



ACTIVIDAD QA-Auto-Reportes-Karate

Realizado por:

Juan David Cardona

Docente:

Juan Esteban Pineda

29 de Marzo de 2023

Sofka U

ÍNDICE

1.Introducción	3
1.1 Alcance	3
1.2 Fuera del Alcance	3
2. Estrategia	3
Resultados Pruebas del Feature “Create Todos on the Page”	4
• Escenario: Succesfull Task Creation Postman	4
• Escenario: Succesfull Task Creation Srenity	5
Resultados Pruebas del Feature “Update Todos on the Page”	6
• Escenario: Task Update Successful Postman	6
• Escenario: Task Update Successful Serenity	6
Resultados Pruebas Karate Feature “Obtener la lista de NFTs de CoinGecko usando Karate”	8
• Escenario: Obtener la lista de NFTs disponibles Postman	8
• Escenario: Obtener la lista de NFTs filtrado por el id de la blockchain	8
• Resultado de ambos escenarios Karate	9
Resultados Pruebas Karate Feature “Eliminar una tarea especifica de la API de jsonplaceholder”	10
• Escenario: Eliminar la tarea especifica Postman	10
• Escenario: Eliminar la tarea especifica Karate	10
Conclusiones	11

1.Introducción

1.1 Alcance

El propósito de este documento es describir el proceso de pruebas que se llevará a cabo para verificar el funcionamiento de dos servicios, un servicio de POST para crear una nueva tarea con su descripción, un servicio UPDATE para actualizar los campos de una tarea existente que se requiera actualizar.

En karate se probara un servicio GET que permite obtener una lista de ntfs de la api de coingecko y por último un servicio DELETE que permite borrar las tareas existentes en la base de datos de la api Jsonplaceholder.

Para este ciclo de pruebas se realizarán únicamente pruebas funcionales de los servicios y de la plataforma para asegurar que los servicios a probar cumplan con lo esperado.

1.2 Fuera del Alcance

Los factores no funcionales como el rendimiento, la seguridad informática, la usabilidad, la escalabilidad, etc no se probaran en este proceso de prueba, ya que en este ciclo de las pruebas no se tienen definidos los criterios para evaluar estos elementos.

Para algunos de los servicios no se tienen en cuenta los escenarios negativos ya que las apis aceptan como cuerpo de la petición cualquier cosa y siempre devuelven código de estado exitoso independientemente del cuerpo de la solicitud.

2. Estrategia

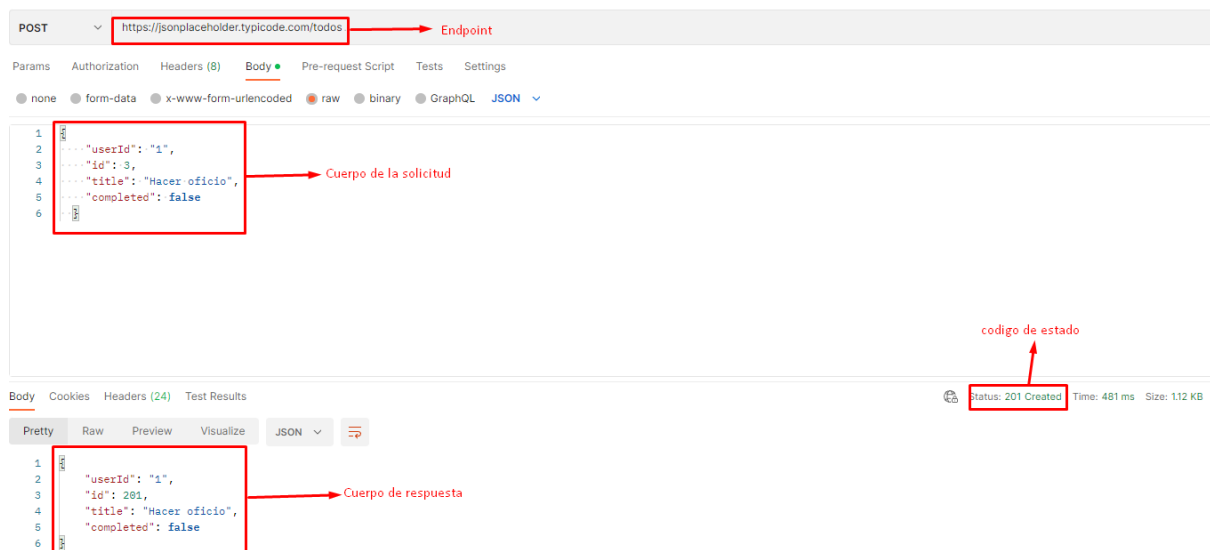
Para la estrategia se tendrán a cabo una serie de actividades descritas a continuación:

- Identificar los requerimientos y funcionalidades principales del software a probar.

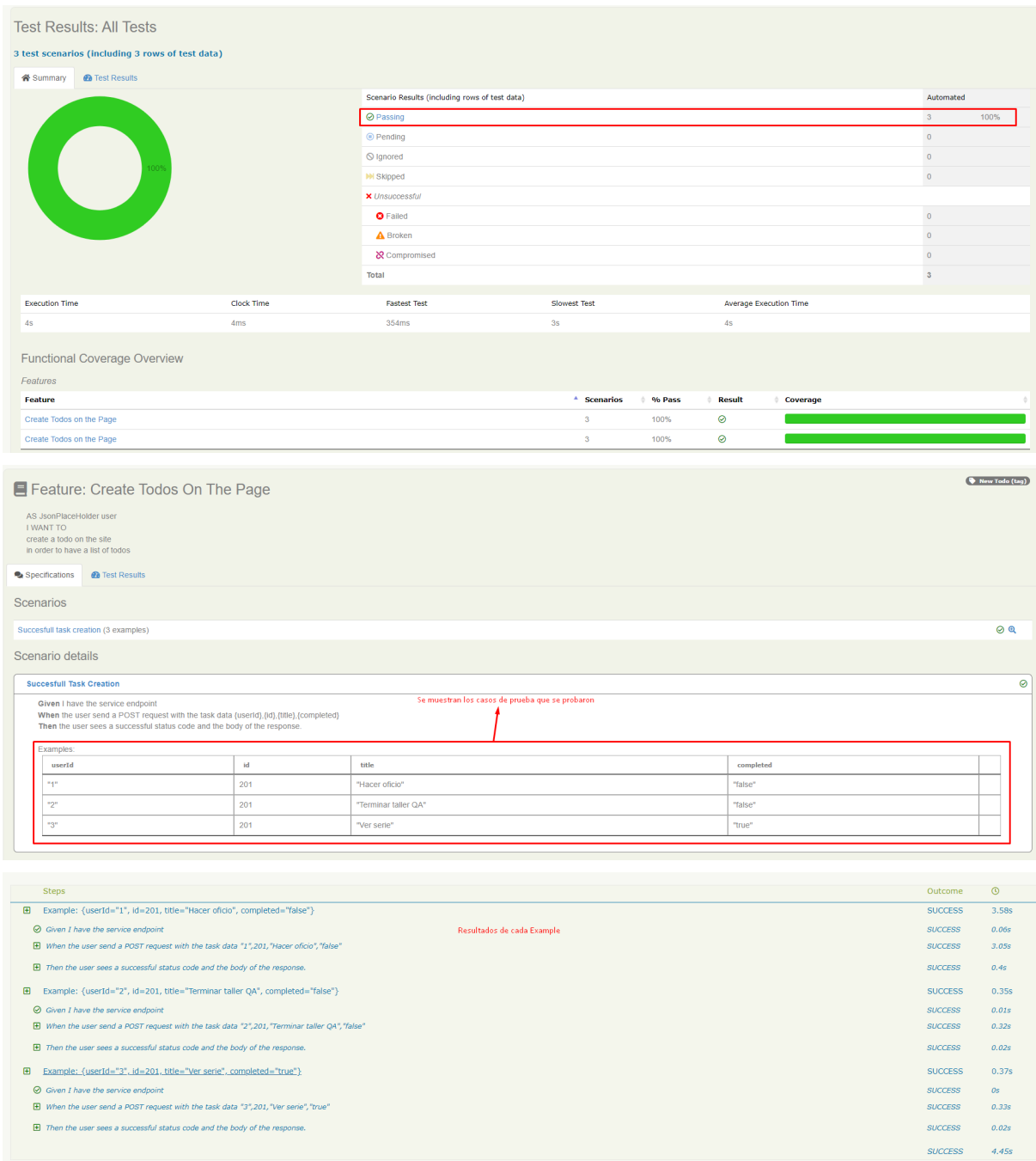
- Identificar los escenarios de prueba basados en los requerimientos y funcionalidades.
- Diseñar los casos de prueba para cada uno de los escenarios de prueba identificados.
- Registrar los resultados de las pruebas en un informe de pruebas.
- Comunicar los defectos encontrados al equipo de desarrollo para su corrección y hacer seguimiento hasta su resolución.

Resultados Pruebas del Feature “Create Todos on the Page”

- Escenario: Succesfull Task Creation Postman



● Escenario: Succesfull Task Creation Srenity



Feature: Create Todos On The Page

AS JsonPlaceholder user
I WANT TO
create a todo on the site
in order to have a list of todos

SpecificationsTest Results

Scenarios

Successful task creation (3 examples)

Scenario details

Successful Task Creation

Given I have the service endpoint
When the user send a POST request with the task data (userId, id, title), (completed)
Then the user sees a successful status code and the body of the response.

Se muestran los casos de prueba que se probaron

Examples:

userId	id	title	completed
"1"	201	"Hacer oficio"	"false"
"2"	201	"Terminar taller QA"	"false"
"3"	201	"Ver serie"	"true"

Steps

Outcome

Example: {userId="1", id=201, title="Hacer oficio", completed="false"}

Given I have the service endpoint

When the user send a POST request with the task data "1",201,"Hacer oficio", "false"

Then the user sees a successful status code and the body of the response.

Example: {userId="2", id=201, title="Terminar taller QA", completed="false"}

Given I have the service endpoint

When the user send a POST request with the task data "2",201,"Terminar taller QA", "false"

Then the user sees a successful status code and the body of the response.

Example: {userId="3", id=201, title="Ver serie", completed="true"}

Given I have the service endpoint

When the user send a POST request with the task data "3",201,"Ver serie", "true"

Then the user sees a successful status code and the body of the response.

Resultados de cada Example

SUCCESS3.58s

SUCCESS0.06s

SUCCESS3.05s

SUCCESS0.4s

SUCCESS0.35s

SUCCESS0.01s

SUCCESS0.32s

SUCCESS0.02s

SUCCESS0.37s

SUCCESS0s

SUCCESS0.33s

SUCCESS0.02s

SUCCESS4.45s

Resultados Pruebas del Feature “Update Todos on the Page”

- Escenario: Task Update Successful Postman

The screenshot shows a Postman interface for a PUT request. The URL is `https://jsonplaceholder.typicode.com/todos/3`, labeled as the **endpoint**. The request body is a JSON object: `{ "userId": "10", "title": "Hacer oficio", "completed": false }`, labeled as the **Cuerpo de solicitud**. The response status is `Status: 200 OK`, labeled as the **codigo de estado**. The response body is a JSON object: `{ "userId": "10", "title": "Hacer oficio", "completed": false, "id": 3 }`, labeled as the **Cuerpo de respuesta**.

- Escenario: Task Update Successful Serenity

The screenshot shows the Serenity BDD Test Results page. A green donut chart indicates a 100% pass rate. The table below shows the results for the 'Update Todos on the Page' feature.

Feature	Scenarios	% Pass	Result	Coverage
Update Todos on the Page	3	100%	⊙	<div></div>
Update Todos on the Page	3	100%	⊙	<div></div>

Feature: Update Todos On The Page

AS JsonPlaceholder user
I WANT TO
update an existing task on the site
in order to change the task definition or status

SpecificationsTest Results

Scenarios

Task update successful (3 examples)

Scenario details

Task Update Successful

Given I have the service endpoint and the task id [id]
When I send a PUT request with the task data that I want to update {userId} {title} {completed}
Then I should see a successful status code and the response body updated.

Casos de Prueba

id	userId	title	completed
1	"2"	"Hacer oficina"	"true"
2	"10"	"	"false"
3	""	"Cenar"	"

Steps

Outcome

0

Example: {id=1, userId="2", title="Hacer oficina", completed="true"}

SUCCESS2.99s

Given I have the service endpoint and the task id 1

SUCCESS0.05s

When I send a PUT request with the task data that I want to update "2","Hacer oficina","true"

SUCCESS2.76s

Then I should see a successful status code and the response body updated.

SUCCESS0.12s

Example: {id=2, userId="10", title="", completed="false"}

SUCCESS0.34s

Given I have the service endpoint and the task id 2

SUCCESS0s

When I send a PUT request with the task data that I want to update "10","", "false"

SUCCESS0.31s

Then I should see a successful status code and the response body updated.

SUCCESS0.01s

Example: {id=3, userId="", title="Cenar", completed=""}

SUCCESS0.33s

Given I have the service endpoint and the task id 3

SUCCESS0s

When I send a PUT request with the task data that I want to update "", "Cenar", ""

SUCCESS0.31s

Then I should see a successful status code and the response body updated.

SUCCESS0.01s

SUCCESS3.78s

Resultados Pruebas Karate Feature “Obtener la lista de NFTs de CoinGecko usando Karate”

- Escenario: Obtener la lista de NFTs disponibles Postman

GET <https://api.coingecko.com/api/v3/nfts/list> Endpoint

Status: 200 OK Time: 378 ms Size: 17.39 KB Save Response

codigo de estado

Listado de nfts

```
1 {
2   "id": "souiggly",
3   "contract_address": "0x36f379480E6c68CDF4488B282F9b685c56adc60",
4   "name": "Souiggly",
5   "asset_platform_id": "ethereum",
6   "symbol": "S-"
7 },
8 {
9   "id": "autoglyphs",
10  "contract_address": "0xd4e4078ca3495de5b1d4cb434bebc5a986197782",
11  "name": "Autoglyphs",
12  "asset_platform_id": "ethereum",
13  "symbol": "A"
14 },
15 {
16   "id": "spacepunksclub",
17   "contract_address": "0x45db714f24f6a313669c41683847f1d49e78ba07",
18   "name": "SpacePunksClub",
19   "asset_platform_id": "ethereum",
20   "symbol": "P"
21 },
22 {
23   "id": "meebits",
24   "contract_address": "0x78d29488f1102bfc23c34f18275b8f23b87168c7",
25   "name": "Meebits",
26   "asset_platform_id": "ethereum",
27   "symbol": "M"
28 },
29 {
30   "id": "xagaboga"
31 }
```

- Escenario: Obtener la lista de NFTs filtrado por el id de la blockchain

GET <https://api.coingecko.com/api/v3/nfts/meebits> id nft

endpoint


Status: 200 OK Time: 151 ms Size: 1.8 KB Save Response

codigo de estado

info nft especifico

```
1 {
2   "id": "meebits",
3   "contract_address": "0x78d29488f1102bfc23c34f18275b8f23b87168c7",
4   "asset_platform_id": "ethereum",
5   "name": "Meebits",
6   "image": {
7     "small": "https://assets.coingecko.com/nft_contracts/images/28/small/meebits.png?16444852788"
8   },
9   "description": "The Meebits are 20,000 unique 3D voxel characters, created by a custom generative algorithm, then registered on the Ethereum blockchain.",
10  "links": {
11    "homepage": "https://meebits.larvalabs.com/",
12    "twitter": "https://twitter.com/larvalabs",
13    "discord": ""
14  },
15  "native_currency": "ethereum",
16  "floor_price": {
17    "native_currency": 2.89,
18    "usd": 5229.49
19  },
20  "market_cap": {
21    "native_currency": 57817,
22    "usd": 104804684
23  },
24  "volume_24h": {
25    "native_currency": 283.25,
26    "usd": 367649
27  },
28  "floor_price_in_usd_24h_percentage_change": 1.07268,
29  "number_of_unique_addresses": 6593.0,
30  "number_of_unique_addresses_24h_percentage_change": -0.13632,
31  "total_supply": 19999.0
32 }
```


● Resultado de ambos escenarios Karate



4
0

Scenarios

2023-03-29 03:27:42 p. m.

[1:8] Obtener la lista de NFTs disponibles

[2:1:23] Obtener la lista de NFTs filtrado por el id de la blockchain

[2:2:24] Obtener la lista de NFTs filtrado por el id de la blockchain

[2:3:25] Obtener la lista de NFTs filtrado por el id de la blockchain

Summary | Tags | Feature: `src/test/java/features/nftApi.feature` | **Obtener la lista de NFTs de CoinGecko usando Karate** Como usuario interesado en los NFTs, Quiero obtener una lista de los NFTs disponibles en CoinGecko Para poder conocer los NFTs populares y estar al día con las últimas tendencias

Scenario: [1:8] Obtener la lista de NFTs disponibles	Resultados primer escenario que lista todos los NFTs	ms: 915
9 Given url 'https://api.coingecko.com/api/v3/nfts/list'		66
10 When method get		803
11 Then status 200		0
12 And match response contains (id: '#string', contract_address: '#string', name: '#string', asset_platform_id: '#string', symbol: '#string')		46

Scenario: [2:1:23] Obtener la lista de NFTs filtrado por el id de la blockchain	Resultados del segundo escenario outline que filtra por id	ms: 336
15 Given url 'https://api.coingecko.com/api/v3'		1
16 And path 'nfts/*-squiggly'		5
17 When method get		329
18 Then status 200		0
19 And match response contains (id: 'squiggly', contract_address: '0x36f379400de6c68cdf4408b282f8b685c56adc60', name: '#string', asset_platform_id: '#string')		1
20 And match \$id == 'squiggly'		2

Scenario: [2:2:24] Obtener la lista de NFTs filtrado por el id de la blockchain		ms: 332
15 Given url 'https://api.coingecko.com/api/v3'		0
16 And path 'nfts/*-angry-ape-army'		0
17 When method get		330
18 Then status 200		0
19 And match response contains (id: 'angry-ape-army', contract_address: '0x77640cf3f89a4f1b5ca3a1e5c873f5b12ebf87e', name: '#string', asset_platform_id: '#string')		0
20 And match \$id == 'angry-ape-army'		1

Scenario: [2:3:25] Obtener la lista de NFTs filtrado por el id de la blockchain		ms: 317
15 Given url 'https://api.coingecko.com/api/v3'		1
16 And path 'nfts/*-meebits'		1
17 When method get		314
18 Then status 200		0
19 And match response contains (id: 'meebits', contract_address: '0x7bd29408f11D2bFC23c34f18275bBf23b8716Bc7', name: '#string', asset_platform_id: '#string')		0
20 And match \$id == 'meebits'		1

15:27:41.954 request:

1 > GET https://api.coingecko.com/api/v3/nfts/squiggly

1 > Host: api.coingecko.com

1 > Connection: Keep-Alive

1 > User-Agent: Apache-HttpClient/4.5.14 (Java/11.0.18)

1 > Accept-Encoding: gzip,deflate

15:27:42.280 response time in milliseconds: 326

1 < 200

1 < Date: Wed, 29 Mar 2023 20:27:41 GMT

1 < Content-Type: application/json; charset=utf-8

1 < Transfer-Encoding: chunked

1 < Connection: keep-alive

1 < X-Frame-Options: SAMEORIGIN

1 < X-XSS-Protection: 0

1 < X-Content-Type-Options: nosniff

1 < X-Download-Options: noopen

1 < X-Permitted-Cross-Domain-Policies: none

1 < Referrer-Policy: strict-origin-when-cross-origin

1 < Cache-Control: public, max-age=120

1 < Access-Control-Allow-Origin: *

1 < Access-Control-Allow-Methods: POST, PUT, DELETE, GET, OPTIONS

1 < Access-Control-Request-Method: *

1 < Access-Control-Allow-Headers: Origin, X-Requested-With, Content-Type, Accept, Authorization

1 < Access-Control-Expose-Headers: link, per-page, total

1 < Vary: Accept-Encoding, Origin

1 < ETag: W/"45670e0e017751623369269210e43e77"

1 < X-Request-Id: 736d7109-6700-4244-bbdf-77553d9258bb

1 < X-RunTime: 0.011184

1 < Alternate-Protocol: 443;npn-spdy/2

1 < CF-Cache-Status: EXPIRED

1 < Expires: Wed, 29 Mar 2023 20:29:41 GMT

1 < Set-Cookie: __cf_bmac=fU4LjPduuSfM2UveDyKfxtUdhswqX1Z17suZyfn7s-1680121661-0-ATjL3sWqIpIch5lqgQACs5rbow68YDYG1N3vCpmsdAPXEWsF1Z3t1e6/916HNBUPUcuxT4YxsZ8DnX3uPao; path=/; expires=Wed, 29-Mar-23 20:57:41 GMT; domain=.api.coingecko.com; HttpOnly; Secure; SameSite=None

1 < Server: cloudflare

1 < CF-RAY: 7afadaddcf08dc7-NIA


1 {"id":"squiggly","contract_address":"0x36f379400de6c68cdf4408b282f8b685c56adc60","asset_platform_id":"ethereum","name":"Squiggly","image":{"small":"https://assets.coingecko.com/nft_contracts/images/1332/small/squiggly.png?1661232628"},"description":"Randomly generated and fully on-chain squiggly lines, the first project in the Atlantes series. Only 100 pieces were created during the minting process. The on-chain generator has now been shut forever so they are only available in the secondary market. Project was launched in October 2020.\r\n\r\nCurve type refers to the type of bezier curve used in the on chain algorithm. All curve types had an equal probability being created.\r\n\r\nAny autolinker who called the end auction function for a given auction was credited as the creator of that Squiggly as they generated the seed to create the art for the auction winner and new owner.\r\n\r\n\r\ncreated by natealex","links":{"homepage":"https://www.squiggly.wtf/","twitter":"","discord":"https://discord.gg/bku7e58"},"native_currency":"ethereum","floor_price":{"native_currency":"14.95","usd":27043},"market_cap":{"native_currency":1495.0,"usd":2704295},"volume_24h":{"native_currency":0.0,"usd":0.0},"floor_price_in_usd_24h_percentage_change":0.89546,"number_of_unique_addresses":54.0,"number_of_unique_addresses_24h_percentage_change":0.0,"total_supply":100.0}

Resultados Pruebas Karate Feature “Eliminar una tarea especifica de la API de jsonplaceholder”

- Escenario: Eliminar la tarea especifica Postman



- Escenario: Eliminar la tarea especifica Karate

 <div>2 0</div> <div>Scenarios</div> <div>2023-03-29 03:35:59 p. m.</div> <div>[1:1:16] Eliminar la tarea especifica</div> <div>[1:2:17] Eliminar la tarea especifica</div>		Summary Tags Feature: <code>src/test/java/features/deletetask.feature</code> Eliminar una tarea especifica de la API de jsonplaceholder <small>Como usuario interesado en eliminar una de mis tareas, Quiero eliminar una de las tareas agregadas Para poder dejar solo las que me interesan</small>	
Scenario: [1:1:16] Eliminar la tarea especifica		ms: 863	
9 Given url 'https://jsonplaceholder.typicode.com'		63	
10 And path '/todos/*10'		3	
11 When method delete		750	
12 Then status 200		0	
13 And match response == {}		48	
Scenario: [1:2:17] Eliminar la tarea especifica		ms: 310	
9 Given url 'https://jsonplaceholder.typicode.com'		1	
10 And path '/todos/*20'		2	
11 When method delete		307	
12 Then status 200		0	
13 And match response == {}		1	

```
15:35:58.493 request:
1 > DELETE https://jsonplaceholder.typicode.com/todos/10
1 > Host: jsonplaceholder.typicode.com
1 > Connection: Keep-Alive
1 > User-Agent: Apache-HttpClient/4.5.14 (Java/11.0.18)
1 > Accept-Encoding: gzip,deflate

15:35:58.964 response time in milliseconds: 466
1 < 200
1 < Date: Wed, 29 Mar 2023 20:35:57 GMT
1 < Content-Type: application/json; charset=utf-8
1 < Content-Length: 2
1 < Connection: keep-alive
1 < X-Powered-By: Express
1 < X-Ratelimit-Limit: 1800
1 < X-Ratelimit-Remaining: 999
1 < X-Ratelimit-Reset: 1680122209
1 < Vary: Origin, Accept-Encoding
1 < Access-Control-Allow-Credentials: true
1 < Cache-Control: no-cache
1 < Pragma: no-cache
1 < Expires: -1
1 < X-Content-Type-Options: nosniff
1 < Etag: W/"2-vyGp6PvF04RvsFtPo1McReyIC8"
1 < Via: 1.1 vegur
1 < CF-Cache-Status: DYNAMIC
1 < Report-To: {"endpoints":[{"url":"https://a.nel.cloudflare.com/report/v3?s=389tN5Ha4CT3%2FR30v01h3P5P04Pcrd0xUfztnfy2PTinnjegx1lr2fyp0o3fuwayP1QeH4Q5Ru3%2Brt0gz03Ck2B8JTv1fy90eJ6ndDB/C0EcxS8s0S0p8EufjECQYsF8uR0z%2Bh5mTsQ1w5%2FGv"}],"group":"cf-nel","max_age":604800}
1 < NEL: {"success_fraction":0,"report_to":"cf-nel","max_age":604800}
1 < Server: cloudflare
1 < CF-RAY: 7afae6fdeabf02c2-HIA
1 < alt-svc: h3=":443"; ma=86400, h3-29=":443"; ma=86400
1 {}
```

En el when se puede ver la solicitud que se hizo

La respuesta con el codigo

Cuerpo de la respuesta

Conclusiones

Lo más importante de este taller para mi fue el descubrimiento de la herramienta karate que permite entender de una manera muy sencilla como se hace una prueba de manera automatizada. También me pone a pensar que quizás sea más fácil probar algunos servicios apoyándose de esta herramienta porque al estar integrado con cucumber solo es necesario escribir un buen feature y no hay que preocuparse de escribir más código.