**Ejercicio 3 (QAA-03) examen Falabella**

* Algo muy importante es que debemos saber para qué sirve cada método:

1. GET (Para obtener un recurso del servidor)
2. POST (Para crear un recurso del servidor)
3. PUT (Para editar un recurso del servidor)
4. PATCH (Para editar un solo campo de un recurso del servidor)
5. DELETE (Para eliminar un recurso del servidor)

**Y demas**

**Explicacion ejercicio 1**

* Se usa como host miWebApi.com y endpoint api/temperatura.
* Se uso prueba de casos limite.
* Se combinan posibles valores borde de los header [Country, City] y el request [fecha] por separado con sus posibles resultados.

**Explicacion ejercicio 2**

1. Debemos saber el dominio y el endpoint para poder pinchar el servicio, en el caso contrario no vamos a poder probar

En este caso específico el dominio es:

[**https://jsonplaceholder.typicode.com**](https://jsonplaceholder.typicode.com/)

Los endpoint en este examen los elegidos fueron 3:

**users**

**photos**

**todos**

1. Para automatizar los casos de pruebas se va a usar lenguaje JavaScript dentro de Postman en la pestaña TESTS

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Como se puede observar la sintaxis para los métodos es igual a la de un método en JavaScript :

**pm.test(“texto de lo que validamos con ese script” , function () {**

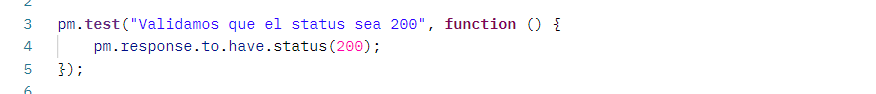
**//Adentro del método el codigo**

**});**

1. Se uso prueba de casos de uso y borde.
2. En nuestro caso vamos a validar lo siguiente:

Hay 72 script automatizados, unos estan en la consulta, unos unos estan en la creacion, unos estan en la edicion, y otros unos estan en la eliminacion (GET, POST, PUT, PATCH, DELETE) vamos a explicar los principales scripts:

1. Status 200 (**Muy importante para validar los estados**)



1. Status 201 (**Muy importante para validar los estados**)

pm.**test**("Validamos que el status sea 201", **function** () {

pm.response.to.have.status(201);

});

1. La estructura del objeto (**Importante ya que validamos que sean los campos correctos que acordamos en el contrato con el cliente**)

Texto

Descripción generada automáticamente

1. Los datos del objeto (**Importante ya que validamos que sean los datos correctos que acordamos en el contrato con el cliente**)

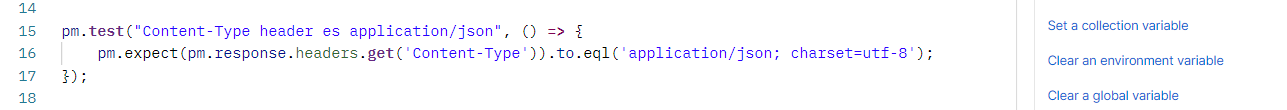
pm.**test**("Validamos que los datos de los posts sea la correcta", () **=>** {

pm.expect(pm.response.json().title).to.equal("Prueba tittle 1");

pm.expect(pm.response.json().url).to.equal("https://via.placeholder.com/600/92c952"); pm.expect(pm.response.json().thumbnailUrl).to.equal("https://via.placeholder.com/150/92c952");

});

1. Que el Content - Type header sea application/json: charset=utf8 (**Validamos que el header contenga Content - Type header sea application/json: charset=utf8)**



1. Tiempo de respuesta (**Validamos el tiempo de respuesta que sea menor a lo acordado**)

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

1. EndPoint incorrecto 404 Not Found (Validamos **que al ingresar un endpoint incorrecto responda lo correcto**)

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

1. Objeto en específico (Validamos **que al buscar un objeto en especifico por su id nos devuelva ese objeto**)

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

**Acá antes creamos una constante que captura el response en formato json**

1. Validamos seguridad status 401 (**En este caso no está, pero es muy importante validar que nos pida un token antes de hacer la ejecución**)

**TENEMOS 3 FORMAS DE EJCUTAR NUESTROS SCRIPT AUTOMATIZADOS**

**Individual mente por request**

1. Solo deberíamos elegir el método, el ambiente y hacer clic en el botón SEND (Azul)

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

1. Luego podemos ir a la pestaña que dice test results para ver los resultados

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

**Ejecutar todos los request de la colección**

1. Nos dirigimos a la colección en especifico y hacemos clic en los tres puntitos y luego en run collection

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

1. Seleccionamos el orden de los request de la colección a ejecutar y luego hacemos clic en run en el botón (azul)

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

1. Podemos observar los resultados

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

**Ejecutar todos los request con Newman**

**Debemos tener instalado**

* Node.js
* NPM
* Newman
* Postman

**Instalación de Newman**

1. Abrimos el CMD y ejecutamos el siguiente comando para instalar Newman ( **npm install -g newman** )
2. Parados en el CMD también vamos a instalar los reportes de html con los siguientes comandos ( **npm install -g newman-reporter-htm**l ) y ( **npm install -g newman-reporter-htmlextra** )

Una vez lista la instalación ya estamos listos para exportar nuestros archivos de Postman

**Exportación**

1. Vamos a ir a Postman y exportar nuestra colección y ambiente en una carpeta determinada por nosotros

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

**Ejecución de los test automatizados**

1. Abrimos el CMD y nos paramos sobre la carpeta que tiene la colección y el ambiente que exportamos, luego ejecutamos el siguiente comando para correr los tests automatizados

**newman run  ("nombre de la coleccion agregando el .json") -e ("nombre del ambiente agregando el .json") -n 1 -r htmlextra**

Texto, Aplicación

Descripción generada automáticamente

**Ver los reportes HTML generados**

1. Nos dirigimos a la carpeta que creamos donde exportamos la colección y el ambiente, al entrar vamos a ver una carpeta que se creó por defecto llamada Newman

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

Vamos a hacer clic en la carpeta newman y vamos a encontrar todos los reportes HTML guardados

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

Hacemos clic en algunos de ellos y veremos el reporte HTML con muchos detalles

Interfaz de usuario gráfica, Sitio web

Descripción generada automáticamente

Interfaz de usuario gráfica

Descripción generada automáticamente con confianza media

**Por ultimo se creo un ambiente en el cual lo reutilizamos para el host y los endpoints**

Reutilizamos las variables para usar ese ambiente en otras colecciones

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente