API Rest avec Laravel 5.6 Passport Authentication - Reset Password (Partie 4)

Alfredo Barron
2 août 2018 · 6 min de lecture

Nous apprendrons à créer un système de réinitialisation de mot de passe

https://www.vecteezy.com

Étape 1. Mettre à jour la migration

Dans la première étape, nous devons mettre à jour le fichier de migration `database/migrations/xxx_create_password_resets_table.php` comme le code ci-dessous:

```
fonction publique up ()
{
    Schema :: create ('password_resets', function (Blueprint $
table) {
```

```
            $ table-> increments ('id');
             $ table-> string ('email') -> index ();
            $ table-> string ('token');
            $ table-> timestamps ();
        });
    }
```

Étape 2. Créer un modèle PasswordReset

Ouvrez votre terminal ou votre invite de commande et exécutez la commande ci-dessous:

```
php artisan make: model PasswordReset
```

Cette commande va créer un `app/PasswordReset.php` fichier, dans cet ensemble de fichiers entrées remplissables.

```
La classe PasswordReset étend le modèle
{
    protected $ fillable = [
        'email', 'token'
    ];
}
```

Étape 3. Créez des notifications

Nous créons deux notifications `PaswordResetRequest` et `PasswordResetSuccess` , dans votre terminal ou invite de commande, exécutez les commandes ci-dessous:

```
php artisan make: notification PasswordResetRequest

php artisan make: notification PasswordResetSuccess
```

Cette commande créera des fichiers `app/Notifications/PasswordResetRequest.php` et `app/Notifications/PasswordResetSuccess.php` .

Dans le `PasswordResetRequest.php` fichier, ajoutez le code suivant:

```
<? php

namespace App \ Notifications;

utilisez Illuminate \ Bus \ Queueable;
utilisez Illuminate \ Notifications \ Notification;
utilisez Illuminate \ Contracts \ Queue \ ShouldQueue;
utilisez Illuminate \ Notifications \ Messages \ MailMessage;

class PasswordResetRequest étend Notification implémente ShouldQueue
{
    use Queueable;

$ token protégé;

    / **

    * Créez une nouvelle instance de notification.

    *

    *

@return

 void

    * /

    fonction publique __construct ($ token)

    {

$ this-> token = $ token;

    }

    / **

    * Obtenez les canaux de distribution de la notification.

    *

    *

@param
```

```
  mixed  $notifiable
    * @return array
    */
    public function via($notifiable)
    {
        return ['mail'];
    }

     /**
     * Get the mail representation of the notification.
     *
     * @param  mixed  $notifiable
     * @return \Illuminate\Notifications\Messages\MailMessage
     */
     public function toMail($notifiable)
     {
        $url = url('/api/password/find/'.$this->token);

        return (new MailMessage)
            ->line('You are receiving this email because we
received a password reset request for your account.')
            ->action('Reset Password', url($url))
            ->line('If you did not request a password reset, no
further action is required.');
    }

    /**
    * Get the array representation of the notification.
    *
    * @param  mixed  $notifiable
    * @return array
    */
    public function toArray($notifiable)
    {
        return [
            //
        ];
    }
}
```

In the `PasswordResetSuccess.php` file add the next code:

```
<?php

namespace App\Notifications;
```

```
use Illuminate\Bus\Queueable;
use Illuminate\Notifications\Notification;
use Illuminate\Contracts\Queue\ShouldQueue;
use Illuminate\Notifications\Messages\MailMessage;

class PasswordResetSuccess extends Notification implements
ShouldQueue
{
    use Queueable;

    /**
    * Create a new notification instance.
    *
    * @return void
    */
    public function __construct()
    {
        //
    }

    /**
    * Get the notification's delivery channels.
    *
    * @param  mixed  $notifiable
    * @return array
    */
    public function via($notifiable)
    {
        return ['mail'];
    }

    /**
    * Get the mail representation of the notification.
    *
    * @param  mixed  $notifiable
    * @return \Illuminate\Notifications\Messages\MailMessage
    */
    public function toMail($notifiable)
    {
        return (new MailMessage)
            ->line('You are changed your password successful.')
            ->line('If you did change password, no further action is
required.')
            ->line('If you did not change password, protect your
account.');
    }

/**
     * Get the array representation of the notification.
     *
     * @param  mixed  $notifiable
     * @return array
     */
    public function toArray($notifiable)
    {
        return [
            //
```

```
            ];
        }
    }
```

Step 4. Create API Routes

We will create api routes. Laravel provide `routes/api.php` file for write web services route. So, let's add new route on that file.

```
<?php

use Illuminate\Http\Request;

...

Route::group([
    'namespace' => 'Auth',
    'middleware' => 'api',
    'prefix' => 'password'
], function () {
    Route::post('create', 'PasswordResetController@create');
    Route::get('find/{token}', 'PasswordResetController@find');
    Route::post('reset', 'PasswordResetController@reset');
});
```

Step 5: Install Dependencies

We use Carbon package to help with dates, in your terminal run bellow command:

```
composer require nesbot/carbon
```

Step 6: Create Controller

In this step we have to create new controller and three api method. So let's create `PasswordResetController` and put bellow code:

```
<?php

namespace App\Http\Controllers\Auth;

use App\Http\Controllers\Controller;
use Illuminate\Http\Request;
```

```
use Carbon\Carbon;
use App\Notifications\PasswordResetRequest;
use App\Notifications\PasswordResetSuccess;
use App\User;
use App\PasswordReset;

class PasswordResetController extends Controller
{
    /**
     * Create token password reset
     *
     * @param  [string] email
     * @return [string] message
     */
    public function create(Request $request)
    {
        $request->validate([
            'email' => 'required|string|email',
        ]);

        $user = User::where('email', $request->email)->first();

        if (!$user)
            return response()->json([
                'message' => 'We can't find a user with that e-mail
address.'
            ], 404);

        $passwordReset = PasswordReset::updateOrCreate(
            ['email' => $user->email],
            [
                'email' => $user->email,
                'token' => str_random(60)
             ]
        );

        if ($user && $passwordReset)
            $user->notify(
                new PasswordResetRequest($passwordReset->token)
            );

        return response()->json([
            'message' => 'We have e-mailed your password reset
link!'
        ]);
    }

    /**
     * Find token password reset
     *
     * @param  [string] $token
     * @return [string] message
     * @return [json] passwordReset object
     */
    public function find($token)
    {
```

```
        $passwordReset = PasswordReset::where('token', $token)
            ->first();

        if (!$passwordReset)
            return response()->json([
                'message' => 'This password reset token is invalid.'
            ], 404);

        if (Carbon::parse($passwordReset->updated_at)-
>addMinutes(720)->isPast()) {
            $passwordReset->delete();
            return response()->json([
                'message' => 'This password reset token is invalid.'
            ], 404);
        }

        return response()->json($passwordReset);
    }

     /**
     * Reset password
     *
     * @param  [string] email
     * @param  [string] password
     * @param  [string] password_confirmation
     * @param  [string] token
     * @return [string] message
     * @return [json] user object
     */
    public function reset(Request $request)
    {
        $request->validate([
            'email' => 'required|string|email',
            'password' => 'required|string|confirmed',
            'token' => 'required|string'
        ]);

        $passwordReset = PasswordReset::where([
            ['token', $request->token],
            ['email', $request->email]
        ])->first();

        if (!$passwordReset)
            return response()->json([
                'message' => 'This password reset token is invalid.'
            ], 404);

        $user = User::where('email', $passwordReset->email)-
>first();

        if (!$user)
            return response()->json([
                'message' => 'We can't find a user with that e-mail
address.'
            ], 404);
```

```
            $user->password = bcrypt($request->password);
            $user->save();

            $passwordReset->delete();

            $user->notify(new PasswordResetSuccess($passwordReset));

            return response()->json($user);
        }
    }
```

Now we are ready to run our example so run bellow command to quick run:

```
php artisan serve
```

. . .

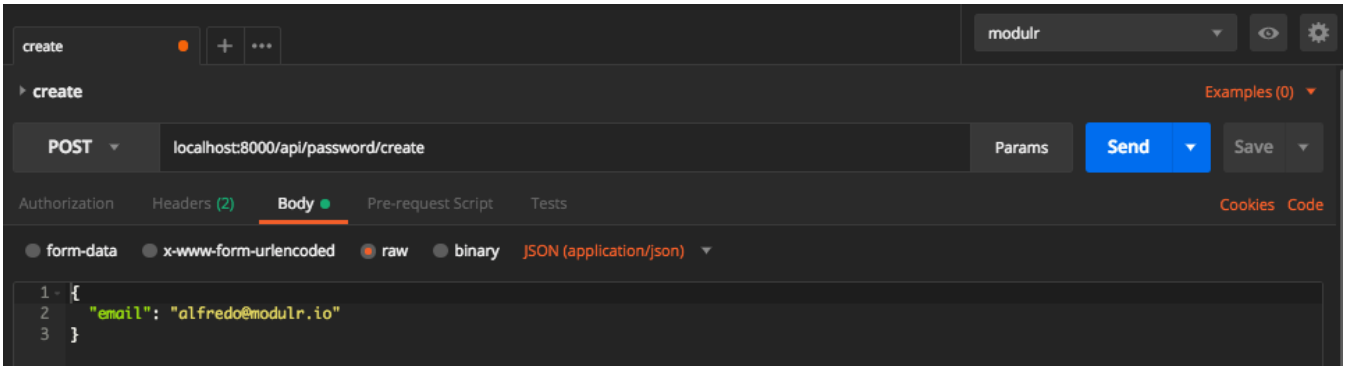
Tests

Now, we can simple test by rest client tools (Postman), So I test it and you can see below screenshots.

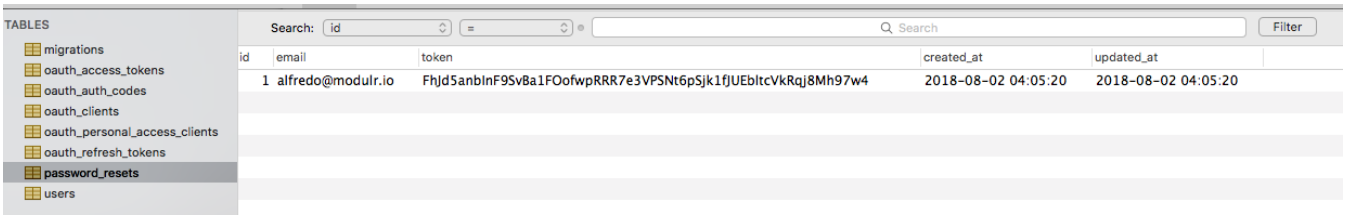
> *In this api you have to set two header as listed below:*

```
Content-Type: application/json
X-Requested-With: XMLHttpRequest
```

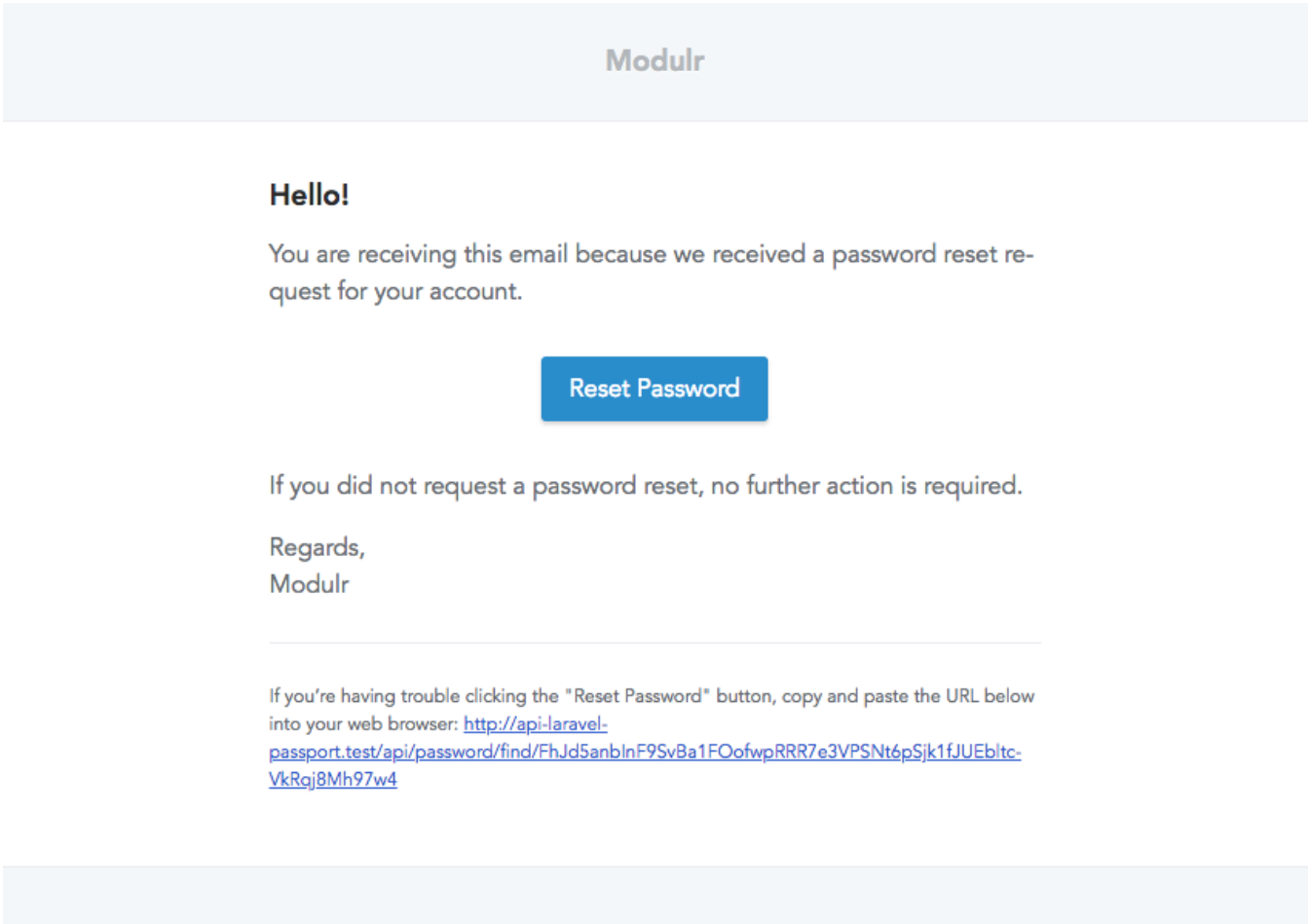
Request Password Reset



When request password reset will created token in data base

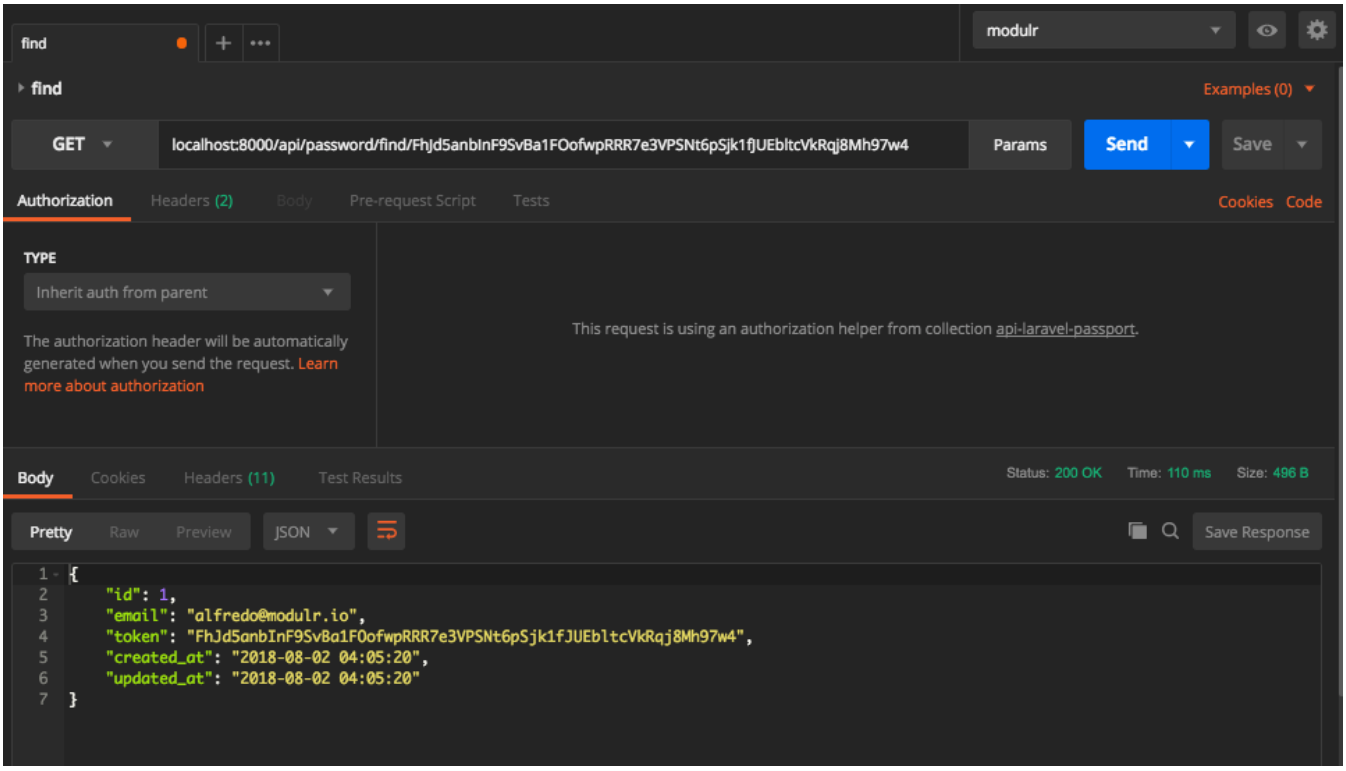


When request password reset will receive a email with the link to Reset Password.

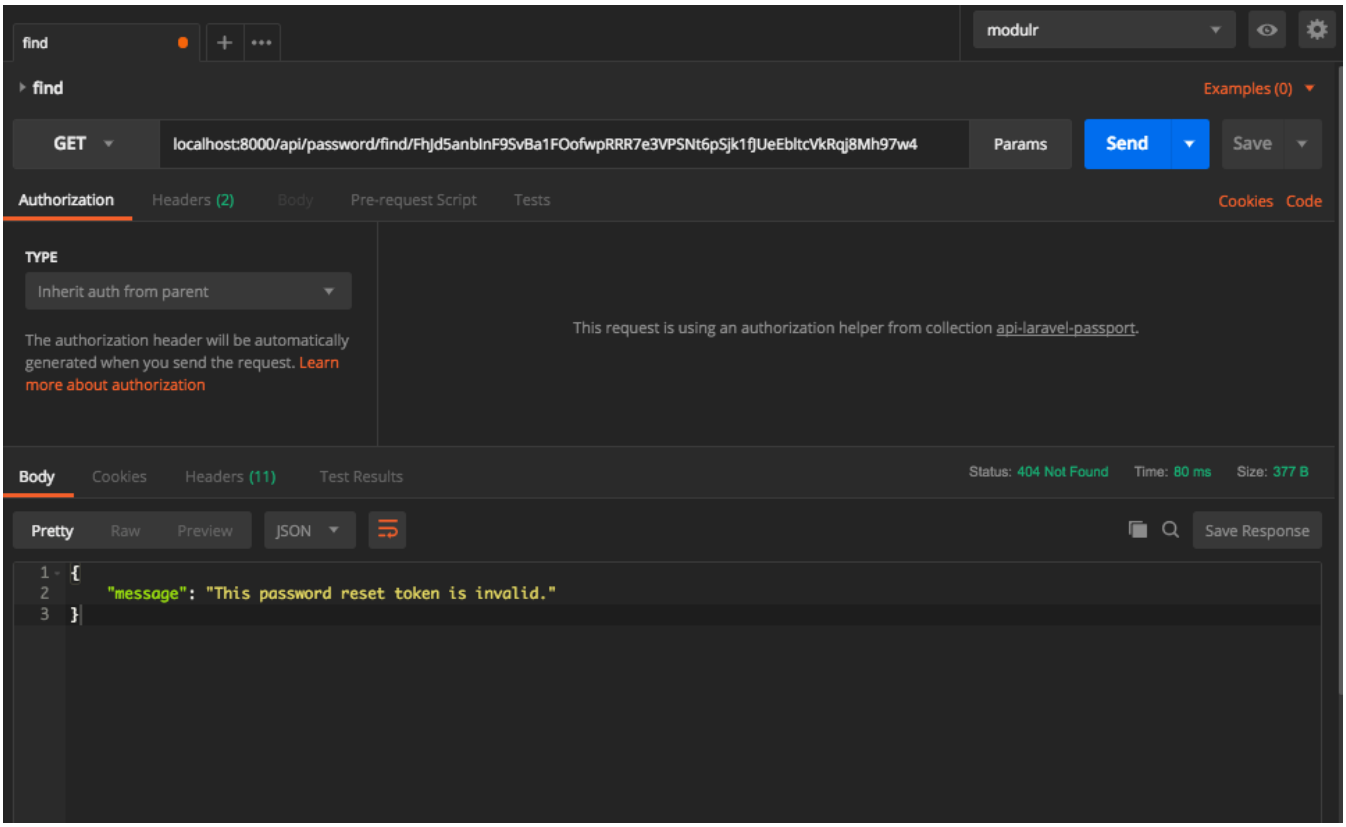


Find Password Reset

If the token is active response with token data

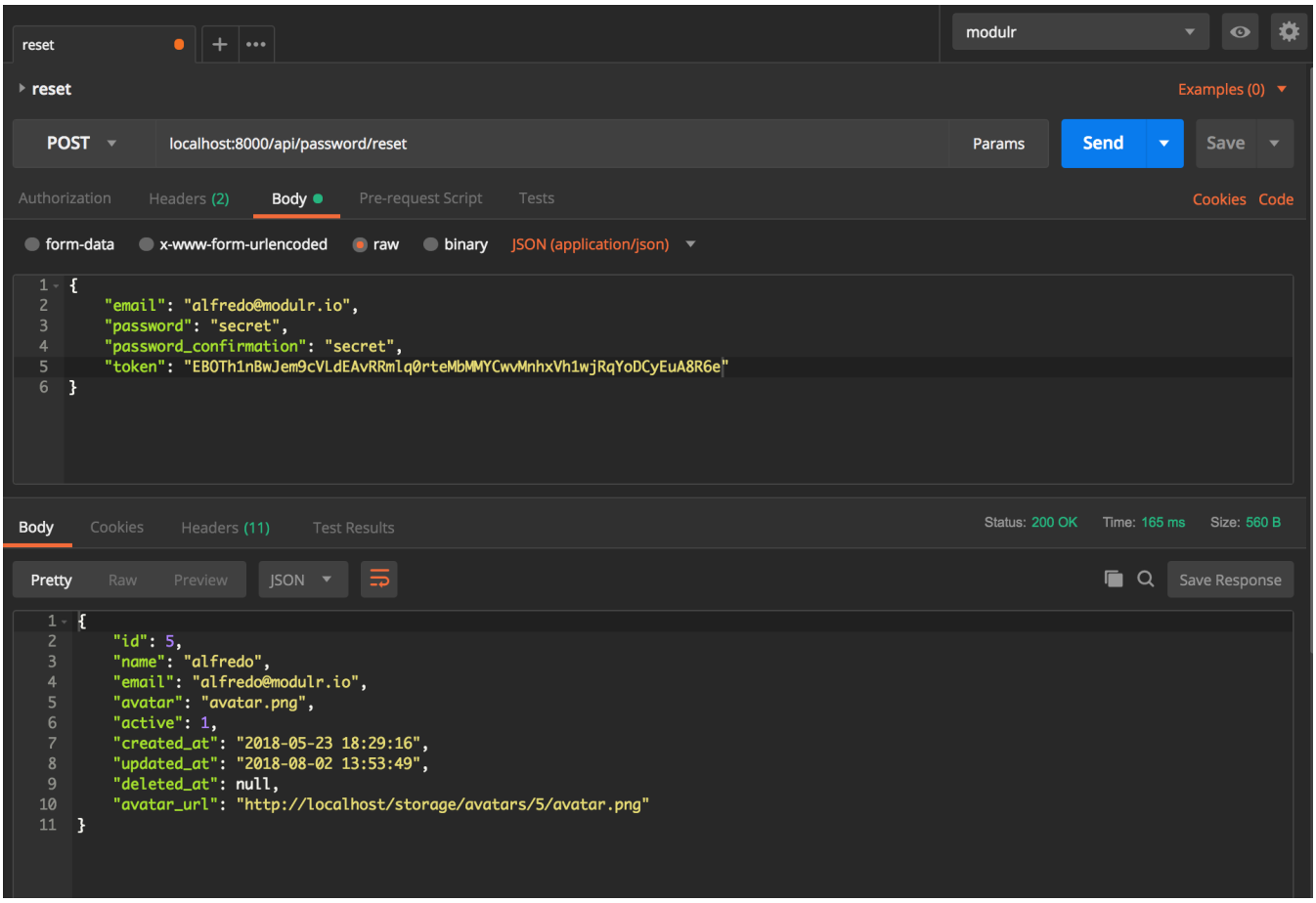


If the token is not active the response is a 404 error

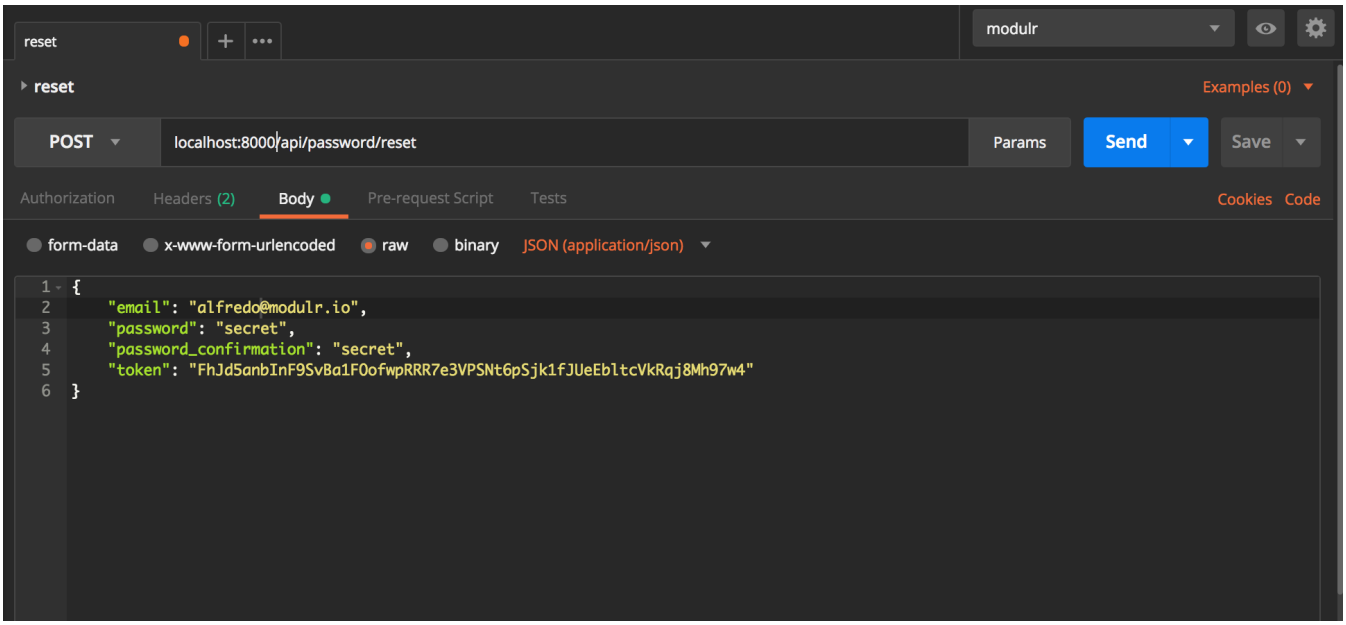


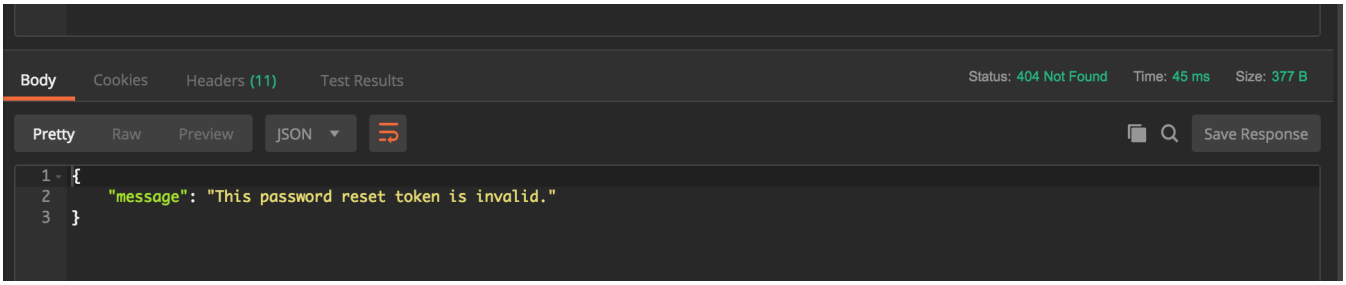
Reset Password

If the token is active response with User data



If the token is not active the response is a 404 error





. . .

> *Thanks for reading! I'm Alfredo Barrón, Feel free to connect with me via Twitter.*

Part 1. Passport Authentication

Part 2. Confirm account + notifications

Part 3. Generate avatar

Part 4. Reset Password

Part 5. Send Notifications with Queues on Redis

Ressources

Collections de GitHub

Postman

)

Laravel Authentification Mots de passe API Passeport

A propos aidejuridique de l'