Wayker Arenas

Tech Test QA Automation Engineer

Mind the test cases are being delivered in a word file because the instructions requested, yet, a far better format to do this would be an excel (if working with Microsoft office is mandatory and no other tool to handle test cases is suitable)

# PostMan Test Suite

General assumptions:

1. The “latest” year is 2021
2. The “requirement” is to validate USA national data from the source “Census Bureau”

General preconditions

1. The API is reachable.
2. There is internet connection prior to executing the tests.

|  |  |  |  |
| --- | --- | --- | --- |
| Method: GetPopulation | | | |
| https://datausa.io/api/data?drilldowns=Nation&measures=Population | | | |
| Evaluating API responses | | | |
| 1 | Test case summary | Schema Validation | Validate the proper data schema |
| Steps | Validate objects | |
| Validate properties | |
| Validate required objects | |
| Expected Result | Response has the proper schema | |
| 2 | Test case summary | Status code validation | Proper response after sending the postman request |
| Steps | Validate 200 status | |
| Expected result | Response status code is 200 | |
| 3 | Test case summary | Population data integrity test | |
| Step | Validate data length | |
| Expected result | Population data is not empty | |
| Data extraction | | | |
| 4 | Test case summary | Validate the array contains years 2019, 2020 and 2021 | |
| Steps | Define the suitable years should contain the data (according to the actual body content) | |
| Validate each suitable year is included into the objects | |
| Expected results | Population data is retrieved for year 2019 | |
| Population data is retrieved for year 2020 | |
| Population data is retrieved for year 2021 | |
| 5 | Test case summary | Validate the existence of the latest year (fixed) | |
| Step | Compare to the year 2021 | |
| Expected result | The observable Latest year is correctly extracted | |
| 6 | Test case summary | Validate the existence of the latest year (dynamic extraction) | |
| Steps | Query the GetNationalLatestYearData to validate the latest year | |
| Compare with the current body | |
| Expected result | The latest year from the GetNationalLatestYearData API match with this body current year | |
| Error handling | | | |
| 7 | Test case summary | Validate response when invalid parameters are entered in the request | |
|  | Take the API URL and add a non-existent additional parameter | |
|  | Send the request again | |
| Expected result | Knowing an additional parameter will bring an empty body, then we should receive that, making the test successful because the body indicates invalid parameters | |

|  |  |  |  |
| --- | --- | --- | --- |
| Method: GetNationalLatestYearData | | | |
| https://datausa.io/api/data?drilldowns=Nation&measures=Population&year=latest | | | |
| Evaluating API responses | | | |
| 1 | Test case summary | Schema Validation | Validate the proper data schema |
| Steps | Validate objects | |
| Validate properties | |
| Validate required objects | |
| Expected Result | Response has the proper schema | |
| 2 | Test case summary | Status code validation | Proper response after sending the postman request |
| Steps | Validate 200 status | |
| Expected result | Response status code is 200 | |
| 3 | Test case summary | Population data integrity test | |
| Step | Validate data length | |
| Expected result | Population data is not empty | |
| Data extraction | | | |
| 4 | Test case summary | Validate the data contains the proper Nation ID | |
| Steps | Define the nation ID for USA | |
| Compare the information vs that nation ID | |
| Expected results | Proper ID Nation for United States | |
| 5 | Test case summary | Validate the existence of the latest year (fixed) | |
| Step | Compare to the year 2021 | |
| Expected result | The observable Latest year is correctly extracted | |
| 6 | Test case summary | Validate usability of population data | |
| Steps | Get the Population object | |
| Validate is greater than zero | |
| Expected result | Population value is greater than zero | |
| 7 | Test case summary | Validate source information is present in the data | |
| Steps | Check source length | |
| Expected result | Source information is present | |
| 8 | Test case summary | Validate source precedence (assuming this a part of a requirement) | |
| Steps | Check source comes from “Census Bureau” | |
| Expected result | Correct source name | |
| 9 | Test case summary | Validate source description is included | |
| Steps | Check the annotations source description object to be a string | |
| Expected result | Source description has values | |
| 10 | Test case summary | Validate data source URL is a valid URL | |
| Steps | Check URL structure | |
| Expected result | Dataset link has a valid URL | |
| Error handling | | | |
| 11 | Test case summary | Validate response when invalid parameters are entered in the request | |
|  | Take the API URL and add a non-existent additional parameter | |
|  | Send the request again | |
| Expected result | Knowing an additional parameter will bring an empty body, then we should receive that, making the test successful because the body indicates invalid parameters | |

|  |  |  |  |
| --- | --- | --- | --- |
| Method: GetStateLatestYearData | | | |
| https://datausa.io/api/data?drilldowns=State&measures=Population&year=latest | | | |
| Evaluating API responses | | | |
| 1 | Test case summary | Schema Validation | Validate the proper data schema |
| Steps | Validate objects | |
| Validate properties | |
| Validate required objects | |
| Expected Result | Response has the proper schema | |
| 2 | Test case summary | Status code validation | Proper response after sending the postman request |
| Steps | Validate 200 status | |
| Expected result | Response status code is 200 | |
| 3 | Test case summary | Population data integrity test | |
| Step | Validate data length | |
| Expected result | Population data is not empty | |
| Data extraction | | | |
| 4 | Test case summary | Validate the data contains all states + federal district of columbia + puerto Rico | |
| Steps | Count amount of data nodes | |
| Expected results | Validate response contains the number of expected nodes | |
| 5 | Test case summary | Validate all the states belong to the latest year (fixed) | |
| Step | Compare all the state year objects to the year 2021 | |
| Expected result | All states belong to the latest year | |
| 6 | Test case summary | Validate a given state ID | |
| Steps | Select one state (Texas in the example) | |
| Take note of the state ID | |
|  | Compare the body with the noted ID | |
| Expected result | The ID match with the Texas ID | |
| 7 | Test case summary | Validate each state has a unique ID | |
| Steps | First validate the state IDs is not empty | |
|  | Then compare each ID against the list and validate none of them is duplicate | |
| Expected result | Each state has a unique ID | |
| 8 | Test case summary | Population values are positive and integers | |
| Steps | Check all the population objects to be a number | |
| Check all the population objects to be greater than zero | |
| Expected result | Population values are positive integers | |
| 9 | Test case summary | Validate each state has a unique slug | |
| Steps | First validate the state Slugs is not empty | |
|  | Then compare each slug against the list and validate none of them is duplicate | |
| Expected result | Each state has a unique slug | |
| Error handling | | | |
| 10 | Test case summary | Validate response when invalid parameters are entered in the request | |
|  | Take the API URL and add a non-existent additional parameter | |
|  | Send the request again | |
| Expected result | Knowing an additional parameter will bring an empty body, then we should receive that, making the test successful because the body indicates invalid parameters | |

# Cypress Project in Visual Studio Code Test Suite

1. Basic Validations (NBABasicValidations)
   * Veriry the “Sign In” Button is visible on the homepage
   * Verify the clicking on the “Sign In” button displays the second button
   * Verify “the Sign in with NBA ID” button is displayed
   * Verify the clicking of the “Sign in with NBA ID” displays the log in page
2. Date pick validations
   * Date picker is visible
   * Date picker contain today in it
   * Date picker contains today -3 days
   * All the fourth days can be selected
3. Wrong email format (NBAWrongFormatLogIn)
4. Wrong Log in (NBAWrongLogIn)
5. Successful Log In (NBASuccessfulSignIn)

For the Accessibility test, install

npm install --save-dev cypress-axe

1. Accessibility Test (NBAAccessibilityTest)
2. Some basic performance tests
   * Page Load Time: Measure the time it takes for a specific page to fully load. This includes the time taken to fetch and render all page resources such as HTML, CSS, JavaScript, images, and other assets.
   * Time to First Byte: Measure the time taken for the server to respond to the initial request sent by the browser. A low TTFB indicates a fast server response time, while a high TTFB may indicate server-side performance issues or network latency.
3. Some Basic Security Test
   * Test for SQL Injection: Attempt to inject SQL commands into input fields to check if the application properly sanitizes and escapes user inputs to prevent SQL injection attacks.
   * Test for Cross-Site Scripting (XSS): Inject malicious scripts into input fields to see if the application sanitizes user inputs and prevents XSS attacks.