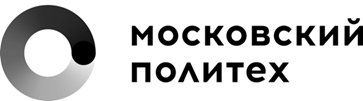
Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение высшего образования

«МОСКОВСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

(МОСКОВСКИЙ ПОЛИТЕХ)



ЛАБОРАТОРНАЯ РАБОТА 5

**Проектирование API. Проектирование SOAP API. ФОРМАТЫ XML И XSD, описание WSDL**

По курсу

**Проектирование Веб-сервисов**

ВыполнилКилеев С.И**.**

**студент группы 211-321**

Проверил

Пардаев А.А.

Москва, 2024

**Лабораторная работа 5**

**Цель работы:** 1. Познакомиться с различными видами API.

2. Изучить структуру WSDL и описать на этом языке API по протоколу SOAP.

**Задачи:**

Индивидуальная работа

1. Описать преимущества и недостатки SOAP, REST, GraphQL, WebSocket, gRPC

2. Свести в одну сравнительную таблицу SOAP, REST, GraphQL, WebSocket, gRPC по параметрам:

* формат данных;
* протокол;
* сложность;
* тип связи;
* сценарий использования.

3. Проанализировать предоставленный файл WSDL, разобрать и описать структуру и типы данных, описанные в нем.

4. Создать файл XSD, определяющий схему для всех типов данных, используемых в заданном файле WSDL.

5. Сформировать перечень операций для выполнения выбранной функции по результату лабораторной работы 2.

6. Составить на языке WSDL описание сервиса выбранной функции.

**Отчёт по выполнению:**

Описание преимуществ и недостатков SOAP, REST, GraphQL, WebSocket, gRPC.

### **SOAP (Simple Object Access Protocol)**

**Преимущества:**

* **Стандартизация:** SOAP строго стандартизирован, что обеспечивает согласованность взаимодействия между различными системами.
* **Безопасность:** Поддерживает WS-Security, обеспечивая высокий уровень безопасности, включая шифрование и аутентификацию.
* **Надежность:** Обеспечивает надежную доставку сообщений и поддержку транзакций.
* **Контрактно-ориентированный:** Использует WSDL (Web Services Description Language) для определения четкого контракта между клиентом и сервером.

**Недостатки:**

* **Сложность:** Более сложен в реализации и настройке по сравнению с другими протоколами.
* **Накладные расходы:** Использует XML, что увеличивает объем передаваемых данных и снижает производительность.
* **Меньшая гибкость:** Менее гибок в сравнении с REST или GraphQL, что ограничивает возможности адаптации к изменяющимся требованиям.
* **Ограниченная поддержка:** Меньше подходит для современных веб-приложений и мобильных платформ.

### **REST (Representational State Transfer)**

**Преимущества:**

* **Простота:** Легок в понимании и реализации, основывается на стандартных HTTP-методах (GET, POST, PUT, DELETE).
* **Гибкость:** Поддерживает различные форматы данных, такие как JSON, XML и другие.
* **Масштабируемость:** Хорошо подходит для масштабируемых и распределенных систем.
* **Широкая поддержка:** Большое количество инструментов, библиотек и сообществ для разработки и поддержки.

**Недостатки:**

* **Отсутствие строгого контракта:** Может привести к проблемам совместимости и интеграции между различными системами.
* **Безопасность:** Требует дополнительных мер для обеспечения безопасности, так как не предоставляет встроенных механизмов как SOAP.
* **Ограниченные возможности для сложных операций:** Трудно реализовать транзакции и сложные бизнес-операции.
* **Кэширование:** Хотя REST поддерживает кэширование, в некоторых случаях это может быть сложно реализовать эффективно.

### **GraphQL**

**Преимущества:**

* **Гибкость запросов:** Клиенты могут запрашивать только необходимые им данные, что уменьшает объем передаваемой информации.
* **Сильная типизация:** Схема GraphQL обеспечивает четкую структуру данных, упрощая разработку и интеграцию.
* **Поддержка сложных запросов:** Позволяет выполнять сложные запросы и получать связанные данные за один запрос.
* **Реальное время:** Поддерживает подписки для получения обновлений в реальном времени.

**Недостатки:**

* **Сложность реализации:** Требует более сложной настройки сервера и управления схемами.
* **Кэширование:** Из-за гибкости запросов может быть сложнее реализовать эффективное кэширование.
* **Производительность:** Сложные запросы могут негативно влиять на производительность сервера.
* **Меньшая зрелость:** По сравнению с REST, GraphQL менее зрел и имеет меньше инструментов и библиотек.

### **WebSocket**

**Преимущества:**

* **Двусторонняя коммуникация:** Обеспечивает постоянное соединение между клиентом и сервером для обмена данными в реальном времени.
* **Низкая задержка:** Подходит для приложений, требующих мгновенного обмена данными, таких как чаты, игры, финансовые приложения.
* **Эффективность:** Снижает накладные расходы на установку соединений по сравнению с традиционными HTTP-запросами.

**Недостатки:**

* **Сложность масштабирования:** Управление большим количеством постоянных соединений может быть трудоемким.
* **Совместимость:** Не все прокси и брандмауэры поддерживают WebSocket, что может вызывать проблемы с подключением.
* **Безопасность:** Требует дополнительных мер для обеспечения безопасности соединений.
* **Не подходит для всех сценариев:** Неэффективен для стандартных CRUD-операций.

### **gRPC (gRPC Remote Procedure Calls)**

**Преимущества:**

* **Высокая производительность:** Использует Protocol Buffers и HTTP/2, что обеспечивает низкую задержку и высокую пропускную способность.
* **Сильная типизация и контракты:** Использует .proto файлы для определения сервисов и сообщений, обеспечивая четкие контракты.
* **Поддержка стриминга:** Позволяет реализовывать одностороннее и двустороннее стриминг-соединения.

**Недостатки:**

* **Сложность:** Более сложен в настройке и использовании по сравнению с REST.
* **Ограниченная поддержка в браузерах:** Требует специальных прокси или решений для использования в браузерах.
* **Использование Protocol Buffers:** Могут быть сложны для разработчиков, не знакомых с этим форматом.
* **Меньшая популярность:** Не так широко распространен, как REST или GraphQL, что может ограничивать доступность ресурсов и инструментов.

Сравнительная таблица SOAP, REST, GraphQL, WebSocket, gRPC.

|  | SOAP | REST | GraphQL | WebSocket | gRPC |
| --- | --- | --- | --- | --- | --- |
| Формат данных | XML | Как правило, JSON, но может поддерживать несколько форматов | JSON | Как правило, JSON, но может поддерживать несколько форматов | Protocol Buffers (бинарный формат) |
| Протокол | HTTP/HTTPS, SMTP, XMPP, и другие | HTTP/HTTPS | HTTP/HTTPS | WS/WSS (WebSocket secure) | HTTP/2 |
| Сложность | Высокая | От низкого до умеренного | Умеренная | Умеренная | От умеренной до высокой |
| Тип связи | Односторонняя связь (запрос/ответ) | Односторонняя связь (запрос/ответ) | Односторонняя связь (запрос/ответ) | Двусторонняя связь (полный дуплекс) | Двунаправленная потоковая передача |
| Сценарий использования | Распределенные энтерпрайз- среды, унаследованные системы | Общее назначение, широкое применение | Когда требуется гибкая и эффективная загрузка данных | Приложения в режиме реального времени, когда серверу необходимо пушить обновления | Микросервисы, системы, критичные к производительности |

Анализ предоставленного файла WSDL, разбор и описание архитектуры.

Сервис BookServiceSOAP: состоит из трёх методов:  
1. AddBook - Добавление новой книги. Чтобы добавить новую книгу необходимо: 1. ID, 2. Title, 3. Author.

2. GetAllBook - Получение всех книг в рамках БД.

3. GetBook - Получить книгу по определённому ID.

В качестве ответа получаем объект book.

В addBookResponse мы в качестве ответа получаем объект book, где минимальное значение элементов = 0, а максимальное = 1

В getBookResponse мы в качестве ответа получаем объект book, где минимальное значение элементов = 0, а максимальное = 1,

В getAllBooksResponse мы в качестве ответа получаем объекты book, где минимальное значение элементов = 0, а максимальное = неограниченно.

Создать файл XSD, определяющий схему для всех типов данных, используемых в заданном файле WSDL.

Создан 1 файл xml с типами данных: book.xml

Сформировать перечень операций для выполнения выбранной функции по результату лабораторной работы 2.

Была выбрана функция: Просматривать и отвечать на вопросы пользователей

Операции:

1. Получить список вопросов: связанные с темой, на вход получаем тему, которую выбрал пользователь, а на выходе отображаем список всех вопросов, связанные с темой
2. Получить страницу вопроса: на входе мы имеем выбранный пользователем вопрос, а на выходе отображаем страницу вопроса
3. Добавить ответ на вопрос: на входе получаем сообщение в текстовом виде от пользователя, а на выходе отображаем ответ на странице вопроса

Составить на языке WSDL описание сервиса выбранной функции.

Создан 1 файл xml: commentary.xml