

# Plano de teste - API ServeRest

## 1. Apresentação

Este documento apresenta o **planejamento revisado** de testes para a API ServeRest. O objetivo é validar as funcionalidades de usuários, login, produtos e carrinhos, com **cobertura ampliada** e **abordagem orientada à automação** para garantir a qualidade contínua da aplicação.

## 2. Objetivo

Melhorias nesta versão:

- Padronização de nomenclatura dos cenários (CT+ nome descritivo).
- Inclusão de evidências de execução por meio de relatórios Robot Framework (Log.html, Report.html) e prints automáticos de resposta.
- Aumento da cobertura para endpoints antes não testados (carrinhos, concluir e cancelar compra).
- Priorização baseada em risco e impacto para execução contínua (CI/CD).

## 3. Escopo

Inclui as rotas e funcionalidades:

- CRUD Usuários (Cadastro de vendedores)
- Autenticação (Login)
- CRUD Produtos
- **Gestão de Carrinhos** (criar, listar, concluir compra e cancelar compra)

## 4. Análise

- Base: documentação Swagger e User Stories (US001, US002, US003).
- Cenários incluem **casos alternativos** (e-mail inválido, token expirado, carrinho com produtos indisponíveis).
- Inclusão de **validação de status code + payload** para cada teste.

## 5. Técnicas Aplicadas

- Testes funcionais baseados em requisitos e User Stories
- Análise de valores limite e validação de dados

- Testes baseados em riscos para priorização
- Testes exploratórios para casos alternativos
- Testes automatizados de API com Robot Framework + RequestsLibrary.
- Validação de contrato JSON usando biblioteca JSONSchemaLibrary.
- Evidência automatizada (logs, prints de resposta).

## 6. Mapa Mental da Aplicação

 [https://miro.com/app/board/uXjVJbYJy6E=/?share\\_link\\_id=166741494127](https://miro.com/app/board/uXjVJbYJy6E=/?share_link_id=166741494127) Conectar a conta do Miro

## 7. Cenários de Teste Planejados

ID	US Ref.	Cenário	Entrada / Pré-condição	Resultado Esperado	Status Code
CT001	US001	Criar usuário válido	Nome, e-mail válido, senha 5-10 caracteres	Usuário criado com ID	201
CT002	US001	Criar usuário com e-mail duplicado	E-mail já cadastrado	Este email já está sendo usado	400
CT003	US001	Criar usuário com provedor não permitido	E-mail @gmail.com	Erro provedor inválido	400
CT004	US001	Criar usuário com provedor	E-mail @hotmail.com	Erro provedor inválido	400

		não permitido			
CT005	US001	Atualizar usuário inexistente (PUT cria novo)	ID inexistente	Novo usuário criado	201
CT006	US001	Deletar usuário inexistente	ID inválido	Nenhum registro excluído	200
CT007	US001	Listar usuários	-	Retorna lista	200
CT008	US002	Login válido	Usuário válido	Token gerado	200
CT009	US002	Login com usuário inexistente	E-mail inválido	Email e/ou senha inválidos	401
CT010	US002	Login com senha incorreta	Senha errada	Email e/ou senha inválidos	401
CT011	US003	Criar produto válido	Nome único	Produto criado	201
CT012	US002	Acesso com token expirado (Cadastrando produto)	Token expirado	Token ausente, inválido ou expirado	401
CT013	US003	Criar produto duplicado	Nome já existente	Já existe produto com esse nome	400

CT014	US003	Criar produto sem autenticação	Sem token	Token de acesso ausente, inválido, expirado ou usuário do token não existe mais	401
CT015	US003	Atualizar produto inexistente	ID inexistente	Cadastro realizado com sucesso	201
CT016	US003	Excluir produto em carrinho	Produto vinculado	Erro exclusão	400
CT017	US003	Listar produtos	-	Lista produtos	200
CT018	US004	Criar carrinho válido	Produto disponível + token	Carrinho criado	201
CT019	US004	Concluir compra com carrinho válido	Carrinho aberto	Registro excluído com sucesso	200
CT020	US004	Cancelar compra com carrinho válido	Carrinho aberto	Registro excluído com sucesso. Estoque dos	200

				produtos reabasteci do
--	--	--	--	------------------------------

## 8. Priorização da Execução

Prioridade	Casos de Teste	Justificativa
Alta	CT001, CT007, CT011, CT013, CT015, CT017, CT018	Casos críticos de criação, login e fluxo de compra
Média	CT002, CT004, CT005, CT008, CT009, CT010, CT012, CT014, CT019	Validações e fluxos alternativos
Baixa	CT006, CT016	Consultas simples, baixo impacto

## 9. Matriz de Risco

Risco	Probabilidad e	Impacto	Nível	Mitigação
Usuário com e-mail duplicado	Alta	Alto	Alto	Validação rigorosa de cadastro
Login com credenciais inválidas	Alta	Alto	Alto	Mensagens claras e bloqueios
Produto criado com nome duplicado	Média	Alto	Alto	Validação de nome no backend

Exclusão de produto em carrinho ativo	Baixa	Alto	Médio	Bloqueio de exclusão pelo sistema
---------------------------------------	-------	------	-------	-----------------------------------

## 10. Cobertura de Testes

O cálculo da cobertura de teste foi feito a partir do método path Coverage, que verifica a cobertura da suíte de testes de acordo com os endpoints que a API possui. Onde a análise é realizada pela quantidade de URI( URL + URN (*Resource name*)) que a API possui, e se for a mesma URI para métodos diferentes, considera-se apenas um, onde o ideal é realizar ao menos uma requisição para verificar cada endpoint.

- A API ServeRest Possui 9 Endpoints, sendo eles: **/Login, /usuarios, /usuarios/{\_id}, /produtos, /produtos/{\_id}, /carrinhos, /carrinhos/{\_id}, /carrinhos/concluir-compra, /carrinhos/cancelar-compra.**
- Testes Realizados:
- Criar usuários; criar usuário duplicado; criar usuário com Gmail; deletar usuário inexistente; Listar Usuários; Fazer login com usuário existente; Fazer login com usuário inexistente; Fazer login com senha incorreta; Acesso com token expirado; Criar produto válido; Criar produto com nome duplicado; Criar produto sem autenticação; Atualizar produto inexistente (PUT cria novo); Excluir produto em carrinho;

### Endpoints testados diretamente:

- **/login** → testado (vários testes de login)
- **/usuarios** → testado (criação, listagem)
- **/usuarios/{\_id}** → testado (deletar usuário inexistente)
- **/produtos** → testado (criar produtos, autenticação, nome duplicado)
- **/produtos/{\_id}** → testado (atualizar produto inexistente, excluir produto em carrinho)
- **/carrinhos** → **não testado diretamente** (apenas usado para verificar vínculo produto)

### Endpoints testados vs total

- Total de endpoints: 9
- Endpoints com testes diretos realizados: 5
- Endpoints sem testes diretos: 4

## Calculo de cobertura Path Coverage (endpoints)

Cobertura= Numero de endpoints testados/Total de endpoints×100 = $5/9 \times 100 \approx 55.56\%$

### Resultado:

Métrica	Valor
Total de endpoints	9
Endpoints testados	5
Endpoints não testados	4
Cobertura Path Coverage (%)	55,56%

## Cobertura de Testes (Revisada)

- **Total de endpoints:** 9
- **Endpoints testados:** 9
- **Cobertura Path Coverage:** 100% (melhoria em relação aos 55,56% anteriores)

## 11. Testes Candidatos à Automação

- Criação/atualização/exclusão de usuários
- Login com token
- Criação/atualização/exclusão de produtos
- Testes de permissão e autenticação
- Relatórios automáticos

## 12. Evidências de Execução

- Cada teste gera:
  - Log detalhado ( `Log.html` )
  - Relatório resumido ( `Report.html` )

## CT001:

**TEST** CT001: Cadastrar novo usuário com sucesso

Full Name: Test Serverest.CT001: Cadastrar novo usuário com sucesso  
Tags: POST, USUARIOS  
Start / End / Elapsed: 20250911 17:35:19.663 / 20250911 17:35:19.955 / 00:00:00.292  
Status: **PASS**

- **KEYWORD** \${resp} = `test_serverest.Cadastrar_usuario_valido 201` 00:00:00.261  
Start / End / Elapsed: 20250911 17:35:19.672 / 20250911 17:35:19.933 / 00:00:00.261
- + **KEYWORD** \${json} = `common.importar_json_estatico json_usuarios.json` 00:00:00.037
- + **KEYWORD** \${body} = `Collections.Get.From.Dictionary ${json} usuario_valido` 00:00:00.001
- + **KEYWORD** \${email\_random} = `String.Generate.Random.String 8 [LOWER]` 00:00:00.002
- + **KEYWORD** `Collections.Set.To.Dictionary ${body} email ${email_random}@robot.com` 00:00:00.001
- **KEYWORD** \${resposta} = `RequestLibrary.POST.On.Session serverest /usuarios json=${body} expected_status=${status_esperado}` 00:00:00.214  
Documentation: Sends a POST request on a previously created HTTP Session.  
Start / End / Elapsed: 20250911 17:35:19.718 / 20250911 17:35:19.932 / 00:00:00.214  
17:35:19.916 **[INFO]** POST Request : url=https://serverest.dev/usuarios  
path\_url:/usuarios  
headers:{'User-Agent': 'python-requests/2.32.5', 'Accept-Encoding': 'gzip, deflate', 'Accept': '/\*', 'Connection': 'keep-alive', 'Content-Type': 'application/json; charset=utf-8'}  
body:{'name': 'Andressa Mota', 'email': 'sxjrtmj@robot.com', 'password': 'Senha123', 'administrador': 'true'}
- 17:35:19.917 **[INFO]** POST Response : url=https://serverest.dev/usuarios  
status=201, reasonCreated  
headers:{'access-control-allow-origin': '\*', 'x-dns-prefetch-control': 'off', 'x-frame-options': 'SAMEORIGIN', 'strict-transport-security': 'max-age=15552000; includeSubDomains', 'x-download-options': 'noopener', 'x-content-type-options': 'nosniff', 'x-xss-protection': '1; mode=block', 'content-type': 'application/json; charset=utf-8', 'x-cloud-trace-context': '39918344efdfaf38fd20fe75d258e91', 'date': 'Thu, 11 Sep 2025 20:35:19 GMT', 'server': 'Google Frontend', 'Content-Length': '82'}  
body:  
{"message": "Cadastro realizado com sucesso",  
"\_id": "cjqQWIMPJHUEInOB"}

## CT002:

**TEST** CT002: Cadastrar usuário com e-mail duplicado

Full Name: Test Serverest.CT002: Cadastrar usuário com e-mail duplicado  
Tags: POST, USUARIOS  
Start / End / Elapsed: 20250911 17:35:19.964 / 20250911 17:35:20.397 / 00:00:00.433  
Status: **PASS**

- **KEYWORD** \${resp} = `test_serverest.Cadastrar_usuario_duplicado 400` 00:00:00.424  
Start / End / Elapsed: 20250911 17:35:19.967 / 20250911 17:35:20.391 / 00:00:00.424
- + **KEYWORD** \${json} = `common.importar_json_estatico json_usuarios.json` 00:00:00.009
- + **KEYWORD** \${body} = `Collections.Get.From.Dictionary ${json} usuario_duplicado` 00:00:00.001
- + **KEYWORD** `RequestLibrary.POST.On.Session serverest /usuarios json=${body} expected_status=any` 00:00:00.202
- **KEYWORD** \${resposta} = `RequestLibrary.POST.On.Session serverest /usuarios json=${body} expected_status=${status_esperado}` 00:00:00.206  
Documentation: Sends a POST request on a previously created HTTP Session.  
Start / End / Elapsed: 20250911 17:35:20.182 / 20250911 17:35:20.388 / 00:00:00.206  
17:35:20.384 **[INFO]** POST Request : url=https://serverest.dev/usuarios  
path\_url:/usuarios  
headers:{'User-Agent': 'python-requests/2.32.5', 'Accept-Encoding': 'gzip, deflate', 'Accept': '/\*', 'Connection': 'keep-alive', 'Content-Type': 'application/json; charset=utf-8', 'Content-Length': '119'}  
body:{'name': 'Usuario Duplicado', 'email': 'usuario\_duplicado@teste.com', 'password': 'Senha123', 'administrador': 'false'}
- 17:35:20.385 **[INFO]** POST Response : url=https://serverest.dev/usuarios  
status=400, reason=Bad Request  
headers:{'access-control-allow-origin': '\*', 'x-dns-prefetch-control': 'off', 'x-frame-options': 'SAMEORIGIN', 'strict-transport-security': 'max-age=15552000; includeSubDomains', 'x-download-options': 'noopener', 'x-content-type-options': 'nosniff', 'x-xss-protection': '1; mode=block', 'content-type': 'application/json; charset=utf-8', 'x-cloud-trace-context': 'e60c69eca7695169fd20fe75d258e6d', 'date': 'Thu, 11 Sep 2025 20:35:20 GMT', 'server': 'Google Frontend', 'Content-Length': '53'}  
body:  
{"message": "Este email já está sendo usado"}

## CT003:

**TEST** CT003: Cadastrar usuário com provedor bloqueado (gmail)

Full Name: Test Serverest.CT003: Cadastrar usuário com provedor bloqueado (gmail)  
Tags: POST, USUARIOS  
Start / End / Elapsed: 20250911 17:35:20.400 / 20250911 17:35:20.863 / 00:00:00.463  
Status: **FAIL**  
Message: Url: <https://serverest.dev/usuarios> Expected status: 201!= 400

- **KEYWORD** \${resp} = `test_serverest.Cadastrar_usuario_provedor_bloqueado_gmail 400` 00:00:00.404  
Start / End / Elapsed: 20250911 17:35:20.402 / 20250911 17:35:20.806 / 00:00:00.404
- + **KEYWORD** \${json} = `common.importar_json_estatico json_usuarios.json` 00:00:00.005
- + **KEYWORD** \${body} = `Collections.Get.From.Dictionary ${json} usuario_${provedor}` 00:00:00.001
- **KEYWORD** \${resposta} = `RequestLibrary.POST.On.Session serverest /usuarios json=${body} expected_status=${status_esperado}` 00:00:00.368  
Documentation: Sends a POST request on a previously created HTTP Session.  
Start / End / Elapsed: 20250911 17:35:20.410 / 20250911 17:35:20.778 / 00:00:00.368  
17:35:20.610 **[INFO]** POST Request : url=https://serverest.dev/usuarios  
path\_url:/usuarios  
headers:{'User-Agent': 'python-requests/2.32.5', 'Accept-Encoding': 'gzip, deflate', 'Accept': '/\*', 'Connection': 'keep-alive', 'Content-Type': 'application/json', 'Content-Length': '105'}  
body:{'name': 'Usuario Gmail', 'email': 'usuario@gmail.com', 'password': 'Senha123', 'administrador': 'false'}
- 17:35:20.610 **[INFO]** POST Response : url=https://serverest.dev/usuarios  
status=201, reasonCreated  
headers:{'access-control-allow-origin': '\*', 'x-dns-prefetch-control': 'off', 'x-frame-options': 'SAMEORIGIN', 'strict-transport-security': 'max-age=15552000; includeSubDomains', 'x-download-options': 'noopener', 'x-content-type-options': 'nosniff', 'x-xss-protection': '1; mode=block', 'content-type': 'application/json; charset=utf-8', 'x-cloud-trace-context': '199e161e08965f8d0972ad59802a01e', 'date': 'Thu, 11 Sep 2025 20:35:20 GMT', 'server': 'Google Frontend', 'Content-Length': '82'}  
body:  
{"message": "Cadastro realizado com sucesso",  
"\_id": "CNIaFZarjxR0Dj2"}
- 17:35:20.610 **[INFO]** `_:` C:\Users\Andressa\AppData\Roaming\Python\Python313\site-packages\urllib3\connectionpool.py:1097: InsecureRequestWarning: Unverified HTTPS request is being made to host 'serverest.dev'. Adding certificate verification is strongly advised. See: <https://urllib3.readthedocs.io/en/latest/advanced-usage.html#tls-warnings>  
warnings.warn(\_)

## CT004:

00 REPORT

```

[-] TEST CT004: Cadastrar usuário com provedor bloqueado (hotmail)
Full Name: Test Serverest.CT004: Cadastrar usuário com provedor bloqueado (hotmail)
Tags: POST, USUARIOS
Start / End / Elapsed: 20250911 17:35:20.878 / 20250911 17:35:21.133 / 00:00:00.255
Status: FAIL
Message: Uri https://serverest.dev/usuarios Expected status: 201 != 400
  - [KEYWORD] ${resp} = test_serverest.Cadastrar_usuario_provedor_bloqueado hotmail 400
    Start / End / Elapsed: 20250911 17:35:20.882 / 20250911 17:35:21.124 / 00:00:00.242
    + [KEYWORD] ${json} = common.importar.json.estatico json_usuarios.json
    + [KEYWORD] ${body} = Colecoes.Get.From.Dictionary ${json} usuario_S[provedor]
    - [KEYWORD] ${resposta} = RequestLibrary POST On Session serverest /usuarios json=${body} expected_status=${status_esperado}
      Documentation: Sends a POST request on a previously created HTTP Session.
      Start / End / Elapsed: 20250911 17:35:20.899 / 20250911 17:35:21.122 / 00:00:00.223
      17:35:21.102 [INFO] POST Request : url=https://serverest.dev/usuarios
      path_url=usuarios
      headers={User-Agent: 'python-requests/2.32.5', 'Accept-Encoding': 'gzip, deflate', 'Accept': '*', 'Connection': 'keep-alive', 'Content-Type': 'application/json', 'Content-Length': '109'}
      body={{"name": "Usuario Hotmail", "email": "usuario@hotmail.com", "password": "Senha123", "administrador": "false"}}

      17:35:21.104 [INFO] POST Response : url=https://serverest.dev/usuarios
      status=201, reasonCreated
      headers={'access-control-allow-origin': '', 'x-dns-prefetch-control': 'off', 'x-frame-options': 'SAMEORIGIN', 'strict-transport-security': 'max-age=15552000; includeSubDomains', 'x-download-options': 'noopener', 'x-content-type-options': 'nosniff', 'x-xss-protection': '1; mode=block', 'content-type': 'application/json; charset=utf-8', 'x-cloud-trace-context': '22362d0c909973cf0972ad5980d2a1d7', 'date': 'Thu, 11 Sep 2025 20:35:21 GMT', 'server': 'Google Frontend', 'Content-Length': '82'}
      body={
        "message": "Cadastro realizado com sucesso",
        "_id": "msqdfmwokwJeqjAsT"
      }

      17:35:21.105 [INFO] C:\Users\Andressa\AppData\Roaming\Python\Python313\site-packages\urllib3\connectionpool.py:1097: InsecureRequestWarning: Unverified HTTPS request is being made to host 'serverest.dev'. Adding certificate verification is strongly advised. See: https://urllib3.readthedocs.io/en/latest/advanced-usage.html#tls-warnings
      warnings.warn(

```

## CT005:

00:00:265

```

[-] TEST CT005: Atualizar usuário inexistente
Full Name: Test Serverest.CT005: Atualizar usuário inexistente
Tags: PUT, USUARIOS
Start / End / Elapsed: 20250911 17:35:21.138 / 20250911 17:35:21.403 / 00:00:00.265
Status: PASS
  - [KEYWORD] ${resp} = test_serverest.Atualizar_usuario_inexistente 201
    Start / End / Elapsed: 20250911 17:35:21.142 / 20250911 17:35:21.382 / 00:00:00.240
    + [KEYWORD] ${json} = common.importar.json.estatico json_usuarios.json
    + [KEYWORD] ${body} = Colecoes.Get.From.Dictionary ${json} usuario_valido
    + [KEYWORD] ${email_rand} = String.Generate.Random.String 8 [LOWER]
    + [KEYWORD] Colecoes.Set.To.Dictionary ${body} email ${email_rand}@robot.com
    - [KEYWORD] ${resposta} = RequestLibrary.PUT On Session serverest /usuarios/64646 json=${body} expected_status=${status_esperado}
      Documentation: Sends a PUT request on a previously created HTTP Session.
      Start / End / Elapsed: 20250911 17:35:21.171 / 20250911 17:35:21.380 / 00:00:00.209
      17:35:21.368 [INFO] PUT Request : url=https://serverest.dev/usuarios/64646
      path_url=usuarios/64646
      headers={User-Agent: 'python-requests/2.32.5', 'Accept-Encoding': 'gzip, deflate', 'Accept': '*', 'Connection': 'keep-alive', 'Content-Type': 'application/json', 'Content-Length': '105'}
      body={{"name": "Andressa Mota", "email": "hhkrtrn@robot.com", "password": "Senha123", "administrador": "true"}}

      17:35:21.368 [INFO] PUT Response : url=https://serverest.dev/usuarios/64646
      status=201, reasonCreated
      headers={'access-control-allow-origin': '', 'x-dns-prefetch-control': 'off', 'x-frame-options': 'SAMEORIGIN', 'strict-transport-security': 'max-age=15552000; includeSubDomains', 'x-download-options': 'noopener', 'x-content-type-options': 'nosniff', 'x-xss-protection': '1; mode=block', 'content-type': 'application/json; charset=utf-8', 'x-cloud-trace-context': '3eebeffea3e18e0972ad5980d2aec0', 'date': 'Thu, 11 Sep 2025 20:35:21 GMT', 'server': 'Google Frontend', 'Content-Length': '82'}
      body={
        "message": "Cadastro realizado com sucesso",
        "_id": "5dTHD9hsJKg3tbd"
      }

      17:35:21.368 [INFO] C:\Users\Andressa\AppData\Roaming\Python\Python313\site-packages\urllib3\connectionpool.py:1097: InsecureRequestWarning: Unverified HTTPS request is being made to host 'serverest.dev'. Adding certificate verification is strongly advised. See: https://urllib3.readthedocs.io/en/latest/advanced-usage.html#tls-warnings
      warnings.warn(

```

## CT006:

00:00:227

```

[-] TEST CT006: Deletar usuário inexistente
Full Name: Test Serverest.CT006: Deletar usuário inexistente
Tags: DELETE, USUARIOS
Start / End / Elapsed: 20250911 17:35:21.414 / 20250911 17:35:21.641 / 00:00:00.227
Status: PASS
  - [KEYWORD] ${resp} = test_serverest.Deletar_usuario_inexistente 200
    Start / End / Elapsed: 20250911 17:35:21.416 / 20250911 17:35:21.620 / 00:00:00.204
    - [KEYWORD] ${resposta} = RequestLibrary.DELETE On Session serverest /usuarios/IDinexistente expected_status=${status_esperado}
      Documentation: Sends a DELETE request on a previously created HTTP Session.
      Start / End / Elapsed: 20250911 17:35:21.417 / 20250911 17:35:21.616 / 00:00:00.199
      17:35:21.609 [INFO] DELETE Request : url=https://serverest.dev/usuarios/IDinexistente
      path_url=usuarios/IDinexistente
      headers={User-Agent: 'python-requests/2.32.5', 'Accept-Encoding': 'gzip, deflate', 'Accept': '*', 'Connection': 'keep-alive', 'Content-Type': 'application/json', 'Content-Length': '0'}
      body=None

      17:35:21.609 [INFO] DELETE Response : url=https://serverest.dev/usuarios/IDinexistente
      status=200, reasonOK
      headers={'access-control-allow-origin': '', 'x-dns-prefetch-control': 'off', 'x-frame-options': 'SAMEORIGIN', 'strict-transport-security': 'max-age=15552000; includeSubDomains', 'x-download-options': 'noopener', 'x-content-type-options': 'nosniff', 'x-xss-protection': '1; mode=block', 'content-type': 'application/json; charset=utf-8', 'x-cloud-trace-context': 'a3a596d2a7bcd854a0972ad5980d2ac16', 'date': 'Thu, 11 Sep 2025 20:35:21 GMT', 'server': 'Google Frontend', 'Content-Length': '46'}
      body={
        "message": "Nenhum registro excluido"
      }

      17:35:21.610 [INFO] C:\Users\Andressa\AppData\Roaming\Python\Python313\site-packages\urllib3\connectionpool.py:1097: InsecureRequestWarning: Unverified HTTPS request is being made to host 'serverest.dev'. Adding certificate verification is strongly advised. See: https://urllib3.readthedocs.io/en/latest/advanced-usage.html#tls-warnings
      warnings.warn(
      17:35:21.615 [INFO] ${resposta} = <Response [200]>

```

## CT007:

- TEST CT007: Listar usuários

Full Name: Test Serverest.CT007: Listar usuários

Tags: GET, USUARIOS

Start / End / Elapsed: 20250911 17:35:21.646 / 20250911 17:35:22.138 / 00:00:00.492

Status: PASS

- KEYWORD \${resp} = test\_serverest.Listar\_usuarios 200 Start / End / Elapsed: 20250911 17:35:21.651 / 20250911 17:35:22.115 / 00:00:00.466 Documentation: Sends a GET request on a previously created HTTP Session.
- KEYWORD \${status} = RequestLibrary.GET On Session serverest /usuarios expected\_status=\${status\_esperado} Start / End / Elapsed: 17:35:22.092 INFO get Request.url=https://serverest.dev/usuarios path.url=/usuarios headers={'User-Agent': 'python-requests/2.32.5', 'Accept-Encoding': 'gzip, deflate', 'Accept': '\*/\*', 'Connection': 'keep-alive', 'Content-Type': 'application/json'} body=None
- 17:35:22.094 INFO GET Response : url=https://serverest.dev/usuarios status=200 headers={...} headers['Access-Control-Allow-Origin']: '', 'X-DNS-Prefetch-Control': 'off', 'X-Frame-Options': 'SAMEORIGIN', 'Strict-Transport-Security': 'max-age=15552000; includeSubDomains', 'X-Download-Options': 'noopen', 'X-Content-Type-Options': 'nosniff', 'X-XSS-Protection': '1; mode=block', 'Content-Type': 'application/json; charset=utf-8', 'X-Cloud-Trace-Context': '08ebef6765e2bc7a0972ad5980d2ae5', 'Date': 'Thu, 11 Sep 2025 20:35:21 GMT', 'Server': 'Google Frontend', 'Content-Length': '66766'} body={ "quantidade": 307, "usuarios": [ { "nome": "Natália Dias", "email": "1537492080@getnada.com", "password": "398f8Gn1j", "administrador": "true", "\_id": "80Mggesy1TzIvPv" }, { "nome": "Bella da Paz", "email": "8236971058@getnada.com", "password": "87SW3DsNEv", "administrador": "false", } ] }

## CT08:

- TEST CT008: Login com usuário válido

Full Name: Test Serverest.CT008: Login com usuário válido

Tags: LOGIN, POST

Start / End / Elapsed: 20250911 17:35:22.146 / 20250911 17:35:22.425 / 00:00:00.279

Status: PASS

- KEYWORD \${resp} = test\_serverest.Login\_com\_credenciais user\_valido 200 Start / End / Elapsed: 20250911 17:35:22.148 / 20250911 17:35:22.394 / 00:00:00.246 Documentation: Sends a POST request on a previously created HTTP Session.
- + KEYWORD \${json} = common.importar\_json\_estatico json\_login.json Start / End / Elapsed: 20250911 17:35:22.163 / 20250911 17:35:22.389 / 00:00:00.226 Documentation: Sends a POST request on a previously created HTTP Session.
- + KEYWORD \${body} = Collections.Get From Dictionary \${json} \${tipo\_usuario} Start / End / Elapsed: 17:35:22.366 INFO POST Request : url=https://serverest.dev/login path.url=/login headers={'User-Agent': 'python-requests/2.32.5', 'Accept-Encoding': 'gzip, deflate', 'Accept': '\*/\*', 'Connection': 'keep-alive', 'Content-Type': 'application/json', 'Content-Length': '47'} body=b'{"email": "fulano@da.com", "password": "teste"}'
- 17:35:22.366 INFO POST Response : url=https://serverest.dev/login status=200, reason=OK headers={...} headers['Access-Control-Allow-Origin']: '', 'X-Dns-Prefetch-Control': 'off', 'X-Frame-Options': 'Sameorigin', 'Strict-Transport-Security': 'max-age=15552000; includeSubdomains', 'X-Download-Options': 'noopen', 'X-Content-Type-Options': 'nosniff', 'X-Xss-Protection': '1; mode=block', 'Content-Type': 'application/json; charset=utf-8', 'X-Cloud-Trace-Context': '4c17ce5c8e32bd6972ad5980d2ae09,o=1', 'Date': 'Thu, 11 Sep 2025 20:35:22 GMT', 'Server': 'Google Frontend', 'Content-Length': '265'} body={ "message": "Login realizado com sucesso", "authorization": "Bearer eyJhbGciOiJIUzI1NiwiZW5kcC161kpXVC39.eY3lbWpBC16ImZibGFrUbBXYSjB20ILC3wYXNzd29yZC1GInRl3RliwiwF0IJoxNzU3HjIyOTiyLC1leHaoIE3NT2MjHmJ39.K6vfhwGpn1Q wq9JBPvNUxG81p2g081r\_1VFPv"
- 17:35:22.367 INFO C:\Users\Andressa\AppData\Roaming\Python\Python311\site-packages\urllib3\connectionpool.py:1097: InsecureRequestWarning: Unverified HTTPS request is being made to host 'serverest.dev'. Adding certificate verification is strongly advised. See: https://urllib3.readthedocs.io/en/latest/advanced-usage.html#ssl\_warnings warnings.warn(

## CT009:

- TEST CT009: Login com usuário inexistente

Full Name: Test Serverest.CT009: Login com usuário inexistente

Tags: LOGIN, POST

Start / End / Elapsed: 20250911 17:35:22.431 / 20250911 17:35:22.731 / 00:00:00.300

Status: PASS

- KEYWORD \${resp} = test\_serverest.Login\_com\_credenciais user\_invalido 401 Start / End / Elapsed: 20250911 17:35:22.433 / 20250911 17:35:22.681 / 00:00:00.248 Documentation: Sends a POST request on a previously created HTTP Session.
- + KEYWORD \${json} = common.importar\_json\_estatico json\_login.json Start / End / Elapsed: 20250911 17:35:22.441 / 20250911 17:35:22.689 / 00:00:00.235 Documentation: Sends a POST request on a previously created HTTP Session.
- KEYWORD \${status} = RequestLibrary.POST On Session serverest /login json=\${body} expected\_status=\${status\_esperado} Start / End / Elapsed: 17:35:22.641 INFO POST Request : url=https://serverest.dev/login path.url=/login headers={'User-Agent': 'python-requests/2.32.5', 'Accept-Encoding': 'gzip, deflate', 'Accept': '\*/\*', 'Connection': 'keep-alive', 'Content-Type': 'application/json', 'Content-Length': '50'} body=b'{"email": "naoexiste@qa.com", "password": "teste"}'
- 17:35:22.642 INFO POST Response : url=https://serverest.dev/login status=401, reason=Unauthorized headers={...} headers['Access-Control-Allow-Origin']: '', 'X-Dns-Prefetch-Control': 'off', 'X-Frame-Options': 'Sameorigin', 'Strict-Transport-Security': 'max-age=15552000; includeSubdomains', 'X-Download-Options': 'noopen', 'X-Content-Type-Options': 'nosniff', 'X-Xss-Protection': '1; mode=block', 'Content-Type': 'application/json; charset=utf-8', 'X-Cloud-Trace-Context': '0b35ef72221c5a00972ad5980d2a9ad', 'Date': 'Thu, 11 Sep 2025 20:35:22 GMT', 'Server': 'Google Frontend', 'Content-Length': '48'} body={ "message": "Email e/ou senha inválidos" }

## CT010:

```

[= TEST] CT010: Login com senha incorreta
Full Name: Test Serverest.CT010: Login com senha incorreta
Tags: LOGIN, POST
Start / End / Elapsed: 20250911 17:35:22.733 / 20250911 17:35:22.996 / 00:00:00.263
Status: PASS
- [KEYWORD] ${resp} = test_serverest.Login com credenciais user_senha_errada 401
  Start / End / Elapsed: 20250911 17:35:22.734 / 20250911 17:35:22.998 / 00:00:00.234
+ [KEYWORD] ${json} = common.importar json estatico json_login.json
+ [KEYWORD] ${body} = Collections.Get From Dictionary ${json} ${tipo_usuario}
- [KEYWORD] ${resposta} = RequestLibrary POST On Session serverest /login json=${body} expected_status=${status_esperado}
  Documentation: Sends a POST request on a previously created HTTP Session.
  Start / End / Elapsed: 20250911 17:35:22.746 / 20250911 17:35:22.964 / 00:00:00.218
  17:35:22.949 [INFO] POST Request : url=https://serverest.dev/login
    path_url=/login
    headers={'User-Agent': 'python-requests/2.32.5', 'Accept-Encoding': 'gzip, deflate', 'Accept': '/+', 'Connection': 'keep-alive', 'Content-Type': 'application/json', 'Content-Length': '53'}
    body=b'{"email": "fulano@qa.com", "password": "senhaerrada"}'
    body={
      "message": "Email e/ou senha inválidos"
    }
  17:35:22.949 [INFO] POST Response : url=https://serverest.dev/login
    status=401, reason=unauthorized
    headers={'access-control-allow-origin': '*', 'x-dns-prefetch-control': 'off', 'x-frame-options': 'SAMEORIGIN', 'strict-transport-security': 'max-age=15552000; includeSubDomains', 'x-download-options': 'noopen', 'x-content-type-options': 'nosniff', 'x-xss-protection': '1; mode=block', 'content-type': 'application/json; charset=utf-8', 'x-cloud-trace-context': '39a9990395c034cf0972ad5980d2a9c7', 'date': 'Thu, 11 Sep 2025 20:35:22 GMT', 'server': 'Google Frontend', 'Content-Length': '48'}
    body={
      "message": "Email e/ou senha inválidos"
    }

00:00:00.234
00:00:00.006
00:00:00.004
00:00:00.218

```

## CT011:

```

[= TEST] CT011: Cadastrar novo produto com sucesso
Full Name: Test Serverest.CT011: Cadastrar novo produto com sucesso
Tags: POST, PRODUTOS
Start / End / Elapsed: 20250911 17:35:23.000 / 20250911 17:35:23.499 / 00:00:00.499
Status: PASS
- [KEYWORD] ${token} = test_serverest.Obter Token de Autenticação
  Start / End / Elapsed: 20250911 17:35:23.237 / 20250911 17:35:23.474 / 00:00:00.237
+ [KEYWORD] ${resp} = test_serverest.Cadastrar produto valido ${token} 201
  Start / End / Elapsed: 20250911 17:35:23.237 / 20250911 17:35:23.474 / 00:00:00.237
+ [KEYWORD] ${headers} = Bulk.Create Dictionary Authorization=${token}
+ [KEYWORD] ${nome_prod} = String.Generate Random String 10 Produto [LOWER]
+ [KEYWORD] ${body} = Bulk.Create Dictionary nome=${nome_prod} preco=500 descricao=Mouse Gamer quantidade=100
- [KEYWORD] ${resposta} = RequestLibrary POST On Session serverest /produtos ${produtos json=${body}} headers=${headers} expected_status=${status_esperado}
  Documentation: Sends a POST request on a previously created HTTP Session.
  Start / End / Elapsed: 20250911 17:35:23.256 / 20250911 17:35:23.469 / 00:00:00.213
  17:35:23.458 [INFO] POST Request : url=https://serverest.dev/produtos
    path_url=/produtos
    headers={'User-Agent': 'python-requests/2.32.5', 'Accept-Encoding': 'gzip, deflate', 'Accept': '/+', 'Connection': 'keep-alive', 'Content-Type': 'application/json', 'Authorization': 'Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJlbWFpbC16Zm1bdFub0BX5jb20iLCJwYXNzd29yZC16InRlc3RlIiwiaWF0IjoxNzU3NjIyOTIxLCJleHAiOjE3NTc2MjNmN9.Zol2gSiOsV7fw-3i1THGhLeVu7E3mm8u0Pj1-qk', 'Content-Length': '87'}
    body=b'{"name": "nccbgmnp1u", "preco": "500", "descricao": "Mouse Gamer", "quantidade": "100"}'
    body={
      "message": "Cadastro realizado com sucesso",
      "_id": "1umam8Spv8KIQ0JB"
    }
  17:35:23.458 [INFO] POST Response : url=https://serverest.dev/produtos
    status=201, reason=Created
    headers={'access-control-allow-origin': '*', 'x-dns-prefetch-control': 'off', 'x-frame-options': 'SAMEORIGIN', 'strict-transport-security': 'max-age=15552000; includeSubDomains', 'x-download-options': 'noopen', 'x-content-type-options': 'nosniff', 'x-xss-protection': '1; mode=block', 'content-type': 'application/json; charset=utf-8', 'x-cloud-trace-context': 'c7888b684203ed40972ad5980d2a9ab', 'date': 'Thu, 11 Sep 2025 20:35:23 GMT', 'server': 'Google Frontend', 'Content-Length': '82'}
    body={
      "message": "Cadastro realizado com sucesso",
      "_id": "1umam8Spv8KIQ0JB"
    }

00:00:00.232
00:00:00.237
00:00:00.004
00:00:00.003
00:00:00.005
00:00:00.213
00:00:00.499

```

## CT012:

```

[= TEST] CT012: Cadastrar produto com token expirado
Full Name: Test Serverest.CT012: Cadastrar produto com token expirado
Tags: POST, PRODUTOS
Start / End / Elapsed: 20250911 17:35:23.512 / 20250911 17:35:23.762 / 00:00:00.250
Status: PASS
- [KEYWORD] ${resp} = test_serverest.Cadastrar produto valido Bearer eyJhbGciOiUzI1NilsInR5cCI6IkpXVCJ9eyJlbWFpbC16Zm1bdFub0BX5jb20iLCJwYXNzd29yZC16InRlc3RlIiwiaWF0IjoxNzU3NjIyOTIxLCJleHAiOjE3NTc2MTCzMTJ9.aJqclsNcva0fyPeJrFYh0vVX81h6A_Iy-2o_401
  Start / End / Elapsed: 20250911 17:35:23.514 / 20250911 17:35:23.745 / 00:00:00.231
+ [KEYWORD] ${headers} = Bulk.Create Dictionary Authorization=${token}
+ [KEYWORD] ${nome_prod} = String.Generate Random String 10 Produto [LOWER]
+ [KEYWORD] ${body} = Bulk.Create Dictionary nome=${nome_prod} preco=500 descricao=Mouse Gamer quantidade=100
- [KEYWORD] ${resposta} = RequestLibrary POST On Session serverest /produtos ${produtos json=${body}} headers=${headers} expected_status=${status_esperado}
  Documentation: Sends a POST request on a previously created HTTP Session.
  Start / End / Elapsed: 20250911 17:35:23.522 / 20250911 17:35:23.740 / 00:00:00.218
  17:35:23.722 [INFO] POST Request : url=https://serverest.dev/produtos
    path_url=/produtos
    headers={'User-Agent': 'python-requests/2.32.5', 'Accept-Encoding': 'gzip, deflate', 'Accept': '/+', 'Connection': 'keep-alive', 'Content-Type': 'application/json', 'Authorization': 'Bearer eyJhbGciOiIzI1NilsInR5cCI6IkpXVCJ9.eyJlbWFpbC16Zm1bdFub0BX5jb20iLCJwYXNzd29yZC16InRlc3RlIiwiaWF0IjoxNzU3NjE2NzEyLCJleHAiOjE3NTc2MTCzMTJ9.aJqcIShcvaeFyEr7Hv0vX1h6A_Iy78ECCa--y-2o', 'Content-Length': '87'}
    body=b'{"name": "crvcutkpt1", "preco": "500", "descricao": "Mouse Gamer", "quantidade": "100"}'
    body={
      "message": "Token de acesso ausente, inválido, expirado ou usuário do token não existe mais"
    }
  17:35:23.724 [INFO] POST Response : url=https://serverest.dev/produtos
    status=401, reason=unauthorized
    headers={'access-control-allow-origin': '*', 'x-dns-prefetch-control': 'off', 'x-frame-options': 'SAMEORIGIN', 'strict-transport-security': 'max-age=15552000; includeSubDomains', 'x-download-options': 'noopen', 'x-content-type-options': 'nosniff', 'x-xss-protection': '1; mode=block', 'content-type': 'application/json; charset=utf-8', 'x-cloud-trace-context': '442c21ca74de3de50972ad5980d2a9bd', 'date': 'Thu, 11 Sep 2025 20:35:23 GMT', 'server': 'Google Frontend', 'Content-Length': '103'}
    body={
      "message": "Token de acesso ausente, inválido, expirado ou usuário do token não existe mais"
    }

00:00:00.231
00:00:00.001
00:00:00.002
00:00:00.218
00:00:00.250
00:00:00.722
00:00:00.231

```

## CT013:

**TEST** CT013: Cadastrar produto duplicado

Full Name: Test Serverest.CT013: Cadastrar produto duplicado

Tags: POST, PRODUTOS

Start / End / Elapsed: 20250911 17:35:23.772 / 20250911 17:35:24.517 / 00:00:00.745

Status: **PASS**

```

+ KEYWORD ${token} = test_serverest.Obter Token de Autenticação
- KEYWORD ${resp} = test_serverest.Cadastrar produto duplicado ${token} 400
Start / End / Elapsed: 20250911 17:35:24.026 / 20250911 17:35:24.484 / 00:00:00.458
+ KEYWORD ${headers} = bultn.Create Dictionary Authorization=${token}
+ KEYWORD ${body} = bultn.Create Dictionary nome=Produto Duplicado Teste preco=100 descricao=Item Teste quantidade=5
+ KEYWORD RequestLibrary POST On Session serverest /produtos json=${body} headers=${headers} expected_status=any
- KEYWORD ${resposta} = RequestLibrary POST On Session serverest /produtos json=${body} headers=${headers} expected_status=${status_esperado}
Documentation: Sends a POST request on a previously created HTTP Session.
Start / End / Elapsed: 20250911 17:35:24.254 / 20250911 17:35:24.480 / 00:00:00.226
17:35:24.468 [INFO] POST Request : url=https://serverrest.dev/produtos
path.Url=/produtos
headers={'User-Agent': 'python-requests/2.32.5', 'Accept-Encoding': 'gzip, deflate', 'Accept': '*/*', 'Connection': 'keep-alive', 'Content-Type': 'application/json', 'Authorization': 'Bearer eyJhbGciOiJIUzI1NiIsInScIi6IkpxCj9.ejJlWfpbcI6zmzb0Fu08XyS5jb28lCjwXNid29yZC16Inrlc3RlliwH0ijoxNzU3NjIyOTizLCleHai0je3NTc2MjHIMjN9.z0l2GigosV7
body=b'{"name": "Produto Duplicado Teste", "preco": "100", "descricao": "Item Teste", "quantidade": "5"}'
body+={

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

17:35:24.469 [INFO] POST Response : url=https://serverrest.dev/produtos
status=400, reason=Bad Request
headers='access-control-allow-origin': "", 'x-dns-prefetch-control': 'off', 'x-frame-options': 'SAMEORIGIN', 'strict-transport-security': 'max-age=15552000; includeSubDomains', 'x-download-options': 'nopopen', 'x-content-type-options': 'nosniff', 'x-xss-protection': '1; mode=block', 'content-type': 'application/json; charset=utf-8', 'x-cloud-trace-context': 'e236f1293e45588a0972a0d5980d2a362', 'date': 'Thu, 11 Sep 2025 20:35:24 GMT', 'server': 'Google Frontend', 'Content-Length': '33'}
body={

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

```

## CT014:

**TEST** CT014: Cadastrar produto sem autenticação

Full Name: Test Serverest.CT014: Cadastrar produto sem autenticação

Tags: POST, PRODUTOS

Start / End / Elapsed: 20250911 17:35:24.523 / 20250911 17:35:24.764 / 00:00:00.241

Status: **PASS**

```

- KEYWORD ${resp} = test_serverest.Cadastrar produto sem autenticação 401
Start / End / Elapsed: 20250911 17:35:24.528 / 20250911 17:35:24.737 / 00:00:00.209
+ KEYWORD ${body} = bultn.Create Dictionary nome=Produto Sem Token preco=200 descricao=Teste quantidade=3
- KEYWORD ${resposta} = RequestLibrary POST On Session serverest /produtos json=${body} expected_status=${status_esperado}
Documentation: Sends a POST request on a previously created HTTP Session.
Start / End / Elapsed: 20250911 17:35:24.532 / 20250911 17:35:24.736 / 00:00:00.204
17:35:24.725 [INFO] POST Request : url=https://serverrest.dev/produtos
path.Url=/produtos
headers={'User-Agent': 'python-requests/2.32.5', 'Accept-Encoding': 'gzip, deflate', 'Accept': '*/*', 'Connection': 'keep-alive', 'Content-Type': 'application/json', 'Content-Length': '86'}
body=b'{"name": "Produto Sem Token", "preco": "200", "descricao": "Teste", "quantidade": "3"}'
body+={

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

17:35:24.726 [INFO] POST Response : url=https://serverrest.dev/produtos
status=401, reason=Unauthorized
headers='access-control-allow-origin': "", 'x-dns-prefetch-control': 'off', 'x-frame-options': 'SAMEORIGIN', 'strict-transport-security': 'max-age=15552000; includeSubDomains', 'x-download-options': 'nopopen', 'x-content-type-options': 'nosniff', 'x-xss-protection': '1; mode=block', 'content-type': 'application/json; charset=utf-8', 'x-cloud-trace-context': 'a35c2287e4ceafef972ad5980d2a877b', 'date': 'Thu, 11 Sep 2025 20:35:24 GMT', 'server': 'Google Frontend', 'Content-Length': '103'
body={

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

```

## CT015:

**TEST** CT015: Atualizar produto inexistente

Full Name: Test Serverest.CT015: Atualizar produto inexistente

Tags: PRODUTOS, PUT

Start / End / Elapsed: 20250911 17:35:24.769 / 20250911 17:35:25.255 / 00:00:00.486

Status: **PASS**

```

+ KEYWORD ${token} = test_serverest.Obter Token de Autenticação
- KEYWORD ${resp} = test_serverest.Atualizar produto inexistente ${token} 201
Start / End / Elapsed: 20250911 17:35:25.008 / 20250911 17:35:25.246 / 00:00:00.238
+ KEYWORD ${headers} = bultn.Create Dictionary Authorization=${token}
+ KEYWORD ${nome_produto} = string.Generate Random String 10 Produto [LOWER]
+ KEYWORD ${body} = bultn.Create Dictionary nome=${nome_produto} preco=150 descricao=Update Teste quantidade=10
- KEYWORD ${resposta} = RequestLibrary PUT On Session serverest /produtos/1dinexistente000 json=${body} headers=${headers} expected_status=${status_esperado}
Documentation: Sends a PUT request on a previously created HTTP Session.
Start / End / Elapsed: 20250911 17:35:25.035 / 20250911 17:35:25.244 / 00:00:00.209
17:35:25.225 [INFO] PUT Request : url=https://serverrest.dev/produtos/1dinexistente000
path.Url=/produtos/1dinexistente000
headers={'User-Agent': 'python-requests/2.32.5', 'Accept-Encoding': 'gzip, deflate', 'Accept': '*/*', 'Connection': 'keep-alive', 'Content-Type': 'application/json', 'Authorization': 'Bearer eyJhbGciOiJIUzI1NiIsInScIi6IkpxCj9.ejJlWfpbcI6zmzb0Fu08XyS5jb28lCjwXNid29yZC16Inrlc3RlliwH0ijoxNzU3NjIyOTizLCleHai0je3NTc2MjHIMjR9.3Qoyil7Spv9o
XaBImuvuShnjFSHeH0ta0jys', 'Content-Length': '87'}
body=b'{"name": "ptjsadrtr", "preco": "150", "descricao": "Update Teste", "quantidade": "10"}'
body+={

}
"message": "Cadastro realizado com sucesso",
"_id": "CmBHQgSiLAzyv4lE"
}

17:35:25.226 [INFO] PUT Response : url=https://serverrest.dev/produtos/1dinexistente000
status=201, reason=Created
headers='access-control-allow-origin': "", 'x-dns-prefetch-control': 'off', 'x-frame-options': 'SAMEORIGIN', 'strict-transport-security': 'max-age=15552000; includeSubDomains', 'x-download-options': 'nopopen', 'x-content-type-options': 'nosniff', 'x-xss-protection': '1; mode=block', 'content-type': 'application/json; charset=utf-8', 'x-cloud-trace-context': 'e34d1a11ead9f59872ad5980d2a5db', 'date': 'Thu, 11 Sep 2025 20:35:25 GMT', 'server': 'Google Frontend', 'Content-Length': '92'
body={

}
"message": "Cadastro realizado com sucesso",
"_id": "CmBHQgSiLAzyv4lE"
}

```

## CT016:

**REPORT**

00:00:00,000

```

- TEST CT016: Excluir produto vinculado a carrinho
Full Name: Test Serverest.CT016: Excluir produto vinculado a carrinho
Tags: DELETE, PRODUTOS
Start / End / Elapsed: 20250911 17:35:25.258 / 20250911 17:35:26.365 / 00:00:01.107
Status: PASS

+ KEYWORD ${token} = test_serverest.Obter Token de Autenticação
+ KEYWORD test_serverest.Limpar carrinho existente ${token}
- KEYWORD ${resp} = test_serverest.Excluir produto em carrinho ${token} 400
Start / End / Elapsed: 20250911 17:35:25.689 / 20250911 17:35:26.338 / 00:00:00.649
+ KEYWORD ${headers} = Built Create Dictionary Authorization=${token}
+ KEYWORD ${resp_produto} = test_serverest.Cadastrar produto valido ${token} 201
+ KEYWORD ${id_produto} = Built Set Variable ${resp_produto.json()['_id']}
+ KEYWORD test_serverest.Criar carrinho ${token} ${id_produto} 201
- KEYWORD ${resposta_delete} = RequestLibrary DELETE On Session serverest /produtos/${id_produto} headers=${headers} expected_status=${status_esperado} 00:00:00.212
Documentation: Sends a DELETE request on a previously created HTTP Session.
Start / End / Elapsed: 20250911 17:35:26.322 / 20250911 17:35:26.334 / 00:00:00.212
17:35:26.324 [INFO] DELETE Request : url=https://serverest.dev/produtos/t9A04XXtzentefav
path_url=produtos/t9A04XXtzentefav
headers=[{"User-Agent": "python-requests/2.32.5", "Accept-Encoding": "gzip, deflate", "Accept": "*/*", "Connection": "keep-alive", "Content-Type": "application/json", "Authorization": "Bearer eyJhbGciOiJIaWEgIkHSUzIiLCJpYXQiXzIiLCJleHAiOiJ0ZXN0ZW1jY2lIaG9tZW1jMjM1MjV9.5diYe3jg44dPou_mWh93172HdgbqewqaxXn-U1U", "Content-Length": "0"}]
body=None

17:35:26.324 [INFO] DELETE Response : url=https://serverest.dev/produtos/t9A04XXtzentefav
status=400, reason=Bad Request
headers=[{"access-control-allow-origin": "", "x-dns-prefetch-control": "off", "x-frame-options": "SAMEORIGIN", "strict-transport-security": "max-age=31536000; includeSubDomains", "x-download-options": "noopener", "x-content-type-options": "nosniff", "x-xss-protection": "1; mode=block", "content-type": "application/json; charset=utf-8", "x-cloud-trace-context": "c6957a8a0f98755f0e972ad5980d2abfb9", "date": "Thu, 11 Sep 2025 20:35:26 GMT", "server": "Google Frontend", "Content-Length": "135"}]
body=
message: "Não é permitido excluir produto que faz parte de carrinho",
idcarrinhos: [

```

## CT017:

**REPORT**

00:00:00,264

```

- TEST CT017: Listar produtos
Full Name: Test Serverest.CT017: Listar produtos
Tags: GET, PRODUTOS
Start / End / Elapsed: 20250911 17:35:26.369 / 20250911 17:35:26.633 / 00:00:00.264
Status: PASS

- KEYWORD ${resposta} = test_serverest.Listar produtos 200
Start / End / Elapsed: 20250911 17:35:26.371 / 20250911 17:35:26.613 / 00:00:00.242
- KEYWORD ${resposta} = RequestLibrary GET On Session serverest /produtos expected_status=${status_esperado} 00:00:00.237
Documentation: Sends a GET request on a previously created HTTP Session.
Start / End / Elapsed: 20250911 17:35:26.372 / 20250911 17:35:26.609 / 00:00:00.237
17:35:26.592 [INFO] GET Request : url=https://serverest.dev/produtos
headers=[{"User-Agent": "python-requests/2.32.5", "Accept-Encoding": "gzip, deflate", "Accept": "*/*", "Connection": "keep-alive", "Content-Type": "application/json"}]
body=None

17:35:26.593 [INFO] GET Response : url=https://serverest.dev/produtos
status=200, reason=ok
headers=[{"access-control-allow-origin": "", "x-dns-prefetch-control": "off", "x-frame-options": "SAMEORIGIN", "strict-transport-security": "max-age=31536000; includeSubDomains", "x-download-options": "noopener", "x-content-type-options": "nosniff", "x-xss-protection": "1; mode=block", "content-type": "application/json; charset=utf-8", "x-cloud-trace-context": "170da143490b5b61e0972ad5980d2ad64", "date": "Thu, 11 Sep 2025 20:35:26 GMT", "server": "Google Frontend", "Content-Length": "26646"}]
body=
{
    "quantidade": 125,
    "produtos": [
        {
            "name": "Awesome Fresh Fish",
            "preco": 500,
            "descricao": "The Football Is Good For Training And Recreational Purposes",
            "quantidade": 10,
            "_id": "6d8fRQXmrxwv1"
        },
        {
            "name": "Produto Teste 30",
            "preco": 15,
            "descricao": "Descrição do Produto Teste",
            ...
        }
    ]
}

```

## CT018:

**REPORT**

00:00:00,000

```

- TEST CT018: Criar carrinho com sucesso
Full Name: Test Serverest.CT018: Criar carrinho com sucesso
Tags: CARRINHOS, POST
Start / End / Elapsed: 20250911 17:35:26.640 / 20250911 17:35:27.570 / 00:00:00.930
Status: PASS

+ KEYWORD ${token} = test_serverest.Obter Token de Autenticação
+ KEYWORD test_serverest.Limpar carrinho existente ${token}
+ KEYWORD ${resp_produto} = test_serverest.Cadastrar produto valido ${token} 201
+ KEYWORD ${id_produto} = Built Set Variable ${resp_produto.json()['_id']}
- KEYWORD ${resp_carrinho} = test_serverest.Criar carrinho ${token} ${id_produto} 201
Start / End / Elapsed: 20250911 17:35:27.346 / 20250911 17:35:27.565 / 00:00:00.219
+ KEYWORD ${headers} = Built Create Dictionary Authorization=${token}
+ KEYWORD ${produto_dicionario} = Built Create Dictionary idProduto=${id_produto} quantidade=2
+ KEYWORD @${produtos_lista} = Built Create List ${produto_dicionario}
+ KEYWORD ${body} = Built Create Dictionary produtos=${produtos_lista}
- KEYWORD ${resposta} = RequestLibrary POST On Session serverest /carrinhos json=${body} headers=${headers} expected_status=${status_esperado} 00:00:00.208
Documentation: Sends a POST request on a previously created HTTP Session.
Start / End / Elapsed: 20250911 17:35:27.355 / 20250911 17:35:27.563 / 00:00:00.208
17:35:27.560 [INFO] POST Request : url=https://serverest.dev/carrinhos
path_url=carrinhos
headers=[{"User-Agent": "python-requests/2.32.5", "Accept-Encoding": "gzip, deflate", "Accept": "*/*", "Connection": "keep-alive", "Content-Type": "application/json", "Authorization": "Bearer eyJhbGciOiJIaWEgIkHSUzIiLCJpYXQiXzIiLCJleHAiOiJ0ZXN0ZW1jY2lIaG9tZW1jMjM1MjZ9.InpPQVCEj4x8y7Xcb1jhRxK2f7kA52B44eu6l0ac", "Content-Length": "68"}]
body={{"produtos": [{"idProduto": "fcCmqt8dW51zxbR", "quantidade": "2"}]}

17:35:27.561 [INFO] POST Response : url=https://serverest.dev/carrinhos
status=201, reason=Created
headers=[{"access-control-allow-origin": "", "x-dns-prefetch-control": "off", "x-frame-options": "SAMEORIGIN", "strict-transport-security": "max-age=31536000; includeSubDomains", "x-download-options": "noopener", "x-content-type-options": "nosniff", "x-xss-protection": "1; mode=block", "content-type": "application/json; charset=utf-8", "x-cloud-trace-context": "f9210c11c0421b038972ad5980d2a0d3", "date": "Thu, 11 Sep 2025 20:35:27 GMT", "server": "Google Frontend", "Content-Length": "82"}]

```

## CT019:

```

- [TEST] CT019: Concluir compra de um carrinho
Full Name: Test Serverest CT019: Concluir compra de um carrinho
Tags: CARRINHOS, DELETE
Start / End / Elapsed: 20250911 17:35:27.574 / 20250911 17:35:28.732 / 00:00:01.158
Status: PAss
  + [KEYWORD] ${token} = int_serveret. Obter Token de Autenticação
  + [KEYWORD] int_serveret. Limpar carrinho existente ${token}
  + [KEYWORD] ${resp_produto} = int_serveret. Cadastrar produto valido ${token} 201
  + [KEYWORD] ${id_produto} = num. Set Variable ${resp_produto.json()['_id']}
  + [KEYWORD] int_serveret. Criar carrinho ${token} ${id_produto} 201
  - [KEYWORD] ${resp_concluir} = int_serveret. Concluir compra ${token} 200
    Start / End / Elapsed: 20250911 17:35:28.478 / 20250911 17:35:28.702 / 00:00:00.224
  - [KEYWORD] ${headers} = num. Create Dictionary Authorization=${token}
    Documentation: Creates and returns a dictionary based on the given items.
    Start / End / Elapsed: 20250911 17:35:28.479 / 20250911 17:35:28.480 / 00:00:00.001
    17:35:28.479 [INFO] ${headers} = ('Authorization': 'Bearer eyJhbGciOiJIUzI1NiIsInR5cCIkIkpXVCJ9.eyJlbWFpbC16InZibGFub0BxYS5jb28lCjwVNzd29yZC16InRlc3RlliwiWF0ijoxNzUNJiyOTi1LCJleHAIOJE3NTc2MjMjMj9.g7fuk-pKMTqoWCK_8m3mCH2VfZbxoADbmh2d4'
  - [KEYWORD] ${resposta} = requestkey DELETE On Session serverest /carrinhos/concluir-compra headers=${headers} expected_status=${status_esperado}
    Documentation: Sends a DELETE request on a previously created HTTP Session.
    Start / End / Elapsed: 20250911 17:35:28.480 / 20250911 17:35:28.499 / 00:00:00.019
    17:35:28.485 [INFO] DELETE Request: /carrinhos/concluir-compra
      path_url=/carrinhos/concluir-compra
      headers='User-Agent': 'python-requests/2.32.5', 'Accept-Encoding': 'gzip, deflate', 'Accept': '/', 'Connection': 'keep-alive', 'Content-type': 'application/json', 'Authorization': 'Bearer eyJhbGciOiJIUzI1NiIsInR5cCIkIkpXVCJ9.eyJlbWFpbC16InZibGFub0BxYS5jb28lCjwVNzd29yZC16InRlc3RlliwiWF0ijoxNzUNJiyOTi1LCJleHAIOJE3NTc2MjMjMj9.g7fuk-pKMTqoWCK_8m3mCH2VfZbxoADbmh2d4', 'Content-Length': '0'
      body={}
    } message: 'Registro excluido com sucesso'
  17:35:28.486 [INFO] DELETE Response: url=https://serverest.dev/carrinhos/concluir-compra
  status=200, reason=None
  headers='access-control-allow-origin': '*', 'x-dns-prefetch-control': 'off', 'x-frame-options': 'SAMEORIGIN', 'strict-transport-security': 'max-age=15552000; includeSubDomains', 'x-download-options': 'noopen', 'x-content-type-options': 'nosniff', 'x-xss-protection': '1; mode=block', 'content-type': 'application/json; charset=utf-8', 'x-cloud-trace-context': '25d1eb0012feef1d07848ffcc57456', 'date': 'Thu, 11 Sep 2025 20:35:28 GMT', 'server': 'Google Frontend', 'Content-length': '51'
  body='{"message": "Registro excluido com sucesso"}'
}

```

## CT020:

```

- [TEST] CT020: Cancelar compra de um carrinho
Full Name: Test Serverest CT020: Cancelar compra de um carrinho
Tags: CARRINHOS, DELETE
Start / End / Elapsed: 20250911 17:35:28.740 / 20250911 17:35:30.055 / 00:00:01.315
Status: PAss
  + [KEYWORD] ${token} = int_serveret. Obter Token de Autenticação
  + [KEYWORD] int_serveret. Limpar carrinho existente ${token}
  + [KEYWORD] ${resp_produto} = int_serveret. Cadastrar produto valido ${token} 201
  + [KEYWORD] ${id_produto} = num. Set Variable ${resp_produto.json()['_id']}
  + [KEYWORD] int_serveret. Criar carrinho ${token} ${id_produto} 201
  - [KEYWORD] ${resp_cancelar} = int_serveret. Cancelar carrinho ${token} 200
    Start / End / Elapsed: 20250911 17:35:29.779 / 20250911 17:35:30.015 / 00:00:00.236
  - [KEYWORD] ${headers} = num. Create Dictionary Authorization=${token}
  - [KEYWORD] ${resposta} = requestkey DELETE On Session serverest /carrinhos/cancelar-compra headers=${headers} expected_status=${status_esperado}
    Documentation: Sends a DELETE request on a previously created HTTP Session.
    Start / End / Elapsed: 20250911 17:35:29.787 / 20250911 17:35:30.009 / 00:00:00.222
    17:35:29.996 [INFO] DELETE Request: /carrinhos/cancelar-compra
      path_url=/carrinhos/cancelar-compra
      headers='User-Agent': 'python-requests/2.32.5', 'Accept-Encoding': 'gzip, deflate', 'Accept': '/', 'Connection': 'keep-alive', 'Content-type': 'application/json', 'Authorization': 'Bearer eyJhbGciOiJIUzI1NiIsInR5cCIkIkpXVCJ9.eyJlbWFpbC16InZibGFub0BxYS5jb28lCjwVNzd29yZC16InRlc3RlliwiWF0ijoxNzUNJiyOTi4LCJleHAIOJE3NTc2MjMjMj9.BwBd1q74olx_FH58ApLXBFev6PAx1MAR_AkEdhVrd', 'Content-Length': '0'
      body={}
    } message: 'Registro excluido com sucesso. Estoque dos produtos reabastecido'
  17:35:29.997 [INFO] DELETE Response: url=https://serverest.dev/carrinhos/cancelar-compra
  status=200, reason=None
  headers='access-control-allow-origin': '*', 'x-dns-prefetch-control': 'off', 'x-frame-options': 'SAMEORIGIN', 'strict-transport-security': 'max-age=15552000; includeSubDomains', 'x-download-options': 'noopen', 'x-content-type-options': 'nosniff', 'x-xss-protection': '1; mode=block', 'content-type': 'application/json; charset=utf-8', 'x-cloud-trace-context': 'fF#B009cdfb228f51d87484ffcc57456', 'date': 'Thu, 11 Sep 2025 20:35:29 GMT', 'server': 'Google Frontend', 'Content-length': '86'
  body='{"message": "Registro excluido com sucesso. Estoque dos produtos reabastecido"}'
}

```

## 12. Matriz de Rastreabilidade

US Ref.	Acceptance Criteria Principal	Casos de Teste Relacionados
US001	Campos obrigatórios, e-mail único, restrições	CT001, CT002, CT003, CT004, CT005, CT006
US002	Login correto, bloqueio de usuários inválidos	CT007, CT008, CT009, CT010
US003	CRUD produtos, autenticação obrigatória, restrições	CT011, CT012, CT013, CT014, CT015, CT016

