UNIVERSIDADE FEDERAL DO CEARÁ

CAMPUS QUIXADÁ

CURSO DE GRADUAÇÃO EM ENGENHARIA DE SOFTWARE

GERARDO MAGELA DOS SANTOS FILHO

FILA DIGITAL: SOLUÇÃO *MOBILE* PARA CLÍNICAS MÉDICAS

QUIXADÁ

2025

GERARDO MAGELA DOS SANTOS FILHO

FILA DIGITAL: SOLUÇÃO *MOBILE* PARA CLÍNICAS MÉDICAS

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Engenharia de Software do Campus Quixadá da Universidade Federal do Ceará, como requisito parcial à obtenção do grau de bacharel em Engenharia de Software.

Orientador: Prof. Dr. Antônio Joel Ramiro de Castro.

QUIXADÁ

2025

Dados Internacionais de Catalogação na Publicação
Universidade Federal do Ceará
Sistema de Bibliotecas
Gerada automaticamente pelo módulo Catalog, mediante os dados fornecidos pelo(a) autor(a)

S235f Santos Filho, Gerardo Magela dos.
Fila digital: solução mobile para clínicas médicas / Gerardo Magela dos Santos Filho. – 2025.
52 f. : il. color.

Trabalho de Conclusão de Curso (graduação) – Universidade Federal do Ceará, Campus de Quixadá, Curso de Engenharia de Software, Quixadá, 2025.
Orientação: Prof. Dr. Antônio Joel Ramiro de Castro.

1. Desenvolvimento de tecnologia. 2. Padrões de software. 3. Softwares. I. Título.

CDD 005.1

GERARDO MAGELA DOS SANTOS FILHO

FILA DIGITAL: SOLUÇÃO *MOBILE* PARA CLÍNICAS MÉDICAS

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Engenharia de Software do Campus Quixadá da Universidade Federal do Ceará, como requisito parcial à obtenção do grau de bacharel em Engenharia de Software.

Aprovada em: 30/07/2025.

BANCA EXAMINADORA

Prof. Dr. Antônio Joel Ramiro de Castro (Orientador)
Universidade Federal do Ceará (UFC)

Prof. Me. Thiago Werlley Bandeira da Silva
Universidade Federal do Ceará (UFC)

Prof. Dr. Jose Gadelha Da Silva Filho
Universidade Estadual do Ceará (UECE)

À minha família, por sua capacidade de acreditar em mim e investir em mim. Mãe, seu cuidado e dedicação foi que deram, em alguns momentos, a esperança para seguir. Pai, sua presença significou segurança e certeza de que não estou sozinho nessa caminhada.

AGRADECIMENTOS

Agradeço profundamente à minha família, que sempre esteve ao meu lado, oferecendo amor, paciência e incentivo nos momentos mais desafiadores da minha trajetória. Aos meus amigos, que souberam respeitar meus momentos de silêncio e comemoraram comigo cada conquista, deixo também minha sincera gratidão. O apoio de cada um foi essencial não apenas durante o período da graduação, mas em diversos momentos da minha vida, contribuindo para a pessoa e o profissional que estou me tornando.

Ao meu orientador por não ter desistido de mim depois de tantos semestres.

Aos professores participantes da banca examinadora Thiago Werlley Bandeira da Silva e José Gadelha da Silva Filho pelo tempo, pelas colaborações e sugestões.

RESUMO

Este documento técnico descreve o processo de desenvolvimento do aplicativo Fila Digital, que tem como objetivo realizar retirada de senhas em um sistema de fila online para clínicas médicas. O aplicativo multiplataforma almeja otimizar o gerenciamento de filas para atendimento em clínicas médicas de forma online. Foi desenvolvido com uso da biblioteca *React Native* com *Expo*, na linguagem de programação *Typescript* . O Fila Digital permite a retirada de senha, acompanhamento da fila de senhas chamadas e o tempo de espera que o usuário tem até ser atendido. Este relatório apresenta o aplicativo Fila Digital, seu processo de implementação, e as escolhas técnicas em torno de seu desenvolvimento.

Palavras-chave: react native; multiplataforma; typescript; fila; clínica médica

ABSTRACT

This technical document describes the development process of the Fila Digital application, which aims to enable ticket withdrawal in an online queue system for medical clinics. The cross-platform application seeks to optimize queue management for medical clinic appointments in an online environment. It was developed using the React Native library with Expo, in the TypeScript programming language. Fila Digital allows users to withdraw a ticket, monitor the queue of called numbers, and view the estimated waiting time until they are attended to. This report presents the Fila Digital application, its implementation process, and the technical decisions surrounding its development.

Keywords: react native; cross-plataform; typescript; queue; medical clinics

LISTA DE FIGURAS

LISTA DE CÓDIGOS-FONTE

SUMÁRIO

1 INTRODUÇÃO

A alta demanda por atendimentos em clínicas médicas tem causado desconforto e longas filas de espera em muitas clínicas e unidades de saúde. Esse problema afeta diretamente a qualidade de vida dos pacientes e gera aglomeração desnecessária (Barbosa *et al.*, 2019). Diante desse fato, é clara a necessidade de uma solução tecnológica capaz de otimizar o processo de atendimento, oferecendo conforto e autonomia para os pacientes, além de facilitar a gestão de fluxo de pessoas nas unidades de saúde.

O uso da tecnologia em ambientes clínicos é uma das estratégias que tem se mostrado bastante útil e utilizado cada vez mais na sociedade atual. Aplicativos móveis desempenham um papel essencial nesse caso, pois aproveitam a facilidade de uso dos *smartphones* na sociedade para oferecer acesso rápido e prático a serviços diversos, incluindo agendamento de consultas e acompanhamento de exames (Ribeiro *et al.*, 2021).

O desenvolvimento de *software* vem evoluindo nos últimos anos para a criação de múltiplos produtos, vindos de uma plataforma inspirada em uma arquitetura comum e integrados com outros sistemas por meio de redes de atores e artefatos (Manikas, 2016). Por conta de alto índice de uso de *smartphones* e também pela sua praticidade de uso, cria-se uma procura maior em desenvolver uma aplicação *mobile*. A partir disso, propõe-se que o aplicativo realize o gerenciamento e criação de filas online, permitindo que, através da aplicação, os responsáveis possam executar as funcionalidades que o aplicativo irá possuir.

Frameworks como o *React Native*, que permitem a criação de aplicações *mobile* tanto para *Android* quanto para *iOS* com uma única base de código, tornam-se cada vez mais populares entre os desenvolvedores (Dabit, 2021). Por estes motivos, almeja-se propor uma solução composta do desenvolvimento de uma aplicação *mobile* em React Native utilizando *Typescript* e *Expo*, com o objetivo de permitir que os pacientes acompanhem em tempo real sua posição na fila para que possam ir ao estabelecimento apenas quando for necessário, gerando conforto, segurança e eficiência. Tendo em vista que os pacientes poderão se movimentar livremente em locais com acesso à internet, ao invés de permanecer no setor de espera. A solução propõe atender tanto os pacientes quanto os profissionais do atendimento das clínicas.

Este trabalho está organizado da seguinte maneira: a seção 2 apresenta os conceitos necessários para o entendimento deste trabalho. Na seção 3, será feito um relato dos trabalhos relacionados. Na seção 4, são apresentados os procedimentos metodológicos que foram utilizados neste trabalho, e na seção 5 serão mostrados os resultados preliminares.

1.1 Objetivos

1.1.1 Objetivo geral

Desenvolver uma aplicação *mobile* para o controle de senhas em filas online para consultórios médicos, com a funcionalidade de visualizar o tempo de espera médio dos atendimentos baseado nas chamadas de senhas e visualizar as últimas senhas chamadas.

1.1.2 Objetivos específicos

- Desenvolvimento de um aplicativo para antecipar a fila com uma senha virtual.
- Implementar o sistema para clientes retirarem senhas.
- Implementar um sistema para o administrador controlar o fluxo de senhas.

2 FUNDAMENTAÇÃO TEÓRICA

Esta seção tem o objetivo de apresentar os conceitos e ferramentas que serão abordados neste trabalho.

2.1 Desenvolvimento *Mobile*

O desenvolvimento para dispositivos móveis tem crescido significativamente nos últimos anos. Com os *smartphones* superando a quantidade de computadores em empresas e domicílios, várias pessoas migraram para a utilização dessa tecnologia (Gaea, 2022). Devido à sua facilidade de uso e fácil acesso, muitas empresas têm buscado oferecer soluções e comodidade para seus clientes utilizarem alguns de seus serviços. Por esse motivo, há uma certa competitividade dominada entre as empresas que utilizam como principal sistema operacional o *Android*, cerca de 72% dos usuários, e *iOS*, com aproximadamente 27% de usuários (GlobalStat, 2022).

O desenvolvimento de aplicações para um sistema operacional específico é conhecido como desenvolvimento nativo, ou seja, uma aplicação que irá funcionar apenas em *Android* ou *iOS*. No entanto, com o avanço da tecnologia, é possível utilizar de um desenvolvimento híbrido no qual uma mesma aplicação servirá tanto para *Android* quanto *iOS*, trazendo uma otimização maior durante o desenvolvimento.

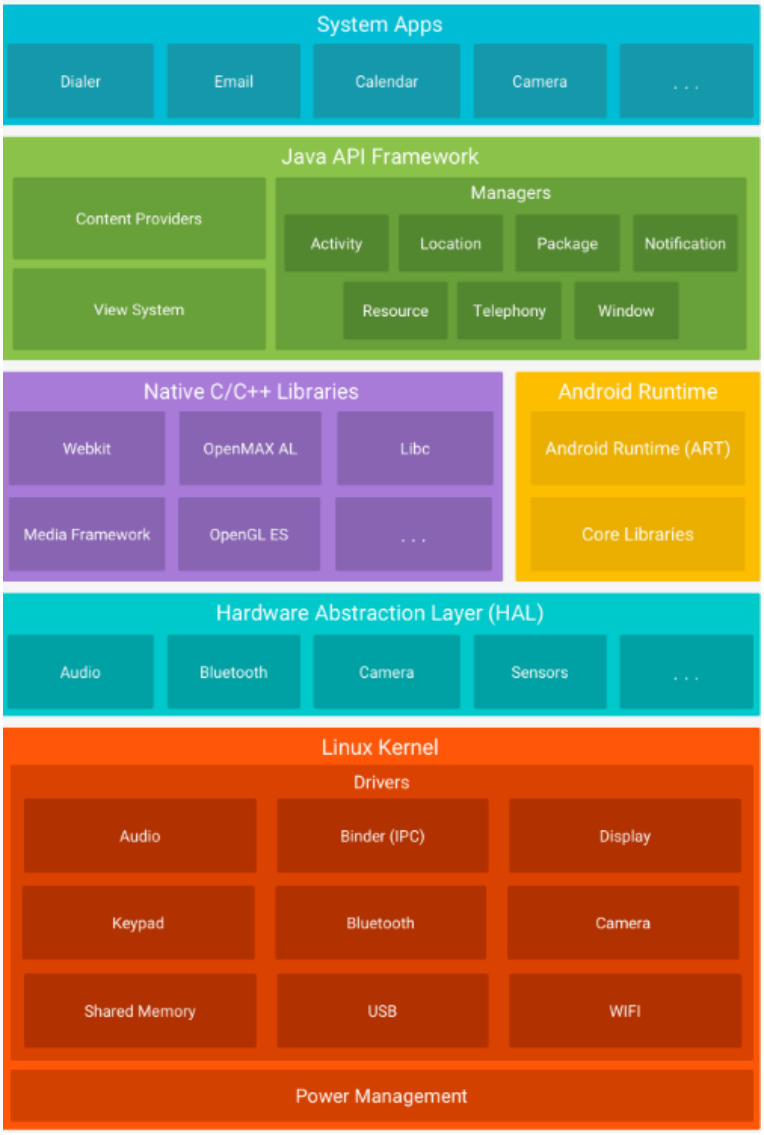
2.1.1 Android

O *Android* é um sistema operacional *open source* baseado no *kernel* Linux para o controle das tarefas do sistema como gerenciamento de memória, pilha de rede, segurança, gerenciamento de processos e modelo de *driver* (GoogleInc, 2022). O *kernel* funciona basicamente como uma camada de abstração do *hardware* e a pilha de *software*, onde futuramente o desenvolvimento passou a ser de responsabilidade da *Open Handset Alliance* (OHA), um consórcio de empresas que busca melhorar os dispositivos móveis.

Segundo o autor Lee (2012), a vantagem de se utilizar o *Android* é que ele proporciona uma abordagem única para o desenvolvimento de *apps* no qual os programadores precisam apenas desenvolver para este sistema operacional e seus aplicativos vão ser capazes de rodar em uma quantidade enorme de dispositivos *Android*, que possuem maior número de usuários ativos. Para entender o funcionamento do *Android* é preciso conhecer sua arquitetura, como é mostrado

na Figura 1.

Figura 1 – Arquitetura do sistema *Android*



Fonte: GoogleInc (2022).

A arquitetura *Android* possui cinco camadas (GoogleInc, 2022):

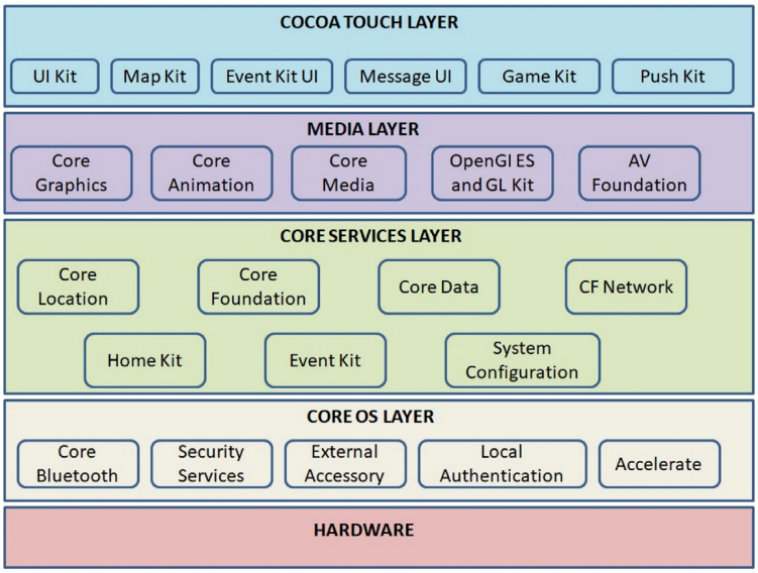
- **Aplicativos:** É a primeira camada, onde fica os aplicativos nativos e aplicativos instalados do dispositivo.
- **Framework:** A próxima camada é o local onde os componentes de gerenciamento das *activities, views* e janelas são encontrados para que os desenvolvedores possam utilizá-las no desenvolvimento.
- **Android Runtime:** Nesta camada é onde os aplicativos são executados em um processo com uma instancia própria do android runtime, projetado para executar várias maquinas virtuais em dispositivos com pouca memoria.

– **Bibliotecas:** Camada onde é encontrada várias bibliotecas e contem todo o código das principais funcionalidades do sistema.
– **Kernel linux:** E na última camada e de mais baixo nível, o *kernel* serve como uma camada de abstração entre o *hardware* e as camadas superiores.

2.1.2 iOS

iOS é o sistema operacional desenvolvido pela empresa *Apple* baseada no sistema MAC OS X e feito para atender a necessidade de *smartphones, iPads, Apple TV* desenvolvidas pela *Apple*. Este sistema faz uma abstração da comunicação do aplicativo com o *hardware*. Apesar de ser baseada no *MacOS*, o *iOS* tem tecnologias que facilitam o uso de *iphones* como interfaces de múltiplos toques e suporte de acelerômetro (Mendonça *et al.*, 2011). Como é mostrado na Figura 2, as diferentes camadas que a arquitetura *iOS* possui são:

Figura 2 – Arquitetura do sistema iOS



Fonte: Garg e Baliyan (2021).

– ***CocoaTouch*:** Camada que fornece a infraestrutura para implementar as aplicações de interface do iPhone, semelhante ao framework do Android.
– ***Media*:** Camada que fornece recursos de audio e video, tecnologia de experiencia de multimidia encontradas nas bibliotecas fornecidas.
– **Core services:** Esta camada fornece serviços esseciais do sistema.
– ***Core OS*:** Camada que se encontra o kernel do sistema, os drivers e as interfaces.
– ***Hardware*:** Por fim, esta camada contem os chips físicos, os circuitos do iOS.

2.1.3 React Native

React Native é um *framework open source javascript*, introduzido pelo *Facebook* em 2015, para desenvolvimento de aplicações híbridas, tanto pra *Android* quanto *iOS*. "Aprenda uma vez, escreva em todo lugar"(Meta Platforms, Inc., 2025), lema adotado pela empresa, esta plataforma permite aos desenvolvedores reutilizar conhecimentos das tecnologias *web* para o desenvolvimento de aplicativos móveis, devido à similaridade com o *ReactJS* por ser sua base, porém ao invés do foco ser os navegadores, o *React Native* foca em plataformas *mobile*, fazendo com que desenvolvedores *web* possam criar aplicações *mobile* que realmente parecem ser "nativas". Porém o *React Native* não possui o mesmo comportamento de um site, executando no *WebView*, é feita uma interpretação dos componentes primitivos escritos em *Javascript* e os renderizam como componentes nativos do sistema. O *React native* utiliza bibliotecas próprias ou de terceiros para acessar as camadas nativas (Bezerra, 2022).

Este *framework*, parecido com *React* para *web*, usa uma mistura de *XML* e *Javascript*, conhecida como *JSX*, para a escrita de suas aplicações e então cria uma "ponte"para a chamada das *APIs* de renderização nativa em *Objective-C*, para *iOS*, ou *Java*, para *Anroid*. Dessa forma, o aplicativo será renderizado usando componentes reais de *interface mobile* e não as *webviews* e possuirá a aparência de qualquer outra aplicação móvel, podendo acessar também recursos da plataforma como a câmera do celular e *GPS*, para obtenção da localização do usuário. Atualmente, o *React Native* suporta *iOS* e *Android* mas tem margem para expandir para outras plataformas futuramente. No entanto, uma limitação que esta plataforma possui é a de poder utilizar apenas os recursos suportados pelo ambiente *React Native* ou os fornecidos por terceiros (Eisenman, 2015).

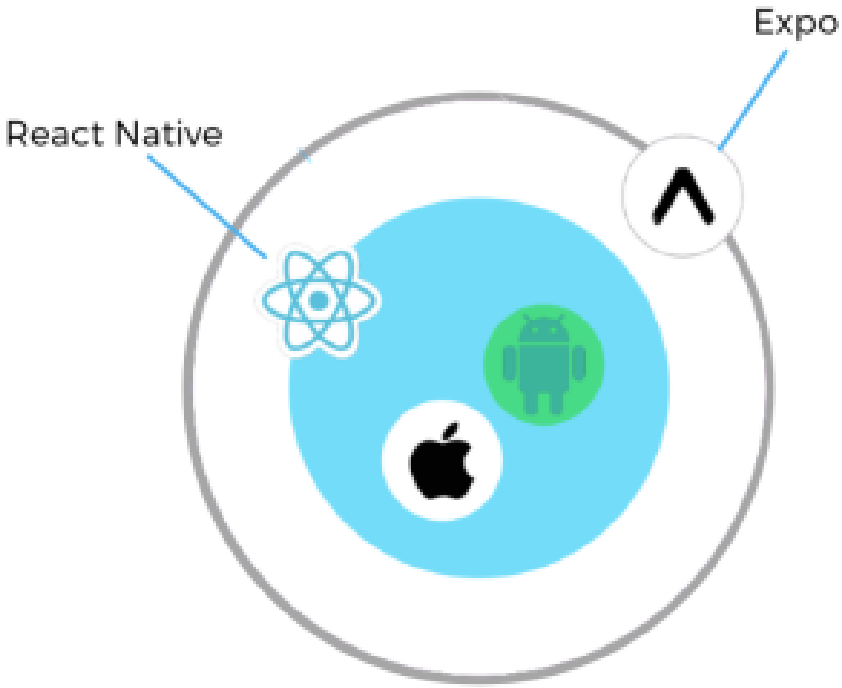
2.1.4 Expo

O *Expo* é um *framework* feito para ajudar programadores com o desenvolvimento de aplicações *React* em geral, mas principalmente para utilização em aplicativos móveis com *React Native*. Ele é formado por um conjunto de ferramentas e serviços criados com o foco nos ambientes do *React Native* e das plataformas nativas, que auxiliam no desenvolvimento, construção, implantação, testes e na execução de simuladores nos sistemas *Android* e *iOS* utilizando a mesma base de código (Expo Documentation, 2025).

O *Expo* funciona como uma camada de abstração para uma aplicação *React Native*.

Ele elimina a necessidade do desenvolvedor ter de lidar com códigos das plataformas nativas, o que pode ocorrer quando se está utilizando apenas o *React Native*. Dessa forma, é preciso lidar somente com *JavaScript* no ambiente de desenvolvimento (FALCÃO, 2022). Como visto na Figura 3, por estar em uma camada acima do *React Native*, algumas *APIs* ainda não estão disponíveis em seu ambiente. Por conta disso, não é possível acessar a *API* de *bluetooth*, por exemplo, porém o *Expo* está evoluindo gradualmente para sanar esses problemas.

Figura 3 – Camadas de abstração do React Native e Expo



Fonte: (FALCÃO, 2022).

2.2 *Design Thinking*

Segundo Silva *et al.* (2023), o *Design Thinking* pode ser definido como uma abordagem metodológica focada no ser humano, utilizada para o desenvolvimento de soluções inovadoras. A metodologia é composta por empatia, definição, ideação, prototipagem e teste, que permitem ao desenvolvedor compreender as necessidades dos usuários e validá-las em ciclos curtos.

Além disso, (Rosado; Dias, 2024) destacam que o *Design Thinking* vem sendo amplamente adotado em projetos acadêmicos e corporativos devido à sua flexibilidade e capacidade de integrar diferentes perspectivas. A abordagem gera uma maior colaboração entre equipes e promove a experimentação contínua, reduzindo riscos e custos relacionados a retrabalhos. Com

isso, essa metodologia é cada vez mais utilizada em diversas áreas como: saúde, tecnologia e educação.

2.3 Filas em clínicas médicas

A grande procura por atendimento em clínicas médicas tem gerado filas enormes, muitas vezes, e, consequentemente, um maior tempo de espera para ser atendido, causando desconforto nos pacientes. A demora no acesso aos serviços compromete a qualidade do atendimento e gera insatisfação dos pacientes (Barbosa *et al.*, 2019). Com essa problemática, algumas iniciativas surgem no meio do atendimento na área de saúde que visam modernizar o atendimento médico por meio de ferramentas, sistemas ou aplicativos. Aplicações móveis, como a citada neste trabalho, têm uma proposta de reduzir o tempo de espera nas clínicas, melhorar o fluxo de pacientes e facilitar o tempo médio do que falta para ser atendido.

2.4 Notificações em tempo real com WebSockets

As notificações *push* são um mecanismo básico em qualquer aplicação *mobile* para informar aos usuários sobre eventos nas aplicações em uso ou em segundo plano que estejam a utilizar. No caso de aplicações referentes a clínicas médicas, o aplicativo deve informar aos pacientes quando sua vez de ser atendido estiver próxima, otimizando o fluxo de atendimento e a experiência de usuário.

Em relação à comunicação em tempo real, por meio de *WebSockets* ou *Socket* e *STOMP*, eles possibilitam que a funcionalidade atualize imediatamente na posição da fila, melhorando a experiência do usuário ao evitar que o mesmo seja mantido em tela para que possa ser visualizado se a senha foi chamada ou não, um funcionamento crucial em ambientes que exigem precisão nas informações apresentadas (MDN Web Docs, 2023).

3 TRABALHOS RELACIONADOS

De fato, os celulares estão cada vez mais presentes em nossa vida, facilitando as tarefas diárias e a forma como interagimos com a tecnologia diretamente. Nesta seção, serão apresentados diversos trabalhos que utilizam um aplicativo para uma solução dos problemas apresentados nos artigos. Por fim, serão apresentadas comparações entre os trabalhos relacionados e o que cada um pode agregar a este projeto.

3.1 Sistema para gestão da fila de espera em pronto-atendimento pediátrico usando aplicativo móvel

Neste artigo, o autor (Souza, 2016) apresenta o projeto de um sistema baseado em gerenciamento de filas online para pronto atendimento pediátrico através de *smartphones*. Ele descreve que o objetivo do aplicativo é melhorar a forma como os pacientes têm acesso ao atendimento dos serviços prestados pelo pronto socorro.

Em respostas aos mais diversos problemas apresentados, o autor, que foi o responsável pelo desenvolvimento do sistema, permitiu que os usuários acompanhassem a posição na qual estavam na fila em tempo real e recebessem notificações sobre a chamada da fila, mostrando o progresso do atendimento no qual se encontram. Além disso, o aplicativo oferece aos profissionais suporte para facilitar a gestão dos atendimentos e comunicação com os pacientes. A implementação do aplicativo em unidades de pronto-atendimento indicaram uma melhoria substancial na satisfação dos usuários e na otimização dos processos internos promovendo um atendimento mais ágil e organizado.

A motivação para esta solução vem dos problemas constantes em diversos hospitais do Brasil. Porém, não existem pesquisas de aplicativo que abordem esse problema. Neste aspecto, com baixos estudos com intervenções no problema e que buscam minimizar a dificuldade da superlotação pelo uso de novas tecnologias, (Souza, 2016) propôs este estudo. Ele adota uma abordagem com a aplicação de um sistema utilizando como apoio um aplicativo móvel, no qual analisa os efeitos da fila de espera da unidade do pronto-atendimento.

Neste trabalho, o autor utilizou o desenvolvimento das páginas estáticas do aplicativo com o construtor, em sua versão paga, de aplicativos da Fábrica de Aplicativos, onde possui o básico dos *templates* a ser utilizados em plataformas *Android* e *iOS* utilizando HTML5. As funcionalidades principais nas páginas dinâmicas do aplicativo utiliza da ferramenta *Adobe*

Dreamweaver, escrito em *HTML, SQL, PHP e Javascript*.

Com isso, espera-se que a avaliação das metodologias propostas pelo autor contribua com o desenvolvimento do aplicativo para redução das filas em clínicas médicas, visto que também tem a intenção de avaliar a satisfação dos usuários.

3.2 Waitz: uma aplicação web para filas de espera em hospitais e clínicas médicas

Segundo o trabalho do autor (Santos, 2024), a plataforma propõe solucionar os problemas de filas médicas, mostrando o processo de implantação e também os resultados encontrados em sua pesquisa. O autor cita que a plataforma será colaborativa e em tempo real, onde os pacientes poderão realizar *check-ins* e *check-outs*, visualizar a quantidade de pacientes em espera de atendimento numa unidade médica e também avaliar, por comentários, tanto os atendimentos quanto os locais.

A plataforma foi construída em uma aplicação web intuitiva, utilizando o *framework React* e *Google maps*, com um *web server* para controlar a lógica de eventos e um banco de dados. Essa abordagem tem a finalidade de criar uma interface de usuário, onde estão disponibilizadas as unidades de saúde em um mapa, para os usuários conseguir localizar as unidades mais próximas de sua localização. No lado da camada de acesso, foi utilizada a biblioteca *Express* e *SocketIO* para a criação de uma *API*, os quais tratarão todos os dados e eventos provenientes dos navegadores e também fornecerão informações relevantes para os usuários (Santos, 2024).

Com isso, espera-se que a avaliação da implantação propostas pelo autor contribua com o desenvolvimento do aplicativo para redução das filas em clínicas médicas, visto que também tem a intenção de avaliar a satisfação dos usuários e a utilização de ferramentas mencionadas.

3.3 Desenvolvimento de *app* para monitoramento de filas de um instituto oftalmológico

O autor levanta diversos pontos referentes à gestão de sistema na área da saúde, principalmente com o aumento da demanda por esses serviços. Logo, Gomes (2019) propôs o desenvolvimento de um aplicativo móvel para monitorar o fluxo de pacientes em serviços de saúde, com foco na redução do tempo de espera e melhoria da organização do atendimento (Bomfim *et al.*, 2022).

Em seus estudos, foi feita uma análise quantitativa dos processos de chegada, bus-

cando informações acerca do desempenho do atendimento, tal como o tempo médio de espera, tempo médio de atendimento, ou a probabilidade de encontrar o sistema ocioso ou lotado (Lim et al., 2014), tempos estes realizados através de dados obtidos pela equipe de tecnologia do instituto. Eles avaliaram os dados de filas por um determinado período de tempo para calcular o tempo médio de espera para haver uma comparação de evolução entre o uso da solução proposta para as filas atualmente sem a solução.

O autor utilizou como ferramenta o *Figma*, para prototipagem, *Firebase*, para o *backend*, e a construção do aplicativo com *Android Studio* com *plugins flutter* e *dart*, emulador de *android* oferecido pelo próprio *Android Studio* e também um *smartphone* com sistema operacional *Android*. Espera-se que a avaliação das funcionalidades, metodologias e requisitos desse autor sejam úteis para a construção da solução proposta para este projeto.

3.4 *E-Queue Mobile Application*

Os autores responsáveis do artigo *E-Queue Mobile Application*, buscam desenvolver um aplicativo para substituir o uso de *tickets* de papel em sistemas de gerenciamento de filas. Segundo eles, o uso excessivo de senhas em papel não só apresenta uma ineficácia, como também gera um problema com impacto ambiental por conta do desperdício de papel. A aplicação visa eliminar esses problemas ao permitir que o usuário obtenha o número da fila diretamente por meio de seu *smartphone*, sem a necessidade de impressão física.

O sistema permite que os clientes visualizem, em tempo real, sua posição na fila e o tempo estimado de espera para o atendimento. Ao alterar sua posição aproximada, o aplicativo envia notificações automáticas para os usuários, alertando-os quando sua vez está próxima. Essa funcionalidade oferece maior flexibilidade aos usuários, que podem se movimentar livremente em locais com acesso à internet, ao invés de permanecer no setor de espera.

Além disso, o aplicativo possibilita que o usuário realize ações diretamente por meio do *smartphone*, como cancelar ou adiar a senha, restringindo que o sistema seja apenas informativo e transformando-o em uma interface interativa. Tal interação permite que outras pessoas na fila sejam notificadas antecipadamente, mantendo o fluxo de atendimento e evitando atrasos.

O modelo do sistema foi pensado com foco em dispositivos *Android* 4 e envolve funcionalidades essenciais como login, retirada de senha, visualização da lista de espera, estimativa do tempo de atendimento, envio de notificações, além de permitir ações como cancelamento

ou adiamento. Os objetivos centrais foram proporcionar liberdade ao usuário, reduzir custos operacionais e tornar o processo de espera mais eficiente e sustentável.

3.5 Comparação dos trabalhos

Na Tabela I, apresenta-se uma tabela comparativa entre os trabalhos citados nas subseções anteriores relacionado ao proposto deste trabalho. A solução proposta pelos autores possui apenas uma versão *web* responsiva e outra mobile apenas para *Android*. Um ponto positivo no trabalho *E-Queue Mobile Application*, é o fato de o usuário poder cancelar ou adiar a senha obtida, dessa forma, os outros usuários na fila são notificados sobre a nova posição da fila em que se encontram. A ideia deste trabalho é trazer uma solução envolvendo tempo de espera médio de atendimento em uma plataforma híbrida, tanto pra *android* quanto para *iOS*, devido a grande variação do uso de sistemas operacionais nos *smartphones*.

Tabela 1 – Comparativo entre os trabalhos relacionados e o proposto

Trabalho	Tecnologia	Aplicação	Plataforma
Souza (2016)	Fabrica de aplicativo	Gestão de fila em pronto-atendimento pediátrico com check-in remoto e tempo estimado de espera	Mobile
Santos (2024)	React, Express	Gerenciamento de fila de espera em clínicas e hospitais, com suporte a agendamento e atualização de posição	Web
Bomfim *et al.* (2022)	Android studio, Firebase	Monitoramento da fila de um instituto oftalmológico com estimativa de tempo e visualização em tempo real	Mobile Android
Este Trabalho	React Native, Expo, TypeScript	Controle de filas em clínicas médicas com acompanhamento em tempo real, alertas e priorização	Mobile Híbrido

Fonte: Elaborado pelo autor.

4 METODOLOGIA

Nessa seção serão mostradas as etapas da metodologia, as coletas de informações, para definir as fases de desenvolvimento e o escopo do projeto baseada no *Design Thinking*, processo no qual será utilizado para o desenvolvimento deste projeto. (Silva *et al.*, 2023) destacam que o *Design Thinking* tem-se mostrado um referencial relevante para a resolução de problemas complexos no campo de inovação em saúde.

Esse trabalho se trata de uma construção de um sistema baseado em um conhecimento previamente existente, unido a uma necessidade de solução de um problema em relação ao tema proposto. Com isso, foram exploradas informações sobre o tema escolhido através de outros estudos na área e foi identificado que há pouco estudo sobre isso, com muito o que ainda ser organizado para solucionar um problema recorrente das filas em clínicas médicas.

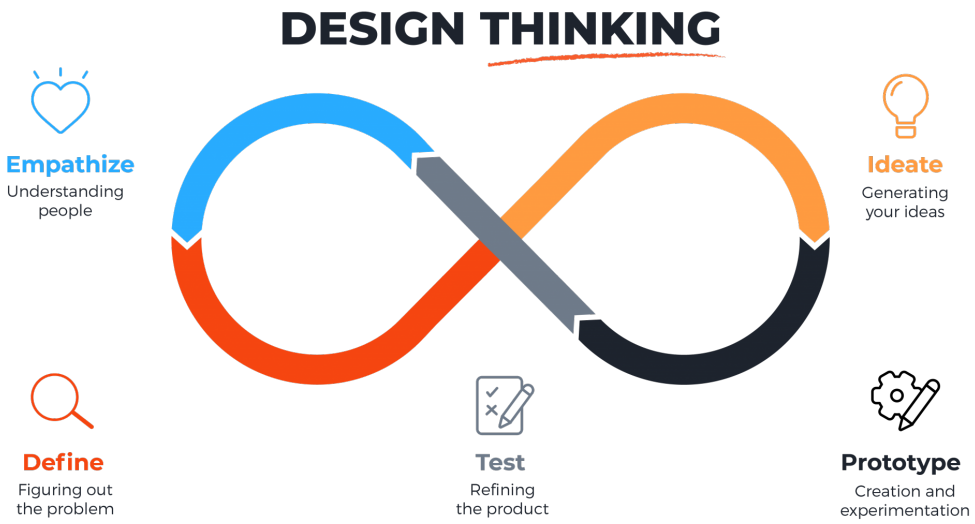
Segundo o autor (Gil, 1991), "as pesquisas descritivas têm como objetivo primordial a descrição das características de determinada população ou fenômeno", que é de fato no que esse trabalho focou, opiniões do público sobre o desenvolvimento de um aplicativo mobile híbrido de gerenciamento e controle de filas online para clínicas médicas e sua adoção através de *feedback* de usuários.

4.1 Design Thinking

Este método foi adotado para a construção dos projetos. Considerando que o período de tempo desde a concepção até à entrega dos protótipos seja o mais rápido possível, a escolha desse método visa justamente a objetividade e uma visão realista do trabalho a ser executado, com prazos e objetivos bem definidos.

O intuito é escolher um problema, realizar a solução a partir de uma abordagem, criar o protótipo, testar e colocar essa solução em prática, como podem ser vistas na Figura 4 onde indica cada etapa. Apesar dessa metodologia ter várias etapas, elas podem ser abordadas de forma iterativa e etapas que já foram realizadas podem ser revisitadas quando necessário.

Figura 4 – Etapas do *Design Thinking*



Fonte: (MAQE, 2022).

4.2 Escolha da ferramenta a ser utilizada

Nesta fase foram avaliadas ferramentas para o desenvolvimento de projetos de aplicação móveis existentes no mercado, com a intenção de escolher o melhor que se encaixe no uso do projeto. Dentre as opções a escolhida o *React Native* foi selecionado devido sua capacidade de gerar aplicações multiplataforma.

4.3 Análise e Desenvolvimento do aplicativo Fila Digital

Durante o desenvolvimento do aplicativo, foram feitas análises da viabilidade do produto através de pesquisas na *internet*, como *blogs*, *sites* e artigos. Também foi feito um protótipo das telas para guiar a escolha de qual *design* ou disposição de componentes utilizar. Após essas etapas, iniciou-se o desenvolvimento do Fila Digital, seguindo os conceitos do *Design Thinking*.

4.4 Protótipo

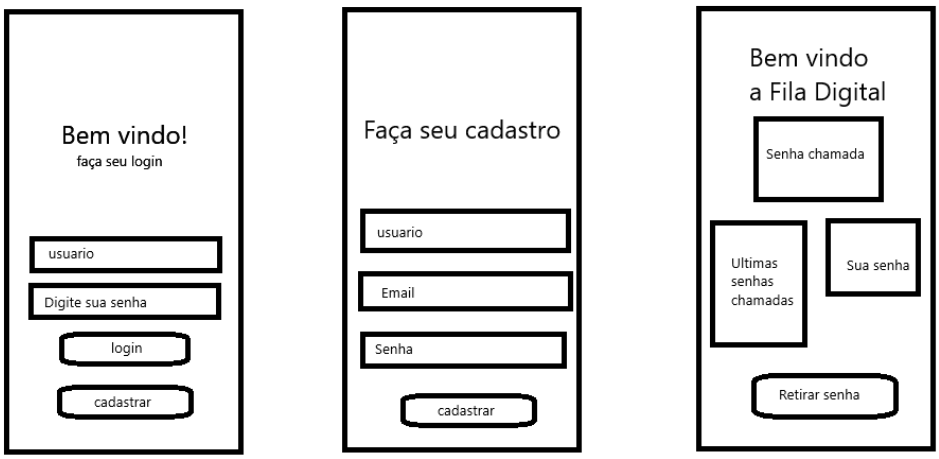
O sistema é um aplicativo *mobile*, logo foram desenvolvidos protótipos para avaliar o *design*, navegação e disposição dos componentes em tela, a fim de escolher a estrutura ideal para uma boa usabilidade por parte dos usuários.

Esse protótipo foi criado de forma simples utilizando a ferramenta *Paint*, como

apresentado na Figura 5, para se ter uma noção de como a *interface* poderia ficar, e utilizou-se como referência um sistema de chamada de senhas utilizado em algumas *fast foods*. O protótipo, de média fidelidade, consiste no *design* inicial das principais telas que estarão presentes no aplicativo.

Ao fim deste trabalho, será apresentado um protótipo desenvolvido em *React Native* com as principais *features* do aplicativo e algumas funcionalidades básicas de navegação entre as telas. O aplicativo foi elogiado e serviu de base para a versão apresentada neste trabalho.

Figura 5 – Protótipo das telas principais



Fonte: Elaborado pelo autor.

4.5 Resultados para análise

Os resultados referentes à coleta de dados serão analisados com o objetivo de reunir mais informações em relação à qualidade e usabilidade da aplicação, para que seja melhor aproveitado para uso em versões futuras.

5 DESENVOLVIMENTO DO APLICATIVO FILA DIGITAL

Esta seção tem como objetivo descrever o processo de desenvolvimento das telas do aplicativo Fila Digital. Apesar do *framework* utilizado funcionar de forma híbrida, maior parte dos testes foram realizados em dispositivo *Android*, devido à facilidade de criação de *build* para testes em dispositivos reais e que sistemas *iOS* exigem *macOS* para geração de arquivo.

Como citado nas metodologias, a abordagem adotada para o desenvolvimento será incremental, com fases definidas e entregáveis em cada etapa da seguinte forma:

- **Planejamento:** Proposta do projeto e identificação do problema.
- ***Design*:** *Design* das telas.
- **Desenvolvimento:** Programação em *React Native* e correções de *bugs*.
- **Protótipo:** Visualização do que foi construído.
- **Implantação:** Criação da *build* para instalação em dispositivos móveis.

5.1 Definição do problema

Como apresentado na introdução deste trabalho, diversas clínicas médicas enfrentam problemas em relação à quantidade de pacientes nos consultórios e dificuldades em gerenciar o tempo das pessoas, forçando os pacientes a aguardarem por horas para serem atendidos, tempo esse que poderia ser melhor aproveitado fora dos estabelecimentos.

No entanto, poucos estudos foram feitos levando em consideração esse tipo de solução. Como consequência, clínicas médicas lotadas, insatisfação de clientes, cancelamentos, desistências e um alto tempo de espera para ser atendido.

5.2 Proposta da solução

Pensando nisso, o aplicativo Fila Digital propõe soluções para minimizar esses impactos e gerar conforto e otimização de tempo para os usuários do aplicativo. Ele se apresenta como uma opção para administradores de empresas médicas e pacientes que buscam um melhor gerenciamento de filas e pacientes que buscam otimizar o tempo necessário em espera para atendimento.

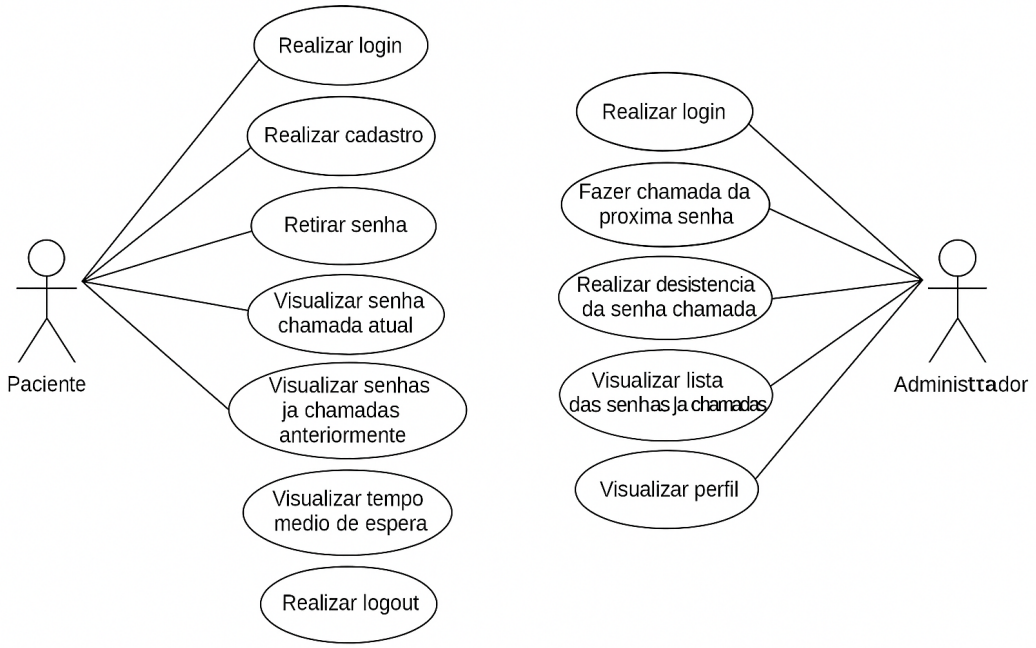
Além de poder ver quais senhas já foram chamadas sem precisar se deslocar até o estabelecimento, também é possível realizar a retirada de senhas, seja ela normal ou prioritária, para seu atendimento desejado na clínica desejada, visualizar o tempo médio de espera até a

chamada da senha retirada, receber notificações para alertar a chamada de senha do paciente. O aplicativo propõe ser simples e de fácil uso.

5.3 Descrição de funcionalidades

A Figura 6 apresenta as funcionalidades básicas do aplicativo no qual os usuários, pacientes e administradores, podem realizar. A seguir, são descritos os casos de uso presentes no diagrama.

Figura 6 – Diagrama geral de casos de uso



Fonte: Elaborado pelo autor.

5.3.1 Caso de uso: Realizar login

Ao iniciar o aplicativo, o usuário é apresentado à tela de *login* na qual deve preencher os campos de usuário e senha e prosseguir no botão Entrar. Após isso, é chamada uma *API* na qual vai fazer a verificação das informações informadas e sua validade no *backend*. No caso de sucesso, o usuário é redirecionado à tela principal das funcionalidades, que pode variar de acordo com o tipo de usuário (paciente ou administrador). Caso as informações estejam erradas, uma notificação aparecerá em tela informando que o usuário ou senha estão incorretas.

5.3.2 Caso de uso: Realizar cadastro

O usuário poderá efetuar *login* no aplicativo após realizar o cadastro, disponível por meio de um *link* na tela de *login* que direciona para o formulário de registro. Após o preenchimento correto de seus dados, o usuário será cadastrado e redirecionado para a tela inicial do aplicativo. Como o *backend* utilizado é externo, não foi possível especificar o tipo de usuário no momento do cadastro, o que torna o registro exclusivo para pacientes. Para ter acesso ao painel de administradores do sistema, é necessário utilizar um cadastro de usuário específico como *admin*.

5.3.3 Caso de uso: Retirar senha

Após o usuário de tipo paciente realizar o *login* e ser redirecionado para a tela principal do aplicativo, é apresentada as principais funcionalidades do Fila digital, uma delas é onde o usuário poderá retirar a senha de atendimento, podendo ser senha normal ou senha prioritária.

5.3.4 Caso de uso: Visualizar senha chamada atual

Após o *login*, os usuários poderão visualizar a senha que está sendo chamada atualmente no período mostrado, na parte superior do aplicativo, não exigindo uma retirada de senha para poder visualizar a informação.

5.3.5 Caso de uso: Visualizar senhas chamadas anteriormente

Os usuários podem visualizar as senhas chamadas anteriormente pelo sistema, com uma leve divergência devido à disposição de tela onde o usuário paciente é limitado a apenas duas últimas chamadas, enquanto o usuário administrador consegue visualizar as últimas seis chamadas.

5.3.6 Caso de uso: Visualizar tempo médio de espera

Após o usuário paciente retirar uma senha, é possível visualizar o tempo médio de espera até que seja chamado ao guichê de atendimento.

5.3.7 Caso de uso: Realizar logout

No canto superior esquerdo da tela há um botão do tipo *hamburger button* que, ao ser tocado, abrirá uma modal com as informações de perfil e um botão de sair. Ao clicar em sair, o usuário será redirecionado para a página de login.

5.3.8 Caso de uso: Realizar chamada da próxima senha

Após o login como administrador, a tela com as principais funcionalidades estará disponível para uso. Uma delas é o botão de chamar próxima senha que, ao ser acionado, realizará a chamada da próxima senha da fila registrada na base de dados.

5.3.9 Caso de uso: Realizar desistência da senha chamada

Após o login como administrador, ao acessar o painel de administração, é possível realizar a chamada de senhas. Com uma senha em atendimento, o administrador pode indicar que o paciente desistiu do atendimento, registrando a ausência ou outro possível problema. Assim, o status do paciente na base de dados é atualizado para “desistiu”.

5.3.10 Caso de uso: Realizar atendimento da senha chamada

Após o login como administrador, ao acessar o painel de administração, é possível realizar a chamada de senhas. Com uma senha em atendimento, o administrador pode indicar que o paciente foi atendido, atualizando o status na base de dados para “atendido com sucesso”.

5.4 Interface

As cores escolhidas para o *Fila Digital* são tons de azul combinados com branco, como ilustrado na Figura 7. Essa paleta foi definida para proporcionar harmonia visual e não comprometer a legibilidade dos textos sobre os diferentes fundos dos componentes.

Figura 7 – Paleta de cores do aplicativo

Fonte: https://www.colorhexa.com/.

A logomarca adotada pode ser vista na Figura 8 tendo uma fonte estilizada e moderna

com os padrões das cores da paleta de cores adotada no projeto.

Figura 8 – logomarca do aplicativo

Fonte: Elaborado pelo autor.

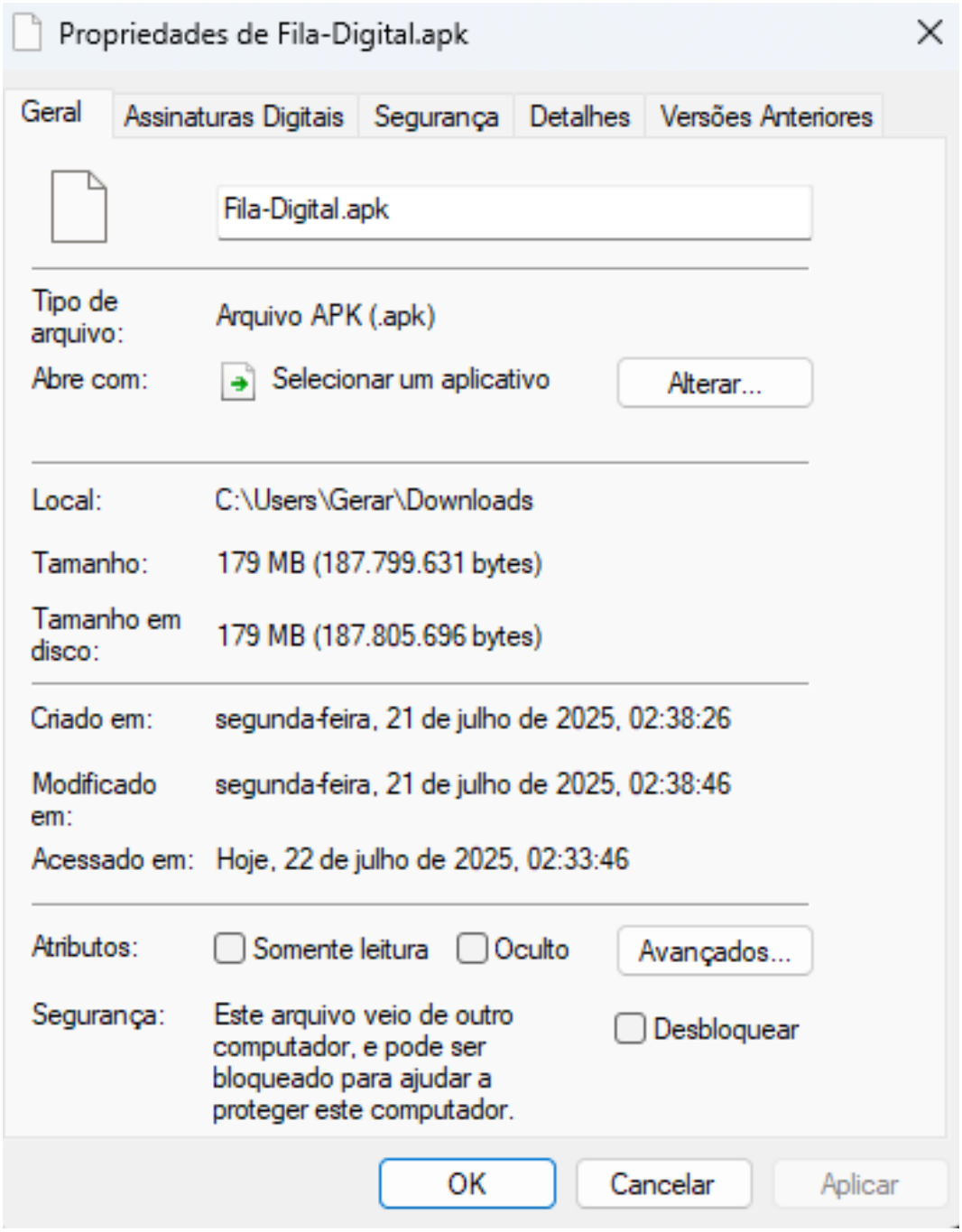
5.5 Produto

Nesta seção, serão apresentadas as telas do aplicativo e seu fluxo de utilização de forma detalhada. Será mostrada a versão utilizada em um dispositivo real, um *tablet Android*, de cada interface através de capturas de tela e para o dispositivo *iOS* será feito através da simulação gerada pela *build* de desenvolvimento, tendo em vista que o dispositivo utilizado para desenvolver foi *Windows* onde não foi possível gerar um executável da *build* para *iOS*.

5.5.1 Dados quantitativos

Após sua finalização, o aplicativo Fila Digital possui um tamanho total de 370 MB (*megabytes*) em disco, já sua versão *APK* possui o tamanho de 179 MB, como pode ser visto na Figura 9. Sua versão de simulação para *iOS* ocupa 98,4 MB. O tamanho dessa *build* se encaixa na média dos aplicativos que são disponibilizados nas lojas dos dispositivos como *play store* e *app store*.

Figura 9 – Tamanho, megabytes, do APK



Fonte: Elaborado pelo autor.

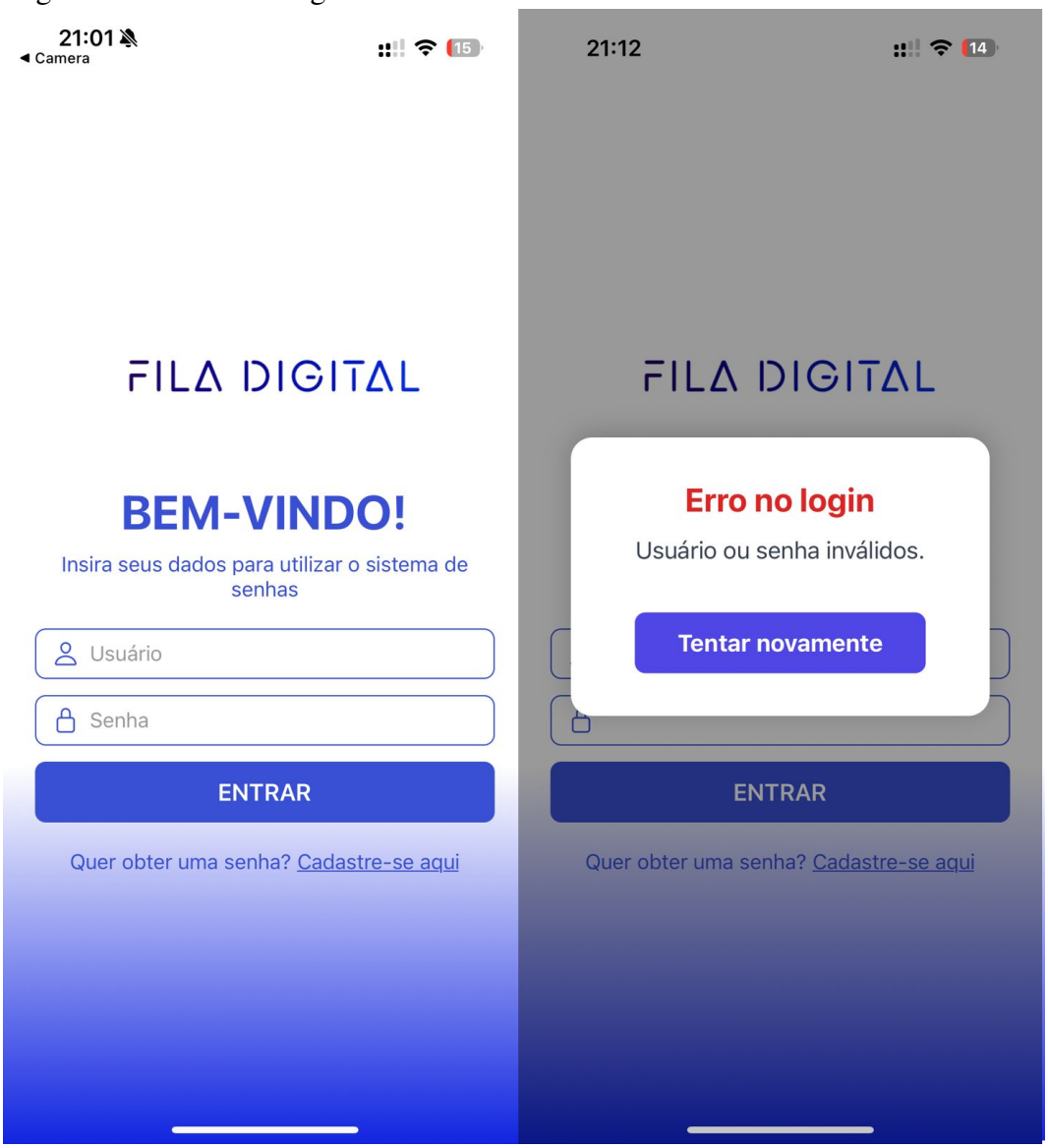
5.5.2 Navegação

O fluxo de navegação no aplicativo Fila Digital é realizado através dos componentes dispostos em tela, campo para digitação e botões para navegação entre as páginas. Botões estes que tratam justamente do acesso para a tela de *login*, a tela principal de senhas, a tela de cadastro e a tela de perfil.

5.5.3 Fluxo de login e cadastro

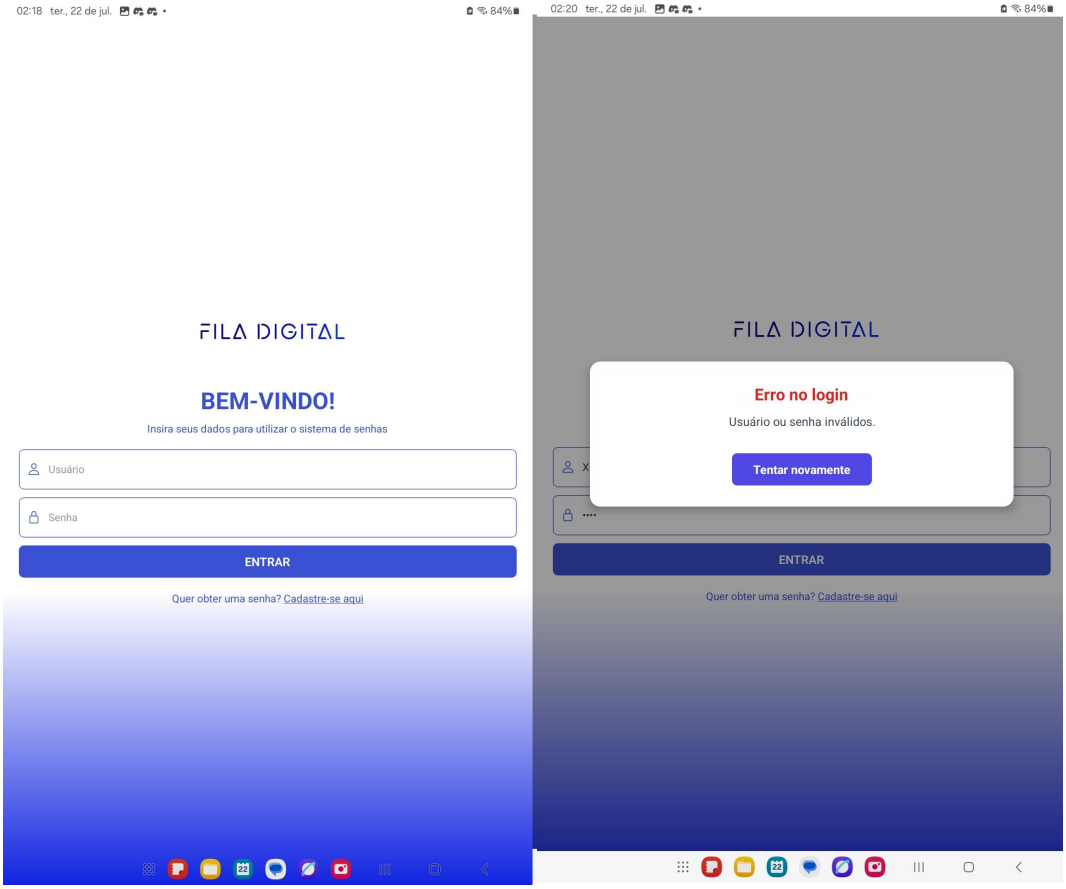
Ao executar o aplicativo, os usuários terão acesso à tela inicial, como mostra na Figura 10 e 11, com as informações de usuário e senha, para serem preenchidas, o botão de entrar e o link de realizar cadastro. Para ter acesso à tela principal com as funcionalidades da fila de senha, o usuário precisará estar logado. Caso o usuário ainda não possua um cadastro, será necessário criar uma conta, apresentada na Figura 12 e 13, fornecendo além do nome de usuário, o *email* e uma senha. As telas do fluxo de autenticação são fornecidas pelo serviço de autenticação do *backend* utilizado.

Figura 10 – Fluxo de login - iOS



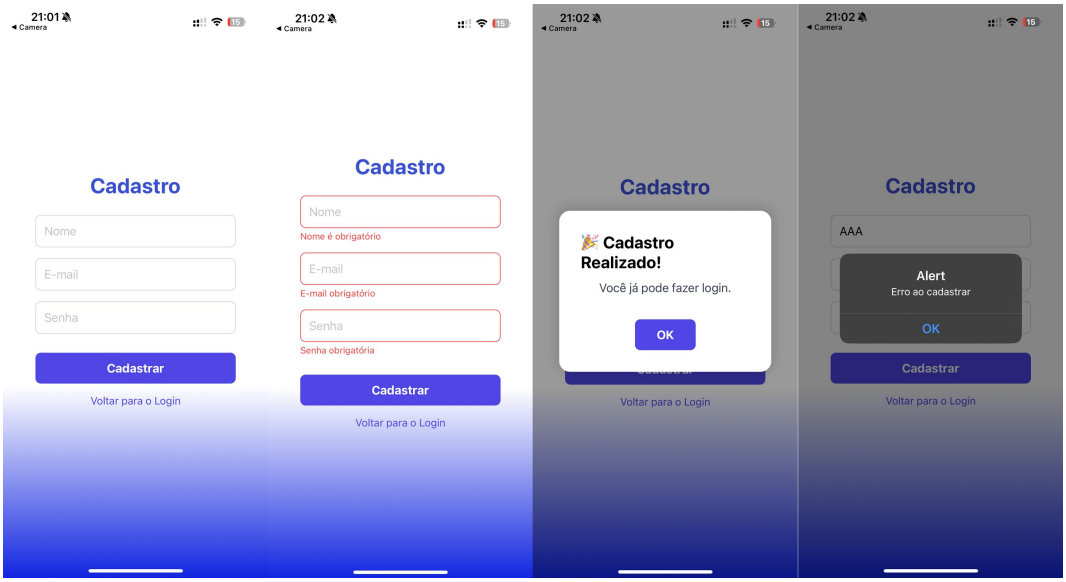
Fonte: Elaborado pelo autor.

Figura 11 – Fluxo de login - Android



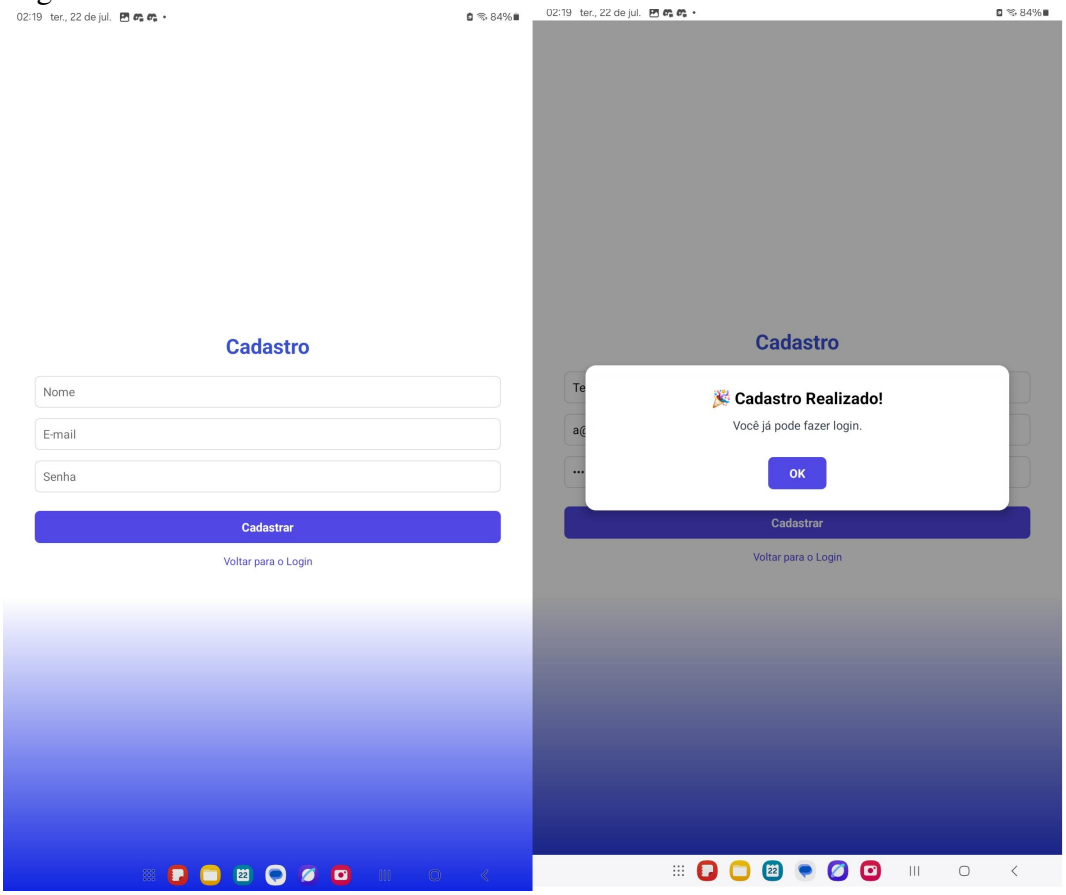
Fonte: Elaborado pelo autor.

Figura 12 – Fluxo de cadastro - iOS



Fonte: Elaborado pelo autor.

Figura 13 – Fluxo de cadastro - Android

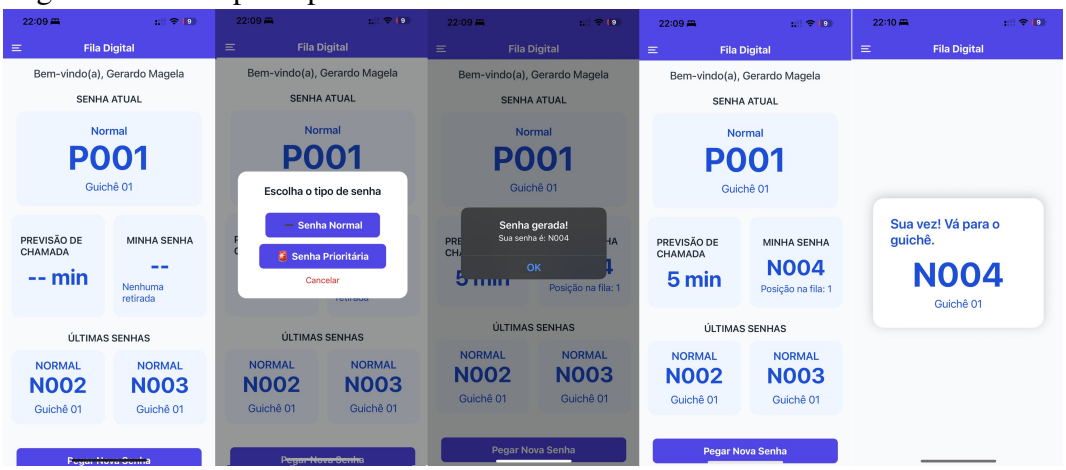


Fonte: Elaborado pelo autor.

5.5.4 Fluxo de telas principais

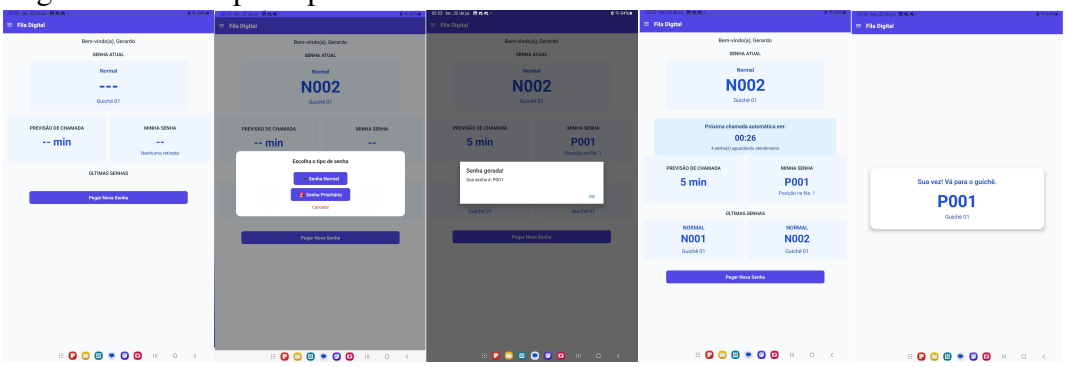
A primeira tela exibida após logar com sucesso no aplicativo, para o usuário paciente, apresenta a tela onde consta as principais funcionalidades do app. Para estes usuarios, é apresentado uma tela com alguns cards com informações como senha atual da fila, tempo de espera, a senha do usuario, as ultimas senhas chamadas e o botão de retirar senha. Esse botão carrega uma modal com mais dois botões com a retirada de senha normal e a senha prioritária.

Figura 14 – Tela principal - iOS



Fonte: Elaborado pelo autor.

Figura 15 – Tela principal - Android

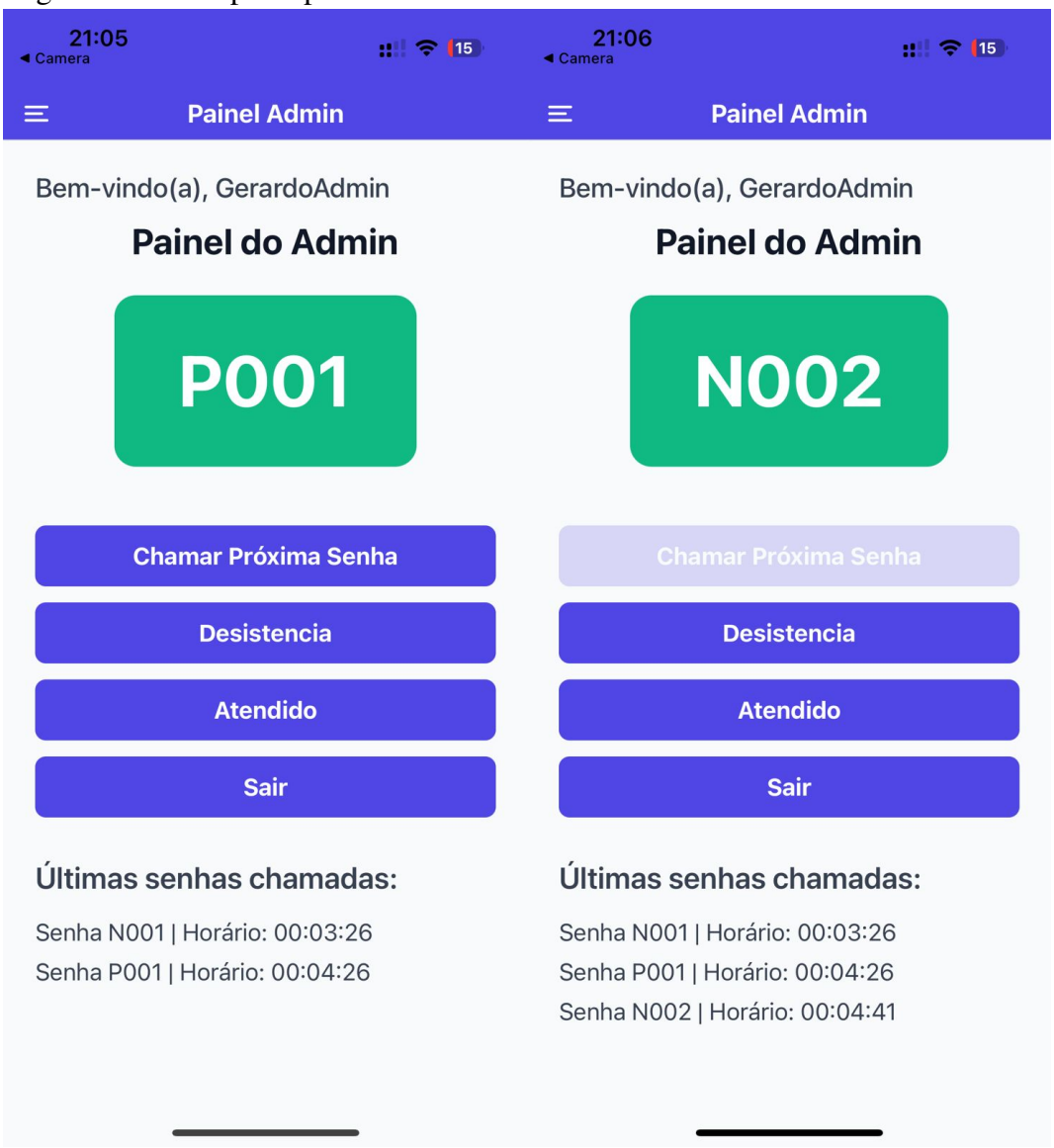


Fonte: Elaborado pelo autor.

5.5.5 Fluxo de telas principais de administrador

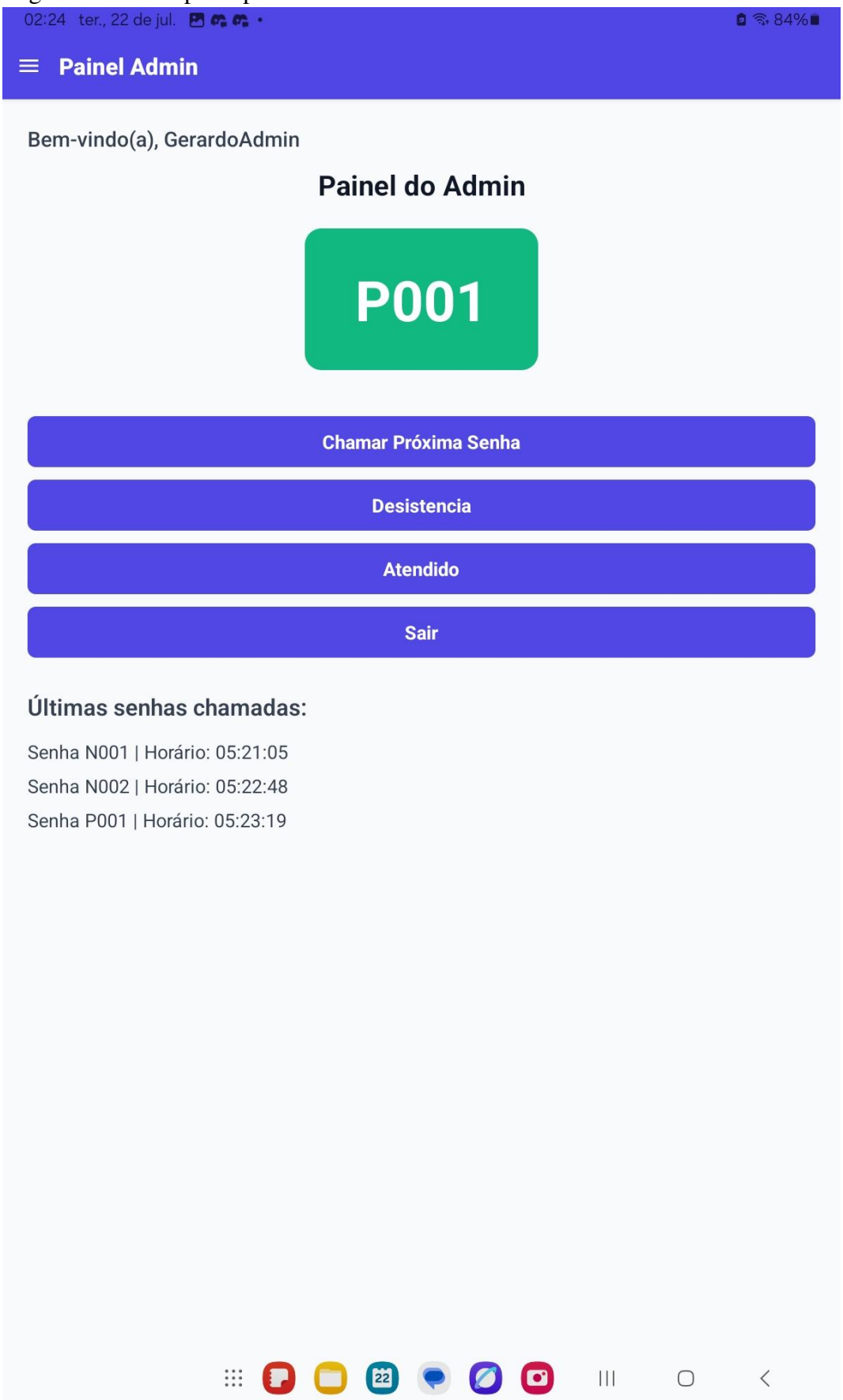
A tela exibida após logar com sucesso no aplicativo, nas Figuras 14 e 15, para o usuário administrador, apresenta o painel de administrador, onde constam as principais funcionalidades do *app* como a de chamada da próxima senha registrada na base de dados, o botão de desistência que informa se aquela senha chamada desistiu ou se ausentou de ser atendido e, por fim, o botão de atendido, que indica que a senha do paciente foi chamada e atendida com sucesso.

Figura 16 – Tela principal de administrador - iOS



Fonte: Elaborado pelo autor.

Figura 17 – Tela principal de administrador - Android

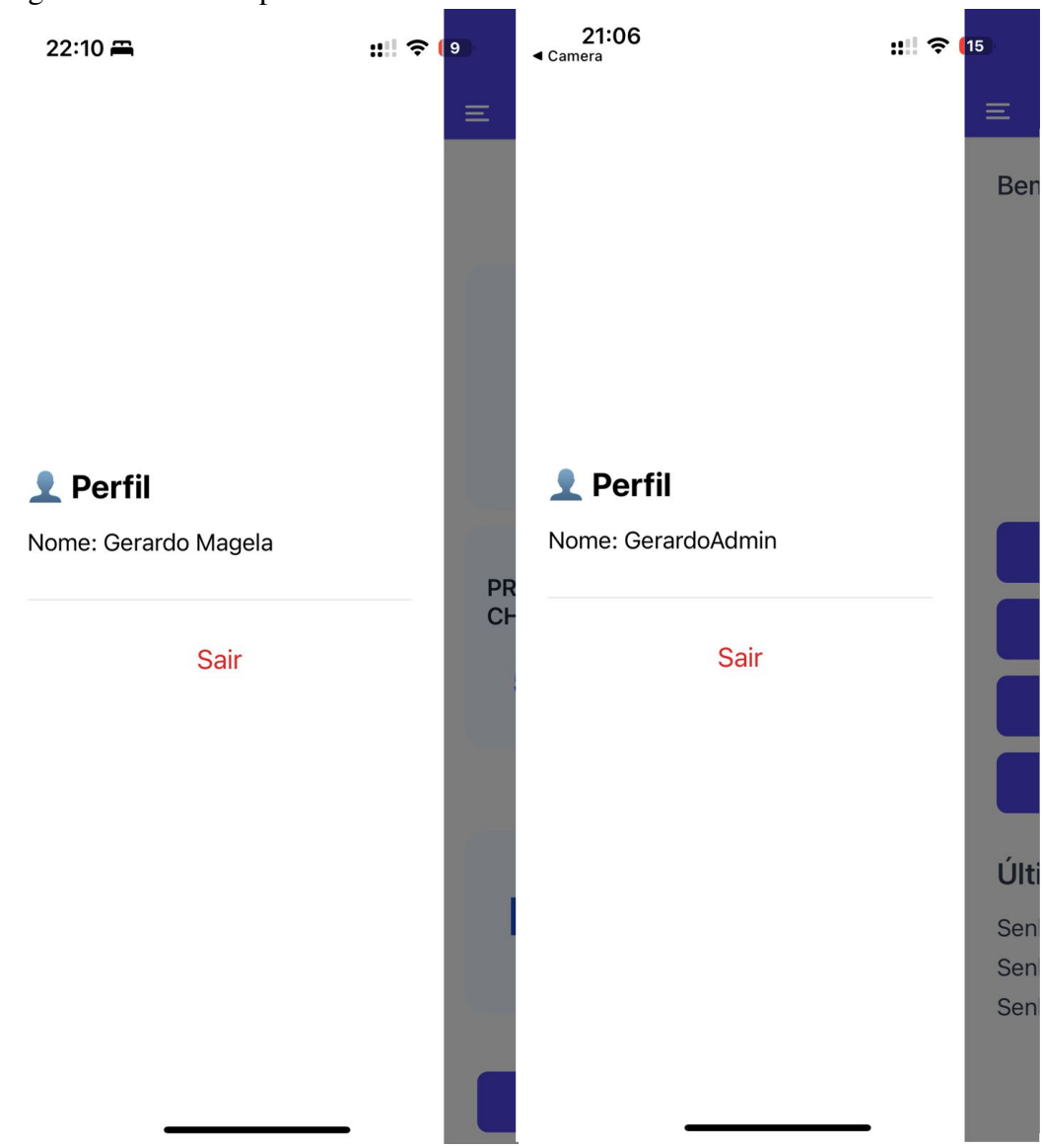


Fonte: Elaborado pelo autor.

5.5.6 Tela de perfil

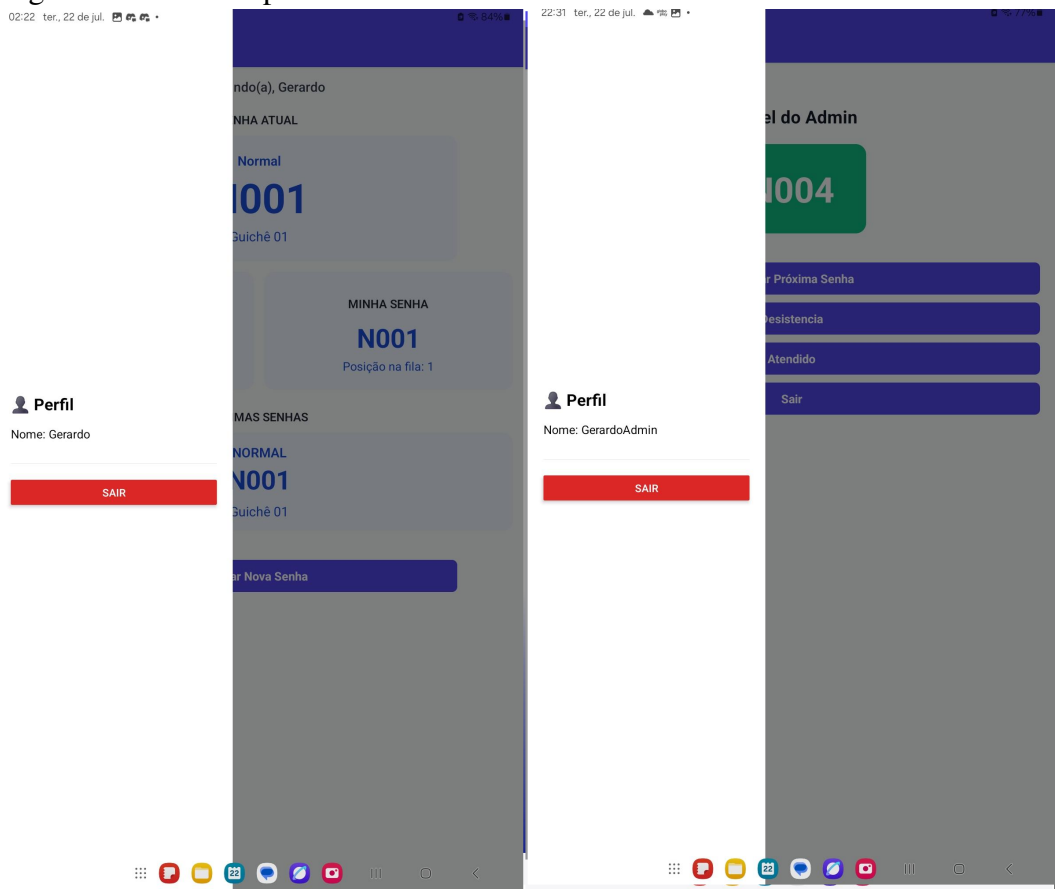
A tela de perfil implementada consta apenas o nome do usuário que está logado e o botão de sair para realizar o *logout* do sistema e retornar para a página inicial de login, como visto nas Figuras 18 e 19.

Figura 18 – Tela de perfil - iOS



Fonte: Elaborado pelo autor.

Figura 19 – Tela de perfil - Android



Fonte: Elaborado pelo autor.

5.6 Desenvolvimento

Este capítulo será apresentado como foi realizado o desenvolvimento do aplicativo Fila Digital utilizando a biblioteca *React Native* e será apresentada as configurações iniciais, como executá-lo, disposição das pastas e o método de versionamento do código.

É possível criar projetos com *React Native* de duas maneiras: utilizando o *React Native CLI* ou *Expo CLI*. Nesse trabalho, foi optado desenvolver com *Expo* devido a praticidade e facilidade de se iniciar um projeto, além de permitir o uso de um dispositivo *Android* para visualizar a evolução do desenvolvimento desde o começo.

Mas para utilizá-lo, é necessário instalar o *NodeJS*, que se encontra na sua versão 22.17.1 e que acompanha o gerenciador de pacotes *NPM (Node Package Manager)*. Após instalar o *NPM*, foi instalado o *Expo* com o comando mostrado no Código-fonte 1:

Código-fonte 1 Comando para instalar o Expo

```
1 npx create-expo-app fila-digital --template expo-template-
```

```
  blank-typescript
```

Dessa forma, é criado um novo projeto com um *template* pré-configurado em *typescript* para dar início ao desenvolvimento. Após a execução do comando, é gerada uma pasta contendo a estrutura básica do projeto. Ao entrar nessa pasta, basta instalar as dependências do projeto para poder ser iniciado, comando exibido no código-fonte 2.

Código-fonte 2 Comando para executar o Expo

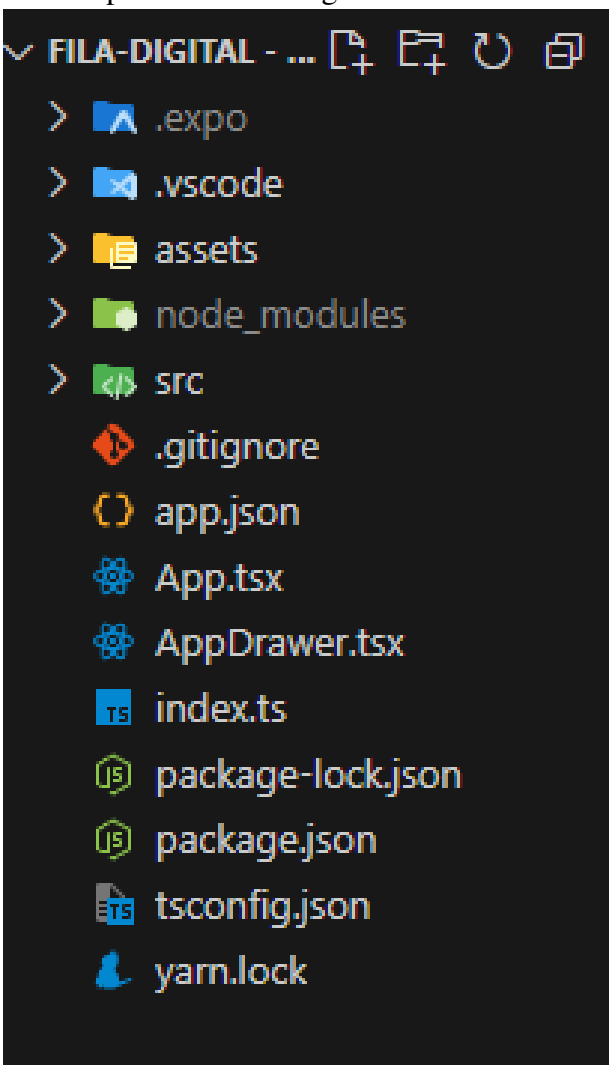
```
1 npx expo start
```

Figura 20 – Resultado do comando

Fonte: Elaborado pelo autor.

Na Figura 21 é possível observar as estruturas das pastas que o aplicativo possui. Nela estão a pasta de *assets*, contendo imagens utilizadas no aplicativo, como logo e umas imagens que são carregadas durante a *build* do projeto, e a pasta *sounds* onde possui o som de notificação de chamada de senha.

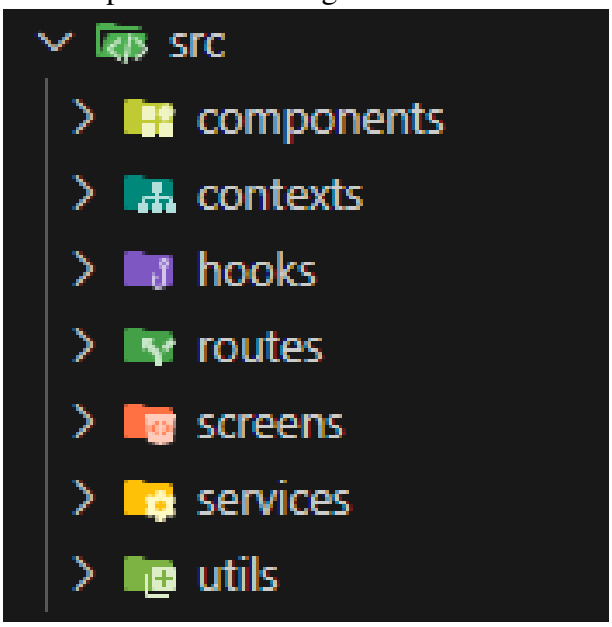
Figura 21 – Estrutura de pastas do Fila Digital



Fonte: Elaborado pelo autor.

A Figura 22 apresenta a pasta *src*, a principal pasta do projeto. Ela possui 7 pastas: a *components* onde carrega os componentes básicos da aplicação; a pasta *contexts* que possui configuração da lógica do aplicativo, como *login*, cadastrar; a pasta *hooks* onde possui um *hook* personalizado para realizar a chamada de senha de forma automática, utilizado no período de testes da aplicação; a pasta *routes* que possui a lógica de redirecionamento entre as páginas do Fila Digital; a pasta *screens* onde se encontram todas as páginas da aplicação; a pasta *services* que foi onde os serviços do *backend* foram organizados; e por fim a pasta *utils* onde serão armazenados informações em comum entre a aplicação.

Figura 22 – Estrutura de pastas do Fila Digital



Fonte: Elaborado pelo autor.

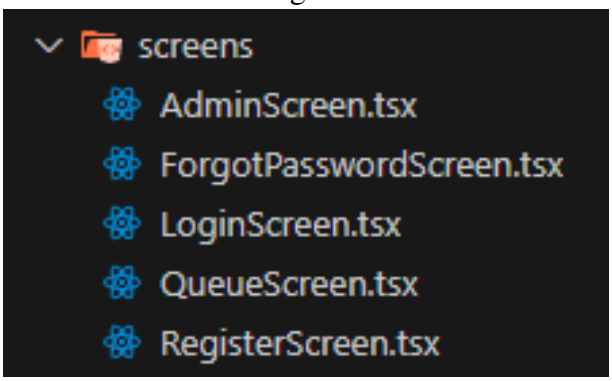
5.6.1 Telas do aplicativo

Como pode ser visto na Figura 23, a pasta *screens* contém telas do aplicativo Fila Digital contendo os seguintes arquivos:

- ***AdminScreen*:** Esta é a tela onde o usuário administrador será redirecionado após o login com sucesso. Nela, contem as principais funcionalidades para realizar o controle da fila, podendo realizar a chamada da próxima senha da fila como também indicar se a senha chamada está sendo atendido ou se houve desistência.
- ***ForgotPasswordScreen*:** Esta página foi criada com o intuito de realizar o recurso de resetar a senha em casa de esquecimento dos dados feitos no cadastro. Porém, por estar utilizando o *backend* de outro projeto externo, não foi possível realizar essa funcionalidade devido a ausência dessa logica nos serviços disponibilizados, servindo assim para uma melhoria futura do projeto.
- **LoginScreen:** Esta tela é a primeira a ser carregada ao abrir o aplicativo. Ela possui a logomarca da aplicação logo acima do pequeno formulário de login, contendo o usuário e a senha a serem preenchidos com as informações criadas no cadastro.
- ***QueueScreen*:** Essa tela é onde fica organizado as funcionalidades do usuário paciente. Nela, possui as informações sobre a senha que está sendo chamada atualmente, juntamente com as ultimas senhas chamadas e, caso retire uma senha, é possível visualizar sua senha, a posição na fila e também o tempo médio de espera.

– ***RegisterScreen*:** Por fim, esta página contem informações sobre o formulário de cadastro para que seja possível criar a conta de usuário e ser utilizada na pagina de *login*.

Figura 23 – Arquivo de telas do Fila Digital



Fonte: Elaborado pelo autor.

5.7 Configurações e dependencias

O arquivo package.json tem o propósito de listar os pacotes dependentes do projeto que foram utilizados, especificar as versões destes pacotes que o projeto utiliza, além de tornar a compilação reproduzível e fácil de compartilhar com outros desenvolvedores.

O código fonte 3 mostra os pacotes que devem ser utilizados para executar o aplicativo durante o desenvolvimento. Estes pacotes são adicionados a uma pasta de dependências chamada *node modules* à medida em que são instalados. Para realizar a instalação das dependências é necessário utilizar o gerenciamento de pacotes informado nesse trabalho, NPM, com o comando: *npm install*.

Código-fonte 3 Lista de dependencias do projeto

```
{
  "name": "fila-digital",
  "version": "1.0.0",
  "main": "index.ts",
  "scripts": {
    "start": "expo start",
    "android": "expo run:android",
    "ios": "expo run:ios",
    "web": "expo start --web"
  },
```

```
  "dependencies": {
    "@expo/webpack-config": "^19.0.1",
    "@hookform/resolvers": "^5.0.1",
    "@react-native-async-storage/async-storage": "2.1.2",
    "@react-navigation/drawer": "^7.3.12",
    "@react-navigation/native": "^7.1.6",
    "@react-navigation/stack": "^7.2.10",
    "@stomp/stompjs": "^7.1.1",
    "axios": "^1.10.0",
    "expo": "^53.0.0",
    "expo-av": "~15.1.7",
    "expo-font": "~13.3.2",
    "expo-haptics": "~14.1.4",
    "expo-image": "~2.3.2",
    "expo-linear-gradient": "~14.1.5",
    "expo-linking": "~7.1.7",
    "expo-router": "~5.1.3",
    "expo-splash-screen": "~0.30.10",
    "expo-status-bar": "~2.2.3",
    "expo-system-ui": "~5.0.10",
    "expo-web-browser": "~14.2.0",
    "react": "19.0.0",
    "react-dom": "19.0.0",
    "react-hook-form": "^7.56.1",
    "react-native": "0.79.5",
    "react-native-gesture-handler": "~2.24.0",
    "react-native-reanimated": "~3.17.4",
    "react-native-safe-area-context": "5.4.0",
    "react-native-screens": "~4.11.1",
    "react-native-web": "^0.20.0",
    "sockjs-client": "^1.6.1",
    "yup": "^1.6.1",
```

```
    "zustand": "^5.0.3"
  },
  "devDependencies": {
    "@babel/core": "^7.26.0",
    "@types/react": "~19.0.10",
    "@types/sockjs-client": "^1.5.4",
    "typescript": "~5.8.3"
  },
  "private": true
}
```

Outro arquivo importante e que já é gerado de forma padrão, na raíz do projeto, durante a criação do projeto *Expo* é o *App.tsx*. O código-fonte 4 apresenta o conteúdo do arquivo. Ele é o primeiro a ser carregado quando o projeto é executado e a partir dele todos os outros componentes e telas do aplicativo Fila Digital são acessados.

Código-fonte 4 Arquivo App.tsx

```
import { StatusBar } from "expo-status-bar";
import { AuthProvider } from "./src/contexts/AuthContext";
import { QueueProvider } from "./src/contexts/QueueContext
  ";
import AppRoutes from "./src/routes/AppRoutes";
import { SafeAreaProvider } from "react-native-safe-area-
  context";
import "react-native-gesture-handler";

export default function App() {
  return (
    <SafeAreaProvider>
      <AuthProvider>
        <QueueProvider>
          <StatusBar style="auto" />
```

```
14            <AppRoutes />
15          </QueueProvider>
16        </AuthProvider>
17      </SafeAreaProvider>
18    );
19  }
```

Outro arquivo importante é o *AppRoute.tsx* mostrado no Código-fonte 5, onde é criada a lógica de redirecionamento entre as páginas de acordo com a ação do botão para a mudança de página. Nesse arquivo, é realizada uma pequena validação no usuário para saber se o mesmo está logado, e caso não esteja, ele é redirecionado para a tela de *login*.

Código-fonte 5 Arquivo *AppRoutes.tsx*

```
1  import { NavigationContainer } from "@react-navigation/
       native";
2  import {
3    createStackNavigator,
4    TransitionPresets,
5  } from "@react-navigation/stack";
6  import { useAuth } from "../contexts/AuthContext";
7  import LoginScreen from "../screens/LoginScreen";
8  import RegisterScreen from "../screens/RegisterScreen";
9  import ForgotPasswordScreen from "../screens/
       ForgotPasswordScreen";
10 import AppDrawer from "../../AppDrawer";
11
12 const Stack = createStackNavigator();
13
14 export default function AppRoutes() {
15   const { user } = useAuth();
16
17   return (
```

```
18      <NavigationContainer>
19        <Stack.Navigator
20          screenOptions={{
21            headerShown: false,
22            ...TransitionPresets.SlideFromRightIOS,
23          }}
24        >
25          {user ? (
26            <Stack.Screen name="AppDrawer" component={
                 AppDrawer} />
27          ) : (
28            <>
29              <Stack.Screen name="Login" component={
                   LoginScreen} />
30              <Stack.Screen name="Cadastro" component={
                   RegisterScreen} />
31              <Stack.Screen
32                name="EsqueciSenha"
33                component={ForgotPasswordScreen}
34              />
35            </>
36          )}
37        </Stack.Navigator>
38      </NavigationContainer>
39    );
40  }
```

Um ponto importante que vale destacar é sobre a utilização do *backend* nesse projeto. Foi realizada uma pesquisa até que se encontrasse um *backend* que se adaptasse ao projeto realizado. Com isso, esse *backend* possui as seguintes responsabilidades que podem ser utilizadas no desenvolvimento o Fila Digital:

– ***REST Controllers*:** fornecem os pontos de entrada para o frontend, permitindo operações

como retirada de senha, visualização do estado da fila e comandos do atendente.

- ***WebSocket via STOMP*:** utilizado para notificações em tempo real aos clientes conectados, sem necessidade de *polling*.
- **Services:** encapsulam a lógica de negócio da aplicação, como regras de fila e prioridade.
- ***Repositórios JPA*:** abstraem o acesso aos dados persistidos no banco de dados.
- ***Apache Kafka*:** utilizado como mecanismo de mensageria assíncrona entre serviços e para atualização de estado da fila.

5.8 Versionamento de código

Para o armazenamento do código e controle de versão foi utilizado o *Github*. Essa ferramenta garante de forma fácil revisar o código e observar o histórico de mudanças. Caso tenha algum erro durante o desenvolvimento que impeça a execução, é possível recuperar a versão anterior funcional do projeto, que está mais estável.

A cada implementação de uma nova funcionalidade, foram realizados *commits* para que fosse criado um controle de versão mais eficiente. Dessa forma, cria-se uma quantidade segura de versões estáveis caso seja preciso fazer algum retorno de versão.

6 CONCLUSÕES E TRABALHOS FUTUROS

Este trabalho mostrou o processo de criação do aplicativo, desde a sua concepção até o seu desenvolvimento, destacando as características *cross-platform* da biblioteca *React Native* com *Expo*. Foi observado que essas ferramentas escolhidas para a criação do projeto agilizou o desenvolvimento do aplicativo, ganhando tempo no desenvolvimento.

Algumas das dificuldades encontradas nesse período de criação do Fila Digital serviram para entender os tipos de vantagens que uma biblioteca multiplataforma tem a oferecer, principalmente por problemas encontrados em manter o funcionamento em dispositivos *iOS*. Porém graças ao *Expo*, a aceleração do desenvolvimento permitiu que fosse achada uma solução do projeto. Por não possuir uma equipe de desenvolvimento, e o conhecimento prévio em desenvolvimento *mobile* não ser tão relevante, foram buscadas soluções simples e práticas que exigiam uma menor curva de aprendizado, fazendo-se necessário a utilização de mais bibliotecas além do *React Native* para manter o funcionamento, o que pode ter ocasionado uma queda no desempenho e aumento de tamanho em disco do arquivo.

Como trabalhos futuros, a expectativa é de aprimorar ainda mais o aplicativo criando um *backend* próprio para a aplicação, além de adicionar mais recursos úteis para os usuários de modo que facilite ainda mais a usabilidade e gere interesse real em utilizá-lo. Pretende-se criar uma funcionalidade de chat para entrar em contato com algum atendente para possíveis tira-dúvidas, fazer uma avaliação de usabilidade com os usuários, seja paciente ou administrador, realizar refatorações de código e melhorias de interface e, por fim, gerar o *build* para dispositivos *iOS* e obter uma maior gama de variedade de usuários em outros sistemas operacionais.

Com isso, conclui-se que o Fila Digital tem tudo para melhorar o gerenciamento de atendimento e filas em clínicas médicas onde ainda operam com filas físicas e processos manuais, trazendo agilidade para o fluxo de pessoas no estabelecimento.

REFERÊNCIAS

BARBOSA, D. V. S.; RODRIGUES, J. A.; LIMA, P. H. d. O. Tempo de espera e absenteísmo na atenção especializada na região metropolitana do espírito santo, brasil. **Saúde em Debate**, v. 43, n. spe5, p. 190–204, 2019. Disponível em: https://www.scielosp.org/article/sdeb/2019.v43nspe5/190-204/. Acesso em: 17 jul. 2025.

BEZERRA, F. S. R. **Desenvolvimento Nativo vs Ionic vs React Native**: uma análise comparativa do suporte à acessibilidade em android. Trabalho de Conclusão de Curso (Graduação em Computação) – Universidade Federal do Ceará, Campus de Fortaleza, Fortaleza, 2022. Disponível em: https://repositorio.ufc.br/bitstream/riufc/58979/1/2021_tcc_fsrbezerra.pdf. Acesso em: 22 jul. 2025.

BOMFIM, A. d. S.; AMARAL, T. M.; RAMOS, R. A. Desenvolvimento de app para monitoramento de filas de um instituto oftalmológico. **Research, Society and Development**, v. 11, n. 16, p. e272111622583, 2022. Disponível em: https://rsdjournal.org/index.php/rsd/article/view/22583. Acesso em: 22 jul. 2025.

DABIT, N. **React Native in Action**. [*S. l.*]: Manning Publications, 2021. ISBN 9781617294051.

EISENMAN, B. **Learning react native**: Building native mobile apps with javascript. [*S. l.*]: "O'Reilly Media, Inc.", 2015.

Expo Documentation. **Documentação oficial do Expo**. 2025. Disponível em: https://docs.expo.dev/. Acesso em: 22 jul. 2025.

FALCÃO, F. D. Trabalho de Conclusão de Curso (Graduação), **Desenvolvimento do aplicativo turistando beberibe utilizando React Native**. Fortaleza, CE, Brasil: [*S. n.*], 2022. Disponível em: https://repositorio.ufc.br/handle/riufc/69029. Acesso em: 22 jul. 2025.

GAEA. 2022. Disponível em: https://gaea.com.br/desenvolvimento-mobile/. Acesso em: 11 julho 2022.

GARG, S.; BALIYAN, N. Comparative analysis of Android and iOS from security viewpoint. **Computer Science Review**, v. 40, p. 100372, maio 2021. ISSN 15740137. Disponível em: https://linkinghub.elsevier.com/retrieve/pii/S1574013721000125. Acesso em: 21 jul. 2025.

GIL, A. C. **Como Elaborar Projetos de Pesquisa**. São Paulo: Atlas, 1991.

GLOBALSTAT. 2022. Disponível em: https://gs.statcounter.com/os-market-share/mobile/worldwide. Acesso em: 11 julho 2022.

GOOGLEINC. 2022. Disponível em: https://developer.android.com/guide/platform?hl=pt-br. Acesso em: 11 julho 2022.

LEE, W.-M. **Beginning android 4 application Development**. [*S. l.*]: John Wiley & Sons, 2012.

MAQE. **The Design Thinking Process**: How does it work? 2022. Disponível em: https://www.maqe.com/insight/the-design-thinking-process-how-does-it-work/. Acesso em: 22 jul. 2025.

MDN Web Docs. **WebSockets - Real-time communication**. 2023. Disponível em: https://developer.mozilla.org/en-US/docs/Web/API/WebSocket. Acesso em: 17 jul. 2025.

MENDONÇA, V. R. L. de; BITTAR, T. J.; DIAS, M. de S. Um estudo dos sistemas operacionais android e ios para o desenvolvimento de aplicativos. In: **Anais do Encontro Nacional de Computação dos Institutos Federais – ENACOMP 2011**. [*S. n.*], 2011. Disponível em: https://www.enacomp.com.br/2011/anais/trabalhos-aprovados/pdf/enacomp2011_submission_54.pdf. Acesso em: 21 jul. 2025.

Meta Platforms, Inc. **React Native Documentation**. 2025. Disponível em: https://reactnative.dev/. Acesso em: 23 jul. 2025.

RIBEIRO, G.; OLIVEIRA, P.; SOUZA, V. Aplicativos móveis para a área da saúde: uma revisão sistemática da literatura. **Revista Brasileira de Informática em Saúde**, v. 17, n. 2, p. 45–62, 2021.

ROSADO, L. H.; DIAS, G. H. A metodologia design thinking nas pesquisas científicas e a pertinência de sua apropriação pela ciência da informação. **Encontros Bibli: Revista Eletrônica de Biblioteconomia e Ciência da Informação**, v. 29, n. esp, p. 1–18, 2024. Disponível em: https://www.scielo.br/j/eb/a/Tfj5cvf5YBY8sYsMdhKvJsK. Acesso em: 23 jul. 2025.

SANTOS, A. V. D. Trabalho de Conclusão de Curso (Graduação), **Waitz**: uma aplicação web para filas de espera em hospitais e clínicas médicas. Natal, RN, Brasil: [*S. n.*], 2024. Disponível em: https://repositorio.ufrn.br/items/706cfe5a-f977-40a1-b8f5-e3c4f01d9e63. Acesso em: 22 jul. 2025.

SILVA, N. R. d.; COSTA, R.; LOCKS, M. O. H.; SEBOLD, L. F. Design thinking: uma abordagem para a pesquisa e inovação na enfermagem. **Cogitare Enfermagem**, v. 28, p. e91552, 2023. Disponível em: https://www.scielo.br/j/cenf/a/gNm6vxwY4GnzhyqrpDyy64n. Acesso em: 23 jul. 2025.

SOUZA, G. E. d. **Sistema para gestão da fila de espera em pronto-atendimento pediátrico usando aplicativo móvel**. Dissertação (Dissertação (Mestrado em Engenharia Biomédica)) – Universidade Tecnológica Federal do Paraná, Curitiba, PR, Brasil, 2016. Disponível em: http://repositorio.utfpr.edu.br/jspui/handle/1/2697. Acesso em: 22 jul. 2025.