**Ejercicio 1:**

Request -> Fecha DD-MM-AAAA

2 Headers:

1. Country: Argentina o Chile
2. City: Chile (Santiago, Arica, Chiloe), Argentina(BuenosAires, SanJuan)

Results

1. Positivo: codigo -> http 200 y status -> ok
2. Error:
   1. “client error” y código 400 : en caso de error en la request
   2. “server error” y código 500: en caso de error de conexión

Casos de prueba

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| # de prueba | Inputs | | | Expected Response | |
| date | Country | City | Json | Status Code |
| 1 | 28-02-2022 | Argentina | BuenosAires | {  temperaturaActual: 23,  temperaturaDiaSiguiente: 25  } | 200 |
| 2 | 28-02-2022 | Chile | BuenosAires | {  message: “client error”  } | 400 |
| 3 | 28-02-2022 | Argentina | Santiago | {  message: “client error”  } | 400 |
| 4 | 28-02-2022 | Chile | Chiloe | {  temperaturaActual: 23,  temperaturaDiaSiguiente: 25  } | 200 |
| 5 | 28-02-2022 | Honduras | San Pedro Sula | {  message: “Pais y ciudad incorrectos”  } | 400 |
| 6 | 2022-02-01 | Honduras | San Pedro Sula | {  message: “client error”  } | 400 |
| 7 | 28-02-2022 | Argentina | BuenosAires | {  Message: “server error”  } | 500 |

**Ejercicio 2:**

Escogí estas 3:

1. Posts 100
2. Comments 500
3. Albums 100

Casos de prueba POSTS:

1. GET <https://jsonplaceholder.typicode.com/posts> -> status code 200, y debe retornar un array de 100 posts
2. GET <https://jsonplaceholder.typicode.com/posts>/1 -> status code 200, y debe retornar un elemento con su respectivo id 1
3. GET <https://jsonplaceholder.typicode.com/posts>/200 -> status code 404, y no retorna nada
4. DELETE <https://jsonplaceholder.typicode.com/posts/99> -> status code 200, y no retorna nada

Casos de prueba COMMENTS:

1. GET https://jsonplaceholder.typicode.com/comments-> status code 200, y debe retornar un array de 500 comments
2. GET <https://jsonplaceholder.typicode.com/comments/3> -> status code 200, y debe retornar un elemento con su respectivo id 1 y su respectivo postId
3. GET <https://jsonplaceholder.typicode.com/comments/600> -> status code 404, y no retorna nada
4. DELETE <https://jsonplaceholder.typicode.com/comments/99> -> status code 200, y no retorna nada

Casos de prueba ALBUMS:

1. GET https://jsonplaceholder.typicode.com/albums-> status code 200, y debe retornar un array de 100 albums
2. GET <https://jsonplaceholder.typicode.com/albums/3> -> status code 200, y debe retornar un elemento con su respectivo id 1 y su respectivo postId
3. GET <https://jsonplaceholder.typicode.com/albums/300> -> status code 404, y no retorna nada
4. GET <https://jsonplaceholder.typicode.com/albums/A30> -> status code 404, y no retorna nada
5. DELETE <https://jsonplaceholder.typicode.com/albums/99> -> status code 200, y no retorna nada

URL postman: https://go.postman.co/workspace/KM-Workspace~1b84f0a2-d926-443d-89e5-dcae43f78799/collection/15655709-100fbb7d-d180-4e31-a9cc-7bc4093f8c83

**Ejercicio 3:**

1. Para el primer ejercicio, básicamente lo que hice fue sacar cuales eran las entradas y las posibles salidas para asi poder generar todos los casos de prueba a testear en esa api.

Hay 3 inputs: date, country y city. Con estos 3 inputs se pueden generar muchas variaciones para probar esta api, ya sea el date incorrecto y un country y una city correctos, etc. Y de salida tenemos lo que es el json que devuelve la api y el status code, y en base a todo lo que mencione, mi approach fue usar la técnica de partición de equivalencia, lo explico con un ejemplo

Por ejemplo para chile puede haber hecho mas casos de prueba combinando con cada una de las 3 ciudades disponibles para chile, pero en este caso solo tomo uno para confirmar en que la API este funcionando cuando recibe una ciudad valida, y luego tengo otra prueba para cuando la ciudad es invalida.

Como pueden ver en la tabla que realice coloque en un color rojo para identificar en que campo esta el error por el cual genera un resultado negativo.

Asi mismo quiero aclarar que el ultimo caso de prueba el 7, en este caso con los inputs no hay forma de controlar que el server este down, asi que simplemente hice un caso con inputs validos y que el server devolvió el server error como message

1. Para este ejercicio básicamente lo que hice fue entrar al sitio para hacer un exploratory testing para ver de que trataba la api, y asi pude determinar que casos de prueba se podrían ejecutar, aca en estas pruebas fueron bien básicas, asi mismo como las automatizadas fueron bien básicas, porque por ejemplo en api’s donde se involucra seguridad, hay que hacer pruebas de que sin un token por ejemplo entonces no te devuelva la info. Solicitada.

De igual forma las pruebas que si hice fueron como que ta rápido contesta el server, sabiendo que esto es algo fundamental en una API, que la respuesta no sea demasiado del tiempo debido, porque al final esto terminara impactando a los usuarios finales en tiempos muy largos de espera.