САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

Дисциплина: Бэк-энд разработка

Отчет

Лабораторная работа 6

Выполнил:

Сахно Ярослав

К3441

Проверил:
Добряков Д. И.

Санкт-Петербург

2026 г.

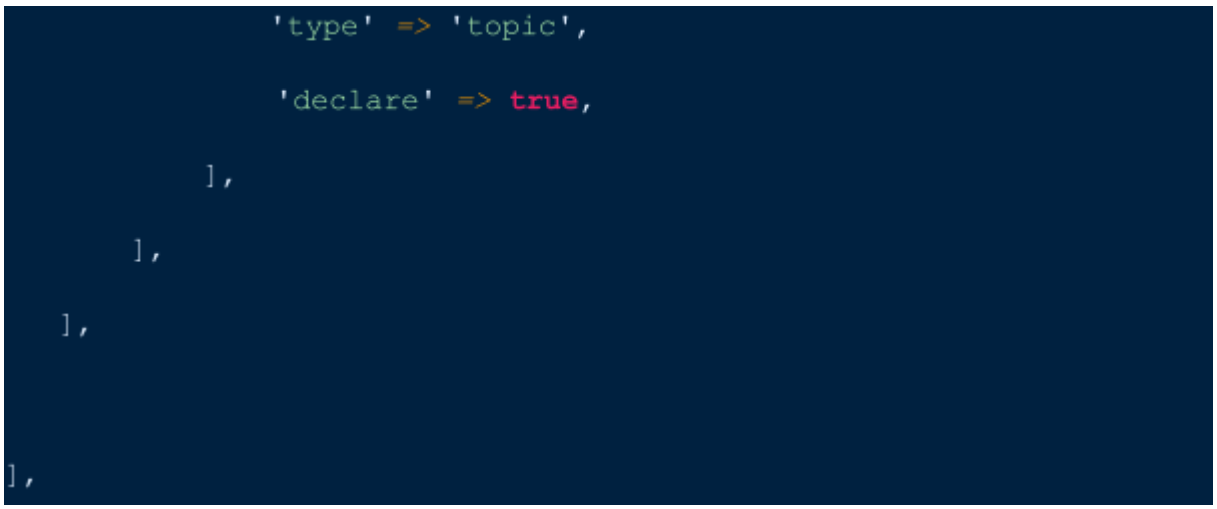
Задача

- подключить и настроить rabbitMQ/kafka;
- реализовать межсервисное взаимодействие посредством rabbitMQ/kafka.

Ход работы

Установил пакет vladimir-yuldashev/laravel-queue-rabbitmq и добавил настройки (см. Листинг 1).

```
    // ... redis, sync, database ...

    'rabbitmq' => [

        'driver'           => 'rabbitmq',

        'queue'            => env('RABBITMQ_QUEUE', 'default'),

        'connection'       =>
PhpAmqpLib\Connection\AMQPLazyConnection::class,


        'hosts' => [

            [

                'host'       => env('RABBITMQ_HOST', '127.0.0.1'),

                'port'       => env('RABBITMQ_PORT', 5672),

                'user'       => env('RABBITMQ_USER', 'guest'),

                'password'   => env('RABBITMQ_PASSWORD', 'guest'),

                'vhost'      => env('RABBITMQ_VHOST', '/'),

            ],

        ],


        'options' => [

            'exchange' => [

                'name' => 'hwsys.events',    // единая topic-биржа
```



```
                'type' => 'topic',
                'declare' => true,
            ],
        ],
    ],

],
```

Листинг 1 - config/queue.php

Пример публикации события Auth-service (см. Листинг 2).

```
namespace App\Jobs;

use App\Models\User;
use Illuminate\Contracts\Queue\ShouldQueue;
use Illuminate\Queue\InteractsWithQueue;

class PublishUserRegistered implements ShouldQueue
{
    use InteractsWithQueue;

    public $queue = 'user.registered'; // routing-key

    public function __construct(
        public readonly User $user
    ) {}

    public function tags(): array
```

```
    {
        return ['event', 'user.registered'];
    }

    public function handle(): void
    {
        // сам факт помещения Job в очередь отправит payload в
RabbitMQ.
    }

    public function middleware(): array
    {
        return [];
    }

    public function payload(): array
    {
        return [
            'id'    => $this->user->id,
            'email' => $this->user->email,
            'name' => $this->user->name,
            'role' => $this->user->role,
            'at'    =>
            now()->toISOString(),
        ];
    }
```

Листинг 2 - app/Jobs/PublishUserRegistered.php

Пример джоб консьюмера Auth-service (см. Листинг 3).

```php
namespace App\Jobs;

use App\Models\ExternalUser;
use Illuminate\Contracts\Queue\ShouldQueue;

class SyncUserFromAuth implements ShouldQueue
{
    public $queue = 'user.registered.course';

    public function __construct(
        public readonly array $payload
    ) {}

    public function handle(): void
    {
        ExternalUser::updateOrCreate(
            ['id' =>
            $this->payload['id']], [
                'email' => $this->payload['email'],
                'name' => $this->payload['name'],
                'role' => $this->payload['role'],
            ]
        );
    }
```

Листинг 3 - app/Jobs/SyncUserFromAuth.php

Вывод

В микросервисы были внедрена очередь RabbitMQ, для обмена данными между собой.