Test Plan Avaliaí

1 Introdução

Avaliaí é um sistema de classificação de disciplinas desenvolvido por estudantes e para estudantes, tendo por objetivo principal permitir que alunos consigam visualizar os pontos fortes e fracos das disciplinas que desejam ou irão cursar, tal como o que outras pessoas acharam dessas disciplinas, com base em suas experiências. O sistema conta com funcionalidades como: permitir a avaliação de disciplinas, fazer comentários sobre disciplinas, sugerir a criação de uma nova disciplina no sistema, classificar comentários de outros alunos.

2 Arquitetura

Para o front-end, o Avaliaí utiliza o framework Next.JS, uma estrutura da web de desenvolvimento front-end React, que consequentemente possui uma arquitetura orientada a componentes, contribuindo para a sua consequente reutilização.

O back-end utiliza o Spring-Boot, na linguagem Java, e ferramentas como o Insomnia, que permite fazer requisições de API. Para a documentação dos endpoints, foi utilizada a ferramenta Swagger.

No ambiente mobile, o framework Flutter foi utilizado.

Para que o front-end e o back-end estejam integrados, é utilizada uma API que segue os padrões REST. A aplicação envia uma requisição a um determinado endpoint, dependendo de sua função, para ter acesso aos serviços da API, que retorna um código em JSON.

O grupo optou por utilizar o RabbitMQ como serviço de mensageria, necessário para a requisição de disciplinas e consequente aprovação ou reprovação pelo administrador.

3 Funcionalidades

Funcionalidade	RF 009 - O usuário deve ser capaz de criar uma conta com o e-mail PUC Minas.
Comportamento Esperado	Inserindo corretamente o nome, e-mail, senha, instituição e curso, o usuário será redirecionado para a tela de login. Na apresentação de alguma falha, o sistema deve indicar o campo obrigatório a ser corrigido pelo usuário.
Verificações	● O e-mail do usuário deve possuir a assinatura @sga.pucminas.br ● Todos os campos devem ser obrigatórios. ● Redirecionar o usuário para tela de login. ● Exibir mensagem de falha no caso de campo obrigatório incompleto ou preenchido erroneamente.
Critérios de Aceite	● O e-mail do usuário deve possuir a assinatura @sga.pucminas.br ● Todos os campos devem ser obrigatórios. ● Redirecionar o usuário para tela de login. ● Exibir mensagem de falha caso o campo obrigatório esteja incompleto ou preenchido erroneamente.

Funcionalidade	Login
Comportamento Esperado	Inserindo corretamente seu e-mail PUC Minas e a senha, o usuário será redirecionado para a página inicial da aplicação.
Verificações	● Usuário logado com sucesso no sistema

	● E-mail Inválido ● Senha inválida ● Usuário não preencheu campo obrigatório
Critérios de Aceite	● Em caso de sucesso, redirecionar usuário para página inicial

Funcionalidade	RF 004 - O usuário deve ser capaz de visualizar a lista de disciplinas ofertadas pelo curso de engenharia de software.
Comportamento Esperado	Ao entrar na tela inicial, o usuário irá ver uma lista contendo todas as disciplinas de seu curso (nesse caso, engenharia de software).
Verificações	Disciplinas resgatadas do banco
Critérios de Aceite	Todas as disciplinas cadastradas do curso aparecerem

Funcionalidade	RF 001 - O usuário deve conseguir avaliar uma determinada disciplina com uma nota de 1 a 5
Comportamento Esperado	Ao abrir a página da disciplina, o usuário deve ser capaz de selecionar uma nota para a disciplina (de 1 a 5)
Verificações	● Mensagem de confirmação em caso positivo de avaliação.
Critérios de Aceite	Avaliação recente da disciplina influenciar na média de avaliação

Funcionalidade	RF 005 - O usuário deve ser capaz de pesquisar determinada disciplina por seu nome no sistema.
Comportamento Esperado	Ao digitar um texto na barra de pesquisa, deverá aparecer os resultados das disciplinas correspondentes. Deve ser exibida uma mensagem caso não haja livros correspondentes a pesquisa do usuário.
Verificações	● Pesquisa encontrou os resultados esperados ● Pesquisa não encontrou resultados
Critérios de Aceite	Retornar a disciplina encontrada ou não

Funcionalidade	RF 002 - O usuário pode incluir em sua avaliação um texto para justificar a nota daquela matéria.
Comportamento Esperado	Ao avaliar a disciplina, o usuário deve ser capaz de também adicionar uma mensagem de texto, justificando sua avaliação. A mensagem de texto aparecerá juntamente com sua avaliação.
Verificações	
Critérios de Aceite	Associar o comentário do usuário com sua respectiva avaliação na barra de comentários

Funcionalidade	RF 003 - O usuário deve conseguir aprovar ou desaprovar a avaliação de outros usuários através de um sistema de upvote e downvote

Comportamento Esperado	Ao analisar os comentários de outros estudantes, o usuário deve ser capaz de avaliá-los positiva ou negativamente (like-dislike) Sua avaliação impactará na nota do comentário avaliado.
Verificações	● Avaliação positiva ● Avaliação negativa
Critérios de Aceite	Alterar a nota do comentário avaliado.

Funcionalidade	RF 006 - O usuário deve ser capaz de filtrar disciplinas por período no sistema.
Comportamento Esperado	Ao observar a listagem de disciplinas, o usuário deve ser capaz de filtrá-las por período. Feito o processo, só serão mostradas as disciplinas que atendam aos requisitos estabelecidos.
Verificações	● Filtro escolhido
Critérios de Aceite	Mostrar apenas as disciplinas que atendam aos filtros estabelecidos.

Funcionalidade	RF 007 - O usuário deve visualizar dados das disciplinas, como carga horária, modalidade(síncrona)
Comportamento Esperado	Ao selecionar uma disciplina, o usuário deve ser redirecionado para uma página especial e ser capaz de ver todos os seus dados – carga horária, modalidade, curso, etc.

Verificações	
Critérios de Aceite	Mostrar todos os dados da disciplina selecionada.

Funcionalidade	RF 010 - O usuário deve ser capaz de denunciar comentários.
Comportamento Esperado	Ao analisar a seção de comentários, o usuário deve poder denunciá-los. Uma mensagem aparece na tela quando a denúncia foi concluída.
Verificações	
Critérios de Aceite	Usuário denunciado.

Funcionalidade	RF 011 - O usuário preenche formulários de requisição para adicionar disciplinas.
Comportamento Esperado	Ao clicar na opção "requerer disciplina”, o usuário deve ser direcionado para uma tela onde preencherá as informações da disciplina. Após preenchido corretamente, uma mensagem informa ao usuário que a requisição passará por um processo de análise.
Verificações	
Critérios de Aceite	Requisição direcionada para os moderadores.

Funcionalidade	RF 012 - O moderador valida o banimento de um usuário.

Comportamento Esperado	Quando algum usuário é denunciado, o moderador recebe uma notificação para avaliar o caso. A partir disso, ele pode escolher banir o usuário (por tempo temporário ou vitalício) ou não.
Verificações	● Banir usuário por tempo determinado ● Banir usuário por tempo indeterminado ● Não banir
Critérios de Aceite	Decisão refletida na conta do usuário denunciado.

Funcionalidade	RF 013 - O moderador valida a adição de uma disciplina.
Comportamento Esperado	Quando algum usuário requisita uma disciplina, o moderador recebe uma notificação a respeito. Ele pode aprová-la ou não.
Verificações	● Disciplina aprovada ● Disciplina desaprovada
Critérios de Aceite	Decisão reflete na existência da disciplina no sistema.

4 Estratégia de Teste

- **Escopo de Testes**

O plano de testes abrange todas as funcionalidades descritas na tabela acima; ou seja, todos os requisitos funcionais do sistema.

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.

- Testes Automatizados: Serão realizados testes end-to-end na funcionalidade de Login.

- Testes Manuais: Todas as funcionalidades serão testadas manualmente, seguindo a documentação de Cenários de teste e destes Test Plan.

- **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.

As seguintes ferramentas serão utilizadas no teste:

Ferramenta	**Time**	**Descrição**
Insomnia	Desenvolvimento	Ferramenta para realização de testes de API
JUnit	Desenvolvimento	Framework utilizada para testes unitários Java
Selenium	Desenvolvimento	Prover evidências dos testes

5 Classificação de Bugs

Os Bugs serão classificados com as seguintes severidades:

ID	**Nivel de Severidade**	**Descrição**
1	Bloqueadora	• Bug que bloqueia o teste de uma função ou feature. Causa crash na aplicação. • Ex: O Botão não funciona impedindo o uso completo da funcionalidade.
2	Grave	• Funcionalidade não funciona como o esperado

		● Ex: Input incomum causa efeitos irreversíveis
3	Moderada	● Funcionalidade não atinge certos critérios de aceitação, mas sua funcionalidade em geral não é afetada ● Ex: Mensagem de erro ou sucesso não é exibida
4	Pequena	● Quase nenhum impacto na funcionalidade. porém atrapalha a experiência ● Ex: Erro ortográfico ● Ex: Pequenos erros de UI

6 Definição de Pronto

Será considerada ***pronta*** as funcionalidades que passarem pelas verificações e testes descritas nestes Test Plan e não apresentarem bugs com a severidade acima de Pequena.