|  |
| --- |
| Alura - 10/03/2022  **TEST PLAN** |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Plan DantasIT** | | | |
| **Cliente** | Alura | **Projeto** | AluraPic |
| **Elaborado por** | Felipe Dantas | **Última atualização** | 10/03/2022 |

# INTRODUÇÃO

Alura Pic é um sistema de gestão de fotos, o sistema foi criado com o propósito de dar suporte para a formação em Angular. Conta com funcionalidades simples de Cadastro, Login e Manipulação de fotos.

# **Arquitetura**

O framework utilizado para a implementação do front-end do AluraPic é o Angular, que possui uma arquitetura orientada a componentes.

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 | Ao digitar e-mail, nome completo, usuário e senha irá efetuar um cadastro na plataforma | Senha com tamanho mínimo de 8 caracteres |  |
| Login | Ao digitar seu usuário e senha corretamente o usuário irá logar na plataforma | Login no Sistema  Usuário inválido  Campo Obrigatório | O Nome e a Senha devem ser obrigatórios  O sistema deve funcionar em todos os navegadores |

# **Estratégia de Testes**

**Escopo de Testes:**

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

Esse plano de testes exclui a funconalidade de upload de fotos.

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 Automatizados: serão realizados testes end-to-end na funcionalidade de Login

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

Versão Beta: será lançada uma versão beta para 20 usuários pré-cadastrados antes do release.

**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 utilizado para testes unitários |
| Selenium | Qualidade | Ferramenta para testes end-to-end |

# **Classificação de Bugs**

Os bugs serão classificados com as seguintes severidades:

|  |  |  |
| --- | --- | --- |
| **ID** | **NÍVEL DE SEVERIDADE** | **DESCRIÇÃO** |
| 1 | Blocker | Bug que bloqueia o teste de uma função ou feature causa crash na aplicação  O botão não funciona impedindo o uso completo da funcionalidade  Bloqueia release |
| 2 | Crítica | Feature funciona pobremente:  Input esperados causam crash ou efeitos indesejados  Input incomum causa efeitos irreversíveis |
| 3 | Moderada | Feature não atinge certos critérios de aceitação, mas sua funcionalidade em geral não é afetada:  Mensagem de erro ou sucesso não é exibida  Input incomum causar efeitos indesejados, mas contornáveis |
| 4 | Pequena | Quase nenhum impacto na funcionalidade, mas ainda é um erro válido:  Erro ortográfico  Pequenos erros de UI |

1. **Definição de Pronto**

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