САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

Дисциплина: Бэк-энд разработка

Отчет

Лабораторная работа #6

Выполнил:
Ребров Сергей

Группа:
K3439

Проверил:
Добряков Д. И.

Санкт-Петербург

2026 г.

Задача

● подключить и настроить rabbitMQ/kafka;

● реализовать межсервисное взаимодействие посредством rabbitMQ/kafka.

Ход работы

В ходе выполнения работы был установлен пакет rabbitmq и добавлены настройки.

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

Пример:

Auth-service осуществляет публикацию события UserRegistered, которое содержит идентификатор пользователя и основные регистрационные данные и предназначено для информирования других компонентов системы о факте успешного создания учетной записи.

```
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',
	public function __construct(
		public readonly User $user
	) {}
	public function tags(): array
	{
		return ['event', 'user.registered'];
	}
	public function handle(): void
	{


	}
	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(),
            ];
        }
}
```

Вывод

В результате выполнения работы было настроено взаимодействие микросервисов с использованием брокера сообщений RabbitMQ и реализован механизм публикации событий, обеспечивающий асинхронный обмен данными между сервисами системы.