МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение высшего образования

«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

АЭРОКОСМИЧЕСКОГО ПРИБОРОСТРОЕНИЯ»

КАФЕДРА КОМПЬЮТЕРНЫХ ТЕХНОЛОГИЙ

И ПРОГРАММНОЙ ИНЖЕНЕРИИ (КАФЕДРА №43)

|  |  |  |
| --- | --- | --- |
| ОТЧЕТ ЗАЩИЩЕН С ОЦЕНКОЙ: |  |  |

ПРЕПОДАВАТЕЛЬ:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| асс. | / |  | / |  | / | Д. А. Кочин |
| (должность, учёная степень, звание) |  | (подпись) |  | (дата защиты) |  | (инициалы, фамилия) |

ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ №3

«Разработка простого AJAX приложения Spring»

ПО КУРСУ: «Технология разработки серверных информационных систем»

|  |  |  |  |
| --- | --- | --- | --- |
| РАБОТУ ВЫПОЛНИЛ (-А) СТУДЕНТ (-КА): | 4936 | / | Е. И. Белов |
|  | (номер группы) |  | (инициалы, фамилия) |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | / |  | / | 04.10.2021 |
|  |  | (подпись студента) |  | (дата отчета) |

Санкт-Петербург 2021

**Лабораторная работа 3.**

**Разработка простого AJAX приложения Spring.**

**Задание на лабораторную работу:**

1 Создайте один или несколько форм и контроллеров для страниц приложения. Подключите библиотеку angularJS.

2 Реализуйте задачи из лабораторной работы 1 средствами AJAX и REST-сервисов, разработанных в предыдущей лабораторной работе.

**Вариант задания 2:**

2. Поликлиника (запись на прием к врачу)

**Описание разрабатываемого продукта:**

На данном этапе разработки были реализованы функции добавления записи, удаления отдельных записей, вывода отдельной и всех записей, редактирования средствами AJAX и REST-сервисов.

Начальные данные в базе данных:

Registration reg1 = new Registration(1L, "Вася", "Доктор 1", "2021-10-09T18:00");  
Registration reg2 = new Registration(2L, "Коля", "Доктор 2", "2021-10-08T15:00");  
Registration reg3 = new Registration(3L, "Михаил", "Доктор 3","2021-10-07T14:00");

**Текст основных фрагментов кода**:

Информация об одной записи хранится в классе (id – идентификатор записи, fio – фио посетителя, doctor – фио доктора, date – дата записи):

public class Registration {  
  
 private Long regId;  
 private String regFIO;  
 private String regDOCTOR;  
 private String regDATE;  
  
 public Registration() {  
  
 }

Rest-сервисы вывода реализованы методом GET.

@RequestMapping(value = "/registrations", //  
 method = RequestMethod.*GET*, //  
 produces = { MediaType.*APPLICATION\_JSON\_VALUE*, //  
 MediaType.*APPLICATION\_XML\_VALUE* })  
@ResponseBody  
public List<Registration> getRegistrations() {  
 return registrationDAO.getAllRegistrations();  
}

Rest-сервис добавления реализованы с помощью метода POST.

@RequestMapping(value = "/registration", //  
 method = RequestMethod.*POST*, //  
 produces = { MediaType.*APPLICATION\_JSON\_VALUE*, //  
 MediaType.*APPLICATION\_XML\_VALUE* })  
@ResponseBody  
public Registration addRegistration(@RequestBody RegistrationForm registrationForm) {  
  
 System.*out*.println("(Service Side) Создаем запись пациенту " + registrationForm.getRegFIO() + " к врачу " + registrationForm.getRegDOCTOR());  
  
 return registrationDAO.addRegistration(registrationForm);  
}

Rest-сервис удаления реализован с помощью метода DELETE.

@RequestMapping(value = "/registration/{regId}", //  
 method = RequestMethod.*DELETE*, //  
 produces = { MediaType.*APPLICATION\_JSON\_VALUE*, MediaType.*APPLICATION\_XML\_VALUE* })  
@ResponseBody  
public void deleteRegistration(@PathVariable("regId") Long regId) {  
  
 System.*out*.println("(Service Side) Удаляем записть с Id: " + regId);  
  
 registrationDAO.deleteRegistration(regId);  
}

Rest-сервис изменения записи реализован с помощью метода PUT.

@RequestMapping(value = "/registration", //  
 method = RequestMethod.*PUT*, //  
 produces = { MediaType.*APPLICATION\_JSON\_VALUE*, //  
 MediaType.*APPLICATION\_XML\_VALUE* })  
@ResponseBody  
public Registration updateRegistration(@RequestBody RegistrationForm registrationForm) {  
  
 System.*out*.println("(Service Side) Изменяем запись c Id: " + registrationForm.getRegId());  
  
 return registrationDAO.updateRegistration(registrationForm);  
}

Код приложения приведен в приложении А.

**Выводы:**

В ходе лабораторной работы была подключена билеотека angularJS, собран проект веб-приложения на Maven. Реализованы задачи из лабораторной работы 1 средствами AJAX и REST-сервисов, разработанных в предыдущей лабораторной работе.

**Приложение А**

**pom.xml**

*<?*xml version="1.0" encoding="UTF-8"*?>*<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
 xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 https://maven.apache.org/xsd/maven-4.0.0.xsd">  
 <modelVersion>4.0.0</modelVersion>  
 <parent>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-parent</artifactId>  
 <version>2.5.5</version>  
 <relativePath/> *<!-- lookup parent from repository -->* </parent>  
 <groupId>com.example</groupId>  
 <artifactId>lab3</artifactId>  
 <version>0.0.1-SNAPSHOT</version>  
 <name>lab3</name>  
 <description>lab3</description>  
 <properties>  
 <java.version>1.8</java.version>  
 </properties>  
 <dependencies>  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-thymeleaf</artifactId>  
 </dependency>  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-web</artifactId>  
 </dependency>  
  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-devtools</artifactId>  
 <scope>runtime</scope>  
 <optional>true</optional>  
 </dependency>  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-test</artifactId>  
 <scope>test</scope>  
 </dependency>  
 </dependencies>  
  
 <build>  
 <plugins>  
 <plugin>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-maven-plugin</artifactId>  
 </plugin>  
 </plugins>  
 </build>  
  
</project>

**application.properties**

sprinspring.thymeleaf.cache=falseg.thymeleaf.cache=false

**Lab3Application.java**

package com.example.lab2;  
  
import org.springframework.boot.SpringApplication;  
import org.springframework.boot.autoconfigure.SpringBootApplication;  
  
@SpringBootApplication  
public class Lab2Application {  
  
 public static void main(String[] args) {  
 SpringApplication.*run*(Lab2Application.class, args);  
 }  
  
}

**MainController.java**

package com.example.lab3.controller;  
  
import org.springframework.stereotype.Controller;  
import org.springframework.web.bind.annotation.RequestMapping;  
  
@Controller  
public class MainController {  
  
 @RequestMapping("/")  
 public String welcome() {  
 return "index";  
 }  
}

**MainRestController.java**

package com.example.lab3.controller;  
  
import java.util.List;  
  
import com.example.lab3.dao.RegistrationDAO;  
import com.example.lab3.model.Registration;  
import com.example.lab3.model.RegistrationForm;  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.http.MediaType;  
import org.springframework.web.bind.annotation.PathVariable;  
import org.springframework.web.bind.annotation.RequestBody;  
import org.springframework.web.bind.annotation.RequestMapping;  
import org.springframework.web.bind.annotation.RequestMethod;  
import org.springframework.web.bind.annotation.ResponseBody;  
import org.springframework.web.bind.annotation.RestController;  
  
@RestController  
public class MainRESTController {  
  
 @Autowired  
 private RegistrationDAO registrationDAO;  
  
 @RequestMapping(value = "/registrations", *//* method = RequestMethod.GET, *//* produces = { MediaType.APPLICATION\_JSON\_VALUE, *//* MediaType.APPLICATION\_XML\_VALUE })  
 @ResponseBody  
 public List<Registration> getRegistrations() {  
 return registrationDAO.getAllRegistrations();  
 }  
  
 @RequestMapping(value = "/registration/{regId}", *//* method = RequestMethod.GET, *//* produces = { MediaType.APPLICATION\_JSON\_VALUE, *//* MediaType.APPLICATION\_XML\_VALUE })  
 @ResponseBody  
 public Registration getRegistration(@PathVariable("regId") Long regId) {  
 return registrationDAO.getRegistration(regId);  
 }  
  
 @RequestMapping(value = "/registration", *//* method = RequestMethod.POST, *//* produces = { MediaType.APPLICATION\_JSON\_VALUE, *//* MediaType.APPLICATION\_XML\_VALUE })  
 @ResponseBody  
 public Registration addRegistration(@RequestBody RegistrationForm registrationForm) {  
  
 System.out.println("(Service Side) Создаем запись пациенту " + registrationForm.getRegFIO() + " к врачу " + registrationForm.getRegDOCTOR());  
  
 return registrationDAO.addRegistration(registrationForm);  
 }  
  
 @RequestMapping(value = "/registration", *//* method = RequestMethod.PUT, *//* produces = { MediaType.APPLICATION\_JSON\_VALUE, *//* MediaType.APPLICATION\_XML\_VALUE })  
 @ResponseBody  
 public Registration updateRegistration(@RequestBody RegistrationForm registrationForm) {  
  
 System.out.println("(Service Side) Изменяем запись c Id: " + registrationForm.getRegId());  
  
 return registrationDAO.updateRegistration(registrationForm);  
 }  
  
 @RequestMapping(value = "/registration/{regId}", *//* method = RequestMethod.DELETE, *//* produces = { MediaType.APPLICATION\_JSON\_VALUE, MediaType.APPLICATION\_XML\_VALUE })  
 @ResponseBody  
 public void deleteRegistration(@PathVariable("regId") Long regId) {  
  
 System.out.println("(Service Side) Удаляем записть с Id: " + regId);  
  
 registrationDAO.deleteRegistration(regId);  
 }  
  
}

**RegistrationDAO.java**

package com.example.lab3.dao;  
  
import java.util.ArrayList;  
import java.util.Collection;  
import java.util.HashMap;  
import java.util.List;  
import java.util.Map;  
import java.util.Set;  
  
import com.example.lab3.model.Registration;  
import com.example.lab3.model.RegistrationForm;  
import org.springframework.stereotype.Repository;  
  
@Repository  
public class RegistrationDAO {  
  
 private static final Map<Long, Registration> *regMap* = new HashMap<Long, Registration>();  
  
 static {  
 *initRegs*();  
 }  
  
 private static void initRegs() {  
 Registration reg1 = new Registration(1L, "Вася", "Доктор 1", "2021-10-09T18:00");  
 Registration reg2 = new Registration(2L, "Коля", "Доктор 2", "2021-10-08T15:00");  
 Registration reg3 = new Registration(3L, "Михаил", "Доктор 3","2021-10-07T14:00");  
  
 *regMap*.put(reg1.getRegId(), reg1);  
 *regMap*.put(reg2.getRegId(), reg2);  
 *regMap*.put(reg3.getRegId(), reg3);  
 }  
  
 public Long getMaxRegId() {  
 Set<Long> keys = *regMap*.keySet();  
 Long max = 0L;  
 for (Long key : keys) {  
 if (key > max) {  
 max = key;  
 }  
 }  
 return max;  
 }  
  
 public Registration getRegistration(Long regId) {  
 return *regMap*.get(regId);  
 }  
  
 public Registration addRegistration(RegistrationForm regForm) {  
 Long regId= this.getMaxRegId()+ 1;  
 regForm.setRegId(regId);  
 Registration newReg = new Registration(regForm);  
  
 *regMap*.put(regId, newReg);  
 return newReg;  
 }  
  
 public Registration updateRegistration(RegistrationForm regForm) {  
 Registration reg = this.getRegistration(regForm.getRegId());  
 if(reg != null) {  
 reg.setRegFIO(regForm.getRegFIO());  
 reg.setRegDOCTOR(regForm.getRegDOCTOR());  
 reg.setRegDATE(regForm.getRegDATE());  
 }  
 return reg;  
 }  
  
 public void deleteRegistration(Long regId) {  
 *regMap*.remove(regId);  
 }  
  
 public List<Registration> getAllRegistrations() {  
 Collection<Registration> c = *regMap*.values();  
 return new ArrayList<>(c);  
 }  
  
}

**Registration.java**

package com.example.lab3.model;  
  
public class Registration {  
  
 private Long regId;  
 private String regFIO;  
 private String regDOCTOR;  
 private String regDATE;  
  
 public Registration() {  
  
 }  
  
 public Registration(RegistrationForm regForm) {  
 this.regId = regForm.getRegId();  
 this.regFIO = regForm.getRegFIO();  
 this.regDOCTOR = regForm.getRegDOCTOR();  
 this.regDATE = regForm.getRegDATE();  
 }  
  
 public Registration(Long Id, String FIO, String DOCTOR, String DATE) {  
 this.regId = Id;  
 this.regFIO = FIO;  
 this.regDOCTOR = DOCTOR;  
 this.regDATE = DATE;  
 }  
  
 public Long getRegId() {  
 return regId;  
 }  
  
 public void setRegId(Long regId) {  
 this.regId = regId;  
 }  
  
 public String getRegFIO() {  
 return regFIO;  
 }  
  
 public void setRegFIO(String regFIO) {  
 this.regFIO = regFIO;  
 }  
  
 public String getRegDOCTOR() {  
 return regDOCTOR;  
 }  
  
 public void setRegDOCTOR(String regDOCTOR) {  
 this.regDOCTOR = regDOCTOR;  
 }  
  
 public String getRegDATE() {  
 return regDATE;  
 }  
  
 public void setRegDATE(String regDATE) {  
 this.regDATE = regDATE;  
 }  
}

**RegistrationForm.java**

package com.example.lab3.model;  
  
public class RegistrationForm {  
  
 private Long regId;  
 private String regFIO;  
 private String regDOCTOR;  
 private String regDATE;  
  
 public Long getRegId() {  
 return regId;  
 }  
  
 public void setRegId(Long regId) {  
 this.regId = regId;  
 }  
  
 public String getRegFIO() {  
 return regFIO;  
 }  
  
 public void setRegFIO(String regFIO) {  
 this.regFIO = regFIO;  
 }  
  
 public String getRegDOCTOR() {  
 return regDOCTOR;  
 }  
  
 public void setRegDOCTOR(String regDOCTOR) {  
 this.regDOCTOR = regDOCTOR;  
 }  
  
 public String getRegDATE() {  
 return regDATE;  
 }  
  
 public void setRegDATE(String regDATE) {  
 this.regDATE = regDATE;  
 }  
}

**main.js**

var app = angular.*module*("RegistrationManagement", []);  
  
app.controller("RegistrationController", function($scope, $http) {  
  
 $scope.registrations = [];  
 $scope.registrationForm = {  
 regId: 1,  
 regFIO: "",  
 regDOCTOR: "",  
 regDATE: ""  
 };  
  
 *// ЗАГРУЗКА ДАННЫХ С СЕРВЕРА* \_refreshRegistrationData();  
  
 *// HTTP POST/PUT методы для добавления/изменения записи* $scope.submitRegistration = function() {  
  
 var method = "";  
 var url = "";  
  
 if ($scope.registrationForm.regId === -1) {  
 method = "POST";  
 url = '/registration';  
 } else {  
 method = "PUT";  
 url = '/registration';  
 }  
  
 $http({  
 method: method,  
 url: url,  
 data: angular.toJson($scope.registrationForm),  
 headers: {  
 'Content-Type': 'application/json'  
 }  
 }).then(\_success, \_error);  
 };  
  
 $scope.createRegistration = function() {  
 \_clearFormData();  
 }  
  
 *// HTTP DELETE- удаление записи* $scope.deleteRegistration = function(registration) {  
 $http({  
 method: 'DELETE',  
 url: '/registration/' + registration.regId  
 }).then(\_success, \_error);  
 };  
  
 *//В случае измененич записи* $scope.editRegistration = function(registration) {  
 $scope.registrationForm.regId = registration.regId;  
 $scope.registrationForm.regFIO = registration.regFIO;  
 $scope.registrationForm.regDOCTOR = registration.regDOCTOR;  
 $scope.registrationForm.regDATE = registration.regDATE;  
 };  
  
 *// Приватный метод  
 // HTTP GET- получить все записи* function \_refreshRegistrationData() {  
 $http({  
 method: 'GET',  
 url: '/registrations'  
 }).then(  
 function(res) { *// успех* $scope.registrations = res.data;  
 },  
 function(res) { *// ошибка* console.log("Error: " + res.status + " : " + res.data);  
 }  
 );  
 }  
  
 function \_success(res) {  
 \_refreshRegistrationData();  
 \_clearFormData();  
 }  
  
 function \_error(res) {  
 var data = res.data;  
 var status = res.status;  
 var header = res.header;  
 var config = res.config;  
 *alert*("Error: " + status + ":" + data);  
 }  
  
 *// Очистка формы* function \_clearFormData() {  
 $scope.registrationForm.regId = -1;  
 $scope.registrationForm.regFIO = "";  
 $scope.registrationForm.regDOCTOR = "";  
 $scope.registrationForm.regDATE = ""  
 }  
});

**index.html**

<!DOCTYPE HTML>  
<html xmlns:th="http://www.thymeleaf.org">  
<head>  
 <title>БОЛЬНИЦА</title>  
 <script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.6.7/angular.js"></script>  
  
 <script th:src="@{/main.js}"></script>  
 <link th:href="@{/main.css}" rel="stylesheet" />  
 <head>  
<body ng-app="RegistrationManagement" ng-controller="RegistrationController">  
<h3>  
 Запись к врачу  
</h3>  
<form ng-submit="submitRegistration()">  
 <table border="0">  
 <tr>  
 <td>Текущий Id</td>  
 <td>{{registrationForm.regId}}</td>  
 </tr>  
 <tr>  
 <td>Ваше ФИО</td>  
 <td><input type="text" ng-model="registrationForm.regFIO" /></td>  
 </tr>  
 <tr>  
 <td>Лечащий врач</td>  
 <td><input type="text" ng-model="registrationForm.regDOCTOR" /></td>  
 </tr>  
 <tr>  
 <td>Дата</td>  
 <td><input type="text" ng-model="registrationForm.regDATE" /></td>  
 </tr>  
 <tr>  
 <td colspan="2">  
 <input type="submit" value="Подтвердить" class="blue-button" />  
 </td>  
 </tr>  
 </table>  
</form>  
<br/>  
<a class="create-button" ng-click="createRegistration()">Добавить запись в таблицу</a>  
<table border="1">  
 <tr>  
 <th>Id</th>  
 <th>ФИО</th>  
 <th>Доктор</th>  
 <th>Дата</th>  
 <th>Удалить</th>  
 <th>Изменить</th>  
 </tr>  
 *<!-- $scope.registrations -->* <tr ng-repeat="registration in registrations">  
 <td> {{ registration.regId }}</td>  
 <td> {{ registration.regFIO }}</td>  
 <td >{{ registration.regDOCTOR }}</td>  
 <td >{{ registration.regDATE }}</td>  
 <td>  
 <a ng-click="deleteRegistration(registration)" class="delete-button">Удалить</a>  
 </td>  
 <td>  
 <a ng-click="editRegistration(registration)" class="edit-button">Изменить</a>  
 </td>  
 </tr>  
</table>  
</body>  
</html>