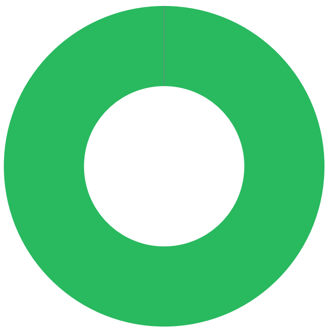
Test for getting a resource

Test for getting a resource

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

Started by	Start time	Estimated	Time spent
Ni Wayan Amanda Putri Astawa	2024-03-24 06:22:59	00:00:00	00:00:06

Environment	Milestone
Development	Release 1.0.0

Getting a specific post using a valid ID

Status	Time spent	Assignee
Passed	00:00:01	Ni Wayan Amanda Putri Astawa

Results

Result 1

Status	Time spent	User	Defects
Passed	00:00:01	Ni Wayan Amanda Putri Astawa	-

Finish time

2024-03-24 06:23:11

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 GET request and enter the base URL
Input data	https://jsonplaceholder.typicode.com/posts/1
Expected result	Method becomes GET and base URL is added
Status	Untested

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

Step	4
Action	Click the send button to send the GET request to the endpoint
Expected result	JSON response displays post data based on ID
Status	Untested

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

Input data	pm.test("Response status code is 404", function () {
	    pm.expect(pm.response.code).to.equal(404);
	});
	
	pm.test("Response has the required Content-Type header", function () {
	    pm.expect(pm.response.headers.get("Content-Type")).to.include("application/json");
	});
	
	pm.test("Response body is an empty JSON object", function () {
	    const responseData = pm.response.json();
	
	    pm.expect(responseData).to.be.an('object').and.to.be.empty;
	});
	
	pm.test("UserId field should not be present in the response", function () {
	    const responseData = pm.response.json();
	
	    pm.expect(responseData).to.not.have.property('userId');
	});
	
	pm.test("Id field is a non-negative integer", function () {
	    const responseData = pm.response.json();
	
	    pm.expect(responseData.id).to.exist.and.to.be.a('number').and.to.satisfy((id) => id >= 0, "Id must be a non-negative integer");
	});

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

Getting a non-existing post

Status	Time spent	Assignee
Passed	00:00:04	Ni Wayan Amanda Putri Astawa

Results

Result 1

Status	Time spent	User	Defects
Passed	00:00:04	Ni Wayan Amanda Putri Astawa	-

Finish time

2024-03-24 06:23:17

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 GET request and enter the base URL with an ID that does not exist. (Example: ID = 999)
Input data	https://jsonplaceholder.typicode.com/posts/999
Expected result	Method becomes GET and base URL is added
Status	Untested

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

Step	4
Action	Click the send button to send the GET request to the endpoint
Expected result	JSON response displays 404 (Not Found)
Status	Untested

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

Input data	pm.test("Response status code is 404", function () {
	    pm.expect(pm.response.code).to.equal(404);
	});
	
	pm.test("Response has the required Content-Type header", function () {
	    pm.expect(pm.response.headers.get("Content-Type")).to.include("application/json");
	});
	
	pm.test("Response body is an empty JSON object", function () {
	    const responseData = pm.response.json();
	
	    pm.expect(responseData).to.be.an('object').and.to.be.empty;
	});
	
	pm.test("UserId field should not be present in the response", function () {
	    const responseData = pm.response.json();
	
	    pm.expect(responseData).to.not.have.property('userId');
	});
	
	pm.test("Id field is a non-negative integer", function () {
	    const responseData = pm.response.json();
	
	    pm.expect(responseData.id).to.exist.and.to.be.a('number').and.to.satisfy((id) => id >= 0, "Id must be a non-negative integer");
	});

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