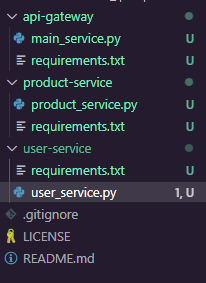
Отчет по практической работе №3

Выполнил студент ЭФБО-09-23 Швецов Демьян

Мой репозиторий: https://github.com/twisted-none/soundzen

Сделал микросервисы:



from flask import Flask, jsonify, request

app = Flask(\_\_name\_\_)

*# Тестовые данные продуктов*

products = [

    {'id': 1, 'name': 'Ноутбук ASUS', 'price': 65000, 'category': 'Электроника', 'stock': 15},

    {'id': 2, 'name': 'Смартфон Samsung', 'price': 35000, 'category': 'Электроника', 'stock': 25},

    {'id': 3, 'name': 'Клавиатура Logitech', 'price': 3500, 'category': 'Периферия', 'stock': 40},

    {'id': 4, 'name': 'Монитор LG 27"', 'price': 22000, 'category': 'Электроника', 'stock': 10},

    {'id': 5, 'name': 'Мышь Razer', 'price': 2800, 'category': 'Периферия', 'stock': 50},

]

@app.route('/')

def home():

    return jsonify({

        'service': 'Product Service',

        'status': 'running',

        'version': '1.0'

    })

@app.route('/products', *methods*=['GET'])

def get\_products():

    category = request.args.get('category')

    if category:

        filtered = [p for p in products if p['category'].lower() == category.lower()]

        return jsonify(filtered)

    return jsonify(products)

@app.route('/products/<int:product\_id>', *methods*=['GET'])

def get\_product(*product\_id*):

    product = next((p for p in products if p['id'] == *product\_id*), None)

    if product:

        return jsonify(product)

    return jsonify({'error': 'Продукт не найден'}), 404

@app.route('/products', *methods*=['POST'])

def create\_product():

    data = request.get\_json()

    new\_product = {

        'id': len(products) + 1,

        'name': data.get('name'),

        'price': data.get('price'),

        'category': data.get('category'),

        'stock': data.get('stock', 0)

    }

    products.append(new\_product)

    return jsonify(new\_product), 201

@app.route('/health', *methods*=['GET'])

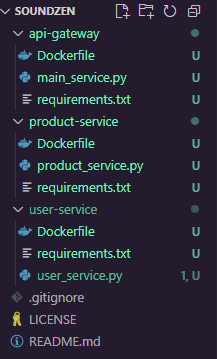
def health():

    return jsonify({'status': 'healthy', 'service': 'product-service'})

if \_\_name\_\_ == '\_\_main\_\_':

    app.run(*host*='0.0.0.0', *port*=5002, *debug*=True)

Сделал Dockerfile для микросервисов:



*# Используем официальный образ Python*

FROM python:3.11-slim

*# Устанавливаем рабочую директорию*

WORKDIR */app*

*# Копируем файл с зависимостями*

COPY *requirements.txt* *.*

*# Устанавливаем зависимости*

RUN *pip* *install* *--no-cache-dir* *-r* *requirements.txt*

*# Копируем код приложения*

COPY *main\_service.py* *.*

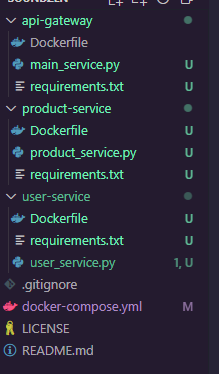
*# Открываем порт*

EXPOSE *5000*

*# Запускаем приложение*

CMD *[*"python"*,* "main\_service.py"*]*

Создал docker-compose.yml файл для одновременного запуска всех файлов:



version: '3.8'

services:

*# User Service*

  user-service:

    build:

      context: ./user-service

      dockerfile: Dockerfile

    container\_name: user-service

    ports:

      - "5001:5001"

    networks:

      - microservices-network

    healthcheck:

      test: ["CMD", "curl", "-f", "http://localhost:5001/health"]

      interval: 30s

      timeout: 10s

      retries: 3

      start\_period: 40s

    restart: unless-stopped

*# Product Service*

  product-service:

    build:

      context: ./product-service

      dockerfile: Dockerfile

    container\_name: product-service

    ports:

      - "5002:5002"

    networks:

      - microservices-network

    healthcheck:

      test: ["CMD", "curl", "-f", "http://localhost:5002/health"]

      interval: 30s

      timeout: 10s

      retries: 3

      start\_period: 40s

    restart: unless-stopped

*# API Gateway (Main Service)*

  api-gateway:

    build:

      context: ./api-gateway

      dockerfile: Dockerfile

    container\_name: api-gateway

    ports:

      - "5000:5000"

    environment:

      - USER\_SERVICE\_URL=http://user-service:5001

      - PRODUCT\_SERVICE\_URL=http://product-service:5002

    depends\_on:

      - user-service

      - product-service

    networks:

      - microservices-network

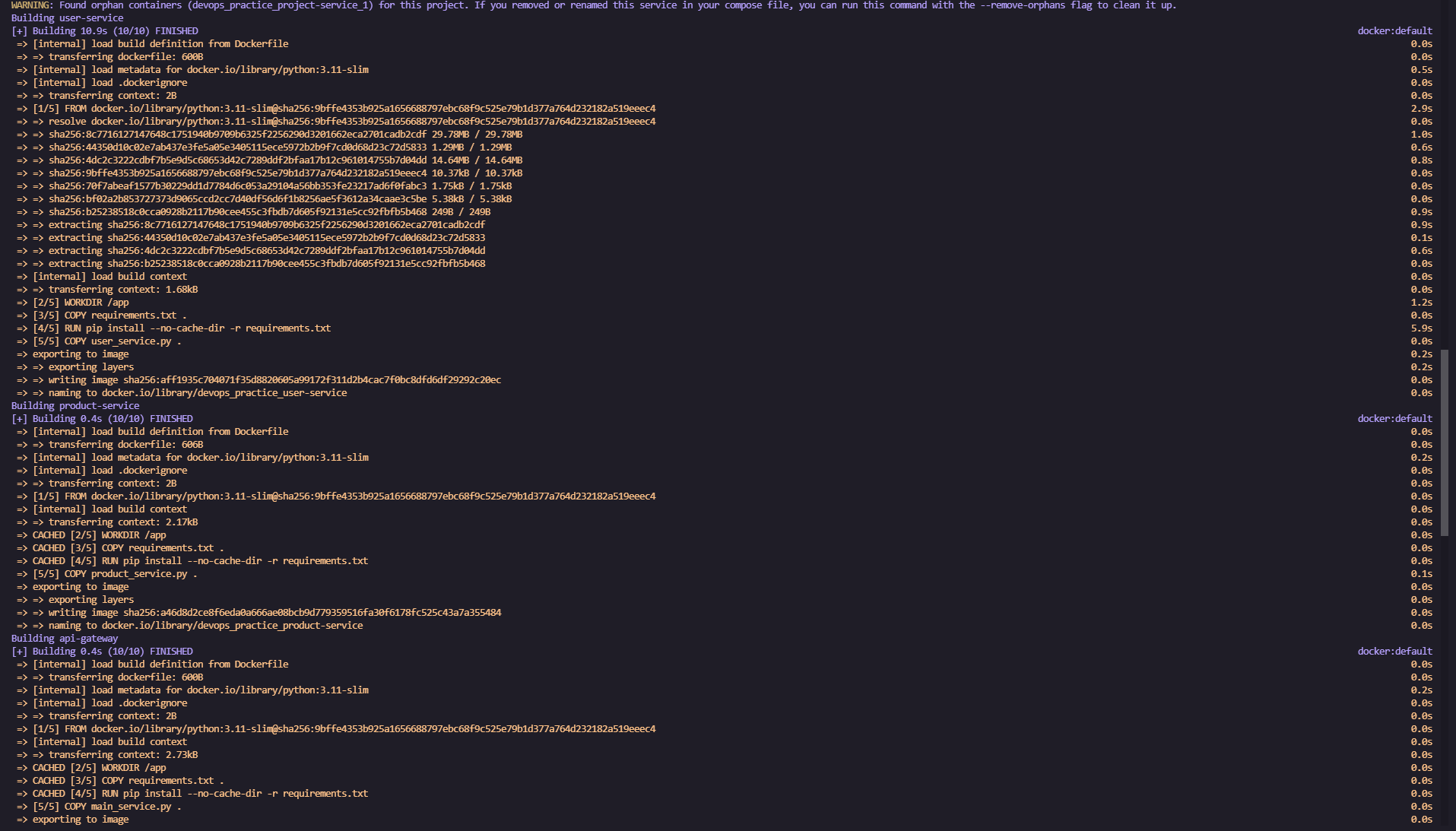
    restart: unless-stopped

networks:

  microservices-network:

    driver: bridge

Запустил микросервисы на своем хостинге (ВМ):



Проверил работоспособность микросервисов:

