# 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 email, nome completo, usuário e senha irá efetuar um cadastro na plataforma e o  Usuário deverá ser redirecionado para a tela de login.  Deve indicar o campo obrigatório a ser corrigido pelo usuário. | * Senha min 8 caracteres * Todos os campos devem ser obrigatórios. * Exibir uma mensagem de confirmação em caso “positivo”. * Redirecionar o usuário para a tela de login. * Exibir a mensagem “falha” em caso de usuário existente | * Após logar com user e senha,exibir mensagem “logado com sucesso.” |
| Login | Ao digitar seu usuário e senha corretamente o usuário irá logar na plataforma.  Ao tentar se logar e falhar 3 vezes consecutivas o usuário terá que esperar 15 minutos para tentar logar novamente.  O sistema deve aceitar usuários que já estão logados em outro device se logar novamente | * Login do sistema com sucesso. * Usuário inválido. * Usuário não preencher campo obrigatório. * Senha incorreta. * Senha incorreta 3 vezes. | * Ter acessibilidade no sistema * Redimensionar a Tela |

# Estratégia de Teste

* **Escopo de Testes**

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

Esse plano de testes exclui a funcionalidade 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 Cenário de teste e destes TestPan.

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

* **Ambiente e Ferramentas**

Sinalizar onde será o ambiente de teste e as ferramentas que serão utilizadas.

Os testes serão feitos no ambiente 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 podem classificados com as seguintes severidades:

| ID | Nivel de Severidade | Descrição |
| --- | --- | --- |
| 1 | Blocker |  |
| 2 | Critical |  |
| 3 | Major |  |
| 4 | Minor |  |

# Definição de Preparado

# Definir o que a tarefa precisa para estar no backlog.

# Definição de Pronto

# Definir o que a tarefa precisa para estar pronta.

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.