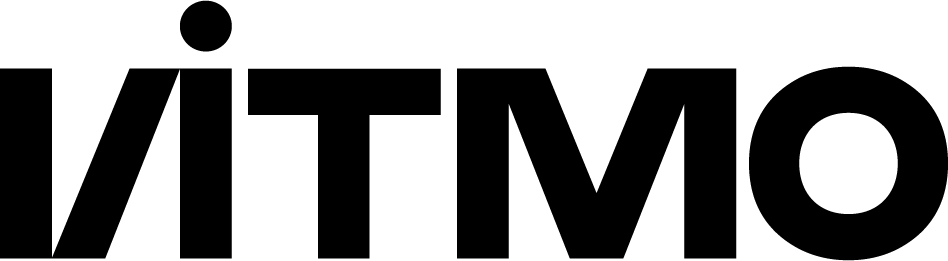
Федеральное государственное автономное образовательное учреждение высшего образования

«Научно-образовательная корпорация ИТМО»

Факультет Программной Инженерии и Компьютерной Техники

Лабораторная работа по СОА №1

Вариант 8907

Выполнил: Онишков Валерий Вячеславович

Группа: P3414

Преподаватель:Цопа Евгений Алексеевич

Санкт-Петербург, 2025г.

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 Vehicle, описание которого приведено ниже:

public class Vehicle {

private long 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.LocalDate creationDate; //Поле не может быть null, Значение этого поля должно генерироваться автоматически

private float enginePower; //Значение поля должно быть больше 0

private Integer numberOfWheels; //Поле может быть null, Значение поля должно быть больше 0

private Integer distanceTravelled; //Поле может быть null, Значение поля должно быть больше 0

private FuelType fuelType; //Поле может быть null

}

public class Coordinates {

private Float x; //Максимальное значение поля: 61, Поле не может быть null

private double y;

}

public enum FuelType {

KEROSENE,

DIESEL,

PLASMA;

}

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:

* Вернуть массив объектов, значение поля name которых содержит заданную подстроку.
* Вернуть массив объектов, значение поля name которых начинается с заданной подстроки.
* Вернуть массив объектов, значение поля fuelType которых больше заданного.

Эти операции должны размещаться на отдельных URL.

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

* /search/by-type/{type} : найти все транспортные средства заданного типа
* /fix-distance/{vehicle-id} : "скрутить" счётчик пробега транспортного средства с заданным id до нуля

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.

**Вопросы к защите лабораторной работы:**

1. Подходы к проектированию приложений. "Монолитная" и сервис-ориентированная архитектура.
2. Понятие сервиса. Общие свойства сервисов.
3. Основные принципы SOA. Подходы к реализации SOA, стандарты и протоколы.
4. Общие принципы построения и элементы сервис-ориентированных систем.
5. Понятие веб-сервиса. Определение, особенности, отличия от веб-приложений.
6. Категоризация веб-сервисов. RESTful и SOAP. Сходства и отличия, области применения.
7. RESTful веб-сервисы. Особенности подхода. Понятия ресурса, URI и полезной нагрузки (payload).
8. Виды RESTful-сервисов. Интерпретация методов HTTP в RESTful.
9. Правила именования ресурсов в RESTful сервисах.
10. Спецификация RESTful-сервисов. Стандарт OpenAPI.
11. Автодокументирование RESTful-сервисов. Swagger Editor, Swagger UI (и Swagger Codegen).
12. Архитектурный принцип HATEOAS.

Ссылка на swagger:

<https://github.com/OnishkovValera/SOA/blob/main/shop_service_openapi.yaml>  
https://github.com/OnishkovValera/SOA/blob/main/vehicle\_service\_openapi.yaml

Ссылка на helious:

<https://se.ifmo.ru/~s368608/swagger-ui/dist/index.html>

Вывод:

КРУТА(В ходе выполнения лабораторной работы был разработан веб-сервис, полностью соответствующий подходу RESTful.)