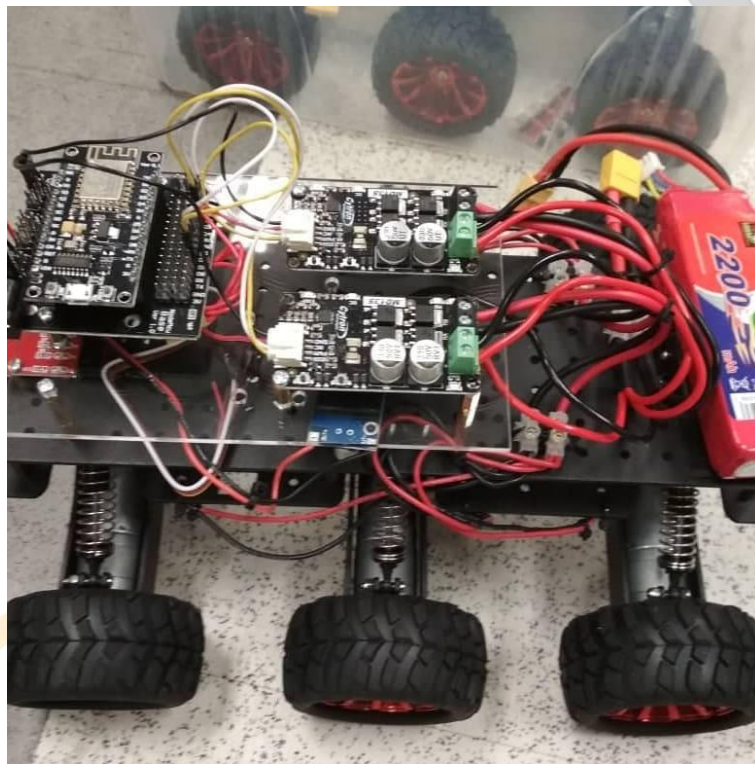


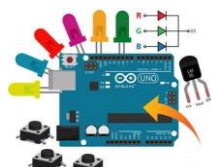
6 Wheel Web based Robot(6WWR)



Oleh NADI ELECZONE SOLUTIONS
AZARUL FAHMIN BIN AB HAMID

azarul@nadiEleczone.com.my

013-4094377



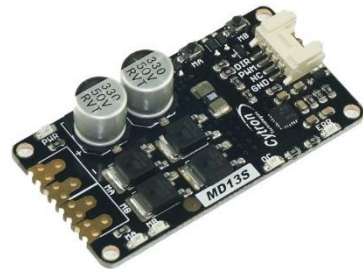
Material List



NodeMCU board



USB cable



DC Motor driver



NodeMCU expansion board



Jumper



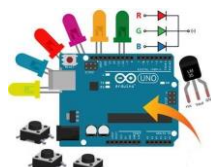
Step down converter module



6WD robot base



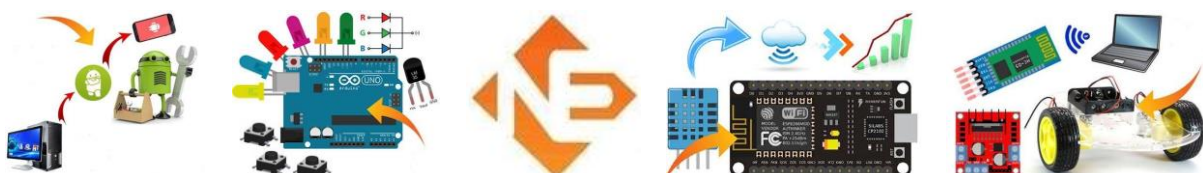
Line following module

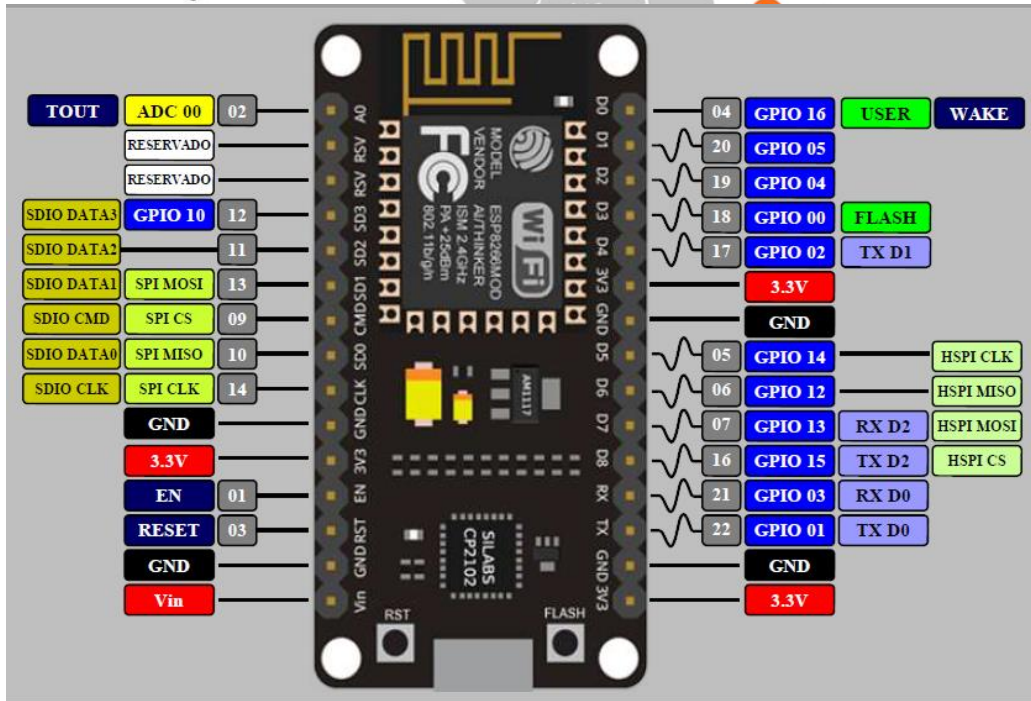


NodeMCU Introduction

NodeMCU is an open source **Internet of Things (IoT)** platform. It includes firmware which runs on the **ESP8266 Wi-Fi SoC** from **Espressif Systems**, and hardware which is based on the **ESP-12** module. The term "**NodeMCU**" by default refers to the firmware rather than the development kits. The firmware uses the **Lua** scripting language. It is based on the **eLua** project, and built on the **Espressif Non-OS SDK** for **ESP8266**. It uses many open source projects, such as **lua-cjson**, and **spiffs**.

As **Arduino.cc** began developing new MCU boards based on non-AVR processors like the **ARM/SAM** MCU and used in the **Arduino Due**, they needed to modify the **Arduino IDE** so that it would be relatively easy to change the **Integrated Development Environment (IDE)** to support alternate tool chains to allow **Arduino C/C++** to be compiled down to these new processors. They did this with the introduction of the **Board Manager** and the **SAM Core**. A "core" is the collection of software components required by the **Board Manager** and the **Arduino IDE** to compile an **Arduino C/C++** source file down to the target MCU's machine language. Some creative **ESP8266** enthusiasts have developed an **Arduino** core for the **ESP8266 WiFi SoC** that is available at the **GitHub ESP8266 Core** webpage. This is what is popularly called the "**ESP8266 Core for the Arduino IDE**" and it has become one of the leading software development platforms for the various **ESP8266** based modules and development boards, including **NodeMCUs**.

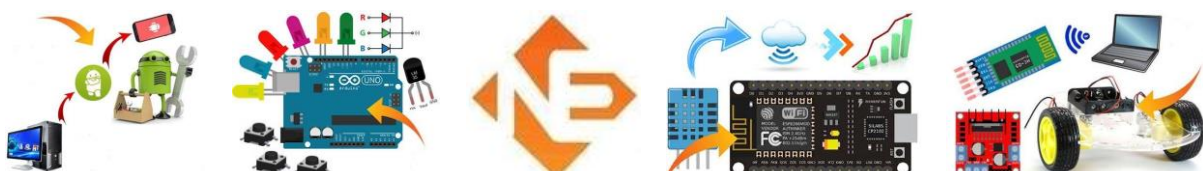




The picture shows the basic parts of the NodeMCU board equipped with the ESP-12 module.

NodeMCU provides access to the **GPIO (General Purpose Input/Output)** and for developing purposes below should be referenced.

IO index	ESP8266 pin	IO index	ESP8266 pin
0 [*]	GPIO16	7	GPIO13
1	GPIO5	8	GPIO15
2	GPIO4	9	GPIO3
3	GPIO0	10	GPIO1
4	GPIO2	11	GPIO9
5	GPIO14	12	GPIO10
6	GPIO12		

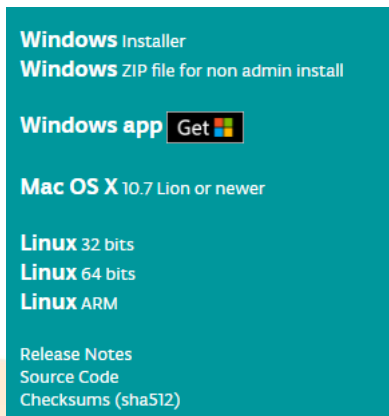


Arduino IDE Software Installation

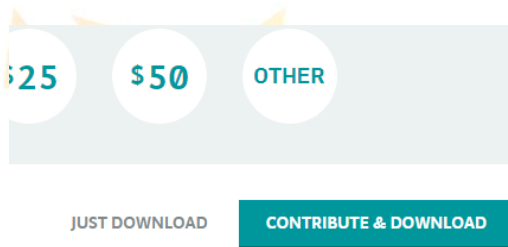


Step 1 : Go to website

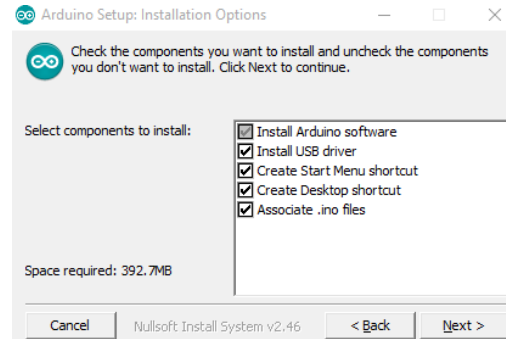
<https://www.arduino.cc/en/Main/Software>



Step 2 : Click *Download*, and choose *Windows Installer* to be download.

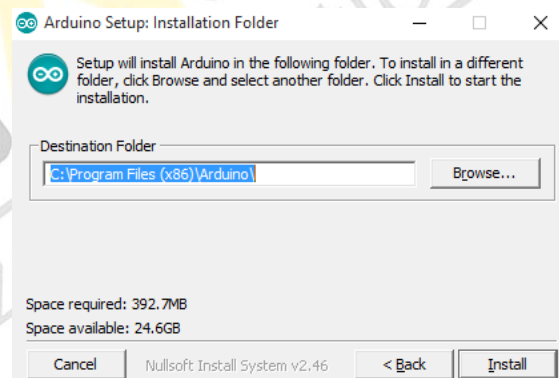


Step 3 : In the Download field, you can select JUST DOWNLOAD or CONTRIBUTE & DOWNLAOD to continue downloading the software.

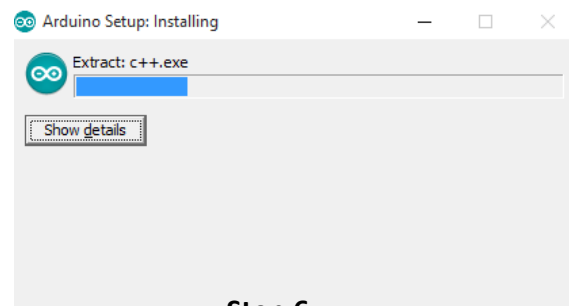


Step 4 :

After download, run the Arduino IDE software installer and click Next.

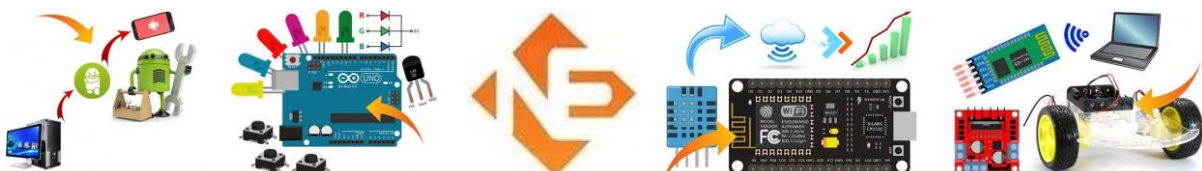


Step 5 : Click Install to start the software installation process.



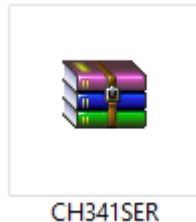
Step 6 :

Wait for the Extract process to finish all files and software installation successfully.

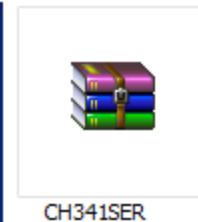




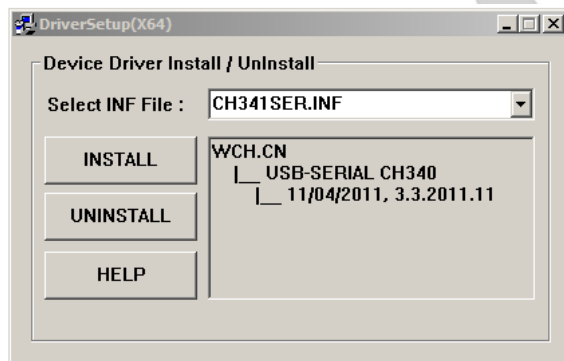
USB converter Driver for NodeMCU



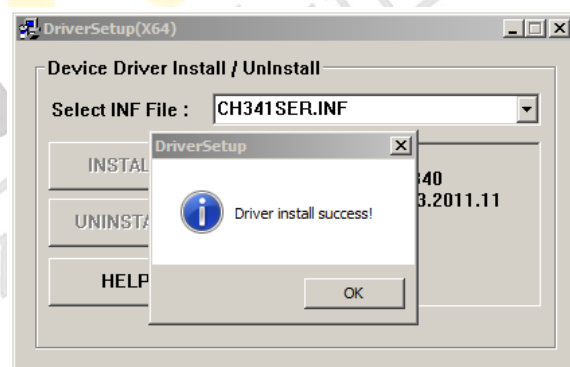
Step 1 : Unzip file CH341SER



Step 2: Double-click on the CH341SER.EXE file to begin installing NodeMCU board driver.

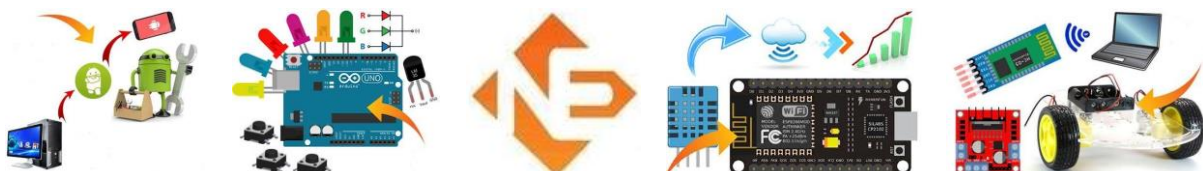


Step 3: Click on the INSTALL button to begin installing Arduino Uno board driver.

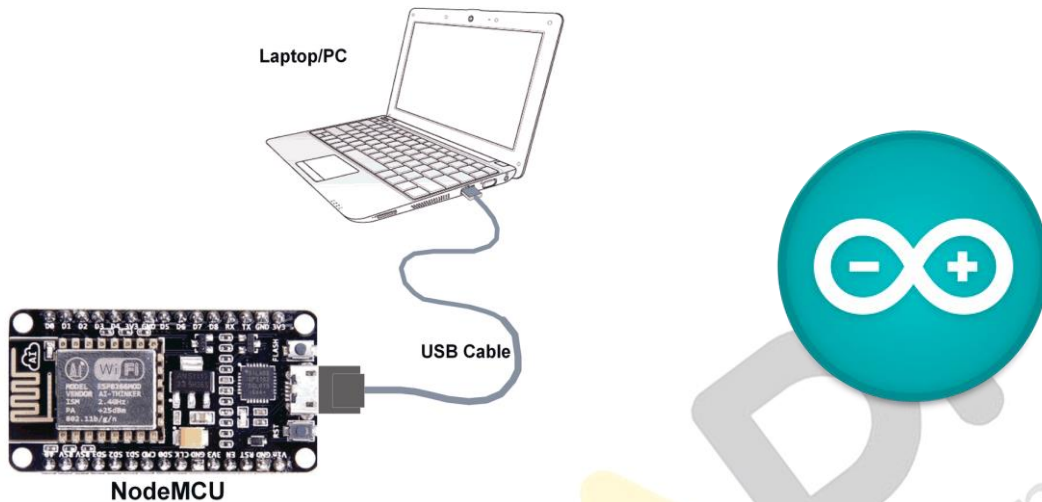


Step 4: Click the OK button after seeing the "Driver install success!" Message. This indicates that the Arduino Uno board driver is successfully installed.

* Refer to Nadi Eleczone Youtube video for CH341 driver installation at
<https://www.youtube.com/user/Cakzone>

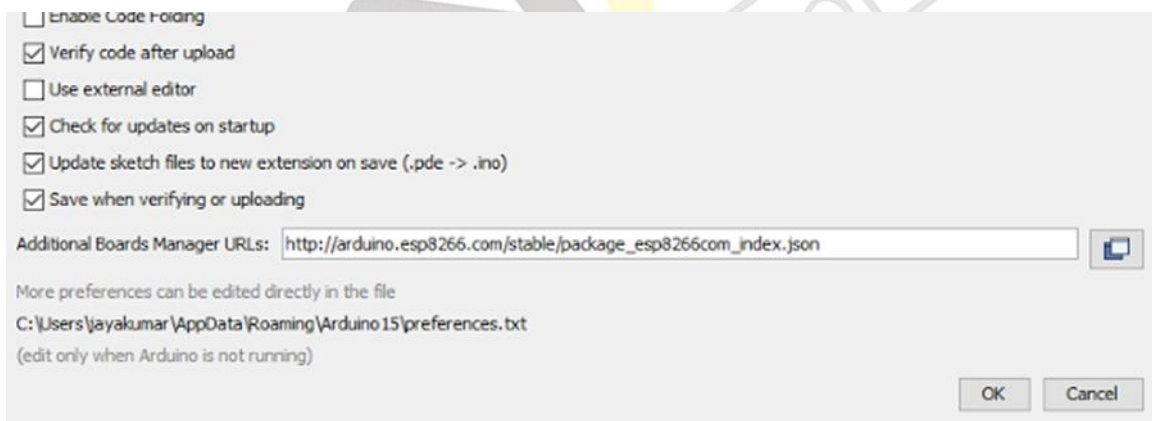


NodeMCU board properties setup

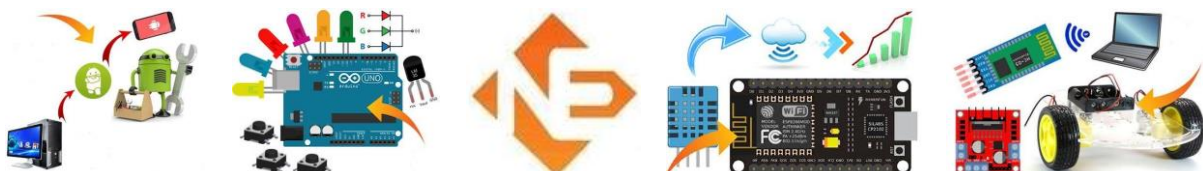


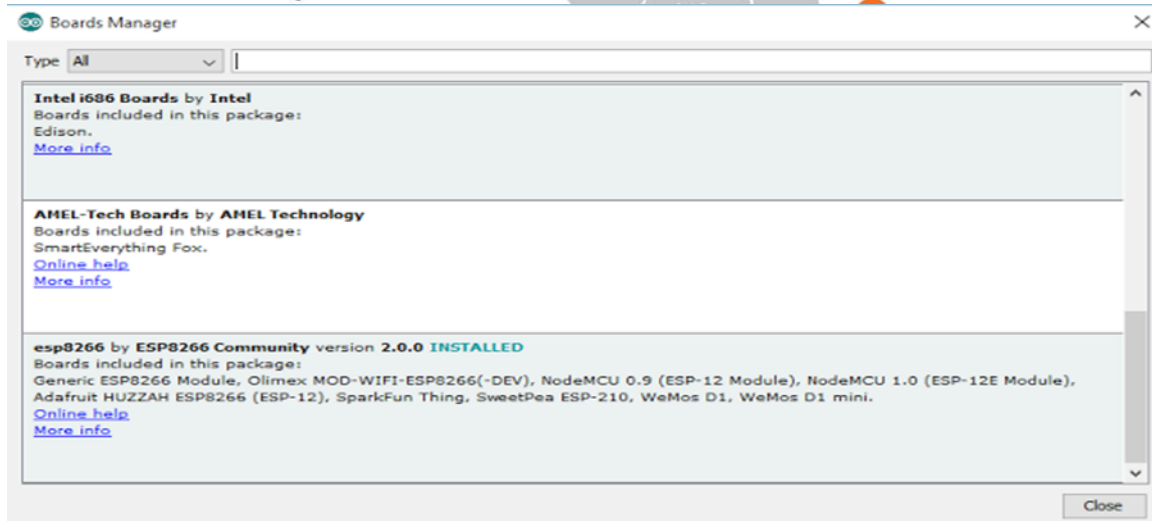
Step 1: Connect the NodeMCU board to the computer.

Step 2: Click on the Arduino icon to open the Arduino IDE software

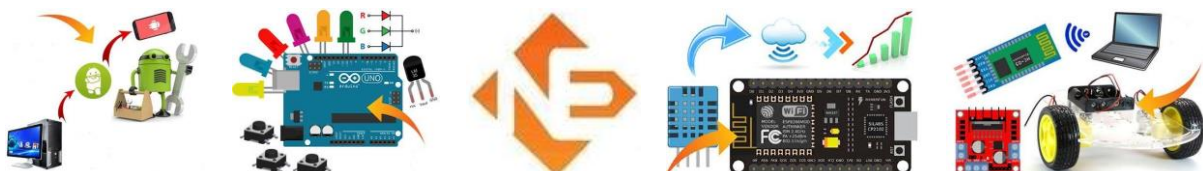


Step 3 : Go to File → Preferences copy this json url and paste http://arduino.esp8266.com/stable/package_esp8266com_index.json into Additional Board Manager URLs: then click OK

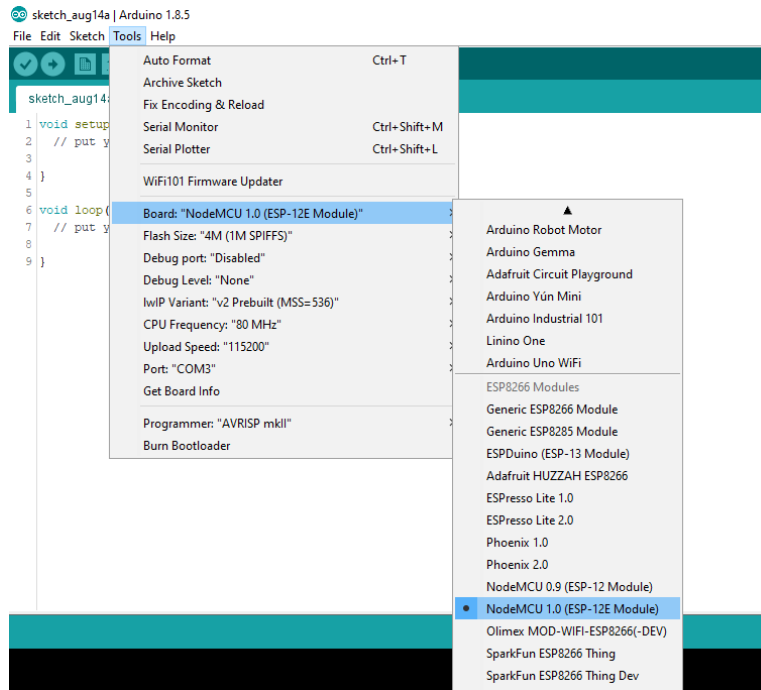




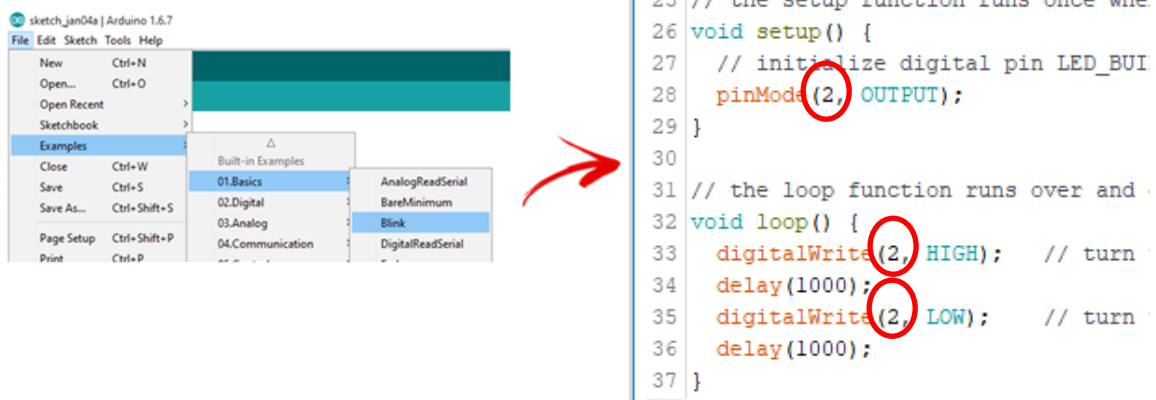
Step 4: Go to Tools→Boards→Board Manager click Board Manager and then wait for a few minute until nodeMcu library link appear then click Install at esp8266 by ESP8266 community



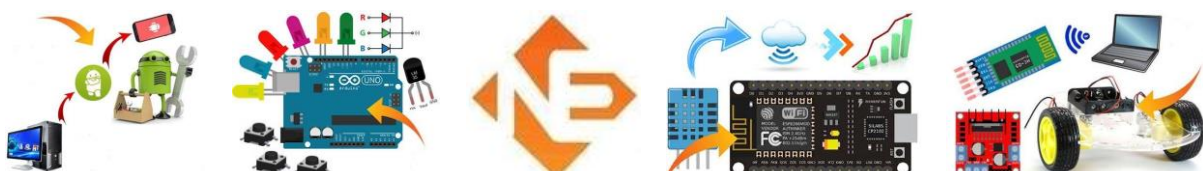
Upload your first sketch

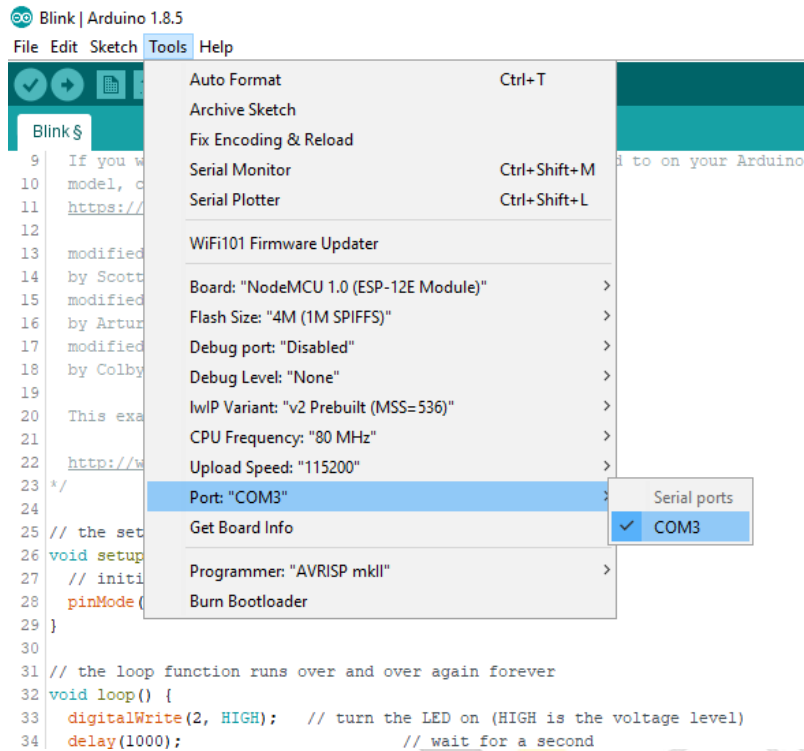


Step 1: Set the NodeMCU board connection with Arduino IDE, click on Tools >> Boards >> and choose NodeMCU 1.0 (ESP-12E Module)

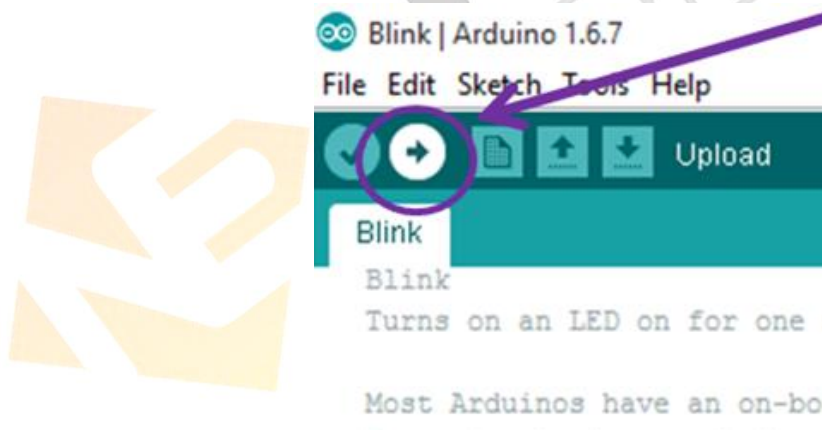


Step 2: Open an existing code example from Arduino IDE software by clicking File >> Examples >> 01 Basics >> Blink. And change the code like the red circle on the image.

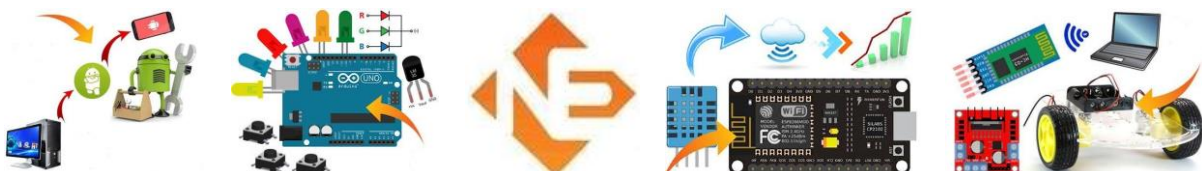




Step 3: Set Com port number NodeMCU board connection with Arduino IDE, click on Tools >> Ports >> and select COMX

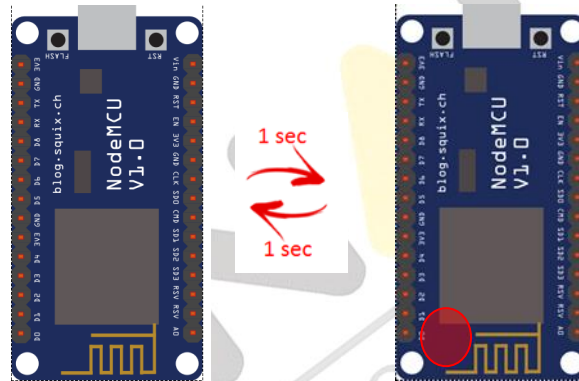


Step 4: Upload the code to the NodeMCU board by clicking on the Upload button.

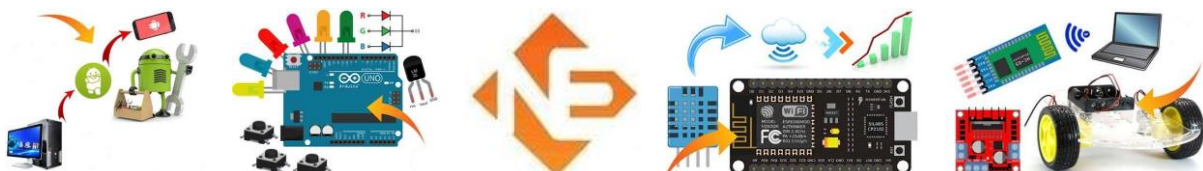




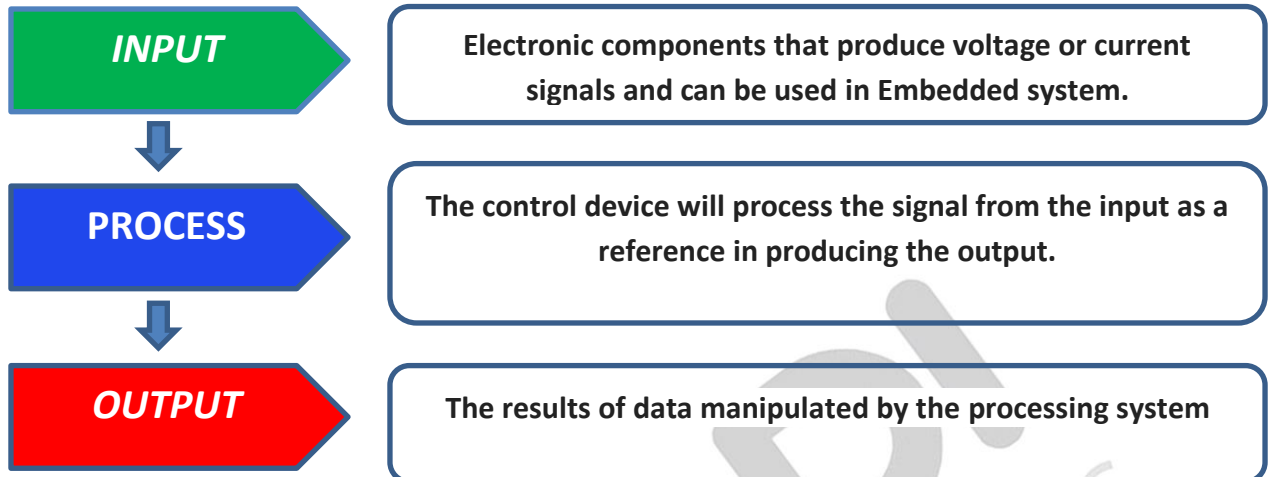
Step 5: Done Uploading message displayed indicating code successfully uploaded to NodeMCU board.



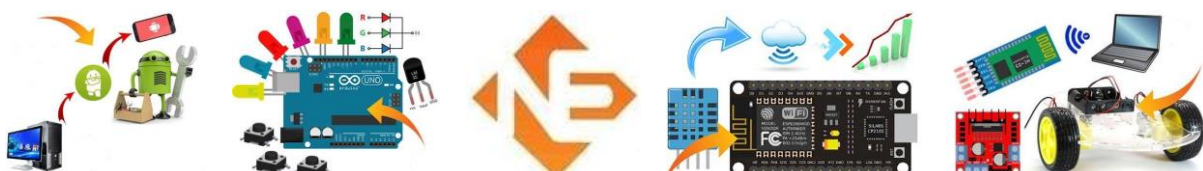
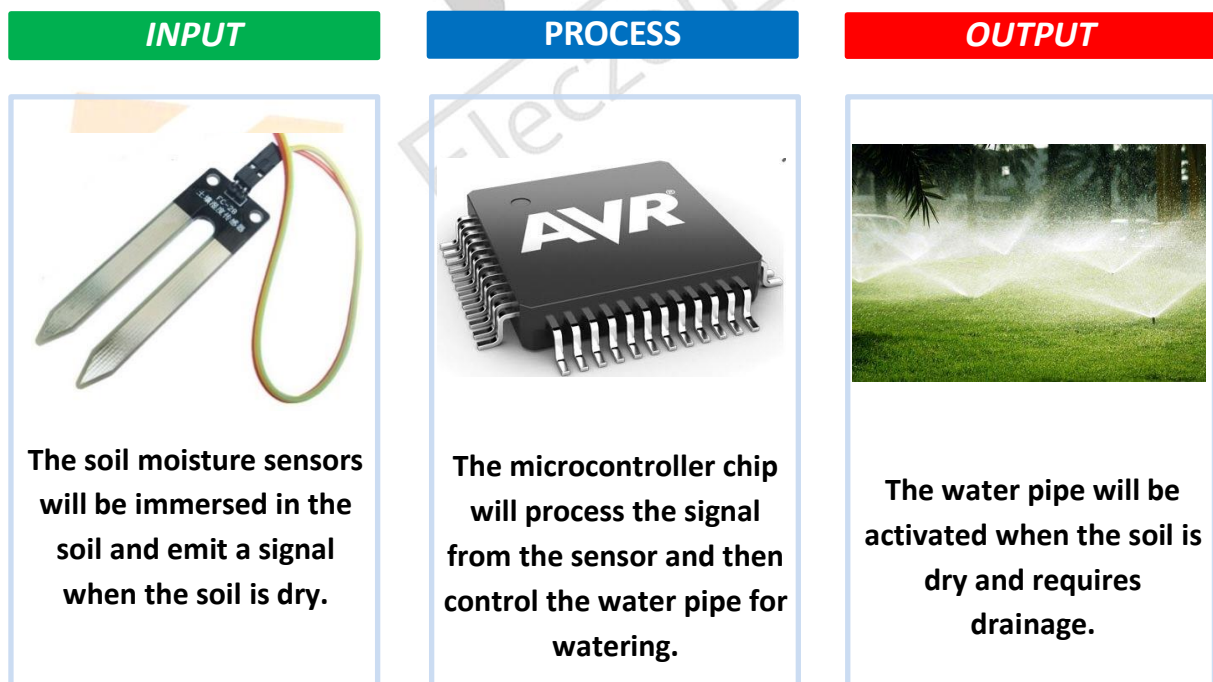
Step 6: Visible lights on the NodeMCU board start blinking with a time interval for 1 second.



Input and Output Concept



Example of system Automatic Groundwater System

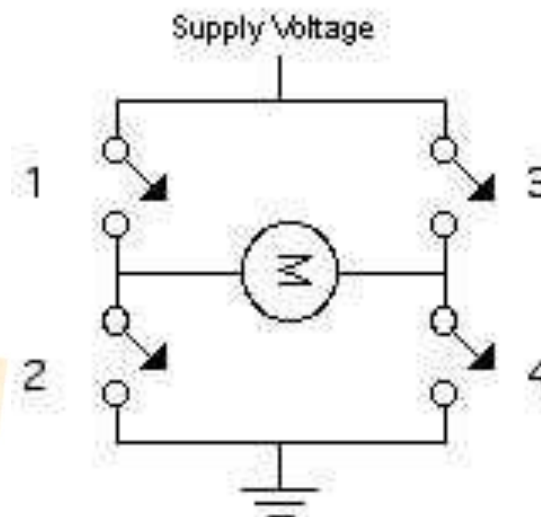


Project 1 – Controlling DC Motor : This project

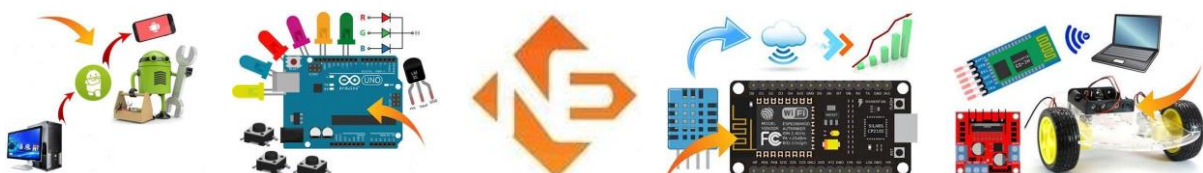
is to identify how to control the DC motor by using the instructions from NodeMCU. To make NodeMCU can control DC motor we need a DC motor driver module first. There are two easily controllable parameters of a DC motor, direction and speed. To control the direction, the polarity of the motor is reversed. To control the speed, the input voltage is varied using pulsewidth modulation.

Direction Control

To control a DC motor from a microcontroller, you use switching arrangement known as an H bridge. It looks like this:

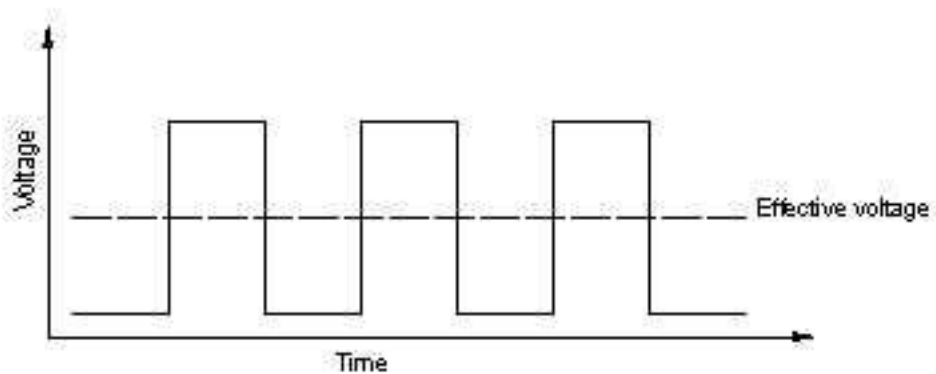


When switches 1 and 4 are closed and 2 and 3 are open, voltage flows from the supply to 1 to the motor to 4 to ground. When 2 and 3 are closed and 1 and 4 are open, polarity is reversed, and voltage flows from the supply to 3 to the motor to 2 to ground.

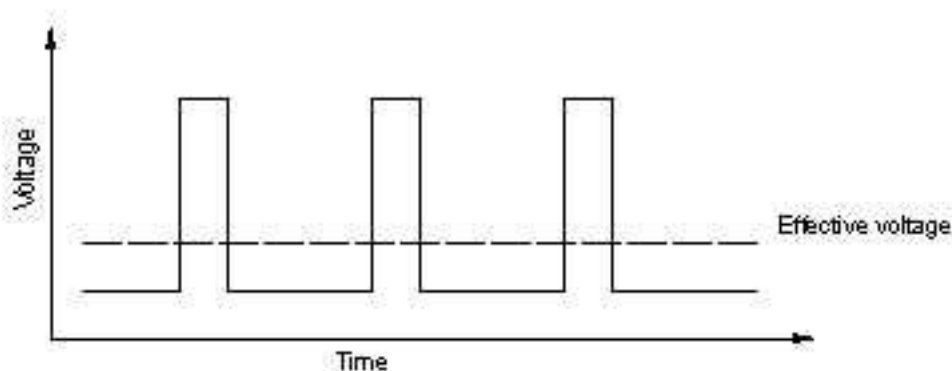


Speed

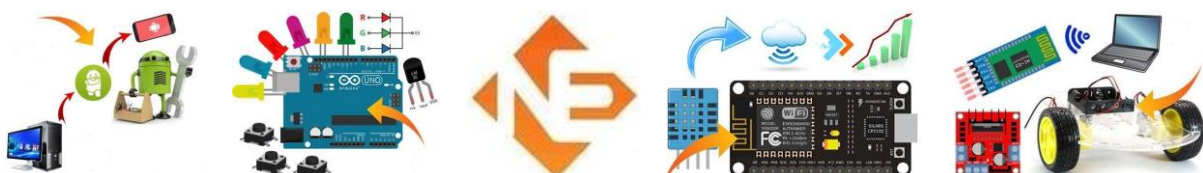
A DC motor's speed is proportional to the supplied voltage. If the voltage drops too far, the motor won't get enough power to turn, but within a certain range, usually 50% of the rated voltage, the motor will run at varying speeds. The most effective way to adjust the speed is by using pulsewidth modulation. This means that you pulse the motor on and off at varying rates, to simulate a voltage. Here are some examples of puleswidths and the voltages they would simulate:



When the time that the voltage is high (the duty cycle) is half the total time in question, the effective voltage is about half the total voltage.



When the duty cycle is reduced to one quarter of the total time, the effective voltage is about one quarter of the total voltage.



List of needed material



NodeMCU
board



NodeMCU
expansion
board



Jumper



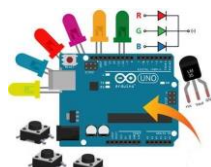
DC motor
Driver board



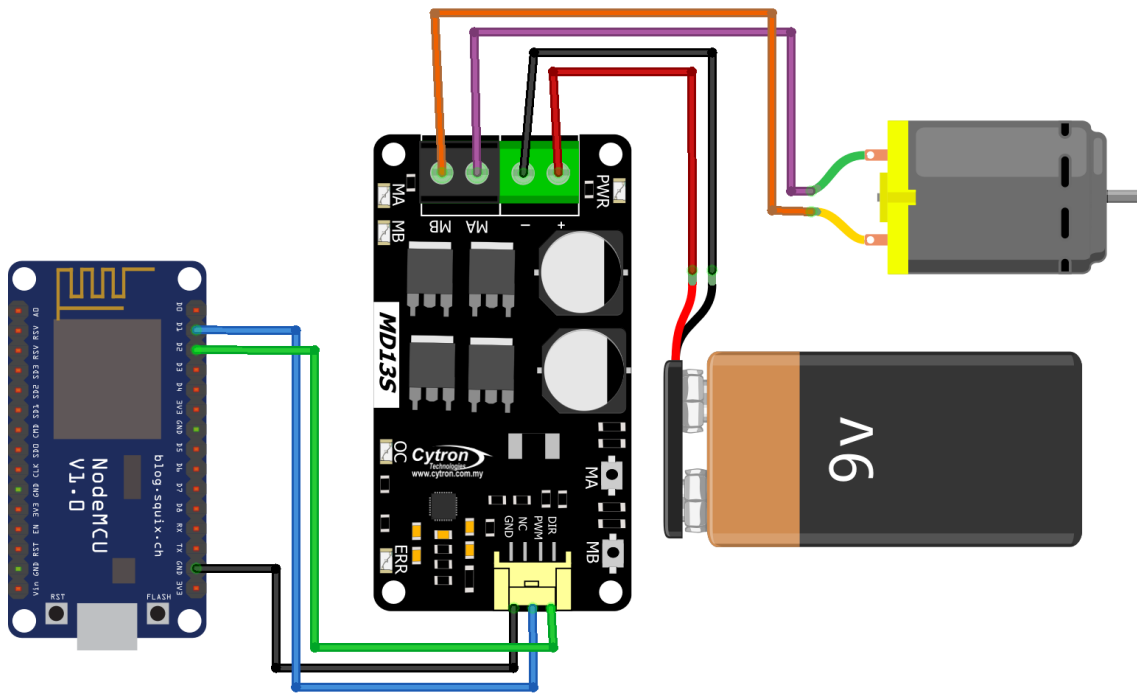
LiPo Battery



DC motor



Build the Circuit



Learn the basic code: The basic code below is the code structure that should always be present in order for a code to be implemented.

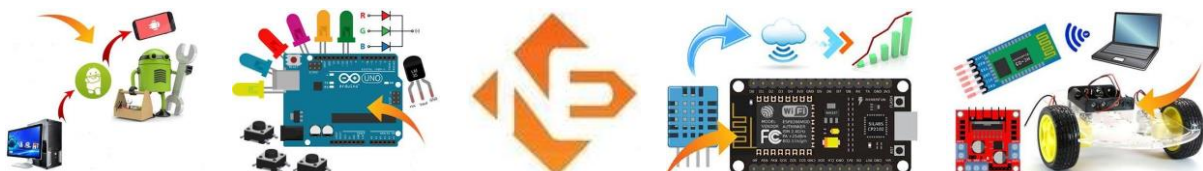
Basic of code

```
void setup()  
{  
  
}
```

In the setup bracket, the code will only run once from top to bottom when the controller is power up.

```
void loop()  
{  
  
}
```

Within the loop bracket, the code will be executed from top to bottom and will be repeated until controller is shut down.



DC motor driver board

Brush motor is the most widely used motor because of the ease of operation. Just supply two terminals of brush motor with DC voltage, it will start rotating! Yet, you need a good motor driver to control the speed and direction from a microcontroller. [MD10C](#) is a great motor driver and now it is already in Revision 3. Anyway, we would like to introduce MD13S where the S stands for SMD (Surface Mount Device)

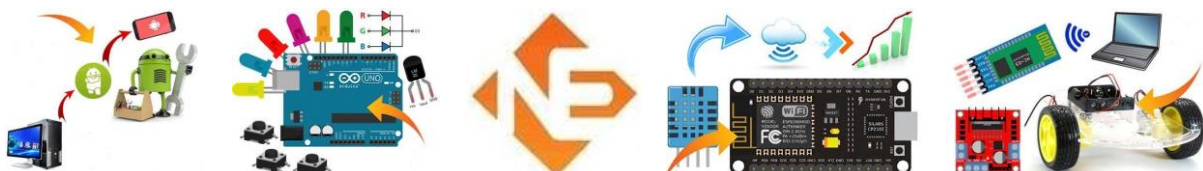
MD13S has great improvement in current protection which MD10C does not have. With new circuit and design, MD13S is able to support current up to 30A max(10 seconds). And the continuous current is at 13A without heatsink (at room temperature of 25°C).

Features:

- Bi-directional control for 1 brushed DC motor.
- Support motor voltage from 6V to 30VDC
- Maximum current up to 13A continuous and 30A peak (10 seconds).
- [GROVE](#) compatible
- 3.3V and 5V logic level input.
- Solid state components provide faster response time and eliminate the wear and tear of mechanical relay.
- Fully NMOS H-Bridge for better efficiency and no heat sink is required.
- Speed control PWM frequency up to 20KHz (Actual output frequency is same as input frequency).
- Support both locked-antiphase and sign-magnitude PWM operation, NOT RC (Radio Control) PWM.
- SMD compatible
- RoHS, FCC and CE compliance
- Dimension: 61mm x 33mm

For other details please go to this link:

<https://docs.google.com/document/d/1icu1GVDxZhUn3ADSUc3JknNcmUMdPcsnJ4MhxOPRo-I/view>

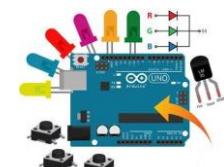
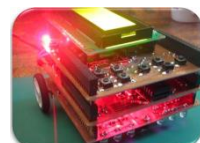
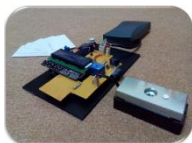


Programming Code

```
const int pwm_right = 5;
const int dir_right = 4;

void setup() {
  Serial.begin(115200);
  delay(10);
  // Initialize the output variables as outputs
  pinMode(pwm_right, OUTPUT);
  pinMode(dir_right, OUTPUT);
  // Set outputs to LOW
  digitalWrite(pwm_right, LOW);
  digitalWrite(dir_right, LOW);
}

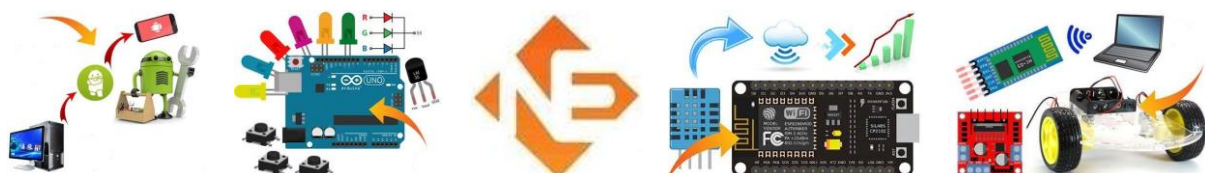
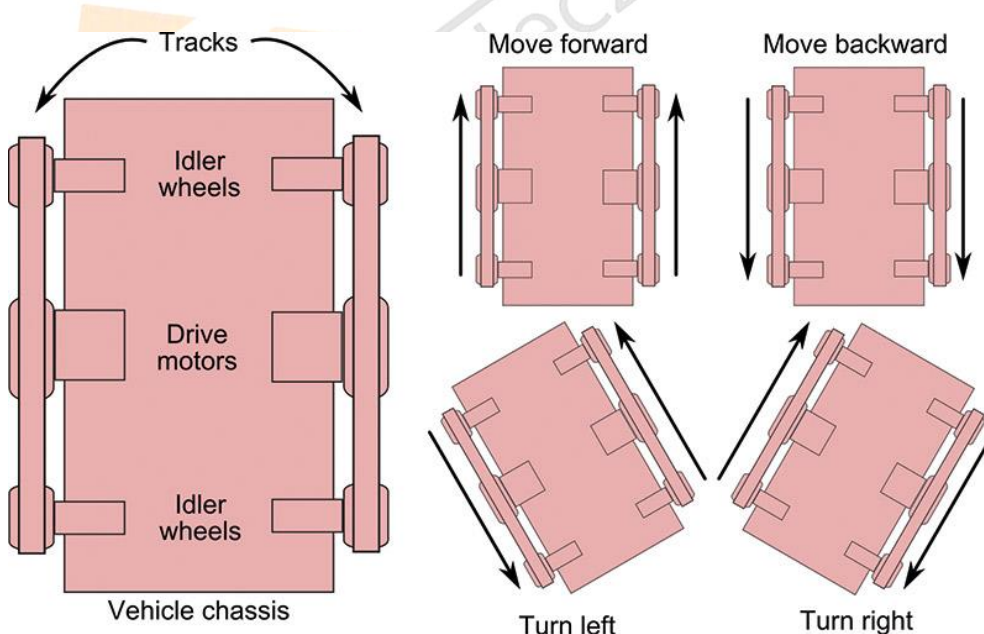
void loop() {
  // put your main code here, to run repeatedly:
  digitalWrite(pwm_right, HIGH);
  digitalWrite(dir_right, LOW);
  delay(2000);
  digitalWrite(pwm_right, HIGH);
  digitalWrite(dir_right, HIGH);
  delay(2000);
  digitalWrite(pwm_right, LOW);
  digitalWrite(dir_right, LOW);
  delay(2000);
}
```



Project 2 – Controlling Robot Direction

: This project is to identify how to control the Robot direction by using the instructions from NodeMCU. Wheeled robots are robots that navigate around the ground using motorized wheels to propel themselves. This design is simpler than using treads or legs and by using wheels they are easier to design, build, and program for movement in flat, not-so-rugged terrain. They are also more well controlled than other types of robots. Disadvantages of wheeled robots are that they cannot navigate well over obstacles, such as rocky terrain, sharp declines, or areas with low friction. Wheeled robots are most popular among the consumer market, their differential steering provides low cost and simplicity. Robots can have any number of wheels, but three wheels are sufficient for static and dynamic balance. Additional wheels can add to balance; however, additional mechanisms will be required to keep all the wheels in the ground, when the terrain is not flat. For this robot direction instruction, we are using Serial Monitor of Arduino IDE as an input. We will send a capital character such as A, B, C and D to make the robot move forward, backward, left, right and stop.

Direction Control



To move forward the left DC motor will rotate counter-clockwise (CCW) and the right motor will rotate clockwise (CW). Then, to move backward the left DC motor will rotate CW and the right motor will rotate CCW. After that, to move left both of DC motor will rotate CW. Lastly, to move left both of DC motor will rotate CCW

List of needed material



NodeMCU
board



NodeMCU
expansion
board



Jumper



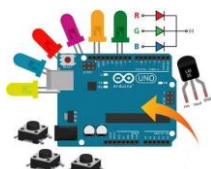
DC motor
Driver board



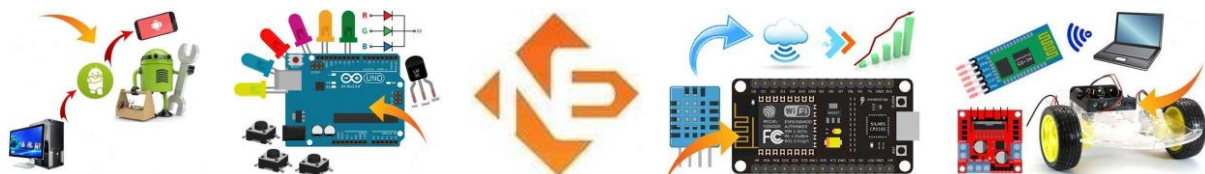
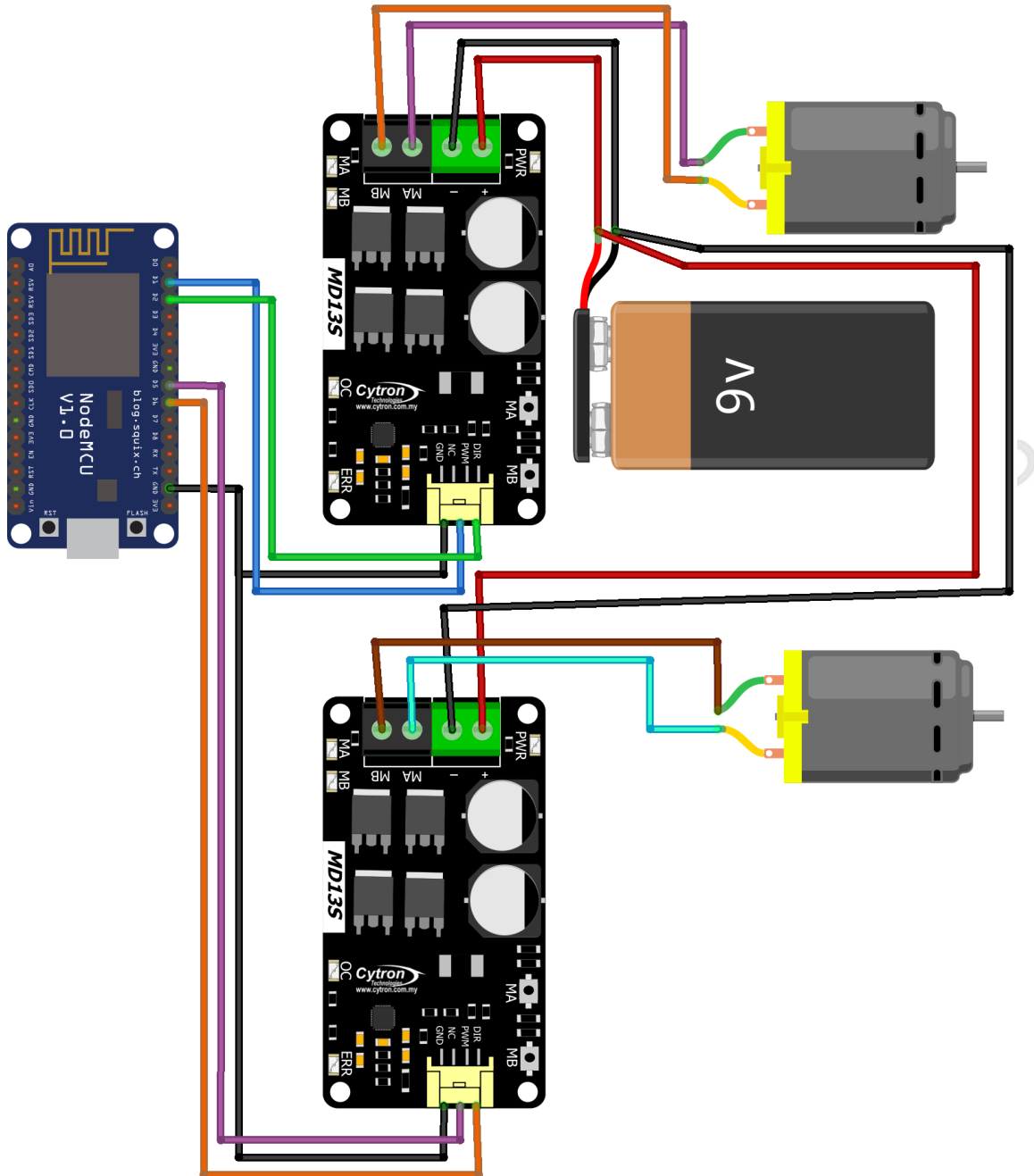
LiPo Battery



DC motor



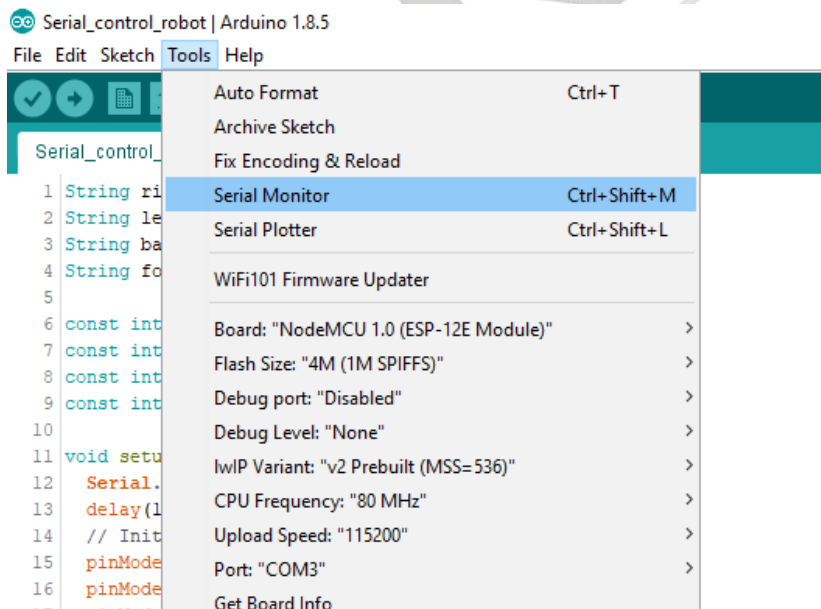
Build the Circuit



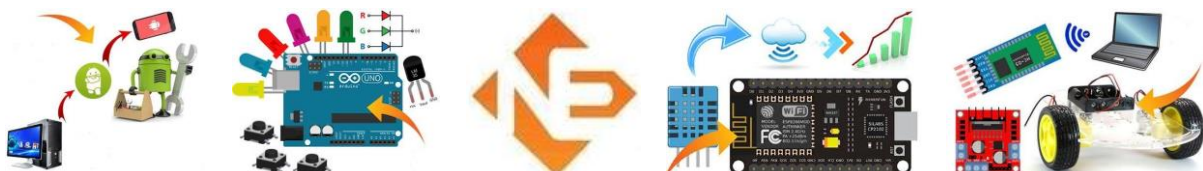
Arduino Serial Monitor

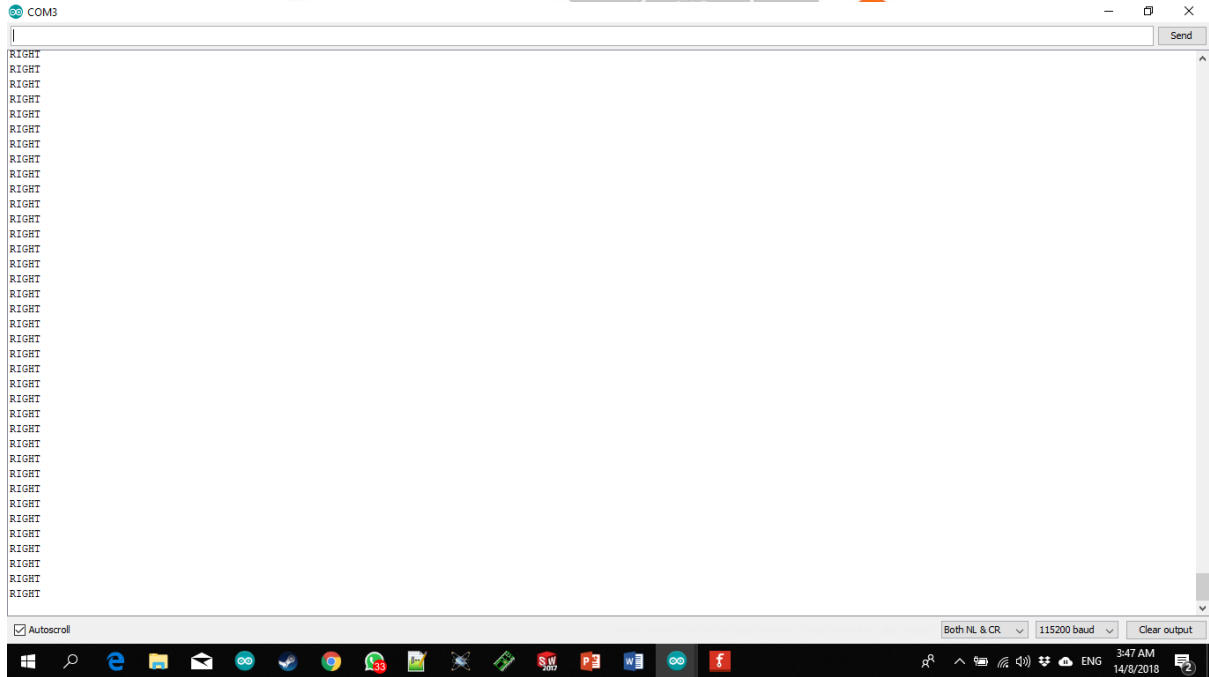
The Arduino IDE has a feature that can be a great help in debugging sketches or controlling Arduino from your computer's keyboard. **The Serial Monitor** is a separate pop-up window that acts as a separate terminal that communicates by receiving and sending Serial Data. Serial Data is sent over a single wire (but usually travels over USB in our case) and consists of a series of 1's and 0's sent over the wire. Data can be sent in both directions.

Recognizing the Arduino IDE serial monitor, this function lets you see the value given by NodeMCU board to the computer via USB medium and send data from computer to NodeMCU board. After clicking on the upload button and receiving the Done Uploading message, you need to click on Tools >> Serial Monitor. Make sure the baudrate on the program and Serial Monitor are the same.



Choosing Serial Monitor and you will get pop up window like the below images.



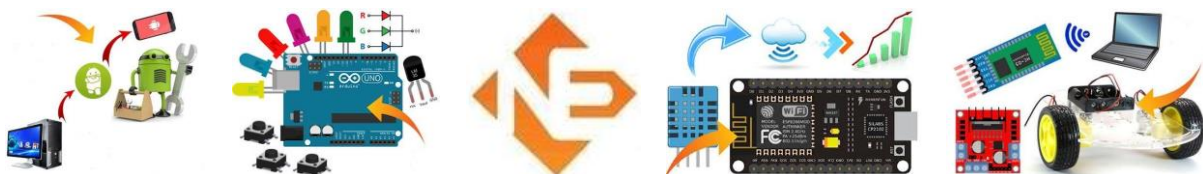


Programming Code

```
String rightState = "off";
String leftState = "off";
String backwardState = "off";
String forwardState = "off";

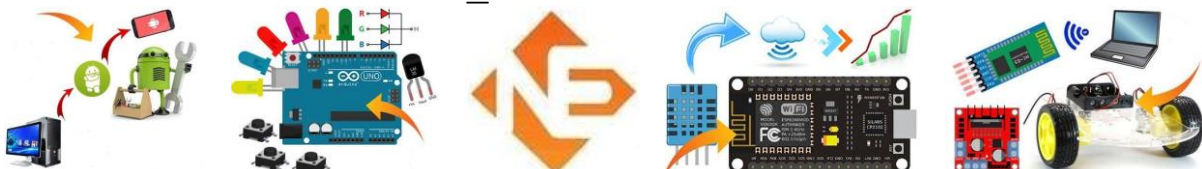
const int pwm_right = 5;
const int dir_right = 4;
const int pwm_left = 14;
const int dir_left = 12;

void setup() {
  Serial.begin(115200);
  delay(10);
  // Initialize the output variables as outputs
  pinMode(pwm_right, OUTPUT);
  pinMode(dir_right, OUTPUT);
  pinMode(pwm_left, OUTPUT);
  pinMode(dir_left, OUTPUT);
}
```

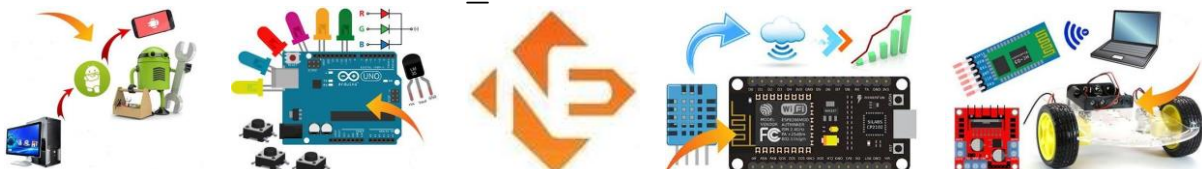


```
// Set outputs to LOW
digitalWrite(pwm_right, LOW);
digitalWrite(dir_right, LOW);
digitalWrite(pwm_left, LOW);
digitalWrite(dir_left, LOW);
}

void loop() {
  char c = Serial.read();
  if (c == 'A') {
    forwardState = "off";
    rightState = "off";
    leftState = "on";
    backwardState = "off";
  } if (c == 'B') {
    forwardState = "off";
    rightState = "off";
    leftState = "off";
    backwardState = "off";
  } if (c == 'C') {
    forwardState = "off";
    rightState = "off";
    leftState = "off";
    backwardState = "on";
  } if (c == 'D') {
    forwardState = "off";
    rightState = "on";
    leftState = "off";
    backwardState = "off";
  } if (c == 'E') {
    forwardState = "on";
    rightState = "off";
    leftState = "off";
    backwardState = "off";
  }
  if (forwardState == "on") {
    Serial.println("FORWARD");
    digitalWrite(pwm_right, HIGH);
    digitalWrite(dir_right, LOW);
    digitalWrite(pwm_left, HIGH);
  }
}
```



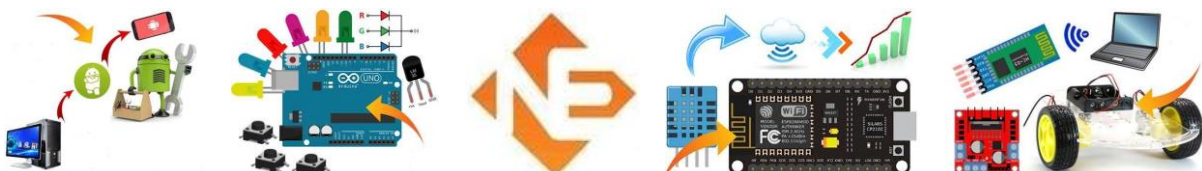
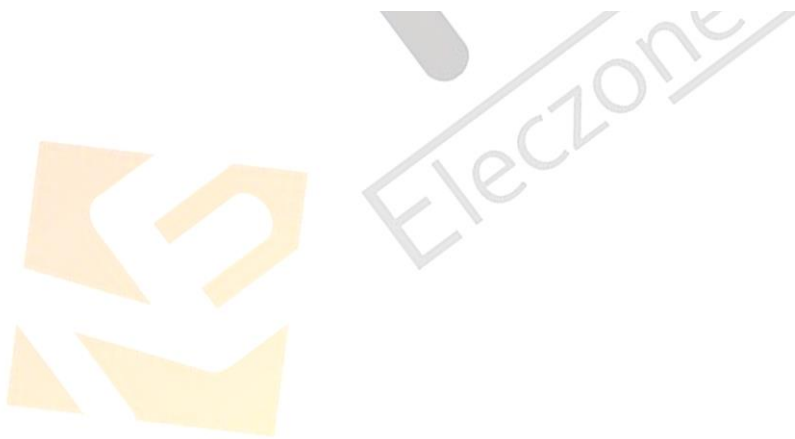

```
digitalWrite(dir_left, HIGH);
delay(2);
digitalWrite(pwm_right, LOW);
digitalWrite(dir_right, LOW);
digitalWrite(pwm_left, LOW);
digitalWrite(dir_left, LOW);
delay(8);
}
if (leftState == "on") {
  Serial.println("LEFT");
  digitalWrite(pwm_right, HIGH);
  digitalWrite(dir_right, LOW);
  digitalWrite(pwm_left, HIGH);
  digitalWrite(dir_left, LOW);
  delay(2);
  digitalWrite(pwm_right, LOW);
  digitalWrite(dir_right, LOW);
  digitalWrite(pwm_left, LOW);
  digitalWrite(dir_left, LOW);
  delay(8);
}
if (rightState == "on") {
  Serial.println("RIGHT");
  digitalWrite(pwm_right, HIGH);
  digitalWrite(dir_right, HIGH);
  digitalWrite(pwm_left, HIGH);
  digitalWrite(dir_left, HIGH);
  delay(2);
  digitalWrite(pwm_right, LOW);
  digitalWrite(dir_right, LOW);
  digitalWrite(pwm_left, LOW);
  digitalWrite(dir_left, LOW);
  delay(8);
}
if (backwardState == "on") {
  Serial.println("BACKWARD");
  digitalWrite(pwm_right, HIGH);
  digitalWrite(dir_right, HIGH);
  digitalWrite(pwm_left, HIGH);
  digitalWrite(dir_left, LOW);
```



```

delay(2);
digitalWrite(pwm_right, LOW);
digitalWrite(dir_right, LOW);
digitalWrite(pwm_left, LOW);
digitalWrite(dir_left, LOW);
delay(8);
}
if (forwardState == "off" && leftState == "off" &&
rightState == "off" && backwardState == "off") {
    Serial.println("STOP");
    digitalWrite(pwm_right, LOW);
    digitalWrite(dir_right, LOW);
    digitalWrite(pwm_left, LOW);
    digitalWrite(dir_left, LOW);
}
}

```

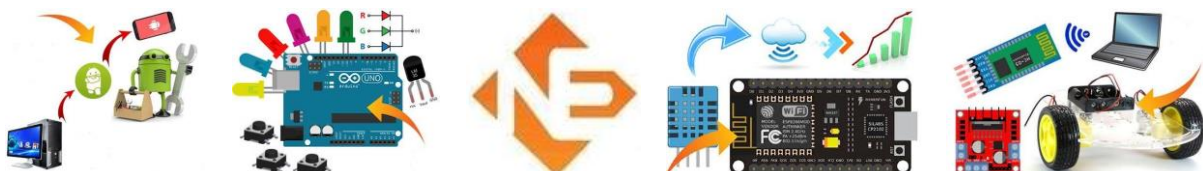


Project 3 – Creating Web Interface : The user

interface (UI), in the industrial design field of human–computer interaction, is the space where interactions between humans and machines occur. The goal of this interaction is to allow effective operation and control of the machine from the human end, whilst the machine simultaneously feeds back information that aids the operators' decision-making process. Examples of this broad concept of user interfaces include the interactive aspects of computer operating systems, hand tools, heavy machinery operator controls, and process controls. The design considerations applicable when creating user interfaces are related to or involve such disciplines as ergonomics and psychology. Generally, the goal of user interface design is to produce a user interface which makes it easy (self-explanatory), efficient, and enjoyable (user-friendly) to operate a machine in the way which produces the desired result. This generally means that the operator needs to provide minimal input to achieve the desired output, and also that the machine minimizes undesired outputs to the human.

HTML

Hypertext Markup Language (HTML) is the standard markup language for creating web pages and web applications. With Cascading Style Sheets (CSS) and JavaScript, it forms a triad of cornerstone technologies for the World Wide Web. Web browsers receive HTML documents from a web server or from local storage and render the documents into multimedia web pages. HTML describes the structure of a web page semantically and originally included cues for the appearance of the document. HTML elements are the building blocks of HTML pages. With HTML constructs, images and other objects such as interactive forms may be embedded into the rendered page. HTML provides a means to create structured documents by denoting structural semantics for text such as headings, paragraphs, lists, links, quotes and other items. HTML elements are delineated by *tags*, written using angle brackets. Tags such

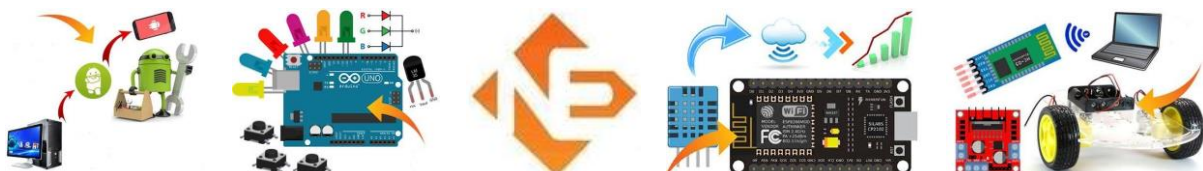


as `` and `<input />` directly introduce content into the page. Other tags such as `<p>` surround and provide information about document text and may include other tags as sub-elements. Browsers do not display the HTML tags, but use them to interpret the content of the page. HTML can embed programs written in a scripting language such as JavaScript, which affects the behavior and content of web pages. Inclusion of CSS defines the look and layout of content.

HTML markup consists of several key components, including those called *tags* (and their *attributes*), character-based *data types*, *character references* and *entity references*. HTML tags most commonly come in pairs like `<h1>` and `</h1>`, although some represent *empty elements* and so are unpaired, for example ``. The first tag in such a pair is the *start tag*, and the second is the *end tag* (they are also called *opening tags* and *closing tags*). Another important component is the HTML *document type declaration*, which triggers standards mode rendering. The following is an example of the classic "Hello, World!" program:

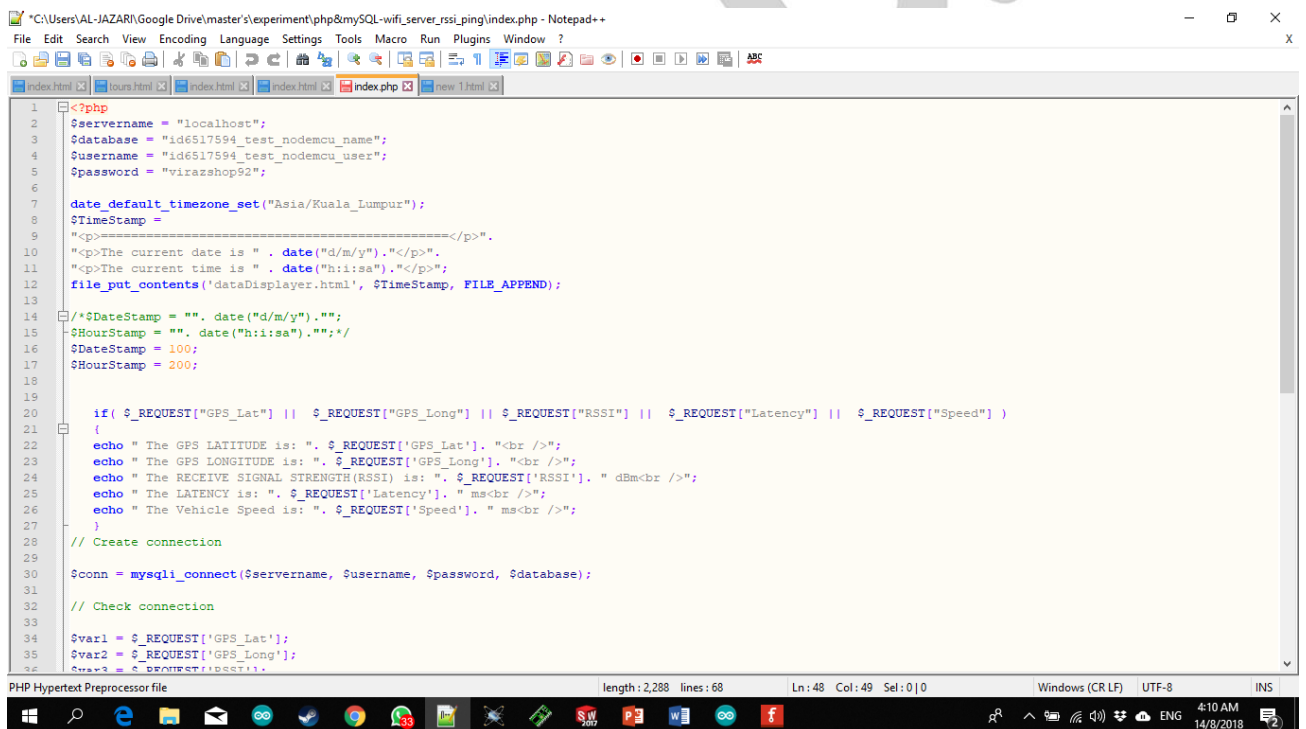
```
<!DOCTYPE html>
<html>
  <head>
    <title>This is a title</title>
  </head>
  <body>
    <p>Hello world!</p>
  </body>
</html>
```

The text between `<html>` and `</html>` describes the web page, and the text between `<body>` and `</body>` is the visible page content. The markup text `<title>This is a title</title>` defines the browser page title.



Notepad++

Notepad++ is a text editor and source code editor for use with Microsoft Windows. It supports tabbed editing, which allows working with multiple open files in a single window. The project's name comes from the C increment operator. Notepad++ is distributed as free software. To download the software, you can go to <https://notepad-plus-plus.org/download/v7.5.8.html>. The interface of Notepad++ is like below image.

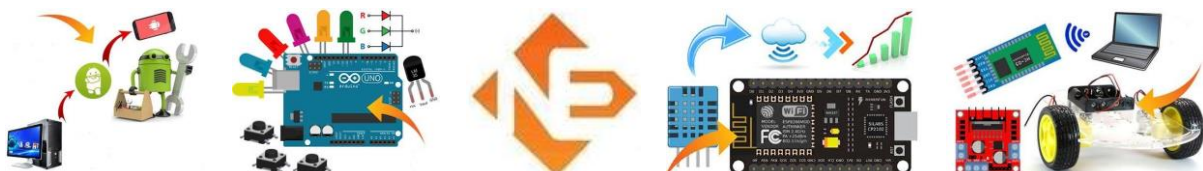


```

1 <?php
2 $servername = "localhost";
3 $database = "id6517594_test_nodemcu_name";
4 $username = "id6517594_test_nodemcu_user";
5 $password = "virazshop92";
6
7 date_default_timezone_set("Asia/Kuala_Lumpur");
8 $TimeStamp =
9
10 "cp>The current date is ". date("d/m/y"). "</p>".
11 "cp>The current time is ". date("h:i:sa"). "</p>";
12 file_put_contents('dataDisplay.html', $TimeStamp, FILE_APPEND);
13
14 /*$DateStamp = "". date("d/m/y")."";
15 $HourStamp = "". date("h:i:sa")."";*/
16 $DateStamp = 100;
17 $HourStamp = 200;
18
19
20 if( $_REQUEST["GPS_Lat"] || $_REQUEST["GPS_Long"] || $_REQUEST["RSSI"] || $_REQUEST["Latency"] || $_REQUEST["Speed"] )
21 {
22 echo " The GPS LATITUDE is: ". $_REQUEST['GPS_Lat']. "<br />";
23 echo " The GPS LONGITUDE is: ". $_REQUEST['GPS_Long']. "<br />";
24 echo " The RECEIVE SIGNAL STRENGTH(RSSI) is: ". $_REQUEST['RSSI']. " dBm<br />";
25 echo " The LATENCY is: ". $_REQUEST['Latency']. " ms<br />";
26 echo " The Vehicle Speed is: ". $_REQUEST['Speed']. " ms<br />";
27 }
28 // Create connection
29
30 $conn = mysqli_connect($servername, $username, $password, $database);
31
32 // Check connection
33
34 $var1 = $_REQUEST['GPS_Lat'];
35 $var2 = $_REQUEST['GPS_Long'];
36 $var3 = $_REQUEST['RSSI'];

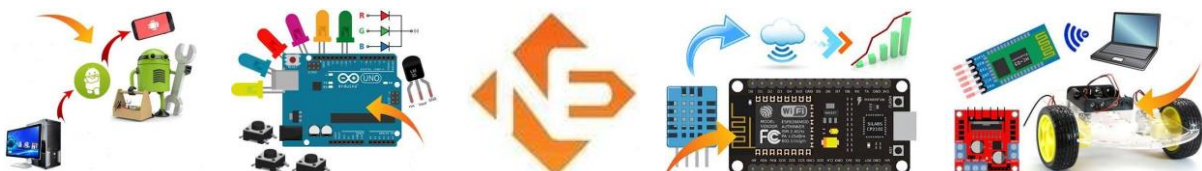
```

We will write the HTML code in this software to design the web interface for our robot.



Programming Code

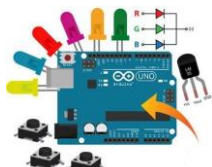
```
<!DOCTYPE HTML>
<html>
  <head>
    <meta name='apple-mobile-web-app-capable'
content='yes' />
    <meta name='apple-mobile-web-app-status-
bar-style' content='black-translucent' />
  </head>
  <body bgcolor = \"#f7e6ec\">
    <hr/><hr>
    <h4><center> Esp8266 Robot Control 6WWR
</center></h4>
    <hr/><hr>
    <br><br>
    <br><br>
    <center>
      FORWARD
      <a
href=\"/forwardon\"/><button>Turn On </button></a>
      <a
href=\"/forwardoff\"/><button>Turn Off
</button></a><br />
    </center>
    <br><br>
    <center>
      LEFT
      <a href=\"/lefton\"/><button>Turn
On </button></a>
      <a href=\"/leftoff\"/><button>Turn
Off </button></a><br />
    </center>
    <br><br>
    <center>
      RIGHT
      <a href=\"/righton\"/><button>Turn
On </button></a>
```



```

<a
href=\"/rightoff\"><button>Turn Off
</button></a><br />
</center>
<br><br>
<center>
BACKWARD
<a
href=\"/backwardon\"><button>Turn On </button></a>
<a
href=\"/backwardoff\"><button>Turn Off
</button></a><br />
</center>
<br><br>
<center>
<table border=\"5\">
<tr>
<td>Light 1 is ON</td>
<br/>
<td>Light 2 is on</td>
</tr>
<tr>
<td>Light 3 is on</td>
<td>Light 4 is ON</td>
</tr>
</table>
</center>
</html>

```



Project 4 – Webserver Robot Control :

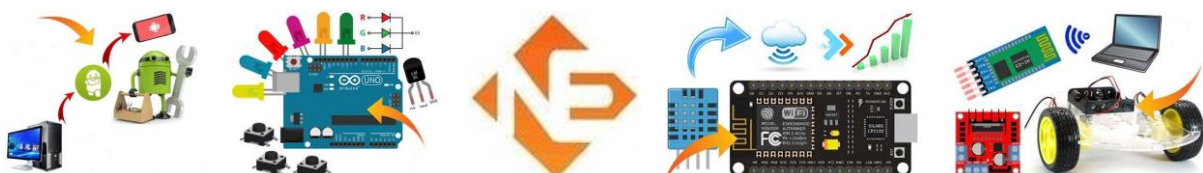
Web server refers to [server software](#), or hardware dedicated to running said software, that can serve contents to the [World Wide Web](#). A web server processes incoming network requests over the [HTTP](#) protocol (and several other related protocols).

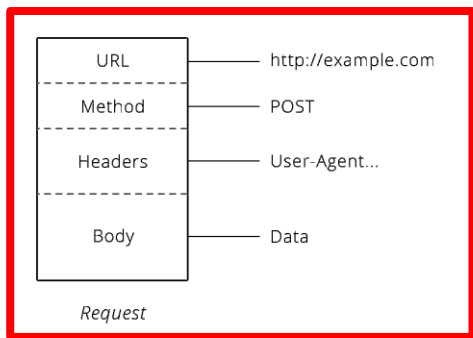
HTTP

The Hypertext Transfer Protocol (HTTP) is designed to enable communications between clients and servers. HTTP works as a request-response protocol between a client and server. A web browser may be the client, and an application on a computer that hosts a web site may be the server.

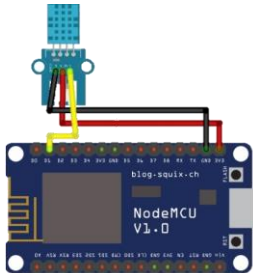
Example: A client (browser) submits an HTTP request to the server; then the server returns a response to the client. The response contains status information about the request and may also contain the requested content.

As per below image, this is example how are our robot getting instruction from pc or smartphone.

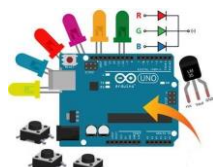
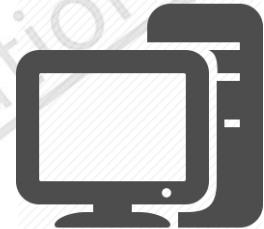
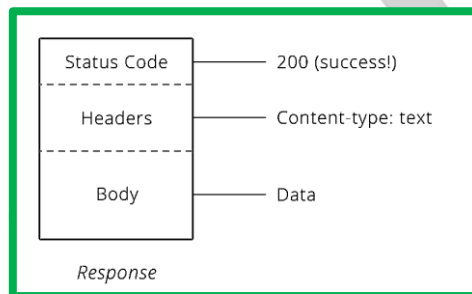




HTTP Request



HTTP Response



List of needed material



NodeMCU
board



NodeMCU
expansion
board



Jumper



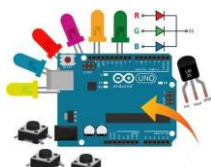
DC motor
Driver board



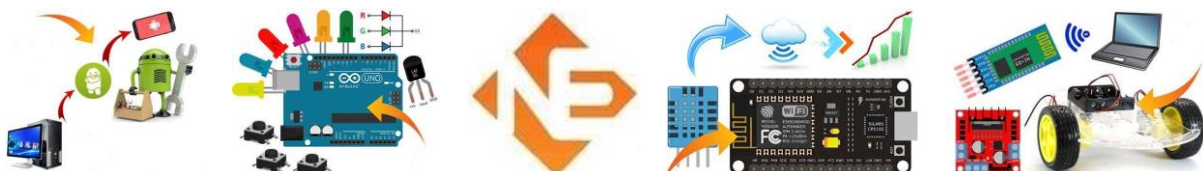
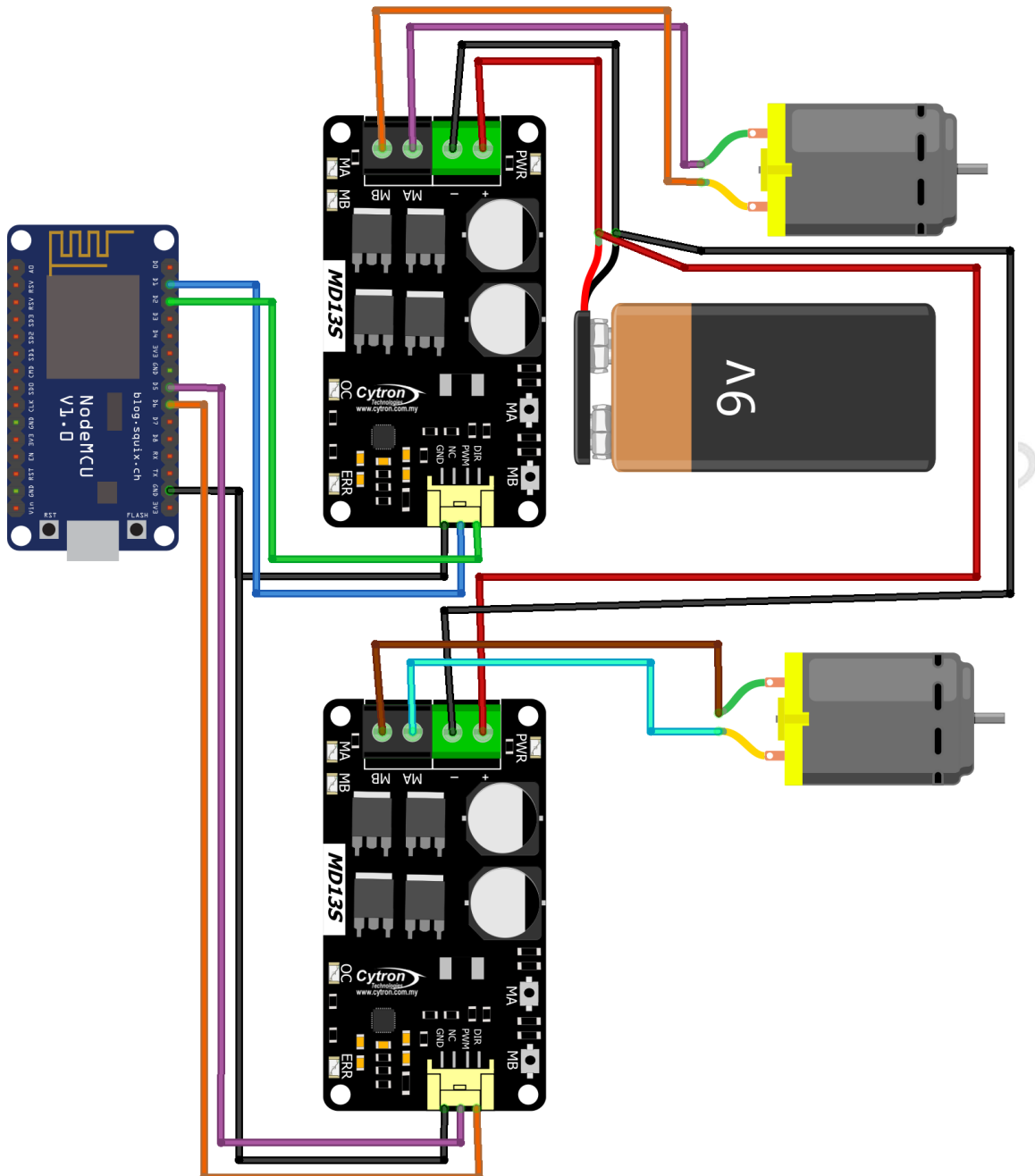
LiPo Battery



DC motor



Build the Circuit



Programming Code

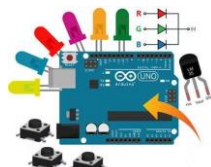
```
#include <ESP8266WiFi.h>

const char* ssid = "gulai itik";
const char* password = "itiksedap";
long counter;
String rightState = "off";
String leftState = "off";
String backwardState = "off";
String forwardState = "off";

const int pwm_right = 5;
const int dir_right = 4;
const int pwm_left = 14;
const int dir_left = 12;
WiFiServer server(80);

void setup() {
  Serial.begin(115200);
  delay(10);
  // Initialize the output variables as outputs
  pinMode(pwm_right, OUTPUT);
  pinMode(dir_right, OUTPUT);
  pinMode(pwm_left, OUTPUT);
  pinMode(dir_left, OUTPUT);
  // Set outputs to LOW
  digitalWrite(pwm_right, LOW);
  digitalWrite(dir_right, LOW);
  digitalWrite(pwm_left, LOW);
  digitalWrite(dir_left, LOW);

  // Connect to WiFi network
  Serial.println();
  Serial.println();
  Serial.print("Connecting to ");
  Serial.println(ssid);
```




```
WiFi.begin(ssid, password);

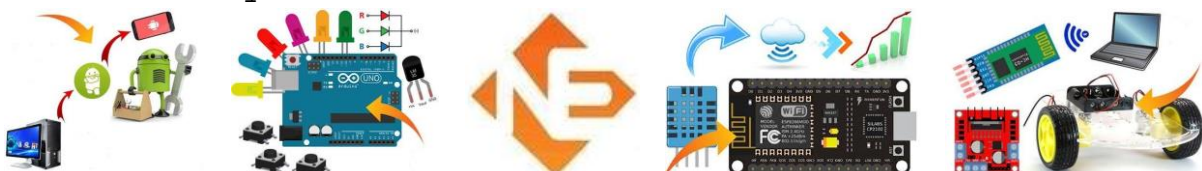
while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
}
Serial.println("");
Serial.println("WiFi connected");

// Start the server
server.begin();
Serial.println("Server started");

// Print the IP address
Serial.print("Use this URL to connect: ");
Serial.print("http://");
Serial.print(WiFi.localIP());
Serial.println("/");
}

void loop() {
    // Check if a client has connected
    // unsigned long interrupt_time = millis();
    counter++;
    Serial.print("COUNTER");
    Serial.println(counter);
    if (counter > 5){
        Serial.println("LOL");
        counter = 0;
    }
    WiFiClient client = server.available();
    if (!client) {
        return;
    }

    // Wait until the client sends some data
    Serial.println("new client");
    if (!client.available()) {
        delay(1);
    }
}
```



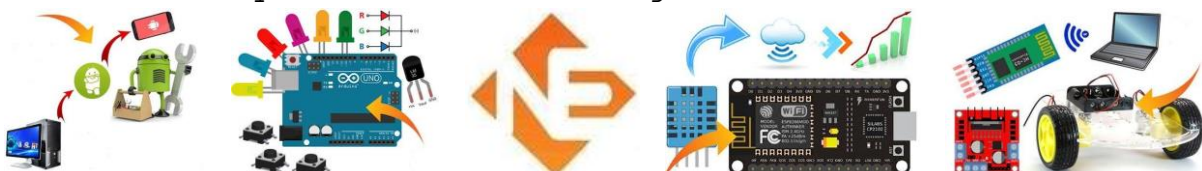
```

    }
    // Read the first line of the request
    String request = client.readStringUntil('\r');
    Serial.println(request);
    client.flush();

    // Match the request

    if (request.indexOf("/lefton") >= 0) {
        Serial.println("GPIO 5 on");
        forwardState = "off";
        rightState = "off";
        leftState = "on";
        backwardState = "off";
    } if (request.indexOf("/leftoff") >= 0) {
        Serial.println("GPIO 5 off");
        forwardState = "off";
        rightState = "off";
        leftState = "off";
        backwardState = "off";
    } if (request.indexOf("/backwardon") >= 0) {
        Serial.println("GPIO 4 on");
        forwardState = "off";
        rightState = "off";
        leftState = "off";
        backwardState = "on";
    } if (request.indexOf("/backwardoff") >= 0) {
        Serial.println("GPIO 4 off");
        forwardState = "off";
        rightState = "off";
        leftState = "off";
        backwardState = "off";
    } if (request.indexOf("/righton") >= 0) {
        Serial.println("GPIO 16 on");
        forwardState = "off";
        rightState = "on";
        leftState = "off";
        backwardState = "off";
    } if (request.indexOf("/rightoff") >= 0) {

```



 Website : <http://nadiieleczonesolutions.blogspot.com/>
 Facebook Page : <https://www.facebook.com/eleczone>
 Instagram : <http://instagram.com/eleczone>

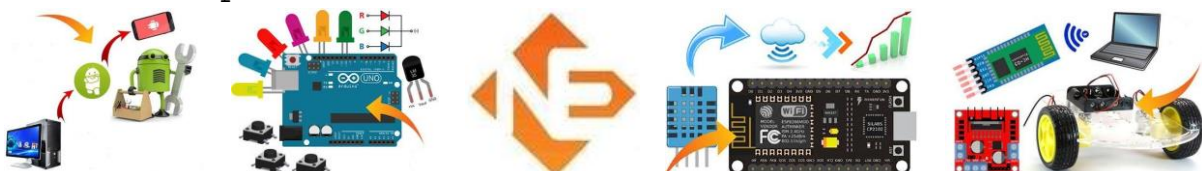
```

Serial.println("GPIO 16 off");
forwardState = "off";
rightState = "off";
leftState = "off";
backwardState = "off";
} if (request.indexOf("/forwardon") >= 0) {
Serial.println("GPIO 14 on");
forwardState = "on";
rightState = "off";
leftState = "off";
backwardState = "off";

} if (request.indexOf("/forwardoff") >= 0) {
Serial.println("GPIO 14 off");
forwardState = "off";
rightState = "off";
leftState = "off";
backwardState = "off";
}
// Set ledPin according to the request
//digitalWrite(ledPin, value);

// Return the response
client.println("HTTP/1.1 200 OK");
client.println("Content-Type: text/html");
client.println(""); // do not forget this one
client.println("<!DOCTYPE HTML>");
client.println("<html>");
client.println("<head>");
client.println("<meta name='apple-mobile-web-app-
capable' content='yes' />");
client.println("<meta name='apple-mobile-web-app-
status-bar-style' content='black-translucent' />");
client.println("</head>");
client.println("<body bgcolor = \"#f7e6ec\">");
client.println("<hr/><hr>");
client.println("<h4><center> Esp8266 Electrical
Device Control </center></h4>");
client.println("<hr/><hr>");
client.println("<br><br>");

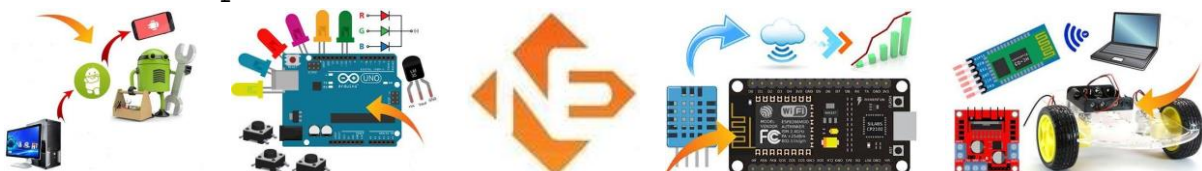
```



```

client.println("<br><br>");
client.println("<center>");
client.println("FORWARD");
client.println("<a
href=\""/forwardon\"\"><button>Turn On
</button></a>");
client.println("<a
href=\""/forwardoff\"\"><button>Turn Off
</button></a><br />");
client.println("</center>");
client.println("<br><br>");
client.println("<center>");
client.println("LEFT");
client.println("<a href=\""/lefton\"\"><button>Turn
On </button></a>");
client.println("<a
href=\""/leftoff\"\"><button>Turn Off
</button></a><br />");
client.println("</center>");
client.println("<br><br>");
client.println("<center>");
client.println("RIGHT");
client.println("<a
href=\""/righton\"\"><button>Turn On </button></a>");
client.println("<a
href=\""/rightoff\"\"><button>Turn Off
</button></a><br />");
client.println("</center>");
client.println("<br><br>");
client.println("<center>");
client.println("BACKWARD");
client.println("<a
href=\""/backwardon\"\"><button>Turn On
</button></a>");
client.println("<a
href=\""/backwardoff\"\"><button>Turn Off
</button></a><br />");
client.println("</center>");
client.println("<br><br>");
client.println("<center>");

```

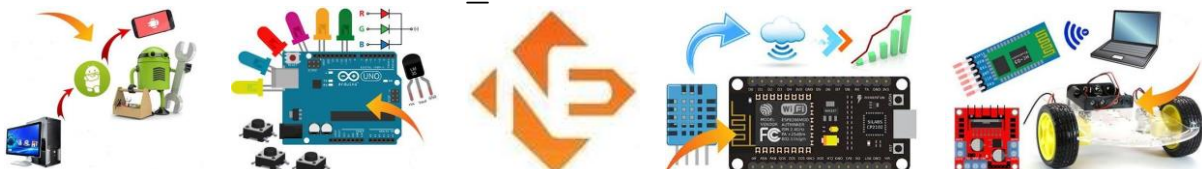



```

client.println("<table border=\"5\">");
client.println("<tr>");
if (forwardState == "off") {
    client.print("<td>Light 1 is OFF</td>");
}
if (forwardState == "on") {
    client.print("<td>Light 1 is ON</td>");
}
client.println("<br />");
if (leftState == "off") {
    client.print("<td>Light 2 is off</td>");
}
if (leftState == "on") {
    client.print("<td>Light 2 is on</td>");
}
client.println("</tr>");
client.println("<tr>");
if (rightState == "off") {
    client.print("<td>Light 3 is Off</td>");
}
if (rightState == "on") {
    client.print("<td>Light 3 is on</td>");
}
if (backwardState == "off") {
    client.print("<td>Light 4 is Off</td>");
}
if (backwardState == "on") {
    client.print("<td>Light 4 is ON</td>");
}
client.println("</tr>");
client.println("</table>");
client.println("</center>");
client.println("</html>");
Serial.println("Client disonnected");
Serial.println("");
}

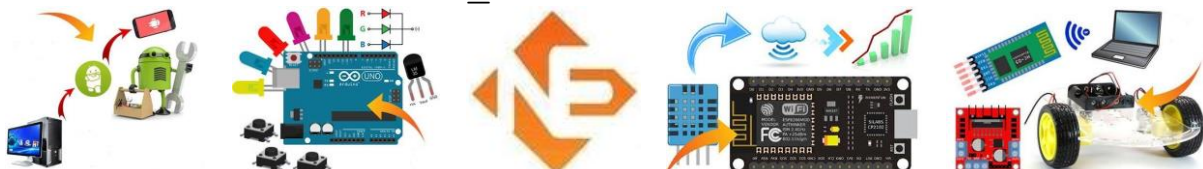
if (forwardState == "on") {
    Serial.println("FORWARD");
    digitalWrite(pwm_right, HIGH);
    digitalWrite(dir_right, LOW);

```





```
digitalWrite(pwm_left, HIGH);  
digitalWrite(dir_left, HIGH);  
delay(2);  
digitalWrite(pwm_right, LOW);  
digitalWrite(dir_right, LOW);  
digitalWrite(pwm_left, LOW);  
digitalWrite(dir_left, LOW);  
delay(8);  
}  
if (leftState == "on") {  
  Serial.println("LEFT");  
  digitalWrite(pwm_right, HIGH);  
  digitalWrite(dir_right, LOW);  
  digitalWrite(pwm_left, HIGH);  
  digitalWrite(dir_left, LOW);  
  delay(8);  
  digitalWrite(pwm_right, LOW);  
  digitalWrite(dir_right, LOW);  
  digitalWrite(pwm_left, LOW);  
  digitalWrite(dir_left, LOW);  
  delay(2);  
}  
if (rightState == "on") {  
  Serial.println("RIGHT");  
  digitalWrite(pwm_right, HIGH);  
  digitalWrite(dir_right, HIGH);  
  digitalWrite(pwm_left, HIGH);  
  digitalWrite(dir_left, HIGH);  
  delay(8);  
  digitalWrite(pwm_right, LOW);  
  digitalWrite(dir_right, LOW);  
  digitalWrite(pwm_left, LOW);  
  digitalWrite(dir_left, LOW);  
  delay(2);  
}  
if (backwardState == "on") {  
  Serial.println("BACKWARD");  
  digitalWrite(pwm_right, HIGH);  
  digitalWrite(dir_right, HIGH);  
  digitalWrite(pwm_left, HIGH);
```



```
digitalWrite(dir_left, LOW);
delay(2);
digitalWrite(pwm_right, LOW);
digitalWrite(dir_right, LOW);
digitalWrite(pwm_left, LOW);
digitalWrite(dir_left, LOW);
delay(8);
}
if (forwardState == "off" && leftState == "off" &&
rightState == "off" && backwardState == "off") {
    digitalWrite(pwm_right, LOW);
    digitalWrite(dir_right, LOW);
    digitalWrite(pwm_left, LOW);
    digitalWrite(dir_left, LOW);
}
}
```

Belajar Melalui Kit Elektronik



Siapa kami? Kami adalah TIM Nadi Eleczone Solutions yang berpusat di Pulau Pinang

NADI adalah suatu PLATFORM untuk menggalakkan perkembangan TEKNOLOGI

Matlamat utama NADI ditubuhkan adalah untuk bersama sama melahirkan generasi yang berinovasi serta mampu mengikuti perkembangan teknologi semasa.

NADI merealisasikan matlamat dengan mereka bentuk kit elektronik sesuai dengan trend teknologi semasa seperti robotik dan Internet of Things.



KOMPONEN ELEKTRONIK NADI menjual komponen elektronik yang menjurus kepada Mikropengawal atau Komputer Dalam Cip, kami mempunyai lebih daripada 300 pilihan komponen elektronik.

SEMINAR TEKNOLOGI NADI menyediakan Kursus dan Seminar bersesuaian dengan trend teknologi semasa iaitu Arduino, Robotik, Internet of Things dan Android Apps.

INOVASI NADI menyediakan penyelesaian berasaskan teknologi terhadap masalah yang dihadapi pelanggannya. Contohnya mereka bentuk sistem penyiraman tanah secara automatik untuk kegunaan pertanian.

Kenali TIM NADI :

Pengurus Am & jualan : NurNadiah

Pengurus Teknikal : Muhammad Firdaus

Pengurus Jurulatih : Azarul Fahmin

Pengurus Pemasaran : Muhammad Ilyasaa

nadiEleczone.lelong.my

www.facebook.com/eleczone

