**sketch.ino:**

```
#include <Wire.h>
#include <WiFi.h>
#include <DHT.h>
#include <PubSubClient.h>
#include <Arduino.h>
#include <Adafruit_MPU6050.h>
#include <Adafruit_Sensor.h>

#define DHT_PIN 12
#define PULSE_PIN 35
#define MQTT_SERVER "broker.emqx.io"
#define MQTT_PORT 1883
const char *ssid = "Wokwi-GUEST";
const char *password = "";
#define MQTT_TOPIC_HR "/heartRate"
#define MQTT_TOPIC_TEMP "/tempValue"
#define MQTT_TOPIC_HUM "/humValue"
#define MQTT_TOPIC_ACCEL "/accelData"
#define MQTT_TOPIC_GYRO "/gyroData"
WiFiClient espClient;
PubSubClient client(espClient);
DHT dht(DHT_PIN, DHT22);
Adafruit_MPU6050 mpu;
void setup() {
  Wire.begin(23, 22); // SDA on GPIO 23, SCL on GPIO 22
  Serial.begin(115200);
  Serial.println("Hello, ESP32!");
  connectToWiFi();

  client.setServer(MQTT_SERVER, MQTT_PORT);
```

```
  dht.begin();

  // Initialize MPU6050
  if (!mpu.begin()) {
    Serial.println("MPU6050 not connected!");
    while (1);
  }
  Serial.println("MPU6050 connected!");
}

void loop() {
  if (!client.connected()) {
    reconnect();
  }

  float temperature = dht.readTemperature();
  float humidity = dht.readHumidity();

  // Read pulseValue from PULSE_PIN
  int16_t pulseValue = analogRead(PULSE_PIN);

  // Convert pulseValue to voltage
  float voltage = pulseValue * (3.3 / 4095.0);

  // Calculate heartRate from voltage (simplified example, adjust as needed)
  int heartRate = (voltage / 3.3) * 675;

  // Print HeartRate
  Serial.print("Heart Rate: ");
  Serial.print(heartRate);
```

```
  Serial.print(" Temp: ");
  Serial.print(temperature);
  Serial.print(" Humidity: ");
  Serial.println(humidity);

  // Read MPU6050 data
  sensors_event_t accel, gyro, temp;
  mpu.getEvent(&accel, &gyro, &temp);

  Serial.print("Accel: ");
  Serial.print(accel.acceleration.x); Serial.print(", ");
  Serial.print(accel.acceleration.y); Serial.print(", ");
  Serial.println(accel.acceleration.z);

  Serial.print("Gyro: ");
  Serial.print(gyro.gyro.x); Serial.print(", ");
  Serial.print(gyro.gyro.y); Serial.print(", ");
  Serial.println(gyro.gyro.z);

  client.publish(MQTT_TOPIC_HR, String(heartRate).c_str());
  client.publish(MQTT_TOPIC_TEMP, String(temperature).c_str());
  client.publish(MQTT_TOPIC_HUM, String(humidity).c_str());

  String accelData = String(accel.acceleration.x) + "," + String(accel.acceleration.y) + "," +
String(accel.acceleration.z);
  String gyroData = String(gyro.gyro.x) + "," + String(gyro.gyro.y) + "," + String(gyro.gyro.z);

  client.publish(MQTT_TOPIC_ACCEL, accelData.c_str());
  client.publish(MQTT_TOPIC_GYRO, gyroData.c_str());

  delay(1000); // Increased delay to 1 second for more stability
```

```
}

void connectToWiFi() {
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(1000);
    Serial.println("Connecting to WiFi...");
  }
  Serial.println("Connected to WiFi");
}

void reconnect() {
  while (!client.connected()) {
    if (client.connect("esp32_neopixel_controller")) {
      Serial.println("Connected to MQTT");
      // subscribeToCommands();
    } else {
      Serial.print("Failed, rc=");
      Serial.print(client.state());
      Serial.println(" Retrying in 5 seconds...");
      delay(5000);
    }
  }
}
```

## diagram.json:

```
{
  "version": 1,
  "author": "ProCoding Whitehat JR",
  "editor": "wokwi",
  "parts": [
```

```
  { "type": "board-esp32-devkit-c-v4", "id": "esp", "top": 0, "left": 0, "attrs": {} },
  { "type": "chip-pulsesensor", "id": "chip1", "top": 154.62, "left": 148.8, "attrs": {} },
  {
    "type": "wokwi-dht22",
    "id": "dht1",
    "top": -18.9,
    "left": -149.4,
    "attrs": { "temperature": "68.7", "humidity": "36.5" }
  },
  { "type": "wokwi-mpu6050", "id": "imu1", "top": 51.82, "left": 155.92, "attrs": {} }
 ],
 "connections": [
  [ "esp:TX", "$serialMonitor:RX", "", [] ],
  [ "esp:RX", "$serialMonitor:TX", "", [] ],
  [ "chip1:GND", "esp:GND.1", "black", [ "h-28.8", "v67.2", "h-134.4", "v-38.4" ] ],
  [ "dht1:VCC", "esp:5V", "red", [ "v0" ] ],
  [ "esp:5V", "chip1:VCC", "red", [ "h-33.41", "v38.4", "h163.2", "v-67.2" ] ],
  [ "esp:GND.1", "dht1:GND", "black", [ "h0" ] ],
  [ "dht1:SDA", "esp:12", "green", [ "v0" ] ],
  [ "chip1:OUT0", "esp:35", "green", [ "h-38.4", "v-211.2", "h-134.4", "v115.2" ] ],
  [ "imu1:GND", "esp:GND.1", "black", [ "v0" ] ],
  [ "imu1:VCC", "esp:3V3", "red", [ "v0" ] ],
  [ "imu1:SCL", "esp:22", "green", [ "v0" ] ],
  [ "imu1:SDA", "esp:23", "green", [ "v0" ] ]
 ],
 "dependencies": {}
}
```

## Pulsesensor.chip.json:

```
{
 "name": "pulse-sensor",
 "author": "ProCoding Whitehat JR",
```

```
  "pins": [
    "GND",
    "VCC",
    "",
    "OUT0",
    "",
    "",
    ""
  ],
  "controls": [
    {
      "id": "pulseValue",
      "label": "Heart Rate",
      "type": "slider",
      "min":0,
      "max":675
    }
  ]
}
```

## Pulsesensor.chip.c:

```
//Pulse sensor reference: https://www.eecs.yorku.ca/~jr/res/m/MD/PulseSensor.pdf
  //https://www.elprocus.com/pulse-sensor-working-principle-and-its-applications/
#include "wokwi-api.h"
#include <stdio.h>
#include <stdlib.h>

typedef struct {
  pin_t pin;
  int pulseValue;
} chip_data_t;
```

```
void chip_timer_callback(void *data) {
  chip_data_t *chip_data = (chip_data_t*)data;

  // Read pulseValue in pulseValue variable
  int pulseValue = attr_read(chip_data->pulseValue);
  // Calculate volts
  float volts = pulseValue *3.3/ 675;
  // Send volts on the pin
  printf("%f \n", volts);
  pin_dac_write(chip_data->pin, volts);

}

void chip_init() {
  printf("Hello from custom chip!\n");
  chip_data_t *chip_data = (chip_data_t*)malloc(sizeof(chip_data_t));

  // Initialize pulseValue
  chip_data->pulseValue = attr_init("pulseValue", 0);
  // Initialize pin
  chip_data->pin = pin_init("OUT0", ANALOG);



  const timer_config_t config = {
    .callback = chip_timer_callback,
    .user_data = chip_data,
  };
  timer_t timer_id = timer_init(&config);
  timer_start(timer_id, 1000, true);
}
```

## Libraries.c:

# Wokwi Library List

# See https://docs.wokwi.com/guides/libraries

# Automatically added based on includes:

DHT sensor library

PubSubClient

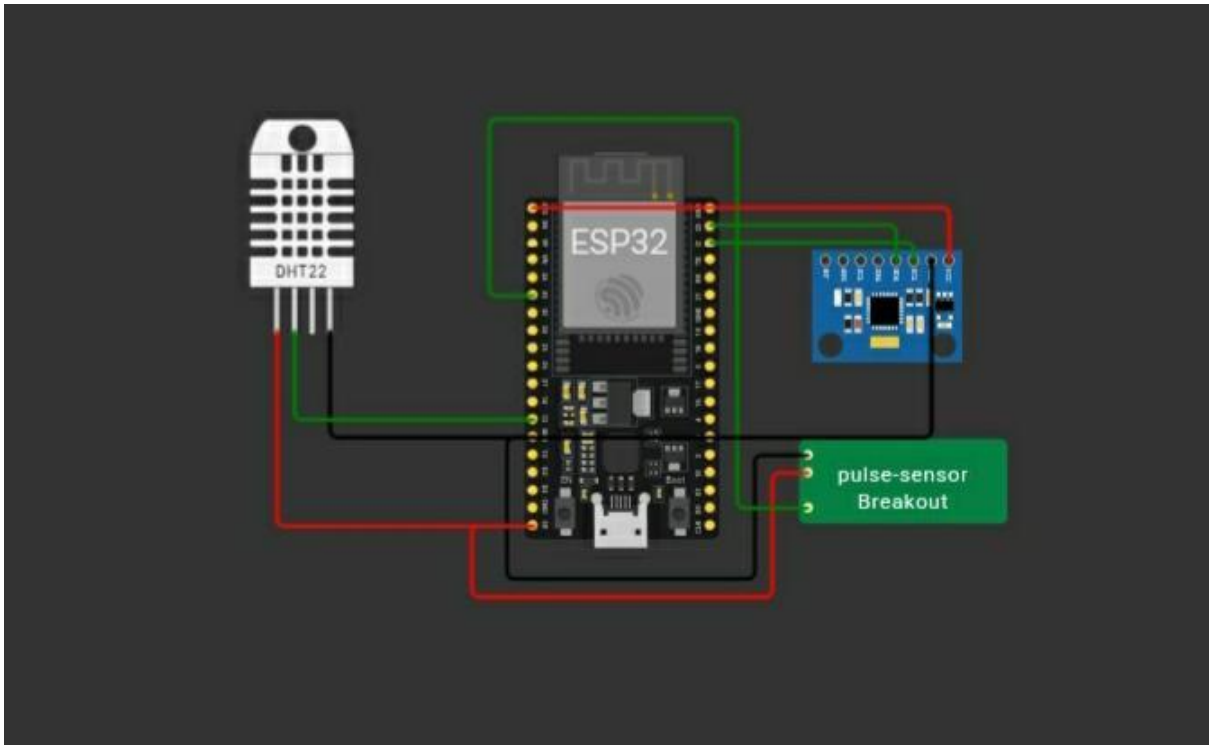
DHT sensor library for ESPx

Adafruit MPU6050

## Library Manager:

- DHT sensor library
- PubSubClient
- DHT sensor

## Simulation:




## OUTPUT :

Hello, ESP32!

Connecting to WiFi...

Connecting to WiFi...

Connected to WiFi

MPU6050 connected!

Connected to MQTT

Heart Rate: 0 Temp: 68.70 Humidity: 36.50

Accel: 0.00, 0.00, 9.81

Gyro: 0.00, 0.00, 0.00