

**Analise de testes**

**API REST Serverest**



Escrito por:

Leonardo Da Silva Oliveira

Sumário:

[Introdução: 3](#_Toc889753638)

[Sobre os testes: 3](#_Toc572551070)

[Analise por pastas: 3](#_Toc95596097)

[Pasta de teste de fluxo: 3](#_Toc1778723857)

[Pasta de teste regressivos: 4](#_Toc338788557)

[Análise dos resultados em comparação com as prioridades: 7](#_Toc759984243)

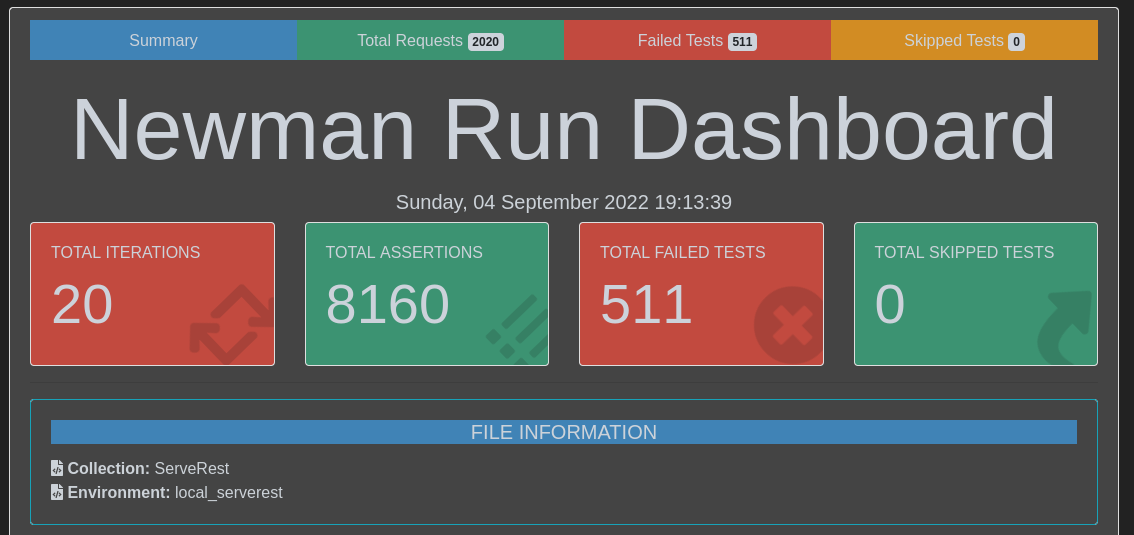
[Conclusão: 8](#_Toc556067396)

# Introdução:

Este documento tem como objetivo avaliar os testes realizados sobre a API Serverest, realizados no período de 22/08/2022 a 04/09/2022, por Leonardo da Silva Oliveira, nos ambientes de teste Postman e Newman. Para visualização dos testes neste documento serão utilizadas imagens dos testes feitos no newman, mas ambas as abordagens resultam nos mesmos resultados.

# Sobre os testes:

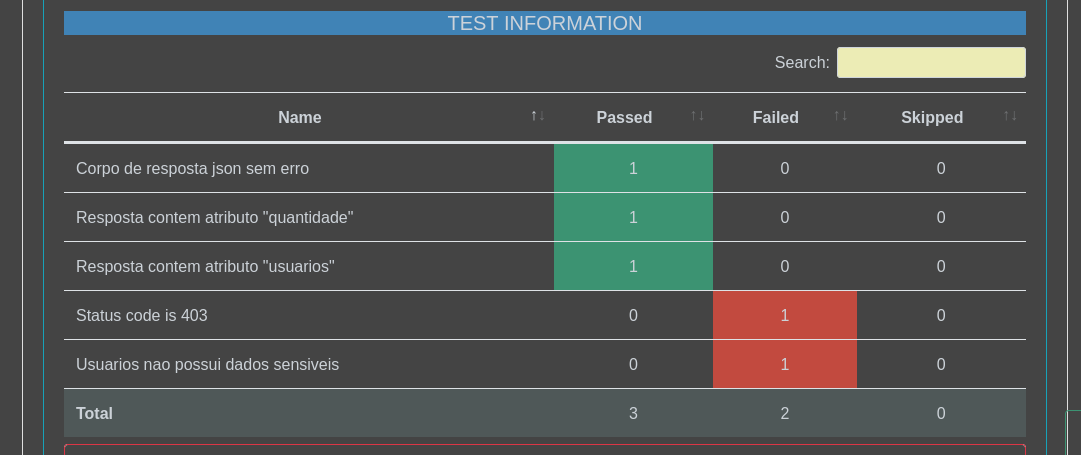
Neste documento estará sendo analisado um arquivo de testes onde rodaram 20 interações do script de testes totalizando 2020 chamadas de verbos, 8671 testes, sendo destes 8160 testes esperados e 511 testes onde a resposta não correspondia com as expectativas da equipe de testes.



# Analise por pastas:

## Pasta de teste de fluxo:

Os fluxos 1 e 2 de lojas e de 1 a 3 de usuários foram satisfatórios, quando chegamos ao fluxo de número 4 de usuários chegamos ao primeiro erro de teste das interações, que se repete em todas as interações, o erro ocorre no verbo FU04-07 na listagem de usuários, onde não se mostra necessário o token bearer, mas a partir desse verbo é possível visualizar dados sensíveis de todos os usuários como: “email”, “password”, “administrator”, e “\_id”. Além disso era esperado o status code 403 forbiden para tais dados, no entanto é retornado 200 OK.

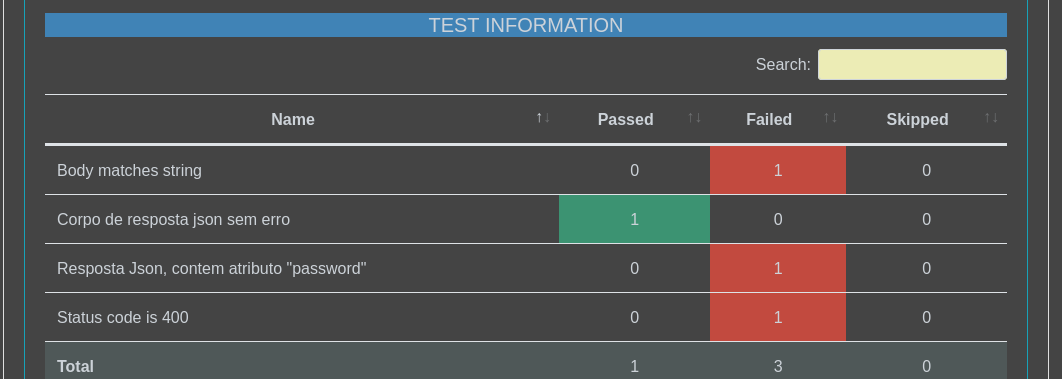


## Pasta de teste regressivos:

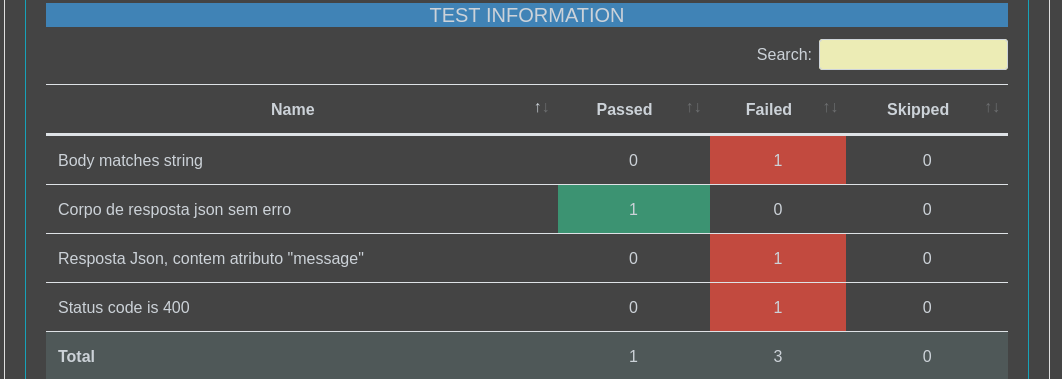
Nos testes regressivos nos deparamos com erros em 4 pastas distintas dos testes regressivos:

* Usuarios/cadastro auto/psswd-attack ID = RU02
* usuarios/ID-buscar-usuario ID = RU04
* Usuarios/ID-excluir-usuario ID = RU05
* Usuarios/ID-editar-usuario-auto ID = RU06

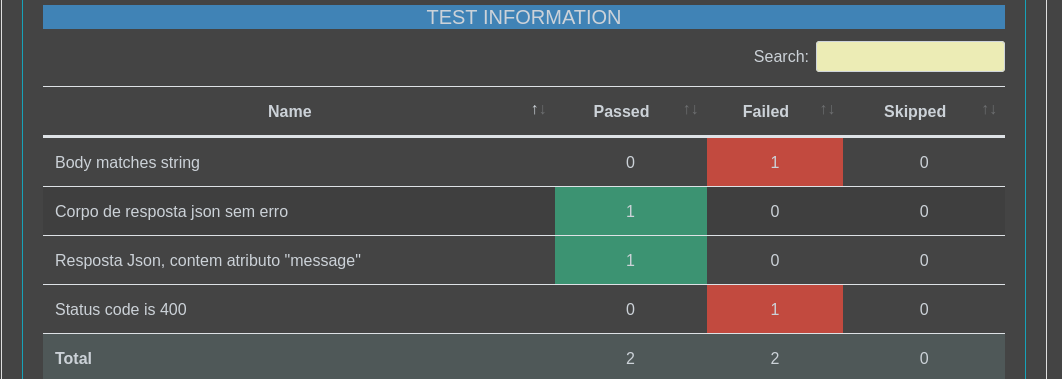
Na primeira pasta psswd-attack, verbo RU02-02, se tenta realizar um cadastro de um usuário com senha vazia, contendo apenas uma string com espaço “ “, se esperava que a API continuasse com o comportamento visto no verbo anterior RU02-01, mas ele considera o espaço uma string suficiente para cadastrar o usuário.

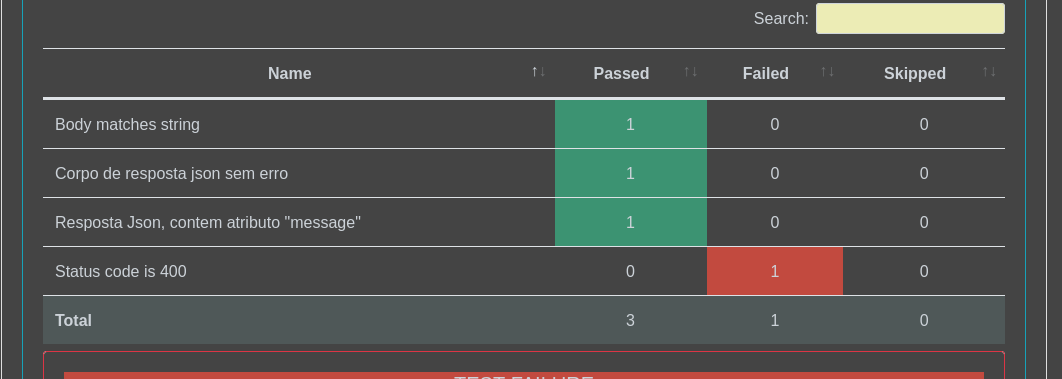


Na segunda pasta ID-buscar-usuario o primeiro erro ocorre no verbo RU04-01 onde com header ID presente, porém vazio se esperava um status 400 Bad Request ID não encontrado, no entanto ele retorna a lista com todos os usuários presentes na API.

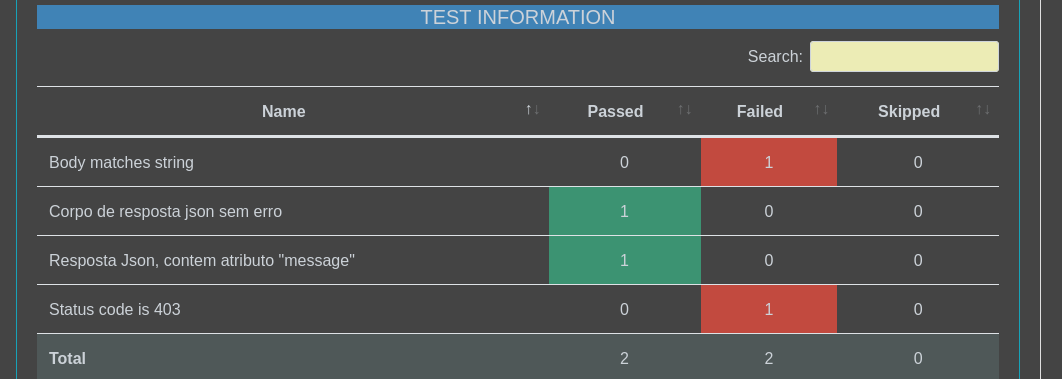
O segundo erro dessa mesma pasta o verbo RU04-03 é o mesmo encontrado em FU04-07, já explicado o motivo do erro.

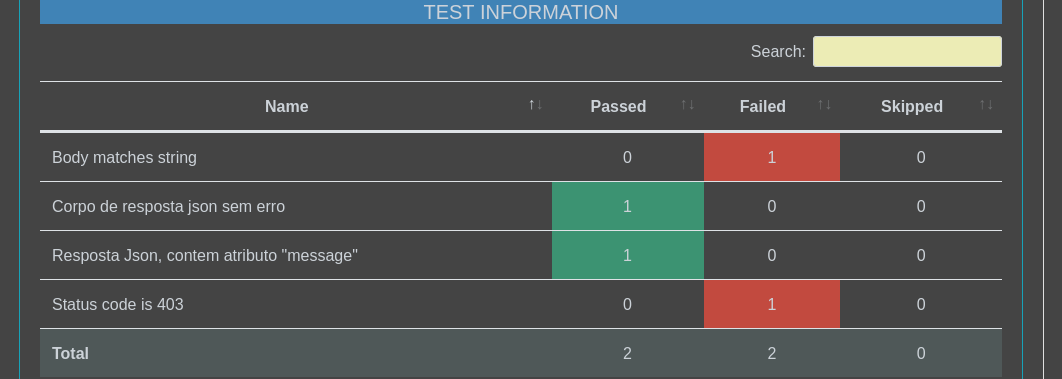
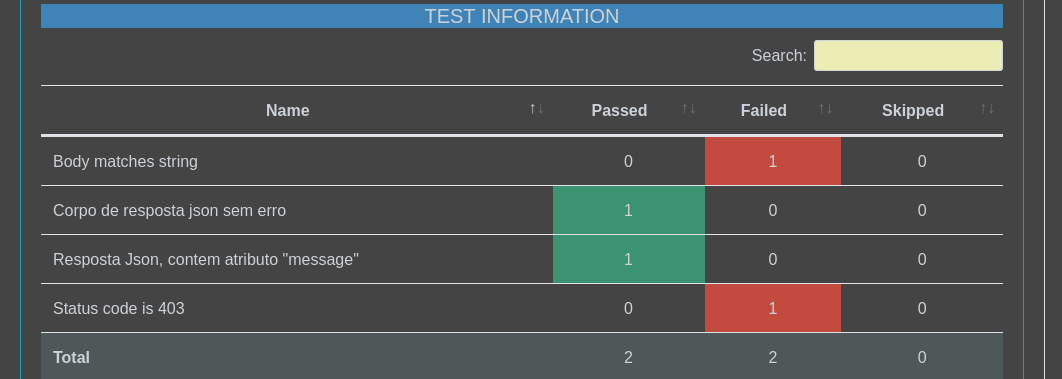
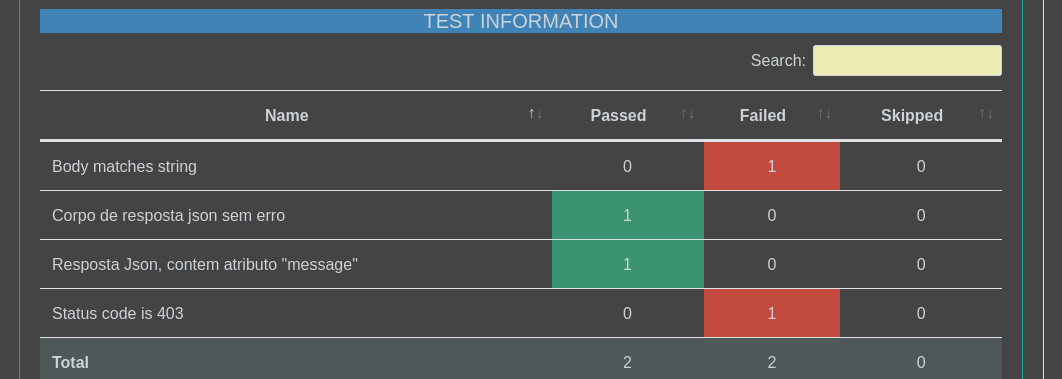
A terceira pasta ID-excluir-usuario possui dois verbos, ambos ocorrem casos de erro, no primeiro verbo RU05-01 é enviado o header ID vazio, onde ele deveria retornar 400 Bad Request por falta de ID no entanto ele retorna 405 com menssagem *“Não é possível realizar DELETE em /usuarios/. Acesse http://localhost:3000 para ver as rotas disponíveis e como utilizá-las.”* ou seja o verbo parece tentar realizar o delete em todos os usuários, o que parece ser uma falha grave.

O segundo erro da pasta o verbo RU05-02 com uma string qualquer no header, porém sem outro paramêtro header que é ”content-type application/json” removido se espera um erro 400 Bad Request já que o ID fornecido é inválido e não está no sistema. No entanto ele retorna com status 200 OK e menssagem correta ”Nenhum registro excluído”.

A última pasta dos testes regressivos com erro é a pasta ID-editar-usuario-auto, onde o primeiro erro ocorre no verbo RU06-03 igual aos erros RU04-03 e FU04-07, já detalhado acima.

O segundo erro da pasta no verbo RU06-04 ocorre quando logamos com uma conta recém criada, não administrativa, consegue utilizar o verbo para alterar a conta de outro usuário que não o dele. Se esperava outra resposta e outro status code como o 403 Forbiden no entanto a ação passa e a segunda conta é alterada.

O terceiro erro da pasta ocorre no verbo RU06-05 onde o usuário sem poder administrativo se concede tais poderes alterando sua conta. Novamente, se esperava status code 403 Forbiden e menssagem correspondente, porém o verbo executa a ação e status 200 OK.

Para os últimos erros da pasta no verbo RU06-07 se tenta alterar uma conta sem token de acesso no header, o erro ocorre pois o verbo executa mesmo assim, o mesmo ocorre no verbo RU06-08 onde se utiliza um token não valido. Nos dois casos se esperava status 403 Forbiden e menssagem correspondente, no entanto isso não ocorreu e as ações passaram.

# Análise dos resultados em comparação com as prioridades:

O primeiro ponto de priorização é alcançado pela API, onde ela se mostra capaz de executar a funcionalidade de loja e usuários.

Já no segundo ponto de priorização a API falha na capacidade delimitar poderes dos usuários e administradores onde que com o verbo PUT:usuarios é possível e facilmente modificar o nível de uma conta não administrativa para uma.

O terceiro ponto de priorização também se mostra falho pois com o verbo GET:usuarios?\_ID vazio é possível visualizar todos os usuários inclusive suas credenciais, emails, senhas e ids.

Os demais pontos atingem as expectativas da equipe de testes.

# Conclusão:

A ServeRest é uma excelente ferramenta para aprendizado de testes em API, mas não signifique que seja perfeita, existem falhas óbvias na estrutura de usuários quando se diz em respeito a proteção de dados e proteção de poderes administradores. A equipe de testes irá reportar as issues de acordo com o padrão pré-estabelecido.