CENTRO ESTADUAL DE EDUCAÇÃO TECNOLÓGICA PAULA SOUZA

ETEC ZONA LESTE

Ensino Médio com Habilitação Profissional de Técnico em Desenvolvimento de Sistemas - AMS

Bruna Selis de Sousa

Bruno Silveira Dionisio

Danielle Reis Pereira

Gabriel Vinicius Silva

Guilherme Chaves Cordeiro da Silva

Henrique Lopes de Araujo

EROS: incentivo ao autocuidado através de uma aplicação mobile

São Paulo

2022

Bruna Selis de Sousa

Bruno Silveira Dionisio

Danielle Reis Pereira

Gabriel Vinicius Silva

Guilherme Chaves Cordeiro da Silva

Henrique Lopes de Araujo

EROS: incentivo ao autocuidado através de uma aplicação mobile

Trabalho de Conclusão de Curso apresentado ao Curso do Ensino Médio com Habilitação Profissional de Técnico em Desenvolvimento de Sistemas AMS da Etec Zona Leste, orientado pela Professora Vilma Cardoso dos Santos como requisito final para obtenção do título de Técnico em Desenvolvimento de Sistemas.

São Paulo

2022

DEDICATÓRIA

Dedicamos esse projeto a nós mesmos.

E aos demais que nos ajudaram dando força para chegarmos até aqui.

Enfim dedicamos a todos que fizeram parte dessa difícil etapa de nossas vidas.

AGRADECIMENTOS

A

Prof. Vilma Cardoso

Pela orientação e ajuda.

Prof. Marlon

Pelo alento e incentivo.

EPÍGRAFE

“Cada adversidade, cada fracasso, cada dor de cabeça carrega consigo a semente de um benefício igual ou maior.” (Napoleon Hill)

RESUMO

Esse trabalho busca apresentar um aplicativo com o intuito de demonstrar a forma correta de seguir cuidados capilares e estéticos no cotidiano do usuário. Com base na dificuldade que as pessoas têm em encontrar e seguir corretos cronogramas e cuidados com sua aparência, muitas vezes seguindo até mesmo meios contraindicados por especialistas, que acabam tendo um efeito não desejado e insatisfatório podendo até serem nocivos, buscamos solucionar isso através da criação de cronogramas capilares e rotinas de cuidados faciais personalizados, de acordo com o tipo de cabelo e pele informadas na hora do cadastro, sendo assim a aplicação tem a finalidade de promover uma forma simples e eficaz que possibilita o usuário alcançar os resultados que deseja, sem correr riscos tendo não só uma melhora estética, mas também na autoestima. O aplicativo foi desenvolvido com as ferramentas mais atuais do mercado.

Palavras-chave: React Native. JavaScript. Aplicativo Móvel. Estética. Beleza.

ABSTRACT

This work seeks to present an application in order to demonstrate the correct way to follow hair and aesthetic care in the daily life of the user. Based on the difficulty that people have in finding and following correct schedules and taking care of their appearance, often following even means contraindicated by specialists, which end up having an unwanted and unsatisfactory effect and may even be harmful, we seek to solve this by creating of hair schedules and personalized facial care routines, according to the type of hair and skin informed at the time of registration, so the application aims to promote a simple and effective way that allows the user to achieve the results he wants, without taking risks having not only an aesthetic improvement, but also in self-esteem. The application was developed with the most current tools on the market.

Keywords: React Native. Javascript. Mobile App. Aesthetics. Beauty.

LISTA DE ILUSTRAÇÔES

Figura 1 - Estrutura básica HTML 14

Figura 2- Código HTML do formulário 15

Figura 3 - Formulário simples HTML 17

Figura 4 - Atributo style no próprio elemento 18

Figura 5 - Propriedade CSS dentro da tag <style> 19

Figura 6 - Declaração do CSS com um arquivo externo 20

Figura 7 - Anatomia regra de estilo 20

Figura 8 - Código CSS do formulário 21

Figura 9 - Tela de cadastro estilizada com CSS 23

Figura 10 - Exemplo de função JavaScript 24

Figura 11 - Exemplo Objeto 25

Figura 12 - Exemplo evento JavaScript 25

Figura 13 - Exemplo do código criado no JSX 26

Figura 14 - Component React 27

Figura 15 - Tela de cadastro React Native 27

Figura 16 – Exemplo associação entre ator e caso de uso 29

Figura 17 – Exemplo de generalização de um ator 29

Figura 18 – Exemplo de estender o relacionamento entre dois casos de uso 30

Figura 19 – Exemplo de inclusão 30

Figura 20 - Estrutura Firebase 32

Figura 21 – Diagrama de Caso de uso 34

Figura 22 - Diagrama de Atividades: Login 35

Figura 23 - Diagrama de Atividades: Criação Cronograma para Pele 35

Figura 24 - Diagrama de Atividades: Criação Cronograma para o Cabelo 36

Figura 25 - Diagrama de Atividades: Perfil do Usuário 36

Figura 26 - Diagrama de Sequência: Cadastro 37

Figura 27 - Diagrama de Sequência: Login 38

Figura 28 - Diagrama de Sequência: Questionário Cabelo 38

Figura 29 - Diagrama de Sequência: Questionário Pele 39

Figura 30 - Diagrama de Sequência: Cronograma Cabelo 39

Figura 31 - Diagrama de Sequência: Cronograma Pele 40

Figura 32 - Tela Inicial 41

Figura 33 - Tela Registro 42

Figura 34 - Tela Login 43

Figura 35 - Tela Tipo Cabelo 44

Figura 36- Tela Tipo Cabelo A 45

Figura 37 - Tela Tipo Cabelo B 46

Figura 38 - Tela Tipo Cabelo C 47

Figura 39 - Tela Tipo Cabelo D 48

Figura 40 - Tela Tipo Cabelo 49

Figura 41 - Tela Tipo Pele 50

Figura 42 - Tela Problema Pele 51

Figura 43 - Tela Cronograma Capilar 52

Figura 44 - Tela Como Hidratar o Cabelo 53

Figura 45 - Tela Tutorial Nutrição Cabelo 54

Figura 46 - Tela Tutorial Reconstrução Capilar 55

Figura 47 - Tela Cronograma SkinCare 56

Figura 48 - Tela Tutorial Limpeza 57

Figura 49 - Tela Tutorial Esfoliação da Pele 58

Figura 50 - Tela Uso Correto Máscara 59

Figura 51 - Tela Hidratação Pele 60

Figura 52 - Tela Tutorial Proteção Facial.....61

Figura 53 - Tela Perfil Usuário.....62

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

Application Programming Interface (API)

Banco de Dados (BD)

Banco de Dados Não Relacional (NOSQL)

Cascading Style Sheets (CSS)

Document Object Model (DOM)

Hyper Text Markup Language (HTML)

JavaScript (JS)

JavaScript XML (JSX)

React Native (RN)

Standard Query Language (SQL)

Unified Modeling Language (UML)

SUMÁRIO

1. INTRODUÇÃO 12

2. REFERENCIAL TEÓRICO 13

2.1 HTML 13

2.2 CSS 18

2.3 JAVASCRIPT 23

2.3.1 JSON 25

2.3.2 REACT NATIVE 26

2.4 UML 27

2.5 Banco de Dados 31

2.5.1 Firebase 31

3. DESENVOLVIMENTO 33

3.1 LEVANTAMENTO DE REQUISITOS 33

3.2 Diagrama de Caso de Uso 34

3.3 Diagrama de Atividade 34

3.4 Diagrama de Sequência 37

3.5 TELAS 40

4. CONCLUSÃO 63

REFERÊNCIAS 64

1. INTRODUÇÃO

Atualmente, nota-se que a aparência se tornou muito mais do que apenas uma questão de estética, mas também uma questão de autoestima e autoconfiança. Com isso, muitas pessoas buscam melhorar esse aspecto para se destacarem em suas vidas pessoais e profissionais. No entanto, a maioria se vê insciente em relação aos cuidados com a aparência física, muitas vezes não tendo disponibilidade de tempo para procurarem e pesquisarem o que sua pele e cabelo precisam e quais os produtos ideais.

Sendo assim, a proposta deste trabalho é a criação de uma plataforma que oriente os cuidados faciais e capilares do usuário em seu cotidiano de forma individualizada, oferecendo ajuda ao utilizador para identificar seu tipo de pele e seu tipo de cabelo, estabelecendo um cronograma facial e capilar e indicando os produtos mais apropriados de acordo com os dados fornecidos pelo indivíduo.

Esse aplicativo foi desenvolvido com o intuito de trazer mais praticidade para o dia a dia dos usuários, para que assim, consigam cuidar de si mesmos de uma forma mais simples e fácil.

Para sua elaboração, foi utilizado o framework *React Native* para a construção da interface do usuário e a plataforma *Firebase* para banco de dados.

2. REFERENCIAL TEÓRICO

Este capítulo contém o embasamento teórico das tecnologias que serão utilizadas para a elaboração do projeto de pesquisa do aplicativo Eros.

2.1 HTML

Flatschart (2011) explica que o HTML (*HyperText Markup Language* – Linguagem de Marcação de Hipertexto) é a principal linguagem utilizada na web; ela permite a criação de documentos estruturados em títulos, parágrafos, formulários, tabelas, listas, links e em muitos outros elementos nos quais podem ser incorporadas imagens e objetos como, por exemplo, uma animação ou um vídeo.

De acordo com Almeida (2002), o HTML foi inicialmente concebido como uma solução para a publicação de documentos científicos em meios eletrônicos, ganhou popularidade e se tornou padrão para a Internet. Diversos tipos de aplicações, como navegadores, editores, programas de e-mail, bancos de dados e outros, tornam possível atualmente o uso intensivo da linguagem.

Ainda seguindo os ensinamentos de Flatschart (2011), outras linguagens podem ser incluídas juntamente ao HTML para trazer a página web desenvolvida mais interatividade e funcionalidade para com o usuário, como, por exemplo, o Javascript e o PHP que permitem o acesso a informações de banco de dados.

Segundo Tittel e Noble (2014), o HTML é um conjunto de *tags* responsáveis pela marcação do conteúdo de uma página no navegador. Há diversas *tags* e cada uma possui uma função ou significado específico. Uma *tag* é definida com caracteres < e >, sendo que algumas *tags* podem receber informações extras dentro de sua definição chamada de atributo.

De acordo com Pedroso (2007) é importante ressaltar algumas características da linguagem, como o fato de que nem todas as marcações e seus recursos são suportados por qualquer navegador. Arquivos HTML podem possuir as extensões ‘html’ ou ‘htm’, entretanto, os *browsers* são capazes de exibir documentos de ambas as extensões. O HTML não faz distinção entre maiúscula e minúscula, ou seja, não é *sensitive case*.

Pedroso (2007) continua dizendo que um programa HTML é dividido em três partes básicas: a estrutura principal, o cabeçalho e o corpo do programa. Definido assim,

todo programa deve começar com a *tag* <HTML> e ser encerrado com < /HTML>, sendo essa a estrutura principal. A área do cabeçalho fica entre as tags <HEAD> e </HEAD>. O corpo do programa é delimitado pelo par de tags <BODY> e < /BODY>.

A figura 1 representa uma estrutura básica HTML.

Figura 1 - Estrutura básica HTML

```
<HTML>
<HEAD>
<TITLE>Aqui deve ser colocado o título da página<TITLE>
</HEAD>
<BODY>
    Código HTML que fará o site aparecer.
</BODY>
</HTML>
```

Fonte: Autoria própria - 2022

- <!DOCTYPE html>: Não se trata de uma tag, mas sim de uma instrução especial. Quando não utilizada, pode ocorrer de algumas tags e estilizações não funcionarem corretamente, sendo essa instrução um indicador para o navegador utilizar a versão mais recente do HTML;
- <HTML>: É a tag de abertura da página, identificando o início de um documento;
- <HEAD></HEAD>: É a área reservada para o título da página e informações sobre o documento;
- <TITLE></TITLE>: Nesta tag, que deve ficar entre os comandos <HEAD> e </HEAD>, é onde será colocado o título da página, sendo esse título o que irá aparecer na barra do navegador;
- <BODY></BODY>: É onde será colocado a maioria dos comandos HTML, sendo essas tags que delimitam o corpo do programa.

De acordo com Silva (2008), formulários são projetados para que os dados fornecidos pelo usuário sejam coletados e processados por um script. Os campos

dos formulários na web são chamados de controles de formulário. Segue exemplo de um formulário em HTML na imagem 2:

Figura 2- Código HTML do formulário

```
<body>
      <div id="cadastro">
        <form method="post" action="">
          <h1>Cadastro Simples</h1>

          <p>
            <label id+"nome-label" for="nome>Nome Completo </label>
            <input id="nome" name="nome" required="required" type="text" placeholder="Nome" />
          <p>

          <p>
            <label for="cpf">CPF </label>
            <input id="cpf" name="cpf" required="required" type="text" placeholder="Apenas números" />
          </p>

          <p>
            <label for="data_nasc">Data de Nascimento</label>
            <input id="data_nasc" name="data_nasc" required="required" type="text" placeholder="DD/MM/AAAA" />
          </p>

          <p>
            <label for="email">E-mail </label>
            <input id="email" name"email" required="required" type="email" placeholder="contato@email.com"/>
          </p>

          <p>
            <label for="senha">Senha </label>
            <input id="senha" name="senha" required="required" type="password" placeholder="ex. 1234"/>
          </p>

          <p>
            <label for="confi_senha">Confrime a Senha </label>
            <input id="confir_senha" nome="senha" required="required" type="password" placeholder="ex. 1234"/>
          </p>
           <div class="buttons">
             <p>
               <button id="Cadastrar">Cadastrar</button>
               <button id="Cancelar">Cancelar</button>
             </p>
           </div>
        </form>
      </div>
    </div>
  </div>
</body>
</html>
```

Fonte: Autoria própria, 2022.

Segundo Silva (2008), essas são as funcionalidades de cada elemento que será citado abaixo:

- <form></form>: É o elemento utilizado para criar um formulário HTML para entrada do usuário;
- method=”post”: O atributo method especifica o método HTTP a ser usado ao enviar os dados do formulário. O método post é recomendado para ser utilizado quando os dados enviados pelo formulário forem confidenciais ou pessoais;

- action="": É um atributo que deve ficar entre as *tags* <form> e </form>, responsável por definir a ação que deve ser executada quando o formulário for enviado;
- <h1></h1>: Define um título;
- : É a *tag* que define um parágrafo, que sempre começa em uma nova linha e adiciona automaticamente um espaço em branco antes e depois do mesmo;
- <label></label>: Define uma legenda ou um rótulo para os demais elementos do formulário.
- for: Atributo da tag <label>, deve ser o mesmo que o atributo id para ligá-los;
- <input type="text">: Define um campo de linha única para entrada de texto;
- <input type="password">: Define um campo de senha;
- id: Atributo da tag <input>, deve ser o mesmo que o atributo for para ligá-los;
- value: Atributo da tag <input>, especifica um valor inicial para um campo de entrada;
- <button></button>: Tag utilizada para criar botões normais.

Segue código transferido para *web* na imagem 3:

Figura 3 - Formulário simples HTML

Fonte: Autoria própria – 2022.

2.2 CSS

De acordo com Jobstraibizer (2009), o termo *CSS*, proveniente de *Cascading Style Sheets,* que significa Folhas de Estilo em Cascata, é uma linguagem de estilo (ou de formatação) fantástica para construção do layout de suas páginas ou sites, sendo utilizada para definir a apresentação de documentos escritos em uma linguagem de marcação, como HTML ou XML.

Munzlinger (2010) diz os diversos benefícios do uso de CSS, como: a separação entre o conteúdo e seu aspecto/apresentação; incomplexidade de correção ou aperfeiçoamento do código das páginas do site; reutilização de estilos de formatação; concordância com os padrões internacionais W3C de desenvolvimento web.

Segundo Tittel (2014), há três formas de declararmos o CSS em um documento. A primeira delas é adicionando um atributo *style* no próprio elemento HTML que se deseja estilizar, onde é exemplificado na imagem 4.

Figura 4 - Atributo style no próprio elemento

```
<p> style="color: blue; background-color: yellow;">
O conteudo desta tag será exibido em azul com fundo amarelo no navegador!
</p>
```

Fonte: Autoria própria, 2022.

A segunda é declarando as propriedades do CSS dentro de uma tag <style> e indicando os elementos que nos referimos através de um seletor CSS, o qual é mostrada na imagem 5.

Figura 5 - Propriedade CSS dentro da tag <style>

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title> Sobre a MusicDot</title>
    <style>
      p {
        color: blue;
        background-color: yellow;
      }
    </style>
  </head>
  <body>
    <p>
      O conteudo desta tag será exibido em azul com fundo amarelo!
    </p>
    <p>
      <strong>Támbem</strong> será exibido em azul com fundo amarelo!
    </p>
  </body>
</html>
```

Fonte: Autoria propria, 2022

A terceira maneira e a considerada mais vantajosa é declarar os estilos do documento com um arquivo externo com a extensão .css. Porém, para declarar o CSS em um arquivo à parte é necessário indicar no documento HTML a ligação entre ele e o arquivo com extensão .css. A indicação deve ser feita dentro da tag <head>, sendo representada na imagem 6.

Figura 6 - Declaração do CSS com um arquivo externo

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>MusicDot | Sobre a empresa</title>
    <!-- Inclusão do arquivo CSS -->
    <link rel="stylesheet" href="estilos.css">
  </head>
  <body>
    <p>
      O conteudo desta tag será exibido em azul com fundo amarelo!
    </p>
    <p>
      <strong>Também</strong> será exibido em azul com fundo amarelo!
    </p>
  </body>
</html>
```

Fonte: Autoria própria, 2022.

Segundo Silva (2008), “Define-se regra de estilo como a menor porção do código capaz de causar efeito de estilização no conteúdo de um elemento da marcação. Uma regra de estilo mínima compõe-se de três partes distintas: um seletor, uma propriedade e um valor para a propriedade”. Veja a anatomia de uma regra de estilo na figura 7:

Figura 7 - Anatomia regra de estilo

```
seletor {propriedade: valor;}
```

Fonte: Autoria própria, 2022.

Ele prossegue dizendo que o seletor é o alvo da regra de estilo, sendo o que aponta para o elemento HTML que se deseja estilizar. A propriedade define o que será estilizado, e o valor define o quanto ou como será estilizado.

Segundo Collison (2008) pode-se dividir os seletores CSS em 5 categorias:

- Seletor simples (seleciona elementos com base no nome, id, classe);
- Seletor de combinação (seleciona elementos com base em um relacionamento específico entre eles);
- Seletor de pseudo-classe (seleciona elementos com base em um determinado estado);

- Seletor de pseudo-elementos (seleciona e estilize uma parte de um elemento);
- Seletor de atributo (seleciona elementos com base em um atributo ou valor de atributo).

O código do formulário da imagem anterior sendo estilizado com CSS pode ser observado na imagem 8.

Figura 8 - Código CSS do formulário

```
*{
    margin: 0;
    padding: 0:
    box-sizing; border-box;
    font=family: 'Roboto', sans-serif;
}
body{
    display: flex;
    align-items: center;
    justify-content: center;
    flex-direction: column;
    height: 100vh;
    background-image:linear-gradient(45deg,rgb(37, 33, 53), rgb(32, 81,243));
}
form{
    background-color: aliceblue;
    border-radius: .75rem;
    height: 100%;
    width: 20rem;
    display: flex;
    align-items: center;
    justify-content: space-around;
    flex-direction: column;
    padding: 1.5rem;
    margin: 5rem;
}
input{
    test-decoration: none;
    padding: .25rem;
    border-radius: .78rem;
    display: flex;
    flex-direction: column;
    align-items: center;
    margin: .2rem;
    margin-bottom: .78rem;
}
```

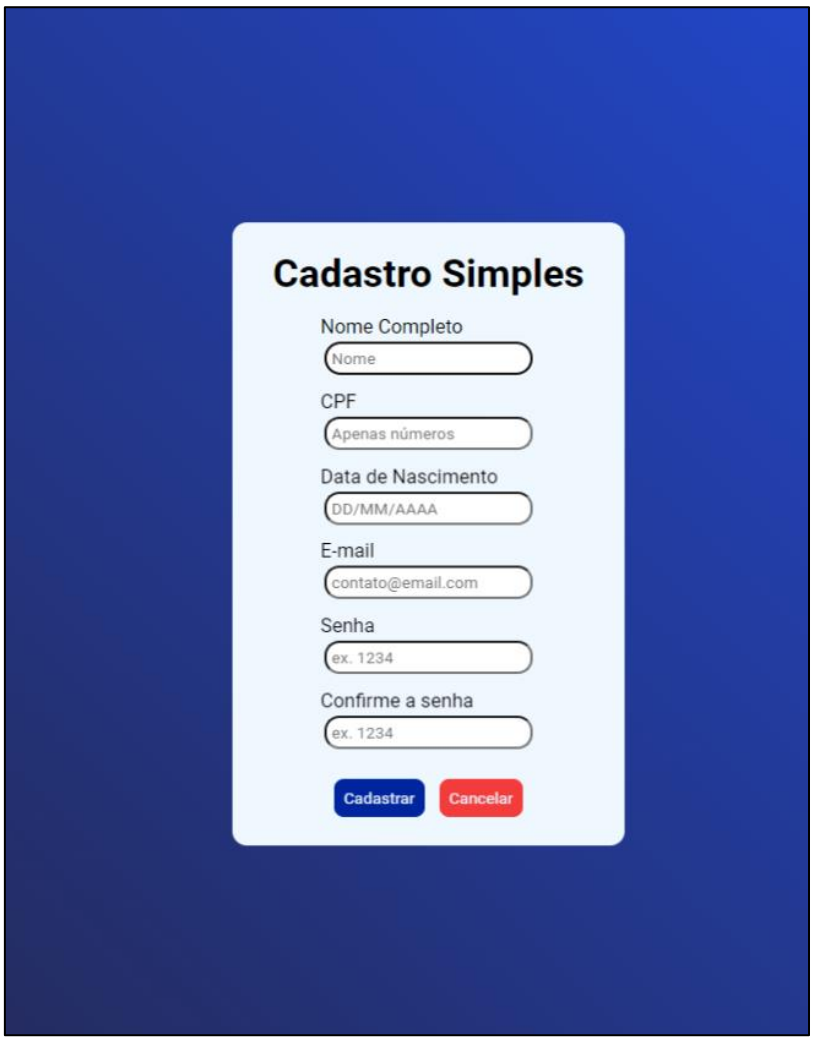
Fonte: Autoria própria, 2022.

- *Margin*: Utilizadas para criar espaços ao redor dos elementos;
- *Padding*: Tem a função de gerar espaço ao redor do conteúdo de um elemento;
- *Display*: Propriedade CSS mais importante de controle de *layout* que especifica se ou como o elemento será exibido;

- *Font-family*: Especifica a fonte do texto;
- *Background-color*: Adiciona uma cor de fundo;
- *Text-align*: Usado para definir o alinhamento horizontal de um texto;
- *Height*: Define a altura de um elemento;
- *Width*: Define a largura de um elemento;
- *Border-radius*: Propriedade utilizada para adicionar bordas arredondadas a um elemento.

Na imagem 9 se mostra o resultado na *web* da estilização vinda da ilustração anterior.

Figura 9 - Tela de cadastro estilizada com CSS



Fonte: Autoria própria, 2022.

2.3 JAVASCRIPT

Como explicado por Silva (2010), foi criado pela Netscape em parceria com a Sun MicroSystems em 1995, JavaScript tem a finalidade de fornecer um meio de adicionar interatividade a uma página web.

De acordo com Flanagan (2012), atualmente em sua 6° versão (ECMAScript 6) o Javascript é uma linguagem de programação de alto nível, presente em jogos, web, celulares, tablets, tv. Na web o HTML é responsável por exibir textos na tela, o CSS é responsável por estilizar a página e o Javascript é responsável pela interação da página, criando toda a parte logica e funcional.

Variáveis são todos os dados armazenados que possam ser acessados e tratados posteriormente, esses dados podem ser:

Números – Int, *float* ou *double*;

Textos – *String*;

Booleans - *true*, *false* (verdadeiro ou falso);

Arrays – dados armazenados em forma de lista.

Como Silveira (2013) e Almeida (2013) explicam, as variáveis em JavaScript não precisam ser declarativas, ou seja, não precisa ser passado a priori sendo referenciados apenas pelas palavras “var”; “let” ou “const” seguidas por um sinal de igual (=) e logo após o valor dessa variável (número, texto, *array* ou *boolean*).

Silva (2010) nos mostra que funções são blocos de ações pré-definidas com objetivo de execução de tarefa ou cálculo de valor, funções são definidas pela palavra “*function*” seguidas pelo nome da função e o sinal de parênteses para a passagem de parâmetros (valores que serão utilizados dentro da função) seguidos pela abertura e fechamento de chaves “{ }“, dentro dessas chaves que os argumentos da função estarão contidos, as quais indicam qual será a ação executada pela ação. Com as funções sendo exemplificadas dentro da imagem 10.

Figura 10 - Exemplo de função JavaScript

```
function square(numero) {
    return numero * numero;
}
```

Fonte: Autoria própria, 2022.

Voltando às explicações de Flanagan *(2012), Document Object Model* (DOM) é a forma como a linguagem de programação entende a tela que está sendo exibida, o que nos possibilita fazer alterações na mesma, toda a alteração na página exibida é uma alteração na DOM.

Como Silva (2010) explica, um objeto é a junção de dados, armazenando em uma unidade organizados em pares de nome e valor.

Novamente com Silveira (2013) e Almeida (2013), JavaScript é uma linguagem orientada a objetos, ou seja, uma linguagem que nos permite fazer a atribuição propriedades, a ligação de nomes a valores, que serão vistos e interpretados como um só, o nome dado a propriedade será entendido como o valor que lhe é atribuído. O exemplo de um objeto é mostrado na figura 11.

Figura 11 - Exemplo Objeto

```
var meuCarro = new Object();
meuCarro.fabricacao = "Ford";
meuCarro.modelo = "Mustang";
meuCarro.ano = 1969;
```

Fonte: Autoria própria, 2022.

Como Almeida (2015) explica, Módulos Javascript é como denominamos a forma de compartilhamento de informações entre arquivos, assim quando necessário é possível apenas importar o módulo que seja necessário utilizar para a criação do projeto.

Voltando com os ensinamentos de Silveira (2013) e Almeida (2013), eventos são ações pré-definidas pelo sistema que serão utilizadas quando requisitadas, exemplo, quando o usuário clica em um botão e uma informação é exibida na tela, essa informação foi exibida por causa da função ativada através do evento. Tais eventos são demonstrados na prática com a figura 12.

Figura 12 - Exemplo evento JavaScript

```
btn.onclick = function() {
    var rdnCol = 'rgb(' + random(255) + ',' + random(255) + ',' +
    random(255) = ')';
    document.body.style.backgroundColor = rdnCol;
}
```

Fonte: Autoria própria, 2022.

No exemplo acima é mostrada a função passado para o evento de clique no botão, essa função irá mudar a cor do botão quando ele for clicado.

2.3.1 JSON

Criado em 2006 por Douglas Crockford o *JavaScript Object Notation* (Notação de objetos JavaScript) ou JSON, é um subconjunto da linguagem de programação JavaScript, o JSON não é necessariamente uma linguagem de programação, mas sim um formato para troca de dados.

De acordo com Smith (2015), JSON pode ser considerado como um padrão para troca de dados, podendo ser utilizado como formato de dados quando houver alguma troca de dados. Uma troca de dados pode ocorrer tanto de cliente para o servidor como de servidor para servidor, mas o JSON não se limita somente a esses dois tipos de troca de dados, limitar o JSON para somente essas duas operações seria extremamente restritivo para a plataforma.

2.3.2 REACT NATIVE

Criado pelo Facebook em 2015, React Native é voltado para desenvolvimento de aplicativos móveis de multiplataforma. Por ser baseado no React, ele utiliza de muitas de suas funções, como elementos de interface de Virtual DOM e JSX.

Segundo Escudelario e Pinho (2020) React Native é um *Framework* (conjunto de códigos prontos, para auxiliar na criação de sites e aplicativos) Javascript para o desenvolvimento front-end (visual) de aplicações nativas mobile, ou seja, utiliza a linguagem de programação JavaScript para criar toda a parte visual de aplicativos tanto IOS(Apple) como Android(Google).

De acordo com Escudelario (2020) e Pinho (2020), JSX é uma extensão da sintaxe do Javascript que permite a “mistura" de HTML com JavaScript, sua utilização não é obrigatória no desenvolvimento, mas ao utilizá-la facilita a identificação de falha.

Segue a imagem de um código simples criado com JSX na imagem 13:

Figura 13 - Exemplo do código criado no JSX

```
<label>
    First Name:
    <input name="firstName" />
</label>

<label>
    Last Name:
    <input name="lastName" />
</label>
```

Fonte: Autoria própria, 2022.

Com o React Native é possível dividir o código em pequenas frações e trabalhar em cada uma de forma isolada, isso chamamos de componentes, semelhantes a funções JavaScript, com a diferença de que dentro de *components* é possível fazer “mistura” HTML, CSS e JavaScript de forma homogênea no código**.** Segue exemplo na imagem 14:

Figura 14 - Component React

```
function Welcome(props) {
    return <h1>Olá, {props.name}</h1>;
}
```

Fonte: Autoria própria, 2022.

Diferentes de outros frameworks com finalidade semelhante, com React Native todo o código desenvolvido, no final é convertido a linguagem nativa do sistema operacional, tornando a aplicação muito mais fluida ao usuário. Para exemplificar React Native na prática, foi desenvolvida uma tela simples de cadastro, sendo mostrada na imagem 15.

Figura 15 - Tela de cadastro React Native

Fonte: Autoria própria, 2022.

2.4 UML

Segundo Guedes (2011), 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 e que pode ser aplicada a todos os domínios da aplicação. O principal objetivo dessa linguagem é auxiliar na definição

das características, que podem e é indicado que sejam definidas antes do sistema ser desenvolvido, como estrutura lógica, requisitos, comportamento, a dinâmica de seus processos, entre outros.

De acordo com Booch (2006), para criar um *software* de qualidade duradoura e que satisfaça as necessidades do usuário é necessária uma arquitetura de fundação sólida que possa ser adaptado às novas mudanças do programa. Portanto, a modelagem é a parte central das atividades que levam à implantação de um bom sistema.

Ele prossegue dizendo as vantagens e objetivos alcançados com a modelagem, como por exemplo, a ajuda que os modelos fornecem na visualização de como o sistema é ou como se espera que seja, além de proporcionar um guia para a construção da aplicação e documentar as decisões tomadas.

Bezerra (2007) diz que há a fase de levantamento de requisitos que deve identificar dois tipos de requisitos: os funcionais e os não-funcionais. Os requisitos funcionais são aquilo que o cliente quer que o sistema realize, ou seja, as funcionalidades do *software*. Já os requisitos não-funcionais correspondem às restrições, condições, consistências e validações que devem ser levadas a partir do resultado sobre os requisitos funcionais. Desse modo surge a Regra de negócios que são basicamente impostos sobre os requisitos não funcionais, sendo políticas, normas e condições estabelecidas pela empresa que devem ser seguidas na execução de uma funcionalidade.

Como Larman explica (2007), o diagrama de caso de uso é um conjunto que representa toda a funcionalidade de um sistema capturando e modelando os requisitos funcionais. Ele é composto basicamente pelos atores, que são usuários que interagem com o sistema, podendo ser uma pessoa, organização ou sistema externo; o cenário, que se trata da sequência de eventos quando um usuário interage com o sistema; e a comunicação, que é o que liga um ator com um caso de uso.

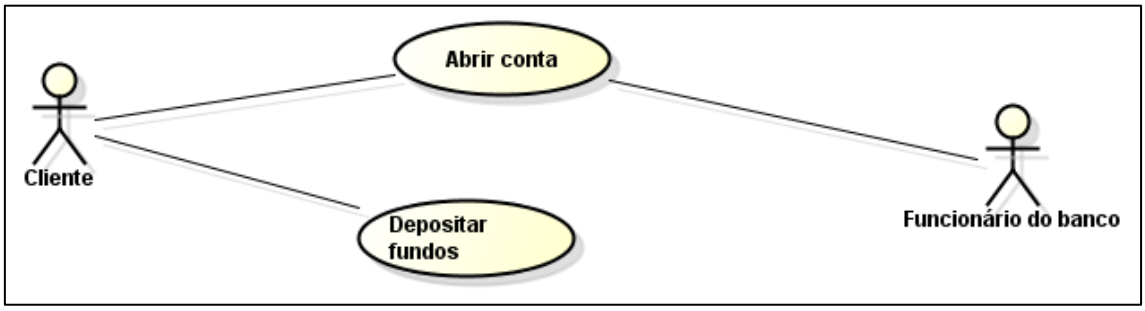
Segundo Lima (2014), os tipos de relacionamento entre casos de uso ou entre os atores e casos de uso são:

- Associação

Cada ator deve estar ligado a pelo menos um caso de uso, porém um ator pode estar ligado a vários casos de uso e vários atores podem estar ligados a um caso de uso.

A figura 16 apresenta um exemplo de caso de uso com associação entre ator e caso de uso:

Figura 16 – Exemplo associação entre ator e caso de uso



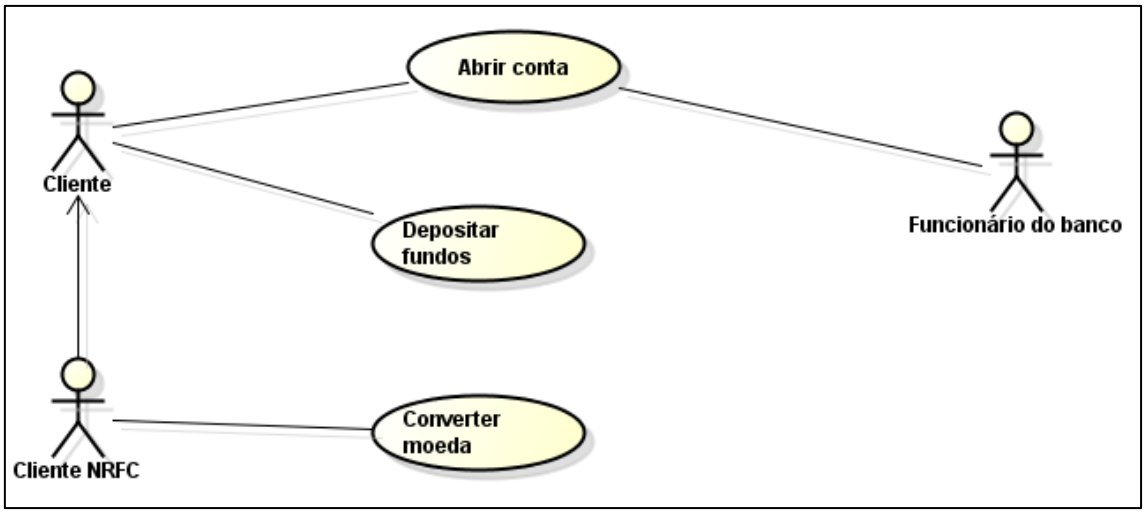
Fonte: Autoria própria, 2022.

- Generalização

Uma generalização indica que o primeiro é uma especialização do segundo. A seta de generalização é uma linha sólida com uma seta apontando para o caso de uso pai.

Veja um exemplo de generalização de um ator na figura 17:

Figura 17 – Exemplo de generalização de um ator



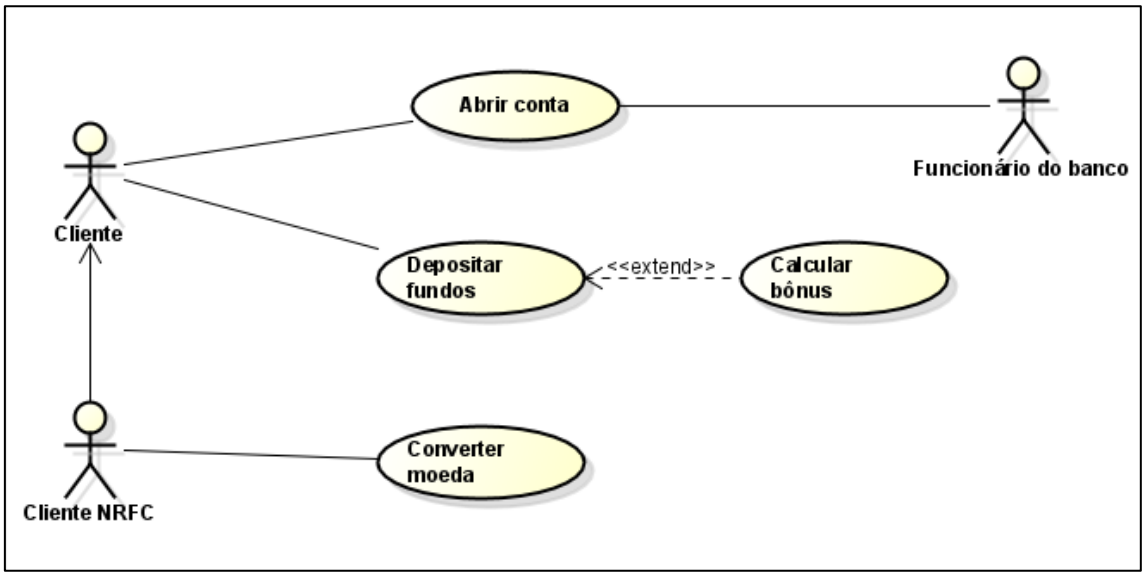
Fonte: Autoria própria, 2022.

- Extensão

Neste tipo de relacionamento é estendido o caso de uso e adicionado mais funcionalidade ao sistema. A extensão é representada por uma linha tracejada e rotulada com <<extend>>, apontando para o caso de uso estendido.

Na figura 18 é mostrado um relacionamento estendido:

Figura 18 – Exemplo de estender o relacionamento entre dois casos de uso



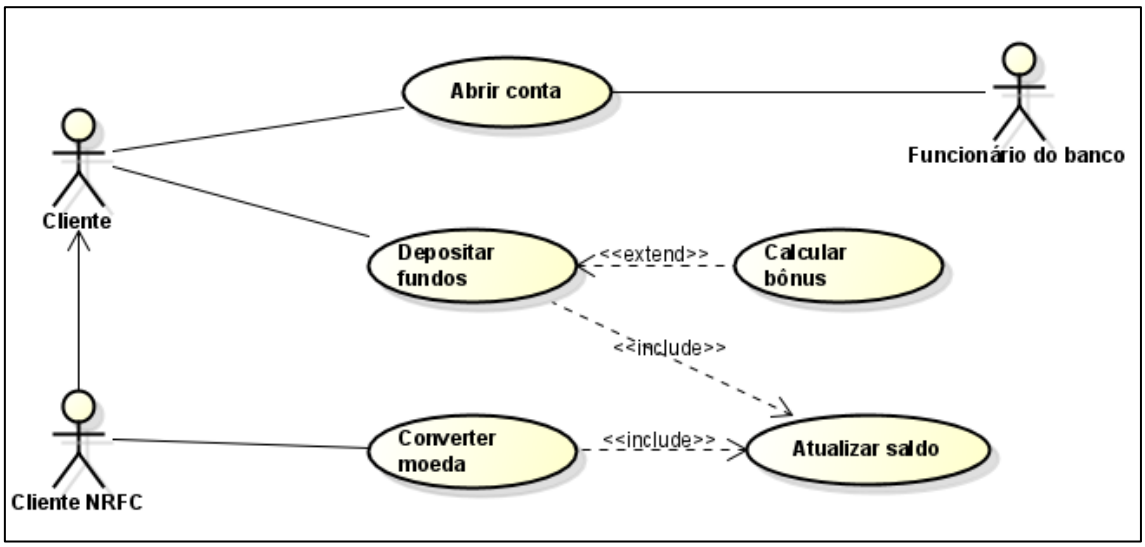
Fonte: Autoria própria, 2022.

- Inclusão

Como o próprio nome diz, um caso de uso inclui outro, sendo assim, quando um caso de uso for executado, o caso de uso incluído também será executado automaticamente.

Na figura 19 está sendo apresentado um exemplo de relacionamento de inclusão:

Figura 19 – Exemplo de inclusão



Fonte: Autoria própria, 2022.

2.5 Banco de Dados

Segundo Date (2004), um banco de dados é basicamente um sistema computadorizado de registros, pode ser considerado como o equivalente eletrônico de um armário de arquivos; ou seja, ele é um repositório ou recipiente para uma coleção de dados computadorizados. Os usuários deste sistema podem solicitar que o sistema realize diversas operações envolvendo tais arquivos, por exemplo: Acrescentar novos arquivos ao banco de dados, inserir, buscar, excluir e alterar dados em arquivos existentes.

Com as explicações de Ramakrishnan e Gehrke (2008), se entende que sistemas de gerenciamento de banco de dados nos dias de hoje são uma ferramenta indispensável para o gerenciamento de informações. Quando se está fazendo um banco de dados, temos a opção de fazer um banco de dados relacional, dando ao banco uma certa organização e meios de interação, o que pode parecer superior, mas não é necessariamente o caso, um banco de dados SQL pode ser mais lento, mais caro e mais complexo para armazenar números imensos de informações. E para isto que há opções mais viáveis como o banco de dados NoSQL, ou banco não relacional, mesmo que para muitos pela falta de complexos relacionamentos este possa parecer inferior, mas é uma ótima opção para guardar os dados, pois diferente de um banco SQL, este consegue ser muito mais rápido e fácil de programar e melhor para armazenamentos massivos, pois devido à falta de relacionamentos, acaba dando uma maior velocidade ao banco, o que pode ser muito favorável em caso de aplicações mais dinâmicas e rápidas.

Sendo explicado por Sadalage e Fowler (2019), Bancos de Dados NoSQL atuam sem o uso de um esquema, permitindo que sejam adicionados livremente campos aos registros do banco de dados, sem ter de definir primeiro quaisquer mudanças na estrutura base.

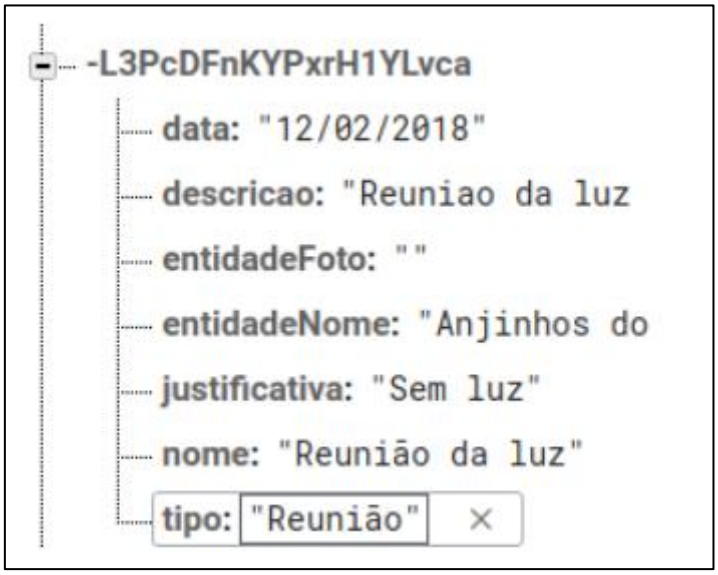
2.5.1 Firebase

Segundo Khennedy (2021), um dos principais dilemas que se pode ter quando se vai construir um *software* é a criação, organização e manutenção dos dados. E é para isto que o Google Firebase serve, ele é um banco de dados angular de tipo *NoSQL*, ou seja, não relacional. Este é um banco de dados com o foco para desenvolvedores mobile e web, providenciando recursos de atualização em tempo real, além da

segurança e uso dos servidores da própria Google, assim facilitando em muito o desenvolvimento de um banco de dados de uma empresa, e ainda por cima lhes dando uma variedade de funções e recursos para o uso.

De acordo com Cheng (2017), o Firebase tem uma estrutura bem diferente de outras bases de dados. Cada base de dados é guardada em uma árvore de objeto *JSON*. Esta estrutura de árvore é flexível com todo tipo de dado, como podemos ver na figura 20.

Figura 20 - Estrutura Firebase



Fonte: CHENG, 2017.

Como explica Kheronn (2021), sendo um banco angular, o Firebase é um framework JavaScript com construção SPA, ou seja, permitindo que quando um usuário faça alguma interação na página, a mesma não tenha que atualizar devido a sua função em que só usa uma única página para tal. Angular é uma plataforma para facilitar a construção de aplicativos, combinando *templates*, injeção de dependências, integrado às melhores práticas de desenvolvimento

3. DESENVOLVIMENTO

Este capítulo contém o desenvolvimento do aplicativo Eros.

3.1 LEVANTAMENTO DE REQUISITOS

Abaixo serão apresentados os requisitos funcionais, os requisitos não funcionais e as regras de negócio da aplicação.

1. Requisitos Funcionais

RF 1: Cadastro de usuário

RF 2: Login de Usuário

RF 3: Autenticação do usuário

RF 4: Requisição das informações do usuário

RF 5: Criação do cronograma de skincare

RF 6: Criação de cronograma capilar

RF 7: Listagem dos tratamentos

RF 8: Possibilidade de mudança das informações pessoais do usuário

2. Requisitos não funcionais

RNF 1: Cor padrão dos botões

RNF 2: Cor padrão do background das telas

RNF 3: Design intuitivo

RNF 4: Notificações do cronograma

RNF 5: Armazenamento disponível

RNF 6: Sistema operacional do aparelho

RNF 7: Versão do sistema operacional

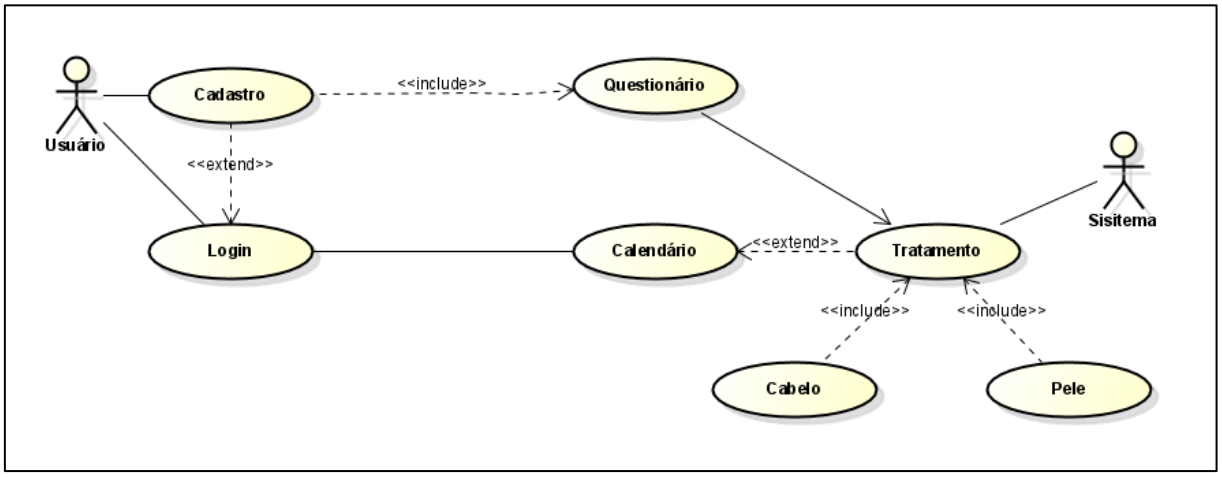
3. Regras de Negócio

RN: Um perfil por usuário

3.2 Diagrama de Caso de Uso

Na figura 21 está sendo representado o diagrama de caso de uso geral do aplicativo Eros, mostrando a interação do usuário com as funcionalidades do sistema, partindo do cadastro ou login, passando pelo questionário e chegando ao ponto principal, a proposta de tratamento e cronograma.

Figura 21 – Diagrama de Caso de uso



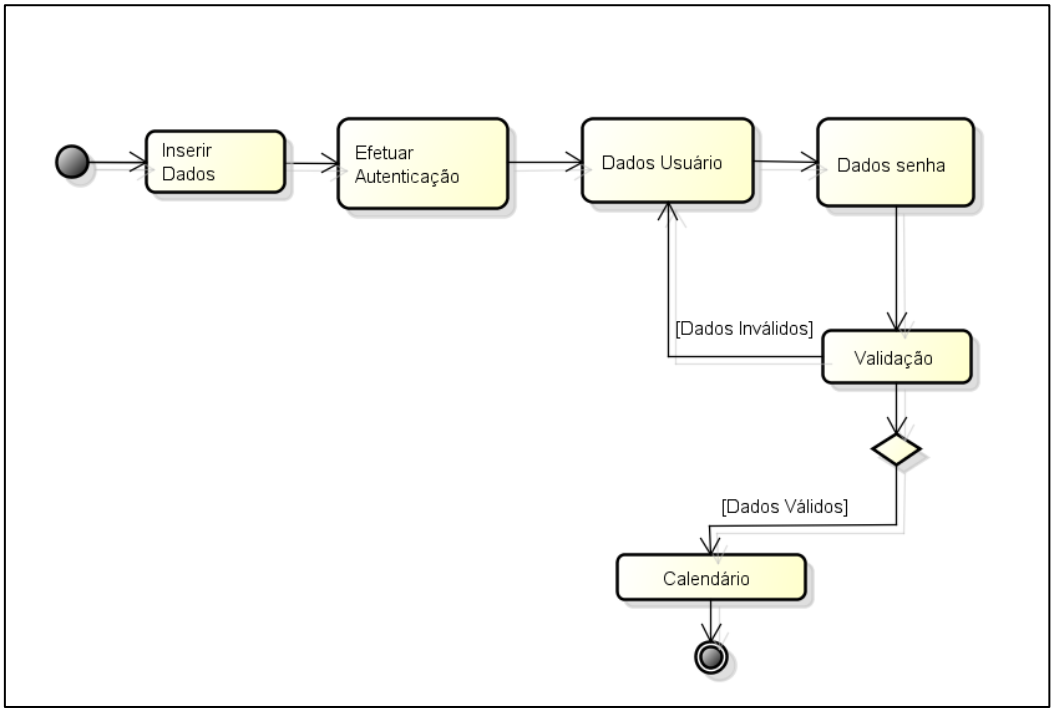
Fonte: Autoria própria, 2022.

3.3 Diagrama de Atividade

Os diagramas de atividade foram divididos em quatro imagens, ilustrando as atividades de realizar login, questionário de pele, questionário de cabelo e edição de perfil.

A figura 22 descreve, através das atividades o fluxo para realizar o acesso do usuário em sua conta, inserindo o seu nome de usuário e sua senha, tendo a validação de sua conta com o banco de dados.

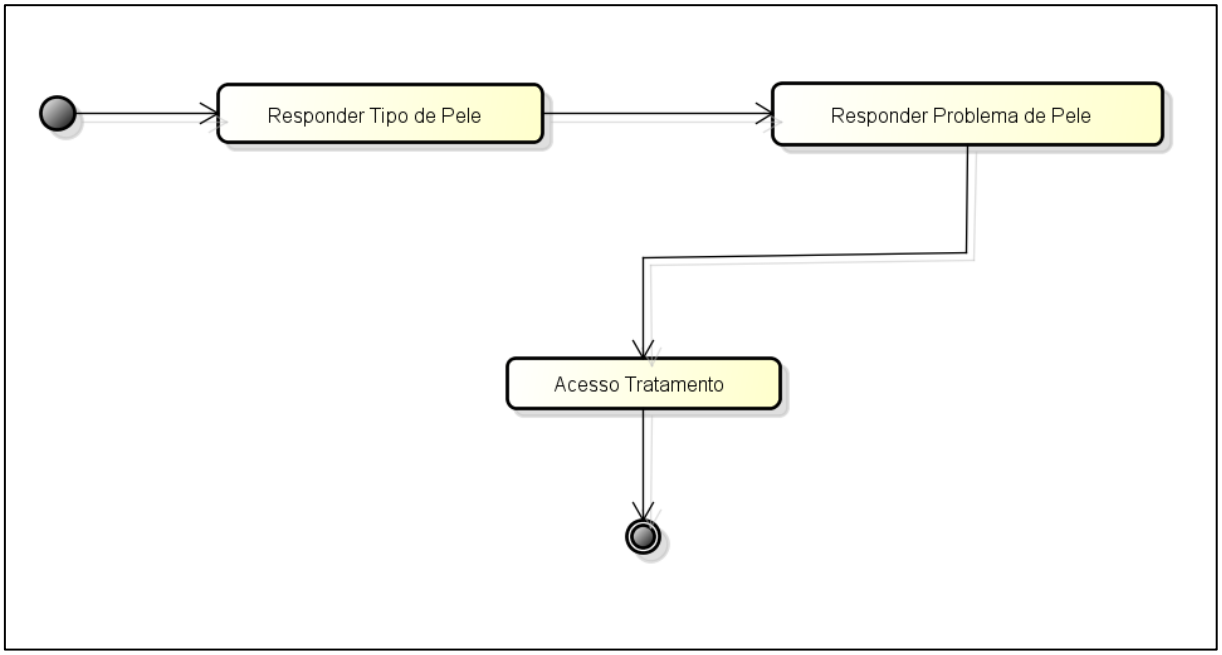
Figura 22 - Diagrama de Atividades: Login



Fonte: Autoria própria, 2022.

A imagem 23 descreve, através das atividades, o fluxo para realizar a criação do cronograma de tratamento para a sua pele.

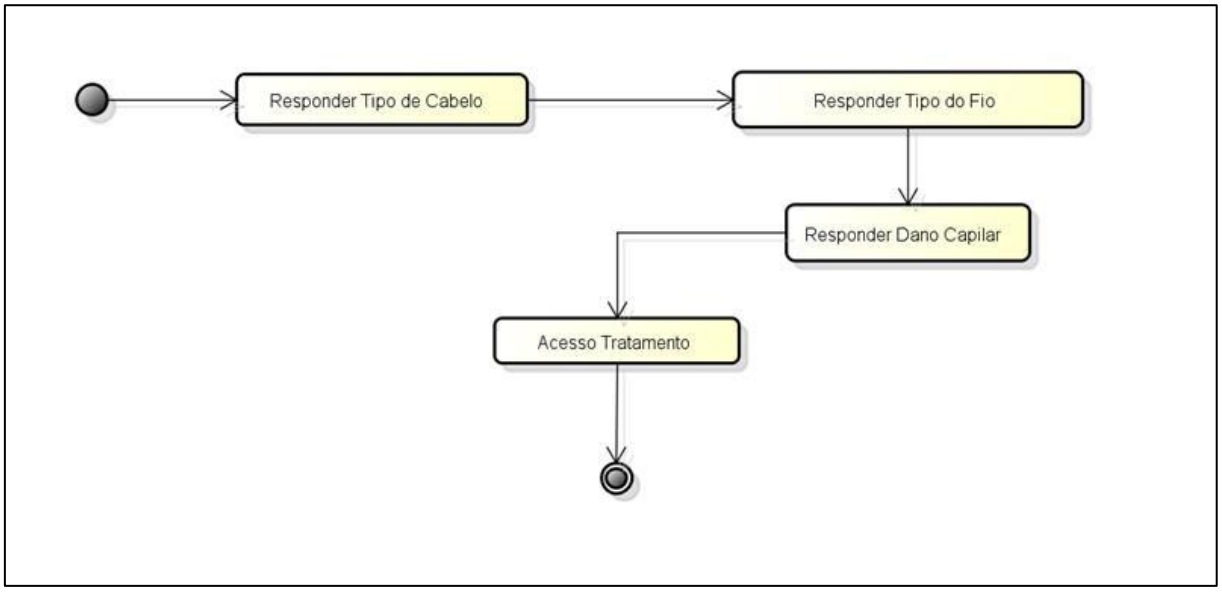
Figura 23 - Diagrama de Atividades: Criação Cronograma para Pele



Fonte: Autoria própria, 2022.

A imagem 24 descreve, através das atividades o fluxo para realizar a criação do cronograma de tratamento para o seu cabelo.

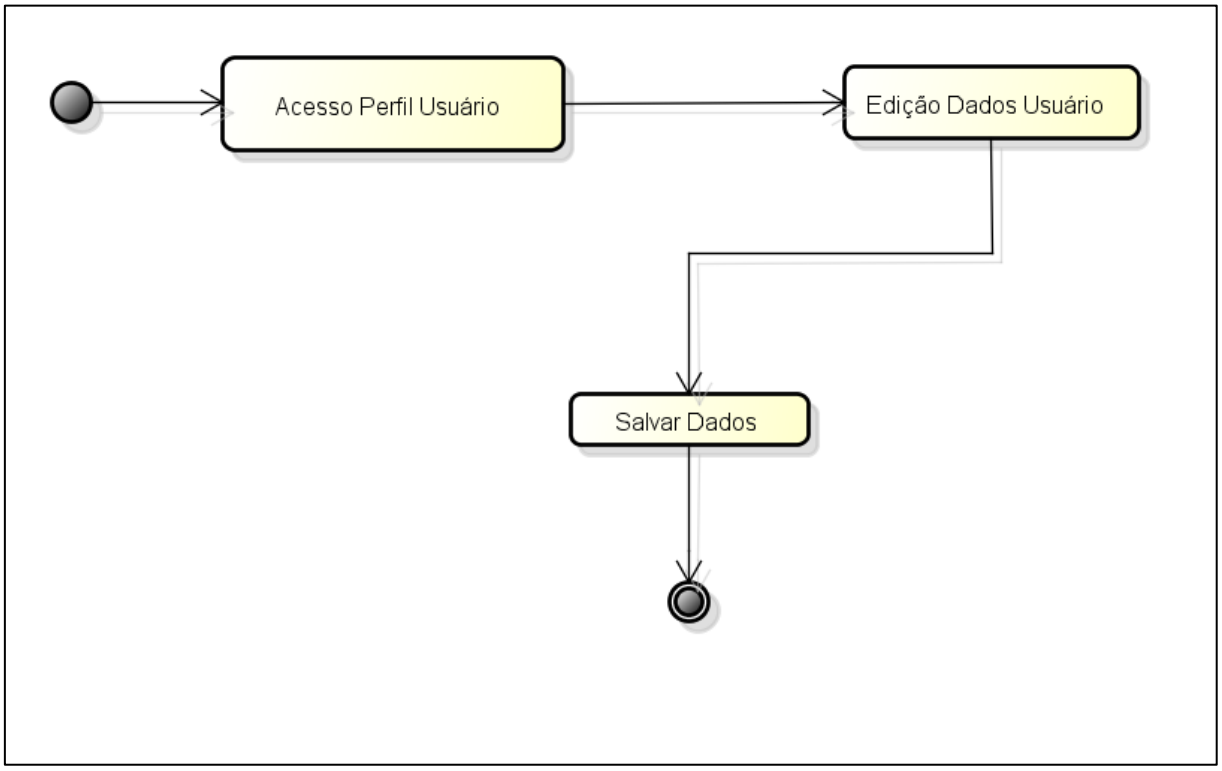
Figura 24 - Diagrama de Atividades: Criação Cronograma para o Cabelo



Fonte: Autoria própria, 2022.

A imagem 25 descreve, através das atividades, o fluxo para realizar a alteração de suas informações em seu perfil, tendo sua alteração também no banco de dados.

Figura 25 - Diagrama de Atividades: Perfil do Usuário



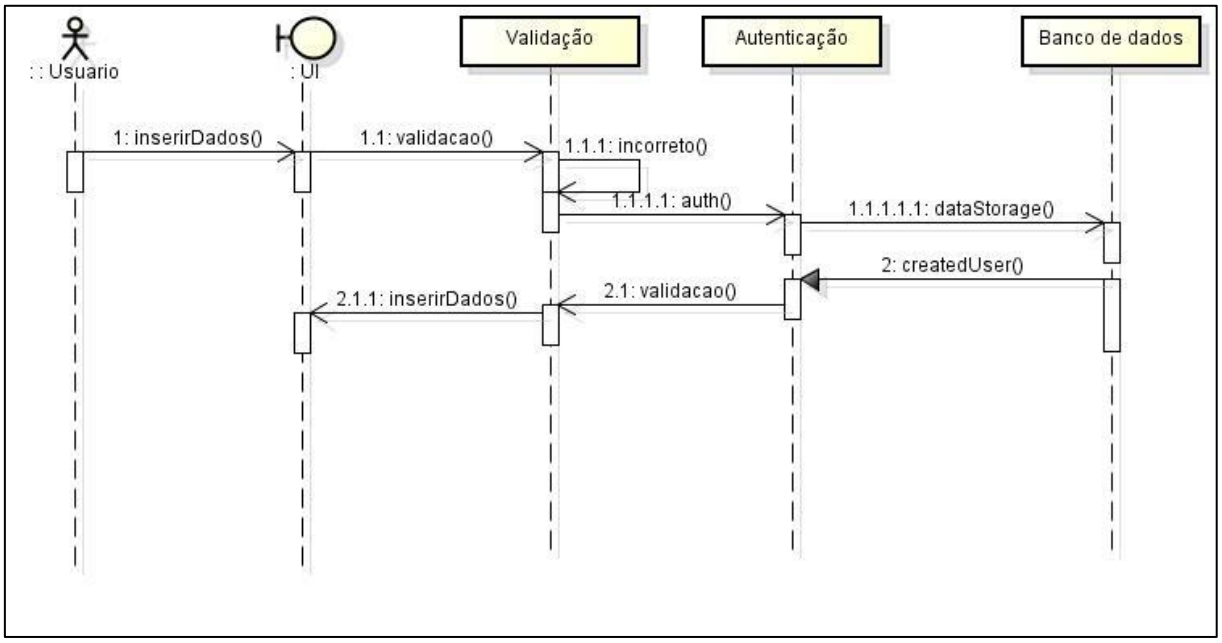
Fonte: Autoria própria, 2022.

3.4 Diagrama de Sequência

Os diagramas de sequência foram divididos em seis partes para melhor entendimento do projeto, sendo eles: cadastro, *login*, questionário para reunir informações da pele e cabelo do usuário e verificação do cronograma de pele e cabelo do usuário.

Na imagem 26, pode-se observar pelo diagrama o processo para a realização do cadastro de uma nova conta na aplicação, ilustrando passo a passo do processo, com a inserção no banco até a confirmação para o usuário de que foi efetuada o cadastro.

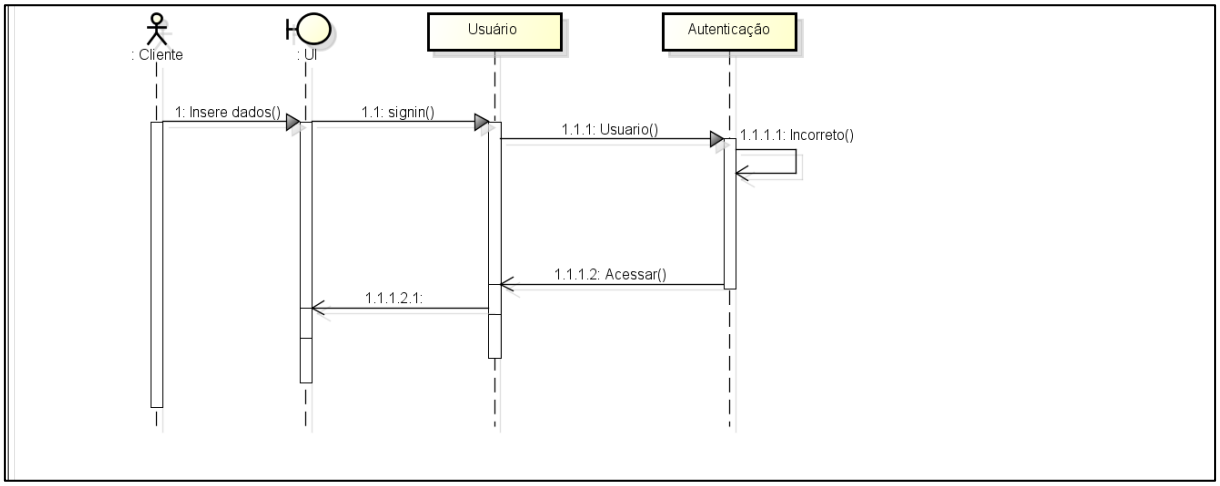
Figura 26 - Diagrama de Sequência: Cadastro



Fonte: Autoria própria, 2022.

Na imagem 27, pode ser observado pelo diagrama o processo para a realização completa do *login* do usuário, desde a inserção de dados até a confirmação de seus dados.

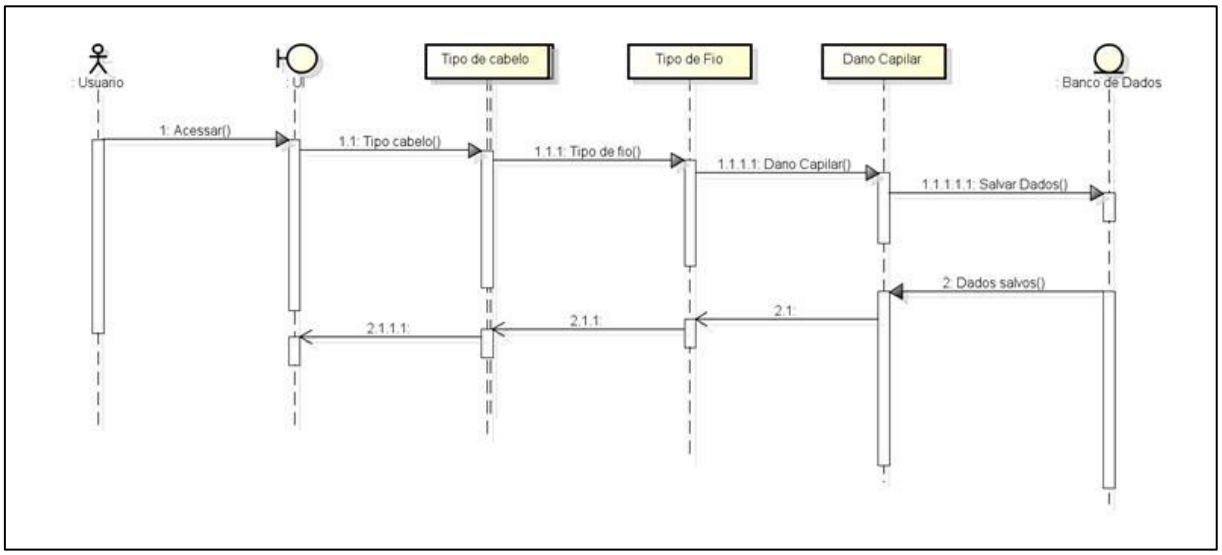
Figura 27 - Diagrama de Sequência: Login



Fonte: Autoria própria, 2022.

O questionário sobre a criação do tratamento sobre o cabelo do usuário pode ser observado na imagem 28, sendo ilustrado o respondimento das perguntas até o armazenamento dos dados no banco de dados da aplicação.

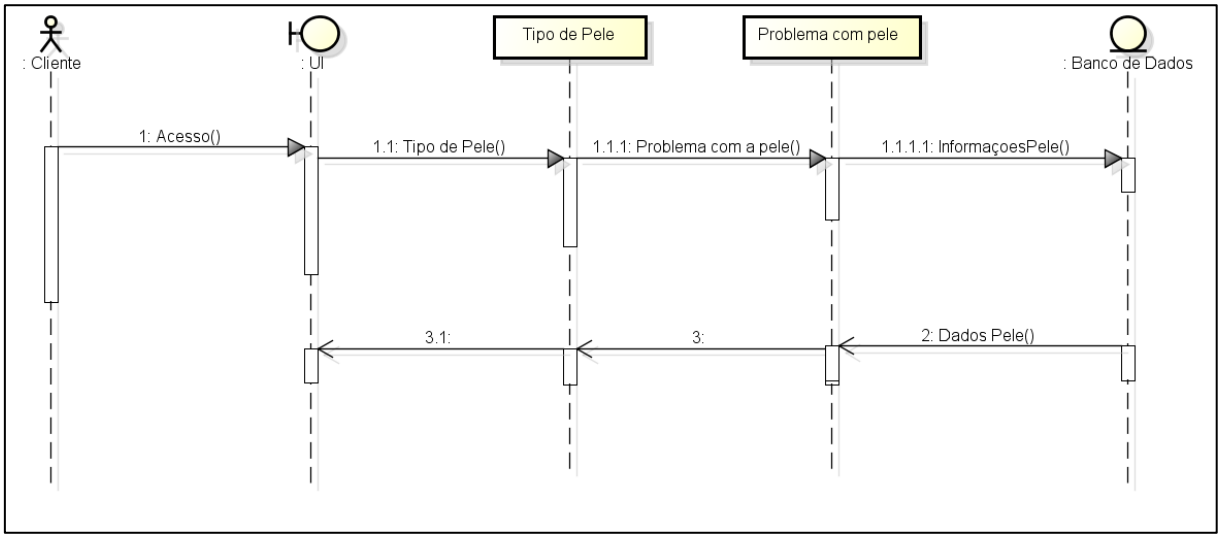
Figura 28 - Diagrama de Sequência: Questionário Cabelo



Fonte: Autoria própria, 2022.

O questionário sobre a criação do tratamento sobre a pele do usuário pode ser observado na imagem 29, sendo ilustrado o respondimento das perguntas até o armazenamento dos dados no banco de dados da aplicação.

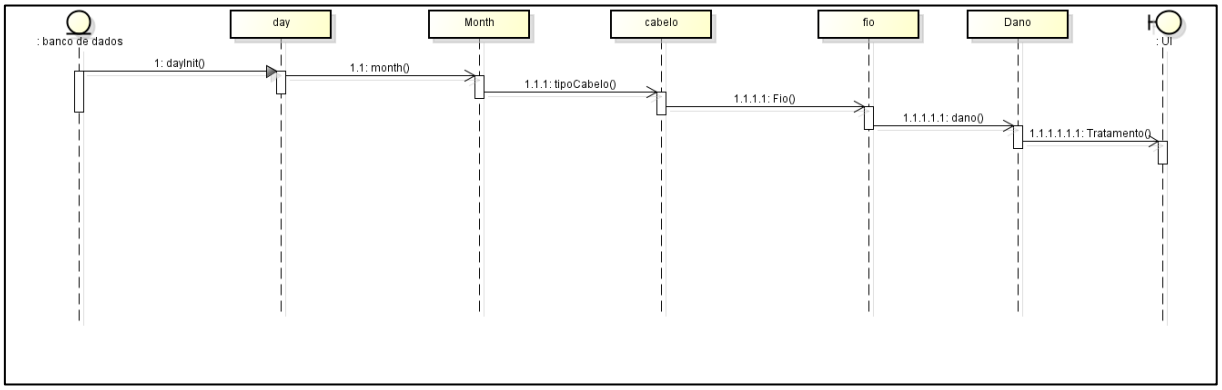
Figura 29 - Diagrama de Sequência: Questionário Pele



Fonte: Autoria própria, 2022.

A imagem 30 apresenta os passos para que seja mostrado ao usuário o cronograma e tratamento para seu cabelo.

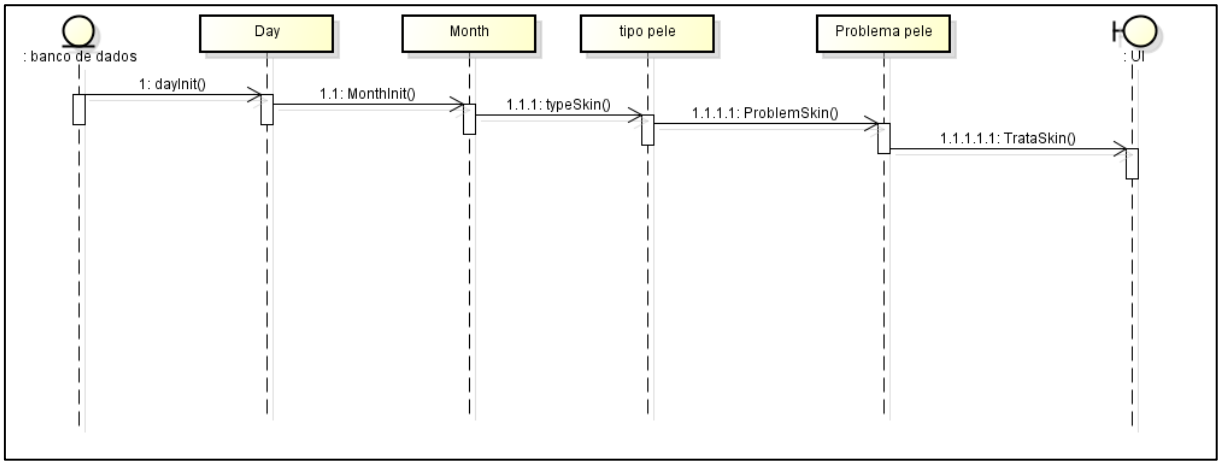
Figura 30 - Diagrama de Sequência: Cronograma Cabelo



Fonte: Autoria própria, 2022.

A imagem 31 apresenta os passos para que seja mostrado ao usuário o cronograma e tratamento para sua pele.

Figura 31 - Diagrama de Sequência: Cronograma Pele



Fonte: Autoria própria, 2022.

3.5 TELAS

Nesta seção serão apresentadas as telas do aplicativo Eros, contendo as informações para que se torne ainda melhor a experiência do usuário.

A tela inicial de nosso aplicativo se abre com uma breve animação, disponibilizando dois botões que servem para acessar à aplicação, juntamente do logo de nosso projeto, igualmente mostra na imagem 32 a seguir:

Figura 32 - Tela Inicial

Fonte: Autoria própria, 2022.

Ambos os botões contêm a função de levar o usuário para a tela desejada, sendo tanto para a tela de entrar com suas informações quanto para a de cadastro de uma nova conta. De acordo com a imagem 33 pode se observar como o usuário pode se registar.

Figura 33 - Tela Registro

Fonte: Autoria própria, 2022.

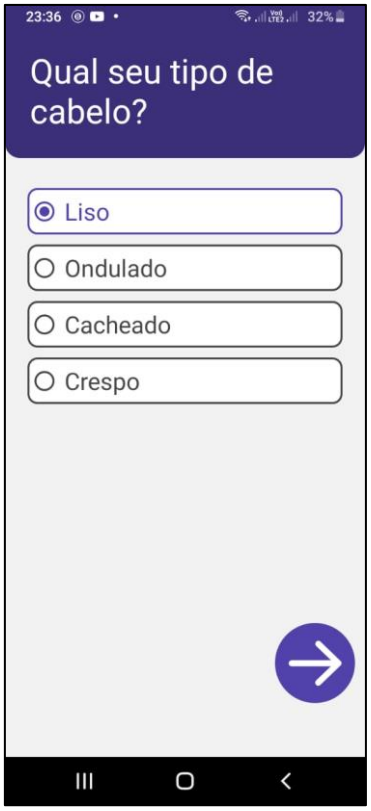
Para se registrar no aplicativo Eros, é necessário fornecer três informações para que a conta do novo usuário seja registrada no sistema: Seu nome de usuário, e-mail desejado para a criação da conta e sua senha, com esse último item sendo necessária a confirmação da senha, sendo necessária o mínimo de 6 caracteres para que possa ser registrada em nosso sistema usando o banco de dados *Firebase*. Após a criação da conta, o usuário é redirecionado para a tela seguinte, a de entrar com a sua conta que já está registrada, igualmente é mostrada na imagem 34.

Figura 34 - Tela Login

Fonte: Autoria própria, 2022.

A funcionalidade desta tela é possível somente com o usuário possuindo uma conta no banco de dados conectado ao projeto. O acesso ao aplicativo se torna possível com a utilização desta tela em questão, onde o usuário precisa informar o e-mail e a senha que foram registrados, os mesmos que foram usados na imagem ***x***. Se for o primeiro acesso do usuário ao aplicativo, será direcionado para um breve questionário que iremos reunir suas informações físicas para o desenvolvimento automático de seu tratamento de pele e cabelo, que é mostrado na imagem 35.

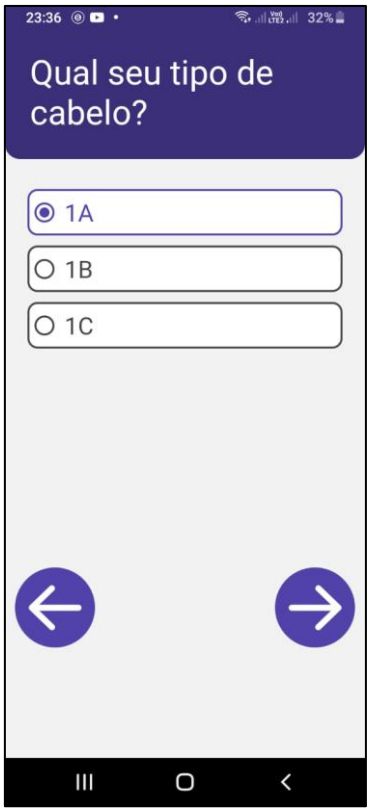
Figura 35 - Tela Tipo Cabelo



Fonte: Autoria própria, 2022.

A primeira pergunta do questionário do aplicativo Eros é sobre o tipo de cabelo da pessoa, onde para informar ao sistema, basta selecionar na lista qual é a informação correta sobre o próprio usuário, para que dessa forma, na criação do cronograma, seja informado os produtos corretos para o seu tipo de cabelo. Mesmo com isso, é necessário responder qual o formato do fio, indo desde a letra “A” até a letra “C. A coleta de dados sobre o fio de cabelo liso é mostrada na imagem 36 a seguir.

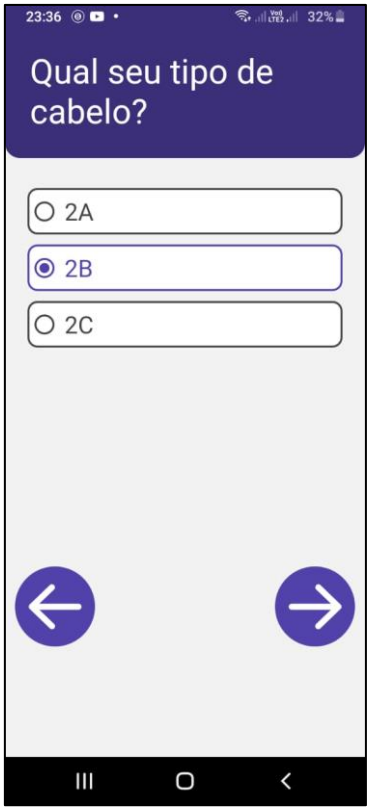
Figura 36- Tela Tipo Cabelo A



Fonte: Autoria própria, 2022.

Caso o usuário tenha informado o tipo de cabelo errado, é necessário voltar para a tela anterior, usando o ícone de seta apontada para o lado esquerdo, estando posicionada no canto inferior esquerdo. Caso o usuário tenha informado corretamente o seu tipo e formato do fio de seu cabelo, basta usar a opção de “próximo”, que é representada pela seta apontada para a direita, localizada no canto inferior direito da tela. Com a imagem 37 é possível observar as opções para os formatos do fio ondulado.

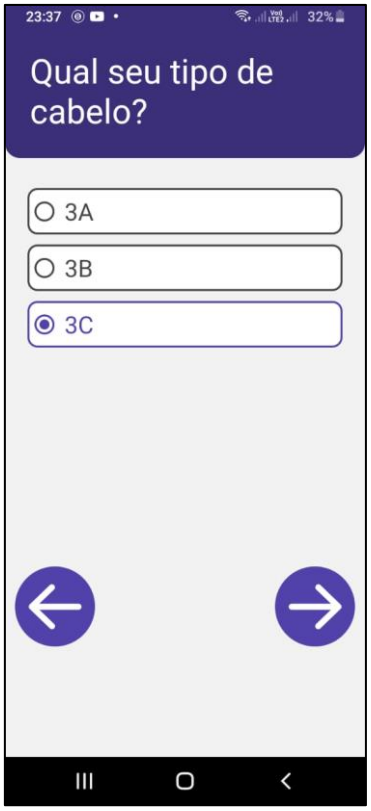
Figura 37 - Tela Tipo Cabelo B



Fonte: Autoria própria, 2022.

Caso o usuário tenha informado o tipo de cabelo errado, é necessário voltar para a tela anterior, usando o ícone de seta apontada para o lado esquerdo, estando posicionada no canto inferior esquerdo. Caso o usuário tenha informado corretamente o seu tipo e formato do fio de seu cabelo, basta usar a opção de “próximo”, que é representada pela seta apontada para a direita, localizada no canto inferior direito da tela. Com a imagem 38 é possível observar as opções para os formatos do fio cacheado.

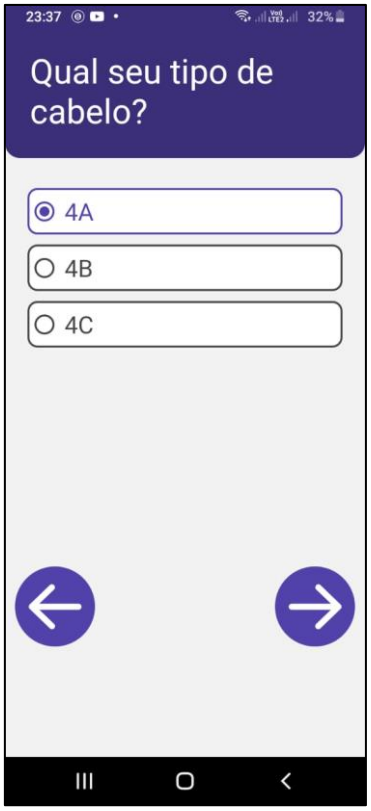
Figura 38 - Tela Tipo Cabelo C



Fonte: Autoria própria, 2022.

Caso o usuário tenha informado o tipo de cabelo errado, é necessário voltar para a tela anterior, usando o ícone de seta apontada para o lado esquerdo, estando posicionada no canto inferior esquerdo. Caso o usuário tenha informado corretamente o seu tipo e formato do fio de seu cabelo, basta usar a opção de “próximo”, que é representada pela seta apontada para a direita, localizada no canto inferior direito da tela. Com a imagem 39 é possível observar as opções para os formatos do fio crespo.

Figura 39 - Tela Tipo Cabelo D



Fonte: Autoria própria, 2022.

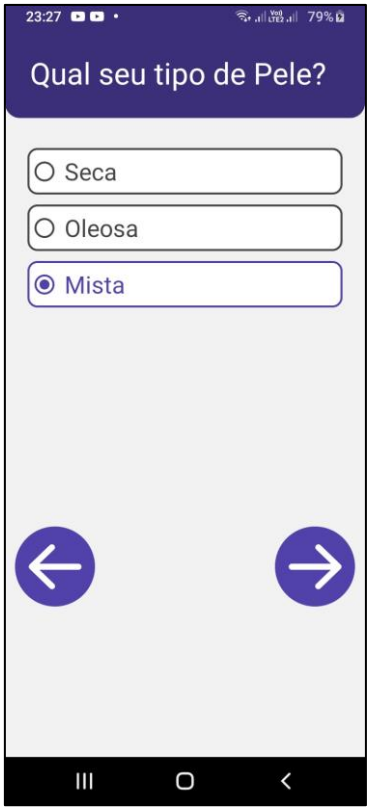
Caso o usuário tenha informado o tipo de cabelo errado, é necessário voltar para a tela anterior, usando o ícone de seta apontada para o lado esquerdo, estando posicionada no canto inferior esquerdo. Caso o usuário tenha informado corretamente o seu tipo e formato do fio de seu cabelo, basta usar a opção de “próximo”, que é representada pela seta apontada para a direita, localizada no canto inferior direito da tela, onde levará o usuário para a próxima etapa do aplicativo, igualmente é mostrada na imagem 40.

Figura 40 - Tela Tipo Cabelo

Fonte: Autoria própria, 2022.

Esta é a terceira etapa do aplicativo, onde é necessário que o usuário informe qual o estado de seu cabelo, sendo esses: Saudável, Pouco danificado e Muito danificado. É preciso reunir essa informação, visto que a intensidade do tratamento pode variar dependendo do estado que o cabelo do usuário se encontra no início do tratamento. O próximo estágio do questionário é mostrado na imagem 41.

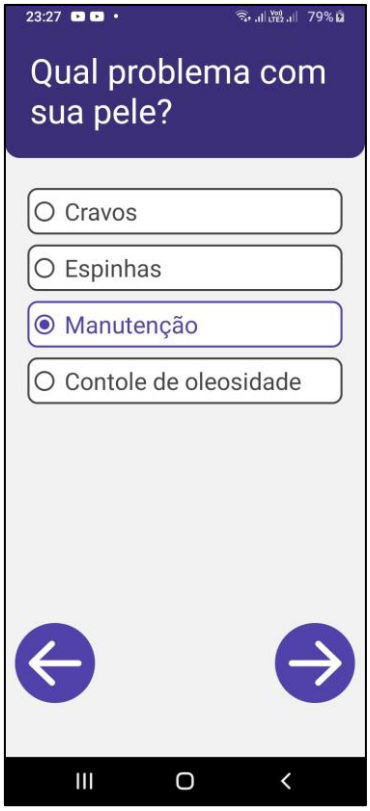
Figura 41 - Tela Tipo Pele



Fonte: Autoria própria, 2022.

Na quarta etapa do questionário, é coletada a informação do estado da pele do usuário, onde nessa tela que o sistema do aplicativo começa a separar quais são os melhores produtos para que o usuário possa utilizar e assim começar a cuidar melhor de sua pele. Na imagem 42 é coletada a última informação para que o cronograma completo do usuário seja criado.

Figura 42 - Tela Problema Pele



Fonte: Autoria própria, 2022.

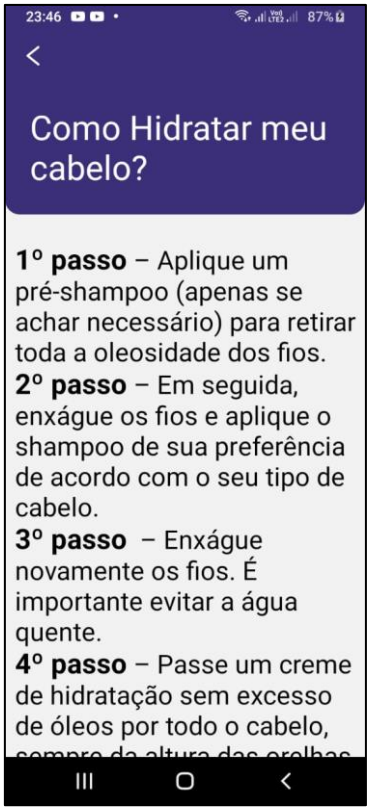
Com a última etapa do questionário, o usuário precisa informar qual o tipo específico de tratamento ele deseja fazer, assim informando quais os produtos de beleza que melhor se encaixam com as suas exigências. Após o questionário ser respondido, o usuário será direcionado para o tratamento, sendo a primeira tela o Tratamento capilar, como é mostrado na figura 43.

Figura 43 - Tela Cronograma Capilar

Fonte: Autoria própria, 2022.

Cada etapa do tratamento, seja Hidratação, Nutrição ou Reconstrução contém uma tela explicando como realizar cada uma delas. O número antes da etapa do tratamento informa qual o dia do mês que cada uma delas deverá ser feito. Na imagem 44 será explicada o passo a passo de como fazer uma hidratação no cabelo.

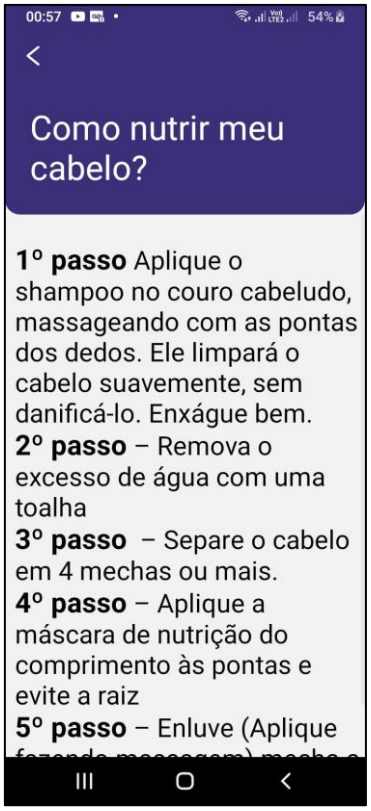
Figura 44 - Tela Como Hidratar o Cabelo



Fonte: Autoria própria, 2022.

Nessa imagem estão inseridos os dados para que possa realizar a hidratação corretamente, onde para poder voltar a tela de tratamento, basta selecionar a opção de voltar que está localizada no canto superior esquerdo da tela. Na imagem 45 será explicada o passo a passo de como fazer uma nutrição no cabelo.

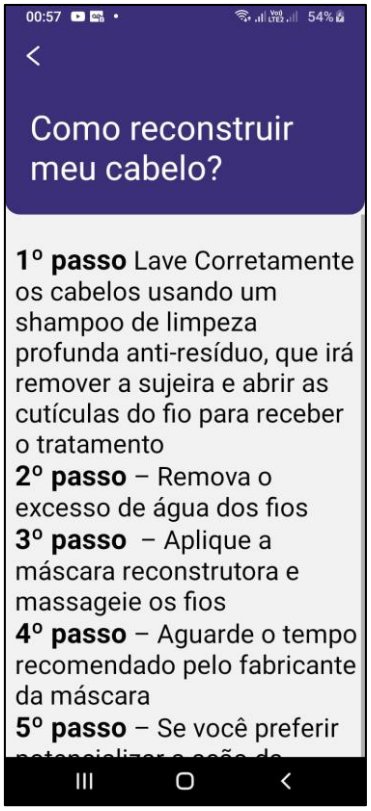
Figura 45 - Tela Tutorial Nutrição Cabelo



Fonte: Autoria própria, 2022.

Nessa imagem estão inseridos os dados para que possa realizar a nutrição corretamente, onde para poder voltar a tela de tratamento, basta selecionar a opção de voltar que está localizada no canto superior esquerdo da tela. Na imagem 46 será explicada o passo a passo de como fazer uma reconstrução no cabelo.

Figura 46 - Tela Tutorial Reconstrução Capilar



Fonte: Autoria própria, 2022.

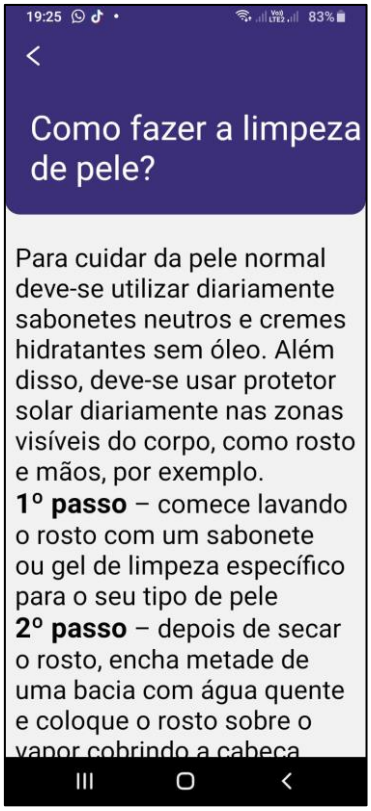
Nessa imagem estão inseridos os dados para que possa realizar a nutrição corretamente, onde para poder voltar a tela de tratamento, basta selecionar a opção de voltar que está localizada no canto superior esquerdo da tela. Voltando a tela de calendário utilizando a função de voltar, também é possível acessar o tratamento de *skincare* do usuário, clicando no botão escrito “Pele” logo acima do próprio calendário. A tela a ser exibida é mostrada conforme a imagem 47 a seguir.

Figura 47 - Tela Cronograma SkinCare

Fonte: Autoria própria, 2022.

Cada etapa do tratamento, seja Limpeza, Esfoliação, Máscara, Hidratação e Proteção, contém uma tela explicando como realizar cada uma delas. O número antes da etapa do tratamento informa qual o dia do mês que cada uma delas deverá ser feito. Na imagem 48, será explicada o passo a passo de como fazer uma Limpeza em sua pele.

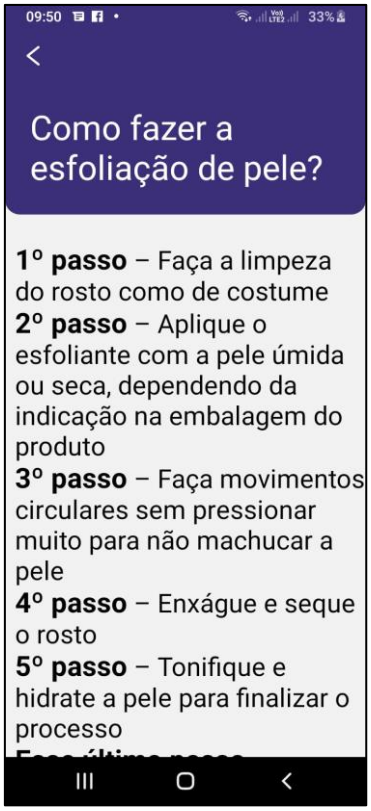
Figura 48 - Tela Tutorial Limpeza



Fonte: Autoria própria, 2022.

Nessa imagem estão inseridos os dados para que possa realizar a limpeza corretamente, onde para poder voltar a tela de tratamento, basta selecionar a opção de voltar que está localizada no canto superior esquerdo da tela. Na imagem 49 será explicada o passo a passo de como fazer uma Esfoliação na pele.

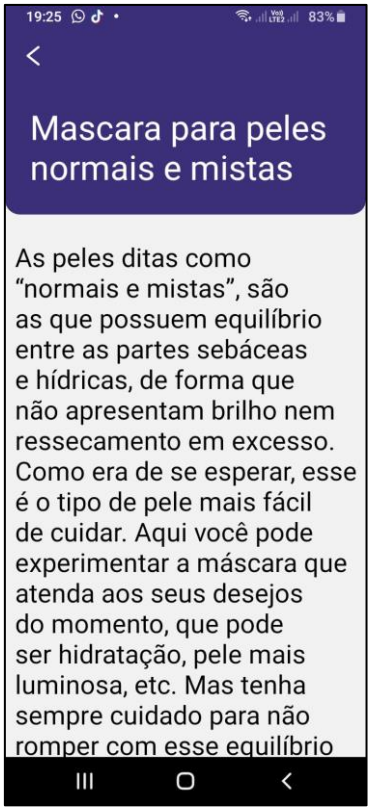
Figura 49 - Tela Tutorial Esfoliação da Pele



Fonte: Autoria própria, 2022.

Nessa imagem estão inseridos os dados para que possa realizar a limpeza corretamente, onde para poder voltar a tela de tratamento, basta selecionar a opção de voltar que está localizada no canto superior esquerdo da tela. Na imagem 50 será explicada o passo a passo de como usar a máscara corretamente.

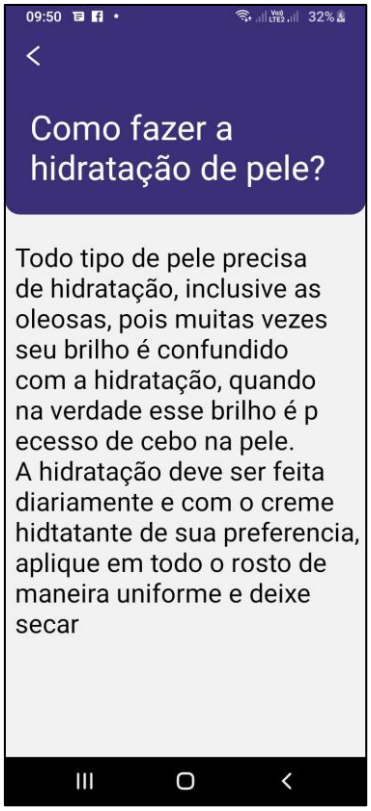
Figura 50 - Tela Uso Correto Máscara



Fonte: Autoria própria, 2022.

Nessa imagem estão inseridos os dados para que possa realizar a limpeza corretamente, onde para poder voltar a tela de tratamento, basta selecionar a opção de voltar que está localizada no canto superior esquerdo da tela. Na imagem 51 será explicada o passo a passo de como realizar a hidratação facial corretamente.

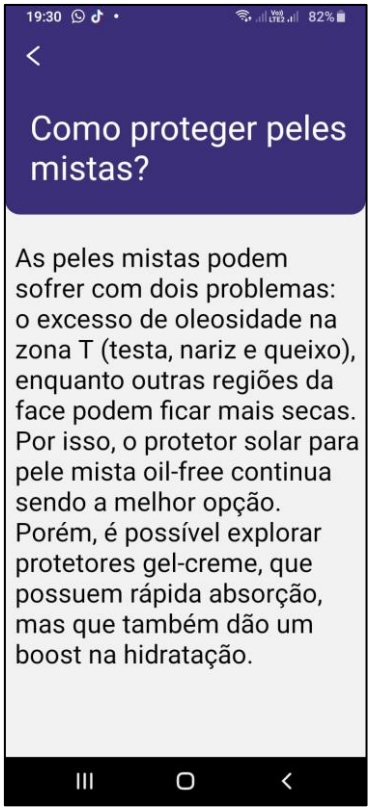
Figura 51 - Tela Hidratação Pele



Fonte: Autoria própria, 2022.

Nessa imagem estão inseridos os dados para que possa realizar a limpeza corretamente, onde para poder voltar a tela de tratamento, basta selecionar a opção de voltar que está localizada no canto superior esquerdo da tela. Na imagem 52 será explicada o passo a passo de como realizar a proteção facial corretamente.

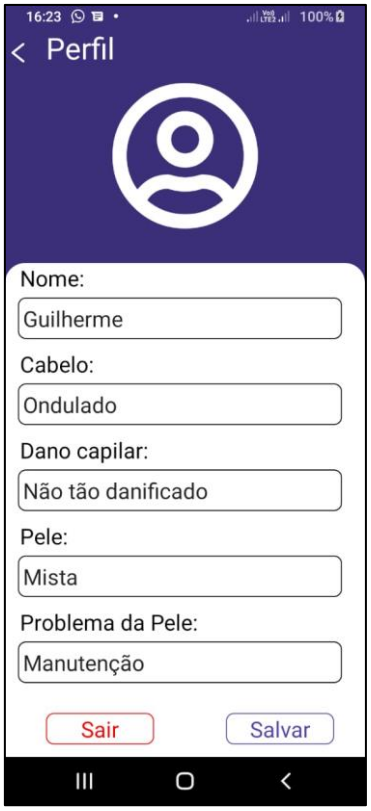
Figura 52 - Tela Tutorial Proteção Facial



Fonte: Autoria própria, 2022.

Nessa imagem estão inseridos os dados para que possa realizar a proteção corretamente, onde para poder voltar a tela de tratamento, basta selecionar a opção de voltar que está localizada no canto superior esquerdo da tela. Voltando a tela de calendário utilizando a função de voltar, também é possível acessar o perfil do usuário, clicando no botão escrito característico logo acima do próprio calendário. A tela a ser exibida é mostrada conforme a imagem 53 a seguir.

Figura 53 - Tela Perfil Usuário



Fonte: Autoria própria, 2022.

No perfil do usuário, as informações necessárias estão sendo exibidas, como seu tipo de pele, cabelo, problema de pele, dano capilar, seu nome de usuário, um botão para poder sair de sua conta onde será direcionado para a tela inicial da aplicação Eros e outro botão para salvar os novos dados.

4. **CONCLUSÃO**

Dado o presente momento que nos encontramos hoje, com as pessoas sempre buscando o padrão considerado perfeito e buscando aprovação dos demais, como de seus amigos por exemplo, acabam ficando com vergonha delas mesmas ou até mesmo tendo uma baixa autoestima. Tendo como base esse problema, foi feito um estudo de campo para analisar se uma aplicação para celulares poderia servir como auxílio e incentivo para pessoas que buscam cuidar melhor tanto da sua pele quanto de seu cabelo.

O questionário que foi aberto para o público responder, foram recolhidas as informações de que nem todo mundo está satisfeito consigo mesmo e que mesmo buscando melhorar a si mesmo, não sabiam como começar esses cuidados.

Tendo o resultado dessa pesquisa de campo, foi decidido que a criação do aplicativo Eros poderia instruir as pessoas a cuidarem melhor tanto de sua pele quanto de seu cabelo de um modo claro e objetivo. Sendo assim, a utilização de uma aplicação pode permitir a qualquer pessoa a ter uma rotina de cuidados com ela mesma, tendo o objetivo inicial da criação de um aplicativo para poder permitir com que a pessoa se sinta melhor e eleve sua autoestima atingido.

REFERÊNCIAS

ALMEIDA, Flávio. **Mean full stack JavaScript para aplicações web com MongoDB, Express, Angular e Node**. São Paulo: Casa do Código, 2015. 367 p.

ALMEIDA, Maurício Barcellos. **Uma introdução ao XML, sua utilização na Internet e alguns conceitos complementares**. 2002. Disponível em: https://www.scielo.br/j/ci/a/QQ3HX7jLBfTG3gpvPdfGK9j/abstract/?lang=pt. Acesso em: 23 ago. 2022.

BEZERRA, Eduardo. **Princípios de Análise e Projetos de Sistemas com UML**. 2007.

CHENG, Fu. **Build Mobile Apps with Ionic 2 and Firebase**: Hybrid Mobile App Development. Estados Unidos: Apress, 2017. 252 p.

COLLISON, Simon. **Desenvolvendo CSS na Web - do Iniciante ao Profissional**. São Paulo: Alta Books, 2008. 360 p.

DATE, Christopher. **Introdução a Sistemas de Bancos de Dados**. 8. Ed. São Paulo: Elsevier Brasil, 2004. 865 p.

ESCUDELARIO, Bruna; PINHO, Diego. **React Native**: desenvolvimento de aplicativos mobile com react. São Paulo: Casa do Código, 2020. 380 p.

FLANAGAN, David. **Javascript o guia definitivo**: ative suas páginas web. 6. ed. São Paulo: Bookman, 2012. 1080 p.

FLATSCHART, Fábio. **HTML 5**: embarque imediato. São Paulo: Brasport, 2011. 256 p.

JOBSTRAIBIZER, Flávia. **Criação de sites com o CSS**. São Paulo: Universo dos Livros, 2009. 144 p.

KHENNEDY, Kheronn. **Angular 11 e Firebase**: Construindo uma aplicação integrada com a plataforma do Google. Brasil: Casa do Código, 2021. 177 p.

MUNZLINGER, Elizabete. **Introdução à Tecnologia Web**. 2010. Disponível em: http://www.elizabete.com.br/site/Outros/Entradas/2011/2/20_Tecnologia_Web_files/18-CSS-Sintaxe.pdf. Acesso em: 23 ago. 2022.

PEDROSO, Robertha Pereira. **Apostila de HTML**. 2007. Disponível em: http://www.telecom.uff.br/pet/petws/downloads/apostilas/HTML.pdf. Acesso em: 23 ago. 2022.

RAMAKRISHNAN, Raghu; GEHRKE, Johannes. **Sistemas de gerenciamento de banco de dados**. 3. ed. Brasil: McGraw Hill Brasil, 2008. 905 p.

SADALAGE, Pramod; FOWLER, Martin. **NoSQL Essencial**: Um Guia Conciso para o Mundo Emergente da Persistência Poliglota. Brasil: Novatec Editora, 2019. 216 p.

SILVEIRA, Paulo; ALMEIDA, Adriano. **Lógica de Programação**: crie seus primeiros programas usando javascript e html. São Paulo: Casa do Código, 2013. 160 p.

SILVA, Maurício Samy. **Jquery**: a biblioteca do programador javascript. 3. ed. São Paulo: Novatec, 2013. 544 p.

SILVEIRA, Paulo; ALMEIDA, Adriano. **Lógica de Programação Criei seus primeiros programas usando javascript e html**. São Paulo: Casa do Código, 2013. 160 p.

SILVA, Maurício Samy. **JavaScriptGuia do Programador**. São Paulo: Novatec, 2010. 608 p.

SILVA, Maurício Samy. **Criando sites com HTML: Sites de Alta Qualidade com HTML e CSS**. São Paulo: Novatec, 2008. 431 p.

SMITH, Ben. JSON Básico: Conheça o formato de dados preferido da web. São Paulo: Novatec, 2015.

TITTEL, Ed; NOBLE, Jeff. **HTML, XHTML e CSS Para Leigos**. São Paulo: Alta Books, 2014. 412 p.

TOLEDO, Suely Alves de; MANZANO, José Augusto N. G.. **Guia de Orientação E Desenvolvimento de Sites**: html, xhtml, css e javascript/jscript. 4. ed. São Paulo: Érica, 2009. 384 p.