LAPORAN PRAKTIKUM TEKNIK KENDALI DAN MESIN LISTRIK

“ Internet of Things ”

DISUSUN OLEH:

Kelompok 2

ARM 2

Ade Surya Pramudya	(19/441188/SV/16540)
Mahesa Audriansyah Agatha	(19/441203/SV/16555)

TEKNOLOGI REKAYASA MESIN

DEPARTEMEN TEKNIK MESIN SEKOLAH VOKASI

UNIVERSITAS GADJAH MADA

YOGYAKARTA

2022

I. Deskripsi Kasus

Dalam proyek ini kami memiliki 3 kasus yang bersangkutan dengan *Internet of Things* (IOT) dimana setiap kasus memiliki beberapa cara untuk mengatasi dan menyelesaikan tugasnya, kasus tersebut diantaranya :

1. Protokol IOT berbasis HTTP dan MQTT
2. IOT Cloud provider/service
3. Implementasi Simple Home Automation berbasis Device yang ada di pasaran

II. Analisa Persiapan dan Tahap Pengerjaan

1. Protokol IOT berbasis HTTP dan MQTT

A. Protokol IOT berbasis MQTT

✓ **Komponen dan Ekstensi aplikasi yang digunakan**

1. Komponen yang digunakan
   - ESP32 board dengan chip ESP-WROOM-32
   - Resistor
   - Kabel jumper
   - LED 2 buah
2. Persiapan Ekstensi aplikasi yang digunakan

   A. Install dan jalankan Mosquitto Broker

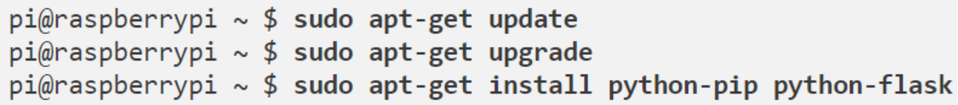
   MQTT adalah singkatan dari Message Queuing Telemetry Transport. Mosquitto MQTT adalah protokol pesan sederhana, yang dirancang untuk perangkat terbatas dengan bandwidth rendah. Jadi, ini adalah solusi sempurna untuk bertukar data antara beberapa perangkat IoT.

   Komunikasi MQTT berfungsi sebagai sistem publish dan subscribe. Perangkat memublikasikan pesan tentang topik tertentu. Semua perangkat yang berlangganan topik tersebut menerima pesan tersebut.

   Broker MQTT bertanggung jawab untuk menerima semua pesan, memfilter pesan, memutuskan siapa yang tertarik padanya, dan kemudian menerbitkan pesan ke semua klien yang berlangganan.

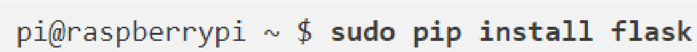
   B. Python Web Server with Flask

   Untuk menginstal Flask, kami harus menginstal pip.



```
pi@raspberrypi ~ $ sudo apt-get update
pi@raspberrypi ~ $ sudo apt-get upgrade
pi@raspberrypi ~ $ sudo apt-get install python-pip python-flask
```

   Kemudian, kami menggunakan pip untuk menginstal Flask dan dependensinya



```
pi@raspberrypi ~ $ sudo pip install flask
```

   C. Install Python Paho-MQTT

Paket Paho-MQTT menyediakan kelas klien yang memungkinkan aplikasi untuk terhubung ke broker MQTT untuk mempublikasikan pesan, dan untuk berlangganan topik dan menerima pesan yang dipublikasikan. Dalam contoh ini, server web Python akan mempublikasikan pesan ke ESP32 untuk mengaktifkan dan menonaktifkan GPIO.

Untuk menginstal paho-mqtt jalankan perintah berikut

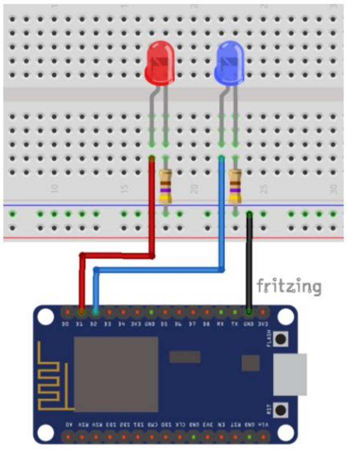
```
pip install paho-mqtt
```

D. Instal Aplikasi Penunjang

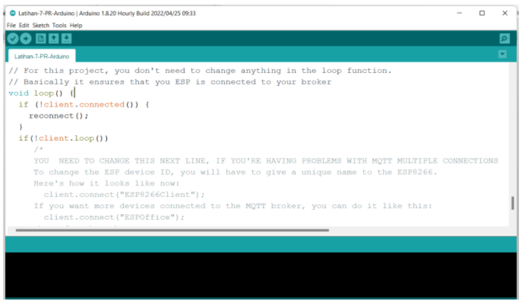
Kami menggunakan Visual Studio Code untuk menunjang pembuatan coding python dan html yang akan dibuat.

- ✓ **Tahap Pengerjaan**

1. Pembuatan Skema Rangkaian



2. Pembuatan ESP32 code di Arduino IDE


Latihan-7-PR-Arduino | Arduino 1.8.20 Hourly Build 2022/04/25 09:33
File Edit Sketch Tools Help
Latihan-7-PR-Arduino

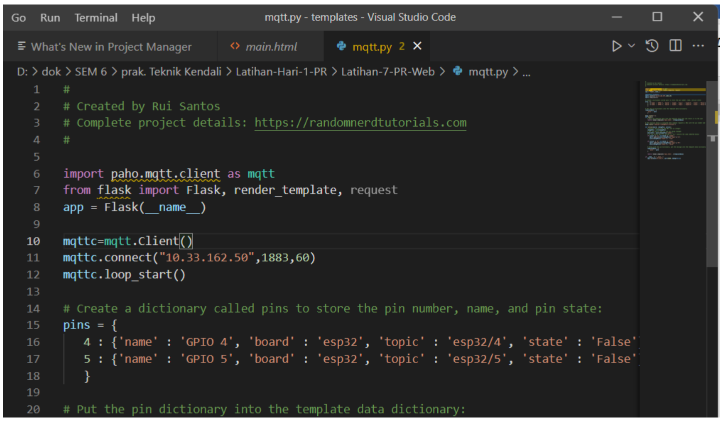
```
// For this project, you don't need to change anything in the loop function.
// Basically it ensures that you ESP is connected to your broker
void loop() {
  if (!client.connected()) {
    reconnect();
  }
  if(!client.loop())
     /*
     YOU  NEED TO CHANGE THIS NEXT LINE, IF YOU'RE HAVING PROBLEMS WITH MQTT MULTIPLE CONNECTIONS
     To change the ESP device ID, you will have to give a unique name to the ESP8266.
     Here's how it looks like now:
       client.connect("ESP8266Client");
     If you want more devices connected to the MQTT broker, you can do it like this:
       client.connect("ESPOffice");
```

```
// Loading the ESP32 WiFi library and the PubSubClient library
#include <WiFi.h>
#include <PubSubClient.h>
#define WIFI_TIMEOUT_MS 20000

// Change the credentials below, so your ESP8266 connects to your router
const char* ssid = "UGM-Hotspot";
const char* password = "";
```

```
// Change the variable to your Raspberry Pi IP address, so it connects to
your MQTT broker
const char* mqtt_server = "10.33.162.50";

// Initializes the espClient
WiFiClient espClient;
PubSubClient client(espClient);

// Connect an LED to each GPIO of your ESP8266
const int ledGPIO5 = 27;
const int ledGPIO4 = 26;

// Don't change the function below. This functions connects your ESP8266
to your router
void connectToWiFi(){
  Serial.print("");
  Serial.println("Connecting to WiFi");
  WiFi.mode(WIFI_STA);
  WiFi.begin(ssid, password);

  unsigned long startAttemptTime = millis();

  while (WiFi.status() != WL_CONNECTED && millis () -
startAttemptTime < WIFI_TIMEOUT_MS){
    Serial.print(".");
    delay(500);
   }

   if(WiFi.status() != WL_CONNECTED){
    Serial.println("Failed!");
   }
   else {
    Serial.print("Connected");
    Serial.println(WiFi.localIP());
   }
}

// This functions is executed when some device publishes a message to a
topic that your ESP8266 is subscribed to
// Change the function below to add logic to your program, so when a
device publishes a message to a topic that
// your ESP8266 is subscribed you can actually do something
void callback(String topic, byte* message, unsigned int length) {
  Serial.print("Message arrived on topic: ");
  Serial.print(topic);
  Serial.print(". Message: ");
  String messageTemp;

  for (int i = 0; i < length; i++) {
    Serial.print((char)message[i]);
    messageTemp += (char)message[i];
  }
  Serial.println();

  // Feel free to add more if statements to control more GPIOs with MQTT

  // If a message is received on the topic home/office/esp1/gpio2, you check if the message is either 1 or 0. Turns the ESP GPIO according to the message
  if(topic=="esp32/4"){
      Serial.print("Changing GPIO 4 to ");
      if(messageTemp == "1"){
        digitalWrite(ledGPIO4, HIGH);
        Serial.print("On");
      }
      else if(messageTemp == "0"){
        digitalWrite(ledGPIO4, LOW);
        Serial.print("Off");
      }
  }
  if(topic=="esp32/5"){
      Serial.print("Changing GPIO 5 to ");
      if(messageTemp == "1"){
        digitalWrite(ledGPIO5, HIGH);
        Serial.print("On");
      }
      else if(messageTemp == "0"){
        digitalWrite(ledGPIO5, LOW);
        Serial.print("Off");
      }
  }
  Serial.println();
}

// This functions reconnects your ESP8266 to your MQTT broker
// Change the function below if you want to subscribe to more topics with your ESP8266
```

```
void reconnect() {
  // Loop until we're reconnected
  while (!client.connected()) {
    Serial.print("Attempting MQTT connection...");
    // Attempt to connect
     /*
     YOU  NEED TO CHANGE THIS NEXT LINE, IF YOU'RE HAVING PROBLEMS WITH MQTT MULTIPLE CONNECTIONS
     To change the ESP device ID, you will have to give a unique name to the ESP8266.
     Here's how it looks like now:
       if (client.connect("ESP8266Client")) {
     If you want more devices connected to the MQTT broker, you can do it like this:
       if (client.connect("ESPOffice")) {
     Then, for the other ESP:
       if (client.connect("ESPGarage")) {
      That should solve your MQTT multiple connections problem

     THE SECTION IN loop() function should match your device name
    */
    if (client.connect("ESP32Client")) {
      Serial.println("connected");
      // Subscribe or resubscribe to a topic
      // You can subscribe to more topics (to control more LEDs in this example)
      client.subscribe("esp32/4");
      client.subscribe("esp32/5");
    } else {
      Serial.print("failed, rc=");
      Serial.print(client.state());
      Serial.println("try again in 5 seconds");
      // Wait 5 seconds before retrying
      delay(5000);
    }
  }
}

// The setup function sets your ESP GPIOs to Outputs, starts the serial communication at a baud rate of 115200
// Sets your mqtt broker and sets the callback function
// The callback function is what receives messages and actually controls the LEDs
void setup() {
  pinMode(ledGPIO4, OUTPUT);
  pinMode(ledGPIO5, OUTPUT);

  Serial.begin(115200);
  connectToWiFi();
  client.setServer(mqtt_server, 1883);
  client.setCallback(callback);
}

// For this project, you don't need to change anything in the loop function.
// Basically it ensures that you ESP is connected to your broker
void loop() {
  if (!client.connected()) {
    reconnect();
  }
  if(!client.loop())
     /*
     YOU  NEED TO CHANGE THIS NEXT LINE, IF YOU'RE HAVING PROBLEMS WITH MQTT MULTIPLE CONNECTIONS
     To change the ESP device ID, you will have to give a unique name to the ESP8266.
     Here's how it looks like now:
       client.connect("ESP8266Client");
     If you want more devices connected to the MQTT broker, you can do it like this:
       client.connect("ESPOffice");
     Then, for the other ESP:
       client.connect("ESPGarage");
      That should solve your MQTT multiple connections problem

     THE SECTION IN recionnect() function should match your device name
    */
    client.connect("ESP32Client");
}
```

3. Pembuatan Python Script MQTT di Visual Studio Code
   Ini adalah skrip inti dari aplikasi kami. Ini mengatur server web dan ketika tombol-tombol ini ditekan, ia menerbitkan pesan MQTT ke ESP32.

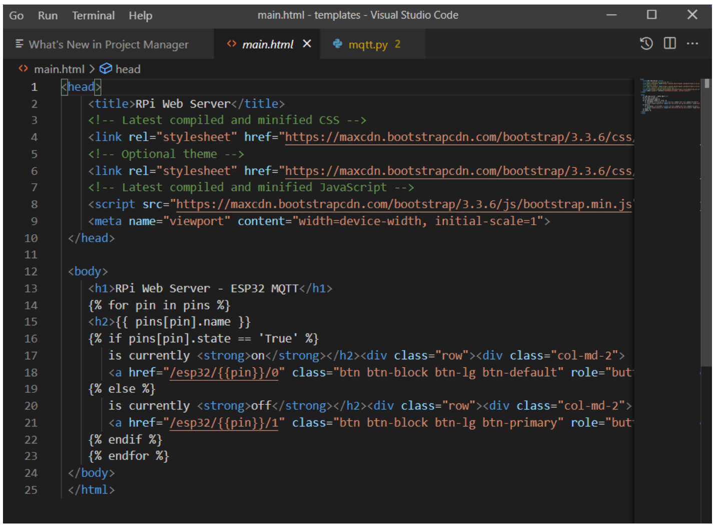

Go
Run
Terminal
Help
mqtt.py - templates - Visual Studio Code
What's New in Project Manager
main.html
mqtt.py 2
D: > dok > SEM 6 > prak. Teknik Kendali > Latihan-Hari-1-PR > Latihan-7-PR-Web > mqtt.py > ...

```
#
# Created by Rui Santos
# Complete project details: https://randomnerdtutorials.com
#

import paho.mqtt.client as mqtt
from flask import Flask, render_template, request
app = Flask(__name__)

mqttc=mqtt.Client()
mqttc.connect("10.33.162.50",1883,60)
mqttc.loop_start()

# Create a dictionary called pins to store the pin number, name, and pin state:
pins = {
   4 : {'name' : 'GPIO 4', 'board' : 'esp32', 'topic' : 'esp32/4', 'state' : 'False'
   5 : {'name' : 'GPIO 5', 'board' : 'esp32', 'topic' : 'esp32/5', 'state' : 'False'
   }

# Put the pin dictionary into the template data dictionary:
```

```
#
# Created by Rui Santos
# Complete project details: https://randomnerdtutorials.com
#

import paho.mqtt.client as mqtt
from flask import Flask, render_template, request
app = Flask(__name__)

mqttc=mqtt.Client()
mqttc.connect("10.33.162.50",1883,60)
mqttc.loop_start()

# Create a dictionary called pins to store the pin number, name, and pin state:
pins = {
   4 : {'name' : 'GPIO 4', 'board' : 'esp32', 'topic' : 'esp32/4', 'state' : 'False'},
   5 : {'name' : 'GPIO 5', 'board' : 'esp32', 'topic' : 'esp32/5', 'state' : 'False'}
   }

# Put the pin dictionary into the template data dictionary:
templateData = {
   'pins' : pins
   }

@app.route("/")
def main():
   # Pass the template data into the template main.html and return it to the user
   return render_template('main.html', **templateData)

# The function below is executed when someone requests a URL with the pin number and action in it:
```

```
@app.route("/<board>/<changePin>/<action>")

def action(board, changePin, action):
  # Convert the pin from the URL into an integer:
  changePin = int(changePin)
  # Get the device name for the pin being changed:
  devicePin = pins[changePin]['name']
  # If the action part of the URL is "on," execute the code indented below:
  if action == "1" and board == 'esp32':
    mqttc.publish(pins[changePin]['topic'],"1")
    pins[changePin]['state'] = 'True'

  if action == "0" and board == 'esp32':
    mqttc.publish(pins[changePin]['topic'],"0")
    pins[changePin]['state'] = 'False'

  # Along with the pin dictionary, put the message into the template data dictionary:
  templateData = {
    'pins' : pins
  }

  return render_template('main.html', **templateData)

if __name__ == "__main__":
  app.run(host='0.0.0.0', port=8080, debug=False)
```

4. Pembuatan File HTML


Go
Run
Terminal
Help
main.html - templates - Visual Studio Code
What's New in Project Manager
main.html
mqtt.py 2
main.html > head

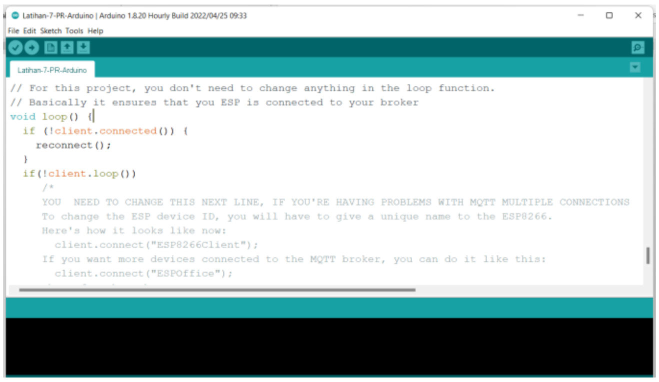
```
<head>
    <title>RPi Web Server</title>
    <!-- Latest compiled and minified CSS -->
    <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.6/css,
    <!-- Optional theme -->
    <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.6/css,
    <!-- Latest compiled and minified JavaScript -->
    <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.6/js/bootstrap.min.js
    <meta name="viewport" content="width=device-width, initial-scale=1">
</head>

<body>
    <h1>RPi Web Server - ESP32 MQTT</h1>
    {% for pin in pins %}
    <h2>{{ pins[pin].name }}
    {% if pins[pin].state == 'True' %}
        is currently <strong>on</strong></h2><div class="row"><div class="col-md-2">
        <a href="/esp32/{{pin}}/0" class="btn btn-block btn-lg btn-default" role="but
    {% else %}
        is currently <strong>off</strong></h2><div class="row"><div class="col-md-2">
        <a href="/esp32/{{pin}}/1" class="btn btn-block btn-lg btn-primary" role="but
    {% endif %}
    {% endfor %}
</body>
</html>
```

```
<head>
   <title>RPi Web Server</title>
   <!-- Latest compiled and minified CSS -->
```

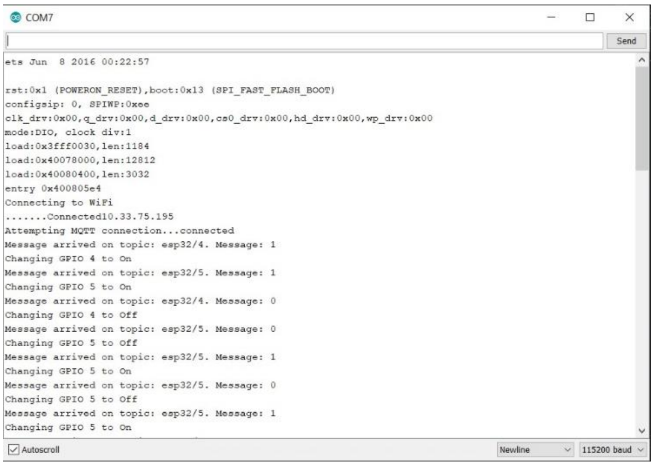
```
    <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.6/css/bootstrap.min.css" integrity="sha384-1q8mTJOASx8j1Au+a5WDVnPi2lkFfwwEAa8hDDdjZlpLegxhjVME1fgjWPGmkzs7" crossorigin="anonymous">
    <!-- Optional theme -->
    <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.6/css/bootstrap-theme.min.css" integrity="sha384-fLW2N01lMqjakBkx3l/M9EahuwpSfeNvV63J5ezn3uZzapT0u7EYsXMjQV+0En5r" crossorigin="anonymous">
    <!-- Latest compiled and minified JavaScript -->
    <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.6/js/bootstrap.min.js" integrity="sha384-0mSbJDEHialfmuBBQP6A4Qrprq5OVfW37PRR3j5ELqxss1yVqOtnepnHVP9aJ7xS" crossorigin="anonymous"></script>
    <meta name="viewport" content="width=device-width, initial-scale=1">
 </head>

 <body>
    <h1>RPi Web Server - ESP32 MQTT</h1>
    {% for pin in pins %}
    <h2>{{ pins[pin].name }}
    {% if pins[pin].state == 'True' %}
       is currently <strong>on</strong></h2><div class="row"><div class="col-md-2">
       <a href="/esp32/{{pin}}/0" class="btn btn-block btn-lg btn-default" role="button">Turn off</a></div></div>
    {% else %}
       is currently <strong>off</strong></h2><div class="row"><div class="col-md-2">
       <a href="/esp32/{{pin}}/1" class="btn btn-block btn-lg btn-primary" role="button">Turn on</a></div></div>
    {% endif %}
    {% endfor %}
 </body>
 </html>
```

✓ Upload Program dan Launch the Web Server

1. Upload ESP32 Program Code di Arduino IDE

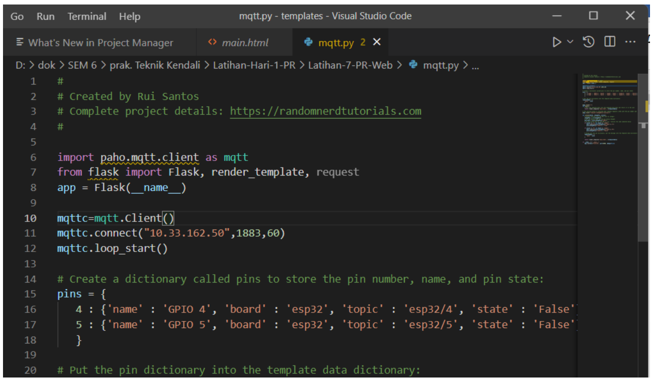

Latihan-7-PR-Arduino | Arduino 1.8.20 Hourly Build 2022/04/25 09:33
File Edit Sketch Tools Help
Latihan-7-PR-Arduino

```
// For this project, you don't need to change anything in the loop function.
// Basically it ensures that you ESP is connected to your broker
void loop() {
  if (!client.connected()) {
    reconnect();
  }
  if(!client.loop())
     /*
     YOU  NEED TO CHANGE THIS NEXT LINE, IF YOU'RE HAVING PROBLEMS WITH MQTT MULTIPLE CONNECTIONS
     To change the ESP device ID, you will have to give a unique name to the ESP8266.
     Here's how it looks like now:
       client.connect("ESP8266Client");
     If you want more devices connected to the MQTT broker, you can do it like this:
       client.connect("ESPOffice");
```

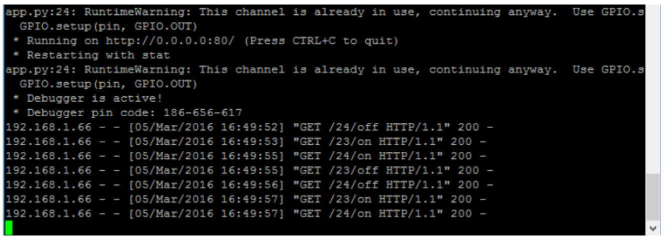


```
ets Jun  8 2016 00:22:57

rst:0x1 (POWERON_RESET),boot:0x13 (SPI_FAST_FLASH_BOOT)
configsip: 0, SPIWP:0xee
clk_drv:0x00,q_drv:0x00,d_drv:0x00,cs0_drv:0x00,hd_drv:0x00,wp_drv:0x00
mode:DIO, clock div:1
load:0x3fff0030,len:1184
load:0x40078000,len:12812
load:0x40080400,len:3032
entry 0x400805e4
Connecting to WiFi
.......Connected10.33.75.195
Attempting MQTT connection...connected
Message arrived on topic: esp32/4. Message: 1
Changing GPIO 4 to On
Message arrived on topic: esp32/5. Message: 1
Changing GPIO 5 to On
Message arrived on topic: esp32/4. Message: 0
Changing GPIO 4 to Off
Message arrived on topic: esp32/5. Message: 0
Changing GPIO 5 to Off
Message arrived on topic: esp32/5. Message: 1
Changing GPIO 5 to On
Message arrived on topic: esp32/5. Message: 0
Changing GPIO 5 to Off
Message arrived on topic: esp32/5. Message: 1
Changing GPIO 5 to On
```

2. Run mqtt program with pyhton app.py

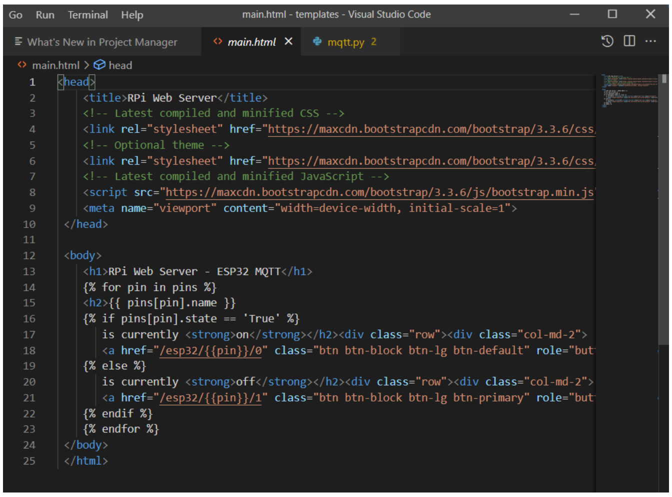

Go Run Terminal Help
mqtt.py - templates - Visual Studio Code
What's New in Project Manager
main.html
mqtt.py 2
D: > dok > SEM 6 > prak. Teknik Kendali > Latihan-Hari-1-PR > Latihan-7-PR-Web > mqtt.py > ...

```
#
# Created by Rui Santos
# Complete project details: https://randomnerdtutorials.com
#

import paho.mqtt.client as mqtt
from flask import Flask, render_template, request
app = Flask(__name__)

mqttc=mqtt.Client()
mqttc.connect("10.33.162.50",1883,60)
mqttc.loop_start()

# Create a dictionary called pins to store the pin number, name, and pin state:
pins = {
   4 : {'name' : 'GPIO 4', 'board' : 'esp32', 'topic' : 'esp32/4', 'state' : 'False'
   5 : {'name' : 'GPIO 5', 'board' : 'esp32', 'topic' : 'esp32/5', 'state' : 'False'
   }

# Put the pin dictionary into the template data dictionary:
```



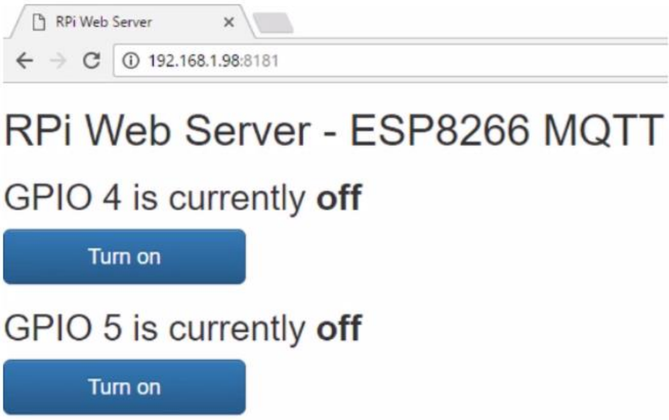
```
app.py:24: RuntimeWarning: This channel is already in use, continuing anyway.  Use GPIO.s
  GPIO.setup(pin, GPIO.OUT)
 * Running on http://0.0.0.0:80/ (Press CTRL+C to quit)
 * Restarting with stat
app.py:24: RuntimeWarning: This channel is already in use, continuing anyway.  Use GPIO.s
  GPIO.setup(pin, GPIO.OUT)
 * Debugger is active!
 * Debugger pin code: 186-656-617
192.168.1.66 - - [05/Mar/2016 16:49:52] "GET /24/off HTTP/1.1" 200 -
192.168.1.66 - - [05/Mar/2016 16:49:53] "GET /23/on HTTP/1.1" 200 -
192.168.1.66 - - [05/Mar/2016 16:49:55] "GET /24/on HTTP/1.1" 200 -
192.168.1.66 - - [05/Mar/2016 16:49:55] "GET /23/off HTTP/1.1" 200 -
192.168.1.66 - - [05/Mar/2016 16:49:56] "GET /24/off HTTP/1.1" 200 -
192.168.1.66 - - [05/Mar/2016 16:49:57] "GET /23/on HTTP/1.1" 200 -
192.168.1.66 - - [05/Mar/2016 16:49:57] "GET /24/on HTTP/1.1" 200 -
```

3. Run HTML code di Visual Studio Code


Go Run Terminal Help
main.html - templates - Visual Studio Code
What's New in Project Manager
main.html
mqtt.py 2
main.html > head

```
<head>
    <title>RPi Web Server</title>
    <!-- Latest compiled and minified CSS -->
    <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.6/css/
    <!-- Optional theme -->
    <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.6/css/
    <!-- Latest compiled and minified JavaScript -->
    <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.6/js/bootstrap.min.js
    <meta name="viewport" content="width=device-width, initial-scale=1">
</head>

<body>
    <h1>RPi Web Server - ESP32 MQTT</h1>
    {% for pin in pins %}
    <h2>{{ pins[pin].name }}
    {% if pins[pin].state == 'True' %}
        is currently <strong>on</strong></h2><div class="row"><div class="col-md-2">
        <a href="/esp32/{{pin}}/0" class="btn btn-block btn-lg btn-default" role="but
    {% else %}
        is currently <strong>off</strong></h2><div class="row"><div class="col-md-2">
        <a href="/esp32/{{pin}}/1" class="btn btn-block btn-lg btn-primary" role="but
    {% endif %}
    {% endfor %}
</body>
</html>
```

4. Demonstrasi Web Server



RPi Web Server - ESP8266 MQTT

GPIO 4 is currently **off**

GPIO 5 is currently **off**

Untuk membuka alamat web server yang dibuat, kami menggukan ip addres dan penyesuaian port yang dipakai.

kedua button "**Turn on**" bisa dioperasikan untuk connect ke rangkaian ESP32 yang memberikan perintah on/off pada kedua LED yang tersambung.

Status "**Turn on**"dan "**off**" pada web akan berubah menjadi "**Turn Off**"dan "**on**"jika kami tekan tombol **Turn on.**

2. IOT Cloud provider/service

A. IOT Cloud provider/service with ThingSpeak

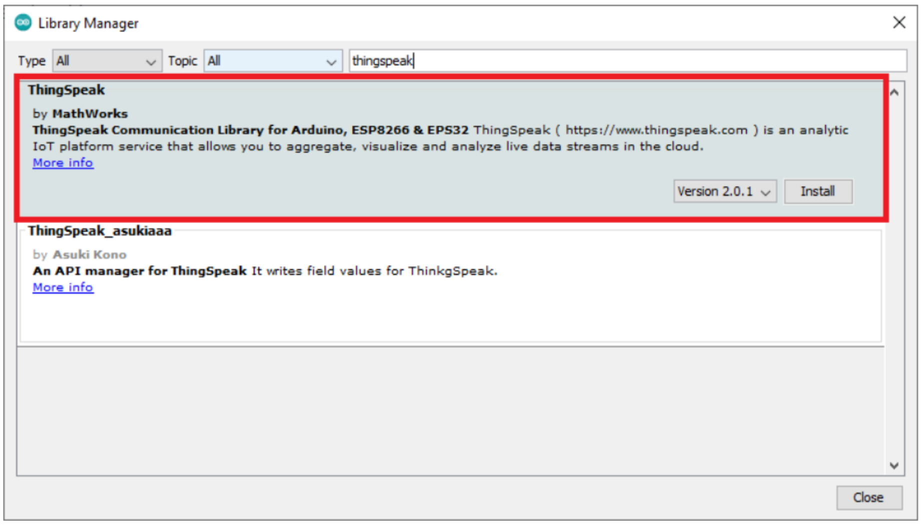
✓ **Komponen dan Ekstensi aplikasi yang digunakan**

1. Komponen yang digunakan
    - ESP32 board dengan chip ESP-WROOM-32
    - DHT11 Sensor module
    - Kabel jumper
2. Ekstensi aplikasi yang diperlukan

    A. Install ThingSpeak Library di Arduino IDE

    Untuk mengirim pembacaan sensor ke ThingSpeak, kami akan menggunakan library thingspeak-arduino. Library ini diinstall melalui

Arduino Library Manager. Buka **Sketch** > **Include Library** > **Manage Libraries...** dan cari "ThingSpeak" di Library Manager. Lalu install



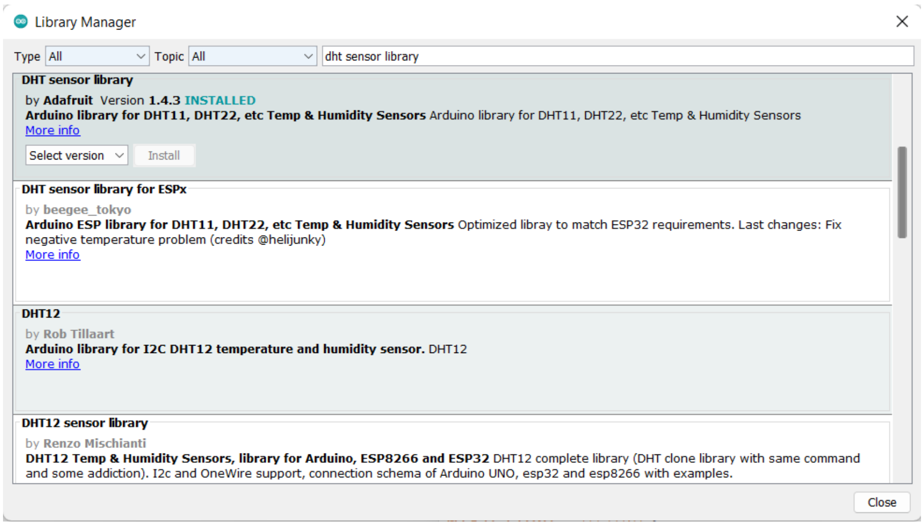
thingspeak

ThingSpeak
by **MathWorks**
ThingSpeak Communication Library for Arduino, ESP8266 & EPS32 ThingSpeak (https://www.thingspeak.com) is an analytic IoT platform service that allows you to aggregate, visualize and analyze live data streams in the cloud.
More info

ThingSpeak_asukiaaa
by **Asuki Kono**
An API manager for ThingSpeak It writes field values for ThinkgSpeak.
More info

B. Install DHT Sensor Library di Arduino IDE

Sensor DHT merupakan paket sensor yang berfungsi untuk mengukur suhu dan kelembaban udara sekaligus yang dialamnya terdapat thermistor tipe NTC (Negative Temperature Coefficient) untuk mengukur suhu, sebuah sensor kelembapan dengan karkteristik resistif terhadap perubahan kadar air di udara serta terdapat chip yang di dalamnya melakukan beberapa konversi analog ke digital dan mengeluarkan output dengan format single-wire bi-directional.



dht sensor library

DHT sensor library
by **Adafruit** Version **1.4.3** INSTALLED
Arduino library for DHT11, DHT22, etc Temp & Humidity Sensors Arduino library for DHT11, DHT22, etc Temp & Humidity Sensors
More info

DHT sensor library for ESPx
by **beegee_tokyo**
Arduino ESP library for DHT11, DHT22, etc Temp & Humidity Sensors Optimized libray to match ESP32 requirements. Last changes: Fix negative temperature problem (credits @helijunky)
More info

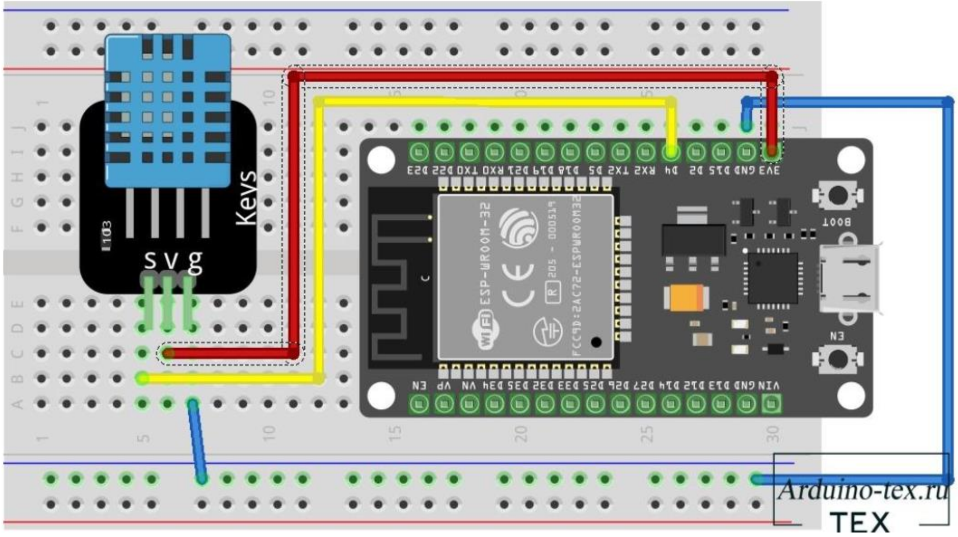
DHT12
by **Rob Tillaart**
Arduino library for I2C DHT12 temperature and humidity sensor. DHT12
More info

DHT12 sensor library
by **Renzo Mischianti**
DHT12 Temp & Humidity Sensors, library for Arduino, ESP8266 and ESP32 DHT12 complete library (DHT clone library with same command and some addiction). I2c and OneWire support, connection schema of Arduino UNO, esp32 and esp8266 with examples.

✓ **Tahap Pengerjaan**

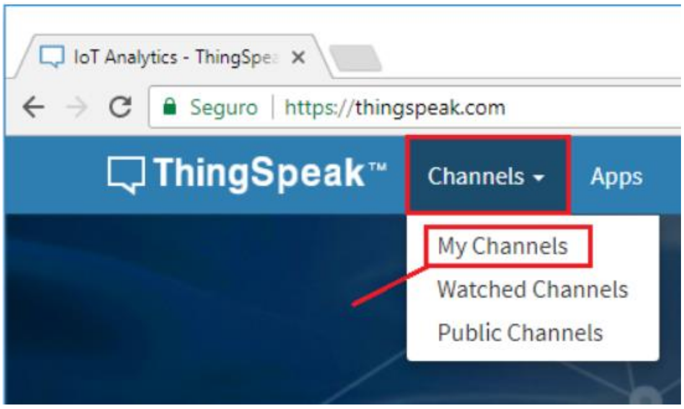
1. Pembuatan skema jalur sensor dan ESP32

   Pembuatan skematik diagram kabel sensor menuju pin ESP323 pada GPIO4, VCC sebesar 3.3V, serta ground

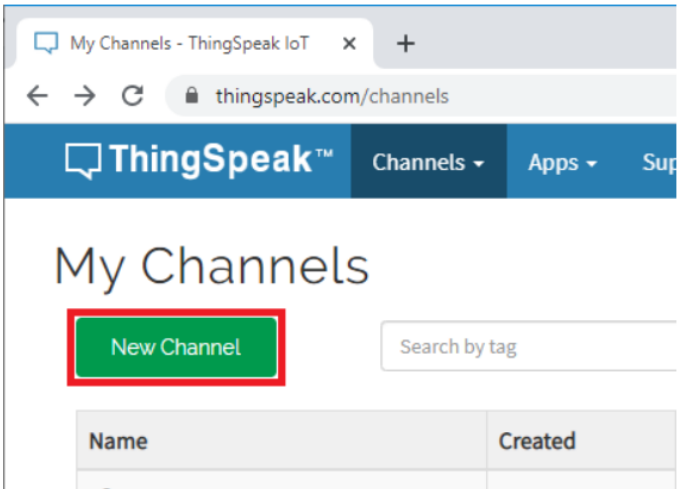


2. Pembuatan Channel pada ThingSpeak.com

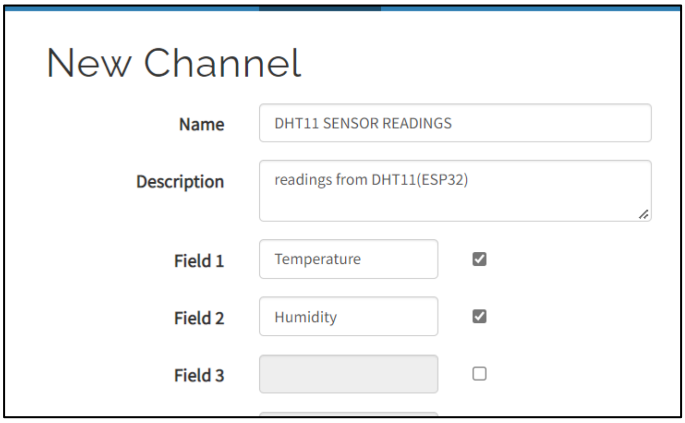
A. open the “**Channels**” tab and select “**My Channels**“.



B. Press the “New Channel” button to create a new channel.



C. Ketik nama yang diinginkan dan tambahkan deskripsi. Dalam contoh ini, kami akan mempublikasikan suhu dan kelembaban.



New Channel

Name: DHT11 SENSOR READINGS

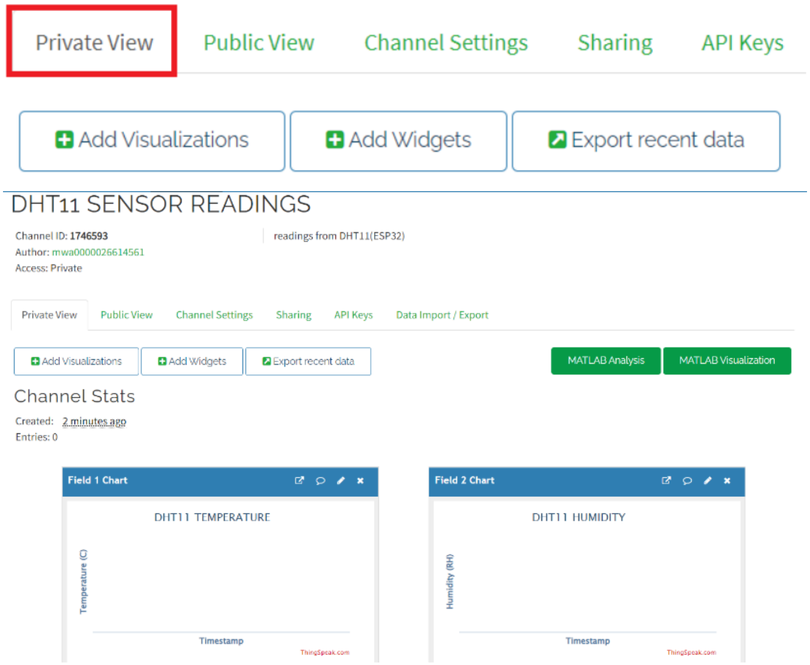
Description: readings from DHT11(ESP32)

Field 1: Temperature ☑

Field 2: Humidity ☑

Field 3: □

D. Customizing Chart
Buat chart sejumlah 2 untuk pengukuran suhu dan kelembaban dengan nilai x adalah waktu



DHT11 SENSOR READINGS

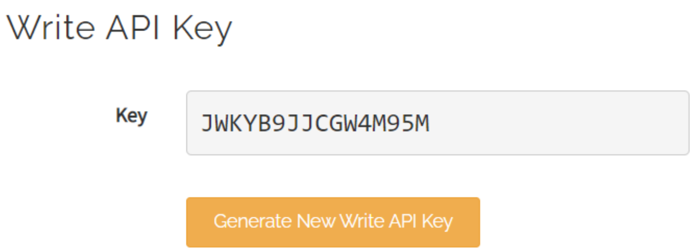
Channel ID: **1746593**
Author: mwa0000026614561
Access: Private

readings from DHT11(ESP32)

Channel Stats

Created: 2 minutes ago
Entries: 0

E. API Keys write for ESP32 code
Salin kode API Key milik channel yang telah dibuat ke Arduino IDE code untuk menerima perintah reading



Write API Key

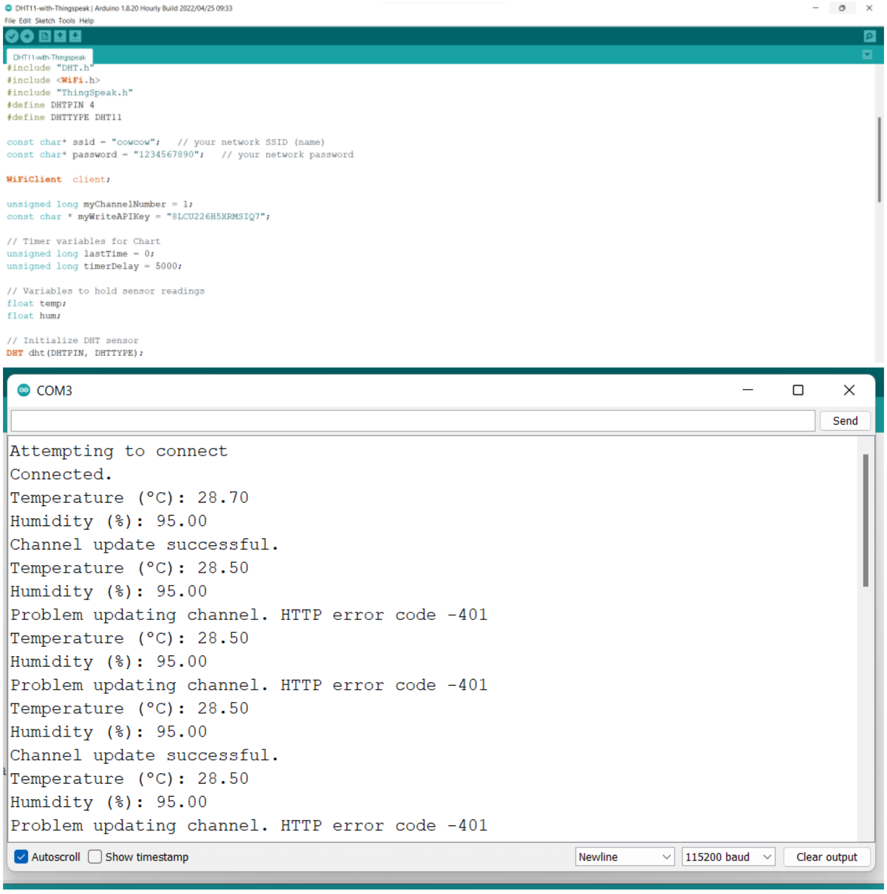
Key: JWKYB9JJCGW4M95M

3. Arduino IDE code ESP32 Publish Sensor Readings to ThingSpeak

```
#include "DHT.h"
#include <WiFi.h>
#include "ThingSpeak.h"
#define DHTPIN 4
#define DHTTYPE DHT11

const char* ssid = "cowcow";   // your network SSID (name)
const char* password = "1234567890";   // your network password

WiFiClient  client;

unsigned long myChannelNumber = 1;
const char * myWriteAPIKey = "8LCU226H5XRMSIQ7";

// Timer variables for Chart
unsigned long lastTime = 0;
unsigned long timerDelay = 5000;

// Variables to hold sensor readings
float temp;
float hum;

// Initialize DHT sensor
DHT dht(DHTPIN, DHTTYPE);


void setup() {
  Serial.begin(115200);  //Initialize serial
  dht.begin();

  WiFi.mode(WIFI_STA);

  ThingSpeak.begin(client);  // Initialize ThingSpeak
}

void loop() {

  if ((millis() - lastTime) > timerDelay) {

    // Connect or reconnect to WiFi
    if(WiFi.status() != WL_CONNECTED){
      Serial.print("Attempting to connect");
      while(WiFi.status() != WL_CONNECTED){
        WiFi.begin(ssid, password);
        delay(5000);
      }
      Serial.println("\nConnected.");
    }

    hum = dht.readHumidity();
    Serial.print("Humidity (RH): ");
    Serial.println(hum);
    ThingSpeak.setField(1, hum);

    temp = dht.readTemperature();
    Serial.print("Temperature (ºC): ");
    Serial.println(temp);
    ThingSpeak.setField(2, temp);

    // Check if any reads failed and exit early (to try again).
    if (isnan(temp) || isnan(hum)) {
      Serial.println(F("Failed to read from DHT sensor!"));
      return;
    }

    // Write to ThingSpeak. There are up to 8 fields in a channel,
allowing you to store up to 8 different
    // pieces of information in a channel.  Here, we write to field 1.
    int x = ThingSpeak.writeFields(myChannelNumber,
myWriteAPIKey);
    //uncomment if you want to get temperature in Fahrenheit
    //int x = ThingSpeak.writeField(myChannelNumber, 1,
temperatureF, myWriteAPIKey);

    if(x == 200){
      Serial.println("Channel update successful.");
    }
    else{
      Serial.println("Problem updating channel. HTTP error code " +
String(x));
    }
    lastTime = millis();
  }
}
```

✓ Upload Code and Chart Result in ThingSpeak

1. Upload code and run Serial Monitor ESP32 in Arduino IDE

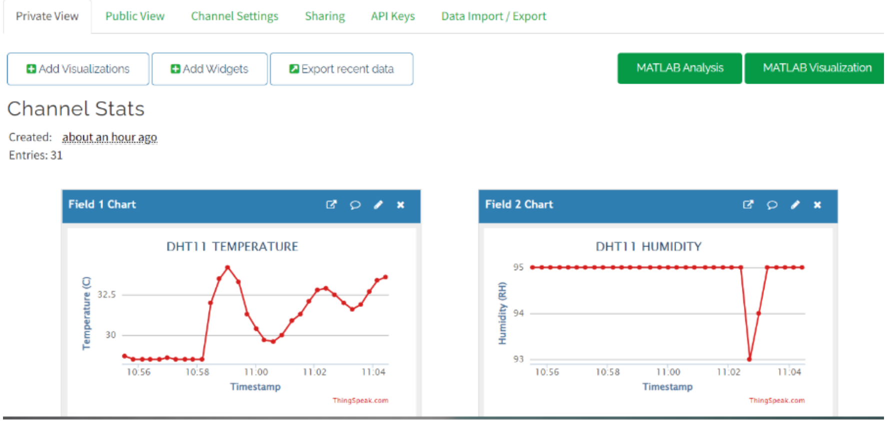

DHT11-with-Thingspeak | Arduino 1.8.20 Hourly Build 2022/04/25 09:33
File Edit Sketch Tools Help
DHT11-with-Thingspeak
COM3
Send
☑ Autoscroll ☐ Show timestamp
Newline
115200 baud
Clear output

```
#include "DHT.h"
#include <WiFi.h>
#include "ThingSpeak.h"
#define DHTPIN 4
#define DHTTYPE DHT11

const char* ssid = "cowcow";   // your network SSID (name)
const char* password = "1234567890";   // your network password

WiFiClient  client;

unsigned long myChannelNumber = 1;
const char * myWriteAPIKey = "8LCU226H5XRMSIQ7";

// Timer variables for Chart
unsigned long lastTime = 0;
unsigned long timerDelay = 5000;

// Variables to hold sensor readings
float temp;
float hum;

// Initialize DHT sensor
DHT dht(DHTPIN, DHTTYPE);
```

```
Attempting to connect
Connected.
Temperature (°C): 28.70
Humidity (%): 95.00
Channel update successful.
Temperature (°C): 28.50
Humidity (%): 95.00
Problem updating channel. HTTP error code -401
Temperature (°C): 28.50
Humidity (%): 95.00
Problem updating channel. HTTP error code -401
Temperature (°C): 28.50
Humidity (%): 95.00
Channel update successful.
Temperature (°C): 28.50
Humidity (%): 95.00
Problem updating channel. HTTP error code -401
```

2. Dynamic Chart Result in ThingSpeak

Berikut adalah ThingSpeak Realtime grafik hasil dari pembacaan sensor DHT11 untuk pengecekan suhu dan kelembaban.



Channel Stats

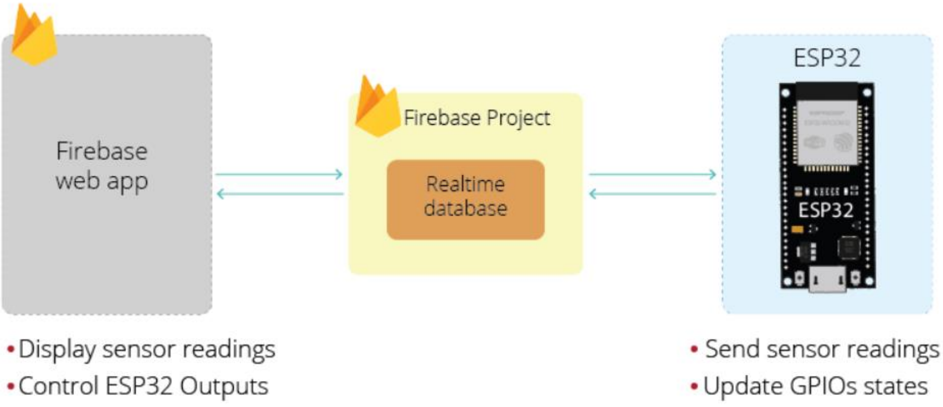
Created: about an hour ago

Entries: 31

B. IOT Cloud provider/service with Firebase

Firebase adalah platform pengembangan aplikasi seluler Google yang mencakup banyak layanan untuk mengelola data dari aplikasi iOS, Android, atau web.

Dalam tutorial ini, kita akan membuat project Firebase dengan database realtime, dan kita akan menggunakan ESP32 untuk menyimpan dan membaca data dari database. ESP32 dapat berinteraksi dengan database dari mana saja di dunia selama terhubung ke internet.



✓ Tahap Pengerjaan

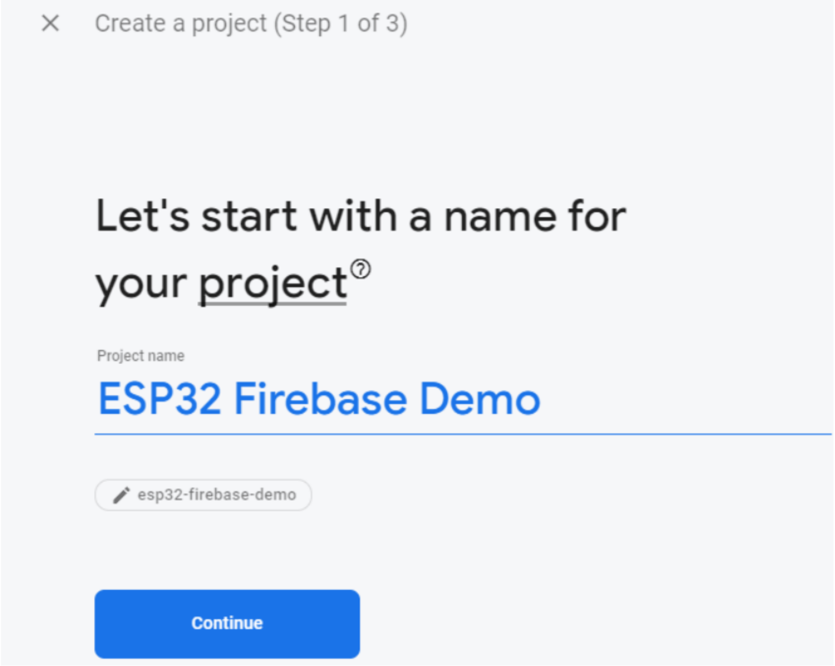
1. Setup Firebase account and create new project

- Create New Project

A. Go to Firebase and sign in using a Google Account.

B. Click *Get Started*, and then *Add project* to create a new project.

C. Give a name to your project, for example: *ESP32 Firebase Demo*.



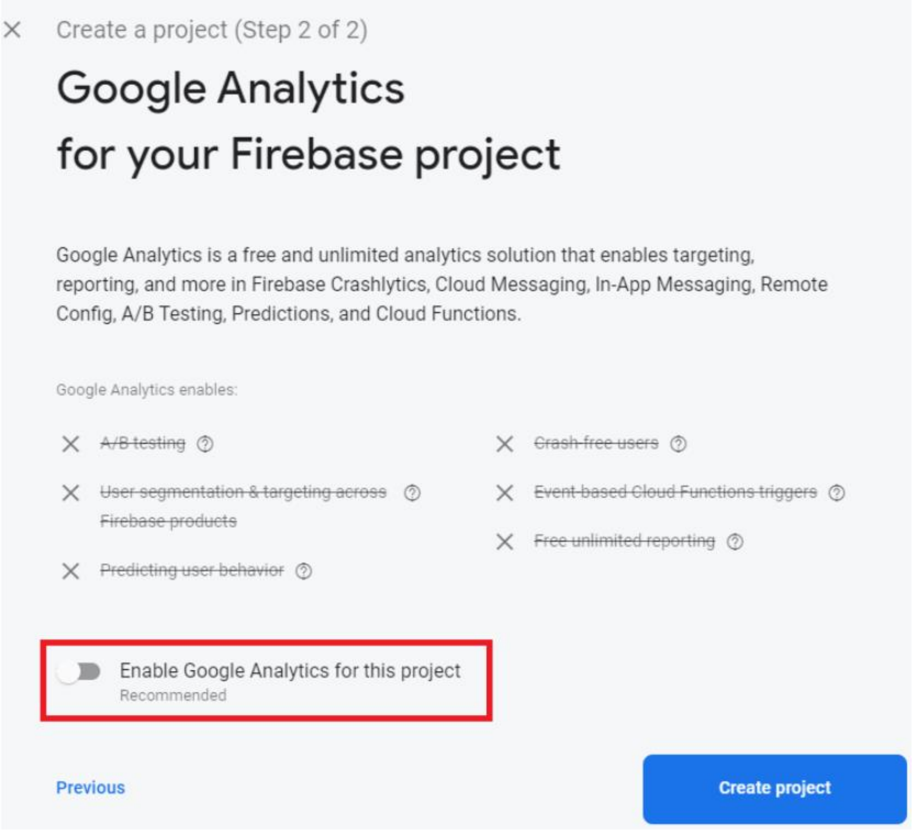
Let's start with a name for your project

Project name

ESP32 Firebase Demo

esp32-firebase-demo

D. Disable the option *Enable Google Analytics* for this project as it is not needed and click *Create project*.



Create a project (Step 2 of 2)

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, Predictions, and Cloud Functions.

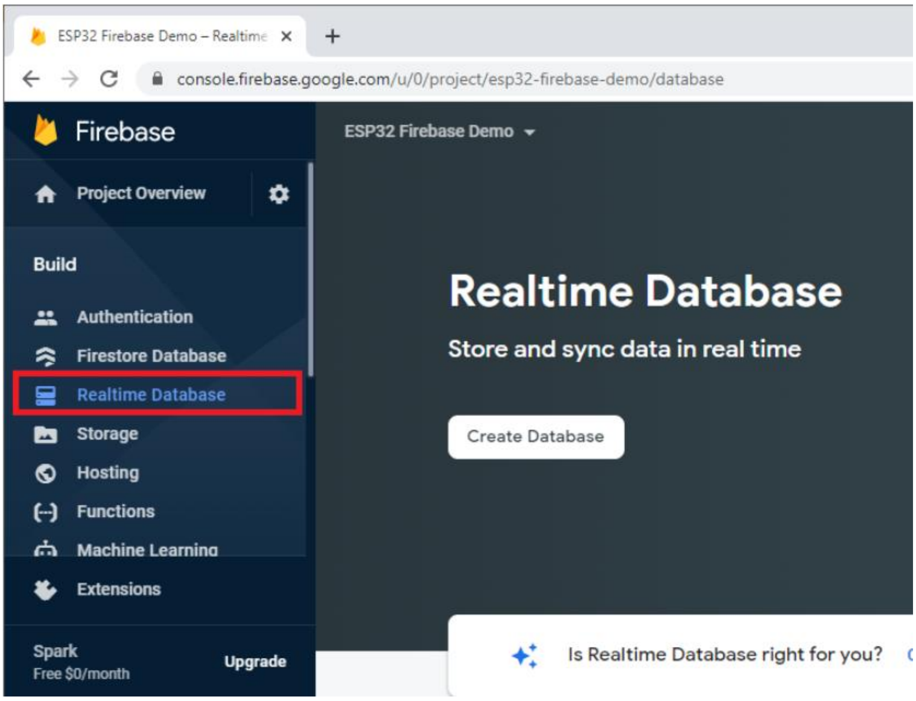
Google Analytics enables:

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

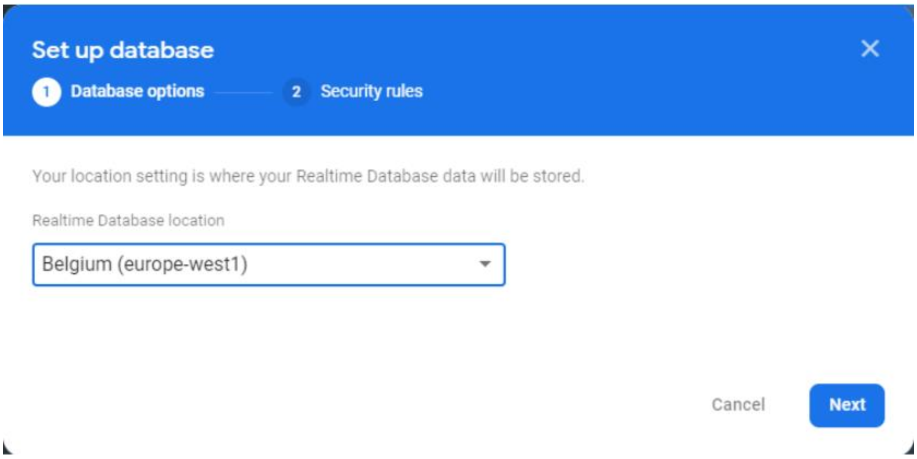
○ Enable Google Analytics for this project
Recommended

- Creating Realtime database

A. On the left sidebar click on *Realtime Database* and then, click on *Create Database*.



B. Select database location. It should be the closest to your location.



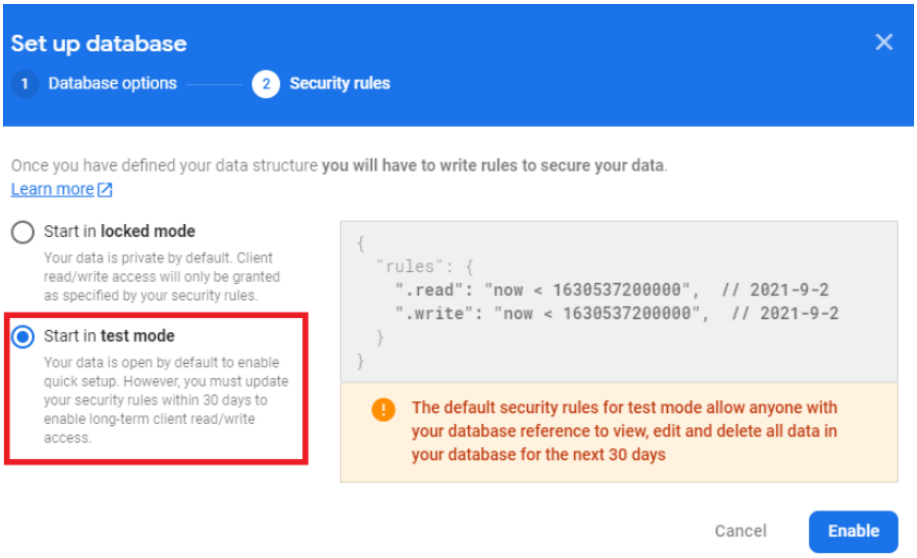
Set up database

1 Database options — 2 Security rules

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

Realtime Database location: Belgium (europe-west1)

C. Set up security rules for database. For testing purposes, select *Start in test mode*. In later tutorials you'll learn how to secure database using database rules.



Set up database

1 Database options — 2 Security rules

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

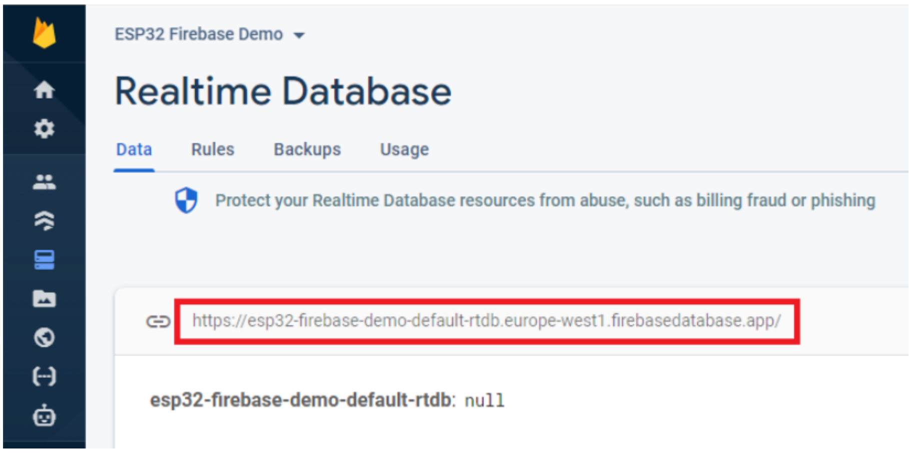
○ Start in **locked mode**
Your data is private by default. Client read/write access will only be granted as specified by your security rules.

● 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 < 1630537200000",  // 2021-9-2
    ".write": "now < 1630537200000",  // 2021-9-2
  }
}
```

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

D. Your database is now created. copy and save the database URL—highlighted in the following image—because you'll need it later in your ESP32 code.



ESP32 Firebase Demo

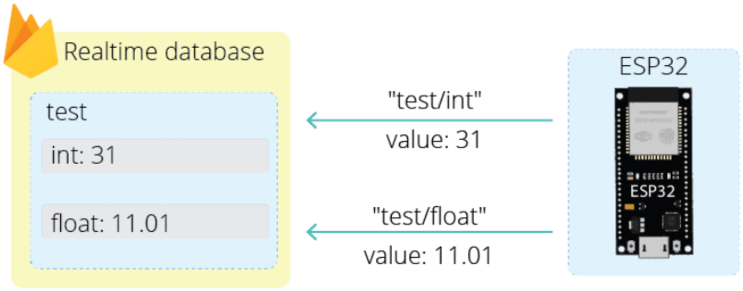
Realtime Database

Protect your Realtime Database resources from abuse, such as billing fraud or phishing

https://esp32-firebase-demo-default-rtdb.europe-west1.firebasedatabase.app/

esp32-firebase-demo-default-rtdb: null

- ESP32 Store Data Program to Firebase database



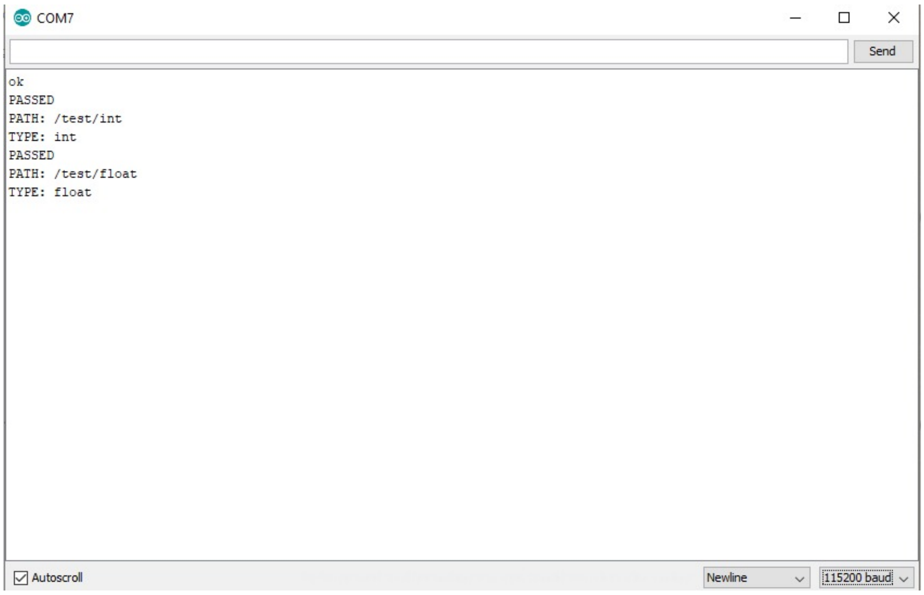
A. Arduino IDE ESP32 Store Data Code

```
#include <Arduino.h>
#include <WiFi.h>
#include <Firebase_ESP_Client.h>

//Provide the token generation process info.
#include "addons/TokenHelper.h"
//Provide the RTDB payload printing info and other helper functions.
#include "addons/RTDBHelper.h"

// Insert your network credentials
#define WIFI_SSID "cowcow"
#define WIFI_PASSWORD "1234567890"

// Insert Firebase project API Key
#define API_KEY "AIzaSyDG9NryMvf4GzvHL8sDzFfwJ3D4-Ar1dSA"

// Insert RTDB URLefine the RTDB URL */
#define DATABASE_URL "https://esp32-firebase-demo-60370-default-rtdb.europe-west1.firebasedatabase.app/"

//Define Firebase Data object
FirebaseData fbdo;

FirebaseAuth auth;
FirebaseConfig config;

unsigned long sendDataPrevMillis = 0;
int count = 0;
bool signupOK = false;

void setup(){
  Serial.begin(115200);
  WiFi.begin(WIFI_SSID, WIFI_PASSWORD);
  Serial.print("Connecting to Wi-Fi");
  while (WiFi.status() != WL_CONNECTED){
    Serial.print(".");
    delay(300);
  }
  Serial.println();
  Serial.print("Connected with IP: ");
  Serial.println(WiFi.localIP());
  Serial.println();

  /* Assign the api key (required) */
  config.api_key = API_KEY;

  /* Assign the RTDB URL (required) */
  config.database_url = DATABASE_URL;

  /* Sign up */
  if (Firebase.signUp(&config, &auth, "", "")){
    Serial.println("ok");
    signupOK = true;
  }
  else{
    Serial.printf("%s\n", config.signer.signupError.message.c_str());
  }

  /* Assign the callback function for the long running token generation
task */
  config.token_status_callback = tokenStatusCallback; //see
addons/TokenHelper.h

  Firebase.begin(&config, &auth);
  Firebase.reconnectWiFi(true);
}

void loop(){
  if (Firebase.ready() && signupOK && (millis() - sendDataPrevMillis
> 15000 || sendDataPrevMillis == 0)){
    sendDataPrevMillis = millis();
    // Write an Int number on the database path test/int
    if (Firebase.RTDB.setInt(&fbdo, "test/int", count)){
      Serial.println("PASSED");
      Serial.println("PATH: " + fbdo.dataPath());
      Serial.println("TYPE: " + fbdo.dataType());
    }
    else {
      Serial.println("FAILED");
      Serial.println("REASON: " + fbdo.errorReason());
    }
    count++;

    // Write an Float number on the database path test/float
    if (Firebase.RTDB.setFloat(&fbdo, "test/float", 0.01 + random(0,100))){
      Serial.println("PASSED");
      Serial.println("PATH: " + fbdo.dataPath());
      Serial.println("TYPE: " + fbdo.dataType());
    }
    else {
      Serial.println("FAILED");
      Serial.println("REASON: " + fbdo.errorReason());
    }
  }
}
```

B. Upload Code for Demonstration

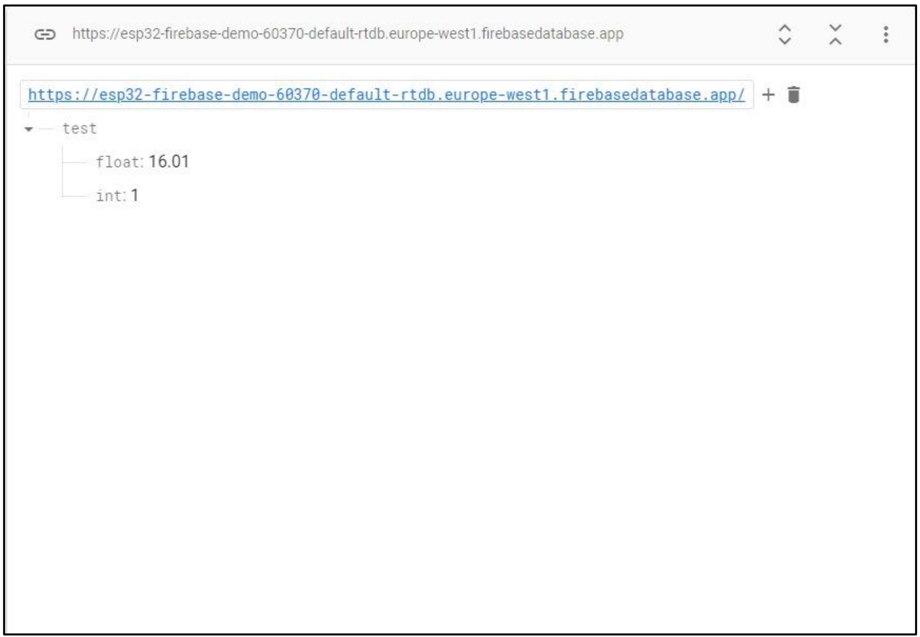
Upload the code to your ESP32 board. Don’t forget to insert your network credentials, database URL path, and the project API key.

After uploading the code, open the Serial Monitor at a baud rate of 115200 and press the ESP32 on-board reset button so it starts running the code.



```
ok
PASSED
PATH: /test/int
TYPE: int
PASSED
PATH: /test/float
TYPE: float
```

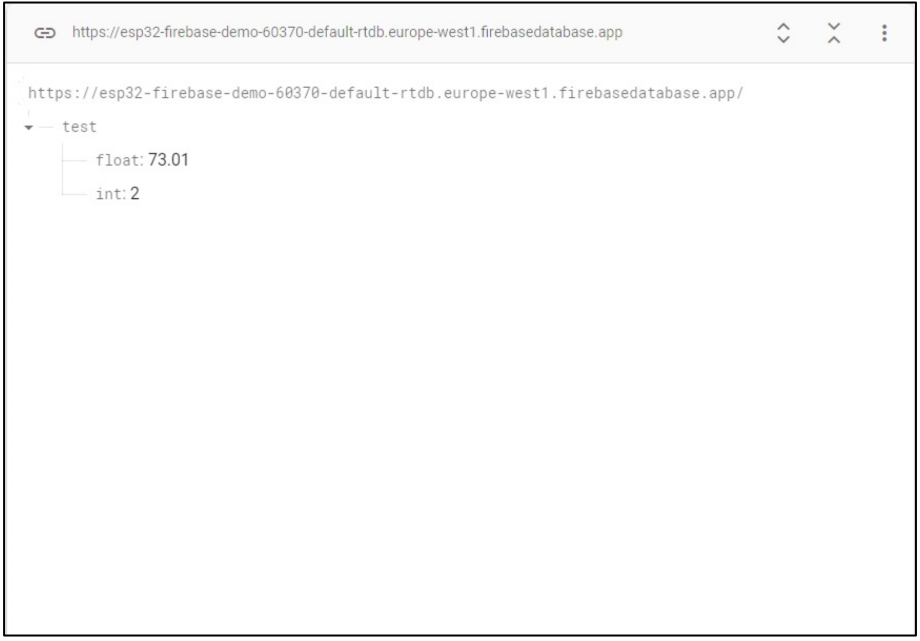
Go to your project's Firebase Realtime database, and you'll see the values saved on the different node paths. Every 15 seconds, it saves a new value. The database blinks when new values are saved.



https://esp32-firebase-demo-60370-default-rtdb.europe-west1.firebasedatabase.app

https://esp32-firebase-demo-60370-default-rtdb.europe-west1.firebasedatabase.app/

```
test
    float: 16.01
    int: 1
```

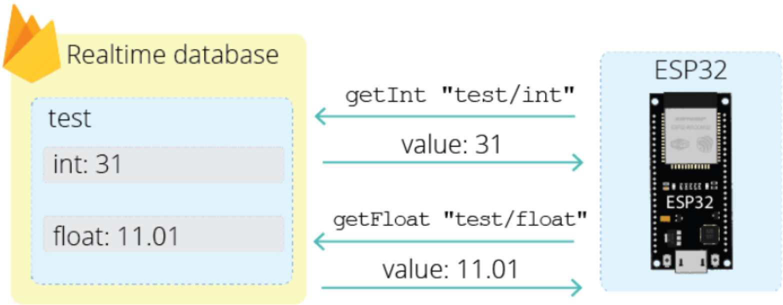


https://esp32-firebase-demo-60370-default-rtdb.europe-west1.firebasedatabase.app

https://esp32-firebase-demo-60370-default-rtdb.europe-west1.firebasedatabase.app/

```
test
    float: 73.01
    int: 2
```

- ESP32 Read from Firebase database

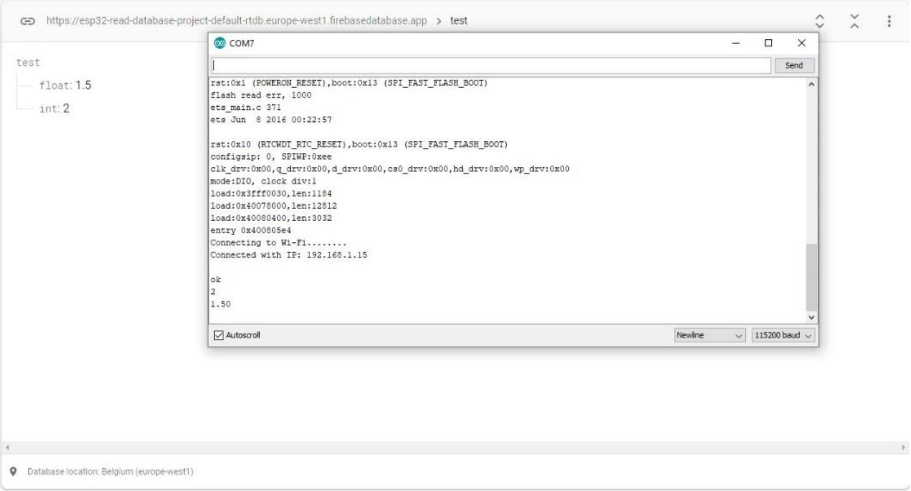


A. Arduino IDE Code

```
#include <Arduino.h>
#include <WiFi.h>
#include <Firebase_ESP_Client.h>

//Provide the token generation process info.
#include "addons/TokenHelper.h"
//Provide the RTDB payload printing info and other helper functions.
#include "addons/RTDBHelper.h"

// Insert your network credentials
#define WIFI_SSID "cowcow"
#define WIFI_PASSWORD "1234567890"

// Insert Firebase project API Key
#define API_KEY "AIzaSyDG9NryMvf4GzvHL8sDzFfwJ3D4"

// Insert RTDB URLefine the RTDB URL */
#define DATABASE_URL "https://esp32-firebase-demo-60370-
default-rtdb.europe-west1.firebasedatabase.app/"

//Define Firebase Data object
FirebaseData fbdo;

FirebaseAuth auth;
FirebaseConfig config;

unsigned long sendDataPrevMillis = 0;
int intValue;
float floatValue;
bool signupOK = false;

void setup() {
  Serial.begin(115200);
  WiFi.begin(WIFI_SSID, WIFI_PASSWORD);
  Serial.print("Connecting to Wi-Fi");
  while (WiFi.status() != WL_CONNECTED) {
    Serial.print(".");
    delay(300);
  }
  Serial.println();
  Serial.print("Connected with IP: ");
  Serial.println(WiFi.localIP());
  Serial.println();

  /* Assign the api key (required) */
  config.api_key = API_KEY;

  /* Assign the RTDB URL (required) */
  config.database_url = DATABASE_URL;

  /* Sign up */
  if (Firebase.signUp(&config, &auth, "", "")) {
    Serial.println("ok");
    signupOK = true;
  }
  else {
    Serial.printf("%s\n", config.signer.signupError.message.c_str());
  }

  /* Assign the callback function for the long running token generation
task */
  config.token_status_callback = tokenStatusCallback; //see
addons/TokenHelper.h

  Firebase.begin(&config, &auth);
  Firebase.reconnectWiFi(true);
}

void loop() {
  if (Firebase.ready() && signupOK && (millis() - sendDataPrevMillis
> 15000 || sendDataPrevMillis == 0)) {
    sendDataPrevMillis = millis();
    if (Firebase.RTDB.getInt(&fbdo, "/test/int")) {
      if (fbdo.dataType() == "int") {
        intValue = fbdo.intData();
        Serial.println(intValue);
      }
    }
    else {
      Serial.println(fbdo.errorReason());
    }

    if (Firebase.RTDB.getFloat(&fbdo, "/test/float")) {
      if (fbdo.dataType() == "float") {
        floatValue = fbdo.floatData();
        Serial.println(floatValue);
      }
    }
    else {
      Serial.println(fbdo.errorReason());
    }
  }
}
```

B. Upload Code for Demonstration

Upload the code to your board. Then, open the Serial Monitor at a baud rate of 115200. After a few seconds, it will print the values saved on the database.


https://esp32-read-database-project-default-rtdb.europe-west1.firebasedatabase.app > test
COM7
Send
Database location: Belgium (europe-west1)

```
test
    float: 1.5
    int: 2
```

```
rst:0x1 (POWERON_RESET),boot:0x13 (SPI_FAST_FLASH_BOOT)
flash read err, 1000
ets_main.c 371
ets Jun  8 2016 00:22:57

rst:0x10 (RTCWDT_RTC_RESET),boot:0x13 (SPI_FAST_FLASH_BOOT)
configsip: 0, SPIWP:0xee
clk_drv:0x00,q_drv:0x00,d_drv:0x00,cs0_drv:0x00,hd_drv:0x00,wp_drv:0x00
mode:DIO, clock div:1
load:0x3fff0030,len:1184
load:0x40078000,len:12812
load:0x40080400,len:3032
entry 0x400805e4
Connecting to Wi-Fi........
Connected with IP: 192.168.1.15

ok
2
1.50
```

☑ Autoscroll   Newline   115200 baud