**PLAN DE PRUEBAS**

1. **OBJETIVO DEL PLAN DE PRUEBAS**

El objetivo de este plan de pruebas es asegurar que todos los endpoints de la API respondan correctamente y de acuerdo a las especificaciones establecidas

1. **ALCANCE**

Este plan de pruebas cubre la selección y automatización de cuatro operaciones básicas de CRUD (Crear - POST , Leer - GET , Actualizar - PUT , Eliminar- DELETE ) para una API, asegurando que los endpoints funcionan correctamente y responden con los datos esperados.

**2.1. Endpoints a probar**

Los endpoints a probar serán

GET : https://reqres.in/[/api/users?page=2](https://reqres.in/api/users?page=2)

POST: https://reqres.in/[/api/users](https://reqres.in/api/users?page=2)

PUT: https://reqres.in/[/api/users](https://reqres.in/api/users?page=2)/2

DELETE : https://reqres.in/[/api/users](https://reqres.in/api/users?page=2)/2

**2.1. Tipos de pruebas**

Este plan cubre:

* Pruebas funcionales y de validación de datos para los endpoints mencionados
* Pruebas automatizadas para los endpoints mencionados anteriormente, verificando su correcto funcionamiento teniendo en cuenta los resultados esperados.

1. **CONFIGURACIÓN DEL ENTORNO**

Se requiere un entorno con Java instalado, Maven para la gestión de dependencias, la dependencia (herramienta) Karate y la herramienta postman para el consumo del servicio previo a la automatización de los escenarios.

1. **CASOS DE PRUEBA**
2. Primer método: GET https://reqres.in/api /users?page=2:

Exitoso

* Given: Se indica la URL del endpoint que se va a probar con una pagina existente por ejemplo GET https://reqres.in/api /users?page=2:
* When: Realiza el llamado al método
* Then: Verifica que la respuesta tenga un código de estado HTTP 200.
* And: Se comprueba alguno de los campos en la respuesta para asegurar que los datos son correctos, por ejemplo "page": 2,

Fallido

* Given: Se indica la URL del endpoint que se va a probar con una pagina no existente , por ejemplo : GET https://reqres.in/api /users?page=2:.
* When: Realiza el llamado al método GET
* Then: Verifica que la respuesta tenga un código de estado HTTP 400 (Se obtiene 200).
* And: Comprueba que no se obtiene nada en el response

1. Segundo método POST https://reqres.in/[/api/users](https://reqres.in/api/users?page=2)

Exitoso

* Given: Establece la URL del endpoint y el body de la solicitud con los campos Name y Job válidos .
* When: Realiza una solicitud al método POST.
* Then: Verifica que la respuesta tenga un código de estado HTTP 201.
* And: Se realiza la verificacion de algunos campos en la respuesta coincidan con los datos enviados, ejemplo: los campos id y createdAt no sean nulos.

Fallido

* Given: Establece la URL del endpoint y en el body data que no corresponde a los tipos de datos de los campos name y job
* When: Realiza una solicitud al método POST.
* Then: Verifica que la respuesta tenga un status code 400 (se obtiene status 200)

1. Método PUT https://reqres.in/[/api/users](https://reqres.in/api/users?page=2)/2

Exitoso

* Given: Establece la URL del endpoint con un nombre de usuario y el cuerpo de la solicitud.
* When: Realiza una solicitud al método PUT.
* Then: Verifica que la respuesta tenga un código de estado HTTP 200.
* And: Se verifica que los campos en la respuesta coincidan con los tipos de datos definidos para cada uno de ellos.

Fallido

* Given: Establece la URL del endpoint con un nombre de usuario vacío y job valido
* When: Realiza una solicitud al método PUT.
* Then: Verifica que la respuesta tenga un código de estado diferente a HTTP 400. (Se obtiene 200)

1. Método DELETE https://reqres.in/[/api/users](https://reqres.in/api/users?page=2)/2

Exitoso

* Given: Establece la URL del endpoint con un ID de usuario.
* When: Realiza una solicitud HTTP DELETE.
* Then: Verifica que la respuesta tenga un código de estado HTTP 204, indicando que la operación fue exitosa y no hay contenido en la respuesta.

Fallido

* Given: Establece la URL del endpoint con un ID de usuario.
* When: Realiza una solicitud HTTP DELETE.
* Then: Verifica que la respuesta tenga un código de estado diferente a HTTP 204, indicando que la operación no fue exitosa y no hay contenido en la respuesta.

1. **GESTION DE RESULTADOS**

Karate genera reportes en formato HTML que se pueden encontrar en el directorio target/karate-reports

**5.1 Análisis de Resultados**

Los resultados de las pruebas se revisarán después de cada ejecución y se ahondarán en las fallas para corregir cualquier problema identificado

1. **MANTENIMIENTO DE LAS PRUEBAS**

**6.1 Actualización de casos de prueba**

Cada vez que haya un cambio en la API, los casos de prueba correspondientes deben ser actualizados y verificados

**6.2 Revisión periódica**

Realizar una revisión trimestral para asegurar que los casos de prueba estén actualizados y sean efectivos.

### **CONCLUSIÓN**

**7.1 Resumen**

Este plan de pruebas asegura que todos los endpoints principales de la API están cubiertos y probados adecuadamente.