Planejamento de Testes - Módulo PRODUTOS

1. Apresentação

Este documento detalha o planejamento de testes para o módulo de Produtos da API ServeRest, com o objetivo de validar os requisitos da User Story US 003. O plano aborda o ciclo de vida completo dos produtos (CRUD) e suas regras de negócio, como a necessidade de autenticação e a validação de dependências com outros módulos.

2. Objetivo

O objetivo principal é garantir a qualidade, segurança e conformidade do CRUD de Produtos. Os testes visam validar que apenas usuários autenticados podem gerenciar produtos, que as regras de negócio (como unicidade de nome e dependências) são respeitadas, e que a API se comporta conforme o esperado em todos os cenários.

3. Escopo

Não pode faltar:	• Testar todos os endpoints do CRUD de Produtos (/produtos). • Validar a camada de autorização (necessidade de token Bearer) para todas as operações. • Testar as regras de negócio: unicidade de nome de produto, impossibilidade de excluir produto em um carrinho. • Validar o comportamento específico do PUT (criar se não existir). • Gerenciar todo o ciclo de vida dos testes no QALity (Jira). • Automatizar os cenários de regressão com Robot Framework.
É bom ter:	• Testes de busca e filtragem de produtos. • Validação dos tipos de dados para cada campo do produto.

Fora do escopo:	• Testes dos módulos de Usuários e Login (são pré-condições). • Testes aprofundados do módulo de Carrinhos (usado apenas para validar a dependência). • Testes de performance do endpoint de produtos.

4. Análise

A estratégia será dividida em fases para garantir uma abordagem estruturada e eficiente:

Fase 1: Testes Manuais e Exploratórios

Ferramenta: Postman.

Objetivo: Realizar a primeira rodada de execução baseada nos casos de teste definidos. O foco é a validação funcional inicial, a descoberta de defeitos óbvios e a compreensão aprofundada do comportamento da API, indo além da documentação (Swagger).

Fase 2: Refinamento e Desenvolvimento da Automação

Ferramenta: Robot Framework.

Objetivo: Após a validação manual, os testes candidatos à automação serão desenvolvidos. Esta fase foca em criar uma suíte de testes robusta que possa ser executada rapidamente para garantir a regressão.

Fase 3: Execução Automatizada e Relatórios

Ferramenta: Robot Framework, QALity, EC2.

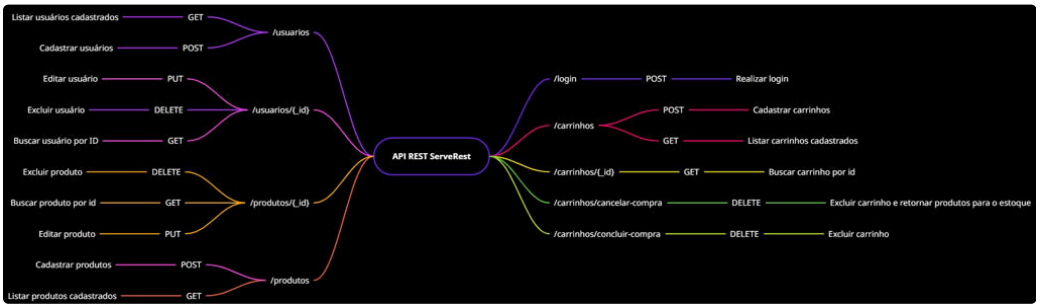
Objetivo: Executar a suíte de testes automatizados a partir da instância EC2 dedicada, garantindo que novas alterações não introduziram defeitos. Os resultados serão reportados e gerenciados no QALity.

5. Técnicas aplicadas

- Testes Baseados em Requisitos: Mapeamento direto dos Critérios de Aceitação para os casos de teste.
- Particionamento de Equivalência: Uso de tokens válidos e inválidos; nomes de produtos novos e duplicados.
- Teste de Transição de Estado: Validação do ciclo de vida de um produto: criado → listado → atualizado → excluído.

- Análise de Valor Limite: Teste de campos numéricos como preço e quantidade com valores limite (ex: 0, valor máximo).

6. Mapa mental da aplicação



7. Cenários de Testes

- Cenário: Autorização de Acesso - US003-CE1

ID do Caso de Teste	Descrição	Pré-condições	Dados de Entrada	Resultado Esperado	Prioridade
US003-CE1-CT1	Tentar cadastrar um produto sem autenticação	N/A	POST /produtos sem token de autorização	Status 401 Unauthorized. Mensagem "Token de acesso ausente, inválido ou expirado".	Alta
US003-CE1-CT2	Tentar listar produtos sem autenticação	Massa de dados com os produtos cadastrados	GET /produtos sem token de autorização	A API permite listar sem token, então o resultado esperado é Status 200 OK.	Média

US003-CE1-CT3	Tentar excluir um produto sem autenticação	Produto cadastrado	DELETE /produtos/{id} sem token de autorização	Status 401 Unauthorized. Mensagem "Token de acesso ausente, inválido ou expirado".	Alta

- Cenário: Autorização de Acesso - US003-CE2

ID do Caso de Teste	Descrição	Pré-Condições	Dados de Entrada	Resultado Esperado	Prioridade
US003-CE2-CT1	Cadastrar um novo produto com sucesso	Usuário autenticado	Token válido, dados de produto válidos e nome único	Status 201 Created. Mensagem "Cadastro realizado com sucesso".	Alta
US003-CE2-CT2	Tentar cadastrar produto com nome já existente	Produto "Logitech MX Master 3" já existe	Token válido, dados de produto com nome "Logitech MX Master 3"	Status 400 Bad Request. Mensagem "Já existe produto com esse nome".	Alta

- Cenário: Atualização de Produtos (PUT) - US003-CE3

ID do Caso de Teste	Descrição	Pré-Condições	Dados de Entrada	Resultado Esperado	Prioridade
US003-CE3-CT1	Criar um novo produto via PUT com ID inexistente	Usuário autenticado	Token válido, ID de produto não existente, dados de produto válidos	Status 201 Created. Mensagem "Cadastro realizado com sucesso".	Alta
US003-CE3-CT2	Tentar criar produto via PUT com nome já existente	Produto "Logitech MX Master 3" já existe	Token válido, ID de produto não existente, dados com nome "Logitech MX Master 3"	Status 400 Bad Request. Mensagem "Já existe produto com esse nome".	Alta

- Cenário: Exclusão de Produtos (DELETE) - US003-CE4

ID do Caso de Teste	Descrição	Pré-Condições	Dados de Entrada	Resultado Esperado	Prioridade
US003-CE4-CT1	Excluir um produto com sucesso	Usuário autenticado; produto existe e não está em nenhum carrinho	Token válido, ID do produto a ser excluído	Status 200 OK. Mensagem "Registro excluído com sucesso".	Alta

US003-CE4-CT2	Tentar excluir produto que está em um carrinho	Usuário autenticado; produto foi adicionado a um carrinho	Token válido, ID do produto que está no carrinho	Status 400 Bad Request. Mensagem "Não é permitido excluir produto que faz parte de carrinho".	Alta

8. Priorização da execução dos cenários de teste

A priorização da execução dos cenários de teste será baseada em uma combinação de fatores, incluindo a criticidade da funcionalidade, o risco associado, a frequência de uso esperada e a dependência de outros módulos. A prioridade de cada caso de teste já está indicada nas tabelas de cenários (Alta, Média, Baixa).

- Prioridade Alta: Casos de teste que cobrem funcionalidades críticas, cenários de alto risco (segurança, falhas que impedem o uso principal do sistema) e fluxos de usuário principais. Estes serão executados primeiro e com maior frequência.
- Prioridade Média: Casos de teste que cobrem funcionalidades importantes, cenários de risco moderado e fluxos alternativos. Serão executados após os de alta prioridade.
- Prioridade Baixa: Casos de teste que cobrem funcionalidades menos críticas, cenários de baixo risco ou casos de borda menos prováveis. Serão executados quando houver tempo e recursos disponíveis.

9. Matriz de Risco

Risco	Impacto	Probabilidade	Mitigação	Contigência
Acesso e manipulação não autorizada de produtos	Crítico	Baixa	• Cobertura total dos cenários de autorização,	• Invalidação imediata de todos os tokens de sessão. •

			testando todas as rotas protegidas com e sem token válido.	Análise de logs para identificar acessos indevidos.
Falha ao cadastrar/atualizar produtos (ex: nome duplicado)	Alto	Média	• Testes automatizados que validam as regras de unicidade para POST e PUT.	• Correção emergencial (hotfix). • Comunicação com os usuários para orientar sobre a falha.
Exclusão de produto com dependência (em carrinho)	Alto	Média	• Testes de integração que simulam a adição de um produto a um carrinho antes de tentar a exclusão.	• Rollback da versão da API. • Script para correção manual do banco de dados, se necessário.
Criação incorreta de produto via PUT	Média	Média	• Casos de teste específicos para validar o	• Correção emergencial (hotfix). • Ferrament

			comportamento de "criar se não existir" do PUT.	as para limpeza de dados duplicados ou incorretos.

10. Cobertura de testes

- **Path Coverage (input)**

Testes Automatizados	5
Quantidade de Endpoints	5
Cobertura	100%

- **Operator Coverage (input)**

Quantidade de operações da API estão automatizados	5
Quantidade total de operações da API REST	5
Cobertura	100%

- **Parameter Coverage (input)**

Quantidade total de parâmetros cobertos na suítes de testes	5
Quantidade total de parâmetros nos métodos da API	5
Cobertura	100%

- **Parameter Value Coverage (input)**

Quantidade total de valores diferentes enviados	15
Quantidade total de valores que podem assumir.	15

Cobertura	100%

- **Content-Type Coverage (input e output)**

Quantidade total de content-type em cada operação cobertos pela suíte de testes	1
Quantidade total de content-type em todas as operações da API	1
Cobertura	100%

- **Operation Flow (input)**

Fluxos Possíveis	2
Fluxos Automatizados	2
Cobertura	100%

- **Response Properties Body Coverage (Output)**

Número total de todas as propriedades de todos os objetos que pode ser obtido na resposta da API	7
Número de propriedades da resposta que os testes estão cobrindo	7
Cobertura	100%

- **Status Code Coverage (Output)**

Status codes da API	4
Status codes cobertos na API	4
Cobertura	100%

11. Testes candidatos a automação

- US003-CE1-CT1: Validação de autorização no POST.
- US003-CE2-CT1: Cadastro de produto (caminho feliz).

- US003-CE2-CT2: Tentativa de cadastro com nome duplicado.
- US003-CE3-CT1: Criação de produto via PUT (regra de negócio chave).
- US003-CE4-CT1: Exclusão de produto (caminho feliz).
- US003-CE4-CT2: Tentativa de exclusão de produto com dependência.

12. Ferramentas, Ambiente e Infraestrutura

- Gerenciamento e Versionamento:
  - Jira com QALity: Para gestão completa dos casos de teste, planejamento de ciclos e reporte de defeitos.
  - Git: Para controle de versão de toda a documentação (Plano de Testes) e do código-fonte da automação.
- Ferramentas de Teste:
  - Postman: Utilizado para os testes manuais e exploratórios.
  - Robot Framework: Ferramenta principal para o desenvolvimento dos testes automatizados.
- Infraestrutura e Ambiente de Teste:
  - O ambiente será configurado em duas instâncias EC2 na AWS para simular um ambiente mais realista.
  - Configuração: Amazon Linux com Node.js e NPM para hospedar a API ServeRest.

13. Cronograma

Atividade	Início	Término
Planejamento de Testes	8 de set. de 2025	9 de set. de 2025
Sessão de Testes	9 de set. de 2025	11 de set. de 2025
Relatório de Testes	12 de set. de 2025	12 de set. de 2025

14. Referências

- Test Coverage Criteria for RESTful Web APIs - Alberto Martin-Lopez, Sergio Segura e Antonio Ruiz-Cortés
- ISTQB Certified Tester Foundation Level (CTFL) 4.0 Syllabus
- ISO/IEC 25010:2011 – System and Software Quality Models
- ISO/IEC/IEEE 29119 – Software Testing
- IEEE 829 – Test Documentation

- User Story: US 003 - [API] Produtos
- Documentação da API ServeRest (Swagger)