LAPORAN PRAKTIKUM INTERNET OF THINGS (IoT)

FAKULTAS VOKASI, UNIVERSITAS BRAWIJAYA

PRAKTIK PEMBUATAN API

*Sholihul Fadjri Triwibowo*

*Fakultas Vokasi, Universitas Brawijaya*

*Email:* [*fadjritwibowo20@student.ub.ac.id*](mailto:fadjritwibowo20@student.ub.ac.id)

ABSTRACT

Internet of Things (IoT) memungkinkan pemantauan lingkungan secara real-time melalui integrasi sensor dengan sistem berbasis web. Dalam praktik ini, dilakukan implementasi sistem pemantauan suhu dan kelembapan menggunakan sensor DHT11 yang dikendalikan melalui ESP32. Data dari sensor dikirim ke server menggunakan API yang dikembangkan dengan Laravel 11 dan diakses secara publik melalui Ngrok. Tujuan dari penelitian ini adalah untuk mengembangkan sistem yang dapat mengirimkan dan menampilkan data lingkungan secara real-time pada platform berbasis web. Hasil praktik menunjukkan bahwa sistem dapat beroperasi dengan baik dalam mengirimkan data suhu dan kelembapan secara periodik ke server, serta memungkinkan pengguna untuk mengakses informasi tersebut melalui endpoint API. Penggunaan Laravel 11 sebagai backend API memberikan kemudahan dalam pengelolaan data, sedangkan Ngrok memungkinkan akses dari jaringan eksternal tanpa konfigurasi kompleks. Dengan demikian, sistem ini dapat diterapkan dalam berbagai skenario pemantauan lingkungan secara jarak jauh.

*Keywords — IoT, suhu, kelembapan, Laravel 11, Ngrok, ESP32, API*

1. Pendahuluan
   1. Latar Belakang

Perkembangan teknologi Internet of Things (IoT) semakin memberikan kemudahan dalam pemantauan kondisi lingkungan secara real-time. Salah satu aplikasi penting IoT adalah pemantauan suhu dan kelembapan, yang dapat diterapkan dalam berbagai bidang seperti pertanian, kesehatan, industri, dan sistem peringatan dini. Dengan adanya sistem pemantauan ini, pengguna dapat mengakses data lingkungan secara akurat dan mengambil tindakan yang diperlukan berdasarkan informasi yang diperoleh.

Dalam praktik ini, dirancang dan dikembangkan sistem pemantauan suhu dan kelembapan berbasis IoT menggunakan sensor DHT11 yang terhubung dengan mikrokontroler ESP32. Data yang dikumpulkan oleh sensor dikirimkan ke server melalui API yang dibangun menggunakan framework Laravel 11. Untuk memastikan aksesibilitas data dari jaringan eksternal tanpa konfigurasi jaringan yang kompleks, digunakan layanan Ngrok sebagai tunneling service.

Pemanfaatan Laravel 11 dalam pembuatan API memberikan kemudahan dalam pengelolaan data, sedangkan penggunaan Ngrok memungkinkan akses ke API dari berbagai perangkat tanpa perlu konfigurasi tambahan pada jaringan lokal. Dengan sistem ini, pengguna dapat memantau perubahan suhu dan kelembapan secara real-time melalui antarmuka berbasis web atau aplikasi pihak ketiga yang terhubung dengan API.

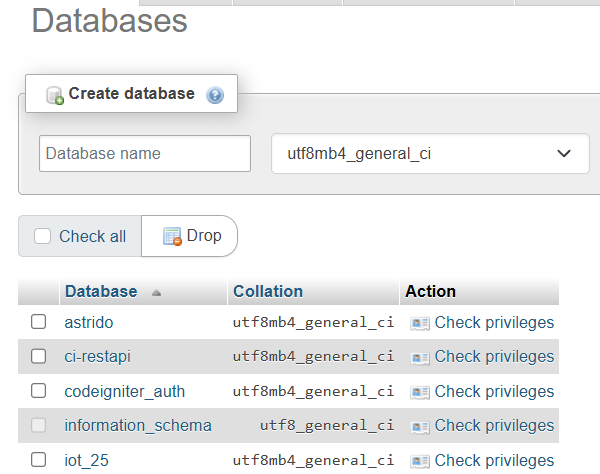
Diharapkan bahwa penelitian dan implementasi ini dapat memberikan wawasan tentang bagaimana mengintegrasikan IoT dengan teknologi web modern untuk menciptakan solusi pemantauan lingkungan yang efisien dan mudah diakses.

* 1. Tujuan Eksperimen

1. Mengembangkan sistem pemantauan suhu dan kelembapan
2. Membuat API menggunakan Laravel 11
3. Mengintegrasikan IoT dengan teknologi web
4. Metodologi
   1. Alat dan Bahan
      1. Alat: Visual Studio Code, Ngrok, XAMPP, PlatformIO, Laravel 11, Postman, Wokwi
      2. Bahan: ESP32, Sensor DHT11
   2. Langkah Implementasi
      1. Buat folder proyek Laravel menggunakan perintah

**composer create-project laravel/laravel Sensor**

* + 1. Buat database dengan nama iot\_25 di PHPMyAdmin



* + 1. Di dalam folder proyek Laravel, buat file model dan migration untuk TransaksiSensor dengan command **php artisan make:model TransaksiSensor -m**
    2. Ubah file 2025\_02\_25\_032031\_create\_transaksi\_sensors\_table.php

<?php

use Illuminate\Database\Migrations\Migration;

use Illuminate\Database\Schema\Blueprint;

use Illuminate\Support\Facades\Schema;

return new class extends Migration

{

    /\*\*

     \* Run the migrations.

     \*/

    public function up(): void

    {

        Schema::create('transaksi\_sensor', function (Blueprint $table) {

            $table->id('id')->startingValue(1); // Menetapkan AUTO\_INCREMENT dimulai dari 1

            $table->string('nama\_sensor', 255); // varchar(255)

            $table->integer('nilai1', false)->length(255); // int(255)

            $table->integer('nilai2', false)->length(255); // int(255)

            $table->timestamps(); // Menambahkan created\_at dan updated\_at

        });

    }

    /\*\*

     \* Reverse the migrations.

     \*/

    public function down(): void

    {

        Schema::dropIfExists('transaksi\_sensors');

    }

};

* + 1. Kemudian ubah juga isi file dari model TransaksiSensor.php

<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;

use Illuminate\Database\Eloquent\Model;

class TransaksiSensor extends Model

{

    use HasFactory;

    /\*\*

     \* The table associated with the model.

     \*

     \* @var string

     \*/

    protected $table = 'transaksi\_sensor';

    /\*\*

     \* The attributes that are mass assignable.

     \*

     \* @var array

     \*/

    protected $fillable = [

        'nama\_sensor',

        'nilai1',

        'nilai2',

    ];

    /\*\*

     \* The attributes that should be hidden for arrays.

     \*

     \* @var array

     \*/

    protected $hidden = [];

    /\*\*

     \* The attributes that should be cast.

     \*

     \* @var array

     \*/

    protected $casts = [];

}

* + 1. Kemudian jalankan perintah **php artisan migrate** untuk membuat tabel di database
    2. Buat resource dengan menjalankan perintah **php artisan make:resource TransaksiSensorResource** lalu ubah isi file

<?php

namespace App\Http\Resources;

use Illuminate\Http\Request;

use Illuminate\Http\Resources\Json\JsonResource;

class TransaksiSensorResource extends JsonResource

{

    /\*\*

     \* Transform the resource into an array.

     \*

     \* @param  \Illuminate\Http\Request  $request

     \* @return array

     \*/

    public function toArray($request)

    {

        return [

            'id' => $this->id,

            'nama\_sensor' => $this->nama\_sensor,

            'nilai1' => $this->nilai1,

            'nilai2' => $this->nilai2,

        ];

    }

}

* + 1. Buat API controller dengan menjalankan perintah **php artisan make:controller Api/TransaksiSensorController** dan ubah isinya dengan

<?php

namespace App\Http\Controllers\Api;

use Illuminate\Http\Request;

use App\Models\TransaksiSensor;

use App\Http\Controllers\Controller;

use App\Http\Resources\TransaksiSensorResource;

class TransaksiSensorController extends Controller

{

    /\*\*

     \* index

     \*

     \* @return \Illuminate\Http\Response

     \*/

    public function index()

    {

        // Get all transactions from TransaksiSensor model, paginated

        $transaksiSensors = TransaksiSensor::latest()->paginate(5);

        // Return a collection of transactions as a resource

        return TransaksiSensorResource::collection($transaksiSensors);

    }

    /\*\*

 \* Store a newly created resource in storage.

 \*

 \* @param  \Illuminate\Http\Request  $request

 \* @return \Illuminate\Http\Response

 \*/

public function store(Request $request)

{

    $validatedData = $request->validate([

        'nama\_sensor' => 'required|string|max:255',

        'nilai1' => 'required|integer',

        'nilai2' => 'required|integer',

    ]);

    $transaksiSensor = TransaksiSensor::create($validatedData);

    return new TransaksiSensorResource($transaksiSensor);

}

/\*\*

 \* Display the specified resource.

 \*

 \* @param  int  $id

 \* @return \Illuminate\Http\Response

 \*/

public function show($id)

{

    $transaksiSensor = TransaksiSensor::findOrFail($id);

    return new TransaksiSensorResource($transaksiSensor);

}

/\*\*

 \* Update the specified resource in storage.

 \*

 \* @param  \Illuminate\Http\Request  $request

 \* @param  int  $id

 \* @return \Illuminate\Http\Response

 \*/

public function update(Request $request, $id)

{

    $validatedData = $request->validate([

        'nama\_sensor' => 'required|string|max:255',

        'nilai1' => 'required|integer',

        'nilai2' => 'required|integer',

    ]);

    $transaksiSensor = TransaksiSensor::findOrFail($id);

    $transaksiSensor->update($validatedData);

    return new TransaksiSensorResource($transaksiSensor);

}

/\*\*

 \* Remove the specified resource from storage.

 \*

 \* @param  int  $id

 \* @return \Illuminate\Http\Response

 \*/

public function destroy($id)

{

    $transaksiSensor = TransaksiSensor::findOrFail($id);

    $transaksiSensor->delete();

    return response()->json(['message' => 'Deleted successfully'], 204);

}

}

* + 1. Buat route khusus API dengan menjalankan perintah **php artisan install:api** dan ubah isinya menjadi

<?php

use Illuminate\Auth\Middleware\Authenticate;

use Illuminate\Http\Request;

use Illuminate\Support\Facades\Route;

Route::get('/user', function (Request $request) {

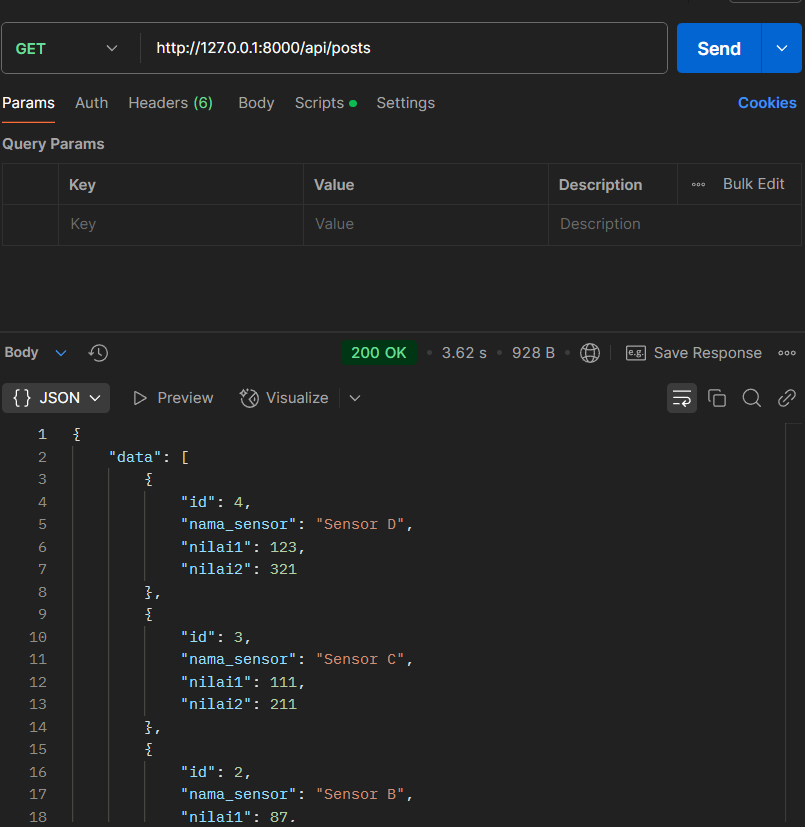
    return $request->user();

})->middleware(Authenticate::using('sanctum'));

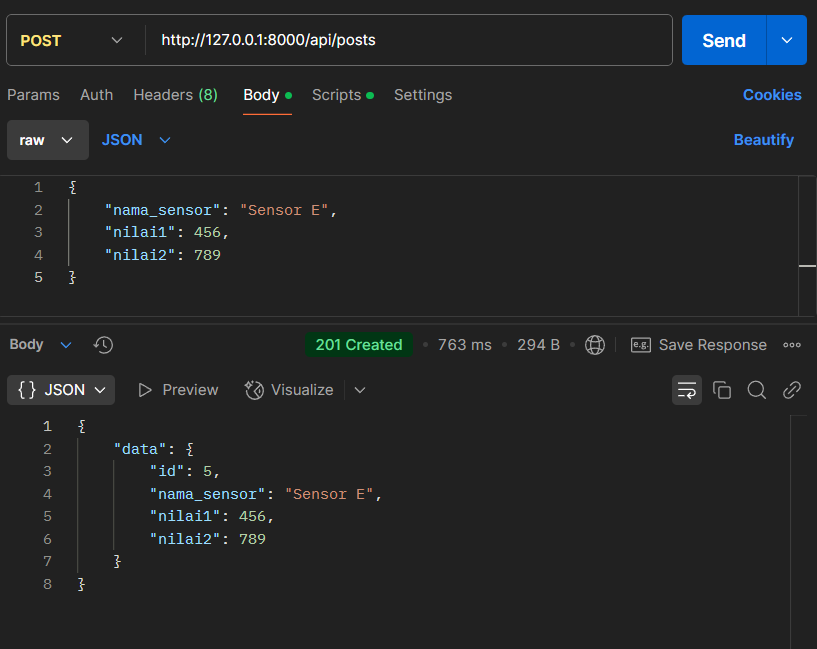
//posts

Route::apiResource('/posts', App\Http\Controllers\Api\TransaksiSensorController::class);

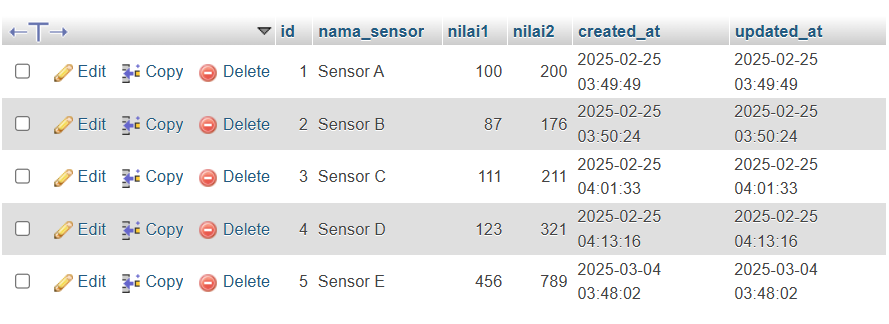
* + 1. Jalankan perintah **php artisan serve** dan lakukan percobaan dengan Postman, pilih method **GET** untuk mengambil data dari database, kemudian klik **SEND**

****

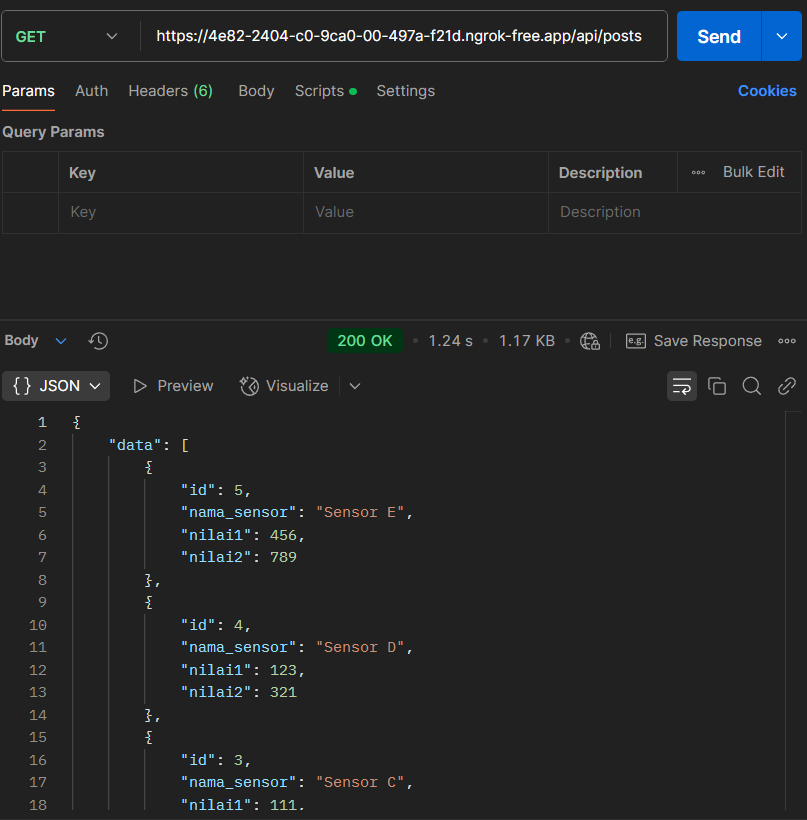
* + 1. Lakukan **POST** untuk menambah data ke database, dengan cara ubah bagian **body**

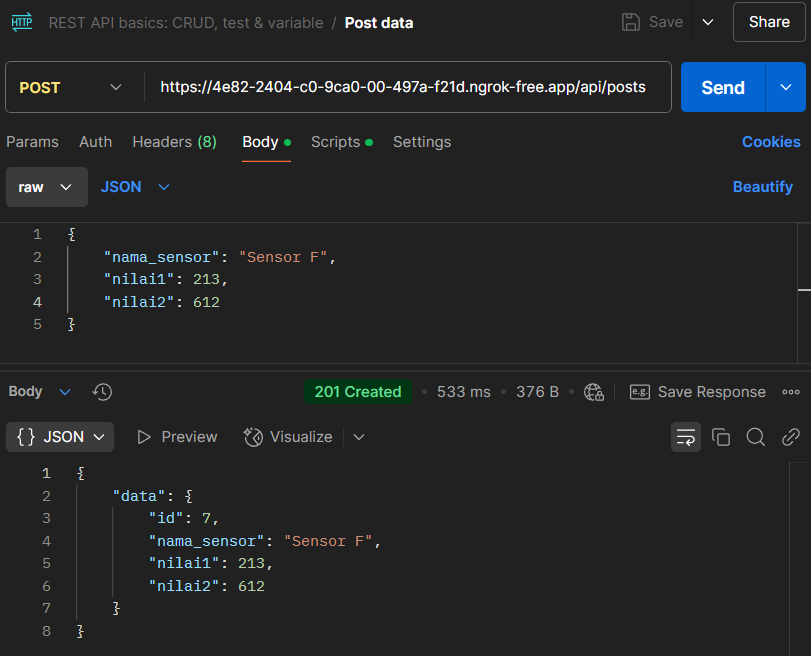


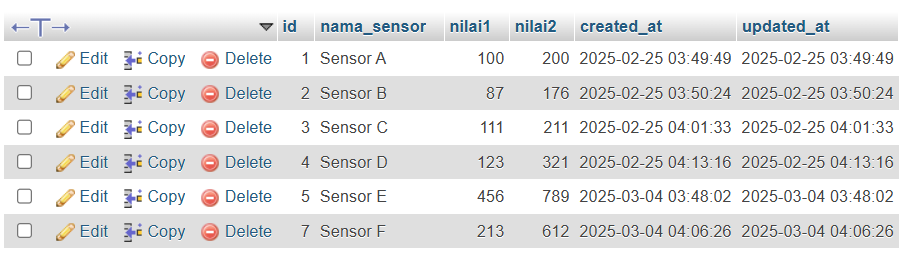
Lalu cek manual di PHPMyAdmin



* + 1. Buka ngrok lalu, jalankan perintah **ngrok http** [**http://localhost:8000**](http://localhost:8000)**,** lalu didapatkan link sebagai berikut [**https://4e82-2404-c0-9ca0-00-497a-f21d.ngrok-free.app**](https://4e82-2404-c0-9ca0-00-497a-f21d.ngrok-free.app)
    2. Di Postman, lakukan percobaan **GET** dan **POST** API dengan URL sebagai berikut [**https://4e82-2404-c0-9ca0-00-497a-f21d.ngrok-free.app/api/posts**](https://4e82-2404-c0-9ca0-00-497a-f21d.ngrok-free.app/api/posts)







1. Hasil dan Pembahasan
   1. Hasil Eksperimen

Eksperimen ini bertujuan untuk mengembangkan sistem pemantauan suhu dan kelembapan berbasis IoT dengan menggunakan sensor DHT11 yang dikendalikan oleh ESP32 virtual pada platform Wokwi. Data yang diperoleh dari sensor dikirimkan ke server melalui API yang dikembangkan menggunakan Laravel 11, dan akses eksternal difasilitasi dengan bantuan Ngrok. Proses implementasi dimulai dengan konfigurasi ESP32 virtual pada Wokwi, di mana sensor DHT11 dihubungkan dan diprogram menggunakan PlatformIO di Visual Studio Code. Mikrokontroler dikodekan untuk membaca data suhu dan kelembapan setiap interval tertentu, kemudian mengirimkan data tersebut ke endpoint API Laravel melalui metode HTTP POST.

Di sisi server, Laravel 11 digunakan untuk mengelola data yang dikirimkan dari ESP32. Endpoint API dikembangkan untuk menerima data, menyimpannya dalam database MySQL yang berjalan di XAMPP, dan menyediakan akses bagi pengguna untuk melihat data tersebut melalui permintaan HTTP GET. Untuk mengatasi keterbatasan akses lokal, Ngrok digunakan sebagai tunneling service agar API dapat diakses dari jaringan eksternal.

Data yang diterima API berhasil disimpan dalam tabel database MySQL tanpa adanya duplikasi atau kehilangan data. Struktur tabel yang digunakan mencakup kolom untuk waktu pengiriman, nilai suhu, dan nilai kelembapan. Endpoint API Laravel dapat diakses melalui browser atau Postman dengan menggunakan URL yang dihasilkan oleh Ngrok. Hasilnya menunjukkan bahwa data terbaru dapat ditampilkan secara real-time, memungkinkan pengguna untuk mengakses informasi lingkungan dari perangkat apa pun yang terhubung ke internet.

Dari hasil eksperimen ini, sistem pemantauan suhu dan kelembapan berbasis IoT yang dikembangkan dengan Laravel 11 dan Ngrok dapat bekerja dengan baik. Data dapat dikirim, disimpan, dan diakses secara real-time, memberikan gambaran kondisi lingkungan yang akurat. Implementasi ini menunjukkan bahwa kombinasi IoT dengan teknologi web modern dapat digunakan untuk berbagai aplikasi pemantauan jarak jauh dengan tingkat efektivitas yang tinggi.