**Serverest**

# Introdução

O serverest é uma API REST gratuita que simula uma loja virtual com intuito de servir de material de estudos de testes de API.

# Arquitetura

O Sistema é desenvolvido na linguagem JAVASCRIPT.

Para o armazenamento, consulta e alteração de dados da aplicação, será usada uma API que disponibiliza um conjunto de serviços REST. Funciona como na imagem, em que o cliente (aplicação) envia uma requisição à uma URL (endpoint) para acessar os serviços da API, e tem como retorno um objeto na forma de um JSON.

# Funcionalidades

|  |  |  |  |
| --- | --- | --- | --- |
| **Funcionalidades** | **Comportamento Esperado** | **Verificações** | **Critérios de Aceite** |
| Cadastro(Usuário) | Ao digitar nome, email, ,senha, Administrador(Boolean) irá efetuar um cadastro na API. | * Senha min 8 caracteres e no máximo 18 * Exibir uma mensagem de confirmação em caso positivo com o id do usuario. * Exibir a mensagem de falha em caso de usuário existente. |  |
| Cadastro(Produto) | Ao digitar nome, preço, ,descrição, quantidade irá efetuar um cadastro de produto na API. | * Exibir uma mensagem de confirmação em caso positivo com o id do produto. * Exibir a mensagem de falha em caso de produto com mesmo nome existente. |  |
| Cadastro(Carrinho) | Ao digitar o id do produto e a quantidade, o carrinho será criado.  Só poderá cadastrar um carrinho por usuario.  Ao cadastrar um carrinho, ao inserir um produto, a quantidade do mesmo será reduzida no Sistema. | * Exibir uma mensagem de confirmação em caso positivo com o id do carrinho. * Exibir a mensagem de falha em caso do carrinho possui um produto com mesmo nome existente, ou se houver mais de um carrinho, ou o produto não possuir quantidade disponivel. |  |
| Edição(Usuario) | Ao inserir o id do usuário , o nome, email, password, Administrador(Boolean) irá atualizar o usuario com sucesso. | Exibir uma mensagem de confirmação em caso positivo .  Exibir a mensagem de falha em caso do email já esteja sendo usado. |  |
| Edição(Produto) | Ao inserir o id do produto , o nome, preço, descrição e quantidade, irá atualizar o produto com sucesso.  Se o id do usuario nao for encontrado, o Sistema irá criar um novo usuario. | Exibir uma mensagem de confirmação em caso positivo .  Exibir a mensagem de falha em caso do nome do produto já esteja sendo usado. |  |
| Remover (Usuario) | Ao inserir o id do usuario para remover, o mesmo será removido.  Não é permitido excluir usuario com carrinho vinculado. | Exibir uma mensagem de confirmação em caso positivo .  Exibir a mensagem de falha em caso do usuario possuir um carrinho cadastrado. |  |
| Remover (Produto) | Ao inserir o id do produto, sera removido com sucesso.  Não poderá remover um produto que estiver em um carrinho. | Exibir uma mensagem de confirmação em caso positivo .  Exibir a mensagem de falha em caso do produto estiver vinculado a um carrinho. |  |
| Remover (Carrinho) – concluir compra | Ao concluir uma compra, o carrinho é excluido.  O carrinho a ser removido será o que está vinculado ao usuario. | Exibir uma mensagem de confirmação em caso positivo .  Exibir a mensagem de falha em caso do token nao for encontrado. |  |
| Remover (Carrinho) – cancelar compra | Ao cancelar uma compra, o carrinho é excluido e o produto volta para o estoque.  O carrinho a ser removido será o que está vinculado ao usuario. | Exibir uma mensagem de confirmação em caso positivo .  Exibir a mensagem de falha em caso do token nao for encontrado. |  |
| Listar(Usuario) | Ao buscar os usuarios, deverá retornar uma lista com os usuarios e a quantidade cadastrados. | Em caso de sucesso, será listado os usuarios. |  |
| Listar(Produto) | Ao buscar os produtos, deverá retornar uma lista com os produtos e a quantidade cadastrados. | Em caso de sucesso, será listado os produtos. |  |
| Listar(Carrinho) | Ao buscar o carrinho, sera retornado um unico por usuario, exibindo quantidade, os produtos do carrinho, preço total, quantidade total e o id do usuario. | Em caso de sucesso, será listado os carrinhos. |  |
| Listar(Usuario)- Por id | Ao inserir o id do usuario, o mesmo será listado com nome, email, senha, administrador(Boolean) e o id.  Se o usuario não for encontrado, será exibida uma mensagem “usuario nao encontrado”. | Em caso de sucesso, será listado o usuario pelo id pesquisado |  |
| Listar(Produto)- Por id | Ao inserir o id do produto, o mesmo será listado com nome, preço, descrição, quantidade e o id.  Se o produto não for encontrado, será exibida uma mensagem “produto nao encontrado”. | Em caso de sucesso, será listado o produto pelo id pesquisado |  |
| Listar(Carrinho)- Por id | Ao buscar o carrinho por id, sera retornado um unico por usuario, exibindo os produtos do carrinho, preço total, quantidade total e o id do usuario. | Em caso de sucesso, será listado o carrinho pelo id pesquisado |  |
| Login | Ao inserir email e senha, previamente cadastrado, o usuario irá logar no Sistema, tendo um token(bearer) como resposta | Em caso de sucesso, o usuario irá entrar no sistema.  Em caso de Email/senha invalidos, sera exibida uma mensagem: “Email e/ou senha inválidos”. |  |

# Estratégia de Teste

* **Escopo de Testes**

O plano de testes abrange todas as funcionalidades descritas na tabela acima.

Serão executados testes conforme a descrição abaixo.

Testes Automatizados: Serão realizados testes end-to-end para todas as funcionalidades do Sistema.

* **Ambiente e Ferramentas**

Os testes serão feitos do ambiente de homologação, e contém as mesmas configurações do ambiente de produção com uma massa de dados gerada previamente pelo time de qualidade.

As seguintes ferramentas serão utilizadas no teste:

|  |  |  |
| --- | --- | --- |
| Ferramenta | Time | Descrição |
| POSTMAN | Qualidade | Ferramenta para realização de testes de API |
| Robot Framework | Qualidade | Ferramenta de teste End-to-end, e tambem para a criação dos cenarios com a sintaxe Gherkin. |

# Classificação de Bugs

Os Bugs serão classificados com as seguintes severidades:

|  |  |  |
| --- | --- | --- |
| ID | Nivel de Severidade | Descrição |
| 1 | Blocker | * Bug que bloqueia o teste de uma função ou feature causa crash na aplicação. * Botão não funciona impedindo o uso completo da funcionalidade. * Bloqueia a entrega. |
| 2 | Grave | * Funcionalidade não funciona como o esperado * Input incomum causa efeitos irreversíveis |
| 3 | Moderada | * Funcionalidade não atinge certos critérios de aceitação, mas sua funcionalidade em geral não é afetada * Mensagem de erro ou sucesso não é exibida |
| 4 | Pequena | * Quase nenhum impacto na funcionalidade porém atrapalha a experiência * Erro ortográfico * Pequenos erros de UI |

# Definição de Pronto

Será considerada pronta as funcionalidades que passarem pelas verificações e testes descritas nestes TestPlan, não apresentarem bugs com a severidade acima de Minor, e passarem por uma validação de negócio de responsabilidade do time de produto.