> select board ->NodeMCU 1.0
14. Select tools -> select port.
15. Upload the program.
16. Connect the circuit as per the circuit diagram.
17. Open Bluetooth app "Bluetooth Terminal HC-05"
18. Connect to HC-05.
19. Tap on off buttons respectively to turn on turn off light.

Code:

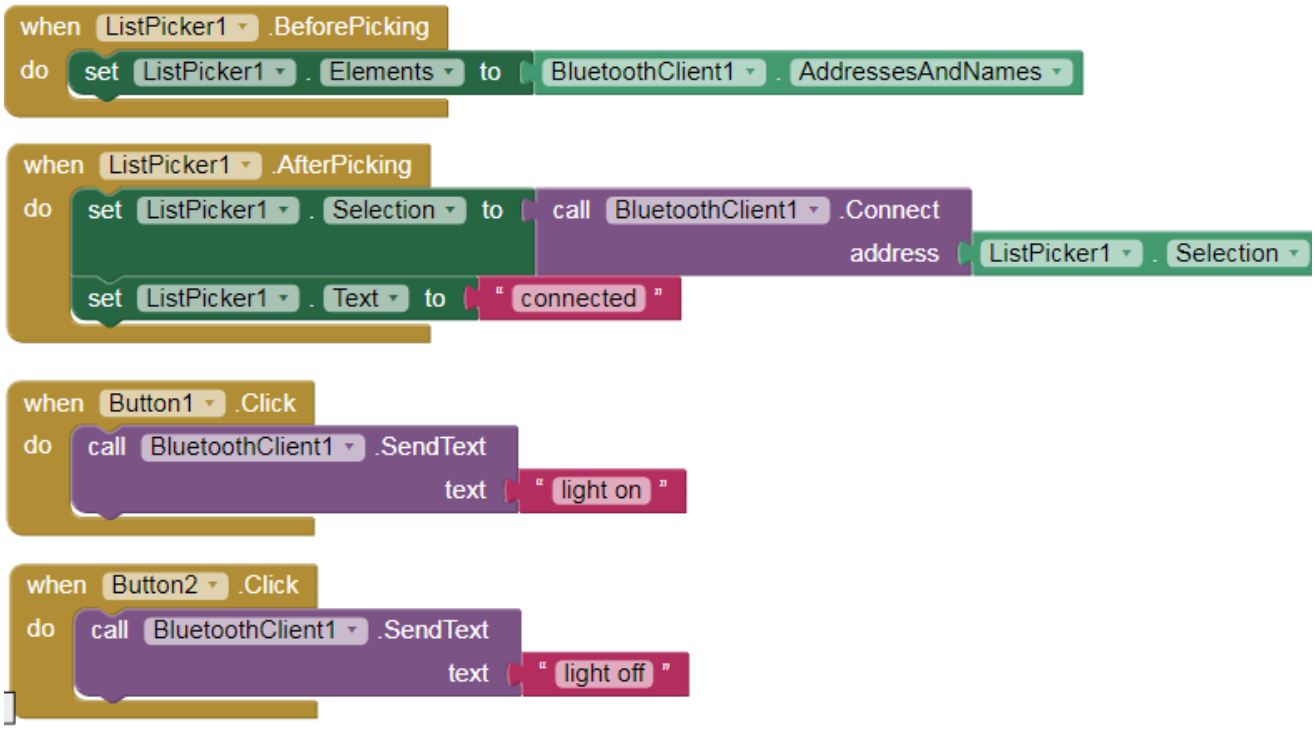
```
String voice = "";
void setup()
{
  Serial.begin(9600);
  pinMode(D1,OUTPUT);
}
void loop()
{
  while(Serial.available())
  {
    char c = Serial.read();
    voice = voice + c;
  }
 Serial.println(voice);
```

```
 if(voice == "light on")
 {
  digitalWrite(D1,LOW);// Relay module needs a low to TURN ON
 }
 if(voice == "light off")
 {
  digitalWrite(D1,HIGH);// Relay module needs a HIGH to TURN OFF
 }
 delay(1000);
}
```

Designer



Blocks



Circuit connections

Relay Module	NodeMCU
VCC	+5V(External Supply)
IN1	D1
IN2	-
GND	GND
NO1-COM1	Connect Load

Viva Questions

1. **What is MITappinventor?**
2. **What is HC05 module?**
3. **What are the pins used in SPI communication?**
4. **What are the pins used in I2C communication?**
5. **What is soil moisture sensor?**

Result:

Relay is controlled using Android app created using MIT appinventor.

EXPERIMENT 10

Displaying humidity and temperature data on a web-based application

AIM:

To write a arduino program that displays DHT11 sensor data on a simple web page hosted on the ESP32

COMPONENTS REQUIRED:

1. Computer
2. NodeMCU,
3. Micro USB cable
4. DHT11 sensor
5. Jumper wires

Procedure:

1. Open Arduino IDE
2. Open new sketch and type the program.
3. Compile the program.
4. Connect the NODEMCU board with USB cable to the computer.
5. Select tools -> select board ->NodeMCU 1.0
6. Select tools -> select port.
7. Upload the program.
8. Connect the circuit as per the circuit diagram.
9. Paste the URL found in Serial monitor (http://192.168.43.249) in either phone or computer browser to see DHT11 sensor values.

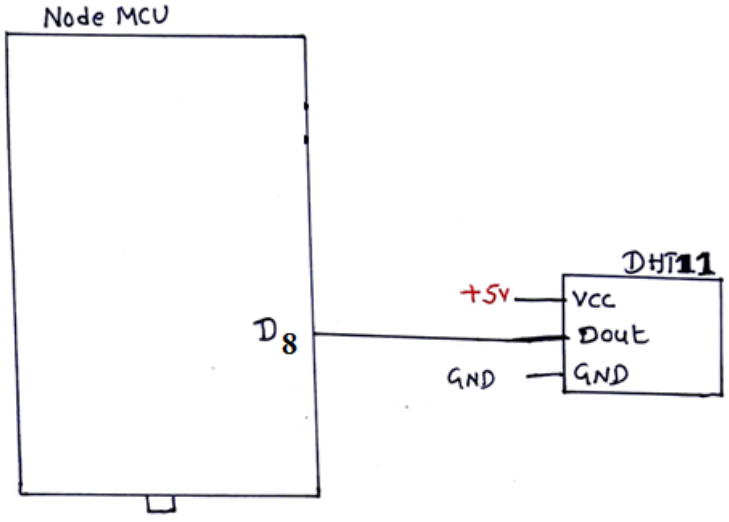
PROGRAM

```
#include <ESP8266WiFi.h>
#include <ESP8266WebServer.h>
#include "DHT.h"

// Uncomment one of the lines below for whatever DHT sensor type you're
using!
#define DHTTYPE DHT11   // DHT 11
//#define DHTTYPE DHT21   // DHT 21 (AM2301)
//#define DHTTYPE DHT22   // DHT 22  (AM2302), AM2321

/*Put your SSID & Password*/
const char* ssid = "happyfy";  // Enter SSID here
const char* password = "asdfghjkl";  //Enter Password here

ESP8266WebServer server(80);

// DHT Sensor
uint8_t DHTPin = D8;

// Initialize DHT sensor.
```

```
DHT dht(DHTPin, DHTTYPE);

float Temperature;
float Humidity;

void setup() {
  Serial.begin(115200);
  delay(100);

  pinMode(DHTPin, INPUT);

  dht.begin();

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

  //connect to your local wi-fi network
  WiFi.begin(ssid, password);

  //check wi-fi is connected to wi-fi network
  while (WiFi.status() != WL_CONNECTED) {
  delay(1000);
  Serial.print(".");
  }
  Serial.println("");
  Serial.println("WiFi connected..!");
  Serial.print("Got IP: ");  Serial.println(WiFi.localIP());

  server.on("/", handle_OnConnect);
  server.onNotFound(handle_NotFound);

  server.begin();
  Serial.println("HTTP server started");

}
void loop() {

  server.handleClient();

}

void handle_OnConnect() {

 Temperature = dht.readTemperature(); // Gets the values of the temperature
  Humidity = dht.readHumidity(); // Gets the values of the humidity
  server.send(200, "text/html", SendHTML(Temperature,Humidity));
}

void handle_NotFound(){
  server.send(404, "text/plain", "Not found");
}

String SendHTML(float Temperaturestat,float Humiditystat){
  String ptr = "<!DOCTYPE html> <html>\n";
```

```
  ptr +="<head><meta name=\"viewport\" content=\"width=device-width,
initial-scale=1.0, user-scalable=no\">\n";
  ptr +="<title>ESP8266 Weather Report</title>\n";
  ptr +="<style>html { font-family: Helvetica; display: inline-block;
margin: 0px auto; text-align: center;}\n";
  ptr +="body{margin-top: 50px;} h1 {color: #444444;margin: 50px auto
30px;}\n";
  ptr +="p {font-size: 24px;color: #444444;margin-bottom: 10px;}\n";
  ptr +="</style>\n";
  ptr +="</head>\n";
  ptr +="<body>\n";
  ptr +="<div id=\"webpage\">\n";
  ptr +="<h1>ESP8266 NodeMCU Weather Report</h1>\n";

  ptr +="<p>Temperature: ";
  ptr +=(int)Temperaturestat;
  ptr +="°C</p>";
  ptr +="<p>Humidity: ";
  ptr +=(int)Humiditystat;
  ptr +="%</p>";

  ptr +="</div>\n";
  ptr +="</body>\n";
  ptr +="</html>\n";
  return ptr;
}
```

Circuit connections



Viva Questions

1. **What is DHT11 sensor?**

2. **What is DHT22 sensor?**

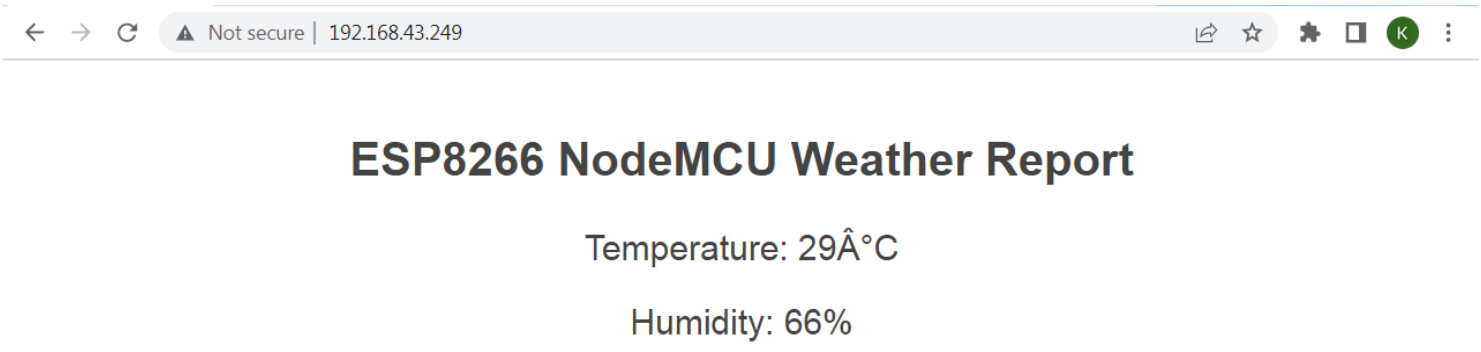
3. **What is arduino board?**

4. **What is arduino mega board ?**

5. **What is ESP32?**

Result:

Screenshot of web page can be seen below:



Temperature and Humidity are displayed on the web pages hosted by ESP32.

EXPERIMENT 11
UAV/Drone

AIM:

To Demonstrate of UAV elements, Flight Controller

COMPONENTS REQUIRED:

1. BLDC Motor
2. Properller
3. Lipo battery
4. Quadcopter frame
5. GPS
6. Flight controller
7. RC Receiver
8. RGBD camera

What is a flight controller?

Physically, a flight controller is nothing more than a circuit board with electronic chips on them. You can compare them to the motherboard and processor in your laptop. The flight controller is the brain of a drone. A small box filled with intelligent electronics and software, which monitors and controls everything the drone does. And just like the brains of different organisms, flight controllers also vary in sizes and complexity. (picture of different flight controllers)

What does a flight controller for drones do?

I noticed that everything a flight controller does can be classified within one of three categories: Sensing, controlling, and communicating.

Perception (sensing)

The flight controller is connected to a set of sensors. These sensors give the flight controller information about like its height, orientation, and speed. Common sensors include an Inertial Measurement Unit (IMU) for determining the angular speed and acceleration, a barometer for the height, and distance sensors for detecting obstacles. Just like how we perceive as humans, the drone filters a lot of this information and fuses some to get more efficient and precise information. Advanced flight controllers can sense more precisely and detect differences more quickly.

Controlling

Aside from sensing what's going on, a flight controller… unsurprisingly controls the motion of the drone. The drone can rotate and accelerate by creating speed differences between each of its four motors. The flight controller uses the data gathered by the sensors to calculate the desired speed for each of the four motors. The flight controller sends this desired speed to the Electronic Speed Controllers (ESC's), which translates this desired speed into a signal that the motors can understand.

Calculating the movements, fusing and filtering the sensory information, and estimating the safety and durability of a flight is all done by an algorithm. A fancy word that is used a lot nowadays which in essence nothing more than a set of strict rules that every microchip on the board has to apply to. The most commonly used flight control algorithm is called PID control: Proportional Integral Derivative control. Within this area, there is a lot of research going on, which resulted in INDI: Incremental Nonlinear Dynamic Inversion. This algorithm reads out and reacts to incoming information way faster, therefore making the drone flight more stable.

Communicating

A key part of a flight controller is communication. A part of the sensor's job is to give out information that needs to be translated clearly for a pilot to read, which means efficiently. An obvious thing to communicate is its battery level, which can decide if a pilot wants to fly further or return to the charge.

But communication goes further than from flight controller to human pilot; with the entrance of auto-pilot programs in the drone industry, flight controllers need to communicate with other computer systems about its flight destination and how to get there. Communication is mostly done with wi-fi and radio frequencies right now, but cellular solutions are also already in use.

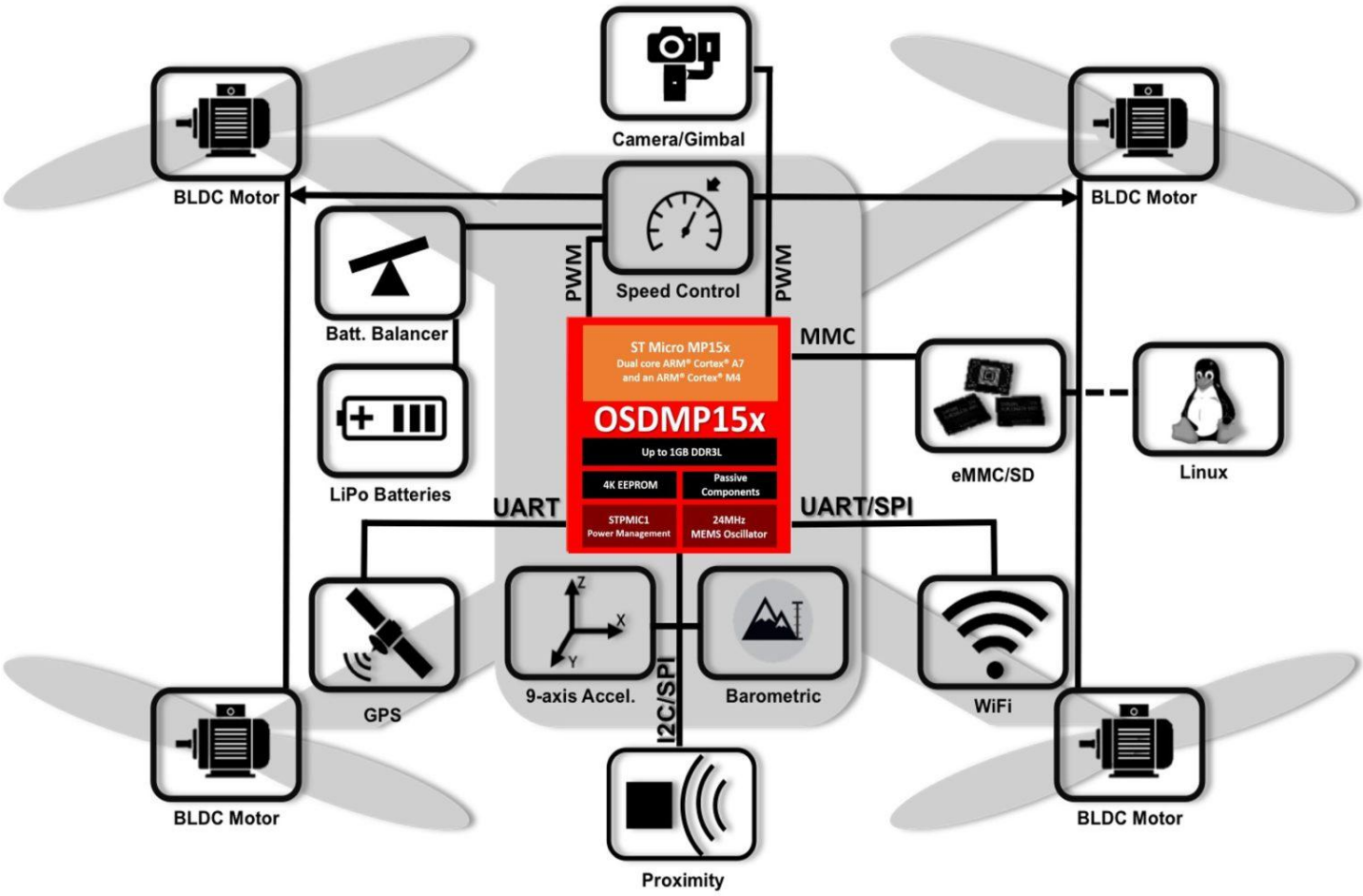


Fig: UAV elements and Flight controller

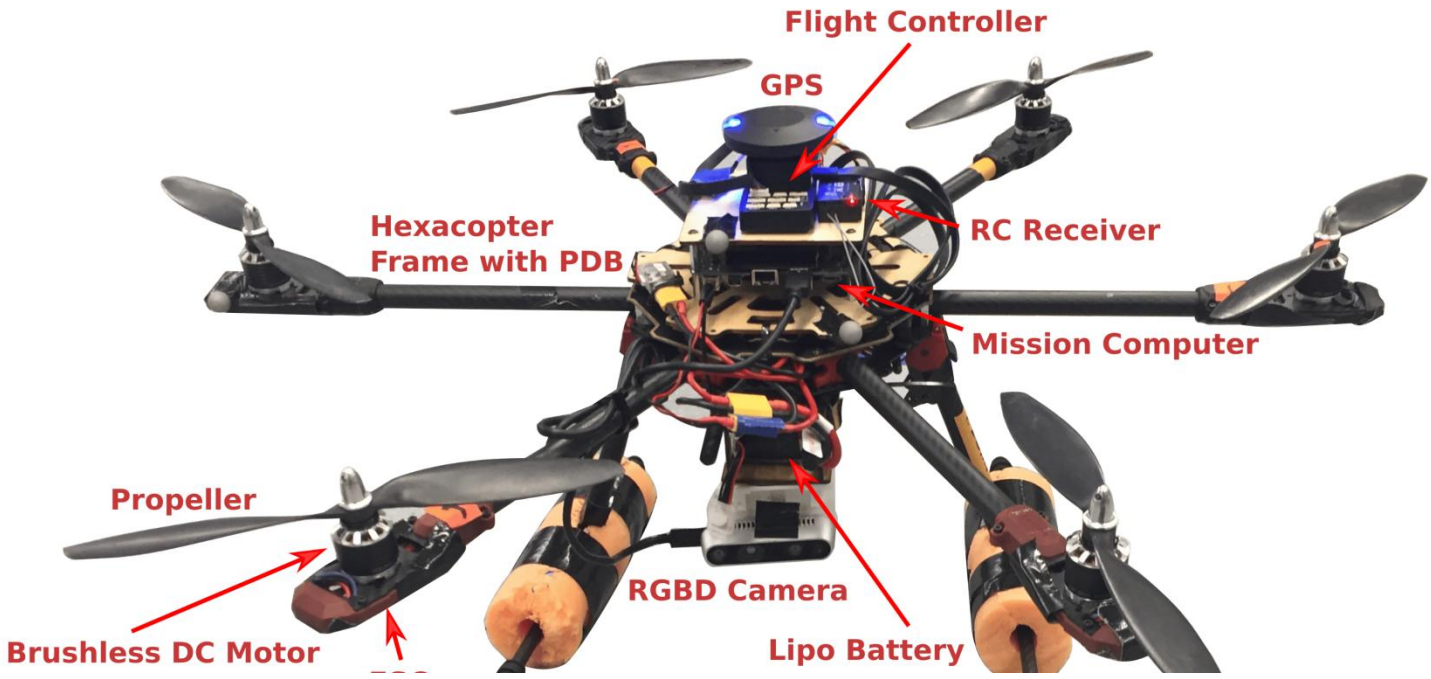


Fig: UAV elements and Flight controller

What kind of flight controllers are there?

There are a lot of different flight controllers on the market. They range from very basic to expensive systems. To make it a bit more comprehensible, I made four categories based on their users.

- **FC's for hobbyists/builders –** Easy to install and perfect for people that do not want to spend large amounts of money from the get-go.
- **Racing FC's –** Designed to be very lightweight, precise, and responsive. Most of them are in the €50,- or less range.
- **FC's for filming** – Although mostly bought included in a drone with a camera, these flight controllers are more focussed on creating fluent shots and accessible handling for a pilot. Within this segment, Chinese company Dà-Jiāng Innovations (better known as DJI) is a household name.
- **Commercial FC's** – The latest segment to evolve in the previous years. These are for the most advanced drones, capable of safe flying and transporting high-value cargo. The biggest players in this field are DJI and Pixhawk, but new flight controllers like Auterion's Skynode and Fusion Engineering's Fusion Reflex are also promising flight controllers in the industry.

Viva Questions

1. **What is GPS?**
2. **What is BLDC motor?**
3. **What is autonomous navigation?**
4. **What is tuning of flight controller?**
5. **What is PWM controller?**

Result:

Various components of Drone and flight controller are demonstrated.

EXPERIMENT 12

Python program to read GPS coordinates from Flight Controller

AIM:

To write python program to read GPS coordinates form flight controller

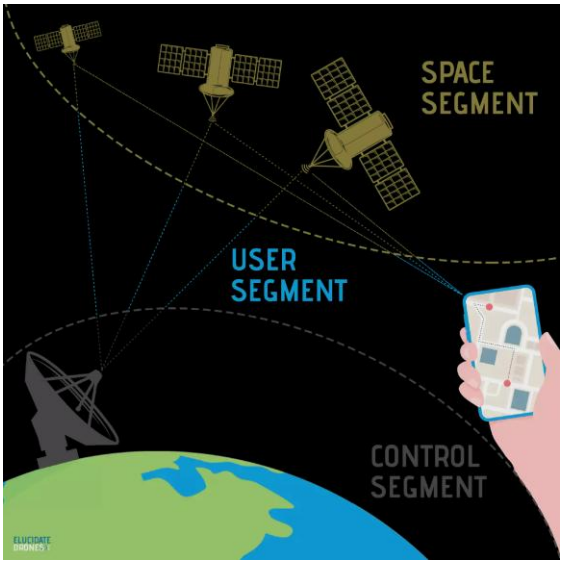
COMPONENTS REQUIRED:

1. Drone flight controller with GPS
2. Receiver Module

Navigating a drone through the skies is akin to orchestrating a symphony of technology, and at the heart of this harmonious dance is the marvel of navigation systems. Picture a drone as a modern-day adventurer equipped with an ensemble of sensors, each playing a crucial role in steering it across the vast landscapes.

Among these technological maestros, the Global Positioning System (*GPS*) receiver stands out as a virtuoso. Nestled onboard every outdoor-bound drone, the GPS receiver, part of the *User Segment*, performs a captivating act. It taps into a cosmic ballet with GPS Satellites, receiving their signals and translating them into a mesmerizing display of latitude and longitude values.

These latitude and longitude coordinates, the secret language of locations, join forces to form what we fondly call geographical coordinates or GPS coordinates. It's as if the drone speaks the Earth's own dialect, allowing us to pinpoint its exact whereabouts with ease.



Unveiling Drone's Geographical Coordinates

Setting the Stage

Picture this: Your drone, a digital voyager, is waiting for your commands. Before we unleash its potential, let's establish a connection. Meet the protagonist of our story, the `vehicle` object, the gateway to your drone's inner workings. Depending on your choice, you might name it `drone`, `quadcopter`, or perhaps, something even more whimsical.

Python

```
drone = dronekit.connect('udpin:127.0.0.1:14551', baud=115200)
```

With the stage set, let's unveil the script that will cast a spotlight on your drone's whereabouts.

The Script: Geo Coords Unveiled

Behold, the magical script named `geo_coords.py` – a script that will unravel the geographical coordinates of your airborne companion. Feel free to clone the script's repository if you're feeling tech-savvy!

geo_coords.py

```python
#!/usr/bin/env python

#.....................................................................
# Author  :  Saiffullah Sabir Mohamed
# Github  :  https://github.com/TechnicalVillager
# Website :  http://technicalvillager.github.io/
# Source  :  https://github.com/TechnicalVillager/geo-coords-dronekit/
#.....................................................................

# Import Necessary Packages
from dronekit import connect

# Connecting the Vehicle
vehicle = connect('udpin:127.0.0.1:14551', baud=115200)

# Printing Vehicle's Latitude
print("Vehicle's Latitude          =  ", vehicle.location.global_relative_frame.lat)

# Printing Vehicle's Longitude
print("Vehicle's Longitude         =  ", vehicle.location.global_relative_frame.lon)
```

Source: Link

Cue the Drumroll: Execution

The moment has arrived! Execute the script on your terminal with the following command:

```
python geo_coords.py
```

The Grand Reveal: Your Drone's Coordinates

And there it is – the grand reveal! Feast your eyes on the geographical coordinates, painting a vivid picture of your drone's location in the vast canvas of the skies.

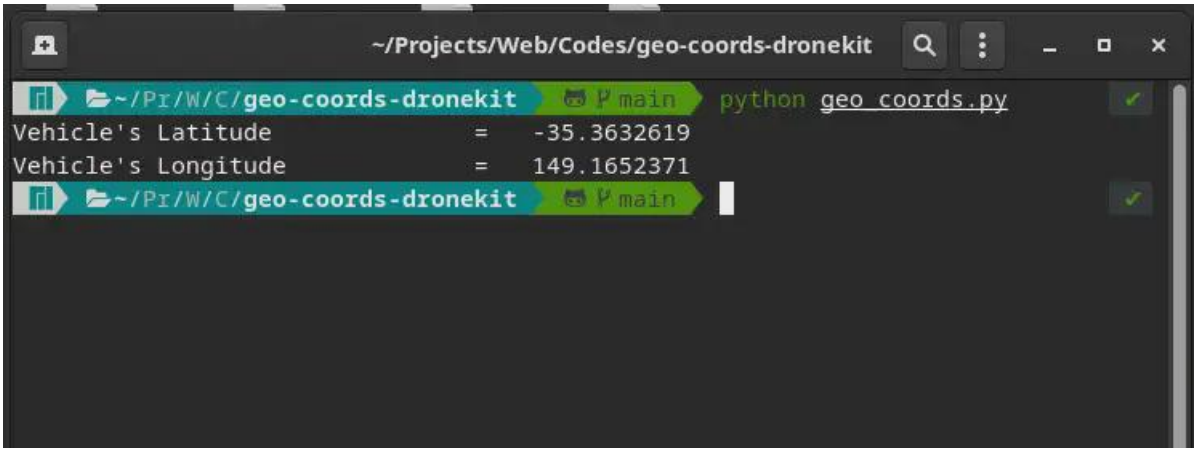


Image Credits: Dhulkarnayn, Elucidate Drones

How to Retrieve the Altitude of Your Drone?

Welcome to the next leg of our drone expedition – where we unravel the secrets of your drone's altitude. Brace yourself as we delve into the code, unlocking not just the height, but a trifecta of data – altitude, latitude, and longitude.

Setting the Scene: Altitude Unleashed

In this act, the spotlight is on the `geo_coords_alt.py` script, a magical piece of code that orchestrates the revelation of your drone's altitude. If you've been following our journey and cloned the repository, you're already in possession of this enchanting script.

Feel the anticipation building? Let's explore the script together.

Altitude Unveiled: The Script

geo_coords_alt.py

```
#!/usr/bin/env python

#.....................................................................
# Author  :  Saiffullah Sabir Mohamed
# Github  :  https://github.com/TechnicalVillager
# Website :  http://technicalvillager.github.io/
# Source  :  https://github.com/TechnicalVillager/geo-coords-dronekit/
#.....................................................................

# Import Necessary Packages
from dronekit import connect

# Connecting the Vehicle
vehicle = connect('udpin:127.0.0.1:14551', baud=115200)

# Printing Vehicle's Latitude
print("Vehicle's Latitude           =  ", vehicle.location.global_relative_frame.lat)

# Printing Vehicle's Longitude
print("Vehicle's Longitude          =  ", vehicle.location.global_relative_frame.lon)

# Printing Vehicle's Altitude
print("Vehicle's Altitude (in meters)  =  ", vehicle.location.global_relative_frame.alt)
```

Source: Link

Cue the Drumroll: Execution

The time has come to execute this script and witness the spectacle. Open your terminal and let the magic unfold:

```
python geo_coords_alt.py
```

The Grand Reveal: Altitude and Beyond

Behold the result – not just the altitude but a fusion of coordinates, creating a visual symphony of your drone's location.

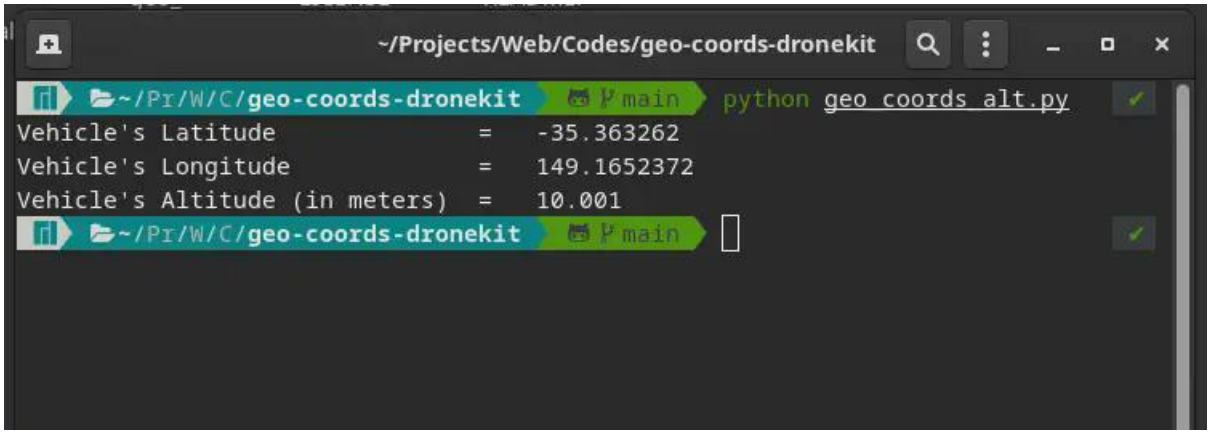


Image Credits: Dhulkarnayn, Elucidate Drones

Note

Immerse yourself in the world of simulated flight using ArduPilot's Software In The Loop (SITL) simulation and let the Mission Planner be your guide through the digital skies.

Unveiling the Heading Angle of Your Drone

Our journey continues as we turn our attention to the heading angle of your drone. This angle, much like a compass, guides your drone through the ever-expanding digital landscape. Let's explore the script `geo_coords_w_heading.py` together.

Setting the Stage: Heading to the Horizon

If you've been captivated by our journey and cloned the repository, you're just a step away from unlocking the heading angle script.

The Script: Navigating the Skies

geo_coords_w_heading.py

```
#!/usr/bin/env python

#.....
# Author  :  Saiffullah Sabir Mohamed
# Github  :  https://github.com/TechnicalVillager
# Website :  http://technicalvillager.github.io/
# Source  :  https://github.com/TechnicalVillager/geo-coords-dronekit/
#.....

# Import Necessary Packages
from dronekit import connect

# Connecting the Vehicle
vehicle = connect('udpin:127.0.0.1:14551', baud=115200)

# Printing Vehicle's Latitude
print("Vehicle's Latitude          = ", vehicle.location.global_relative_frame.lat)
```

```
18
19# Printing Vehicle's Longitude
20print("Vehicle's Longitude          = ", vehicle.location.global_relative_frame.lon)
21
22# Printing Vehicle's Altitude
23print("Vehicle's Altitude (in meters)  = ", vehicle.location.global_relative_frame.alt)
24
25# Printing Vehicle's Heading Angle
26print("Vehicle's Heading             = ", vehicle.heading)
```

Source: Link

Cue the Drumroll: Execution

Prepare for the grand reveal! Execute the script and let the heading angle unfold before you:

python geo_coords_w_heading.py

The Grand Reveal: Heading to the Digital Horizon

Witness not just the heading angle but a symphony of data – latitude, longitude, altitude – guiding your drone through the digital expanse.

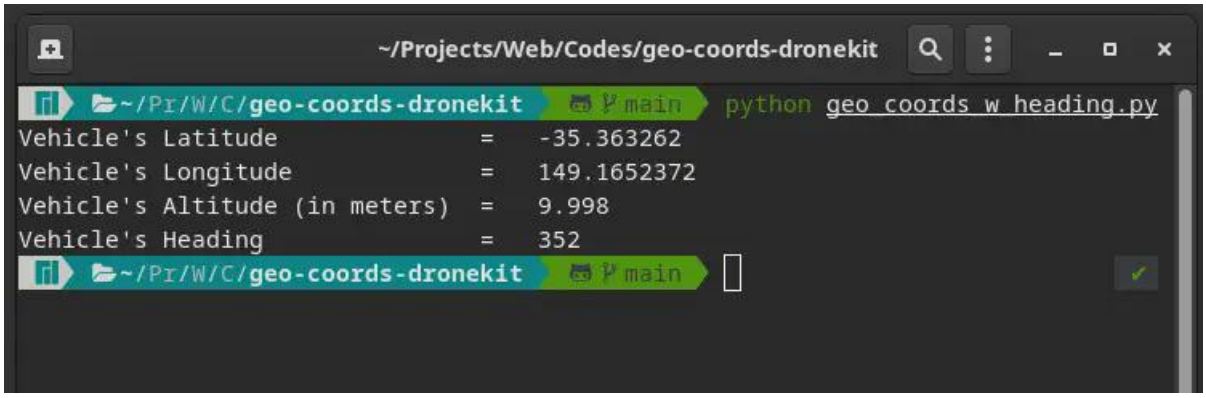


Image Credits: Dhulkarnayn, Elucidate Drones

Viva Questions

1. **What are the applications of drone?**
2. **What are the prerequisites for a person to fly a drone?**
3. **What is electronic speed controller of a drone?**
4. **What are different components of Drone?**
5. **What is propeller?**

Result:

GPS coordinates of drone are read using python.

ADDITIONAL EXPERIMENT 1

Design an IoT based air pollution control system which monitors the air pollution by measuring carbon monoxide, ammonia, etc and gives alarm or sends message when the pollution level is more than permitted range.

AIM:

To write a program to read gas value and send the data to Blynk IoT cloud and show it on the web and mobile dashboard and generating notification if gas value exceeds a threshold value.

COMPONENTS REQUIRED:

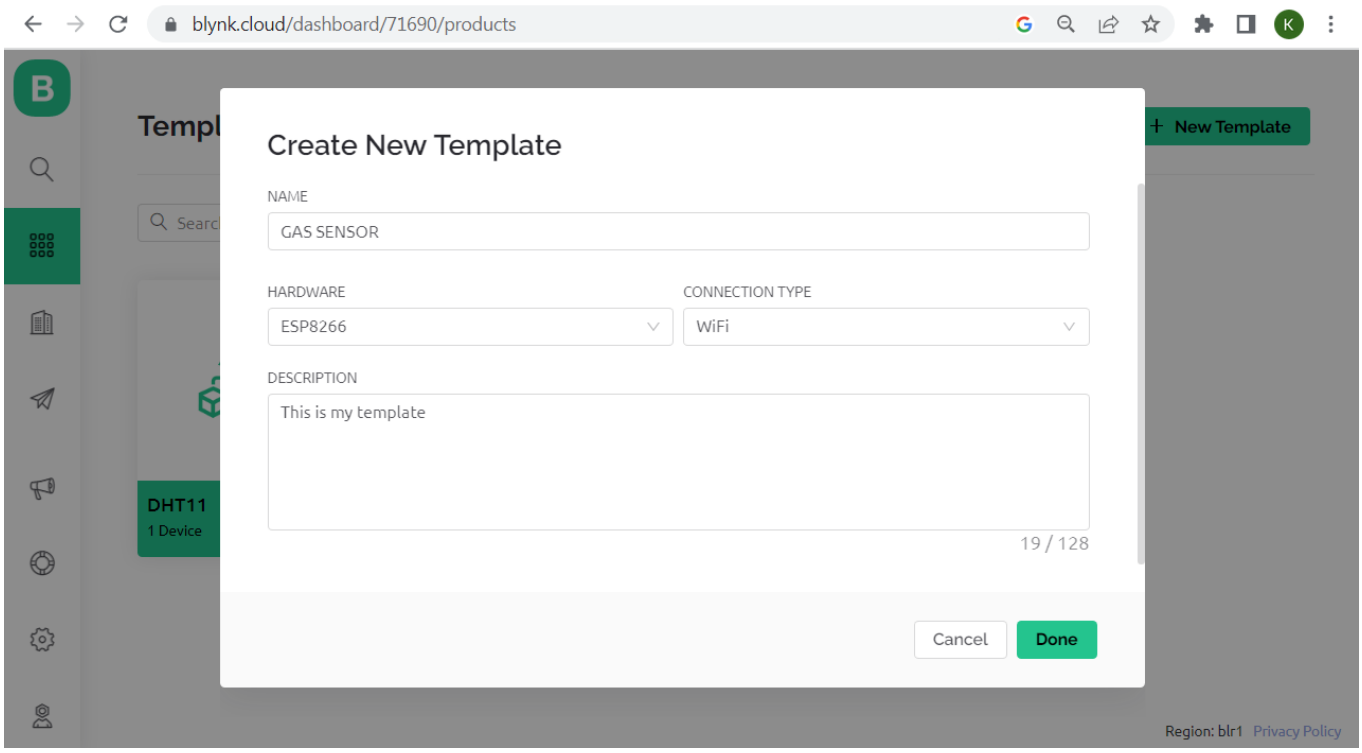
1. NodeMCU
2. Micro USB cable
3. MQ7 Sensor
4. Jumper wires

Procedure:

1. Open Arduino IDE
2. Open new sketch and type the program.
3. Compile the program.
4. Connect the NODEMCU board with USB cable to the computer.
5. Select tools -> select board ->NodeMCU 1.0
6. Select tools -> select port.
7. Upload the program.
8. Connect the circuit as per the circuit diagram.
9. Open web dashboard in blynk.
10. Observe the output.

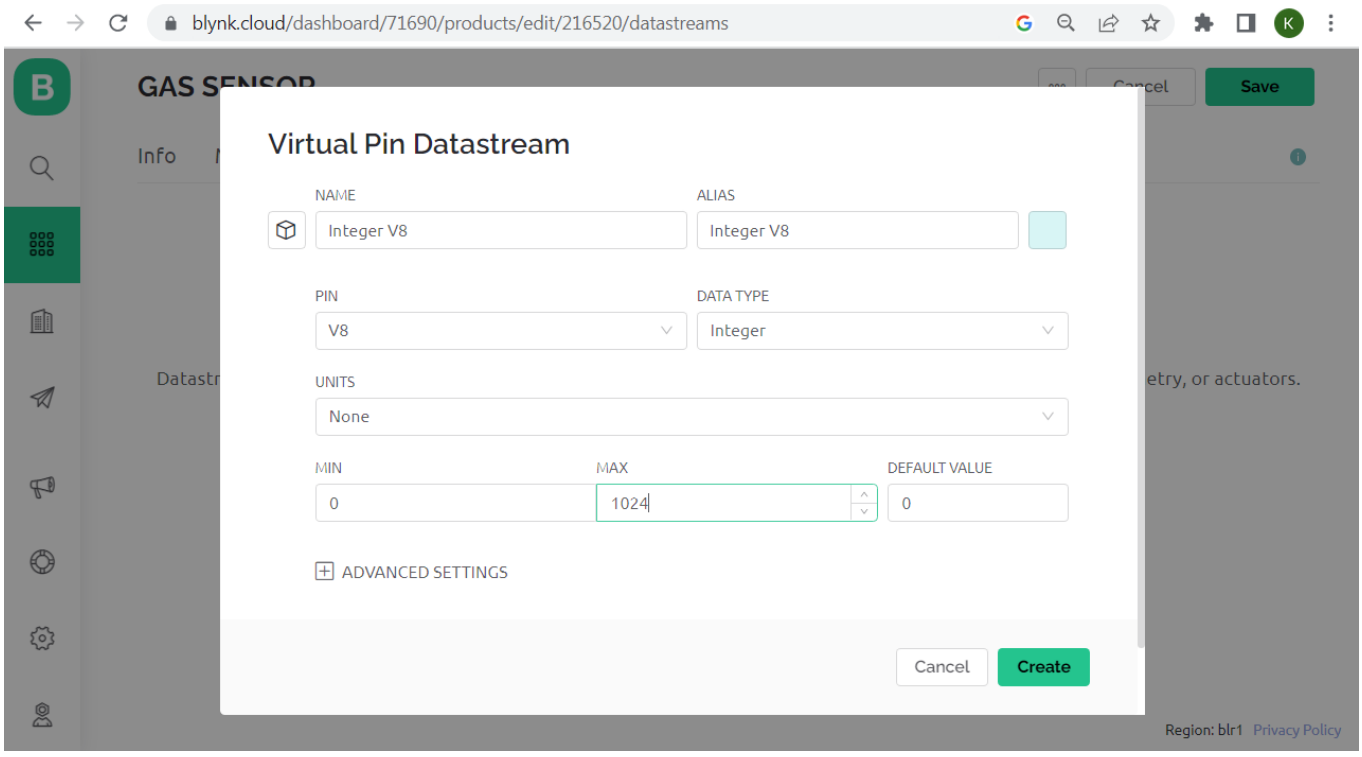
Step 2: Creating template

1. Open https://blynk.io
2. Login into blynk
3. Click on the templates menu in the left side.
4. Click on + New Template. Create New Template window opens.
5. In the NAME field type GAS SENSOR(or any appropriate name)
6. In the HARDWARE select ESP8266.
7. In the CONNECTION TYPE WiFi.
8. Click on Done

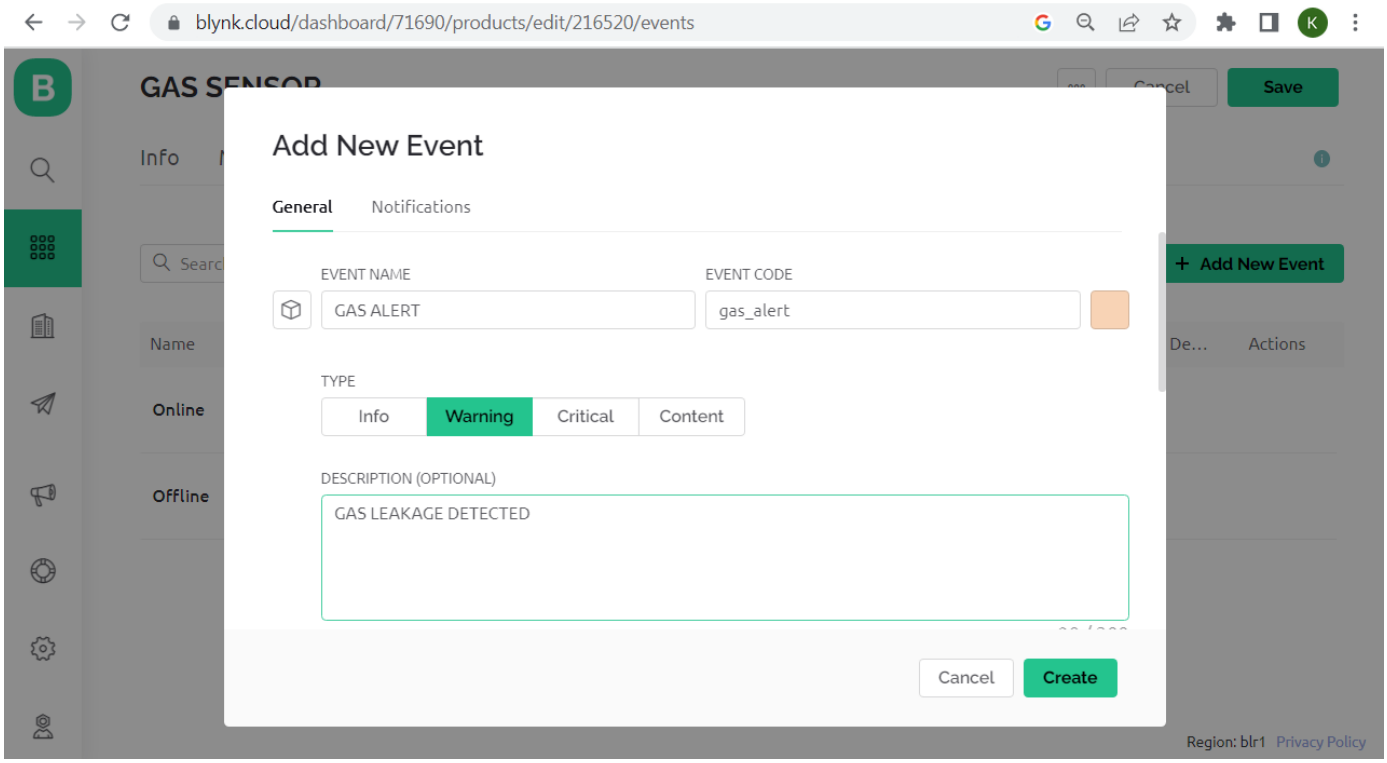


Step 3: Adding Datastreams

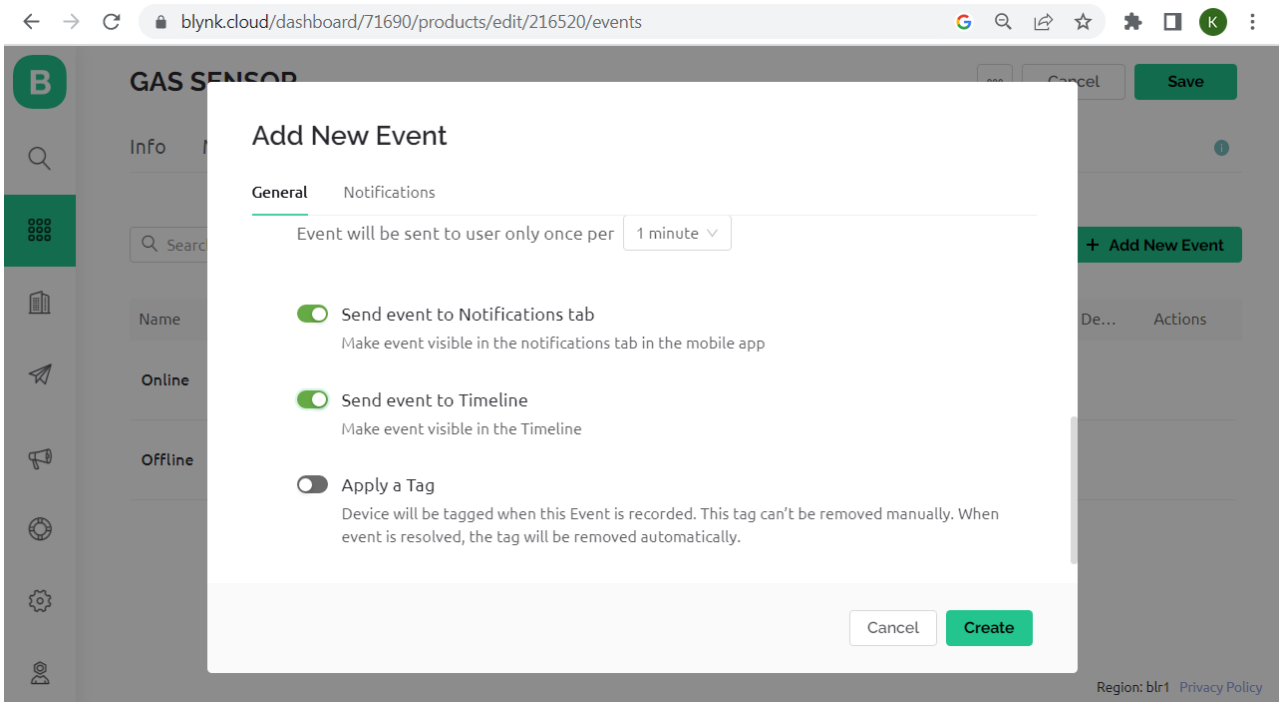
1. In the templates menu open the template GAS SENSOR
2. Go to Datastreams
3. Click on +New Datastream, select virtual Pin. Virtual Pin Datastream window opens.
4. Give the name as DoubleV8,pin as V8, DATA TYPE as Integer,set MIN to 0, MAX to 1024, DEFAULT VALUE to 0. Click on create.



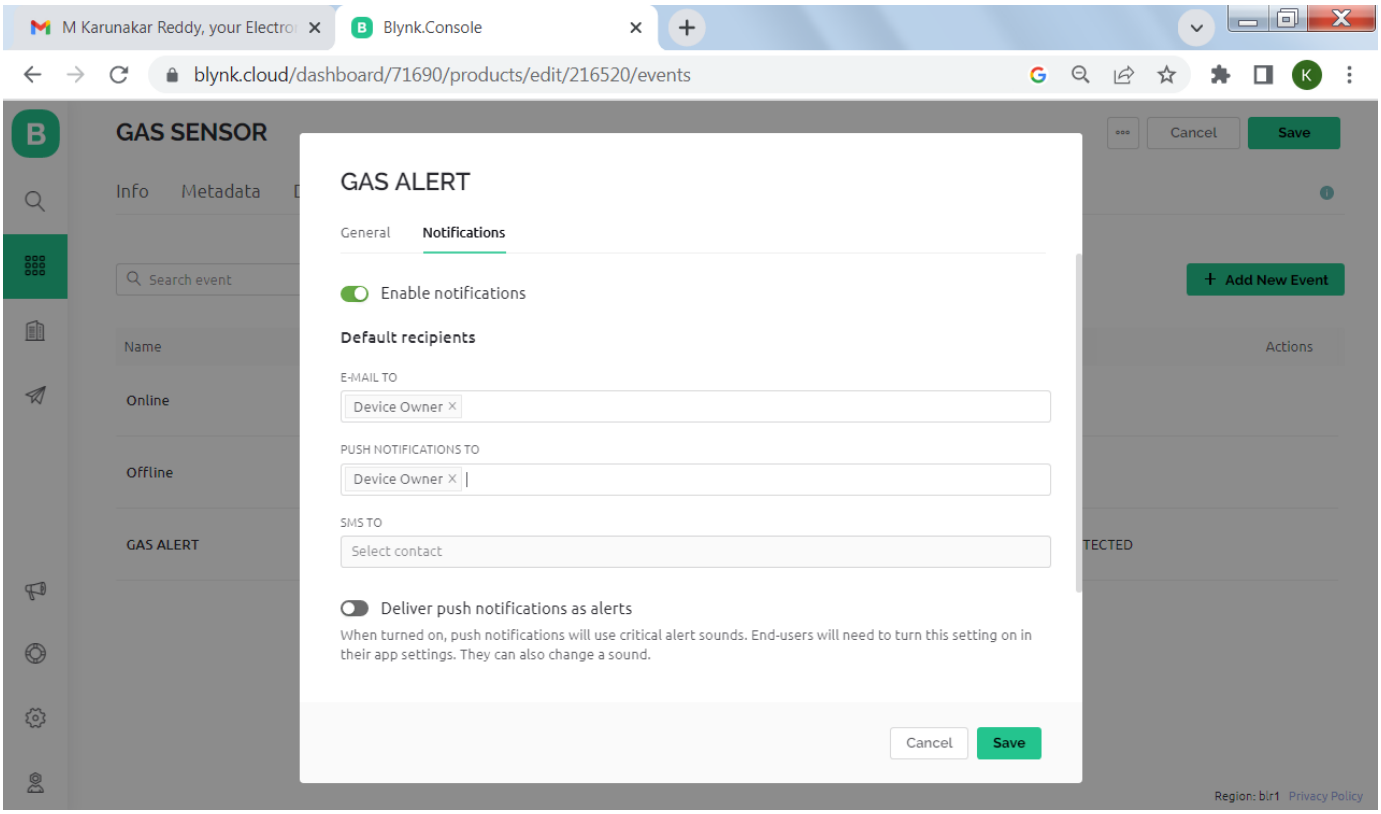
5. Click on Events-> click on +Add New Event.
6. In the general section, give the EVENT NAME as GAS ALERT
7. Click on Warning, in the DESCRIPTION, type GAS LEAKAGE DETECTED



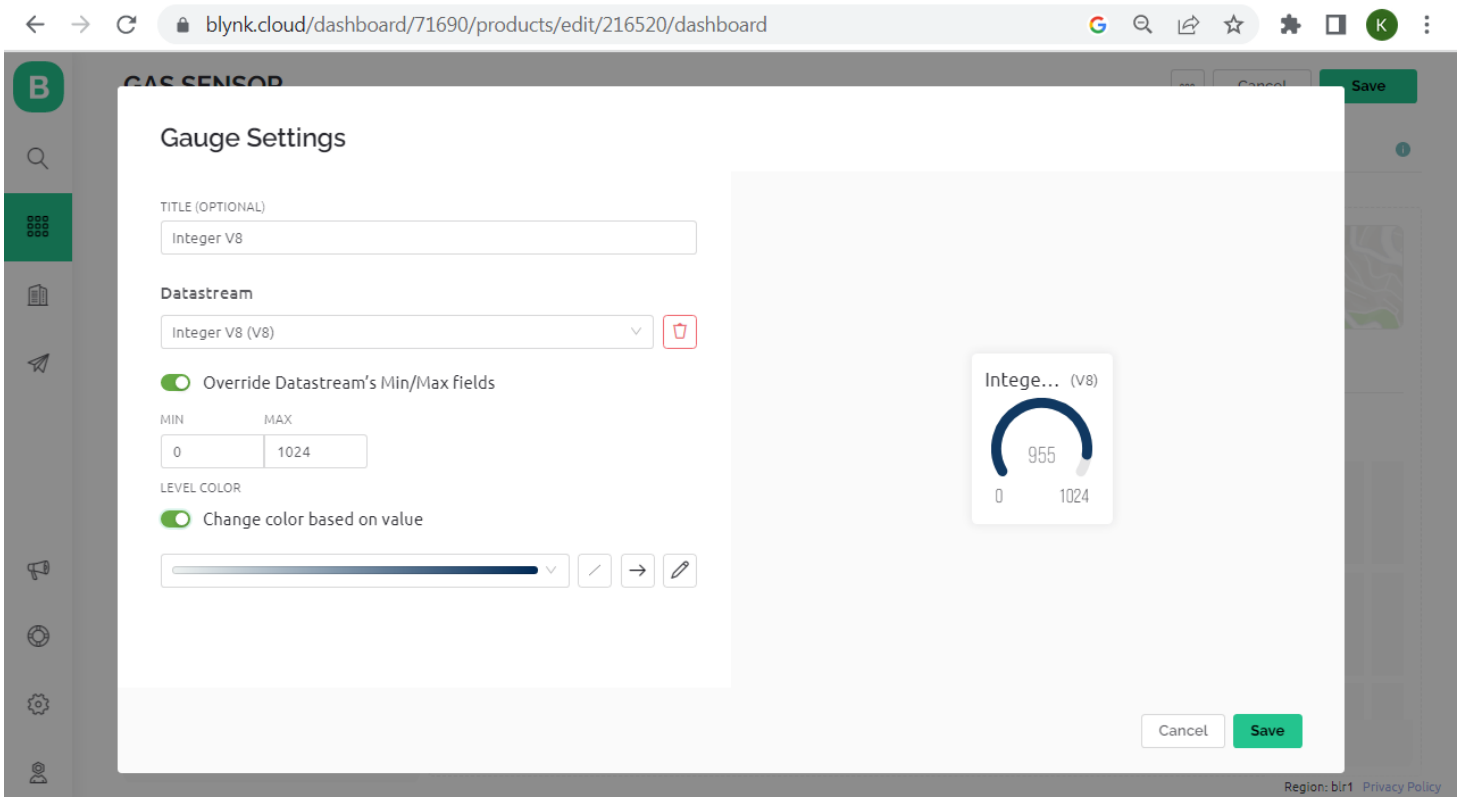
8. Scroll down, in the Event will be sent to user only once per 1 Minute.
9. Turn on radio buttons for Send event to Notifications tab, Send event to Timeline



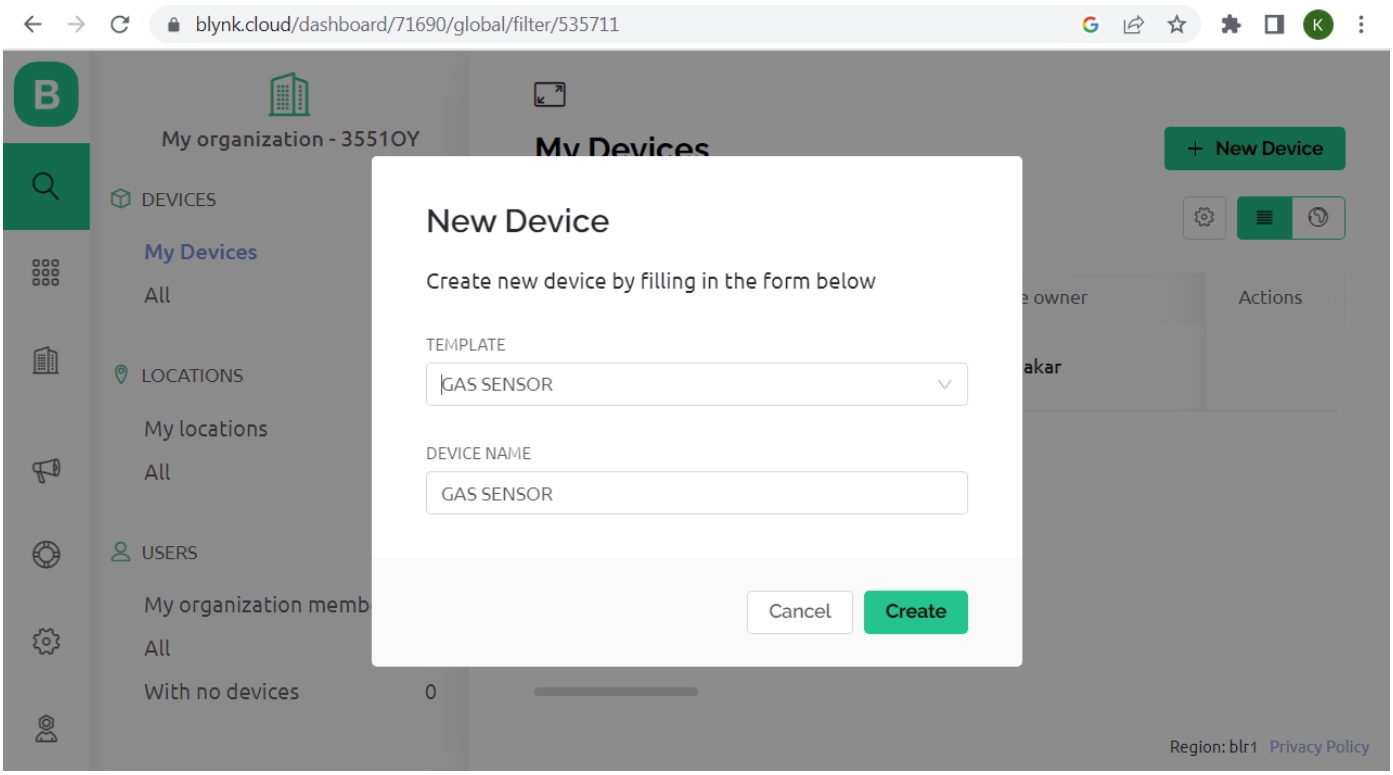
10. Click on Notifications, turn on radio button for Enable Notifications
11. Under E-MAIL FIELD , SELECT Device Owner, Under PUSH NOTIFICATIONS TO field select Device Owner. Click on create (or save)



12. Click on web dashboard add guage(choose datastream V8) with the values shown in the picture below.



10. Click on search symbol to create new device. Create device by selecting the template GAS sensor and let the device name be the same name as template, as shown in picture below.

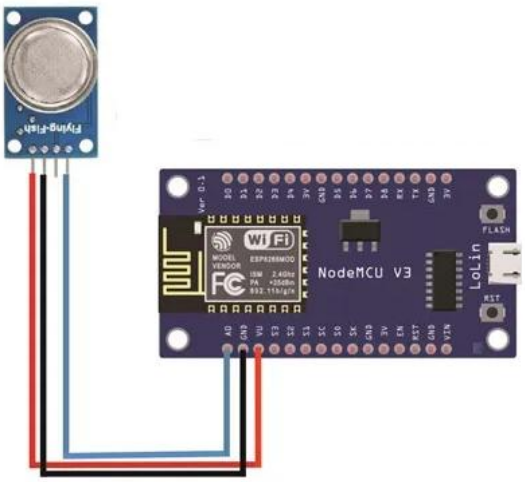


11. Now the device is created.

EXP A1: gas sensor

```
#define BLYNK_TEMPLATE_ID "TMPL4raE8f18"
#define BLYNK_DEVICE_NAME "Gas Leakage"
#define BLYNK_AUTH_TOKEN "s-O1oO5G4Xgu7nndsffEPZ6HdD81V74ZQpFy"

#define BLYNK_PRINT Serial
#include <ESP8266WiFi.h>
#include <BlynkSimpleEsp8266.h>

char auth[] = BLYNK_AUTH_TOKEN;

char ssid[] = "happyfy";  // type your wifi name
char pass[] = "asdfghjkl";  // type your wifi password
int data = 0;

BlynkTimer timer;

void sendSensor(){

 int data = analogRead(A0);
 Blynk.virtualWrite(V0, data);
  Serial.print("Smoke value: ");
  Serial.println(data);

  if(data > 600){

    Blynk.logEvent("gas_alert","Gas Leakage Detected");
  }
}
```

```
void setup()
{
  pinMode(A0, INPUT);
  Serial.begin(115200);
  Blynk.begin(auth, ssid, pass);
  timer.setInterval(2500L,
   sendSensor);
}

void loop()
{
  Blynk.run();
  timer.run();
}
```



Circuit connections

MQ7 sensor	NodeMCU
+	5V supply
out	A0
-	GND

Viva Questions

1. **What is gas sensor?**
2. **What gases are sensed by MQ2, MQ3, MQ4**
3. **What gases are sensed by MQ5, MQ6, MQ7**
4. **What gases are sensed by MQ8, MQ9, MQ135**
5. **What is the principle of operation of gas sensor?**
6. **What is analog sensor?**
7. **What is digital sensor?**

Result:

Gas Sensor data is shown on web dashboard and mobile dashboard, When gas value exceeds 600, notification and email alert are sent.

ADDITIONAL EXPERIMENT 2

Design an IoT based system which measures the physical and chemical properties of the water and displays the measured values.

AIM:

To write a program to read PH value and send the data to Blynk IoT cloud and show it on the web and mobile dashboard.

COMPONENTS REQUIRED:

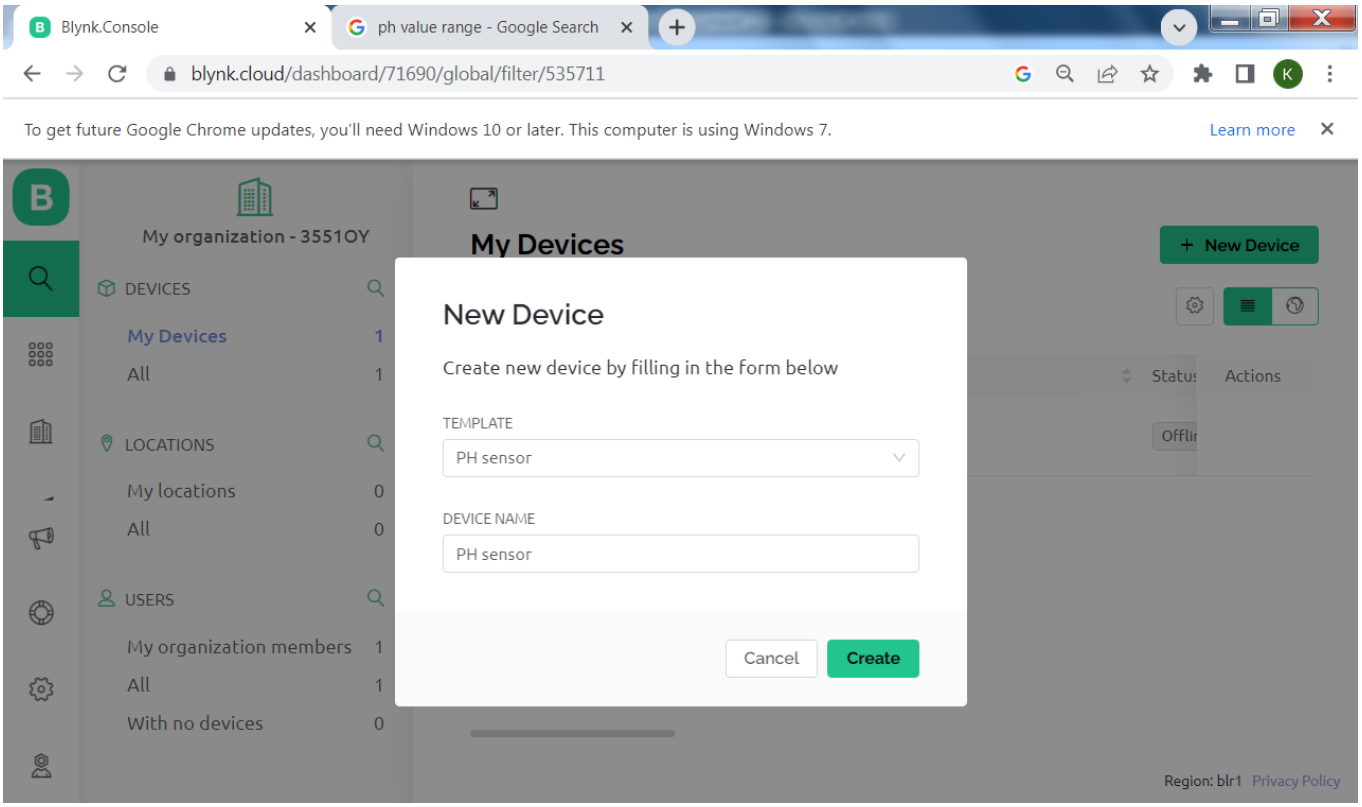
1. NodeMCU
2. Micro USB cable
3. PH Sensor
4. Jumper wires

Procedure:

1. Open Arduino IDE
2. Open new sketch and type the program.
3. Compile the program.
4. Connect the NODEMCU board with USB cable to the computer.
5. Select tools -> select board ->NodeMCU 1.0
6. Select tools -> select port.
7. Upload the program.
8. Connect the circuit as per the circuit diagram.
9. Open web dashboard in blynk.
10. Observe the output.

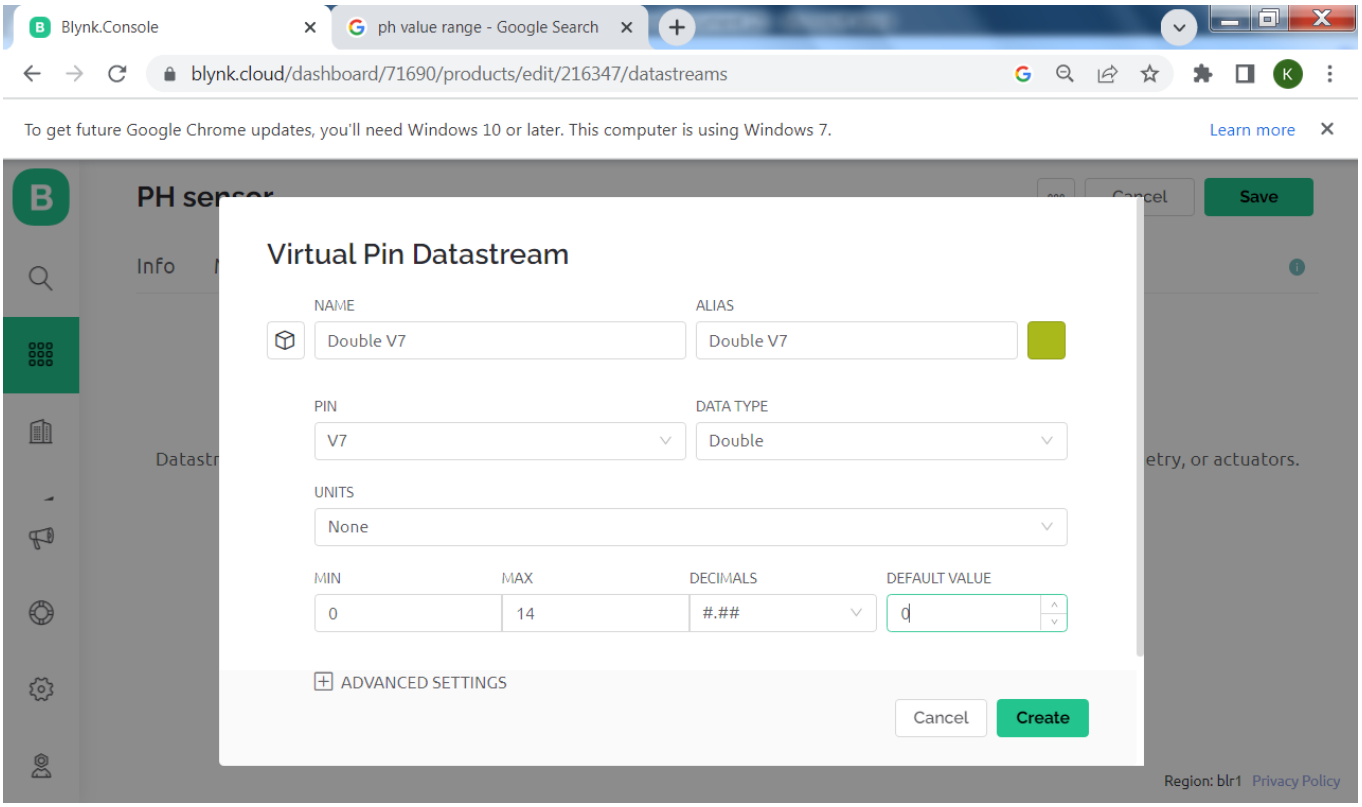
Step 2: Creating template

1. Open https://blynk.io
2. Login into blynk
3. Click on the templates menu in the left side.
4. Click on + New Template. Create New Template window opens.
5. In the NAME field type PH sensor(or any appropriate name)
6. In the HARDWARE select ESP8266.
7. In the CONNECTION TYPE WiFi.
8. Click on Done

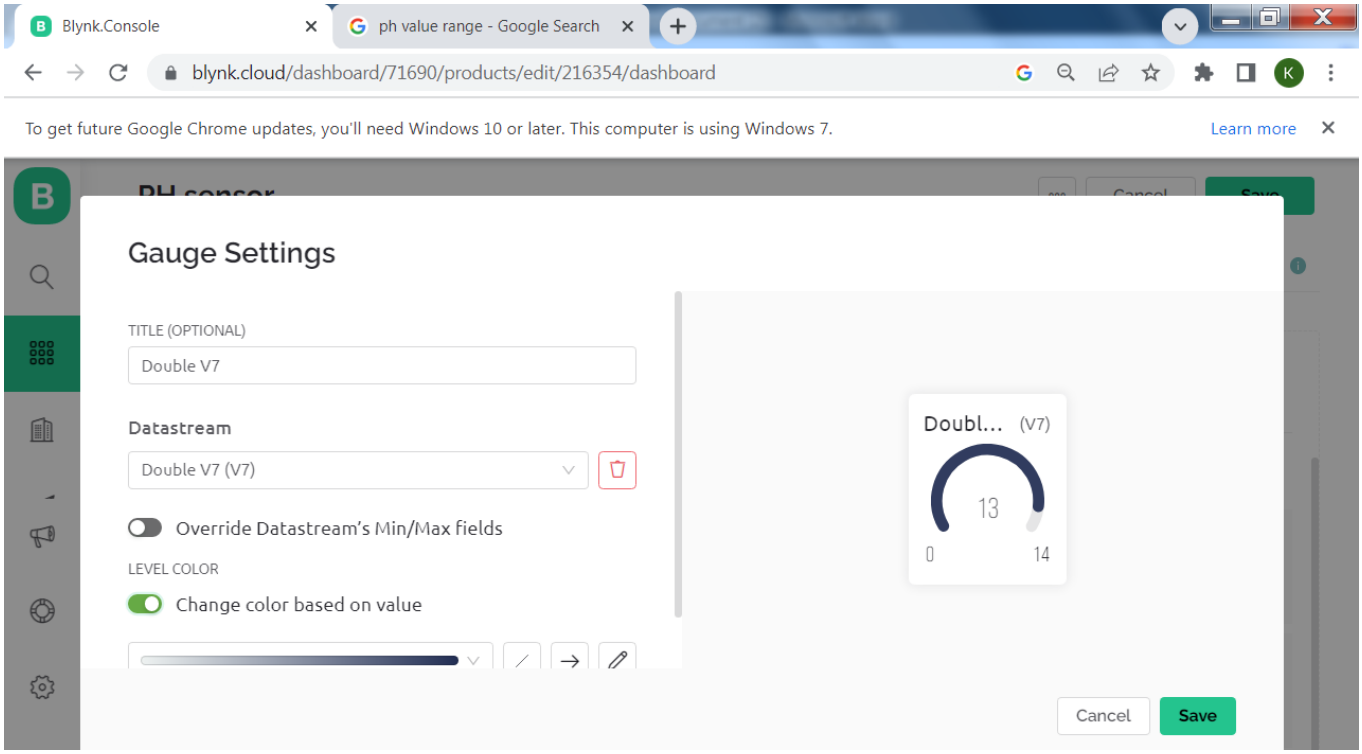


Step 3: Adding Datastreams

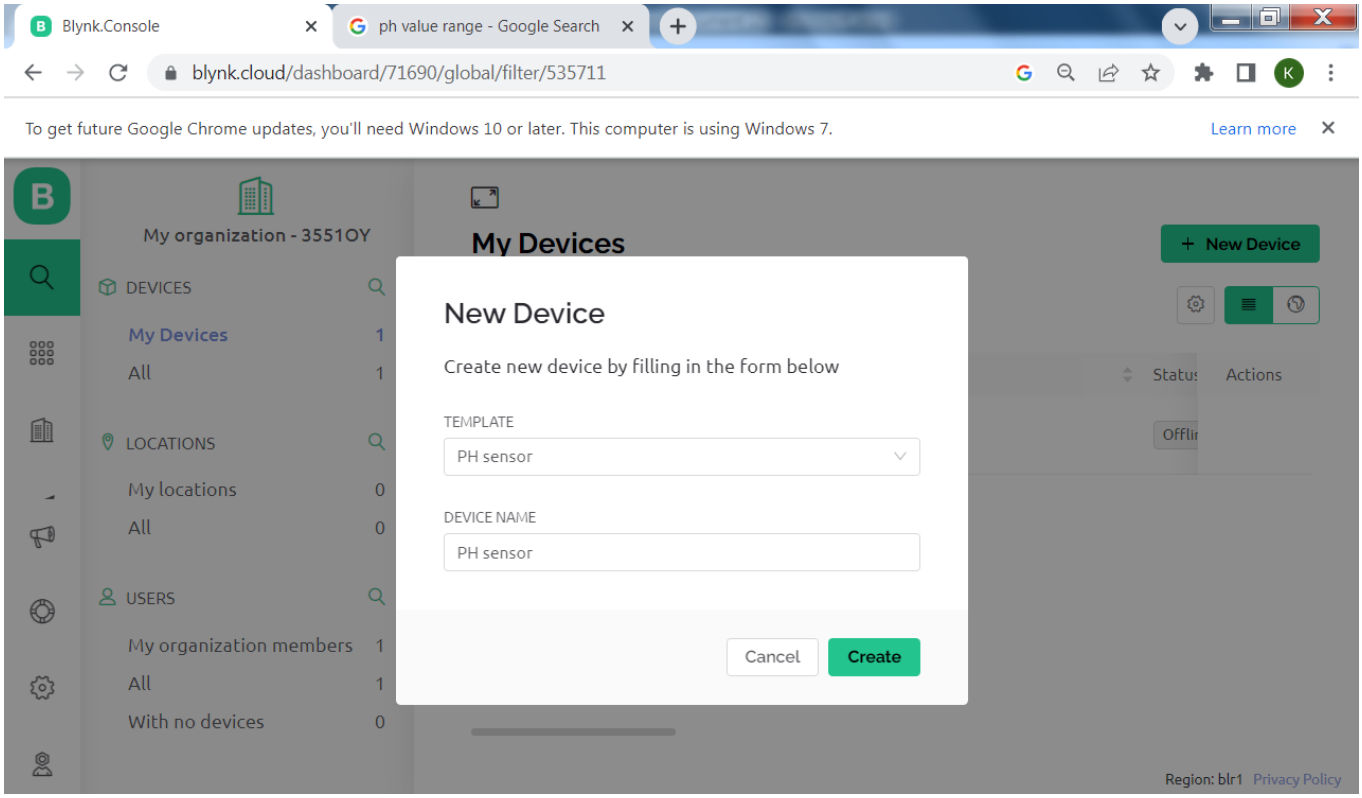
1. In the templates menu open the template DHT11
2. Go to Datastreams
3. Click on +New Datastream, select virtual Pin. Virtual Pin Datastream window opens.
4. Give the name as DoubleV7,pin as V7, DATA TYPE as Double, MIN to 0, MAX to 14 DECIMALS as #.## , DEFAULT VALUE to 0. Click on create.



5. Click on web dashboard add guage(choose datastream V7)



6. Click on search symbol to create new device. Create device selecting the template PH sensor and let the device name be the same name as template,as shown in picture below.

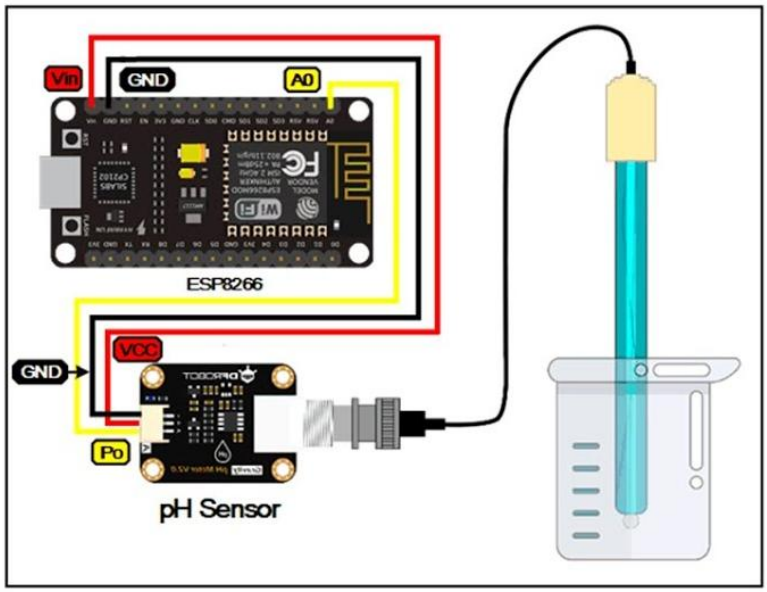


7. Now the device is created with the name as PH sensor

EXP A2: PH sensor

```
float ph;
float Value=0;

#define BLYNK_TEMPLATE_ID "TMPL6c-qkw-Z"
#define BLYNK_DEVICE_NAME "DHT11"
#define BLYNK_AUTH_TOKEN "kvaAn4C6LM6VHct6SKf1FQQnT8yS_cyM"

#define BLYNK_PRINT Serial

#include <ESP8266WiFi.h>
#include <BlynkSimpleEsp8266.h>

BlynkTimer timer;

char auth[] = BLYNK_AUTH_TOKEN;

char ssid[] = "happyfy";
char pass[] = "asdfghjkl";

void sendSensor()
{
    Value= analogRead(A0);
    Serial.print(Value);
    Serial.print(" | ");
    float voltage=Value*(3.3/4095.0);
    ph=(3.3*voltage);
    Serial.println(ph);

  Blynk.virtualWrite(V7, ph);
 }
void setup(){
  Serial.begin(9600);
  Blynk.begin(auth, ssid, pass);
  pinMode(A0, OUTPUT);
  // Setup a function to be called every second
  timer.setInterval(1000L, sendSensor);
}

void loop(){
  Blynk.run();
  timer.run();
}
```

Circuit diagram



Circuit connections

PH sensor	NodeMCU
+	5V supply
out	A0
-	GND

Viva Questions

1. **What is ph sensor?**

2. **What are API keys in Thingspeak?**

3. **What is A0 pin in nodeMCU?**

4. **What is accelerometer sensor?**

5. **What is hall effect sensor?**

Result:

PH Sensor data is shown on web dashboard and mobile dashboard.