Laravel Telescope

Telescope est un nouvel assistant pour déboguer une application Laravel. IL nous donne accès à une foule d'informations sur les requêtes qui entrent dans l'application, sur les exceptions, les requêtes à la base de données, les files d'attente (queues), les mails, les notifications, le cache…

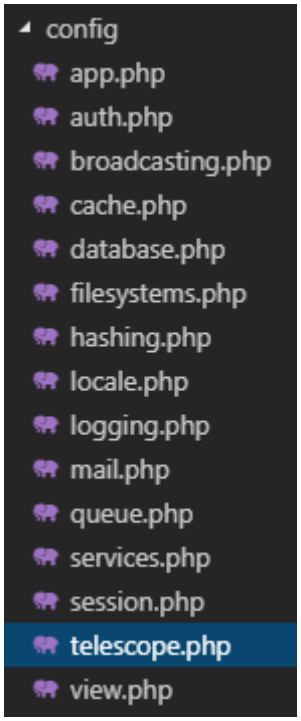
Telescope en est encore au stade beta mais il est déjà largement utilisable. Je vous propose dans cet article de regarder un peu ses possibilités. On va ainsi enfin disposer pour Laravel d'un outil digne de ce nom !

Installation

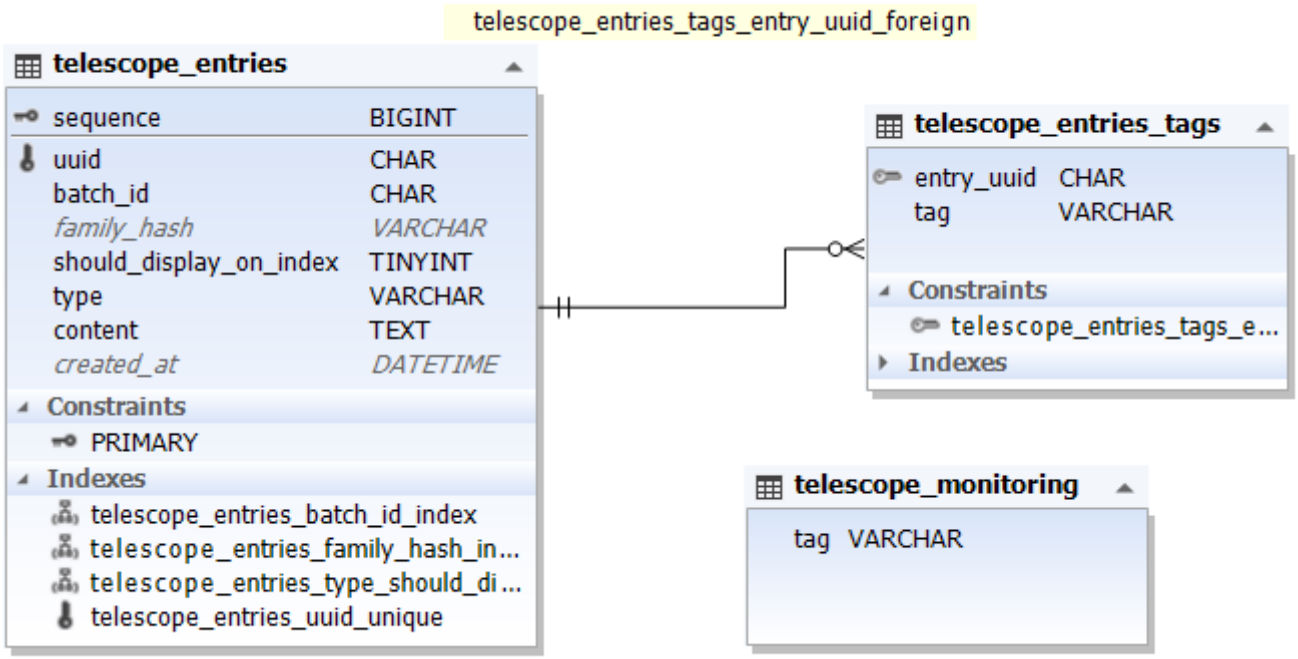
Pour installer Telescope c'est tout simple, mais il faut au minimum la version 5.7.7 de Laravel :

```
composer require laravel/telescope --dev
php artisan telescope:install
php artisan migrate
```

On se retrouve avec un fichier de configuration :



Dans ce fichier on trouve par défaut comme driver **database** avec une connexion **MySQL**. Avec la migration qu'on a faite on se retrouve avec ces 3 tables :



On a tous ces observateurs par défaut :

```
'watchers' => [
    Watchers\CacheWatcher::class => env('TELESCOPE_CACHE_WATCHER', true),
    Watchers\CommandWatcher::class => env('TELESCOPE_COMMAND_WATCHER', true),
    Watchers\DumpWatcher::class => env('TELESCOPE_DUMP_WATCHER', true),
    Watchers\EventWatcher::class => env('TELESCOPE_EVENT_WATCHER', true),
    Watchers\ExceptionWatcher::class => env('TELESCOPE_EXCEPTION_WATCHER', true),
    Watchers\JobWatcher::class => env('TELESCOPE_JOB_WATCHER', true),
    Watchers\LogWatcher::class => env('TELESCOPE_LOG_WATCHER', true),
    Watchers\MailWatcher::class => env('TELESCOPE_MAIL_WATCHER', true),
    Watchers\ModelWatcher::class => env('TELESCOPE_MODEL_WATCHER', true),
    Watchers\NotificationWatcher::class => env('TELESCOPE_NOTIFICATION_WATCHER', true),
```

```
    Watchers\QueryWatcher::class => [
        'enabled' => env('TELESCOPE_QUERY_WATCHER', true),
        'slow' => 100,
    ],

    Watchers\RedisWatcher::class => env('TELESCOPE_REDIS_WATCHER', true),
    Watchers\RequestWatcher::class => env('TELESCOPE_REQUEST_WATCHER', true),
    Watchers\ScheduleWatcher::class => env('TELESCOPE_SCHEDULE_WATCHER', true),
],
```

On peut donc enlever ceux qui ne nous sont pas utiles.

Un réglage important est le nombre maximum d’enregistrements par observateur :

```
'limit' => env('TELESCOPE_LIMIT', 100),
```

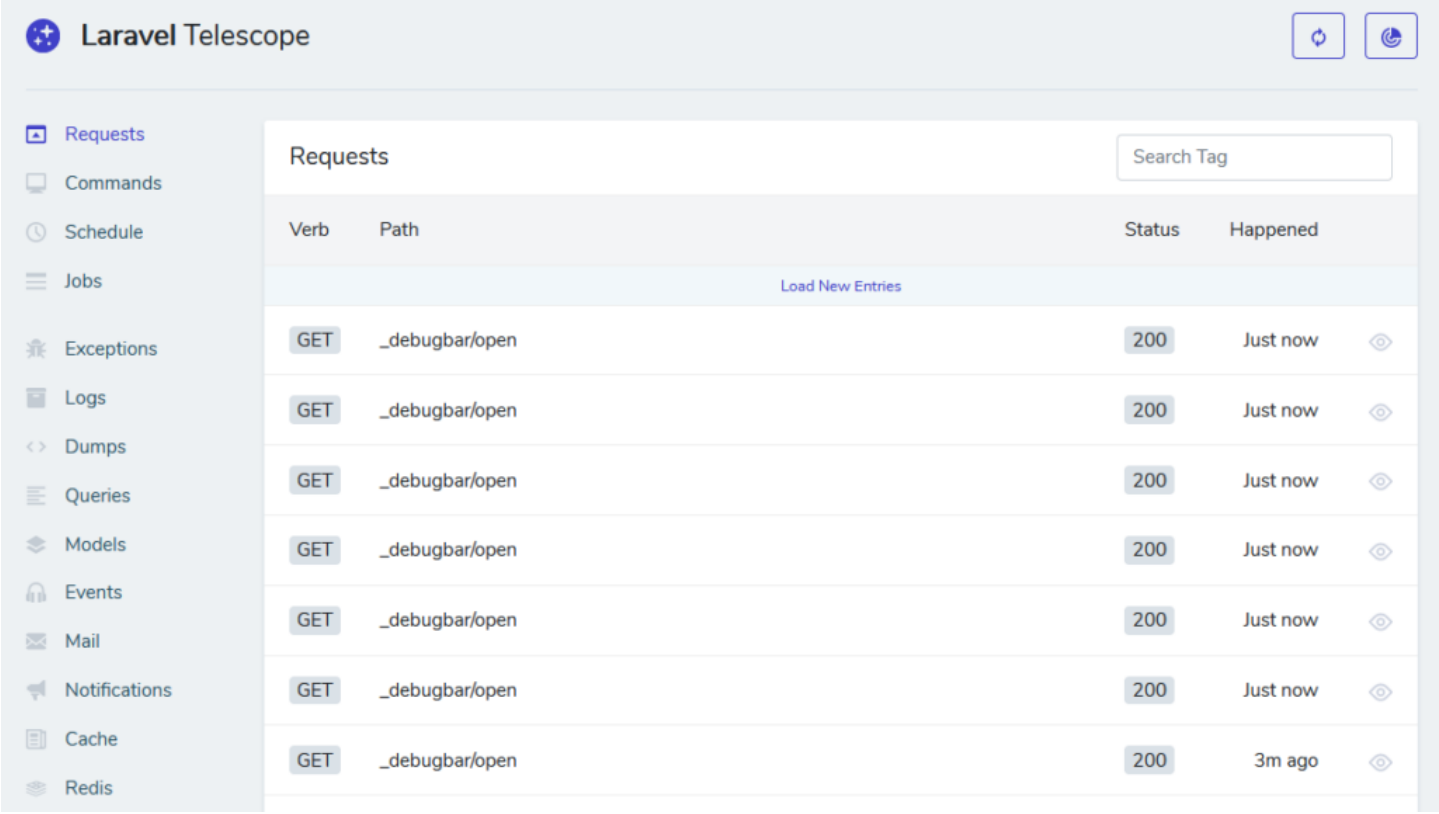
Par défaut on a 100, évidemment si on augmente cette valeur il y aura un impact sur les données stockées dans la base.

Pour accéder à Telescope il faut utiliser cette url :

http://mondomaine/telescope

Cette url est aussi réglable dans la configuration :

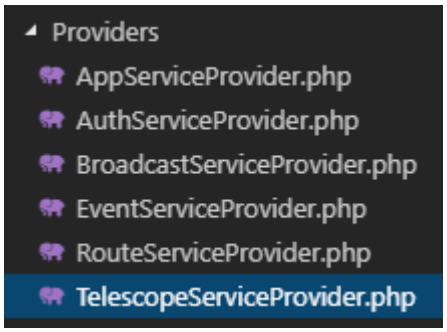
```
'path' => 'telescope',
```



Par défaut Telescope n'est accessible que dans l'environnement local :

```
APP_ENV=local
```

C'est logique puisque que c'est comme ça qu'on développe mais il peut arriver d'en avoir besoin dans l'environnement de production. Telescope crée un provider :



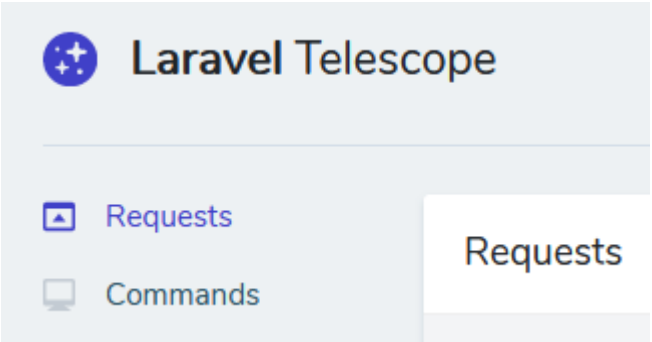
Dans ce provider on trouve une autorisation où vous pouvez ajouter les mails des personnes autorisées dans l'environnement de production :

```
protected function gate()
{
    Gate::define('viewTelescope', function ($user) {
        return in_array($user->email, [
```

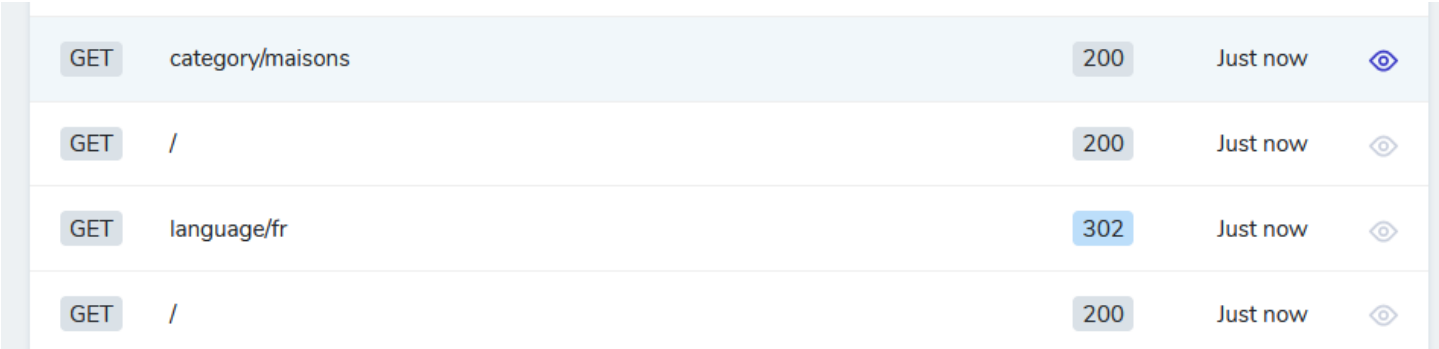
```
            'bestmomo@chezlui.net'
        ]);
    });
}
```

Maintenant qu'on a fait le tour pour l'installation et les réglage ouvrons un peu la boîte…

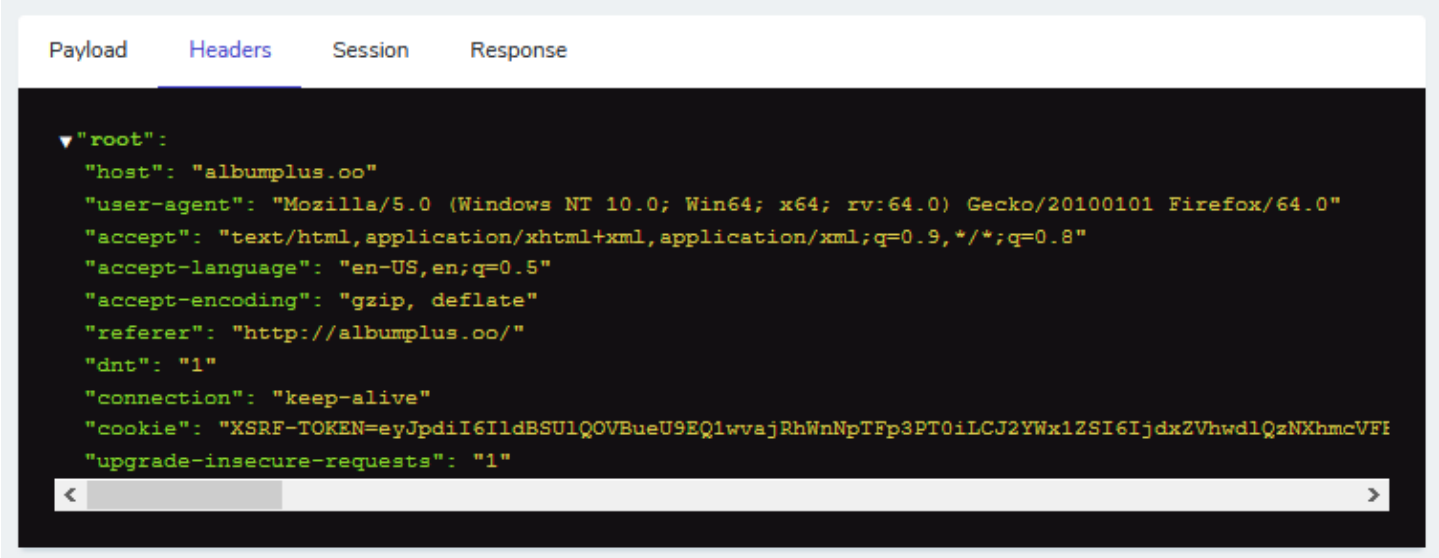
Les requêtes



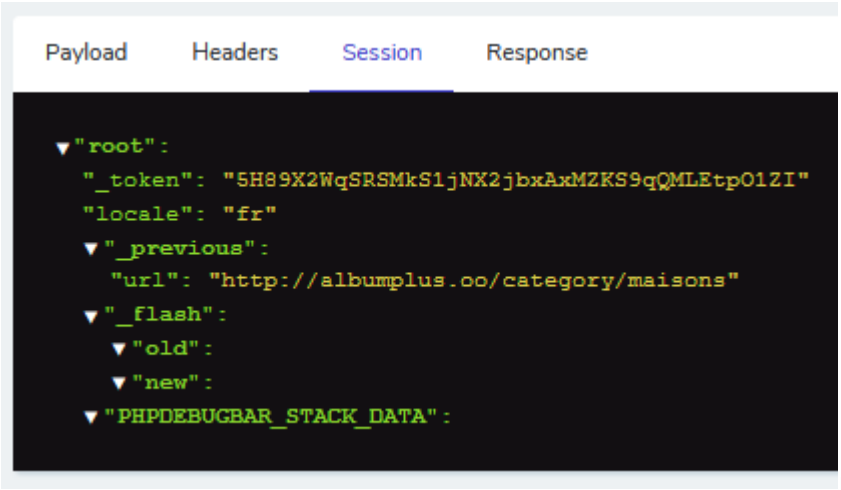
On a la liste des requêtes avec le verbe, l'url, le code HTTP et quand elle est passée. On dispose aussi de l'icône d'un œil pour obtenir plus de renseignements concernant une requête :



On dispose alors de tous les détails comme les **headers** :

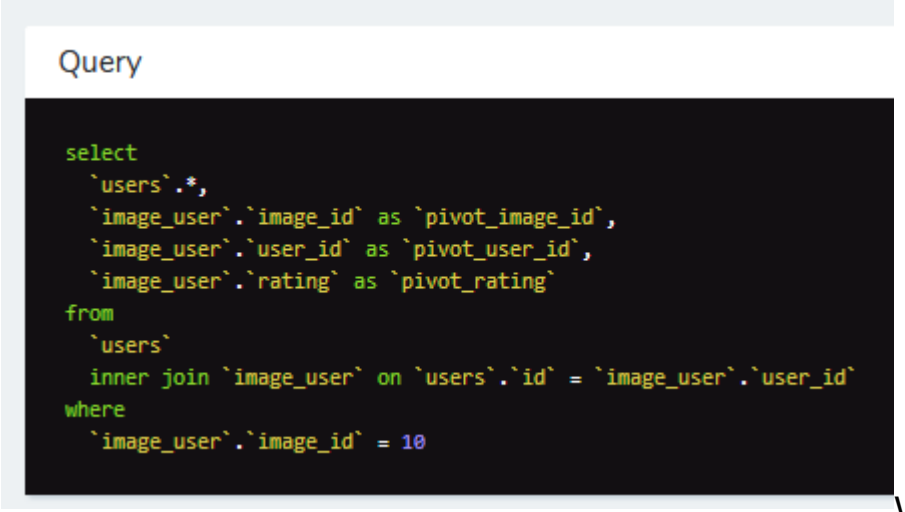


La session :



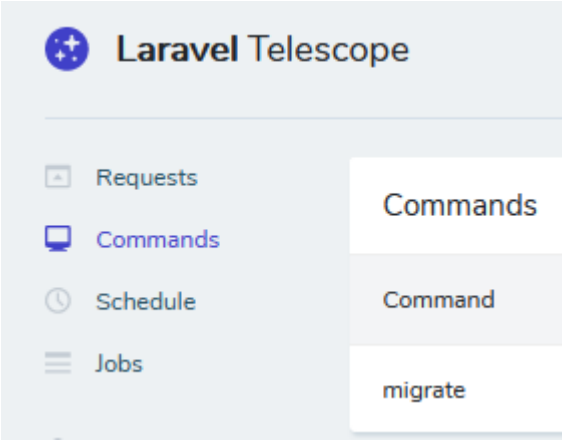
Les requêtes générées à la base :

Avec pour chacune le détail en cliquant sur l’icône :

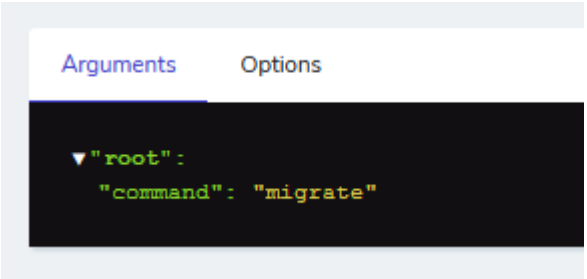


Vraiment pratique tout ça !

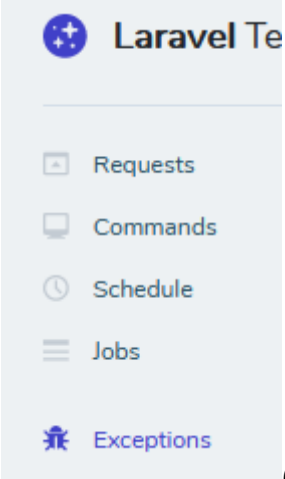
Les commandes



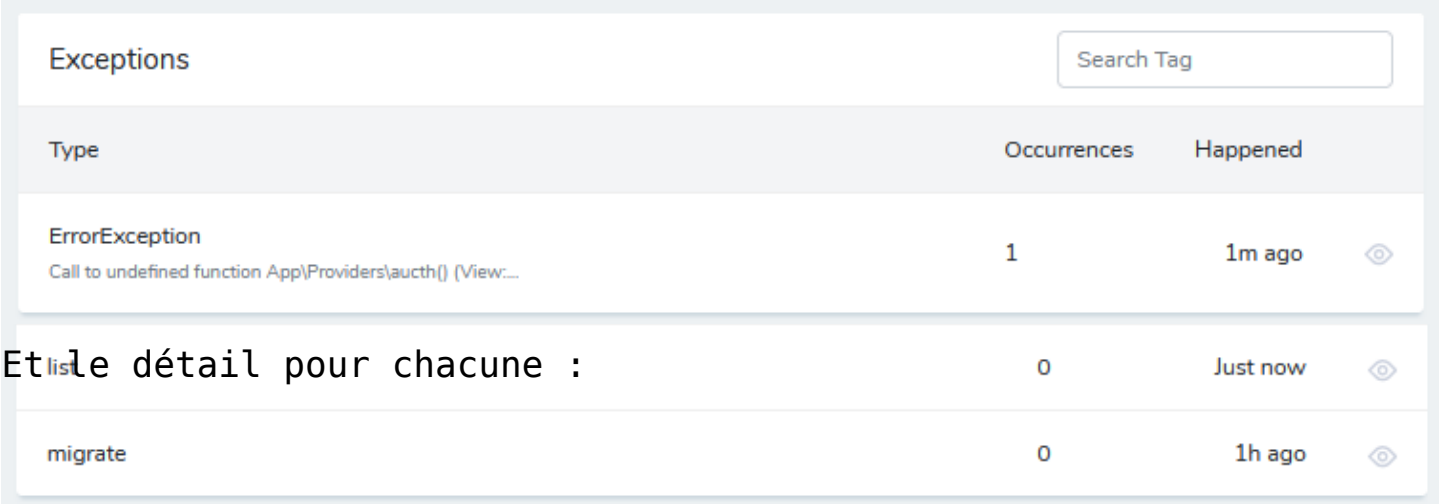
Pour les commande Artisan c’est le même principe : on a la liste des commandes passées et on peut obtenir le détail :



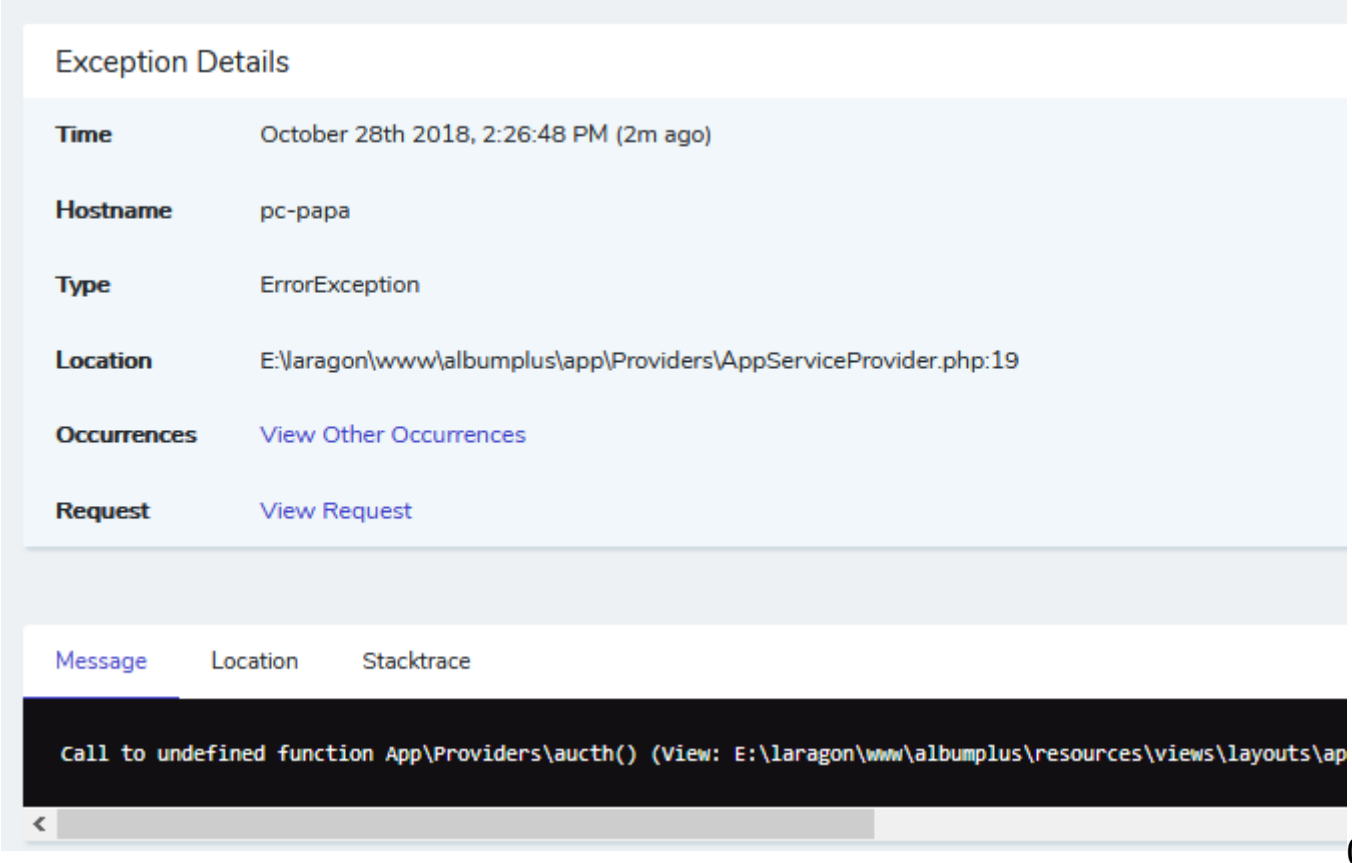
Les exceptions



On a la liste des exceptions :

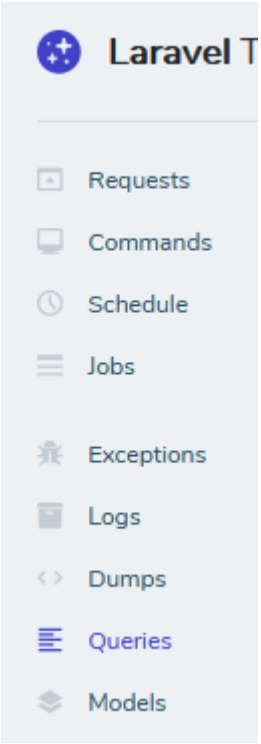


Et le détail pour chacune :

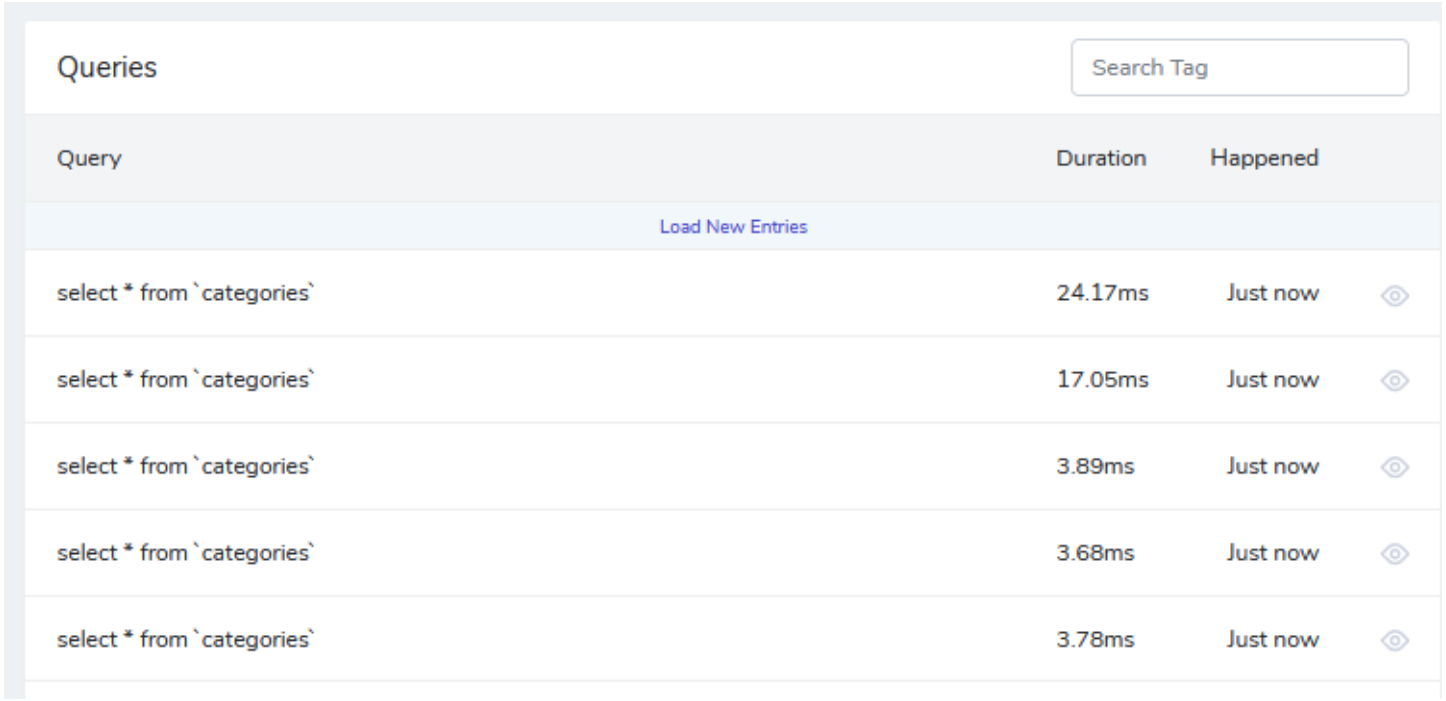


On a un lien pour accéder à la requête concernée et aux autres occurrences.

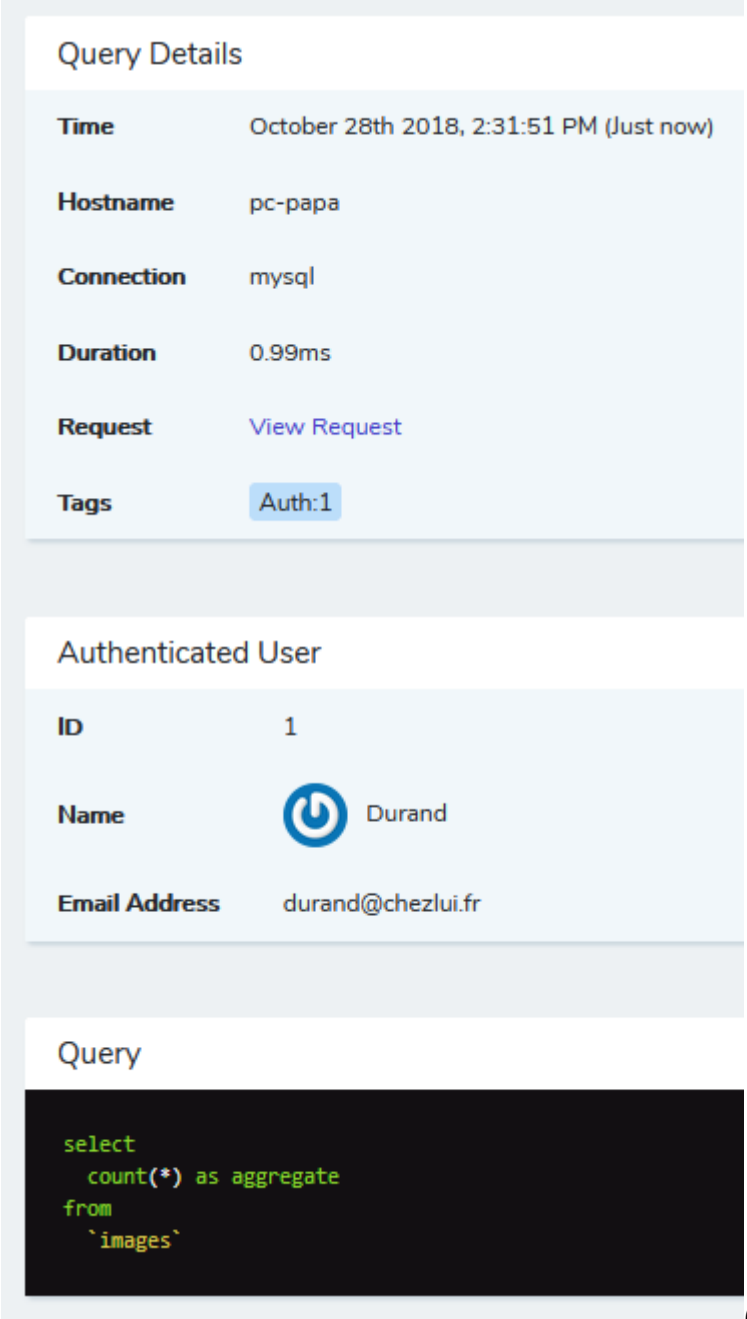
Les requêtes à la base



Là aussi on a la liste avec un formulaire de recherche :

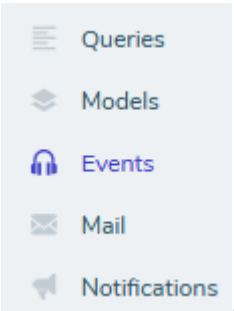


Et pour chaque requête le détail :

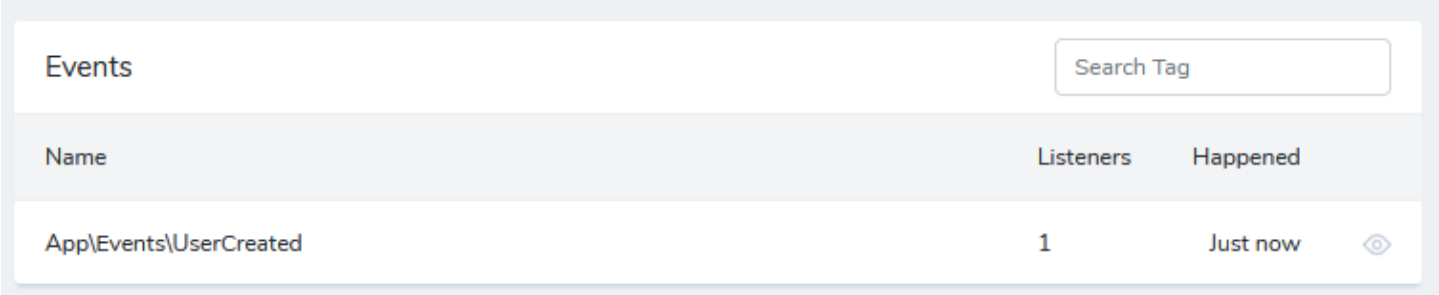


On dispose de l'utilisateur connecté et du détail de la requête.

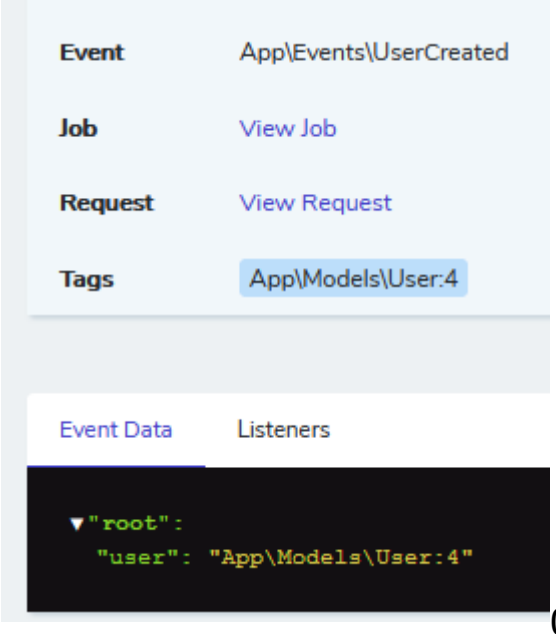
Les événements



On a aussi la liste :

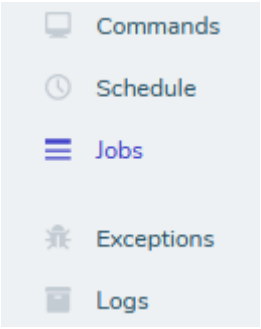


Et le détail :

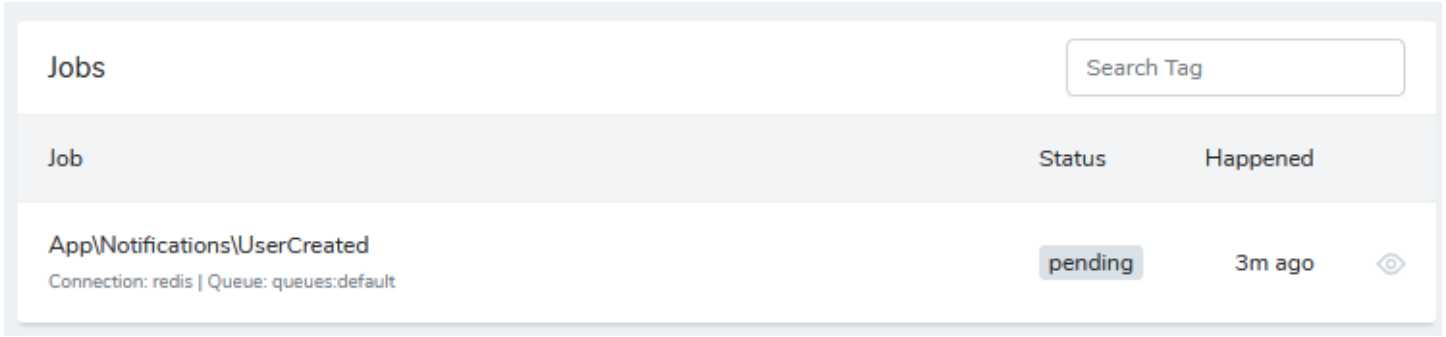


On a un lien pour accéder au job associé (ici pour l'envoi du mail de confirmation) et à la requête.

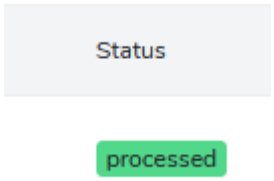
Les jobs



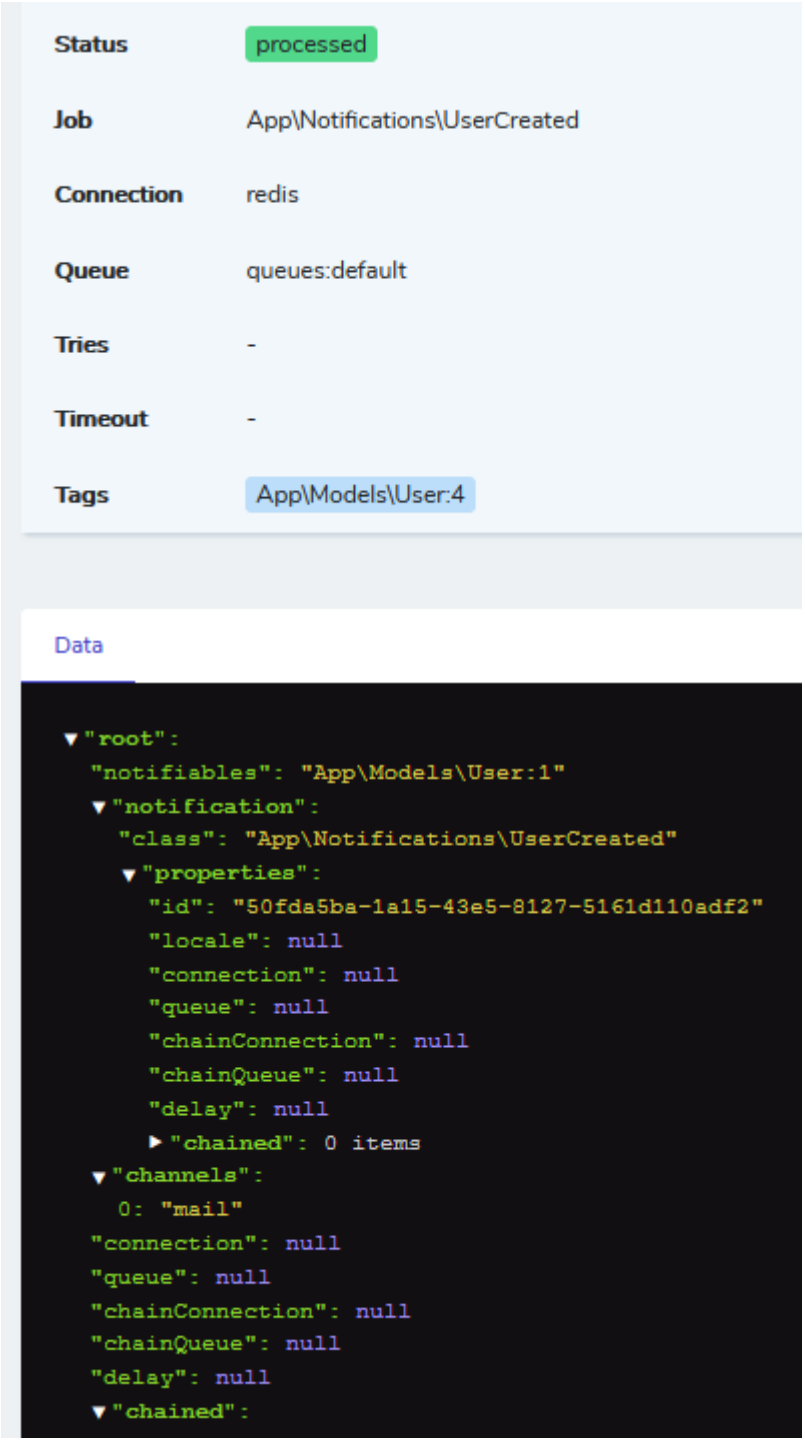
Pour les jobs on a aussi la liste avec le statut pour chacun :



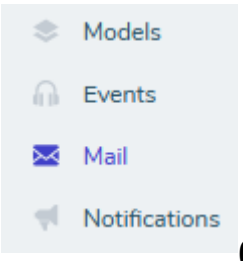
Là on voit que c’est en attente, après exécution on a cet aspect :



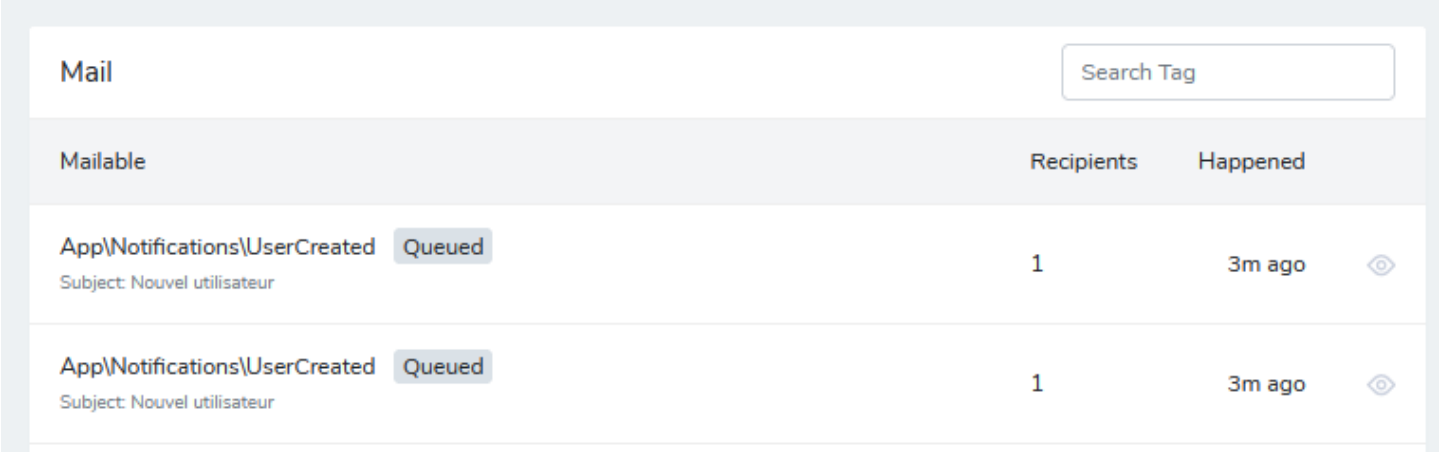
On peut aussi accéder au détail :



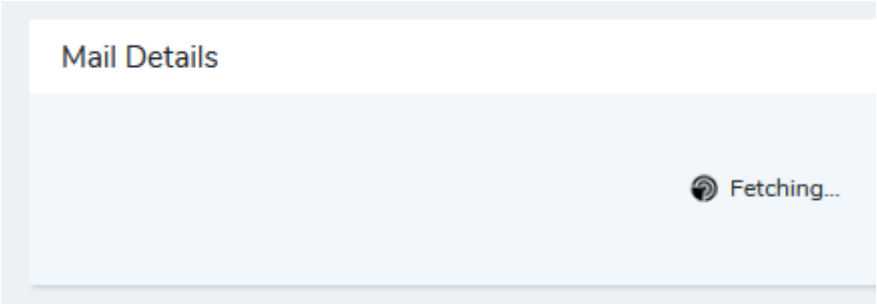
Les mails



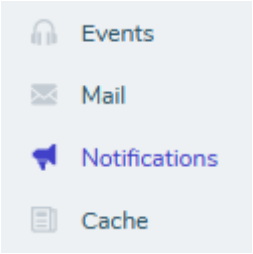
On a la liste des mails envoyés :



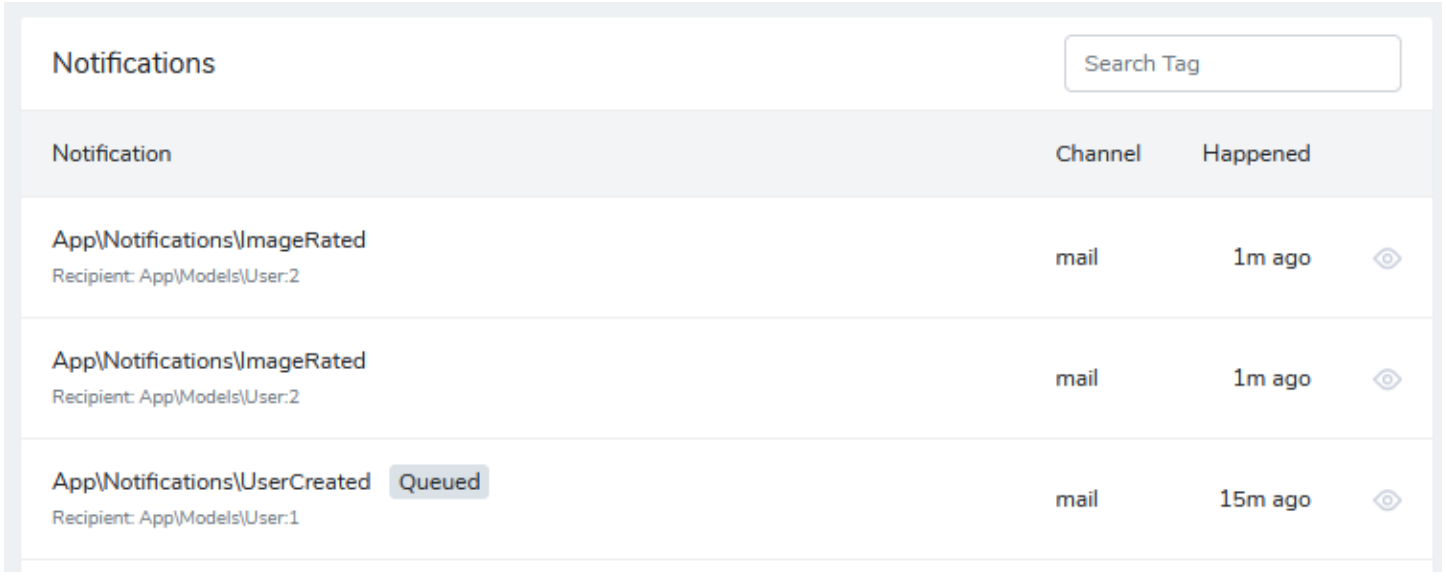
Il est précisé s’ils sont dans une file d’attente. Par contre je n’ai pas réussi à accéder au détail des mails… On verra dans la prochaine version… Je suis resté coincé ici :



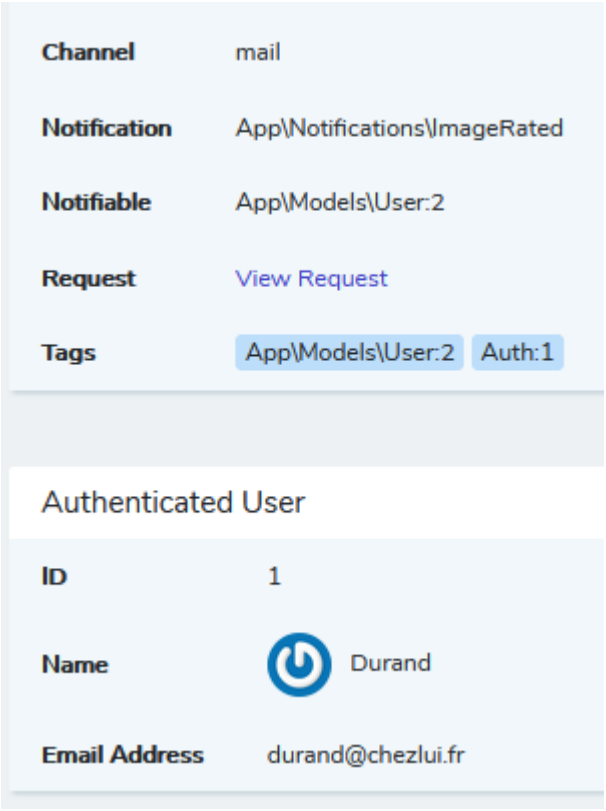
Les notifications



On a la liste des notifications :



On peut accéder aux détails :



Conclusion

Je n’ai pas passé en revue toutes les possibilités, je vous laisse les découvrir !