LAPORAN PRAKTIKUM INTERNET OF THINGS (IoT)

Fakultas Vokasi, Universitas Brawijaya

**Praktik Simulasi Relay, Button, dan LED**

**Praktik Simulasi Pembuatan Sensor Jarak (Ultrasonik)**

**Praktik Simulasi** **Pembuatan API Menggunakan Laravel dan Ngrok**

****

*Yoga Muhammad Fauzi*

*Fakultas Vokasi, Universitas Brawijaya*

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

Abstrak

Internet of Things (IoT) memungkinkan pengembangan sistem cerdas dalam berbagai bidang, termasuk manajemen lalu lintas. Dalam eksperimen ini, dilakukan simulasi **sensor kelembaban suhu** menggunakan platform **Wokwi**, yang kemudian dipindahkan ke **Visual Studio Code (VSCode)** untuk pengembangan lebih lanjut. Tujuan eksperimen ini adalah memahami cara kerja mikrokontroler serta menguji kompatibilitas kode antara simulator berbasis web dan lingkungan pengembangan lokal.

Metode yang digunakan meliputi pemrograman dalam bahasa **C/C++**, pengujian simulasi di Wokwi, serta pemindahan proyek ke VSCode dengan bantuan **PlatformIO**. Setelah dipindahkan ke VSCode, kode dapat dijalankan dengan baik setelah dilakukan beberapa penyesuaian konfigurasi.

Kesimpulan dari eksperimen ini adalah Wokwi sangat membantu dalam tahap awal pengembangan proyek IoT sebelum implementasi pada perangkat keras. Sementara itu, penggunaan VSCode memberikan fleksibilitas lebih dalam debugging dan optimasi kode. Dengan kombinasi kedua platform ini, proses pengembangan sistem lalu lintas berbasis IoT menjadi lebih efisien dan terstruktur.

*Kata kunci: Internet of Things, Sensor Kelembaban Suhu, Wokwi, Visual Studio Code, PlatformIO.*

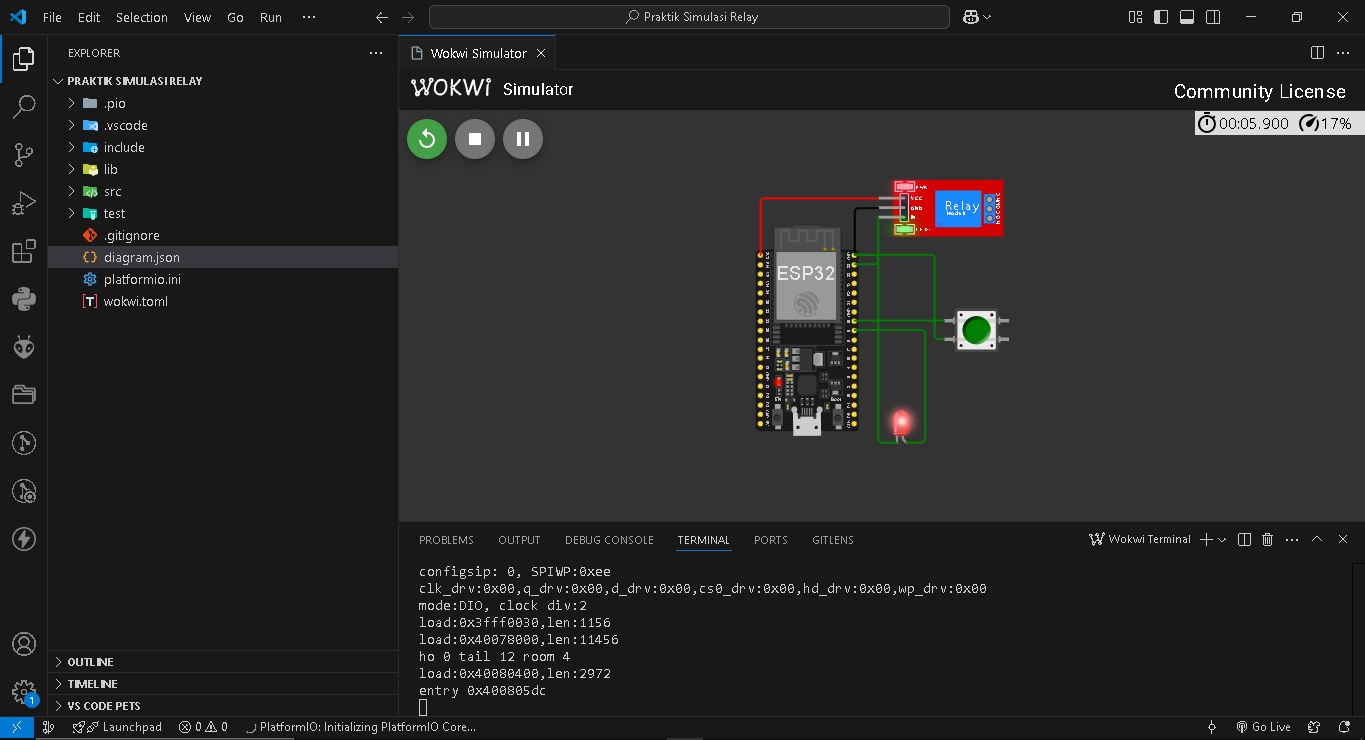
**Praktik Simulasi Relay, Button, dan LED**

1. Pendahuluan
2. Latar Belakang

Internet of Things adalah sebuah konsep yang menggambarkan jaringan perangkat fisik yang saling terhubung dan berbagi data satu sama lain melalui internet. Salah satu implementasi sederhana IoT yang sering dijumpai yaitu praktik simulasi relay, button, dan led dapat dikendalikan secara digital.

Dalam praktikum ini, dilakukan simulasi Relay, Button, dan LED menggunakan Platform Wokwi, yaitu sebuah simulator berbasis web yang mendukung berbagai mikrokontroler Arduino. Platform ini memungkinkan pengujian kode tanpa memerlukan perangkat keras fisik. Setelah dibuat, proyek akan dipindahkan ke aplikasi Visual Studio Code (VSCode) untuk pengembangan lebih lanjut.

1. Tujuan eskperimen
2. Dapat mensimulasikan sensor kelembaban dan suhu menggunakan platform Wokwi untuk memahami dasar-dasar pemrograman mikrokontroler.
3. Mempelajari integrasi Arduino dengan simulator.
4. Menganalisis cara kerja program dalam membaca data dari Relay, Button, dan LED.
5. Menambahkan pemahaman tentang pemrograman sistem Internet of Things (IoT).
6. Metodologi
7. Alat dan bahan
8. Laptop
9. Internet
10. Software VSCode
11. Langkah Implementasi
12. Pembuatan simulasi Relay, Button, LED
13. Membuka halaman website WOKWI (<https://wokwi.com/>), lalu pilih ESP32
14. Scroll kebawah sampai menemukan “Starter Templates” lalu pilih / klik ESP32
15. Klik tombol + untuk menambahkan komponen
16. Pilih Relay Module
17. Pilih Push Button
18. Pilih LED
19. Sambungkan port VCC pada Relay Module ke port 3V3 pada ESP
20. Sambungkan port GND pada Relay Module ke port GND pada ESP
21. Sambungkan port IN pada Relay Module ke port Angka pada ESP (Contoh esp:23)
22. Sambungkan port 1.1.l pada Button ke port Angka pada ESP (Contoh esp:19)
23. Sambungkan port 1.2.l pada Button ke port GND pada ESP
24. Sambungkan port Bengkok pada LED ke port Angka pada ESP (Contoh esp:18)
25. Sambungkan port Lurus pada LED ke port GND pada ESP
26. Atur codingan di sebelah kiri agar simulasi relay, button, led berjalan
27. Jalankan project dengan menekan tombol start warna hijau disebelah kanan
28. Menjalankan project di Visual Studio Code / VsCode
29. Buka aplikasi VsCode
30. Install extension Wokwi Simulator
31. Setelah Install extension wokwi, tekan tombol ctrl+shift+p, lalu ketikan wokwi dan klik “Request a New License”
32. Klik open
33. Tautkan akun wokwi sam seperti sebelumnya, setelah itu klik “GET YOUR LICENSE”
34. Setelah diarahkan ke VsCode, klik open
35. Dipojok kanan bawah akan muncul notifikasi, maka license wokwi sudah aktif selama 1 bulan
36. Install extension PlaformIO IDE
37. Setelah install, klik logo semut sebelah kiri lalu tunggu loading sampai muncul notifikasi restart VsCode, lakukan restart dengan menutup aplikasi lalu membukanya kembali
38. Klik kembali logo semut / PlatformIO IDE, lalu klik New Project
39. Masukan nama project, Board menyesuaikan dengan ESP32, dll, lalu klik finish
40. Tunggu loading hingga selesai
41. Setelah selesai maka akan terbuat sebuah folder berisi file-file
42. Klik folder project sampai menutup, lalu klik tombol centang dibagian bawah kiri, tunggu hingga loading setelah
43. Setelah selesai, cek folder .pio dan folder esp32doit sampai muncul beberapa file, jika gagal lakukan lagi cara diatas
44. Buat file baru dengan klik kanan pada folder project, lalu pilih new file
45. Buat 2 file dengan nama diagram.json dan wokwi.toml
46. Buka kembali wokwi pada browser lalu klik “docs” disebelah profile pojok kanan atas
47. Pada bagian kiri pilih “VS Code Extension”, lalu pilih “Project Config”
48. Salin code dan paste pada file wokwi.toml di VsCode
49. Cari file “firmware.bin” lalu klik kanan dan klik “Copy Relative Path”
50. Paste pada file wokwi bagian “firmware”
51. Cari file “firmware.elf” lalu klik kanan dan klik “Copy Relative Path”
52. Paste pada file wokwi bagian “elf”, lalu simpan ctrl+s
53. Buka kembali project di browser, lalu klik bagian diagram.json, copy semua code
54. Paste kan ke dalam file diagram.json yang ada pada VsCode
55. Buka kembali project di browser, lalu klik bagian sketch.ino, copy semua code
56. Paste kan ke dalam file main.cpp yang berada di folder src, lalu save
57. Start project dengan ctrl+shift+p, ketikan wokwi dan pilih yang Start Simulator
58. Maka project berhasil berjalan, jika gagal tekan tombol centang yang dibawah, dan jalankan ulang



1. Hasil dan Pembahasan
2. Hasil Eksperimen
3. Simulasi Praktik Relay, Button dan LED di Wokwi

* Program berjalan dengan baik, dengan Temperature dan Humidity dapat diatur sesuai kebutuhan
* Tidak ada error dalam simulasi, dan perubahan waktu siklus dapat dilakukan dengan mudah melalui kode.

1. Pemindahan Proyek ke VSCode

* Kode berhasil dipindahkan dan dijalankan di VSCode menggunakan PlatformIO.
* Beberapa konfigurasi tambahan diperlukan, seperti pemilihan board yang sesuai dan penyesuaian pustaka.
* Proses debugging lebih mudah dilakukan dibandingkan dengan simulator Wokwi.

1. Analisis dan Evaluasi

* Penggunaan Wokwi sangat membantu dalam tahap pengembangan awal sebelum implementasi pada perangkat keras.
* VSCode memberikan lebih banyak fleksibilitas dalam debugging dan pengelolaan kode.
* Tidak ada kendala besar dalam pemindahan kode, tetapi diperlukan pemahaman lebih lanjut tentang konfigurasi PlatformIO untuk menghindari error.

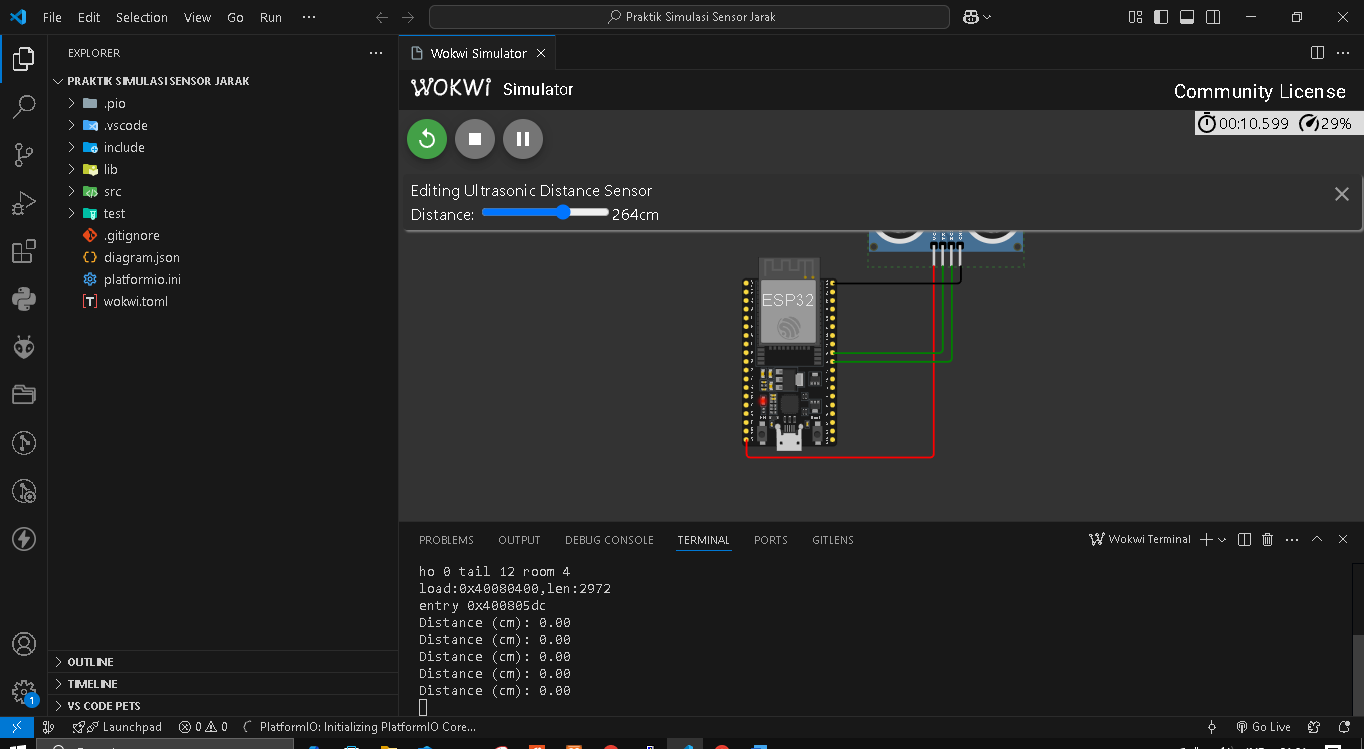
**Praktik Simulasi Pembuatan Sensor Jarak (Ultrasonik)**

1. Pendahuluan
2. Latar Belakang

Internet of Things adalah sebuah konsep yang menggambarkan jaringan perangkat fisik yang saling terhubung dan berbagi data satu sama lain melalui internet. Salah satu implementasi sederhana IoT yang sering dijumpai yaitu Praktik Simulasi Pembuatan Sensor Jarak (Ultrasonik) yang dapat dikendalikan secara digital.

Dalam praktikum ini, dilakukan simulasi sensor kelembaban dan suhu menggunakan Platform Wokwi, yaitu sebuah simulator berbasis web yang mendukung berbagai mikrokontroler Arduino. Platform ini memungkinkan pengujian kode tanpa memerlukan perangkat keras fisik. Setelah dibuat, proyek akan dipindahkan ke aplikasi Visual Studio Code (VSCode) untuk pengembangan lebih lanjut.

1. Tujuan eskperimen
2. Dapat mensimulasikan sensor jarak (ultrasonik) menggunakan platform Wokwi untuk memahami dasar-dasar pemrograman mikrokontroler.
3. Mempelajari integrasi Arduino dengan simulator.
4. Menganalisis cara kerja program dalam membaca data dari jarak secara otomatis.
5. Menambahkan pemahaman tentang pemrograman sistem Internet of Things (IoT).
6. Metodologi
7. Alat dan bahan
8. Laptop
9. Internet
10. Software VSCode
11. Langkah Implementasi
12. Pembuatan simulasi sensor jarak (ultrasonik)
13. Membuka halaman website WOKWI (<https://wokwi.com/>), lalu pilih ESP32
14. Scroll kebawah sampai menemukan “Starter Templates” lalu pilih / klik ESP32
15. Klik tombol + untuk menambahkan komponen
16. Pilih HC-SR04 Ultrasonik Distance Sensor
17. Sambungkan port VCC pada Sensor ke port 5V pada ESP
18. Sambungkan port TRIG pada Sensor ke port angka pada ESP (contoh: esp:18)
19. Sambungkan port ECHO pada Sensor ke port angka pada ESP (contoh: esp:5)
20. Sambungkan port GND pada Sensor ke port GND pada ESP
21. Atur codingan di sebelah kiri agar sensor jarak berjalan
22. Jalankan project dengan menekan tombol start warna hijau disebelah kanan
23. Menjalankan project di Visual Studio Code / VsCode
24. Buka aplikasi VsCode
25. Install extension Wokwi Simulator
26. Setelah Install extension wokwi, tekan tombol ctrl+shift+p, lalu ketikan wokwi dan klik “Request a New License”
27. Klik open
28. Tautkan akun wokwi sam seperti sebelumnya, setelah itu klik “GET YOUR LICENSE”
29. Setelah diarahkan ke VsCode, klik open
30. Dipojok kanan bawah akan muncul notifikasi, maka license wokwi sudah aktif selama 1 bulan
31. Install extension PlaformIO IDE
32. Setelah install, klik logo semut sebelah kiri lalu tunggu loading sampai muncul notifikasi restart VsCode, lakukan restart dengan menutup aplikasi lalu membukanya kembali
33. Klik kembali logo semut / PlatformIO IDE, lalu klik New Project
34. Masukan nama project, Board menyesuaikan dengan ESP32, dll, lalu klik finish
35. Tunggu loading hingga selesai
36. Setelah selesai maka akan terbuat sebuah folder berisi file-file
37. Klik folder project sampai menutup, lalu klik tombol centang dibagian bawah kiri, tunggu hingga loading setelah
38. Setelah selesai, cek folder .pio dan folder esp32doit sampai muncul beberapa file, jika gagal lakukan lagi cara diatas
39. Buat file baru dengan klik kanan pada folder project, lalu pilih new file
40. Buat 2 file dengan nama diagram.json dan wokwi.toml
41. Buka kembali wokwi pada browser lalu klik “docs” disebelah profile pojok kanan atas
42. Pada bagian kiri pilih “VS Code Extension”, lalu pilih “Project Config”
43. Salin code dan paste pada file wokwi.toml di VsCode
44. Cari file “firmware.bin” lalu klik kanan dan klik “Copy Relative Path”
45. Paste pada file wokwi bagian “firmware”
46. Cari file “firmware.elf” lalu klik kanan dan klik “Copy Relative Path”
47. Paste pada file wokwi bagian “elf”, lalu simpan ctrl+s
48. Buka kembali project di browser, lalu klik bagian diagram.json, copy semua code
49. Paste kan ke dalam file diagram.json yang ada pada VsCode
50. Buka kembali project di browser, lalu klik bagian sketch.ino, copy semua code
51. Paste kan ke dalam file main.cpp yang berada di folder src, lalu save
52. Start project dengan ctrl+shift+p, ketikan wokwi dan pilih yang Start Simulator
53. Maka project berhasil berjalan, jika gagal tekan tombol centang yang dibawah, dan jalankan ulang



1. Hasil dan Pembahasan
2. Hasil Eksperimen
3. Simulasi Sensor Jarak Ultrasonik di Wokwi

* Program berjalan dengan baik, dengan Temperature dan Humidity dapat diatur sesuai kebutuhan
* Tidak ada error dalam simulasi, dan perubahan waktu siklus dapat dilakukan dengan mudah melalui kode.

1. Pemindahan Proyek ke VSCode

* Kode berhasil dipindahkan dan dijalankan di VSCode menggunakan PlatformIO.
* Beberapa konfigurasi tambahan diperlukan, seperti pemilihan board yang sesuai dan penyesuaian pustaka.
* Proses debugging lebih mudah dilakukan dibandingkan dengan simulator Wokwi.

1. Analisis dan Evaluasi

* Penggunaan Wokwi sangat membantu dalam tahap pengembangan awal sebelum implementasi pada perangkat keras.
* VSCode memberikan lebih banyak fleksibilitas dalam debugging dan pengelolaan kode.
* Tidak ada kendala besar dalam pemindahan kode, tetapi diperlukan pemahaman lebih lanjut tentang konfigurasi PlatformIO untuk menghindari error.

**Praktik Simulasi Pembuatan API Menggunakan Laravel dan Ngrok**

1. Pendahuluan
2. Latar Belakang

Internet of Things adalah sebuah konsep yang menggambarkan jaringan perangkat fisik yang saling terhubung dan berbagi data satu sama lain melalui internet. Salah satu implementasi sederhana IoT yang sering dijumpai yaitu Praktik Simulasi Pembuatan API menggunakan Laravel dan Ngrok.

Dalam praktikum ini, dilakukan pembuatan API menggunakan Laravel 11 dan Ngrok menggunakan Platform VSCode untuk pengembangan lebih lanjut.

1. Tujuan eskperimen
2. Dapat mensimulasikan API menggunakan platform VSCode untuk memahami dasar-dasar pemrograman.
3. Menganalisis cara kerja program dalam membaca data dari API Laravel.
4. Menambahkan pemahaman tentang pemrograman sistem API.
5. Metodologi
6. Alat dan bahan
7. Laptop
8. Internet
9. Software VSCode
10. Langkah Implementasi
11. Pembuatan API
12. Membuat folder laravel dengan perintah **composer create-project --prefer-dist laravel/laravel:^11.0 (nama folder anda)** pada cmd
13. Lalu ketik **cd (nama folder yang sudah dibuat)**
14. Lalu Ketik **code .** maka akan diarahkan ke vscode
15. Buat database menggunakan Xampp dengan nama iot\_25
16. Ubah konfigurasi **.env** di laravel

|  |
| --- |
| 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 model  **TransaksiSensor.php** dengan cara menjalankan perintah berikut di terminal **php artisan make:model TransaksiSensor -m**
2. 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 = [];  } |

1. Buat migration dengan perintah **php artisan make:migration create\_transaksi\_sensors\_table**
2. 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');      }  }; |

1. Kemudian jalankan perintah berikut untuk membuat tabel : **php artisan migrate**
2. Buat Resource dengan menjalankan perintah : **php artisan make:resource TransaksiSensorResource**
3. Ubah isi file **TransaksiSensorResource.php** yang ada di folder app-Http-Resources 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,          ];      }  } |

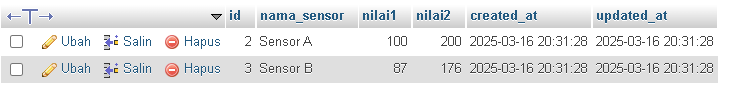
1. J Buat API controller dengan menjalankan perintah : **php artisan make:controller Api/TransaksiSensorController**
2. Ubah isi file **app/Http/Controllers/Api/TransaksiSensorController.php**

|  |
| --- |
| <?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);  }  } |

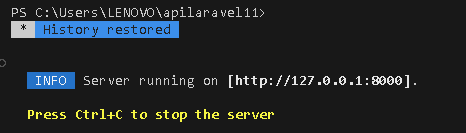
1. Buat route khusus API dengan menjalankan perintah :  **php artisan install:api**
2. Buka file **routes/api.php**  dan ubah isi file menjadi :

|  |
| --- |
| <?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); |

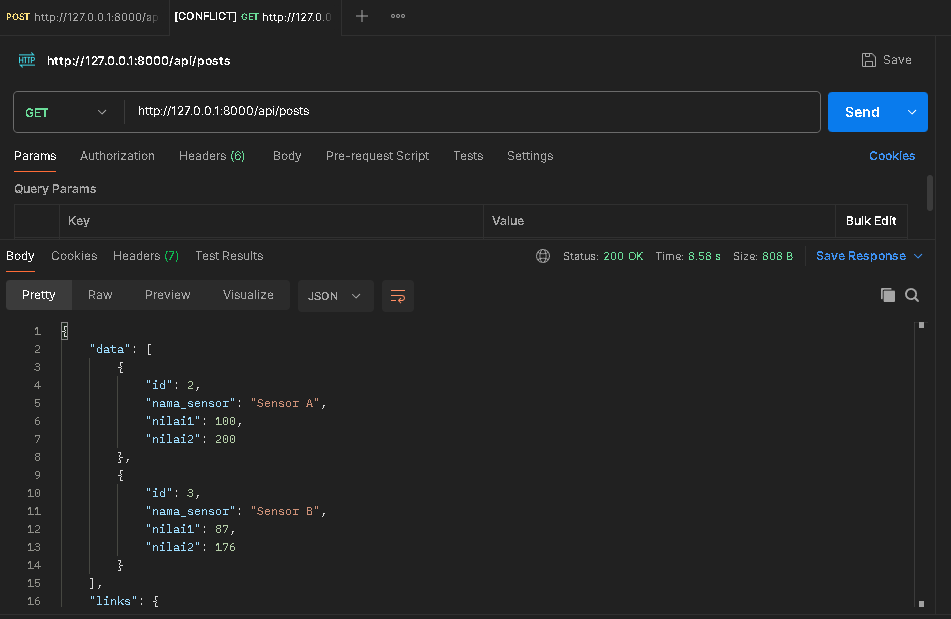
1. Kemudian pastikan routes telah terbentuk dengan menjalankan perintah : **php artisan route:list**
2. Buatlah data sesuai tabel yang ada di database



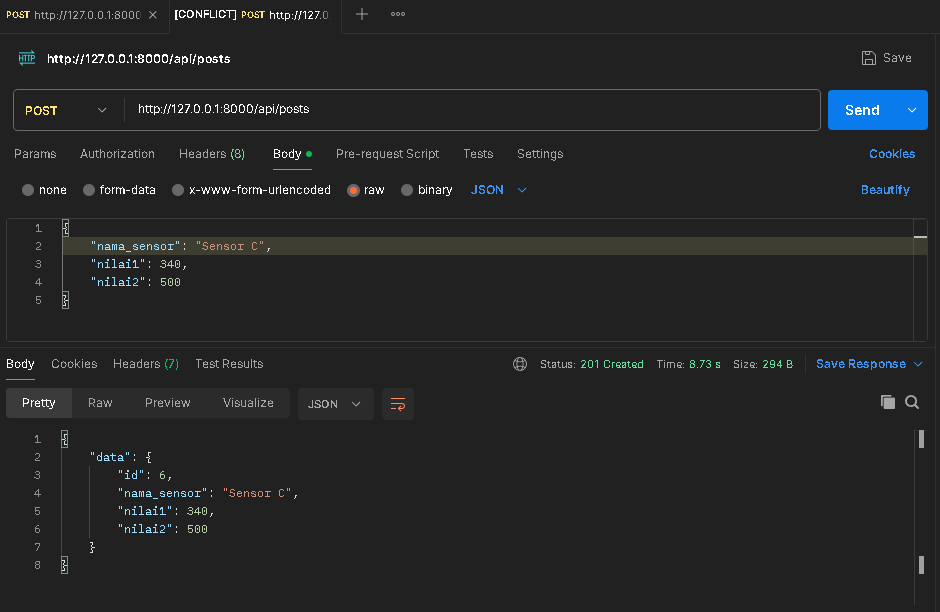
1. Untuk melakukan percobaan akses api, pastikan aplikasi laravel dijalankan dengan perintah : **php artisan serve**



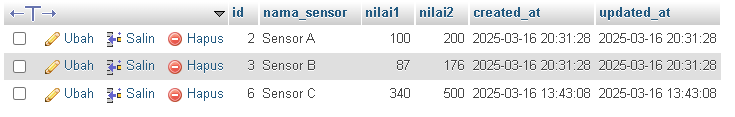
1. Menjalankan project di Postman
2. Download aplikasi postman pada link berikut : <https://www.postman.com/downloads/>
3. Lakukan prosedur instalasi dan jalankan aplikasi postman.
4. Pada bagian URL masukkan alamat server laravel  **http://127.0.0.1:8000/api/posts**
5. Atau bisa diakses melalui url : **http://localhost:8000/api/posts**
6. Pilih method **GET** untuk mengambil data dari database , kemudian klik tombol **SEND**



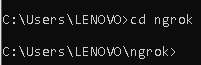
1. Pastikan data dikembalikan dalam bentuk json seperti tampilan contoh diatas
2. API telah berfungsi untuk mengambil data dari database. Langkah berikutnya adalah melakukan percobaan insert data ke tabel di database menggunakan API. Caranya adalah mengganti method menjadi POST kemudian pada bagian header ubah menjadi sebagai berikut:
3. Pada bagian body ubah menjadi sebagai berikut
4. Kemudian klik send. Pastikan data berhasil di-insert kedatabase seperti tampilan berikut



1. Check manual di phpmyadmin, pastikan data baru masuk



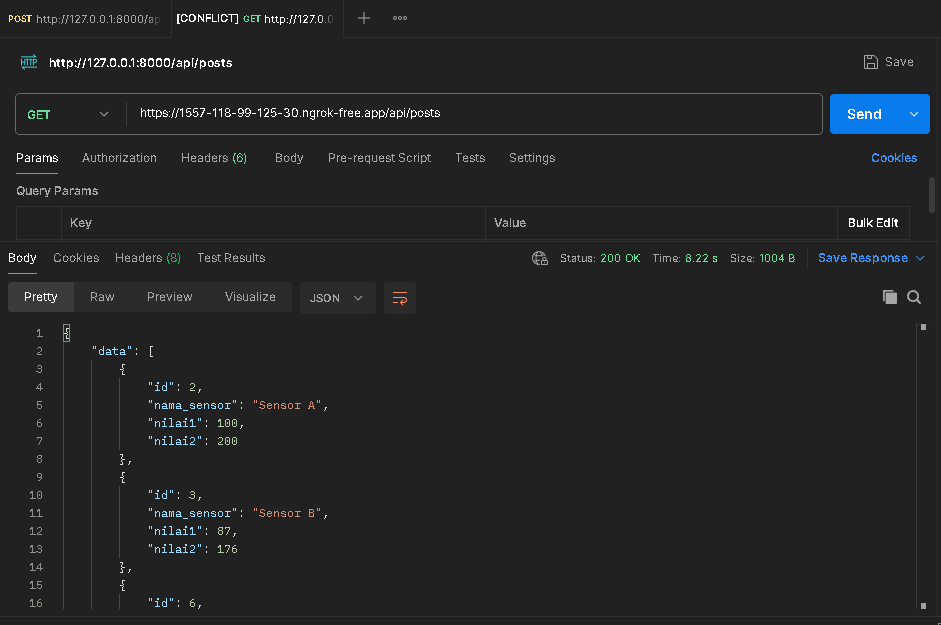
1. Langkah berikutnya adalah mengonline-kan API menggunakan service ngrok sehingga API dapat diakses melalui device iot atau simulasi wokwi iot. Download dan install aplikasi ngrok pada URL : <https://dashboard.ngrok.com/signup> kemudian lakukan registrasi. Login ke web ngrok, kemudian download aplikasi ngrok sesuai sistem operasi
2. Lakukan ekstraksi
3. Buka command prompt dari alamat folder ekstraksi seperti berikut :



1. Kemudian jalankan perintah sesuai yang ada di akun ngrok :



1. Kemudian jalankan perintah berikut untuk mengonline kan laravel melalui port 8000 ngrok http [http://localhost:8000](http://localhost:8080)
2. 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.
3. Untuk melakukan percobaan GET api , maka URL harus ditambahkan alamat endpoint  menjadi sebagai berikut:
4. Berikutnya lakukan percobaan melakukan insert data baru melalui API [https://e521-2405-8740-6315-3520-2099-2415-5077-c12f.ngrok-free.app**/api/posts**](https://e521-2405-8740-6315-3520-2099-2415-5077-c12f.ngrok-free.app/api/posts)Ubah method menjadi POST dan parameter header dan body sesuaikan
5. Maka project berhasil berjalan, jika gagal tekan tombol centang yang dibawah, dan jalankan ulang



1. Hasil dan Pembahasan
2. Hasil Eksperimen
3. Simulasi API

* Program berjalan dengan baik, dengan melakukan CRUD menggunakan API
* Tidak ada error dalam simulasi

1. Analisis dan Evaluasi

* Penggunaan API sangat membantu
* VSCode memberikan lebih banyak fleksibilitas dalam debugging dan pengelolaan kode..