NAMA : L HAFIDL ALKHAIR

KELAS : TRKJ 2C

NIM : 2023903430060

IOT

1. **Pengambilan Data HTTP Menggunakan ESP32**

menghubungkan ESP32 ke jaringan WiFi dan melakukan permintaan HTTP untuk mengambil data dari server.

#include <WiFi.h>

#include <HTTPClient.h>

const char\* ssid = "HOMESTAY 2.4G";

const char\* password = "ckcklahkita";

void setup() {

Serial.begin(115200);

delay(4000);

WiFi.begin(ssid, password);

while (WiFi.status() != WL\_CONNECTED) {

delay(1000);

Serial.println("Connecting to WiFi..");

}

Serial.println("Connected to the WiFi network");

}

void loop() {

if ((WiFi.status() == WL\_CONNECTED)) { //Check the current connection status

HTTPClient http;

http.begin("https://pnl.ac.id/id"); //Specify the URL

int httpCode = http.GET(); //Make the request

if (httpCode > 0) { //Check for the returning code

String payload = http.getString();

Serial.println(httpCode);

Serial.println(payload);

}

else {

Serial.println("Error on HTTP request");

}

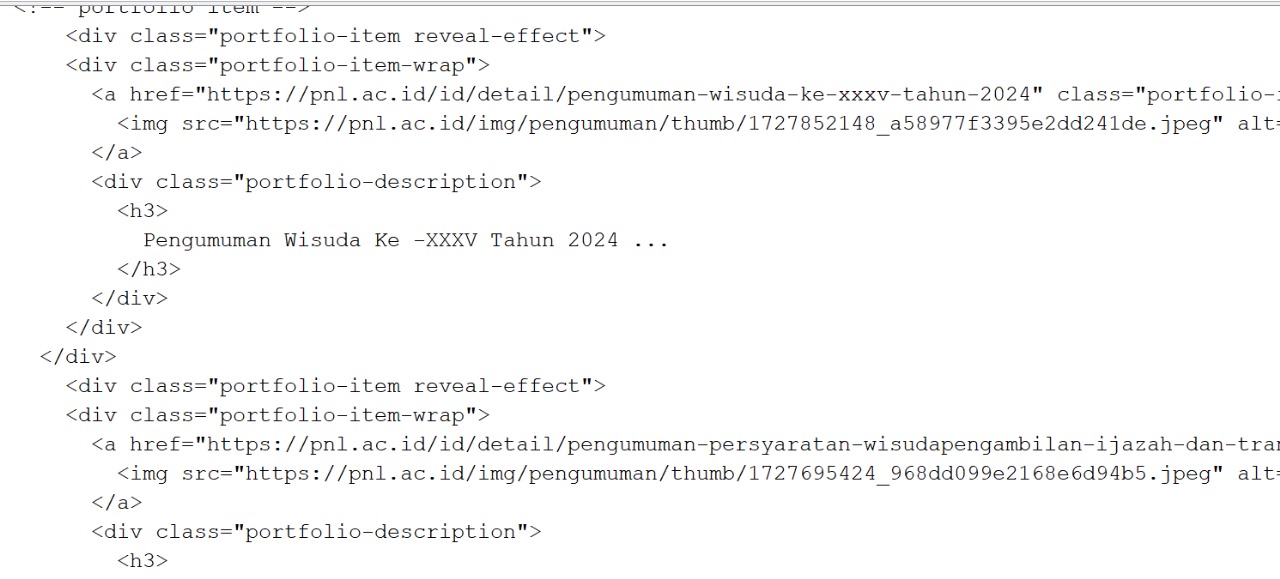
http.end(); //Free the resources

}

delay(10000);

}

Pada program itu maka hasilnya akan seperti ini :



Kita dapat melihat pada hasil, kita mendapatkan data pada website pnl.ac.id

1. WifiScan Esp32

Untuk memindai jaringan WiFi yang tersedia di sekitar ESP32.

#include "WiFi.h"

void setup()

{

Serial.begin(115200);

// Set WiFi to station mode and disconnect from an AP if it was previously connected

WiFi.mode(WIFI\_STA);

WiFi.disconnect();

delay(100);

Serial.println("Setup done");

}

void loop()

{

Serial.println("scan start");

// WiFi.scanNetworks will return the number of networks found

int n = WiFi.scanNetworks();

Serial.println("scan done");

if (n == 0) {

Serial.println("no networks found");

} else {

Serial.print(n);

Serial.println(" networks found");

for (int i = 0; i < n; ++i) {

// Print SSID and RSSI for each network found

Serial.print(i + 1);

Serial.print(": ");

Serial.print(WiFi.SSID(i));

Serial.print(" (");

Serial.print(WiFi.RSSI(i));

Serial.print(")");

Serial.println((WiFi.encryptionType(i) == WIFI\_AUTH\_OPEN)?" ":"\*");

delay(10);

}

}

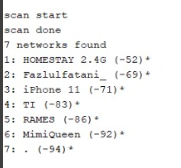
Serial.println("");

// Wait a bit before scanning again

delay(5000);

}

**Hasilnya :**

****

Program ini memungkinkan pengguna untuk melihat jaringan WiFi yang tersedia dan kekuatan sinyalnya, berguna untuk menemukan jaringan yang dapat dihubungkan.

1. **WebSocket**

implementasi server WebSocket menggunakan ESP32 dengan Arduino IDE, yang memungkinkan pengguna untuk mengontrol output (LED) secara remote melalui halaman web

/\*\*\*\*\*\*\*\*\*

Rui Santos

Complete project details at https://RandomNerdTutorials.com/esp32-websocket-server-arduino/

The above copyright notice and this permission notice shall be included in all

copies or substantial portions of the Software.

\*\*\*\*\*\*\*\*\*/

// Import required libraries

#include <WiFi.h>

#include <AsyncTCP.h>

#include <ESPAsyncWebServer.h>

// Replace with your network credentials

const char\* ssid = "HOMESTAY 2.4G";

const char\* password = "ckcklahkita ";

bool ledState = 0;

const int ledPin = 2;

// Create AsyncWebServer object on port 80

AsyncWebServer server(80);

AsyncWebSocket ws("/ws");

const char index\_html[] PROGMEM = R"rawliteral(

<!DOCTYPE HTML><html>

<head>

<title>ESP Web Server</title>

<meta name="viewport" content="width=device-width, initial-scale=1">

<link rel="icon" href="data:,">

<style>

html {

font-family: Arial, Helvetica, sans-serif;

text-align: center;

}

h1 {

font-size: 1.8rem;

color: white;

}

h2{

font-size: 1.5rem;

font-weight: bold;

color: #143642;

}

.topnav {

overflow: hidden;

background-color: #143642;

}

body {

margin: 0;

}

.content {

padding: 30px;

max-width: 600px;

margin: 0 auto;

}

.card {

background-color: #F8F7F9;;

box-shadow: 2px 2px 12px 1px rgba(140,140,140,.5);

padding-top:10px;

padding-bottom:20px;

}

.button {

padding: 15px 50px;

font-size: 24px;

text-align: center;

outline: none;

color: #fff;

background-color: #0f8b8d;

border: none;

border-radius: 5px;

-webkit-touch-callout: none;

-webkit-user-select: none;

-khtml-user-select: none;

-moz-user-select: none;

-ms-user-select: none;

user-select: none;

-webkit-tap-highlight-color: rgba(0,0,0,0);

}

/\*.button:hover {background-color: #0f8b8d}\*/

.button:active {

background-color: #0f8b8d;

box-shadow: 2 2px #CDCDCD;

transform: translateY(2px);

}

.state {

font-size: 1.5rem;

color:#8c8c8c;

font-weight: bold;

}

</style>

<title>ESP Web Server</title>

<meta name="viewport" content="width=device-width, initial-scale=1">

<link rel="icon" href="data:,">

</head>

<body>

<div class="topnav">

<h1>ESP WebSocket Server</h1>

</div>

<div class="content">

<div class="card">

<h2>Output - GPIO 2</h2>

<p class="state">state: <span id="state">%STATE%</span></p>

<p><button id="button" class="button">Toggle</button></p>

</div>

</div>

<script>

var gateway = `ws://${window.location.hostname}/ws`;

var websocket;

window.addEventListener('load', onLoad);

function initWebSocket() {

console.log('Trying to open a WebSocket connection...');

websocket = new WebSocket(gateway);

websocket.onopen = onOpen;

websocket.onclose = onClose;

websocket.onmessage = onMessage; // <-- add this line

}

function onOpen(event) {

console.log('Connection opened');

}

function onClose(event) {

console.log('Connection closed');

setTimeout(initWebSocket, 2000);

}

function onMessage(event) {

var state;

if (event.data == "1"){

state = "ON";

}

else{

state = "OFF";

}

document.getElementById('state').innerHTML = state;

}

function onLoad(event) {

initWebSocket();

initButton();

}

function initButton() {

document.getElementById('button').addEventListener('click', toggle);

}

function toggle(){

websocket.send('toggle');

}

</script>

</body>

</html>

)rawliteral";

void notifyClients() {

ws.textAll(String(ledState));

}

void handleWebSocketMessage(void \*arg, uint8\_t \*data, size\_t len) {

AwsFrameInfo \*info = (AwsFrameInfo\*)arg;

if (info->final && info->index == 0 && info->len == len && info->opcode == WS\_TEXT) {

data[len] = 0;

if (strcmp((char\*)data, "toggle") == 0) {

ledState = !ledState;

notifyClients();

}

}

}

void onEvent(AsyncWebSocket \*server, AsyncWebSocketClient \*client, AwsEventType type,

void \*arg, uint8\_t \*data, size\_t len) {

switch (type) {

case WS\_EVT\_CONNECT:

Serial.printf("WebSocket client #%u connected from %s\n", client->id(), client->remoteIP().toString().c\_str());

break;

case WS\_EVT\_DISCONNECT:

Serial.printf("WebSocket client #%u disconnected\n", client->id());

break;

case WS\_EVT\_DATA:

handleWebSocketMessage(arg, data, len);

break;

case WS\_EVT\_PONG:

case WS\_EVT\_ERROR:

break;

}

}

void initWebSocket() {

ws.onEvent(onEvent);

server.addHandler(&ws);

}

String processor(const String& var){

Serial.println(var);

if(var == "STATE"){

if (ledState){

return "ON";

}

else{

return "OFF";

}

}

return String();

}

void setup(){

// Serial port for debugging purposes

Serial.begin(115200);

pinMode(ledPin, OUTPUT);

digitalWrite(ledPin, LOW);

// Connect to Wi-Fi

WiFi.begin(ssid, password);

while (WiFi.status() != WL\_CONNECTED) {

delay(1000);

Serial.println("Connecting to WiFi..");

}

// Print ESP Local IP Address

Serial.println(WiFi.localIP());

initWebSocket();

// Route for root / web page

server.on("/", HTTP\_GET, [](AsyncWebServerRequest \*request){

request->send\_P(200, "text/html", index\_html, processor);

});

// Start server

server.begin();

}

void loop() {

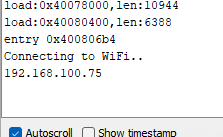
ws.cleanupClients();

digitalWrite(ledPin, ledState);

}

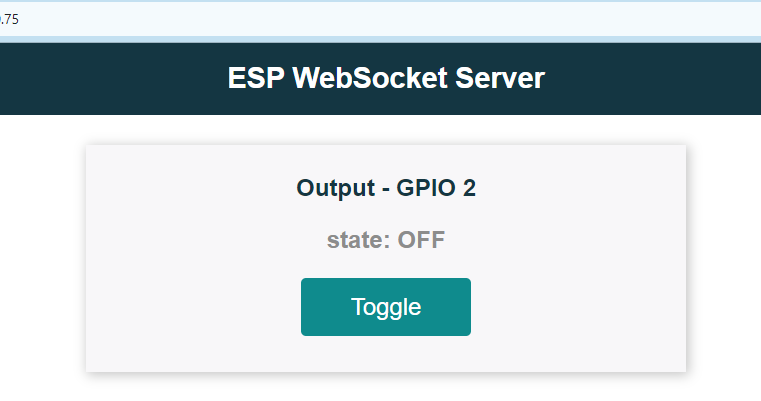
**Hasilnya**

Untuk mengakses ip kita harus masuk ke serial monitor terlebih dahulu untuk mendapatakn ip tersebut



Disini kita mendapatkan ip 192.168.100.75

Kemudian ke google salin ip yang telah kita dapatkan



Pada saat ‘Toggle’ on led akan nya dan ‘Off’ led akan padam karena disini kita tidak memiliki LED maka kita dapat melihatnya pada lampu esp32 warna biru yang artinya menyala.

1. **Kesimpulan**

Pada praktikum kali ini kita dapat melihat esp32 bisa melakukan scaning wifi, pengambilan data pada web, dan mengidupkan lampu LED menggunakan Web socket.