Levantamento de Requisitos - Sistema 'Cores do Amanhã'

1. Visão Geral do Sistema

Nome do Sistema: Cores do Amanhã
Objetivo: Gerenciar o acesso de crianças a atividades culturais e artísticas, viabilizando o cadastro de alunos, responsáveis, professores e apoiadores, além de permitir que administradores tenham controle e visualização de relatórios e doações.

2. Requisitos Funcionais (RF)

- RF01: O sistema deve permitir o cadastro de alunos com vínculo a um responsável.
- RF02: O sistema deve permitir a inscrição de alunos em aulas de música, desenho, etc.
- RF03: O sistema deve disponibilizar os horários das aulas cadastradas.
- RF04: O sistema deve permitir o cadastro de apoiadores com opção de doações pontuais ou assinatura mensal.
- RF05: O sistema deve permitir o cadastro de professores, com seus horários disponíveis e disciplinas ministradas.
- RF06: O sistema deve permitir que administradores acessem e gerenciem todos os cadastros.
- RF07: O sistema deve gerar relatórios de doações mensais.
- RF08: O sistema deve permitir login diferenciado por perfil: aluno/responsável, apoiador, professor e administrador.

3. Requisitos Não Funcionais (RNF)

- RNF01: O sistema deve ser responsivo e acessível para dispositivos móveis.
- RNF02: O sistema deve estar disponível 99% do tempo (alta disponibilidade).
- RNF03: Os dados dos usuários devem ser armazenados de forma segura e com criptografia de senhas.
- RNF04: O sistema deve ser capaz de suportar ao menos 200 acessos simultâneos.
- RNF05: O sistema deve seguir os critérios de usabilidade definidos pela norma ISO/IEC 25010.

4. Critérios de Aceitação com Cucumber (BDD)

Feature: Cadastro de aluno com responsável

Como responsável

Quero cadastrar meu filho para atividades culturais

Para que ele possa participar das aulas programadas

Scenario: Cadastro de novo aluno e responsável

Dado que estou na página de cadastro

Quando preencho corretamente os dados do aluno e do responsável

E clico em "Cadastrar"

Então o sistema deve salvar os dados

E exibir a mensagem "Cadastro realizado com sucesso"

Feature: Doação por assinatura

Como apoiador

Quero me inscrever para uma doação mensal

Para que eu possa apoiar a ONG continuamente

Scenario: Cadastro de apoiador com assinatura mensal

Dado que estou na página de doações

Quando seleciono a opção "Doação mensal"

E informo meus dados e forma de pagamento

E clico em "Confirmar"

Então o sistema deve registrar a assinatura

E exibir a mensagem "Obrigado pelo seu apoio mensal!"

5. Plano de Testes

Testes de Registro e Autenticação

Teste 001 – Cadastro de Apoiador

- **ID:** T001
- **Requisito:** RF01 - O sistema deve permitir o cadastro de novos usuários com diferentes perfis.
- **Descrição:** Verificar se é possível cadastrar um novo apoiador através do formulário de registro.

- **Componente Testado:** RegistroApoiador (Assumido)
- **Dados de Entrada:** Nome Completo, Email, CPF, Senha, Confirmação de Senha.
- **Resultado Esperado:** Mensagem de sucesso "Apoiador cadastrado com sucesso!" e redirecionamento para a página de login.
- **Critérios de Aceitação:**
    - Todos os campos obrigatórios são preenchidos.
    - Email deve ser um formato válido.
    - CPF deve ser um formato válido e único (simulado).
    - Senhas devem ser iguais e ter segurança mínima (ex: 8 caracteres).
    - A API de cadastro é chamada com os dados corretos.

Teste 002 – Cadastro de Aluno

- **ID:** T002
- **Requisito:** RF01 - O sistema deve permitir o cadastro de novos usuários com diferentes perfis.
- **Descrição:** Verificar se é possível cadastrar um aluno através do formulário de registro.
- **Componente Testado:** RegistroAluno (Assumido)
- **Dados de Entrada:** Nome, idade do aluno; nome, CPF e telefone do responsável.
- **Resultado Esperado:** Mensagem de sucesso "Aluno cadastrado com sucesso!" e redirecionamento.
- **Critérios de Aceitação:**
    - Todos os campos obrigatórios são preenchidos.
    - Idade do aluno é válida (ex: 4 a 17 anos).
    - CPF do responsável é válido.
    - A API de cadastro é chamada com os dados corretos.

Teste 003 – Cadastro de Professor

- **ID:** T003
- **Requisito:** RF01 - O sistema deve permitir o cadastro de novos usuários com diferentes perfis.
- **Descrição:** Verificar se é possível cadastrar um professor através do formulário de registro.
- **Componente Testado:** CriarPerfilProfessor (ou RegistroProfessor)
- **Dados de Entrada:** Nome Completo, Email, CPF, Telefone (opcional), Matéria Principal (opcional).
- **Resultado Esperado:** Mensagem de sucesso e limpeza dos campos do formulário.
- **Critérios de Aceitação:**
    - Todos os campos obrigatórios são preenchidos (nome, email, CPF).
    - Email e CPF são válidos.
    - console.log chamado com os dados corretos.

Teste 004 – Login de Usuário

- **ID:** T004
- **Requisito:** RF02 - O sistema deve permitir que usuários registrados façam login.
- **Descrição:** Verificar se um usuário pode fazer login com credenciais válidas.
- **Componente Testado:** Login (Assumido)
- **Dados de Entrada:** Email e Senha válidos.
- **Resultado Esperado:** Redirecionamento para o dashboard/perfil do usuário e token de autenticação armazenado.
- **Critérios de Aceitação:**
    - A API de login é chamada com as credenciais corretas.
    - Em caso de sucesso, o useAuth é atualizado e o usuário é redirecionado.

Testes de Navegação e Layout (Navbar, Sidebars)

Teste 005 – Navegação da Barra de Navegação (Navbar)

- **ID:** T005
- **Requisito:** RF03 - O sistema deve ter uma barra de navegação responsiva com links funcionais.
- **Descrição:** Verificar se a Navbar exibe os links corretos de acordo com o status de autenticação do usuário e se a navegação ocorre para as URLs esperadas.
- **Componente Testado:** Navbar
- **Dados de Entrada:**
    - Status de usuário: Não logado, Logado (Apoiador, Aluno, Professor, Administrador).
    - Tamanho da tela: Desktop, Mobile.
- **Resultado Esperado:**
    - Links "Login" e "Registro" visíveis quando não logado; "Meu Perfil" e "Sair" visíveis quando logado.
    - Dropdowns "Apoie" e "Perfil" funcionam corretamente.
    - Sidebar mobile abre/fecha.
    - Todos os links levam às URLs corretas (ex: /login, /menu-registro, /perfil, /perfil-aluno, /assinaturas, etc.).
    - logout é chamado e redireciona para /login ao clicar em "Sair".
- **Critérios de Aceitação:**
    - Links são encontrados por seus nomes acessíveis.
    - Atributos href dos links correspondem às rotas.
    - Funções mockNavigate e mockLogout são chamadas corretamente.
    - Classes CSS de estado (ex: active) são aplicadas/removidas.
    - Comportamento responsivo de visibilidade dos elementos.

Teste 006 – Navegação da Sidebar do Administrador

- **ID:** T006
- **Requisito:** RF04 - O sistema deve fornecer um painel administrativo com navegação específica para administradores.
- **Descrição:** Verificar se a sidebar do perfil do administrador exibe os links corretos e se a navegação interna entre seções funciona.
- **Componente Testado:** SidebarPerfilADM
- **Dados de Entrada:**
    - Seção ativa (secaoAtiva): "relatorios", "criarTurma", etc.
    - Estado do menu mobile (isMobileOpen).
- **Resultado Esperado:**
    - Todos os links de navegação administrativa estão presentes.
    - O link da secaoAtiva tem a classe "active".
    - Clicar em um link chama setSecaoAtiva com a seção correta.
    - Em mobile, o botão de hambúrguer abre/fecha o sidebar e clicar em um link também fecha o sidebar.
- **Critérios de Aceitação:**
    - Links são encontrados por seus nomes acessíveis.
    - mockSetSecaoAtiva é chamada corretamente.
    - mockToggleMobileMenu é chamada corretamente em cenários mobile.
    - Classes CSS (active, mobile-open) são aplicadas.

Teste 007 – Navegação da Sidebar do Aluno

- **ID:** T007
- **Requisito:** RF05 - O sistema deve fornecer um painel do aluno com navegação para suas informações e aulas.
- **Descrição:** Verificar se a sidebar do perfil do aluno exibe os links corretos e se a navegação interna entre seções funciona.

- **Componente Testado:** SidebarPerfilAluno
- **Dados de Entrada:** Similar ao T006, adaptado para as seções do aluno.
- **Resultado Esperado:** Similar ao T006, adaptado para as seções do aluno (Dados Pessoais, Faltas, Minhas Matérias, Minha Sala, Deletar Conta).
- **Critérios de Aceitação:** Similar ao T006.

Teste 008 – Navegação da Sidebar do Professor

- **ID:** T008
- **Requisito:** RF06 - O sistema deve fornecer um painel do professor com navegação para suas turmas e chamada.
- **Descrição:** Verificar se a sidebar do perfil do professor exibe os links corretos e se a navegação interna entre seções funciona.
- **Componente Testado:** SidebarPerfilProfessor
- **Dados de Entrada:** Similar ao T006, adaptado para as seções do professor.
- **Resultado Esperado:** Similar ao T006, adaptado para as seções do professor (Dados Pessoais, Turmas / Chamada).
- **Critérios de Aceitação:** Similar ao T006.

Testes de Formulários e Gerenciamento (ADM)

Teste 009 – Criação de Card de Doação

- **ID:** T009
- **Requisito:** RF07 - O administrador deve ser capaz de criar novos cards de doação.
- **Descrição:** Verificar se o formulário para criar cards de doação funciona corretamente.
- **Componente Testado:** CriarCardDoacao
- **Dados de Entrada:** Título, Descrição, Meta, URL da Imagem.
- **Resultado Esperado:** Formulário preenchido com dados corretos, submissão dispara console.log com os dados e campos são limpos.

- **Critérios de Aceitação:**
    - Todos os campos são renderizados e podem ser preenchidos.
    - handleSubmit é chamado com os dados do formulário.
    - Campos são limpos após a submissão.

Teste 010 – Criação de Turma

- **ID:** T010
- **Requisito:** RF08 - O administrador deve ser capaz de criar novas turmas e associar professores.
- **Descrição:** Verificar se o formulário para criar turmas funciona corretamente.
- **Componente Testado:** CriarTurma
- **Dados de Entrada:** Nome da Turma, Descrição, Professor Responsável.
- **Resultado Esperado:** Formulário dispara console.log e alert de desenvolvimento ao ser submetido.
- **Critérios de Aceitação:**
    - Todos os campos são renderizados.
    - handleSubmit é chamado.
    - alert é disparado.

Testes de Gerenciamento de Aluno

Teste 011 – Matérias Disponíveis para Inscrição

- **ID:** T011
- **Requisito:** RF09 - Alunos devem poder visualizar matérias disponíveis e se inscrever.
- **Descrição:** Verificar se o componente exibe as matérias, permite a inscrição via modal e trata o cenário de lista vazia.
- **Componente Testado:** MateriasAluno

- **Dados de Entrada:** Lista de matérias (com dados, vazia), interação do usuário (clique, checkbox).
- **Resultado Esperado:**
    - Matérias exibidas corretamente em cards ou mensagem de "nenhuma matéria".
    - Modal de inscrição abre com detalhes da matéria.
    - Inscrição confirmada via checkbox e botão: dispara console.log/alert e fecha modal.
    - Botão de confirmação desabilitado sem aceitar termos.
- **Critérios de Aceitação:**
    - Elementos visíveis corretos para cada cenário de dados.
    - Modal abre/fecha e exibe os dados da matéria selecionada.
    - Funções handleAbrirModal, handleFecharModal, handleTogglePresenca, handleSalvarChamada são chamadas.

Teste 012 – Minhas Aulas Inscritas

- **ID:** T012
- **Requisito:** RF10 - Alunos devem poder visualizar as aulas em que estão inscritos.
- **Descrição:** Verificar se o componente exibe as aulas inscritas em uma tabela e a mensagem de lista vazia.
- **Componente Testado:** SalaAluno
- **Dados de Entrada:** Lista de aulas (com dados, vazia).
- **Resultado Esperado:**
    - Tabela de aulas com dados corretos ou mensagem de "nenhuma aula".
- **Critérios de Aceitação:**
    - Cabeçalhos da tabela estão presentes.
    - Dados de aula são renderizados nas células da tabela.

Testes de Doações e Assinaturas

Teste 013 – Página de Listagem de Doações

- **ID:** T013
- **Requisito:** RF11 - O sistema deve exibir as iniciativas de doação ativas.
- **Descrição:** Verificar se a página de doações carrega, exibe os cards e lida com estados de carregamento/erro/vazio.
- **Componente Testado:** Doacoes
- **Dados de Entrada:** Respostas da API (sucesso com dados, sucesso vazio, falha).
- **Resultado Esperado:**
    - Mensagens de "Carregando" ou "Erro" ou "Nenhuma doação" são exibidas.
    - Cards de doação são exibidos com informações e barras de progresso corretas.
    - Links de navegação para "doar" funcionam.
- **Critérios de Aceitação:**
    - Spinners/alertas são visíveis/ocultos.
    - Dados da API são renderizados nos cards.
    - axios.get é chamado corretamente.

Teste 014 – Página de Pagamento de Doações

- **ID:** T014
- **Requisito:** RF12 - Usuários devem poder realizar doações para iniciativas específicas.
- **Descrição:** Verificar o fluxo de seleção de valor, método de pagamento, preenchimento de dados (condicional) e submissão da doação.
- **Componente Testado:** DoacoesPagamento

- **Dados de Entrada:** location.state (com/sem ID do projeto), valor da doação, método de pagamento (cartão, Pix, Boleto), dados do doador (logado/não logado, completo/incompleto).
- **Resultado Esperado:**
    - Detalhes do projeto são carregados e exibidos.
    - Interface de pagamento se adapta ao método selecionado.
    - Validações de campos funcionam, exibindo alertas.
    - Submissão da API (axios.post) com payload correto.
    - Redirecionamento para a página /doar e alertas de sucesso/erro.
- **Critérios de Aceitação:**
    - axios.get para detalhes do projeto é chamado.
    - axios.post para doar é chamado com payload e headers corretos.
    - Mensagens de alerta e navegação são acionadas conforme esperado.
    - Campos são exibidos/ocultos condicionalmente.

Teste 015 – Página de Assinatura e Pagamento

- **ID:** T015
- **Requisito:** RF13 - Apoiadores devem poder escolher e gerenciar planos de assinatura.
- **Descrição:** Verificar o fluxo de seleção de plano, método de pagamento, preenchimento de dados do cartão (se aplicável), e submissão da assinatura.
- **Componente Testado:** AssinaturaPagamento
- **Dados de Entrada:** Token de autenticação, plano atual do usuário, seleção de novo plano, método de pagamento, dados do cartão.
- **Resultado Esperado:**
    - Redirecionamento para login se não houver token.
    - Exibição do plano atual do usuário.

    - Seleção de planos e métodos de pagamento funciona.
    - Dados do cartão são preenchidos corretamente.
    - Validações funcionam (alertas para campos incompletos, plano já ativo).
    - Confirmação de troca de plano via window.confirm.
    - Submissão da API (axios.post) com payload correto.
    - Redirecionamento para /perfil e alertas de sucesso/erro.
- **Critérios de Aceitação:**
    - axios.get e axios.post são chamados com dados e headers corretos.
    - Alerta e window.confirm são disparados corretamente.
    - Redirecionamento ocorre.

Testes de Deleção

Teste 016 – Deleção de Conta

- **ID:** T016
- **Requisito:** RF14 - Usuários devem ser capazes de solicitar a exclusão de suas contas.
- **Descrição:** Verificar o fluxo de exclusão de conta com modal de confirmação e validações.
- **Componente Testado:** DeletarConta
- **Dados de Entrada:** Interações de usuário (clique em botão, preenchimento de campos do modal: motivo, senha, confirmação de texto).
- **Resultado Esperado:**
    - Mensagens de alerta e informações sobre a exclusão são exibidas.
    - Modal de confirmação abre e fecha corretamente.
    - Botão de "Deletar permanentemente" é habilitado/desabilitado com base nas validações.

    - Confirmação dispara console.log com motivo e fecha modal.

- **Critérios de Aceitação:**
    - Modal é interativo e exibe os campos esperados.
    - Botão de deleção responde às validações de input.
    - handleDeletarConta é chamado apenas em caso de confirmação válida.

6. Aplicação de Critérios de Qualidade (ISO/IEC 25010)

Funcionalidade: Casos de uso testados com BDD (Cucumber)
Confiabilidade: Testes automatizados e plano de testes documentado
Usabilidade: Layout responsivo e intuitivo para público-alvo diverso
Eficiência: Requisições otimizadas; arquitetura escalável
Manutenibilidade: Código modular com separação por responsabilidades
Segurança: Senhas com hash, autenticação JWT, validação de entrada
Portabilidade: Aplicação web responsiva para desktop e mobile