**Test Plan Editar Categorias**

**1**                **Introdução**

Editar categorias é um sistema de e-commerce onde o propósito dele é editar as categorias a fim de alterar a organização de nossos produtos

**2**                **Arquitetura**

O framework utilizado para a implementação do back-end do Editar Categorias  é o Visual Studio, que possui uma arquitetura orientada a Objetos.

Para o armazenamento e consulta de dados das categorias, 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.

**3**                **Funcionalidades**

|  |  |  |  |
| --- | --- | --- | --- |
| **Funcionalidades** | **Comportamento Esperado** | **Verificações** | **Critérios de Aceite** |
| **Edição** | * Ao editar uma categoria o usuário deve ser direcionado para a tela onde deve inserir data de modificação e hora de modificação. * Ao digitar a data e hora de modificação o usuário deve ser direcionado para a tela onde vai ativar a categoria editada. * Deve ser exibido uma caixa de texto para o usuário escrever a categoria que deseja editar.   . | * Nome da categoria (até 128 caracteres; somente alfabeto); * Status booleano; * Data de criação; * Data de modificação.   ●                Usuário não preencher campos obrigatórios   * Ultrapassar  128 caracteres | * Editar a categoria que foi cadastrada; * Deve ser possível ativar e inativar a categoria; * Deve ser possível alterar o nome da categoria; * Toda categoria alterada deve ser registrada com a data e hora de modificação. |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**4**                **Estratégia de Test**

●                **Escopo de Testes**

O plano de testes abrange todas as funcionalidades descritas na tabela acima. Esse plano de testes exclui a funcionalidade Cadastrar e Pesquisar Categorias.

Serão executados testes em todos os níveis conforme a descrição abaixo.

Testes Unitários: o código terá uma cobertura de 60% de testes unitários, que são de responsabilidade dos desenvolvedores.

Testes de Integração: Serão executados testes de integração em todos os endpoints, e esses testes serão de responsabilidade do time de qualidade.

Testes Manuais: Todas as funcionalidades serão testadas manualmente pelo time de qualidade seguindo a documentação de Cenários de teste e destes TestPlan.

●                **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 |
| Jasmine | Desenvolvimento | Framework utilizada para testes unitários |
| Selenium | Qualidade | Ferramenta para testes end-to-end |

**5**                **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 |

**6**                **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.