**Testing the CRUD Operations and performance of List of Users**

DESCRIPTION :

Given an API which has details about Users of an Organization. Test CRUD operations of this API using Postman and automate using Rest Assured and validate the response. Check the performance of an API using Jmeter.

In this document – Postman operations are elaborated. RestAssured and Jmeter steps are described in other separate document.

REQUISITES :

●Java 1.8

●Url(https://reqres.in/api/users?page=2)

●Postman

REQUIREMENTS TO BE MET :

* Send a **GET** request using Postman to API and check list of users
* Validate the Status codes generated for the above requests
* Sending a **POST** request using Postman to API to add user and check the response.

GET – Request:

1. Get single user details using fixed user id - GetSingleUser\_UserID\_2 (Get Request)

Response: The API returns a JSON object representing the user information. The response includes the following fields:

* id (integer): The unique identifier of the user.
* First\_name (string): The name of the user.
* Last\_name(string): The last name of the user.
* email (string): The email address of the user.
* avatar: The image link of the user.
* Support.url (string): The url of the support is mentioned.
* Support.text (string): The text description of the support

1. Get single user details using variable id in place of User id - {{URL}}/api/users/{{C\_ID}}
2. Get single user using snippet - <https://reqres.in/api/users/2>

These test scripts are used to validate various aspects of the API response.

1. Test: "Status code is 200"
   * Validates that the response status code is 200 (OK).
2. Test: "Response time is less than 200ms"
   * Verifies that the response time is below 200 milliseconds.
3. Test: "Your test name"
   * Extracts the response body as JSON data and checks if the value of the "first\_name" property is 'Janet'.
   * Logs the value of the "first\_name" property to the console.
4. Test: "Status code name has string"
   * Verifies that the response status code is "OK".
5. Test: "Schema is valid"
   * Defines a schema that describes the expected structure of the response.
   * Validates two data arrays, "data1" and "data2," against the defined schema using TV4 library.
   * Expects the validation to be true for both data arrays.
6. Test: "Content-Type is present"
   * Ensures that the "Content-Type" header of the response contains "application/json".

These test scripts are executed within Postman after making an API request, allowing you to automate the validation of the response data. The tests help ensure that the API is functioning as expected and that the response adheres to the specified criteria.

Top of Form

Bottom of Form

1. Get user details using CSV file input - GetSingleUser\_Userid\_CSV – CSV file name (userid\_postman.csv)

Description: This test script is used to validate the "id" field in the response csv data against the "user\_id" value provided in the iteration data. It helps ensure that the returned user ID matches the expected user ID.

Run collection option is used and specific request is selected for run and iterations and file path are mentioned and the collection is run

1. Get user details using Json file input - GetSingleUSer\_Userid\_Json – Json file name(userid\_jsontype.json)

Description: This test script is used to validate the "id" field in the response JSON data against the "user\_id" value provided in the iteration data. It helps ensure that the returned user ID matches the expected user ID.

1. Get list of users with page number specified - ListUsers\_pagenumber\_1
2. Get lsit of users with page number specified as variable - ListUsers\_pagenumber\_variable – using this url for get request - {{URL}}/api/users?page={{Pg\_No}}
3. Get list of users using page number field is received from csv file - ListUsers\_pagenumber\_CSV (Pg\_No\_postman.csv)
4. Get list of users using page number field is received from json file - ListUsers\_pagenumber\_Json – (Pg\_No\_jsontype.json

POST Request:

1. Post – create single user – Name – CreateUser - <https://reqres.in/api/users>

Json body is provided and test script is written to check status code.

1. Post – create user using inputs from CSV file - CreateUser\_CSV - <https://reqres.in/api/users> - CSV file name (postusers\_csvdata.csv
2. Post – create user using inputs from Json file - CreateUSer\_Json- <https://reqres.in/api/users> - Json file name (postusers\_jsontype.js

The collection is created as json file name - Phase4\_reqres.postman\_collection.json

And also uploaded in git hub