DBC, Módulo de API (Postman, Script)

QA, Rafael Linhares e Vitor Poeta

---

1. Documentação de Cenários de Teste (Gherkin)

Funcionalidade: Autenticação de Usuário (Login)

Endpoint: POST /login

- **Cenário 01: Login com credenciais válidas (Positivo)**
  - **Dado** que possuo um usuário cadastrado com email "rafael.postman@qa.com.br" e senha "1234".
  - **Quando** envio uma requisição POST para "/login" com essas credenciais.
  - **Então** o status da resposta deve ser **200 OK**.
  - **E** o corpo da resposta deve conter um token de autorização "Bearer".

- **Cenário 02: Login com senha incorreta (Negativo)**
  - **Dado** que possuo um usuário cadastrado com email "rafael.postman@qa.com.br".
  - **Quando** envio uma requisição POST para "/login" com a senha incorreta "senhaerrada".
  - **Então** o status da resposta deve ser **401 Unauthorized**.
  - **E** a mensagem de erro deve ser "Email e/ou senha inválidos".

- **Cenário 03: Login com formato de email inválido (Negativo)**
  - **Dado** que tento logar com um email sem formatação correta "https://www.google.com/search?q=emailsemarroba.com".
  - **Quando** envio uma requisição POST para "/login".
  - **Então** o status da resposta deve ser **400 Bad Request** (ou o erro de validação correspondente da API).

Funcionalidade: Cadastro de Usuários

Endpoint: POST /usuarios

- **Cenário 04: Cadastrar novo usuário administrador (Positivo)**
    - **Dado** que tenho dados de um novo usuário válido.
    - **Quando** envio uma requisição POST para "/usuarios" com "administrador": "true".
    - **Então** o status da resposta deve ser **201 Created**.
    - **E** a mensagem deve ser "Cadastro realizado com sucesso".

- **Cenário 05: Cadastro com email duplicado (Negativo)**
    - **Dado** que já existe um usuário cadastrado com o email "rafael.postman@qa.com.br".
    - **Quando** tento cadastrar outro usuário com o mesmo email.
    - **Então** o status da resposta deve ser **400 Bad Request**.
    - **E** a mensagem deve ser "Este email já está sendo usado".

- **Cenário 06: Cadastro com campo obrigatório vazio (Negativo)**
    - **Dado** que tento cadastrar um usuário sem preencher a "password".
    - **Quando** envio a requisição POST para "/usuarios".
    - **Então** o status da resposta deve ser **400 Bad Request**.
    - **E** a API deve informar que o campo é obrigatório..

Funcionalidade: Cadastro de Produtos

Endpoint: POST /produtos

Pré-requisito: Usuário deve estar logado e ser administrador

- **Cenário 07: Cadastrar produto com sucesso (Positivo)**
    - **Dado** que estou autenticado com um token de usuário administrador.
    - **Quando** envio um POST para "/produtos" com nome "Teclado Postman", preco 150, descricao "Teclado Gamer" e quantidade 10.
    - **Então** o status da resposta deve ser **201 Created**.
    - **E** a mensagem deve ser "Cadastro realizado com sucesso".

- **Cenário 08: Cadastrar produto com nome duplicado (Negativo)**
    - **Dado** que estou autenticado e já existe um produto "Teclado Postman".
    - **Quando** tento cadastrar novamente um produto com nome "Teclado Postman".
    - **Então** o status da resposta deve ser **400 Bad Request**.
    - **E** a mensagem deve ser "Já existe produto com esse nome"

.

- **Cenário 09: Cadastrar produto sem Token de acesso (Negativo)**
    - **Dado** que não realizei login (sem token no Header).
    - **Quando** tento enviar um POST para "/produtos".
    - **Então** o status da resposta deve ser **401 Unauthorized**.
    - **E** a mensagem deve ser "Token de acesso ausente, inválido, expirado...".

Funcionalidade: Editar Produtos

Endpoint: PUT /produtos/ {_id}

Pré-requisito: Usuário deve estar logado e ser administrador

- **Cenário 10: Editar produto com sucesso (Positivo)**
    - **Dado** que estou autenticado com um token de usuário administrador.
    - **E** existe um produto cadastrado chamado “Logitech MX Vertical”.
    - **E** coloco o id do Logitech MX Vertical no campo “id do produto”.
    - **Quando** envio um PUT para "/produtos/{_id}", nome Logitech MX Vertical , preco 999, descricao "Mouse" e quantidade 381.
    - **Então** o status da resposta deve ser **200 OK.**
    - **E** a mensagem deve ser "Registro alterado com sucesso".

- **Cenário 11: Editar produto sem permissão de administrador (Negativo)**
    - **Dado** que estou autenticado com um token de usuário normal.
    - **E** existe um produto cadastrado chamado “Logitech MX Vertical”.
    - **E** coloco o id do Logitech MX Vertical no campo “id do produto”.
    - **Quando** envio um PUT para "/produtos/{_id}", nome Logitech MX Vertical , preco 1000, descricao "Mouse" e quantidade 199.
    - **Então** o status da resposta deve ser **403 Forbidden.**
    - **E** a mensagem deve ser "Rota exclusiva para administradores".

.

- **Cenário 12: Editar quantidade do produto para número negativo (Negativo)**
    - **Dado** que estou autenticado com um token de usuário administrador.
    - **E** existe um produto cadastrado chamado “Logitech MX Vertical”.
    - **E** coloco o id do Logitech MX Vertical no campo “id do produto”.
    - **Quando** envio um PUT para "/produtos/{_id}", nome Logitech MX Vertical , preco 999, descricao "Mouse" e quantidade -1.
    - **Então** o status da resposta deve ser **400 Bad Request.**
    - **E** a mensagem deve ser "Quantidade deve ser maior ou igual a 0".

Funcionalidade: Excluir Produtos

Endpoint: DELETE /produtos/ {_id}

Pré-requisito: Usuário deve estar logado e ser administrador

- **Cenário 13: Excluir produto com sucesso (Positivo)**
  - **Dado** que estou autenticado com um token de usuário administrador.
  - **E** existe um produto cadastrado chamado “Logitech MX Horizontal”.
  - **E** coloco o id do Logitech MX Horizontal no campo “id do produto”.
  - **Quando** envio um DELETE para "/produtos/{_id}"
  - **Então** o status da resposta deve ser **200 OK.**
  - **E** a mensagem deve ser "Registro excluído com sucesso".

- **Cenário 14: Excluir produto que não existe (Negativo)**
  - **Dado** que estou autenticado com um token de usuário normal.
  - **E** coloco o id de um produto inexistente no campo “id do produto”.
  - **Quando** envio um DELETE para "/produtos/{_id}"
  - **Então** o status da resposta deve ser **200 OK.**
  - **E** a mensagem deve ser "Nenhum registro excluído".

.

- **Cenário 15: Excluir produto sem Token de Acesso (Negativo)**
  - **Dado** que estou autenticado com um token de administrador inválido.
  - **E** existe um produto cadastrado chamado “Logitech MX Horizontal”.
  - **E** coloco o id do Logitech MX Horizontal no campo “id do produto”.
  - **Quando** envio um DELETE para "/produtos/{_id}"
  - **Então** o status da resposta deve ser **401 Unauthorized.**
  - **E** a mensagem deve ser "Token de acesso ausente, inválido, expirado...".

Funcionalidade: Excluir Usuários

Endpoint: DELETE /usuarios/ {_id}

- **Cenário 16: Excluir usuário com sucesso (Positivo)**
  - **Dado** que estou autenticado com um token de usuário.
  - **E** possuo o id de um usuário cadastrado
  - **Quando** envio um DELETE para "/usuarios/{_id}"

    - **Então** o status da resposta deve ser **200 OK.**
    - **E** a mensagem deve ser "Registro excluído com sucesso".

- **Cenário 17: Excluir usuário que não existe (Negativo)**
    - **Dado** que estou autenticado com um token de usuário normal.
    - **E** coloco no campo de id, um id inexistente
    - **Quando** envio um DELETE para "/usuarios/{_id}"
    - **Então** o status da resposta deve ser **200 OK.**
    - **E** a mensagem deve ser "Nenhum registro excluído".

.

- **Cenário 18: Excluir usuário sem Token de Acesso (Negativo)**
    - **Dado** que estou autenticado com um token de usuário inválido.
    - **Quando** envio um DELETE para "/usuarios/{_id}"
    - **Então** o status da resposta deve ser **401 Unauthorized.**
    - **E** a mensagem deve ser "Token de acesso ausente, inválido, expirado...".

Funcionalidade: Editar Usuários

Endpoint: PUT /usuarios/ {_id}

- **Cenário 19: Editar usuário com sucesso (Positivo)**
    - **Dado** que estou autenticado com um token de usuário.
    - **E** possuo o id de um usuário cadastrado
    - **Quando** envio um PUT para "/usuarios/{_id}" nome Vitor QATeste , email vitorqa@gmail.com, password “teste” e administrador false.
    - **Então** o status da resposta deve ser **200 OK.**
    - **E** a mensagem deve ser "Registro alterado com sucesso".

- **Cenário 20: Editar usuário com formato de e-mail inválido (Negativo)**
    - **Dado** que estou autenticado com um token de usuário.
    - **E** possuo o id de um usuário cadastrado
    - **Quando** envio um PUT para "/usuarios/{_id}" nome Vitor QATeste , email vitorqagmail.com, password “teste” e administrador false.
    - **Então** o status da resposta deve ser **400 Bad request.**
    - **E** a mensagem deve ser "Email deve ser um email válido".

.

- **Cenário 21: Editar usuário com um e-mail já cadastrado (Negativo)**
    - **Dado** que estou autenticado com um token de usuário.

- **E** possuo o id de um usuário cadastrado
- **E** que já existe outro email vitorqa@gmail.com cadastrado
- **Quando** envio um PUT para "/usuarios/{_id}" nome Vitor QATeste , email vitorqa@gmail.com, password “teste” e administrador false.
- **Então** o status da resposta deve ser **400 Bad request.**
- **E** a mensagem deve ser "Este email já está sendo usado".

Funcionalidade: Listagem e Busca de Usuários

Endpoint: `GET /usuarios`

- **Cenário 22: Listar todos os usuários (Positivo)**
  - **Dado** que existem usuários cadastrados na base.
  - **Quando** envio uma requisição GET para "/usuarios" sem parâmetros.
  - **Então** o status da resposta deve ser **200 OK**.
  - **E** o corpo da resposta deve conter a chave "quantidade" maior que 0.
  - **E** deve retornar uma lista na chave "usuarios".

- **Cenário 23: Buscar usuário por Nome inexistente (Negativo/Vazio)**
  - **Dado** que não existe nenhum usuário com o nome "NomeQueNaoExiste123".
  - **Quando** envio um GET para "/usuarios" com o parâmetro query `nome="NomeQueNaoExiste123"`.
  - **Então** o status da resposta deve ser **200 OK**.
  - **Mas** a "quantidade" deve ser **0**.
  - **E** a lista de "usuarios" deve estar vazia.

- **Cenário 24: Buscar usuário por ID inexistente na Listagem (Negativo/Vazio)**
  - **Dado** que utilizo um ID que não existe "id_falso_123".
  - **Quando** envio um GET para "/usuarios" com o parâmetro query `_id="id_falso_123"`.
  - **Então** o status da resposta deve ser **200 OK**.
  - **Mas** a "quantidade" deve ser **0**.

Funcionalidade: Buscar Usuário por ID

Endpoint: `GET /usuarios/{_id}`

- **Cenário 25: Buscar usuário por ID existente (Positivo)**

    - **Dado** que possuo o `_id` de um usuário previamente cadastrado.
    - **Quando** envio uma requisição GET para "/usuarios/{_id}".
    - **Então** o status da resposta deve ser **200 OK**.
    - **E** o corpo da resposta deve conter os dados do usuário (nome, email, password, etc).

- **Cenário 26: Buscar usuário por ID inexistente (Negativo)**
    - **Dado** que utilizo um `_id` que não existe no banco de dados (ex: "0000000000000000").
    - **Quando** envio uma requisição GET para "/usuarios/id_inexistente_123".
    - **Então** o status da resposta deve ser **400 Bad Request**.
    - **E** a mensagem deve ser "Usuário não encontrado".

- **Cenário 27: Buscar usuário após exclusão (Negativo)**
    - **Dado** que tinha um usuário válido, mas acabei de realizar a exclusão dele (DELETE).
    - **Quando** tento buscar novamente os dados desse usuário pelo `_id`.
    - **Então** o status da resposta deve ser **400 Bad Request**.
    - **E** a mensagem deve ser "Usuário não encontrado".

Funcionalidade: Listagem e Busca de Produtos

Endpoint: `GET /produtos`

- **Cenário 28: Listar todos os produtos (Positivo)**
    - **Dado** que existem produtos cadastrados na loja.
    - **Quando** envio uma requisição GET para "/produtos" sem parâmetros.
    - **Então** o status da resposta deve ser **200 OK**.
    - **E** o corpo da resposta deve conter a chave "quantidade" maior que 0.
    - **E** a lista "produtos" deve conter itens com nome, preço, descrição e quantidade.

- **Cenário 29: Buscar produto por Nome inexistente (Negativo/Vazio)**
    - **Dado** que não existe nenhum produto chamado "Nave Espacial X".
    - **Quando** envio uma requisição GET para "/produtos" filtrando por `nome="Nave Espacial X"`.
    - **Então** o status da resposta deve ser **200 OK**.
    - **Mas** a chave "quantidade" deve ser **0**.
    - **E** a lista "produtos" deve estar vazia.

- **Cenário 30: Buscar produto por Preço específico inexistente (Negativo/Vazio)**
    - **Dado** que não existe nenhum produto custando exatos R$ 99999.
    - **Quando** envio uma requisição GET para "/produtos" filtrando por `preco=0`.
    - **Então** o status da resposta deve ser **200 OK**.
    - **Mas** a chave "quantidade" deve ser **0**.

Funcionalidade: Buscar Produto por ID

Endpoint: `GET /produtos/{_id}`

- **Cenário 31: Buscar produto por ID existente (Positivo)**
    - **Dado** que possuo o `_id` de um produto cadastrado "Mouse Postman".
    - **Quando** envio uma requisição GET para "/produtos/{_id}" utilizando esse ID.
    - **Então** o status da resposta deve ser **200 OK**.
    - **E** o corpo da resposta deve trazer os detalhes corretos (nome, preço, descrição).

- **Cenário 32: Buscar produto por ID inexistente (Negativo)**
    - **Dado** que utilizo um ID que não existe na base (ex: "id_produto_falso").
    - **Quando** envio uma requisição GET para "/produtos/id_produto_falso".
    - **Então** o status da resposta deve ser **400 Bad Request**.
    - **E** a mensagem deve ser "Produto não encontrado".

- **Cenário 33: Buscar produto após exclusão (Negativo)**
    - **Dado** que eu tinha um produto válido, mas realizei a exclusão dele (DELETE).
    - **Quando** tento buscar novamente os dados desse produto pelo seu `_id` antigo.
    - **Então** o status da resposta deve ser **400 Bad Request**.
    - **E** a mensagem deve ser "Produto não encontrado".

Funcionalidade: Cadastro de Carrinho de Compras

Endpoint: `POST /carrinhos`

- **Cenário 34: Cadastrar carrinho com sucesso (Positivo)**
    - **Dado** que estou logado com um usuário que ainda não possui carrinho.
    - **E** que existe um produto cadastrado com estoque disponível.
    - **Quando** envio uma requisição POST para "/carrinhos" com o ID desse produto e quantidade 1.
    - **Então** o status da resposta deve ser **201 Created**.

    - **E** a mensagem deve ser "Cadastro realizado com sucesso".

- **Cenário 35: Tentar cadastrar segundo carrinho para mesmo usuário (Negativo)**
    - **Dado** que eu já realizei o cadastro de um carrinho anteriormente.
    - **Quando** tento enviar uma nova requisição POST para "/carrinhos".
    - **Então** o status da resposta deve ser **400 Bad Request**.
    - **E** a mensagem deve ser "Não é permitido ter mais de 1 carrinho".

- **Cenário 37: Cadastrar carrinho com produto inexistente (Negativo)**
    - **Dado** que estou logado.
    - **Quando** envio uma requisição POST para "/carrinhos" com um `idProduto` que não existe.
    - **Então** o status da resposta deve ser **400 Bad Request**.
    - **E** a mensagem deve ser "Produto não encontrado".

Funcionalidade: Cancelar Compra (Limpar Carrinho)

Endpoint: `DELETE /carrinhos/cancelar-compra`

- **Cenário 36: Cancelar carrinho existente com sucesso (Positivo)**
    - **Dado** que possuo um carrinho de compras cadastrado.
    - **Quando** envio uma requisição DELETE para "/carrinhos/cancelar-compra".
    - **Então** o status da resposta deve ser **200 OK**.
    - **E** a mensagem deve confirmar que o registro foi excluído e o estoque reabastecido.

- **Cenário 38: Tentar cancelar sem possuir carrinho (Positivo/Idempotente)**
    - **Dado** que não possuo nenhum carrinho de compras ativo.
    - **Quando** envio uma requisição DELETE para "/carrinhos/cancelar-compra".
    - **Então** o status da resposta deve ser **200 OK**.
    - **E** a mensagem deve ser "Não foi encontrado carrinho para esse usuário".

- **Cenário 39: Tentar cancelar sem Token de acesso (Negativo)**
    - **Dado** que não realizei login (sem token no Header).
    - **Quando** envio uma requisição DELETE para "/carrinhos/cancelar-compra".
    - **Então** o status da resposta deve ser **401 Unauthorized**.
    - **E** a mensagem deve ser "Token de acesso ausente, inválido, expirado...".

Funcionalidade: Concluir Compra

Endpoint: `DELETE /carrinhos/concluir-compra`

- **Cenário 40: Concluir compra com sucesso (Positivo)**
  - **Dado** que possuo um carrinho de compras cadastrado.
  - **E** estou logado com um token de acesso válido (admin/usuário).
  - **Quando** envio uma requisição DELETE para "/carrinhos/concluir-compra".
  - **Então** o status da resposta deve ser **200 OK**.
  - **E** a mensagem deve ser “Registro excluído com sucesso”.

- **Cenário 41: Tentar concluir compra sem possuir carrinho (Positivo/Idempotente)**
  - **Dado** que não possuo nenhum carrinho de compras ativo.
  - **E** estou logado com um token de acesso válido (admin/usuário).
  - **Quando** envio uma requisição DELETE para "/carrinhos/concluir-compra".
  - **Então** o status da resposta deve ser **200 OK**.
  - **E** a mensagem deve ser "Não foi encontrado carrinho para esse usuário".

- **Cenário 42: Tentar concluir compra sem Token de acesso (Negativo)**
  - **Dado** que não realizei login (sem token no Header).
  - **Quando** envio uma requisição DELETE para "/carrinhos/concluir-compra".
  - **Então** o status da resposta deve ser **401 Unauthorized**.
  - **E** a mensagem deve ser "Token de acesso ausente, inválido, expirado...".

Funcionalidade: Listagem de carrinhos

Endpoint: `GET /carrinhos`

- **Cenário 43: Listar todos os carrinhos (Positivo)**
  - **Dado** que existem carrinhos cadastrados no usuário logado.
  - **Quando** envio uma requisição GET para "/carrinhos" sem parâmetros.
  - **Então** o status da resposta deve ser **200 OK**.
  - **E** o corpo da resposta deve conter a chave "quantidade" maior que 0.
  - **E** a lista de carrinhos deve conter produtos com idProduto, quantidade e precoUnitario, ao fim do carrinho deve aparecer precoTotal, quantidadeTotal, idUsuario e _id.

- **Cenário 44: Buscar carrinho por id inexistente (Negativo/Vazio)**
    - **Dado** que não existe nenhum carrinho cadastrado com esse id
    - **Quando** envio uma requisição GET para "/carrinho" filtrando por `id="123"`.
    - **Então** o status da resposta deve ser **200 OK**.
    - **Mas** a chave "quantidade" deve ser **0**.
    - **E** a lista "carrinhos" deve estar vazia.

- **Cenário 45: Buscar produto por Preço Total inexistente (Negativo/Vazio)**
    - **Dado** que não existe nenhum carrinho com total custando exatos R$ 99999.
    - **Quando** envio uma requisição GET para "/carrinhos" filtrando por `precoTotal=99999`.
    - **Então** o status da resposta deve ser **200 OK**.
    - **Mas** a chave "quantidade" deve ser **0**.

Funcionalidade: Listagem de carrinhos por ID

Endpoint: `GET /carrinhos/{_id}`

- **Cenário 46: Buscar carrinho por ID existente (Positivo)**
    - **Dado** que possuo o `_id` de um carrinho cadastrado.
    - **Quando** envio uma requisição GET para "/carrinhos/{_id}" utilizando esse ID.
    - **Então** o status da resposta deve ser **200 OK**.
    - **E** o corpo da resposta deve trazer os detalhes corretos.

- **Cenário 47: Buscar produto por ID inexistente (Negativo)**
    - **Dado** que utilizo um ID que não existe na base (ex: _id = 1234567890123456).
    - **Quando** envio uma requisição GET para "/carrinhos/_id".
    - **Então** o status da resposta deve ser **400 Bad Request**.
    - **E** a mensagem deve ser "Carrinho não encontrado".

- **Cenário 48: Buscar produto após exclusão (Negativo)**
    - **Dado** que eu tinha um carrinho válido, mas realizei a exclusão dele (DELETE).
    - **Quando** tento buscar novamente os dados desse carrinho pelo seu `_id` antigo.
    - **Então** o status da resposta deve ser **400 Bad Request**.
    - **E** a mensagem deve ser "Carrinho não encontrado".

2. Relatório de Bugs

BUG-01: Exclusão de usuário permitida, sem autenticação (Falha de Segurança)

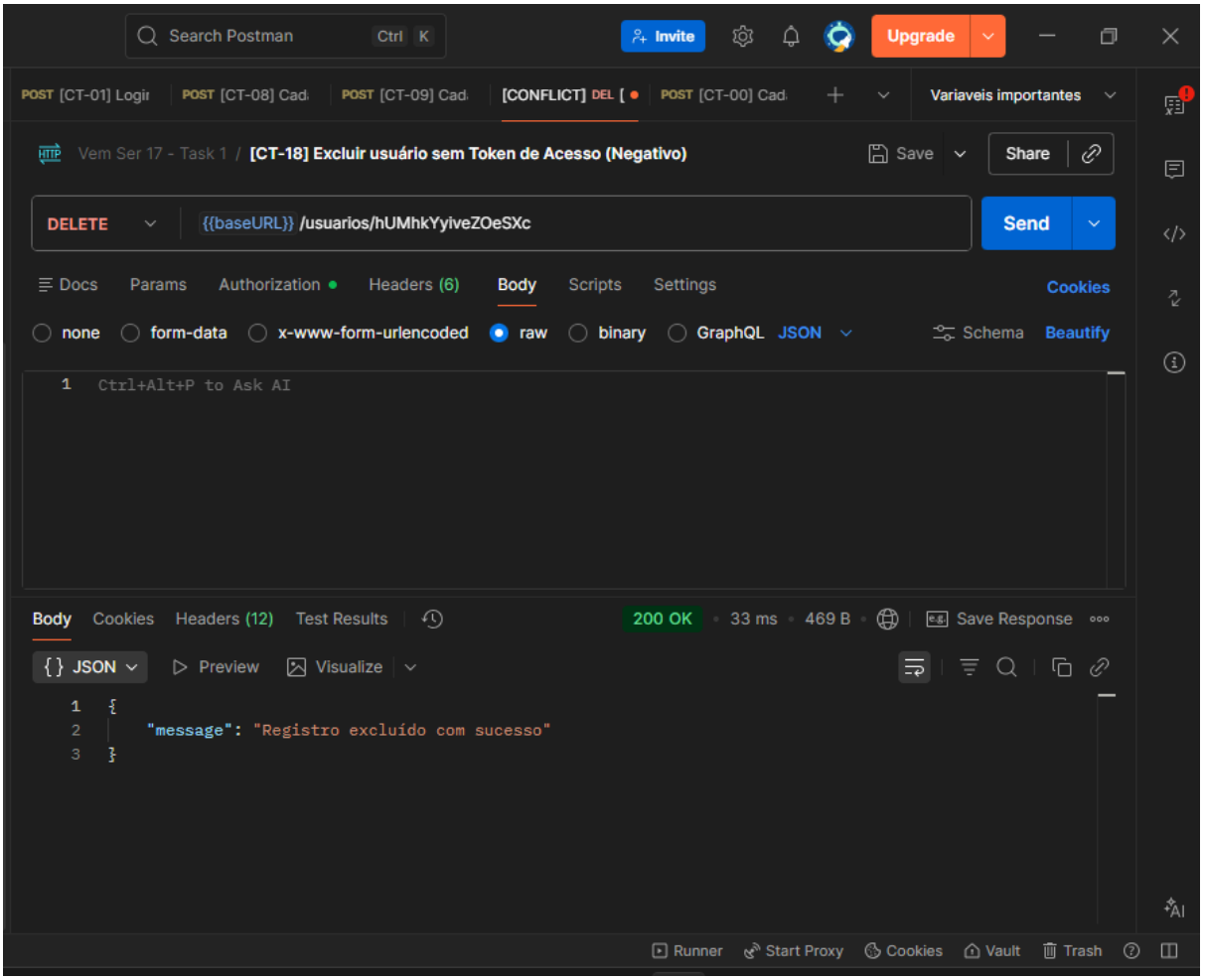
- **Severidade:** 🔴 Crítica (Blocker)
- **Endpoint:** `DELETE /usuarios/{_id}`
- **Descrição:** O endpoint de exclusão não está validando a presença ou validade do Token JWT. Isso permite que usuários não autenticados (anônimos) excluam os registros do banco.

- **Passo a Passo (Reprodução):**
  - Abrir o Postman.
  - Criar uma requisição `DELETE` para `http://localhost:3000/usuarios/[ID-DE-UM-USUARIO]`.
  - Na aba "Authorization", selecionar "No Auth" (garantindo que nenhum token está sendo enviado).
  - Clicar em "Send".

- **Resultado Esperado:**
  - Status HTTP: `401 Unauthorized`
  - Body: Mensagem de erro informando falta de token.

- **Resultado Obtido:**
  - Status HTTP: `200 OK`
  - Body: `Request Body: { "message": "Registro excluído com sucesso" }`

- **Evidência:**


Search Postman
Ctrl K
Invite
Upgrade
POST [CT-01] Logir
POST [CT-08] Cad
POST [CT-09] Cad
[CONFLICT] DEL [
POST [CT-00] Cad
Variaveis importantes
Vem Ser 17 - Task 1 / [CT-18] Excluir usuário sem Token de Acesso (Negativo)
Save
Share
DELETE
{{baseURL}} /usuarios/hUMhkYyiveZOeSXc
Send
Docs
Params
Authorization
Headers (6)
Body
Scripts
Settings
Cookies
none
form-data
x-www-form-urlencoded
raw
binary
GraphQL
JSON
Schema
Beautify
1 Ctrl+Alt+P to Ask AI
Body
Cookies
Headers (12)
Test Results
200 OK
33 ms
469 B
Save Response
{} JSON
Preview
Visualize
1 {
2 "message": "Registro excluído com sucesso"
3 }
Runner
Start Proxy
Cookies
Vault
Trash