**1.Digital I/O Interface - Multicolor LED, IR Sensor, PIR, Slot Sensor.**

AIM:

* To interface and control RGB LED with Arduino.
* To Interface a IR and PIR sensor with Arduino to detect motion and display "Stop" or "Go" depending on the output from the sensor.
* To Interface a slot sensor with Arduino to detect object and control LED based on sensor output.

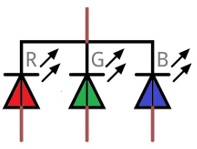
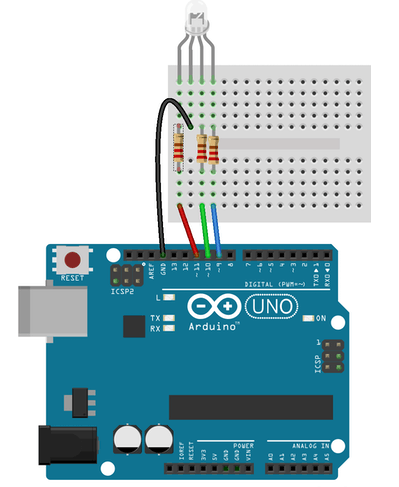
**APPPARATUS REQUIRED:**

| **S.No** | **Name of the equipment** | **Specifications** | **Quantity** |
| --- | --- | --- | --- |
| 1 | Arduino | UNO | 1 |
| 2 | RGB common cathode LED |  | 1 |
| 3 | Passive Infrared (PIR)Sensor |  | 1 |
| 4 | IR Sensor |  | 1 |
| 5 | Slot Sensor |  | 1 |
| 6 | Resistors | 220ohms | 3 |
| 7 | Jumper wires |  | required |
| 8 | Breadboard |  | 1 |
| 9 | 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

**SCHEMATIC DIAGRAM:**



### PROGRAMME:

int red = 2, green = 3, blue = 4;

void setup() {

pinMode(2, OUTPUT);

pinMode(3, OUTPUT);

pinMode(4, OUTPUT);

}

void loop() {

RGB(1,0,0); // RED

delay(1000);

RGB(0,1,0); // GREEN

delay(1000);

RGB(0,0,1); // BLUE

delay(1000);

}

void RGB(int x, int y, int z){

digitalWrite(red, x);

digitalWrite(green, y);

digitalWrite(blue, z);

}

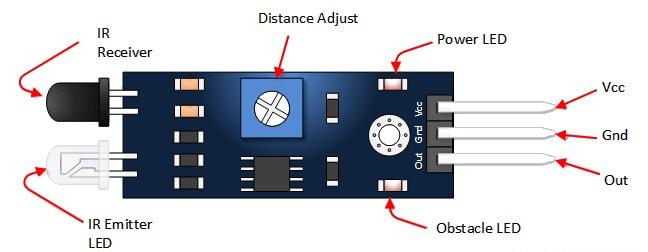
**OBSERVATIONS:**

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

# IR SENSOR

#### **PINOUT:**

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



**SCHEMATIC DIAGRAM:**

### 

### PROGRAMME:

int ir = 2; // connect ir sensor to arduino pin 2

int led = 3; // conect Led to arduino pin 3

void setup(){

pinMode (ir, INPUT); // sensor pin INPUT

pinMode (led, OUTPUT); // Led pin OUTPUT

}

void loop(){

int statusSensor = digitalRead(ir);

if (statusSensor == 1){

digitalWrite(led, 0); // LED LOW

}

else{

digitalWrite(led, 1); // LED High

}

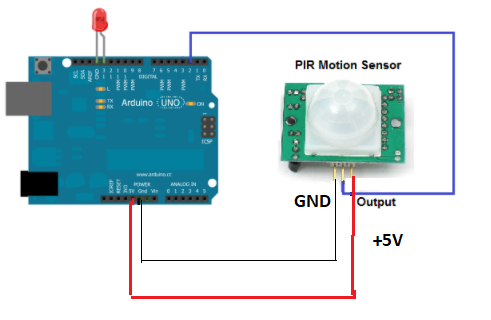
}

**OBSERVATIONS:**

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

### Passive Infrared (PIR) Sensor

**SCHEMATIC DIAGRAM:**



### PROGRAMME:

int pirSensor = 2;

int led = 13;

void setup() {

pinMode(pirSensor, INPUT);

pinMode(led, OUTPUT);

}

void loop() {

int sensorValue = digitalRead(pirSensor);

if(sensorValue == 1){

digitalWrite(led, 1);

}

else{

digitalWrite(led, 0);

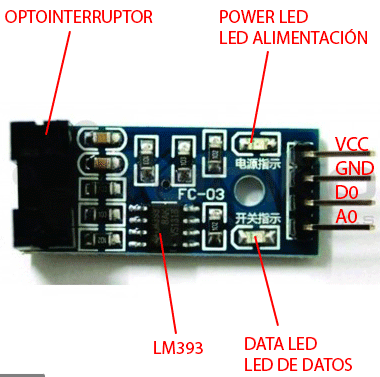
}

}

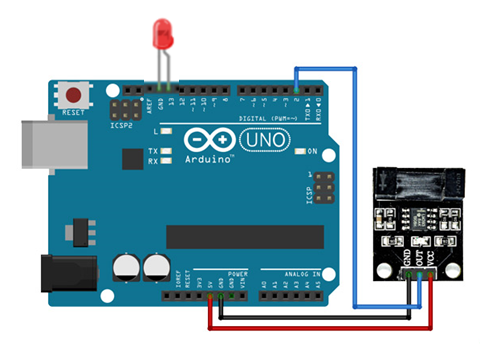
**OBSERVATIONS:**

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

**LM393 Speed Sensor Module (H206)**



**SCHEMATIC DIAGRAM:**



### PROGRAMME:

int slotSensor = 2;

int led = 13;

void setup() {

pinMode(slotSensor, INPUT);

pinMode(led, 13);

}

void loop() {

int value = digitalRead(slotSensor);

if(value == 1){

digitalWrite(led, 1);

}

else{

digitalWrite(led, 0);

}

}

**OBSERVATIONS:**

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

**RESULT:**

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

**Analog Read and Write - Potentiometer, Temperature Sensor, Led Brightness Control**

**AIM:**

* To control LED brightness using potentiometer with Arduino
* To observe the Temperature and display in serial monitor.

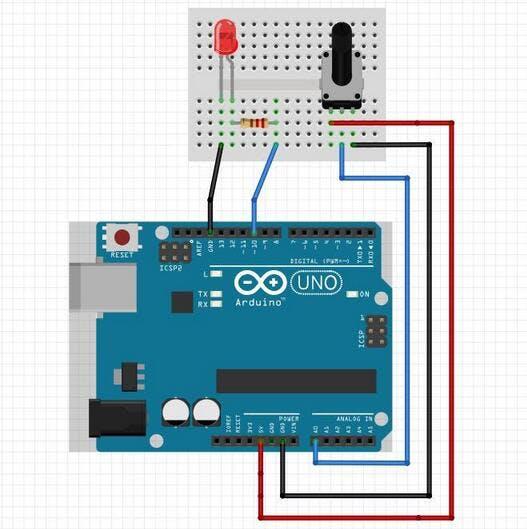
**APPPARATUS REQUIRED:**

| **S.No** | **Name of the equipment** | **Specifications** | **Quantity** |
| --- | --- | --- | --- |
| 1 | Arduino | UNO | 1 |
| 2 | Potentio Meter | 1K Ω | 1 |
| 3 | LED |  | 1 |
| 4 | Temperature Sensor | DTH11 | 1 |
| 5 | Resistor | 220ohms | 1 |
| 6 | Jumper wires |  | required |
| **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. Write and upload the program in the Arduino-IDE.
5. By varying the potentiometer observe the LED intensity.

**SCHEMATIC DIAGRAM:**



**PROGRAMME:**

int LED\_PIN = 10;

void setup() {

Serial.begin(9600);

pinMode(LED\_PIN, OUTPUT);

}

void loop() {

int analogValue = analogRead(A0);

int brightness = map(analogValue, 0, 1023, 0, 255);

analogWrite(LED\_PIN, brightness);

Serial.print("Analog: ");

Serial.print(analogValue);

Serial.print(", Brightness: ");

Serial.println(brightness);

delay(100);

}

**OBSERVATIONS:**

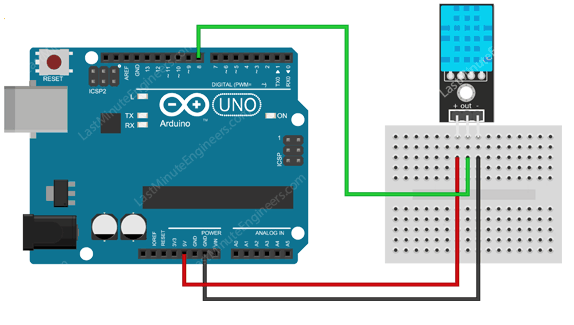
| **S.No.** | **Potentiometer Value**  **(K Ω)** | **LED Status** |
| --- | --- | --- |
| 1. | Min | ON with high brightness |
| 2. | Mid Value | ON with less brightness |
| 3. | Max | OFF |

**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. Add the required library files.
6. Observe the temperature and Humidity values

**Temperature sensor DTH11:**

**SCHEMATIC DIAGRAM:**



**PROGRAMME:**

#include <dht.h>  
dht DHT;  
#define DHT11\_PIN 8  
void setup(){  
  Serial.begin(9600);  
}  
void loop(){  
  int chk = 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 |  |  |
| 3 |  |  |

**RESULT:**

Hence

* LED brightness has been controlled using potentiometer
* Temperature and Humidity has been measured using Temperature sensor and displayed using serial monitor.

**DC MOTOR CONTROL - DC MOTOR SPEED AND DIRECTION CONTROL**

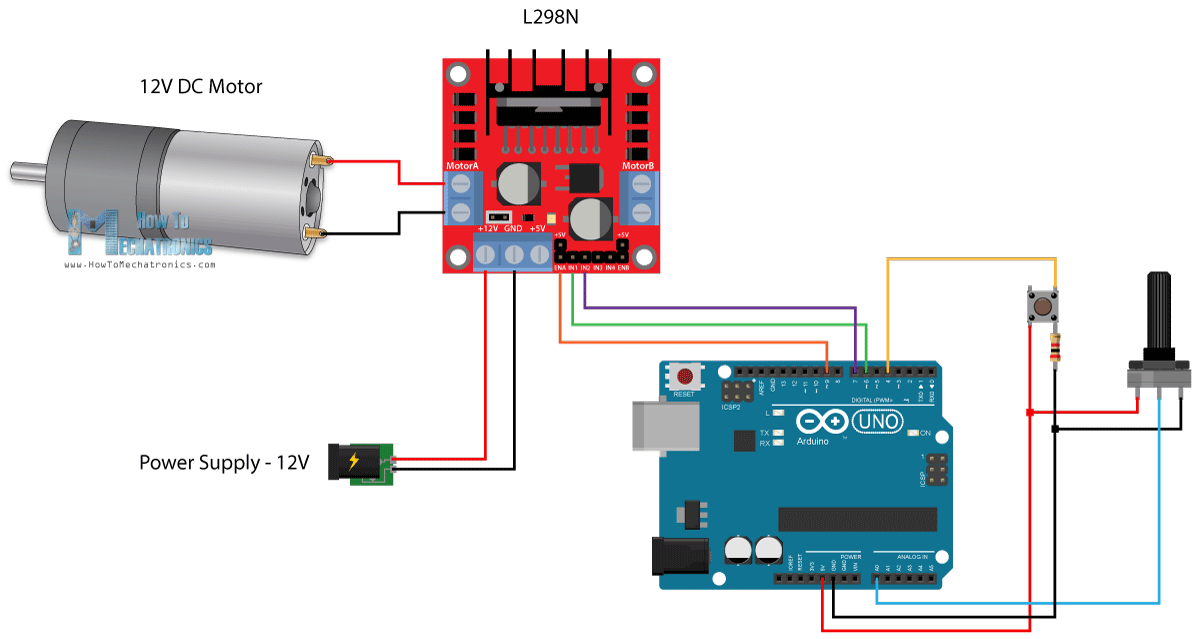
**AIM:**

To control DC Motor Speed and Direction using Arduino.

**APPPARATUS 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 | Resistor | 220ohms | 1 |
| 6 | Sparkfun push button Switch |  | 1 |
| 7 | Rechargeable Battery | 5V | 1 |
| 8 | Jumper wires |  | required |
| 9 | Breadboard |  | 1 |
| 10 | Computer with Arduino IDE software | IDE 1.8.14 | 1 |

**SCHEMATIC DIAGRAM:**



**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. Control the speed and direction of DC motor by varying potentiometer and switch.

**PROGRAMME:**

int in1 = 6, in2 = 7, enA = 9, Speed = 0, val = 0;

void setup(){

Serial.begin(9600);

pinMode(enA, OUTPUT);

pinMode(in1, OUTPUT);

pinMode(in2, OUTPUT);

Serial.begin(9600);

}

void loop(){

val = analogRead(A0); //Read potentiometer value to change the Motor speed

Speed = map(val,0,1023,0,255);

if (Serial.available() &gt; 0){

char state = Serial.read();

if (state == &#39;1&#39;){

analogWrite(enA,Speed);

digitalWrite(in1,1);

digitalWrite(in2,0);

Serial.println(&quot;FORWARD&quot;);

delay(1000);

}

if (state == &#39;0&#39;){

analogWrite(enA,Speed);

digitalWrite(in1,0);

digitalWrite(in2,1);

Serial.println(&quot;BACKWARD&quot;);

delay(1000);

}}}

**OBSERVATIONS:**

By varying the potentiometer, speed of the DC motor is controlled and each time button is pressed, it changes the rotation direction of the motor.

**RESULT:**

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

**READ DATA FROM SENSOR AND SEND IT TO A REQUESTING CLIENT**

**(USING SOCKET COMMUNICATION)**

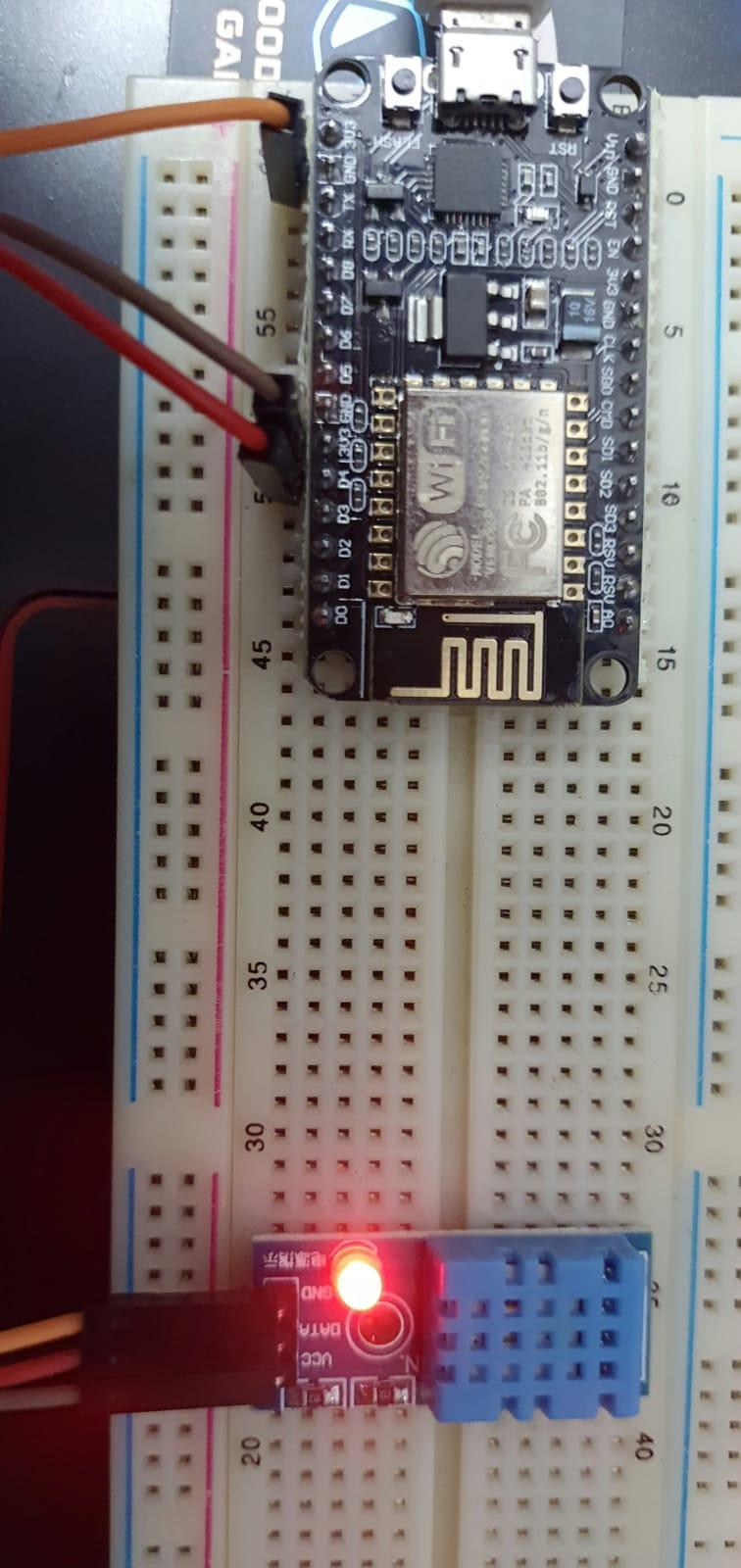
**AIM:**

To read data from sensor and send it to a requesting client (using socket communication)

**Experimental Requirements:**

| S. NO | Name of the Equipment | Specifications | Quantity |
| --- | --- | --- | --- |
| 1 | NodeMCU (ESP8266) |  | 1 |
| 2 | DHT 11 Sensor |  | 1 |
| 3 | Jumper Wires |  | 1 bunch |
| 4 | Computer with Arduino IDE software | IDE 1.8.14 |  |

**CIRCUIT CONNECTIONS:**

****

**PROCEDURE:**

1. Open Arduino IDE. Install the required libraries for NodeMCU and DHT 11 sensor from library manager.
2. Create a new file and write the required program for socket communication.
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 NodeMCU.
6. Copy the ip address in the Serial Monitor and paste it in the Web browser.
7. Observe the required output from Web browser.

**CODE:**

const int refresh=3;//3 seconds

#include "DHT.h"

#define DHTPIN 2 // GPIO 2

#define DHTTYPE DHT11 // DHT 11

DHT dht(DHTPIN, DHTTYPE);

float tValue;

float hValue;

#include <ESP8266WiFi.h>

#include <WiFiClient.h>

#include <ESP8266WebServer.h>

#include <ESP8266mDNS.h>

#ifndef STASSID

#define STASSID "USERNAME" // Your WiFi SSID

#define STAPSK "PASSWORD" //Your WiFi password

#endif

const char\* ssid = STASSID;

const char\* password = STAPSK;

ESP8266WebServer server(80);

void sendTemp() {

String page = "<!DOCTYPE html>\n\n";

page +=" <meta http-equiv='refresh' content='";

page += String(refresh);// how often temperature is read

page +="'/>\n";

page +="<html>\n";

page +="<body>\n";

page +="<h1>SOCKET COMMUNICATION</h1>\n";

page +="<p style=\"font-size:50px;\">Temperature<br/>\n";

page +="<p style=\"color:red; font-size:50px;\">";

page += String(tValue, 2);

page +="<p style=\"font-size:50px;\">Humidity<br/>\n";

page +="<p style=\"color:red; font-size:50px;\">";

page += String(hValue, 2);

page +="</p>\n";

page +="</body>\n";

page +="</html>\n";

server.send(200, "text/html",page);

}

void handleNotFound() {

String message = "File Not Found\n\n";

message += "URI: ";

message += server.uri();

message += "\nMethod: ";

message += (server.method() == HTTP\_GET) ? "GET" : "POST";

message += "\nArguments: ";

message += server.args();

message += "\n";

for (uint8\_t i = 0; i < server.args(); i++) {

message += " " + server.argName(i) + ": " + server.arg(i) + "\n";

}

server.send(404, "text/plain", message);

}

void setup(void) {

dht.begin();

Serial.begin(115200);

WiFi.mode(WIFI\_STA);

WiFi.begin(ssid, password);

Serial.println("");

// Wait for connection

while (WiFi.status() != WL\_CONNECTED)

{

delay(500);

Serial.print(".");

}

Serial.println("");

Serial.print("Connected to ");

Serial.println(ssid);

Serial.print("IP address: ");

Serial.println(WiFi.localIP());

if (MDNS.begin("CONNECTION")) {

Serial.println("MDNS responder started");

}

server.on("/", sendTemp);

server.on("/inline", []() {

server.send(200, "text/plain", "this works as well");

});

server.onNotFound(handleNotFound);

server.begin();

Serial.println("HTTP server started");

}

void loop(void) {

server.handleClient();

MDNS.update();

float c = dht.readTemperature();// Read temperature as Celsius (the default)

float h = dht.readHumidity();

Serial.print(c);

Serial.print(" ");

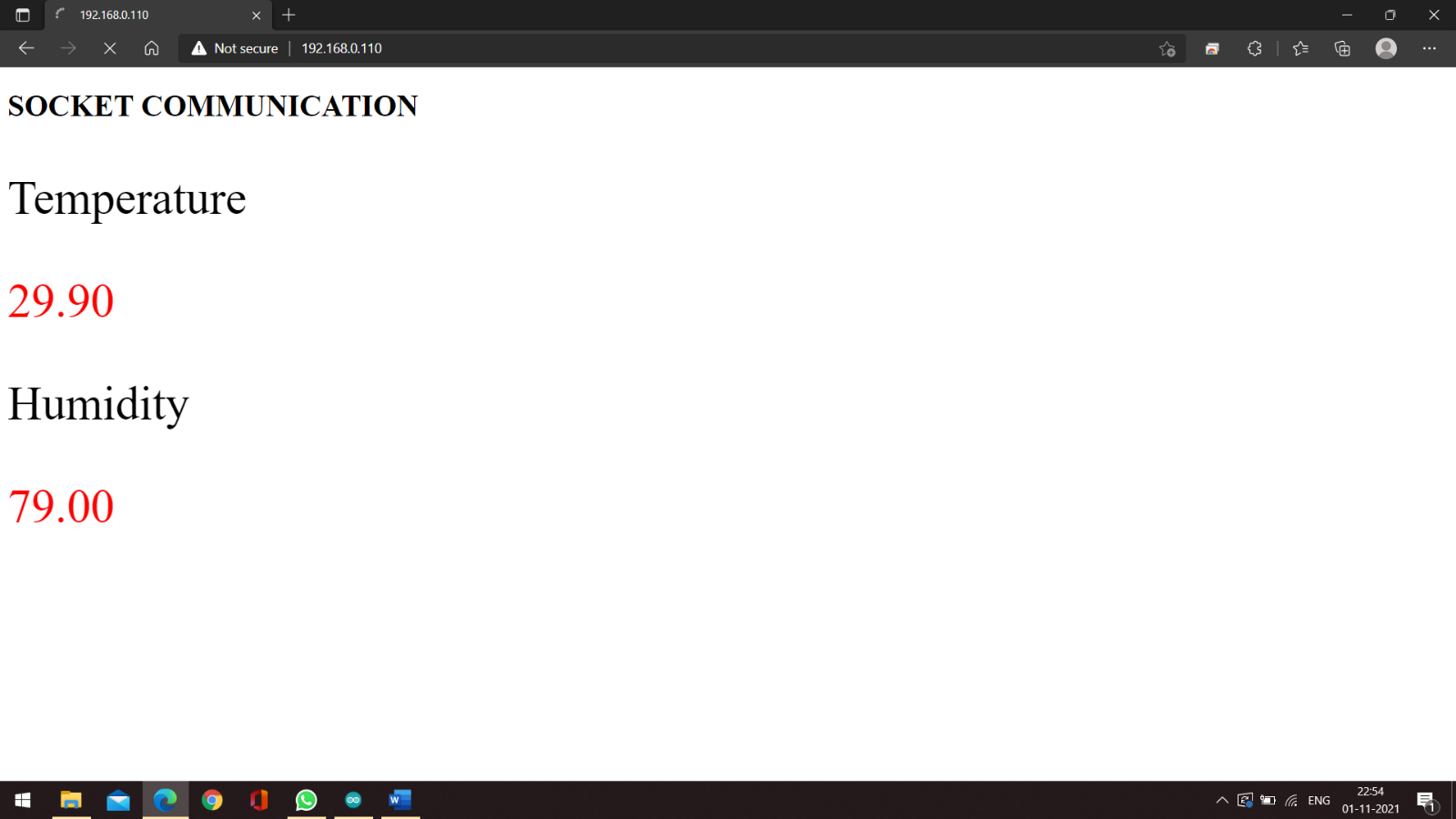
Serial.println(h);

tValue = c;

hValue = h;

delay(300);}

**OUTPUT:**

****

**RESULT:**

Hence, reading the data from DHT 11 sensor and sending it to a requesting client is performed using Socket communication.

**FABRICATION AND DIRECTION CONTROL OF WHEELED ROBOT USING ARDUINO**

**AIM:**

To fabricate wheeled robot and control its direction using Arduino.

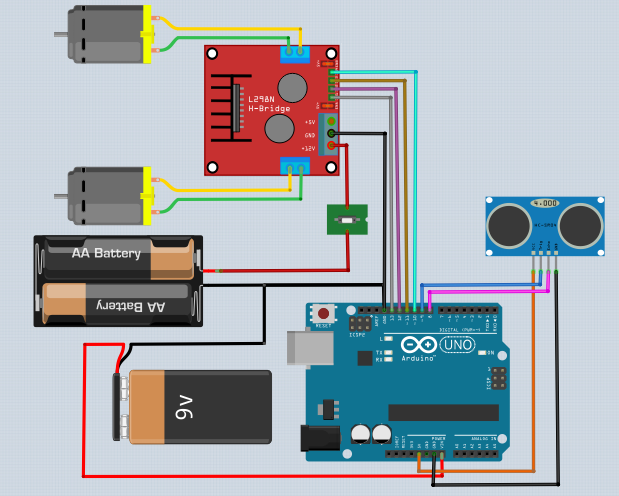
**APPPARATUS REQUIRED:**

| **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 |  | required |
| 5 | Rechargeable Battery 9-12 v |  | 1 |
| 6 | Bread board |  | 1 |
| 9 | Computer with Arduino IDE software | IDE 1.8.14 | 1 |

**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:**

int in1 = 3;

int in2 = 4;

int in3 = 5;

int in4 = 6;

int val = 0;

void setup() {

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

Serial.begin(9600);

pinMode(in1, OUTPUT);

pinMode(in2, OUTPUT);

pinMode(in3, OUTPUT);

pinMode(in4, OUTPUT);

}

void loop() {

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

if(Serial.available() > 0){

val = Serial.read();

delay(5);

if(val == 'w'){

digitalWrite(3,1);

digitalWrite(4,0);

digitalWrite(5,1);

digitalWrite(6,0);

Serial.println("Forward");

}

if(val == 'a'){

digitalWrite(3,1);

digitalWrite(4,0);

digitalWrite(5,0);

digitalWrite(6,1);

Serial.println("Left");

}

if(val == 'd'){

digitalWrite(3,0);

digitalWrite(4,1);

digitalWrite(5,1);

digitalWrite(6,0);

Serial.println("Right");

}

if(val == 's'){ digitalWrite(3,0); digitalWrite(4,1); digitalWrite(5,0); digitalWrite(6,1); Serial.println("Backward"); } if(val == 'x'){ digitalWrite(3,1); digitalWrite(4,1); digitalWrite(5,1); digitalWrite(6,1); Serial.println("Stop"); } }

}

**OBSERVATIONS:**

Obstacle Avoiding Robot automatically sense the obstacle in front of it and avoid 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.

**SERIAL COMMUNICATION - DEVICE CONTROL**

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

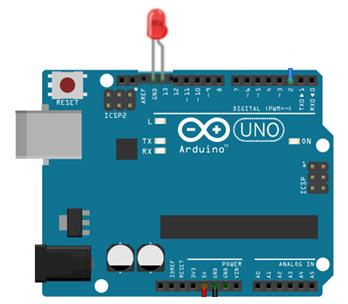
**APPPARATUS REQUIRED:**

| **S.No** | **Name of the equipment** | **Specifications** | **Quantity** |
| --- | --- | --- | --- |
| 1 | Arduino | UNO | 1 |
| 2 | LED |  | 1 |
| 3 | Resistor | 470Ω | 1 |
| 4 | Jumper wires |  | required |
| 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,
   1. Press ctrl+shift+M
   2. Tools/Serial Monitor
   3. 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:**

****

**PROGRAMME:**

// Serial Communication

int led = 12;

int val = 0;

void setup() {

Serial.begin(9600);

pinMode(led, OUTPUT);

}

void loop() {

if (Serial.available() > 0); {

val = Serial.read();

delay(5);

if (val == '1') {

digitalWrite(led, 1);

Serial.println("LED is ON");

}

if (val == '0') {

digitalWrite(led, 0);

Serial.println("LED is OFF");

}

}

}

**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.

**WIRELESS MODULE INTERFACE – BLUETOOTH AND Wi-Fi**

**AIM:**

To

* transfer data using Bluetooth with Arduino
* control LED at Receiver through Switch at Transmitter using Wi-Fi module interface with Arduino.

**APPPARATUS REQUIRED:**

| **S.No** | **Name of the equipment** | **Specifications** | **Quantity** |
| --- | --- | --- | --- |
| 1 | Arduino | UNO | 1 |
| 2 | Bluetooth Module | HC-05 | 1 |
| 3 | Wi-Fi Module | ESP8266 | 2 |
| 4 | Sparkfun push button Switch |  | 1 |
| 5 | LED |  | 1 |
| 6 | Resistor | 220 Ω |  |
| 7 | Jumper wires |  | required |
| 8 | Breadboard |  | 1 |
| 9 | Computer with Arduino IDE software | IDE 1.8.14 | 1 |
| 10 | Mobile with Blutooth Terminal App |  |  |

**BLUETOOTH MODULE INTERFACE:**

**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.

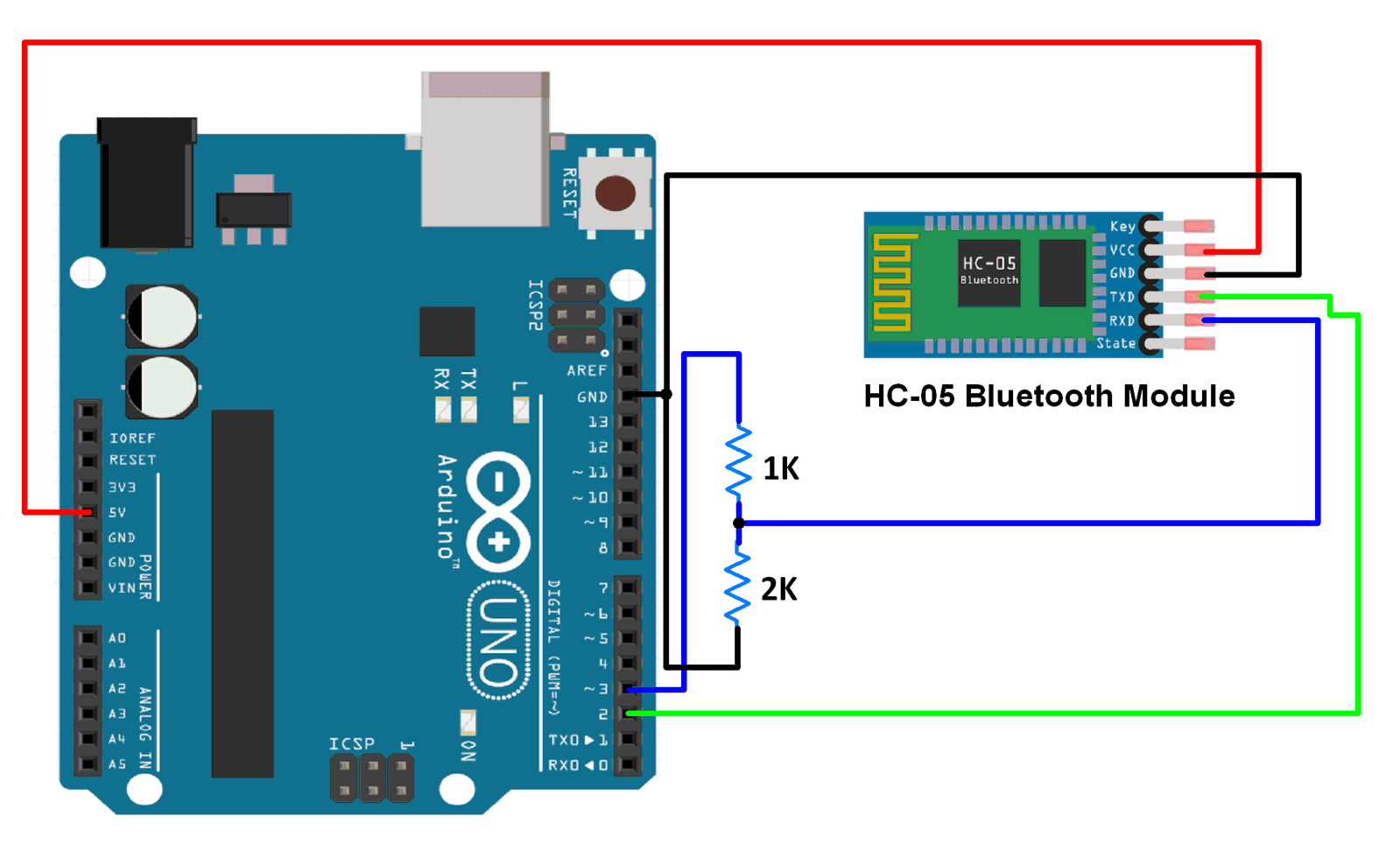
**Rx (Receiver) -** Used for broadcasting data serially over bluetooth.

**State -**Used to check if the bluetooth is working properly.

**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.

**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.

**WI-FI MODULE INTERFACE:**

**3V3**: - 3.3 V Power Pin.

**GND**: - Ground Pin.

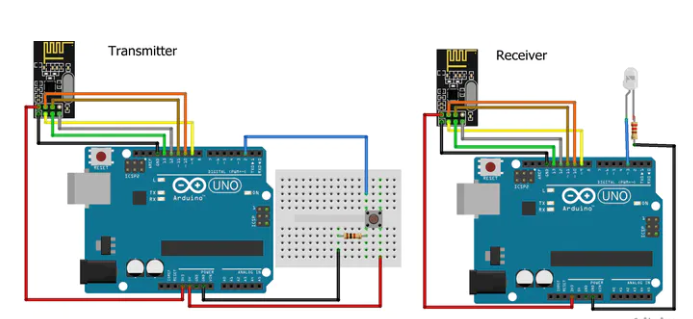
**RST**: - Active Low Reset Pin.

**EN**: - Active High Enable Pin.

**TX**: - Serial Transmit Pin of UART.

**RX**: - Serial Receive Pin of UART.

**SCHEMATIC DIAGRAM:**



**PROCEDURE:**

1. Build the circuits according to the schematic diagrams
2. Use Arduino Uno on the Arduino Desktop IDE.
3. Select board type and port for transmitter and receiver.
4. Write and upload the programs for transmitter and receiver in the Arduino-IDE.
5. Control LED at receiver by using Switch at Transmitter.

**PROGRAMME:**

#include <ESP8266WiFi.h>

const char\* ssid = "realme narzo 30"; // Your Wi-Fi Name

const char\* password = "indu1234"; // 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("");

}

**RESULT:**

Hence

* A message has been transmitted from Smartphone via Bluetooth to the Arduino
* LED at Receiver has been controlled through Switch at Transmitter using Wi-Fi module interface with Arduino.

**WIRELESS CONTROL OF WHEELED ROBOT USING WIFI**

**AIM**:  To built a simple Robot (robotic car) that can be controlled over WiFi Network i.e. the user inputs for direction of the movement of the Robot are provided through the WiFi (with the help of a simple HTML Page).

**APPPARATUS REQUIRED:**

| **S.No** | **Name of the equipment** | **Specifications** | **Quantity** |
| --- | --- | --- | --- |
| 1 | NodeMCU |  | 1 |
| 2 | L298N |  | 1 |
| 3 | Wheel Robo set |  | 1 |
| 4 | Jumper wires |  | required |
| 5 | Rechargeable Battery 9-12 v |  | 1 |
| 6 | Bread board |  |  |
| 9 | Computer with Arduino IDE software | IDE 1.8.14 | 1 |

**PROGRAMME:**

#include <ESP8266WiFi.h>

#include <WiFiClient.h>

#include <ESP8266WebServer.h>

// Replace with your network credentials

const char\* ssid = "TILAB";

const char\* password = "Arduinonano";

ESP8266WebServer server(80); //instantiate server at port 80 (http port)

String page = ""; //For the Web Server

String page2=""; //For updating Status of motor 1

String page3=""; //For updating status of motor 2

void setup(void)

{

//the HTML of the web page

page = "<center><h1>Motor Control Web Server</h1><body><p><a href=\"Forward\"><button>Forward</button></a><p><a href=\"Backward\"><button>Backward</button></a></p><p><a href =\"Left\"><button>Left</button></a>&nbsp;<a href=\"Stop\"><button>Stop</button></a><a href=\"Right\"><button>Right</button></a></p></body></center>";

pinMode(D5, OUTPUT); // inputs for motor 1

pinMode(D6,OUTPUT);

pinMode(D7,OUTPUT); // inputs for motor 2

pinMode(D8,OUTPUT);

pinMode(LED\_BUILTIN,OUTPUT); // For status of WiFi connection

delay(1000);

Serial.begin(115200);

WiFi.begin(ssid, password); //begin WiFi connection

Serial.println("");

// Wait for connection

while (WiFi.status() != WL\_CONNECTED) {

delay(500);

Serial.print(".");

}

digitalWrite(LED\_BUILTIN,HIGH); // when connected turns high

Serial.println("");

Serial.print("Connected to ");

Serial.println(ssid);

Serial.print("IP address: ");

Serial.println(WiFi.localIP()); //provides IP address

server.on("/", [](){

server.send(200, "text/html", page+page2);

});

server.on("/Forward",Forward);

server.on("/Backward",Backward);

server.on("/Left",Left);

server.on("/Right",Right);

server.on("/Stop",[](){ // turns all the motor input pins low

page2="<center><p> motor 1 Status : Off</p></center>";

page3="<center><p> motor 2 Status : off</p></center>";

server.send(200,"text/html",page+page2+page3);

digitalWrite(D5,LOW);

digitalWrite(D6,LOW);

digitalWrite(D7,LOW);

digitalWrite(D8,LOW);

delay(200);

});

server.begin();

Serial.println("Web server started!");

}

void loop(void)

{

server.handleClient();

}

void Forward()

{

digitalWrite(D5,HIGH);

digitalWrite(D6,LOW);

page2="<center><p> motor 1 Status : Forward </p></center>";

server.send(200,"text/html", page+page2+page3);

delay(200);

}

void Left()

{

page3="<center><p> motor 2 Status : Left</p></center>";

server.send(200,"text/html",page+page2+page3);

digitalWrite(D7,HIGH);

digitalWrite(D8,LOW);

delay(200);

}

void Right()

{

page3="<center><p> motor 2 Status : Right</p></center>";

server.send(200,"text/html",page+page2+page3);

digitalWrite(D8,HIGH);

digitalWrite(D7,LOW);

delay(200);

}

void Backward()

{

page2="<center><p> motor 1 Status : Backward</p></center>";

server.send(200, "text/html", page+page2+page3);

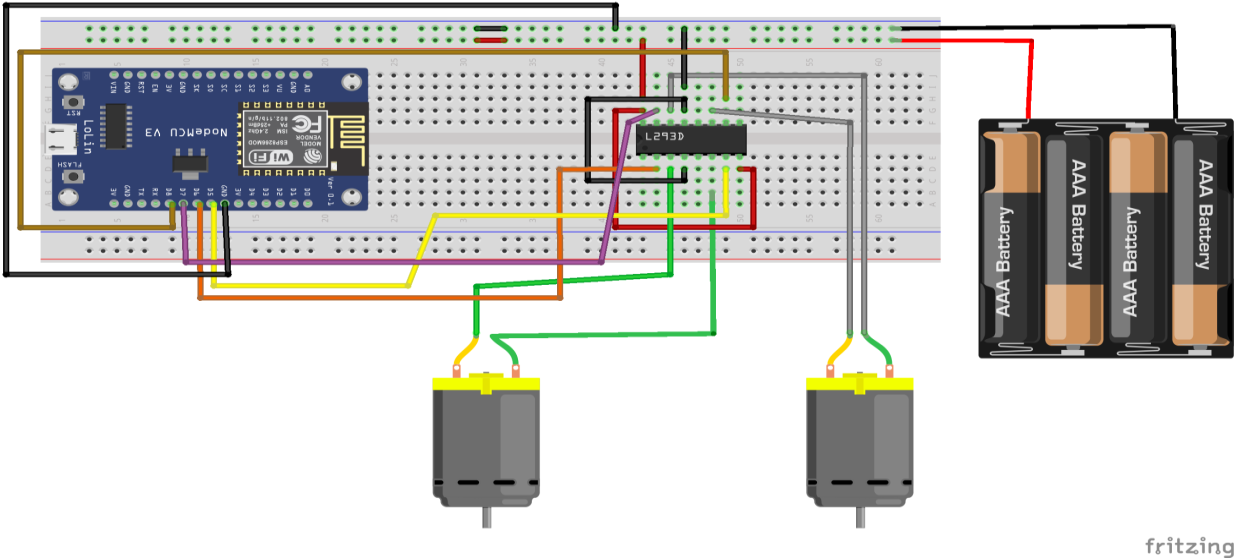
digitalWrite(D6, HIGH);

digitalWrite(D5,LOW);

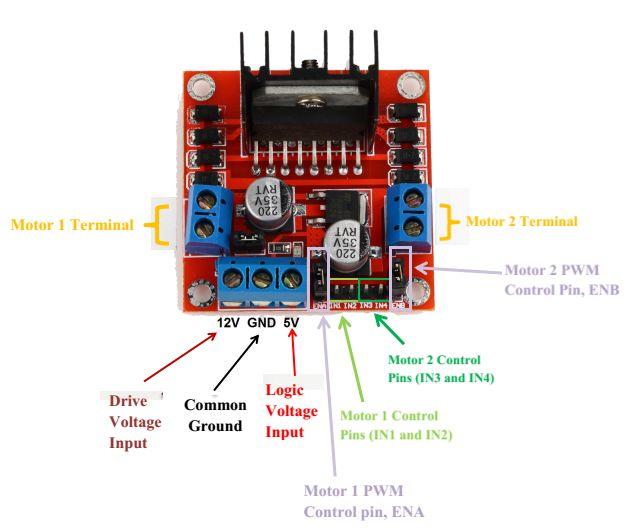
delay(200);

}

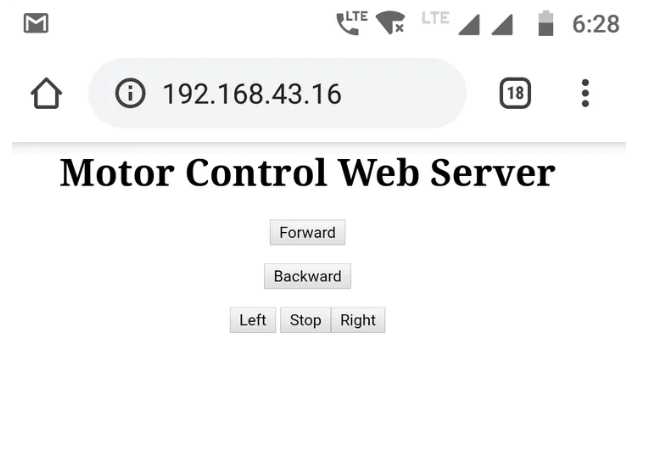
**SCHEMATIC DIAGRAM:**



L298N-Working



**WIFI enabled IP to operate ROBOT**

. **RESULT:**

Hence the WiFi Controlled Robot is controlled with the help of an HTML Web Page (which can be accessed using any web browser on a computer/Mobile phone that is connected to the same WiFi Network).

**Basic Android App Development using MIT App Inventor**

**AIM:**

* To develop Basic Android App Development using MIT App Inventor

**APPPARATUS REQUIRED:**

| **S.No** | **Name of the equipment** | **Specifications** | **Quantity** |
| --- | --- | --- | --- |
| 1 | NodeMCU | ESP8266 | 1 |
| 2 | Computer with Arduino IDE software | IDE 1.8.14 | 1 |
| 3 | Mobile phone with 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.

#include <ESP8266WiFi.h>

const char\* ssid = "xxxxxxxx";

const char\* password = "xxxxxxxxx";

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](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

**Smart Home Android App Development using App Inventor and Arduino.**

**AIM:**

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

**APPPARATUS REQUIRED:**

| **S.No** | **Name of the equipment** | **Specifications** | **Quantity** |
| --- | --- | --- | --- |
| 1 | NodeMCU | ESP8266 | 1 |
| 2 | RGB common cathode LED/Lamp |  | 1 |
| 3 | 5V relay |  | 1 |
| 4 | Jumper wires |  |  |
| 5 | Breadboard |  | 1 |
| 6 | Computer with Arduino IDE software | IDE 1.8.14 | 1 |
| 7 | Mobile phone with wi-fi connectivity |  | 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.

**SCHEMATIC DIAGRAM:**

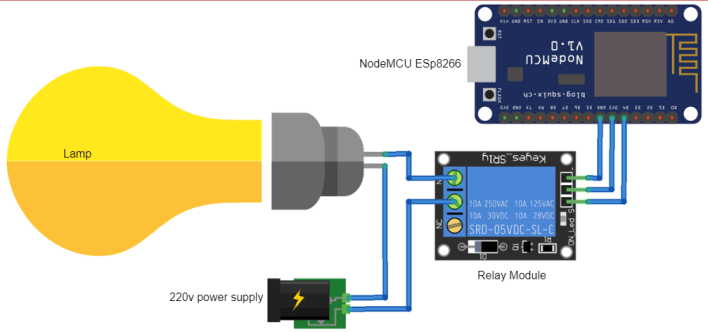


Fig: Wiring of Relay with Node MCU to Control Home Appliance.

| **NodeMCU** | **Relay** |
| --- | --- |
| **Vcc** | **Vcc** |
| **GND** | **GND** |
| **D4** | **Input** |

**Pin connection of relay with NodeMCU ESP8266**

Now we have to make an Android APP using MIT App Inventor for controlling home appliances.

### PROGRAMME:

#include <ESP8266WiFi.h>

const char\* ssid = "Enter Your WiFi Name ";

const char\* password = "Enter Your WiFi Password";

WiFiServer server(80);

void setup() {

Serial.begin(115200); //Default Baud Rate for NodeMCU

delay(10);

pinMode(2, OUTPUT); // Connect Relay to NodeMCU's D4 Pin

digitalWrite(2, 0);

// 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");

// Start the server

server.begin();

Serial.println("Server started");

// Print the IP address

Serial.println(WiFi.localIP());

}

void loop() {

// Check if a client has connected

WiFiClient client = server.available();

if (!client) {

return;

}

// Wait until the client sends some data

Serial.println("new client");

while(!client.available()){

delay(1);

}

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

Serial.println(req);

client.flush();

int val;

if (req.indexOf("/gpio/0") != -1)

val = 0;

else if (req.indexOf("/gpio/1") != -1)

val = 1;

else {

Serial.println("invalid request");

client.stop();

return;

}

// Set GPIO2 according to the request

digitalWrite(2, val);

client.flush();

String s = "HTTP/1.1 200 OK\r\nContent-Type: text/html\r\n\r\n<!DOCTYPE HTML>\r\n<html>\r\nGPIO is now ";

s += (val)?"high":"low";

s += "</html>

**RESULT:**

Hence developed Smart Home Android App Development using App Inventor and Arduino.