LAPORAN PRAKTIKUM INTERNET OF THINGS (IoT)

Fakultas Vokasi, Universitas Brawijaya

# Praktik Pembuatan API Menggunakan Laravel 11 dan Ngrok

*Ricelvin Candra Hagana*

*Fakultas Vokasi, Universitas Brawijaya*

*Email:* [*ricelvincandrahagana@gmail.com*](mailto:ricelvincandrahagana@gmail.com)

## Abstract

Praktikum ini bertujuan untuk mengimplementasikan pembuatan Application Programming Interface (API) menggunakan framework Laravel 11 dan melakukan eksposisi API lokal ke jaringan publik melalui Ngrok, dengan memanfaatkan Visual Studio Code (VSCode) sebagai lingkungan pengembangan dan Wokwi sebagai simulator untuk menguji interaksi antara API dengan perangkat virtual.

Pada praktikum ini, mahasiswa merancang API sederhana untuk manajemen data (contoh: produk atau pengguna) dengan endpoint dasar (GET, POST, PUT, DELETE) dan validasi input. Kode program dikembangkan di VSCode dengan dukungan ekstensi Laravel untuk debugging dan optimasi. Simulator Wokwi digunakan untuk mensimulasikan permintaan (request) ke API dari lingkungan virtual, seperti IoT device atau aplikasi eksternal, guna menguji responsivitas dan kehandalan API. Ngrok diintegrasikan untuk mengekspos API lokal ke internet, memungkinkan pengujian end-to-end melalui aplikasi pihak ketiga atau alat seperti Postman.

Hasil praktikum menunjukkan keberhasilan pembuatan API dengan fungsionalitas CRUD (Create, Read, Update, Delete), respons JSON yang terstruktur, serta kemampuan Ngrok dalam menghubungkan lingkungan lokal ke jaringan eksternal. Simulasi di Wokwi membuktikan bahwa API dapat menerima dan memproses permintaan dari perangkat virtual dengan akurasi tinggi. Kombinasi Laravel 11, Ngrok, VSCode, dan Wokwi terbukti efektif sebagai sarana pembelajaran pengembangan API yang hemat biaya, fleksibel, dan minim risiko kesalahan konfigurasi infrastruktur fisik. Praktikum ini menjadi dasar bagi mahasiswa untuk memahami alur kerja pengembangan API modern, integrasi sistem, dan pengujian simulasi sebelum deployment ke lingkungan produksi.

*Keywords : API, Laravel 11, Ngrok, VScode, Wokwi Simulator*

## 1. Introduction (Pendahuluan)

1.1. Latar Belakang

Perkembangan teknologi berbasis web service dan microservices menjadikan Application Programming Interface (API) sebagai komponen krusial dalam integrasi sistem dan pertukaran data antar-aplikasi. API memungkinkan komunikasi antara frontend (antarmuka pengguna) dan backend (server/database), serta antar-platform yang berbeda. Pembelajaran pengembangan API modern memerlukan pemahaman framework seperti Laravel, yang menyediakan fitur lengkap untuk membangun API RESTful secara efisien, termasuk routing, middleware, dan manajemen database melalui Eloquent ORM. Namun, pengembangan dan pengujian API secara konvensional seringkali terkendala kebutuhan infrastruktur server fisik, risiko keamanan saat mengekspos jaringan lokal, serta kompleksitas konfigurasi lingkungan pengembangan.

Ngrok muncul sebagai solusi untuk mengatasi keterbatasan akses jaringan lokal dengan membuat tunnel yang mengubah alamat localhost menjadi URL publik yang aman. Alat ini memungkinkan pengujian API secara real-time oleh pengguna eksternal atau perangkat IoT tanpa perlu deployment ke server cloud. Di sisi lain, Visual Studio Code (VSCode) sebagai Integrated Development Environment (IDE) menyediakan ekosistem lengkap untuk pengembangan API dengan dukungan ekstensi seperti PHP Intelephense dan Laravel Artisan, mempercepat proses penulisan kode, debugging, dan pengujian.

Simulator Wokwi digunakan dalam praktikum ini untuk mereplikasi lingkungan uji coba API, seperti permintaan (request) dari perangkat IoT virtual atau aplikasi pihak ketiga, sehingga mahasiswa dapat memvalidasi respons API tanpa bergantung pada perangkat fisik. Kombinasi Laravel 11, Ngrok, VSCode, dan Wokwi menciptakan alur kerja yang terintegrasi: pembangunan API di lingkungan lokal, eksposisi ke jaringan publik via Ngrok, dan simulasi interaksi melalui Wokwi.

Praktikum ini dirancang untuk memberikan pengalaman langsung dalam pengembangan API berbasis Laravel 11, memanfaatkan tools simulasi dan eksposisi jaringan sebagai alternatif efisien untuk pembelajaran yang hemat biaya, fleksibel, dan minim risiko infrastruktur. Melalui kegiatan ini, mahasiswa diharapkan mampu memahami prinsip RESTful API, manajemen autentikasi, serta teknik pengujian end-to-end sebelum implementasi di lingkungan produksi.

1.2. Tujuan Eksperimen

* Memahami konsep dasar API (Application Programming Interface) dan penerapannya di Laravel 11.
* Menggunakan Ngrok untuk mengakses API secara publik melalui tunneling
* Menguji dan memastikan API berfungsi dengan baik menggunakan tools seperti Postman atau browser.

## 2. Methodology (Metodologi)

2.1. Tools & Materials (Alat dan Bahan)

* Laptop
* Koneksi Internet
* Aplikasi VScode

Xampp

* Laravel 11
* Ngrok
* Postman

2.2. Implementation Steps (Langkah Implementasi)

1. Buat Proyek Framework Laravel 11 menggunakan Herd
2. Buat database bernama iot\_25 di phpmyadmin
3. Buka proyek laravel 11 pada vs code dan edit file .env pada bagian DB\_CONNECTION seperti berikut:

DB\_CONNECTION=mysql

DB\_HOST=127.0.0.1

DB\_PORT=3306

DB\_DATABASE=iot\_25

DB\_USERNAME=root

DB\_PASSWORD=caberg2010

1. Buat file TransaksiSensor.php menggunakan perintah : php artisan make:model TransaksiSensor.php -m
2. Kemudian ubah file database\migrations\2025\_03\_08\_003518\_create\_transaksi\_sensors\_tab le.php seperti kode dibawah ini

<?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. Ubah isi file app/Models/TransaksiSensor.php dengan kode berikut

<?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.

\*

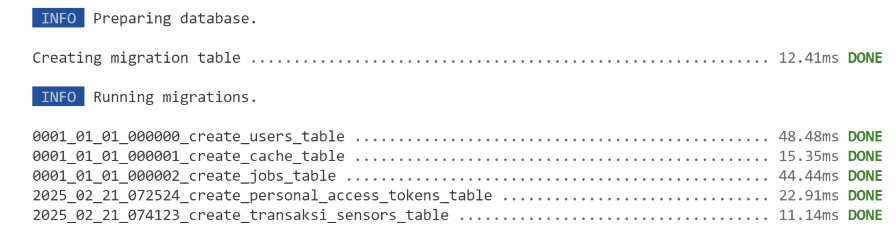
* + @var array

\*/ protected $casts = [];

}

1. Kemudian jalankan perintah php artisan migrate untuk membuat tabel pada database.

**php artisan migrate**



1. Buat Resource dengan menjalankan perintah php artisan make:resource

**TransaksiSensorResource**

1. Lalu ubah isi file app\Http\Resources\TransaksiSensorResource.php dengan kode dibawah

<?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 perintah **php artisan make:controller Api/TransaksiSensorController**

1. Lalu ubah isi file **app/Http/Controllers/Api/TransaksiSensorController.php dengan kode dibawah**

<?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 perintah **php artisan install:api** jika ada pilihan yes/no pilih “yes”

1. Ubah isi file routes/api.php dengan kode berikut

<?php

use Illuminate\Auth\Middleware\Authenticate; use Illuminate\Http\Request; use Illuminate\Support\Facades\Route;

Route::get('/user', function (Request $request) { return $request->user();

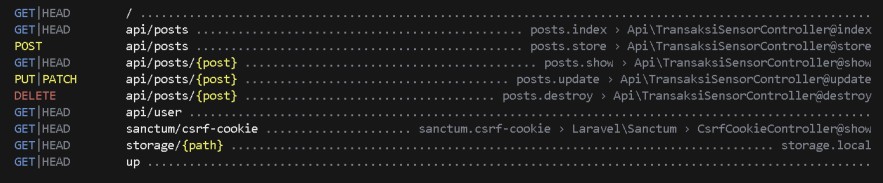
})->middleware(Authenticate::using('sanctum'));

//posts

Route::apiResource('/posts',

App\Http\Controllers\Api\TransaksiSensorController::class);

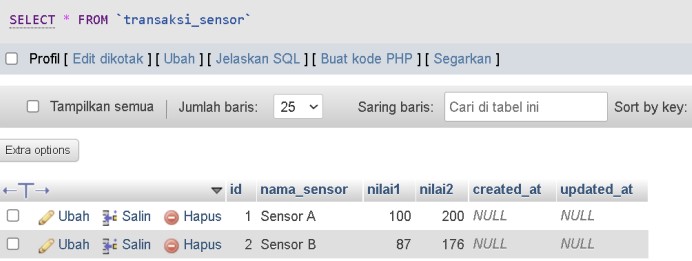
1. Pastikan route sudah terbentuk dengan memasukkan perintah php artisan route:list, dan pastikan tampilan pada terminal seperti gambar dibawah ini



1. Testing dengan menggunakan tools postman. Jika belum punya, download postman dan lakukan instalasi. Untuk mencoba mengakses API, pastikan laravel berjalan dengan perintah **php artisan serve**

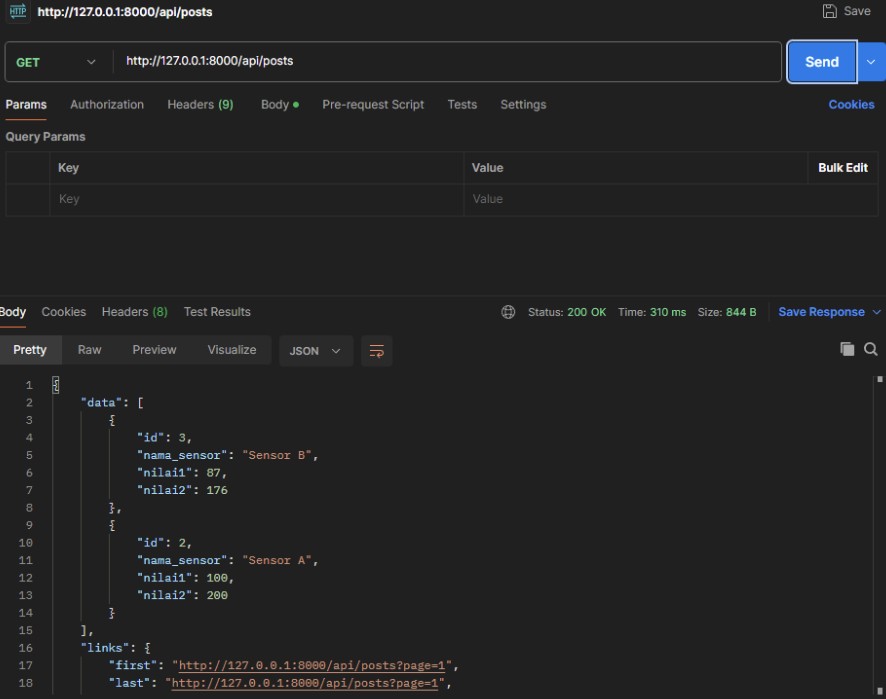


1. Lalu pastikan ada data pada tabel transaksi\_sensor, misalnya



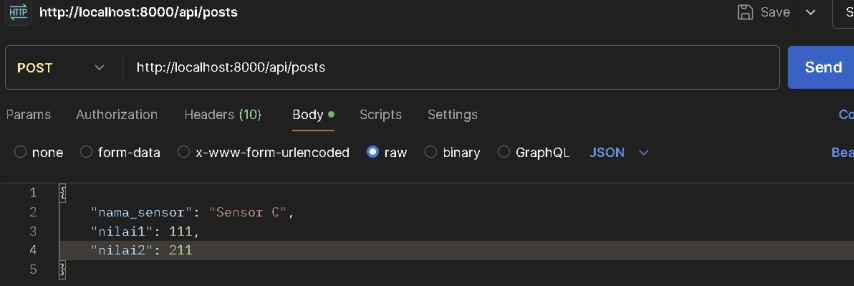
1. Pada bagan url, masukkan alamat server laravel dan route API-nya :

http://localhost:8000/api/posts , lalu pilih method GET dan klik Send seperti gambar dibawah

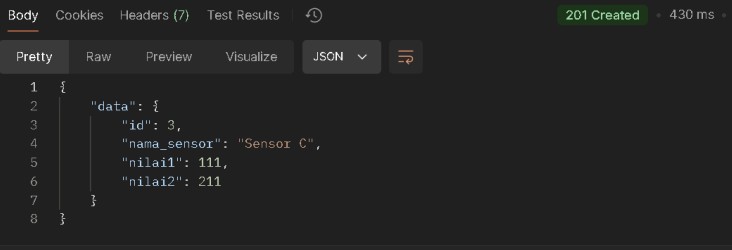


1. Lakukan percobaan insert data ke database dengan mengganti method menjadi POST lalu pastikan Content-Type pada Headers adalah application/json

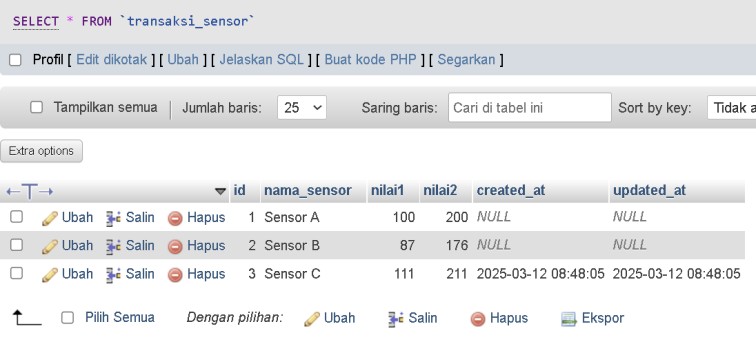
1. kemudian pada Body gunakan raw dan format json, lalu isi seperti berikut



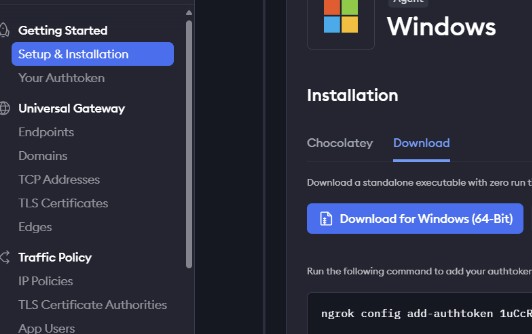
1. Kemudian klik send, Jika berhasil maka hasilnya akan seperti ini



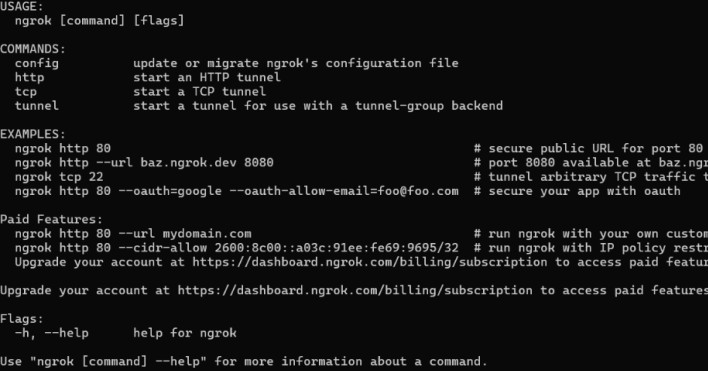
1. Jangan lupa cek di phpmyadmin untuk memastikan datanya benar-benar bertambah



1. Langkah berikutnya adalah mengonline-kan API menggunakan service ngrok sehingga API dapat diakses melalui device iot atau simulasi wokwi iot (secara Internet)



1. ekstrak file zip yang sudah di download dan klik 2x file ngrok.exe untuk membuka cmd.



1. Kemudian jalankan perintah sesuai yang ada di akun ngrok
2. Jalankan perintah berikut untuk mengonline-kan laravel melalui port

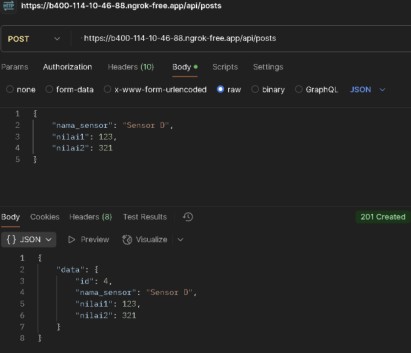
8000

**ngrok http** [**http://localhost:8000**](http://localhost:8000/)

1. Lakukan percobaan untuk mengambil data dari database menggunakan link yang telah disalin, gunakan method GET dan tambahkan route

/api/posts.

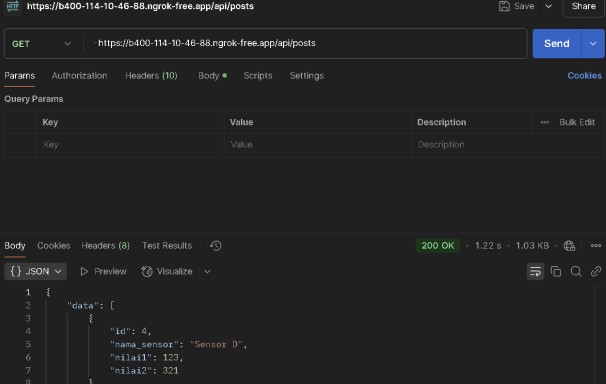
1. Berikutnya adalah melakukan perobaan untuk menambahkan data pada database, gunakan method POST, lalu pada Body, pilih raw dan format json



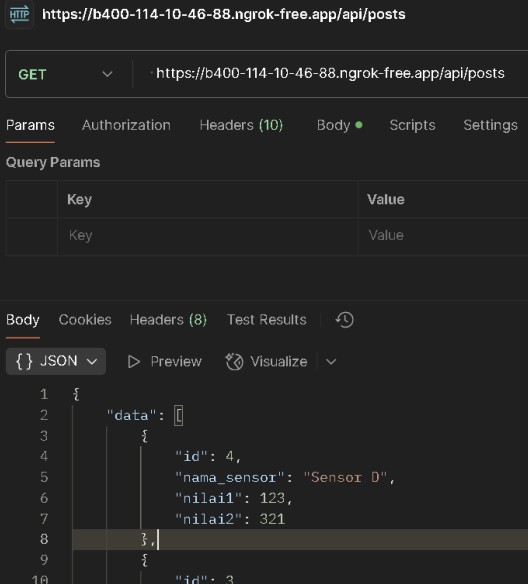
## 3. Results and DIscussion (Hasil dan Pembahasan)

3.1. Experimenetal Results (Hasil Eksperimen)

mengambil data dari database menggunakan API



menambahkan data ke database menggunakan API & Ngrok:



## 4. Appendix (Lampiran

terminal dan cmd ketika API & Ngrok berhasil mengambil dan menambahkan data pada database

