GitHub CRUD Operations API Testing

Prepared by: Mayank Rusia Date: (15/10/2024)

Table of Contents

1. Project Overview/Introduction
2. Test Plan
3. Test Cases Document
4. API Documentation
5. Challenges & Learnings
6. Test Summary Report
7. Conclusion

1. Project Overview/Introduction

Project Title:

GitHub CRUD Operations API Testing

Objective:

The objective of this project is to test and validate GitHub's REST API by performing CRUD operations on repositories. The focus is on using Postman to execute and validate four core actions: Create, Read, Update, and Delete. The project leverages Bearer Token authentication for secure access to the GitHub API and ensures the API responses meet the expected behavior in real-world scenarios.

Used Elements:

- Postman: For sending HTTP request & validating API responses.
- Bearer Token: For Authentication & secure API access.
- GitHub REST API: The API being tested.

Technologies:

- API Testing: POST, PATCH, GET, DELETE methods.
- Bearer Token: Handling access token-based authentication.

2. Test Plan

Scope:

This project covers testing the following CRUD operations provided by the GitHub API:
1. Create new Repository (POST)
2. Retrieve Repositories details (GET)
3. Update existing Repository (PATCH)
4. Delete existing Repository (DELETE)

Test Strategy:

Manual Testing: 'Postman' tool is used to manually execute API requests for each CRUD operation and wrote Post-response test scripts to check responses getting after performing CRUD operation.

Test Environment:

• Postman: For manual requests.
• Authentication: Bearer token for GitHub API access
• GitHub Account: The project uses GitHub account

Tools:

• Postman
• Bearer Token (For securing API access)

3. Test Cases Document

4. API Documentation

Provide detailed descriptions of each API endpoint, request body, and expected response format.

API Document: "https://docs.github.com/en/rest/repos/repos?apiVersion=2022-11-28"

API Operation	Method	Endpoint	Request body	Response
Create Repo	POST	/user/repos	{ "name": "new-repo", "description": "Test repo", "private": false }	201 Created, JSON object with repo details

Update Repo	PATCH	/repos/{owner}/{repo}	{ "name": "updated-repo", "description": "Updated description", “private”: false }	200 OK, JSON object with updated repo details
Get Repos	GET	/user/repos	NA	200 OK, Array of repository objects
Delete Repo	DELETE	/repos/{owner}/{repo}	NA	204 No Content

5. Challenges & Learnings

Challenges:

1. **Bearer Token Management**: Managing the Bearer token for authentication was crucial. Ensuring the token was valid and properly included in each request required careful handling. Occasionally, the token would expire or become invalid, leading to failed requests.

2. **Error Handling for Negative Test Cases**: When performing negative test cases (e.g., trying to create a repository with an existing name or invalid input), capturing and validating the correct error messages was essential. This required clear understanding of GitHub's API error response formats.

3. **Positive and Negative Test Case Execution**: Ensuring that both positive and negative test cases were well-structured and executed correctly required careful planning. This included creating valid input for positive tests and deliberately invalid inputs for negative tests.

Learnings:

• Understood API testing techniques using Postman.

• Gained experience in handling real-world scenarios such as error responses and reason behind them & how to handle them.

• **Effective Use of Bearer Tokens**: Gained practical experience in using Bearer tokens for API authentication, including how to securely manage and include them in API requests.

• **CRUD Operations Proficiency**: Developed a solid understanding of how to perform simple CRUD operations with the GitHub API, including the structure of requests and handling responses.

• **Validation of API Responses**: Improved skills in validating both successful and error responses from the API, ensuring that the application behaves as expected under various conditions.

• **Test Case Design**: Learned the importance of structuring test cases to cover a range of scenarios, including edge cases, to ensure comprehensive API testing.

• **Monitoring API Behavior**: Became proficient in monitoring and interpreting API responses, including understanding HTTP status codes and their implications for both successful and failed requests.

• **Error Handling Techniques**: Developed techniques for effectively handling and validating errors in API responses, enhancing the robustness of the testing process.

6. Test Summary Report

Test Summary	**Value**
Total Test Cases	35
Passed	29
Failed	6
Response Times	Recorded response times for each CRUD operation.

7. Conclusion

This GitHub CRUD Operation API Testing Project has significantly enhanced my skills in API testing. By utilizing Postman, successfully performed basic CRUD operations – Create Repo, Fetch Repo details, update repo details, Delete repo. While effectively managing Bearer Token for authentication.

Through this project, I gained hands-on experience in crafting API requests, validating responses, handling both positive & negative test cases. I have also performed other operations on this project, running through Newman (CLI mode), Generated HTML reports, which help me to generate test reports.

Overall, this project deepened my understanding of API testing practices & importance of reliable web services, positioning me well for future roles in software testing.