**Les files d’attentes**

Nous pouvons utiliser **php artisan queue:listen** à la place de **queue:work.**

On peut voir toutes les erreurs sur des tâches en consultant notre base de données ou en affichant cela dans notre terminal par la commande:

**php artisan queue:failed**

De ce fait, nous pouvons par exemple choisir d’afficher toutes les tâches non exécutées dans notre **dashboard.** Nous pourrons également relancer ces tâches en faisant un coup de

**php artisan queue:retry all** (où nous pouvons remplacer **all** par soit l’**id** de la tâche à relancer, encore préciser la tâche qu’on aimerait relancer.

**php artisan queue:flush** pour effacer toutes les tâches qui ont échoués

**php artisan queue:forget id** qui nous permettra d’oublier une tâche qui a échoué

**!!** Si nous sommes intéressés par un système d’interface qui nous permet de voir tout ce qui concerne les tâches et les jobs de notre application, nous pouvons utiliser **Laravel Horizon** mais à noter que cela n’est implémenté que pour le type de connexion **régis**.

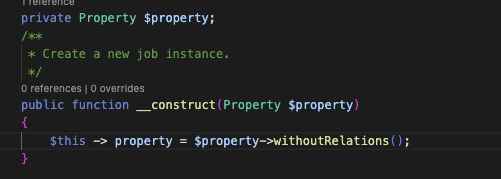
**Parlons un peu des jobs**

Si en exécutant notre tâches on rajoute un délai et qu’il se trouve que le bien a été supprimé dans ce délai, cela peut nous générer une erreur.

Pour gérer cela, nous pouvons ajouter une propriété spéciale dans notre class **Job** qui va indiquer que si le model est supprimé alors la tâche va automatiquement être supprimé:

**public $deleteWhenMissingModels = true;**

**Autre cas:**

Si nous avons des tâches que nous devons exécuter en tâche de fond, nous ne devons pas sérialiser tout notre model **Property** dans le constructeur de notre **job**. Nous pouvons juste récupérer **l’id** du bien en y retirant toute relation de ce fait nous récupérons un model plus léger mais aussi dans le cas où notre model a été modifié cela nous facilite la tâche et du coup moins d’informations à sauvegarder dans la base de données.

Si lorsque nous faisons les traitement nous voulons récupérer les dernières informations du bien au cas où cela serait modifié nous pouvons rajouter dans notre méthode **handle():**

***info:*** *si nous avons plusieurs transactions* ***sql*** *qui doivent s’exécuter avant le déclenchement de nos tâches, il serait très important dans notre fichier de configuration* ***queue.php*** *de changer la clé ‘****after\_commit’*** *à* ***true.*** *De ce fait, il attendra que toutes les transactions soient effectuées avant d’exécuter les tâches, car dans certains cas il se pourrait que l’on travaille avec des données erronées vu que les tâches se sont exécutées avant que les opérations soient terminées.*