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 ПРИЛОЖЕНИЯ С АСИНХРОННОЙ ОЧЕРЕДЬЮ СООБЩЕНИЙ |
| 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**

Целью работы является реализация простой системы распределенной репликации (“писатели-читатели”).  
1 Скачайте и разверните Apache Kafka  
2 Модифицируйте свое приложение со встраиваемой базой данных так, чтобы его можно было запустить в нескольких экземплярах на разных портах  
3 Реализуйте в рамках своего приложения Producer и Consumer такие, что  
 a. Producer при каждой операции записи оповещает соответствующий топик  
 b. Consumer при получении информации из топика записывает обновление в локальную (встроенную в приложение) базу  
4 Продемонстрируйте, что информация, записанная одним приложением, доступна второму приложению.

**Текст программы**

**Lab3Application**

package com.example.Lab3;  
  
import com.google.gson.Gson;  
import org.springframework.boot.SpringApplication;  
import org.springframework.boot.autoconfigure.SpringBootApplication;  
import org.springframework.kafka.annotation.EnableKafka;  
import org.springframework.kafka.annotation.KafkaListener;  
  
@SpringBootApplication  
public class Lab3Application {  
  
  
 public static void main(String[] args) {  
 SpringApplication.*run*(Lab3Application.class, args);  
 }  
  
}

**Lesson**

package com.example.Lab3;  
  
import jakarta.persistence.\*;  
  
@Entity  
public class Lesson {  
 @Override  
 public String toString() {  
 return "group:" + num\_group +", audience:"+audience + ", teacher:"+teacher+", subject:"+name+", number:"+number;  
 }  
 public Lesson() {  
 }  
  
 public Lesson(String num\_group, String audience, String teacher, String name, int number) {  
 this.num\_group = num\_group;  
 this.audience = audience;  
 this.teacher = teacher;  
 this.name = name;  
 this.number = number;  
 }  
  
 @Id  
 @GeneratedValue(strategy = GenerationType.*AUTO*)  
 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 getNum\_group(){  
 return num\_group;  
 }  
 public void setNum\_group(String num\_group){  
 this.num\_group = num\_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;  
 }  
  
}

**Lesson Repository**

package com.example.Lab3;  
  
import org.springframework.data.repository.CrudRepository;  
public interface LessonRepository extends CrudRepository<Lesson,Long> {  
}

**Service**

package com.example.Lab3;  
  
import com.example.Lab3.Lesson;  
  
import java.util.List;  
  
public interface Service{  
  
 void create(Lesson lesson);  
  
 List<Lesson> getAll();  
  
 void delete(int id);  
}

**ServiceImpl**

package com.example.Lab3;  
  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.stereotype.Service;  
  
import java.util.ArrayList;  
import java.util.List;  
  
@Service  
public class ServiceImpl implements com.example.Lab3.Service {  
  
 @Autowired  
 private LessonRepository LR;  
 @Override  
 public void create(Lesson lesson) {  
 LR.save(lesson);  
 }  
  
 @Override  
 public List<Lesson> getAll() {  
 List<Lesson> lessons = (List<Lesson>) LR.findAll();  
 return lessons;  
 }  
  
 @Override  
 public void delete(int id) {  
 LR.deleteById((long) id);  
 }  
}

**SimpleController**

package com.example.Lab3;  
  
import com.google.gson.Gson;  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.http.HttpStatus;  
import org.springframework.http.ResponseEntity;  
import org.springframework.kafka.annotation.EnableKafka;  
import org.springframework.kafka.annotation.KafkaListener;  
import org.springframework.kafka.core.KafkaTemplate;  
import org.springframework.stereotype.Controller;  
import org.springframework.web.bind.annotation.\*;  
  
import java.util.List;  
import java.util.UUID;  
  
@EnableKafka  
@Controller  
public class SimpleController {  
 @Autowired  
 private KafkaTemplate<String,String> kafkaTemplate;  
 private final Service service;  
 @Autowired  
 public SimpleController(Service service){  
 this.service = service;  
 }  
  
 @KafkaListener(topics = "test")  
 public void lessonListener(String les){  
 System.*out*.println("KafkaListener is called");  
 Gson gson = new Gson();  
 Lesson lesson = gson.fromJson(les,Lesson.class);  
 service.create(lesson);  
 }  
 @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){  
  
 Lesson les = new Lesson(group,audience,teacher,name,number);  
 Gson gson = new Gson();  
  
 kafkaTemplate.send("test", UUID.*randomUUID*().toString(),gson.toJson(les));  
  
// 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*);  
 }  
  
 @DeleteMapping("/lessons/{id}")  
 public ResponseEntity<?> delete(@PathVariable(name="id") int id){  
 service.delete(id);  
 return new ResponseEntity<>(HttpStatus.*OK*);  
  
 }  
}

}

**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">  
 <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>

**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>  
 <head>  
 <meta content="text/html" charset="utf-8">  
 <title>Расписание занятий студентов</title>  
 </head>  
 <body>  
 <h2>Расписание занятий студентов</h2>  
 <a href="main.html">View schedule</a><br>  
 </body>  
</html>

**application.properties**

spring.jpa.hibernate.ddl-auto=update  
spring.datasource.url=jdbc:mysql://localhost:3306/trsis  
spring.datasource.username=root  
spring.datasource.password=12345  
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver  
#spring.jpa.show-sql: true  
  
spring.kafka.consumer.group-id=app.1

**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>3.1.4</version>  
 <relativePath/> <!-- lookup parent from repository -->  
 </parent>  
 <groupId>com.example</groupId>  
 <artifactId>Lab3</artifactId>  
 <version>0.0.1-SNAPSHOT</version>  
 <name>Lab3</name>  
 <description>Demo project for Spring Boot</description>  
 <properties>  
 <java.version>17</java.version>  
 </properties>  
 <dependencies>  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-web</artifactId>  
 </dependency>  
  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-test</artifactId>  
 <scope>test</scope>  
 </dependency>  
  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-thymeleaf</artifactId>  
 </dependency>  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-devtools</artifactId>  
 <optional>true</optional>  
 </dependency>  
 <dependency>  
 <groupId>org.springframework</groupId>  
 <artifactId>spring-webmvc</artifactId>  
 <version>6.0.9</version>  
 </dependency>  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-data-jpa</artifactId>  
 </dependency>  
 <dependency>  
 <groupId>mysql</groupId>  
 <artifactId>mysql-connector-java</artifactId>  
 <version>8.0.28</version>  
 </dependency>  
 <dependency>  
 <groupId>org.apache.kafka</groupId>  
 <artifactId>kafka\_2.12</artifactId>  
 <version>2.6.0</version>  
 </dependency>  
 <dependency>  
 <groupId>com.fasterxml.jackson.core</groupId>  
 <artifactId>jackson-databind</artifactId>  
 <version>2.11.0</version>  
 </dependency>  
 <dependency>  
 <groupId>com.google.code.gson</groupId>  
 <artifactId>gson</artifactId>  
 <version>2.9.0</version>  
 </dependency>  
 <dependency>  
 <groupId>org.springframework.kafka</groupId>  
 <artifactId>spring-kafka</artifactId>  
 </dependency>  
 </dependencies>  
  
 <build>  
 <plugins>  
 <plugin>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-maven-plugin</artifactId>  
 </plugin>  
 </plugins>  
 </build>  
  
</project>