



# API Serverest

Yalim Alam Schust dos Santos  
Julho/2022

## 1. Introdução

Esse é um plano de teste realizado em cima da API Serverest, em que serão testadas as funcionalidades da API e se sua documentação está correta. A documentação da API pode ser localizada em: <https://serverest.dev/>

## 2. Objetivo

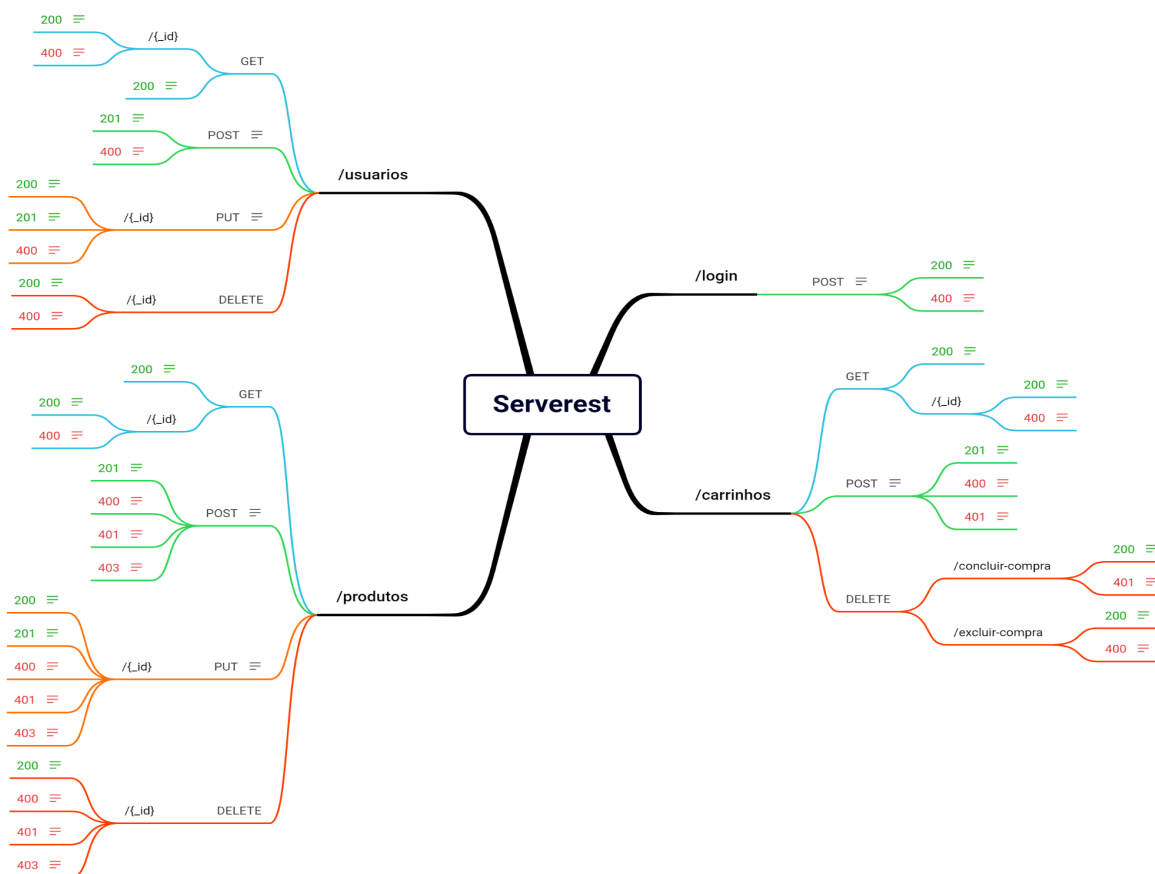
Testar os endpoints da API para verificar se sua funcionalidade está funcionando e se a documentação está correta.

## 3. Escopo

O que será testado na API é:

- Todos os seus endpoints, sendo testado tanto as respostas com status code positivos quanto os negativos;
- A sua documentação para verificar se o status code e o retorno estão corretos.

## 4. Mapa Mental



Outra forma de acessar o mapa mental é acessando a pasta “documents” e abrindo o arquivo “Mapa Mental.xmind” com o programa XMind.

## 5. Suíte de casos de teste

Testes que serão realizados:

**OBS:** Os casos de testes **azuis** são testes positivos e os **vermelhos** são testes negativos

### 5.1 Testes da rota /usuarios

T01 - Buscar todos os usuários
<b>Descrição:</b> Deve buscar todos os usuários cadastrados
<b>Rota:</b> /usuarios
<b>Método:</b> GET
<b>Etapas do Teste:</b> 1 - Criar requisição; 2 - Realizar requisição.
<b>Status Code:</b> 200

T02 - Buscar um usuário em específico
<b>Descrição:</b> Deve buscar um usuário em específico utilizando seu ID
<b>Rota:</b> /usuarios/{_id}
<b>Método:</b> GET
<b>Etapas do Teste:</b> 1 - Pegar o ID de um usuário cadastrado; 2 - Criar requisição; 3 - Realizar requisição.
<b>Status Code:</b> 200

### T03 - Postar um novo usuário

**Descrição:**

Deve postar um novo usuário

**Rota:**

/usuarios

**Método:**

POST

**Parâmetro:**

```
{  
  "nome": " ",  
  "email": " ",  
  "password": " ",  
  "administrador": " "  
}
```

**Etapas do Teste:**

- 1 - Criar requisição;
- 2 - Realizar requisição.

**Status Code:**

201

### T04 - Modificar as informações de um usuário

**Descrição:**

Deve modificar as informações de um usuário existente

**Rota:**

/usuarios/{\_id}

**Método:**

PUT

**Parâmetro:**

```
{  
  "nome": " ",  
  "email": "Mantém o mesmo e-mail",  
  "password": " ",  
  "administrador": " "  
}
```

**Etapas do Teste:**

- 1 - Pegar o ID de um usuário cadastrado;
- 2 - Criar requisição;
- 3 - Realizar requisição.

<b>Status Code:</b> 200
----------------------------

<b>T05 - Criar um novo usuário com o método PUT</b>
---

<b>Descrição:</b>
-------------------

Deve criar um novo usuário modificando o e-mail de um usuário que já exista
---

<b>Rota:</b>
--------------

/usuarios/{_id}
-----------------

<b>Método:</b>
----------------

PUT
-----

<b>Parâmetro:</b>
-------------------

<pre>{   "nome": " ",   "email": " ",   "password": " ",   "administrador": " " }</pre>
---

<b>Etapas do Teste:</b>
-------------------------

- |   |
|---|
| <ul style="list-style-type: none"><li>1 - Gerar um ID novo para criar um novo usuário;</li><li>2 - Criar requisição;</li><li>3 - Realizar requisição.</li></ul> |
|---|

<b>Status Code:</b>
---------------------

201
-----

<b>T06 - Deletar um usuário</b>
---------------------------------

<b>Descrição:</b>
-------------------

Deve deletar a conta de um usuário
------------------------------------

<b>Rota:</b>
--------------

/usuarios/{_id}
-----------------

<b>Método:</b>
----------------

DELETE
--------

<b>Etapas do Teste:</b>
-------------------------

- |   |
|---|
| <ul style="list-style-type: none"><li>1 - Pegar o ID de um usuário cadastrado</li><li>2 - Criar requisição</li><li>3 - Realizar requisição.</li></ul> |
|---|

<b>Status Code:</b>
---------------------

200
-----

<b>T07 - Deletar um usuário que não exista</b>
--

<b>Descrição:</b>
-------------------

Deve deletar a conta de um usuário que não esteja cadastrado no sistema
---

<b>Rota:</b>
--------------

/usuarios/{_id}
-----------------

<b>Método:</b>
----------------

DELETE
--------

<b>Etapas do Teste:</b>
-------------------------

- |   |
|---|
| 1 - Criar requisição passando um ID inexistente<br>2 - Realizar requisição. |
|---|

<b>Status Code:</b>
---------------------

200
-----

<b>T08 - Buscar um usuário que não exista</b>
---

<b>Descrição:</b>
-------------------

Deve tentar fazer a busca de um usuário que não esteja cadastrado no sistema
--

<b>Rota:</b>
--------------

/usuarios/{_id}
-----------------

<b>Método:</b>
----------------

GET
-----

<b>Etapas do Teste:</b>
-------------------------

- |  |
|--|
| 1 - Criar requisição passando um ID inexistente;<br>2 - Realizar requisição. |
|--|

<b>Status Code:</b>
---------------------

400
-----

<b>Mensagem de erro esperada:</b>
-----------------------------------

Usuário não encontrado
------------------------

<b>T09 - Postar um novo usuário com email já cadastrado</b>
---

<b>Descrição:</b>
-------------------

Deve tentar fazer o cadastro de um novo usuário que já tenha seu email cadastrado no sistema
--

<b>Rota:</b> /usuarios
<b>Método:</b> POST
<b>Parâmetro:</b> { "nome": " ", "email": "Algum e-mail já cadastrado no sistema", "password": " ", "administrador": " " }
<b>Etapas do Teste:</b> 1 - Buscar e-mail de um usuário já cadastrado no sistema 2 - Criar requisição; 3 - Realizar requisição.
<b>Status Code:</b> 400
<b>Mensagem de erro esperada:</b> Este email já está sendo usado

<b>T10 - Modificar as informações de usuário passando um outro e-mail que já exista</b>
<b>Descrição:</b> Deve tentar modificar as informações de usuário passando, como parâmetro, um email de outro usuário cadastrado
<b>Rota:</b> /usuarios/{_id}
<b>Método:</b> PUT
<b>Parâmetro:</b> { "nome": " ", "email": "E-mail de outro usuário cadastrado no sistema", "password": " ", "administrador": " " }
<b>Etapas do Teste:</b> 1 - Buscar e-mail de um outro usuário já cadastrado no sistema 2 - Criar requisição; 3 - Realizar requisição.
<b>Status Code:</b> 400

<b>Mensagem de erro esperada:</b> Este email já está sendo usado
---

<b>T11 - Deletar usuário com carrinho cadastrado</b>
--

<b>Descrição:</b> Deve tentar deletar um usuário que tenha um carrinho cadastrado
--

<b>Rota:</b> /usuarios/{_id}
---------------------------------

<b>Método:</b> DELETE
--------------------------

<b>Etapas do Teste:</b> 1 - Criar requisição passando o ID de um usuário com carrinho registrado; 2 - Realizar requisição.
--

<b>Status Code:</b> 400
----------------------------

<b>Mensagem de erro esperada:</b> Não é permitido excluir usuário com carrinho cadastrado
--

## 5.2 Testes da rota /login

<b>T12 - Realizar login com sucesso</b>
---

<b>Descrição:</b> Deve realizar o login com sucesso
--

<b>Parâmetro:</b> { "email": " ", "password": " " }
---

<b>Rota:</b> /login
------------------------

<b>Método:</b> POST
------------------------

<b>Etapas do Teste:</b> 1 - Pegar o e-mail e senha de um usuário cadastrado; 2 - Criar requisição; 3 - Realizar requisição.
--



<b>Status Code:</b> 200
----------------------------

<b>T13 - Falhar ao tentar realizar um login</b>
---

<b>Descrição:</b> Deve falhar ao tentar fazer login
--

<b>Parâmetro:</b> { "email": "Digitar e-mail errado", "password": "Digitar senha errada" }
--

<b>Rota:</b> /login
------------------------

<b>Método:</b> POST
------------------------

<b>Etapas do Teste:</b> 1 - Criar requisição; 2 - Realizar requisição.
--

<b>Status Code:</b> 400
----------------------------

<b>Mensagem de erro esperada:</b> Email e/ou senha inválidos
---

### 5.3 Testes da rota /produtos

<b>T14 - Buscar todos os produtos</b>
---------------------------------------

<b>Descrição:</b> Deve buscar todos os produtos cadastrados
--

<b>Rota:</b> /produtos
---------------------------

<b>Método:</b> GET
-----------------------

<b>Etapas do Teste:</b> 1 - Criar requisição; 2 - Realizar requisição.
--

<b>Status Code:</b>
---------------------

200
-----

#### T15 - Buscar um produto em específico

**Descrição:**

Deve buscar um produto em específico utilizando seu ID

**Rota:**

/produtos/{\_id}

**Método:**

GET

**Etapas do Teste:**

- 1 - Pegar o ID de um produto cadastrado;
- 2 - Criar requisição;
- 3 - Realizar requisição.

**Status Code:**

200

#### T16 - Postar um novo produto

**Descrição:**

Deve fazer o cadastro de um novo produto

**Parâmetro:**

```
{
  "nome": " ",
  "preco": number,
  "descricao": " ",
  "quantidade": number
}
```

**Rota:**

/produtos

**Método:**

POST

**Etapas do Teste:**

- 1 - Logar com um usuário administrador
- 2 - Criar requisição;
- 3 - Realizar requisição.

**Status Code:**

201

### T17 - Modificar as informações de um produto

**Descrição:**

Deve modificar as informações de um produto cadastrado no sistema

**Parâmetro:**

```
{  
  "nome": " ",  
  "preco": number,  
  "descricao": " ",  
  "quantidade": number  
}
```

**Rota:**

/produtos/{\_id}

**Método:**

PUT

**Etapas do Teste:**

- 1 - Logar com um usuário administrador
- 2 - Buscar o ID de um produto cadastrado;
- 3 - Criar requisição mudando um de seus parâmetros;
- 4 - Realizar requisição.

**Status Code:**

200

### T18 - Postar novo produto pelo método PUT

**Descrição:**

Deve modificar as informações de um produto cadastrado no sistema

**Parâmetro:**

```
{  
  "nome": " ",  
  "preco": number,  
  "descricao": " ",  
  "quantidade": number  
}
```

**Rota:**

/produtos/{\_id}

**Método:**

PUT

**Etapas do Teste:**

- 1 - Logar com um usuário administrador
- 2 - Gerar um ID novo para criar um novo produto;
- 3 - Criar requisição;
- 4 - Realizar requisição.

<b>Status Code:</b> 201
----------------------------

<b>T19 - Deletar um produto</b>
---------------------------------

<b>Descrição:</b> Deve deletar um produto do sistema
---

<b>Rota:</b> /produtos/{_id}
---------------------------------

<b>Método:</b> DELETE
--------------------------

<b>Etapas do Teste:</b> 1 - Logar com um usuário administrador 2 - Pegar o ID de um produto cadastrado 3 - Criar requisição 4 - Realizar requisição.
--

<b>Status Code:</b> 200
----------------------------

<b>T20 - Deletar um produto que não exista</b>
--

<b>Descrição:</b> Deve deletar um produto que não esteja cadastrado no sistema
---

<b>Rota:</b> /produtos/{_id}
---------------------------------

<b>Método:</b> DELETE
--------------------------

<b>Etapas do Teste:</b> 1 - Logar com um usuário administrador 2 - Criar requisição passando um ID inexistente 3 - Realizar requisição.
--

<b>Status Code:</b> 200
----------------------------

<b>T21 - Falhar ao pesquisar produto que não exista</b>
---

<b>Descrição:</b>
-------------------

Deve falhar ao tentar fazer a busca de um produto que não esteja cadastrado no sistema
<b>Rota:</b> /produtos/{_id}
<b>Método:</b> GET
<b>Etapas do Teste:</b> 1 - Buscar um ID inexistente; 2 - Criar requisição; 3 - Realizar requisição.
<b>Status Code:</b> 400
<b>Mensagem de erro esperada:</b> Produto não encontrado

<b>T22 - Falhar ao criar produto cujo nome já exista no sistema</b>
<b>Descrição:</b> Deve falhar ao tentar fazer o cadastro de um novo produto cujo nome já esteja cadastrado no sistema
<b>Parâmetro:</b> { "nome": "nome de produto já cadastrado", "preco": number, "descricao": " ", "quantidade": number }
<b>Rota:</b> /produtos
<b>Método:</b> POST
<b>Etapas do Teste:</b> 1 - Logar com um usuário administrador 2 - Buscar o nome de um produto já cadastrado; 3 - Criar requisição; 4 - Realizar requisição.
<b>Status Code:</b> 400
<b>Mensagem de erro esperada:</b> Já existe produto com esse nome

### T23 - Falhar ao tentar postar um produto logado com um token inválido

**Descrição:**

Deve falhar ao tentar postar um novo produto logado com um token inválido, ausente, expirado ou que o usuário não exista mais no sistema

**Parâmetro:**

```
{  
  "nome": " ",  
  "preco": number,  
  "descricao": " ",  
  "quantidade": number  
}
```

**Rota:**

/produtos

**Método:**

POST

**Etapas do Teste:**

- 1 - Buscar um token inválido;
- 2 - Criar requisição;
- 3 - Realizar requisição.

**Status Code:**

401

**Mensagem de erro esperada:**

Token de acesso ausente, inválido, expirado ou usuário do token não existe mais

### T24 - Falhar ao tentar postar um novo produto sem ser administrador

**Descrição:**

Deve falhar ao tentar fazer o cadastro de um novo produto logado com um usuário que não é administrador

**Parâmetro:**

```
{  
  "nome": " ",  
  "preco": number,  
  "descricao": " ",  
  "quantidade": number  
}
```

**Rota:**

/produtos

**Método:**

POST

**Etapas do Teste:**

- 1 - Logar com usuário que não é administrador;
- 2 - Criar requisição;
- 3 - Realizar requisição.

**Status Code:**

403

**Mensagem de erro esperada:**

Rota exclusiva para administradores

**T25 - Falhar ao modificar produto colocando um novo nome que já exista****Descrição:**

Deve falhar ao tentar modificar um produto, colocando o nome de outro produto que já esteja cadastrado no sistema

**Parâmetro:**

```
{  
  "nome": "nome de outro produto já cadastrado",  
  "preco": number,  
  "descricao": " ",  
  "quantidade": number  
}
```

**Rota:**

/produtos/{\_id}

**Método:**

PUT

**Etapas do Teste:**

- 1 - Logar com um usuário administrador;
- 2 - Buscar ID de um produto cadastrado;
- 3 - Buscar o nome de um outro produto já cadastrado;
- 4 - Criar requisição;
- 5 - Realizar requisição.

**Status Code:**

400

**Mensagem de erro esperada:**

Já existe produto com esse nome

**T26 - Falhar ao tentar modificar um produto logado com um token inválido****Descrição:**

Deve falhar ao tentar fazer a modificação de um produto logado com um token inválido, ausente, expirado ou que o usuário não exista mais no sistema

<b>Parâmetro:</b> <pre>{   "nome": " ",   "preco": number,   "descricao": " ",   "quantidade": number }</pre>
<b>Rota:</b> /produtos/{_id}
<b>Método:</b> PUT
<b>Etapas do Teste:</b> 1 - Buscar um token inválido; 2 - Buscar ID de um produto cadastrado no sistema; 3 - Criar requisição; 4 - Realizar requisição.
<b>Status Code:</b> 401
<b>Mensagem de erro esperada:</b> Token de acesso ausente, inválido, expirado ou usuário do token não existe mais

T27 - Falhar ao tentar modificar um produto sem ser administrador
<b>Descrição:</b> Deve falhar ao tentar modificar um produto logado com um usuário que não é administrador
<b>Parâmetro:</b> <pre>{   "nome": " ",   "preco": number,   "descricao": " ",   "quantidade": number }</pre>
<b>Rota:</b> /produtos/{_id}
<b>Método:</b> PUT
<b>Etapas do Teste:</b> 1 - Logar com usuário que não é administrador; 2 - Buscar ID de um produto cadastrado; 3 - Criar requisição; 4 - Realizar requisição.



**Status Code:**

403

**Mensagem de erro esperada:**

Rota exclusiva para administradores

**T28 - Falhar ao tentar deletar produto que esteja em um carrinho****Descrição:**

Deve falhar ao tentar deletar um produto que esteja dentro de um carrinho

**Rota:**

/produtos/{\_id}

**Método:**

DELETE

**Etapas do Teste:**

- 1 - Logar com usuário administrador;
- 2 - Buscar ID de um produto cadastrado que esteja em um carrinho;
- 3 - Criar requisição;
- 4 - Realizar requisição.

**Status Code:**

400

**Mensagem de erro esperada:**

Não é permitido excluir produto que faz parte de carrinho

**T29 - Falhar ao tentar deletar produto logado com um token inválido****Descrição:**

Deve falhar ao tentar deletar um produto logado com um token inválido, ausente, expirado ou que o usuário não exista mais no sistema

**Rota:**

/produtos/{\_id}

**Método:**

DELETE

**Etapas do Teste:**

- 1 - Buscar um token inválido;
- 2 - Buscar ID de um produto cadastrado;
- 3 - Criar requisição;
- 4 - Realizar requisição.

**Status Code:**

401

**Mensagem de erro esperada:**

Token de acesso ausente, inválido, expirado ou usuário do token não existe mais

**T30 - Falhar ao tentar deletar um produto sem ser administrador****Descrição:**

Deve falhar ao tentar deletar um produto logado com um usuário que não é administrador

**Rota:**

/produtos/{\_id}

**Método:**

DELETE

**Etapas do Teste:**

- 1 - Logar com usuário que não é administrador;
- 2 - Buscar ID de um produto cadastrado;
- 3 - Criar requisição;
- 4 - Realizar requisição.

**Status Code:**

403

**Mensagem de erro esperada:**

Rota exclusiva para administradores

**5.4 Testes da rota /carrinhos****T31 - Buscar todos os carrinhos****Descrição:**

Deve buscar todos os carrinhos cadastrados

**Rota:**

/carrinhos

**Método:**

GET

**Etapas do Teste:**

- 1 - Criar requisição;
- 2 - Realizar requisição.

**Status Code:**

200

**T32 - Buscar um carrinho em específico**

<b>Descrição:</b> Deve buscar um carrinho em específico pelo seu ID
<b>Rota:</b> /carrinhos/{_id}
<b>Método:</b> GET
<b>Etapas do Teste:</b> 1 - Buscar o ID de um carrinho; 2 - Criar requisição; 3 - Realizar requisição.
<b>Status Code:</b> 200

T33 - Postar um novo carrinho
<b>Descrição:</b> Deve fazer o cadastro de um novo carrinho no sistema
<b>Parâmetro:</b> <pre>{   "produtos": [     {       "idProduto": " ",       "quantidade": number     },     {       "idProduto": " ",       "quantidade": number     }   ] }</pre>
<b>Rota:</b> /carrinhos
<b>Método:</b> POST
<b>Etapas do Teste:</b> 1 - Logar com um usuário 2 - Buscar ID de um produto; 3 - Criar requisição; 4 - Realizar requisição.
<b>Status Code:</b> 201

**T34 - Concluir compra de um carrinho****Descrição:**

Deve fazer a conclusão da compra de um carrinho

**Rota:**

/carrinhos/concluir-compra

**Método:**

DELETE

**Etapas do Teste:**

- 1 - Logar com um usuário com carrinho cadastrado;
- 2 - Criar requisição;
- 3 - Realizar requisição.

**Status Code:**

200

**T35 - Concluir compra sem que exista um carrinho vinculado ao usuário****Descrição:**

Deve fazer a conclusão da compra de um carrinho sem que o usuário tenha um carrinho vinculado à conta

**Rota:**

/carrinhos/concluir-compra

**Método:**

DELETE

**Etapas do Teste:**

- 1 - Logar com um usuário sem carrinho cadastrado;
- 2 - Criar requisição;
- 3 - Realizar requisição.

**Status Code:**

200

**T36 - Cancelar compra de um carrinho****Descrição:**

Deve fazer o cancelamento da compra de um carrinho

**Rota:**

/carrinhos/cancelar-compra

**Método:**  
DELETE

**Etapas do Teste:**  
1 - Logar com um usuário com carrinho cadastrado;  
2 - Criar requisição;  
3 - Realizar requisição.

**Status Code:**  
200

### T37 - Cancelar compra sem que exista um carrinho vinculado ao usuário

**Descrição:**  
Deve fazer o cancelamento da compra de um carrinho sem que o usuário tenha um carrinho vinculado à conta

**Rota:**  
/carrinhos/cancelar-compra

**Método:**  
DELETE

**Etapas do Teste:**  
1 - Logar com um usuário sem carrinho cadastrado;  
2 - Criar requisição;  
3 - Realizar requisição.

**Status Code:**  
200

### T38 - Falhar ao pesquisar carrinho que não exista

**Descrição:**  
Deve falhar ao tentar fazer a busca de um carrinho que não esteja cadastrado no sistema

**Rota:**  
/carrinhos/{\_id}

**Método:**  
GET

**Etapas do Teste:**  
1 - Buscar um ID inexistente;  
2 - Criar requisição;  
3 - Realizar requisição.

**Status Code:**  
400

**Mensagem de erro esperada:**  
Carrinho não encontrado

### T39 - Falhar ao postar carrinho com produto duplicado

**Descrição:**

Deve falhar ao tentar fazer o cadastro de um carrinho com 2 produtos iguais

**Parâmetro:**

```
{
  "produtos": [
    {
      "idProduto": "abc",
      "quantidade": number
    },
    {
      "idProduto": "abc",
      "quantidade": number
    }
  ]
}
```

**Rota:**

/carrinhos

**Método:**

POST

**Etapas do Teste:**

- 1 - Logar com um usuário;
- 2 - Buscar ID de um produto;
- 3 - Criar requisição;
- 4 - Realizar requisição.

**Status Code:**

400

**Mensagem de erro esperada:**

Não é permitido possuir produto duplicado

### T40 - Falhar ao postar carrinho em usuário que já tenha um carrinho cadastrado

**Descrição:**

Deve falhar ao tentar fazer o cadastro de um novo carrinho para um usuário que já tenha um carrinho cadastrado

**Parâmetro:**

```
{
  "produtos": [
    {
```

<pre>       "idProduto": " ",       "quantidade": number     },     {       "idProduto": " ",       "quantidade": number     }   ] }</pre>
<b>Rota:</b> /carrinhos
<b>Método:</b> POST
<b>Etapas do Teste:</b> 1 - Logar com um usuário que tenha um carrinho cadastrado; 2 - Buscar ID de um produto; 3 - Criar requisição; 4 - Realizar requisição.
<b>Status Code:</b> 400
<b>Mensagem de erro esperada:</b> Não é permitido ter mais de 1 carrinho

T41 - Falhar ao postar carrinho com produto que não exista
<b>Descrição:</b> Deve falhar ao tentar fazer o cadastro de um carrinho cujo produto não esteja cadastrado no sistema
<b>Parâmetro:</b> <pre> {   "produtos": [     {       "idProduto": "ID de produto que não exista",       "quantidade": number     },     {       "idProduto": " ",       "quantidade": number     }   ] }</pre>
<b>Rota:</b> /carrinhos
<b>Método:</b>

POST

**Etapas do Teste:**

- 1 - Logar com um usuário que tenha um carrinho cadastrado;
- 2 - Buscar ID de um produto que não esteja cadastrado no sistema;
- 3 - Criar requisição;
- 4 - Realizar requisição.

**Status Code:**

400

**Mensagem de erro esperada:**

Produto não encontrado

**T42 - Falhar ao postar carrinho com produto sem quantidade o suficiente**

**Descrição:**

Deve falhar ao tentar fazer o cadastro de um carrinho cujo produto não tenha a quantidade que é pedida no parâmetro

**Parâmetro:**

```
{
  "produtos": [
    {
      "idProduto": " ",
      "quantidade": Quantidade a mais que o produto tenha
    },
    {
      "idProduto": " ",
      "quantidade": number
    }
  ]
}
```

**Rota:**

/carrinhos

**Método:**

POST

**Etapas do Teste:**

- 1 - Logar com um usuário;
- 2 - Buscar ID de um produto que não tenha quantidade suficiente;
- 3 - Criar requisição;
- 4 - Realizar requisição.

**Status Code:**

400

**Mensagem de erro esperada:**

Produto não possui quantidade suficiente



#### T43 - Falhar ao tentar postar um carrinho logado com um token inválido

**Descrição:**

Deve falhar ao tentar postar um novo carrinho logado com um token inválido, ausente, expirado ou que o usuário não exista mais no sistema

**Parâmetro:**

```
{
  "produtos": [
    {
      "idProduto": " ",
      "quantidade": number
    },
    {
      "idProduto": " ",
      "quantidade": number
    }
  ]
}
```

**Rota:**

/carrinhos

**Método:**

POST

**Etapas do Teste:**

- 1 - Buscar um token inválido;
- 2 - Buscar ID de um produto;
- 3 - Criar requisição;
- 4 - Realizar requisição.

**Status Code:**

401

**Mensagem de erro esperada:**

Token de acesso ausente, inválido, expirado ou usuário do token não existe mais

#### T44 - Falhar ao concluir compra de carrinho logado com um token inválido

**Descrição:**

Deve falhar ao tentar concluir a compra de um carrinho logado com um token inválido, ausente, expirado ou que o usuário não exista mais no sistema

**Rota:**

/carrinhos/concluir-compra

**Método:**

DELETE

**Etapas do Teste:**

- 1 - Buscar um token inválido;

2 - Criar requisição; 3 - Realizar requisição.
<b>Status Code:</b> 401
<b>Mensagem de erro esperada:</b> Token de acesso ausente, inválido, expirado ou usuário do token não existe mais

<b>T45 - Falhar ao cancelar compra de carrinho logado com um token inválido</b>
<b>Descrição:</b> Deve falhar ao tentar cancelar a compra de um carrinho logado com um token inválido, ausente, expirado ou que o usuário não exista mais no sistema
<b>Rota:</b> /carrinhos/cancelar-compra
<b>Método:</b> DELETE
<b>Etapas do Teste:</b> 1 - Buscar um token inválido; 2 - Criar requisição; 3 - Realizar requisição.
<b>Status Code:</b> 401
<b>Mensagem de erro esperada:</b> Token de acesso ausente, inválido, expirado ou usuário do token não existe mais

## 6. Estratégia de Teste

A estratégia de teste vai ser montar um fluxo principal de testes que vai conter os seguintes testes:

- T03 - Postar um novo usuário;
- T12 - Realizar login com sucesso;
- T33 - Postar um novo carrinho;
- T34 - Concluir compra de um carrinho.

Nos testes que necessitarem de login, o usuário que deverá estar logado é o criado no teste T03.

Após isso, a estratégia será realizar os testes positivos, negativos e do fluxo principal e verificar a sua funcionalidade e se o retorno está de acordo com a documentação da API.

## 7. Priorização de Teste

A priorização dos testes será realizada da seguinte forma:

Prioridade	Tipo de teste
Prioridade 1	Fluxo principal
Prioridade 2	Funcionalidade dos testes positivos
Prioridade 3	Funcionalidade dos testes negativos
Prioridade 4	Verificação se os resultados estão de acordo com a documentação

## 8. Ambiente de testes

Ambiente usado nos testes: ambiente de produção da API

## 9. Candidatos para automação

Os seguintes testes serão automatizados:

- Testes do fluxo principal
- Testes positivos
- Testes negativos da rota /login e /carrinhos

Os testes positivos serão automatizados pois é importante estar sempre funcionando.

Apenas os testes negativos da rota /login e /carrinhos serão automatizados pois foram consideradas as duas rotas mais importantes em relação aos testes negativos.

No total, de 45 testes, 31 deles foram automatizados.

## 10. Ferramentas

As seguintes ferramentas foram utilizadas no decorrer dos testes:

Ferramenta	Onde foi utilizado
Google Drive	Escrita do plano de testes e do relatório de testes

Swagger	Visualização da documentação da API
Visual Studio Code	Realização dos testes automatizados
Postman	Realização dos testes que não foram automatizados
XMind	Criação do mapa mental
Git	Versionamento de código
Github	Postagem do projeto