

**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



**SUMÁRIO**

- 1. 10**
  - 1.1. 12**
  - 1.2. 12**
  - 1.3. 12**
- 2. 13**
  - 2.1. 13**
  - 2.2. 15**
  - 2.3. 17**
  - 2.4. 19**
  - 2.5. 19**
  - 2.6. 22**
  - 2.7. 22**
  - 2.8. 25**
  - 2.9. 25**
  - 2.10. 25**
  - 2.11. 28**
  - 2.12. 29**
  - 2.13. 30**
  - 2.14. 31**
  - 2.15. 31**
  - 2.16. 32**
  - 2.17. 33**
    - 2.17.1. 33**
    - 2.17.2. 34**
    - 2.17.3. 35**
    - 2.17.4. 36**
  - 2.18. 37**
- 3. 38**
  - 3.1. 38**
  - 3.2. 39**
- 4. 41**

## 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 delineadas 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, vêm 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.

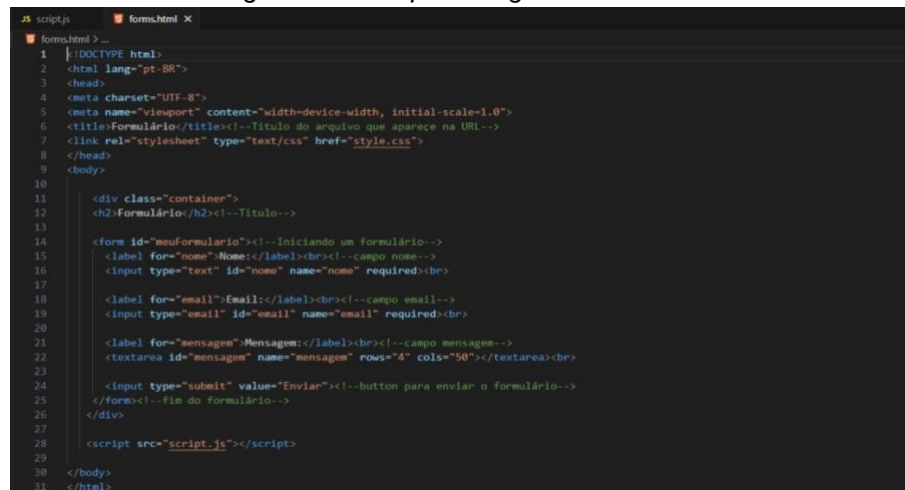
Conforme Duckett (2016) a base para uma criação de páginas, são as tags ou elementos, de modo que o navegador saiba o que você deseja colocar. As principais tags do HTML são:

- **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:

Figura 1 - Exemplo Código Básico HTML



```
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="meuformulário"><!-- 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"><!-- botão 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. 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 a baixo demonstrará o resultado da codificação sem a estilização:

Figura 2 - Resultado Código Básico HTML

## Formulário

Nome:

Email:

Mensagem:

Enviar

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

```
style.css
D:\> Exemp-TCC\HTML > style.css > ...
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

```
style.css
D:\> Exemp-TCC\HTML > style.css > ...
29
30 .container {
31   background-color: #fff;
32   padding: 20px;
33   border-radius: 8px;
34   box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
35   max-width: 400px;
36   width: 100%;
37   text-align: center;
38 }
39 h2 {
40   margin-bottom: 20px;
41   color: #333;
42 }
43 label {
44   display: block;
45   margin-bottom: 5px;
46   color: #888;
47   text-align: left;
48 }
49 input[type="text"],
50 input[type="email"],
51 textarea {
52   width: 100%;
53   padding: 10px;
54   margin-bottom: 20px;
55   border: 1px solid #ccc;
56   border-radius: 4px;
57   box-sizing: border-box;
58 }
59 input[type="submit"] {
60   background-color: #4CAF50;
61   color: white;
62   padding: 10px 20px;
63   border: none;
64   border-radius: 4px;
65   cursor: pointer;
66   transition: background-color 0.3s ease;
67 }
68 input[type="submit"]:hover {
69   background-color: #45a049;
70 }
```

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*



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 Millete (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*

A imagem mostra um editor de código com o tema escuro. No topo, há uma aba com o nome 'JS script.js' e um ícone de fechar. O código JavaScript está escrito em uma fonte monoespaçada com coloração de sintaxe. O código define um listener de evento para o botão de envio de um formulário, prevenindo o envio real e exibindo uma mensagem de sucesso no navegador.

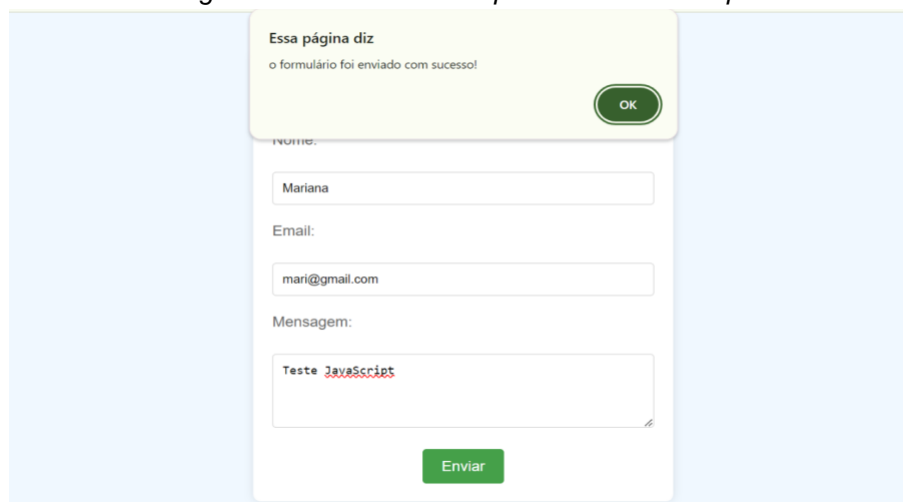
```
JS script.js x
JS script.js > ...
1 document.getElementById('meuFormulario').addEventListener('submit', function(event) {
2     event.preventDefault(); // Evita o envio real do formulário
3     alert('o formulário foi enviado com sucesso!');
4 })
```

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*

The image shows a web form interface. At the top, a yellow notification box contains the text "Essa página diz o formulário foi enviado com sucesso!" and an "OK" button. Below this, the form has three input fields: "Nome:" with the value "Mariana", "Email:" with the value "mari@gmail.com", and "Mensagem:" with the value "Teste JavaScript". A green "Enviar" button is located at the bottom of the form.

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.

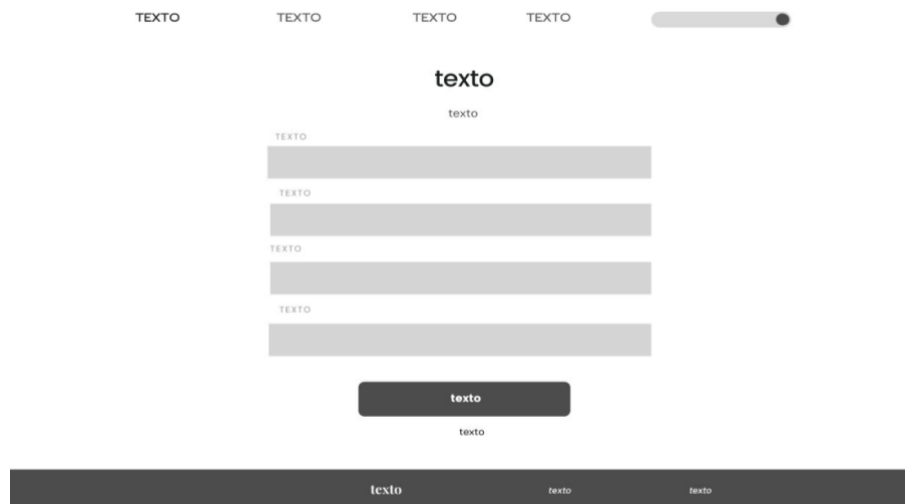
Consoante Mendonça (2017), para criar um design eficiente é necessário se planejar com antecedência, porque desenvolver uma aplicação sem ter um planejamento antes, é semelhante a construir uma casa sem ter uma planta de sua arquitetura.

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.

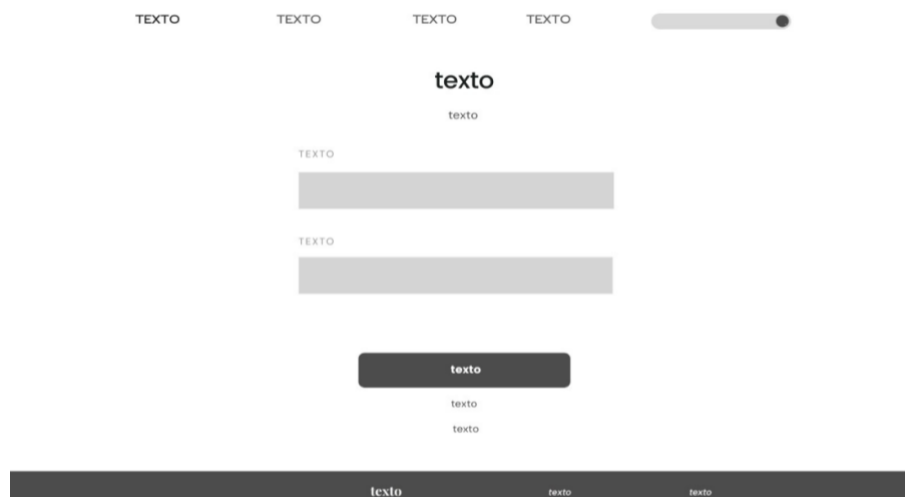
*Figura 8 - Exemplo Wireframe Baixa Cadastro*



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 "Crie sua Conta" é seguido por um link "Faça login para continuar.". O formulário principal contém campos para: NOME (preenchido com "Mariana"), EMAIL (preenchido com "mar@gmail.com.br"), SENHA (mascarada com pontos) e DATA DE NASCIMENTO (com um botão "Selecionar"). Um botão laranja "Sign up" está abaixo dos campos. Abaixo do botão, há um link "Já tem registro? Faça login.". Na base da página, uma barra laranja contém o texto "Entre em contato:", o e-mail "ola@grandeste.com.br", o telefone "(02) 3450-7890" e um ícone de WhatsApp.

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 atribuídas cores e símbolos à tela imaginada.

Figura 11 - Exemplo Wireframe Alta Login

The wireframe shows a login page layout. At the top is a header with navigation links: HOME, SOBRE NÓS, CONTATO, and PARCERIAS. Below the header is a login section titled 'Login' with the instruction 'Faça login para continuar.' It contains two input fields labeled 'NOME' and 'SENHA'. Below these fields is an orange 'Sign up' button, followed by a link 'Esqueceu sua senha?' and a 'Singup' link. At the bottom is an orange footer bar with the text 'Entre em contato:', the email 'alo@grandeste.com.br', the phone number '(02) 3456-7890', and a social media icon.

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 atribuídas 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 brainstorm e wireframes, proporcionando um código fonte em CSS, facilitando o desenvolvimento, sendo assim, o figma permite que crie ilustrações em forma vetorial.

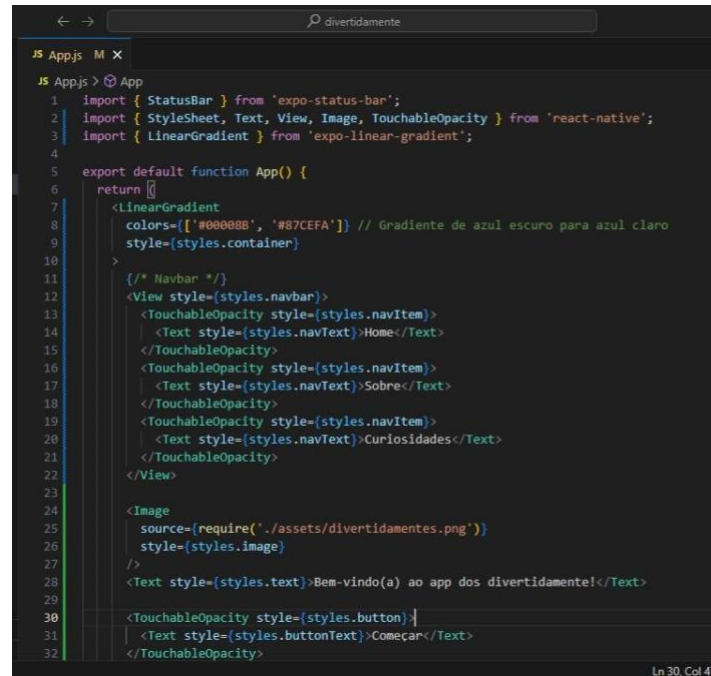
## 2.7. Expo

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

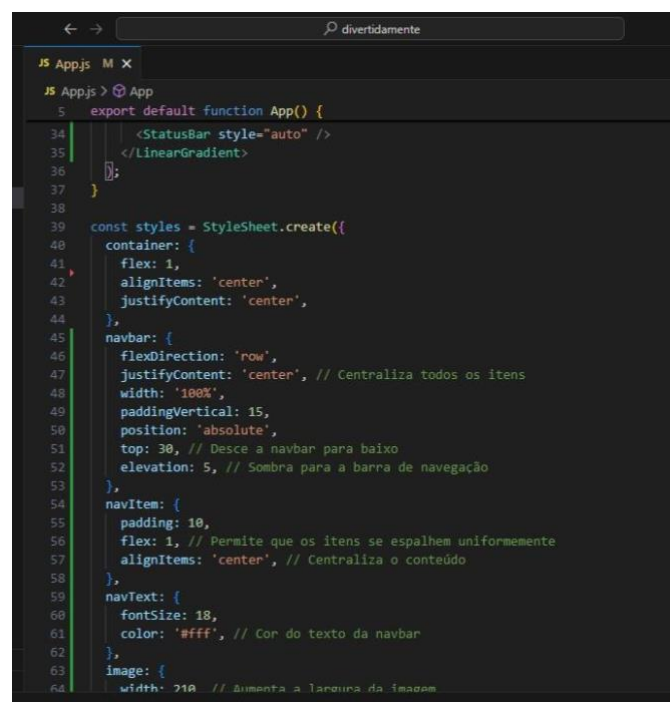


```
JS App.js M X
JS App.js > App
1 import { StatusBar } from 'expo-status-bar';
2 import { StyleSheet, Text, View, Image, TouchableOpacity } from 'react-native';
3 import { LinearGradient } from 'expo-linear-gradient';
4
5 export default function App() {
6   return (
7     <LinearGradient
8       colors={['#000000', '#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 }
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
Ln 30, Col 47
```

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

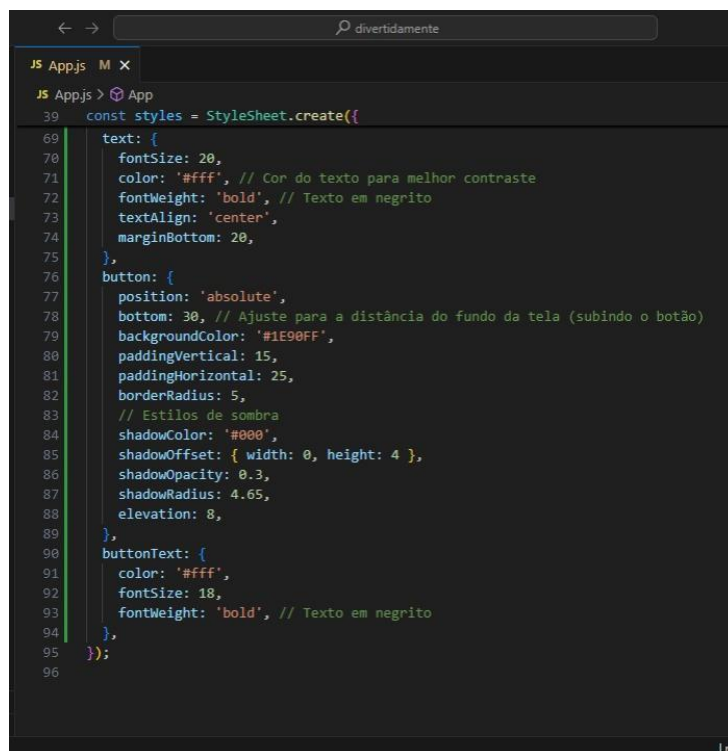


```
JS App.js M X
JS App.js > App
5 export default function App() {
34   <StatusBar style="auto" />
35   </LinearGradient>
36   </>
37 }
38
39 const styles = StyleSheet.create({
40   container: {
41     flex: 1,
42     alignItems: 'center',
43     justifyContent: 'center',
44   },
45   navbar: {
46     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
65   }
66 });
```

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*

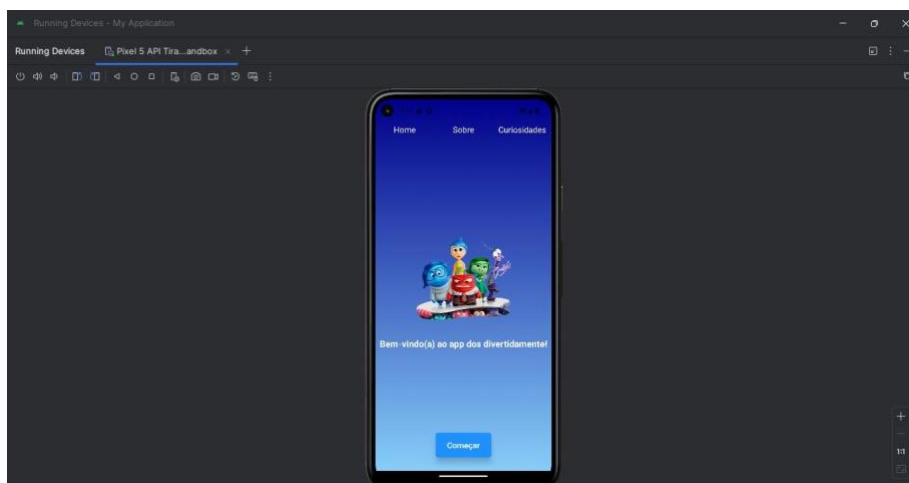


```
JS Appjs M X
JS Appjs > App
39 const styles = StyleSheet.create({
69   text: {
70     fontSize: 20,
71     color: '#fff', // Cor do texto para melhor contraste
72     fontWeight: 'bold', // Texto em negrito
73     textAlign: 'center',
74     marginBottom: 20,
75   },
76   button: {
77     position: '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 do 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.

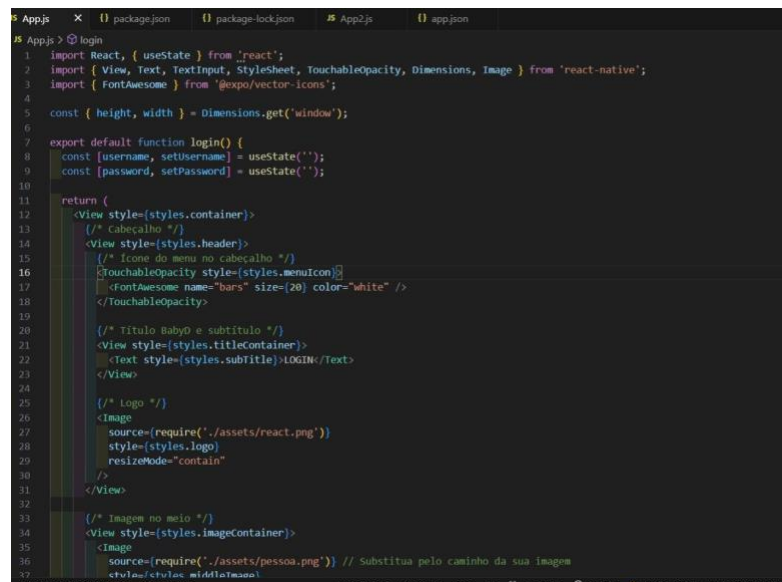
A seguir, algumas das principais funções do React:

- 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*



```

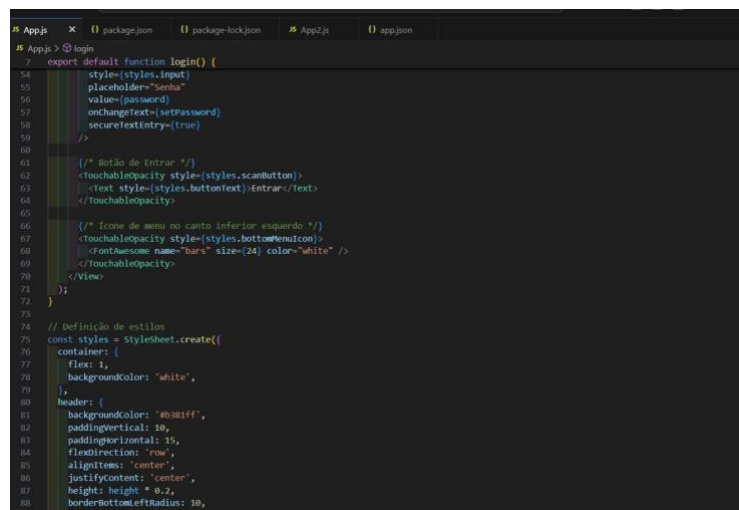
1 import React, { useState } from 'react';
2 import { View, Text, TextInput, StyleSheet, TouchableOpacity, Dimensions, Image } from 'react-native';
3 import { FontAwesomeIcon } from '@expo/vector-icons';
4
5 const { height, width } = Dimensions.get('window');
6
7 export default function login() {
8   const [username, setUsername] = useState('');
9   const [password, setPassword] = useState('');
10
11   return (
12     <View style={styles.container}>
13       <View style={styles.header}>
14         <TouchableOpacity style={styles.menuicon}>
15           <FontAwesomeIcon name="bars" size={20} color="white" />
16         </TouchableOpacity>
17
18         <View style={styles.titleContainer}>
19           <Text style={styles.subTitle}>LOGIN</Text>
20         </View>
21
22         <Image
23           source={require('./assets/react.png')}
24           style={styles.logo}
25           resizeMode="contain"
26         />
27
28         <View style={styles.imageContainer}>
29           <Image
30             source={require('./assets/pessoa.png')} // Substitua pelo caminho da sua imagem
31             style={styles.middleImage}
32           />
33         </View>
34       </View>
35     </View>
36   );
37 }

```

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*



```

7 export default function login() {
8   const [username, setUsername] = useState('');
9   const [password, setPassword] = useState('');
10
11   return (
12     <View style={styles.container}>
13       <View style={styles.header}>
14         <TouchableOpacity style={styles.menuicon}>
15           <FontAwesomeIcon name="bars" size={20} color="white" />
16         </TouchableOpacity>
17
18         <View style={styles.titleContainer}>
19           <Text style={styles.subTitle}>LOGIN</Text>
20         </View>
21
22         <Image
23           source={require('./assets/react.png')}
24           style={styles.logo}
25           resizeMode="contain"
26         />
27
28         <View style={styles.imageContainer}>
29           <Image
30             source={require('./assets/pessoa.png')} // Substitua pelo caminho da sua imagem
31             style={styles.middleImage}
32           />
33         </View>
34       </View>
35     </View>
36   );
37 }
38
39 // Definição de estilos
40 const styles = StyleSheet.create({
41   container: {
42     flex: 1,
43     backgroundColor: 'white',
44   },
45   header: {
46     backgroundColor: 'white',
47     paddingVertical: 10,
48     paddingHorizontal: 15,
49     flex-direction: 'row',
50     align-items: 'center',
51     justify-content: 'center',
52     height: height * 0.2,
53     borderBottomLeftRadius: 10,
54     borderBottomRightRadius: 10,
55   },
56   menuicon: {
57     width: 20,
58     height: 20,
59     margin: 0 auto,
60   },
61   titleContainer: {
62     width: 100,
63     height: 20,
64     margin: 0 auto,
65   },
66   subTitle: {
67     color: 'black',
68     font-weight: 'bold',
69   },
70   logo: {
71     width: 50,
72     height: 50,
73     margin: 0 auto,
74   },
75   imageContainer: {
76     width: 100,
77     height: 100,
78     margin: 0 auto,
79   },
80   middleImage: {
81     width: 50,
82     height: 50,
83     margin: 0 auto,
84   },
85   button: {
86     width: 100,
87     height: 40,
88     margin: 10 auto,
89     backgroundColor: 'black',
90     color: 'white',
91     borderRadius: 10,
92     display: 'flex',
93     align-items: 'center',
94     justify-content: 'center',
95   },
96   buttonText: {
97     color: 'white',
98     font-weight: 'bold',
99   },
100   bottomMenuicon: {
101     width: 20,
102     height: 20,
103     margin: 0 auto,
104   },
105 });

```

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*

```

75 const styles = StyleSheet.create({
76   container: {
77     // ...
78   },
79   header: {
80     backgroundColor: '#3333ff',
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  },
105  subtitle: {
106    fontSize: 30,
107    color: 'white',
108    marginTop: 5,
109    textAlign: 'center',
110  },
111  logo: {
112    width: 40,

```

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

```

X package.json  package-lock.json  App2.js  app.json
a @ tolin
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: '#8888ff',
    borderWidth: 1,
    borderRadius: 25,
    paddingLeft: 15,
    marginVertical: 10,
    alignSelf: 'center',
  },
});

```

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.

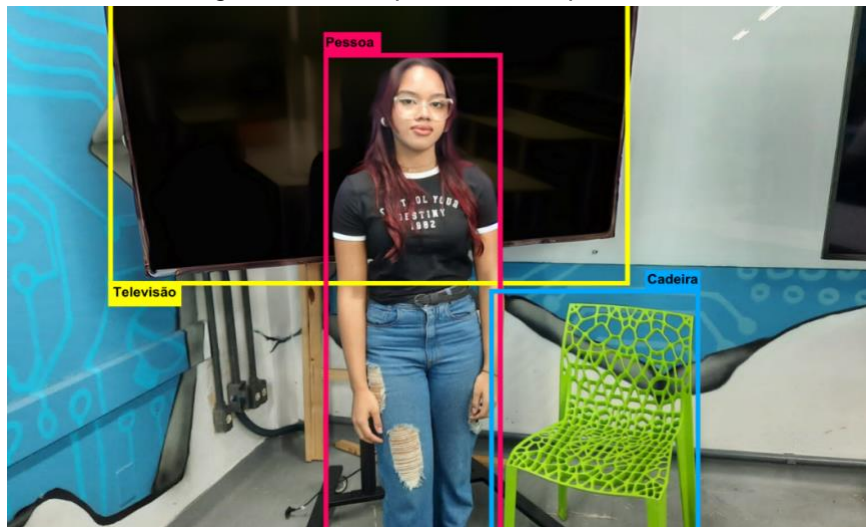
### **2.11. Visão Computacional**

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**

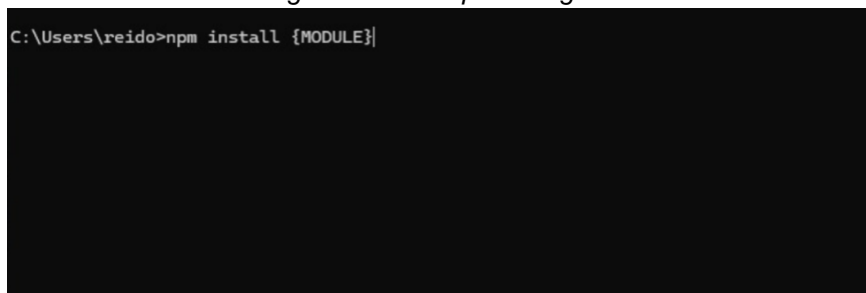
De acordo com a documentação oficial do Firebase (2023), é uma plataforma de Backend-as-a-Service (BaaS), que fornece infraestrutura de back-end pronta para quem desenvolve aplicativos, repleta de recursos, sem preocupações com hospedagem.

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*

A screenshot of a Windows command prompt window. The title bar is not visible. The text inside the window shows the command prompt path 'C:\Users\reido>' followed by the command 'npm install {MODULE}'. The text is white on a black background.

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.

*Figura 24 - Segundo Exemplo Código NMP*



Fonte: Autoria Própria, 2024.

Na figura 24, foi feito um exemplo do NPM (Node Package Manager) para a iniciação de um servidor de desenvolvimento. Abaixo uma breve explicação do código:

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 as ferramentas, linguagens e conceitos utilizados, conforme documentado anteriormente. 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.

#### **DESCREVER OS OBJETIVOS E O PUBLICO ALVO, DELIMITAÇÃO DE ESPAÇO E ETC.**

##### **3.1. 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.

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.

### **WIREFRAMES DE BAIXA**

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.

### **WIREFRAMES DE ALTA**

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.

### **PRINT DE TELA**

O resultado final do site foi avaliado positivamente, com uma taxa de conversão inicial de X%, 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.

## **3.2. 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.

## **WIREFRAMES DE BAIXA**

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.

## **WIREFRAMES DE ALTA**

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.

## **PRINT DE TELA**

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.

## **DIAGRAMAS UML**



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 estão trechos principais do código em Node.js.

## PRINT CÓDIGO

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 final foi avaliado positivamente, com uma taxa de conversão inicial de X%, 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**

## 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](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/](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.

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-seguidor-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 Computacional-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](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/cartogalo/web/Cap\\_Livro/2010\\_CLivro\\_WVC\\_Galo\\_et al.pdf](https://docs.fct.unesp.br/docentes/cartogalo/web/Cap_Livro/2010_CLivro_WVC_Galo_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%C3%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.ghtml>. 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-da-seguranca-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.I.]. 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