



Planejamento de Testes (Sprint 6) - API ServeRest

1. Resumo

Este documento descreve o planejamento de testes da **API ServeRest**, uma API gratuita que simula uma loja virtual. O plano define a estratégia, as técnicas de teste, o escopo, o mapeamento dos endpoints e os critérios de entrada, saída e definição de pronto.

O foco está na validação funcional da API por meio da execução de testes exploratórios estruturados, garantindo cobertura adequada dos fluxos de usuário cliente e administrador, com clareza, rastreabilidade e organização.

2. Objetivos dos Testes

- Validar o funcionamento correto dos endpoints da API.
- Verificar a aplicação das regras de negócio.
- Identificar falhas funcionais e/ou inconsistências.
- Avaliar o comportamento da API ao longo dos fluxos de usuário e administrador.

3. Escopo dos Testes

3.1 Em Escopo

- Autenticação/Login
- Usuários
- Produtos
- Carrinhos
- Testes de segurança

3.2 Fora do Escopo

- Testes de performance
- Testes na Interface frontend

4. Mapeamento dos Endpoints

Módulo	Endpoint	Método	Descrição
Login	/login	POST	Autenticação de usuários



		PUT / DELETE	usuários
Produtos	/produtos	GET / POST / PUT / DELETE	Gestão de produtos
Carrinhos	/carrinhos	GET / POST / DELETE	Gestão de carrinhos

5. Estratégia de Testes

A estratégia de testes adotada é baseada em **Session-Based Test Management (SBTM)**, com a execução de sessões de testes exploratórios contendo missão definida, tempo delimitado e registro sistemático dos achados.

As sessões serão organizadas por **fluxos funcionais**, contemplando:

- Fluxo do **usuário cliente**
- Fluxo do **usuário administrador**

Para apoiar a exploração dos endpoints, será utilizada a **heurística VADER**, uma abordagem exploratória utilizada para orientar a análise de APIs, considerando os **Verbs HTTP (Verbs)** utilizados, as **Ações (Actions)** executadas por cada endpoint, os **Dados (Data)** manipulados nas requisições e respostas, o tratamento de **Erros (Errors)** e o **Controle de Acesso e Perfis (Roles)**. Essa heurística auxilia na identificação sistemática de comportamentos inesperados, falhas funcionais e inconsistências ao longo dos fluxos da aplicação.

6. Técnica de Testes

Para o projeto dos cenários de teste será utilizada a técnica de **Particionamento de Equivalência**, permitindo a definição de classes de dados **válidas e inválidas** para cada endpoint.

A técnica será aplicada dentro dos fluxos de usuário e administrador, garantindo cobertura funcional eficiente com um conjunto reduzido de cenários representativos.

7. Justificativa

O uso do **SBTM** permite estruturar testes exploratórios de forma controlada e rastreável.

A heurística **VADER** orienta a identificação de falhas relacionadas a ações, dados, respostas e permissões.

O **Particionamento de Equivalência** reduz redundância e aumenta a eficiência dos testes.



8. Cenários de Teste

Os cenários foram definidos com base no **Particionamento de Equivalência**, contemplando entradas válidas e inválidas para cada endpoint.

8.1 Cenários Funcionais – Positivos

Login

ID	Endpoint	Método	Descrição	Resultado Esperado
CT01	/login	POST	Login com credenciais válidas	Status 200 – Login realizado

Usuários

ID	Endpoint	Método	Descrição	Resultado Esperado
CT02	/usuarios	GET	Listar usuários	Status 200
CT03	/usuarios/{id}	GET	Consultar usuário existente	Status 200
CT04	/usuarios	POST	Cadastrar usuário válido	Status 201
CT05	/usuarios/{id}	PUT	Atualizar usuário existente	Status 200
CT05.1	/usuarios/{id}	PUT	Criar usuário via PUT	Status 201
CT06	/usuarios/{id}	DELETE	Excluir usuário sem carrinho	Status 200

Carrinhos

ID	Endpoint	Método	Descrição	Resultado Esperado
----	----------	--------	-----------	--------------------



			carrinhos	
CT08	/carrinhos/{id}	GET	Consultar carrinho existente	Status 200
CT09	/carrinhos	POST	Criar carrinho válido	Status 201
CT10	/carrinhos/concluir-compra	DELETE	Concluir compra	Status 200
CT11	/carrinhos/cancelar-compra	DELETE	Cancelar compra	Status 200

Produtos

ID	Endpoint	Método	Descrição	Resultado Esperado
CT12	/produtos	GET	Listar produtos	Status 200
CT13	/produtos/{id}	GET	Consultar produto existente	Status 200
CT14	/produtos	POST	Cadastrar produto (admin)	Status 201
CT15	/produtos/{id}	PUT	Atualizar produto	Status 200
CT16	/produtos/{id}	DELETE	Excluir produto sem vínculo	Status 200

8.2 Cenários Funcionais – Negativos

Login

ID	Endpoint	Método	Descrição	Resultado Esperado
CT17	/login	POST	Realizar login com e-	Status 401 – E-mail e/ou



				Sem inválidos	inválidos
--	--	--	--	------------------	-----------

Usuários

ID	Endpoint	Método	Descrição	Resultado Esperado
CT18	/usuarios/{id}	GET	Consultar usuário inexistente	Status 400 – Usuário não encontrado
CT19	/usuarios	POST	Cadastrar usuário com e-mail já cadastrado	Status 400 – E-mail já cadastrado
CT20	/usuarios/{id}	PUT	Atualizar usuário com e-mail já cadastrado	Status 400 – E-mail já cadastrado
CT21	/usuarios/{id}	DELETE	Excluir usuário com carrinho cadastrado	Status 400 – Usuário com carrinho cadastrado

Carrinhos / Compras

ID	Endpoint	Método	Descrição	Resultado Esperado
CT22	/carrinhos/{id}	GET	Consultar carrinho inexistente	Status 400 – Carrinho não encontrado
CT23	/carrinhos	POST	Criar carrinho com dados inválidos	Status 400 – Algo deu errado
CT24	/carrinhos	POST	Criar carrinho sem autenticação	Status 401 – Token ausente, inválido ou expirado



	oncluir-compra		compra sem autenticação	Token ausente, inválido ou expirado
CT26	/carrinhos/cancelar-compra	DELETE	Cancelar compra sem autenticação	Status 401 – Token ausente, inválido ou expirado

Produtos

ID	Endpoint	Método	Descrição	Resultado Esperado
CT27	/produtos/{id}	GET	Consultar produto inexistente	Status 400 – Produto não encontrado
CT28	/produtos	POST	Cadastrar produto com nome já existente	Status 400 – Já existe produto com esse nome
CT29	/produtos	POST	Cadastrar produto sem autenticação	Status 401 – Token ausente, inválido ou expirado
CT30	/produtos	POST	Cadastrar produto sem perfil administrador	Status 403 – Rota exclusiva para administradores
CT31	/produtos/{id}	PUT	Atualizar produto sem autenticação	Status 401 – Token ausente, inválido ou expirado
CT32	/produtos/{id}	PUT	Atualizar produto sem perfil	Status 403 – Rota exclusiva para



CT33	/produtos/{id}	PUT	Atualizar produto que já faz parte do carrinho	Status 400 - Produto faz parte do carrinho
CT34	/produtos/{id}	DELETE	Excluir produto vinculado a carrinho	Status 400 – Produto faz parte de carrinho
CT35	/produtos/{id}	DELETE	Excluir produto sem autenticação	Status 401 – Token ausente, inválido ou expirado
CT36	/produtos/{id}	DELETE	Excluir produto sem perfil administrador	Status 403 – Rota exclusiva para administr

9. Riscos e Prioridades

Risco	Impacto	Prioridade	Mitigação
Falha no login	Alto	Alta	Priorizar testes de autenticação
Falha no fluxo de carrinho	Alto	Alta	Testar fluxo completo
Aceitar dados inválidos	Médio	Média	Cenários negativos
Acesso indevido	Alto	Média	Testes sem autenticação

10. Critérios de Entrada

Os testes somente poderão ser iniciados quando **todas** as condições abaixo forem atendidas:

- Documentação básica da API ServeRest disponível e acessível.
- Endpoints mapeados e compreendidos conforme comportamento observado da API.



- Ferramenta de testes de API configurada (Postman).
- Definição prévia dos **fluxos de usuário cliente e administrador** a serem testados.
- Definição da **meta mínima de cobertura funcional**, estabelecida em **80% dos endpoints e fluxos da aplicação**.
- Mapa mental dos endpoints disponível para orientar a exploração dos testes.

11. Critérios de Saída

Os testes serão considerados encerrados somente quando **todas** as condições abaixo forem atendidas:

- Execução de testes cobrindo **no mínimo 80% da aplicação**, considerando:
 - Endpoints
 - Métodos HTTP
 - Fluxos de usuário e administrador
- Todos os cenários planejados (positivos e negativos) executados ou devidamente justificados.
- Registro e documentação de todos os defeitos encontrados.
- Consolidação das evidências de teste.
- Geração dos seguintes **artefatos finais**:
 - Collection do Postman contendo os testes executados
 - Evidências das requisições e respostas
 - Lista de bugs identificados
 - Relatórios das sessões exploratórias (SBTM)

12. Definição de Pronto (Definition of Done)

O ciclo de testes será considerado **concluído** quando:

- A cobertura funcional atingir **no mínimo 90% dos fluxos mapeados da aplicação**.
- Todos os riscos críticos identificados forem testados ao menos três vezes.
- Não existirem defeitos críticos abertos sem registro ou análise.
- Todos os artefatos de teste estiverem organizados e versionados.
- O plano de testes e os resultados estiverem revisados.
- O documento final estiver pronto para entrega (PDF).

14. Evidências

- Prints das requisições



- ## 15. Cronograma

Atividade	Início	Fim
Análise da API	02/02/2026	04/02/2026
Elaboração do plano	04/02/2026	05/02/2026
Criação de cenários	06/02/2026	06/02/2026

[illegible]