Compass UOL

Relatório de Teste

SERVEREST

Relatório de planejamento de teste, apresentado ao Evangelista Matheus Domingos Locatelli, como parte das exigências do challenge.

Recife
2022

Sumário

Relatório de testes da API SERVEREST

1. INTRODUCÃO

1.1. Objetivo

Este Relatório de Avaliação do Teste descreve os resultados dos testes da API SERVEREST em termos de cobertura de teste e análise de possíveis falhas, bugs e tratamentos.

1.2. Escopo

Este Relatório de Avaliação do Teste é aplicado a API SERVEREST. Os testes conduzidos estão descritos no Plano de Teste previamente explanados e segue em anexo junto este. Este Relatório de Avaliação deve ser utilizado para o seguinte:

- avaliar a capacidade de aceitação e apropriação do(s) comportamento(s) de desempenho da API;
- avaliar a capacidade de aceitação dos testes;
- identificar aprimoramentos para aumentar a cobertura do teste e/ou sua qualidade.

2. Resumo de Resultados de Testes

Os casos de teste definidos no Conjunto de Teste para a API foram executados seguindo a estratégia de teste definida no Plano de Teste.

A Cobertura de teste (consulte a seção 4.0 do plano de teste) em termos de cobertura dos casos de uso e requisitos de teste definidos no Plano de Teste foi concluída.

A análise dos testes (conforme mostrado no postman) indica que há problemas **significativos de bugs e tratamentos não cobertos na API**. Os testes foram baseados inicialmente na documentação fornecida e moldados a necessidade da regra de negócios, com isso fora detectado em alguns pontos, como na rota login falhas de tratamento e bugs relacionado à aspectos não tratados. A Equipe de desenvolvimento

gerenciara os recursos para avaliar adicionalmente esses resultados de teste e determinar alternativas.

3. Cobertura do Teste

Os testes a serem executados na API estão definidos na seção 4 do Plano de Teste e no ServeRest postman collection (aplicado ao Postman), juntamente com seus critérios de conclusão. Os resultados da cobertura do teste são as seguintes:

Taxa de Casos de Teste Executados = 57/57 = 100%

Taxa de validações executadas = 369/378 = 97,61%

Taxa de Casos de Teste com Êxito = 57/57 = 100%

4. Análise de testes

Esta seção resume os resultados da análise gerada utilizando o Postman, conforme o documento ServeRest.postman_test_run anexado. A seção 5 recomenda ações para tratar as descobertas da análise.

Como resultado da aplicação dos testes conseguimos constatar algumas falhas, bugs e aspectos não tratados na API e/ou no Swagger, gerando um total de 2,39% em erros, conforme abaixo.

4.1. /login

- MA-01 - **Realizar login com email e senha vazia**: Modificar na API o tratamento para email e senha em branco, conforme documentação.
- MA-02 - **Realizar login com email em formato invalido**: Modificar na API o tratamento para email inválido, conforme documentação.
- BF-01 - **Realizar login com login e/ou senha errada**: Ao tentar realizar login no servidor com email e senha errada obtivemos o status code 401 e não o esperado conforme documentação Swagger, status code 400;

- MA-03 - **Realizar login com request diferente do padrão**: Modificar na API o tratamento para a requisição diferente do esperado, conforme documentação. Esperava-se o status code 400, mas retornou o 500.
- MA-04 - **Realizar login sem os campos obrigatórios**: Modificar na API o tratamento para requisições sem um body enviado, conforme swagger.

4.2. /usuários

- AS-01 - **Cadastrar usuário - sem email**: Falta acrescentar o tratamento ao Swagger no caso de ausência de email na requisição, ao status code 400;
- AS-02 - **Cadastrar usuário - email invalido**: Falta acrescentar o tratamento ao Swagger no caso de email invalido na requisição ao status code 400;
- AS-03 - **Cadastrar usuário - sem body**: Esperava-se “Os campos são obrigatórios.” como resposta.

4.3. /produtos

- AS-04 - **Cadastrar produto - Quantidade negativa**: Falta acrescentar o tratamento ao Swagger no caso de criar produto com quantidade negativa, ao status code 400.
- AS-05 - **Cadastrar produto - preço sendo menor que zero**: Falta acrescentar o tratamento ao Swagger no caso de criar produto com valor inferior a zero, no status code 400;

4.4. /carrinhos

- MAS-01 – **Cadastrar carrinho - quantidade de produtos negativa**: Falta acrescentar ao Swagger e modificar o tratamento na API para ser mais informativo;

5. Ações Sugeridas

As ações recomendadas são as seguintes:

5.1. /login

- MA-01 – Modificar na API o tratamento: Email e/ou senha inválidos, conforme swagger;
- MA-02 – Modificar na API o tratamento: Email e/ou senha inválidos, conforme swagger;
- BF-01 – Corrigir o status code de 401 para 400, conforme Swagger;
- MA-03 – Corrigir status code para 400 e acrescentar ao Swagger tratamento para valores fora do padrão esperado, como em falta de aspas: “Adicione aspas em todos

os valores.”

- MA-04 – Modificar na API o tratamento: Email e/ou senha inválidos, conforme swagger;

5.2. /usuário

- AS-01 – Adicionar o tratamento já presente na API ao Swagger: Email não pode ficar em branco;
- AS-02 – Adicionar o tratamento já presente na API ao Swagger: Email deve ser email válido;
- AS-03 - Modificar o tratamento presente na API para: “Os campos são obrigatórios.” e acrescentar ao Swagger no status code 400;

5.3. /produtos

- AS-04 – Acrescentar ao Swagger no status code 400 o tratamento: “Quantidade deve ser maior ou igual a 0.”;
- AS-05 – Acrescentar ao Swagger no status code 400 o tratamento: “Preço deve ser um número positivo.”.

5.4. /carrinhos

- MAS-01 – Acrescentar o tratamento ao status code 400 no Swagger: “Quantidade de produtos deve ser um número positivo.” E modificar o tratamento presente na API para: “Quantidade de produtos deve ser um número positivo.”.

Observações:

**MA – Modificar na API;*

***AS – Acrescentar ao Swagger;*

****BF – Bug funcional;*

*****MAS – Modificar na API e acrescentar ao Swagger.*