

# SoftHair - Aplicativo de Gestão de Salões de Beleza e Clínicas de Estética

Cesar E. Silva<sup>1</sup>, Luiz Guilherme S. Oliveira<sup>2</sup>

<sup>1</sup>Instituto Federal do Paraná (IFPR)

Caixa Postal 780 – 85860-000 – Foz do Iguaçu – PR – Brazil

**Resumo.** *A área da beleza vem crescendo a cada dia, com o aumento no número de clientes e também quantidade de produtos vendidos movimentando a economia desse setor. O Brasil atualmente é o quarto no ranking mundial em consumo de produtos de higiene pessoal, perfumaria e cosméticos (FORBES). Só por este dado é possível entender o grande potencial que o segmento tem. Mesmo diante da forte crise que o Brasil enfrentou, a média anual de crescimento desse setor nos últimos 10 anos foi 4,1%. Em 2019 uma pesquisa encomendada pela Beauty Fair (Feira Internacional de Beleza Profissional) em que foram registrados cerca de 500 mil salões de beleza em todo o Brasil. Um grande fator que tem contribuído para um número tão expressivo é o aumento significativo de franquias no setor, sejam de salões de beleza, esmalterias ou design de sobrancelhas, por exemplo. Na verdade, o número de salões de beleza no país é muito maior, já que quase 50% dos salões ainda estão na informalidade. E como estamos numa era digital, onde os processos acontecem de forma ágil, ferramentas surgem para facilitar a vida das pessoas, inclusive na área da estética e serviços relacionados a área da beleza. Com toda a informação afirmando o quão promissor é a área da beleza, surgiu a ideia de criar um aplicativo que auxiliasse os colaboradores a realizar uma tarefa vital nesse ramo de negócio, o agendamento. A aplicação SoftHair tem então como missão eliminar as agendas tradicionais de papel utilizadas em uma parte significativa dos salões de beleza e clínicas de estética, principalmente os que atuam na informalidade, tornando mais ágil e fácil o controle dos agendamentos.*

## 1. Introdução

O setor de beleza no Brasil é um dos mais promissores. São vários aspectos que colaboram para esse fato: inclusão das classes “D” e “E” decorrente do aumento de renda, inserção da mulher no mercado de trabalho, lançamento constante de novos produtos, elevação da expectativa de vida, entre outros.

A população apresenta alta demanda por produtos e serviços de qualidade e procedimentos específicos que contribuam para a elevação da autoestima e do bem estar e atendam às necessidades de higiene pessoal.

Mesmo diante de cenários econômicos de crise nos anos recentes, o Brasil está em 4º lugar no ranking mundial com relação ao mercado de beleza e cuidados pessoais, de acordo com a revista Forbes. Segundo o IBGE, o brasileiro gasta mais com beleza do que com comida. Então, este é um dos melhores ramos para se investir atualmente. (SALOMÃO, 2020).

Este trabalho tem como objetivo o desenvolvimento de um aplicativo *mobile*, que visa substituir as agendas de papel utilizadas em parte considerável dos espaços de beleza, por agendamentos de forma digital. Esta área foi escolhida visto o crescente número de microempreendedores individuais e de micro empresas que estão optando por este mercado.

## **2. Escopo**

O desenvolvimento do aplicativo *mobile* teve como base o espaço de beleza Kelly Zuchi - Terapia Capilar. Um problema de administração identificado relacionava-se com as agendas de papel, nas quais haviam muitas rasuras e letras mal escritas. Com o objetivo de eliminar estas agendas surgiu a ideia de transferi-las para um meio digital, e pela praticidade foi escolhido um aplicativo, onde administradores dos salões de beleza e clínicas de estética e seus colaboradores pudessem controlar seus agendamentos.

## **3. Metas**

Para entregar todas as funcionalidades do aplicativo, seguiu-se o planejamento do projeto, atentando-se ao cronograma. Dividindo-se as tarefas entre os membros da equipe e efetuando-se o versionamento de código para garantir agilidade e consistência na construção do *App*. Abaixo segue a lista das principais funções do *software*.

- Manter usuários para acesso ao sistema;
- Manter serviços;
- Manter clientes;
- Manter produtos;
- Controle de agendamento do cliente;

## 4. Requisitos e Regras de Negócio

### 4.1. Requisitos Funcionais

Tabela 1. Requisitos Funcionais

Código	Requisito	Descrição do requisito funcional
RF01	Cadastrar prestador	O cadastro dos prestadores no sistema deverá ser efetuado com nome, <i>e-mail</i> e senha. Esse cadastro será realizado pelo administrador.
RF02	Cadastrar serviços	Cadastrar os tipos de serviços. O sistema permitirá a manutenção de inclusão, alteração, consulta e exclusão de dados, essas funcionalidades estarão a cargo do administrador.
RF03	Cadastrar clientes	Cadastrar o cliente no aplicativo. A manutenção dos dados pelo sistema permitirá a inclusão, exclusão, consulta e alteração, a cargo do administrador e do prestador de serviço.
RF04	Cadastrar agendamento	Administrador e prestador podem registrar um agendamento.
RF05	Validação de dados do cadastro	O cadastro de usuários deverá ser validado pelo banco de dados, os campos e-mail serão únicos, evitando duplicidade.

## 4.2. Requisitos Não Funcionais

Tabela 2. Requisitos Não-Funcionais

Código	Descrição do requisito não-funcional
RNF01	O <i>framework React Native</i> será utilizado no desenvolvimento do aplicativo.
RNF02	O sistema será desenvolvido em linguagem <i>JavaScript</i>
RNF03	O sistema utilizará o banco de dados <i>PostgreSQL</i> .
RNF04	O <i>Knex.js</i> será utilizado na manipulação de <i>Querys</i> no banco de dados.
RNF05	O <i>framework Express.js</i> será utilizado no desenvolvimento das requisições HTTP na API <i>Rest</i> .
RNF06	O servidor <i>web</i> utilizado será o <i>Node.js</i> .
RNF07	O aplicativo será implementado na plataforma <i>Android</i> na versão 10.
RNF08	O servidor <i>web</i> fará uso do utilitário de monitoramento de processos <i>nodemon.js</i> .
RNF09	A ferramenta <i>Postman</i> será utilizada para realizar os testes das requisições HTTP
RNF10	Validação de <i>token JWT (JSON WEB TOKEN)</i> será implementado no <i>backend</i> com o <i>Node.js</i> , para autenticação de requisições entre as duas partes servidor <i>web</i> e o aplicativo, a cada solicitação irá incluir o <i>JWT</i> permitindo que prestador continue acessando o sistema com o <i>token</i> válido, caso o <i>token</i> tenha expirado retorna a tela de <i>login</i> .

### 4.3. Regras de Negócio

Tabela 3. Regras de Negócio

Código	Descrição regra de negócio
RN01	Para o aplicativo ter acesso ao servidor <i>web</i> e para realizar a manipulação no banco de dados , terá como protocolo de segurança a implementação de geração de <i>token</i> .
RN02	A alteração de senha de acesso ao aplicativo somente será realizada pelo prestador.
RN03	Somente será permitido iniciar um serviço com agendamento criado.
RN04	O prestador de serviço terá um único cliente por agendamento.
RN05	O prestador de serviço terá um agendamento único por horário.
RN06	O agendamento terá uma verificação prévia na data e hora antes da inserção, para evitar duplicidade.
RN07	Os campos de cadastro tal como o nome deverá ser obrigatoriamente preenchido.
RN08	O campo e-mail deverá ser único.

## 5. Metodologia

A metodologia do projeto envolveu basicamente três etapas:

(1) Desenvolvimento e testes da API *Rest*, que faz a comunicação por meio de requisições HTTP das operações lógicas do aplicativo, onde o aplicativo que é parte do *front end (interface)*, a qual o usuário interage e com o servidor *backend*; (2) levantamento das rotinas e atividades a qual o aplicativo envolveriam, para suprir as dificuldades no segmento

da beleza que poderão agregar benefícios para os usuários deste *software*. (3) Implementação do banco de dados e as suas relações entre as entidades levantadas no escopo inicial do projeto.

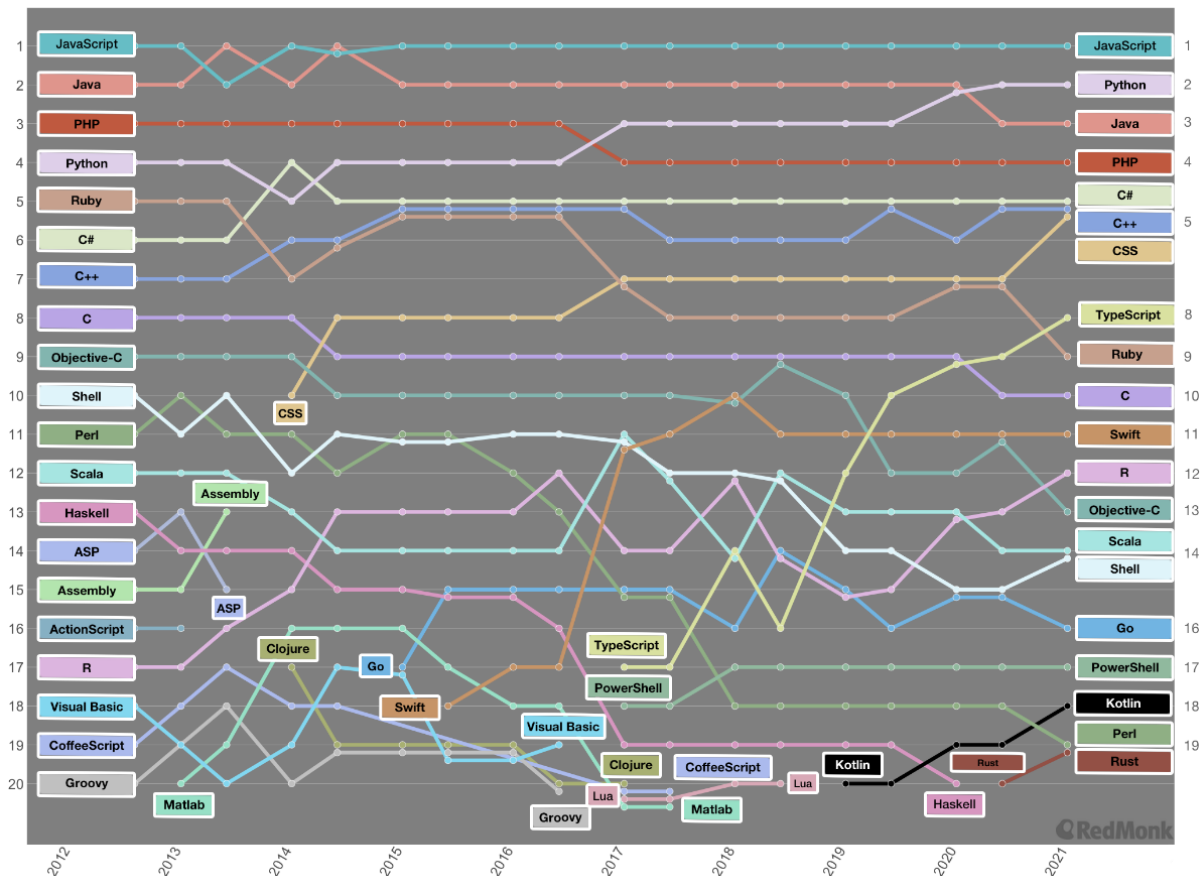
As tecnologias computacionais adotadas foram: linguagem de programação *Javascript* com uso do ambiente de execução em *Node.js* para implementação da *API Rest*, no lado *backend* da aplicação, agregando a *API* foi utilizado o método *JWT (JSON Web Token)* que permite a autenticação entre as requisições entre as duas aplicações, para os testes das chamadas às HTTP e agilidade na integração destes foi utilizada a ferramenta *Postman*. Para a implementação do aplicativo no lado *frontend* empregou-se a linguagem de programação *JavaScript* com o *framework React Native* na plataforma *Android*, Aplicou-se o *PostgreSQL* para armazenamento dos dados, servidor local *Node.js*, e como ambiente de desenvolvimento o *VSCode*. As telas do aplicativo foram implementados no *software* de prototipagem *BuilderX*. O controle de versão de código foi realizado na plataforma *GitHub*.

## **6. Tecnologias utilizadas na API Rest.**

### **6.1 Javascript**

A linguagem de programação *JavaScript* possibilita a implementação de recursos complexos em páginas web. Um *site* que não está mostrando simplesmente informações estáticas, e sim apresentando conteúdo que se atualiza em um intervalo de tempo, mapas interativos ou gráficos 3D animados, entre outros itens muito enriquecedores que tornam um *website* interessante de ser navegado. Nesta página web o *JavaScript* certamente está presente.

O *JavaScript* foi criado como um complemento para o navegador da *Netscape*. Por certo período foi rotulado como uma linguagem ruim, bagunçada e lenta. Com uma evolução significativa ela se tornou organizada e rápida. Segundo a *RedMonk*, uma empresa de analistas do setor focada em desenvolvedores de software, a linguagem *JavaScript* ocupa o primeiro lugar entre as 20 linguagens mais populares no mundo, à frente das linguagens *Python* e *Java*, este estudo foi realizado com respostas de mais de 17 mil desenvolvedores em 159 países, esta foi aferida entre as datas de setembro de 2012 a janeiro de 2021 (Figura 1). Em outra pesquisa da empresa de análise de mercado, a *SlashData*. De acordo com o estudo, mais de cinco milhões de desenvolvedores se juntaram à comunidade *JavaScript* desde 2017. “Mesmo em setores onde *JavaScript* é menos popular, como em ciência de dados e realidade aumentada e virtual, mais de um quinto dos desenvolvedores a usa em seus projetos”, diz a *SlashData*.



**Figura 1. Gráfico de ranking das 20 linguagens de programação mais populares.**  
**Fonte:www.redmonk.com**

O *JavaScript* não se restringe apenas aos navegadores. Essa linguagem possibilita a criação de aplicações nativas. O *React Native* o qual será explicado no capítulo 7.1, e o *NativeScript* são ferramentas importantes para gerar aplicativos para dispositivos móveis. Essas bibliotecas permitem implementar telas com XML, e estilizar com a linguagem CSS, sendo convertidas para telas nativas de cada plataforma, como *Android* e *iOS*. As ações são escritas com *JavaScript*.

Uma aplicação nativa ao invés de uma aplicação com um navegador, oferece melhor performance do que uma aplicação híbrida. As aplicações híbridas dependem do navegador padrão do sistema, com a possibilidade de criar um código que o navegador daquele dispositivo não suporte. Em uma aplicação nativa, por ser independente de navegadores, não há preocupação se haverá suporte para as funcionalidades.

## 6.2. Node.js

O *Node.js* se caracteriza como um ambiente de execução *JavaScript*. Com ele, é possível criar aplicações sem depender do *browser* para isso. Com alta capacidade de escalabilidade, boa flexibilidade, e produtividade. O *Node.js* é uma plataforma muito versátil e que pode ser usada em inúmeros cenários. Seu gerenciador de pacotes o *npm* (*Node Package Manager*) é classificado como o maior repositório de *softwares* disponível, sendo que um desses pacotes mais famoso é um *framework* voltado a desenvolvimento web, chamado *Express.js* entre outros *packages* que auxiliam no desenvolvimento como o *nodemon* e *Knex*, destes citados foram usados no desenvolvimento do projeto.

## 6.3. Estrutura da API Rest

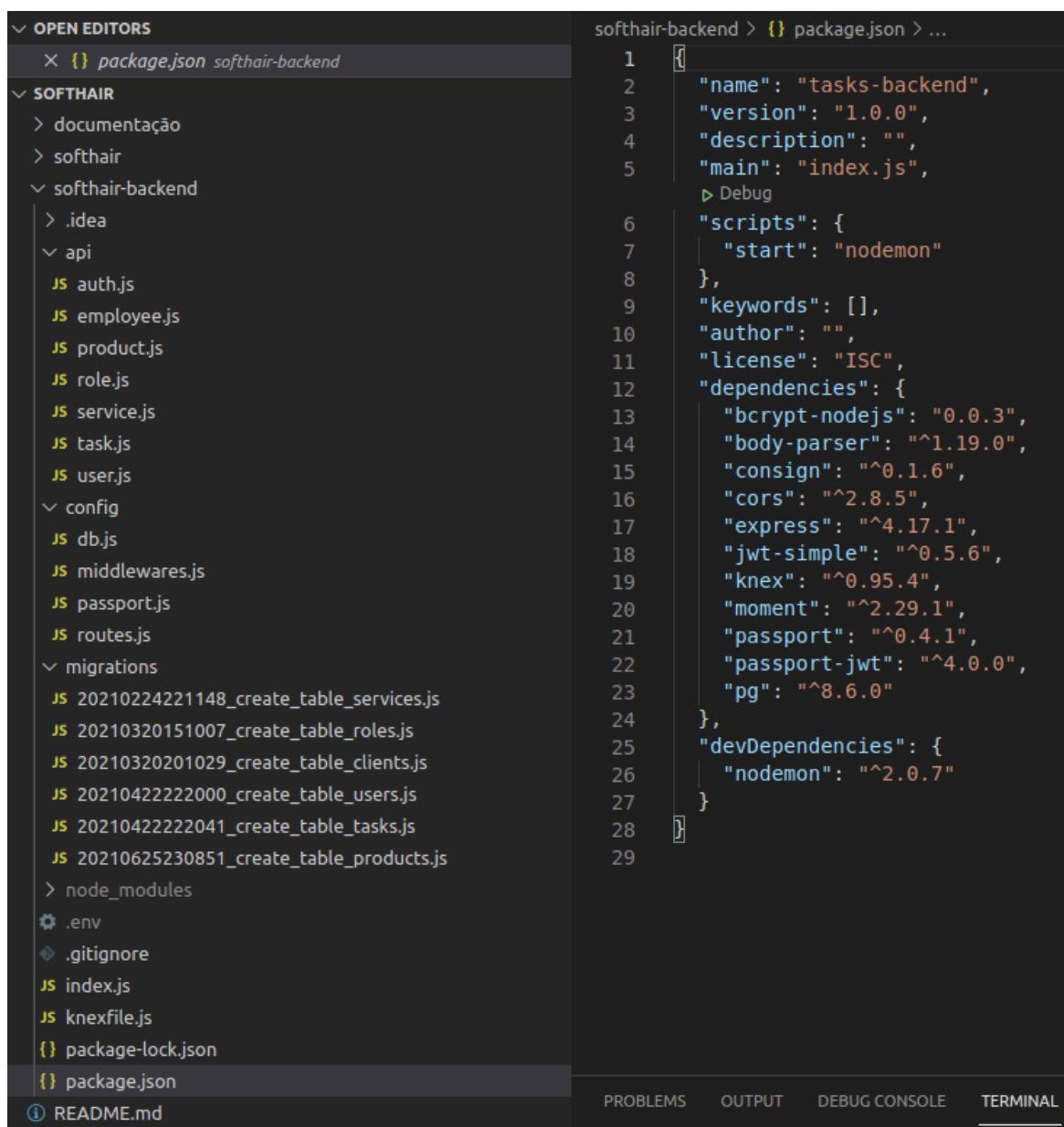


Figura 2. Tela da estrutura de diretórios do projeto da API.



A Figura 2 apresenta como está organizado o projeto no lado servidor conhecido também como *backend*, detalha-se algumas subpastas e arquivos de *scripts*, entre estes o arquivo em destaque é o *package.json* que apresenta as bibliotecas e suas versões utilizadas no desenvolvimento da API, o projeto está disponível através do link, “<https://github.com/luizsett7/softhair>”.

#### 6.4. Testes de comunicação da API pela ferramenta *Postman*

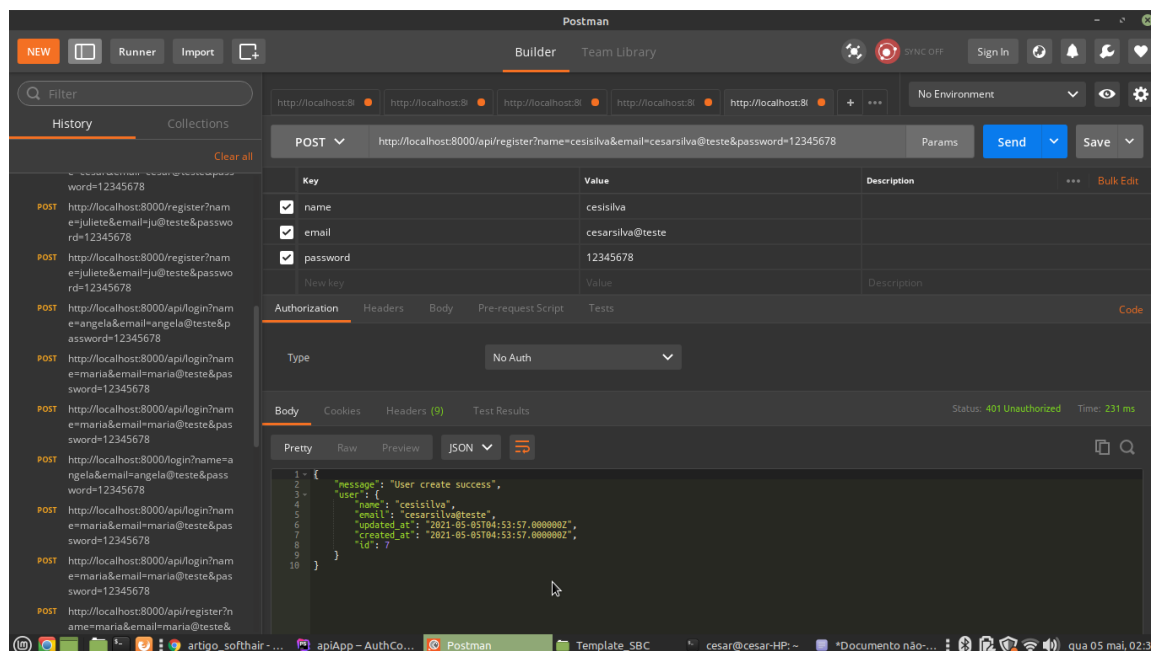


Figura 3. Tela do *Postman* com os resultados das chamadas HTTP.

Com a utilização da ferramenta *Postman*, pode ser realizada os testes das requisições da API, mesmo antes de implementar a interação do aplicativo *front end* com o lado *backend*, passando parâmetros nos campos solicitados e no seu retorno imprime os dados que foram definidos na modelagem, assim as chamadas aos métodos da API podem ser corrigidas e testadas sem necessitar passar pelo aplicativo.

#### 6.5. *JWT (JSON Web Token)*

É um método RCT 7519 padrão da indústria para realizar autenticação entre duas partes por meio de um *token* assinado que autentica uma requisição HTTP. Esse *token* é um código que armazena objetos *JSON* com os dados que permitem a autenticação das requisições.

Através de uma requisição HTTP ao *endpoint* de autenticação da API. Nela o usuário envia, no corpo da requisição dados como *e-mail* e senha.

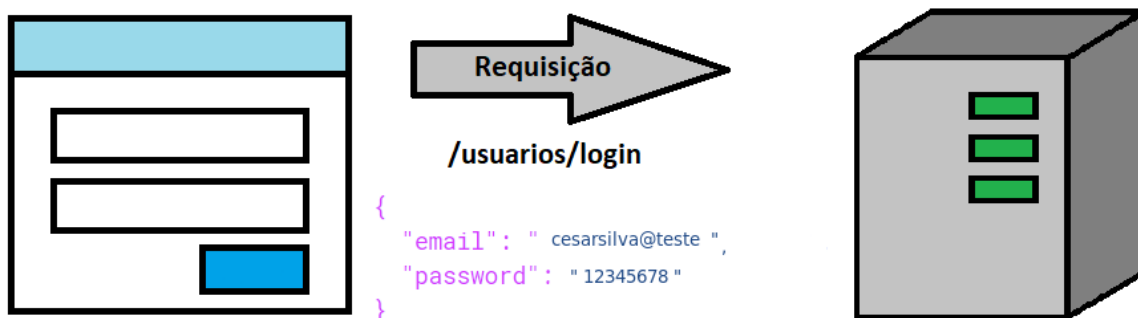


Figura 4. Usuário enviando requisição com dados de autenticação.

Uma vez que os dados enviados pelo usuário tenham sido autenticados no servidor, este criará um *token JWT* assinado com um segredo interno da API e enviará este *token* de volta ao usuário.



Figura 5. Servidor envia o *token* assinado para o usuário.

Provido com o *token* autenticado, o usuário possui acesso aos *endpoints* da aplicação que antes eram restritos.

## 6.6. Express.js

O *Express* é um *framework* para aplicativo da *web* do *Node.js*, leve e flexível que fornece um conjunto robusto de recursos para aplicativos *web* e *mobile*.

Com uma miríade de métodos utilitários HTTP e *middleware* a seu dispor, criar uma API robusta é rápido e fácil.

## 6.7. Knex.js

É um módulo do *Node.js* para fazer a criação de banco de dados e manipulação dos dados, porém com ele podemos manter um padrão de *Querys* no qual, se mudar o tipo do banco de dados ainda assim a aplicação continuará funcionando.

## 6.8. Nodemon.js

É uma ferramenta que ajuda a desenvolver aplicativos baseados em *Node.js*, reiniciando automaticamente o aplicativo quando mudanças de arquivo no diretório são detectadas.

## 6.9. Código-fonte

O código-fonte escrito no decorrer da implementação deste projeto está disponível no *GitHub*, um sistema de controle de versões, utilizado no desenvolvimento de software. Endereço do projeto: <https://github.com/luizsett7/softhair>.

# 7. Especificações do aplicativo *mobile*

## 7.1. React Native

O *React Native* é um *framework* baseado no *React*, desenvolvido pelo *Facebook*, que permite o desenvolvimento de aplicações *mobile*, para várias plataformas como por exemplo *Android* e *iOS*, utilizando apenas *Javascript*.

Desenvolver *apps* para dispositivos móveis era complexo, pois além de ter que aprender as linguagens *Objective-C* (*iOS*) e *Java* (*Android*), o desenvolvedor não aproveitava o código de uma plataforma para outra, fazendo com que as empresas tivessem equipes distintas de desenvolvimento para cada sistema operacional, tornando o projeto lento e caro. Com a utilização do *React Native*, o código pode ser reaproveitado em até 100% entre as plataformas, podendo fazer com que o custo e a duração do projeto caíam significativamente.

## 7.2. Prototipação

Para a etapa de construção das telas utilizou-se da ferramenta *BuilderX* que possibilita criar o *design* das interfaces e também conta com o recurso de gerar componentes que podem ser integrados facilmente ao *app*. Um exemplo de componente pode ser visualizado na Figura 6.

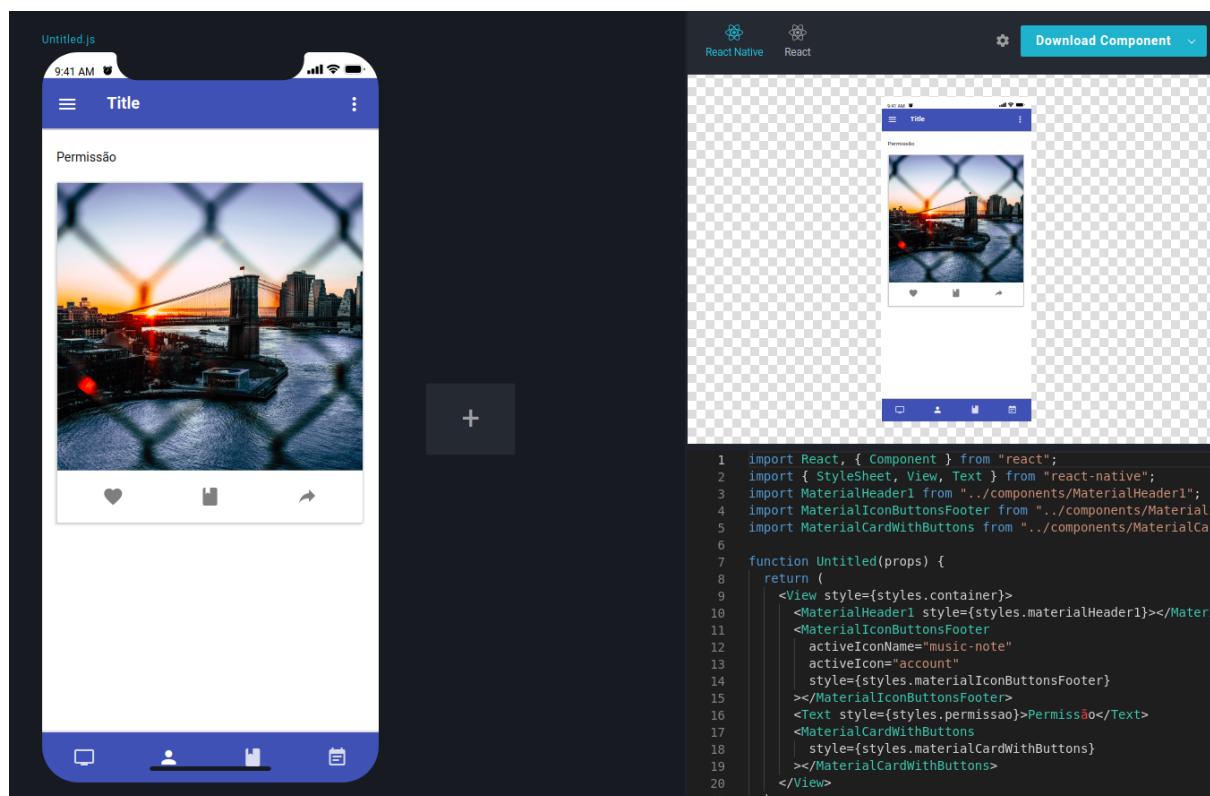


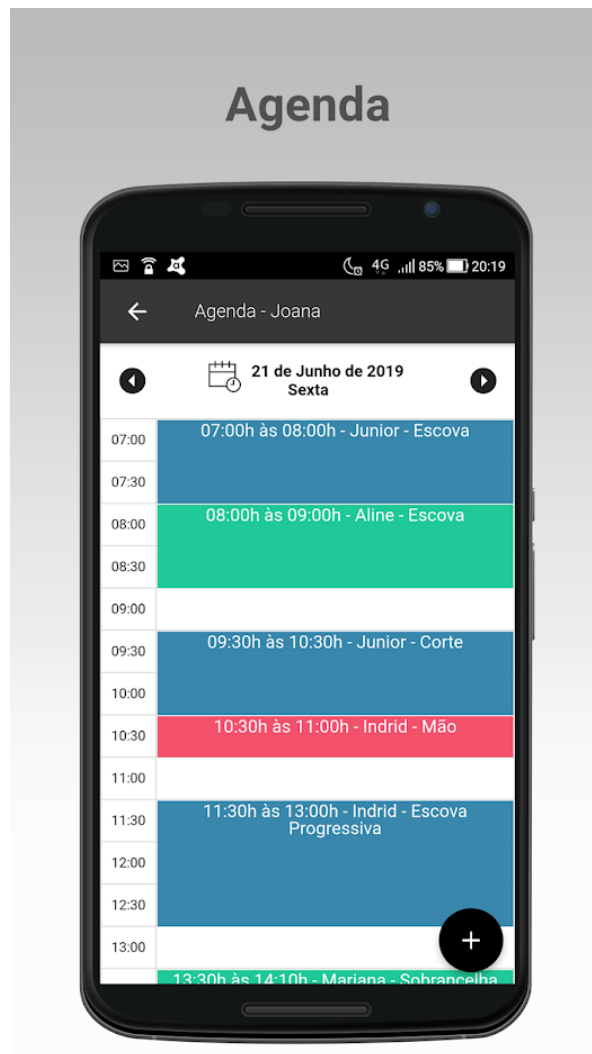
Figura 6. Tela de criação de interfaces do BuilderX.

## 8. Trabalhos Relacionados

### 8.1 *Salon Soft* - Agenda e Sistema para Salão de Beleza

O *App Salon Soft* é um aplicativo que permite o cadastro de clientes, serviços e profissionais, tanto pelo computador quanto pelo celular, de forma sincronizada. A agenda permite ainda o acesso dos profissionais pelo celular, com total integração da equipe, facilitando o cadastro de novos clientes e agendamentos de forma sincronizada entre o celular e o computador, além da possibilidade de enviar lembretes de agendamentos para os clientes.

O aplicativo *SoftHair* e o *Salon Soft* tem o agendamento como elemento central, o *Salon Soft* tem como destaque sua integração da plataforma *web* com a *mobile*.

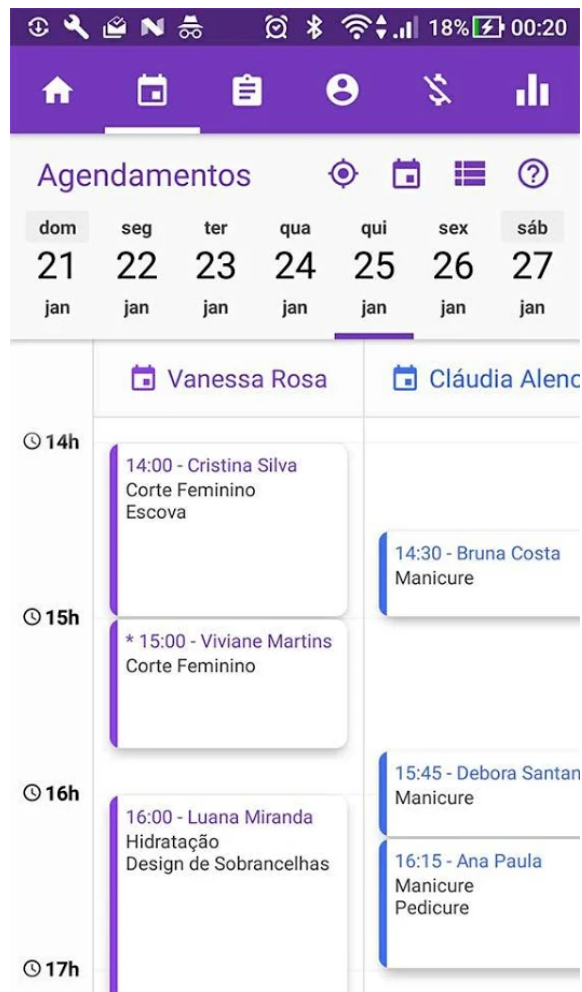


**Figura 7. Tela de agenda do *Salon Soft*.**

## **8.2 *Pocket Agenda* - Gerenciamento de Salão de Beleza**

O *Pocket Agenda* é um aplicativo gratuito de gestão para salões de beleza. Utilizado para cadastrar clientes, controlar agendamentos, criar comandas, calcular comissões, cadastrar despesas e visualizar relatórios.

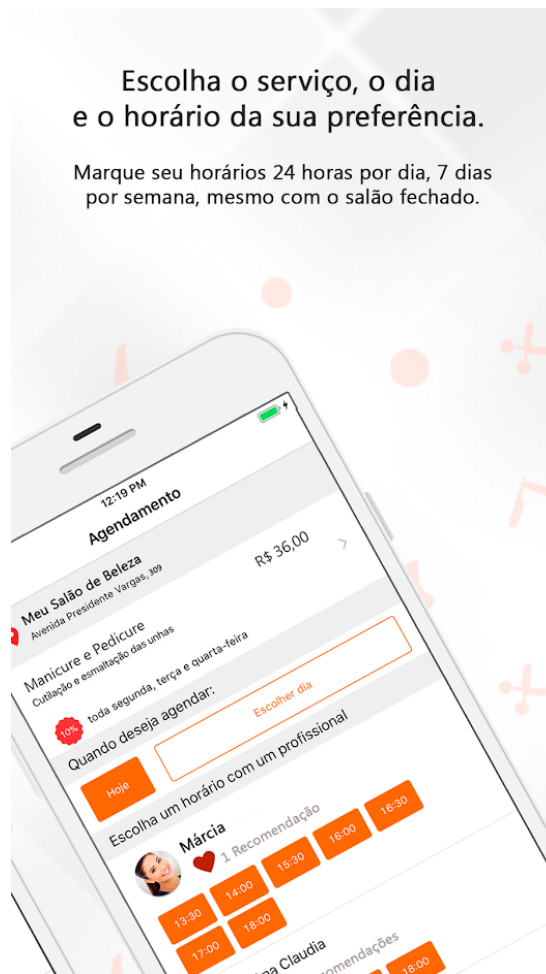
O *Pocket Agenda* e o *SoftHair* tem como núcleo o agendamento, o *Pocket Agenda* é muito completo trazendo como diferencial entre suas funcionalidades a criação de comandas e cálculo de comissões.



**Figura 8. Tela de agenda do *Pocket Agenda*.**

### 8.3 Trinks.com

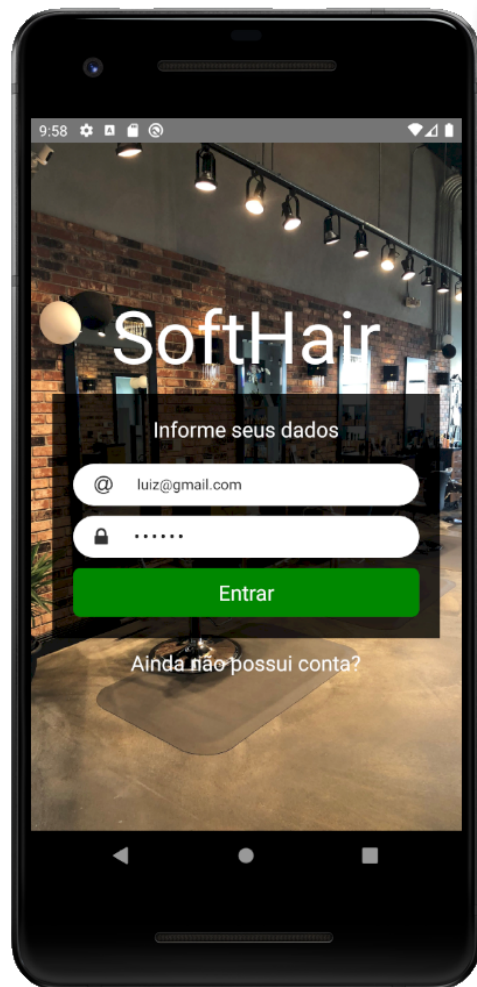
O *Trinks.com* foi desenvolvido para marcar agendamento *on-line* em salões de beleza e clínicas de estética. O estabelecimento receberá o agendamento no mesmo instante. Há possibilidade de escolher entre diversos espaços de beleza. O *SoftHair* tem como ambição possuir algumas funcionalidades do *Trinks.com*, permitindo que os clientes realizem os seus próprios agendamentos, até mesmo com o salão fechado.



**Figura 9. Tela de agendamento do *Trinks.com*.**

## **9. Resultados Obtidos**

Apresenta-se nesta seção as telas de interação do usuário com o sistema. Na Figura 10 observa-se a tela de login com os campos obrigatórios *e-mail* e senha e também com a opção de criar uma nova conta.



**Figura 10. Tela de *Login*.**

Conforme mencionado anteriormente no capítulo de API, o *login* visto na Figura 10 utiliza-se do *JWT* para autenticar o usuário e permitir que ele realize as funções do sistema.





**Figura 11. Tela de criação de conta.**

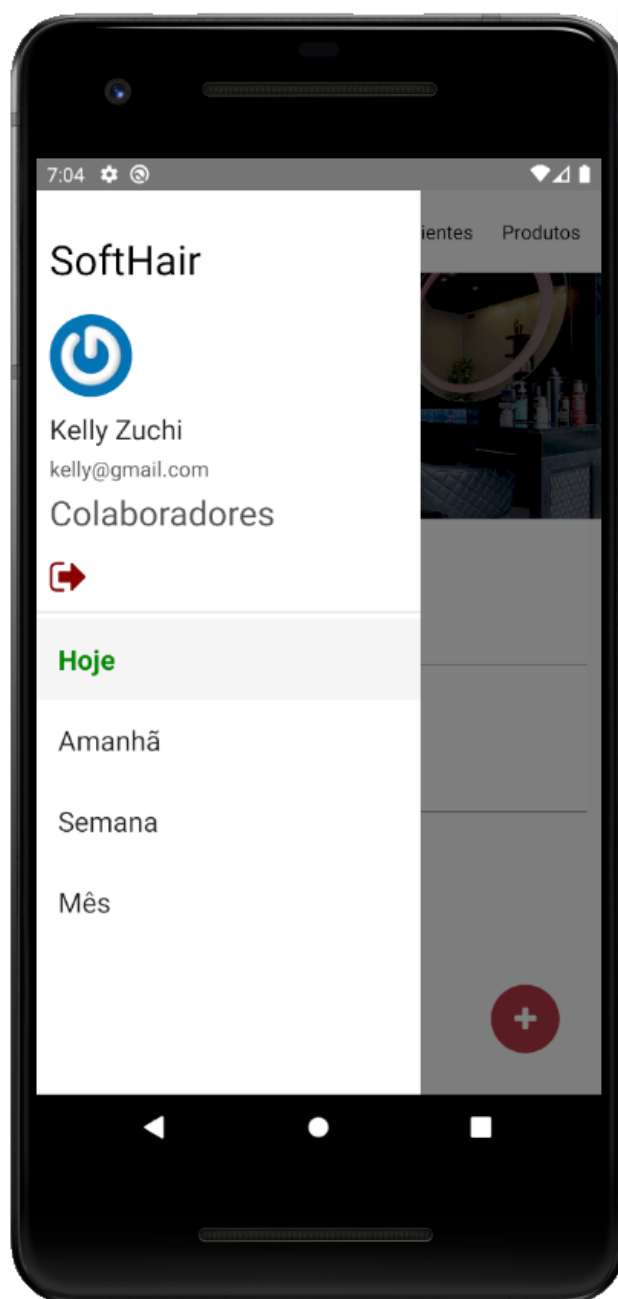
Na Figura 11 observa-se os campos necessários para criação de uma nova conta. Segundo a regra de negócio 11 o campo de e-mail deverá ser único sendo definida no banco de dados pelo comando *table.string('email').notNull().unique()*. Com o objetivo de evitar que o campo nome ficasse em branco utilizou-se uma validação (regra de negócio 10) conforme trecho de código a seguir.

```
if(!login.nome || !login.nome.trim()) { // Verifica se existe um nome

    Alert.alert('Dados inválidos', 'Nome não informado!') // Caso não exista exibe um alerta

    return

}
```



**Figura 12. Tela com menu lateral.**

A figura 12 mostra o menu no qual é possível navegar para verificar os agendamentos por períodos determinados e também para visualizar a lista de colaboradores.

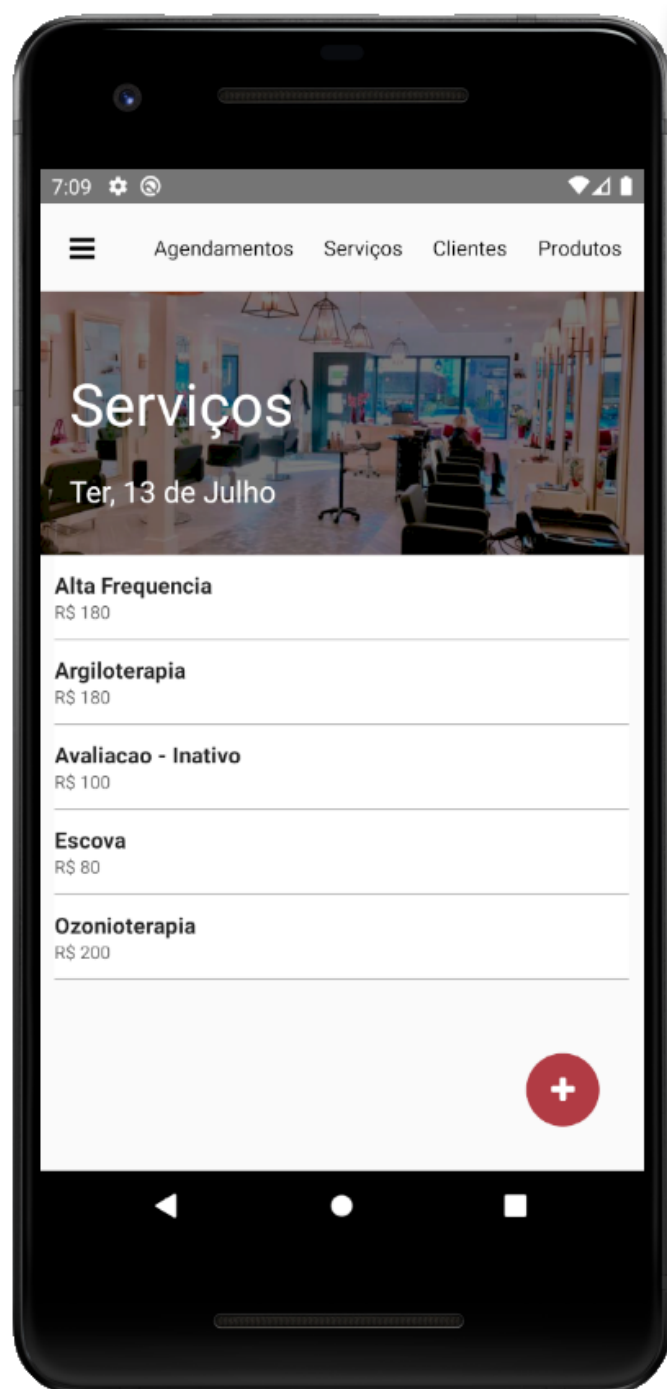


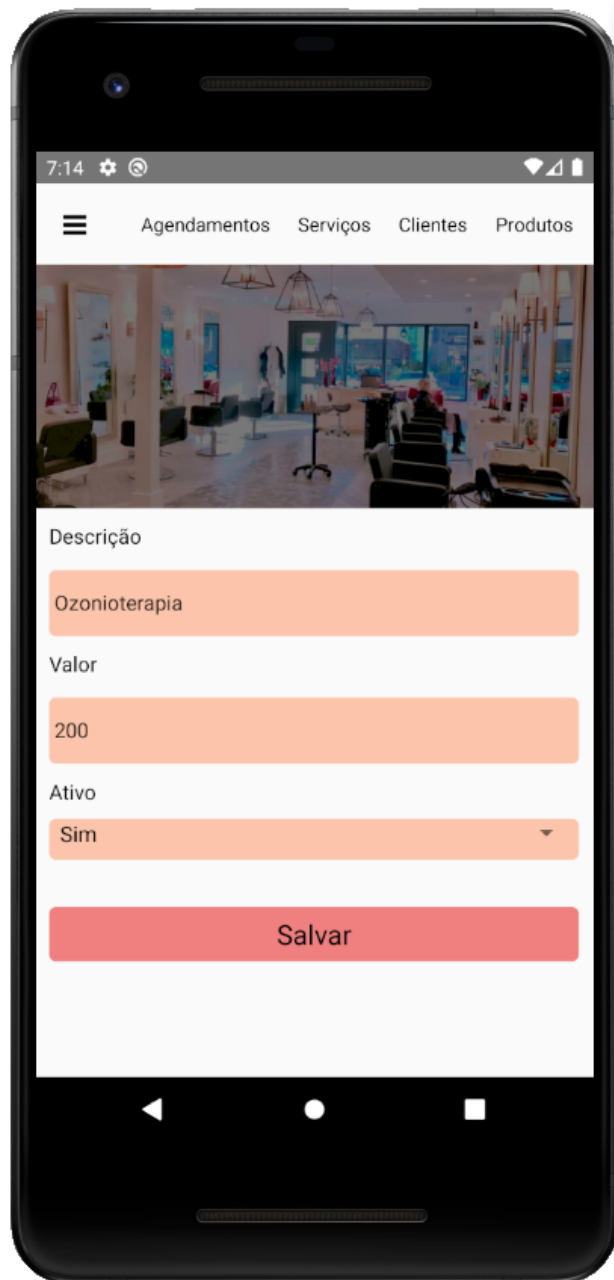
Figura 13. Tela com lista de serviços.

Mostra-se na figura 13 uma lista contendo os serviços com descrição, valor e se está inativo.



**Figura 14. Tela de cadastro de serviços.**

Na figura 14 está demonstrado a tela de inserção de serviços com campo descrição e valor.



**Figura 15. Tela de edição de serviços.**

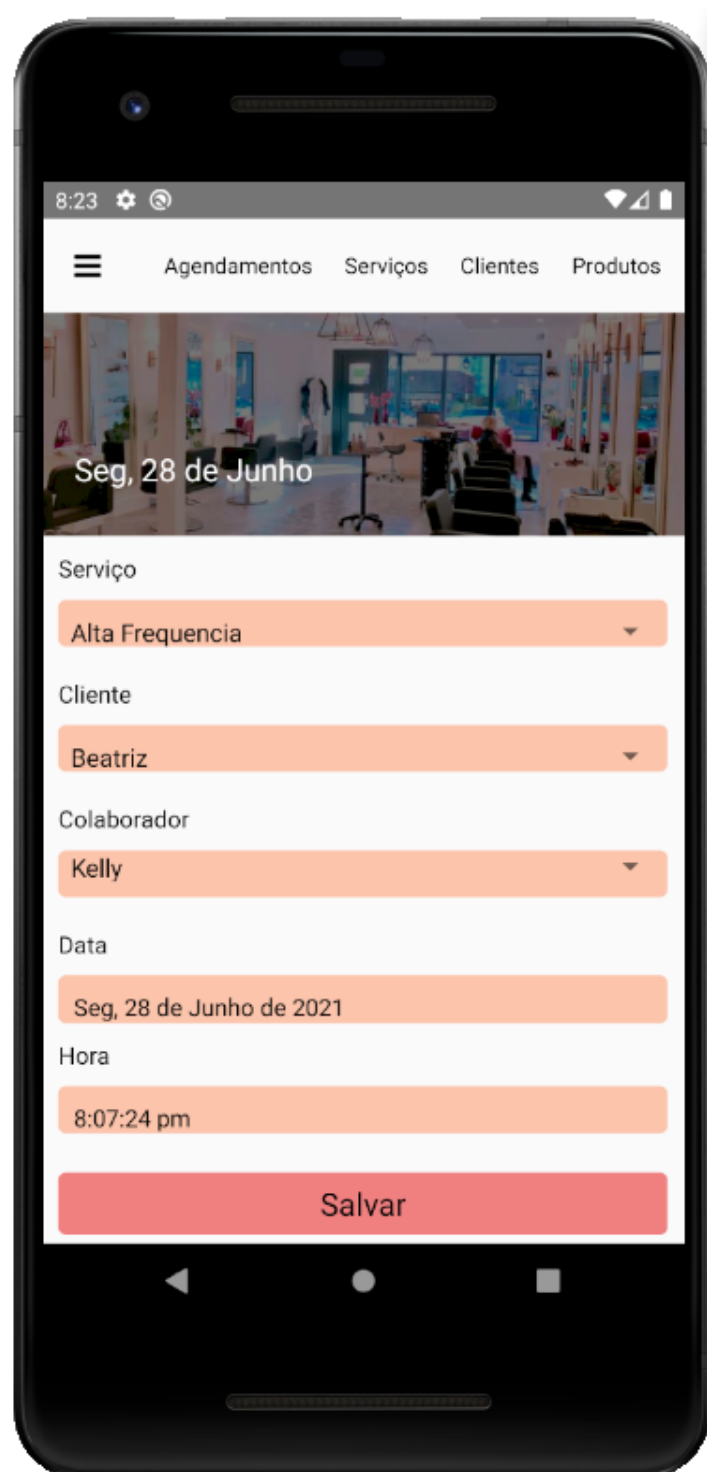
A figura 15 mostra a tela de edição do serviço com os campos descrição, valor e se está ativo.

A Figura 16 demonstra a tela de lista de agendamentos que concede as seguintes funcionalidades de gerenciamento, inserção, visualização, atualização e remoção.

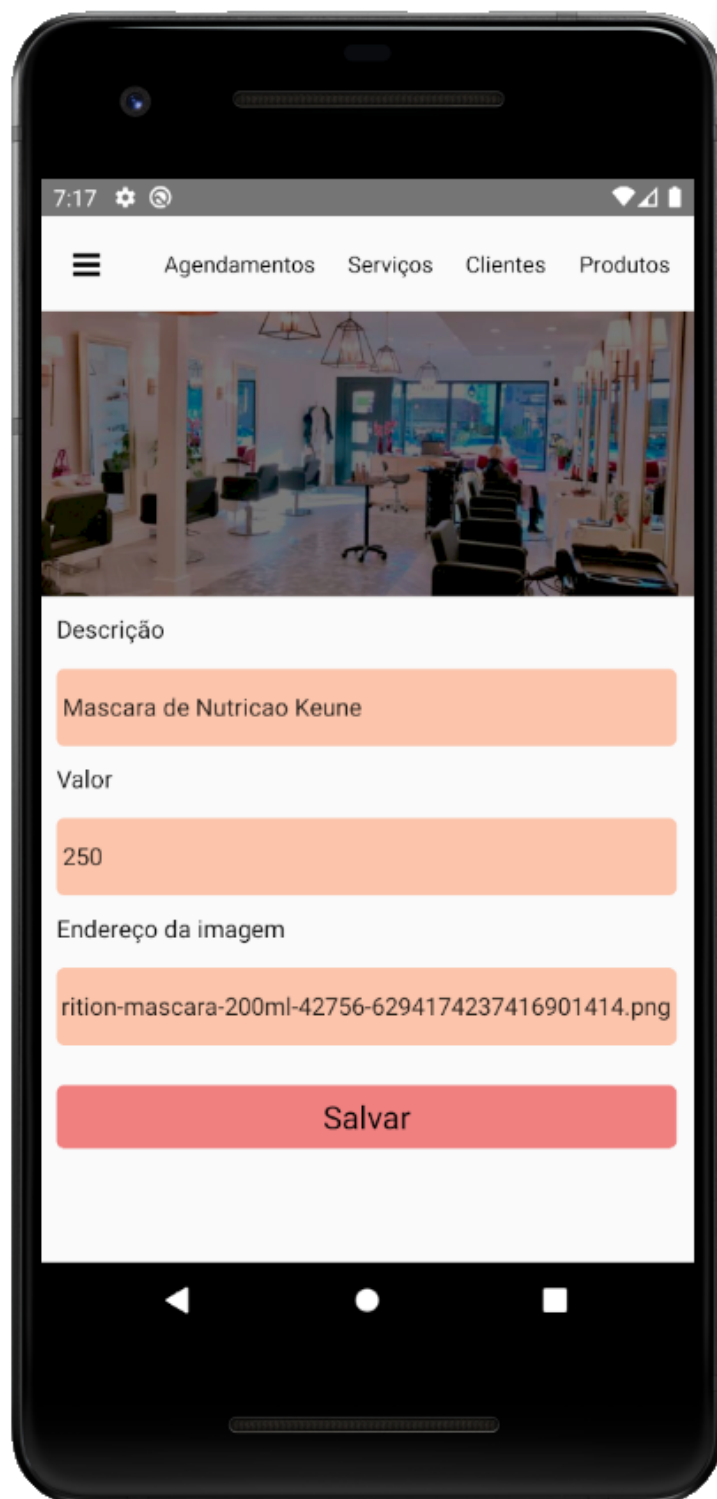


**Figura 16. Tela de Lista de Agendamentos.**

Na Figura 17 observa-se a tela de inserção de agendamento. Este cadastro possui uma validação na data e hora para evitar agendamentos no mesmo horário, permitindo somente agendamentos no intervalo de trinta minutos, conforme regra de negócio 9.



**Figura 17. Tela de Cadastro de Agendamento.**



**Figura 18. Tela de Cadastro de Produtos.**

Na Figura 18 demonstra-se a tela de cadastro de produtos. O campo endereço da imagem necessita ser preenchido com um *link* com final de extensão de imagens como .jpg ou .png. A imagem do produto cadastrado será mostrado na figura a seguir.



Na Figura 19 revela-se a tela de catálogo de produtos.

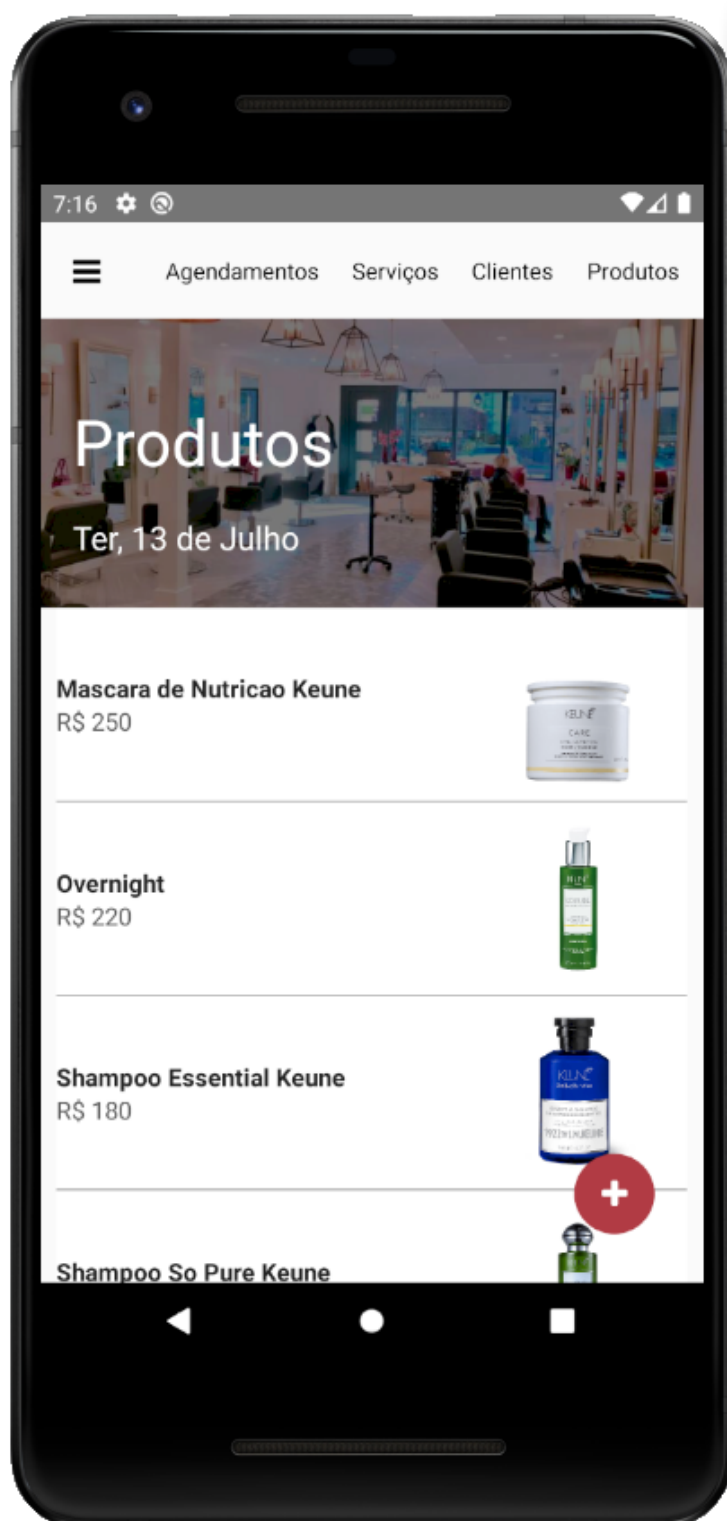


Figura 19. Tela de Catálogo de Produtos.

## 10. CONCLUSÃO

Ao final do desenvolvimento do aplicativo *SoftHair*, conclui-se primeiro que a construção desta ferramenta foi de importância ímpar para o enriquecimento do conhecimento em tecnologias que antes eram desconhecidas tais como *React Native* e *Node.js*. Em seguida notou-se que o *App* cumpriu todas as funcionalidades para o qual foi projetado, que são: cadastro de usuário, cadastro de serviços, cadastro de clientes, cadastro de produtos e o principal cadastro de agendamentos. Ao longo do desenvolvimento da aplicação levando-se em consideração várias pesquisas de aplicativos relacionados, observou-se que o *SoftHair* tem muito a crescer, tanto no âmbito dos proprietários de salões de beleza e clínicas de estética e seus colaboradores, com a adição das principais funcionalidades de gestão de espaços de beleza, como controle de estoque, administração financeira, relatórios, cálculo de comissões. E também no aspecto do cliente no qual deverá ser agregada a função de agendamento *on-line*, otimizando o tempo tanto do profissional quanto do cliente.

Um diferencial para a versão 2022 do *app*, no qual os outros aplicativos comparados não tem, é o aluguel de espaço de trabalho, no qual uma prestadora de serviço poderá alugar uma cadeira no salão de beleza, por exemplo. Ao alugar o espaço será fornecido o aplicativo *SoftHair* para possibilitar a verificação de comissão pelos serviços prestados. Sendo que uma porcentagem estabelecida é retida por quem concede a estação de trabalho.

Durante a fase final de testes foi detectado que o aplicativo possui algumas fragilidades, como a possibilidade de criar agendamento no passado, não permite o agendamento para dois colaboradores no mesmo horário e está limitado a agendamentos com intervalo de trinta minutos sendo que há serviços que duram mais do que esse tempo. Essas falhas serão sanadas no futuro juntamente com a incorporação de novas funcionalidades.

## 11. REFERÊNCIAS

- ANDRADE, A. P. **O que é o Express.js?**, 2021. Disponível em:  
<https://www.treinaweb.com.br/blog/o-que-e-o-express-js>. Acesso em: 03 abr. 2021.
- BACEGA, V. **Banco de Dados com Knex**, 2021. Disponível em:  
<https://pt.linkedin.com/pulse/banco-de-dados-com-knex-victor-bacega>. Acesso em: 12 jun. 2021.
- BECKER, L. **O que é React Native?**, 2021. Disponível em:  
<https://www.organicadigital.com/blog/o-que-e-react-native>. Acesso em: 29 mar. 2021.
- BOSCARINO, A. S. **Como o JWT funciona**, 2019. Disponível em:  
<https://www.devmedia.com.br/como-o-jwt-funciona/40265>. Acesso em: 08 mai. 2021.
- BUENO, D. C. **O que é Postman?**, 2019. Disponível em:  
<https://blog.bsource.com.br/tecnologia/2019/10/29/o-que-e-postman>. Acesso em: 06 abr. 2021.
- FLANAGAN, David. **JavaScript: o guia definitivo**. Tradução J.E.N. Tortello. 6. ed. Porto Alegre: Bookman, 2013.
- FITZPATRICK, K. **RedMonk Top 20 Languages Over Time**, Portland, 02 mar. 2021. Disponível em:  
<https://redmonk.com/kfitzpatrick/2021/03/02/redmonk-top-20-languages-over-time-january-2021>. Acesso em: 28 abr. 2021.
- HANASHIRO, A. **O que se pode fazer com JavaScript hoje em dia?** São Paulo, 2018. Disponível em:  
<https://www.treinaweb.com.br/blog/o-que-se-pode-fazer-com-javascript-hoje-em-dia>. Acesso em: 26 abr. 2021.
- HASSIJA, V. **What is BuilderX?**, 2019. Disponível em:  
<https://support.builderx.io/hc/en-us/articles/360037876694-What-is-BuilderX>. Acesso em: 23 jun. 2021.
- LIMA, A. **O Que é GitHub e Para Que é Usado?**, 2021. Disponível em:  
<https://www.hostinger.com.br/tutoriais/o-que-github>. Acesso em 12 mai. 2021.
- MARINHO, T. **Execução em tempo de desenvolvimento dos projetos em Node.js**, 2020. Disponível em:  
<https://blog.rocketseat.com.br/ferramentas-de-compilacao-execucao-em-tempo-de-desenvolvimento-dos-projetos-em-node-js>. Acesso em: 12 jun. 2021.

RIGUES, R. **JavaScript se consolida como a linguagem de programação mais popular**, São Paulo, 2020. Disponível em:  
<https://olhardigital.com.br/2020/10/22/noticias/javascript-se-consolida-como-a-linguagem-de-programacao-mais-popular>. Acesso em: 27 abr. 2021.

ROVEDA, U. **JavaScript: o que é, para que serve e como funciona o JS?**, 2021.  
Disponível em: <https://kenzie.com.br/blog/javascript> Acesso em: 27 abr. 2021.

SALOMÃO, R. M. **O promissor Mercado de Salões de Beleza do Brasil**, Belo Horizonte, 2020. Disponível em: <https://buyco.com.br/blog/mercado/mercado-de-saloes-de-beleza>.  
Acesso em: 8 mar. 2021.

SOUZA, I. **Saiba o que é Node.js, como ele funciona e como usá-lo no seu site**, 2020.  
Disponível em: <https://rockcontent.com/br/blog/node-js>. Acesso em: 02 abr. 2021.

WEBER, M. **O Brasil é o quarto maior mercado de beleza e cuidados pessoais do mundo**, 2020. Disponível em:  
<https://www.forbes.com.br/principal/2020/07/brasil-e-o-quarto-maior-mercado-de-beleza-e-cuidados-pessoais-do-mundo>.