

**САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ
ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО**

Дисциплина: Бэк-энд разработка

Отчет

ЛР 4 - контейнеризация написанного приложения
средствами docker

Выполнил:
Сергеев Виктор
К3341

Проверил:
Добряков Д. И.

Санкт-Петербург

2025 г.

Задача

- реализовать Dockerfile для каждого сервиса;
- написать общий docker-compose.yml;
- настроить сетевое взаимодействие между сервисами.

Ход работы

Чтобы упаковать каждый сервис в контейнер, были написаны Dockerfile-ы для каждого из них. Логика Dockerfile-а следующая:

- берётся образ ноды, который будет выступать загрузчиком зависимостей, в который загружаются package*.json файлы и запускается загрузка библиотек
- берётся образ ноды, в котором уже будет поднимать приложение, в него копируются из загрузчика установленные зависимости, копируются исходники приложения и оно запускается

```
user_service > Dockerfile > ...
1  FROM node:20 AS dependencies
2
3  ENV NODE_ENV=production
4
5  WORKDIR /user_service
6  COPY package.json package-lock.json ./
7  RUN npm ci --omit=optional
8
9  FROM node:20 AS prod
10
11 ENV NODE_ENV=production
12
13 WORKDIR /user_service
14 COPY . .
15 COPY --from=dependencies /user_service/node_modules ./node_modules
16
17 EXPOSE 3000
18
19 CMD ["npm", "run", "prod"]
20
```

Рисунок 1 - пример Dockerfile для сервиса пользователей

Для каждого сервиса Dockerfile практически идентичны.

Далее требовалось подключить эти сервисы через docker-compose. Сперва была создана внутренняя сеть docker-a:

```
88 networks:
89   backend-microservices-network:
90     driver: bridge
```

Рисунок 2 - сеть докера для приложения

Теперь подключаются контейнеризированные сервисы:

```
Run Service
user_service:
  container_name: user-service
  build: ./user_service
  restart: always
  ports:
    - 3001:3000
  env_file:
    - ./user_service/.env.prod
  depends_on:
    - user_db
  networks:
    - backend-microservices-network
```

Рисунок 3 - пример подключения сервиса в компоузе

Наконец, запускаются все сервисы для проверки работоспособности приложения.

	name	container id	image	ports	cpu (%)	memory usage	memory (%)	status	restart	actions	
<input type="checkbox"/>	backend-express	-	-	-	2.26%	689.69MB / 40.46GB	9.99%	0E	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	social-db	2d4687baad7a	postgres:11	5433:5432	0%	37.29MB / 6.74GiB	0.54%	0E	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	user-db	2788c47a6364	postgres:11	5433:5432	0%	34.4MB / 6.74GB	0.5%	0E	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	recipe-db	ba40fbccd211	postgres:11	5434:5432	0.01%	36.8MB / 6.74GB	0.53%	0E	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	user-service	af1ca833d0e6	backend-ex	3001:3000	0.77%	185.5MB / 6.74GiB	2.69%	0E	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	recipe-service	f75347821c4a	backend-ex	3002:3000	0.7%	181.7MB / 6.74GiB	2.63%	0E	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	social-service	470b776498e5	backend-ex	3003:3000	0.78%	214MB / 6.74GB	3.1%	0E	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Рисунок 4 - контейнеры в docker desktop

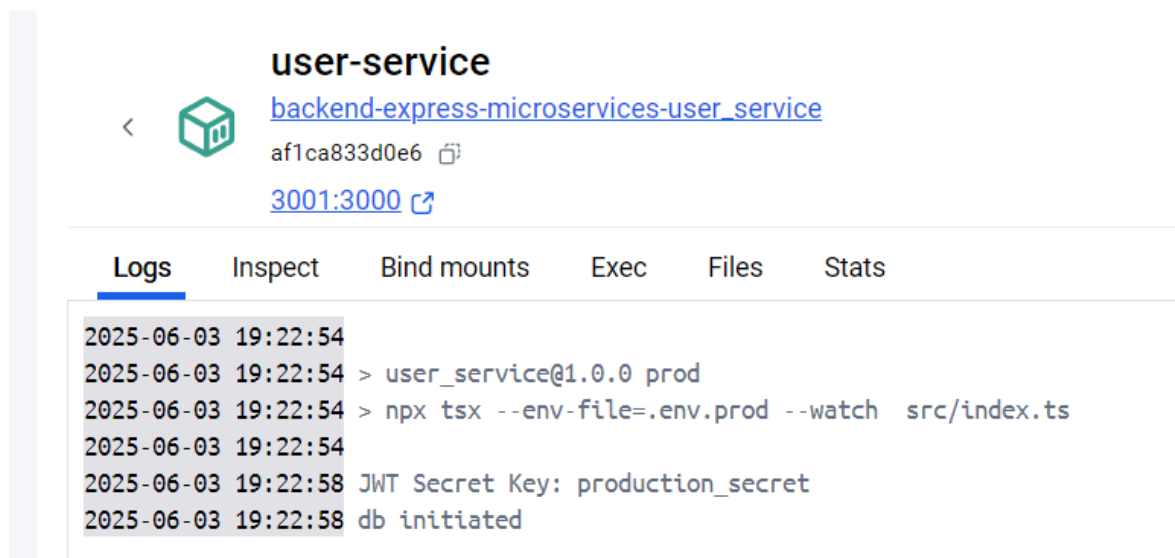


Рисунок 5 - лог сервиса пользователей

Вывод

В процессе работы ранее написанные микросервисы приложения были упакованы в контейнеры с помощью средств docker. Был написан общий компоуз файл, который поднимает всё приложение вместе с базами данных. В компоуз файле была настроена внутренняя сеть докера для осуществления сетевого взаимодействия между сервисами.