

**САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ  
ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО**

**Дисциплина:** Бэк-энд разработка

**Отчет**

**Лабораторная работа №4**

Выполнил:  
Жаров Александр  
К33402

Проверил:  
Добряков Д. И.

Санкт-Петербург

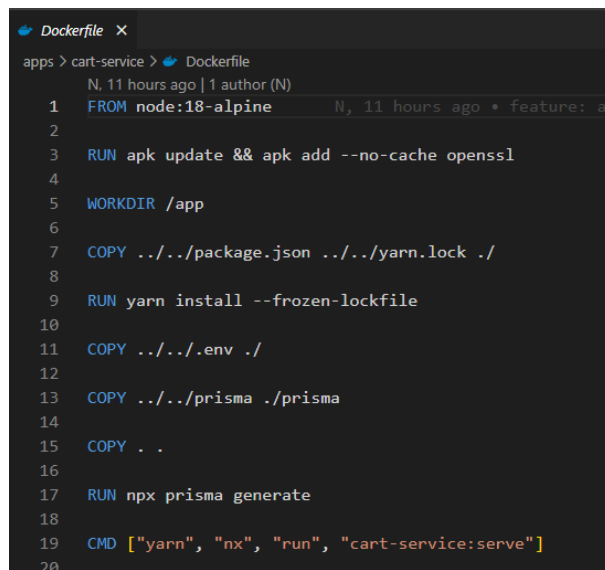
2024 г.

## Задача

Необходимо упаковать ваше приложение в docker-контейнеры и обеспечить сетевое взаимодействие между различными частями вашего приложения, а также настроить общение микросервисов между собой посредством RabbitMQ. Делать это можно как с помощью docker-compose так и с помощью docker swarm. При разумном использовании swirl вы получите дополнительные баллы.

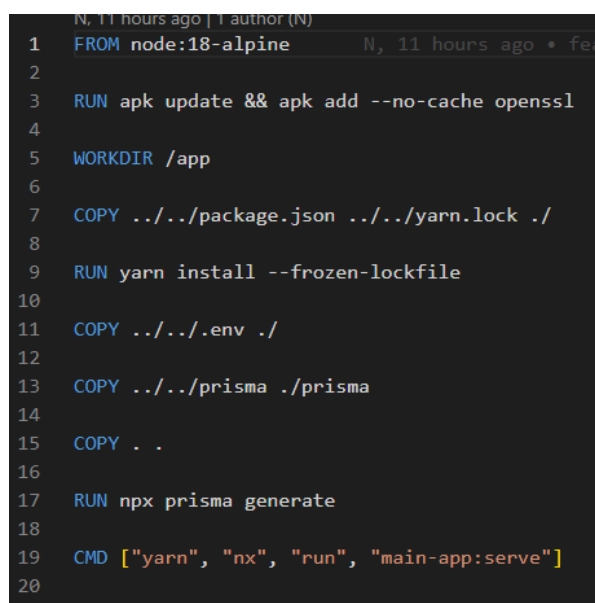
## Ход работы

1. Для начала устанавливаем Docker
2. Затем был созданы DockerFile для каждого микросервиса



```
apps > cart-service > Dockerfile
N, 11 hours ago | 1 author (N)
1 FROM node:18-alpine N, 11 hours ago * feature: ad
2
3 RUN apk update && apk add --no-cache openssl
4
5 WORKDIR /app
6
7 COPY ../../package.json ../../yarn.lock ./
8
9 RUN yarn install --frozen-lockfile
10
11 COPY ../../.env ./
12
13 COPY ../../prisma ./prisma
14
15 COPY . .
16
17 RUN npx prisma generate
18
19 CMD ["yarn", "nx", "run", "cart-service:serve"]
20
```

Рисунок 1 – DockerFile микросервиса



```
N, 11 hours ago | 1 author (N)
1 FROM node:18-alpine N, 11 hours ago * fea
2
3 RUN apk update && apk add --no-cache openssl
4
5 WORKDIR /app
6
7 COPY ../../package.json ../../yarn.lock ./
8
9 RUN yarn install --frozen-lockfile
10
11 COPY ../../.env ./
12
13 COPY ../../prisma ./prisma
14
15 COPY . .
16
17 RUN npx prisma generate
18
19 CMD ["yarn", "nx", "run", "main-app:serve"]
20
```

Рисунок 2 – DockerFile основного сервиса

3. Следующим шагом в корневой директории был создан docker-compose файл, в котором мы описали как, где и что запускать

```
1 version: '3.8'
2 services:
3   cart-service:
4     build:
5       context: .
6       dockerfile: apps/cart-service/Dockerfile
7     ports:
8       - '3001:3000'
9     networks:
10      - postgres
11
12   main-app:
13     build:
14       context: .
15       dockerfile: apps/main-app/Dockerfile
16     ports:
17       - '127.0.0.1:8000:8000'
18     networks:
19       - postgres
20     command: sh -c "yarn prisma migrate deploy && yarn nx run main-app:serve"
21
22   database:
23     image: postgres:16-alpine
24     environment:
25       - POSTGRES_USER=postgres
26       - POSTGRES_PASSWORD=123456789
27     ports:
28       - '9000:5432'
29     volumes:
30       - pgdata:/var/lib/postgresql/data
31     networks:
32       - postgres
33
34   prisma-studio:
35     container_name: prisma-studio
36     image: timothyjmiller/prisma-studio:latest
37     restart: unless-stopped
38     env_file:
39       - .env
40     ports:
41       - 5555:5555
42     networks:
43       - postgres
44
45   rabbitmq:
46     image: rabbitmq:3-management
47     ports:
48       - '5672:5672'
49       - '15672:15672'
50     networks:
51       - postgres
52
53 volumes:
54   pgdata:
55
56 networks:
57   postgres:
58     driver: bridge
```

Рисунок 3 – docker-compose

Для каждого микросервиса мы задаем контекст, путь к докерфайлу из которого будет создаваться образ, а также сеть в которой будет запущена программа. Также мы прописываем правила запуска для prisma и базы данных на postgres используя скаченные образы из интернета.

## 4. Запускаем приложение и проверяем работоспособность

```
main-app-1 | [Nest] 214 - 05/29/2024, 9:25:11 PM LOG [InstanceLoader] UserModule dependencies initialized +1ms
main-app-1 | [Nest] 214 - 05/29/2024, 9:25:11 PM LOG [InstanceLoader] AuthModule dependencies initialized +1ms
main-app-1 | [Nest] 214 - 05/29/2024, 9:25:11 PM LOG [RoutesResolver] AppController (/api): +139ms
main-app-1 | [Nest] 214 - 05/29/2024, 9:25:11 PM LOG [RoutesResolver] UserController (/api/user): +1ms
main-app-1 | [Nest] 214 - 05/29/2024, 9:25:11 PM LOG [RouterExplorer] Mapped (/api/user/me, GET) route +7ms
main-app-1 | [Nest] 214 - 05/29/2024, 9:25:11 PM LOG [RoutesResolver] AuthController (/api/auth): +1ms
main-app-1 | [Nest] 214 - 05/29/2024, 9:25:11 PM LOG [RouterExplorer] Mapped (/api/auth/login, POST) route +1ms
main-app-1 | [Nest] 214 - 05/29/2024, 9:25:11 PM LOG [RouterExplorer] Mapped (/api/auth/logout, GET) route +1ms
main-app-1 | [Nest] 214 - 05/29/2024, 9:25:11 PM LOG [RouterExplorer] Mapped (/api/auth/register, POST) route +1ms
main-app-1 | [Nest] 214 - 05/29/2024, 9:25:11 PM LOG [RouterExplorer] Mapped (/api/auth/refresh, GET) route +1ms
main-app-1 | [Nest] 214 - 05/29/2024, 9:25:11 PM LOG [RoutesResolver] CartController (/api/cart): +1ms
main-app-1 | [Nest] 214 - 05/29/2024, 9:25:11 PM LOG [RouterExplorer] Mapped (/api/cart/my-cart, GET) route +1ms
main-app-1 | [Nest] 214 - 05/29/2024, 9:25:11 PM LOG [RouterExplorer] Mapped (/api/cart/add-product, POST) route +1ms
main-app-1 | [Nest] 214 - 05/29/2024, 9:25:11 PM LOG [RouterExplorer] Mapped (/api/cart/remove-product/:productId, DELETE) route +1ms
main-app-1 | [Nest] 214 - 05/29/2024, 9:25:11 PM LOG [RouterExplorer] Mapped (/api/cart/update-product/:productId, PATCH) route +2ms
main-app-1 | [Nest] 214 - 05/29/2024, 9:25:11 PM LOG [RoutesResolver] ProductController (/api/products): +0ms
main-app-1 | [Nest] 214 - 05/29/2024, 9:25:11 PM LOG [RouterExplorer] Mapped (/api/products, GET) route +1ms
main-app-1 | [Nest] 214 - 05/29/2024, 9:25:11 PM LOG [RouterExplorer] Mapped (/api/products/:id, GET) route +1ms
main-app-1 | [Nest] 214 - 05/29/2024, 9:25:11 PM LOG [RouterExplorer] Mapped (/api/products/products-with-discounts/:discountId, GET) route +1ms
main-app-1 | [Nest] 214 - 05/29/2024, 9:25:11 PM LOG [RoutesResolver] DiscountController (/api/discounts): +0ms
main-app-1 | [Nest] 214 - 05/29/2024, 9:25:11 PM LOG [RouterExplorer] Mapped (/api/discounts, GET) route +1ms
main-app-1 | [Nest] 214 - 05/29/2024, 9:25:11 PM LOG [RoutesResolver] CategoryController (/api/category): +0ms
main-app-1 | [Nest] 214 - 05/29/2024, 9:25:11 PM LOG [RouterExplorer] Mapped (/api/category, GET) route +1ms
main-app-1 | [Nest] 214 - 05/29/2024, 9:25:12 PM LOG [NestApplication] Nest application successfully started +10ms
main-app-1 | [Nest] 214 - 05/29/2024, 9:25:12 PM LOG # Application is running on: http://localhost:8000/api
rabbitmq-1 | 2024-05-29 21:25:15.255093+00:00 [info] <0.594.0> Management plugin: HTTP (non-TLS) listener started on port 15672
rabbitmq-1 | 2024-05-29 21:25:15.255513+00:00 [info] <0.625.0> Statistics database started.
rabbitmq-1 | 2024-05-29 21:25:15.295029+00:00 [info] <0.624.0> Starting worker pool 'management_worker_pool' with 3 processes in it
rabbitmq-1 | 2024-05-29 21:25:15.296043+00:00 [info] <0.643.0> Prometheus metrics: HTTP (non-TLS) listener started on port 15692
rabbitmq-1 | 2024-05-29 21:25:15.301953+00:00 [info] <0.534.0> Ready to start client connection listeners
rabbitmq-1 | 2024-05-29 21:25:15.377708+00:00 [info] <0.687.0> started TCP listener on [::]:5672
rabbitmq-1 | 2024-05-29 21:25:15.377708+00:00 [info] <0.690.0> accepting AMQP connection <0.690.0> (172.18.0.4:59926 -> 172.18.0.3:5672)
rabbitmq-1 | 2024-05-29 21:25:15.459873+00:00 [info] <0.690.0> connection <0.690.0> (172.18.0.4:59926 -> 172.18.0.3:5672): user 'guest' authenticated and gra
rted access to vhost '/'
rabbitmq-1 | 2024-05-29 21:25:15.479376+00:00 [warning] <0.780.0> Deprecated features: 'transient_nexxcl_queues': Feature 'transient_nexxcl_queues' is depr
ecated.
rabbitmq-1 | 2024-05-29 21:25:15.479376+00:00 [warning] <0.780.0> By default, this feature can still be used for now.
rabbitmq-1 | 2024-05-29 21:25:15.479376+00:00 [warning] <0.780.0> Its use will not be permitted by default in a future minor RabbitMQ version and the feature
will be removed
rabbitmq-1 | 2024-05-29 21:25:15.479376+00:00 [warning] <0.780.0> To continue using this feature when it is not permitted by default, set the following param
eter in your configuration:
```

Рисунок 4 – Успешный запуск в консоли

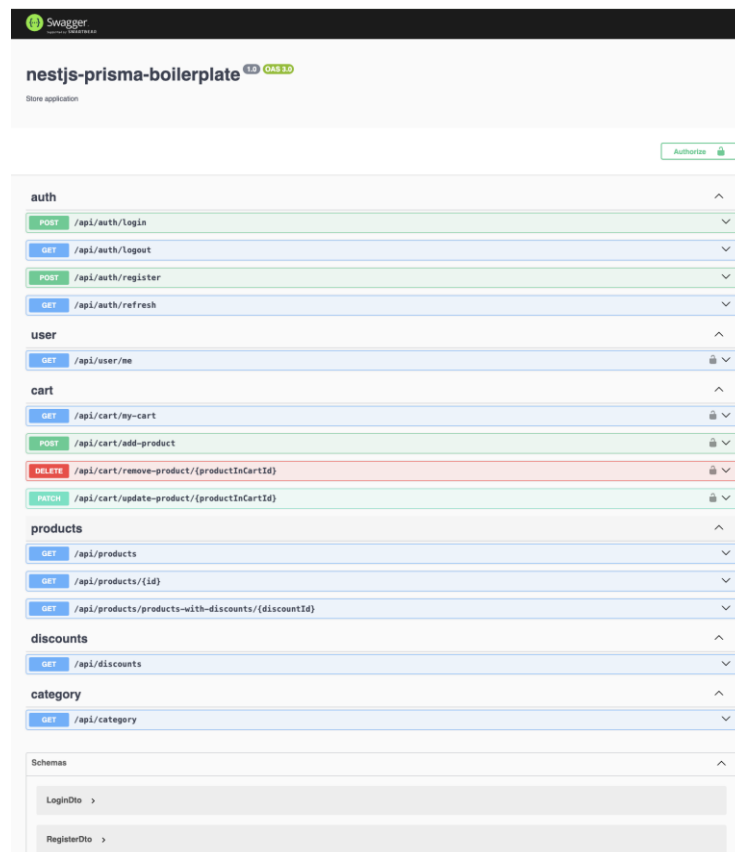


Рисунок 5 – проверка работоспособности

## Вывод

В ходе работы я научился упаковывать приложение в docker-контейнеры, запускать из контейнера сразу несколько микросервисов, а также

синхронизировать работу микросервисов между собой по средствам Rabbitmq.