МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение
высшего образования
«МИРЭА – Российский технологический университет»

РТУ МИРЭА

Институт информационных технологий (ИИТ)

Кафедра практической и прикладной информатики(ППИ)

ОТЧЕТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ

по дисциплине «Микросервисная архитектура»

Практическое задание № 6

Студент группы *ИКБО-25-20, Валиков Кирилл Юрьевич* _____ подпись)

Преподаватель *Лаптев И.А.*

_____ подпись)

Отчет представлен _____202__г.

1 Предметная область

В качестве предметной области, в рамках которой будут написаны микросервисы, была выбрана следующая тема – “Составление заявок собственников в сфере ЖКХ”.

Было написано 2 микросервиса:

- Микросервис пользователя – микросервис, для создания пользователя, получение данных пользователя по id.
- Микросервис заявок – микросервис для отображения всех заявок, создания заявок, выставления статуса определённой заявки.

2 Микросервис пользователя

```java
package com.example.pr6_firstmicroservice.Models;

import jakarta.persistence.*;
import lombok.*;

9 usages
@Setter
@Getter
@Builder
@NoArgsConstructor
@AllArgsConstructor
@Entity
@Table(
        name = "user",
        schema = "public"
)
public class User {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;

    @Column(nullable = false)
    private String nickname;

}
```

Рисунок 1. User.java

```java
package com.example.pró_firstmicroservice.Repositories;

import com.example.pró_firstmicroservice.Models.User;
import org.springframework.data.jpa.repository.JpaRepository;
import org.springframework.stereotype.Repository;

import java.util.Optional;
2 usages
@Repository
public interface UserRepository extends JpaRepository<User, Long> {

    1 usage
    Optional<User> findByNickname(String Nickname);


}
```

Рисунок 2. UserRepository.java

```java
package com.example.pró_firstmicroservice.Controllers;

import com.example.pró_firstmicroservice.DTO.Appeal;
import com.example.pró_firstmicroservice.Exceptions.BadRequestException;
import com.example.pró_firstmicroservice.Models.User;
import com.example.pró_firstmicroservice.Repositories.UserRepository;
import com.example.pró_firstmicroservice.Services.UserServiceClient;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.*;


import java.util.List;
import java.util.Optional;

no usages
@RestController
public class UserController {

    5 usages
    @Autowired
    private UserRepository userRepository;

    1 usage
    @Autowired
    private UserServiceClient userServiceClient;

    no usages
    @PostMapping("/users/create_appeal")
    public ResponseEntity<String> CreateAppeal(@RequestBody Appeal appeal) {
        userServiceClient.CreateAppeal(appeal);
        return ResponseEntity.ok().body("Всё прошло успешно!");
    }


```

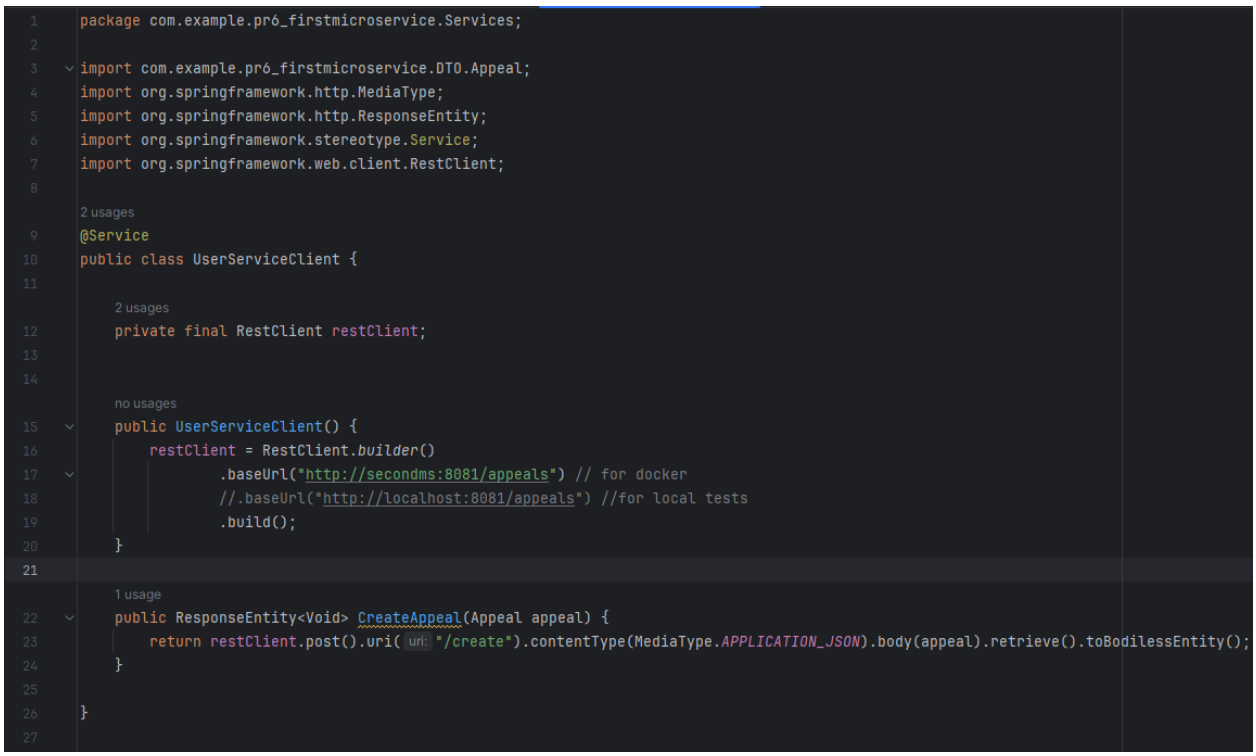
Рисунок 3. UserContoller.java (1)

```
    no usages
    @PostMapping("/users/create")
    public ResponseEntity<?> CreateUser(@RequestBody User userfordb) {
        userRepository.findByNickname(userfordb.getNickname()).ifPresentOrElse(user -> {
            throw new BadRequestException(String.format("Пользователь с таким никнеймом '%s' уже существует!",
                    userfordb.getNickname()));
        }, () -> {
            userRepository.save(userfordb);
        });
        return ResponseEntity.ok().body(userfordb);
    }

    no usages
    @GetMapping("/users/{id}")
    public ResponseEntity<?> GetUserById(@PathVariable Long id) {
        if(userRepository.findById(id).isPresent()) {
            Optional<User> userfromdb = userRepository.findById(id);
            return ResponseEntity.ok().body(userfromdb);
        } else {
            return ResponseEntity.ok().body("Пользователя с таким id = " + id + " не существует!");
        }
    }

    no usages
    @GetMapping("/users/get_all")
    public ResponseEntity<?> GetAllUsers() {
        List<User> all_users = userRepository.findAll();
        return ResponseEntity.ok().body(all_users);
    }

}
```

Рисунок 4. UserContoller.java (2)

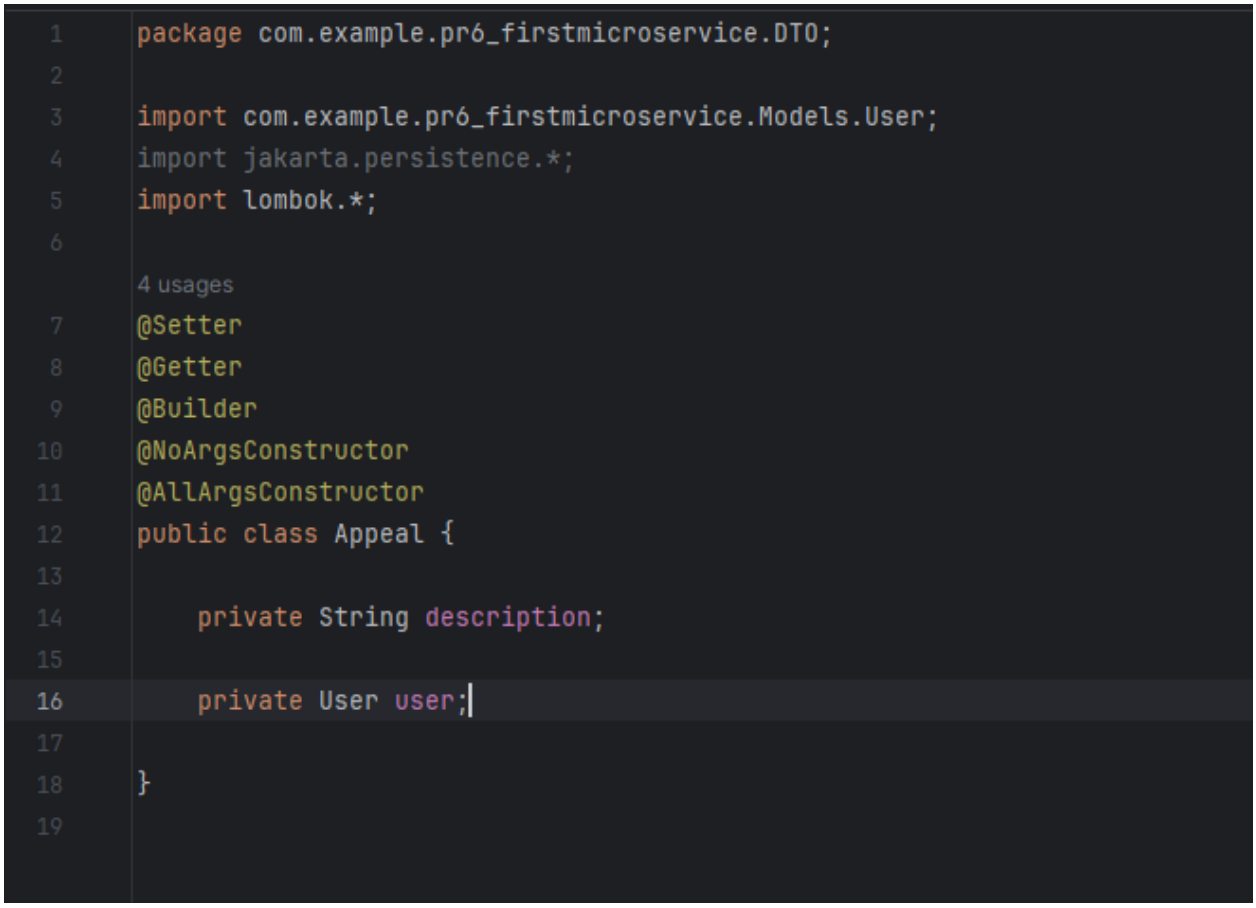
```
package com.example.pró_firstmicroservice.Exceptions;

import org.springframework.http.HttpStatus;
import org.springframework.web.bind.annotation.ResponseStatus;

2 usages
@ResponseStatus(HttpStatus.BAD_REQUEST)
public class BadRequestException extends RuntimeException {
    1 usage
    public BadRequestException (String message) {
        super(message);
    }
}
```

Рисунок 5. BadRequestException.java



```java
package com.example.pr6_firstmicroservice.Services;

import com.example.pr6_firstmicroservice.DTO.Appeal;
import org.springframework.http.MediaType;
import org.springframework.http.ResponseEntity;
import org.springframework.stereotype.Service;
import org.springframework.web.client.RestClient;

2 usages
@Service
public class UserServiceClient {

    2 usages
    private final RestClient restClient;


    no usages
    public UserServiceClient() {
        restClient = RestClient.builder()
                .baseUrl("http://secondms:8081/appeals") // for docker
                //.baseUrl("http://localhost:8081/appeals") //for local tests
                .build();
    }

    1 usage
    public ResponseEntity<Void> CreateAppeal(Appeal appeal) {
        return restClient.post().uri( uri: "/create").contentType(MediaType.APPLICATION_JSON).body(appeal).retrieve().toBodilessEntity();
    }

}
```

Рисунок 6. UserServiceClient.java



```java
package com.example.pr6_firstmicroservice.DTO;

import com.example.pr6_firstmicroservice.Models.User;
import jakarta.persistence.*;
import lombok.*;

4 usages
@Setter
@Getter
@Builder
@NoArgsConstructor
@AllArgsConstructor
public class Appeal {

    private String description;

    private User user;

}
```

Рисунок 7. AppealDTO.java

3 Микросервис заявок

```java
package com.example.pr6_secondmicroservice.Models;


import com.fasterxml.jackson.annotation.JsonIgnore;
import jakarta.persistence.*;
import lombok.*;

10 usages
@Setter
@Getter
@Builder
@NoArgsConstructor
@AllArgsConstructor
@Entity
@Table(
        name = "appeal",
        schema = "public"
)
public class Appeal {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;

    @Column(nullable = false)
    private String description;

    @ManyToOne
    private UserDTO user;

    @Enumerated(EnumType.STRING)
    @Column(name = "status", nullable = false)
    private Status status;

}
```

Рисунок 8. Appeal.java

```java
package com.example.pr6_secondmicroservice.Repositories;

import ...

2 usages
@Repository
public interface AppealRepository extends JpaRepository<Appeal, Long> {

}
```

Рисунок 9. AppealRepository.java

```java
package com.example.pró_secondmicroservice.Controllers;

import com.example.pró_secondmicroservice.Models.Appeal;
import com.example.pró_secondmicroservice.Models.Status;
import com.example.pró_secondmicroservice.Models.UserDTO;
import com.example.pró_secondmicroservice.Repositories.AppealRepository;
import com.example.pró_secondmicroservice.Services.AppealServiceClient;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.*;

import java.util.List;
import java.util.Optional;

no usages
@RestController
public class AppealController {

    8 usages
    @Autowired
    private AppealRepository appealRepository;

    1 usage
    @Autowired
    private AppealServiceClient appealServiceClient;

    no usages
    @PutMapping("appeals/{id}")
    public ResponseEntity<?> UpdateStatus (@PathVariable Long id, @RequestBody Appeal appeal) {
        Appeal appealfordb = new Appeal(appealRepository.findById(id).get().getId(),
                appealRepository.findById(id).get().getDescription(),
                appealRepository.findById(id).get().getUser(),
                appeal.getStatus());
        appealRepository.save(appealfordb);
        return ResponseEntity.ok().body("Статус заявки успешно обновлён");
    }
```

Рисунок 10. AppealContoller.java (1)

```java
    @PostMapping("/appeals/create")
    public ResponseEntity<String> CreateAppeal (@RequestBody Appeal appeal) {
        appeal.setStatus(Status.ACCEPTED_FOR_WORK);
        UserDTO userDTO = appealServiceClient.GetUserById(appeal.getUser().getId());
        if (!(userDTO == null)) {
            appealRepository.save(appeal);
            return ResponseEntity.ok().body("Ваша заявка успешно сохранена!");
        } else {
            return ResponseEntity.ok().body("Пользователя с таким ID не существует!");
        }
    }

    no usages
    @GetMapping("/appeals/{id}")
    public ResponseEntity<?> GetAppealById(@PathVariable Long id) {
        if(appealRepository.findById(id).isPresent()) {
            Optional<Appeal> appealfromdb = appealRepository.findById(id);
            return ResponseEntity.ok().body(appealfromdb);
        } else {
            return ResponseEntity.ok().body("Заявки с таким id = " + id + " не существует!");
        }
    }

    no usages
    @GetMapping("/appeals/get_all")
    public ResponseEntity<?> GetAllAppeal() {
        List<Appeal> all_appeals = appealRepository.findAll();
        return ResponseEntity.ok().body(all_appeals);
    }
}
```

Рисунок 11. AppealContoller.java (2)

```java
package com.example.pró_secondmicroservice.Exceptions;

import org.springframework.http.HttpStatus;
import org.springframework.web.bind.annotation.ResponseStatus;

no usages
@ResponseStatus(HttpStatus.BAD_REQUEST)
public class BadRequestException extends RuntimeException {
    no usages
    public BadRequestException (String message) { super(message); }
}
```

Рисунок 12. BadRequestException.java

```java
package com.example.pró_secondmicroservice.Services;

import com.example.pró_secondmicroservice.Models.UserDTO;
import org.springframework.stereotype.Service;
import org.springframework.web.client.RestClient;

import java.util.List;

2 usages
@Service
public class AppealServiceClient {

    2 usages
    private final RestClient restClient;


    no usages
    public AppealServiceClient() {
        restClient = RestClient.builder()
                .baseUrl("http://firstms:8080/users") //for docker
                //.baseUrl("http://localhost:8080/users") //for local tests
                .build();
    }

    1 usage
    public UserDTO GetUserById(Long id) { return restClient.get().uri( uri: "/" + id).retrieve().body(UserDTO.class); }
}
```

Рисунок 13. AppealServiceClient.java

```java
package com.example.pr6_secondmicroservice.Models;

import com.fasterxml.jackson.annotation.JsonIgnore;
import jakarta.persistence.*;
import lombok.*;

import java.util.HashSet;
import java.util.Set;

6 usages
@Setter
@Getter
@Builder
@NoArgsConstructor
@AllArgsConstructor
@Entity

@Table(
        name = "user",
        schema = "public"
)
public class UserDTO {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;

    @Column(nullable = false)
    private String nickname;

    @OneToMany(mappedBy="user",cascade = CascadeType.ALL, fetch = FetchType.LAZY)
    @JsonIgnore
    private Set<Appeal> appeals = new HashSet<>();

}
```

Рисунок 14. UserDTO.java

```java
package com.example.pr6_secondmicroservice.Models;

3 usages
public enum Status {

    1 usage
    ACCEPTED_FOR_WORK,
    no usages
    IN_WORK,
    no usages
    DONE

}
```

Рисунок 15. Status.java

4 Docker

```
FROM eclipse-temurin:21-jdk-jammy
ARG JAR_FILE=target/*.jar
COPY ${JAR_FILE} pr6_first-microservice-0.0.1-SNAPSHOT.jar
EXPOSE 8080
ENTRYPOINT ["java","-jar","/pr6_first-microservice-0.0.1-SNAPSHOT.jar"]
```

Рисунок 16. Dockerfile – Микросервис пользователя

```
FROM eclipse-temurin:21-jdk-jammy
ARG JAR_FILE=target/*.jar
COPY ${JAR_FILE} pr6_second-microservice-0.0.1-SNAPSHOT.jar
EXPOSE 8081
ENTRYPOINT ["java","-jar","/pr6_second-microservice-0.0.1-SNAPSHOT.jar"]
```

Рисунок 17. Dockerfile – Микросервис заявок

docker-compose.yml

C: > Users > Kvali > Downloads > university > Микросервисы > Мои практики > pr6 > docker-compose.yml

```
version: '3.8'
services:
  firstms:
    container_name: firstms
    image: first_ms
    build:
      context: C:\Users\Kvali\Downloads\university\Микросервисы\Мои практики\pr6\pr6_first-microservice
      dockerfile: Dockerfile
    ports:
      - 8080:8080
    depends_on:
      - secondms
    environment:
      - SPRING_DATASOURCE_URL=jdbc:postgresql://db/postgres
  secondms:
    container_name: secondms
    image: second_ms
    build:
      context: C:\Users\Kvali\Downloads\university\Микросервисы\Мои практики\pr6\pr6_second-microservice
      dockerfile: Dockerfile
    ports:
      - 8081:8081
    depends_on:
      - db
    environment:
      - SPRING_DATASOURCE_URL=jdbc:postgresql://db/postgres
  db:
    image: postgres:14.7-alpine
    ports:
      - 5433:5432
    environment:
      POSTGRES_USER: postgres
      POSTGRES_PASSWORD: Hellomydearfriend666

```

Рисунок 18. docker-compose.yml

5 Проверка

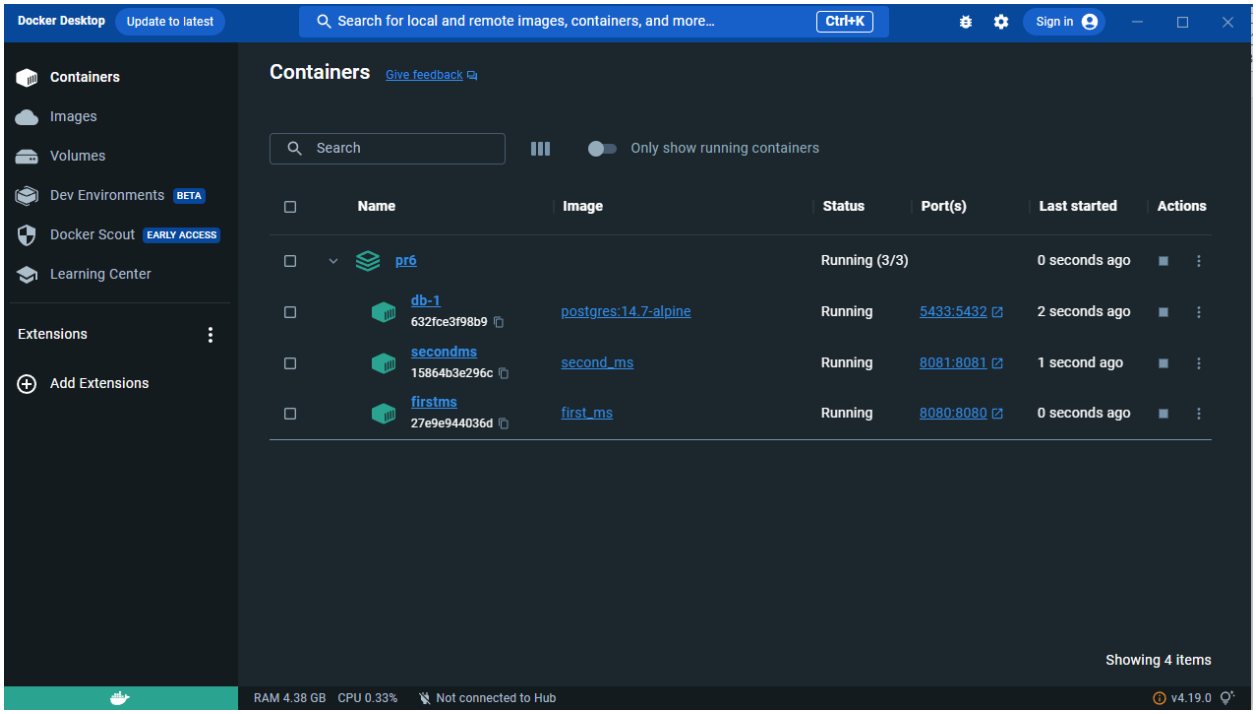


Рисунок 19. Запущенные контейнеры

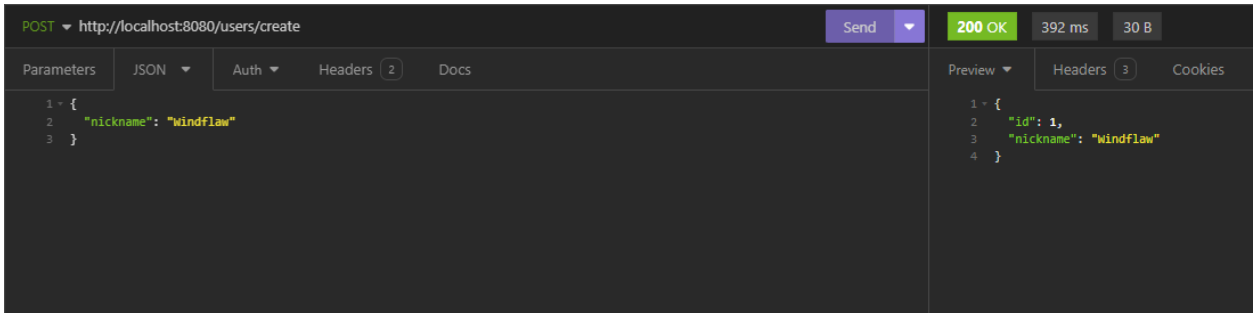


Рисунок 20. Проверка создания пользователя (1)

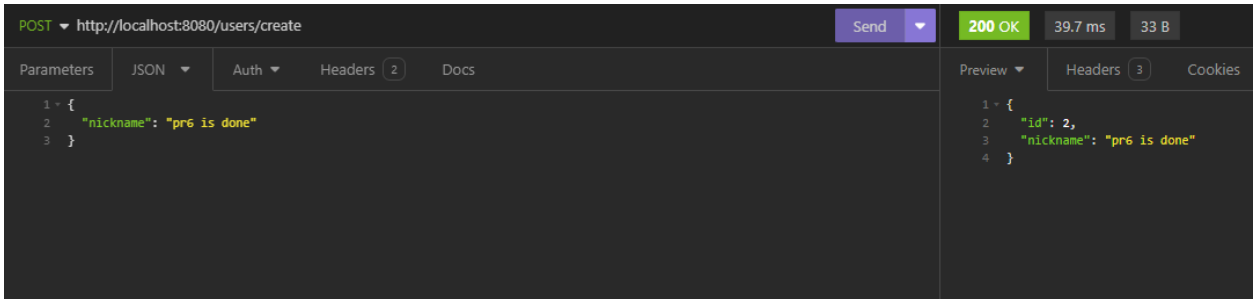


Рисунок 21. Проверка создания пользователя (2)

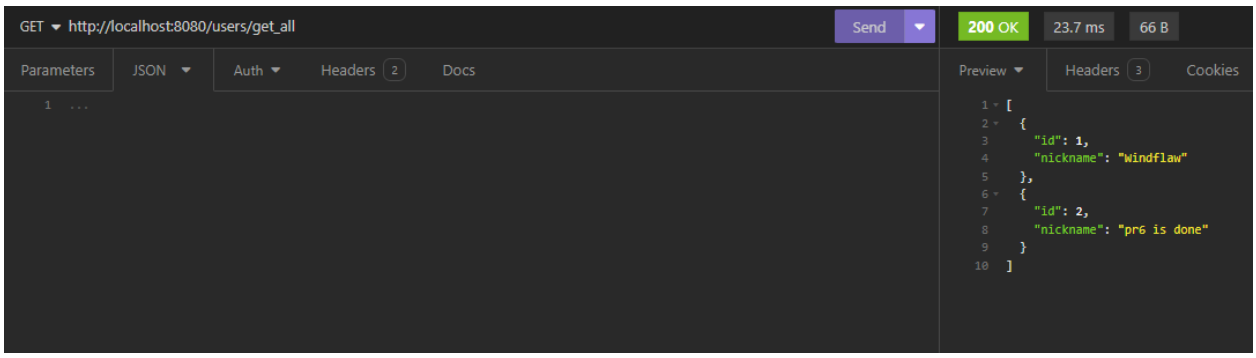


Рисунок 22. Проверка получения всех пользователей

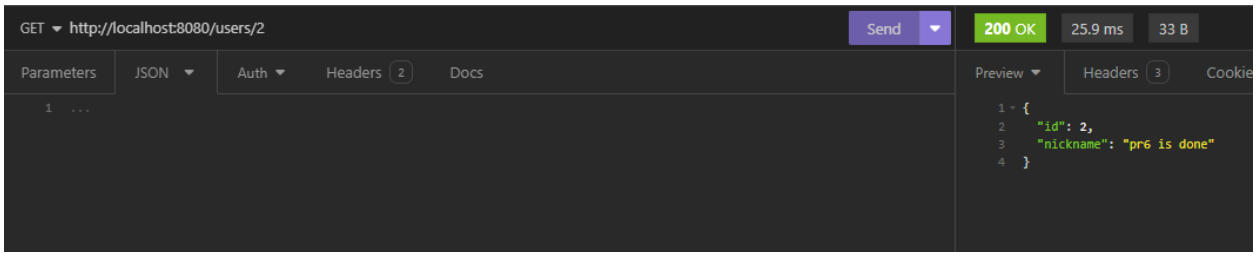


Рисунок 23. Проверка получения пользователя по id

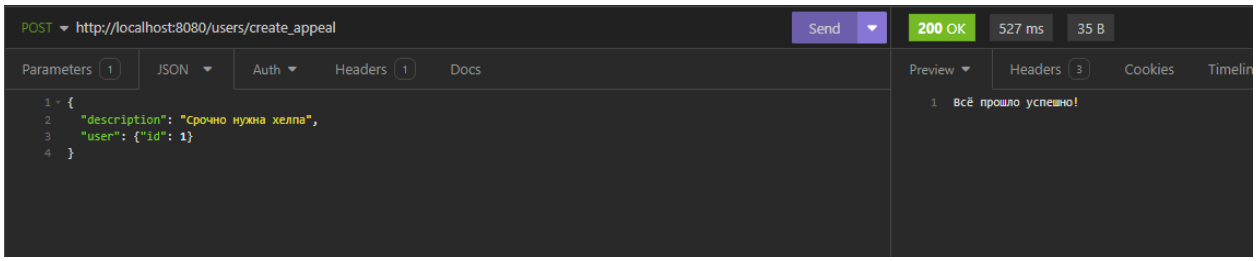


Рисунок 24. Создание заявки через микросервис пользователя (1)

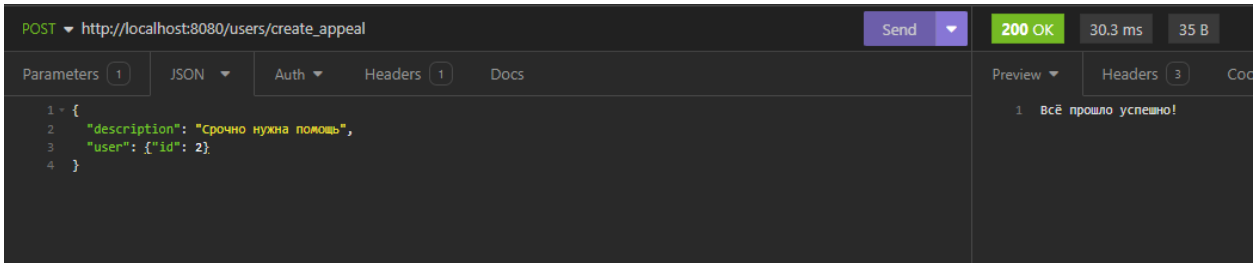


Рисунок 25. Создание заявки через микросервис пользователя (2)

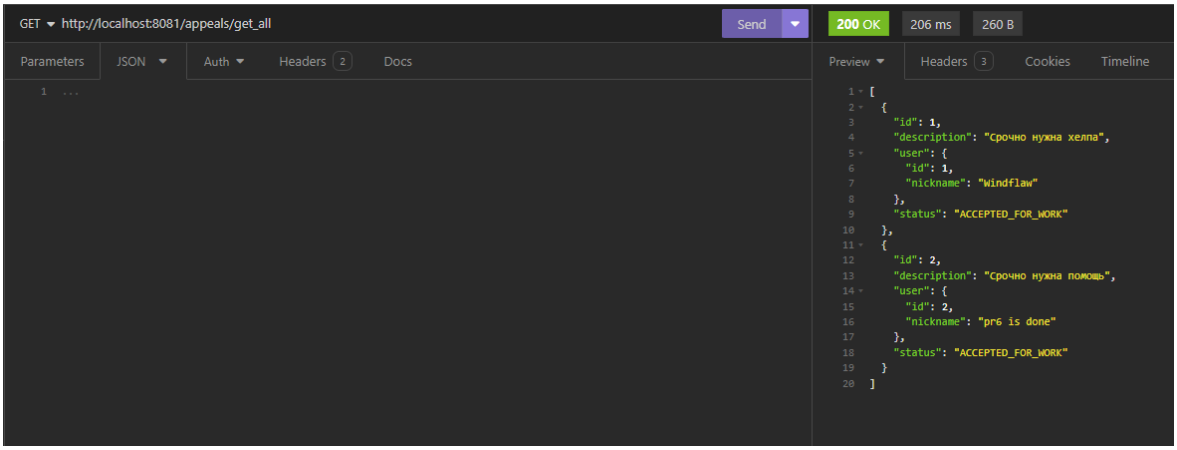


Рисунок 26. Вывод всех заявок