САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

Дисциплина: Бэк-энд разработка

Отчет

Лабораторная работа №4

Выполнил:

Пиотуховский Александр

K3341

Проверил:
Добряков Д. И.

Санкт-Петербург

2025 г.

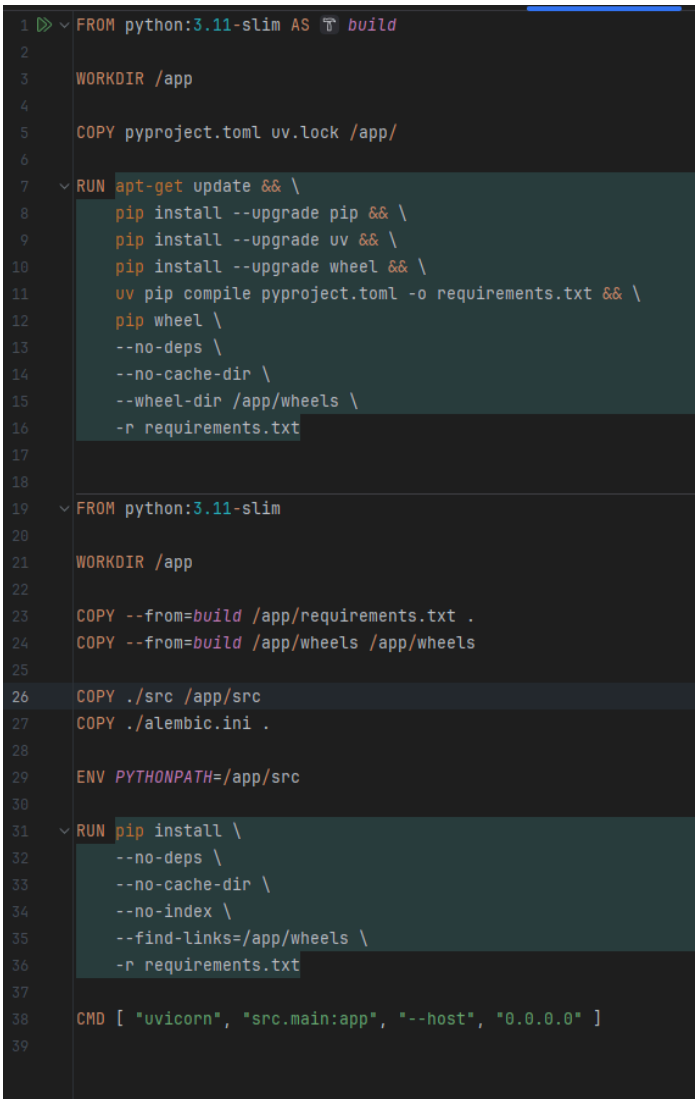
Задача

- реализовать Dockerfile для каждого сервиса;
- написать общий docker-compose.yml;
- настроить сетевое взаимодействие между сервисами.

Ход работы

1. Создание Dockerfile

Для создания образа сервиса с фильмами был написан Dockerfile с использованием multi-stage сборки, чтобы минимизировать размер итогового образа. Далее из этого образа будет запущен контейнер с сервисом. На рисунке 1 изображён этот Dockerfile.



```
FROM python:3.11-slim AS build

WORKDIR /app

COPY pyproject.toml uv.lock /app/

RUN apt-get update && \
    pip install --upgrade pip && \
    pip install --upgrade uv && \
    pip install --upgrade wheel && \
    uv pip compile pyproject.toml -o requirements.txt && \
    pip wheel \
    --no-deps \
    --no-cache-dir \
    --wheel-dir /app/wheels \
    -r requirements.txt


FROM python:3.11-slim

WORKDIR /app

COPY --from=build /app/requirements.txt .
COPY --from=build /app/wheels /app/wheels

COPY ./src /app/src
COPY ./alembic.ini .

ENV PYTHONPATH=/app/src

RUN pip install \
    --no-deps \
    --no-cache-dir \
    --no-index \
    --find-links=/app/wheels \
    -r requirements.txt

CMD [ "uvicorn", "src.main:app", "--host", "0.0.0.0" ]
```

Рисунок 1 – Dockerfile для сервиса фильмов

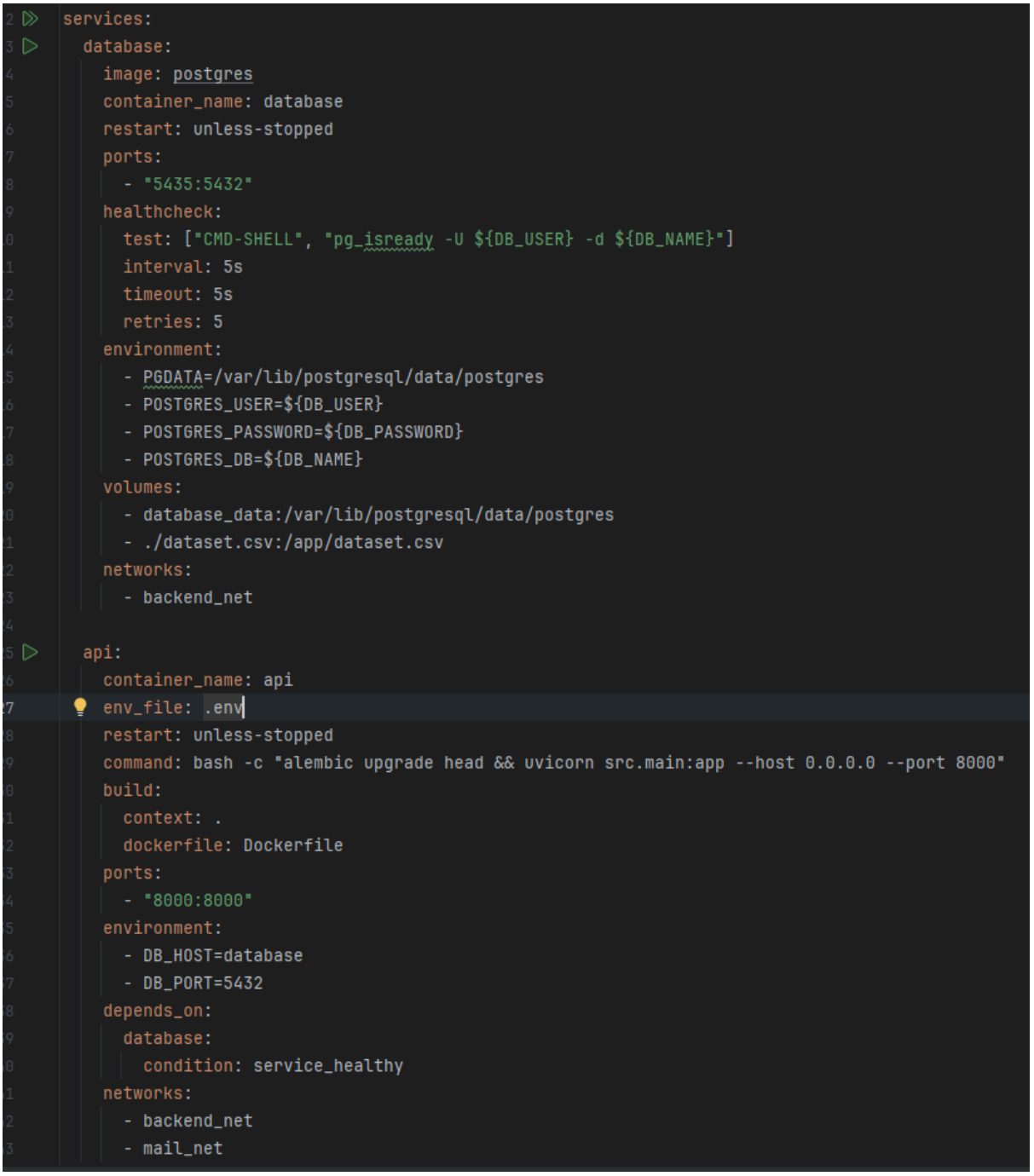
Для остальных сервисов были аналогично созданы Dockerfile, но предназначенные для сборки приложений на node.js. На рисунке 2 изображен Dockerfile сервиса отправки писем.

```
FROM node:18-alpine AS builder

WORKDIR /app

COPY package.json package-lock.json tsconfig.json ./

RUN npm ci

COPY src ./src
RUN npm run build


FROM node:18-alpine

WORKDIR /app

COPY --from=builder /app/package.json /app/package-lock.json ./
COPY --from=builder /app/node_modules ./node_modules
COPY --from=builder /app/dist ./dist

CMD ["node", "dist/index.js"]
```

Рисунок 2 – Dockerfile Для сервиса отправки писем

2. Создание единого docker-compose файла

Для удобства запуска каждого сервиса был написан docker-compose файл со всей необходимой инфраструктурой. На рисунке 3 изображён docker-compose файл для сервиса фильмов.



```yaml
services:
  database:
    image: postgres
    container_name: database
    restart: unless-stopped
    ports:
      - "5435:5432"
    healthcheck:
      test: ["CMD-SHELL", "pg_isready -U ${DB_USER} -d ${DB_NAME}"]
      interval: 5s
      timeout: 5s
      retries: 5
    environment:
      - PGDATA=/var/lib/postgresql/data/postgres
      - POSTGRES_USER=${DB_USER}
      - POSTGRES_PASSWORD=${DB_PASSWORD}
      - POSTGRES_DB=${DB_NAME}
    volumes:
      - database_data:/var/lib/postgresql/data/postgres
      - ./dataset.csv:/app/dataset.csv
    networks:
      - backend_net

  api:
    container_name: api
    env_file: .env
    restart: unless-stopped
    command: bash -c "alembic upgrade head && uvicorn src.main:app --host 0.0.0.0 --port 8000"
    build:
      context: .
      dockerfile: Dockerfile
    ports:
      - "8000:8000"
    environment:
      - DB_HOST=database
      - DB_PORT=5432
    depends_on:
      database:
        condition: service_healthy
    networks:
      - backend_net
      - mail_net
```

Рисунок 3 – docker-compose для сервиса фильмов

3. Настройка сетевого взаимодействия между контейнерами

Для взаимодействия сервисов внутри сети были созданы bridge сети, с помощью которых сервисы могут коммуницировать друг с другом.

Вывод

В ходе выполнения лабораторной работы 4 были изучены инструменты для контейнеризации приложений, такие как docker и docker compose. Все сервисы работают в изолированных контейнерах и общаются между собой при помощи внутренней сети.