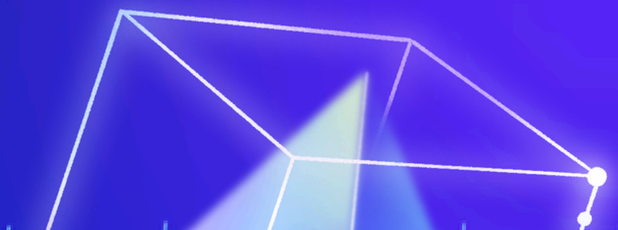
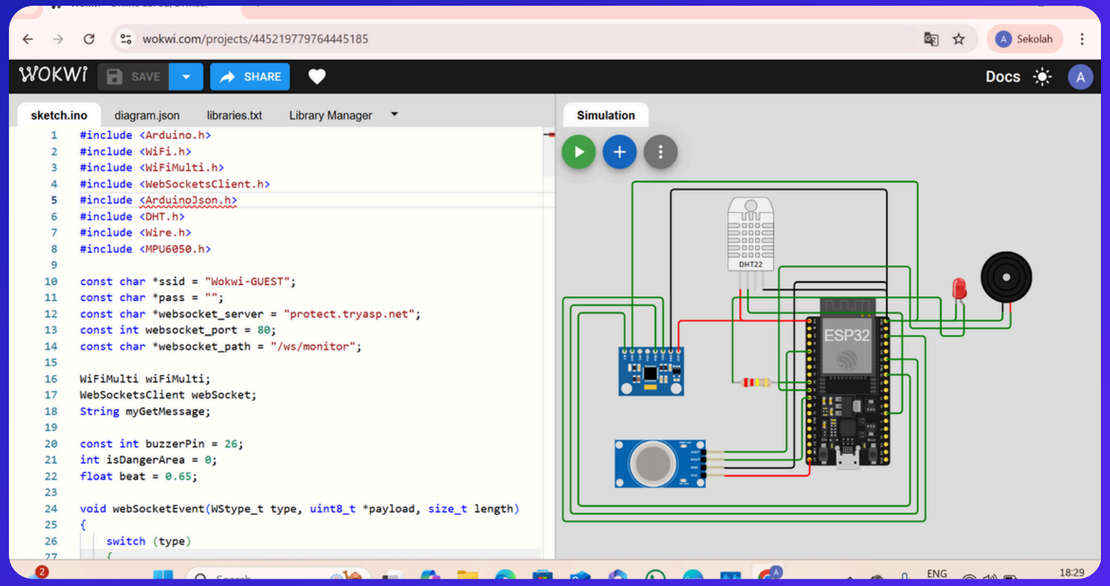
# HELM IOT UNTUK PEMANTAUAN KEAMANAN PEKERJA TAMBANG BATU BARA SECARA REALTIME BERBASIS DASHBOARD WEBSITE

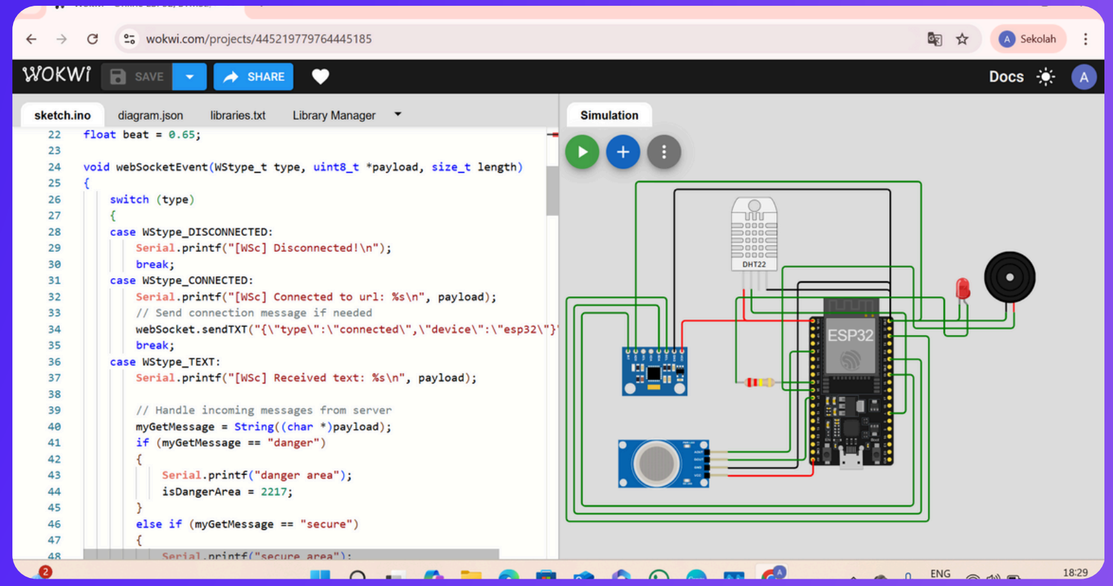
# KELOMPOK

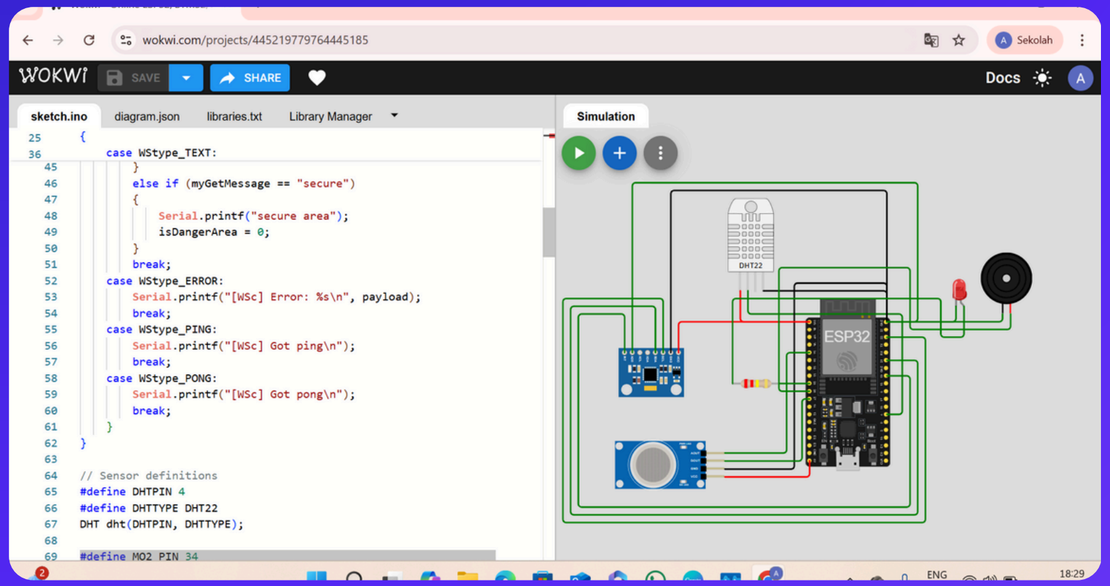
1. Afrizal Rizky (231110008) Judul Lampu Lalu Lintas
2. Mochammad Lutfi Bintoro Syaiful (231110001) judul asn mengubah suhu menjadi kode morse
3. Andreas Kiew(231110030) judul Helm IoT untuk Pemantauan Keamanan Pekerja Tambang Batu Bara secara realtime berbasis dashboard website

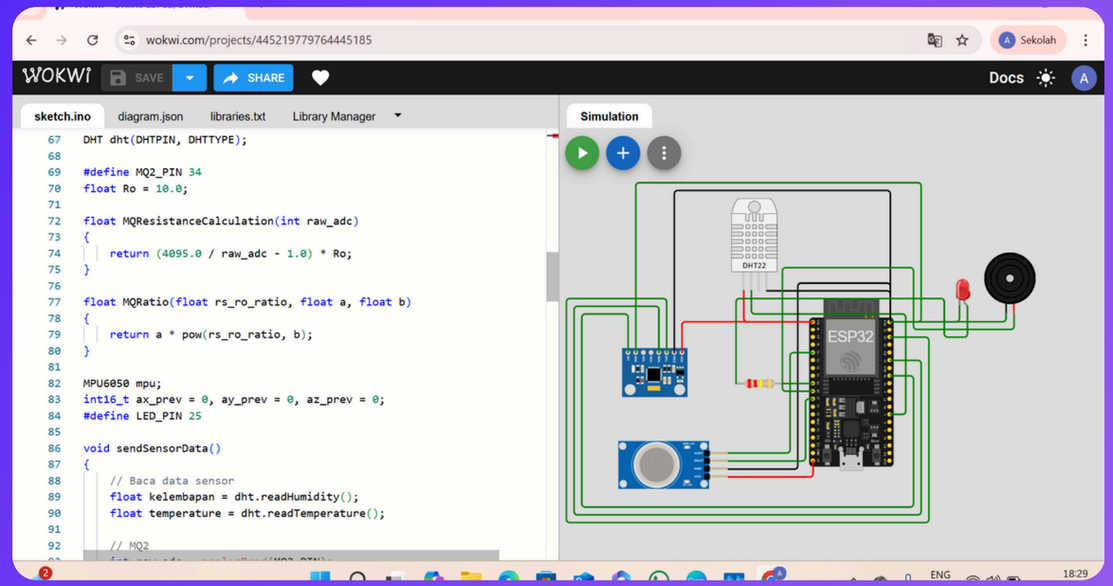


# PROGRAM NYA

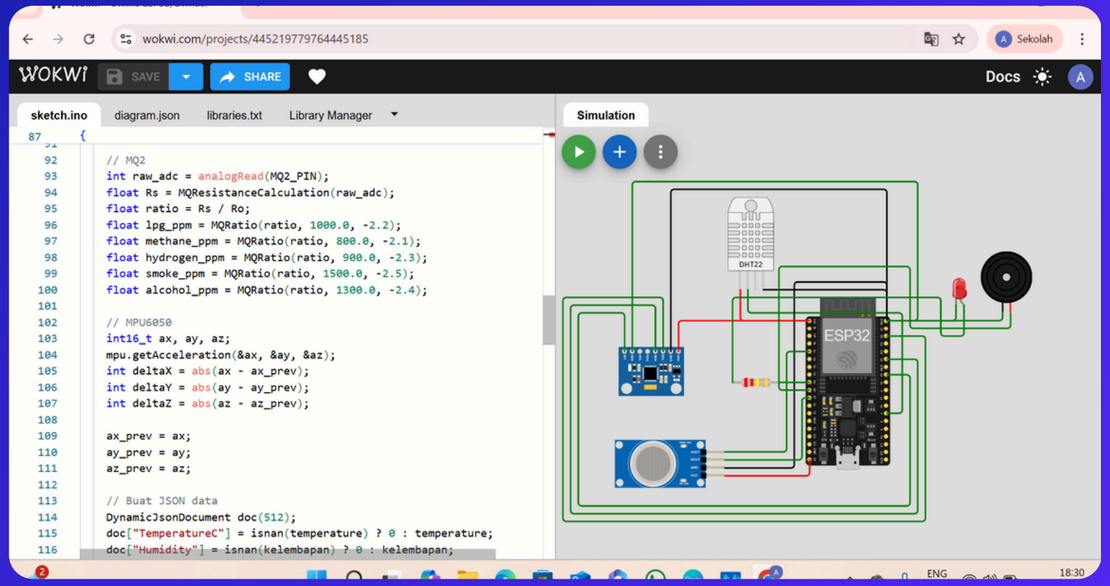

wokwi.com/projects/445219779764445185
WOKWI
SAVE
SHARE
Docs
sketch.ino
diagram.json
libraries.txt
Library Manager
Simulation
#include <Arduino.h>
#include <WiFi.h>
#include <WiFiMulti.h>
#include <WebSocketsClient.h>
#include <ArduinoJson.h>
#include <DHT.h>
#include <Wire.h>
#include <MPU6050.h>
const char *ssid = "Wokwi-GUEST";
const char *pass = "";
const char *websocket_server = "protect.tryasp.net";
const int websocket_port = 80;
const char *websocket_path = "/ws/monitor";
WiFiMulti wiFiMulti;
WebSocketsClient webSocket;
String myGetMessage;
const int buzzerPin = 26;
int isDangerArea = 0;
float beat = 0.65;
void webSocketEvent(WStype_t type, uint8_t *payload, size_t length)
{
switch (type)
DHT22
ESP32

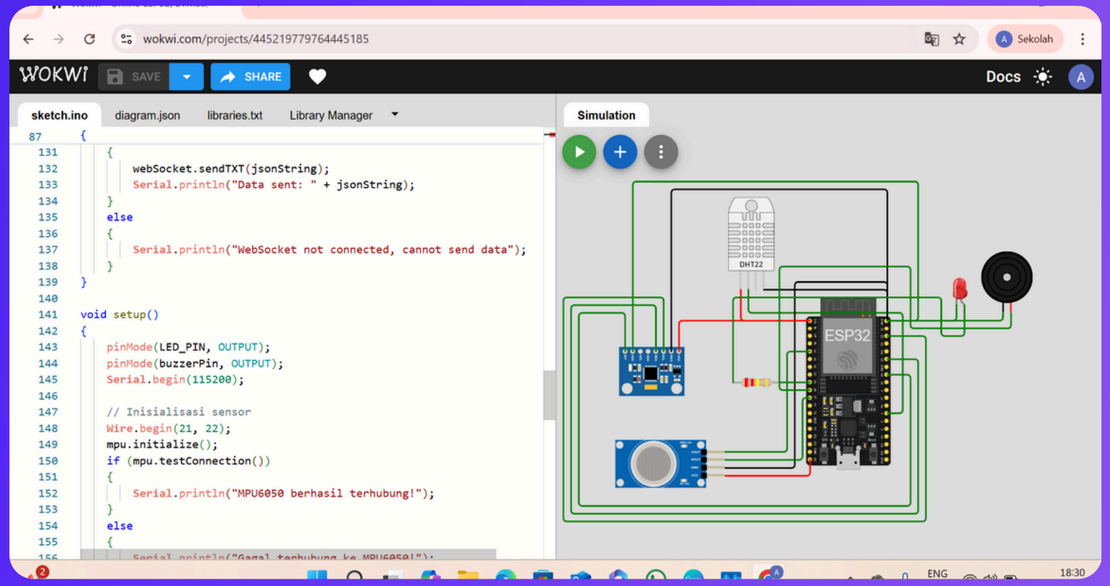

wokwi.com/projects/445219779764445185
WOKWI
SAVE
SHARE
Docs
sketch.ino
diagram.json
libraries.txt
Library Manager
Simulation
float beat = 0.65;
void webSocketEvent(WStype_t type, uint8_t *payload, size_t length)
{
switch (type)
{
case WStype_DISCONNECTED:
Serial.printf("[WSc] Disconnected!\n");
break;
case WStype_CONNECTED:
Serial.printf("[WSc] Connected to url: %s\n", payload);
// Send connection message if needed
webSocket.sendTXT("{\"type\":\"connected\",\"device\":\"esp32\"}
break;
case WStype_TEXT:
Serial.printf("[WSc] Received text: %s\n", payload);
// Handle incoming messages from server
myGetMessage = String((char *)payload);
if (myGetMessage == "danger")
{
Serial.printf("danger area");
isDangerArea = 2217;
}
else if (myGetMessage == "secure")
DHT22
ESP32

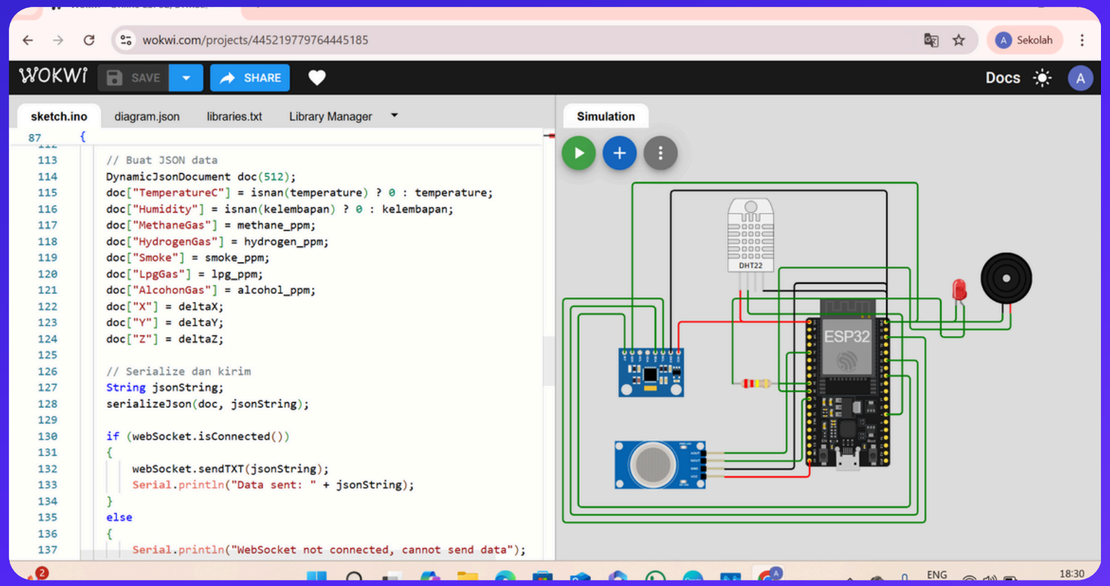

wokwi.com/projects/445219779764445185
WOKWI
SAVE
SHARE
Docs
sketch.ino
diagram.json
libraries.txt
Library Manager
Simulation
{
case WStype_TEXT:
}
else if (myGetMessage == "secure")
{
Serial.printf("secure area");
isDangerArea = 0;
}
break;
case WStype_ERROR:
Serial.printf("[WSc] Error: %s\n", payload);
break;
case WStype_PING:
Serial.printf("[WSc] Got ping\n");
break;
case WStype_PONG:
Serial.printf("[WSc] Got pong\n");
break;
}
// Sensor definitions
#define DHTPIN 4
#define DHTTYPE DHT22
DHT dht(DHTPIN, DHTTYPE);
#define MQ2_PIN 34
DHT22
ESP32

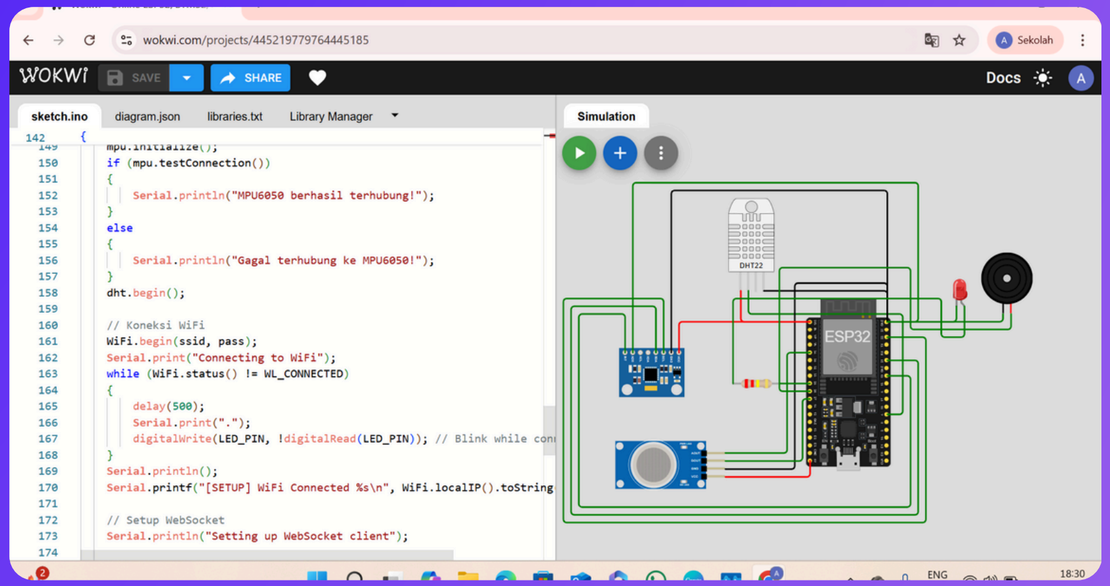

wokwi.com/projects/445219779764445185
WOKWI
SAVE
SHARE
Docs
sketch.ino
diagram.json
libraries.txt
Library Manager
Simulation
DHT dht(DHTPIN, DHTTYPE);
#define MQ2_PIN 34
float Ro = 10.0;
float MQResistanceCalculation(int raw_adc)
{
return (4095.0 / raw_adc - 1.0) * Ro;
}
float MQRatio(float rs_ro_ratio, float a, float b)
{
return a * pow(rs_ro_ratio, b);
}
MPU6050 mpu;
int16_t ax_prev = 0, ay_prev = 0, az_prev = 0;
#define LED_PIN 25
void sendSensorData()
{
// Baca data sensor
float kelembapan = dht.readHumidity();
float temperature = dht.readTemperature();
// MQ2
DHT22
ESP32

# PROGRAM NYA

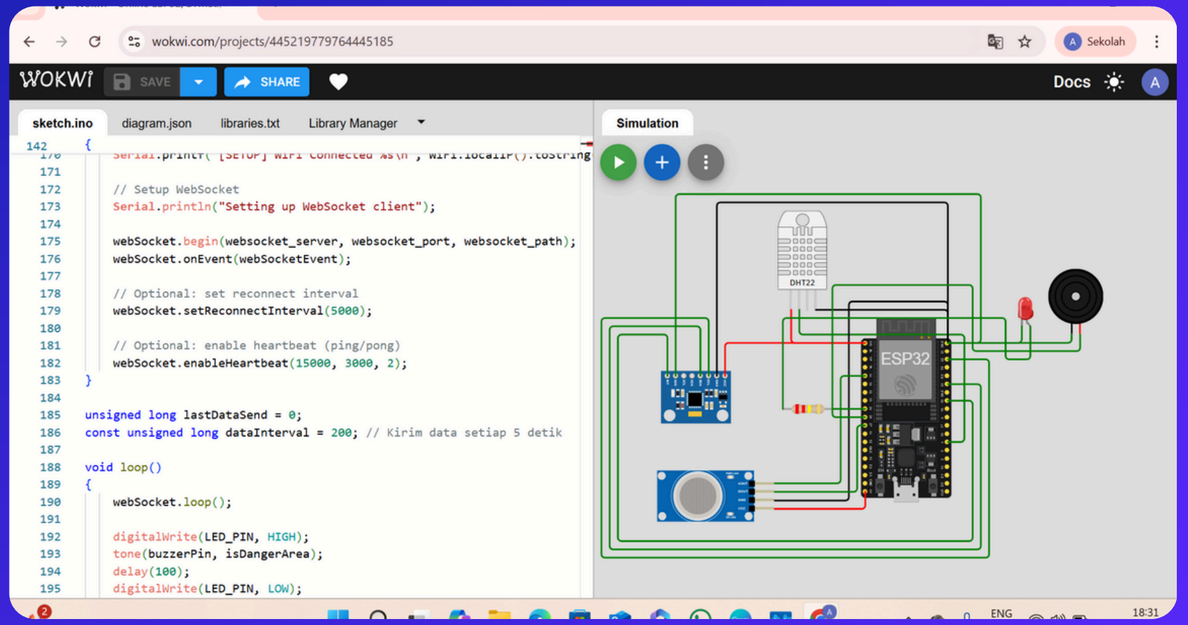

wokwi.com/projects/445219779764445185
WOKWI
SAVE
SHARE
Docs
sketch.ino
diagram.json
libraries.txt
Library Manager
Simulation
87 {
92 // MQ2
93 int raw_adc = analogRead(MQ2_PIN);
94 float Rs = MQResistanceCalculation(raw_adc);
95 float ratio = Rs / Ro;
96 float lpg_ppm = MQRatio(ratio, 1000.0, -2.2);
97 float methane_ppm = MQRatio(ratio, 800.0, -2.1);
98 float hydrogen_ppm = MQRatio(ratio, 900.0, -2.3);
99 float smoke_ppm = MQRatio(ratio, 1500.0, -2.5);
100 float alcohol_ppm = MQRatio(ratio, 1300.0, -2.4);
101
102 // MPU6050
103 int16_t ax, ay, az;
104 mpu.getAcceleration(&ax, &ay, &az);
105 int deltaX = abs(ax - ax_prev);
106 int deltaY = abs(ay - ay_prev);
107 int deltaZ = abs(az - az_prev);
108
109 ax_prev = ax;
110 ay_prev = ay;
111 az_prev = az;
112
113 // Buat JSON data
114 DynamicJsonDocument doc(512);
115 doc["TemperatureC"] = isnan(temperature) ? 0 : temperature;
116 doc["Humidity"] = isnan(kelembapan) ? 0 : kelembapan;
DHT22
ESP32
ENG
18:30

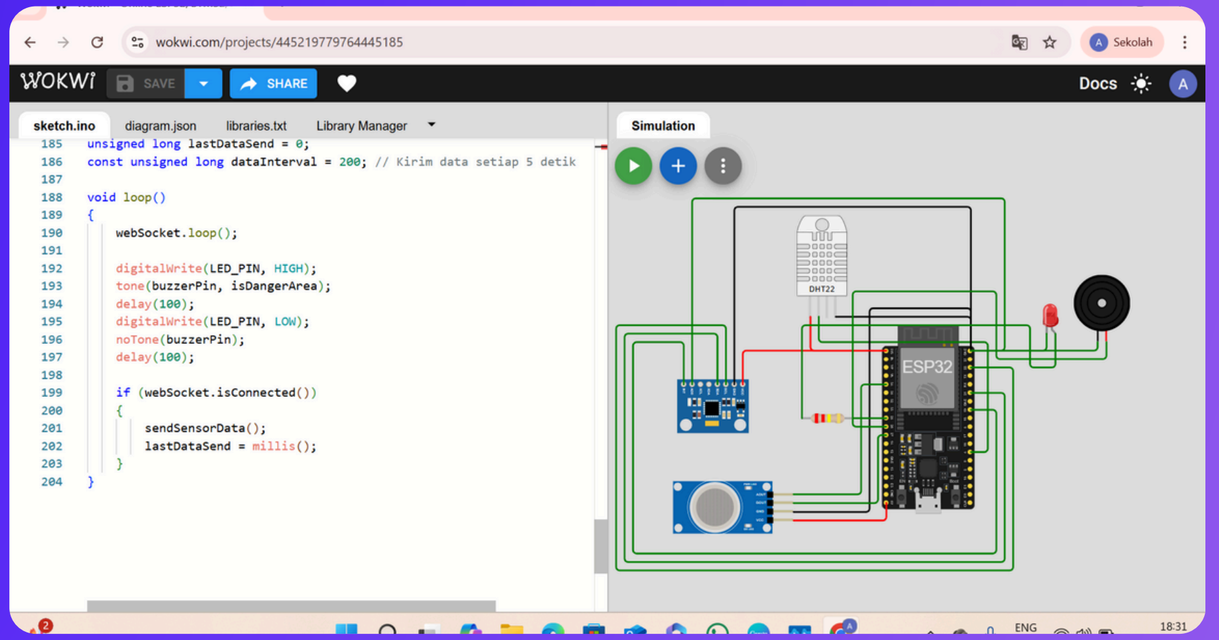

wokwi.com/projects/445219779764445185
WOKWI
SAVE
SHARE
Docs
sketch.ino
diagram.json
libraries.txt
Library Manager
Simulation
87 {
131 {
132 webSocket.sendTXT(jsonString);
133 Serial.println("Data sent: " + jsonString);
134 }
135 else
136 {
137 Serial.println("WebSocket not connected, cannot send data");
138 }
139 }
140
141 void setup()
142 {
143 pinMode(LED_PIN, OUTPUT);
144 pinMode(buzzerPin, OUTPUT);
145 Serial.begin(115200);
146
147 // Inisialisasi sensor
148 Wire.begin(21, 22);
149 mpu.initialize();
150 if (mpu.testConnection())
151 {
152 Serial.println("MPU6050 berhasil terhubung!");
153 }
154 else
155 {
DHT22
ESP32
ENG
18:30


wokwi.com/projects/445219779764445185
WOKWI
SAVE
SHARE
Docs
sketch.ino
diagram.json
libraries.txt
Library Manager
Simulation
87 {
113 // Buat JSON data
114 DynamicJsonDocument doc(512);
115 doc["TemperatureC"] = isnan(temperature) ? 0 : temperature;
116 doc["Humidity"] = isnan(kelembapan) ? 0 : kelembapan;
117 doc["MethaneGas"] = methane_ppm;
118 doc["HydrogenGas"] = hydrogen_ppm;
119 doc["Smoke"] = smoke_ppm;
120 doc["LpgGas"] = lpg_ppm;
121 doc["AlcohonGas"] = alcohol_ppm;
122 doc["X"] = deltaX;
123 doc["Y"] = deltaY;
124 doc["Z"] = deltaZ;
125
126 // Serialize dan kirim
127 String jsonString;
128 serializeJson(doc, jsonString);
129
130 if (webSocket.isConnected())
131 {
132 webSocket.sendTXT(jsonString);
133 Serial.println("Data sent: " + jsonString);
134 }
135 else
136 {
137 Serial.println("WebSocket not connected, cannot send data");
DHT22
ESP32
ENG
18:30

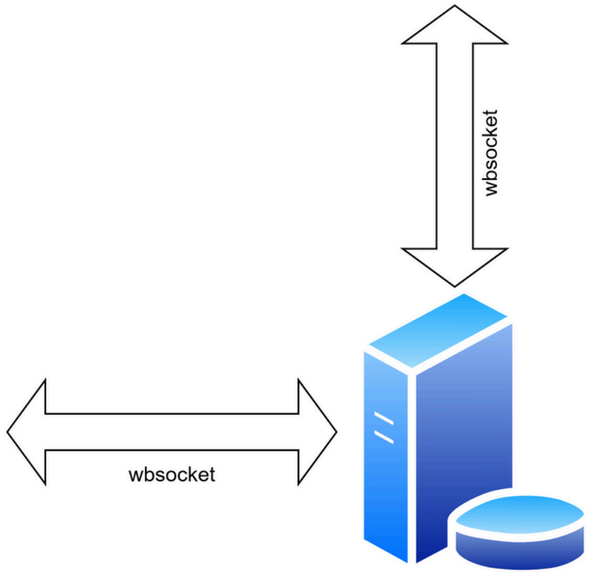

wokwi.com/projects/445219779764445185
WOKWI
SAVE
SHARE
Docs
sketch.ino
diagram.json
libraries.txt
Library Manager
Simulation
142 {
150 if (mpu.testConnection())
151 {
152 Serial.println("MPU6050 berhasil terhubung!");
153 }
154 else
155 {
156 Serial.println("Gagal terhubung ke MPU6050!");
157 }
158 dht.begin();
159
160 // Koneksi WiFi
161 WiFi.begin(ssid, pass);
162 Serial.print("Connecting to WiFi");
163 while (WiFi.status() != WL_CONNECTED)
164 {
165 delay(500);
166 Serial.print(".");
167 digitalWrite(LED_PIN, !digitalRead(LED_PIN)); // Blink while con
168 }
169 Serial.println();
170 Serial.printf("[SETUP] WiFi Connected %s\n", WiFi.localIP().toString
171
172 // Setup WebSocket
173 Serial.println("Setting up WebSocket client");
174
DHT22
ESP32
ENG
18:30

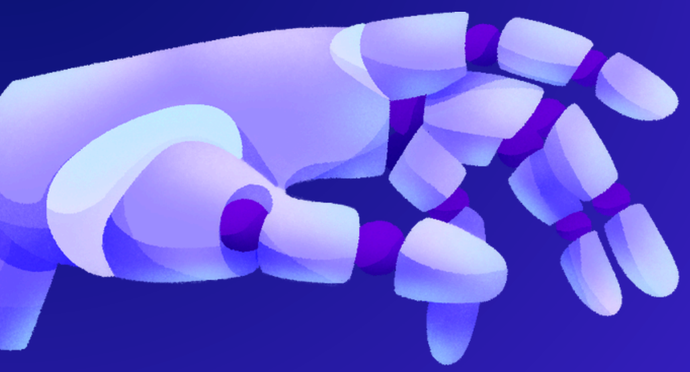
# PROGRAM NYA


wokwi.com/projects/445219779764445185
WOKWI
SAVE
SHARE
Docs
sketch.ino
diagram.json
libraries.txt
Library Manager
Simulation
DHT22
ESP32
142 {
171
172 // Setup WebSocket
173 Serial.println("Setting up WebSocket client");
174
175 webSocket.begin(websocket_server, websocket_port, websocket_path);
176 webSocket.onEvent(webSocketEvent);
177
178 // Optional: set reconnect interval
179 webSocket.setReconnectInterval(5000);
180
181 // Optional: enable heartbeat (ping/pong)
182 webSocket.enableHeartbeat(15000, 3000, 2);
183 }
184
185 unsigned long lastDataSend = 0;
186 const unsigned long dataInterval = 200; // Kirim data setiap 5 detik
187
188 void loop()
189 {
190 webSocket.loop();
191
192 digitalWrite(LED_PIN, HIGH);
193 tone(buzzerPin, isDangerArea);
194 delay(100);
195 digitalWrite(LED_PIN, LOW);
ENG
18:31


wokwi.com/projects/445219779764445185
WOKWI
SAVE
SHARE
Docs
sketch.ino
diagram.json
libraries.txt
Library Manager
Simulation
DHT22
ESP32
185 unsigned long lastDataSend = 0;
186 const unsigned long dataInterval = 200; // Kirim data setiap 5 detik
187
188 void loop()
189 {
190 webSocket.loop();
191
192 digitalWrite(LED_PIN, HIGH);
193 tone(buzzerPin, isDangerArea);
194 delay(100);
195 digitalWrite(LED_PIN, LOW);
196 noTone(buzzerPin);
197 delay(100);
198
199 if (webSocket.isConnected())
200 {
201 sendSensorData();
202 lastDataSend = millis();
203 }
204 }
ENG
18:31

# DIAGRAM NYA





# HALAMAN JUDUL

## Helm IoT b untuk Pemantauan Keamanan Pekerja Tambang Batu Bara Secara Realtime Berbasis Dashboard Website

- Isi Slide:
- Sebuah inovasi helm cerdas yang dirancang untuk mendeteksi kondisi lingkungan berbahaya seperti suhu tinggi, gas beracun, dan pergerakan pekerja. Helm ini terhubung ke sistem dashboard website yang menampilkan data keselamatan secara realtime.
- Tambahan Visual:
- Gambar helm tambang dengan sensor
- Ikon WiFi, suhu, gas, dashboard
- Warna tema: hitam–oranye–abu (nuansa tambang)

# LATAR BELAKANG

Isi Teks:

Pekerjaan di tambang batu bara memiliki tingkat risiko tinggi.

Pekerja sering terpapar gas beracun (metana, karbon monoksida), suhu ekstrem, dan bahaya fisik seperti jatuh atau terjebak.

📉 Sistem pemantauan manual membutuhkan waktu dan sering terlambat dalam memberikan peringatan.

📈 Dibutuhkan alat pemantau otomatis, cepat, dan real-time untuk menjamin keselamatan pekerja.

Contoh Kasus Nyata:

- Ledakan tambang akibat gas metana bocor tanpa terdeteksi.
- Pekerja pingsan karena suhu dan kelembapan ekstrem.

# DESAIN SISTEM HELM IOT

Komponen Utama:

1. ESP32 – mikrokontroler utama (pengolah & pengirim data).
2. DHT22 – sensor suhu dan kelembapan.
3. Sensor Gas MQ – pendeteksi gas metana, CO, LPG, dan asap.
4. MPU6050 – sensor percepatan dan giroskop (gerakan).
5. Buzzer dan LED – indikator bahaya.
6. Dashboard Website – menampilkan data sensor secara realtime.

Fungsi Umum:

Helm mengumpulkan data lingkungan dan kondisi pekerja → ESP32 mengolah data → mengirim ke dashboard website → memberi alarm jika bahaya terdeteksi.

# ESP32 (MIKROKONTROLER UTAMA)

Fungsi:

- Mengontrol semua sensor (DHT22, MQ, MPU6050).
- Mengirimkan data ke server menggunakan WiFi melalui WebSocket.
- Mengaktifkan LED & buzzer berdasarkan kondisi bahaya.

Kegunaan:

Sebagai “otak” helm yang mengatur seluruh sistem pemantauan.

Contoh:

Ketika gas metana mencapai 60.000 ppm → ESP32 secara otomatis menyalakan LED merah, membunyikan buzzer, dan mengirim sinyal peringatan ke dashboard.

# SENSOR DHT22 (SUHU & KELEMBAPAN)

Fungsi:

Mengukur suhu udara (°C) dan kelembapan (%) di sekitar pekerja.

Kegunaan di Tambang:

- Mendeteksi area tambang dengan suhu ekstrem.
- Memantau kondisi kelembapan tinggi yang bisa mempengaruhi pernapasan pekerja.

Contoh Nyata:

Saat suhu mencapai 45°C dan kelembapan 85%, sistem mengirim peringatan ke dashboard agar pekerja beristirahat atau keluar dari area panas.

# SENSOR GAS (MQ SERIES)

Fungsi:

Mendeteksi berbagai jenis gas seperti:

- Metana ($CH_4$)
- Karbon monoksida (CO)
- Asap (Smoke)
- LPG
- Alkohol gas

Kegunaan:

- Menghindari potensi kebakaran dan ledakan akibat kebocoran gas.
- Memastikan area kerja memiliki kadar udara aman.

Contoh Nyata:

Jika kadar metana mencapai 60.000 ppm, LED merah menyala, buzzer aktif, dan dashboard menampilkan status “BAHAYA – GAS TINGGI”.

# SENSOR MPU6050 (GERAKAN & KESEIMBANGAN)

Fungsi:

Mengukur percepatan (X, Y, Z) dan rotasi (giroskop) helm untuk mendeteksi posisi pekerja.

Kegunaan:

- Mendeteksi pergerakan abnormal seperti jatuh atau kehilangan keseimbangan.
- Mengetahui apakah pekerja sedang diam, berjalan, atau terjatuh.

Contoh:

Jika nilai akselerasi < 1 selama 10 detik → dashboard menampilkan “Pekerja tidak bergerak – kemungkinan jatuh.”

# BUZZER & LED (SISTEM PERINGATAN)

- Fungsi:
- Buzzer: Mengeluarkan suara keras sebagai alarm bahaya.
- LED: Menunjukkan kondisi sistem secara visual.
- Kegunaan:
- Memberikan tanda cepat kepada pekerja tanpa perlu melihat layar.
- Warna LED menunjukkan status:
- 🟢 Hijau = Aman
- 🟡 Kuning = Waspada
- 🔴 Merah = Bahaya
- Contoh:
- Suhu tinggi + gas berbahaya = LED merah nyala dan buzzer berbunyi keras.

# DASHBOARD WEBSITE (MONITORING REALTIME)

Fungsi:

- Menampilkan data sensor (suhu, kelembapan, gas, gerakan).
- Memberi notifikasi otomatis jika ada kondisi bahaya.
- Menyimpan histori data dan status pekerja.

Kegunaan:

Memudahkan pengawasan jarak jauh tanpa turun langsung ke area tambang.

# CONTOH DASHBOARD WEBSITE (MONITORING REALTIME)

| Nama Helm | Suhu | Gas (ppm) | Gerakan | Status |
|---|---|---|---|---|
| Helm #1 | 45°C | 62.000 | Diam | ⚠️Bahaya |
| Helm #2 | 50°C | 15.000 | Aktif | ✅Aman |

# ALUR KERJA SISTEM

Langkah-langkah Proses:

1. Helm IoT aktif dan terhubung ke jaringan WiFi.
2. Sensor membaca data suhu, gas, dan gerakan.
3. ESP32 mengolah data dan menentukan kondisi.
4. Data dikirim ke server melalui WebSocket.
5. Dashboard menampilkan status realtime.
6. Jika bahaya → buzzer dan LED aktif otomatis.

Kegunaan Alur:

Menjamin pengiriman data cepat, akurat, dan tanpa jeda waktu.

# KESIMPULAN & MANFAAT

Kesimpulan:

Helm IoT berbasis ESP32 dan sensor DHT22, MQ, MPU6050 mampu:

- Memonitor kondisi lingkungan tambang secara realtime.
- Memberikan peringatan otomatis terhadap bahaya.
- Mengirim data keselamatan ke dashboard untuk pengawasan jarak jauh.

Manfaat:

✅ Meningkatkan keselamatan pekerja tambang.

✅ Mengurangi risiko kecelakaan akibat gas dan suhu ekstrem.

✅ Memudahkan sistem pengawasan modern berbasis IoT.

✅ Efisien, akurat, dan mudah dikembangkan lebih lanjut.

# THANK YOU!