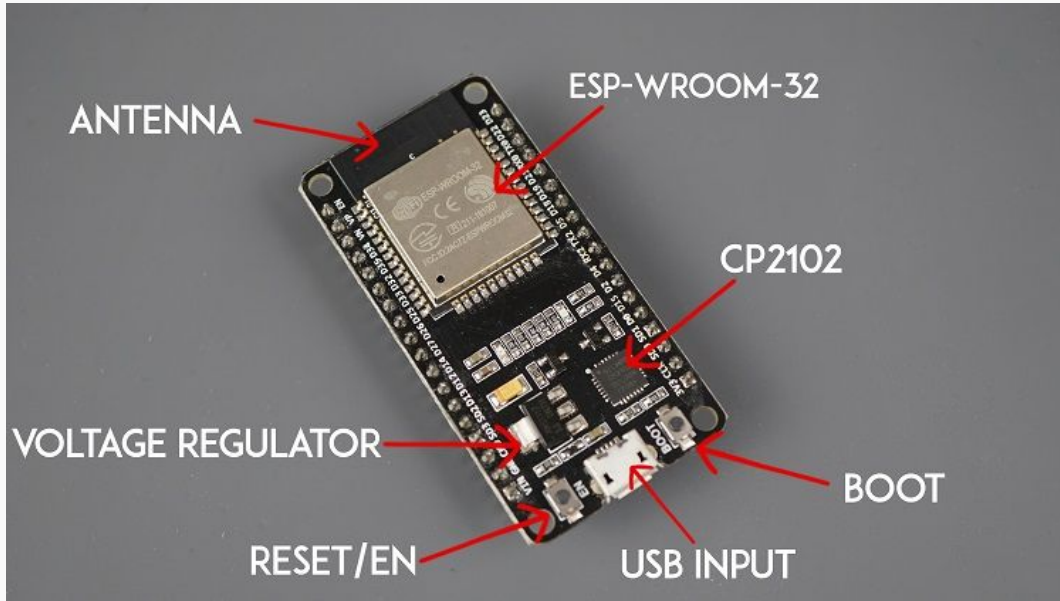


# SIC Batch 5

Week 3 - Hands On ESP32 & API

# ESP32



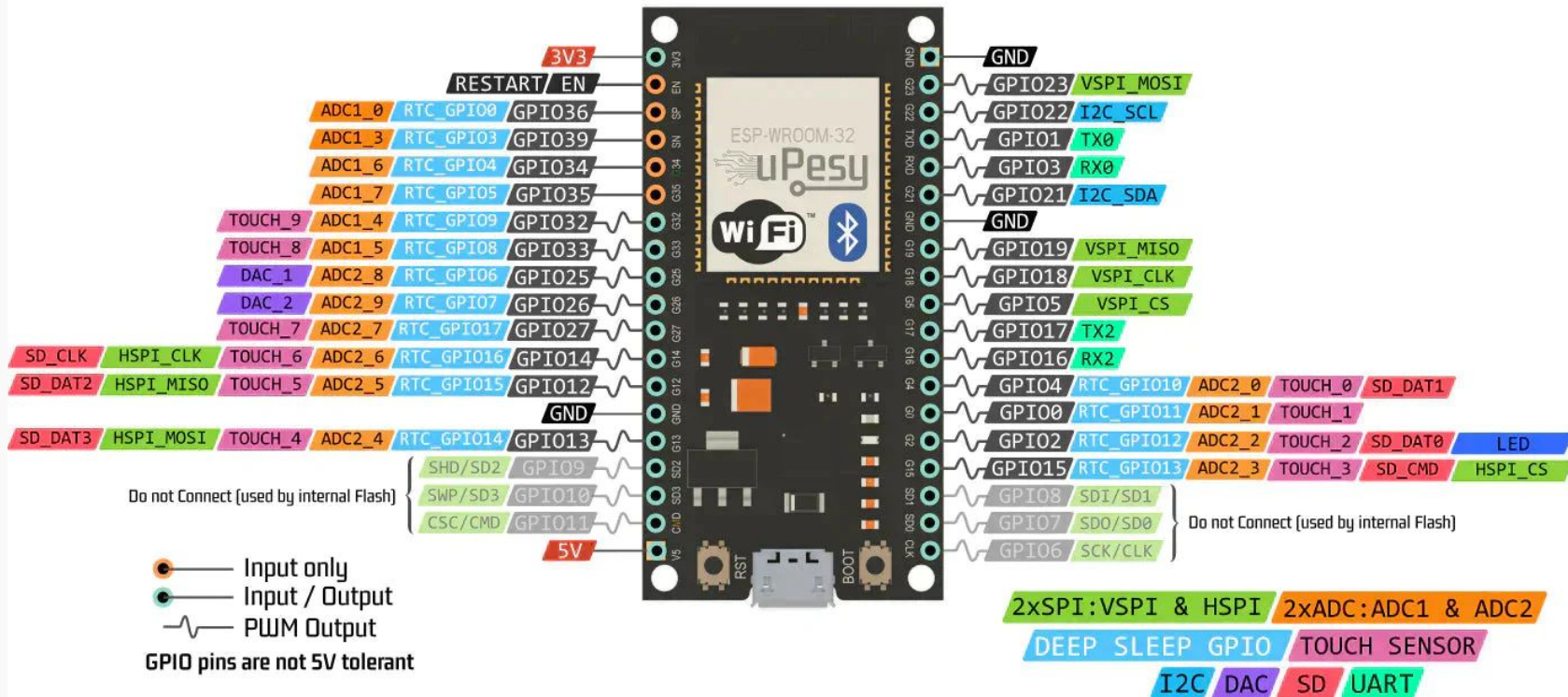
ESP32 adalah mikrokontroler yang dikembangkan oleh Espressif Systems, yang telah menjadi pilihan populer di dunia pengembangan perangkat IoT (Internet of Things).

Dengan berbagai fitur yang canggih, ESP32 menawarkan kemampuan koneksi nirkabel yang kuat, sumber daya yang lebih besar, dan fleksibilitas yang tinggi.

# ESP32 PINOUT

Notes: you don't need to remember all of the pins :)

## ESP32 Wroom DevKit Full Pinout



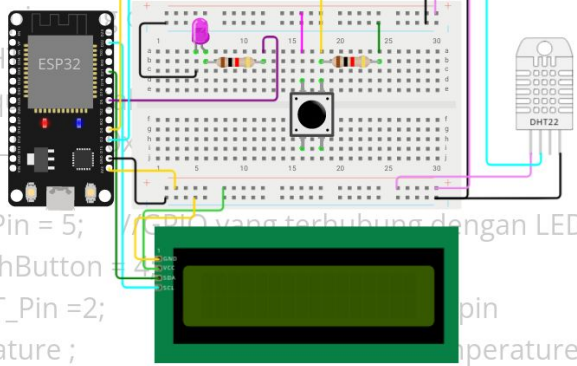
# Wiring Diagram with Wokwi

sketch.ino

```
//Inisiasi nilai
#include <DHT.h>
#include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd(0x27, 16, 2);

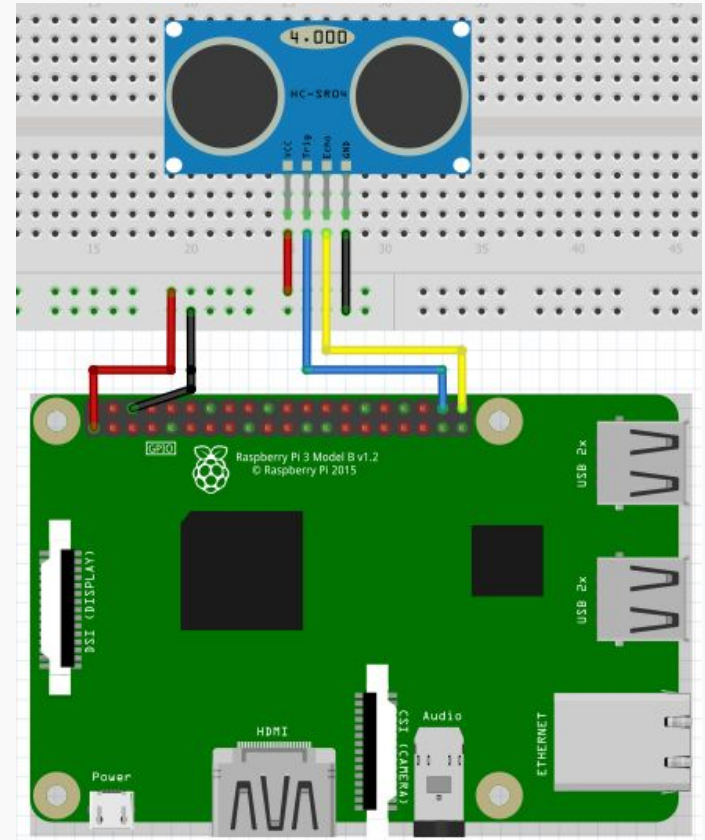
const int ledPin = 5;
const int pushButton = 2;
const int DHT_Pin = 2;

float temperature;
float humidity;
```

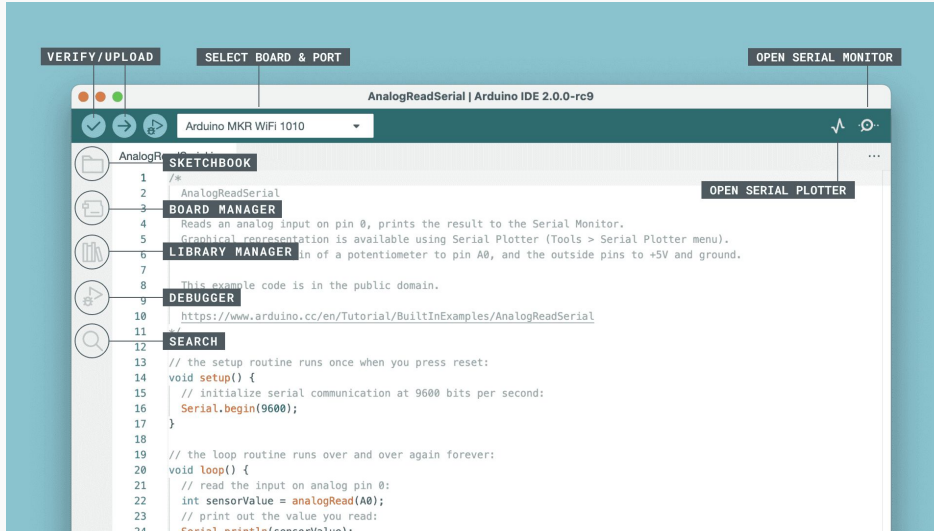


WOKWI

Source: wokwi



# Pengenalan Arduino IDE

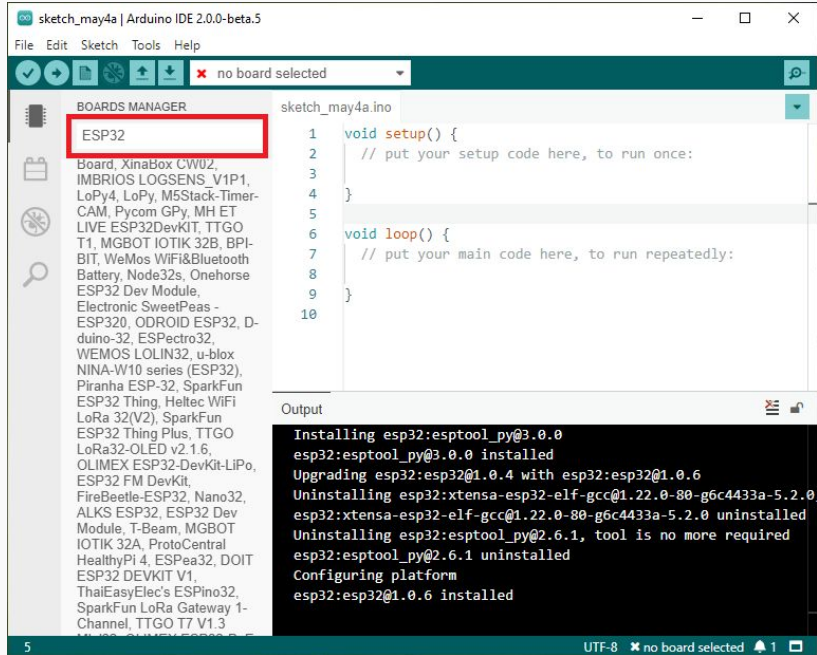


Sumber: Arduino

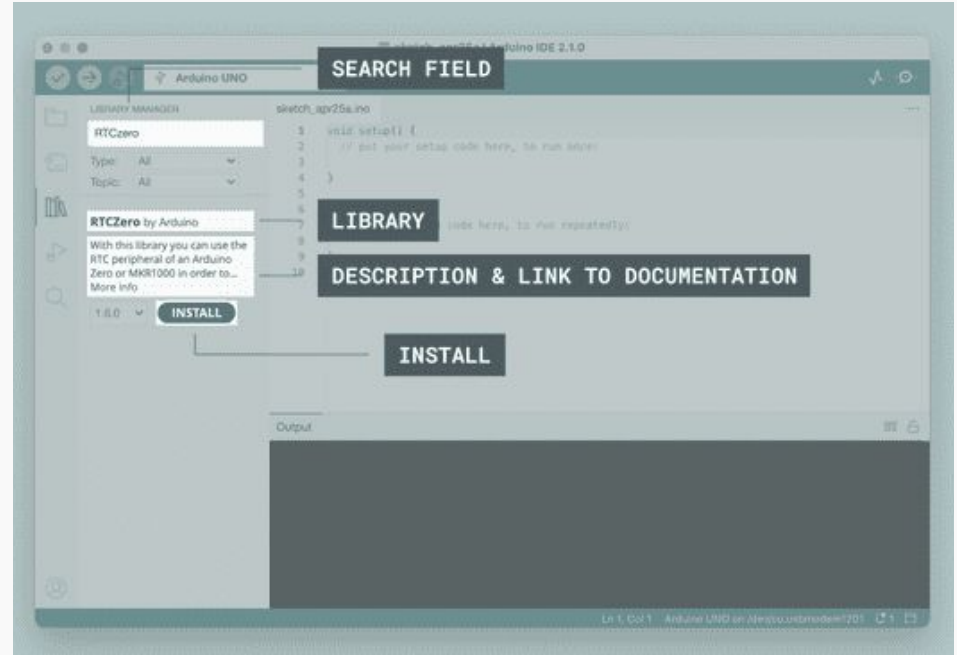
Arduino Integrated Development Environment, atau yang dikenal sebagai Arduino IDE, menyediakan semua dukungan perangkat lunak yang diperlukan untuk menyelesaikan proyek Arduino.

Namun selain untuk Arduino, software ini juga dapat kita gunakan untuk melakukan upload program ke ESP32 dengan extension yang diberikan oleh Arduino IDE.

# Pengenalan Arduino IDE

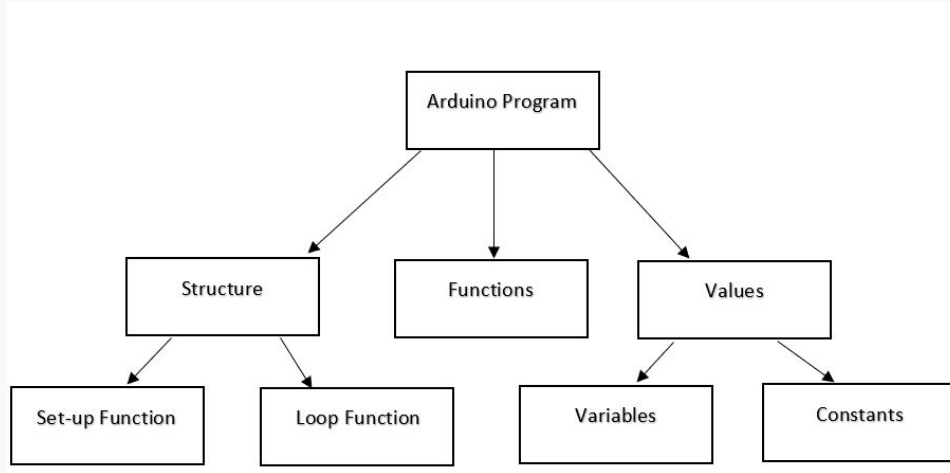


Install dan Setting ESP32



Install Library

# Pengenalan Arduino IDE



- Struktur Program Arduino IDE

```
int pin1 = 5;

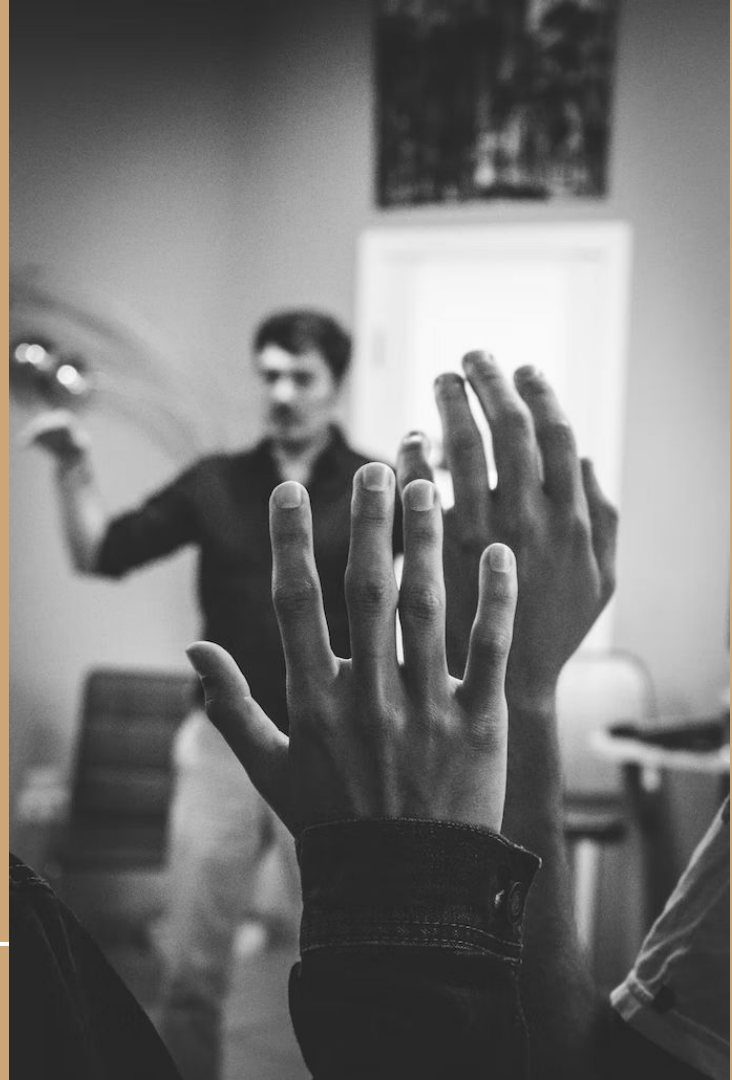
void setup() {
  Serial.begin(9600);
  pinMode(pin1, INPUT);
}

void loop() {
  if (digitalRead(pin1) == HIGH) {
    Serial.write('high');
  }
  else {
    Serial.write('low');
  }

  delay(1000);
}
```



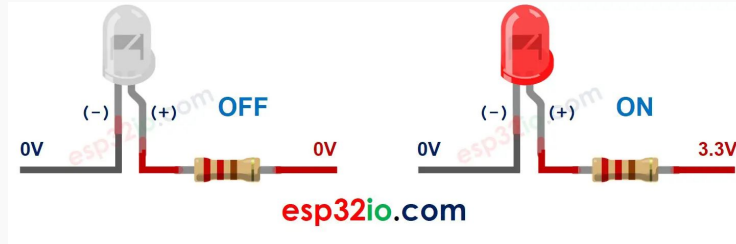
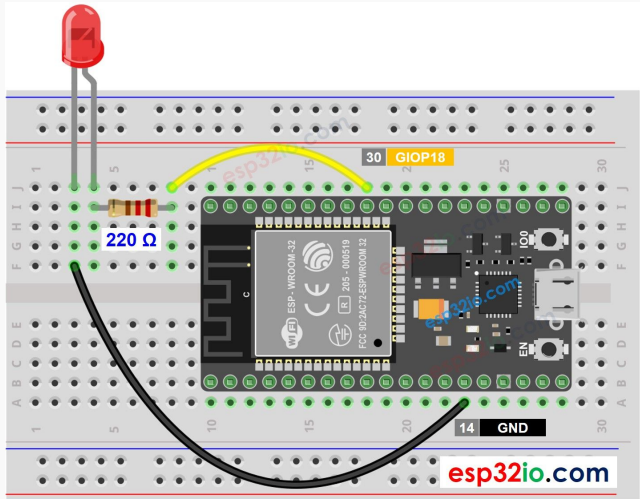
# Challenges





# Challenge!

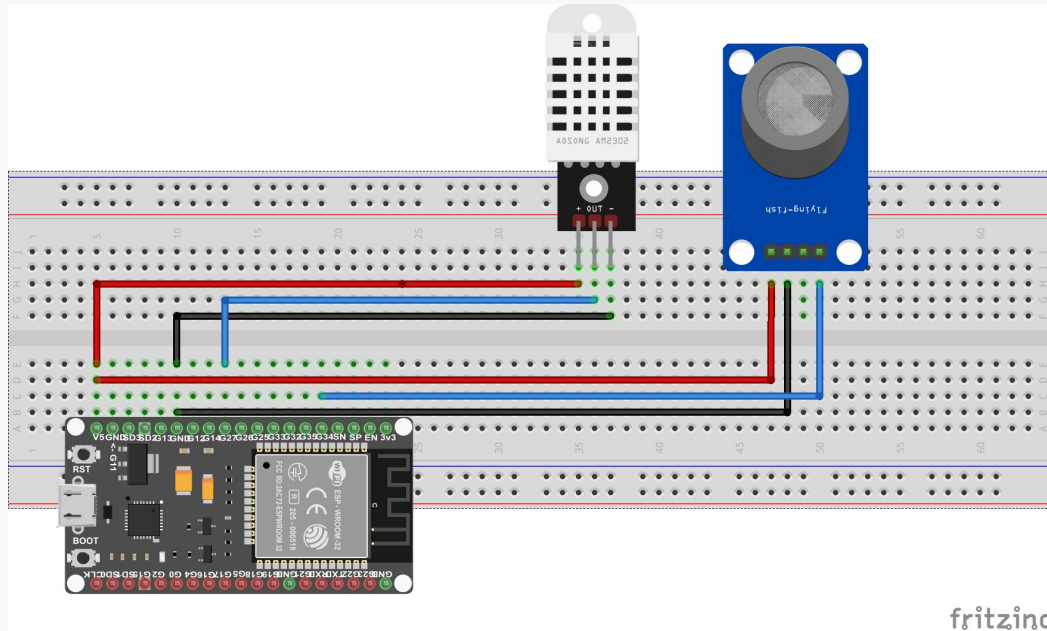
Buatlah simple file untuk melakukan blink LED dengan schematic berikut



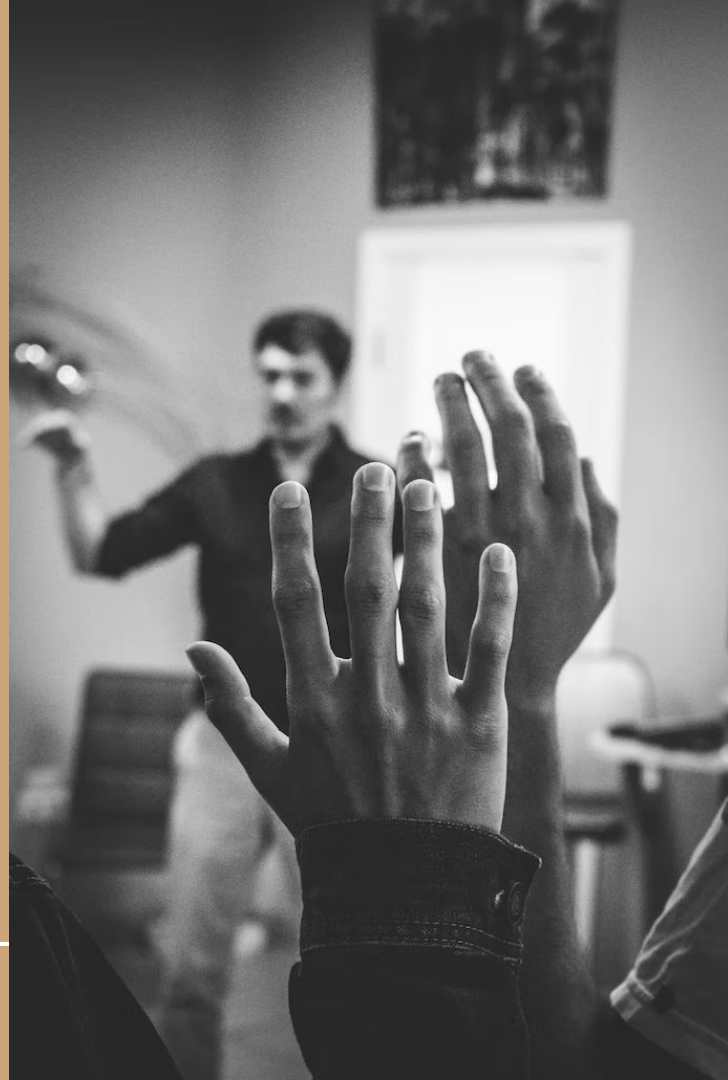
Sumber: <https://esp32io.com/tutorials/esp32-led-blink>

# Challenge!

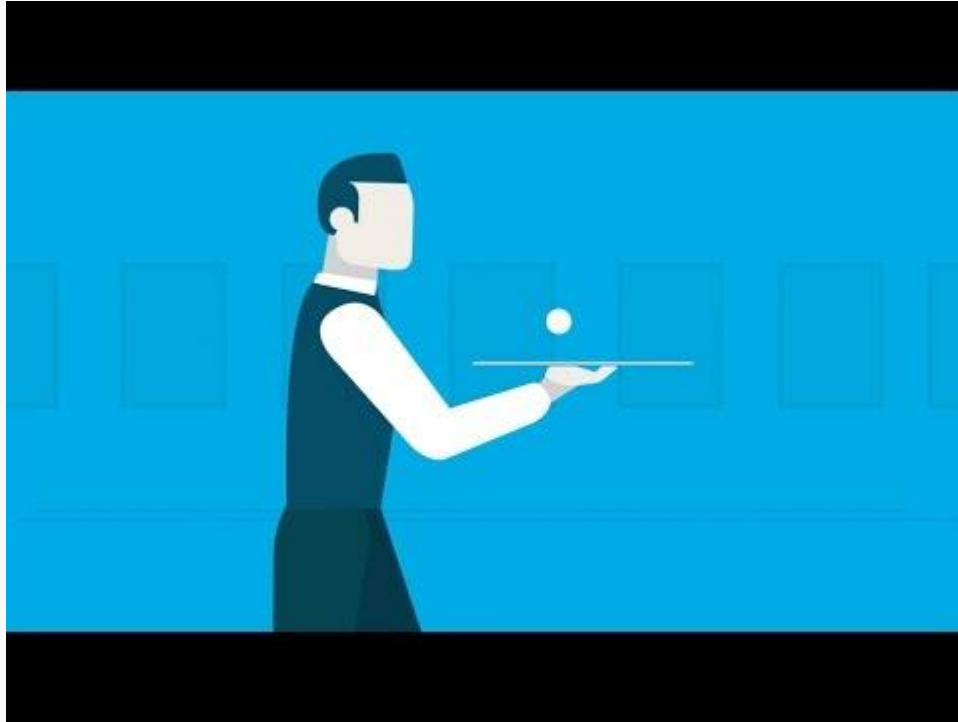
- Bacalah data dari sensor yang kalian miliki dan tampilkan pada Serial Monitor
- Kirimkan datanya ke MQTT Broker dengan topic /sensor/data



Q&A



# Intro ( What is API )



# Intro Flask

Flask is a widely adopted Python framework for building web applications. It allows Python developers to use their preferred language with all of its assets while building scalable and fast-to-start Python web applications.



# Intro Flask

- Flask installation and getting started

## Install Flask

Within the activated environment, use the following command to install Flask:

```
$ pip install Flask
```

<https://flask.palletsprojects.com/en/2.1.x/installation/>

# Intro Flask

- Flask minimum application

```
#!/usr/bin/python
# coding: utf-8
from flask import Flask
```

← Import flask library

```
app = Flask(__name__)
```

← Initiate Flask Instances

```
@app.route('/')
def entry_point():
    return 'Hello World!'
```

← Flask Routing

```
if __name__ == '__main__':
    app.run(debug=True)
```

← Run Flask app



# Flask Routing

- Flask Routing
  - Dengan fungsi `route()` kita dapat membuat beberapa endpoint untuk aplikasi Flask kita. Sehingga kita menentukan sebuah URL akan menuju pada fungsi yang mana

Use the **`route()`** decorator to bind a function to a URL.

```
@app.route('/')
def index():
    return 'Index Page'

@app.route('/hello')
def hello():
    return 'Hello, World'
```

# GET & POST API Flask

- Flask HTTP Method
  - Flask HTTP Method membuat kita bisa menentukan metode HTTP apa yang diperbolehkan dalam routes dan apa yang kita ingin lakukan dengan metode tersebut

## HTTP Methods

Web applications use different HTTP methods when accessing URLs. You should familiarize yourself with the HTTP methods as you work with Flask. By default, a route only answers to `GET` requests. You can use the `methods` argument of the `route()` decorator to handle different HTTP methods.

```
from flask import request

@app.route('/login', methods=['GET', 'POST'])
def login():
    if request.method == 'POST':
        return do_the_login()
    else:
        return show_the_login_form()
```

# GET & POST API Flask

- Flask Processing Data
  - Dengan library requests bawaan dari Flask kita dapat memproses data yang dikirimkan melalui API

```
from flask import request
```

Dan untuk mengakses data tersebut kita dapat menggunakan *syntax*:

```
body = request.get_json() # mendapatkan request body dalam JSON  
params = request.args.get('params') # mendapatkan requests parameters dengan key params  
form = request.form.get('form') # mendapatkan requests form data dengan nama form
```

# GET & POST API Flask

- Flask JSON Response
  - Flask menyediakan interface yang mudah untuk kita mengembalikan response JSON, dapat menggunakan function jsonify() atau cukup mengembalikannya dalam bentuk dictionary

## APIs with JSON

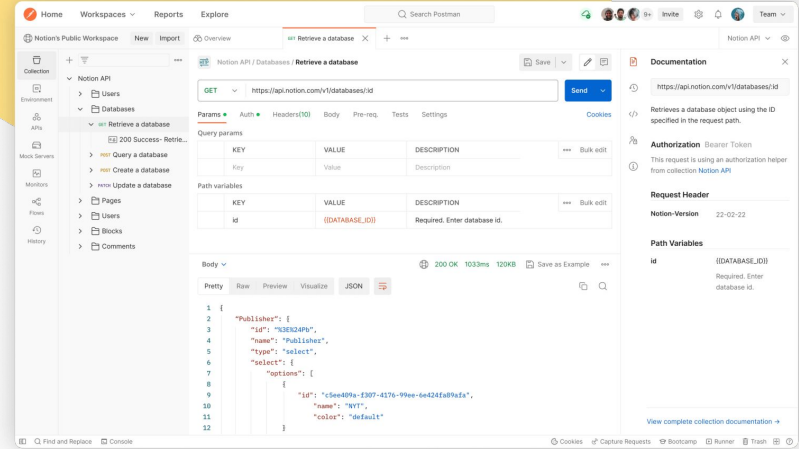
A common response format when writing an API is JSON. It's easy to get started writing such an API with Flask. If you return a `dict` from a view, it will be converted to a JSON response.

```
@app.route("/me")
def me_api():
    user = get_current_user()
    return {
        "username": user.username,
        "theme": user.theme,
        "image": url_for("user_image", filename=user.image),
    }
```

# POSTMAN

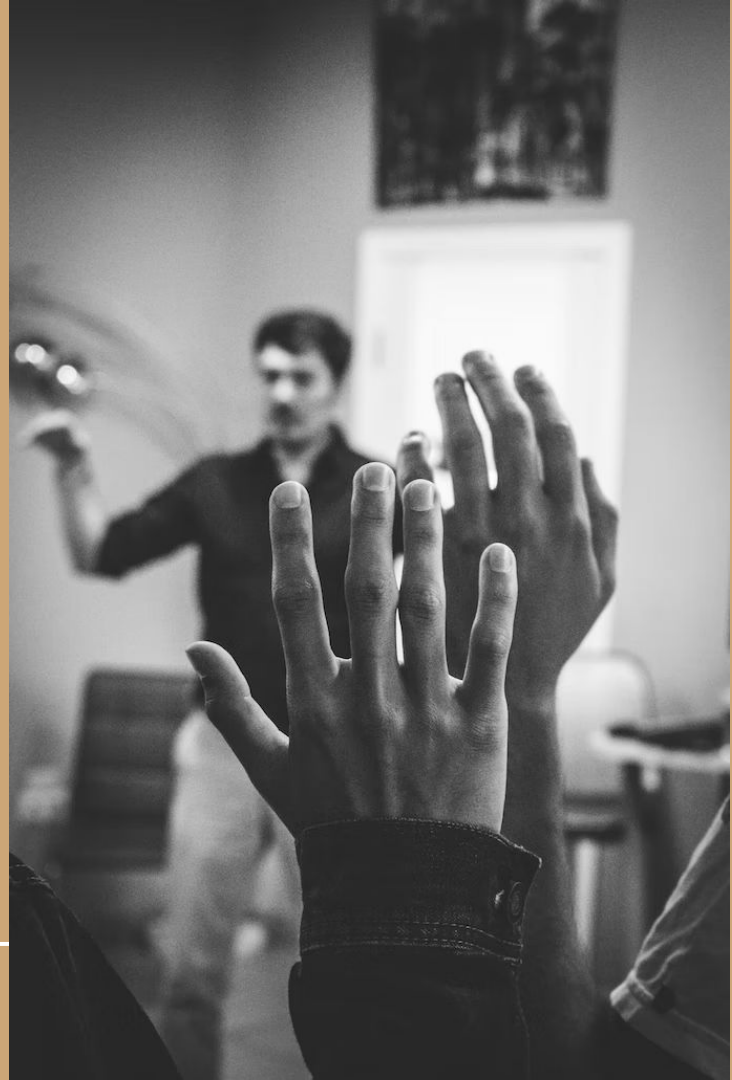
Postman is an API(application programming interface) development tool that helps to build, test and modify APIs.

It has the ability to make various types of HTTP requests(GET, POST, PUT, PATCH), save environments for later use, converting the API to code for various languages(like JavaScript, and Python).



Download here:  
[https://www.postman.com/downloads/?utm\\_source=postman-home](https://www.postman.com/downloads/?utm_source=postman-home)

# Challenges



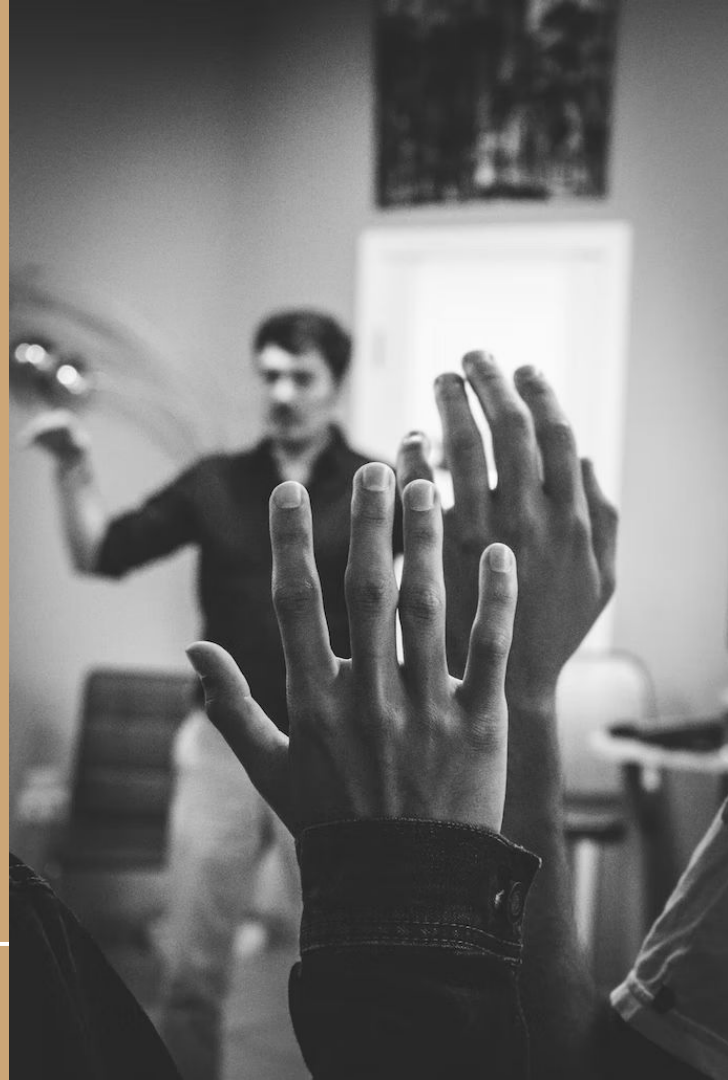
# Challenge!

Buatlah sebuah aplikasi Flask yang terdiri dari beberapa kondisi berikut

1. POST API dengan
  - a. route /sensor/data
  - b. 2 buah data ( buat dummy i.e temperature, kelembapan ) dan timestamp
  - c. Simpan pada temporary list dan response message sukses
2. GET API dengan
  - a. route /sensor/data
  - b. Response dengan seluruh data dalam temporary list



Q&A



# Tech Assignment

Buatlah sebuah rangkaian pada ESP32 menggunakan ESP32 kit yang terdiri dari beberapa kondisi berikut

1. .ino files untuk mengambil 1 buah data sensor dan mengirimkannya ke local server melalui HTTP REST API ( POST Method )
2. .py files untuk menerima data REST API dan mengembalikan response yang sesuai
3. Foto fisik dari rangkaian ESP32