

Nama : Ridho Akbarsyah Ramadhan

NIM : 18104020

Kelas : S1 SE 02 A

## **Tutorial Membuat Restful API dengan CodeIgniter**

Apa itu RESTful API?

RESTful API merupakan implementasi dari API (*Application Programming Interface* ([https://en.wikipedia.org/wiki/Application\\_programming\\_interface](https://en.wikipedia.org/wiki/Application_programming_interface))).

REST (REpresentational State Transfer) adalah suatu arsitektur metode komunikasi yang menggunakan protocol HTTP untuk pertukaran data dan metode ini sering diterapkan dalam pengembangan aplikasi. Mungkin terdengar rumit tapi sebenarnya tidak. Tujuannya untuk menjadikan system dengan performa yang baik, cepat, dan mudah untuk dikembangkan terutama dalam pertukaran dan komunikasi data.

### **Step #1. Design EndPoint RESTful API**

Ini penting!

Sebelum membuat RESTful API, ada baiknya di kita deznisikan dulu EndPoint dari RESTful API yang akan dibuat. EndPoint merupakan routes dari API yang akan kita buat. RESTful API menggunakan HTTP verbs

([https://en.wikipedia.org/wiki/Hypertext\\_Transfer\\_Protocol#Request\\_methods](https://en.wikipedia.org/wiki/Hypertext_Transfer_Protocol#Request_methods)).

HTTP verbs yang umum digunakan adalah GET, POST, PUT, dan DELETE. GET untuk mendapatkan data dari server atau lebih dikenal dengan istilah READ, POST untuk meng-CREATE new data, PUT untuk UPDATE data, dan DELETE untuk menghapus data. Atau lebih dikenal dengan istilah CRUD (Create-Read-Update-Delete)

Pada tutorial kali ini, saya akan sharing bagaimana membuat RESTful API sederhana untuk mengambil data dari server (GET), membuat data baru ke server (POST), mengupdate data ke server (PUT), dan menghapus data ke server (DELETE) dari suatu table di database yaitu table product.

### **Step #2. Buat Database dan Table**

Buat sebuah database baru dengan MySQL, Anda dapat menggunakan tools seperti SQLyog, PHPMyAdmin atau sejenisnya. Disini saya membuat database dengan nama **restful\_db**. Jika Anda membuat database dengan nama yang sama itu lebih baik.

Untuk membuat database dengan MySQL, dapat dilakukan dengan mengeksekusi query berikut:

```
CREATE DATABASE restful_db;
```

Perintah SQL diatas akan membuat sebuah database dengan nama **restful\_db**. Selanjutnya, buat sebuah table di dalam database **restful\_db**. Disini saya membuat sebuah table dengan nama **product**. Jika Anda membuat table dengan nama yang sama itu lebih baik. Untuk membuat table product, dapat dilakukan dengan mengeksekusi perintah SQL berikut:

```
CREATE TABLE product(  
product_id INT(11) PRIMARY KEY AUTO_INCREMENT,  
product_name VARCHAR(200),  
product_price DOUBLE  
)ENGINE=INNODB;
```

Selanjutnya, insert beberapa data kedalam table product dengan mengeksekusi query berikut:

```
INSERT INTO product(product_name,product_price) VALUES  
( 'Tesla Model 3', '100000'),  
( 'Toyota Fortuner', '900000'),  
( 'Honda Jazz', '800000'),  
( 'Hyundai Palisade', '700000'),  
( 'Suzuki Katana', '600000');
```

Perintah SQL diatas akan menginput 5 data kedalam table **product**.

### Step #3. Instalasi CodeIgniter 4

Download zle Codeigniter 4 pada link berikut:

<https://codeigniter.com>

(<https://codeigniter.com>) Kemudian extract di web server Anda.

Jika Anda menggunakan XAMPP, extract di folder:

```
C:\xampp\htdocs
```

Pada tutorial ini, saya menggunakan XAMPP.

Kemudian rename (ganti nama) menjadi “restfulapi” seperti gambar berikut:

18104020_Ridho-Akbaryah-Ramadhan	28/12/2020 21.59	File folder	
dashboard	11/07/2020 15.11	File folder	
img	11/07/2020 15.11	File folder	
practice_laravel	23/12/2020 08.11	File folder	
restfulapi	28/12/2020 21.46	File folder	
webalizer	11/07/2020 15.11	File folder	
xampp	11/07/2020 15.11	File folder	
applications	27/08/2019 21.02	Firefox HTML Doc...	4 KB
bitnami	27/08/2019 21.02	CSS File	1 KB
favicon	16/07/2015 22.32	Icon	31 KB
index	16/07/2015 22.32	PHP Source File	1 KB

Kemudian open project restful api dengan code editor, disini saya menggunakan Sublime Text 3

#### Step #4. Membuat koneksi ke database

Buka zle “Database.php” yang terdapat pada folder “app/Config”, kemudian temukan kode berikut:

```
public $default = [
    'DSN' => '',
    'hostname' => 'localhost',
    'username' => '',
    'password' => '',
    'database' => '',
    'DBDriver' => 'MySQLi',
    'DBPrefix' => '',
    'pConnect' => false,
    'DBDebug' => (ENVIRONMENT !== 'production'),
    'cacheOn' => false,
    'cacheDir' => '',
    'charset' => 'utf8',
    'DBCollat' => 'utf8_general_ci',
    'swapPre' => '',
    'encrypt' => false,
    'compress' => false,
    'strictOn' => false,
    'failover' => [],
    'port' => 3306,
];
```

Kemudian ubah menjadi seperti berikut:

```
'DSN' => '',
'hostname' => 'localhost',
'username' => 'root',
```

```
'password' => '',
'database' => 'restful_db',
'DBDriver' => 'MySQLi',
'DBPrefix' => '',
'pConnect' => false,
'DBDebug' => (ENVIRONMENT !== 'production'),
'cacheOn' => false,
'cacheDir' => '',
'charset' => 'utf8',
'DBCollat' => 'utf8_general_ci',
'swapPre' => '',
'encrypt' => false,
'compress' => false,
'strictOn' => false,
'failover' => [],
'port' => 3306,
];
```

Selanjutnya, Ini penting!

Agar Anda memiliki interface yang baik untuk menangani error, temukan zle env pada root project, kemudian rename (ganti nama) menjadi .env dan open zle tersebut.

Kemudian temukan kode berikut:

```
# CI_ENVIRONMENT = production
```

Kemudian ubah menjadi seperti berikut:

```
CI_ENVIRONMENT = development
```

Itu artinya Anda masuk ke mode development, mode ini akan membantu Anda mempermudah melacak error saat Anda membangun project.

### Step #5. Membuat file Model

Buat sebuah file model bernama “ProductModel.php” pada folder “app/Models”, kemudian ketikkan kode berikut:

```
<?php namespace App\Models;
use CodeIgniter\Model;
class ProductModel extends Model
{
protected $table = 'product';
protected $primaryKey = 'product_id';
protected $allowedFields =
['product_name', 'product_price'];
}
```

### Step #6. Membuat file Controller

Buat sebuah file controller bernama “Products.php” pada folder “app/Controllers”, kemudian ketikkan kode berikut:

```
<?php namespace App\Controllers;
use CodeIgniter\RESTful\ResourceController;
use CodeIgniter\API\ResponseTrait;
use App\Models\ProductModel;
class Products extends ResourceController
{
use ResponseTrait;
// get all product
public function index()
{
$model = new ProductModel();
$data = $model->findAll();
return $this->respond($data, 200);
}
// get single product
public function show($id = null)
{
$model = new ProductModel();
```

```

$data      =      $model->getWhere(['product_id'      =>
$id])->getResult();
if($data){
return $this->respond($data);
}else{
return $this->failNotFound('No Data Found with id '.$id);
}
}
// create a product
public function create()
{
$model = new ProductModel();
$data = [
'product_name' => $this->request->getPost('product_name'),
'product_price'      =>
$this->request->getPost('product_price')
];
$data      =      json_decode(file_get_contents("php://input
/php://input"));
//$data = $this->request->getPost();
$model->insert($data);
$response = [
'status' => 201,
'error' => null,
'messages' => [
'success' => 'Data Saved'
]
];
return $this->respondCreated($data, 201);
}
// update product
public function update($id = null)
{
$model = new ProductModel();

```

```
$json = $this->request->getJSON();  
if($json){  
    $data = [  
        'product_name' => $json->product_name,  
        'product_price' => $json->product_price  
    ];  
}else{  
    $input = $this->request->getRawInput();  
    $data = [  
        'product_name' => $input['product_name'],  
        'product_price' => $input['product_price']  
    ];  
}  
// Insert to Database  
$model->update($id, $data);  
$response = [  
    'status' => 200,  
    'error' => null,  
    'messages' => [  
        'success' => 'Data Updated'  
    ]  
];  
return $this->respond($response);  
}  
// delete product  
public function delete($id = null)  
{  
    $model = new ProductModel();  
    $data = $model->find($id);  
    if($data){  
        $model->delete($id);  
        $response = [  
            'status' => 200,  
            'error' => null,
```

```

'messages' => [
'success' => 'Data Deleted'
]
];
return $this->respondDeleted($response);
}else{
return $this->failNotFound('No Data Found with id '.$id);
}
}
}
}

```

CodeIgniter 4 telah memberikan kemudahan bagi web developer dalam membuat RESTful API. Dapat dilihat pada controller Products.php diatas, Dengan hanya mengextends ResourceController ([http://codeigniter.com/user\\_guide/incoming/restful.html](http://codeigniter.com/user_guide/incoming/restful.html)) kita telah dapat membuat RESTful API. Tidak hanya itu, Kita juga bisa dengan mudah membuat response dengan menggunakan API ResponseTrait ([http://codeigniter.com/user\\_guide/outgoing/api\\_responses.html](http://codeigniter.com/user_guide/outgoing/api_responses.html)).

### Step #7. Konfigurasi Routes.php

Langkah terakhir yang tidak kalah pentingnya yaitu melakukan sedikit konfigurasi pada file **Routes.php** yang terdapat pada folder “**app/Config**”. Buka file “**Routes.php**” pada folder “**app/Config**”, kemudian temukan kode berikut:

```
$routes->get('/', 'Home::index');
```

Kemudian, ganti menjadi berikut:

```
$routes->resource('products');
```

### Step #8. Aktifkan CORS (Cross-Origin Resources Sharing)

Ini penting!

Agar resources dapat diakses di luar domain, kita perlu mengaktifkan CORS. Untuk mengaktifkan CORS, buat file bernama “**Cors.php**” pada folder “**app/Filters**”. Kemudian ketikkan kode berikut:

```

<?php namespace App\Filters;
use CodeIgniter\HTTP\RequestInterface;
use CodeIgniter\HTTP\ResponseInterface;

```



```

use CodeIgniter\Filters\FilterInterface;
Class Cors implements FilterInterface
{
public function before(RequestInterface $request, $arguments
= null)
{
header('Access-Control-Allow-Origin: *');
header("Access-Control-Allow-Headers: X-API-KEY, Origin,
X-Requested-Wi
header("Access-Control-Allow-Methods: GET, POST, OPTIONS,
PUT, DELETE"
$method = $_SERVER['REQUEST_METHOD'];
if ($method == "OPTIONS") {
die();
}
}
public function after(RequestInterface $request,
ResponseInterface $respons
{
// Do something here
}
}

```

Selanjutnya buka file **“Filters.php”** yang terdapat pada folder **“app/Config”**.

Kemudian temukan kode berikut:

```

public $aliases = [
'csrf' => \CodeIgniter\Filters\CSRF::class,
'toolbar' => \CodeIgniter\Filters\DebugToolbar::class,
'honeypot' => \CodeIgniter\Filters\Honeypot::class,
];

```

Kemudian tambahkan cors filter seperti berikut:

```

public $aliases = [
'csrf' => \CodeIgniter\Filters\CSRF::class,
'toolbar' => \CodeIgniter\Filters\DebugToolbar::class,
'honeypot' => \CodeIgniter\Filters\Honeypot::class,

```

```
'cors' => \App\Filters\Cors::class,  
];
```

Selanjutnya deznisikan "cors" pada public globals seperti berikut:

```
public $globals = [  
  'before' => [  
    'cors'  
    //'honeypot'  
    // 'csrf',  
  ],  
  'after' => [  
    'toolbar',  
    //'honeypot'  
  ],  
];
```

## Step #9. Testing

Untuk menguji coba API yang telah kita buat, ada banyak tools yang dapat digunakan.

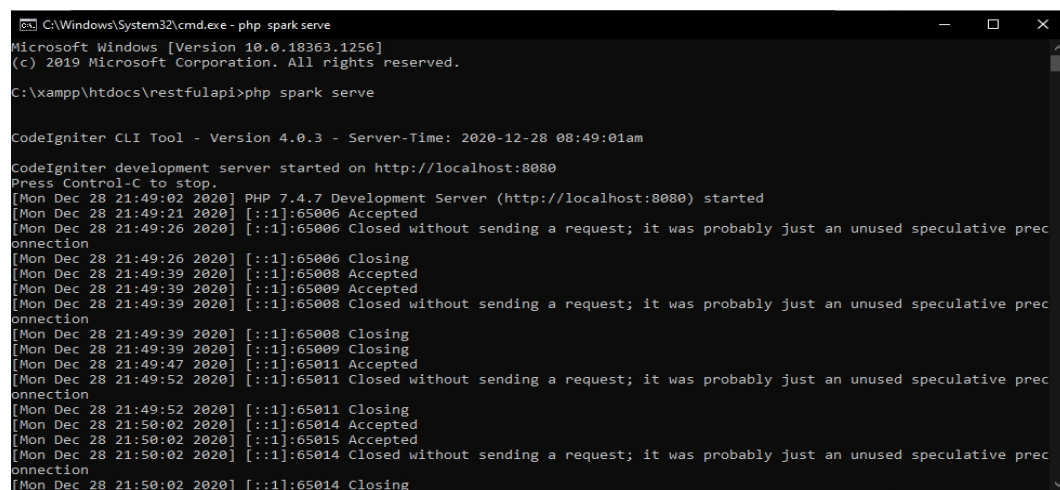
Pada tutorial ini, saya menggunakan POSTMAN.

Saya juga menyarankan Anda untuk menggunakan POSTMAN. Anda dapat mendownload POSTMAN di website resminya:

<https://www.getpostman.com/> (<https://www.getpostman.com/>) Download dan Install POSTMAN di komputer Anda kemudian open. Jalankan project dengan mengetikkan perintah berikut pada Terminal / Command Prompt

```
php spark serve
```

Seperti gambar berikut:



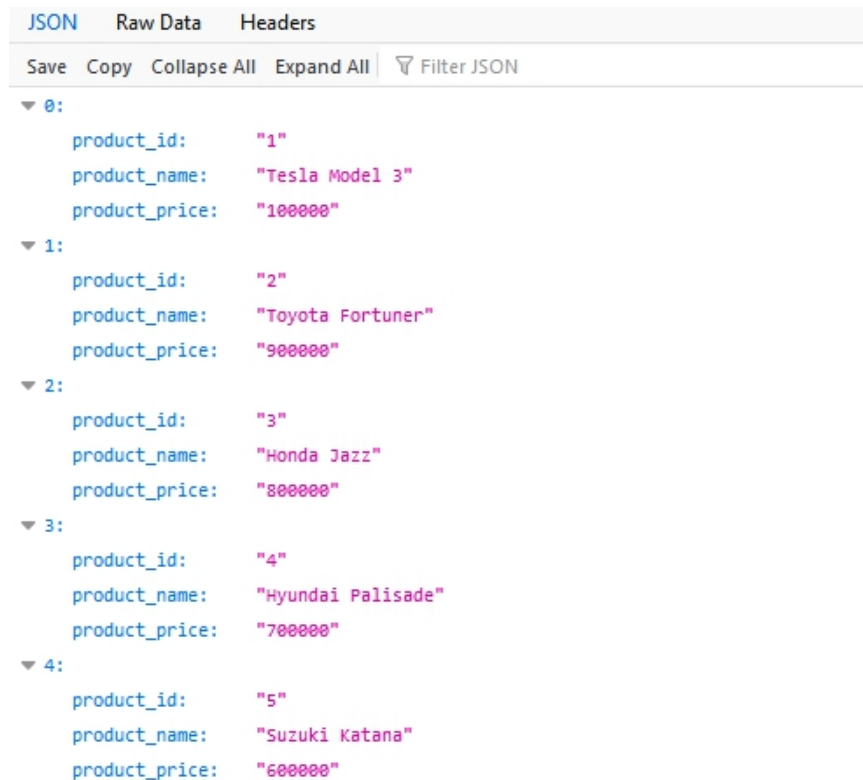
```
C:\Windows\System32\cmd.exe - php spark serve  
Microsoft Windows [Version 10.0.18363.1256]  
(c) 2019 Microsoft Corporation. All rights reserved.  
  
C:\xampp\htdocs\restfulapi>php spark serve  
  
CodeIgniter CLI Tool - Version 4.0.3 - Server-Time: 2020-12-28 08:49:01am  
  
CodeIgniter development server started on http://localhost:8080  
Press Control-C to stop.  
[Mon Dec 28 21:49:02 2020] PHP 7.4.7 Development Server (http://localhost:8080) started  
[Mon Dec 28 21:49:21 2020] [::1]:65006 Accepted  
[Mon Dec 28 21:49:26 2020] [::1]:65006 Closed without sending a request; it was probably just an unused speculative prec  
onnection  
[Mon Dec 28 21:49:26 2020] [::1]:65006 Closing  
[Mon Dec 28 21:49:39 2020] [::1]:65008 Accepted  
[Mon Dec 28 21:49:39 2020] [::1]:65009 Accepted  
[Mon Dec 28 21:49:39 2020] [::1]:65008 Closed without sending a request; it was probably just an unused speculative prec  
onnection  
[Mon Dec 28 21:49:39 2020] [::1]:65008 Closing  
[Mon Dec 28 21:49:39 2020] [::1]:65009 Closing  
[Mon Dec 28 21:49:47 2020] [::1]:65011 Accepted  
[Mon Dec 28 21:49:52 2020] [::1]:65011 Closed without sending a request; it was probably just an unused speculative prec  
onnection  
[Mon Dec 28 21:49:52 2020] [::1]:65011 Closing  
[Mon Dec 28 21:50:02 2020] [::1]:65014 Accepted  
[Mon Dec 28 21:50:02 2020] [::1]:65015 Accepted  
[Mon Dec 28 21:50:02 2020] [::1]:65014 Closed without sending a request; it was probably just an unused speculative prec  
onnection  
[Mon Dec 28 21:50:02 2020] [::1]:65014 Closing
```

## 1. Get All Product (GET)

Kembali ke POSTMAN, dan ketikkan URL berikut pada kolom URL Postman:

`http://localhost:8080/products` (`http://localhost:8080/products`)

Pilih method GET, kemudian klik tombol Send, maka Akan terlihat hasilnya seperti gambar berikut:



Pada gambar diatas dapat dilihat bahwa EndPoint untuk GET semua data product berjalan dengan baik.

## 2. Get Single Product (GET)

Ketikkan URL berikut pada kolom URL untuk mendapatkan single product:

`http://localhost:8080/products/12` (`http://localhost:8080/products/12`)

JSON	Raw Data	Headers
Save	Copy	Collapse All Expand All Filter JSON
▼ 0:		
product_id:	"5"	
product_name:	"Suzuki Katana"	
product_price:	"600000"	

Pada gambar diatas dapat dilihat bahwa EndPoint untuk GET single product berjalan dengan baik.

**Perhatian: Anda mungkin tidak memiliki product dengan id=5, harap disesuaikan dengan data yang Anda miliki!**

### 3. Create New Product (POST)

Ketikan URL berikut pada kolom URL untuk meng-create new product:

`http://localhost:8080/products` (`http://localhost:8080/products`)

Pilih method POST => Body => form-urlencoded => Masukkan KEY dan VALUE => klik Send. Seperti gambar berikut:

JSON	Raw Data	Headers
Save	Copy	Collapse All Expand All Filter JSON
▼ 0:		
product_id:	"1"	
product_name:	"Tesla Model 3"	
product_price:	"100000"	
▼ 1:		
product_id:	"2"	
product_name:	"Toyota Fortuner"	
product_price:	"900000"	
▼ 2:		
product_id:	"3"	
product_name:	"Honda Jazz"	
product_price:	"800000"	
▼ 3:		
product_id:	"4"	
product_name:	"Hyundai Palisade"	
product_price:	"700000"	
▼ 4:		
product_id:	"5"	
product_name:	"Suzuki Katana"	
product_price:	"600000"	

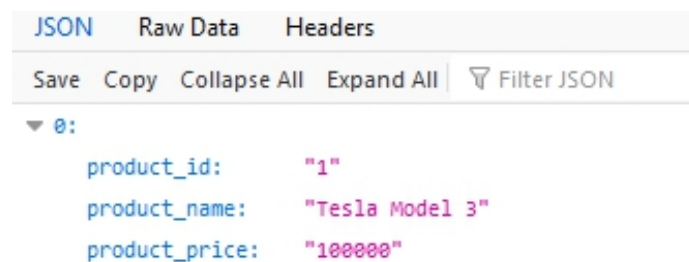
Jika Anda mendapatkan response dengan `product_id`, `product_name`, `product_price` seperti gambar diatas, maka data berhasil tersimpan di database.

#### 4. Update Product (PUT)

Ketikan URL berikut pada kolom URL untuk meng-update product:

`http://localhost:8080/products/1` (`http://localhost:8080/products/1`)

Pilih method PUT => Body => form-urlencoded => Masukkan KEY dan VALUE => klik Send. Seperti gambar berikut:



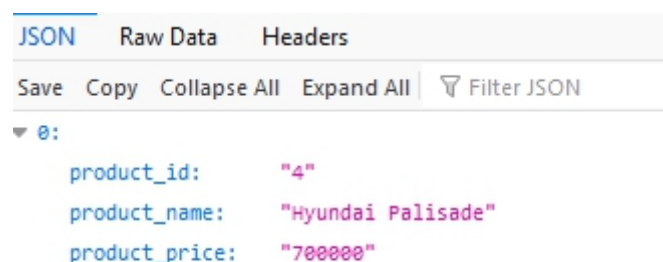
Jika Anda mendapatkan response dengan `product_id`, `product_name`, `product_price` seperti gambar diatas, maka data berhasil terupdate di database sesuai dengan id product yang di update.

#### 5. Delete Product (DELETE)

Ketikan URL berikut pada kolom URL untuk meng-hapus product:

`http://localhost:8080/products/4` (`http://localhost:8080/products/4`)

Pilih method DELETE, kemudian klik Send. Seperti gambar berikut



Jika Anda mendapatkan response dengan nama  
product\_id,product\_name,product\_price seperti gambar diatas, maka  
data berhasil terhapus di database