Lumen Step **php artisan serve --host=127.0.0.1** OR **php -S localhost:8000 -t public**

**Cretae Project Lumen**

*Php artisan untuk membuat model dan controller tidak bisa di lumen.utk melihat perintah php artisan:*

* **php artisan list**
* **composer create-project --prefer-dist laravel/lumen MyApp**
* **cd Myapp**
* **php -S localhost:8000 -t public**
* **buka localhost:8000**

**Membuat Setting DB**

* *buka file .env dan setting sesuai database:*

DB\_CONNECTION**=**mysql

DB\_HOST**=**127.0.0.1

DB\_PORT**=**3306

DB\_DATABASE**=**my\_db\_laravel

DB\_USERNAME**=**root

DB\_PASSWORD**=**

* *Silahkan buka file app.php di dalam folder* ***bootstrap/app.php*** *dan uncomment kode dibawah ini :*

$app**->**withFacades();

$app**->**withEloquent();

* **Membuat Migration**

php artisan make:migration create\_users\_table

*maka akan dibuatkan data file migration di folder* ***database\migrations\namafilemigration.php***

* **buka file tersebut dan isi dengan:**

**use** Illuminate\Support\Facades\Schema;

**use** Illuminate\Database\Schema\Blueprint;

**use** Illuminate\Database\Migrations\Migration;

class CreateUsersTable *extends* *Migration*

{

*public* function up()

    {

        Schema**::**create('users', function (Blueprint **$table**) {

            $table**->**increments('id');

            $table**->**string('name');

            $table**->**string('email');

            $table**->**string('phone');

            $table**->**string('latest\_article\_published');

            $table**->**timestamps();

        });

    }

*public* function down()

    {

        Schema**::**dropIfExists('users');

    }

}

*Dan jalankan migrate utk eksekusi table:*

php artisan migrate

* **Buat Model**

*Buat model di folder di folder =>* ***app/User.php***

**namespace** App;

**use** Illuminate\Database\Eloquent\Model;

class Author *extends* *Model*

{

*protected* $fillable **=** [

        'name', 'email', 'phone', 'latest\_article\_published'

    ];

*/\*\**

*\* The attributes excluded from the model's JSON form.*

*\**

*\* @var array*

*\*/*

*protected* $hidden **=** [];

}

* **Edit Routes**

*Edit routes di =>* ***routes/web.php***

**<?**php

$router**->**get('/', function () **use** ($router) {

**return** $router**->***app***->**version();

});

$router**->**get('/', function () **use** ($router) {

**return** $router**->***app***->**version();

});

$router**->**group(['prefix' **=>** 'api'], function () **use** ($router) {

  $router**->**get('users',  ['uses' **=>** 'UserController@showAllUsers']);

  $router**->**get('users/{id}', ['uses' **=>** 'UserController@showOneUser']);

  $router**->**post('users', ['uses' **=>** 'UserController@create']);

  $router**->**delete('users/{id}', ['uses' **=>** 'UserController@delete']);

  $router**->**put('users/{id}', ['uses' **=>** 'UserController@update']);

});

* **Membuat Controller users**

*Buat controller user dgn nama dan di folder* => ***app/Http/Controllers/UserController.php dan isi:***

**namespace** App\Http\Controllers;

**use** App\User;

**use** Illuminate\Http\Request;

**use** Illuminate\Support\Facades\Validator;

class UsersController *extends* *Controller*

{

*public* function index()

    {

        $users **=** User**::**all();

**return** response()**->**json([

            'success' **=>** true,

            'message' **=>**'List Semua User',

            'data'    **=>** $users

        ], 200);

    }

*public* function store(Request **$request**)

{

    $validator **=** Validator**::**make($request**->**all(), [

        'username'   **=>** 'required',

        'password' **=>** 'required',

    ]);

**if** ($validator**->**fails()) {

**return** response()**->**json([

            'success' **=>** false,

            'message' **=>** 'Username and password Wajib Diisi!',

            'data'   **=>** $validator**->**errors()

        ],401);

    } **else** {

        $user **=** User**::**create([

            'username'     **=>** $request**->**input('username'),

            'password'   **=>** $request**->**input('password'),

            'phone'   **=>** $request**->**input('phone'),

            'address'   **=>** $request**->**input('address'),

        ]);

**if** ($user) {

**return** response()**->**json([

                'success' **=>** true,

                'message' **=>** 'User Berhasil Disimpan!',

                'data' **=>** $user

            ], 201);

        } **else** {

**return** response()**->**json([

                'success' **=>** false,

                'message' **=>** 'User Gagal Disimpan!',

            ], 400);

        }

    }

}

*public* function show(**$id**)

{

   $user **=** User**::**find($id);

**if** ($user) {

**return** response()**->**json([

           'success'   **=>** true,

           'message'   **=>** 'Detail User!',

           'data'      **=>** $user

       ], 200);

   } **else** {

**return** response()**->**json([

           'success' **=>** false,

           'message' **=>** 'User Tidak Ditemukan!',

       ], 404);

   }

}

*public* function update(Request **$request**, **$id**)

{

    $validator **=** Validator**::**make($request**->**all(), [

        'username'   **=>** 'required',

        'password' **=>** 'required',

    ]);

**if** ($validator**->**fails()) {

**return** response()**->**json([

            'success' **=>** false,

            'message' **=>** 'Semua Kolom Wajib Diisi!',

            'data'   **=>** $validator**->**errors()

        ],401);

    } **else** {

        $user **=** User**::**whereId($id)**->**update([

            'username'     **=>** $request**->**input('username'),

            'password'   **=>** $request**->**input('password'),

            'phone'   **=>** $request**->**input('phone'),

            'address'   **=>** $request**->**input('address'),

        ]);

**if** ($user) {

**return** response()**->**json([

                'success' **=>** true,

                'message' **=>** 'User Berhasil Diupdate!',

                'data' **=>** $user

            ], 201);

        } **else** {

**return** response()**->**json([

                'success' **=>** false,

                'message' **=>** 'User Gagal Diupdate!',

            ], 400);

        }

    }

}

*public* function destroy(**$id**)

{

    $user **=** User**::**whereId($id)**->**first();

    $user**->**delete();

**if** ($user) {

**return** response()**->**json([

            'success' **=>** true,

            'message' **=>** 'User Berhasil Dihapus!',

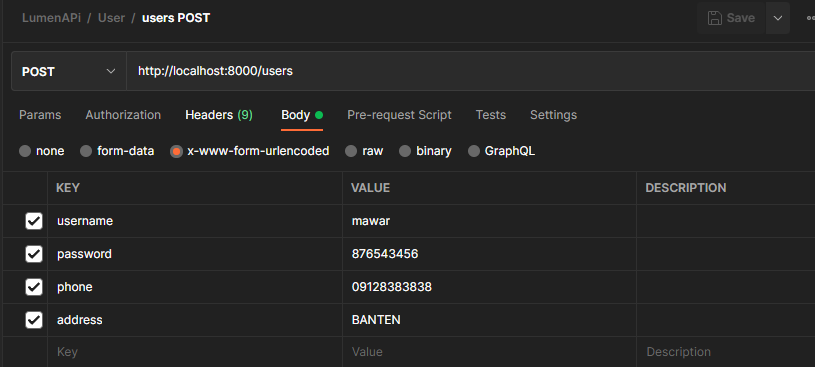
        ], 200);

    }

}

}

* **Akses Routes dengan Postman**

Akses restful dengan postman dgn methode **GET,POST,PUT,DELETE . example**

**MEMBUAT RESTFULL EN*ABLE CORS***

*Agar API bisa diakses diluar aplikasi maka diaktifkan CORS*

* Membuat Request Provider

*Buat file di folder* *app/Providers/CatchAllOptionsRequestsProvider.php*

**<?**php

**namespace** App\Providers;

**use** Illuminate\Support\ServiceProvider;

class CatchAllOptionsRequestsProvider *extends* *ServiceProvider* {

*public* function register()

    {

        $request **=** app('request');

**if** ($request**->**isMethod('OPTIONS'))

        {

            app()**->**options($request**->**path(), function() { **return** response('', 200); });

        }

    }

}

* **Membuaut middleware cors**

*Buat file di =>* ***app/Http/middleware/CorsMiddleware.php***

**<?**php **namespace** App\Http\Middleware;

**use** Closure;

class CorsMiddleware

{

*public* function handle(**$request**, Closure **$next**)

    {

        $headers **=** [

            'Access-Control-Allow-Origin'      **=>** '\*',

            'Access-Control-Allow-Methods'     **=>** 'POST, GET, OPTIONS, PUT, DELETE',

            'Access-Control-Allow-Credentials' **=>** 'true',

            'Access-Control-Max-Age'           **=>** '86400',

            'Access-Control-Allow-Headers'     **=>** 'Content-Type, Authorization, X-Requested-With'

        ];

**if** ($request**->**isMethod('OPTIONS'))

        {

**return** response()**->**json('{"method":"OPTIONS"}', 200, $headers);

        }

        $response **=** $next($request);

**foreach**($headers **as** $key **=>** $value)

        {

            $response**->**header($key, $value);

        }

**return** $response;

    }

}

* **Register Provider dan middleware**

*Buka file =>* ***bootsrap/app.php*** *dan tambahkan di bawah variable $app seperti berikut:*

$app**->**withFacades();

$app**->**withEloquent();

*// Register Middleware*

 $app**->**middleware([

     App\Http\Middleware\CorsMiddleware**::class**

 ]);

*// Register Service Providers*

 $app**->**register(App\Providers\CatchAllOptionsRequestsProvider**::class**);

**Tools-tools**

**Agar php artisan generate key bisa install:**

composer **require** flipbox**/**lumen**-**generator

*Inside your bootstrap/app.php file, add:*

$app->register(Flipbox\LumenGenerator\LumenGeneratorServiceProvider::class);

php artisan key:generate

**Middleware custom**

$app**->**middleware([

   App\Http\Middleware\UmurMiddleware**::class**

]);

**Middle ware global**

$app**->**middleware([

    App\Http\Middleware\UmurMiddleware**::class**

 ]);

**Services JWT API**

* *Create file migration*

php artisan make:migration create\_users\_table --create=users

* migration file edit up function

public function up()

{

Schema::create('users', function (Blueprint $table) {

$table->bigIncrements('id');

$table->string('name');

$table->string('email')->unique()->notNullable();

$table->string('password');

$table->timestamps();

});

}

* *Run migration:*

php artisan migrate

* *Create file app/Http/Controllers/AuthController.php*

<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;

use App\User;

class AuthController extends Controller

{

public function register(Request $request)

{

//validate incoming request

$this->validate($request, [

'name' => 'required|string',

'email' => 'required|email|unique:users',

'password' => 'required|confirmed',

]);

try {

$user = new User;

$user->name = $request->input('name');

$user->email = $request->input('email');

$plainPassword = $request->input('password');

$user->password = app('hash')->make($plainPassword);

$user->save();

//return successful response

return response()->json(['user' => $user, 'message' => 'CREATED'], 201);

} catch (\Exception $e) {

//return error message

return response()->json(['message' => 'User Registration Failed!'], 409);

}

}

}

* *Instal jwt:*

composer require tymon/jwt-auth

* *create file config/auth.php*

//config.auth.php

<?php

return [

'defaults' => [

'guard' => 'api',

'passwords' => 'users',

],

'guards' => [

'api' => [

'driver' => 'jwt',

'provider' => 'users',

],

],

'providers' => [

'users' => [

'driver' => 'eloquent',

'model' => \App\User::class

]

]

];

* Create *User model(app/User.php)*

<?php

namespace App;

use Illuminate\Auth\Authenticatable;

use Laravel\Lumen\Auth\Authorizable;

use Illuminate\Database\Eloquent\Model;

use Illuminate\Contracts\Auth\Authenticatable as AuthenticatableContract;

use Illuminate\Contracts\Auth\Access\Authorizable as AuthorizableContract;

use Tymon\JWTAuth\Contracts\JWTSubject;

class User extends Model implements AuthenticatableContract, AuthorizableContract, JWTSubject

{

use Authenticatable, Authorizable;

protected $fillable = [

'name', 'email'

];

protected $hidden = [

'password',

];

public function getJWTIdentifier()

{

return $this->getKey();

}

public function getJWTCustomClaims()

{

return [];

}

}

* *Make some changes to bootstrap/app.php*

//After

$app->routeMiddleware([

'auth' => App\Http\Middleware\Authenticate::class,

]);

//before

// $app->register(App\Providers\AppServiceProvider::class);

// $app->register(App\Providers\AuthServiceProvider::class);

// $app->register(App\Providers\EventServiceProvider::class);

//After

// $app->register(App\Providers\AppServiceProvider::class);

$app->register(App\Providers\AuthServiceProvider::class);

// $app->register(App\Providers\EventServiceProvider::class);

$app->register(Tymon\JWTAuth\Providers\LumenServiceProvider::class);

* *Add a global respondWithToken method to Controller class in app/Http/Controllers/Controller.php*

...

//import auth facades

use Illuminate\Support\Facades\Auth;

//Add this method to the Controller class

protected function respondWithToken($token)

{

return response()->json([

'token' => $token,

'token\_type' => 'bearer',

'expires\_in' => Auth::factory()->getTTL() \* 60

], 200);

}

* *Add a login method to your AuthController class in app/Http/Controllers/AuthController.php*

//import auth facades

use Illuminate\Support\Facades\Auth;

public function login(Request $request)

{

//validate incoming request

$this->validate($request, [

'email' => 'required|string',

'password' => 'required|string',

]);

$credentials = $request->only(['email', 'password']);

if (! $token = Auth::attempt($credentials)) {

return response()->json(['message' => 'Unauthorized'], 401);

}

return $this->respondWithToken($token);

}

* *Create a file app/Http/Controllers/UserController.php*

<?php

namespace App\Http\Controllers;

use Illuminate\Support\Facades\Auth;

use App\User;

class UserController extends Controller

{

public function \_\_construct()

{

$this->middleware('auth');

}

public function profile()

{

return response()->json(['user' => Auth::user()], 200);

}

public function allUsers()

{

return response()->json(['users' => User::all()], 200);

}

public function singleUser($id)

{

try {

$user = User::findOrFail($id);

return response()->json(['user' => $user], 200);

} catch (\Exception $e) {

return response()->json(['message' => 'user not found!'], 404);

}

}

}

* *Add a couple of routes to routes/web.php*

$router->group(['prefix' => 'api'], function () use ($router) {

// Matches "/api/register

$router->post('register', 'AuthController@register');

// Matches "/api/login

$router->post('login', 'AuthController@login');

// Matches "/api/profile

$router->get('profile', 'UserController@profile');

// Matches "/api/users/1

//get one user by id

$router->get('users/{id}', 'UserController@singleUser');

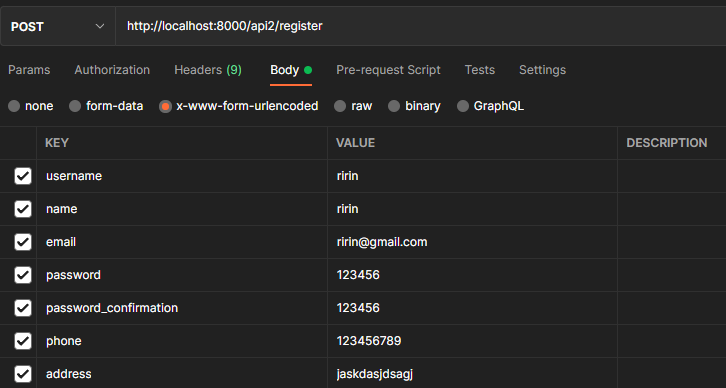
// Matches "/api/users

$router->get('users', 'UserController@allUsers');

});

**Access postman:**

**Register Postman**



**POST http://localhost:8000/api2/login**

email:raflesnfgln**@**gmail**.**com

password:12345678

**GET http://localhost:8000/api2/users => with bearer token**

INFO ! :

For access localhost api lumen from frontend you should:

php artisan serve **--**host**=**127.0**.**0.1

php artisan serve --host=0.0.0.0 --port=5000

**DOCS**

* Publish jwt *php artisan vendor:publish --provider="Tymon\JWTAuth\Providers\LaravelServiceProvider"*
* **https://jwt-auth.readthedocs.io/en/docs/quick-start/#update-your-user-model**
* **https://jwt-auth.readthedocs.io/en/develop/quick-start/**
* *https://auth0.com/blog/developing-restful-apis-with-lumen/*
* *https://medium.com/@ridhofebriansa/cara-menggunakan-jwt-json-web-token-pada-lumen-5-8-dan-postgresql-1018120fa735*
* *https://dev.to/ndiecodes/build-a-jwt-authenticated-api-with-lumen-2afm*
* *https://github.com/ndiecodes/lumen-auth-example*
* *https://www.avyatech.com/rest-api-with-laravel-8-using-jwt-token/*
* *https://santrikoding.com/search?q=jwt*