

VERSION 4.0  
AGUSTUS, 24, 2023



# PEMROGRAMAN WEB

## PHP REST API DAN MYSQL DATABASE (BACKEND) – MODUL 5

DISUSUN OLEH:  
M. SYAUQI AMIQ AMRULLAH  
ALIF FATWA RAMADHANI

DIAUDIT OLEH:  
AMINUDIN, S.KOM., M.CS.

LAB. INFORMATIKA  
UNIVERSITAS MUHAMMADIYAH MALANG

## PEMROGRAMAN WEB

---

### PERSIAPAN MATERI

- [https://www.w3schools.com/php/php\\_oop\\_what\\_is.asp](https://www.w3schools.com/php/php_oop_what_is.asp)
  - <https://www.w3schools.com/php/default.asp>
- 

### TUJUAN

Mahasiswa mampu menggunakan MySQL dan melakukan komunikasi database dengan PHP, Modul ini telah disesuaikan dengan beberapa poin penilaian Materi Uji Kompetensi.

### PERSIAPAN SOFTWARE/APLIKASI

#### Hardware dan Infrastruktur

- Laptop/PC
- Koneksi Internet

#### Software

- Text Editor: Visual Studio Code (Recomended)
- Extension VSCode: PHP Intelephense (Code Formatter agar Rapih)
- PHP Versi 8.1 atau lebih
- Chrome Extension: JSON Viewer
- MySQL 8.0+

#### Settings:

- Default Formatter pada VSCode di setting dengan extension PHP Intelephense

---

### NOTES

**Silahkan mencari tutorial mengenai cara penginstalan MySQL pada device kalian masing – masing atau jika kalian sudah menginstall XAMPP silahkan aktifkan MySQL Database dan Apache Server nya.**

**Untuk di MacOS silahkan di explore untuk penginstalan MySQL Server nya, bisa menggunakan XAMPP seperti di Windows.**

---

## MATERI

### MYSQL

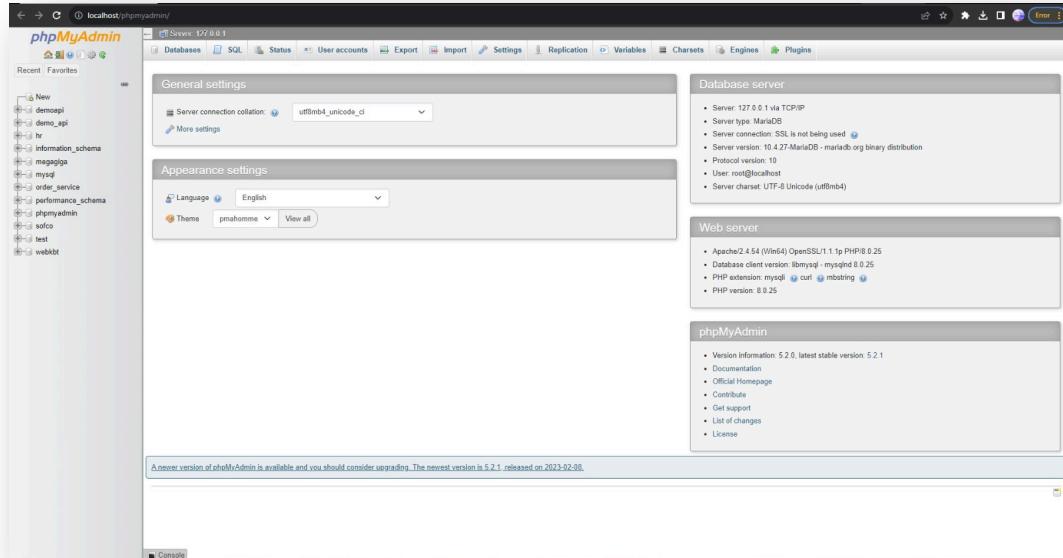
MySQL merupakan sistem manajemen database yang bersifat open-source yang menggunakan perintah dasar atau bahasa pemrograman yang berupa structured query language (SQL) yang cukup populer di dunia teknologi. MySQL berguna sebagai database.

#### A. Membuat Database dan Tabel dengan phpMyAdmin

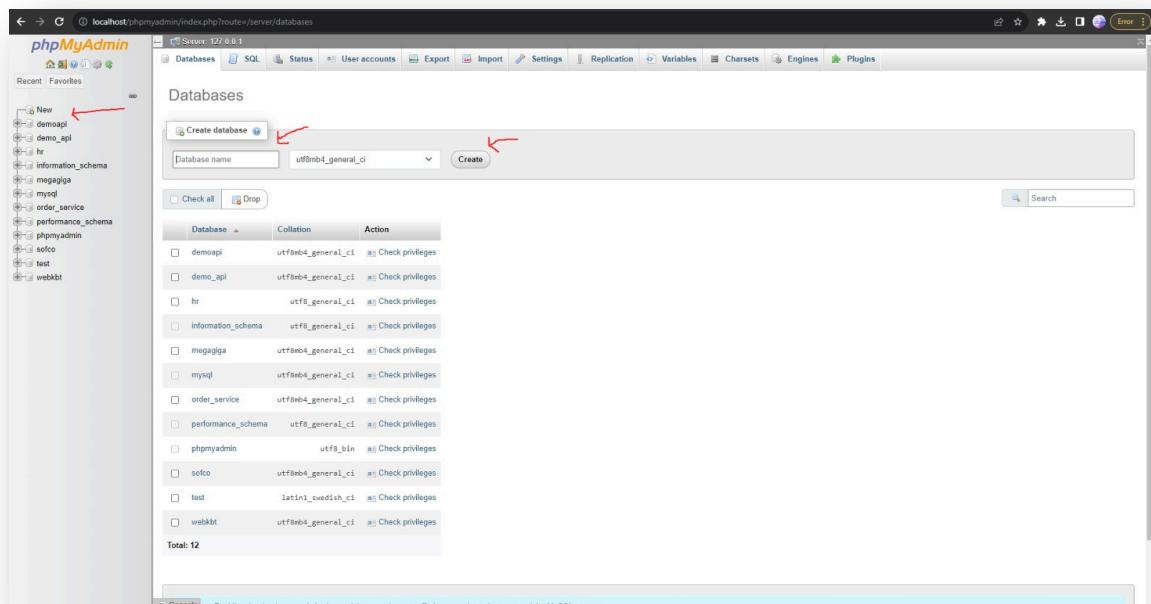
Sebelum membuat database, pastikan kalian sudah menginstall MySQL pada device kalian dan jalankan MySQL Server kalian, jika menggunakan XAMPP pastikan status MySQL sudah berjalan seperti pada gambar berikut:



Setelah itu klik tombol “Admin” pada bagian Module MySQL dan akan muncul tampilan berbasis web seperti berikut:

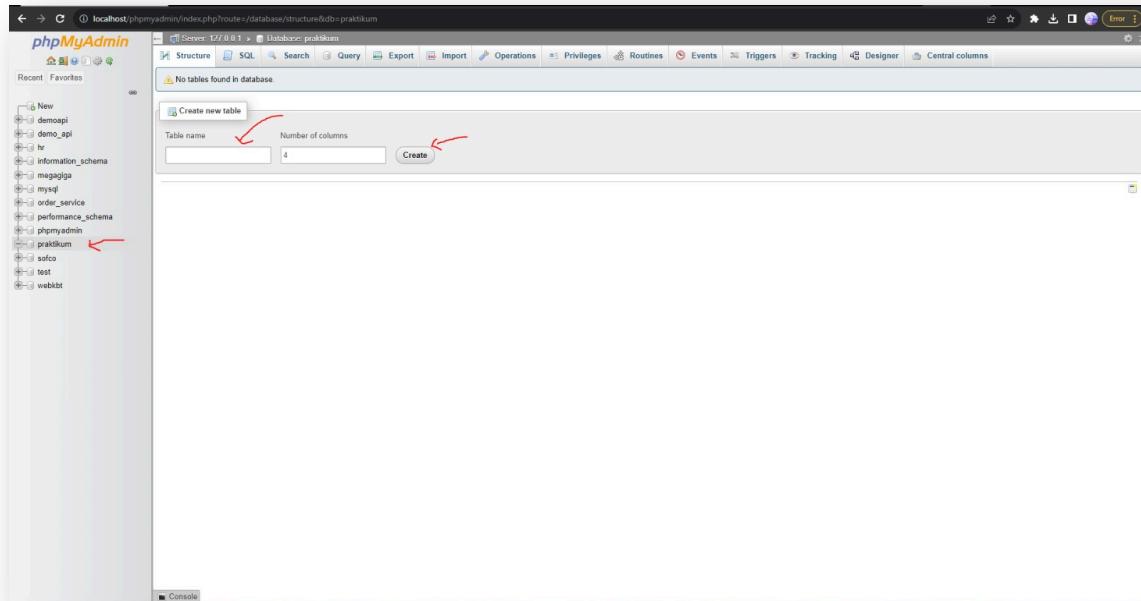


Selanjutnya adalah membuat sebuah database baru seperti pada gambar dibawah:

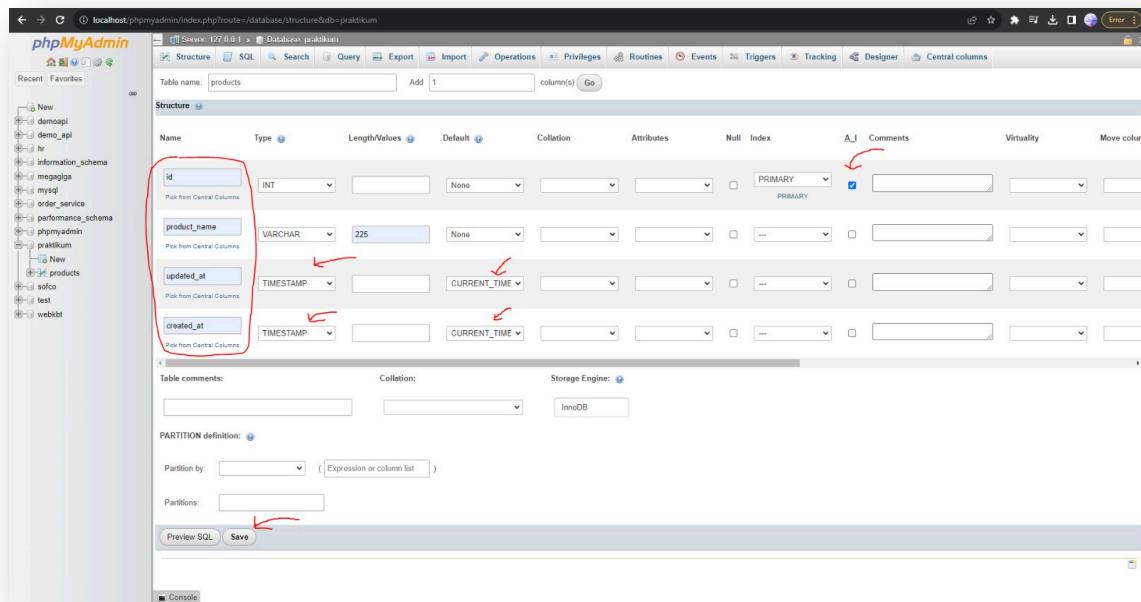


Mengacu pada Gambar diatas silahkan klik “New” pada sidebar kiri lalu pada halaman Database silahkan membuat database baru dengan cara input “Database Name” dan klik tombol “Create” sesuai petunjuk tanda panah yang diberikan.

Setelah membuat database, langkah selanjutnya yaitu silahkan untuk membuat sebuah table seperti pada gambar dibawah:



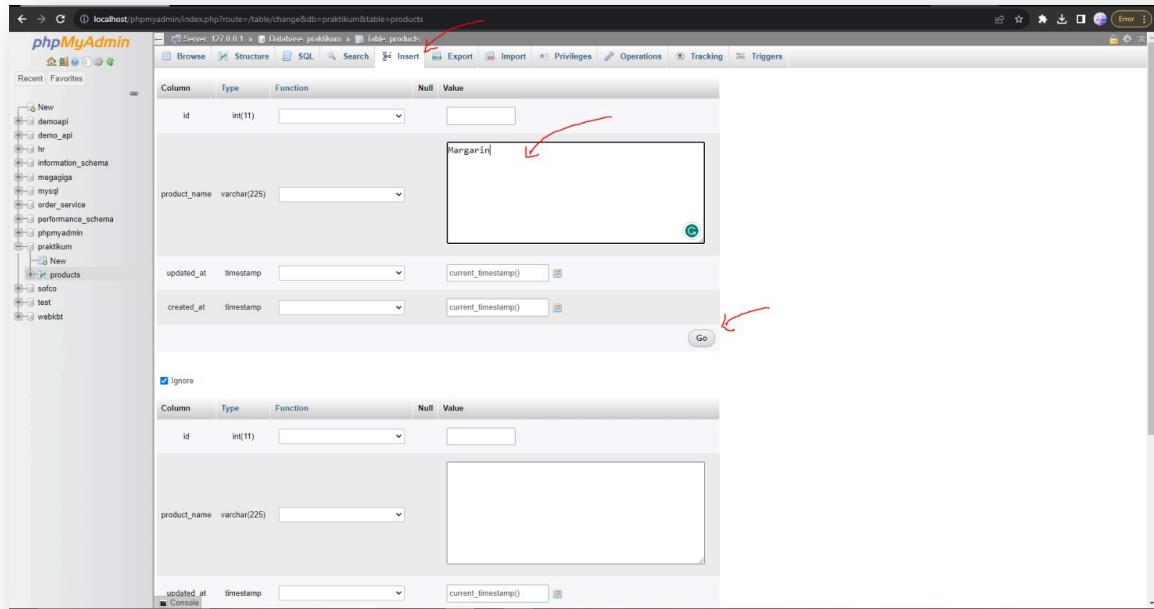
Mengacu pada Gambar diatas silahkan kalian klik database yang telah dibuat pada sidebar kiri, lalu pada menu “Structure” silahkan kalian membuat table dengan nama “products” dengan jumlah column 4 lalu klik “Create” maka akan muncul menu berikut:



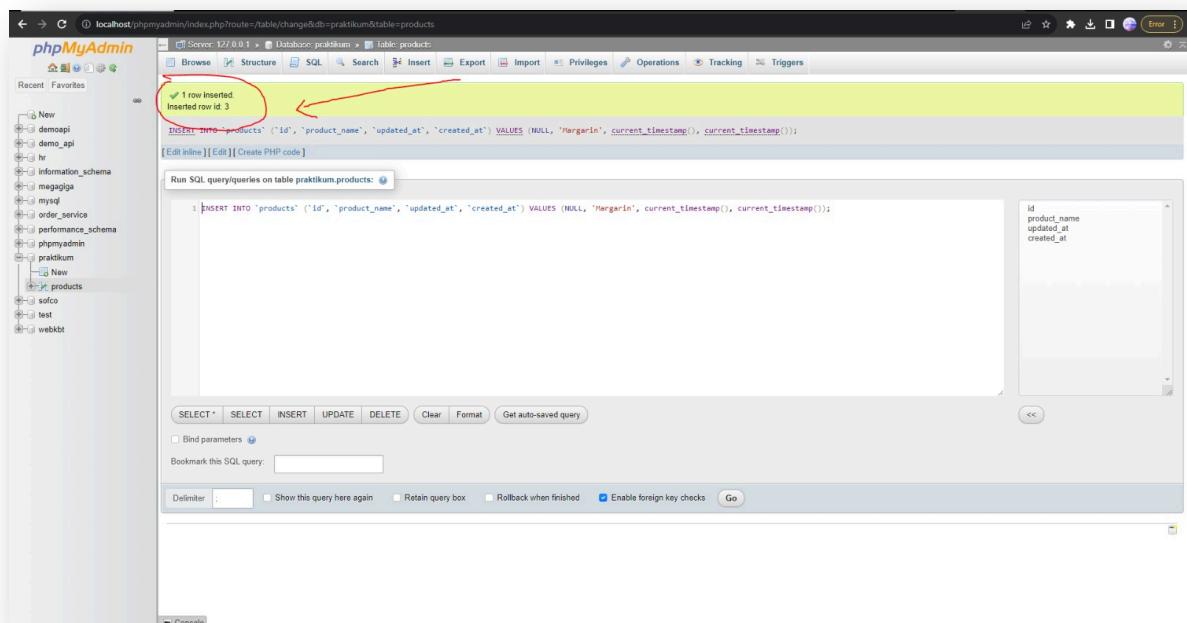
Silahkan memasukan attribute seperti pada gambar, terdapat 4 column dengan field “id”, “product\_name”, “updated\_at”, dan “created\_at”. Untuk column “id” silahkan centang pada bagian A\_1 yang menandakan field tersebut Auto Increment, lalu jangan lupa setting Type untuk tipe datanya dan default value untuk field “updated\_at” dan “created\_at”, setelah itu klik “save”.

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	<b>id</b>	int(11)	utf8mb4_general_ci		No	None	AUTO_INCREMENT		<b>Change</b> <b>Drop</b> <b>More</b>
2	<b>product_name</b>	varchar(225)	utf8mb4_general_ci		No	None			<b>Change</b> <b>Drop</b> <b>More</b>
3	<b>updated_at</b>	timestamp			No	current_timestamp()			<b>Change</b> <b>Drop</b> <b>More</b>
4	<b>created_at</b>	timestamp			No	current_timestamp()			<b>Change</b> <b>Drop</b> <b>More</b>

Setelah itu untuk memastikan table dan field nya telah terbuat silahkan klik table “products” yang dapat diakses pada sidebar kiri dan masuk ke menu “Structure”, seharusnya struktur table akan terlihat seperti pada gambar.



Langkah selanjutnya adalah klik menu “Insert” lalu cukup isikan pada field “product\_name” dengan value yang kalian inginkan, lalu klik “Save”. Maka akan muncul pesan seperti berikut:



Setelah itu silahkan klik menu “Browse” dan akan muncul seluruh data yang sudah kalian buat seperti pada gambar berikut:

The screenshot shows the phpMyAdmin interface with the database 'praktikum' selected. Under the 'Tables' section, 'products' is selected. The 'Browse' tab is active, displaying the contents of the 'products' table. A single row is highlighted with a red circle, showing the columns: id, product\_name, updated\_at, and created\_at. The row details are: id=3, product\_name='Margarin', updated\_at='2023-08-23 20:02:43', and created\_at='2023-08-23 20:02:43'. Below the table, there are buttons for Edit, Copy, Delete, and Export.

Ulangi langkah tersebut untuk membuat minimal 5 data “products”.

Selamat, kalian sudah membuat database yang siap untuk diintegrasikan melalui PHP, selanjutnya kita akan membuat Backend Web Service yang nantinya akan menghasilkan sebuah API yang akan menyajikan data dari database kita dengan format JSON, namun sebelumnya kita akan mempelajari konsep RESTful API terlebih dahulu dan menginstall Postman sebagai tools untuk testing RESTful API kita.

## HTTP DAN POSTMAN

### A. REST API

API merupakan singkatan dari application programming interfaces, yang secara singkatnya API bisa digunakan oleh developer untuk mengizinkan dan mengintegrasikan aplikasi yang berada di platform yang berbeda ataupun perangkat yang berbeda untuk bisa saling terkoneksi antara satu dengan yang lainnya.

Tujuan utama kenapa API ini digunakan oleh developer dalam pembuatan sebuah perangkat lunak adalah untuk saling berbagi data antara aplikasi yang sama dari platform yang sama ataupun berbeda. Tentu saja penggunaan API tidak hanya sesederhana itu, API juga bisa digunakan untuk menyediakan function sehingga para developer tidak perlu membuat fungsi lagi dalam coding nya hanya perlu memanggilnya saja.

Sedangkan **REST** merupakan kependekan dari **Representational State Transfer**. REST merupakan sebuah web service yang berjalan di client dan server dimana yang memiliki sifat stateless. Yang mempunyai arti bahwa setiap request yang dikirim oleh aplikasi maka harus menyertakan semua parameter dan datanya dengan lengkap.

Secara singkat cara kerjanya adalah client REST (aplikasinya) akan menjalankan request ke server REST, ketika request terkirim ke server selanjutnya REST server akan memberikan respon. Kemudian client REST akan menampilkan responnya atau jika tidak, client akan melakukan pemrosesan yang kain.

Respon dari REST Server ke REST client dapat diberikan dalam berbagai format, yaitu ada HTML, XML, Json, atau beberapa format yang lain. Namun format yang paling sering dipakai oleh para developer adalah Json, karena formatnya lebih mudah untuk digunakan dan dipelajari.

Untuk bisa menggunakan REST ini, REST memiliki sebuah standarisasi yang harus dipakai dalam implementasinya, standarisasi REST ini menggunakan URL dan juga HTTP verb. Dengan menggunakan standarisasi yang menggunakan URL maka developer dapat melakukan beberapa operasi berdasarkan HTTP verbs. Secara teknis HTTP verbs ini berupa `$_SERVER['REQUEST_METHOD']`.

**Ada 5 method request HTTP yang sangat umum digunakan pada arsitektur REST ini, yaitu :**

<b>GET</b>	Digunakan untuk mengakses data atau membaca data yang ada pada resource.
------------	--

<b>POST</b>	Digunakan untuk men-create atau membuat sebuah resource baru.
<b>PUT</b>	Digunakan untuk memperbaharui sebuah resource, atau menambahnya.
<b>PATCH</b>	Digunakan untuk mengupdate kumpulan data (field) yang ada di dalam resource secara partial.
<b>DELETE</b>	Digunakan untuk menghapus resource.

Selain itu terdapat juga HTTP Status Code, yang merupakan jawaban server atas permintaan data oleh client. Jawaban ini berbentuk kode tiga digit, yang memiliki arti di setiap masing-masing status code yang dikembalikan.

<b>1xx - Information Responses</b>	HTTP request berfungsi untuk membaca dan menjelaskan permintaan pesan yang dikirimkan oleh client. Permintaan atau request yang dikirimkan melalui aplikasi web browser seperti Mozilla Firefox, Google Chrome, Internet Explorer, dan lainnya.
<b>2xx - Successful Responses</b>	Kode status ini menunjukkan tindakan yang diminta oleh klien telah diterima, dipahami, diterima dan diproses dengan sukses.
<b>3xx - Redirection Messages</b>	Kode status ini menunjukkan bahwa tindakan lebih lanjut perlu dilakukan oleh agen pengguna untuk memenuhi permintaan.

<b>4xx - Client Error Responses</b>	Kode status ini menunjukkan bahwa ada kemungkinan kesalahan dalam permintaan yang mencegah server untuk memprosesnya.
<b>5xx - Server Error Responses</b>	Kode status ini menunjukkan bahwa server mengalami galat/error internal saat mencoba untuk memproses permintaan tersebut. Kesalahan ini cenderung dari server sendiri, tidak berkaitan dengan permintaan.

## B. POSTMAN

Postman adalah developing tools yang membantu penggunanya untuk membangun, menguji, dan memodifikasi API. Ia menawarkan para developer berbagai fitur dan fungsi yang penting sehingga kinerjanya dapat berlangsung mudah dan sederhana.

Untuk menginstall Postman silahkan download Postman pada link berikut dan silahkan melakukan instalasi secara mandiri:

<https://www.postman.com/downloads/>

---

## CODELAB

### LATIHAN

Pada kegiatan codelab kalo ini kita akan melakukan sebuah interaksi dengan MySQL Database menggunakan PHP, dan kita akan membuat 4 Endpoint REST API yang dapat melakukan aksi Create, Read, Update, Delete.

**Rincian endpoint yang kita buat adalah sebagai berikut:**

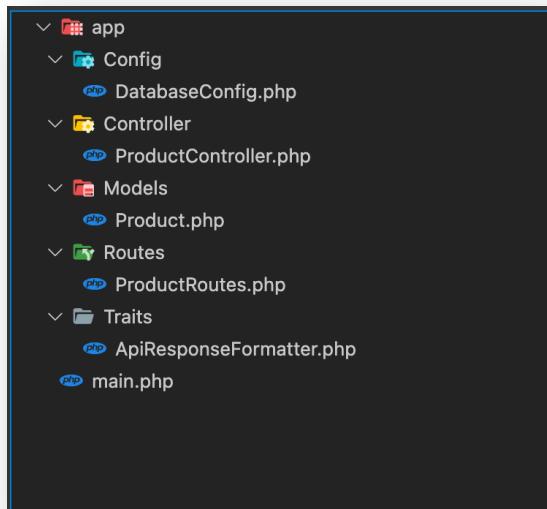
GET	http://127.0.0.1:8000/api/product
-----	-----------------------------------

<b>GET</b>	http://127.0.0.1:8000/api/product/{id}
<b>POST</b>	http://127.0.0.1:8000/api/product
<b>PUT</b>	http://127.0.0.1:8000/api/product
<b>DELETE</b>	http://127.0.0.1:8000/api/product

**Sebelum membuat kode pastikan kalian telah menyalakan MySQL kalian dan membuat database serta tabelnya seperti contoh pada materi diatas. Kode pada codelab ini berdasarkan database dan table pada contoh materi praktikum.**

Mari kita membuat kode nya, silahkan buat sebuah folder project dengan nama sembarang yang berisi folder dan file seperti berikut dan buka di vscode:

#### Struktur Folder:



Pada file **DatabaseConfig.php**:



```
1 <?php
2
3 namespace app\Config;
4
5 class DatabaseConfig
6 {
7     // SETTING KONFIGURASI DATABASE KALIAN
8     public $host = "localhost";
9     public $user = "root";
10    public $password = "";
11    public $database_name = "praktikum";
12    public $port = 3306;
13 }
14
```

Pastikan value dari setiap variable disesuaikan dengan settingan MySQL kalian. File ini digunakan untuk mendefinisikan konfigurasi database kita.

Pada file **ApiResponseFormatter.php**:



```
1 <?php
2
3 namespace app\Traits;
4
5 // UNTUK FORMATTING RESPONSE
6 trait ApiResponseFormatter
7 {
8     public function apiResponse($code = 200, $message = "success", $data = [])
9     {
10         // DARI PARAMETER AKAN DI FORMAT MENJADI SEPERTI DIBAWAH INI
11         return json_encode([
12             "code" => $code,
13             "message" => $message,
14             "data" => $data
15         ]);
16     }
17 }
18
```

Pada file ApiResponseFormatter.php terdapat Trait yang berfungsi untuk melakukan formatting terhadap response JSON kita nantinya.

Pada file **Product.php**:

```
● ● ●
1 <?php
2
3 namespace app\Models;
4
5 include "app\Config\DatabaseConfig.php";
6
7
8 use app\Config\DatabaseConfig;
9 use mysqli;
10
11 class Product extends DatabaseConfig
12 {
13
14     public $conn;
15
16     public function __construct()
17     {
18         // CONNECT KE DATABASE MYSQL
19         $this->conn = new mysqli($this->host, $this->user, $this->password, $this->database_name, $this->port);
20         // Check connection
21         if ($this->conn->connect_error) {
22             die("Connection failed: " . $this->conn->connect_error);
23         }
24     }
25
26     // PROSES MENAMPILKAN SEMUA DATA
27     public function findAll()
28     {
29         $sql = "SELECT * FROM products";
30         $result = $this->conn->query($sql);
31         $this->conn->close();
32         $data = [];
33         while ($row = $result->fetch_assoc()) {
34             $data[] = $row;
35         }
36
37         return $data;
38     }
39 }
```

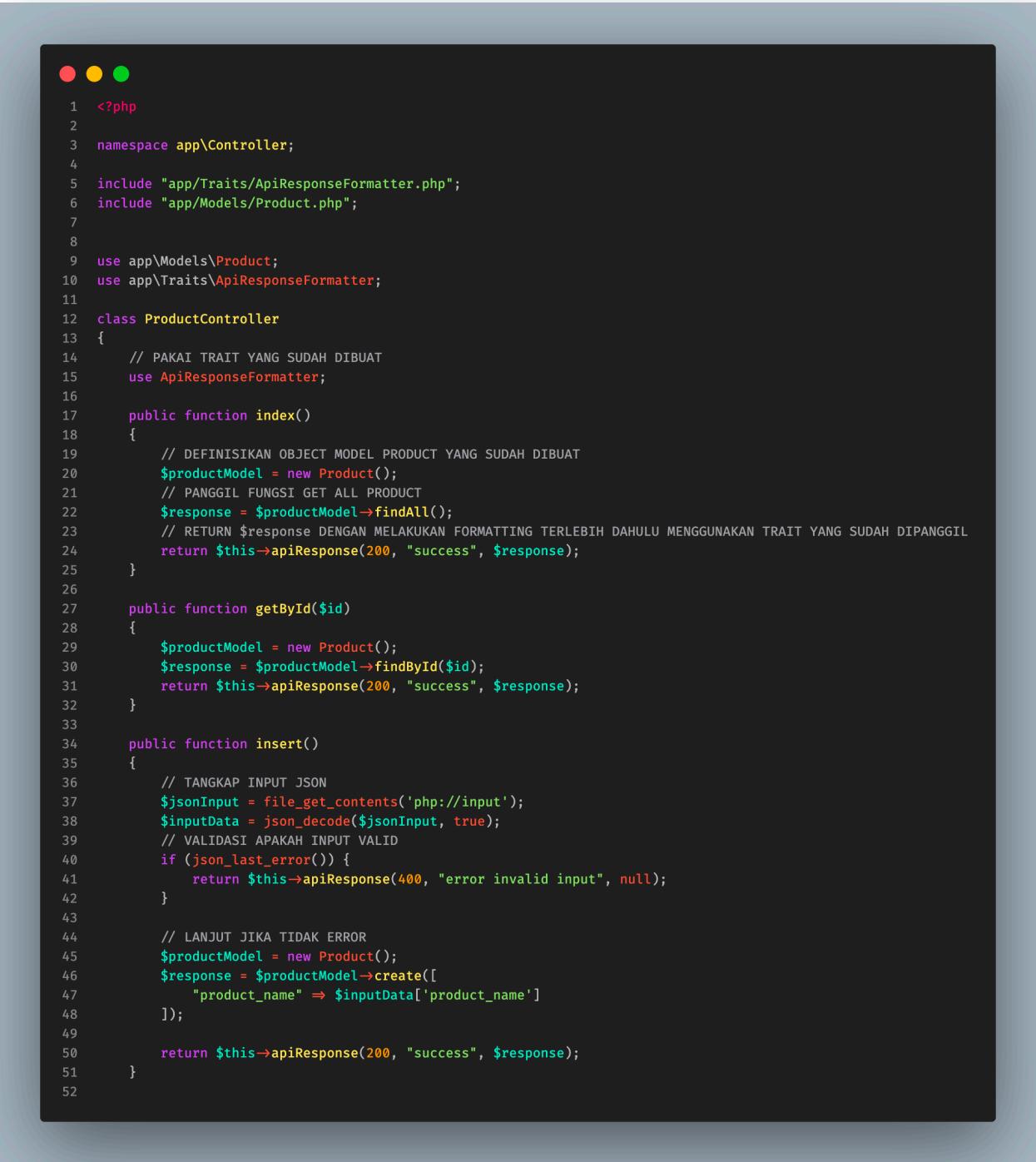
```

1 // PROSES MENAMPAILAN DATA DENGAN ID
2 public function findById($id)
3 {
4     $sql = "SELECT * FROM products WHERE id = ?";
5     $stmt = $this->conn->prepare($sql);
6     $stmt->bind_param("i", $id);
7     $stmt->execute();
8     $result = $stmt->get_result();
9     $this->conn->close();
10    $data = [];
11    while ($row = $result->fetch_assoc()) {
12        $data[] = $row;
13    }
14
15    return $data;
16 }
17
18
19 // PROSES INSERT DATA
20 public function create($data)
21 {
22     $productName = $data['product_name'];
23     $query = "INSERT INTO products (product_name) VALUES (?)";
24     $stmt = $this->conn->prepare($query);
25     $stmt->bind_param("s", $productName);
26     $stmt->execute();
27     $this->conn->close();
28 }
29
30 // PROSES UPDATE DATA DENGAN ID
31 public function update($data, $id)
32 {
33     $productName = $data['product_name'];
34
35     $query = "UPDATE products SET product_name = ? WHERE id = ?";
36     $stmt = $this->conn->prepare($query);
37     // huruf "s" berarti tipe parameter product_name adalah String dan huruf "i" berarti parameter id adalah integer
38     $stmt->bind_param("si", $productName, $id);
39     $stmt->execute();
40     $this->conn->close();
41 }
42
43 // PROSES DELETE DATA DENGAN ID
44 public function destroy($id)
45 {
46     $query = "DELETE FROM products WHERE id = ?";
47     $stmt = $this->conn->prepare($query);
48     // huruf "i" berarti parameter pertama adalah integer
49     $stmt->bind_param("i", $id);
50     $stmt->execute();
51     $this->conn->close();
52 }
53 }
54

```

Pada file Product.php bisa dikatakan adalah sebuah Model yang nantinya kita akan berinteraksi dengan database kita melalui file ini.

Pada file **ProductController.php**:

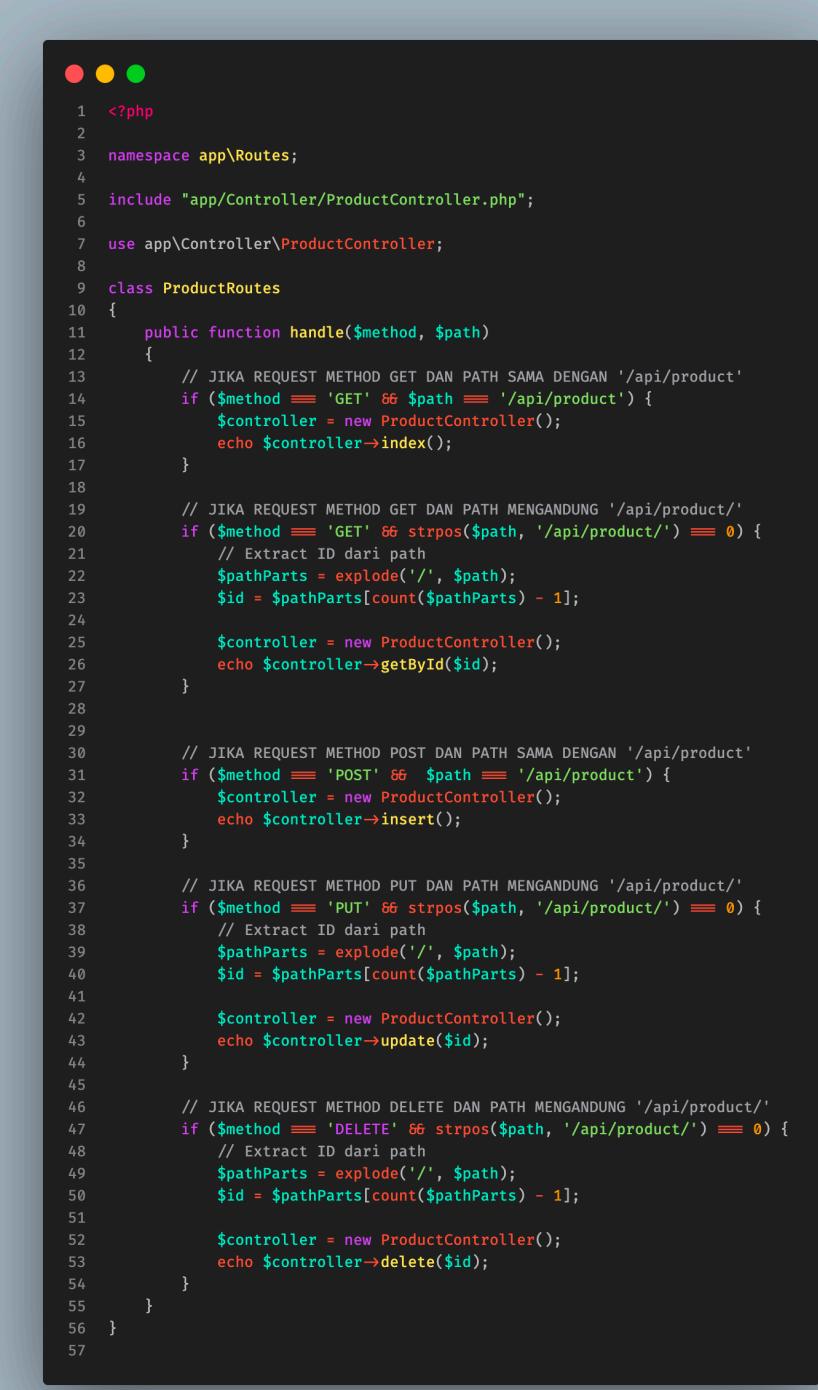


```
1 <?php
2
3 namespace app\Controller;
4
5 include "app/Traits/ApiResponseFormatter.php";
6 include "app/Models/Product.php";
7
8
9 use app\Models\Product;
10 use app\traits\ApiResponseFormatter;
11
12 class ProductController
13 {
14     // PAKAI TRAIT YANG SUDAH DIBUAT
15     use ApiResponseFormatter;
16
17     public function index()
18     {
19         // DEFINISIKAN OBJECT MODEL PRODUCT YANG SUDAH DIBUAT
20         $productModel = new Product();
21         // PANGGIL FUNGSI GET ALL PRODUCT
22         $response = $productModel->findAll();
23         // RETURN $response DENGAN MELAKUKAN FORMATTING TERLEBIH DAHULU MENGGUNAKAN TRAIT YANG SUDAH DIPANGGIL
24         return $this->apiResponse(200, "success", $response);
25     }
26
27     public function getById($id)
28     {
29         $productModel = new Product();
30         $response = $productModel->findById($id);
31         return $this->apiResponse(200, "success", $response);
32     }
33
34     public function insert()
35     {
36         // TANGKAP INPUT JSON
37         $jsonInput = file_get_contents('php://input');
38         $inputData = json_decode($jsonInput, true);
39         // VALIDASI APAKAH INPUT VALID
40         if (json_last_error()) {
41             return $this->apiResponse(400, "error invalid input", null);
42         }
43
44         // LANJUT JIKA TIDAK ERROR
45         $productModel = new Product();
46         $response = $productModel->create([
47             "product_name" => $inputData['product_name']
48         ]);
49
50         return $this->apiResponse(200, "success", $response);
51     }
52 }
```

```
● ● ●  
1  public function update($id)  
2  {  
3      // TANGKAP INPUT JSON  
4      $jsonInput = file_get_contents('php://input');  
5      $inputData = json_decode($jsonInput, true);  
6      // VALIDASI APAKAH INPUT VALID  
7      if (json_last_error()) {  
8          return $this->apiResponse(400, "error invalid input", null);  
9      }  
10     // LANJUT JIKA TIDAK ERROR  
11     $productModel = new Product();  
12     $response = $productModel->update([  
13         "product_name" => $inputData['product_name']  
14     ], $id);  
15     return $this->apiResponse(200, "success", $response);  
16 }  
17  
18 public function delete($id)  
19 {  
20  
21     $productModel = new Product();  
22     $response = $productModel->destroy($id);  
23  
24     return $this->apiResponse(200, "success", $response);  
25 }  
26 }  
27 }  
28 }
```

Pada file ProductController.php bisa disebut sebuah Controller dimana berfungsi untuk memproses permintaan yang dilakukan melalui API kita.

Pada file **ProductRoutes.php**:



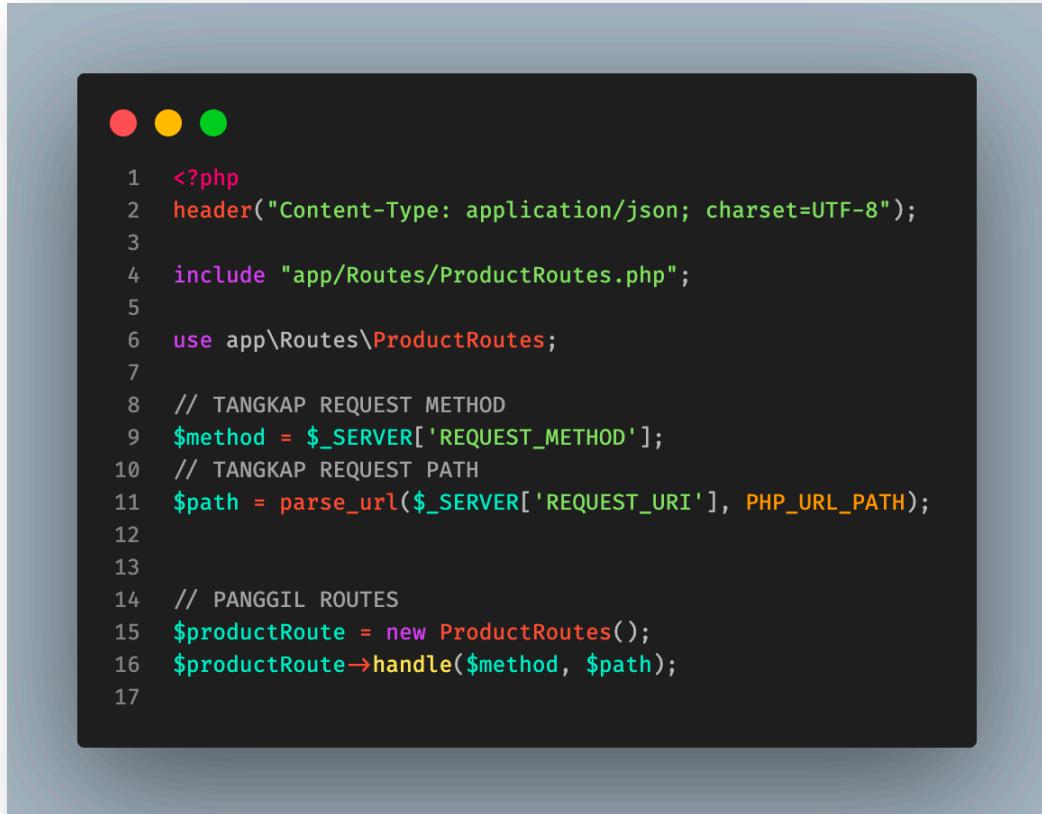
```

1  <?php
2
3  namespace app\Routes;
4
5  include "app\Controller\ProductController.php";
6
7  use app\Controller\ProductController;
8
9  class ProductRoutes
10 {
11     public function handle($method, $path)
12     {
13         // JIKA REQUEST METHOD GET DAN PATH SAMA DENGAN '/api/product'
14         if ($method === 'GET' && $path === '/api/product') {
15             $controller = new ProductController();
16             echo $controller->index();
17         }
18
19         // JIKA REQUEST METHOD GET DAN PATH MENGANDUNG '/api/product/'
20         if ($method === 'GET' && strpos($path, '/api/product/') === 0) {
21             // Extract ID dari path
22             $pathParts = explode('/', $path);
23             $id = $pathParts[count($pathParts) - 1];
24
25             $controller = new ProductController();
26             echo $controller->getById($id);
27         }
28
29
30         // JIKA REQUEST METHOD POST DAN PATH SAMA DENGAN '/api/product'
31         if ($method === 'POST' && $path === '/api/product') {
32             $controller = new ProductController();
33             echo $controller->insert();
34         }
35
36         // JIKA REQUEST METHOD PUT DAN PATH MENGANDUNG '/api/product/'
37         if ($method === 'PUT' && strpos($path, '/api/product/') === 0) {
38             // Extract ID dari path
39             $pathParts = explode('/', $path);
40             $id = $pathParts[count($pathParts) - 1];
41
42             $controller = new ProductController();
43             echo $controller->update($id);
44         }
45
46         // JIKA REQUEST METHOD DELETE DAN PATH MENGANDUNG '/api/product/'
47         if ($method === 'DELETE' && strpos($path, '/api/product/') === 0) {
48             // Extract ID dari path
49             $pathParts = explode('/', $path);
50             $id = $pathParts[count($pathParts) - 1];
51
52             $controller = new ProductController();
53             echo $controller->delete($id);
54         }
55     }
56 }
57

```

Pada file ProductRoutes.php adalah file untuk mendefinisikan alamat route API kita.

Pada file **main.php**:



```
1 <?php
2 header("Content-Type: application/json; charset=UTF-8");
3
4 include "app/Routes/ProductRoutes.php";
5
6 use app\Routes\ProductRoutes;
7
8 // TANGKAP REQUEST METHOD
9 $method = $_SERVER['REQUEST_METHOD'];
10 // TANGKAP REQUEST PATH
11 $path = parse_url($_SERVER['REQUEST_URI'], PHP_URL_PATH);
12
13
14 // PANGGIL ROUTES
15 $productRoute = new ProductRoutes();
16 $productRoute->handle($method, $path);
17
```

Pada file main.php adalah file yang berfungsi sebagai entrypoint dari Web Service Backend kita.

Setelah semua selesai, silahkan buka terminal pada root project kita dan jalankan perintah seperti berikut:

**“php -S localhost:8000 main.php”**

Setelah itu silahkan lakukan pengujian dengan aplikasi Postman, yang nantinya hasilnya akan sebagai berikut:

## (GET ALL PRODUCT) [GET] : http://localhost:8000/api/product

The screenshot shows a Postman collection named "New Collection / New Request". A GET request is made to "http://localhost:8000/api/product". The response status is 200 OK, with a time of 32 ms and a size of 523 B. The response body is a JSON object:

```

1 {
2   "code": 200,
3   "message": "success",
4   "data": [
5     {
6       "id": "2",
7       "product_name": "Roti",
8       "updated_at": "2023-08-25 03:22:57",
9       "created_at": "2023-08-25 03:22:57"
10    },
11    {
12      "id": "3",
13      "product_name": "Kecap",
14      "updated_at": "2023-08-25 03:23:53",
15      "created_at": "2023-08-25 03:23:53"
16    },
17    {
18      "id": "4",
19      "product_name": "Minuman",
20      "updated_at": "2023-08-25 03:24:08",
21      "created_at": "2023-08-25 03:24:08"
22    }
23  ]
24 }

```

## (GET PRODUCT BY ID) [GET] : http://localhost:8000/api/product/{id}

HTTP New Collection / New Request

GET  Send

Params Authorization Headers (7) Body Pre-request Script Tests Settings Cookies

Body (none form-data x-www-form-urlencoded raw binary GraphQL)

This request does not have a body

Body Cookies Headers (5) Test Results Status: 200 OK Time: 17 ms Size: 312 B Save as Example

Pretty Raw Preview Visualize JSON

```
1 "code": 200,
2 "message": "success",
3 "data": [
4   {
5     "id": 3,
6     "product_name": "Kecap",
7     "updated_at": "2023-08-25 03:23:53",
8     "created_at": "2023-08-25 03:23:53"
9   }
10 ]
11 ]
12 ]
```

(CREATE PRODUCT) [POST] : http://localhost:8000/api/product

New Collection / New Request

POST http://localhost:8000/api/product

Params Authorization Headers (9) Body Pre-request Script Tests Settings Cookies Beautify

Body (JSON)

```
1 ...
2   "product_name": "Saos Tomat"
3 ...
```

Body Cookies Headers (5) Test Results Status: 200 OK Time: 52 ms Size: 213 B Save as Example ...

Pretty Raw Preview Visualize JSON

```
1 ...
2   "code": 200,
3   "message": "success",
4   "data": null
5 ...
```

(UPDATE PRODUCT BY ID) [PUT] : http://localhost:8000/api/product/{id}

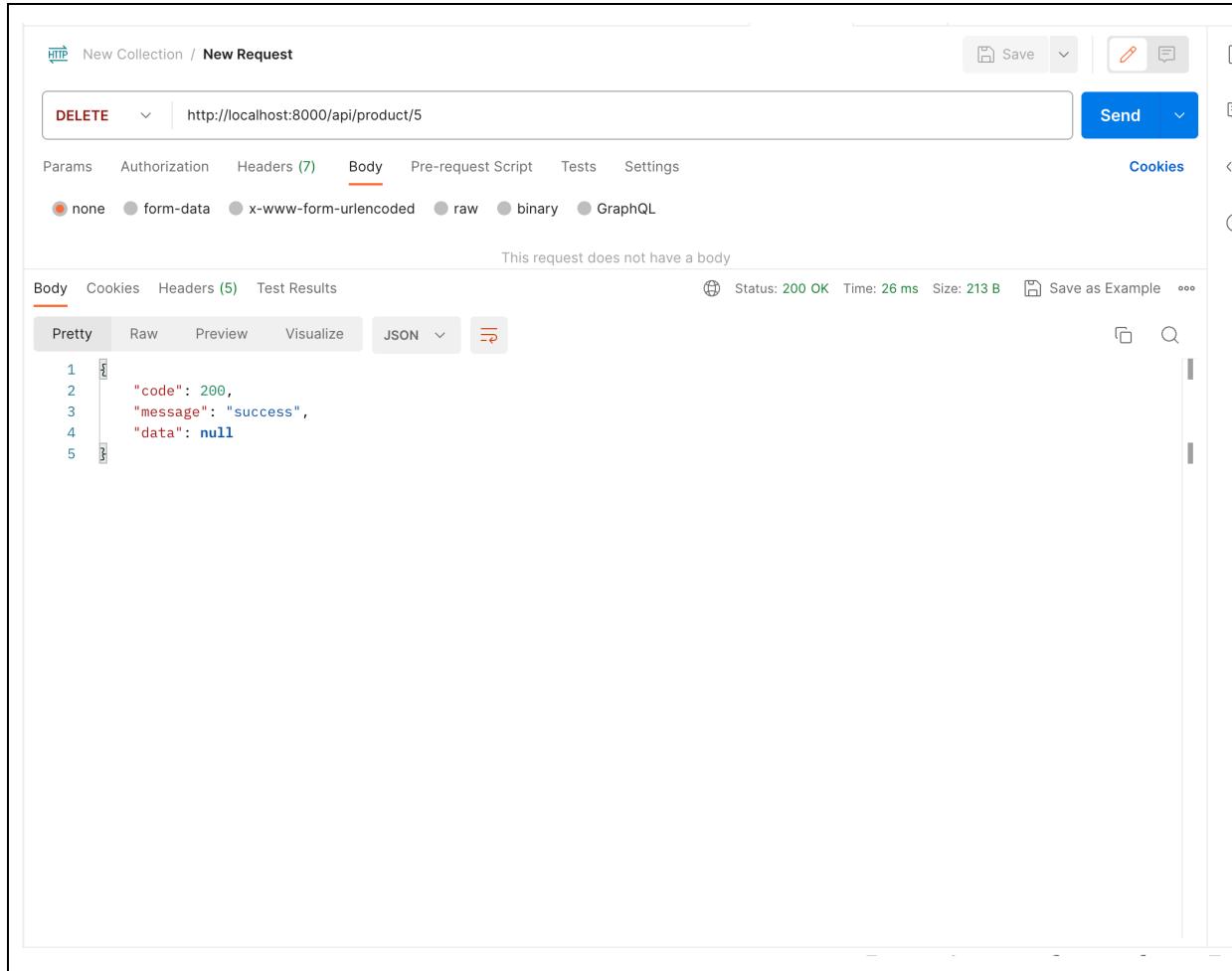
The screenshot shows the Postman application interface. At the top, it says "HTTP New Collection / New Request". Below that, a "PUT" method is selected with the URL "http://localhost:8000/api/product/4". The "Body" tab is active, showing the JSON payload:

```
1
2 ... "product_name": "Berubah menjadi Jeruk"
3
```

Below the body, the response status is shown as "Status: 200 OK Time: 19 ms Size: 213 B". The response body is displayed in "Pretty" format:

```
1
2   "code": 200,
3   "message": "success",
4   "data": null
5
```

At the bottom of the main window, there is a note: "(DELETE PRODUCT BY ID) [DELETE] : http://localhost:8000/api/product/{id}".



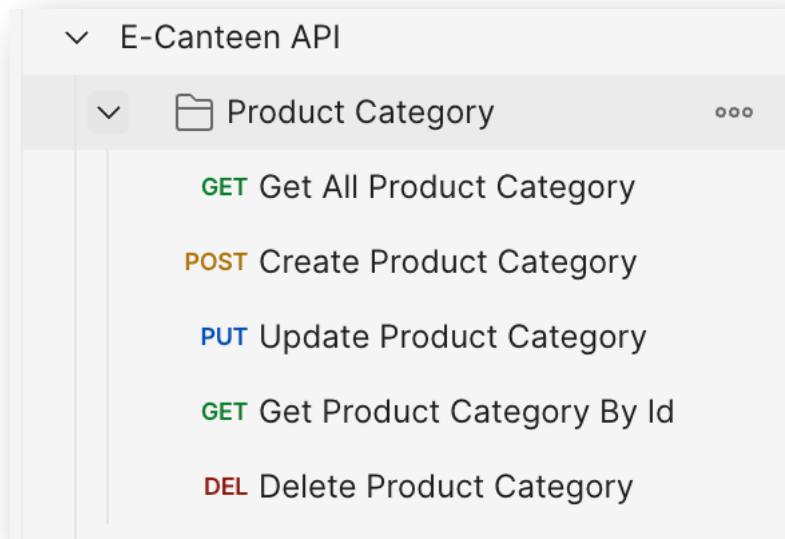
Selamat kalian berhasil membuat REST API sederhana yang terkoneksi dengan MySQL serta menerapkan konsep Clean Code dan OOP, tentunya banyak aspek yang belum sempurna dari codelab kita dari segi Security, Best Practice dan lain – lainnya.

---

# TUGAS

## TUGAS 1

Buatlah sebuah Create, Read, Update, Delete (CRUD) Endpoint API seperti pada CODELAB diatas dan tambahkan relasi pada database nya, terapkan konsep JOIN table dan buat collection pada Postman yang berisi Endpoint yang kalian buat masing – masing sesuai dengan HTTP Method nya (GET, POST, PUT, DELETE, dll) seperti pada gambar berikut:



Gambar diatas hanya sebagai contoh Collection pada Postman, kalian bebas menggunakan tema apapun dan improvisasi serta kreatifitas apapun selama masih didalam lingkup materi, jelaskan pada asisten mengenai Endpoint yang kalian buat!

#### NOTES:

Asisten akan mengecek pemahaman anda mengenai materi pada modul dan konsep dari REST API serta komunikasi PHP dengan database.

---

#### KRITERIA & DETAIL PENILAIAN

Kriteria	Presentase Penilaian
Dapat melakukan koneksi PHP pada MySQL	20%
Dapat menerapkan akses basis data pada PHP	40%
Dapat menjawab pertanyaan dari asisten laboratorium	10%

Dapat menyajikan data dari database dengan REST API berformat JSON	30%
--	-----