Fundação Educacional do Município de Assis
Instituto Municipal de Ensino Superior de Assis
Campus "José Santilli Sobrinho"

LEONARDO MANOEL BATISTA BETETTO

APRIMORAMENTO DA GESTÃO NO PROJETO CARA:
DESENVOLVIMENTO DE UM SISTEMA INTEGRADO

Assis/SP
2024

Fundação Educacional do Município de Assis
Instituto Municipal de Ensino Superior de Assis
Campus "José Santilli Sobrinho"

LEONARDO MANOEL BATISTA BETETTO

APRIMORAMENTO DA GESTÃO NO PROJETO CARA:
DESENVOLVIMENTO DE UM SISTEMA INTEGRADO

Trabalho de conclusão de curso apresentado ao Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas do Instituto Municipal de Ensino Superior de Assis – IMESA e a Fundação Educacional do Município de Assis – FEMA, como requisito parcial à obtenção do Certificado de Conclusão.

Orientando(a): Leonardo Manoel Batista Betetto
Orientador(a): Célio Desiró

Assis/SP
2024

Betetto, Leonardo Manoel Batista

B562a Aprimoramento da gestão no projeto CARA: desenvolvimento de um sistema integrado / Leonardo Manoel Batista Betetto. -- Assis, 2024. -- 50p.

Trabalho de Conclusão de Curso (Graduação em Análise e Desenvolvimento de Sistemas) -- Fundação Educacional do Município de Assis (FEMA), Instituto Municipal de Ensino Superior de Assis (IMESA), 2024.

Orientador: Prof. Esp. Célio Desiró.

1. Escolas Organização e administração. 2. Software. 3. Gerenciamento. I Desiró, Célio. II Título.

CDD 003

Elaborada por Anna Carolina Antunes de Moraes – Bibliotecária – CRB-8/10982

APRIMORAMENTO DA GESTÃO NO PROJETO CARA: DESENVOLVIMENTO DE UM SISTEMA INTEGRADO

LEONARDO MANOEL BATISTA BETETTO

Trabalho de Conclusão de Curso apresentado ao Instituto Municipal de Ensino Superior de Assis, como requisito do Curso de Graduação, avaliado pela seguinte comissão examinadora:

Orientador: Esp. Célio Desiró

Examinador: Dr. Almir Rogério Camolesi

Assis/SP
2024

DEDICATÓRIA

Dedico este trabalho aos meus pais, Roney Leonardo Betetto e Maria Ivete Batista Betetto, e à minha irmã, Barbara Batista Betetto.

AGRADECIMENTOS

Agradeço a Deus por me proporcionar força, determinação e sabedoria para superar as dificuldades e concluir mais uma etapa importante em minha vida. Sou grato também aos meus pais e familiares, que estiveram ao meu lado durante toda a trajetória, incentivando e apoiando nos momentos difíceis.

Gostaria de expressar minha gratidão ao meu orientador, Célio Desiró, pelo auxílio em todo o processo. Agradeço também aos professores Luiz Carlos Begosso, Diomara Martins Reigato Barros e Alex Sandro Poletto pelo apoio na documentação e desenvolvimento do banco de dados.Por fim, agradeço aos meus colegas de classe, que sempre me incentivaram e estiveram ao meu lado, em especial Gustavo Alexandre Roldam e Erick Sikina Duarte, que em diversos momentos tiraram dúvidas e ajudaram em situações difíceis.

RESUMO

O avanço da Tecnologia da Informação (TI) tem proporcionado oportunidades significativas para otimizar processos em diversos setores, incluindo o educacional e projetos sociais. A necessidade de atender demandas e cumprir prazos curtos tem levado empresas e organizações a adotarem soluções tecnológicas. No contexto educacional, a TI tornou-se aliada na gestão de alunos, matrículas, cursos e colaboradores. A gestão escolar envolve muitos processos e informações ligadas diretamente à eficiência e rapidez. Serviços manuais demandam tempo e atenção, sendo propícios a erros e perca de prazos e demandas. Portanto, faz-se necessário aderir a serviços informatizados para agilizar processos e otimizar atividades rotineiras. Este trabalho focaliza a criação de um sistema de gestão personalizado para o Projeto CARA, um serviço de convivência e fortalecimento de vínculos (SCFV), utilizando React.js, procurando trazer maior rapidez e eficiência nos processos desenvolvidos durante o ano todo.

Palavras-chave: Gestão escolar; Eficiência; Processos; aplicativo; ReactJS.

ABSTRACT

The advancement of Information Technology (IT) has provided significant opportunities to optimize processes across various sectors, including education and social projects. The need to meet demands and adhere to tight deadlines has led companies and organizations to adopt technological solutions. In the educational context, IT has become a valuable ally in managing students, enrollments, courses, and staff. School management involves many processes and information directly related to efficiency and speed. Manual services require time and attention, making them prone to errors and delays. Therefore, it is necessary to adopt computerized services to streamline processes and optimize routine activities. This work focuses on creating a customized management system for the CARA Project, a service for social interaction and strengthening bonds, using React.js to bring greater speed and efficiency to the processes carried out throughout the year.

Keywords: School management; Efficiency; Processes; Application; ReactJS.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

SUMÁRIO

1. INTRODUÇÃO

Com o avanço da tecnologia da informação (TI), tornaram-se amplas as oportunidades em agilizar e otimizar os processos. Uma empresa hoje torna-se obrigada a aderir a meios tecnológicos para suprir demandas e cumprir curtos prazos de entrega. No meio escolar, a TI se tornou uma grande aliada de escolas e centros de ensino, proporcionando que gestores controlem os alunos, matrículas, cursos, colaboradores e demais meios escolares.

Em projetos sociais de menor escala, como o Projeto CARA, que é uma iniciativa social em que visa o desenvolvimento comportamental e profissional de adolescentes para a inclusão ao mercado de trabalho, a organização de tarefas e documentos relacionados à gestão se torna um desafio significativo, especialmente quando se trata da administração de um Serviço de Convivência e Fortalecimento de Vínculos (SCFV).

> O **SCFV** integra o conjunto de serviços do SUAS, oferecendo à população que vivencia situações de vulnerabilidades sociais, novas oportunidades de reflexão acerca da realidade social, contribuindo dessa forma para a planejamento de estratégias e na construção de novos projetos de vida (MEDEIROS, 2023, p.).

Nesse contexto, o presente trabalho tem como objetivo principal a criação de um sistema de gestão que lide com as diversas demandas específicas do SCFV em questão, auxiliando de maneira eficaz nas atividades cotidianas. Contudo, devido à diversidade de atividades propostas por sistemas semelhantes, foi necessário o desenvolvimento de um sistema personalizado capaz de abranger todas as necessidades do Projeto CARA.

Segundo Franklin e Samuel Filho (2020), foi enfrentado um problema com tarefas manuais referente ao tempo gasto para exercê-las, enquanto a isso, foi desenvolvido um Sistema de Gestão Escolar com o uso da Linguagem *Dart* com *Framework Flutter*, e ao automatizar as tarefas operacionais, o tempo gasto nessas atividades pode ser significativamente reduzido. O aplicativo proposto visa facilitar os processos gerenciais da instituição educacional, proporcionando uma maior organização e eficiência.

Oliveira e Oliveira (2020) observam uma carência em sistemas que auxiliam os professores a realizarem suas chamadas diárias de forma prática e eficiente. Com o aplicativo chamado Chamon, criado para ser utilizado em sistemas Android, foi possível desenvolver um crachá

que possui um código QR (*Quick Response Code*) ou em português, Código de Resposta Rápida único, que o professor possa escanear e, assim, contabilizar e informar os responsáveis por meio de mensagens no celular.

Conforme exposto por Monteiro e Lara (2016), a ausência da interação em tempo real entre o aluno e professor pode fazer com que a aula seja seguida estritamente pelo cronograma proposto, sem a possibilidade de que assuntos não compreendidos sejam aprofundados pelo fato de que a dúvida não consiga chegar ao professor. Com esse problema em mente, foi possível desenvolver um aplicativo na plataforma Android em que sua utilização será de uso tanto do professor como do aluno, permitindo uma maior interação e dinamicidade durante as aulas.

1.1. OBJETIVOS

1.1.1. Objetivos gerais

O objetivo primordial deste trabalho de conclusão de curso é a criação de um sistema integrado abrangente, com foco em otimizar a gestão do Serviço de Convivência e Fortalecimento de Vínculos destacado. Esse sistema visa preencher as lacunas decorrentes da gestão manual, abrangendo não apenas os processos internos e a documentação relacionada, mas também a eficaz administração dos adolescentes envolvidos no Projeto CARA.

1.1.2. Objetivos específicos

Como objetivos específicos, desenvolveram-se as seguintes funcionalidades responsáveis para a construção do aplicativo:

- Desenvolver um *software* integrado capaz de simplificar e automatizar o processo de chamadas escolares, tornando-o mais eficiente e acessível;
- Criar uma funcionalidade no *software* que permita a realização de matrículas de forma ágil, proporcionando uma experiência simplificada para os usuários;
- Estabelecer um sistema de gerenciamento de alunos matriculados, com informações detalhadas sobre cada aluno, facilitando o acompanhamento acadêmico e administrativo;

- Desenvolver módulos de gerenciamento de turmas, tornando mais fácil a organização e alocação de estudantes em diferentes grupos;
- Implementar um sistema abrangente de gerenciamento de atividades em sala de aula, incluindo a possibilidade de criação, atribuição e acompanhamento de tarefas;
- Incorporar uma funcionalidade de agendamento de atividades extracurriculares para promover uma gestão mais eficaz das atividades fora da sala de aula;
- Criar relatórios automatizados de presença, fornecendo uma visão geral das frequências dos alunos e permitindo uma análise mais eficaz da participação nas aulas;
- Gerar relatórios detalhados sobre as atividades em sala de aula, fornecendo informações valiosas para avaliação e melhoria contínua do processo educacional.

Os objetivos específicos mencionados visam aprimorar a eficiência, a acessibilidade e a organização dos processos destacados, simplificando as tarefas diárias relacionadas à administração do Projeto CARA.

1.2. JUSTIFICATIVA

O avanço da tecnologia da informação (TI) tem revolucionado como empresas e instituições lidam com seus processos, tornando-os mais ágeis, eficientes e aptos a atender as demandas em constante evolução. No setor educacional, a TI tem desempenhado um papel crucial ao oferecer ferramentas que permitem a gestão eficaz de alunos, matrículas, cursos, colaboradores e outros aspectos relacionados ao ambiente escolar.

No entanto, quando observamos projetos sociais de menor escala, como o Projeto CARA, que visa oferecer um serviço de convivência e fortalecimento de vínculos (SCFV), enfrentamos desafios significativos na organização e administração das tarefas e documentos relacionados à gestão. Dentro deste contexto, a necessidade de adotar meios tecnológicos para otimizar a gestão do Projeto CARA se torna imperativa. Esse projeto social, assim como muitos outros, lida com uma quantidade significativa de informações, incluindo dados de participantes, atividades, relatórios e documentos essenciais para sua operação eficaz.

Assim, considerando o desafio que o Projeto CARA enfrenta na gestão de suas atividades e documentos, a introdução de um sistema de gestão com base em TI é justificada não

apenas pela eficiência operacional, mas também por seu potencial de aprimorar o impacto social e contribuir para a consecução dos objetivos do projeto.

1.3. MOTIVAÇÃO

A ausência de recursos informatizados no ambiente escolar frequentemente resulta em dificuldades para uma gestão eficiente dos processos, que, muitas vezes, dependem de métodos manuais. Reconhecendo esse desafio, surgiu a motivação para desenvolver um *software* que atenda às necessidades do Projeto CARA e, assim, aprimore a eficiência tanto na administração escolar e nas salas de aula.

Esse *software* será elaborado para abranger as demandas específicas do SCFV destacado, oferecendo uma solução que visa otimizar os processos de gestão, tornando-os mais ágeis e eficazes. Além disso, busca-se melhorar a dinâmica das atividades em sala de aula, em que os educadores podem se beneficiar das funcionalidades e recursos oferecidos pela plataforma.

1.4. PERSPECTIVAS DE CONTRIBUIÇÃO

Espera-se que esse trabalho represente uma contribuição para a gestão escolar, proporcionando melhorias na organização e na eficiência dos processos educacionais. O objetivo é que o aplicativo desenvolvido possa atender as demandas específicas do SCFV, oferecendo soluções e funcionalidades que promovam a integração e a simplificação dos procedimentos.

1.5. METODOLOGIA

Para alcançar os objetivos propostos, o projeto seguirá as seguintes etapas: primeiramente, será realizada uma fase de pesquisa e entendimento, isso inclui estudos teóricos para examinar projetos similares e reuniões diretas com o gestor a fim de obter todos os requisitos necessários para o desenvolvimento do *software*. Após a conclusão dessa fase inicial, os requisitos levantados serão analisados. Para essa análise, será usado o *software Astah Community*, que auxiliará na criação de mapas mentais, diagramas de casos de uso,

diagramas de atividade, de sequência, de classes. Por fim, a elaboração do protótipo funcional do projeto ocorrerá com o auxílio da plataforma *Figma*.

Posteriormente, a etapa de desenvolvimento, a qual será iniciada com a implementação do banco de dados. Opto por empregar o *PostgreSQL*, um banco de dados relacional, e a ferramenta de mapeamento objeto-relacional (ORM) Prisma. Essa escolha visa garantir a estruturação e estabilidade do banco para suportar entradas e saídas de dados. Para o desenvolvimento dos processos de criação, leitura, atualização e exclusão (CRUD) e demais operações, utilizando *Typescript*, *ReactJS* e o *Framework NextJS* para o *Front-end*, além do *NodeJS* para o *Back-end*. Essa abordagem, baseada em tecnologias modernas e eficazes, permitirá construir um software robusto e adequado aos requisitos estabelecidos.

Após a conclusão da implementação do *software*, será finalizado com a fase de testes, aplicando testes unitários e modulares. Esse processo visará garantir que a solução seja entregue com funcionamento adequado, pronto para uma futura implantação.

1.6. ESTRUTURA DO PROJETO

O presente trabalho está estruturado da seguinte maneira: No Capítulo 1, estabelece-se a contextualização do problema, o objetivo, o público-alvo, metodologia empregada e a perspectiva de contribuição e cronograma.

No Capítulo 2, é apresentada as tecnologias principais e auxiliares utilizadas durante o desenvolvimento do software.

O Capítulo 3 aborda a modelagem do trabalho, exibindo todos os diagramas elaborados ao longo do processo de desenvolvimento.

O Capítulo 4 demonstra o software, exibindo os principais fluxos demonstrados com imagens de suas respectivas funcionalidades.

Por fim, o capítulo 5 apresenta a conclusão e trabalhos futuros.

2. METODOLOGIAS E FERRAMENTAS UTILIZADAS

Para o desenvolvimento do *software* proposto, foram selecionadas tecnologias que visam atingir todos os objetivos de forma eficiente. Essa escolha baseou-se em uma análise detalhada das necessidades do projeto, considerando diversos fatores, como desempenho, facilidade de manutenção, compatibilidade e custo.

Para os diagramas, será empregada as ferramentas *Astah UML*[1], *Miro*[2] e o *software DbDesigner*[3]. No *front-end*, será utilizado o *framework NextJS*[4], que faz uso da linguagem *TypeScript*[5] para o desenvolvimento. Além disso, para estilização das páginas será utilizado o *framework Tailwind CSS*[6]. Para o desenvolvimento do *back-end*, será utilizada a linguagem *JavaScip*t em conjunto com *NodeJS*[7].

Para o banco de dados, foi escolhido o *PostgreSQL*, que será hospedado na plataforma *SupaBase*[8]. Já no armazenamento e versionamento da aplicação, será utilizada a plataforma *GitHub*, que permitirá um controle e armazenamento dos códigos. Por fim, a IDE (*Integrated Development Environment*) usada no desenvolvimento do *back-end* e *front-end* será o *Visual Studio Code*[9].

2.1. DIAGRAMAS

Os diagramas são fundamentais para a visualização e planejamento de sistemas, proporcionando uma compreensão clara das estruturas e processos envolvidos. Eles ajudam a organizar ideias, definir fluxos de trabalho e mapear as relações entre diferentes componentes. Para criar esses diagramas, existem várias ferramentas especializadas que oferecem recursos variados, facilitando a construção de representações visuais detalhadas e precisas. Essas ferramentas são essenciais para o desenvolvimento e documentação de

[1] https://astah.net/products/astah-uml/
[2] https://miro.com/pt/
[3] https://dbdesigner.en.softonic.com/
[4] https://nextjs.org/
[5] https://www.typescriptlang.org/
[6] https://tailwindcss.com/
[7] https://nodejs.org/en
[8] https://supabase.com/
[9] https://code.visualstudio.com/

projetos complexos, permitindo que as ideias sejam transformadas em modelos visuais compreensíveis e eficientes.

2.1.1. Figma

O *Figma* [10]é uma ferramenta de design em que permite que o usuário crie, edite e compartilhe projetos. Diferenciando-se de outras ferramentas tradicionais, o *Figma* é acessível diretamente pelo navegador, eliminando a necessidade de instalação de *software* e simplificando o processo de compartilhamento e colaboração.

2.1.2. DbDesigner

O *DbDesigner* é uma ferramenta CASE de modelagem de dados que permite modelar, criar editar e gerenciar bancos de dados. Além disso, a ferramenta também possibilita engenharia reversa, gerando o modelo ER a partir de um banco já existente.

2.1.3. Astah UML

Astah UML é uma ferramenta de modelagem UML desenvolvida pela Astah. O Astah UML oferece suporte a uma variedade de diagramas UML, incluindo diagramas de casos de uso, diagramas de classes, diagramas de sequência, diagramas de atividade, diagramas de estado e muitos outros. Ele fornece uma interface gráfica intuitiva que permite aos usuários desenhar e editar diagramas de forma eficiente.

2.1.4. Miro

Miro é uma plataforma *online* colaborativa que permite às equipes colaborarem em tempo real em projetos de forma visual. Ela é amplamente utilizada para facilitar processos de

[10] https://www.figma.com/pt-br/

brainstorming, criação de diagramas, planejamento de projetos, desenvolvimento de mapas mentais, criação de *wireframes*, entre outras atividades de colaboração visual.

2.2. BACK-END

O back-end é a parte do sistema responsável por gerenciar a lógica, o armazenamento de dados e a comunicação entre o servidor e o front-end. Ele garante que as funcionalidades principais funcionem corretamente, processando solicitações, executando operações de banco de dados e gerenciando a autenticação e autorização de usuários. Ferramentas e tecnologias diversas são utilizadas no desenvolvimento do back-end, permitindo a criação de aplicações robustas e escaláveis, que suportam o tráfego de dados e garantem a segurança e eficiência do sistema.

2.2.1. JavaSript

JavaScript é uma linguagem de programação de alto nível, interpretada e orientada a objetos, amplamente utilizada no desenvolvimento *web*. Criada originalmente para tornar as páginas *web* interativas, ela evoluiu para se tornar uma linguagem de propósito geral, sendo executada não apenas nos navegadores *web*, mas também em servidores, dispositivos móveis e até mesmo em aplicações de *desktop*.

2.2.2. NodeJS

O *Node.js* é uma plataforma de código aberto baseada no motor V8 do Google Chrome, projetada para construir aplicativos de rede escaláveis. Ele utiliza *JavaScript* como linguagem de programação principal e é especialmente adequado para aplicativos que requerem comunicação em tempo real, manipulação de entrada/saída intensiva de dados ou que precisam lidar com muitos clientes simultâneos.

A principal característica do Node.js é sua capacidade de executar *JavaScript* no lado do servidor, o que antes era restrito ao ambiente do navegador. Isso significa que os desenvolvedores podem usar *JavaScript* tanto no *frontend* quanto no *backend* de seus

aplicativos, o que pode levar a uma maior coesão e reutilização de código entre as camadas da aplicação.

2.3. FRONT-END

O front-end é a parte do sistema que interage diretamente com o usuário, responsável pela interface gráfica e pela experiência de navegação. Ele traduz os dados e funcionalidades do back-end em elementos visuais e interativos, permitindo que os usuários utilizem o sistema de maneira intuitiva. No desenvolvimento do front-end, são empregadas diversas ferramentas e tecnologias que facilitam a criação de interfaces dinâmicas, responsivas e esteticamente agradáveis, garantindo uma experiência de usuário positiva e acessível em diferentes dispositivos e plataformas.

2.3.1. NextJS

Next.*JS* é um *framework* de *React*.*JS* utilizado para construir aplicações *web* modernas e escaláveis. Ele oferece funcionalidades avançadas, como renderização do lado do servidor (SSR), geração de páginas estáticas, roteamento simples e otimização de desempenho, facilitando o desenvolvimento de aplicações rápidas e eficientes.

> O principal objetivo desse framework é a produção de código em menos tempo possível, bem como a eficiência do mesmo. O Next.js foi criado e está sendo mantido pela equipe da Vercel e, ele é um framework open-source, ou seja, toda comunidade pode contribuir com o seu desenvolvimento (Marchiori, 2023)

2.3.2. TypeScript

TypeScript é uma linguagem de programação de código aberto desenvolvida e mantida pela Microsoft. Ela é uma extensão da linguagem *JavaScript* que adiciona recursos de tipagem estática opcionais e outros recursos de programação orientada a objetos ao *JavaScript*.

2.3.3. Tailwind CSS

Tailwind CSS é uma estrutura de desenvolvimento de CSS de código aberto que utiliza uma abordagem diferente de outras estruturas como *Bootstrap* ou *Foundation*. Em vez de fornecer componentes pré-estilizados, o *Tailwind CSS* oferece um conjunto abrangente de classes utilitárias que podem ser combinadas para construir interfaces de usuário personalizadas.

Em vez de escrever *CSS* personalizado, os desenvolvedores usam as classes utilitárias fornecidas pelo *Tailwind CSS* diretamente em seus elementos *HTML* para estilizá-los. Essas classes são projetadas para serem simples e intuitivas, seguindo uma convenção de nomenclatura consistente.

2.4. BANCO DE DADOS

O banco de dados é o coração do sistema, responsável por armazenar, organizar e gerenciar as informações de maneira estruturada e acessível. Ele permite que os dados sejam consultados, inseridos, atualizados e excluídos de forma eficiente, garantindo a integridade e consistência das informações. Diversas ferramentas e tecnologias de banco de dados são utilizadas para criar estruturas que suportam grandes volumes de dados, oferecem segurança e permitem consultas rápidas e eficazes. Um banco de dados bem projetado é essencial para o desempenho e a escalabilidade do sistema, assegurando que as informações estejam sempre disponíveis e protegidas.

2.4.1. PostgreSQL

Segundo a Microsoft (2024), o *PostgreSQL* é um banco de dados relacional de *software* livre com suporte de 30 anos de desenvolvimento, sendo um dos bancos de dados relacionais mais estabelecidos disponíveis. A popularidade do PostgreSQL com desenvolvedores e administradores se deve à sua flexibilidade e integridade notáveis.

2.5. VERSIONAMENTO

O versionamento de código é uma prática essencial no desenvolvimento de software, permitindo o controle e o gerenciamento das diferentes versões de um projeto ao longo do tempo. Com ele, os desenvolvedores podem acompanhar as mudanças, colaborar de forma eficiente em equipe e reverter a qualquer versão anterior, se necessário. Ferramentas de versionamento de código permitem criar ramificações para desenvolvimento paralelo, gerenciar conflitos de código e garantir que o histórico completo do projeto seja mantido de forma organizada e segura. Essa prática é fundamental para a continuidade e integridade do desenvolvimento, especialmente em projetos de grande escala e com múltiplos colaboradores.

2.5.1. GitHub

O *GitHub*[11] é um serviço baseado em nuvem que hospeda um sistema de controle de versão chamado *Git*. Essa plataforma *online* permite que os usuários armazenem códigos na nuvem, além de poderem contribuir entre si para produção de projetos privados ou *Open Source.*

2.6. IDE (AMBIENTE DE DESENVOLVIMENTO INTEGRADO)

Uma IDE (Ambiente de Desenvolvimento Integrado) é uma ferramenta essencial para programadores, oferecendo um ambiente unificado onde o código pode ser escrito, testado e depurado. As IDEs combinam vários recursos, como editores de código com destaque de sintaxe, autocompletar, depuradores, controle de versionamento e integração com outras ferramentas de desenvolvimento. Esses ambientes facilitam o trabalho dos desenvolvedores, aumentando a produtividade e ajudando a evitar erros comuns

[11] https://github.com/

2.6.1. Visual Studio Code

O *Visual Studio Code*, ou simplesmente VS Code, é um editor de código de código aberto desenvolvido pela Microsoft. Essa *IDE* permite o desenvolvimento com diversas linguagens como *TypeScript*, *JavaScript*, *Python*, *Java*, *C#* entre outras. Além disso, é possível desenvolver páginas *web* com *HTML* e *CSS*.

3. DESENVOLVIMENTO DO TRABALHO

Nesse capítulo, será apresentada uma visão geral dos processos envolvidos na criação do *software*, como levantamento de requisitos e planejamento do sistema a partir dos diagramas.

3.1. REQUISITOS DO SISTEMA

Os requisitos do sistema são a base essencial para o desenvolvimento de software, representando um conjunto de especificações que definem o que o sistema deve fazer e como deve se comportar. Eles estabelecem as diretrizes e expectativas para o projeto, orientando todas as fases do desenvolvimento, desde a concepção até a implementação e manutenção. A clareza e a precisão na definição desses requisitos são cruciais para garantir que o produto final atenda às necessidades dos usuários e funcione de acordo com as expectativas.

3.1.1. Levantamento de Requisitos

O levantamento de requisitos é o início do desenvolvimento, no qual pretende-se compreender e identificar as necessidades que o cliente espera ser resolvidas pelo *software* requisitado, sendo assim, estabelecendo as funções do sistema.

Dessa forma, foram determinados os requisitos que o sistema deve cumprir:

- Fazer login;
- Manter Aluno;
- Manter Professor;
- Manter Matrícula;
- Manter Cursos;
- Manter Lista de Chamada;
- Manter Faltas Disciplinares;
- Manter Atividades Escolares;
- Manter Disciplinas;

- Manter Turma;
- Manter Conteúdo de Aula;
- Controlar Frequência;

3.1.2. Mapa Mental

O mapa mental é um diagrama que ajuda a explicar conceitos complicados de maneira simples e objetiva. Um mapa mental ajuda organizar as ideias com o objetivo de criar e manter uma melhor compreensão das informações com uma representação visual.

Ao ser aplicado ao levantamento de requisitos, ele simplifica a visualização das necessidades do projeto, identificando as relações entre os requisitos e promovendo um melhor entendimento de ideias entre os membros do time de desenvolvimento.

A figura 3 demonstra o mapa mental criado para o presente trabalho.

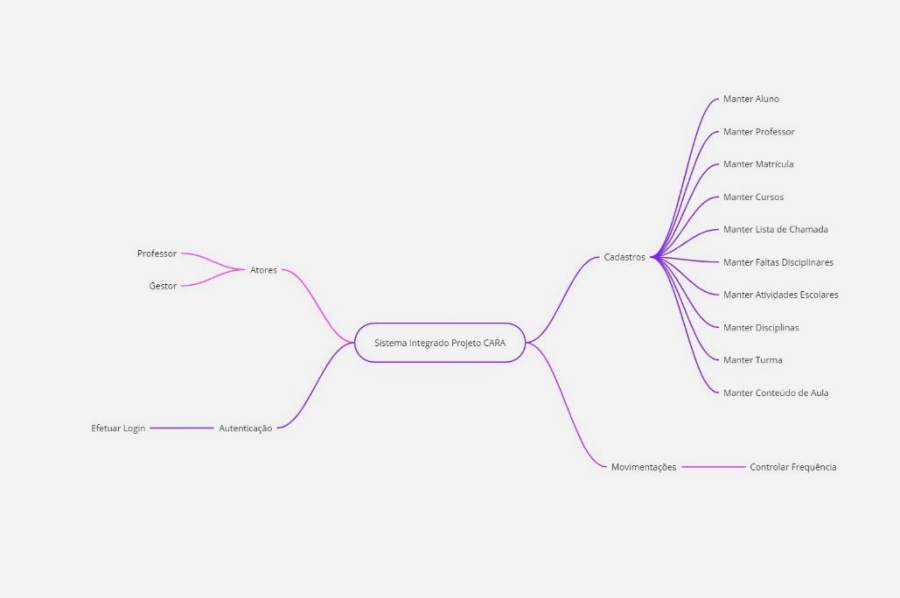


Figura 1: Mapa Mental

Fonte: Autoria própria.

3.2. CASOS DE USO

Os casos de uso descrevem a função que um sistema desempenha para alcançar a meta do usuário. “Um caso de uso deve produzir um resultado observável que seja valioso para o usuário do sistema” (IBM, 2021, online).

3.2.1. Agentes

Os agentes representam as funções de um usuário que interagirá com o sistema que está sendo modelado. “O usuário pode ser um humano, uma organização, uma máquina ou outro sistema externo” (IBM 2021, online).

Os atores que atuarão sobre o sistema proposto são:

- Gestor;
- Professor.

3.2.2. Diagrama de Caso de Uso

A figura 4 representa o diagrama de casos de uso referente ao Sistema Integrado para o Projeto CARA.

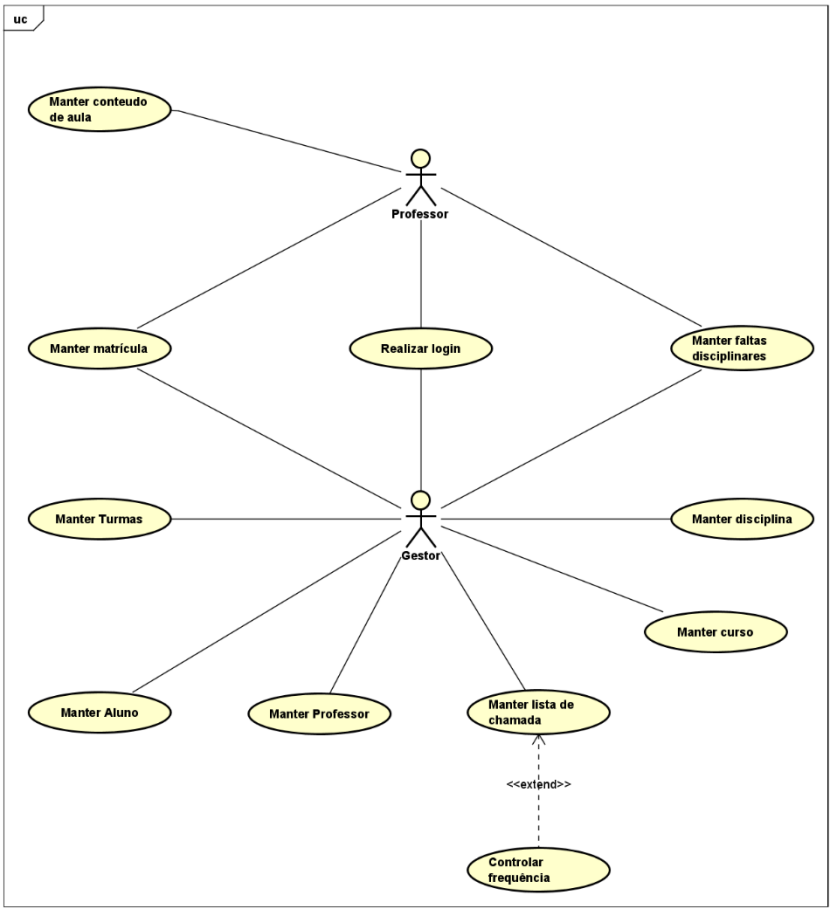


Figura 2: Diagrama de Caso de Uso.

Fonte: Autoria própria.

3.3. NARRATIVAS DE CASO DE USO

As narrativas de caso de uso são uma maneira de relatar os casos de uso de um sistema de forma que seja de fácil compreendimento para todos que estão envolvidos no processo de desenvolvimento, mesmo que não tenha conhecimento técnico pela área. Uma narrativa de caso de uso descreve como um usuário ou "ator" interage com o sistema para alcançar seu objetivo.

A seguir, serão apresentadas as narrativas de caso de uso propostas para o presente trabalho.

3.3.1. Manter conteúdo de aula

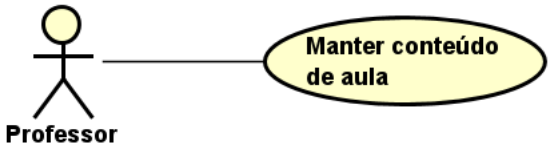


Figura 3: Diagrama UC Manter Conteúdo de Aula.

Fonte: Autoria própria.

1. Finalidade/ Objetivo	Gerenciar conteúdos dados em aula.
2. Atores	Professor.
3. Pré-condição	Estar *logado* no sistema.
4. Evento Inicial	O ator deve selecionar iniciar a funcionalidade conteúdos de aula.
5. Fluxo Principal	a) O sistema oferecerá uma interface onde o ator poderá selecionar a turma desejada; b) O ator seleciona a turma; c) O sistema apresenta uma lista de com todos os conteúdos já cadastrados [A1]; d) O ator seleciona o conteúdo desejado; [A2] [A3];

	e) O sistema apresenta todas as informações referentes ao conteúdo desejado [A5]; f) O ator aperta no botão de voltar; g) Caso de uso encerrado.
6. Fluxo Alternativo	A1 – Incluir novo conteúdo a) O ator escolhe a opção “novo conteúdo”; b) O sistema apresenta todos os campos a serem preenchidos; c) O ator preenche os campos e aperta em salvar [E1] [T1]; d) O sistema exibe uma mensagem “Conteúdo cadastrado com sucesso”; e) Volta ao fluxo principal, letra “a”. A2 – Alterar conteúdo a) O usuário seleciona a opção alterar conteúdo; b) O sistema exibirá todas as informações do conteúdo, permitindo que possa ser alterado; c) O ator aperta no botão “salvar”; d) O sistema exigirá que o ator confirme as alterações realizadas no conteúdo [A4]; e) O sistema exibe uma mensagem “Conteúdo editado com sucesso”; f) Volta ao fluxo principal, letra “a”. A3 – Deletar conteúdo a) O ator seleciona a opção “deletar”; b) O sistema exige que o ator confirme a exclusão do conteúdo [A4]; c) O sistema exibe uma mensagem “Conteúdo excluído com sucesso”; d) Volta ao fluxo principal, letra “a”. A4 – O ator cancela a exclusão a) O ator escolhe cancelar a alteração ou exclusão; b) O sistema exibe uma mensagem “Operação cancelada”; c) Volta ao fluxo principal, letra “a”.

	A5 – Emitir relatório O sistema emitirá um relatório de sala, contendo todas as informações do conteúdo dado no dia.
7. Fluxo de Exceção	E1 – Campos preenchidos incorretamente a) O sistema informa quais campos estão digitados incorretamente; b) Retorna ao fluxo alternativo A1 na letra "c", mantendo os dados que foram digitados corretamente.
8. Testes	T1 – Testar campos O sistema deve testar se todos os campos foram preenchidos corretamente.

Tabela 1: Manter Conteúdo de Aula.

Fonte: Autoria própria.

3.3.2. Manter matrícula

Figura 4: Diagrama UC Manter Matrícula.

Fonte: Autoria própria.

1. Finalidade/ Objetivo	Gerenciar matrícula dos alunos.
2. Atores	Professor e Gestor.
3. Pré-condição	Estar *logado* no sistema.

4. Evento Inicial	O ator deve selecionar iniciar a funcionalidade matrículas.
5. Fluxo Principal	a) O sistema oferecerá uma interface onde o ator poderá informar o nome do aluno; b) O ator informa o nome [A1]; c) O sistema apresenta uma lista de alunos conforme foi pesquisado [E1]; d) O ator seleciona o aluno desejado [A2] [A3]; e) O sistema apresenta todas as informações referentes a matrícula do aluno desejado; f) O ator aperta no botão de voltar; g) Caso de uso encerrado.
6. Fluxo Alternativo	A1 – Incluir nova matrícula a) O ator escolhe a opção nova matrícula; b) O sistema apresenta todos os campos a serem preenchidos; c) O ator preenche os campos e aperta em salvar [E2] [T1]; d) O sistema exibe uma mensagem “Matrícula realizada com sucesso”; e) Volta ao fluxo principal, letra “a”. A2 – Alterar matrícula a) O usuário seleciona a opção alterar doados de matrícula; b) O sistema exibirá todas as informações de matrícula, permitindo que possa ser alterado; c) O ator aperta no botão de salvar; d) O sistema exigirá que o ator confirme as alterações realizadas na matrícula [A4]; e) O sistema exibe uma mensagem “Matrícula editada com sucesso”; f) Volta ao fluxo principal, letra “a”. A3 – Deletar matrícula a) O ator seleciona a opção “deletar”; b) O sistema exige que o ator confirme a exclusão da matrícula [A4];

	c) O sistema exibe uma mensagem “Matrícula excluída com sucesso”; d) Volta ao fluxo principal, letra “a”. A4 – O ator cancela a exclusão a) O ator escolhe cancelar a alteração ou exclusão; b) O sistema exibe uma mensagem “Operação cancelada”; c) Volta ao fluxo principal, letra “a”.
7. Fluxo de Exceção	E1 – Aluno não encontrado a) O sistema exibe uma mensagem “Aluno não encontrado”; b) O campo de busca é resetado e retorna ao fluxo principal, letra “b”. E2 – Campos preenchidos incorretamente a) O sistema informa quais campos estão digitados incorretamente; b) Retorna ao fluxo alternativo A1 na letra “c”, mantendo os dados que foram digitados corretamente.
8. Testes	T1 – Testar campos O sistema deve testar se todos os campos foram preenchidos corretamente.

Tabela 2: Manter Matrícula.

Fonte: Autoria própria.

3.3.3. Controlar frequência

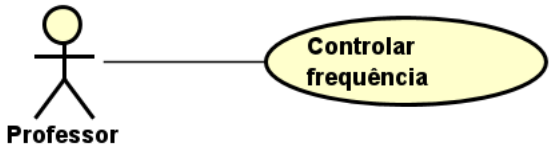


Figura 5: Diagrama UC Controlar Frequência.

Fonte: Autoria própria.

1. Finalidade/ Objetivo	Controlar a frequência dos adolescentes durante as aulas.
2. Atores	Professor.
3. Pré-Condição	Estar logado no sistema.
4. Evento Inicial	O professor inicia o processo abrindo a lista de chamada e selecionando a turma desejada.
5. Fluxo Principal	a) O sistema oferece uma interface intuitiva e de fácil entendimento; b) O sistema mostrará todos os alunos cadastrados na turma selecionada [E1]; c) O professor deverá selecionar se o aluno está presente ou não na aula do dia [A1] [T1]; d) Caso de uso encerrado.
6. Fluxo Alternativo	A1 – O professor não informou a presença do aluno a) O sistema emitirá um alerta informando que o professor deverá informar a presença do aluno; b) O sistema retornará ao fluxo principal letra “b”.
7. Fluxo de Exceção	E1 – O aluno não está cadastrado na turma a) O professor deverá informar a secretaria para que possa realizar o cadastro do aluno.

8. Testes	T1 – O sistema testará se todos os campos foram preenchidos corretamente.

Tabela 3: Controlar Frequência

Fonte: Autoria própria

3.3.4. Manter faltas disciplinares

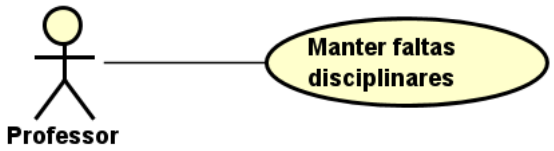


Figura 6: Diagrama UC Manter Faltas Disciplinares.

Fonte: Autoria própria.

1. Finalidade/ Objetivo	Gerenciar faltas disciplinares.
2. Atores	Professor.
3. Pré-condição	Estar *logado* no sistema.
4. Evento Inicial	O ator deve selecionar iniciar a funcionalidade faltas disciplinares.
5. Fluxo Principal	a) O sistema oferecerá uma interface onde o ator poderá informar o nome do aluno; b) O ator informa o nome; c) O sistema apresenta uma lista de aluno conforme foi pesquisado [E1]; d) O ator seleciona o professor desejado [A1] [A2] [A3]; e) O sistema apresenta todas as informações referentes ao cadastro do professor desejado; f) O ator aperta no botão de voltar; g) Caso de uso encerrado.

6. Fluxo Alternativo	A1 – Incluir nova falta disciplinar a) O ator escolhe a opção novo cadastro; b) O sistema apresenta todos os campos a serem preenchidos; c) O ator preenche os campos e aperta em salvar [E2] [T1]; d) O sistema exibe uma mensagem “Cadastro realizado com sucesso”; e) Volta ao fluxo principal, letra “a”. A2 – Alterar falta disciplinar a) O usuário seleciona a opção alterar falta disciplinar; b) O sistema exibirá todas as informações da falta disciplinar, permitindo que possa ser alterado; c) O ator aperta no botão de salvar; d) O sistema exigirá que o ator confirme as alterações realizadas na falta disciplinar [A4]; e) O sistema exibe uma mensagem “falta disciplinar editada com sucesso”; f) Volta ao fluxo principal, letra “a”. A3 – Deletar falta disciplinar a) O ator seleciona a opção “deletar”; b) O sistema exige que o ator confirme a exclusão do cadastro [A4]; c) O sistema exibe uma mensagem “falta disciplinar excluída com sucesso”; d) Volta ao fluxo principal, letra “a”. A4 – O ator cancela a exclusão a) O ator escolhe cancelar a alteração ou exclusão; b) O sistema exibe uma mensagem “Operação cancelada”; c) Volta ao fluxo principal, letra “a”.
7. Fluxo de Exceção	E1 – Aluno não encontrado a) O sistema exibe uma mensagem “Aluno não encontrado”;

	b) O campo de busca é resetado e retorna ao fluxo principal, letra “b”. E2 – Campos preenchidos incorretamente a) O sistema informa quais campos estão digitados incorretamente; b) Retorna ao fluxo alternativo A1 na letra “c”, mantendo os dados que foram digitados corretamente.
8. Testes	T1 – Testar campos O sistema deve testar se todos os campos foram preenchidos corretamente.

Tabela 4: Manter Faltas Disciplinares.

Fonte: Autoria própria.

3.3.5. Manter professor

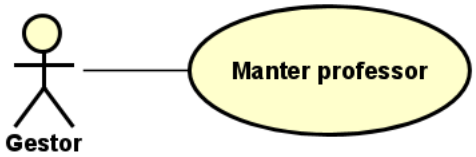


Figura 7: Diagrama UC Manter Professor.

Fonte: Autoria própria.

1. Finalidade/ Objetivo	Gerenciar os professores.
2. Atores	Gestor.
3. Pré-condição	Estar *logado* no sistema.
4. Evento Inicial	O ator deve selecionar iniciar a funcionalidade Professor.
5. Fluxo Principal	a) O sistema oferecerá uma interface onde o ator poderá informar o nome do professor;

	b) O ator informa o nome [A1]; c) O sistema apresenta uma lista de professores conforme foi pesquisado [E1]; d) O ator seleciona o professor desejado [A2] [A3]; e) O sistema apresenta todas as informações referentes ao cadastro do professor desejado; f) O ator aperta no botão de voltar; g) Caso de uso encerrado.
6. Fluxo Alternativo	A1 – Incluir novo cadastro a) O ator escolhe a opção novo cadastro; b) O sistema apresenta todos os campos a serem preenchidos; c) O ator preenche os campos e aperta em salvar [E2] [T1]; d) O sistema exibe uma mensagem “Cadastro realizado com sucesso”; e) Volta ao fluxo principal, letra “a”. A2 – Alterar cadastro a) O usuário seleciona a opção alterar doados do cadastro; b) O sistema exibirá todas as informações do cadastro, permitindo que possa ser alterado; c) O ator aperta no botão de salvar; d) O sistema exigirá que o ator confirme as alterações realizadas no cadastro [A4]; e) O sistema exibe uma mensagem “Cadastro editado com sucesso”; f) Volta ao fluxo principal, letra “a”. A3 – Deletar cadastro a) O ator seleciona a opção “deletar”; b) O sistema exige que o ator confirme a exclusão do cadastro [A4]; c) O sistema exibe uma mensagem “Cadastro excluído com sucesso”; d) Volta ao fluxo principal, letra “a”.

	A4 – O ator cancela a exclusão a) O ator escolhe cancelar a alteração ou exclusão; b) O sistema exibe uma mensagem “Operação cancelada”; c) Volta ao fluxo principal, letra “a”.
7. Fluxo de Exceção	E1 – Professor não encontrado. a) O sistema exibe uma mensagem “Professor não encontrado”; b) O campo de busca é resetado e retorna ao fluxo principal, letra “b”. E2 – Campos preenchidos incorretamente a) O sistema informa quais campos estão digitados incorretamente; b) Retorna ao fluxo alternativo A1 na letra “c”, mantendo os dados que foram digitados corretamente.
8. Testes	T1 – Testar campos O sistema deve testar se todos os campos foram preenchidos corretamente.

Tabela 5: Manter Professor.

Fonte: Autoria própria.

3.4. DIAGRAMA DE CLASSES

O diagrama de classes é uma ferramenta fundamental para a modelagem de sistemas orientados a objetos. Sendo utilizado para representar a estrutura estática de um sistema, incluindo suas classes, atributos, métodos e os relacionamentos entes elas.

Esse tipo de diagrama é uma das representações UML mais utilizadas por serem fundamentais na comunicação da estrutura do sistema para os membros do time de desenvolvimento. A figura 19 demonstra o diagrama de classes do sistema proposto.

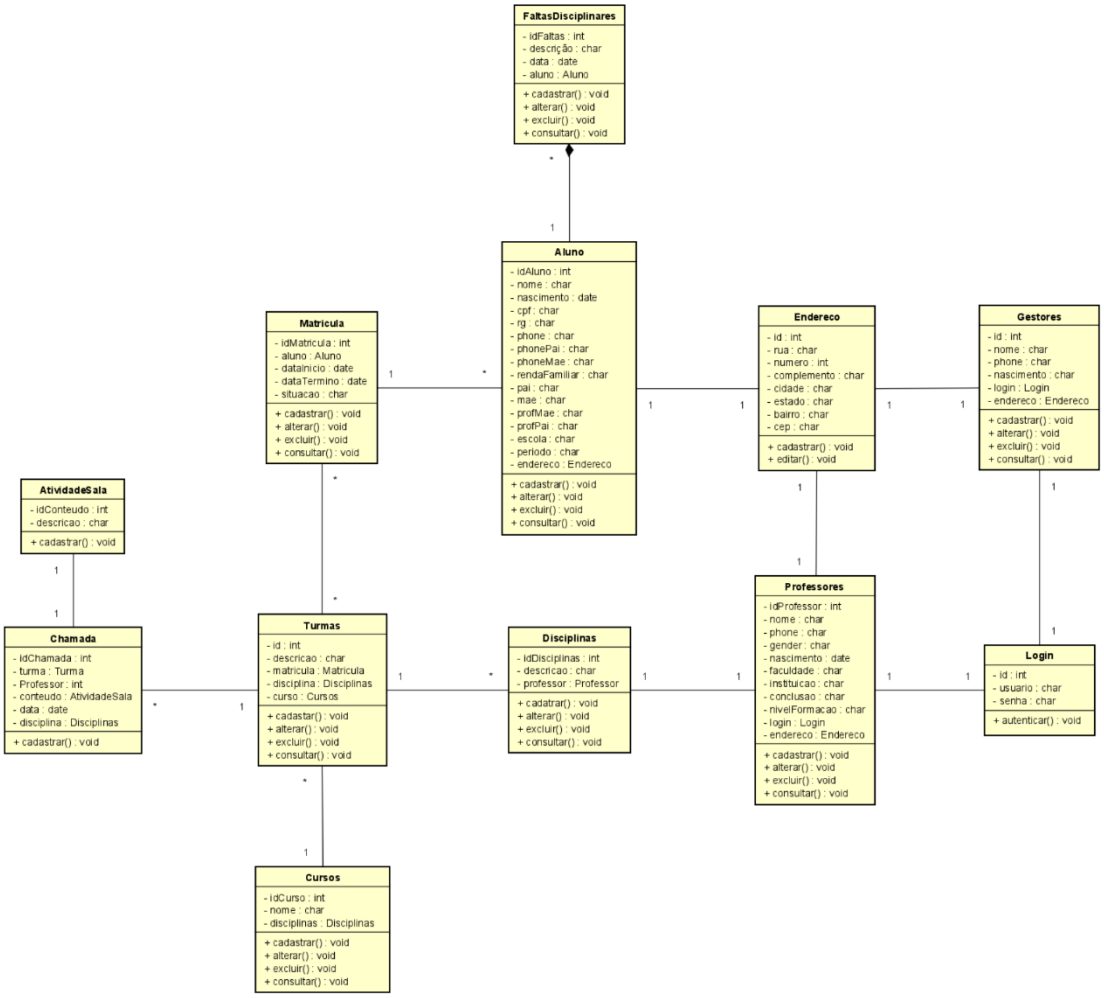


Figura 8: Diagrama de Classes.

Fonte: Autoria própria.

3.5. DIAGRAMA ENTIDADE-RELACIONAMENTO

Segundo Lucidchart ([20??]), o diagrama entidade-relacionamento (DER), ou diagrama ER, mostra como as entidades se relacionam entre si. Diagramas ER são comumente utilizados para projetar bancos de dados relacionais e usam um conjunto definido de símbolos, tais como retângulos, diamantes, ovais e linhas de conexão para representar a interconectividade de entidades, relacionamentos e seus atributos,

A figura 20 representa o DER desenvolvido para este projeto.

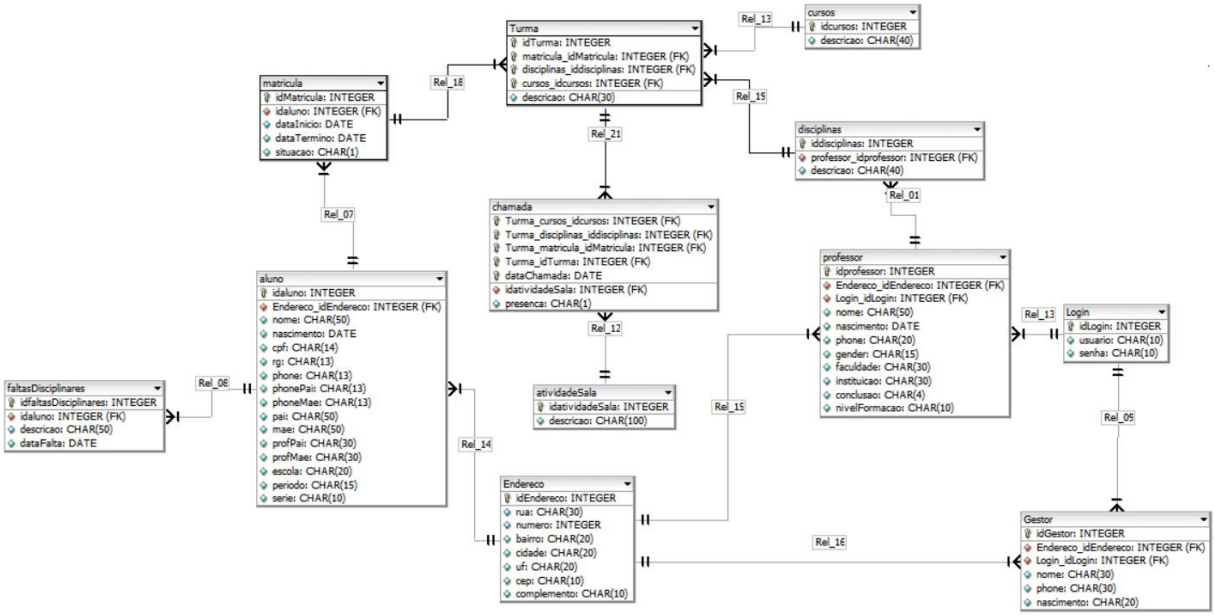


Figura 9: Diagrama Entidade-Relacionamento.

Fonte: Autoria própria.

3.6. ESTRUTURA ANALÍTICA DO PROJETO

A Estrutura Analítica do Projeto (EAP), é uma ferramenta usada na gestão de projetos para auxiliar na organização e planejamento do projeto. Esse tipo de diagrama divide o projeto em partes menores, facilitando a visualização e gerenciamento das atividades menores que precisam ser feitas para conclusão do projeto.

A figura 2 mostra a EAP desenvolvida para este projeto.

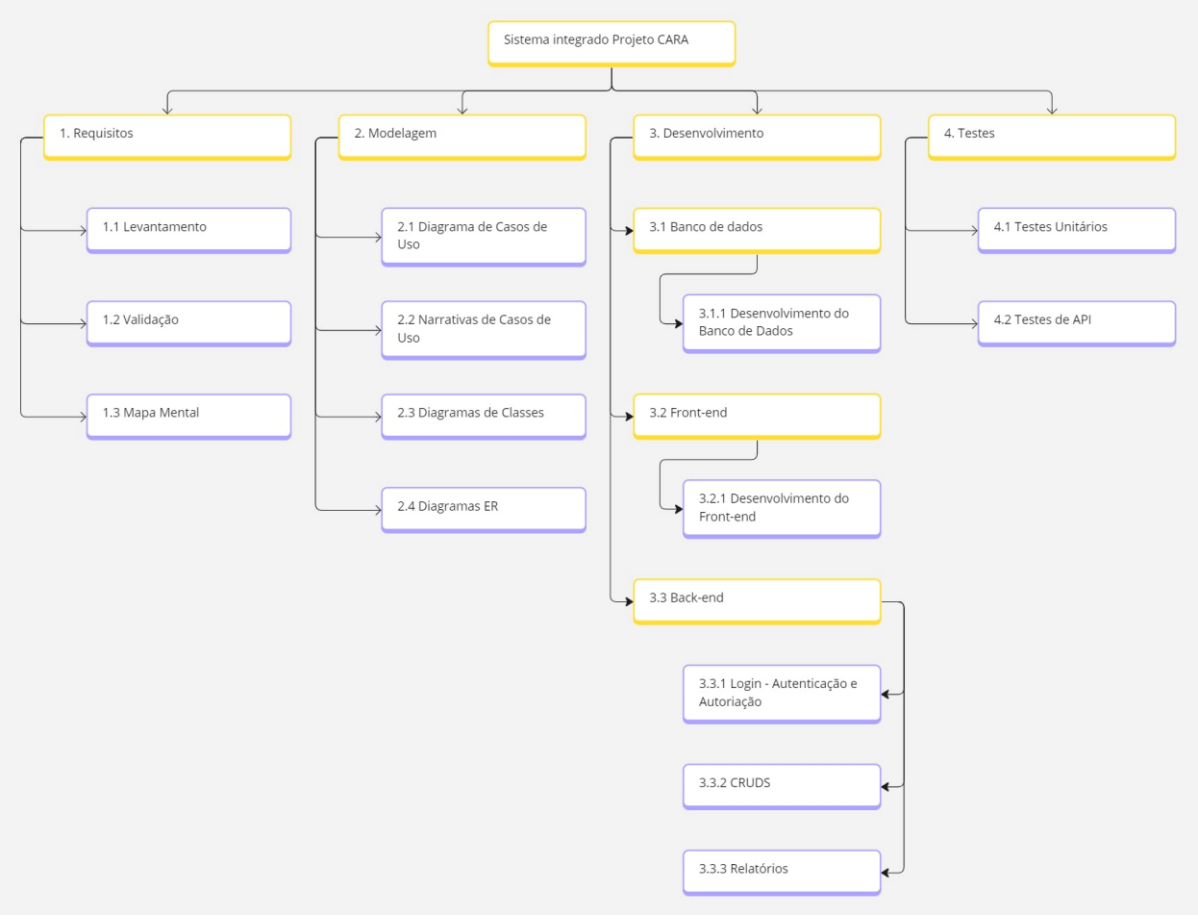


Figura 10: Estrutura Analítica do Projeto.

Fonte: Autoria própria.

4. DESENVOLVIMENTO DO SOFTWARE

Este capítulo tem o objetivo de detalhar as principais funcionalidades do software, oferecendo uma maior compreensão de como esse sistema pode impactar positivamente e SCFV como o Projeto CARA.

4.1. LOGIN

A tela de login é o ponto de entrada do sistema, onde o usuário insere seu código e senha fornecidos pelo gestor. Após a autenticação, o usuário obtém acesso às funcionalidades completas do software, garantindo uma experiência segura.

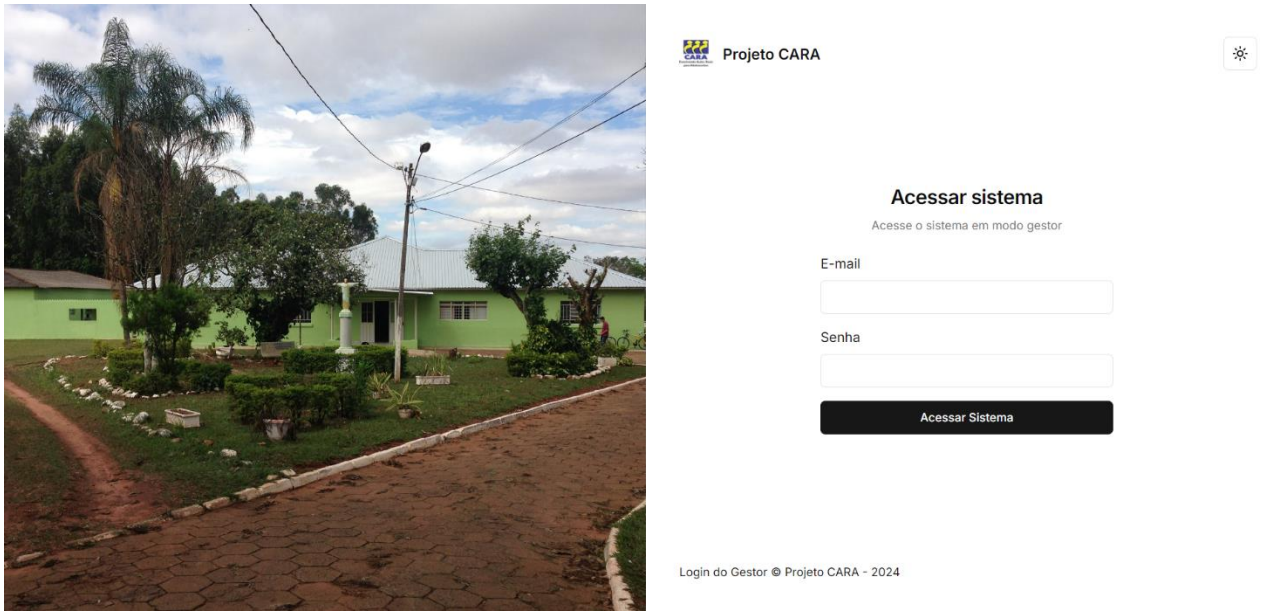
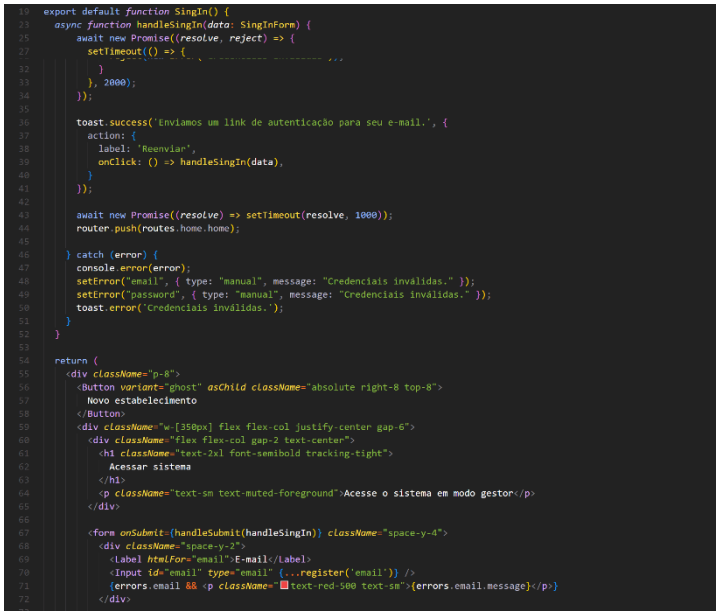


Figura 11: Tela de login

Fonte: Autoria própria.



```
export default function SingIn() {
  async function handleSingIn(data: SingInForm) {
      await new Promise((resolve, reject) => {
        setTimeout(() => {
          }
        }, 2000);
      });

      toast.success('Enviamos um link de autenticação para seu e-mail.', {
        action: {
          label: 'Reenviar',
          onClick: () => handleSingIn(data),
        }
      });

      await new Promise((resolve) => setTimeout(resolve, 1000));
      router.push(routes.home.home);

    } catch (error) {
      console.error(error);
      setError("email", { type: "manual", message: "Credenciais inválidas." });
      setError("password", { type: "manual", message: "Credenciais inválidas." });
      toast.error('Credenciais inválidas.');
    }
  }

  return (
    <div className="p-8">
      <Button variant="ghost" asChild className="absolute right-8 top-8">
        Novo estabelecimento
      </Button>
      <div className="w-[350px] flex flex-col justify-center gap-6">
        <div className="flex flex-col gap-2 text-center">
          <h1 className="text-2xl font-semibold tracking-tight">
            Acessar sistema
          </h1>
          <p className="text-sm text-muted-foreground">Acesse o sistema em modo gestor</p>
        </div>

        <form onSubmit={handleSubmit(handleSingIn)} className="space-y-4">
          <div className="space-y-2">
            <Label htmlFor="email">E-mail</Label>
            <Input id="email" type="email" {...register('email')} />
            {errors.email && <p className="text-red-500 text-sm">{errors.email.message}</p>}
          </div>
```

Figura 12: Código Tela de login

Fonte: Autoria própria.

4.2. MENU LATERAL

O menu lateral oferece fácil acesso a todas as funcionalidades do software, organizadas em grupos conforme suas categorias. Nele, o usuário encontra opções para cadastros, listagens e diversas outras funções do sistema, garantindo uma navegação intuitiva e eficiente.

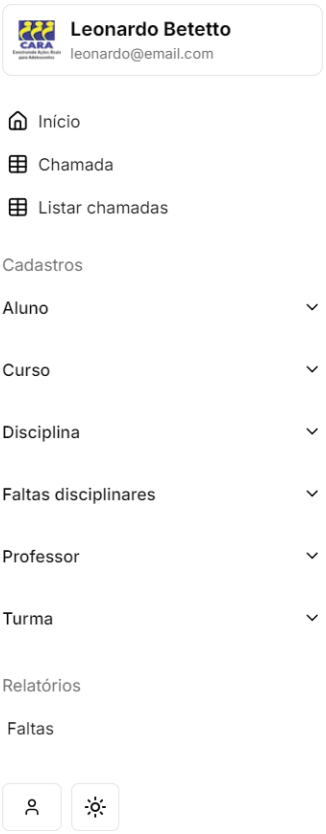
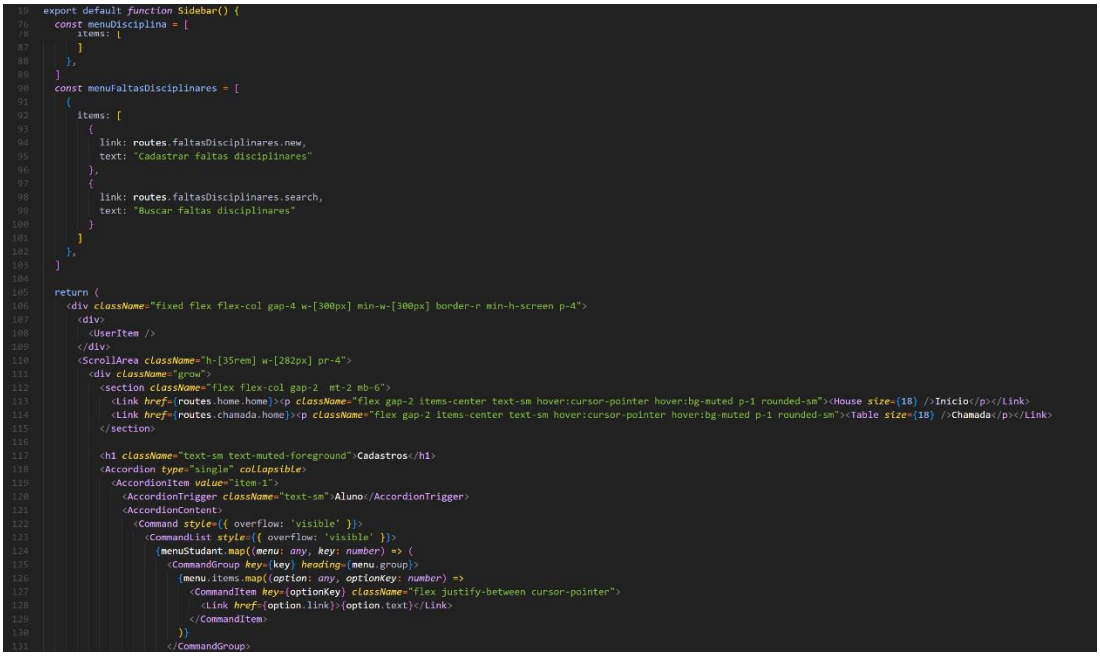


Figura 13: Menu lateral

Fonte: Autoria própria.



```
export default function Sidebar() {
  const menuDisciplina = [
      items: [
      ]
    },
  ]
  const menuFaltasDisciplinares = [
    {
      items: [
        {
          link: routes.faltasDisciplinares.new,
          text: "Cadastrar faltas disciplinares"
        },
        {
          link: routes.faltasDisciplinares.search,
          text: "Buscar faltas disciplinares"
        }
      ]
    },
  ]

  return (
    <div className="fixed flex flex-col gap-4 w-[300px] min-w-[300px] border-r min-h-screen p-4">
      <div>
        <UserItem />
      </div>
      <ScrollArea className="h-[35rem] w-[282px] pr-4">
        <div className="grow">
          <section className="flex flex-col gap-2  mt-2 mb-6">
            <Link href={routes.home.home}><p className="flex gap-2 items-center text-sm hover:cursor-pointer hover:bg-muted p-1 rounded-sm"><House size={18} />Inicio</p></Link>
            <Link href={routes.chamada.home}><p className="flex gap-2 items-center text-sm hover:cursor-pointer hover:bg-muted p-1 rounded-sm"><Table size={18} />Chamada</p></Link>
          </section>

          <h1 className="text-sm text-muted-foreground">Cadastros</h1>
          <Accordion type="single" collapsible>
            <AccordionItem value="item-1">
              <AccordionTrigger className="text-sm">Aluno</AccordionTrigger>
              <AccordionContent>
                <Command style={{ overflow: 'visible' }}>
                  <CommandList style={{ overflow: 'visible' }}>
                    {menuStudant.map((menu: any, key: number) => (
                      <CommandGroup key={key} heading={menu.group}>
                        {menu.items.map((option: any, optionKey: number) =>
                          <CommandItem key={optionKey} className="flex justify-between cursor-pointer">
                            <Link href={option.link}>{option.text}</Link>
                          </CommandItem>
                        )}
                      </CommandGroup>
```

Figura 14: Código Menu Lateral

Fonte: Autoria própria.

4.3. PÁGINA INICIAL

A página inicial do software oferece uma visão geral das principais informações educacionais. Nela, o usuário encontra campos que indicam quantos alunos, professores e cursos estão cadastrados. Além disso, a página inicial inclui atalhos diretos para a matrícula de novos alunos e para a realização de chamadas, facilitando o acesso rápido às funções mais utilizadas no sistema.

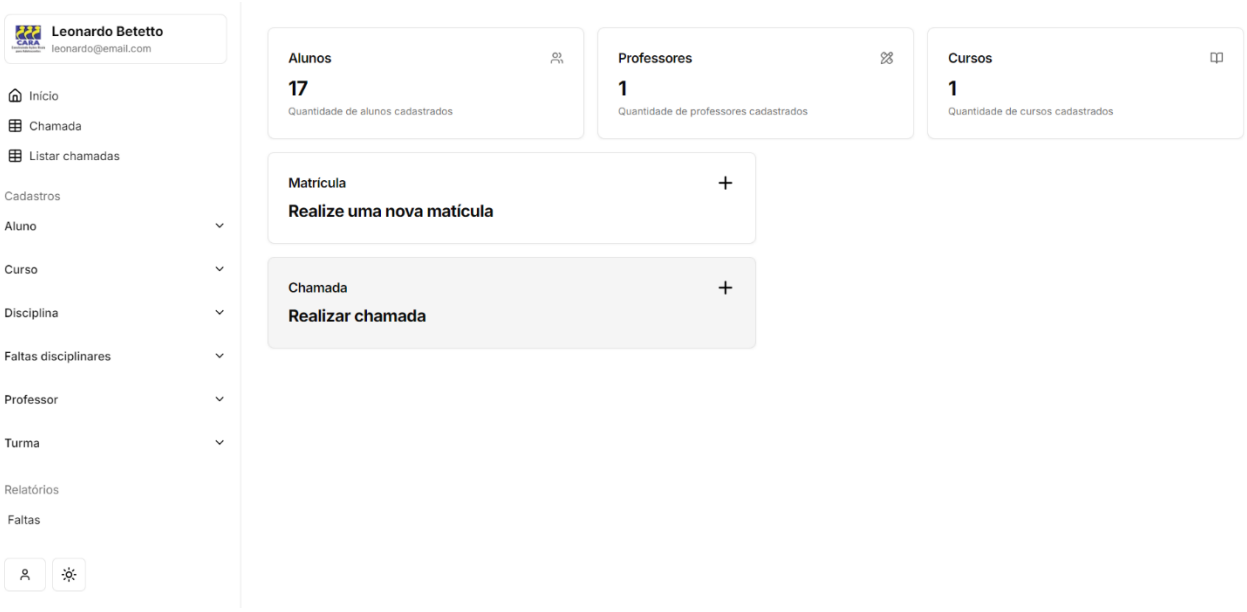


Figura 15: Página inicial

Fonte: Autoria própria.

```
import CardChamada from "./components/button-chamada";
import CourseCountCard from "./components/course-count-card";
import CardAddStudent from "./components/new-add-student";
import StudentCountCard from "./components/student-count-card";
import TeacherCountCard from "./components/teacher-count-card";

export default function HomeGestor() {
    return (
        <main className="flex flex-col gap-4">
            <div className="grid grid-cols-3 gap-4">
                <StudentCountCard />
                <TeacherCountCard />
                <CourseCountCard />
            </div>
            <CardAddStudent />
            <CardChamada />
        </main>
    )
```

Figura 16: Código Página inicial

Fonte: Autoria própria.

4.4. CADASTRO

A tela de cadastro é essencial para o sistema, permitindo o registro de professores, alunos, cursos e turmas de forma simples e organizada. Ela foi projetada para ser fácil de usar, facilitando a inserção das informações necessárias. Cada área de cadastro é adaptada para as necessidades específicas, garantindo que todos os dados sejam registrados corretamente, ajudando a manter o sistema sempre atualizado.

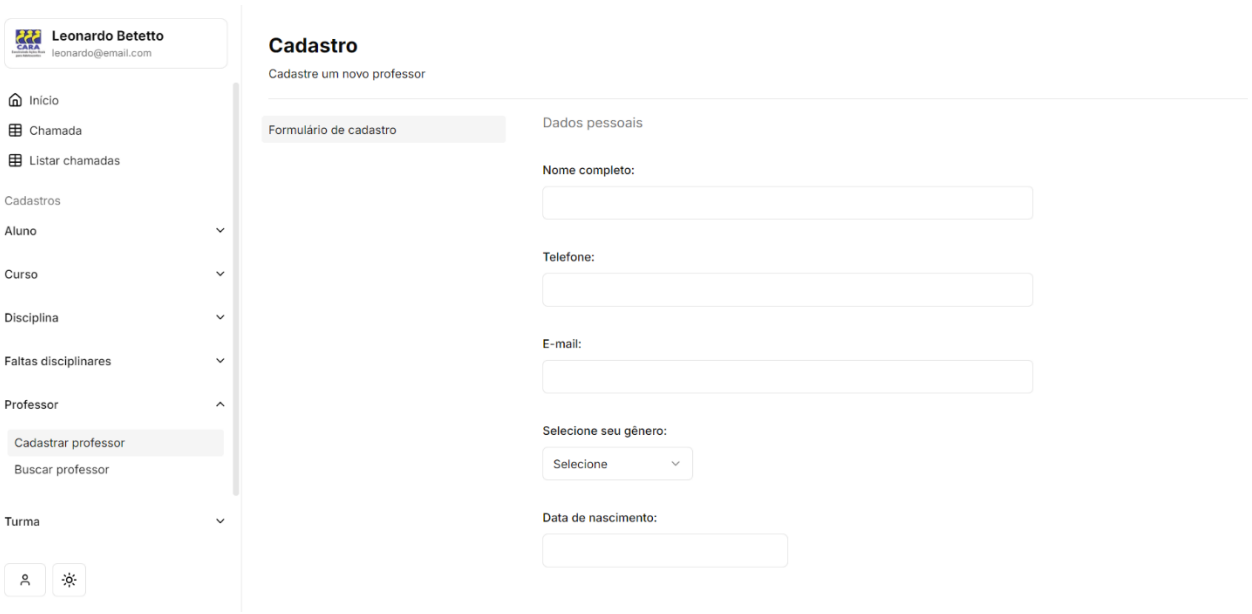


Figura 17: Cadastro

Fonte: Autoria própria.

```
      toast({
        title: "Sucesso!",
        description: "Professor criado com sucesso!",
      });

      reset();
    } catch (error) {
      toast({
        title: "Erro",
        description: "Ocorreu um erro ao criar o professor.",
        variant: "destructive",
      });
    }
  };

  return (
    <ScrollArea className="h-[34rem] w-[853px] pr-[250px]">
      <Form {...form}>
        <form onSubmit={form.handleSubmit(onSubmit)} className="space-y-8 p-2 pt-0">
          <h1 className="text-m text-muted-foreground">Dados pessoais</h1>
          <FormField
            control={form.control}
            name="teacherName"
            render={({ field }) => (
              <FormItem>
                <FormLabel>Nome completo:</FormLabel>
                <FormControl>
                  <Input {...field} />
                </FormControl>
                <FormMessage />
              </FormItem>
            )}
          />
          <FormField
            control={form.control}
            name="phone"
            render={({ field }) => (
              <FormItem>
                <FormLabel>Telefone:</FormLabel>
                <FormControl>
                  <Input {...field} type="tel" />
                </FormControl>
                <FormMessage />
              </FormItem>
            )}
          />
```

Figura 18: Código Cadastro Professor

Fonte: Autoria própria.

4.5. LISTAGEM

A tela de listagem permite visualizar todos os cadastros feitos no sistema, com as informações mais importantes exibidas de forma clara e acessível. Além disso, ela oferece botões de ação que permitem excluir um cadastro ou visualizar mais detalhes, facilitando a gestão e o controle de todos os registros de forma prática e eficiente.

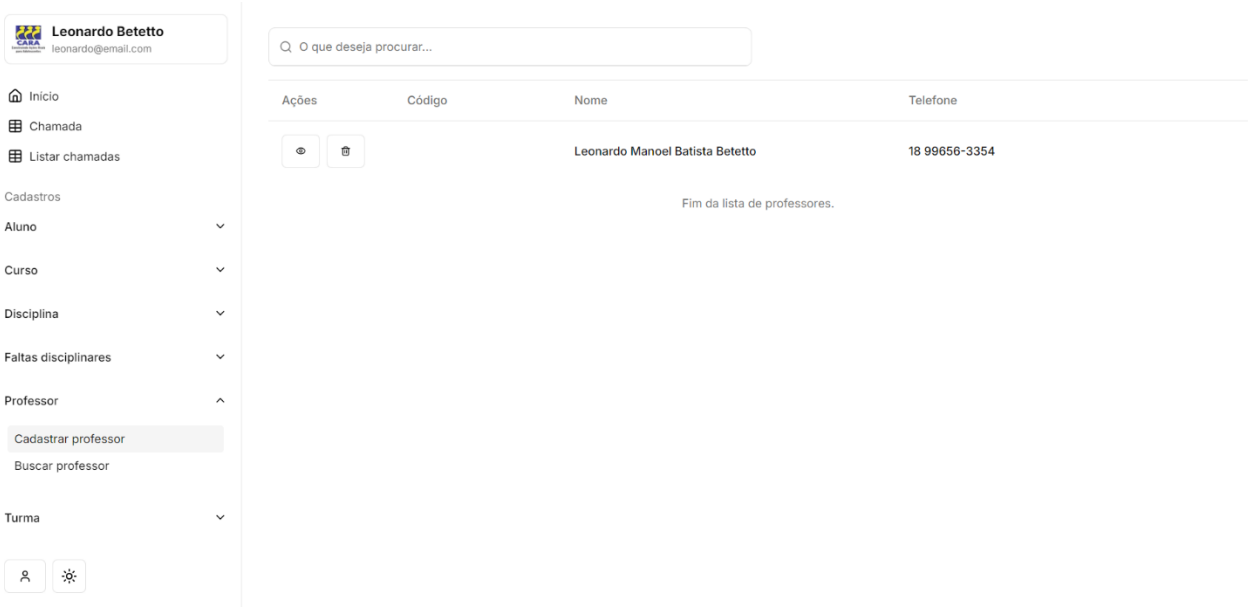


Figura 19: Listagem dos cadastros

Fonte: Autoria própria.

```
export function TeacherRows() {
            {professors.map((professor) => (
                                                    <span>{professor.curso}</span>
                                                    <span>{professor.instituicao}</span>
                                                    <span>Concluído em {professor.anoConclusao}</span>
                                                </div>
                                            </div>
                                        </DialogDescription>
                                        <DialogFooter>
                                            <Link href={routes.professor.edit(professor.id)}>
                                                <Button variant={"outline"}>Editar</Button>
                                            </Link>
                                        </DialogFooter>
                                    </DialogContent>
                                </Dialog>

                                <AlertDialog>
                                    <AlertDialogTrigger asChild>
                                        <Button variant="outline">
                                            <Trash2 className="h-3 w-3" />
                                            <span className="sr-only">Excluir professor</span>
                                        </Button>
                                    </AlertDialogTrigger>
                                    <AlertDialogContent>
                                        <AlertDialogHeader>
                                            <AlertDialogTitle>Excluir professor?</AlertDialogTitle>
                                            <AlertDialogDescription>
                                                Esta ação não poderá ser desfeita. Excluindo este professor, ele será removido permanentemente do sistema!
                                            </AlertDialogDescription>
                                        </AlertDialogHeader>
                                        <AlertDialogFooter>
                                            <AlertDialogCancel>Cancelar</AlertDialogCancel>
                                            <AlertDialogAction onClick={() => handleDeleteProfessor(professor.id)}>Excluir</AlertDialogAction>
                                        </AlertDialogFooter>
                                    </AlertDialogContent>
                                </AlertDialog>
                            </div>
                        </DialogTrigger>
                    </Dialog>
                </TableCell>
                <TableCell className="font-mono text-xs font-medium">
                    {professor.codeProfessor}
                </TableCell>
                <TableCell className="font-medium">
                    {professor.name}
                </TableCell>
                <TableCell className="font-medium">
                    {professor.telefone}
                </TableCell>
            </TableRow >
        ))
        }
```

Figura 20: Código Listagem Professor

Fonte: Autoria própria.

4.6. VISUALIZAÇÃO

Na tela de visualização detalhada do cadastro o usuário tem acesso completo a todas as informações registradas. Um modal será aberto, exibindo os dados de forma organizada e fácil de consultar. Além disso, há um botão disponível para editar as informações diretamente, permitindo que sejam realizados ajustes necessários de maneira rápida e eficiente.

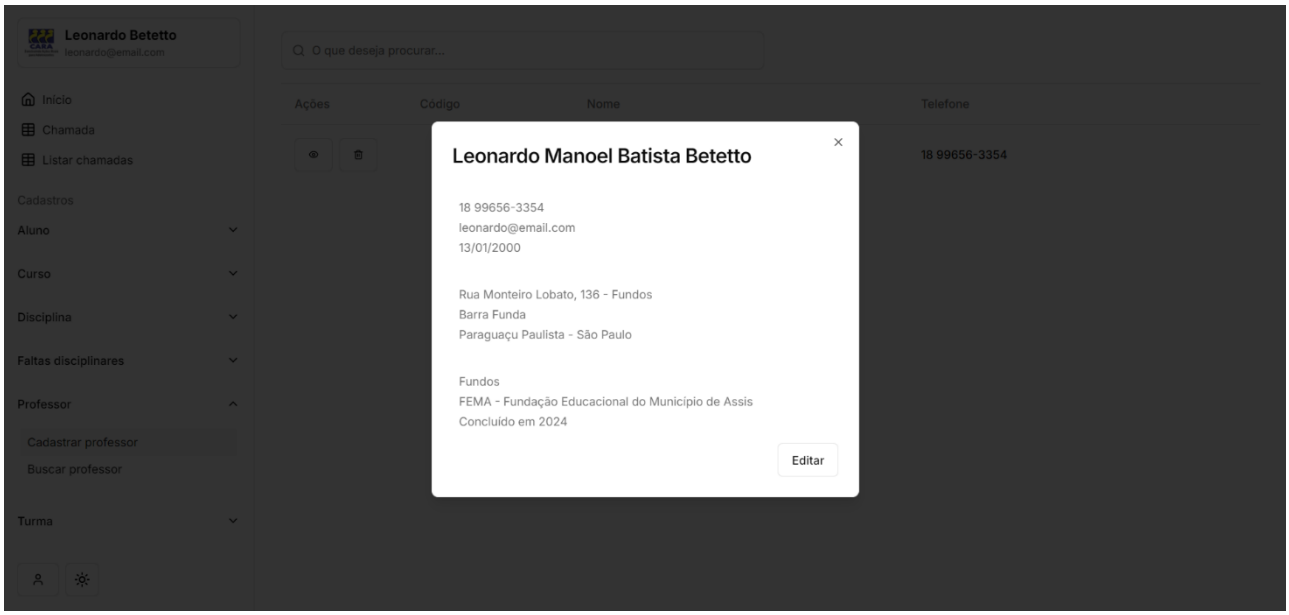


Figura 21: Visualização dos dados

Fonte: Autoria própria.

4.7. EDIÇÃO

A tela de edição permite a atualização dos dados já cadastrados no sistema. Ao acessá-la, o usuário é levado a uma interface semelhante à de criação, onde todos os campos estarão preenchidos com as informações existentes. Aqui, é possível alterar qualquer campo disponível, garantindo que os dados estejam sempre atualizados e corretos de acordo com as necessidades atuais.

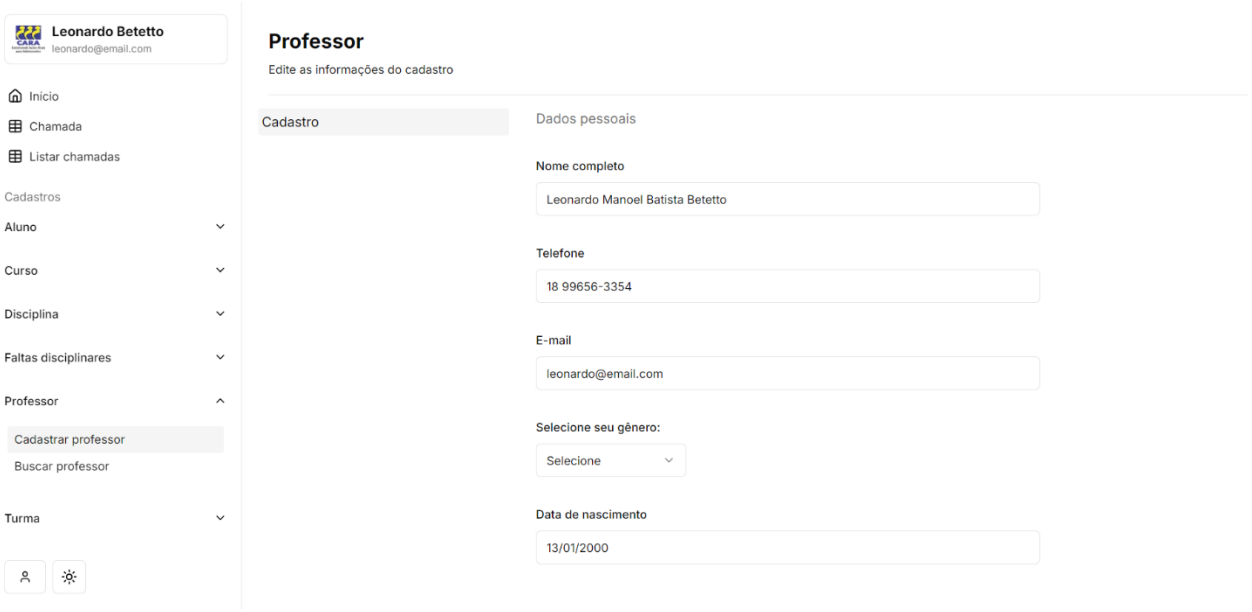


Figura 22: Edição de cadastro

Fonte: Autoria própria.

```
useEffect(() => {
    if (id) {
        fetchUserData(id as string)
            .then((data) => {
                form.reset({
                    teacherName: data.name,
                    phone: data.telefone,
                    email: data.email,
                    gender: data.gender ?? '',
                    nascimento: data.nascimento,
                    rua: data.rua,
                    numero: data.numero,
                    complemento: data.complemento,
                    bairro: data.bairro,
                    cidade: data.cidade,
                    estado: data.estado,
                    course: data.curso,
                    instituicao: data.instituicao,
                    conclusion: data.anoConclusao,
                    nivelFormacao: data.nivelFormacao ?? ''
                });
            })
            .catch((error) => console.error("Error fetching user data:", error));
    } else {
        console.error("ID não encontrado na URL");
    }
}, [form, id]);

const onSubmit = async (data: Schema) => {
    if (id) {
        console.log('teste')
        try {
            const payload: TeacherPayload = {
                teacherName: data.teacherName,
                phone: data.phone,
                email: data.email,
                nascimento: data.nascimento,
                rua: data.rua,
                numero: data.numero,
                complemento: data.complemento,
                bairro: data.bairro,
                cidade: data.cidade,
                estado: data.estado,
                course: data.course,
                instituicao: data.instituicao,
                conclusion: data.conclusion,
                gender: data.gender || '',
                nivelFormacao: data.nivelFormacao || ''
            };
```

Figura 23: Código Listagem Professor

Fonte: Autoria própria.

4.8. PERFIL DO USUÁRIO

A tela de visualização dos dados do usuário logado permite que o usuário veja todas as suas informações pessoais armazenadas no sistema de forma clara e organizada. Nessa tela, todos os dados são apresentados de maneira acessível, facilitando a consulta.

Além de visualizar seus dados, a tela oferece a opção de editá-los diretamente. Com botões de ação, você pode alterar qualquer informação necessária, garantindo que seus dados estejam sempre atualizados.

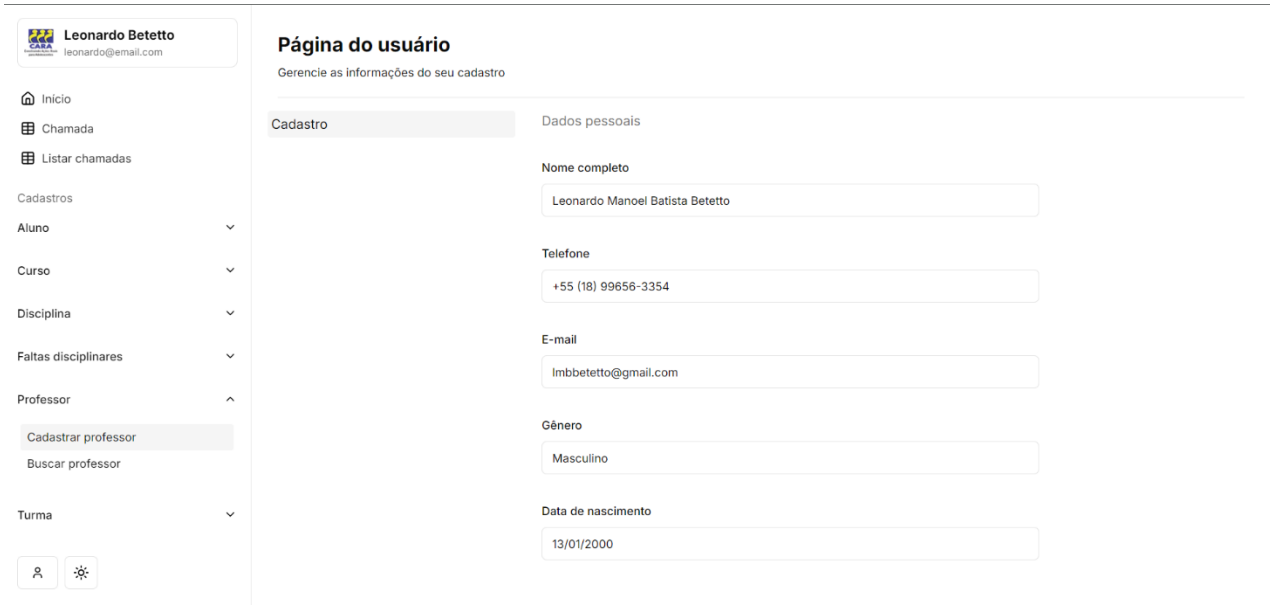


Figura 24: Visualização e edição de usuário

Fonte: Autoria própria.

5. CONCLUSÃO DO TRABALHO

O presente trabalho teve como objetivo construir uma aplicação *WEB* que atendesse as necessidades de um SCFV, em específico o Projeto CARA, permitindo que se tenham maior controle sobre os processos internos, proporcionando maior segurança e maior eficiência no gerenciamento de recursos.

O sistema foi desenvolvido utilizando diversas tecnologias como *TypeScript*, *Next.Js*, *Tailwind, Node.Js* e *PostgreSQL*.

Após uma análise da aplicação desenvolvida, é possível concluir que todos os objetivos propostos pelo presente trabalho foram concluídos.

5.1. TRABALHOS FUTUROS

Durante o desenvolvimento deste trabalho, foram identificadas possíveis funcionalidades que podem ser implementadas em trabalhos futuros, tais como: implementação de gráficos que facilitem na visualização e análise de alguns dados e uma aplicação onde seja possível que os alunos acompanhem suas faltas.

REFERÊNCIAS

FRANKLIN, Matheus Mião; SAMUEL FILHO, Ronaldo Aparecido. **Desenvolvimento de um Sistema de Gestão Escolar com o uso da Linguagem Dart com Framework Flutter**. In: Revista e-F@tec, n.1, 2020, Garça, Brasil, v.10, out, 10p.

IBM DOCUMENTATION. **Diagramas de Caso de Uso**. [*S. l.*], 2021. Disponível em: https://www.ibm.com/docs/pt-br/rsm/7.5.0?topic=diagrams-use-case. Acesso em: 13 mar. 2024.

LUCIDCHART. **O que é um diagrama entidade relacionamento?** [*S. l.*], [20--?]. Disponível em: https://www.lucidchart.com/pages/pt/o-que-e-diagrama-entidade-relacionamento. Acesso em: 13 mar. 2024.

MARCHIORI, Lucas. **Next JS: o que é, para que serve e por que usar?** [*S. l.*], 27 jan. 2023. Disponível em: https://blog.betrybe.com/tecnologia/next-js/#1. Acesso em: 12 fev. 2024.

MEDEIROS, Juliana. **SCFV: Tudo o que você precisa saber sobre o Serviço de Convivência e Fortalecimento de Vínculos**. Gesuas, 02 jun. 2023. Disponível em: https://blog.gesuas.com.br/scfv/. Acesso em: 20 out. 2023.

MONTEIRO, Robson Luiz; LARA, Silvana Maria Affonso de. **Utilização de aplicativo móvel como ferramenta de apoio educacional**. In: WORKSHOP DE INOVAÇÃO, PESQUISA, ENSINO E EXTENSÃO, 2., 2016, São Carlos, SP. Anais... São Carlos, SP: IFSP, 2016. p. 121-124.

NEVES, Vinicios. **React: o que é, como funciona e um guia dessa popular ferramenta JS**. [*S. l.*], 17 jan. 2023. Disponível em: https://www.alura.com.br/artigos/react-js#o-que-e-react-js?. Acesso em: 12 fev. 2024.

OLIVEIRA, Renan Gomes de; OLIVEIRA, Eduardo Gomes de. **Desenvolvimento e Avaliação de um Aplicativo para Gestão de Frequência Escolar**. 2020, Catagueses, MG. p. 1-29.

PMI - PROJECT MANAGEMENT INSTITUTE. **Guia PMBOK®: Um Guia para o Conjunto de Conhecimentos em Gerenciamento de Projetos, sétima edição**, Pennsylvania: PMI, 2021.