

**САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ
ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО**

Дисциплина: Бэк-энд разработка

Отчет

Лабораторная работа №3

Выполнил:
Жаров Александр
К33402

Проверил:
Добряков Д. И.

Санкт-Петербург

2024 г.

Задача

Необходимо реализовать отдельный микросервис, выполняющий какую-либо содержательную функцию из всего арсенала функций вашего приложения.

Ход работы

1. Для вынесения в микросервис мы выбрали модель cart. Мы изменили структуру проекта, выделив отдельную директорию apps в которой и будут находиться наши микросервисы, поскольку мы используем структуру монорепозиторий, было решено использовать их.

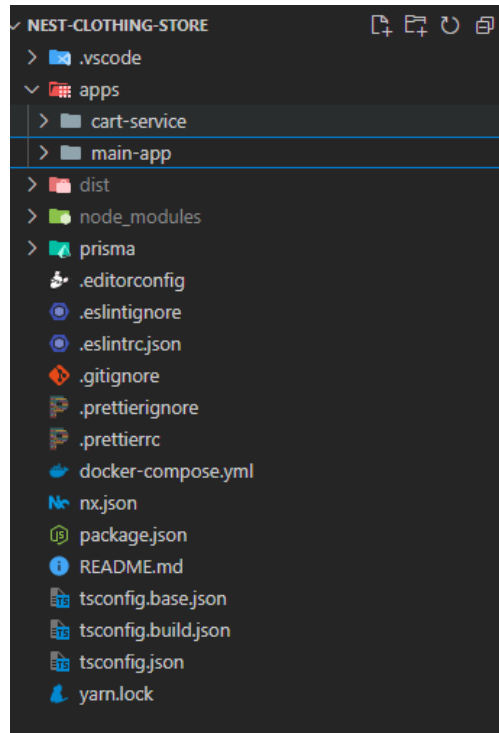


Рисунок 1 – Структура проекта

2. Для каждого микросервиса был написан project.json, описывающий сборку проекта

```

"name": "cart-service",
"$schema": "../../node_modules/nx/schemas/project-schema.json",
"sourceRoot": "apps/cart-service/src",
"projectType": "application",
"tags": [],
"targets": {
  "build": {
    "executor": "@nx/webpack:webpack",
    "outputs": ["{options.outputPath}"],
    "defaultConfiguration": "production",
    "options": {
      "target": "node",
      "compiler": "tsc",
      "outputPath": "dist/apps/cart-service",
      "main": "apps/cart-service/src/main.ts",
      "tsConfig": "apps/cart-service/tsconfig.app.json",
      "webpackConfig": "apps/cart-service/webpack.config.ts"
    },
    "configurations": {
      "development": {},
      "production": {
        "optimization": true,
        "inspect": false
      }
    }
  },
  "serve": {
    "executor": "@nx/js:node",
    "defaultConfiguration": "development",
    "options": {
      "buildTarget": "cart-service:build"
    },
    "configurations": {
      "development": {
        "buildTarget": "cart-service:build:development"
      },
      "production": {
        "buildTarget": "cart-service:build:production"
      }
    }
  },
  "docker-build": {
    "executor": "nx:run-commands",
    "options": {
      "command": "docker build -t cart-service:latest -f apps/cart-service/Dockerfile ."
    }
  }
}

```

Рисунок 2 – project.json

3. Т.к теперь всей логике корзины унесена в отдельный сервис, в основном сервисе мы получаем данные от удаленного микросервиса.

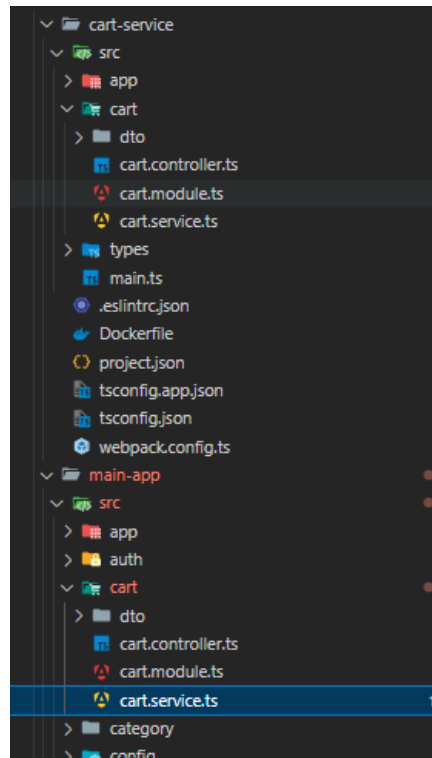


Рисунок 3 – модуль корзины

```
import { Injectable, Inject } from '@nestjs/common';
import { ClientProxy } from '@nestjs/microservices';
import { ProductInCartCreateDto, ProductInCartUpdateDto } from '../dto/productInCart.dto';

N, last week | 1 author (N)
@Injectable()
export class CartService {
  constructor(@Inject('CART_SERVICE') private readonly client: ClientProxy) {}

  getUserCart(userId: number) {
    return this.client.send({ cmd: 'getUserCart' }, { userId });
  }

  addProductToCart(userId: number, productDto: ProductInCartCreateDto) {
    return this.client.send({ cmd: 'addProductToCart' }, { userId, productDto });
  }

  removeProductFromCart(productInCartId: number) {
    return this.client.send({ cmd: 'removeProductFromCart' }, { productInCartId });
  }

  updateProductInCart(productInCartId: number, productDto: ProductInCartUpdateDto) {
    return this.client.send({ cmd: 'updateProductInCart' }, { productInCartId, productDto });
  }

  createCart(userId: number) {
    return this.client.send({ cmd: 'createCart' }, { userId });
  }
}
```

Рисунок 4 – CartService в главном сервисе

4. Для запуска проекта теперь используются следующие команды

```
{
  "name": "@nest-clothing-store/source",
  "version": "0.0.0",
  "license": "MIT",
  > Debug
  "scripts": {
    "main-app:serve": "yarn nx run main-app:serve",
    "cart-service:serve": "yarn nx run cart-service:serve",
    "dev": "yarn nx run-many -t serve -p cart-service main-app --output-style stream"
  },
  "private": true,
}
```

Рисунок 5 – скрипты запуска

Вывод

В ходе работы я изучил микросервисную архитектуру и научился выносить функционал приложения на nestjs в отдельный микросервис.