PLANEJAMENTO DE TESTES

API SERVEREST

COMPASS UOL

DOCUMENTO APRESENTADO AO EVANGELISTA MATHEUS DOMINGOS LOCATELLI, COMO PARTE DE UM PROJETO FINAL PARA COMPASS UOL.

SUMÁRIO

- INTRODUÇÃO

1.1 VISÃO GERAL

Este documento tem como visto apresentar todas as funcionalidades e descrever bem o uso da **API SERVEREST.** Os testes feitos para comprovar as funcionalidades devidas da API, serão documentados nesse documento. A **API SERVEREST** tem como intuito ser uma loja virtual, onde se tem diversas funcionalidades sendo elas por exemplo cadastrar um usuário, fazer login com o mesmo e dentre outras...

1.2 PARA QUE SERVE ESTE DOCUMENTO

Dentro desse Documento vai estar documentada todas as mudanças significativas na API, visando que todos os resultados vão ser documentados, sendo assim ocorrerá uma grande otimização na API.

1.3 OBJETIVOS DE TESTE

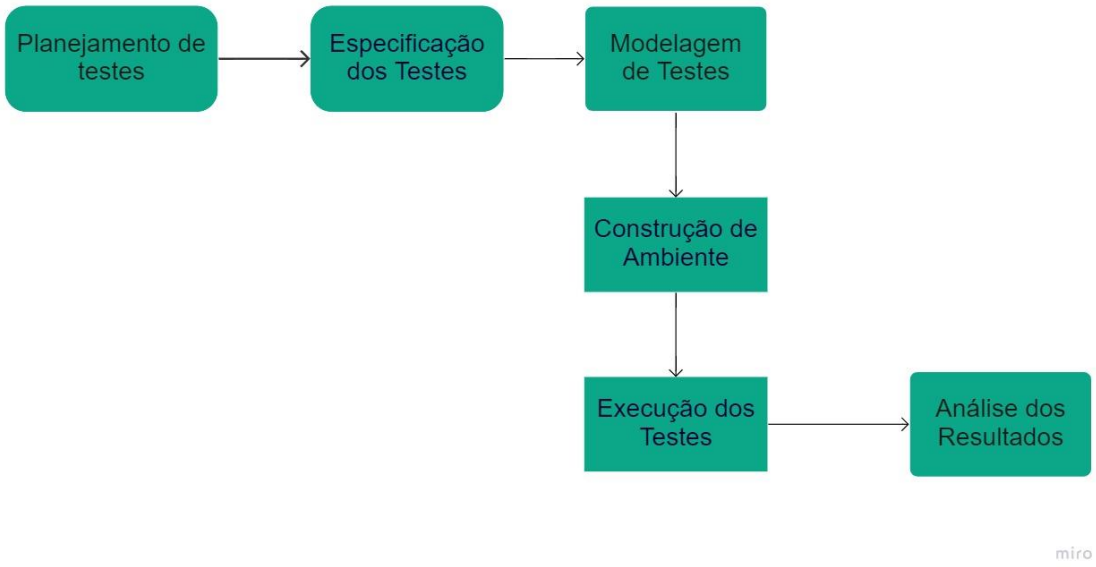
Este documento tem como principal Objetivo:

- Identificar informações de projeto existentes e os componentes de software que devem ser testados.
- Listar os Requisitos a Testar recomendados (alto nível).
- Recomendar e descrever as estratégias de teste a serem empregadas.
- Identificar os recursos necessários e prover uma estimativa dos esforços de teste.
- Listar os elementos resultantes do projeto de testes.

- ESCOPO

A **API SERVEREST** passará pelos testes unitário, funcional, integração, End - To - End e de regressão. Estes testes vão lidar com a qualidade funcional da API testada, a execução do sistema e também os testes ocorreram de forma manual Para garantir a Performance e Compatibilidade da API testada.

2.1 PROCESSOS DE TESTES



Planejamento de Testes: Nesta etapa vai ser elaborado todos os planos de teste da API, basicamente tem como olhar as tarefas que vão ser feitas. Onde vão ser visto os métodos e as demais coisas vão ser definidas, incluindo a forma de acompanhamento do Projeto.

Especificação dos Testes: Basicamente etapa em que a equipe vai analisar o esforço a ser feito, baseado nas expectativas dos Clientes.

Modelagem de Testes: Na etapa de modelagem dos testes á serem feitos, tem como visão e embasamento o que será testado, isto é constituído por um item do componente do sistema que pode ser verificado por um ou mais casos de teste.

Construção do Ambiente: Nesta etapa ocorre á construção de todo um ambiente de testes que visa um ambiente que esteja pronto para sofrer a bateria de testes.

Execução dos Testes: Aqui os Testes serão executados, através da combinação de casos de teste, Ou seja, é hora de transformar condições em Procedimentos.

2.2 TESTE FUNCIONAL

Basicamente os testes funcionais faz com que a API cumpra os requisitos que dela são esperados. Basicamente esse teste reflete à visão do usuário, se baseando nas entradas e determinando as saídas esperadas da **API SERVEREST.**

2.3 TESTE DE UNIDADE

Nesta etapa vamos pegar uma unidade que no caso é a menor unidade testável para o computador, esse Teste consiste em validar dados válidos e inválidos da API sendo aplicado pelo próprio desenvolvedor da API ou o analista de teste.

2.4 TESTE DE INTEGRAÇÃO

Resumidamente os testes de integração servirá para avaliar a funcionalidade dos módulos da **API SERVEREST.** O teste de integração serve para encontrar erros e falhas nas várias interfaces da API.

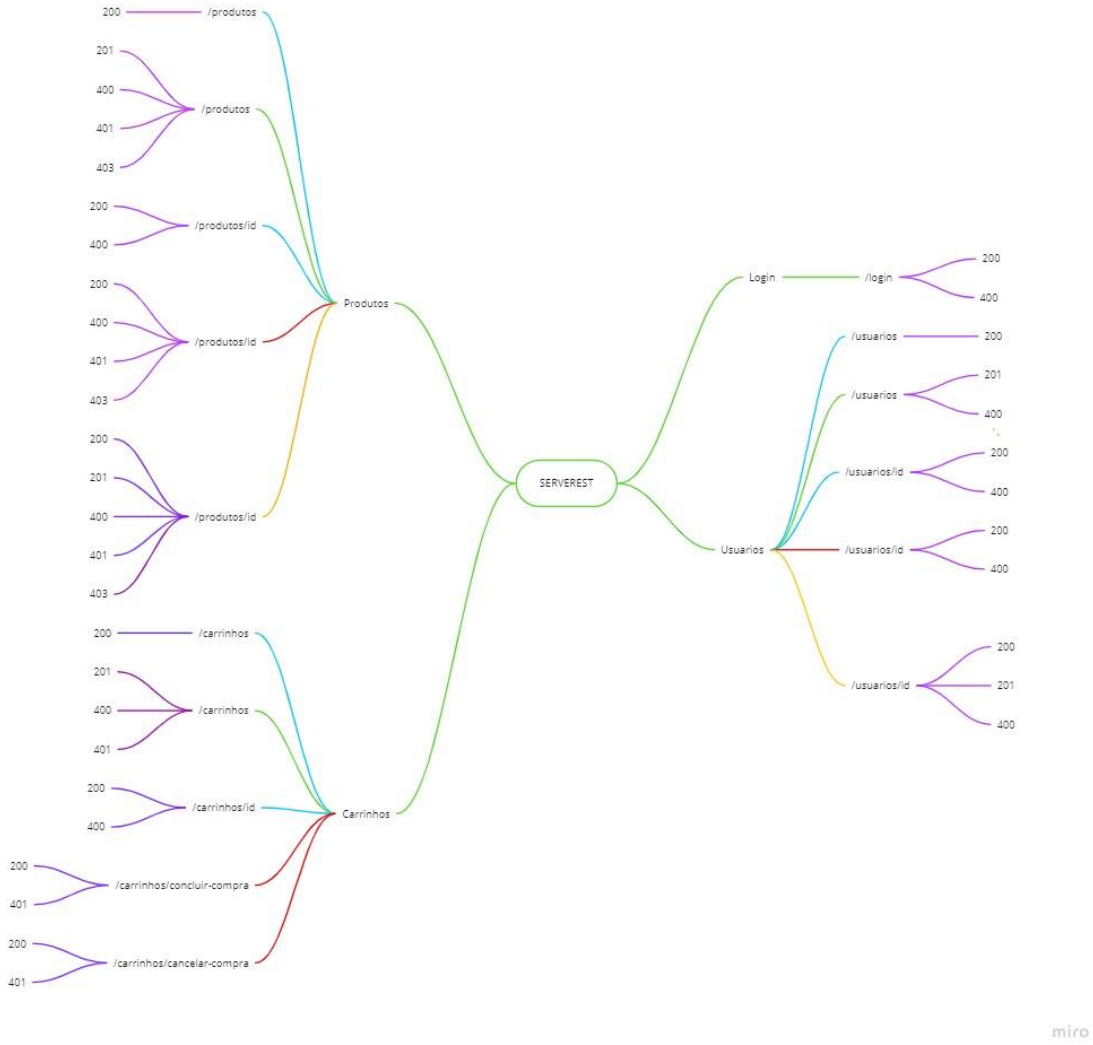
2.5 TESTE DE END-TO-END

O teste end-to-end servirá para ver como a **API SERVEREST** se comporta com a maneira em que vários usuários interagem com ela, valida a regra de negócio e se cada etapa do processo foi concluída corretamente.

2.6 TESTE DE REGRESSÃO

Basicamente o teste de regressão será responsável na **API SERVEREST** para validar as aplicações críticas que necessitam ser testadas com frequência, funcionalidades que são alteradas também serão validadas.

- MAPA MENTAL



Obs: Mapa mental da API acima cos os verbos indicados pelas cores dos mesmos como referenciados na API.

- CASOS DE TESTE

- LOGIN

- Realizar login corretamente inserindo email e senha válidos;
- Realizar login com o campo de email vazio;
- Realizar login com o campo de senha vazio;
- Realizar login com o campo de email inválido;
- Realizar login com o campo de senha inválido;

- USUARIOS

- Cadastrar Usuários

- Cadastrar um usuário com os campos (nome, email, password e administrator) válidos;
- Cadastrar um usuário com o campo nome vazio;
- Cadastrar um usuário com o campo email vazio;
- Cadastrar um usuário com o campo email inválido;
- Cadastrar um usuário com o campo administrator vazio;
- Cadastrar um usuário com o campo administrator inválido;

- Buscar usuários por Id

- Buscar um usuário com Id válido;
- Buscar um usuário com Id não existente;

- Excluir um Usuário

- Excluir um usuário com Id válido;
- Excluir um usuário com Id não existente;

- Alterar dados de um Usuário

- Alterar os dados de um usuário com os campos (nome, email, password e administrator) válidos;
- Alterar um usuário com o campo nome vazio;
- Alterar um usuário com o campo email vazio;
- Alterar um usuário com o campo email inválido;
- Alterar um usuário com o campo administrator vazio;
- Alterar um usuário com o campo administrator inválido;

- PRODUTOS

- Cadastrar Produtos

- Cadastrar um produto com os campos (nome, preco, descricao e quantidade) válidos;

- Cadastrar um produto com o campo nome vazio;
- Cadastrar um produto com o campo preco vazio;
- Cadastrar um produto com o campo preco inválido;
- Cadastrar um produto com o campo preco com valor String;
- Cadastrar um produto com o campo quantidade vazio;
- Cadastrar um produto com o campo quantidade inválido;
- Cadastrar um produto com o campo descrição vazio;

- Buscar produto por Id

- Buscar produto com Id válido;
- Buscar produto com Id inválido;

- Excluir produto

- Excluir produto com Id válido;
- Excluir produto com Id inválido;

- Alterar Produtos

- Alterar um produto com o campo nome vazio;
- Alterar um produto com o campo preco vazio;
- Alterar um produto com o campo preco inválido;
- Alterar um produto com o campo preco com valor String;
- Alterar um produto com o campo quantidade vazio;
- Alterar um produto com o campo quantidade inválido;
- Alterar um produto com o campo descrição vazio;

- CARRINHOS

- Cadastrar um Carrinho

- Cadastrar um carrinho com os Campos Id e quantidade válidos;
- Cadastrar um carrinho com o campo Id inválido;
- Cadastrar um carrinho com o campo Id vazio;
- Cadastrar um carrinho com o campo quantidade com o valor String;
- Cadastrar um carrinho com o campo quantidade vazio;

- Buscar um Carrinho por Id

- Buscar um Carrinho com Id válido;
- Buscar um carrinho com Id inválido;

- Concluir Compra

- Concluir compra com o token válido;
- Concluir compra com o token inválido;

- Cancelar Compra

- Cancelar compra com o token válido;
- Cancelar compra com o token inválido;

- ESTRATEGIA DE TESTE

O objetivo da estratégia de teste é fornecer uma melhora na API SERVEREST, os testes serão todos feitos no **POSTMAN**, ferramenta que serve para realizar os testes a serem feitos na API. Os critérios de avaliação dos testes feitos são estes:

- Cobertura de teste com base em Código
- Número de defeitos
- Cobertura de teste com base nos requisitos

Os testes terão um nível de aceitação quando basicamente 95% dos casos de teste tiverem sido executados com êxito. Os testes executados na API serão os E2E e de regressão.

- CANDIDATOS A TESTE DE REGRESSÃO

Tendo em vista que os testes de regressão são testes que vão validar se nas novas versões do software surgiram defeitos. Portanto é escolhidos alguns casos de teste prioritários na utilização da API para validar a versão do sistema, sendo eles na **API SERVEREST:**

- Cadastrar usuário
- Realizar login
- Cadastrar produto
- Listar produto
- Cadastrar carrinho
- Cancelar/ concluir-compra
- Cancelar /cancelar-compra

-CANDIDATOS A TESTE DE END-TO-END

Testes E2E são feitos para testar um fluxo do começo ao fim. Com o intuito de replicar a realidade dos usuários nos testes feitos com a intenção de validar o que está sendo testado, na **API SERVEREST** os candidatos a testes E2E são eles:

-Login de usuário

- Cadastrar usuário
- Realizar login

- Efetuar Compra

- Realizar Login
- Listar produtos
- Cadastrar carrinhos
- Concluir compra

- Alterar dados de um usuário

- Realizar login
- Listar usuários
- Editar usuário

- Exclusão de um usuário

- Listar usuário
- Excluir usuário

- Editar dados de um produto

- Realizar login
- Listar produtos
- Editar produto

- Cancelamento de Compra

- Realizar login
- Listar produtos
- Cadastrar carrinhos
- Cancelar compra

- PONTOS CRÍTICOS

Os pontos críticos na **API SERVEREST** são basicamente os endpoints de cadastro de usuário, realização de login, cadastro de produtos, cadastro de carrinhos e cancelar/concluir á compra. Se houver algum tipo de erro ou falha na API principalmente no endpoint de realização de login, haverá sérios riscos porque com ele falhando não podemos dar seguimento aos outros Endpoints da API.

- BUGS ENCONTRADOS

Com o passar dos testes se percebe que a **API SERVEREST** possui alguns erros e alguns aspectos que podem ser melhorados. Os principais erros que encontrei durante os testes na API foram o de status de retorno das requisições. Por exemplo em um teste para realizar a busca de um usuário com Id não existente, o teste te retorna um código “400” onde deveria ser “404” que significa não encontrado e seria o correto e o que a API deveria entregar. Outra Falha que ocorre na API, é nos seus campos de respostas, os campos de resposta da API são bem escassos em um erro com mensagens que poderiam ser mais especificas.

- SOLUÇÕES

Como solução deveria ser adicionado os status de código que faltam na API, a correção desse erro seria de grande importância para otimização da API. Solucionando a falha nos campos de resposta, ou seja, melhorando as respostas que a API leva para o usuário, para uma maior compreensão do usuário que está usando ou até mesmo os analistas de teste que vão fazer todo aquele esquema de testes e em que seus campos de resposta te devolve algo tão escasso principalmente no campo de “token”.

- TESTES AUTOMATIZADOS(POSTMAN)

A ferramenta de automatização dos testes foi o POSTMAN. O foco principal foi nos testes de regressão e nos testes E2E. Os testes automatizados ocorreram em todos os casos de teste que estão listados mais acima, cada endpoint teve testes para validar suas devidas atividades.

- REPORT HTML VIA NEWMAN