

**САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ
ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО**

Дисциплина: Бэк-энд разработка

Отчет

Лабораторная работа №4

Выполнил:

Ананьев Никита

К3340

Проверил:

Добряков Д. И.

Санкт-Петербург

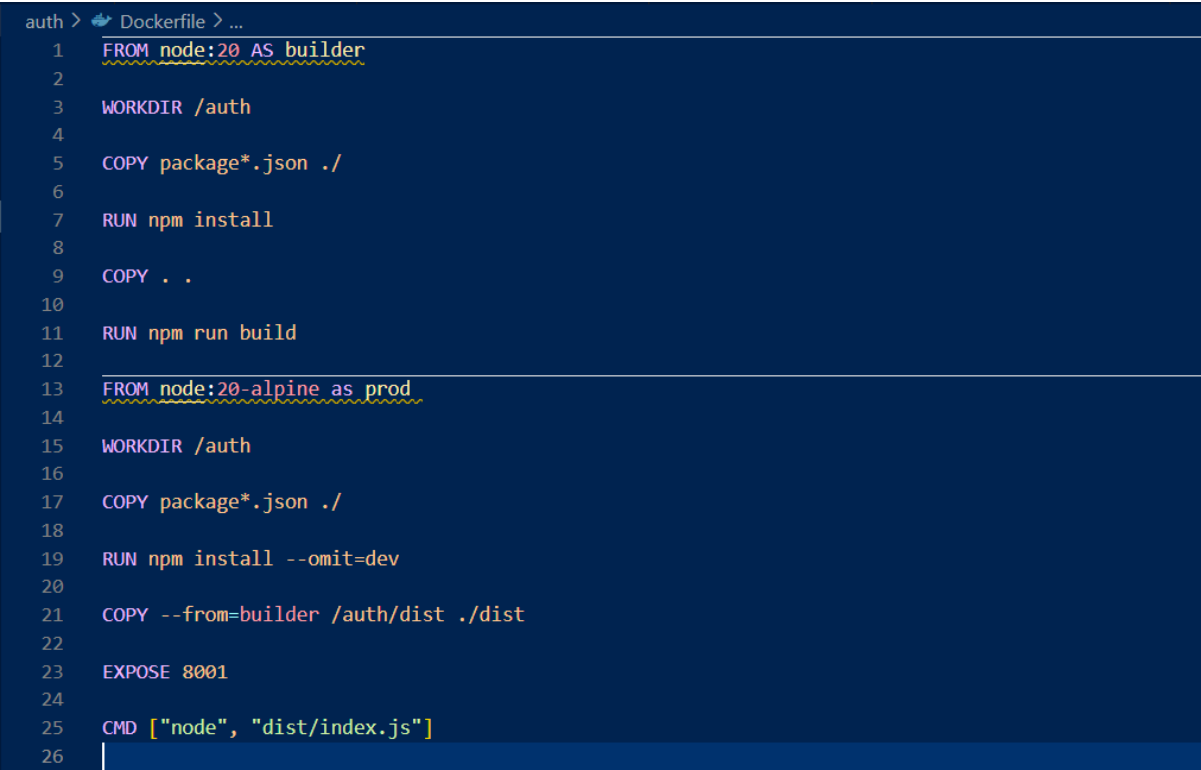
2025 г.

Задача

Контейнеризация написанного приложения средствами docker.

Ход работы

Для запуска сервиса в докере, необходимо написать файл создания образа или же Dockerfile (см. рисунок 1):



```
auth > Dockerfile > ...
1  FROM node:20 AS builder
2
3  WORKDIR /auth
4
5  COPY package*.json ./
6
7  RUN npm install
8
9  COPY . .
10
11 RUN npm run build
12
13 FROM node:20-alpine as prod
14
15 WORKDIR /auth
16
17 COPY package*.json ./
18
19 RUN npm install --omit=dev
20
21 COPY --from=builder /auth/dist ./dist
22
23 EXPOSE 8001
24
25 CMD ["node", "dist/index.js"]
26
```

Рисунок 1 - пример Dockerfile для сервиса профиля

Используем multistage сборку - на первом этапе собираем наше приложение в единый компактный js файл, на втором берем только этот файл и нужные для работы приложения зависимости. Открываем нужный нам порт, чтобы в дальнейшем можно было общаться с контейнером по сети, запускаем наше приложение.

Примерно такие же действия необходимо повторить и для остальных сервисов

Для удобного запуска и настройки сетевого взаимодействия между несколькими контейнерами используется инструмент docker compose. Чтобы воспользоваться им, необходимо создать и описать наши контейнеры в файле с названием docker-compose.yml (см. рисунок 2):

```
1  version: '3'
2
3  services:
4    db:
5      container_name: db_container
6      image: postgres:alpine
7      restart: always
8      ports:
9        - 5432:5432
10     volumes:
11       - pgdata:/var/lib/postgresql/data
12     env_file:
13       - db.env
14
15   auth:
16     container_name: auth_container
17     restart: always
18     build: ./auth
19     ports:
20       - 8001:8001
21     env_file:
22       - ./auth/auth.env
23     depends_on:
24       - db
25
26   rent:
27     container_name: rent_container
28     restart: always
29     build: ./rent
30     ports:
31       - 8003:8003
32     env_file:
33       - ./rent/rent.env
34     depends_on:
35       - db
36
37   messenger:
38     container_name: messenger_container
39     restart: always
40     build: ./messenger
41     ports:
42       - 8002:8002
43     env_file:
44       - ./messenger/messenger.env
45     depends_on:
46       - db
47
48   volumes:
49     pgdata: {}
50
```

Рисунок 2 - docker compose файл

Для упрощения был создан один контейнер postgres (но с разными бд). В настоящей системе с микросервисной архитектурой это был бы отдельный (или даже несколько) контейнер на каждый сервис.

Вывод

Сегодня Docker является практически жизненно важным инструментом для деплоя веб-приложений, так как при его использовании не нужно беспокоиться за несовместимости систем, зависимостей, используемых в приложении. Docker-compose - удобный инструмент, который значительно ускоряет процесс запуска и остановки контейнеров, настройки взаимодействия между ними.