

# **LAPORAN PRAKTIKUM IOT**

## **SIMULASI RELAY, SENSOR JARAK, DAN PEMBUATAN API**

*Yasiva Nurul Ramadhan – 233140707111067 – T4A – Teknologi Informasi*

*Fakultas Vokasi, Universitas Brawijaya*

*Email: [ysvlhn@student.ub.ac.id](mailto:ysvlhn@student.ub.ac.id)*

Internet of Things (IoT) adalah teknologi yang memungkinkan perangkat elektronik untuk saling terhubung dan berkomunikasi melalui jaringan internet. Laporan ini mencakup tiga praktikum yang berkaitan dengan IoT, yaitu: (1) Simulasi penggunaan relay, button, dan LED menggunakan ESP32 di platform Wokwi, (2) Simulasi sensor jarak ultrasonik untuk pengukuran jarak menggunakan ESP32, dan (3) Pembuatan API berbasis Laravel 11 dan pengujian dengan Postman serta Ngrok untuk konektivitas online. Hasil dari ketiga praktikum ini menunjukkan bahwa penggunaan simulasi dapat mempermudah pengembangan sistem IoT sebelum implementasi perangkat keras secara langsung. Pemrograman berbasis ESP32 dengan C++ serta backend API berbasis Laravel memungkinkan komunikasi yang efisien antara perangkat dan sistem cloud. Dengan pemahaman ini, pengembangan aplikasi IoT dapat dilakukan secara lebih sistematis dan terstruktur.

Kata kunci— ESP32, relay, sensor ultrasonik, Laravel, API

### **1. Pendahuluan**

#### **1.1 Latar belakang**

Internet of Things (IoT) telah berkembang pesat dalam berbagai bidang seperti industri, kesehatan, dan smart home. Teknologi ini memungkinkan perangkat elektronik untuk berkomunikasi dan saling berinteraksi secara otomatis melalui jaringan internet. Dengan adanya IoT, berbagai perangkat dapat dikendalikan secara jarak jauh, memberikan kemudahan dalam operasional serta meningkatkan efisiensi dan efektivitas di berbagai sektor. Penerapan IoT juga mendukung perkembangan sistem otomatisasi yang semakin canggih, termasuk dalam bidang pengendalian perangkat elektronik, pengukuran jarak, dan komunikasi data.

Dalam praktikum ini, dilakukan tiga percobaan untuk memahami prinsip dasar dari IoT. Percobaan pertama berfokus pada kontrol perangkat elektronik menggunakan relay, button, dan LED. Percobaan kedua menerapkan sensor ultrasonik untuk pengukuran jarak dalam simulasi. Sementara itu, percobaan ketiga membahas pembuatan API untuk komunikasi antara perangkat IoT dan sistem backend. Dengan pendekatan ini, diharapkan pemahaman terhadap implementasi IoT dapat diperoleh secara lebih komprehensif sebelum diterapkan dalam perangkat keras sesungguhnya.

## **1.2 Tujuan Eksperimen**

Praktikum ini bertujuan untuk:

1. Memahami dasar penggunaan ESP32 dalam simulasi sistem IoT.
2. Menerapkan sensor ultrasonik untuk pengukuran jarak dalam simulasi.
3. Mempelajari cara membangun API berbasis Laravel untuk komunikasi data IoT.
4. Menggunakan platform simulasi seperti Wokwi untuk menguji sistem sebelum implementasi perangkat keras.

## **2. Metodologi**

### **2.1 Alat dan Bahan**

Dalam praktikum ini, alat dan bahan yang digunakan meliputi:

1. Komputer/Laptop dengan koneksi internet
2. Platform Wokwi
3. ESP32 Devkit
4. Software Visual Studio Code
5. Sensor Ultrasonik HC-SR04
6. Relay, LED, dan push button
7. Laravel 11, Postman, dan Ngrok
8. Database MySQL (phpMyAdmin)

### **2.2 Langkah Implementasi**

#### **2.2.1 Simulasi Relay, Button & LED**

1. Membuka Wokwi dan membuat proyek baru dengan ESP32.
2. Menambahkan relay, LED, dan tombol pada diagram rangkaian.
3. Menulis kode program dalam C++ untuk mengontrol relay dan LED berdasarkan input tombol.
4. Melakukan kompilasi dan menjalankan simulasi.

#### **2.2.2 Simulasi Sensor Jarak**

1. Membuka Wokwi dan memilih ESP32 sebagai mikrokontroler.
2. Menambahkan sensor ultrasonik HC-SR04 pada diagram.
3. Menulis kode untuk membaca jarak dari sensor ultrasonik.
4. Melakukan kompilasi kode dan menjalankan simulasi.

#### **2.2.3 Pembuatan API**

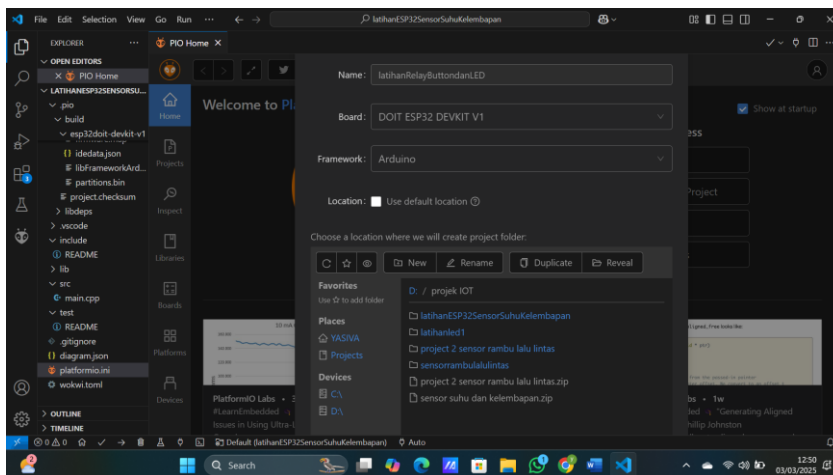
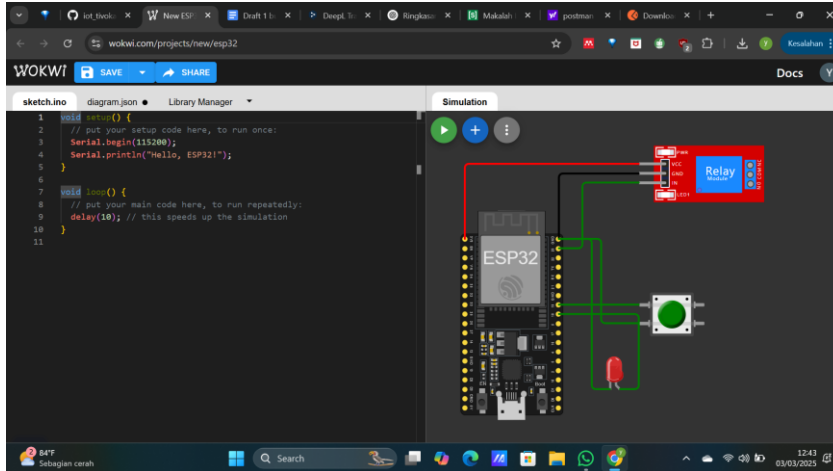
1. Membuat database di phpMyAdmin dengan nama `iot_25`.
2. Membuat proyek Laravel 11 dan mengatur file konfigurasi.
3. Mengembangkan endpoint API untuk pengolahan data sensor.
4. Menggunakan Postman untuk menguji endpoint API dengan metode GET dan POST.
5. Menjalankan Ngrok untuk menghubungkan API secara online.

### 3. Hasil dan Pembahasan

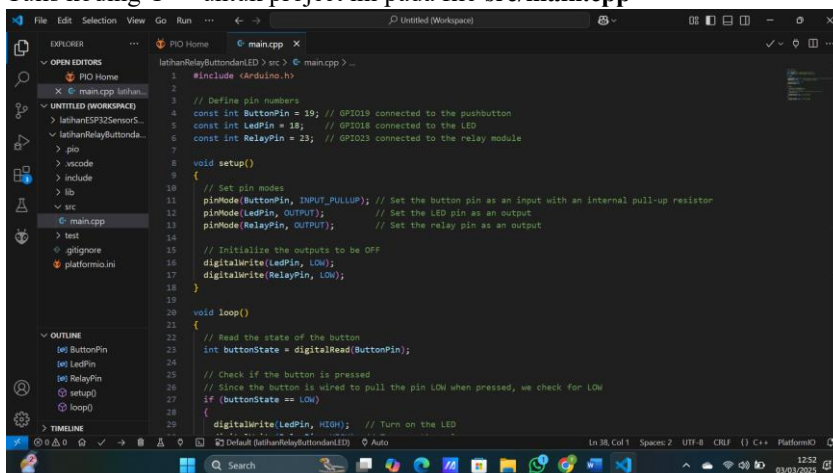
#### 3.1 Simulasi Relay, Button & LED

Hasil dari praktikum ini adalah sebagai berikut:

1. Buka web wokwi.com lalu membuat diagram ESP32 dan sensor Relay.



2. Tulis koding C++ untuk project ini pada file src/main.cpp



```

1 // lathanRelayButtonLED > src > main.cpp > loop()
2 #include <Arduino.h>
3
4 // Define pin numbers
5 const int ButtonPin = 19; // GPIO19 connected to the pushbutton
6 const int LedPin = 18; // GPIO18 connected to the LED
7 const int RelayPin = 23; // GPIO23 connected to the relay module
8
9 void setup()
10 {
11     // Set pin modes
12     pinMode(ButtonPin, INPUT_PULLUP); // Set the button pin as an input with an internal pull-up resistor
13     pinMode(LedPin, OUTPUT); // Set the LED pin as an output
14     pinMode(RelayPin, OUTPUT); // Set the relay pin as an output
15 }
16
17 void loop()
18 {
19 }
20
21 // End of file

```

Linking .pio\build\esp32doit-devkit-v1\firmware.elf  
Retrieving maximum program size .pio\build\esp32doit-devkit-v1\firmware.elf  
Checking size .pio\build\esp32doit-devkit-v1\firmware.elf  
Advanced Memory Usage is available via "PlatformIO Home > Project Inspect"  
RAM: [=====] 6.4K (used 21112 bytes from 32768 bytes)  
Flash: [=====] 18.1K (used 239968 bytes from 1310720 bytes)  
Building .pio\build\esp32doit-devkit-v1\firmware.bin  
esptool.py v4.5.1  
Creating esp32 image...  
Merged 2 ELF sections  
Successfully created esp32 image.  
===== [SUCCESS] Took 10.90 seconds =====  
Terminal will be reused by tasks, press any key to close it.

3. Buat file baru wokwi.toml, dan isikan file tersebut dengan koding sebagai berikut:

```

1 [wokwi]
2 version = 1
3 firmware = ".pio\build\esp32doit-devkit-v1\firmware.bin"
4 elf = ".pio\build\esp32doit-devkit-v1\firmware.elf"
5

```

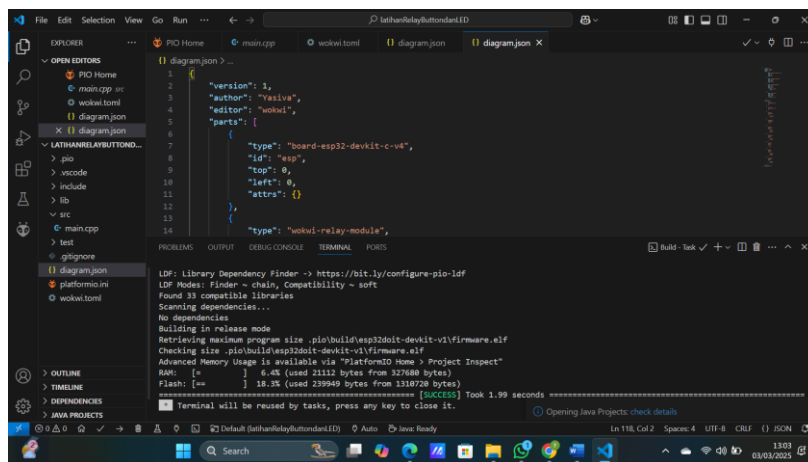
4. Buat file baru diagram.json dan copy paste dari diagram.json pada platform online wokwi.com

```

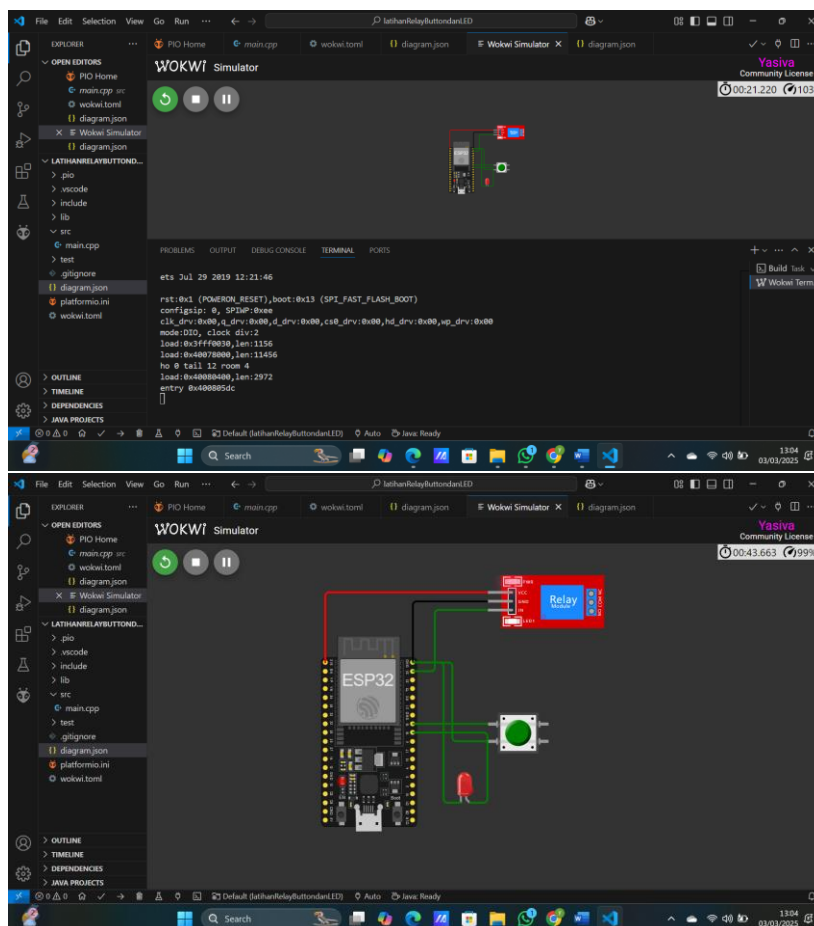
1 {
2     "version": 1,
3     "author": "Yasiva",
4     "editor": "wokwi",
5     "parts": [
6         {
7             "type": "board-esp32-devkit-c-v4",
8             "id": "esp",
9             "top": 0,
10            "left": 0,
11            "attrs": {}
12        },
13        {
14            "type": "wokwi-relay-module",
15            "id": "relay1",
16            "top": 67,
17            "left": 182.4,
18            "attrs": {}
19        },
20        {
21            "type": "wokwi-pushbutton",
22            "id": "btn1",
23            "top": 83,
24            "left": 182.4,
25            "attrs": {
26                "color": "green",
27                "xray": "1"
28            }
29        }
30    ]
31 }

```

5. Melakukan compile pada file main.cpp



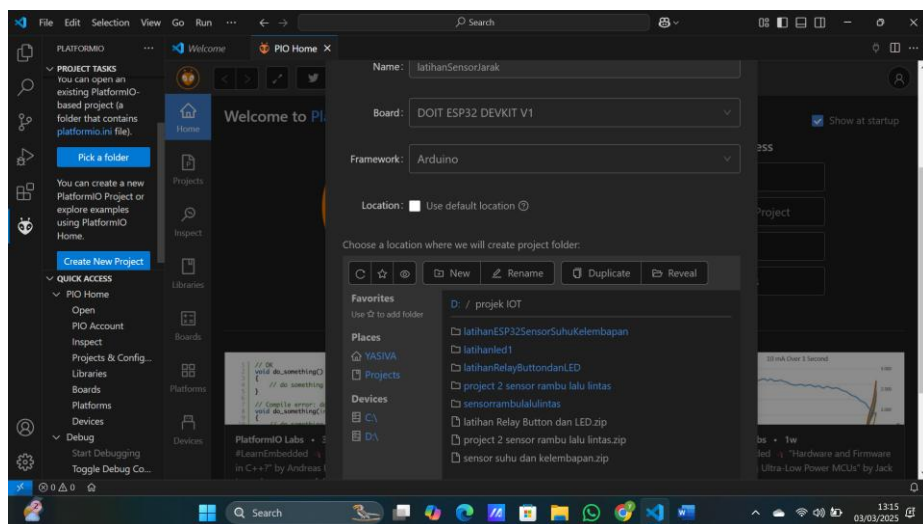
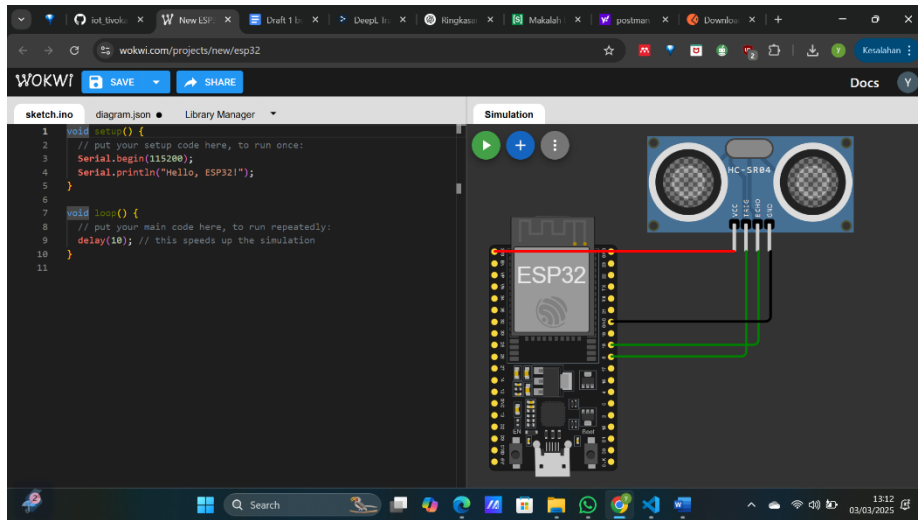
6. Menjalankan simulasi



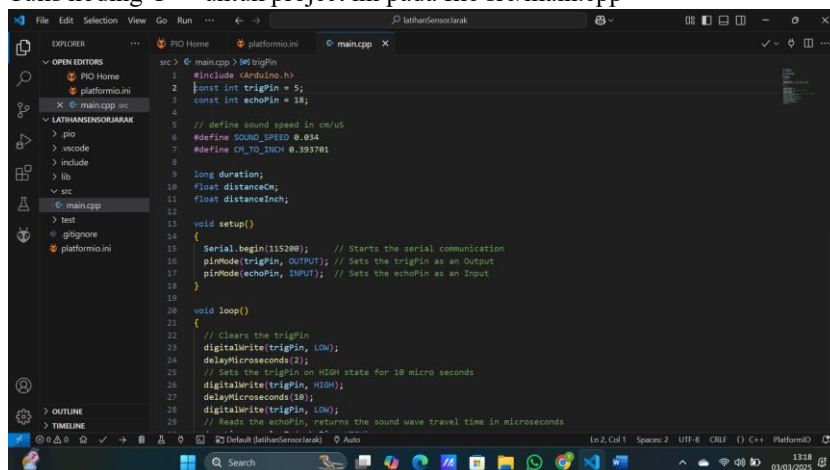
### 3.2 Simulasi Sensor Jarak

Hasil dari praktikum ini adalah sebagai berikut:

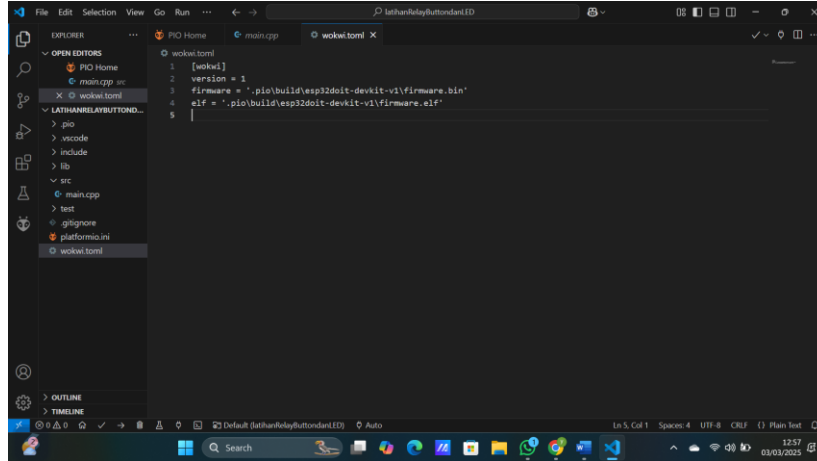
1. Buka web wokwi.com lalu membuat diagram ESP32 dan sensor Ultrasonik HC-SR04.



2. Tulis koding C++ untuk project ini pada file src/main.cpp

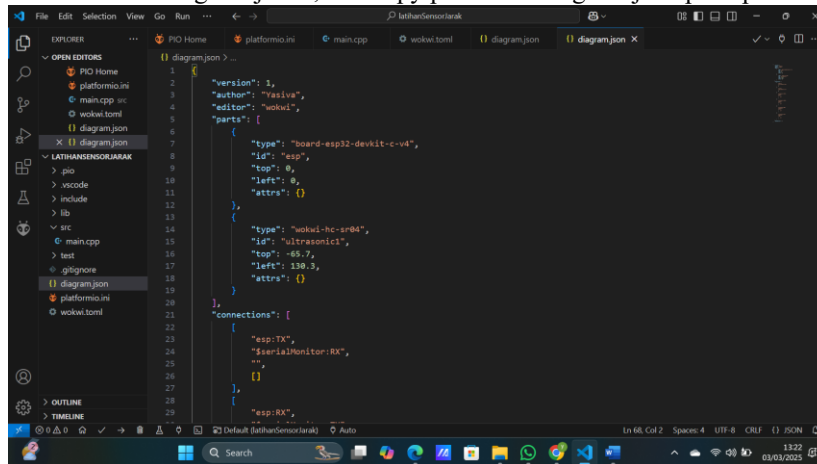


3. Buat file baru wokwi.toml, dan isikan file tersebut dengan koding sebagai berikut:



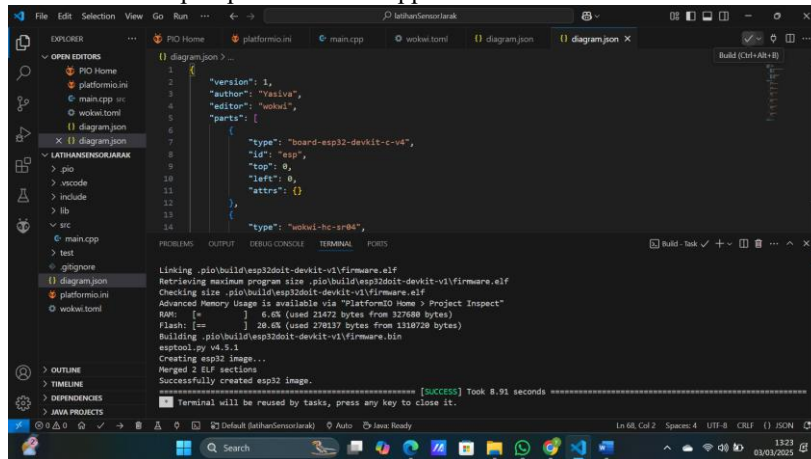
```
1 [wokwi]
2 version = 1
3 firmware = '.pio/build/esp32doit-devkit-v1/firmware.bin'
4 elf = '.pio/build/esp32doit-devkit-v1/firmware.elf'
5
```

4. Buat file baru diagram.json , dan copy paste dari diagram.json pada platform online wokwi.com



```
1 {
2   "version": 1,
3   "author": "Yasiva",
4   "editor": "wokwi",
5   "parts": [
6     {
7       "type": "board-esp32-devkit-c-v4",
8       "id": "esp",
9       "top": 0,
10      "left": 0,
11      "attrs": {}
12    },
13    {
14      "type": "wokwi-hc-sr04",
15      "id": "ultrasonic1",
16      "top": -65.7,
17      "left": 130.3,
18      "attrs": {}
19    }
20  ],
21   "connections": [
22     {
23       "esp:TX",
24       "SerialMonitor:RX",
25       ""
26     },
27     {
28       "esp:RX",
29       ""
30     }
31  ]
32}
```

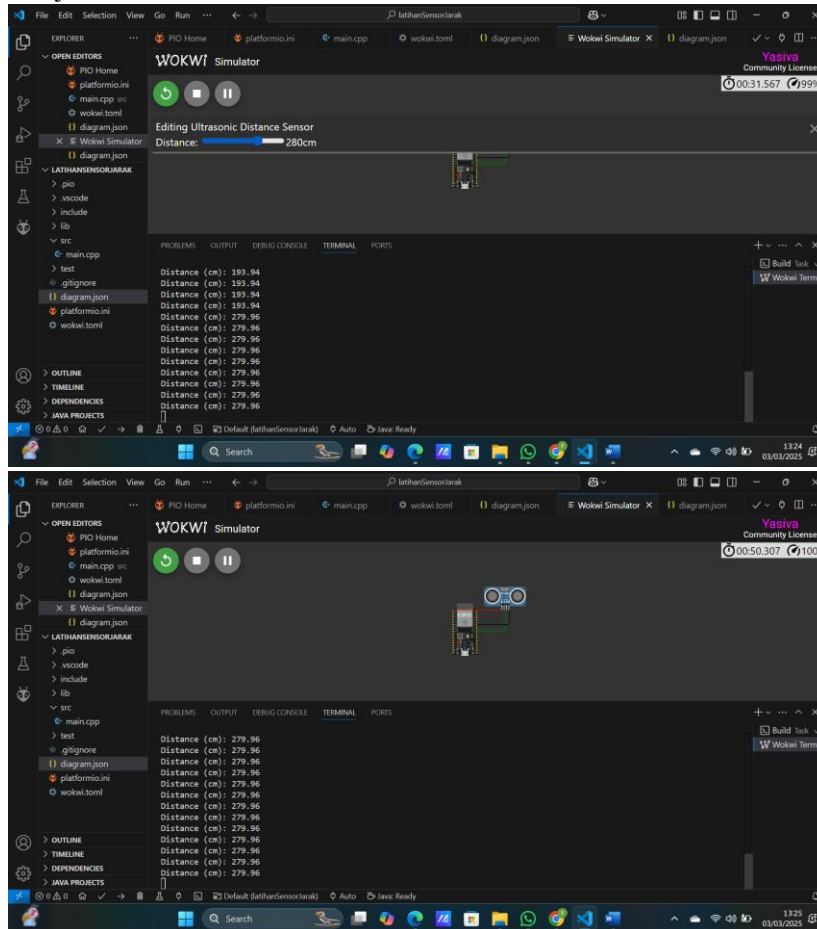
5. Melakukan compile pada file main.cpp



```
1 {
2   "version": 1,
3   "author": "Yasiva",
4   "editor": "wokwi",
5   "parts": [
6     {
7       "type": "board-esp32-devkit-c-v4",
8       "id": "esp",
9       "top": 0,
10      "left": 0,
11      "attrs": {}
12    },
13    {
14      "type": "wokwi-hc-sr04",
15      "id": "ultrasonic1",
16      "top": -65.7,
17      "left": 130.3,
18      "attrs": {}
19    }
20  ],
21   "connections": [
22     {
23       "esp:TX",
24       "SerialMonitor:RX",
25       ""
26     },
27     {
28       "esp:RX",
29       ""
30     }
31  ]
32}
```

```
Linking .pio/build/esp32doit-devkit-v1/firmware.elf
Retrieving maximum program size .pio/build/esp32doit-devkit-v1/firmware.elf
Checking size .pio/build/esp32doit-devkit-v1/firmware.elf
Advanced Memory Usage is available via "PlatformIO Home" > "Project Inspector"
RAM: [=====] 6.6% (used 21472 bytes from 327680 bytes)
Flash: [=====] 28.4% (used 276137 bytes from 1310720 bytes)
Building .pio/build/esp32doit-devkit-v1/firmware.bin
esptool.py v4.3.1
Creating esp32 image...
Merged 2 ELF sections
Successfully created esp32 image.
===== [SUCCESS] Took 8.91 seconds =====
Terminal will be reused by tasks, press any key to close it.
```

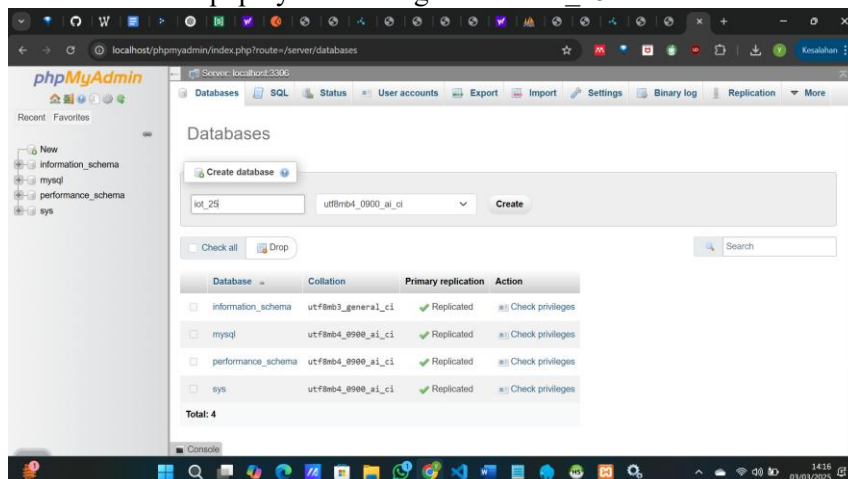
## 6. Menjalankan simulasi



## 3.3 Pembuatan API

Hasil dari praktikum ini adalah sebagai berikut:

### 1. Buat database di phpmyadmin dengan nama iot\_25





## 2. Membuat File Laravel

```
config Sets config options
create-project Creates new project from a package into given directory
depends [why] Shows which packages cause the given package to be installed
diagnose Diagnoses the system to identify common errors
dump-autoload [dumpautoload] Dumps the autoloader
exec Executes a vendored binary/script
fund Discover how to help fund the maintenance of your dependencies
global Allows running commands in the global composer dir ($COMPOSER_HOME)
help Display help for a command
init Creates a basic composer.json file in current directory
install [i] Installs the project dependencies from the composer.lock file if present, or falls back on th
composer.json
licenses Shows information about licenses of dependencies
list List commands
outdated Shows a list of installed packages that have updates available, including their latest version
[why-not] Shows which packages prevent the given package from being installed
reinstall Uninstalls and reinstalls the given package names
remove [re]uninstall] Removes a package from the require or require-dev
require [r] Adds required packages to your composer.json and installs them
run-script [run] Runs the scripts defined in composer.json
search Searches for packages
self-update [selfupdate] Updates composer.phar to the latest version
show [info] Shows information about packages
status Shows a list of locally modified packages
suggests Shows package suggestions
update [u]upgrade] Updates your dependencies to the latest version according to composer.json, and updat
es the composer.lock file
validate Validates a composer.json and composer.lock

C:\Users\YASIVA>composer global require laravel/installer
```

```
projek IOT
C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.26100.3194]
(c) Microsoft Corporation. All rights reserved.

D:\projek IOT>laravel new

Laravel

What is the name of your project?:
> latihan12
```

```
C:\Windows\System32\cmd.exe
run 'npm fund' for details

found 0 vulnerabilities
npm notice
npm notice New major version of npm available! 10.9.0 -> 11.1.0
npm notice Changelog: https://github.com/npm/cli/releases/tag/v11.1.0
npm notice To update run: npm install -g npm@11.1.0
npm notice

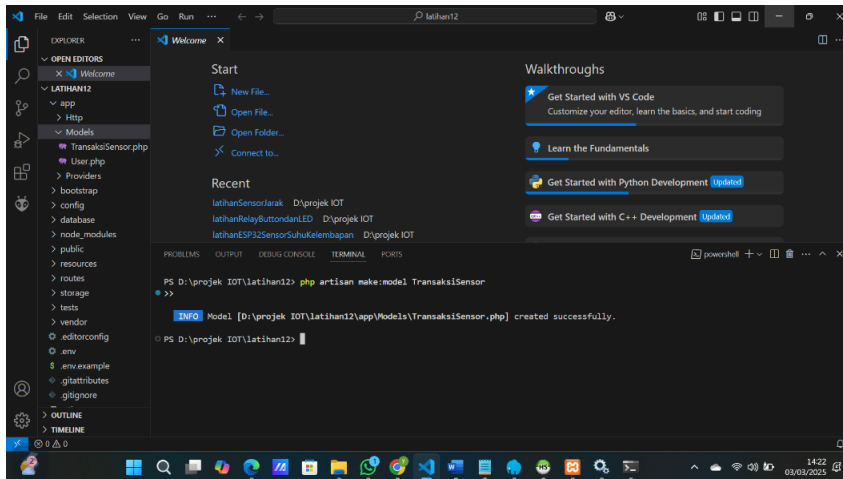
> build
> vite build

vite v6.2.0 building for production...
transforming...
53 modules transformed.
rendering chunks...
computing gzip size...
public/build/manifest.json 0.27 kB | gzip: 0.15 kB
public/build/assets/app-080c111.css 19.43 kB | gzip: 4.58 kB
public/build/assets/app-083ay3e-.js 35.03 kB | gzip: 14.03 kB
✓ built in 341ms
[INFO] Application ready in [latihan12]. You can start your local development using:

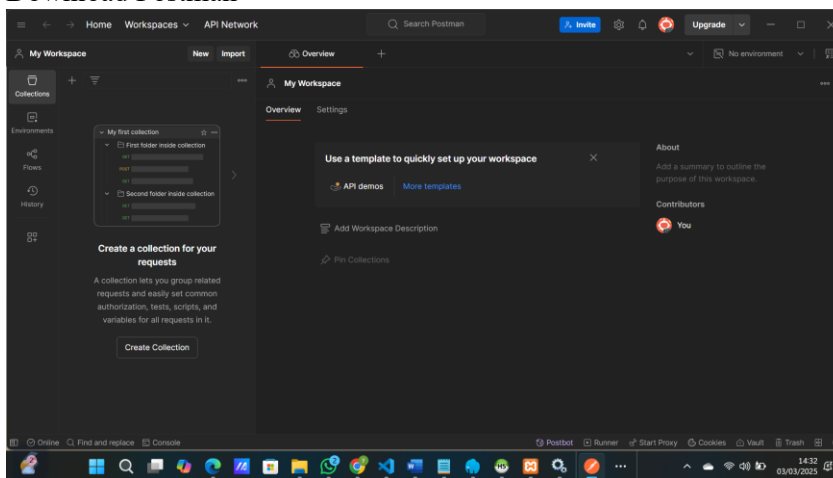
→ cd latihan12
→ composer run dev

New to Laravel? Check out our documentation. Build something amazing!

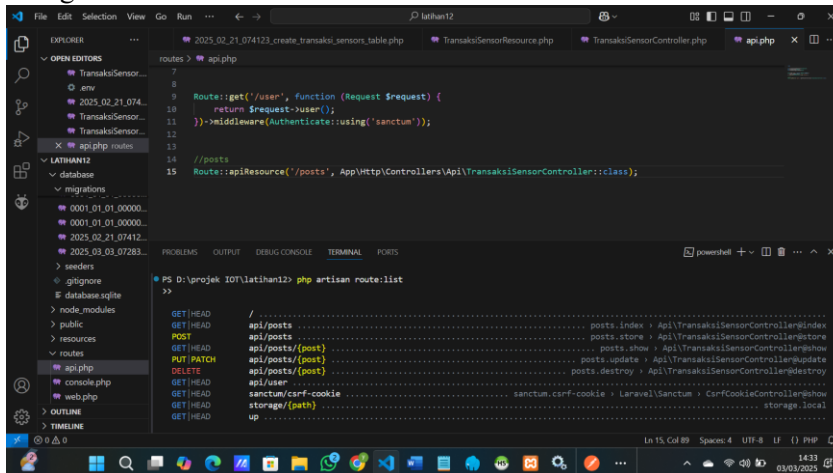
D:\projek IOT>
C:\Users\YASIVA>
```



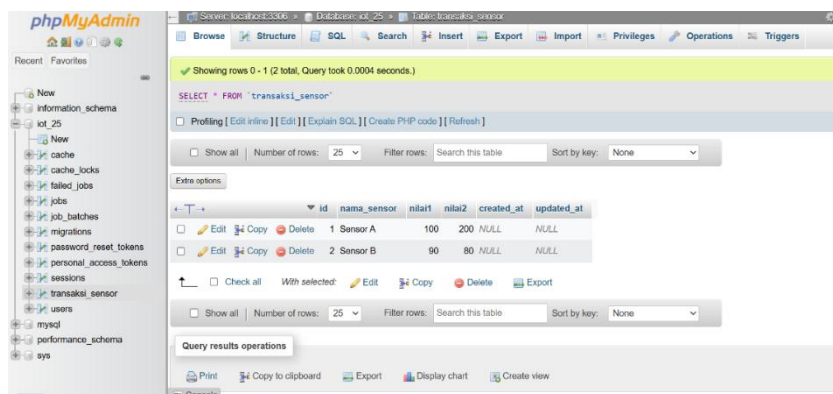
### 3. Download Postman



### 4. Mengecek lokasi route



5. Dua baris data pada tabel transaksi\_sensor pada database iot\_25



Showing rows 0 - 1 (2 total, Query took 0.0004 seconds.)

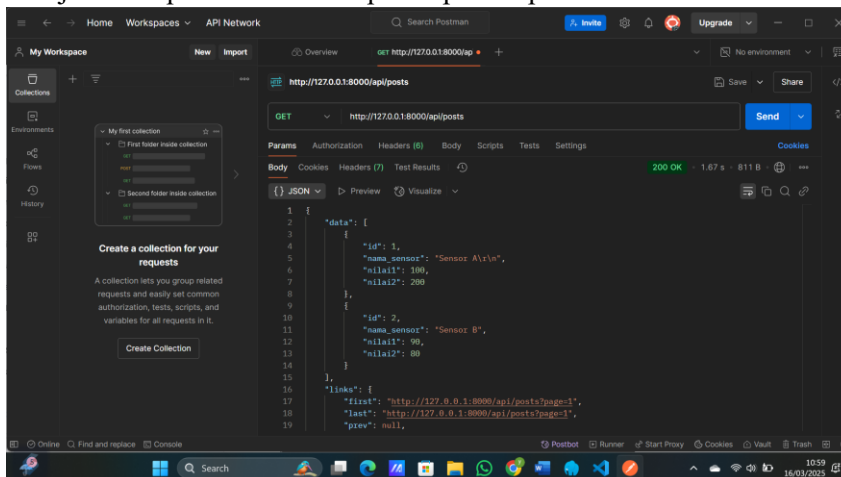
```
SELECT * FROM `transaksi_sensor`
```

Extra options

	id	nama_sensor	nilai1	nilai2	created_at	updated_at
<input type="checkbox"/>	1	Sensor A	100	200	NULL	NULL
<input type="checkbox"/>	2	Sensor B	90	80	NULL	NULL

Query results operations

6. Menjalankan prosedur berikut pada aplikasi postman

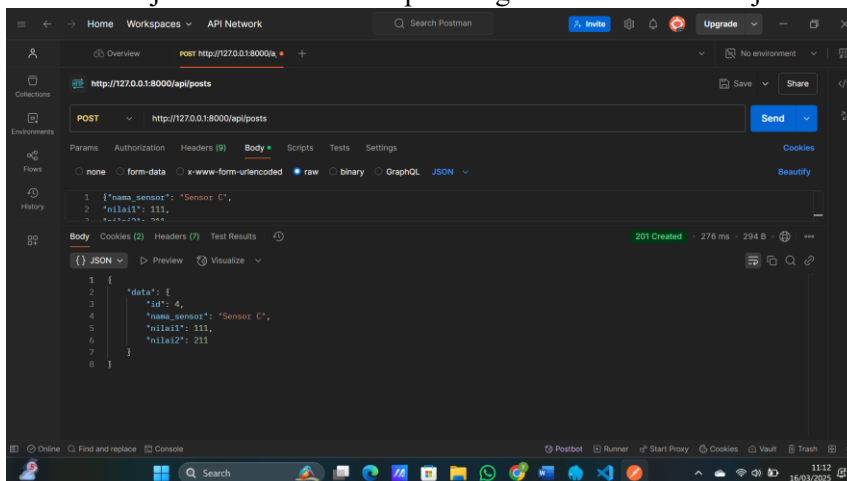


GET http://127.0.0.1:8000/api/posts

200 OK · 1.67 s · 811 B

```
{
  "data": [
    {
      "id": 1,
      "nama_sensor": "Sensor A (rin)",
      "nilai1": 100,
      "nilai2": 200
    },
    {
      "id": 2,
      "nama_sensor": "Sensor B",
      "nilai1": 90,
      "nilai2": 80
    }
  ],
  "links": {
    "first": "http://127.0.0.1:8000/api/posts?page=1",
    "last": "http://127.0.0.1:8000/api/posts?page=1",
    "prev": null,
    "next": null
  }
}
```

7. Melakukan percobaan insert data ke tabel di database menggunakan API, dengan mengganti method menjadi POST kemudian pada bagian header ubah menjadi sebagai berikut:



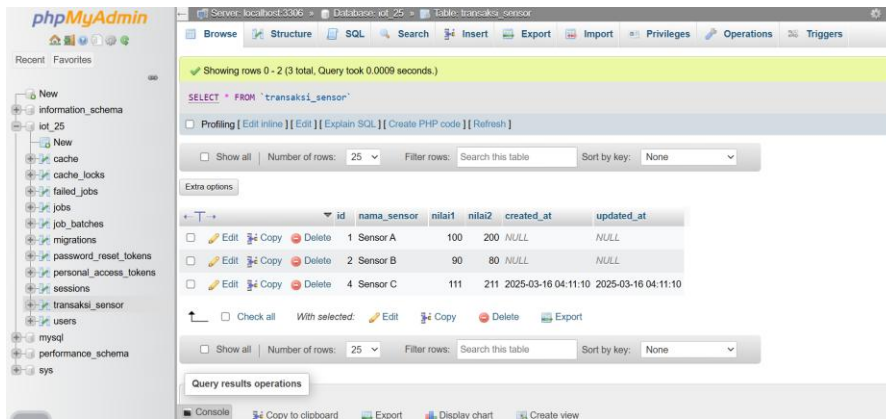
POST http://127.0.0.1:8000/api/posts

201 Created · 276 ms · 294 B

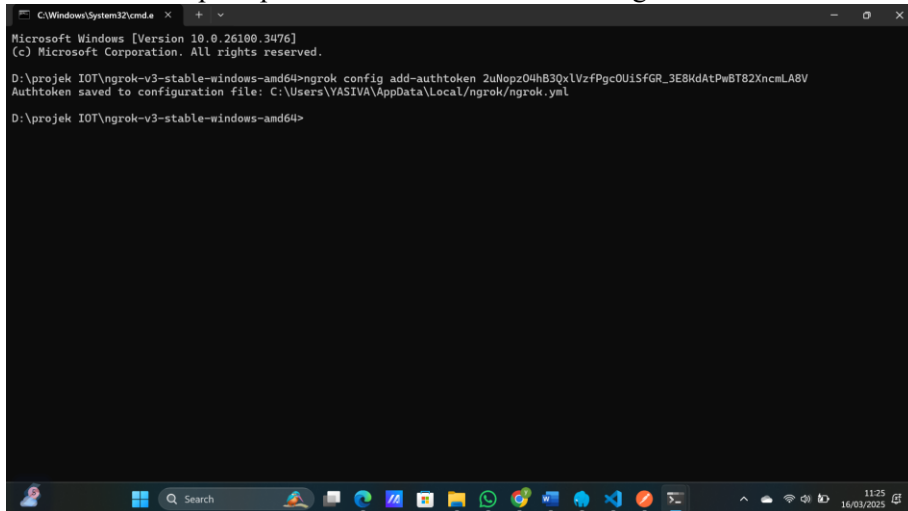
```
{
  "nama_sensor": "Sensor C",
  "nilai1": 111,
  "nilai2": 211
}
```

Body

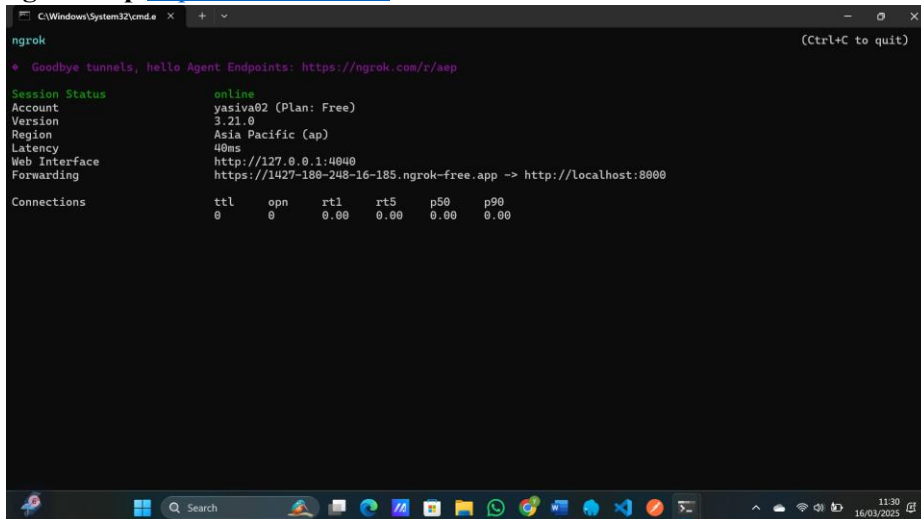
```
{
  "data": {
    "id": 4,
    "nama_sensor": "Sensor C",
    "nilai1": 111,
    "nilai2": 211
  }
}
```



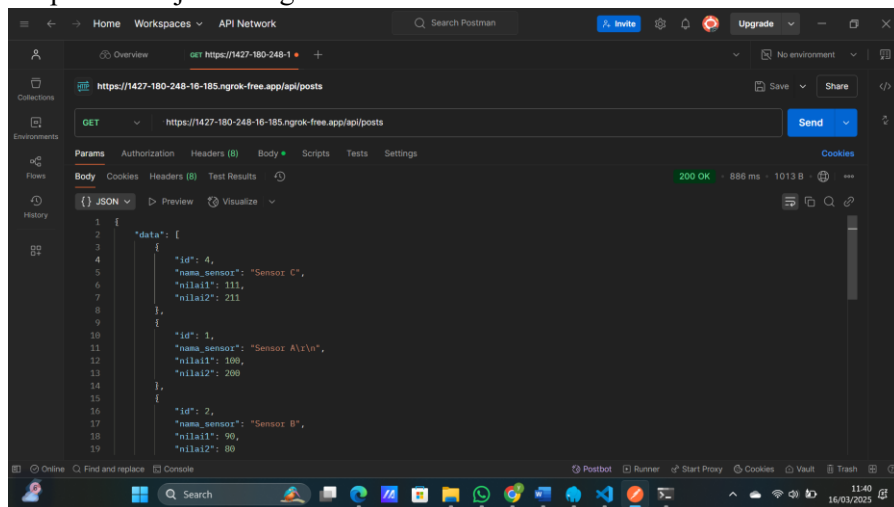
8. Buka command prompt dari alamat folder ekstraksi ngrok



9. Jalankan perintah berikut untuk mengonlinekan laravel melalui port 8000  
ngrok http <http://localhost:8000>



10. Untuk melakukan percobaan GET api , maka URL harus ditambahkan alamat endpoint menjadi sebagai berikut:



11. Ubah method menjadi POST dan parameter header dan body sesuaikan:

