Министерство 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 №4

По Разработка корпоративных приложений

Тема Связи между таблицами

Руководитель

Кульков Я.Ю.

(фамилия, инициалы)

(подпись) (дата)

Студент ПИН - 121

(группа)

Ермилов М.В.

(фамилия, инициалы)

(подпись) (дата)

Муром 2024

Лабораторная работа №4

Тема: связи между таблицами.

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

Ход работы: задание: создать набор сущностей в зависимости от варианта (либо по теме курсовой работы). Реализовать web-приложение для работы с соответствующими данными. Базовые действия: вывод информации о всех экземплярах и ввод нового. Реализовать валидацию данных. Реализовать поиск по различным критериям. Задание для варианта 7: сущность – Военнослужащий; атрибуты : фамилия; имя; отчество; национальность; дата рождения (год, месяц число); должность; звание.

Листинг кода 1 – MainController.java

package ru.kafpin.firsttt.controllers;

import jakarta.validation.Valid;

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

import org.springframework.stereotype.Controller;

import org.springframework.ui.Model;

import org.springframework.validation.BindingResult;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.ModelAttribute;

import org.springframework.web.bind.annotation.PathVariable;

import org.springframework.web.bind.annotation.PostMapping;

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.util.List;

@Controller

public class MainController {

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("/")

public String RootPage(Model model){

return "redirect:/soldiers";

}

@GetMapping("/kazarma")

public String KazarmaPage(Model model){

return "redirect:/soldiers";

}

@GetMapping("/soldiers")

public String SoldiersPage(Model model){

List<Soldier> allSoldiers = soldierService.getAllSoldiers();

model.addAttribute("soldiers", allSoldiers);

model.addAttribute("post", new Post());

return "soldiers"; }

@GetMapping("/posts")

public String PostsPage(Model model){

List<Post> allPosts = postService.getAllPosts();

model.addAttribute("posts", allPosts);

return "posts";

}

@GetMapping("/soldiers/details/{id}")

public String soldierDetailsPage(Model model, @PathVariable("id") Long id) {

Soldier selectedSoldier = soldierService.getSoldierById(id);

model.addAttribute("selectedSoldier", selectedSoldier);

return "soldierdetails";

}

@GetMapping("/posts/details/{id}")

public String postDetailsPage(Model model, @PathVariable("id") Long id) {

Post selectedPost = postService.getPostById(id);

model.addAttribute("selectedPost", selectedPost);

return "postdetails";

}

@GetMapping("/soldiers/delete/{id}")

public String deleteSoldierById(@PathVariable("id") Long id){

soldierService.deleteSoldierById(id);

return "redirect:/soldiers";

}

@GetMapping("/posts/delete/{id}")

public String deletePostById(@PathVariable("id") Long id){

postService.deletePostById(id);

return "redirect:/posts";

}

@GetMapping("/soldiers/add")

public String addSoldier(Model model){

model.addAttribute("soldier", new Soldier());

return "addeditsoldier";

}

@GetMapping("/posts/add")

public String addPost(Model model){

model.addAttribute("post", new Post());

return "addeditpost";

}

@PostMapping("/soldiers/add")

public String mainSoldierForm(@ModelAttribute("soldier") @Valid Soldier soldier, BindingResult result){

if(result.hasErrors()){

return "addeditsoldier";

}

soldierService.addEditSoldier(soldier);

return "redirect:/soldiers";

}

@PostMapping("/posts/add")

public String mainPostForm(@ModelAttribute("post") @Valid Post post, BindingResult result){

if(result.hasErrors()){

return "addeditpost";

}

postService.addEditPost(post);

return "redirect:/posts";

}

@GetMapping("/soldiers/update/{id}")

public String editSoldier(Model model, @PathVariable("id") Long id) {

model.addAttribute("soldier", soldierService.getSoldierById(id));

return "addeditsoldier";

}

@GetMapping("/posts/update/{id}")

public String editPost(Model model, @PathVariable("id") Long id) {

model.addAttribute("post", postService.getPostById(id));

return "addeditpost";

}

@PostMapping("/findsoldiersbypost")

public String getSoldiers(@ModelAttribute Post post,

Model model){

if(post.getTitle().isEmpty())

return "redirect:/kazarma";

List<Soldier> soldiers = soldierService

.getAllSoldiersByPost(post.getTitle());

model.addAttribute("soldiers", soldiers);

model.addAttribute("post", new Post());

return "soldiers";

}

}

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(cascade = CascadeType.ALL)

@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 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;

@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) {

return soldierRepository.findById(id).get();

}

public void deleteSoldierById(Long id) {

if (soldierRepository.existsById(id))

soldierRepository.deleteById(id);

}

public void addEditSoldier(Soldier soldier) {

soldierRepository.save(soldier);

}

}

Листинг кода 5 – Post.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 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")

private List<Soldier> soldiers;

}

Листинг кода 6 – 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> {

}

Листинг кода 7 – PostService.java

package ru.kafpin.firsttt.services;

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;

@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) {

return postRepository.findById(id).get();

}

public void deletePostById(Long id) {

if (postRepository.existsById(id))

postRepository.deleteById(id);

}

public void addEditPost(Post post) {

postRepository.save(post);

}

}

Листинг кода 8 – addeditsoldier.html

<!DOCTYPE HTML>

<html xmlns:th="https://www.thymeleaf.org">

<head>

<title>Добавление военнослужащего</title>

<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.min.css">

<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.1.1/jquery.min.js"></script>

<script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.1.3/js/bootstrap.min.js"></script>

</head>

<body>

<div class="container">

<h1>Информация о военнослужащем</h1>

<a class="btn" href="/kazarma">К списку военнослужащих</a>

<form class="customfrm" action="#" th:action="@{/soldiers/add}" th:object="${soldier}" method="post">

<div class="form-group">

<input type="hidden" class="form-control" th:field="\*{id}" />

<div class="text-danger" th:if="${#fields.hasErrors('surname')}" th:errors="\*{surname}"/>

<p>Фамилия: <input type="text" class="form-control" placeholder="Введите фамилию" th:field="\*{surname}" /></p>

<div class="text-danger" th:if="${#fields.hasErrors('name')}" th:errors="\*{name}"/>

<p>Имя: <input type="text" class="form-control" placeholder="Введите имя" th:field="\*{name}" /></p>

<p>Отчество: <input type="text" class="form-control" placeholder="Введите отчество" th:field="\*{patronym}" /></p>

<div class="text-danger" th:if="${#fields.hasErrors('nation')}" th:errors="\*{nation}"/>

<p>Национальность: <input type="text" class="form-control" placeholder="Введите национальность" th:field="\*{nation}" /></p>

<p>Дата рождения: <input type="date" class="form-control" content="" th:field="\*{birthday}" th:value="${{birthday}}" /></p>

<p>Должность: <input type="text" class="form-control" placeholder="Введите должность" th:field="\*{post.title}" /></p>

<div class="text-danger" th:if="${#fields.hasErrors('rank')}" th:errors="\*{rank}"/>

<p>Звание: <input type="text" class="form-control" placeholder="Введите звание" th:field="\*{rank}" /></p>

</div>

<p class="singleRow">

<input type="submit" class="btn btn-success" th:if="${soldier.id==null}" value="Добавить"/>

<input type="submit" class="btn btn-success" th:if="${soldier.id!=null}" value="Изменить"/>

<input type="reset" class="btn btn-secondary" value="Очистить поля ввода" />

</p>

</form>

</div>

</body>

</html>

Листинг кода 9 – soldierdetails.html

<!DOCTYPE html>

<html lang="en" xmlns:th="http://www.thymeleaf.org">

<head>

<meta charset="UTF-8">

<title>Информация о военнослужащем</title>

<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.min.css">

<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.1.1/jquery.min.js"></script>

<script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.1.3/js/bootstrap.min.js"></script>

</head>

<body>

<div class="container">

<h1>Военнослужащий</h1>

<a class="btn btn-secondary" th:href="@{'/soldiers/update/' + ${selectedSoldier.id}}">Редактировать</a>

<ul class="list-group">

<li class="list-group-item">ID: <span th:text="${selectedSoldier.id}"></span></li>

<li class="list-group-item">Фамилия: <span th:text="${selectedSoldier.surname}"></span></li>

<li class="list-group-item">Имя: <span th:text="${selectedSoldier.name}"></span></li>

<li class="list-group-item">Отчество: <span th:text="${selectedSoldier.patronym}"></span></li>

<li class="list-group-item">Национальность: <span th:text="${selectedSoldier.nation}"></span></li>

<li class="list-group-item">Дата рождения: <span th:text="${selectedSoldier.birthday}"></span></li>

<li class="list-group-item">Должность: <span th:text="${selectedSoldier.post.title}"></span></li>

<li class="list-group-item">Звание: <span th:text="${selectedSoldier.rank}"></span></li>

</ul>

<a th:href="@{'/kazarma'}" class="btn btn-primary mt-3">Назад к списку</a>

</div>

</body>

</html>

Листинг кода 10 – soldiers.html

<!DOCTYPE html>

<html lang="en" xmlns:th="http://www.thymeleaf.org">

<head>

<meta charset="UTF-8">

<title>Военнослужащие</title>

<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.min.css">

<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.1.1/jquery.min.js"></script>

<script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.1.3/js/bootstrap.min.js"></script>

</head>

<body>

<div class="container">

<h1>Военнослужащие</h1>

<span th:if="${soldiers.size() == 0}">Список военнослужащих пустой</span>

<table th:if="${soldiers.size() > 0}" class="table table-hover">

<thead class="thead-dark">

<tr>

<td>ID</td>

<td>Фамилия</td>

<td>Имя</td>

<td>Отчество</td>

<td>Национальность</td>

<td style="min-width: 110px;">Дата рождения</td>

<td>Должность</td>

<td>Звание</td>

<td></td>

<td></td>

<td></td>

</tr>

</thead>

<tbody>

<tr th:each="soldier : ${soldiers}">

<td th:text="${soldier.id}" />

<td th:text="${soldier.surname}" />

<td th:text="${soldier.name}" />

<td th:text="${soldier.patronym}" />

<td th:text="${soldier.nation}" />

<td th:text="${soldier.birthday}" />

<td th:text="${soldier.post.title}" />

<td th:text="${soldier.rank}" />

<td><a class="btn btn-info" th:href="@{'/soldiers/details/' + ${soldier.id}}">Подробнее</a></td>

<td><a class="btn btn-warning" th:href="@{'/soldiers/update/' + ${soldier.id}}">Редактировать</a></td>

<td><a class="btn btn-danger" th:href="@{'/soldiers/delete/' + ${soldier.id}}">Удалить</a></td>

</tr>

</tbody>

</table>

<a class="btn" th:href="@{'/soldiers/add'}">Добавить военнослужащего</a>

</div>

<div class="container">

<form class="customfrm" action="#" th:action="@{/findsoldiersbypost}" th:object="${post}" method="post">

<div class="form-group">

<p>Должность: <input type="text" class="form-control" placeholder="Введите должность" th:field="\*{title}"/></p>

</div>

<p class="singleRow">

<input type="submit" class="btn btn-success" value="Поиск" />

<input type="reset" class="btn btn-secondary" value="Очистить поле ввода" />

</p>

</form>

</div>

</body>

</html>

Листинг кода 11 – addeditpost.html

<!DOCTYPE HTML>

<html xmlns:th="https://www.thymeleaf.org">

<head>

<title>Добавление должности</title>

<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.min.css">

<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.1.1/jquery.min.js"></script>

<script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.1.3/js/bootstrap.min.js"></script>

</head>

<body>

<div class="container">

<h1>Информация о должности</h1>

<a class="btn" href="/posts">К списку должностей</a>

<form class="customfrm" action="#" th:action="@{/posts/add}" th:object="${post}" method="post">

<div class="form-group">

<input type="hidden" class="form-control" th:field="\*{id}" />

<div class="text-danger" th:if="${#fields.hasErrors('title')}" th:errors="\*{title}"/>

<p>Название: <input type="text" class="form-control" placeholder="Введите название" th:field="\*{title}"/></p>

</div>

<p class="singleRow">

<input type="submit" class="btn btn-success" th:if="${id==null}" value="Добавить"/>

<input type="submit" class="btn btn-success" th:if="${id!=null}" value="Изменить"/>

<input type="reset" class="btn btn-secondary" value="Очистить поля ввода" />

</p>

</form>

</div>

</body>

</html>

Листинг кода 12 – postdetails.html

<!DOCTYPE html>

<html lang="en" xmlns:th="http://www.thymeleaf.org">

<head>

<meta charset="UTF-8">

<title>Информация о должности</title>

<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.min.css">

<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.1.1/jquery.min.js"></script>

<script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.1.3/js/bootstrap.min.js"></script>

</head>

<body>

<div class="container">

<h1>Должность</h1>

<a class="btn btn-secondary" th:href="@{'/posts/update/' + ${selectedPost.id}}">Редактировать</a>

<ul class="list-group">

<li class="list-group-item">ID: <span th:text="${selectedPost.id}"></span></li>

<li class="list-group-item">Название: <span th:text="${selectedPost.title}"></span></li>

</ul>

<a th:href="@{'/posts'}" class="btn btn-primary mt-3">Назад к списку</a>

</div>

</body>

</html>

Листинг кода 13 – posts.html

<!DOCTYPE html>

<html lang="en" xmlns:th="http://www.thymeleaf.org">

<head>

<meta charset="UTF-8">

<title>Должности</title>

<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.min.css">

<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.1.1/jquery.min.js"></script>

<script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.1.3/js/bootstrap.min.js"></script>

</head>

<body>

<div class="container">

<h1>Должности</h1>

<span th:if="${posts.size() == 0}">Список должностей пустой</span>

<table th:if="${posts.size() > 0}" class="table table-hover">

<thead class="thead-dark">

<tr>

<td>ID</td>

<td>Название</td>

<td></td>

<td></td>

<td></td>

</tr>

</thead>

<tbody>

<tr th:each="post : ${posts}">

<td th:text="${post.id}" />

<td th:text="${post.title}" />

<td><a class="btn btn-info" th:href="@{'/posts/details/' + ${post.id}}">Подробнее</a></td>

<td><a class="btn btn-warning" th:href="@{'/posts/update/' + ${post.id}}">Редактировать</a></td>

<td><a class="btn btn-danger" th:href="@{'/posts/delete/' + ${post.id}}">Удалить</a></td>

</tr>

</tbody>

</table>

<a class="btn" th:href="@{'/posts/add'}">Добавить должность</a>

</div>

</body>

</html>

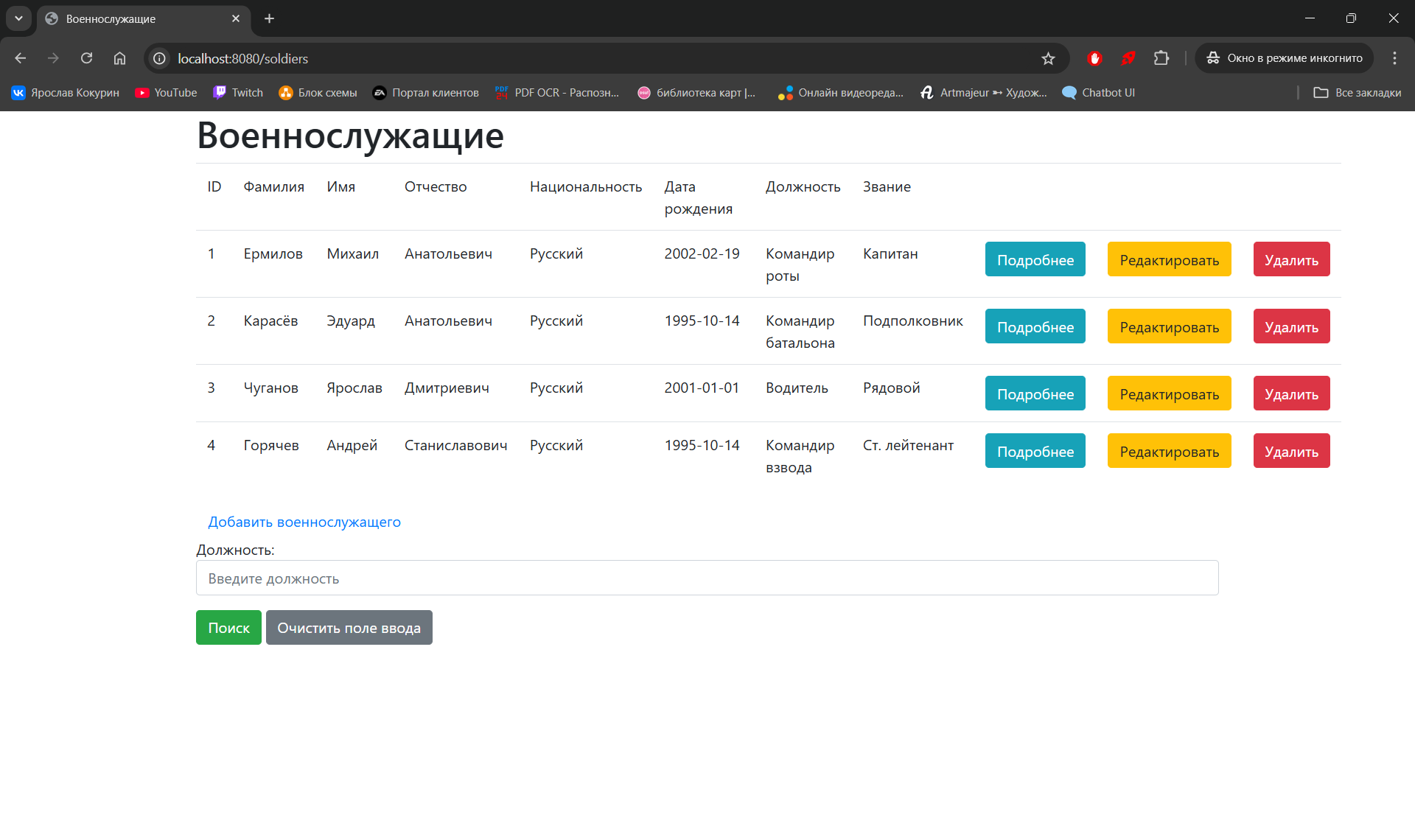


Рисунок 1 – Главная страница

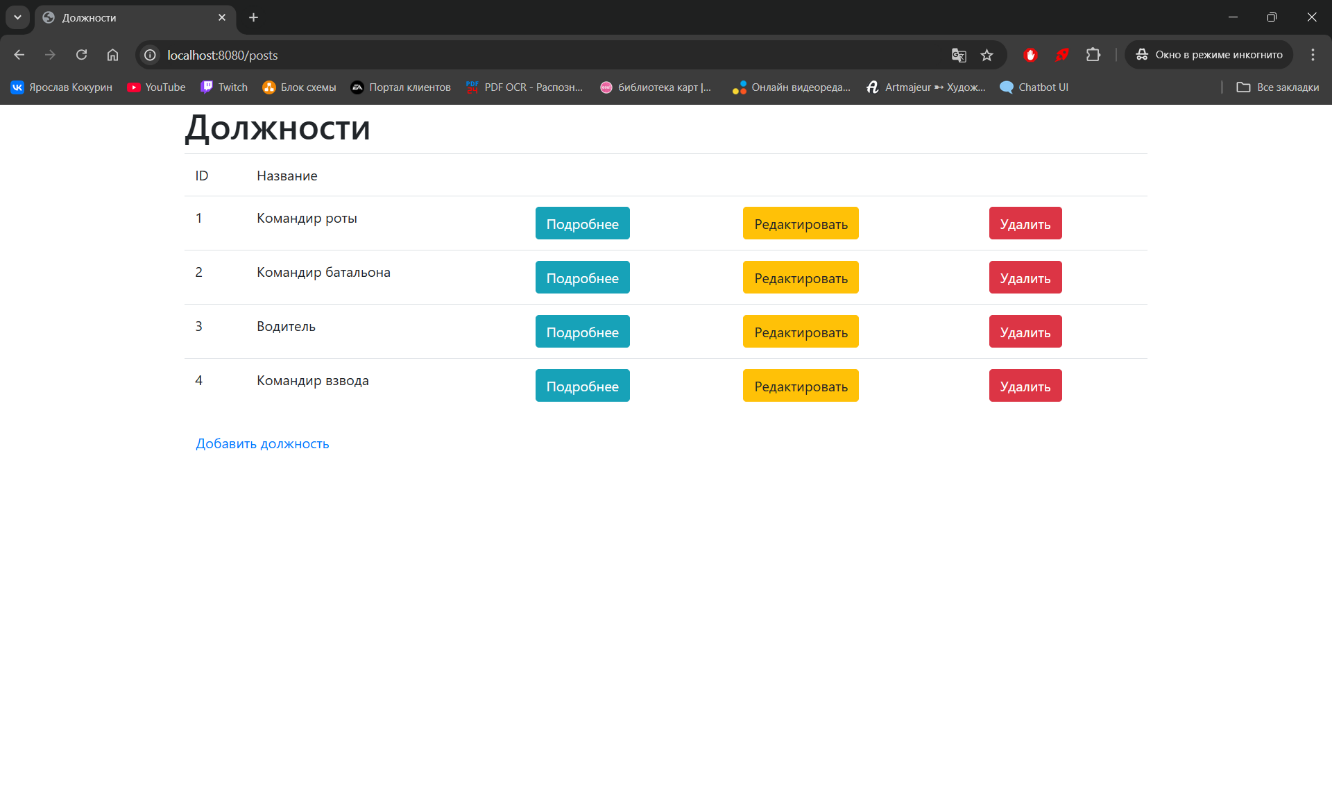


Рисунок 2 – Страница с должностями

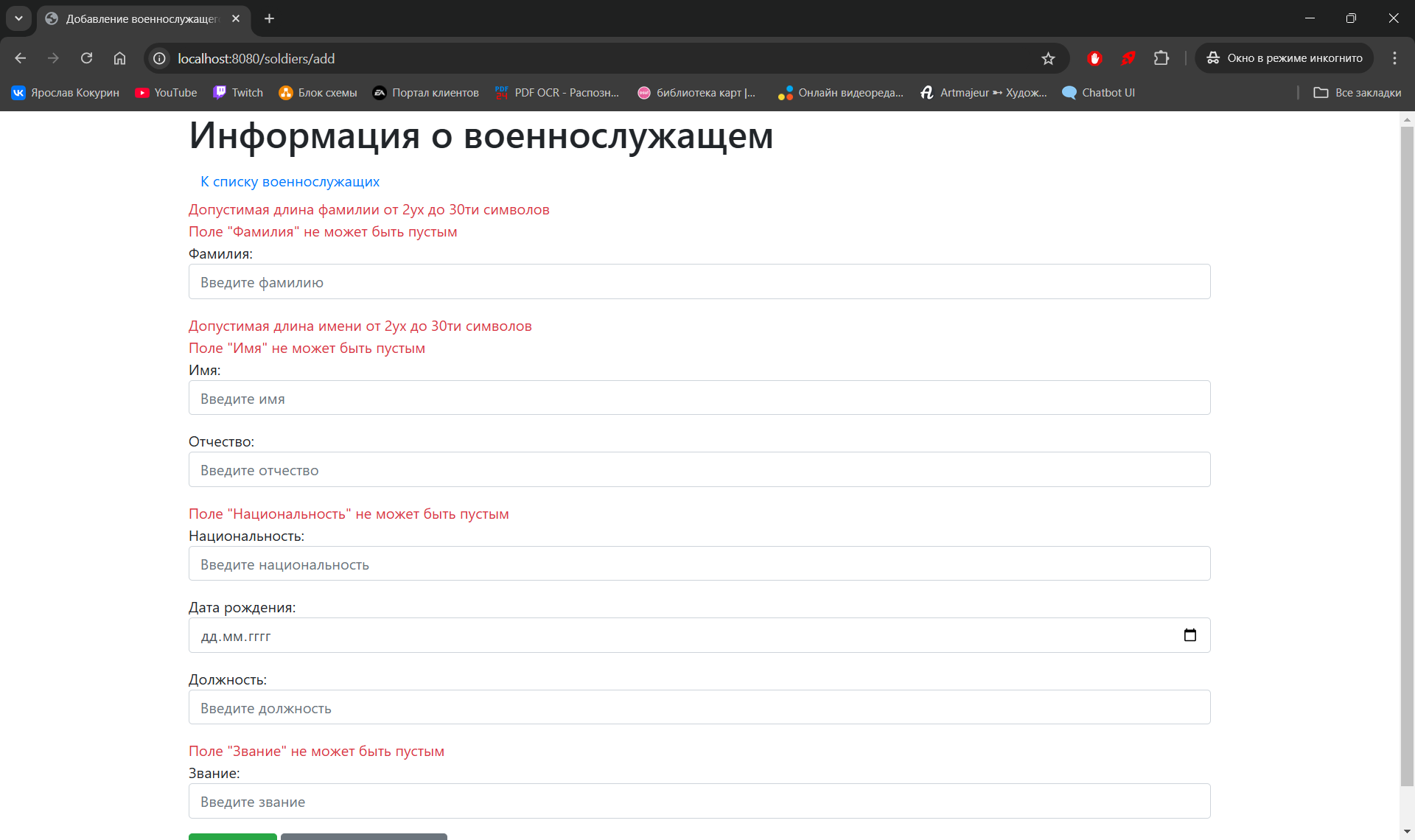


Рисунок 3 – Валидация при добавлении военнослужащего

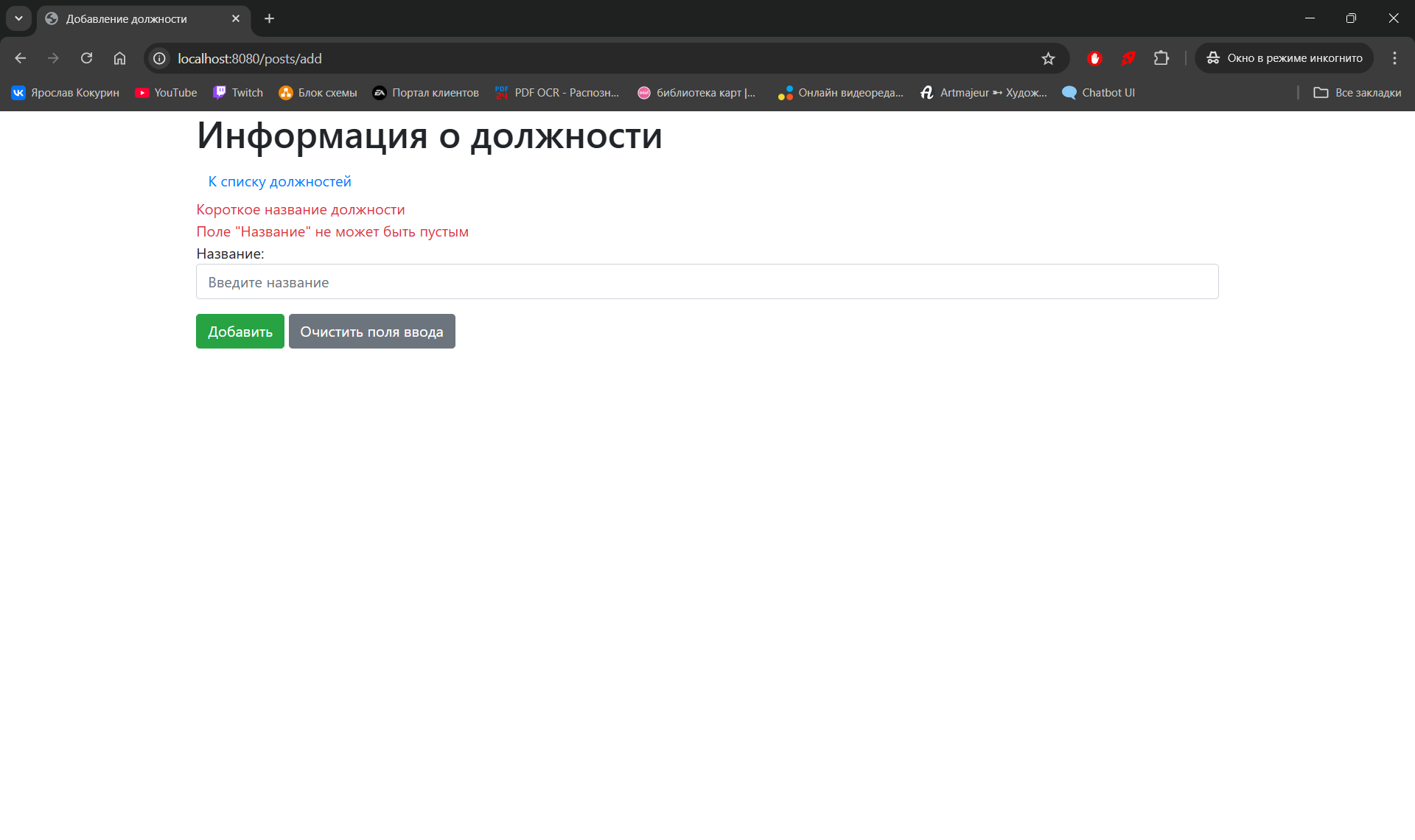


Рисунок 4 – Валидация при добавлении должности

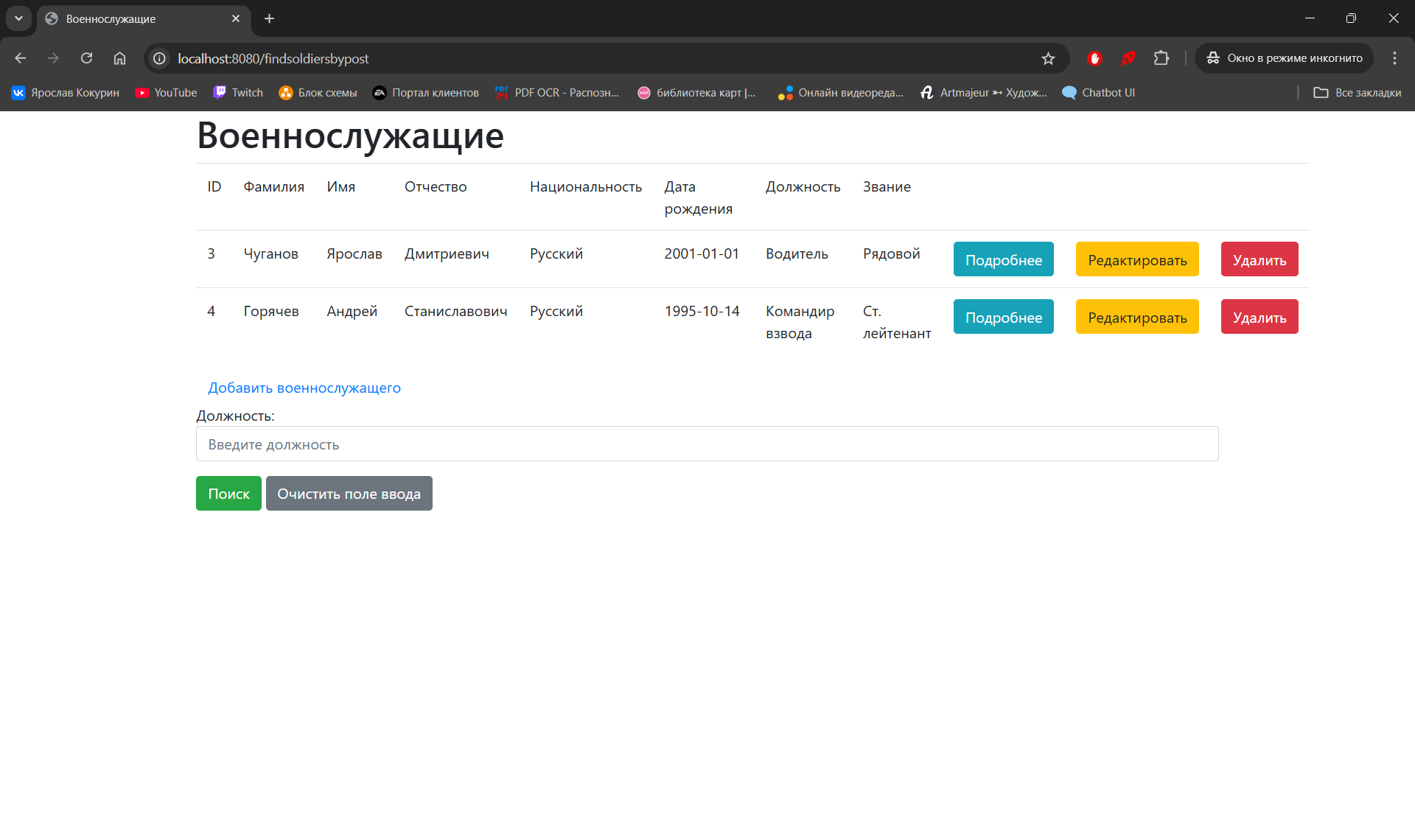


Рисунок 5 – Поиск военнослужащих, в должности которых содержится подстрока «во»

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