

# Plano de Teste

**Projeto:** ServeRest 2.29.7

## 1. Objetivo

Garantir a qualidade da aplicação, validando as regras de negócio e o funcionamento das APIs REST, de modo que atendam aos requisitos especificados e mantenham a integridade.

## 2. Escopo

Os testes contemplarão as funcionalidades essenciais da aplicação expostas pelas APIs REST, com foco nos fluxos críticos para o negócio.

**Recursos a serem Testados:**

- **/login**
  - POST
    - Fazer Login
- **/usuários**
  - GET
    - Listar Usuários Cadastrados
  - POST
    - Cadastrar Usuários
  - PUT
    - Editar Usuário
  - DELETE
    - Deletar Usuário
- **/produtos**
  - GET
    - Listar Produtos Cadastrados
  - POST
    - Cadastrar Produto
  - PUT
    - Editar Produto
  - DELETE
    - Deletar Produto
- **/carrinhos**
  - GET
    - Buscar Carrinho
    - Buscar Carrinho por ID
  - POST
    - Cadastrar Carrinho
  - DELETE
    - Concluir Compra
    - Cancelar Compra

### 3. Análise

#### Visão Geral do Produto

O ServeRest é uma API REST de um e-commerce, permitindo:

- Cadastro e autenticação de **administradores** e **clientes**.
- Operação de **produtos** e **carrinho**.

#### Foco dos Testes

Garantir que as **regras de negócios** sejam validadas.

### 4. Metodologias:

- [Mapa Mental](#)
- SBTM

#### Estratégia de Teste:

- Teste de validação de regras de negócio
- Heurística CRUD

#### Recursos:

- **Equipe:** 1 Bolsista de Quality Engineer
- **Ferramentas:** Postman

### 5. Cenários de testes planejados

[Tabela com cenários de teste:](#)

CENÁRIO	CASO DE TESTE	Descrição	STATUS
001- Fazer cadastro como um vendedor	CT001.001 - Cadastro de um email inválido		PASSOU
	CT001.002 - Cadastro de um e-mail do provedor Gmail	De acordo com a documentação não deve ser cadastrado um usuário com um e-mail do domínio Gmail	FALHOU
	CT001.003 - Cadastro de um e-mail do provedor Hotmail	De acordo com a documentação não deve ser cadastrado um usuário com um e-mail do domínio Hotmail	FALHOU
	CT001.004 - Cadastro com um email já utilizado	Não deve ser possível cadastrar um e-mail já utilizado	PASSOU

CT001.005 - Cadastro com uma senha de tamanho inválido

De acordo com a documentação uma senha deve ter no mínimo 5 caracteres e no máximo 10 caracteres

FALHOU

CT001.006 - Cadastro com campos ausentes

PASSOU

## 6. Priorização

Do ponto de vista de negócios, falhas nestes fluxos podem impactar diretamente a receita, a experiência do usuário e a credibilidade do sistema. Dado essa análise os testes foram priorizados da seguinte forma.

- **Alta Prioridade:**
  - Cadastro e login de clientes e vendedores
  - CRUD de produtos
  - Validações de negócios
- **Média Prioridade:**
  - Edição de usuários
- **Baixa Prioridade:**
  - Buscar usuário por ID

## 7. Matriz de risco

RISCO	Probabilidade	Impacto	Fator de Risco
Cadastro aceita e-mail inválido	5	4	20
Carrinho aceita a compra sem ter estoque	4	5	20
API deleta um produto que está em um carrinho	4	4	16
Cadastrar um produto com um nome já utilizado	4	5	20
Quantidade do estoque não ser modificada ao concluir ou cancelar compra	4	5	20
Um cliente atualiza os dados de um produto	3	5	15
API deletar um usuário com carrinho	3	3	9
Falha ao encontrar um carrinho pelo ID	2	3	6

## 8. Cobertura de testes

### Operator Coverage (input)

Confere a cobertura de testes de todos os métodos existentes na API REST (GET, POST, PUT, DELETE...).

### Endpoints automatizados:

/login - POST

C3S-5 Fazer Login com uma credencial não cadastrada

C3S-6 Fazer Login com credenciais válidas

/usuarios - POST

C32-10 Fazer cadastro com credenciais inválidas

C3S-24 Cadastrar um usuário com Gmail

C3S-25 Cadastrar um usuário com Hotmail

C3S-9 Cadastro com um e-mail já utilizado

C3S-25 Cadastrar um Cliente com e-mail utilizado por um Administrador

C3S-26 Cadastrar um Administrador com e-mail utilizado por um Cliente

/produtos - POST

C3S-31 Cadastrar Produto Válido

C3S-12 Cadastrar um produto com um nome já utilizado

C3S-30 Cadastrar Produto como Cliente

/produtos - PUT

C3S-17 Editar um Produto sem Token de Autenticação

C3S-16 Editar um Produto com Token de Cliente

**COBERTURA: 56%**

## 9. Testes candidatos à automação

- CRUD de produtos (fluxo principal).
- Cadastro de usuários (Diferentes cenários)
- Validação de senha (limites e tamanhos).
- Restrições do e-mail no cadastro.

## 10. Sessões de Teste

### Ficha de Sessão de Teste (SBTM)

#### Visão Geral da Sessão

- ID da Sessão: SESSÃO - 001
  - Testador(a): Rafael Soares
  - Data: 11/09/2025
  - Início: 15:00
  - Fim: 16:00
  - Duração Planejada: 60 minutos
  - Duração Real: 32 minutos
  - Técnica Aplicada: Valor limite, Teste Negativo
- 

#### Charter

##### ***Será testado: Cadastro de um usuário:***

CT001.001 - Cadastro de um email inválido  
CT001.002 - Cadastro de um e-mail do provedor Gmail  
CT001.003 - Cadastro de um e-mail do provedor Hotmail  
CT001.004 - Cadastro com um email já utilizado  
CT001.005 - Cadastro com uma senha de tamanho inválido  
CT001.006 - Cadastro com campos ausentes  
CT001.029 - Editar um usuário

***Com o objetivo de:*** verificar como o sistema se comporta

---

## Ambiente e Configuração

- **Ambiente de Teste:** Ambiente de Produção
  - **Versão do Software:** 2.29.7
  - **Navegador / Plataforma:** Postman
  - **Sistema Operacional:** Windows 10 Home
- 

## Tarefas executadas

### A. API de Cadastro de Usuários:

#### 1. Cadastro de usuário com e-mail inválido.

##### Ação:

Foi enviada uma requisição POST para `/usuarios` com os seguintes dados:

```
{
  "nome": "Fulano da Silva",
  "email": "beltrano@aa",
  "password": "teste",
  "administrador": "true"
}
```

##### Observação:

O sistema rejeitou a requisição, retornando o **código 400** com a mensagem:

`"email deve ser um email válido"`

##### Resultado esperado:

Falha no cadastro, pois o formato do email é inválido.

#### 2. Cadastro de usuário com provedor de e-mail não permitido(Gmail e Hotmail).

##### Ação:

Foi enviada duas requisição POST para `/usuarios` com os seguintes dados:

```
{
  "nome": "Google da Silva",
  "email": "Google@gmail.com",
  "password": "teste",
}
```

```
    "administrador": "true"
  }
  {
    "nome": "Microsoft da Silva",
    "email": "Microsoft@hotmail.com",
    "password": "teste",
    "administrador": "true"
  }
}
```

**Observação:**

O sistema aprovou a requisição, cadastrando o usuário e retornando o código 201 com a mensagem:

```
{
  "message": "Cadastro realizado com sucesso,
  "_id": "jogf0DIILXsqxNFS2"
}
```

**Resultado esperado:**

Falha no cadastro, pois esses provedores não são permitidos.

### 3. Cadastrar um usuário com um e-mail já utilizado.

**Ação:**

Foi enviada uma requisição POST para `/usuarios` com os seguintes dados:

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

**Observação:**

O sistema rejeitou a requisição, e retornando o código 400 com a mensagem:

```
{
  "message": "Este email já está sendo usado"
}
```

**Resultado esperado:**

Falha no cadastro, pois esse e-mail já está em uso.

### 4. Cadastrar um usuário com uma senha fora do tamanho especificado.

**Ação:**

Foi enviada duas requisição POST para `/usuarios` com os seguintes dados:

```
{
  "nome": "Fulano da Silva",
  "email": "fulano@qa.com.br",
  "password": "123",
  "administrador": "true"
}
{
  "nome": "Fulano da Silva",
  "email": "fulano@qa.com.br",
  "password": "12345678900",
  "administrador": "true"
}
```

**Observação:**

O sistema aceitou as requisições, cadastrando o usuário e retornando status 201 com a mensagem:

```
{
  "message": "Cadastro realizado com sucesso,
  "_id": "jogf0DIIXsqxNFS2"
}
```

**Resultado esperado:**

O sistema deveria rejeitar a requisição e retornar status 400, já que na documentação está especificado que o tamanho da senha deve ser  $\geq 3$  e  $\leq 11$  caracteres.

## 5. Cadastrar um usuário com campos ausentes.

**Ação:**

Foi enviada duas requisição POST para `/usuarios` com os seguintes dados:

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

**Observação:**

O sistema rejeitou as requisições, e retornando status 400 com a mensagem:



```

{
  "nome": "nome não pode ficar em branco",
  "email": "email não pode ficar em branco",
  "password": "password é obrigatório",
  "administrador": "administrador deve ser 'true' ou
'false'"
}

```

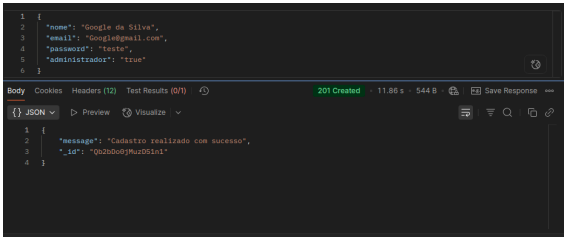
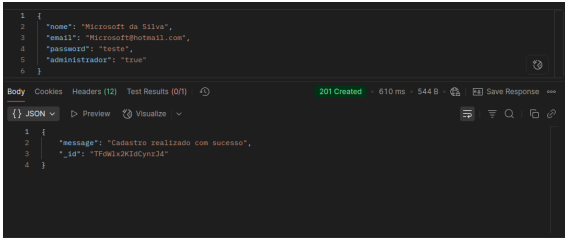
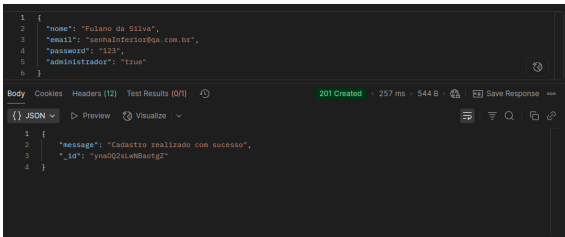
### Resultado esperado:

Falha no cadastro pois não deve ser possível é obrigatório ter todos campos preenchidos.

## Resultados e Resumo

### Bugs Encontrados

Listagem dos defeitos formalmente registrados.

ID	Título/Resumo	Prioridade	Imagem
[BUG-001]	API aceita a criação de um usuário com o provedor Gmail	[Média]	
[BUG-002]	API aceita a criação de um usuário com o provedor Hotmail	[Média]	
[BUG-003]	API permite cadastro de usuário com senha fora do limite definido	[Alta]	

## 7. 5. Problemas, Riscos e Questões

- **[QUESTÃO #1]:** A restrição dos provedores de e-mail Gmail e Hotmail é para todos usuários ou apenas os vendedores?
- **[QUESTÃO #2]:** É possível editar um usuário sem autenticação, isso pode ser um grave problema de segurança.

## Ficha de Sessão de Teste (SBTM)

### Visão Geral da Sessão

- ID da Sessão: SESSÃO - 002
  - Testador(a): Rafael Soares
  - Data: 11/09/2025
  - Início: 15:00
  - Fim: 16:00
  - Duração Planejada: 60 minutos
  - Duração Real: 16 minutos
  - Técnica Aplicada: Teste baseado em regras de negócio
- 

### Charter

***Será testado: Login de usuário***

CT002.001 - Login com e-mail e/ou senha não cadastrado

CT002.002 - Login com e-mail e senha válidos

CT002.003 - Login com e-mail e/ou senha inválidos

***Com o objetivo de: validar a API de Login***

---

### Ambiente e Configuração

- Ambiente de Teste: Ambiente de Produção
  - Versão do Software: 2.29.7
  - Navegador / Plataforma: Postman
  - Sistema Operacional: Windows 10 Home
- 

### Tarefas executadas

#### A. API de Login de Usuários:

##### 1. Login com e-mail e/ou senha não cadastrados

#### Ação:

Foi enviada uma requisição POST para `/login` com os seguintes dados:

```
{
  "email": "nãoCadastrado@teste.com"
  "password": "naoCadastrada"
}
```

**Observação:**

O sistema rejeitou a requisição, retornando o **código 401** com a mensagem:

```
{
  "message": "Email e/ou senha inválidos"
}
```

**Resultado esperado:**

Falha no login pois os dados não estão cadastrados.

## 2. Login com e-mail e senha inválidos

**Ação:**

Foi enviada uma requisição POST para `/login` com os seguintes dados:

```
{
  "email": "!!!!!!_Ç@!_!@_!@_!",
  "password": "!L)DL!_@_!_!_!"
}
```

**Observação:**

O sistema rejeitou a requisição, retornando o **código 400** com a mensagem:

```
{
  "email": "email deve ser um email válido"
}
```

**Resultado esperado:**

Falha no login pois os dados são inválidos.

---

**Resultados e Resumo**

**Bugs Encontrados**

*Listagem dos defeitos formalmente registrados.*

ID	Título/Resumo	Prioridade	Imagem

**Problemas, Riscos e Questões**

## Ficha de Sessão de Teste (SBTM)

### Visão Geral da Sessão

- ID da Sessão: SESSÃO - 003
  - Testador(a): Rafael Soares
  - Data: 11/09/2025
  - Início: 16:30
  - Fim: 17:30
  - Duração Planejada: 60 minutos
  - Duração Real: 19 minutos
  - Técnica Aplicada: Teste baseado em regras de negócio
- 

### Charter

***Será testado: Gerenciamento de produtos***

CT005.001 - Cadastrar um produto com um nome já utilizado  
CT005.002 - Excluir um produto que está dentro de um carrinho  
CT005.003 - Criar um produto novo caso não exista um ID do produto  
CT005.004 - Atualizar um produto sendo cliente

***Com o objetivo de: validar as regras de negócios no endpoint /produtos***

---

### Ambiente e Configuração

- Ambiente de Teste: Ambiente de Produção
  - Versão do Software: 2.29.7
  - Navegador / Plataforma: Postman
  - Sistema Operacional: Windows 10 Home
-

## Tarefas executadas

### A. Endpoint de Produtos:

#### 1. Cadastrar um produto com um nome já utilizado

##### Ação:

Foi enviada uma requisição POST para `/produtos` com os seguintes dados:

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

##### Observação:

O sistema rejeitou a requisição, retornando o código **400** com a mensagem:

```
{
  "message": "Já existe produto com esse nome"
}
```

##### Resultado esperado:

Falha ao cadastrar pois já existe um produto com esse nome.

#### 2. Apagar um produto que está dentro de um carrinho

##### Ação:

Foi enviada uma requisição DELETE para `/produtos/:_id` com o id de um produto que estava presente em um carrinho:

##### Observação:

O sistema rejeitou a requisição, retornando o código **400** com a mensagem:

```
{
  "message": "Não é permitido excluir produto que faz parte
de carrinho",
  "idCarrinhos": [
    "9uSQn5LdgY27Lh8J",
    "qbMqntef4iT0wWfg"
  ]
}
```

**Resultado esperado:**

A requisição deveria ser rejeitada pois o produto está dentro de um carrinho.

**3. Editar um produto com sem permissão****Ação:**

Foi enviada uma requisição PUT para `/produtos/:_id` com os seguintes dados:

1. Utilizando token de autenticação de um usuário cliente

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

2. Sem autenticação

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

**Observação:**

O sistema rejeitou as duas requisições, retornando o código 403 e 401 respectivamente com a mensagem:

caso 1

```
{
  "message": "Rota exclusiva para administradores"
}
```

caso 2

```
{
  "message": "Token de acesso ausente, inválido, expirado ou usuário do token não existe mais"
}
```

**Resultado esperado:**

A requisição deveria ser rejeitada pois é um endpoint exclusivo para administradores.



#### 4. Editar um produto com um id novo

##### Ação:

Foi enviada uma requisição PUT para `/produtos/:_id` com os seguintes dados:

1. Utilizando token de autenticação de um usuário cliente

```
{
  "nome": "Logitech MX Vertical 2",
  "preco": 470,
  "descricao": "Produto gerado através do PUT",
  "quantidade": 1
}
```

##### Observação:

O sistema aceitou a requisição, retornando o **código 201** com a mensagem:

```
{
  "message": "Cadastro realizado com sucesso",
  "_id": "BKbi035qUXFklgTC"
}
```

##### Resultado esperado:

Caso não seja encontrado o produto com a ID informada um novo produto é cadastrado.

#### 5. Adicionar um novo carrinho a um usuário que já possui um carrinho

##### Ação:

Foi enviada uma requisição POST para `/carrinhos` com os seguintes dados:

2. Utilizando token de autenticação de um usuário cliente

```
{
  "produtos": [
    {
      "idProduto": "BeeJh51z3k6kSIzA",
      "quantidade": 3
    }
  ]
}
```

##### Observação:

O sistema rejeitou a requisição, retornando o **código 400** com a mensagem:

```
{
```

```
"message": "Não é permitido ter mais de 1 carrinho"
}
```

**Resultado esperado:**

Caso o usuário já tenha um carrinho deve ser cancelado a criação de outro carrinho para o usuário;

---

**Resultados e Resumo****Bugs Encontrados**

*Listagem dos defeitos formalmente registrados.*

ID	Título/Resumo	Prioridade	Imagem

**Problemas, Riscos e Questões****11. Mapeamento de Issues**

Link para o Jira do Projeto para acompanhamento:

<https://rafaelsoares-pb.atlassian.net/jira/software/projects/C3S/summary?atlOrigin=eyJ...>