

# TESTE DE ADMISSÃO

**Função:** Analista de Qualidade - Integrações

## 1. Instruções

- Leia atentamente o material abaixo antes de iniciar esse teste.
- Sua tarefa será realizar testes na API pública disponibilizada, e produzir cenários de testes em Gherkin com base nos testes executados.
- Para a execução das requisições, poderá ser utilizado o Postman.
- Você poderá fazer observações referente aos requisitos caso necessário ao final deste documento.

## 2. API

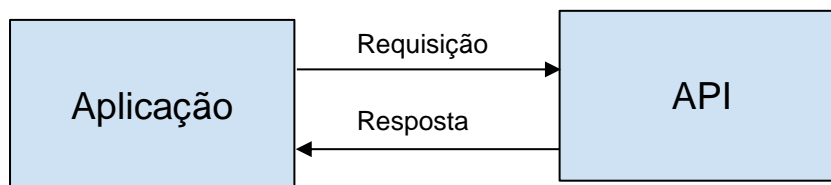
Através do link abaixo, está documentada uma série de API's disponibilizadas publicamente para consultar dados de serviços brasileiros.

Para os exercícios vamos utilizar os métodos CEP V2, CPTEC - Buscar localidades e CPTEC - Previsão oceânica.

<https://brasilapi.com.br/docs>

## 3. Arquitetura

### Chamada de uma API



---

A chamada de uma API tem como objetivo obter uma resposta com base nos parâmetros e regras estipuladas para o método que será requisitado.

## 4. Exercícios

**E1:** Ao utilizar a ação de importar, na requisição abaixo, há um erro em sua execução. Com base na documentação, quais correções devem ser feitas para a requisição ser executada com sucesso.

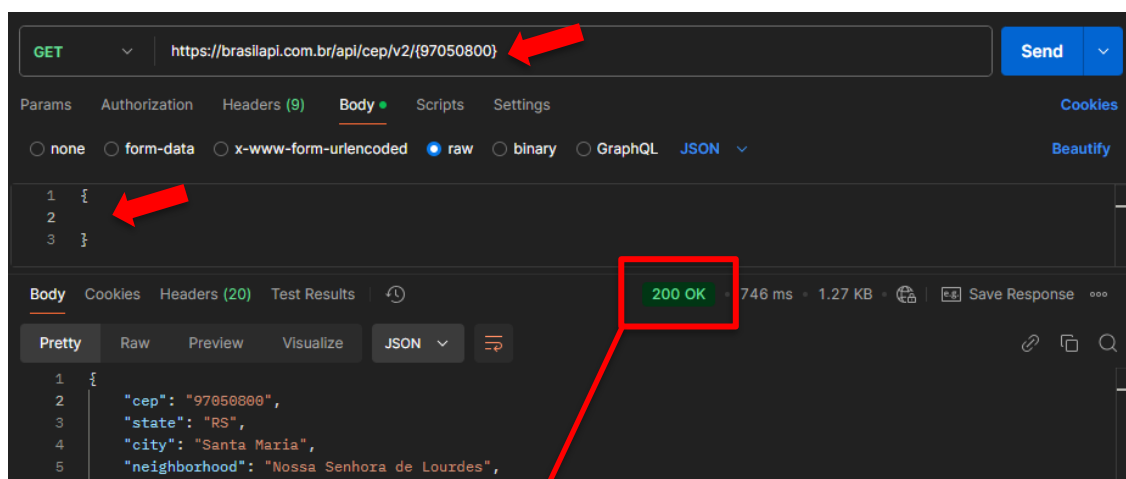
```
curl --location --globoff --request GET 'https://brasilapi.com.br/api/cep/v2/{cep}' \
--header 'Content-Type: application/json' \
--data '{
  "cep": "97050800"
}'
```

**R1:**

Realizada a Importação do código cURL e enviado, o erro apresentado é este abaixo, que estou incluindo no documento apenas para esclarecimento do cenário realizado:

```
{
  "name": "CepPromiseError",
  "message": "CEP deve conter exatamente 8 caracteres.",
  "type": "validation_error",
  "errors": [
    {
      "message": "CEP informado possui mais do que 8 caracteres.",
      "service": "cep_validation"
    }
  ]
}
```

Para resolver o problema, pesquisando entendi o valor do CEP poderia ser utilizado diretamente na URL, então incluí o valor do CEP onde estava “cep” e removi o conteúdo do body.



Desta forma tive a resposta 200 OK. Considerando isto, arrisco dizer que o código cURL poderia ser desta forma:

```
curl --location --globoff --request GET 'https://brasilapi.com.br/api/cep/v2/97050800' \
--header 'Content-Type: application/json'
```

**E2:** Com base nas respostas que a API de consultar dados do endereço pelo CEP retornou, foi possível identificar o que estava sendo enviado incorretamente?

Qual sua sugestão para tornar mais amigável ao usuário as respostas apresentadas?

Que informações não estão claras na documentação técnica da API que são relevantes estarem explícitas?

**R2:**

Penso que não, pois a mensagem apresentada foi "CEP deve conter exatamente 8 caracteres." E considerando que o CEP estava sendo informado – no local equivocado – então não fica claro para quem está lendo este resultado sobre qual é de fato o problema apresentado.

Eu consideraria algo mais intuitivo, que dê fato desse uma direção para a solução do problema, e não um simples “**Error**”, mas aí depende das possibilidades de tratamento para a mensagem apresentada, para facilitar a experiência do usuário. Talvez um return mais voltado para algo do tipo “Por favor, verifique se o valor do CEP foi informado no local correto e no formato correto.” ou ainda “Aparentemente o valor do CEP não foi informado corretamente na URL”, talvez até “Verifique se para a requisição utilizada está correto o preenchimento do respectivo body”.

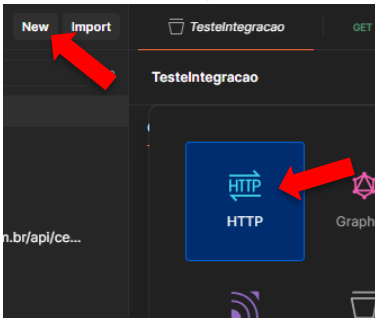
Com base nos testes de API que já trabalhei, sinto falta de um descritivo de todas as mensagens possíveis de retorno de forma clara para cada erro possível considerando os atributos dos métodos utilizados.

**E3:** Montar, consultar e evidenciar uma chamada para o método CEP V2 utilizando um CEP válido.

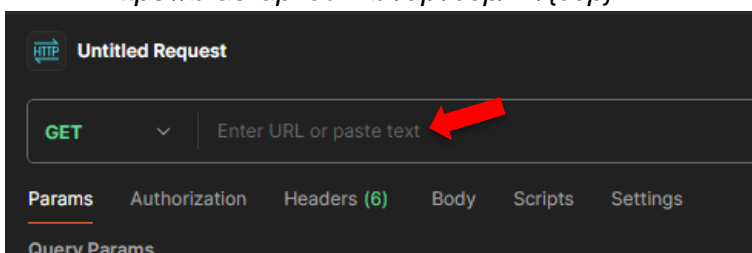
**R3:**

Montando o método CEP V2 considerando que o Postman foi previamente instalado e está aberto na collection desejada.

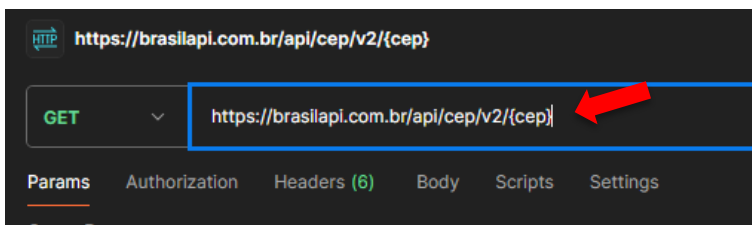
Passo 1) Clicar em New e após em HTTP, conforme destacado:



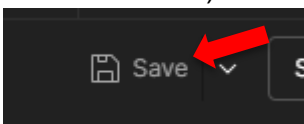
Passo 2) Nos campos apresentados, informar a URL desejada, no caso é <https://brasilapi.com.br/api/cep/v2/{cep}>



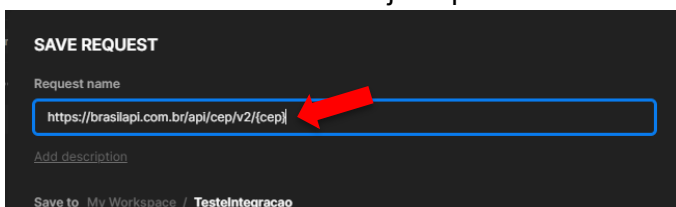
Ficando desta forma:



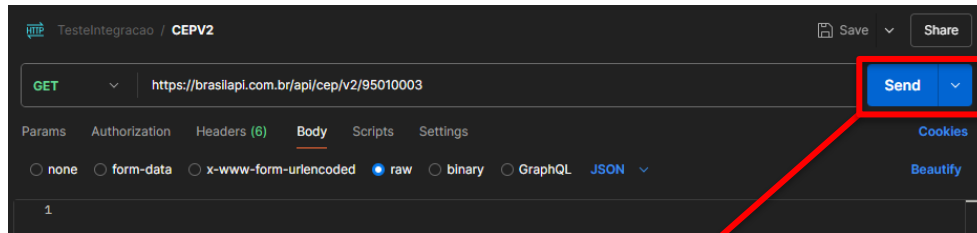
Passo 3) Clicar em "Save":



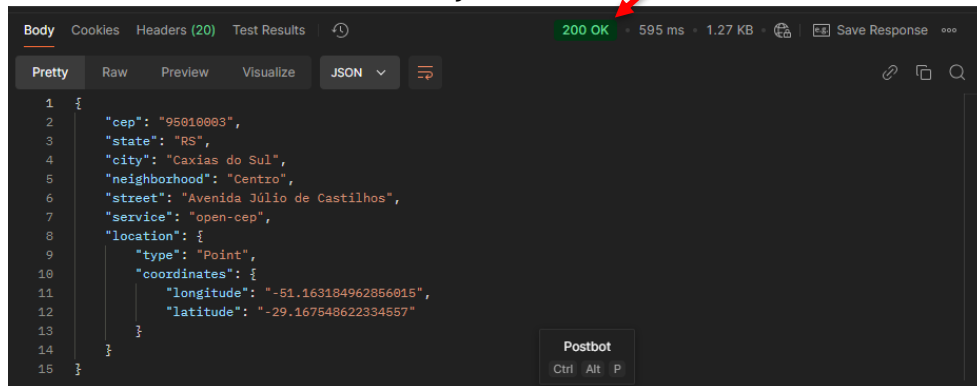
Informar o nome desejado para o Método e novamente "Save":



Consultando com um CEP válido com método salvo CEPV2, no caso utilizado o CEP 95010003, informado no final da URL.



Ao ser realizado o envio (botão “Send”) após alguns segundos, apresentado resultado de sucesso, em conformidade com documentação da API.

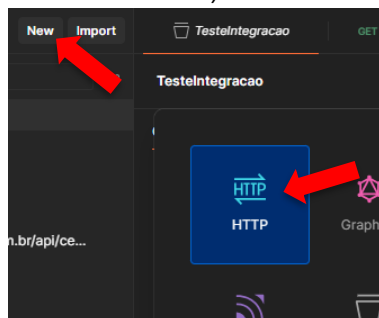


**E4:** Montar, consultar e evidenciar uma chamada para o método Buscar localidade.

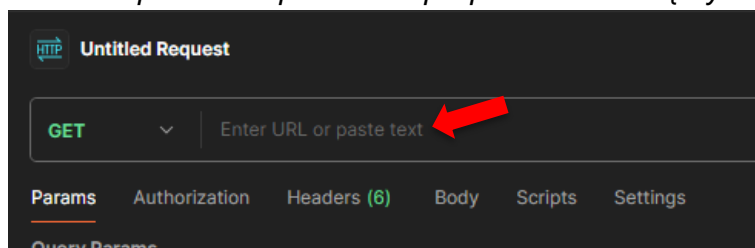
**R4:**

Montando o método Buscar localidades considerando que o Postman foi previamente instalado e está aberto na collection desejada.

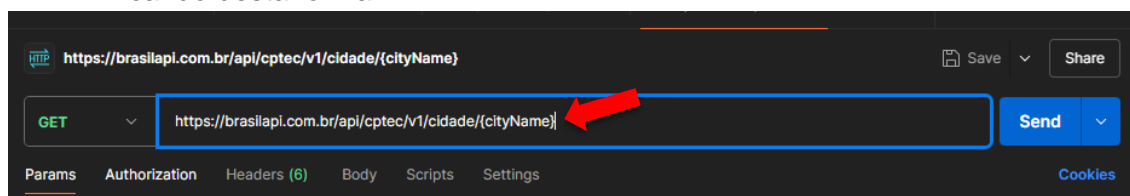
Passo 1) Clicar em New e após em HTTP, conforme destacado:



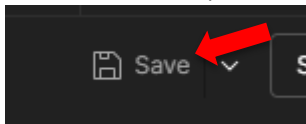
Passo 2) Nos campos apresentados, informar a URL desejada, no caso é *https://brasilapi.com.br/api/cptec/v1/cidade/{cityName}*



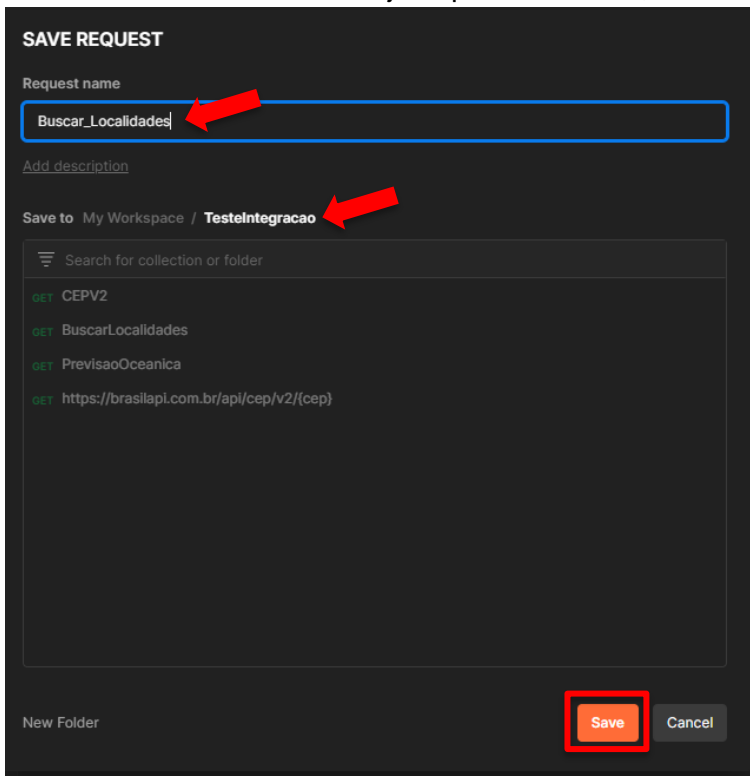
Ficando desta forma:



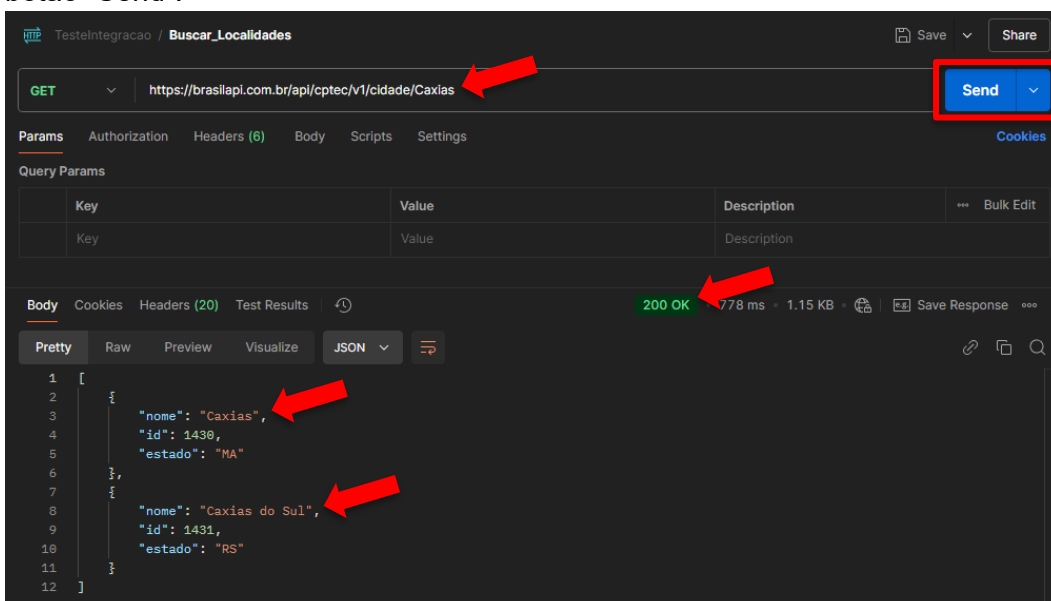
Passo 3) Clicar em “Save”:



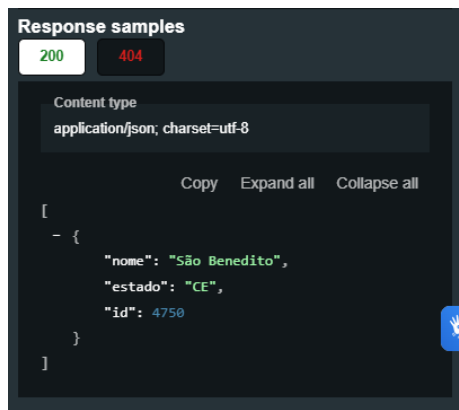
Informar o nome desejado para o método, selecionar o workspace desejado, e novamente “Save”:



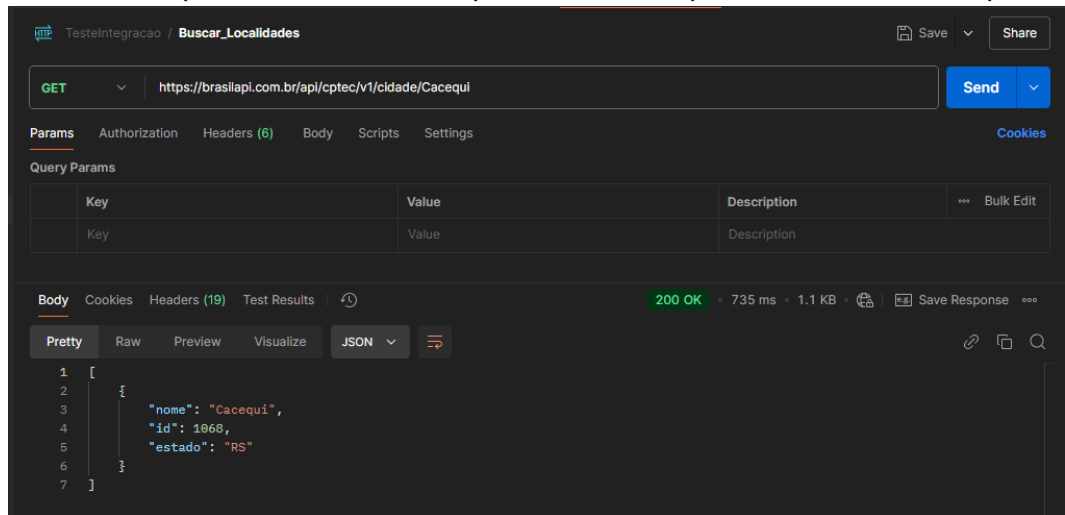
Consultando com o método Buscar Localidade, nesse caso utilizado o valor “Caxias”, informado no final da URL, sabendo-se que desta forma teria mais de uma cidade para ser localizada ao ser utilizado o botão “Send”:



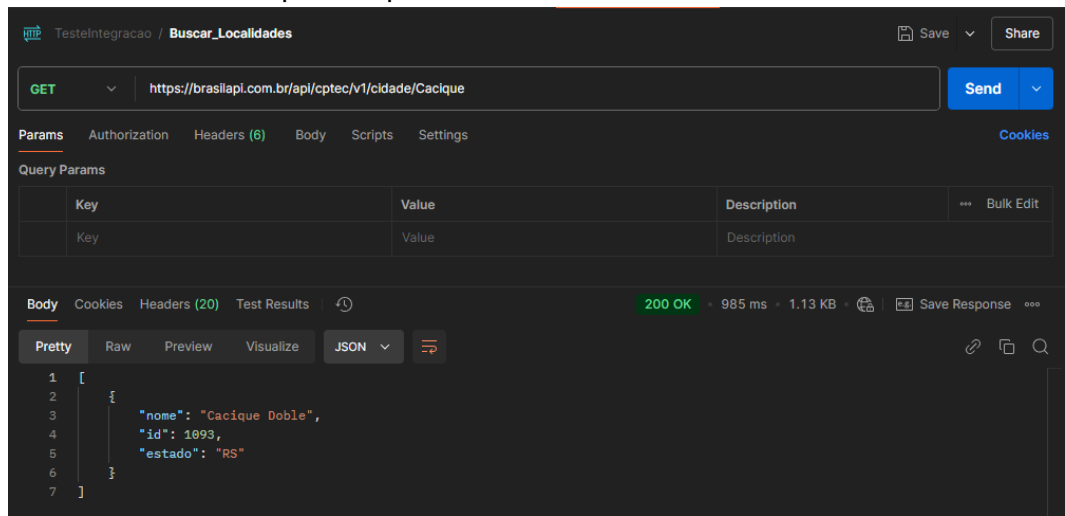
Destaque para o resultado Request Successful (200 OK), de acordo com o exemplo apresentado no site da Brasil API:



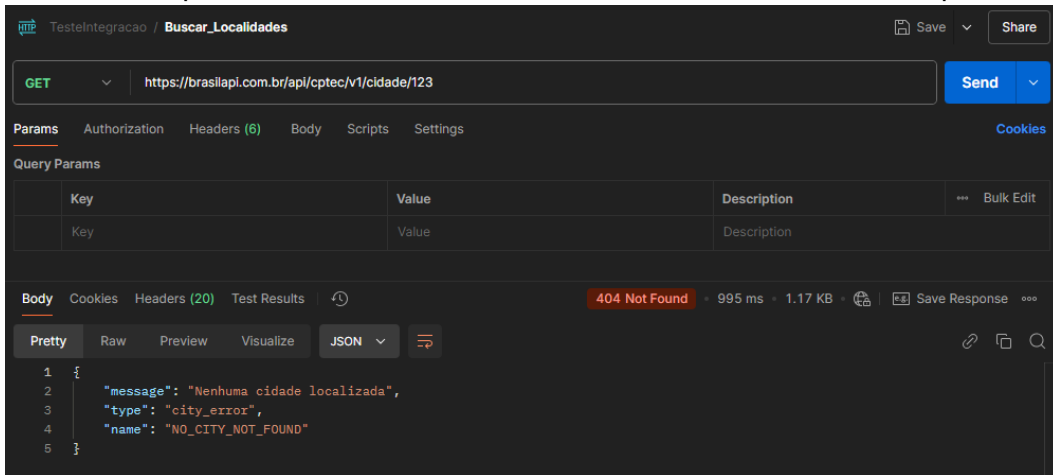
O enunciado deste exercício solicita “uma chamada”, mas ampliando as possibilidades, outros cenários são possíveis, como exemplo, o nome completo de uma cidade, apresentando um único retorno.



Ou um nome parcial que retorne ainda assim uma única cidade:



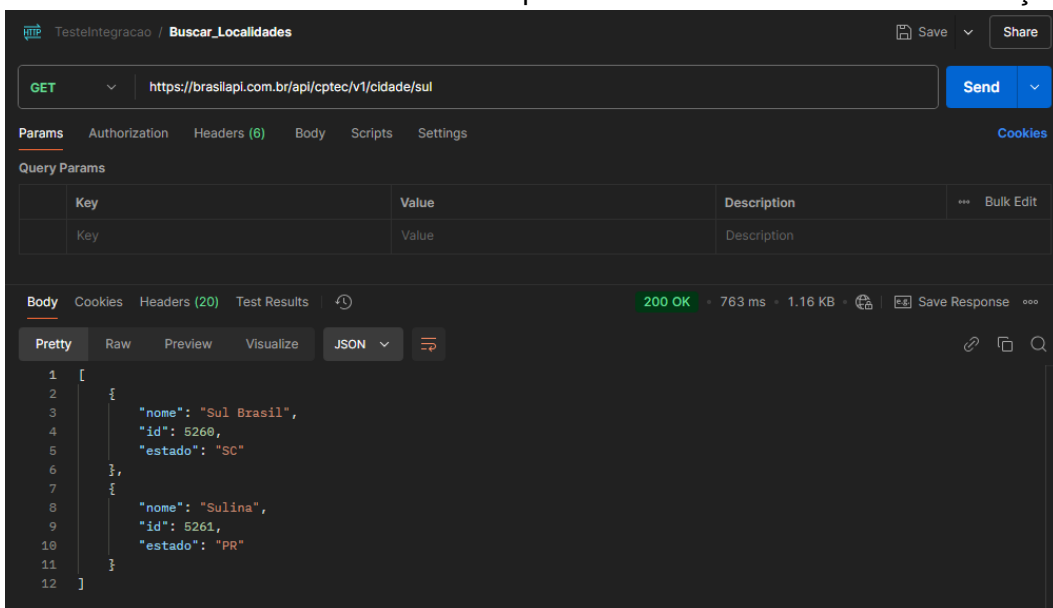
Outra possibilidade ainda é informando valor inválido, de forma que não encontre um resultado.



The screenshot shows a REST client interface with the URL `https://brasilapi.com.br/api/cptec/v1/cidade/123`. The response status is **404 Not Found**. The response body is displayed in JSON format:

```
1 {
2   "message": "Nenhuma cidade localizada",
3   "type": "city_error",
4   "name": "NO_CITY_NOT_FOUND"
5 }
```

Percebeu-se durante a realização dos testes que a consulta é realizada utilizando-se aparentemente o valor do parâmetro como parte inicial do nome da cidade somente. Conforme exemplo abaixo, pesquisando a palavra “sul”, os resultados apresentados não contemplam a cidade de Caxias do Sul, não vi no site da Brasil API um detalhamento que oriente o usuário sobre essa condição da busca.



The screenshot shows a REST client interface with the URL `https://brasilapi.com.br/api/cptec/v1/cidade/sul`. The response status is **200 OK**. The response body is displayed in JSON format:

```
1 [
2   {
3     "nome": "Sul Brasil",
4     "id": 5260,
5     "estado": "SC"
6   },
7   {
8     "nome": "Sulina",
9     "id": 5261,
10    "estado": "PR"
11  }
12 ]
```

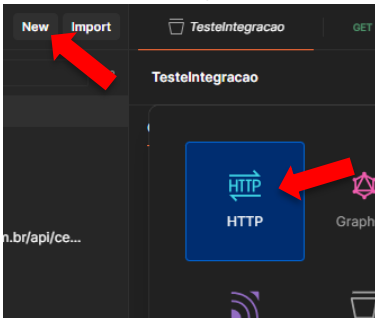


**E5:** Montar, consultar e evidenciar uma chamada para o método Previsão oceânica.

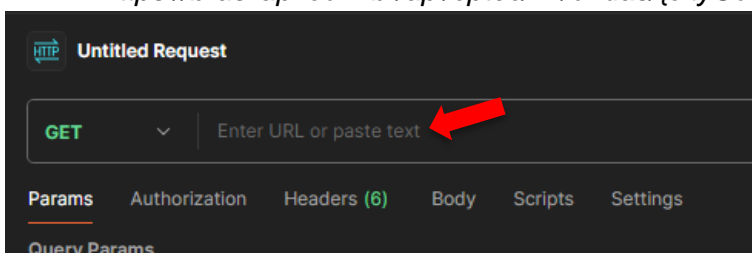
**R5:**

Montando o método de Previsão oceânica considerando que o Postman foi previamente instalado e está aberto na collection desejada.

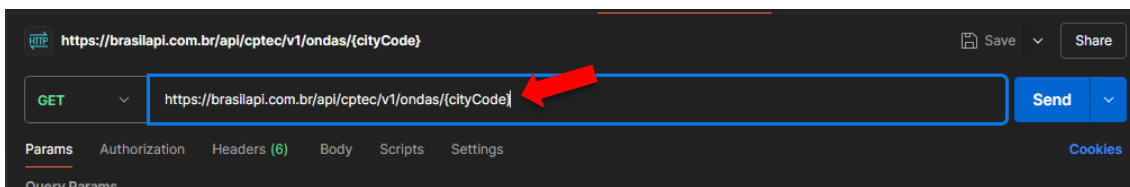
Passo 1) Clicar em New e após em HTTP, conforme destacado:



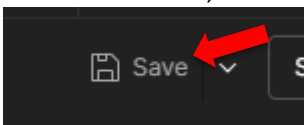
Passo 2) Nos campos apresentados, informar a URL desejada, no caso é `https://brasilapi.com.br/api/cptec/v1/ondas/{cityCode}`



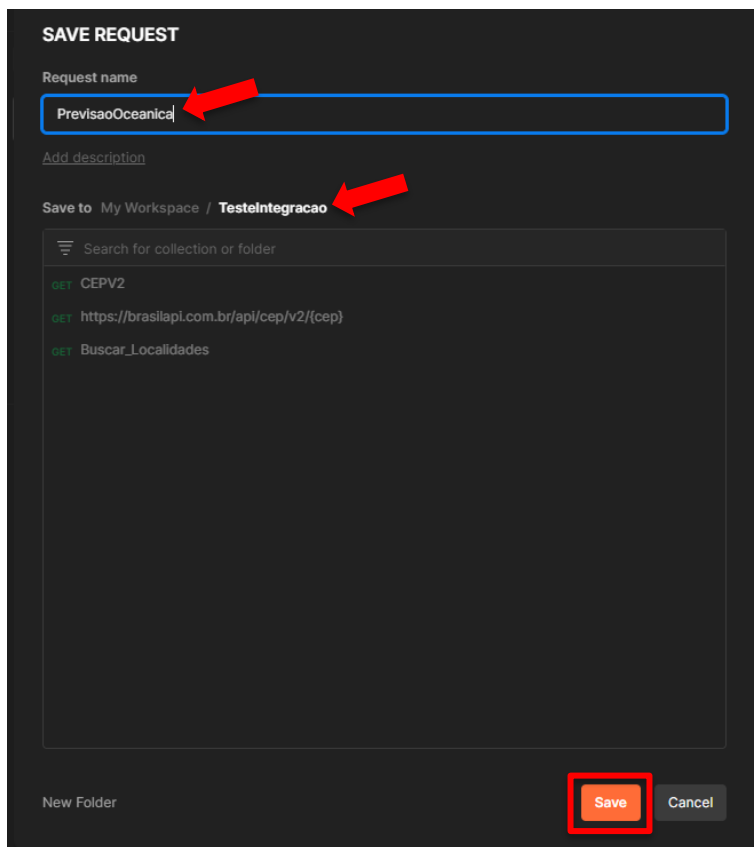
Ficando desta forma:



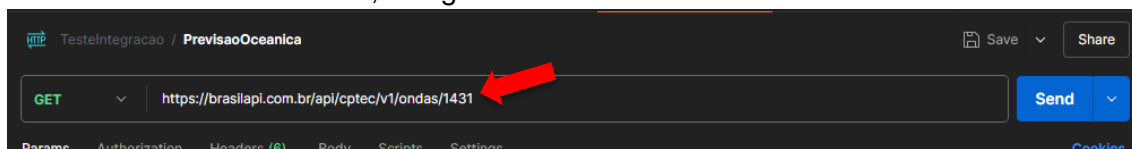
Passo 3) Clicar em “Save”:



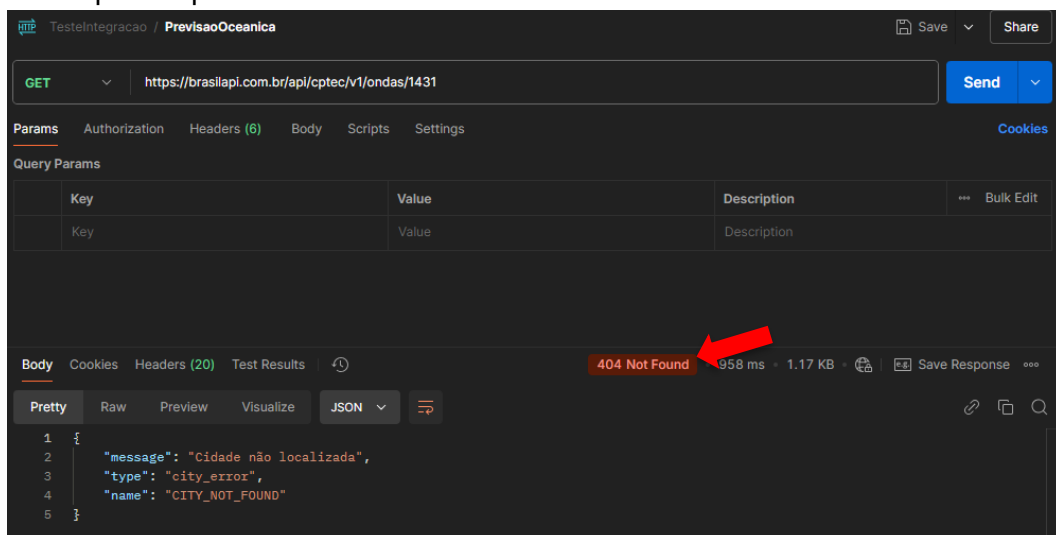
Informar o nome desejado para o método, selecionar o workspace desejado, e novamente “Save”:



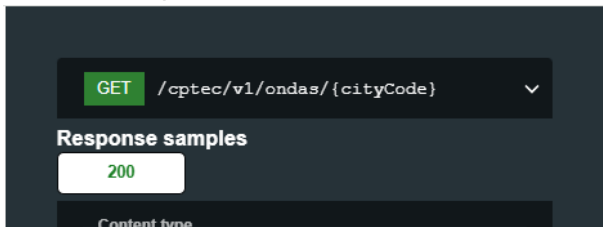
Consultando com o método Previsão Oceânica, nesse caso utilizado o valor “1431”, visto em testes anteriores neste documento, código este informado no final da URL:



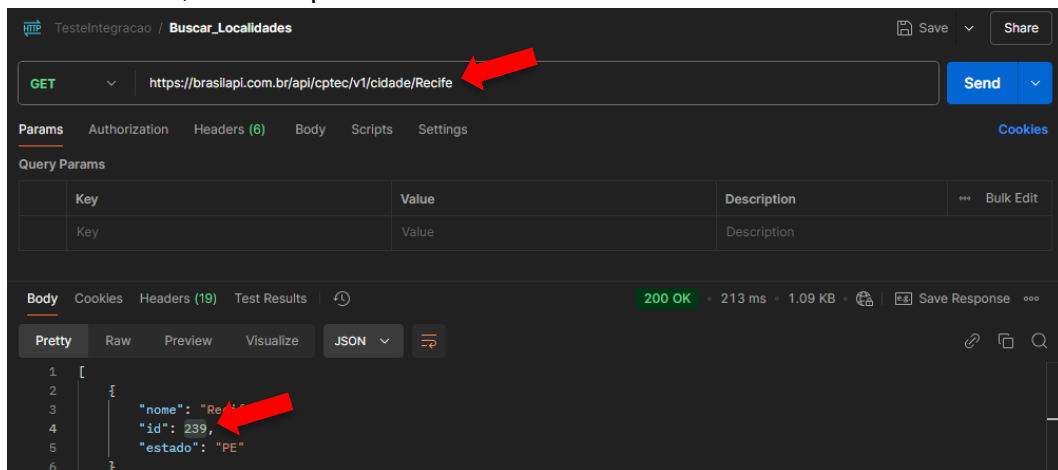
Neste caso não foi localizada a cidade e, por uma dedução do testador, deve ter relação com o fato de que esta cidade não é litorânea. Visto que essa condição não está descrita no site da Brasil API, esta é uma hipótese provável.



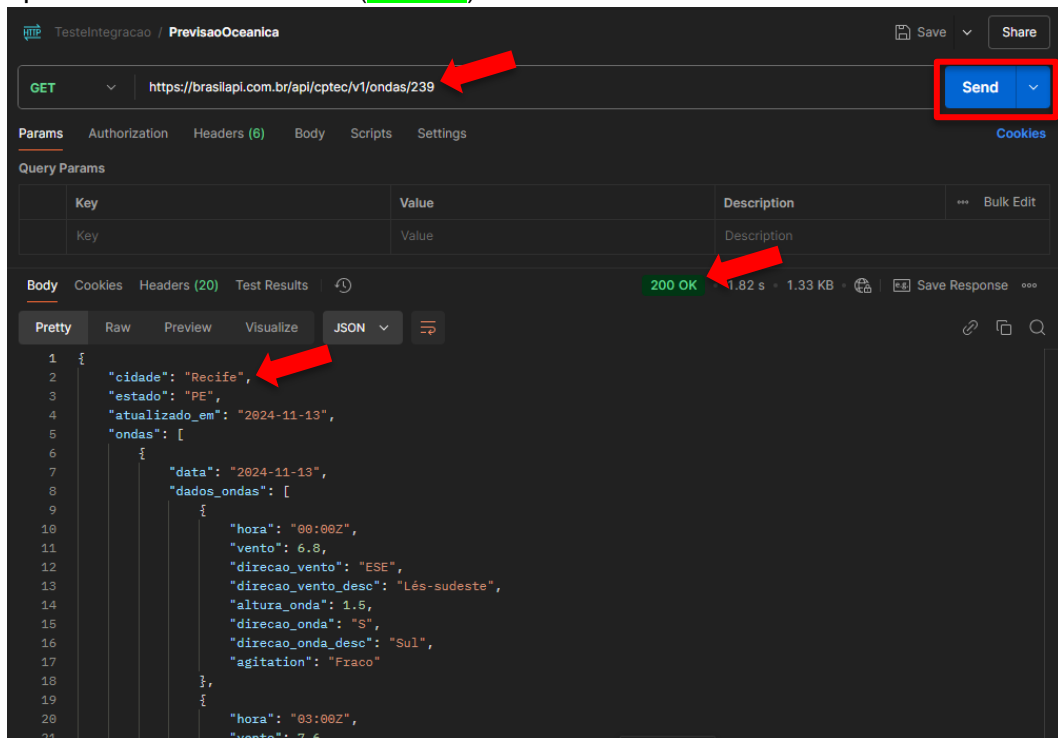
Nesse caso, uma observação, no site da Brasil API não é apresentado um exemplo de mensagem de erro 404 para este método.



Então, voltando para o método de Buscar Localidades, realizada busca utilizando como cidade o valor “Recife”, onde foi possível identificar o id 239.



Utilizando este código 239 no método de Previsão Oceânica, ao ser enviada a requisição, resultado apresentado com sucesso (200 OK).



Dados retornados estão de acordo com o exemplo disponível no site Brasil API:

Response samples

200

Content type  
application/json; charset=utf-8

Copy Expand all Collapse all

```
{
  "cidade": "Rio de Janeiro",
  "estado": "RJ",
  "atualizado_em": "2020-12-27",
  "ondas": [
    - {
      "data": "27-12-2020",
      "dados_ondas": [
        - {
          "vento": 5.2,
          "direcao vento": "E",
```

## 5. Cenários de Testes em Gherkin

Com base nos exercícios realizados, escolher ao menos um cenário positivo e outro negativo e descrevê-los, utilizando a linguagem Gherkin.

*Cenário positivo:*

**Feature:** Consulta com método Buscar Localidades

**Scenario:** Consulta de cidades onde as mesmas palavras tenham vários resultados

**Given** o usuário tem acesso ao método de Buscar Localidades no Postman

**When** o usuário faz uma requisição para a API informando no método o valor de exemplo "Porto Alegre"

**Then** a resposta no Postman deve conter o status "200 OK"

**And** o corpo da resposta deve conter a lista de cidades que contenham as palavras "Porto Alegre" no nome

**And** a lista deve conter a cidade de "Porto Alegre" do estado RS

**And** a lista pode ter outras cidades com estas palavras como parte parcial do nome

*1º Cenário negativo:*

**Feature:** Consulta de Previsão Oceânica

**Scenario:** Consulta de ondas para uma cidade não litorânea

**Given** o usuário tem acesso ao método de Previsão Oceânica no Postman

**And** o usuário quer consultar as ondas para uma cidade que não tem litoral

**When** o usuário faz uma requisição para a API informando um código de uma cidade sem litoral

**Then** a resposta deve conter o status "404 Not Found"

**And** o corpo da resposta deve conter uma mensagem amigável informando que o código de cidade informando é inválido para consulta de ondas

**But** não deve retornar dados de ondas que é o objetivo do método

*2º Cenário negativo:*

**Feature:** Consulta de CEP V2

**Scenario:** Teste de performance da API de busca por CEP V2

**Given** o usuário tem acesso ao método de busca por CEP V2 no Postman

**And** o usuário pretende testar a resistência a API

**When** o usuário faz 1000 requisições simultâneas para a API com valor de CEP válido

**Then** pelo menos uma das respostas apresenta status de erro

**And** o tempo de resposta é maior do que o critério de desempenho aceitável

**But** a API não deve retornar status OK para todas as requisições, demonstrando que não suportou a carga

## 6. Observações

Coloque aqui comentários adicionais e limitações que você tenha identificado nos cenários de teste que descreveu.

Resposta:

Os métodos utilizados da Brasil API são objetivos e práticos para utilização, eu não conhecia essa plataforma e achei bastante interessante, apesar de particularmente esperar uma documentação mais abrangente.

A respeito dos cenários que descrevi entre os exercícios E3 a E5, são métodos sem grande complexidade, bastante intuitivos, mas se fosse um projeto no qual tivesse trabalhado eu teria pensado em melhores mensagens de retorno para o usuário em casos de erros. No mais, em casos de sucesso, os bodys são bem organizados, as informações são apresentadas dentro do esperado, percebi pequenas diferenças entre o resultado que tive no Postman e os exemplos no site, mas sem impacto significativo.

Falando sobre os cenários de descrevi em Gherkin, espero ter apresentado um resultado ao menos dentro do esperado pelo avaliador deste teste, pois é algo que não tenho muita experiência, então procurei ser coerente na estrutura lógica dos testes propostos, apresentando verificações que sejam realmente pertinentes e que fossem aplicáveis em um ambiente de testes.