

PROTEJA SUA PRIVACIDADE EM TOD

COMPRE AGORA

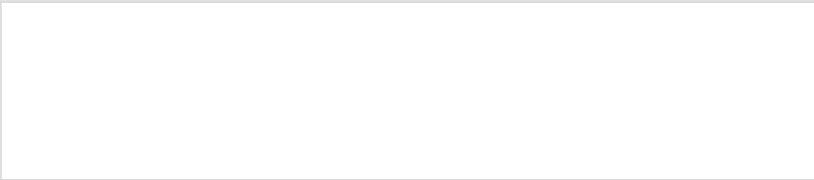
Home > Laravel >

# Laravel JWT Authentication Tutorial



By **Krunal**

Last updated **Feb 24, 2018**



By clicking the subscribe button you will never miss the new articles!

**Subscribe**

**Laravel JWT Authentication Tutorial** we will discuss today. Setting up **JWT Token-based Authentication** in **Laravel 5.6** is easy. The traditional process of interacting with a website is that you log in from the login page. Next, you perform your

desired actions and then log out. However, in the case of **REST API**, the process is entirely different. **JWT (JSON Web Token)** is usually used to send information that can be trusted and verified using a digital signature.

#### Content Overview [hide]

- 1 When we can use JSON Web Tokens
- 2 JWT Authentication
- 3 Laravel JWT Authentication Tutorial
- 4 Step 1: Install and configure Laravel 5.6
- 5 Step 2: Update the config/app.php for JWT package
- 6 Step 3: Register the user into the database.
- 7 Step 4: Test in Postman.
- 8 Step 5: Login the user.
- 9 Step 6: Include middleware to protect the resources.

## When we can use JSON Web Tokens

Let us say; you have a mobile application that needs to communicate with the server. Now, a server needs to identify that whom they are talking. HTTP is the stateless protocol. So we need to create a mechanism that resolves our issue that is how the server will identify that the user is new or old. Is it already registered with the application or not. Is it authorized a user or not? In that scenario, we can use **JWT Authentication**.

## JWT Authentication

1. A user sends a signup post request to the server and server creates a user and JWT token on that database and returns **JWT token** as a response.
2. **JWT** is stored either in the local storage of the browser or any other storage mechanisms.
3. When a user makes another request, it needs to append that token in the request header.
4. The server checked that token and based on whether the JWT token is valid or not; it returns a response.
5. Until a user logs manually out of the application and destroys that token from the local storage, it always checks for the token. If the token is valid, then it can access the particular resources.
6. If the token is either destroyed or manipulated then, user

redirects to the login screen and he needs to fill the username and password.

7. If the username and password are valid then in response it sends a **JWT token** back to the user. Again same procedure, the user stores it in local storage and send that token for every request to verify with the server that he has trusted user and can able to access the particular resources.

# Laravel JWT Authentication Tutorial

As usual, we start this project by installing fresh **Laravel 5.6**.

## Step 1: Install and configure Laravel 5.6

Install Laravel by the following command.

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

Configure the database.

Now, install the third-party jwtauth package by typing the following command.

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

It will install the package in the **vendor** folder and our **composer.json** file will be updated.

## Step 2: Update the config/app.php for JWT package

Go to the **config >> app.php** file and add the following.

```
'providers' => [
    ....
    'Tymon\JWTAuth\Providers\JWTAuthServiceProvi
der',
],
'aliases' => [
    ....
    'JWTAuth' => 'Tymon\JWTAuth\Facades\JWTAuth'
,
    'JWTFactory' => 'Tymon\JWTAuth\Facades\J
WTFactory',
],
```

To publish the configuration file in Laravel, you need to run following line of code :

```
php artisan vendor:publish --provider="Tymon\JWT
Auth\Providers\JWTAuthServiceProvider"
```

Now for token encryption, I need to generate a secret key by running following line of code :

```
php artisan jwt:generate
```

If you find an error like this after hit the above command.

```
ReflectionException : Method Tymon\JWTAuth
\Commands\JWTGenerateCommand::handle() does not
exist
```

You need to do the following step.

Go to **JWTGenerateCommand.php** file located in **vendor/tymon/src/Commands** and paste this part of code `public function handle() { $this->fire(); }`

You can find more about this issue.

## Step 3: Register the user into the database.

Migrate the tables into the database by the following command.

```
php artisan migrate
```

Now, create two controller files for user registration and authentication.

1. **APIRegisterController.php**
2. **APILoginController.php**

Type the following command to generate it.

```
php artisan make:controller APIRegisterController  
php artisan make:controller APILoginController
```

Also, register the api routes inside **routes >> api.php** file.

```
Route::post('user/register', 'APIRegisterController@register');  
Route::post('user/login', 'APILoginController@login');
```

First, we code the **register function** inside **APIRegisterController.php** file.

```
<?php

namespace App\Http\Controllers;
use App\Http\Controllers\Controller;
use Illuminate\Http\Request;
use App\User;
use JWTFactory;
use JWTAuth;
use Validator;
use Response;

class APIRegisterController extends Controller
{
    public function register(Request $request)
    {
        $validator = Validator::make($request->all(), [
            'email' => 'required|string|email|max:255|unique:users',
            'name' => 'required',
            'password' => 'required'
        ]);
        if ($validator->fails()) {
            return response()->json($validator->errors());
        }
        User::create([
            'name' => $request->get('name'),
            'email' => $request->get('email'),
            'password' => bcrypt($request->get('password')),
        ]);
        $user = User::first();
        $token = JWTAuth::fromUser($user);

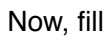
        return Response::json(compact('token'));
    }
}
```

What I have done is that first check the validation, and then if all the form data seems right then, it will register the user in the database and returns a JWT token. We are generating the token based on the User object. You can create the token from anything as you want. You can find more guidance [here](#).

## Step 4: Test in Postman.

We are registering the user through postman. So let us do that.

First, we check the validation.



Yikes!! We have successfully registered the user and get back the JWT token. Now save this token in the local storage and when we need to access any protected resource then pass this token as Auth Bearer to get request, and you can obtain that route.

## Step 5: Login the user.

We have already defined the **login** route in the **api.php** file. Now, go the **APILoginController.php** file and code the **login** function.

```
// APILoginController.php

<?php

namespace App\Http\Controllers;
use App\Http\Controllers\Controller;
use Illuminate\Http\Request;
use Validator;
use JWTFactory;
use JWTAuth;
use App\User;
use Illuminate\Support\Facades\Auth;

class APILoginController extends Controller
{
    public function login(Request $request)
    {
        $validator = Validator::make($request->all(), [
            'email' => 'required|string|email|max:255',
            'password'=> 'required'
        ]);
        if ($validator->fails()) {
            return response()->json($validator->errors());
        }
        $credentials = $request->only('email', 'password');
        try {
            if (! $token = JWTAuth::attempt($credentials)) {
                return response()->json(['error' => 'invalid_credentials'], 401);
            }
        } catch (JWTException $e) {
            return response()->json(['error' => 'could_not_create_token'], 500);
        }
        return response()->json(compact('token'))
    };
}
```

If the email and password are correct then we can generate the JWT token otherwise we get an error. Now check this in the



[illegible]

The `tymondesigns/jwt-auth` package provides us by default two middlewares.

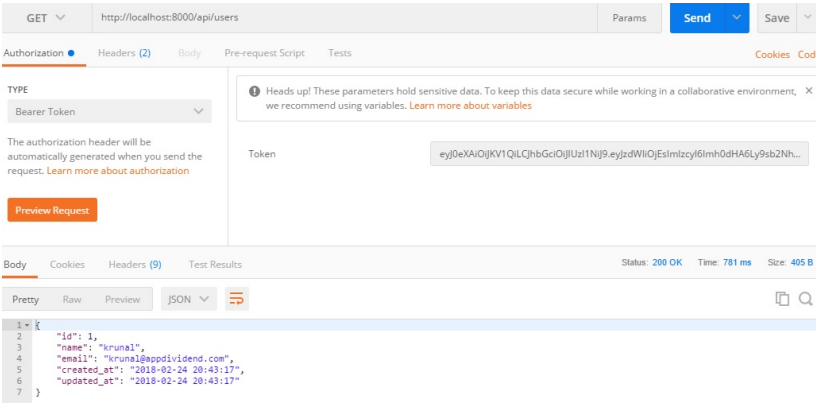
- We just need to register these middlewares into **app >> Http >> Kernel.php** file.

Define one route that needs to be protected via **JWT Token Authentication**.

Now, login to the application and get the token. Now, we can use


this token in the get request like the following.

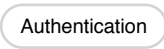
```
Authorization: Bearer {yourtokenhere}
```




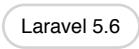
And, we are getting the User back. So Our fully functional **Laravel JWT Authentication Tutorial Example** is working. Finally, **Laravel JWT Authentication Tutorial** is over. I have put this code on Github. So please check that out as well.


FORK ME

 API

 Authentication

 Laravel

 Laravel 5.6

 Laravel API Authentication



### Krunal

I am Web Developer and Blogger. I have created this website for the web developers to understand complex concepts in an easy manner.

[report this ad](#)

## 11 Comments



Biobii Says 9 months ago

How to use JWT with multiple Models (also multiple tables). I mean like User and Admin models. Please create the tutorial for this case, thank you!



Renan Says 9 months ago

You should implement it yourself using guards. Please look at my tutorial about how to build a multi auth application with Laravel:  
<https://medium.com/@renandiett/laravel-5-4-trabalhando-com-autentica%C3%A7%C3%B5es-independentes-sem-packages-adicionais-6e50c11a0b79>

Please note that the text was made with my language (Portuguese).

Furthermore you may try DevMarketer tutorial:  
<https://www.youtube.com/watch?list=PLwAKR305CRO9S6KVHMJYqZpjPzGPWuQ7Q&v=iKRLrJXNN4M>



**Paul Says** 📅 7 months ago

You can make a check to your models and if the checks are okay by your clause or filters, then use the JWTFACORY to create a token and send.



**Webfacer Says** 📅 9 months ago

how do i destroy a token? for logout the user?



**Arjun Says** 📅 9 months ago

Step1: Define logout route in routes/api.php

```
Route::group(['middleware' => 'jwt.auth'],  
function(){  
Route::post('auth/logout',  
'AuthController@logout');  
});
```

Step2: In controller

```
public function logout()  
{  
JWTAuth::invalidate();  
return response([  
'status' => 'success',  
'msg' => 'Logged out Successfully.'  
, 200);  
}
```



**Meganathan Says** 📅 7 months ago

Hi,

I am getting

```
{  
"error": "token_not_provided"  
}
```

Even, i am sending Authorization header. Please help.  
its urgent issue. Needs to resolve ASAP

**Indra Says** 📅 7 months ago



Can you explain to me how to have that Bearer type authorization and what is your header set up?



**Dmitry Says** 📅 4 months ago

Hello!

I have cloned you GitHub repository and I have installed your app. When I send POST request to /api/user/register I get error message: SQLSTATE[HY000]: General error: 1364 Field 'api\_token' doesn't have a default value.

How can i fix it?



**Python For Beginners Says** 📅 4 months ago

Cool. Definitely gott my internet working now. Thanks a lot guys.



**Anubhav Says** 📅 2 weeks ago

How to secure this token can any one help me out here ?



**Luongyen Says** 📅 4 days ago

I want to get the api to pass to the router for the client after logon how to do

This site uses Akismet to reduce spam. [Learn how your comment data is processed.](#)