

**САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ
ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО**

Дисциплина: Бэк-энд разработка

Отчет

Лабораторная работа 4

Выполнил:

Иконенко Данил

Группа К33402

Проверил:

Добряков Д. И.

Санкт-Петербург

2022 г.

Задача

Необходимо упаковать ваше приложение в docker-контейнеры и обеспечить сетевое взаимодействие между различными частями вашего приложения.

Ход работы

Dockerfile основного сервиса:

```
1  # Build
2  FROM node:18-alpine as builder
3  ENV NPM_CONFIG_PREFIX=/home/node/.npm-global
4  ENV PATH=$PATH:/home/node/.npm-global/bin
5  RUN mkdir -p /home/node/app/node_modules && chown -R node:node /home/node/app
6  WORKDIR /home/node/app
7  COPY --chown=node:node package*.json ./
8  USER node
9  RUN npm config set unsafe-perm true
10 RUN npm install -g typescript
11 RUN npm install -g ts-node
12 RUN npm install
13 COPY --chown=node:node . .
14 RUN npm run build
15
```

```
16 # Install
17 FROM node:18-alpine
18 ENV NODE_ENV=production
19 RUN mkdir -p /home/node/app/node_modules && chown -R node:node /home/node/app
20 WORKDIR /home/node/app
21 COPY --chown=node:node package*.json ./
22 USER node
23 RUN npm install --production
24 COPY --from=builder /home/node/app/dist ./dist
25 COPY --chown=node:node .sequelizerc ./dist/
26 COPY --chown=node:node src/configs/settings.ini ./dist/configs/settings.ini
27 COPY --chown=node:node src/sequelize/index.js ./dist/sequelize/index.js
28 COPY --chown=node:node src/seeder/*.js ./dist/seeder/
29 COPY --chown=node:node docker-entrypoint.sh .
30 RUN chmod +x ./docker-entrypoint.sh
31
32 EXPOSE 5000
33 ENTRYPOINT ./docker-entrypoint.sh
```

Dockerfile сервиса авторизации:

```

1  # Build
2  FROM node:18-alpine as builder
3  ENV NPM_CONFIG_PREFIX=/home/node/.npm-global
4  ENV PATH=$PATH:/home/node/.npm-global/bin
5  RUN mkdir -p /home/node/app/node_modules && chown -R node:node /home/node/app
6  WORKDIR /home/node/app
7  COPY --chown=node:node package*.json ./
8  USER node
9  RUN npm config set unsafe-perm true
10 RUN npm install -g typescript
11 RUN npm install -g ts-node
12 RUN npm install
13 COPY --chown=node:node . .
14 RUN npm run build
15

```

```

16 # Install
17 FROM node:18-alpine
18 ENV NODE_ENV=production
19 RUN mkdir -p /home/node/app/node_modules && chown -R node:node /home/node/app
20 WORKDIR /home/node/app
21 COPY --chown=node:node package*.json ./
22 USER node
23 RUN npm install --production
24 COPY --from=builder /home/node/app/dist ./dist
25 COPY --chown=node:node src/configs/settings.ini ./dist/configs/settings.ini
26
27 EXPOSE 5001
28 CMD [ "node", "dist/index.js" ]

```

docker-compose.yml:

```

1  version: '3.9'
2  >> services:
3  >   auth_db:
4      image: mysql
5      environment:
6          - MYSQL_USER=mysql
7          - MYSQL_PASSWORD=mysql
8          - MYSQL_DATABASE=mysql
9          - MYSQL_ROOT_PASSWORD=mysql
10     expose:
11         - 3306
12     volumes:
13         - volume_auth:/var/lib/mysql
14     healthcheck:
15         test: [ "CMD", "mysqladmin" ,"ping", "-h", "localhost" ]
16         timeout: 20s
17         retries: 10
18 >   main_db:
19     image: mysql
20     environment:
21         - MYSQL_USER=mysql
22         - MYSQL_PASSWORD=mysql
23         - MYSQL_DATABASE=mysql

```

```

24         - MYSQL_ROOT_PASSWORD=mysql
25     expose:
26         - 3306
27     volumes:
28         - volume_main:/var/lib/mysql
29     healthcheck:
30         test: [ "CMD", "mysqladmin" ,"ping", "-h", "localhost" ]
31         timeout: 20s
32         retries: 10
33 >   auth_service:
34     build:
35         context: auth_service
36     depends_on:
37         auth_db:
38             condition: service_healthy
39

```

```

40 >   main_service:
41     build:
42         context: main_service
43     ports:
44         - "5000:5000"
45     depends_on:
46         main_db:
47             condition: service_healthy
48     volumes:
49         volume_auth:
50         volume_main:

```

Вывод

Я упаковал микросервисное приложение в контейнеры Docker и настроил их взаимодействие при помощи docker-compose