**LAPORAN PRAKTIKUM INTERNET OF THINGS (IoT)**

**Pembuatan API Menggunakan Laravel 11 dan Ngrok**

*Author : Nelly Fananda Melani*

*Fakultas Vokasi, Universitas Brawijaya*

*Email:* [*nellyfananda15@student.ub.ac.id*](mailto:nellyfananda15@student.ub.ac.id)

**Abstract**

This practicum aims to develop and implement an API using Laravel 11 and integrate it with the Ngrok service for public access. The development is carried out in the Visual Studio Code (VSCode) environment, utilizing Postman for API testing, XAMPP as the database server, and the Command Prompt (CMD) terminal as a command execution tool. The API development process includes initial Laravel configuration, designing models, controllers, and routes, as well as testing endpoints to ensure optimal data communication. By leveraging Ngrok, the API can be accessed remotely without requiring additional server configuration, making it highly practical for development and testing purposes. Furthermore, this practicum explores authentication and data management techniques to enhance security and efficiency in API interactions. The integration of Laravel’s built-in features, such as middleware and request validation, ensures structured and reliable data processing. The results of this practicum demonstrate that the Laravel 11-based API functions effectively, enabling efficient and flexible data exchange. Additionally, this implementation provides a deeper understanding of modern web service development, particularly in the context of IoT-based applications and distributed systems. This project highlights the importance of API connectivity in enabling seamless communication between devices and services, making it a valuable skill for future advancements in IoT technology.

**Keywords**— Laravel 11, API, Ngrok, Postman, XAMPP, Internet of Things (IoT).

**1. Introduction (Pendahuluan)**

Perkembangan teknologi berbasis Internet of Things (IoT) dan aplikasi berbasis web telah mendorong kebutuhan akan layanan API (Application Programming Interface) yang efisien dan mudah diakses. API memungkinkan pertukaran data antara berbagai sistem dan perangkat, sehingga menjadi komponen penting dalam pengembangan aplikasi modern, termasuk dalam sistem IoT, otomasi, dan layanan berbasis cloud.

Laravel adalah salah satu framework PHP yang populer untuk pengembangan web, termasuk dalam pembuatan API. Versi terbaru, Laravel 11, menawarkan berbagai fitur yang mempermudah pembuatan API, seperti dukungan bawaan untuk routing, middleware, autentikasi, serta manajemen database yang lebih efisien. Namun, dalam tahap pengembangan API, salah satu tantangan yang sering dihadapi adalah pengujian dan akses API dari perangkat lain tanpa harus melakukan deployment ke server langsung.

Untuk mengatasi kendala tersebut, digunakan Ngrok, sebuah alat yang memungkinkan pengembang membuka akses API lokal ke internet dengan mudah. Dengan Ngrok, API yang dikembangkan di lingkungan lokal dapat diuji secara remote tanpa perlu konfigurasi server tambahan. Selain itu, pengujian API dilakukan menggunakan Postman, sebuah aplikasi yang membantu dalam mengirim permintaan HTTP dan menganalisis respons dari API.

Dalam praktikum ini, dilakukan proses pembuatan API menggunakan Laravel 11 di lingkungan Visual Studio Code (VSCode), dengan XAMPP sebagai server database dan Command Prompt (CMD) sebagai alat eksekusi perintah. Praktikum ini mencakup konfigurasi awal Laravel, pembuatan model, controller, dan route, serta pengujian endpoint API menggunakan Postman. Ngrok digunakan untuk membuat API dapat diakses secara online tanpa memerlukan deployment ke server eksternal.

Tujuan dari praktikum ini adalah untuk memberikan pemahaman dasar tentang pengembangan API menggunakan Laravel 11, integrasi dengan Ngrok, serta metode pengujian menggunakan Postman. Dengan pemahaman ini, diharapkan pengembang dapat membangun API yang efisien, aman, dan dapat diakses dari berbagai perangkat, sehingga mendukung pengembangan sistem berbasis IoT, aplikasi web, dan teknologi berbasis cloud.

**1.1 Background of the IoT Practicum (Latar Belakang Praktikum IoT)**

Dalam era transformasi digital, Application Programming Interface (API) memainkan peran penting dalam memungkinkan komunikasi antara berbagai aplikasi dan sistem. API menjadi komponen utama dalam pengembangan aplikasi web dan mobile modern, memungkinkan pertukaran data yang efisien serta integrasi antar platform. Secara khusus, RESTful API banyak digunakan dalam aplikasi Internet of Things (IoT), sistem otomatisasi, serta layanan berbasis cloud yang membutuhkan interaksi data secara real-time.

Salah satu framework yang populer untuk pengembangan API adalah Laravel, sebuah framework berbasis PHP yang dikenal karena efisiensi, keamanan, dan kemudahan penggunaannya. Versi terbaru, Laravel 11, menghadirkan fitur yang lebih baik untuk pengembangan API, seperti autentikasi bawaan, sistem routing yang kuat, serta dukungan middleware yang lebih optimal. Namun, dalam tahap pengembangan, pengujian API secara lokal sering kali menjadi tantangan, terutama ketika API perlu diakses secara remote untuk integrasi dengan sistem eksternal.

Untuk mengatasi tantangan ini, digunakan Ngrok, sebuah layanan tunneling yang memungkinkan API lokal diakses secara publik melalui internet dengan aman. Dengan integrasi Ngrok, pengembang dapat mengekspos API Laravel lokal tanpa perlu melakukan deployment ke server khusus, sehingga pengujian dan debugging dapat dilakukan dengan lebih efisien. Selain itu, Postman menjadi alat penting dalam pengembangan API, memungkinkan pengguna untuk mengirim permintaan HTTP, menguji endpoint, serta menganalisis respons API secara interaktif.

Praktikum ini berfokus pada pembuatan dan implementasi API menggunakan Laravel 11 di lingkungan pengembangan lokal dengan Visual Studio Code (VSCode), menggunakan XAMPP sebagai server database, serta Command Prompt (CMD) sebagai alat eksekusi perintah. Praktikum ini mencakup aspek fundamental dalam pengembangan API, seperti konfigurasi Laravel, pembuatan model, controller, dan route, serta pengujian endpoint menggunakan Postman. Dengan memanfaatkan Ngrok, API yang dibuat dapat diakses secara remote untuk keperluan pengujian dan integrasi lebih lanjut.

Melalui praktikum ini, peserta diharapkan dapat memperoleh pengalaman langsung dalam pengembangan API, autentikasi, manajemen database, dan akses remote menggunakan Ngrok. Pemahaman ini menjadi dasar bagi implementasi API yang lebih kompleks, termasuk yang digunakan dalam aplikasi IoT, komputasi awan, serta sistem berbasis web. Dengan memahami cara kerja API dan strategi integrasinya, pengembang akan lebih siap dalam membangun sistem yang skalabel dan aman untuk berbagai kebutuhan teknologi di masa depan.

**1.2 Experimental Objectives (Tujuan Eksperimen)**

Eksperimen ini bertujuan untuk:

1. Membangun dan mengimplementasikan API menggunakan Laravel 11 sebagai framework backend yang efisien dan modern.
2. Memahami konsep dasar API RESTful, termasuk metode HTTP (GET, POST, PUT, DELETE) dan cara kerja komunikasi antara klien dan server.
3. Mengkonfigurasi Laravel 11 untuk mendukung pengembangan API, termasuk pembuatan model, controller, route, dan middleware untuk mengelola data secara efektif.
4. Menguji API menggunakan Postman, memastikan setiap endpoint berfungsi dengan baik serta dapat mengolah permintaan dan respons secara optimal.
5. Mengintegrasikan API dengan Ngrok untuk memungkinkan akses publik terhadap API yang dikembangkan di lingkungan lokal tanpa perlu deployment ke server eksternal.
6. Memahami cara kerja autentikasi dan keamanan API, termasuk implementasi token-based authentication menggunakan Laravel Sanctum atau Passport.
7. Menganalisis manfaat API dalam pengembangan sistem berbasis IoT, cloud computing, dan aplikasi terdistribusi, serta bagaimana API dapat digunakan untuk pertukaran data antar perangkat dan layanan.

**2. Methodology (Metodologi)**

Eksperimen ini dilakukan dengan membangun dan menguji API berbasis Laravel 11 serta menghubungkannya dengan layanan Ngrok untuk akses publik. Langkah-langkah dalam metodologi ini meliputi:

1. Instalasi dan Konfigurasi Lingkungan Pengembangan: Instal XAMPP sebagai server database, instal Laravel 11 menggunakan Composer di VSCode, dan konfigurasi database MySQL di file .env.
2. Pembuatan API Menggunakan Laravel 11: Buat Model, Controller, dan Route untuk API, implementasi CRUD (Create, Read, Update, Delete), serta gunakan Laravel Migration dan Seeder untuk database.
3. Pengujian API Menggunakan Postman: Kirim permintaan GET, POST, PUT, DELETE, serta periksa respons API (status kode HTTP, validasi input, JSON output).
4. Integrasi Ngrok untuk Akses Publik: Jalankan Ngrok untuk membuka akses API online dan uji URL publik dari Ngrok melalui Postman atau browser.
5. Analisis dan Evaluasi API: Evaluasi kinerja API dalam menangani permintaan, identifikasi masalah keamanan, efisiensi kode, optimasi query, serta analisis manfaat Laravel 11 dan Ngrok dalam pengembangan API.

**2.1 Tools & Materials (Alat dan Bahan)**

Eksperimen ini dilakukan secara virtual tanpa memerlukan perangkat keras fisik. Alat dan bahan yang digunakan dalam proyek ini meliputi:

1. Laptop/PC dengan Visual Studio Code sebagai lingkungan pengembangan.
2. XAMPP sebagai server database MySQL.
3. Laravel 11 sebagai framework backend untuk membangun API.
4. Composer untuk mengelola dependensi Laravel.
5. Postman untuk menguji endpoint API.
6. Ngrok untuk membuat API dapat diakses secara publik.
7. Terminal CMD atau Git Bash untuk menjalankan perintah Laravel.
8. Koneksi internet untuk mengakses dan menjalankan Laravel serta Ngrok.

**2.2 Implementation Steps (Langkah Implementasi)**

Eksperimen ini dilakukan melalui beberapa langkah utama:

**1. Instalasi Ngrok**

Install Ngrok

<https://ngrok.com/downloads/windows?tab=download>

Buka PowerShell dengan Administrator

lalu ketikkan kode berikut untuk masuk ke folder

cd C:\ProgramData\

Jalankan Perintah Instalasi Chocolatey

Set-ExecutionPolicy Bypass -Scope Process -Force; [System.Net.ServicePointManager]::SecurityProtocol = [System.Net.ServicePointManager]::SecurityProtocol -bor 3072; iex ((New-Object System.Net.WebClient).DownloadString('https://community.chocolatey.org/install.ps1'))

Cek Chocolatey

choco -v

**2. Instalasi Laravel**

Buka Terminal atau Command Prompt dan jalankan perintah berikut untuk menginstal Laravel secara global:

composer global require laravel/installer

Buat proyek Laravel baru dengan perintah:

composer create-project --prefer-dist laravel/laravel LARAVEL-11

Masuk ke folder proyek Laravel yang baru dibuat:

cd C:\xampp\htdocs\

cd LARAVEL-11

Jalankan Laravel untuk memastikan instalasi berhasil:

php artisan serve

Jika berhasil, kita akan melihat output seperti ini:

Server running on [http://127.0.0.1:8000]

**3. Konfigurasi Database**

Jalankan XAMPP dan aktifkan Apache serta MySQL.

Buka phpMyAdmin di browser <http://localhost/phpmyadmin>

Buat database di phpmyadmin dengan nama **iot\_25**

Sebuah gambar berisi teks, cuplikan layar, Font, garis

Konten yang dihasilkan AI mungkin salah.

Buka file .env di proyek Laravel (VScode) dan ubah konfigurasi database:

DB\_CONNECTION=mysql

DB\_HOST=127.0.0.1

DB\_PORT=3306

DB\_DATABASE=iot\_25

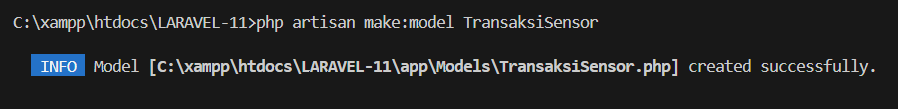
DB\_USERNAME=root

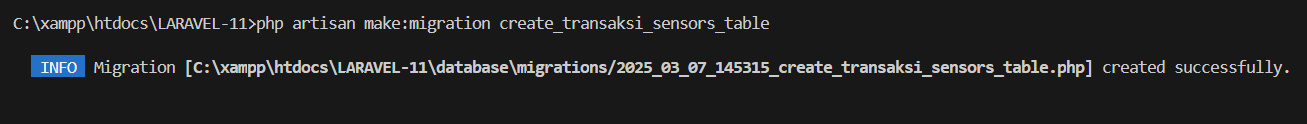
DB\_PASSWORD=

**MULAI PROJECT**

Buat file model  **TransaksiSensor.php** dengan cara menjalankan perintah :

**php artisan make:model TransaksiSensor**





Kemudian ubah file **2025\_02\_21\_074123\_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');

}

};

Kemudian ubah isi file **app/Models/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 = [];

}

**Kemudian jalankan perintah berikut untuk membuat tabel :**

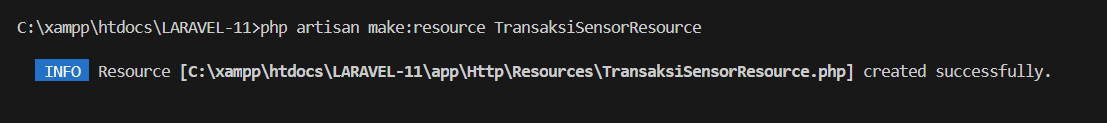
**php artisan migrate**

**Sebuah gambar berisi teks, software, Software multimedia, cuplikan layar

Konten yang dihasilkan AI mungkin salah.**

**Buat Resource dengan menjalankan perintah :**

**php artisan make:resource TransaksiSensorResource**

****

**Sebuah gambar berisi teks, cuplikan layar, software, Software multimedia

Konten yang dihasilkan AI mungkin salah.**

**Ubah isi file TransaksiSensorResource.php dengan isi file berikut :**

<?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,

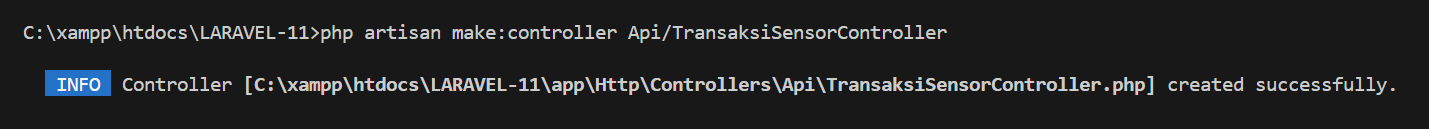
];

}

}

**Buat API controller dengan menjalankan perintah :**

**php artisan make:controller Api/TransaksiSensorController**

****

**Ubah isi file app/Http/Controllers/Api/TransaksiSensorController.php**

<?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);

}

}

**Sebuah gambar berisi teks, cuplikan layar, software, Software multimedia

Konten yang dihasilkan AI mungkin salah.**

**Buat route khusus API dengan menjalankan perintah :**

**php artisan install:api**

**Sebuah gambar berisi teks, cuplikan layar, software, Software multimedia

Konten yang dihasilkan AI mungkin salah.**

**Buka file routes/api.php  dan ubah isi file 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);

Sebuah gambar berisi teks, software, Software multimedia, Perangkat lunak grafia

Konten yang dihasilkan AI mungkin salah.

**Kemudian pastikan routes telah terbentuk dengan menjalankan perintah :**

**php artisan route:list**

pastikan tampilan sebagai berikut :

**Sebuah gambar berisi teks, cuplikan layar, software, Software multimedia

Konten yang dihasilkan AI mungkin salah.**

**Untuk melakukan testing, gunakan tools postman dengan langkah sebagai berikut :**

Download aplikasi postman pada link berikut :<https://www.postman.com/downloads/>

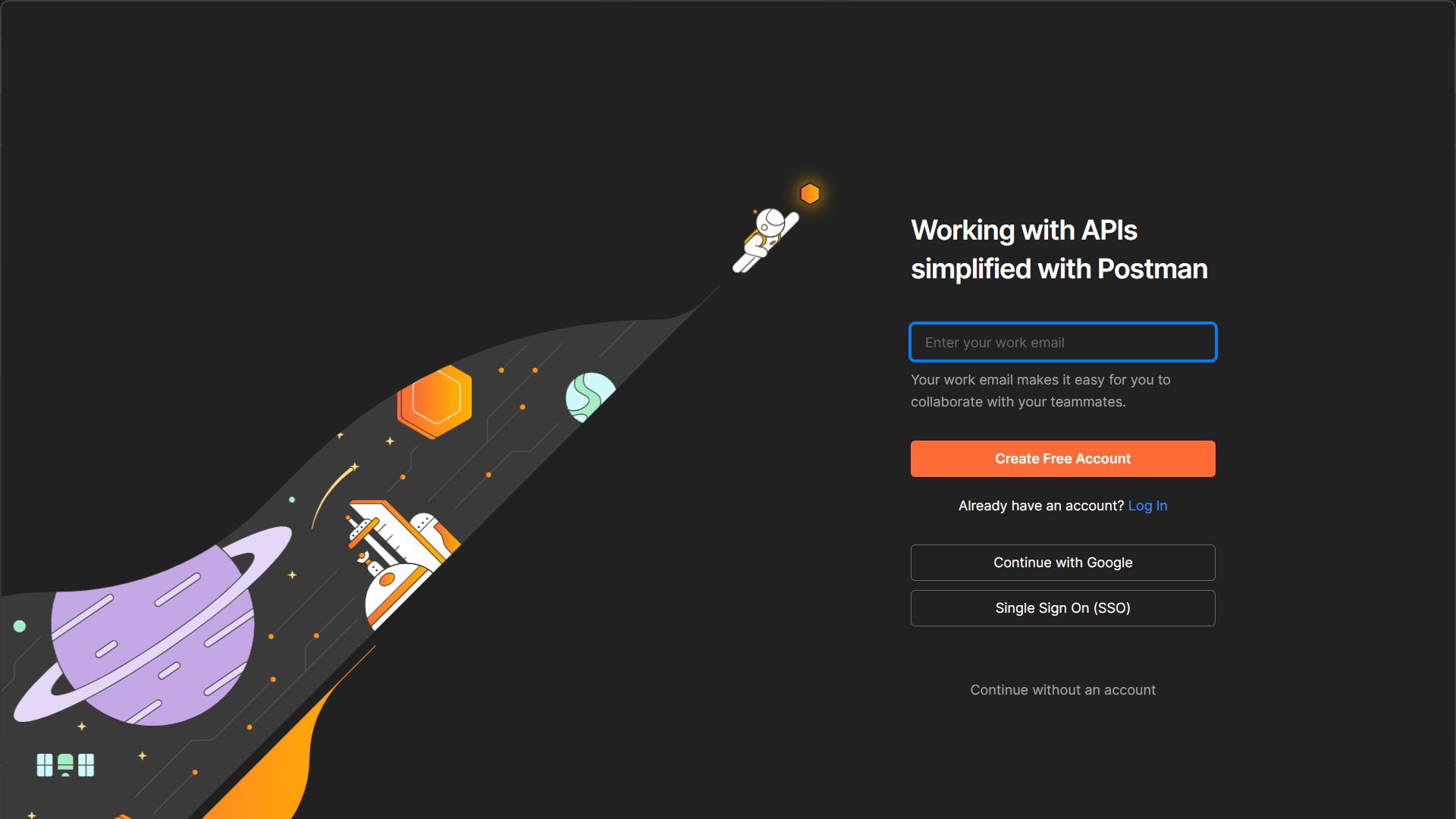
**Sebuah gambar berisi teks, cuplikan layar, software, Ikon komputer

Konten yang dihasilkan AI mungkin salah.**

**Lakukan prosedur instalasi dan jalankan aplikasi postman.**

**Sebuah gambar berisi teks, cuplikan layar, software, Software multimedia

Konten yang dihasilkan AI mungkin salah.**

****

**Sebuah gambar berisi teks, cuplikan layar, software, Software multimedia

Konten yang dihasilkan AI mungkin salah.**

**Untuk melakukan percobaan akses api, pastikan aplikasi laravel dijalankan dengan perintah :**

**php artisan serve**

**Sebuah gambar berisi teks, cuplikan layar, Font

Konten yang dihasilkan AI mungkin salah.**

Pastikan telah data yang dimasukkan kedalam tabel di database. Pada contoh berikut, telah ada 2 baris data pada **tabel transaksi\_sensor** pada **database iot\_25**

**Buat Tabel transaksi\_sensor**

CREATE TABLE IF NOT EXISTS transaksi\_sensor (

id INT AUTO\_INCREMENT PRIMARY KEY,

nama\_sensor VARCHAR(50) NOT NULL,

nilai1 INT NOT NULL,

nilai2 INT NOT NULL,

created\_at DATETIME DEFAULT CURRENT\_TIMESTAMP,

updated\_at DATETIME DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP

);

**Tambahkan data**

INSERT INTO transaksi\_sensor (id, nama\_sensor, nilai1, nilai2, created\_at, updated\_at)

VALUES

(2, 'Sensor A', 100, 200, '2025-02-21 15:21:12', '2025-02-21 15:21:12'),

(3, 'Sensor B', 87, 176, '2025-02-22 23:44:24', '2025-02-22 23:44:24');

**Sebuah gambar berisi teks, cuplikan layar, software, Ikon komputer

Konten yang dihasilkan AI mungkin salah.**

Untuk mengambil data diatas melalui aplikasi postman, jalankan prosedur berikut :

**Sebuah gambar berisi teks, cuplikan layar, software, Software multimedia

Konten yang dihasilkan AI mungkin salah.**

Pada bagian URL masukkan alamat server laravel <http://127.0.0.1:8000/api/posts>

Atau bisa diakses melalui url : <http://localhost:8000/api/posts>

Pilih method **GET** untuk mengambil data dari database , kemudian klik tombol **SEND**

Pastikan data dikembalikan dalam bentuk json seperti tampilan contoh diatas

{

"data": [

{

"id": 3,

"nama\_sensor": "Sensor B",

"nilai1": 87,

"nilai2": 176

},

{

"id": 2,

"nama\_sensor": "Sensor A",

"nilai1": 100,

"nilai2": 200

}

],

"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

},

"meta": {

"current\_page": 1,

"from": 1,

"last\_page": 1,

"links": [

{

"url": null,

"label": "&laquo; Previous",

"active": false

},

{

"url": "http://127.0.0.1:8000/api/posts?page=1",

"label": "1",

"active": true

},

{

"url": null,

"label": "Next &raquo;",

"active": false

}

],

"path": "http://127.0.0.1:8000/api/posts",

"per\_page": 5,

"to": 2,

"total": 2

}

}

**Info!**

Jika ada error seperti dibawah ini

**Sebuah gambar berisi teks, cuplikan layar, software, Software multimedia

Konten yang dihasilkan AI mungkin salah.**

Coba ikuti langkah berikut

Masuk ke terminal Vscode dan ketikkan perintah di bawah ini

Sebuah gambar berisi teks, cuplikan layar, Font

Konten yang dihasilkan AI mungkin salah.

Jalankan ulang di postman

<http://127.0.0.1:8000/api/posts>

**Sebuah gambar berisi teks, cuplikan layar, software, Software multimedia

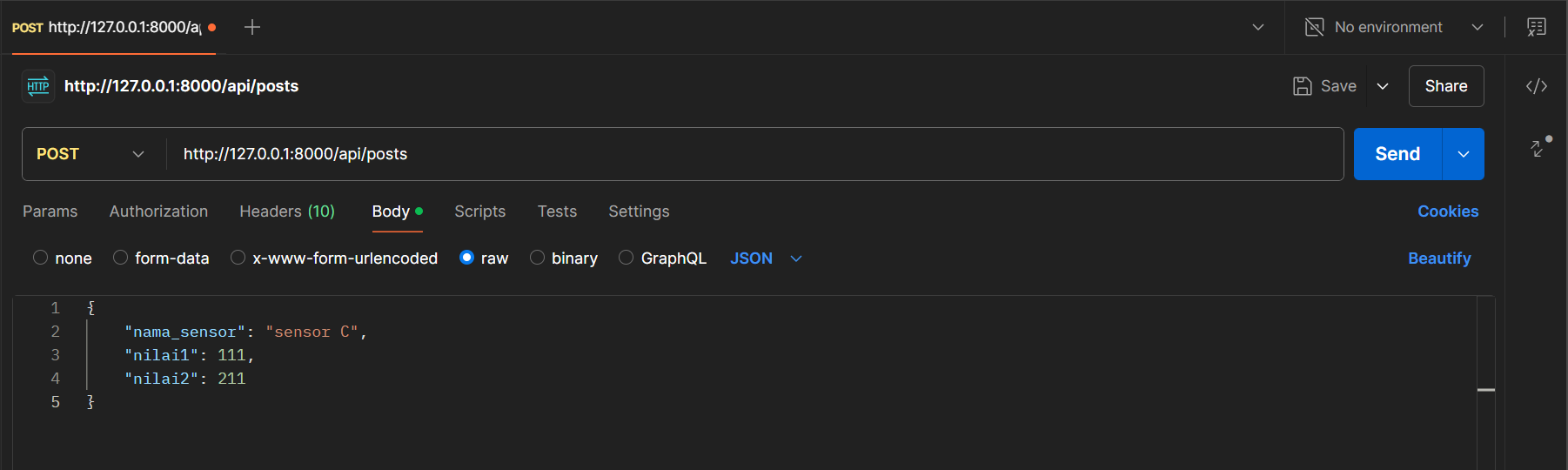
Konten yang dihasilkan AI mungkin salah.**

API telah berfungsi untuk mengambil data dari database. Langkah berikutnya adalah melakukan percobaan insert data ke tabel di database menggunakan API. Caranya adalah mengganti method menjadi POST kemudian pada bagian header ubah menjadi sebagai berikut

Sebuah gambar berisi teks, software, Software multimedia, cuplikan layar

Konten yang dihasilkan AI mungkin salah.

Pada bagian body ubah menjadi sebagai berikut



Kemudian klik send. Pastikan data berhasil di-insert kedatabase seperti tampilan berikut

Sebuah gambar berisi teks, cuplikan layar, software, Software multimedia

Konten yang dihasilkan AI mungkin salah.

Check manual di phpmyadmin, pastikan data baru masuk

Sebuah gambar berisi teks, cuplikan layar, software, Ikon komputer

Konten yang dihasilkan AI mungkin salah.

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

Download dan install aplikasi ngrok pada URL   :  <https://dashboard.ngrok.com/signup>

kemudian lakukan registrasi.

Login ke web ngrok, kemudian download aplikasi ngrok sesuai sistem operasi

Sebuah gambar berisi teks, cuplikan layar, software, Software multimedia

Konten yang dihasilkan AI mungkin salah.

Lakukan ekstraksi

Sebuah gambar berisi teks, cuplikan layar, Font, garis

Konten yang dihasilkan AI mungkin salah.

Buka command prompt dari alamat folder ekstraksi seperti berikut :

Sebuah gambar berisi teks, cuplikan layar, software, Software multimedia

Konten yang dihasilkan AI mungkin salah.

Kemudian jalankan perintah  sesuai yang ada di akun ngrok :

.\ngrok config add-authtoken 2tzIZUPQHlxX2QKLxXNjjrTL3OF\_5gSx59msspx7YHwMGu858

Sebuah gambar berisi teks, cuplikan layar, Font

Konten yang dihasilkan AI mungkin salah.

Sebuah gambar berisi teks, Font, cuplikan layar

Konten yang dihasilkan AI mungkin salah.

**Kemudian jalankan perintah berikut untuk mengonline kan laravel melalui port 8000**

.\ngrok http 8000

Sebuah gambar berisi teks, Font, cuplikan layar

Konten yang dihasilkan AI mungkin salah.

Kemudian lakukan percobaan menggunakan postman menggunakan URL yang diberikan oleh ngrok. Pada contoh ini, ngrok memberikan URL publik yang dapat diakses melalui internet pada alamat , sesuaikan dengan milik Anda.

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



[https://9b3c-2001-448a-50a0-62e1-7088-4955-4b83-d09d.ngrok-free.app**/api/posts**](https://9b3c-2001-448a-50a0-62e1-7088-4955-4b83-d09d.ngrok-free.app/api/posts)

**Sebuah gambar berisi teks, cuplikan layar, software, Software multimedia

Konten yang dihasilkan AI mungkin salah.**

Berikutnya lakukan percobaan melakukan insert data baru melalui API

<https://9b3c-2001-448a-50a0-62e1-7088-4955-4b83-d09d.ngrok-free.app/api/posts>

Ubah method menjadi POST dan parameter header dan body sesuaikan

**Sebuah gambar berisi teks, software, Software multimedia, Font

Konten yang dihasilkan AI mungkin salah.**

Sebuah gambar berisi teks, cuplikan layar, software, Software multimedia

Konten yang dihasilkan AI mungkin salah.

Sampai disini API yang dibangun menggunakan laravel sudah dapat berjalan dengan baik dan dapat diakses melalui URL publik.

Sebuah gambar berisi teks, cuplikan layar, software, Ikon komputer

Konten yang dihasilkan AI mungkin salah.

Pada bab berikutnya,  akses API akan dilakukan melalui simulator WOKWI ESP32 yang telah dibuat pada bab sebelumnya.

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

**3.1 Experimental Results (Hasil Eksperimen)**

Pembuatan dan pengujian API menggunakan Laravel 11 serta integrasi dengan Ngrok telah berhasil diimplementasikan. Seluruh proses pengembangan berjalan dengan baik tanpa error yang signifikan.

Hasil pengujian menunjukkan bahwa:

**Hasil pengujian menunjukkan bahwa:**

* Endpoint API berhasil dibuat dan diuji menggunakan Postman, dengan respons sesuai ekspektasi untuk setiap metode HTTP (GET, POST, PUT, DELETE).
* Koneksi database MySQL berjalan dengan baik, memungkinkan pertukaran data antara API dan database tanpa hambatan.
* Integrasi Ngrok berhasil, sehingga API dapat diakses secara publik melalui URL yang dihasilkan oleh Ngrok.
* Validasi input dan output API berjalan sesuai skenario pengujian, memastikan data yang dikirim dan diterima sesuai format yang diharapkan.

**Berikut adalah rata-rata hasil simulasi:**

* Kecepatan respons API: Waktu eksekusi rata-rata cukup cepat dan stabil dalam menangani permintaan.
* Keakuratan data: Data yang dikirim dan diterima sesuai dengan struktur JSON yang diharapkan.
* Stabilitas sistem: API dapat diakses secara berulang tanpa error atau downtime selama pengujian.

**3.2 Discussion (Pembahasan)**

Keberhasilan eksperimen ini menunjukkan bahwa API yang dibangun menggunakan Laravel 11 dapat berfungsi dengan baik dalam menangani permintaan HTTP dan mengelola data dengan database MySQL. Integrasi dengan Ngrok memungkinkan API diakses secara publik tanpa memerlukan konfigurasi server tambahan, sehingga mempercepat proses pengujian dan pengembangan.

**Faktor yang Mempengaruhi Hasil:**

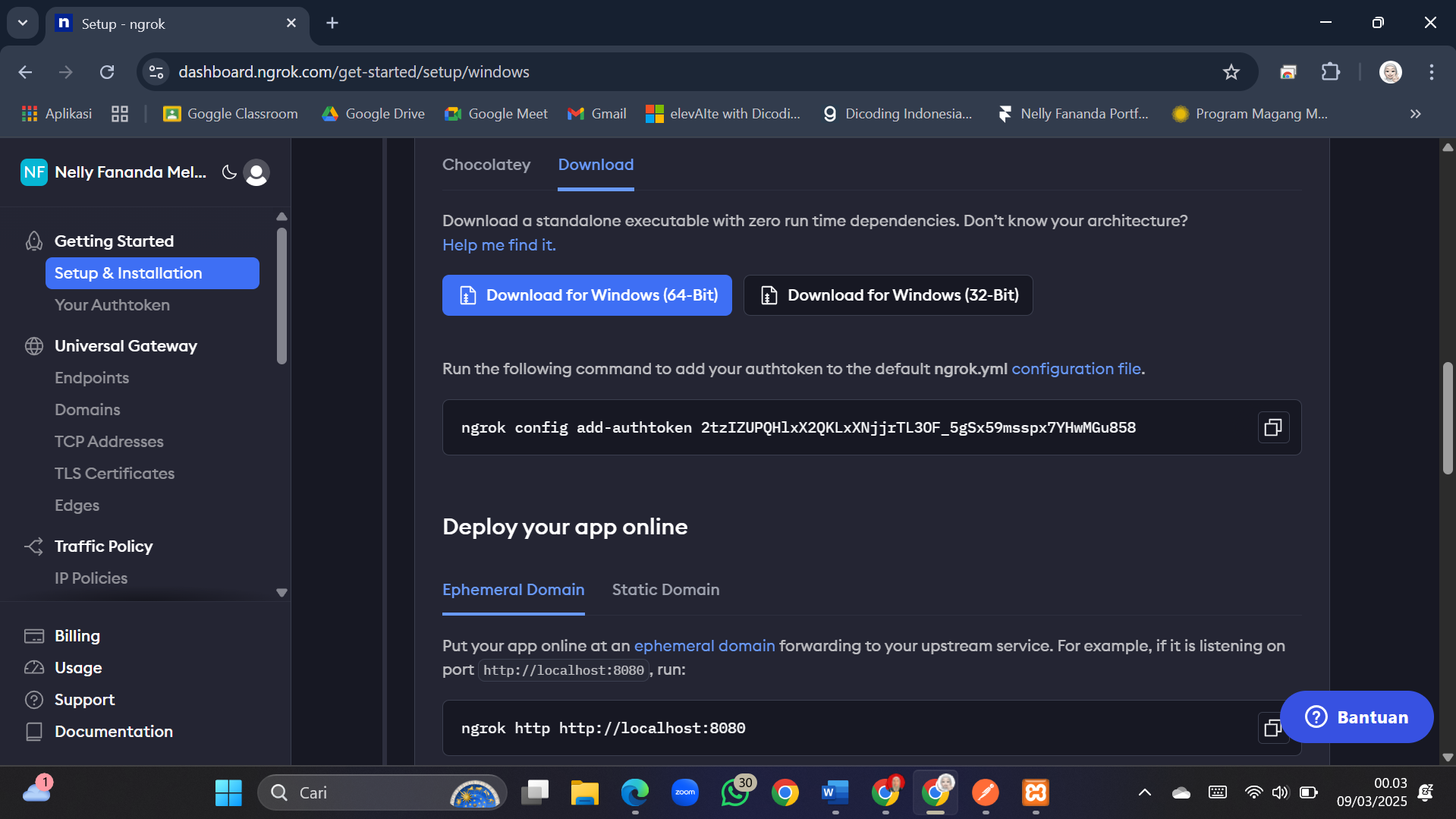
* Logika Pemrograman: Jika implementasi controller atau route tidak sesuai, API dapat mengembalikan respons yang salah atau mengalami error.
* Konfigurasi Database: Kesalahan dalam pengaturan koneksi database di file .env dapat menyebabkan API gagal mengakses atau menyimpan data.
* Keamanan API: Tanpa middleware dan validasi input yang tepat, API dapat rentan terhadap serangan seperti SQL Injection dan Cross-Site Scripting (XSS).
* Koneksi Internet: Ngrok memerlukan koneksi internet yang stabil untuk menjaga aksesibilitas API secara publik.
* Pengelolaan Request: Jika API tidak dioptimalkan, jumlah permintaan yang tinggi dapat menyebabkan bottleneck dan memperlambat respons.

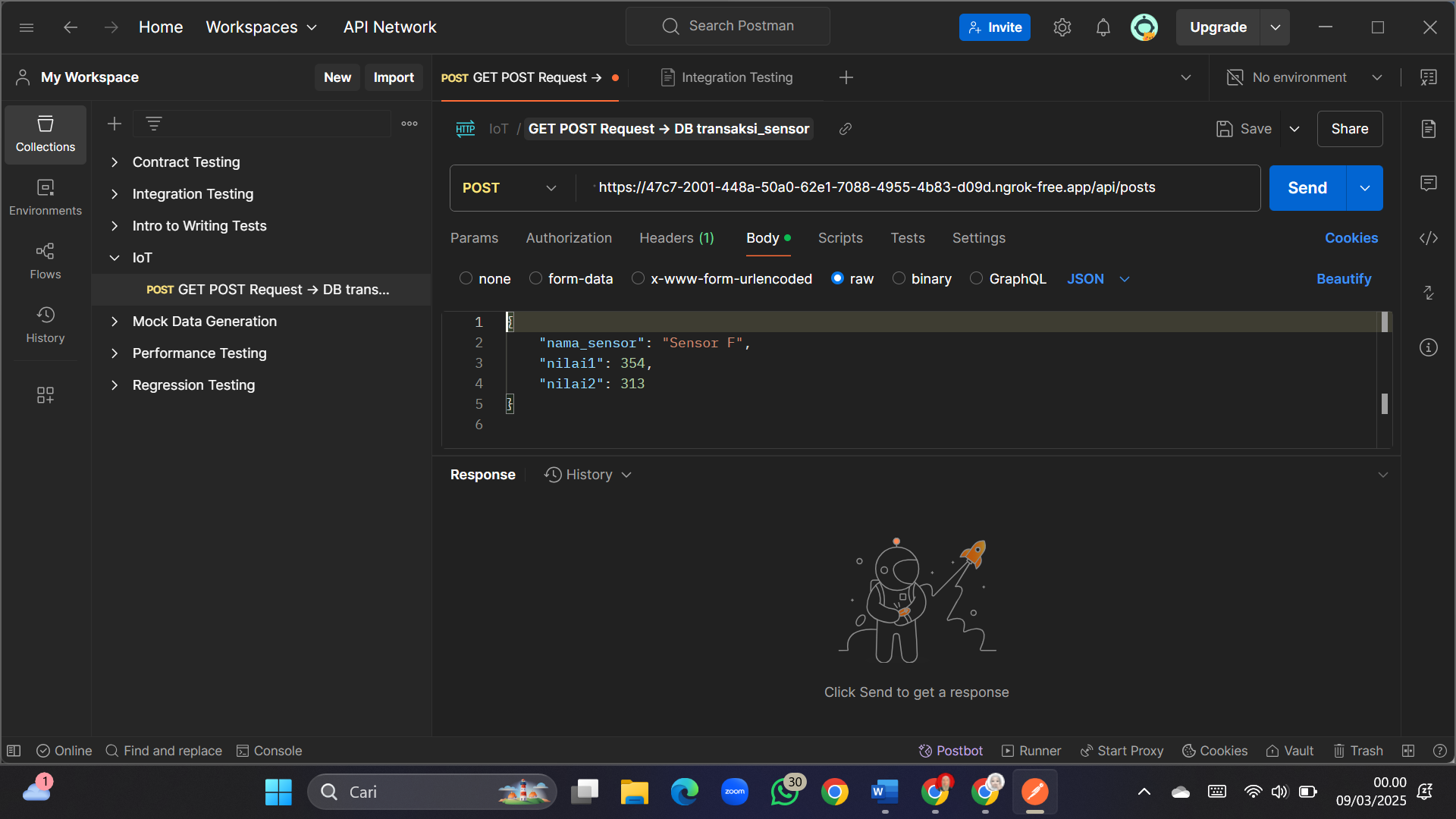
**Peningkatan yang Dapat Dilakukan:**

* Menambahkan Middleware Keamanan: Seperti autentikasi token (JWT) untuk membatasi akses ke API.
* Optimasi Query Database: Menggunakan indexing dan eager loading untuk mempercepat eksekusi query.
* Implementasi Caching: Memanfaatkan Redis atau Laravel Cache untuk mengurangi beban database.
* Logging dan Monitoring: Menggunakan Laravel Log atau alat eksternal seperti Postman Monitor untuk melacak error dan performa API.
* Integrasi dengan Frontend atau Mobile App: Menghubungkan API dengan aplikasi berbasis web atau mobile untuk penggunaan lebih luas.

Hasil eksperimen ini dapat menjadi dasar bagi pengembangan sistem berbasis API yang lebih kompleks, seperti aplikasi IoT, sistem manajemen data, atau layanan berbasis cloud.

**4. Appendix (Lampiran)**

****

****

