ГУАП КАФЕДРА № 43

ОТЧЕТ

ЗАЩИЩЕН С ОЦЕНКОЙ ПРЕПОДАВАТЕЛЬ

|  |  |  |
| --- | --- | --- |
| ассистент |  | П.А.Степанов |
| должность, уч. степень, звание | подпись, дата | инициалы, фамилия |

|  |
| --- |
| ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ №2 |
| «Разработка ресурса REST/JSON сервиса» |
| по дисциплине: ТЕХНОЛОГИЯ РАЗРАБОТКИ СЕРВЕРНЫХ ИНФОРМАЦИОННЫХ СИСТЕМ |

РАБОТУ ВЫПОЛНИЛА

|  |  |  |
| --- | --- | --- |
| СТУДЕНТКА ГР. | 4232 | Е.А.Уткина |
|  | подпись, дата | инициалы, фамилия |

Санкт-Петербург

2024

1. Цель работы:

Разработка ресурса REST/JSON сервиса.

1. Задание:

1. Определите перечень Rest-сервисов, выполняющих те же действия, что и в лабораторной работ. Внимательно отнеситесь к вопросу какой HTTP метод использует тот или иной сервис и какие коды HTTP он может возвращать.

2. Опишите Ваш API с помощью OpenAPI v3. Приведите Ваш yaml файл в отчете.

3. Средствами swagger сгенерируйте сервер spring.

4. Возьмите REST контроллер из сгенерированного сервера, вставьте в свое приложение (не забыв поправить имя пакета).

5. Реализуйте приложение.

Вариант 14: Сдача недвижимости в аренду.

1. Листинг программы:

Класс Apartment:

package org.openapitools.model;  
  
import java.util.Objects;  
import com.fasterxml.jackson.annotation.JsonProperty;  
import io.swagger.v3.oas.annotations.media.Schema;  
  
  
import javax.annotation.Generated;  
  
*/\*\*  
 \* Apartment  
 \*/*@Generated(value = "org.openapitools.codegen.languages.SpringCodegen", date = "2024-09-27T18:19:31.380153700+03:00[Europe/Moscow]", comments = "Generator version: 7.8.0")  
public class Apartment {  
  
 private String city;  
  
 private String address;  
  
 private Integer price;  
  
 public Apartment(String city, String address, Integer price) {  
 this.city = city;  
 this.address = address;  
 this.price = price;  
 }  
  
 public Apartment city(String city) {  
 this.city = city;  
 return this;  
 }  
  
 */\*\*  
 \* apartment's city  
 \* @return city  
 \*/* @Schema(name = "city", description = "apartment's city", requiredMode = Schema.RequiredMode.*NOT\_REQUIRED*)  
 @JsonProperty("city")  
 public String getCity() {  
 return city;  
 }  
  
 public void setCity(String city) {  
 this.city = city;  
 }  
  
 public Apartment address(String address) {  
 this.address = address;  
 return this;  
 }  
  
 */\*\*  
 \* apartment's address  
 \* @return address  
 \*/* @Schema(name = "address", description = "apartment's address", requiredMode = Schema.RequiredMode.*NOT\_REQUIRED*)  
 @JsonProperty("address")  
 public String getAddress() {  
 return address;  
 }  
  
 public void setAddress(String address) {  
 this.address = address;  
 }  
  
 public Apartment price(Integer price) {  
 this.price = price;  
 return this;  
 }  
  
 */\*\*  
 \* rent's price per month  
 \*  
 \* @return price  
 \*/* @Schema(name = "price", description = "rent's price per month", requiredMode = Schema.RequiredMode.*NOT\_REQUIRED*)  
 @JsonProperty("price")  
 public Integer getPrice() {  
 return price;  
 }  
  
 public void setPrice(Integer price) {  
 this.price = price;  
 }  
  
 @Override  
 public boolean equals(Object o) {  
 if (this == o) {  
 return true;  
 }  
 if (o == null || getClass() != o.getClass()) {  
 return false;  
 }  
 Apartment apartment = (Apartment) o;  
 return Objects.*equals*(this.city, apartment.city) &&  
 Objects.*equals*(this.address, apartment.address) &&  
 Objects.*equals*(this.price, apartment.price);  
 }  
  
 @Override  
 public int hashCode() {  
 return Objects.*hash*(city, address, price);  
 }  
  
 @Override  
 public String toString() {  
 StringBuilder sb = new StringBuilder();  
 sb.append("class Apartment {\n");  
 sb.append(" city: ").append(toIndentedString(city)).append("\n");  
 sb.append(" address: ").append(toIndentedString(address)).append("\n");  
 sb.append(" price: ").append(toIndentedString(price)).append("\n");  
 sb.append("}");  
 return sb.toString();  
 }  
  
 */\*\*  
 \* Convert the given object to string with each line indented by 4 spaces  
 \* (except the first line).  
 \*/* private String toIndentedString(Object o) {  
 if (o == null) {  
 return "null";  
 }  
 return o.toString().replace("\n", "\n ");  
 }  
}

Класс ApartmentService:

package org.openapitools.services;  
  
import org.openapitools.model.Apartment;  
import org.springframework.stereotype.Service;  
  
import java.util.ArrayList;  
import java.util.List;  
  
@Service  
public class ApartmentService {  
 private final List<Apartment> apartments = new ArrayList<>();  
  
 public List<Apartment> getAllApartments() {  
 return apartments;  
 }  
  
 public Apartment addApartment(String city, String address, Integer price) {  
 Apartment newApartment = new Apartment(city, address, price);  
 apartments.add(newApartment);  
 return newApartment;  
 }  
  
 public Apartment deleteApartment(int id) {  
 if (id >= 0 && id < apartments.size()) {  
 return apartments.remove(id);  
 }  
 return null;  
 }  
}

Класс AddApiController:

package org.openapitools.api;  
  
import org.openapitools.model.Apartment;  
  
  
import org.openapitools.services.ApartmentService;  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.http.HttpStatus;  
import org.springframework.http.MediaType;  
import org.springframework.http.ResponseEntity;  
import org.springframework.stereotype.Controller;  
import org.springframework.web.bind.annotation.\*;  
import org.springframework.web.multipart.MultipartFile;  
import org.springframework.web.context.request.NativeWebRequest;  
  
import javax.validation.constraints.\*;  
import javax.validation.Valid;  
  
import java.util.List;  
import java.util.Map;  
import java.util.Optional;  
import javax.annotation.Generated;  
  
@Generated(value = "org.openapitools.codegen.languages.SpringCodegen", date = "2024-09-27T18:19:31.380153700+03:00[Europe/Moscow]", comments = "Generator version: 7.8.0")  
@Controller  
@RequestMapping("${openapi.apartmentRentOpen.base-path:}")  
public class AddApiController implements AddApi {  
  
 private final NativeWebRequest request;  
 private final ApartmentService apartmentService;  
  
 @Autowired  
 public AddApiController(NativeWebRequest request, ApartmentService apartmentService) {  
 this.request = request;  
 this.apartmentService = apartmentService;  
 }  
  
 @Override  
 public Optional<NativeWebRequest> getRequest() {  
 return Optional.*ofNullable*(request);  
 }  
  
 @Override  
 public ResponseEntity<Apartment> addApartment(String city, String address, Integer price) {  
 Apartment newApartment = apartmentService.addApartment(city, address, price);  
 return new ResponseEntity<>(newApartment, HttpStatus.*CREATED*);  
 }  
}

Класс ApartmentsApiController:

package org.openapitools.api;  
  
import org.openapitools.model.Apartment;  
  
  
import org.openapitools.services.ApartmentService;  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.http.HttpStatus;  
import org.springframework.http.MediaType;  
import org.springframework.http.ResponseEntity;  
import org.springframework.stereotype.Controller;  
import org.springframework.web.bind.annotation.PathVariable;  
import org.springframework.web.bind.annotation.RequestBody;  
import org.springframework.web.bind.annotation.RequestHeader;  
import org.springframework.web.bind.annotation.RequestMapping;  
import org.springframework.web.bind.annotation.CookieValue;  
import org.springframework.web.bind.annotation.RequestParam;  
import org.springframework.web.bind.annotation.RequestPart;  
import org.springframework.web.multipart.MultipartFile;  
import org.springframework.web.context.request.NativeWebRequest;  
  
import javax.validation.constraints.\*;  
import javax.validation.Valid;  
  
import java.util.ArrayList;  
import java.util.List;  
import java.util.Map;  
import java.util.Optional;  
import javax.annotation.Generated;  
  
@Generated(value = "org.openapitools.codegen.languages.SpringCodegen", date = "2024-09-27T18:19:31.380153700+03:00[Europe/Moscow]", comments = "Generator version: 7.8.0")  
@Controller  
@RequestMapping("${openapi.apartmentRentOpen.base-path:}")  
public class ApartmentsApiController implements ApartmentsApi {  
  
 private final NativeWebRequest request;  
  
 private final ApartmentService apartmentService;  
  
 @Autowired  
 public ApartmentsApiController(NativeWebRequest request, ApartmentService apartmentService) {  
 this.request = request;  
 this.apartmentService = apartmentService;  
 }  
  
 @Override  
 public Optional<NativeWebRequest> getRequest() {  
 return Optional.*ofNullable*(request);  
 }  
  
 @Override  
 public ResponseEntity<List<Apartment>> getApartment() {  
 List<Apartment> apartments = apartmentService.getAllApartments();  
 return ResponseEntity.*ok*(apartments);  
 }  
}

Класс DeleteApiController:

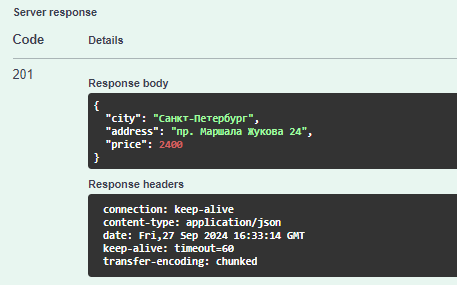
package org.openapitools.api;  
  
import org.openapitools.model.Apartment;  
  
  
import org.openapitools.services.ApartmentService;  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.http.HttpStatus;  
import org.springframework.http.MediaType;  
import org.springframework.http.ResponseEntity;  
import org.springframework.stereotype.Controller;  
import org.springframework.web.bind.annotation.PathVariable;  
import org.springframework.web.bind.annotation.RequestBody;  
import org.springframework.web.bind.annotation.RequestHeader;  
import org.springframework.web.bind.annotation.RequestMapping;  
import org.springframework.web.bind.annotation.CookieValue;  
import org.springframework.web.bind.annotation.RequestParam;  
import org.springframework.web.bind.annotation.RequestPart;  
import org.springframework.web.multipart.MultipartFile;  
import org.springframework.web.context.request.NativeWebRequest;  
  
import javax.validation.constraints.\*;  
import javax.validation.Valid;  
  
import java.util.List;  
import java.util.Map;  
import java.util.Optional;  
import javax.annotation.Generated;  
  
@Generated(value = "org.openapitools.codegen.languages.SpringCodegen", date = "2024-09-27T18:19:31.380153700+03:00[Europe/Moscow]", comments = "Generator version: 7.8.0")  
@Controller  
@RequestMapping("${openapi.apartmentRentOpen.base-path:}")  
public class DeleteApiController implements DeleteApi {  
  
 private final NativeWebRequest request;  
 private final ApartmentService apartmentService;  
  
 @Autowired  
 public DeleteApiController(NativeWebRequest request, ApartmentService apartmentService) {  
 this.request = request;  
 this.apartmentService = apartmentService;  
 }  
  
 @Override  
 public Optional<NativeWebRequest> getRequest() {  
 return Optional.*ofNullable*(request);  
 }  
  
 @Override  
 public ResponseEntity<Apartment> deleteApartment(Integer id) {  
 Apartment deletedApartment = apartmentService.deleteApartment(id);  
 if (deletedApartment != null) {  
 return new ResponseEntity<>(deletedApartment, HttpStatus.*OK*);  
 } else {  
 return new ResponseEntity<>(HttpStatus.*NOT\_FOUND*);  
 }  
 }  
}

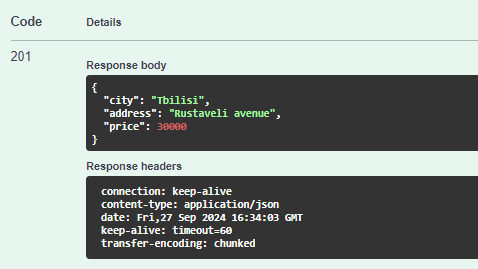
Файл openapi.yaml:

openapi: 3.0.3  
info:  
 description:  
 REST API for lab2  
 license:  
 name: Apache 2.0  
 url: https://www.apache.org/licenses/LICENSE-2.0.html  
 title: Apartment rent OpenAPI  
 version: 1.0.0  
servers:  
 - description: developer's server  
 url: http://localhost:8080/  
tags:  
 - description: "Apartment methods"  
 name: Apartment  
paths:  
 /apartments:  
 get:  
 description: "get all the apartments"  
 operationId: getApartment  
 responses:  
 "200":  
 content:  
 application/json:  
 schema:  
 items:  
 $ref: '#/components/schemas/Apartment'  
 type: array  
 description: ok  
 "500":  
 description: server's error  
 summary: get the list of all apartments  
 tags:  
 - Apartment  
 x-accepts:  
 - application/json  
 x-tags:  
 - tag: Apartment  
 /add:  
 post:  
 description: "add new apartment"  
 operationId: addApartment  
 parameters:  
 - description: city  
 explode: true  
 in: query  
 name: city  
 required: true  
 schema:  
 type: string  
 - description: address  
 explode: true  
 in: query  
 name: address  
 required: true  
 schema:  
 type: string  
 style: form  
 - description: price per month  
 explode: true  
 in: query  
 name: price  
 required: true  
 schema:  
 type: integer  
 responses:  
 "201":  
 content:  
 application/json:  
 schema:  
 $ref: '#/components/schemas/Apartment'  
 description: info about added apartment  
 "500":  
 description: server's error  
 summary: add an appartment  
 tags:  
 - Apartment  
 x-accepts:  
 - application/json  
 x-tags:  
 - tag: Apartment  
 /delete:  
 delete:  
 description: "delete apartment by id"  
 operationId: deleteApartment  
 parameters:  
 - description: id  
 explode: true  
 in: query  
 name: id  
 required: true  
 schema:  
 type: integer  
 style: form  
 responses:  
 "200":  
 content:  
 application/json:  
 schema:  
 $ref: '#/components/schemas/Apartment'  
 description: info about deleted apartment  
 "404":  
 description: not found  
 summary: delete an apartment  
 tags:  
 - Apartment  
 x-accepts:  
 - application/json  
 x-tags:  
 - tag: Apartment  
components:  
 schemas:  
 Apartment:  
 example:  
 city: Saint-Petersburg  
 address: Marshala Zhukova avenue 24  
 price: 2400  
 properties:  
 city:  
 description: apartment's city  
 type: string  
 address:  
 description: apartment's address  
 type: string  
 price:  
 description: rent's price per month  
 type: integer  
 type: object

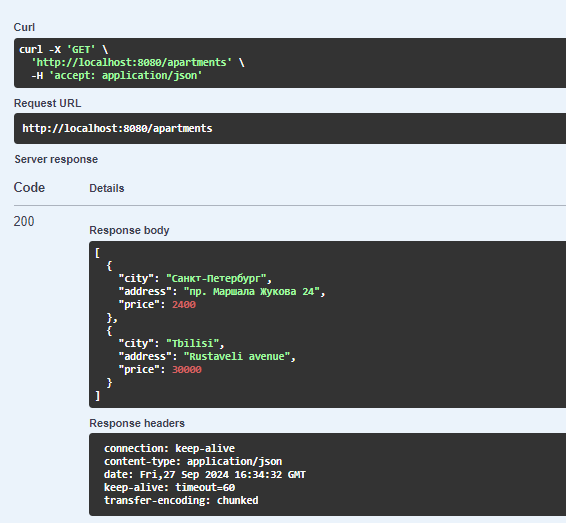
1. Пример выполнения программы:

Добавление квартиры:

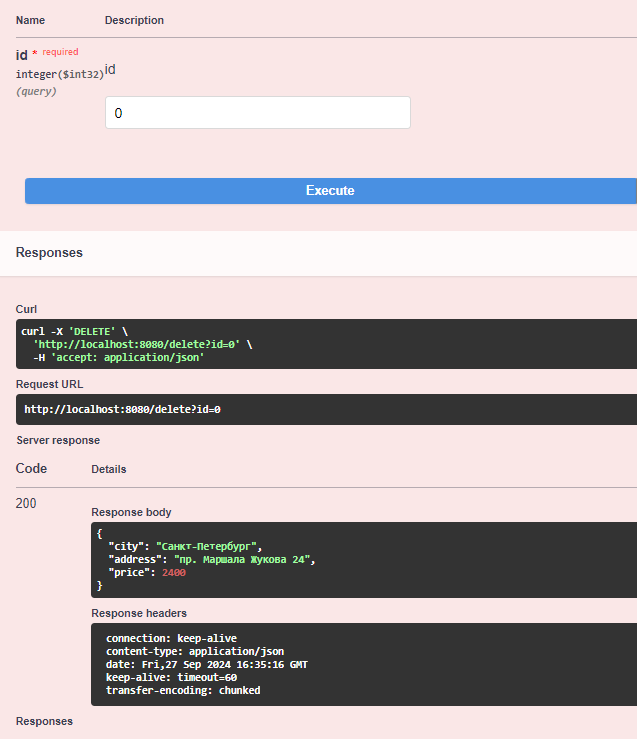




Получение списка квартир:



Удаление квартиры:



1. Вывод:

Разработано REST-приложение для учета сдачи недвижимости в аренду.