DEPT OF COMPUTER AND COMMUNICATION ENGINEERING

Internet of Things: Foundations and Applications Lab

MMH: ITFL316064E

Group: Nguyễn Văn Quang Huy– 19119030

Lê Bá Minh Đạt– 19119049

Introduction

The ESP8266 serves as the system's primary processor and the project's objective is to construct an IoT system for smart home with all of the necessary characteristics. The system created for this project enables the user to view the temperature, humidity, and temperature of gas and receive notifications through the interface of a web page using data obtained from sensors and transmitted via the ESP8266 to Google Firebase, which is connected to the aforementioned website. The solution additionally offers a web interface for managing the connected ESP8266 gadgets.The whole report is divided into 4 parts:

1. Build a website.

2. Create Google Firebase.

3. Firebase link with web interface.

4. Hardware preparation.

5. Project Implementation.

1. Build a website

1.1 File HTML

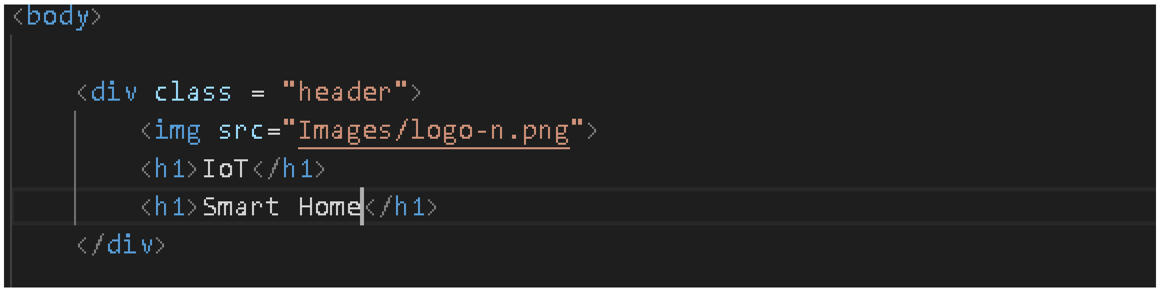
To build a website, we first create a file with the extension html, specifically here index.html.

```
index.html          12/18/2022 5:18 PM          Chrome HTML Do...          5 KB
```

Next we will build a web interface with 3 parts: Header, Content and finally Footer:

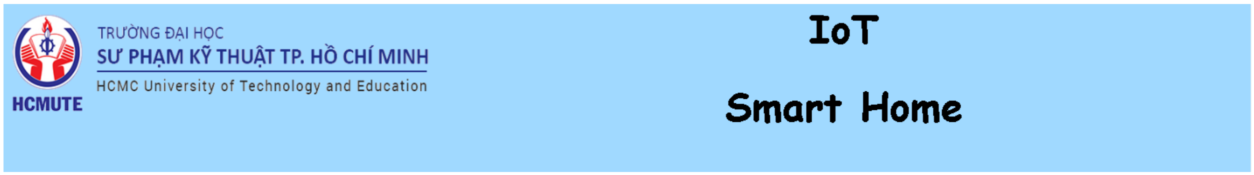
❖ Firstly, starts with the Header:

This section will include the school's logo and 2 texts with the content IOT and Smart Home.

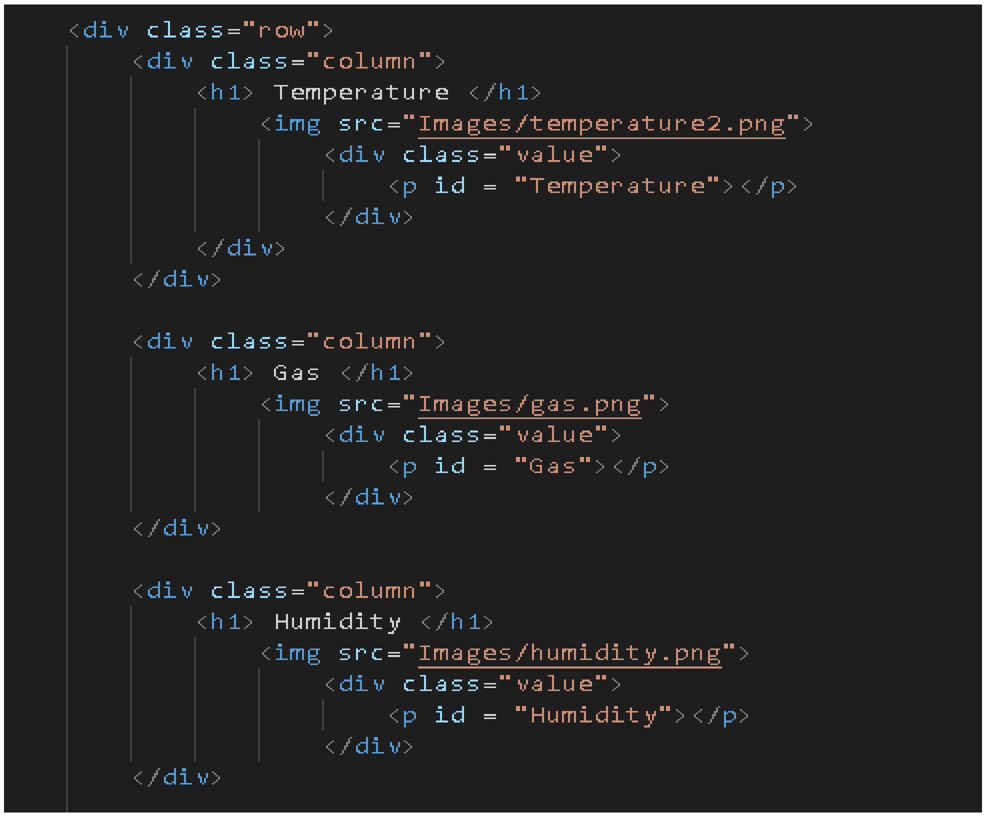


```
<body>

    <div class = "header">
        <img src="Images/logo-n.png">
        <h1>IoT</h1>
        <h1>Smart Home</h1>
    </div>
```

The result we get on the web interface will be:


TRƯỜNG ĐẠI HỌC
SƯ PHẠM KỸ THUẬT TP. HỒ CHÍ MINH
HCMC University of Technology and Education
HCMUTE
IoT
Smart Home

❖ Secondly, the Content section:

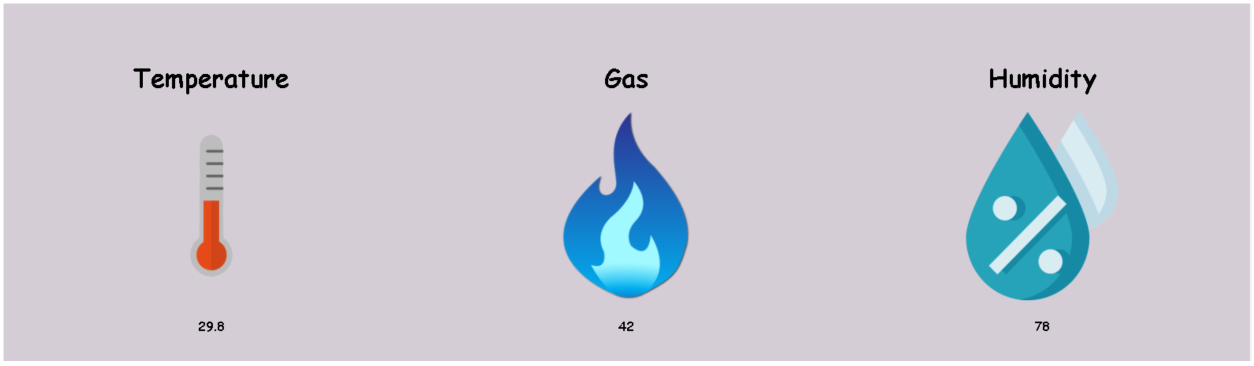


```
<div class="row">
    <div class="column">
        <h1> Temperature </h1>
            <img src="Images/temperature2.png">
                <div class="value">
                    <p id = "Temperature"></p>
                </div>
        </div>
    </div>

    <div class="column">
        <h1> Gas </h1>
            <img src="Images/gas.png">
                <div class="value">
                    <p id = "Gas"></p>
                </div>
    </div>

    <div class="column">
        <h1> Humidity </h1>
            <img src="Images/humidity.png">
                <div class="value">
                    <p id = "Humidity"></p>
                </div>
    </div>
```

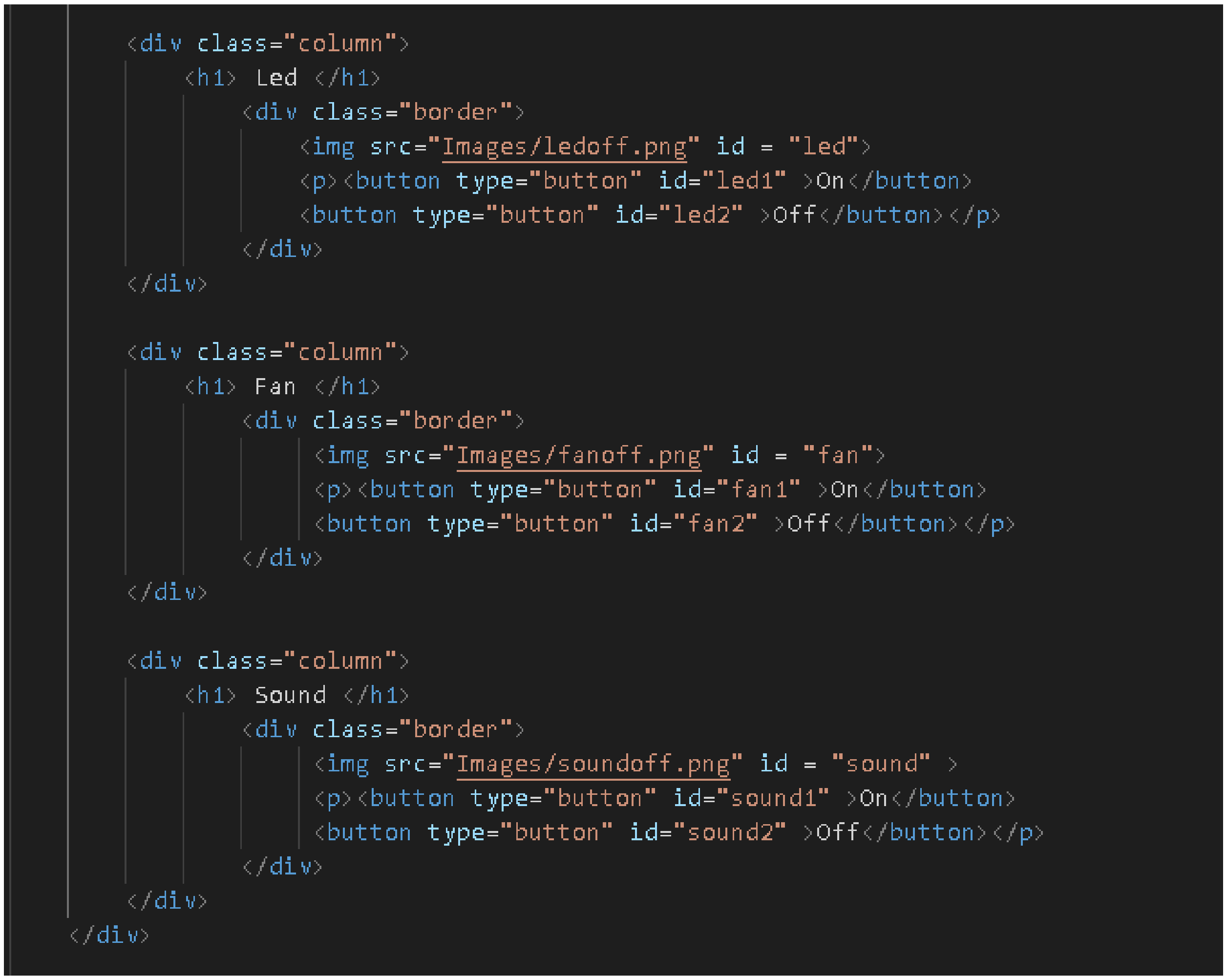
Here we create class row to contain class column inside it.

In these class columns will contain the images and names of the sensors that we use in this project.

The result we get on the web interface will be:



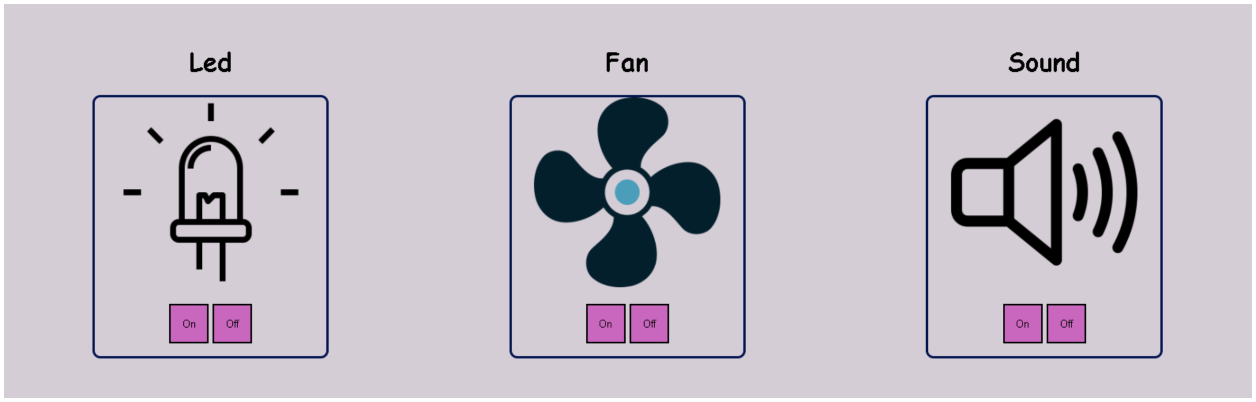
The parameters below the figure are the parameters that we will get through the connection between firebase and the web interface.



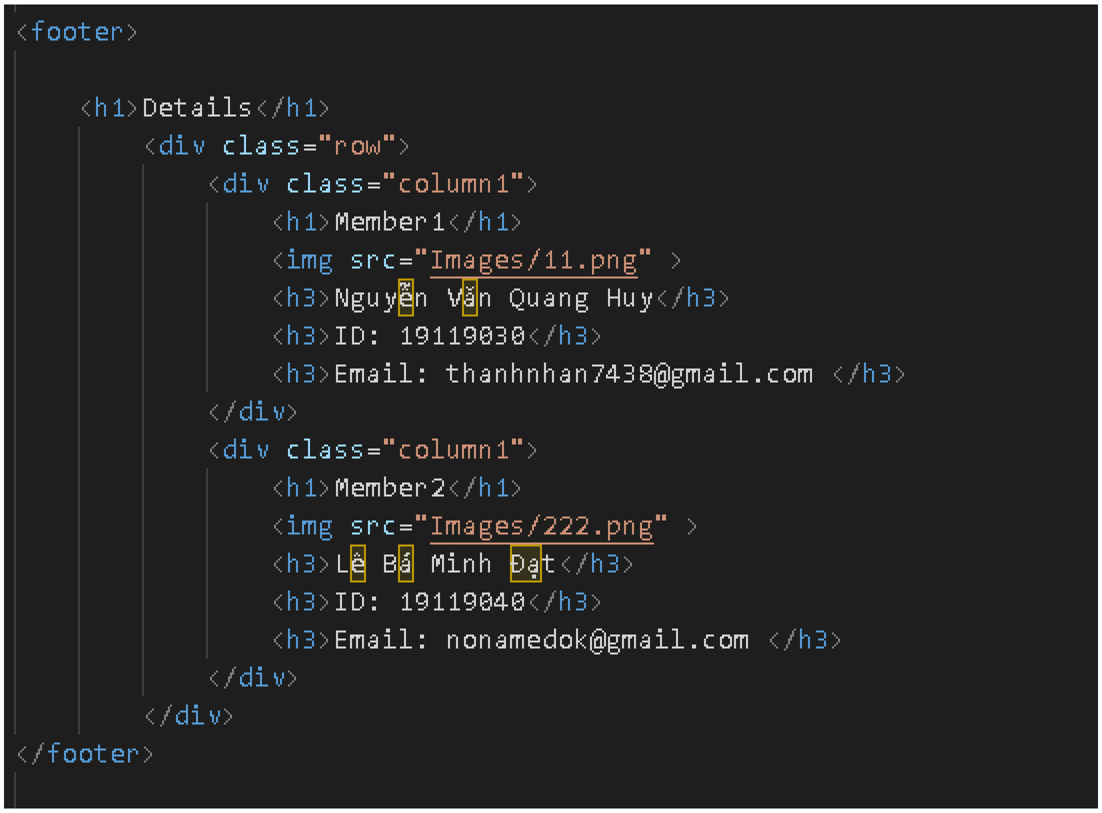
```
    <div class="column">
        <h1> Led </h1>
            <div class="border">
                <img src="Images/ledoff.png" id = "led">
                <p><button type="button" id="led1" >On</button>
                <button type="button" id="led2" >Off</button></p>
            </div>
    </div>

    <div class="column">
        <h1> Fan </h1>
            <div class="border">
                <img src="Images/fanoff.png" id = "fan">
                <p><button type="button" id="fan1" >On</button>
                <button type="button" id="fan2" >Off</button></p>
            </div>
    </div>

    <div class="column">
        <h1> Sound </h1>
            <div class="border">
                <img src="Images/soundoff.png" id = "sound" >
                <p><button type="button" id="sound1" >On</button>
                <button type="button" id="sound2" >Off</button></p>
            </div>
    </div>
</div>
```

Here, this section will show the user information and images of the control device. In addition, the user can interact with it through the button on it.

The result we get on the web interface will be:



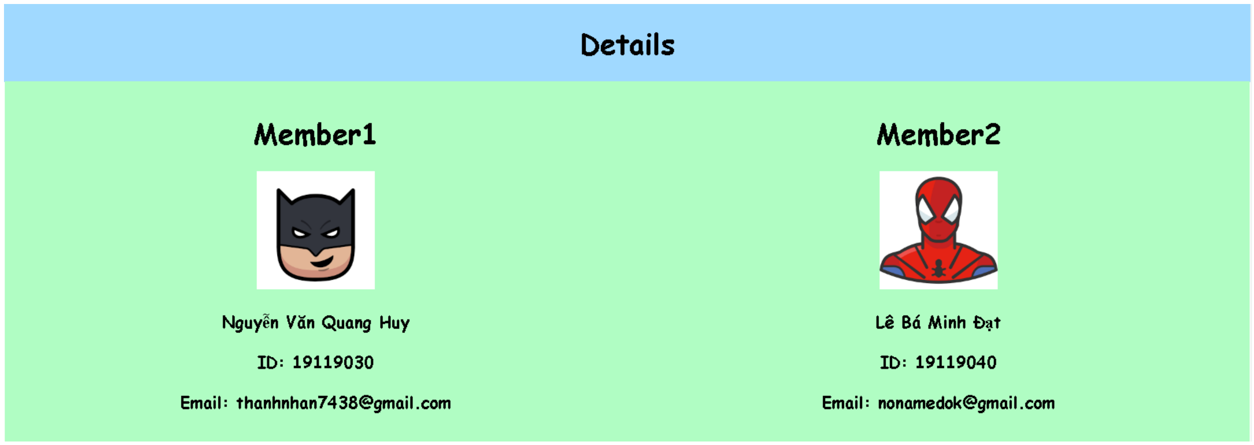
❖ finally the Footer section:



```
<footer>

    <h1>Details</h1>
        <div class="row">
            <div class="column1">
                <h1>Member1</h1>
                <img src="Images/11.png" >
                <h3>Nguyễn Văn Quang Huy</h3>
                <h3>ID: 19119030</h3>
                <h3>Email: thanhnhan7438@gmail.com </h3>
            </div>
            <div class="column1">
                <h1>Member2</h1>
                <img src="Images/222.png" >
                <h3>Lê Bá Minh Đạt</h3>
                <h3>ID: 19119040</h3>
                <h3>Email: nonamedok@gmail.com </h3>
            </div>
        </div>
</footer>
```

This section will contain the personal information of each member.

The result we get on the web interface will be:



Details

Member1

Nguyễn Văn Quang Huy

ID: 19119030

Email: thanhnhan7438@gmail.com

Member2

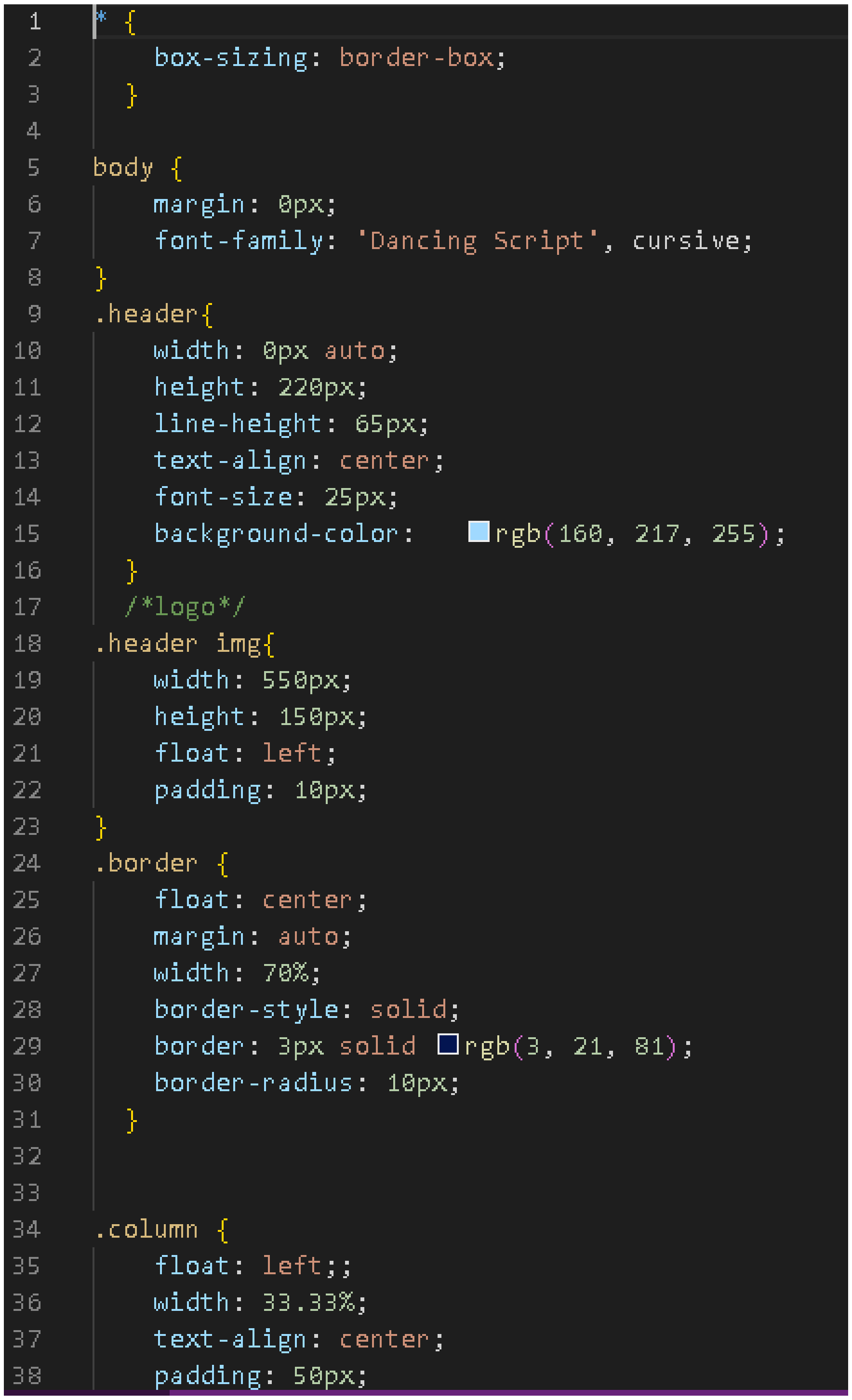
Lê Bá Minh Đạt

ID: 19119040

Email: nonamedok@gmail.com

1.2 File CSS

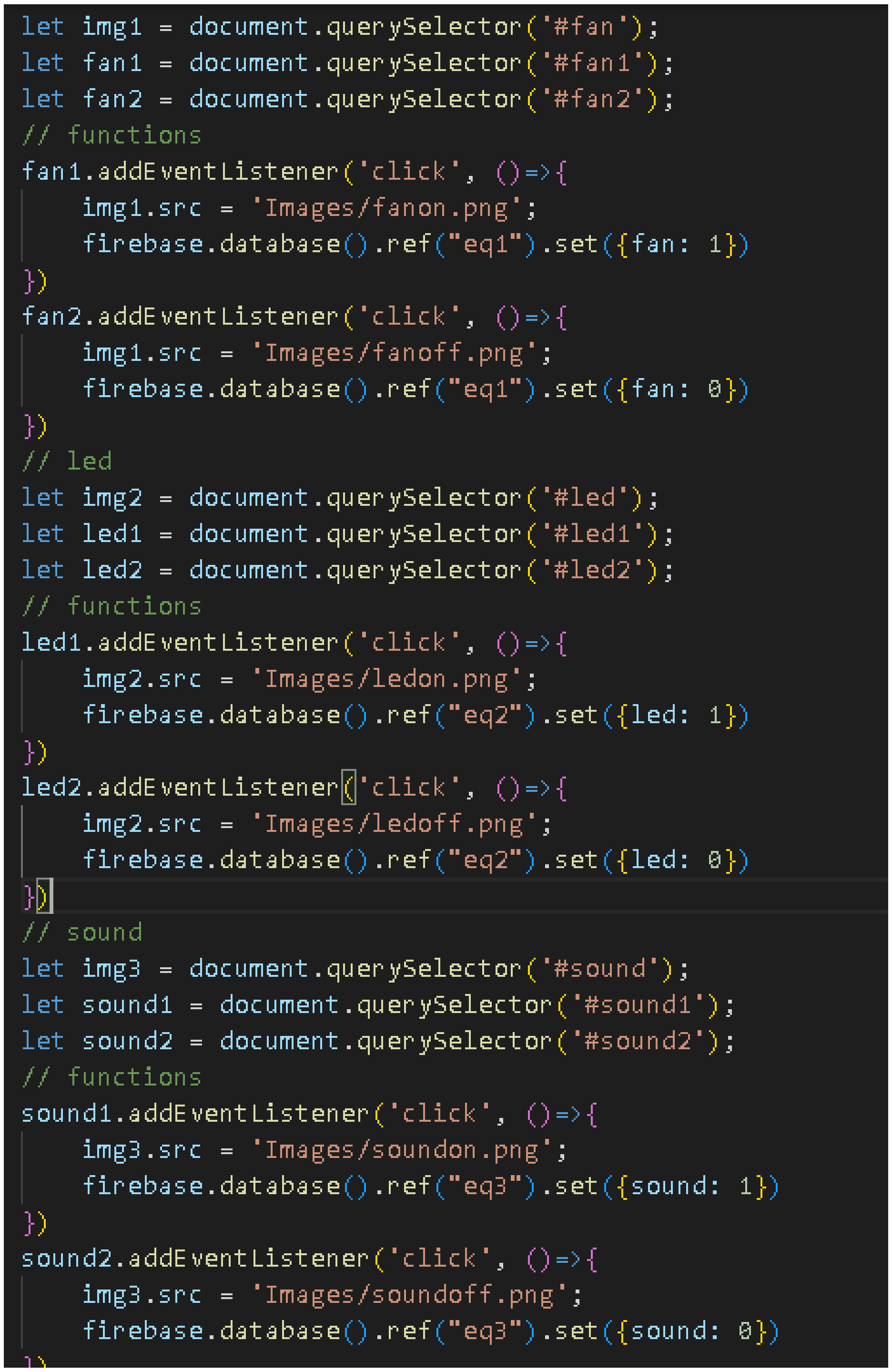
In addition, we will create a css file to decorate the web interface that we have just created.



```
* {
    box-sizing: border-box;
  }

body {
    margin: 0px;
    font-family: 'Dancing Script', cursive;
}
.header{
    width: 0px auto;
    height: 220px;
    line-height: 65px;
    text-align: center;
    font-size: 25px;
    background-color:   rgb(160, 217, 255);
  }
  /*logo*/
.header img{
    width: 550px;
    height: 150px;
    float: left;
    padding: 10px;
}
.border {
    float: center;
    margin: auto;
    width: 70%;
    border-style: solid;
    border: 3px solid rgb(3, 21, 81);
    border-radius: 10px;
  }


.column {
    float: left;;
    width: 33.33%;
    text-align: center;
    padding: 50px;
```

1.3 File Junction

This juntion.js file was created with the purpose of helping users to interact with the web interface through the push of a button.



```
let img1 = document.querySelector('#fan');
let fan1 = document.querySelector('#fan1');
let fan2 = document.querySelector('#fan2');
// functions
fan1.addEventListener('click', ()=>{
    img1.src = 'Images/fanon.png';
    firebase.database().ref("eq1").set({fan: 1})
})
fan2.addEventListener('click', ()=>{
    img1.src = 'Images/fanoff.png';
    firebase.database().ref("eq1").set({fan: 0})
})
// led
let img2 = document.querySelector('#led');
let led1 = document.querySelector('#led1');
let led2 = document.querySelector('#led2');
// functions
led1.addEventListener('click', ()=>{
    img2.src = 'Images/ledon.png';
    firebase.database().ref("eq2").set({led: 1})
})
led2.addEventListener('click', ()=>{
    img2.src = 'Images/ledoff.png';
    firebase.database().ref("eq2").set({led: 0})
})
// sound
let img3 = document.querySelector('#sound');
let sound1 = document.querySelector('#sound1');
let sound2 = document.querySelector('#sound2');
// functions
sound1.addEventListener('click', ()=>{
    img3.src = 'Images/soundon.png';
    firebase.database().ref("eq3").set({sound: 1})
})
sound2.addEventListener('click', ()=>{
    img3.src = 'Images/soundoff.png';
    firebase.database().ref("eq3").set({sound: 0})
```

These commands will change the image on the web interface every time we press On or Off and also change the logic value of the control device on firebase.

2. Firebase

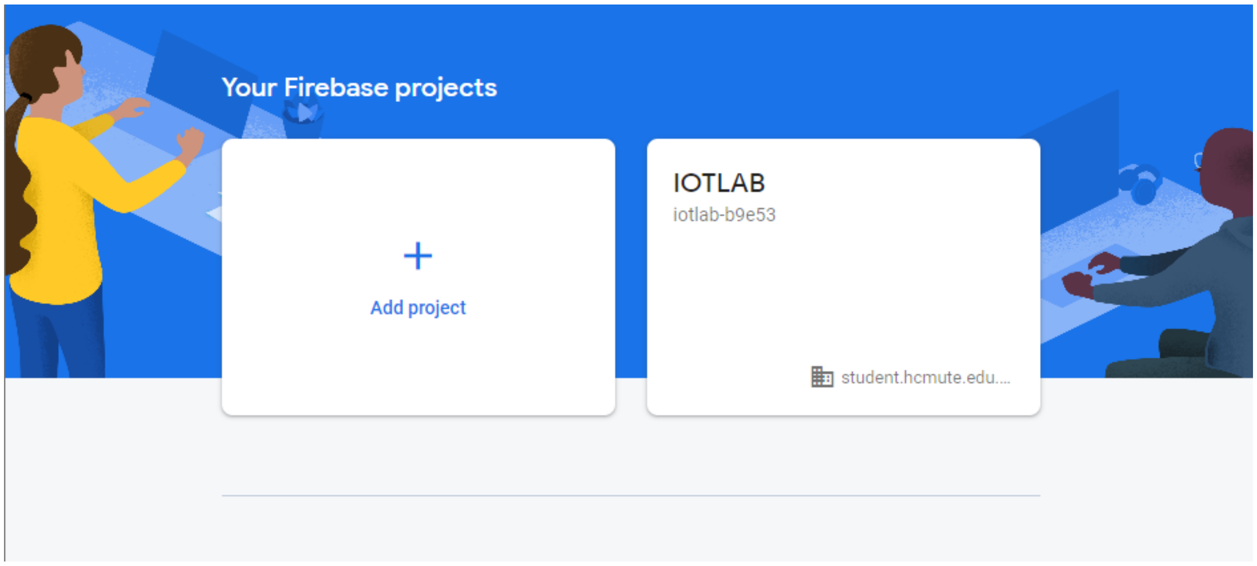
Firebase is a cloud-based database service. Accompanied by an extremely powerful server system of Google. Its main function is to help users program applications by simplifying database operations.

Specifically, simple API application programming interfaces. The aim is to increase the number of users and get more profits.

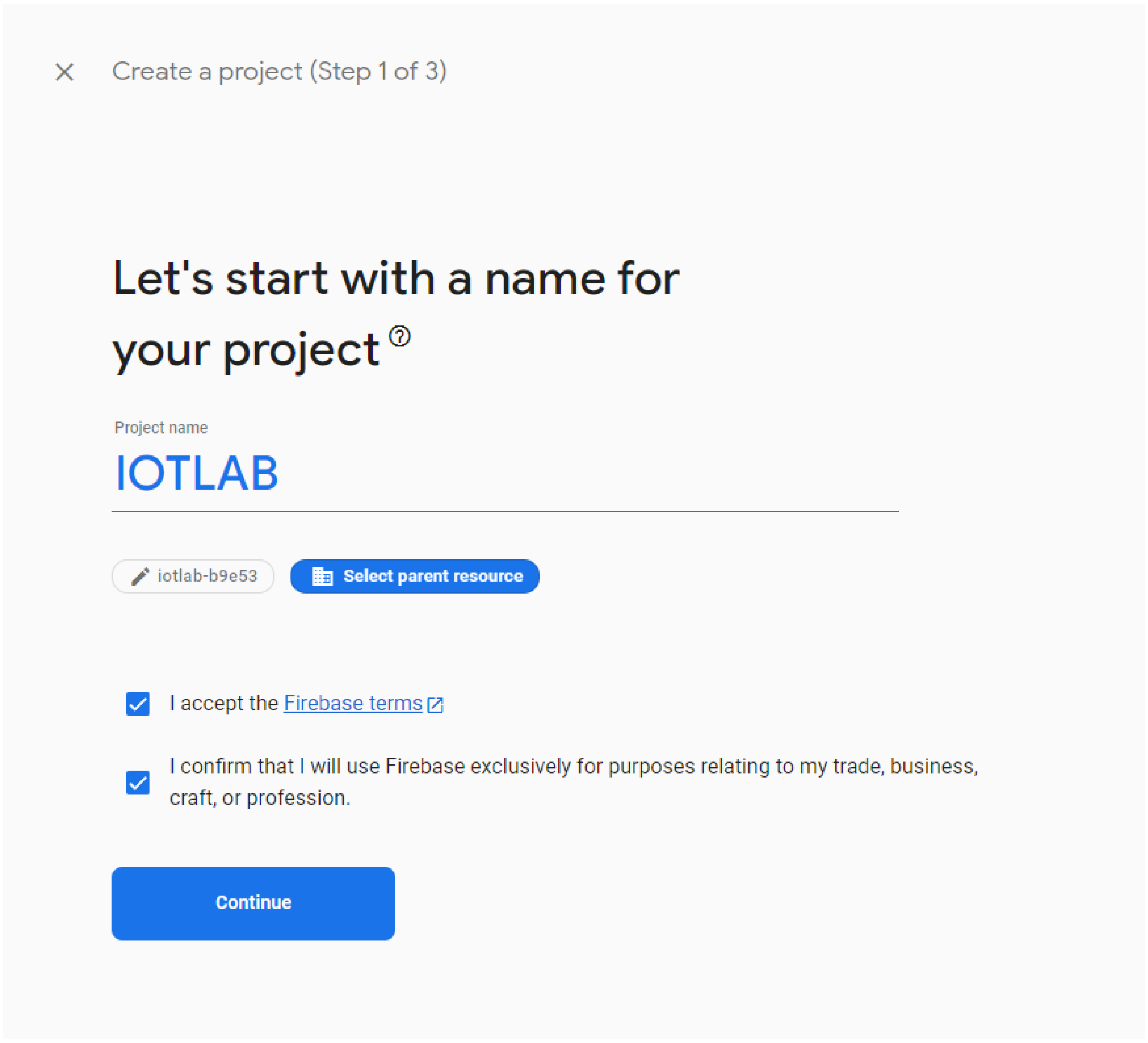
In particular, it is also a versatile service and extremely good security. Firebase supports both Android and IOS platforms. It's no wonder that many developers choose Firebase as the first platform to build apps for millions of users worldwide.

There are some steps to create a Firebase's project which will be shown below:

Step 1: Create new project: Add project



Step 2: Naming for your project



Create a project (Step 1 of 3)

Let's start with a name for your project

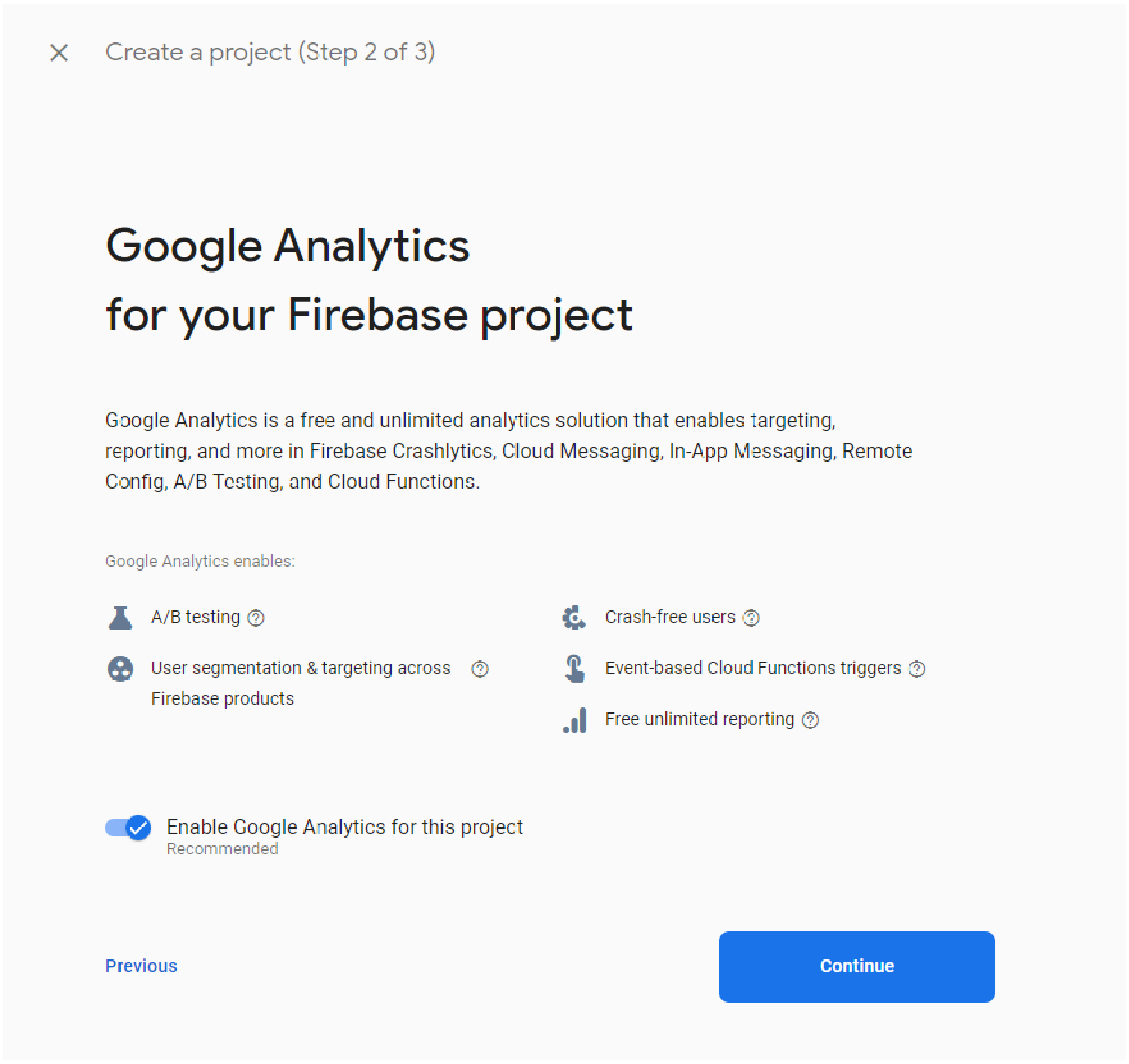
Project name

IOTLAB

✓ I accept the Firebase terms

✓ I confirm that I will use Firebase exclusively for purposes relating to my trade, business, craft, or profession.

Step 3 : Continue → Choose default account for firebase → Create project → Waitting



Create a project (Step 2 of 3)

Google Analytics
for your Firebase project

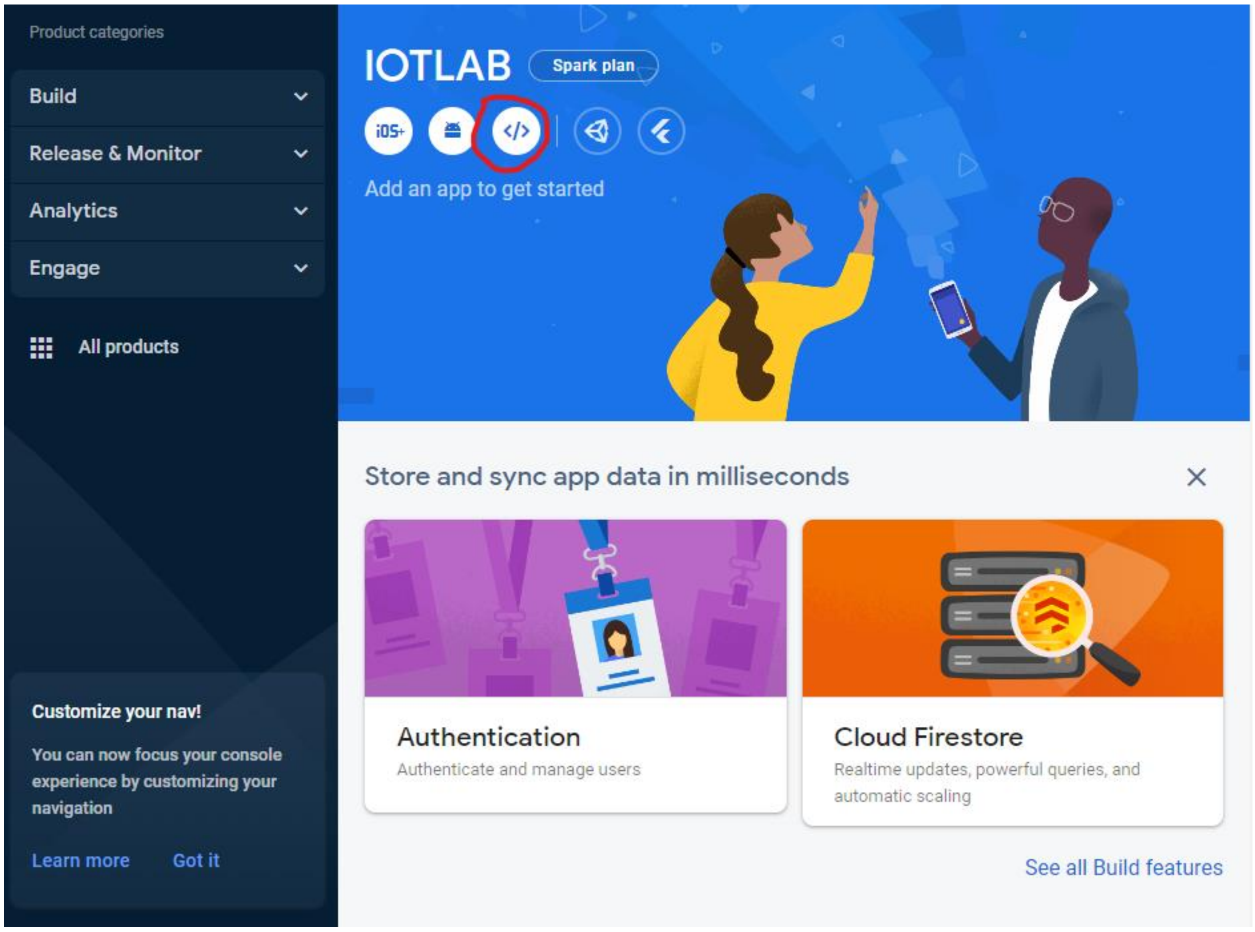
Google Analytics is a free and unlimited analytics solution that enables targeting, reporting, and more in Firebase Crashlytics, Cloud Messaging, In-App Messaging, Remote Config, A/B Testing, and Cloud Functions.

Google Analytics enables:

- A/B testing
- User segmentation & targeting across Firebase products
- Crash-free users
- Event-based Cloud Functions triggers
- Free unlimited reporting

[✓] Enable Google Analytics for this project
Recommended

Step 4: After building the project, we will get a new window, here we choose the circled icon in the picture to proceed to create the web app.



Customize your nav!

You can now focus your console experience by customizing your navigation

Store and sync app data in milliseconds

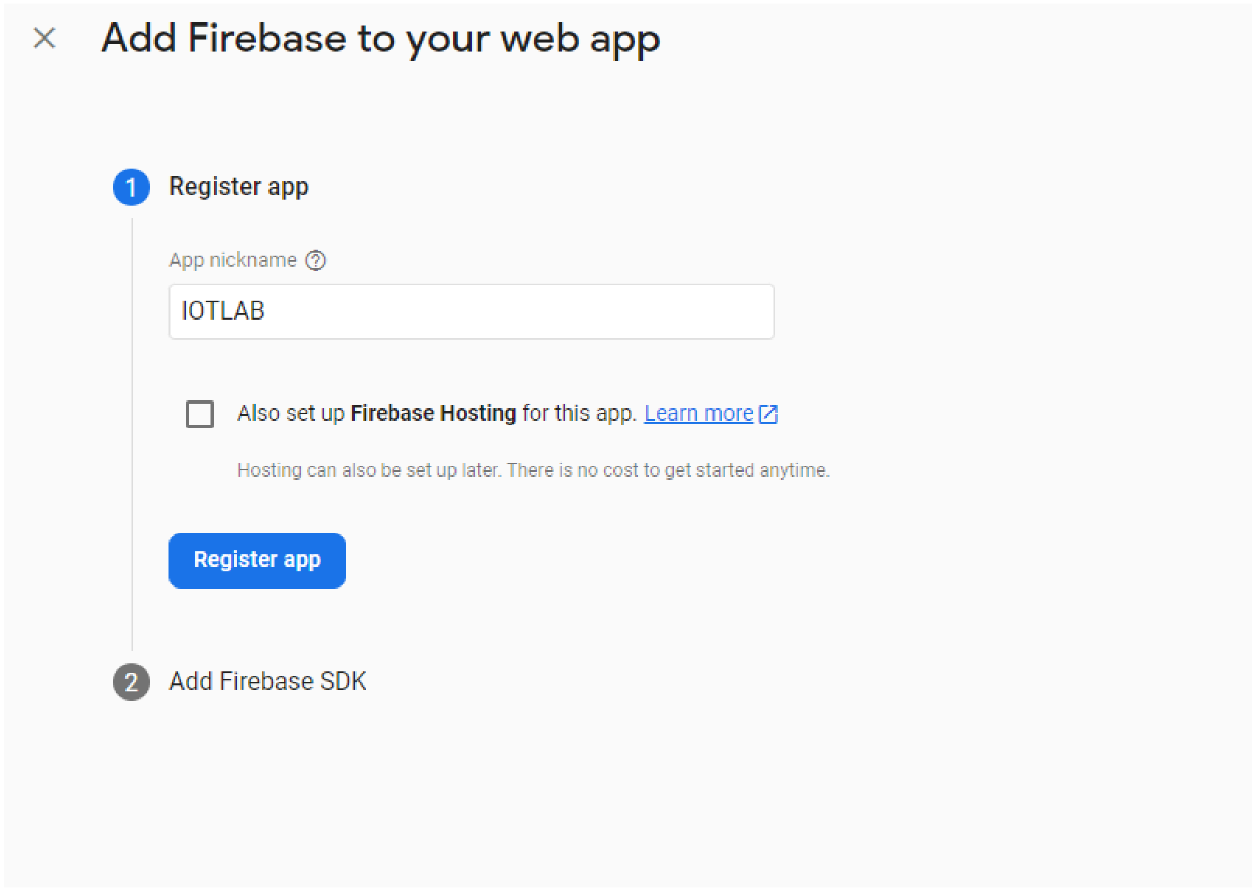
Authentication

Authenticate and manage users

Cloud Firestore

Realtime updates, powerful queries, and automatic scaling

Step 5: Naming the web app



Add Firebase to your web app

1 Register app

App nickname

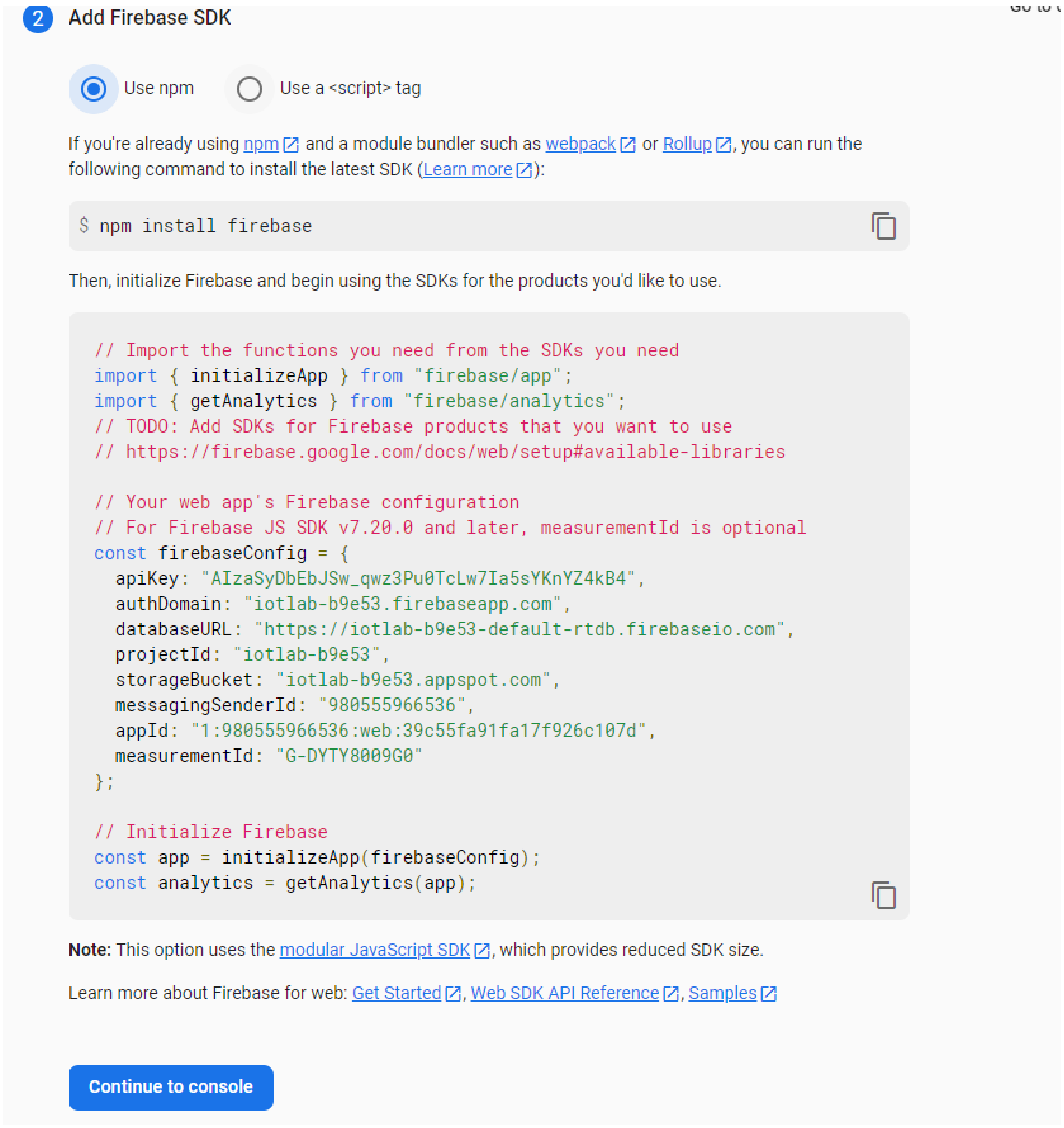
IOTLAB

☐ Also set up **Firebase Hosting** for this app. Learn more

Hosting can also be set up later. There is no cost to get started anytime.

2 Add Firebase SDK

Step 6: Select continue to console to create a web app



If you're already using npm and a module bundler such as webpack or Rollup, you can run the following command to install the latest SDK (Learn more):

```
$ npm install firebase
```

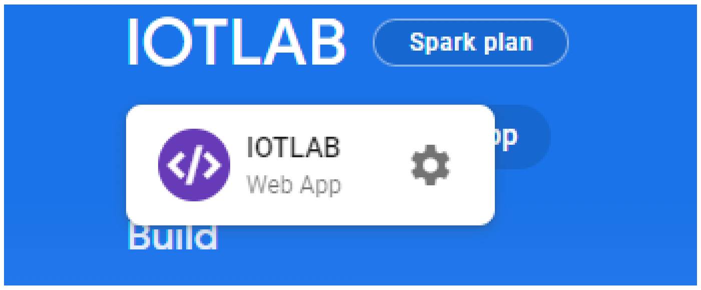
Then, initialize Firebase and begin using the SDKs for the products you'd like to use.

```
// Import the functions you need from the SDKs you need
import { initializeApp } from "firebase/app";
import { getAnalytics } from "firebase/analytics";
// TODO: Add SDKs for Firebase products that you want to use
// https://firebase.google.com/docs/web/setup#available-libraries

// Your web app's Firebase configuration
// For Firebase JS SDK v7.20.0 and later, measurementId is optional
const firebaseConfig = {
  apiKey: "AIzaSyDbEbJSw_qwz3Pu0TcLw7Ia5sYKnYZ4kB4",
  authDomain: "iotlab-b9e53.firebaseapp.com",
  databaseURL: "https://iotlab-b9e53-default-rtdb.firebaseio.com",
  projectId: "iotlab-b9e53",
  storageBucket: "iotlab-b9e53.appspot.com",
  messagingSenderId: "980555966536",
  appId: "1:980555966536:web:39c55fa91fa17f926c107d",
  measurementId: "G-DYTY8009G0"
};

// Initialize Firebase
const app = initializeApp(firebaseConfig);
const analytics = getAnalytics(app);
```

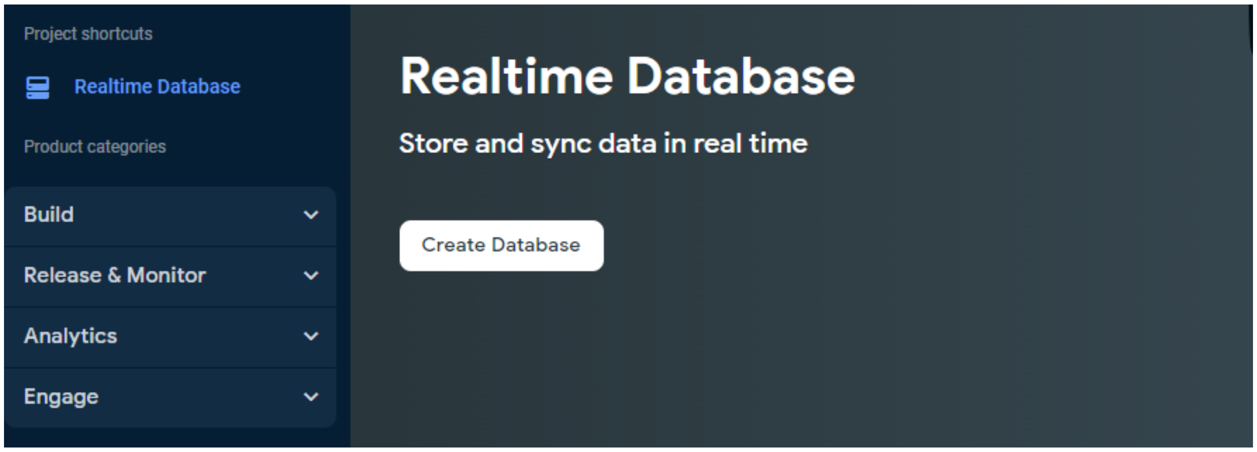
Note: This option uses the modular JavaScript SDK, which provides reduced SDK size.

Learn more about Firebase for web: Get Started, Web SDK API Reference, Samples

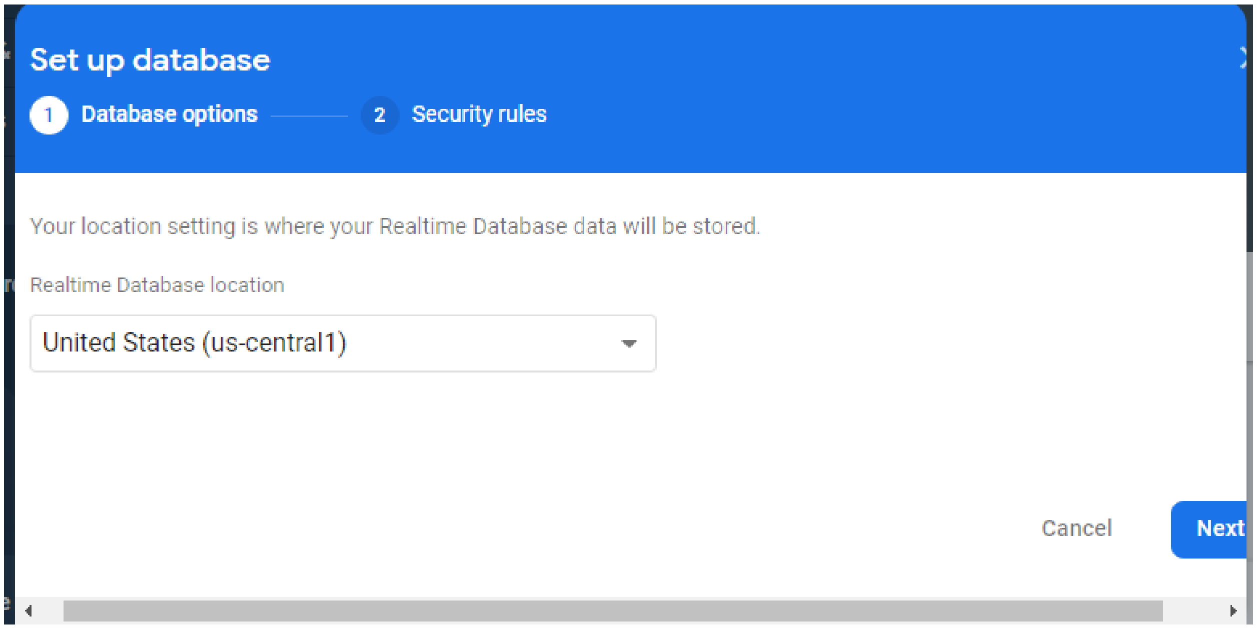
- **After successful creation, we will get as shown below:**



Step 7: Create Database



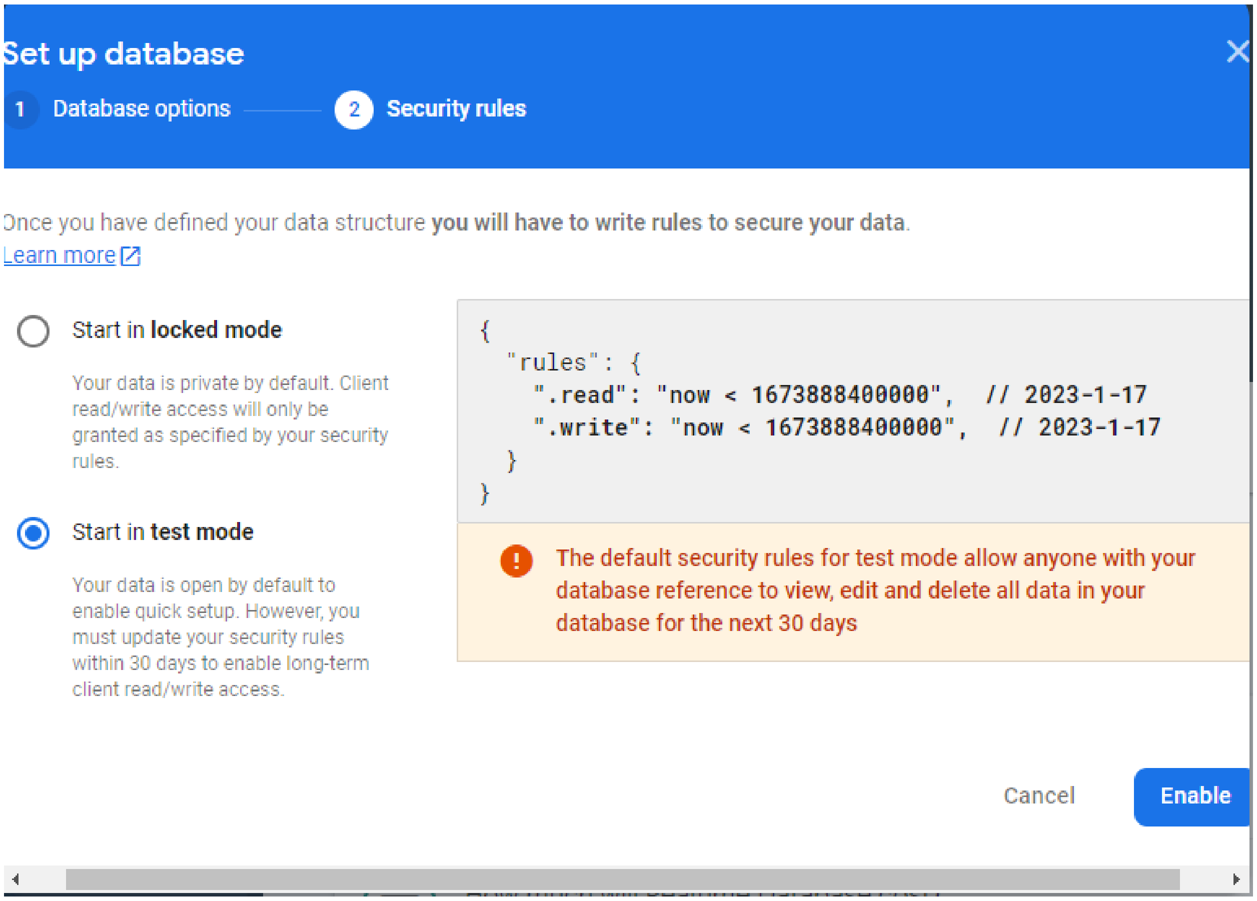
Step 8: Default realtime database is United States-->Next



Your location setting is where your Realtime Database data will be stored.

Realtime Database location: United States (us-central1)

Step 9: Start in Test mode → Enable



Set up database

Once you have defined your data structure **you will have to write rules to secure your data.**
Learn more

Start in **locked mode**

Your data is private by default. Client read/write access will only be granted as specified by your security rules.

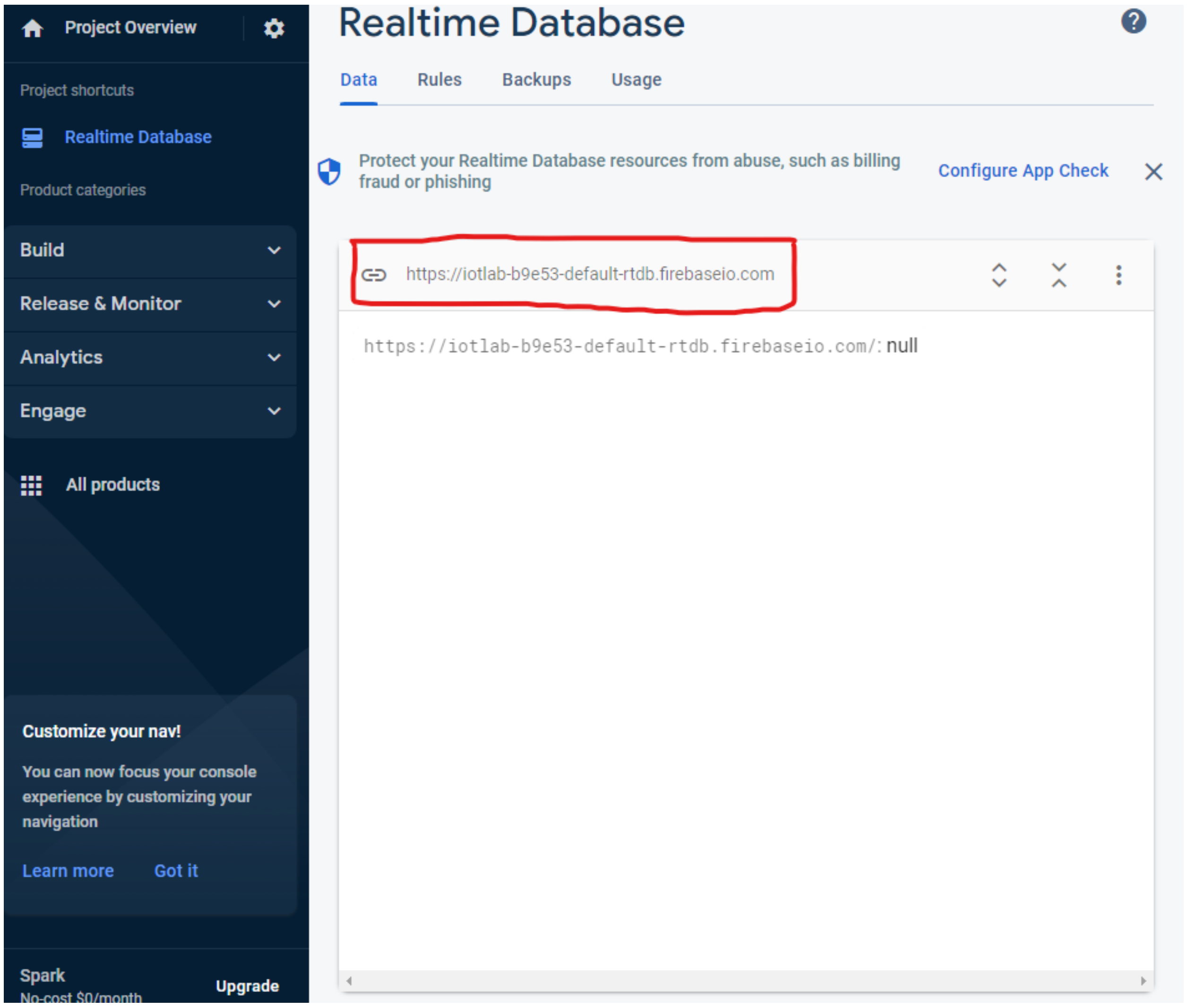
[✓ Start in **test mode**]

Your data is open by default to enable quick setup. However, you must update your security rules within 30 days to enable long-term client read/write access.

```
{
  "rules": {
    ".read": "now < 1673888400000",   // 2023-1-17
    ".write": "now < 1673888400000",   // 2023-1-17
  }
}
```

The default security rules for test mode allow anyone with your database reference to view, edit and delete all data in your database for the next 30 days

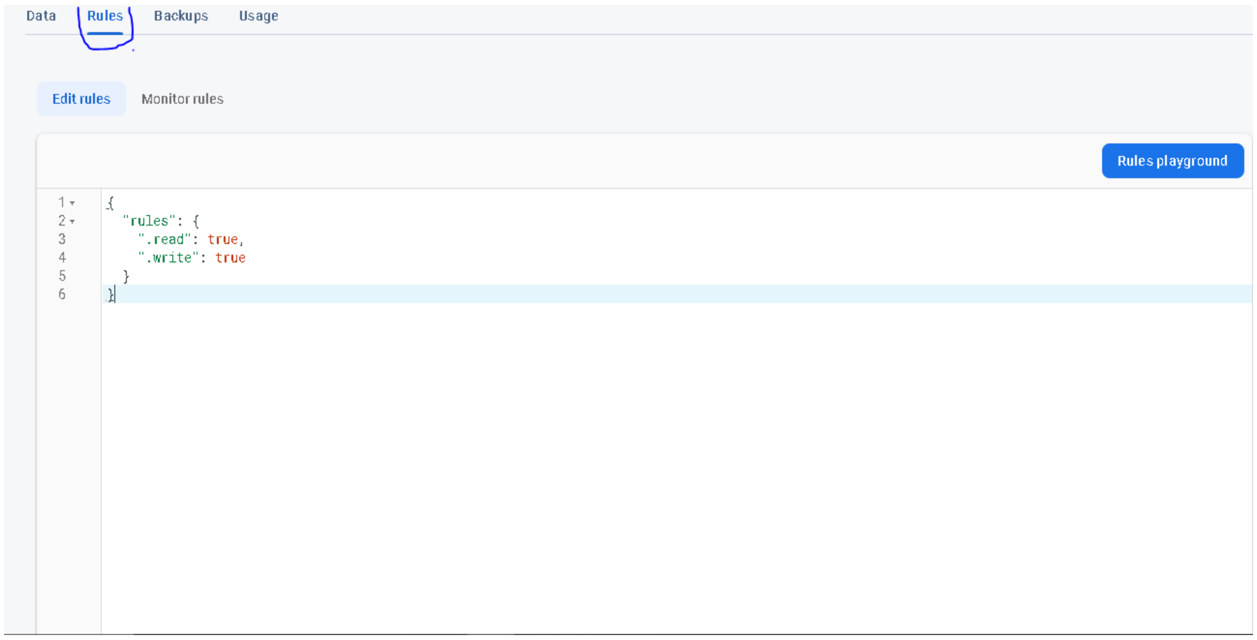
- **Once created, we will get an address as shown in the picture, this address will be the host of the firebase web app**



https://iotlab-b9e53-default-rtdb.firebaseio.com

```
https://iotlab-b9e53-default-rtdb.firebaseio.com/: null
```

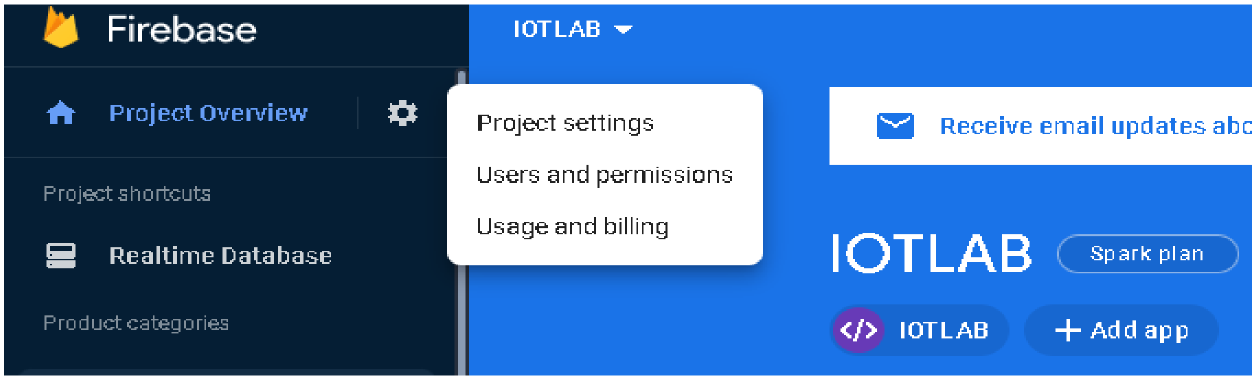
- **In addition, we need to go to the rules section next to it and change the read and write to ' true' instead of 'false' like the original. This is required if we want to link between firebase and web interface.**

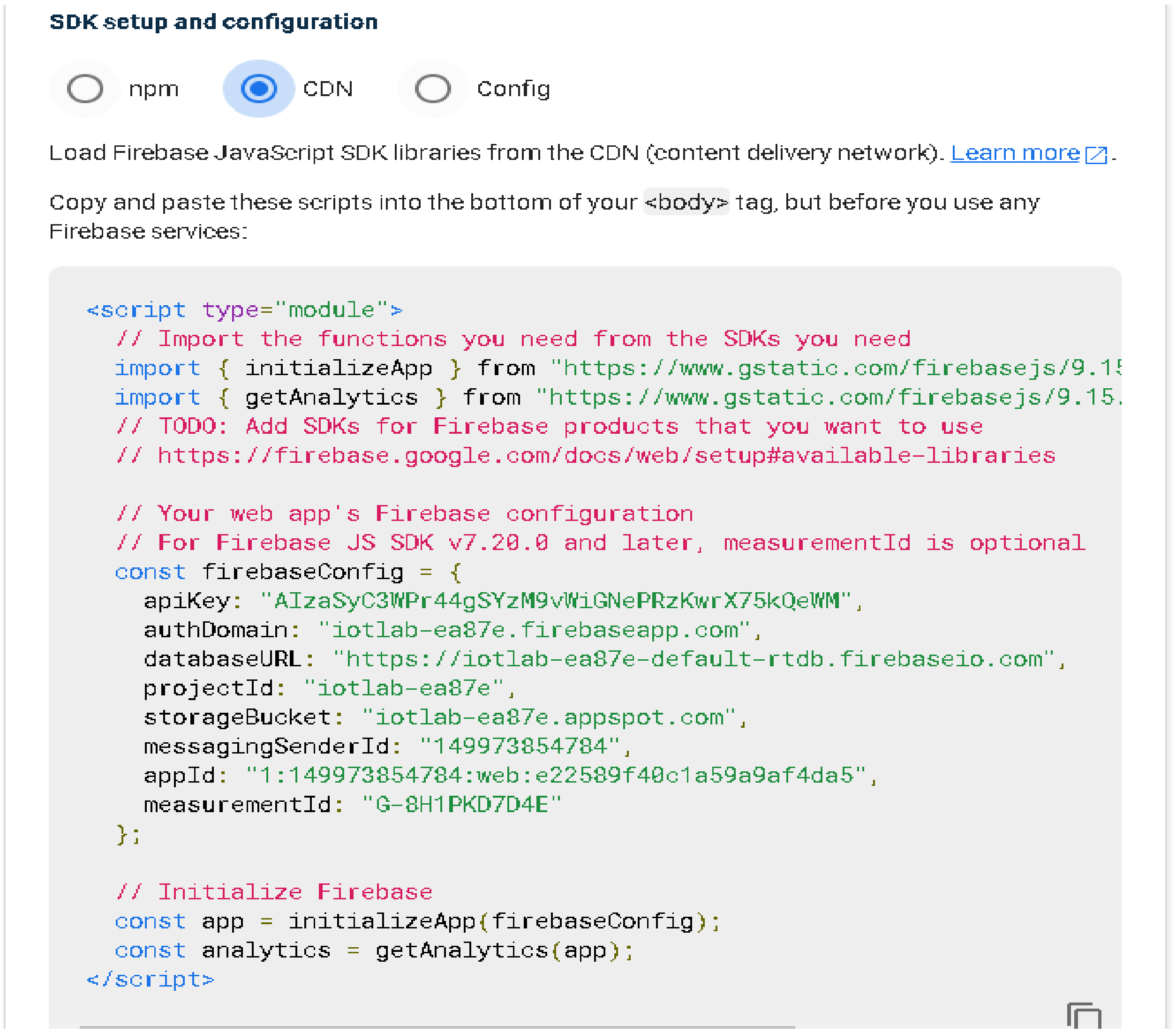


```
{
  "rules": {
    ".read": true,
    ".write": true
  }
}
```

3. Firebase link with web interface

First we choose Project settings as shown in the picture




SDK setup and configuration

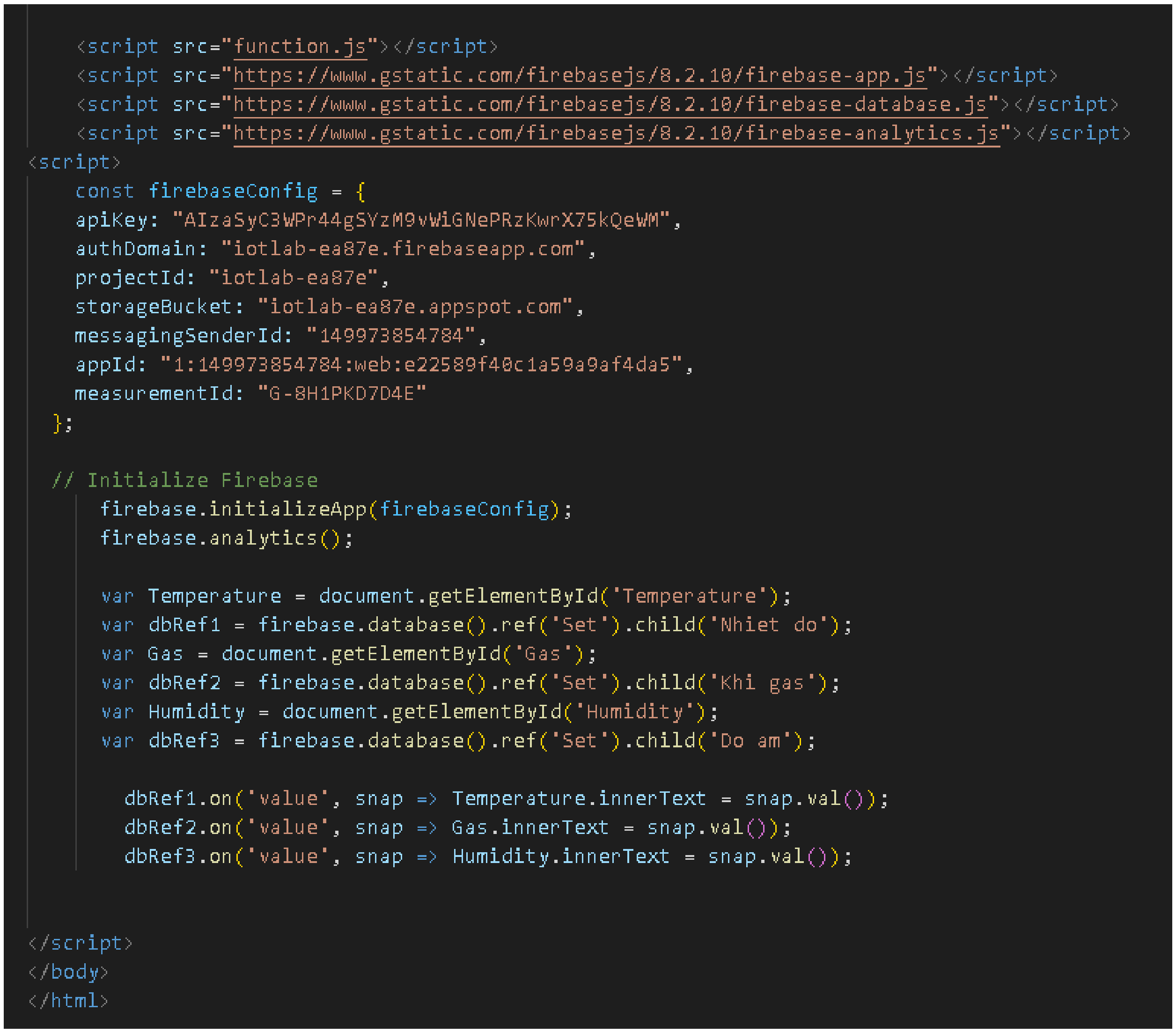
npm [✓ CDN] Config

Load Firebase JavaScript SDK libraries from the CDN (content delivery network). Learn more.

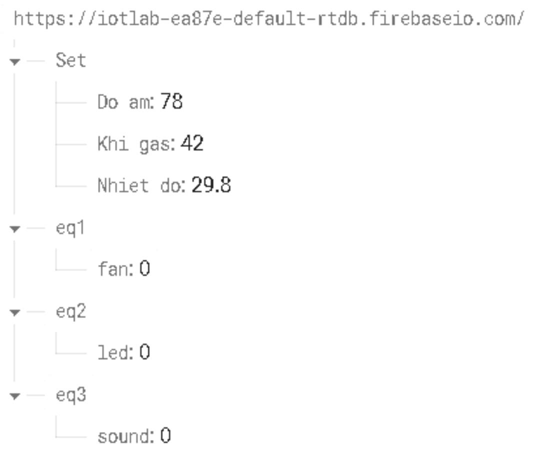
Copy and paste these scripts into the bottom of your `<body>` tag, but before you use any Firebase services:

```
<script type="module">
  // Import the functions you need from the SDKs you need
  import { initializeApp } from "https://www.gstatic.com/firebasejs/9.15
  import { getAnalytics } from "https://www.gstatic.com/firebasejs/9.15.
  // TODO: Add SDKs for Firebase products that you want to use
  // https://firebase.google.com/docs/web/setup#available-libraries

  // Your web app's Firebase configuration
  // For Firebase JS SDK v7.20.0 and later, measurementId is optional
  const firebaseConfig = {
    apiKey: "AIzaSyC3WPr44gSYzM9vWiGNePRzKwrX75kQeWM",
    authDomain: "iotlab-ea87e.firebaseapp.com",
    databaseURL: "https://iotlab-ea87e-default-rtdb.firebaseio.com",
    projectId: "iotlab-ea87e",
    storageBucket: "iotlab-ea87e.appspot.com",
    messagingSenderId: "149973854784",
    appId: "1:149973854784:web:e22589f40c1a59a9af4da5",
    measurementId: "G-8H1PKD7D4E"
  };

  // Initialize Firebase
  const app = initializeApp(firebaseConfig);
  const analytics = getAnalytics(app);
</script>
```

We will copy these scripts to put in the html file.



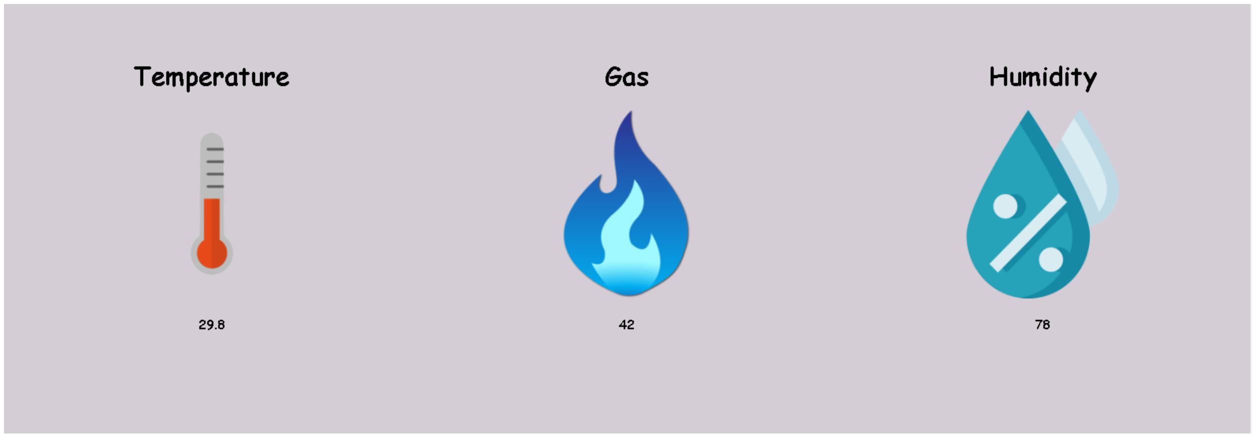
```
    <script src="function.js"></script>
    <script src="https://www.gstatic.com/firebasejs/8.2.10/firebase-app.js"></script>
    <script src="https://www.gstatic.com/firebasejs/8.2.10/firebase-database.js"></script>
    <script src="https://www.gstatic.com/firebasejs/8.2.10/firebase-analytics.js"></script>
<script>
    const firebaseConfig = {
    apiKey: "AIzaSyC3WPr44gSYzM9vWiGNePRzKwrX75kQeWM",
    authDomain: "iotlab-ea87e.firebaseapp.com",
    projectId: "iotlab-ea87e",
    storageBucket: "iotlab-ea87e.appspot.com",
    messagingSenderId: "149973854784",
    appId: "1:149973854784:web:e22589f40c1a59a9af4da5",
    measurementId: "G-8H1PKD7D4E"
  };

  // Initialize Firebase
      firebase.initializeApp(firebaseConfig);
      firebase.analytics();

      var Temperature = document.getElementById('Temperature');
      var dbRef1 = firebase.database().ref('Set').child('Nhiet do');
      var Gas = document.getElementById('Gas');
      var dbRef2 = firebase.database().ref('Set').child('Khi gas');
      var Humidity = document.getElementById('Humidity');
      var dbRef3 = firebase.database().ref('Set').child('Do am');

        dbRef1.on('value', snap => Temperature.innerText = snap.val());
        dbRef2.on('value', snap => Gas.innerText = snap.val());
        dbRef3.on('value', snap => Humidity.innerText = snap.val());


</script>
</body>
</html>
```



```
https://iotlab-ea87e-default-rtdb.firebaseio.com/
▾ — Set
      — Do am: 78
      — Khi gas: 42
      — Nhiet do: 29.8
▾ — eq1
      — fan: 0
▾ — eq2
      — led: 0
▾ — eq3
      — sound: 0
```

Next we will go to firebase, realtime database to set the name and sensor parameters that we want.

If the link is successful, we will get the same parameters on the web interface as the parameters we just set in the firebase.



The parameters below the image are exactly the same as the values we just set on firebase.

3. Hardware preparation

3.1 ESP8266

Figure 3.1 ESP8266

Esp8266 is a integrated circuit Wifi connection, this module is manufactured with 17 GPIO pins (General-Purpose Input/Output) that can be assigned to various functions by programming the appropriate registers for each D pin (Digital pin) or A pin (Analog pin) with Arduino programming software, typically the Arduino IDE.

In the case of this project will use:

+ Pin A0 connects to Module MQ3.

+ Pin D0 is used to turn on / off the LED.

+ Pin D1 is used to turn on / off the Buzzer.

+ Pin D5 connected to Module DHT11.

+ Pin D7 is used to turn on / off the Fan.

3.2 DHT11 – Tempertature Humidity Sensor

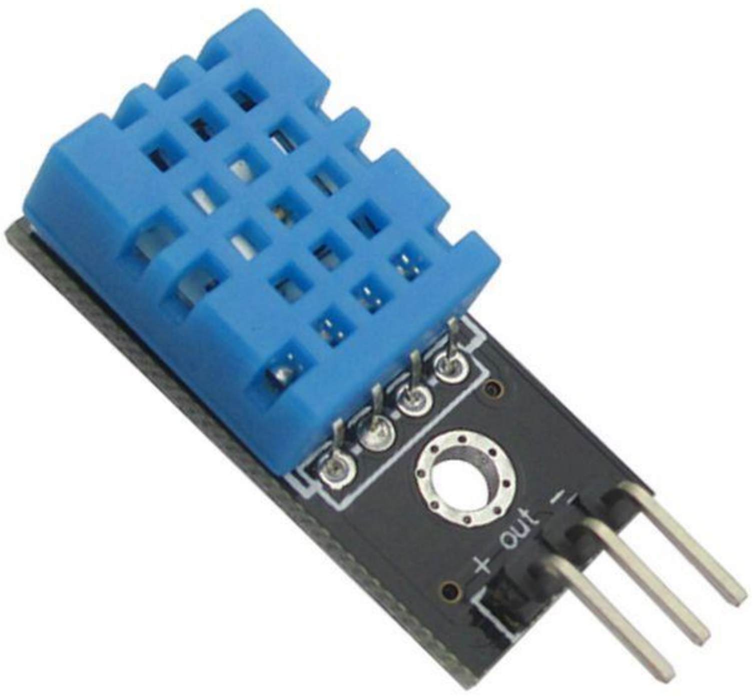


Figure 3.2 DHT11

In order to monitor humidity and temperature degree instantly, the DHT11 sensor employs a thermistor and a capacitive humidity sensor. It can be readily interfaced with any microcontrollers like Arduino, Raspberry Pi, etc. The dielectric between the two electrodes of the humidity detecting capacitor, which consists of two electrodes, is a moisture-retaining substrate. With a change in humidity level, the capacitance value changes. The measuring IC transforms these altered resistance values into digital form and conveniently obtains the data via 1-wire connection (1-wire digital communication only data transmission). The sensor's integrated signal preprocessing enables the user to obtain precise data without performing any calculations.

3.3 Gas Sensor

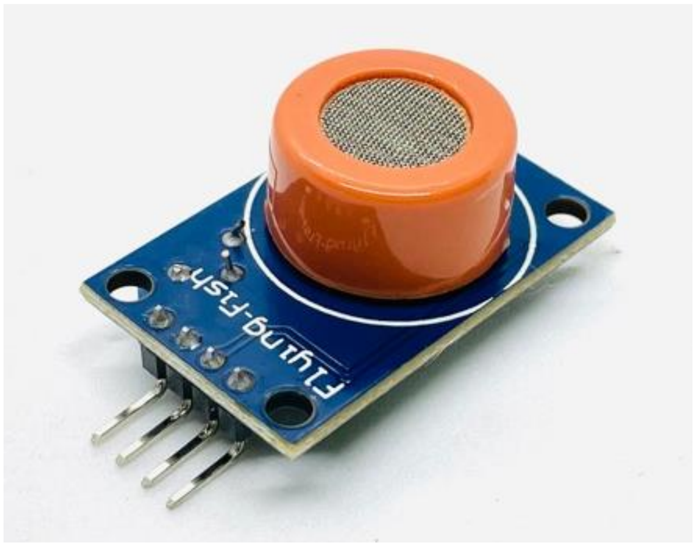


Figure 3.3 Gas sensor

For detecting gas leaks, the Grove - Gas Sensor(MQ3) module is helpful (in home and industry). It is capable of detecting CO, CH4, Benzine, Hexane, LPG, and Alcohol. Measurements may be made as soon as feasible thanks to its high sensitivity and quick reaction time. The potentiometer can be used to modify the sensor's sensitivity.

This sensor has an Analog output. This must be linked to one of the microcontrollers' Analog sockets. The A0 analog pin is used in the examples in this report, which attach this module to the ESP8266's A0 port.

3.4 5V Relay Module

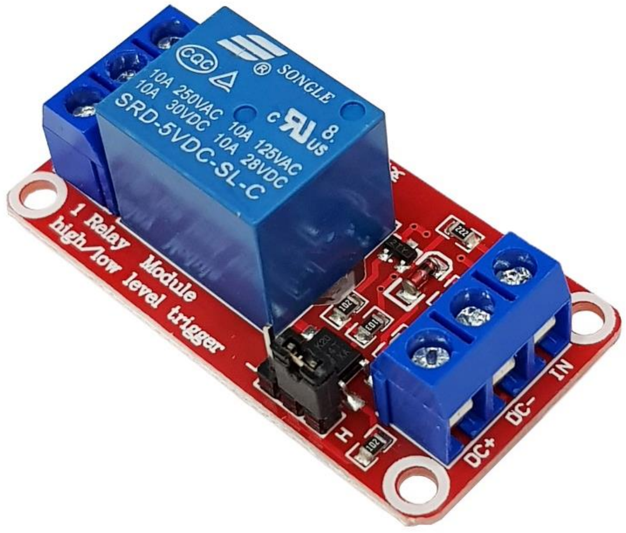


Figure 3.4 5V Relay Module

A relay is an electromagnetic switch that can turn on or off a significantly bigger current using a relatively little amount of current. There is an electromagnet inside the relay (a coil that becomes a temporary magnet when current flows through it).

means that when a device is turned on using a tiny current and it activates ("pulls"), a considerably bigger current is used by the other device.

3.5 Mini cooling fan DC 5V

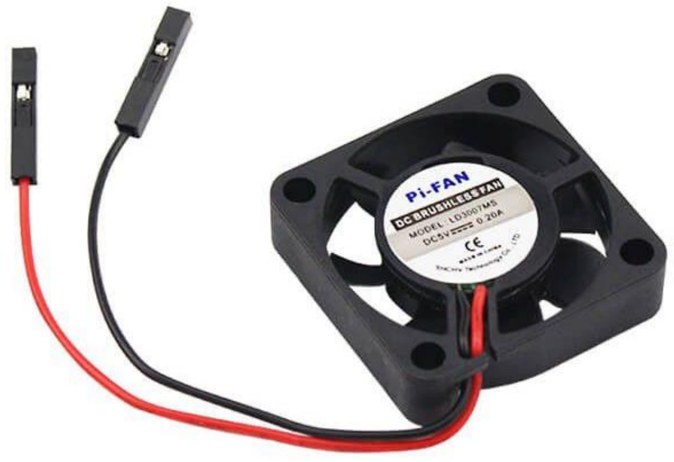


Figure 3.5 Mini cooling fan DC 5V

This DC 5V mini fan is a mini fan for ventilation. In this report is used to trigger high or low level by button designed on Web.

3.6 Buzzer 5V

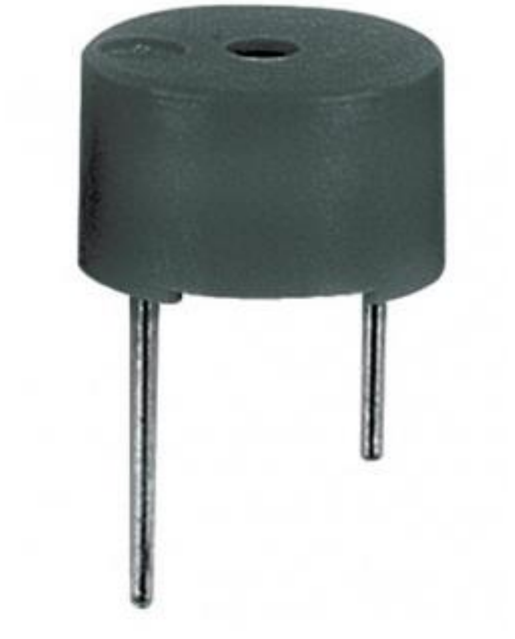


Figure 3.6 Buzzer 5V

This buzzer is an active buzzer, which simply means that even when you just offer steady DC power, it will buzz at a predetermined frequency (2300±300 Hz) on its own. One benefit of an active buzzer is that it may still make noise when linked to a microcontroller, such as an Arduino, by just driving a normal high output on the pin. The advantages of this include that sound can be produced without the usage of processor power, hardware timers, or additional programming.

3.7 LED

Figure 3.7 LED

When current passes through a light-emitting diode (LED), a semiconductor device, light is released. Recombining electrons and electron holes in the semiconductor results in the release of energy in the form of photons. The energy needed for electrons to pass through the semiconductor's band gap determines the hue of the light, which corresponds to the energy of the photons. A layer of light-emitting phosphor or several semiconductors can be used to create white light on a semiconductor device.

3.8 Wiring Diagram

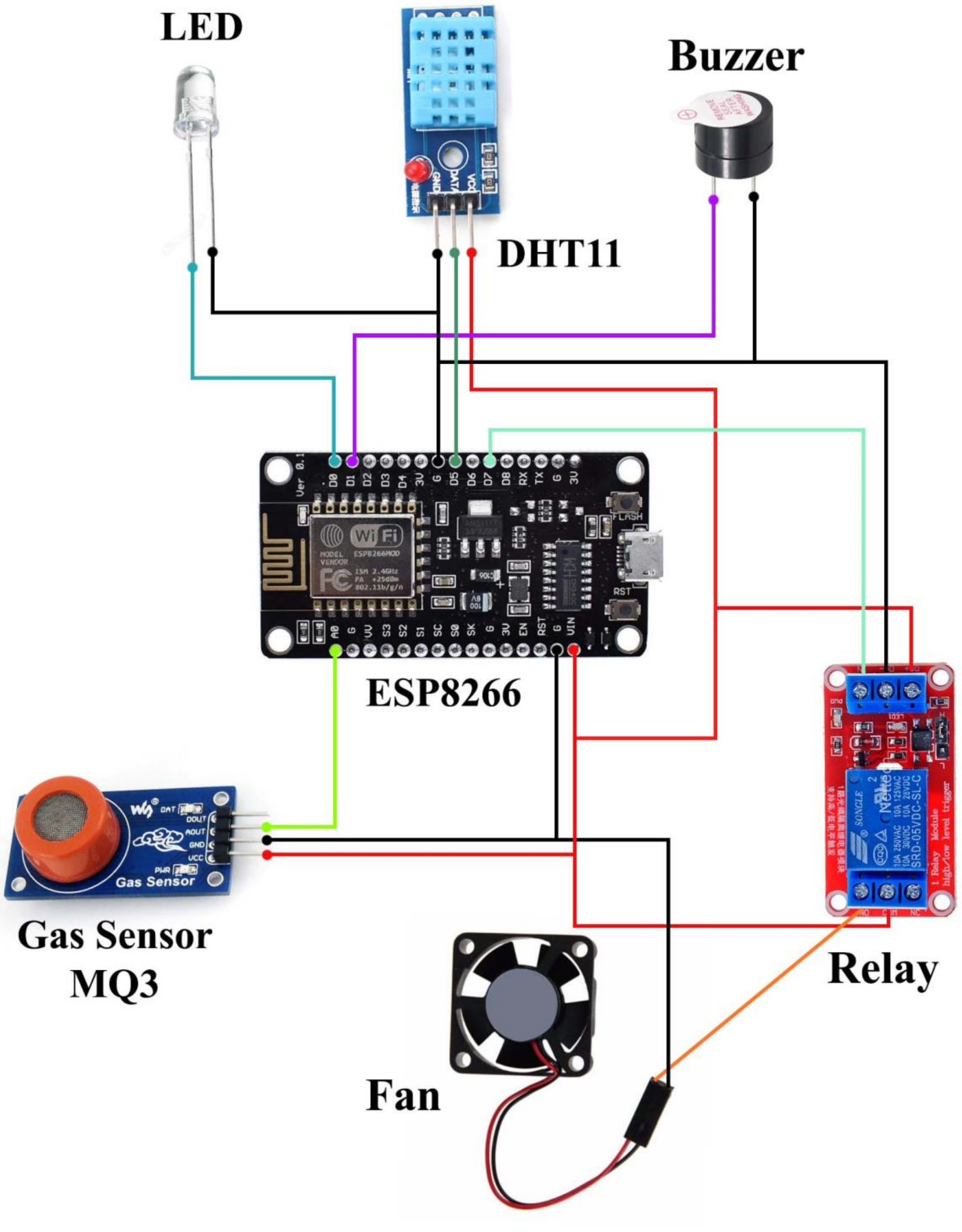
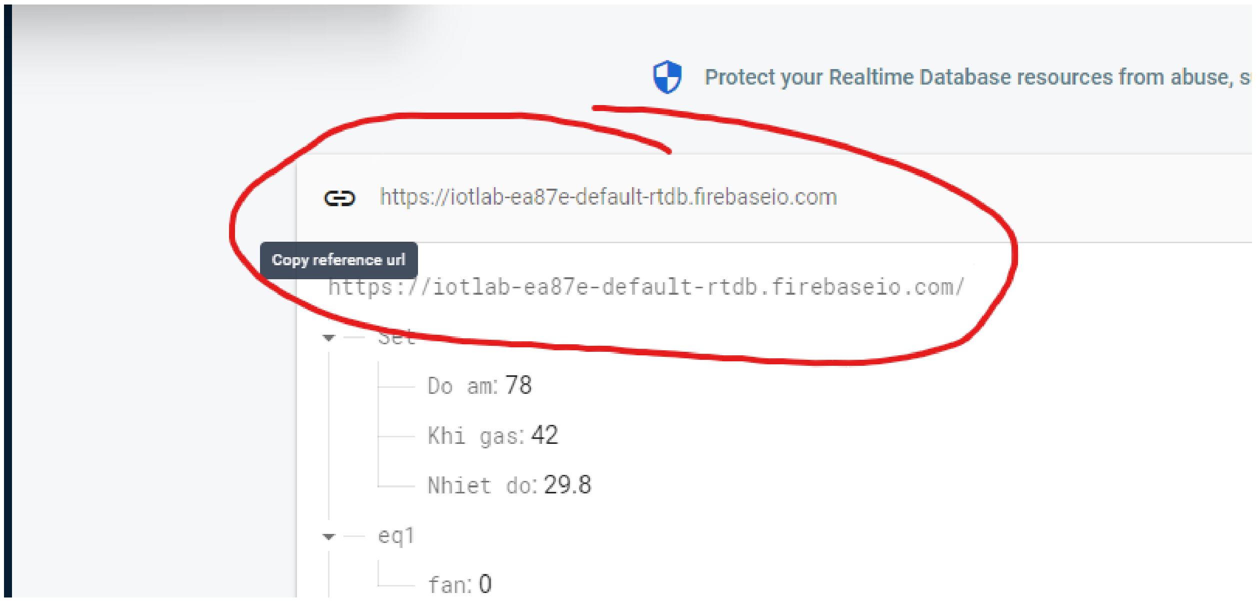


Figure 3.8 Diagram

4. Project Implementation

4.1 Link Firebase with Esp8266

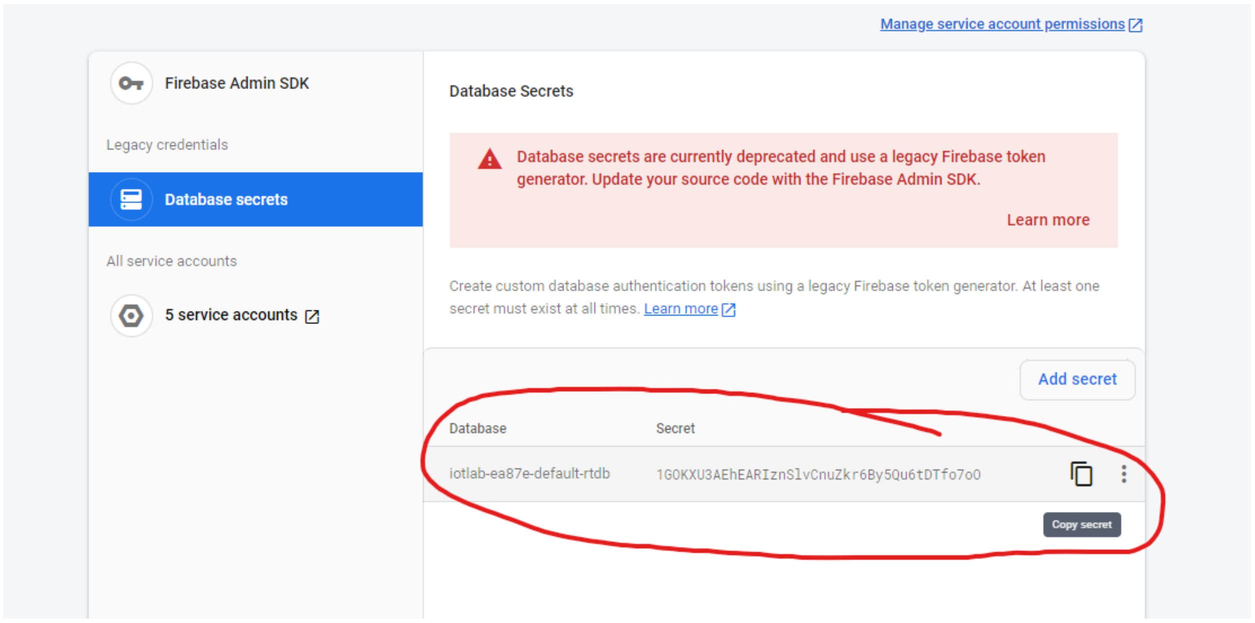
Take the URL and Auth Token to verify the link between Esp8266 and Firebase.



https://iotlab-ea87e-default-rtdb.firebaseio.com

```
https://iotlab-ea87e-default-rtdb.firebaseio.com/
  Set
    Do am: 78
    Khi gas: 42
    Nhiet do: 29.8
  eq1
    fan: 0
```

Figure 4.1.1 URL on Firebase Website



Database Secrets

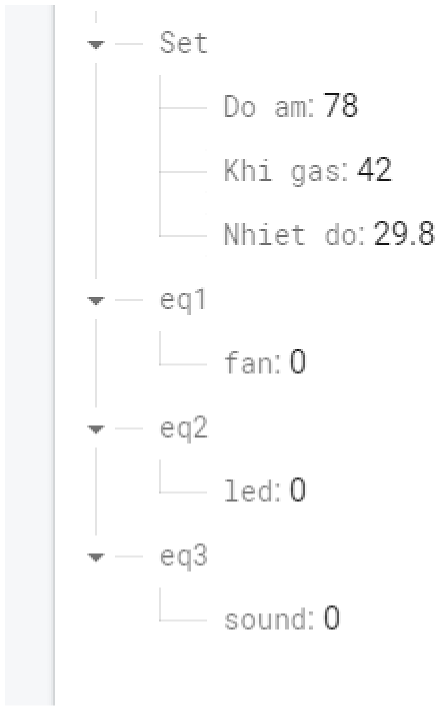
Database secrets are currently deprecated and use a legacy Firebase token generator. Update your source code with the Firebase Admin SDK.

Create custom database authentication tokens using a legacy Firebase token generator. At least one secret must exist at all times. Learn more

Database	Secret
iotlab-ea87e-default-rtdb	1GOKXU3AEhEARIznSlvCnuZkr6By5Qu6tDTfo7oO

Figure 4.1.2 Take the Auth Token

Initialize values on Firebase linked to the Web to code into the Arduino IDE.

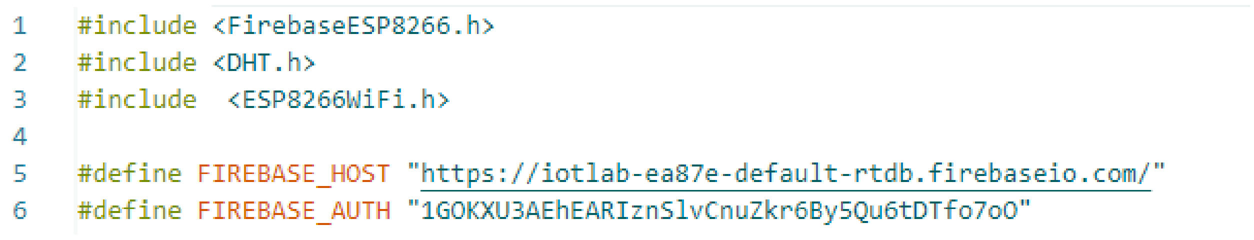


```
Set
    Do am: 78
    Khi gas: 42
    Nhiet do: 29.8
eq1
    fan: 0
eq2
    led: 0
eq3
    sound: 0
```

Figure 4.1.3 Values already initialized

4.2 Loading code into the Esp8266

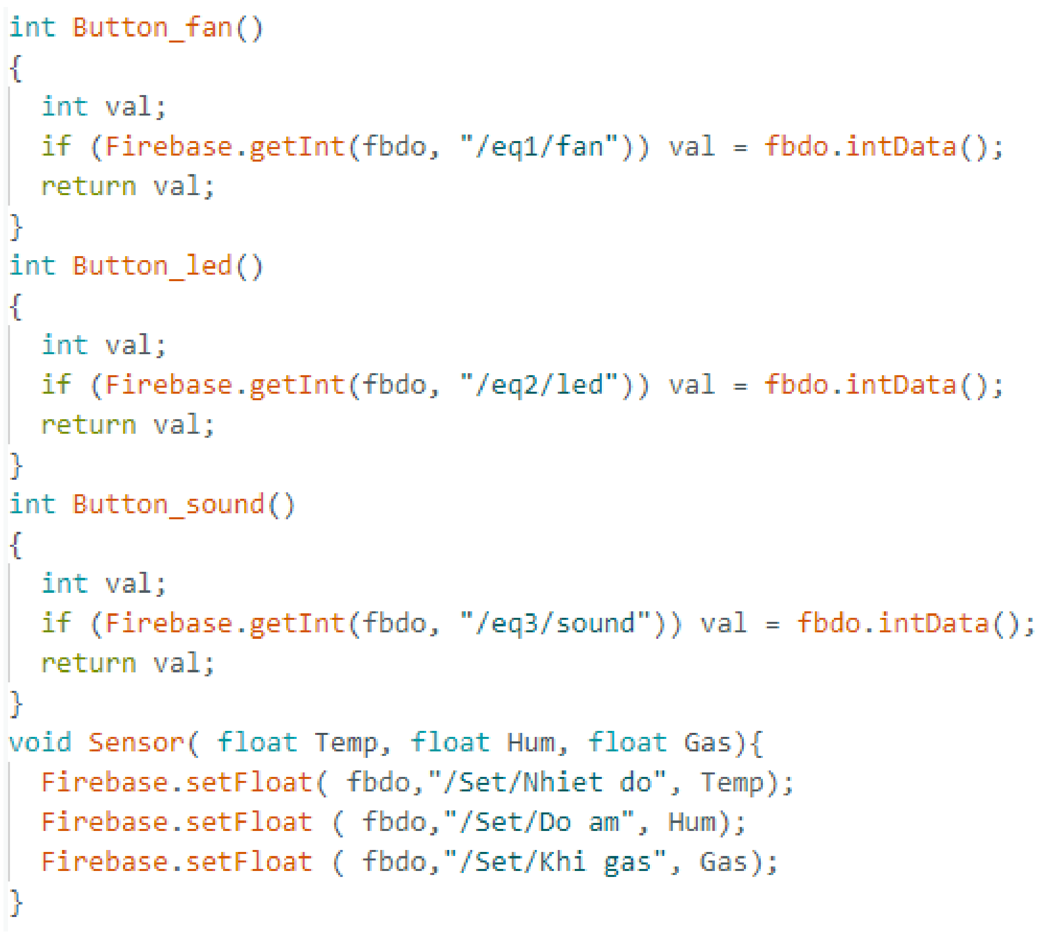
Include firebaseEsp8266 library and define URL and Auth Token.



```
#include <FirebaseESP8266.h>
#include <DHT.h>
#include  <ESP8266WiFi.h>

#define FIREBASE_HOST "https://iotlab-ea87e-default-rtdb.firebaseio.com/"
#define FIREBASE_AUTH "1GOKXU3AEhEARIznSlvCnuZkr6By5Qu6tDTfo7oO"
```

Figure 4.2.1 Values already initialized

Code the commands associated with Firebase about the parameters of sensors and switches of electrical devices.

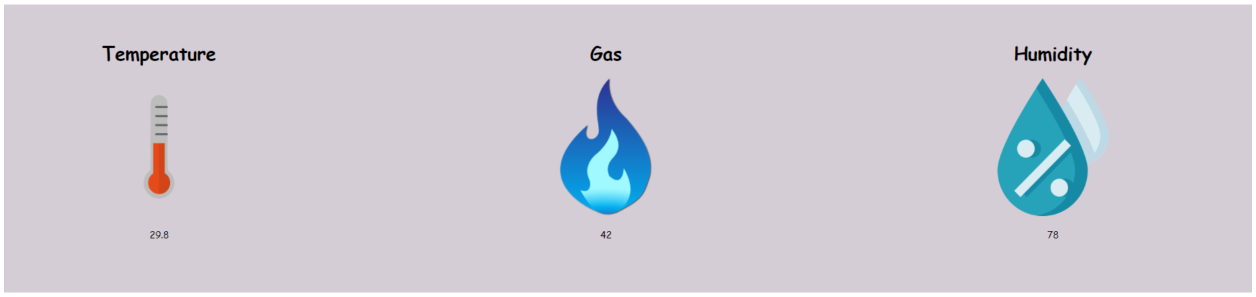


```
int Button_fan()
{
  int val;
  if (Firebase.getInt(fbdo, "/eq1/fan")) val = fbdo.intData();
  return val;
}
int Button_led()
{
  int val;
  if (Firebase.getInt(fbdo, "/eq2/led")) val = fbdo.intData();
  return val;
}
int Button_sound()
{
  int val;
  if (Firebase.getInt(fbdo, "/eq3/sound")) val = fbdo.intData();
  return val;
}
void Sensor( float Temp, float Hum, float Gas){
  Firebase.setFloat( fbdo,"/Set/Nhiet do", Temp);
  Firebase.setFloat ( fbdo,"/Set/Do am", Hum);
  Firebase.setFloat ( fbdo,"/Set/Khi gas", Gas);
}
```

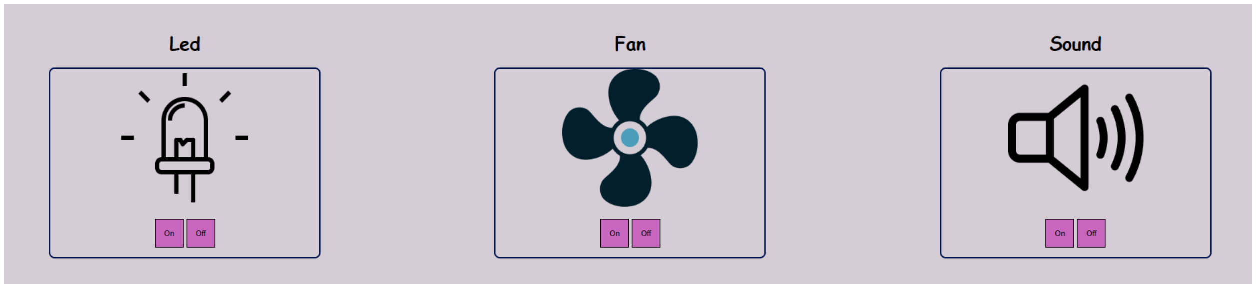
Figure 4.2.2 Initialize the values of sensors and buttons on the Arduino IDE

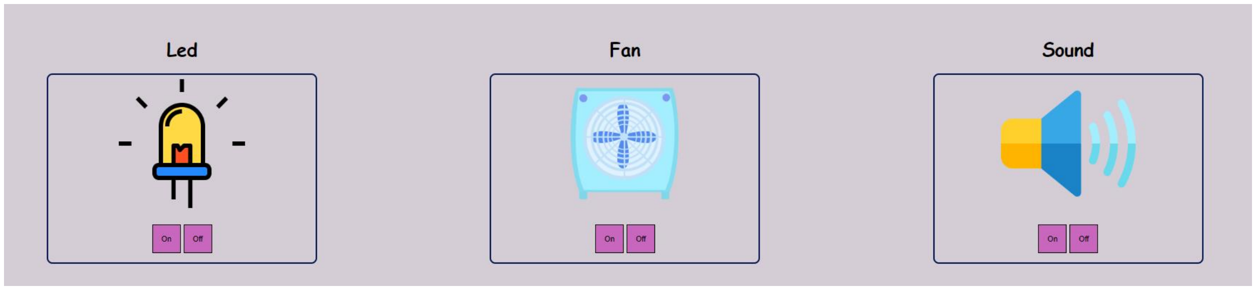
4.3 Simulate changes from sensors and turn devices on and off from the Website

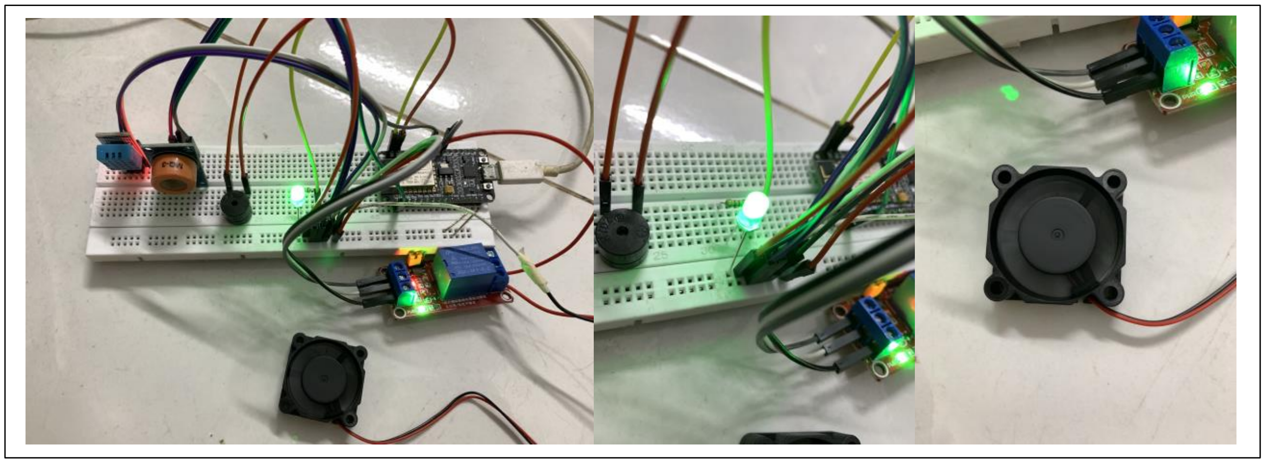
Variable Temperature, Humidity and Gas values are synchronized on Web and Firebase.



Turning on and off electrical appliances is also set up on the Web.







References

[1] Esp8266: https://en.wikipedia.org/wiki/ESP8266

[2] DHT11 – Temperature & Humidity Sensor: https://pdf1.alldatasheet.com/datasheet-pdf/view/1440068/ETC/DHT11.html

[3] MQ3 – Gas sensor: https://www.sparkfun.com/datasheets/Sensors/MQ-3.pdf

[4] Fan 5V: https://www.tme.eu/Document/c0757d73f5c4db1ceebc4553060fdf04/gm0503pfv2-8gn(d03012030g-00).pdf

[5] Buzzer: https://www.addicore.com/Active-Buzzer-5V-p/ad146.htm

[6] LED: https://www.farnell.com/datasheets/1498852.pdf

.

Link video: https://youtu.be/B9XNjSRFXQY