САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

Дисциплина: Бэк-энд разработка

Отчет

Лабораторная работа №4

Выполнил:
Плахтий Марк
Вячеславович
Группа К3340

Проверил:
Добряков Д. И.

Санкт-Петербург

2025 г.

Задание

Реализовать Dockerfile для каждого микросервиса, написать общий docker-compose.yml и настроить сетевое взаимодействие между сервисами в архитектуре микросервисов.

Ход работы

1. Анализ архитектуры микросервисов

Проект состоит из трех микросервисов:

- **user-service** (порт 3001) - управление пользователями и аутентификация
- **property-service** (порт 3002) - управление недвижимостью (здания, квартиры)
- **contract-service** (порт 3003) - управление контрактами аренды

2. Реализация Dockerfile для каждого сервиса

Для каждого сервиса был создан Dockefile:

```
FROM node: 18 - alpine

WORKDIR / app

# Copy package files
COPY package *.json./

# Install dependencies
RUN npm install

# Copy source code
COPY. .

# Build the application
RUN npm run build

# Expose port
EXPOSE 3003

# Start the application
CMD["npm", "start"]
```

3. Создание общего docker-compose.yml

```
version: '3.8'
services:
user - service:
build:
context: ./ user - service
dockerfile: Dockerfile
ports:
- "3001:3001"
environment:
- PORT=3001
    - NODE_ENV=development
volumes:
- /app/node_modules
networks:
- microservices - network
property - service:
build:
context: ./ property - service
dockerfile: Dockerfile
ports:
- "3002:3002"
environment:
- PORT=3002
    - NODE_ENV=development
volumes:
- /app/node_modules
networks:
- microservices - network
contract - service:
build:
context: ./ contract - service
dockerfile: Dockerfile
ports:
- "3003:3003"
environment:
- PORT=3003
    - NODE_ENV=development
        - USER_SERVICE_URL=http://user-service:3001
- PROPERTY_SERVICE_URL=http://property-service:3002
volumes:
- /app/node_modules
depends_on:
- user - service
    - property - service
networks:
- microservices - network

networks:
microservices - network:
driver: bridge
```

4. Настройка сетевого взаимодействия

4.1 Создание общей сети

В docker-compose.yml создается общая сеть microservices-networkс драйвером bridge, которая позволяет всем сервисам взаимодействовать друг с другом.

4.2 Конфигурация переменных окружения

Для contract-service настроены переменные окружения:
- USER_SERVICE_URL=http://user-service:3001 - URL для обращения к user-service
- PROPERTY_SERVICE_URL=http://property-service:3002 - URL для обращения к property-service

4.3 Зависимости между сервисами

Contract-service имеет зависимости от user-service и property-service, что гарантирует правильный порядок запуска контейнеров.

5. Особенности реализации

5.1 Использование Node.js Alpine образа

Все сервисы используют легковесный образ node:18-alpine, что уменьшает размер контейнеров и время сборки.

5.2 Многоэтапная сборка

Dockerfile'ы реализуют следующую последовательность:

1. Копирование package.json файлов
2. Установка зависимостей
3. Копирование исходного кода
4. Сборка TypeScript приложения
5. Запуск приложения

5.3 Health checks

Каждый сервис предоставляет endpoint /healthдля проверки работоспособности.

Вывод

В ходе лабораторной работы была успешно реализована контейнеризация микросервисной архитектуры с использованием Docker. Созданы Dockerfile'ы для каждого из трех сервисов (user-service, property-service, contract-service), настроен общий docker-compose.yml файл для оркестрации контейнеров и реализовано сетевое взаимодействие между сервисами.

Основные достижения:

- Все сервисы успешно контейнеризированы с использованием оптимизированных Dockerfile'ов
- Настроена общая сеть для межсервисного взаимодействия
- Реализованы зависимости между сервисами для корректного порядка запуска
- Настроены переменные окружения для конфигурации URL'ов сервисов
- Обеспечена изоляция и масштабируемость микросервисов

Архитектура готова к развертыванию в продакшн-среде и может быть легко масштабирована при необходимости.