САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

Дисциплина: Бэк-энд разработка

Отчет

Лабораторная работа №5

Выполнил:

Платонова Александра

Группа K3439

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.

Проект: Прикладное программное обеспечение деятельности отдела заселения муниципальных общежитий администрации города.

Ход работы

В проекте реализовано четыре основных микросервиса, каждый из которых отвечает за свою независимую предметную область. Hostel Service полностью управляет инфраструктурой общежитий, включая адреса, здания, комнаты, оборудование и технические характеристики. Resident Service сосредоточен на данных о гражданах, их персональной информации, документах, категориях очереди и параметрах нуждаемости. CheckInOut Service отвечает за все процессы, связанные с проживанием, а именно за договоры, заселение, выселение, переселения и связанные события. Payment Service выделен отдельно для финансового учёта, включая тарифы, начисления, платежи и контроль задолженностей, чтобы обеспечить точность и независимость денежных операций.

Итоговая структура проекта представлена на рисунке 1.

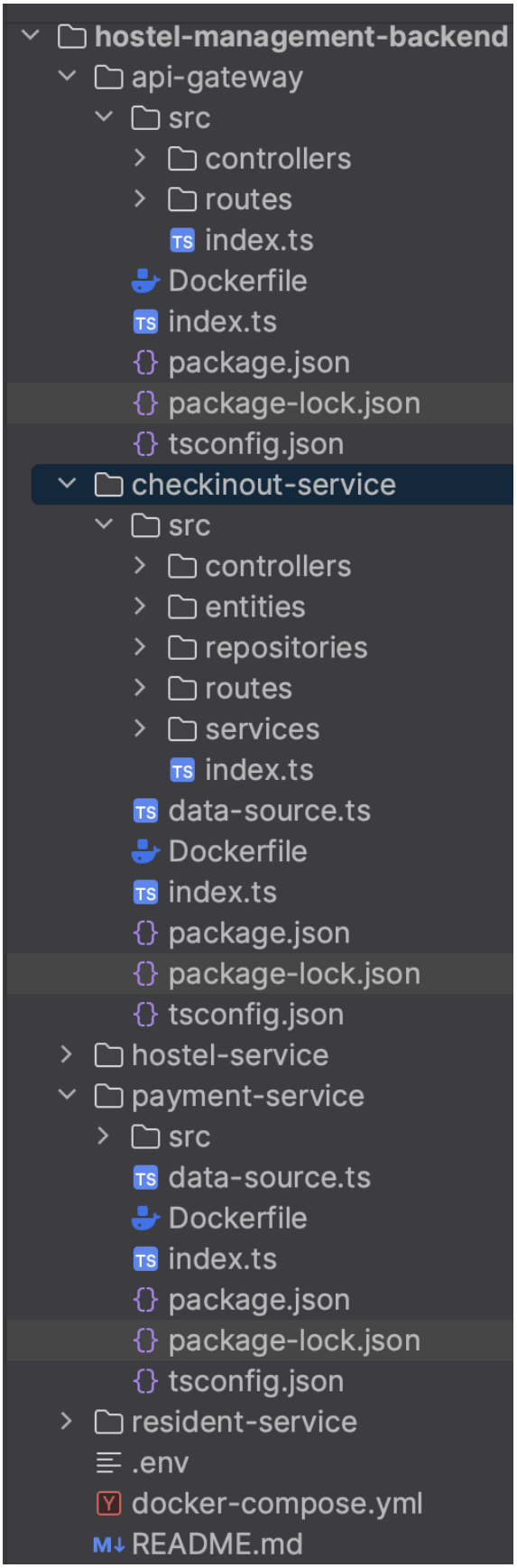

Рисунок 1 – Итоговая структура проекта

Ниже приведен листинг docker-compose файла.

Листинг 1 – docker-compose файл

```
services:
  # PostgreSQL для Hostel Service
  hostel-db:
    image: postgres:16
    restart: always
    environment:
      POSTGRES_USER: hostel_user
      POSTGRES_PASSWORD: hostel_pass_2026
      POSTGRES_DB: hostel_db
    ports:
      - "5433:5432"
    volumes:
      - hostel-db-data:/var/lib/postgresql/data
    healthcheck:
      test: ["CMD-SHELL", "pg_isready -U hostel_user"]
      interval: 10s
      timeout: 5s
      retries: 5


  # PostgreSQL для Resident Service
  resident-db:
    image: postgres:16
    restart: always
    environment:
      POSTGRES_USER: resident_user
```

```
    POSTGRES_PASSWORD: resident_pass_2026
    POSTGRES_DB: resident_db
  ports:
    - "5434:5432"
  volumes:
    - resident-db-data:/var/lib/postgresql/data
  healthcheck:
    test: ["CMD-SHELL", "pg_isready -U resident_user"]
    interval: 10s
    timeout: 5s
    retries: 5

# PostgreSQL для CheckInOut Service
checkinout-db:
  image: postgres:16
  restart: always
  environment:
    POSTGRES_USER: checkinout_user
    POSTGRES_PASSWORD: checkinout_pass_2026
    POSTGRES_DB: checkinout_db
  ports:
    - "5435:5432"
  volumes:
    - checkinout-db-data:/var/lib/postgresql/data
  healthcheck:
    test: ["CMD-SHELL", "pg_isready -U checkinout_user"]
    interval: 10s
```

```
    timeout: 5s
    retries: 5

# PostgreSQL для Payment Service
payment-db:
  image: postgres:16
  restart: always
  environment:
    POSTGRES_USER: payment_user
    POSTGRES_PASSWORD: payment_pass_2026
    POSTGRES_DB: payment_db
  ports:
    - "5436:5432"
  volumes:
    - payment-db-data:/var/lib/postgresql/data
  healthcheck:
    test: ["CMD-SHELL", "pg_isready -U payment_user"]
    interval: 10s
    timeout: 5s
    retries: 5

# API Gateway
api-gateway:
  build:
    context: ./api-gateway
    dockerfile: Dockerfile
  restart: always
```

```
    ports:
      - "3000:3000"
    environment:
      NODE_ENV: development
      PORT: 3000
      HOSTEL_SERVICE_URL: http://hostel-service:3001
      RESIDENT_SERVICE_URL: http://resident-service:3002
      CHECKINOUT_SERVICE_URL: http://checkinout-service:3003
      PAYMENT_SERVICE_URL: http://payment-service:3004
    depends_on:
      - hostel-service
      - resident-service
      - checkinout-service
      - payment-service
    networks:
      - backend-net

  # Hostel Service
  hostel-service:
    build:
      context: ./hostel-service
      dockerfile: Dockerfile
    restart: always
    environment:
      NODE_ENV: development
      PORT: 3001
DATABASE_URL:
postgres://hostel_user:hostel_pass_2026@hostel-db:5432/hostel_
db
```

```
    depends_on:
      hostel-db:
        condition: service_healthy
    networks:
      - backend-net
  # Resident Service
  resident-service:
    build:
      context: ./resident-service
      dockerfile: Dockerfile
    restart: always
    environment:
      NODE_ENV: development
      PORT: 3002
DATABASE_URL:
postgres://resident_user:resident_pass_2026@resident-db:5432/r
esident_db
    depends_on:
      resident-db:
        condition: service_healthy
    networks:
      - backend-net
  # CheckInOut Service
  checkinout-service:
    build:
      context: ./checkinout-service
      dockerfile: Dockerfile
    restart: always
```

```
    environment:
      NODE_ENV: development
      PORT: 3003
DATABASE_URL:
postgres://checkinout_user:checkinout_pass_2026@checkinout-db:
5432/checkinout_db
    depends_on:
      checkinout-db:
        condition: service_healthy
    networks:
      - backend-net
  # Payment Service
  payment-service:
    build:
      context: ./payment-service
      dockerfile: Dockerfile
    restart: always
    environment:
      NODE_ENV: development
      PORT: 3004
DATABASE_URL:
postgres://payment_user:payment_pass_2026@payment-db:5432/paym
ent_db
    depends_on:
      payment-db:
        condition: service_healthy
    networks:
      - backend-net
networks:
```

```
  backend-net:
    driver: bridge
volumes:
  hostel-db-data:
  resident-db-data:
  checkinout-db-data:
  payment-db-data:
```

Вывод

В ходе выполнения лабораторной работы система была разделена на 4 микро-сервиса, выделенных ранее в ДЗ1. Для каждого микро-сервиса были написаны соответствующие контроллеры, репозитории, описаны сущности и уровень бизнес-логики. Между сервисами настроено сетевое взаимодействие через API Gateway.