***Тест-кейсы для запроса GET /posts?userId=<id>&title=<title>***

В первом задании созданы автотесты, которые выполняют запрос GET с некоторыми условиями и проверяют статус ответа на запрос, заполненность боди и количество записей в ответе. Для написания тест-кейсов решено описать тесты, используя разное окружение.

Запросы GET можно исполнить, используя платформу Postman. Так как запрос GET не имеет body, интересующие нас параметры заводим во вкладке Params.

|  |
| --- |
| **TEST-1. Убедиться, что при запросе GET возвращается пост с существующими в системе userId и title.**  ***Description:*** Убедиться, что при запросе GET с использованием параметров userId и title в ответ возвращается пост, содержащий в себе значения этих параметров.  ***Precondition:*** для теста берется пост из https://jsonplaceholder.typicode.com/posts со значением id=2. Тест проводится на платформе Postman.  ***Priority:*** critical  ***Steps:***   1. Выполните запрос GET/posts с query params:   userId:1  title:qui est esse  ***Expected result:***   1. В ответ пришло 200ОК с телом. 2. [ 3. { 4. "userId": 1, 5. "id": 2, 6. "title": "qui est esse", 7. "body": "est rerum tempore vitae\nsequi sint nihil reprehenderit dolor beatae ea dolores neque\nfugiat blanditiis voluptate porro vel nihil molestiae ut reiciendis\nqui aperiam non debitis possimus qui neque nisi nulla" 8. } 9. ] |
| **TEST-2.** **Убедиться, что при запросе GET с не существующим в системе пользователем и с существующим в системе заголовком возвращается пустой список.**  ***Description:*** Убедиться, что при запросе GET с использованием query параметров userId и title в ответ не возвращается пост, так как запрашиваемого значения userId нет в базе данных.  ***Precondition:*** Значение параметра title берется из поста с id=2, максимальное значение userId на момент тестирования равен 10, поэтому для теста берем значение 11. Тест проводится на платформе Postman.  ***Priority:*** critical  ***Steps:***   1. Выполните запрос GET/posts с query params:   userId:11  title:qui est esse  ***Expected result:***   1. В ответ пришло 200ОК с телом.   [] |
| **TEST-3.** **Убедиться, что при запросе GET с существующим в системе пользователем и не существующим в системе заголовком возвращается пустой список.**  ***Description:*** Убедиться, что при запросе GET с использованием query параметров userId и title в ответ не возвращается пост, так как запрашиваемого значения title нет в базе данных.  ***Precondition:*** Значение параметра title не должно быть в баз данных, значение параметра userId должно быть в базе данных. для теста берется пост из https://jsonplaceholder.typicode.com/posts со значением id=2. Тест проводится на платформе Postman.  ***Priority:*** critical  ***Steps:***   1. Выполните запрос GET/posts с query params:   userId:1  title: PRIMER  ***Expected result:***   1. В ответ пришло 200ОК с телом.   [] |
| **TEST-4.** **Убедиться, что при запросе GET с не существующими в системе пользователем и заголовком возвращается пустой список.**  ***Description:*** Убедиться, что при запросе GET с использованием query параметров userId и title в ответ не возвращается пост, так как запрашиваемого значения userId и title нет в базе данных.  ***Precondition:*** Значения параметров userId и title не должно быть в базе данных на момент тестирования. Тест проводится на платформе Postman.  ***Priority:*** critical  ***Steps:***   1. Выполните запрос GET/posts с query params:   userId:11  title: PRIMER  ***Expected result:***   1. В ответ пришло 200ОК с телом.   [] |
| **TEST-5.** **Убедиться, что нельзя запросить пост, используя невалидный userId.**  ***Description:*** Убедиться, что при запросе GET с использованием query параметров userId и title в ответ не возвращается пост, так как тип данных userId – string, а не number.  ***Precondition:***. Значение title есть в базе данных на момент тестирования. для теста берется пост из https://jsonplaceholder.typicode.com/posts со значением id=2. Тест проводится на платформе Postman.  ***Priority:*** medium  ***Steps:***   1. Выполните запрос GET/posts с query params:   userId:Katya  title:qui est esse  ***Expected result:***   1. В ответ пришло 200ОК с телом.   [] |

Для тестирования через командную строку cmd Windows применяю кроссплатформенную служебную программу командной строки cURL.

|  |
| --- |
| **TEST-6.** **Убедиться, что нельзя запросить пост с userId, превышающим формат числа.**  ***Description:*** Убедиться, что при запросе GET с использованием query параметров userId и title в ответ не возвращается пост, так как запрашиваемое значение userId превышает формат числа.  ***Precondition:*** Значение параметра userId имеет значение строго больше, чем 9007199254740991. Значение title есть в базе данных на момент тестирования. для теста берется пост из https://jsonplaceholder.typicode.com/posts со значением id=7.  Тестирование проводится при помощи cmd Windows.  ***Priority:*** medium  ***Steps:***   1. Ввести в командную строку значение: curl "https://jsonplaceholder.typicode.com/posts?userId=9007199254740992&title=magnam+facilis+autem"   ***Expected result:***   1. В командную строку вернулся пустой список.   [] |
| **TEST-7.** **Убедиться, что нельзя запросить пост с userId – пустым полем.**  ***Description:*** Убедиться, что при запросе GET с использованием query параметров userId и title в ответ не возвращается пост, так как запрашиваемое значение userId не заполнено, является пустым полем без символов.  ***Precondition:*** Значение title есть в базе данных на момент тестирования. Для теста берется пост из https://jsonplaceholder.typicode.com/posts с значением id=7. Тестирование проводится при помощи cmd Windows.  ***Priority:*** meduim  ***Steps:***  1. Ввести в командную строку значение:  curl "https://jsonplaceholder.typicode.com/posts?userId&title=magnam+facilis+autem"  ***Expected result:***   1. В командную строку вернулся пустой список.   [] |
| **TEST-8.** **Убедиться, что на запрос GET c существующим userId и частью текста от существующего title возвращаются posts с данными query параметров.**  ***Description:*** Убедиться, что при запросе GET с использованием query параметров userId и части значения параметра title в ответ возвращаются все посты, отвечающие заданным параметрам.  ***Precondition:*** Значение userId есть в базе данных на момент тестирования.  Значение title есть в базе данных на момент тестирования, для запроса из значения title выбираем одно слово, в нашем примере слово qui.  Тестирование проводится при помощи cmd Windows.  ***Priority:*** high   1. ***Steps:*** Ввести в командную строку значение:   curl https://jsonplaceholder.typicode.com/posts?userId=1&title\_like=qui  ***Expected result:***   1. В командную строку вернулся список с постами, имеющими id [2, 3, 6, 10] |
| **TEST-9.** **Убедиться, что нельзя запросить пост с существующим userId и добавлением текста к существующему title.**  ***Description:*** Убедиться, что при запросе GET с использованием query параметров userId и title в ответ не возвращается пост, так как к значению title добавлен дополнительный текст.  ***Precondition:*** Значение userId есть в базе данных на момент тестирования.  Значение title есть в базе данных на момент тестирования, для запроса к значению title добавляем слово primer в конец заголовка.  для теста берется пост из https://jsonplaceholder.typicode.com/posts с значением id=7.  Тестирование проводится при помощи cmd Windows.  ***Priority:*** high  ***Steps:***  1. Ввести в командную строку значение:  curl "https://jsonplaceholder.typicode.com/posts?userId=1&title=magnam+facilis+autem+primer"  ***Expected result:***   1. В командную строку вернулся пустой список.   [] |
| **TEST-10.** **Убедиться, что нельзя запросить пост с существующим userId и title с измененным порядком слов.**  ***Description:*** Убедиться, что при запросе GET с использованием query параметров userId и title в ответ не возвращается пост, так как в запросе присутствует значение title из базы данных с измененным порядком слов в нем.  ***Precondition:*** Значение userId есть в базе данных на момент тестирования.  Значение title есть в базе данных на момент тестирования, для запроса последнее слово из строки title ставим в начало строки title, например «My name is Kate» преобразуем в «Kate My name is»  для теста берется пост из https://jsonplaceholder.typicode.com/posts с значением id=7.  Тестирование проводится при помощи cmd Windows.  ***Priority:*** high  ***Steps:***   1. Ввести в командную строку значение:   curl "https://jsonplaceholder.typicode.com/posts?userId=1&title=autem+magnam+facilis"  ***Expected result:***   1. В командную строку вернулся пустой список.   [] |

Запросы GET можно выполнять через строку запроса в браузере.

Для того, чтобы получить данные по query параметрам «nameId» и «title», необходимо дополнить строку запроса с полным списком постов (100 штук) следующим выражением запроса: ?userId=<id>&title=<title>.

|  |
| --- |
| **TEST-11.** **Убедиться, что нельзя запросить пост с существующими userId и title, в котором присутствуют двойные пробелы между словами.**  ***Description:*** Убедиться, что при запросе GET с использованием query параметров userId и title в ответ не возвращается пост, так как в запросе присутствует значение title из базы данных с двойными пробелами между словами.  ***Precondition:*** Значение userId есть в базе данных на момент тестирования.  Значение title есть в базе данных на момент тестирования, для запроса необходимо изменить title: добавить дополнительный пробел между словами в titile.  для теста берется пост из https://jsonplaceholder.typicode.com/posts с значением id=2.  Тестирование проводится при помощи cmd Windows.  ***Priority:*** high  ***Steps:***   1. Открыть страницу в браузере Google Chrome https://jsonplaceholder.typicode.com/posts и добавить в строку запроса query параметры: posts?userId=1&title=qui%20%20est%20%20esse   ***Expected result:***   1. В командную строку вернулся пустой список.   [] |
| **TEST-12.** **Убедиться, что при запросе поста по критериям id пользователя и заголовок, в ответ возвращается пост с элементами: userid, id, title, body.**  ***Description:*** Убедиться, что при запросе GET с использованием query параметров userId и title в ответ возвращается пост с элементами: userid, id, title, body.  ***Precondition:*** Значение userId есть в базе данных на момент тестирования.  Значение title есть в базе данных на момент тестирования. для теста берется пост из https://jsonplaceholder.typicode.com/posts со значением id=2.  ***Priority:*** critical  ***Steps:***   1. Открыть страницу в браузере Google Chrome https://jsonplaceholder.typicode.com/posts и добавить в строку запроса query параметры: posts?userId=1&title=qui%20est%20esse   ***Expected result:***   1. На страницу браузера вернулся список posts, содержащий в себе оба значения query параметров. А также поля id и body. 2. [ 3. { 4. "userId": 1, 5. "id": 2, 6. "title": "qui est esse", 7. "body": "est rerum tempore vitae\nsequi sint nihil reprehenderit dolor beatae ea dolores neque\nfugiat blanditiis voluptate porro vel nihil molestiae ut reiciendis\nqui aperiam non debitis possimus qui neque nisi nulla" 8. } 9. ] |

Создадим несколько тест-кейсов для написания автотестов. Окружение для автотеста: операционная система Windows, среда разработки WebStorm, программа cypress.io, браузер Google Chrome.

|  |
| --- |
| **TEST-13.** **Убедиться, что при запросе постов с существующим userId и двумя существующими titile, в ответ возвращается два поста.**  ***Description:*** Убедиться, что при запросе GET с использованием query параметров userId и title (две записи) в ответ возвращается два поста, отвечающие запрашиваемым параметрам.  ***Precondition:*** Значения title есть в базе данных на момент тестирования и относятся к одному userId.  cypress.io установлен через среду разработки WebStorm, используя инструкцию https://docs.cypress.io/guides/getting-started/installing-cypress.  ***Priority:*** hight  ***Steps:***   1. В среде разработки в созданный файл js вставить код :   ***describe***("API\_test for two title",()=> { ***it***("GET two posts of one userId", ()=>{  ***cy***.request('GET','https://jsonplaceholder.typicode.com/posts?userId=1&title=qui est esse&title=nesciunt quas odio').then((response) =>{  ***expect***(response).to.have.property('status',200)  ***expect***(response.body).to.not.be.null  ***expect***(response.body).to.have.length(2)  })})  })   1. Через терминал среды разработки запустить cypress.io при помощи команды cypress open. 2. В появившемся окне cypress.io выбрать файл js, в который записан вышеупомянутый код и запустить его. 3. Дождаться открытия нового окна в браузере.   ***Expected result:***   1. Создан файл js с кодом для автотеста. 2. Появилось диалоговое окно со списком файлов js. 3. Открылось окно в браузере, которое запустило автотест. 4. В появившемся окне, слева, в списке около «GET two posts of one userId» появилась зеленая галочка, свидетельствующая о том, что test passed. |
| **TEST-14. Убедиться, что нельзя запросить пост с userId, состоящим из пробела.**  ***Description:*** Убедиться, что при запросе GET с использованием query параметров userId и title в ответ не возвращается пост, так как запрашиваемое значение userId заполнено только символами пробела  ***Precondition:*** Значение title есть в базе данных на момент тестирования.  cypress.io установлен через среду разработки WebStorm, используя инструкцию https://docs.cypress.io/guides/getting-started/installing-cypress.  ***Priority:*** high  ***Steps:***   1. В среде разработки в созданный файл js вставить код :   ***describe***("API\_test for userId with space",()=> { ***it***("GET empty list - the reason userId with only one space", ()=>{  ***cy***.request('GET','https://jsonplaceholder.typicode.com/posts?userId= &title=qui est esse').then((response) =>{  ***expect***(response).to.have.property('status',200)  ***expect***(response.body).to.not.be.null  ***expect***(response.body).to.have.length(0)  })})  })   1. Через терминал среды разработки запустить cypress.io при помощи команды cypress open. 2. В появившемся окне cypress.io выбрать файл js, в который записан вышеупомянутый код и запустить его. 3. Дождаться открытия нового окна в браузере.   ***Expected result:***   1. Создан файл js с кодом для автотеста. 2. Появилось диалоговое окно со списком файлов js. 3. Открылось окно в браузере, которое запустило автотест. 4. В появившемся окне, слева, в списке около «GET empty list - the reason userId with only one space» появилась зеленая галочка, свидетельствующая о том, что test passed. |