FECAP

**PROJETO.....**

**Requisitos da disciplina Modelagem de Software e Arquitetura de Sistemas**

São Paulo

2024

**INTEGRANTES DO PROJETO e RA’S**

|  |  |
| --- | --- |
| **Emilly Mickeli Depine da Silva** | **23025480** |
| **Fernando Jose dos Santos** | **23025299** |
| **Sergio Ricardo Pedote Junior** | **23747441** |
| **Kaua Silva Rocha** | **23025404** |

Sumário

[1 INTRODUÇÃO 3](#_Toc164097807)

[2. DOCUMENTO DE ABERTURA DO PROJETOS 3](#_Toc164097808)

[3. REQUISITOS DE SISTEMA 4](#_Toc164097809)

[3.1 REQUISITOS FUNCIONAIS DE SOFTWARE 4](#_Toc164097810)

[3.2 REQUISITOS NÃO FUNCIONAIS DE SOFTWARE 5](#_Toc164097811)

[4. CASOS DE USO 5](#_Toc164097812)

[5. ARQUITETURA DO SISTEMA 6](#_Toc164097813)

[6. REFERÊNCIAS BIBLIOGRÁFICAS 6](#_Toc164097814)

# 1 INTRODUÇÃO

Necessidade para o Sistema

Nos dias de hoje, as ONGs desempenham um papel crucial na solução de problemas sociais, ambientais e humanitários. No entanto, muitas dessas organizações enfrentam desafios significativos na captação e gerenciamento de voluntários. Do outro lado, muitas pessoas desejam contribuir voluntariamente com seu tempo e habilidades, mas não sabem onde encontrar oportunidades que se alinhem aos seus interesses e disponibilidade.

O sistema Conexão Solidária foi concebido para preencher essa lacuna, criando uma plataforma centralizada onde voluntários podem encontrar oportunidades de atuação em diversas ONGs, e estas, por sua vez, podem gerenciar e recrutar voluntários de forma eficiente e organizada.

Funções do Sistema

O Conexão Solidária oferece as seguintes funções principais:

Cadastro e Autenticação de Usuários: Permitir que tanto voluntários quanto representantes de ONGs se registrem e acessem o sistema de forma segura.

Gerenciamento de Perfis: Permitir que os usuários atualizem suas informações pessoais, habilidades, áreas de interesse e disponibilidade.

Publicação e Visualização de Projetos: Permitir que ONGs publiquem detalhes de projetos e que voluntários busquem e visualizem essas oportunidades.

Candidatura a Projetos: Facilitar o processo de candidatura de voluntários a projetos de seu interesse e permitir que ONGs gerenciem essas candidaturas.

Comunicação e Notificações: Facilitar a comunicação entre voluntários e ONGs e enviar notificações relevantes aos usuários.

Relatórios e Análises: Oferecer ferramentas para que as ONGs acompanhem o engajamento dos voluntários e avaliem o impacto de seus projetos.

Integração com Outros Sistemas

O Conexão Solidária foi projetado para se integrar de forma fluida com outros sistemas e serviços, garantindo uma experiência completa e eficiente para seus usuários. Algumas das integrações previstas incluem:

Sistemas de Email: Para envio de notificações e confirmações de registro.

Redes Sociais: Para facilitar o compartilhamento de oportunidades de voluntariado e aumentar a visibilidade das ONGs.

Plataformas de Pagamento: Caso as ONGs precisem arrecadar fundos para seus projetos ou pagar taxas de inscrição para eventos.

APIs de Geolocalização: Para ajudar voluntários a encontrarem oportunidades próximas de sua localização.

Alinhamento com Objetivos Estratégicos

O desenvolvimento do Conexão Solidária está alinhado com os objetivos globais de negócio e estratégicos da organização que encomendou o software, promovendo a responsabilidade social e o engajamento comunitário. Os principais objetivos atendidos incluem:

Aumento do Impacto Social: Ao facilitar o recrutamento de voluntários, o sistema ajuda as ONGs a realizarem mais projetos e a terem um maior impacto nas comunidades que atendem.

Eficiência Operacional: Automatizando e centralizando o processo de gerenciamento de voluntários, o sistema permite que as ONGs se concentrem mais em suas atividades principais e menos em tarefas administrativas.

Engajamento da Comunidade: Oferecendo uma plataforma onde indivíduos podem facilmente encontrar e participar de atividades voluntárias, o sistema fortalece o senso de comunidade e promove a participação cívica.

Transparência e Confiabilidade: Ao fornecer uma interface clara e relatórios detalhados, o sistema aumenta a transparência das atividades das ONGs e a confiança dos voluntários no processo.

Em suma, o Conexão Solidária não apenas facilita a conexão entre voluntários e ONGs, mas também promove uma cultura de solidariedade e apoio mútuo, contribuindo para um impacto positivo duradouro na sociedade.

A necessidade do sistema é a carência da divulgação de ONG’s no momento atual, a função principal é a divulgação de diversas ONG’s e através do cadastro do usuário disponibilizar uma lista de voluntários disponíveis, a objetivo de divulgar para o publico atende o objetivo filantropo do site.

# 2. DOCUMENTO DE ABERTURA DO PROJETOS

**Prefácio**

Leitores do Documento

Este documento é destinado a diversos públicos, incluindo:

Desenvolvedores de Software: Engenheiros de software responsáveis pela implementação e manutenção do sistema Conexão Solidária. Eles encontrarão especificações detalhadas dos requisitos funcionais e não funcionais, além de descrições técnicas da arquitetura e dos componentes do sistema.

Gerentes de Projeto: Profissionais que coordenam e gerenciam o desenvolvimento do sistema, garantindo que os prazos sejam cumpridos e que os requisitos sejam atendidos. Eles utilizarão este documento para acompanhar o progresso e alinhar as atividades da equipe.

Testadores de Software: Especialistas responsáveis por validar e verificar a funcionalidade do sistema. Este documento servirá como base para a criação de casos de teste e para garantir que o sistema atenda aos requisitos definidos.

Representantes de ONGs: Usuários finais que usarão o sistema para gerenciar suas organizações e recrutar voluntários. Embora possam não precisar de todos os detalhes técnicos, eles poderão consultar as seções de funcionalidades para entender melhor como o sistema irá atender às suas necessidades.

Voluntários: Usuários que se cadastrarão no sistema para encontrar oportunidades de voluntariado. Eles poderão se beneficiar das descrições das funcionalidades relacionadas ao gerenciamento de perfil e candidatura a projetos.

Stakeholders: Inclui patrocinadores, investidores e outros interessados que buscam uma visão geral do sistema, seu propósito e como ele atende aos objetivos estratégicos da organização.

Histórico de Versões

| Versão | Data | Autor | Justificativa | Resumo das Mudanças |
| --- | --- | --- | --- | --- |
| 1.0 | 01/04/2024 | João da Silva | Criação do documento inicial | Versão inicial do documento, incluindo introdução, definição de requisitos, arquitetura e prefácio. |
| 1.1 | 15/04/2024 | Maria Oliveira | Revisão e inclusão de requisitos adicionais | Adição de novos requisitos funcionais e não funcionais, ajustes na arquitetura do sistema. |
| 1.2 | 01/05/2024 | Carlos Ferreira | Ajustes de usabilidade e segurança | Melhorias nas descrições de usabilidade, implementação de novos requisitos de segurança e ajustes na seção de casos de uso. |
| 1.3 | 15/05/2024 | Ana Paula Costa | Revisão final antes do início do desenvolvimento | Atualização das especificações com base no feedback dos stakeholders, preparação para a fase de desenvolvimento. |

Justificativa para a Criação de Novas Versões

A criação de novas versões deste documento é justificada pelas seguintes razões:

Evolução dos Requisitos: À medida que novos requisitos são identificados ou os existentes são modificados, é essencial atualizar o documento para refletir essas mudanças.

Melhoria Contínua: Com base no feedback dos usuários e stakeholders, novas versões são criadas para melhorar a clareza, usabilidade e funcionalidade do sistema.

Correções de Erros: Correções de erros e omissões nas versões anteriores justificam a emissão de uma nova versão para garantir a precisão e a completude do documento.

Preparação para Novas Fases: Antes de iniciar novas fases do desenvolvimento, como a implementação ou o teste, uma revisão e atualização do documento garantem que todas as partes estejam alinhadas e preparadas.

Cada nova versão do documento incorpora as mudanças necessárias para garantir que o sistema Conexão Solidária atenda às necessidades de seus usuários e cumpra seus objetivos estratégicos de maneira eficaz e eficiente.

**Glossário**

Autenticação

Processo de verificação da identidade de um usuário que tenta acessar um sistema, geralmente através da verificação de um nome de usuário e senha.

API (Application Programming Interface)

Conjunto de definições e protocolos usado para desenvolver e integrar software de aplicativos, permitindo que diferentes sistemas se comuniquem entre si.

CRUD (Create, Read, Update, Delete)

Acrônimo para as quatro operações básicas realizadas em bases de dados ou sistemas de gerenciamento de conteúdo: criar, ler, atualizar e excluir dados.

Frontend

Parte do sistema que interage diretamente com o usuário, geralmente representada pela interface gráfica do usuário (GUI). No contexto deste documento, refere-se à interface construída usando tecnologias como React.js.

Backend

Parte do sistema responsável pela lógica de negócios, processamento de dados e interação com o banco de dados. Geralmente é executado em servidores e interage com o frontend através de APIs.

Banco de Dados

Sistema organizado para armazenamento, gerenciamento e recuperação de dados. No contexto deste documento, refere-se ao uso de sistemas de gerenciamento de banco de dados relacionais como MySQL ou PostgreSQL.

Escalabilidade

Capacidade de um sistema de lidar com um aumento no volume de trabalho ou de ser ampliado para acomodar esse crescimento.

Disponibilidade

Medida do tempo durante o qual um sistema está operacional e acessível para uso, geralmente expressa como uma porcentagem de tempo de atividade em um período especificado.

Usabilidade

Facilidade com que um usuário pode utilizar um sistema ou aplicativo para realizar suas tarefas. Envolve aspectos como eficiência, eficácia e satisfação do usuário.

Segurança

Conjunto de medidas e práticas adotadas para proteger os dados e recursos do sistema contra acesso não autorizado, uso indevido ou ataques.

Requisitos Funcionais

Especificações das funcionalidades que um sistema deve possuir para atender às necessidades dos usuários e realizar as tarefas desejadas.

Requisitos Não Funcionais

Especificações de critérios que julgam a operação de um sistema, como desempenho, usabilidade, confiabilidade etc., mas não se referem diretamente às funcionalidades específicas.

ONG (Organização Não Governamental)

Entidade privada sem fins lucrativos que desenvolve atividades de interesse público, frequentemente em áreas como assistência social, educação, saúde, e meio ambiente.

Voluntário

Indivíduo que se oferece espontaneamente para prestar serviços sem remuneração, geralmente em projetos sociais ou humanitários.

JWT (JSON Web Token)

Padrão aberto (RFC 7519) que define uma maneira compacta e segura de transmitir informações entre partes como um objeto JSON. É frequentemente usado para autenticação e troca de informações seguras na web.

Middleware

Software que atua como intermediário entre diferentes sistemas ou componentes de software, facilitando a comunicação e a gestão de dados entre eles.

LGPD (Lei Geral de Proteção de Dados)

Legislação brasileira que regula o tratamento de dados pessoais e estabelece direitos para os titulares de dados e obrigações para as empresas e organizações que processam esses dados.

React.js

Biblioteca JavaScript de código aberto para a construção de interfaces de usuário, mantida pelo Facebook e por uma comunidade de desenvolvedores individuais e empresas.

Node.js

Ambiente de execução JavaScript server-side, construído sobre o motor V8 do Google Chrome, permitindo a construção de aplicativos escaláveis e de alto desempenho.

HTTPS (HyperText Transfer Protocol Secure)

Versão segura do HTTP, o protocolo de comunicação usado na web. Utiliza criptografia SSL/TLS para proteger a integridade e a confidencialidade dos dados transmitidos entre o navegador e o servidor.

AWS (Amazon Web Services)

Plataforma de serviços em nuvem oferecida pela Amazon, que fornece infraestrutura como serviço (IaaS), plataforma como serviço (PaaS) e software como serviço (SaaS).

W3C WCAG (World Wide Web Consortium Web Content Accessibility Guidelines)

Diretrizes internacionais para tornar o conteúdo web mais acessível para pessoas com deficiência. A versão 2.1 das diretrizes inclui recomendações para melhorar a acessibilidade.

Scrum

Framework ágil para o desenvolvimento de produtos complexos, utilizado para organizar e gerenciar o trabalho de equipes através de iterações chamadas sprints.

Este glossário fornece definições claras e acessíveis para os termos técnicos usados no documento, facilitando a compreensão para leitores de diferentes níveis de conhecimento e experiência.

**Definição de requisitos de usuário**

O sistema Conexão Solidária é projetado para atender dois tipos principais de usuários: Voluntários e Representantes de ONGs. Abaixo são descritos os serviços fornecidos para cada tipo de usuário, os requisitos não funcionais do sistema e as normas de produto e processos que devem ser seguidos.

Tipos de Usuários

Usuário Voluntário

Descrição: Indivíduos que desejam se cadastrar como voluntários para participar de atividades e projetos sociais organizados pelas ONGs.

Usuário Representante de ONG

Descrição: Representantes das ONGs que utilizam o sistema para gerenciar suas organizações e recrutar voluntários para seus projetos.

Serviços Fornecidos ao Usuário

Para Usuários Voluntários

Cadastro e Autenticação

Descrição: Permitir que voluntários se registrem e façam login no sistema.

Funcionalidades:

Voluntários devem poder criar uma conta fornecendo nome, email e senha.

O sistema deve enviar um email de confirmação após o registro.

Voluntários devem poder fazer login com email e senha.

Recuperação de senha deve estar disponível.

Gerenciamento de Perfil

Descrição: Permitir que voluntários visualizem e editem suas informações pessoais e disponibilidades.

Funcionalidades:

Voluntários devem poder atualizar informações pessoais, como nome e email.

Voluntários devem poder especificar suas habilidades e áreas de interesse.

Voluntários devem poder definir sua disponibilidade para atividades.

Visualização de Projetos

Descrição: Permitir que voluntários vejam uma lista de projetos disponíveis e seus detalhes.

Funcionalidades:

Voluntários devem poder buscar e filtrar projetos por categoria, localização e data.

Voluntários devem poder ver detalhes de cada projeto, incluindo descrição, datas e requisitos.

Candidatura a Projetos

Descrição: Permitir que voluntários se candidatem a projetos de seu interesse.

Funcionalidades:

Voluntários devem poder se candidatar a projetos diretamente pela interface do sistema.

O sistema deve registrar a candidatura e informar o status (pendente, aprovado, rejeitado).

Para Usuários Representantes de ONGs

Cadastro e Autenticação

Descrição: Permitir que representantes de ONGs se registrem e façam login no sistema.

Funcionalidades:

Representantes devem poder criar uma conta fornecendo nome, email e senha.

O sistema deve enviar um email de confirmação após o registro.

Representantes devem poder fazer login com email e senha.

Recuperação de senha deve estar disponível.

Gerenciamento de ONGs

Descrição: Permitir que representantes registrem e gerenciem informações sobre suas ONGs.

Funcionalidades:

Representantes devem poder registrar uma ONG fornecendo nome, descrição, área de atuação e informações de contato.

Representantes devem poder editar informações da ONG.

O sistema deve permitir a visualização pública das ONGs registradas.

Criação e Gestão de Projetos

Descrição: Permitir que representantes criem e gerenciem projetos sociais.

Funcionalidades:

Representantes devem poder criar projetos fornecendo nome, descrição, datas e localização.

Representantes devem poder editar e excluir projetos existentes.

O sistema deve permitir a visualização pública dos projetos disponíveis.

Gestão de Voluntários

Descrição: Permitir que representantes visualizem e gerenciem as candidaturas de voluntários.

Funcionalidades:

Representantes devem poder acessar uma lista de voluntários disponíveis.

Representantes devem poder filtrar voluntários por habilidades, disponibilidade e localização.

Representantes devem poder aprovar ou rejeitar candidaturas de voluntários.

Requisitos Não Funcionais

Desempenho

Descrição: O sistema deve responder rapidamente às ações do usuário.

Requisitos:

Tempo de resposta máximo de 2 segundos para ações de usuário em condições normais.

Capacidade de suportar até 1000 usuários simultâneos sem degradação de desempenho.

Segurança

Descrição: O sistema deve proteger os dados dos usuários contra acesso não autorizado.

Requisitos:

Utilização de HTTPS para todas as comunicações.

Armazenamento de senhas de forma segura utilizando hashing.

Implementação de autenticação e autorização para acesso aos dados sensíveis.

Usabilidade

Descrição: O sistema deve ser fácil de usar e acessível para todos os tipos de usuários.

Requisitos:

Interface intuitiva com navegação clara.

Compatibilidade com navegadores modernos e dispositivos móveis.

Conformidade com as normas de acessibilidade W3C WCAG 2.1.

Escalabilidade

Descrição: O sistema deve ser capaz de crescer conforme a demanda aumenta.

Requisitos:

Arquitetura modular que permita a adição de novos componentes sem grandes alterações.

Capacidade de escalar horizontalmente adicionando novos servidores conforme necessário.

Disponibilidade

Descrição: O sistema deve estar disponível para uso quase o tempo todo.

Requisitos:

Disponibilidade mínima de 99.9%.

Implementação de estratégias de redundância e recuperação de desastres.

Manutenibilidade

Descrição: O sistema deve ser fácil de manter e atualizar.

Requisitos:

Código bem documentado e modular.

Utilização de boas práticas de desenvolvimento e padrões de codificação.

Implementação de testes automatizados para garantir a qualidade do código.

Normas de Produto e Processos

Normas de Segurança: Conformidade com a LGPD (Lei Geral de Proteção de Dados) para proteção dos dados dos usuários.

Normas de Desenvolvimento: Adoção de metodologias ágeis como Scrum para o desenvolvimento e manutenção do sistema.

Normas de Acessibilidade: Conformidade com as diretrizes W3C WCAG 2.1 para garantir a acessibilidade do sistema para usuários com deficiências.

Estas definições garantem que o sistema atenda às necessidades dos diferentes tipos de usuários e mantenha um alto padrão de qualidade, segurança e desempenho.

**Arquitetura do sistema**

Frontend (Cliente)

Tecnologia Utilizada: React

Funções:

Interface de usuário: Responsável pela interação com o usuário, exibição de dados e coleta de entradas.

Comunicação com o Backend: Realiza requisições HTTP para obter e enviar dados ao servidor.

Componentes Reutilizáveis:

Componentes de Interface: Botões, formulários, tabelas e outros elementos de interface que podem ser reutilizados em diferentes partes da aplicação.

Serviços de API: Módulos que encapsulam chamadas às APIs do backend, facilitando a manutenção e reutilização.

Backend (Servidor)

Tecnologia Utilizada: Node.js com Express

Funções:

Processamento de lógica de negócios: Gestão de regras de negócio e validação de dados.

Gestão de dados: Interação com o banco de dados para criar, ler, atualizar e deletar dados (operações CRUD).

Provisão de APIs RESTful: Disponibiliza endpoints para que o frontend possa realizar operações de dados.

Componentes Reutilizáveis:

Middlewares: Funções reutilizáveis que podem ser aplicadas a múltiplas rotas para manipulação de requisições, autenticação, e gerenciamento de erros.

Serviços de Banco de Dados: Módulos que encapsulam a lógica de acesso ao banco de dados, permitindo reutilização e separação de preocupações.

Banco de Dados

Tecnologia Utilizada: MySQL

Funções:

Armazenamento de dados: Mantém dados persistentes do sistema, incluindo informações sobre usuários, organizações, eventos e outras entidades.

Gerenciamento de transações: Garante a integridade dos dados durante operações de múltiplas etapas.

Componentes Reutilizáveis:

Esquemas e Modelos: Estruturas de dados definidas que podem ser reutilizadas em diferentes partes do sistema para assegurar consistência.

Fluxo de Dados

Usuário Interage com o Frontend:

O usuário acessa a aplicação web e interage com a interface React.

A aplicação React envia requisições HTTP para o backend ao realizar operações como visualizar, adicionar ou editar dados.

Requisições do Frontend para o Backend:

O servidor Express recebe as requisições do frontend.

Middlewares processam as requisições (e.g., autenticação, validação).

Interação do Backend com o Banco de Dados:

O servidor Express utiliza serviços de banco de dados para interagir com o MySQL.

Executa operações CRUD no banco de dados conforme solicitado pelas requisições do frontend.

Respostas do Backend para o Frontend:

O servidor Express processa os dados recebidos do banco de dados e envia respostas apropriadas de volta ao frontend.

O frontend recebe os dados e atualiza a interface do usuário de acordo.

Diagrama de Alto Nível da Arquitetura

plaintext

Copiar código

[Usuário] | v [Frontend (React)] | v [Backend (Node.js + Express)] | v [Banco de Dados (MySQL)]

Conclusão

Esta arquitetura modular e escalável permite que o Projeto Conexão Solidária seja facilmente mantido e expandido. A separação clara entre frontend, backend e banco de dados, combinada com o uso de componentes reutilizáveis, facilita a colaboração entre desenvolvedores e a implementação de novas funcionalidades, garantindo a eficiência e a sustentabilidade do sistema.

**Especificação de requisitos do sistema**

Requisitos Funcionais

1. Gestão de Usuários

RF1.1: O sistema deve permitir o cadastro de novos usuários, capturando informações como nome, e-mail, senha, e perfil (voluntário, ONG, administrador).

RF1.2: O sistema deve permitir o login de usuários utilizando e-mail e senha.

RF1.3: O sistema deve permitir a recuperação de senha por meio do e-mail cadastrado.

RF1.4: O sistema deve permitir a edição de informações do perfil do usuário.

RF1.5: O sistema deve permitir a exclusão de contas de usuário.

2. Gestão de ONGs

RF2.1: O sistema deve permitir o cadastro de ONGs, incluindo nome, descrição, área de atuação, endereço, contato e dados de representantes.

RF2.2: O sistema deve permitir a edição das informações cadastradas das ONGs.

RF2.3: O sistema deve permitir a exclusão de ONGs.

RF2.4: O sistema deve permitir que ONGs publiquem projetos e campanhas, detalhando objetivos, necessidades, e formas de apoio.

3. Gestão de Projetos e Campanhas

RF3.1: O sistema deve permitir que ONGs criem, editem e excluam projetos e campanhas.

RF3.2: O sistema deve permitir que usuários visualizem uma lista de projetos e campanhas, podendo filtrar por área de atuação, localização, e data.

RF3.3: O sistema deve permitir que usuários se candidatem para participar como voluntários em projetos e campanhas.

4. Comunicação e Notificações

RF4.1: O sistema deve enviar notificações por e-mail para confirmar cadastro de usuário, recuperação de senha, e outras ações relevantes.

RF4.2: O sistema deve permitir a comunicação entre usuários e ONGs através de mensagens internas.

RF4.3: O sistema deve enviar notificações de novas campanhas e oportunidades de voluntariado para os usuários cadastrados.

5. Gestão de Eventos

RF5.1: O sistema deve permitir que ONGs criem, editem e excluam eventos.

RF5.2: O sistema deve permitir que usuários visualizem e se inscrevam em eventos.

Requisitos Não Funcionais

1. Desempenho

RNF1.1: O sistema deve ser capaz de suportar até 1000 usuários simultâneos sem degradação perceptível de desempenho.

RNF1.2: As respostas às requisições do usuário devem ser fornecidas em menos de 2 segundos na maioria dos casos.

2. Segurança

RNF2.1: Todas as senhas de usuários devem ser armazenadas de forma segura utilizando hashing e salting.

RNF2.2: O sistema deve utilizar HTTPS para todas as comunicações entre o frontend e o backend.

RNF2.3: O sistema deve ter controles de acesso baseados em papéis para garantir que apenas usuários autorizados possam executar determinadas ações.

3. Usabilidade

RNF3.1: A interface do usuário deve ser intuitiva e fácil de navegar, seguindo princípios de design de usabilidade.

RNF3.2: O sistema deve ser acessível, seguindo as diretrizes WCAG (Web Content Accessibility Guidelines) para garantir que pessoas com deficiências possam utilizá-lo.

4. Manutenibilidade

RNF4.1: O código do sistema deve ser modular e bem documentado para facilitar a manutenção e a adição de novas funcionalidades.

RNF4.2: O sistema deve utilizar práticas de desenvolvimento ágil, permitindo atualizações frequentes e integradas de forma contínua.

Interfaces com Outros Sistemas

1. Sistema de Autenticação por Terceiros

IF1.1: O sistema deve permitir a autenticação de usuários utilizando contas de terceiros, como Google ou Facebook, através do OAuth 2.0.

2. Serviços de Pagamento

IF2.1: O sistema deve integrar-se com serviços de pagamento (como PayPal, Stripe) para permitir doações online para as ONGs.

3. API de Geolocalização

IF3.1: O sistema deve integrar-se com APIs de geolocalização (como Google Maps) para permitir que usuários encontrem ONGs, projetos e eventos próximos de sua localização.

Detalhes Adicionais dos Requisitos Não Funcionais

1. Escalabilidade

RNF1.3: O sistema deve ser escalável horizontalmente, permitindo adicionar mais servidores para lidar com o aumento da carga de usuários e dados.

2. Backup e Recuperação

RNF2.4: O sistema deve realizar backups regulares dos dados e permitir a recuperação dos mesmos em caso de falha ou perda de dados.

3. Compatibilidade

RNF3.3: O sistema deve ser compatível com os navegadores mais populares (Chrome, Firefox, Safari, Edge) e com dispositivos móveis (iOS e Android).

Conclusão

Esta especificação de requisitos fornece uma base sólida para o desenvolvimento do Projeto Conexão Solidária, assegurando que tanto os requisitos funcionais quanto os não funcionais sejam atendidos. A arquitetura do sistema, combinada com as interfaces definidas, permite uma abordagem modular e escalável, facilitando a integração e a expansão futuras.

**Modelos do sistema**

Modelo de Componentes do Sistema

O modelo de componentes do sistema ilustra a interação entre os principais componentes da arquitetura do Projeto Conexão Solidária, mostrando como o frontend, o backend e o banco de dados se comunicam e interagem.

[Usuário]

|

v

[Frontend (React)]

|

| (HTTP Requests)

v

[Backend (Node.js + Express)]

| |

| | (SQL Queries)

| v

| [Banco de Dados (MySQL)]

|

| (API Integrations)

v

[Serviços de Terceiros]

Modelo de Fluxo de Dados

Este modelo mostra o fluxo de dados entre diferentes componentes e módulos do sistema, destacando as principais operações realizadas.

[Usuário] <-- interação --> [Interface de Usuário (React)]

|

v

[Controle de Autenticação (React)] -- requisição de login --> [API de Autenticação (Express)]

|

v

[Controle de Usuário (React)] -- requisição de dados --> [API de Usuário (Express)]

|

v

[Interface de Gerenciamento de ONGs (React)] -- requisição de dados --> [API de ONGs (Express)]

|

v

[Controle de Projetos e Campanhas (React)] -- requisição de dados --> [API de Projetos e Campanhas (Express)]

|

v

[Comunicação e Notificações (React)] -- requisição de dados --> [API de Notificações (Express)]

|

v

[Banco de Dados (MySQL)]

Modelo de Casos de Uso

Este modelo mostra os principais casos de uso do sistema, representando as interações entre os atores (usuários, ONGs, administradores) e o sistema.

[Usuário]

|

| (Cadastrar-se, Logar, Editar Perfil)

v

[Gerenciamento de Usuários (React)]

|

| (Visualizar Projetos, Candidatar-se)

v

[Gerenciamento de Projetos (React)]

|

| (Visualizar ONGs, Contatar ONGs)

v

[Gerenciamento de ONGs (React)]

|

| (Receber Notificações)

v

[Sistema de Notificações (React)]

|

v

[API de Notificações (Express)]

Modelo Entidade-Relacionamento (ER)

Este modelo mostra o esquema do banco de dados, destacando as principais entidades e seus relacionamentos.

Entidade: Usuário

Atributos: ID, Nome, Email, Senha, Perfil

Entidade: ONG

Atributos: ID, Nome, Descrição, Área de Atuação, Endereço, Contato, RepresentanteID

Entidade: Projeto

Atributos: ID, Nome, Descrição, ONGID, DataInicio, DataFim, Localização

Entidade: Candidatura

Atributos: ID, UsuarioID, ProjetoID, DataCandidatura, Status

Relacionamentos:

Usuário 1 - N Candidatura

ONG 1 - N Projeto

Projeto 1 - N Candidatura

Modelo de Sequência

Este modelo mostra a sequência de interações entre o usuário, o frontend, o backend e o banco de dados durante uma operação típica, como a candidatura a um projeto.

Usuário Interface React API Express MySQL

| | | |

1. Clicar "Candidatar-se" -----------------> |

| | | |

2. Enviar Dados --------------> | |

| | | |

3. Verificar Autenticação ----------------> |

| | | |

4. Validar Dados ------------------------->| |

| | | |

5. Inserir Candidatura --------------------------> |

| | | |

6. Confirmação <--------------------------------------- |

| | | |

7. Atualizar UI <--------------------- |

Conclusão

Os modelos acima fornecem uma visão abrangente da arquitetura do sistema do Projeto Conexão Solidária, destacando a distribuição de funções entre os componentes do sistema, o fluxo de dados, os casos de uso principais e os relacionamentos entre entidades no banco de dados. Estes diagramas e descrições servem como referência para o desenvolvimento e a manutenção do sistema, assegurando uma compreensão clara e compartilhada entre todos os membros da equipe.

**Evolução do sistema**

Pressupostos Fundamentais

Tecnologia Atual: O sistema é baseado em tecnologias web modernas, incluindo React para o frontend, Node.js com Express para o backend e MySQL para o banco de dados.

Arquitetura Modular: A arquitetura do sistema é modular, permitindo a substituição ou atualização de componentes individuais sem afetar o restante do sistema.

Escalabilidade: O sistema é projetado para ser escalável, tanto horizontal quanto verticalmente, para suportar um número crescente de usuários e dados.

Segurança: A segurança dos dados e das transações é um aspecto crítico, com autenticação segura e comunicações criptografadas.

Usabilidade: O sistema é projetado para ser intuitivo e acessível, atendendo às diretrizes de usabilidade e acessibilidade.

Mudanças Previstas

1. Evolução de Hardware

Aumento de Capacidade de Servidores: À medida que o número de usuários cresce, será necessário aumentar a capacidade dos servidores ou adicionar mais servidores para garantir um desempenho consistente.

Adoção de Serviços em Nuvem: A migração para uma infraestrutura de nuvem (como AWS, Azure, ou Google Cloud) pode ser necessária para aproveitar a escalabilidade, a flexibilidade e a disponibilidade oferecidas pelos serviços em nuvem.

2. Mudanças nas Necessidades dos Usuários

Novas Funcionalidades: Com base no feedback dos usuários, novas funcionalidades podem ser introduzidas, como suporte para doações recorrentes, relatórios detalhados de impacto, e integração com novas plataformas de mídia social.

Personalização: O sistema pode evoluir para oferecer experiências mais personalizadas, utilizando técnicas de aprendizado de máquina para recomendar projetos e eventos relevantes aos usuários com base em suas preferências e histórico de interações.

Aprimoramento de Usabilidade e Acessibilidade: Melhorias contínuas na interface do usuário e na acessibilidade serão necessárias para garantir que o sistema permaneça fácil de usar para todos os usuários, incluindo aqueles com deficiências.

3. Integração com Novos Sistemas

Novas APIs e Serviços Externos: A integração com novos serviços externos pode ser necessária à medida que novas tecnologias e plataformas se tornam disponíveis. Isso pode incluir novos métodos de pagamento, serviços de geolocalização, ou plataformas de comunicação.

Automação e IA: A adoção de automação e inteligência artificial pode melhorar a eficiência e a eficácia do sistema. Exemplos incluem chatbots para suporte ao usuário, análise preditiva para identificar tendências de voluntariado, e sistemas de recomendação para conectar usuários a projetos relevantes.

4. Regulamentações e Políticas

Conformidade com Novas Leis e Regulamentos: O sistema deve ser adaptável para se manter em conformidade com novas leis e regulamentos de proteção de dados e privacidade, como o GDPR na Europa ou a LGPD no Brasil.

Segurança e Proteção de Dados: Melhorias contínuas nas práticas de segurança serão necessárias para proteger contra ameaças emergentes e garantir a proteção dos dados dos usuários.

Diretrizes para Evolução

Desenvolvimento Ágil: Adotar metodologias ágeis para permitir uma resposta rápida às mudanças nas necessidades dos usuários e nas tecnologias.

Documentação Clara e Completa: Manter documentação clara e completa para todos os componentes do sistema, facilitando futuras manutenções e evoluções.

Modularidade e Flexibilidade: Continuar a projetar o sistema com modularidade e flexibilidade em mente, permitindo a fácil substituição ou atualização de componentes.

Monitoramento e Feedback Contínuo: Implementar sistemas de monitoramento e coleta de feedback para identificar áreas de melhoria e novas necessidades dos usuários de forma proativa.

Investimento em Segurança: Continuar investindo em medidas de segurança avançadas para proteger contra ameaças cibernéticas e garantir a conformidade com regulamentações de proteção de dados.

Conclusão

A evolução do sistema do Projeto Conexão Solidária baseia-se em pressupostos fundamentais de tecnologia moderna, modularidade, escalabilidade, segurança e usabilidade. As mudanças previstas abrangem a evolução do hardware, as necessidades dos usuários, a integração com novos sistemas e a conformidade com regulamentações. Seguindo diretrizes claras para a evolução, o sistema pode crescer e se adaptar às demandas futuras, garantindo sua relevância e eficácia contínuas.

**Apêndices**

A. Requisitos de Hardware

Os requisitos de hardware abaixo descrevem as configurações mínimas e ideais para garantir o bom funcionamento do sistema do Projeto Conexão Solidária.

A.1. Configurações de Servidor

Servidor de Desenvolvimento

Processador: Quad-core 2.5 GHz ou superior

Memória RAM: 8 GB

Armazenamento: 256 GB SSD

Sistema Operacional: Windows 10, macOS 10.15+, ou Ubuntu 18.04+

Conectividade: Conexão de Internet de alta velocidade

Servidor de Produção

Processador: Octa-core 3.0 GHz ou superior

Memória RAM: 16 GB ou superior

Armazenamento: 1 TB SSD ou superior, com RAID para redundância

Sistema Operacional: Linux (Ubuntu 20.04 LTS recomendado)

Conectividade: Conexão de Internet de alta velocidade com redundância

Outros: Backup automático, balanceamento de carga, firewall configurável

A.2. Configurações de Cliente

Computador Desktop/Laptop

Processador: Dual-core 2.0 GHz ou superior

Memória RAM: 4 GB

Armazenamento: 50 GB HDD ou SSD

Sistema Operacional: Windows 10, macOS 10.14+, ou Linux (distribuição moderna)

Navegador: Chrome, Firefox, Safari, ou Edge (últimas versões)

Dispositivo Móvel

Sistema Operacional: Android 8.0+ ou iOS 12.0+

Armazenamento: 50 MB de espaço livre para armazenamento de cache e dados temporários

Navegador: Chrome, Safari, ou navegador padrão do dispositivo

# 3. REQUISITOS DE SISTEMA

## 3.1 REQUISITOS FUNCIONAIS DE SOFTWARE

Necessários 6 requisitos

|  |  |
| --- | --- |
| **RFS01** | |
| Função | Permitir que novos usuários se cadastrem no sistema. |
| **Descrição** | O sistema deve permitir que novos usuários se registrem fornecendo suas informações básicas. Isso inclui validação de email e criação de senha segura. |
| Entradas | Nome, Email, Senha. |
| Fonte | Interface de Cadastro de Usuário. |
| **Saídas** | Confirmação de cadastro, Mensagens de erro (se aplicável). |
| **Ação** | O sistema valida as entradas, armazena os dados do usuário no banco de dados e envia um email de confirmação. |
|  |  |
| **RFS02** | |
| **Função** | Autenticar usuários registrados para acessar o sistema. |
| **Descrição** | O sistema deve permitir que usuários registrados façam login fornecendo email e senha. |
| **Entradas** | Email, Senha. |
| **Fonte** | Interface de Login. |
| **Saídas** | Token de autenticação, Mensagens de erro (se aplicável). |
| **Ação** | O sistema verifica as credenciais, gera um token de autenticação para o usuário e permite o acesso ao sistema. |

**RFS03: Gerenciamento de ONGs**

Função: Permitir que representantes de ONGs registrem e gerenciem suas organizações no sistema.

Descrição: O sistema deve permitir que representantes de ONGs registrem suas organizações, incluindo informações detalhadas como nome, descrição, área de atuação e contato.

Entradas: Nome da ONG, Descrição, Área de Atuação, Endereço, Contato.

Fonte: Interface de Gerenciamento de ONGs.

Saídas: Confirmação de cadastro, Mensagens de erro (se aplicável).

Ação: O sistema armazena as informações da ONG no banco de dados e associa a ONG ao representante que a registrou.

**RFS04: Criação e Gerenciamento de Projetos**

Função: Permitir que ONGs registrem e gerenciem projetos sociais.

Descrição: O sistema deve permitir que ONGs registrem novos projetos, incluindo detalhes como nome, descrição, datas de início e fim, e localização.

Entradas: Nome do Projeto, Descrição, Data de Início, Data de Fim, Localização.

Fonte: Interface de Gerenciamento de Projetos.

Saídas: Confirmação de criação do projeto, Mensagens de erro (se aplicável).

Ação: O sistema armazena os dados do projeto no banco de dados e permite a edição e exclusão de projetos.

**RFS05: Candidatura a Projetos**

Função: Permitir que usuários se candidatem a projetos sociais registrados no sistema.

Descrição: O sistema deve permitir que usuários visualizem projetos disponíveis e se candidatem a participar.

Entradas: ID do Projeto, ID do Usuário.

Fonte: Interface de Projetos.

Saídas: Confirmação de candidatura, Mensagens de erro (se aplicável).

Ação: O sistema registra a candidatura do usuário no banco de dados, atualiza o status da candidatura e notifica o usuário e a ONG responsável.

**RFS06: Sistema de Notificações**

Função: Notificar usuários sobre atualizações relevantes.

Descrição: O sistema deve enviar notificações aos usuários sobre atualizações relevantes, como novas mensagens, mudanças no status da candidatura e novos projetos.

Entradas: Tipo de Notificação, ID do Usuário, Mensagem.

Fonte: Sistema Interno.

Saídas: Notificação (email, push notification, ou mensagem no sistema).

Ação: O sistema envia a notificação ao usuário através do meio apropriado (email, push notification, ou mensagem no sistema) e registra o envio da notificação no banco de dados.

## 3.2 REQUISITOS NÃO FUNCIONAIS DE SOFTWARE

Necessários 6 requisitos

|  |  |
| --- | --- |
| **RFS01** | |
| **Função** | Garantir que o sistema responda em tempo hábil. |
| **Descrição** | sistema deve ser capaz de processar requisições de usuários em até 2 segundos em condições normais de carga. |
| **Entradas** | Solicitações de usuários. |
| **Fonte** | Interface de Usuário, APIs. |
| **Saídas** | Respostas às requisições |
| **Ação** | O sistema processa as requisições e retorna os resultados aos usuários dentro do tempo especificado. |
|  |  |
| RFS02 | |
| **Função** | Proteger os dados do sistema contra acesso não autorizado. |
| **Descrição** | O sistema deve utilizar autenticação, criptografia e outras medidas de segurança para proteger os dados dos usuários. |
| **Entradas** | Credenciais de Login, Dados Sensíveis. |
| **Fonte** | Interface de Login, Banco de Dados. |
| **Saídas** | Token de Autenticação, Dados Criptografados. |
| **Ação** | O sistema verifica credenciais, criptografa dados sensíveis e garante que apenas usuários autenticados possam acessar recursos protegidos. |

**RNF03: Escalabilidade**

Função: Permitir que o sistema cresça conforme necessário.

Descrição: O sistema deve ser capaz de escalar horizontalmente para suportar um aumento no número de usuários e volume de dados.

Entradas: Aumento no tráfego e dados.

Fonte: Infraestrutura de TI.

Saídas: Desempenho Consistente, Capacidade Adicional.

Ação: O sistema deve ser capaz de adicionar novos servidores ou recursos conforme necessário para manter o desempenho.

**RNF04: Usabilidade**

Função: Garantir que o sistema seja fácil de usar para todos os tipos de usuários.

Descrição: A interface do sistema deve ser intuitiva e acessível, atendendo aos padrões de usabilidade e acessibilidade.

Entradas: Interações do Usuário.

Fonte: Interface de Usuário.

Saídas: Experiência de Usuário Positiva.

Ação: O sistema deve proporcionar uma navegação intuitiva, com mensagens de ajuda e validações claras, garantindo acessibilidade para usuários com deficiências.

**RNF05: Disponibilidade**

Função: Garantir que o sistema esteja disponível para uso em tempo integral.

Descrição: O sistema deve ter uma disponibilidade mínima de 99.9%, evitando interrupções no serviço.

Entradas: Solicitações de Serviço.

Fonte: Infraestrutura de TI.

Saídas: Serviço Disponível.

Ação: O sistema deve ser implementado com redundâncias e planos de recuperação de desastres para minimizar o tempo de inatividade.

**RNF06: Manutenibilidade**

Função: Facilitar a manutenção e evolução do sistema.

Descrição: O sistema deve ser modular e bem documentado para facilitar atualizações, correções de bugs e adição de novas funcionalidades.

Entradas: Solicitações de Manutenção.

Fonte: Equipe de Desenvolvimento.

Saídas: Código Atualizado, Documentação.

Ação: O sistema deve ser desenvolvido com boas práticas de engenharia de software, utilizando padrões de codificação, testes automatizados e documentação clara para cada módulo.

# 4. CASOS DE USO

Apresentar 3 casos de uso do sistema

**Caso de Uso 1: Cadastro de Usuário**

Título: Cadastro de Usuário

Descrição: Permitir que novos usuários se cadastrem no sistema fornecendo suas informações básicas e criando uma conta.

Atores:

Usuário

Pré-condições:

O usuário deve ter acesso à internet e ao site ou aplicativo do sistema.

Fluxo Principal:

O usuário acessa a página de cadastro no site ou aplicativo.

O usuário preenche o formulário de cadastro com seu nome, email e senha.

O usuário clica no botão "Cadastrar".

O sistema valida as informações fornecidas.

O sistema cria uma nova conta para o usuário e armazena as informações no banco de dados.

O sistema envia um email de confirmação para o endereço fornecido pelo usuário.

O usuário recebe uma mensagem de confirmação de cadastro bem-sucedido na tela.

Fluxos Alternativos:

FA1: O usuário fornece informações inválidas ou incompletas.

O sistema exibe mensagens de erro indicando os campos que precisam ser corrigidos.

O usuário corrige as informações e tenta novamente.

Pós-condições:

O usuário tem uma nova conta registrada no sistema e recebe um email de confirmação.

**Caso de Uso 2: Criação de Projeto por ONG**

Título: Criação de Projeto por ONG

Descrição: Permitir que representantes de ONGs registrem novos projetos sociais no sistema.

Atores:

Representante de ONG

Pré-condições:

O representante deve estar autenticado no sistema.

A ONG deve estar registrada no sistema.

Fluxo Principal:

O representante acessa a seção de projetos no painel da ONG.

O representante clica no botão "Criar Novo Projeto".

O representante preenche o formulário com o nome do projeto, descrição, datas de início e fim, e localização.

O representante clica no botão "Salvar".

O sistema valida as informações fornecidas.

O sistema cria um novo registro de projeto e armazena as informações no banco de dados.

O sistema exibe uma mensagem de confirmação da criação do projeto.

Fluxos Alternativos:

FA1: O representante fornece informações inválidas ou incompletas.

O sistema exibe mensagens de erro indicando os campos que precisam ser corrigidos.

O representante corrige as informações e tenta novamente.

Pós-condições:

O novo projeto é registrado no sistema e está disponível para visualização pelos usuários.

**Caso de Uso 3: Candidatura a Projeto**

Título: Candidatura a Projeto

Descrição: Permitir que usuários se candidatem a projetos sociais disponíveis no sistema.

Atores:

Usuário

Representante de ONG (para aprovar ou rejeitar candidaturas)

Pré-condições:

O usuário deve estar autenticado no sistema.

O usuário deve ter um perfil de voluntário.

Fluxo Principal:

O usuário navega até a lista de projetos disponíveis.

O usuário seleciona um projeto de interesse.

O usuário clica no botão "Candidatar-se".

O sistema exibe um formulário de candidatura (se aplicável).

O usuário preenche o formulário e clica em "Enviar".

O sistema registra a candidatura e armazena as informações no banco de dados.

O sistema exibe uma mensagem de confirmação de candidatura.

Fluxos Alternativos:

FA1: O usuário tenta se candidatar a um projeto já encerrado ou indisponível.

O sistema exibe uma mensagem informando que o projeto não está mais aceitando candidaturas.

Pós-condições:

A candidatura do usuário é registrada no sistema e fica pendente para aprovação pela ONG responsável.

# 5. ARQUITETURA DO SISTEMA

A arquitetura do sistema para o Projeto Conexão Solidária é projetada para ser modular, escalável e segura. A arquitetura é dividida em vários componentes que interagem entre si para fornecer as funcionalidades desejadas. Esses componentes incluem o front-end, o back-end, o banco de dados e a infraestrutura de suporte. Abaixo está uma visão geral em alto nível da arquitetura do sistema.

# 6. REFERÊNCIAS BIBLIOGRÁFICAS

SOMMERVILLE, I. **Engenharia de Software.** 11ª Edição. São Paulo: Pearson Addison-Wesley, 2017.

Fowler, M. (2002). Patterns of Enterprise Application Architecture. Addison-Wesley Professional.

Este livro fornece uma visão abrangente sobre os padrões arquiteturais usados no desenvolvimento de aplicações empresariais, incluindo a separação de preocupações e a organização de componentes em sistemas modulares.

Gamma, E., Helm, R., Johnson, R., & Vlissides, J. (1994). Design Patterns: Elements of Reusable Object-Oriented Software. Addison-Wesley Professional.

Um recurso clássico sobre padrões de design que descreve soluções comuns para problemas recorrentes no design de software orientado a objetos.

Bass, L., Clements, P., & Kazman, R. (2003). Software Architecture in Practice. Addison-Wesley Professional.

Martin, R. C. (2008). Clean Code: A Handbook of Agile Software Craftsmanship. Prentice Hall.

Um guia essencial para a escrita de código limpo e de fácil manutenção, focando em boas práticas e princípios de design que melhoram a qualidade do software.

Hunt, A., & Thomas, D. (1999). The Pragmatic Programmer: From Journeyman to Master. Addison-Wesley.

Este livro abrange uma ampla gama de tópicos importantes para desenvolvedores de software, incluindo técnicas práticas para melhorar a eficiência e a qualidade do código.

PostgreSQL Global Development Group. (2021). PostgreSQL 13 Documentation.

Documentação oficial do PostgreSQL, fornecendo informações detalhadas sobre o uso, configuração e administração do banco de dados PostgreSQL.