Google Maps API End-to-End Validation

This repository contains test scenarios for validating the Google Maps API functionalities, including adding, updating, and deleting and retrieving a place. These scenarios ensure that the APIs work correctly and as expected. Below, you will find a detailed explanation of the workflow in simple terms.

Table of Contents

Overview

This project involves verifying that a place can be **added, Updated, Deleted, and Retrieved** using APIs. I use a library called RestAssured to interact with the API and Cucumber framework to write tests in plain English.

Feature File: Describes the test scenario in simple language (Gherkin). It outlines the steps to add a place and verify its details.

Step Definitions: Contains the actual code that runs each step described in the feature file. It:

- Prepares the data needed to add a place.
- Sends a request to the API to add the place.
- Checks the API response to ensure the place was added successfully.
- Retrieves the added place's details to verify everything is correct.

We have three main scenarios to validate the Google Maps APIs:

1. **Add a Place**: Ensures a place can be added successfully.
2. **Update a Place**: Ensures the address of a place can be updated successfully.
3. **Delete a Place**: Ensures a place can be deleted successfully.

Test Scenarios

Adding a Place

This scenario tests whether a place can be added using the AddPlaceAPI.

Steps:

1. Provide the necessary details for the place, such as name, language, and address.
2. Call the addPlaceAPI with a POST request.
3. Verify that the API call is successful with a status code of 200.
4. Ensure the response body contains the status "OK" and the scope "APP".
5. Confirm that the place ID created maps to the provided name using the getPlaceAPI.

Updating a Place

This scenario tests whether the address of an existing place can be updated using the UpdatePlaceAPI.

Steps:

1. Provide the new address for the place.
2. Call the updatePlaceAPI with a PUT request.
3. Verify that the API call is successful with a status code of 200.
4. Ensure the response body contains the message "Address successfully updated".
5. Confirm that the updated address maps to the new address using the getPlaceAPI.

Deleting a Place

This scenario tests whether a place can be deleted using the DeletePlaceAPI.

Steps:

1. Provide the details for deleting the place.
2. Call the deletePlaceAPI with a POST request.
3. Verify that the API call is successful with a status code of 200.
4. Ensure the response body contains the status "OK".
5. Confirm that after deleting the place, the message "Get operation failed, looks like place_id doesn't exist" is obtained using the getPlaceAPI.
6. Ensure the API call returns a status code of 404 after deletion.