

LAPORAN PRAKTIKUM INTERNET OF THINGS (IoT)

(Simulasi Praktik Akses API Melalui Simulasi WOKWI)



Muhammad Kadavi
Fakultas Vokasi, Universitas Brawijaya
Email: kadavi2945@student.ub.ac.id

Abstrak

Laporan praktikum ini menguji bagaimana cara menghubungkan ESP32 ke API menggunakan simulasi WOKWI. ESP32 dihubungkan ke Wi-Fi untuk mengakses data dari API Laravel yang dipublikasikan via Ngrok, lalu dimodifikasi dengan sensor DHT22 untuk mengukur suhu dan kelembaban. Hasilnya, ESP32 berhasil mengirim data sensor ke database MySQL melalui API dengan metode HTTP POST. Eksperimen ini membuktikan bahwa kombinasi ESP32, WOKWI, dan Laravel-Ngrok dapat digunakan untuk membuat sistem IoT sederhana yang terintegrasi dengan database.

Pendahuluan

API kini menjadi bagian penting dalam pembuatan aplikasi web atau mobile untuk menghubungkan sistem dengan server. Namun, pengembang sering terkendala saat ingin menguji API yang masih berjalan di komputer lokal, karena tidak bisa diakses dari perangkat lain. Penelitian ini menggunakan Laravel 11, framework PHP yang memudahkan pembuatan API berkat fitur seperti pengaturan alur data dan koneksi database, serta Ngrok untuk mengubah server lokal menjadi tautan publik sementara. Tujuannya adalah menunjukkan cara praktis membuat dan menguji API sehingga pengembang bisa fokus pada fungsi utamanya tanpa kesulitan teknis rumit.

Latar Belakang

Sistem IoT seperti sensor atau perangkat pintar juga semakin populer. Penggunaannya tidak hanya di rumah, tetapi juga di pertanian, atau industri, untuk mengotomasi sesuatu menggunakan data yang matang. Untuk dapat menyimpan dan memproses data dari perangkat IoT ke cloud, dapat dihubungkan dan dipisahkan dengan API. Namun, menghubungkan perangkat IoT ke API dapat menjadi tugas yang sangat sulit, terutama jika yang membaca tidak memiliki peralatan fisik yang siap. Salah satu contoh penelitian ini adalah menggunakan ESP32 yang disimulasikan di platform Wokwi, saat itu juga baru digunakan untuk mengetes bagaimana cara mengakses API Laravel. Laravel dijadikan sumber api-nya karena relatif sederhana untuk membuatnya. Sedangkan platform Wokwi adalah alat simulasi kode maupun koneksi internet yang nyata sehingga tidak perlu punya perangkat fisik sendiri. Penelitian ini hanya dijadikan sebagai contoh atau tumpuan untuk belajar saja, agar dapat memahami bagaimana data dari IoT bisa dikirim melalui server menggunakan API.

Tujuan

1. Memahami bagaimana API dibuat dengan menggunakan laravel 11
2. Menggunakan ngrok agar endpointnya bisa diakses secara global
3. Menguji endpoint API menggunakan Postman atau browser.
4. Mengintegrasikan antara api laravel dan wokwi internet
5. Mengirim data sensor suhu dan kelembaban ke database menggunakan metode HTTP POST.
6. Melakukan simulasi iot virtual dan verifikasi keberhasilan akses API di WOKWI.

Metodologi

Eksperimen ini dilakukan dengan mensimulasikan bagaimana api bekerja dalam mengirim data ke dalam database dan menyimpannya di database.

Software & Hardware

Software	Hardware
Laravel 11	Laptop
Php dan composer	ESP32 (Virtual)
Mysql	Sensor Kelembapan DHT22 (virtual)
Postman	
VScode	
Platform IO	
WOKWI	

2.2 Implementation Steps (Langkah Implementasi)

1. 2.2.1 Menjalankan API Laravel

2. Jalankan perintah berikut di terminal untuk mengaktifkan API Laravel:

3. `php artisan serve --host=0.0.0.0 --port=8080`

4. Perintah ini memastikan API dapat diakses dari IP manapun pada port 8080.

5. 2.2.2 Menyiapkan Simulasi di WOKWI

6. **Buat file baru di WOKWI dan tambahkan file `wokwi.toml`**

7. `[wokwi]`

8. `version = 1`

9. `firmware = '.pio\build\esp32doit-devkit-v1\firmware.bin'`

10. `elf =`

`'C:\Users\mokor\Documents\PlatformIO\Projects\wokwi_internet\.pio\build\esp32doit-devkit-v1\firmware.elf'`

11. **Buat file `diagram.json`**

12. `{`

13. `"version": 1,`

14. `"author": "Uri Shaked",`

15. `"editor": "wokwi",`

16. `"parts": [`

17. `{ "type": "board-esp32-devkit-c-v4", "id": "esp", "top": 0, "left": 0, "attrs": {} }]`

18. `],`

19. `"connections": [`

20. `["esp:TX", "$serialMonitor:RX", "", []],`

21. `["esp:RX", "$serialMonitor:TX", "", []]`

22. `]`

23. `}`

24. **Tambahkan file `platformio.ini` dengan konfigurasi berikut:**

25. `monitor_speed = 115200`

26. `lib_deps = adafruit/DHT sensor library`

27. 2.2.3 Implementasi Kode

28. **Buat file `main.cpp` dengan kode berikut:**

29. `#include <WiFi.h>`

30. `#include <HTTPClient.h>`

31. `#include "DHT.h"`

32.

33. `#define DHTPIN 27`

34. `#define DHTTYPE DHT22`

35. `DHT dht(DHTPIN, DHTTYPE);`

36.

37. `const char* ssid = "Wokwi-GUEST";`

38. `const char* password = "";`

39. `const char* serverUrl = "http://your-ngrok-url/api/posts";`

40. `unsigned long previousMillis = 0;`

41. `const long interval = 5000;`

42.

43. `void setup() {`

44. `Serial.begin(115200);`

45. `WiFi.begin(ssid, password);`

46. `while (WiFi.status() != WL_CONNECTED) {`

47. `delay(500);`

48. `Serial.print(".");`

49. `}`

50. `Serial.println(" Terhubung!");`

51. `dht.begin();`

52. `}`

53.

54. `void loop() {`

55. `unsigned long currentMillis = millis();`

56. `if (currentMillis - previousMillis >= interval) {`

57. `previousMillis = currentMillis;`

58. `float h = dht.readHumidity();`

59. `float t = dht.readTemperature();`

60. `if (isnan(h) || isnan(t)) {`

61. `Serial.println("Sensor gagal dibaca!");`

62. `return;`

```

63.         }
64.         HTTPClient http;
65.         http.begin(serverUrl);
66.         http.addHeader("Content-Type", "application/json");
67.         String payload = "{\"nama_sensor\":\"Sensor GD\", \"nilai1\":\" +
String(h) + \", \"nilai2\":\" + String(t) + \"}\"";
68.         Serial.println(payload);
69.         int httpResponseCode = http.POST(payload);
70.         Serial.print("Kode status HTTP: ");
71.         Serial.println(httpResponseCode);
72.         http.end();
73.     }
74. }

```

75. **2.2.4 Menjalankan Simulasi**

76. Gunakan perintah berikut untuk menjalankan simulasi di WOKWI:

```

77. Wokwi Start Simulator
78.

```

79. **3. Results and Discussion (Hasil dan Pembahasan)**

80. **ESP32 berhasil terhubung ke WiFi Wokwi-GUEST**

81. Menghubungkan ke WiFi... Terhubung!

82. **ESP32 berhasil mengakses API Laravel**

```

83. Kode status HTTP: 200
84. Respons dari server: {"success":true,"message":"Data tersimpan"}
85.

```

3. Results and Discussion (Hasil dan Pembahasan)

Hasil dari simulasi menunjukkan bahwa endpoint api dapat menampilkan data dan juga mengirim data dalam bentuk json dengan menggunakan metode GET dan POST, lalu menyimpannya di dalam database.

3.1 Experimental Results (Hasil Eksperimen)

phpMyAdmin interface showing the database structure and data for the 'transaksi_sensor' table.

Database: **iot_25** | Table: **transaksi_sensor**

	id	nama_sensor	nilai1	nilai2	created_at	updated_at
<input type="checkbox"/>	2	Sensor A	100	200	NULL	NULL
<input type="checkbox"/>	3	Sensor B	87	176	NULL	NULL
<input type="checkbox"/>	4	Sensor C	111	211	2025-03-07 06:54:58	2025-03-07 06:54:58
<input type="checkbox"/>	5	Sensor D	123	456	2025-03-14 07:26:43	2025-03-07 07:26:43
<input type="checkbox"/>	6	Sensor AD	123	234	2025-03-14 07:51:57	2025-03-14 07:51:57
<input type="checkbox"/>	7	Sensor AD	123	234	2025-03-14 07:53:46	2025-03-14 07:53:46
<input type="checkbox"/>	8	Sensor kd	125	284	2025-03-14 07:54:37	2025-03-14 07:54:37
<input type="checkbox"/>	9	Sensor io	125	284	2025-03-14 07:55:58	2025-03-14 07:55:58
<input type="checkbox"/>	10	Sensor io	125	284	2025-03-14 08:08:55	2025-03-14 08:08:55
<input type="checkbox"/>	11	Sensor GD	51	27	2025-03-14 08:10:02	2025-03-14 08:10:02
<input type="checkbox"/>	12	Sensor GD	83	80	2025-03-14 08:10:10	2025-03-14 08:10:10
<input type="checkbox"/>	13	Sensor GD	83	80	2025-03-14 08:10:21	2025-03-14 08:10:21

Wokwi Simulator interface showing an ESP32 microcontroller connected to a DHT22 sensor. The terminal displays HTTP responses from the server.

WOKWI Simulator

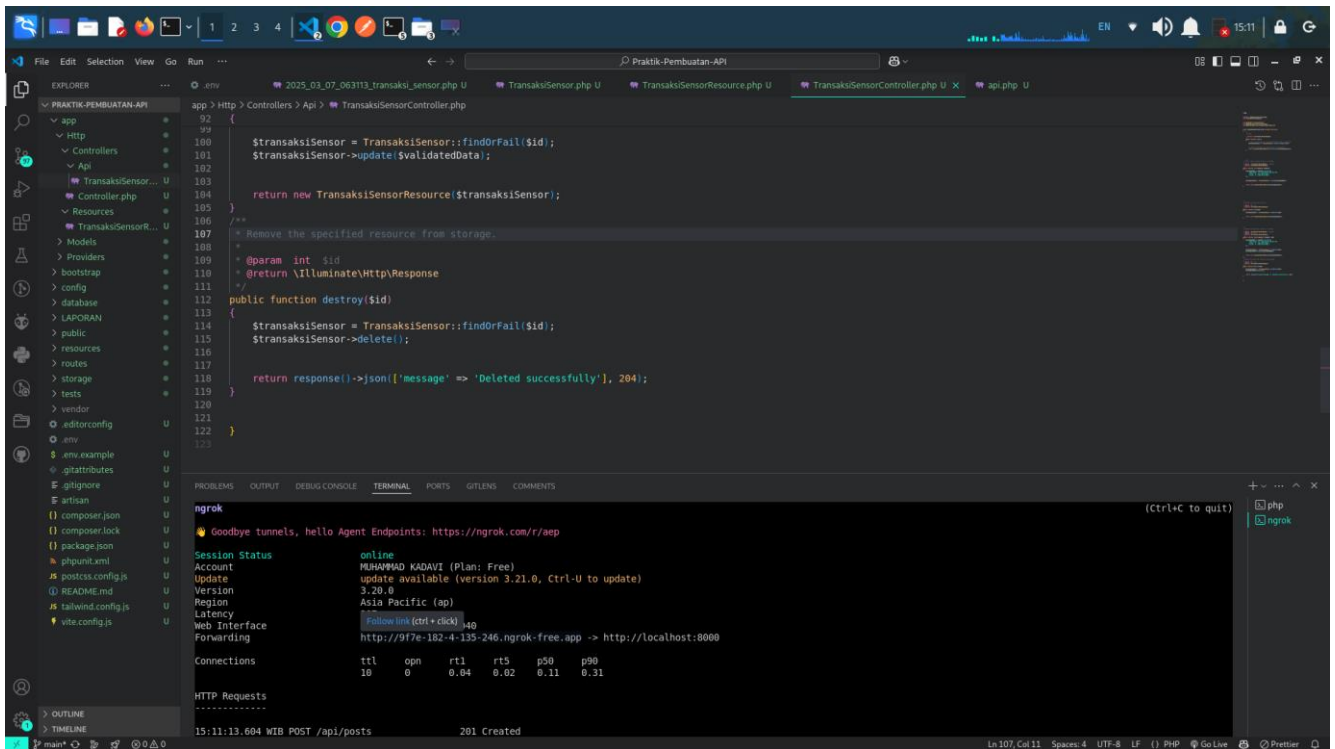
ESP32

DHT22

Terminal Output:

```
Kode respons HTTP: 201
Respons dari server:
{"data":{"id":14,"nama_sensor":"Sensor GD","nilai1":83,"nilai2":80}}
{"nama_sensor":"Sensor GD", "nilai1":83.00, "nilai2":80.00}
Kode respons HTTP: 201
Respons dari server:
{"data":{"id":15,"nama_sensor":"Sensor GD","nilai1":83,"nilai2":80}}
{"nama_sensor":"Sensor GD", "nilai1":83.00, "nilai2":80.00}
Kode respons HTTP: 201
Respons dari server:
{"data":{"id":16,"nama_sensor":"Sensor GD","nilai1":83,"nilai2":80}}
{"nama_sensor":"Sensor GD", "nilai1":83.00, "nilai2":80.00}
Kode respons HTTP: 201
Respons dari server:
{"data":{"id":17,"nama_sensor":"Sensor GD","nilai1":83,"nilai2":80}}
{"nama_sensor":"Sensor GD", "nilai1":83.00, "nilai2":80.00}
Kode respons HTTP: 201
Respons dari server:
{"data":{"id":18,"nama_sensor":"Sensor GD","nilai1":83,"nilai2":80}}
{"nama_sensor":"Sensor GD", "nilai1":83.00, "nilai2":80.00}
```

4. Appendix (Lampiran, jika diperlukan)



The screenshot shows a Visual Studio Code editor with a PHP project named 'Praktik-Pembuatan-API'. The file explorer on the left shows the project structure, including controllers, resources, and models. The main editor displays the 'TransaksiSensorController.php' file, which contains the following code:

```
92 {
93     $transaksiSensor = TransaksiSensor::findOrFail($id);
94     $transaksiSensor->update($validatedData);
95 }
96
97 return new TransaksiSensorResource($transaksiSensor);
98 }
99
100 /**
101  * Remove the specified resource from storage.
102  *
103  * @param int $id
104  * @return \Illuminate\Http\Response
105  */
106 public function destroy($id)
107 {
108     $transaksiSensor = TransaksiSensor::findOrFail($id);
109     $transaksiSensor->delete();
110 }
111
112 return response()->json(['message' => 'Deleted successfully'], 204);
113 }
114 }
```

The terminal window at the bottom shows the output of the 'ngrok' command, indicating that the application is running successfully on port 8000. The terminal output includes the following information:

```
ngrok
Goodbye tunnels, hello Agent Endpoints: https://ngrok.com/r/r/aeap

Session Status
Account      online
Account     MUHAMMAD KADAVI (Plan: Free)
Update      update available (version 3.21.0, Ctrl-U to update)
Version      3.20.0
Region      Asia Pacific (ap)
Latency      140
Web Interface http://917e-182-4-135-246.ngrok-free.app -> http://localhost:8000

Connections
t1l  opn  r1l  r1s  p50  p90
10   0     0.04 0.02 0.11 0.31

HTTP Requests
-----
15:11:13.604 WIB POST /api/posts 201 Created
```

```

1 #include <Arduino.h>
2 #include <WiFi.h>
3 #include <HTTPClient.h>
4
5
6 #include "DHT.h"
7
8
9
10
11 #define DHTPIN 27
12 #define DHTTYPE DHT22
13
14
15
16
17 DHT dht(DHTPIN, DHTTYPE);
18
19
20
21
22
23
24 // Ganti dengan kredensial WiFi Anda
25 const char* ssid = "Wokwi-GUEST";
26 const char* password = "";
27
28
29 unsigned long previousMillis = 0;
30 const long interval = 5000; // Interval 5 detik (5000 ms)
31
32
33 void setup() {
34   Serial.begin(115200);
35
36   // Hubungkan ke WiFi
37   WiFi.begin(ssid, password);
38   Serial.print("Menghubungkan ke WiFi");
39   while (WiFi.status() != WL_CONNECTED) {
40     delay(500);
41     Serial.print(".");
42   }
43   Serial.println(" Terhubung!");
44
45
46
47
48
49
50
51
52 dht.begin();
53
54 // Tunggu sebentar agar koneksi stabil
55 delay(1000);
56 }
57
58
59 void loop() {
60   unsigned long currentMillis = millis();
61
62
63   // Lakukan POST setiap interval yang telah ditentukan
64   if (currentMillis - previousMillis >= interval) {
65     previousMillis = currentMillis;
66
67
68
69
70
71
72     float h = round(dht.readHumidity());
73     // Read temperature as Celsius (the default)
74     float t = round(dht.readTemperature());
75
76
77     // Check if any reads failed and exit early (to try again).
78     if (isnan(h) || isnan(t)) {
79       Serial.println(F("Failed to read from DHT sensor!"));
80       return;
81     }
82
83
84
85     // Compute heat index in Celsius (isFahrenheit = false)
86     float hic = dht.computeHeatIndex(t, h, false);
87
88
89
90
91 // Inisialisasi HTTPClient
92 HTTPClient http;
93 String url = "http://917e-182-4-135-246.ngrok-free.app/api/posts"; // Ganti dengan URL ngrok yang benar
94
95
96 http.begin(url); // Menggunakan HTTP, bukan HTTPS
97 http.addHeader("Content-Type", "application/json");
98
99
100
101
102 String payload = "{\"nama_sensor\":\"Sensor GD\", \"nilai1\":\"" + String(h) + "\", \"nilai2\":\"" + String(t) + "\"}";
103 ;
104
105 Serial.println(payload); // Untuk melihat apakah payload sudah terbentuk dengan benar
106
107
108 // Kirim POST request
109 int httpStatusCode = http.POST(payload);
110
111 // Tampilkan kode respons HTTP
112 Serial.print("Kode respons HTTP: ");
113 Serial.println(httpStatusCode);
114
115
116 // Tampilkan respons dari server jika request berhasil
117 if (httpStatusCode == 200 || httpStatusCode == 201) {
118   String response = http.getString();
119   Serial.println("Respons dari server:");
120   Serial.println(response);
121 } else {
122   Serial.println("Gagal mengirim data");
123 }
124
125
126 // Tutup koneksi HTTP
127 http.end();
128 }
129 }
130

```