MODUL 14
MEMBANGUN WEB SERVICE RESTFul API LARAVEL (2)

CAPAIAN PEMBELAJARAN

1. Mampu membuat aplikasi Web Service menggunakan RESTful API menggunakan Laravel
2. Mampu menguji pertukaran dengan aplikasi Postman
3. Mampu menguji pertukaran data antar server dengan format JSON menggunakan curl PHP

KEBUTUHAN ALAT/BAHAN/SOFTWARE

1. Sistem Operasi Linux/Window 10
2. PHP 5.x, diutamakan versi 7.x keatas
3. Database MySQL
4. Web server Apache
5. Composer
6. Text Editor Notepad++ atau Visual Studio Code
7. Postman
8. Jaringan lokal
9. 2 unit web server

DASAR TEORI

4. Menggunakan Kelas ixudrea/Curl

Untuk mengakses web service menggunakan RESTful disederhanakan menggunakan kelas `ixudra/Curl`. Paket ini menyediakan antarmuka yang mudah untuk mengirim permintaan cURL dari aplikasi web menggunakan PHP. Perintahnya mirip dengan perintah Query Builder di Laravel. Selain itu, menyediakan beberapa method utilitas yang memungkinkan dengan mudah menambahkan opsi tertentu ke permintaan.

5. Mengirim permintaan GET

Untuk mengirim permintaan GET, harus menggunakan metode `get()` yang disediakan oleh paket:

Contoh penggunaan:

```
use Ixudra\Curl\Facades\Curl;

    //mengirim permintaan GET ke: http://www.contoh.com/api
    $respon = Curl::to('http://www.contoh.com/api')
              ->get();
```

6. Mengirim permintaan POST

Permintaan dengan method `post()`, bekerja mirip dengan permintaan GET, cara penggunaan method `post()` seperti pada contoh program berikut:

```
use Ixudra\Curl\Facades\Curl;

    //mengirim permintaan POST ke: http://www.contoh.com/api
    $respon = Curl::to('http://www.contoh.com/api')
              ->post();
```

7. Mengirim permintaan PUT

Permintaan dengan menggunakan method `put()`, bekerja mirip dengan permintaan POST, cara penggunaan method `put()` seperti pada contoh program berikut:

```
use Ixudra\Curl\Facades\Curl;

  //mengirim permintaan PUT ke: http://www.contoh.com/api/1
  //dengan argumen 'no' = '1' menggunakan JSON
    $respon = Curl::to('http://www.contoh.com/api/1')
              ->withData(array('no' => '1'))
              ->asJson()
              ->put();
```

8. Mengirim permintaan PATCH

Permintaan dengan menggunakan method `patch()` bekerja mirip dengan permintaan `POST`, cara penggunaan method `patch()` seperti pada contoh program berikut:

```
use Ixudra\Curl\Facades\Curl;
```

```
    //mengirim permintaan PATCH ke: http://www.contoh.com/api/1
    //dengan argumen 'no' = '1' menggunakan JSON
    $respon = Curl::to('http://www.contoh.com/api/1')
              ->withData(array( 'no' => '1' ))
              ->asJson()
              ->patch();
```

9. Mengirim permintaan DELETE

Permintaan dengan menggunakan method `delete()`, bekerja hampir sama dengan permintaan GET, cara menggunakan method `delete()`, seperti pada contoh program berikut:

```
use Ixudra\Curl\Facades\Curl;

    // mengirim permintaan DELETE ke:
    // http://www.contoh.com/api/1
    // menggunakan JSON
    $respon= Curl::to('http://www.contoh.com/api/1')
              ->asJson()
              ->delete();
```

10. Mengirim file melalui Curl

Untuk mengirim file melalui permintaan `POST`, bisa menggunakan method `withFile()`. Contoh penggunaan method `withFile()` seperti pada kode program berikut:

```
use Ixudra\Curl\Facades\Curl;

 $respon = Curl::to('http://www.contoh.com/api')
        ->withData(array('no' =>'1'))
        ->withFile('image_1',
               '/path/to/dir/image1.png','image/png',
               'imageName1.png' )
        ->withFile('image_2',
               '/path/to/dir/image2.png','image/png',
               'imageName2.png' )
        ->post();
```

11. Mengunduh file

Untuk mengunduh file, Anda dapat menggunakan method `download()`. Contoh penggunaan method `download()`, seperti pada program berikut:

```
use Ixudra\Curl\Facades\Curl;

    // Download gambar file dari:
    // http://www.contoh.com/badi.png

    $respon = Curl::to('http://www.contoh.com/badi.png')
              ->withContentType('image/png')
              ->download('/path/to/dir/image.png');
```

PRAKTIK

1. **Instalasi paket ixudra/Curl**
   Cara instalasi bisa dengan cara mengubah dan menambahkan file `composer.json` dengan perintah seperti berikut:

```
{
     "require": {
            "ixudra/curl": "6.*"
        }
    }
```

2. Atau menggunakan perintah *command* seperti berikut:

   ```
   composer require ixudra/curl
   ```

3. Setelah berhasil menginstal paket, buka file `config/app.php` dan tambahkan penyedia layanan dan alias.

```
'providers' => [
    ......
    Ixudra\Curl\CurlServiceProvider::class,
    ...
],

'aliases' => [
    .....
    'Curl' => Ixudra\Curl\Facades\Curl::class,
]
```

4. Buatlah Controller baru menggunkan nama `KotaClientCurlController`, dengan artisan berikut;

```
php artisan make:controller KotaClientCurlContoller --resource
```

2. Buka `KotaClientCurlContoller.php` kemudian tambahkan seperti pada kode program berikut:

```php
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;
use Ixudra\Curl\Facades\Curl;

class KotaClientCurlController extends Controller
{
    /**
     * Display a listing of the resource.
     *
     * @return \Illuminate\Http\Response
     */
    public function index()
    {
     $data = Curl::to('http://localhost:8000/api/kota/')
                    ->get();
     $kota=json_decode($data);
     return view('kotaClientCurl.index',compact('kota'));
     }

    /**
     * Show the form for creating a new resource.
     *
     * @return \Illuminate\Http\Response
     */
    public function create()
    {
        return view('kotaClientCurl.create');
    }

    /**
     * Store a newly created resource in storage.
     *
     * @param  \Illuminate\Http\Request  $request
     * @return \Illuminate\Http\Response
     */
    public function store(Request $request)
    {
        Curl::to('http://localhost:8000/api/kota/create')
           ->withData(['propinsi_id' => $request->propinsi_id,
                    'nama_kota' => $request->nama_kota,
                    'user_id'=>2,])
         ->asJson(true)
```

```
        ->post();

     //kembalikan ke tampilan tabel
       $data = Curl::to('http://localhost:8000/api/kota/')
                     ->get();
       $kota=json_decode($data);
     return view('kotaClientCurl.index',compact('kota'));

    }

    /**
     * Display the specified resource.
     *
     * @param  int  $id
     * @return \Illuminate\Http\Response
     */
    public function show($id)
    {

    }

    /**
     * Show the form for editing the specified resource.
     *
     * @param  int  $id
     * @return \Illuminate\Http\Response
     */
    public function edit($id)
    {
      $data = Curl::to('http://localhost:8000/api/kota/'.$id)
                  ->get();
      $kota=json_decode($data);
      return view('kotaClientCurl.edit',compact('kota'));

    }

    /**
     * Update the specified resource in storage.
     *
     * @param  \Illuminate\Http\Request  $request
     * @param  int  $id
     * @return \Illuminate\Http\Response
     */
    public function update(Request $request, $id)
    {
        Curl::to('http://localhost:8000/api/kota/'.$id)
           ->withData(['propinsi_id' => $request->propinsi_id,
                   'nama_kota' => $request->nama_kota,])
           ->asJson(true)
           ->put();

       $data = Curl::to('http://localhost:8000/api/kota/')
                     ->get();
       $kota=json_decode($data);
```

```
     return view('kotaClientCurl.index',compact('kota'));

    }

    /**
     * Remove the specified resource from storage.
     *
     * @param  int  $id
     * @return \Illuminate\Http\Response
     */
    public function destroy($id)
    {
        //
    }
}
```

3. Buatlah view simpan ke folder views/kotaClientCurl/… :
   - Views index.blade.php
   - Views create.blade.php
   - Views edit.blade.php
   - Views show.blade.php

4. Tambah `routes/web.php` seperti pada skrip berikut:

```
Route::get('kotaClientCurl/', 'KotaClientCurlController@index')
       ->name('kotaClientCurl.index');

Route::get('kotaClientCurl/create',
                 'KotaClientCurlController@create')
       ->name('kotaClientCurl.create');

Route::post('kotaClientCurl/store','KotaClientCurlController@store')
       ->name('kotaClientCurl.store');

Route::get('kotaClientCurl/{id}/edit',
                   'KotaClientCurlController@edit')
        ->name('kotaClientCurl.edit');

Route::get('kotaClientCurl/{id}/show',
                   'KotaClientCurlController@show')
       ->name('kotaClientCurl.show');

Route::delete('kotaClientCurl/{id}/destroy',
                   'KotaClientCurlController@destroy')
       ->name('kotaClientCurl.destroy');

Route::put('kotaClientCurl/{id}/update',
                    'KotaClientCurlController@update')
       ->name('kotaClientCurl.update');
```

5. Ujikan dengan dua server secara lokal/virtual, gunakan c*ommand prompt*

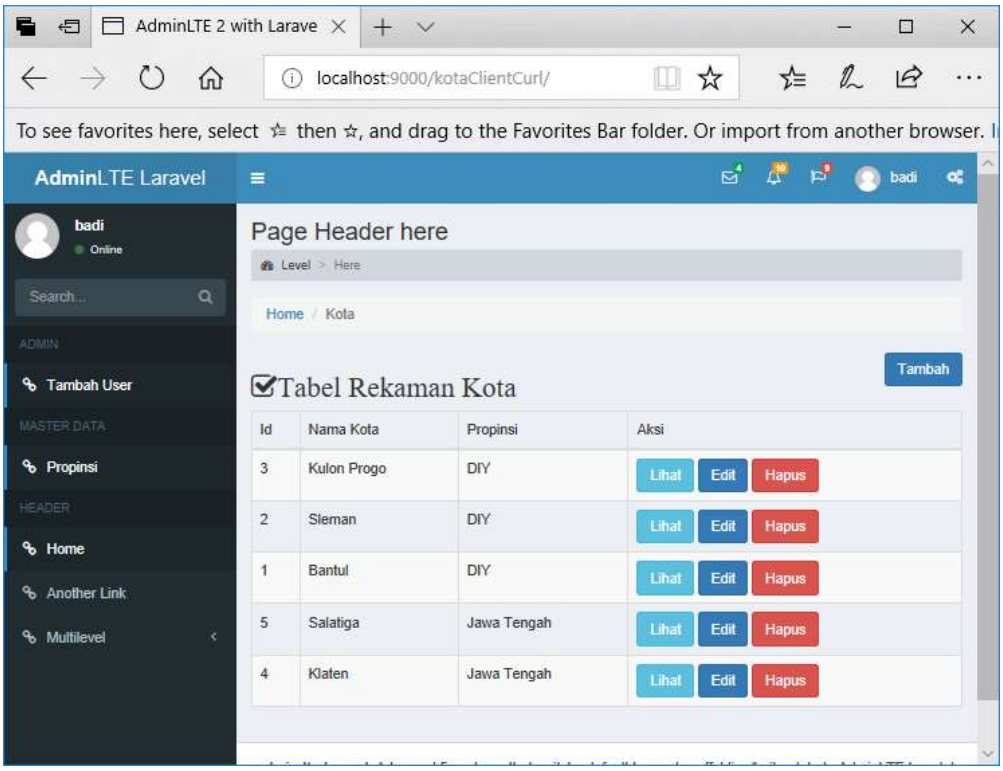
   Sebagai Server penyedia web service seperti berikut:

```
C:\xampp\htdocs\sipusta>php -S localhost:8000 -t public
PHP 7.1.11 Development Server started at Wed Aug 14 09:56:01 2019
Listening on http://localhost:8000
```

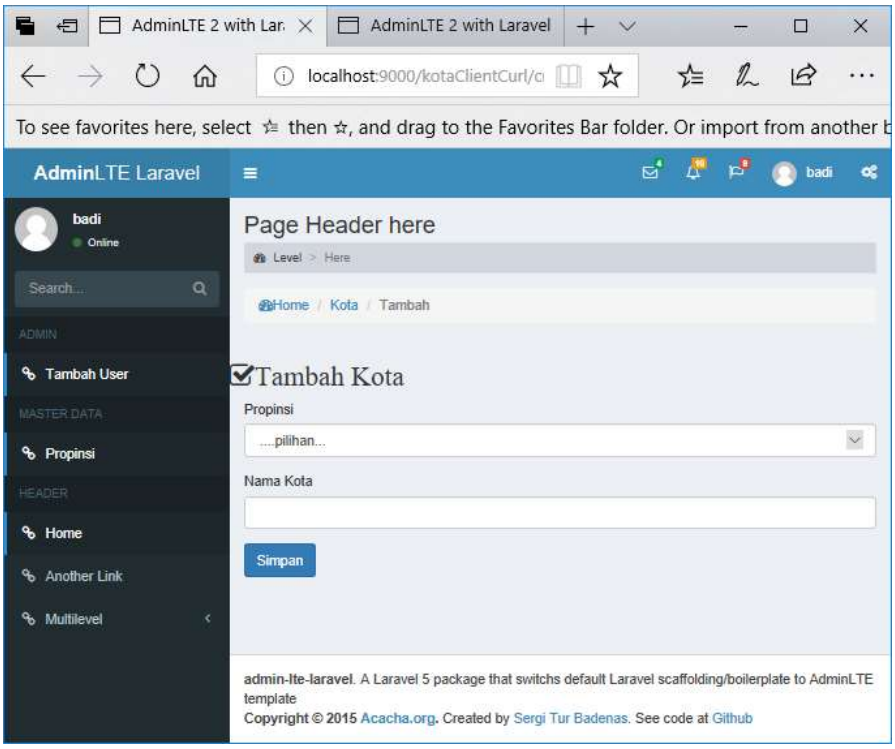
   Sebagai Server mengakses web service seperti berikut:

```
C:\xampp\htdocs\proyek1>php -S localhost:9000 -t public
PHP 7.1.11 Development Server started at Wed Aug 14 15:17:23 2019
Listening on http://localhost:9000
```

6. Menguji membaca semua rekaman dengan url http://localhost:9000/kotaClientCurl/, seperti pada gambar berikut:



7. Menguji menambah rekaman dengan url http://localhost:9000/kotaClientCurl/create, seperti pada gambar berikut:



LATIHAN

1. Sebuah instansi kependudukan telah menyediakan tabel rekaman kependudukan, terdiri:

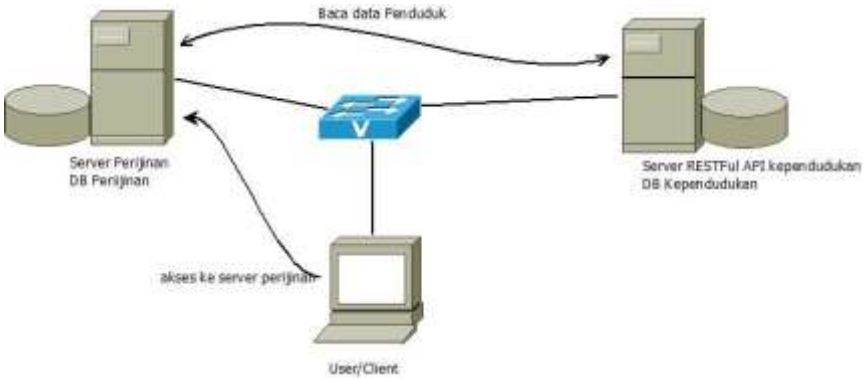
   NIK, nama, tempat lahir, tanggal lahir, jenis kelamin, gol darah, alamat RT/RW, Kel/Desa, Kecamatan, Agama, Status perkawinan [kawin, belum kawin], Pekerjaan [PNS, TNI/POLRI, swasta, Petani, Nelayan, Buruh], Kewarga negaraan [WNI, WNA], berlaku sampai tanggal.(Lihat data KTP)

   Buat Web service RESTFul API, agar supaya bisa diakses oleh instansi lain yang membutuhkan data kependudukan.

2. Buatlah proto tipe web untuk proses perizinan, terdiri dari:

`NIK, nama, alamat, pekerjaan` membaca data dari web service (kependudukan), dan `tanggal permohonan, jenis usaha [warnet, mini market, warung makan, salon, …], lokasi, luas area,`

8. Gunakan jaringan Lokal seperti gambar berikut, *ip address* menyesuaikan, 2 komputer server dan 1 client. Kerjakan kelompok 3 orang



TUGAS

1. Bagian Kepolisian akan melakukan perekaman untuk pembuatan SIM, membutuhkan data kependudukan. Buatlah aplikasi untuk perekaman data SIM.
2. Bagian SAMAT untuk pembayaran pajak membutuhkan data KTP dari kependudukan. Buatlah program untuk pembayaran pajak kendaraan bermotor.

REFERENSI

1. Andres Castelo, Laravel API Tutorial: How to Build and Test a RESTful API, https://www.toptal.com/laravel/restful-laravel-api-tutorial
2. cURL Functions, https://www.php.net/manual/en/function.curl-init.php