DEPARTMENT OF

COMPUTER SCIENCE AND ENGINEERING

INTERNET OF THINGS LAB (20ES1452)

MANUAL

PVP-20 REGULATIONS

II B.TECH II SEM

PRASAD V. POTLURI SIDDHARTHA INSTITUTE OF TECHNOLOGY

(Autonomous, Accredited by NBA & NAAC, an ISO 9001:2015 certified institution)

(Sponsored by Siddhartha Academy of General & Technical Education)

VIJAYAWADA– 520 007

INDEX

Expt.No:1	Digital I/O Interface Blink LED & Multicolor LED
Dt:	

AIM:

- To interface and control **Blink LED & Multicolor** (RGB) LED with Arduino.

EXPERIMENTAL REQUIREMENTS:

S.No	Name of the equipment	Specifications	Quantity
1	Arduino	UNO	1
2	RGB common cathode LED		1
3	Jumper wires		As per requirement
4	Breadboard		1
5	Computer with Arduino IDE software	IDE 1.8.15	1

PROCEDURE:

1) Build the circuit according to the schematic diagram
2) Use Arduino Uno on the Arduino Desktop IDE.
3) Select board type and port.
4) Create a new file and write the required program.
5) Save the file and check for the errors by compiling it.
6) Upload the program in the Arduino-IDE
7) Observe the Output.

Blink LED

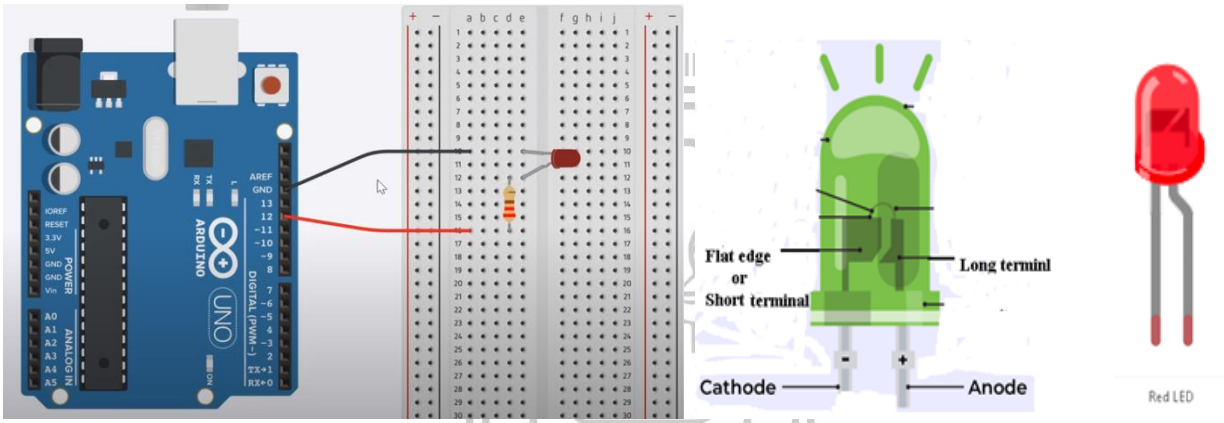
THEORY:

LED (Light Emitting Diode) is a two-terminal electronic device. The two terminals are called as Cathode and Anode. The long terminal is called Anode (positive terminal), and the shorter terminal is called Cathode (Negative terminal) .

LED emits light when the current passes through its terminals. LED's are used in various applications. It is also used as an ON/OFF indicator in different electronic devices.

Connect the long terminal of the LED to the digital pin (D13) correspondent to the *LED_BUILTIN* constant. Connect the short terminal of the LED to the GND. The LED will turned ON and OFF for a specified duration.

SCHEMATIC DIAGRAM:



PROGRAMME:

```
int led_pin= 13;

void setup() {
    pinMode (led_pin, OUTPUT);     // Initialize digital pin LED_BUILTIN as an output
 }
 // The loop function runs over and over again forever.
void loop() {
    digitalWrite(led_pin, HIGH);      //turn LED ON
    delay(500);                       //Wait for a second
    digitalWrite(led_pin, LOW);   //turn LED OFF
    delay(500);                       //Wait for a second
}
```

Multicolor LED

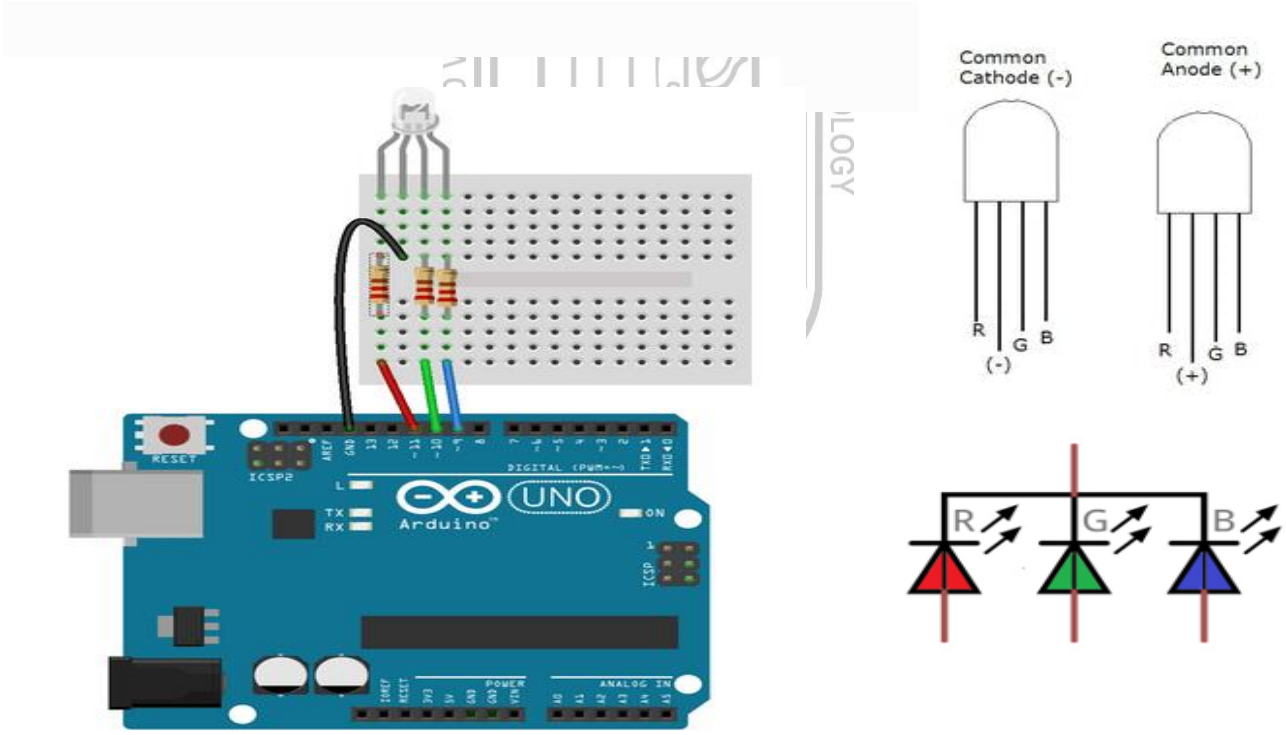
THEORY:

The RGB led consists of three different LEDs, red, green and blue. Many other colors can be obtained by mixing up these colors. The Arduino has aa analog write function which is used to obtain different colors for Arduino RGB led.

RGB LED Schematic

There are two types of RGB LEDs; the common cathode one and the common anode. In the common cathode RGB LED, the cathode of all the LEDs is common and a PWM signals is fed to the anode of LEDs while in the common anode RGB LED, the anode of all the LEDs is common and a PWM signals is fed to the cathode of LEDs.

SCHEMATIC DIAGRAM:



PROGRAMME:

```
int red= 11; // Connect R terminal to Pin 11 on Arduino
int green = 10; // Connect G terminal to Pin 10 on Arduino
int blue = 9; // Connect B terminal to Pin 9 on Arduino
void setup() {
pinMode(red, OUTPUT); // Configure red pin as OUTPUT pin
pinMode(green, OUTPUT); // Configure green pin as OUTPUT pin
pinMode(blue, OUTPUT); // Configure blue pin as OUTPUT pin
}
void loop() {
  RGB_color(255, 0, 0); // LED Light color is RED
  delay(1000);//Waits for 1 second
  RGB_color(0, 255, 0); // LED Light color is Green
  delay(1000);//Waits for 1 second
  RGB_color(0, 0, 255); // LED Light color is Blue
  delay(1000);//Waits for 1 second
 }
void RGB_color(intred_value, intgreen_value, intblue_value)
 {  analogWrite(red, red_value);
  analogWrite(green, green_value);
  analogWrite(blue, blue_value);
}
```

OBSERVATIONS:

S.No.	RGB Color Value	LED Color
1.	255, 0, 0	Red
2.	0, 255, 0	Green
3.	0, 0, 255	Blue

PROGRAMME:

```
int red = 11;
int green = 10;
int blue = 9;
void setup(){
pinMode(11, OUTPUT);
pinMode(10, OUTPUT);
pinMode(9, OUTPUT);
}
void loop(){
digitalWrite(red , 1);          // turn the RED LED on (HIGH is the voltage level)
digitalWrite(green, 0);
digitalWrite(blue, 0);
delay(1000); // wait for a second
digitalWrite(red, 0);
digitalWrite(green, 1);         // turn the GREEN LED on (HIGH is the voltage level)
digitalWrite(blue, 0);
delay(1000);
digitalWrite(red, 0);
digitalWrite(green, 0);
digitalWrite(blue, 1);          // turn the BLUE LED on (HIGH is the voltage level)
delay(1000);
}
```


OBSERVATIONS:

S.No.	RGB Color Value	LED Color
1.	1, 0, 0	Red
2.	0, 1, 0	Green
3.	0, 0, 1	Blue

RESULT:

Hence different Digital I/O Interfaces like Blink led, Multicolor LED are connected and controlled with Arduino board and corresponding outputs are observed.

Expt.No:2	Digital I/O Interface IR Sensor & Slot Sensor
Dt:	

AIM:

- To Interface a IR with Arduino to detect motion and display " Object detected " or " No Object " depending on the output from the sensor.
- To Interface a slot sensor with Arduino to detect object and control LED based on sensor output.

APPARATUS REQUIRED:

S.No	Name of the equipment	Specifications	Quantity
1	Arduino	UNO	1
2	IR Sensor		1
3	Slot Sensor		1
4	Jumper wires		required
5	Breadboard		1
6	Computer with Arduino IDE software	IDE 1.8.15	1

PROCEDURE:

1) Build the circuit according to the schematic diagram
2) Use Arduino Uno on the Arduino Desktop IDE.
3) Select board type and port.
4) Create a new file and write the required program.
5) Save the file and check for the errors by compiling it.
6) Upload the program in the Arduino - IDE
7) Observe the Output.

IR SENSOR:

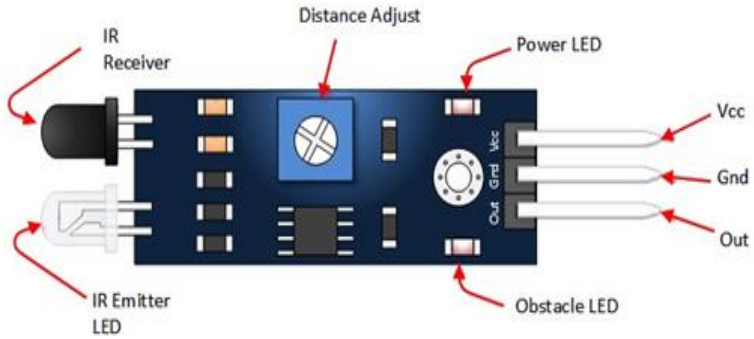
An **Infrared sensor** is an electronic module which is used to sense certain physical appearance of its surroundings by either emitting and/or detecting **infrared** radiation. **IR sensors** are also capable of determining the heat being emitted by an object and detecting motion.

IR SENSOR WORKING:

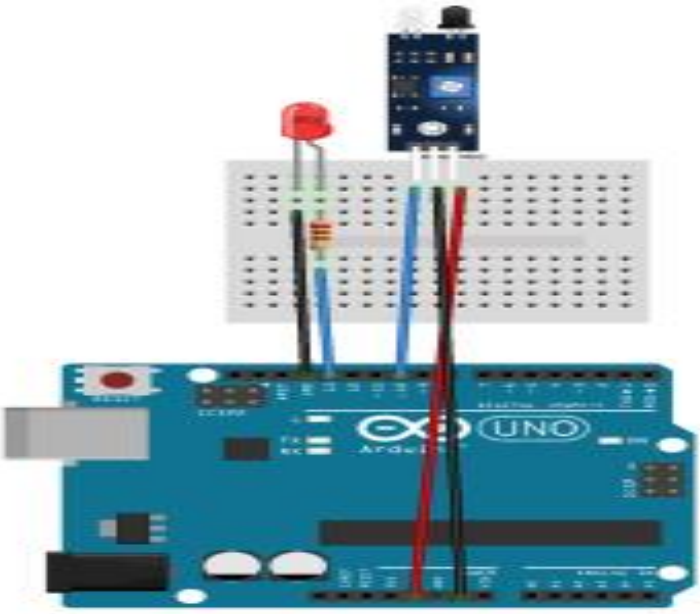
Here an IR sensor is used for detecting obstacles. IR transmitter transmits IR signal, as that signal detects any obstacle in its path, the transmitted IR signal reflects back from the obstacle and received by the receiver.

PINOUT:

1. VCC: 3.3V-5V power input pin
2. GND: 0V power pin
3. OUT: Digital Output Pin



SCHEMATIC DIAGRAM:



PROGRAMME:

```
int LED = 13;       // Connect RED LED anode at Arduino pin 13
int IR = 10;         // Connect IR out at Arduino pin 10
int Value = HIGH;  // Low Means No Obstacle
void setup() {
pinMode(LED, OUTPUT);         // Configure LED pin as OUTPUT pin
pinMode(IR, INPUT);           // Configure IR pin as INPUT pin
Serial.begin(9600);           // opens serial port, sets data rate to 9600 bps
}
void loop() {
Value = digitalRead(IR);      // read the value of the IR Sensor from pin 10
if (Value == LOW) {
Serial.println("Object detected"); // Prints data to the serial port
digitalWrite(LED, HIGH);      //Sets the LED ON
  }
else {
Serial.println("No Object");  // Prints data to the serial port
digitalWrite(LED, LOW);       //Sets the LED OFF
  }
delay(1000);                  //Waits for 1 second
}
```

OBSERVATIONS:

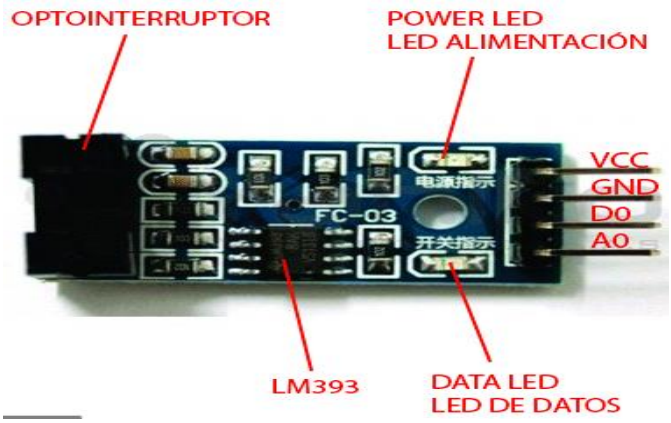
S.No.	OBSTACLE	LED Status
1.	YES	ON
2.	NO	OFF

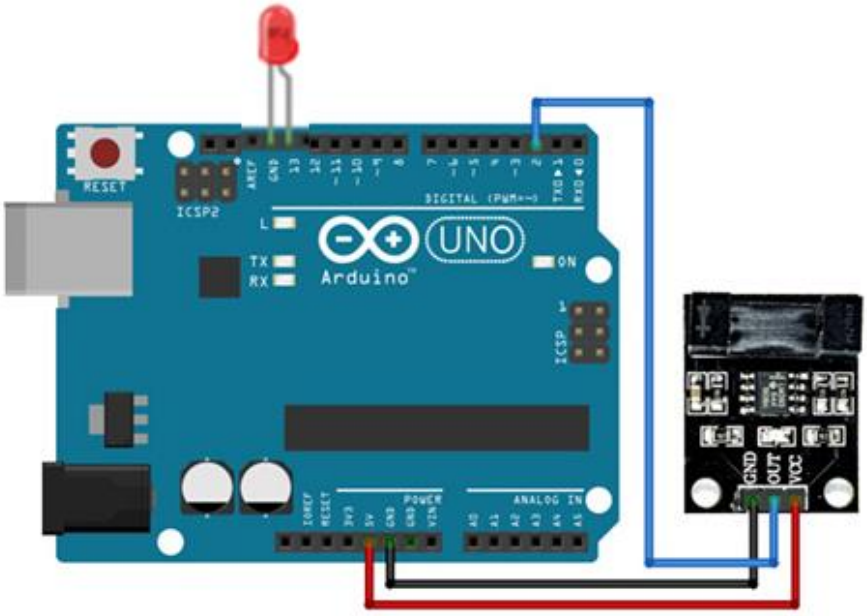
Slot Sensor:

LM393 Speed Sensor Module (H206)

H206 speed sensor module is composed of infrared light sensor and LM393 voltage comparator IC, so it is named as LM393 speed sensor. The module also includes a grid plate, which must be mounted on the rotating shaft of the motor.

The infrared light sensor is composed of infrared LED and phototransistor, separated by a small gap. As shown below, the entire sensor device is placed in a black housing. The grid plate is composed of grooves, and the plate is arranged between the gaps of the infrared light sensors so that the sensor can sense the gap in the grid plate. Each gap in the grid plate triggers the IR sensor when passing through the gap; a comparator is then used to convert these triggers into a voltage signal. The comparator uses ON Semiconductor's LM393 IC. The module has three pins, two of which are used to power the module, and one output pin is used to count the number of triggers.





PROGRAMME:

```
int sensor = 2;// connect slot sensor to Arduino pin 2
int LED = 13; // connect Led to Arduino pin 13
void setup(){
Serial.begin(9600); // opens serial port, sets data rate to 9600 bps
pinMode(sensor,INPUT);
}
void loop() {
int statusSensor = digitalRead(sensor); //Reads data from sensor
if (statusSensor == 1)
{
Serial.println("HIGH"); // Prints data to the serial port
digitalWrite(LED, HIGH); //Sets the LED ON
}
else
{
digitalWrite(LED, LOW); //Sets the LED OFF
Serial.println("LOW"); // Prints data to the serial port
}
}
```


OBSERVATIONS:

S.No.	OBSTACLE	LED Status
1.	YES	ON
2.	NO	OFF

RESULT:

Hence different Digital I/O Interfaces like IR Sensor and Slot Sensor connected and controlled with Arduino board and corresponding outputs are observed.

Expt.No:3	Analog Read and Write Potentiometer, Led Brightness Control
Dt:	

AIM:

- To observe the different resistance values of potentiometer with Arduino.
- To control LED brightness using potentiometer with Arduino.

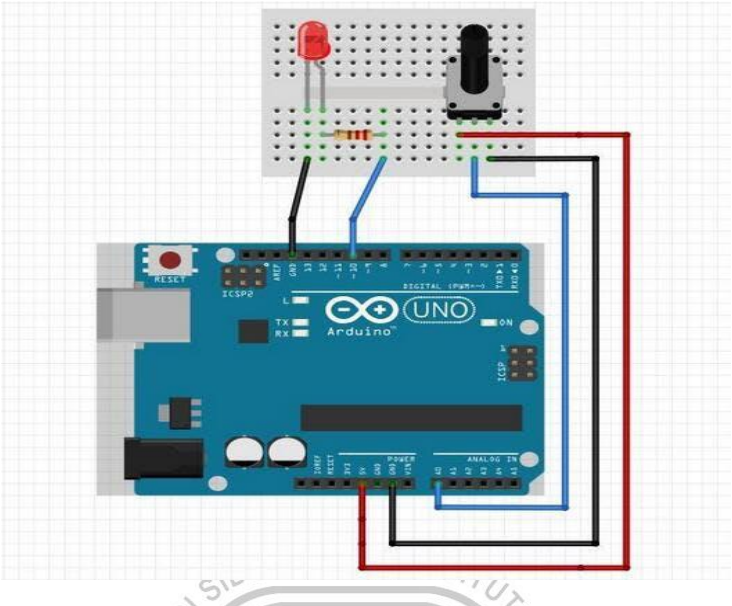
APPARATUS REQUIRED:

S.No	Name of the equipment	Specifications	Quantity
1	Arduino	UNO	1
2	Potentio Meter	1K Ω	1
3	LED		1
4	Jumper wires		As per requirement
5	Breadboard		1
6	Computer with Arduino IDE software	IDE 1.8.14	1

PROCEDURE:

1) Build the circuit according to the schematic diagram
2) Use Arduino Uno on the Arduino Desktop IDE.
3) Select board type and port.
4) Create a new file and write the required program.
5) Save the file and check for the errors by compiling it.
6) Upload the program in the Arduino-IDE
7) By varying the potentiometer observe the LED brightness.

SCHEMATIC DIAGRAM:



PROGRAMME:

//POTENTIOMETER

```
void setup(){
Serial.begin(9600);
}
void loop(){
int analogValue = analogRead(A0);
Serial.print("Analog: "); // print out the value
Serial.println(analogValue);
delay(500);
}
```


LED Brightness Control

THEORY:

The analog input is a pin with ADC (Analog-to-Digital Converter) function. It can convert the externally input analog signal into a digital signal that can be recognized during chip operation, so as to realize the function of reading in the analog value. The Arduino analog input function has 10-bit precision, that is, it can convert a voltage signal of 0 to 5V into an integer form of 0 to 1024. Utilize the analogRead() function to read input voltage values by the potentiometer, and then use the analogWrite() function to control the brightness of the LED light.

PROGRAMME:

```
int LED_PIN =10;  // the PWM pin the LED is attached to
void setup() {
Serial.begin(9600);  // initialize serial communication at 9600 bits per second:
pinMode(LED_PIN, OUTPUT);  // Configure LED pin as output:
}
void loop() {
intanalogValue=analogRead(A0); //reads the input on analog pin A0(value between 0 and 1023)
int brightness=map(analogValue,0,1023,0,255);/ scales it to brightness(value between 0 and 255)
analogWrite(LED_PIN, brightness);  // sets the brightness LED that connects to pin 10
Serial.print("Analog: ");    // print out the value
Serial.print(analogValue);
Serial.print(", Brightness: ");   // print out the value
Serial.println(brightness);
delay(1000);//waits for 1 second
}
```

OBSERVATIONS:

S.No.	Potentiometer Value (K Ω)	LED Status
1.	Min	ON with high brightness
2.	Mid Value	ON with less brightness
3.	Max	OFF

RESULT:

Hence

- Observed the different resistance values of potentiometer and display in serial monitor.
- LED brightness has been controlled using potentiometer

Expt.No:4	Analog Read and Write
Dt:	Temperature Sensor

AIM:

- To observe the Temperature and Humidity, display in serial monitor with Arduino .

APPARATUS REQUIRED:

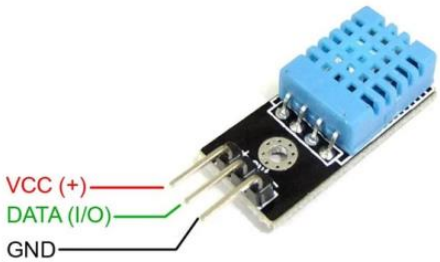
S.No	Name of the equipment	Specifications	Quantity
1	Arduino	UNO	1
2	Temperature Sensor	DTH11	1
3	Jumper wires		required
4	Breadboard		1
5	Computer with Arduino IDE software	IDE 1.8.15	1

PROCEDURE:

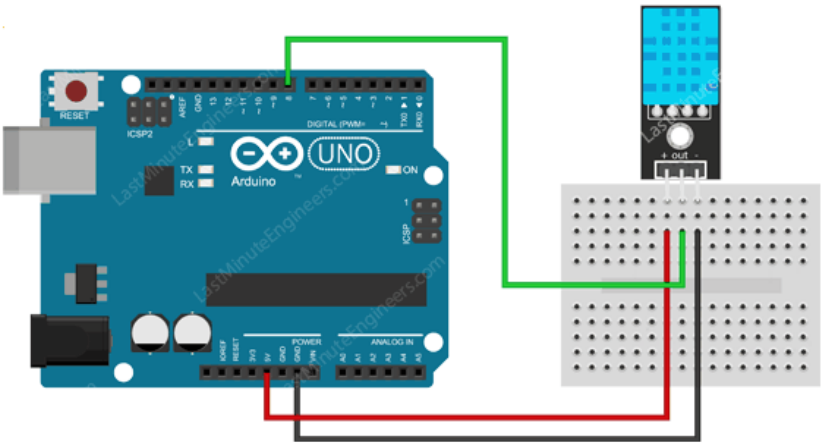
1) Build the circuit according to the schematic diagram
2) Use Arduino Uno on the Arduino Desktop IDE.
3) Select board type and port.
4) Write and upload the programin the Arduino-IDE.
5) Add the required library files.
6) Observe the temperature and Humidity values

Temperature sensor DTH11:

The digital temperature and humidity sensor DHT11 is a composite sensor that contains a calibrated digital signal output of temperature and humidity. The technology of a dedicated digital modules collection and the temperature and humidity sensing technology are applied to ensure that the product has high reliability and excellent long-term stability. The sensor includes a resistive sense of wet component and an NTC temperature measurement device, and is connected with a high-performance 8-bit microcontroller.



SCHEMATIC DIAGRAM:



PROGRAMME:

```
#include <dht.h>
    dht DHT;
    #define DHT11_PIN 8
    void setup(){
      Serial.begin(9600);
    }
    void loop(){
      intchk = DHT.read11(DHT11_PIN);
      Serial.print("Temperature = ");
      Serial.println(DHT.temperature);
      Serial.print("Humidity = ");
      Serial.println(DHT.humidity);
      delay(1000);
    }
```


OBSERVATIONS:

S.No.	Temperature	Humidity
1		
2		

RESULT:

Hence Temperature and Humidity has been measured using Temperature sensor and displayed using serial monitor.

Expt.No:5	DC MOTOR CONTROL DC MOTOR SPEED AND DIRECTION CONTROL
Dt:	

AIM:

To control DC Motor Speed and Direction using Arduino.

APPARATUS REQUIRED:

S.No	Name of the equipment	Specifications	Quantity
1	Arduino	UNO	1
2	DC Motor	12V High Torque	1
3	L298N Motor driver		1
4	Potentio Meter	10K Ω	1
5	Rechargeable Battery	5V	1
6	Jumper wires		As per requirement
7	Breadboard		1
8	Computer with Arduino IDE software	IDE 1.8.14	1

PROCEDURE:

1) Build the circuit according to the schematic diagram
2) Use Arduino Uno on the Arduino Desktop IDE.
3) Select board type and port.
4) Create a new file and write the required program.
5) Save the file and check for the errors by compiling it.
6) Upload the program in the Arduino-IDE
7) Control the speed and direction of DC motor by varying potentiometer and Serial input.

THEORY:

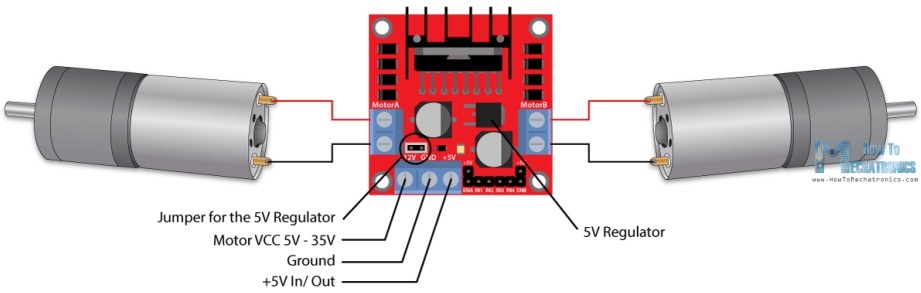
The L298N is a dual-channel H-Bridge motor driver capable of driving a 2x DC motors, making it ideal for building two-wheel robots.

Power Supply: From *'Vs' pin* the H-Bridge gets its power for driving the motors which can be 5 to 35V. *'Vss'* is used for driving the logic circuitry which can be 5 to 7V. And they both sink to a common ground named *'GND'*.

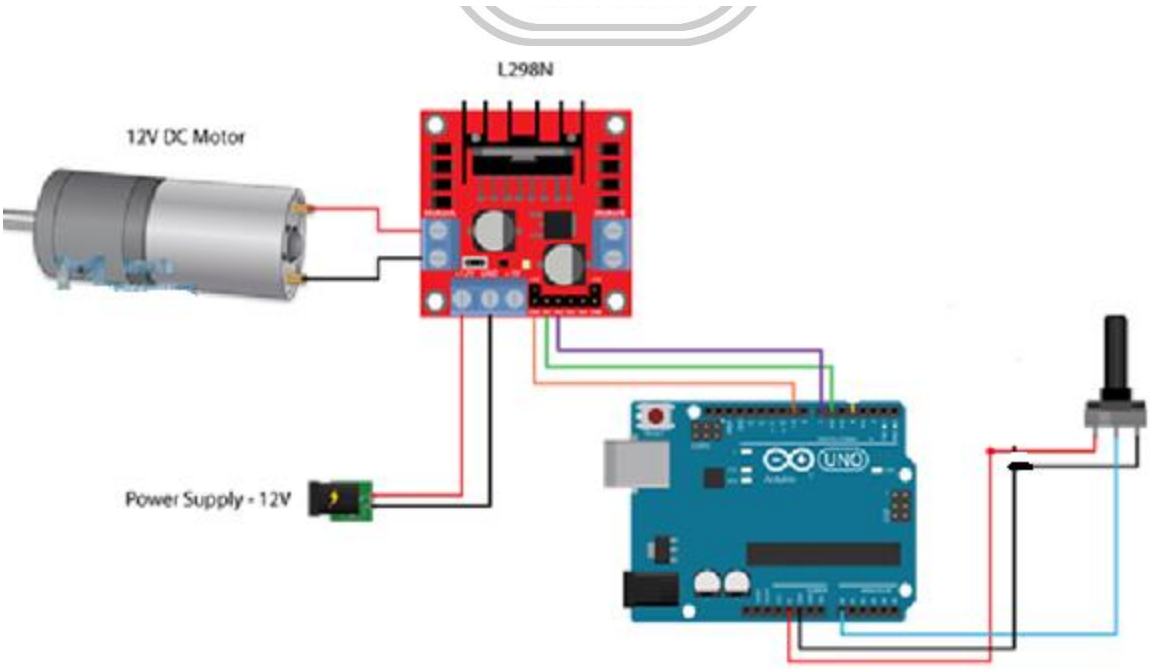
Output Pins: The L298N motor driver's output channels for the motor A and B are broken out to the edge of the module with two 3.5mm-pitch screw terminals.

Direction Control Pins: The IN1 and IN2 pins control the direction of the motor A while IN3 and IN4 control the direction of the motor B.

Speed Control Pins: ENA and ENB are used to turn the motors ON, OFF and control its speed. The module usually comes with a jumper on these pins. When this jumper is in place, the motor is enabled and spins at maximum speed. If speed of motors is to be controlled remove the jumpers and connect them to PWM-enabled pins on Arduino.



SCHEMATIC DIAGRAM:



PROGRAMME:

```
// DC motor direction and speed control
int in1 = 6;            // Connect IN1 to Arduino pin 6
int in2 = 7;            // Connect IN2 to Arduino pin 7
intenA = 9;             // Connect enA to Arduino pin 9
int Speed = 0;
intval = 0;
void setup()
{
Serial.begin(9600);                 // opens serial port, sets data rate to 9600 bps
pinMode(enA, OUTPUT);               // Configure IN1 pin as OUTPUT pin
pinMode(in1, OUTPUT);               // Configure IN2 pin as OUTPUT pin
pinMode(in2, OUTPUT);               // Configure enA pin as OUTPUT pin
}
// Enter value 0 or 1 to select motor direction, 0 for Backward, 1 for Forward
void loop()
{
val = analogRead(A0);               //Read potentiometer value to change the Motor speed
Speed = map(val,0,1023,0,255);
// Anti clockwise rotation
if (Serial.available() > 0)
{
char state = Serial.read();         //Reads incoming serial data
 // FORWARD rotation
if (state == '1')
   {
analogWrite(enA,Speed);                    //Writes an analog value (PWM wave) to a pin
digitalWrite(in1,1);
digitalWrite(in2,0);
Serial.println("FORWARD");      // Prints data to the serial port
delay(1000); //waits for 1 second
   }
```

```
// BACKWARD rotation
if (state == '0')
{
analogWrite(enA,Speed);
digitalWrite(in1,0);
digitalWrite(in2,1);
Serial.println("BACKWARD");              // Prints data to the serial port
delay(1000); //waits for 1 second
}
}
}
```

OBSERVATIONS:

By varying the potentiometer, speed of the DC motor is controlled and it changes the rotation direction of the motor depending on control input from serial monitor.

RESULT:

Hence DC Motor Speed and Direction has been controlled using Arduino.

Expt.No: 6	SERIAL COMMUNICATION - DEVICE CONTROL
Dt:	

AIM: To control LED using a serial monitor in Arduino IDE.

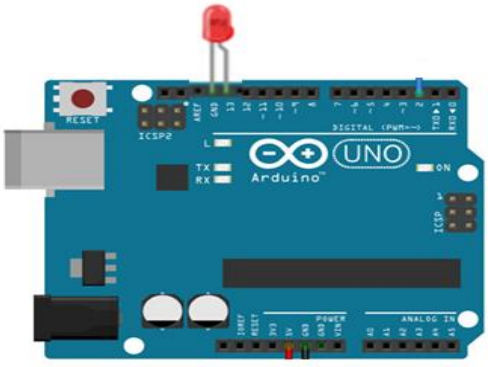
APPARATUS REQUIRED:

S.No	Name of the equipment	Specifications	Quantity
1	Arduino	UNO	1
2	LED		1
3	Resistor	470Ω	1
4	Jumper wires		As per requirement
5	Breadboard		1
6	Computer with Arduino IDE software	IDE 1.8.14	1

PROCEDURE:

1) Build the circuit according to the schematic diagram
2) Use Arduino Uno on the Arduino Desktop IDE.
3) Select board type and port.
4) Write and upload the program in the Arduino-IDE.
5) open the serial monitor by,
    a. Press ctrl+shift+M
    b. Tools/Serial Monitor
    c. Or click the magnifier icon in the top right corner of the Arduino IDE.
6) Use Decimal 1 for LED ON and decimal 0 for LED OFF

SCHEMATIC DIAGRAM:



THEORY:

A Serial communication is a method of transmitting data as a train of 1s and 0s, i.e. only one bit for a clock cycle of binary coded data. Or one bit at a time, sequentially on a communication channel. It is a contrast to parallel communication, whereas in parallel system multiple bits are sent through several parallel paths. In a parallel system number of bits transmitted per clock is equal to the number of parallel paths. That is, a 4-bit channel can transmit any of the 16 binary states for a single clock cycle. In a serial communication to transfer a byte, the data is transferred as a sequence of 8 bits as one by one. Each bit is either high state 1s or low state 0s.

In Arduino boards, the serial connection can be made either via serial port (type B USB) or by digital pins 0 (RX) and 1 (TX). An Arduino IDE includes a serial monitor, a built-in terminal to communicate with an Arduino board.

PROGRAMME:

```
void setup(){
  pinMode(13,OUTPUT);           // Configure 13 pin as OUTPUT pin
  Serial.begin(9600);           // opens serial port, sets data rate to 9600 bps
}
// Enter value 1 or 0 to control LED ON / OFF state
void loop(){
  if(Serial.available()>0){
    charstate=Serial.read();        //Reads incoming serial data
    if(state=='1'){
digitalWrite(13,HIGH);              //Sets the LED ON
Serial.println("LED ON");           // Prints data to the serial port
    }
    if(state=='0'){
      digitalWrite(13,LOW);         //Sets the LED OFF
Serial.println("LED OFF");          // Prints data to the serial port
    }
  }
  delay(1000);              //waits for 1 second
}
```

OBSERVATIONS:

S.No.	Input from Serial Monitor	LED Status
1.	0	OFF
2.	1	ON

RESULT: Hence LED has been controlled using commands from serial monitor.

Expt.No: 7	FABRICATION AND DIRECTION CONTROL OF WHEELED ROBOT USING ARDUINO
Dt:	

AIM:

To fabricate wheeled robot and control its direction using Arduino.

EXPERIMENTAL REQUIREMENTS:

S.No	Name of the equipment	Specifications	Quantity
1	Arduino	UNO	1
2	Motor driver	L298N	1
3	Wheel Robo set		1
4	Jumper wires		As per requirement
5	Rechargeable Battery 9-12 v		1
6	Bread board		1
9	Computer with Arduino IDE software	IDE 1.8.14	1

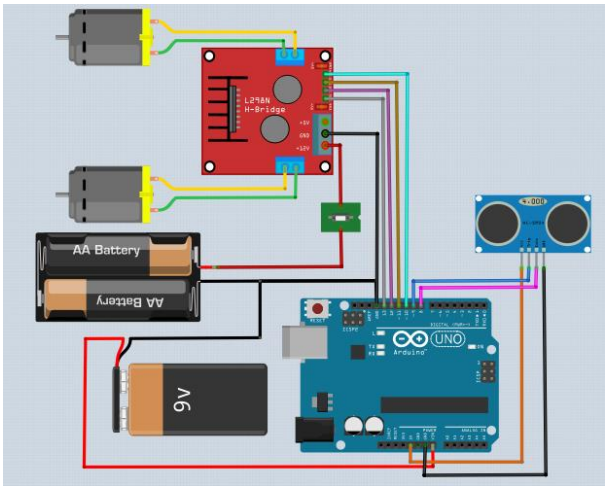
THEORY:

Obstacle Avoiding Robot is an intelligent device that can automatically sense the obstacle in front of it and avoid them by turning itself in another direction. This design allows the robot to navigate in an unknown environment by avoiding collisions.

PROCEDURE:

1) Assemble the circuit as shown in the schematic given below.
2) Use Arduino Uno on the Arduino Desktop IDE.
3) Select board type and port.
4) Write and upload the program in the Arduino-IDE.
5) Turn on the toggle switch and watch the robot avoiding obstacles.

SCHEMATIC DIAGRAM:



PROGRAMME:

```
inttrigPin = 9;      // trig pin of HC-SR04
intechoPin = 8;     // Echo pin of HC-SR04
intrevleft = 10;       //REVerse motion of Left motor
intfwdleft = 11;       //ForWarD motion of Left motor
intrevright = 12;      //REVerse motion of Right motor
intfwdright = 13;      //ForWarD motion of Right motor
long duration, distance;
void setup() {
  // delay(random(500,2000));   // delay for random time
Serial.begin(9600);
pinMode(revleft, OUTPUT);      // set Motor pins as output
pinMode(fwdleft, OUTPUT);
pinMode(revright, OUTPUT);
pinMode(fwdright, OUTPUT);
pinMode(trigPin, OUTPUT);         // set trig pin as output
pinMode(echoPin, INPUT);          //set echo pin as input to capture reflected waves
}
void loop() {
digitalWrite(trigPin, LOW);
delayMicroseconds(2);
digitalWrite(trigPin, HIGH);     // send waves for 10 us
delayMicroseconds(10);
duration = pulseIn(echoPin, HIGH); // receive reflected waves
distance = duration / 58.2;   // convert to distance
delay(10);
```

```
    // If you dont get proper movements of your robot then alter the pin numbers
if (distance > 19)
  {
digitalWrite(fwdright, HIGH);                  // move forward
digitalWrite(revright, LOW);
digitalWrite(fwdleft, HIGH);
digitalWrite(revleft, LOW);
  }
if (distance < 18)
  {
digitalWrite(fwdright, LOW);  //Stop
digitalWrite(revright, LOW);
digitalWrite(fwdleft, LOW);
digitalWrite(revleft, LOW);
delay(500);
digitalWrite(fwdright, LOW);      //move backward
digitalWrite(revright, HIGH);
digitalWrite(fwdleft, LOW);
digitalWrite(revleft, HIGH);
delay(500);
digitalWrite(fwdright, LOW);  //Stop
digitalWrite(revright, LOW);
digitalWrite(fwdleft, LOW);
digitalWrite(revleft, LOW);
delay(100);
digitalWrite(fwdright, HIGH);
digitalWrite(revright, LOW);
digitalWrite(revleft, LOW);
digitalWrite(fwdleft, LOW);
delay(500);
  }
}
```

OBSERVATIONS:

Obstacle Avoiding Robot automatically senses the obstacle in front of it and avoids them by turning itself in another direction by avoiding collisions.

RESULT:

Hence wheeled robot is fabricated and controlled its direction using Arduino to avoid collisions.

Expt.No: 8 Dt:	WIRELESS MODULE INTERFACE –Wi-Fi

AIM:

- To control LED through app using smart phone and NodeMCU.

EXPERIMENTAL REQUIREMENTS:

S.No	Name of the equipment	Specifications	Quantity
1	Arduino	UNO	1
2	NodeMCU with ESP8266		1
3	LED		1
4	Jumper wires		As per requirement
5	Breadboard		1
6	Computer with Arduino IDE software	IDE 1.8.14	1
7	Mobile with Bluetooth Terminal App		

PROCEDURE:

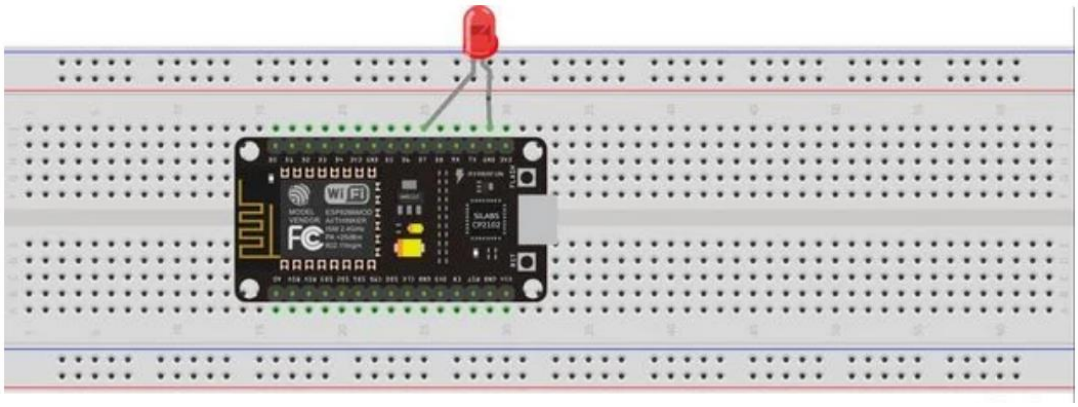
1) Open Arduino IDE. Install the required libraries for NodeMCU.

File→ Preferences→ Additional Boards Manager URLs:

http://arduino.esp8266.com/stable/package_esp8266com_index.json

2) Create a new file and write the required program.
3) Save the file and check for the errors by compiling it.
4) Build the circuit according to the schematic diagram.
5) Set the board and port number. Upload the program to the NodeMCU.
6) Copy the IP address in the Serial Monitor and paste it in the Web browser.
7) Control LED using ON & OFF buttonsfrom Web browser.

SCHEMATIC DIAGRAM:



PROGRAMME:

```
#include <ESP8266WiFi.h>
const char* ssid = "";    //  Your Wi-Fi Name
const char* password = "";   // Wi-Fi Password
int LED = 2;   // led connected to GPIO2 (D4)
WiFiServer server(80);
void setup()
{
Serial.begin(115200); //Default Baudrate
pinMode(LED, OUTPUT);
digitalWrite(LED, LOW);
Serial.print("Connecting to the Newtork");
WiFi.begin(ssid, password);
while (WiFi.status() != WL_CONNECTED)
  {
delay(500);
Serial.print(".");
  }
Serial.println("WiFi connected");
server.begin();  // Starts the Server
Serial.println("Server started");
Serial.print("IP Address of network: "); // will IP address on Serial Monitor
Serial.println(WiFi.localIP());
Serial.print("Copy and paste the following URL: https://"); // Will print IP address in URL format
```

```
Serial.print(WiFi.localIP());
Serial.println("/");
}
void loop()
{
WiFiClient client = server.available();
if (!client)
  {
return;
  }
Serial.println("Waiting for new client");
while(!client.available())
  {
delay(1);
  }
  String request = client.readStringUntil('\r');
Serial.println(request);
client.flush();
int value = LOW;
if(request.indexOf("/LED=ON") != -1)
  {
digitalWrite(LED, HIGH); // Turn LED ON
value = HIGH;
  }
if(request.indexOf("/LED=OFF") != -1)
  {
digitalWrite(LED, LOW); // Turn LED OFF
value = LOW;
  }
}
```

//*----------HTML Page Code----------*//

```
client.println("HTTP/1.1 200 OK"); //
client.println("Content-Type: text/html");
client.println("");
client.println("<!DOCTYPE HTML>");
```

```
client.println("<html>");
client.print(" CONTROL LED: ");
if(value == HIGH)
  {
client.print("ON");
  }
else
  {
client.print("OFF");
  }
client.println("<br><br>");
client.println("<a href=\"/LED=ON\"\"><button>ON</button></a>");
client.println("<a href=\"/LED=OFF\"\"><button>OFF</button></a><br />");
client.println("</html>");
delay(1);
Serial.println("Client disonnected");
Serial.println("");
}
```


OBSERVATIONS:

Copy the URL on your mobile, LED is turned ON & OFF through the buttons available on mobile.

RESULT:

Hence LED has been controlled through smart phone using NodeMCU.

Expt.No:9	BASIC ANDROID APP DEVELOPMENT USING MIT APP INVENTOR
Dt:	

AIM:

- To develop Basic Android App using MIT App Inventor

EXPERIMENTAL REQUIREMENTS:

S.No	Name of the equipment	Specifications	Quantity
1	NodeMCU	ESP8266	1
2	Computer with wi-fi connectivity		1
3	Mobile phone with MIT app &wi-fi connectivity		1

PROCEDURE:

1. First go to the **MIT Application Inventor** website: http://ai2.appinventor.mit.edu/
2. Then click '**Create Applications**' in the top left corner
3. Now click on '**Projects**'on the next screen and then '**Start a new project**'.
4. Now click on '**Button**'and drag and drop two buttons on the main screen. You can enter your favorite name in the button from the options on the right.
5. Then click '**Connectivity**'and drag and drop the web component to the main screen.
6. Click '**Blocks** 'now to add blocks to your application.
7. Now click on **button 1 in the block menu** and then **click on the marked red option**.
8. After this click on **Web 1**. Scroll down and select the red marked block.
9. Now click on the **text menu and choose the first option**. Enter your **URL** in the text menu.
10. Then click on **Web 1 again and then select the marked red option.**
11. Follow the same procedure for '**Button 2′**.
12. Now that the **app** is ready to download, click on 'Build' to get the simple apk file. Also, there are two options to **download the app APK, by QR code and directly on PC, then install it on Android.**
13. Now your app is ready, and you can control the lighting using the **ON-OFF** button presented in the app.
14. Now we have to upload the code to NodeMCU to create a simple HTTP web server for controlling home applications. We will use the HTTP GET method for communicating between ESP8266 and Android applications.

PROGRAMME:

```
#include<ESP8266WiFi.h>
const char* ssid = "xxxxxxxx";
const char* password = "xxxxxxxxxx";
```

Serial Monitor is started at the default Baud Rate for NodeMCU

```
Serial.begin(115200);
```

Relay Pin is defined to NodeMCU D4 pin i.e. GPIO pin 2.

```
pinMode(2, OUTPUT);
digitalWrite(2, 0);
```

In the void setup function, the function will try to connect to WiFi. This process executes in a loop, which means it runs until there is a connection to WiFi. So be careful before entering your WiFi SSID and Password.

```
void setup() {
// Connect to WiFi network
Serial.println();
Serial.println();
Serial.print("Connecting to ");
Serial.println(ssid);
WiFi.begin(ssid, password);
while(WiFi.status() != WL_CONNECTED){
delay(500);
Serial.print(".");
}
Serial.println(" ");
Serial.println(" WiFi connected");
}
```

In void loop, it check if a client has connected. It Wait until the client sends some data and performs tasks according to input.

```
void loop() {
WiFiClient client = server.available();
if(!client){
return ;
}
Serial.println("new client");
while(!client.available()){
```

delay(1);

}

String req = client.readStringUntil('\r');

}

Now navigate through your browser to check if your web server is working perfectly. Use the following URLs to turn your light **ON or OFF**.

http://192.168.1.40/gpio/1

http://192.168.1.40:/gpio/0

***Note: 192.168.1.40** is the **IP address of NodeMCU**. You can find the IP address of your NodeMCU on **Serial Monitor**. When you **run the code on the Arduino IDE**, it prints your device's IP address on the serial monitor. Therefore, **it will confirm whether the webserver is working or not**.*

RESULT:

Hence Android App is developed using MIT App Inventor

Expt.No: 10	SMART HOME ANDROID APP DEVELOPMENT USING APP INVENTOR AND NODE MCU
Dt:	

AIM:

- To develop Home Automation with MIT App Inventor and ESP8266.

EXPERIMENTAL REQUIREMENTS:

S.No	Name of the equipment	Specifications	Quantity
1	NodeMCU	ESP8266	1
2	RGB common cathode LED		1
3	Jumper wires		As per requirement
4	Breadboard		1
5	Computer with Arduino IDE software&wi-fi connectivity	IDE 1.8.14	1
6	Mobile phone with MIT app &wi-fi connectivity		1

THEORY:

ThingSpeak for IoT Projects:

ThingSpeak is an IoT analytics platform service that allows user to aggregate, visualize, and analyze live data streams in the cloud. User can send data to ThingSpeak from devices, create instant visualization of live data, and send alerts.

PROCEDURE:

I. Make circuit and Write the code

1) Open Arduino IDE. Install the required libraries for Node MCU.

   File→ Preferences→ Additional Boards Manager URLs:

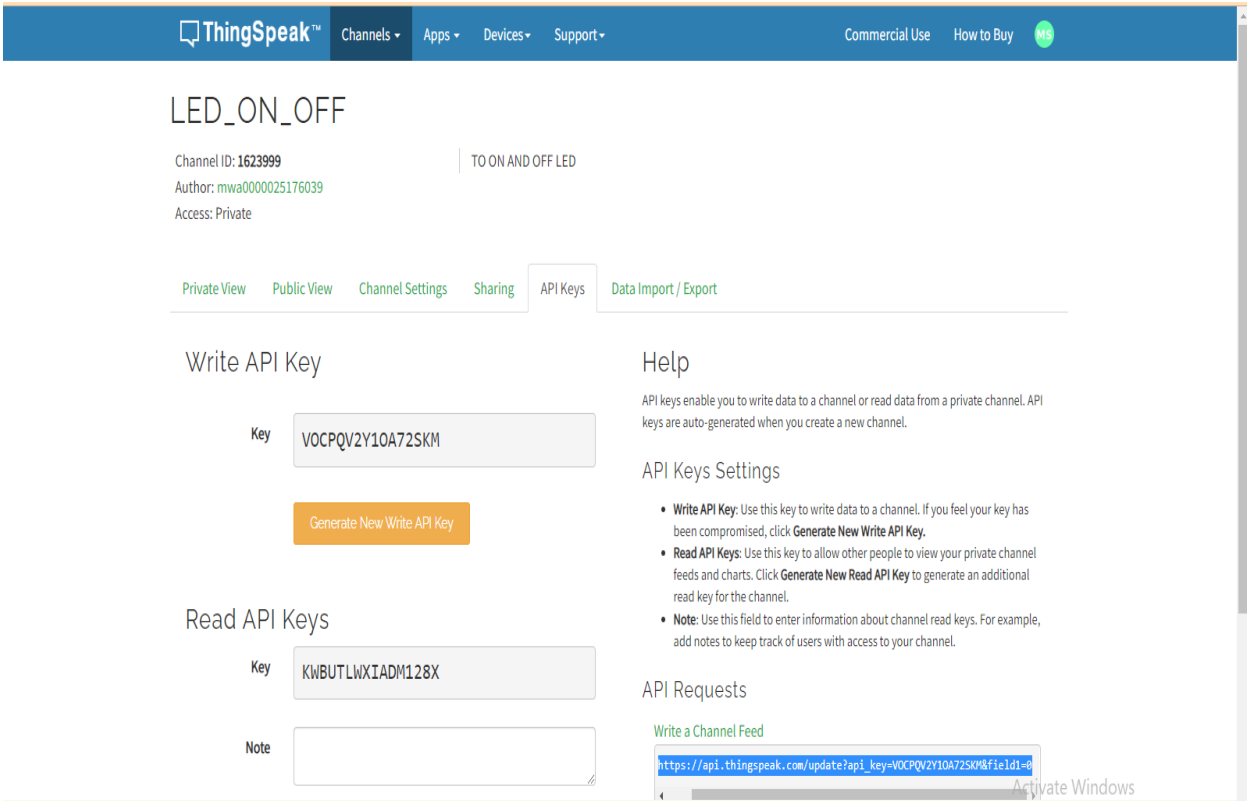
   http://arduino.esp8266.com/stable/package_esp8266com_index.json

2) Create a new file and write the required program .
3) Save the file and check for the errors by compiling it.
4) Connect the circuit as per the circuit diagram.
5) Set the board and port number. Upload the program to the Node MCU.

I. Make Server using ThingSpeak server.

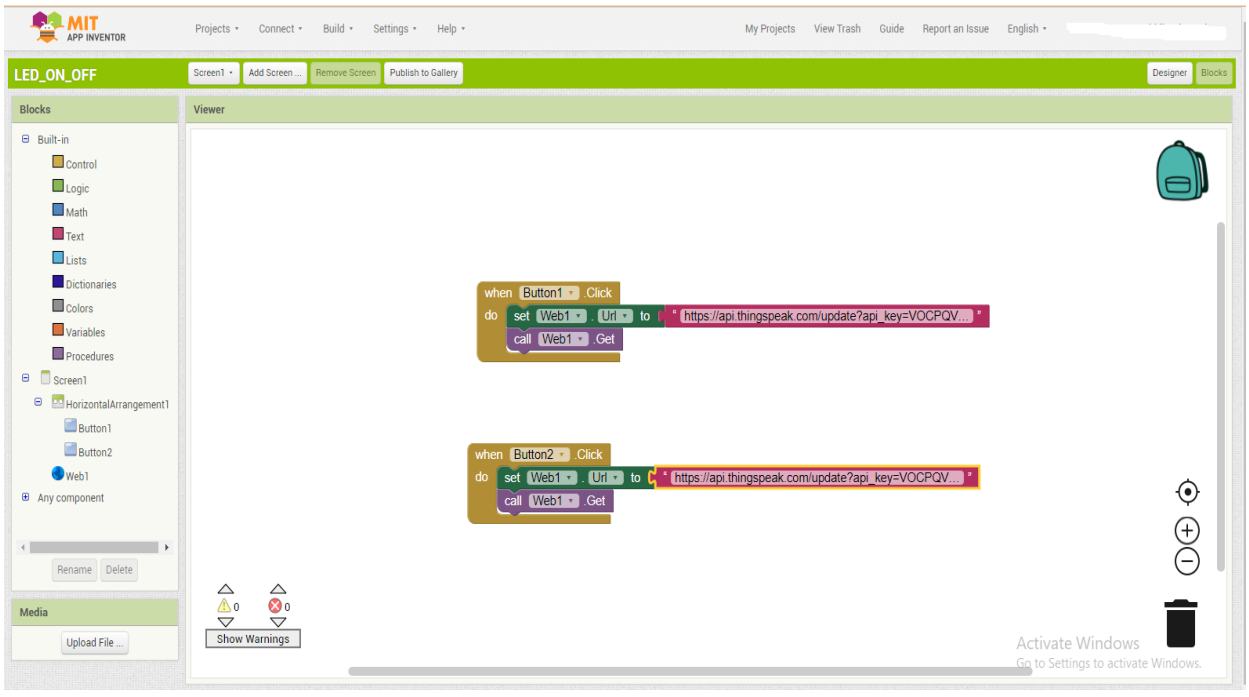
1. Sign In to ThingSpeak using your MathWorks Account credentials, or create a new account.
2. Create a Channel

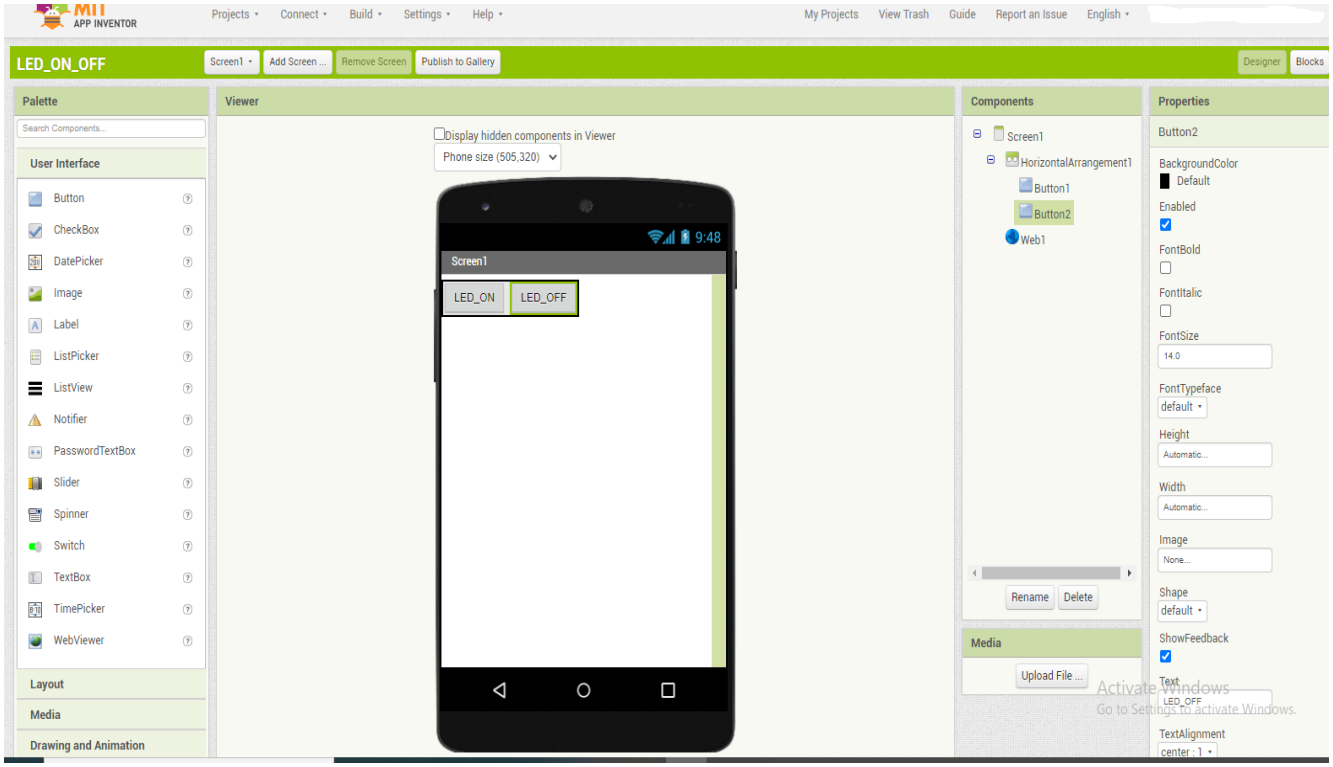
   Click **Channels** > **MyChannels**.
3. On the Channels page, click **New Channel**.
4. Check the boxes next to Fields 1–3. Enter these channel setting values:
   - **Name**:
   - **Field 1**:
5. Click **Save Channel** at the bottom of the settings and get the channel ID & API key.



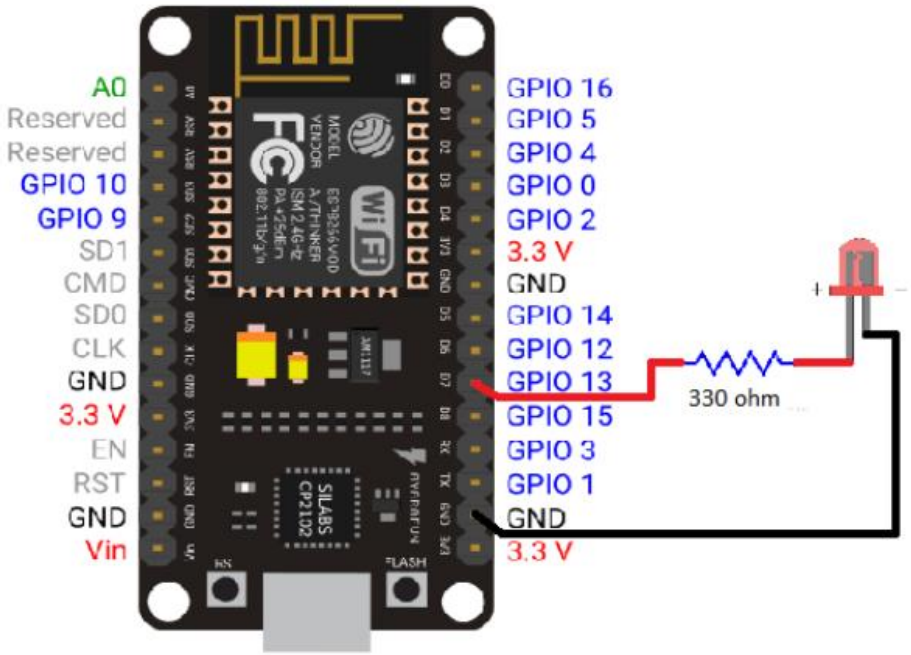
III. Develop Mobile App using MIT App Inventer.

1. Log in to the **MIT Application Inventor** website: http://ai2.appinventor.mit.edu/
2. Then click '**Create Applications**' in the top left corner
3. Now click on '**Projects**' on the next screen and then '**Start a new project**'.
4. Now click on '**Button'** and drag and drop two buttons on the main screen. You can enter your favourite name in the button from the options on the right.
5. Then click '**Connectivity'** and drag and drop the web component to the main screen.
6. Click '**Blocks'** now to add blocks to your application.
7. Now click on **button 1 in the block menu** and then **click on the marked red option**.
8. After this click on **Web 1**. Scroll down and select the red marked block.
9. Now click on the **text menu and choose the first option**. Enter your **URL** in the text menu.
10. Then click on **Web 1 again and then select the marked red option.**
11. Follow the same procedure for **'Button 2'**.
12. Now that the **app** is ready to download, click on 'Build' to get the simple apk file. Also, there are two options to **download the app APK, by QR code and directly on PC, then install it on Android.**
13. Now the app is ready, and it can control the LED using the **ON-OFF** button presented in the app.





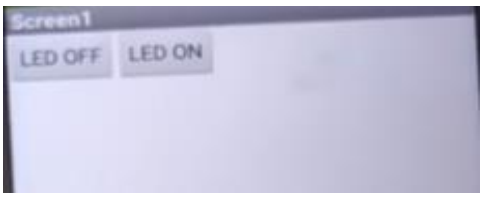
SCHEMATIC DIAGRAM:



PROGRAMME:

```
#include <ThingSpeak.h>              // add library
#include <ESP8266WiFi.h>
WiFiClient  client;
unsigned long counterChannelNumber = 980157;              // Channel ID
const char * myCounterReadAPIKey = "CL8DHB4NZ43291ZI"; // Read API Key
constint FieldNumber1 = 1;                          // The field you wish to read
constint FieldNumber2 = 2;                          // The field you wish to read
void setup(){
pinMode(13,OUTPUT);
Serial.begin(115200);
Serial.println();
WiFi.begin("POCO PHONE", "9004652173");// write wifi name & password
Serial.print("Connecting");
while (WiFi.status() != WL_CONNECTED)
  {
delay(500);
Serial.print(".");
  }
Serial.println();
Serial.print("Connected, IP address: ");
Serial.println(WiFi.localIP());
ThingSpeak.begin(client);
}
void loop(){
int A = ThingSpeak.readLongField(counterChannelNumber, FieldNumber1,
myCounterReadAPIKey);
Serial.println(A);
digitalWrite(13,A);
}
```

OBSERVATIONS:



RESULT:

Android App for Smart Home Automation has been developed using MIT App Inventor.

Expt.No:	WIRELESS MODULE INTERFACE – BLUETOOTH
Dt:	

AIM:

- To transfer data between Arduino UNO and a device using Bluetooth module HC-05.
- To control LED through app using smart phone and NodeMCU.

EXPERIMENTAL REQUIREMENTS:

S.No	Name of the equipment	Specifications	Quantity
1	Arduino	UNO	1
3	Bluetooth Module	HC-05	1
4	LED		1
6	Jumper wires		As per requirement
7	Breadboard		1
8	Computer with Arduino IDE software	IDE 1.8.14	1
9	Mobile with Bluetooth Terminal App		

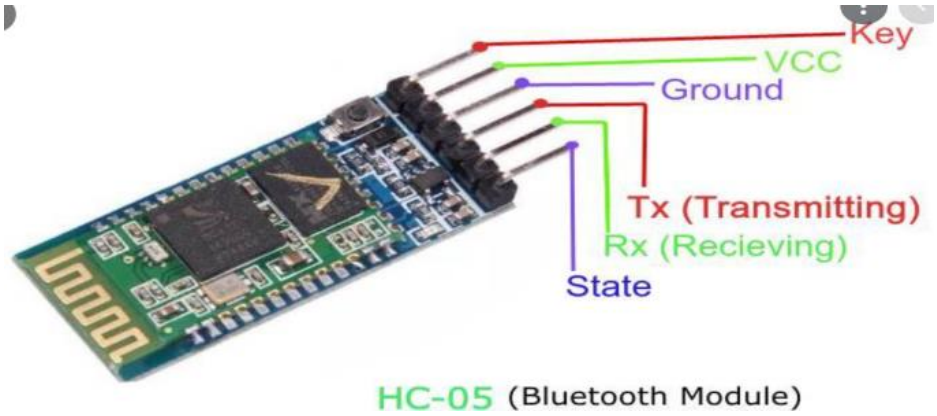
PROCEDURE:

1) Build the circuit according to the schematic diagram
2) Use Arduino Uno on the Arduino Desktop IDE.
3) Select board type and port.
4) Write and upload the program in the Arduino-IDE.
5) Download and install a **Bluetooth terminal** application on phone and use it to connect to the HC-05 Bluetooth module.
6) Data is sent from the Smartphone using the **Bluetooth terminal** application.

Theory:

BLUETOOTH MODULE INTERFACE:

Bluetooth is a one of the wireless technology. HC-05 is a Bluetooth module which can communicate in two way. It is full-duplex. It be used with most micro controllers. Because it operates Serial Port Protocol (SSP). The module communicates with the help of USART (Universal Synchronous/ Asynchronous Receiver/Transmitter) at the baud rate of 9600 and it also support other baud rate. This module can be interfaced with any microcontroller which supports USART. The HC-05 operates in two modes. One is Data mode and other is AT command mode. When the Enable/Key pin is "LOW" the HC-05 is in Data Mode. If that pin set as "HIGH" the module is in AT command mode.



HC-05 (Bluetooth Module)

Enable - This pin is used to set the Data Mode or and AT command mode (set high).

VCC - This is connected to +5V power supply.

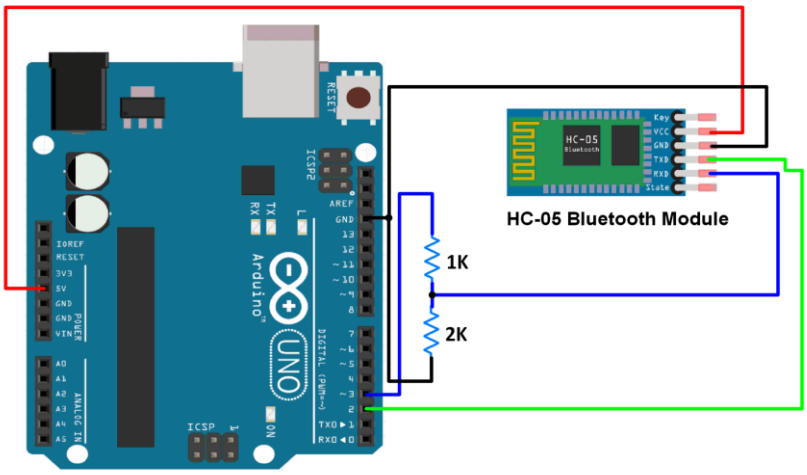
Ground - Connected to ground of powering system.

Tx (Transmitter) - This pin transmits the received data Serially.

Rx (Receiver) - Used for broadcasting data serially over bluetooth.

State -Used to check if the Bluetooth is working properly.

SCHEMATIC DIAGRAM:



PROGRAMME:

```
#include<SoftwareSerial.h>
/* Create object named bt of the class SoftwareSerial */
SoftwareSerial bt(2,3); /* (Rx,Tx) */
void setup() {
bt.begin(9600);          /* Define baud rate for software serial communication */
Serial.begin(9600);      /* Define baud rate for serial communication */
}
void loop() {
if (bt.available())      /* If data is available on serial port */
  {
Serial.write(bt.read()); /* Print character received on to the serial monitor */
  }
}
```

OBSERVATIONS:

A message WELCOME is transmitted from Smartphone via Bluetooth to the Arduino Uno and displayed it on Serial Monitor of PC.

RESULT:

Hence a message has been transmitted from Smartphone via Bluetooth to the Arduino