

Bộ Giáo Dục Và Đào Tạo

Trường Đại Học Hoa Sen

# Báo Cáo Cuối Kỳ Bộ Môn Thực Hành Công Nghệ Thông Tin (2)

Đề tài:

Chương trình dò nồng độ cồn

Giảng viên hướng dẫn:

Trần Thị Trường Thi

Sinh viên thực hiện:

Trần Gia Nguyên Phong

Mã số sinh viên:

22002575

## Mục Lục

Lời Mở Đầu.....	3
Nội dung.....	4
I. Chuẩn bị (Khái quát).....	4
1. Yêu cầu thiết bị và môi trường.....	4
II. Cài đặt môi trường (1 vài ứng dụng).....	10
1. Nodejs.....	10
2. Postman .....	12
3. Mysql.....	13
4. Arduino ide.....	14
5. Nối link kiện IoT .....	16
Code mẫu .....	17
I. Iot (Arduino code).....	17
II. Back-end.....	23
1. Restful API .....	23
2. Mysql Query .....	26
III. Front-end .....	28
Kết luận.....	29
Nguồn tham khảo.....	30

## Lời Mở Đầu

Em xin cảm ơn giảng viên bộ môn thực hành công nghệ thông tin (cô Trần thị Trường Thi) đã tạo cơ hội cho em tiếp xúc với những công nghệ mới và những trải nghiệm mới.

Bài báo cáo này sẽ có sự sai sót do đó mong cô thông cảm và góp ý để giúp em rút kinh nghiệm và cải thiện cho những lần báo cáo sau.

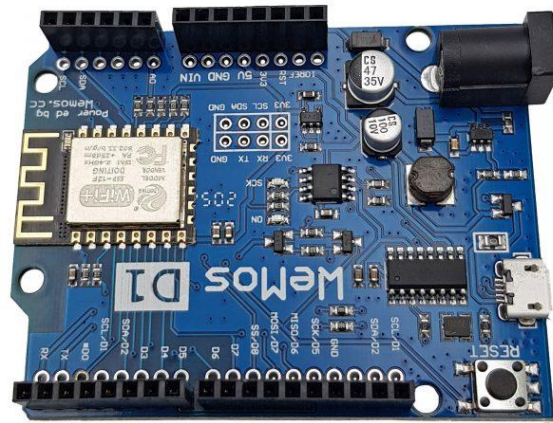
## Nội dung

### I. Chuẩn bị (Khái quát)

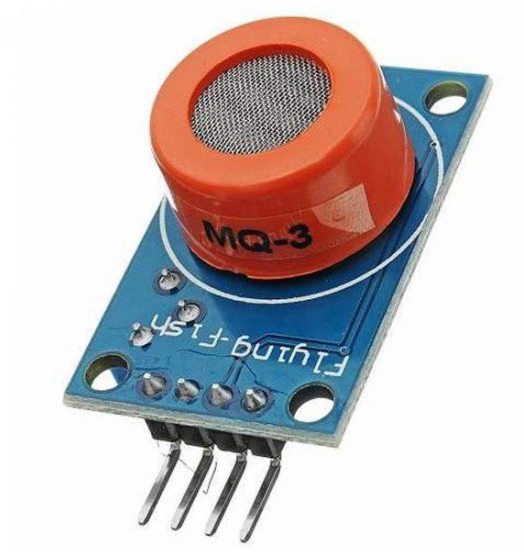
#### 1. Yêu cầu thiết bị và môi trường

##### a. *Thiết bị*

- Wemos D1 R2 (ESP 2866)



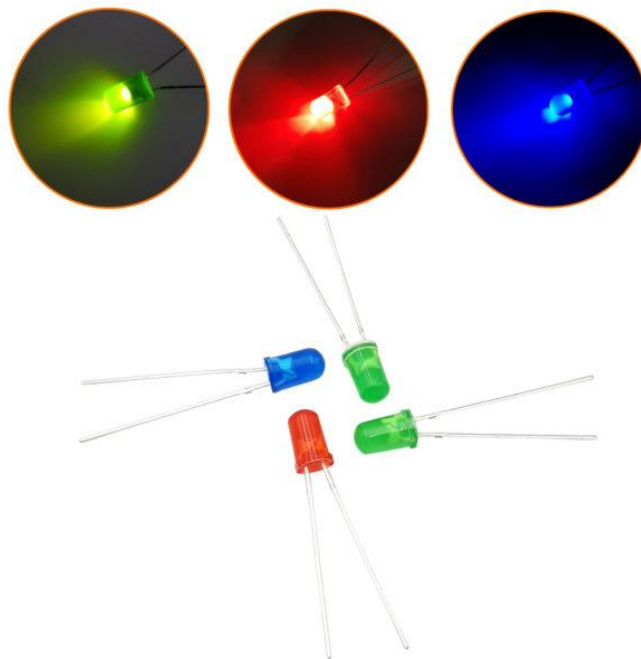
- Cảm biến nồng độ cồn MQ3



- Dây nối (đực cái, cái đực, cái cái)



- Đèn led 5mm



➤ Board test



➤ Còi 5v



- Dây usb to micro-usb



*b. Môi trường*

- Front-end
- Html/css, js



- React js



- Bootstrap 5



- Back-end
  - Nodejs
  - Express.js và những npm's packages (như nodemon, cors, body-parser...)



- Postman



POSTMAN

- Database
  - Mysql (và mysql2 cho npm's package)





- IoT
- Arduino IDE



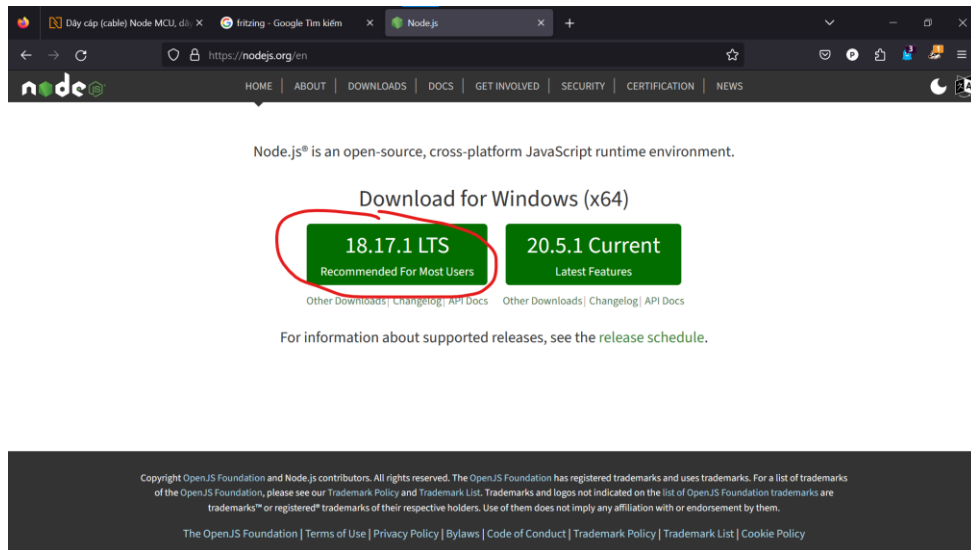
- Fritzing



## II. Cài đặt môi trường (1 vài ứng dụng)

### 1. Nodejs

Link tải: <https://nodejs.org/en> (tải bản LTS)



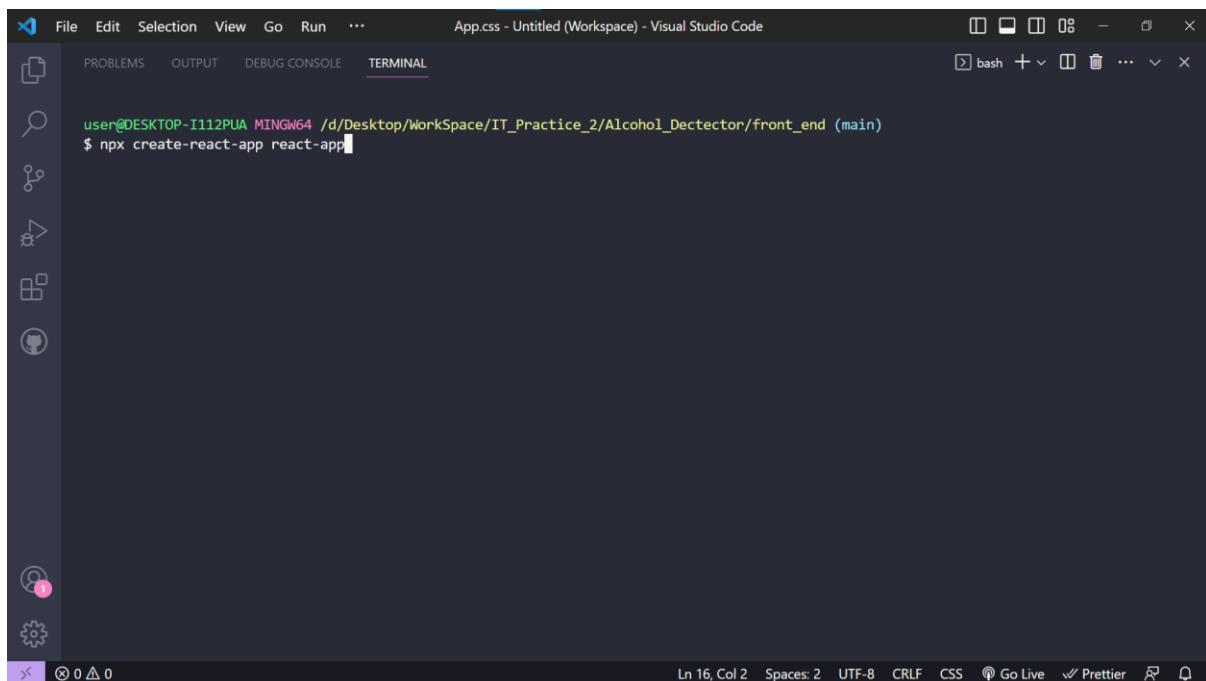
Bắt đầu cài đặt => Sau khi cài đặt, vào PowerShell để kiểm tra bằng lệnh:

**node -v**

```
PS C:\Users\user> node -v
v18.17.0
```

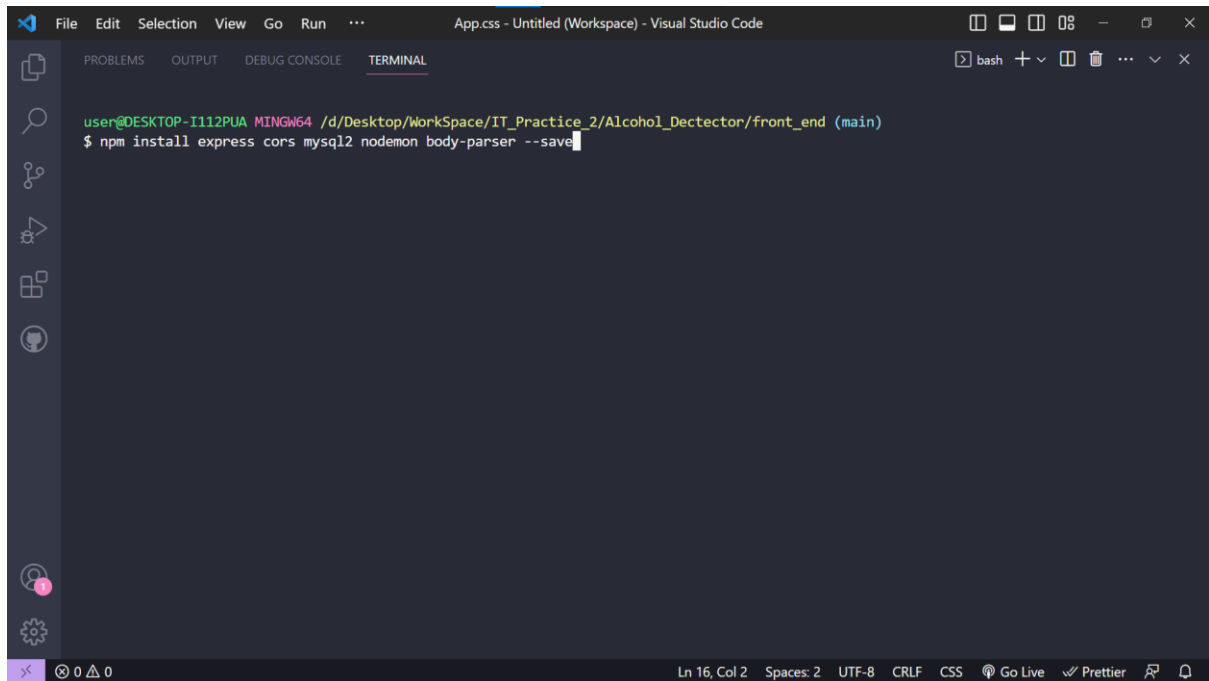
Để tạo 1 reactjs project dùng lệnh như sau (xài git hoặc powershell cái nào cũng được)

**npx create-react-app <filename>**



Để tải express cũng như những packages hỗ trợ thì dùng câu lệnh sau (dùng ở thư mục để phục vụ cho back-end):

**npm install express cors mysql2 nodemon body-parser --save**



The screenshot shows the Visual Studio Code interface with a terminal window open. The terminal title is "App.css - Untitled (Workspace) - Visual Studio Code". The terminal content shows the user's prompt and the command being executed:

```
user@DESKTOP-I112PUA MINGW64 /d/Desktop/WorkSpace/IT_Practice_2/Alcohol_Dectector/front_end (main)
$ npm install express cors mysql2 nodemon body-parser --save
```

The status bar at the bottom indicates the current line and column: "Ln 16, Col 2". Other status bar items include "Spaces: 2", "UTF-8", "CRLF", "CSS", "Go Live", "Prettier", and a search icon.

## 2. Postman

Link tải: <https://www.postman.com/downloads/>

The screenshot shows the Postman website's download page and a preview of the application. The website header includes links for Product, Pricing, Enterprise, Resources and Support, and Explore, along with a 'Launch Postman' button. The main heading is 'Download Postman', followed by a paragraph: 'Download the app to get started using the Postman API Platform today. Or, if you prefer a browser experience, you can try the web version of Postman.' Under the heading 'The Postman app', it says 'Download the app to get started with the Postman API Platform.' Below this, there are three download buttons: 'Windows 64-bit' (highlighted with a red circle and arrow), 'Mac 64-bit', and 'Linux 64-bit'. At the bottom, there is a link to the Privacy Policy and Terms. To the right, a preview of the Postman application interface is shown, featuring a sidebar with 'Collections' and 'Environments', a main workspace with a 'Notion's Public Workspace' and a 'New' button, and a right-hand pane showing a 'GET Retrieve a database' request to 'https://api.notion.com/v1/databases/id'.

**Download Postman**

Download the app to get started using the Postman API Platform today. Or, if you prefer a browser experience, you can try the web version of Postman.

**The Postman app**

Download the app to get started with the Postman API Platform.

[Windows 64-bit](#) [Mac 64-bit](#) [Linux 64-bit](#)

By downloading and using Postman, I agree to the [Privacy Policy](#) and [Terms](#).

The screenshot also shows a preview of the Postman application interface, displaying a workspace with a collection named 'Notion's Public Workspace' and a request named 'Retrieve a database'.

### 3. Mysql

Đầu tiên, tải sql server (bản community).

Link tải: <https://dev.mysql.com/downloads/mysql/>

General Availability (GA) Releases Archives

### MySQL Community Server 8.1.0 Innovation

Select Version:  
8.1.0 Innovation

Select Operating System:  
Microsoft Windows

Windows (x86, 64-bit), MSI Installer (mysql-8.1.0-winx64.msi)	8.1.0	146.9M	<b>Download</b> <small>MD5: 453d729afa2697a7a79d067830e071a6   Signature</small>
Windows (x86, 64-bit), ZIP Archive (mysql-8.1.0-winx64.zip)	8.1.0	236.9M	<b>Download</b> <small>MD5: 40a977d01e565b1d751ca068c823ba16   Signature</small>
Windows (x86, 64-bit), ZIP Archive Debug Binaries & Test Suite (mysql-8.1.0-winx64-debug-test.zip)	8.1.0	676.3M	<b>Download</b> <small>MD5: ec71c02d9e094e17dd764a2ee654fe0   Signature</small>

**\*\*Chú ý:** Khi tải xong, phải tải thêm visual studio rồi tiến hành cài đặt

Sau đó, tải mysql workbench

Link tải: <https://dev.mysql.com/downloads/workbench/>

### MySQL Workbench 8.0.34

Select Operating System:  
Microsoft Windows

Recommended Download:

#### MySQL Installer for Windows

All MySQL Products. For All Windows Platforms. In One Package.

Starting with MySQL 5.6 the MySQL Installer package replaces the standalone MSI packages.

Windows (x86, 32 & 64-bit), MySQL Installer MSI

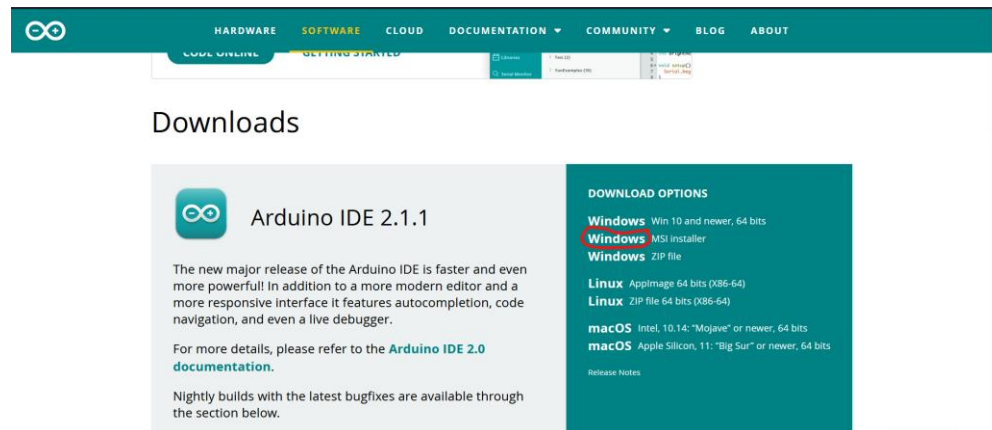
**Go to Download Page >**

Other Downloads:

Windows (x86, 64-bit), MSI Installer (mysql-workbench-community-8.0.34-winx64.msi)	8.0.34	46.4M	<b>Download</b> <small>MD5: ef5294cd0807979c060e1808fc488006   Signature</small>
---	--------	-------	---

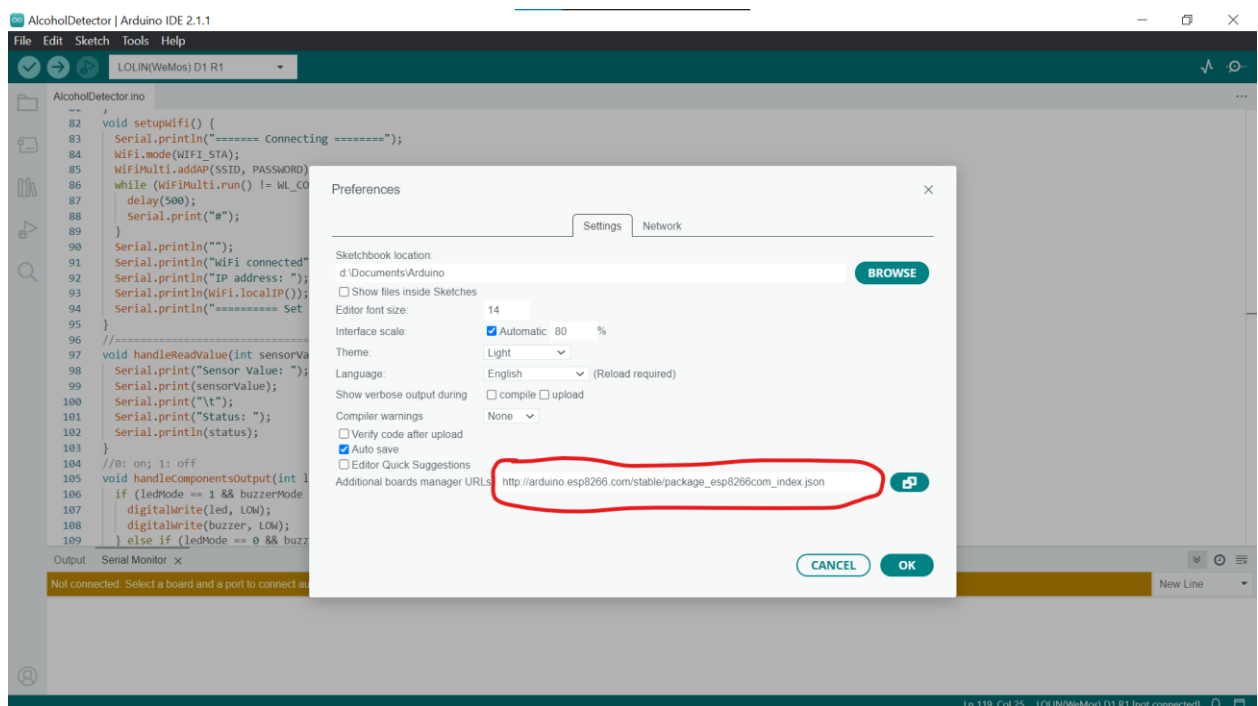
#### 4. Arduino ide

Link tải: <https://www.arduino.cc/en/software>

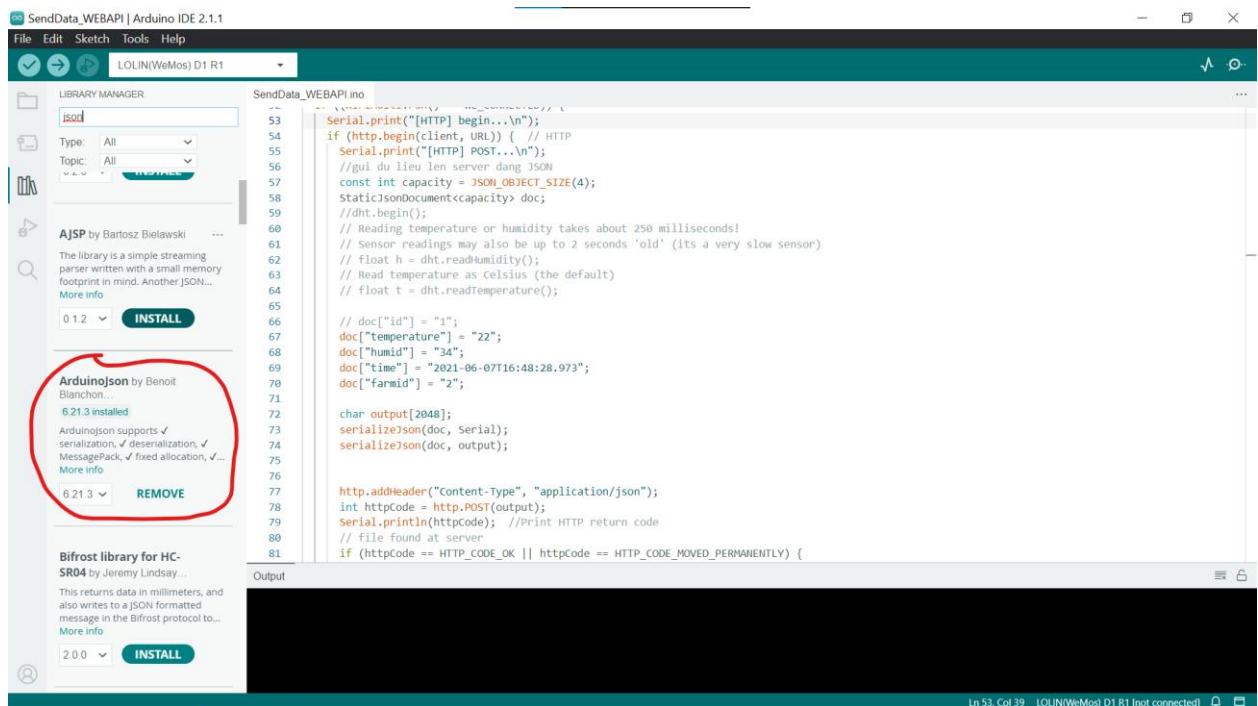


Cài đặt thư viện đối với wemos trong mục preferences (pass đường link vào như hình hướng dẫn):

[http://arduino.esp8266.com/stable/package\\_esp8266com\\_index.json](http://arduino.esp8266.com/stable/package_esp8266com_index.json)



Tải thư viện Arduino JSON (như hình dẫn dưới đây)



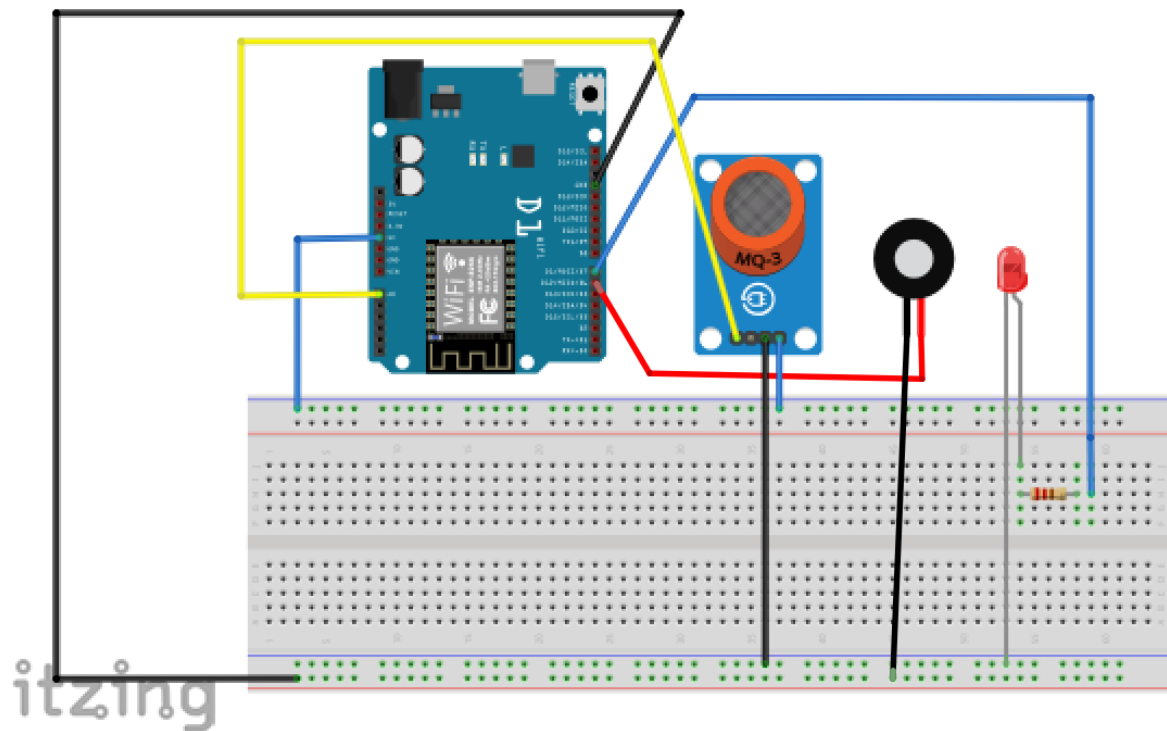
Tải Fritzing

Link tải: [https://arduinofactory.fr/en/download-fritzing-for-free/#Download\\_Fritzing\\_093b\\_for\\_FREE](https://arduinofactory.fr/en/download-fritzing-for-free/#Download_Fritzing_093b_for_FREE)

Rồi sau đó giải nén ra rồi xài (xài bản 64 bits).

## 5. Nối link kiện IoT

Như hình vẽ:





## Code mẫu

### I. lot (Arduino code)

```
1) #include <ESP8266WiFi.h>
2) #include <ESP8266WebServer.h>
3) #include <ESP8266mDNS.h>
4) #include <ESP8266WiFi.h>
5) #include <ESP8266WiFiMulti.h>
6) #include <ESP8266HTTPClient.h>
7) #include <WiFiClient.h>
8) #include <Arduino.h>
9) #include <ArduinoJson.h>
10)
11)ESP8266WiFiMulti WiFiMulti;
12)WiFiClient client;
13)HTTPClient http;
14)
15)#define ssid "thienloc"
16)#define password "Pie@2468"
17)#define url "http://192.168.2.8:8080/person"
18)const char *SSID = ssid;
19)const char *PASSWORD = password;
20)const char *URL = url;
21)
22)//settings
23)#define led D7
24)#define buzzer D6
25)#define sensorAnalog A0
26)#define air 650 //Input air value here
27)#define normal 700 //Input normal value here
28)#define drunk 800 //Input drunk value here
29)//json data
30)float p_StatusIndex;
31)String p_LName = "Input Here";
32)String p_MName = "Input Here";
33)String p_FName = "Input Here";
34)
35)//===== Main Code =====
36)void setup() {
37) // put your setup code here, to run once:
38) Serial.begin(115200);
39) setupComponents();
40) setupWifi();
41)}
42)void loop() {
43) // put your main code here, to run repeatedly:
44) //testSensor();
45) int sensorValue = analogRead(sensorAnalog);
46) if (sensorValue > air) {
```

```

47)   if (sensorValue <= normal) {
48)       handleReadValue(sensorValue, "Normal!");
49)       handleComponentsOutput(1, 1);
50)       p_StatusIndex = sensorValue / 100;
51)       postJSONDataNormal();
52)   } else if (sensorValue > normal && sensorValue <= drunk) {
53)       handleReadValue(sensorValue, "Drinking but within legal
        limits!!");
54)       handleComponentsOutput(0, 1);
55)       p_StatusIndex = sensorValue / 100;
56)       postJSONDataDrinking();
57)   } else {
58)       handleReadValue(sensorValue, "DRUNK!!!");
59)       handleComponentsOutput(0, 0);
60)       p_StatusIndex = sensorValue / 100;
61)       postJSONDataDrunk();
62)   }
63)   delay(10000);
64)   refreshUntilReturnAir();
65) } else {
66)   handleReadValue(sensorValue, "Air");
67)   delay(500);
68) }
69) }
70)
71) //=====
72) void setupComponents() {
73)   pinMode(led, OUTPUT);
74)   pinMode(buzzer, OUTPUT);
75)   for (uint8_t t = 5; t > 0; t--) {
76)       Serial.printf("[SETUP SENSOR] WAIT %d...\n", t);
77)       Serial.flush();
78)       delay(5000);
79)   }
80)   Serial.println("SETUP COMPLETE!!!");
81) }
82) void setupWifi() {
83)   Serial.println("==== Connecting =====");
84)   WiFi.mode(WIFI_STA);
85)   WiFiMulti.addAP(SSID, PASSWORD);
86)   while (WiFiMulti.run() != WL_CONNECTED) {
87)       delay(500);
88)       Serial.print("#");
89)   }
90)   Serial.println("");
91)   Serial.println("WiFi connected");
92)   Serial.println("IP address: ");
93)   Serial.println(WiFi.localIP());

```

```

94) Serial.println("===== Set Up Complete ===== ");
95) }
96) //=====
97) void handleReadValue(int sensorValue, String status) {
98)   Serial.print("Sensor Value: ");
99)   Serial.print(sensorValue);
100)     Serial.print("\t");
101)     Serial.print("Status: ");
102)     Serial.println(status);
103)   }
104)   //0: on; 1: off
105)   void handleComponentsOutput(int ledMode, int buzzerMode) {
106)     if (ledMode == 1 && buzzerMode == 1) {
107)       digitalWrite(led, LOW);
108)       digitalWrite(buzzer, LOW);
109)     } else if (ledMode == 0 && buzzerMode == 1) {
110)       digitalWrite(led, HIGH);
111)       digitalWrite(buzzer, LOW);
112)     } else if (ledMode == 0 && buzzerMode == 0) {
113)       digitalWrite(led, HIGH);
114)       digitalWrite(buzzer, HIGH);
115)     }
116)   }
117)   void refreshUntilReturnAir() {
118)     int sensorValue = analogRead(sensorAnalog);
119)     int updateSensorValue;
120)     int airCheck = air - 50;
121)     int count = 0;
122)     handleComponentsOutput(1, 1);
123)     Serial.println("===== REFRESH!!!
=====");
124)     while (true) {
125)       updateSensorValue = analogRead(sensorAnalog);
126)       if (updateSensorValue > airCheck) {
127)         Serial.printf("=== Loading...%d... ===\n", count++);
128)       } else {
129)         break;
130)       }
131)       delay(500);
132)     }
133)     Serial.println("===== REFRESH COMPLETE!!!!
=====");
134)     delay(2500);
135)     return;
136)   }
137)   //POST JSON Data
138)   void postJSONDataNormal() {
139)     Serial.print("connecting to ");

```

```

140)         if ((WiFiMulti.run() == WL_CONNECTED)) {
141)             Serial.print("[HTTP] begin...\n");
142)             if (http.begin(client, URL)) { // HTTP
143)                 Serial.print("[HTTP] POST...\n");
144)                 //gui du lieu len server dang JSON
145)                 const int capacity = JSON_OBJECT_SIZE(6);
146)                 StaticJsonDocument<capacity> doc;
147)
148)                 doc["p_LName"] = p_LName;
149)                 doc["p_MName"] = p_MName;
150)                 doc["p_FName"] = p_FName;
151)                 doc["p_StatusIndex"] = p_StatusIndex;
152)                 doc["p_Status"] = "Normal!";
153)
154)                 char output[2048];
155)                 serializeJson(doc, Serial);
156)                 serializeJson(doc, output);
157)
158)                 http.addHeader("Content-Type", "application/json");
159)                 int httpCode = http.POST(output);
160)                 Serial.println(httpCode); //Print HTTP return code
161)                 // file found at server
162)                 if (httpCode == HTTP_CODE_OK || httpCode ==
HTTP_CODE_MOVED_PERMANENTLY) {
163)                     String payload = http.getString();
164)                     Serial.println(payload);
165)                     Serial.println("done");
166)                 } else {
167)                     Serial.printf("[HTTP] POST... failed, error: %s\n",
http.errorToString(httpCode).c_str());
168)                 }
169)                 http.end(); //Close connection Serial.println();
170)                 Serial.println("closing connection");
171)             }
172)         }
173)     }
174)     void postJSONDataDrinking() {
175)         Serial.print("connecting to ");
176)         if ((WiFiMulti.run() == WL_CONNECTED)) {
177)             Serial.print("[HTTP] begin...\n");
178)             if (http.begin(client, URL)) { // HTTP
179)                 Serial.print("[HTTP] POST...\n");
180)                 //gui du lieu len server dang JSON
181)                 const int capacity = JSON_OBJECT_SIZE(6);
182)                 StaticJsonDocument<capacity> doc;
183)
184)                 doc["p_LName"] = p_LName;
185)                 doc["p_MName"] = p_MName;

```

```

186)         doc["p_FName"] = p_FName;
187)         doc["p_StatusIndex"] = p_StatusIndex;
188)         doc["p_Status"] = "Drinking but within legal limits!!";
189)
190)         char output[2048];
191)         serializeJson(doc, Serial);
192)         serializeJson(doc, output);
193)
194)         http.addHeader("Content-Type", "application/json");
195)         int httpCode = http.POST(output);
196)         Serial.println(httpCode); //Print HTTP return code
197)         // file found at server
198)         if (httpCode == HTTP_CODE_OK || httpCode ==
    HTTP_CODE_MOVED_PERMANENTLY) {
199)             String payload = http.getString();
200)             Serial.println(payload);
201)             Serial.println("done");
202)         } else {
203)             Serial.printf("[HTTP] POST... failed, error: %s\n",
    http.errorToString(httpCode).c_str());
204)         }
205)         http.end(); //Close connection Serial.println();
206)         Serial.println("closing connection");
207)     }
208) }
209) }
210) void postJSONDataDrunk() {
211)     Serial.print("connecting to ");
212)     if ((WiFiMulti.run() == WL_CONNECTED)) {
213)         Serial.print("[HTTP] begin...\n");
214)         if (http.begin(client, URL)) { // HTTP
215)             Serial.print("[HTTP] POST...\n");
216)             //gui du lieu len server dang JSON
217)             const int capacity = JSON_OBJECT_SIZE(6);
218)             StaticJsonDocument<capacity> doc;
219)
220)             doc["p_LName"] = p_LName;
221)             doc["p_MName"] = p_MName;
222)             doc["p_FName"] = p_FName;
223)             doc["p_StatusIndex"] = p_StatusIndex;
224)             doc["p_Status"] = "DRUNK!!!";
225)
226)             char output[2048];
227)             serializeJson(doc, Serial);
228)             serializeJson(doc, output);
229)
230)             http.addHeader("Content-Type", "application/json");
231)             int httpCode = http.POST(output);

```

```

232)         Serial.println(httpCode); //Print HTTP return code
233)         // file found at server
234)         if (httpCode == HTTP_CODE_OK || httpCode ==
HTTP_CODE_MOVED_PERMANENTLY) {
235)             String payload = http.getString();
236)             Serial.println(payload);
237)             Serial.println("done");
238)         } else {
239)             Serial.printf("[HTTP] POST... failed, error: %s\n",
http.errorToString(httpCode).c_str());
240)         }
241)         http.end(); //Close connection Serial.println();
242)         Serial.println("closing connection");
243)     }
244) }
245) }
246) void testSensor() {
247)     while(true) {
248)         int sensorValue = analogRead(sensorAnalog);
249)         Serial.print("Sensor value: "); Serial.println(sensorValue);
250)         delay(500);
251)     }
252) }

```

## II. Back-end

### 1. Restful API

```
2. var express = require('express');
3. var bodyParser = require("body-parser");
4. var cors = require('cors');
5. var morgan = require('morgan');
6. var mysql = require('mysql2');
7.
8. var app = express();
9. app.use(cors());
10.   app.use(express.static('public'));
11.   app.use(bodyParser.json());
12.   app.use(bodyParser.urlencoded({ extended: true }));
13.
14.   //mysql
15.   var con = mysql.createConnection({
16.     host: "localhost",
17.     port: "3306",
18.     user: "root",
19.     password: "1234",
20.     insecureAuth: true,
21.     database: "alcoholdb"
22.   });
23.   con.connect(function (err) {
24.     if (err) throw err;
25.     console.log("Connected!!!")
26.   });
27.
28.   //Test API
29.   app.get("/", function (req, res) {
30.     res.send("Hello World");
31.   });
32.   //Restful API
33.   app.get("/person", function (req, res) {
34.     // const id = req.params.id;
35.     var sql = "select * from person;";
36.     con.query(sql, function (err, results) {
37.       if (err) throw err;
38.       //console.log(results);
39.       res.send(results);
40.     });
```

```

41.     });
42.     app.get("/person/:id", function (req, res) {
43.         const id = req.params.id;
44.         var sql = "select * from person where id = "+id+"";
45.         con.query(sql, function (err, results) {
46.             if (err) throw err;
47.             //console.log(results);
48.             res.send(results);
49.         });
50.     });
51.     app.post("/person", function(req, res) {
52.         const {p_LName, p_MName, p_FName, p_StatusIndex,
53.             p_Status} = req.body;
54.         var sql = "insert into person (p_LName, p_MName,
55.             p_FName, p_StatusIndex, p_Status) values
56.             ('"+p_LName+"', '"+p_MName+"', '"+p_FName+"', '"+p_StatusIndex+"', '
57.             "+p_Status+"')";
58.         con.query(sql, function (err, results) {
59.             if (err) throw err;
60.             //console.log(results);
61.             // res.send("Add complete!!");
62.             // res.send(results);
63.             res.send("Add Complete ("+p_LName+", "+p_MName+",
64.                 "+p_FName+", "+p_StatusIndex+", "+p_Status+)");
65.         });
66.     });
67.     app.put("/person/:id", function(req, res) {
68.         const {id} = req.params.id;
69.         const {p_ID, p_LName, p_MName, p_FName} = req.body;
70.         var sql = "update person set p_ID = '"+p_ID+"',
71.             p_LName = '"+p_LName+"', p_MName = '"+p_MName+"', p_FName =
72.             '"+p_FName+"' where ID = "+id+"";
73.         con.query(sql, function (err, results) {
74.             if (err) throw err;
75.             //console.log(results);
76.             res.send("Update complete!!");
77.             //res.send(results);
78.         });
79.     });
80.     app.delete("/person/:id", function(req, res) {
81.         const id = req.params.id;
82.         var sql = "delete from person where ID = "+id+"";

```



```
76.         con.query(sql, function (err, results) {
77.             if (err) throw err;
78.             //console.log(results);
79.             res.send("Delete complete!!");
80.             //res.send(results);
81.         });
82.     });
83.
84.     //server
85.     const port = 8080;
86.     var server = app.listen(port, function () {
87.         var host = server.address().address;
88.         console.log(`Server is listening at
            http://${host}:${port}`)
89.     });
```

## 2. Mysql Query

```
create database alcoholdb;
```

```
use alcoholdb;
```

```
create table person(
```

```
    ID int not null auto_increment,
```

```
    p_ID int null,
```

```
    p_LName varchar(255),
```

```
    p_MName varchar(255),
```

```
    p_FName varchar(255),
```

```
    p_StatusIndex float,
```

```
    p_Status varchar(255),
```

```
    -- s_id int,
```

```
    constraint pk_person primary key (ID),
```

```
    constraint uc_person unique (p_ID)
```

```
);
```

```
-- create table sensor(
```

```
--     id int not null auto_increment,
```

```
--     s_id int null,
```

```
--     s_normal float,
```

```
--     s_drunklegal float,
```

```
--     s_drunkillegal float,
```

```
--     constraint pk_sensor primary key (id, s_id)
```

```
-- );
```

```
select * from person;
```

```
insert into person (p_ID, p_LName, p_MName, p_FName, p_StatusIndex, p_Status)
```

```
    values ('p_ID','p_LName','p_MName','p_FName',p_StatusIndex,'p_Status');
```

```
update person set p_ID = 'p_ID', p_LName = 'p_LName', p_MName = 'p_MName', p_FName  
= 'p_FName' where id = id;
```

```
delete from person where id = id;
```

```
delete from person where ID = 3;
```

### III. Front-end

Sẽ cập nhật sau

## Kết luận

Thông qua dự án trên em đã học được rất nhiều kiến thức mới như lần đầu tiếp xúc với reactjs, node & express cũng như IoT...

Em cảm ơn sự đồng hành của cô trong những buổi học qua và cũng như mong muốn gặp lại cô trong quá trình học để học hỏi thêm

Em xin cảm ơn.

## Nguồn tham khảo

Những bài hướng dẫn lab trong lớp.

Arduino document: <https://www.arduino.cc/reference/en/>

Stack Overflow: <https://stackoverflow.com/>

Bootstrap documentation: <https://getbootstrap.com/docs/5.3/getting-started/introduction/>

MDB bootstrap: <https://mdbootstrap.com/>

Npm documentation: <https://docs.npmjs.com/>







