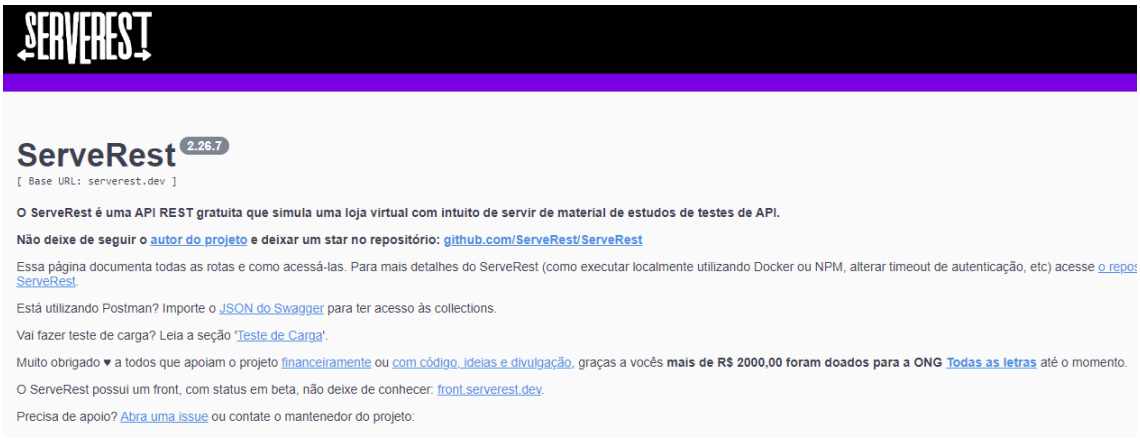
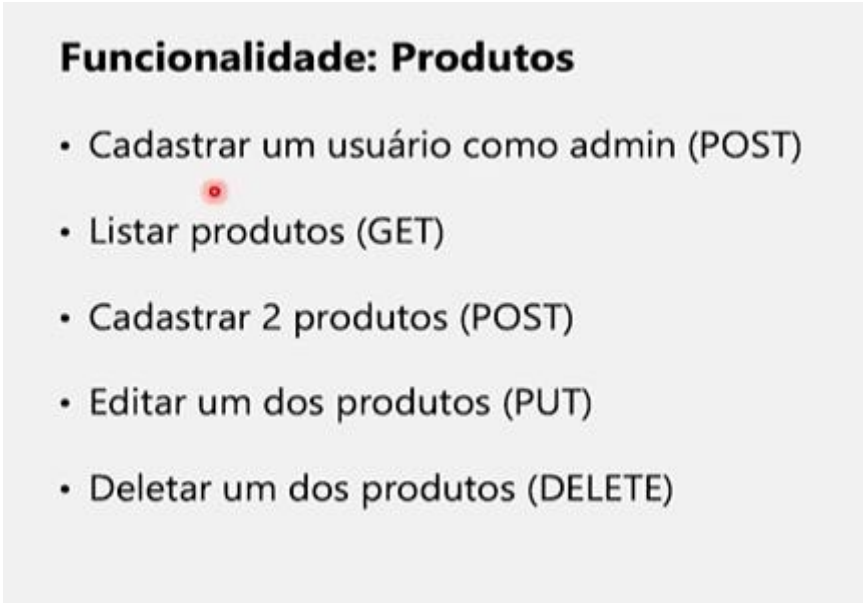
Teste de API e2e usando Postman

Vamos fazer um teste e2e no servidor serverest.dev (servidor que simula uma loja virtual).



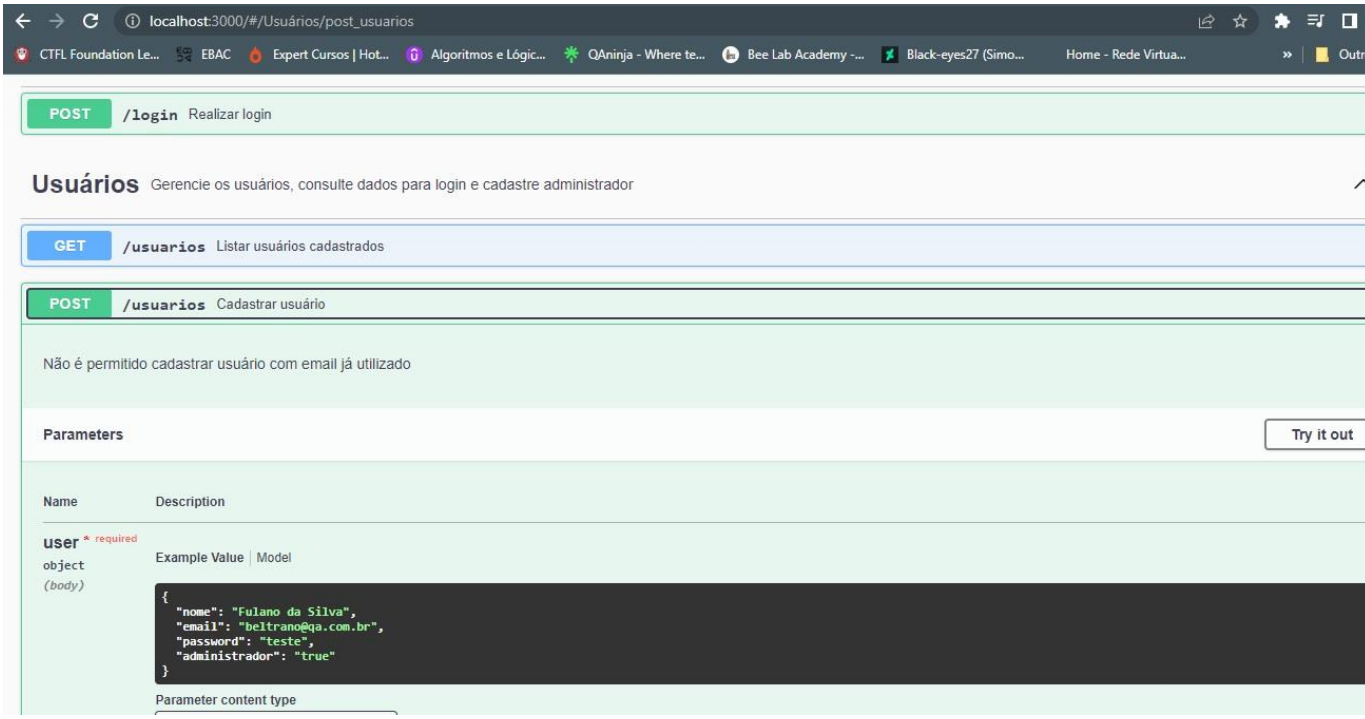
Vamos efetuar os seguintes testes:



Abra o terminal Bash e digite:

Npx serverest – vai abrir o serverest no localhost (no navegador) sobe um servidor na máquina local.

Clique na opção onde se cadastra um produto (No método POST) vai ficar assim:



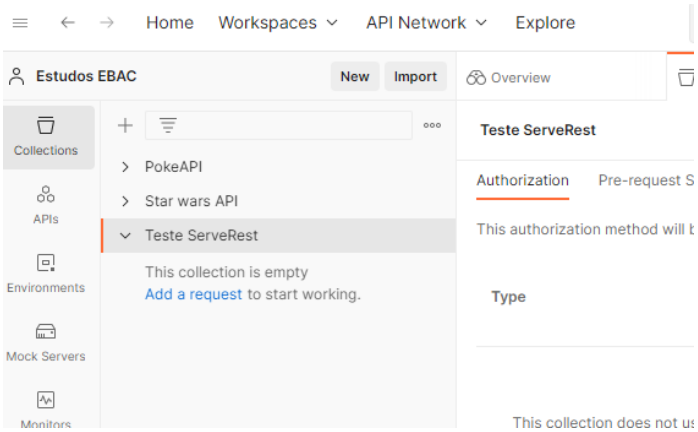
Observe as informações (isso é muito importante), logo acima a mensagem "Não é permitido cadastrar usuário com e-mail já utilizado" é um critério.

Observe o exemplo:

```
{
  "nome": "Fulano da Silva",
  "email": "beltrano@qa.com.br",
  "password": "teste",
  "administrador": "true"
}
```

Agora abra o Postman.

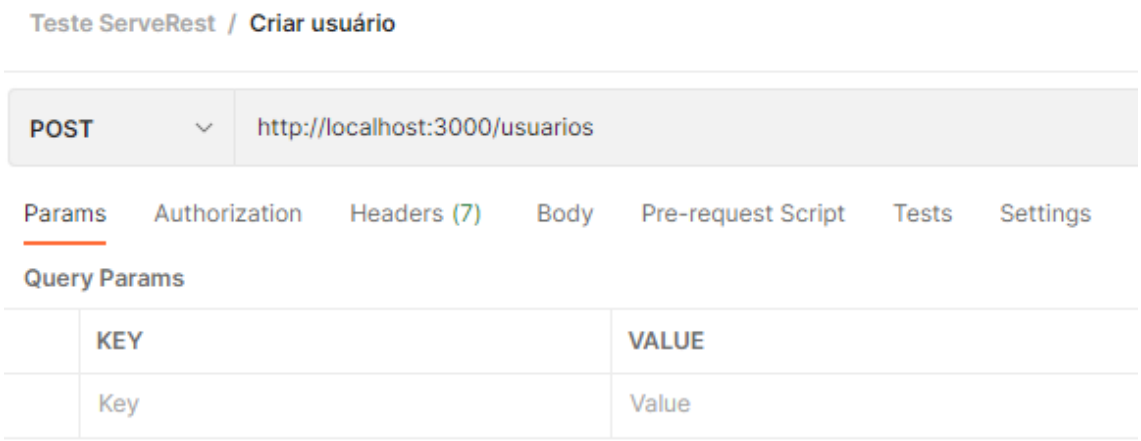
Crie uma nova Collection de nome "Teste ServeRest" assim:



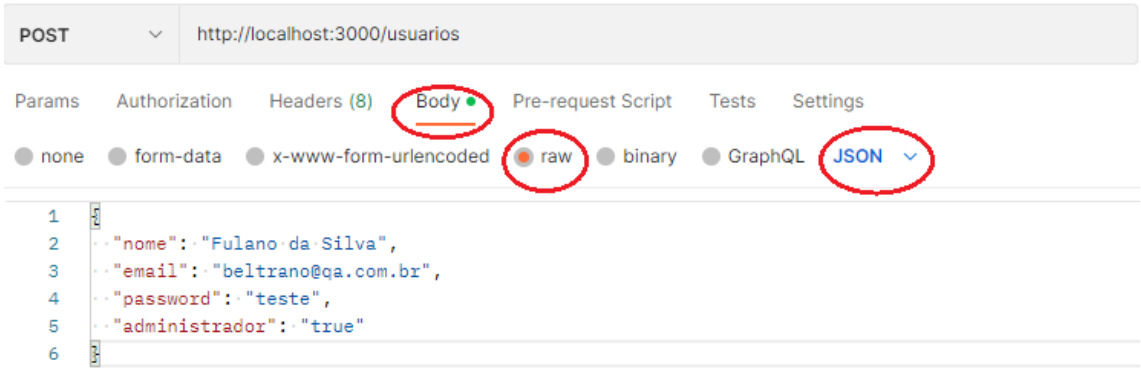
Agora crie uma nova request com o nome “Criar usuário”

Mude o método para POST e no local da URL adicione o endereço http://localhost:3000/ com o end point Usuário, ficara assim: http://localhost:3000/usuario.

Observação: se você clicar em “Send” vai dar erro, será apresentada a mensagem abaixo, falando da obrigatoriedade de colocar os dados do usuário.



Copie os dados de exemplo da API (do método POST) e adicione na opção “Body” do Postman:



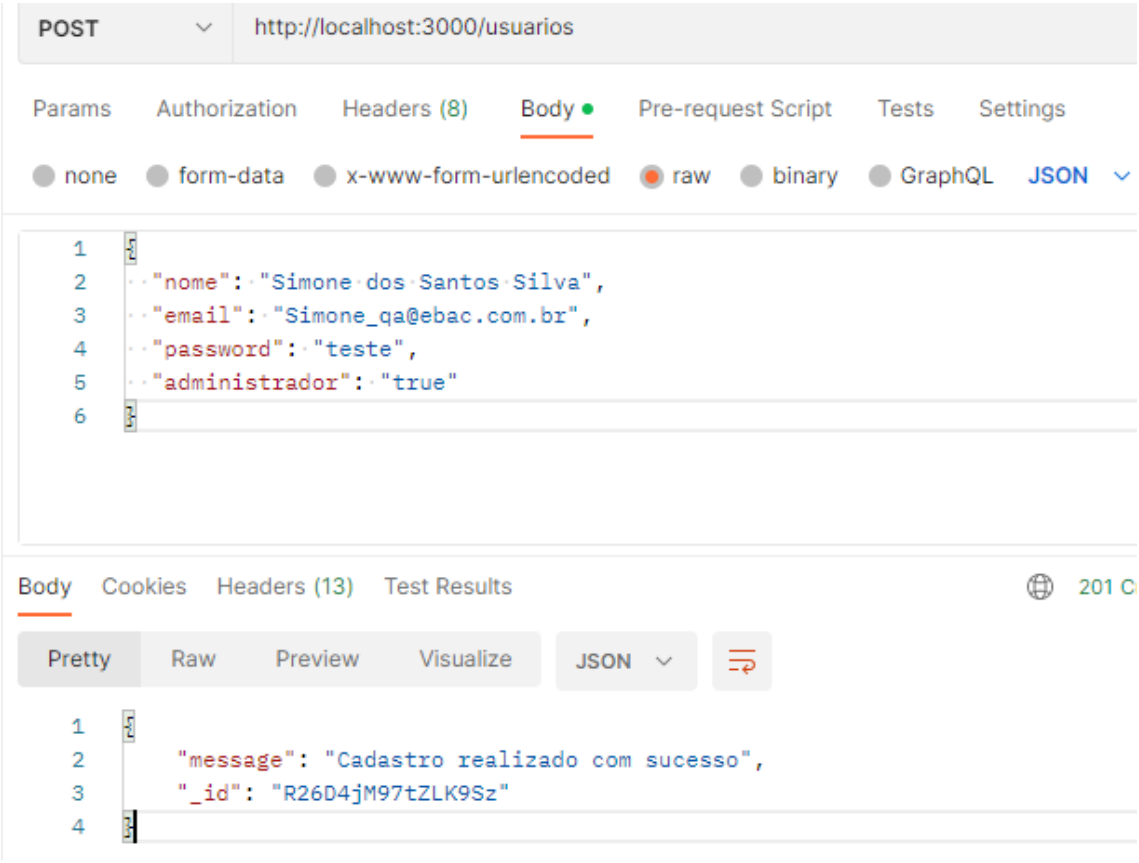
Clique na opção “raw” e mude também a opção de “Text” para Json, conforme mostrado na imagem acima.

Na sequência será necessário mudar os dados da mensagem:

```
{
  "nome": "Simone dos Santos Silva",
  "email": "Simone_qa@ebac.com.br",
  "password": "teste",
  "administrador": "true"
}
```

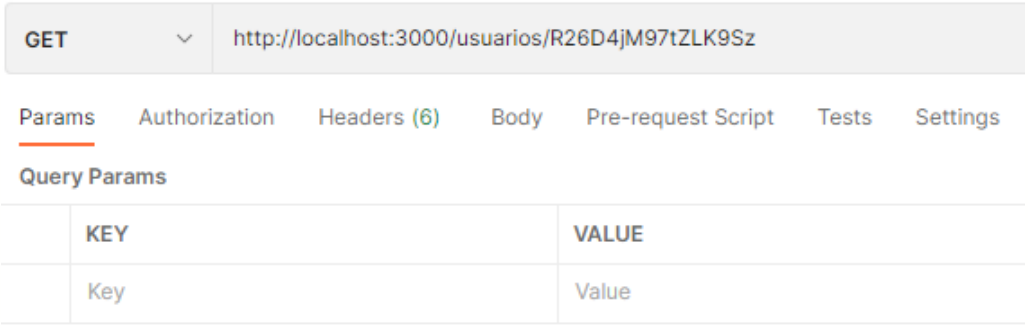
O “administrador” tem que ser “true” para poder ser possível cadastrar o usuário.

Clique em “Send” o cadastro deve ser efetuado com sucesso, conforme mensagem abaixo:



Foi gerado um ID `"R26D4jM97tZLK9Sz"`

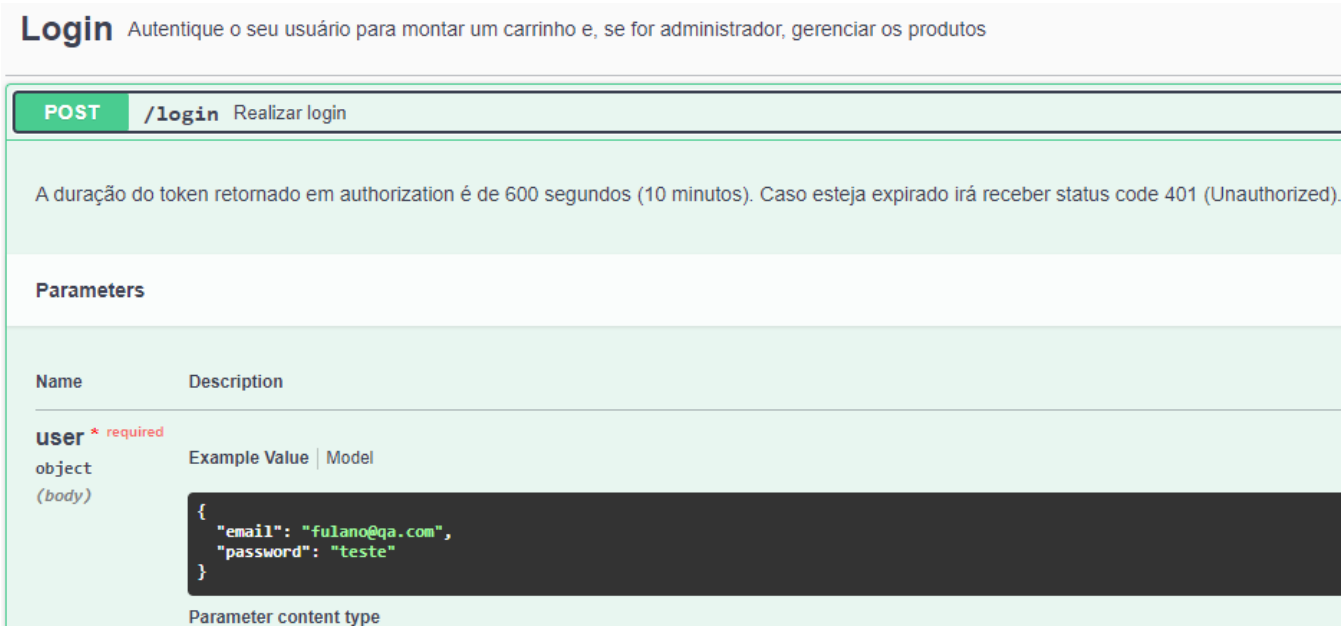
Agora num novo request com o método GET e a URL http://localhost:3000/usuarios/R26D4jM97tZLK9Sz observe que será apresentado os dados do usuário correspondente ao ID informado:



Se tirarmos o ID da URL vai listar todos os usuários cadastrados (com o método GET), portanto toda vez que fizermos um cadastro podemos dar um GET para verificar se foi mesmo cadastrado.

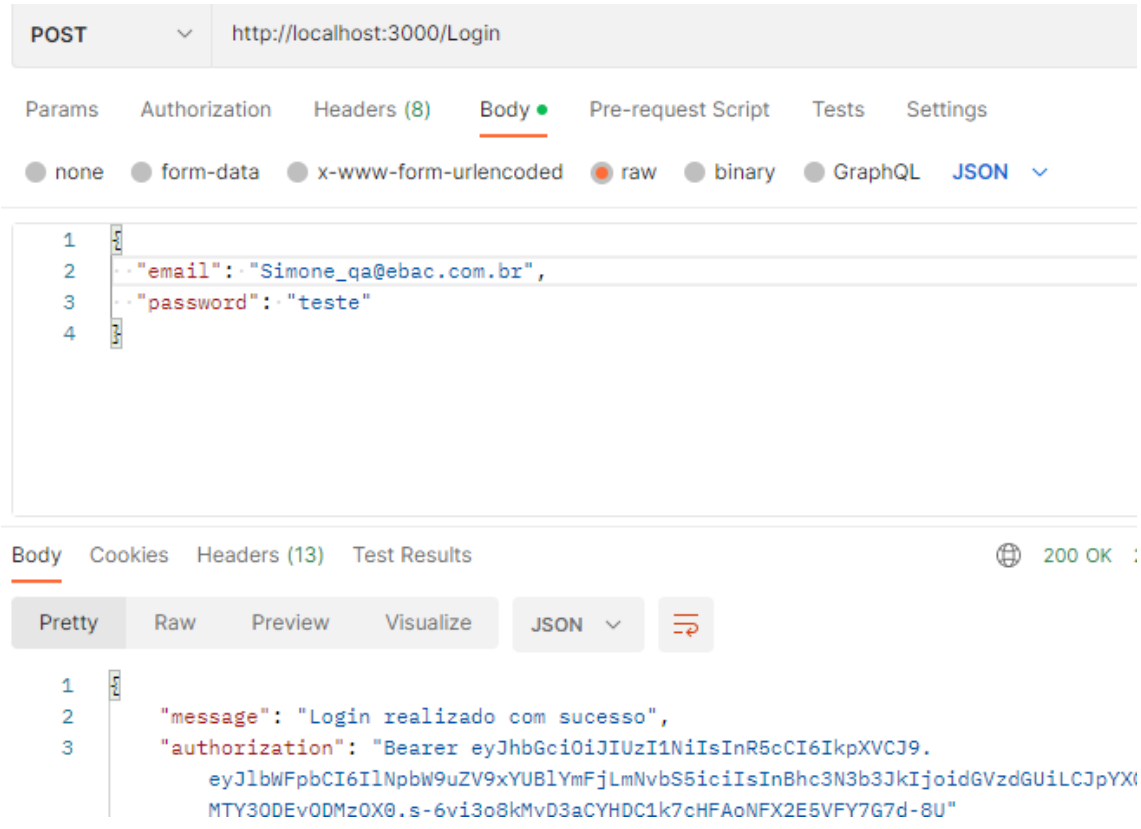
Faça o teste usando GET para listar os usuários e depois salve com o nome "Listar usuários" na Collection "Teste ServeRest".

Agora vamos fazer o login, entenda olhando a documentação como se faz o login:



Usaremos o método POST usando o end point “login” assim: http://localhost:3000/Login.

Em “Body” com a opção “raw” e o “Text” como Json, acrescente os dados de login:



Ele gerou um token de autorização:

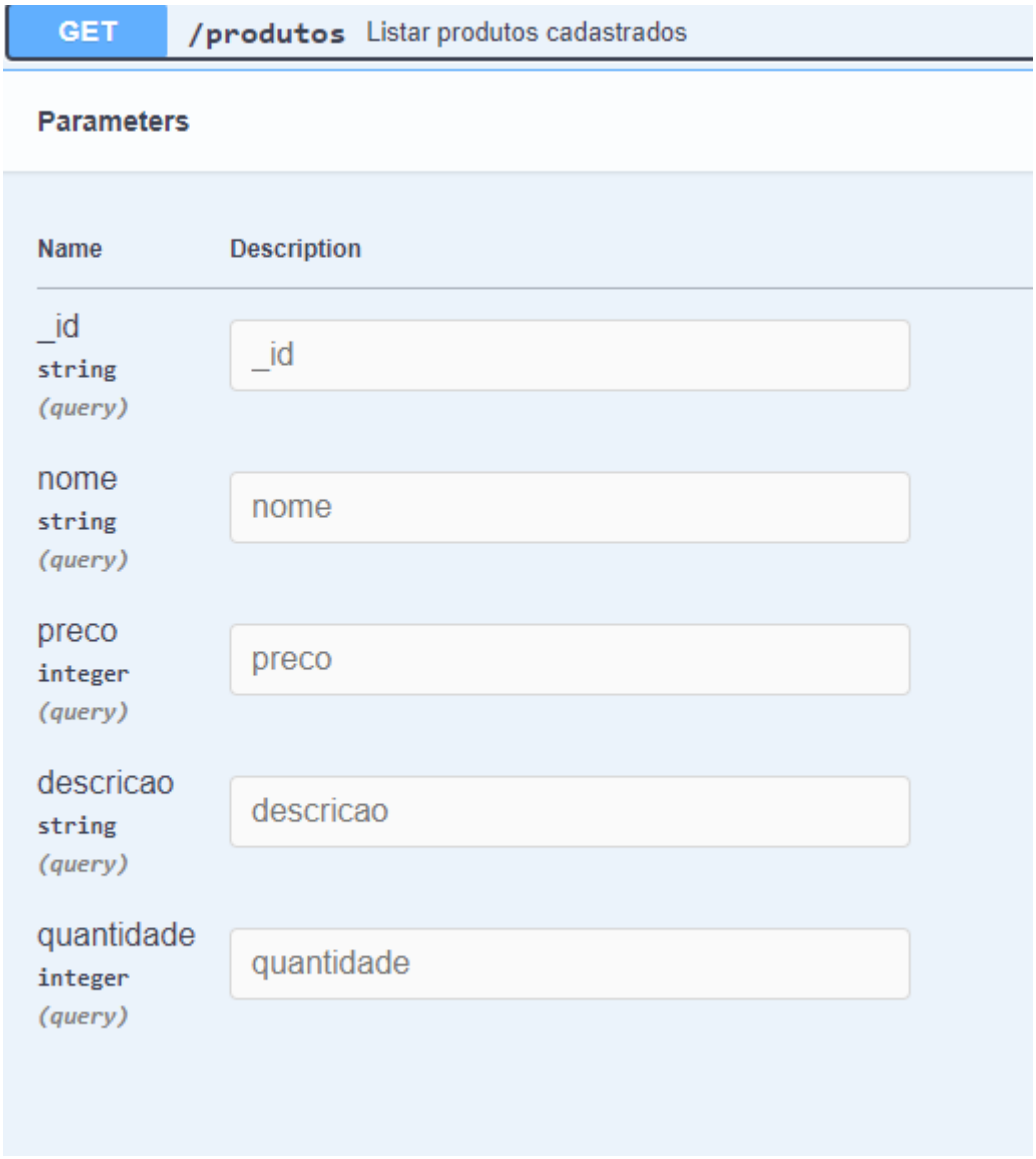
"authorization": "Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJlbWFpbCI6IlNpbW9uZV9xYUBlYmFjLmNvbS5iciIsInBhc3N3b3JkIjoidGVzdGUiLCJpYXQiOjE2NzgxMjc3MzksImV4cCI6MTY3ODEyODMzOX0.s-6vi3o8kMyD3aCYHDC1k7cHFAoNFX2E5VFY7G7d-8U"

Preste bastante atenção nisso, pois por vezes pode -se precisar um token para poder ter acesso a uma certa API.

E agora vamos começar nossos testes em produtos

Para isso, crie uma pasta “três pontinhos, New folder” dentro da Collection “Test Servrest”, feito isso crie um novo request com o nome “Listar produtos”, veja na documentação como se dá um GET em “Produtos”. A URL será http://localhost:3000/Produtos. Veja na imagem abaixo a documentação do método

GET para produtos:



Adicione a URL http://localhost:3000/Produtos e clique em "Send" (ou dá "Enter"), será possível visualizar 2 produtos.

Na opção "Tests" do lado direito, escolha um script pronto de nome "Response body: contain string", e altere o nome para "Validar produto da lista", assim:

```
pm.test("Validar produto da lista", function () {
    pm.expect(pm.response.text()).to.include("string_you_want_to_search");
});
```

Mude o nome da string para nome do produto, assim:

```
pm.test("Validar produto da lista", function () {
    pm.expect(pm.response.text()).to.include("Samsung 60 polegadas");
});
```

Rode o teste (clique em Send), o teste irá passar, olhe na opção "Tests results".

Agora adicione também o teste de "Status code: code is 200", assim:

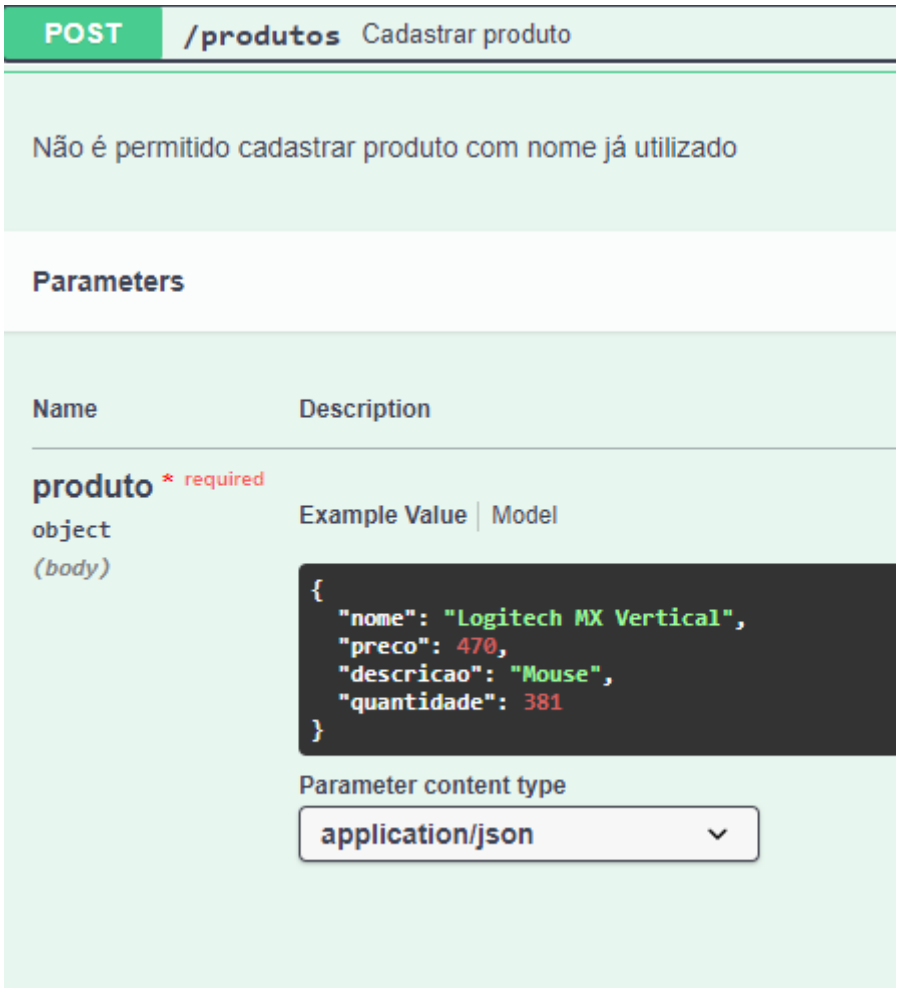
```
pm.test("Status code é 200", function () {
```

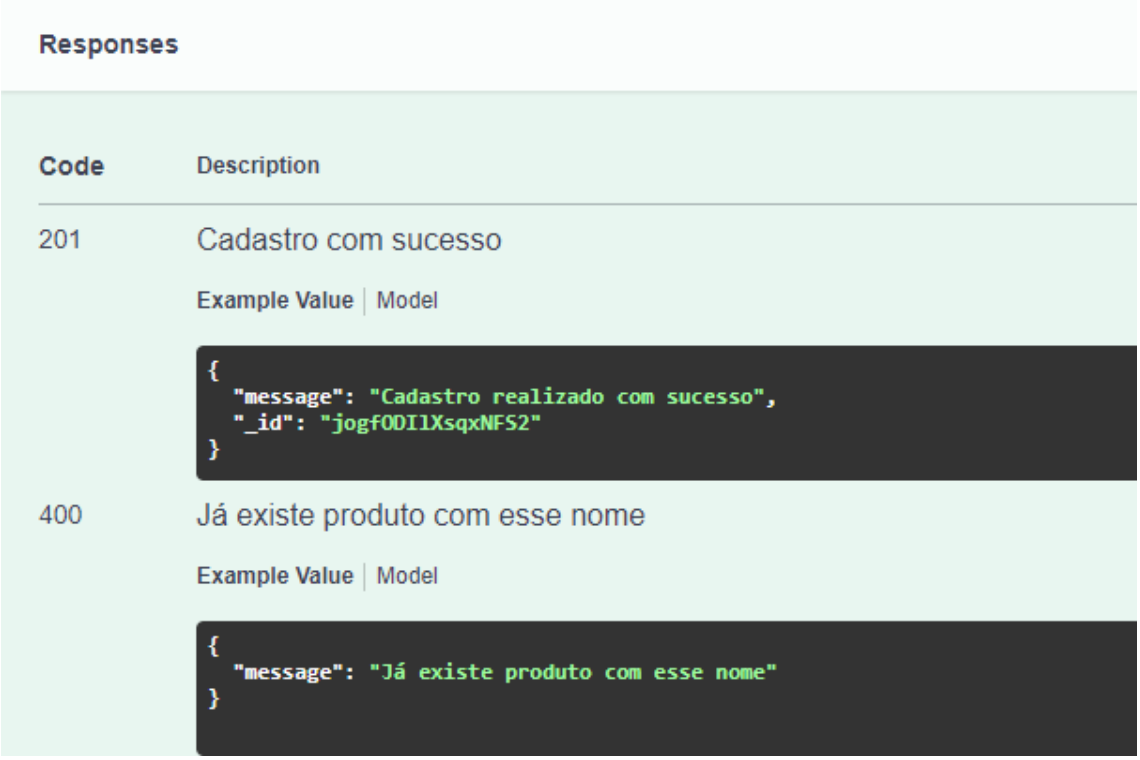
```
    pm.response.to.have.status(200);
});
```

Rode o teste (agora são dois testes).

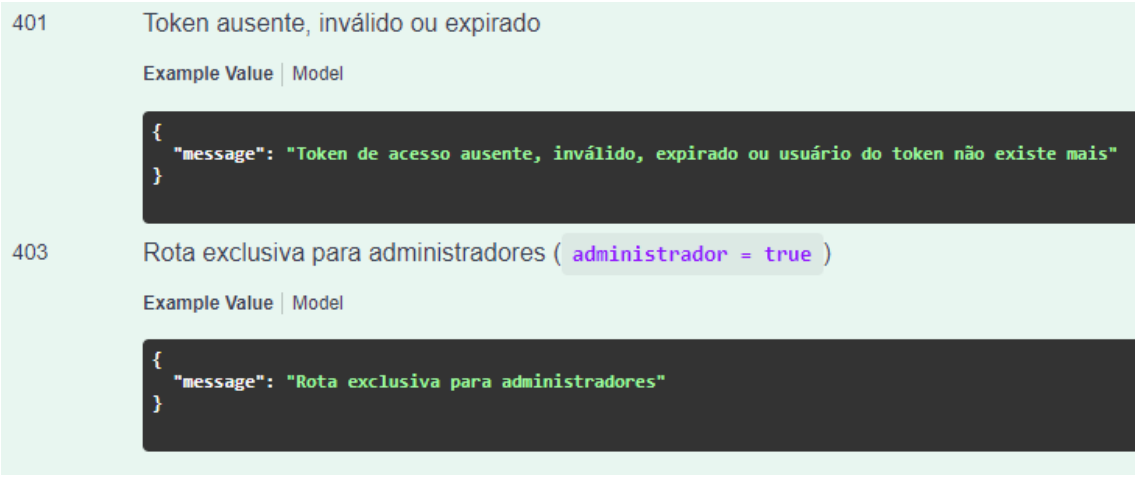
Agora num novo request, vamos cadastrar um produto (na mesma URL http://localhost:3000/Produtos), mas lembre-se para fazer cadastro o método é POST. Olhe na documentação como fazer para cadastrar.

Observe um cadeado do lado direito da imagem acima, significa que para cadastrar um produto temos que ser administrador (por isso já criamos uma conta como administrador).





Possíveis mensagens como resposta, 201 cadastro com sucesso, 400 já existe um produto com esse nome.



401 Token ausente, inválido ou expirado, 403 Rota exclusiva para administradores.

Nos testes é possível validar todas essas mensagens, tanto as de erro como as do caminho feliz, mas vamos fazer apenas a do caminho feliz.

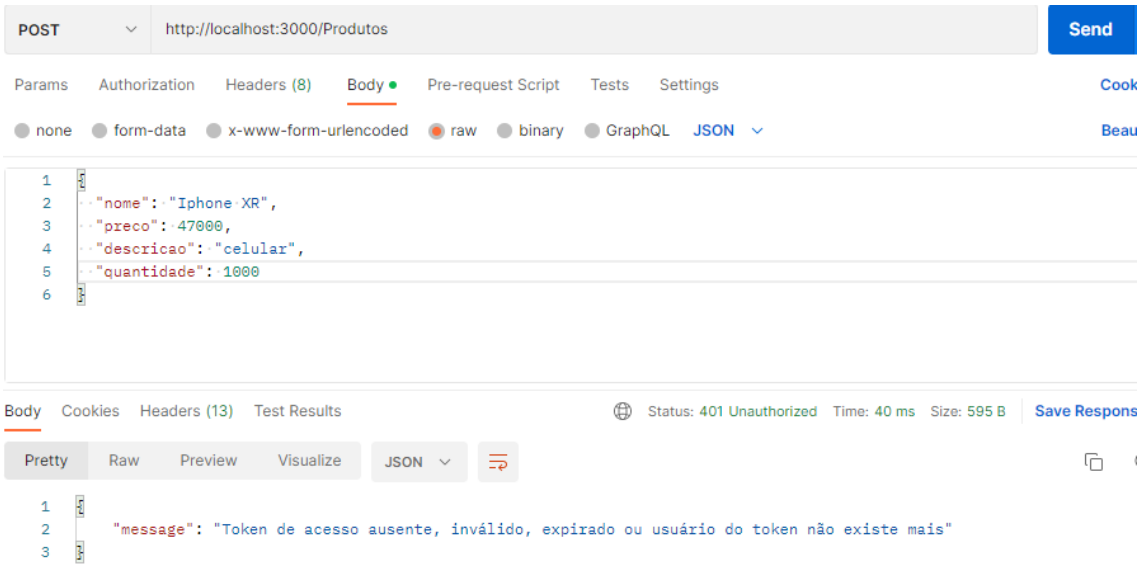
No Postman clique em "Body", marque a opção "raw", e em "Text" escolha a opção Json.

Cole o modelo do produto (exemplo mostrado na documentação) e mude os dados:

```
{
  "nome": "Logitech MX Vertical",
  "preco": 470,
  "descricao": "Mouse",
  "quantidade": 381
```

```
}
```

Após mudar os dados clique em “Send”, aparecerá o erro abaixo:

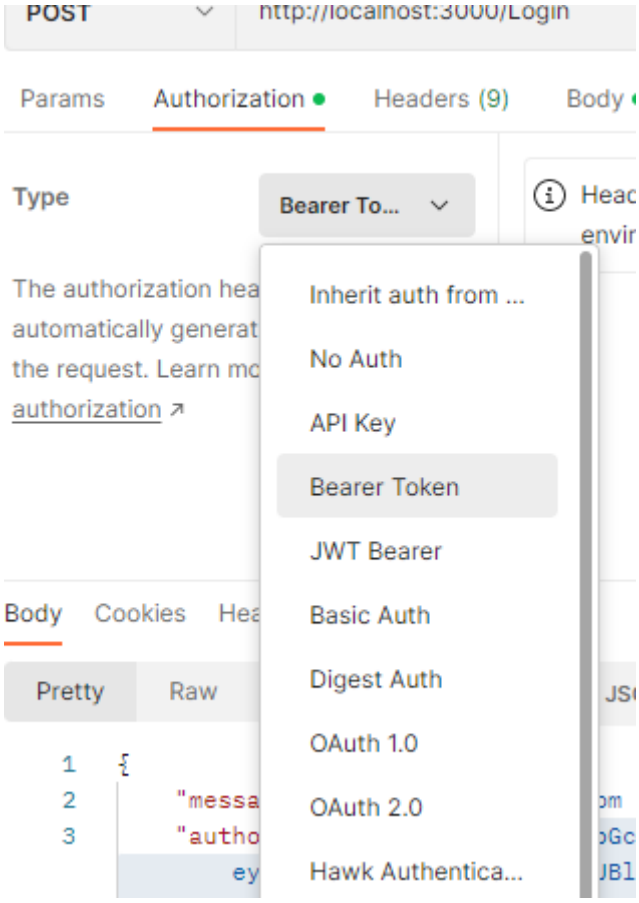


Deu a mensagem de status 401 (não autorizado) e a mensagem de Token de acesso ausente, isso já estava informado no swegger (documentação).

Bem no Postman clique na requisição de login e dê um Send, abaixo será mostrado um Token gerado, copie esse token:

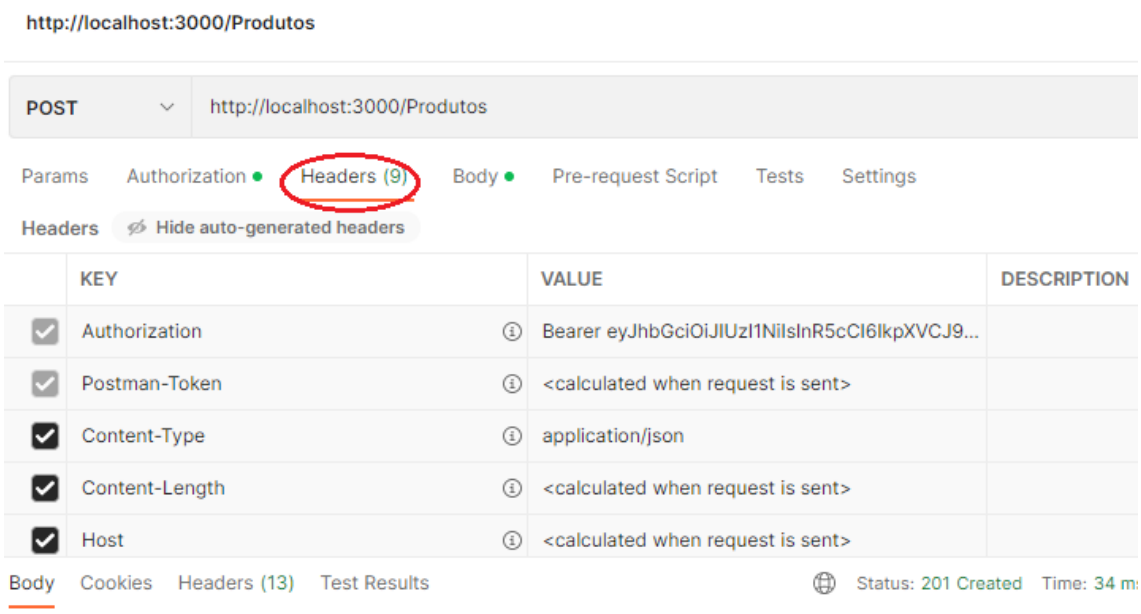
"Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJlbWFpbCI6IlNpbW9uZV9xYUBlYmFjLmNvbS5iciIsInBhc3N3b3JkIjoidGVzdGUiLCJpYXQiOjE2NzgyMTUyMDIsImV4cCI6MTY3ODIxNTgwMn0.PzA8m2OkaTM6cjn1ziSATW6QHh84-6kNHc0lgHZOJGU"

Copie sem as aspas duplas “”, agora clique na opção “Authorization” e escolha o tipo de autorização, na documentação da API sempre é informada o tipo de autorização, desta API é a “Bearer”.



Adicione o token (sem aspas e sem a palavra Bearer) e clique em "Send" no método POST (de cadastro de produtos), se der erro basta gerar um novo token na requisição de Login, copiar e colar o novo token.

Se clicarmos na opção "Headers" é possível visualizar o token como mostra a imagem abaixo:



Clique na request “Listar produtos” e dê um Send, será possível visualizar o produto que cadastramos.

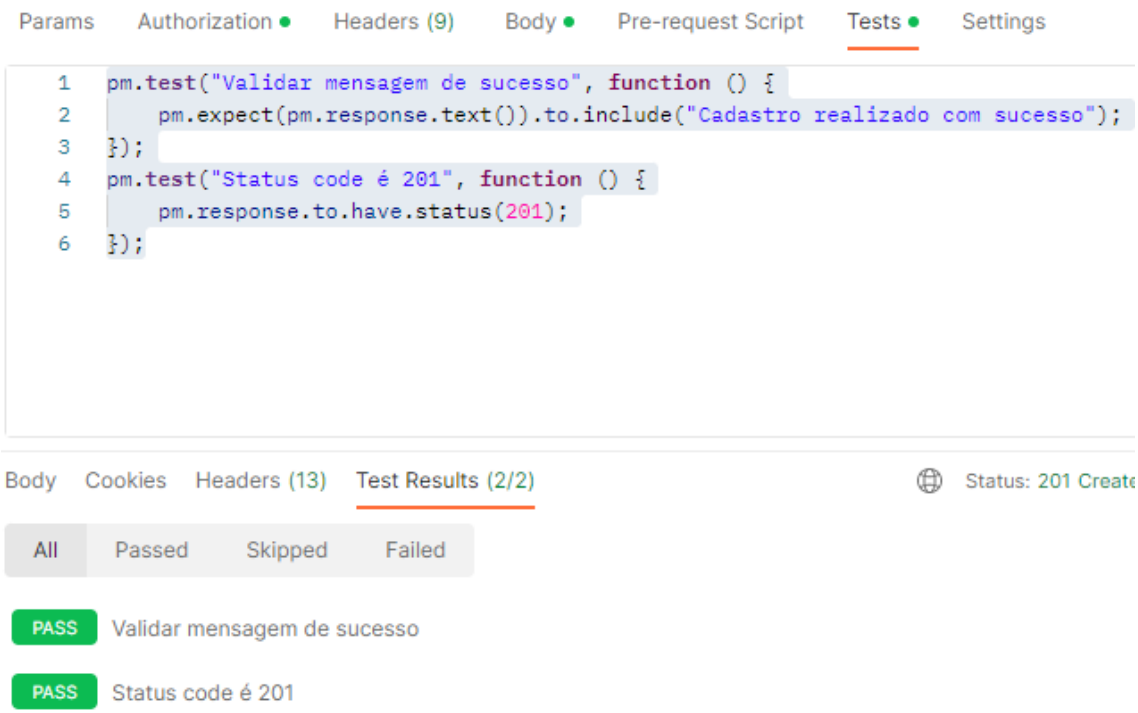
Volte para “Cadastro de produtos” e faça uma asserção (basta copiar o código anterior e mudar alguns dados) assim:

```
pm.test("Validar mensagem de sucesso", function () {
    pm.expect(pm.response.text()).to.include("Cadastro realizado com sucesso");
});
pm.test("Status code é 201", function () {
    pm.response.to.have.status(201);
});
```

O código acima vai validar a mensagem “Cadastro realizado com sucesso”, se der erro de token basta gerar um novo token no request de Login.

Em “Listar produtos é possível visualizar os produtos cadastrados” para ver um produto em específico basta copiar seu ID (na lista de produtos) e colar na URL, ficará assim: http://localhost:3000/Produtos/6t92ZXIbSJMQrndG, irá mostrar o produto correspondente ao ID.

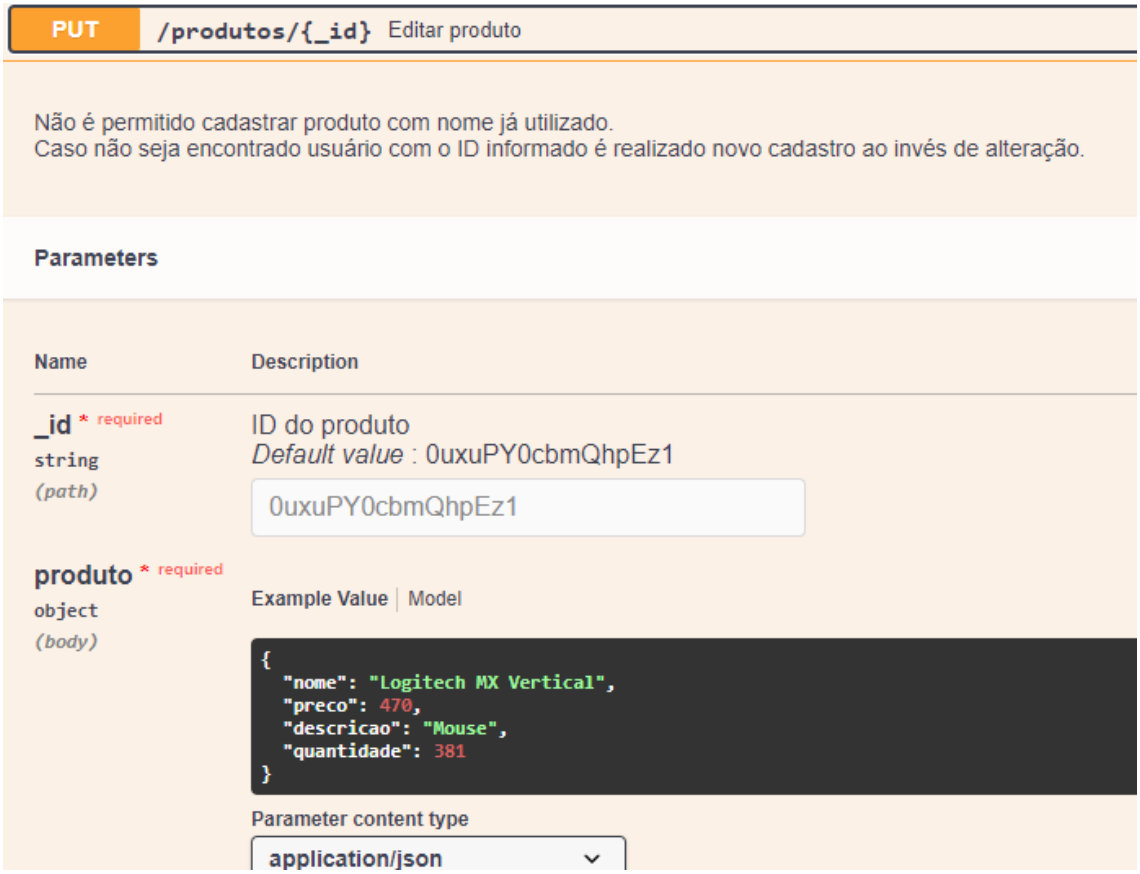
Observação: para realizar o teste acima na opção “Tests” é necessário descrever um novo produto na opção “Body” sem clicar em “Send”, também certifique se o token ainda é válido e só depois disso excute o teste clicando em “Send”, os dois testes devem passar, conforme imagem abaixo:



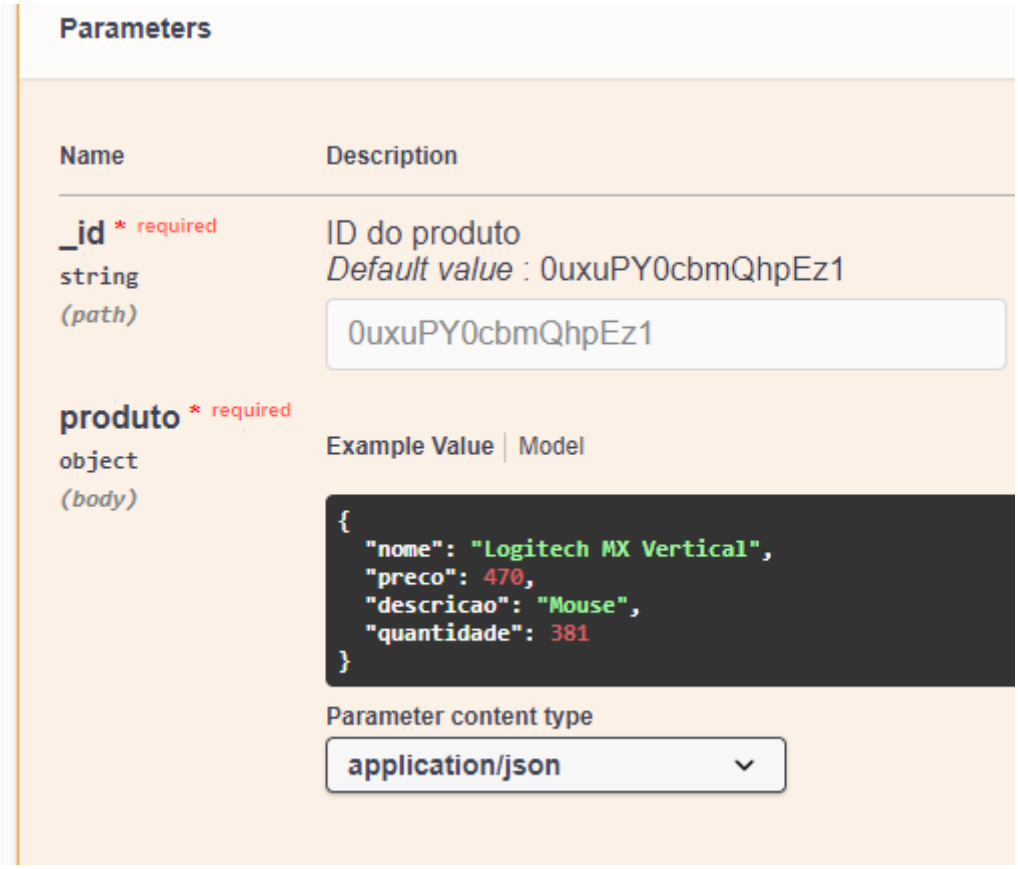
Agora vamos editar um produto, pra isso vamos utilizar o método PUT, adicione uma nova request e mude o método para PUT, em seguida para não ficarmos adicionando toda hora a URL vamos criar uma variável global. Clique em “Enviroments” e depois

em “Globals” e adicione a variável “Local”, na frente do nome a URL da API http://localhost:3000/. De volta a request digite {{local}}/ Produtos no local da URL.

Visualize na documentação Swegger como faz para editar um produto, veja imagem abaixo:



Observe a mensagem acima sobre permissão. Na verdade o método PUT é semelhante ao POST. Esse método vai fazer uma comparação com o que você está enviando com o que já existe na base, se estiver alteração em um dos parâmetros ele vai fazer a alteração. Nesse método também precisa ter o token de acesso. Vamos ter que usar uma mensagem como no POST, segue exemplo abaixo:



E vamos fazer o caminho feliz, terá que apresentar a mensagem de sucesso abaixo:

```
{
  "message": "Registro alterado com sucesso"
}
```

Mensagem 200.

Clique em "Body" marque a opção "raw" e em "Text" escolha Json, cole a mesmagem (payload):

```
{
  "nome": "Logitech MX Vertical",
  "preco": 470,
  "descricao": "Mouse",
  "quantidade": 381
}
```

Mude os parâmetros:

```
{
  "nome": "Iphone XR V2",
  "preco": 3700,
  "descricao": "Telefone celular",
  "quantidade": 500
}
```

Vá na resquest "Listar produtos" e peque o ID do produto Iphone e cole na URL assim: {{local}}/bP2mnphOqA9P7DyV. Na request "Login" gere um novo token, volte para a request PUT clique em "Authorization" e adicione o token (como já fopi feito anteriormente), execute clicando em "Send" deve apresentar a mensagem de sucesso:

```
{
   "message": "Registro alterado com sucesso"
}
```

Volte na request “Listar produtos” e dê um Send para ver a alteração do produto.

Salve o PUT como Editar produto, salve o POST como Cadastrar produto.

Agora vamos usar o método DELETE, para excluir um produto. Crie uma nova request e na URL adicione a variável global.

Clique na request “Listar produtos” e copie o ID do produto TV sansung (pode ser qualquer outro também), cole o ID na URL (na frente da variável global), volte para a request DELETE, não esquece de ver na documentação como deletar um produto, clique em “Tests” e adicione dois scripts Status code: code is 200 e Response body: contem string:

```
pm.test("Status code is 200", function () {
   pm.response.to.have.status(200);
});
pm.test("Body matches string", function () {
   pm.expect(pm.response.text()).to.include("string_you_want_to_search");
});
```

Mude os parâmetros conforme código abaixo:

```
pm.test("Status code é 200", function () {
   pm.response.to.have.status(200);
});
pm.test("Validar mensagem", function () {
   pm.expect(pm.response.text()).to.include("Registro excluído com sucesso");
});
```

Antes de executar o teste, gere um novo token na request “Login” copie e cole em “Authorization”. Agora basta executar o teste, deve apresentar a mensagem de sucesso e em “Tests results” deve passar os dois testes. Salva esse método com o nome “Excluir produto”.