REST API Authentication With Laravel Sanctum

Overview

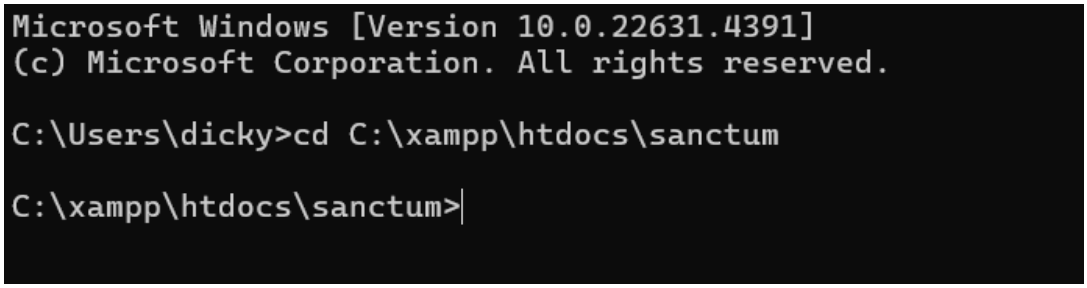
Pada tutorial REST Api Authentication dengan Laravel Sanctum ini kita akan membuat project sederhana yang menyediakan API endpoint untuk proses register, login, logout dan get data. Pada saat user mengirimkan HTTP Request untuk proses register dan login, sistem akan melakukan proses generate api token. API Token ini nanti akan digunakan untuk proses ambil data user dan juga proses logout.

Persiapan

Sebelum kita mulai, ada tools yang harus kita siapkan terlebih dahulu, yaitu Postman. Kita bisa download terlebih dahulu di situs resminnya. Postman ini nanti kita gunakan untuk uji coba Api.

Step 1 - Setup Project

- Buat folder **sanctum** di dalam folder htdocs
- Buka CMD masuk ke direktori **sanctum**



- Install Project Laravel baru menggunakan composer.

```
composer create-project --prefer-dist laravel/laravel laravel-sanctum
```

- Setelah itu masuk ke direktori project

```
cd laravel-sanctum
```

- Buka project menggunakan text editor. Kalau kita pakai visual studio code, kita bisa buka project di visual studio code menggunakan command.

```
code .
```

- Setelah itu kita atur konfigurasi database dengan memodifikasi file .env

```
DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=db_belajar_api
DB_USERNAME=root
DB_PASSWORD=
```

Step 2 - Setup Laravel Sanctum Package

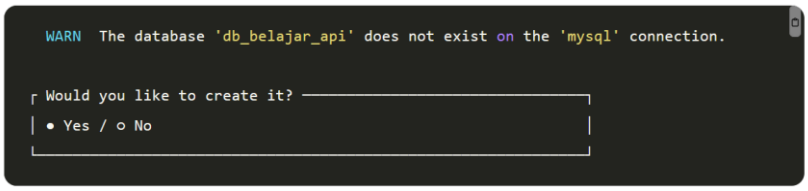
- Versi Laravel pada saat tutorial ini diupdate adalah Laravel versi 11. Pada laravel 11 kita bisa install laravel sanctum dengan run command berikut ini.

```
php artisan install:api
```

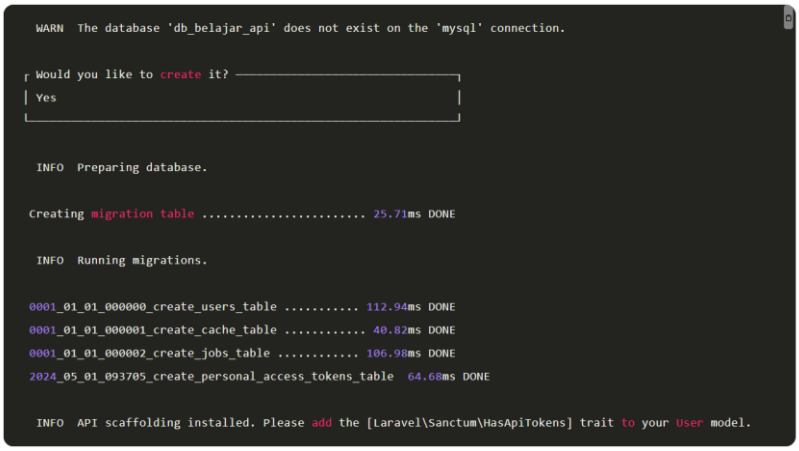
- Tunggu sampai proses install selesai. Diakhir proses install, tampil prompt untuk run database migration.

```
One new database migration has been published. Would you like
    to run all pending database migrations? (yes/no) [yes]:
```

- Ketik yes, lalu tekan enter untuk run database migrations.
- Apabila database db_belajar_api belum kita buat, tampil warning di output terminal.



- Tekan enter untuk melanjutkan.



- Selanjutnya buka file app/Models/User.php
- Lalu kita tambahkan trait Laravel\Sanctum\HasApiTokens

```
<?php

namespace App\Models;

// use Illuminate\Contracts\Auth\MustVerifyEmail;
use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Foundation\Auth\User as Authenticatable;
use Illuminate\Notifications\Notifiable;
use Laravel\Sanctum\HasApiTokens; // tambahkan kode ini

class User extends Authenticatable
{
    use HasFactory, Notifiable, HasApiTokens; // tambahkan
kode ini

    // baris kode lainnya
}
```

Step 3 - Create AuthController

- Pada step 3 ini kita akan membuat controller yang akan menangani proses register, login dan logout melalui api. Buka kembali terminal, lalu run command berikut ini.

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

- Setelah command kita run, kita bisa lihat ada file baru yaitu AuthController.php di direktori app/Http/Controllers/Api. Buka file AuthController.php, lalu kita tambahkan method register(), login(), dan logout()

```
<?php


namespace App\Http\Controllers\Api;


use App\Http\Controllers\Controller;

use App\Models\User;

use Illuminate\Http\Request;

use Illuminate\Support\Facades\Auth;
```

```
use Illuminate\Support\Facades\Hash;
use Illuminate\Support\Facades\Validator;

class AuthController extends Controller
{
    public function register(Request $request)
    {
        $validator = Validator::make($request->all(), [
            'name' => 'required|string|max:255',
            'email' => 'required|string|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' => 'Login success',
            'access_token' => $token,
            'token_type' => 'Bearer'
        ]);
    }

    public function logout()
    {
        Auth::user()->tokens()->delete();
        return response()->json([
            'message' => 'logout success'
        ]);
    }
}
```

- Save kembali file AuthController.php.

Step 4 - Definisikan Route

Langkah berikutnya adalah mendefinisikan route. Berbeda dengan tutorial biasanya, karena tutorial ini tentang api, jadi kita definisikan route di file routes/api.php. Buka file routes/api.php, lalu kita tambahkan route untuk register, login dan juga logout.

```
<?php


use Illuminate\Http\Request;

use Illuminate\Support\Facades\Route;


/*

|--------------------------------------------------------------------------

| API Routes

|--------------------------------------------------------------------------

|

| Here is where you can register API routes for your application. These

| routes are loaded by the RouteServiceProvider within a group which

| is assigned the "api" middleware group. Enjoy building your API!

|

*/


Route::post('/register', [\App\Http\Controllers\Api\AuthController::class, 'register']);

Route::post('/login', [\App\Http\Controllers\Api\AuthController::class, 'login']);
```

```
Route::middleware('auth:sanctum')->get('/user',        function
(Request $request) {
    return $request->user();
});


Route::middleware('auth:sanctum')->group(function () {
    Route::post('/logout',
[\App\Http\Controllers\Api\AuthController::class, 'logout']);
});
```

Step 5 - Uji Coba

- Untuk menguji coba, pertama kita run terlebih dahulu project kita. Buka terminal, lalu run command berikut ini.

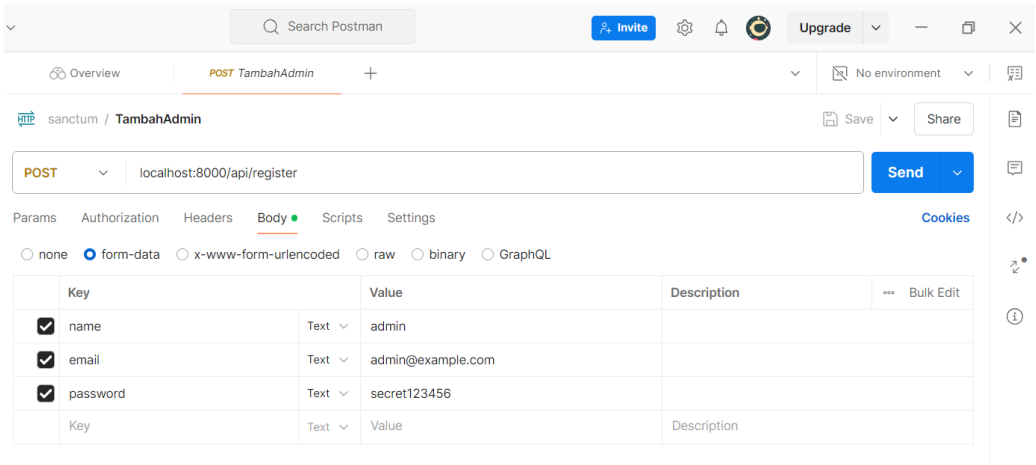
```
php artisan serve
```

- Setelah itu buka postman yang sebelumnya sudah kita siapkan.

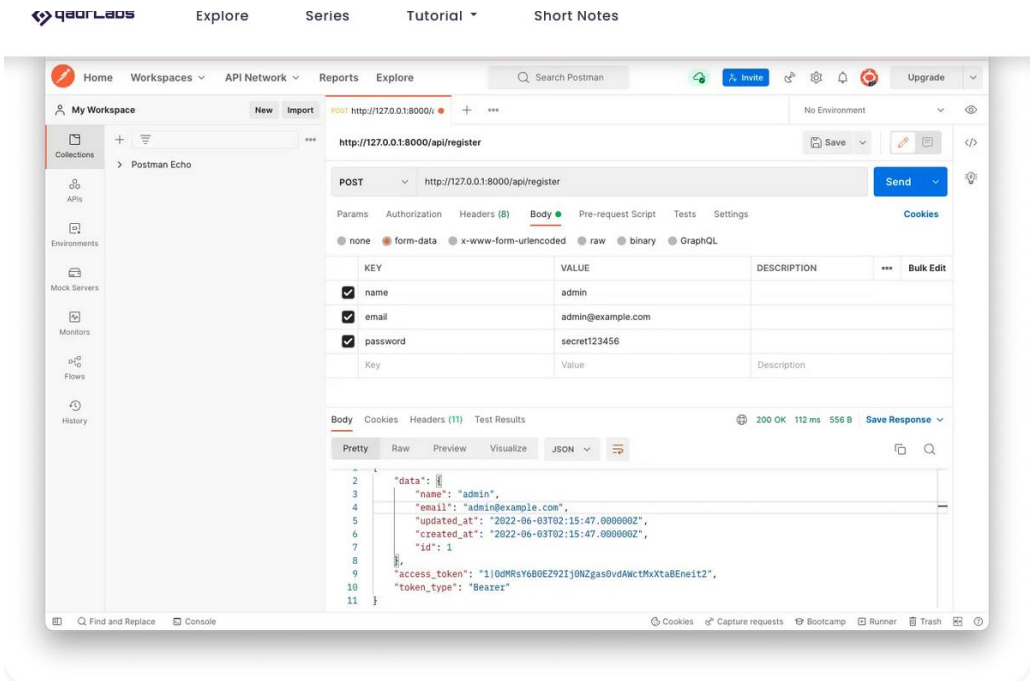
Uji Coba Register

Untuk menguji fitur register, kita tambahkan POST request di dalam postman. Kita atur terlebih dahulu, method request, url dan data yang akan kita kirim.

1. Pada menu HTTP request method kita pilih POST,
2. Pada input url kita arahkan urlnya ke http://127.0.0.1:8000/api/register,
3. Pada tab Body, kita pilih radio button form-data, lalu kita coba masukan sample data name, email dan password, setelah itu kita klik tombol Send untuk mengirim POST request.



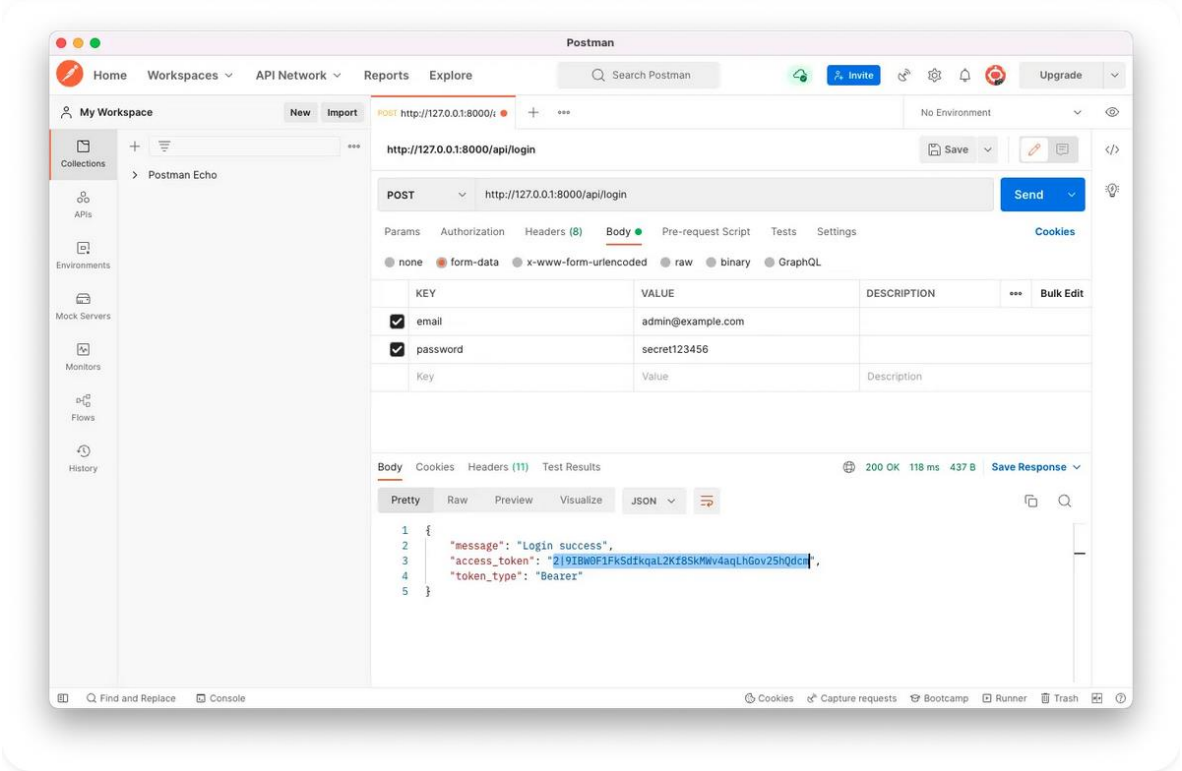
Bisa kita lihat pada gambar di atas, setelah kita kirim POST request terdapat response yang berisi data user, token dan juga token type di panel bawah.



Uji Coba Login

Selanjutnya kita coba proses login, buka kembali postman, lalu kita sesuaikan seperti berikut ini.

1. Pada menu HTTP request method kita pilih POST,
2. Pada input url kita arahkan urlnya ke http://127.0.0.1:8000/api/login,
3. Pada tab Body, kita pilih radio button form-data, lalu kita coba masukan sample data email dan password sesuai dengan yang sudah kita masukan pada saat uji coba register, setelah itu kita klik tombol Send untuk mengirim POST request. Kurang lebih outputnya seperti di gambar di bawah ini.



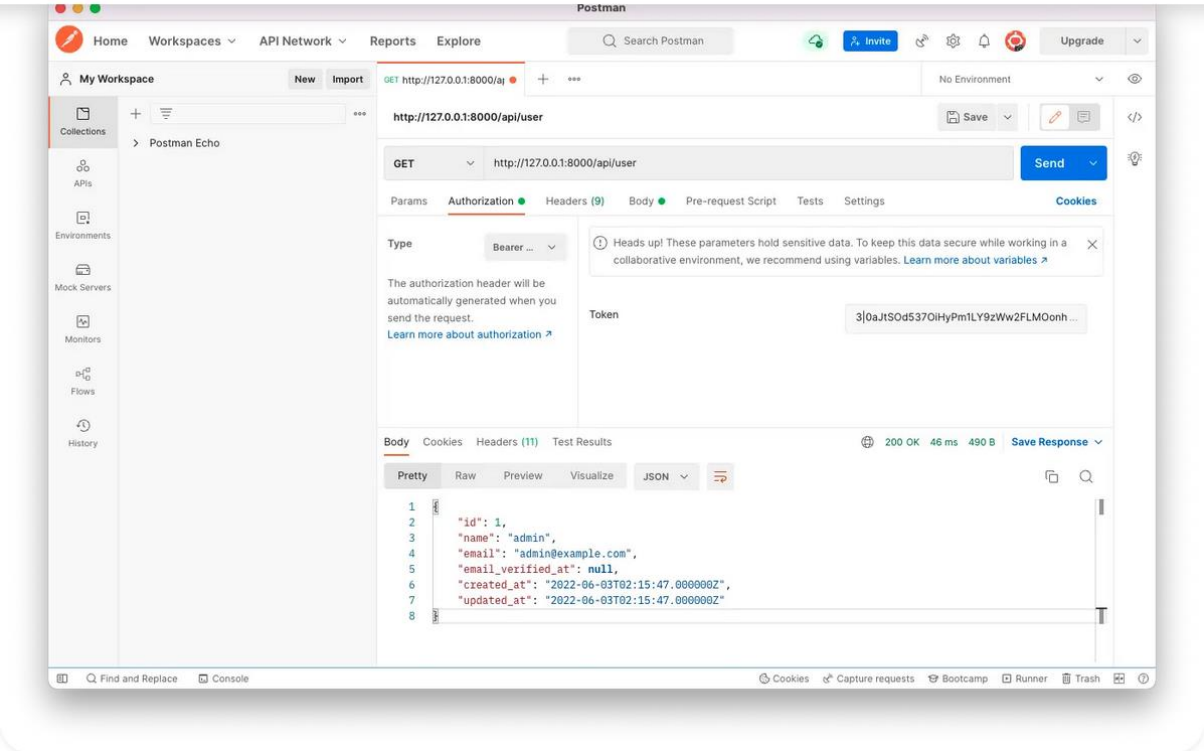
Bisa kita lihat di dalam response terdapat access_token. Kita catat dulu token tersebut, bisa kita copy dulu ke text editor. Token ini nanti akan kita coba gunakan untuk uji coba ambil data user dan logout.

Uji Coba Get Data User

Kalau kita cek kembali di file routes/api.php, route untuk ambil data user itu terdapat middleware auth:sanctum. Jadi kita perlu memasukan token yang sebelumnya kita dapatkan pada saat uji coba login. Sekarang kita buka kembali postman, lalu kita sesuaikan pengaturannya.

1. Pada menu HTTP request method kita pilih GET,
2. Pada input url kita arahkan urlnya ke http://127.0.0.1:8000/api/user,
3. Sekarang buka tab Authorization, pilih type Bearer Token, dan masukan token yang sebelumnya kita dapat pada uji coba login.

Setelah itu kita kirim GET Request dengan menekan tombol Send. Kurang lebih tampilan outputnya seperti gambar di bawah ini.



Uji Coba Logout

Sama seperti pada saat kita ambil data user, asumsinya kita sedang dalam keadaan login dan di route untuk logout terdapat auth:sanctum. Jadi untuk uji coba logout, kita perlu memasukan token juga. Sekarang kita buka kembali postman, lalu kita sesuaikan pengaturannya.

1. Pada menu HTTP request method kita pilih POST,
2. Pada input url kita arahkan urlnya ke http://127.0.0.1:8000/api/logout,
3. Sekarang buka tab Authorization, pilih type Bearer Token, dan masukan token yang sebelumnya kita dapat pada uji coba login.

Setelah itu kita kirim POST request dengan menekan tombol Send. Kurang lebih outputnya seperti pada gambar di bawah ini.