



Pada modul kali ini kita akan mencoba melakukan contoh *REST API CRUD* di PHP dan MongoDB dengan RESTful API. CRUD disini adalah operasi *create*, *read*, *update*, dan *delete*, dimana operasi tersebut akan menambahkan data baru (*create*), mengambil data (*read*), mengubah data yang ada (*update*) dan menghapus data yang ada (*delete*) dari database. REST atau RESTful API sekarang menjadi framework paling populer karena kinerjanya dengan dukungan berbagai jenis format (teks, html, json, dll.).

Di modul ini akan dibuat layanan REST dalam bahasa pemrograman PHP dan melakukan operasi untuk membaca, menyisipkan, memperbarui, dan menghapus data pada database MongoDB. Pastikan pada komputer anda sudah terinstall beberapa aplikasi berikut ini yaitu Apache HTTP Server, PHP, MongoDB (versi yang digunakan pada modul ini adalah Apache HTTP Server 2.4, PHP 7.0.3 dan MongoDB 5.0.3, jika ingin menggunakan versi yang lain silakan disesuaikan saja bisa dicek disini untuk kompatibilitas php dan MongoDB yang ingin digunakan, <https://www.mongodb.com/docs/drivers/php/>), Postman atau alat klien REST lainnya (Apache HTTP Server dan PHP bisa menggunakan paket instalasi Xampp, Wampp atau Laragon).

### A. Project Directory Setup

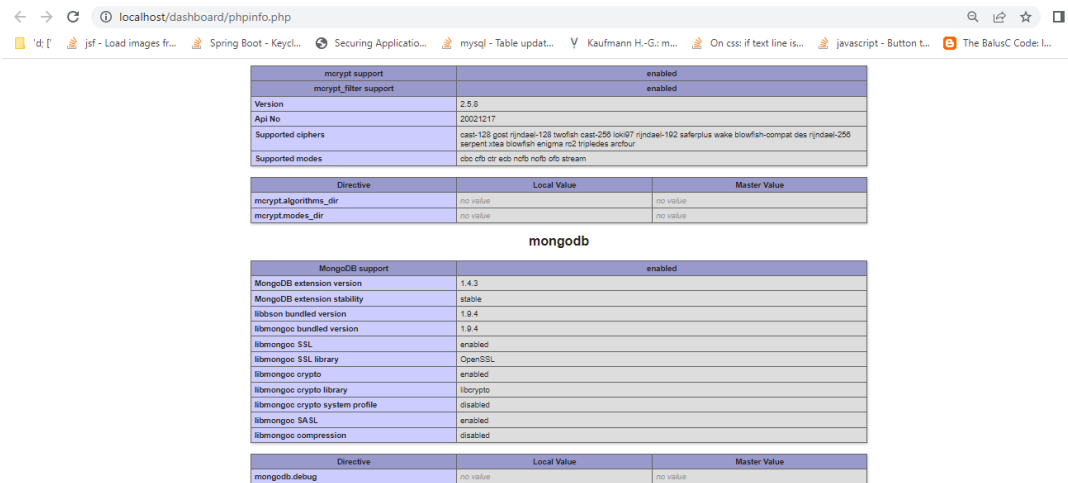
Langkah pertama adalah memeriksa apakah database MongoDB sudah berfungsi dan berjalan dengan baik. Setelah menemukan instance MongoDB sedang berjalan maka coba untuk terhubung ke server MongoDB menggunakan klien MongoDB (MongoDB Compass) di mana nanti bisa memeriksa koleksi dokumen, data dan melakukan query. Selanjutnya silakan di periksa juga apakah server http sudah berjalan dengan baik menggunakan panel kontrol xampp, wampp, atau aplikasi sejenis. Ketika persyaratan di atas terpenuhi maka silakan dibuat beberapa kode PHP untuk membangun REST API. Jadi terlebih dahulu navigasikan ke dalam direktori <lokasi drive fisik>:/xampp/htdocs dan membuat folder bernama ***php-mongodb-rest-api-crud***. Nantinya kita akan menggunakan folder tersebut untuk menyimpan file php yang dibuat untuk contoh REST API CRUD.

Selanjutnya unduh driver MongoDB PHP untuk menjalin komunikasi antara kedua teknologi ini (disesuaikan dengan versi PHP nya). Untuk modul ini pilih file 7.0 Thread Safe (TS) x64 ( <https://pecl.php.net/package/mongodb/1.3.3/windows> ). Kemudian ekstrak arsip dan salin file php\_mongodb.dll ke direktori /xampp/php/ext. Matikan service apache server kemudian buka

file `/xampp/php/php.ini` dan tambahkan baris di bawah ini agar bisa melakukan koneksi ke database MongoDB.

```
extension=php_mongodb.dll
```

Jalankan kembali service apache server lalu cek apakah driver MongoDB sudah terinstall dengan cara buka web browser lalu ketikan url berikut maka akan tampil seperti ini.



Directive	Local Value	Master Value
mongodb.debug	no value	no value

## B. Database Connection

Hal pertama dan terpenting adalah menghubungkan ke MongoDB disini adalah membuat class PHP untuk membuat koneksi database untuk melakukan operasi CRUD. Buat file ***mongodb\_config.php*** dan masukkan kode sumber di bawah ini lalu letakan pada folder ***php-mongodb-rest-api-crud***.

```
1 <?php
2
3 class DbManager {
4
5     //Database configuration
6     private $dbhost = 'localhost';
7     private $dbport = '27017';
8     private $conn;
9
10    function __construct(){
11        //Connecting to MongoDB
12        try {
13            //Establish database connection
14            $this->conn = new MongoDB\Driver\Manager('mongodb://' . $this->dbhost . ':' . $this->dbport);
15        } catch (MongoDB\Driver\Exception $e) {
16            echo $e->getMessage();
17            echo nl2br("\n");
18        }
19    }
20
21    function getConnection() {
22        return $this->conn;
23    }
24
25 }
26
27 >>
```

Pada skrip diatas dibuat class DbManager yang bertanggung jawab untuk membuat koneksi ke MongoDB dengan php. Di sini telah mengambil host database, port dan objek koneksi sebagai variabel instan dan melalui konstruktor akan mencoba untuk terhubung ke Database

Mongo dan untuk setiap kegagalan akan dikirim ke pengecualian. Skrip tersebut ditambahkan juga fungsi `getConnection()` yang akan memberikan objek koneksi yang akan digunakan untuk melakukan operasi CRUD pada database Mongo.

## C. REST Resource – Create

### 1. Create

Buat file *rest\_resource\_create.php* dan salin kode sumber di bawah ini ke dalamnya lalu masukan ke dalam folder *php-mongodb-rest-api-crud*.

```
1 <?php
2
3 // required headers
4 header("Access-Control-Allow-Origin: *");
5 header("Content-Type: application/json; charset=UTF-8");
6 header("Access-Control-Allow-Methods: POST");
7 header("Access-Control-Max-Age: 3600");
8 header("Access-Control-Allow-Headers: Content-Type, Access-Control-Allow-Headers, Authorization, X-Requested-With");
9
10 // include database file
11 include_once 'mongodb_config.php';
12
13 $dbname = 'toko';
14 $collection = 'barang';
15
16 //DB connection
17 $db = new DbManager();
18 $conn = $db->getConnection();
19
20 //record to add
21 $data = json_decode(file_get_contents("php://input", true));
22
23 // insert record
24 $insert = new MongoDB\Driver\BulkWrite();
25 $insert->insert($data);
26
27 $result = $conn->executeBulkWrite("$dbname.$collection", $insert);
28
29 // verify
30 if ($result->getInsertedCount() == 1) {
31     echo json_encode(
32         array("message" => "Record successfully created")
33     );
34 } else {
35     echo json_encode(
36         array("message" => "Error while saving record")
37     );
38 }
39
40 ?>
```

Di sini perhatikan pada skrip telah menambahkan beberapa header yaitu :

"Access-Control-Allow-Origin: \*" - ini menunjukkan permintaan diterima dari mana saja

"Content-Type: application/json; charset=UTF-8" - ini menunjukkan bahwa permintaan adalah data json

"Access-Control-Allow-Methods: POST" - ini menunjukkan bahwa hanya permintaan POST yang diizinkan

Selanjutnya kita membutuhkan class `DbManager`, jadi kita menyertakan file **mongodb\_config.php** di dalamnya. Kemudian definisikan nama database dan nama *collection* untuk melakukan query (database dan koleksi menggunakan materi modul sebelumnya yaitu database toko dengan *collection* barang). Selanjutnya buat objek

## Modul Praktikum 7 : PHP-MongoDB

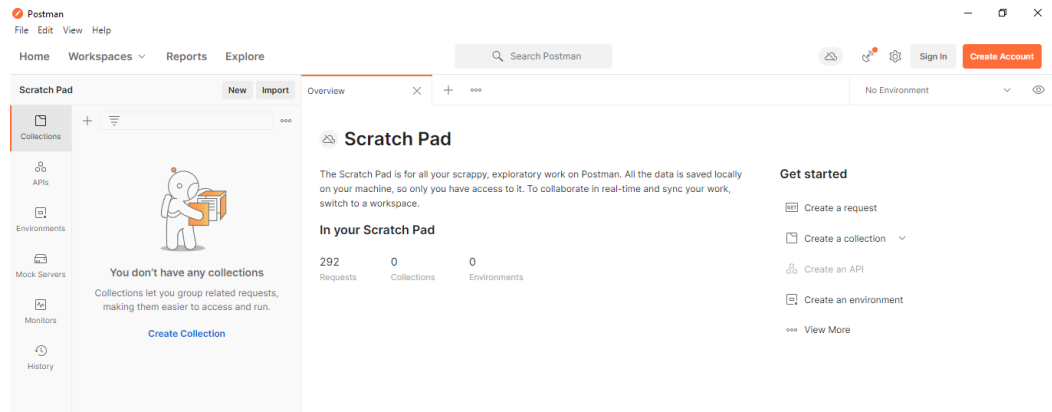
DbManager() baru untuk mendapatkan objek koneksi. Selanjutnya kita membaca data request json menggunakan baris berikut:

```
$data = json_decode(file_get_contents("php://input", true));
```

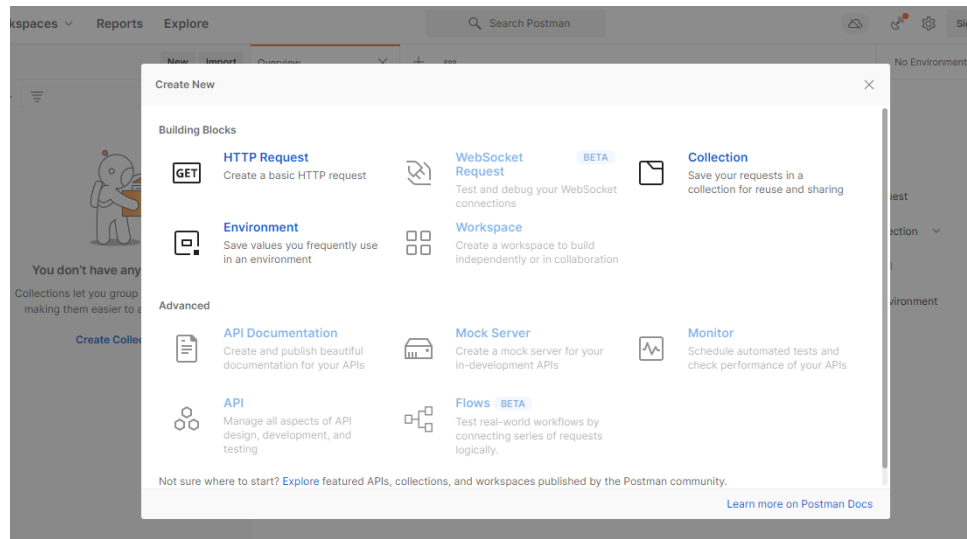
php://input adalah aliran read-only yang memungkinkan membaca data mentah dari request body. Kemudian buatlah instance BulkWrite() dan masukan data permintaan json ke database Mongo menggunakan metode executeBulkWrite().

### 2. Pengujian Fungsi Create

Buka aplikasi postman



Pilih new dan HTTP Request

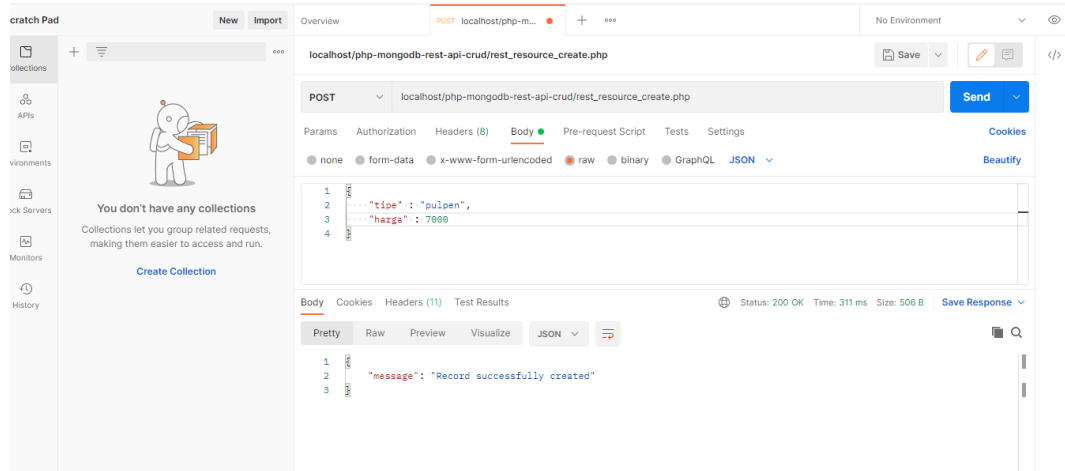


Kemudian isi url sesuai gambar dibawah ini yaitu

**localhost/php-mongodb-rest-api-crud/rest\_resource\_create.php**

lalu pilih fungsi **POST**, pilih body, raw dengan format JSON isi sesuai gambar, lalu klik **SEND** untuk menguji operasi Create menggunakan klien REST.

## Modul Praktikum 7 : PHP-MongoDB



Sekarang periksa MongoDB masuk ke dalam database toko kemudian gunakan perintah `db.barang.find()`, Anda akan menemukan satu record berhasil dimasukkan :

```
> db.barang.find()
{ "_id" : ObjectId("63030380bbdbe7559c122e22"), "harga" : 20000, "tipe" : "buku" }
{ "_id" : ObjectId("6303053cbdbbe7559c122e24"), "harga" : 2500, "tipe" : "pensil" }
{ "_id" : ObjectId("6303056bbdbe7559c122e25"), "harga" : 1000, "tipe" : "penghapus" }
{ "_id" : ObjectId("630349f3d6c89c26b4005843"), "tipe" : "pulpen", "harga" : 7000 }
```

### D. REST Resource – READ

#### 1. Read

Sekarang kita akan melihat bagaimana operasi Baca dalam contoh CRUD. Buat file *rest\_resource\_read.php* dan masukkan kode sumber di bawah ini ke dalamnya.

```
<?php
// required headers
header("Access-Control-Allow-Origin: *");
header("Content-Type: application/json; charset=UTF-8");

// include database file
include_once 'mongodb_config.php';

$dbname = 'toko';
$collection = 'barang';

//DB connection
$db = new DbManager();
$conn = $db->getConnection();

// read all records
$filter = [];
$options = [];
$read = new MongoDB\Driver\Query($filter, $options);

//fetch records
$records = $conn->executeQuery("$dbname.$collection", $read);

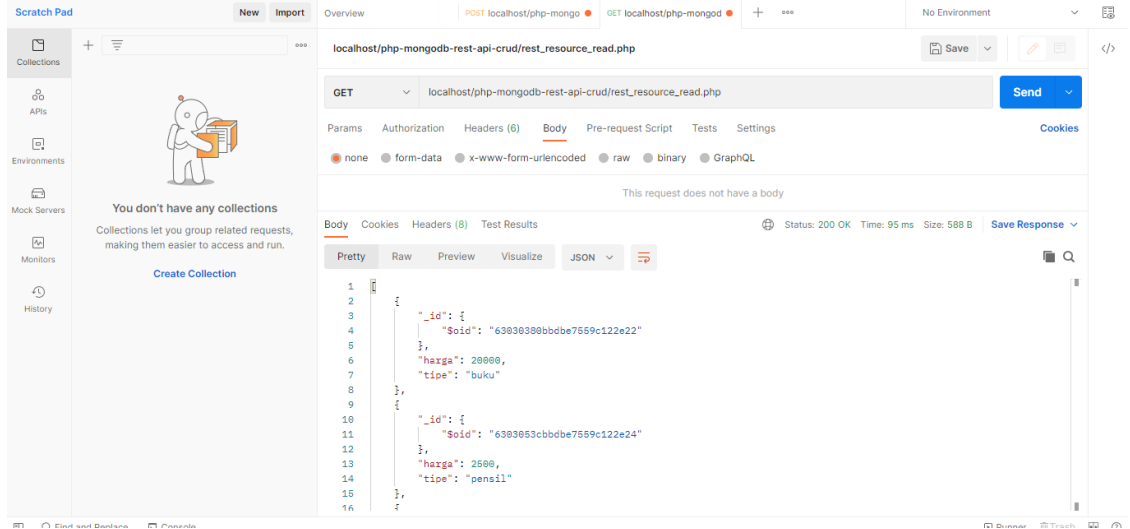
echo json_encode(iterator_to_array($records));
```

Di sini juga telah ditambahkan beberapa header dan data respons di json. Disini juga belum menambahkan metode http apa pun, jadi secara default metode http adalah GET request. Perhatikan dalam file ini ada array `filter[]` dan `option[]` yang digunakan untuk membatasi kueri untuk memilih record berdasarkan filter dan nilai opsi. Disini kita menggunakan metode `executeQuery()` untuk membaca data dari MongoDB. Disini menampilkan format sebagai json dan sebagai kueri mengembalikan beberapa

baris, jadi menggunakan metode `iterator_to_array()` di sini untuk mengulangi array json.

### 2. Pengujian Fungsi Create

Sekarang coba diuji operasi Read dalam contoh CRUD.



## E. REST Resource – UPDATE

### 1. UPDATE

Sekarang akan dilihat bagaimana cara mengerjakan operasi Update contoh REST API CRUD di PHP dan MongoDB. Buat file **rest\_resource\_update.php** dengan kode sumber di bawah ini.

```

1 <?php
2
3 // required headers
4 header("Access-Control-Allow-Origin: *");
5 header("Content-Type: application/json; charset=UTF-8");
6 header("Access-Control-Allow-Methods: PUT");
7 header("Access-Control-Max-Age: 3600");
8 header("Access-Control-Allow-Headers: Content-Type, Access-Control-Allow-Headers, Authorization, X-Requested-With");
9
10 // include database file
11 include_once 'mongodb_config.php';
12
13 $dbname = 'toko';
14 $collection = 'barang';
15
16 //DB connection
17 $db = new DbManager();
18 $conn = $db->getConnection();
19
20 //record to update
21 $data = json_decode(file_get_contents("php://input", true));
22
23 $fields = $data->{'fields'};
24
25 $set_values = array();
26
27 foreach ($fields as $key => $fields) {
28     $arr = (array)$fields;
29     foreach ($fields as $key => $value) {
30         $set_values[$key] = $value;
31     }
32 }

```

```

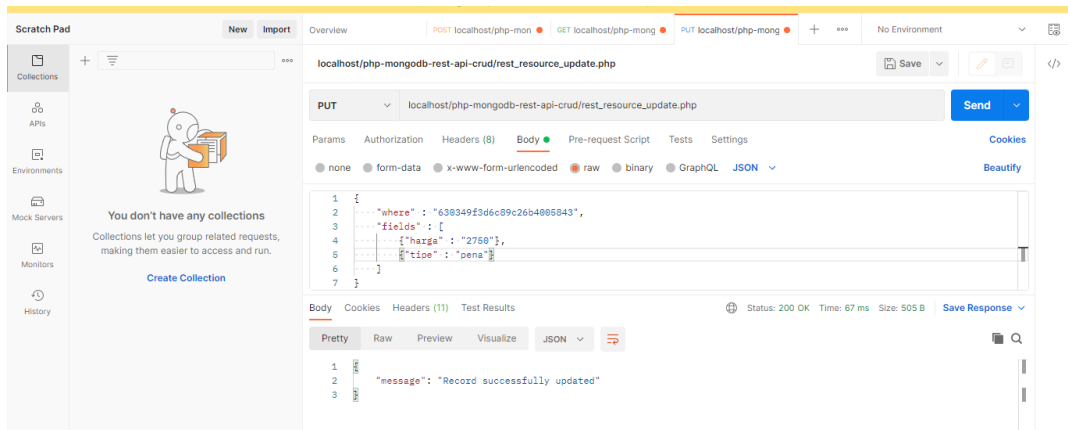
33
34 // _id field value
35 $id = $data->['where'];
36
37 // update record
38 $update = new MongoDB\Driver\BulkWrite();
39 $update->update(
40     ['_id' => new MongoDB\BSON\ObjectId($id)], ['$set' => $set_values], ['multi' => false, 'upsert' => false]
41 );
42
43 $result = $conn->executeBulkWrite("$dbname.$collection", $update);
44
45 // verify
46 if ($result->getModifiedCount() == 1) {
47     echo json_encode(
48         array("message" => "Record successfully updated")
49     );
50 } else {
51     echo json_encode(
52         array("message" => "Error while updating record")
53     );
54 }
55
56

```

Dalam skrip di atas telah ditambahkan beberapa header seperti fungsi Create sebelumnya tetapi disini ditambahkan metode http PUT di sini untuk permintaan klien. Disini untuk membaca permintaan json menggunakan cara yang sama seperti pada fungsi Create. Disini juga dibuat array pembaruan dan klausa where untuk melakukan operasi pembaruan. Kemudian untuk mengeksekusi query menggunakan metode executeBulkWrite() dan memverifikasi hasil pembaruan dan menampilkan hasilnya.

### 2. Pengujian Fungsi Update

Sekarang coba diuji operasi Update dengan REST API CRUD di PHP dan MongoDB. Di sini melakukan Update di sini harga dan tipe untuk \_id yang diberikan.



Sekarang periksa databasenya. Anda melihat harga dan tipe telah berhasil diubah.

```

> db.barang.find()
{ "_id" : ObjectId("63030380bbdb7559c122e22"), "harga" : 20000, "tipe" : "buku" }
{ "_id" : ObjectId("6303053cbbdb7559c122e24"), "harga" : 2500, "tipe" : "pensil" }
{ "_id" : ObjectId("6303056bbdb7559c122e25"), "harga" : 1000, "tipe" : "penghapus" }
{ "_id" : ObjectId("630349f3d6c89c26b4005843"), "tipe" : "pena", "harga" : "2750" }
>

```

### F. REST Resource – DELETE

#### 1. DELETE

Berikut contoh operasi Hapus manajemen CRUD. Buatlah file *rest\_resource\_delete.php* dan masukkan kode sumber di bawah ini ke dalamnya.

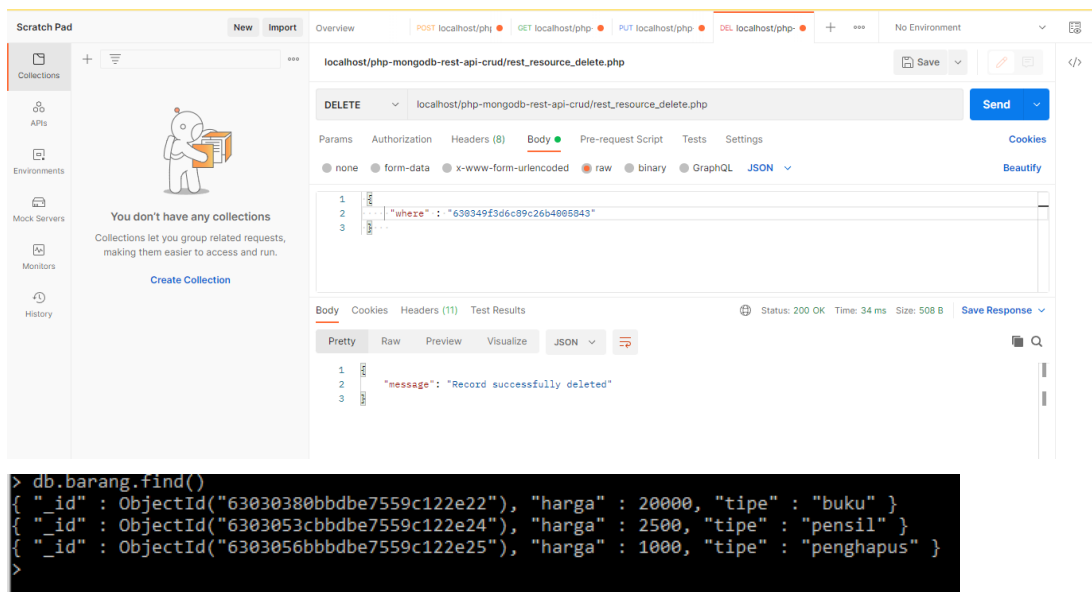
```

1 <?php
2
3 // required headers
4 header("Access-Control-Allow-Origin: *");
5 header("Content-Type: application/json; charset=UTF-8");
6 header("Access-Control-Allow-Methods: DELETE");
7 header("Access-Control-Max-Age: 3600");
8 header("Access-Control-Allow-Headers: Content-Type, Access-Control-Allow-Headers, Authorization, X-Requested-With");
9
10 // include database file
11 include_once 'mongodb_config.php';
12
13 $dbname = 'toko';
14 $collection = 'barang';
15
16 //DB connection
17 $db = new DbManager();
18 $conn = $db->getConnection();
19
20 //record to delete
21 $data = json_decode(file_get_contents("php://input", true));
22
23 //_id field value
24 $id = $data->['where'];
25
26 // delete record
27 $delete = new MongoDB\Driver\BulkWrite();
28 $delete->delete(
29     ['_id' => new MongoDB\BSON\ObjectId($id)],
30     ['limit' => 0]
31 );
32
33 $result = $conn->executeBulkWrite("$dbname.$collection", $delete);
34
35 //print_r($result);
36
37 // verify
38 if ($result->getDeletedCount() == 1) {
39     echo json_encode(
40         array("message" => "Record successfully deleted")
41     );
42 } else {
43     echo json_encode(
44         array("message" => "Error while deleting record")
45     );
46 }
47
48

```

Disini ditambahkan metode http DELETE untuk operasi penghapusan.

#### 2. Pengujian Fungsi Delete



The screenshot shows a REST client interface with a DELETE request to `localhost/php-mongodb-rest-api-crud/rest_resource_delete.php`. The request body is a JSON object: `{ "where": "630349f3d6c09c26b4085843" }`. The response status is 200 OK, and the response body is a JSON object: `{ "message": "Record successfully deleted" }`.

```

> db.barang.find()
{ "_id" : ObjectId("63030380bbdbe7559c122e22"), "harga" : 20000, "tipe" : "buku" }
{ "_id" : ObjectId("6303053cbbdbe7559c122e24"), "harga" : 2500, "tipe" : "pensil" }
{ "_id" : ObjectId("6303056bbdbe7559c122e25"), "harga" : 1000, "tipe" : "penghapus" }
>

```



----- *Sekian Terima Kasih* -----

- <https://docs.mongodb.com/manual/installation/>
- <https://www.mongodb.com/try/download/database-tools>
- <https://roytuts.com/rest-api-crud-example-in-php-and-mongodb/>