

**CENTRO ESTADUAL DE EDUCAÇÃO TECNOLÓGICA
CENTRO PAULA SOUZA**

ETEC ZONA LESTE

**Ensino Médio Com Habilitação Profissional Em
Técnico Em Desenvolvimento De Sistemas**

Layara Miranda de Campos

Mariana Ocireu de Souza

Nicole Milanez Oliveira

**BABYD: Sistema de Reconhecimento Facial Para
Segurança em Creches**

São Paulo

2024

Layara Miranda de Campos

Mariana Ocireu de Souza

Nicole Milanez Oliveira

**BABYD: Sistema de Reconhecimento Facial Para
Segurança em Creches**

Trabalho de Conclusão de Curso
apresentado ao Curso Técnico em
Desenvolvimento de Sistemas da Etec Zona
Leste, orientado pelo Prof. Jeferson Roberto de
Lima, como requisito parcial para obtenção do
título de técnico em Desenvolvimento de
Sistemas.

São Paulo

2024

Dedicamos esse projeto primeiramente a Deus, sem sua permissão nada disso seria possível. Também gostaríamos de dedicar aos nossos coordenadores Jeferson e Rogério, que tanto nos ajudaram com seus ensinamentos, conhecimento e incentivos no decorrer do ano. Ao professor Carlos, pelo apoio, conselhos e o ombro amigo. Aos nossos familiares pelo amor e compreensão, esperamos dá-los orgulho.

Agradecimentos

Agradecemos primeiramente a Deus por ter permitido que tivéssemos saúde e determinação para não desanimar durante a realização desse projeto, por não ter nos desamparado durante todos os anos de estudo e por nos conceder a honra da vida. Agradecemos imensamente aos amigos e familiares por todo o apoio e ajuda, que contribuíram diretamente ou indiretamente para a realização desse trabalho, agradecemos o incentivo nos momentos difíceis e a compreensão da nossa ausência enquanto estávamos empenhados na realização desse trabalho. Agradecemos ao professor Jeferson Roberto de Lima por ter sido nosso orientador e ter desempenhado tal função com dedicação e amizade. Obrigado pelas cobranças, correções minuciosas e paciência. Agradecemos também aqueles que contribuíram de alguma forma, aos nossos colegas de classe, com quem convivemos intensamente durante os últimos anos, aos mentores da IBM, que nos enriqueceram com suas devolutivas, e aos nossos queridos amigos pelo companheirismo e troca de experiências, que permitiram crescer não só como pessoa, mas também como aluno.

Resumo

Este trabalho aborda o tema de um sistema de reconhecimento facial para segurança em creches, que é composto por um aplicativo e um site expositivo. O objetivo desse trabalho é auxiliar na diminuição de problemas relacionados ao cotidiano desse ambiente. O processo de retirada das crianças da creche está envolto em uma série de problemas, os principais e mais preocupantes são os sequestros, confusões de crianças e invasões a instituição, que além de comprometer a integridade física das crianças também compromete a integridade moral e a reputação da creche. Para desenvolver o projeto, a metodologia a ser utilizada incluirá abordagens qualitativas e exploratórias. Os resultados que esperamos alcançar são a personalização no atendimento, otimização do tempo, diminuição de desentendimentos e ocorrências relacionadas a sequestros, além de promover uma maior colaboração e aprendizado tecnológico entre gestores e professores, tornando o atendimento mais fácil e menos conflituoso. Em conclusão, a falta de protocolos rigorosos de segurança pode resultar em acidentes evitáveis e situações de risco, que o sistema busca amenizar.

Palavras-chave: Segurança. Creche. Instituição. Tecnologia. Sistema.

Abstract

This work addresses the theme of a facial recognition system for security in nurseries, which is composed of an application and an exhibition website. The main objective of this work is to assist in the reduction of problems related to the daily life of this environment. The process of removing children from the child garden is involved in a series of problems, the main and most worrying are kidnappings, confusion of children and invasions of the institution, which in addition to compromising the physical integrity of the children also compromises the moral integrity and reputation of the daycare. For that, the methodology to be used will include qualitative and exploratory approaches. As a result, we hope to achieve personalization in service, optimization of time, reduction of misunderstandings and occurrences related to kidnappings, in addition to promoting greater collaboration and technological learning between managers and teachers, making care easier and less conflictive. In conclusion, the lack of strict safety protocols can result in avoidable accidents and risky situations, which the system seeks to mitigate.

Keywords: Security. Nursery, Institution. Technology. System.

LISTA DE ABREVIATURAS E SIGLAS

Application Programming Interface (API)

Cascading Style Sheet (CSS)

Computer Aided Software Engineering (CASE)

Hypertext Markup Language (HTML)

JavaScript Object Notation (JSON)

JavaScript XML (JSX)

Red, Green, Blue (RGB)

Unified Modeling Language (UML)

Uniform Resource Locator (URL)

LISTA DE FIGURAS

| | |
|---|----|
| Figura 1 - Exemplo Código Básico HTML | 17 |
| Figura 2 - Resultado Código Básico HTML | 18 |
| Figura 3 - Exemplo Código Básico CSS..... | 19 |
| Figura 4 - Continuação Exemplo Código Básico CSS..... | 20 |
| Figura 5 - Resultado Exemplo Básico CSS | 21 |
| Figura 6 - Exemplo Código Básico JavaScript | 22 |
| Figura 7 - Resultado Exemplo Básico JavaScript..... | 23 |
| Figura 8 - Exemplo Wireframe Baixa Cadastro | 24 |
| Figura 9 - Exemplo Wireframe Baixa Login | 24 |
| Figura 10 - Exemplo Wireframe Alta Cadastro | 25 |
| Figura 11 - Exemplo Wireframe Alta Login..... | 26 |
| Figura 12 - Exemplo de codificação React Native | 27 |
| Figura 13 - Exemplo de codificação React Native Dois..... | 28 |
| Figura 14 - Exemplo de codificação React Native Três..... | 29 |
| Figura 15 - Emulação no App Expo..... | 29 |
| Figura 16 - Exemplo Código Básico React Native..... | 31 |
| Figura 17 - Continuação Um Código Básico React Native | 32 |
| Figura 18 - Continuação Dois Código Básico React Native | 33 |
| Figura 19 - Continuação Três Código Básico React Native | 34 |
| Figura 20 - Resultado Exemplo React Native..... | 35 |
| Figura 21 - Exemplo Visão Computacional | 36 |
| Figura 22 - Exemplo Reconhecimento de Características | 37 |
| Figura 23 - Exemplo Código NPM..... | 39 |
| Figura 24 - Segundo Exemplo Código NMP..... | 40 |
| Figura 25 - Wireframe de baixa: Home | 48 |
| Figura 26 -Wireframe de baixa: Sobre nós..... | 49 |
| Figura 27 - Wireframe de baixa: Objetivos | 49 |
| Figura 28 - Wireframe de baixa: Membros | 50 |
| Figura 29 - Wireframe de baixa: Galeria | 51 |
| Figura 30 - Wireframe de baixa: Contatos..... | 51 |
| Figura 31 - Wireframe de baixa: Footer..... | 52 |
| Figura 32 - Wireframe de alta: Home | 53 |
| Figura 33 - Wireframe de alta: Sobre nós | 53 |
| Figura 34 - Wireframe de alta: Objetivos..... | 54 |

| | |
|--|----|
| Figura 35 - Wireframe de alta: Membros | 55 |
| Figura 36 - Wireframe de alta: Galeria | 55 |
| Figura 37 - Wireframe de alta: Contatos | 56 |
| Figura 38 - Wireframe de alta: Footer | 56 |
| Figura 39 - Resultado do site | 57 |
| Figura 40 - Wireframe de baixa: Tela inicial | 58 |
| Figura 41 - Wireframe de baixa: Cadastro..... | 59 |
| Figura 42 - Wireframe de baixa: Login | 60 |
| Figura 43 - Wireframe de baixa: Histórico de Responsáveis | 61 |
| Figura 44 - Wireframe de baixa: Registro Realizado | 62 |
| Figura 45 - Wireframe de baixa: Editar Registro | 63 |
| Figura 46 - Wireframe de baixa: Termos de Uso..... | 64 |
| Figura 47 - Wireframe de alta: Tela inicial | 65 |
| Figura 48 - Wireframe de alta: Cadastro | 66 |
| Figura 49 - Wireframe de alta: Login | 67 |
| Figura 50 - Wireframe de alta: Termos de Uso | 71 |
| Figura 51 - Diagrama de Caso de Uso | 73 |
| Figura 52 - Diagrama de Atividade: Atualizar Responsáveis | 78 |
| Figura 53 - Diagrama de Atividade: Autenticar Saída | 78 |
| Figura 54 - Diagrama de Atividade: Cadastrar Responsáveis..... | 79 |
| Figura 55 - Diagrama de Atividade: Escanear Rosto do Responsável...79 | |
| Figura 56 - Diagrama de Atividade: Fazer Login | 80 |
| Figura 57 - Diagrama de Atividade: Notificar Saída do Aluno | 80 |
| Figura 58 - Diagrama de Atividade: Visualizar Responsáveis | 81 |
| Figura 59 - Diagrama de Sequência: Atualizar Responsáveis | 81 |
| Figura 60 - Diagrama de Sequência: Cadastrar Responsáveis..... | 82 |
| Figura 61 - Diagrama de Sequência: Escanear Rosto do Responsável.83 | |
| Figura 62 - Diagrama de Sequência: Fazer Login..... | 83 |
| Figura 63 - Diagrama de Sequência: Notificar Saída do Aluno | 84 |
| Figura 64 - Diagrama de Sequência: Visualizar Responsáveis..... | 85 |
| Figura 65 - Diagrama de Máquina de Estado: Fazer login | 85 |
| Figura 66 - Diagrama de Máquina de Estado: Escanear rosto do responsável..... | 86 |
| Figura 67 - Diagrama de Máquina de Estado: Visualizar responsáveis .86 | |
| Figura 68 - Diagrama de Máquina de Estado: Autenticar saída | 87 |

| | |
|---|----|
| Figura 69 - Diagrama de Máquina de Estado: Fazer cadastro | 87 |
| Figura 70 - Diagrama de Máquina de Estado: Notificar saída do aluno . | 88 |
| Figura 71 - Diagrama de Máquina de Estado: Atualizar responsáveis ... | 88 |
| Figura 72 - Código Reconhecimento Facial | 89 |

LISTA DE QUADROS

| | |
|---|----|
| Quadro 1 - Descrição do caso de uso: Fazer cadastro | 73 |
| Quadro 2 - Descrição do caso de uso: Fazer login | 74 |
| Quadro 3 - Descrição do caso de uso: Atualizar Responsáveis..... | 75 |
| Quadro 4 - Descrição do caso de uso: Escanear Rosto do Responsável | 75 |
| Quadro 5 - Descrição do caso de uso: Solicitar Saída do Aluno | 76 |
| Quadro 6 - Descrição do caso de uso: Autenticar Saída..... | 77 |

SUMÁRIO

| | | |
|-------------|--|----|
| 1..... | INTRODUÇÃO | 13 |
| 1.1..... | A falta de medidas de segurança eficazes nas creches | 15 |
| 1.2.... | Como a tecnologia pode ajudar na segurança das creches | 15 |
| 1.3.... | O reconhecimento facial no ambiente educacional infantil | 15 |
| 2..... | REFERENCIAL TEÓRICO | 16 |
| 2.1..... | HTML | 16 |
| 2.2..... | CSS | 18 |
| 2.3..... | JavaScript | 21 |
| 2.4..... | Visual Studio Code | 23 |
| 2.5..... | Wireframe | 23 |
| 2.6..... | Figma | 26 |
| 2.7..... | Expo | 26 |
| 2.8..... | React | 30 |
| 2.9..... | Framework | 30 |
| 2.10..... | React Native | 30 |
| 2.11..... | Visão Computacional | 35 |
| 2.12..... | OpenCV | 36 |
| 2.13..... | Banco de Dados | 37 |
| 2.14..... | Firebase e Firestores | 38 |
| 2.15..... | NPM | 38 |
| 2.16..... | Node.js | 40 |
| 2.17..... | UML | 40 |
| 2.17.1..... | Diagrama de Caso de Uso | 41 |
| 2.17.2..... | Diagrama de Atividade | 42 |
| 2.17.3..... | Diagrama de Classe | 43 |
| 2.17.4..... | Diagrama de Sequência | 44 |
| 2.18..... | Astah Community | 46 |
| 3..... | DESENVOLVIMENTO | 47 |
| 3.1..... | Sistema BabyD | 47 |
| 3.2..... | Vitrine Web do Projeto BabyD | 47 |
| 3.3..... | Aplicação Móvel do Projeto BabyD | 58 |
| 4..... | CONCLUSÃO | 90 |

1. INTRODUÇÃO

O presente estudo aborda a aplicação da tecnologia de reconhecimento facial em creches por meio de um aplicativo móvel, com o intuito de aprimorar a segurança das crianças e facilitar o controle de acesso para pais e responsáveis. A pesquisa focará na viabilidade técnica, legal e ética dessa solução, levando em consideração o contexto específico das instituições de educação infantil.

A justificativa para este estudo baseia-se na constante preocupação com a segurança das crianças em creches, compartilhada por pais, educadores e gestores. A tecnologia de reconhecimento facial oferece um potencial significativo para aprimorar os sistemas de controle de acesso, reduzindo o risco de acesso não autorizado e facilitando a identificação em situações de emergência. No entanto, a sensibilidade do ambiente exige uma análise criteriosa dos aspectos legais, éticos e pedagógicos envolvidos.

O problema central reside na falta de medidas de segurança eficazes em creches, o que pode aumentar o risco de incidentes como sequestros, abduções ou acessos não autorizados, ameaçando o bem-estar das crianças e a reputação da instituição. Como exemplo, Caldas (2023) relatou o caso de uma criança que foi confundida, entregue a responsável errado e causou tumulto em 3 cidades de Santa Catarina. A questão de pesquisa que orienta este estudo é: como a implementação de um sistema de reconhecimento facial em creches pode contribuir para a segurança das crianças, ao mesmo tempo em que assegura a privacidade e os direitos dos envolvidos?

A hipótese deste estudo diz respeito a implementação de um sistema de reconhecimento facial em creches, alinhado a regulamentação pertinente ao nicho e cuidadosamente delimitado, pode garantir o respeito aos direitos de privacidade e à proteção de dados.

O objetivo geral deste estudo é avaliar a aplicabilidade e os impactos da tecnologia de reconhecimento facial em creches, considerando aspectos técnicos, legais, éticos e pedagógicos, com o intuito de propor um

modelo que garanta a segurança das crianças sem infringir direitos fundamentais.

Para alcançar esse objetivo, foram delineados algumas metas específicas, das quais fazem parte para realizar um levantamento bibliográfico sobre o uso de reconhecimento facial em diferentes contextos, com foco em ambientes escolares; analisar a legislação brasileira e as normas de segurança vigentes para instituições de educação infantil, identificando as implicações do uso de sistemas de reconhecimento facial; Identificar os benefícios e os riscos associados à implementação dessa tecnologia em creches, considerando as perspectivas de pais, educadores, gestores e crianças; propor um modelo de sistema de reconhecimento facial adaptado às necessidades das creches, respeitando a privacidade e os direitos das crianças e de seus responsáveis.

A pesquisa terá um caráter exploratório e descritivo, utilizando uma abordagem qualitativa. As técnicas de coleta de dados incluirão: revisão bibliográfica; entrevistas; análise documental; estudo de caso.

Para a execução do projeto, utilizaremos a bibliografia de autores consagrados como base teórica, orientando-nos por seus conceitos em busca de excelência. Na construção do aplicativo, recorreremos à ferramenta Expo, mencionada por Fuentes (2023). A estrutura da linguagem de programação do referido aplicativo, React Native, será fundamentada nas ideias de Sereno (2018). O conceito central que norteará as funcionalidades do sistema será a Visão Computacional, conforme descrito por Milano e Bazorro (2010). No que diz respeito à biblioteca OpenCV, cuja finalidade é realizar a combinação de pares de características, seguiremos as diretrizes de Barelli (2018). O banco de dados e a plataforma que conectaremos ao sistema serão definidos com base na documentação oficial do Firebase (2023). Para o desenvolvimento do back-end, utilizaremos o Node.js, como explicado por Moraes (2017).

1.1. A falta de medidas de segurança eficazes nas creches

A falta de medidas de segurança em creches é uma preocupação crescente, pois compromete a integridade física e emocional das crianças, como mostrado por Nogueira (2023), onde Davi Guilherme Gomes de 3 anos, foi levado da creche por um desconhecido em Araçatuba. A ausência de protocolos rigorosos, como controle de acesso, treinamento adequado dos funcionários e manutenção das instalações, pode resultar em acidentes evitáveis e situações de risco, como abduções, confusões de crianças entre outros. É essencial que as creches implementem normas de segurança robustas para garantir um ambiente protegido e confiável, onde os pais possam deixar seus filhos com tranquilidade.

1.2. Como a tecnologia pode ajudar na segurança das creches

E para suprir a falta desses recursos e normas, a tecnologia pode desempenhar um papel crucial na segurança das creches, embora ainda seja raro encontrar dispositivos avançados nesses ambientes. A implementação de câmeras de vigilância, sistemas de controle de acesso e alarmes de emergência pode monitorar continuamente as atividades, restringir a entrada de pessoas não autorizadas e proporcionar respostas rápidas em situações de perigo. Além disso, ferramentas de comunicação instantânea permitem que os cuidadores se comuniquem rapidamente com os pais e autoridades em caso de emergências, garantindo um ambiente mais seguro para as crianças.

Vemos a grande importância da tecnologia segurança de escolas. Entre elas, destaca-se a promoção de um ambiente mais seguro, criando um clima favorável à aprendizagem e ao desenvolvimento dos alunos. Além disso, a utilização de tecnologias avançadas, como a inteligência e reconhecimento facial, torna a gestão da escola mais prática e eficiente (Eustáquio, 2019).

1.3. O reconhecimento facial no ambiente educacional infantil

O reconhecimento facial no ambiente educacional infantil é uma tecnologia emergente que visa aumentar a segurança e a eficiência das escolas. Ele pode ser utilizado para monitorar a saída dos alunos, garantindo que apenas pessoas autorizadas acessem as instalações e possam retirar as crianças da instituição. Além disso, essa tecnologia pode ajudar a automatizar a fiscalização de presença, economizando tempo e reduzindo erros. Com um

sistema de reconhecimento facial, os pais participaram e acompanharam de forma mais ativa e visível a proteção e os cuidados diários de seus filhos, com um simples cadastro através de sua face.

A proteção de instituições é uma questão fundamental para garantir um ambiente de aprendizado saudável e seguro. Nesse contexto, o investimento em tecnologias, como câmeras de vigilância, rastreabilidade e controle de acesso por meio de reconhecimento facial, vem se mostrando recursos eficientes no combate à violência (Ristow, 2019).

2. REFERENCIAL TEÓRICO

Neste capítulo será documentado o embasamento teórico, apresentando e descrevendo as tecnologias, ferramentas e conceitos utilizados para o desenvolvimento do projeto de conclusão.

2.1.HTML

Segundo Flatschart (2011), *Hyper Text Markup Language* é a base da *web*, uma linguagem de marcação de hipertexto, definindo como será a estrutura de uma página *web*.

De acordo com Torres (2018) o HTML foi criado para simplificar a transmissão de informações. É imprescindível deixar claro que o HTML não é uma linguagem de programação, mas sim de marcação.

Em concordância com Samy (2008) O hipertexto tem como o objetivo de interligar a outros documentos da *web*. O que torna isso possível são os *links*, que estão presentes nas páginas *web* que costumamos visitar na internet.

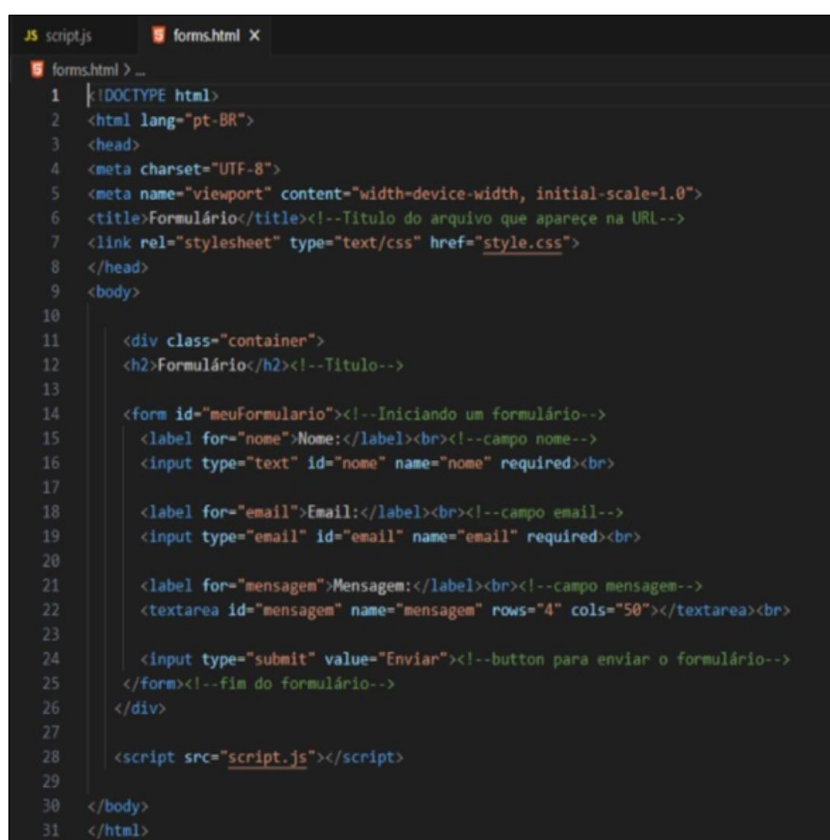
Conforme Duckett (2016) a base para uma criação de páginas, são as *tags* ou elementos, de modo que o navegador saiba o que você deseja colocar. As principais tags do HTML são:

- **head:** Obtém as informações da página e encontra-se a *tag*, cujo nome é *title*, dentro da mesma *tag*.
- **title:** Determinado por definir o título da página e mostrar na parte superior do navegador, onde digita-se a URL da página que deseja visitar.
- **link:** É a função existente no HTML que permite inserir os hiperlinks dentro de elementos no código, por exemplo, imagens e vídeos.

- **body**: É responsável por mostrar todo o conteúdo que está dentro da janela principal do navegador.
- **form**: A função de criar um bloco para que *tags* de formulário e seus dados sejam agrupados e encaminhados para outro documento, em conjunto.
- **main**: Serve para a definir o conteúdo principal dentro do *body*.

A seguir, a figura abaixo mostra um exemplo simples de código-fonte HTML da construção de um formulário.

Figura 1 - Exemplo Código Básico HTML

A screenshot of a code editor with a dark theme. The editor has two tabs at the top: 'script.js' and 'forms.html'. The 'forms.html' tab is active, showing the following HTML code:

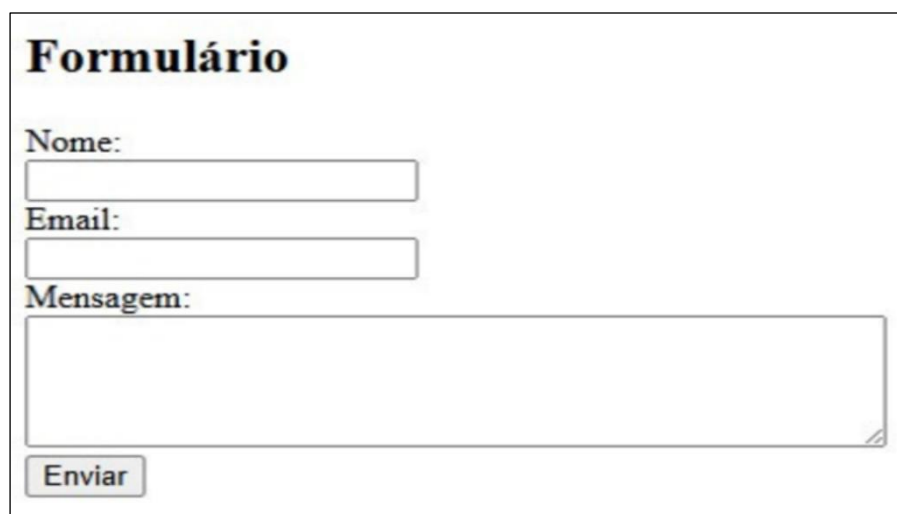
```
1 <!DOCTYPE html>
2 <html lang="pt-BR">
3 <head>
4 <meta charset="UTF-8">
5 <meta name="viewport" content="width=device-width, initial-scale=1.0">
6 <title>Formulário</title><!--Título do arquivo que aparece na URL-->
7 <link rel="stylesheet" type="text/css" href="style.css">
8 </head>
9 <body>
10
11 <div class="container">
12 <h2>Formulário</h2><!--Título-->
13
14 <form id="meuformulario"><!--Iniciando um formulário-->
15 <label for="nome">Nome:</label><br><!--campo nome-->
16 <input type="text" id="nome" name="nome" required><br>
17
18 <label for="email">Email:</label><br><!--campo email-->
19 <input type="email" id="email" name="email" required><br>
20
21 <label for="mensagem">Mensagem:</label><br><!--campo mensagem-->
22 <textarea id="mensagem" name="mensagem" rows="4" cols="50"></textarea><br>
23
24 <input type="submit" value="Enviar"><!--button para enviar o formulário-->
25 </form><!--fim do formulário-->
26 </div>
27
28 <script src="script.js"></script>
29
30 </body>
31 </html>
```

Fonte: Autoria Própria, 2024.

Na figura 1, mostra um exemplo de formulário simples em HTML, que foi criado com três campos: Nome, Email e Mensagem. No código, o atributo 'action' do formulário especifica o URL onde os dados serão enviados quando o formulário for enviado. Cada campo de entrada tem um 'id' e um 'name' atribuídos para identificação e envio dos dados, já o atributo 'required' faz com que os campos sejam obrigatórios, isto é, o formulário não poderá ser enviado se algum dos campos estiver vazio. O último elemento é um botão de envio do formulário `<input type="submit">`.

A figura abaixo demonstrará o resultado da codificação sem a estilização:

Figura 2 - Resultado Código Básico HTML

A imagem mostra um formulário web simples, sem qualquer estilização CSS. O formulário é contido dentro de uma caixa retangular com uma borda preta. No topo, o título "Formulário" está em uma fonte padrão, provavelmente sans-serif. Abaixo do título, há três campos de entrada: "Nome:" seguido de um campo de texto único linha; "Email:" seguido de um campo de texto único linha; e "Mensagem:" seguido de um campo de texto de múltiplas linhas. No canto inferior esquerdo do formulário, há um botão com o texto "Enviar". Todos os elementos são alinhados à esquerda e o layout é funcional e direto.

Fonte: Autoria Própria, 2024.

A codificação em HTML tem como resultado uma página de formulário simples e com cores padrão do navegador em que foi aberto, a forma com que os elementos são apresentados também seguem o padrão do navegador.

2.2.CSS

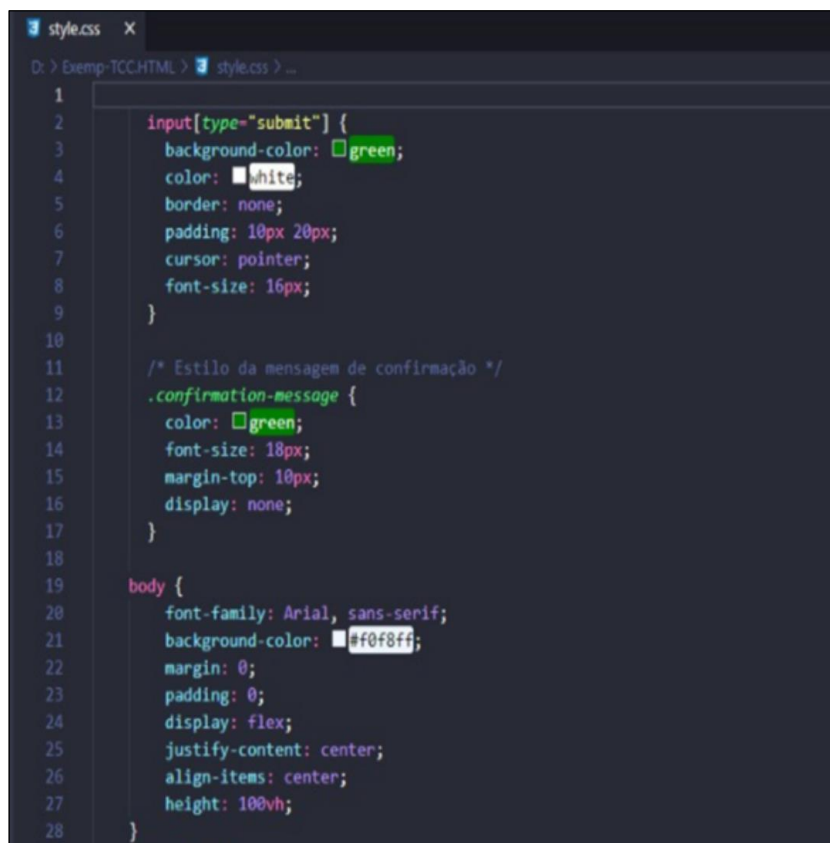
De acordo com Jobstraibizer (2009) o termo CSS, proveniente de *Cascading Style Sheets*, significa folhas de estilo em cascata. É utilizado a apresentação de documentos escritos em uma linguagem de marcação, como por exemplo, o HTML.

Segundo Duckett (2016) o CSS é a estilização da sua página *web*, ela se divide em duas categorias: *Layout* e apresentação, que permite deixar as páginas mais agradáveis, controlando o projeto com a linguagem e criando regras de como o conteúdo deverá ser apresentado.

Para Quierelli (2013) essa linguagem serve para estilizar o conteúdo de páginas, e a sua cor de fundo, tipos de textos, organização dos conteúdos e as imagens.

A seguir, as figuras abaixo mostram um exemplo simples de código-fonte CSS.

Figura 3 - Exemplo Código Básico CSS

A screenshot of a code editor window titled 'style.css'. The editor shows CSS code for styling a submit button and a confirmation message. The code is as follows:

```
1  
2 input[type="submit"] {  
3     background-color: green;  
4     color: white;  
5     border: none;  
6     padding: 10px 20px;  
7     cursor: pointer;  
8     font-size: 16px;  
9 }  
10  
11 /* Estilo da mensagem de confirmação */  
12 .confirmation-message {  
13     color: green;  
14     font-size: 18px;  
15     margin-top: 10px;  
16     display: none;  
17 }  
18  
19 body {  
20     font-family: Arial, sans-serif;  
21     background-color: #f0f8ff;  
22     margin: 0;  
23     padding: 0;  
24     display: flex;  
25     justify-content: center;  
26     align-items: center;  
27     height: 100vh;  
28 }
```

Fonte: Autoria Própria, 2024.

Nesse primeiro bloco da figura 3, estão sendo estilizadas as tags 'body', 'container', 'h2', 'label' e 'input' utilizando os principais elementos que alteram cor, tamanho, margem, alinhamento e justificação.

Figura 4 - Continuação Exemplo Código Básico CSS

```
29 .container {  
30   background-color: #fff;  
31   padding: 20px;  
32   border-radius: 8px;  
33   box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);  
34   max-width: 400px;  
35   width: 100%;  
36   text-align: center;  
37 }  
38 h2 {  
39   margin-bottom: 20px;  
40   color: #333;  
41 }  
42 label {  
43   display: block;  
44   margin-bottom: 8px;  
45   color: #666;  
46   text-align: left;  
47 }  
48 input[type="text"],  
49 input[type="email"],  
50 textarea {  
51   width: 100%;  
52   padding: 10px;  
53   margin-bottom: 20px;  
54   border: 1px solid #ccc;  
55   border-radius: 4px;  
56   box-sizing: border-box;  
57 }  
58 input[type="submit"] {  
59   background-color: #4CAF50;  
60   color: white;  
61   padding: 10px 20px;  
62   border: none;  
63   border-radius: 4px;  
64   cursor: pointer;  
65   transition: background-color 0.3s ease;  
66 }  
67 input[type="submit"]:hover {  
68   background-color: #45a049;  
69 }
```

Fonte: Autoria Própria, 2024.

Nesse segundo bloco da figura 4. estão sendo estilizadas as tags 'textarea', 'input submit' e o hover do botão 'submit' utilizando os principais elementos que alteram cor, tamanho, margem, alinhamento e justificação.

A seguir, o resultado do código HTML estilizado com CSS.

Figura 5 - Resultado Exemplo Básico CSS

A imagem mostra um formulário web centralizado em uma tela de fundo azul clara. O formulário é um retângulo branco com uma borda sutil. No topo, o título "Formulário" está em negrito. Abaixo dele, há três campos de entrada: "Nome:" com um campo de texto, "Email:" com um campo de texto, e "Mensagem:" com um campo de texto maior. No rodapé do formulário, há um botão verde com o texto "Enviar" em branco.

Fonte: Autoria Própria, 2024.

A estilização em HTML atribui a página uma estilização deixando a tela mais bonita e agradável ao usuário.

2.3. JavaScript

De acordo com Flanagan (2011) o JavaScript é descrito como uma linguagem de programação *web* de alto nível e sofisticada, que utiliza dos métodos de orientação a objeto, servindo para quase todas as plataformas e dispositivos modernos.

Segundo Silva (2010), ressalta e acusa as finalidades da linguagem em funcionalidades, efeito e comportamento, completamente atrelado ao HTML ou outra linguagem de manipulação de dados para ser executado, como o Python e o PHP.

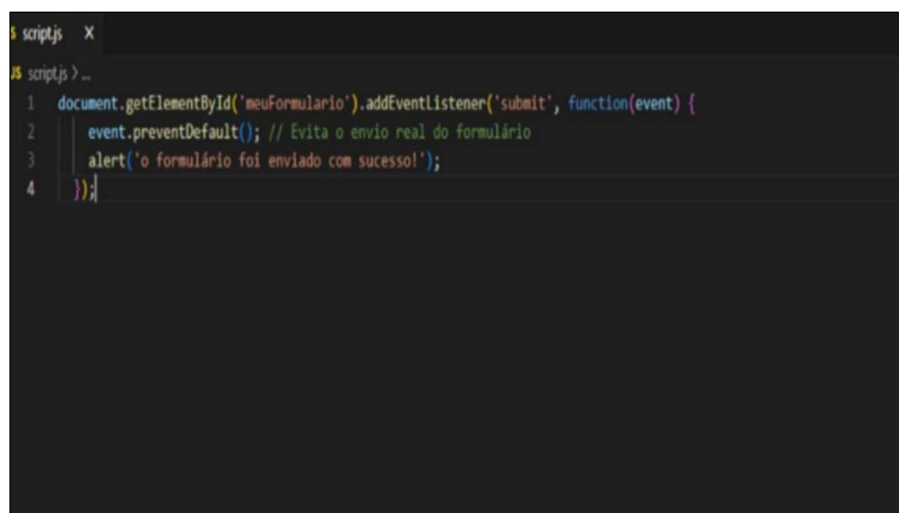
Em concordância com Milleto (2008), o Javascript é padronizado pela ECMA International (European Computer Manufacturers Association) nas especificações ECMA-262' e é considerada pela associação como uma das linguagens de efeito mais famosas e utilizadas.

Alguns dos principais comandos em JavaScript, incluem:

- Declaração de variáveis: 'var', 'let', 'const' são alguns dos comandos para declaração de variáveis.
- Funções: 'function' comando para definir funções.

- Operadores: '+', '-', '*', '/' utilizados em operações matemáticas; '==', '!=', '===' utilizados em comparações de variáveis.
 - Estruturas de controle: 'if', 'else', 'switch', 'for', 'while', 'do-while' comandos para controle de fluxo.
 - Arrays: '[]' para criar *arrays* e métodos como 'push()', 'pop()', 'slice()' para manipulá-los.
 - Objetos: '{}' para criar objetos e acessar suas propriedades usando a notação de ponto ou colchetes.
 - Manipulação do DOM: 'document.getElementById()', 'document.querySelector()', 'addEventListener()' para interagir com elementos HTML.
 - Temporizadores: 'setTimeout()', 'setInterval()' para executar código após um atraso ou em intervalos regulares.
 - Tratamento de erros: 'try', 'catch', 'throw' para lidar com exceções.
 - Expressões regulares: 'RegExp' para trabalhar com padrões de texto.
- A seguir, a imagem mostra um exemplo básico de JavaScript.

Figura 6 - Exemplo Código Básico JavaScript



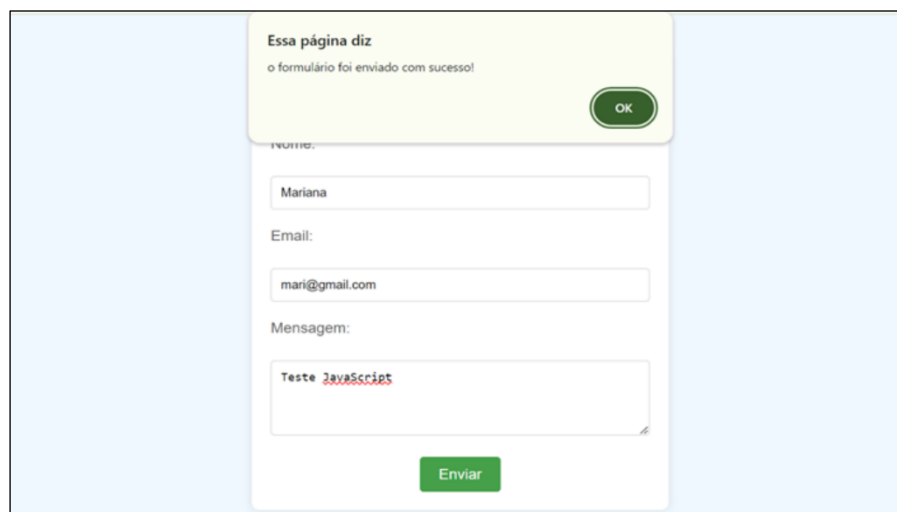
```
1 document.getElementById('meuFormulario').addEventListener('submit', function(event) {
2   event.preventDefault(); // Evita o envio real do formulário
3   alert('o formulário foi enviado com sucesso!');
4 });
```

Fonte: Autoria Própria, 2024.

Na figura 6 foi adicionada uma função em Javascript com o objetivo de quando clicar no botão de “enviar”, aparecerá uma notificação de mensagem dizendo: “o formulário foi enviado com sucesso!” no topo da tela.

Em seguida, o resultado apresentado da codificação.

Figura 7 - Resultado Exemplo Básico JavaScript

A imagem mostra uma interface web com um formulário centralizado. No topo, uma caixa de mensagem amarela com uma borda verde contém o texto "Essa página diz" e "o formulário foi enviado com sucesso!", acompanhado de um botão verde "OK". Abaixo, o formulário possui campos para "Nome:" (contendo "Mariana"), "Email:" (contendo "mari@gmail.com") e "Mensagem:" (contendo "Teste JavaScript"). Um botão verde "Enviar" está na base do formulário. O fundo da interface é um gradiente de azul claro.

Fonte: Autoria Própria, 2024.

Após o salvamento do registro do formulário em Javascript aparecerá a notificação no topo da tela informando que o formulário foi enviado com sucesso.

2.4. Visual Studio Code

Para Pacheco (2022), o Visual Studio Code é um editor de texto de código aberto desenvolvido pela Microsoft que adere a muitas linguagens de programação, que foi criada em 2015 e é uma das mais utilizadas atualmente na área.

2.5. Wireframe

Segundo Teixeira (2011), os Wireframes são um meio prático para planejar um site, podendo de maneira simples exemplificar a funcionalidade e *design* geral das páginas de um projeto.

Consoante Mendonça (2017), para criar um *design* eficiente é necessário se planejar com antecedência, porque desenvolver uma aplicação sem ter um planejamento antes, é semelhante a construir uma casa sem ter uma planta de sua arquitetura.

Os wireframes podem ser de alta, média ou baixa fidelidade. Wireframes de baixa fidelidade são esboços simples e despojados que representam a estrutura básica e o *layout* de uma interface de usuário, sem se preocupar com detalhes visuais como cores, tipografia ou imagens, utilizando formas básicas,

como caixas e linhas. Esses wireframes se concentram na organização e disposição dos elementos, bem como na navegação entre as diferentes partes da interface. Eles são criados na primeira fase do desenvolvimento.

No exemplo das figuras 8 e 9, vemos os wireframes de baixa fidelidade de uma página de cadastro e *login*, representando os campos de forma simplória e sem detalhamento.

Figura 8 - Exemplo Wireframe Baixa Cadastro



Fonte: Autoria Própria, 2024.

Aqui temos a página para realização de cadastro para “criar conta”, com os seguintes campos: Nome, e-mail, senha e data de nascimento. Observa-se que são apresentados de forma simplória na tela imaginada.

Figura 9 - Exemplo Wireframe Baixa Login



Fonte: Autoria Própria, 2024.

Já nesta próxima interface, temos o *login* após a criação da conta, com os seguintes campos: Nome e senha. Observa-se que são apresentados de forma simplória na tela imaginada.

Wireframes de alta fidelidade são representações detalhadas e precisas de uma interface de usuário, quase indistinguíveis da versão final do produto. Eles incluem todos os elementos visuais, como cores, tipografia, imagens e ícones, proporcionando uma visão realista de como a interface se apresentará aos usuários. Criados com ferramentas avançadas de *design*, esses wireframes também podem incorporar interatividade, simulando a navegação e o comportamento da interface. Eles são criados na fase mais avançada do projeto, próximo a entrega final.

No exemplo das figuras 10 e 11, temos a revalidação dos wireframes de baixa fidelidade, agora sendo atribuídos cores, fontes e tamanhos semelhantes a como a interface final irá ficar, assim, tornando a prototipação de alta fidelidade.

Figura 10 - Exemplo Wireframe Alta Cadastro

O wireframe apresenta uma interface de usuário para a criação de uma conta. No topo, há uma barra de navegação com links: HOME, SOBRE NÓS, CONTATO e PARCERIAS. Abaixo, o título principal é 'Crie sua Conta', seguido pelo subtítulo 'Faça login para continuar.'. O formulário contém campos para: NOME (com o exemplo 'Mariana'), EMAIL (com o exemplo 'mar@gmail.com.br'), SENHA (com caracteres ocultos por pontos) e DATA DE NASCIMENTO (com um botão 'Selecionar'). Um botão laranja 'Sign up' está abaixo dos campos. Abaixo do botão, há um link azul: 'Já tem registro? Faça login.'. Na base da página, uma barra laranja contém o texto 'Entre em contato:', o endereço 'alo@grandesite.com.br', o telefone '(02) 3456-7890' e um ícone de QR code.

Fonte: Autoria Própria, 2024.

Aqui temos a página para realização de cadastro para “criar conta”, com os seguintes campos: Nome, e-mail, senha e data de nascimento. Observa-se que foram atribuídos cores e símbolos à tela imaginada.

Figura 11 - Exemplo Wireframe Alta Login

Fonte: Autoria Própria, 2024.

Já nesta próxima interface, temos o *login* após a criação da conta, com os seguintes campos: Nome e senha. Observa-se que foram atribuídos cores e símbolos à tela imaginada.

Assim, os wireframes servem de base para a montagem de sites e aplicativos, e serão utilizadas para esse fundamento.

2.6. Figma

Segundo Villain (2023), o Figma é uma plataforma de *design* reconhecida e usada no mundo inteiro, utilizada principalmente para a prototipação de interfaces e aplicações.

Conforme Staiano (2022), é possível a construção de *brainstorms* e wireframes, proporcionando um código fonte em CSS, facilitando o desenvolvimento, sendo assim, o figma permite que crie ilustrações em forma vetorial.

2.7. Expo

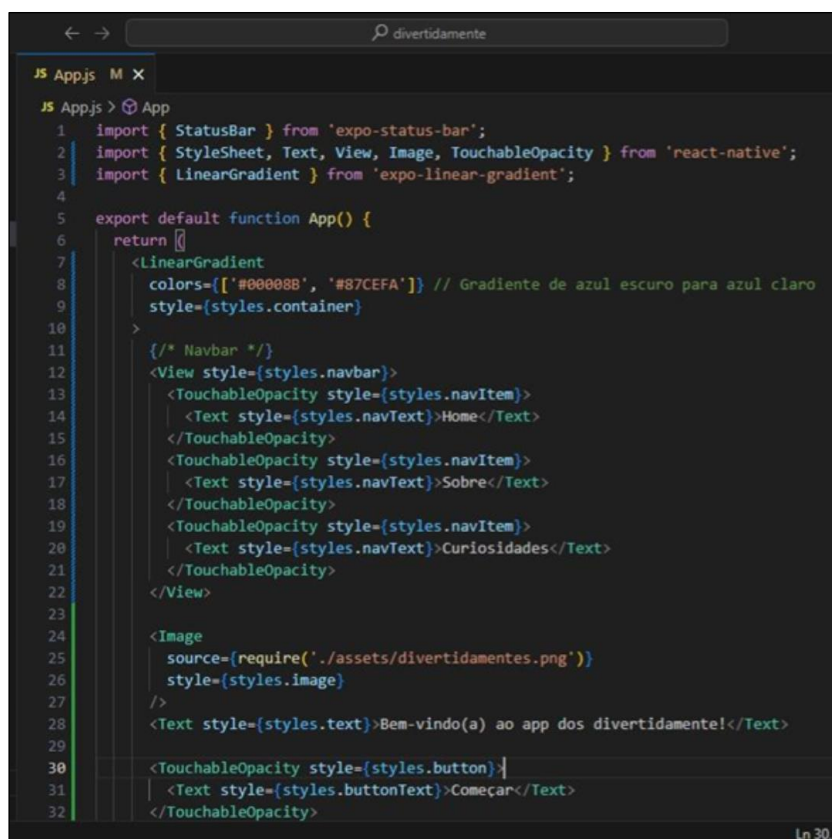
Para Fuentes (2023), o Expo é uma ferramenta utilizada no desenvolvimento *mobile* com React Native que permite o fácil acesso às APIs nativas dos dispositivos sem precisar instalar qualquer dependência ou alterar código nativo.

O Expo é uma ferramenta que ajuda no desenvolvimento de aplicativos, sendo possível criar aplicativos para IOS, Android e *web*. Baseado em JavaScript

ou TypeScript, ele promove suporte para programas no emulador Android ou smartphone.

A seguir, veremos um exemplo de codificação realizada dentro da plataforma Visual Studio Code utilizando React Native:

Figura 12 - Exemplo de codificação React Native



```
JS App.js M X
JS App.js > App
1 import { StatusBar } from 'expo-status-bar';
2 import { StyleSheet, Text, View, Image, TouchableOpacity } from 'react-native';
3 import { LinearGradient } from 'expo-linear-gradient';
4
5 export default function App() {
6   return (
7     <LinearGradient
8       colors={['#00008B', '#87CEFA']} // Gradiente de azul escuro para azul claro
9       style={styles.container}
10     >
11       /* Navbar */
12       <View style={styles.navbar}>
13         <TouchableOpacity style={styles.navItem}>
14           <Text style={styles.navText}>Home</Text>
15         </TouchableOpacity>
16         <TouchableOpacity style={styles.navItem}>
17           <Text style={styles.navText}>Sobre</Text>
18         </TouchableOpacity>
19         <TouchableOpacity style={styles.navItem}>
20           <Text style={styles.navText}>Curiosidades</Text>
21         </TouchableOpacity>
22       </View>
23
24       <Image
25         source={require('./assets/divertidamentes.png')}
26         style={styles.image}
27       />
28       <Text style={styles.text}>Bem-vindo(a) ao app dos divertidamente!</Text>
29
30       <TouchableOpacity style={styles.button}>
31         <Text style={styles.buttonText}>Começar</Text>
32       </TouchableOpacity>
33     </LinearGradient>
34   );
35 }
```

Fonte: Autoria Própria, 2024.

Na figura 12, temos um exemplo básico de codificação utilizando o framework React Native. Nesta imagem, o código está importando algumas extensões do React e a estrutura da aplicação mobile.

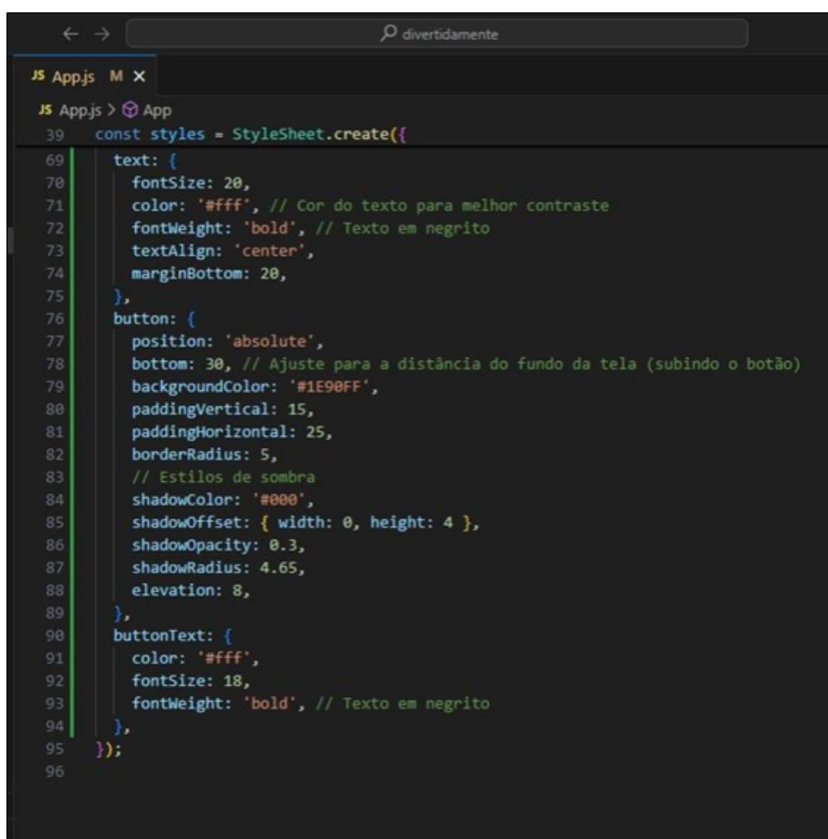
Figura 13 - Exemplo de codificação React Native Dois

```
JS Appjs M X
JS Appjs > App
5 export default function App() {
34   <StatusBar style="auto" />
35   </LinearGradient>
36   };
37 }
38
39 const styles = StyleSheet.create({
40   container: {
41     flex: 1,
42     alignItems: 'center',
43     justifyContent: 'center',
44   },
45   navbar: {
46     flexDirection: 'row',
47     justifyContent: 'center', // Centraliza todos os itens
48     width: '100%',
49     paddingVertical: 15,
50     position: 'absolute',
51     top: 30, // Desce a navbar para baixo
52     elevation: 5, // Sombra para a barra de navegação
53   },
54   navItem: {
55     padding: 10,
56     flex: 1, // Permite que os itens se espalhem uniformemente
57     alignItems: 'center', // Centraliza o conteúdo
58   },
59   navText: {
60     fontSize: 18,
61     color: '#fff', // Cor do texto da navbar
62   },
63   image: {
64     width: 210 // Aumenta a largura da imagem
```

Fonte: Autoria Própria, 2024.

Na figura 13, encontramos a continuação da estrutura da aplicação, e a adição da estilização da tela.

Figura 14 - Exemplo de codificação React Native Três



```

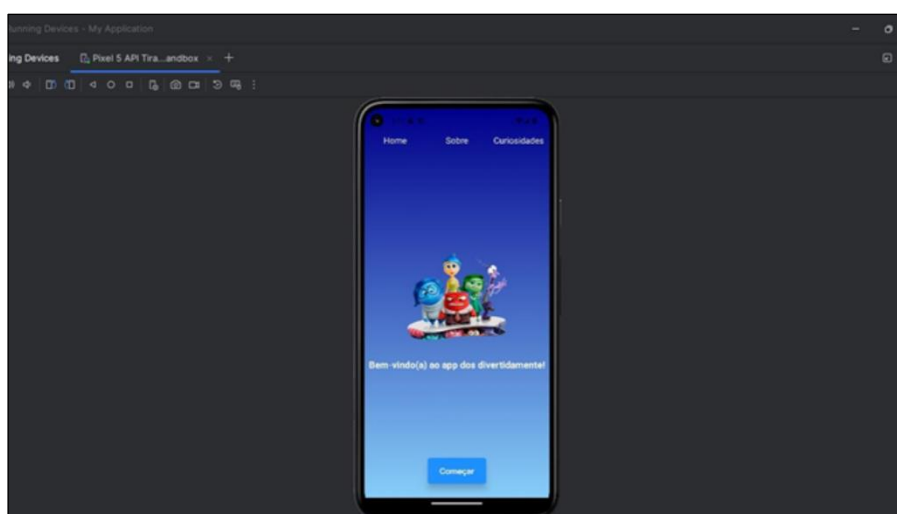
39  const styles = StyleSheet.create({
69    text: {
70      fontSize: 20,
71      color: '#fff', // Cor do texto para melhor contraste
72      fontWeight: 'bold', // Texto em negrito
73      textAlign: 'center',
74      marginBottom: 20,
75    },
76    button: {
77      position: 'absolute',
78      bottom: 30, // Ajuste para a distância do fundo da tela (subindo o botão)
79      backgroundColor: '#1E90FF',
80      paddingVertical: 15,
81      paddingHorizontal: 25,
82      borderRadius: 5,
83      // Estilos de sombra
84      shadowColor: '#000',
85      shadowOffset: { width: 0, height: 4 },
86      shadowOpacity: 0.3,
87      shadowRadius: 4.65,
88      elevation: 8,
89    },
90    buttonText: {
91      color: '#fff',
92      fontSize: 18,
93      fontWeight: 'bold', // Texto em negrito
94    },
95  });
96

```

Fonte: Autoria Própria, 2024.

Na figura 14, contém a estilização completa da aplicação mobile, onde cada parte da estrutura que vimos anteriormente está melhor apresentável.

Figura 15 - Emulação no App Expo



Fonte: Autoria Própria, 2024.

Na figura 15, vemos a emulação do aplicativo através da ferramenta expo, tendo uma visualização plena do funcionamento programado em React Native.

2.8. React

Consoante Silva (2021) React é uma biblioteca de JavaScript que simplifica e acelera o desenvolvimento de interfaces de usuário interativas e eficientes, criada em 2011 pelo engenheiro do Facebook Jordan Walke, foi usada pela primeira vez no *feed* de notícias da empresa.

Conforme Samy (2021) o React é uma biblioteca modular, o que significa que os componentes podem facilmente ser reutilizados e compartilhados entre diferentes partes da aplicação.

2.9. Framework

De acordo com Noronha (2024) framework é um conjunto de bibliotecas que tratam de funcionalidades e estruturas para o desenvolvimento de aplicações, com o objetivo de fornecer soluções para um mesmo problema, permitindo a reutilização do seu código.

Para Bittencourt (2021) diz que é uma ferramenta que lhe permitirá focar apenas no desenvolvimento do projeto, e não em detalhes de configuração.

2.10. React Native

Em concordância com Sereno (2018), é uma tecnologia que permite o desenvolvimento de aplicações híbridas somente com Javascript tendo a opção de utilizar código nativo quando necessário.

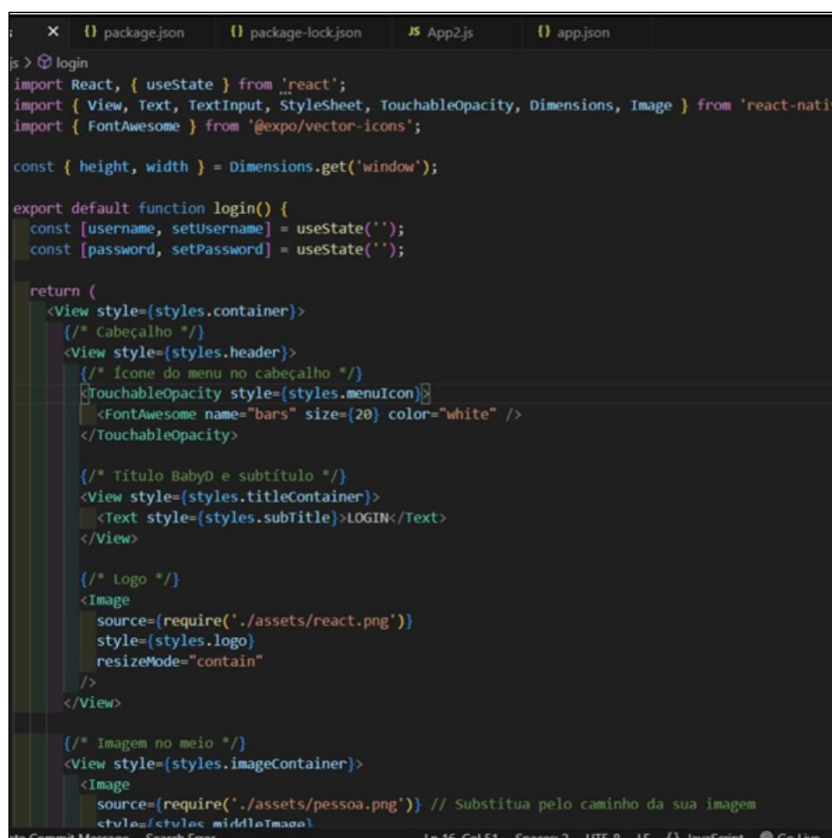
A seguir, algumas das principais funções do React:

- StyleSheet: O gerador de folha de estilos;
- Text: É o enclosure de textos;
- Image: Objeto de imagem;
- View: Uma caixa genérica, tipo *div*;
- Component: Componente do React Native;
- PropTypes: Propriedades;
- `getData(username) {}`: É uma função que utiliza do `fetch API`, onde pega a resposta e converte para JSON;

- } from 'react-native': Permite abstrair o lado Android e iOS para criar um *App* que funcione nos dois sistemas operacionais;
- Input extends Component { : É a definição das propriedades dos componentes. É importante pois serve para validação e inicialização com valor vazio.

As figuras abaixo, mostram um exemplo básico de react com react native.

Figura 16 - Exemplo Código Básico React Native



```
import React, { useState } from 'react';
import { View, Text, TextInput, StyleSheet, TouchableOpacity, Dimensions, Image } from 'react-native';
import { FontAwesomeIcon } from '@expo/vector-icons';

const { height, width } = Dimensions.get('window');

export default function login() {
  const [username, setUsername] = useState('');
  const [password, setPassword] = useState('');

  return (
    <View style={styles.container}>
      <View style={styles.header}>
        <TouchableOpacity style={styles.menuIcon}>
          <FontAwesomeIcon name="bars" size={20} color="white" />
        </TouchableOpacity>

        <View style={styles.titleContainer}>
          <Text style={styles.subTitle}>LOGIN</Text>
        </View>

        <Image
          source={require('./assets/react.png')}
          style={styles.logo}
          resizeMode="contain"
        />
      </View>

      <View style={styles.imageContainer}>
        <Image
          source={require('./assets/pessoa.png')} // Substitua pelo caminho da sua imagem
          style={styles.middleImage}
        />
      </View>
    </View>
  );
}
```

Fonte: Autoria Própria, 2024.

Na figura 16, apresenta um trecho de código React que descreve um componente funcional denominado *Login*. Este componente cria um formulário de *login* com o campo Nome do usuário e o campo senha, e um botão de Entrar. Juntamente com as importações das bibliotecas do próprio React Native.

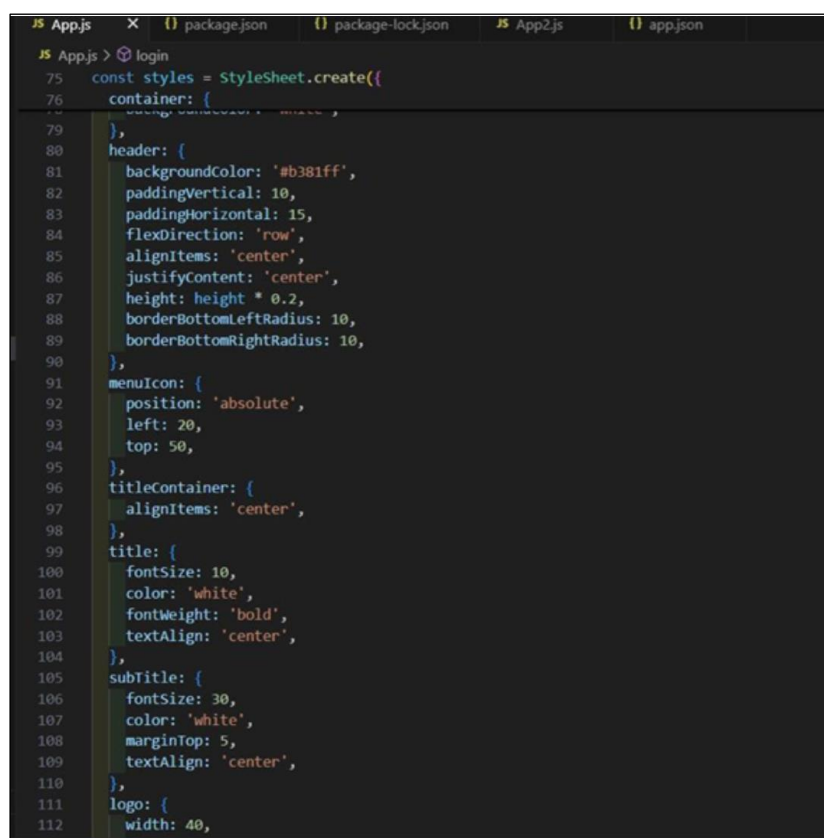
Figura 17 - Continuação Um Código Básico React Native

```
App.js x {} package.json {} package-lock.json JS App2.js {} app.json
JS App.js > login
7 export default function login() {
54   style={styles.input}
55   placeholder="Senha"
56   value={password}
57   onChangeText={setPassword}
58   secureTextEntry={true}
59 }
60
61 /* Botão de Entrar */
62 <TouchableOpacity style={styles.scanButton}>
63   <Text style={styles.buttonText}>Entrar</Text>
64 </TouchableOpacity>
65
66 /* Ícone de menu no canto inferior esquerdo */
67 <TouchableOpacity style={styles.bottomMenuIcon}>
68   <FontAwesome name="bars" size={24} color="white" />
69 </TouchableOpacity>
70 </View>
71 );
72 }
73
74 // Definição de estilos
75 const styles = StyleSheet.create({
76   container: {
77     flex: 1,
78     backgroundColor: 'white',
79   },
80   header: {
81     backgroundColor: '#b381ff',
82     paddingVertical: 10,
83     paddingHorizontal: 15,
84     flexDirection: 'row',
85     alignItems: 'center',
86     justifyContent: 'center',
87     height: height * 0.2,
88     borderBottomLeftRadius: 10,
89     borderBottomRightRadius: 10
90   }
91 });
```

Fonte: Autoria Própria, 2024.

Na figura 17, é a continuação da estrutura da codificação em React, mas a partir desta figura, iniciando a estilização da estrutura.

Figura 18 - Continuação Dois Código Básico React Native

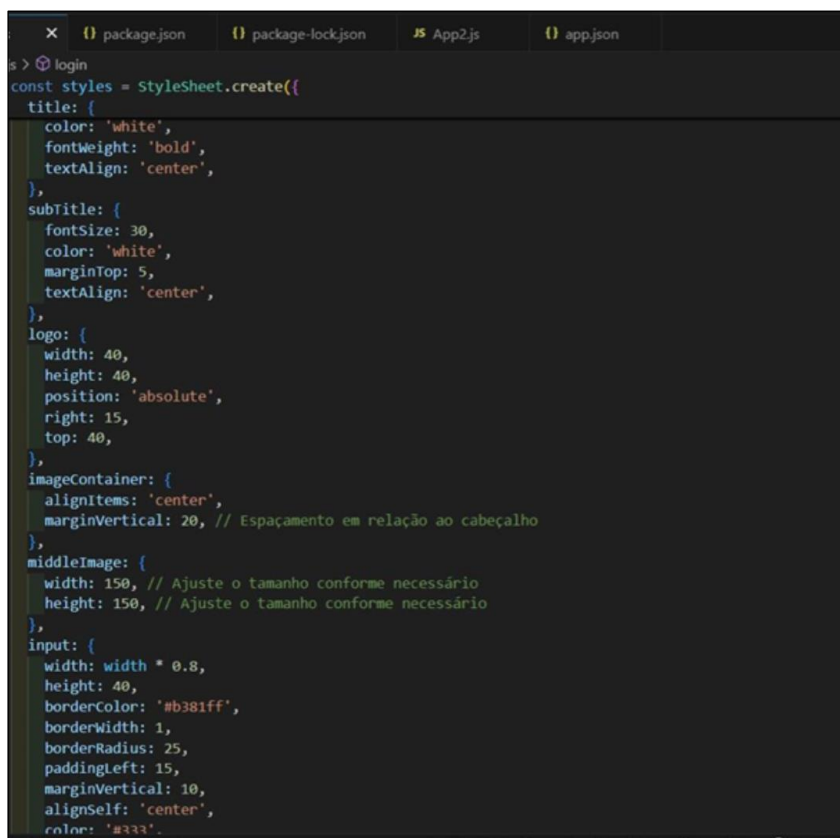


```
JS App.js > login
75 const styles = StyleSheet.create({
76   container: {
77     // ...
78   },
79   header: {
80     backgroundColor: '#b381ff',
81     paddingVertical: 10,
82     paddingHorizontal: 15,
83     flexDirection: 'row',
84     alignItems: 'center',
85     justifyContent: 'center',
86     height: height * 0.2,
87     borderBottomLeftRadius: 10,
88     borderBottomRightRadius: 10,
89   },
90   menuIcon: {
91     position: 'absolute',
92     left: 20,
93     top: 50,
94   },
95   titleContainer: {
96     alignItems: 'center',
97   },
98   title: {
99     fontSize: 10,
100    color: 'white',
101    fontWeight: 'bold',
102    textAlign: 'center',
103  },
104  subtitle: {
105    fontSize: 30,
106    color: 'white',
107    marginTop: 5,
108    textAlign: 'center',
109  },
110  logo: {
111    width: 40,
112  },
113 }
```

Fonte: Autoria Própria 2024.

Na figura 18, refere-se a estilização de toda a estrutura da aplicação, como por exemplo: Tamanho dos campos, cor da *Navbar*, estilo da fonte, Título e subtítulo, a logo dentro da *Navbar* e por fim, o *Header*.

Figura 19 - Continuação Três Código Básico React Native



```

const styles = StyleSheet.create({
  title: {
    color: 'white',
    fontWeight: 'bold',
    textAlign: 'center',
  },
  subTitle: {
    fontSize: 30,
    color: 'white',
    marginTop: 5,
    textAlign: 'center',
  },
  logo: {
    width: 40,
    height: 40,
    position: 'absolute',
    right: 15,
    top: 40,
  },
  imageContainer: {
    alignItems: 'center',
    marginVertical: 20, // Espaçamento em relação ao cabeçalho
  },
  middleImage: {
    width: 150, // Ajuste o tamanho conforme necessário
    height: 150, // Ajuste o tamanho conforme necessário
  },
  input: {
    width: width * 0.8,
    height: 40,
    borderColor: '#b381ff',
    borderWidth: 1,
    borderRadius: 25,
    paddingLeft: 15,
    marginVertical: 10,
    alignSelf: 'center',
    color: 'white',
  },
});

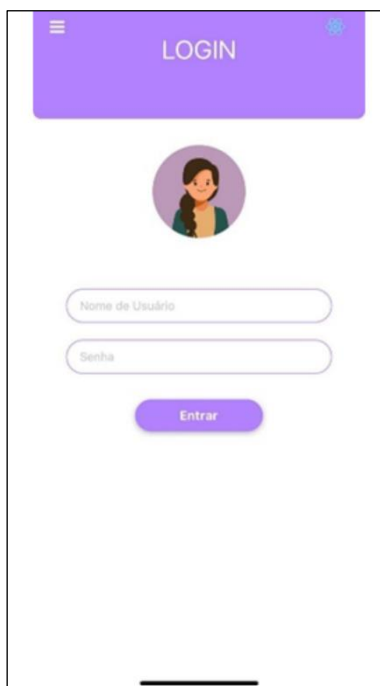
```

Fonte: Autoria Própria, 2024.

Na figura 19, ainda as estilizações da aplicação, dessa vez atribuindo ao *Container* da imagem, que se posiciona ao centro da tela, seu tamanho e os *Inputs*.

A seguir, o resultado da codificação em React com React Native.

Figura 20 - Resultado Exemplo React Native



Fonte: Autoria Própria, 2024.

Na figura 20 está o resultado da codificação, estilização e ligação do React e React Native.

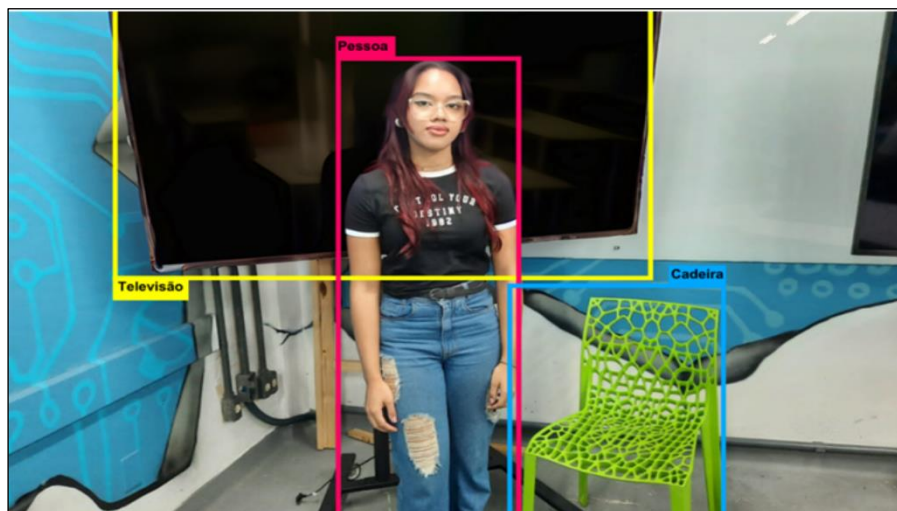
2.11. Visão Computacional

Milano e Bazorro (2010) descreve a visão computacional como uma ciência em que o computador extrai informações significativas das imagens, as quais permitem o reconhecimento e processamento de objetos e outros.

Já Piteri e Rodrigues (2011) demonstram que a visão computacional é utilizada em várias áreas e em várias situações, como por exemplo, na saúde para detectar doenças, nas eleições com o cadastro biométrico, e na segurança com reconhecimento facial e na diferenciação de rostos.

A seguir temos um exemplo de visão computacional.

Figura 21 - Exemplo Visão Computacional



Fonte: Autoria Própria, 2024.

Na figura 21, está um exemplo do que a visão computacional identificaria, como a cadeira, a televisão e a pessoa, delimitando exatamente onde cada item se localiza e o nomeando na legenda.

2.12. OpenCV

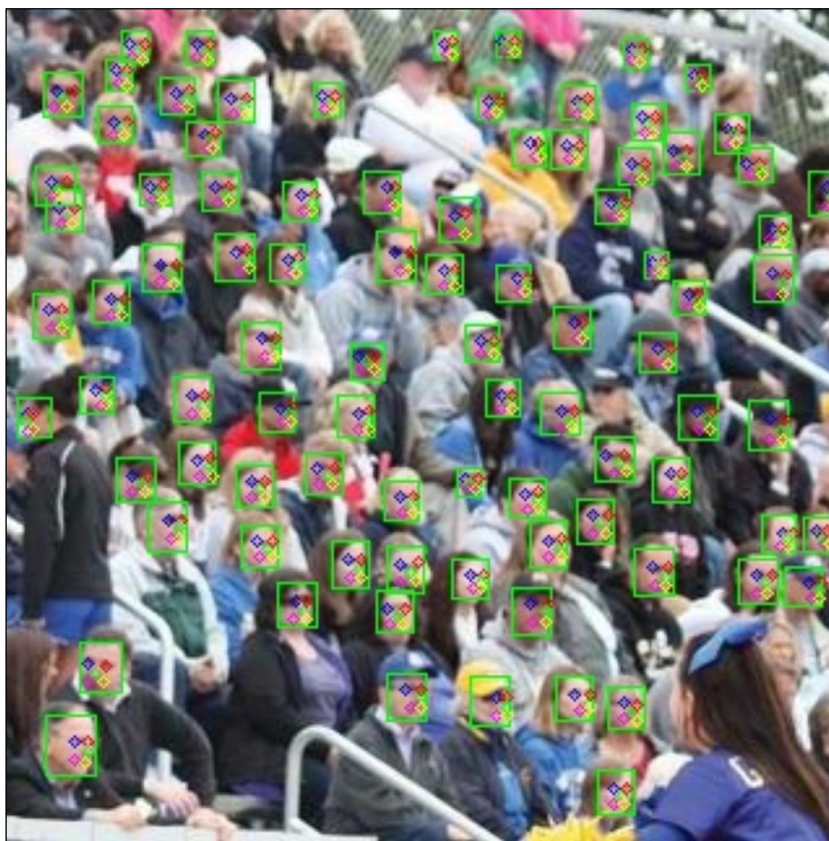
Para Barelli (2018) o OpenCV é uma biblioteca multiplataforma facilitadora para a visão computacional e o processamento de imagens. O OpenCV torna a tarefa de manipular e processar as imagens mais ágil e simples, entregando em um tempo curto todo o processamento necessário.

De acordo com Delai e Dutra (2012), o OpenCV se sobressai dentre as outras bibliotecas pois tem mais de 15 funções, é compatível com mais de 500 linguagens de programação, gratuita e é uma biblioteca *open-source*, o que permite que seja muito utilizada.

A biblioteca tem muito reconhecimento e divulgação feita por seus parceiros, entre eles estão Microsoft Azure, Intel, Roboflow e muitas outras empresas famosas e renomadas de tecnologia e segurança.

Algumas das funções do OpenCV, de acordo com a documentação oficial (2024) são redimensionamento, rotação, recorte, filtragem, equalização de histograma, entre outros.

Figura 22 - Exemplo Reconhecimento de Características



Fonte: OpenCV, 2024.

Na figura 22, a diferenciação armazenada pela biblioteca é feita a partir da detecção de 5 pontos chamados de características, que são distintos em cada pessoa. O retângulo verde define o rosto como um todo, as marcações azuis e vermelho são os olhos, o verde é o nariz, o cor-de-rosa e amarelo as extremidades da boca. Marcado esses pontos, é medido uma distância entre eles, tamanho das características e variações como pontos identificados que fogem desse padrão de características, todas essas informações são criptografadas em pares ordenados, e quando solicitado, as informações que foram cadastradas junto à face são mostradas.

2.13. Banco de Dados

Em concordância com Date (2004), um sistema de banco de dados é um sistema computacional destinado a manter registros, armazenar informações possibilitando que os usuários as acessem e atualizem conforme necessário.

Silva (2020) destacou algumas das principais diferenças entre um banco de dados relacional e não-relacional, dentre elas, a vinculação de tabelas e os

insights possibilitados pelo relacional que não são encontradas no não-relacional, uma vez que este usa vários tipos de modelos sem interconexões.

Os bancos não-relacionais são explicados por Brito (2010) com 4 modelos que os tornam mais funcionais do que o outro modelo citado, sendo os tipos:

Baseado em Chave-Valor;

Baseado em Documentos;

Baseado em Coluna;

Baseado em Grafos.

Alguns exemplos de SGBDs não-relacionais famosos e muito utilizados são Cassandra, Redis, MongoDB e o Firebase Firestore.

2.14. Firebase e Firestores

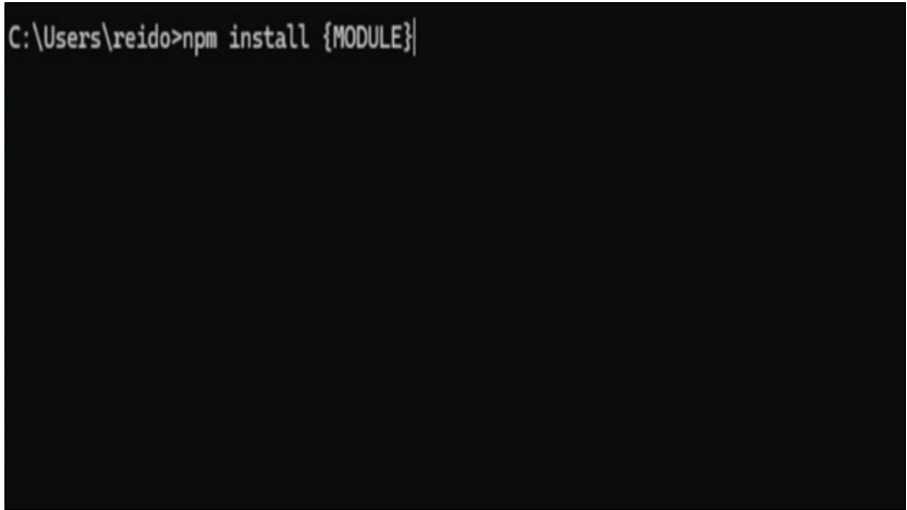
De acordo com a documentação oficial do Firebase (2023), é uma plataforma de *Backend-as-a-Service* (BaaS), que fornece infraestrutura de *back-end* pronta para quem desenvolve aplicativos, repleta de recursos, sem preocupações com hospedagem.

É uma plataforma de desenvolvimento multiplataforma criada pelo Google que fornece dezenas de utilidades, como banco de dados, autenticação, entre outros. O recurso Firestore do Firebase pode armazenar informações coletadas no sistema, e o modelo é baseado na orientação a documentos.

2.15. NPM

De acordo com a documentação oficial do NPM (2023), é o maior registro de *software* do mundo, utilizado para compartilhar e instalar pacotes, além de também ser usado por instituições para gerenciar o desenvolvimento privado.

Figura 23 - Exemplo Código NPM



```
C:\Users\reido>npm install {MODULE}|
```

Fonte: Autoria Própria, 2024.

Na figura 23, foi feito um exemplo do NPM (Node Package Manager) para a instalação de um módulo. O comando "npm install {MODULE}" é utilizado para instalar um módulo específico no contexto de um projeto Node.js. A seguir está uma breve explicação de cada parte do código:

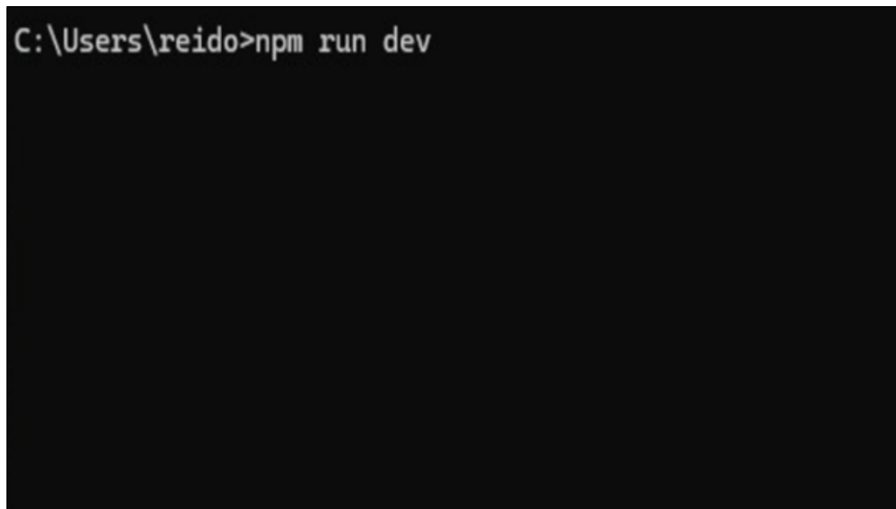
npm: É o comando Node Package Manager, uma ferramenta que facilita a instalação e o gerenciamento de pacotes de software Node.js.

install: É a instrução para npm instalar pacotes.

{MODULE}: É um espaço reservado que deve ser substituído pelo nome do módulo que você deseja instalar.

Na figura 24, foi feito um exemplo do NPM (Node Package Manager) para a iniciação de um servidor de desenvolvimento.

Figura 24 - Segundo Exemplo Código NMP



Fonte: Autoria Própria, 2024.

npm run: é usado para executar *scripts* personalizados definidos no arquivo package.json do seu projeto. O package.json é um arquivo de configuração que contém informações sobre o seu projeto, incluindo *scripts* personalizados que você pode executar.

dev: É geralmente configurado para iniciar um ambiente de desenvolvimento.

2.16. Node.js

O de acordo com Pereira (2018) o Node.js foi desenvolvido em 2009 com a intenção de desmistificar e substituir sistemas *web* bloqueantes, como Java, PHP e o .Net.

Moraes (2017) explica que o Node.js é um ambiente JavaScript orientado a eventos, utiliza a mecânica V8 do Google para utilizar em seu navegador. Em complementação, o Node.js é uma opção melhor do que as outras pois é mais leve e eficiente quando colocado em comparação, além disso, ele tem um maior desempenho em caso de tráfego intenso.

2.17. UML

Em concordância com Guedes (2018) a UML - *Unified Modeling Language* ou Linguagem de Modelagem Unificada, é uma linguagem visual utilizada para modelar *softwares* baseados no paradigma de orientação a objetos.

Segundo Booch (2006), os modelos fornecem uma cópia do projeto de um sistema. Os modelos poderão abranger planos detalhados, assim como planos mais gerais com uma visão panorâmica do sistema considerado.

De acordo com Pereira (2011), esses padrões dizem respeito aos nomes de variáveis, classes, componentes e pacotes, tratando inclusive da documentação e da forma de endentação e quebra de linhas dos programas.

2.17.1. Diagrama de Caso de Uso

Para Larman (2007), o diagrama de caso de uso é um tipo de diagrama utilizado na modelagem de sistemas. Ele é usado para capturar os requisitos funcionais de um sistema, descrevendo as interações entre os atores externos (usuários ou outros sistemas) e o próprio sistema.

- Ator: Representa um papel desempenhado por um usuário e/ou outro sistema que interage com o sistema a ser modelado.
- Caso de Uso: Descreve uma funcionalidade específica existente no sistema. Cada caso de uso é uma sequência de ações que o sistema executa para produzir um resultado observável e útil para os atores.
- Sistema: É representado por um retângulo que contém os casos de uso. Este retângulo define o escopo do sistema modelado, delimitando quais funcionalidades estão incluídas.
- Relações: Existem várias relações importantes em um diagrama de caso de uso:
 - Associação (Association): Representa a comunicação entre um ator e um caso de uso.
 - Inclusão (Include): Indica que um caso de uso incorpora a funcionalidade de outro caso de uso. É usada para modelar comportamentos comuns entre vários casos de uso.
 - Extensão (Extend): Indica que um caso de uso pode estender o comportamento de outro caso de uso em certas condições específicas.
 - Generalização (Generalization): Define uma hierarquia entre atores ou entre casos de uso, indicando uma relação de especialização.

Esses elementos ajudam a representar de forma clara e organizada como os usuários e outros sistemas interagem com o sistema representado, facilitando o entendimento e a comunicação entre desenvolvedores, analistas e outros stakeholders.

2.17.2. Diagrama de Atividade

Guedes (2011) diz que o diagrama de atividade é um tipo de diagrama usado para representar o fluxo de trabalho ou atividades dentro do sistema, descrevendo a sequência de ações e as decisões ao longo do tempo.

- **Transições:** Mostram o fluxo de controle de uma atividade para outra. São representadas por setas que conectam as atividades.
- **Decisões:** Representam pontos onde o fluxo pode divergir com base em uma condição. São mostradas como losangos com uma entrada e múltiplas saídas.
- **Ramos de Confluência:** Permitem a convergência de múltiplos fluxos em um único fluxo. Também são representados por losangos, mas com múltiplas entradas e uma saída.
- **Estados de Início e Fim:** O estado inicial é mostrado por um círculo preenchido, indicando o ponto de partida do fluxo. O estado final é representado por um círculo com um círculo preenchido interno, indicando o término do fluxo.
- **Nó de Junção:** Utilizados para modelar atividades paralelas. Um nó de junção (*fork*) divide o fluxo em múltiplas atividades que podem ser executadas simultaneamente, enquanto um nó de junção (*join*) combina fluxos paralelos em um único fluxo.
- **Swimlanes:** Utilizadas para agrupar atividades que são executadas por diferentes atores ou unidades organizacionais. As swimlanes ajudam a clarificar responsabilidades no fluxo de atividades.
- **Objetos:** Representam objetos que são produzidos ou consumidos durante as atividades. São mostrados como retângulos e podem ser associados às atividades por meio de setas.

Esses elementos combinados permitem a modelagem detalhada dos processos de negócio ou fluxos de trabalho de um sistema, facilitando a visualização e análise.

2.17.3. Diagrama de Classe

Consoante Lima (2005), o diagrama de classes é um tipo de diagrama que descreve a estrutura de um sistema, mostrando suas classes, atributos, métodos e os relacionamentos entre os objetos. Ele é fundamental na modelagem orientada a objetos, ajudando a visualizar a arquitetura do sistema e a entender como os componentes interagem entre si, ele mostra como diferentes elementos interagem entre si dentro de um sistema.

- **Classes:** Representam entidades ou conceitos do domínio do problema. Uma classe é desenhada como um retângulo e possui três compartimentos: o nome da classe, os atributos e as operações (ou métodos).

- **Atributos:** São as propriedades ou características da classe. Eles são listados no segundo compartimento da classe e definem os dados que a classe mantém.

- **Operações (Métodos):** São as funcionalidades ou comportamentos da classe. Listadas no terceiro compartimento da classe, as operações representam o que a classe pode fazer.

- **Relacionamentos:** Mostram como as classes interagem entre si. Existem vários tipos de relacionamentos:

- - **Associação:** Representa uma ligação entre duas classes que indica que os objetos de uma classe são conectados aos objetos de outra classe.

- - **Generalização (Herança):** Mostra uma relação hierárquica onde uma classe derivada (subclasse) herda características e comportamentos de uma classe base (superclasse).

- - **Dependência:** Indica que uma classe usa, mas não possui, outra classe.

- - Agregação: Representa uma relação "todo/parte" onde uma classe (o todo) é composta de outras classes (as partes), mas as partes podem existir independentemente do todo.
- - Composição: É uma forma mais forte de agregação onde as partes não podem existir independentemente do todo.
- Multiplicidade: Define quantos objetos de uma classe podem estar relacionados com um objeto de outra classe. É representada por números próximos às extremidades de uma linha de associação.
- Classes Abstratas: Indicadas por nomes em itálico ou pelo uso da palavra-chave "abstract", são classes que não podem ser instanciadas diretamente e geralmente são usadas como base para outras classes.
- Interfaces: Representam um contrato de funcionalidades que as classes que implementam essa interface devem fornecer. São representadas por um círculo ou um retângulo com a palavra-chave "interface".

Esses elementos combinam-se para formar um diagrama que oferece uma visão clara da estrutura estática de um sistema de software, facilitando a compreensão, a comunicação e o desenvolvimento de software.

2.17.4. Diagrama de Sequência

Segundo a IBM (2021), o diagrama de sequência é um tipo de diagrama usado para representar a interação entre objetos em um sistema ao longo do tempo. Ele descreve como os objetos se comunicam entre si em uma determinada sequência de eventos, mostrando a ordem em que as mensagens são trocadas entre os objetos, os elementos são descritos em detalhes, ilustrando como eles interagem para modelar o comportamento dinâmico de um sistema.

- Lifelines (Linhas de Vida): Representam os participantes na interação, como objetos ou atores. Cada linha de vida é representada como uma linha tracejada vertical.

- Activation Bars (Barras de Ativação): Mostram o período em que um objeto está executando uma ação ou está ativo. São retângulos verticais sobre uma linha de vida.

- Messages (Mensagens): Indicam a comunicação entre as linhas de vida. São tipicamente setas horizontais que podem ser síncronas (com uma ponta de seta sólida) ou assíncronas (com uma ponta de seta em forma de linha). Mensagens de retorno são linhas tracejadas.

- Gates (Portões): Representam os pontos de interação nas fronteiras do diagrama de sequência. São usados para simplificar diagramas complexos ao resumir interações.

- Combined Fragments (Fragmentos Combinados): Envolvem seções de um diagrama de sequência para especificar construtos de fluxo de controle como *loops*, condicionais (*alt*), opções (*opt*) e processamento paralelo (*par*).

- Execution Occurrences (Ocorrências de Execução): Marcam pontos no tempo quando mensagens são enviadas ou recebidas. São pequenos retângulos na linha de vida onde ocorrem as interações.

- State Invariants (Invariantes de Estado): Restrições que devem ser verdadeiras em um determinado ponto no diagrama. São condições colocadas ao longo da linha de vida para impor certos estados.

- Interaction use (Uso de Interação): Permite a inclusão de outros diagramas de sequência para modularizar e reutilizar interações, mostrado como um quadro com uma referência a outro diagrama.

Esses elementos trabalham juntos para ilustrar como os objetos interagem ao longo do tempo para alcançar funcionalidades específicas dentro de um sistema. É de suma importância a criação dos diagramas de sequência na visualização e validação da sequência temporal das operações, tornando-os cruciais tanto para o design quanto para o entendimento do comportamento do sistema.

2.18. Astah Community

Conforme Guedes (2011), o Astah community é uma ferramenta CASE, seu principal propósito é auxiliar na criação de diagramas e modelos para representar sistemas, proporcionando uma melhor comunicação e visualização dos elementos.

3. DESENVOLVIMENTO

Este capítulo descreve o processo de criação do sistema BabyD, apresentando a aplicação das ferramentas, linguagens e conceitos utilizados, conforme documentado anteriormente.

3.1. Sistema BabyD

O sistema é composto por um site e um aplicativo, com a proposta de oferecer uma solução segura e intuitiva que atenda às necessidades de identificação e monitoramento, o ambiente *web* voltado para divulgação e o aplicativo móvel focado na funcionalidade prática do projeto, oferecendo uma identidade visual coesa entre ambos. O desenvolvimento foi dividido em etapas, que incluem desde a análise de requisitos e o design da interface até a programação e os testes finais.

Nosso principal objetivo com o projeto é auxiliar a diminuição de problemas relacionados ao processo de retirada das crianças das creches, como por exemplo, sequestros, troca de crianças, atrasos ou demoras, desentendimentos entre responsáveis e funcionários, entre outros. O nosso público-alvo são as escolas de ensino infantil, especificamente da zona leste de São Paulo.

3.2. Vitrine Web do Projeto BabyD

O site do projeto BabyD foi desenvolvido como uma vitrine informativa, proporcionando uma visão geral do projeto. Ele inclui a disponibilização do apk da versão beta da aplicação, informações de contato e outros dados relevantes. Optamos por desenvolver um site de página única dividido em seções, onde cada uma contém uma informação diferente e específica.

Para o desenvolvimento do sistema, optamos pela metodologia Ágil, devido à sua flexibilidade e foco na entrega contínua de valor. Utilizamos uma combinação de ferramentas e linguagens de programação, incluindo HTML, CSS, JavaScript para o desenvolvimento do site.

Com foco em acessibilidade e atratividade para o público-alvo, o design do site foi pensado para ser visualmente agradável e funcional,

usando cores, imagens e formatos que capturam a atenção dos visitantes e incentivam a interação. Algumas diretrizes de *design* foram definidas, tendo como principais as cores claras e elementos gráficos amigáveis, alinhados com a identidade visual do projeto; textos concisos e informativos, com destaque para os benefícios do sistema e um *call to action* claro para o download do aplicativo; implementação de técnicas de SEO para melhorar o posicionamento do site nos resultados de busca e aumentar a visibilidade do projeto.

Para começar a modelagem, criamos wireframes de baixa fidelidade na plataforma Figma, que ilustram as seções iniciais a serem incluídas na página.

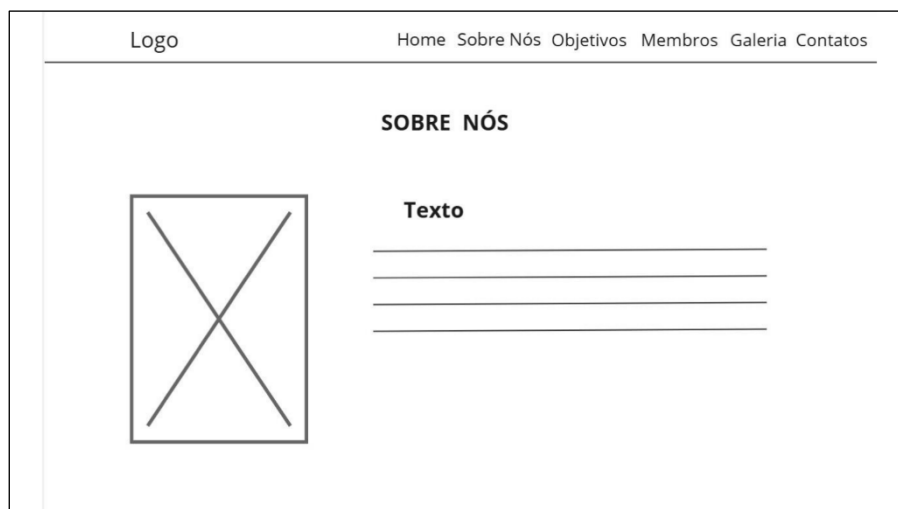
Figura 25 - Wireframe de baixa: Home



Fonte: Autoria própria, 2024.

No cabeçalho, do lado esquerdo, há o item identificado como "Logo", enquanto à direita está um menu de navegação com *links* organizados horizontalmente: "Home", "Sobre Nós", "Objetivos", "Membros", "Galeria" e "Contatos". No centro da tela, no conteúdo principal, há um título com o texto "BabyD: 'O olhar que cuida'" destacado. Abaixo do título, encontra-se um espaço para um texto adicional, representado pela palavra "Texto" em itálico.

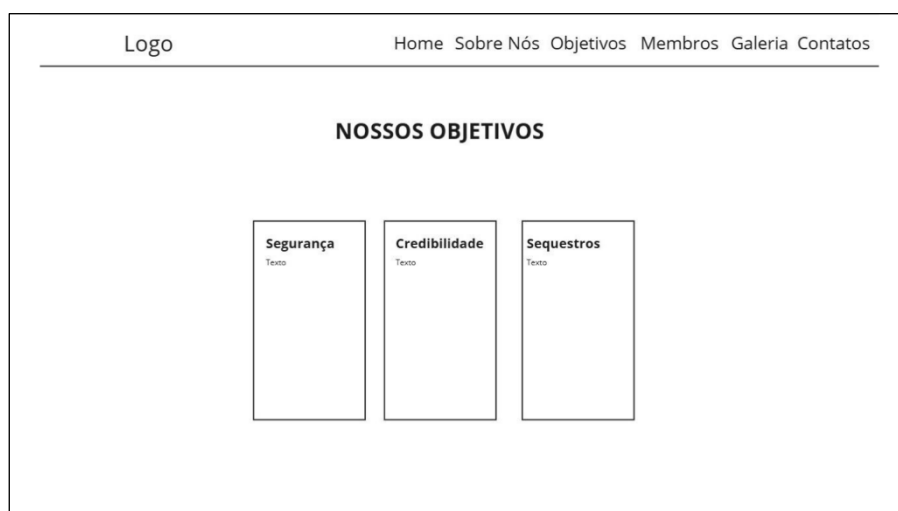
Figura 26 -Wireframe de baixa: Sobre nós



Fonte: Autoria própria, 2024.

No cabeçalho, do lado esquerdo, há o item identificado como "Logo", enquanto à direita está um menu de navegação com *links* organizados horizontalmente: "Home", "Sobre Nós", "Objetivos", "Membros", "Galeria" e "Contatos". No centro da tela, no conteúdo principal, há um título com o texto "Sobre nós" destacado. Abaixo do título, do lado direito, encontra-se um espaço para um texto adicional, representado pela palavra "Texto" seguido por linhas que indicam o tamanho, e do lado esquerdo, uma caixa que indica a presença de uma imagem.

Figura 27 - Wireframe de baixa: Objetivos



Fonte: Autoria própria, 2024.

No cabeçalho, do lado esquerdo, há o item identificado como "Logo", enquanto à direita está um menu de navegação com *links* organizados horizontalmente: "Home", "Sobre Nós", "Objetivos", "Membros", "Galeria" e "Contatos". No centro da tela, no conteúdo principal, há um título com o texto "nossos objetivos" destacado. Abaixo do título, encontram-se cards com um título seguido por um texto.

Figura 28 - Wireframe de baixa: Membros



Fonte: Autoria própria, 2024.

No cabeçalho, do lado esquerdo, há o item identificado como "Logo", enquanto à direita está um menu de navegação com *links* organizados horizontalmente: "Home", "Sobre Nós", "Objetivos", "Membros", "Galeria" e "Contatos". No centro da tela, no conteúdo principal, há um título com o texto "nossos membros" destacado. Abaixo do título, encontram-se cards com fotos de perfil.

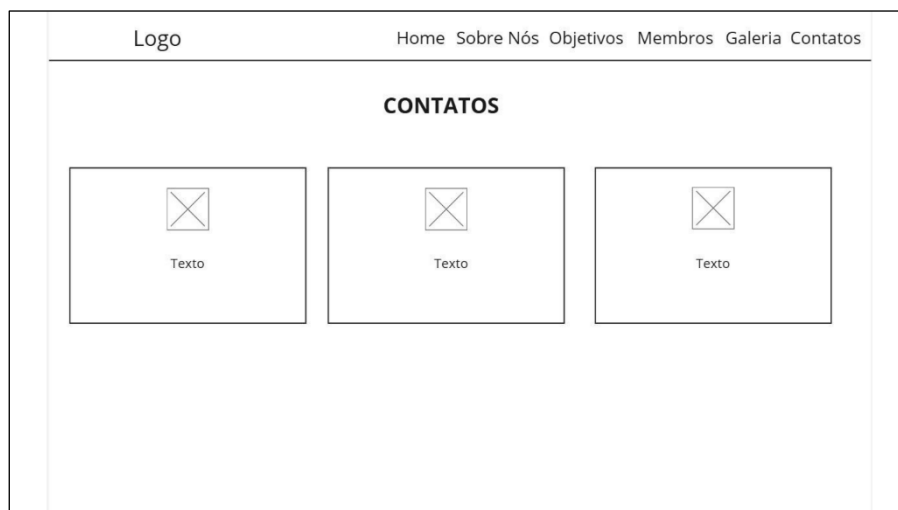
Figura 29 - Wireframe de baixa: Galeria



Fonte: Autoria própria, 2024.

No cabeçalho, do lado esquerdo, há o item identificado como "Logo", enquanto à direita está um menu de navegação com *links* organizados horizontalmente: "Home", "Sobre Nós", "Objetivos", "Membros", "Galeria" e "Contatos". No centro da tela, no conteúdo principal, há um título com o texto "nossa galeria" destacado. Abaixo do título, encontram-se uma seção com imagens.

Figura 30 - Wireframe de baixa: Contatos

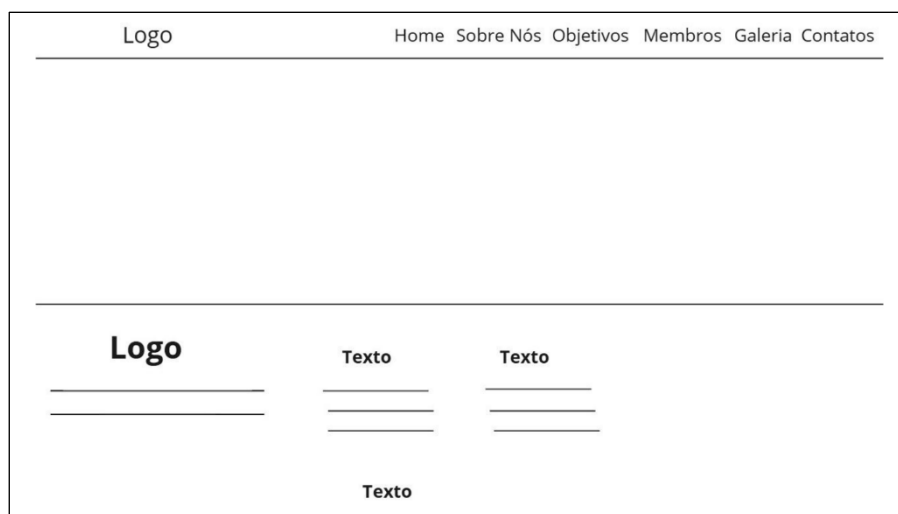


Fonte: Autoria própria, 2024.

No cabeçalho, do lado esquerdo, há o item identificado como "Logo", enquanto à direita está um menu de navegação com *links* organizados horizontalmente: "Home", "Sobre Nós", "Objetivos", "Membros", "Galeria" e "Contatos". No centro da tela, no conteúdo principal, há um título com o

texto "contatos" destacado. Abaixo do título, encontram-se cards, com uma caixa indicando a presença de imagens seguidas por um texto.

Figura 31 - Wireframe de baixa: Footer



Fonte: Autoria própria, 2024.

No cabeçalho, do lado esquerdo, há o item identificado como "Logo", enquanto à direita está um menu de navegação com *links* organizados horizontalmente: "Home", "Sobre Nós", "Objetivos", "Membros", "Galeria" e "Contatos". Na parte inferior da tela, no conteúdo principal, há uma seção destinada para a logo seguida por um texto. Na seção central e direita, um texto em cada coluna.

O *layout* é minimalista e bem-organizado, formando uma interface voltada para a apresentação clara e objetiva das informações, com foco no conteúdo textual e na navegação.

Após essa etapa, realizamos testes de usabilidade para ajustar e aprimorar a interface, permitindo uma experiência mais fluida. As seções foram então redesenhadas em wireframes de alta fidelidade, com detalhes adicionais que ilustram o *layout* final.

Figura 32 - Wireframe de alta: Home

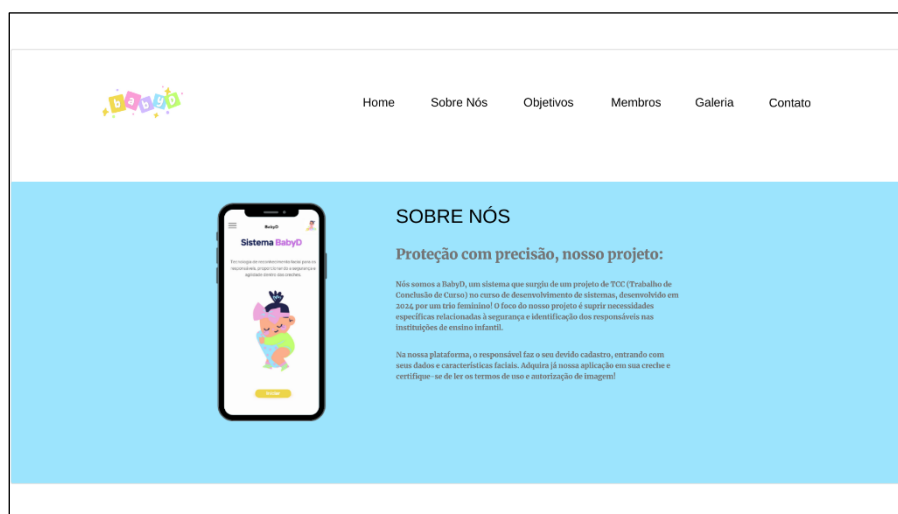


Fonte: Autoria própria, 2024.

No cabeçalho, à esquerda, encontra-se o logotipo do projeto, à direita, há um menu de navegação horizontal com as opções: "Home", "Sobre Nós", "Objetivos", "Membros", "Galeria" e "Contato".

No conteúdo principal, centralizado, está o texto "BabyD: O olhar que cuida", com o nome "BabyD" em destaque na cor rosa. Logo abaixo, há um texto descritivo simples sobre o projeto. Ao final dessa descrição, há um link em azul com o texto "Baixe Já!", que sugere ser um botão de ação para *download*. Na parte inferior do *layout*, há uma forma ondulada em azul claro, funcionando como um elemento decorativo que adiciona suavidade ao design.

Figura 33 - Wireframe de alta: Sobre nós

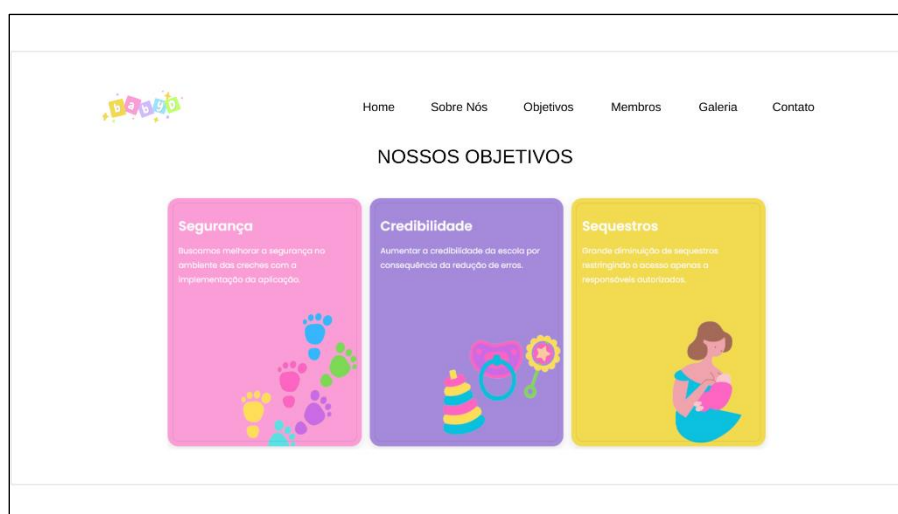


Fonte: Autoria própria, 2024.

No cabeçalho, à esquerda, encontra-se o logotipo do projeto, à direita, há um menu de navegação horizontal com as opções: "Home", "Sobre Nós", "Objetivos", "Membros", "Galeria" e "Contato".

No conteúdo principal, está dividido em duas colunas, sendo a esquerda uma imagem representativa do app, e a direita o título destacado “sobre nós” e um texto falando brevemente sobre o projeto. Essa seção tem destaque azul no fundo, indicando ser uma continuação da anterior.

Figura 34 - Wireframe de alta: Objetivos

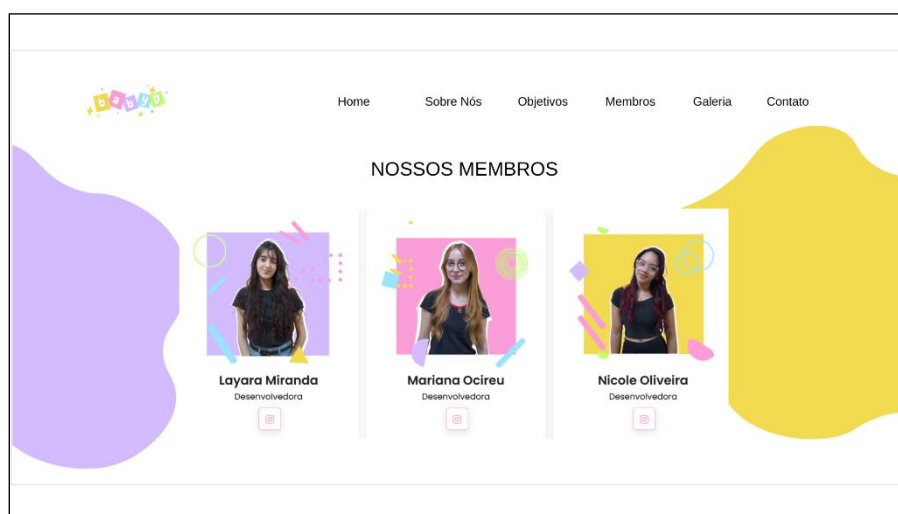


Fonte: Autoria própria, 2024.

No cabeçalho, à esquerda, encontra-se o logotipo do projeto, à direita, há um menu de navegação horizontal com as opções: "Home", "Sobre Nós", "Objetivos", "Membros", "Galeria" e "Contato".

No conteúdo principal, centralizado, está o título "Nossos objetivos", seguido por três cards, cada um deles com um título e uma explicação breve, e uma imagem ilustrativa.

Figura 35 - Wireframe de alta: Membros

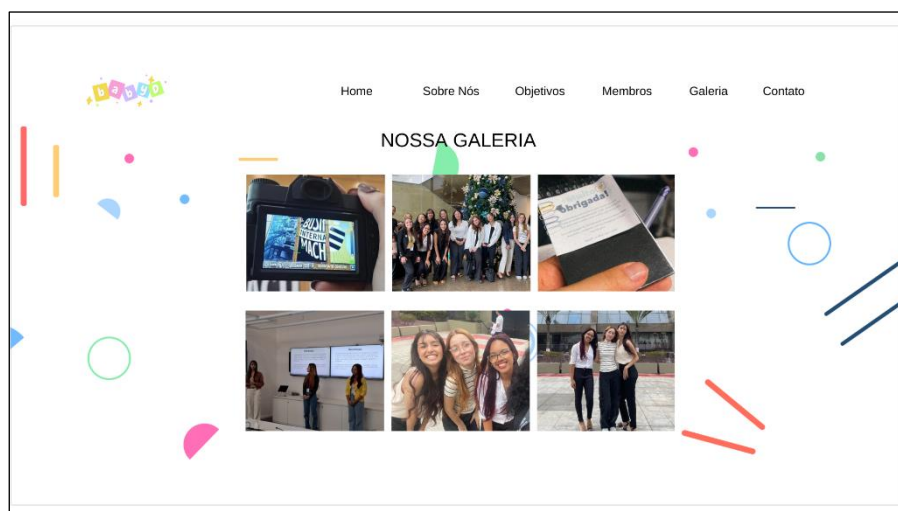


Fonte: Autoria própria, 2024.

No cabeçalho, à esquerda, encontra-se o logotipo do projeto, à direita, há um menu de navegação horizontal com as opções: "Home", "Sobre Nós", "Objetivos", "Membros", "Galeria" e "Contato".

No conteúdo principal, centralizado, está o título "Nossos membros", seguido por *cards* com imagens de perfil de cada integrante do grupo, a função no projeto e um botão com o ícone da rede social instagram.

Figura 36 - Wireframe de alta: Galeria

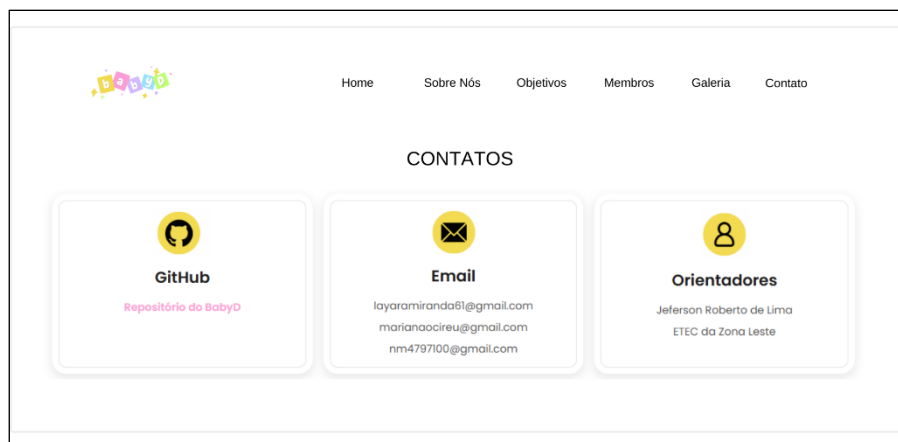


Fonte: Autoria própria, 2024.

No cabeçalho, à esquerda, encontra-se o logotipo do projeto, à direita, há um menu de navegação horizontal com as opções: "Home", "Sobre Nós", "Objetivos", "Membros", "Galeria" e "Contato".

No conteúdo principal, centralizado, está o título "nossa galeria", seguido por imagens do desenvolvimento do projeto.

Figura 37 - Wireframe de alta: Contatos

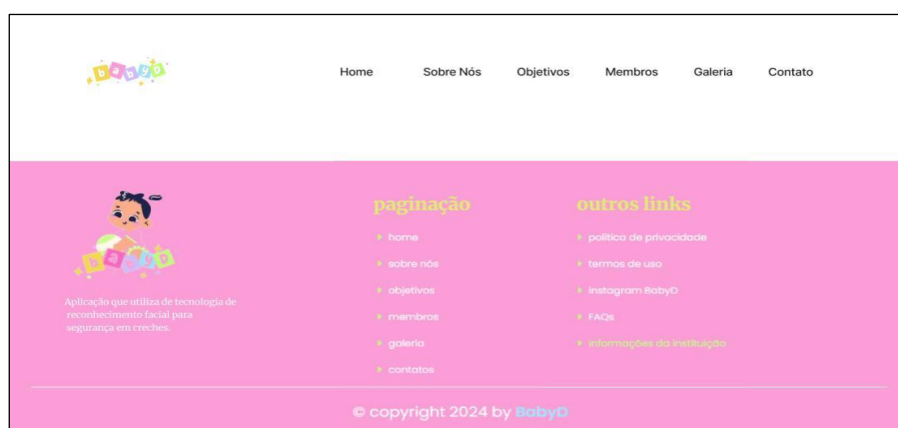


Fonte: Autoria própria, 2024.

No cabeçalho, à esquerda, encontra-se o logotipo do projeto, à direita, há um menu de navegação horizontal com as opções: "Home", "Sobre Nós", "Objetivos", "Membros", "Galeria" e "Contato".

No conteúdo principal, centralizado, está o texto "contatos" seguido por cards com informações de contato. Em cada *card* se encontra um ícone representativo e as informações referentes a cada conteúdo.

Figura 38 - Wireframe de alta: Footer



Fonte: Autoria própria, 2024.

No cabeçalho, à esquerda, encontra-se o logotipo do projeto, à direita, há um menu de navegação horizontal com as opções: "Home", "Sobre Nós", "Objetivos", "Membros", "Galeria" e "Contato".

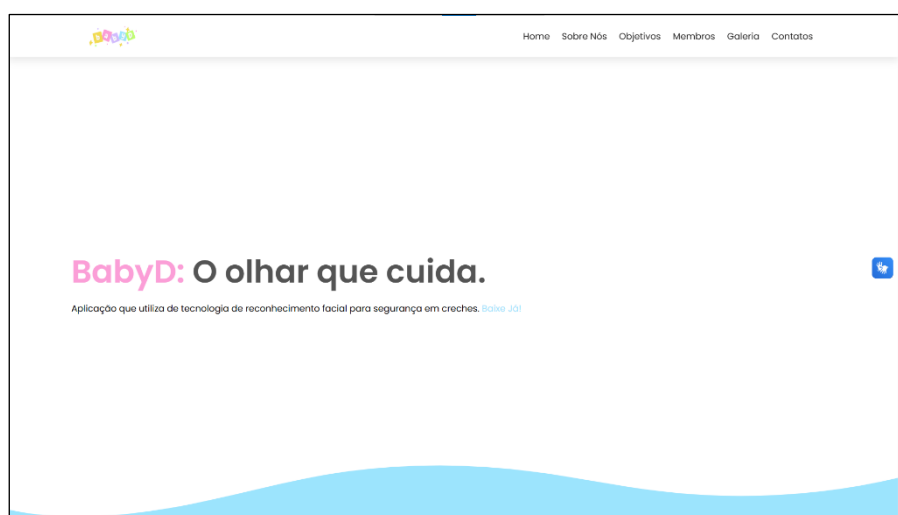
No conteúdo principal, está dividido em três colunas; a coluna esquerda é a seção onde está a logo e o *slogan* do projeto; a seção central se encontra a paginação com os mesmos títulos do cabeçario; e, a coluna direita se encontra os *links* externos, como páginas agregadas ao site.

O protótipo final mostra um *layout* limpo e organizado, adota um design minimalista e amigável, com cores suaves e uma hierarquia visual bem definida. A combinação entre texto descritivo e elementos gráficos resulta em uma interface atraente e informativa, ideal para o sistema

Com os testes e a definição de cores concluídos, programamos o site na plataforma VSCode.

Abaixo está o layout final do site, que atende aos requisitos de design e funcionalidade estabelecidos.

Figura 39 - Resultado do site



Fonte: Autoria própria, 2024.

O resultado do site foi avaliado positivamente, com uma taxa de conversão inicial de 100,8, o que demonstra a atratividade do projeto para o público-alvo. As estratégias de SEO, como otimização de palavras-chave e tags, foram aplicadas para melhorar o posicionamento nos mecanismos de busca.

3.3. Aplicação Móvel do Projeto BabyD

A aplicação móvel concentra as funcionalidades principais do projeto, sendo a mais importante o reconhecimento facial do responsável, conforme descrito anteriormente.

O aplicativo foi projetado para ser intuitivo, a interface do aplicativo foi pensada para facilitar a interação tanto para os responsáveis quanto para os funcionários, utilizando telas padronizadas e com *layout* simplificado, ainda com a identidade visual do projeto, para evitar poluição visual.

O aplicativo foi modelado de forma que suas funcionalidades ficassem concentradas em poucas páginas, aumentando a objetividade e eficiência de tempo. Optamos por desenvolver o aplicativo para a plataforma Android devido à sua maior presença entre o público-alvo. A modelagem inicial das páginas foi feita na plataforma Figma, com a criação de wireframes de baixa fidelidade.

Figura 40 - Wireframe de baixa: Tela inicial

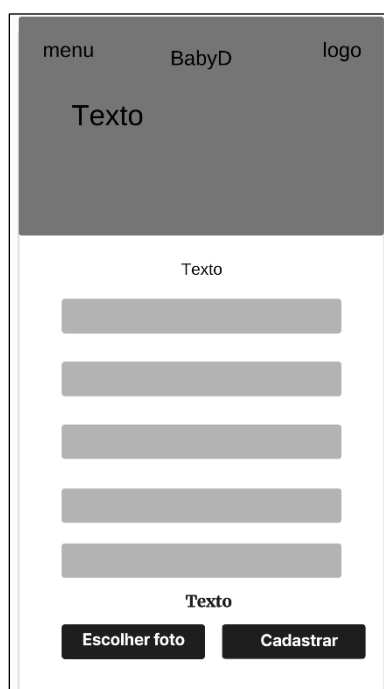


Fonte: Autoria própria, 2024.

No cabeçalho, no canto superior esquerdo, há um item identificado como "menu", enquanto no centro superior está o texto "BabyD". No canto superior direito, há um item identificado como "logo".

Logo abaixo do cabeçalho, encontra-se o título principal, "Sistema BabyD". Abaixo do título principal, há uma seção de texto genérica com a palavra "texto". No centro da interface, há um espaço reservado para uma imagem, representado pela palavra "imagem". Na parte inferior, há um botão de ação identificado como "Iniciar".

Figura 41 - Wireframe de baixa: Cadastro

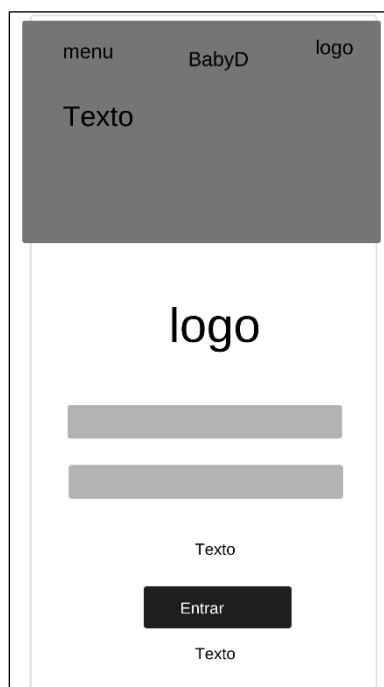


Fonte: Autoria própria, 2024.

No cabeçalho, no canto superior esquerdo, há um item identificado como "menu", enquanto no centro superior está o texto "BabyD". No canto superior direito, há um item identificado como "logo". O cabeçalho está estilizado para que tenha cor diferente do resto da página e um título genérico identificado como texto.

Logo abaixo do cabeçalho, encontra-se o título principal, "texto". Abaixo do título principal, há uma seção de campos genéricos sem identificação, finalizado por uma caixa de texto genérica com a palavra "texto". Na parte inferior, há dois botões de ação identificados como "escolher foto" e "cadastrar".

Figura 42 - Wireframe de baixa: Login



Fonte: Autoria própria, 2024.

No cabeçalho, no canto superior esquerdo, há um item identificado como "menu", enquanto no centro superior está o texto "BabyD". No canto superior direito, há um item identificado como "logo". O cabeçalho está estilizado para que tenha cor diferente do resto da página e um título genérico identificado como texto.

Logo abaixo do cabeçalho, encontra-se uma seção genérica identificada como logo. Abaixo, há uma seção de campos genéricos sem identificação, finalizado por uma caixa de texto genérica com a palavra "texto". Na parte inferior, há dois botões de ação identificados como "escolher foto" e "cadastrar", também seguido por uma caixa genérica de texto.

Figura 43 - Wireframe de baixa: Histórico de Responsáveis



Fonte: Autoria própria, 2024.

No cabeçalho, no canto superior esquerdo, há um item identificado como "menu", enquanto no centro superior está o texto "BabyD". No canto superior direito, há um item identificado como "logo". O cabeçalho está estilizado para que tenha cor diferente do resto da página e um título genérico identificado como texto.

Logo abaixo do cabeçalho, encontra-se o conteúdo principal, uma seção de imagens. Na parte inferior, há um botão de ação identificados como "escanear rosto".

Figura 44 - Wireframe de baixa: Registro Realizado



Fonte: Autoria própria, 2024.

No cabeçalho, no canto superior esquerdo, há um item identificado como "menu", enquanto no centro superior está o texto "BabyD". No canto superior direito, há um item identificado como "logo".

Logo abaixo do cabeçalho, encontra-se o título principal, "Texto". Abaixo do título principal, há uma seção de imagem genérica com a palavra "imagem". No centro da interface, há uma caixa de texto identificada como "texto", seguido por uma seção de campos genéricos sem identificação. Na parte inferior, há um botão de ação identificado como "fechar".

Figura 45 - Wireframe de baixa: Editar Registro



Fonte: Autoria própria, 2024.

No cabeçalho, no canto superior esquerdo, há um item identificado como "menu", enquanto no centro superior está o texto "BabyD". No canto superior direito, há um item identificado como "logo".

Logo abaixo do cabeçalho, encontra-se o título principal, "Texto". Abaixo do título principal, há uma seção de imagem genérica com a palavra "imagem". No centro da interface, há uma caixa de texto identificada como "texto", seguido por uma seção de campos genéricos sem identificação. Na parte inferior, há um botão de ação identificado como "salvar".

Figura 46 - Wireframe de baixa: Termos de Uso



Fonte: Autoria própria, 2024.

No cabeçalho, no canto superior esquerdo, há um item identificado como "menu", enquanto no centro superior está o texto "BabyD". No canto superior direito, há um item identificado como "logo".

Logo abaixo do cabeçalho, encontra-se o título principal, "Texto". Abaixo do título principal, há uma seção de texto genérica com a palavra "texto".

Após essa etapa, realizamos testes de usabilidade para ajustar e aprimorar a interface, permitindo uma experiência mais fluida. As seções foram então redesenhadas em wireframes de alta fidelidade, com detalhes adicionais que ilustram o *layout* final.

Figura 47 - Wireframe de alta: Tela inicial



Fonte: Autoria própria, 2024.

No topo da tela, há um ícone de menu no canto esquerdo, o nome "BabyD" centralizado e o logotipo do app. No corpo da página, o título "Sistema BabyD" destaca o nome estilizado em roxo, enfatizando a identidade do aplicativo. Abaixo, uma descrição resume o propósito do sistema, priorizando segurança e eficiência. Centralizado na tela, uma ilustração colorida de um bebê reforça a temática infantil. Na parte inferior, um botão amarelo com o texto "Iniciar" convida à interação.

Figura 48 - Wireframe de alta: Cadastro

O wireframe apresenta a interface de cadastro do aplicativo BabyD. No topo, há uma barra de cabeçalho com um ícone de menu no canto esquerdo, o nome "BabyD" centralizado e um logotipo no canto direito. Abaixo do cabeçalho, o título "CADASTRO" é exibido em uma caixa rosa. O corpo da página contém o título "Faça seu cadastro:" em roxo, seguido por uma série de campos de entrada para informações pessoais: Nome, Endereço, Email, Senha, RG, CPF, Parentesco e Nome do Aluno. Na base da tela, dois botões amarelos, "Escolher foto" e "Cadastrar", convidam o usuário a interagir e completar o cadastro.

Fonte: Autoria própria, 2024.

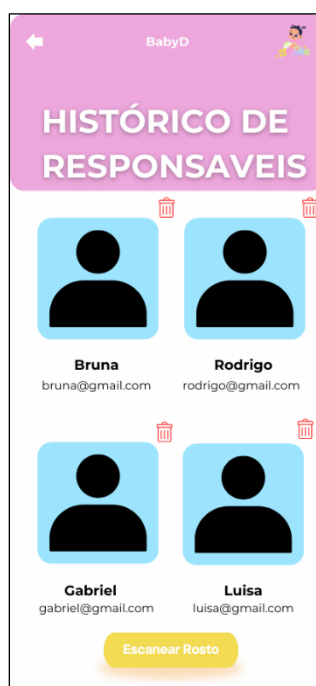
No topo da tela, estilizada com uma caixa rosa, estão um ícone de menu no canto esquerdo, o nome "BabyD" centralizado, o logotipo em destaque e o título "CADASTRO". No corpo da página, o título "Faça seu cadastro" aparece em roxo, destacando a proposta da página. Abaixo, há campos para informações pessoais, como nome e endereço. Na parte inferior, dois botões amarelos com os textos "Escolher foto" e "Cadastrar" convidam os usuários a interagir e completar o cadastro.

Figura 49 - Wireframe de alta: Login

Fonte: Autoria própria, 2024.

No topo da tela, estilizada com uma caixa rosa, estão ícone de menu no canto esquerdo, o nome "BabyD" centralizado e o título "LOGIN". No corpo da página, uma ilustração do logotipo reforça o nome do app. Abaixo, o título "Faça seu login" aparece em roxo, destacando a proposta da página, seguido por dois campos para informações pessoais. Mais abaixo, o texto "Não possui cadastro? Cadastre-se aqui!" destaca a opção de redirecionamento. Na parte inferior, um botão amarelo com o texto "Entrar" convida à interação, acompanhado do *slogan* estilizado em roxo.

Figura 36 – Wireframe de alta: Histórico de Responsáveis



Fonte: Autoria própria, 2024.

No topo, a *navbar* com uma caixa rosa inclui um ícone de menu, o nome "BabyD" centralizado, o logotipo em destaque e o título "HISTÓRICO DE RESPONSÁVEIS". No corpo, há *cards* contendo foto de perfil, nome, e-mail e um ícone de lixeira no canto superior direito. Na parte inferior, um botão amarelo com o texto "Escanear rosto" convida à interação.

Figura 37 – Wireframe de alta: Registro Realizado



Fonte: Autoria própria, 2024.

No topo, a *navbar* inclui um ícone de menu, o nome "BabyD" centralizado e o logotipo do *app*. No corpo, o título "Registro realizado" aparece em roxo, destacando a proposta da página. Abaixo, há uma foto de perfil e o título "Informações do responsável", seguido por campos com dados pessoais como nome e endereço. Na parte inferior, um botão amarelo com o texto "Fechar" encerra a interação.

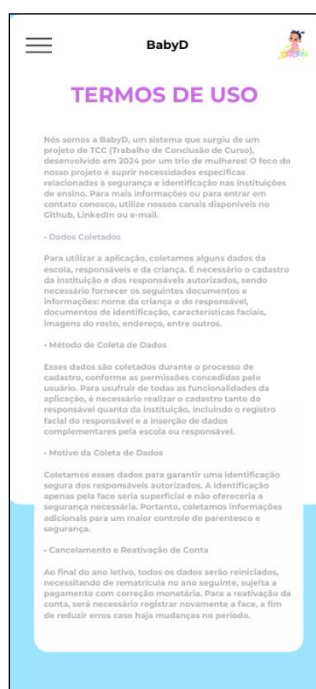
Figura 38 – Wireframe de alta: Editar Registro

O wireframe apresenta a interface de edição de registro. No topo, há uma barra de navegação com um ícone de seta para trás à esquerda, o nome 'BabyD' no centro e um ícone de perfil à direita. Abaixo, o título 'Editar Registro:' é exibido em roxo. Segue-se uma imagem de perfil placeholder (círculo preto sobre fundo azul). O formulário 'Informações do Responsável:' contém sete campos de texto cinza, cada um com um ícone de lápis à direita para edição: Nome, Endereço, Email, RG, CPF, Parentesco e Nome do Aluno. Na base do formulário, um botão amarelo com o texto 'Salvar' finaliza a interação.

Fonte: Autoria própria, 2024.

No topo, a *navbar* inclui um ícone de menu, o nome "BabyD" centralizado e o logotipo do *app*. No corpo, o título "Editar registro" aparece em roxo, destacando a proposta da página. Abaixo, há uma foto de perfil e o título "Informações do responsável", seguido por campos de dados pessoais, como nome e endereço, com ícones de edição ao lado. Na parte inferior, um botão amarelo com o texto "Salvar" finaliza a interação.

Figura 50 - Wireframe de alta: Termos de Uso



Fonte: Autoria própria, 2024.

No topo, a *navbar* inclui um ícone de menu, o nome "BabyD" centralizado e o logotipo do app. No corpo, o título "TERMOS DE USO" aparece em roxo, destacando a proposta da página. Abaixo, é apresentado um texto em tópicos com a cor cinza.

O design apresenta uma interface simples e amigável, utiliza cores suaves e elementos simples, refletindo segurança e conforto para os responsáveis pelas crianças.

Após a prototipação, a programação do aplicativo foi realizada utilizando React, React Native e CSS, com o auxílio das ferramentas Expo e NPM. O *front-end* do aplicativo está finalizado, conforme a prototipação planejada, e as telas finais foram validadas com o público teste.

Com o *front-end* concluído, foi necessário adicionar funcionalidades para tornar as páginas dinâmicas. Utilizamos diagramas UML para identificar e ilustrar as necessidades do ambiente móvel, facilitando o levantamento de requisitos e a integração de funcionalidades.

A seguir, veremos uma listagem dos requisitos funcionais, não-funcionais e regras de negócio do projeto.

Requisitos Funcionais (RF):

- RF01: Fazer Cadastro;
- RF02: Fazer Login;
- RF03: Atualizar Responsáveis;
- RF04: Escanear Rosto do Responsável;
- RF05: Solicitar Saída do Aluno;
- RF06: Autenticar Saída.

Requisitos Não Funcionais (RNF):

- RNF01: Ter uma interface intuitiva;
- RNF02: A aplicação deve ser compatível com a linguagem React Native;
- RNF03: A aplicação deve ser compatível com a linguagem Node.js;
- RNF04: O website deve ser compatível com diversas telas;
- RNF05: O sistema deve fazer a integração entre a biblioteca OpenCV e o banco de dados;
- RNF06: Aplicação não deve falhar em caso de fluxo alto;
- RNF07: O sistema deve ser responsivo;
- RNF08: O sistema deve ser compatível com o Firestore.
- RNF09: Uso de URL amigável.

Regras de Negócios:

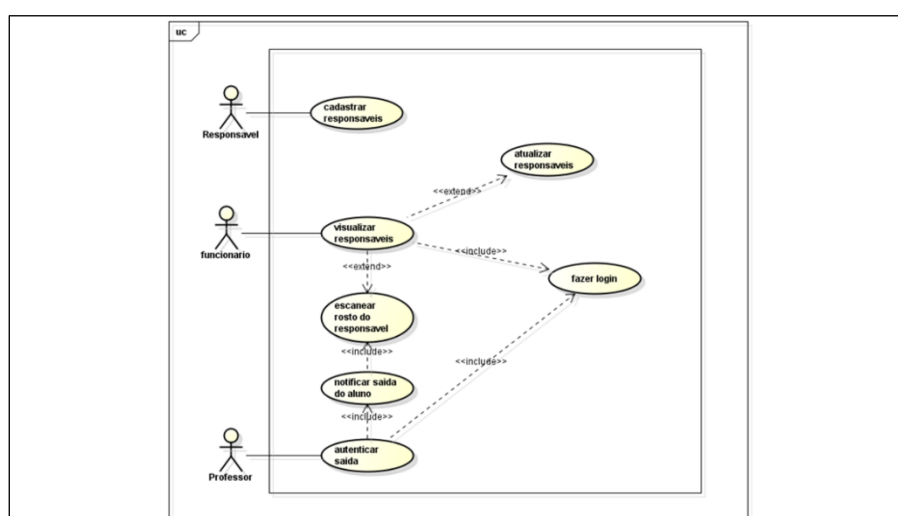
- Todos os usuários devem estar cadastrados para poderem utilizar a aplicação;
- Para liberar a retirada do aluno, o responsável precisa passar pelo o escaneamento do rosto;
- Para que o aluno seja liberado, o professor autenticar a saída;
- Os dados de cadastro do responsável precisam ser checados pelo funcionário;
- O login deve ser autenticado por um sistema de dupla verificação;
- O sistema deve escanear o rosto do responsável durante o cadastro inicial;
- O acesso aos dados dos responsáveis deve ser monitorado somente pelo funcionário;

- O professor tem permissão para autenticar saídas, desde que o responsável solicite a saída.

Os itens acima fazem parte do levantamento de requisitos, parte fundamental da documentação alinhada a UML, e auxiliaram na construção dos diagramas documentados a seguir.

A modelagem se inicia com a criação do diagrama de caso de uso, que reúne os requisitos funcionais em uma exemplificação desenhada do sistema.

Figura 51 - Diagrama de Caso de Uso



Fonte: Autoria própria, 2024.

No sistema representado pelo diagrama, cada ator desempenha funções específicas. O Responsável é responsável por cadastrar novos responsáveis no sistema. O Funcionário tem um papel central, podendo visualizar responsáveis, escanear rostos para verificar identidades e notificar a saída de alunos, sendo necessário realizar *login* para autenticação. Já o Professor é responsável por autenticar a saída dos alunos, garantindo que o processo seja realizado corretamente, também sendo necessário realizar *login* para autenticação.

Com o diagrama montado, vê-se necessário a descrição de cada caso de uso, dando origem aos quadros a seguir.

Quadro 1 - Descrição do caso de uso: Fazer cadastro

| | |
|---------------------|----------------|
| Nome do Caso de Uso | Fazer Cadastro |
| Caso de Uso geral | |

| | |
|--|--|
| Ator Principal | Responsável, Funcionário, Professor |
| Atores Secundários | |
| Resumo | Este caso de uso permite que o usuário insira e guarde seus dados para utilizar o sistema. |
| Pré-Condições | O usuário não deve estar cadastrado no sistema. |
| Pós Condições | O usuário está cadastrado no sistema e pode realizar login. |
| Fluxo Principal | |
| Ações do Ator | Ações do Sistema |
| | 1.O usuário insere os dados na tela de cadastro. |
| 2.O sistema registra os dados. | |
| 3. O sistema valida as informações e cria o cadastro do usuário. | 4.O sistema confirma o cadastro realizado com sucesso |
| Fluxo Alternativo | Se as informações fornecidas forem inválidas, o sistema solicita correções. |

Fonte: Autoria própria, 2024.

O quadro acima descreve o caso de uso “fazer cadastro” e seu passo a passo de forma objetiva.

Quadro 2 - Descrição do caso de uso: Fazer login

| | |
|--|---|
| Nome do Caso de Uso | Fazer Login |
| Caso de Uso Geral | |
| Ator Principal | Responsável, Funcionário, Professor |
| Atores Secundários | |
| Resumo | Autenticar o usuário para permitir acesso ao sistema. |
| Pré-Condições | O usuário deve estar cadastrado no sistema. |
| Pós Condições | O usuário está autenticado no sistema. |
| Fluxo Principal | |
| Ações do Ator | Ações do Sistema |
| | 1. O usuário insere as credenciais. |
| 2. O sistema valida as credenciais e autentica o usuário | |

| | |
|-------------------|--|
| Fluxo Alternativo | Se as informações fornecidas forem inválidas, o sistema informa o erro e solicita novas credenciais. |
| | Se o usuário esquecer a senha, pode solicitar a recuperação da senha. |

Fonte: Autoria própria, 2024.

O quadro acima descreve o caso de uso “fazer login” e seu passo a passo de forma objetiva.

Quadro 3 - Descrição do caso de uso: Atualizar Responsáveis

| | |
|---|---|
| Nome do Caso de Uso | Atualizar responsáveis |
| Caso de Uso Geral | |
| Ator Principal | Funcionário |
| Atores Secundários | |
| Resumo | Este caso de uso permite que o funcionário altere, salve e delete dados de cadastro dos responsáveis. |
| Pré-Condições | O funcionário deve estar logado no sistema. |
| Pós Condições | Salva as informações alteradas. |
| Fluxo Principal | |
| Ações do Ator | Ações do Sistema |
| | 1.O funcionário acessa a tela e confere os responsáveis cadastrados. |
| 3. O sistema valida as informações e as salva no banco. | 2.O ator altera ou exclui dados. |
| 4.O sistema confirma o salvamento realizado com sucesso | |
| Fluxo Alternativo | Se as informações fornecidas tiverem valores inválidos, o sistema solicita correções |

Fonte: Autoria própria, 2024.

O quadro acima descreve o caso de uso “atualizar responsáveis” e seu passo a passo de forma objetiva.

Quadro 4 - Descrição do caso de uso: Escanear Rosto do Responsável

| | |
|---------------------|-------------------------------|
| Nome do Caso de Uso | Escanear rosto do responsável |
|---------------------|-------------------------------|

| | |
|--|---|
| Caso de Uso Geral | |
| Ator Principal | Funcionário |
| Atores Secundários | |
| Resumo | O funcionário faz o escaneamento do rosto do responsável para autenticação e verificação. |
| Pré-Condições | O responsável deve estar cadastrado no sistema. |
| Pós Condições | Habilita-se a geração de mensagem para o professor |
| Fluxo Principal | |
| Ações do Ator | Ações do Sistema |
| | 1. O funcionamento abre a funcionalidade de câmera na aplicação. |
| 3. O sistema valida as características de acordo com o banco | 2. O funcionário faz o scanner do responsável. |
| Fluxo Alternativo | Se as informações fornecidas tiverem valores inválidos, o sistema solicita correções |
| Fluxo Alternativo | Se as características ou face forem inválidas, o sistema informa o erro e solicita repetição do escaneamento. |

Fonte: Autoria própria, 2024.

O quadro acima descreve o caso de uso “escanear rosto do responsável” e seu passo a passo de forma objetiva.

Quadro 5 - Descrição do caso de uso: Solicitar Saída do Aluno

| | |
|----------------------|--|
| Nome do Caso de Uso | Solicitar Saída do Aluno |
| Caso de Uso de Geral | |
| Ator Principal | Funcionário |
| Atores Secundários | |
| Resumo | Gera mensagem de saída para autenticação do professor. |
| Pré-Condições | O funcionário deve escanear o rosto do responsável. |
| Pós Condições | O professor deve autenticar a mensagem para liberar o aluno. |
| Fluxo Principal | |
| Ações do Ator | Ações do Sistema |

| | |
|--|---|
| 1. Após o escaneamento do rosto do responsável, gera a mensagem de saída autorizada. | |
| Fluxo Alternativo | Se a captura falhar, o sistema solicita uma nova tentativa e não gera mensagem. |

Fonte: Autoria própria, 2024.

O quadro acima descreve o caso de uso “solicitar saída do aluno” e seu passo a passo de forma objetiva.

Quadro 6 - Descrição do caso de uso: Autenticar Saída

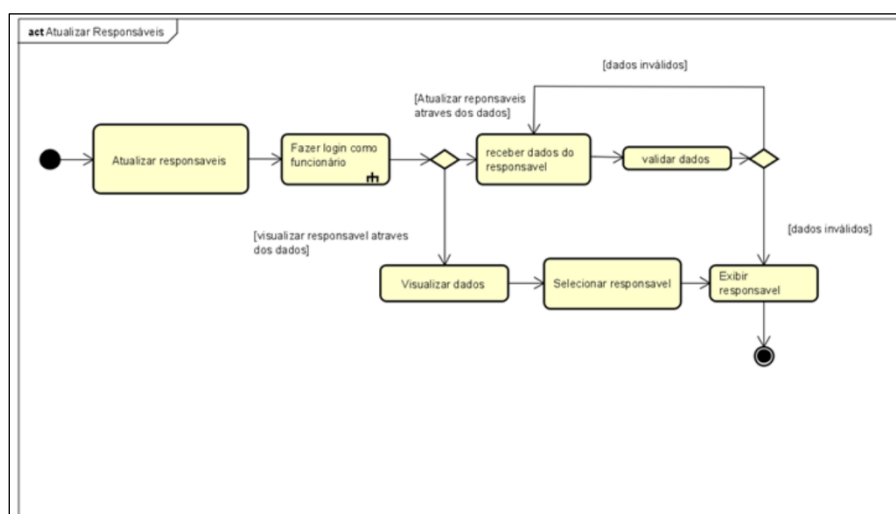
| | |
|---|---|
| Nome do Caso de Uso | Autenticar Saída |
| Caso de Uso Geral | |
| Ator Principal | Professor |
| Atores Secundários | |
| Resumo | Verifica e gerencia notificações enviadas após o caso de uso “escanear rosto do responsável”. |
| Pré-Condições | O professor deve estar autenticado no sistema. |
| Pós Condições | As mensagens são salvas no sistema. |
| Fluxo Principal | |
| Ações do Ator | Ações do Sistema |
| 1.O sistema exibe a lista de mensagens. | |
| | 2.O professor pode verificar ou remover mensagens. |
| 3.O sistema salvo as alterações realizadas. | |
| Fluxo Alternativo | Se houver algum problema na alteração das mensagens, o sistema informa o erro. |
| | Se o sistema não conseguir salvar as alterações, informa o erro e solicita nova tentativa. |

Fonte: Autoria própria, 2024.

O quadro acima descreve o caso de uso “autenticar saída” e seu passo a passo de forma objetiva.

A seguir veremos as descrições mais profundas dos processos em diagramas de atividade.

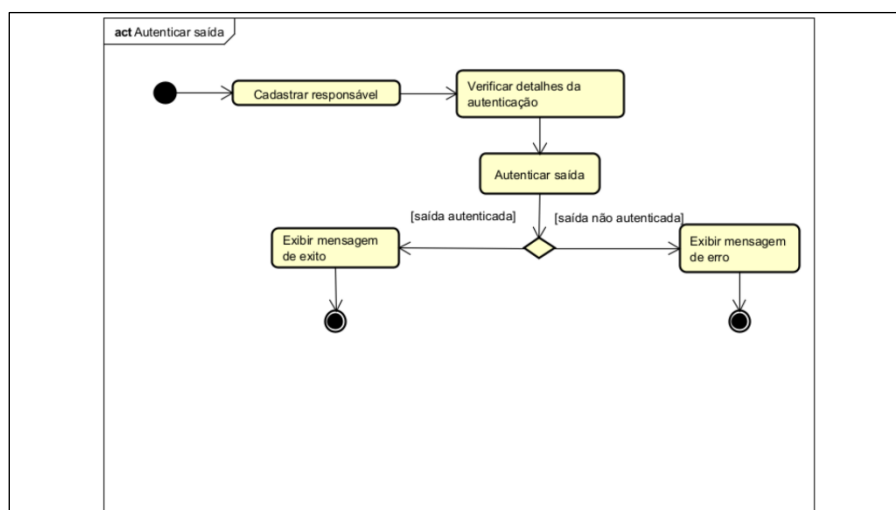
Figura 52 - Diagrama de Atividade: Atualizar Responsáveis



Fonte: Autoria própria, 2024.

O diagrama “atualizar responsáveis” descreve o processo para a atualização dos responsáveis no sistema, passando por cada passo. O usuário, ao entrar no *app* e fazer *login* como funcionário, recebe os dados e os valida, selecionando o responsável.

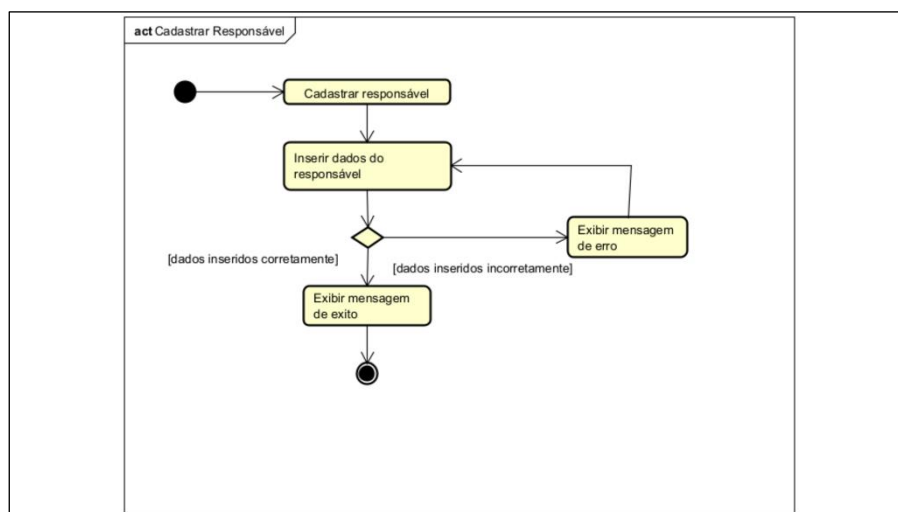
Figura 53 - Diagrama de Atividade: Autenticar Saída



Fonte: Autoria própria, 2024.

O diagrama “autenticar saída” demonstra o processo da autenticação, na qual o usuário, logado como professor, verifica os detalhes da mensagem de notificação, marcando como verdadeiro ou falso a liberação do aluno.

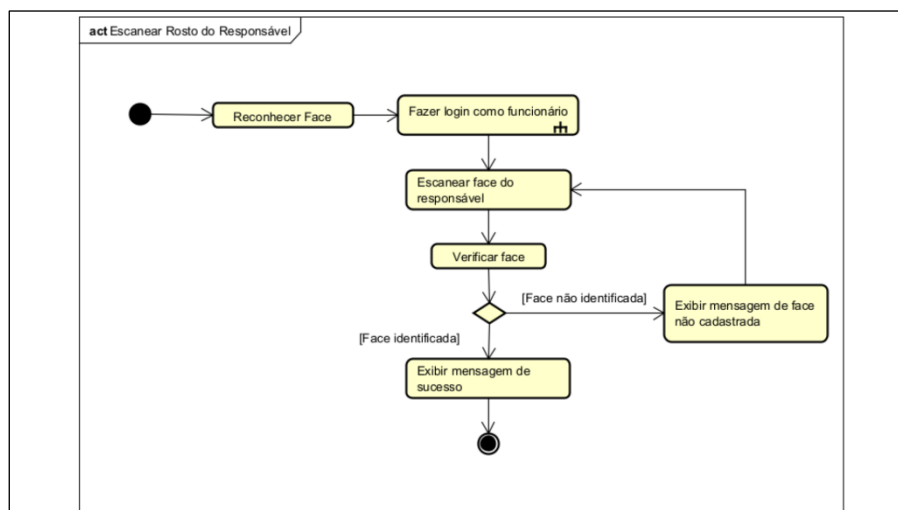
Figura 54 - Diagrama de Atividade: Cadastrar Responsáveis



Fonte: Autoria própria, 2024.

O diagrama “cadastrar responsáveis” representa a realização do cadastro, onde o responsável insere seus dados. Com os dados inseridos, o sistema faz a verificação, se estiver correto, o responsável é cadastrado com sucesso, se estiver incorreto, mostra uma mensagem de erro e solicita correção.

Figura 55 - Diagrama de Atividade: Escanear Rosto do Responsável

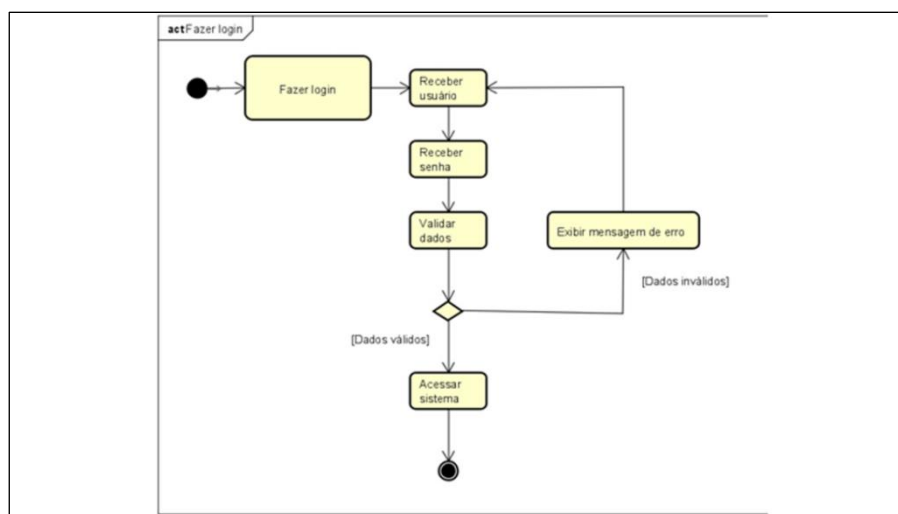


Fonte: Autoria própria, 2024.

O diagrama “escanear rosto do responsável” detalha como é feito o processo de escaneamento. Para realizar o escaneamento, é preciso fazer o login como funcionário, e então, entrar no espaço para escanear o rosto. O sistema valida as características e retorna a tela. Se as características

forem válidas, é retornado os dados de registro, caso contrário, exibe uma mensagem de erro.

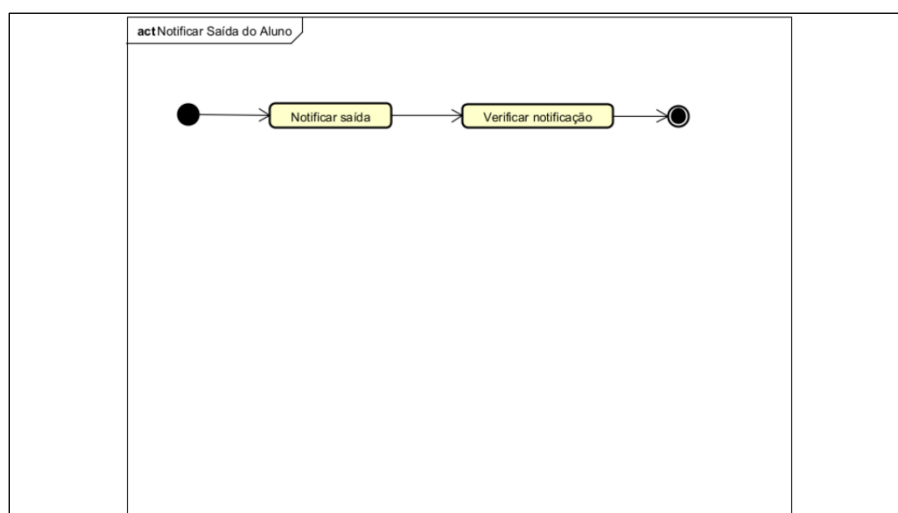
Figura 56 - Diagrama de Atividade: Fazer Login



Fonte: Autoria própria, 2024.

O diagrama explica como é feito o processo de *login* no sistema. Depois de receber o usuário, que se refere a suas credenciais de identificação, é solicitado uma senha, que é conferida pelo sistema. Se os dados forem válidos, o usuário tem acesso ao sistema, mas se os dados forem inválidos, ele não poderá acessar o sistema.

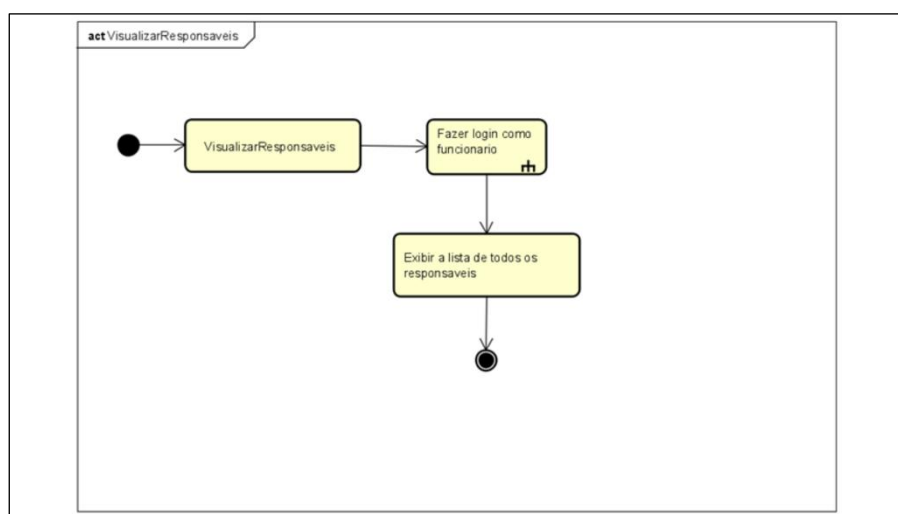
Figura 57 - Diagrama de Atividade: Notificar Saída do Aluno



Fonte: Autoria própria, 2024.

O diagrama “notificar saída do aluno” descreve o processo e gerar a notificação. Após a notificação enviada, é necessária sua verificação para ser aprovada e enviada ao professor.

Figura 58 - Diagrama de Atividade: Visualizar Responsáveis

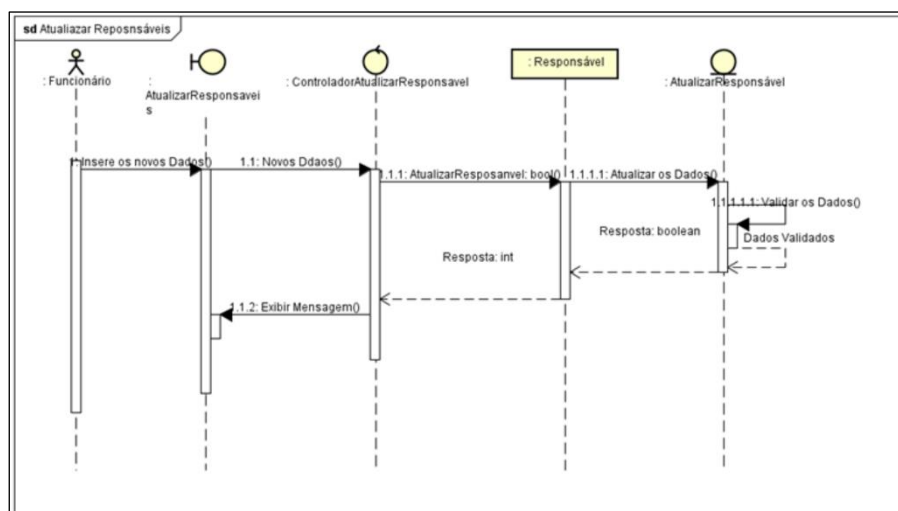


Fonte: Autoria própria, 2024.

O diagrama “visualizar responsáveis” demonstra o que acontece nesse processo. Para a visualização é necessário fazer *login* como funcionário, durante a visualização é exibido todas as informações dos responsáveis.

Em seguida, veremos a descrição dessas atividades representadas em interfaces nos diagramas de sequência.

Figura 59 - Diagrama de Sequência: Atualizar Responsáveis

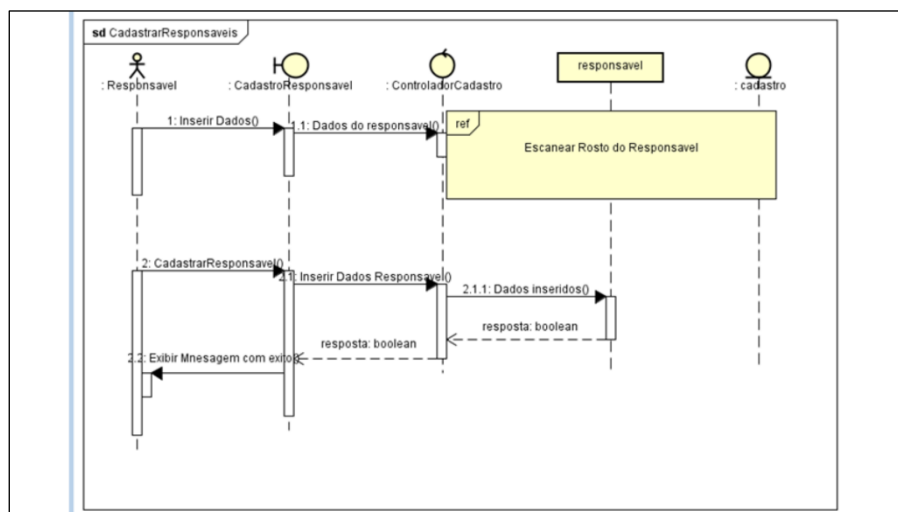


Fonte: Autoria própria, 2024.

O diagrama de sequência “atualizar responsáveis” demonstra o passo a passo desse processo de forma mais detalhada, primeiro é preciso inserir os novos dados na interface chamada "Atualizarresponsável", logo após o *controller* chamado "controladorAtualizarResponsavel" atualiza o

responsável através da classe "responsavel " automaticamente é feito a atualização dos novos dados no banco de dados "Atualizarresponsável". Após a atualização, os dados são validados, retornando para o sistema a resposta como uma "boolean", transformando para "int", e exibindo na interface uma mensagem de sucesso.

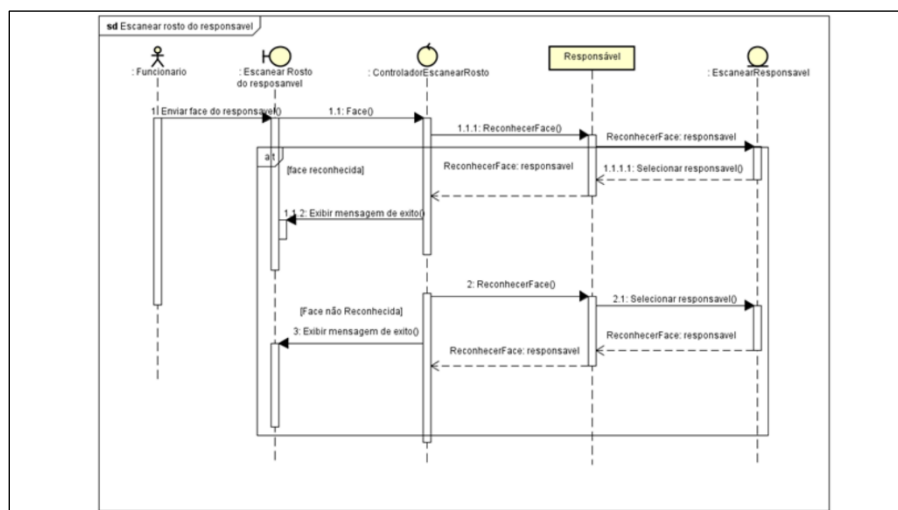
Figura 60 - Diagrama de Sequência: Cadastrar Responsáveis



Fonte: Autoria própria, 2024.

O diagrama “cadastrar responsáveis” demonstra o processo de inserção de dados no sistema. Primeiro é preciso inserir os novos dados na interface chamada "Atualizarresponsável", logo após o *controller* chamado "controladorAtualizarResponsavel" atualiza o responsável através da classe "responsavel " automaticamente é feito a atualização dos novos dados no banco de dados "Atualizarresponsável". Após a atualização, os dados são validados, retornando para o sistema a resposta como uma "boolean", transformando para "int", e exibindo na interface uma mensagem de sucesso.

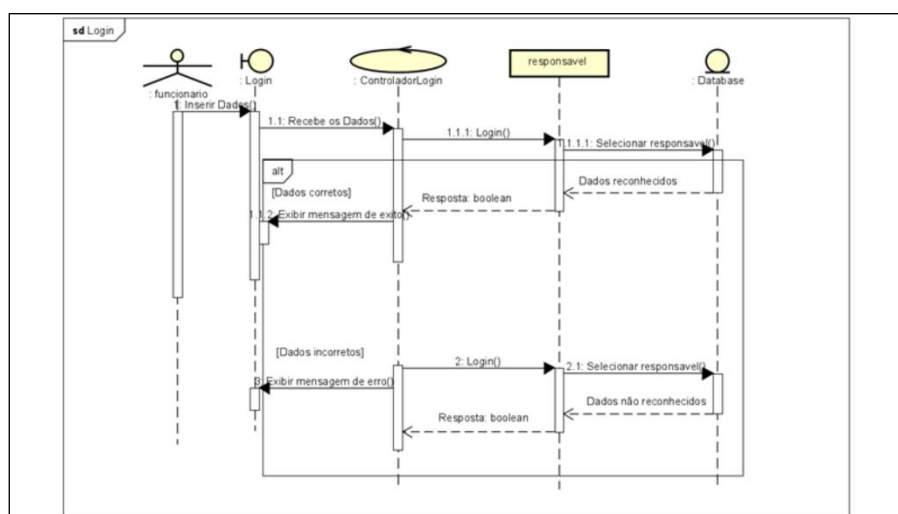
Figura 61 - Diagrama de Sequência: Escanear Rosto do Responsável



Fonte: Autoria própria, 2024.

O diagrama descreve que o funcionário após realizar o escaneamento, é enviado a "face" para o sistema através da interface "Escanear rosto do responsável". No *controller* "ControladorEscanearRosto" faz o reconhecimento dessa face, passando pela classe e enviando para o banco de dados. Depois, é selecionado o responsável para reconhecer a face, se for reconhecida, na interface é exibido uma mensagem de sucesso. Caso contrário é exibido uma mensagem de erro.

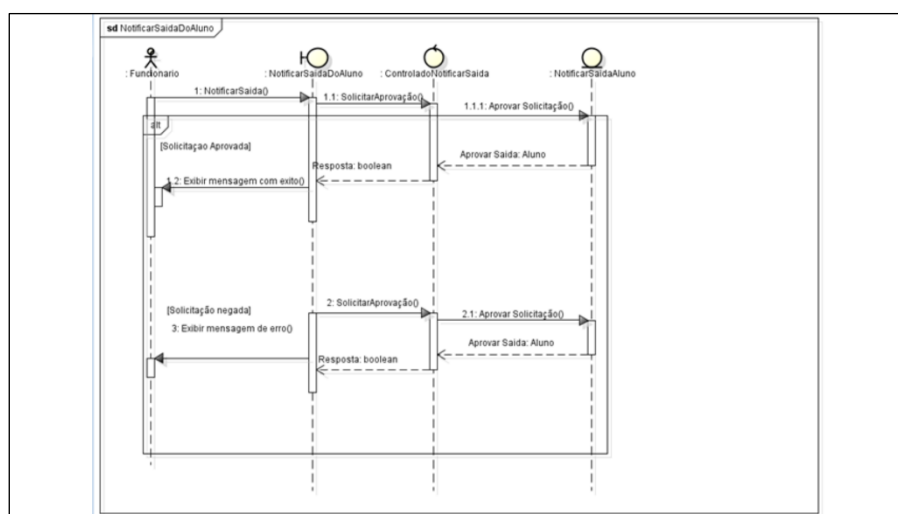
Figura 62 - Diagrama de Sequência: Fazer Login



Fonte: Autoria própria, 2024.

O diagrama demonstra o detalhamento do processo passo a passo. O funcionário insere os dados na interface "Login" e o *controller* "ControladorLogin" recebe os dados inseridos. Através da classe "responsável" é selecionado o responsável no banco de dados "database" que retorna para o sistema como dados reconhecidos, transformando em "boolean" e exibindo uma mensagem de sucesso na interface. Caso os dados não sejam reconhecidos é exibido uma mensagem de erro na interface.

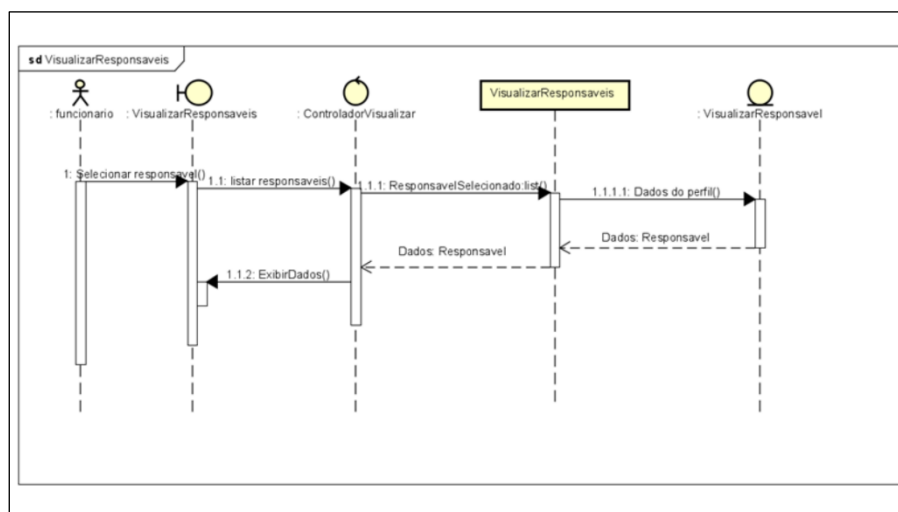
Figura 63 - Diagrama de Sequência: Notificar Saída do Aluno



Fonte: Autoria própria, 2024.

O diagrama acima representa que o funcionário notifica a saída para a interface "NotificarSaidaDoAluno", logo após, é enviado a solitação de aprovação para o *controller* "ControladorNotificarSaida". O banco de dados "NotificarSaidaAluno" recebe a aprovação da solicitação, e retorna a resposta para o sistema como uma "boolean" e exibi na interface uma mensagem de sucesso. Caso contrário, solicitação negada, é exibido uma mensagem de erro.

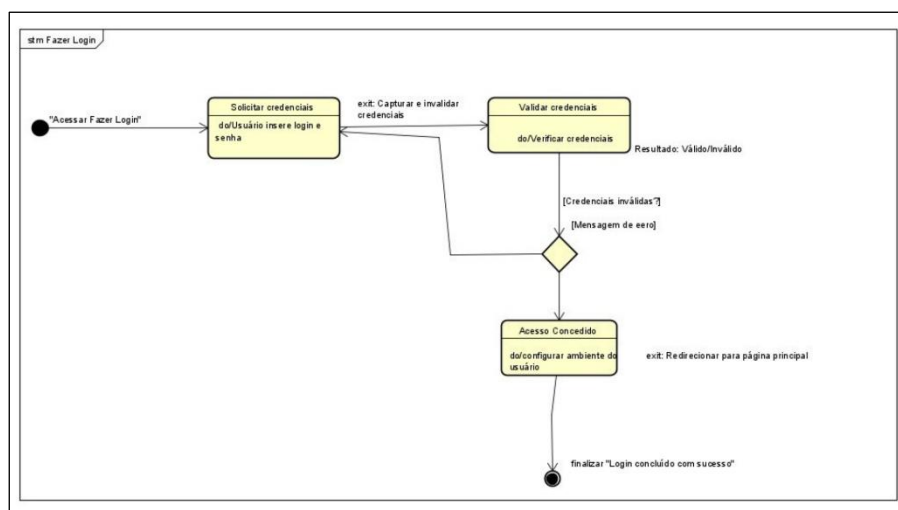
Figura 64 - Diagrama de Sequência: Visualizar Responsáveis



Fonte: Autoria própria, 2024.

O diagrama representa o processo em que o funcionário notifica a saída para a interface "NotificarSaidaDoAluno", logo após, é enviado a solicitação de aprovação para o controller "ControladorNotificarSaida". O banco de dados "NotificarSaidaAluno" recebe a aprovação da solicitação, e retorna a resposta para o sistema como uma "boolean" e exibi na interface uma mensagem de sucesso. Caso contrário, solicitação negada, é exibido uma mensagem de erro.

Figura 65 - Diagrama de Máquina de Estado: Fazer login

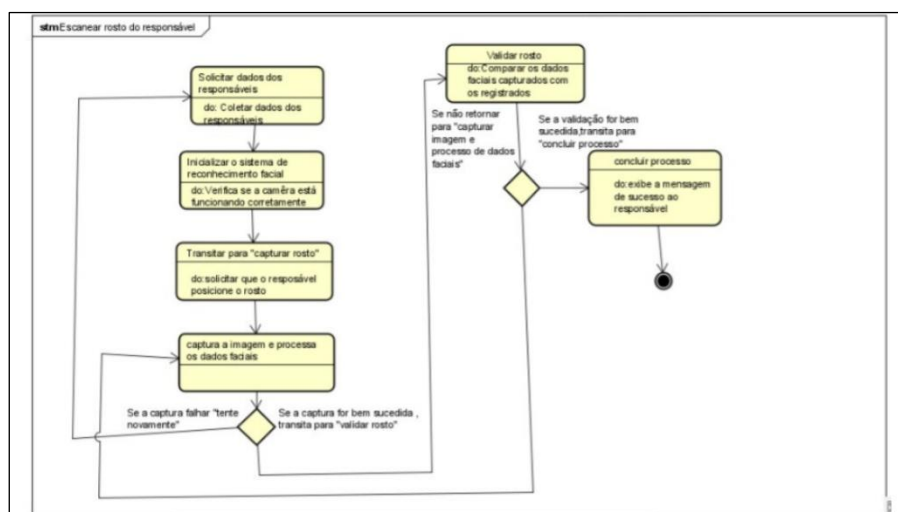


Fonte: Autoria própria, 2024.

O diagrama "fazer login" representa o processo de login no sistema, incluindo a captura e validação de credenciais. O fluxo define etapas para

solicitar credenciais, validá-las, configurar o ambiente do usuário, e redirecionar para a página principal em caso de sucesso.

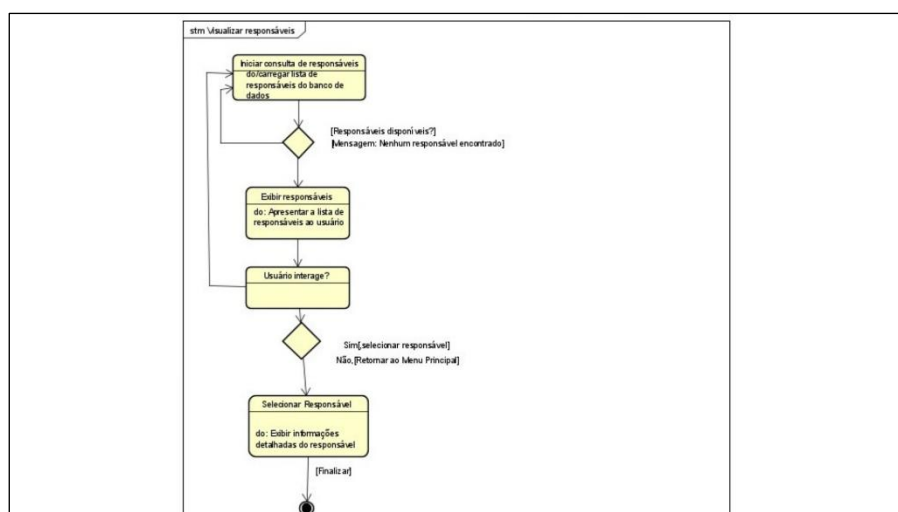
Figura 66 - Diagrama de Máquina de Estado: Escanear rosto do responsável



Fonte: Autoria própria, 2024.

O diagrama detalha o processo de reconhecimento facial. Inclui a inicialização do sistema de câmera, captura de imagem facial, validação com dados registrados e mensagem de sucesso ou erro dependendo do resultado.

Figura 67 - Diagrama de Máquina de Estado: Visualizar responsáveis

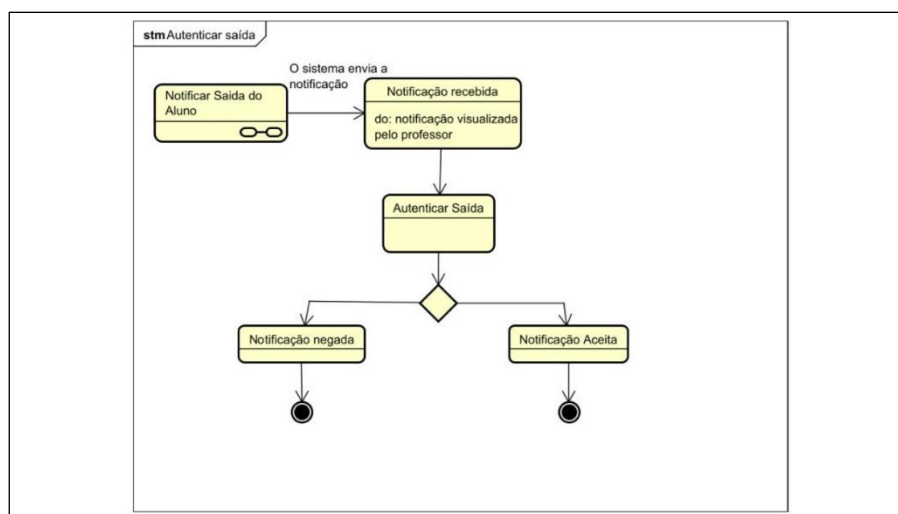


Fonte: Autoria própria, 2024.

O diagrama representa a consulta de responsáveis cadastrados. Carrega a lista do banco de dados e exibe detalhes ou mensagens caso

não encontre registros. O fluxo permite selecionar um responsável ou retornar ao menu principal.

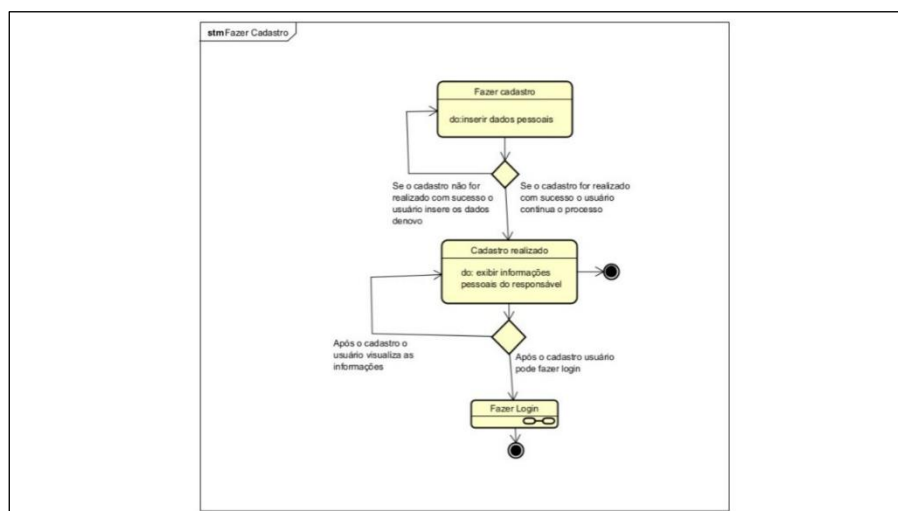
Figura 68 - Diagrama de Máquina de Estado: Autenticar saída



Fonte: Autoria própria, 2024.

Ilustra o processo de notificação e autenticação de saída de alunos. O sistema envia uma notificação que é visualizada pelo professor para aprovação ou negação.

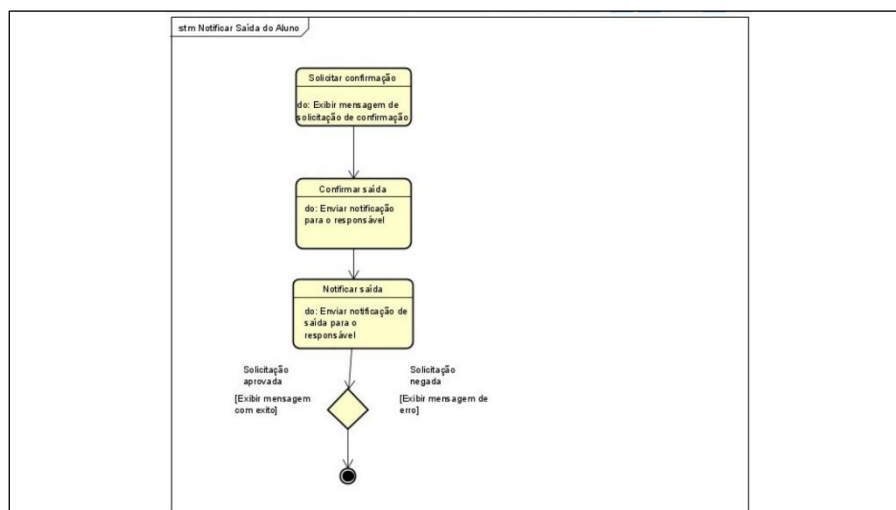
Figura 69 - Diagrama de Máquina de Estado: Fazer cadastro



Fonte: Autoria própria, 2024.

Explica o processo de cadastro no sistema. Após o cadastro, as informações do usuário são exibidas.

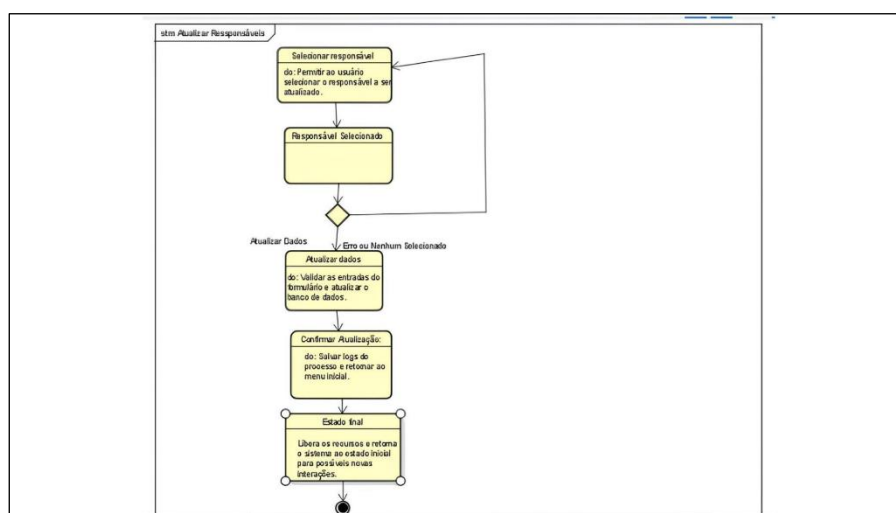
Figura 70 - Diagrama de Máquina de Estado: Notificar saída do aluno



Fonte: Autoria própria, 2024.

Descreve como uma solicitação de saída é gerenciada. Inclui a exibição de mensagens para confirmação, envio de notificações ao responsável, e retorno de mensagens indicando sucesso ou erro.

Figura 71 - Diagrama de Máquina de Estado: Atualizar responsáveis



Fonte: Autoria própria, 2024.

Detalha a atualização de informações de responsáveis. Permite selecionar um responsável, validar entradas no formulário, atualizar o banco de dados, e registrar *logs* do processo antes de retornar ao estado inicial.

Com a devida modelagem, o *back-end* foi programado em Node.js, permitindo a conexão com o banco de dados Firebase e a biblioteca OpenCV para suporte ao reconhecimento facial. Abaixo está o trecho principal do código.

Figura 72 - Código Reconhecimento Facial

```

/* Exibição do WebView se houver foto capturada */
/* Chamando o opencv atraves de um script para realizar a comparação */
{photoUri && ( <WebView originWhitelist={['*']} source={{
  html: `
    <html>
    <head><script src="https://docs.opencv.org/master/opencv.js"></script></head>
    <body>
    <script>
      cv['onRuntimeInitialized'] = () => {
        let img = new Image();
        img.src = '${photoUri}';
        img.onload = () => {
          const canvas = document.createElement('canvas');
          canvas.width = img.width;
          canvas.height = img.height;
          const ctx = canvas.getContext('2d');
          ctx.drawImage(img, 0, 0);
          const src = cv.imread(canvas);
          const gray = new cv.Mat();
          cv.cvtColor(src, gray, cv.COLOR_RGBA2GRAY);
          const classifier = new cv.CascadeClassifier();
          classifier.load('haarcascade_frontalface_default.xml');
          const faces = new cv.RectVector();
          classifier.detectMultiScale(gray, faces);
          const result = faces.size() > 0 ? 'Match encontrado!' : 'Nenhum match.';
          window.ReactNativeWebView.postMessage(result);
          src.delete();
          gray.delete();
          faces.delete();
        }
      }
    </script>
  `
}

```

Fonte: Autoria própria, 2024.

O código acima realiza a detecção facial na imagem capturada usando OpenCV no ambiente de um *WebView*, esse componente é para executar código JavaScript dentro de um ambiente de navegador embutido. Ele carrega e processa a imagem, executa a detecção de rostos e retorna o resultado para o aplicativo React Native. O processamento ocorre no lado do *WebView* porque a biblioteca OpenCV utilizada é baseada em JavaScript e não está diretamente integrada ao React Native.

Para garantir a segurança dos dados, implementamos medidas como autenticação de dois fatores e criptografia. Além disso, o sistema foi projetado para ser escalável, permitindo aumento gradual de usuários sem perda de desempenho.

O resultado foi avaliado positivamente, com uma taxa de conversão inicial de 100,8, mostrando uma boa aceitação, maior segurança e confiança no sistema. As estratégias de SEO, como otimização de palavras-chave e tags, foram aplicadas para melhorar o posicionamento nos mecanismos de busca.

4. CONCLUSÃO

Ainda que obtivamos a validação positiva do projeto, ainda existem uma gama de responsáveis que manifestam suas preocupações quanto à privacidade e proteção de dados, reforçando a importância de adotar políticas claras para garantir o uso responsável de informações e a proteção destas. Acreditamos que a implementação do projeto desde que bem conduzida e transparente quanto ao uso de dados será bem recebida, com relatos positivos dos gestores e funcionários, causando bons resultados de divulgação.

Foram identificados desafios com o custo de implantação que podem ser um pouco elevados quando levado em consideração o treinamento para o uso dele, isso pode dificultar a adoção em instituições menores ou com recursos limitados, concentrando em um público mais nichado. Este fator sugere a necessidade de considerar incentivos ou parcerias de empresas ou instituições para viabilizar o uso dessa tecnologia em maior escala.

Esses resultados demonstram que, embora o sistema seja eficiente e traga benefícios claros em termos de segurança, ainda possui pontos que necessitam ajustes e aprimoramentos, sendo cruciais para a perpetuação do projeto. O uso contínuo e a adaptação desse sistema têm grande potencial em transformar a segurança em creches, desde que sejam adotadas boas práticas de uso e conformidade com normas legais e éticas em paralelo.

Este estudo destaca a importância de aumentar a segurança nas creches por meio da tecnologia de reconhecimento facial, sendo importante ressaltar que a segurança nesses ambientes é não só essencial, mas

também é um direito constitucional, previsto no projeto de lei nº 221, de 24 de março de 2016, o que reforça a importância do projeto.

A proposta visa facilitar o controle de acesso, minimizar problemas como sequestros ou confusões de crianças, ao mesmo tempo, também reconhecendo a necessidade de respeitar a privacidade e os direitos dos responsáveis. É essencial que o uso dessa tecnologia seja acompanhado por normas claras e rígidas por parte da instituição, assegurando que a segurança seja reforçada sem comprometer os direitos fundamentais e manipulado com descrição pelos funcionários.

Destacamos que ainda existem pontos a ser melhorados, como a segurança dos dados, a necessidade de treinamento, o barateamento de implantação e a adaptação da instituição as tecnologias. Ainda que seja preciso constantes melhorias, com a ascensão desse tema e a relevância do sistema, acreditamos que no futuro ele seja amplamente aplicado e divulgado, tendo plataformas, empresas, grupos e ongs parceiras e patrocinadores.

Acreditamos que a aplicação cuidadosa desse sistema pode, de fato, aprimorar a dinâmica de segurança em creches, proporcionando mais tranquilidade para pais e responsáveis ao mesmo tempo que cria um ambiente educacional mais seguro, acolhedor e mais ágil durante esse processo.

REFERÊNCIAS

Barelli, Felipe. **Introdução à Visão Computacional: Uma abordagem prática com Python e OpenCV**. [S. l.: s. n.], 2018. E-book.

Booch, Martin. **UML essencial: um breve guia para a linguagem-padrão de modelagem de objetos**. 3 ed. São Paulo: Ltda, 2005.

Duckett, Jon. **HTML e CSS: projete e construa websites**. 1. ed. Rio de Janeiro: Alta Books, 2016.

Date, C.J. **Introdução a sistemas de bancos de dados**. Disponível em: https://books.google.com.br/books?id=xBeO9LSIK7UC&pg=PP23&source=kp_r ead_ button&hl=ptBR&newbks=1&newbks_redir=0&gboemv=1&ovdme=1&redir_esc=y#v=onepage&q&f=false. Acesso em: mar.2024.

Flanagan, David. **JavaScript, o guia definitivo**. 6 ed. São Paulo: Bookman, editora, 2012.

Flatschart, Fábio; **HTML 5 - Embarque Imediato**. 1 ed. Rio de janeiro: Brasport, 2011.

Guedes, Gilleanes T. A. **UML 2 - Uma Abordagem Prática**. 3ª ed. São Paulo: Novatec Editora, 2018.

Jobstraibizer, Flávia. **Criação de sites com CSS**. 1 ed. São Paulo: Universo dos Livros, 2009.

Larman, Craig. **Utilizando UML e Padrões**. 3 ed. São Paulo: Bookman, editora, 2007.

Mendonça, Ewerton. **Introdução a Web Design: Curso Técnico em Informática: Educação a distância**. Recife: Secretaria Executiva de Educação Profissional de Pernambuco, 2017.

Milleto, Evandro. **Desenvolvimento de Software II: Introdução ao desenvolvimento web, com HTML, CSS JAVASCRIPT E PHP**. Rio Grande do Sul: Instituto Federal de Educação e ciência e tecnologia, Bookman, 2014.

Moraes, W. B. **Construindo aplicações com NodeJS**. 1 ed. São Paulo: Novatec Editora, 2015.

Pereira, C. R. **Aplicações web real-time com NodeJS**. Editora, Casa do código: alura, 2014.

Pereira, Luiz Antônio de Moraes. **Análise e Modelagem de Sistemas com a UML Com Dicas e Exercícios Resolvidos**. 1. ed. Rio de Janeiro: Edição do Autor, 2011.

Pacheco, Marcel. **Desenvolvimento WEB HTML, CSS E JAVASCRIPT para iniciantes** [S. l.: s. n.], 2022. Disponível em: https://www.google.com.br/books/edition/DESENVOLVIMENTO_WEB_HTML_

CSS_

E_JAVASCRIPT/FFbNEAAQBAJ?hl=ptBR&gbpv=1&dq=livro+iniciantes+html&prints ec=frontcover/. Acesso em: abril. 2024.

Quierelli, D. A. **Criando sites com HTML-CSS-PHP: Construindo um projeto - Iniciante**. 1. ed. Santa Catarina: Clube de Autores, 2012.

Sereno, Galvão. **Comprehensive Repository Analysis of Mobile Projects Built with React Native**. Centro de Informatica. Recife: Universidade Federal de Pernambuco, 2018.

Silva, M. S. **JavaScript - Guia do Programador: Guia completo das funcionalidades de linguagem JavaScript**. São Paulo: Novatec Editora, 2010.

Silva, M. S. **React aprenda praticando: Desenvolva aplicações web reais com uso da biblioteca React e de seus módulos auxiliares**. São Paulo: Novatec Editora, 2021.

Samy, M. S. **Criando Sites com HTML: Sites de Alta Qualidade com HTML e CSS**. São Paulo: Novatec Editora, 2008.

Silva, A. L. **UML 2.5 do Requisito a Solução**. São Paulo: Saraiva Educação AS, 2005. E-book.

Staiano, Fábio. **Designing and prototyping interfaces with Figma: Learn essential ux/ui design principles by creating interactive prototypes for mobile, tablet, and desktop**. Birmingham: Packt Publishing, 2022.

Teixeira, Fabricio. **Introdução e boas práticas em UX Design**. São Paulo: Casa do Código, 2014. E-book.

Torres, V. M. (2018). **HTML E SEUS COMPONENTES**. Revista Ada Lovelace, 2, 99–101.

SÃO PAULO (Estado). Projeto de Lei nº 221, de 24 de março de 2016. **Torna obrigatória a instalação e a manutenção de sistema de monitoramento interno de vigilância eletrônica nas escolas infantis e creches, públicas ou privadas**. Assembleia Legislativa do Estado de São Paulo, 2016. Disponível em: <https://al.sp.gov.br>.

Villain, Mateus. **Figma: o que é a ferramenta, Design e uso**. [S. l.]. Alura, 20 jun. 2023. Disponível em: <https://www.alura.com.br/artigos/figma>. Acesso em: abril.2024.

Delai, R. L. Dutra, A. C. **Visão computacional com a openCV - Material apostilado e veículo autônomo**. Disponível em: <https://maua.br/files/082014/visao-computacional-opencv-material-apostilado-veiculo-seguir-autonomo.pdf>. Acesso em: jun.2024.

Fuentes, Guilherme Cardoso. **LightLow: Aplicativo simulador de consumo energético residencial**. 2023. Trabalho de Conclusão de Curso (Curso de

Bacharelado em Sistemas de Informação) - Faculdade De Ciências de Bauru, Bauru, 2023. Disponível em: <https://repositorio.unesp.br/handle/11449/239454>. Acesso em: jun. 2024.

Milano, D. Bazorro, L. H. **Visão computacional**. Disponível em: [https://d1wqtxts1xzle7.cloudfront.net/35825905/2010_IA_FT_UNICAMP_visao Comp utacional-libre.pdf?1417700841=&response-content disposition=inline%3B+filename%3DVISAO_COMPUTACIONAL_Palavras_Ch aves.pdf&Expires=1719690497&Signature=TIEizZ3ByoJ8dxkRoADbcnogtDw2X~yN ZrPok QEGG0yeudsi1dFe7sxdKPluGv7WmprcrL0DuFubyTGgi53YqNB61TFMxf2XkB 5U9s ZbuBMDCpKI~c7Agw95dZy8dXF5GVei5g4qftKIHX8niVcwb79QS1NWk42FV0H VEpq YnjNliS~csK4EYzDHiu~1yQtFFCkG78rYMsGuINHAaPgWHMOzZkoGjJKawRy 8ZzS-L4Dd8SkekK8lw4P-uvld105l2av1~- MLjuUBPrp7GXGV7UHLuxYQxDA4A8pbwKZc~HGKQnV2lipY3Pz3CoxD8rbU6 qqAN JdyiEyjFAZwyA__&Key-Pair-Id=APKAJLOHF5GGSLRBV4ZA](https://d1wqtxts1xzle7.cloudfront.net/35825905/2010_IA_FT_UNICAMP_visao%20computacional-libre.pdf?1417700841=&response-content-disposition=inline%3B+filename%3DVISAO_COMPUTACIONAL_Palavras-Chaves.pdf&Expires=1719690497&Signature=TIEizZ3ByoJ8dxkRoADbcnogtDw2X~yNzrPokQEGG0yeudsi1dFe7sxdKPluGv7WmprcrL0DuFubyTGgi53YqNB61TFMxf2XkB5U9sZbuBMDCpKI~c7Agw95dZy8dXF5GVei5g4qftKIHX8niVcwb79QS1NWk42FV0HVEpqYnjNliS~csK4EYzDHiu~1yQtFFCkG78rYMsGuINHAaPgWHMOzZkoGjJKawRy8ZzS-L4Dd8SkekK8lw4P-uvld105l2av1~-MLjuUBPrp7GXGV7UHLuxYQxDA4A8pbwKZc~HGKQnV2lipY3Pz3CoxD8rbU6qqANJdyiEyjFAZwyA__&Key-Pair-Id=APKAJLOHF5GGSLRBV4ZA). Acesso em: jun.2024.

Piteri, A. M. Rodrigues, C. J. **Fundamentos de Visão computacional**. Disponível em: [https://docs.fct.unesp.br/docentes/cartao/galo/web/Cap_Livro/2010_CLivro_WVC _Gal o_et al.pdf](https://docs.fct.unesp.br/docentes/cartao/galo/web/Cap_Livro/2010_CLivro_WVC_Gal o_et al.pdf). Acesso em: jun.2024.

Silva, G. J. **Análise comparativa de desempenho de banco de dados**. Disponível em: <https://dspace.uniube.br:8443/browse?type=author&value=Silva%2C+Gilmar+Jos%20C3%A9+da>. Acesso em: jun.2024.

Caldas, Joana, G1 SC. **Mulher se apresenta como avó, busca criança errada em escola e mobiliza polícia de 3 cidades em SC**. 2023. Disponível em: <https://www.googom/amp/s/g1.globo.com/google/amp/sc/antacatarina/noticia/2023/04/14/mulher-se-apresenta-como-avo-buscanca-errada-em-escola-e-mobiliza-policia-de-3-cidades-em-sc.ghml>. Acesso em: mar. 2024.

Eustáquio, Leandro. **A tecnologia como aliada da segurança nas escolas**. Disponível em: [https://canaltech.com.br/seguranca/a-tecnologia-como-aliada-daseguranca-nas-escolas-258528/#google_vignette](https://canaltech.com.br/seguranca/a-tecnologia-como-aliada-da-seguranca-nas-escolas-258528/#google_vignette). Acesso em: mar.2024.

FIREBASE. **Make your app the best it can be with Firebase and generative AI**. [S.l.]. FIREBASE. 2023. Disponível em: <https://firebase.google.com/?hl=pt>. Acesso em: mar.2024.

Npm Docs. Disponível em: <https://docs.npmjs.com/>. Acesso em: jun.2024.

IBM. **Modelos e Diagramas UML**. Disponível em:

<https://www.ibm.com/docs/pt-br/rsas/7.5.0?topic=models-uml-diagrams>. Acesso em: jun.2024.

Jennifer, Bittencourt. **O que é um Framework**. Disponível em:

<https://www.alura.com.br/artigos/framework-o-que-e-pra-que-serve-essa-ferramenta#:~:text=O%20framework%20nada%20mais%20%C3%A9,n%C3%A3o%20em%20detalhes%20de%20configura%C3%A7%C3%B5es>. Acesso em: jun.2024.

Esteves, Kaio, SBTInteior. **Araçatuba: criança é levada de creche por desconhecido, mas pais descobrem mal entendido**. 2023. Disponível em:

<https://sbtinterior.com/noticia/aracatuba-crianca-e-levada-de-creche-por-desconhecido-mas-pais-descobrem-mal-entendido,6113794075813.html>. Acesso em: ago.2024.

Ristow, SIMPAX. **Como o reconhecimento facial contribui para a segurança nas escolas. Gestão de Pessoas**. Disponível em:

<https://simpax.com.br/entenda-como-o-reconhecimento-facial-contribui-para-a-seguranca-nas-escolas/>. Acesso em: abril. 2024.

Noronha, Cristiano B. **O que é um Framework**. Disponível em:

<https://balta.io/blog/oque-e-um-framework>. Acesso em: mar.2024