* Estructura errónea. La verdadera estructura es la siguiente.

{

"Dato\_cliente": {

"Nombre": "STRING",

"Apellido": "STRING",

"Direccion": "STRING",

"Email": "STRING",

"Telefono": "NUMBER",

"RUT": "NUMBER"

},

"Dato\_perfil": {

"Cuenta\_ppal": "number",

"Tipo\_cliente": "string",

"Tarjeta\_debito": "number"

},

"Dato\_Sucursal": {

"Sucursal\_nro": "number",

"Ejecutivo": "string",

"Telefono\_ejec": "number",

"Direccion\_suc": "string"

},

"Dato\_productos": {

"Tiene\_prod": "boolean",

"Tipo\_producto": [

"Consumo",

"Hipotecario",

"Linea\_emergencia",

"Linea\_sobregiro"

],

"Puntos": "number",

"Tarjeta\_credito": "number"

},

"Datos\_seguros": {

"Tiene\_prod": "boolean",

"SOAP": "string",

"Seguro\_vehiculo": "string",

"Seguro\_fraude": "string",

"Seguro\_hogar": "string",

"Seguro\_vida": "string"

},

"Datos\_fondos": {

"Fondo\_mutual": "string",

"Ahorro\_previsional": "number",

"ctaRentaDiaria": "string"

},

"Datos\_penalizaciones": {

"InteresesMora\_Sobregiro": "number",

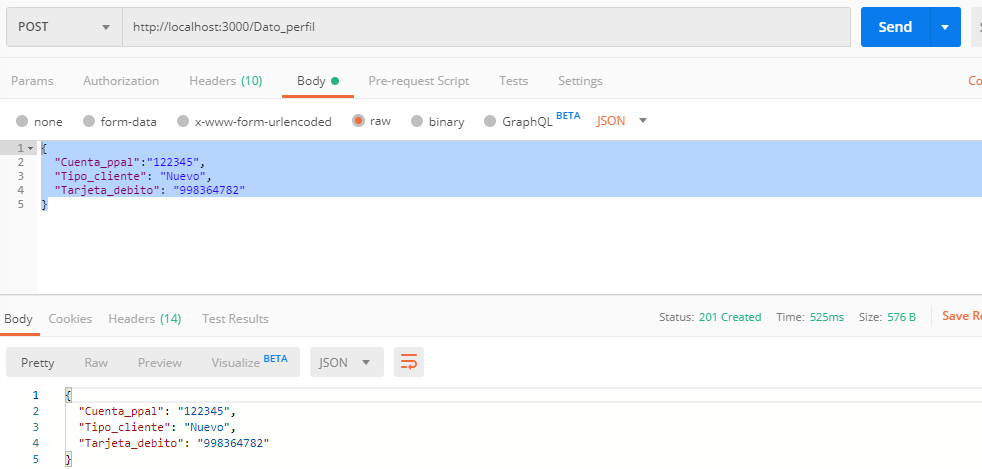
"InteresMora\_TarjetaCredito": "number",

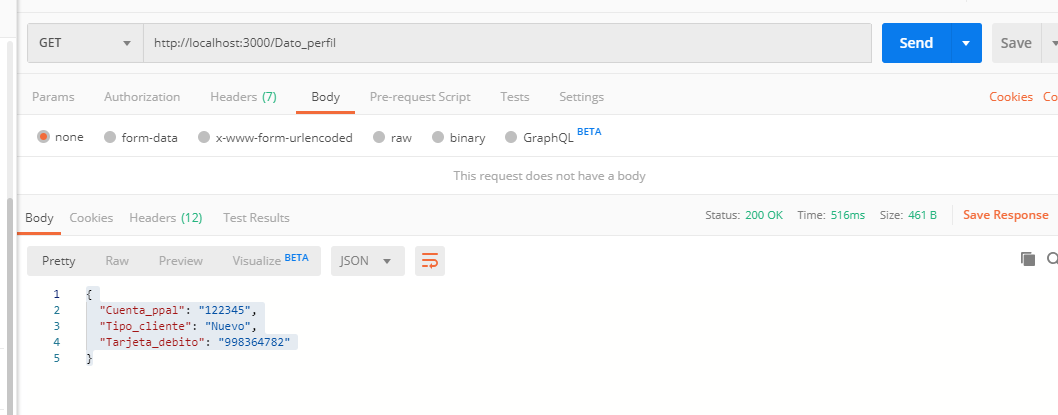
"InteresMora\_Consumo": "number",

"InteresMora\_Hipotecario": "number"

}

* Validar respuesta 200.

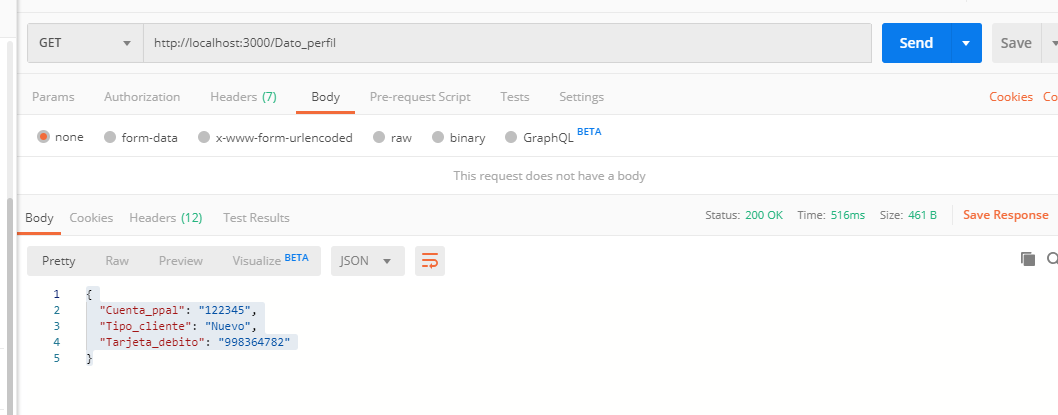




* Validar que los tipos de datos sean correctos.

Los tipos de datos corresponden a los esperado exepto Rut.

* Validar Mensaje OK.



* Validar que el formato sea JSON.

El formato tuvo que ser modificado ya que las variables no estaban entre comillas

* Validar que RUT sea un número.

En realidad rut es un string debido a que existe el digito verificador “K”

2.- Desde la siguiente URL https://restcountries.eu/#api-endpoints-name, tomar el endpoint https://restcountries.eu/rest/v2/name/{name} y realizar las validaciones que se consideren necesarias, se deben escribir minimo 6 tests.

* Obtener todos los valores del endpoint

[GET] <https://restcountries.eu/rest/v2/>

* Ingresar un nuevo valor y que sea guardado apropiadamente.

[POST] <https://restcountries.eu/rest/v2/>

* [GET] <https://restcountries.eu/rest/v2/name/Chile>. Si encuentran el identificador en especifico.
* [PUT] <https://restcountries.eu/rest/v2/name/Chile>-> Actualizar la información de un país en particular
* [DELETE] <https://restcountries.eu/rest/v2/name/Chile>-> Eliminar un país en especifico.

1. Pruebas de Contrato sirven Definir las “interacciones” entre cliente y proveedor. Es decir, qué debe responder el proveedor ante peticiones concretas del consumidor en la cual se crean las pruebas unitarias al respecto. Se puede mockear el comportamiento para observar con anticipación las respuestas preestablecidas, asi se asegurara con anticipación antes de cualquier despliegue cual es el resultado esperado.
2. API Rest. conjunto de restricciones con las que podemos crear un estilo de arquitectura software, las restricciones que definen son: Cliente-servidor, sin estado, cacheable, Interfaz uniforme y sistema de capas.
3. Integracion Continua. Consiste en realizar integraciones automáticas de un proyecto lo mas seguido posible, asi poder detectar fallos lo mas pronto posible.
4. Despliegue Continuo. Un de los puntos mas importante es que no existe intervención humana a la hora de realizar el despliegue de nuestro Software, y se trata de una serie de pasos que deben ejecutarse en orden y forma satisfactoria.
5. Mencione algunas herramientas para realizar pruebas a API y Web Services.

* Postman
* Soapui
* Jmeter

3.- Esta sección es para validar sus conocimientos sobre javascript:

function Sumar(a,b) {

return a + b;

}

function Producto(a,b) {

var c = 20;

return a\*b\*c;

}

Obtener en consola los resultados de las funciones de arriba.

* function Sumar(a,b) {

return a + b;

}

a=prompt('primer numero:','');

b=prompt('segundo numero:','');

b=Sumar(a,b)

console.log(b)

* function Producto(a,b) {

var c = 20;

return a\*b\*c;

}

a=prompt('primer numero:','');

b=prompt('segundo numero:','');

b=Producto(a,b)

console.log(b)

Persona = {

nombre: ‘Juan’,

apellido: ‘Perez’,

telefono: 900199703,

email: ‘juan.perez@gmail.com’

}

Realizar una función que obtenga las propiedades del Objeto Persona.

var Persona = {

nombre: "Juan",

apellido: "Perez",

telefono: "900199703",

email: "juan.perez@gmail.com"

}

console.log(Object.values(Persona))