УНИВЕРСИТЕТ ИТМО

Факультет программной инженерии и компьютерной техники

Направление подготовки 09.03.04 Программная инженерия

Дисциплина «Сервис-ориентированная архитектура»

Лабораторная работа №1

Вариант 11103

Выполнили:

Бордун Анастасия

Иванов Евгений

P34111

Преподаватель

Кривоносов Е.Д.

Санкт-Петербург, 2023 г.

Условие лабораторной работы:

Лабораторная работа #1

Введите вариант: 11103

Внимание! У разных вариантов разный текст задания!

Разработать спецификацию в формате OpenAPI для набора веб-сервисов, реализующего следующую функциональность:

Первый веб-сервис должен осуществлять управление коллекцией объектов. В коллекции необходимо хранить объекты класса `Organization`, описание которого приведено ниже:

```
public class Organization {
    private int id; //Значение поля должно быть больше 0, Значение этого поля должно быть уникальным, Значение
    private String name; //Поле не может быть null, Строка не может быть пустой
    private Coordinates coordinates; //Поле не может быть null
    private java.util.Date creationDate; //Поле не может быть null, Значение этого поля должно генерироваться
    private Double annualTurnover; //Поле не может быть null, Значение поля должно быть больше 0
    private OrganizationType type; //Поле может быть null
    private Address officialAddress; //Поле может быть null
}


public class Coordinates {
    private Double x; //Поле не может быть null
    private long y;
}
public class Address {
    private String street; //Строка не может быть пустой, Поле не может быть null
    private String zipCode; //Поле может быть null
}
public enum OrganizationType {
    COMMERCIAL,
    PUBLIC,
    OPEN_JOINT_STOCK_COMPANY;
}
```

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.
- Информация об объектах коллекции должна передаваться в формате **json**.
- В случае передачи сервису данных, нарушающих заданные на уровне класса ограничения целостности, сервис должен возвращать код ответа http, соответствующий произошедшей ошибке.

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

- Удалить один (любой) объект, значение поля officialAddress которого эквивалентно заданному.
- Рассчитать сумму значений поля annualTurnover для всех объектов.
- Сгруппировать объекты по значению поля annualTurnover, вернуть количество элементов в каждой группе.

Эти операции должны размещаться на отдельных URL.

Второй веб-сервис должен располагаться на URL `/orgdirectory`, и реализовывать ряд дополнительных операций, связанных с вызовом API первого сервиса:

- `/filter/turnover/{min-annual-turnover}/{max-annual-turnover}` : отфильтровать организации по годовому обороту
- `/filter/employees/{min-employees-count}/{max-employees-count}` : отфильтровать организации по количеству сотрудников

Эти операции также должны размещаться на отдельных URL.

Для разработанной спецификации необходимо сгенерировать интерактивную веб-документацию с помощью Swagger UI. Документация должна содержать описание всех REST API обоих сервисов с текстовым описанием функциональности каждой операции. Созданную веб-документацию необходимо развернуть на сервере `helios`.

Код программы находится тут или https://github.com/Bordsiya/soa

Запуск программы: так как данная спецификация запускается как SpringBoot приложение, достаточно просто собрать приложение в jar с помощью *gradle jar*.

Далее перенос приложения на helios через утилиту scp в терминале:
scp -P 2222 build/libs/soa-0.0.1-SNAPSHOT <ИМЯ jar>
s311715@helios.se.ifmo.ru:/home/studs/s311715/SOA/lab_1.jar <DESTINATION>

Запуск на хелиосе: java -jar SOA/lab_1.jar
Теперь приложение доступно на 9099 порту helios(прокинуть порт на локальный 8080 например можно -> ssh -L 8080:localhost:9099 sXXXXXX@helios.se.ifmo.ru -p 2222)
Теперь можно открыть Swagger *http://localhost:8080/swagger-ui/index.html*

Вывод: мы ознакомились с спецификацией open api, которая достаточно удобная и читаемая как для программистов, так и для простых ребят, которые только http коды и знают(фронтендеры). Есть возможность создавать схемы и компоненты, а после ссылаться на них из разных сервисов, что даёт гибкость.