Step 1: Install the JWT Package

Laravel doesn't provide JWT authentication out of the box, so you'll need to install the tymon/jwt-auth package, which is widely used for handling JWT in Laravel.

Run this command to install the package:

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

Step 2: Publish the Configuration

After the package is installed, publish its configuration file using this command:

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

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

This will create a config/jwt.php file where you can configure various JWT settings.

Step 3: Generate the JWT Secret Key

Next, generate a secret key for JWT. This key is used to sign the tokens and should be kept secure.

Run the following command:

```
php artisan jwt:secret
```

This will update your .env file with a new JWT_SECRET key.

Step 4: Set Up Authentication Guards

In the config/auth.php file, modify the api guard to use JWT as the driver:

```
'guards' => [
    'api' => [
        'driver' => 'jwt',
        'provider' => 'users',
    ],
],
```

This tells Laravel to use JWT for API authentication.

Step 5: Add the JWT Trait to Your User Model

In your User model (app/Models/User.php or app/User.php), use the Tymon\JWTAuth\Contracts\JWTSubject interface and implement its required methods.

```php
namespace App\Models;

use Tymon\JWTAuth\Contracts\JWTSubject;
use Illuminate\Foundation\Auth\User as Authenticatable;

class User extends Authenticatable implements JWTSubject
{
    // Implement the required methods for JWT
    public function getJWTIdentifier()
    {
        return $this->getKey();
    }

    public function getJWTCustomClaims()
    {
        return [];
    }
}
```

In end :

```php
public function login(Request $request)
{
    $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(['token' => $token]);
}
```

For example, in `routes/api.php`:

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

You can also use JWT middleware to handle token authentication automatically. In `app/Http/Kernel.php`, add JWT middleware:

```php
'jwt.auth' => \Tymon\JWTAuth\Http\Middleware\Authenticate::class,
'jwt.refresh' => \Tymon\JWTAuth\Http\Middleware\RefreshToken::class,
```

This allows you to apply jwt.auth middleware to routes for automatic JWT handling.