ГУАП

КАФЕДРА № 43

ОТЧЕТ   
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| канд.тех.наук |  |  |  | П.А. Степанов |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

|  |
| --- |
| ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ |
| РАЗРАБОТКА ФОРМЫ ЛОГИНА |
| по курсу: ТЕХНОЛОГИИ РАЗРАБОТКИ СЕРВЕРНЫХ ИНФОРМАЦИОННЫХ СИСТЕМ |
|  |
|  |

РАБОТУ ВЫПОЛНИЛ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| СТУДЕНТ гр. № | 4136 |  |  |  | Н.С. Бобрович |
|  |  |  | подпись, дата |  | инициалы, фамилия |

Санкт-Петербург 2023

**Задание (Вариант 3)**

Расписание занятий в институте

**Описание разрабатываемого продукта**

1 Переведите вебсервер на использование SSL  
2 Реализуйте form-based authentication  
3 Защитите ресурсы, требующие запись и введите аудит.

**Текст программы**

**DBHelper**

package com.example.Lab3;  
  
import java.util.ArrayList;  
import java.util.List;  
  
public class DBHelper {  
 static public List<Lesson> *dbLesson* = new ArrayList<Lesson>();  
}

**Lab4Application**

package com.example.Lab3;  
  
import org.springframework.boot.SpringApplication;  
import org.springframework.boot.autoconfigure.SpringBootApplication;  
  
@SpringBootApplication  
public class Lab3Application {  
  
 public static void main(String[] args) {  
 DBHelper.*dbLesson*.add(new Lesson(0, "4132", "23-09", "Ivanov", "Physics", 1));  
 DBHelper.*dbLesson*.add(new Lesson(1, "4133", "23-08", "Petrov", "Math", 3));  
 DBHelper.*dbLesson*.add(new Lesson(2, "4134", "23-10", "Smirnov", "Programmin", 5));  
 SpringApplication.*run*(Lab3Application.class, args);  
 }  
  
}

**Lesson**

package com.example.Lab4;  
  
public class Lesson {  
 private int id;  
 private String num\_group; // номер группы  
 private String audience; // номер аудитории  
 private String teacher; // фамилия преподавателя  
 private String name; // название предмета  
 private int number; // порядковый номер пары  
 public Lesson(int id,String group,String audience,String teacher,String name,int number){  
 this.id = id;  
 this.num\_group = group;  
 this.audience = audience;  
 this.teacher = teacher;  
 this.name = name;  
 this.number = number;  
 }  
 public int getId(){  
 return id;  
 }  
 public void setId(int id){  
 this.id = id;  
 }  
 public String getGroup(){  
 return num\_group;  
 }  
 public void setGroup(String group){  
 this.num\_group = group;  
 }  
 public String getAudience(){  
 return audience;  
 }  
 public void setAudience(String audience){  
 this.audience = audience;  
 }  
 public String getTeacher(){  
 return teacher;  
 }  
 public void setTeacher(String teacher){  
 this.teacher = teacher;  
 }  
 public String getName(){  
 return name;  
 }  
 public void setName(String name){  
 this.name = name;  
 }  
  
 public int getNumber(){  
 return number;  
 }  
 public void setNumber(int number){  
 this.number = number;  
 }  
  
}

**Service**

package com.example.Lab4;  
  
import java.util.List;  
  
public interface Service{  
  
 void create(String group,String audience,String teacher,String name,int number);  
  
 List<Lesson> getAll();  
  
 List<Lesson> getByGroup(String group);  
  
 boolean delete(int id);  
}

**ServiceImpl**

package com.example.Lab4;  
  
import org.springframework.stereotype.Service;  
  
import java.util.ArrayList;  
import java.util.List;  
  
@Service  
public class ServiceImpl implements com.example.Lab4.Service {  
  
  
 @Override  
 public void create(String group, String audience, String teacher, String name, int number) {  
  
 DBHelper.dbLesson.add(new Lesson(DBHelper.dbLesson.get(DBHelper.dbLesson.size()-1 > 0 ? DBHelper.dbLesson.size()-1 : 0).getId()+1,  
 group,  
 audience,  
 teacher,  
 name,  
 number  
 ));  
  
 }  
  
 @Override  
 public List<Lesson> getAll() {  
 return DBHelper.dbLesson;  
 }  
  
 @Override  
 public List<Lesson> getByGroup(String group) {  
 List<Lesson> les = new ArrayList<>();  
 for(int i = 0; i<DBHelper.dbLesson.size(); ++i){  
 if(DBHelper.dbLesson.get(i).getGroup().contentEquals(group)){  
 les.add(DBHelper.dbLesson.get(i));  
 }  
 }  
 return les;  
 }  
  
 @Override  
 public boolean delete(int id) {  
 for(int i = 0; i<DBHelper.dbLesson.size(); ++i){  
 if(DBHelper.dbLesson.get(i).getId() == id){  
 DBHelper.dbLesson.remove(i);  
 return true;  
 }  
 }  
 return false;  
 }  
}

**SimpleController**

package com.example.Lab3;  
  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.http.HttpStatus;  
import org.springframework.http.ResponseEntity;  
import org.springframework.security.core.annotation.AuthenticationPrincipal;  
import org.springframework.security.core.userdetails.UserDetails;  
import org.springframework.stereotype.Controller;  
import org.springframework.web.bind.annotation.DeleteMapping;  
import org.springframework.web.bind.annotation.GetMapping;  
import org.springframework.web.bind.annotation.PathVariable;  
import org.springframework.web.bind.annotation.PostMapping;  
import org.springframework.web.servlet.ModelAndView;  
  
import java.util.List;  
  
@Controller  
public class SimpleController {  
 private final Service service;  
 @Autowired  
 public SimpleController(Service service){  
 this.service = service;  
 }  
  
 @PostMapping("/lessons/{group}/{audience}/{teacher}/{name}/{number}")  
 public ResponseEntity<?> create(@PathVariable("group") String group,  
 @PathVariable("audience") String audience,  
 @PathVariable("teacher") String teacher,  
 @PathVariable("name") String name,  
 @PathVariable("number") int number){  
 service.create(group,audience,teacher,name,number);  
 return new ResponseEntity<>(HttpStatus.*CREATED*);  
 }  
  
 @GetMapping("/lessons")  
 public ResponseEntity<List<Lesson>> getAll(){  
 final List<Lesson> lessons = service.getAll();  
 return lessons!= null && !lessons.isEmpty()  
 ? new ResponseEntity<>(lessons,HttpStatus.*OK*)  
 : new ResponseEntity<>(HttpStatus.*NOT\_FOUND*);  
 }  
 @GetMapping("/lessons/{group}")  
 public ResponseEntity<List<Lesson>> getGroup(@PathVariable(name="group") String group){  
 final List<Lesson> lessons = service.getByGroup(group);  
 return lessons!= null && !lessons.isEmpty()  
 ? new ResponseEntity<>(lessons,HttpStatus.*OK*)  
 : new ResponseEntity<>(HttpStatus.*NOT\_FOUND*);  
 }  
  
 @DeleteMapping("/lessons/{id}")  
 public ResponseEntity<?> delete(@PathVariable(name="id") int id){  
 final boolean deleted = service.delete(id);  
 return deleted  
 ? new ResponseEntity<>(HttpStatus.*OK*)  
 : new ResponseEntity<>(HttpStatus.*NOT\_MODIFIED*);  
 }  
  
// @GetMapping("/main")  
// public String getMainPage(){  
// return "main";  
// }  
  
 @GetMapping("/main")  
 public ModelAndView getMainPage(@AuthenticationPrincipal UserDetails user){  
 ModelAndView nextPage = new ModelAndView("main");  
 nextPage.addObject("user",user);  
 return nextPage;  
 }  
  
// @GetMapping("/")  
// public ModelAndView defaultPage(@AuthenticationPrincipal UserDetails user){  
// ModelAndView nextPage = new ModelAndView("index");  
// nextPage.addObject("user",user);  
// return nextPage;  
// }  
  
  
}

**SecurityConfiguration**

package com.example.Lab3;  
  
import org.springframework.context.annotation.Bean;  
import org.springframework.context.annotation.Configuration;  
import org.springframework.security.config.annotation.web.builders.HttpSecurity;  
import org.springframework.security.config.annotation.web.configuration.EnableWebSecurity;  
import org.springframework.security.core.userdetails.User;  
import org.springframework.security.core.userdetails.UserDetails;  
import org.springframework.security.core.userdetails.UserDetailsService;  
import org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;  
import org.springframework.security.crypto.password.PasswordEncoder;  
import org.springframework.security.provisioning.InMemoryUserDetailsManager;  
import org.springframework.security.web.SecurityFilterChain;  
  
@Configuration  
@EnableWebSecurity  
public class SecurityConfiguration {  
 @Bean  
 SecurityFilterChain securityFilterChain(HttpSecurity http) throws Exception{  
 return http.authorizeRequests()  
 .requestMatchers("/main").authenticated()  
// .anyRequest().authenticated()  
 .and()  
 .formLogin(form->{})  
 .logout(log->log.logoutSuccessUrl("/login"))  
 .csrf(csrf->csrf.disable())  
 .build();  
 }  
  
 @Bean  
 UserDetailsService users(PasswordEncoder passwordEncoder){  
 UserDetails user = User.*builder*()  
 .username("danis")  
 .password(passwordEncoder.encode("danis"))  
 .roles("USER")  
 .build();  
 return new InMemoryUserDetailsManager(user);  
 }  
  
 @Bean  
 PasswordEncoder passwordEncoder(){  
 return new BCryptPasswordEncoder();  
 }  
}

**main.js**

var app = angular.module("les",[]);  
  
app.controller("lessonController",function($scope,$http){  
 $scope.lessonsList = [];  
  
 $scope.getLessons = function(){  
 $http.get("/lessons").success(function(data){  
 $scope.lessonsList = data;  
 }).error(function(data){  
 if(data.message === "Time is out"){  
 $scope.finishByTimeout();  
 }  
 });  
 };  
  
 $scope.deleteLesson = function(id){  
 $http.delete("/lessons/" + id).success(function(data){  
 for(var i=0;i<$scope.lessonsList.length;++i){  
 if($scope.lessonsList[i].id === id){  
 $scope.lessonsList.splice(i,1);  
 break;  
 }  
 }  
 }).error(function(data){  
 console.log("Error")  
 });  
 };  
  
 $scope.addLesson = function(){  
 $http.post("lessons/"+$scope.group + "/"+$scope.audience + "/"+$scope.teacher + "/" +$scope.name + "/" +$scope.number)  
 .success(function(data){  
 $scope.lessonsList.push(data);  
 }).error(function(data){  
 console.log("Error");  
 });  
 $scope.getLessons();  
 };  
});

**index.html**

<!DOCTYPE html>  
<html xmlns:th="http://www.w3.org/1999/xhtml">  
 <head>  
 <meta content="text/html" charset="utf-8">  
 <title>Расписание занятий студентов</title>  
 </head>  
 <body>  
 <h2>Расписание занятий студентов</h2>  
  
 <div th:if="${user != null}">  
 <h1>User : <span th:text="${user.username}" /></h1>  
 <h1><a href="/logout"> Logout</a> </h1>  
 <h1><a href="/main">View schedule</a> </h1>  
 </div>  
 <div th:if="${user == null}">  
 <h1> Anonymous user please authenticate</h1>  
 <h1><a href="/login">Login</a> </h1>  
 </div>  
 </body>  
</html>

**main.html**

<!DOCTYPE html>  
<html xmlns="http://www.w3.org/1999/xhtml" xmlns:th="http://www.thymeleaf.org" xmlns:sec="http://www.thymeleaf.org/thymeleaf-extras-springsecurity3">  
 <head>  
 <meta http-equiv="Conternt-Type" content="text/html;charset=utf-8">  
 <title>Lessons</title>  
 <script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.4.3/angular.min.js"></script>  
 <script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.4.3/angular-route.min.js"></script>  
 <script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.4.3/angular-resource.min.js"></script>  
  
 <script src="main.js"></script>  
 </head>  
 <body ng-app="les" ng-controller="lessonController">  
 <div >  
 <h2>User name : <span th:text="${user.username}"></span></h2>  
 <h1><a href="/logout">Logout</a> </h1>  
 </div>  
 <h1>Schedule</h1>  
 <div ng-controller="getLessons" ng-show="lessonsList.length > 0">  
 <table id="lessonsTable">  
 <thead>  
 <tr>  
 <th>Number</th>  
 <th>Subject</th>  
 <th>Audience</th>  
 <th>Teacher</th>  
 <th>Group</th>  
 <th></th>  
 </tr>  
 </thead>  
 <tbody>  
 <tr ng-repeat="lesson in lessonsList track by $index" id="{{lesson.id}}">  
 <td>{{lesson.number}}</td>  
 <td>{{lesson.name}}</td>  
 <td>{{lesson.audience}}</td>  
 <td>{{lesson.teacher}}</td>  
 <td>{{lesson.group}}</td>  
 <td><input type="button" ng-click="deleteLesson(lesson.id)" value="Delete"></td>  
 </tr>  
 </tbody>  
 </table>  
 </div><br>  
 <div>  
 <form role="form" ng-submit="addLesson()">  
 <label>Group:<input type="text" name="group" ng-model="group"></label><br>  
 <label>Audience:<input type="text" name="audience" ng-model="audience"></label><br>  
 <label>Teacher:<input type="text" name="teacher" ng-model="teacher"></label><br>  
 <label>Name:<input type="text" name="name" ng-model="name"></label><br>  
 <label>Number:<input type="text" name="number" ng-model="number"></label><br>  
 <input type="submit">  
 </form>  
 </div>  
 </body>  
</html>