Membangun Antarmuka Pemrograman Aplikasi (API) REST dengan PHP

Husni@trunojoyo.ac.id

Daftar Isi

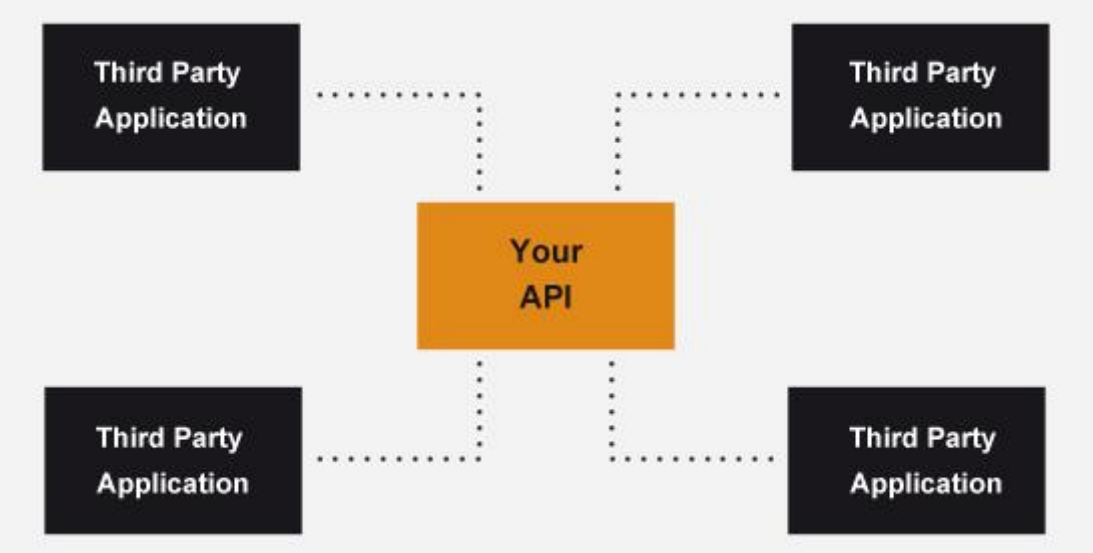
Pendahuluan

Dalam dunia sekarang, aplikasi-aplikasi berbeda pada berbagai perangkat saling terkoneksi dan alasan utama di belakangnya adalah API. Sebelum menjelajah ke dalam REST API, mari kita lihat lebih dahulu apa itu API. Anda mungkin sudah mengetahui apa yang dimaksud API. Karena artikel ini berkaitan dengan REST API, maka ada baiknya API dibahas pula.

API merupakan singkatan bagi Application Programming Interface dan gagasan di balik API adalah untuk menghubungkan aplikasi-aplikasi berbeda apapun platformnya untuk berbagi informasi. Secara umum, API menerima requests dari aplikasi, memrosesnya dan memberikan respon (jawaban balik).

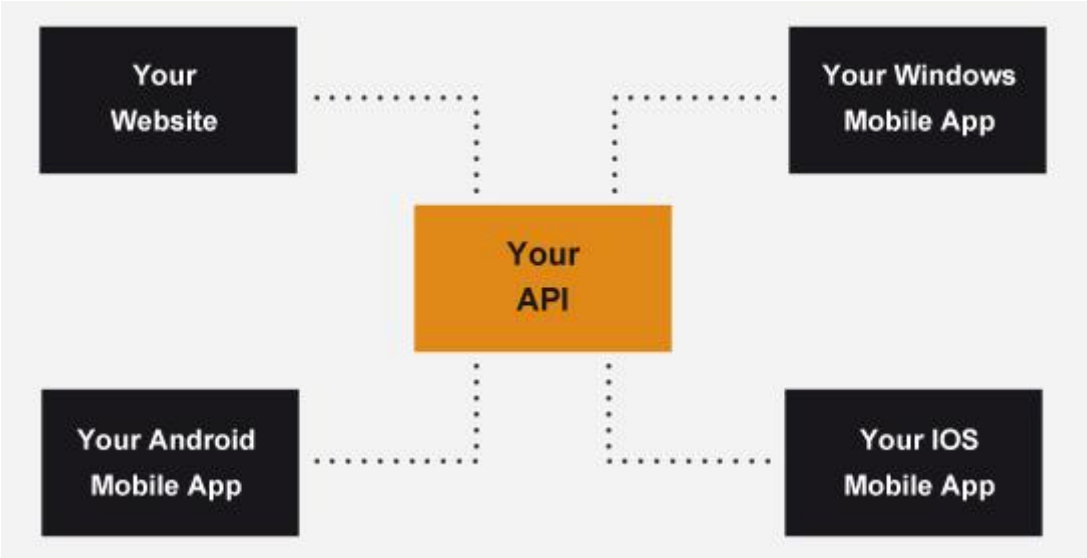
Kita dapat menggunakan API untuk melakukan hal berikut.

a. **Membangun suatu API agar aplikasi pihak ketiga (*third party applications*) terhubung dengan aplikasi kita.**



b. **Mengakses API pihak ketiga untuk menghubungkan dan menggunakan informasi mereka.**

c. **Membangun suatu API untuk menghubungkan aplikasi-aplikasi kita sendiri seperti situs web dan aplikasi perangkat bergerak (*mobile app*).**



Apa itu REST API?

REST merupakan singkatan dari *Representational State Transfer* dan itu berati bahwa request dan responsenya harus mengandung suatu representasi informasi yaitu harus dalam format tertentu. Maksudnya, pada dasarnya, request harus menggunakan metode HTTP yang tepat dan respon harus dalam format seperti JSON atau XML, bukan teks plain.

REST API bukan suatu API baru ataupun standar. REST hanya API normal dengan sehimpunan prinsip dalam pengembangan API yang dapat diakses di Internet. Kita perlu mengikuti himpunan aturan selama pembuatan dan pemanfataan (*consuming*, mengakses, melanggan) REST API.

Berikut ini adalah aturan dasar dari REST API:

1. Gunakan metode HTTP yang tepat saat melakukan pemanggilan API (dari sisi *consumer*). Beriku tini adalah empat metode HTTP utama yang harus digunakan untuk mengirimkan dan menerima request API.
    a. GET untuk membaca satu atau banyak record. read
    b. POST untuk membuat suatu record baru. insert
    c. PUT untuk mengupdate suatu record. update
    d. DELETE untuk menghapus suatu record. delete

2. Gunakan hirarki URL yang tepat, bukan menggunakan string query URL bagi API URL.
    a. Bagus : http://example.com/api/products/1
    b. Buruk : http://example.com/api/products.php?id=1

3. Hindari penggunaan kata kerja sebagai nama sumber daya (*resource*) dalam API URL. Sebagai gantinya gunakan kata benda dan metode HTTP yang tepat.
    a. Bagus : http://example.com/api/products
    b. Buruk : http://example.com/api/products/add

4. Gunakan bentuk jamak untuk nama sumber daya dalam API URL.
    a. Bagus : http://example.com/api/products
    b. Buruk : http://example.com/api/product

5. Gunakan kode respon HTTP untuk menunjukkan status dari request.
6. Data respon harus dalam format JSON atau XML.

Pustaka Client HTTP: cURL

Sebagian besar dari kita sudah mengetahui apa itu HTTP. HTTP merupakan singkatan bagi Hyper Text Transfer Protocol dan itu adalah protokol yang memungkinkan kita untuk mengirimkan informasi (*back and forth*) di web. Kapan pun kita membuat suatu request (permintaan) HTTP maka kita gunakan salah satu dari metode HTTP (GET, POST, PUT, DELETE, dll.).

Jadi, dalam rangka menggunakan REST APIs, kita perlu suatu client yang mempunyai kapabilitas untuk menggunakan semua metode HTTP. Sayangnya, HTML terbatas dalam hal ini. HTML hanya dapat mengirimkan request GET dan POST yang tidak cukup untuk menjadi client pengguna REST API.

Sehingga kita perlu pustakan client HTTP dan salah satunya pustaka client REST API yang cukup bagus adalah cURL. cURL adalah pustaka client HTTP yang sangat popular dan digunakan secara luas dikalangan pengembang PHP. Kita akan menggunakan cURL (nanti) saat mengakses REST API yang telah dibuat sebelumnya.

Membuat REST API Menggunakan PHP

Mari kita membangun suatu REST API sederhana dalam PHP dengan apa yang kita sudah lihat sejauh ini. Katakanlah, kita mempunyai suatu katalog produk online dan kita ingin web site dan aplikasi mobile kita berbagi informasi sama mengenai produk tersebut. Karena itu, kita perlu membangun suatu API yang memungkinkan adanya penambahan (add), perubahan (update), pengambilan (read) dan penghapusan (delete) informasi produk.

Kita menganggap bahwa nama domain kita adalah example.com dan example.com/api/ adalah lokasi dari API yang akan dibangun. Kita perlu menambahkan satu file PHP (**products.php**) ke folder /api/ tersebut. Tabel berikut mengilustrasikan URL-URL dan metode HTTP yang harus digunakan untuk mengerjakan aksi yang tepat dengan API kita.

Metode HTTP	URL	Aksi
GET	/api/products	Meretrieve semua produk
GET	/api/products/5	Meretrieve satu produk yang kunci utamanya 5
POST	/api/products	Menambahkan suatu produk baru
PUT	/api/products/3	Mengupdate satu produk yang kunci utamanya 3
DELETE	/api/products/7	Menghapus satu produk yang kunci utamanya 7

File PHP tersebut (products.php) adalah dimana kita akan meletakkan semua kode API.

Menulis Ulang URL (.htaccess)

Dalam pembangunan suatu API, kita juga perlu menulis ulang (rewrite) URL agar mengikuti aturan REST. Ini dilakukan dengan menambahkan suatu file.htaccess ke folder /api/ dan menempatkan baris-baris berikut ke dalamnya.

```
RewriteEngine On # Turn on the rewriting engine
RewriteRule ^products/?$ products.php [NC,L]
RewriteRule ^products/([0-9]+)/?$ products.php?product_id=$1 [NC,L]
```

Dengan baris-baris di atas, request yang pathnya berbentuk /api/products.php?product_id=5 dapat ditulis dengan /api/products/5. Bukankah bentuk terakhir lebih mudah diingat dan mengamankan rincian request?

Dikarenakan URL rewriting sendiri adalah topik yang besar maka tidak mungkin membahasnya rincian di sini. Jika anda memerlukan teknik dan ingin mengetahui aturan-aturan dalam URL rewriting, kami merekomendasikan anda untuk mengakses link berikut:

https://www.addedbytes.com/articles/for-beginners/url-rewriting-for-beginners/

Identifikasi Request HTTP

Langkah pertama dalam praktek REST API adalah menempatkan kode program PHP untuk mengidentifikasi metode request HTTP yang dikirimkan oleh client (pelanggan) REST API. Berdasarkan pada request tersebut, server (*provider*) menjalankan tindakan yang sesuai.

```
	// Membangun koneksi ke database
	$connection = mysqli_connect('localhost','root','','rest_api');

	//mengambil metode request
	$request_method = $_SERVER["REQUEST_METHOD"];

	//metode apa yang digunakan client?
	switch($request_method) {
		case 'GET':
			// Metode GET, client hanya ingin meretrieve produk
			// dengan product_id tertentu?
			if(!empty($_GET["product_id"])) {
				$product_id = intval($_GET["product_id"]);
				get_products($product_id);
			}
			//jika tidak dengan product_id, berarti semua produk
			else {
				get_products();
			}
			break;

		case 'POST':
			// Metode POST, untuk menambahkan produk baru (Insert)
			insert_product();
			break;
```

```
            case 'PUT':
                  // Metode PUT, client ingin mengupdate produk tertentu
                  $product_id = intval($_GET["product_id"]);
                  update_product($product_id);
                  break;

            case 'DELETE':
                  // Metode DELETE, client iningin menghapus produk tertentu
                  $product_id = intval($_GET["product_id"]);
                  delete_product($product_id);
                  break;

            default:
                  // Jika bukan salah satu dari 4 metode di atas
                  header("HTTP/1.0 405 Metode Tidak Dikenali.");
                  break;
      }
```

Dalam contoh kode di atas, kita pertama-tama menghubungi database yang akan menyimpan semua informasi produkl. Kemudian kita gunakan variabel super global PHP $_SERVER untuk memperoleh metode request HTTP yang digunakan oleh pemanggil API (client). Suatu blok switch case kita gunakan untuk mengerjakan tindakan yang tepat dan sesuai.

Meminta Informasi Produk

Bagaimana kita meretrieve informasi produk? Seperti pada kode sebelumnya, kita harus mempunyai suatu fungsi **get_products()**. Dari namanya dapat diperkirakan bahwa isi fungsi ini adalah mengambil satu atau lebih record informasi produk dari satu atau lebih tabel di database. Jika API hanya meminta satu produk maka product_id dikirimkan sebagai parameter ke fungsi ini. Jika tidak ada prouct_id, maka product_id dianggap 0 dan itu artinya fungsi ini akan meretrieve semua produk.

```
function get_products($product_id=0) {
            global $connection;

            //query mengambil semua produk
            $query = "SELECT * FROM products";

            //hanya mengambil satu produk sesuai product_id
            if($product_id ! = 0) {
                  $query .= " WHERE id = " . $product_id . " LIMIT 1";
            }

            $response = array();
            $result = mysqli_query($connection, $query);

            while($row = mysqli_fetch_array($result)) {
                  $response[] = $row;
            }

            //respon untuk client dalam format JSON
            header('Content-Type: application/json');
            echo json_encode($response);
      }
```

Menambahkan Produk Baru

Bagaimana menambahkan atau menyisipkan informasi produk baru? Kita perlu membuat fungsi insert_product(). Karena metode HTTP POST akan digunakan untuk membuat panggilan API untuk menambahkan produk, kita perlu rincian dari produk baru dari variabel $_POST sendiri.

```
function insert_product() {
            global $connection;
            $product_name = $_POST["product_name"];
            $price = $_POST["price"];
            $quantity = $_POST["quantity"];
            $seller = $_POST["seller"];

            $query = ”INSERT INTO products SET” .
                  “ product_name = '{$product_name}', price = {$price},” .
                  “ quantity = {$quantity}, seller = '{$seller}'”;

            if(mysqli_query($connection, $query)) {
                  $response = array(
                        'status' => 1,
                        'status_message' =>'Produk berhasil ditambahkan.'
                  );
            }
            else {
                  $response = array(
                        'status' => 0,
                        'status_message' =>'Produk GAGAL ditambahkan.'
                  );
            }

            header('Content-Type: application/json');
            echo json_encode($response);
      }
```

Apa nama database dan tabel yang digunakan dalam API di atas? Apa saja field-field yang ada di dalam tabel tersebut? Silakan buat database dan tabel yang sesuai sebelum API ini diujicobakan!

Mengupdate Produk Tertentu

Pada proses meng-update a produk, kita menggunakan fungsi update_product(). Karena PHP tidak mempunyai variabel $_PUT seperti $_GET dan $_POST untuk mengambil nilai-nilai yang dilewatkan, kita dapat memanfaatkan input stream untuk memperoleh nilai-nilai tersebut unntuk megupdate suatu produk. Kita akan melihat bagaimana melewatkan nilai melalui input stream saat mengakses atau mengkonsumsi API.

```
function update_product($product_id) {
            global $connection;
            parse_str(file_get_contents("php://input"), $post_vars);
            $product_name = $post_vars["product_name"];
            $price = $post_vars["price"];
```

```
            $quantity = $post_vars["quantity"];
            $seller = $post_vars["seller"];

            $query = “UPDATE products SET product_name = '{$product_name}',”
                  . “ price = {$price}, quantity = {$quantity}, “
                  . “ seller = '{$seller}' WHERE id = ” . $product_id;

            if(mysqli_query($connection, $query)) {
                  $response = array(
                        'status' => 1,
                        'status_message' =>'Produk berhasil diupdate.'
                  );
            }
            else {
                  $response = array(
                        'status' => 0,
                        'status_message' =>'Produk GAGAL diupdate.'
                  );
            }

            header('Content-Type: application/json');
            echo json_encode($response);
      }
```

Menghapus Produk Tertentu

Pada penghapusan suatu produk, kita menggunakan fungsi delete_product(). Kita memanfaatkan product id dari produk yang akan dihapus dari variabel $_GET.

```
function delete_product($product_id) {
            global $connection;
            $query = "DELETE FROM products WHERE id = ".$product_id;

            if(mysqli_query($connection, $query)) {
                  $response=array(
                        'status' => 1,
                        'status_message' => 'Produk berhasil dihapus.'
                  );
            }
            else {
                  $response = array(
                        'status' => 0,
                        'status_message' => 'Produk GAGAL dihapus.'
                  );
            }

            header('Content-Type: application/json');
            echo json_encode($response);
      }
```

Jika kita mencermati fungsi-fungsi di atas, maka kita akan memahami bahwa kita menggunakan JSON untuk membentuk (*format*) data keluaran (*output*).

Kode Lengkap API dengan fungsi CRUD

Berikut ini adalah kode program lengkap dari file products.php:

```
	// menghubungi database
	$connection=mysqli_connect('localhost','root','','rest_api');

	$request_method=$_SERVER["REQUEST_METHOD"];
	switch($request_method) {
		case 'GET':
			// GET – mengambil informasi produk
			if(!empty($_GET["product_id"])) {
				$product_id=intval($_GET["product_id"]);
				get_products($product_id);
			}
			else {
				get_products();
			}
			break;

		case 'POST':
			// POST – menambahkan produkl baru
			insert_product();
			break;

		case 'PUT':
			// PUT - mengupdate produk tertentu
			$product_id=intval($_GET["product_id"]);
			update_product($product_id);
			break;

		case 'DELETE':
			// DELETE – menghapus produk tertentu
			$product_id=intval($_GET["product_id"]);
			delete_product($product_id);
			break;

		default:
			// metode request tidak valid (salah)
			header("HTTP/1.0 405 Metode Tidak Dikenali ");
			break;
	}

	function insert_product() {
		global $connection;
		$product_name=$_POST["product_name"];
		$price=$_POST["price"];
		$quantity=$_POST["quantity"];
		$seller=$_POST["seller"];

		$query="INSERT INTO products SET “ .
			“ product_name='{$product_name}', price={$price}, “ .
			“ quantity={$quantity}, seller='{$seller}'";
```

```
            if(mysqli_query($connection, $query)) {
                  $response=array(
                        'status' => 1,
                        'status_message' =>'Produk Berhasil Ditambahkan.'
                  );
            }
            else  {
                  $response=array(
                        'status' => 0,
                        'status_message' => Produk GAGAL Ditambahkan.'
                  );
            }

            header('Content-Type: application/json');
            echo json_encode($response);
      }

      function get_products($product_id=0) {
            global $connection;
            $query="SELECT * FROM products";

            if($product_id != 0) {
                  $query.=" WHERE id=".$product_id." LIMIT 1";
            }

            $response=array();
            $result=mysqli_query($connection, $query);

            while($row=mysqli_fetch_array($result)) {
                  $response[]=$row;
            }

            header('Content-Type: application/json');
            echo json_encode($response);
      }

      function delete_product($product_id) {
            global $connection;
            $query="DELETE FROM products WHERE id=".$product_id;

            if(mysqli_query($connection, $query)) {
                  $response=array(
                        'status' => 1,
                        'status_message' => ‘Produk Berhasil Dihapus.'
                  );
            }
            else  {
                  $response=array(
                        'status' => 0,
                        'status_message' =>' Produk GAGAL Dihapus '
                  );
            }
```

```
            header('Content-Type: application/json');
            echo json_encode($response);
      }

      function update_product($product_id) {
            global $connection;
            parse_str(file_get_contents("php://input"),$post_vars);
            $product_name=$post_vars["product_name"];
            $price=$post_vars["price"];
            $quantity=$post_vars["quantity"];
            $seller=$post_vars["seller"];

            $query = “UPDATE products SET product_name='{$product_name}', “
                  . “ price={$price}, quantity={$quantity}, “ .
                  . “ seller='{$seller}' WHERE id=” . $product_id;

            if(mysqli_query($connection, $query)) {
                  $response=array(
                        'status' => 1,
                        'status_message' =>' Produk Berhasil Diperbarui.'
                  );
            }
            else  {
                  $response=array(
                        'status' => 0,
                        'status_message' =>' Produk GAGAL Diperbarui.'
                  );
            }

            header('Content-Type: application/json');
            echo json_encode($response);
      }

      // Tutup koneksi database
      mysqli_close($connection);
```

Mengakses REST API Menggunakan PHP

Sejauh ini kita telah membangun API dan sekarang adalah saatnya memanfaatkan API tersebut, mengkonsumsinya melalui aplikasi atau client. Sebagaimana telah disebutkan sebelumnya, kita akan menggunakan pustaka cURL untuk mengakses API tersebut. Ada beberapa fungsi bawaan (*built in*) yag siap digunakan, di antaranya adalah:

a. Membangun koneksi ke provider : curl_init()
b. Menambahkan data request : curl_setopt()
c. Mengirimkan request : curl_exec()
d. Menutup koneksi : curl_close()

Meminta Semua Produk

Kode berikut digunakan untuk memperoleh informasi tentang semua produk. Kita melewatkan API URL ke fungsi curl_init() untuk membangun koneksi dengan server dan menyimpan *connection handle*-nya dalam variabel $ch. Di sini, kita menetapkan dua opsi menggunakan fungsi curl_setopt(). CURLOPT_HTTPGET digunakan untuk menunjukkan bahwa metode request HTTP adalah GET dan CURLOPT_RETURNTRANSFER digunakan untuk menandakan bahwa respon harus mengembalikan nilainya bukan menghasilkannya keluar secara langsung.

Maka dari itu request dikirim menggunakan fungsi curl_exec() dan menyimpan responnya dalam variabel $response_json. Terakhir, kita menutup koneksi menggunakan curl_close(). Karena respon akan berupa string JSON, kita perlu men-decode string tersebut dan mengubahnya ke suatu array PHP.

```
$url = 'http://example.com/api/products';
$ch = curl_init($url);

curl_setopt($ch, CURLOPT_HTTPGET, true);
curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);

$response_json = curl_exec($ch);
curl_close($ch);
$response = json_decode($response_json, true);
```

Meminta Produk Tertentu

Kode berikut digunakan untuk mendapatkan informasi tentang satu produk tertentu dan ini sangat mirip dengan kode untuk memperoleh informasi mengenai semua produk. Kita melewatkan nilai 5 untuk me-retrieve produk dengan *primary key* (kunci utama) 5.

```
$url = 'http://example.com/api/products/5';
$ch = curl_init($url);

curl_setopt($ch, CURLOPT_HTTPGET, true);
curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);

$response_json = curl_exec($ch);
curl_close($ch);
$response = json_decode($response_json, true);
```

Menambahkan Produk Baru

Kode berikut digunakan untuk menambahkan suatu produk baru. Kali ini kita telah menambahkan dua opsi cURL baru. Opsi CURLOPT_POST digunakan untuk menunjukkan bahwa metode request HTTP adalah POST dan CURLOPT_POSTFIELDS digunakan untuk menempelkan data POST-nya.

```
$data = array(
            'product_name' => 'Television',
            'price' => 1000,
            'quantity' => 10,
            'seller' => 'XYZ Traders'
);
```

```
$url = 'http://example.com/api/products';
$ch = curl_init($url);

curl_setopt($ch, CURLOPT_POST, true);
curl_setopt($ch, CURLOPT_POSTFIELDS, $data);
curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);

$response_json = curl_exec($ch);
curl_close($ch);
$response=json_decode($response_json, true);
```

Mengupdate Produk Tertentu

Kode berikut digunakan untuk mengupdate suat produk. Kita menggunakan opsi CURLOPT_CUSTOMREQUEST untuk menentukan bahwa metode request HTTP adalah PUT. Karena tidak ada konstanta spesifik untuk menempelkan data PUT menggunakan fungsi curl_setopt() maka kita menggunakan opsi CURLOPT_POSTFIELDS yang digunakan dalam request POST. Tetapi kali ini kita tidak melewatkan data sebagai suatu array. Sebagai gantinya, kita lewatkan data sebagai suatu string query menggunakan fungsi http_build_query(). Pemanggilan API demikian akan mengupdate produk dengan kunci utama 3.

```
$data = array(
            'product_name' => 'Laptop',
            'price' => 1200,
            'quantity' => 15,
            'seller' => 'ABC Trading Inc.'
);

$url = 'http://example.com/api/products/3';
$ch = curl_init($url);

curl_setopt($ch, CURLOPT_CUSTOMREQUEST, "PUT");
curl_setopt($ch, CURLOPT_POSTFIELDS, http_build_query($data));
curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);

$response_json = curl_exec($ch);
curl_close($ch);
$response=json_decode($response_json, true);
```

Menghapus Produk Tertentu

Kode berikut digunakan untuk menghapus produk tertentu. Sebagaimana dapat kita lihat, kita menggunakan opsi CURLOPT_CUSTOMREQUEST untuk menetapkan metode request DELETE HTTP dan pemanggilan API ini akan menghapus produk dengan kunci utama 7.

```
$url = 'http://example.com/api/products/7';
$ch = curl_init($url);

curl_setopt($ch, CURLOPT_CUSTOMREQUEST, 'DELETE');
curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);

$response_json = curl_exec($ch);
```

```
curl_close($ch);
$response=json_decode($response_json, true);
```

Rangkuman

Tentu saja, masih ada banyak fitur REST API diluar yang telah dibahas dalam tutorial singkat dan sederhana ini. Gagasan dibalik artikel ini adalah memberikan kita suatu dasar sekaligus juga fondasi yang kuat mengenai REST API. Kami berharap bahwa kita sekarang menjadi lebih jelas dalam memahami apa itu REST API dan bagaimana membangunnya sendiri dari awal, tanpa framework apapun kecuai built-in yang hadir bersama PHP. Silakan share pengalaman dan pertanyaan anda ke husni@trunojoyo.ac.id. Mohon maaf atas kekurangan di dalam artikel ini, semoga bermanfaat ☺.

Sebagian besar ini tutorial ini adalah terjemahan dari halaman web:

https://www.apptha.com/blog/how-to-build-a-rest-api-using-php/