

**САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ
ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО**

Дисциплина: Бэк-энд разработка

Отчет

Лабораторная работа

Выполнила:

Саунин Антон

Группа

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. Делать это можно как с помощью docker-compose так и с помощью docker swarm. При разумном использовании swirl вы получите дополнительные баллы.

Ход работы

В приложении были выделены два микросервиса:

- Авторизация
- Маркет

Для того, чтобы передавать id пользователя из сервиса авторизации в маркет, был написан соответствующий роут:

```
router.get("/getUser", (req: any, res: any) => {  
  res.setHeader("User-Id", req.user.id);  
  res.sendStatus(200);  
});
```

Создадим Dockerfile в микросервисах:

```
Dockerfile X  
auth > Dockerfile > FROM  
1 FROM node:20  
2  
3 WORKDIR /back/auth  
4  
5  
6 COPY package*.json ./  
7 RUN npm install  
8  
9 COPY . .  
10  
11 EXPOSE 5000  
12  
13 CMD ["npm", "run", "dev"]
```

```
Dockerfile X  
market > Dockerfile > FROM  
1 FROM node:20  
2  
3 WORKDIR /back/market  
4  
5  
6 COPY package*.json ./  
7 RUN npm install  
8  
9 COPY . .  
10  
11 EXPOSE 8000  
12  
13 CMD ["npm", "run", "dev"]
```

Gateway было реализовано при помощи nginx:

```
Dockerfile X
gateway > Dockerfile > FROM
1 FROM nginx:latest
2
3 COPY config/gateway.conf /etc/nginx/nginx.conf
4
5 CMD ["nginx", "-g", "daemon off;"]
```

Config:

```
gateway.conf X
gateway > config > gateway.conf
1 worker_processes auto;
2
3 events {
4     worker_connections 1024;
5 }
6
7 http {
8     server {
9         listen 8080;
10
11         location /getUser {
12             internal;
13             proxy_pass http://service-auth:5000/users/getUser;
14             proxy_redirect off;
15             proxy_set_header Host $host;
16             proxy_set_header X-Real-IP $remote_addr;
17             proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
18             proxy_set_header X-Forwarded-Host $server_name;
19             proxy_set_header Content-Length "";
20             proxy_pass_request_body off;
21         }
22
23         location /auth {
24             rewrite ^/auth/(.*) /$1 break;
25
26             proxy_pass http://service-auth:5000;
27             proxy_redirect off;
28             proxy_set_header Host $host;
29             proxy_set_header X-Real-IP $remote_addr;
30             proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
31             proxy_set_header X-Forwarded-Host $server_name;
32         }
33
34         location /market {
35             auth_request /getUser;
36             auth_request_set $user_id $sent_http_user_id;
37
38             rewrite ^/market/(.*) /$1 break;
39
40             proxy_pass http://service-market:8000;
41             proxy_redirect off;
42             proxy_set_header Host $host;
43             proxy_set_header X-Real-IP $remote_addr;
44             proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
45             proxy_set_header X-Forwarded-Host $server_name;
46             proxy_set_header User-Id $user_id;
47         }
48     }
49 }
```

Далее был написан docker-compose:

Gateway:

```
services:
  gateway:
    container_name: gateway
    build:
      context: ./gateway
    restart: always
    ports:
      - 8080:8080
    depends_on:
      - service-auth
      - service-market
      - service-postgres
    networks:
      - app-network
```

Авторизация:

```
service-auth:
  container_name: service-auth
  build:
    context: ./auth
  restart: always
  expose:
    - 5000
  env_file:
    - ./auth/.env
  networks:
    - app-network
  depends_on:
    - service-postgres
```

Маркет:

```
service-market:
  container_name: service-market
  build:
    context: ./market
  restart: always
  expose:
    - 8000
  env_file:
    - ./market/.env
  networks:
    - app-network
  depends_on:
    - service-postgres
```

База данных:

```
service-postgres:
  container_name: service-postgres
  image: postgres:14
  environment:
    POSTGRES_DB: testing
    POSTGRES_USER: postgres
    POSTGRES_PASSWORD: 1122

  expose:
    - 5432:5432
  volumes:
    - postgres_data:/var/lib/postgresql/data

  networks:
    - app-network
```

Проверим запуск:

```
PS K:\6 сем\бэк\итоговые лабы\lab4> docker-compose up
[+] Running 4/0
✓ Container service-postgres Created
✓ Container service-market Created
✓ Container service-auth Created
✓ Container gateway Created
```

Проверим запросы на разные микросервисы:

POST http://localhost:8080/auth/register/ Send

Params Authorization Headers (9) Body Scripts Settings Cookies

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary ☐ GraphQL JSON Beautify

```
1 {
2   "email": "saunin@gmail.com",
3   "password": "12345678"
4 }
```

Body Cookies Headers (8) Test Results 201 Created • 592 ms • 536 B

Pretty Raw Preview Visualize JSON

```
1 {
2   "id": 4,
3   "email": "saunin@gmail.com",
4   "password": "$argon2id$v=19$m=65536,t=3,p=4$2L5cSe5ZmWo+CYGjU/v3jAS07hXox0y70u00FKN2Ryt/
   cFXEtPLWt6UBkQECMH1BM",
5   "updatedAt": "2024-09-17T22:22:48.756Z",
6   "createdAt": "2024-09-17T22:22:48.756Z",
7   "name": null,
8   "phone": null,
9   "adress": null
10 }
```

GET http://localhost:8080/market/products/ Send

Params Authorization Headers (9) Body Scripts Settings Cookies

Bearer Token a collaborative environment, v

The authorization header will be automatically generated when you send the request. Learn more about [Bearer Token](#) authorization.

Token

Body Cookies Headers (8) Test Results

Pretty Raw Preview Visualize JSON

```
1 [
2   {
3     "id": 1,
4     "name": "Summer Dress",
5     "price": 100,
6     "createdAt": "2024-09-09T15:58:27.575Z",
7     "updatedAt": "2024-09-09T15:58:27.575Z"
8   }
9 ]
```

Вывод

В ходе данной работы приложение было разделено на микросервисы и упаковано в docker-контейнеры. Было обеспечено сетевое взаимодействие между различными частями приложения с помощью docker-compose и nginx.