LogicalForest 2
Projeto Final

Felipe Ripplinger Dupont

Plano de testes

Objetivos:

Selecionar, priorizar e planejar os testes das funcionalidades da API ServeRest.

Escopo:

Todos os endpoints presentes.

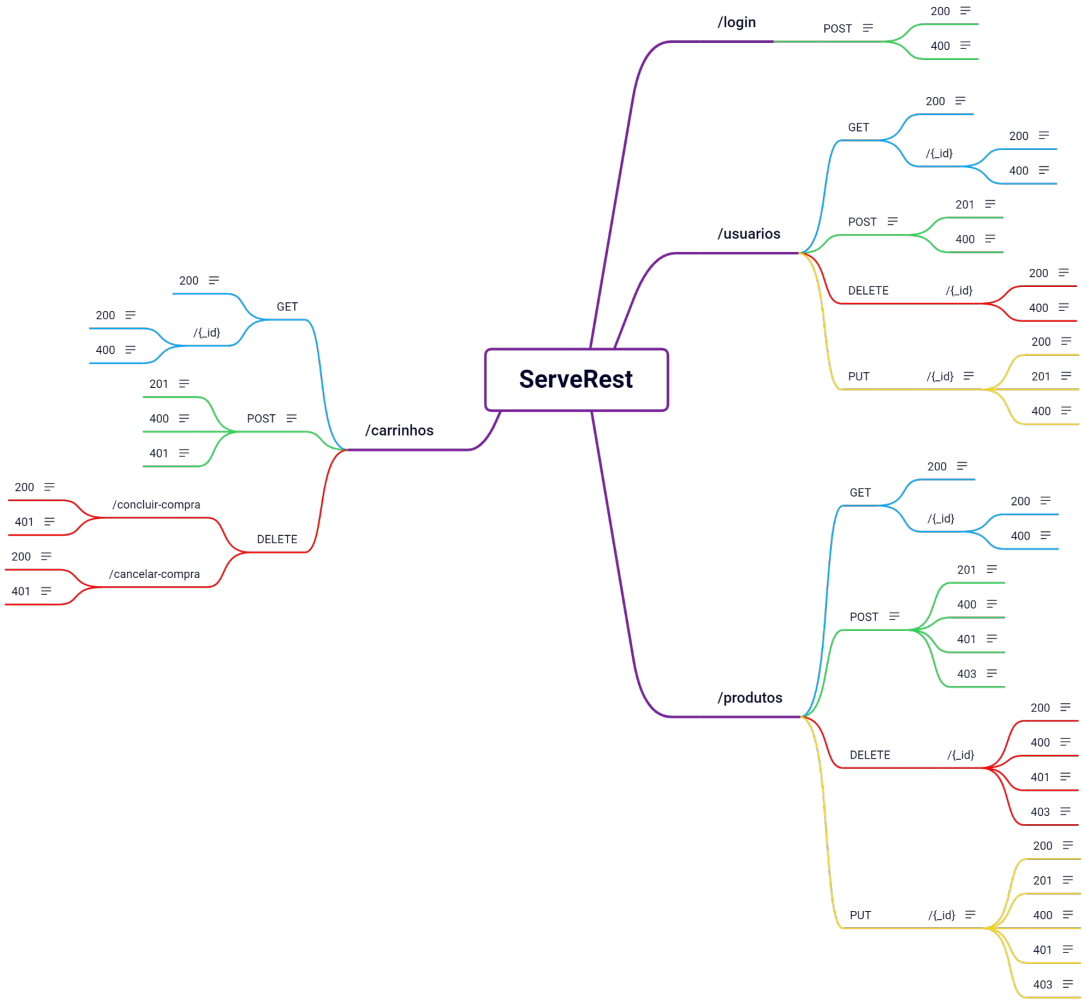
Plano de testes

Prioridade:

Todas as rotas serão testadas conforme sua funcionalidade. Testes de carga e destrutivos não serão aplicados, focando apenas em testes funcionais.

Estratégia:

A estratégia escolhida para os testes foi a de caixa preta. Os tipos de testes que serão realizados se dividem em dois: regressivos e de fluxo.



Informações gerais da API

CODE	TEXTO	DESCRIÇÃO
200	OK	Sucesso.
201	Created	Cadastrado com sucesso.
400	Bad Request	Inválido ou não pode ser realizado.
401	Unauthorized	Token ausente, inválido ou expirado.
403	Forbidden	Rota exclusiva para administradores (administrador = true).
405	Method Not Allowed	Não é possível realizar GET em /{}. Acesse https://serverest.dev para ver as rotas disponíveis e como utilizá-las.

Métodos de autenticação presente

Bearer Token

Formato de retorno e envio padrão

JSON

Suítes de casos de teste

CTP 01-19

CTN 01-21

CFT 01&02

GET
POST
DELETE
PUT

Candidatos para automação:

Todos os testes presentes na suíte serão automatizados e irão gerar um relatório, também automatizado.

CTP05 - Buscar usuário por ID

Descrição:
Deve buscar um usuário com sucesso.

Etapas do Teste:
1.Obter o ID de um usuário cadastrado.
2.Realizar requisição na rota /usuarios/{_id} com o método GET.

Padrão da resposta esperada:
200 - Usuário encontrado


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

CTP11 - Cadastrar produto

Descrição:
Deve cadastrar um novo produto com sucesso.

Etapas do Teste:
1.Logar com sucesso(Usuário administrador).
2.Obter um nome não cadastrado, preço, descrição e quantidade, colocar isso no body da requisição.
3.Realizar requisição na rota /produtos com o método POST.

Método de Autenticação:
Bearer Token.

Padrão do Body:

```
{
  "nome": "Logitech MX Vertical",
  "preco": 470,
  "descricao": "Mouse",
  "quantidade": 381
}
```

Padrão da resposta esperada:
201 - Cadastro com sucesso

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

CTN10 - Excluir produto sem sucesso (Em carrinho)

Descrição:
Não deve excluir um produto caso o mesmo esteja em um carrinho.

Etapas do Teste:
1.Logar com sucesso(Usuário administrador).
2.Obter o ID de um produto cadastrado que esteja em um carrinho.
3.Realizar requisição na rota /produto /{_id} com o método DELETE.

Método de Autenticação:
Bearer Token.

Padrão da resposta esperada:
400- Bad Request


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

CFT02 - Fluxo de usuário administrador

Descrição:
Simular as atividades que um novo usuário administrador pode realizar.

Etapas do Fluxo:
1.Cadastrar como administrador.
2.Logar como administrador.
3.Olhar os produtos cadastrados.
4.Cadastrar um novo produto.
5.Editar um produto cadastrado.
6.Excluir um produto cadastrado.

Status codes:
1. 201
2. 200
3. 200
4. 201
5. 200
6. 200

Ferramentas

Ferramenta	**Função**
Google Docs	Documentação do projeto.
Google Slides	Apresentação em slides do projeto.
Postman	Realização de um mapeamento básico das rotas e para conferir defeitos manualmente.
XMind	Criação do Mapa Mental das rotas e seus respectivos status code.
VSCode(JS,Cypress & Mocha)	Desenvolvimento do código dos testes e report automatizado.
Git	Versionamento do código.
GitHub	Hospedagem de código e arquivos.

Relatório de execução

Análise de resultados:

Todos os 42 testes da suíte de testes foram automatizados com sucesso. Me comprometi durante o desenvolvimento dos testes, que a execução dos testes regressivos não afetasse o ambiente excessivamente.

Cobertura de testes:

Path coverage de 100%, tratando todas as funcionalidades e suas possíveis falhas documentadas.

Relatório de execução

Erros encontrados:

BUG-01
Local afetado: Rotas que exijam uma quantidade do body da requisição.
Descrição: Caso a quantidade a ser enviada seja um número muito grande (maior que 9007199254740991) ou não válido, a api considera como um número não seguro e não envia ele na requisição, então o erro retornado é " não contém 1 valor obrigatório".
Sugestão de correção: Criar uma mensagem de erro específica para a situação, como “O valor da quantidade não é um número válido ”

BUG-02
Local afetado: [POST /login].
Descrição: O erro de email e/ou senha inválidos, na documentação está como se retornasse um código 400, quando na verdade ele retorna um código 401.
Sugestão de correção: Ajustar o código de erro na documentação ou na API para que fiquem idênticos.

BUG-03
Local afetado: [GET /carrinhos] e [GET /carrinhos/{_id}].
Descrição: Os endpoints afetados estão com o body resposta incorreta na documentação, possuindo um par de colchetes extra dentro da chave produtos.
Sugestão de correção: Ajustar o body da resposta na documentação ou na API para que fiquem idênticos.

BUG-04

Local afetado:
[GET /produtos].

Descrição:
O endpoint afetado está com o body resposta incorreta na documentação, possuindo a chave usuários ao invés de produtos.

Sugestão de correção:
Ajustar o body da resposta na documentação para que fique idêntico ao da API.

BUG-05

Local afetado:
Todas as rotas que exijam um body com algum valor.

Descrição:
Ao enviar um body com valores sem aspas, o código de erro retornado é um 500 que não está documentado, o erro também acontece se o body estiver de alguma outra maneira errado. Esse problema não pode ser visto através do site da API apenas por requisições de terceiros.

Sugestão de correção:
Desenvolver mais este código de erro para que ele se adapte melhor a situação e registrar ele na documentação.

Relatório de execução

Sugestões:

- Adicionar os códigos de erro 5xx na documentação.

- Um método PUT para a rota carrinhos, uma função básica, entretanto essencial para qualquer sistema que use carrinhos.

- A implementação do código 404 seria uma boa ideia para rotas não existentes, atualmente elas retornam o erro 405, que segundo o RFC 7231, indica que o método recebido na requisição é conhecido pelo servidor de origem, mas não é suportado pelo recurso de destino, algo que não se encaixa muito bem.