Тестовое задание на позицию «Тестировщик API»

# Часть 1: Исследовательское тестирование (Exploratory Testing)

**Задача:** Изучить работу эндпоинта GET /posts и определить его характеристики.

**Требуется проверить:**

1. Структуру успешного ответа (поля, типы данных, форматы)
2. Наличие и функциональность параметров запроса:
   * userId (фильтрация по пользователю)
   * \_limit (ограничение количества результатов)
   * \_page (пагинация)
3. HTTP-заголовки ответа:
   * Content-Type
   * Наличие заголовков кеширования
   * Прочие служебные заголовки
4. Реакцию на нестандартные ситуации:
   * Запрос с userId=999 (несуществующий пользователь)
   * Запрос с некорректными параметрами (?invalid\_param=value)
   * Запрос с превышением разумного лимита (?\_limit=1000)

**Вопрос для анализа:** Как получить первые 5 постов пользователя с userId=7? Составить полный URL и описать ожидаемый результат.

ВЫПОЛНЕНИЕ

1. Массив данных:

[

{

"userId": int,

"id": int,

"title": "string",

"body": "string"

},

…

]

2. Наличие и функциональность параметров запроса:

|  |  |
| --- | --- |
| userId (фильтрация по пользователю) | Есть |
| ограничение количества результатов | Есть |
| пагинация | Есть |

3. HTTP-заголовки ответа:

|  |  |
| --- | --- |
| Content-Type | Стандарт (application/json; charset=UTF-8) |
| Наличие заголовков кеширования | Не требуется |
| Прочие служебные заголовки | Не требуются |

4. Реакцию на нестандартные ситуации:

|  |  |
| --- | --- |
| Запрос с userId=999 (несуществующий пользователь) | Стандарт (application/json; charset=UTF-8) |
| Запрос с некорректными параметрами (?invalid\_param=value) | Не влияет на результат, Status Code 200 |
| Запрос с превышением разумного лимита (?\_limit=1000) | Выдает все 100 элементов массива, SC 200 |

**Вопрос для анализа:** Как получить первые 5 постов пользователя с userId=7? Составить полный URL и описать ожидаемый результат.

**Ответ :**

<https://jsonplaceholder.typicode.com> /posts?userId=7&\_limit=5

# Часть 2: Тестирование создания ресурса (POST)

**Задача:** Протестировать эндпоинт POST /posts на различные сценарии.

**Тестовые сценарии:**

1. **Успешное создание ресурса**
   * Отправить валидный JSON с полями: title, body, userId
   * Проверить: статус-код, тело ответа, наличие ID в ответе
   * Заголовок: Content-Type: application/json
2. **Обработка невалидных данных**
   * Отсутствие обязательного заголовка Content-Type
   * Неправильный тип данных (userId: "string" вместо числа)
   * Неполные данные (отсутствие поля title)
   * Пустой JSON-объект ({})
   * Для каждого случая зафиксировать код ответа и сообщение об ошибке
3. **Проверка на идемпотентность**
   * Отправить два идентичных POST-запроса
   * Зафиксировать: отличаются ли ID в ответах
   * Сделать вывод о соответствии поведения стандартам REST

ВЫПОЛНЕНИЕ

1. Успешное создание ресурса

|  |  |
| --- | --- |
| Status code | 201 – created |
| Body | {      "title": "igor",      "body": "tester",      "userId": 11,      "id": 101  } |
| Id | есть |

2. Обработка невалидных данных

|  |  |
| --- | --- |
| Отсутствие обязательного заголовка Content-Type | |
| Status code | 201 – created |
| Body | {      "id": 101  } |
| Id | есть |
| Неправильный тип данных (userId: "string" вместо числа) | |
| Status code | 201 – created |
| Body | {  "title": "igor",  "body": "tester",  "userId": "hello",  "id": 101  } |
| Неполные данные (отсутствие поля title) | |
| Status code | 201 – created |
| Body | {  "body": "tester",  "userId": "hello",  "id": 101  } |
| Пустой JSON-объект ({}) | |
| Status code | 201 – created |
| Body | {      "id": 101  } |
|  |  |

# Часть 3: Документирование дефекта

**Задача:** Оформить выявленную особенность работы API в виде формального отчета о дефекте.

**Структура отчета:**

* **Заголовок:** Лаконичное описание проблемы
* **Описание:** Суть проблемы и ее влияние
* **Окружение:** Используемые инструменты и версии
* **Шаги воспроизведения:** Точная последовательность действий
* **Фактический результат:** Поведение системы в настоящее время
* **Ожидаемый результат:** Ожидаемое поведение согласно стандартам
* **Серьезность/Приоритет:** Обоснование выбранного уровня критичности
* **Доказательства:** Скриншоты или экспорт запросов

ВЫПОЛНЕНИЕ

|  |  |
| --- | --- |
| ID | BR\_001 |
| Summary | Метод POST при создании ресурса не отрабатывает ошибки |
| Severity | Minor |
| Priority | Низкий |
| Description | При отправки http запроса методом POST с невалидным телом запроса сервер отвечает: статус код “201 Created” |
| Environment | Postman |
| Steps | Привожу список проверок (запросов), статус код ответа на которые был 201 :  1. Отправка запроса при отсутствии обязательного заголовка Content-Type  2. Отправка запроса при неправильном типе данных (userId: "string" вместо числа)  3. Отправка запроса при неполных данных (отсутствие поля title)  4. Отправка запроса с пустым JSON-объектом ({}) |
| Actual result | Status code: 201 Created |
| Expected result | Status code: 4xx |
| Add info | Ссылка на коллекцию (метод POST – имя POST\_/posts)  https://grzhenevskijigor.postman.co/workspace/Grzhenevskij-Igor's-Workspace~7e75ece6-41e0-430e-8ee6-9ac0c748c74a/collection/45056689-be4701cf-9d8a-43f3-8b1f-28eb76c111c0?action=share&creator=45056689 |