2Министерство науки и высшего образования Российской Федерации

Муромский институт (филиал)

Федерального государственного бюджетного образовательного учреждения высшего образования

«Владимирский государственный университет   
имени Александра Григорьевича и Николая Григорьевича Столетовых»

Факультет ИТР

Кафедра ПИн

ЛАБОРАТОРНАЯ

РАБОТА №5

По Разработка корпоративных приложений

Тема Построение REST API

Руководитель

Кульков Я.Ю.

(фамилия, инициалы)

(подпись) (дата)

Студент ПИН - 121

(группа)

Ермилов М.В.

(фамилия, инициалы)

(подпись) (дата)

Муром 2024

Лабораторная работа №5

Тема: построение REST API.

Цели и задачи: создание Spring-приложения с несколькими таблицами с использованием REST API.

Ход работы: задание: создать набор сущностей в зависимости от варианта. Реализовать REST API с основными CRUD операциями. Протестировать разработанные методы. Задание для варианта 7: сущность – Военнослужащий; атрибуты : фамилия; имя; отчество; национальность; дата рождения (год, месяц число); должность; звание.

Листинг кода 1 – SoldierController.java

package ru.kafpin.firsttt.controllers;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.web.bind.annotation.\*;

import ru.kafpin.firsttt.dto.PostDTO;

import ru.kafpin.firsttt.dto.SoldierDTO;

import ru.kafpin.firsttt.entities.Post;

import ru.kafpin.firsttt.entities.Soldier;

import ru.kafpin.firsttt.services.PostService;

import ru.kafpin.firsttt.services.SoldierService;

import java.time.LocalDate;

@RequestMapping("/api")

@RestController

public class SoldierController {

private SoldierService soldierService;

private PostService postService;

@Autowired

public void setSoldierService(SoldierService soldierService){

this.soldierService = soldierService;

}

@Autowired

public void setPostService(PostService postService){

this.postService = postService;

}

@GetMapping("/soldiers")

public Iterable<Soldier> listOfSoldiers(){

return soldierService.getAllSoldiers();

}

@GetMapping("/posts")

public Iterable<Post> listOfPosts(){

return postService.getAllPosts();

}

@GetMapping("/soldier/{id}")

public Soldier getSoldier(@PathVariable("id") Long id){

return soldierService.getSoldierById(id);

}

@GetMapping("/post/{id}")

public Post getPost(@PathVariable("id") Long id){

return postService.getPostById(id);

}

@PostMapping("/soldier/add")

public Soldier newSoldier(@RequestBody SoldierDTO soldierDTO){

Post post = postService.getPostByTitle(soldierDTO.getPost());

Soldier soldier = new Soldier();

soldier.setSurname(soldierDTO.getSurname());

soldier.setName(soldierDTO.getName());

soldier.setPatronym(soldierDTO.getPatronym());

soldier.setNation(soldierDTO.getNation());

soldier.setBirthday(LocalDate.parse(soldierDTO.getBirthday()));

soldier.setRank(soldierDTO.getRank());

soldier.setPost(post);

return soldierService.addEditSoldier(soldier);

}

@PostMapping("/post/add")

public Post newPost(@RequestBody PostDTO postDTO){

Post post = new Post();

post.setTitle(postDTO.getTitle());

return postService.addEditPost(post);

}

@PutMapping("/soldier/{id}")

public Soldier updateSoldier(@RequestBody SoldierDTO soldierDTO,

@PathVariable("id") Long id){

Soldier existSoldier = soldierService.getSoldierById(id);

Post post = postService.getPostByTitle(soldierDTO.getPost());

Soldier soldier = new Soldier();

soldier.setId(existSoldier.getId());

soldier.setSurname(soldierDTO.getSurname());

soldier.setName(soldierDTO.getName());

soldier.setPatronym(soldierDTO.getPatronym());

soldier.setNation(soldierDTO.getNation());

soldier.setBirthday(LocalDate.parse(soldierDTO.getBirthday()));

soldier.setRank(soldierDTO.getRank());

soldier.setPost(post);

return soldierService.addEditSoldier(soldier);

}

@PutMapping("/post/{id}")

public Post updatePost(@RequestBody PostDTO postDTO,

@PathVariable("id") Long id){

Post existPost = postService.getPostById(id);

Post post = new Post();

post.setId(existPost.getId());

post.setTitle(postDTO.getTitle());

return postService.addEditPost(post);

}

@DeleteMapping("/soldier/{id}")

public void deleteSoldier(@PathVariable("id") Long id){

soldierService.deleteSoldierById(id);

}

@DeleteMapping("/post/{id}")

public void deletePost(@PathVariable("id") Long id){

postService.deletePostById(id);

}

}

Листинг кода 2 – Soldier.java

package ru.kafpin.firsttt.entities;

import jakarta.persistence.\*;

import jakarta.validation.constraints.NotBlank;

import jakarta.validation.constraints.Size;

import lombok.Data;

import lombok.NoArgsConstructor;

import org.springframework.format.annotation.DateTimeFormat;

import java.time.LocalDate;

@Entity

@Table(name = "soldiers")

@Data

@NoArgsConstructor

public class Soldier {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

@Column(name = "id")

private Long id;

@Column(name = "surname", nullable = false)

@NotBlank(message = "Поле \"Фамилия\" не может быть пустым")

@Size(min = 2, max=30, message = "Допустимая длина фамилии от 2ух до 30ти символов")

private String surname;

@Column(name = "name", nullable = false)

@NotBlank(message = "Поле \"Имя\" не может быть пустым")

@Size(min = 2, max=30, message = "Допустимая длина имени от 2ух до 30ти символов")

private String name;

@Column(name = "patronym", nullable = true)

private String patronym;

@Column(name = "nation", nullable = false)

@NotBlank (message = "Поле \"Национальность\" не может быть пустым")

private String nation;

@DateTimeFormat(pattern="yyyy-MM-dd")

@Column(name = "birthday", nullable = false)

private LocalDate birthday;

@NotBlank(message = "Поле \"Звание\" не может быть пустым")

@Column(name = "rank", nullable = false)

private String rank;

@ManyToOne

@JoinColumn(name = "post\_id")

private Post post;

public Soldier(Long id, String surname, String name, String patronym, String nation, LocalDate birthday, String rank, Post post) {

this.id = id;

this.surname = surname;

this.name = name;

this.patronym = patronym;

this.nation = nation;

this.birthday = birthday;

this.rank = rank;

this.post = post;

}

}

Листинг кода 3 – SoldierRepository.java

package ru.kafpin.firsttt.repositories;

import org.springframework.data.repository.CrudRepository;

import org.springframework.stereotype.Repository;

import ru.kafpin.firsttt.entities.Soldier;

import java.util.List;

@Repository

public interface SoldierRepository extends CrudRepository<Soldier, Long> {

List<Soldier> findByPostTitleContainsIgnoreCase(String title);

}

Листинг кода 4 – SoldierService.java

package ru.kafpin.firsttt.services;

import jakarta.persistence.EntityNotFoundException;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Service;

import ru.kafpin.firsttt.entities.Soldier;

import ru.kafpin.firsttt.repositories.SoldierRepository;

import java.util.List;

import java.util.Optional;

@Service

public class SoldierService {

@Autowired

private SoldierRepository soldierRepository;

@Autowired

public void setSoldierRepository(SoldierRepository soldierRepository) {

this.soldierRepository = soldierRepository;

}

public List<Soldier> getAllSoldiers() {

return (List<Soldier>) soldierRepository.findAll();

}

public List<Soldier> getAllSoldiersByPost(String title) {

return (List<Soldier>) soldierRepository.findByPostTitleContainsIgnoreCase(title);

}

public Soldier getSoldierById(Long id) {

Optional<Soldier> soldier = soldierRepository.findById(id);

if(soldier.isEmpty())

throw new EntityNotFoundException("Военнослужащий не найден");

return soldier.get();

}

public void deleteSoldierById(Long id) {

if (soldierRepository.existsById(id))

soldierRepository.deleteById(id);

}

public Soldier addEditSoldier(Soldier soldier) {

return soldierRepository.save(soldier);

}

}

Листинг кода 5 – SoldierDTO.java

package ru.kafpin.firsttt.dto;

import lombok.Data;

import lombok.NoArgsConstructor;

@Data

@NoArgsConstructor

public class SoldierDTO {

private String surname;

private String name;

private String patronym;

private String nation;

private String birthday;

private String rank;

private String post;

}

Листинг кода 6 – Post.java

package ru.kafpin.firsttt.entities;

import com.fasterxml.jackson.annotation.JsonBackReference;

import jakarta.persistence.\*;

import jakarta.validation.constraints.NotBlank;

import jakarta.validation.constraints.Size;

import lombok.Data;

import lombok.NoArgsConstructor;

import java.util.List;

@Entity

@Table(name = "posts")

@Data

@NoArgsConstructor

public class Post {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

@Column(name = "id")

private Long id;

@NotBlank (message = "Поле \"Название\" не может быть пустым")

@Size (min = 5, message = "Короткое название должности")

@Column(name = "title", nullable = false)

private String title;

@OneToMany(mappedBy = "post", cascade = CascadeType.ALL)

@JsonBackReference

private List<Soldier> soldiers;

}

Листинг кода 7 – PostRepository.java

package ru.kafpin.firsttt.repositories;

import org.springframework.data.repository.CrudRepository;

import org.springframework.stereotype.Repository;

import ru.kafpin.firsttt.entities.Post;

@Repository

public interface PostRepository extends CrudRepository<Post, Long> {

Post findByTitle(String title);

}

Листинг кода 8 – PostService.java

package ru.kafpin.firsttt.services;

import jakarta.persistence.EntityNotFoundException;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Service;

import ru.kafpin.firsttt.entities.Post;

import ru.kafpin.firsttt.repositories.PostRepository;

import java.util.List;

import java.util.Optional;

@Service

public class PostService {

@Autowired

private PostRepository postRepository;

@Autowired

public void setSoldierRepository(PostRepository postRepository) {

this.postRepository = postRepository;

}

public List<Post> getAllPosts() {

return (List<Post>) postRepository.findAll();

}

public Post getPostById(Long id) {

Optional<Post> post = postRepository.findById(id);

if(post.isEmpty()){

throw new EntityNotFoundException("Должность не найдена");

}

return post.get();

}

public Post getPostByTitle(String title){

return postRepository.findByTitle(title);

}

public void deletePostById(Long id) {

if (postRepository.existsById(id))

postRepository.deleteById(id);

}

public Post addEditPost(Post post) {

return postRepository.save(post);

}

}

Листинг кода 9 – PostDTO.java

package ru.kafpin.firsttt.dto;

import lombok.Data;

import lombok.NoArgsConstructor;

@Data

@NoArgsConstructor

public class PostDTO {

private String title;

}

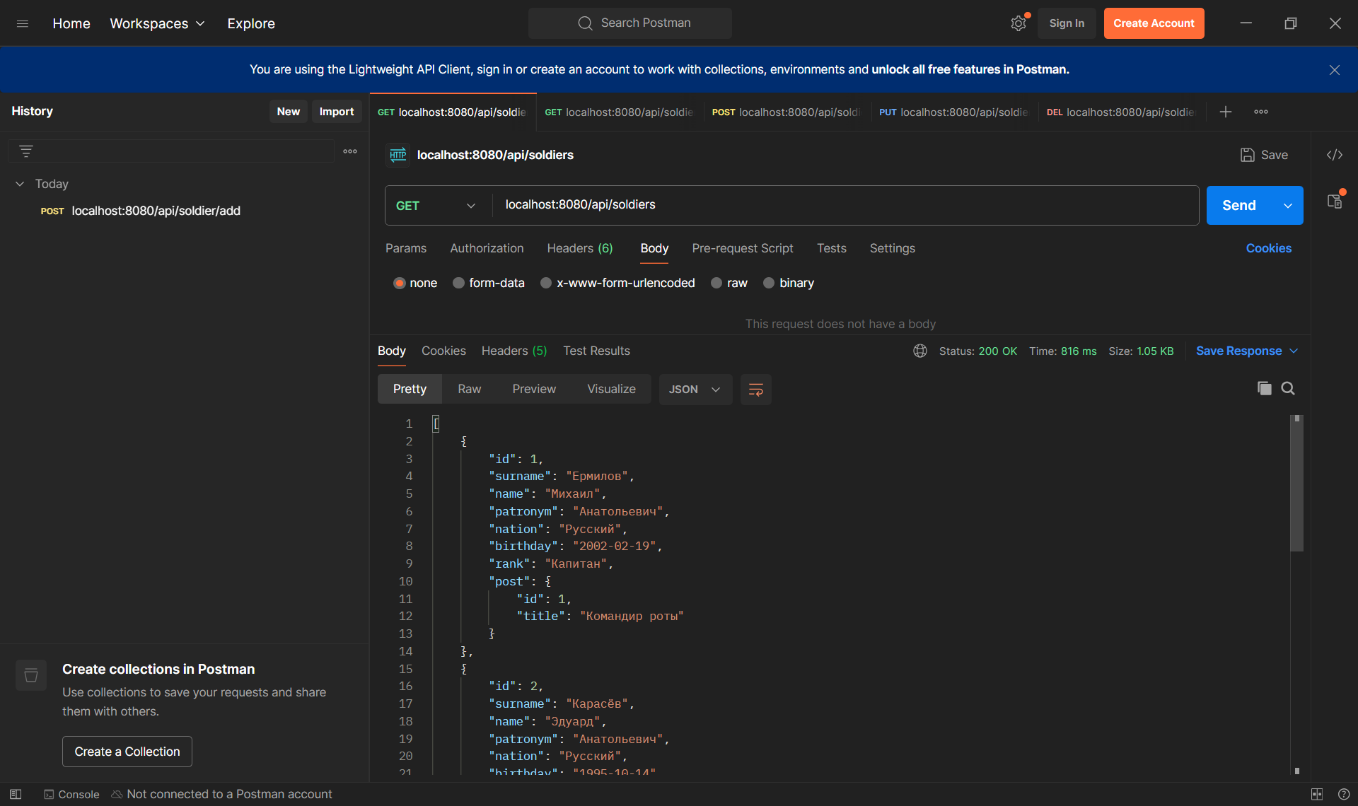


Рисунок 1 – Запрос на получение списка военнослужащих

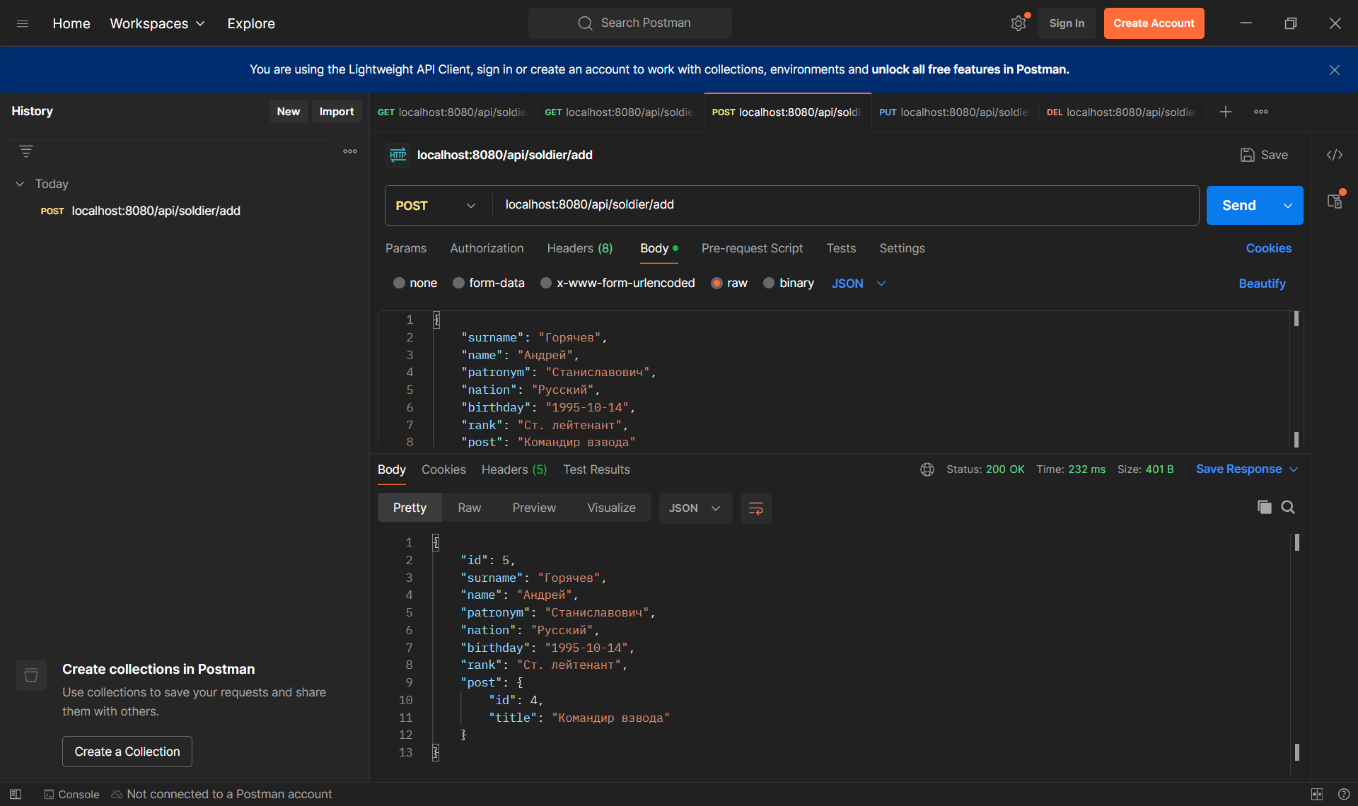


Рисунок 2 – Запрос на добавление военнослужащего

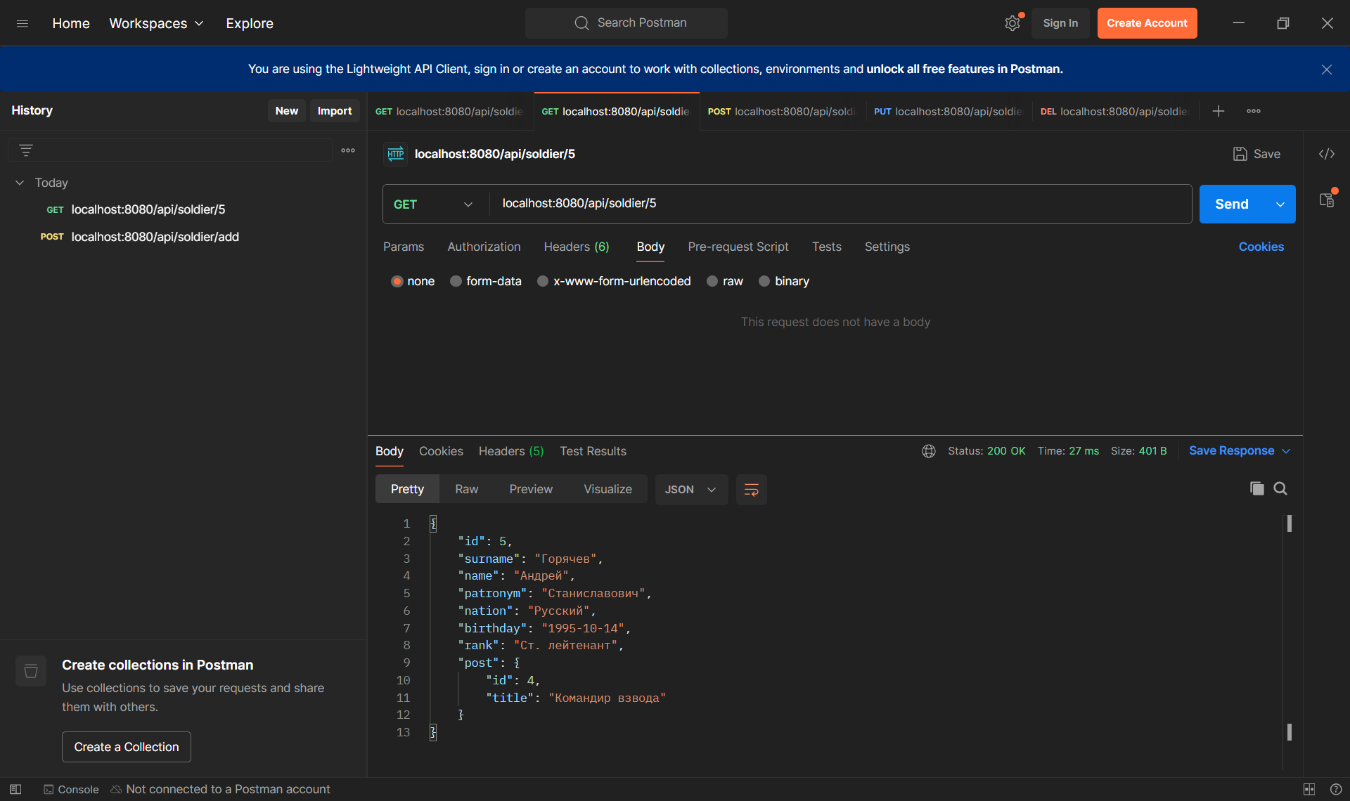


Рисунок 3 – Запрос на получение военнослужащего по id

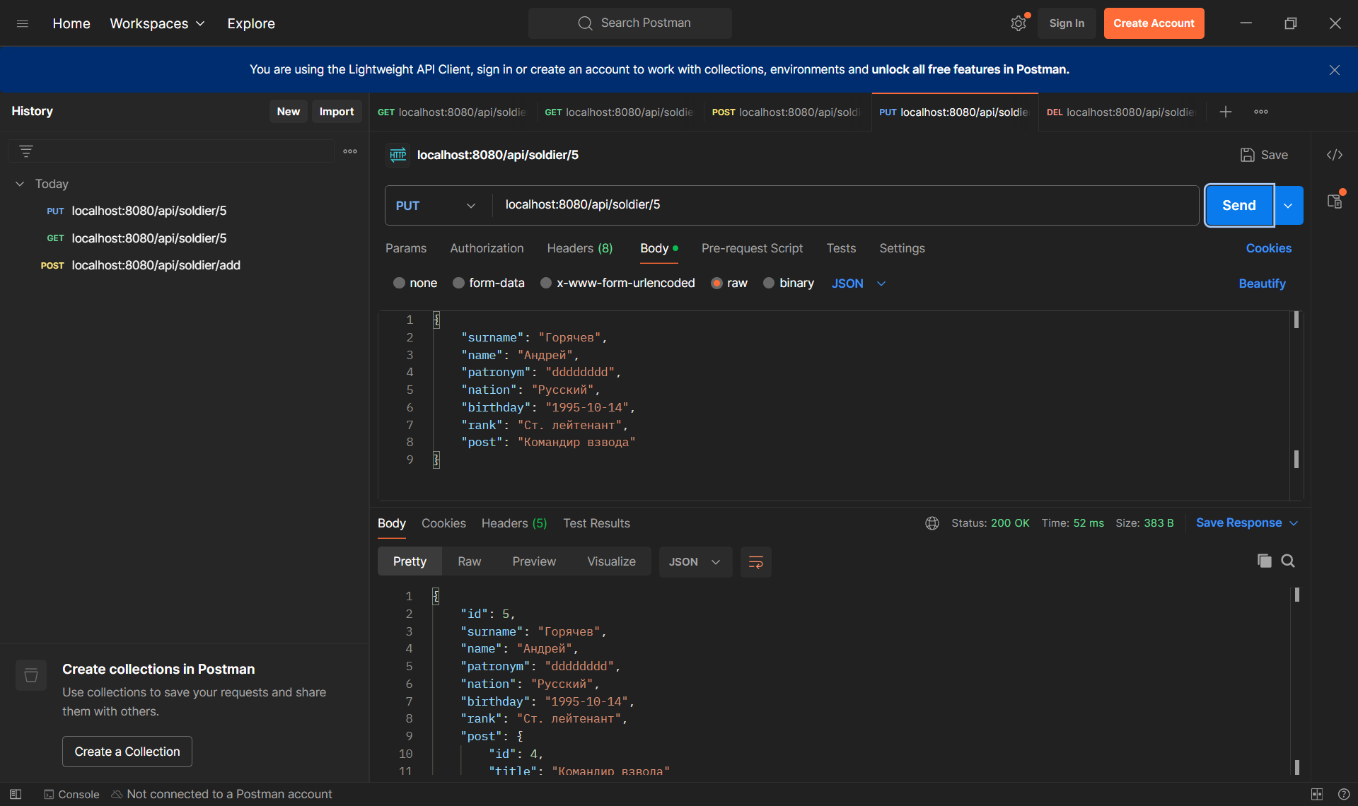


Рисунок 4 – Запрос на изменение военнослужащего

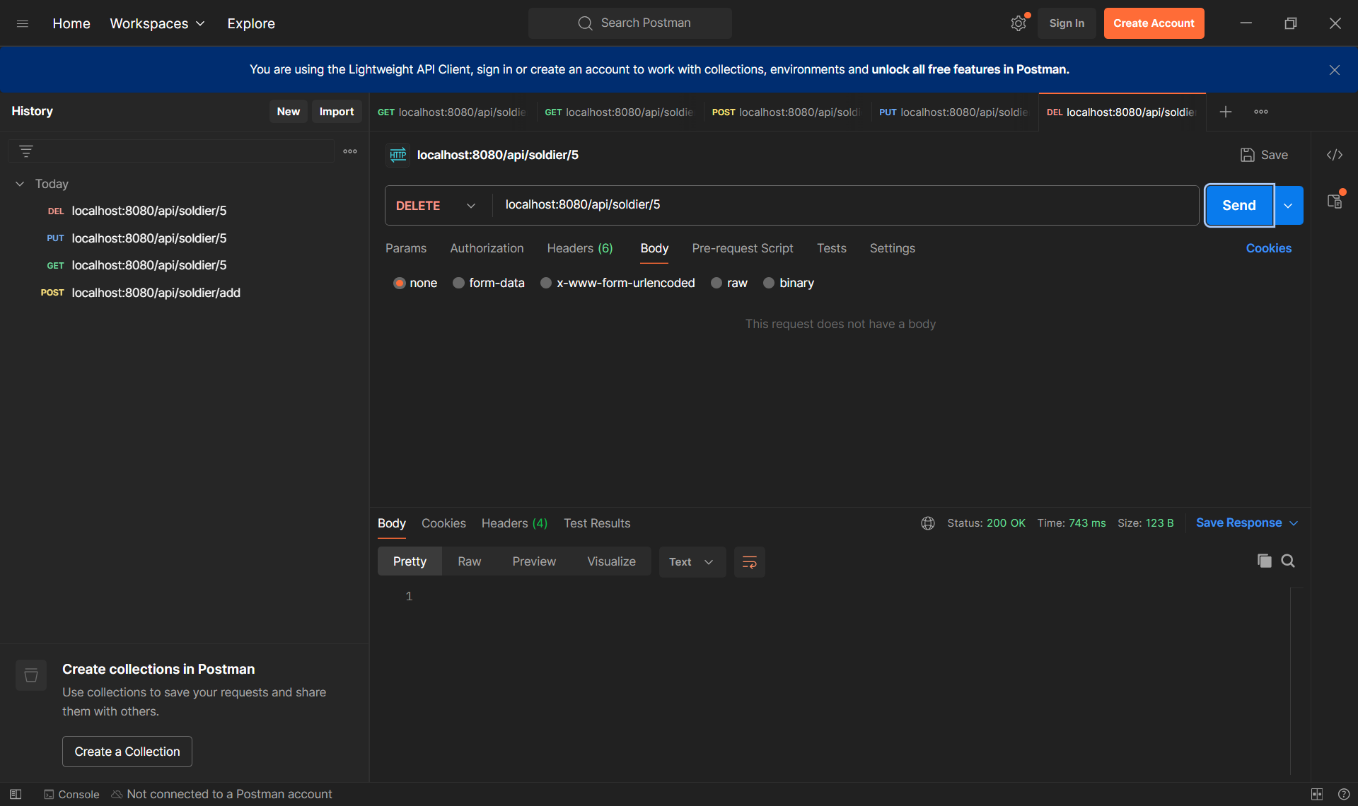


Рисунок 5 – Запрос на удаление военнослужащего

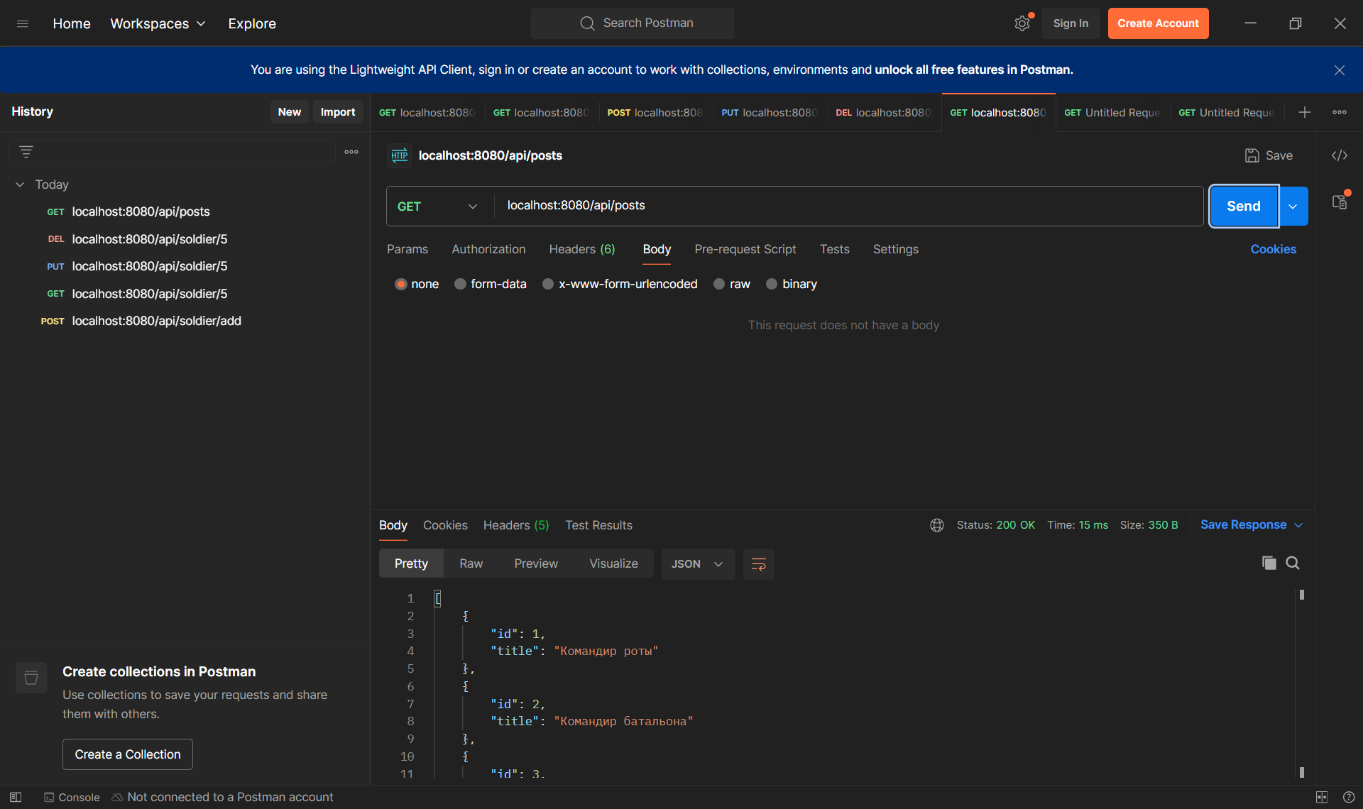


Рисунок 6 – Запрос на получение списка должностей

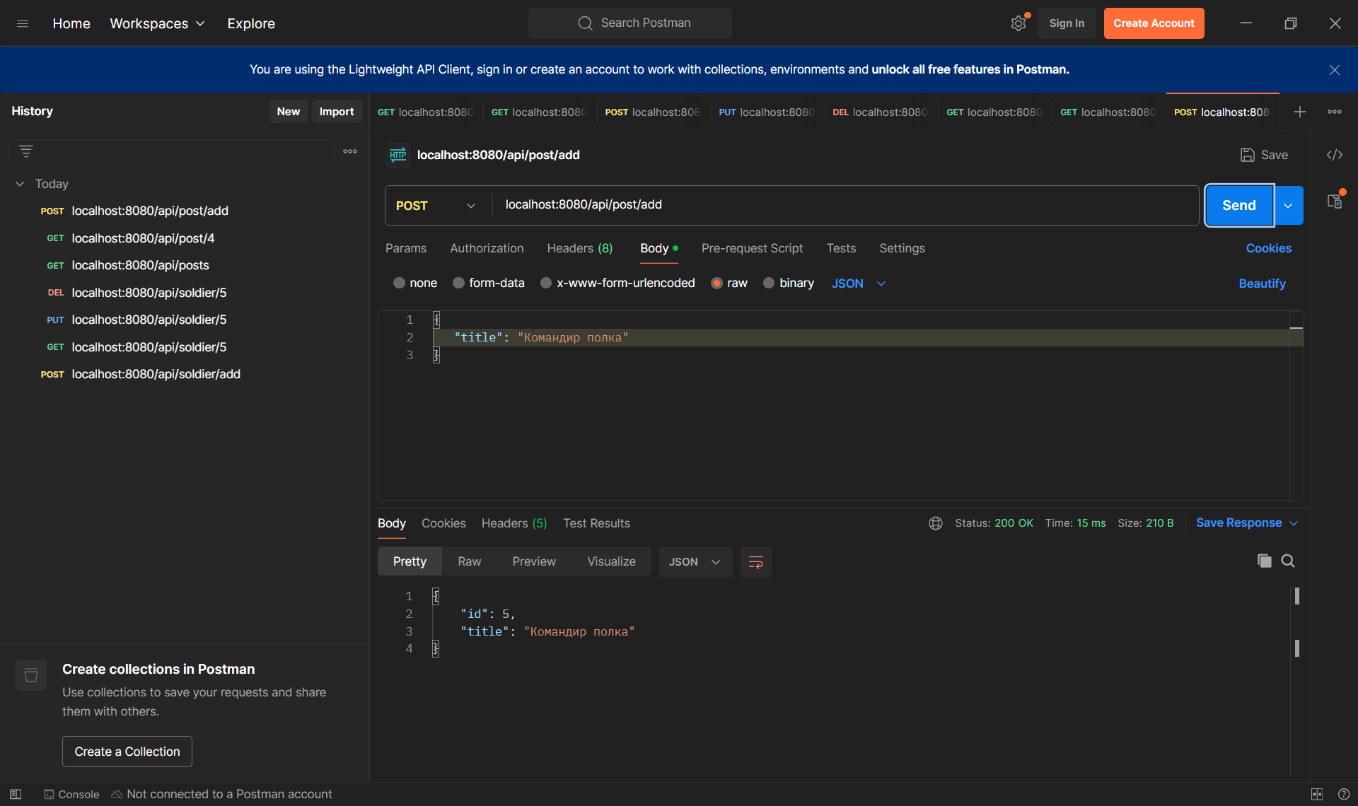


Рисунок 7 – Запрос на добавление должности

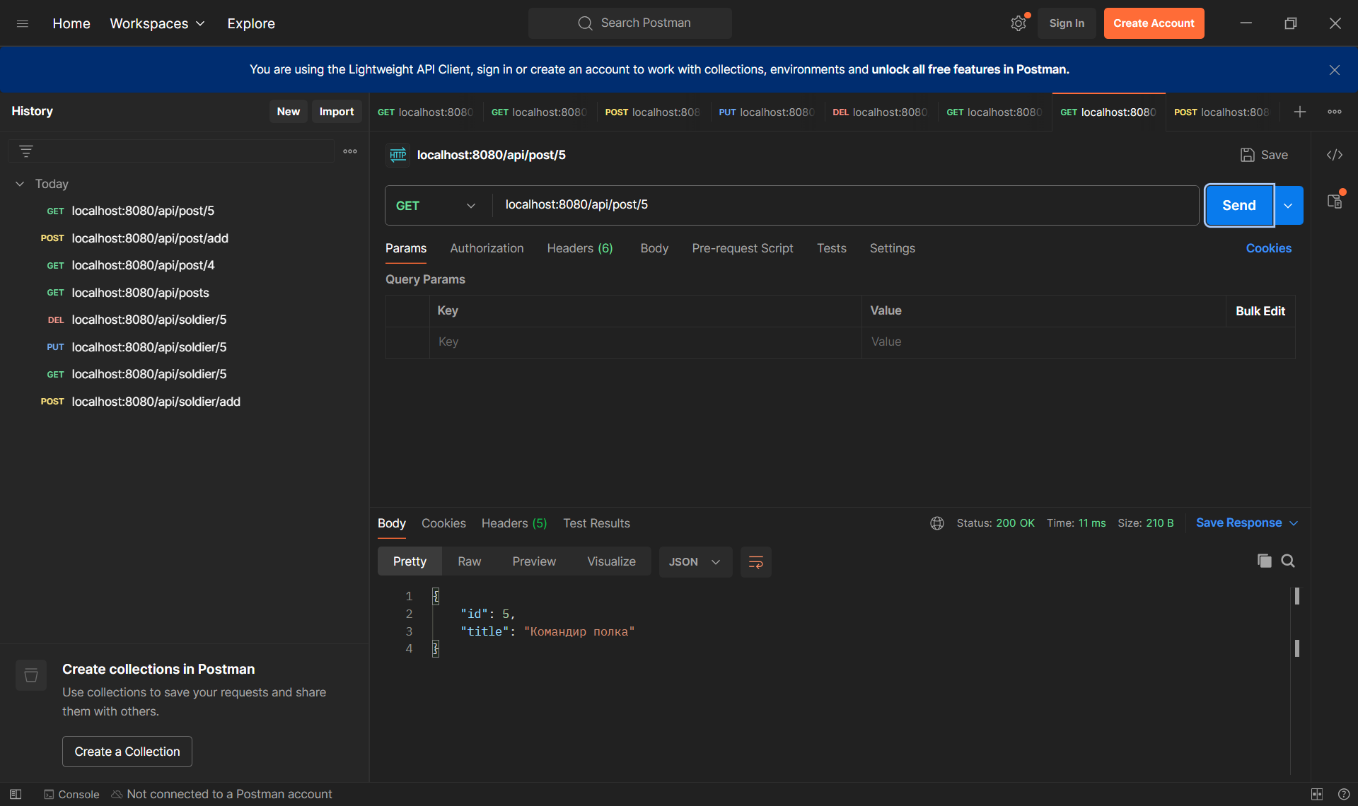


Рисунок 8 – Запрос на получение должности по id

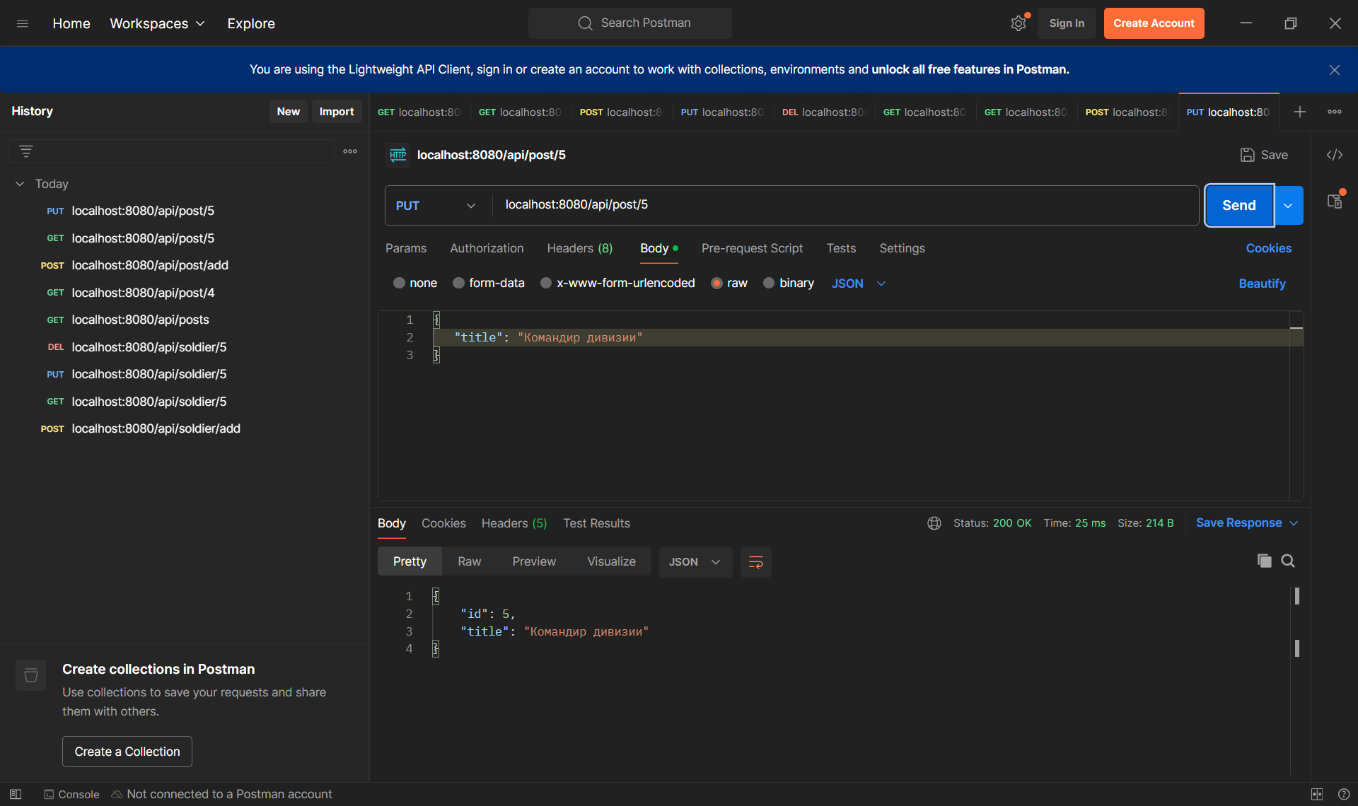


Рисунок 9 – Запрос на изменение должности

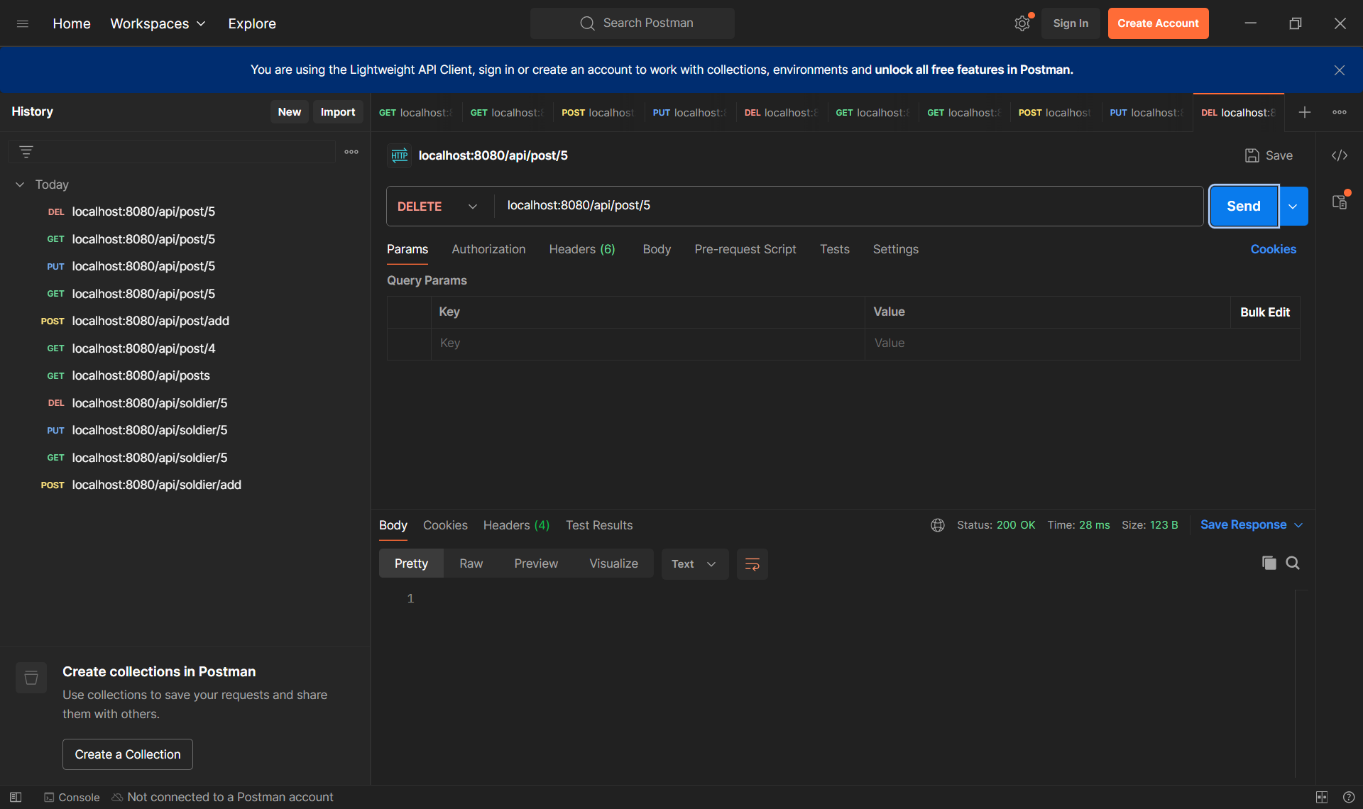


Рисунок 10 – Запрос на удаление должности

Вывод: в ходе работы создали Spring-приложение с несколькими таблицами с использованием REST API.