Plano de testes Serverest

Plano de Teste

Versão 1.0

📜 Histórico de Revisões

Histórico de Revisões			
Versão	**Data**	**Descrição**	**Autor(a)**
1.0	6 de mai. de 2025	Elaboração Inicial	Carolina Hoewell

📑 Sumário

Plano de Teste

1. 📘 Introdução

Este plano de testes detalha a estratégia para a análise aprofundada e a verificação das funcionalidades da API ServeRest. O principal objetivo é assegurar que cada funcionalidade da API opera conforme o esperado, atendendo aos requisitos de negócio e proporcionando uma experiência de integração eficaz e confiável.

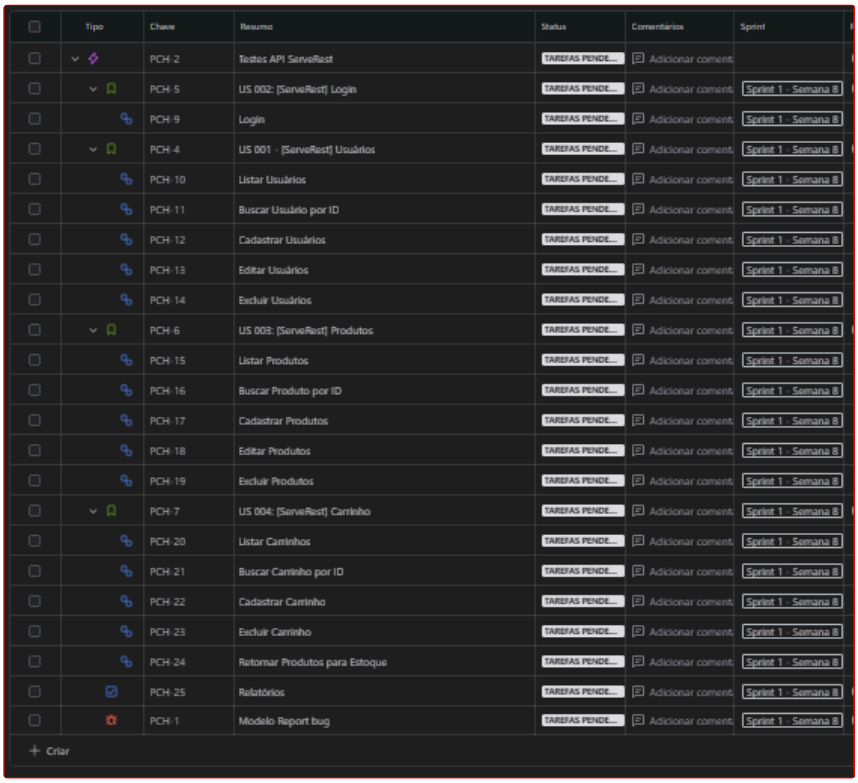
O plano abrangerá a validação individual de cada endpoint e suas respectivas operações (como `GET`, `POST`, `PUT`, `DELETE`), a correta interpretação e processamento dos dados de entrada e saída, o tratamento adequado de diferentes cenários (incluindo

casos de sucesso e de erro), e a conformidade com as especificações da API. Serão empregadas técnicas de teste funcional para verificar o comportamento de cada funcionalidade em detalhes.

1.1 Objetivos do Plano de Testes:

- **Verificar a conformidade funcional:** Garantir que cada funcionalidade da API ServeRest opera de acordo com as especificações e os requisitos definidos, abrangendo o máximo de cenários de uso possíveis.
- **Validar a integridade dos dados:** Assegurar que os dados manipulados pela API (tanto nas requisições quanto nas respostas) sejam consistentes, precisos e estejam em conformidade com os modelos de dados estabelecidos.
- **Avaliar o tratamento de erros:** Verificar se a API lida com erros de forma adequada e informativa, retornando mensagens claras e códigos de status apropriados para diferentes situações de falha.
- **Medir o desempenho da API:** Avaliar a capacidade da API de responder a requisições dentro de limites aceitáveis de tempo de resposta e de suportar uma determinada carga de requisições simultâneas, identificando possíveis gargalos de desempenho.
- **Garantir a segurança da API:** Identificar e mitigar possíveis vulnerabilidades de segurança, como falhas de autenticação, autorização inadequada e proteção contra injeções de dados.
- **Documentar os resultados dos testes:** Registrar de forma clara e detalhada os resultados de cada teste, incluindo os casos de sucesso, as falhas encontradas e as evidências correspondentes, para facilitar a análise e a correção de problemas.
- **Fornecer informações para a tomada de decisão:** Apresentar um panorama claro da qualidade da API ServeRest, fornecendo dados objetivos que auxiliem as equipes de desenvolvimento e de produto na tomada de decisões sobre a implantação e as futuras melhorias da API.
- **Aumentar a confiança na API:** Assegurar que a API ServeRest seja uma solução confiável, garantindo a qualidade do sistema antes da entrega ao cliente.

2. Histórias de usuários



Estrutura de histórias de usuário

3.📌 Escopo do Projeto

2.1 - Dentro do Escopo

2.1.1 - Requisitos a Testar

2.1.1.1 - Requisitos Funcionais

Identificador do Caso de Uso	Nome do Caso de Uso	Descrição
RF01	Login	Validar autenticação de usuários.
RF02	Listar Usuários	Validar listagem de usuários cadastrados.
RF03	Buscar Usuário por ID	Validar ID de usuários cadastrados.
RF04	Cadastrar Usuários	Validar cadastro de usuários pelo administrador.
RF05	Editar Usuários	Validar a edição de usuários cadastrados.
RF06	Excluir Usuários	Validar a exclusão de usuários pelo administrador.
RF07	Listar Produtos	Validar listagem de produtos cadastrados.
RF08	Buscar Produto por ID	Validar ID de produtos cadastrados.
RF09	Cadastrar Produtos	Validar cadastro de produtos pelo administrador.
RF10	Editar Produtos	Validar a edição de produtos cadastrados.
RF11	Excluir Produtos	Validar a exclusão de produtos pelo administrador.
RF12	Listar Carrinhos	Validar listagem de carrinhos cadastrados.
RF13	Buscar Carrinho por ID	Validar ID de carrinhos cadastrados.
RF14	Cadastrar Carrinho	Validar cadastro de carrinho pelo usuário.
RF15	Excluir Carrinho	Validar a exclusão de carrinho pelo usuário.
RF16	Retornar Produtos para Estoque	Validar o retorno de produotas para o estoque após exclusão de carrinho.
RF17	Relatórios	Validar geração e exibição de relatórios.

2.1.1.2 - Requisitos Não-Funcionais

Identificador do Requisito	Nome do Requisito	Descrição
RNF01	Desempenho	Tempo de resposta deve ser inferior a 2 segundos.
RNF02	Segurança	Dados sensíveis devem ser criptografados.
RNF03	Usabilidade	Interface intuitiva e responsiva.
RNF04	Compatibilidade	Funcionamento em navegadores modernos.
RNF05	Acessibilidade	Usabilidade para usuários com deficiência conforme as diretrizes WCAG 2.1.

2.2 - Fora do Escopo

- **Módulos**: Pagamentos (gateways) e Localização (traduções).
- **Dispositivos**: TVs, relógios, IoT.
- **Sistemas**: Windows <10, Android <11, iOS <15.
- **Ferramentas**: Emuladores não oficiais, VPNs.

🔍 3 - Tipos de Teste

- Testes funcionais: para testar os requisitos: RF01, RF02, RF03, RF04, RF05, RF06, RF07, RF08, RF09, RF10, RF11, RF12, RF13, RF14, RF15, RF16, RF17.

Objetivo	**Verificar se cada funcionalidade retorna o resultado esperado.**			
Técnica	( ) Manual		(**X**) Automática	
Estágio do Teste	Integração ()	Sistema (**X**)	Unidade (**X**)	Aceitação ( )
Abordagem	Caixa Branca (**X**)		Caixa Preta (**X**)	
Responsável(is)				

- Testes de desempenho: para testar o requisito RNF01.

Objetivo	**Garantir tempos de resposta aceitáveis sob carga normal.**			
Técnica	( ) Manual		(**X**) Automática	
Estágio do Teste	Integração ( )	Sistema (**X**)	Unidade ( )	Aceitação ( )
Abordagem	Caixa Branca (**X**)		Caixa Preta ( )	
Responsável(is)				

- Testes de segurança: para testar o requisito RNF02.

Objetivo	**Validar proteção contra vulnerabilidades.**

Técnica	(**X**) Manual		( ) Automática	
Estágio do Teste	Integração ( )	Sistema (**X**)	Unidade ( )	Aceitação ( )
Abordagem	Caixa Branca ( )		Caixa Preta (**X**)	
Responsável(is)				

- Teste de usabilidade: para testar o requisito RNF03.

Objetivo	**Validar a usabilidade das interfaces.**			
Técnica	(**X**) Manual		( ) Automática	
Estágio do Teste	Integração ( )	Sistema (**X**)	Unidade ( )	Aceitação (**X**)
Abordagem	Caixa Branca ( )		Caixa Preta (**X**)	
Responsável(is)				

- Teste de portabilidade/compatibilidade: para testar o requisito RNF04.

Objetivo	**Validar a compatibilidade do sistema com diversos ambientes.**			
Técnica	(**X**) Manual		( ) Automática	
Estágio do Teste	Integração ( )	Sistema (**X**)	Unidade ( )	Aceitação (**X**)
Abordagem	Caixa Branca ( )		Caixa Preta (**X**)	
Responsável(is)				

- Teste de acessibilidade: para testar o requisito RNF05.

Objetivo	**Validar a acessibilidade das interfaces.**			
Técnica	(**X**) Manual		( ) Automática	
Estágio do Teste	Integração ( )	Sistema (**X**)	Unidade ( )	Aceitação (**X**)
Abordagem	Caixa Branca ( )		Caixa Preta (**X**)	
Responsável(is)				

🧰 4 - Recursos

4.1 - Ambiente de Teste

- **Hardware**:
  - Desktop/Notebook: processador: 4vCPUs, 8GB RAM ou superior.
- **Software**:
  - SO Desktop/Notebook: Windows 10/11
  - SO Dispositivos Móveis: Android 11 (Red Velvet Cake) e IOS 15 ou superiores.
  - Navegadores Desktop/Notebook: Chrome, EDGE e Firefox.

- Navegadores Dispositivos Móveis: Chrome (Android) e Safari (IOS).

4.2 - Ferramentas

- **Funcionais**:
  - Chrome, EDGE, Firefox, (para testes de interface).
  - Inspeção de elementos do navegador (F12).
- **Não-funcionais**:
  - Cronômetro para medição manual de tempos.
  - Planilha para registro de métricas (Confluence).
  - BrowerStack para testes de compatibilidade.
  - Lighthouse para teste de desempenho.
- **Gerenciamento**:
  - Xmind para mapa mental.
  - Confluence.
  - Jira.
  - Microsoft Teams para comunicação.

3. 🌐 Cenário Operacional

A aplicação é uma API Rest que simula um site de e-commerce, permitindo cadastro de usuários padrão e admin. Está disponível publicamente, tendo sido acessada via navegador desktop. Os testes serão executados manualmente e através de automações.

4. ⚖️ Premissas e Restrições

Premissas:

- A aplicação estará online durante as sessões de teste.
- Deverão ser realizada a maior cobertura possível de testes.
- Os testes deverão ser executados utilizando a documentação da APi como guia.
- As sessões de teste serão realizadas em navegadores atualizados.

Restrições:

- A aplicação é pública e não controlada pela equipe de teste.

5. 🧭 Abordagem dos Testes

5.1 Tipo de Teste

- Teste Funcional (Caixa Preta).
- Teste Exploratórios (por sessão, guiado por técnicas específicas).
- Testes automatizados guiados por swagger

5.2 Técnicas de Teste

- Exploração baseada em objetivo (Charter).
- Inspeção dos fluxos por heurísticas de navegação.
- Ações baseadas em CRUD, múltiplos usuários, interface, requisições e marcação.

5.3 Ferramentas e Estratégias de Teste

- Navegador: Google Chrome / Mozilla Firefox / Microsoft EDGE.
- Ferramenta de captura de tela.

- Postman, Confluence e Jira.

5.4 Estratégia de Testes Exploratórios

Cada membro da Squad utilizou uma técnica exploratória diferente, guiando sua sessão por um objetivo específico:

- **Multi-User e Bookmark:**
  Exploração da aplicação utilizando múltiplos usuários simulando diferentes níveis de acesso e salvando elementos para avaliação posterior.
- **CRUD:**
  A navegação e testes foram guiados pelas quatro ações principais do sistema: Criar, Ler, Atualizar e Deletar. Essa técnica permitiu verificar a completude e consistência das ações básicas do sistema.
- **ALTERFACE:**
  A técnica ALTERFACE permite explorar mudanças rápidas e inesperadas de contexto, verificando como a interface responde a ações fora do esperado, como mudanças bruscas de aba, redimensionamento de tela e manipulações não lineares.
- **Requisições:**
  Testes guiados pela documentação swagger fornecida pela API. Observando reações da aplicação ao lidar com parâmetros malformados, repetição de ações e comportamento da resposta visual.

Cada técnica foi escolhida para ampliar a cobertura da aplicação, maximizando a chance de encontrar falhas que impactem diretamente a experiência do usuário.

As sessões foram registradas com evidências visuais e descrições textuais para discussão e apresentação final.

✅ 6 - Completude dos Testes

- Todos os casos de teste planejados serem devidamente executados.
- Testes de regressão serão executados após correções de bugs.
- O cronograma pode ser ajustado conforme necessidade do projeto.
- Cada membro da Squad executou sua técnica com duração de até 45 minutos.
- Todos os pontos principais da aplicação foram acessados durante os testes.
- Pelo menos 1 comportamento relevante identificado por sessão.

🔄 7 - Priorização dos Testes em Caso de Mudança de Escopo

- Reunir PO, SM, Devs e QA
- Reavaliar:
  - Testes críticos (não podem ser cortados)
  - Novos testes necessários.
- Ajustar plano e prazos.

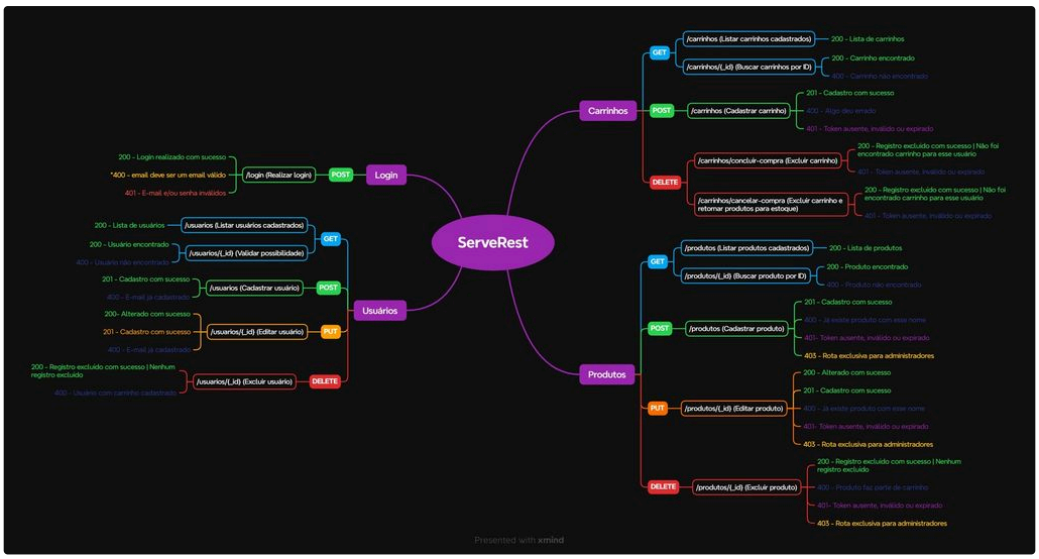
8.📂 Artefatos

8.1 Entrada

Acesso ao site: [Front - ServeRest]

Página inicial da API ServRest

Mapa Mental do site:



Mapa mental da API ServeRest

10. 📢 Gerenciamento de Comunicação

- *Atualização diária no Confluence.*
- Comunicação via Teams.
- Daily Squad.

11. 📅 Cronograma e Marcos

Etapa	Data
Elaboração do Plano de Teste	6 de mai. de 2025
Definição das Técnicas	6 de mai. de 2025
Execução dos Testes Exploratórios	8 de mai. de 2025
Elaboração dos Reports dos Artefatos de Saída	9 de mai. de 2025

Apresentação dos Reports dos Artefatos de Saída	9 de mai. de 2025

12. ⚠️ Riscos

- Instabilidade na comunicação com o site durante a execução dos testes.
- Diferença de interpretação entre testadores sobre funcionalidades.
- Registro incompleto de falhas por falta de documentação de apoio.

13. ✅ Aprovações

Aprovações		
Participante	**Assinatura**	**Data**
Carlos Leonardo Alves Novaes		
Jacques de Jesus Figueiredo Schmitz Junior		
Amanda Cardoso de Almeida		