PLANO DE TESTE

API ServeRest

Thalis Bruno Stertz

2022

ÍNDICE

Plano de teste

API ServeRest v2.26.1 disponível em: https://serverest.dev/#/

1. Introdução

Este documento descreve os testes de diferentes funções na base do código da API ServeRest, que é uma API REST gratuita que tem como objetivo simular uma loja online servindo como material de estudo para QA 's em desenvolvimento.

Os testes representaram situações específicas para garantir que as funções da API funcionem dentro dos parâmetros planejados e esperados, os requisitos testados e os tipos de recursos específicos para cada iteração da API.

2. Objetivos

Efetuar testes na API ServeRest para observar suas funcionalidades, validar se as respostas recebidas estão de acordo com a requisição e se a API possui o conjunto de informações que validam estas ações.

3. Escopo

O escopo deste plano de teste apresenta todas as realizações de teste funcionais que serão realizadas mediante as rotas da aplicação. Para cada rota será realizado testes positivos e negativos, presentes no mapa mental. Como a API ServeRest é pensada para ser uma ferramenta de aprendizado, testes de carga serão dispensados. Além dos testes de carga, testes não funcionais não serão realizados.

Para criar os testes, será realizado o método Cypress através do Visual Studio Code. Os testes serão guiados pelas informações disponíveis no Swagger da aplicação. As medidas críticas que devem ser observadas e validadas ao utilizar o Cypress após os testes são:

1. Os dados de retorno;
2. Validar se a resposta está de acordo;
3. Validar se a funcionalidade está funcionando de acordo;
4. Validar se quando der erro o status está de acordo com os códigos de erro;
5. Validar se a aplicação funciona como deveria.

4. Mapa mental

Disponível em anexo 1.

5. Suíte de casos de teste

Aqui se encontram os testes que serão realizados na API.

5.1 Testes Funcionais

Os teste funcionais foram separados em dois: testes positivos e negativos.

5.1.1 Testes positivos

Os testes positivos foram separados em dois: testes regressivos e de fluxo completo

5.1.1.1 Testes regressivos

Nome	**CTFP01 -** Realizar login
Descrição	Deve realizar login com sucesso
Requisitos	Email e senha de um usuário existente
Etapas do teste	rota /login com o método POST

Asserção	{
	"message": "Login realizado com sucesso",
	"authorization": "Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJlbWFpbCI6ImZ1bGFub0BxYS5jb20iLCJwYXNzd29yZCI6InRlc3RlIiwiaWF0IjoxNTg5NzU4NzQ2LCJleHAiOjE1ODk3Njg3NDZ9.B6TASHV8k9xBerz4NSeFBlAZGSDhZlqESt767M0567I"
	}

Status esperado	200

Nome	**CTFP02 -** Listar usuários cadastrados
Descrição	Deve listar todos os usuários cadastrados
Etapas do teste	rota /usuarios com o método GET

Asserção	{
	"quantidade": 1,
	"usuarios": [
	{
	"nome": "Fulano da Silva",
	"email": "beltrano@qa.com.br",
	"password": "teste",
	"administrador": "true",
	"_id": "0uxuPY0cbmQhpEz1"
	}
	]
	}

Status esperado	200

Nome	**CTFP03 -** Cadastrar usuário
Descrição	Deve cadastrar um novo usuário com sucesso
Requisitos	Nome, email, senha e se é administrador
Etapas do teste	rota /usuarios com o método POST

Asserção	{
	"message": "Cadastro realizado com sucesso",
	"_id": "jogfODIlXsqxNFS2"
	}

Status esperado	201

Nome	**CTFP04 -** Buscar usuário por ID

Descrição	Deve buscar um usuário pelo seu ID
Requisitos	ID do usuário
Etapas do teste	rota /usuários/{_di} com o método GET

Asserção	{
	"nome": "Fulano da Silva",
	"email": "beltrano@qa.com.br",
	"password": "teste",
	"administrador": "true",
	"_id": "0uxuPY0cbmQhpEz1"
	}

Status esperado	200

Nome	**CTFP05 -** Excluir usuário
Descrição	Deve excluir um usuário com sucesso
Requisitos	ID do usuário
Etapas do teste	rota /usuários/{_di} com o método DELETE

Asserção	{
	"message": "Registro excluído com sucesso | Nenhum registro excluído"
	}

Status esperado	200

Nome	**CTFP06 -** Editar usuário
Descrição	Deve editar um usuário com sucesso
Requisitos	ID do usuário
Etapas do teste	rota /usuários/{_di} com o método PUT

Asserção	{
	"message": "Registro alterado com sucesso"
	}

Status esperado	200

Asserção	{
	"message": "Cadastro realizado com sucesso",
	"_id": "jogfODIlXsqxNFS2"
	}

Status esperado	201

Nome	**CTFP07 -** Listar produtos cadastrados
Descrição	Deve listar todos os produtos cadastrados
Etapas do teste	rota /produtos com o método GET

Asserção	{
	"quantidade": 2,
	"usuarios": [
	{
	"nome": "Logitech MX Vertical",
	"preco": 470,
	"descricao": "Mouse",
	"quantidade": 381,
	"_id": "BeeJh5lz3k6kSIzA"
	},
	{
	"nome": "Samsung 60 polegadas",
	"preco": 5240,
	"descricao": "TV",
	"quantidade": 49977,
	"_id": "K6leHdftCeOJj8BJ"
	}
	]
	}

Status esperado	200

Nome	**CTFP08 -** Cadastrar produto
Descrição	Deve cadastrar um novo produto com sucesso
Requisitos	Nome, preço, descrição e a quantidade
Etapas do teste	rota /produtos com o método POST

Asserção	{
	"message": "Cadastro realizado com sucesso",
	"_id": "jogfODIlXsqxNFS2"
	}

Status esperado	201

Nome	**CTFP09 -** Buscar produto por ID
Descrição	Deve buscar um produto pelo seu ID
Requisitos	ID do produto
Etapas do teste	rota /produtos/{_id}com o método GET

Asserção	{
	"nome": "Logitech MX Vertical",
	"preco": 470,
	"descricao": "Mouse",
	"quantidade": 381,
	"_id": "BeeJh5lz3k6kSIzA"
	}

Status esperado	200

Nome	**CTFP10 -** Editar produto
Descrição	Deve editar um produto com sucesso
Requisitos	ID do produto
Etapas do teste	rota /produtos/{_id} com o método PUT

Asserção	{
	"message": "Registro alterado com sucesso"
	}

Status esperado	200

Asserção	{
	"message": "Cadastro realizado com sucesso",
	"_id": "jogfODIlXsqxNFS2"
	}

Status esperado	201

Nome	**CTFP11 -** Excluir produto
Descrição	Deve excluir um produto com sucesso
Requisitos	ID do produto
Etapas do teste	rota /produtos/{_id} com o método DELETE

Asserção	{
	"message": "Registro excluído com sucesso | Nenhum registro excluído"
	}

Status esperado	200

Nome	**CTFP12 -** Listar carrinhos cadastrados

Descrição	Deve listar todos os carrinhos cadastrados
Etapas do teste	rota /carrinhos com o método GET

Asserção	{
	"quantidade": 1,
	"carrinhos": [
	{
	"produtos": [
	[
	{
	"idProduto": "BeeJh5lz3k6kSIzA",
	"quantidade": 1,
	"precoUnitario": 470
	},
	{
	"idProduto": "K6leHdftCeOJj8BJ",
	"quantidade": 2,
	"precoUnitario": 5240
	}
	]
	],
	"precoTotal": 10950,
	"quantidadeTotal": 3,
	"idUsuario": "oUb7aGkMtSEPf6BZ",
	"_id": "aFOUqntef4iaOwWfg"
	}
	]
	}

Status esperado	200

Nome	**CTFP13 -** Cadastrar carrinho
Descrição	Deve cadastrar carrinho com sucesso
Requisitos	ID do produto e quantidade
Etapas do teste	rota /carrinhos com o método POST
Asserção	{

	"message": "Cadastro realizado com sucesso", "_id": "jogfODIlXsqxNFS2" }
Status esperado	201

Nome	**CTFP14 -** Buscar carrinho por ID
Descrição	Deve buscar um carrinho pelo ID
Requisitos	ID do carrinho
Etapas do teste	rota /carrinhos/{_id} com o método GET

Asserção	{
	"produtos": [
	[
	{
	"idProduto": "BeeJh5lz3k6kSIzA",
	"quantidade": 2,
	"precoUnitario": 470
	},
	{
	"idProduto": "K6leHdftCeOJj8BJ",
	"quantidade": 1,
	"precoUnitario": 5240
	}
	]
	],
	"precoTotal": 6180,
	"quantidadeTotal": 3,
	"idUsuario": "oUb7aGkMtSEPf6BZ",
	"_id": "qbMqntef4iTOwWfg"
	}

Status esperado	200

Nome	**CTFP15 -** Concluir compra

Descrição	Deve excluir carrinho com sucesso
Etapas do teste	rota /carrinhos/concluir-compra com o método DELETE

Asserção	{
	"message": "Registro excluído com sucesso | Não foi encontrado carrinho para esse usuário"
	}

Status esperado	200

Nome	**CTFP16 -** Cancelar compra
Descrição	Deve retornar os produtos para o estoque e excluir o carrinho
Etapas do teste	rota /carrinhos/cancelar-compra com o método DELETE

Asserção	{
	"message": "Registro excluído com sucesso | Não foi encontrado carrinho para esse usuário"
	}

Status esperado	200

Nome	**CTFP017 -** Buscar e salvar em um arquivo JSON
Requisitos	Usuário cadastrado
Ação esperada	Usuário salvo

Nome	**CTFP18 -** Buscar usuário de um arquivo JSON

Requisitos	Usuário cadastrado
Ação esperada	Arquivo encontrado

5.1.1.2 Testes de fluxo completo

Para usuário admin:

Nome	CTFP01
Descrição	Deve cadastrar usuário, logar no usuário, buscar produto, excluir carrinho se existir, cadastrar carrinho e concluir compra
Etapas do teste	1. rota /usuarios com método POST 2. rota /login com o método POST 3. rota /produtos/{_id} com o método GET 4. rota /carrinhos/cancelar-compra com método DELETE 5. rota /carrinhos com método POST 6. rota /carrinhos/concluir-compra com método DELETE
Asserção	Todos os passos devem ser realizados com sucesso.
Status esperado	200

Nome	CTFP02
Descrição	Deve cadastrar usuário admin, logar no usuário, buscar produto pelo ID e editar produto
Etapas do teste	1. rota /usuarios com método POST 2. rota /login com o método POST 3. rota /produtos/{_id} com o método GET 4. rota/produtos/{_id} com o método PUT

Asserção	Todos os passos devem ser realizados com sucesso.
Status esperado	200

Nome	CTFP03
Descrição	Deve cadastrar usuário admin, logar no usuário, buscar produto pelo ID e excluir produto
Etapas do teste	
Asserção	Todos os passos devem ser realizados com sucesso.
Status esperado	200

Para usuário não admin:

Nome	CTFP04
Descrição	Deve cadastrar usuário não administrador, logar no usuário, buscar produto, excluir carrinho se existir, cadastrar carrinho e concluir compra
Etapas do teste	1. rota /usuarios com método POST 2. rota /login com o método POST 3. rota /produtos/{_id} com o método GET 4. rota /carrinhos/cancelar-compra com método DELETE 5. rota /carrinhos com método POST 6. rota /carrinhos/concluir-compra com método DELETE
Asserção	Todos os passos devem ser realizados com sucesso.
Status esperado	200

Nome	CTFP05
Descrição	Deve cadastrar usuário não administrador, logar no usuário, buscar produto pelo ID e editar produto
Etapas do teste	1. rota /usuarios com método POST 2. rota /login com o método POST 3. rota /produtos/{_id} com o método GET 4. rota/produtos/{_id} com o método PUT
Asserção	O quarto passo deve falhar
Status esperado	403

5.1.2 Testes negativos

Nome	**CTFN01 -** Não realizar login
Descrição	Não deve realizar login
Requisitos	Email e senha inválidos
Etapas do teste	rota /login com o método POST

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

Status esperado	400

Nome	**CTFN02 -** Não cadastrar usuário
Descrição	Não deve cadastrar usuário
Requisitos	Email já cadastrado

Etapas do teste	rota /usuarios com o método POST

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

Status esperado	400

Nome	**CTFN03 -** Não buscar usuário pelo ID
Descrição	Não deve buscar usuário pelo seu ID
Requisitos	ID do usuário
Etapas do teste	rota /usuarios com o método GET

Asserção	{
	"message": "Usuário não encontrado"
	}

Status esperado	400

Nome	**CTFN04 -** Não excluir usuário
Descrição	Não deve excluir usuário com carrinho cadastrado
Requisitos	ID do usuário
Etapas do teste	rota /usuarios com o método DELETE

Asserção	{
	"message": "Não é permitido excluir usuário com carrinho cadastrado",
	"idCarrinho": "qbMqntef4iTOwWfg"
	}

Status esperado	400

Nome	**CTFN05 -** Não editar usuário
Descrição	Não deve editar usuário utilizando email já cadastrado
Requisitos	ID do usuário
Etapas do teste	rota /usuarios com o método PUT

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

Status esperado	400

Nome	**CTFN06 -** Não cadastrar produto
Descrição	Não deve cadastrar produto com nome já cadastrado
Etapas do teste	rota /produtos com o método POST

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

Status esperado	400

Nome	**CTFN07 -** Não cadastrar produto
Descrição	Não deve cadastrar produto (Token de acesso ausente, inválido ou expirado)
Etapas do teste	rota /produtos com o método POST

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

Status	401

esperado	

Nome	**CTFN08 -** Não cadastrar produto
Descrição	Não deve cadastrar produto (Rota exclusiva para administradores)
Etapas do teste	rota /produtos com o método POST

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

Status esperado	403

Nome	**CTFN09 -** Não buscar produto
Descrição	Não deve buscar um produto pelo seu ID
Requisitos	ID do produto
Etapas do teste	rota /produtos com o método GET

Asserção	{
	"message": "Produto não encontrado"
	}

Status esperado	400

Nome	**CTFN10 -** Não excluir produto
Descrição	Não deve excluir produto (Produto faz parte de carrinho)
Requisitos	ID do produto
Etapas do teste	rota /produtos com o método DELETE

Asserção	{
	"message": "Não é permitido excluir produto que faz parte de carrinho",
	"idCarrinho": [
	"qbMqntef4iTOwWfg, yILJY1eaAUC6hyRc"
	]
	}

Status esperado	400

Nome	**CTFN11 -** Não excluir produto
Descrição	Não deve excluir produto (Token ausente, inválido ou expirado)
Requisitos	ID do produto
Etapas do teste	rota /produtos com o método DELETE

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

Status esperado	401

Nome	**CTFN12 -** Não excluir produto
Descrição	Não deve excluir produto (Rota exclusiva para administradores)
Requisitos	ID do produto
Etapas do teste	rota /produtos com o método DELETE

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

Status esperado	403

Nome	**CTFN13 -** Não editar produto
Descrição	Não deve editar produto (Já existe produto com esse nome)
Requisitos	ID do produto
Etapas do teste	rota /produtos com o método PUT

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

Status esperado	400

Nome	**CTFN14 -** Não editar produto
Descrição	Não deve editar produto (Token ausente, inválido ou expirado)
Requisitos	ID do produto
Etapas do teste	rota /produtos com o método PUT

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

Status esperado	401

Nome	**CTFN15 -** Não editar produto
Descrição	Não deve editar produto (Rota exclusiva para administradores)
Requisitos	ID do produto
Etapas do teste	rota /produtos com o método PUT

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

Status esperado	403

Nome	**CTFN16 -** Não cadastrar carrinho
Descrição	Não deve cadastrar carrinho (Algo deu errado)
Etapas do teste	rota /carrinhos com o método POST

Asserção	{
	"message": "Não é permitido possuir produto duplicado | Não é permitido ter mais de 1 carrinho | Produto não encontrado | Produto não possui quantidade suficiente"
	}

Status esperado	400

Nome	**CTFN17 -** Não cadastrar carrinho
Descrição	Não deve cadastrar carrinho (Token ausente, inválido ou expirado)
Etapas do teste	rota /carrinhos com o método POST

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

Status esperado	401

Nome	**CTFN18 -** Não buscar carrinho
Descrição	Não deve buscar o carrinho pelo seu ID

Requisitos	ID do carrinho
Etapas do teste	rota /carrinhos com o método GET

Asserção	{
	"message": "Carrinho não encontrado"
	}

Status esperado	400

Nome	**CTFN19 -** Não concluir compra
Descrição	Não deve concluir a compra e excluir o carrinho (Token ausente, inválido ou expirado)
Etapas do teste	rota /carrinhos com o método DELETE

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

Status esperado	401

Nome	**CTFN20 -** Não cancelar compra
Descrição	Não deve cancelar a compra e excluir carrinho e retornar os produtos para o estoque (Token ausente, inválido ou expirado)
Etapas do teste	rota /carrinhos com o método DELETE

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

Status esperado	401

Nome	**CTFN21 -** Falhar ao buscar produto não existente
Descrição	Deve falhar ao tentar fazer a busca de um produto que não esteja cadastrado
Etapas do teste	rota /produtos/{_id} com o método GET
Status esperado	400

Nome	**CTFN22 -** Falhar ao buscar carrinho não existente
Descrição	Deve falhar ao tentar fazer a busca de um carrinho que não esteja cadastrado
Etapas do teste	rota /carrinhos/{_id} com o método GET
Status esperado	400

Nome	**CTFN23 -** Falhar ao postar carrinho
Descrição	Deve falhar ao tentar fazer o cadastro de um carrinho com produto que não tenha a quantidade que é exigida no parâmetro

Asserção	{
	"produtos": [
	{
	"idProduto": " ",
	"quantidade": Quantidade a mais que o produto tenha
	},
	{
	"idProduto": " ",
	"quantidade": number
	}
	]
	}

Etapas do teste	rota /carrinhos com o método POST
Status	400

esperado	

Nome	**CTFN24 -** Falhar ao cancelar compra de carrinho
Descrição	Deve falhar ao tentar cancelar a compra de um carrinho (logado com um token inválido, ausente, expirado ou que o usuário não exista mais no sistema)
Etapas do teste	rota /carrinhos/cancelar-compra com o método DELETE
Status esperado	401

6. Estratégia de teste

A abordagem geral e os objetivos das tarefas do teste segue detalhadamente abaixo, e o fluxo da estratégia de teste é representado no Anexo 2.

Ferramentas utilizadas	• Swagger; • Visual Studio Code; • PostMan; • XMind; • Git; • GitHub.
Abordagem	Testes de Caso de Uso
Técnicas	Caixa preta.
Tipos de testes	• Funcionais; • Unidade.
Massa	• Usuário administrador • Usuário não administrador
Melhorias	• Documentação; • Respostas da API.

Notas	Bugs foram encontrados
Riscos	Mal funcionamento da aplicação

Disponível em anexo 2.

7. Priorização de teste

A priorização dos testes foi organizada pensando na maneira mais eficiente de realizar os testes. Sendo assim, foi organizada da seguinte maneira:

Prioridade	Tipo de teste
Prioridade 1	Testes positivos
Prioridade 2	Testes negativos
Prioridade 3	Testes de fluxo
Prioridade 4	Documentação de acordo com os resultados

8. Candidatos para automação

Os testes que serão automatizados são:

- Testes positivos
- Teste negativos (exceto alguns da rota /produtos)
- Testes de fluxo para administradores

Sendo assim, no total de 47 testes, 35 foram automatizados. Todos os testes positivos foram automatizados devido a sua importância, já para os testes negativos que não foram automatizados, foi estabelecido que não possuíam tanta importância quanto os demais.

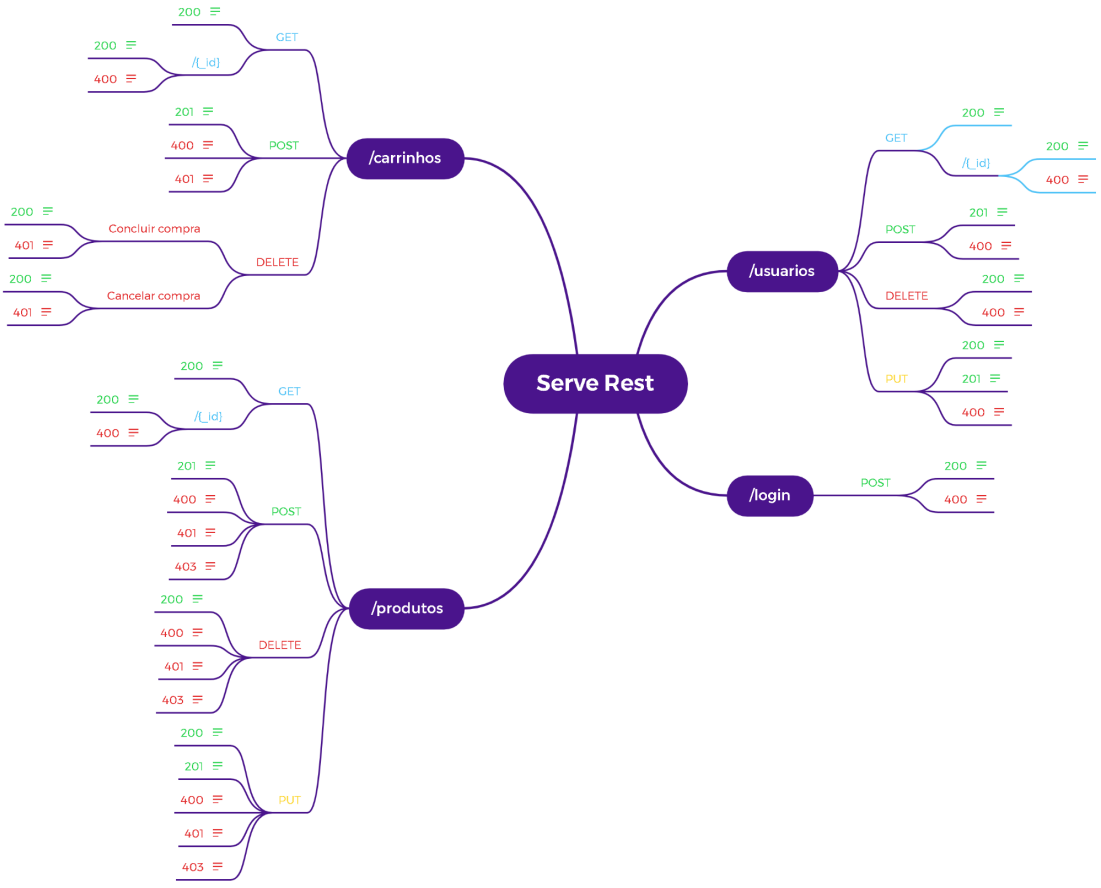
9. Ferramentas

Ferramenta	Versão	Função
Swagger	2.26.1	Documentação da API

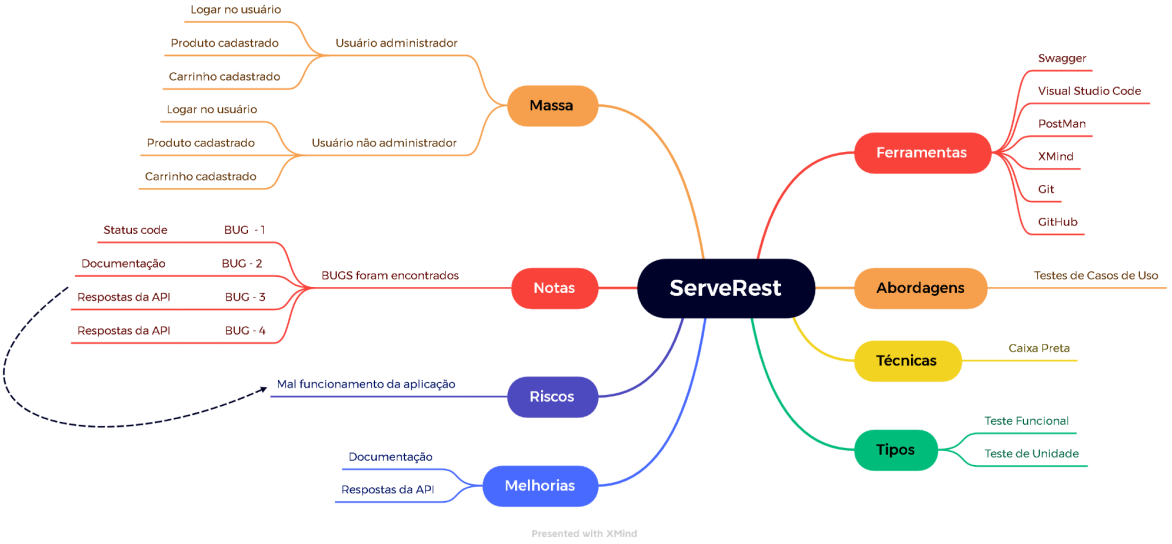
Visual Studio Code	1.69.0	Realização dos testes
XMind	12.0.2	Mapa mental
Git	?	Versionamento de código
GitHub	?	Postagem do projeto
PostMan	9.18.3	Realização dos testes não automatizados

10. Anexos

Anexo 1:



Anexo 2:



Drive com as imagens para um melhor entendimento:

https://drive.google.com/drive/u/3/folders/1qw7bXEJy6SeXd1uQigacDgjKM3wU3j0c