# САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

Дисциплина: Бек-энд разработка

Отчет Домашнее задание №4

Выполнил: Горбатов Дмитрий Алексеевич гр. K33402

Преподаватель: Добряков Давид Ильич

## Задание

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.

## Ход работы

### Docker-compose.yml:

```
docker-compose.yml
     services:
        image: rabbitmq:3-management
        - app-network
        build:
          context: ./auth
         dockerfile: Dockerfile
         - "8000:8000"
         depends_on:
         - rabbitmq
         PORT: 8000
         RABBITMQ_HOST: rabbitmq
         RABBITMQ_PORT: 5672
         command: ["wait-for-it", "rabbitmq:5672", "--", "npm", "start"]
         networks:
24
         - app-network
```

```
build:
     context: ./main
    dockerfile: Dockerfile
   ports:
   - "8001:8001"
   depends on:
   - auth
    PORT: 8001
    AUTH_SERVICE: http://auth:8000
   networks:
   - app-network
 gateway:
   build:
    context: ./gateway
    dockerfile: Dockerfile
   ports:
    - "2000:2000"
   depends on:
    - auth
    - main
   environment:
     PORT: 2000
    AUTH_SERVICE: http://auth:8000
   networks:
   - app-network
networks:
 app-network:
 driver: bridge
```

Dockerfile в микросервисе криптобиржи:

```
main > Dockerfile > ...

1 FROM node:22
2 WORKDIR /usr/src/app
3
4 COPY package*.json ./
5
6 RUN npm install
7
8 COPY . .
9
10 RUN npm run build
11
12 EXPOSE 8001
13
14 CMD ["npm", "start"]
```

Dockerfile в микросервисе аутентификации:

```
auth > Dockerfile > PROM node:22

WORKDIR /usr/src/app

COPY package*.json ./

RUN npm install

COPY . .

RUN npm run build

ADD https://raw.githubusercontent.com/vishnubob/wait-for-it/master/wait-for-it.sh /usr/local/bin/wait-for-it

RUN chmod +x /usr/local/bin/wait-for-it

EXPOSE 8000

CMD ["npm", "start"]
```

Wait-for-it скрипт, который позволяет дождаться запуска rabbitmq.

Dockerfile в gateway, который связывает эти два микросервиса на одном порте:

```
gateway > dockerfile > ...

1 FROM nginx:latest

2

3 COPY config.conf /etc/nginx/nginx.conf

4

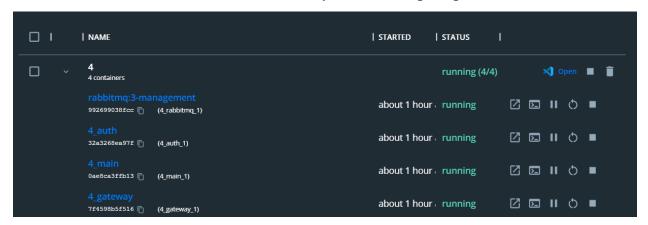
5 CMD ["nginx", "-g", "daemon off;"]

6
```

### Сам конфиг для gateway:

```
worker_processes auto;
events {
   worker_connections 1024;
http {
    server {
       listen 2000;
        location /getId {
           internal;
                             http://auth:8000/users/current;
           proxy_pass
                             off;
           proxy_redirect
           proxy_set_header Host $host;
           proxy_set_header X-Real-IP $remote_addr;
           proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
           proxy_set_header X-Forwarded-Host $server_name;
           proxy_set_header Content-Length "";
           proxy_pass_request_body off;
        location /auth {
           rewrite ^/auth/(.*) /$1 break;
           proxy_pass
                             http://auth:8000;
           proxy redirect
                             off;
           proxy_set_header Host $host;
           proxy_set_header X-Real-IP $remote_addr;
           proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
           proxy_set_header X-Forwarded-Host $server_name;
```

Он позволяет взаимодействовать между собой микросервисам.



#### Вывод:

Упаковал приложение в docker-контейнеры и обеспечил сетевое взаимодействие между различными частями приложения.