МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ федеральное государственное автономное образовательное учреждение высшего образования

«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ АЭРОКОСМИЧЕСКОГО ПРИБОРОСТРОЕНИЯ»

КАФЕДРА КОМПЬЮТЕРНЫХ ТЕХНОЛОГИЙ

И ПРОГРАМНОЙ ИНЖИНЕРИИ

№ 43

ОТЧЕТ   
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

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

|  |
| --- |
| ОТЧЕТ К ЛАБОРАТОРНОЙ РАБОТЕ №2 |
| «Разработка ресурса REST/JSON сервиса» |
| по дисциплине: Технология разработки серверных информационных систем |
|  |
|  |

РАБОТУ ВЫПОЛНИЛ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| СТУДЕНТГР. № | 4936 |  |  |  | Судак.А.А. |
|  |  |  | подпись, дата |  | фамилия, инициалы |

**Санкт-Петербург 2021**

**Задание:**

Выполните следующие задачи:

1. Подключите к проекту Maven фреймворк Spring (spring-boot)
2. Определите перечень Rest-сервисов, выполняющих те же действия, что и в лабораторной работе 1. Внимательно отнеситесь к вопросу какой HTTP метод использует тот или иной сервис и какие коды HTTP он может возвращать. Реализуйте эти сервисы.

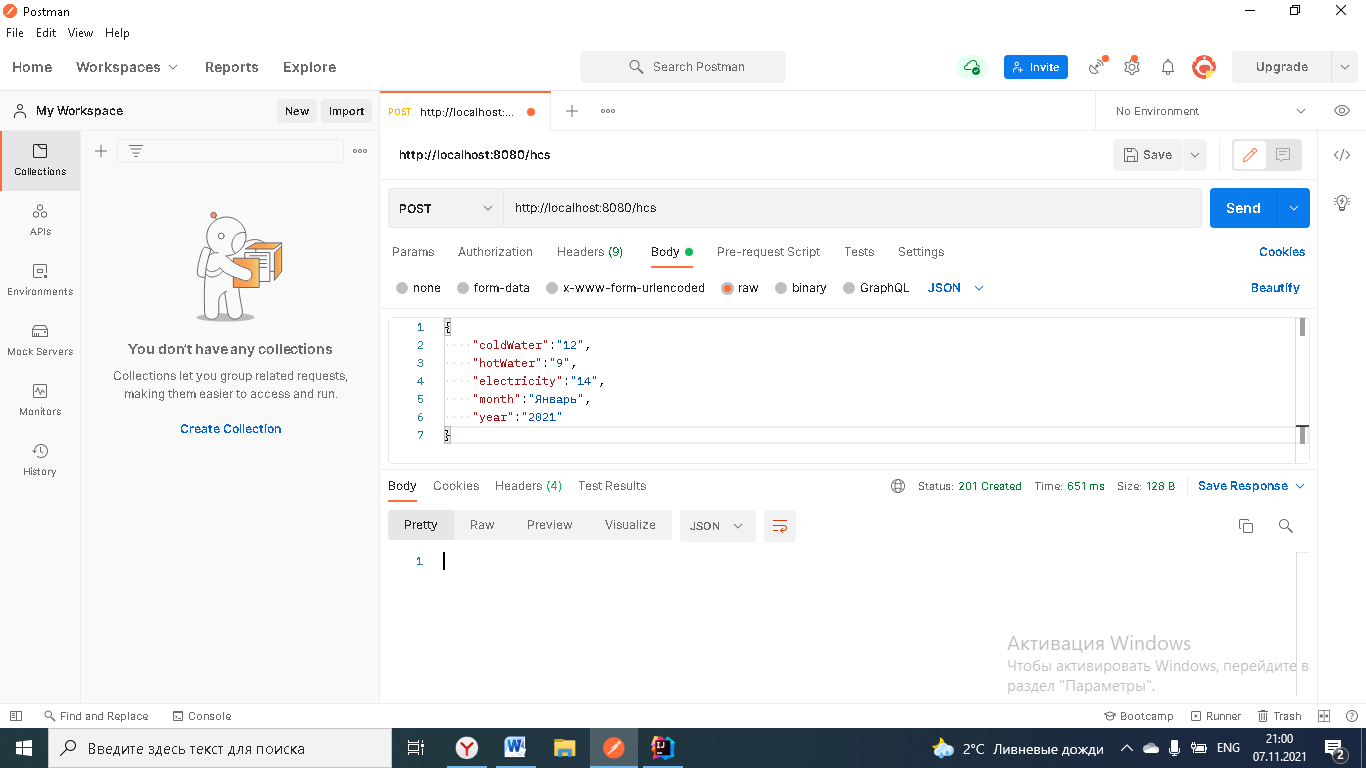
**Вариант 18:**

Учет показателей потребления электроэнергии и водоснабжения

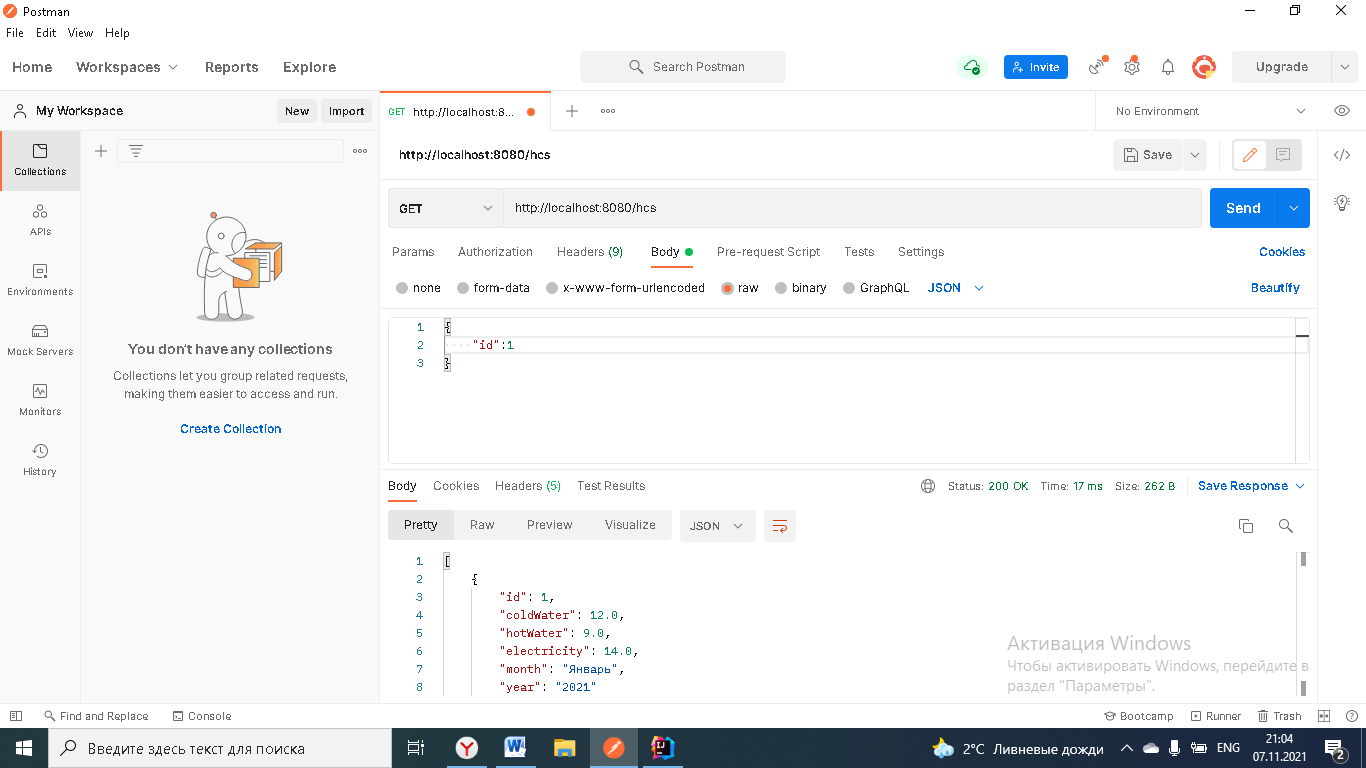
**Ход Работы:**

* GET – Bad Request(400) в случае некорректного запроса, Not Found(404) в случае, если такой элемент не был найден в базе, и OK(200) при успешном выполнении запроса.
* POST – Bad Request(400), если запрос некорректен, и Created(201) в случае успешного выполнение добавления.
* PUT – Bad Request(400), если запрос был некорректный или NOT MODIFIED, если такого элемента нет в базе, и OK(200) при успешном обновлении состояния выбранного элемента.
* DELETE - Bad Request(400), если запрос был некорректный, NOT MODIFIED, если не удалось удалить элемент в базе, и OK(200) при принятии запроса на обработку удаления (или просто начале процесса удаления).
* Проверим работы программы с помощью http запросов в Postman:

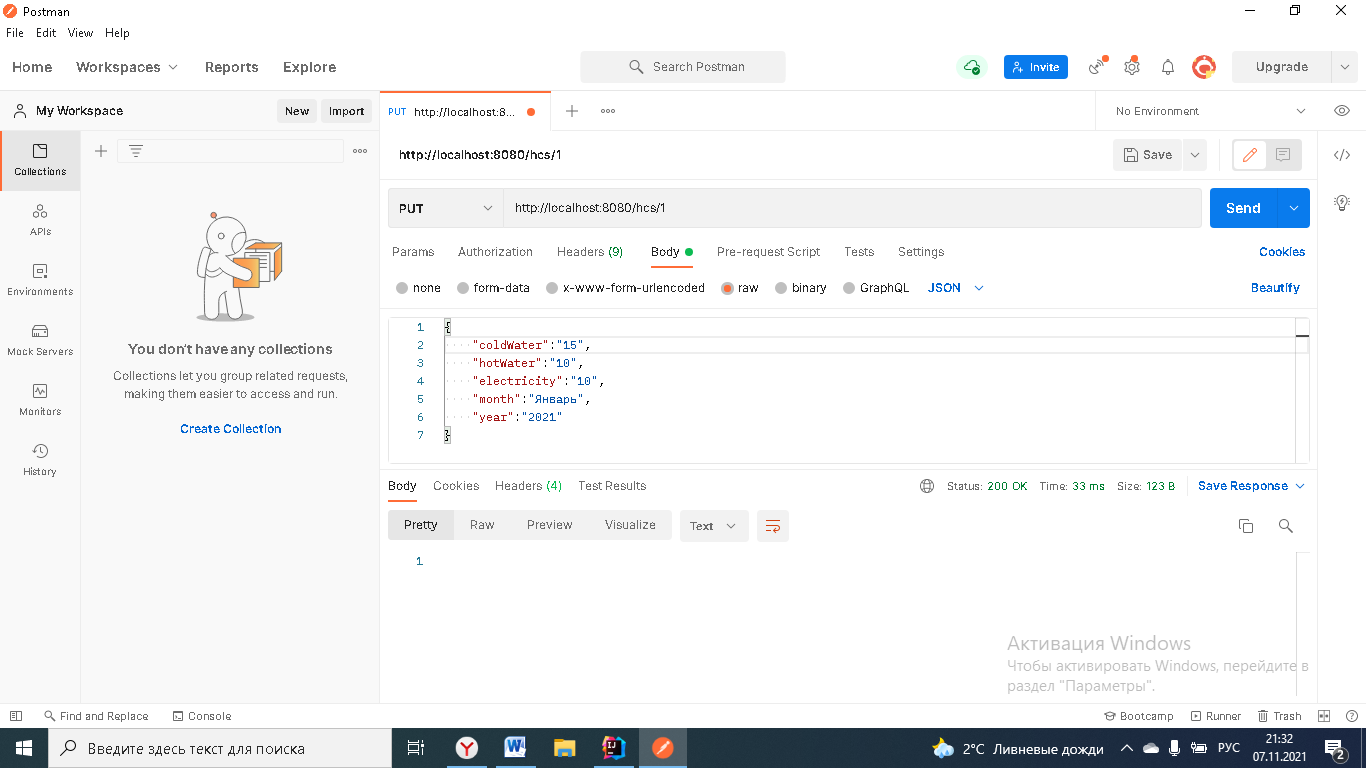
- POST запрос:



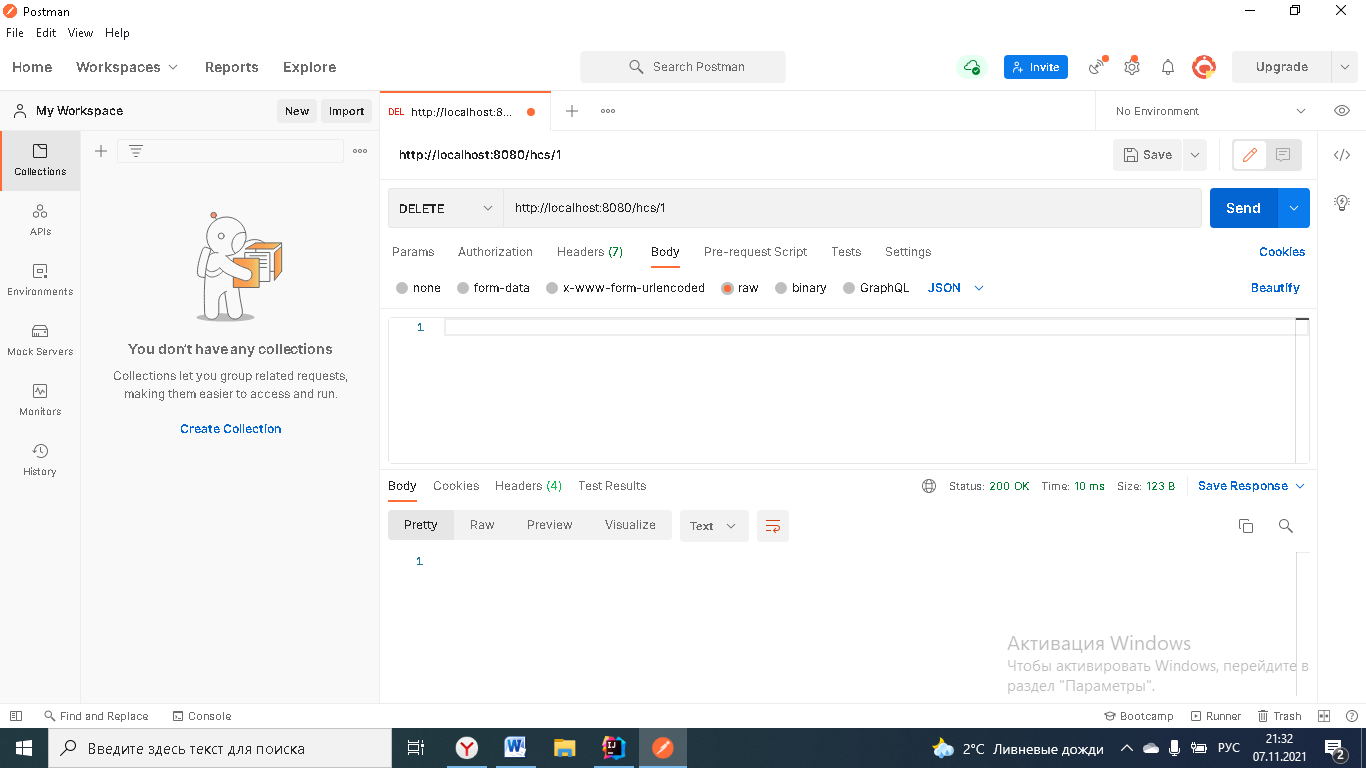
- GET запрос:



-PUT запрос:



-DELETE запрос:



1. Листинг основных файлов:

HCSController.java

package com.mobest1an.labs.LAB2.controllers;  
  
import com.mobest1an.labs.LAB2.services.MetersDataService;  
import com.mobest1an.labs.LAB2.models.MetersData;  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.http.HttpStatus;  
import org.springframework.http.ResponseEntity;  
import org.springframework.web.bind.annotation.\*;  
  
import java.util.List;  
  
@RestController  
@RequestMapping("/hcs")  
public class HCSController {  
  
 private final MetersDataService metersDataService;  
  
 @Autowired  
 public HCSController(MetersDataService metersDataService) {  
 this.metersDataService = metersDataService;  
 }  
  
 @GetMapping  
 public ResponseEntity<List<MetersData>> index() {  
 List<MetersData> allMetersData = metersDataService.index();  
  
 return allMetersData != null && !allMetersData.isEmpty()  
 ? new ResponseEntity<>(allMetersData, HttpStatus.*OK*)  
 : new ResponseEntity<>(HttpStatus.*NOT\_FOUND*);  
 }  
  
  
 @GetMapping("/{id}")  
 public ResponseEntity<MetersData> show(@PathVariable("id") int id) {  
 MetersData metersData = metersDataService.show(id);  
  
 return metersData != null  
 ? new ResponseEntity<>(metersData, HttpStatus.OK)  
 : new ResponseEntity<>(HttpStatus.NOT\_FOUND);  
 }  
  
 @PostMapping()  
 public ResponseEntity<?> create(@RequestBody MetersData metersData) {  
 metersDataService.save(metersData);  
 return new ResponseEntity<>(HttpStatus.*CREATED*);  
 }  
  
 @PutMapping("/{id}")  
 public ResponseEntity<?> update(@PathVariable("id") int id, @RequestBody MetersData metersData) {  
 boolean isUpdated = metersDataService.update(id, metersData);  
  
 return isUpdated  
 ? new ResponseEntity<>(HttpStatus.*OK*)  
 : new ResponseEntity<>(HttpStatus.*NOT\_MODIFIED*);  
 }  
  
 @DeleteMapping("/{id}")  
 public ResponseEntity<?> delete(@PathVariable("id") int id) {  
 boolean isDeleted = metersDataService.delete(id);  
  
 return isDeleted  
 ? new ResponseEntity<>(HttpStatus.OK)  
 : new ResponseEntity<>(HttpStatus.NOT\_MODIFIED);  
 }  
}

MetersData.java:

package com.mobest1an.labs.LAB2.models;  
  
public class MetersData {  
  
 private int id;  
 private double coldWater;  
 private double hotWater;  
 private double electricity;  
 private String month;  
 private String year;  
  
 public MetersData() {}  
  
 public MetersData(int id, double coldWater, double hotWater, double electricity, String month, String year) {  
 this.id = id;  
 this.coldWater = coldWater;  
 this.hotWater = hotWater;  
 this.electricity = electricity;  
 this.month = month;  
 this.year = year;  
 }  
  
 public int getId() {  
 return id;  
 }  
  
 public void setId(int id) {  
 this.id = id;  
 }  
  
 public double getColdWater() {  
 return coldWater;  
 }  
  
 public void setColdWater(double coldWater) {  
 this.coldWater = coldWater;  
 }  
  
 public double getHotWater() {  
 return hotWater;  
 }  
  
 public void setHotWater(double hotWater) {  
 this.hotWater = hotWater;  
 }  
  
 public double getElectricity() {  
 return electricity;  
 }  
  
 public void setElectricity(double electricity) {  
 this.electricity = electricity;  
 }  
  
 public String getMonth() {  
 return month;  
 }  
  
 public void setMonth(String month) {  
 this.month = month;  
 }  
  
 public String getYear() {  
 return year;  
 }  
  
 public void setYear(String year) {  
 this.year = year;  
 }  
}

DataServices.java:

package com.mobest1an.labs.LAB2.models;  
  
public class MetersData {  
  
 private int id;  
 private double coldWater;  
 private double hotWater;  
 private double electricity;  
 private String month;  
 private String year;  
  
 public MetersData() {}  
  
 public MetersData(int id, double coldWater, double hotWater, double electricity, String month, String year) {  
 this.id = id;  
 this.coldWater = coldWater;  
 this.hotWater = hotWater;  
 this.electricity = electricity;  
 this.month = month;  
 this.year = year;  
 }  
  
 public int getId() {  
 return id;  
 }  
  
 public void setId(int id) {  
 this.id = id;  
 }  
  
 public double getColdWater() {  
 return coldWater;  
 }  
  
 public void setColdWater(double coldWater) {  
 this.coldWater = coldWater;  
 }  
  
 public double getHotWater() {  
 return hotWater;  
 }  
  
 public void setHotWater(double hotWater) {  
 this.hotWater = hotWater;  
 }  
  
 public double getElectricity() {  
 return electricity;  
 }  
  
 public void setElectricity(double electricity) {  
 this.electricity = electricity;  
 }  
  
 public String getMonth() {  
 return month;  
 }  
  
 public void setMonth(String month) {  
 this.month = month;  
 }  
  
 public String getYear() {  
 return year;  
 }  
  
 public void setYear(String year) {  
 this.year = year;  
 }  
}

MetersDataService:

package com.mobest1an.labs.LAB2.services;  
  
import com.mobest1an.labs.LAB2.models.MetersData;  
import org.springframework.stereotype.Service;  
  
import java.util.ArrayList;  
import java.util.List;  
  
@Service  
public class MetersDataService implements DataServices {  
  
 private List<MetersData> metersDataList = new ArrayList<>();  
 private static int *RECORD\_COUNT*;  
  
 @Override  
 public List<MetersData> index() {  
 return metersDataList;  
 }  
  
 @Override  
 public MetersData show(int id) {  
 return metersDataList.stream().filter(metersData -> metersData.getId() == id).findAny().orElse(null);  
 }  
  
 @Override  
 public void save(MetersData metersData) {  
 metersData.setId(++*RECORD\_COUNT*);  
 metersDataList.add(metersData);  
 }  
  
 @Override  
 public boolean update(int id, MetersData updatedMetersData) {  
 MetersData metersDataToBeUpdated = show(id);  
 if (metersDataToBeUpdated != null) {  
 metersDataToBeUpdated.setColdWater(updatedMetersData.getColdWater());  
 metersDataToBeUpdated.setHotWater(updatedMetersData.getHotWater());  
 metersDataToBeUpdated.setElectricity(updatedMetersData.getElectricity());  
 metersDataToBeUpdated.setMonth(updatedMetersData.getMonth());  
 metersDataToBeUpdated.setYear(updatedMetersData.getYear());  
 return true;  
 }  
 return false;  
 }  
  
 @Override  
 public boolean delete(int id) {  
 return metersDataList.removeIf(metersData -> metersData.getId() == id);  
 }  
}

Вывод: В ходе работы были получены основные навыки разработки Rest сервисов с помощью фреймворков Spring, разобраны коды состояния HTTP и применены на практике.