САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

Дисциплина: Бэк-энд разработка

Отчет

Лабораторные работы 3 и 4. “Разработка микросервисов и деплой с помощью Docker”

Выполнил:

Тимофеев Николай

Группа

K33402

Проверил:
Добряков Д. И.

Санкт-Петербург

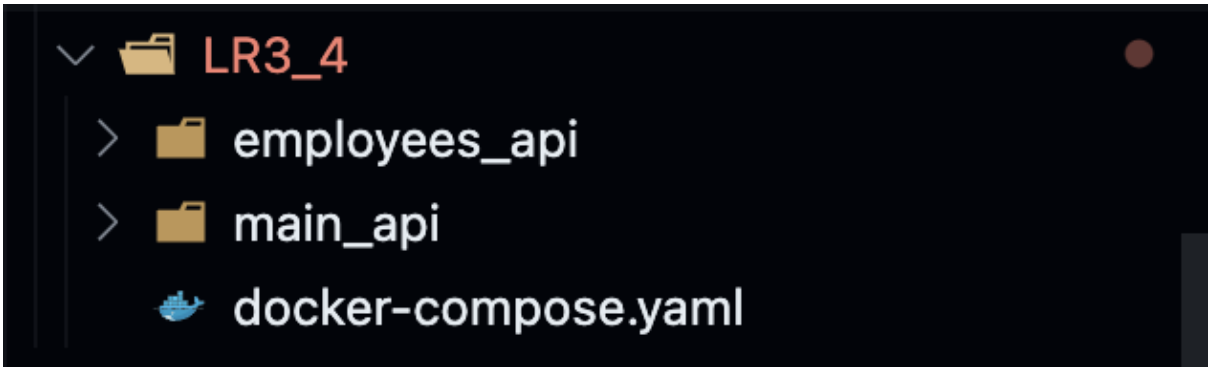
2023 г.

Задача

Разбить существующий монолитный сервис на микросервисы. Связать их средствами Docker.

Ход работы

Итоговая структура проекта:



В сервис employees_api была вынесена вся логика работы с сотрудниками предприятия, в main_api остался основной функционал: авторизация, регистрация, учет товаров на складе и тд.

Был написан docker-compose:

```yaml
version: '3'

services:
  main_api:
    build: ./main_api
    restart: always
    depends_on:
        - empl_api
    ports:
      - '5000:5000'

  empl_api:
    build: ./employees_api
    restart: always
    ports:
      - '5001:5001'

networks:
  default:
    name: lab_network
```

И Dockerfile для каждого из сервисов:

labs > K33402 > Timofeev_Nikolai > LR3_4 > employees_api > Dockerfile > ...

```
1  FROM node:18.14.1-alpine
2  WORKDIR /empl_api
3  COPY . .
4  RUN npm install
5  EXPOSE 5002
6  CMD ["npm", "run", "start"]
```

labs > K33402 > Timofeev_Nikolai > LR3_4 > main_api > Dockerfile > ...

```
1  FROM node:18.14.1-alpine
2  WORKDIR /main_api
3  COPY . .
4  RUN npm install
5  EXPOSE 5000
6  CMD ["npm", "run", "start"]
```

Изменилась и логика работы с сотрудниками в основном сервисе, теперь запрос сначала переходит на микросервис, а клиент получает ответ от него:

```
  get = async (request: any, response: any) => {
    try {
      const url = `http://empl_api:5001/api/v1/employees/${Number(request.params.id)}`;
      const resp = await axios.get(url);
      response.send(resp.data);
    } catch (error: any) {…
    }
  };
  post = async (request: any, response: any) => {
    const { body } = request;
    try {
      const url = `http://empl_api:5001/api/v1/employees/`;
      const resp = await axios.post(url, body);
      response.send(resp.data);
    } catch (error: any) {…
    }
  };
```

```
getAll = async (request: any, response: any) => {
  try {
    const url = `http://empl_api:5001/api/v1/employees/`;
    const resp = await axios.get(url);
    response.send(resp.data);
  } catch (error: any) {…
  }
};

changeStatus = async (request: any, response: any) => {
  try {
      const id = request.params.id
      const url = `http://empl_api:5001/api/v1/employees/${Number(id)}`;
      const resp = await axios.delete(url);
      response.send(resp.data);
  } catch (error: any) {…
  }
}
```

Так как была создана сеть и сервисы связаны, мы можем обращаться к микросервису по адресу empl_api:5001, что очень удобно (адрес контейнера меняется при каждом запуске, а в таком случае докер резолвит его автоматически)

Вывод

В ходе выполнения работы я разработал микросервис, который был включен в общую инфраструктуру сервиса. Написал Dockerfile и docker-compose для быстрого и удобного развертывания приложения на любом устройстве (поддерживающим систему контейнеризации Docker).