آکادمی هواه

بررسی پروژه های دوره و رفع اشکال نهایی

IOT Final Project

Dr Ata Jahangir Moshayedi

Associate Prof,
School of information engineering Jiangxi university of science and technology, China
EMAIL: ajm@jxust.edu.cn

CONTENTS

Why IOT enter in various project

- Today the Automation industry is large because of the IoT products available in the market. Automation is there for many years, but the precision is quite low.
- As the IoT evolution took place, all manufacturing devices and machines got smart. This has helped in monitoring and controlling the energy consumption of machines as well the whole factory.

Student Project

Following the shared format for the project and sending the project(out of 5 you should solve 2) on time to email drajm@qq.com

- 1. implement the project and achieve the result
- 2. Write the comment for each line of code
- 3. show the result first one local WEB then on Arduino IOT
- 4. Send the code, show and demo all steps in the clip
- 5.extend the project based on a new application
- 6 your report should have:
  - AIM OF PROJECT
  - Applicable fields
  - Hardware Use in Project
  - Circuit connection
  - Mind map
  - CODE and Comment
  - Result display

IoT-Based Projects

PROJECT

Project1

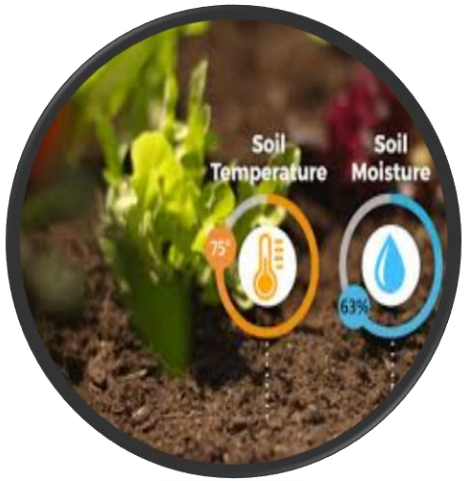
IoT-Based Smart Energy Meter

Project2

IoT-based Vehicle Tracking System

Project3

IoT based Irrigation System



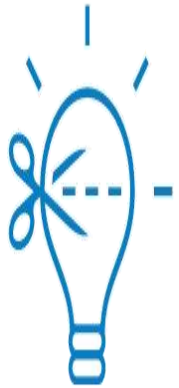
Project4

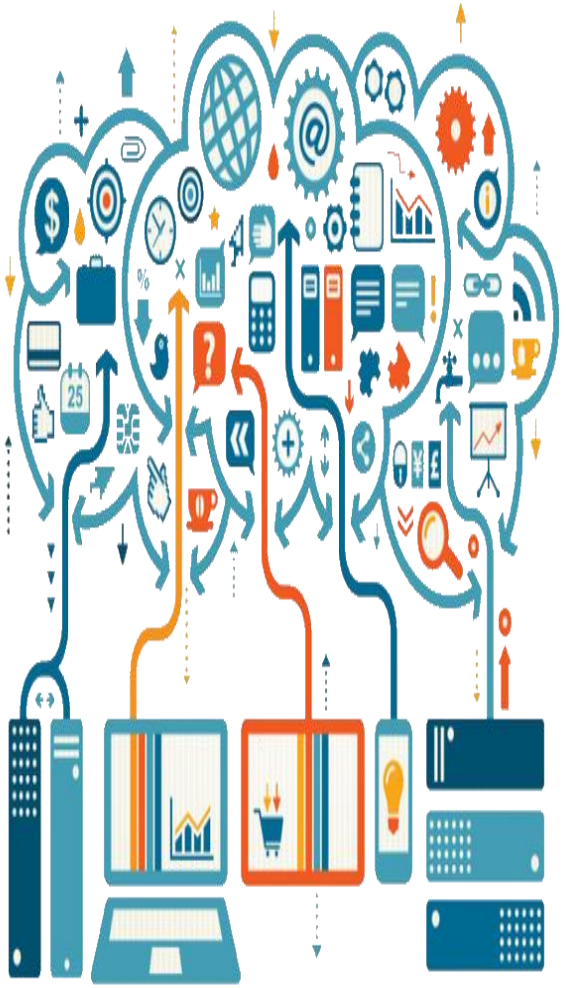
IoT-based car parking System

Project5

IoT-based Gas Detection System(air quality)

پروژه شماره		
1	یک	مصرف بالای انرژی:
		پیمایش مصرف جریان و محاسبه میزان انرژی مصرفی بر روی فضای ابری
2	دو	فضای سبز
		پیمایش میزان رطوبت و دما در ابیاری فضای سبز پارکها بر روی فضای ابری:
3	سه	الودگی هوا:
		پیمایش میزان گازهای الاینده بر روی فضای ابری
4	چهار	ترافیک و استفاده از خودروهای عمومی :
		پیمایش موقعیت اتوبوس واحد و بررسی میزان فاصله مسافر از موقعیت فعلی اتوبوس واحد بر روی فضای ابری





Project1:

IoT Based Smart Energy Meter

Why we need Smart Energy Meter

• Residential Energy

- With the increase in the number of home appliances, the electric bill for consumers has also increased. People are always worried about it, and they are continuously trying to reduce it. IoT offers consumers a good amount of saving with a little investment. Initially, any automation device might be a little costly but it acts as a one-time investment. These devices give instant results and reduce power consumption up to 30%.
- Home Automation devices reduce energy consumption by dimming the lights according to outdoor lighting conditions or by turning off light and fan when no one is in the room. IoT can also help to find the defects in the electrical and electronic system of our homes. It can detect the appliance that is consuming more energy in homes.

• Commercial Energy

- Energy management is equally important to energy saving. Nowadays governments are installing smart energy meters which will directly send the reading to the electricity board, and you'll receive the bill by mail.
- This system no longer needs humans to monitor bills which, will reduce the energy theft. IoT has simplified the way of Energy management with low cost and high precision devices.

Application and Usage

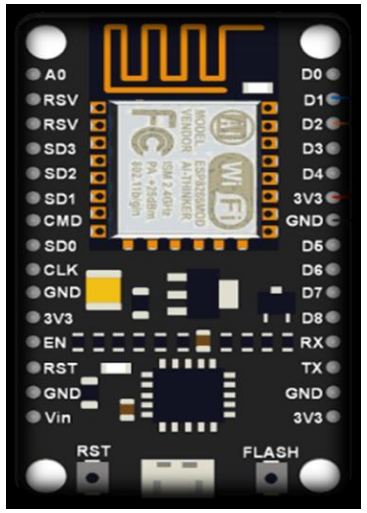
- IOT cloud meters are applied in a wide range of fields, such as residential apartments and buildings, which reduce a lot of burden on property management and operation costs.
-

AIM OF PROJECT

- Electricity meters are connected to the Internet through wireless network, which can conduct remote intelligent control of electricity meters, so that the electricity meter company can achieve integration and intelligence from production and operation, customer service, equipment maintenance, etc.
- Using Arduino IOT cloud to build an intelligent electricity meter based on the Internet of things to monitor the consumption of electric energy

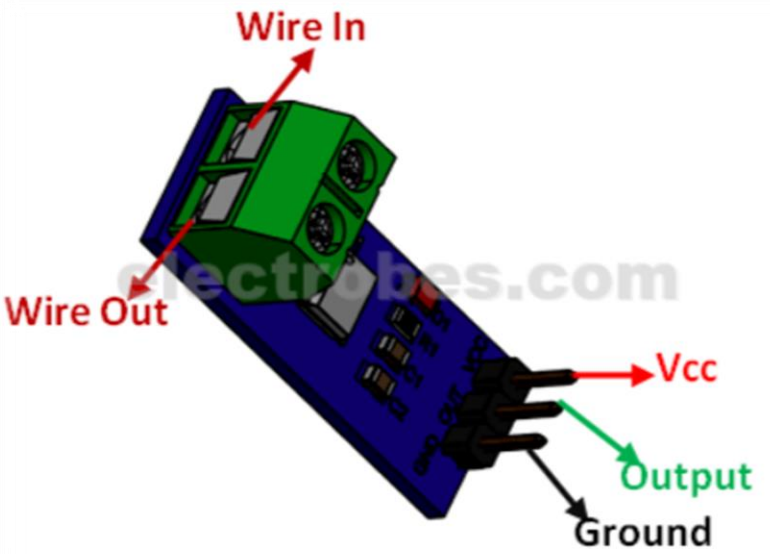
Project 1:Component

1. ACS712 Current Sensor
2. Node MCU ESP8266-12e
3. Jumper Wires
4. Stable load - here 12 watts bulb



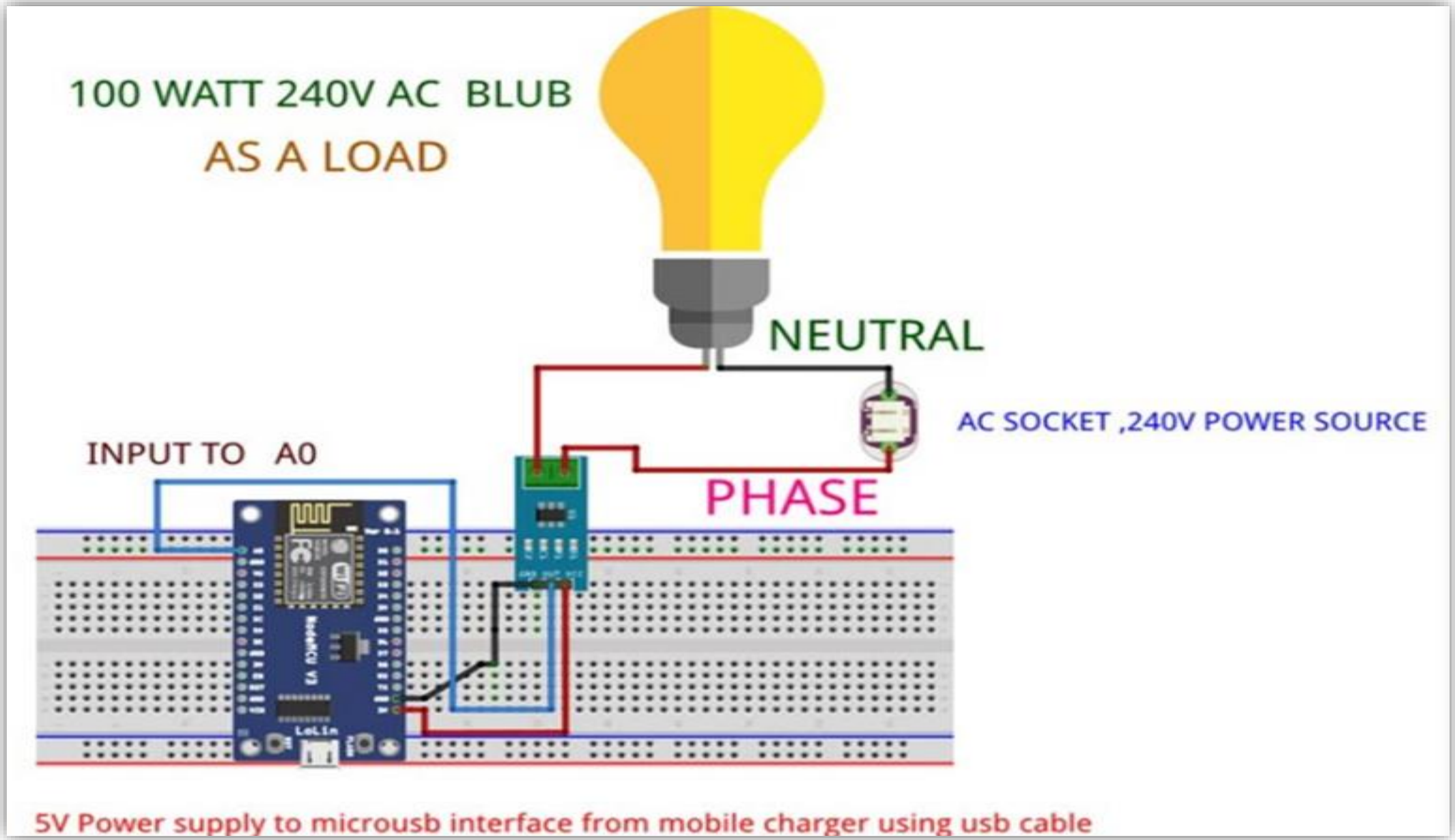
Project 1: Hardware Use in Project

ACS712 Current Sensor:

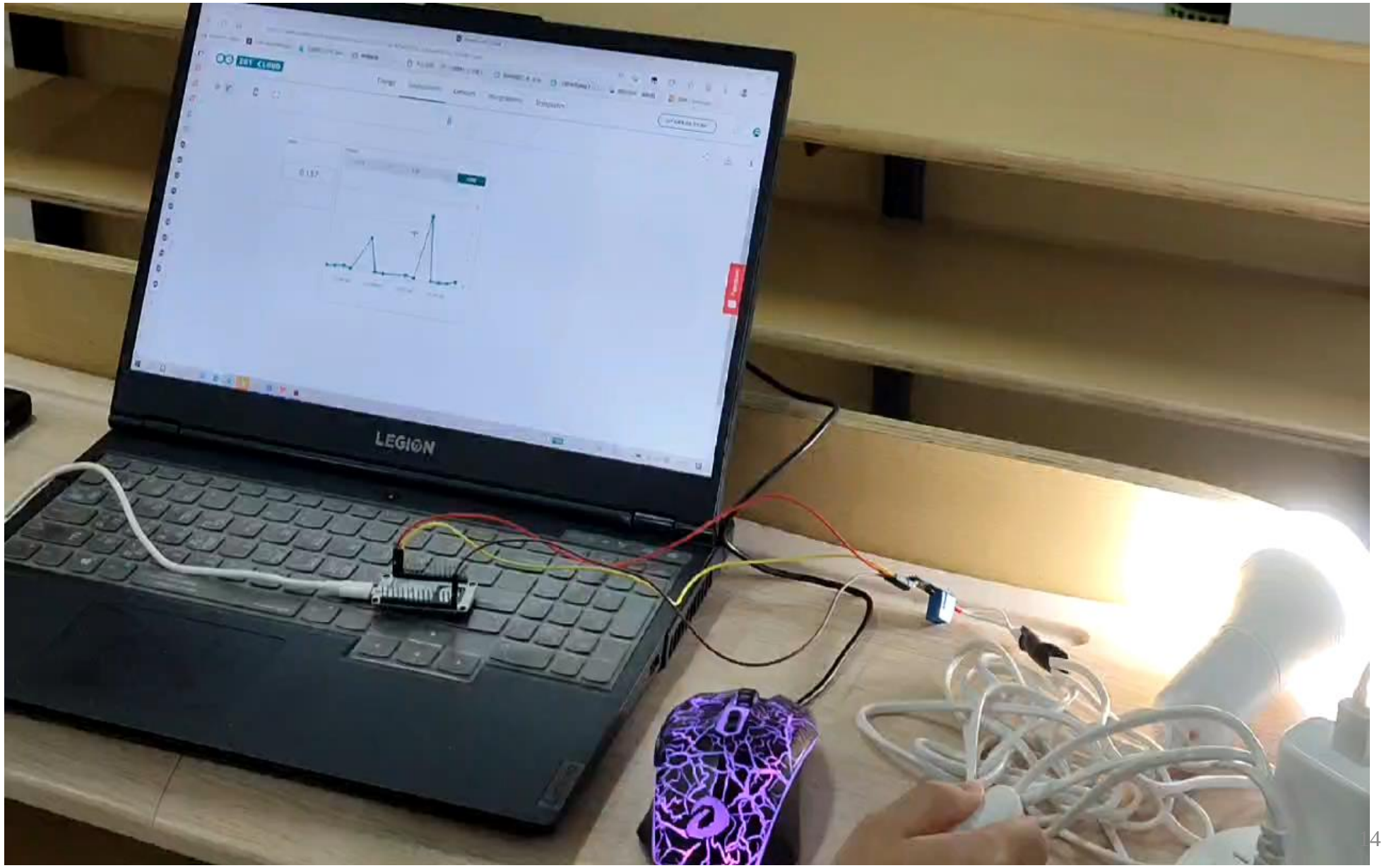


- ACS712 is a current sensor that can operate on both AC and DC. This sensor operates at 5V and produces an analog voltage output proportional to the measured current.
- This tool consists of a series of precision Hall sensors with copper lines.
- The ASC712 output offset voltage is dependable on its operating voltage (generally half of the operating voltage). Since we powered up the module from the ESP 3V output pin the ACS712 module output offset voltage is 1.5 volt (1500 mv) when there is no current flowing. ESP has an onboard voltage divider circuit internally, so we are giving direct input from ACS712 output to the A0 input pin.

Project 1: Hardware Connection



Project 1: Demo Arduino IOT Cloud



Project 1:

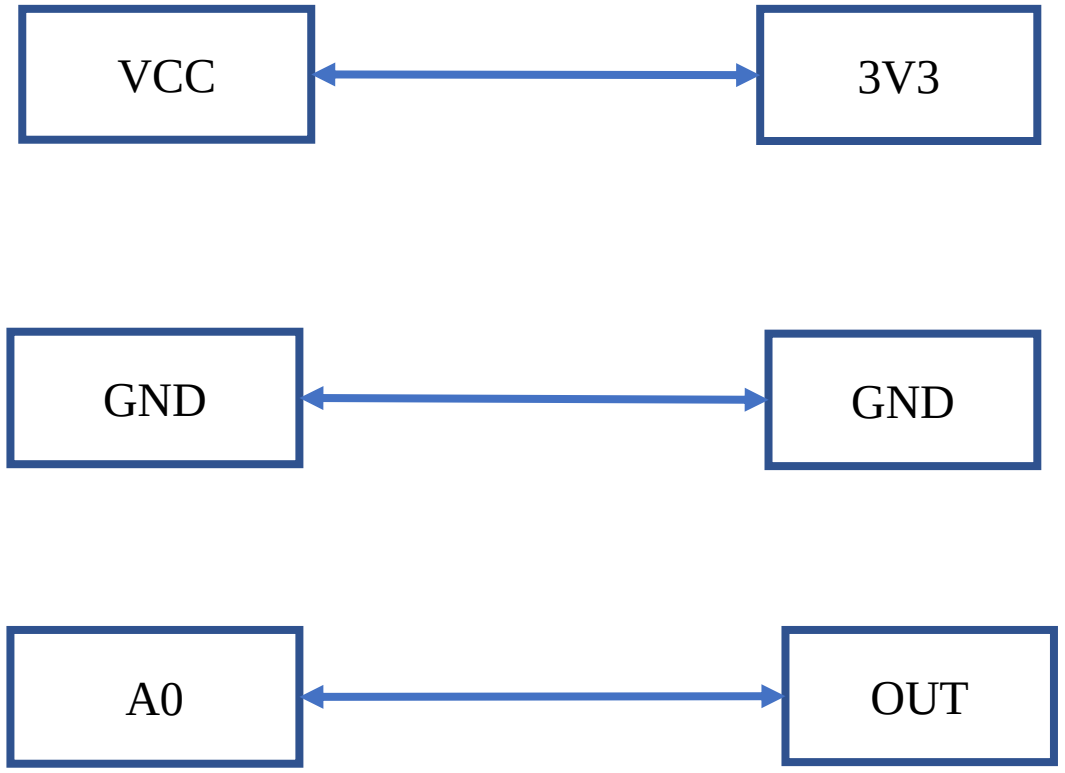

question
answer
question
question
question
question
question


ANSWERS

Project 1: Hardware Connection

Pin Connection:

- As you can see the NodeMCU will be powered through the USB port using a 5V mobile charger and the AC load will be connected to the 220V AC mains through our ACS712 current sensor.
- The sensor has a maximum input voltage on VCC is 5V, but it also works fine in lower voltage. Please note that the ASC712 output offset voltage is dependable on its operating voltage (generally half of the operating voltage). Since we powered up the module from the ESP 3V output pin the ACS712 module output offset voltage is 1.5 volt (1500 mv) when there is no current flowing. ESP has an onboard voltage divider circuit internally, so we are giving direct input from ACS712 output to the A0 input pin.
- Make the connections as shown in the circuit above. I directly soldered the wire between the NodeMCU and ACS712 sensor, but you can also use a breadboard and connecting wires. My setup looks like this below when ready.



Project 1: Brain map

- Iot Based Smart Energy Meter
  - Esp8266
    - function — As a server, processes data and responds to data at the front end
    - Function implementation
      - Using pins to receive data, getVPP() is called to get the voltage
      - According to the voltage calculation, Irms is obtained
      - Send Irms to the forward page
  - IOT cloud
    - create thing
    - Create a variable
    - Create a table and join with variables
    - Display data
    - Run and present the data

Project1:

Code Section

Project 1: CODE

```
#include "ACS712.h"
#include <ESP8266WiFi.h>
#define  WLAN_SSID    "12033"
#define  WLAN_PASS    "120312031234
const int sensorIn = A0;
int mVperAmp = 66;
double Voltage = 0;
double Vp = 0;
double Vrms = 0;
double Irms = 0;
```

//import required libraries

//wifi name

"//wifi pass

//define parameters

//185mV for 5Amp module , 100mV for 10A , 66mv for 20 & 30 Amp module

Project 1: CODE

```
void readData() {
 String data = "{\"current\":\""+ String(current)+"\"}";
 server.send(200, "text/plane", data);
delay(2000);
 current = dht.read.Current();
 Serial.print(current, 1);}
```

readData from HTML

//Send ADC value, temperature and humidity JSON to client ajax request

Project 1: CODE

```
void setup() {
Serial.begin(9600);
delay(50);
initProperties();
ArduinoCloud.begin(ArduinoIoTPreferredConnection);
setDebugMessageLevel(2);
ArduinoCloud.printDebugInfo();
}
```

// Initialize serial and wait for port to open:

// This delay gives the chance to wait for a Serial Monitor without blocking if none is found

//Defined in thingProperties.h

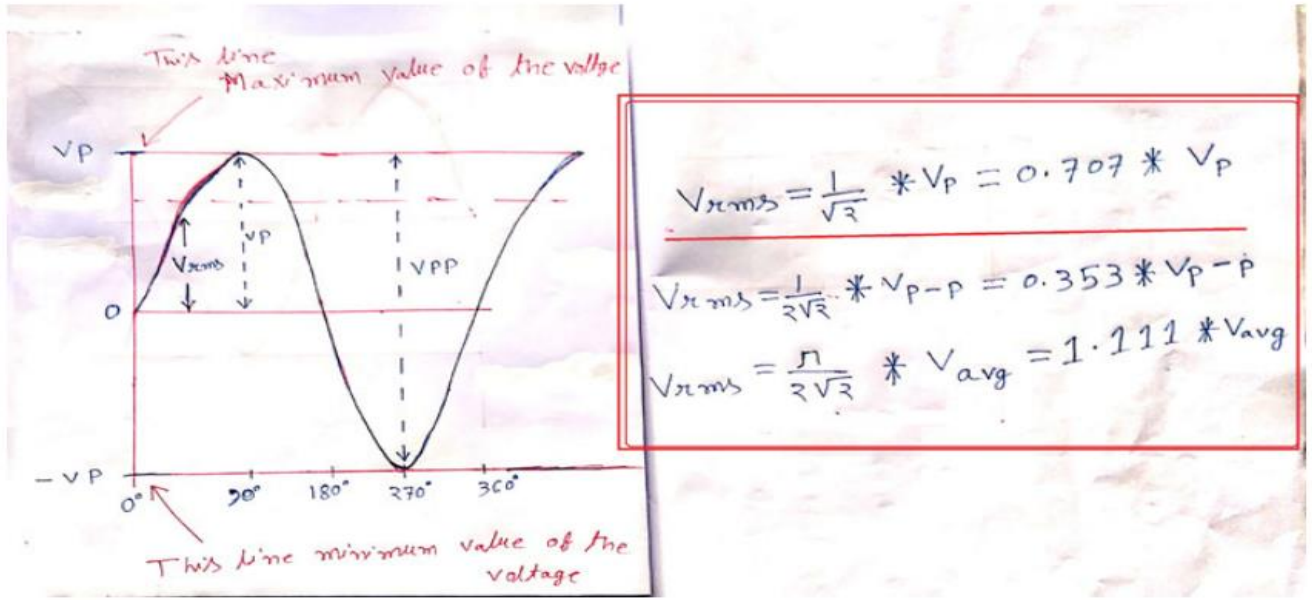
// Connect to Arduino IoT Cloud

Project 1: CODE

If you go through the handmade diagram, I have prepared for you all, you can see that there are two lines, the top line which is the name I have given (Vp) indicates the maximum value of the voltage and bottom is (–Vp) which indicates the minimum value of the voltage. If you take the difference between those two lines, you will get the (Vpp) voltage peak to -peak value.

```
void loop() {
  ArduinoCloud.update();
  Voltage = getVPP();
Vrms = (Voltage / 2.0) * 0.707;
Irms = ((Vrms * 1000) / mVperAmp) ;
 Serial.print(Irms);
 Serial.println(" Amps");
 dian=Irms;
 delay(2000);
}
```

// sq root



Then we return to the main function. Under the loop, you can see we are going to **convert peak voltage to RMS value** using the formulae explained above. Note that we have divided the measured voltage by 2 to get the value of either the positive or negative side.

```
RMS Voltage = Peak Voltage x 0.707
```

Project 1: CODE

After that, we convert the voltage to current, for that we are dividing the Vrms value by the millivolt per amp value of the current sensor (I'm using 30 Amp module so it is 66 mVperamp) and multiply by 1000 so that we get it into the Amp not in milli Amps.

```
Vrms = (Voltage / 2.0) * 0.707; // sq root
  Irms = ((Vrms * 1000) / mVperAmp) ;
  Serial.print(Irms);
  Serial.println(" Amps");
```

Then we will be printing this current value on the serial monitor of Arduino IDE as well as on the MQTT IoT Platform.

```
Serial.print(Irms);
 Serial.println(" Amps");
if (! photocell.publish(Irms))
  {
    Serial.println("Failed");
  }
  else
  {
    Serial.println("OK!");
  }
  delay(2000);
}
```

Project 1: CODE

```
double getVPP()
{
  float result;
  int readValue;
int maxValue = 0;
int minValue = 1024;
 uint32_t start_time = millis();
  while ((millis() - start_time) < 1000)
  {
    readValue = analogRead(sensorIn);
```

//value read from the sensor

// store max value here

// store min value here

//sample for 1 Sec

Project 1: CODE

```
 if (readValue > maxValue)
    {
maxValue = readValue;
    }
    if (readValue < minValue)
    {
minValue = readValue;
    }
 }
result = ((maxValue - minValue) * 5) /
1024.0;
  return result;
}
void onDianChange()  {
}
```

// see if you have a new maxValue

/*record the menimum sensor value*/

/*record the menimum sensor value*/

// Subtract min from max

Project 1: HTML-CODE

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
    <script>
        var Tvalues = [];//Set the tvalues variable
        function getData() {//Define GetData function
            var xhttp = new XMLHttpRequest();//Set xhttp variable and call XMLHttpRequest function
            xhttp.onreadystatechange = function () {
                if (this.readyState == 4 && this.status == 200) {
                    //Push the data in array
                    var txt = this.responseText;
                    var obj = JSON.parse(txt);
```

HTML-CODE

```
Tvalues.push(obj.Current);
                //Update Data Table
                var current = document.getElementById("current");
                current.innerHTML = obj.Current; }};
        xhttp.open("GET", "readData", true); //Handle readData
server on ESP8266
        xhttp.send();}
    </script>
    <style>
        body {
        /*Format body  */
        display: flex;
        justify-content: center;
        align-items: center;
        margin: 0;
        padding: 0; }
```

```
.btn {
        /* Set the width, height, background color and other attributes of BTN */
        width: 200px;          height: 200px;          margin-top: 0px;
margin-left: 153px;
        background: linear-gradient(315deg, #c7d889 0%, #c88403 74%);
        border: none;          border-radius: 100px;          font-family: 'Lato',
sans-serif;
        font-weight: 500;          font-size: 48px;          color: #fff;
        box-shadow: inset 2px 2px 2px 0px rgba(255, 255, 255, .5),
          7px 7px 20px 0px rgba(0, 0, 0, .1),
          4px 4px 5px 0px rgba(0, 0, 0, .1);
        outline: none;          position: relative;
        z-index: 0;}
      .btn:active {
        /* Set the properties of BTN after clicking */
        top: 2px;
      }
```

Project 1: HTML-CODE

```
.button {           /* Set left float */
        float: left;        }
    a {             /* Set to no underline */
        text-decoration: none;        }
    .main {
        /* Set the properties of the main*/
        padding-top: 100px;
        height: 603px;          width: 100%;
        background-image: linear-gradient(to bottom right, #f8c3e8,
#f47990);
    }
    .center {
        /* Set the properties of the center */
        margin: 0 auto;          height: 438px;          width: 500px;
        background-color: rgba(0, 0, 0, .12);
        border-radius: 15px;        }
```

```
h2 {            /* Set the properties of H2 */
        padding-top: 93px;
        text-align: center;
        font-size: 54px;
        color: #fce0ab;        }
  </style>
</head>
<body>
  <div class="main">
    <div class="center">
        <h2>Current magnitude</h2>
        <a href=\"/LED=ON\"\">
            <div onclick="getData()" class="btn"
id="current"></div>
            <!-- Receive data and display -->          </a>
</div>   </div></body></html>
```

Project 1: Arduino IOT

Create Things:

Variables:

Name
dian1

Sync with other Things

Electric Current eg. 1 A

Declaration
```
CloudElectricCurrent dian1;
```

Variable Permission

Read & Write Read Only

Device:

Associated Device

my

ID: 20464094-c087-4391-b735-...

Type: NodeMCU 1.0 (ESP-12E Module)

Status: Offline

Change Detach

Project 1: Arduino IOT

Network: Configure network

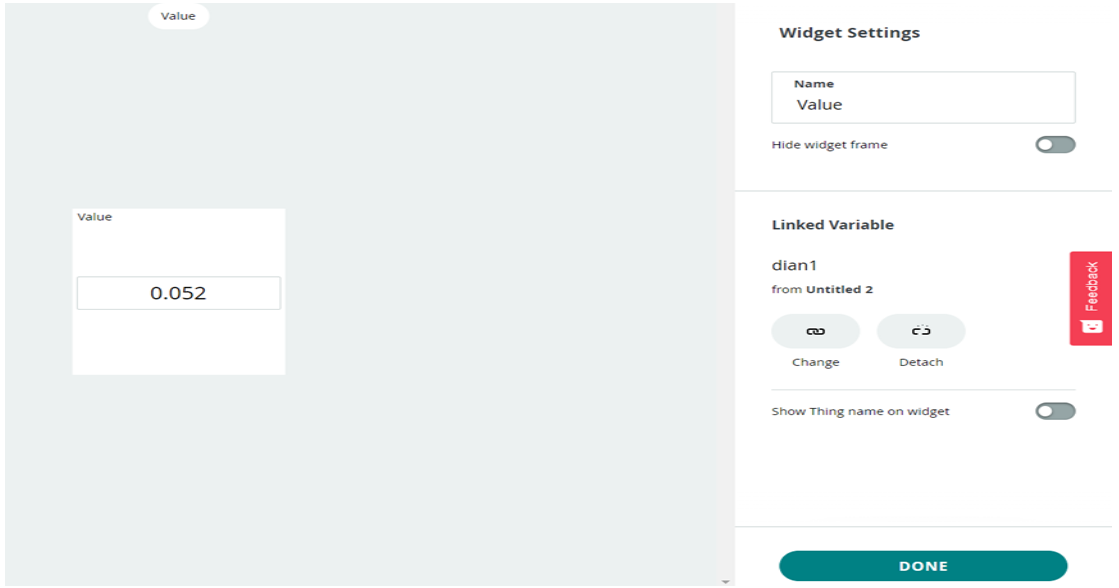
Your will find these network parameters in the secret tab in your sketch, and your device will be able to connect to the network once the sketch will be uploaded.

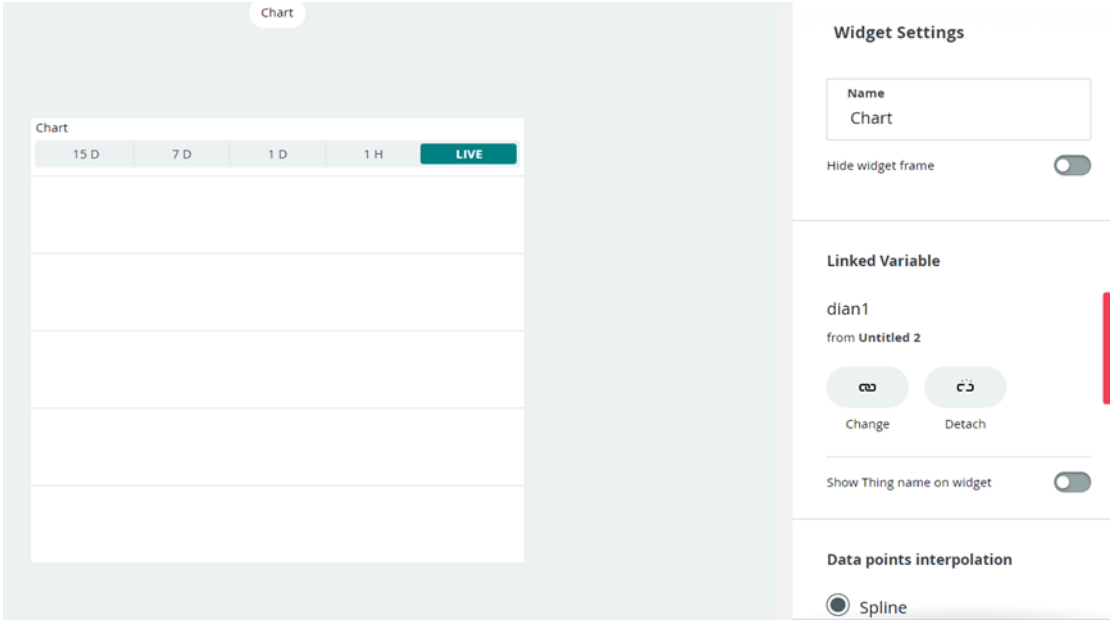
Wi-Fi Name *
12033

Password *
••••••••••••

Secret Key *
••••••••••••••••••••

SAVE

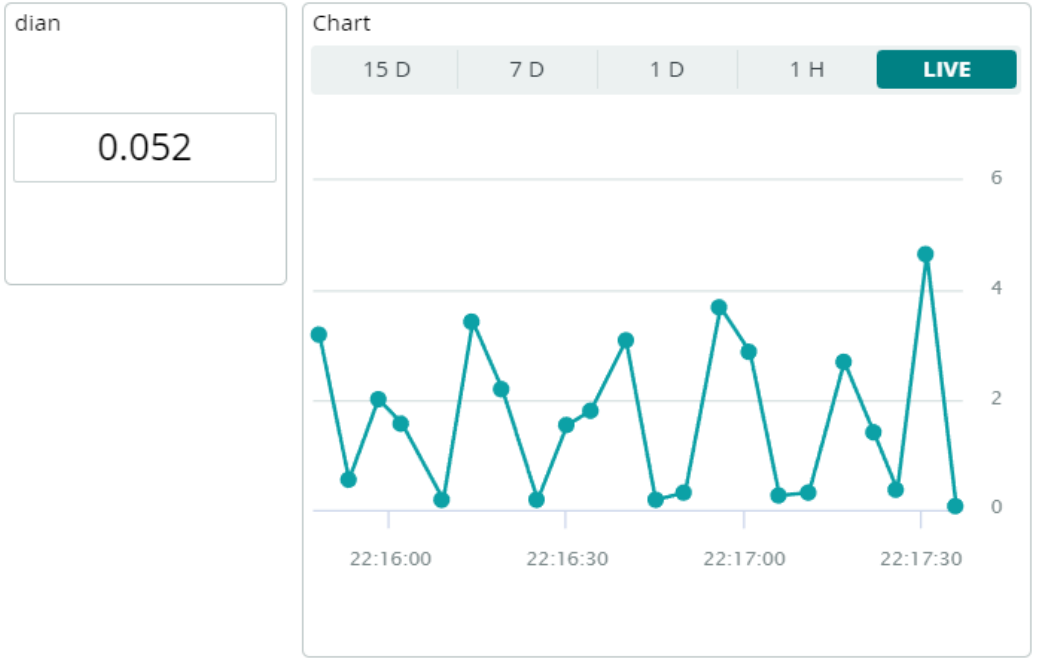




Project 1: Arduino IOT

Running result:

Chart：



Project 1: Testing on Arduino：

project5 | Arduino 1.8.19

文件 编辑 项目 工具 帮助

project5

```
//WiFiClientSecure client;
// Setup the MQTT client class by passing in the WiFi client and MQTT server and login details.
Adafruit_MQTT_Client mqtt(&client, AIO_SERVER, AIO_SERVERPORT, AIO_USERNAME, AIO_KEY);
Adafruit_MQTT_Publish photocell = Adafruit_MQTT_Publish(&mqtt, AIO_USERNAME "/feeds/current");
const int sensorIn = A0;
int mVperAmp = 66; //185mV for 5Amp module , 100mV for 10A , 66mv for 20 & 30 Amp module
double Voltage = 0;
double Vp = 0;
double Vrms = 0;
double Irms = 0;
// Bug workaround for Arduino 1.6.6, it seems to need a function declaration
// for some reason (only affects ESP8266, likely an arduino-builder bug).
void MQTT_connect();
void setup() {
  Serial.begin(9600);
  WiFi.begin(WLAN_SSID, WLAN_PASS);
  while (WiFi.status() != WL_CONNECTED)
  {
    Serial.println("connecting....");
    delay(1000);
  }
  Serial.println("Connected");
}
void loop() {
  MQTT_connect();
  Voltage = getVPP();
    Vrms = (Voltage / 2.0) * 0.707; // sq root
  Irms = ((Vrms * 1000) / mVperAmp) ;
  Serial.print(Irms);
  Serial.println(" Amps");
  if (! photocell.publish(Irms))
  {
    Serial.println("Failed");
  }
  else
  {
    Serial.println("OK!");
  }
  delay(2000);
}
double getVPP()
{
  float result;
  int readValue; //value read from the sensor
  int maxValue = 0; // store max value here
  int minValue = 1024; // store min value here
```

```
Leaving...
Hard resetting via RTS pin...
```

3 NodeMCU 1.0 (ESP-12E Module), 80 MHz, Flash, Disabled (new aborts on oom), Disabled, All SSL ciphers (most compatible), 32KB cache + 32KB IRAM (balanced), Use pgm_read macros for IRA

BANDICAM 班迪录屏 UNREGISTERED

00:00:00

0 bytes / 145.8GB

1920x1080 - (0, 0), (1920, 1080) - 显示器 1

首页 常规 录像 截图 关于

开始 视频 图像

屏幕录制模式 - 全屏

使用此录制模式可以录制全屏内容。

1. 选择全屏录制模式。
2. 然后点击 'REC' 按钮或者摁下录制快捷键。

停止录制 查看在线帮助

录制/停止

录制/停止 F12

捕捉图像 F11

BANDICUT

如何使用屏幕录制模式?

97% 29°C 多云 中 22:19

Project 1: Test on IOT cloud：

https://create.arduino.cc/iot/things

Problem - JustOJ | GDB online Debug... | 江西理工大学 Web... | 哔哩哔哩 (°- °)つ... | 个人空间 | 江西理工大学网上... | 原神地图工具_全地... | 【原神风神瞳】(2.1... | 题目列表 - 编程题 -... | 面板 | Tinkercad

IOT CLOUD

Things | Dashboards | Devices | Integrations | Templates | UPGRADE PLAN

Things

Search and filter Things

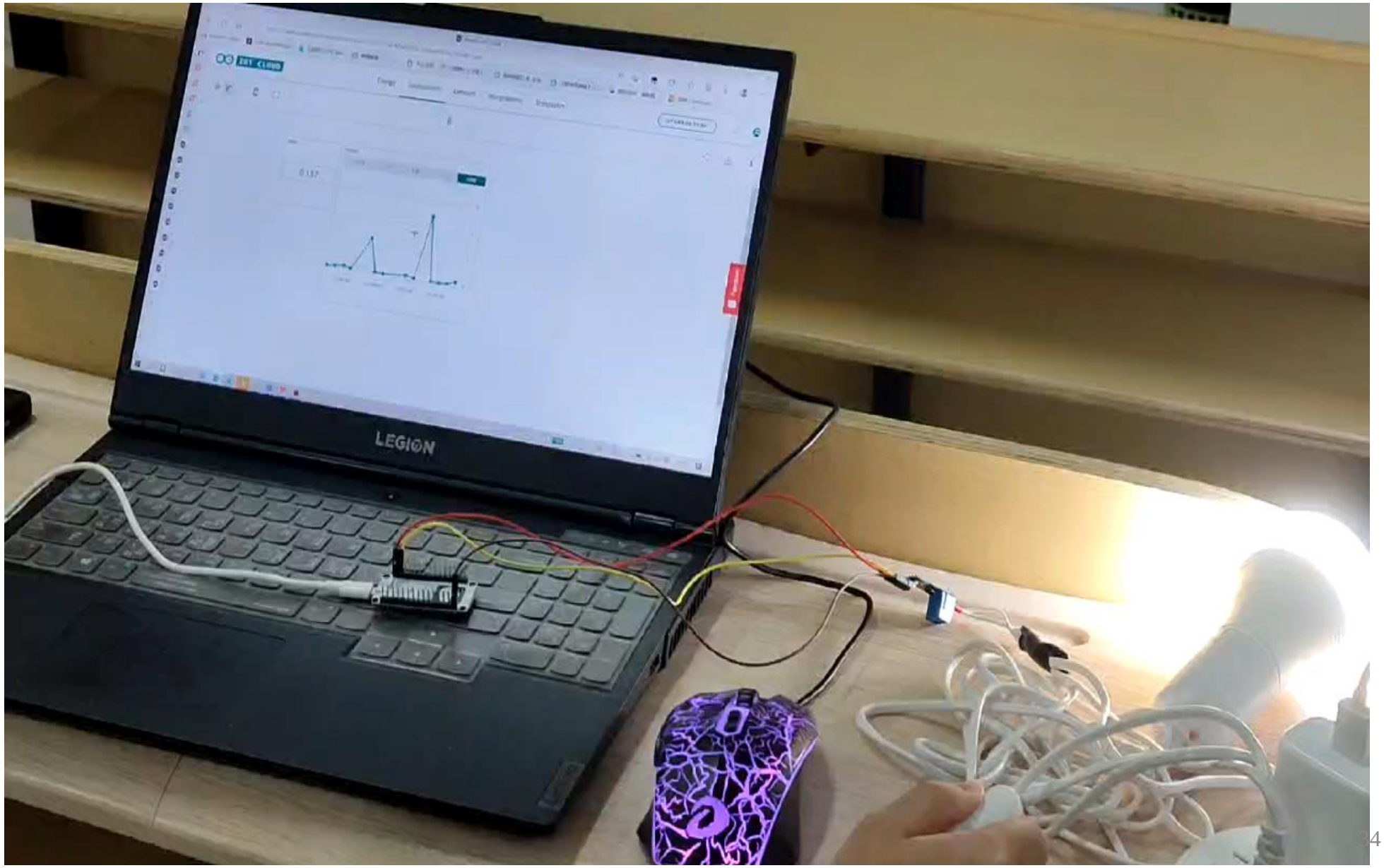
CREATE

Name ↓	Device	Variables	Last Modified
Untitled	my NodeMCU 1.0 (ESP-12E M		

73%

26°C 多云 英 7:35

Project 1: Demo



Project2:

IoT-based Vehicle Tracking System

Why we need …….

01

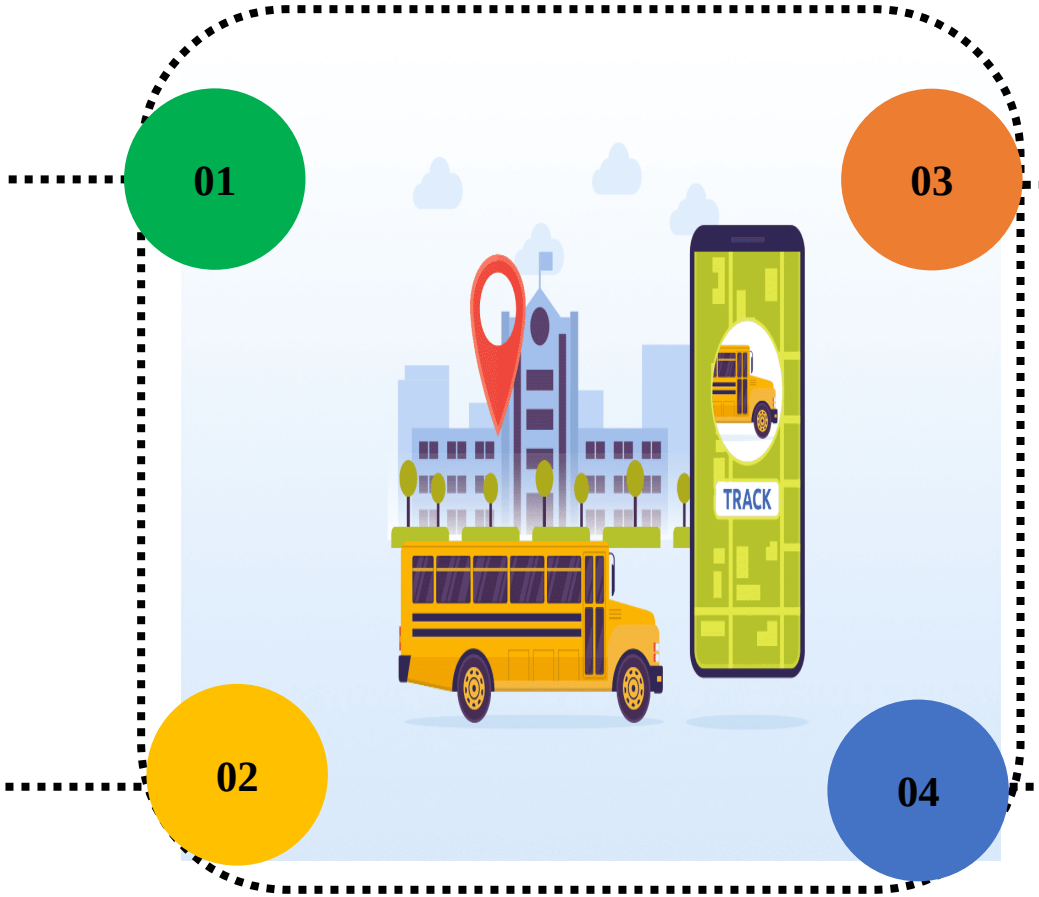
Aim &Background

- Nowadays, security is the most concern for us, whether it is related to our assets like vehicles, homes or our children.
- In this case, GPS tracker devices are very useful. They can be easily used to track the real-time position of the vehicles or assets in case of any emergency like theft, accidents, etc.
- They can also be kept with children to track their location.

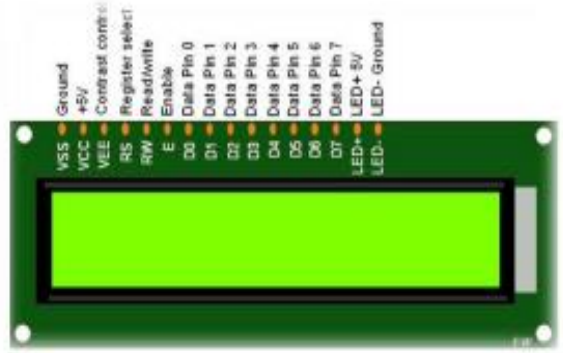
Project 2:Component

NodeMCU ESP8266

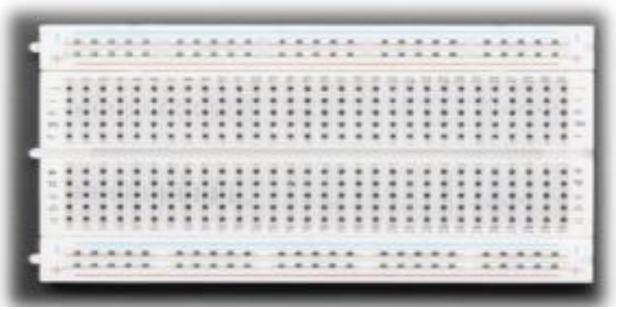
NEO8M GPS Module



16*2 LCD



BreadBoard



Project 2: **Component**

NodeMCU is a open source board which has ESP8266 as a WiFi module .It has file system called SPIFFS to fetch the file like html and so on. It is very easy to use like Arduino.

NodeMCU

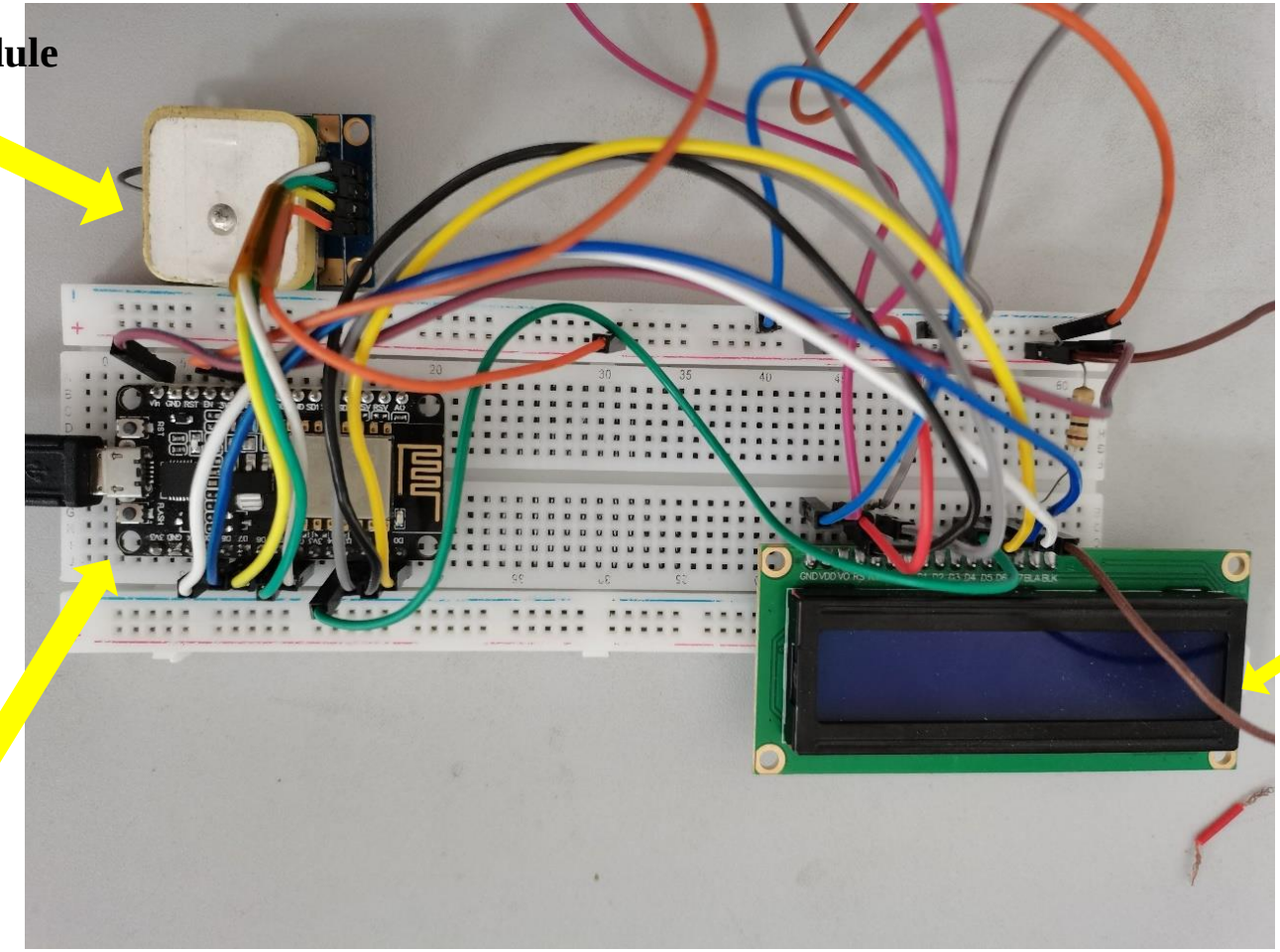
NEO8M GPS Module

	1	2	3	4
Pin	VCC	RX	TX	GND
illustrate	To Power	Receive pin	Translate pin	To Ground

NEO8M GPS Module is a GPS Module,it can use some function of its library to get the longitude and latitude data.The image above shows the function corresponding to its pins.

Project 2: **Connection**

NEO*M GPS Module



16*2 LCD

NodeMCU

Project 2: Demo on Web

From zhao

00:00:00 结束录制

不安全 | 192.168.43.10

code 影视 学习 下载 学校WIFI和工作 娱乐 英语 tool 电子书 红学 tk 服务器 - 轻量应用... gin Matlab2021b哪位... 【软硬结合】Node...

CLASS DESIGN

Get Position Info

欢迎大家观看本组演示
Welcome to this demo

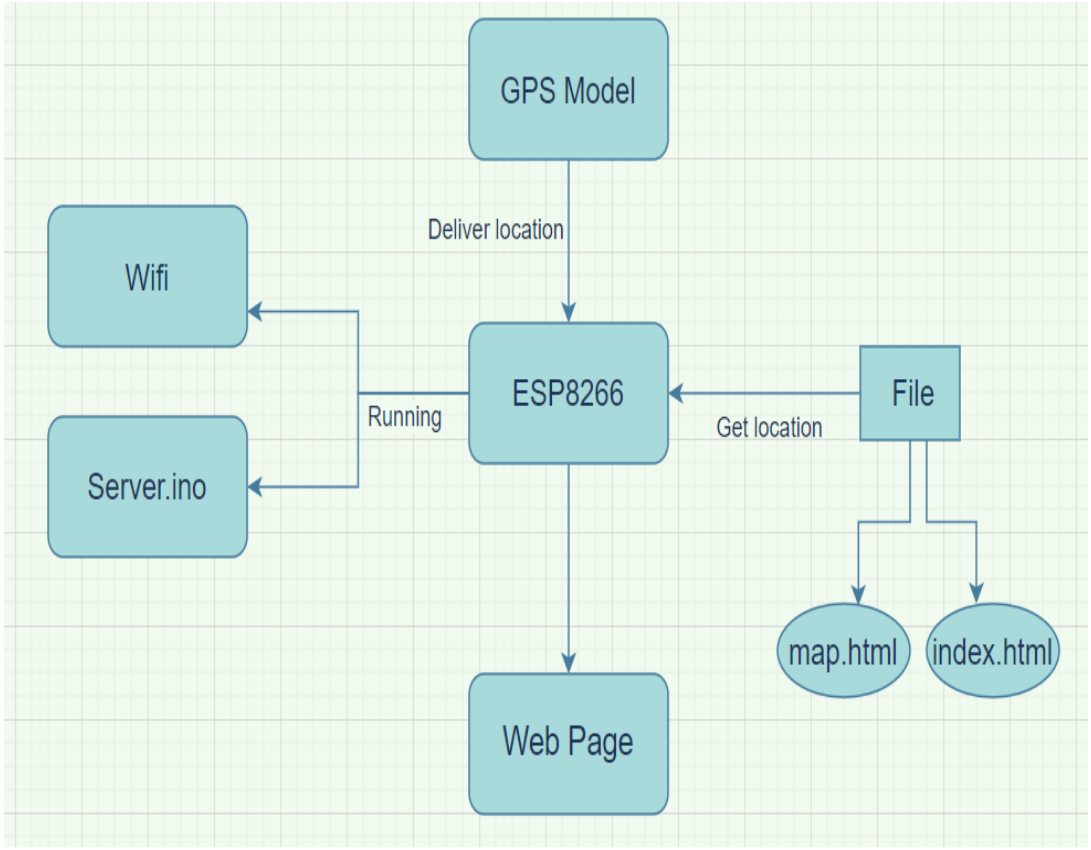
Project 2:


question
answer
question
question
question
question


ANSWERS

Project 2:flow chart and Descriptions

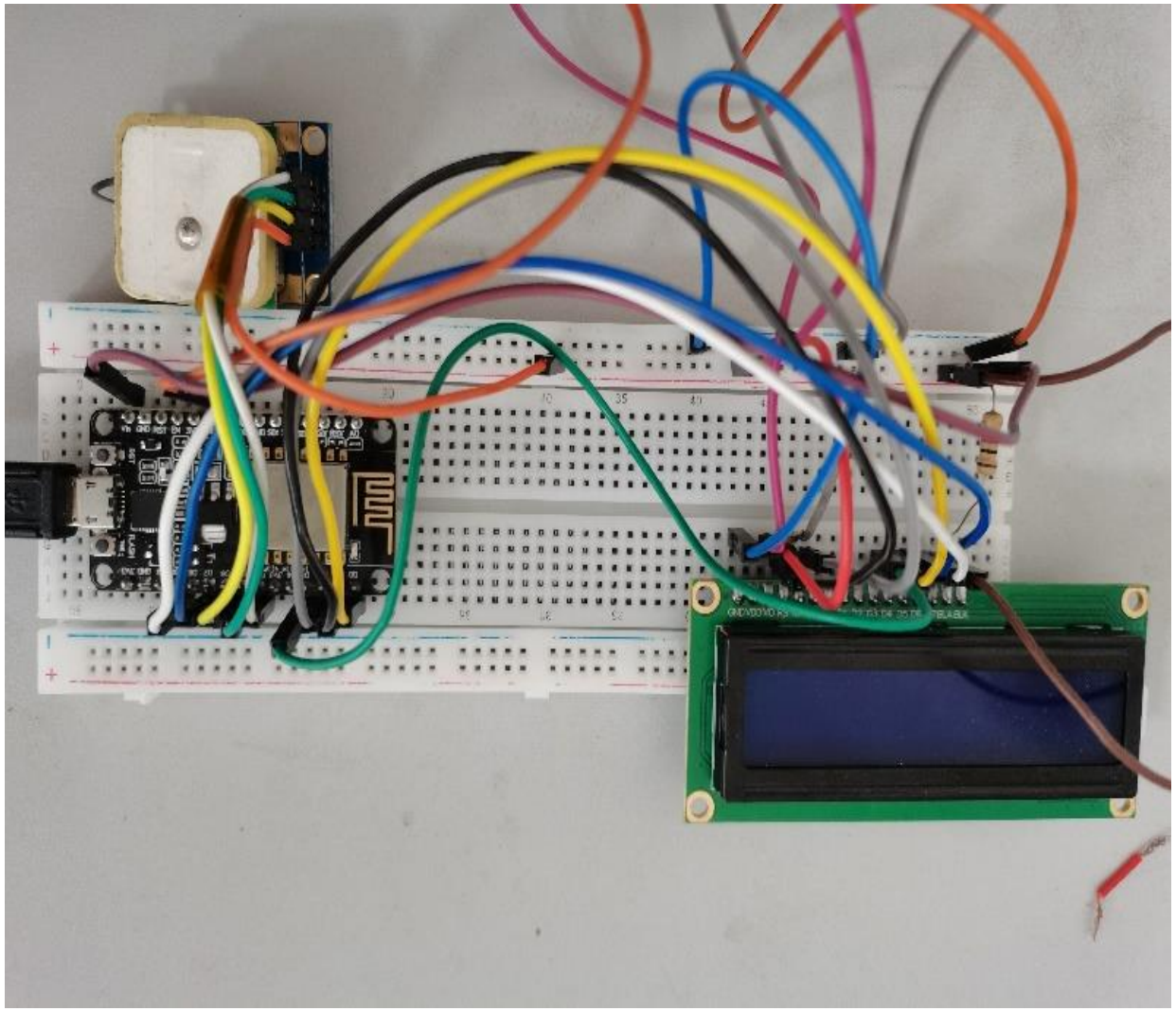
FLOW CHART



Descriptions

- 1,GPS Model will fetch the real-time location to the ESP8266
- 2.And then,in the ESP8266,there are HTML files and some back-end codes.The back end receives the location data from the GPS and processes them.In this part,ESP8266 works as a server,for realizing the function,ESP8266 connects with the WIFI by using the libraries from the third party in the github.Through them,we just need connect the WIFI factitiously,and ESP8266 will connect the same WIFI the next time automatically.After connecting the wifi, we run the server.ino,so ESP8266 is able to run HTML files.
- 3.At last,the IP url will be generated,we can open the web page to check the location in the map.

Project 2: Connection

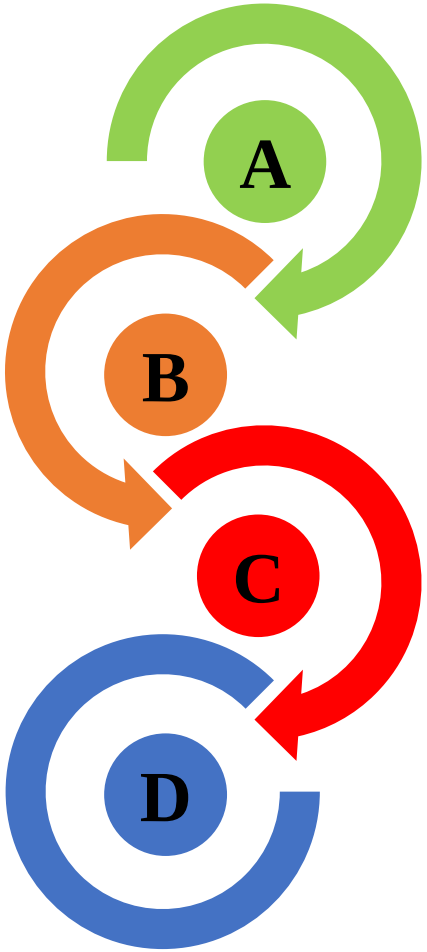


LCD	to	Other Device
GND	——》	GND
VDD	——》	5V
V0	——》	GND
RS	——》	D1
RW	——》	GND
E	——》	D2
D4	——》	D3(NodeMCU)
D5	——》	D0(NodeMCU)
D6	——》	D7(NodeMCU)
D7	——》	RX(NodeMCU)
A	——》	5V
K	——》	GND

NEO8M GPS Module	to	Other Device
RX	——》	D6(NodeMCU)
TX	——》	D5(NodeMCU)
VCC	——》	5V
GND	——》	GND

Project……

- In this project,we have learned how to use IOT furtherly.The location will be shown in the IOT.
- We successfully connect IOT,ESP8266,GPS model.Although,we met many problems and difficulties,like the delivery of the data is very slow,and the data refreshes very slowly.



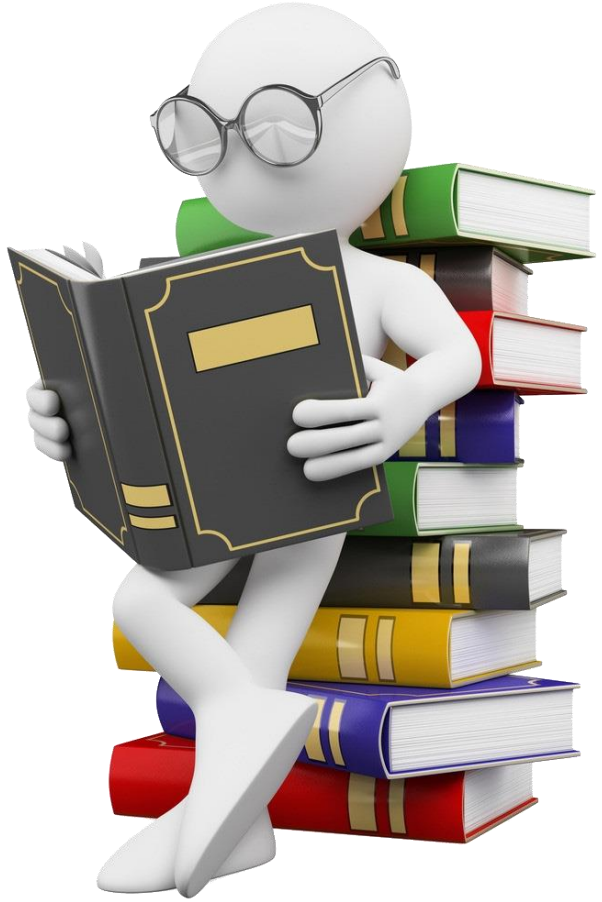
A GPS gets the position

B GPS send the position to the ESP8266

C ESP8266 gets the position and connect IOT

D positon shows in IOT

Library



WiFi manager library

Learn this new function and apply it.It gives great convenience to us when programming

File Flash System

It can read and run the files in the ESP8266 …

Show the position in the map

We make our own web page and the pages
are opened by File Flash System,not print.…

Arduino IDE

Indrtuction

The project is centered on ESP866, using GPS sensors and driven by Arduio built-in libraries and open source third-party libraries. Use Android plug-in to download HTML and other files to the file flash system. Display location information on web pages by responding to client requests.

PROCESS

The following describes the third-party libraries and related code used.

Library

WiFiManager、TinyGPSPlus、ThingSpeak

Arduino plugin

Arduino ESP8266 filesystem uploader

Code

Code and comments

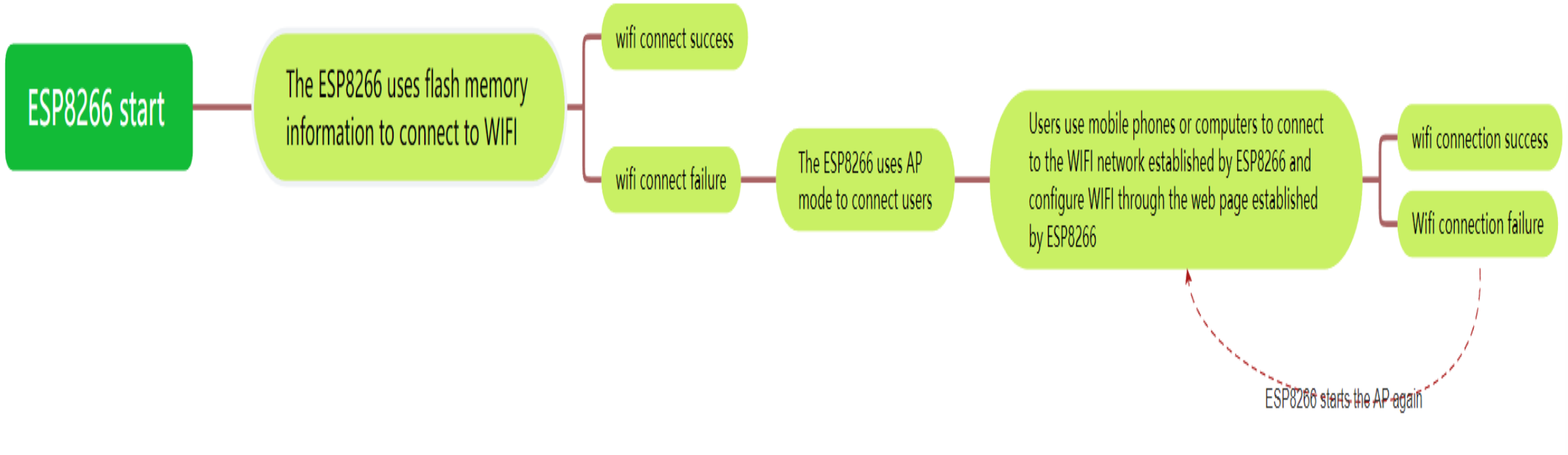
WiFiManager

Espressif ESPx WiFi Connection manager with fallback web configuration portal. The configuration portal is of the captive variety, so on various devices it will present the configuration dialogue as soon as you connect to the created access point.

GitHub：https://github.com/tzapu/WiFiManager

Library WiFiManager

How It Works



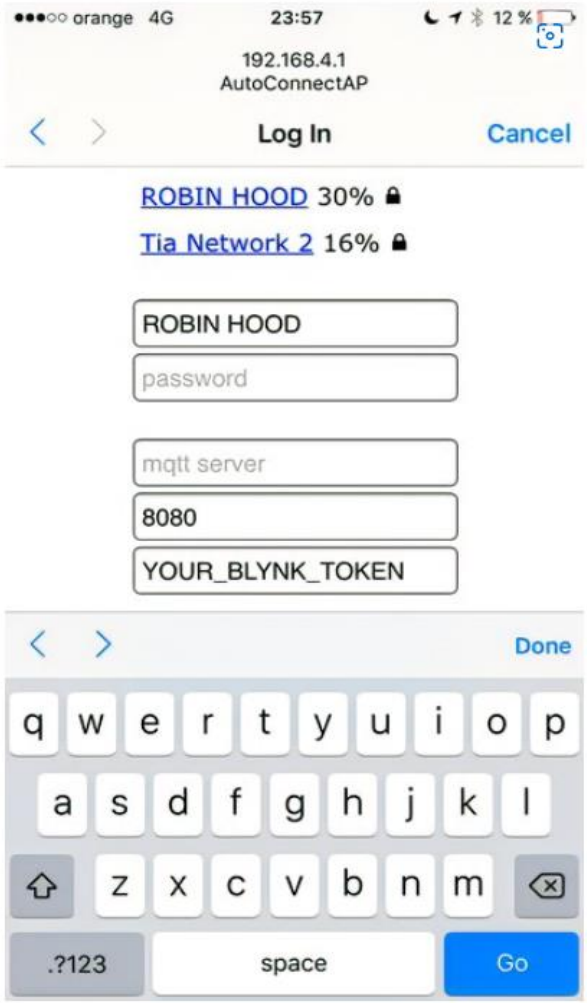
Library WiFiManager

WiFiManager

How It LooKs

README.md

How It Looks



Library WiFiManager

How It Works

- When your ESP starts up, it sets it up in Station mode and tries to connect to a previously saved Access Point
- if this is unsuccessful (or no previous network saved) it moves the ESP into Access Point mode and spins up a DNS and WebServer (default ip 192.168.4.1)
- using any wifi enabled device with a browser (computer, phone, tablet) connect to the newly created Access Point
- because of the Captive Portal and the DNS server you will either get a 'Join to network' type of popup or get any domain you try to access redirected to the configuration portal
- choose one of the access points scanned, enter password, click save
- ESP will try to connect. If successful, it relinquishes control back to your app. If not, reconnect to AP and reconfigure.
- There are options to change this behavior or manually start the configportal and webportal independantly as well as run them in non blocking mode.

Library WiFiManager

WiFiManager

In my code:

```
// establish WiFiManager object
WiFiManager wifiManager;

// Connnectig WiFi automatically
// arguments is te name of conecting ESP8266
wifiManager.autoConnect("AutoConnectAP");
```

With the WiFiManager library, we can set up the ESP8266's WiFi connection without having to change the ESP8266's program.

Library WiFiManager

ThingSpeak

GitHub：https://github.com/mathworks/thingspeak-arduino

This library enables an Arduino or other compatible hardware to write or read data to or from ThingSpeak, an open data platform for the Internet of Things with MATLAB analytics and visualization.
ThingSpeak offers free data storage and analysis of time-stamped numeric or alphanumeric data. Users can access ThingSpeak by visiting http://thingspeak.com and creating a ThingSpeak user account.

ThingSpeak stores data in channels. Channels support an unlimited number of timestamped observations (think of these as rows in a spreadsheet). Each channel has up to 8 fields (think of these as columns in a speadsheet). Check out this video for an overview.

Channels may be public, where anyone can see the data, or private, where only the owner and select users can read the data. Each channel has an associated Write API Key that is used to control who can write to a channel. In addition, private channels have one or more Read API Keys to control who can read from private channel. An API Key is not required to read from public channels. Each channel can have up to 8 fields. One field is created by default.

You can visualize and do online analytics of your data on ThingSpeak using the built in version of MATLAB, or use the desktop version of MATLAB to get deeper historical insight. Visit https://www.mathworks.com/hardware-support/thingspeak.html to learn more.

Library *ThingSpeak*

bool begin(Client & client)

Function: begin

Summary:
Initializes the ThingSpeak library and network settings using the ThingSpeak.com service.

Parameters:
client - EthernetClient, YunClient, TCPClient, or WiFiClient for HTTP connection and WiFiSSLClient, WiFiClientSecure for HTTPS connection created earlier in the sketch.

Returns:
Always returns true

Notes:
This does not validate the information passed in, or generate any calls to ThingSpeak.
Include this ThingSpeak header file after including the client header file and the TS_ENABLE_SSL macro in the user sketch.

Library ThingSpeak

int setField(unsigned int field, String value)

Function: setField

Summary:
Set the value of a single field that will be part of a multi-field update.

Parameters:
field - Field number (1-8) within the channel to set.
value - String to write (UTF8). ThingSpeak limits this to 255 bytes.

Returns:
Code of 200 if successful.
Code of -101 if value is out of range or string is too long (> 255 bytes)

Library ThingSpeak

int writeFields(unsigned long channelNumber, const char * writeAPIKey)

Function: writeFields

Summary:
Write a multi-field update.

Parameters:
channelNumber - Channel number
writeAPIKey - Write API key associated with the channel. *If you share code with others, do _not_ share this key*

Returns:
200 - successful.
404 - Incorrect API key (or invalid ThingSpeak server address)
-101 - Value is out of range or string is too long (> 255 characters)
-201 - Invalid field number specified
-210 - setField() was not called before writeFields()
-301 - Failed to connect to ThingSpeak
-302 - Unexpected failure during write to ThingSpeak
-303 - Unable to parse response
-304 - Timeout waiting for server to respond
-401 - Point was not inserted (most probable cause is the rate limit of once every 15 seconds)

Notes:
Call setField(), setLatitude(), setLongitude(), setElevation() and/or setStatus() and then call writeFields()

Library *TinyGPSPlus*

TinyGPS++ is a new Arduino library for parsing NMEA data streams provided by GPS modules.

Like its predecessor, TinyGPS, this library provides compact and easy-to-use methods for extracting position, date, time, altitude, speed, and course from consumer GPS devices.

However, TinyGPS++'s programmer interface is considerably simpler to use than TinyGPS, and the new library can extract arbitrary data from any of the myriad NMEA sentences out there, even proprietary ones.

GitHub：https://github.com/mikalhart/TinyGPSPlus

Library *TinyGPSPlus*

Uage:

Let’s say you have an Arduino hooked to an off-the-shelf GPS device and you want to display your altitude. You would simply create a TinyGPS++ instance like this:

```
#include "TinyGPS++.h"
TinyGPSPlus gps;
```

Repeatedly feed it characters from your GPS device:

```
while (ss.available() > 0)
  gps.encode(ss.read());
```

Then query it for the desired information:

```
if (gps.altitude.isUpdated())
  Serial.println(gps.altitude.meters());
```

Library *TinyGPSPlus*

TinyGPSPlus

Feeding the Hungry Object:

To get TinyGPS++ to work, you have to repeatedly funnel the characters to it from the GPS module using the encode() method. For example, if your GPS module is attached to pins 4(RX) and 3(TX), you might write code like this:

```
SoftwareSerial ss(4, 3);
void loop()
{
  while (ss.available() > 0)
    gps.encode(ss.read);
  ...
```

After the object has been “fed” you can query it to see if any data fields have been updated:

```
  if (gps.location.isUpdated())
  {
    Serial.print("LAT="); Serial.print(gps.location.lat(), 6);
    Serial.print("LNG="); Serial.println(gps.location.lng(), 6);
  }
} // end loop()
```

Library *TinyGPSPlus*

TinyGPSPlus

```
// GPS
static const uint32_t GPSBaud = 9600;
TinyGPSPlus gps;

static const int RX = D5, TX = D6;
SoftwareSerial soft(RX, TX);
```

Define the global variable GPS with tingpsPlus and soft with SoftwareSerial

```
void RefreshMapData(){
   while (soft.available() > 0){
      if (gps.encode(soft.read())){
         if (gps.location.isValid()){
            double latitude = gps.location.lat();
            double longitude = gps.location.lng();
            latitude_data = String(latitude, 6);
            longitude_data = String(longitude, 6);

            ThingSpeak.setField(1, latitude_data);
            ThingSpeak.setField(2, longitude_data);
            ThingSpeak.writeFields(ch_no, write_api);

            Serial.print(latitude_data);
            Serial.print("    ");
            Serial.println(longitude_data);
            delay(2000);
         }
      }
   }
}
```

Get latitude by GPS sensor

Longitude is obtained by GPS sensor

When the soft serial port receives the data and the data is legal, it receives the double type of data and converts it to the String type, and passes the data into the ThingSpeak Channel through the channel Write API.

Library *FS.h*

- Each ESP8266 comes with a flash drive, much like a small hard drive, in which files we upload are stored. The full name for this Flash is Serial Peripheral Interface Flash File System (SPIFFS).
- In addition to storing uploaded programs, we can also store web files or system configuration files in the ESP8266's flash memory.
- In our program, due to the large amount of HTML file code, we use the file flash system to store a series of our HTML-related files. Through the client request to access the file back to the client to achieve data interaction.
- The HTML file stored in the file flash system, only need to use when open the file to upload to the client, to achieve the front and back end interaction. Effectively avoid code redundancy.

Arduino plugin

Arduino ESP8266 filesystem uploader

Arduino plugin which packs sketch data folder into SPIFFS filesystem image, and uploads the image to ESP8266 flash memory.

Installation

Make sure you use one of the supported versions of Arduino IDE and have ESP8266 core installed.

Download the tool archive from releases page.

In your Arduino sketchbook directory, create directory if it doesn't exist yet. You can find the location of your sketchbook directory in the Arduino IDE at File > Preferences > Sketchbook location.tools

Unpack the tool into directory (the path will look like .tools<sketchbook directory>/tools/ESP8266FS/tool/esp8266fs.jar)

Restart Arduino IDE.

On OS X and Linux based OSs create the tools directory in ~/Documents/Arduino/ and unpack the files there

This project uses this plugin to store HTML related files in a file flash system.

GitHub：https://github.com/esp8266/arduino-esp8266fs-plugin

Demo


From zhao
00:00:00
结束录制
不安全
192.168.43.10
code
影视
学习
下载
学校WIFI和工作
娱乐
英语
tool
电子书
红学
tk
服务器 - 轻量应用...
gin
Matlab2021b哪位...
【软硬结合】Node...
CLASS DESIGN
Get Position Info
欢迎大家观看本组演示
Welcome to this demo

Project2:

Code Section

Project 2: Code

```
#include <LiquidCrystal.h>
#include <ESP8266WiFi.h>
#include <ESP8266WebServer.h>
#include <WiFiManager.h>
#include <FS.h>
#include <TinyGPS++.h>
#include <SoftwareSerial.h>
#include "ThingSpeak.h"


ESP8266WebServer server(80);
WiFiClient client;
LiquidCrystal lcd = LiquidCrystal(D1, D2, D3, D0, D7, 3);
```

//Server
//Clent
//LCD

Project 2: Code

```
static const uint32_t GPSBaud = 9600;
TinyGPSPlus gps;                                    //GPS

// ThingSpeak
unsigned long ch_no = 1784543;    //Replace with Thingspeak Channel number
const char *write_api = "YLML7A9TRV8RLG5X";    // Replace with
                                                   //Thingspeak write API

//software serial
static const int RX = D5, TX = D6;
SoftwareSerial soft(RX, TX);

                                                   //longitude
                                                   //latitude
String longitude_data = "114.490686";
String latitude_data = "25.612273";
```

Project 2: Code

```
void handleUserRequet();
void handleMapDateLongitude();
void handleMapDateLatitud();
String getContentType(String filename);
bool handleFileRead(String path);
void RefreshMapData();


void setup()
{
    Serial.begin(4800);
    soft.begin(GPSBaud);
    lcd.begin(16, 2);
    WiFiManager wifiManager;
```

// Serial Init
//softwareserial Init
//lcd Init
// establish WiFiManager object

Project 2: Code

```
wifiManager.autoConnect("AutoConnectAP");          //arguments is the name of
                                                     conecting ESP8266
Serial.print("WiFi Connected!");
lcd.setCursor(2, 0);
lcd.print("WiFi Connected!");     //Set the delay to facilitate reading of LCD display data
delay(1500);
lcd.clear();


lcd.setCursor(2, 0);
String s1 = "WiFi SSID:";          //Set the delay to facilitate reading of LCD display data
s1 += WiFi.SSID();
lcd.print(s1);
delay(1500);
lcd.clear();
```

Project 2: Code

```
// Start text flash system and output related information
if (SPIFFS.begin())                          // Start the flash file system
  {
     Serial.println("SPIFFS Started.");


     lcd.setCursor(0, 0);
     lcd.print("SPIFFS Succeed");
     lcd.setCursor(0, 1);
     lcd.print("to Start.");
     delay(1500);
     lcd.clear();
  }
```

Project 2: Code

```
else
{
    Serial.println("SPIFFS Failed to Start.");

    lcd.setCursor(0, 0);
    lcd.print("SPIFFS Failed");
    lcd.setCursor(0, 1);
    lcd.print("to Start.");
    delay(1500);
    lcd.clear();
}
```

Project 2: Code

```
    // Set the server response to start the server
    server.on("/readMapLongitude", handleMapDateLongitude);
    server.on("/readMapLatitud", handleMapDateLatitud);
    server.onNotFound(handleUserRequet);
    server.begin();
    Serial.println("HTTP server started");


lcd.setCursor(2, 0);
    lcd.print("HTTP server ");
    lcd.setCursor(3, 1);
    lcd.print("started");
    delay(1500);
    lcd.clear();
```

//Start web service

// The serial port outputs the website service startup information

//The LCD outputs the website service startup information

Project 2: Code

```
    ThingSpeak.begin(client);                    //Start ThingSpeak

    Serial.println("ThingSpeak begin");
    lcd.setCursor(0, 0);
    lcd.print("ThingSpeak begin");
    delay(1500);
    lcd.clear();
}                                                //setup end
```

Project 2: Code

```
void loop()
{
    RefreshMapData();


    lcd.setCursor(0, 0);
    lcd.print("long:");
    lcd.print(longitude_data);
    lcd.setCursor(0, 1);
    lcd.print("lat:");
    lcd.print(latitude_data);
    delay(2000);
    lcd.clear();
```

// Update the latitude and longitude and send it to ThingSpeak.This a functio.

// Output latitude and longitude information through LCD

Project 2: Code

```
 server.handleClient();
}


void handleUserRequet()
{
    String webAddress = server.uri();
    bool fileReadOK = handleFileRead(webAddress);


     {
        server.send(404, "text/plain", "404 Not Found");
    }
}
```

// Processing user requests

// Handles HTTP access to the user's browser

//Get user request url information
// User access is handled
//through the handleFileRead function

Project 2: Code

```
// Check whether the file exists
bool handleFileRead(String path)
{
    if (path.endsWith("/"))
    {
       path = "/index.html";
    }
    String contentType = getContentType(path);
    if (SPIFFS.exists(path))
    {
       if (SPIFFS.exists(path))
```

// If the access address ends with "/"

// Change the access address to /index.html
//for SPIFFS access

// If the accessed file can be found in SPIFFS

Project 2: Code

```
    server.streamFile(file, contentType);
   file.close();
   return true;
   }
   return false;
}
String getContentType(String filename)
{
    if (filename.endsWith(".htm"))
        return "text/html";
    else if (filename.endsWith(".html"))
        return "text/html";
    else if (filename.endsWith(".css"))
```

```
//Attempts to open the file
// And returns the file to the browser
// And close the file
```

```
// If the file is not found, return false
//Get the file type
```

Project 2: Code

```
return "text/css";
    else if (filename.endsWith(".js"))
        return "application/javascript";
    else if (filename.endsWith(".png"))
        return "image/png";
    else if (filename.endsWith(".gif"))
        return "image/gif";
    else if (filename.endsWith(".jpg"))
        return "image/jpeg";
    else if (filename.endsWith(".ico"))
        return "image/x-icon";
    else if (filename.endsWith(".xml")
        return "text/xml";
    else if (filename.endsWith(".pdf"))
        return "application/x-pdf";
    else if (filename.endsWith(".zip"))
        return "application/x-zip";
    else if (filename.endsWith(".gz"))
        return "application/x-gzip";
    return "text/plain";
}
```

Project 2: Code

```
// Update latitude and longitude information and send it to ThingSpeak
void RefreshMapData()
{
    while (soft.available() > 0)
    {
        if (gps.encode(soft.read()))
        {
            if (gps.location.isValid())
            {
                double latitude = gps.location.lat();
                double longitude = gps.location.lng();
                latitude_data = String(latitude, 6);
```

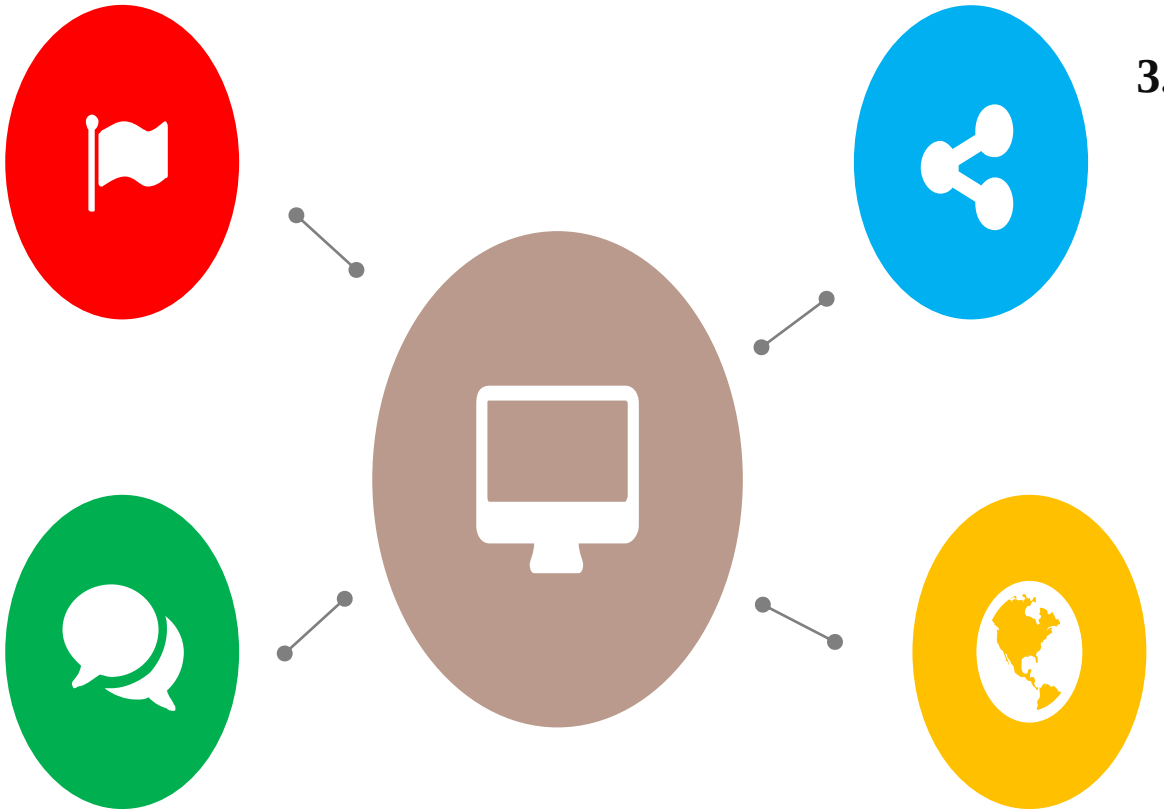
Project 2: Code

```
            ThingSpeak.setField(1, latitude_data);
            ThingSpeak.setField(2, longitude_data);
            ThingSpeak.writeFields(ch_no, write_api);

            Serial.print(latitude_data);
            Serial.print("    ");
            Serial.println(longitude_data);
            delay(2000);
        }
    }
  }
}
```

```
//Send longitude information
void handleMapDateLongitude()
{
   server.send(200, "text/plain", longitude_data);
}

//Send latitude information
void handleMapDateLatitud()
{
   server.send(200, "text/plain", latitude_data);
}
```

Project 2: HTML

1.Set j basic style



3.Creating a Map

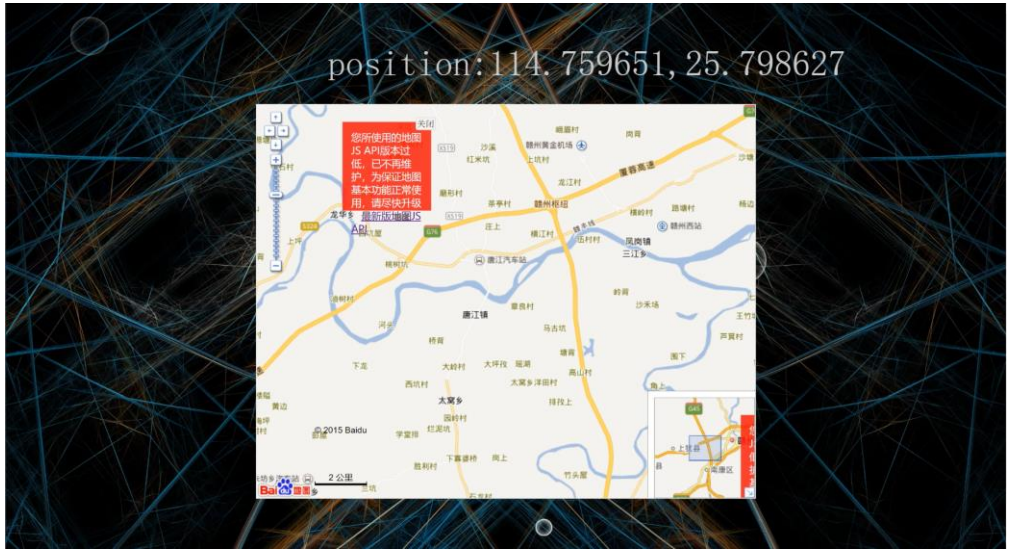
2. Connect ESP8266 to HTML

4. Display location information

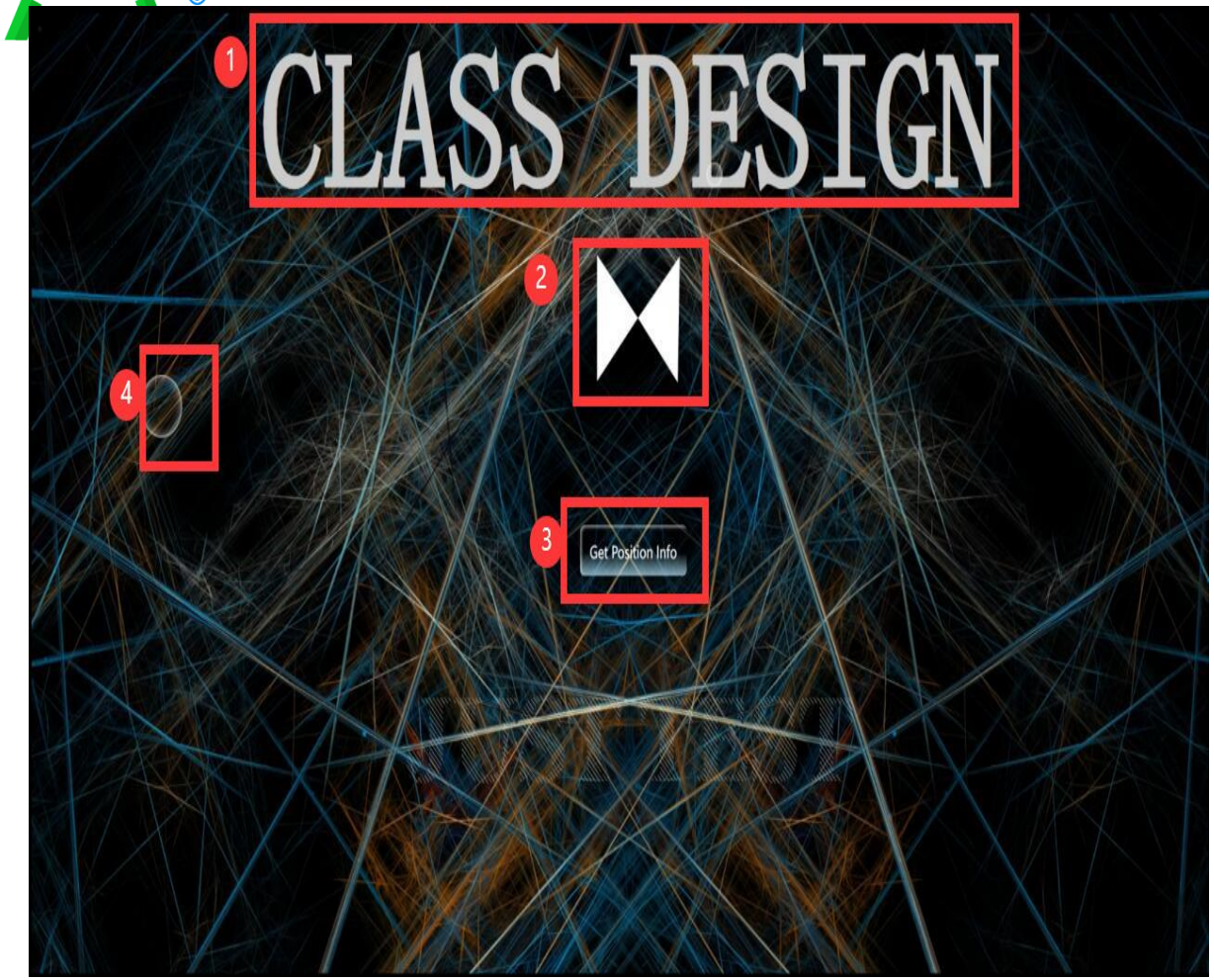
1.Setting basic Styles

The first is the welcome page, which has a title and a button and a rotating log

When you click the welcome page button, you jump to the map page, where the location information is displayed.
The location of the vehicle will be displayed at the center of the map, which is not an online map,
The apiid called baidu map is a static page and the map is set to one second Refresh once, each refresh is based on the latitude and longitude to determine the location of the map center



1.Setting basic Styles



The page will be divided into four parts to explain

1. Add a title and background
2. Add the log
3. Add a rotating button
4. Add floating bubbles

1.Setting basic Styles

1.Add title and background

```
h1 {
      text-align: center;                    //In the center
      color: #cccccc;                        //set color
      font-size: 150px;                       //set size
      font-family: "Cormorant Garamond", serif;   //set font
   }
```

Set their styles in the CSS module and add them to the body

```
body {
      background-image: url(https://w.wallhaven.cc/full/q2/wallhaven-q2pexd.jpg); /*Set the path of the image, the image source network*/
      background-repeat: no-repeat;                    /*Make the image never repeat no matter what size */
      background-size: 100%;                          /*set background size*/
   }
```

1.Setting basic Styles

2.add log

```
.kinetic::after,
    .kinetic::before {                          //Sets the style of the rotation body
        content: '';
        position: absolute;
        top: 0;
        left: 0;
        width: 0;
        height: 0;
        border: 50px solid transparent;
        border-bottom-color: #fff;
        animation: rotateA 2s linear infinite 0.5s; //Let A rotate indefinitely
    }

    .kinetic::before {
        transform: rotate(90deg);
        animation: rotateB 2s linear infinite;  //Set B to rotate indefinitely
    }
    //Add A and B to the triangle of rotation
    @keyframes rotateA {                        //Set the position of A when rotated
                                    //to A different Angle
        0%,
        25% {
            transform: rotate(0deg);
        }
        50%,
        75% {
            transform: rotate(180deg);
        }

        100% {
            transform: rotate(360deg);
        }
    }

    @keyframes rotateB { //Set the position of A when  rotated
                                    //to A different Angle
        0%,
        25% {
            transform: rotate(90deg);
        }

        50%,
        75% {
            transform: rotate(270deg);
        }

        100% {
            transform: rotate(450deg);
        }
    }
```

1.Setting basic Styles

3. add button

//Add a button in the body

```
<button class="custom-btn btn" onclick=window.location.href='map.html'><span>Click!</span><span>Get Position Info</span></button>
```

//Add button background and dimensions

```
.btn {
        position: relative;
        right: 20px;
        bottom: 20px;
        border: none;
        box-shadow: none;
        width: 130px;
        height: 40px;
        line-height: 42px;
        -webkit-perspective: 230px;
        perspective: 230px;
    }
```

1.Setting basic Styles

4. Add bubble

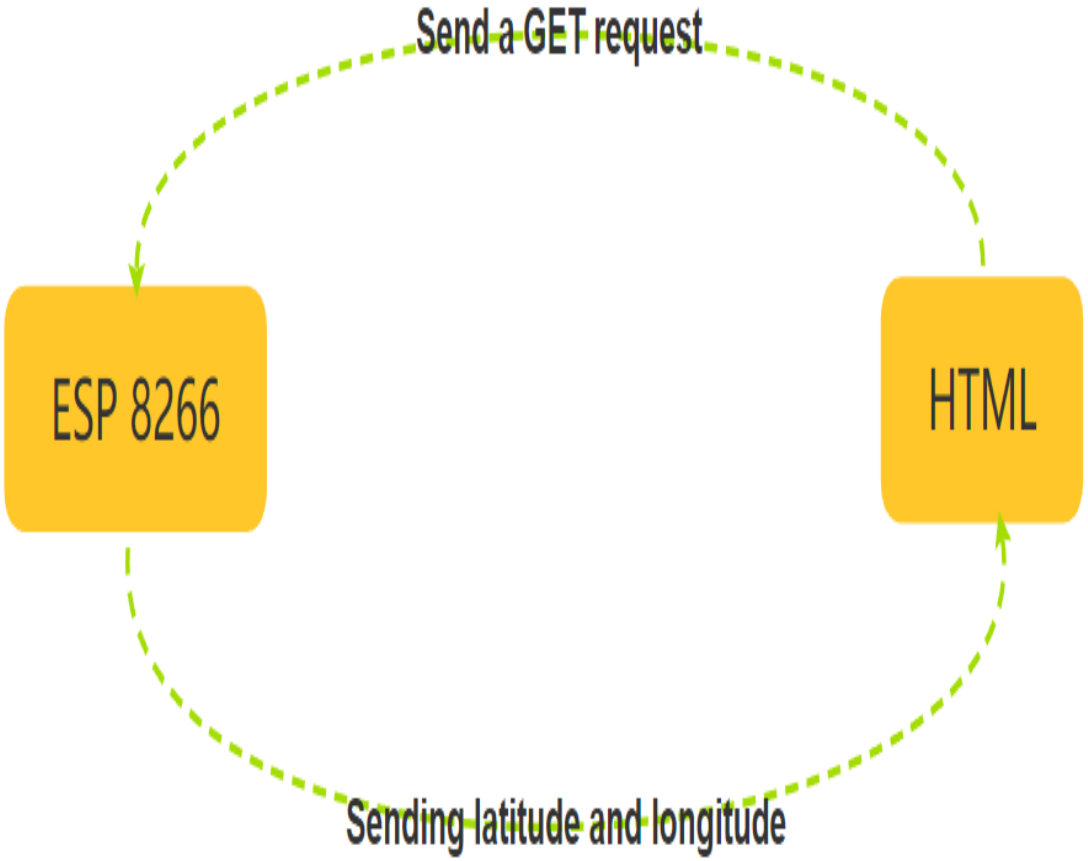
// Sets the basic style of the button

```
.bubble {                          //Sets the basic style
      position: absolute;
      border-radius: 50%;
      border: 2px solid #fff;
      box-shadow: inset 0 0 8px #fff;
      animation: flutter 10s infinite;
      opacity: 0;
   }

   @keyframes flutter {          //Let the bubbles fly up
      0% {
         transform: translateX(0);
         bottom: -100px;
         opacity: 1;
      }
      50% {
         transform: translateX(100px);
         opacity: 0.5;
      }

      100% {
         transform: translateX(0px);
         bottom: 100%;
         opacity: 0;

      }
   }
```

//Add multiple bubbles

```
.bubble:nth-child(1) {
      left: -10%;
      width: 50px;
      height: 50px;
      animation-duration: 9s;
      animation-delay: 0.1s;
   }

   .bubble:nth-child(2) {
      left: 15%;
      width: 20px;
      height: 20px;
      animation-duration: 6s;
      animation-delay: 1.5s;

   }

   .bubble:nth-child(3) {
      left: 20%;
      width: 60px;
      height: 60px;
      animation-duration: 10s;

   }
```

2.Connect ESP8266 to HTML



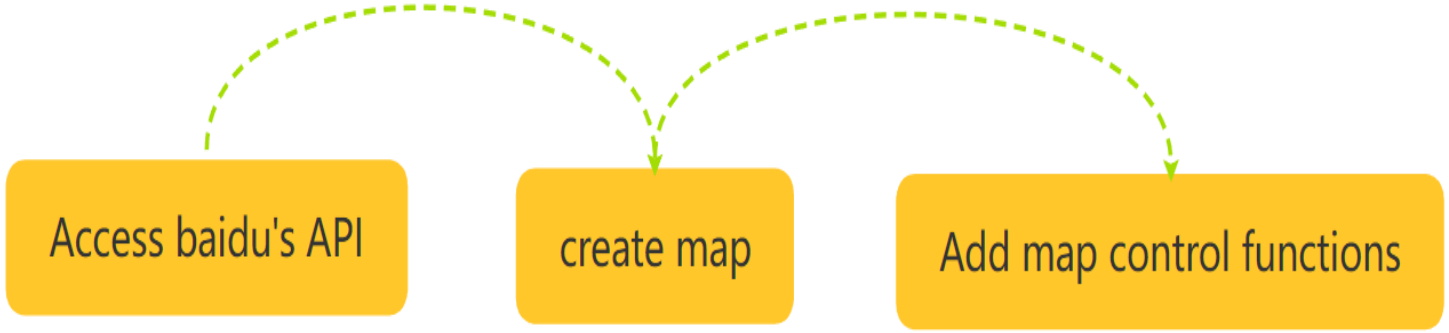
- The basic way ESP8266 connects to HTML is to send information to and from each other using the HTTP network protocol

2.Connect ESP8266 to HTML

```
function getMapX() {                          //Obtain longitude
    var xhttp = new XMLHttpRequest();                          //Creating a Request object
    xhttp.onreadystatechange = function () {          //Sending an HTTP request
      if (this.readyState == 4 && this.status == 200) {//Change the value of x to the value
                                                               //of the ESP8266 response
        document.getElementById("x").innerHTML =
          this.responseText;
          x=this.responseText
      }
    };
    xhttp.open("GET", "readMapLongitude", true);
    xhttp.send();
    return x;
  }

  function getMapY() {                            //Same principle as above
    var xhttp = new XMLHttpRequest();
    xhttp.onreadystatechange = function () {
      if (this.readyState == 4 && this.status == 200) {
        document.getElementById("y").innerHTML =
          this.responseText;
          y=this.responseText
      }
    };
    xhttp.open("GET", "readMapLatitud", true);
    xhttp.send();
    return y;
  }
```

There are two methods on the left, one is to get longitude, the other is to get latitude
They both get latitude and longitude by sending a GET request and receiving the returned data

3.Creating a Map



Here are three steps, first introduce Baidu API, use API to create map, and add map component

```
<meta http-equiv="Content-Type" content="text/html; charset=gb2312" />
    <meta name="keywords" content="百度地图,百度地图API，百度地图自定义工具，百度地图所见即所得工具" />
    <meta name="description" content="百度地图API自定义地图，帮助用户在可视化操作下生成百度地图" />
```

//introduce the Baidu API

3.Creating a Map

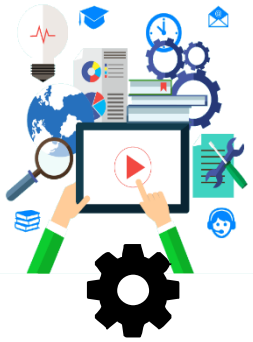
```
//Create and initialize map functions:
    function initMap(x,y) {
        createMap(x,y);//create map
        setMapEvent();//Set the map event
        addMapControl();//Add controls to the map
    }
    //Create map function:
    function createMap(x,y) {
        var map = new BMap.Map("dituContent");//Create a map in baidu Map container
        var point = new BMap.Point(x,y);//Define a central point coordinate
        map.centerAndZoom(point, 13);
//Set the center point and coordinates of the map and display the map in the map container
        window.map = map;//Store the map variable globally
    }

    //Map event setting function:
    function setMapEvent() {
        map.enableScrollWheelZoom();//Enable the map wheel to zoom in and out
    }
```

3.Creating a Map

```
//Map control add function:
function addMapControl() {
    //Adds a zoom control to the map
    var ctrl_nav = new BMap.NavigationControl({ anchor: BMAP_ANCHOR_TOP_LEFT, type:
BMAP_NAVIGATION_CONTROL_LARGE });
    map.addControl(ctrl_nav);
    //Adds a thumbnail control to the map
    var ctrl_ove = new BMap.OverviewMapControl({ anchor: BMAP_ANCHOR_BOTTOM_RIGHT, isOpen: 1 });
    map.addControl(ctrl_ove);
    //Adds a scale control to the map
    var ctrl_sca = new BMap.ScaleControl({ anchor: BMAP_ANCHOR_BOTTOM_LEFT });
    map.addControl(ctrl_sca);
}
```

4. Display location information

- There are four steps, as shown on the left.
- The basic step is to call baidu map API to create a static map.
- Update the map using the longitude and latitude sent by ESP8266 as the center point.



Demo

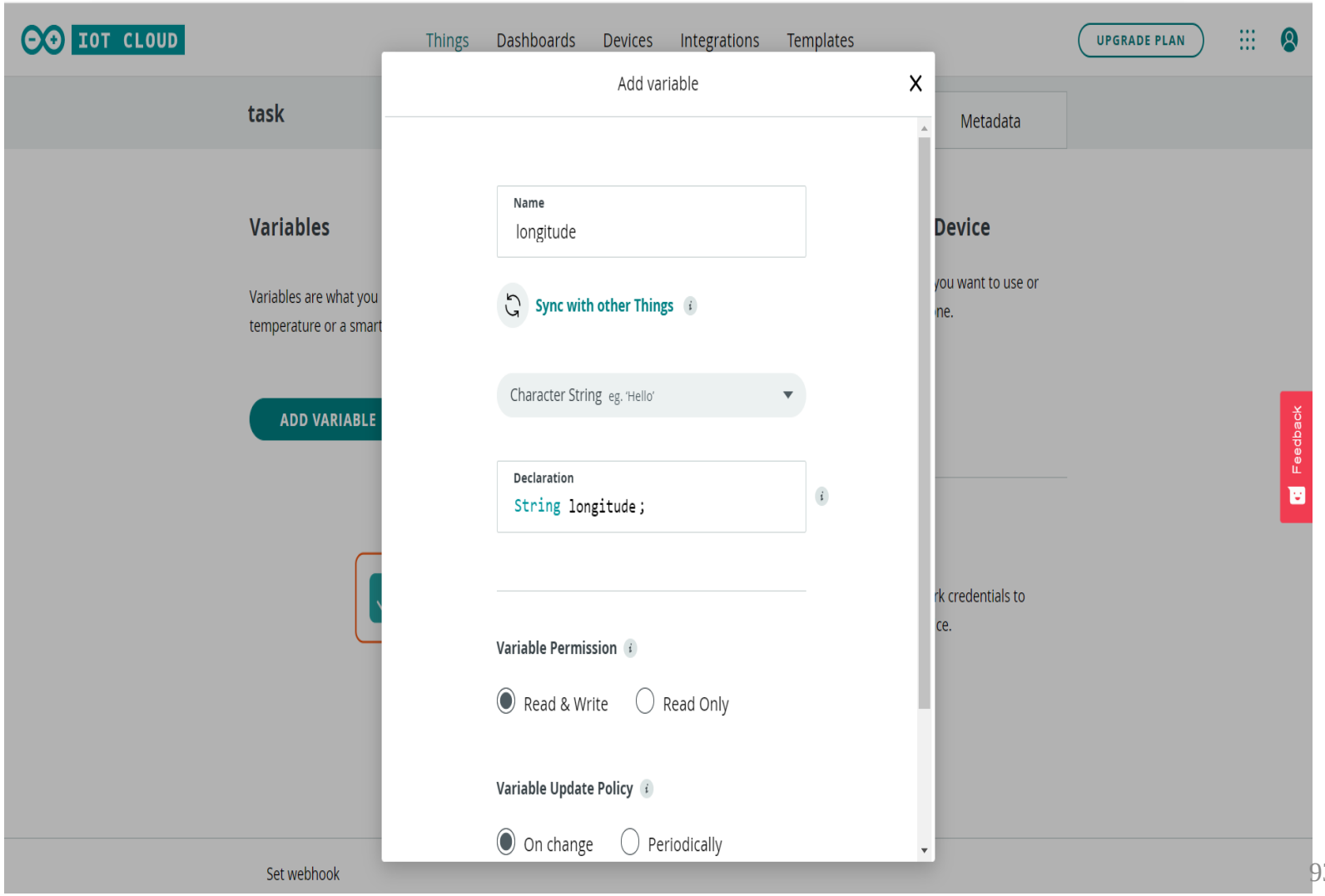

From zhao
00:00:00
结束录制
不安全 | 192.168.43.10
code
影视
学习
下载
学校WIFI和工作
娱乐
英语
tool
电子书
红学
tk
服务器 - 轻量应用...
gin
Matlab2021b哪位...
【软硬结合】Node...
CLASS DESIGN
Get Position Info
欢迎大家观看本组演示
Welcome to this demo

Arduino IOT Cloud

How to create connect to the IOT CLOUD

STEP 1 Create Things

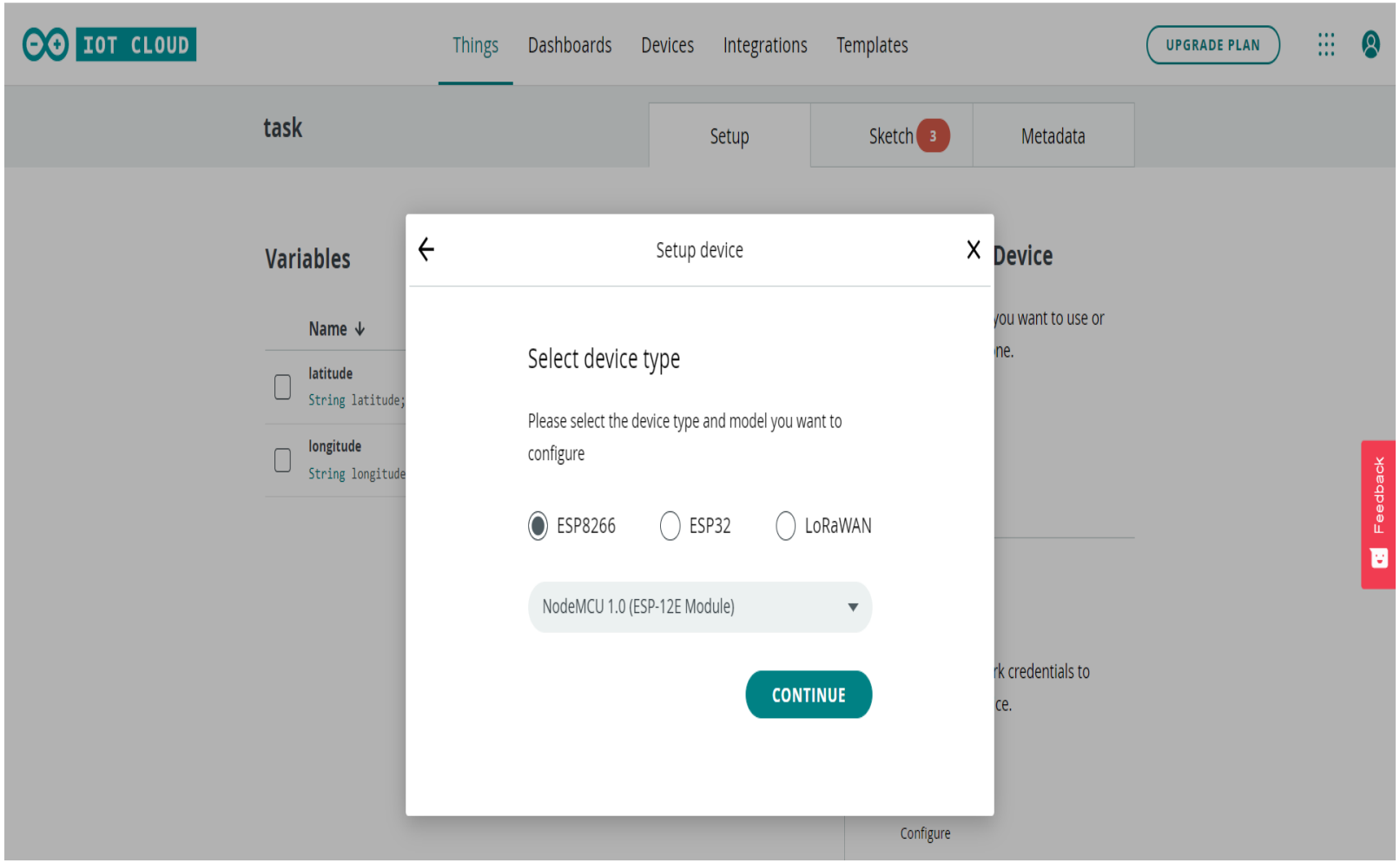
(1)Create Things ,click the Setup.Firstly,we should add two String Variables called longitude and latitude.



Arduino IOT Cloud

How to create connect to the IOT CLOUD

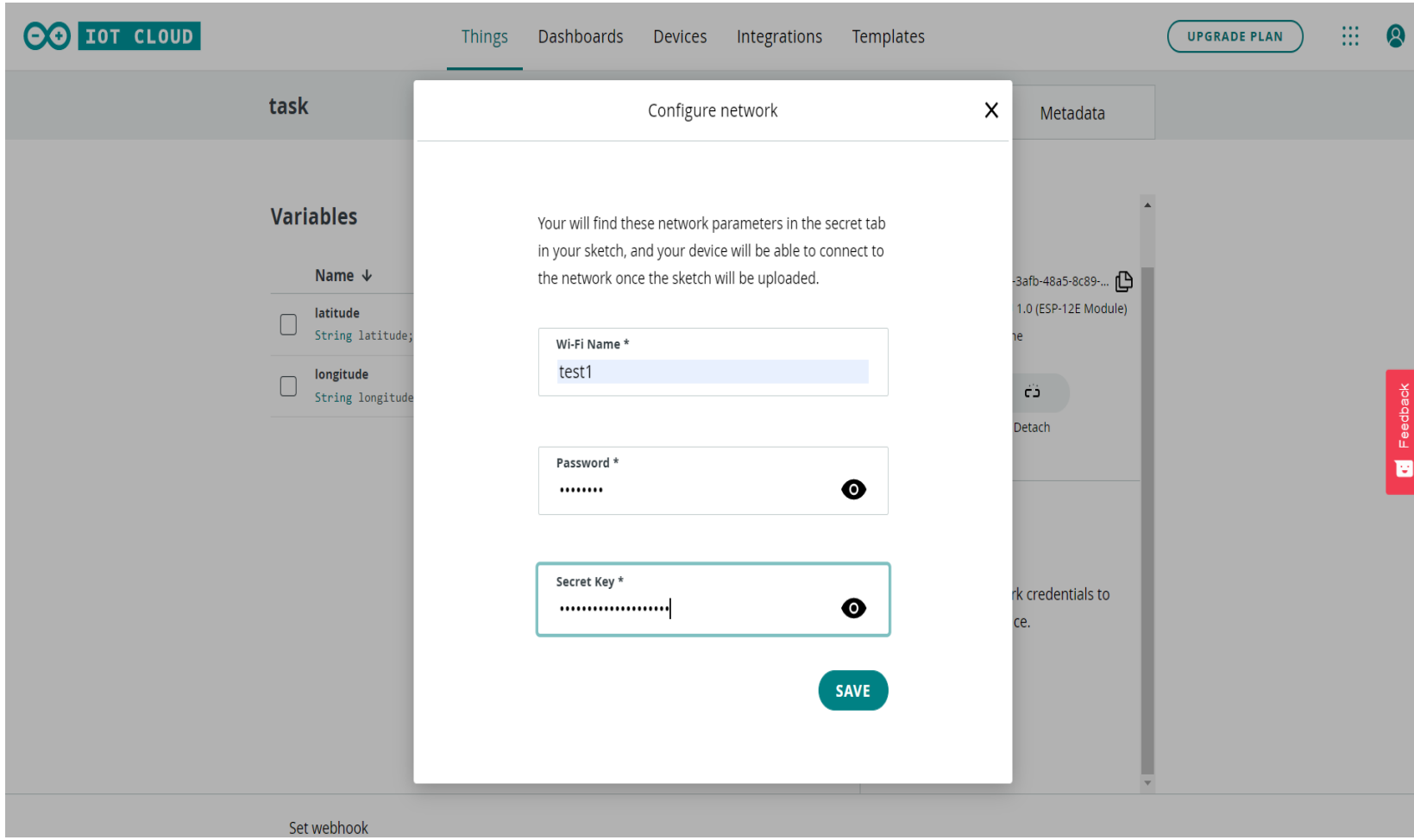
(2)Secondly,setup the device , select the ESP8266 and find the NodeMCU1.0(ESP-12E Module),named it as task,and copy the Secret Key.



Arduino IOT Cloud

How to create connect to the IOT CLOUD

(3)Setup theNetWork, entry the WiFi name, Password and Secret Key.

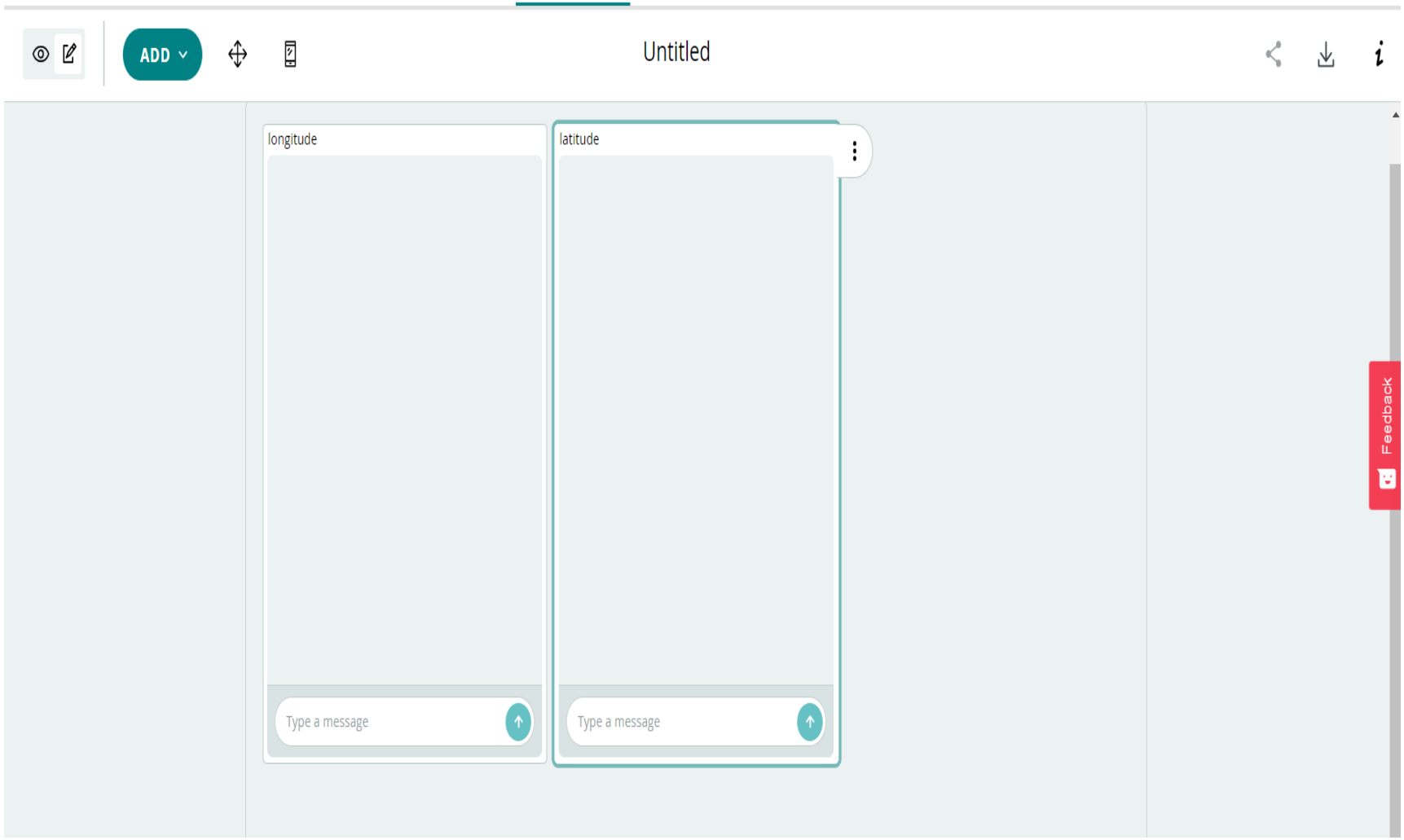


Arduino IOT Cloud

How to create connect to the IOT CLOUD

STEP 2 SetDashboards

Add widget switch , and Linked Variable longitude and latitude to two Message parts.

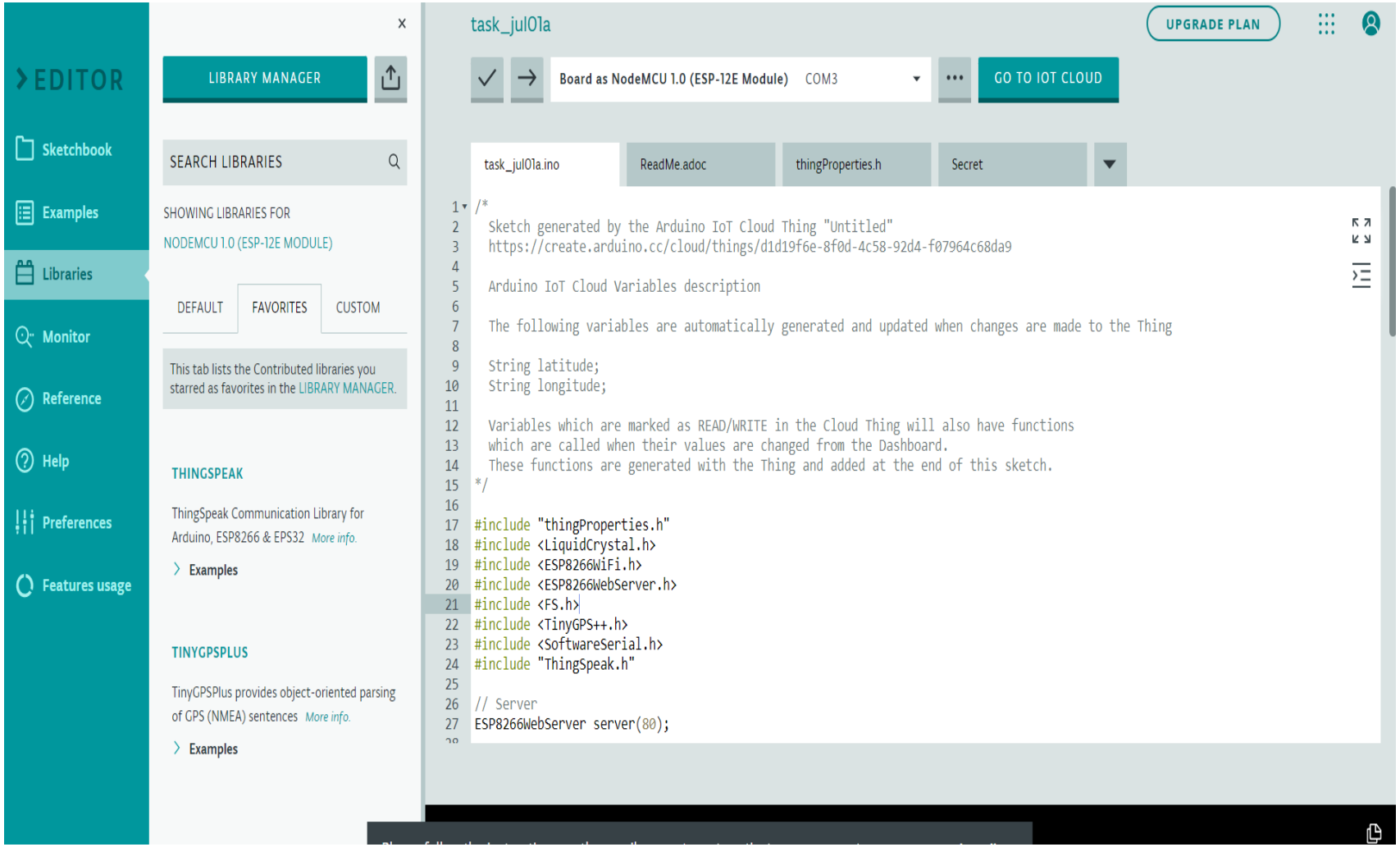


Arduino IOT Cloud

How to create connect to the IOT CLOUD

STEP 3 Sketch

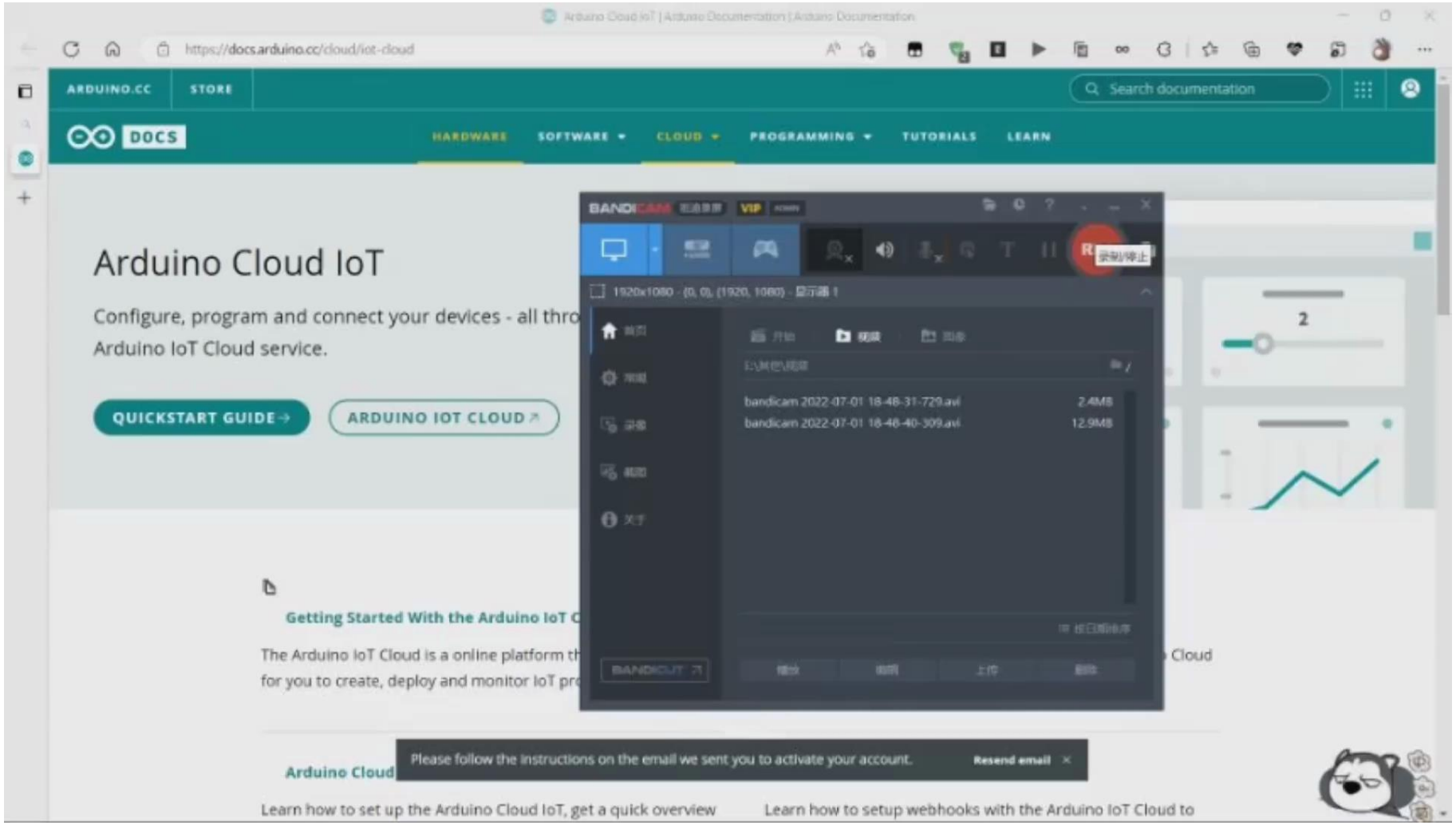
Refine the program , manager the necessary libraries keep the Arduino Agent and select the NodeMCU and Port-com3. Upload the code,then all thing was ready to do the trail in real boards and receive the data we want .



Arduino IOT Cloud

How to create connect to the IOT CLOUD

Show the data



```
#include "thingProperties.h"
#include <TinyGPS++.h>
#include <SoftwareSerial.h>

TinyGPSPlus gps;
SoftwareSerial soft(D5,D6);

 void setup() {
  Serial.begin(9600);
  soft.begin(9600);
  delay(1500);
```

//The header file that the IOT automatically imports

//Soft serial pin definition and use

// Initialize serial and wait for port to open:

// This delay gives the chance to wait for a
// Serial Monitor without blocking if none is found

Project 2: Code

```
  initProperties();                                   // Defined in thingProperties.h

                                                      // Connect to Arduino IoT Cloud
  ArduinoCloud.begin(ArduinoIoTPreferredConnection);

  setDebugMessageLevel(2);
  ArduinoCloud.printDebugInfo();
}

void loop() {
  ArduinoCloud.update();
                                                      // Arduino Cloud update
```

Project 2: Code

```
while (soft.available() > 0)
  {
    if (gps.encode(soft.read()))
    {
      if (gps.location.isValid())
      {
        double latitude = gps.location.lat();
        double longitude = gps.location.lng();

        latitude_data = String(latitude, 6);
        longitude_data = String(longitude, 6);
```

//Judgment criteria

//encode GPS data

//Whether the location is
// legal or not

//Receive double type data

//double data to string

Project 2: Code

```
Serial.print("纬度：  ");
                Serial.print(latitude_data);
                Serial.print("    ");
                Serial.print("经度：  ");
                Serial.println(longitude_data);   //  Serial port print data
                delay(1000);
            }
        }
    }

}
```

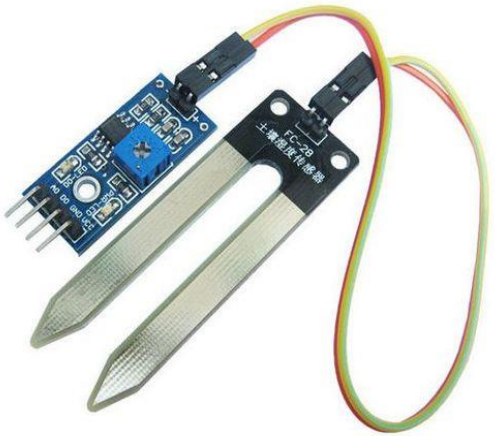
Project3:

IoT based Irrigation System

Project 3: Aim & Study

- Creat our own webservers to log the data with real-time graphs.
- Here we will also create ESP8266 Web Server Data Logger using sht10 Sensor.
- It Can be used to monitor the temperature and humidity of an area.
- So,it can be applicated by the Food Industry、Archives Management Office、Greenhouse、Tobacco Industry,etc.

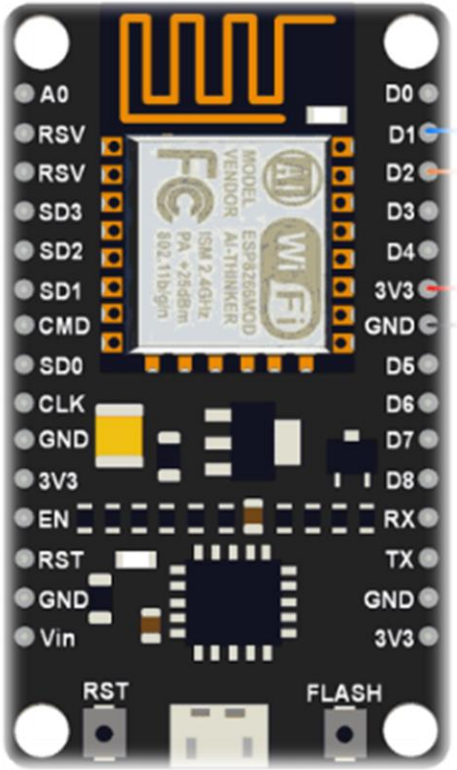
Project 3:Component



Soil Moisture Detection Humidity Sensor

SHT10 module

Node MCU ESP8266-12e

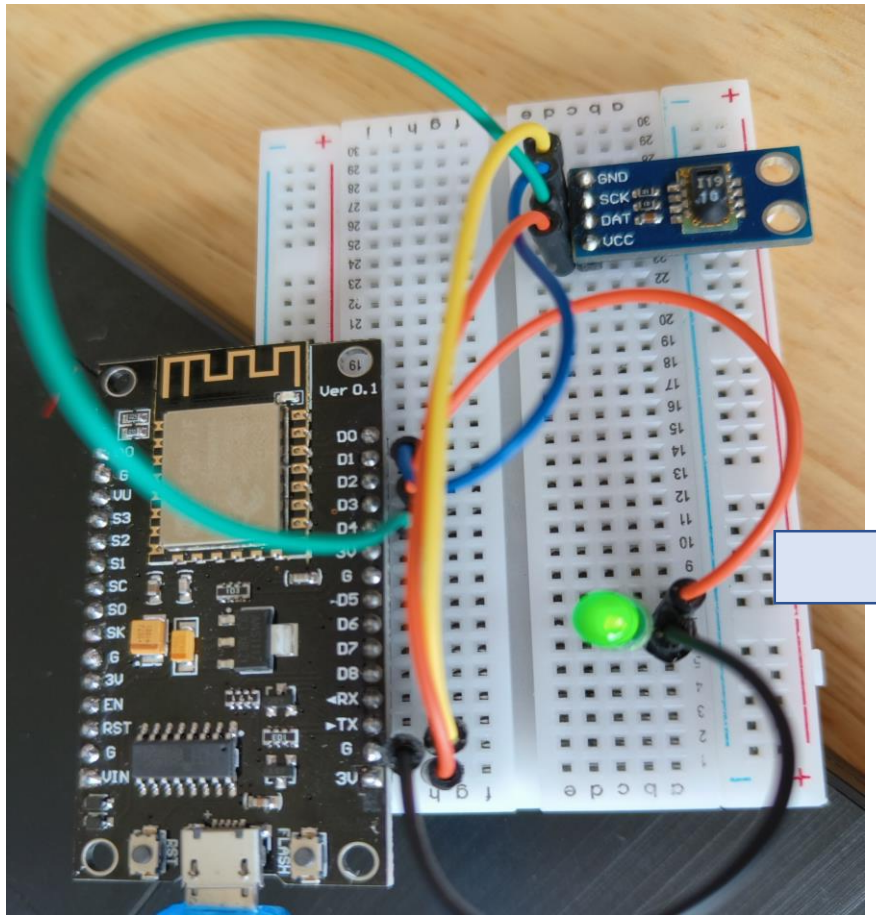


YFS201 Water Flow Sensor

Jumper Wires
(Female to Female)

Project 3:Component

- This system consists of detection object, temperature and humidity sensor, NodeMCU, web terminal,Bread Board,etc.



VCC connects to 3v
GND connects to G
SCK connects to D2
DAT connects to D4
LED connects to D3 and G

Project 3:Component

1、NodeMCU

As a WIFI network communication module, ESP8266 can realize the network transmission of data, and has the characteristics of stable operation and low price. ESP8266 supports three working modes: AP/STA/AP+STA. Select the STA working mode here, and connect ESP8266 as a station to the WiFi network established by the access point.

2、SHT10 Sensor

The power supply voltage range of SHT10 is 2.4v-5.5v, and the recommended power supply voltage is 3.3v. The serial interface of SHT1x has been optimized in terms of sensor signal reading and power consumption;

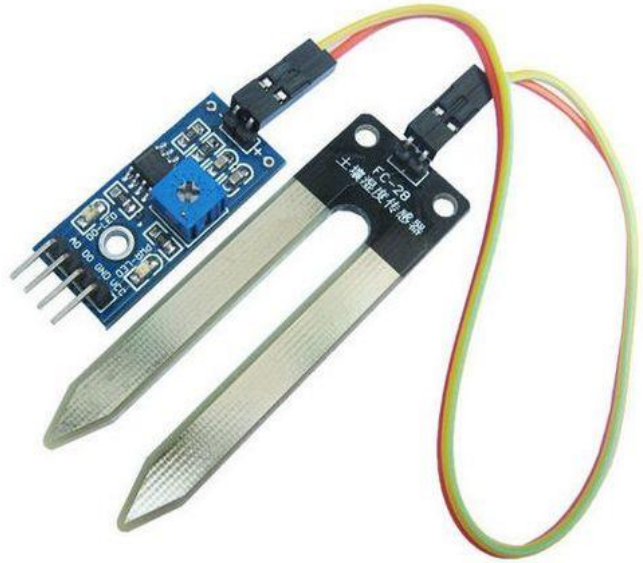
PIN Of the SHT10：

SCK：SCK is used for communication synchronization between the microcontroller and SHT10.

DATA：The DATA pin is a tri-state structure for reading sensor data. When sending a command to the sensor, DATA is valid on the rising edge of SCK and must remain stable when SCK is high, and DATA changes after the falling edge of SCK.

Description

- Soil sensor: the moisture sensor can be used to detect the moisture of the soil, when the soil is dry , the module outputs a high level , whereas the output low. Using this sensor make an automatic watering system , so that your garden plants need people to manage.
  - Adjustable sensitivity (shown in blue digital potentiometer adjustment).
  - Operating Voltage 3.3V-5V.
  - Module Dual Output mode , a simple digital output , analog output more accurate.
  - With fixed bolt hole for easy installation.
  - Small PCB board size: 3cm * 1.6cm.
  - Power indicator (red) and the digital switch output indicator (green).
  - Using LM393 comparator chip, stable.
  - VCC external 3.3V-5V
  - GND GND External
  - DO small board digital output interfaces (0 and 1)
  - AO small board analog output interface

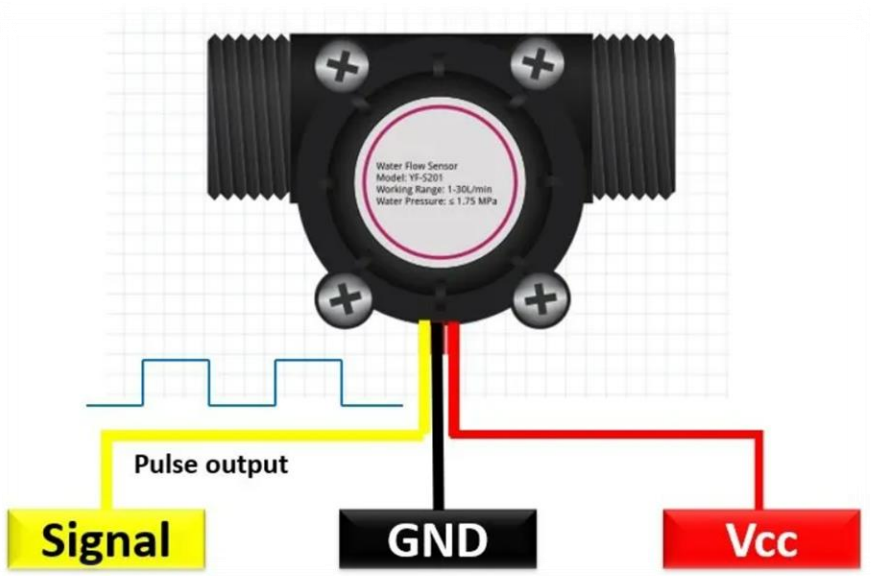


•A soil moisture and humidity module most sensitive to the environment , usually used to detect soil moisture.
•Modules in the soil moisture reach the set threshold , DO port output high when soil moisture exceeds a set threshold, the module D0 output low.
•Digital outputs D0 small plates can be directly connected with the microcontroller through the microcontroller to detect high and low , thereby detecting soil moisture.
•Small plates digital output DO OUR relay module can directly drive the buzzer module , which can be composed of a soil moisture alarm equipment.
•Small plates AO analog output modules can be connected and AD through the AD converter , you can get a more accurate value of soil moisture.

Project 3:Component

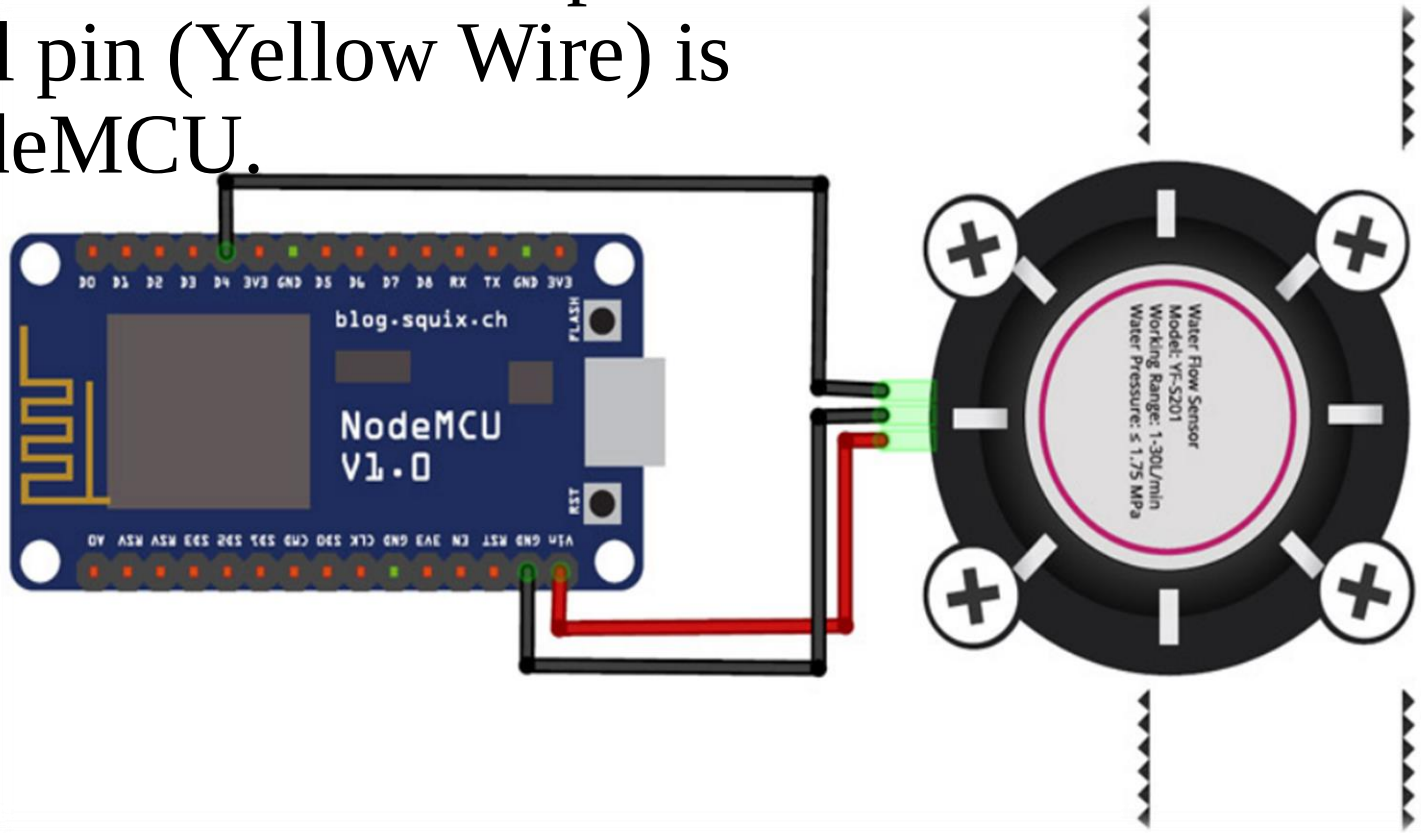
- **Water Flow Sensor:**

Water flow sensors are installed at the water source or pipes to measure the rate of flow of water and calculate the amount of water flowed through the pipe. Rate of flow of water is measured as liters per hour or cubic meters.

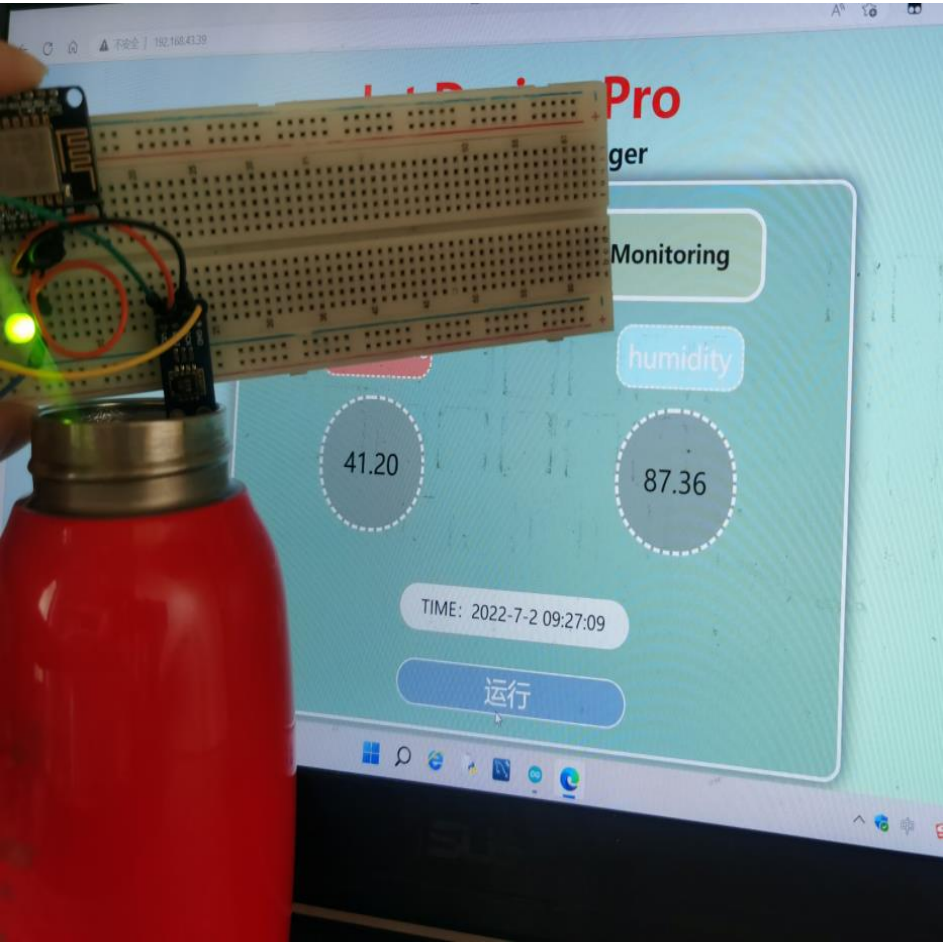


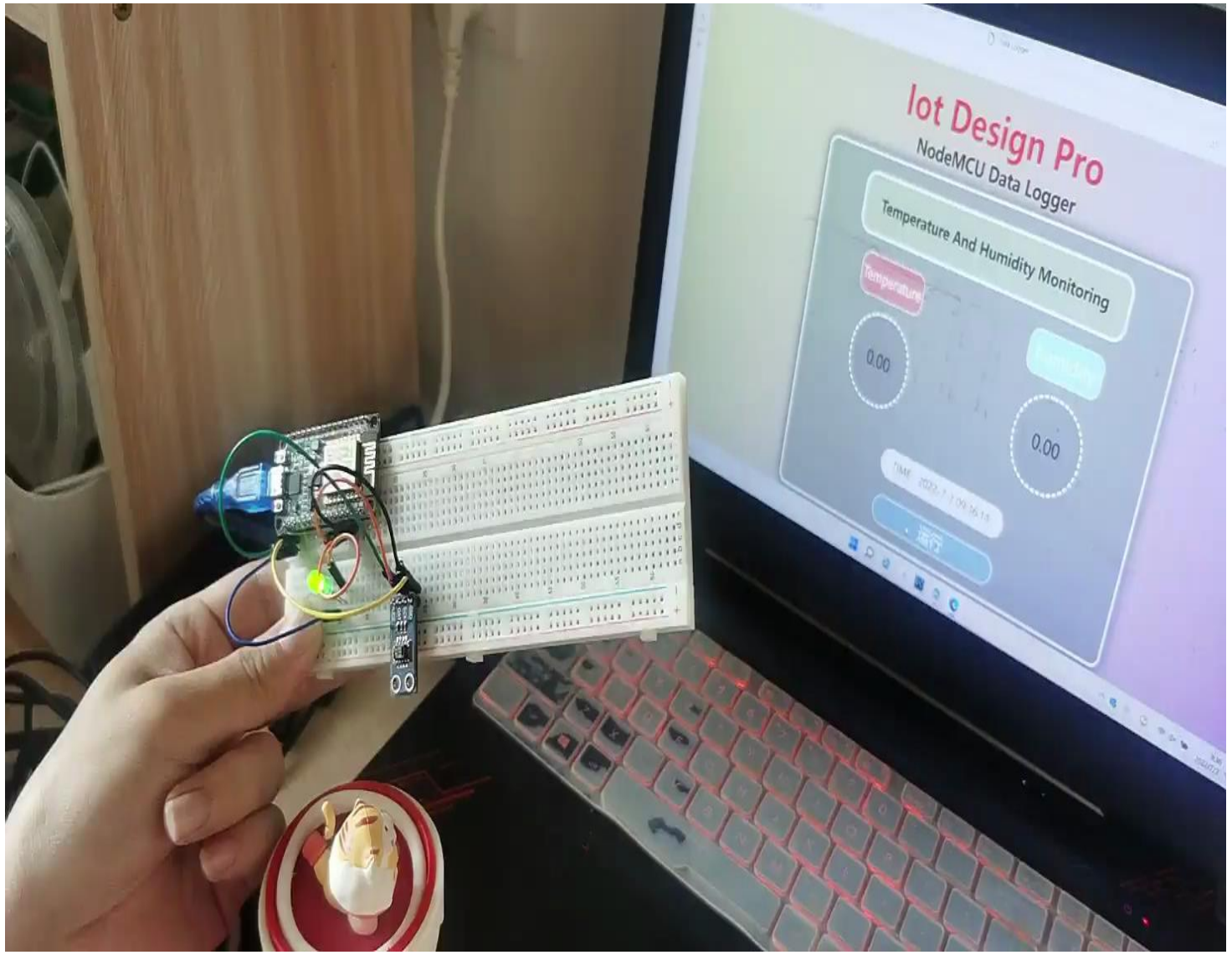
Project 3:Component

- The VCC (Red Wire) and GND (Black Wire) pins of flow sensors are connected to 5V and GND pins of NodeMCU while the Signal pin (Yellow Wire) is connected to D4 pin of NodeMCU.

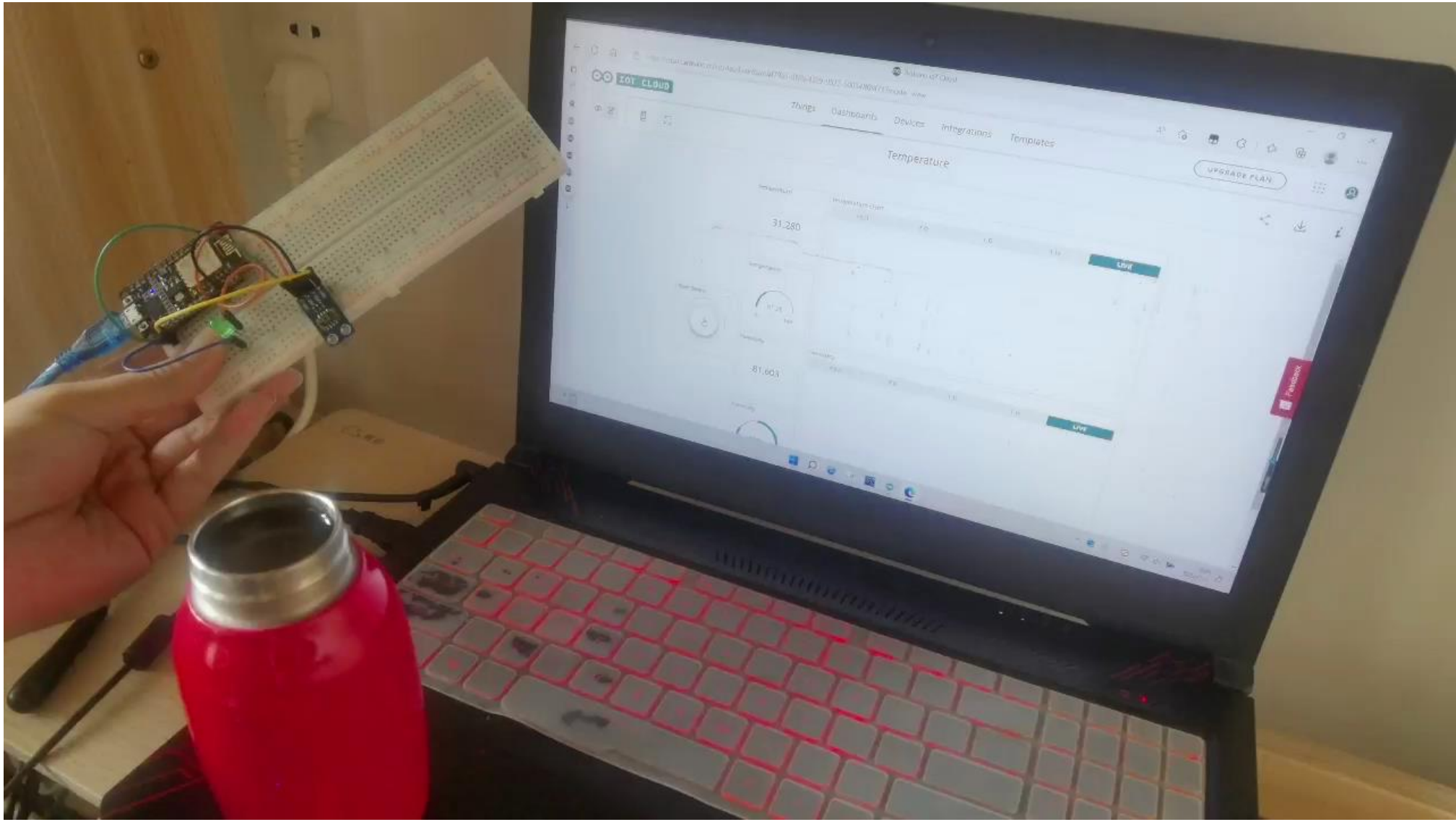


Project 3: DEMO onWeb





Project 3: Demo Arduino IOT Cloud

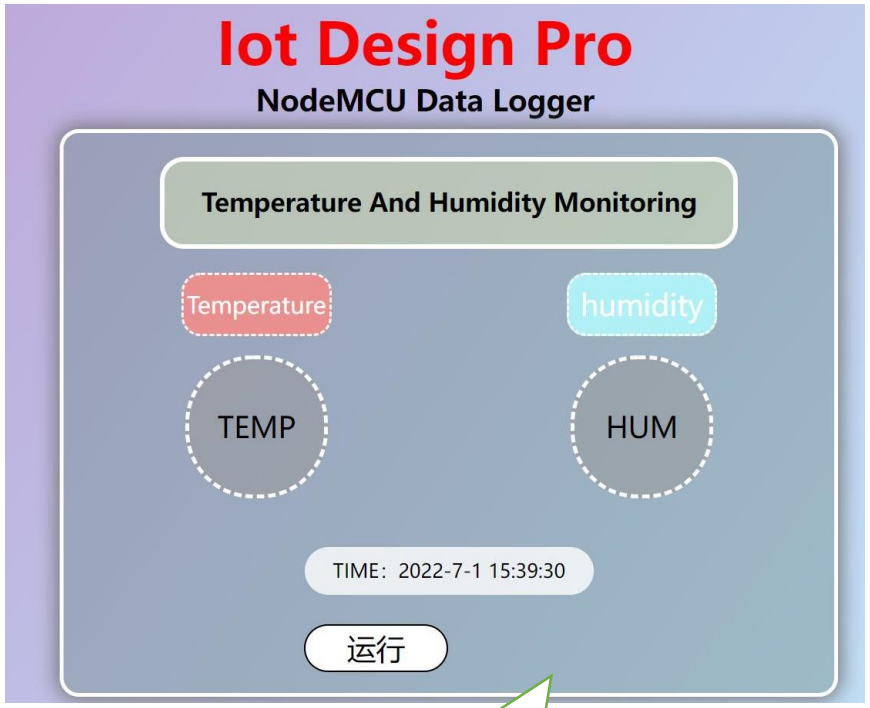


Project 3:

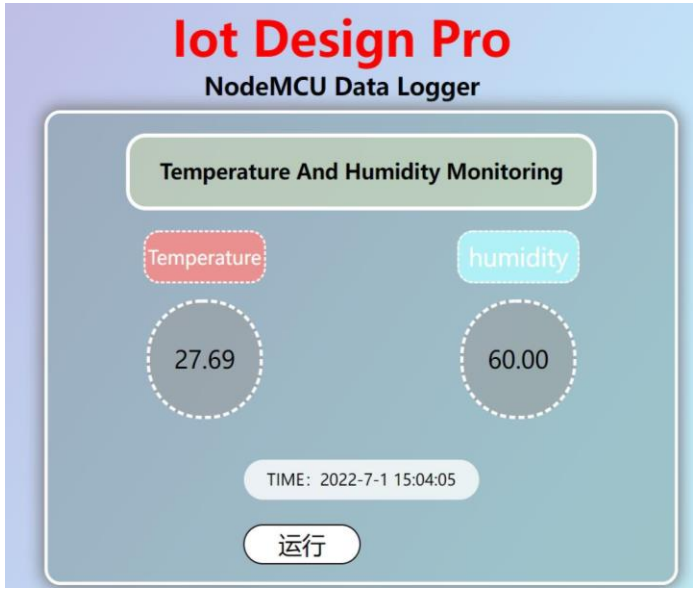

question
answer
question
question
question
question
question
question


QUESTIONS
ANSWERS

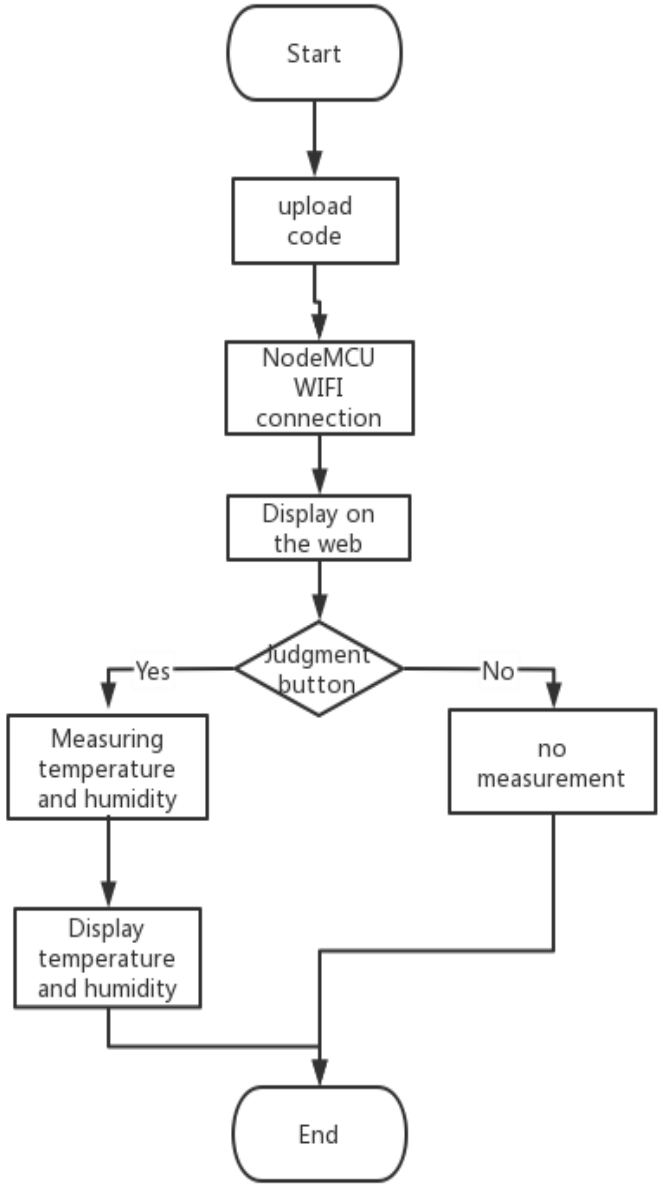
Flow Chart



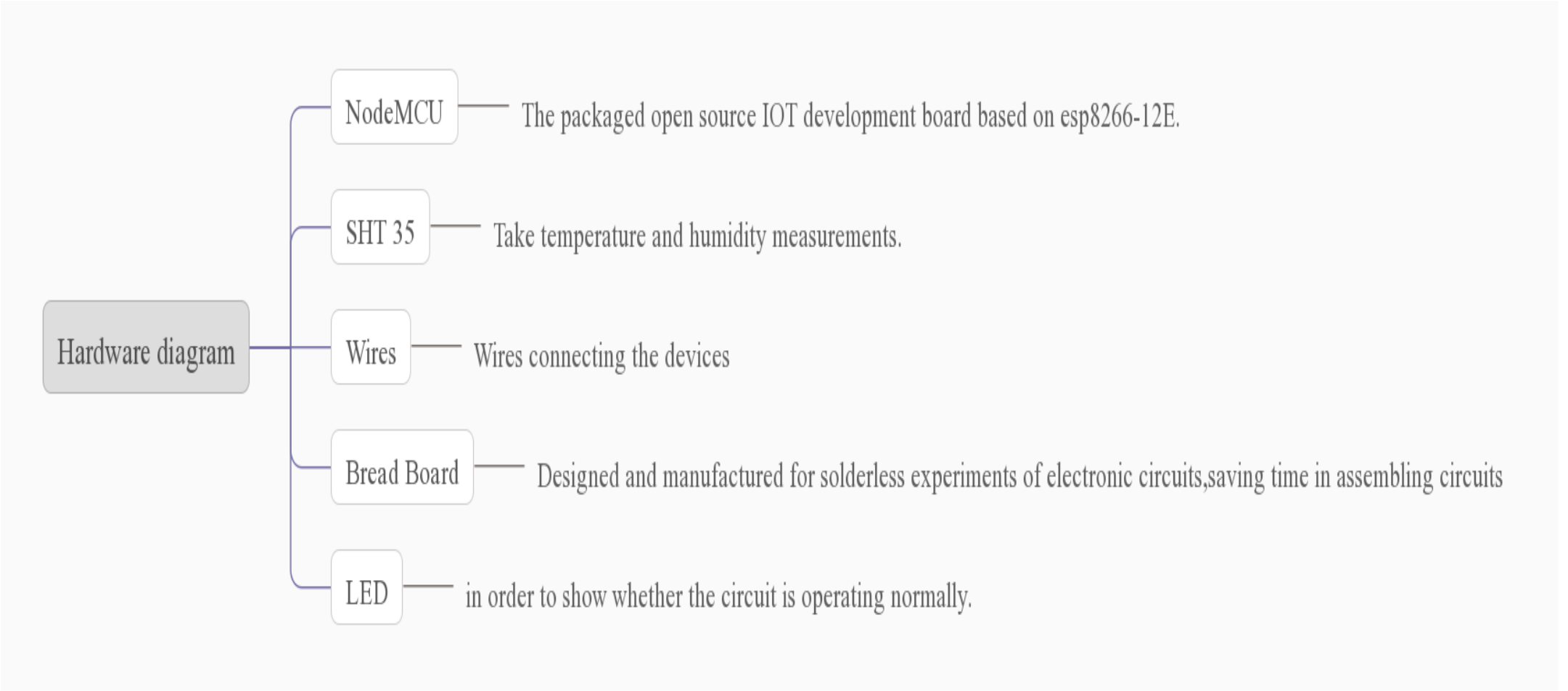
Wrong：No value



normal operation

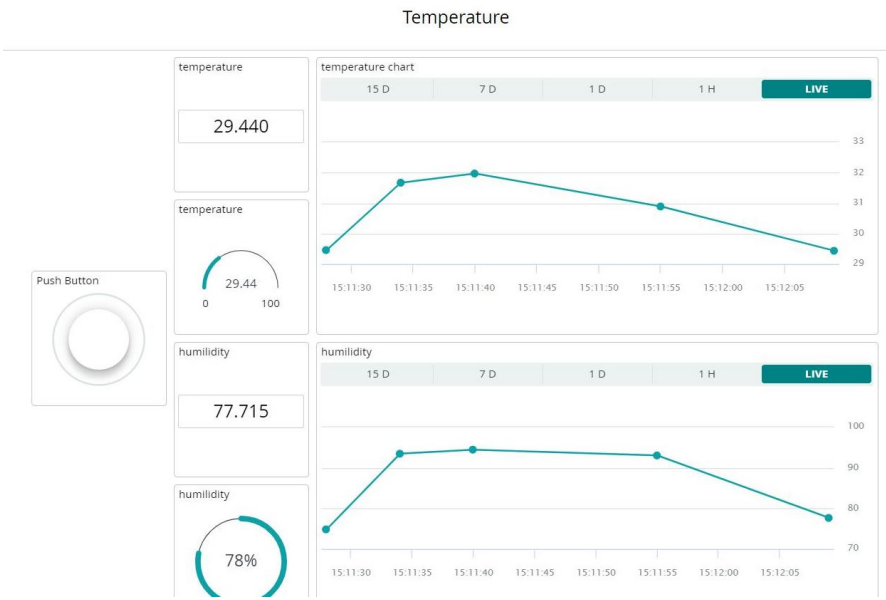


hardware diagram

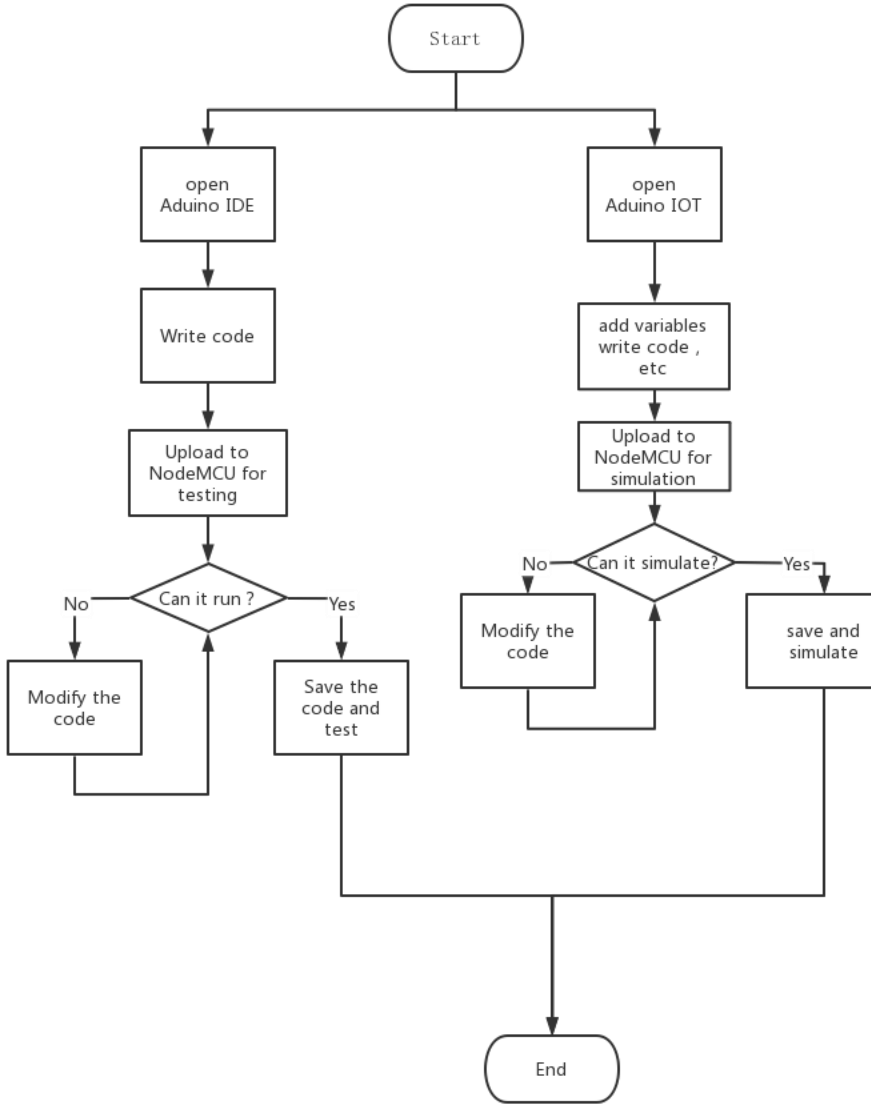


Software Flow Chart

IDE page



IOT page



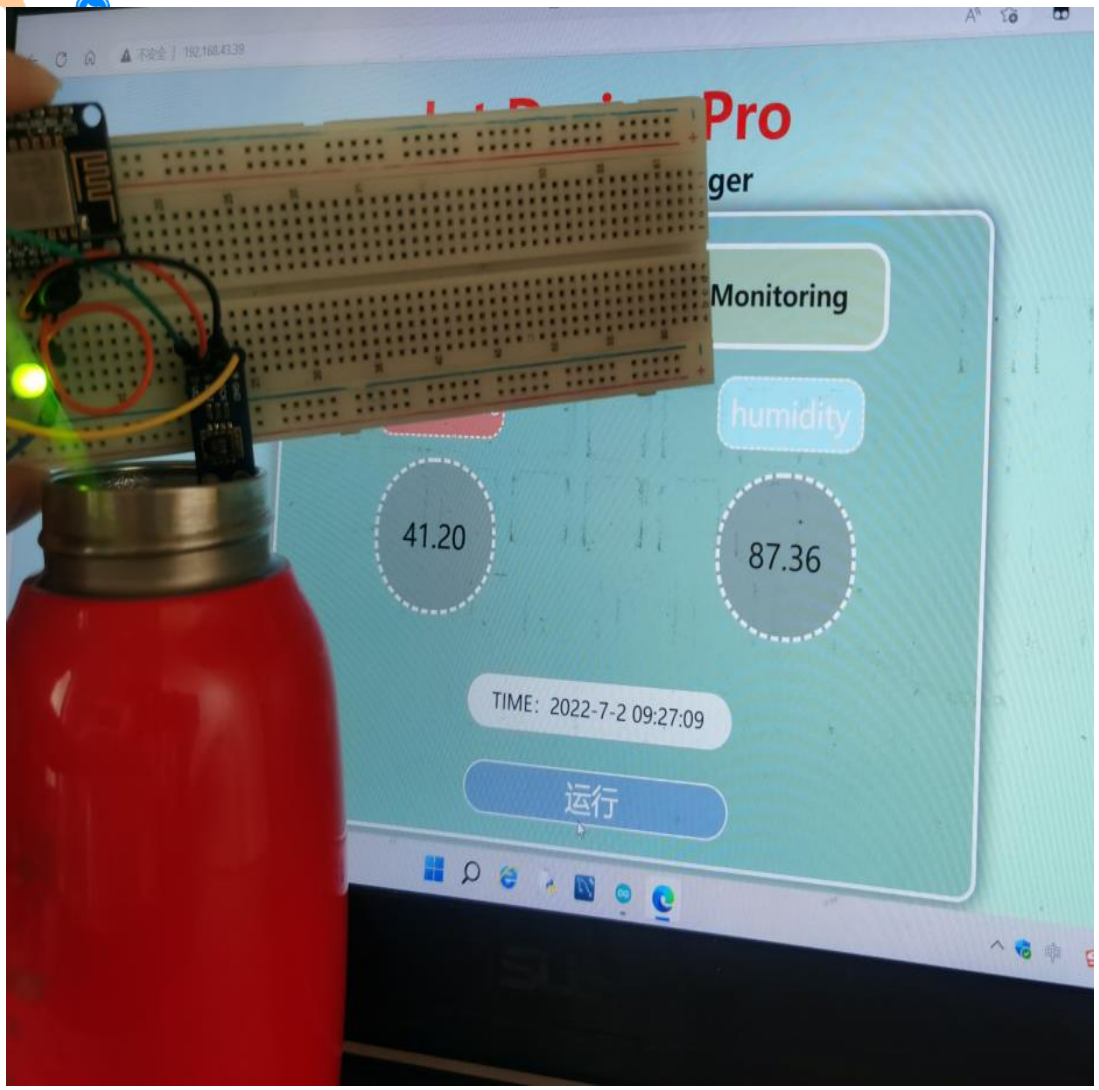
Software diagram

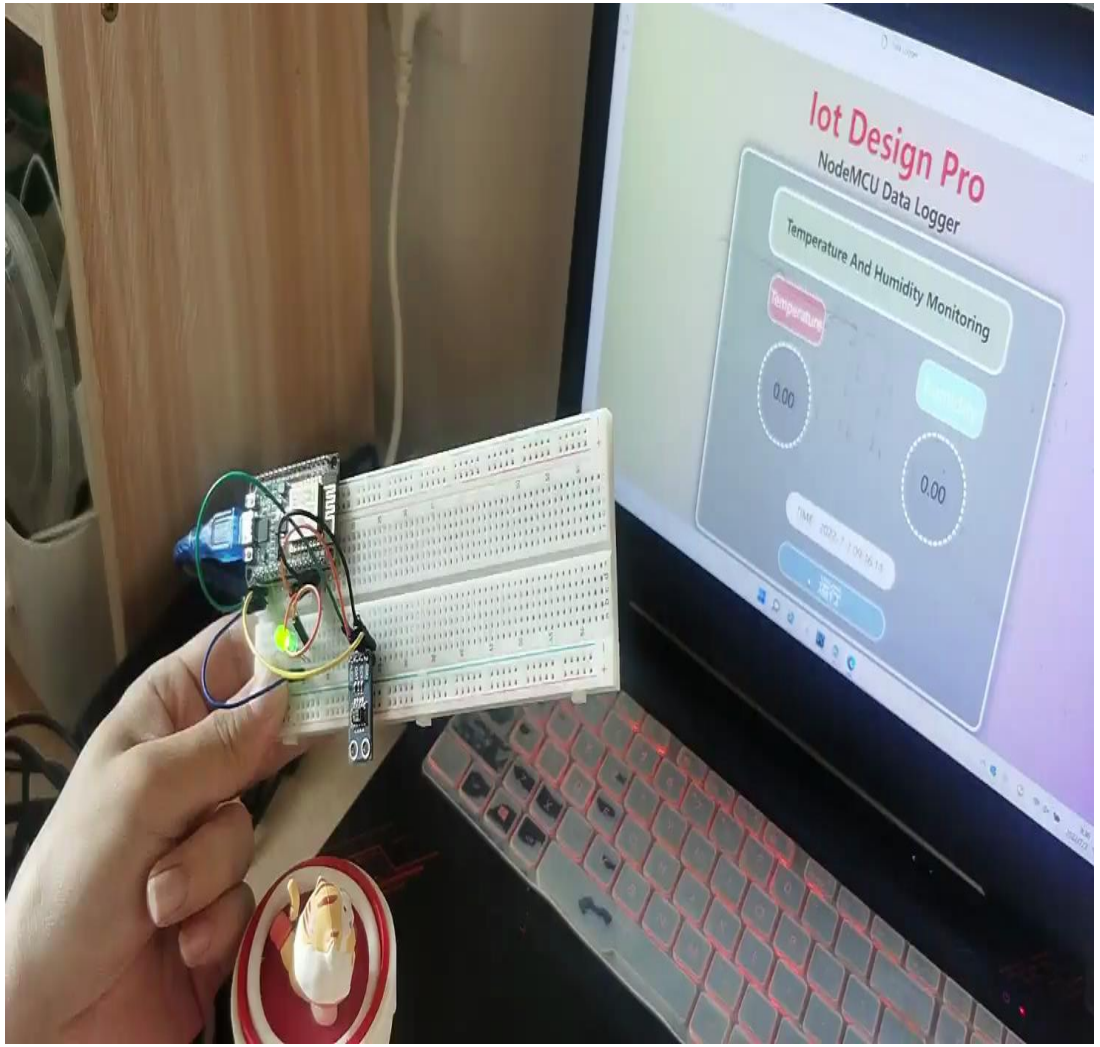
- Software diagram
  - Aduino IOT
    - Function：Helps manufacturers build and monitor connected objects in a fast,easy and secure way.
    - Set variables: temperture、humidity、switch
    - Add Device:NodeMCU 1.0(ESP-12E Moudle)
    - Associated Device & Set Network
    - Write Code and Modify bug
    - Upload to the NodeMCU
    - simulate it
  - Aduino IDE
    - Function：Upload the program from the code to the Arduino board.
    - Write project code.
    - Save and compile，modify bug.
    - Connect nodemcu development board，and select board type and port.
    - Upload to nodemcu
    - View serial port monitor
    - Get sensor data and website

mind Mapping

- NodeMCU Data Logger to Upload Data on Webserver
  - WebServer
    - HTML
      - Title:Indoor Temperature and Humidity Monitoring System.
      - Display of temperature and humidity.
      - button:run
    - CSS
      - Change the background color
      - Modify label style
    - JAVASCRIPT
      - Process the received data
    - AJAX
      - Send data to the web
  - ESP8266
    - programming
      - Initialize,define variables, ssid, password
      - Read temperature and humidity
      - Monitor for client access
        - Simulate client requests and generate control instructions
        - Send response information to client

Web Clip


Pro
ger
Monitoring
humidity
41.20
87.36
TIME: 2022-7-2 09:27:09
运行


Iot Design Pro
NodeMCU Data Logger
Temperature And Humidity Monitoring
0.00
0.00

Project3:

Code Section

The IDE’s Code and the Web’s Code

```
#include <ESP8266WiFi.h>

#include <WiFiClient.h>

#include <ESP8266WebServer.h>

#include <SHT1x.h>

//SSID and Password of your WiFi router

const char* ssid = "新晋小色批";

const char* password = "103168Wan$&#*.";

#define LED 0

#define dataPin  2

#define clockPin 4

WiFiServer server(80);

SHT1x sht1x(dataPin, clockPin);

float temp_c;

float humidity;
```

//throught the WiFi connect the ESP8266

//choose the pin D4 and D2 to the receive the temprature and humilidity

```
ESP8266WebServer webserver(80); //Server on port 80

const char MAIN_page[] PROGMEM = R"=====("(

<!doctype html>

<html>

<meta charset="UTF-8">

<head>

    <meta>
    <style>
        canvas {
            -moz-user-select: none;
            -webkit-user-select: none;
            -ms-user-select: none;
        }
        /* Data Table Styling*/
        * {
            margin: 0;
            padding: 0;
        }
        body {
            width: 100%;
            background-color: rgb(232, 232, 168);
            margin: 0;
            padding: 0;
            font-family: "montserrat";
            background-image: linear-gradient(125deg, rgb(232, 232, 168), rgb(188, 152,
210), #c2e9fb, #C2FFD8);
            background-size: 400%;
            animation: bganimation 15s infinite;
        }
```

//These the Web Page’s Code,including “table style”、“table color”、”background-size” and so on

The Web Page's Code

```
.box {
        width: 800px;
        height: 580px;
        background-color: rgba(102, 134, 129, 0.4);
        border-radius: 20px;
        margin-left: 365px;
        position: relative;
        border: 4px solid #fff;
        box-shadow: 0px 0px 20px 3px #747474;
    }
    @keyframes bganimation {
        0% {
            background-position: 0% 50%;
        }
        50% {
            background-position: 100% 50%;
        }
        100% {
            background-position: 0% 50%;
        }
    }
.box .header {
        width: 600px;
        margin: 0px auto;
        padding: 25px;
    }
    .header h1 {
        text-align: center;
        border-radius: 25px;
        background-color: rgb(215, 226, 179, .5);
        line-height: 85px;
        border: 5px solid #fff;
        font-size: 26px;
    }
    .center ul {
        display: flex;
        justify-content: space-around;
    list-style: none;
    }
```

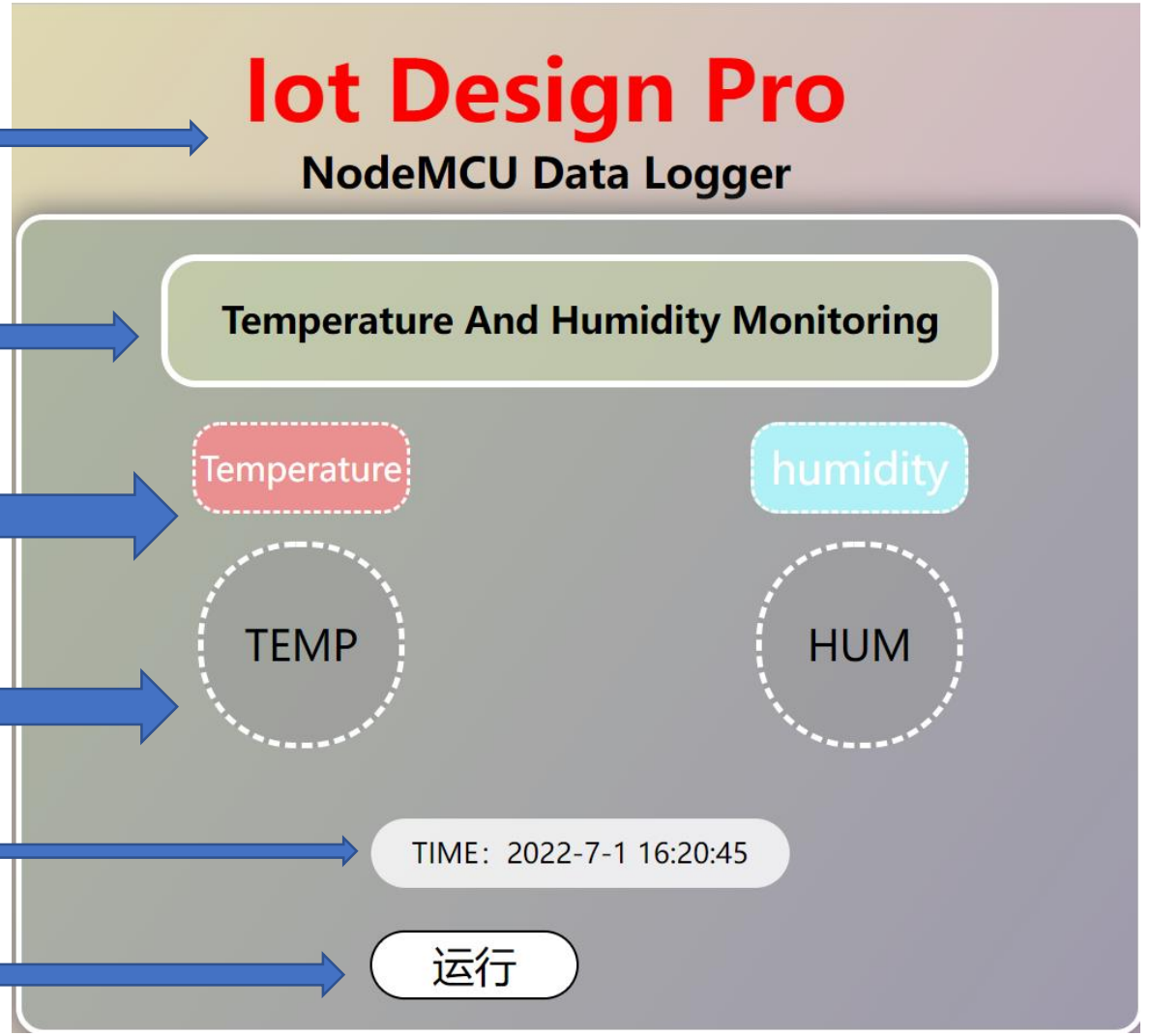
```
        .center ul li {
        width: 140px;
        height: 140px;
        text-align: center;
        line-height: 140px;
        border-radius: 70px;
        font-size: 30px;
        border: 3px dashed #fff;
        background-color: rgb(151, 151, 151, .5);
    }
    .center .title li {
        background-color: rgba(30, 127, 111, 0.4);
        color: #fff;
        border-radius: 20px;
        width: 150px;
        height: 60px;
        line-height: 60px;
        margin-bottom: 20px;
    }
    .center .title #w {
        background-color: #ea9090;
        font-size: 24px;
    }
    .center .title #s {
        background-color: #aff1f6;
    }
    .center #wd {
        border: 4px dashed #fff;
    }
    .center #sd {
        border: 4px dashed #fff;
    }
.times {
        width: 300px;
        height: 50px;
        text-align: center;
        line-height: 50px;
        border: 2px rgb(0, 0, 0);
        border-radius: 50px;
```

```
        font-size: 20px;
        padding: 0 auto;
        margin: 0px auto;
        margin-top: 50px;
        background-color: rgb(255, 255, 255, .8);
    }
    .foot {
        font-size: 20px;
        width: 300px;
        height: 70px;
        margin: 80px auto;
        text-align: center;
        line-height: 70px;
        padding: 15px;
    }
.but {
        width: 150px;
        height: 50px;
        font-size: 30px;
        text-align: center;
        line-height: 50px;
        border-radius: 50px;
        padding: 0 auto;
        margin: 0px auto;
        margin-top: 30px;
        margin-left: 249px;
        background-color: #fff;
        transition: all .3s;
    }
    .but:hover {
        width: 300px;
        background-color: rgb(139, 182, 204);
        border: 2px solid #fff;
        color: #fff;
    }
    .but:active {
        background-color: rgb(86, 85, 85);
        color: #FFF;
    }
  </style>
</head>
```

//These Code also about the Web Page,including the "box"、"box.header"、"center.title"、"button's style"、"color"、"height"、"width"、" background-color" and so on

The Web Page's Code

```
<body>
<title>Data Logger</title>
  <h1 style="text-align:center; color:red;font-size: 60px;">Iot Design Pro</h1>
  <h3 style="text-align:center;font-size: 30px;margin-bottom: 10px;">NodeMCU Data Logger</h3>
<div>
  <div class="box">
    <div class="header">
      <h1>Temperature And Humidity Monitoring</h1>
    </div>
    <div class="center">
      <ul class="title">
        <li id="w">Temperature</li>
        <li id="s">humidity</li>
      </ul>
    <ul>
      <li id="wd">TEMP</li>
      <li id="sd">HUM</li>
    </ul>
  </div>
  <div class="times"></div>
  <button onclick="getData()" class="but">运行</button>
```



Comment:The data

```
var Tvalues = [];
var Hvalues = [];
var timeStamp = [];
unction getData() {
   var xhttp = new XMLHttpRequest();
   xhttp.onreadystatechange = function () {
   if (this.readyState == 4 && this.status == 200) {
      //Push the data in array
      var time = new Date().toLocaleTimeString();
      var txt = this.responseText;
      var obj = JSON.parse(txt);
      Tvalues.push(obj.Temperature);
      Hvalues.push(obj.Humidity);
      timeStamp.push(time);
//Update Data Table
         var wd = document.getElementById("wd");
         var sd = document.getElementById("sd");
         wd.innerHTML = obj.Temperature;
         sd.innerHTML = obj.Humidity;
            }
         };
      xhttp.open("GET", "readData", true); //Handle readData
server on ESP8266
      xhttp.send();
 }
</script>
```

//Line 1 to 14
The part Code functions is to “Push the data in array” ,the data including “temperature”、”humilidity”、”time at now”

//Line 15 to 24
The part Code functions is to “Update Data Web Page’s Table,through the “readData” to receive the new data on ESP8266 and send the data to web

Comment:The data

```
const times = document.querySelector('.times');
function getDate() {
            const date = new Date()
            let h = date.getHours()
            let m = date.getMinutes()
            let s = date.getSeconds()
            h = h < 10 ? '0' + h : h
            m = m < 10 ? '0' + m : m
            s = s < 10 ? '0' + s : s
            times.innerHTML = `TIME：
${date.getFullYear()}-${date.getMonth() + 1}-
${date.getDate()}  ${h}:${m}:${s}`
        }
        getDate()
        setInterval(getDate, 1000)
    </script>
</body>
</html>
)=====";
```

//The part Code function is to by retrieving the current time on the computer,the current time is transmitted to the web page through the server

At this point,the web page’s code ends

Comment:The data

```
void handleRoot() {
 String s = MAIN_page; //Read HTML contents
 webserver.send(200, "text/html", s); //Send web page
}
void readData() {
 String data = "{\"Temperature\":\""+ String(temp_c) +"\",
\"Humidity\":\""+ String(humidity) +"\"}";
 digitalWrite(LED,!digitalRead(LED)); //Toggle LED on
data request ajax
 webserver.send(200, "text/plane", data);
 //Send ADC value, temperature and humidity JSON to
client ajax request
 delay(2000);
 temp_c = sht1x.readTemperatureC();
 humidity = sht1x.readHumidity();
 Serial.println(temp_c, 1);
 Serial.println(humidity, 1);
}
```

//Line 1 to 4
The part code's functions is to save the code we wrote above for the html in an array and upload it to the web page

//Line 5 to end
The part code's functions is to through the "sht1x" library to receive the "tempature's value" and " the humilidity's value",send JSON to client ajax request

Comment:The Wifi and IP

```
void setup ()
{
  Serial.begin(115200);
  Serial.println();
  pinMode(dataPin, INPUT);
  pinMode(clockPin, INPUT);
   WiFi.begin(ssid, password);
  Serial.println("");
   pinMode(LED,OUTPUT);
  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());
  webserver.on("/", handleRoot);
  webserver.on("/readData", readData);
  webserver.begin();
  Serial.println("HTTP server started");
}
void loop(void){
  webserver.handleClient();
  }
```

//Line 1 to 7
Connect to the WiFi router

//Wait for connection

//Line 10 to13
If connection successful show IP address in serial monitor

//Line 14 to 18
IP address assigned to your ESP

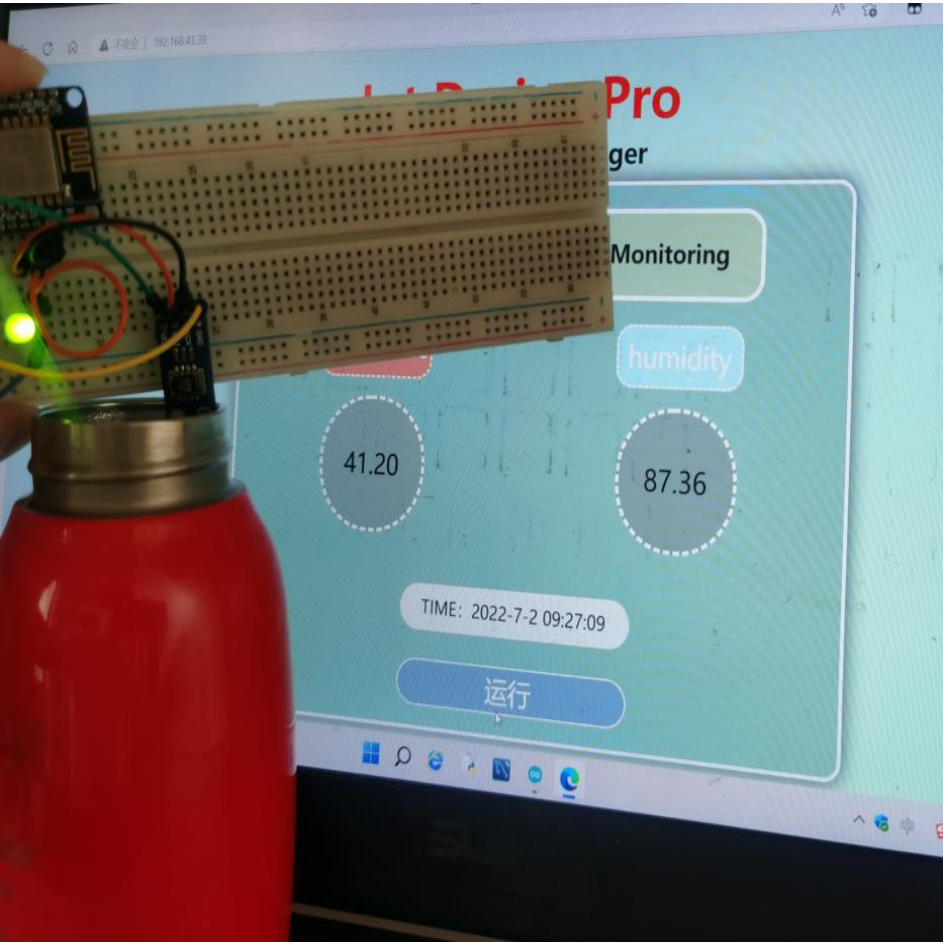
//Line 19
Which routine to handle at root location.This is display page

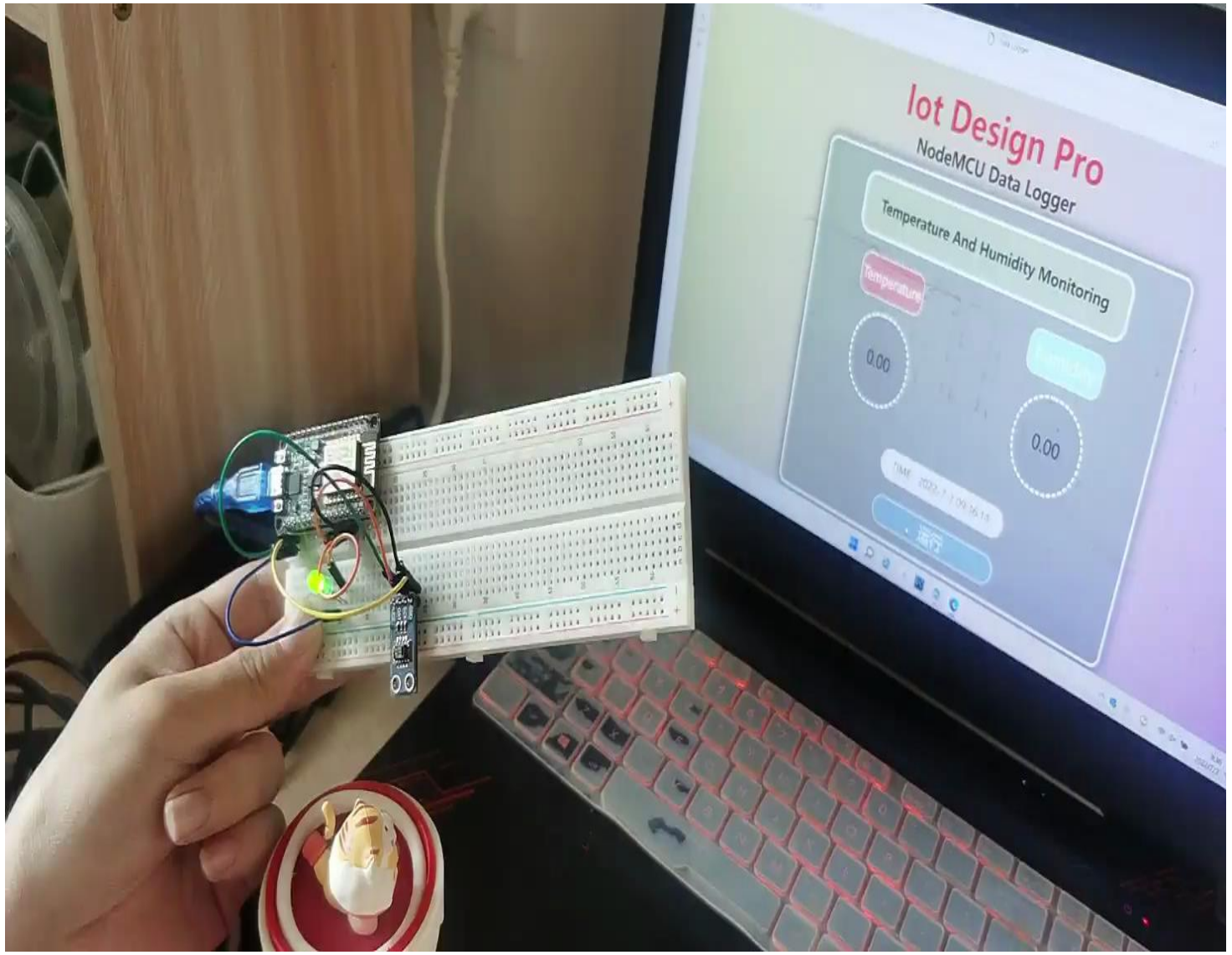
//Line 20
This page is called by Java Script AJAX

//Line 21 to 22
Server start

//end:Cilent request

Web Clip

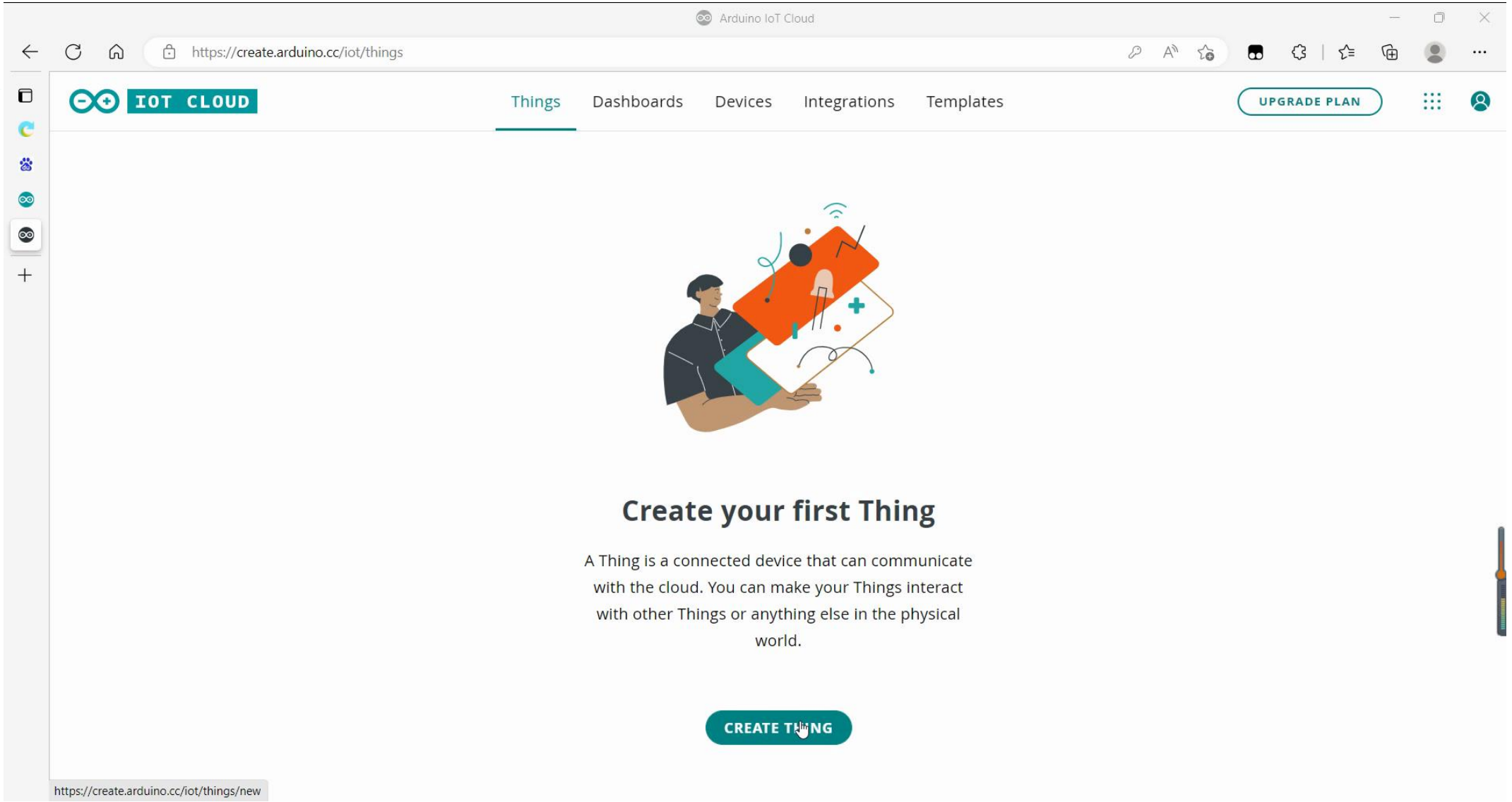

Monitoring
humidity
41.20
87.36
TIME: 2022-7-2 09:27:09
运行


Iot Design Pro
NodeMCU Data Logger
Temperature And Humidity Monitoring
Temperature
0.00
0.00

IOT Cloud Step 1:

- set three variables
- 1)click on “CREAT THING” in the “Things”
- 2)click on “ADD VARIABLES” to add three variables
- the first:name:temperature-->the type:flaot-->choose the “Read&Write” and “On change”
- the second:name:humilidity-->the type:flaot-->choose the “Read&Write” and “On change”
- the third:name:switch1-->the type:switch-->choose the “Read&Write” and “On change”
- 3)click on “select device”-->the name:Group2-->choose “set up 3rd parts”-->choose “NodeMCU 1.0”-->click on “continue”-->remember the “Device ID” and “Secret Key”
- 4)click on “Network”-->input the ”WIFI name”、”Password”、”Secretkey”-->save

IOT Cloud Step 1 Clip:



IOT Cloud Step 2:

click on Dashboards-->click on “BUILD DASHBOARD”-->click on “edit” and “ADD” to add seven Widgets

1)a push botton:the name:switch1-->the link var:connect the value “switch1”

2)the first value:the name:temperature-->the link var:connect the value “temperature”

3)the second value:the name:humilidity-->the link var:connect the value “humilidity”

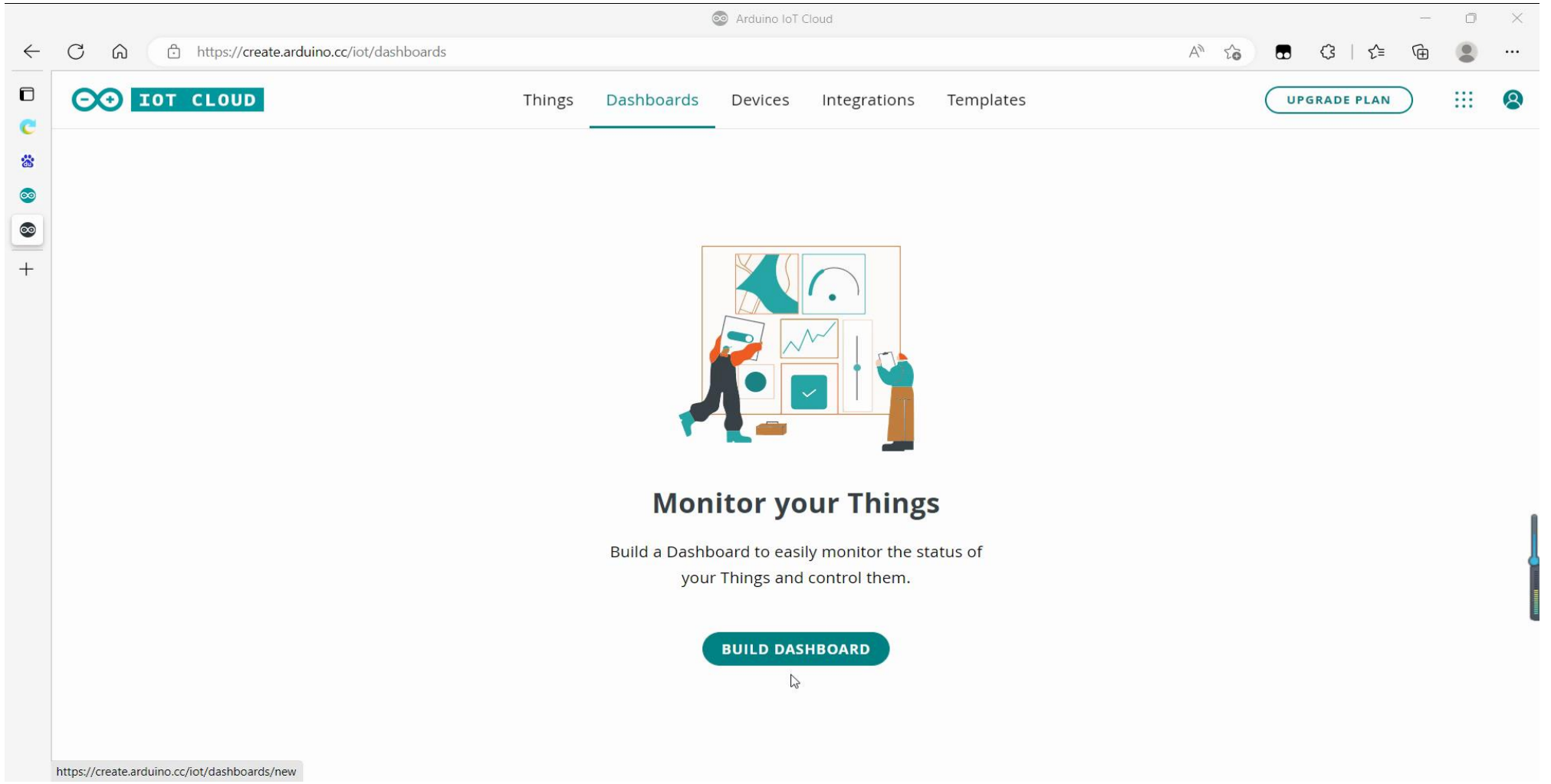
4)a gauge:the name:temperature-->the link var:connect the value “temperature”

5)a Pencertage:the name:humilidity-->the link var:connect the value “humilidity”

6)the first chart:the name:temperature chart-->the link var:connect the value “temperature”

7)the second chart:the name:humilidity chart-->the link var:connect the value “humilidity”

IOT Cloud Step 2 Clip:



IOT Cloud Step 3 and Clip:

come back the “Things”-->click on the “Group2”

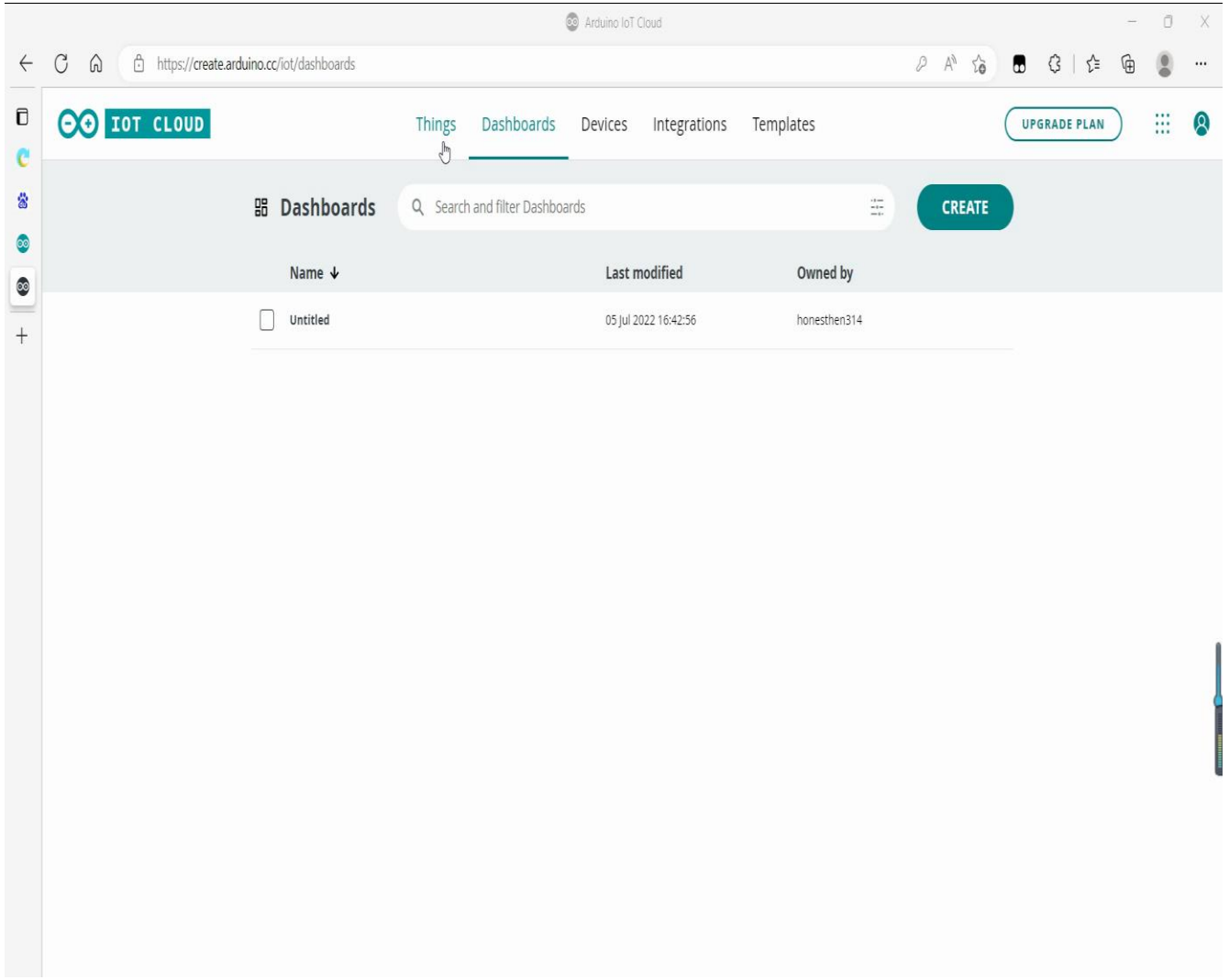
1)check the three variables

2)click on “ sketch”

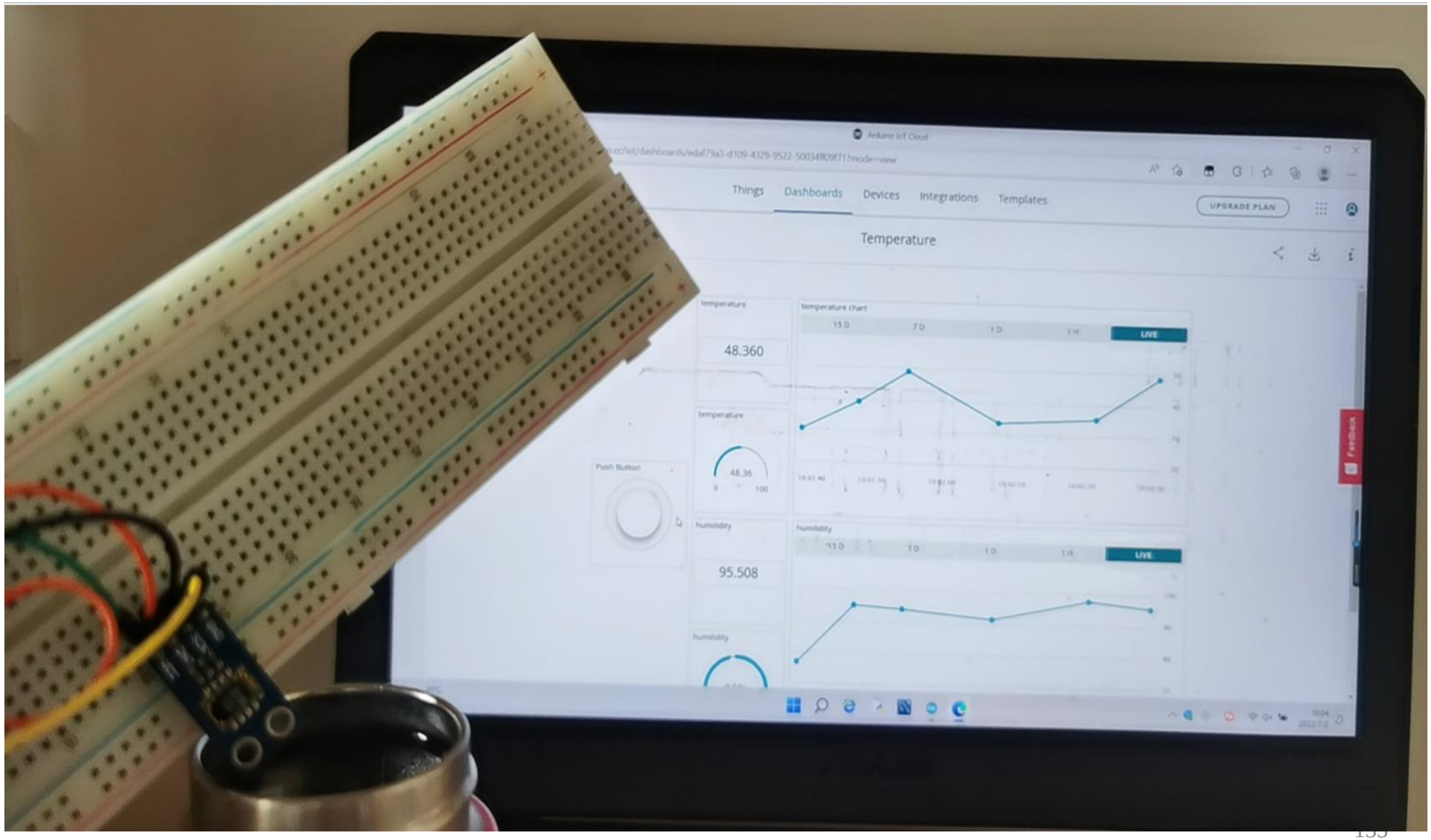
select the “open full editor” and install drivers

3)choose the “NodeMCU 1.0” and text the Code

4)If Success,the set is finish



IOT Cloud Step 3 result:



IOT Cloud Code:

- The Code comment

```
#include "thingProperties.h"
#include <SHT1x-ESP.h>
#define LED 0
#define dataPin  2
#define clockPin 4
SHT1x sht1x(dataPin, clockPin);
void setup() {
  pinMode(dataPin, INPUT);
  pinMode(clockPin, INPUT);
  pinMode(LED, OUTPUT);
  Serial.begin(9600);
   delay(1500);

   initProperties();
   ArduinoCloud.begin(ArduinoIoTPreferredConnection);
/*
     The following function allows you to obtain more information
     related to the state of network and IoT Cloud connection and errors
     the higher number the more granular information you'll get.
     The default is 0 (only errors).
     Maximum is 4
  */
 setDebugMessageLevel(2);
  ArduinoCloud.printDebugInfo();
}
```

//on board LED

//Initialize serial and wait for port to open

//This delay gives the chance to wait for a Serial Monitor without blocking if none is found

//define in thingProperties.h
//Connect to Arduino IOT Cloud

IOT Cloud Code:

- The Code comment

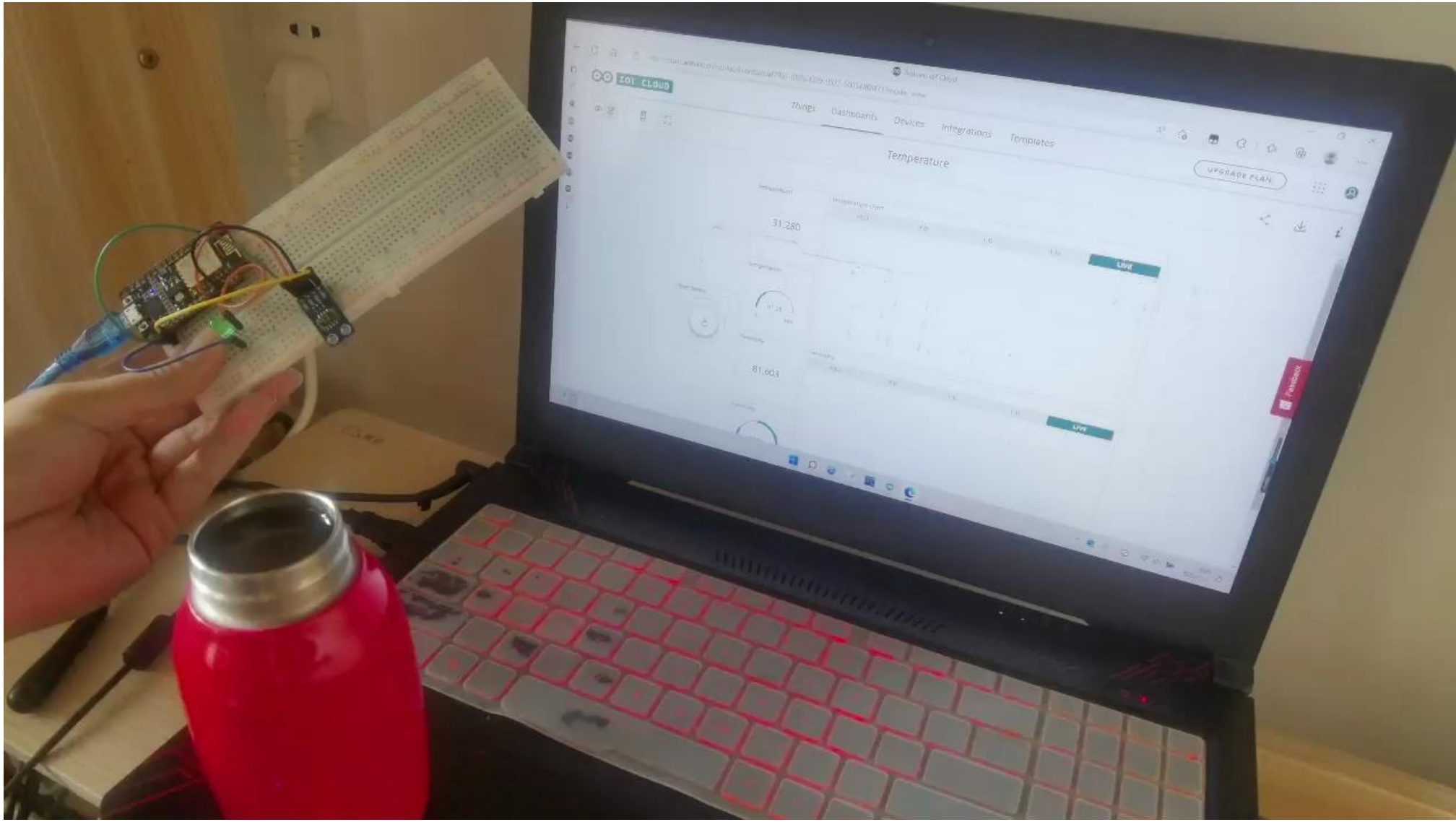
```
void loop() {
  ArduinoCloud.update();
}
void onSwtichChange()  {
  if (swtich) {
    temperature = sht1x.readTemperatureC();
    humidity = sht1x.readHumidity();
    digitalWrite(LED,HIGH);
  }
  else{
     digitalWrite(LED,LOW);
  }
 }
/*
  Since Temperature is READ_WRITE variable, onTemperatureChange() is
  executed every time a new value is received from IoT Cloud.
*/
void onTemperatureChange()  {
 }
/*
  Since Humidity is READ_WRITE variable, onHumidityChange() is
  executed every time a new value is received from IoT Cloud.
*/
void onHumidityChange()  {
 }
```

//our code here

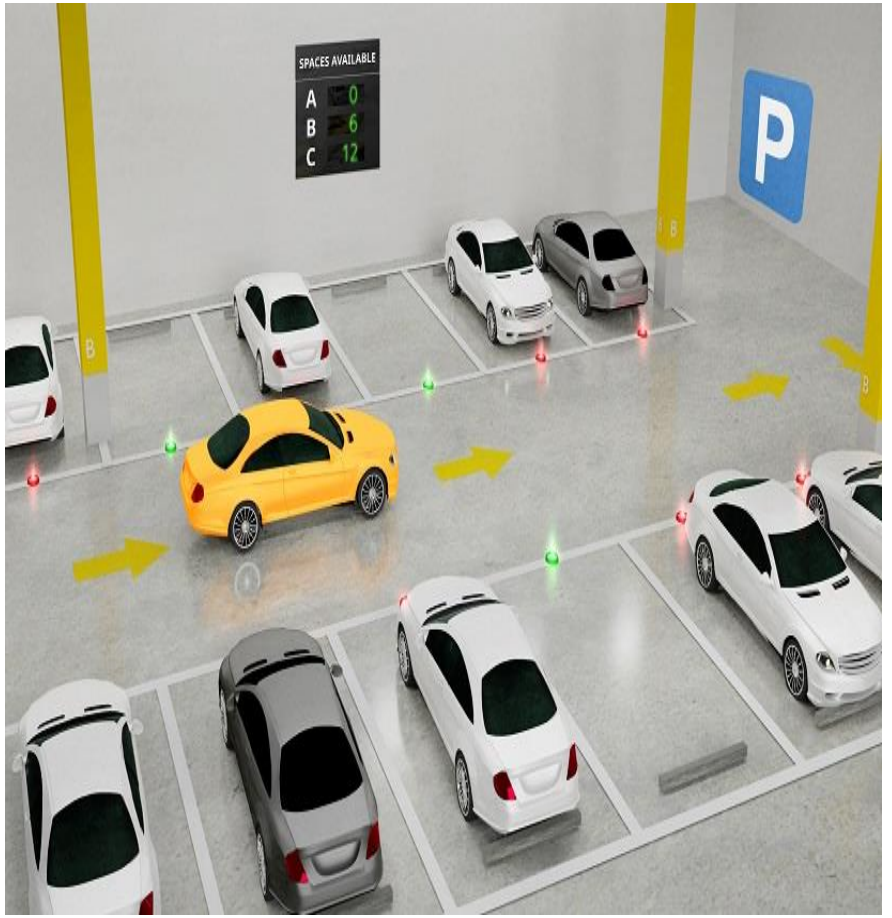
//through the switch to control to read the temperature's data and humilidity's data

//to control the LED(HIGH or LOW)

IOT Cloud Clip



Project4:



IoT based car parking System

Why we need Smart parking system

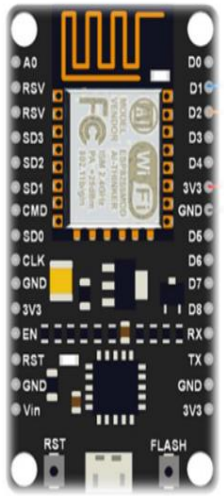
- Now days finding parking in busy areas is very hard and there is no system to get the details of parking availability online. Imagine if you can get the parking slot availability information on your phone and you don’t have roaming around to check the availability.
- This problem can be solved by the **IoT based smart parking system**. Using the IoT based parking system you can easily access the parking slot availability over the internet. This system can completely automate the car parking system. From your entry to the payment and exit all can be done automatically

Project 4:Component

IR Sensor 05

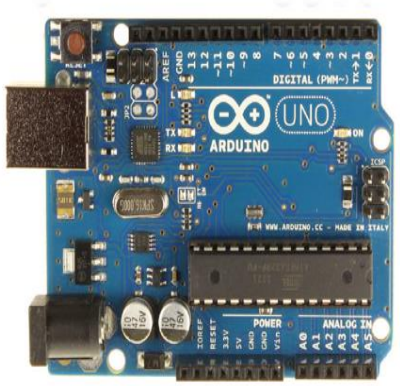
红外传感器*5

Node MCU ESP8266-12e



Nodemcu*1

Arduino uno



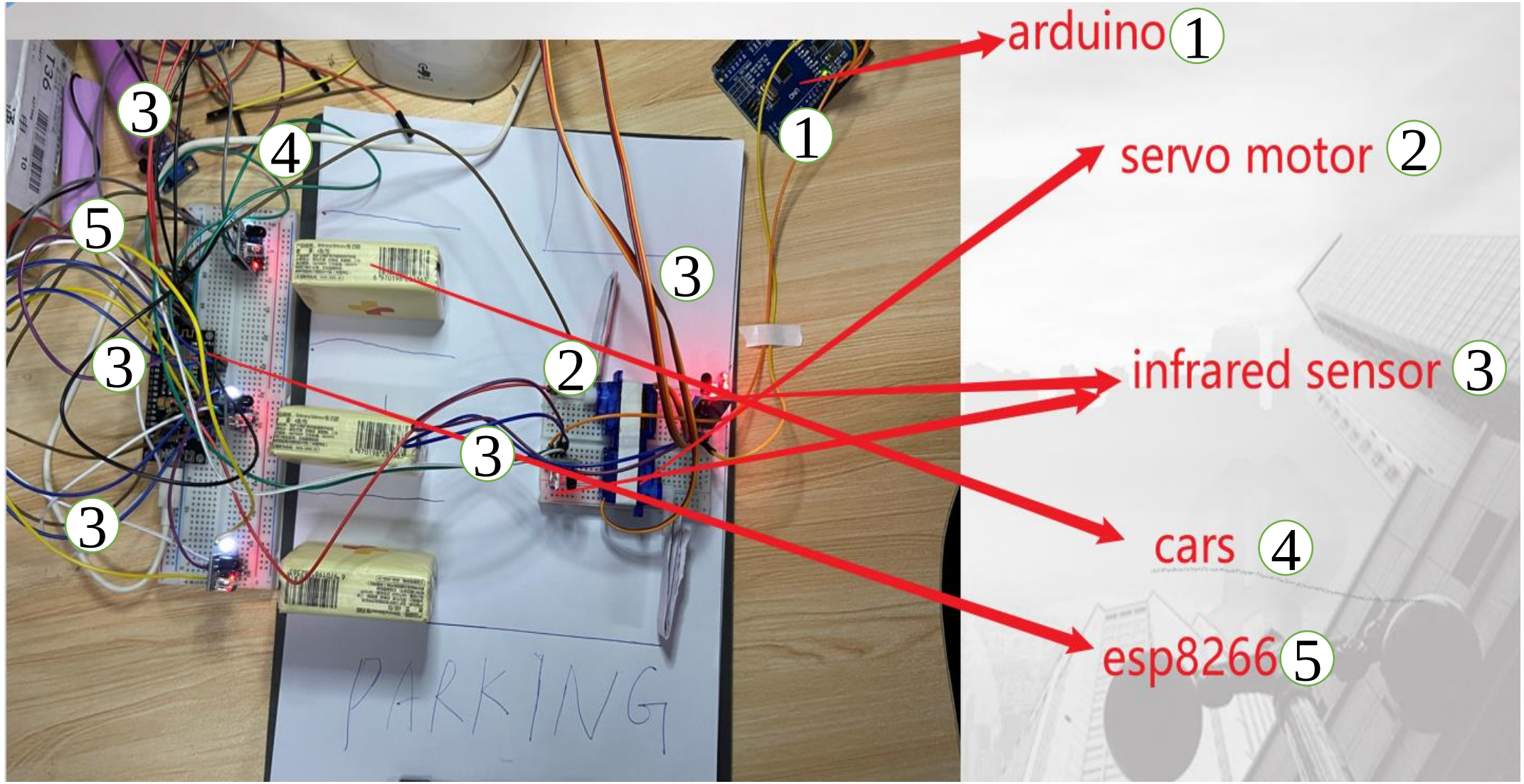
Arduino uno*1

Servo Motor

伺服电机*2

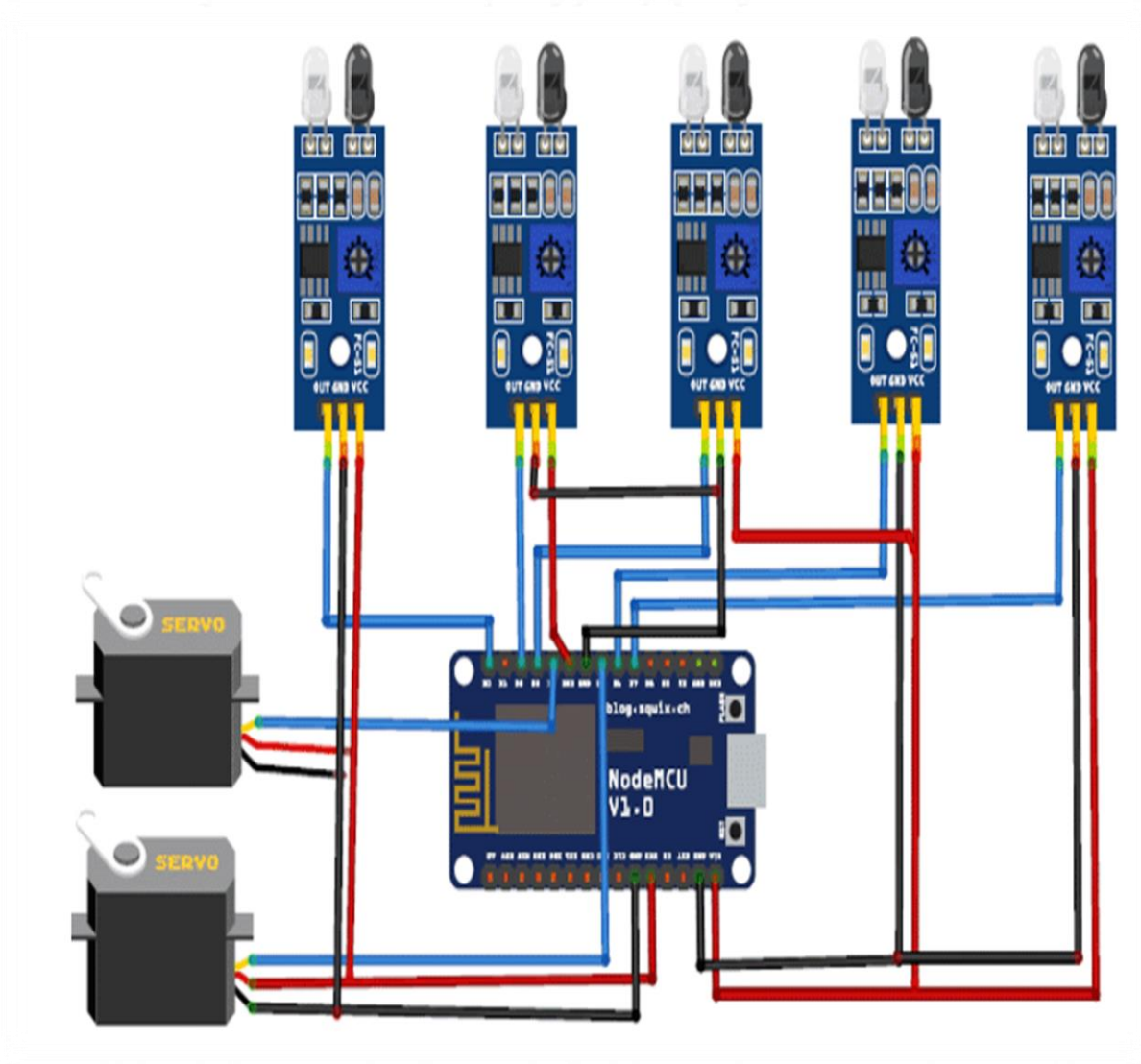
- IR Sesor:Five infrared sensors are used to detect whether vehicles are entering the parking lot and determine whether there is a vacant space.
- Nodemcu ESP8266: Record the parking space sensor and generate web page code.
- Arduino: Record the code that controls the servo motor.
- Servo Motor:Control the parking lot railings through code simulation.

Project 4:Component

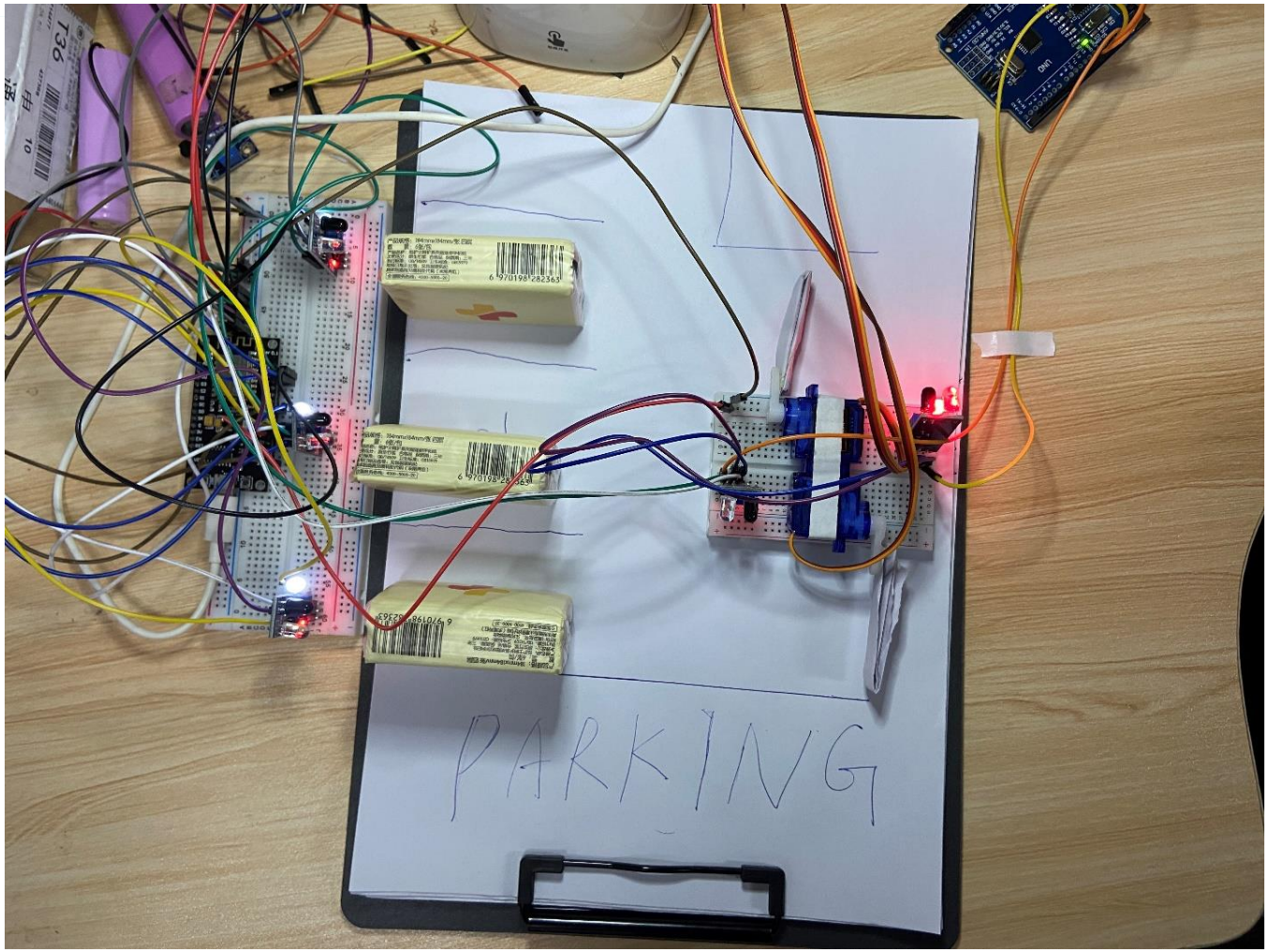


Project 4:Component

- In this Smart Parking System using IOT, we are using five IR Sensors and two servo motors. IR sensors and Servo motors are connected to the NodeMCU.
- NodeMCU controls the complete process and sends the parking availability and parking time information to Adafruit IO so that it can be monitored from anywhere in the world using this platform.
- Two IR sensors are used at entry and exit gate so that it can detect the cars at entry and exit gate and automatically open and close the gate.

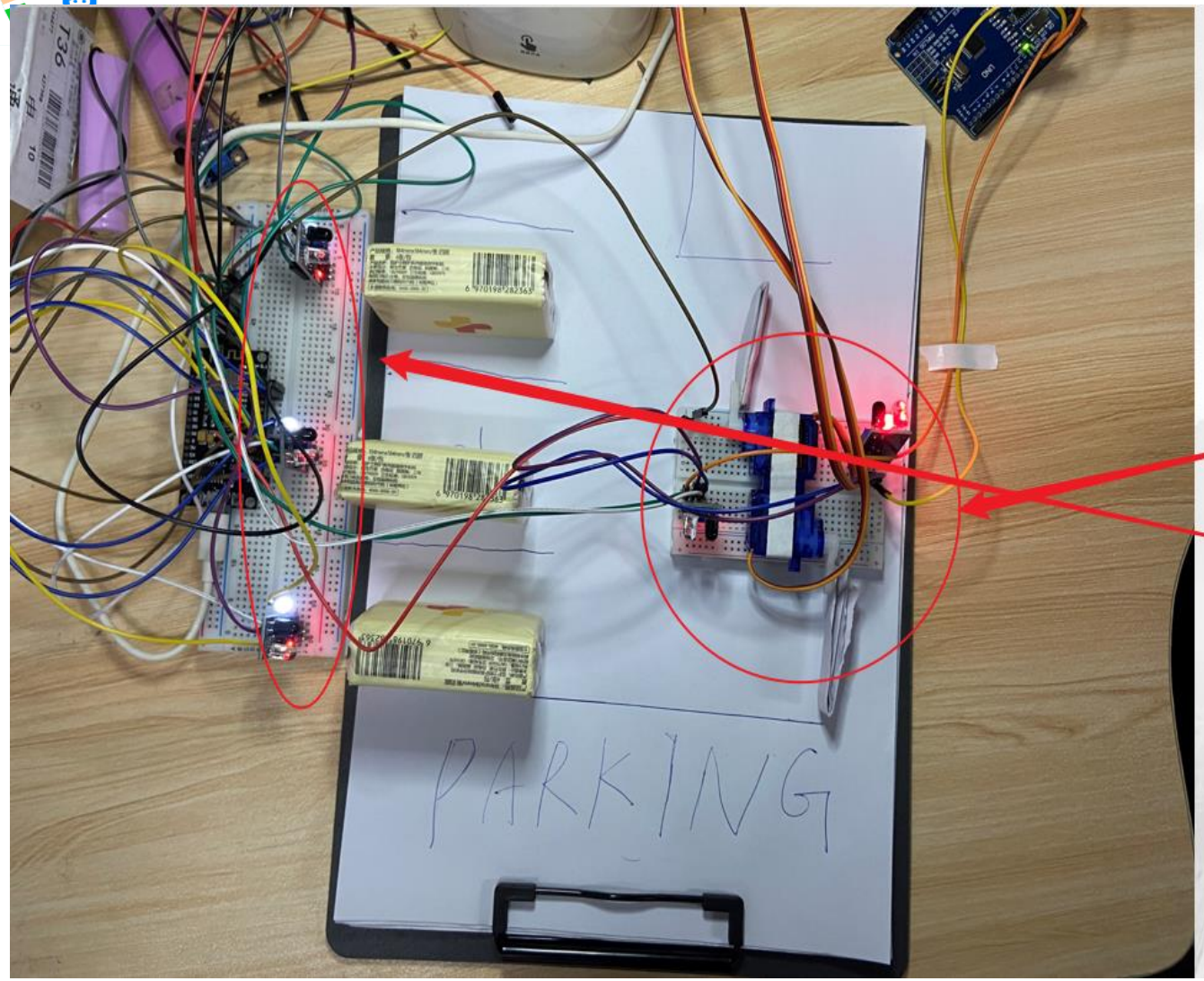


Project 4:Component



Arduino uno control the double servo motor ang the two Infrared sensor
The Nodemcu control the three Infrared sensor

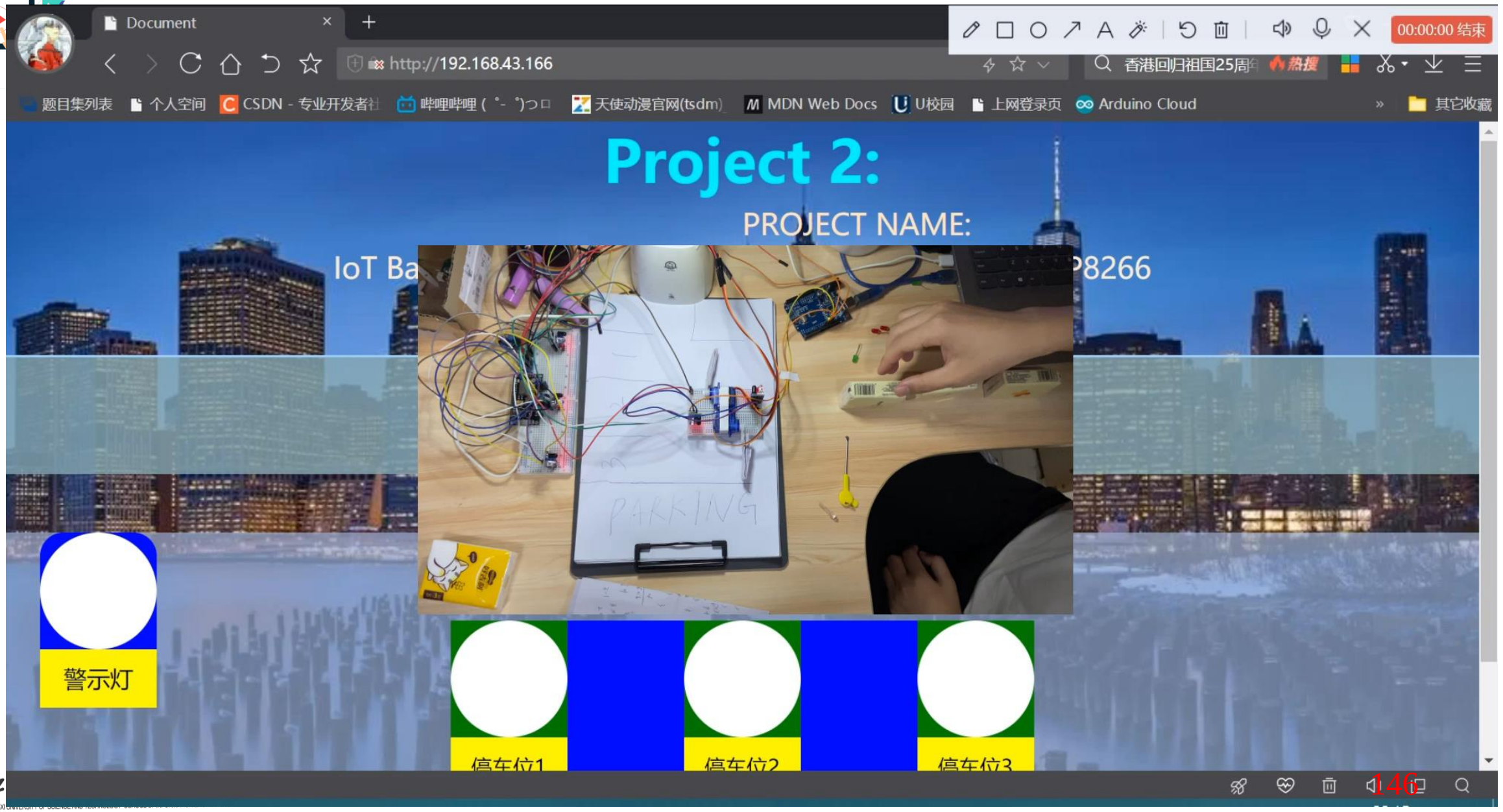
Project 4:Component



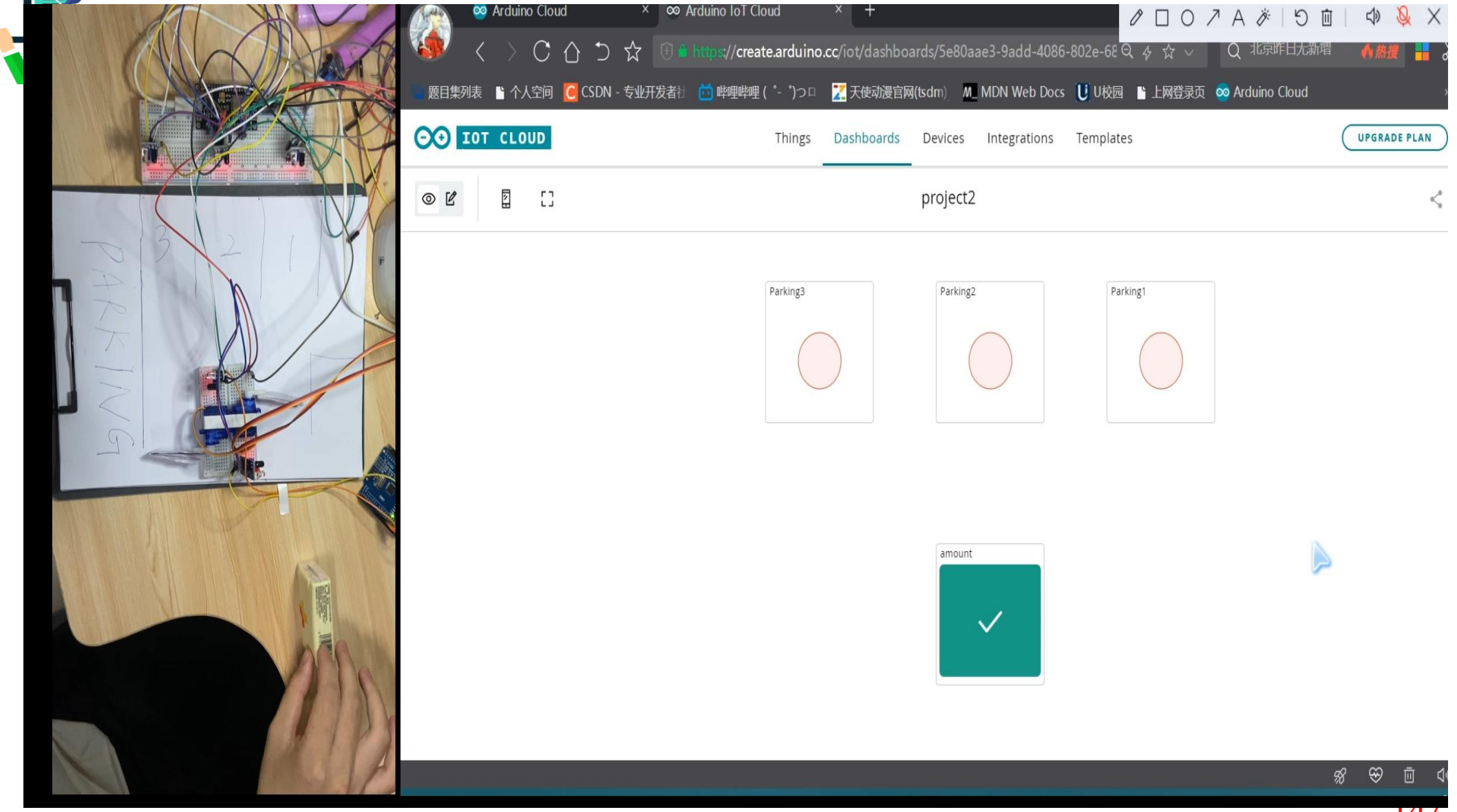
Arduino uno control the double servo motor ang the two Infrared sensor

The Nodemcu control the three Infrared sensor

Project 4: Demo on web



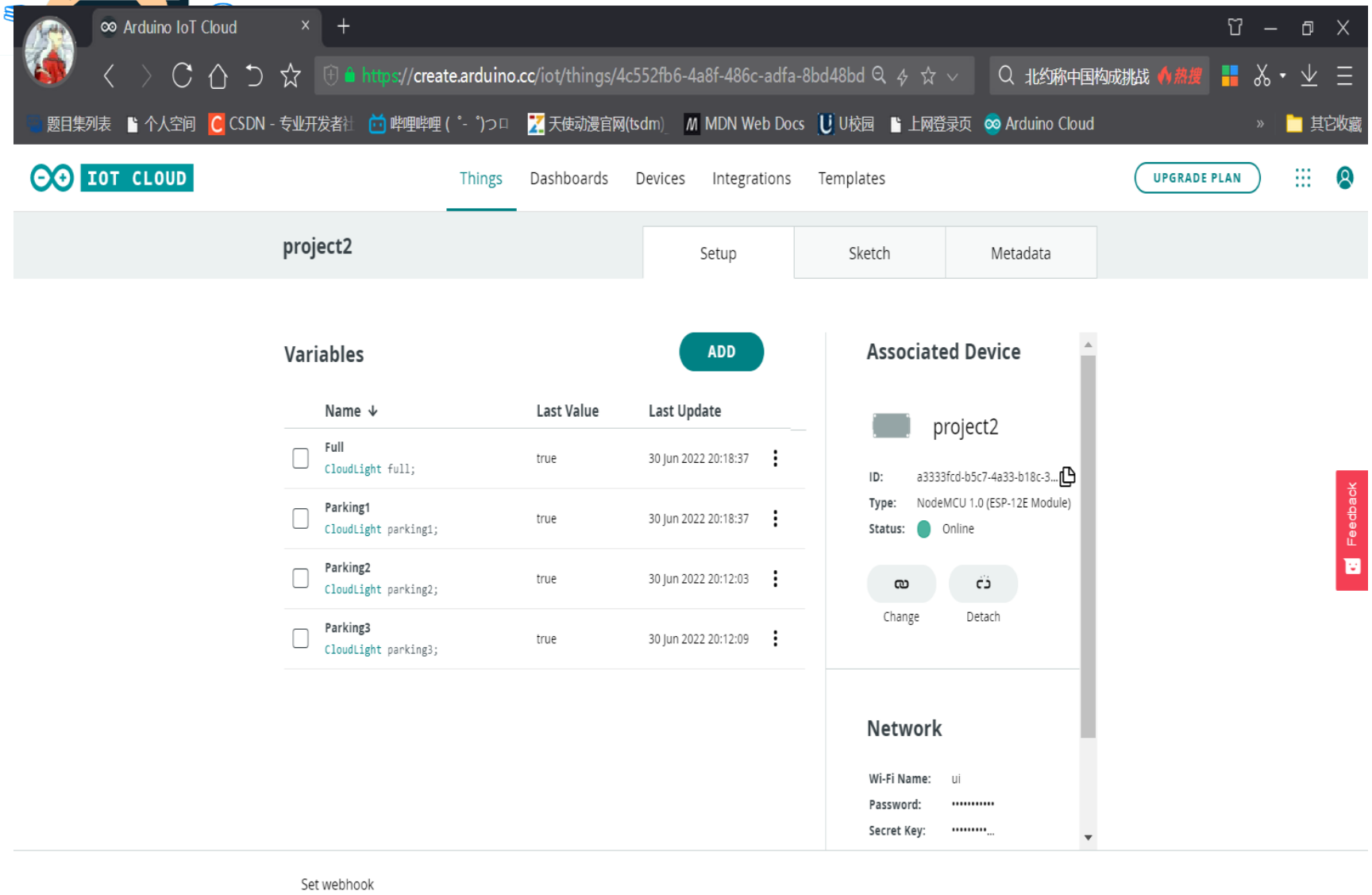
Project 4: Demo Arduino IOT Cloud



Project4:

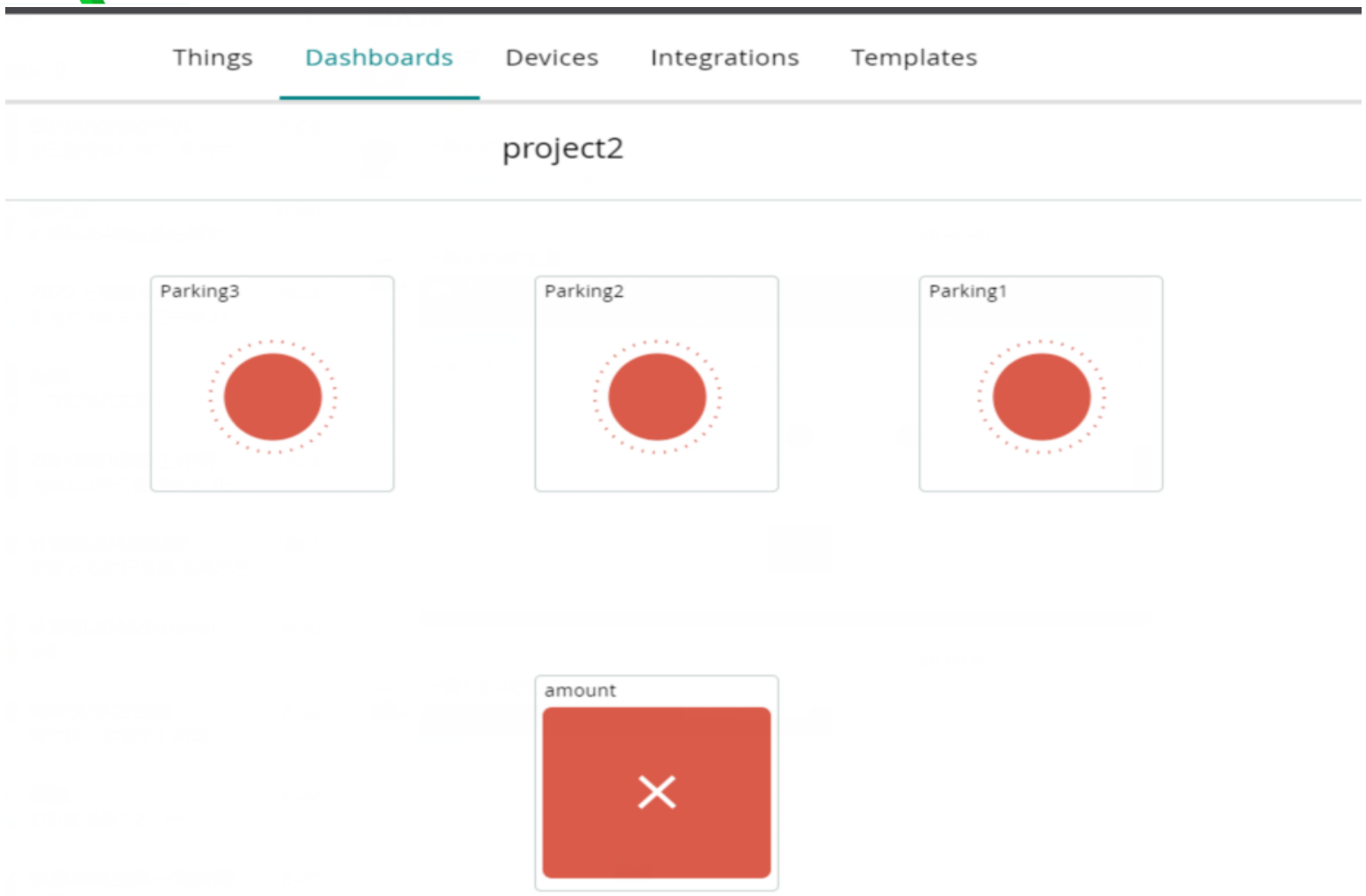
Code Section

Arduino IOT



The first parking space has a car when Paking1 lights up red
And so on
Full lights up when all three parking spaces have cars

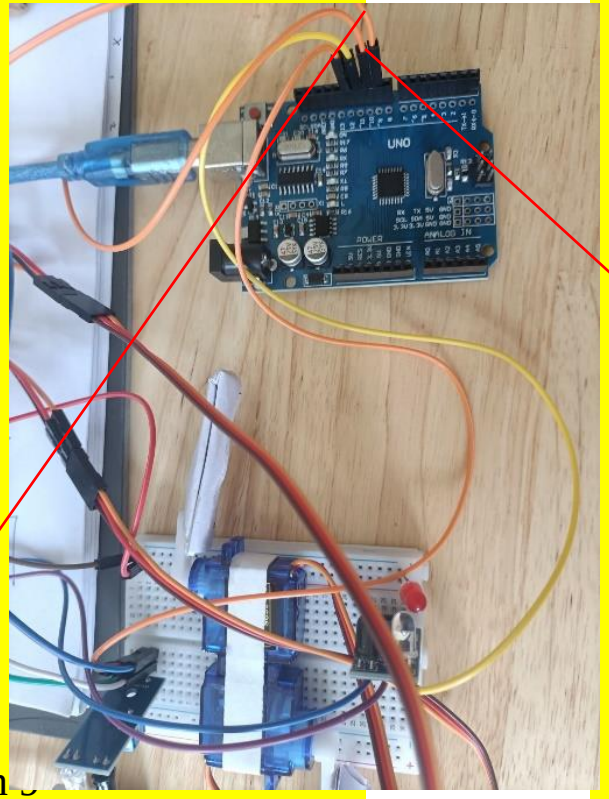
Arduino IOT



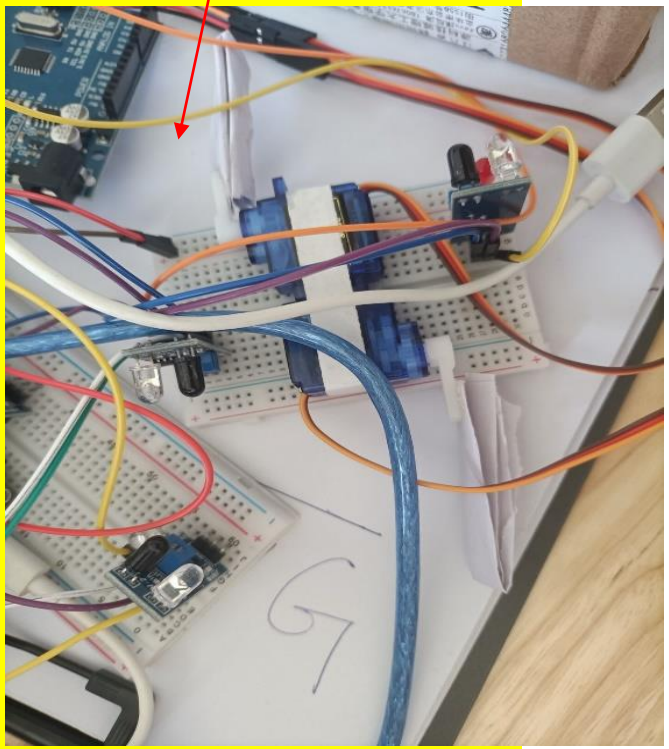
The first parking space has a car when Paking1 lights up red
And so on
Full lights up when all three parking spaces have cars

Arduino code

```
#include<Servo.h>
Servo ms1;
Servo ms2;
int value1=0;
int value2=0;
void setup() {
  pinMode(11,INPUT);
  pinMode(12,INPUT);
}

void loop() {
  int value1=digitalRead(11);
  int value2=digitalRead(12);
  if(value1==LOW)
  {
    ms1.attach(9);//connect to pin 9
    ms1.write(80);
//小于90往0度转,angel=80<90,the stick go to  0°
    delay(680);
```

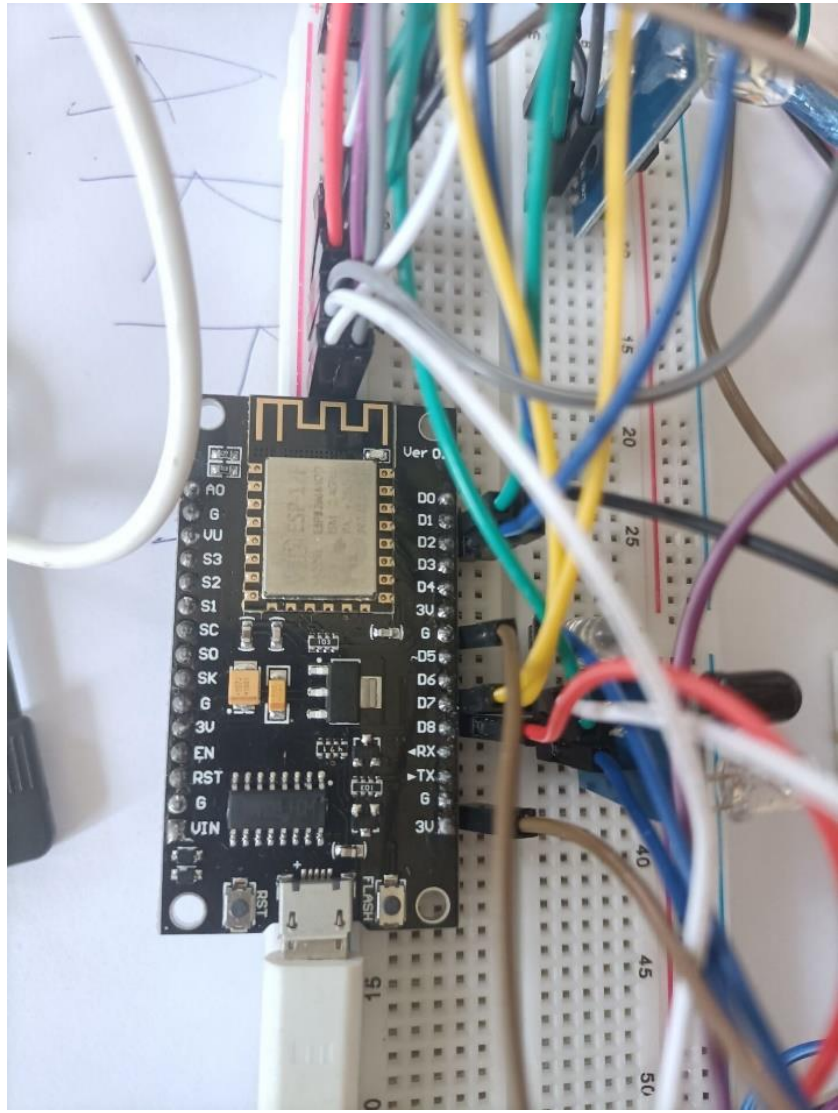


```
 ms1.detach();//disconnect to pin 9
    delay(3000);
    ms1.attach(9);
    ms1.write(100);
//大于90往180度转,angel=100>90,the stick go to  180°
    delay(820);
    ms1.detach();
  }
  if(value2==LOW)
  {
    ms2.attach(10);
    ms2.write(80);
    delay(680);
    ms2.detach();
    delay(3000);
    ms2.attach(10);
    ms2.write(100);
    delay(800);
    ms2.detach();
  }
  delay(3000);
}
```



Explain the esp8266 iot code

```
/*   Sketch generated by the Arduino IoT Cloud Thing "Untitled 2"
   Arduino IoT Cloud Variables description
  The following variables are automatically generated and updated when changes are made to the Thing
  CloudLight full;
  CloudLight parking1;
  CloudLight parking2;
  CloudLight parking3;

  Variables which are marked as READ/WRITE in the Cloud Thing will also have functions
  which are called when their values are changed from the Dashboard.
  These functions are generated with the Thing and added at the end of this sketch.
*/

#include "thingProperties.h"
#define LED 5
#define LED1 4
#define LED2 13
#define LED3 15
void setup() {
  // Initialize serial and wait for port to open:
  Serial.begin(9600);
  // This delay gives the chance to wait for a Serial Monitor without blocking if none is found
  delay(1500);
```



Explain the esp8266 iot code

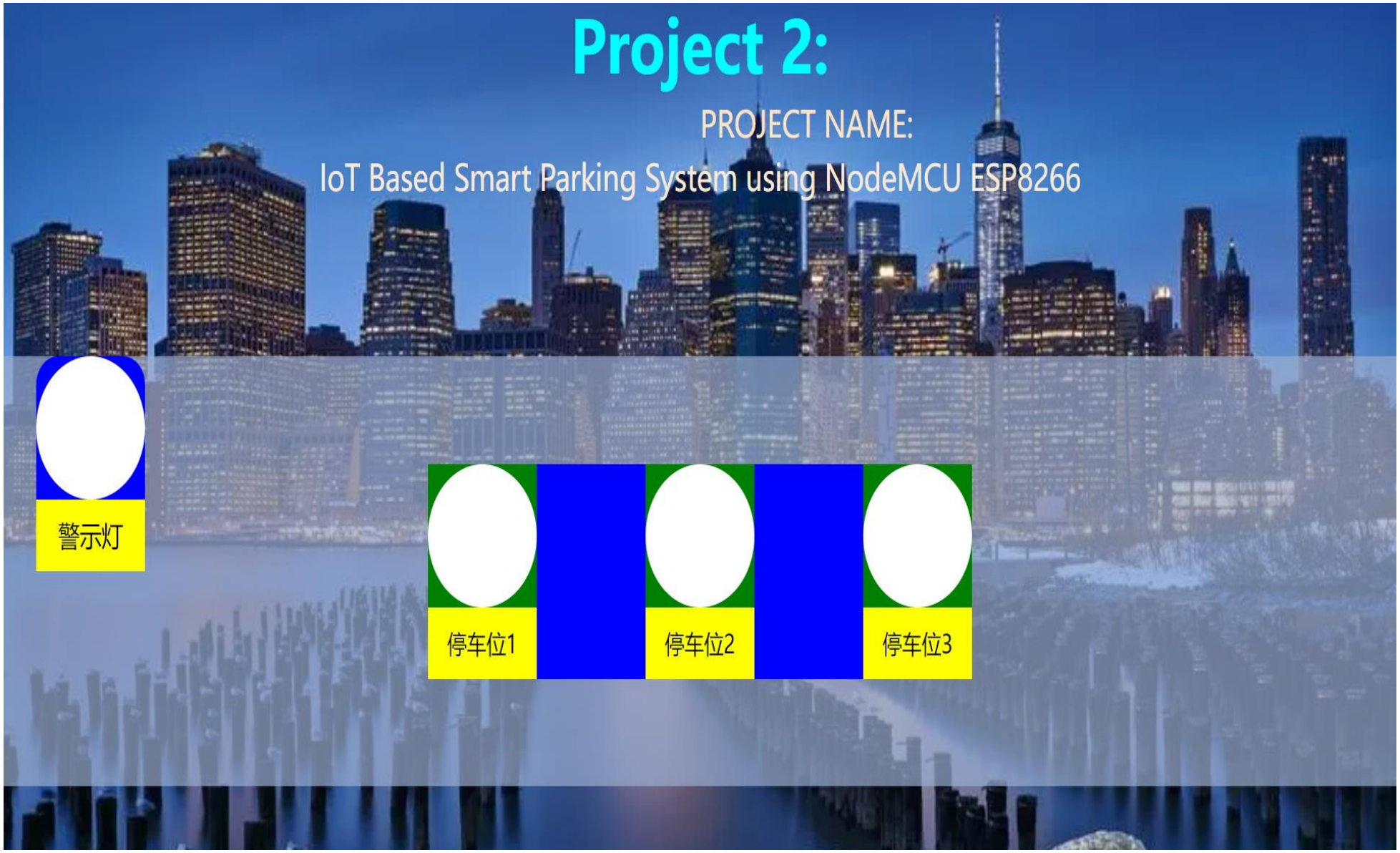
```
pinMode(LED,INPUT);
  pinMode(LED1,INPUT);
  pinMode(LED2,INPUT);
  pinMode(LED3,OUTPUT);
  // Defined in thingProperties.h
  initProperties();

  // Connect to Arduino IoT Cloud
  ArduinoCloud.begin(ArduinoIoTPreferredConnection);

  /*
     The following function allows you to obtain more information
     related to the state of network and IoT Cloud connection and errors
     the higher number the more granular information you'll get.
     The default is 0 (only errors).
     Maximum is 4
 */
  setDebugMessageLevel(2);
  ArduinoCloud.printDebugInfo();
}
```

```
void loop() {
  ArduinoCloud.update();
  // Your code here
  if(digitalRead(LED)==1){
    parking1=LOW;
  }else
  {
    parking1=HIGH;
  }
  if(digitalRead(LED1)==1){
    parking2=LOW;
  }else
  {
    parking2=HIGH;
  }
  if(digitalRead(LED2)==1){
    parking3=LOW;
  }else
  {
    parking3=HIGH;
  }
```

Explain the esp8266 iot code

```
 int l1=digitalRead(LED);
  int l2=digitalRead(LED1);
  int l3=digitalRead(LED2);
  if((l1||l2||l3)==0)
  {
   digitalWrite(LED3,HIGH);
  }
  else
  {
   digitalWrite(LED3,LOW);
  }
  if(digitalRead(LED3)==1){
    full=HIGH;
  }else
  {
    full=LOW;
  }
}

/*
  Since Parking1 is READ_WRITE variable, onParking1Change() is
  executed every time a new value is received from IoT Cloud.
*/
void onParking1Change()  {
  // Add your code here to act upon Parking1 change
if(digitalRead(LED)==1){
    parking1=HIGH;
  }else
  {
    parking1=LOW;
  }
}
/*
Since Parking2 is READ_WRITE variable, onParking2Change() is
  executed every time a new value is received from IoT Cloud.
*/
void onParking2Change()  {
  // Add your code here to act upon Parking2 change
}

/*
  Since Parking3 is READ_WRITE variable, onParking3Change() is
  executed every time a new value is received from IoT Cloud.
*/
void onParking3Change()  {
  // Add your code here to act upon Parking3 change
}

/*
  Since Full is READ_WRITE variable, onFullChange() is
  executed every time a new value is received from IoT Cloud.
*/
void onFullChange()  {
  // Add your code here to act upon Full change}
```

Web pag Demo


Project 2:
PROJECT NAME:
IoT Based Smart Parking System using NodeMCU ESP8266
警示灯
停车位1
停车位2
停车位3

Explain the Web code

```
#include <ESP8266WiFi.h>
 const char* ssid = "钟";
 const char* password = "244466666";
WiFiServer server(80);
 void setup()
{
  Serial.begin(115200); //Default Baudrate
  pinMode(5,INPUT);
 pinMode(4,INPUT);
 pinMode(13,INPUT);
 pinMode(15,OUTPUT);
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();
  Serial.println("Server started");
   Serial.print("IP Address of network: ");
  Serial.println(WiFi.localIP());
  Serial.print("Copy and paste the following URL: https://");
  Serial.print(WiFi.localIP());
  Serial.println("/");
}
```

```
void loop()
{
   int l1=digitalRead(5);
 int l2=digitalRead(4);
 int l3=digitalRead(13);
 if((l1||l2||l3)==LOW)
 {
  digitalWrite(15,HIGH);
 }
 else
 {
  digitalWrite(15,LOW);
 }
 int a=digitalRead(5);
 int b=digitalRead(4);
 int c=digitalRead(13);
 int d=digitalRead(15);
  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();
```

Explain the Web code

```
//*------------------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 lang=\"en\">");
client.println("<head><meta charset=\"UTF-8\">");
  client.println("<meta http-equiv=\"refresh\" content=\"3\">");//刷新
  client.println("<meta http-equiv=\"X-UA-Compatible\" content=\"IE=edge\">");
  client.println("<meta name=\"viewport\" content=\"width=device-width, initial-scale=1.0\">");
  client.println("<title>Document</title></head>");
  client.println("<style>* {padding: 0px;margin: 0px;}");
  client.println("a {text-decoration: none;}");
  client.println("li {list-style: none;}");
```

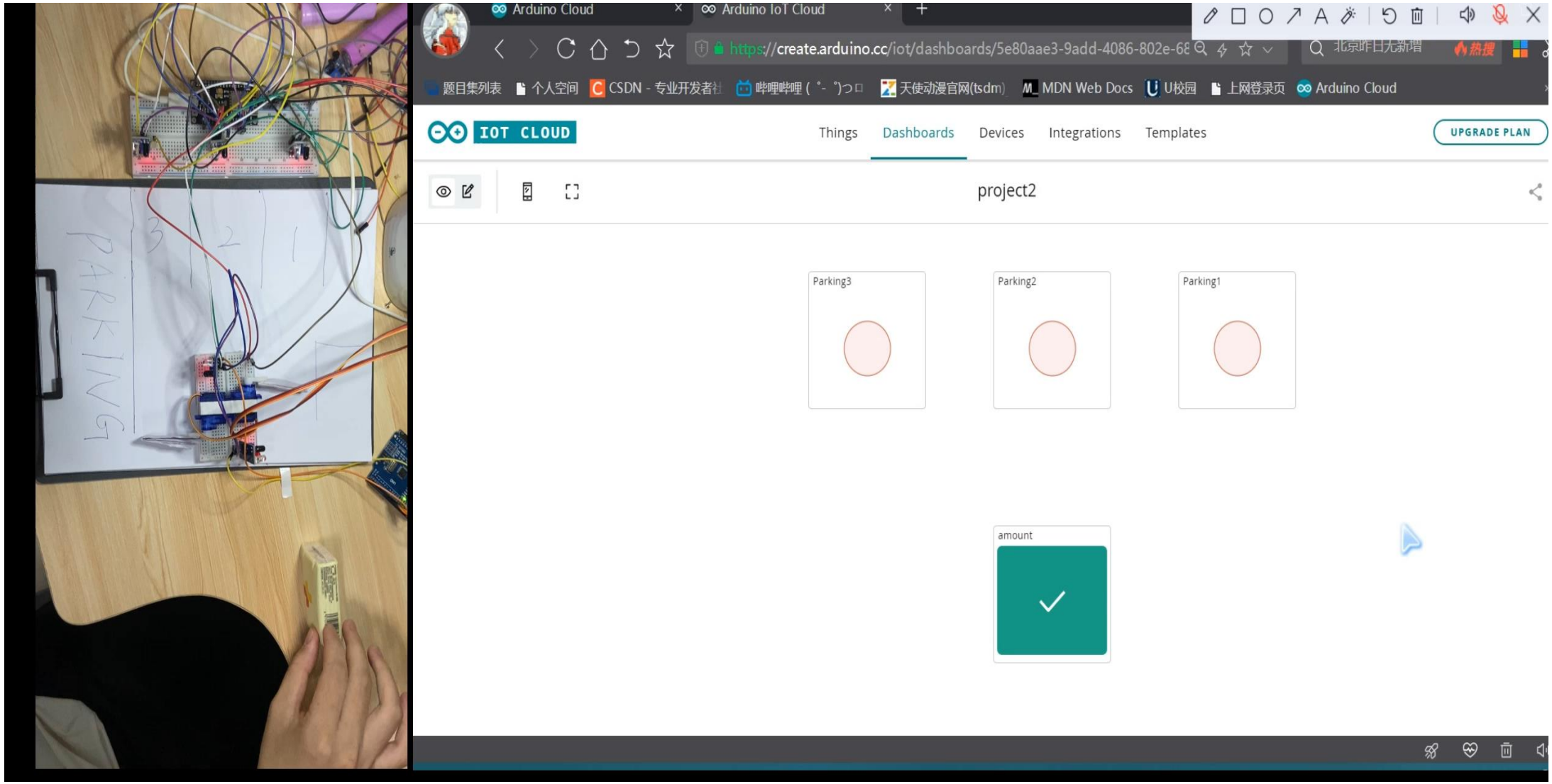
Explain the Web code

```
client.println("body {background: url(https://img2.baidu.com/it/u=1146935794,3666555888&fm=253&fmt=auto&app=120&f=JPEG?w=889&h=500) no-repeat;background-size: cover;}");
 client.println(".header {height: 200px;width: 100%;background:0%;}");
 client.println(".header h1 {height: 60px;width: 100%;text-align: center;font-size: 50px;color: aqua;}");
 client.println(".header div {width: 100%;height: 90px;}");
 client.println(".header div p:nth-child(1) {padding-left: 50%;height: 25px;padding-top: 15px;line-height: 25px;font-size: 25px;color: bisque;}");
 client.println(".header div p:nth-child(2) {height: 50px;text-align: center;line-height: 50px;font-size: 25px;color: antiquewhite;}");
 client.println(".bgbox {width: 100%;height: 300px;position: relative;background: 0%;}");
 client.println(".bgbox .border {width: 100%;height: 300px;position: relative;background: rgb(154, 173, 208,0.7);}");
 client.println(".bgbox .border ul {width: 500px;height: 150px;position: absolute;background: blue;top: 50%;margin-top: -75px;left: 50%;margin-left: -250px;}");
 client.println(".bgbox .border ul li {height: 150px;float: left;margin-right: 100px;background: green;}");
 client.println(".bgbox .border ul li .light {width: 100px;height: 100px;background-color: white;border-radius: 50%;}");
 client.println(".bgbox .border ul li .part {width: 100%;height: 50px;background: yellow;font-size: 18px;text-align: center;line-height: 50px;}");
 client.println(".bgbox .border ul li:nth-child(3) {margin-right: 0px;}");
 client.println(".bgbox .border .ispart {width: 100px;height: 150px;border-radius: 20px 20px 0px 0px;margin-top: 50px;margin-left: 30px;background: blue;}");
 client.println(".bgbox .border .ispart .ispartlight {width: 100px;height: 100px;border-radius: 50%;background: white;}");
 client.println(".bgbox .border .ispart .message { width: 100px;height: 50px;background: yellow;text-align: center;line-height: 50px;font-size: 20px;}</style>");
 client.println("<body>");
 client.println("<div class=\"header\"><h1>Project 2:</h1><div><p>PROJECT NAME:</p><p>IoT Based Smart Parking System using NodeMCU ESP8266</p></div></div>");
 client.println("<div class=\"bgbox\"><div class=\"seabox\"><p class=\"search\"><input type=\"text\"><a href=\"#\"><span>搜索</span></a></p></div>");
 client.println("<div class=\"border\"><ul><li><div class=\"light\"></div><div class=\"part\">停车位1</div></li>");
 client.println("<li><div class=\"light\"></div><div class=\"part\">停车位2</div></li>");
 client.println("<li><div class=\"light\"></div><div class=\"part\">停车位3</div></li></ul>");
 client.println("<div class=\"ispart\"><div class=\"ispartlight\"></div><div class=\"message\">警示灯</div></div></div></div></body>");
 client.println("<script>");
```

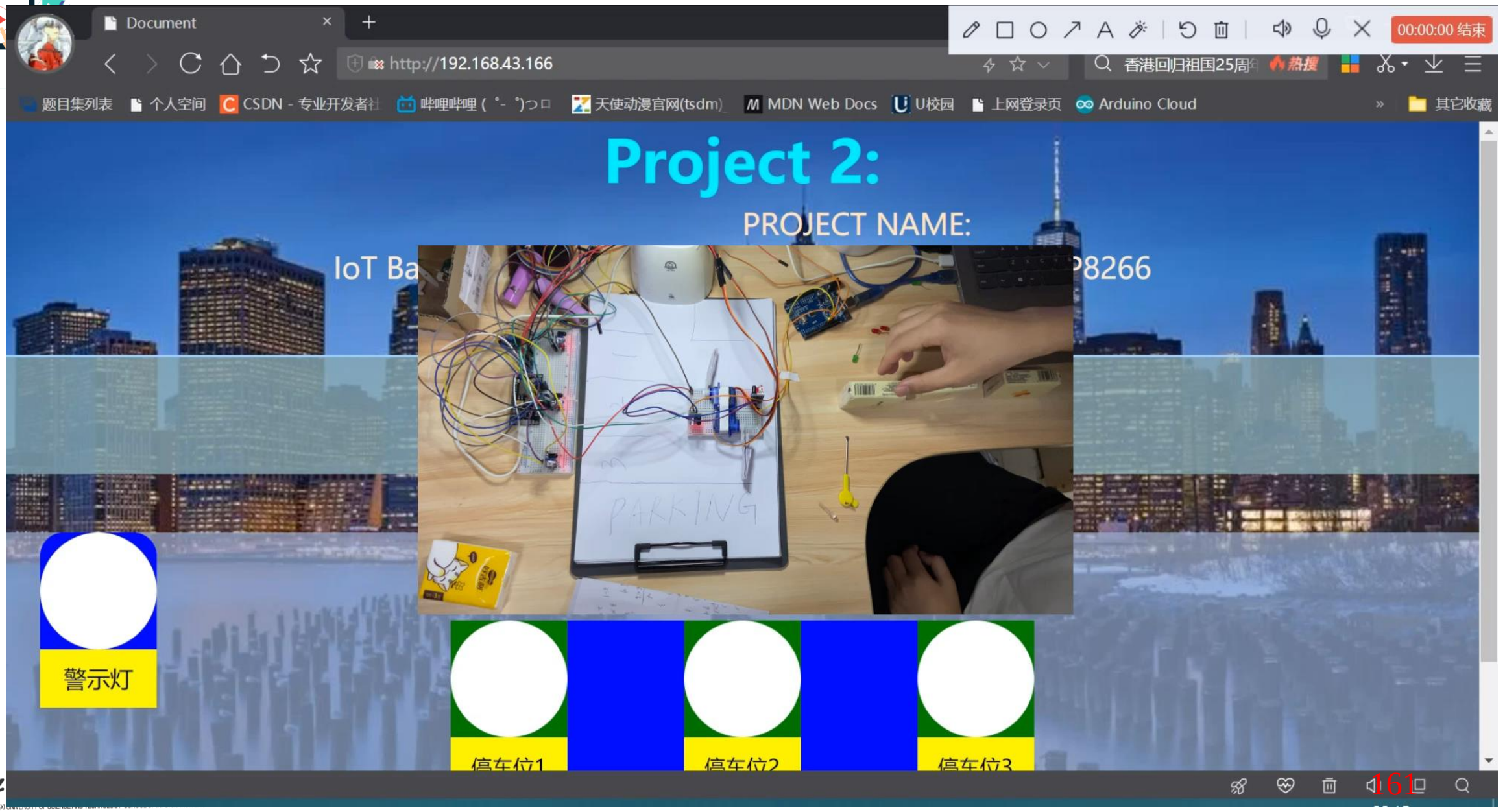
Explain the Web code

```
 client.println("var lis = document.getElementsByClassName('light');");
if(a == LOW)
 {
  client.print("lis[0].style.background = 'red';");
 }
 else
 {
  client.print("lis[0].style.background = 'white';");
 }
 if(b == LOW)
 {
  client.print("lis[1].style.background = 'red';");
 }
 else
 {
  client.print("lis[1].style.background = 'white';");
 }
 if(c == LOW)
 {
  client.print("lis[2].style.background = 'red';");
 }
 else
 {
  client.print("lis[2].style.background = 'white';");
 }
 client.println("var lis1 = document.getElementsByClassName('ispartlight');");
 if(d == HIGH)
  {
   client.print("lis1[0].style.background = 'red';");
  }
  else
  {
   client.print("lis1[0].style.background = 'white';");
  }
  client.println("</script>");

  client.println("</html>");

  delay(1);
  Serial.println("Client disonnected");
  Serial.println("");
}
```

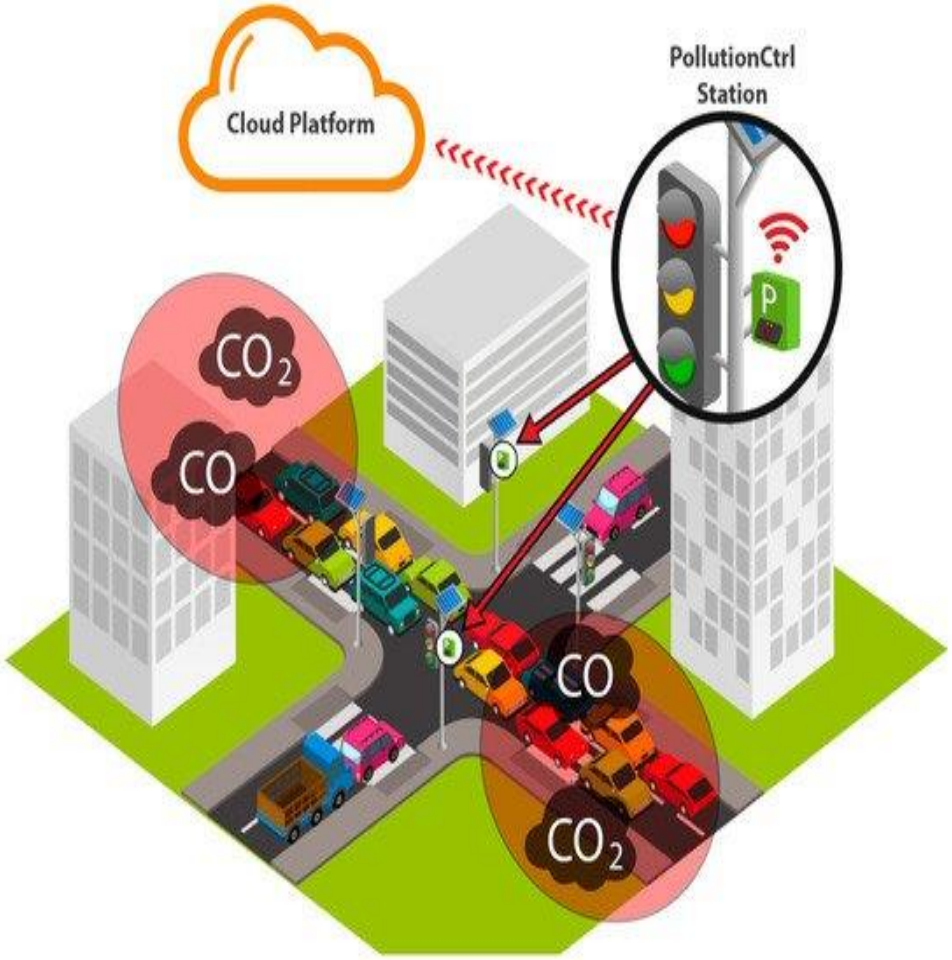
Experimental video in iot



Experimental video in HTML



Project5:



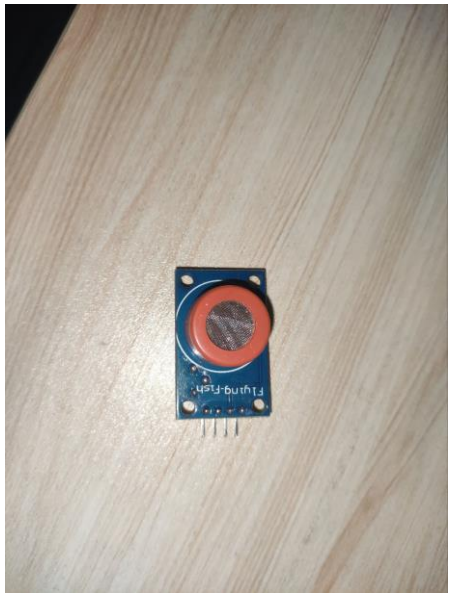
IoT-based car Gas Detection System(air quality)

Project 5: IoT-based car Gas Detection System

- Indoor air pollution has been consistently ranked by the US Environmental Protection Agency (EPA) and its Science Advisory Board to be among the top five environmental public health risks.
- Average person spends an estimated 90% of their time indoors so that poor indoor air quality (IAQ) poses a substantial risk to public health. Poor air quality may cause increased short-term health problems such as fatigue and nausea as well as chronic respiratory diseases, heart disease, and lung cancer.

Air Quality Index (AQI) Values)	Levels of Health Concern	Colors
When the AQI is in this range:	*...air quality conditions are:*	*..as symbolized by this color:*
0-50	Good	Green
51-100	Moderate	Yellow
101-150	Unhealthy for Sensitive Groups	Orange
151 to 200	Unhealthy	Red
201 to 300	Very Unhealthy	Purple
301 to 500	Hazardous	Maroon

Project 5:Component



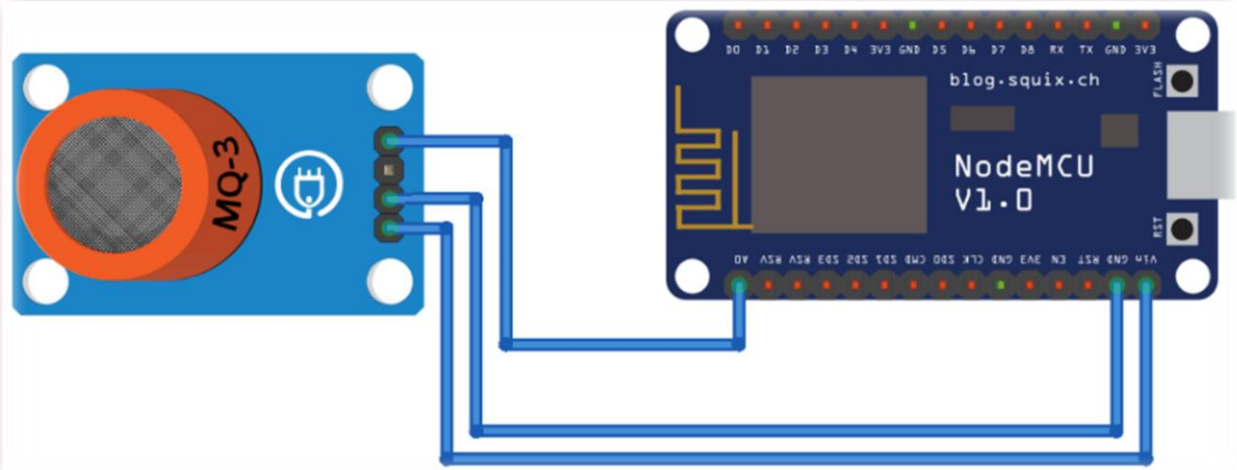
MQ-3

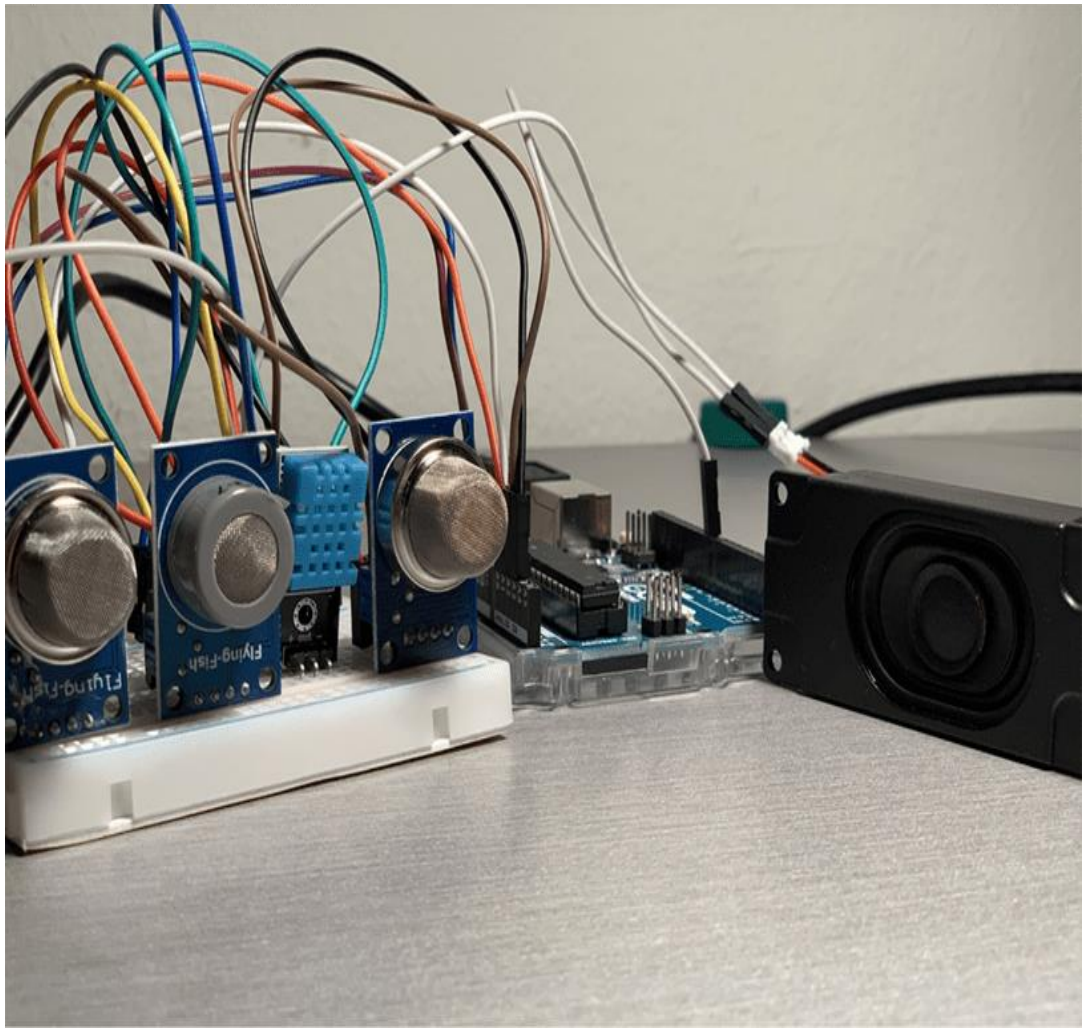
The basic principle of the alcohol sensor MQ-3 can be briefly described as a device that converts the detected alcohol concentration into useful electrical signals. According to the strength of these electrical signals, information related to the presence of the gas to be measured in the environment can be obtained

ESP8266

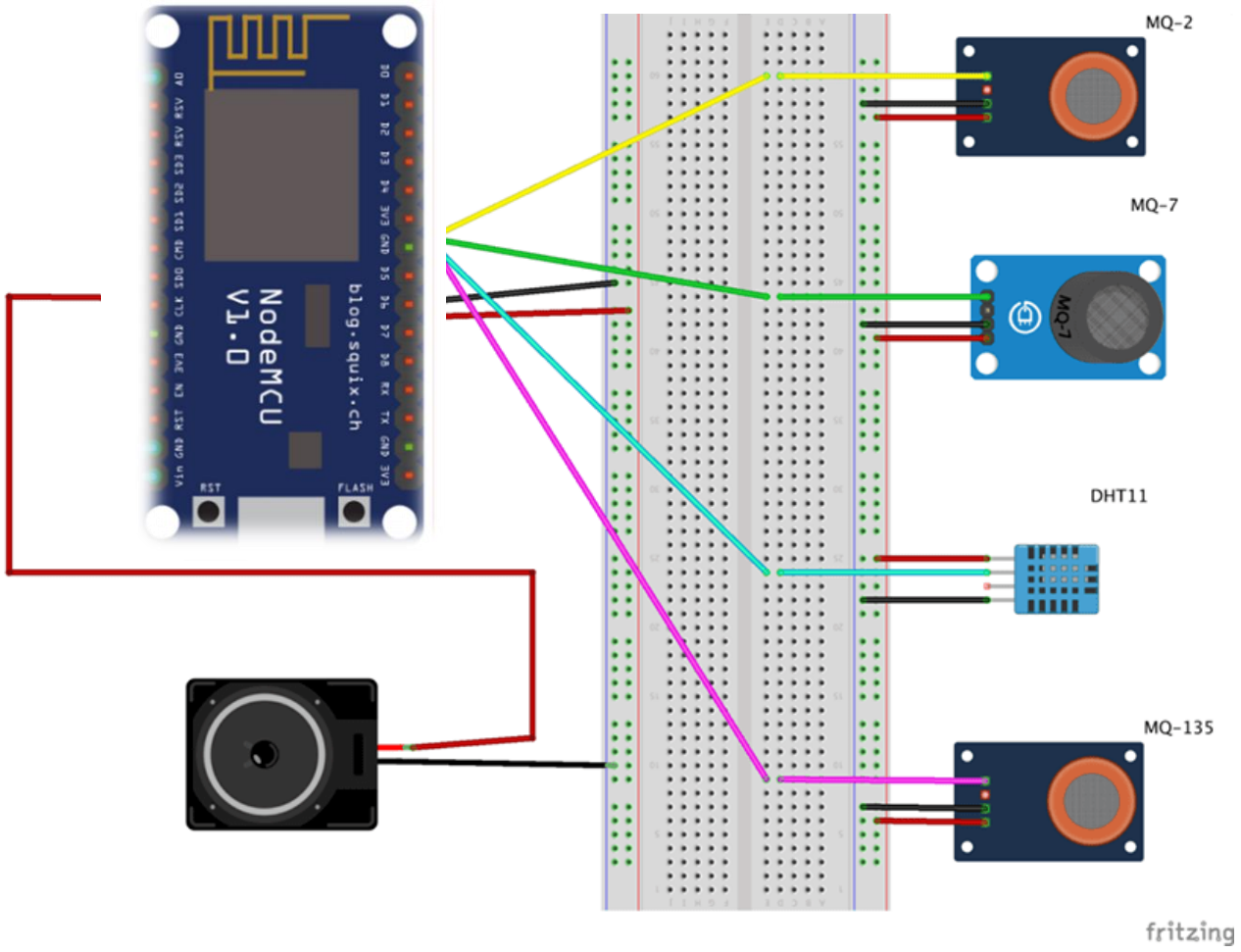
It is a utility system on chip (SoC), which is built in with TCP/IP protocol, since this arbitrary microcontroller can be connected to any Wi-Fi network. It has the power to supervise any other device and can easily be connected to a Wi-Fi network with other systems. It has a huge amount of on-chip variation which is good for the smallest external circuit. The module is an extremely cost effective board.

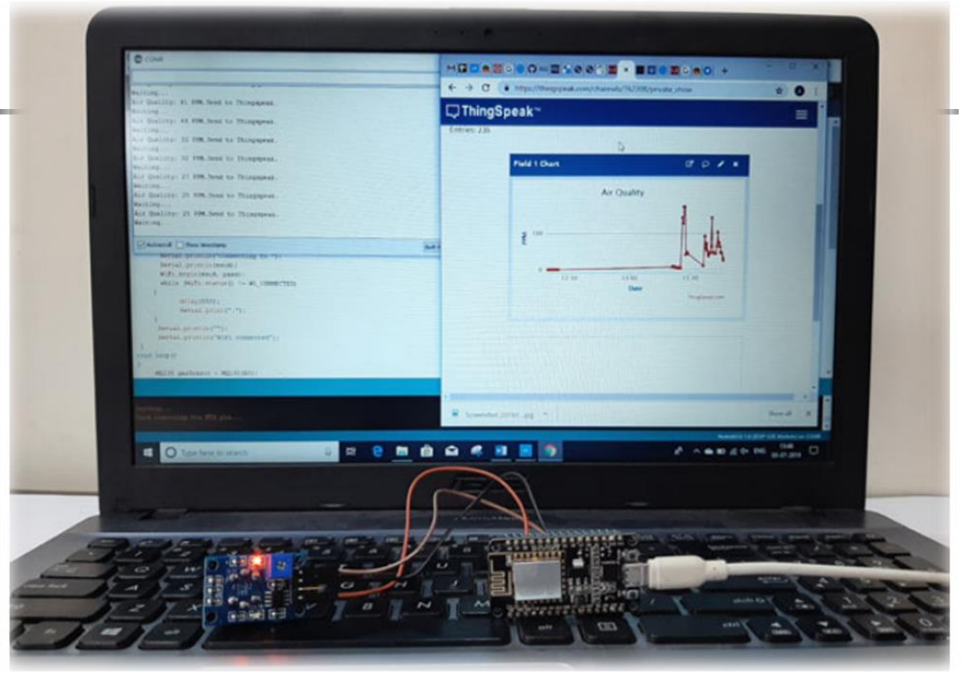
Project 5:Component

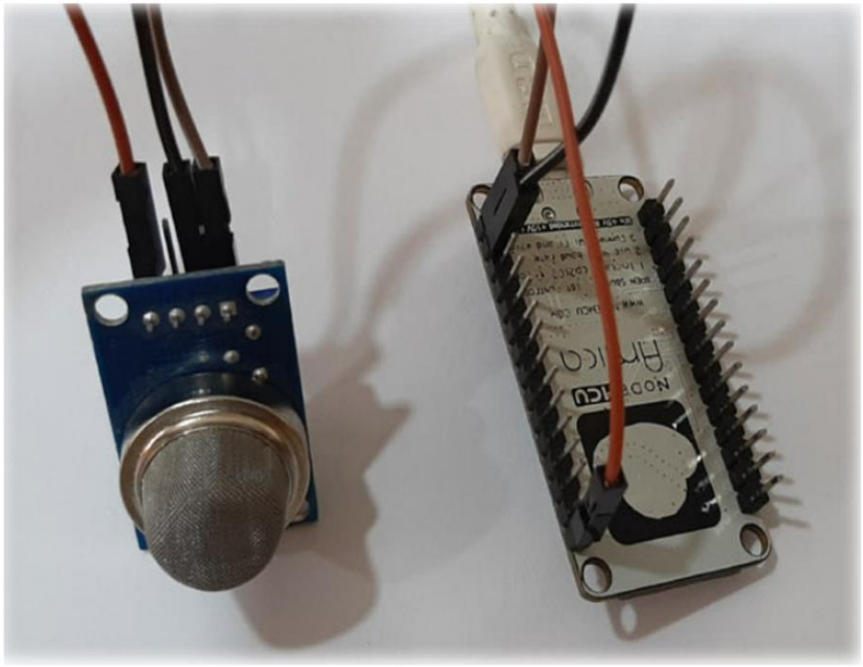


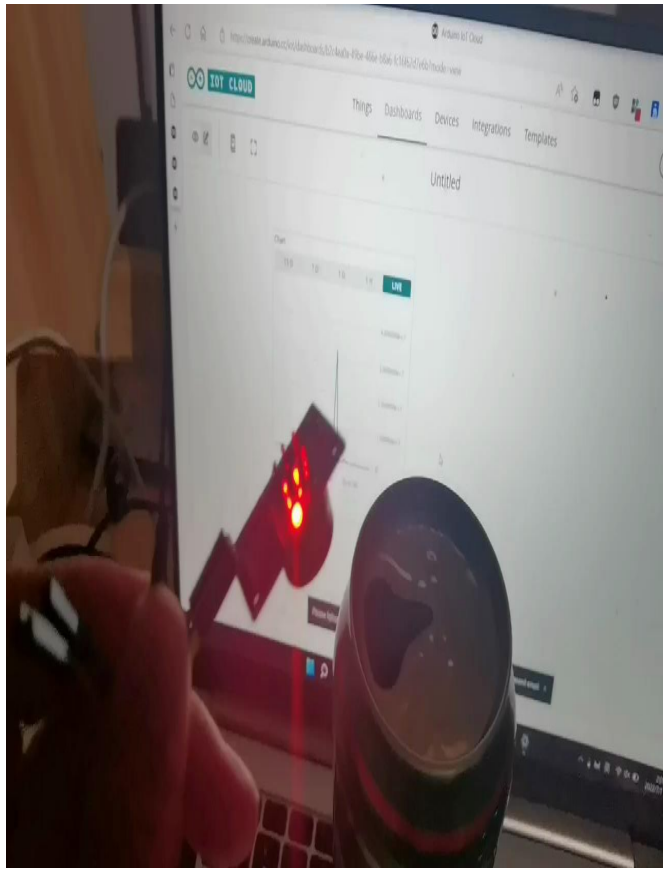


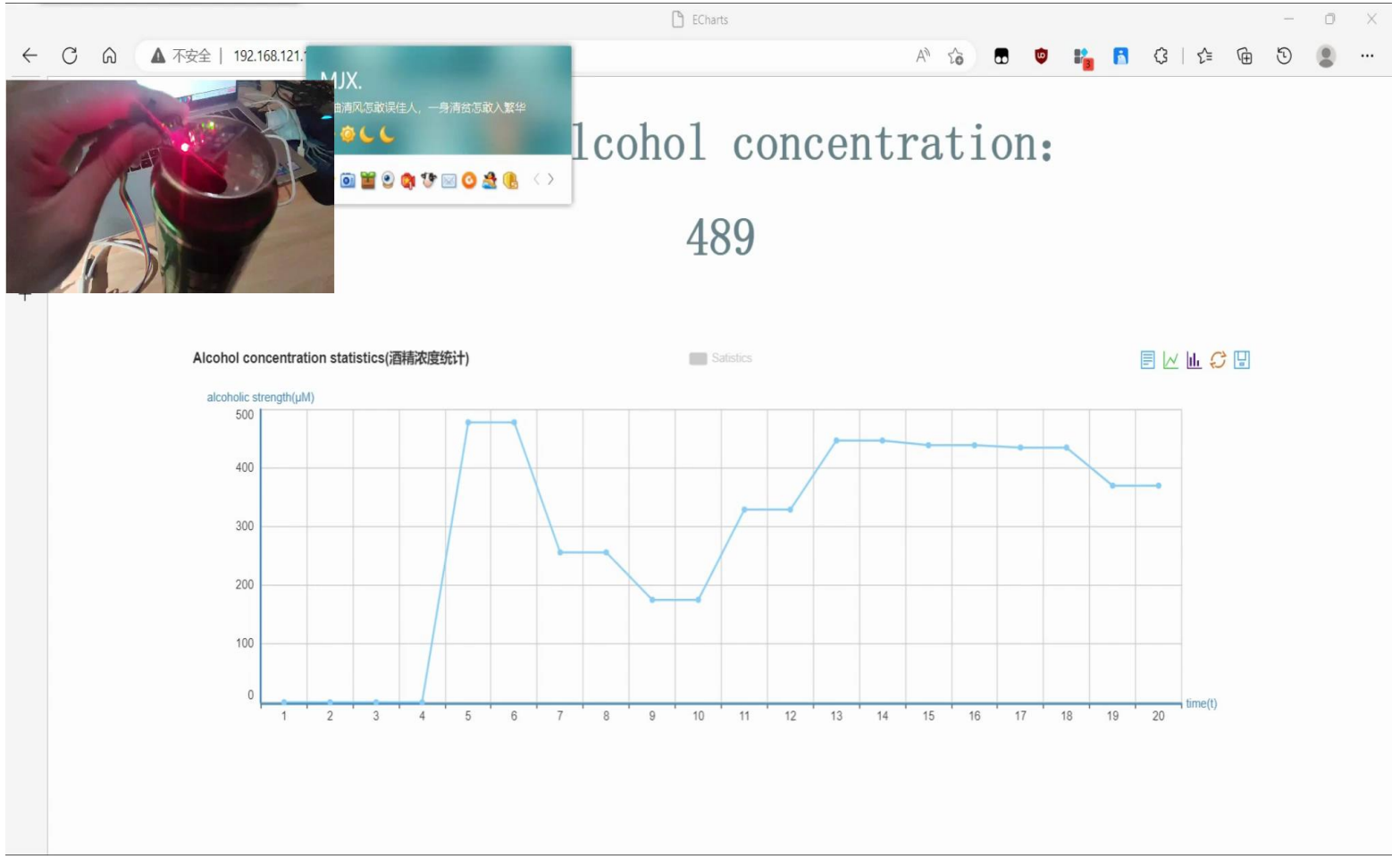
Project 5:Component



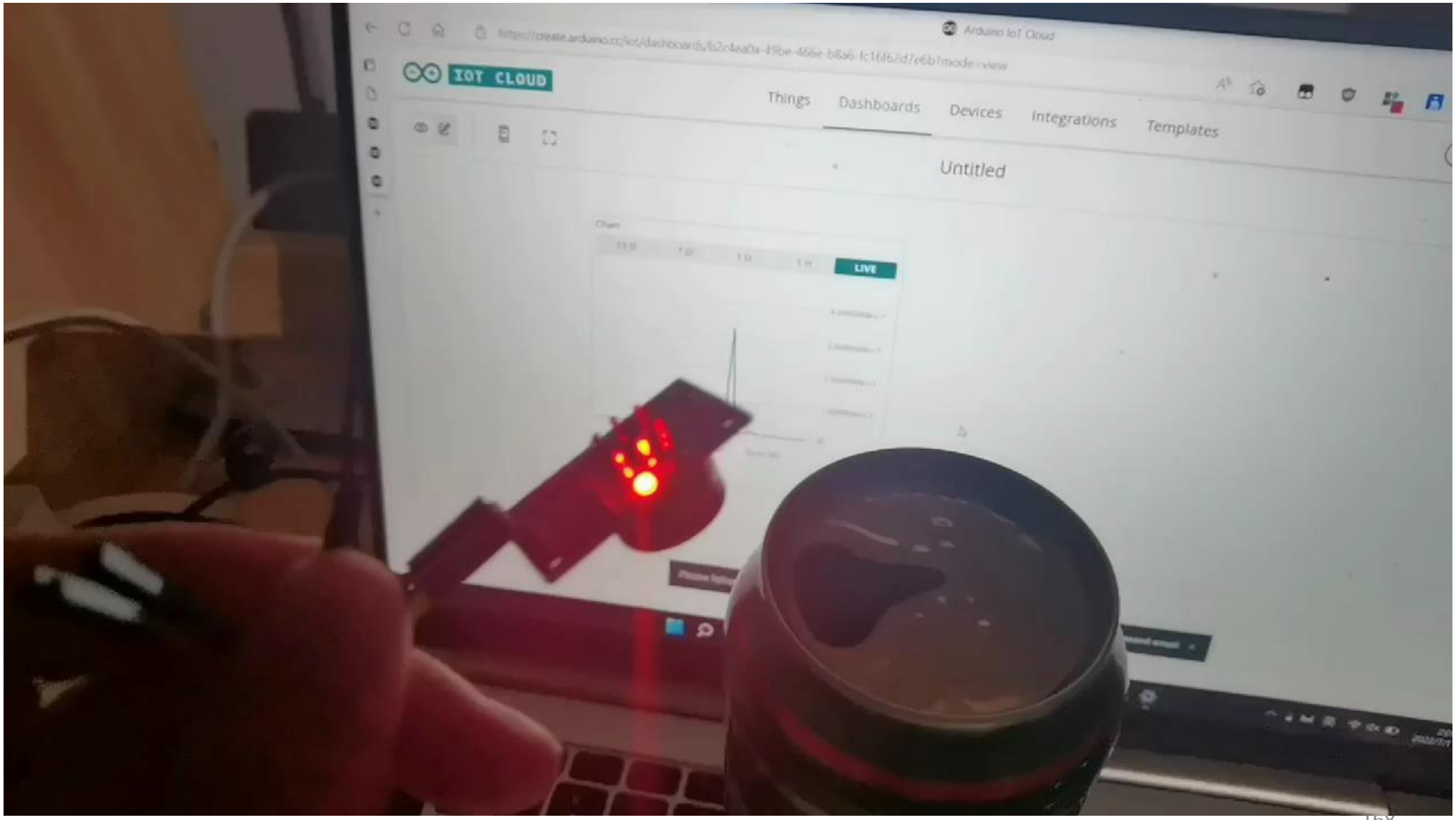





IOT CLOUD
Things
Dashboards
Devices
Integrations
Templates
Untitled


ECharts
不安全 | 192.168.121.
MJX.
lcohol concentration:
489
Alcohol concentration statistics(酒精浓度统计)
Satistics
alcoholic strength(μM)
500
400
300
200
100
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
time(t)

Project 4: Demo Arduino IOT Cloud



Project 4:

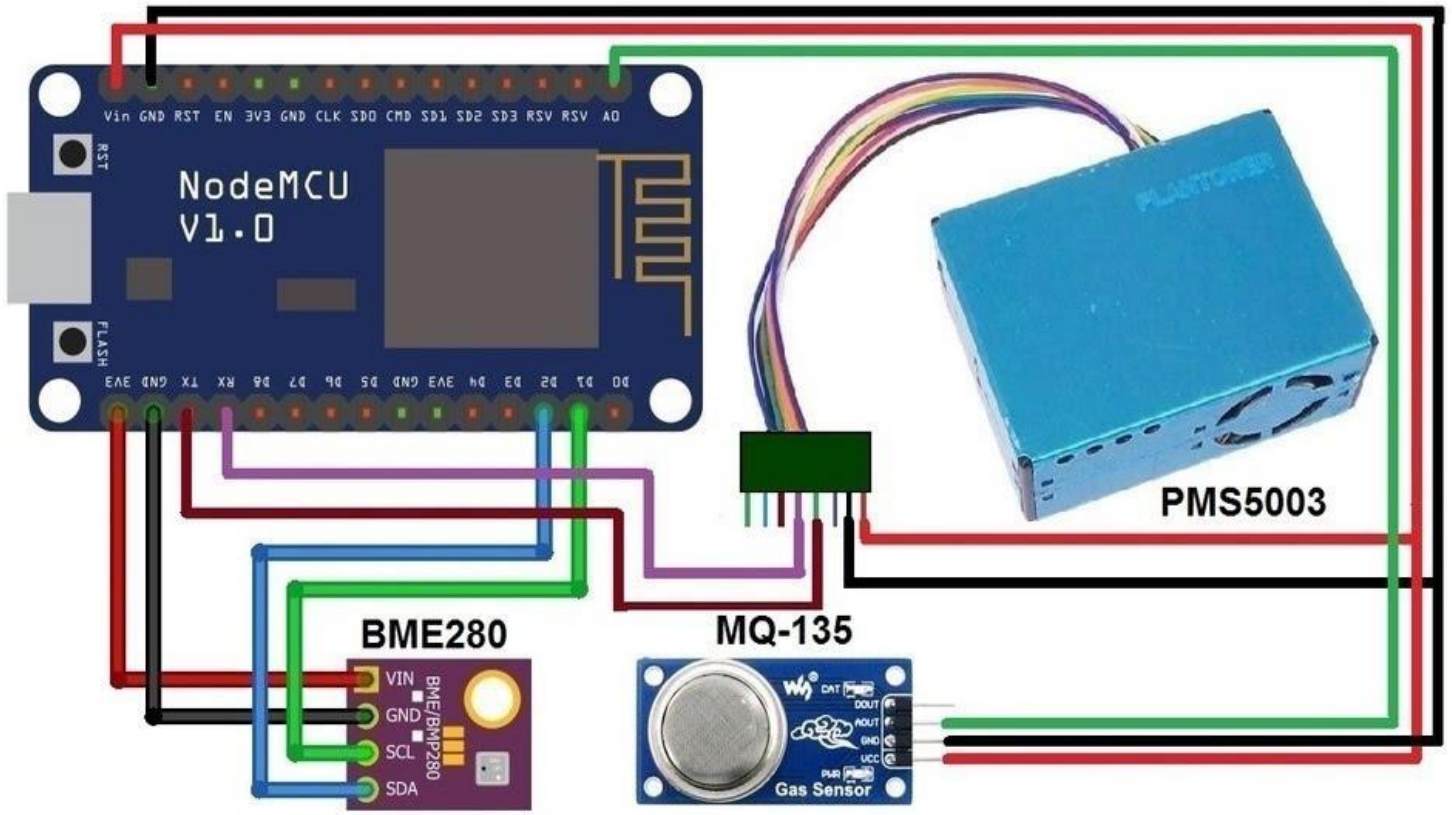

question
answer
question
question
question
question
question

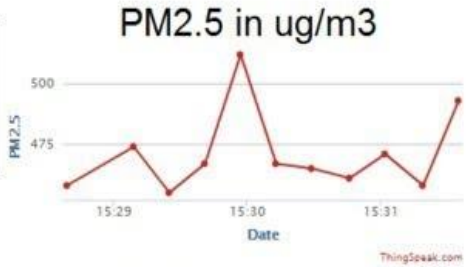

ANSWERS

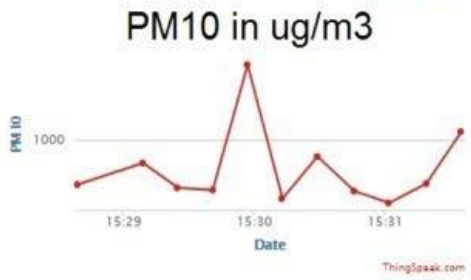
Brain Map

- Alcohol concentration test
  - Alcohol sensor SQ-3 catch
  - A0 Upload data
    - Data collected by correlation function
    - Upload from file flash to web page
- Realize message sharing through LAN
  - Implementation method
    - Establish network service object
    - Enable serial communication
    - Compile related functions

IoT Based Air Pollution Monitoring

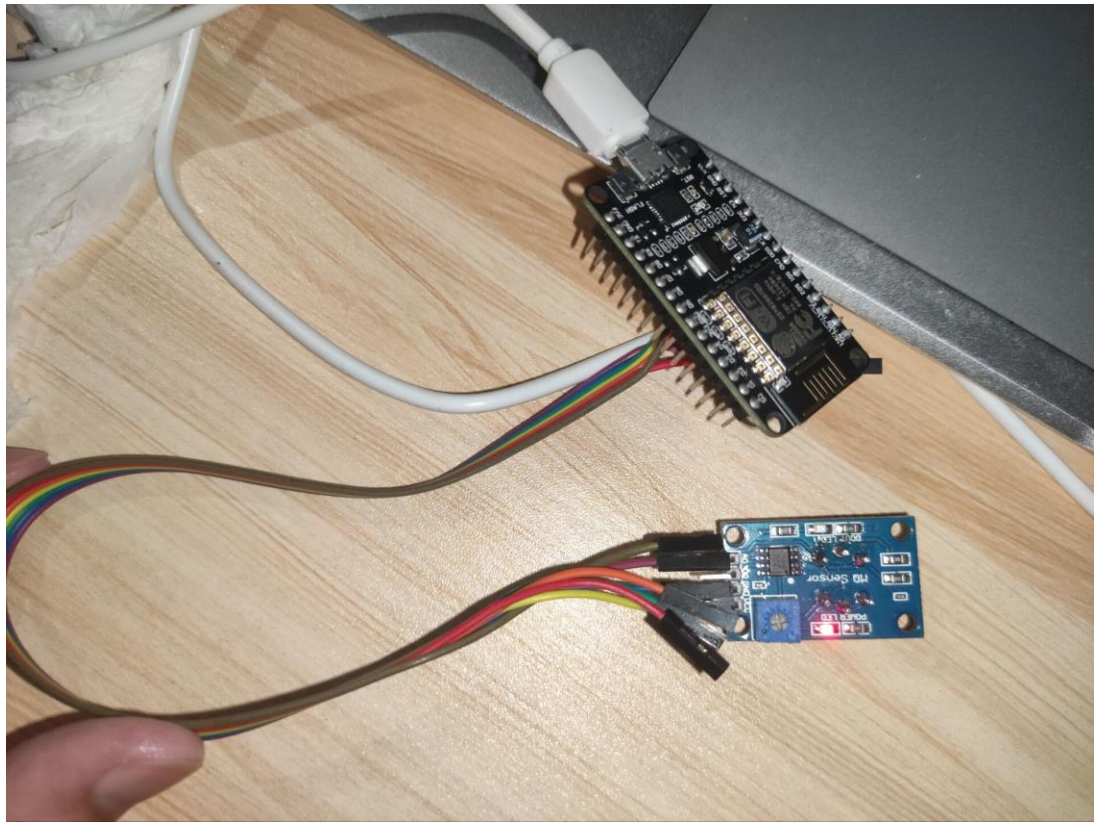


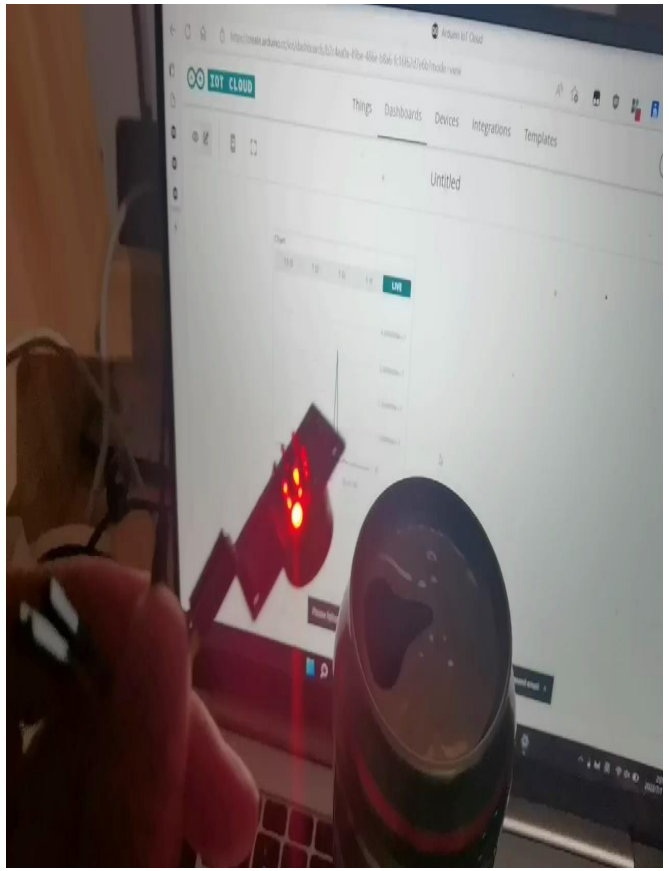


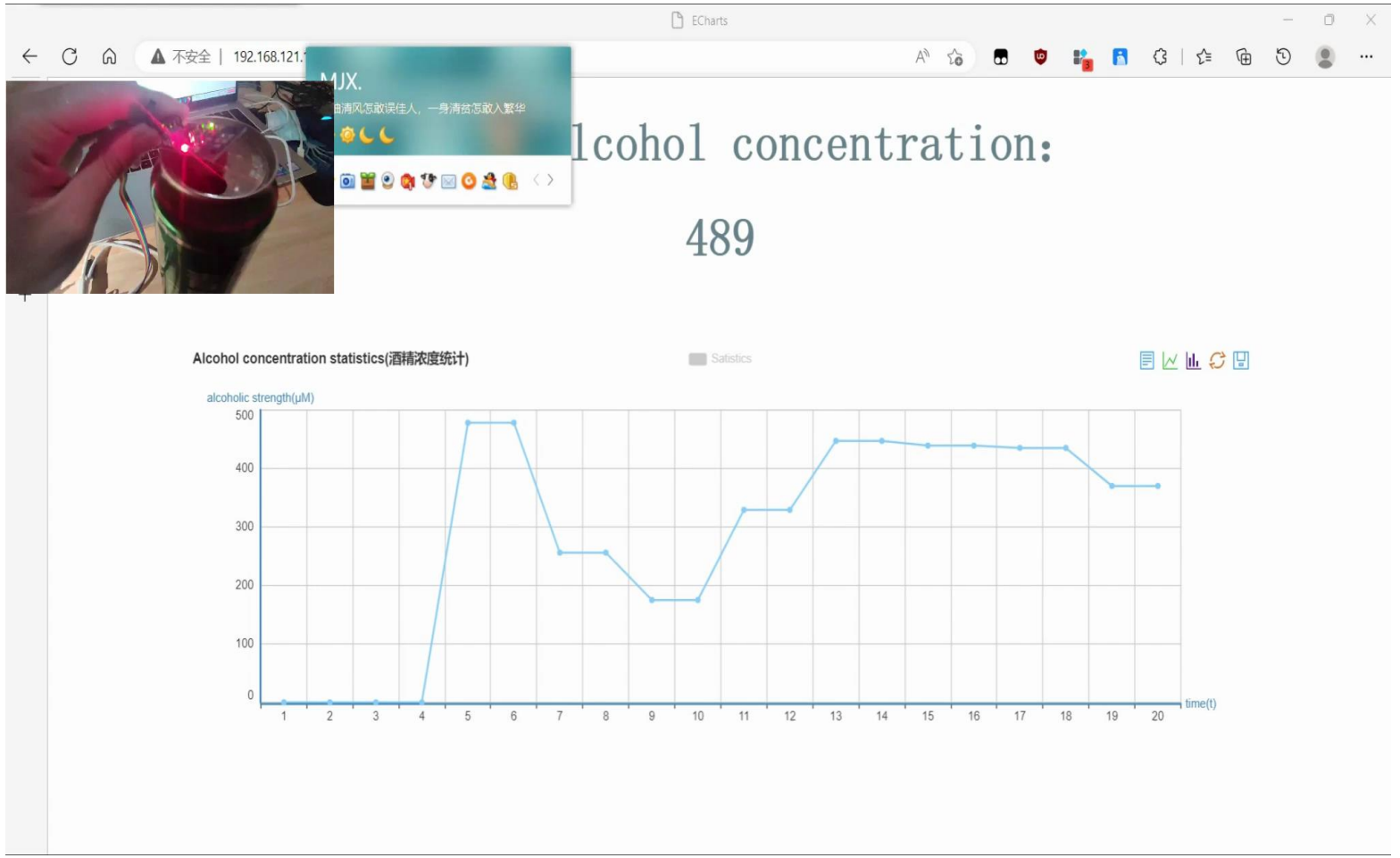


Hardware connection diagram of the project

- The Vcc pin of MQ-3 sensor relates to Vin pin of NodeMCU, and GND pin relates to NodeMCU's GND pin. While the A0 pin relates to A0 pin of NodeMCU as shown in the circuit diagram above. The complete set-up will be powered by the micro-usb port of Node-MCU through a USB cable.




IOT CLOUD
Things
Dashboards
Devices
Integrations
Templates
Untitled
LIVE


ECharts
不安全 | 192.168.121.
MJX.
lcohol concentration:
489
Alcohol concentration statistics(酒精浓度统计)
Satistics
alcoholic strength(μM)
500
400
300
200
100
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
time(t)

Project5:

Code Section

WiFi build

```
ESP8266WiFiMulti wifiMulti;                    // Create an esp8266wifimulti object whose name is 'wifiMulti'

ESP8266WebServer esp8266_server(80);      // Create a web server object that is used to respond to HTTP requests. Listening port (80)

MQ135 gasSensor = MQ135(A0);

void setup(void) {

  Serial.begin(9600);                              // Start serial communication

wifiMulti.addAP(“2451070895的Redmi K30 Pro”, “111111111”);          //Enter a series of WiFi IDs and passwords you need to connect here

  Serial.println(“Connecting ...”);                  // Then try to connect using the password stored here.

}
```

WiFi build

```
void setup(void) {
  Serial.begin(9600);                          // Start serial communication

  wifiMulti.addAP(“2451070895的Redmi K30 Pro”, “11111111”);       // Enter a series of WiFi IDs and passwords you need to connect here
  Serial.println("Connecting ...");                    // Then try to connect using the password  stored here.
  int i = 0;
  while (wifiMulti.run() != WL_CONNECTED) {
 // WiFi multi Run () is the key. Via WiFi multi
  Run(), nodemcu will be in the current Search the WiFi stored by the addap function in the environment.
delay(1000); Serial.print(i++); Serial.print(‘ ’);      // If multiple stored WiFi is found, nodemcu will connect the WiFi signal with the strongest signal.
 }                                                      // Once WiFi is successfully connected, WiFi multi Run() will return "wl_connected". This is also the
condition for the while loop to judge whether to jump out of the loop.
  if(SPIFFS.begin()){                                  //Boot flash file system
    Serial.println("SPIFFS Started.");
  } else {
    Serial.println("SPIFFS Failed to Start.");
  }
```

Web page processing

```
void handleData()
{
  esp8266_server.send(200,"textain",(String)air_quality);
}
void handleUserRequest() {                    //Get user request resource
  String reqResource = esp8266_server.uri();
  Serial.print("reqResource: ");
  Serial.println(reqResource);        //Handle user requested resources   through
the handlefileread function
  bool fileReadOK = handleFileRead(reqResource);

                              //If the resource accessed by the user cannot be found in
SPIFs, reply 404 (not found)
  if (!fileReadOK){
    esp8266_server.send(404, "textain", "404 Not Found");
  }
}
```

Web page making

```
<!DOCTYPE html>
<head>
    <meta charset="utf-8">
    <title>ECharts</title>
</head>
<body>
    <h1>Current alcohol concentration：</h1><h1 id="x">x</h1>
    <br><br>
    <!-- 为ECharts准备一个具备大小（宽高）的Dom -->
    <div id="main" style="width:80%;height:400px;margin: 0 auto;background: #cccccc"></div>
</body>

<style>
    h1 {
        position: relative;
        text-align: center;
        color:          #68838B;
        font-size:  50px;
        font-family: "Cormorant Garamond", serif;
    }
    </style>
<!-- ECharts单文件引入 -->
<script src="http://echarts.baidu.com/build/dist/echarts.js"></script>
<script type="text/javascript">
    var arrX=[], arrY=[];
    var x=0;
    var i = 1;
    var sh = setInterval(function () {
        if(i>20){
            clearInterval(sh);
            myChart.setOption(option);
        }
```

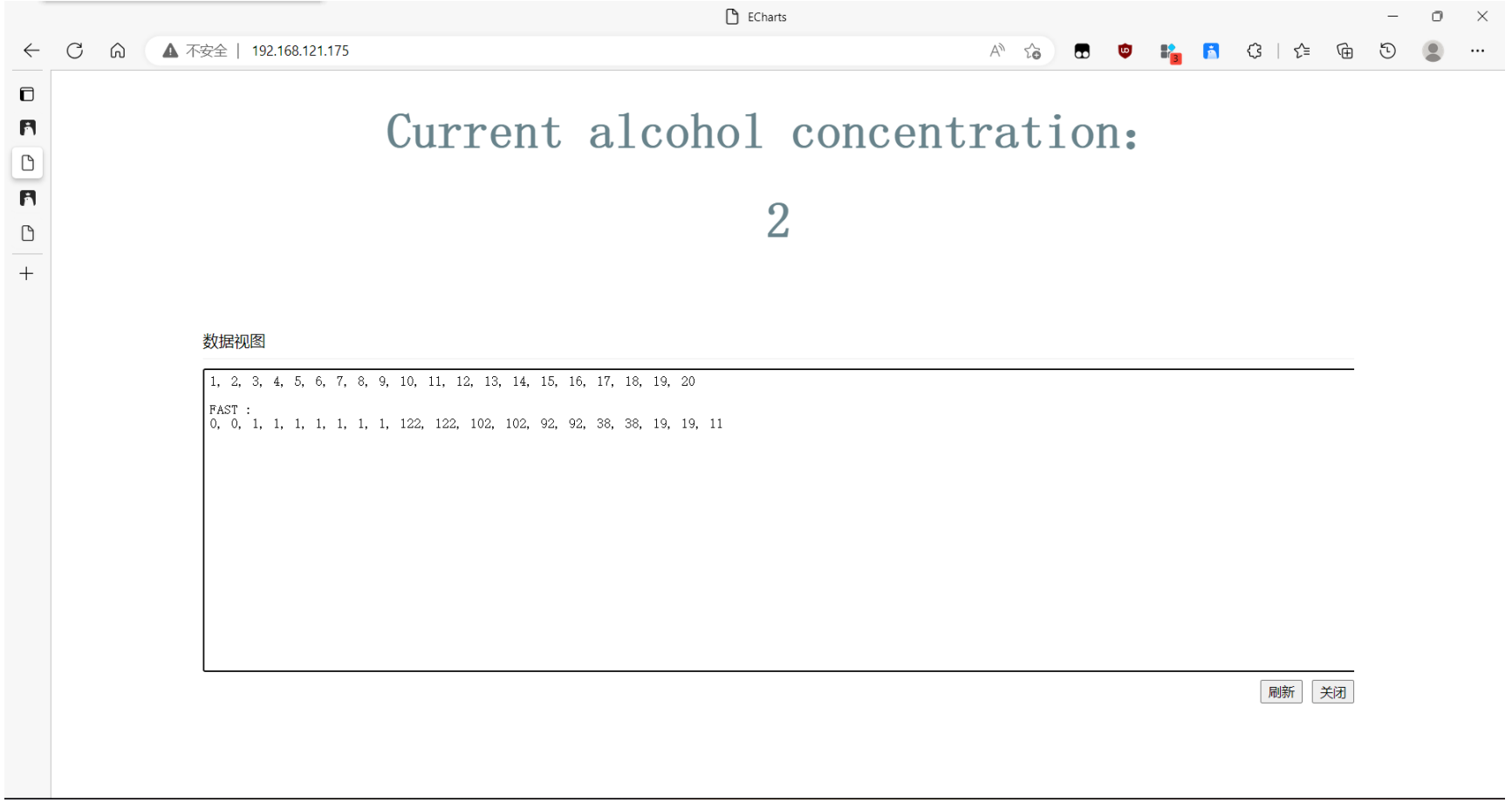
```
arrX.push(i)
        var j = parseInt(getAirQ())
        arrY.push(j)
        i = i + 1
        //Path configuration
        require.config({
            paths: {
                echarts: 'http://echarts.baidu.com/build/dist'
            }
        });
        creatCheart();
    }, 1000);
    function creatCheart(){
        // use
        require(
            [
                'echarts',
                'echarts/chart/bar', //Use the histogram to load the bar module, and          load it on demand
                'echarts/chart/line'
            ],
            function (ec) {
            //Initialize the ecarts chart based on the prepared DOM
                var myChart = ec.init(document.getElementById('main'));

                var option = {//Description of specific details
                    title: {
                        text: 'Alcohol concentration statistics(酒精浓度统计)',
                        textStyle: { //主标题文本样式{"fontSize": 18,"fontWeight": "bolder","color": "#333"}
                            fontSize: 14,
                            fontStyle: 'normal',
                            fontWeight: 'bold',
                        },
                    },
```

Web page making

```
tooltip: {
        trigger: 'axis'
    },
    legend: {
        data: ['Satistics']
    },
    toolbox: { // you can select specific data, histogram, line chart, restore, and save the image
        show: true,
        feature: {
            dataView: {
                show: true,
                readOnly: false
            },
            magicType: {
                show: true,
                type: ['line', 'bar'] //Line chart and histogram are optional                    },
            restore: {
                show: true  //Restore default
            },
            saveAsImage: {
                show: true //Functions stored as pictures
            }
        }
    }, calculable: true,
    //Name:'time'on the horizontal axis
    xAxis: [{
        type: 'category',
        data: arrX,
        name: 'time(t)',
        position: 'left'
    }],
    //name: "alcoholic strength( μ M) "On the ordinate
```

```
    yAxis: [{
            type: 'value',
            name: 'alcoholic strength(μM)',
            position: 'left'
        }],
    series: [{
            name: 'FAST',
            type: 'line',// bar
            data: arrY,
            color: '#CC0066'
        }]
    };

    //Load data for the echorts object
    myChart.setOption(option);
  }
 );
}
function getAirQ() {
  var xhttp = new XMLHttpRequest();
  xhttp.onreadystatechange = function () {
    if (this.readyState == 4 && this.status == 200) {
      document.getElementById("x").innerHTML =
        this.responseText;
      x = this.responseText
    }
  };
  xhttp.open("GET", "data", true);
  xhttp.send();
  return x;
}
</script>
```

We get three types of data result graphs



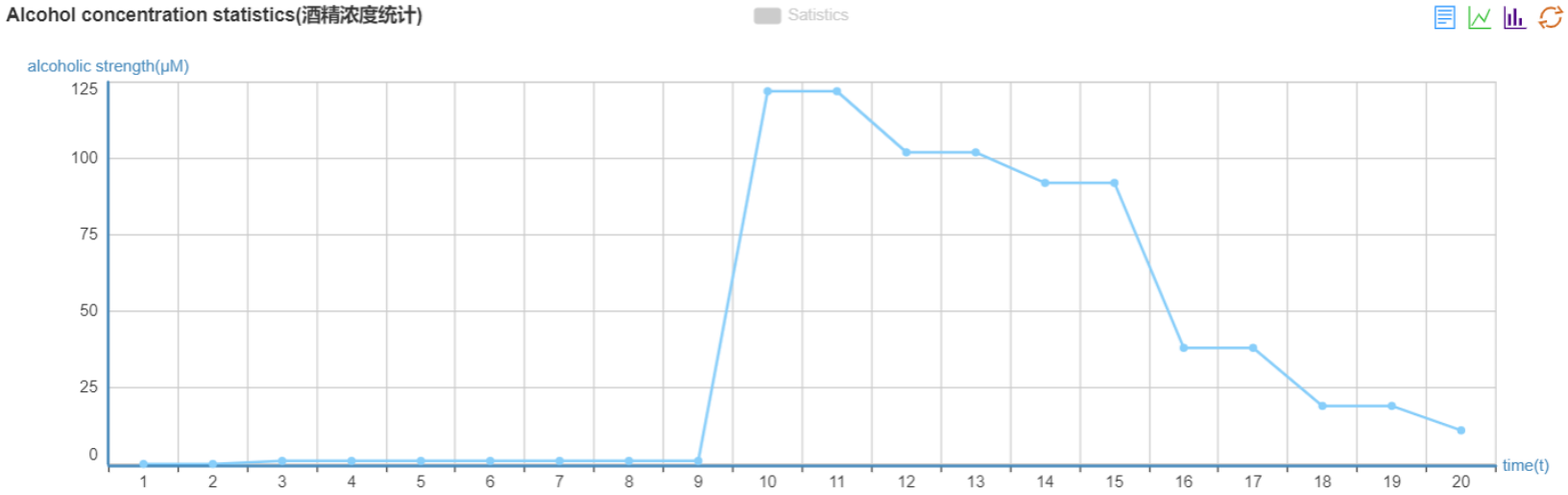
We get three types of data result graphs

ECharts

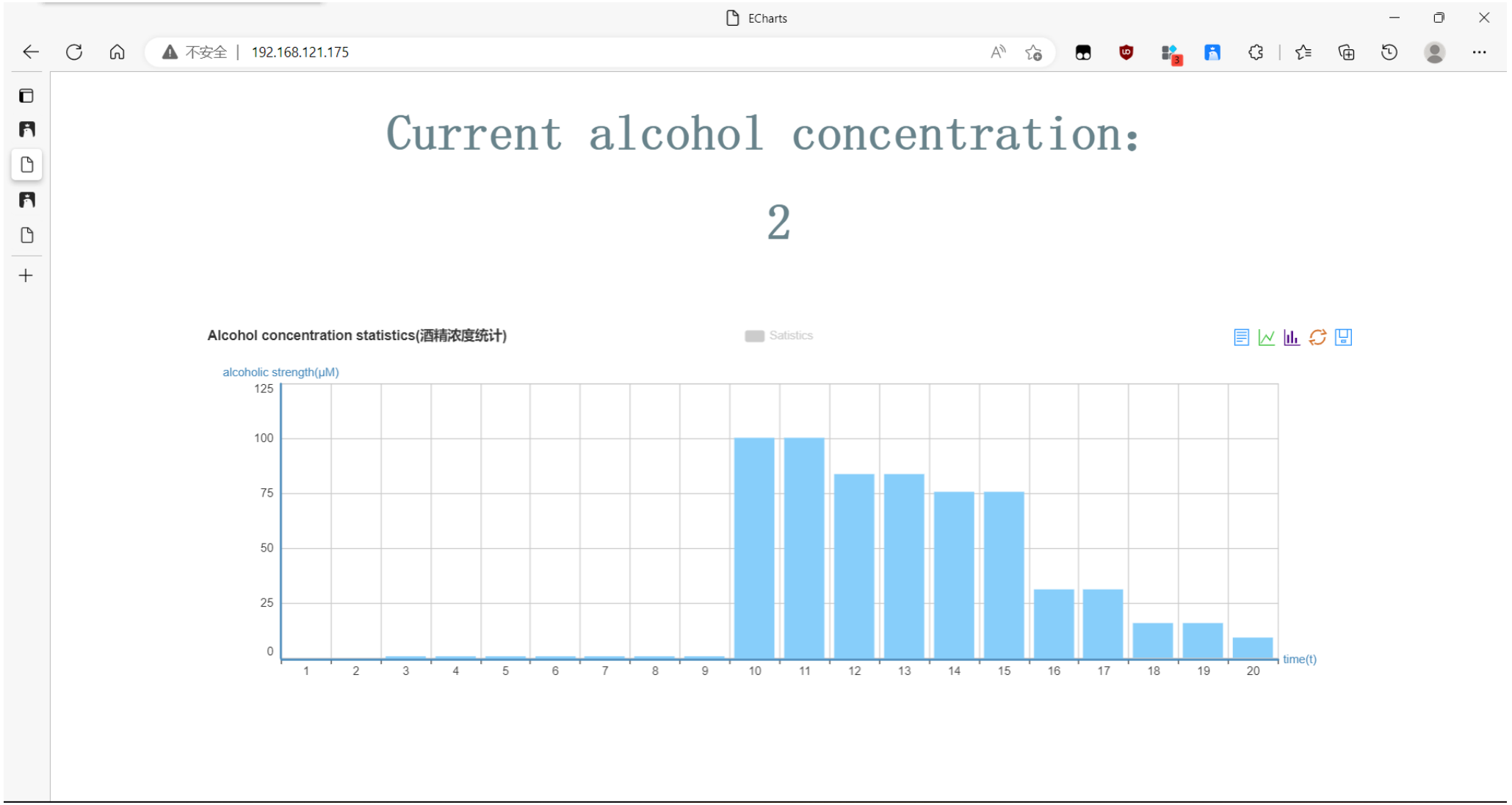
不安全 | 192.168.121.175

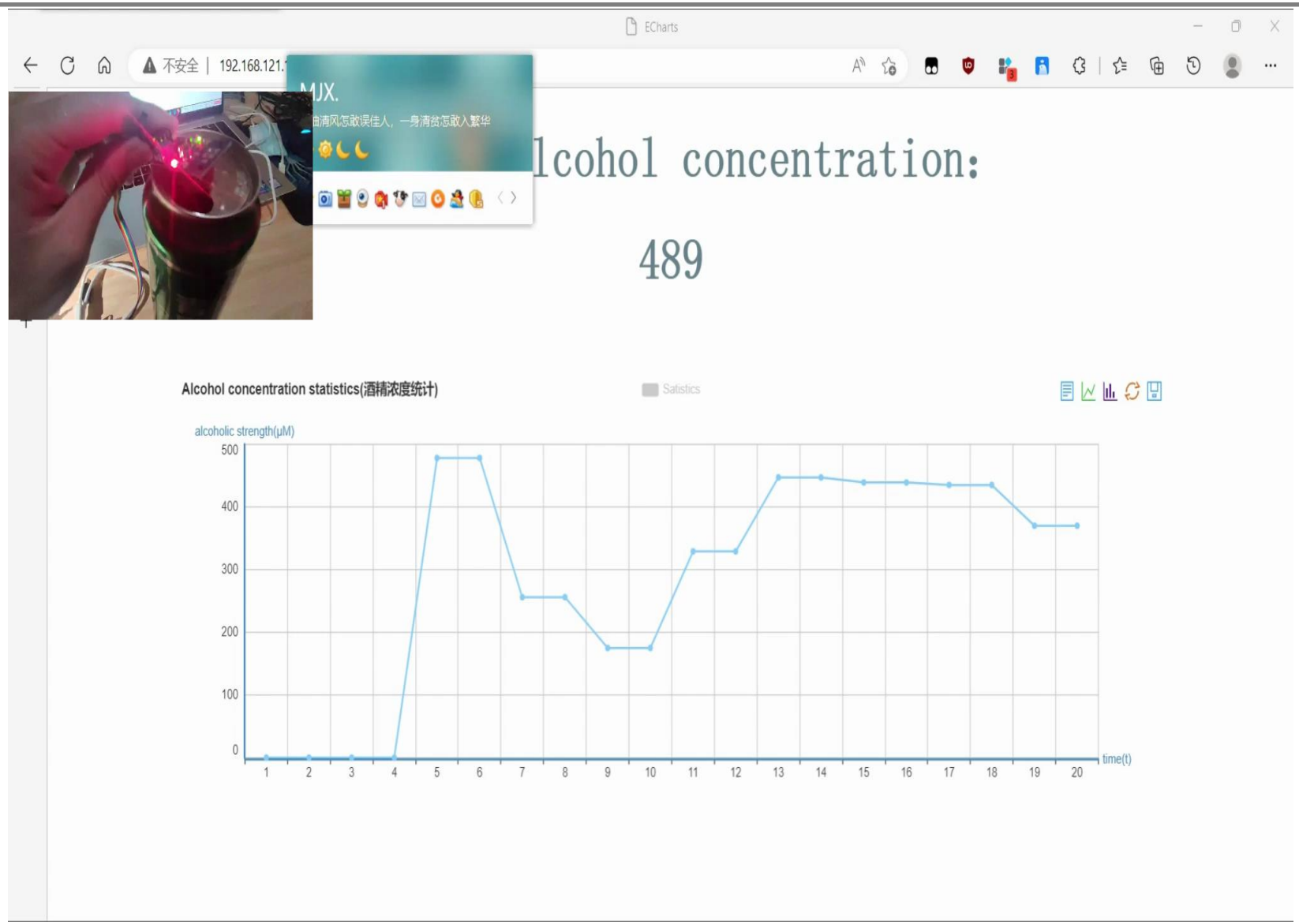
Current alcohol concentration:

2



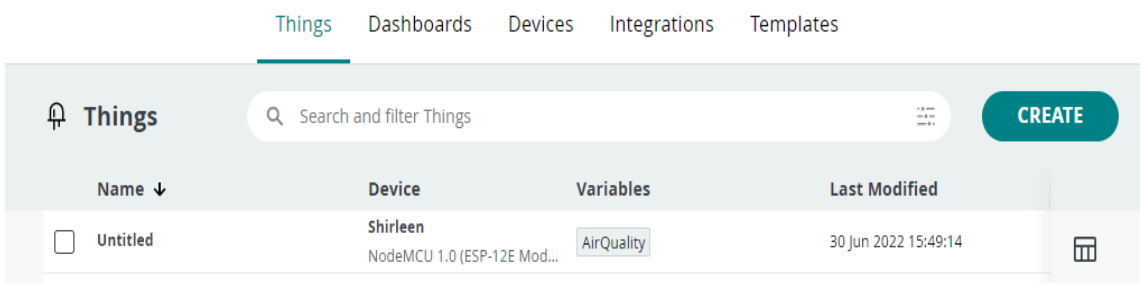
We get three types of data result graphs



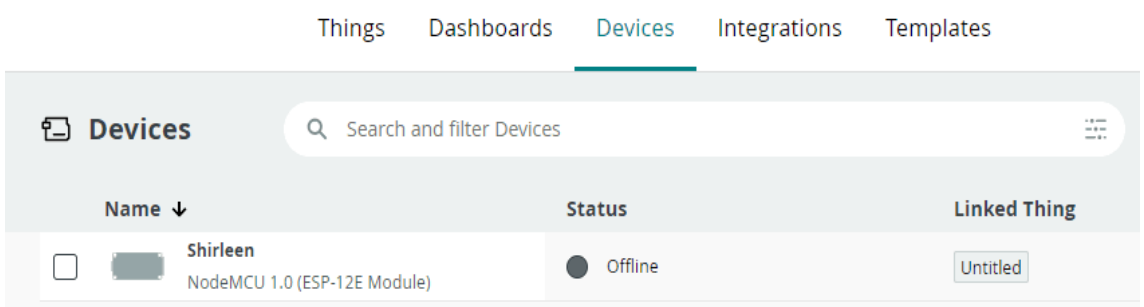

ECharts
不安全 | 192.168.121.
MJX.
lcohol concentration:
489
Alcohol concentration statistics(酒精浓度统计)
Satistics
alcoholic strength(μM)
500
400
300
200
100
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
time(t)

Arduino IOT

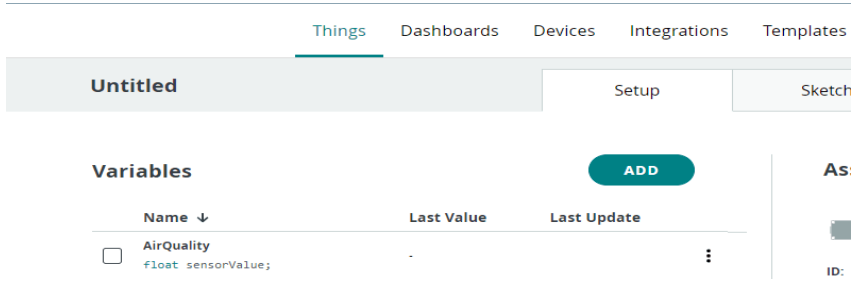
STEP 1 : Create Things



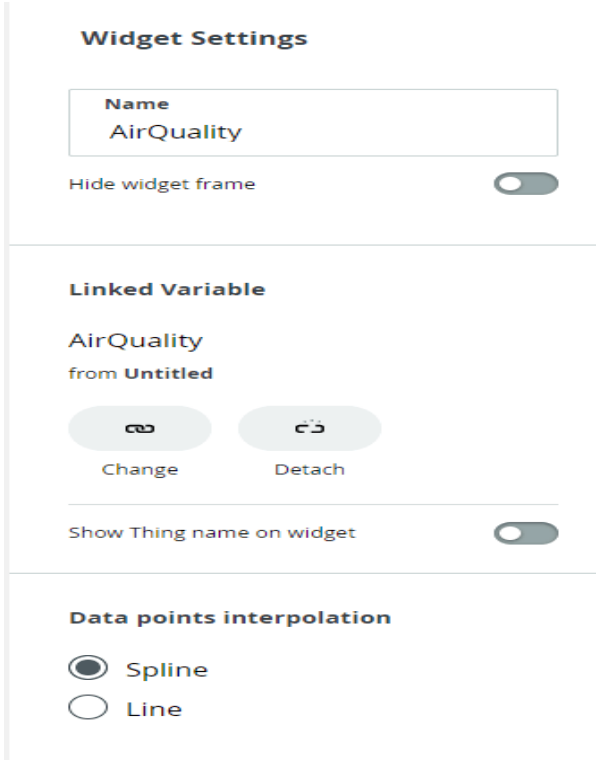
Create Associated Device



Add variable

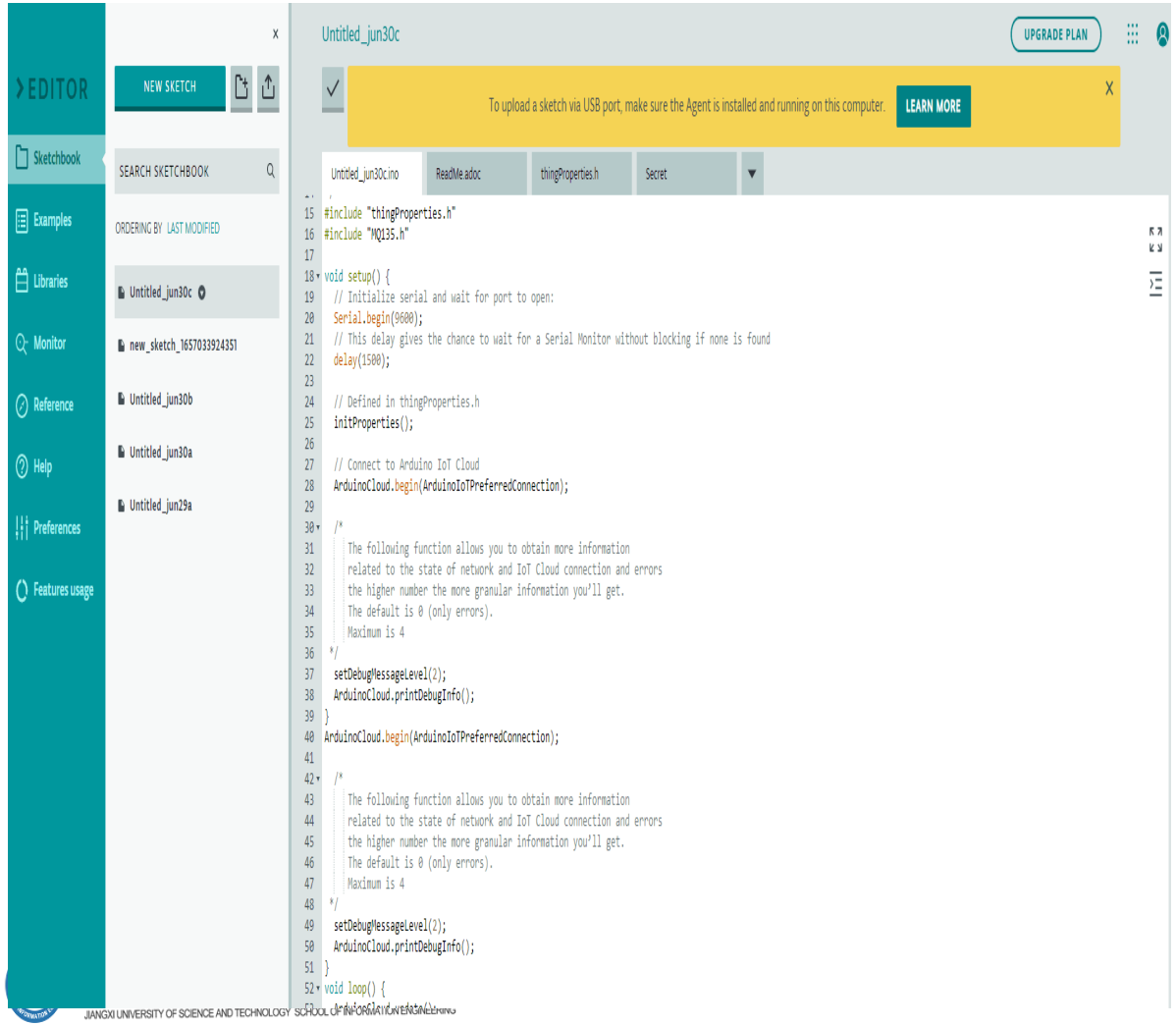


STEP 2:Establish a dashboard association with a variable

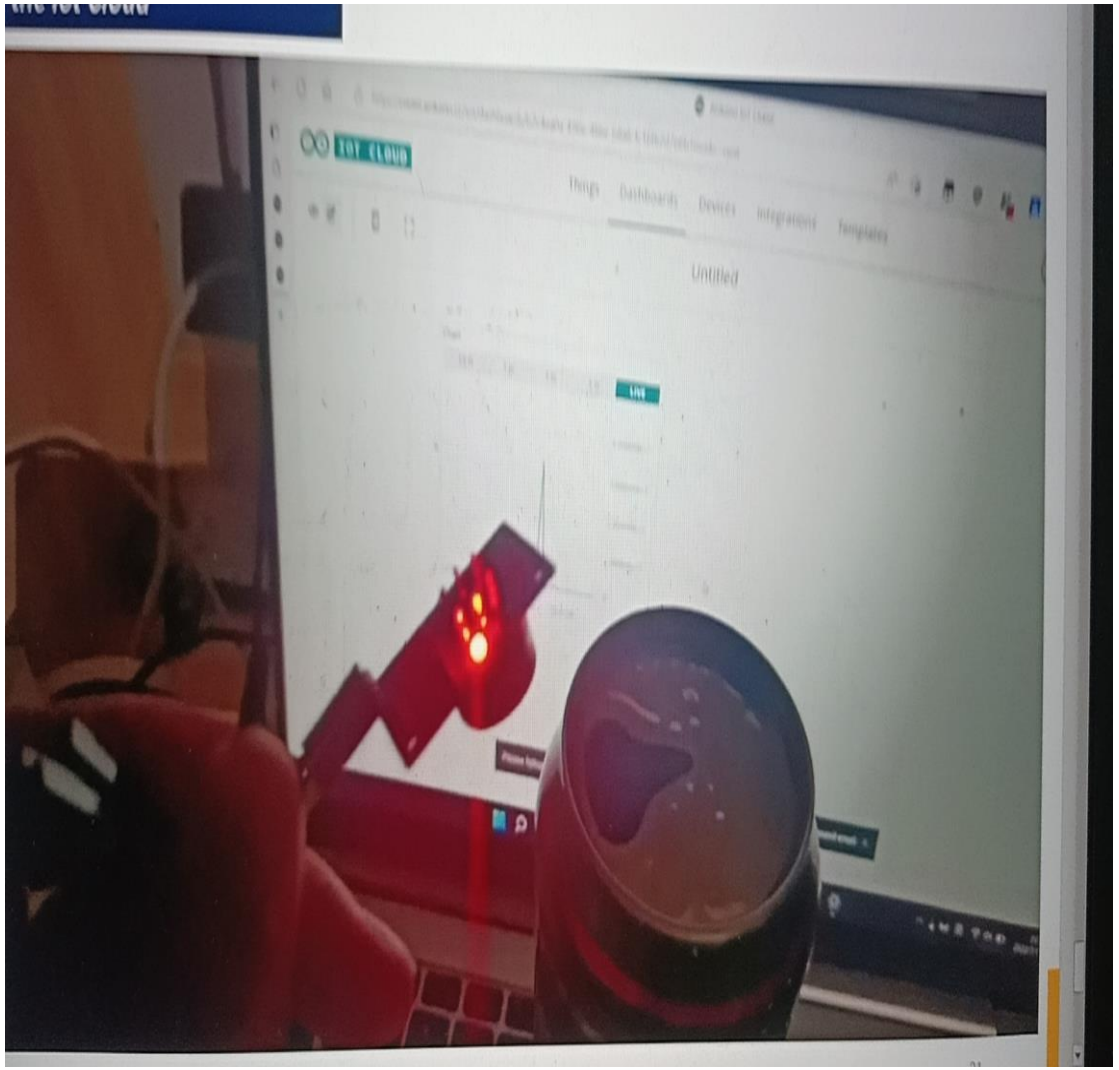


Arduino IOT

STEP 3: Modify the code



STEP 4:Start running



SECOND:The codes we use in the IOT Clou

```
/*
  Sketch generated by the Arduino IoT Cloud Thing "Untitled 2"
  https://create.arduino.cc/cloud/things/a2621a1a-51c2-479b-b440-8140d252da94

  Arduino IoT Cloud Variables description

  The following variables are automatically generated and updated when changes are made to the Thing

  int hahaha;

  Variables which are marked as READ/WRITE in the Cloud Thing will also have functions
  which are called when their values are changed from the Dashboard.
  These functions are generated with the Thing and added at the end of this sketch.
*/
```

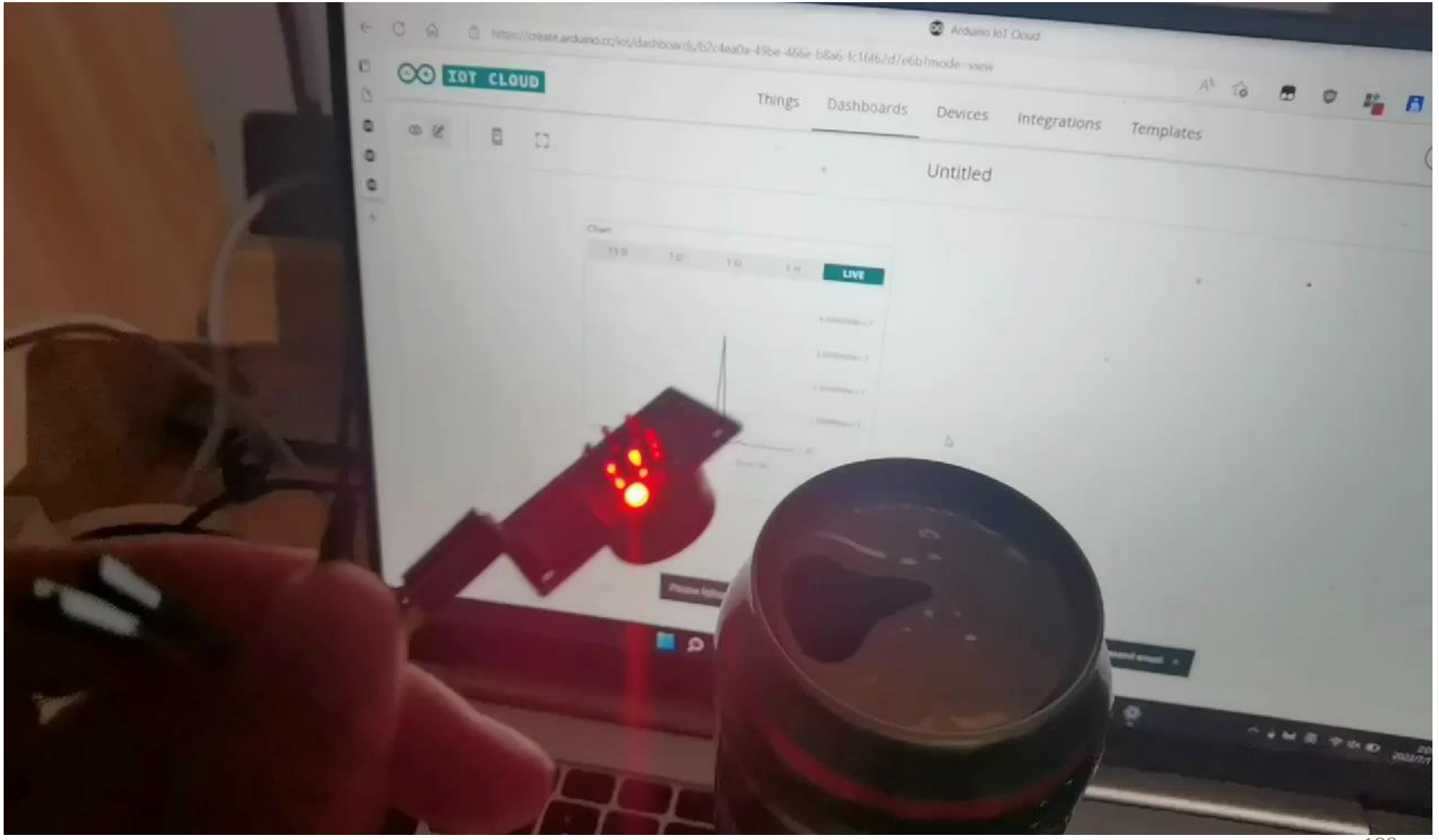
```
#include "thingProperties.h"
#include "MQ135.h"

void setup() {
  // Initialize serial and wait for port to open:
  Serial.begin(9600);
  // This delay gives the chance to wait for a Serial Monitor without blocking if none is found
  delay(1500);

  // Defined in thingProperties.h
  initProperties();

  // Connect to Arduino IoT Cloud
  ArduinoCloud.begin(ArduinoIoTPreferredConnection);

  /*
     The following function allows you to obtain more information
     related to the state of network and IoT Cloud connection and errors
     the higher number the more granular information you'll get.
     The default is 0 (only errors).
     Maximum is 4
  */
  setDebugMessageLevel(2);
  ArduinoCloud.printDebugInfo();
}
```

SECOND The codes we use in the IOT Cloud

```
 ArduinoCloud.begin(ArduinoIoTPreferredConnection);
    /*
     The following function allows you to obtain more information
     related to the state of network and IoT Cloud connection and errors
     the higher number the more granular information you'll get.
     The default is 0 (only errors).
     Maximum is 4
 */
  setDebugMessageLevel(2);
  ArduinoCloud.printDebugInfo();
}
```

```
void loop() {
  ArduinoCloud.update();

  onHahahaChange();

}
/*
  Since Hahaha is READ_WRITE variable, onHahahaChange() is
  executed every time a new value is received from IoT Cloud.
*/
void onHahahaChange()  {
   MQ135 gasSensor = MQ135(A0);
 hahaha = gasSensor.getPPM();
  delay(1000);
  Serial.println(hahaha);
}
```

Next up is a video running in the iot Cloud



"Good scientists are perpetual adolescents. They never grow up."

PETER DOHERTY
Nobel Prize in Physiology or Medicine 1996