САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

Дисциплина: Бэк-энд разработка

Отчет

Лабораторная работа №3

Выполнил:

Пиотуховский Александр

K3341

Проверил:
Добряков Д. И.

Санкт-Петербург

2025 г.

Задача

- выделить самостоятельные модули в вашем приложении;
- провести разделение своего API на микросервисы (минимум, их должно быть 3);
- настроить сетевое взаимодействие между микросервисами.

Ход работы

1. Выделение самостоятельных модулей

После анализа приложения было выделено три самостоятельных модуля, каждый из которых можно сделать отдельным микросервисом. Это сервис авторизации, сервис отправки почтовых писем и сам сервис фильмов

2. Разделение API на микросервисы и настройка сетевого взаимодействия

Для коммуникации сервиса фильмов с сервисом для рассылки электронных писем был задействован брокер сообщений kafka, инфраструктура которого была развёрнута при помощи docker compose (рисунок 1).

```yaml
services:
  zookeeper:
    image: confluentinc/cp-zookeeper:7.4.0
    container_name: zookeeper
    ports:
      - "2181:2181"
    environment:
      ZOOKEEPER_CLIENT_PORT: 2181
    networks:
      - mail_net

  kafka:
    image: confluentinc/cp-kafka:7.4.0
    container_name: kafka
    healthcheck:
      test: ["CMD", "bash", "-c", "echo > /dev/tcp/localhost/9092"]
      interval: 5s
      timeout: 5s
      retries: 5
    depends_on:
      - zookeeper
    ports:
      - "9092:9092"
    environment:
      KAFKA_BROKER_ID: 1
      KAFKA_ZOOKEEPER_CONNECT: "zookeeper:2181"
      KAFKA_LISTENERS: "PLAINTEXT://0.0.0.0:9092"
      KAFKA_ADVERTISED_LISTENERS: "PLAINTEXT://kafka:9092"
      KAFKA_OFFSETS_TOPIC_REPLICATION_FACTOR: 1
    networks:
      - mail_net
```

Рисунок 1 – брокер kafka в docker-compose

На рисунке 2 изображён код микросервиса для отправки почтовых писем. Он слушает топик в брокере и читает сообщения, из которых собирает адрес получателя и код верификации, который нужно отправить. Затем по SMTP отправляется email с кодом на указанный адрес.

```
const kafka = new Kafka({
    clientId: 'mail-sender',
    brokers: KAFKA_BROKERS.split(','),
});

const consumer = kafka.consumer({ groupId: 'mail-group' });

const transporter = nodemailer.createTransport({
    host: MAIL_HOST,
    port: Number(MAIL_PORT),
    secure: false,
    requireTLS: true,
    auth: {
        user: MAIL_USER,
        pass: MAIL_PASS,
    },
    tls: {
        minVersion: 'TLSv1.2',
    },
});

async function sendMail(to: string, code: string) {
    const info = await transporter.sendMail({
        from: `"${MAIL_FROM_NAME}" <${MAIL_USER}>`,
        to,
        subject: 'Ваш одноразовый код',
        text: `Ваш код: ${code}`,
        html: `<p>Ваш код: <b>${code}</b></p>`,
    });
     console.log(`Message sent: ${info.messageId}`);
}

async function run() {
    await consumer.connect();
    // @ts-ignore
    await consumer.subscribe({ topic: KAFKA_TOPIC, fromBeginning: false });
    console.log(`Subscribed to topic ${KAFKA_TOPIC}`);

    await consumer.run({
        eachMessage: async ({ message }) => {
            try {
                if (!message.value) return;
                const payload = JSON.parse(message.value.toString());
                const { email, code } = payload;
                console.log(`Received: ${email}, code=${code}`);
                await sendMail(email, code);
            } catch (err) {
```

Рисунок 2 – микросервис для отправки писем

По аналогии был выделен сервис авторизации. Он является автономным сервисом, который поставляет клиентскому приложению JWT токены, созданные при помощи алгоритма RS256. Идея в том, что токен доступа создаётся с помощью приватного ключа, который есть только у сервиса авторизации. Также есть публичный ключ, с помощью которого сервис с фильмами может проверить токен без синхронного обращения а сервису авторизации. Это положительно сказывается на доступности системы.

Вывод

В ходе выполнения лабораторной работы 3 созданное ранее приложение было поделено на автономные сервисы, каждый из которых может работать на разных серверах и масштабироваться независимо друг от друга.