

AUTENTIFIKASI DAN OTORISASI MENGUNAKAN JWT

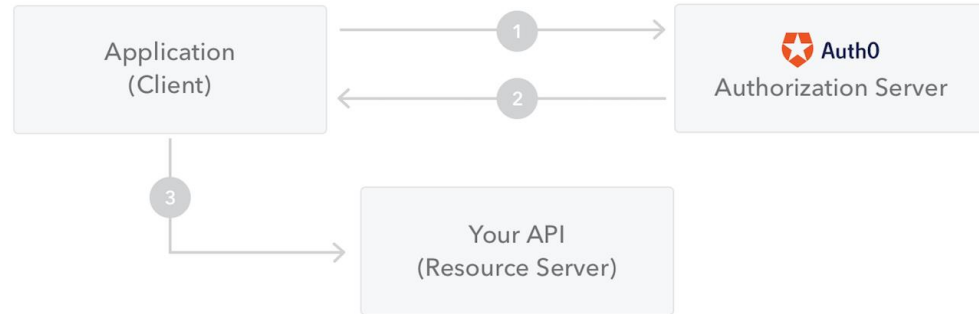
Muhammad Jaya Siraj

Apa itu JWT ?

JSON Web Token (JWT) ini adalah sebuah token berbentuk string panjang yang sangat random yang gunanya sendiri untuk melakukan sistem Autorisasi dan Pertukaran Informasi. Umumnya untuk melakukan login tidak seperti pada aplikasi website biasa dimana kita menggunakan session untuk mengingat siapa yang sedang Login

Bagaimana Cara Kerja JWT ?

Ketika users berhasil melakukan Login maka server akan memberikan sebuah Token. Nanti Token tersebut akan disimpan oleh users pada Local Storage atau Cookies Browser dan bila users ingin mengakses halaman halaman tertentu maka harus menyertakan token tersebut. Untuk itu users akan mengirim balik token yang tersebut sebagai bukti bila user ini sudah melakukan login.



Struktur JWT

- Header
- Payload
- Signature

xxxxx.yyyyyy.zzzzz

Header

Header biasanya terdiri dari dua bagian: jenis token, yaitu JWT, dan algoritma singnature yang digunakan, seperti HMAC SHA256 atau RSA.

```
{  
  "alg": "HS256",  
  "typ": "JWT"  
}
```

Payload

Payload berisi Claims atau statement entitas (biasanya data pengguna) dan beberapa data tambahan lainnya

```
{  
  "sub": "1234567890",  
  "name": "John Doe",  
  "admin": true  
}
```

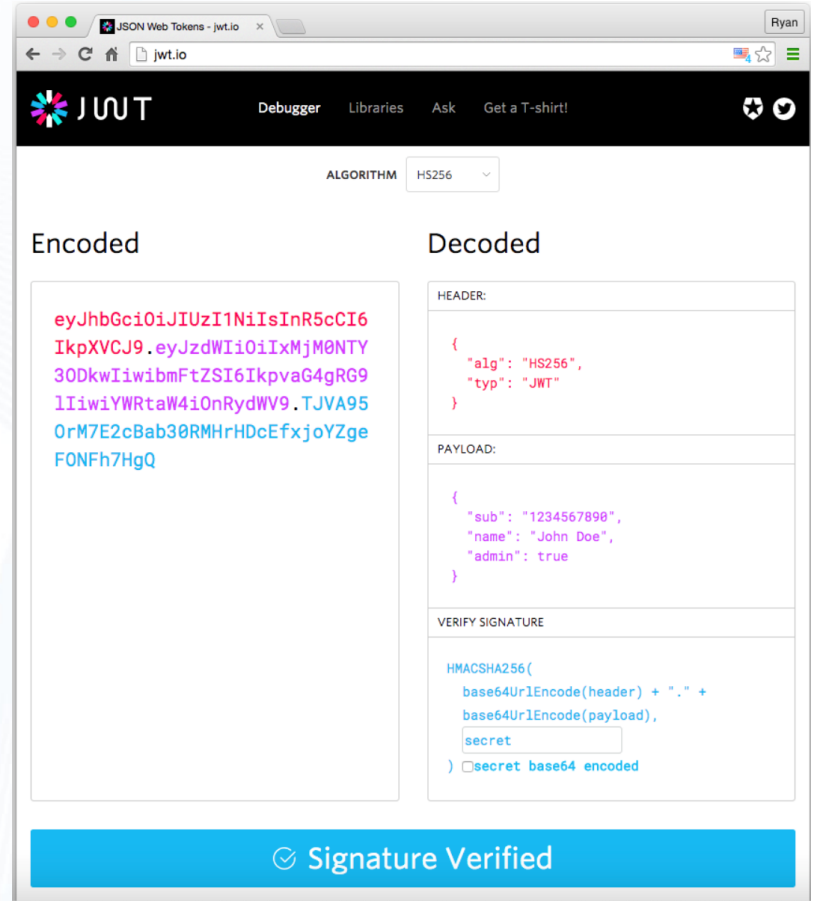
Singature

Signature terdiri dari Header dan payload yang telah di decode dengan algoritma base64, dan dienkripsi dengan menggunakan algoritma yang ditentukan pada header dengan secret key yang kita berikan

```
HMACSHA256(  
    base64UrlEncode(header) + "." +  
    base64UrlEncode(payload),  
    secret)
```

Experiment : jwt.io

Experiment : jwt.io



Instalasi JWT pada Laravel

Install library JWT

```
composer require tymon/jwt-auth
```

Tambahkan providers pada

[config/app.php](#)

```
'providers' => [  
    ....  
    Tymon\JWTAuth\Providers\LaravelServiceProvider::class,  
],
```

Instalasi JWT pada Laravel

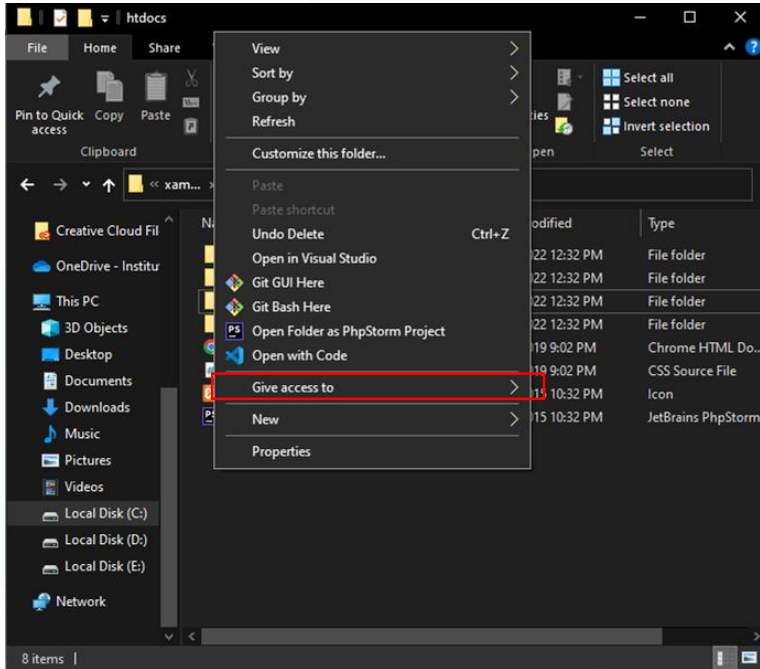
Publish JWT Config

```
php artisan vendor:publish --provider="Tymon\JWTAuth\Providers\LaravelServiceProvider"
```

Generate Secret Token pada .env

```
php artisan jwt:secret
```

Praktek **Kuy!**



Membuka direktori untuk instalasi Laravel pada local host. Misal pada direktori **C:\xampp\htdocs**. Klik kanan lalu pilih **Git Bash Here** atau buka menggunakan CMD / terminal.

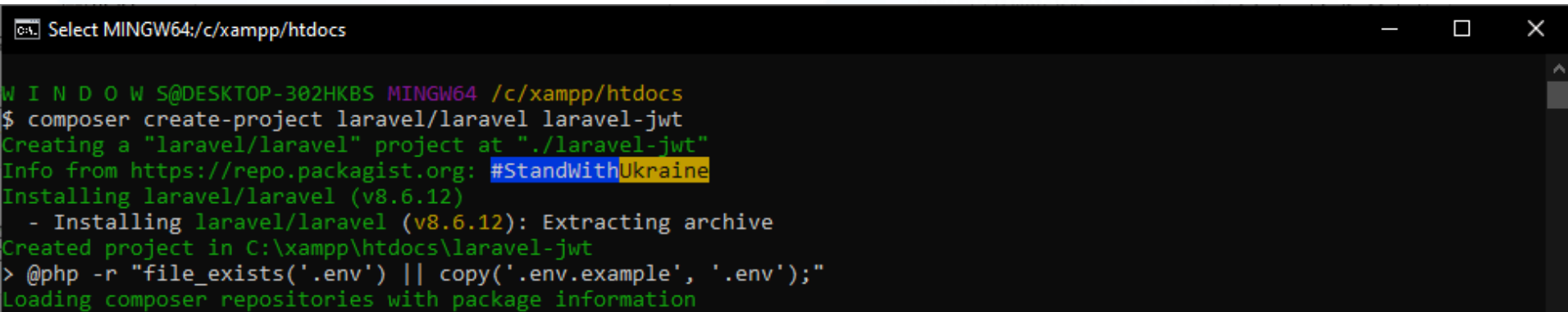
Pastikan perangkat terkoneksi dengan internet. Menginstal Laravel dengan memasukkan perintah

composer create-project Laravel/Laravel *nama-project*

Sebagai contoh

composer create-project Laravel/Laravel laravel-jwt

Tunggu hingga proses instalasi selesai

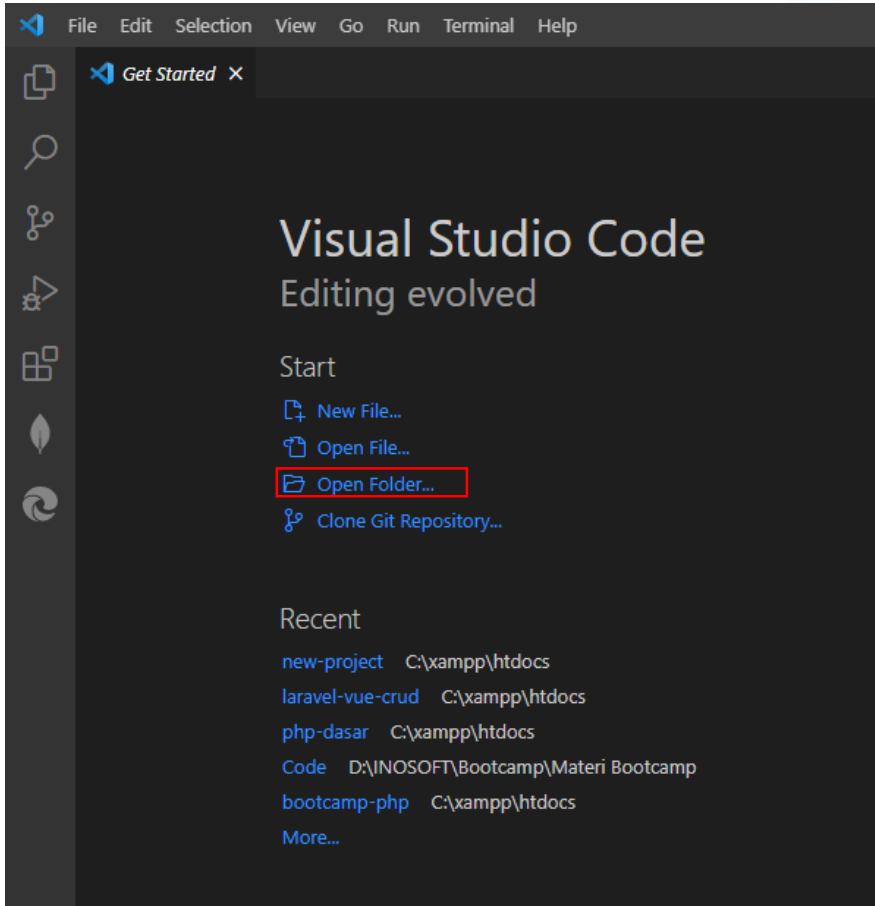


```

C:\> Select MINGW64:/c:/xampp/htdocs

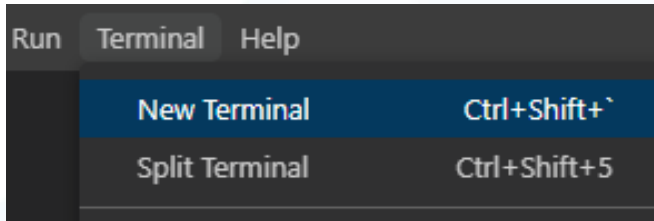
W I N D O W S@DESKTOP-302HKBS MINGW64 /c:/xampp/htdocs
$ composer create-project laravel/laravel laravel-jwt
Creating a "laravel/laravel" project at "./laravel-jwt"
Info from https://repo.packagist.org: #StandWithUkraine
Installing laravel/laravel (v8.6.12)
- Installing laravel/laravel (v8.6.12): Extracting archive
Created project in C:\xampp\htdocs\laravel-jwt
> @php -r "file_exists('.env') || copy('.env.example', '.env');"
Loading composer repositories with package information

```



Membuka folder project yang telah dibuat tadi menggunakan VS Code atau text editor lain.

Pada VS Code pilih menu Open Folder kemudian sistem akan membuka jendela baru lalu pilih folder yang berisikan proyek Laravel yang telah dibuat sebelumnya **(laravel-jwt)**.



Klik menu Terminal lalu pilih sub-menu New Terminal.

Instal library JWT dengan memasukkan perintah

```
composer require tymon/jwt-auth
```

Jika terjadi **error** saat instalasi masukkan perintah

```
composer require tymon/jwt-auth --ignore-platform-reqs
```

```
PS C:\xampp\htdocs\laravel-jwt> composer require tymon/jwt-auth
Info from https://repo.packagist.org: #StandWithUkraine
Using version ^1.0 for tymon/jwt-auth
./composer.json has been updated
Running composer update tymon/jwt-auth
Loading composer repositories with package information
Updating dependencies
Lock file operations: 4 installs, 0 updates, 0 removals
- Locking lcobucci/jwt (3.3.3)
- Locking namshi/jose (7.2.3)
- Locking symfony/polyfill-php56 (v1.20.0)
- Locking tymon/jwt-auth (1.0.2)
Writing lock file
Installing dependencies from lock file (including require-dev)
Package operations: 4 installs, 0 updates, 0 removals
```

Menambahkan providers pada file **config/app.php** dengan sintaks
Tymon\JWTAuth\Providers\LaravelServiceProvider::class

```
172 App\Providers\AppServiceProvider::class,  
173 App\Providers\AuthServiceProvider::class,  
174 // App\Providers\BroadcastServiceProvider::class,  
175 App\Providers\EventServiceProvider::class,  
176 App\Providers\RouteServiceProvider::class,  
177 Tymon\JWTAuth\Providers\LaravelServiceProvider::class,  
178
```

Publish JWT Config, dengan memasukkan perintah
php artisan vendor:publish --
provider="Tymon\JWTAuth\Providers\LaravelServiceProvider"

```
PS C:\xampp\htdocs\laravel-jwt> php artisan vendor:publish --provider="Tymon\JWTAuth\Providers\LaravelServiceProvider"  
Copied File [\vendor\tymon\jwt-auth\config\config.php] To [\config\jwt.php]  
Publishing complete.
```


Membuat/generate secret key dengan memasukkan perintah
php artisan jwt:secret

```
PS C:\xampp\htdocs\laravel-jwt> php artisan jwt:secret  
jwt-auth secret [S3SghJPest28Wfc1YSszeD9pyogEBn19KgtsU3xi1dafxL9W5XoitWQ0fVW6x4C9] set successfully.
```

Setelah proses generate key berhasil maka pada file **.env** akan muncul sintaks untuk menginisialisasi JWT Secret

```
JWT_SECRET=S3SghJPest28Wfc1YSszeD9pyogEBn19KgtsU3xi1dafxL9W5XoitWQ0fVW6x4C9
```

Menambahkan sintaks implements JWT pada file **./app/Models/User.php** dengan sintaks seperti pada gambar

```
9 use Laravel\Sanctum\HasApiTokens;
10 use Tymon\JWTAuth\Contracts\JWTSubject;
11
12 class User extends Authenticatable implements JWTSubject
13 {
14     use HasApiTokens, HasFactory, Notifiable;
15 }
```

Menambahkan implementasi dari fungsi yang ada dari JWTSubject ke **User.php**

```
46 public function getJWTCustomClaims(){
47     return [];
48 }
49
50 public function getJWTIdentifier(){
51     return $this->getKey();
52 }
53 }
54
```

Menambahkan sintaks api pada file **./config/auth.php** dengan sintaks seperti pada gambar

```
16     'defaults' => [  
17         'guard' => 'api',  
18         'passwords' => 'users',  
19     ],
```

```
38     'guards' => [  
39         'web' => [  
40             'driver' => 'session',  
41             'provider' => 'users',  
42         ],  
43         'api' => [  
44             'driver' => 'jwt',  
45             'provider' => 'users',  
46         ],  
47     ],
```

Membuat controller baru dengan memasukkan perintah
php artisan make:controller AuthController

Kemudian tambahkan fungsi **login** seperti pada gambar

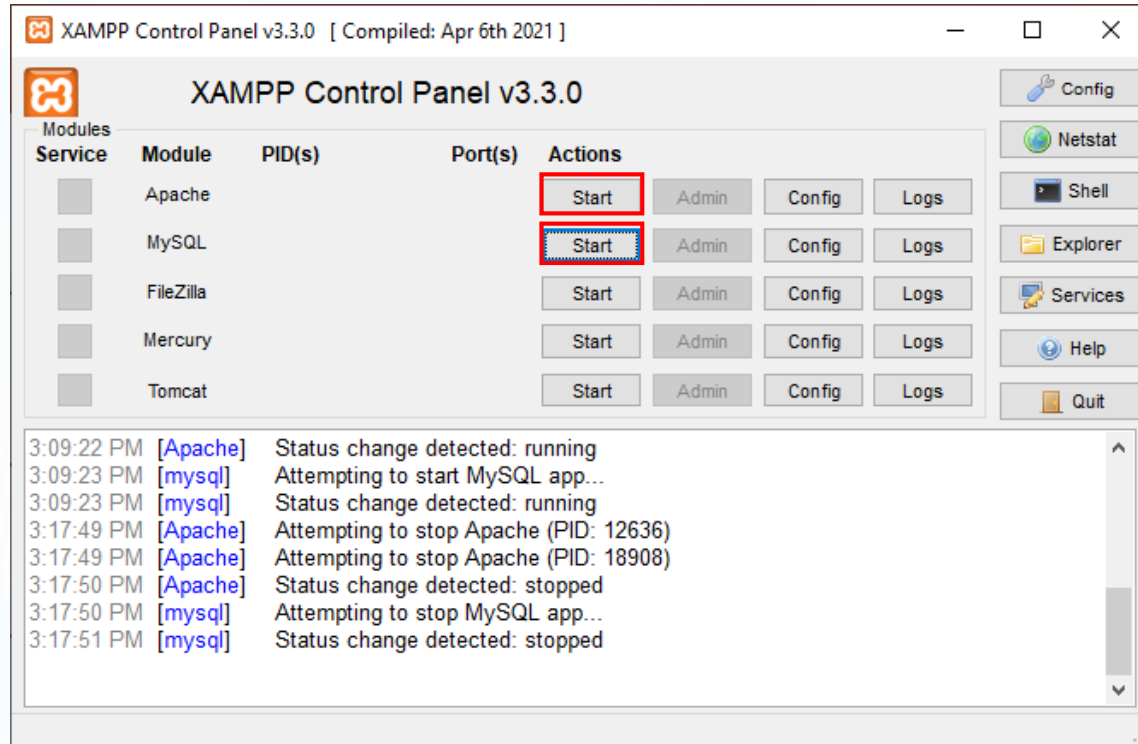
```
public function login(){
    $credentials = request(['email','password']);
    $token = auth() -> attempt($credentials);
    if(!$token){
        return response()->json(['error'=>'Unauthorized'], 401);
    }

    return response()->json([
        'access_token' => $token,
        'token_type' => 'bearer',
        'expires_in' => auth()->factory()->getTTL() * 60
    ]);
}
```

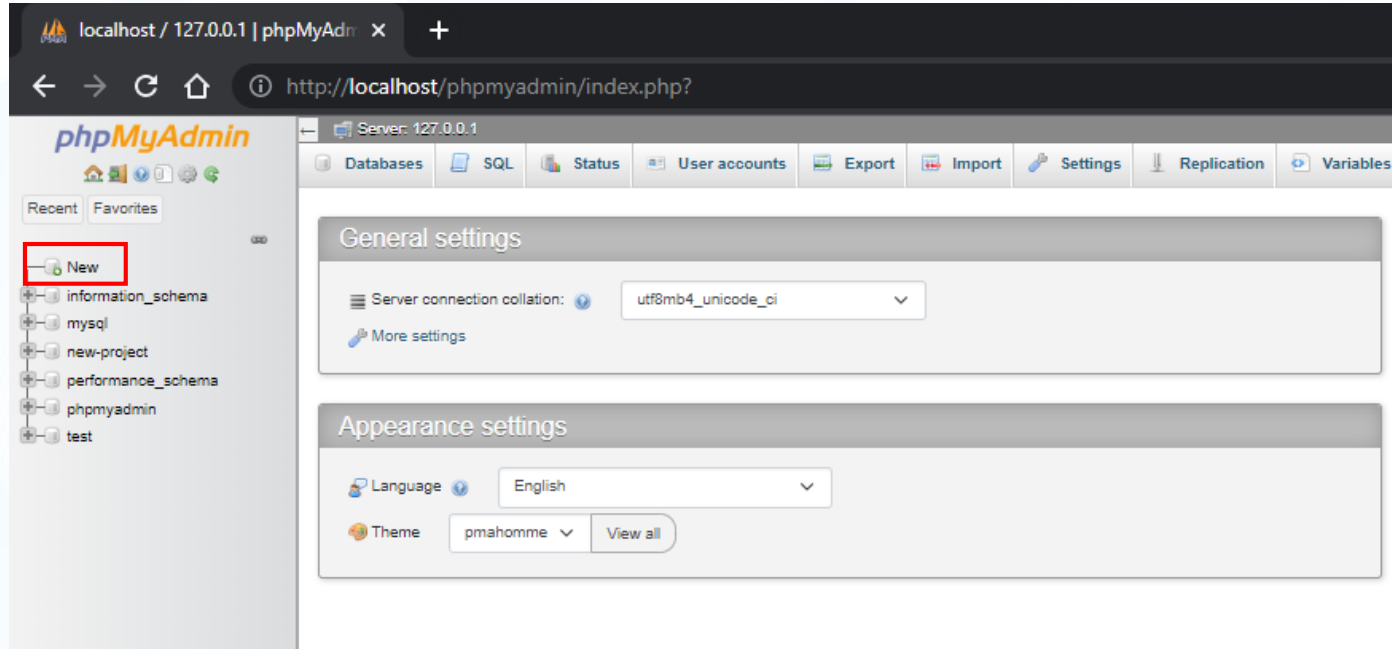
Menambahkan route untuk api pada file **./routes/api.php** dengan sintaks seperti pada gambar

```
Route::group([
    'prefix' => 'auth'
], function() {
    Route::post('login', 'App\Http\Controllers\AuthController@login');
    Route::group([
        'middleware' => 'auth:api'
    ], function(){
        Route::post('logout', 'App\Http\Controllers\AuthController@logout');
        Route::post('refresh', 'App\Http\Controllers\AuthController@refresh');
        Route::get('data', 'App\Http\Controllers\AuthController@data');
    });
});
```

Buka dan jalankan XAMPP dengan menekan tombol Start pada Apache dan MySQL



Buka **phpMyAdmin** kemudian buat database baru dengan menekan tombol **New** seperti pada gambar.



Tambahkan nama database baru, sebagai contoh laravel-jwt. Kemudian menekan tombol **Create** untuk mulai membuat database.

Databases

Create database

laravel-jwt utf8mb4_general_ci

Create

☐ Check all Drop

Mengubah file **.env** pada project laravel yang telah dibuat sebelumnya agar terhubung dengan database yang telah dibuat juga.

```
11 DB_CONNECTION=mysql
12 DB_HOST=127.0.0.1
13 DB_PORT=3306
14 DB_DATABASE=laravel-jwt
15 DB_USERNAME=root
16 DB_PASSWORD=
```

Melakukan migrasi database dengan memasukkan perintah
php artisan migrate

```
PS C:\xampp\htdocs\laravel-jwt> php artisan migrate
Migration table created successfully.
Migrating: 2014_10_12_000000_create_users_table
Migrated: 2014_10_12_000000_create_users_table (102.57ms)
Migrating: 2014_10_12_100000_create_password_resets_table
Migrated: 2014_10_12_100000_create_password_resets_table (49.92ms)
Migrating: 2019_08_19_000000_create_failed_jobs_table
Migrated: 2019_08_19_000000_create_failed_jobs_table (83.60ms)
Migrating: 2019_12_14_000001_create_personal_access_tokens_table
Migrated: 2019_12_14_000001_create_personal_access_tokens_table (77.41ms)
```

Menambahkan data pada database menggunakan seeder dengan mengedit file **DataSeeder.php** pada **.\database\seeders\DatabaseSeeder.php**

```
public function run()
{
    // \App\Models\User::factory(10)->create();
    \App\Models\User::create([
        'name' => 'Jagras',
        'email' => 'jagras@gmail.com',
        'password' => bcrypt('admin')
    ]);
}
```

Mulai menambahkan data melalui seeder dengan memasukkan perintah **php artisan db:seed**

```
PS C:\xampp\htdocs\laravel-jwt> php artisan db:seed
Database seeding completed successfully.
```

Jalankan project dengan memasukkan
php artisan serve

```
PS C:\xampp\htdocs\laravel-jwt> php artisan serve
Starting Laravel development server: http://127.0.0.1:8000
[Tue Aug 23 15:12:27 2022] PHP 7.4.29 Development Server (http://127.0.0.1:8000) started
[Tue Aug 23 15:12:32 2022] 127.0.0.1:58453 Accepted
[Tue Aug 23 15:12:32 2022] 127.0.0.1:58454 Accepted
[Tue Aug 23 15:12:33 2022] 127.0.0.1:58454 Closing
```

Buka dan jalankan aplikasi Postman seperti pada gambar

The screenshot displays the Postman application interface. The top navigation bar includes 'Home', 'Workspaces', 'API Network', and 'Explore'. The main workspace is titled 'My Workspace' and shows a collection of requests. The selected request is a POST method to the endpoint 'http://127.0.0.1:8000/api/auth/login'. The 'Body' tab is active, and the 'form-data' radio button is selected. The form data table contains two entries: 'email' with the value 'jagras@gmail.com' and 'password' with the value 'admin'. The 'Body' tab is also highlighted in the bottom navigation bar. The response is displayed in the 'JSON' view, showing a successful login response with an 'access_token' and 'token_type'.

POST http://127.0.0.1:8000/api/auth/login

Params Authorization Headers (9) **Body** Pre-request Script Tests Settings Cookies

none **form-data** x-www-form-urlencoded raw binary GraphQL

KEY	VALUE	DESCRIPTION
<input checked="" type="checkbox"/> email	jagras@gmail.com	
<input checked="" type="checkbox"/> password	admin	

Body Cookies Headers (10) Test Results 200 OK 1046 ms 694 B Save Response

Pretty Raw Preview Visualize **JSON**

```
1 {
2   "access_token": "eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJpc3MiOiJodHRwOi1wvXC8xMjcucM4wLjE6ODAwMFwvYXBpXC9hdXRoXC9sb2dpbiIsImhhdCI6MTY2MTI0Mjk1MywiZXhwIjoxNjYxMjQ2NTUzLCJyYmY0IjE2NjE5NDI6NTMsImp0aSI6Inp1VGZ5YmtnCHV2eFk4Z3U1LCJzdWIiOiJlbnBydHJzYmY0IjYzZg5NDlMnNjAwYWRlMzllNzAxYzQwMDg3MmRlN2E1OTc2ZjZjZjQ.VTEg1H5wV_D9Tb1SDV7jGznGzo8FJR0VqZ3hdTow0c",
3   "token_type": "bearer",
4   "expires_in": 3600
5 }
```


Buka <https://jwt.io/> melalui browser kemudian paste **access_token** tadi ke **Encoded**.

Encoded

PASTE A TOKEN HERE

```
eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJpc3MiOiJodHRwOlwvXC8xMjcuMC4wLjE6ODAwMFwvYXBpXC9hdXRoXC9sb2dpbiIsImVudCI6MTY2MTc2MTkyMywiZXhwIjoxNjYxNzY1NTIzLCJuYmYiOiJlE2NjE3NjE5MjMsImp0aSI6IjZ4dTZ0aFhWOHZFc2lwSm0iLCJzdWIiOiJEsInBydiI6IjIzYmQ1Yzg5NDlmNjAwYWRiMzllNzAxYzQwMDg3MmRiN2E1OTc2ZjciOiJ0UjUrJskwkMSNphm2wedqrUIi5RM6oxVuBDAWWQ4mH8
```

Decoded

EDIT THE PAYLOAD AND SECRET

HEADER: ALGORITHM & TOKEN TYPE

```
{  
  "typ": "JWT",  
  "alg": "HS256"  
}
```

PAYLOAD: DATA

```
{  
  "iss": "http://127.0.0.1:8000/api/auth/login",  
  "iat": 1661761923,  
  "exp": 1661765523,  
  "nbf": 1661761923,  
  "jti": "6xu6NhXV8vEsipJm",  
  "sub": 1,  
  "prv": "23bd5c8949f600adb39e701c400872db7a5976f7"  
}
```

Menambahkan fungsi **logout** dan **data** pada file **AuthController.php**

```
public function logout(){
    auth()->logout();

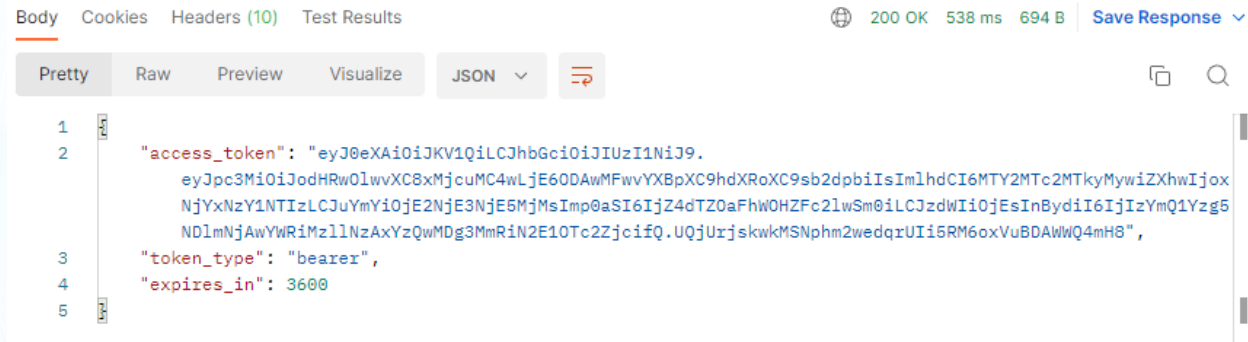
    return response()->json(['message' => 'Successfully logged out'], 200);
}

public function refresh(){

}

public function data(){
    return response()->json(auth()->user());
}
```

Buka Postman kemudian copy **access_token** yang sebelumnya.



Jalankan postman seperti pada gambar dibawah ini.

The screenshot shows the Postman interface for an HTTP GET request to `http://127.0.0.1:8000/api/auth/data`. The **Headers** tab is selected, showing 9 headers. One header is visible: **Authorization** with the value `Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1...`. A red box highlights the `Authorization` key and its value. A red arrow points from the `Bearer` part of the token to a red box containing the text `"Bearer" + access_token`.

KEY	VALUE	DESCRIPTION	...	Bulk Edit	Presets
<input checked="" type="checkbox"/> Authorization	Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1...				
Key	Value	Description			

Setelah menjalankan dengan klik tombol **Send**, maka akan menampilkan hasil data yang telah diinputkan sebelumnya.

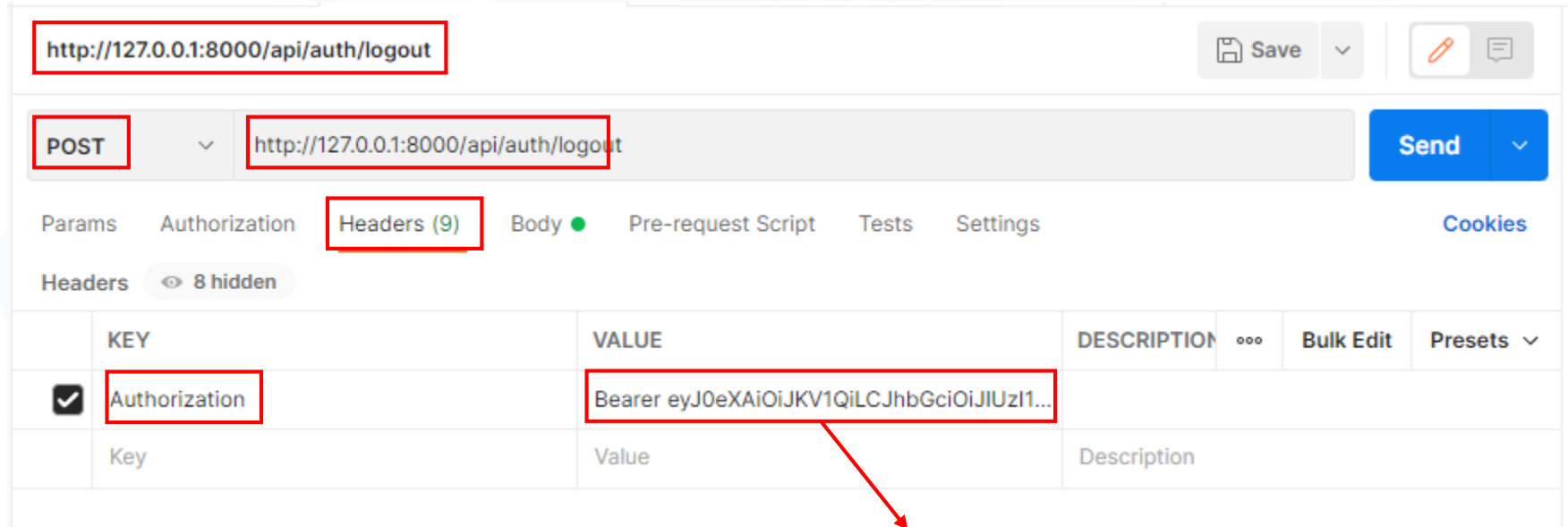


The screenshot shows the 'Body' tab of a web browser's developer tools. The response is a JSON object with the following fields:

```
1 {  
2   "id": 1,  
3   "name": "jagras",  
4   "email": "jagras@gmail.com",  
5   "email_verified_at": null,  
6   "created_at": "2022-08-29T07:20:36.000000Z",  
7   "updated_at": "2022-08-29T07:20:36.000000Z"  
8 }
```

The status bar at the top right indicates a 200 OK response with a 489 ms response time and 470 B of data. There are also icons for 'Save Response' and a search icon.

Jalankan fungsi Logout pada Postman seperti pada gambar.



`"Bearer " + access_token`

Menambahkan fungsi **refresh** pada file **AuthController.php**

```
public function refresh(){  
    return response()->json([  
        'access_token' => auth()->refresh(),  
        'token_type' => 'bearer',  
        'expires_in' => auth()->factory()->getTTL() * 60  
    ]);  
}
```

POST http://127.0.0.1:8000/api/auth/login

Params Authorization Headers (10) Body Pre-request Script Tests Settings

none form-data x-www-form-urlencoded raw binary GraphQL

	KEY	VALUE	DESCRIPTION
<input checked="" type="checkbox"/>	email	jagras@gmail.com	
<input checked="" type="checkbox"/>	password	admin	
	Key	Value	Description

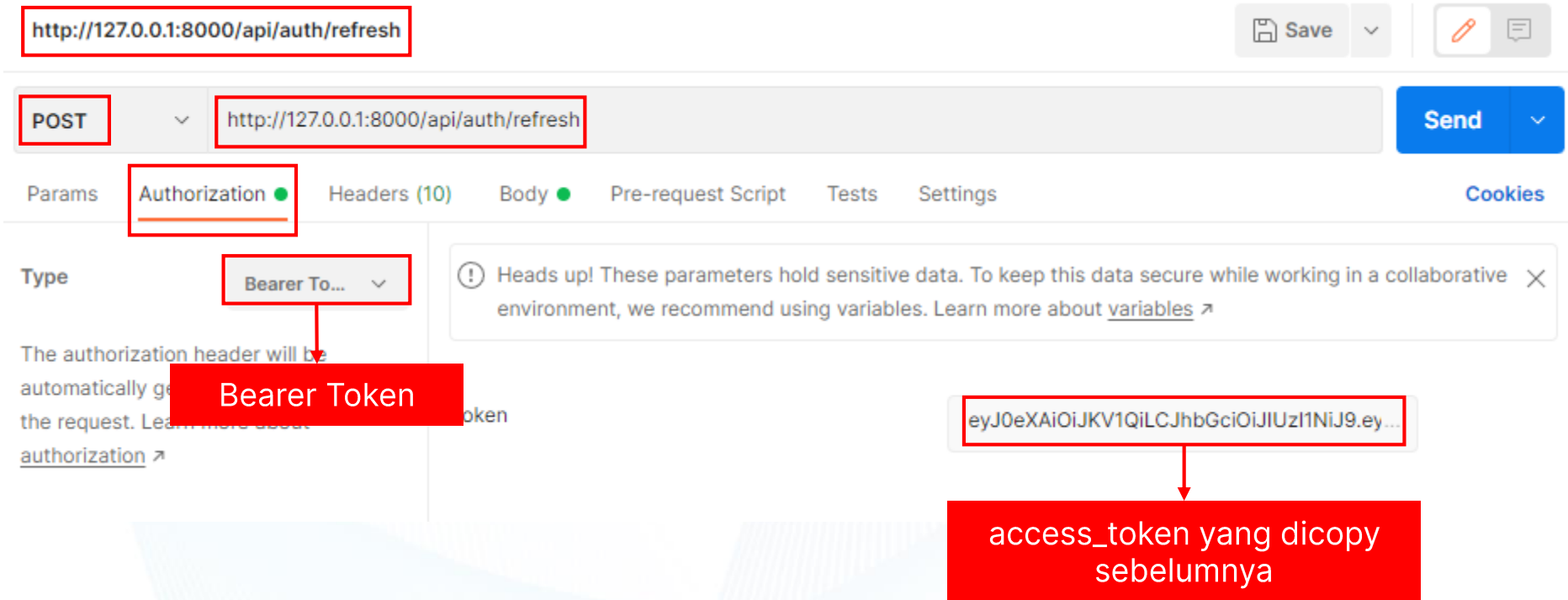
Status: 200 OK Time: 461 ms Size: 694 B Save Response

Pretty Raw Preview Visualize JSON

```

1 {
2   "access_token": "eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJpc3MiOiJodHRwOlwyXC8xMjcuMC4wLjE6ODAwMFwvYXBpXC9hdXRocXc9sb2dpbiIsImldCI6MTY2MTg0NzM3NCwiZXhwIjoxeDQ0DUwOZC0LCJuYmYiOiJlbnRlc2Njc2ZjciQ.ZlTPPhfzv-izGD11_ibuBM026q5pa5cjEJTnULSNIxc",
3   "token_type": "bearer",
4   "expires_in": 3600
5 }
```

Jalankan fungsi Logout pada Postman seperti pada gambar.



Terima **Kasih**