

# Configurar o Postman para executar APIs no vManage

## Contents

[Introduction](#)

[Requisitos do sistema](#)

[Informações de Apoio](#)

[Configurar o Postman para executar as APIs](#)

[Etapa 1. Abra o Postman e crie uma nova solicitação HTTP.](#)

[Etapa 2. Autentique com suas credenciais de nome de usuário e senha no vManage.](#)

[Etapa 3. Solicitar um token](#)

[Etapa 4. Continue para executar outra API para vManage.](#)

[Etapa 5. Fechar a sessão](#)

[Executar chamadas à API em um ambiente automatizado](#)

[Como salvar o token em uma variável?](#)

[Como limpar o cookie SESSIONID para novas sessões?](#)

[Como utilizar o Collection Runner](#)

## Introduction

Este documento descreve como executar Application Programming Interfaces (APIs) com Postman.

## Requisitos do sistema

- Postman instalado
- Acesso ao vManage e às credenciais de nome de usuário e senha

**Observação:** Se você não tiver o Postman, baixe-o de <https://www.postman.com/downloads/>

## Informações de Apoio

Os verbos HTTP primários ou mais comumente usados (ou métodos, como são chamados corretamente) são POST, GET, PUT, PATCH e DELETE.

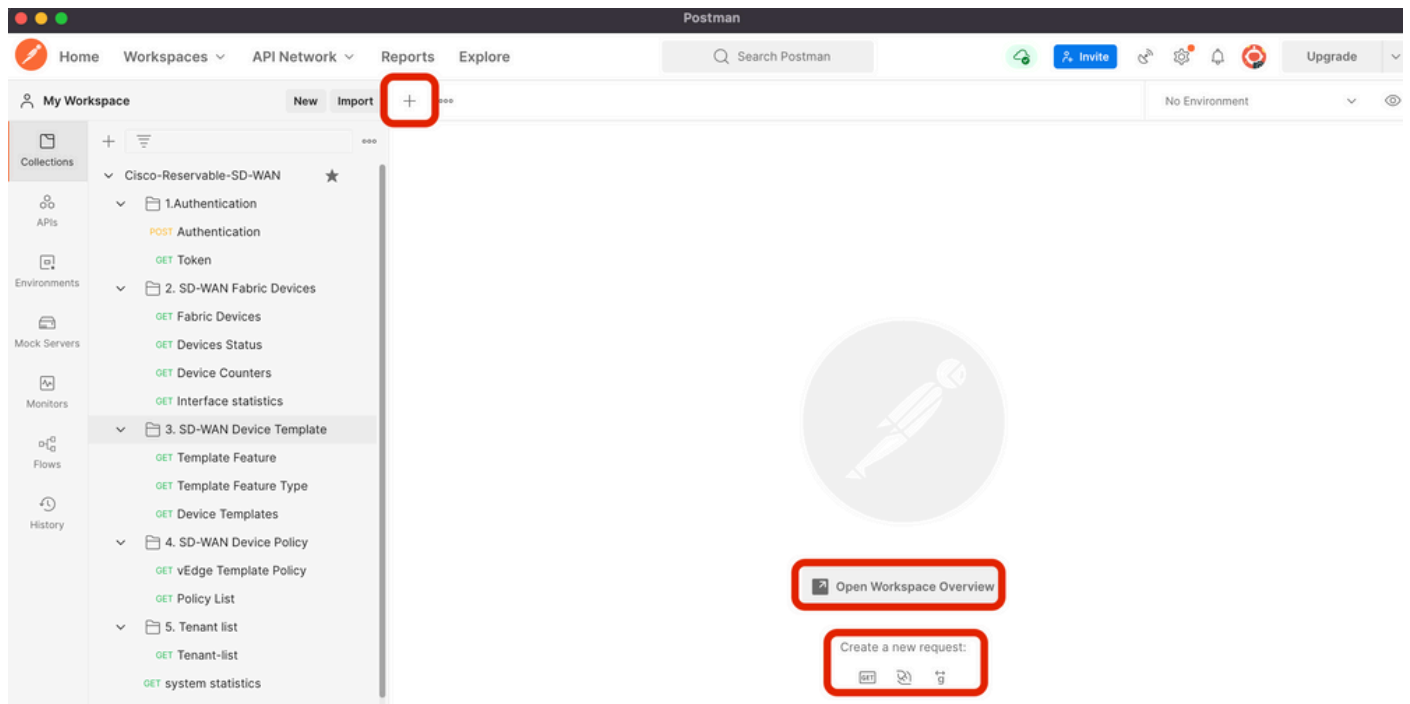
Elas correspondem a criar, ler, atualizar e excluir (ou CRUD) operações, respectivamente.

Há uma série de outros verbos, também, mas são utilizados com menos frequência. Desses métodos menos frequentes, OPTIONS e HEAD são usados com mais frequência do que outros.

## Configurar o Postman para executar as APIs

**Etapa 1. Abra o Postman e crie uma nova solicitação HTTP.**

Você pode criar novas solicitações HTTP se clicar em qualquer uma das opções destacadas.



Crie uma nova solicitação HTTP.

## Etapa 2. Autentique com suas credenciais de nome de usuário e senha no vManage.

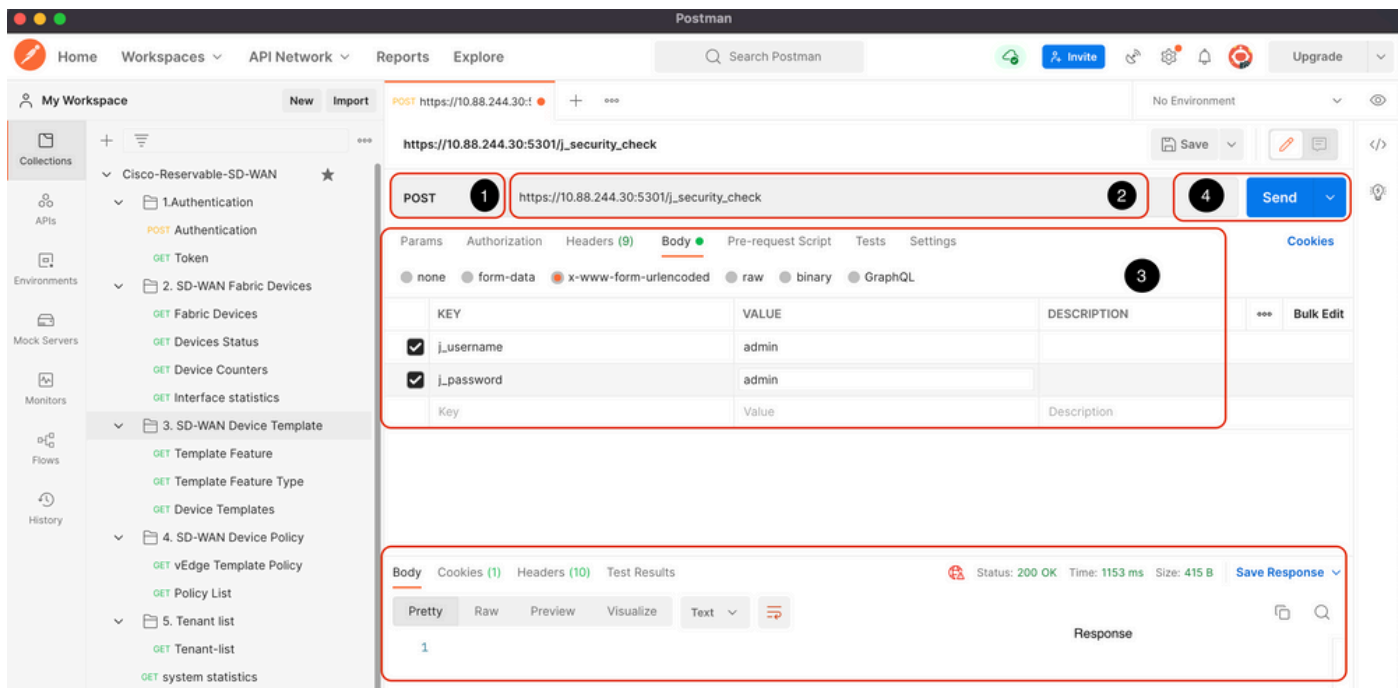
Criar outra solicitação HTTP.

1. Selecione **POST** como seu verbo HTTP.
2. Adicione [https://<vmanage-ip>/j\\_security\\_checknext](https://<vmanage-ip>/j_security_checknext) ao POST.
3. Clique em **Body** e adicione como **KEY** os parâmetros **j\_username** e **j\_password** e seus valores, respectivamente.
4. Clique em **Enviar**.

**Observação:** neste exemplo, o endereço ip do vManage é 10.88.244.30 e a porta é 5301

**Observação:** como valores de nome de usuário e senha, usamos admin.

Preencha os parâmetros em Postman.



Autenticação vManage.

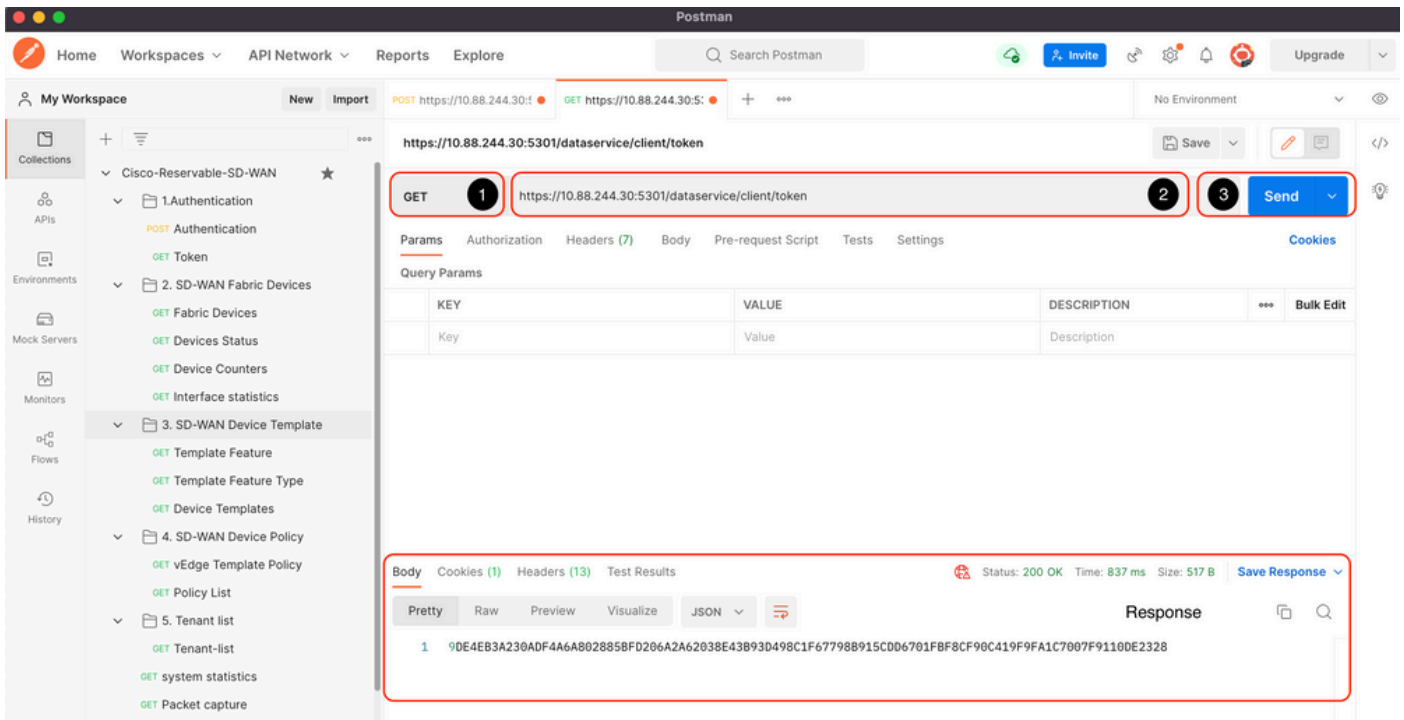
**Cuidado:** a resposta desta chamada à API deve estar vazia

### Etapa 3. Solicitar um token

1. Selecione **GET** como seu verbo HTTP.
2. Adicione os detalhes da chamada à API ao lado de GET <https://<vmanage-ip>/dataservice/client/token>
3. Clique em **Enviar**

**Observação:** desde a versão 19.2.1 do vManage, tornou-se obrigatório que um usuário conectado com êxito precise enviar um token X-XSRG-TOKEN ou CSRF para cada operação POST/PUT/DELETE por chamada de API.

Uma vez executada a chamada à API, você obtém uma string de resposta no corpo. Salve essa string. A imagem mostrada exemplifica a saída In Postman.



Solicite um token para o vManage

**Aviso:** se você não obteve um token como mostrado na imagem, repita a etapa.

## Etapa 4. Continue para executar outra API para vManage.

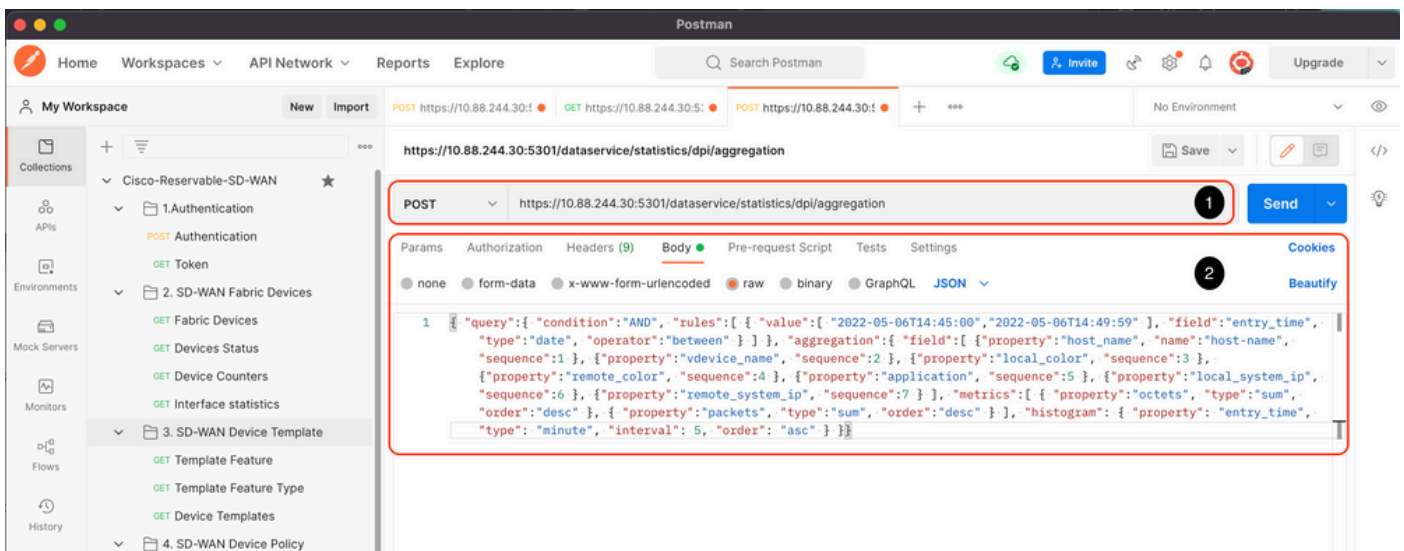
Este exemplo envolve uma solicitação POST

1. Selecione a chamada à API a ser executada, no nosso caso é <https://dataservice/statistics/dpi/aggregation>

**Dica:** se você deseja explorar outras chamadas de API, vá para a URL do vManage <https://vmanage-ip:port/apidocs>

2. Colete o corpo da chamada da API.

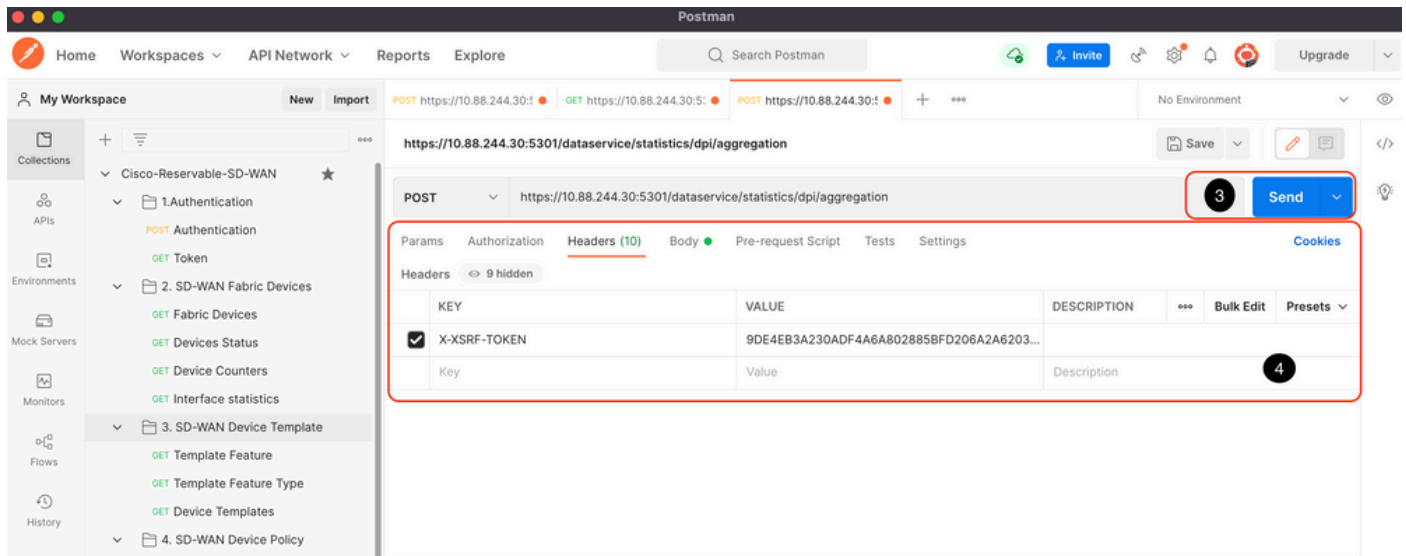
**Observação:** esta chamada à API contém um corpo no formato JSON



3. Clique em **Cabeçalho** e adicione como **Chave** a string **X-XSRF-TOKEN** como valor.

4. Clique em **Enviar**.

A imagem mostrada mostra como sua chamada API deve aparecer.



Chamada à API de agregação de DPI.

## Etapa 5. Fechar a sessão

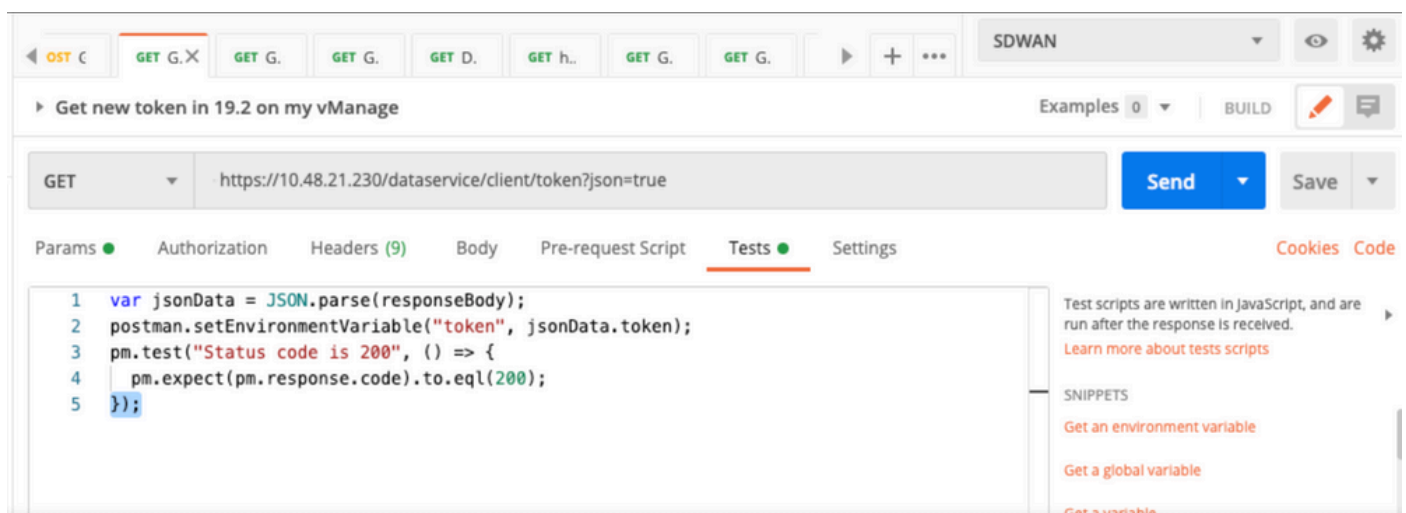
Depois de recuperar todas as informações necessárias do vManage e/ou dos dispositivos, libere recursos do vManage e elimine a possibilidade de usuários mal-intencionados usarem sua sessão.

## Executar chamadas à API em um ambiente automatizado

Salvar cookies e variáveis a serem usados em chamadas de API subsequentes

### Como salvar o token em uma variável?

Salve o token em uma variável para reutilização subsequente.



Salvar o token em uma variável

Quando solicitarmos o token no formato JSON, armazenemo-o. Use a guia **Testes** e cole as linhas mostradas.

```
var jsonData = JSON.parse(responseBody);  
postman.setEnvironmentVariable("token", jsonData.token);
```

Depois, qualquer chamada de API pode usar uma variável de token.

Key	Value	Description
<input checked="" type="checkbox"/> Host	<calculated when request is sent>	
<input checked="" type="checkbox"/> User-Agent	PostmanRuntime/7.26.3	
<input checked="" type="checkbox"/> Accept	*/*	
<input checked="" type="checkbox"/> Accept-Encoding	gzip, deflate, br	
<input checked="" type="checkbox"/> Connection	keep-alive	
<input checked="" type="checkbox"/> X-XSRF-TOKEN	{{token}}	
<input checked="" type="checkbox"/> Content-Type	application/json	

Usar a variável de token

## Como limpar o cookie SESSIONID para novas sessões?

Sempre que você executar a chamada à API para sair do, use JSESSIONID.

Não podemos usar nenhuma autenticação básica como fizemos em versões anteriores. Em vez disso, fornecemos apenas credenciais e salvamos a ID em nosso cookie. Antes disso, podemos usar um pré-teste para limpar todos os cookies ou cookies específicos.

```
1 const jar = pm.cookies.jar();  
2  
3 jar.clear(pm.request.url, function (error) {  
4 // error - <Error>  
5 });
```

Limpar cookies

Isso é feito por meio do código inserido no Script de pré-solicitação.

## Como utilizar o Collection Runner

Agora que temos um ambiente no qual podemos executar sessões e salvar dados específicos para cada sessão, você pode executar uma sequência de chamadas usando o Collection Runner.

Selecione a ordem dos eventos que deseja repetir, selecione a contagem de repetições para que Postman possa executar as chamadas de API, o número selecionado de vezes com resultados por execução.

Choose a collection or folder

Search for a collection or folder

Viptela

POST https://10.48.21.230/apidocs

POST https://10.48.21.230/dataservice/device

POST https://10.48.21.230:443/dataservice/statistics/approute

POST https://10.48.21.230:443/dataservice/statistics/approute

POST https://10.48.21.230:443/dataservice/statistics/approute

Environment **SDWAN**

Iterations

Delay  ms

Data

Save responses ⓘ

Keep variable values ⓘ

Run collection without using stored cookies

Save cookies after collection run ⓘ

RUN ORDER

Deselect All Select All Reset

- POST Get JSESSIONID in newer release(s)
- GET Get new token in 19.2 on my vManage
- GET Get server info with in-correct token
- GET Get server information 19.2 lab vManage with correct token
- POST https://10.48.21.230/apidocs
- POST https://10.48.21.230/dataservice/device
- POST https://10.48.21.230:443/dataservice/statistics/approute
- POST https://10.48.21.230:443/dataservice/statistics/approute
- POST https://10.48.21.230:443/dataservice/statistics/approute
- POST https://10.48.21.230:443/dataservice/statistics/approute
- GET https://10.48.21.230:443/dataservice/statistics/approute
- POST https://10.48.21.230/dataservice/system/device
- POST https://10.48.21.230:443/dataservice/template/device/config/config
- POST https://10.48.21.230/dataservice/system/device/fileupload
- PUT https://mtv5-sdwan-vman-1
- POST https://10.48.21.230/dataservice/system/device
- POST https://10.48.21.230/dataservice/system/device
- GET Get new token in 19.2
- GET inventory call
- GET https://10.48.21.230/dataservice/alarms
- GET Get alarms on my lab vManage

*Executor de Coleção*

Na "biblioteca" de chamadas, coloque-as em uma determinada ordem para obter um fluxo/ordem específico a ser executado.

Coloque um resultado para verificar se você recebe um valor de 200 OK ou outro como resposta e trate-o como aprovado ou reprovado.

The screenshot shows the Postman interface for a REST client. The request is a GET method to the URL `https://10.48.21.230/dataservice/client/token?json=true`. The 'Tests' tab is active, containing the following JavaScript code:

```
1 var jsonData = JSON.parse(responseBody);
2 postman.setEnvironmentVariable("token", jsonData.token);
3 pm.test("Status code is 200", () => {
4   pm.expect(pm.response.code).to.eql(200);
5 });
```

The response is displayed in the 'Body' tab, showing a JSON object with a 'token' property:

```
1 {
2   "token": "23AE920117579F0EF9D470C2DE837A74C292D6A5929E098E06AB6358D399A61BD99B23D17D836D36EE0BAF764E1B10D52059"
3 }
```

Additional details shown include 'Status: 200 OK', 'Time: 67 ms', and 'Size: 550 B'. The interface also shows various tabs like Params, Authorization, Headers, Body, Pre-request Script, Tests, and Settings.

Verificar código de resposta

```
pm.test("Status code is 200", () => {
  pm.expect(pm.response.code).to.eql(200);
});
```

Então podemos ver aprovado ou reprovado em nossas execuções.



Collection Runner Run Results My Workspace Run In Command Line Docs

20 PASSED 0 FAILED Viptela SDWAN just now

Run Summary Export Results Retry New

Iteration 1

- POST Get JSESSIONID in newer release(s) https://10.48.21.230/j\_se... Viptela / Get JSESSIONID in newer ...
  - Status code is 200
- GET Get new token in 19.2 on my vManage https://10.48.21.230/dat... Viptela / Get new token in 19.2 on... 200 OK 53 ms 550 B
  - Status code is 200
- GET Get server info with in-correct token https://10.48.21.230/dat... Viptela / Get server info with in-co... 403 Forbidden 56 ms 583 B
  - Status code is 403
- GET Get server information 19.2 lab vManage with correct token https://10.48.21.230/dat... Viptela / Get server information 1... 200 OK 49 ms 486 B
  - Status code is 200

Iteration 2

- POST Get JSESSIONID in newer release(s) https://10.48.21.230/j\_se... Viptela / Get JSESSIONID in newer ...
  - Status code is 200
- GET Get new token in 19.2 on my vManage https://10.48.21.230/dat... Viptela / Get new token in 19.2 on... 200 OK 48 ms 550 B
  - Status code is 200
- GET Get server info with in-correct token https://10.48.21.230/dat... Viptela / Get server info with in-co... 403 Forbidden 49 ms 583 B
  - Status code is 403

Console

Execução automatizada

## Sobre esta tradução

A Cisco traduziu este documento com a ajuda de tecnologias de tradução automática e humana para oferecer conteúdo de suporte aos seus usuários no seu próprio idioma, independentemente da localização.

Observe que mesmo a melhor tradução automática não será tão precisa quanto as realizadas por um tradutor profissional.

A Cisco Systems, Inc. não se responsabiliza pela precisão destas traduções e recomenda que o documento original em inglês ([link fornecido](#)) seja sempre consultado.