

PRAKTIK PENGEMBANGAN BACK END

16 November 2024

FUNGSI CREATE DI EXPRESS.JS

a. Apa itu CRUD

CRUD adalah singkatan dari Create, Read, Update, dan Delete, yang merupakan empat fungsi dasar yang digunakan untuk mengelola data dalam basis data dan penting untuk sistem manajemen basis data. Fungsi-fungsi ini sering digunakan sebagai dasar untuk membangun aplikasi web.

Pertimbangkan panel admin atau backend apa pun, CRUD akan menjadi fungsi pentingnya untuk mengelola data.

Membuat / create

Operasi create digunakan untuk **menambahkan data baru ke dalam basis data**. Misalnya, jika Anda memiliki basis data pelanggan dan ingin menambahkan pelanggan baru ke dalam basis data, Anda dapat menggunakan operasi create. Hal ini dapat melibatkan penambahan rekaman baru ke tabel pelanggan dengan nama pelanggan, alamat, dan informasi lainnya.

Membaca / read

Operasi baca digunakan untuk **mengambil data dari basis data**. Misalnya, jika Anda ingin mengambil daftar semua pelanggan dalam basis data, Anda akan menggunakan operasi baca. Ini dapat melibatkan permintaan ke tabel pelanggan untuk mengambil semua rekaman dan menampilkannya di situs web atau aplikasi.

Memperbarui / update

Operasi pembaruan digunakan untuk **mengubah data yang ada dalam basis data**. Misalnya, jika Anda ingin memperbarui alamat pelanggan dalam basis data, Anda akan menggunakan operasi pembaruan. Ini dapat melibatkan perubahan catatan dalam tabel pelanggan dengan informasi alamat baru.

Menghapus / delete

Operasi hapus digunakan untuk **menghapus data dari basis data** . Misalnya, jika Anda ingin menghapus pelanggan dari basis data, Anda akan menggunakan operasi hapus. Ini dapat melibatkan penghapusan rekaman dari tabel pelanggan.

b. Apa Itu Model

Model menjadi salah satu bagian penting dari konsep [MVC](#) (Model-View-Controller) pada framework Laravel yang bertanggung jawab untuk mengatur interaksi antara aplikasi dengan database. Dalam konsep MVC, Model dipadukan dengan Controller dan View untuk membentuk sebuah fitur atau halaman pada aplikasi Laravel kita. Model bertanggung jawab untuk memproses data dari database dan mengembalikan data tersebut ke Controller.

c. Apa itu Controller

Dalam konteks Laravel, controller adalah kelas PHP yang bertanggung jawab untuk memproses permintaan HTTP dari pengguna dan mengembalikan respons yang sesuai. Controller menyatukan logika aplikasi dan memastikan bahwa permintaan dari pengguna diarahkan dengan benar ke model yang sesuai atau ke tampilan yang tepat.

d. Apa itu Routes / routing

Routing adalah salah satu komponen inti dalam aplikasi web yang memungkinkan Anda mendefinisikan alamat URL yang akan dipetakan ke controller tertentu dalam aplikasi Anda. Dalam Laravel, routing digunakan untuk mengarahkan semua request HTTP ke handler / method yang tepat.

Routing di Laravel sangat fleksibel dan memberi Anda kontrol penuh atas setiap request ditangani oleh aplikasi Anda. Ini

memungkinkan struktur yang rapi dan mudah dikelola untuk aplikasi yang kompleks.

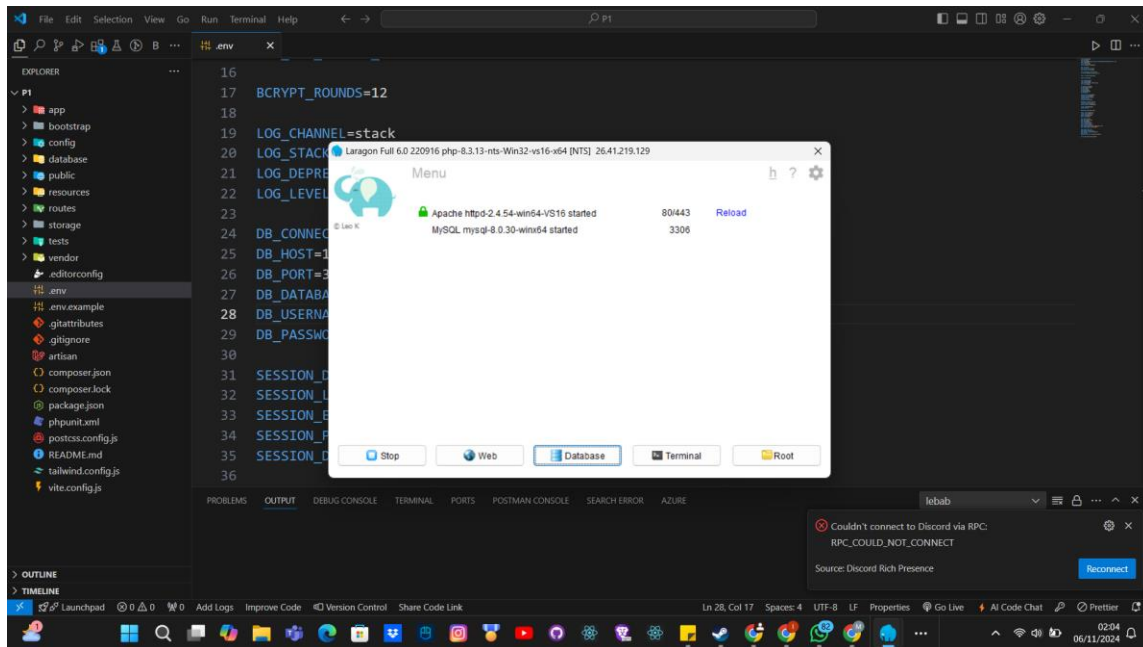
e. Konsep MVC (Model, View, Controller)

MVC adalah sebuah pendekatan perangkat lunak yang memisahkan aplikasi logika dari presentasi. MVC memisahkan aplikasi berdasarkan komponen-komponen aplikasi, seperti : manipulasi data, controller, dan user interface.

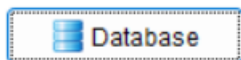
1. Model, Model mewakili struktur data. Biasanya model berisi fungsi-fungsi yang membantu seseorang dalam pengelolaan basis data seperti memasukkan data ke basis data, pembaruan data dan lain-lain.
2. View, View adalah bagian yang mengatur tampilan ke pengguna. Bisa dikatakan berupa halaman web.
3. Controller, Controller merupakan bagian yang menjembatani model dan view.

e. Proses Pembuatan Fungsi Create di Laravel

1. Buatlah database di phpMyAdmin, Untuk dipercobaan ini menggunakan Nama database : Percobaan1



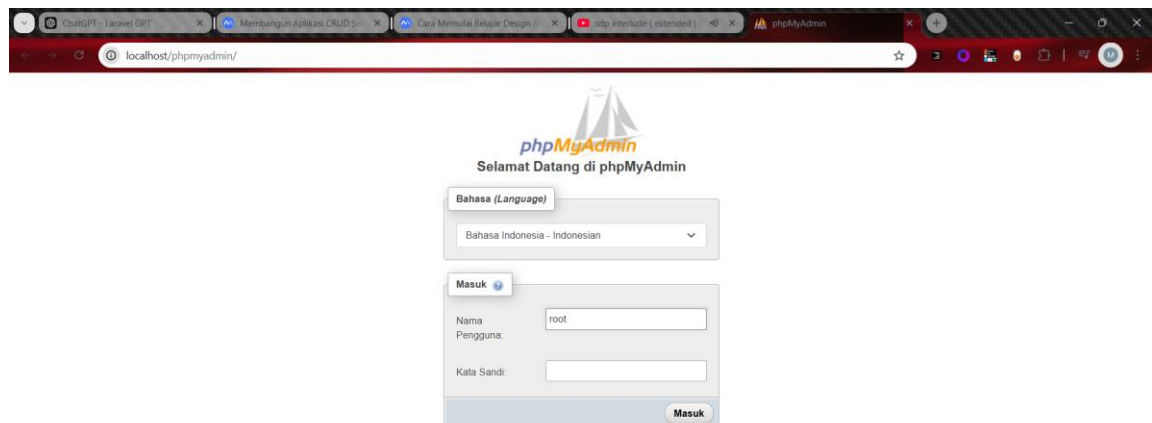
- a) Klik Database



- b) Kemudian masuk ke phpmyadminnya

Username : root

Pw :



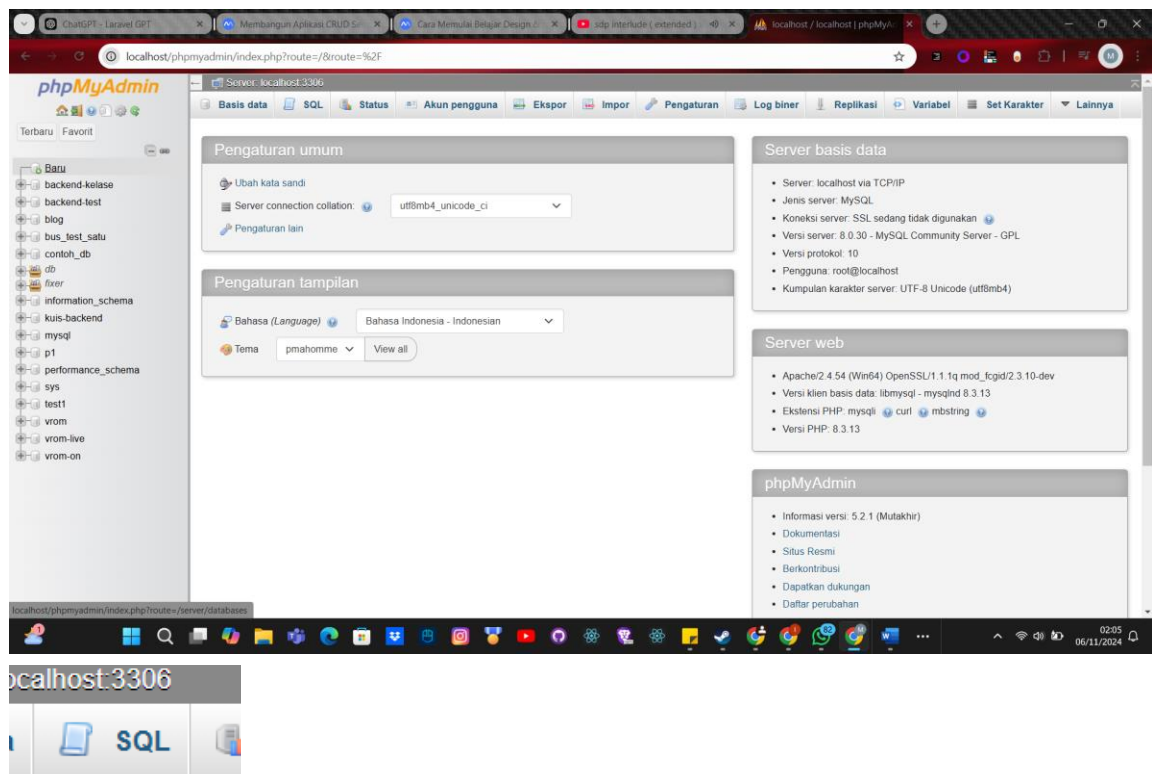
c) Setelah itu buatlah databasenya dengan nama percobaan2

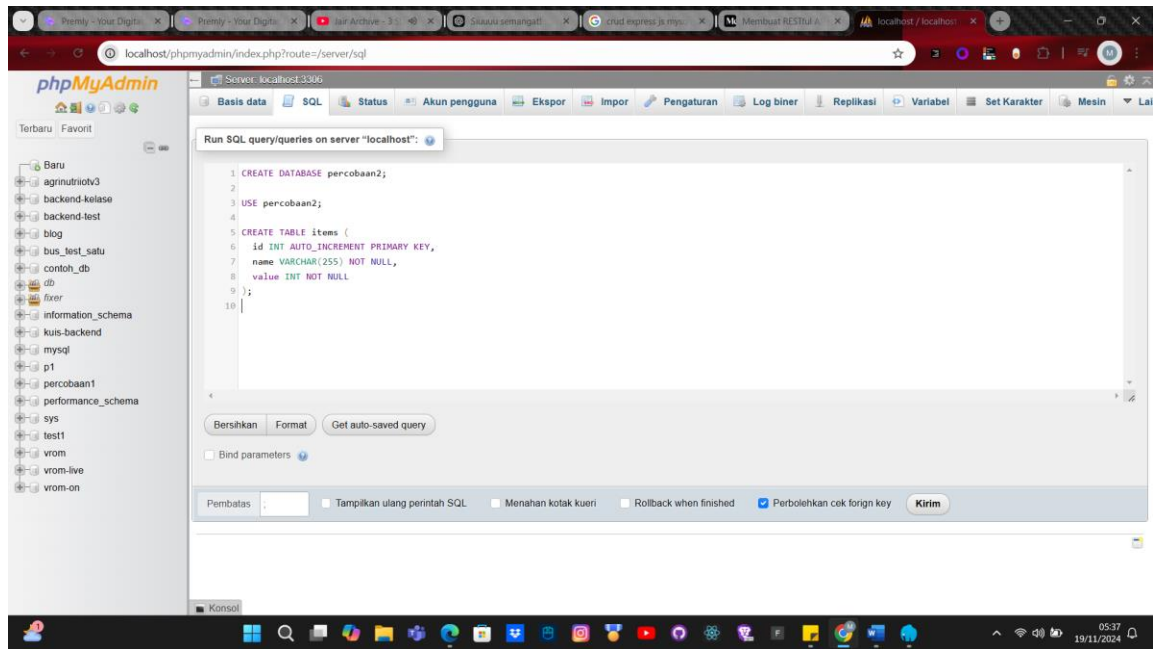
```
CREATE DATABASE percobaan2;
```

```
USE percobaan2;
```

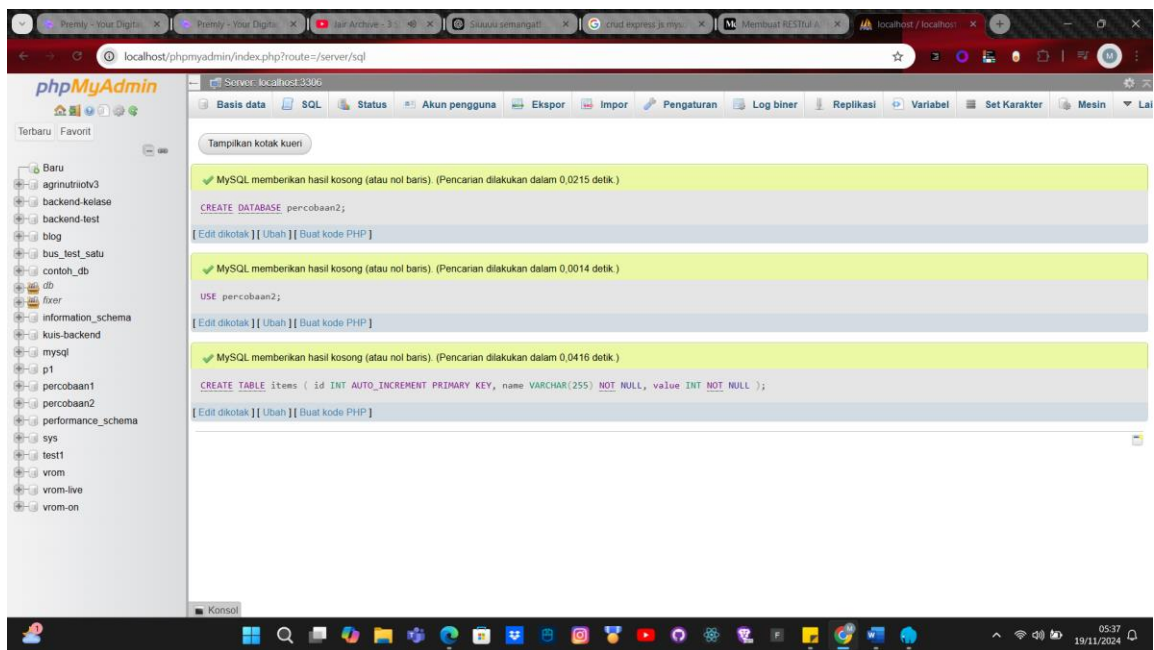
```
CREATE TABLE items (  
  id INT AUTO_INCREMENT PRIMARY KEY,  
  name VARCHAR(255) NOT NULL,  
  value INT NOT NULL  
);
```

Dengan code sql ini :

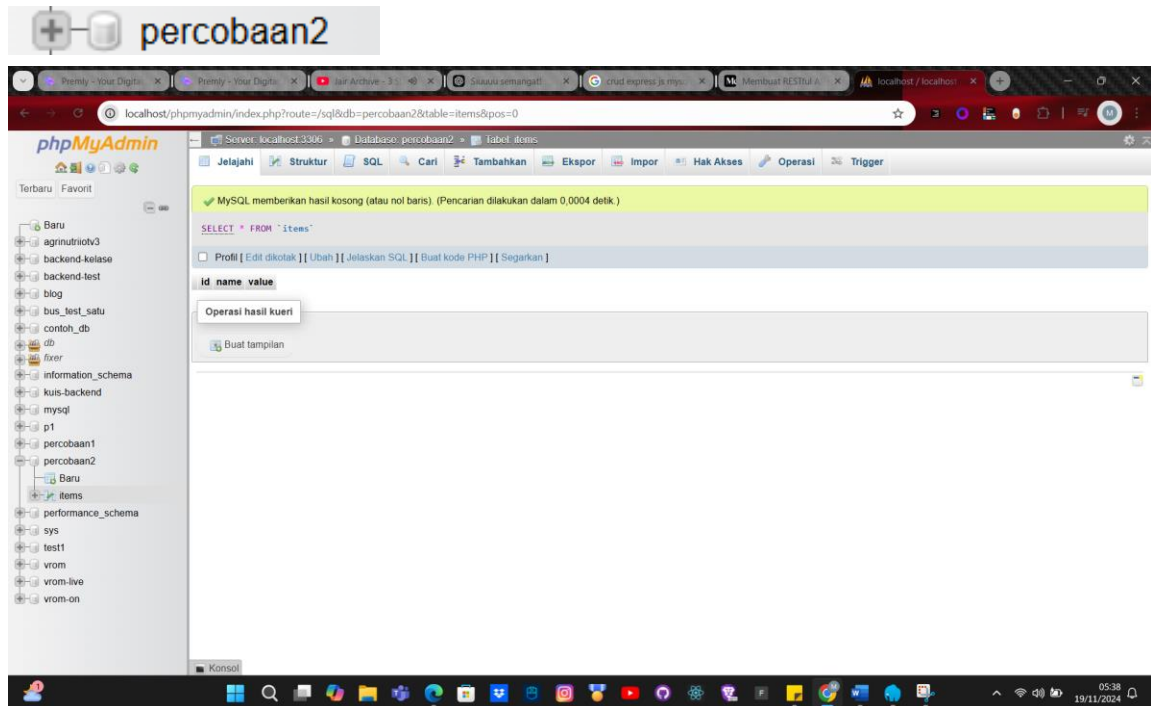




Kemudian klik Kirim



- d) Kemudian di cek apakah database sudah terbuat atau belum, jika sudah maka isinya sementara seperti ini



2. Setelah itu pergi ke framework express yang sudah kalian download kemudian, buka terminal dan install dependensi/package body-parser mysql, dengan command ini :

“ npm install express body-parser mysql “

This screenshot shows the Visual Studio Code interface with a project named 'PE1'. The Explorer sidebar on the left shows the file structure: 'node_modules', 'index.js', 'package-lock.json', and 'package.json'. The main editor displays the 'index.js' file with the following code:

```
1 const express = require('express');
2
3 const app = express();
4
5 app.use() => {
6   console.log('We got request');
7 }
8
9 app.listen(8080, () => {
10   console.log('Server is running on http://localhost:8080');
11 });
```

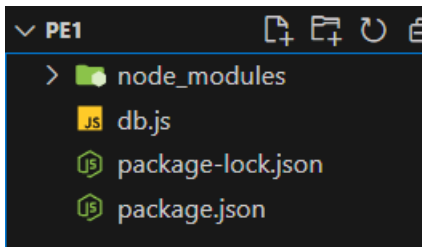
The bottom panel shows the 'TERMINAL' tab with a PowerShell session. The command `npm install express body-parser mysql` has been executed, resulting in the following output:

```
PS D:\water1 asdos\backend\PE1> npm install express body-parser mysql
added 12 packages, and audited 78 packages in 3s
13 packages are looking for funding
  run 'npm fund' for details
found 0 vulnerabilities
PS D:\water1 asdos\backend\PE1>
```

This screenshot shows the same Visual Studio Code interface as the first one, but the terminal is empty. The 'index.js' file in the main editor contains the same code as in the first screenshot:

```
1 const express = require('express');
2
3 const app = express();
4
5 app.use() => {
6   console.log('We got request');
7 }
8
9 app.listen(8080, () => {
10   console.log('Server is running on http://localhost:8080');
11 });
```


3. Kemudian, buka kembali codenya dan lakukanlah Setup Koneksi MySQL dengan cara membuat file dengan nama “db.js”,

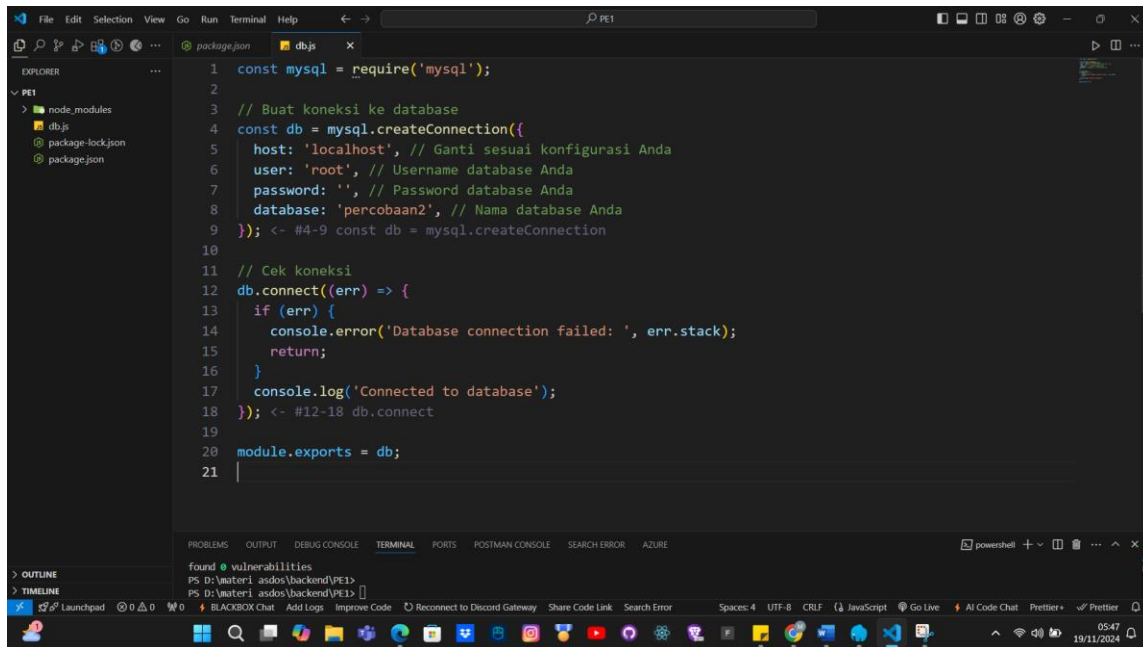


```
const mysql = require('mysql');

// Buat koneksi ke database
const db = mysql.createConnection({
  host: 'localhost', // Ganti sesuai konfigurasi
  Anda
  user: 'root', // Username database Anda
  password: '', // Password database Anda
  database: 'percobaan2', // Nama database Anda
});

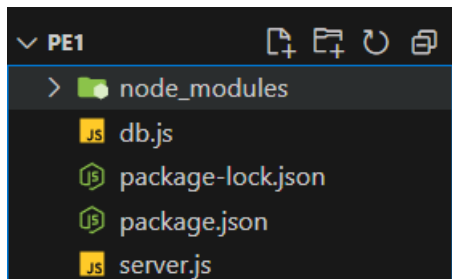
// Cek koneksi
db.connect((err) => {
  if (err) {
    console.error('Database connection failed: ',
err.stack);
    return;
  }
  console.log('Connected to database');
});

module.exports = db;
```



```
1 const mysql = require('mysql');
2
3 // Buat koneksi ke database
4 const db = mysql.createConnection({
5   host: 'localhost', // Ganti sesuai konfigurasi Anda
6   user: 'root', // Username database Anda
7   password: '', // Password database Anda
8   database: 'percobaan2', // Nama database Anda
9 }); <- #4-9 const db = mysql.createConnection
10
11 // Cek koneksi
12 db.connect((err) => {
13   if (err) {
14     console.error('Database connection failed: ', err.stack);
15     return;
16   }
17   console.log('Connected to database');
18 }); <- #12-18 db.connect
19
20 module.exports = db;
21
```

4. Setelah itu buatlah setup servernya, dengan cara membuat file lagi
Bernama server.js, dan diisi dengan code ini



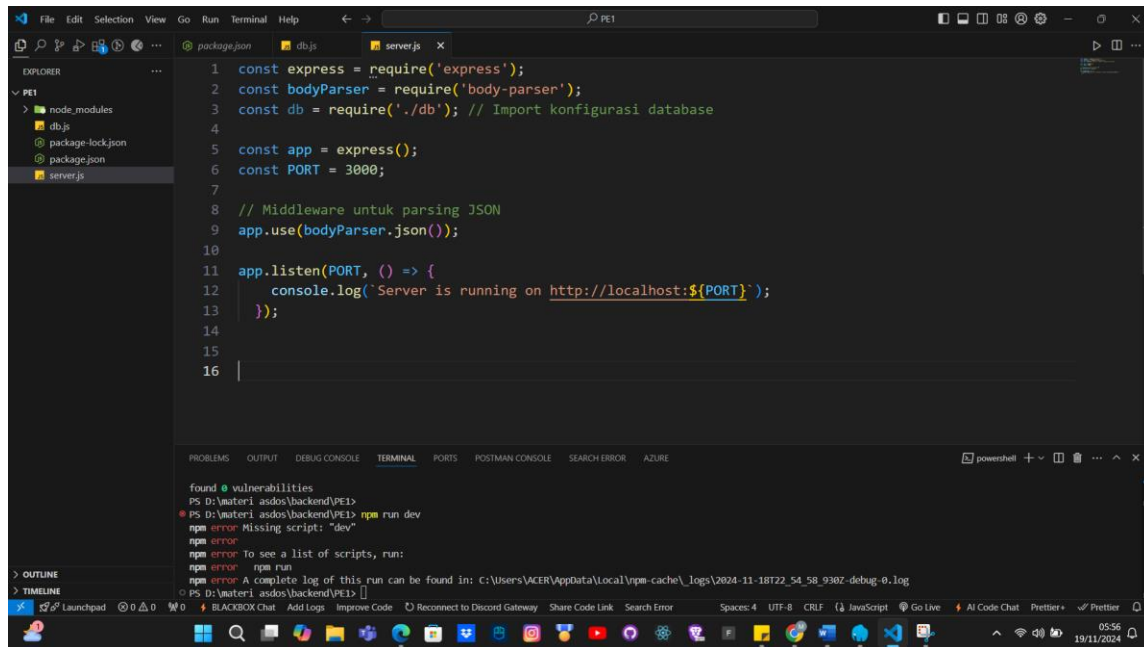
```
const express = require('express');
const bodyParser = require('body-parser');
const db = require('./db'); // Import konfigurasi database

const app = express();
const PORT = 3000;

// Middleware untuk parsing JSON
app.use(bodyParser.json());

app.listen(PORT, () => {
```

```
console.log(`Server is running on http://localhost:${PORT}`);
});
```



5. Kemudian tambahkanlah function create, di server.js :

```
// POST route untuk membuat item baru
app.post('/items', (req, res) => {
  const { name, value } = req.body;

  // Validasi input
  if (!name || !value) {
    return res.status(400).json({ message: 'Name
and Value are required' });
  }

  // Query SQL untuk insert data
```

```

    const query = 'INSERT INTO items (name, value)
VALUES (?, ?)';
    db.query(query, [name, value], (err, result) =>
{
    if (err) {
        console.error('Error inserting data: ',
err);
        return res.status(500).json({ message:
'Database error' });
    }

    res.status(201).json({
        message: 'Item created successfully',
        item: { id: result.insertId, name, value },
    });
});
});
});

```

The screenshot shows a VS Code editor with a project named 'PE1'. The file explorer on the left shows the project structure, including 'node_modules', 'db.js', 'package-lock.json', 'package.json', and 'server.js'. The code editor displays the 'server.js' file, which contains the following code:

```

1  const express = require('express');
2  const bodyParser = require('body-parser');
3  const db = require('./db'); // Import konfigurasi database
4
5  const app = express();
6  const PORT = 3000;
7
8  // Middleware untuk parsing JSON
9  app.use(bodyParser.json());
10
11 app.listen(PORT, () => {
12   console.log('Server is running on http://localhost:${PORT}');
13 });
14
15 // POST route untuk membuat item baru
16 app.post('/items', (req, res) => {
17   const { name, value } = req.body;
18
19   // Validasi input
20   if (!name || !value) {
21     return res.status(400).json({ message: 'Name and Value are required' });
22   }
23
24   // Query SQL untuk insert data
25   const query = 'INSERT INTO items (name, value) VALUES (?, ?)';
26   db.query(query, [name, value], (err, result) => {
27     if (err) {
28       console.error('Error inserting data: ', err);
29       return res.status(500).json({ message: 'Database error' });
30     }
31
32     res.status(201).json({
33       message: 'Item created successfully',
34       item: { id: result.insertId, name, value },
35     });
36   });
37   // < #27-37 db.query
38 }); // < #17-38 app.post
39

```

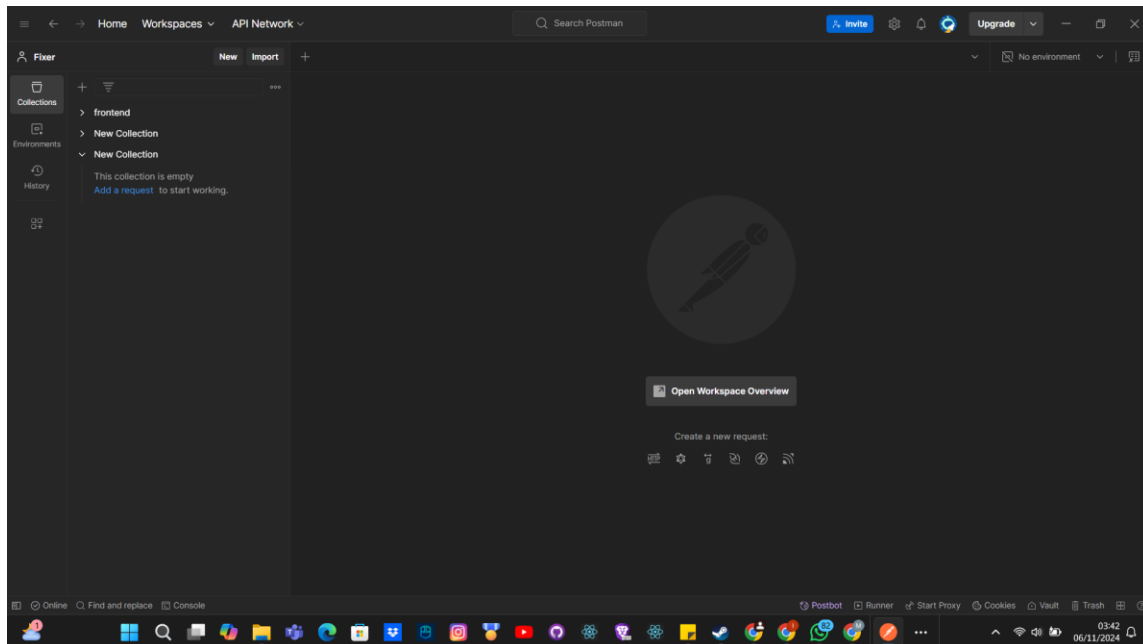
The status bar at the bottom shows the current file is 'server.js', the encoding is 'UTF-8', and the line endings are 'CRLF'. The date and time are 19/11/2024, 05:57.

6. Kemudian buka kembali terminal dan nyalakan projectnya dengan
comman :
node server.js



```
PS D:\materi asdos\backend\PE1> node server.js
Server is running on http://localhost:3000
Connected to database
```

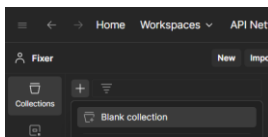
7. Kemudian buka postman, setelah itu buatlah collectionnya dengan nama
project yang kalian buat.



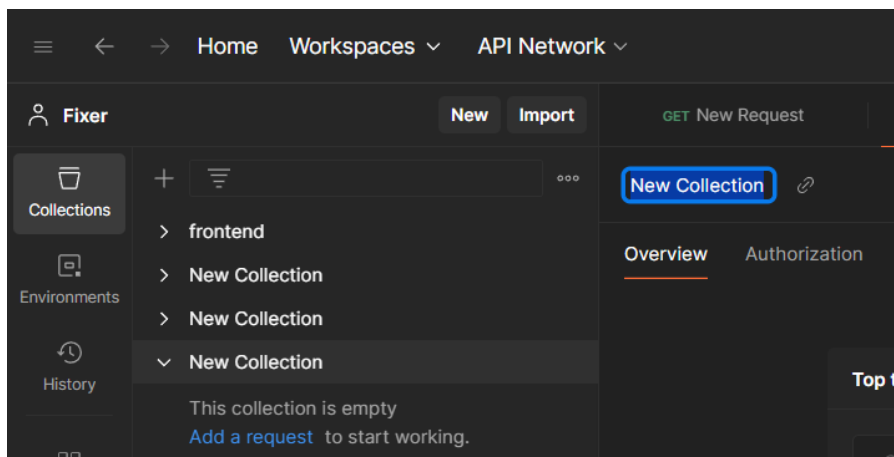
Klik tombol tambah



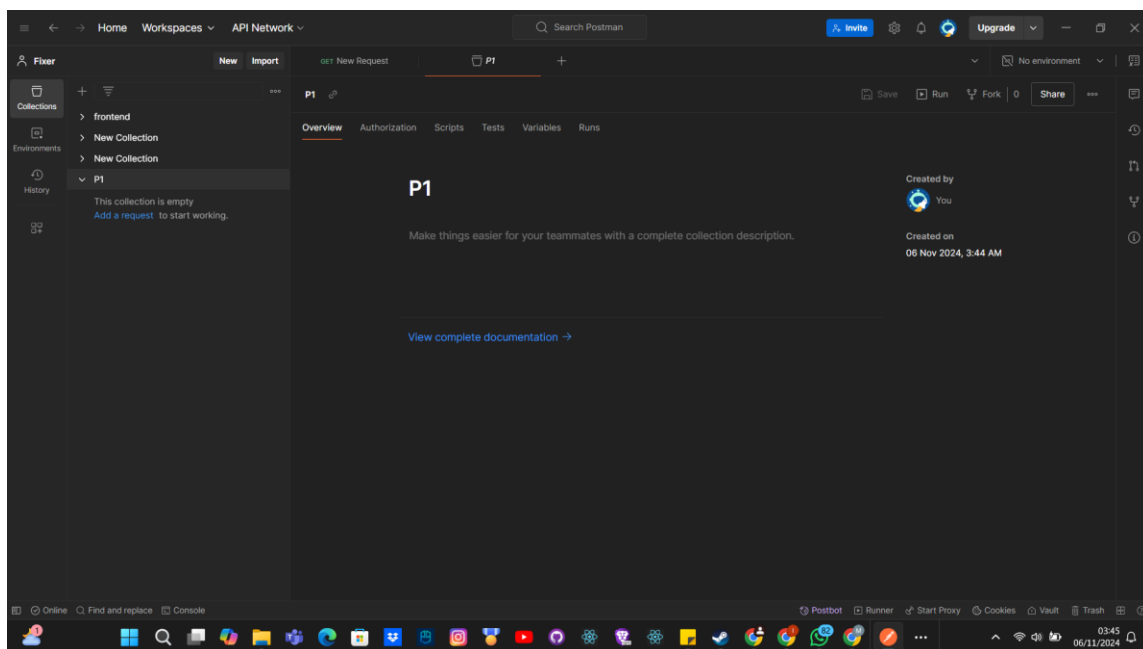
Kemudian pilih yang blank collection



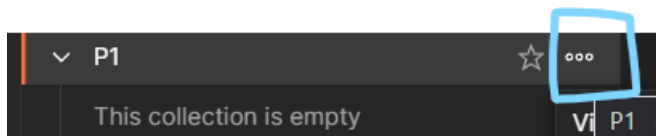
Setelah itu jangan lupa untuk melakukan rename namanya dengan project kalian, dengan klik new collection (yang ditandai biru)



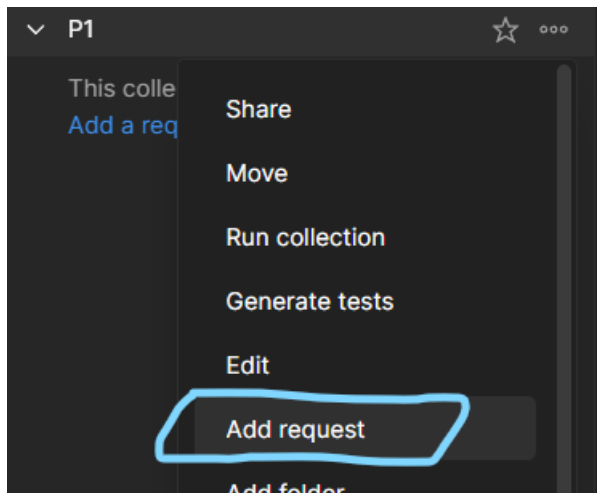
Contoh :



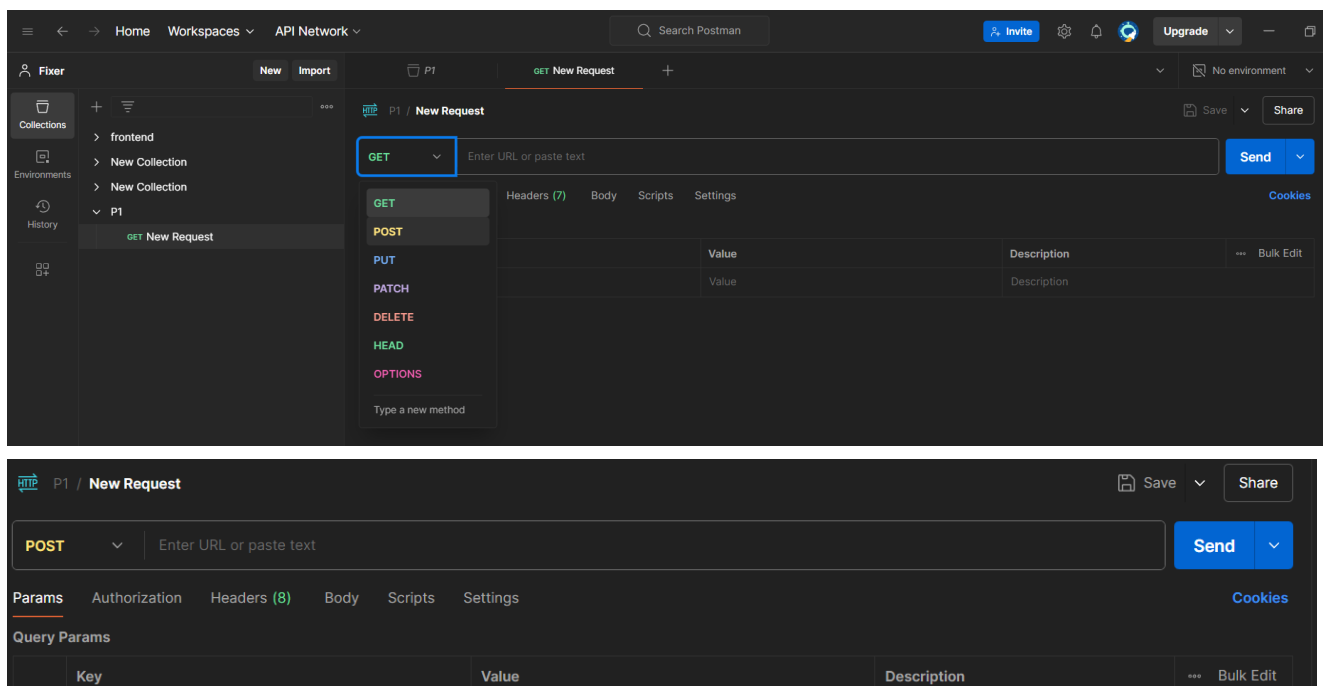
8. Setelah terbuat, klik titik tiga yang berada di sebelah nama project



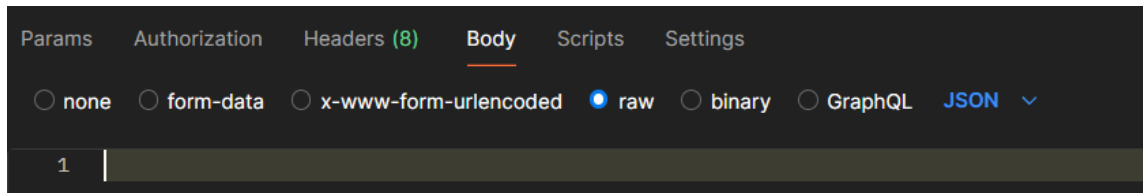
9. Kemudian klik add request



10. Setelah itu gantilah requestnya dari sebelumnya Get menjadi Post

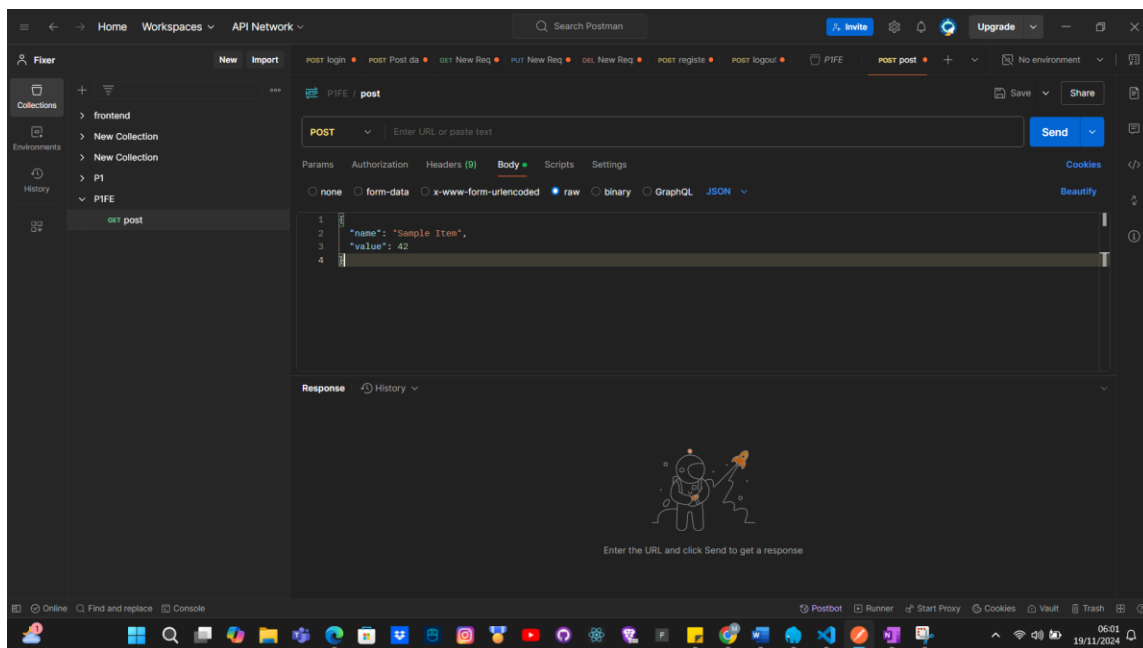
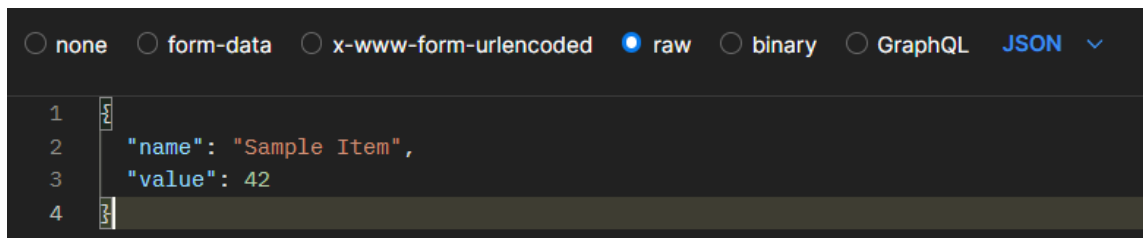


11. Setelah itu masukkan lah contoh inputannya dalam format json ini
dengan cara klik dibawah nya url, pilih yang Body, kemudian formatnya
pilih yang json



12. Setelah itu isilah dengan format data yang sesuai dengan controllernya :
Contoh :

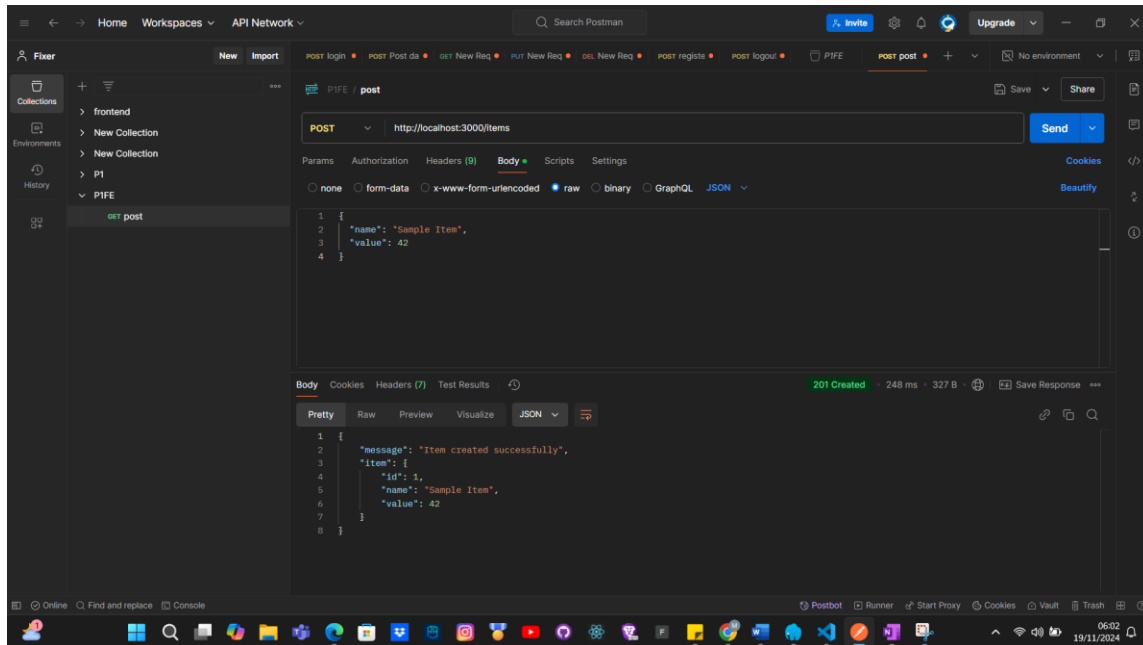
```
{  
  "name": "Sample Item",  
  "value": 42  
}
```



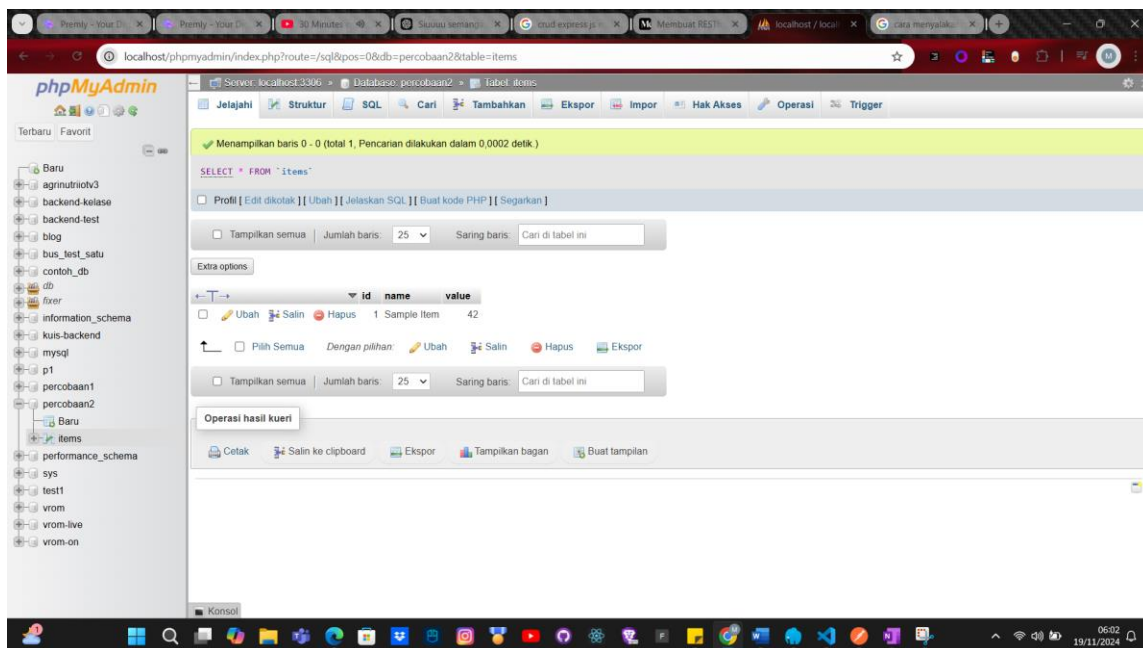
13. Setelah itu isilah url atasnya sesuai dengan routes yang dibuat :



14. Kemudian klik send dan hasilnya seperti ini



15. Kemudian cek di database, apakah sudah terbuat atau belum



2. TUGAS

Jawablah Pertanyaan dari Soal ini |

1. Cobalah untuk melakukan semua percobaan diatas
2. Kemudian buatlah versi project kalian sendiri
3. minimal sudah 3 table (boleh pakai table dari framework sebelumnya)
yang bisa melakukan proses create

PENGUMPULAN TUGAS DAN DEADLINE

- a. Pengumpulan Tugas di Google Drive
- b. Deadline = 19 November 2024 | 23.00 WIB

DAFTAR PUSTAKA

<https://medium.com/@kevinffa0107/pengertian-mvc-model-view-controller-pada-framework-laravel-20f261ccf233>

<https://medium.com/chevalier-lab/membuat-restful-api-node-js-express-mysql-crud-c4a1512600b6>

<https://santrikoding.com/tutorial-set/tutorial-nodejs-express-mysql>

<https://expressjs.com/>