

Entender e usar as APIs do Catalyst Center

Contents

[Introdução](#)

[Informações de Apoio](#)

[Pré-requisitos](#)

[Requisitos](#)

[Componentes Utilizados](#)

[Informações de Apoio](#)

[Introdução ao HTTP](#)

[A solicitação HTTP](#)

[A Resposta HTTP](#)

[Métodos HTTP \(Verbos\)](#)

[Códigos de status HTTP](#)

[Visão geral do REST](#)

[APIs Catalyst Center e REST](#)

[Recuperar um token de autenticação usando Bruno](#)

[1. Identificar o Ponto de Extremidade da API](#)

[2. Configurar Bruno para Autenticação](#)

[3. Defina um Cabeçalho Global no Bruno](#)

[Exemplo 1: Criar um andar do site](#)

[Passo 1: Obter a ID do Pai](#)

[Passo 2: Criar a tribuna](#)

[Exemplo 2: Recuperar informações de interface](#)

[Passo 1: Identificar a ID do dispositivo](#)

[Passo 2: Crie uma solicitação parametrizada no Bruno](#)

[Conclusão](#)

Introdução

Este documento descreve como acessar e utilizar as APIs CaTC Northbound (Intent) usando um cliente de API e as ferramentas de desenvolvedor incorporadas da plataforma.

Informações de Apoio

Ele abrange conceitos básicos de HTTP/REST e fornece exemplos práticos para automação e

teste.

Pré-requisitos

Requisitos

A Cisco recomenda o conhecimento destes tópicos:

- Protocolos HTTP/1.1 e códigos de status.
- Estilo arquitetônico REST.
- Noções básicas sobre o Cisco Catalyst Center (CaTC)
- Entendimento básico de Bruno (ou clientes API semelhantes, como Postman).

Componentes Utilizados

As informações neste documento são baseadas nas seguintes versões de hardware e software:

- Cisco Catalyst Center versões 2.3.7.x e 3.x
- Bruno versão 4.1.0

As informações neste documento foram criadas a partir de dispositivos em um ambiente de laboratório específico. Todos os dispositivos utilizados neste documento foram iniciados com uma configuração (padrão) inicial. Se a rede estiver ativa, certifique-se de que você entenda o impacto potencial de qualquer comando.

Informações de Apoio

As APIs de Intenção do Catalyst Center permitem que os usuários interajam com o controlador fora da interface gráfica do usuário. Isso permite a automação de fluxos de trabalho personalizados, integração com ferramentas de terceiros e gerenciamento de rede eficiente em escala.

Introdução ao HTTP

O protocolo HTTP é a base da comunicação de dados para a Web. Ele adere a um modelo

cliente-servidor em que o cliente envia uma solicitação e o servidor retorna uma resposta.

A solicitação HTTP

Uma solicitação HTTP consiste em quatro partes:

1. Linha de solicitação: Especifica o método (incluindo GET e POST), o URI e a versão do protocolo (como GET /api/v1/nodes HTTP/1.1).
2. Cabeçalhos: Forneça metadados como credenciais de autenticação e tipo de conteúdo.
3. Linha vazia: Separa cabeçalhos do corpo.
4. Corpo da mensagem (opcional): Contém dados enviados ao servidor (comuns em POST/PUT/PATCH).

A Resposta HTTP

Uma resposta HTTP espelha a estrutura da solicitação:

1. Linha de status: Inclui o código de status (como 200 OK).
2. Cabeçalhos: Detalhes sobre o servidor e os dados de resposta.
3. Corpo da mensagem: Os dados reais solicitados (geralmente no formato JSON para o Catalyst Center).

Métodos HTTP (Verbos)

- GET: Recupera dados de um recurso.
- POST: Cria um novo recurso.
- PUT: Substitui um recurso existente.
- CORREÇÃO: Aplica modificações parciais a um recurso.
- DELETE: Remove um recurso.

Códigos de status HTTP

Os códigos de status HTTP são códigos numéricos padronizados retornados por um servidor como parte de uma resposta HTTP. Eles indicam o resultado geral de uma solicitação de cliente e ajudam a entender se a solicitação foi bem-sucedida, requer ação adicional ou encontrou um erro.

Os códigos são agrupados em cinco categorias principais, de acordo com o seu primeiro dígito.

- Os códigos 1xx são informativos e indicam que a solicitação foi recebida e que o processamento continua.
- Os códigos 2xx indicam uma solicitação bem-sucedida, o que significa que o servidor recebeu, entendeu e processou a solicitação com êxito.
- Os códigos 3xx são usados para redirecionamento e indicam que uma ação adicional pode ser necessária para concluir a solicitação.
- Os códigos 4xx representam erros do cliente, o que significa que há um problema com a solicitação enviada pelo cliente.
- Os códigos 5xx representam erros de servidor, o que significa que o servidor encontrou um problema ao tentar processar uma solicitação.

Visão geral do REST

REST (Representational State Transfer) é um estilo arquitetônico para projetar aplicativos em rede. As APIs RESTful usam HTTP como mecanismo de transporte e geralmente usam JSON para troca de dados devido à sua natureza leve.

Quando um cliente interage com uma API REST, o servidor retorna uma representação do estado atual de um recurso. Essa representação é transferida para o cliente como parte da resposta e pode ser expressa em um formato diferente.

REST não é um protocolo. Em vez disso, é um estilo hierárquico que define um conjunto de princípios e restrições para a construção de sistemas distribuídos e APIs. Nas APIs modernas, o JSON é um dos formatos mais comuns porque é leve e fácil de processar.

Uma interação REST típica segue um modelo baseado no cliente (o qual foi mencionado anteriormente no artigo). O cliente envia uma solicitação HTTP usando métodos como GET, POST, PUT, PATCH ou DELETE. O servidor processa e retorna uma resposta HTTP contendo as informações solicitadas ou o resultado da operação.

As APIs RESTful geralmente usam HTTP como o mecanismo de comunicação para executar operações em recursos. Neste modelo, o cliente identifica um recurso/ponto final e usa o método HTTP para indicar a ação a ser executada nesse recurso. Uma API REST pode ser descrita usando vários elementos principais. Primeiro, o URI é usado para identificar o recurso ou ponto de extremidade que o cliente deseja acessar. O URI fornece o local ou o identificador desse recurso dentro da API. Segundo, o cliente usa métodos HTTP padrão, como GET, POST, PUT, PATCH e DELETE para definir a operação de solicitação. Por exemplo, GET pode ser usado para recuperar um recurso, POST para criar um e DELETE para remover um por completo. Finalmente, o recurso é trocado através de uma representação, que é definida por seu tipo de mídia. Essa representação é a forma na qual os dados do recurso são transferidos entre o cliente e o servidor, como JSON ou XML.

APIs Catalyst Center e REST

O CaTC ajuda a simplificar as operações de rede do campus centralizando a automação, a configuração, o gerenciamento de políticas, o monitoramento e a análise. Permite que os administradores de rede implantem e gerenciem dispositivos com mais eficiência, mantenham políticas consistentes, monitorem o comportamento da rede e melhorem a visibilidade operacional geral. Esses recursos podem ser acessados por meio da interface gráfica do usuário (GUI), bem como por meio de APIs e integrações externas.

A plataforma CaTC também fornece APIs que podem ser usadas para criar integrações personalizadas e automatizar operações de rede. Combinando APIs CaTC com soluções Cisco existentes ou aplicativos de terceiros, as organizações podem ampliar a funcionalidade da plataforma e criar fluxos de trabalho personalizados para operações ou requisitos comerciais específicos.

Depois de abordar os conceitos básicos, este documento se concentra nas APIs ascendentes, também chamadas de APIs de intenção, no Cisco Catalyst Center. Essas APIs expõem recursos específicos do Catalyst Center por meio de uma abstração baseada em políticas do objetivo comercial, permitindo que os administradores e desenvolvedores se concentrem no resultado desejado em vez das etapas de configuração individuais necessárias para atingi-lo.

As APIs Catalyst Center Intent são APIs RESTful que usam o formato de dados JSON para interagir com a plataforma. Eles podem ser usados para descobrir informações de rede, recuperar dados operacionais e executar tarefas de controle e automação suportadas na rede.

Recuperar um token de autenticação usando Bruno

Antes de fazer chamadas à API, você deve se autenticar para receber um token.

1. Identificar o Ponto de Extremidade da API

1. Faça login no Catalyst Center.
2. Navegue até Menu > Plataforma > Developer Toolkit.
3. No menu à esquerda, selecione Authentication > Authentication API.

The screenshot shows the Cisco Catalyst Center Developer Toolkit interface. The top navigation bar includes the Cisco logo, 'Catalyst Center', and 'Platform / Developer Toolkit'. Below this, there are tabs for 'APIs', 'Integration Flows', and 'Event Notifications'. A search bar is present on the right. On the left, a sidebar lists various categories: 'Appliance', 'Authentication' (highlighted with a red box), 'No Subdomain', 'Cisco DNA Center System', 'Connectivity', 'Ecosystem Integrations', 'Event Management', 'Integrations', 'Know Your Network', 'Operational Tasks', 'Policy', 'Site Management', 'System', and 'System Settings'. The main content area is titled 'Authentication' and contains a table of APIs. The table has columns for 'Method', 'Name', 'Description', 'URL', and 'Actions'. Three APIs are listed: 'importCertificate' (POST, URL: /dna/intent/api/v1/certificate), 'importCertificateP12' (POST, URL: /dna/intent/api/v1/certificate-p12), and 'Authentication API' (POST, URL: /dna/system/api/v1/auth/token, highlighted with a red box). Below the table, the 'Cisco DNA Center System' is listed.

Method	Name	Description	URL	Actions
POST	importCertificate	This API enables a user to import a PEM certificate and its key for the controller and/or disaster recovery.	/dna/intent/api/v1/certificate	...
POST	importCertificateP12	This API enables a user to import a PKCS12 certificate bundle for the controller and/or disaster recovery.	/dna/intent/api/v1/certificate-p12	...
POST	Authentication API	API to obtain an access token, which remains valid for 1 hour. The token obtained using this API is required to be set as value to the X-Auth-Token HTTP...	/dna/system/api/v1/auth/token	...

Kit de Ferramentas do Desenvolvedor do Catalyst Center

4. Copie a URL (geralmente POST /dna/system/api/v1/auth/token).

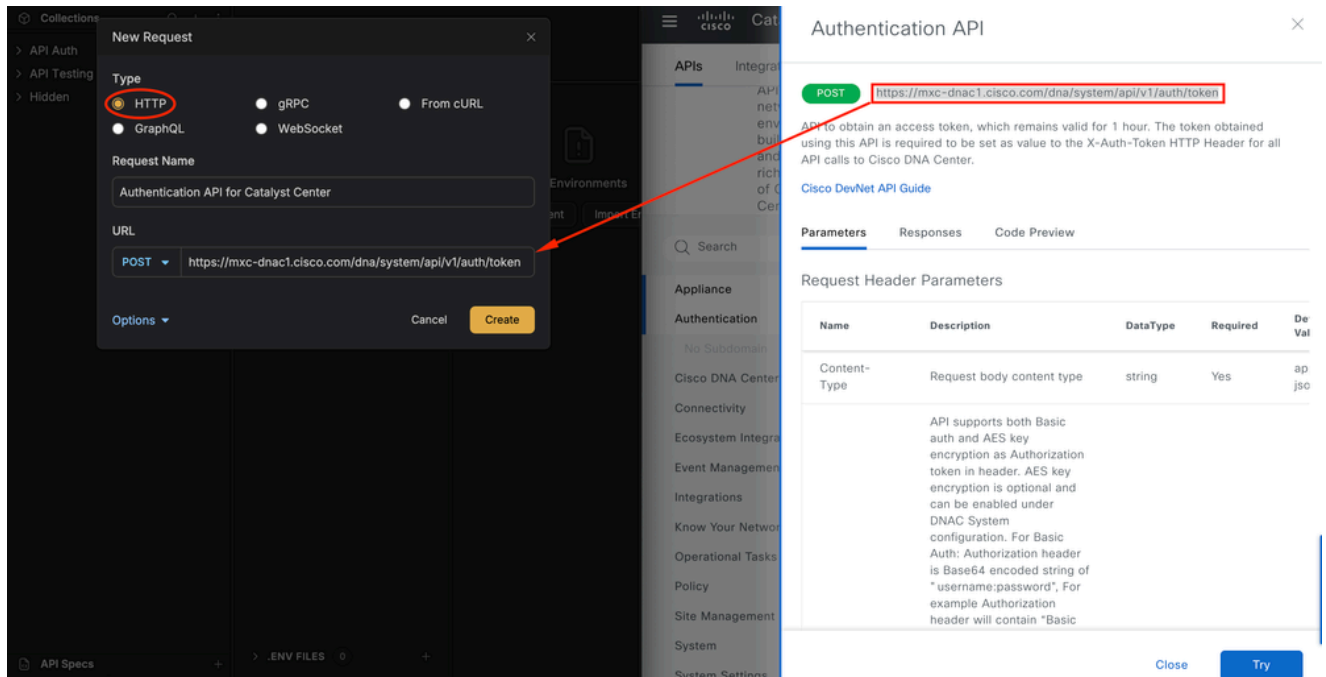
2. Configurar Bruno para Autenticação

1. No Bruno, crie uma Nova Solicitação.
2. Defina Type como HTTP e o Method como POST.

The screenshot shows the Bruno API client interface. The left sidebar displays a list of collections: 'API Auth', 'API Testing', and 'Hidden'. The main workspace is titled 'My Workspace' and shows a 'New Request' button highlighted with a red box. Below this, there are buttons for 'New Folder', 'New App', 'New JS File', 'Run', 'Clone', 'OpenAPI', 'Rename', 'Share', 'Generate Docs', 'Collapse', 'Reveal in Finder', 'Settings', 'Open in Terminal', 'Move into Workspace', and 'Remove'. The right pane shows 'No Environments' with buttons for 'Create Environment' and 'Import Environment'.

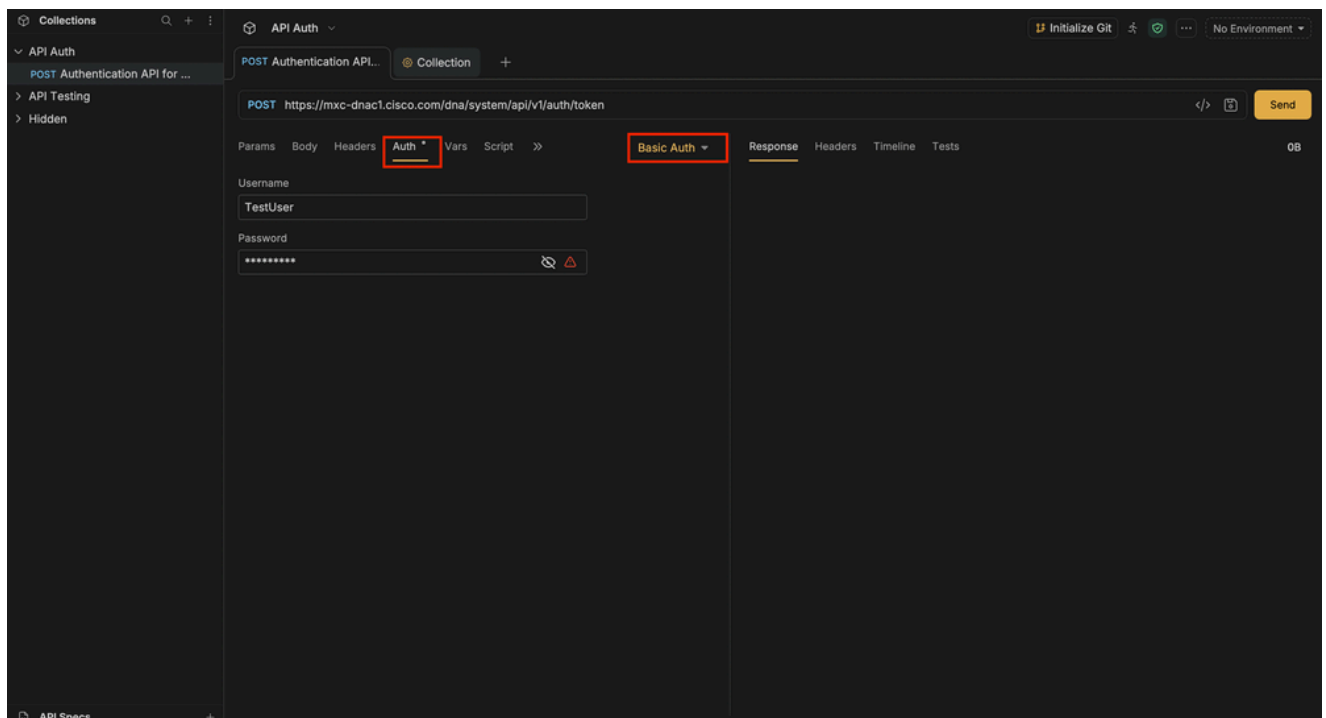
Criando uma nova solicitação no Bruno

3. Insira a URL para a API de autenticação.



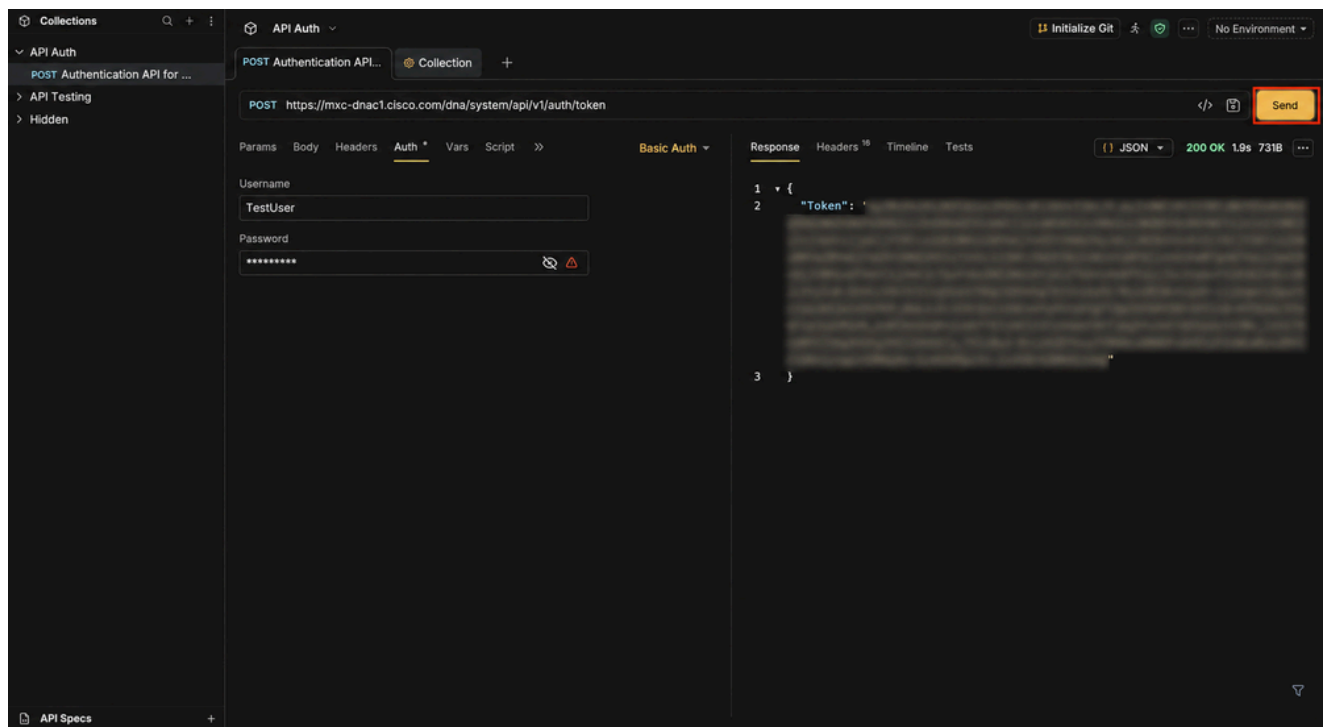
Criando uma nova solicitação no Bruno

4. Navegue até a guia Auth e selecione Basic Auth.



Autenticação Básica para API de Autenticação

5. Insira suas credenciais do Catalyst Center SUPER ADMIN.
6. Clique em Enviar.



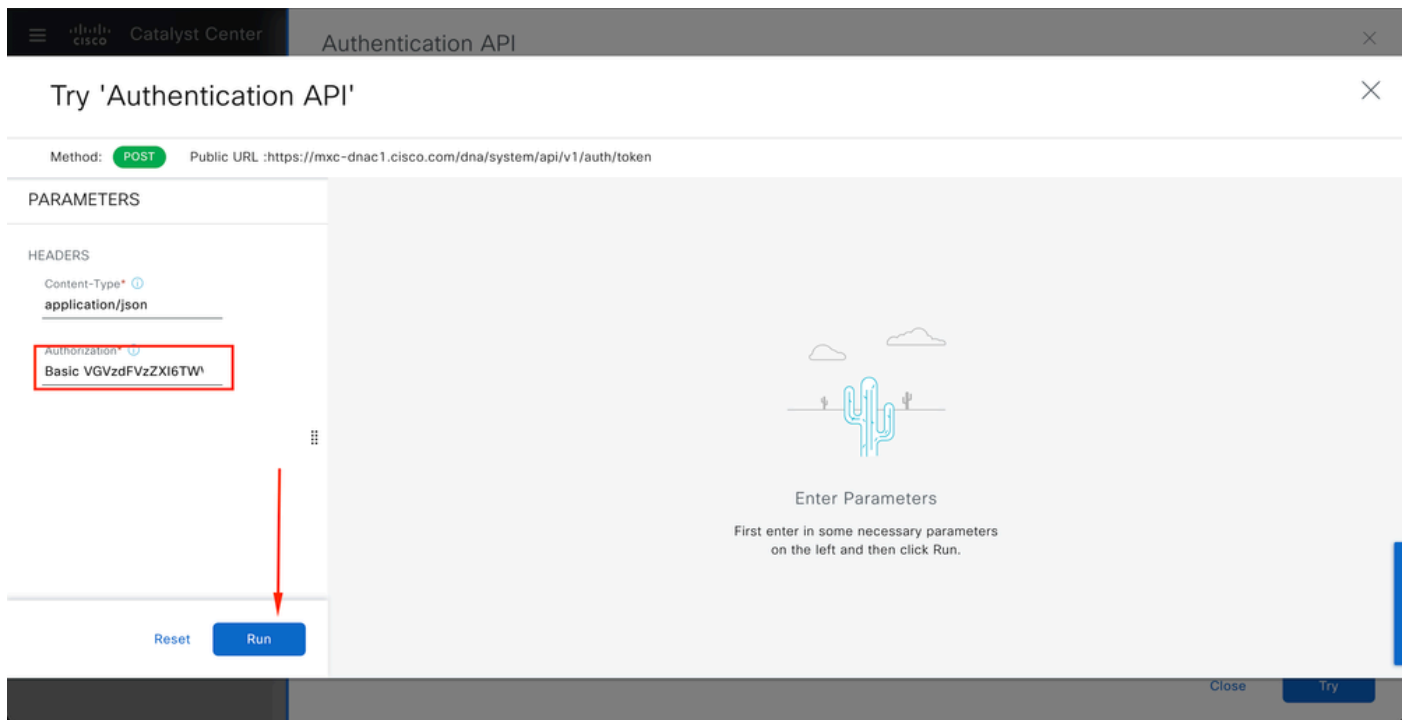
Execução da API de autenticação

O servidor retorna um status 200 OK e um payload JSON contendo o Token.

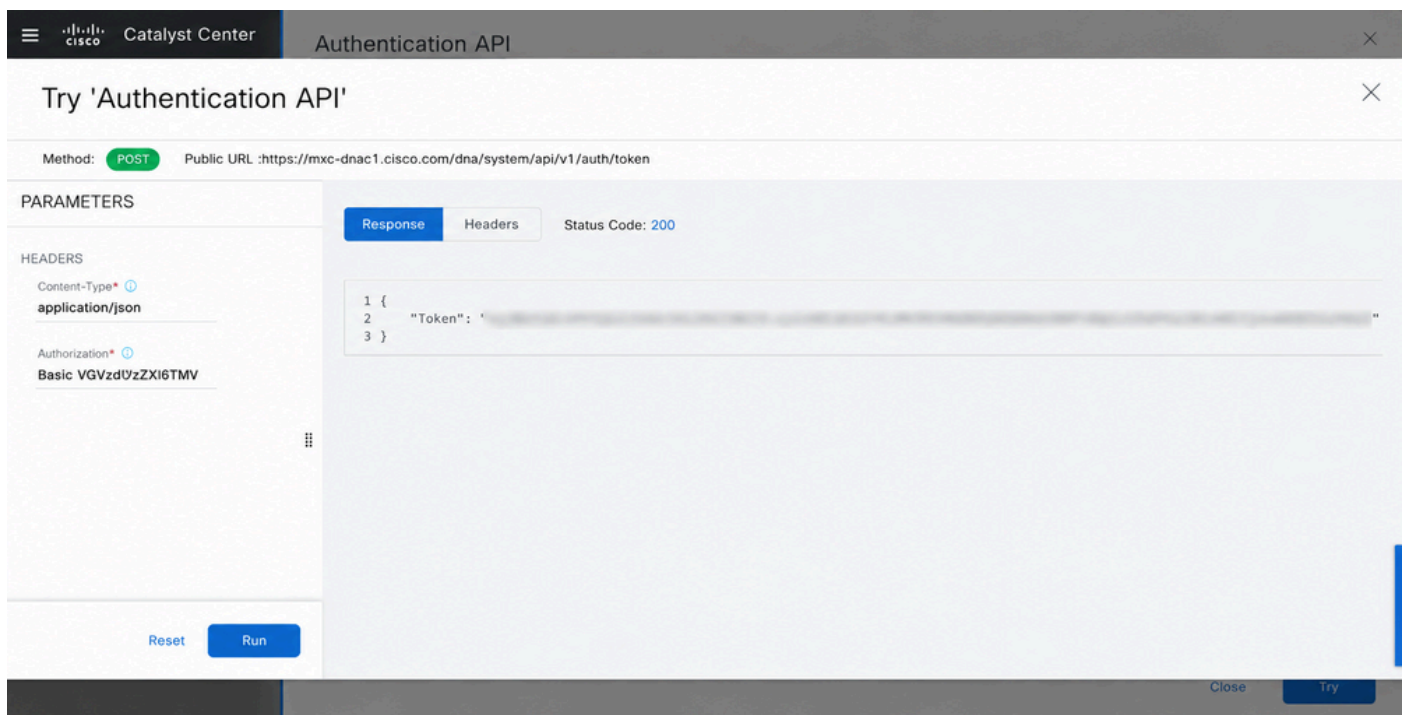
Alternative (Terminal): You can also generate the token by base64-encoding your `username:password` and

```
[redacted] ~ % echo -n 'TestUser:VGvzdFVzZXI6VGvzdDEyMyEhIQ==' | base64
VGvzdFVzZXI6VGvzdDEyMyEhIQ==
```

Obtendo a codificação base64 para o nome de usuário e a senha



Inserindo Autenticação Básica com nome de usuário e senha base64 codificados



Obtendo o token de autenticação da GUI do Catalyst Center.

Neste ponto, você tem o Token de Autenticação, na resposta Bruno ou na resposta do Catalyst Center Developer Toolkit.

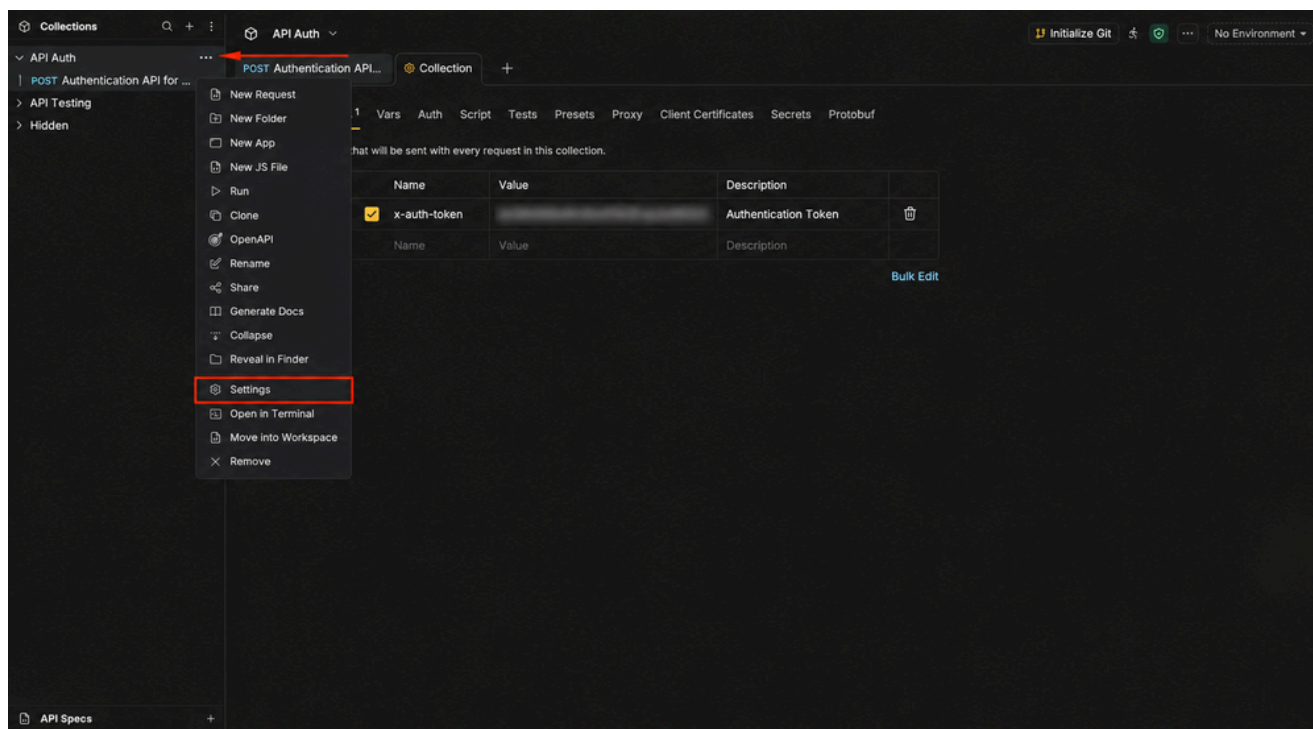
Agora adicione o token como um Cabeçalho nas configurações da coleção no Bruno, isso facilita qualquer outra solicitação criada dentro da coleção para usar o token, de modo que você não

tenha necessidade de inserir o token manualmente como um cabeçalho para cada solicitação.

3. Defina um Cabeçalho Global no Bruno

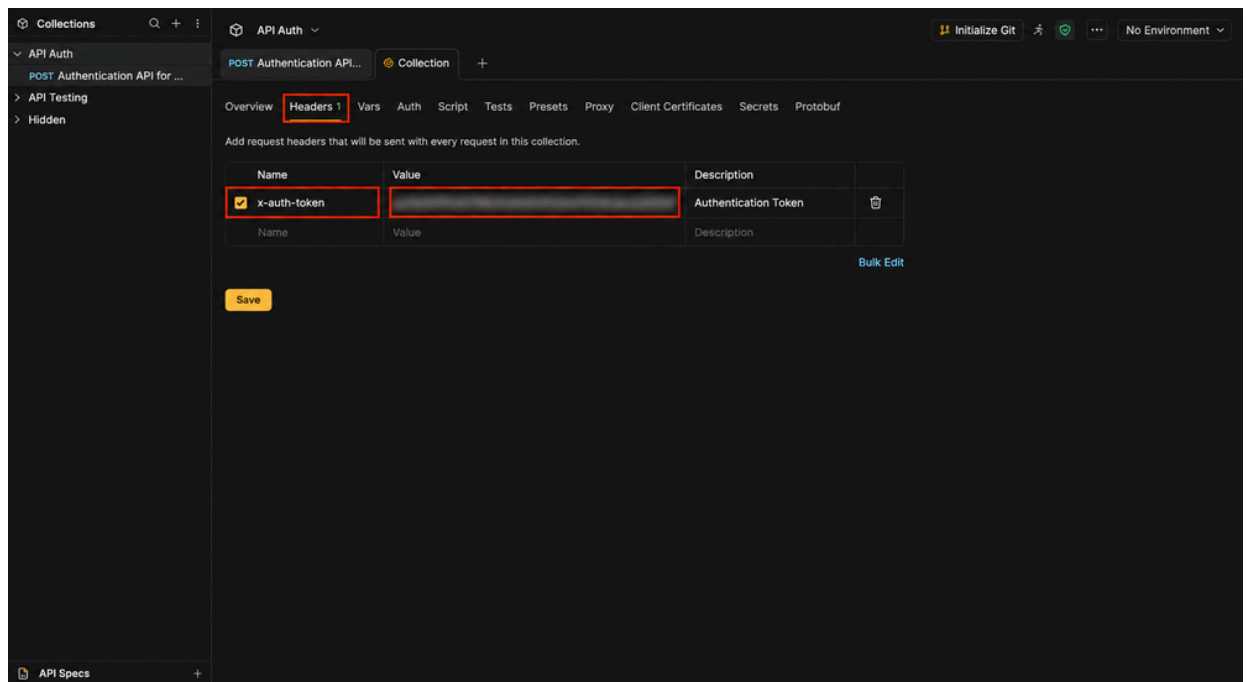
Para evitar a reautenticação para cada solicitação, defina o token como um cabeçalho em nível de coleção:

1. Clique com o botão direito do mouse em sua coleção e selecione Configurações.



Configurações da Coleção

2. Vá até a guia Cabeçalhos.
3. Adicionar um novo cabeçalho:
 - Nome: x-auth-token
 - Valor: [Cole seu token aqui]



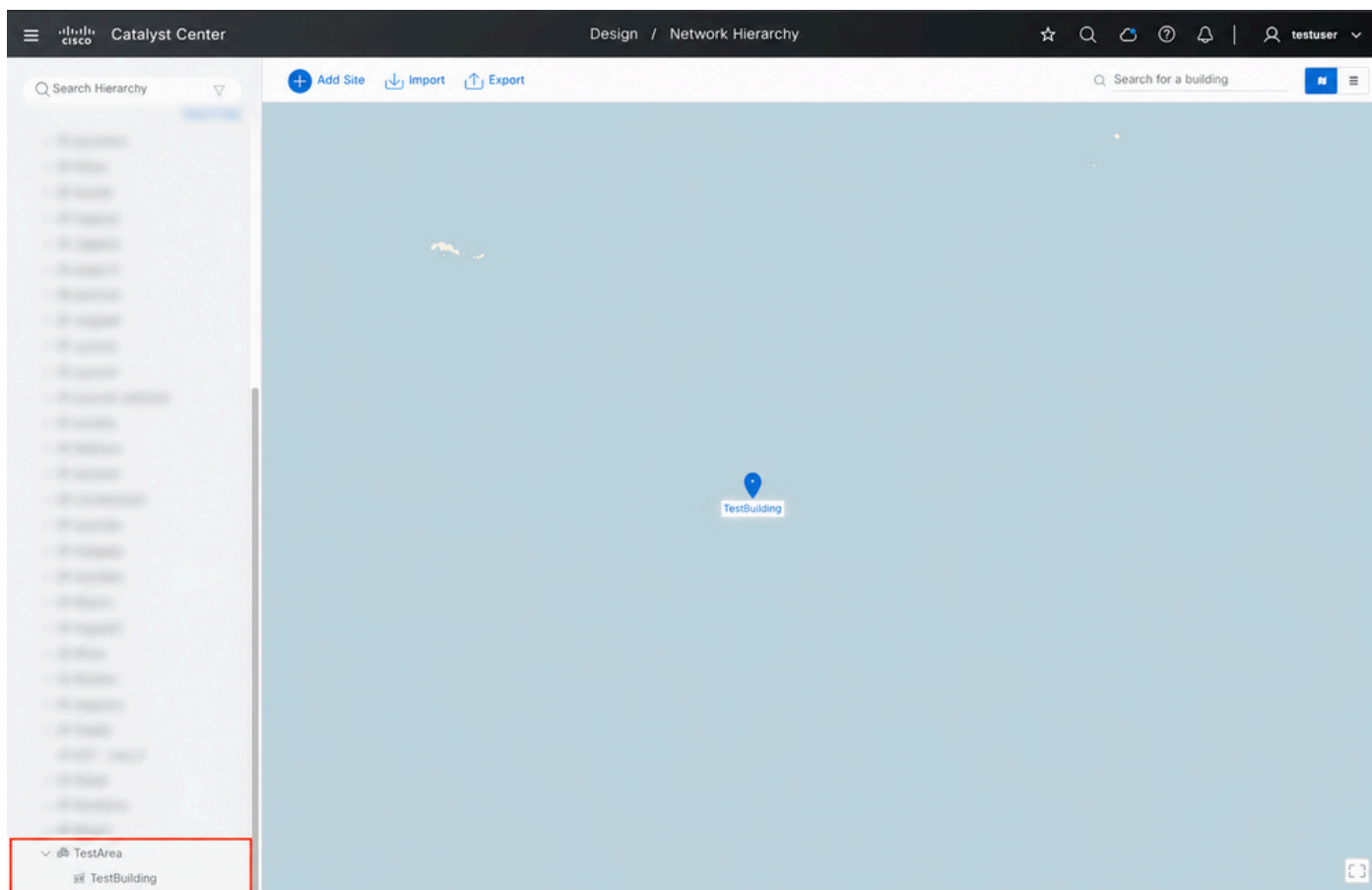
Inserindo cabeçalhos globais para a Coleção de API no Bruno.

4. Click Save.

Neste ponto, você gerou o token de autenticação para fazer qualquer outra solicitação de API do Catalyst Center.

Exemplo 1: Criar um andar do site

Este exemplo demonstra como criar um andar dentro da hierarquia de rede (Global > TestArea > TestBuilding).

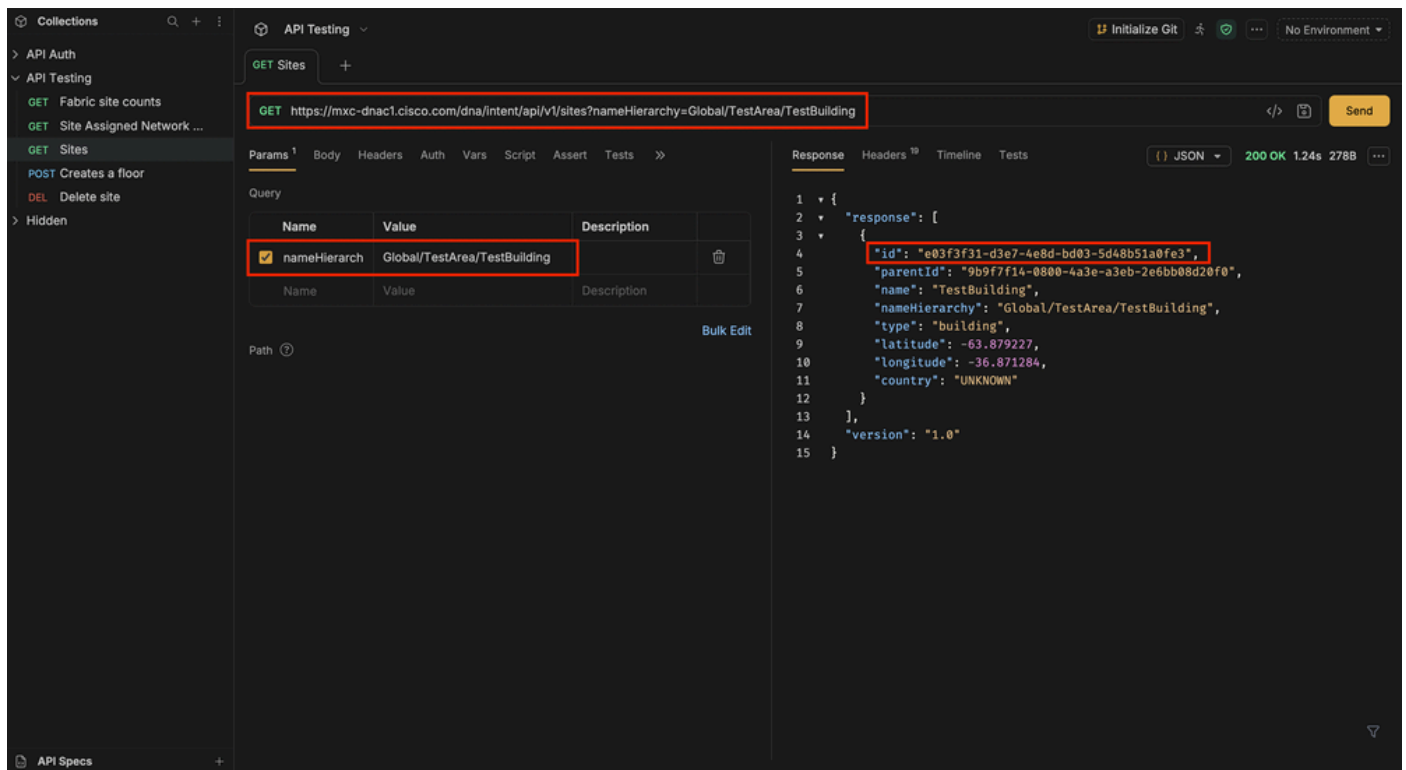


Hierarquia de rede

Passo 1: Obter a ID do Pai

Use a API GET `/dna/intent/api/v1/sites` para localizar a ID de TestBuilding.

- Parâmetro de consulta: `nameHierarchy=Global/TestArea/TestBuilding`
- Copiar o ID da resposta; esta é sua `parentId`.



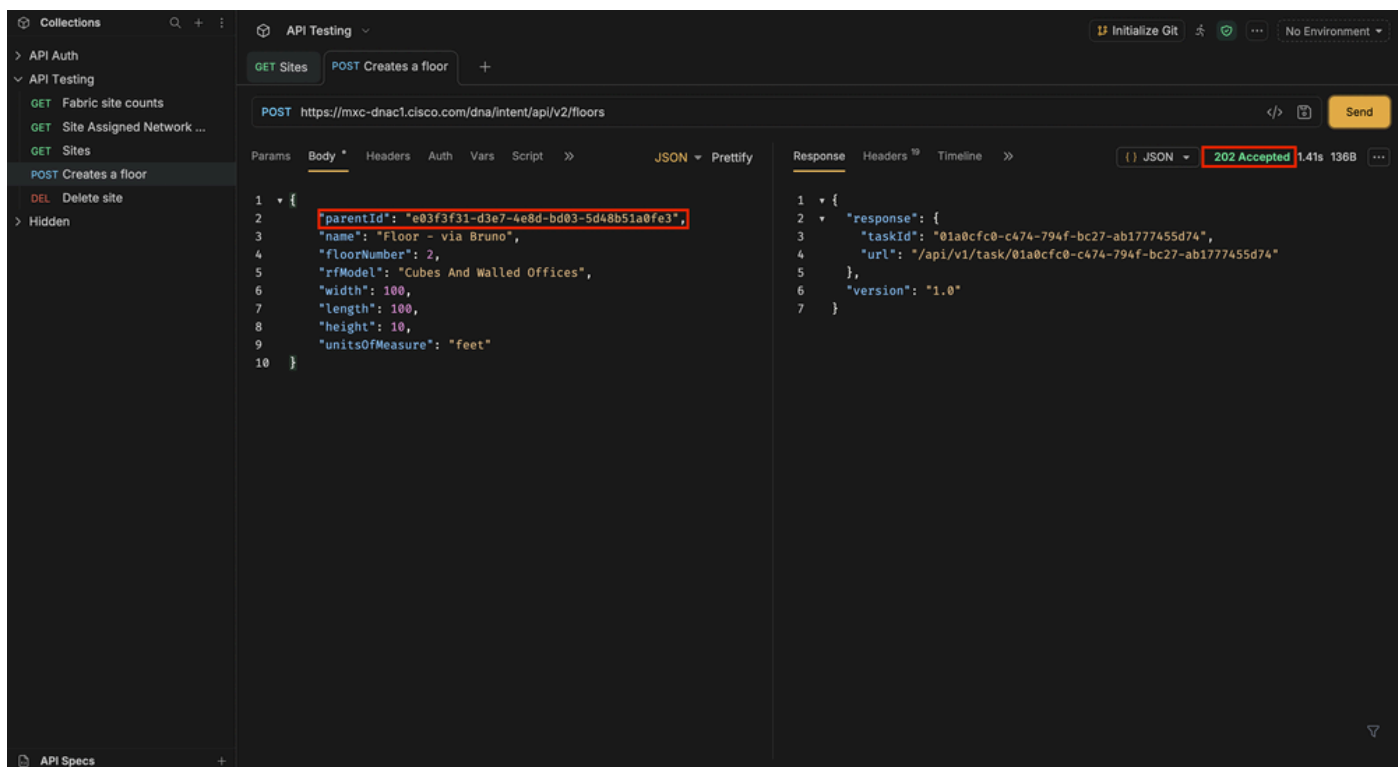
Solicitação GET para /dna/intent/api/v1/sites?nameHierarchy=Global/TestArea/TestBuilding

Passo 2: Criar a tribuna

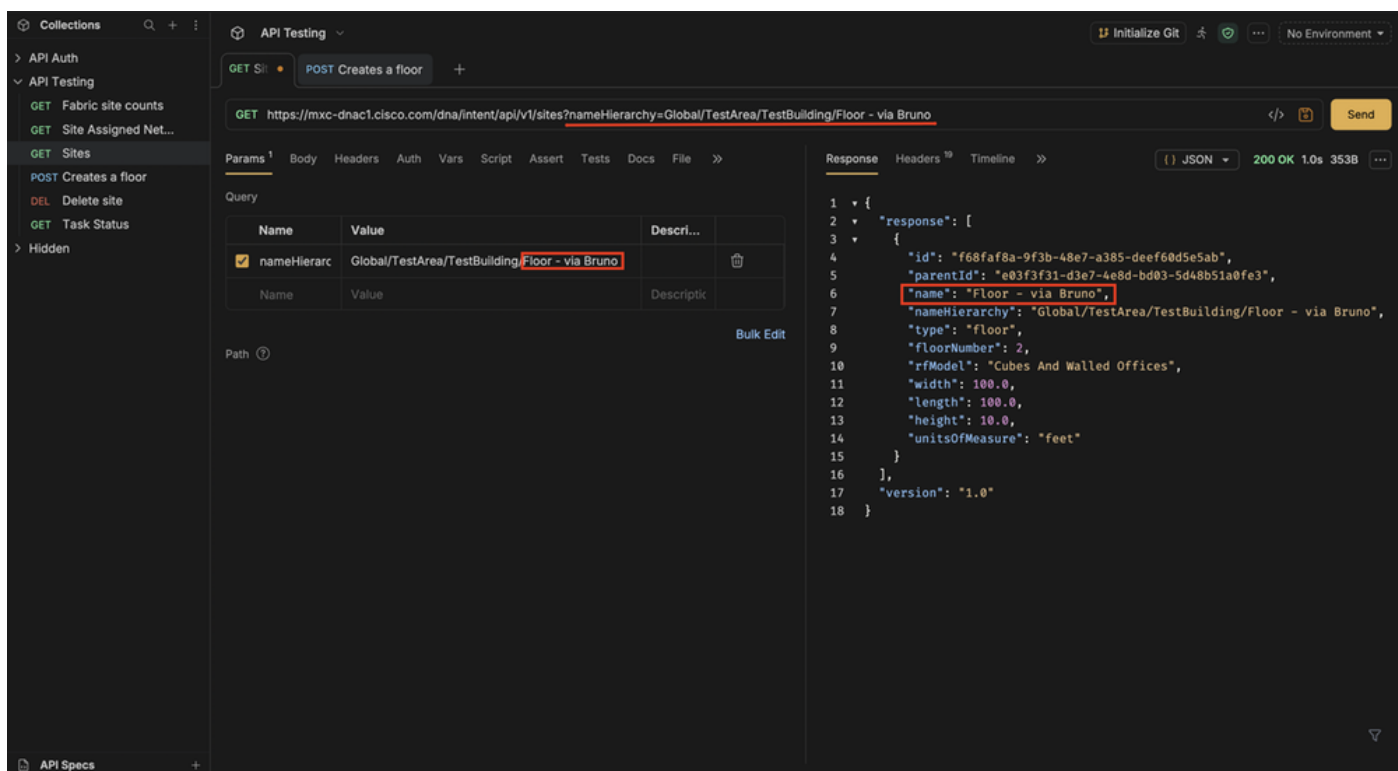
1. Crie uma solicitação de POST para /dna/intent/api/v2/floor.
2. Na guia Corpo (JSON), informe os parâmetros necessários:

```
{
  "name": "Floor 1",
  "parentId": "your-building-id-here",
  "floorNumber": 1,
  "rfModel": "Cubic-Office",
  "width": 100,
  "length": 100,
  "height": 10,
  "unitsOfMeasure": "feet"
}
```

3. Clique em Enviar. Uma resposta bem-sucedida confirma a criação da tribuna.



Solicitação POST para criar a tribuna



API GET para validar a criação de chão de fábrica através de uma solicitação POST.

Finalmente, para validar se a tribuna foi criada com êxito, você pode executar a API anterior e modificar a consulta para corresponder ao caminho hierárquico completo, como mostrado acima.

Exemplo 2: Recuperar informações de interface

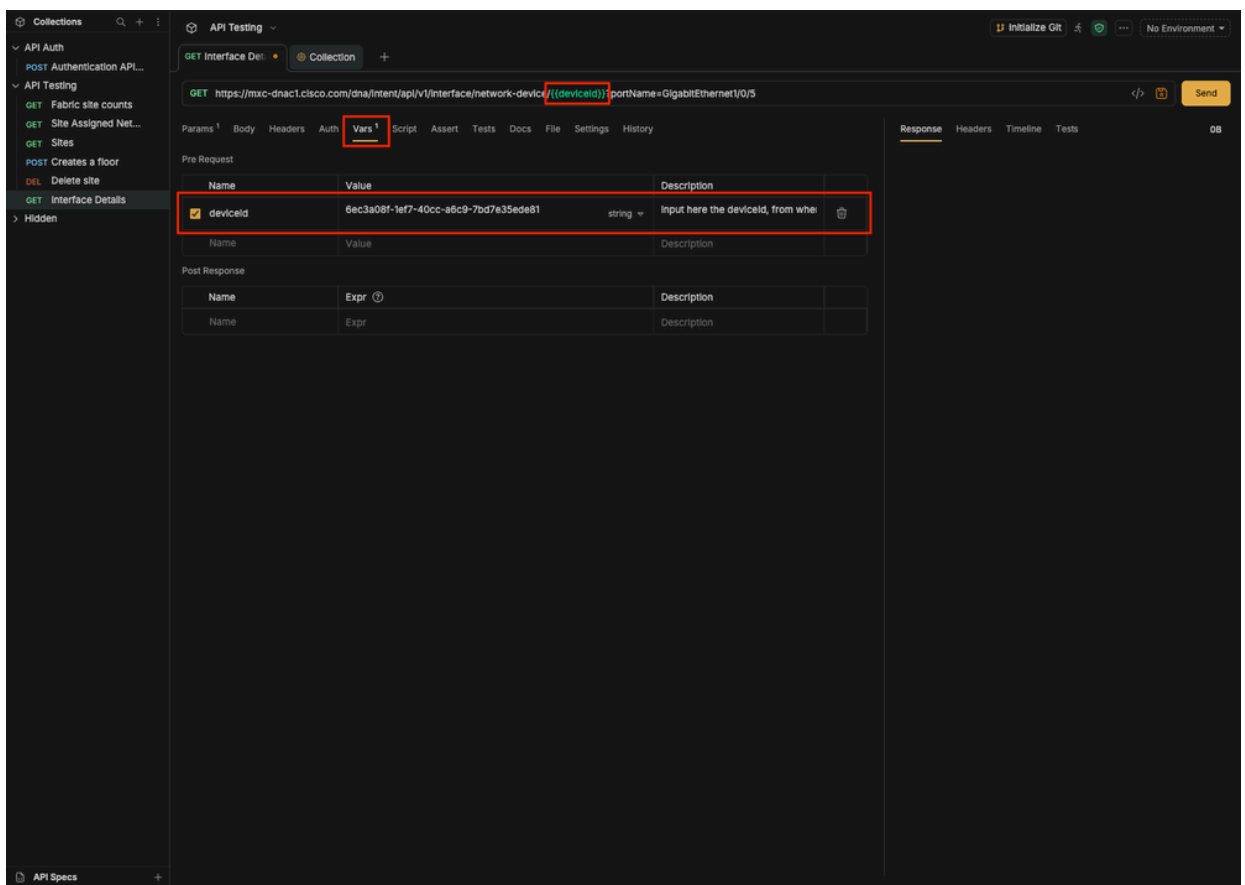
Este exemplo mostra como obter o status de uma interface específica em um dispositivo de rede.

Passo 1: Identificar a ID do dispositivo

Na GUI do Catalyst Center, vá para Inventory, clique no seu dispositivo e veja o URL no seu navegador. O último segmento é a ID do dispositivo.

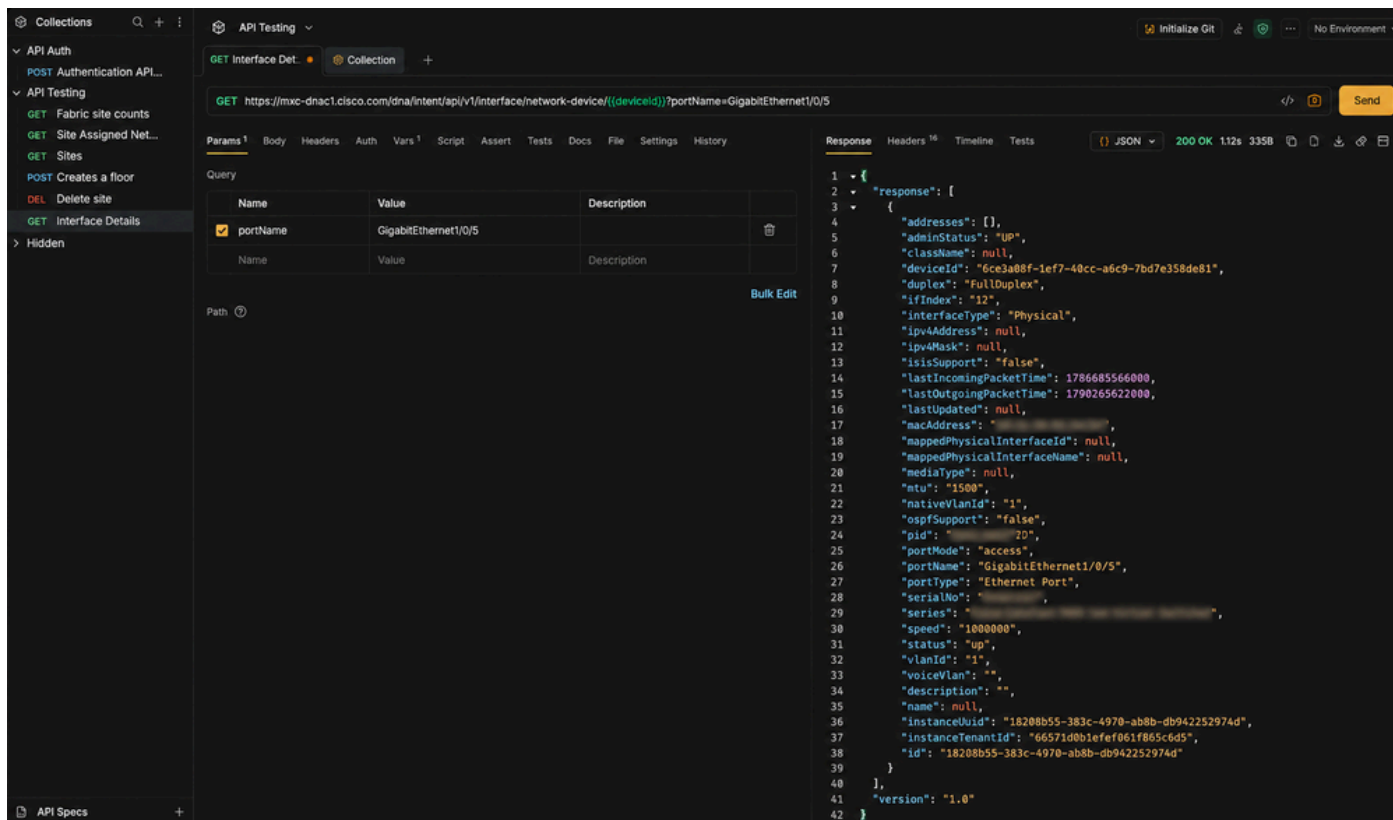
Passo 2: Crie uma solicitação parametrizada no Bruno

1. Crie uma solicitação GET para `/dna/intent/api/v1/interface/network-device/{{deviceid}}`.
2. Na guia Vars da solicitação, adicione uma variável chamada `deviceid` e cole sua ID.
3. Para filtrar uma porta específica, adicione um Parâmetro de consulta:
 - Chave: `nome`
 - Valor: `GigabitEthernet1/0/5`



Criando variáveis no Bruno

4. Clique em Enviar.



The screenshot shows the Postman API client interface. On the left, the 'Collections' panel is open, showing a collection named 'API Testing' with a sub-collection 'GET Interface Details'. The main area displays a GET request to the URL `https://mxc-dnac1.cisco.com/dna/intent/api/v1/interface/network-device/{{deviceId}}?portName=GigabitEthernet1/0/5`. The 'Params' tab is active, showing a single parameter: `portName` with the value `GigabitEthernet1/0/5`. The 'Response' tab shows the JSON response, which is a 200 OK status. The response body is a JSON object containing various interface attributes, including `addresses`, `adminStatus`, `className`, `deviceId`, `duplex`, `ifIndex`, `interfaceType`, `ipv4Address`, `ipv4Mask`, `isisSupport`, `lastIncomingPacketTime`, `lastOutgoingPacketTime`, `lastUpdated`, `macAddress`, `mappedPhysicalInterfaceId`, `mappedPhysicalInterfaceName`, `mediaType`, `mtu`, `nativeVlanId`, `ospfSupport`, `pid`, `portMode`, `portName`, `portType`, `serialNo`, `series`, `speed`, `status`, `vlanId`, `voiceVlan`, `description`, `name`, `instanceBuildId`, `instanceTenantId`, and `id`.

Name	Value	Description
portName	GigabitEthernet1/0/5	

```
1 {
2   "response": [
3     {
4       "addresses": [],
5       "adminStatus": "UP",
6       "className": null,
7       "deviceId": "6ce3a08f-1ef7-40cc-a6c9-7bd7e358de81",
8       "duplex": "FullDuplex",
9       "ifIndex": "12",
10      "interfaceType": "Physical",
11      "ipv4Address": null,
12      "ipv4Mask": null,
13      "isisSupport": "false",
14      "lastIncomingPacketTime": 1786685566000,
15      "lastOutgoingPacketTime": 1790265622000,
16      "lastUpdated": null,
17      "macAddress": null,
18      "mappedPhysicalInterfaceId": null,
19      "mappedPhysicalInterfaceName": null,
20      "mediaType": null,
21      "mtu": "1500",
22      "nativeVlanId": "1",
23      "ospfSupport": "false",
24      "pid": "20",
25      "portMode": "access",
26      "portName": "GigabitEthernet1/0/5",
27      "portType": "Ethernet Port",
28      "serialNo": null,
29      "series": null,
30      "speed": "1000000",
31      "status": "up",
32      "vlanId": "1",
33      "voiceVlan": null,
34      "description": null,
35      "name": null,
36      "instanceBuildId": "18208b55-383c-4970-ab8b-db942252974d",
37      "instanceTenantId": "6657140b1efef061f865c6d5",
38      "id": "18208b55-383c-4970-ab8b-db942252974d"
39    }
40  ],
41  "version": "1.0"
42 }
```

Solicitação GET aplicando uma consulta

Conclusão

As APIs ascendentes do Catalyst Center fornecem flexibilidade poderosa para administradores de rede. Ao fazer a transição do gerenciamento baseado em GUI para fluxos de trabalho orientados por API, você pode automatizar tarefas repetitivas, garantir a consistência da configuração e integrar a inteligência de rede em seu ecossistema de TI mais amplo.

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.