

Plano de teste

# API ServeRest

Responsável:

Sablina Alves de Carvalho

Compass UOL

Julho de 2022

# Sumário

1. Introdução .....	3
2. Objetivo .....	3
3. Escopo .....	3
4. Mapa mental .....	4
5. Suítes de caso de testes .....	5
5.1 Testes negativos .....	5
5.2 Testes positivos .....	11
6. Estratégia de teste .....	16
7. priorização de teste .....	17
8. Candidatos para automação .....	17
9. Ferramentas .....	17
10. Observações complementares .....	18

## **1. Introdução**

Esse plano de teste é referente a API serverest, que foi desenvolvida com foco em simular uma loja virtual, e fazer disto como uma API para estudos e aprendizado. Ao elaborar este plano de teste, foi usado user story e critérios de aceitação como orientação para o desenvolvimento dos casos de suítes de testes.

## **2. Objetivo**

Todos os testes que serão propostos nesse documento têm como o objetivo de assegurar que a API se comporte conforme o esperado, para que assim possamos relatar os eventos ocorridos durante a execução deste plano de teste, buscando encontrar:

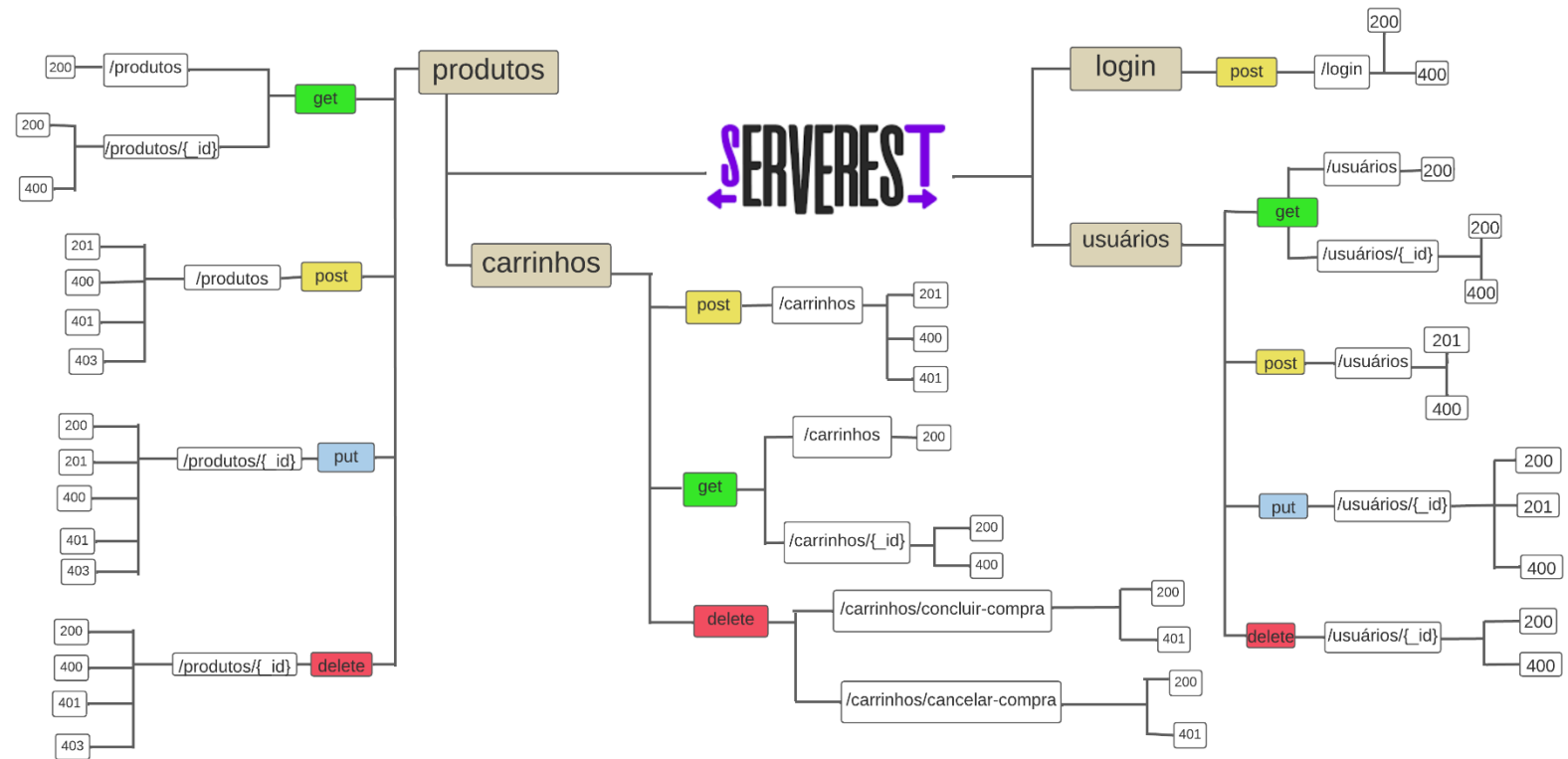
- Possíveis bugs nas rotas de requisição;
- Erros de status code;
- Comportamentos indevidos;
- Implicância no banco de dados;

## **3. Escopo**

Como o objetivo da API é um sistema de compras, é inevitável que durante o processo de criação acabe gerando bugs e falhas. Então, este plano de teste vai buscar esses bugs que foram gerados no caminho.

Esses testes serão importantes para as possíveis aplicações que irão fazer uso desta API, assim evitando transtornos aos usuários que vão usufruir.

## 4. Mapa mental



## 5. Suítes de caso de teste

### 5.1 Testes negativos

#### 5.1.1 Endpoint usuários

Editar usuário por ID		
<b>Endpoint:</b> usuarios		<b>Tipo de requisição:</b> PUT
<b>Rota:</b> /usuarios/{_id}		
<b>Pré-condição:</b> Um usuário já deve está cadastrado		
ID	Descrição	Resultado esperado
001	Editar usuário com e-mail já utilizado	Retorna a mensagem “este email já está sendo usado”
002	Editar usuário com e—mail vazio	Retorna a mensagem “email não pode ficar em branco”
003	Editar usuário com nome vazio	Retornar a mensagem “nome não pode ficar em branco”
004	Editar usuário com o nome e senha vazia	Retornar a mensagem “nome não pode ficar em branco” “password não pode ficar em branco”
Pós-condição: Os testes devem retornar 400 Bad request		

Busca usuário por ID		
<b>Endpoint:</b> usuarios		<b>Tipo de requisição:</b> GET
<b>Rota:</b> /usuarios/{_id}		
<b>Pré-condição:</b> -----		
ID	Descrição	Resultado esperado
001	Buscar usuário com ID inválido	Retorna a mensagem “Usuário não encontrado”
Pós-condição: O teste deve ter retornado 400 Bad request		

Excluir usuário por ID		
<b>Endpoint:</b> usuarios		<b>Tipo de requisição:</b> GET
<b>Rota:</b> /usuarios/{_id}		
<b>Pré-condição:</b> -----		
ID	Descrição	Resultado esperado
001	Deletar usuário por ID inválido	Retorna a mensagem “nenhum registro excluído”
Pós-condição: O teste deve ter retornado o status code: 200 OK		

Cadastrar usuários		
<b>Endpoint:</b> usuarios		<b>Tipo de requisição:</b> POST
<b>Rota:</b> /usuarios		
<b>Pré-condição:</b> para realização do teste de ID 2, um e-mail já deve está cadastrado		
ID	Descrição	Resultado esperado
001	Cadastrar usuário com e—mail vazio	Retorna a mensagem “email não pode ficar em branco”
002	Cadastrar usuário com e-mail já cadastrado	Retorna a mensagem “este email já está sendo usado”
003	Cadastrar usuário com o nome vazio	Retornar a mensagem “nome não pode ficar em branco”
Pós-condição: todos os testes devem ter retornado 400 Bad request		

### 5.1.2 Endpoint login

Realizar login		
<b>Endpoint:</b> login		<b>Tipo de requisição:</b> POST
<b>Rota:</b> /{_id}		
<b>Pré-condição:</b> Necessário ter um usuário cadastrado		
ID	Descrição	Resultado esperado
001	Realizar login com email vazio	Retorna a mensagem “email não pode ficar em branco”
002	Realizar login com senha inválida	Retorna a mensagem “Email e/ou senha inválidos”
003	Realizar login com email inválido	Retorna a mensagem “Email e/ou senha inválidos”
004	Realizar login com senha vazia	Retorna a mensagem “password não pode ficar em branco”
005	Realizar login com senha e email vazios	Retorna a mensagem “email não pode ficar em branco” “password não pode ficar em branco”
Pós-condições: O teste do ID 1 deve ter apresentado um status code de 400 O teste do ID 2 deve ter apresentado um status code de 401 O teste do ID 3 deve ter apresentado um status code de 401 O teste do ID 4 deve ter apresentado um status code de 400 O teste do ID 5 deve ter apresentado um status code de 400		

### 5.1.3 Endpoint produtos

Cadastrar produto		
<b>Endpoint:</b> produtos		<b>Tipo de requisição:</b> POST
<b>Rota:</b> /produtos		
<b>Pré-condição:</b> Necessário ter um usuário cadastrado para realizar login		
ID	Descrição	Resultado esperado
001	Cadastrar produto com nome existente	Retorna a mensagem “Já existe um produto com esse nome”
002	Cadastrar produto sem o token	Retorna a mensagem “Token de acesso ausente, inválido, expirado ou usuário do token não existe mais”
003	Cadastrar produto sem descrição	Retorna a mensagem “descrição não pode ficar em branco”
004	Cadastrar produto sem nome	Retorna a mensagem “nome não pode ficar em branco”
Pós-condição: O teste do id 001 deve ter retornado o status code 400 O teste do id 002 deve ter retornado o status code 401 O teste do id 003 deve ter retornado o status code 400 O teste do id 004 deve ter retornado o status code 400		

Excluir produto por ID		
<b>Endpoint:</b> produtos		<b>Tipo de requisição:</b> DELETE
<b>Rota:</b> /produtos/{_id}		
<b>Pré-condição:</b> Necessário ter um produto cadastrado		
ID	Descrição	Resultado esperado
001	Excluir produto que está no carrinho	Retorna a mensagem “Não é permitido excluir produto que faz parte do carrinho”
002	Excluir produto com token ausente	Retorna a mensagem “Token de acesso ausente, inválido, expirado ou usuário do token não existe mais”
003	Excluir produto sem ser administrador	Retornar a mensagem “Rota exclusiva para administradores”
Pós-condição: O teste do id 001 deve ter retornado o status code 400 O teste do id 002 deve ter retornado o status code 401 O teste do id 003 deve ter retornado o status code 403		



Editar produto por ID		
<b>Endpoint:</b> produtos		<b>Tipo de requisição:</b> PUT
<b>Rota:</b> /produtos/{_id}		
<b>Pré-condição:</b> Necessário ter um produto cadastrado		
ID	Descrição	Resultado esperado
001	Editar produto com nome existente	Retorna a mensagem “Já existe produto com esse nome”
002	Editar produto sem o token	Retorna a mensagem “Token de acesso ausente, inválido, expirado ou usuário do token não existe mais”
003	Editar produto sem ser administrador	Retornar a mensagem “Rota exclusiva para administradores”
Pós-condição: O teste do id 001 deve ter retornado o status code 400 O teste do id 002 deve ter retornado o status code 401 O teste do id 003 deve ter retornado o status code 403		

Buscar produto por ID		
<b>Endpoint:</b> produtos		<b>Tipo de requisição:</b> GET
<b>Rota:</b> /produtos/{_id}		
<b>Pré-condição:</b> -----		
ID	Descrição	Resultado esperado
001	Buscar produto por ID inválido	Retorna a mensagem “Produto não encontrado”
Pós-condição: O teste deve ter retornado o status code 400		

#### 5.1.4 Endpoint carrinhos

Cadastrar carrinho		
<b>Endpoint:</b> carrinho		<b>Tipo de requisição:</b> POST
<b>Rota:</b> /carrinho		
<b>Pré-condição:</b> Necessário ter um usuário cadastrado e um produto		
ID	Descrição	Resultado esperado
001	Cadastrar carrinho com produto inválido	Retorna a mensagem “Produto não encontrado”
002	Cadastrar carinho com token inválido	Retorna a mensagem “Token de acesso ausente, inválido, expirado ou usuário do token não existe mais”
003	Cadastrar carrinho com token vazio	Retorna a mensagem “Token de acesso ausente, inválido, expirado ou usuário do token não existe mais”
Pós-condição: O teste do id 001 deve ter retornado o status code 400 O teste do id 002 deve ter retornado o status code 401 O teste do id 003 deve ter retornado o status code 401		

Buscar carrinho por ID		
<b>Endpoint:</b> carrinhos		<b>Tipo de requisição:</b> GET
<b>Rota:</b> /carrinhos/{_id}		
<b>Pré-condição:</b> -----		
ID	Descrição	Resultado esperado
001	Buscar carrinho por ID inválido	Retorna a mensagem “Carrinho não encontrado”
Pós-condição: O teste deve ter retornado o status code 400		

Excluir carrinho		
<b>Endpoint:</b> carrinhos		<b>Tipo de requisição:</b> DELETE
<b>Rota:</b> /carrinhos/cancelar-compra		
<b>Pré-condição:</b> Necessário ter um usuário com carrinho cadastrado		
ID	Descrição	Resultado esperado
001	Excluir carrinho com token inválido	Retorna a mensagem “Token de acesso ausente, inválido, expirado ou usuário do token não existe mais”
Pós-condição: O teste deve ter retornado o status code 401		

Excluir carrinho		
<b>Endpoint:</b> carrinhos		<b>Tipo de requisição:</b> DELETE
<b>Rota:</b> /carrinhos/concluir-compra		
<b>Pré-condição:</b> Necessário ter um usuário com carrinho cadastrado		
ID	Descrição	Resultado esperado
001	Excluir carrinho com token inválido	Retorna a mensagem “Token de acesso ausente, inválido, expirado ou usuário do token não existe mais”
Pós-condição: O teste deve ter retornado o status code 401		

## 5.2 Testes positivos

### 5.2.1 Endpoint usuários

Cadastrar usuários		
<b>Endpoint:</b> usuarios		<b>Tipo de requisição:</b> POST
<b>Rota:</b> /usuarios		
<b>Pré-condição:</b> -----		
ID	Descrição	Resultado esperado
001	Cadastrar usuários	Retorna a mensagem “Cadastro realizado com sucesso”
Pós-condição: O teste deve ter retornado o status code 201		

Buscar usuário por ID		
<b>Endpoint:</b> usuarios		<b>Tipo de requisição:</b> GET
<b>Rota:</b> /usuarios/{_id}		
<b>Pré-condição:</b> Necessário ter um usuário cadastrado		
ID	Descrição	Resultado esperado
001	Buscar usuário por ID	Retorna as informações do usuário
Pós-condição: O teste deve ter retornado o status code 200		

Deletar usuário por ID		
<b>Endpoint:</b> usuarios		<b>Tipo de requisição:</b> DELETE
<b>Rota:</b> /usuarios/{_id}		
<b>Pré-condição:</b> Necessário ter um usuário cadastrado		
ID	Descrição	Resultado esperado
001	Excluir usuário por ID	Retorna a mensagem “Registro excluído com sucesso”
Pós-condição: O teste deve ter retornado o status code 200		

Listar usuários cadastrados		
<b>Endpoint:</b> usuarios		<b>Tipo de requisição:</b> GET
<b>Rota:</b> /usuarios		
<b>Pré-condição:</b> -----		
ID	Descrição	Resultado esperado
001	Buscar usuários cadastrados	Retorna as informações dos usuários cadastrados
Pós-condição: O teste deve ter retornado o status code 200		

Editar usuário por ID		
<b>Endpoint:</b> usuarios		<b>Tipo de requisição:</b> PUT
<b>Rota:</b> /usuarios/{_id}		
<b>Pré-condição:</b> Necessário ter um usuário cadastrado		
ID	Descrição	Resultado esperado
001	Editar usuário por ID	Retorna a mensagem “Registro alterado com sucesso”
Pós-condição: O teste deve ter retornado o status code 200		

### 5.2.2 Endpoint login

Realizar login		
<b>Endpoint:</b> usuarios		<b>Tipo de requisição:</b> POST
<b>Rota:</b> /usuarios		
<b>Pré-condição:</b> Necessário ter um usuário cadastrado		
ID	Descrição	Resultado esperado
001	Realizar login	Retorna a mensagem "Login realizado com sucesso"
Pós-condição: O teste deve ter retornado o status code 200		

Cadastrar produto		
<b>Endpoint:</b> produtos		<b>Tipo de requisição:</b> POST
<b>Rota:</b> /produtos		
<b>Pré-condição:</b> Necessário obter um token		
ID	Descrição	Resultado esperado
001	Cadastrar produto	Retorna a mensagem "Cadastro realizado com sucesso"
Pós-condição: O teste deve ter retornado o status code 200		

### 5.2.3 Endpoint produto

Listar produtos		
<b>Endpoint:</b> produtos		<b>Tipo de requisição:</b> GET
<b>Rota:</b> /produtos		
<b>Pré-condição:</b> -----		
ID	Descrição	Resultado esperado
001	Listar produtos cadastrados	Retorna a mensagem as informações do produto
Pós-condição: O teste deve ter retornado o status code 200		

Editar produto		
<b>Endpoint:</b> produtos		<b>Tipo de requisição:</b> PUT
<b>Rota:</b> /produtos/{_id}		
<b>Pré-condição:</b> Necessário ter um produto cadastrado		
ID	Descrição	Resultado esperado
001	Editar produto por ID	Retorna a mensagem “Registro alterado com sucesso”
Pós-condição: O teste deve ter retornado o status code 200		

Buscar produto por ID		
<b>Endpoint:</b> produtos		<b>Tipo de requisição:</b> Get
<b>Rota:</b> /produtos/{_id}		
<b>Pré-condição:</b> Necessário ter um produto cadastrado		
ID	Descrição	Resultado esperado
001	Buscar produto por id	Retorna as informações do produto
Pós-condição: O teste deve ter retornado o status code 200		

Excluir produto		
<b>Endpoint:</b> produtos		<b>Tipo de requisição:</b> DELETE
<b>Rota:</b> /produtos/{_id}		
<b>Pré-condição:</b> Necessário ter um produto cadastrado		
ID	Descrição	Resultado esperado
001	Buscar produto por id	Retorna a mensagem “Registro excluído com sucesso”
Pós-condição: O teste deve ter retornado o status code 200		

#### 5.2.4 Endpoint Carrinho

Listar carrinho		
<b>Endpoint:</b> carrinhos		<b>Tipo de requisição:</b> GET
<b>Rota:</b> /carrinhos		
<b>Pré-condição:</b> -----		
ID	Descrição	Resultado esperado
001	Listar carrinhos	Retorna as informações de carrinho
Pós-condição: O teste deve ter retornado o status code 200		

Cadastrar carrinho		
<b>Endpoint:</b> carrinhos		<b>Tipo de requisição:</b> POST
<b>Rota:</b> /carrinhos		
<b>Pré-condição:</b> Necessário ter um usuário e produto cadastrados		
ID	Descrição	Resultado esperado
001	Cadastrar carrinho	Retorna a mensagem “Cadastro realizado com sucesso”
Pós-condição: O teste deve ter retornado o status code 200		

Concluir compra		
<b>Endpoint:</b> carrinhos		<b>Tipo de requisição:</b> DELETE
<b>Rota:</b> /carrinhos/concluir-compra		
<b>Pré-condição:</b> Necessário ter um carrinho cadastrado		
ID	Descrição	Resultado esperado
001	Listar carrinhos	Retorna a mensagem “Registro excluído com sucesso”
Pós-condição: O teste deve ter retornado o status code 200		

Excluir carrinho		
<b>Endpoint:</b> carrinhos		<b>Tipo de requisição:</b> DELETE
<b>Rota:</b> /carrinhos/cancelar-compra		
<b>Pré-condição:</b> Necessário ter um carrinho		
ID	Descrição	Resultado esperado
001	Excluir carrinho	Retorna a mensagem “Registro excluído com sucesso. Estoque dos produtos reabastecido”
Pós-condição: O teste deve ter retornado o status code 200		

Buscar carrinho		
<b>Endpoint:</b> carrinhos		<b>Tipo de requisição:</b> GET
<b>Rota:</b> /carrinhos		
<b>Pré-condição:</b> Necessário ter um carrinho cadastrado		
ID	Descrição	Resultado esperado
001	Buscar carrinho por ID	Retorna as informações de todos os carrinhos
Pós-condição: O teste deve ter retornado o status code 200		

## 6. Estratégias de teste

### 6.2 Ambiente de teste

O ambiente principal para realizar as suítes de teste é em um ambiente local, sendo assim, será necessário rodar a API ServeRest na máquina onde será realizado os testes. Para melhor desempenho durante execução das suítes de teste, é desejável que a máquina escolhida apresente as seguintes configurações:







- Memória de 8GB ou superior;
- Core i5 ou superior;



- A partir do Windows 8.

## 7. Priorização de teste

Todos os endpoints e suas rotas fazem parte da suíte de teste, mas para cadastrar um produto, criar carrinho e concluir compra é necessário realizar um cadastro de usuário e obter um token a partir do login, então segue a priorização de teste:

Alta prioridade =  Média prioridade =  Baixa prioridade = 	Priorização dos testes		
			
Cadastrar usuário	X		
Realizar login	X		
Cadastrar carrinho		X	
Concluir compra		X	

Os demais casos de teste podem ser considerados de baixa prioridade, visto que o ciclo principal da API é:

Realizar cadastro -> Realizar login -> Cadastrar carrinho -> Concluir compra

## 8. Candidatos para automação

Todos os testes são candidatos para automação, dessa forma, na medida que novas rotas e endpoints fossem acrescentadas seria muito complexo manter os testes manuais, além de se tornarem exaustivos e ter uma perda de tempo considerável.

## 9. Ferramentas

A ferramenta para realizar os testes é o Postman, uma vez que é conseguimos:

- Criar scripts de automação;
- Compartilhar o ambiente para trabalho em equipe;
- Documentar API.

## **10. Considerações finais**

Este plano de teste foi desenvolvido pesando nos usuários que irão fazer uso da API serveRest, para que tenham a melhor experiência possível.

**Obs:** A medida que for surgindo novas rotas e endpoints, será adicionado mais testes nas suítes de casos.