**Это первая лаба, она сделана.**

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 SpaceMarine, описание которого приведено ниже:

public class SpaceMarine {

private Long id; //Поле не может быть null, 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.ZonedDateTime creationDate; //Поле не может быть null, Значение этого поля должно генерироваться автоматически

private double health; //Значение поля должно быть больше 0

private AstartesCategory category; //Поле может быть null

private Weapon weaponType; //Поле не может быть null

private MeleeWeapon meleeWeapon; //Поле не может быть null

private Chapter chapter; //Поле не может быть null

}

public class Coordinates {

private Double x; //Поле не может быть null

private Integer y; //Поле не может быть null

}

public class Chapter {

private String name; //Поле не может быть null, Строка не может быть пустой

private String parentLegion;

private long marinesCount; //Значение поля должно быть больше 0, Максимальное значение поля: 1000

private String world; //Поле не может быть null

}

public enum AstartesCategory {

DREADNOUGHT,

AGGRESSOR,

SUPPRESSOR;

}

public enum Weapon {

BOLTGUN,

MELTAGUN,

MISSILE\_LAUNCHER;

}

public enum MeleeWeapon {

CHAIN\_AXE,

MANREAPER,

LIGHTING\_CLAW,

POWER\_FIST;

}

public class StarShip{

private String name;

private integer campacity;

private double length;

private double width;

private double height;

private StarShipType starShipType

}

public enum StarShipType{

FRIGATE,

BATTLESHIP,

CRUISER

}

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 xml.
* 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:

* Вернуть один (любой) объект, значение поля weaponType которого является минимальным.
* Вернуть количество объектов, значение поля chapter которых больше заданного.
* Вернуть массив объектов, значение поля weaponType которых меньше заданного.

Эти операции должны размещаться на отдельных URL.

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

* /create/{id}/{name} : создать новый десантный корабль и сохранить его в БД
* /{starship-id}/unload-all : высадить всех десантников с корабля

Эти операции также должны размещаться на отдельных URL.

Ссылка на спецификацию

<https://se.ifmo.ru/~s282365/soa_lab1/?urls.primaryName=Collection%20Web%20Service>

**Вторая лаба**

на основе разработанной в рамках лабораторной работы #1 спецификации реализовать два веб-сервиса и использующее их API клиентское приложение.

**Требования к реализации и развёртыванию сервисов:**

* Первый ("вызываемый") веб-сервис должен быть реализован на фреймворке JAX-RS и развёрнут в окружении под управлением сервера приложений Payara.
* Второй веб-сервис должен быть реализован на фреймворке JAX-RS, развёрнут в окружении под управлением ещё одного экземпляра сервера приложений Payara и вызывать REST API первого сервиса.
* **Для обоих сервисов необходимо реализовать все функции, задокументированные в API, в строгом соответствии со спецификацией!**
* Доступ к обоим сервисам должен быть реализован с по протоколу https с самоподписанным сертификатом сервера. Доступ к сервисам посредством http без шифрования должен быть запрещён.

**Требования к клиентскому приложению:**

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

Оба веб-сервиса и клиентское приложение должны быть развёрнуты на сервере helios.

Вот пример такой же лабы , только там другой сервер приложений и немного другие операции во 2 веб сервисе и другие дополнительные операции в 1 веб сервисе.

<https://github.com/EgorMIt/SOA-Lab2>