

Домашняя работа
по предмету:

Продвинутое программирование на PHP — Laravel

Выполнил: Байборodin Александр
Урок 10. Встроенные возможности Laravel

Цели практической работы:

Научиться:

- создавать асинхронные задачи и вызывать их;
- настраивать очередь через базу данных и добавлять в неё задачи;
- выполнять задачи через планировщик задач Laravel.

В рамках практической работы вы реализуете очистку файла логирования приложения с помощью асинхронной задачи, помещенной в планировщик задач.

Что нужно сделать:

1. Создайте новый проект Laravel или откройте уже существующий.
2. Создайте новую ветку вашего репозитория от корневой (main или master).
3. Создайте миграцию для очереди через базу данных командой `php artisan queue:table`.
4. Выполните миграцию.
5. Пропишите в файле `.env` `QUEUE_CONNECTION=database`.
6. Создайте класс `ClearCache.php` с помощью команды `php artisan make:job ClearCache`.
7. В файле `ClearCache.php` пропишите код для очистки лог-файла.

```
/**
 * Execute the job.
 *
 * @return void
 */
public function handle()
```

8. Поместите вызов Job в планировщик задач Laravel в файле `app/Console/Kernel.php`.

```
protected function schedule(Schedule $schedule)
{
    $schedule->job( job: ClearCache::class
```


9. Запустите очередь командой `php artisan queue:listen`.

10. Запустите планировщик задач командой `php artisan schedule:work` и не закрывайте терминал.

Решение:

1.

Создадим новый проект Laravel:

 MINGW64:/c:/laravelapp/lesson10

```
A@LAPTOP-RHI13GV6 MINGW64 /c:/laravelapp/lesson10
$ composer create-project --prefer-dist laravel/laravel
Creating a "laravel/laravel" project at "./lesson10"
Cannot use laravel/laravel's latest version v12.0.3 as
```

```
- Installing spatie/laravel-ignition (2.9.1): Extracting archive
64 package suggestions were added by new dependencies, use `composer suggest` to
see details.
```

Generating optimized autoload files

> Illuminate\Foundation\ComposerScripts::postAutoloadDump

> @php artisan package:discover --ansi

INFO Discovering packages.

```
laravel/sail .....
laravel/sanctum .....
laravel/tinker .....
nesbot/carbon .....
nunomaduro/collision .....
nunomaduro/termwind .....
spatie/laravel-ignition .....
```

81 packages you are using are looking for funding.

Use the `composer fund` command to find out more!

> @php artisan vendor:publish --tag=laravel-assets --ansi --force

```
A@LAPTOP-RHI13GV6 MINGW64 /c/laravelapp/lesson10
$ ls
lesson10/
```

```
A@LAPTOP-RHI13GV6 MINGW64 /c/laravelapp/lesson10
$ cd lesson10
```

```
A@LAPTOP-RHI13GV6 MINGW64 /c/laravelapp/lesson10/lesson10
$ ls
```

2. Создадим миграции для очереди через базу данных:

```
A@LAPTOP-RHI13GV6 MINGW64 /c/laravelapp
$ php artisan queue:table
```

3. Выполним миграции:

```
A@LAPTOP-RHI13GV6 MINGW64 /c/laravelapp
$ php artisan queue:table
```

4. Настроим очереди в файле .env:

- Откроем файл .env и установим:

```
QUEUE_CONNECTION=database
```

```
APP_NAME=Laravel
APP_ENV=local
APP_KEY=base64:bx+DC1ckCZKfqekxV17EkzyUw7WExSkZ65EAABpLKLc=
APP_DEBUG=true
APP_URL=http://localhost
```

```
LOG_CHANNEL=stack
LOG_DEPRECATIONS_CHANNEL=null
LOG_LEVEL=debug
```

```
DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=database
DB_USERNAME=root
DB_PASSWORD=
```

```
BROADCAST_DRIVER=log
CACHE_DRIVER=file
FILESYSTEM_DISK=local
QUEUE_CONNECTION=database
SESSION_DRIVER=file
SESSION_LIFETIME=120
```

```
MEMCACHED_HOST=127.0.0.1
```

```
REDIS_HOST=127.0.0.1
REDIS_PASSWORD=null
REDIS_PORT=6379
```

```
MAIL_MAILER=smtp
MAIL_HOST=mailpit
MAIL_PORT=1025
MAIL_USERNAME=null
MAIL_PASSWORD=null
MAIL_ENCRYPTION=null
MAIL_FROM_ADDRESS="hello@example.com"
MAIL_FROM_NAME="${APP_NAME}"

AWS_ACCESS_KEY_ID=
AWS_SECRET_ACCESS_KEY=
AWS_DEFAULT_REGION=us-east-1
AWS_BUCKET=
AWS_USE_PATH_STYLE_ENDPOINT=false

PUSHER_APP_ID=
PUSHER_APP_KEY=
PUSHER_APP_SECRET=
PUSHER_HOST=
PUSHER_PORT=443
PUSHER_SCHEME=https
PUSHER_APP_CLUSTER=mt1

VITE_APP_NAME="${APP_NAME}"
VITE_PUSHER_APP_KEY="${PUSHER_APP_KEY}"
VITE_PUSHER_HOST="${PUSHER_HOST}"
VITE_PUSHER_PORT="${PUSHER_PORT}"
VITE_PUSHER_SCHEME="${PUSHER_SCHEME}"
VITE_PUSHER_APP_CLUSTER="${PUSHER_APP_CLUSTER}"
```

5. Создадим класс ClearCache:

```
A@LAPTOP-RHI13GV6 MINGW64 /c/laravelapp/lesson10/lesson10
$ php artisan make:job ClearCache
```

```
TNEO Job [C:\laravelapp\lesson10\lesson10\app\Jobs\ClearCache.php]
```

6. Реализуем очистку лог-файла в ClearCache.php:

- Откроем файл app/Jobs/ClearCache.php и добавим код для очистки лог-файла:

```
public function handle()
{
    file_put_contents(storage_path('logs/laravel.log'), "");
}
```

7. Добавим задачу в планировщик:

- Откроем файл app/Console/Kernel.php и добавим задачу в метод schedule:

```
protected function schedule(Schedule $schedule)
{
    $schedule->job(new ClearCache)->hourly();
}
```

```
<?php
```

```
namespace App\Jobs;
```

```

use Illuminate\Bus\Queueable;
use Illuminate\Contracts\Queue\ShouldQueue;
use Illuminate\Foundation\Bus\Dispatchable;
use Illuminate\Queue\InteractsWithQueue;
use Illuminate\Queue\SerializesModels;

class ClearCache implements ShouldQueue
{
    use Dispatchable, InteractsWithQueue, Queueable, SerializesModels;

    /**
     * Create a new job instance.
     */
    public function __construct()
    {
        //
    }

    /**
     * Execute the job.
     */
    public function handle()
    {
        file_put_contents(storage_path('logs/laravel.log'), '');
    }
}

```

8. Запустим очередь обработки задач:

- Запустите очередь для обработки задач:

```

A@LAPTOP-RHI13GV6 MINGW64 /c/laravelapp/
$ php artisan queue:listen

```

INFO Processing jobs from the 'default' queue.

9. Запустим планировщик задач:

```

A@LAPTOP-RHI13GV6 MINGW64 /c/laravelapp/
$ php artisan schedule:work

```

INFO Running scheduled tasks every n