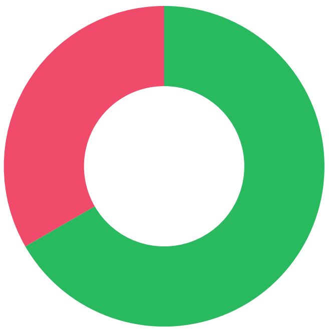
Test for creating a resource feature

Test for creating a resource feature

Completion chart	Completion stats	Completion rate
	● Untested (0) ● Passed (2) ● Failed (1) ● Blocked (0) ● Retest (0) ● Skipped (0) ● Invalid (0)	100%

Started by	Start time	Estimated	Time spent
Ni Wayan Amanda Putri Astawa	2024-03-24 06:11:56	00:00:00	00:02:32

Environment	Milestone
Development	Release 1.0.0

Create a new post with valid data

Status	Time spent	Assignee
Passed	00:00:02	Ni Wayan Amanda Putri Astawa

Results

Result 1

Status	Time spent	User	Defects
Passed	00:00:02	Ni Wayan Amanda Putri Astawa	-

Finish time

2024-03-24 06:12:03

Steps

Step	1
Action	Click "Add request" in the Collection that has been created
Expected result	One new request added
Status	Untested

Step	2
Action	Add a POST request and enter the BASE URL
Input data	https://jsonplaceholder.typicode.com/posts
Expected result	Method becomes POST and base URL is added
Status	Untested

Step	3
Action	Enter JSON data in the "body" field
Input data	{ "title": "foo", "body": "bar", "userId": 1 }
Expected result	Request body added
Status	Untested

Step	4
Action	Click the save button

Expected result	Changes are saved
Status	Untested

Step	5
Action	Click the send button to send the POST request to the endpoint
Expected result	JSON response displays new post data
Status	Untested

Step	6
Action	Input the testing script in the "test" field

Input data

```
pm.test("Response status code is 201", function () {
  pm.expect(pm.response.code).to.equal(201);
});




pm.test("Response has the required fields - title, body, userId, and id", function () {
  const responseData = pm.response.json();

  pm.expect(responseData).to.be.an('object');
  pm.expect(responseData).to.have.property('title');
  pm.expect(responseData).to.have.property('body');
  pm.expect(responseData).to.have.property('userId');
  pm.expect(responseData).to.have.property('id');
});




pm.test("Title and body are non-empty strings", function () {
  const responseData = pm.response.json();

  pm.expect(responseData).to.be.an('object');
  pm.expect(responseData.title).to.be.a('string').and.to.have.lengthOf.at.least(1, "Title should not be
empty");
  pm.expect(responseData.body).to.be.a('string').and.to.have.lengthOf.at.least(1, "Body should not be
empty");
});




pm.test("UserId and id should be non-negative integers", function () {
  const responseData = pm.response.json();

  pm.expect(responseData.userId).to.be.a('number').and.to.be.at.least(0);
  pm.expect(responseData.id).to.be.a('number').and.to.be.at.least(0);
});




pm.test("Content-Type is application/json", function () {
    pm.expect(pm.response.headers.get("Content-Type")).to.include("application/json");
});
```

Expected result	Script for testing added
Status	Untested

Step	7
Action	Click the save button
Expected result	Changes are saved
Status	Untested

Step	8
Action	Click the send button to view the test results and receive a response from the server
Expected result	Testing results are displayed
Status	Untested

RA-4 / Creating a resource

Create a new post without inserting JSON data

Status	Time spent	Assignee
Failed	00:02:10	Ni Wayan Amanda Putri Astawa

Results

Result 1

Status	Time spent	User	Defects
Failed	00:02:10	Ni Wayan Amanda Putri Astawa	D-1 (Open)

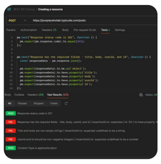
Finish time

2024-03-24 06:16:52

Comment

JSON response displays 201 (Created)

Attachments



Create_a_new_po...

Steps

Step	1
Action	Click "Add request" in the Collection that has been created
Expected result	One new request added
Status	Passed

Step	2
Action	Add a POST request and enter the BASE URL
Input data	https://jsonplaceholder.typicode.com/posts
Expected result	Method becomes POST and base URL is added
Status	Passed

Step	3
Action	Click the save button
Expected result	Changes are saved
Status	Passed

Step	4
Action	Click the send button to send the POST request to the endpoint
Expected result	JSON response displays new post data
Status	Failed

Step	5
Action	Enter the testing script in the "test" field

Input data

```
pm.test("Response status code is 201", function () {
  pm.expect(pm.response.code).to.equal(201);
});




pm.test("Response has the required fields - title, body, userId, and id", function () {
  const responseData = pm.response.json();

  pm.expect(responseData).to.be.an('object');
  pm.expect(responseData).to.have.property('title');
  pm.expect(responseData).to.have.property('body');
  pm.expect(responseData).to.have.property('userId');
  pm.expect(responseData).to.have.property('id');
});




pm.test("Title and body are non-empty strings", function () {
  const responseData = pm.response.json();

  pm.expect(responseData).to.be.an('object');
  pm.expect(responseData.title).to.be.a('string').and.to.have.lengthOf.at.least(1, "Title should not be empty");
  pm.expect(responseData.body).to.be.a('string').and.to.have.lengthOf.at.least(1, "Body should not be empty");
});




pm.test("UserId and id should be non-negative integers", function () {
  const responseData = pm.response.json();

  pm.expect(responseData.userId).to.be.a('number').and.to.be.at.least(0);
  pm.expect(responseData.id).to.be.a('number').and.to.be.at.least(0);
});




pm.test("Content-Type is application/json", function () {
    pm.expect(pm.response.headers.get("Content-Type")).to.include("application/json");
});
```

Expected result	Script for testing added
Status	Untested

Step	6

Action	Click the save button
Expected result	Changes are saved
Status	Untested

Step	7
Action	Click the send button to view the test results and receive a response from the server
Expected result	Testing results are displayed
Status	Untested

Create a new post with incorrect data format

Status	Time spent	Assignee
Passed	00:00:19	Ni Wayan Amanda Putri Astawa

Results

Result 1

Status	Time spent	User	Defects
Passed	00:00:19	Ni Wayan Amanda Putri Astawa	-

Finish time

2024-03-24 06:17:58

Steps

Step	1
Action	Click "Add request" in the Collection that has been created
Expected result	One new request added
Status	Untested

Step	2
Action	Add a POST request and enter the BASE URL
Input data	https://jsonplaceholder.typicode.com/posts
Expected result	Method becomes POST and base URL is added
Status	Untested

Step	3
Action	Enter the wrong JSON data in the "body" field
Input data	{ title: 'foo', body: 'bar', userId: 1, }
Expected result	Request body added
Status	Untested

Step	4
Action	Click the save button

Expected result	Changes are saved
Status	Untested

Step	5
Action	Click the send button to send the POST request to the endpoint
Expected result	JSON response displays
Status	Untested

Step	6
Action	Enter the testing script in the "test" field

Input data

```
pm.test("Response status code is 201", function () {
  pm.expect(pm.response.code).to.equal(201);
});




pm.test("Response has the required fields - title, body, userId, and id", function () {
  const responseData = pm.response.json();

  pm.expect(responseData).to.be.an('object');
  pm.expect(responseData).to.have.property('title');
  pm.expect(responseData).to.have.property('body');
  pm.expect(responseData).to.have.property('userId');
  pm.expect(responseData).to.have.property('id');
});




pm.test("Title and body are non-empty strings", function () {
  const responseData = pm.response.json();

  pm.expect(responseData).to.be.an('object');
  pm.expect(responseData.title).to.be.a('string').and.to.have.lengthOf.at.least(1, "Title should not be
empty");
  pm.expect(responseData.body).to.be.a('string').and.to.have.lengthOf.at.least(1, "Body should not be
empty");
});




pm.test("UserId and id should be non-negative integers", function () {
  const responseData = pm.response.json();

  pm.expect(responseData.userId).to.be.a('number').and.to.be.at.least(0);
  pm.expect(responseData.id).to.be.a('number').and.to.be.at.least(0);
});




pm.test("Content-Type is application/json", function () {
    pm.expect(pm.response.headers.get("Content-Type")).to.include("application/json");
});
```

Expected result	Script for testing added
Status	Untested

Step	7
Action	Click the save button
Expected result	Changes are saved
Status	Untested

Step	8
Action	Click the send button to view the test results and receive a response from the server
Expected result	Testing results are displayed
Status	Untested