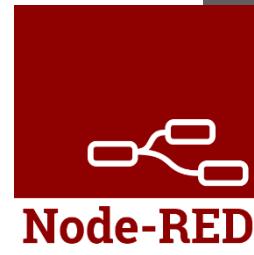


# BÁO CÁO THIẾT KẾ 3D VÀ WEB ĐỒ ÁN CUỐI KÌ

---



*Trường Đại học Khoa học Tự nhiên*

*Vật lý Đại cương 2*

*Lớp 19CLC9*

*Nhóm 10*

Hồ Ngọc Minh Đức 19127368

Nguyễn Phương Vy 19127088

Nguyễn Kim Thị Tố Nga 19127219

Lê Quốc Hòa Cao Xuân Nam Đặng Hoài Thương

---

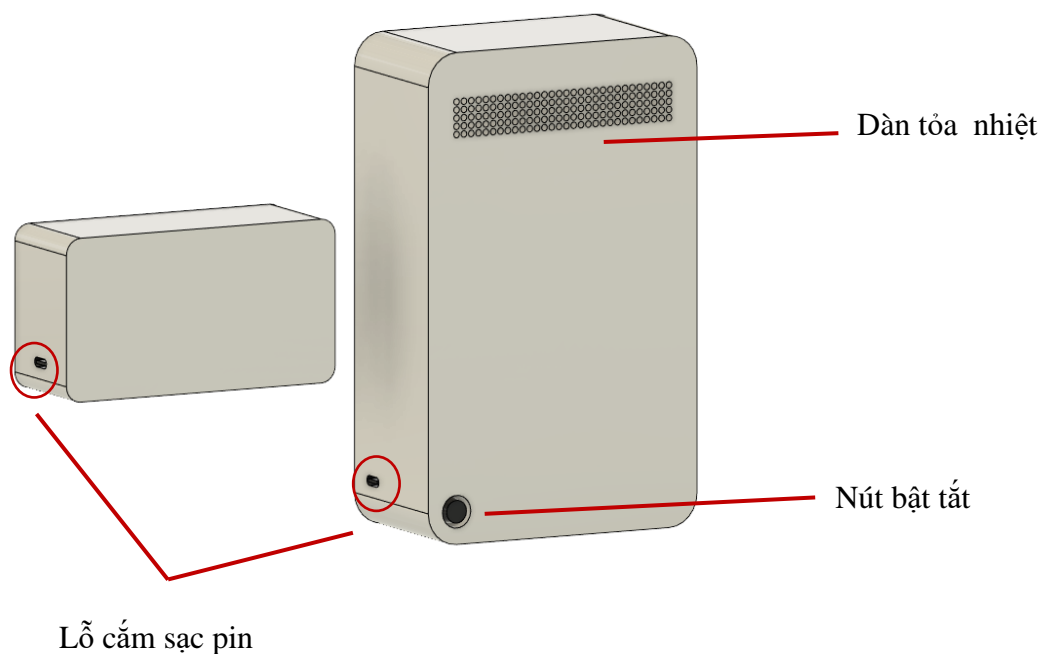
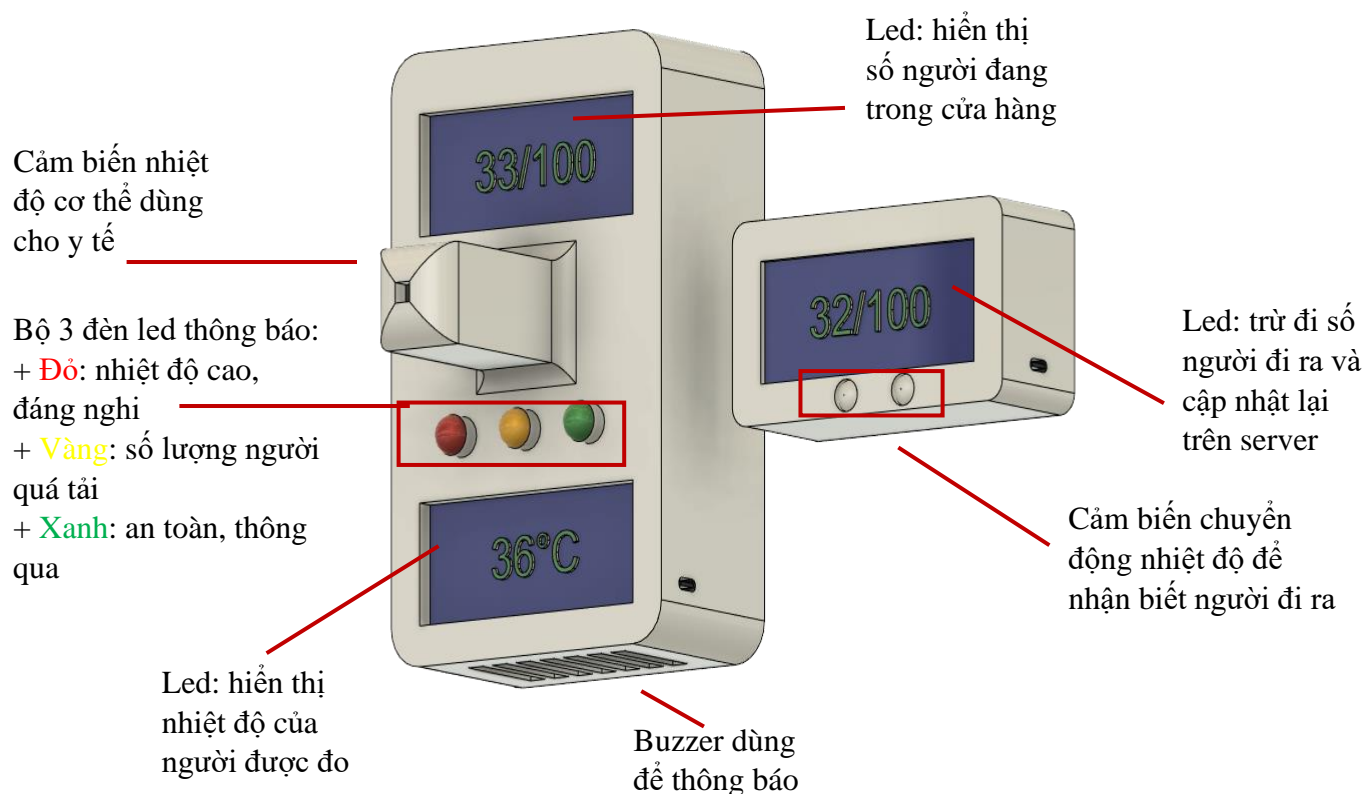
*Năm học*  
*(2019-2023)*

# Contents

<b>I. Thiết kế 3D .....</b>	<b>3</b>
<b>II. Website sản phẩm.....</b>	<b>4</b>
<b>1.Login/Logout .....</b>	<b>4</b>
<b>2.List store .....</b>	<b>14</b>
<b>a.Add Store .....</b>	<b>16</b>
<b>b.Setting Shop .....</b>	<b>19</b>
<b>c.Add device.....</b>	<b>30</b>
<b>3.Realtime .....</b>	<b>33</b>
<b>4.About Us .....</b>	<b>35</b>

# I. Thiết kế 3D

*Sản phẩm : Bộ nhiệt kế đo trán không tiếp xúc*



## II. Website sản phẩm

Web sản phẩm của nhóm được thiết kế bằng công cụ Node red, kết hợp lưu trữ dữ liệu trên nền tảng database **Firestore** và **ThingSpeak**. Web có các chức năng phù hợp giúp hỗ trợ theo dõi tình hình cũng như cập nhật trên các thiết bị tại nhiều cửa hàng.

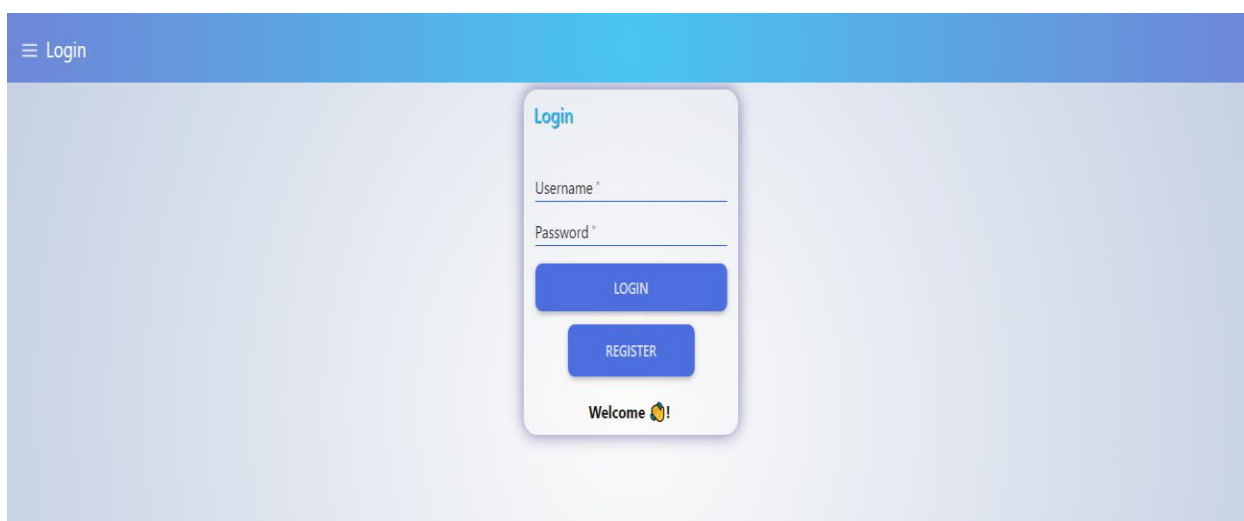
**Link website:** <http://19clc9-nhom10.mybluemix.net/ui>

Vai trò từng node sẽ được phân tích dựa trên các chức năng của website.

### 1. Login/Logout

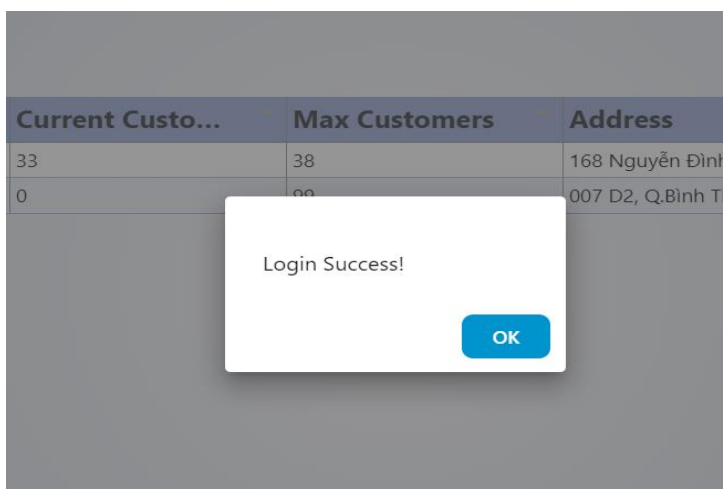
#### ❖ Login

Khi truy cập vào website, người dùng sẽ nhận được yêu cầu đăng nhập từ trang **Login**.

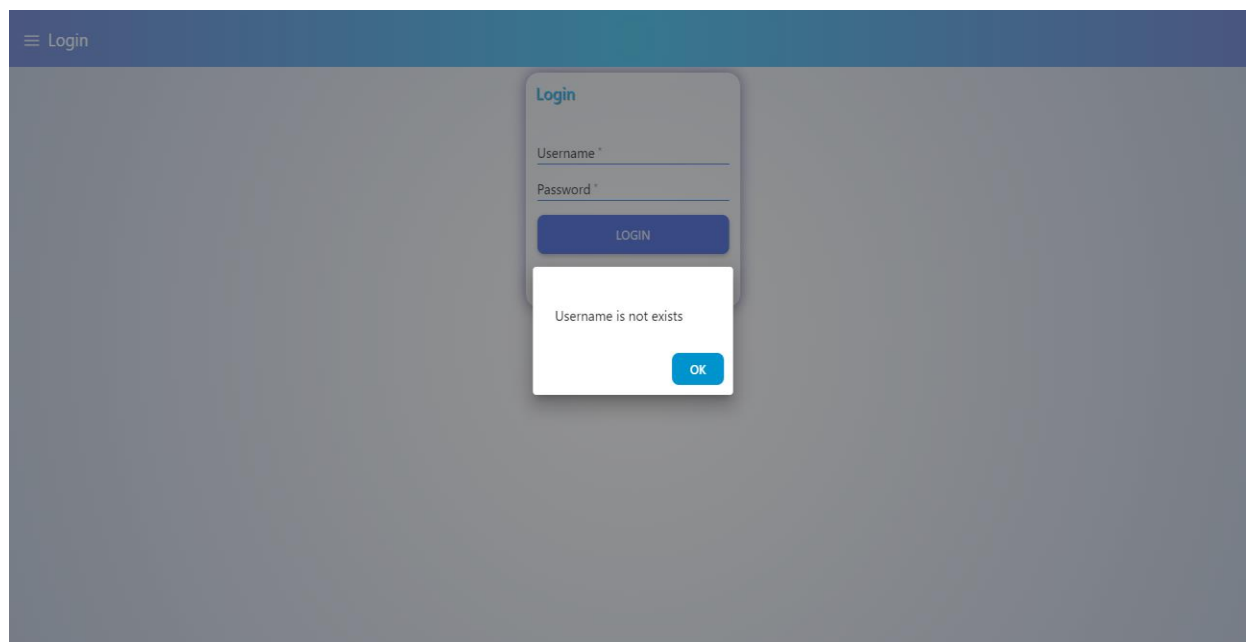


Dữ liệu thông tin tài khoản được kiểm tra từ database trong **Firestore**.

Nếu thông tin đăng nhập thành công, màn hình sẽ xuất ra thông báo **“Login Success!”** và web tự động chuyển qua tab **Home**.

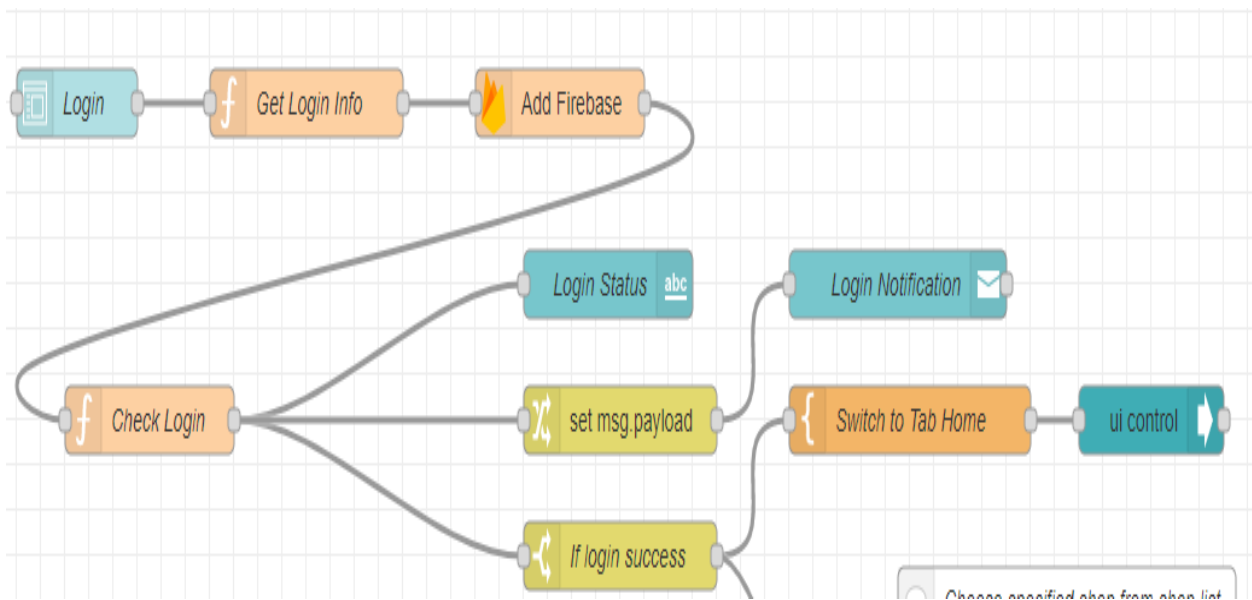


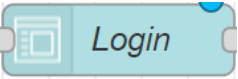
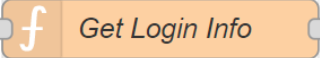
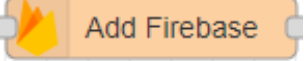
Ngược lại nếu mật khẩu đăng nhập không đúng hoặc tên tài khoản không tồn tại thì xuất hiện bảng thông báo **“Password is incorrect” / “Username is not exists”**.


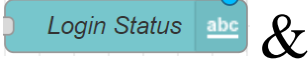
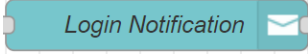
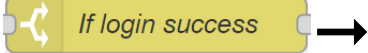



**Flow được thiết kế như sau:**


Cần cài đặt thư viện **node-red-contrib-firebase-data** để sử dụng database trên Firebase (vào trang web chính của Firebase để tạo tài khoản lưu trữ dữ liệu).



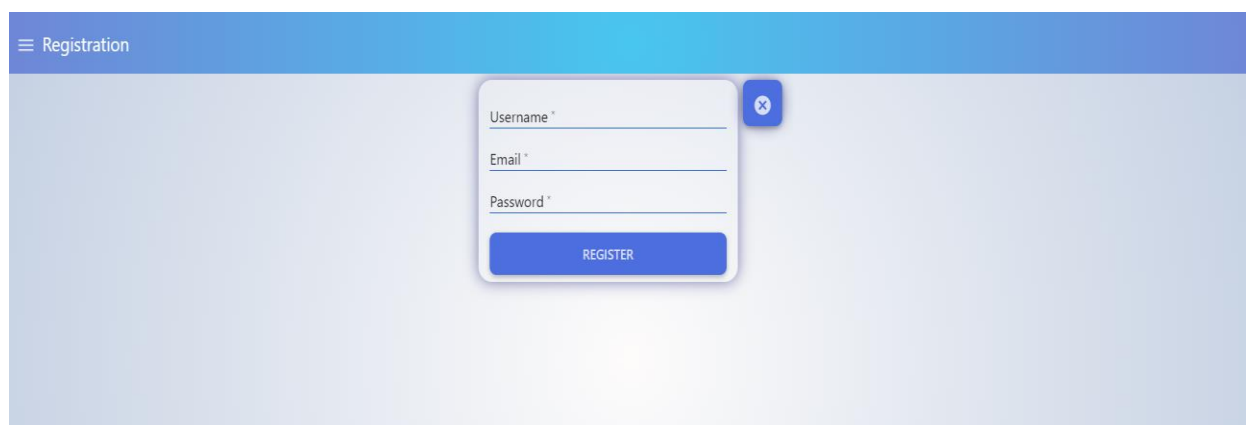
Node	Vai trò
	Node <i>ui_form</i> để input username và password
 	<pre> 1 flow.set("username", msg.payload.username); 2 flow.set("password", msg.payload.password); 3 return msg; </pre> <p>Lấy giá trị username và password từ payload để đưa vào node firebase kiểm tra</p>

	<pre> 1 username = flow.get("username"); 2 password = flow.get("password"); 3 status = true; 4 str = ""; 5 if (username in msg.payload.admins == false){ 6     status = false; 7     str = "Username is not exists"; 8 } 9 else if (password != msg.payload.admins[username]){ 10    status = false; 11    str = "Password is incorrect"; 12 } 13 else { 14    status = true; 15    str = "Login Success!"; 16 } 17 18 msg.payload = {status: status, str: str}; 19 return msg; </pre> <p>Kiểm tra username và password trong flow database, trả về string trong msg.payload để thông báo ra màn hình</p>
 	<p>Node text để thông báo ở khung đăng nhập, node notification (lấy giá trị từ node set msg.payload) thông báo ở góc phải phía trên màn hình</p>
 	<p>Nếu check đăng nhập thành công chuyển sang tab Home bằng node template</p>

### ❖ Register

Nếu người dùng chưa có tài khoản thì có thể đăng kí tài khoản bằng cách bấm vào nút  .

Màn hình sẽ tự động đưa người dùng đến trang **Register** để đăng ký các thông tin *username, password, email*.



Registration

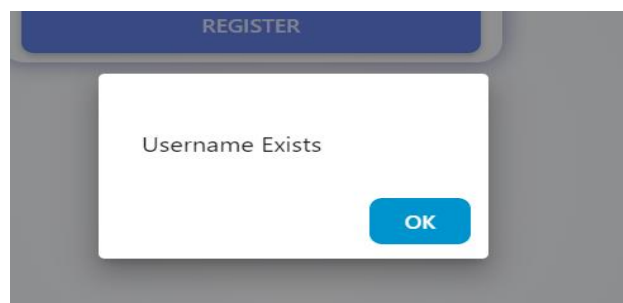
Username \*

Email \*

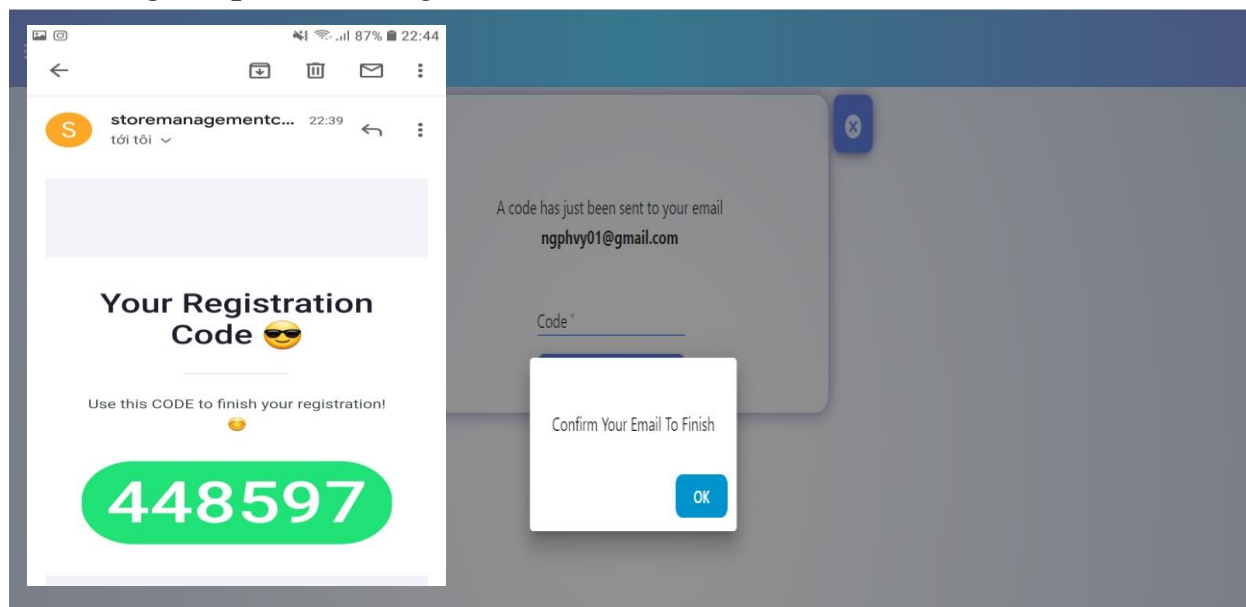
Password \*

REGISTER

Nếu Username đã tồn tại thì xuất ra thông báo và yêu cầu đăng nhập lại.



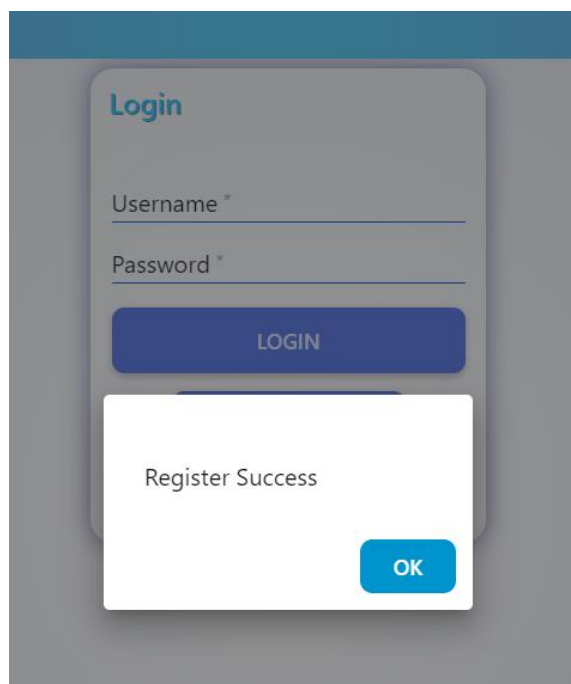
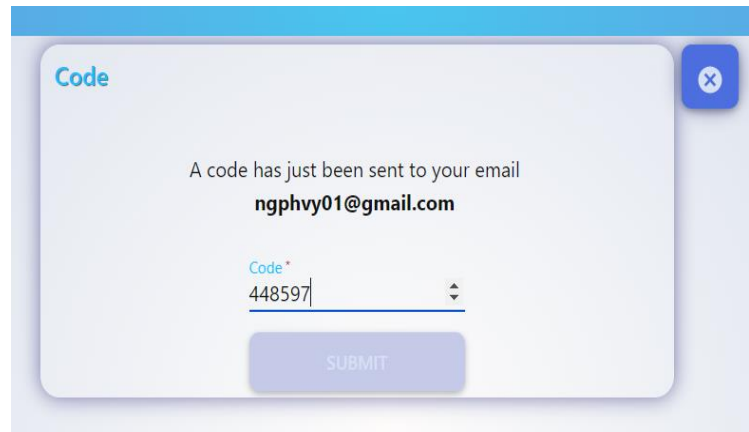
Nếu đăng nhập thành công:



Một mã code sẽ được gửi đến email vừa được đăng kí để xác nhận thông tin.

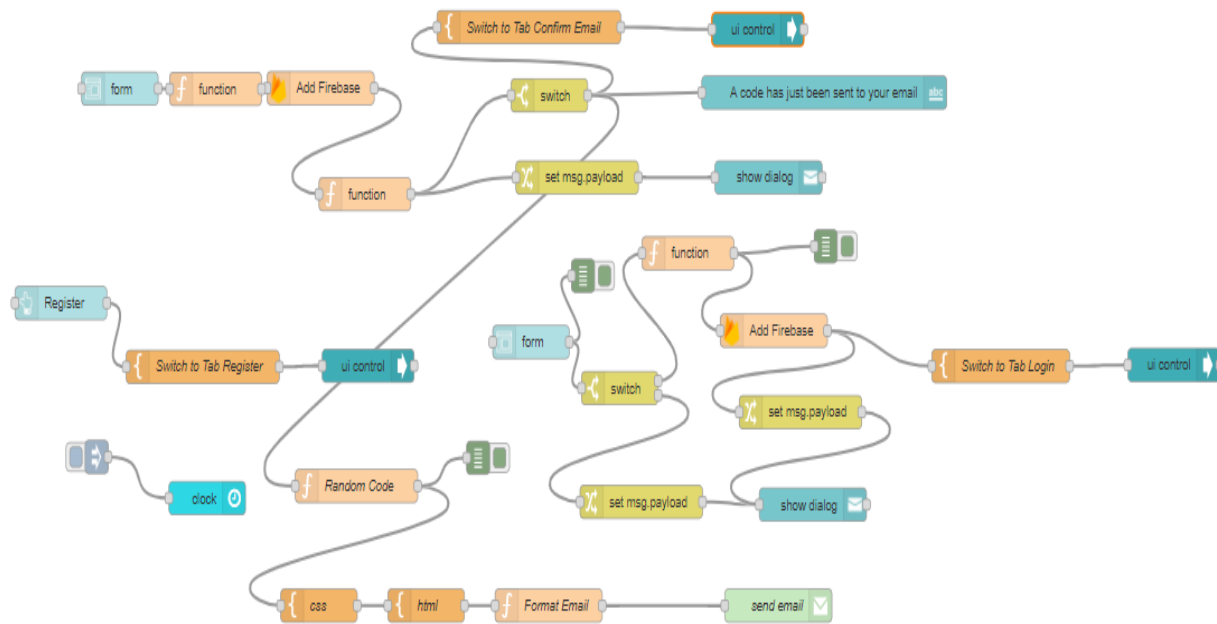




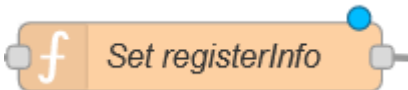

Nhập mã code vừa nhận được là bạn đã đăng kí thành công.

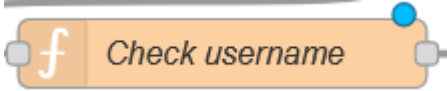
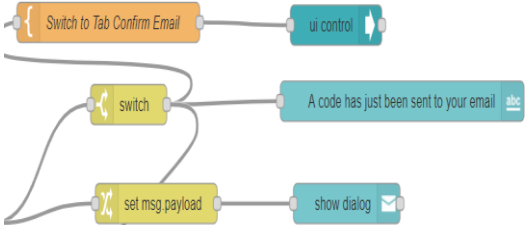
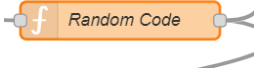


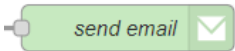


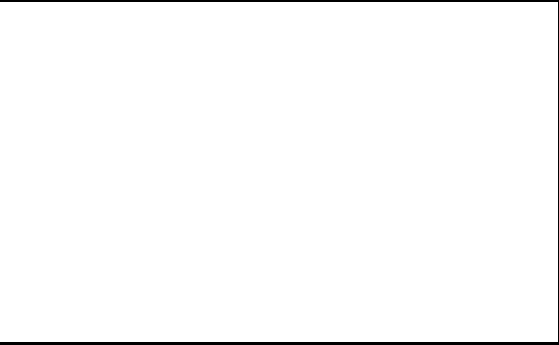


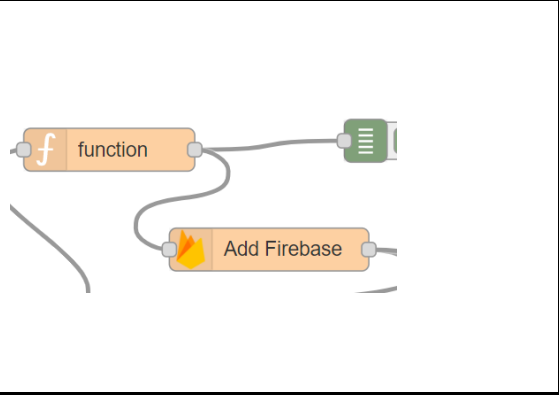
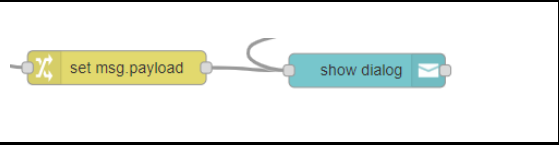
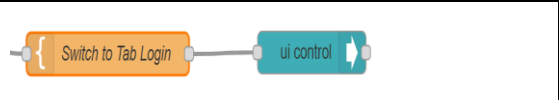
Trang web sẽ trở về phần **Login** để người dùng đăng nhập lại vào tài khoản.

**Phần flow được thực hiện như sau:**



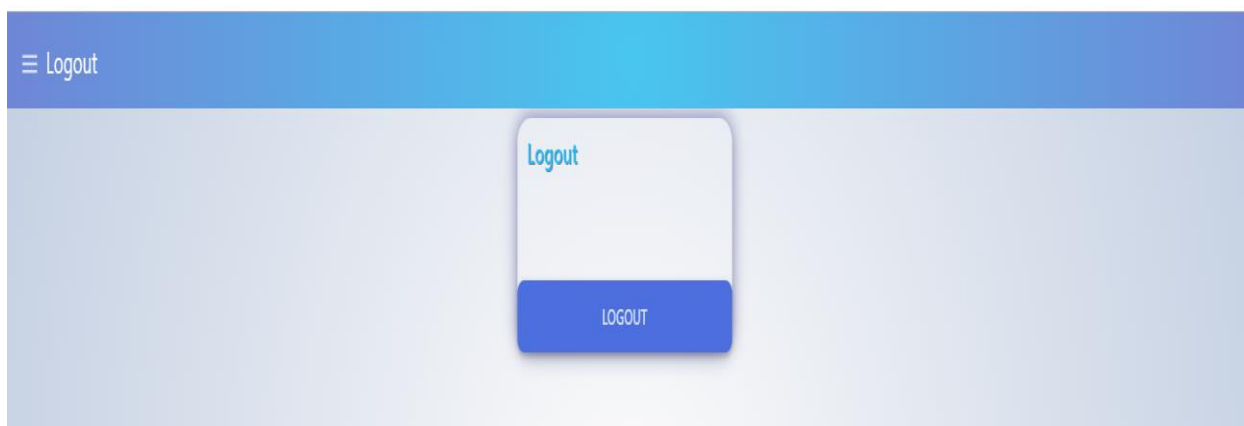
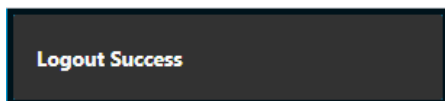
Node	Vai trò
	Bấm nút <b>Register</b> ở tab <b>Login</b> để chuyển qua tab <b>Register</b> nhờ <b>ui_control</b> .
	Node <b>ui_form</b> cho phép người dùng nhập vào các thông tin cần thiết để đăng kí.
	Đưa các thông tin vừa được đăng kí vào <b>msg.payload</b> <pre> 1 msg.username = msg.payload.username; 2 msg.password = msg.payload.password; 3 msg.email = msg.payload.email; 4 flow.set("registerInfo", msg.payload); 5 return msg; </pre>
	Truy suất database trên <b>Firebase</b> để kiểm tra thông tin người đăng kí bằng method <b>get</b>

	<p>Kiểm tra trong thông tin lấy được từ database (<i>msg.payload</i>) .</p> <p>Nếu <i>username</i> đã tồn tại thì trả về status thông báo.</p> <p>Ngược lại thì gán <i>status</i> là <i>confirm email</i></p> <pre> 1 admins = msg.payload; 2 msg.status = ""; 3 msg.result = false; 4 if (msg.username in admins){ 5     msg.status = "Username Exists"; 6     return msg; 7 } 8 msg.result = true; 9 msg.status = "Confirm Your Email To Finish"; 10 return msg; </pre>
	<p>Từ status được nhận được sẽ xuất ra thông báo cho người dùng biết được thông tin có hợp lệ hay chưa. Nếu hợp lệ thì chuyển qua tab <b>Confirm email</b> .</p>
	<p>Xuất ra một đoạn code 6 kí tự bất kì để gửi qua email, đồng thời lưu đoạn code vào flow.</p> <pre> 1 msg.code = Math.floor(100000 + Math.random() * 900000); 2 msg.to = ""; 3 flow.set("code", msg.code); 4 return msg; </pre>
	<p>Dùng để thiết kế email được gửi đến người dùng.</p>
	<p>Thiết lập nội dung email (<i>payload</i>), chủ đề (<i>topic</i>) và email nhận (<i>msg.email</i>)</p> <pre> 2 msg={ 3     payload:msg.payload, 4     topic:"Store Managemet Registration", 5     to: msg.email 6 }; 7 8 9 return msg; </pre>
	<p>Cài đặt thư viện <b>node-red-node-email</b> để gửi email với <i>payload</i> vừa nhận được bằng email chính của web.</p>

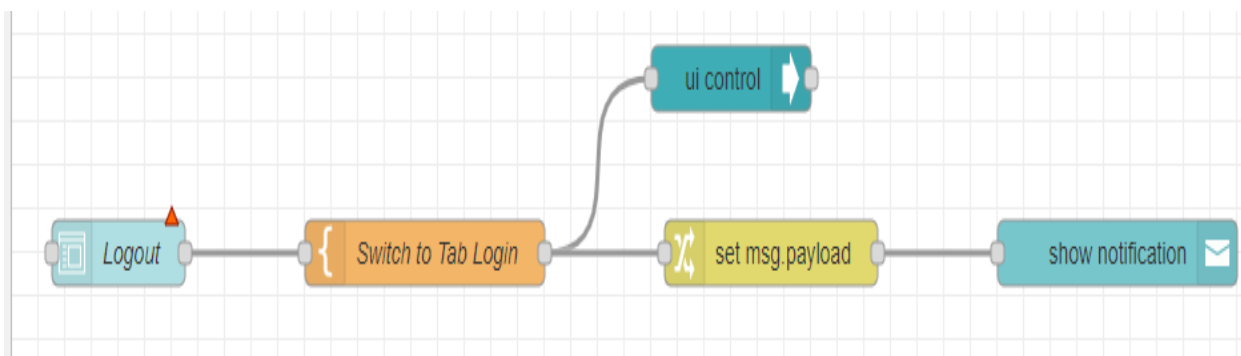
	
	<p>Node <b>form</b> cho phép người dùng nhập mã code vừa nhận được qua email.</p>
	<p>Kiểm tra code với đoạn code được lưu trong flow trước đó.</p>
	<p>Nếu code nhập đúng thì node <b>function</b> sẽ lấy các thông tin đăng kí được lưu trong <b>flow</b>.</p> <pre> 1 data = flow.get("registerInfo"); 2 msg.payload = {[data.username]: { 3     "email": data.email, 4     "password": data.password 5 }} 6 return msg; </pre> <p>Thông tin sẽ được thêm vào database.</p>
	<p>Xuất ra thông báo đã thêm thành công hoặc sai mã code.</p>
	<p>Trở về tab <b>Login</b> để người dùng đăng nhập lại.</p>




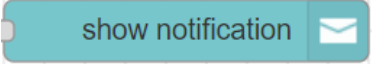
## ❖ Logout

Khi người dùng muốn đăng xuất thì vào tab **Logout** và chọn button **Logout** → xuất hiện thông báo **Logout Success** góc trên bên phải của web và chuyển về tab **Login**



Về phần flow:



Node	Vai trò
 	Khi nhấn nút button Log out, quay lại tab Login bằng node template và <i>ui_control</i>
 	Khi được chuyển qua tab Login, gửi msg.payload và thông báo lên góc phải bên trên là “Logout Success”

## 2. List store


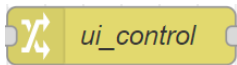
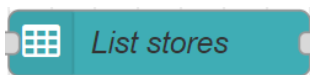
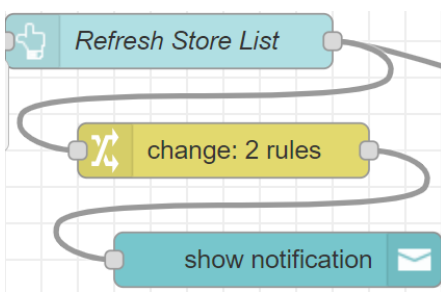
Ở tab **Home**, web sẽ load danh sách các cửa hàng từ **Firestore** để hiển thị danh sách các cửa hàng cùng số lượng thiết bị, số lượng người cho phép, số người hiện có và vị trí của mỗi cửa hàng.

Shop	Number devices	Current Custo...	Max Customers	Address
Siêu thị 1	1	33	38	168 Nguyễn Đình Chiểu, Phường 6, Quận 3, Thành phố Hồ Chí Minh
Siêu thị 2	0	0	99	007 D2, Q.Bình Thạnh, TP HCM

Ngoài ra, ở đầu danh sách có thêm 2 button để reload lại danh sách cửa hàng và đăng ký cửa hàng mới.

**Flow của phần list store được thực hiện như sau:**

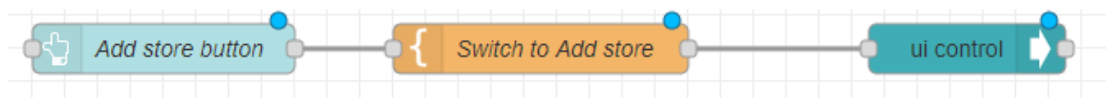


Node	Vai trò
	<p>Từ Firebase thông tin các store trong payload và đưa vào mảng trong payload để trả về.</p> <pre> 1 data = [] 2 for (store in msg.payload) 3   data.push(msg.payload[store]); 4 flow.set("storeData", msg.payload); 5 6 msg.payload = data; 7 return msg; </pre>
	<p>Tùy chỉnh hiển thị của list như độ dài, rộng, các column trong bảng</p>
	<p>Node dùng để hiển thị danh sách cửa hàng từ thông tin đã lấy</p>
	<p>Button gửi payload reload danh sách và thông báo reload success</p>

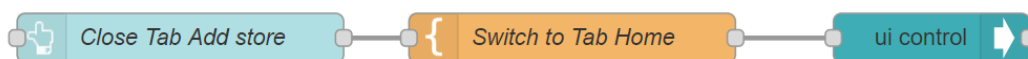
### a. Add Store

Trang **Add Store** cho phép người dùng nhập các thông tin cần thiết của cửa hàng và **submit**.

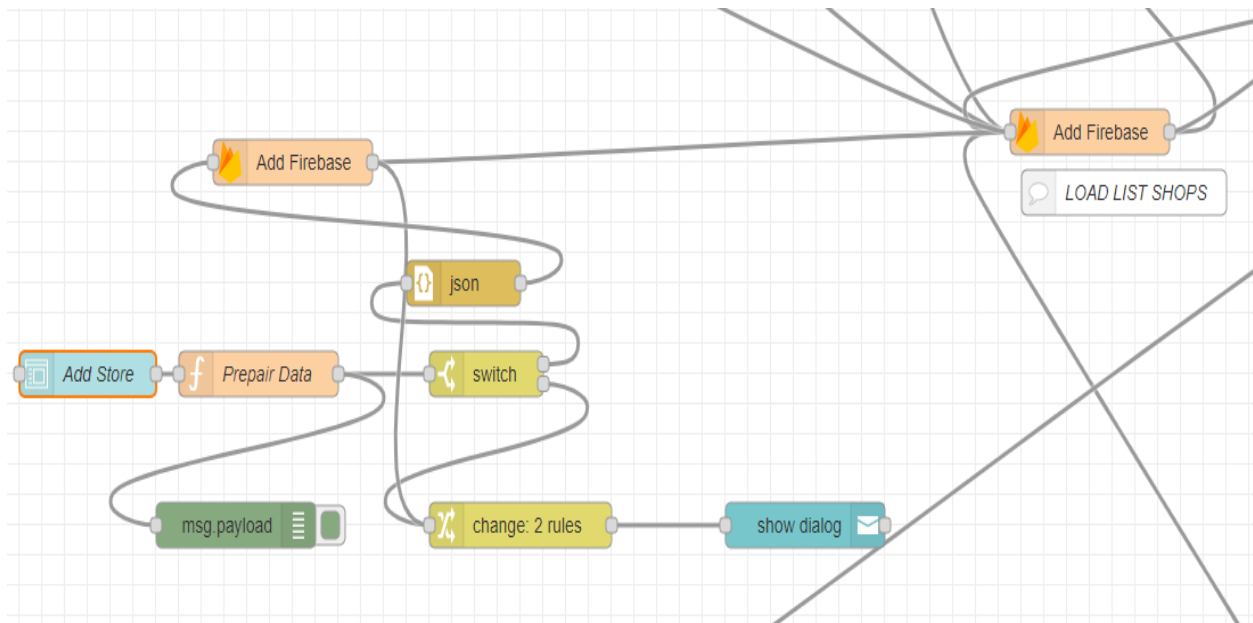
Nút **Add Store** hiển thị ở đầu danh sách cửa hàng cho phép người dùng đăng ký một cửa hàng mới vào danh sách hiện hành. Node **Add store button** sẽ chuyển trực tiếp sang trang **Add store** nhờ vào node **ui\_control** khi được người dùng bấm vào.

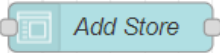
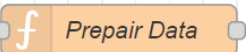


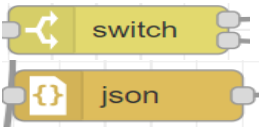
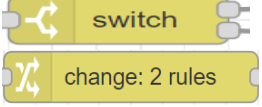


Ngoài ra còn có nút close cho phép người dùng trở về tab Home.







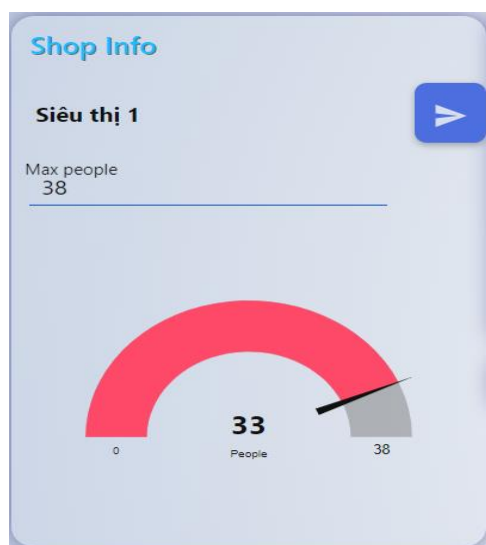
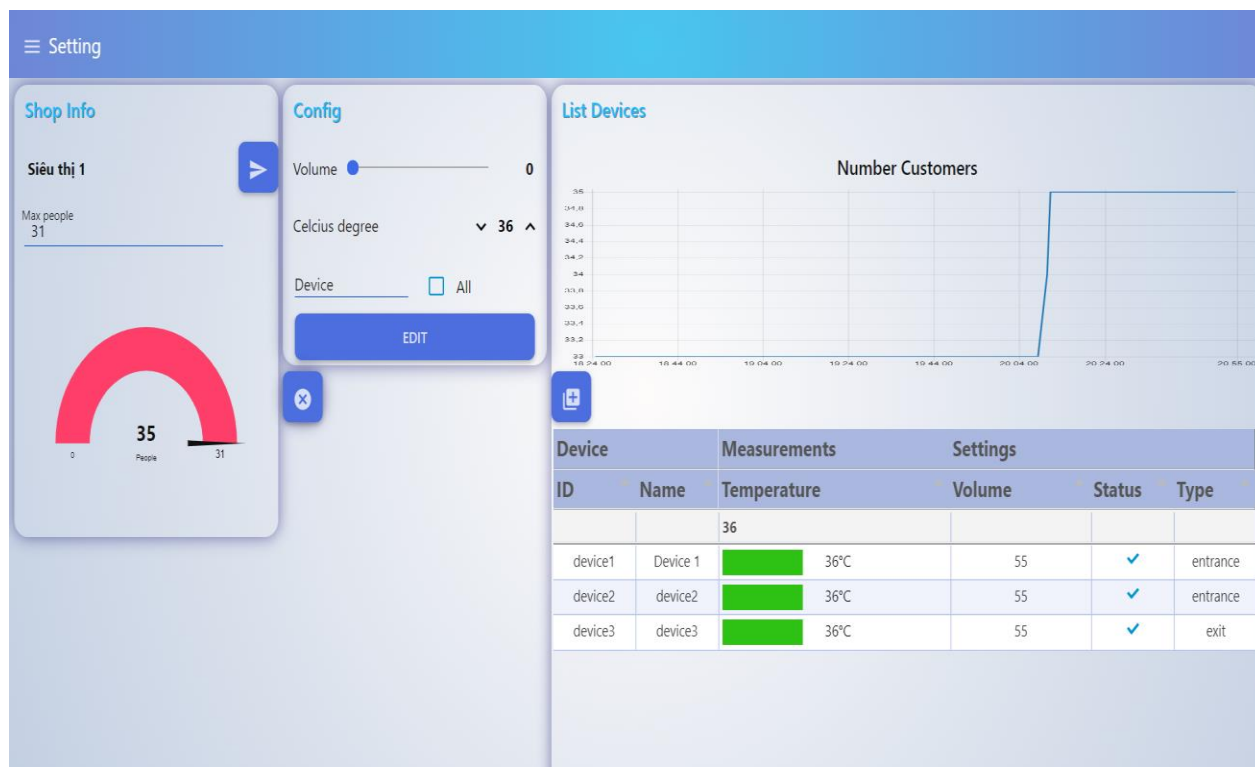
Node	Vai trò
	Node ui_form cho phép người dùng nhập thông tin store cần thêm như id, name, address, max_customers, sau đó truyền thông tin vào node function.
	<p>Lấy thông tin của các store hiện có từ flow (<i>storeData</i>). Kiểm tra xem store cần thêm đã tồn tại hay chưa. Nếu tồn tại rồi thì trả về thông báo.</p> <pre> 1 data = flow.get("storeData"); 2 newStore = msg.payload; 3 if (newStore.storeId in data == true){ 4     msg.result = false; 5     msg.topic = "Store ID Exists!"; 6     return msg; 7 } </pre> <p>Đưa các thông tin của store mới <i>newStore</i> vào msg.payload. Sau đó trả về tin nhắn thông báo thêm thành công.</p>

	<pre> 8  msg.payload = { 9    [newStore.id]: { 10      "id": newStore.id, 11      "name": newStore.name, 12      "address": newStore.address, 13      "devices": null, 14      "maxPeople": newStore.maxPeople, 15      "num_customers": 0, 16      "num_devices": 0, 17      "mqtt": {"port": 1883, 18              "server": "test.mosquitto.org", 19              "topic": "19clc9/nhom10/"+newStore.id 20            }, 21      "thingspeak": { 22        "writeKey": newStore.writeKey, 23        "readKey": newStore.readKey, 24        "id": newStore.thingspeakid, 25      } 26    } 27  }; 28  msg.result = true; 29  msg.topic = "Add store "+newStore.name+" Success"; 30  return msg; </pre>
	<p>[TRUE] Nếu node switch nhận kết quả true từ msg.result thì data về cửa hàng trong msg.payload sẽ được đưa vào node json và gửi lên Firebase</p>
	<p>[FALSE] Nếu node switch nhận kết quả false từ msg.result thì người dùng sẽ nhận được thông báo không thành công</p>
	<p>Cập nhật thông tin của store mới vào database Firebase bằng phương thức update. Sau đó flow sẽ chuyển đến node Firebase tiếp theo để cập nhật lại thông tin vào bảng các store.</p>
	<p>Thông báo store đã tồn tại hoặc đã được tạo thành công.</p>

## b. Setting Shop

Phần trang Setting store có giao diện như dưới đây:

Ở danh sách các cửa hàng, khi nhấp chọn một cửa hàng người dùng sẽ được chuyển tới tab **Setting** của cửa hàng: gồm bảng **Shop info**, **Config** và **List Device**



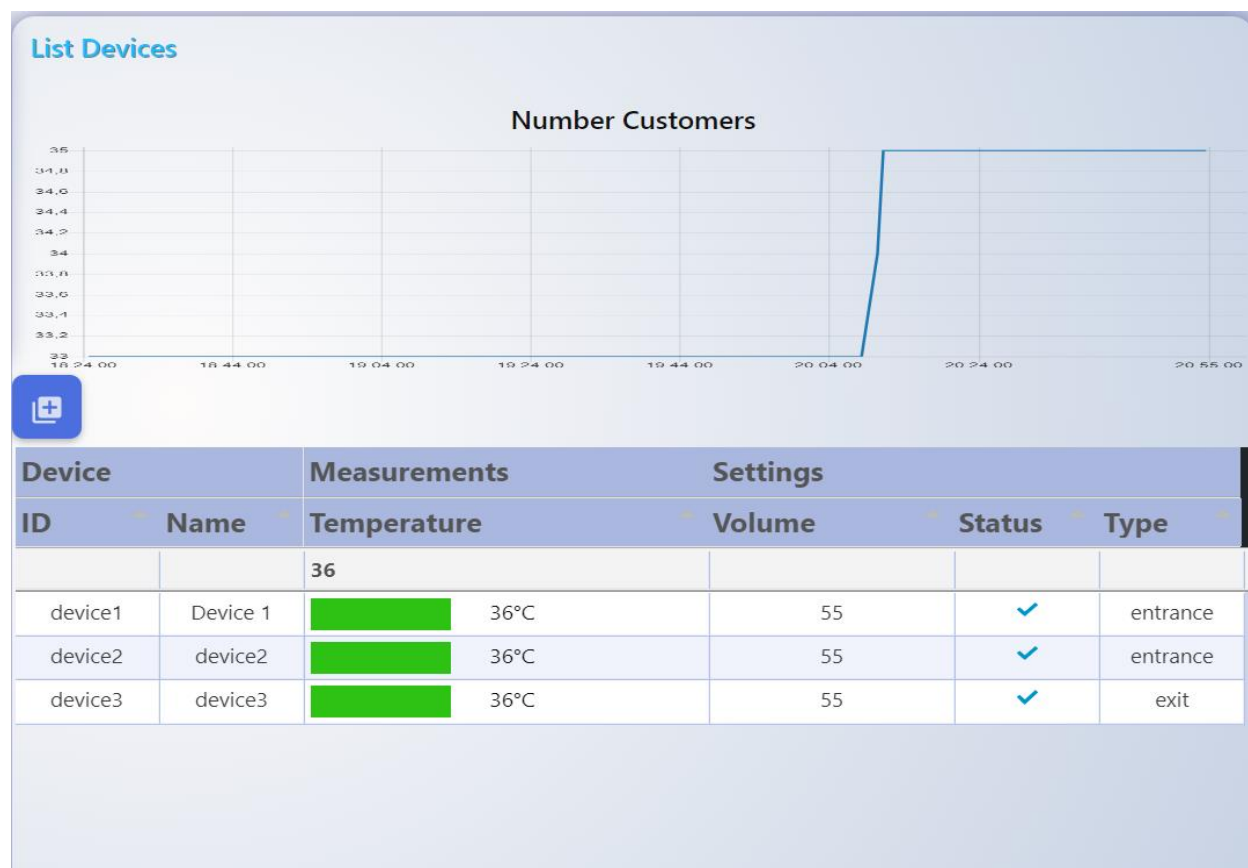
**Shop Info** hiển thị tên cửa hàng, số người tối đa và nút điều chỉnh cùng với gauge cho biết số người hiện có trong cửa hàng.

Bảng **Config** dùng để thay đổi nhiệt độ, Volume của 1 hoặc tất cả các thiết bị. Khi Config xong chọn **Edit**, thông tin thiết bị của cửa hàng đó sẽ được thay đổi.

Bất kỳ thông số nào được điều chỉnh sẽ đều cập nhật về database cũng như các table, các biểu đồ có liên quan.

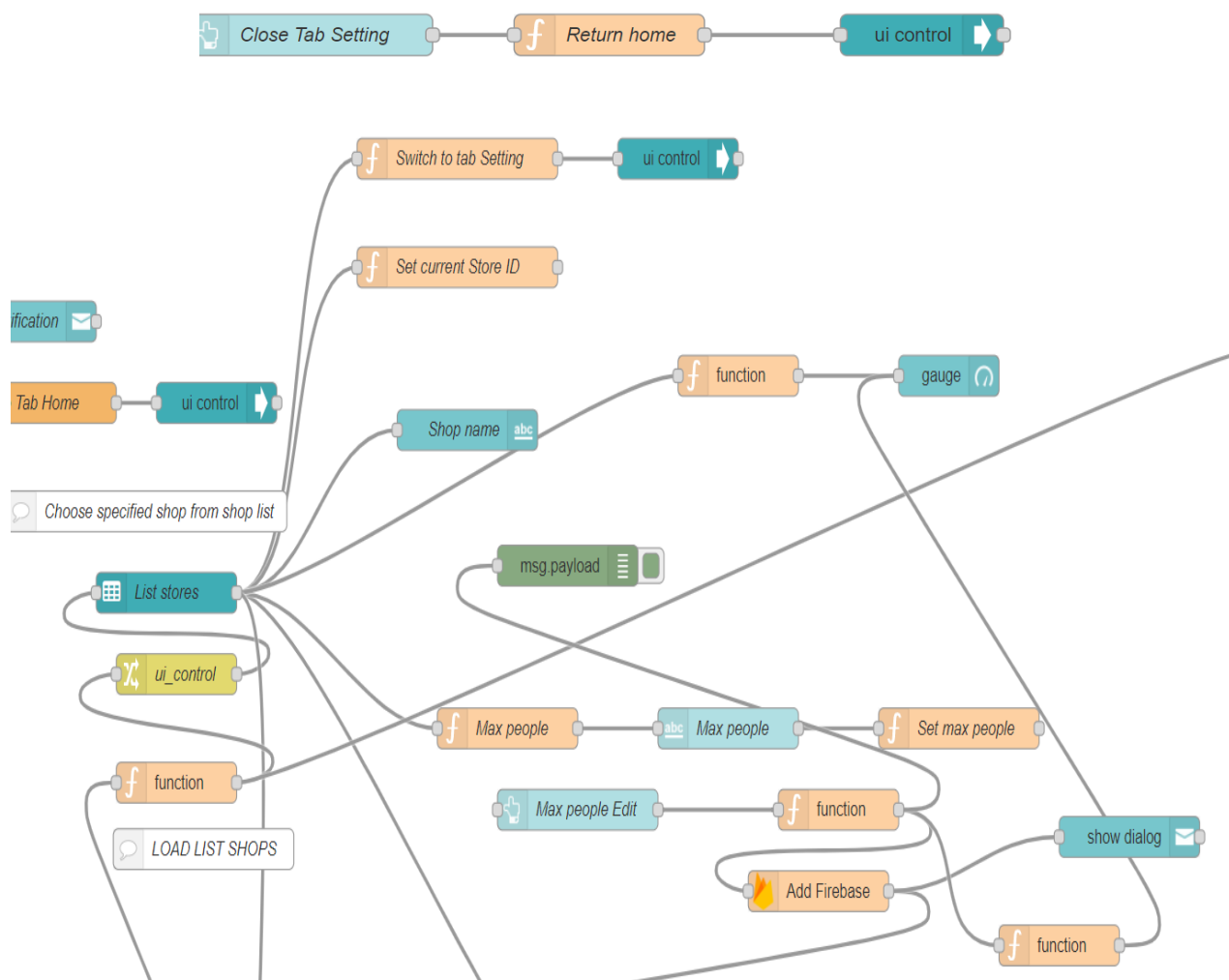
The Config interface displays two main settings: 'Celcius degree' set to 36 with up/down arrows, and 'Volume' represented by a slider bar currently at 0. Below these is a 'Device' dropdown menu and an 'All' checkbox. A large blue 'EDIT' button is at the bottom.

**List devices** hiển thị danh sách thiết bị và tình trạng thiết bị (như nhiệt độ cho phép, âm lượng, tình trạng hoạt động, chức năng (cửa vào, cửa ra)) được sử dụng ở cửa hàng. Biểu đồ trên cùng sẽ hiển thị số người qua từng thời điểm trong cửa hàng. Ở góc trái phía trên là button thêm một thiết bị vào danh sách.



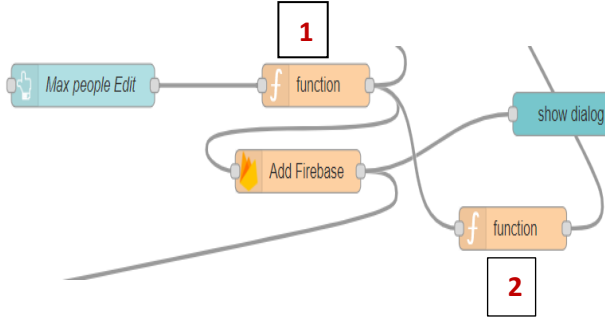


Nút close để đóng tab Setting trở về tab Home.

Phần flow bằng *Shop Info* được cấu hình như sau:



Node	Vai trò
	Chuyển từ tab Home (list store) sang tab Setting bằng cách cấu hình node function như sau <pre>msg.payload = {"tab":"Setting"}; return msg;</pre>
	Đưa shop_id được chọn trong bảng danh sách các store vào flow <pre>flow.set("currentStore", msg.payload.id); return msg;</pre>
	Hiện tên của store được chọn ở tab Setting

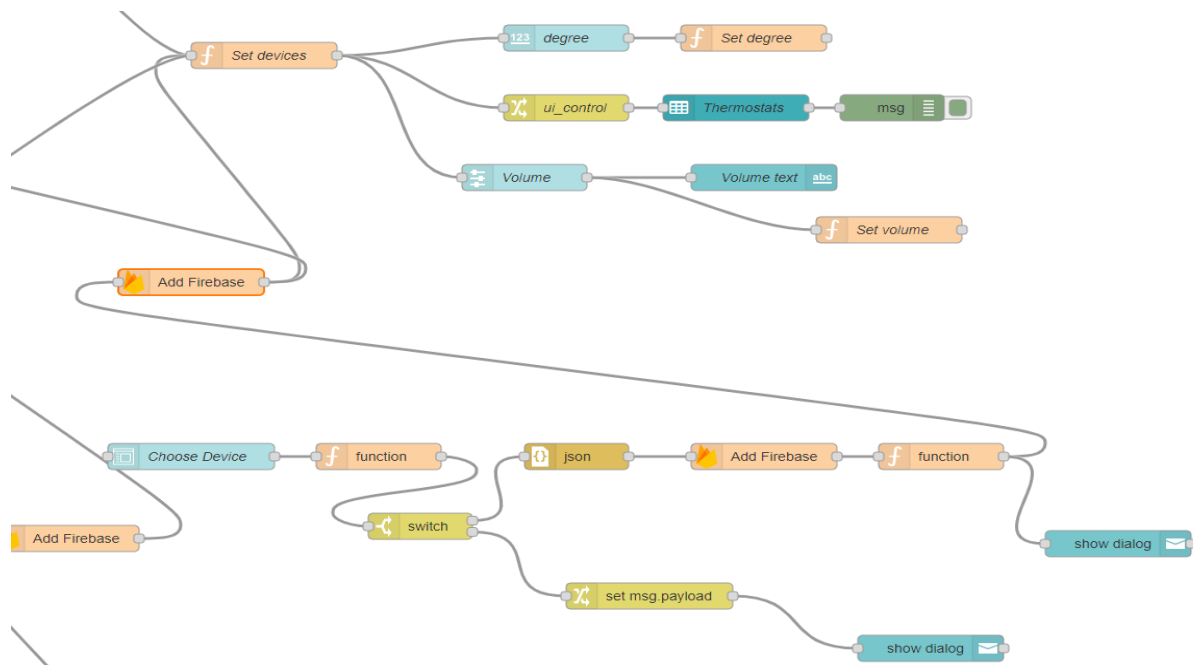
	<p>Node function <b>Max people</b> lấy số người tối đa từ bảng</p> <pre> 1 msg.payload = parseInt(msg.payload.maxPeople); 2 flow.set("maxPeople", msg.payload); 3 return msg; </pre> <p>Sau đó truyền vào node text để hiển thị. Đồng thời người dùng có thể chỉnh sửa số người tối đa ở node này và thông tin sẽ được đưa vào flow bằng node <b>Set max people</b></p> <pre> 1 flow.set("maxPeople", msg.payload); 2 return msg; </pre>
	<p>Node <b>function</b> sẽ cấu hình thông số min, max, và số khách hàng hiện tại lấy được từ bảng danh sách hiển thị lên <b>gauge</b>.</p> <pre> 1 min = 0; 2 max = parseInt(msg.payload.maxPeople); 3 data = msg.payload.num_customers; 4 msg = {payload:data,ui_control:{min:min,max:max},topic: ""}; 5 return msg; </pre>
	<p>Khi nhấn nút <b>Max people Edit</b>, số người tối đa sau khi được nhập sẽ được chuyển vào node function.</p> <p>Node <b>function1</b> có nhiệm vụ đưa <b>max_people</b> vào <b>msg.payload</b> và id của store vào <b>msg.dbPath</b>.</p> <pre> 1 msg.dbPath = "store/"+flow.get("currentStore"); 2 msg.payload = {"maxPeople": flow.get("maxPeople")}; 3 return msg; </pre> <p>Node <b>Add Firebase</b> cập nhật thông tin từ <b>function1</b> bằng phương thức <b>update</b> và hiện thông báo ra màn hình.</p> <p>Node <b>function2</b> sẽ phân tích lại thông tin về số người để hiện lại trong <b>gauge</b>.</p>


```


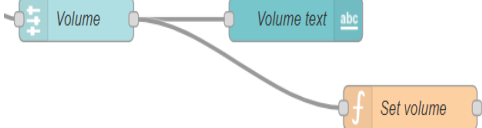

1 data = flow.get("storeData");
2 currentStore = flow.get("currentStore");
3
4 min = 0;
5 max = parseInt(msg.payload.maxPeople);
6 num = data[currentStore].num_customers;
7 msg = {payload:num,ui_control:{min:min,max:max},topic: ""};
8 return msg;
9

```

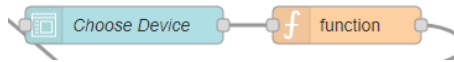
### Flow bảng Config và List device:



Node	Vai trò và cấu hình
	<p>Lưu thông tin của các thiết bị (device) có trong cửa hàng vào flow.</p> <pre> 1 flow.set("devices", msg.payload.devices); 2 devices = []; 3 4 for (device in msg.payload.devices) 5     devices.push(msg.payload.devices[device]); 6 7 msg.payload = devices; 8 return msg; </pre> <p>Đưa thông tin các device vừa được truyền tới vào mảng và đưa vào msg.payload để trả về.</p>

	<p>Hiển thị nhiệt độ của device đồng thời cho phép người dùng chỉnh lại nhiệt độ, sau đó truyền nhiệt độ vừa được chỉnh vào flow qua node <b>function</b></p> <pre> 1 flow.set("degree", msg.payload); 2 return msg; </pre>
	<p>Cho phép người dùng chỉnh độ lớn volume phát ra loa của thiết bị đồng thời hiển thị độ lớn đó.</p> <p>Node <b>function</b> sẽ lưu lại thông tin vừa chỉnh vào flow</p> <pre> 1 flow.set("volume", msg.payload); 2 return msg; </pre>
	<p>Node <b>change</b> hỗ trợ chuyển thông tin của các device về dạng <b>tabulator</b> để dễ dàng định dạng cho node <b>ui_table</b> với tên cột, giá trị của cột, độ dài, rộng của bảng,....</p> <pre> 1 { 2   "tabulator": { 3     "columns": [ 4       { 5         "formatterParams": { 6           "target": "_blank" 7         }, 8         "title": "Device", 9         "columns": [ 10          { 11            "formatterParams": { 12              "target": "_blank" 13            }, 14            "title": "ID", 15            "field": "id", 16            "width": 100, 17            "align": "center", 18            "color": "red" 19          }, 20          { 21            "formatterParams": { 22              "target": "_blank" 23            }, 24            "title": "Name", 25            "field": "name", 26            "width": 100, 27            "align": "center", </pre>





Node **Choose Device** cho phép người dùng chọn cấu hình cho 1 device hoặc cho tất cả device trong store.

Node **function** sẽ nhận lệnh từ node chọn trên, lưu thông tin vào biến **editDevices** và lấy ra thông tin các devices có sẵn trong flow.

```

1 msg.result = false;
2 editDevices = [msg.payload["Edit device"]] ;
3 devices = flow.get("devices");
  
```

Nếu người dùng chọn **All** thì lưu thông tin các device biến **editDevices**. Nếu chọn 1 device nhất định thì kiểm tra xem device được chọn có tồn tại hay không.

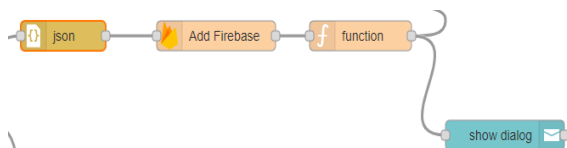
```

if (msg.payload.All)
  editDevices = Object.keys(devices);
else if (msg.payload["Edit device"] in devices == false){
  msg.payload = '';
  return msg;
}
  
```

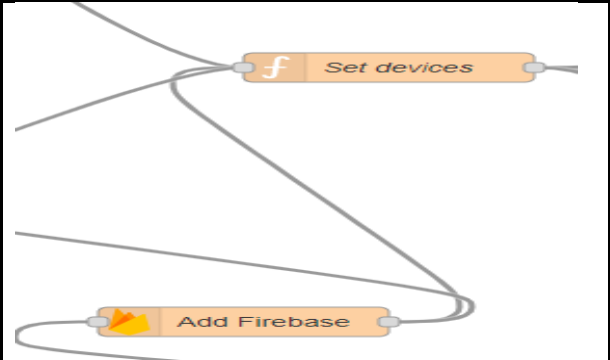
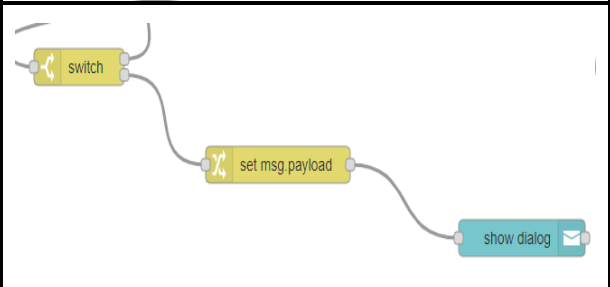
Cấu hình device theo thông số được nhập trước đó và lưu vào **msg.payload**, lấy id của store và device vào **msg.dbPath**.

```

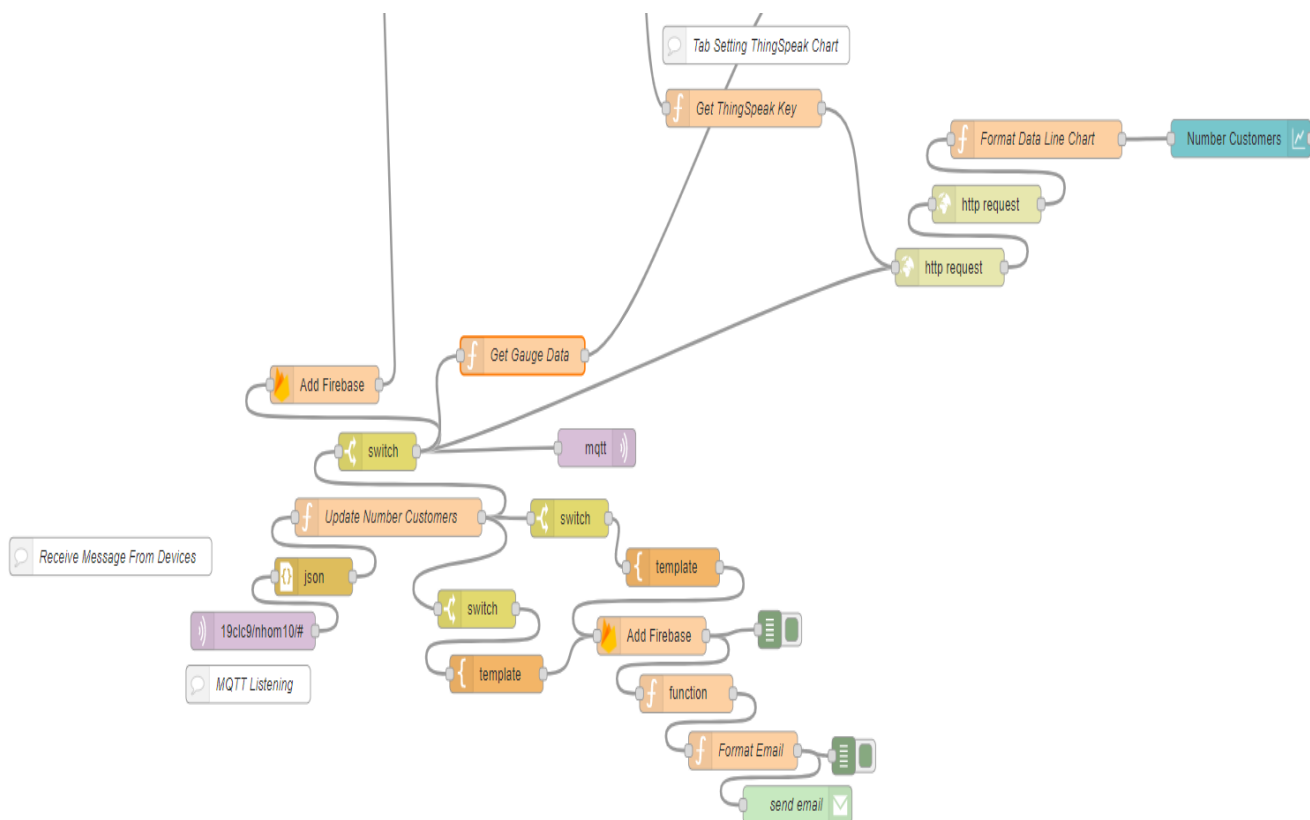
10 msg.payload = {};
11 msg.dbPath = "store/"+flow.get("currentStore")+"/devices";
12 for (let device of editDevices){
13   volume = flow.get("volume");
14   degree = flow.get("degree");
15   devices[device].df_volume = volume;
16   devices[device].df_temp = degree;
17   msg.payload[device] = devices[device];
18 }
19 msg.result = true;
20 return msg;
  
```

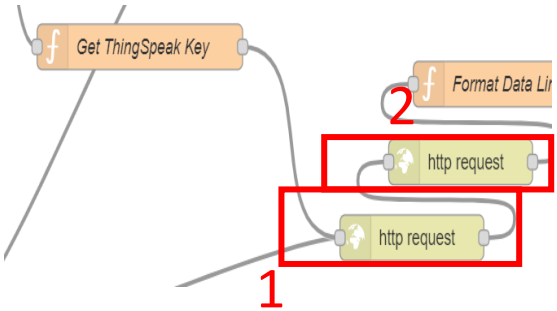
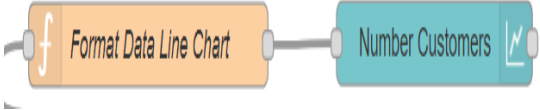


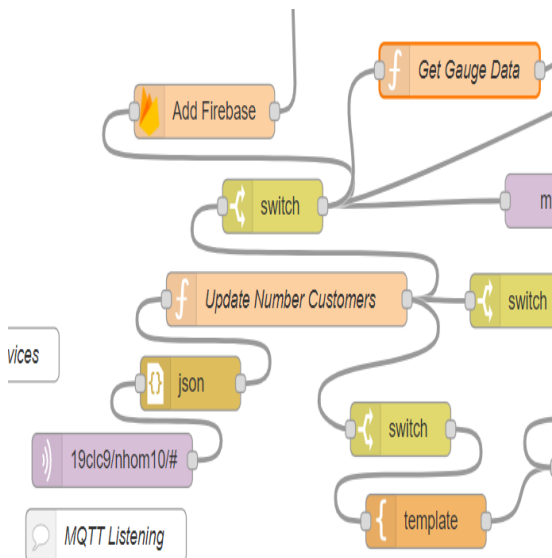
Update thông tin cần sửa nhận được từ node trên vào database và xuất ra thông báo đã sửa. Đồng thời cập nhật **msg.dbPath** bằng đường dẫn của store hiện tại.

	<p>Dựa theo đường dẫn vừa được cấu hình phía trên nhằm lấy lại data từ Firebase và truyền lại node <b><i>Set devices</i></b> như trên để cập nhật vào bảng danh sách các device của store.</p>
	<p>Nếu device được chọn không tồn tại thì xuất ra thông báo.</p>

### Flow phần đồ thị trong List device:



Node	Vai trò
	<p>Node <b>Get ThingSpeak Key</b> dùng để lấy ra thông tin <b>thingspeak</b> và số khách hàng của cửa hàng được nhập chọn từ danh sách cửa hàng ở tab <b>Setting</b>.</p> <pre> 1 msg.writeKey = msg.payload.thingspeak.writeKey; 2 msg.readKey = msg.payload.thingspeak.readKey; 3 msg.id = msg.payload.thingspeak.id; 4 msg.payload = {"num_customers" : msg.payload.num_customers}; 5 return msg; </pre> <p>2 node <b>http request</b> kết nối với <b>thingspeak</b>:</p> <ul style="list-style-type: none"> <li>- node <b>1</b> dùng để viết kết quả số người lên <b>thingspeak</b> được lấy từ node <b>function</b>.</li> <li>- node <b>2</b> dùng để đọc 10 kết quả gần nhất.</li> </ul>
	<p>Phân tích các thông tin được lấy từ <b>Thingspeak</b> cho vào mảng <b>data</b></p> <pre> 1 data = []; 2 for (let point of msg.payload.feeds){ 3   time = Date.parse(point.created_at.replace('T',' ')); 4   // time = point.created_at 5   // time = time.substring(time.indexOf('T')+1, time.length); 6   data.push({'x': time, 'y': parseFloat(point.field1)}); 7 } </pre> <p>Cấu hình thông tin để hiển thị lên <b>ui_chart</b></p> <pre> 10 msg.payload = { 11   "series": ["temp"], 12   "data": [data], 13   "labels": ["temp"] 14 }; 15 return msg; </pre>



Nhóm sử dụng **MQTT** với server public **test.mosquitto.org port 1883**.

Node MQTT in sẽ nhận thông tin từ thiết bị của nhóm, chuyển dữ liệu thành **data json**

Node **Update Number Customers** sẽ cập nhật số người được gửi đến để update lên **Firestore** cũng như **gauge**.

Kiểm tra **store** chứa **device** có tồn tại hay không. Nếu không thì trả về **False**.

```
5 if ( device.storeId in data == false || "devices" in data[device.storeId] == false){
7   return msg;
3 }
```

Kiểm tra **device** có tồn tại hay không. Nếu không thì trả về **False**.

```
devices = data[device.storeId].devices;
if (device.deviceId in devices == false)
  return msg;
```

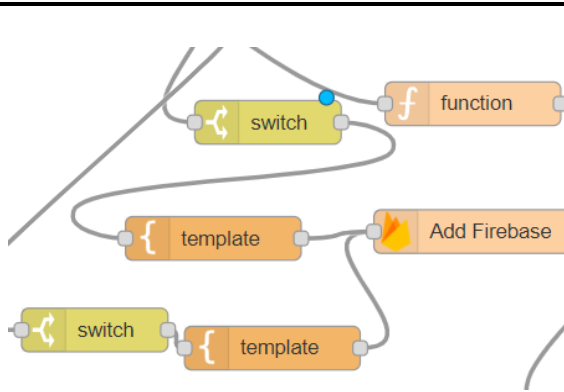
Nếu device ở lối vào (**entrance**) thì cộng 1,

Lối ra (**exit**) thì trừ 1. Nếu số lượng người hoặc nhiệt độ vượt quá ngưỡng cho phép thì gán **True**.

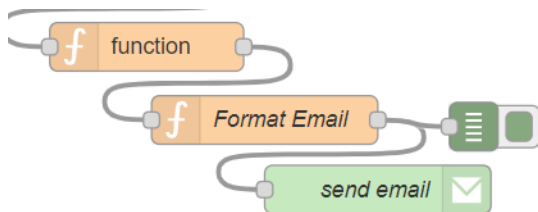
```
13 currentPeople = data[device.storeId].num_customers;
14 if (device.type == "entrance"){
15   currentPeople += 1;
16   if (currentPeople >= data[device.storeId].maxPeople)
17     msg.overPeople = true;
18   if (device.temp > device.maxTemp){
19     msg.overTemp = true;
20     msg.temp = device.temp;
21   }
22 }
23 else if (currentPeople > 0) {
24   currentPeople -=1;
25 }
```

Thiết lập các thông tin cần thiết để gửi đi cho database, thinkspeak, server MQTT.

```
26 msg.result = true;
27 msg.dbPath = "store/"+device.storeId;
28 msg.payload = {"num_customers" : currentPeople};
29 msg.topic = data[device.storeId].mqtt.topic;
30 msg.writeKey = data[device.storeId].thingspeak.writeKey;
31 msg.readKey = data[device.storeId].thingspeak.readKey;
32 msg.id = data[device.storeId].thingspeak.id;
33 msg.storeId = device.storeId;
34 msg.maxPeople = data[device.storeId].maxPeople;
35 msg.num_customers = currentPeople;
36 return msg;
```



Cấu hình message để chuẩn bị gửi thông báo cho người dùng qua email nếu nhiệt độ và số người vượt ngưỡng (True).  
Lấy thông tin email từ Firebase.



Node **function** đưa các email nhận được vào mảng để trả về.

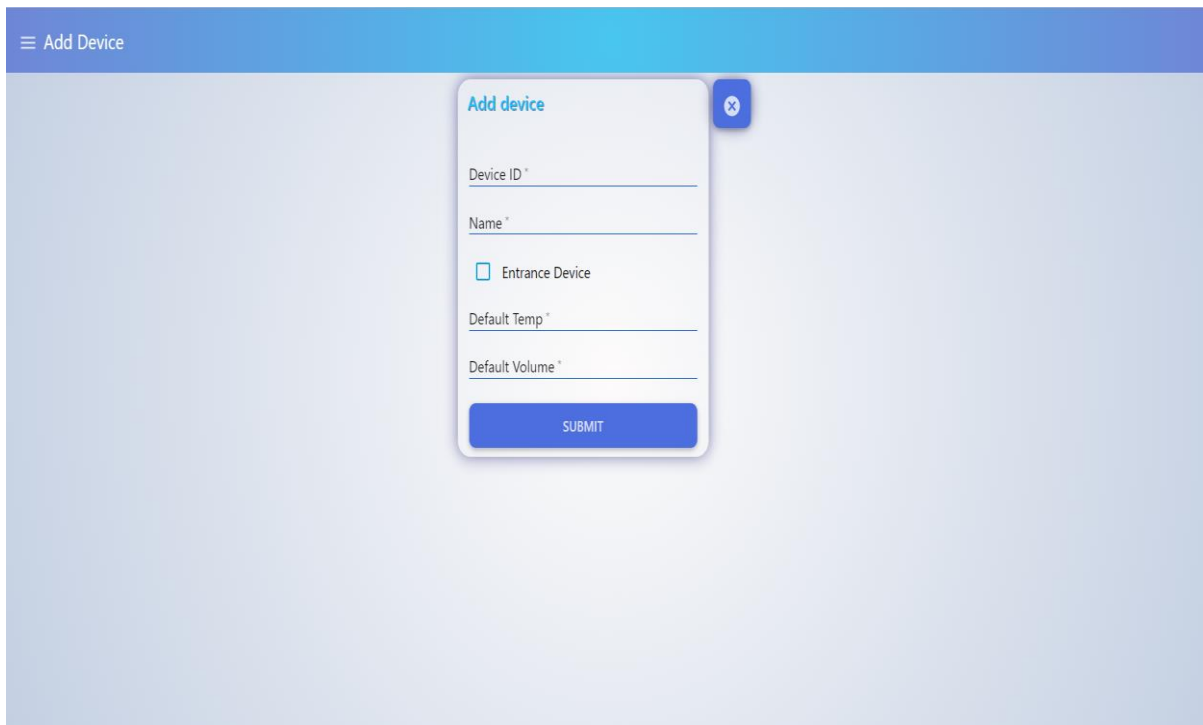
```
1 emails = [];
2 accounts = msg.payload.admins;
3 for (let account in accounts){
4     emails.push(accounts[account].email);
5 }
6 msg.emails = emails;
7 return msg;
```

Node **format email** thiết lập thông tin gửi nhận để gửi email qua node **send email**.

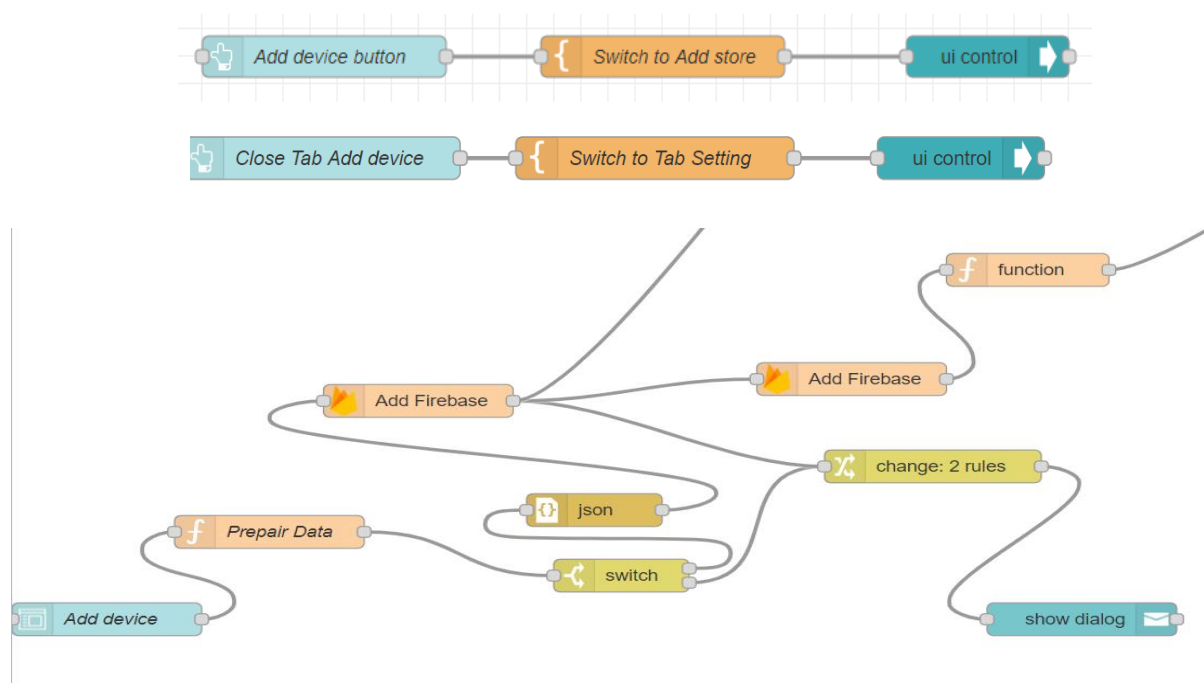
```
1
2 msg={
3     payload: msg.content,
4     topic: "Store Managemet Alert",
5     to: msg.emails
6 };
7
8
9 return msg;
```



### c. Add device

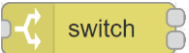
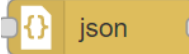
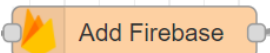
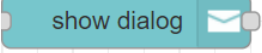
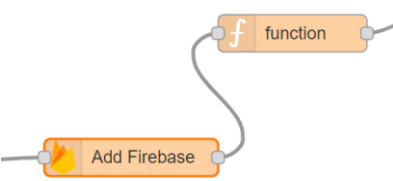
Khi chọn button {+} ở tab Setting của cửa hàng, người dùng sẽ chuyển tới tab **Add device** để thêm một thiết bị mới danh sách trên database của cửa hàng. Đồng thời sẽ có một button đóng để đóng tab trở về tab Setting.



Phần Flow được cấu hình như sau:



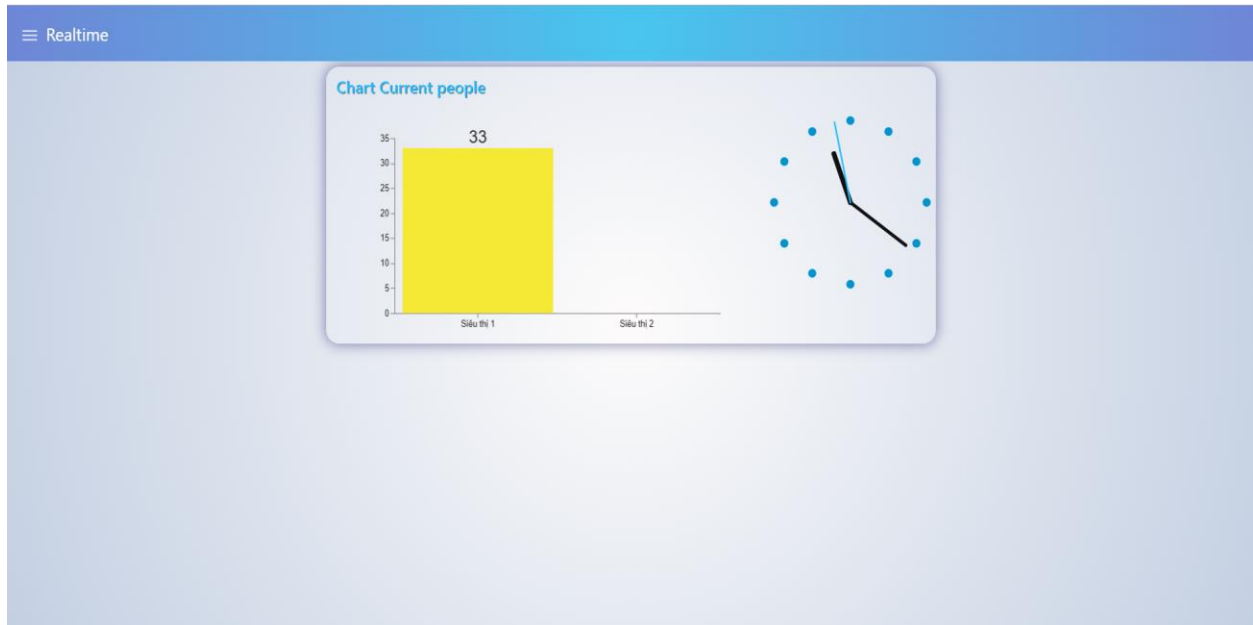
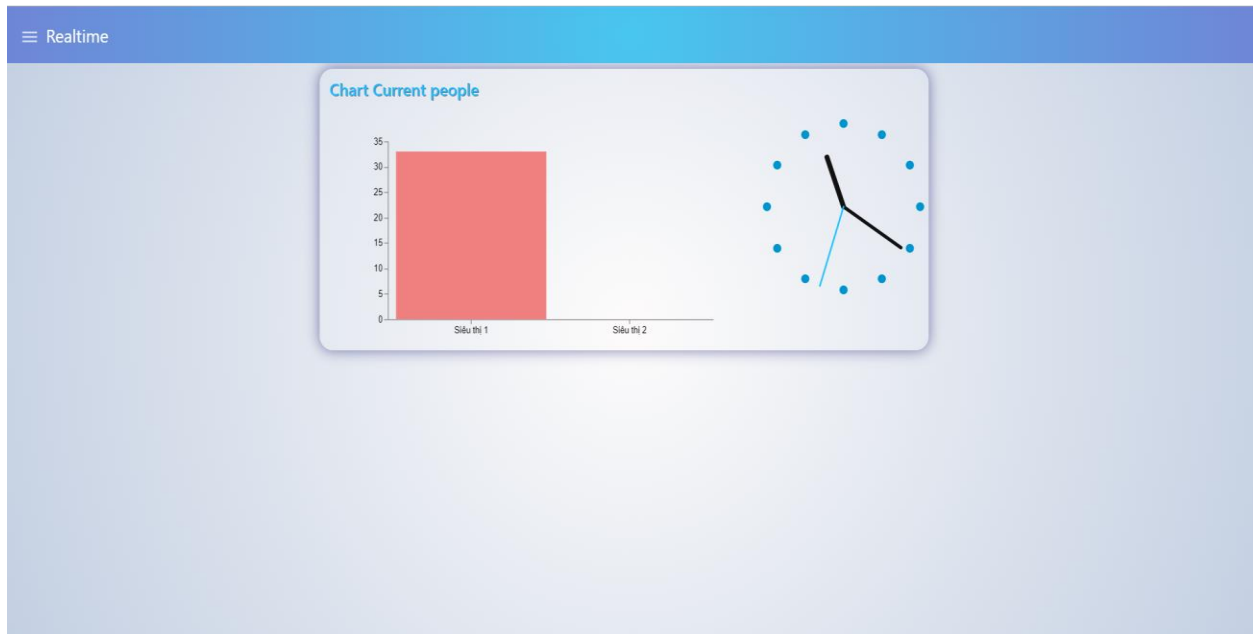
Node	Vai trò
 Add device	<p>Node ui_form cho phép người dùng nhập thông tin device cần thêm như id, name, nhiệt độ, volume,...</p>
 Prepair Data	<p>Lấy thông tin các device của store trong flow , thông tin của device cần thêm từ <i>msg.payload</i> và đường dẫn trong database của store hiện tại.</p> <pre> 1 data = flow.get("storeData"); 2 store = flow.get("currentStore").replace("/store/", ""); 3 newDevice = msg.payload; 4 msg.dbPath = "store/"+store </pre> <p>Nếu device chưa tồn tại trong danh sách thì tạo một mảng trống để sử dụng lưu thông tin mới. Ngược lại thì trả về thông báo device đã tồn tại rồi.</p> <pre> if ("devices" in data[store] == false){     msg.payload = {"devices": []}; } else if (newDevice.id in data[store]["devices"] == true){     msg.result = false;     msg.topic = "Device ID Exists!";     return msg; } </pre> <p>Thiết lập thông tin của device mới.</p> <pre> device = {     "id": newDevice.id,     "name": newDevice.name,     "status": true } </pre> <p>Kiểm tra xem device được sử dụng cho lối vào <i>entrance</i> (cần có các thông tin về volume và nhiệt độ) hay lối ra <i>exit</i>.</p> <pre> 19 if (newDevice.type == true){ 20     type = "entrance"; 21     device["df_volume"] = newDevice.df_volume; 22     device["df_temp"] = newDevice.df_temp; 23 } 24 else type = "exit"; 25 device['type'] = type; </pre> <p>Thiết lập đường dẫn <i>msg.dbPath</i> và thông báo đã thêm thành công.</p> <pre> 26 if ("devices" in msg.payload){ 27     msg.payload["devices"] = { 28         [newDevice.id]: device 29     }; 30 } 31 else { 32     msg.payload = { 33         [newDevice.id]: device 34     }; 35     msg.dbPath = "store/"+store+"/devices" 36 } 37 msg.result = true; 38 msg.topic = "Add device "+newDevice.name+" Success"; 40 return msg; </pre>

 switch	<p>Nếu device đã tồn tại thì chuyển đến node thông báo, ngược lại chuyển đến node firebase</p>
 json	<p>Chuyển đổi giữa chuỗi json và object</p>
 Add Firebase	<p>Cập nhật thông tin của device mới vào database bằng phương thức <b>update</b> qua đường dẫn <b>msg.dbPath</b></p>
 show dialog	<p>Thông báo device đã tồn tại hoặc đã được tạo thành công</p>
	<p>Sau khi đã thêm vào database, cần lấy lại thông tin của store bằng node <b>Add Firebase</b> và đưa thông tin vào store đang sửa bằng :</p> <pre> 1 msg.payload = msg.payload[flow.get("currentStore")]; 2 return msg; </pre>

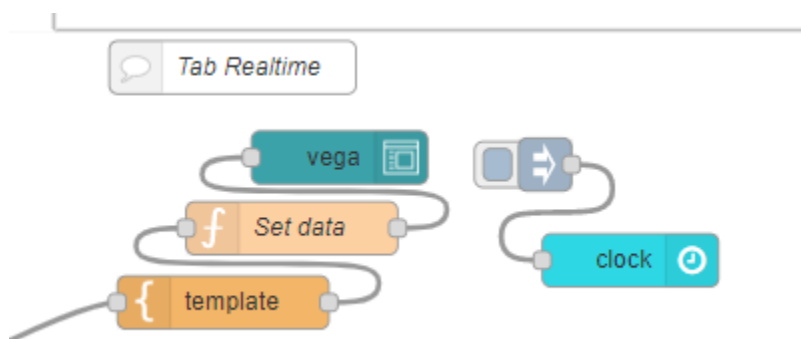


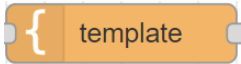
### 3. Realtime


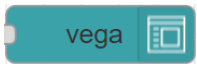
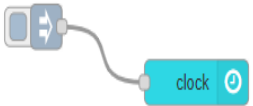
Chọn tab Realtime ở menu, tab này sẽ lấy dữ liệu từ Firebase và hiển thị số lượng người ở thời gian thực tại các cửa hàng. Ngoài ra còn có thêm một chiếc đồng hồ cho biết thời gian hiện tại giúp người dùng dễ dàng quan sát hơn.



Flow được cấu hình như sau:

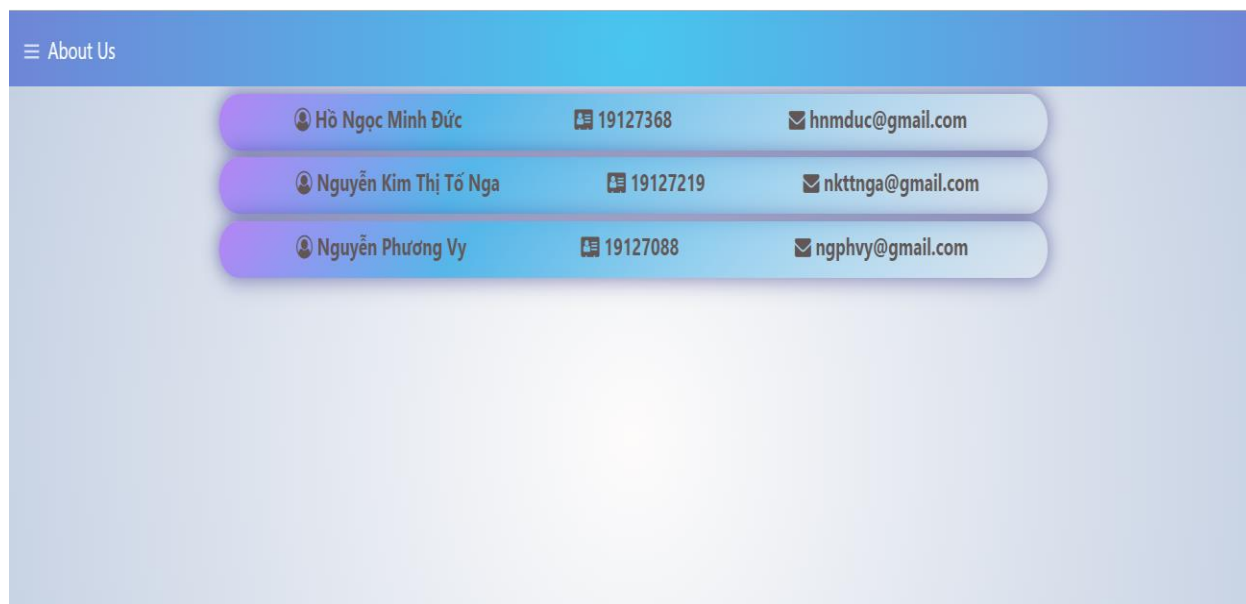


Node	Vai trò
	<p>Tùy chỉnh hiển thị của biểu đồ như độ dài rộng, hiển thị số trên mỗi cột khi rê chuột tới (<i>signal</i>), nội dung các cột (<i>scales</i>), màu sắc hoặc hiển thị chữ(<i>marks</i>),.....</p> <pre> 1 { 2   "width": 400, 3   "height": 200, 4   "padding": 5, 5 6   "data": [ 7     { 8       "name": "table", 9       "values": [ 10 11     ] 12   } 13 ], 14 15 "signals": [ 16   { 17     "name": "tooltip", 18     "value": {}, 19     "on": [ 20       { "events": "rect:mouseover", "update": "datum" }, 21       { "events": "rect:mouseout", "update": "{}" } 22     ] 23   } 24 ], 25 26 "scales": [ 27   { 28     "name": "xscale", 29     "type": "band", 30     "domain": { "data": "table", "field": "shop" }, 31     "range": "width", 32     "padding": 0.05, 33     "round": true 34   } 35 ] 36 }</pre>

	<p>Lấy dữ liệu về cửa hàng trên flow qua <i>storeData</i> đã được thêm trước đó, thêm vào mảng data mới với <i>shop</i> là tên store, <i>amount</i> là số lượng người hiện tại. Gán <i>msg.payload.data[0].values</i> bằng mảng data để hiển thị lên <i>node vega</i>.</p> <pre> 1 data = []; 2 storeData = flow.get("storeData"); 3 for (store in storeData){ 4   data.push({ 5     "shop": storeData[store].name, 6     "amount": storeData[store].num_customers 7   }) 8 } 9 msg.payload.data[0].values = data; 10 return msg; </pre>
	<p>Cài đặt thư viện <i>node-red-node-ui-vega</i> để sử dụng node vega hiển thị dữ liệu dưới dạng biểu đồ.</p>
	<p>Cài đặt thư viện <i>node-red-contrib-ui-clock</i> để hiển thị đồng hồ.</p>

## 4. About Us

Tab About Us ở Menu sẽ hiển thị thông tin của nhóm, của sản phẩm, lượt sử dụng, đánh giá và hình ảnh sản phẩm.



\*\*\*