МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«МИРЭА – Российский технологический университет»
РТУ МИРЭА

Институт комплексной безопасности и специального приборостроения

Кафедра «Цифровые технологии обработки данных»

ОТЧЕТ
по практической работе
«№2. Схемы слабоструктурированных данных»
по дисциплине «Нереляционные системы управления базами данных»

Выполнил Смирнов И.А.
фамилия, имя, отчество

шифр 21Б0700 группа БСБО-11-21

Проверил к.т.н., доцент Ильин Д.Ю.
ученая степень, должность *фамилия, имя, отчество*

Москва 2023г.

Цель практической работы

Цель настоящей практической работы – научиться использовать инструменты описания структуры и валидации слабоструктурированных данных.

Задачи практической работы

Для достижения поставленной цели необходимо решить следующие задачи:

1. Спроектировать содержание набора данных согласно варианту задания, т. е. исходя из выбранной предметной области.
2. Реализовать схему слабоструктурированных данных для формата согласно варианту задания.
3. Подготовить несколько примеров слабоструктурированных данных, соответствующих результатам проектирования.
4. Подготовить несколько примеров слабоструктурированных данных, несоответствующих результатам проектирования или содержащих ошибки относительно ожидаемой схемы данных.
5. Выбрать программную библиотеку для проверки соответствия данных разработанной схеме.
6. Проверить соответствие корректных и некорректных примеров данных.
7. Подготовить ответы на контрольные вопросы.
8. Составить отчет о проведенной работе.

Вариант задания

Предметная область: Интернет-магазин

Тим схемы данных: JSON Schema

Ход работы

JSON Schema, по которой будет происходить валидация.

```
{
  "type": "object",
  "properties": {
    "book": {
      "type": "object",
      "properties": {
        "title": {
          "type": "string",
          "minLength": 1
        },
        "author": {
          "type": "string",
```

```
          "minLength": 1
        },
        "price": {
          "type": "number",
          "minimum": 0
        },
        "details": {
          "type": "object",
          "properties": {
            "genre": {
              "type": "string",
              "minLength": 1
            },
            "year": {
              "type": "integer",
              "minimum": 0
            },
            "availability": {
              "type": "boolean",
              "default": true
            }
          },
          "required": [
            "genre",
            "year",
            "availability"
          ],
          "additionalProperties": false
        },
        "reviews": {
          "type": "object",
          "properties": {
            "user": {
              "type": "string"
            },
            "rating": {
              "type": "number",
              "minimum": 0,
              "maximum": 5
            },
            "comment": {
              "type": "string"
            }
          },
          "required": [
            "user",
            "rating",
            "comment"
          ],
          "additionalProperties": false
        }
      },
      "required": [
        "title",
        "author",
        "price",
        "details",
        "reviews"
      ],
      "additionalProperties": false
    },
    "date": {
      "type": "string",
```

```
      "format": "date"
    },
    "email": {
      "type": "string",
      "format": "email",
      "minLength": 3
    }
  },
  "required": [
    "book",
    "date",
    "email"
  ],
  "additionalProperties": false
}
```

Листинг 1 – JSON Schema.

```
{
  "book": {
    "title": "The Shining",
    "author": "Stephen King",
    "price": 24.99,
    "details": {
      "genre": "Horror",
      "year": 1977,
      "availability": true
    },
    "reviews": {
      "user": "HorrorFan1",
      "rating": 4.8,
      "comment": "A terrifying masterpiece!"
    }
  },
  "date": "2023-12-03",
  "email": "horror@example.com"
}
```

Листинг 2 – пример валидных данных.

```
{
  "book": {
    "title": "Eugene Onegin",
    "author": "Alexander Pushkin",
    "price": "twenty dollars",
    "details": {
      "genre": "Verse Novel",
      "availability": null
    }
  },
  "date": "2023-12-05: 11-30-00",
  "email": "literature@example.com"
}
```

Листинг 3 – пример не валидных данных.

Код программы для проверки валидности данных. Для проверки валидности использовалась библиотека Ajv и ajv-formats для добавления поддержки стандартных форматов типа email и data.

```
const schema = require('./schema.json');
const val_ex_1 = require('./example/valid/ex.json');
const val_ex_2 = require('./example/valid/ex_2.json');
const inval_ex_1 = require('./example/invalid/ex.json');
```

```
const inval_ex_2 = require('./example/invalid/ex_2.json');

const Ajv = require("ajv");
const addFormats = require("ajv-formats");


const ajv = new Ajv();
addFormats(ajv)

const validData = [
    val_ex_1, val_ex_2
];
const invalidData = [
    inval_ex_1, inval_ex_2
];
const validate = ajv.compile(schema);

validData.forEach((data, index) => {
    const valid = validate(data);
    console.log(`Валидные данные ${index + 1}: ${valid ? "Верно" : "Неверно"}`);
});

invalidData.forEach((data, index) => {
    const valid = validate(data);
    console.log(`Невалидные данные ${index + 1}: ${valid ? "Неверно" :
"Верно"}`);
});
```

Листинг 4 – код программы.

Результат

В результате работы валидированные данные прошли проверку, не валидированные не прошли.

```
"C:\Program Files\nodejs\node.exe" C:\Users\batis\Desktop\NoSql\Laba_2\index.js
Валидные данные 1: Верно
Валидные данные 2: Верно
Невалидные данные 1: Верно
Невалидные данные 2: Верно

Process finished with exit code 0
```

Рис 1. Результат программы.

Контрольные вопросы

1. Каковы области применения схем слабоструктурированных форматах?

   Области применения схем данных в слабоструктурированных форматах:

   - Валидация данных: Схемы данных позволяют проверять структуру и формат данных, что особенно важно для слабоструктурированных форматов, где не так явно задана структура.
   - Документация: Схемы предоставляют средство документирования структуры данных, что упрощает понимание и использование данных.
   - Согласованность данных: Использование схем помогает обеспечить согласованность данных между различными системами и приложениями

2. Как в схеме данных задаются ограничения по допустимым значениям?

   - Ограничения на типы данных: Схема может определять допустимые типы данных для элементов.
   - Ограничения на длину и формат строк: Схема может указывать максимальную длину строк, формат даты и времени и другие ограничения.
   - Ограничения на значения элементов: С помощью ограничений, таких как minOccurs, maxOccurs, minInclusive, maxInclusive и других.

3. Какие библиотеки могут использоваться для проверки соответствия схеме данных?

   - Встроенные библиотеки языка программирования: Например, в C# используется библиотека System.Xml.Schema для валидации XML по XSD-схеме.
   - Xerces (Apache XML Project): Популярная библиотека для валидации XML-документов с использованием схем.

- Libxml2: Библиотека для работы с XML, включая валидацию по схеме.

4. Каким образом возможна интеграция средств проверки данных в информационную систему?
    - API и библиотеки: Интегрировать проверку данных можно с использованием соответствующих API и библиотек, предоставляемых средствами языка программирования.
    - Сервисы валидации: Возможна интеграция с веб-сервисами, предоставляющими функциональность валидации данных.
5. Какие средства проверки схемы данных, помимо рассмотренных, сществуют?
    - JSON Schema: Для валидации JSON-данных.
    - Relax NG: Альтернативный язык схем для валидации XML.
    - Schematron: Позволяет выражать более сложные правила валидации, не ограничиваясь только структурой данных.

Вывод

В ходе выполнения практической работы я успешно освоил инструменты описания структуры и валидации слабоструктурированных данных. Моя цель заключалась в изучении использования библиотеки Ajv для работы с JSON-схемами.

В начале работы я спроектировал структуру набора данных для интернет-магазина книг, учитывая различные категории документов: информацию о книге, дату и электронную почту. Затем реализовал схему данных в формате JSON, включая описание типов данных, форматов и обязательных полей.

Создал примеры валидных данных, соответствующих разработанной схеме, а также примеры невалидных данных для тестирования. Выбрал и использовал библиотеку Ajv для проверки соответствия данных схеме, проведя проверку как корректных, так и некорректных примеров.

В итоге проделанной работы я приобрел практические навыки работы с JSON-схемами, что позволило мне успешно выполнить поставленные задачи и более глубоко разобраться в процессе валидации данных в слабоструктурированных форматах.