Laravel 8: REST API Authentication dengan Sanctum

Laravel 8 Sanctum - Laravel sanctum menyediakan featherweight authentication system untuk Single Page Application (SPA), mobile application dan API berbasis token yang sederhana. Sanctum memungkinkan setiap pengguna aplikasi menghasilkan beberapa token API untuk akun mereka. Token ini dapat diberikan kemampuan atau cangkupan yang menentukan tindakan mana yang diizinkan untuk dilakukan oleh token.

Langkah 1: Install Laravel

```
//via Laravel Installer
composer global require laravel/installer
laravel new laravel8-sanctum

//via Composer
composer create-project laravel/laravel laravel8-sanctum
```

ada langkah yang pertama ini, kita perlu menginstall laravel versi terbaru (saat ini versi 8) yang akan kita coba untuk implementasi membuat REST API authentication menggunakan sanctum. Untuk installasi laravel bisa menggunakan laravel installer atau menggunakan composer seperti contoh di atas.

Silahkan memilih salah satu cara yang ingin digunakan untuk installasi laravel. Dari kedua contoh perintah installasi laravel di atas, akan sama-sama menghasilkan atau generate laravel project dengan nama laravel8-sanctum.

Tunggu hingga proses installasi selesai dan jika sudah selesai, jangan lupa untuk masuk ke direktori project menggunakan perintah **cd laravel8-sanctum**.

angkah 2: Setup Database

```
DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=laravel8_sanctum
```

```
DB_USERNAME=root
DB_PASSWORD=
```

Copy

Selanjutnya, buat database baru untuk percobaan membuat membuat REST API authentication di laravel 8 menggunakan sanctum. Jika kamu menggunakan xampp sebagai local development, silahkan buat database baru di localhost/phpmyadmin. Disini saya beri contoh, saya membuat database baru dengan nama **laravel8_sanctum**. Kemudian jangan lupa juga untuk menyesuaikan **DB_DATABASE** pada file .env seperti pada contoh di atas.

Langkah 3: Install Sanctum

```
composer require laravel/sanctum
```

Copy

Kita dapat menginstall sanctum melalui composer package manager dengan perintah seperti di atas.

```
php artisan vendor:publish --
provider="Laravel\Sanctum\SanctumServiceProvider"
```

Copy

Selanjutnya, kita harus publish sanctum configuration dan file migration menggunakan perintah artisan vendor:publish atau seperti perintah di atas. File sanctum configuration akan ditempatkan di direktori config.

```
php artisan migrate
```

Copy

Kemudian, kita harus menjalankan database migration. Sanctum akan membuat satu table di database untuk menyimpan token API.

Langkah 4: Buat AuthController

```
php artisan make:controller API/AuthController
```

Copy

Di langkah keempat, kita buat file controller baru dengan nama AuthController di dalam folder app/Http/Controllers/API menggunakan perintah seperti di atas. File controller ini nantinya akan kita gunakan untuk membuat logic authentication di laravel 8 menggunakan sanctum.

```php
<?php
namespace App\Http\Controllers\API;

use App\Http\Controllers\Controller;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Hash;
use Auth;
use Validator;
use App\Models\User;

class AuthController extends Controller
{
    public function register(Request $request)
    {
        $validator = Validator::make($request->all(),[
            'name' => 'required|string|max:255',
            'email' => 'required|string|email|max:255|unique:users',
            'password' => 'required|string|min:8'
        ]);

        if($validator->fails()){
            return response()->json($validator->errors());
        }

        $user = User::create([
            'name' => $request->name,
            'email' => $request->email,
            'password' => Hash::make($request->password)
         ]);

        $token = $user->createToken('auth_token')->plainTextToken;

        return response()
            ->json(['data' => $user,'access_token' => $token,
'token_type' => 'Bearer', ]);
```

```
    }

    public function login(Request $request)
    {
        if (!Auth::attempt($request->only('email', 'password')))
        {
            return response()
                ->json(['message' => 'Unauthorized'], 401);
        }

        $user = User::where('email', $request['email'])-
>firstOrFail();

        $token = $user->createToken('auth_token')->plainTextToken;

        return response()
            ->json(['message' => 'Hi '.$user->name.', welcome to
home','access_token' => $token, 'token_type' => 'Bearer', ]);
    }

    // method for user logout and delete token
    public function logout()
    {
        auth()->user()->tokens()->delete();

        return [
            'message' => 'You have successfully logged out and the
token was successfully deleted'
        ];
    }
}
```

Copy

Setelah file AuthController.php berhasil digenerate, sekarang silahkan buka file tersebut dan ubah semua kodenya menjadi seperti di atas. Di file AuthController.php ini, kita membuat method register, login dan logout.

Penjelasan method-method tersebut:

- **Register**. Di method ini, kita menambahkan validasi untuk name, email dan password. Jika data POST request gagal divalidasi, maka akan mengirimkan response error dari validasi tersebut. Tapi, jika POST request berhasil divalidasi, maka data dari POST request akan disimpan di table users dan akan membuat token baru, serta akan mengirimkan response json yang berisikan detail dari data yang telah ditambahkan beserta token yang telah berhasil dibuat.
- **Login**. Di method ini, kita tambahkan logic untuk memeriksa apakah email dan password yang diinput benar-benar cocok dengan salah satu data di table users. Jika data gagal ditemukan di tables users, maka response yang dihasilkan akan berstatus 401 atau Unauthorized. Tapi jika data tersebut berhasil ditemukan, maka akan membuatkan token baru untuk user tersebut yang akan disimpan di table personal_access_tokens.
- **Logout**. Method ini akan menghapus user session dengan menghapus semua token milik user tersebut di table personal_access_token.

Langkah 5: Define Route

```
//API route for register new user
Route::post('/register',
[App\Http\Controllers\API\AuthController::class, 'register']);
//API route for login user
Route::post('/login',
[App\Http\Controllers\API\AuthController::class, 'login']);

//Protecting Routes
Route::group(['middleware' => ['auth:sanctum']], function () {
    Route::get('/profile', function(Request $request) {
        return auth()->user();
    });

    // API route for logout user
    Route::post('/logout',
[App\Http\Controllers\API\AuthController::class, 'logout']);
});
```

Copy

Selanjutnya, buka file routes/api.php dan tambahkan kode route seperti di atas. Disini kita menambahkan route baru yaitu register, login, profile dan logout. Untuk route profile dan logout, kita menggunakan sanctum authenticated guard ('middleware' => ['auth:sanctum']). Artinya, kedua route tersebut hanya dapat diakses oleh pengguna yang telah diautentikasi atau mengakses menggunakan token.

Langkah 6: Testing API

Sekarang waktunya menguji REST API authentication yang telah kita buang menggunakan laravel 8 dan sanctum.

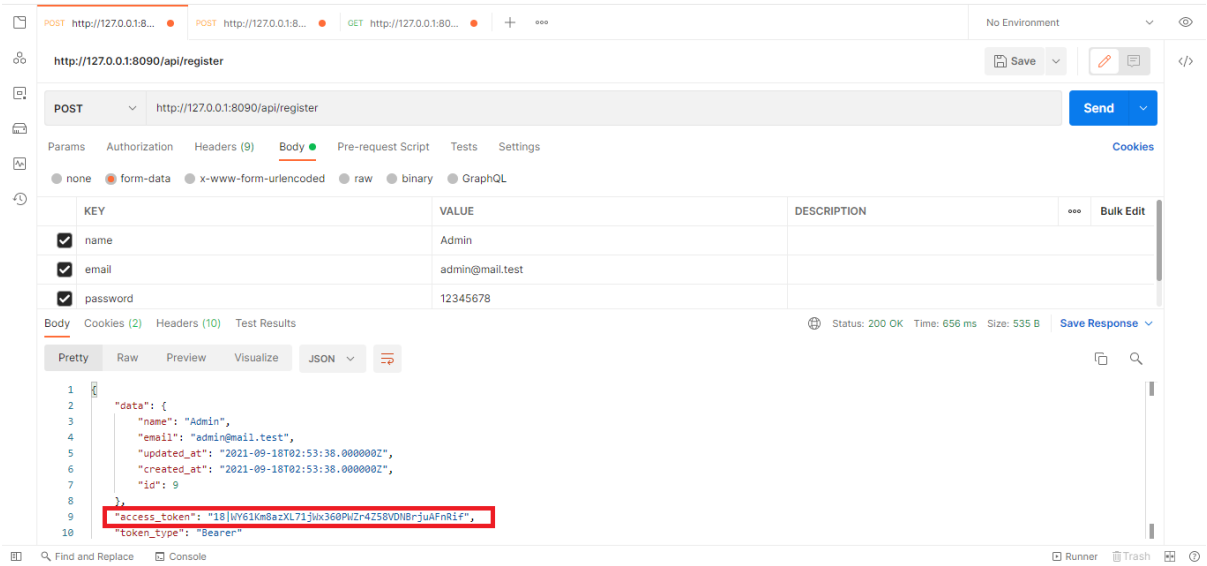
```
php artisan serve
```

Copy

Pertama, pastikan kamu sudah menjalankan laravel project kamu dengan perintah php artisan serve. Dengan perintah tersebut, by default laravel akan memberikan akses laravel project kamu dengan URL 127.0.0.1:8000, tapi kamu juga bisa mengubah portnya sesuai keinginan dengan menambahkan seperti --port=8090.

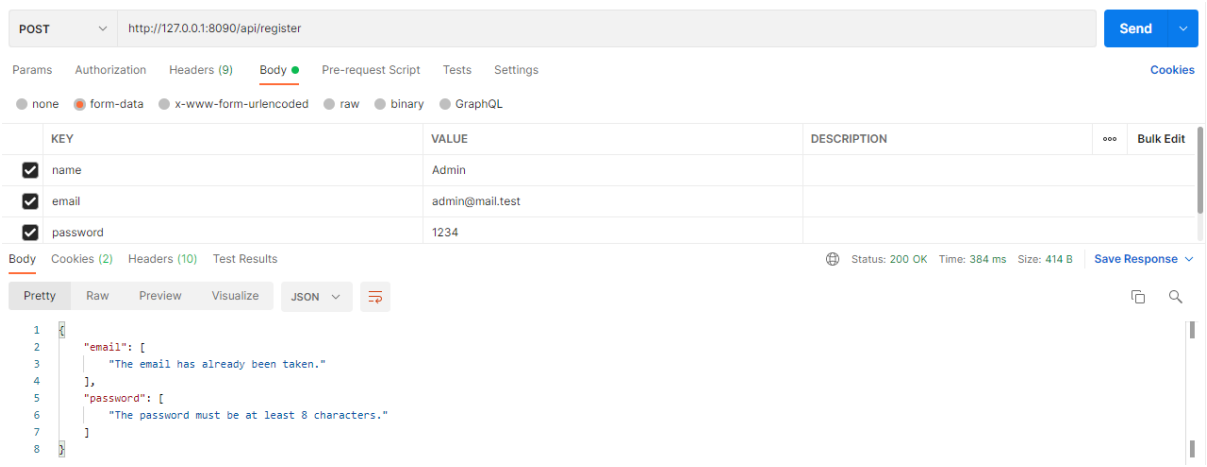
Kita dapat mengakses API yang telah kita buat dengan mengakses URL 127.0.0.1:8000/api. Pada langkah testing API ini, saya akan memberikan contoh-contoh bagaimana cara membuat request API menggunakan Postman seperti di bawah ini.

Register



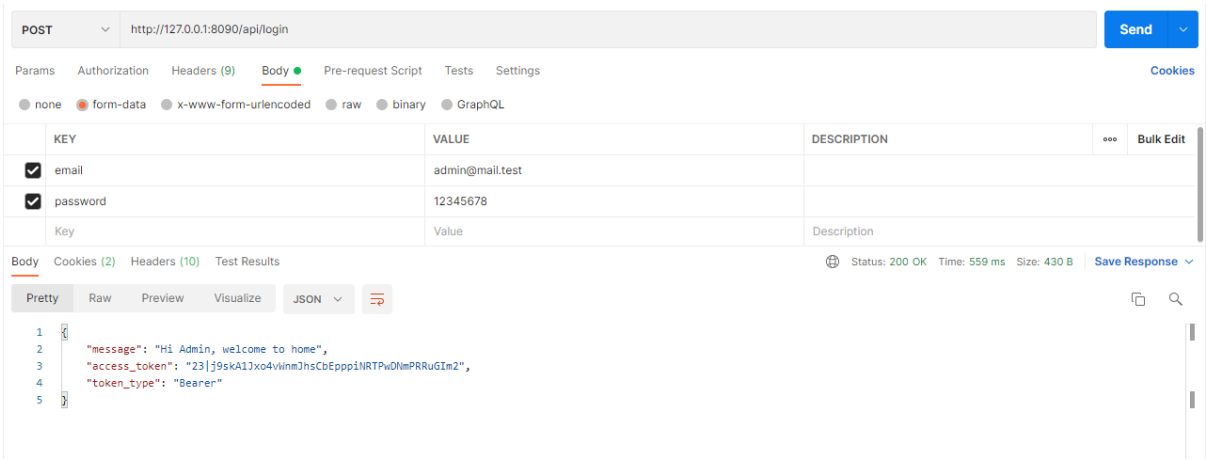
Untuk pengujian awal, kita coba untuk register terlebih dahulu. Untuk register atau menambahkan user baru, buat **POST** request ke 127.0.0.1:8000/api/register. Kemudian pada tab **Body**, silahkan tambahkan key name, email dan password beserta value yang ingin didaftarkan. Karena pada method register di file AuthController tadi kita telah menambahkan validasi untuk password minimal harus 8 karakter, jadi untuk value password pastikan minimal berisi 8 karakter (seperti gambar di atas).

Jika key dan value sudah terisi, lanjutkan dengan klik button **Send**. Maka data yang telah kita inputkan berhasil ditambahkan ke table users dan kita berhasil mendapatkan token untuk kita gunakan.



Jika kita coba register dengan email yang sudah ada di table users dan mengisi password kurang dari 8 karakter, maka data tidak akan ditambahkan ke table users dan akan mengirimkan response seperti gambar di atas.

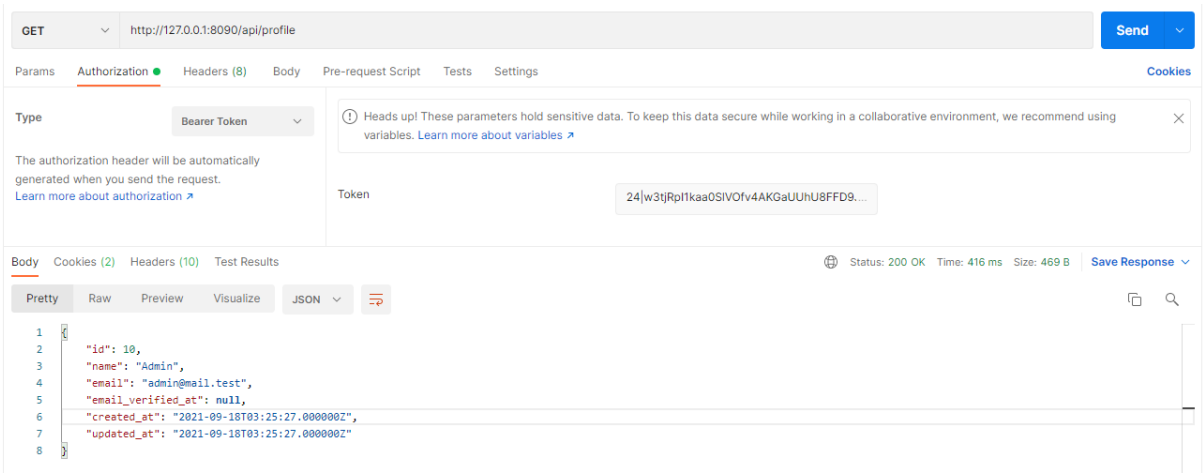
Login



Setelah berhasil register, sekarang kita coba untuk login menggunakan email dan password yang telah di daftarkan atau register. Buat POST request ke 127.0.0.1:8000/api/login, kemudian pada body, tambahkan key email dan password beserta valuenya.

Jika sudah menambahkan key dan value, lanjutnya dengan klik button Send. Maka hasilnya, jika email yang kita inputkan ditemukan di table users dan password yang kita inputkan juga benar milik email tersebut, maka response yang dihasilkan akan seperti gambar di atas dengan status HTTP 200. Tapi jika value yang kita inputkan salah, maka response yang ditampilkan akan unauthorized atau status HTTP 401.

Get Profile Data

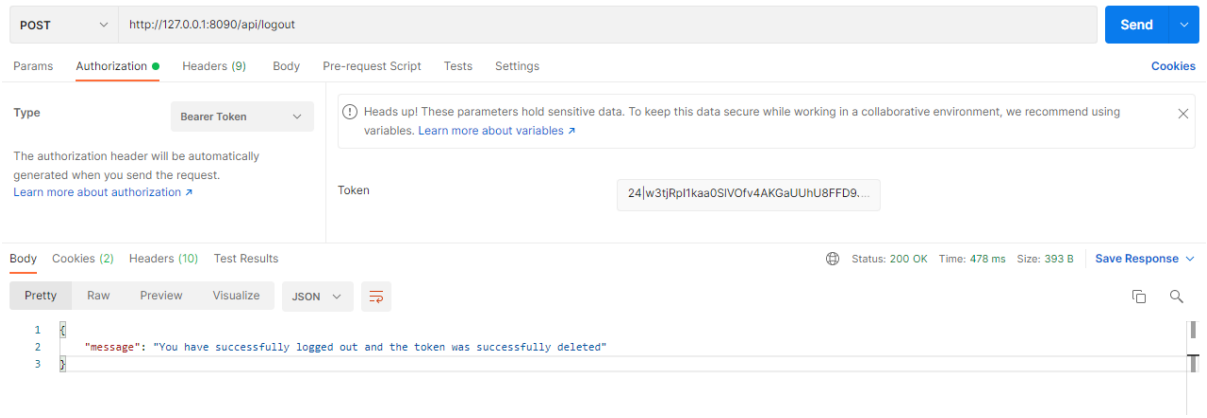


Oke, sekarang kita akan coba mendapatkan detail data user yang sedang kita gunakan untuk login. Karena pada file routes/api.php, kita telah membungkus route api/profile dengan 'middleware' => ['auth:sanctum'], jadi untuk dapat mengakses profile, kita harus login terlebih dahulu.

Untuk menampilkan detail user yang kita gunakan untuk login, buat GET request ke 127.0.0.1:8000/api/profile. Kemudian pada tab Authorization, pilih type Beared Token dan masukkan token yang telah didapatkan saat login tadi.

Kemudian klik button Send. Jika token yang kita masukkan benar, maka response yang ditampilkan akan seperti gambar di atas dengan data detail user.

Logout



Percobaan terakhir, Kita akan coba REST API untuk user logout. Buat POST request ke 127.0.0.1:8000/api/logout, pada tab Authorization pilih type Bearer Token dan masukkan token yang telah didapatkan saat berhasil login tadi.

Klik Send, maka semua token yang dimiliki user tersebut akan dihapus dan response message yang ditampilkan akan seperti gambar di atas.