**Тест-план по тестированию API “Restful-booker”**

**СОДЕРЖАНИЕ**

История изменений документа (Revision History)

1. Введение (Introduction)

2. Объем работ (Scope of work)

3. Приемочные критерии (Quality and Acceptance Criteria)

4. Факторы, влияющие на успешное завершение проекта (Critical Success Factors)

5. Оценка рисков (Risk Assessment)

6. Ресурсы (Resources)

7. Документация (Test Documentation)

8. Тестовая стратегия (Test Strategy)

9. Расписание (Test Schedule)

**ИСТОРИЯ ИЗМЕНЕНИЙ ДОКУМЕНТА**

|  |  |  |
| --- | --- | --- |
| **Дата** | **Автор** | **Версия и вид изменений** |
| 20.01.2023 | Божкова И.В. | 1.0 Создание |
|  |  |  |

**1.** **ВВЕДЕНИЕ (INTRODUCTION)**

Данный документ описывает методы и подходы, которые будут применяться при тестировании API Restful-booker.

API Resrful-booker – игровая площадка, предназначенная для тестирования функций создания, чтения, обновления и удаления данных.

C детальной документацией к API можно ознакомиться, пройдя по ссылке – <https://restful-booker.herokuapp.com/apidoc/index.html>. В основе тестирования мы будем использовать данную документацию.

Цель составления тест-плана:

- определить объем работ, какие функции API будут протестированы;

- определить критерии оценки качества работы на проекте;

- описать правила, которых необходимо придерживаться, чтобы проект был вовремя завершен;

- оценить риски, которые могут возникнуть на проекте;

- определить ресурсы, используемые на проекте;

- описать перечень документов, которые должны быть подготовлены за время работы над проектом;

- разработать и описать тестовую стратегию по процессу тестирования;

- определить временные рамки процесса тестирования;

**2. ОБЪЕМ РАБОТ (SCOPE OF WORK)**

Согласно тест-плану будет проводиться тестирование следующих функций API, представленных в Таблице 1.

Таблица 1

|  |  |  |
| --- | --- | --- |
| **Функция** | **Операция/Метод** | **Описание** |
| 1. Аутентификация (Auth) | Создать токен (CreateToken)  *Метод - POST* | Создает новый токен для доступа к PUT и DELETE/booking  <https://restful-booker.herokuapp.com/apidoc/index.html#api-Auth-CreateToken> |
| 2. Бронирование (Booking) | Получить идентификаторы всех бронирований (GetBookingids)  *Метод - GET* | Возвращает идентификаторы всех бронирований, существующих в API. Может принимать необязательные строки запроса для поиска и возврата подмножества идентификаторов бронирования.  <https://restful-booker.herokuapp.com/apidoc/index.html#api-Booking-GetBookings> |
| Получить конкретное бронирование по идентификатору (GetBooking)  *Метод - GET* | Возвращает конкретное бронирование на основе предоставленного идентификатора бронирования  <https://restful-booker.herokuapp.com/apidoc/index.html#api-Booking-GetBookings> |
| Создать новое бронирование (CreateBooking)  *Метод - POST* | Создает новое бронирование в API <https://restful-booker.herokuapp.com/apidoc/index.html#api-Booking-CreateBooking> |
| Обновить текущее бронирование (UpdateBooking)  *Метод - PUT* | Обновляет текущее бронирование <https://restful-booker.herokuapp.com/apidoc/index.html#api-Booking-UpdateBooking> |
| Частично обновить бронирование (PartialUpdateBooking)  *Метод - PATCH* | Частично обновляет текущее бронирование  <https://restful-booker.herokuapp.com/apidoc/index.html#api-Booking-PartialUpdateBooking> |
| Удалить бронирование (DeleteBooking)  *Метод - DELETE* | Удаляет бронирование. Возвращает идентификаторы всех бронирований. Может принимать необязательные строки запроса для поиска и возврата подмножества идентификаторов бронирования  <https://restful-booker.herokuapp.com/apidoc/index.html#api-Booking-DeleteBooking> |
| 3. Пакетный межсетевой щуп (PING – Package Internet Groper) | Проверить работоспособность API (HealthCheck) | Простая конечная точка проверки работоспособности, чтобы убедиться, что API запущен и работает  <https://restful-booker.herokuapp.com/apidoc/index.html#api-Ping-Ping> |

Также возможно проведедение нефункционального тестирования:

- нагрузочное тестирование,

- стресс тестирование,

- тестирование безопасности.

Характеристика оборудования и операционной системы, на которых будет проводиться тестирование API:

- персональный компьютер VivoBook\_ASUS Laptop TP412FAC\_TP412FA, Intel(R) Core(TM) i5-10210U CPU @ 1.60GHz 2.11 GHz;

- операционная система Window 11 Pro (x64), версия 21H2;

- установлен браузер Google Chrome v. 109.0.5414.75 (Официальная сборка), (64 бит).

**3. ПРИЕМОЧНЫЕ КРИТЕРИИ (QUALITY AND ACCEPTANCE CRITERIA)**

Критерии оценки качества:

- веб-сервис должен работать в соответствии с документацией API - <https://restful-booker.herokuapp.com/apidoc/index.html#api-Auth>;

- все основные функции, описанные в документации к API, должны быть покрыты тестами;

- производительность должна быть покрыта тестами;

- найденные дефекты должны быть задокументированы.

**4. ФАКТОРЫ, НЕОБХОДИМЫЕ ДЛЯ УСПЕШНОГО ЗАВЕРШЕНИЯ ПРОЕКТА (CRITICAL SUCCESS FACTORS)**

Для проведения тестирования и успешного завершения проекта необходимо проверить и принять во внимание следующие факторы:

- документация к API «заморожена», в ней не появляются новые функции либо не происходит изменение старых;

- имеется полноценный доступ к инструментам и программным средствам, посредством которых будет проводится тестирование;

- имеется четкий график работы: анализа данных, подготовки документации, проведения тестирования, документирования дефектов, устранения ошибок. Работы проводятся согласно графику.

**5. ОЦЕНКА РИСКОВ (RISK ASSESSMENT)**

Риск – это существующий или развивающийся фактор процесса, который обладает потенциально негативным воздействием на этот процесс. Любой риск может перерасти в проблему. Ниже в таблице 2 предоставлен список рисков с вероятностью возникновения и влиянием на работу веб-сервиса при тестировании, а также вариантом минимизации этих рисков.

Таблица 2

|  |  |  |  |
| --- | --- | --- | --- |
| **Риск** | **Вероятность** | **Влияние** | **Минимизация рисков** |
| Изменения в документации к API, которые не были запланированы | Очень низкая | Увеличение затрат на тестирование, увеличение длительности проекта | Отслеживание изменений в программах по контролю за задачами; оценка временных рамок проекта с учетом возможных рисков. |
| Сбой в работе сервера, базы данных | Низкая | Невозможность тестирования, увеличение времени и затрат на проект | Использование кластера сервера/кластера БД. |
| Изменения в работе веб-сервиса, которые не были запланированы (изменение кода) | Очень низкая | Увеличение затрат на проект, увеличение длительности проекта | Отслеживание изменений в программах по контролю за задачами; оценка временных рамок проекта с учетом возможных рисков. |

**6. РЕСУРСЫ (RESOURCES)**

Информация о лицах, проводящих тестирование, ролях и обязанностях приведена в таблице 3.

Таблица 3

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **№** | **Роль на проекте** | **ФИО** | **Обязанности** | **Локация** |
| 1. | QA инженер | Божкова И.В. | 1. Составление плана работ.  2. Анализ документации к API  3. Создание тестовой документации.  4. Составление коллекции запросов для Postman.  5. Проведение функционального, нагрузочного тестирования,  6. Составление TRR – отчета о результатах тестирования. | г. Могилев, РБ |

Инструментарий для тестирования приведен в таблице 4.

Таблица 4

|  |  |  |
| --- | --- | --- |
| **№** | **Инструмент** | **Описание** |
| 1. | MS Office | Написание тест-плана, оформление отчета о тестировании |
| 2. | TestRail | Составление тест-кейсов |
| 3. | Jira | Составление баг-репортов |
| 4. | Postman | Составление коллекции запросов, функциональное тестирование API |
| 5. | JMeter | Проведение нагрузочного тестирования, стресс тестирования |
| 6. | GitHub | Ресурс для сохранения артефактов в репозитории |

**7. ТЕСТОВАЯ ДОКУМЕНТАЦИЯ (TEST DOCUMENTATION)**

Перечень документов, которые планируется подготовить за время/во время тестирования, представлен в таблице 5.

Таблица 5

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **Название** | **Ответственное лицо** | **Когда оформляется** |
| 1. | “Restful-booker” тест-план | Божкова И.В. | Единожды перед началом тестирования |
| 2. | Тест-сьют «Авторизация» | Божкова И.В. | После тест-плана, перед началом тестирования |
| 3. | Тест-сьют «Бронирование» | Божкова И.В. | После тест-плана, перед началом тестирования |
| 4. | Тест-сьют «Проверка работоспособности» (Ping) | Божкова И.В. | После тест-плана, перед началом тестирования |
| 7. | Коллекция запросов GET, POST, PUT, PATCH, DELETE | Божкова И.В. | После готовых тест-сьютов, перед началом тестирования |
| 8. | Баг-репорты | Божкова И.В. | После обнаружения багов |
| 9. | Отчет о результатах тестирования (TRR) | Божкова И.В. | После проведения тестирования |

**8. ТЕСТОВАЯ СТРАТЕГИЯ (TEST STRATEGY)**

В этой главе описаны тестовая стратегия с методами и видами тестирования, а также высокоуровневые сценарии тестирования, на основании которых будут составлены наборы тест-кейсов и коллекций и проведено тестирование API “restful-booker”.

**Метод тестирования (по знанию кода):**

- API будет протестирован с использованием подхода «черный ящик», при котором не используется знание о внутреннем устройстве (коде) тестируемого объекта.

**Виды тестирования (по целям):**

- функциональное,

- нефункциональное.

Основные задачи функционального тестирования API:

- убедиться, что реализация API работает правильно, без ошибок;

- гарантировать, что реализация API работает в соответствии с документацией к API.

Нефункциональное тестирование проверит корректность работы нефункциональных требований:

- нагрузочное тестирование (определим поведение системы под нагрузкой)

Ручное функциональное тестирование рассматривается в качестве основного применяемого вида тестирования API.

Применяемые **уровни функционального тестирования (по степени важности):**

- дымовое тестирование (тестирование работы самых важных функций – *«аутентификация», «бронирование»);*

- тестирование критического пути (тестирование часто используемых функций веб-сервиса);

Для проектирования тест-кейсов выделены следующие **группы тестовых сценариев:**

- основное позитивное тестирование,

- позитивное тестирование с необязательными параметрами по API документации,

- негативное тестирование с валидными входными данными,

- негативное тестирование с недопустимыми входными данными.

При тестировании API будут применяться **2 потока:**

- изолированное тестирование запроса (выполнение одного запроса API и проверка ответа),

- многоступенчатый поток с несколькими запросами (POST-GET-PATCH-DELETE).

На основании документации составлена таблица с возможными вызовами API и их действиями (таблица 6).

Таблица 6

|  |  |
| --- | --- |
| **Вызов API** | **Действия** |
| POST/ https://restful-booker.herokuapp.com/auth | Создает новый токен авторизации для доступа к изменению и удалению бронирования |
| GET/ https://restful-booker.herokuapp.com/booking | Показывает “id” всех бронирований. Возможно использование фильтров по имени/фамилии/дате заезда/дате выезда |
| GET/ https://restful-booker.herokuapp.com/booking/:id | Показывает конкретное бронирование на основании предоставленного «id» бронирования |
| POST/ https://restful-booker.herokuapp.com/booking | Создает новое бронирование |
| PUT/ https://restful-booker.herokuapp.com/booking/:id | Обновляет текущее бронирование |
| PATCH/ https://restful-booker.herokuapp.com/booking/:id | Частично обновляет бронирование |
| DELETE/ https://restful-booker.herokuapp.com/booking/1 | Удаляет бронирование |
| GET/ https://restful-booker.herokuapp.com/ping | Проверяет работоспособность API |

Ниже в таблице 7 представлены основные сценарии тестирования, на основании которых будут составлены тест-кейсы и проведено тестирование. Категории сценариев тестирования применяются к каждой функции: Аутентификация, Бронирование, PING.

Таблица 7

|  |  |  |
| --- | --- | --- |
| **Категория сценария тестирования** | **Категория тестового действия** | **Описание действия теста** |
| **1.Позитивные тесты с действительными параметрами** | | |
| Выполнение вызова API c действительными обязательными параметрами | Проверить код состояния | Все запросы должны возвращать код состояния HTTP 2XX согласно документации. |
| Подтвердить правильность загрузки | 1. В ответе правильно сформированный объект JSON.  2. Структура ответа соответствует модели данных (проверка схемы: имена и типы полей соответствуют ожидаемым, включая вложенные объекты; значения полей соответствуют ожидаемым). |
| Проверить состояние | Для операций POST, DELETE, PATCH, PUT убедиться, что действие в системе было выполнено правильно:  - выполнить соответствующий запрос GET и проверить ответ. |
| Проверить заголовки | 1. Проверить, что заголовки HTTP соответствуют ожидаемым согласно документации (обязательность, регистронезависимость и др.) |
| Проверить производительность | Ответ получен своевременно в разумные ожидаемые сроки, как определено в плане тестирования. |
| **2. Позитивные тесты с необязательными параметрами** | | |
| Выполнение вызова API с действительными обязательными параметрами и действительными необязательными параметрами.  Выполнить те же тесты, что и в пункте #1, но включая дополнительные параметры эндпоинта (например, фильтр). | Проверить код состояния | Все запросы должны возвращать код состояния HTTP 2XX согласно документации. |
| Проверить правильность загрузки | 1. Проверить структуру и содержание ответа, как в пункте #1.  2. Проверить параметры  - фильтрации: ответ фильтруется по указанному значению,  - проверить, что заголовки Cookie и Authorization могут использоваться как альтернатива друг другу для доступа к эндпоинту. |
| Проверка состояния | См. пункт #1 |
| Проверка заголовков | См. пункт #1 |
| Проверка производительности | См. пункт #1 |
| **3. Негативные тесты с валидными данными** | | |
| Выполнять вызовы API с допустимыми входными данными, которые пытаются выполнить незаконные операции, например:  - удалить бронирование, которое не существует,  - обновить бронирование недопустимыми действительными данными,  - попытаться совершить незаконную операцию (например, удаление бронирования без разрешения). | Проверить код состояния | 1. Проверить, что отправлен ошибочный код состояния HTTP (не 2xx), соответствующий ошибке.  2. Проверить, что код состояния HTTP соответствует типу ошибки, как определено в спецификации. |
| Проверить правильность загрузки | 1. Проверить, что получен ответ об ошибке  2. Проверить, что есть четкое и понятное поле сообщения об ошибке/описание ошибки.  3. Проверить, что описание ошибки правильное для данного типа ошибки и соответствует общей спецификации. |
| Проверка заголовков | См. пункт #1 |
| Проверка производительности | Проверить, что ошибка получена своевременно. |
| **4. Негативные тесты с невалидными данными** | | |
| Выполнение вызовов API с невалидными данными, например:  - токен авторизации отсутствует или недействителен,  - недействительный Cookie,  - отсутствуют необходимые параметры,  - недействительное значение для параметров эндпоинта (загрузка с недопустимой моделью (нарушает схему), загрузка с неполной моделью (отсутствующие поля или обязательные вложенные сущности), недействительные значения во вложенных полях сущности, неподдерживаемые методы для эндпоинтов).  - некорректным контентом в запросе,  - неверным типом содержимого в загрузке,  - контентом с неправильной структурой,  - переполнением значений параметров, например:  \* попытка создать бронирование с id заголовком более 200 символов,  \*попытка GET запроса с указанием id бронирования длиной 1000 символов  \*переполнение тела ответа  \*огромный JSON в теле запроса.  - проверка граничных значений,  - пустые подобъекты в загрузочных данных,  - недопустимые символы в параметрах или загрузочных данных,  - использование неправильных заголовков HTTP (например, Content-Type) | Проверить коды состояния | См. пункт #3 |
| Проверка загрузки | См. пункт #3 |
| Проверка заголовков | См. пункт #1 |
| Проверка производительности | См. пункт #3 |

При проведении нагрузочного тестирования будем использовать следующие данные:

- количество пользователей: 100/800

- время наращивания запросов: 60с

- количество запросов: 20

Метрики, по которым будем оценивать работоспособность – среднее время отклика, количество запросов.

При тестировании API могут быть выявлены дефекты в работе веб-сервиса. Они должны быть задокументированы в баг-трекинговой системе.

Каждый баг-репорт содержит следующую информацию о дефекте:

- краткое описание, представляющее собой описание проблемы,

- местонахождение дефекта,

- шаги для воспроизведения ошибки,

- ожидаемый и фактический результат,

- может содержать дополнительную информацию о дефекте в виде прикрепленных скриншотов.

Всем дефектам в обязательном порядке присваивается статус приоритета (очередности исправления дефекта) и серьезности (влияния дефекта на работоспособность веб-сервиса).

Степени приоритета и серьезности дефектов представлены в таблице 8.

Таблица 8

|  |  |  |
| --- | --- | --- |
| **№** | **Статус** | **Описание** |
| **Приоритет (Priority)** | | |
| 1. | Высокий (High) | Исправление дефекта в первую очередь |
| 2. | Средний (Medium) | Исправление дефекта после того, как исправили все дефекты с приоритетом «Высокий» |
| 3. | Низкий (Low) | Исправление дефекта после того, как были исправлены все более приоритетные дефекты |
| **Серьезность (Severity)** | | |
| 1. | Блокирующий (Blocker) | Блокирует дальнейшую работу с веб-сервисом |
| 2. | Критичный (Critical) | Нарушает работу основного функционала, делает невозможным использование основного функционала |
| 3. | Важный (Major) | Затрудняет работу основного функционала, а использование дополнительных функций становится невозможным |
| 4. | Не особо важный (Minor) | Относительно мало влияет на функционал системы |
| 5. | Незначительный (Trivial) | Не влияет на функционал системы |

Тестирование может быть частично либо полностью приостановлено, если произойдут следующие события:

- возникнут проблемы работы сервера, БД;

- возникнет ошибка функции, а документация к API не содержит никаких изменений;

- возникнет серьезная проблема, препятствующая продолжению тестирования.

**9. РАСПИСАНИЕ (TEST SCHEDULE)**

|  |  |  |  |
| --- | --- | --- | --- |
| **Задача** | **Продолжительность** | **Начало/Завершение** | **Ответственный** |
| Анализ документации к API | 1 день | 21.01.2023/21.01.2023 | Божкова И.В. |
| Написание Тест-Плана | 2,5 дня | 22.01.2023/24.01.2023 | Божкова И.В. |
| Составление тест-кейсов | 3,5 дня | 24.01.2023/27.01.2023 | Божкова И.В. |
| Составление коллекции тестов в Postman | 2 дня | 28.01.2023/29.01.2023 | Божкова И.В. |
| Проведение тестирования и составление баг-репортов | 4 дня | 06.01.2023/09.02.2023 | Божкова И.В. |
| Написание отчета о результатах тестирования | 1 день | 10.02.2023/10.02.2023 | Божкова И.В. |

Итого планируемое время на подготовку, проведение тестирования и оформления отчета о результатах – 15 дней.