Университет ИТМО

Факультет программной инженерии и компьютерной техники

Лабораторная работа №1
Сервис-ориентированная архитектура

Выполнил: Кривошейкин Сергей

Группа № P34122

Преподаватель: Усков И. В.

г. Санкт-Петербург

2021

Задание

```
public class Flat {
    private Integer id; //Значение поля должно быть больше 0, Значение этого поля должно быть уникальным, Значение этого поля должно генерироваться автоматически
    private String name; //Поле не может быть null, Строка не может быть пустой
    private Coordinates coordinates; //Поле не может быть null
    private java.time.LocalDateTime creationDate; //Поле не может быть null, Значение этого поля должно генерироваться автоматически
    private Float area; //Значение поля должно быть больше 0
    private Long numberOfRooms; //Значение поля должно быть больше 0
    private Double price; //Значение поля должно быть больше 0
    private Float kitchenArea; //Поле может быть null, Значение поля должно быть больше 0
    private View view; //Поле может быть null
    private House house; //Поле не может быть null
}
public class Coordinates {
    private long x;
    private double y; //Максимальное значение поля: 22
}
public class House {
    private String name; //Поле не может быть null
    private Integer year; //Значение поля должно быть больше 0
    private Integer numberOfFloors; //Максимальное значение поля: 75, Значение поля должно быть больше 0
    private Integer numberOfFlatsOnFloor; //Значение поля должно быть больше 0
    private Integer numberOfLifts; //Значение поля должно быть больше 0
}
public enum View {
    YARD,
    BAD,
    GOOD,
    TERRIBLE;
}
```

Веб-сервис должен удовлетворять следующим требованиям:

- API, реализуемый сервисом, должен соответствовать рекомендациям подхода RESTful.
- Необходимо реализовать следующий базовый набор операций с объектами коллекции: добавление нового элемента, получение элемента по ИД, обновление элемента, удаление элемента, получение массива элементов.
- Операция, выполняемая над объектом коллекции, должна определяться методом HTTP-запроса.
- Операция получения массива элементов должна поддерживать возможность сортировки и фильтрации по любой комбинации полей класса, а также возможность постраничного вывода результатов выборки с указанием размера и порядкового номера выводимой страницы.
- Все параметры, необходимые для выполнения операции, должны передаваться в URL запроса.
- Данные коллекции, которыми управляет веб-сервис, должны храниться в реляционной базе данных.
- Информация об объектах коллекции должна передаваться в формате **xml**.
- В случае передачи сервису данных, нарушающих заданные на уровне класса ограничения целостности, сервис должен возвращать код ответа http, соответствующий произошедшей ошибке.
- Веб-сервис должен быть "упакован" в веб-приложение, которое необходимо развернуть на сервере приложений **WildFly**.

Помимо базового набора, веб-сервис должен поддерживать следующие операции над объектами коллекции:

- Вернуть количество объектов, значение поля price которых меньше заданного.
- Вернуть массив объектов, значение поля name которых содержит заданную подстроку.
- Вернуть массив объектов, значение поля name которых начинается с заданной подстроки.

Эти операции должны размещаться на отдельных URL.

Требования к клиентскому приложению:

- Клиентское приложение может быть написано на любом веб-фреймворке, который можно запустить на сервере helios.
- Клиентское приложение должно обеспечить полный набор возможностей по управлению объектами коллекции, предоставляемых веб-сервисом -- включая сортировку, фильтрацию и постраничный вывод.
- Клиентское приложение должно преобразовывать передаваемые сервисом данные в человеко-читаемый вид -- параграф текста, таблицу и т.д.
- Клиентское приложение должно информировать пользователя об ошибках, возникающих на стороне сервиса, в частности, о том, что сервису были отправлены невалиданые данные.

Веб-сервис и клиентское приложение должны быть развёрнуты на сервере helios.

API

- PUT
  http://localhost:8080/soa_lab1_back-1.0-SNAPSHOT/api/flats/23

  ```
  <flat>
  <name>mmmm</name>
  <coordinates>
  <x>0</x>
  <y>3.0</y>
  </coordinates>
  <view>BAD</view>
  <house>
  <name>ddf</name>
  </house>
  </flat>
  ```

- POST
  http://localhost:8080/soa_lab1_back-1.0-SNAPSHOT/api/flats/

  <flat>
  <name>bbbbb</name>
  <coordinates>
  <x>0</x>
  <y>3.0</y>
  </coordinates>
  <view>BAD</view>
  <house>
  <name>ddf</name>
  </house>
  </flat>

- GET
  http://localhost:8080/soa_lab1_back-1.0-SNAPSHOT/api/flats/

- GET
  http://localhost:8080/soa_lab1_back-1.0-SNAPSHOT/api/flats/17

- DELETE
  http://localhost:20080/soa_lab1_back-1.0-SNAPSHOT/api/flats/2

- GET
  http://localhost:20080/soa_lab1_back-1.0-SNAPSHOT/api/flats/pricelower?string=4

- GET
  http://localhost:20080/soa_lab1_back-1.0-SNAPSHOT/api/flats/namescontain?string=d

- GET
  http://localhost:20080/soa_lab1_back-1.0-SNAPSHOT/api/flats/namesstart?string=d

Код: https://github.com/Serzh721/SOA_lab1

Вывод: в ходе выполнения данной лабораторной работы я реализовал приложение по управлению коллекцией с использованием Java Servlets и ReactJS.