UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

JEAN MICHEL KREMER OVIEDO

APLICATIVO PARA DISPOSITIVOS MÓVEIS PARA LOCALIZAÇÃO DE SERVIÇOS GERAIS

TOLEDO

2022

JEAN MICHEL KREMER OVIEDO

APLICATIVO PARA DISPOSITIVOS MÓVEIS PARA LOCALIZAÇÃO DE SERVIÇOS GERAIS

Trabalho de conclusão de curso de graduação apresentada como requisito para obtenção do título de tecnólogo em Sistemas para Internet da Universidade Tecnológica Federal do Paraná (UTFPR).
Orientador: Prof. Dr. Roberto Milton Scheffel.

TOLEDO

2022

JEAN MICHEL KREMER OVIEDO

APLICATIVO PARA DISPOSITIVOS MÓVEIS PARA LOCALIZAÇÃO DE SERVIÇOS GERAIS

Trabalho de Conclusão de Curso de Graduação apresentado como requisito para obtenção do título de Tecnólogo em Sistemas para Internet da Universidade Tecnológica Federal do Paraná (UTFPR).

Data de aprovação: 9 de dezembro de 2022

Eduardo Pezutti Belato dos Santos
Mestre em Ciências da Computação e Matemática Computacional
Universidade Tecnológica Federal do Paraná

Ivan Luiz Salvadori
Doutor em Ciência da Computação
Universidade Tecnológica Federal do Paraná

Roberto Milton Scheffel
Doutor em Ciência da Computação
Universidade Tecnológica Federal do Paraná

TOLEDO

2022

AGRADECIMENTOS

Gostaria de agradecer primeiramente à Deus.

Agradeço também aos meus pais pelo incentivo e investimento dedicados na minha educação.

Sou grato pela confiança acreditada na minha proposta pelo meu professor Dr. Roberto Milton Scheffel, orientador do meu trabalho. Obrigado por me orientar, manter motivado e auxiliado durante todo o processo.

Gostaria de agradecer também à Universidade Tecnológica Federal do Paraná e todo seu corpo docente.

RESUMO

Oviedo. Jean M. K. APLICATIVO MOBILE PARA PRESTAÇÃO DE SERVIÇOS GERAIS. 59f. Trabalho de Conclusão de Curso – Curso de Tecnologia em Sistemas para Internet, Universidade Tecnológica Federal do Paraná. Toledo, 2022.

As tecnologias estão sendo cada vez mais essenciais no cotidiano das pessoas, e com o aumento da popularização dos dispositivos móveis nos últimos anos, a demanda por aplicações que facilitam o dia a dia cresce constantemente. As residências eventualmente necessitam de algum reparo ou reforma, e para atender essa demanda é sempre bom contratar um profissional qualificado. Entretanto, na maioria das vezes, encontrar alguém para esse serviço é bastante complicado devido a falta de uma plataforma que repasse as informações necessárias. O objetivo deste projeto é elaborar um estudo sobre as últimas tecnologias na área de aplicações mobile, no intuito de desenvolver um aplicativo contendo as funções de procura e anúncio de prestação de serviços gerais. Ao término desse projeto pretende-se obter um produto viável e funcional, contendo funcionalidades que auxiliem na busca e divulgação de serviços.

Palavras-chave: Dispositivos Móveis, Serviços

ABSTRACT

Oviedo. Jean M. K. MOBILE APPLICATION TO PROVIDE GENERAL SERVICES. 59f. Completion of course work – Course of Technology in Internet Systems, Federal Technological University of Paraná. Toledo, 2022.

Technologies are becoming increasingly essential in people's daily lives, and with the increasing popularity of mobile devices in recent years, the demand for applications that facilitate everyday life is constantly growing. Homes eventually need some repair or renovation, and to meet this demand it is always good to hire a qualified professional. However, most of the time, finding someone for this service is quite complicated due to the lack of a platform that provides the necessary information. The objective of this project is to carry out a study on the latest technologies in the area of mobile applications, in order to develop an application containing the search and advertisement functions for the provision of general services. At the end of this project, the intention is to obtain a viable and functional product, containing functionalities that help in the search and dissemination of services.

Keywords: Mobile Devices, Services

LISTA DE IMAGENS

LISTA DE TABELAS

LISTA DE DIAGRAMAS

SUMÁRIO

1 INTRODUÇÃO

O uso de dispositivos móveis cresce a cada dia. 40% dos brasileiros aumentaram o uso de seus dispositivos móveis durante o pico da pandemia. 59% acredita que vai continuar usando na mesma medida atual mesmo após o período pandêmico. 20% não ficam mais de 30 minutos longe do celular. 19% conseguem no máximo 1 hora longe do celular. É o que mostra a mais recente pesquisa especialmente encomendada pela Digital Turbine, plataforma de mídia on-device de apps pré-instalados em smartphones (DIGITAL TURBINE, 2021).

De acordo com as informações da Pesquisa Nacional por Amostra de Domicílios Contínua - Tecnologia da Informação e Comunicação (PNAD Contínua TIC, 2018) o uso de celular para acessar a internet cresceu no Brasil. Atualmente é o principal meio de acesso a rede no país, utilizado por quase todos os brasileiros.

Devido ao aumento do uso de aparelhos celulares, também houve um crescimento na demanda por aplicações que proporcionem soluções simples e práticas. Essas aplicações também são conhecidas como aplicativos móveis.

Atualmente, as pessoas levam um estilo de vida agitado, manter todas as atividades domésticas e os eletrodomésticos em ordem pode ser cansativo neste mundo acelerado. Não são todas as pessoas que possuem tempo hábil e até mesmo habilidades para realizar trocas e consertos de itens danificados em casa. Deste modo, contratar um profissional qualificado é interessante para solucionar questões específicas como por exemplo: troca de fechaduras, rede elétrica e outros pequenos reparos.

Com isso, o foco deste trabalho é desenvolver um aplicativo mobile que proporcione facilidade na busca e divulgação de prestações de serviços gerais.

1.1 Objetivos

1.1.1 Objetivos Gerais

Este trabalho tem como objetivo desenvolver um aplicativo Mobile, através do qual será possível localizar e anunciar prestações de serviço, com informações de horários, reputação, preço médio e localização dos profissionais cadastrados. Permitirá, assim, que os profissionais conectem-se de maneira fácil e prática às pessoas que precisam dos serviços destes profissionais.

1.1.2 Objetivos Específicos

Os objetivos específicos do presente trabalho são:

- Permitir acesso aos usuários no aplicativo sem necessidade de registro.
- Permitir os usuários que busquem ou divulguem algum serviço.
- Facilitar aos usuários que encontrem serviços perto da sua região.
- Realizar a ponte de contato entre o contratante e o prestador de serviço utilizando direcionamento via aplicativos de mensagem instantânea como Whatsapp ou Telegram.

2 EMBASAMENTO TEÓRICO

Neste capítulo será abordado todo o fundamento que foi utilizado como base para a realização desse projeto. Serão abordados os conceitos da plataforma Android, Framework React Native que possibilita o desenvolvimento de aplicações Mobile, plataforma Firebase que também é uma facilitadora no desenvolvimento de aplicativos e a API do Google Maps com soluções de local e mapeamento.

2.1 Plataforma Android

Android é um sistema operacional de código aberto e baseado em Linux. Foi introduzido pela primeira vez em 5 de novembro de 2007, foi originalmente desenvolvido pela Android Inc. e posteriormente adquirido pelo Google (CHAUHAN, 2020).

O Android, basicamente foi pensado como um sistema operacional móvel, mas não se limita apenas a dispositivos móveis, é utilizado em vários outros, como celulares, tablets, televisores, etc.

O Android fornece uma estrutura de aplicativos rica que nos permite criar aplicações inovadoras para dispositivos móveis. No mercado, há uma grande variedade de dispositivos de Hardware com o sistema operacional Android.

O Android Marketplace (Google PlayStore) tem poucas restrições quanto ao conteúdo ou funcionalidade de um aplicativo Android, deste modo, o desenvolvedor pode distribuir seu aplicativo pelo marketplace e também em outros canais de distribuição, como a loja de aplicativos da Amazon.

Como o Android atende a um número maior de usuários com dispositivos móveis, os aplicativos tendem a serem mais baratos, da mesma forma, mais Downloads gratuitos também estão disponíveis, comparado a outras plataformas. Isso é um incentivo, especialmente para novas empresas que lançam seus aplicativos iniciais, pois, aplicações gratuitas e/ou baratas ajudam gradativamente a popularizar a organização por trás dele e o próprio aplicativo (SPENCER, 2022).

2.2 React Native

React Native é uma biblioteca de componentes JavaScript que permite a criação de aplicativo Mobile, projetada para criar aplicativos em diversas plataformas como Android, iOS e também aplicações Web, utilizando a mesma base de código (BUDZINSKI, 2022).

O React Native foi lançado pela primeira vez pelo Facebook, um projeto do Hackathon com o intuito de solucionar um grande problema na empresa, manter duas bases de código para sua aplicação, e o React native foi uma resposta direta para esse problema.

> Antes do surgimento do React Native, desenvolver aplicações nativas simultâneas era bem mais complexo e muito mais caro, afinal, além de o desenvolvedor ter que aprender duas plataformas e linguagensdiferentes, não era possível aproveitar partes do código de uma plataforma para a outra. (ESCUDELARIO e PINHO, 2021)

Esse Framework usa JavaScript para compilar a interface do usuário do aplicativo, mas utilizando visualizações nativas do sistema operacional. O React Native usa o conceito de “Bridge”, ele permite a assíncrona comunicação entre os elementos nativo e JavaScript. Elementos JavaScript e nativos são tecnologias completamente diferentes, mas possuem a capacidade de se comunicarem (PATERSKA, 2021).

Desta forma, o React Native permite o desenvolvimento eficaz para várias plataformas ao mesmo tempo. Usar a mesma base de código para diversas plataformas traz vários benefícios como: manutenção mais fácil e barata, desenvolvimento mais rápido e também um processo de integração mais facilitada para novos desenvolvedores que ingressam no projeto.

Para esse projeto o React Native foi escolhido devido ao tempo e performance que o framework oferece. Pois é possível acompanhar em tempo real as alterações que são aplicadas no código. Além disso, é uma tecnologia bem estabelecida e consolidada, com uma grande comunidade e adeptos.

2.3 Firebase

O Firebase é uma plataforma do Google que auxilia os desenvolvedores a criar, gerenciar e expandir seus aplicativos com facilidade. O Firebase é um BaaS (Backend as a Service), ou seja, fornece serviços como: banco de dados, armazenamento, notificações, autenticação e outros serviços que estão prontos para serem integrados na aplicação (MIRANDA; SINGH, 2018).

Como os serviços são hospedados na nuvem, os desenvolvedores podem efetuar o dimensionamento sob demanda sem problemas. O Firebase é uma opção de desenvolvimento de aplicações adequada que pode auxiliar os desenvolvedores e reduzir drasticamente o tempo de lançamento no mercado para o desenvolvimento de aplicativos.

O Firebase torna a criação de aplicativos conveniente e ajuda a manter os custos baixos, ela reduz a necessidade de desenvolver o código de back- end padrão. O mesmo permite que empresas e desenvolvedores padronizem o ambiente de Back-end em uma tecnologia única e fácil de aprender.

Gerenciar ou se preocupar com a infraestrutura do servidor não é mais um problema, o Firebase vem com uma arquitetura sem servidor que exige que os usuários paguem com base nas solicitações. Como resultado, há menos preocupações relacionadas ao dimensionamento devido à melhor eficiência (BATSCHINSKI, 2022).

O Firebase oferece aos desenvolvedores uma lista abrangente de produtos para auxiliá-los no processo de desenvolvimento de aplicações. Oferece duas opções de banco de dados são o Firestore e o Realtime Database. O Firebase cobre todo o ciclo de desenvolvimento de aplicativos e a plataforma contém recursos para criar, liberar e monitorar aplicativos.

Escalar servidores não é uma tarefa fácil, o firebase vem com uma arquitetura sem servidor que exige que os usuários paguem com base nas solicitações, e não tem necessidade de se preocupar ou gerenciar a infraestrutura do servidor. Serão cobrados apenas quando o servidor está sendo utilizado devido à sua natureza sem servidor.

O Firebase oferece plano gratuito e baseado no uso para seus usuários. No entanto, não há um plano fixo com preços uniformes para os usuários escolherem ou começarem, na Figura 1 apresenta os planos.

Figura 1 – Planos Firebase

Plano	Descrição
Spark (plano gratuito)	O Plano Spark do Firebase está disponível gratuitamente, com hospedagem de 10 GB, SSL, vários sites, domínio personalizado e diversas outras ofertas. Possui alguns recursos úteis, como Firebase ML, Real-time Database, Cloud Firestore e Test Lab. Estas são algumas das opções que você pode aproveitar com o Firebase Spark Plan.
Blaze (Pague conforme o uso)	O Plano Blaze do Firebase está disponível em um modelo de preço flexível pré-pago. Ele vem com todos os recursos do Spark Plan, além de alguns extras para melhorar a experiência de desenvolvimento. Com este plano, os usuários pagam US$ 0,026/GB pela hospedagem e o mesmo por cada GB de armazenamento. Muitas das restrições do Spark Plan são levantadas com o plano Firebase Blaze. A calculadora Blaze Plan disponível no site do Firebase pode ser uma ferramenta útil para calcular os custos estimados.

Fonte: Back4app, 2022

Para utilizar o Firebase é necessário acessar o Console do Firebase para criar um novo projeto. Pode ser utilizado uma conta Google para conectar-se, ou criar uma nova conta gratuita caso não tenha, e para isso, basta clicar em “Começar”.

A escolha do Firebase para esse projeto foi devido a facilidade na hora de desenvolver a aplicação, não precisamos nos preocupar muito com o back-end por causa das suas diversas funcionalidades.

2.3.1 Cloud Firestore

O Cloud Firestore é um banco de dados de documento NoSQL que proporciona serviços como armazenamento, sincronização e consulta por meio do aplicativo em escala global (FIREBASE, 2022). Seus dados são armazenados na forma de objetos também conhecidos como documentos. Ele tem um par chave-valor e pode armazenar qualquer tipo de dados, como Strings, dados binários e até árvores JSON.

Utilizando o Cloud Firestore é possível salvar os dados na nuvem e, ao mesmo tempo, manter os usuários sincronizados em tempo real. Ele também oferece suporte para acesso offline para aplicativos móveis que é essencialmente útil.

A escolha desse banco para o projeto foi devida as informações da aplicação sempre estarem atualizadas.

2.3.2 Firebase Authentication

A autenticação do Firebase é uma ferramenta utilizada para autenticar usuários de forma segura e rápida. Oferece um fluxo claro para o método de autenticação e login. Essa solução fornece autenticação drop-in que é usada para implementar a autenticação em dispositivos móveis e sites.

A autenticação também trabalha com casos extremos, como recuperação e vinculação de contas, que podem ser sensíveis à segurança e propensos a erros. O SDK do Firebase Authentication oferece métodos para gerenciar e criar usuários que utilizam os próprios endereços de e-mail e senha para realizar login, e permite que os usuários se conectem com as contas do Google, Facebook, Twitter e Github (FIREBASE, 2022). A autenticação do Firebase também trabalha com o envio de e-mails de redefinição de senha entre outras funcionalidades.

No projeto essa função vai economizar bastante tempo pois conta com várias funcionalidades de autenticação e verificações.

2.4 Banco de Dados Baseados em Documentos

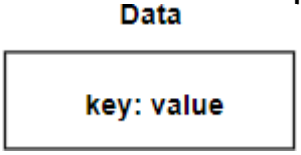
Esse banco de dados é projetado para armazenar e consultar dados como documentos do tipo JSON. A natureza semi estruturada, flexível e hierárquica dos documentos e bancos de dados de documentos proporciona à aplicação uma evolução conforme sua necessidade (AMAZON, 2022).

Esse tipo de banco de dados também é conhecido como NoSQL ou também podem ser chamados de “não relacionais” ou “não SQL” para destacar o fato de que podem acessar grandes volumes de dados em constante mudança e não estruturados de maneiras diferentes de um banco de dados relacional (SQL) com linhas e tabelas.

Os bancos de dados NoSQL geralmente são ideias para muitos aplicativos modernos, como dispositivos móveis, Web e jogos, que exigem banco de dados escaláveis, flexíveis, de alta performance e altamente funcionais.

Por utilizar NoSQL, ele trabalha com o conceito de documentos. Como mostrado na Figura 2, para representar os dados é utilizado o uso de chave/valor.

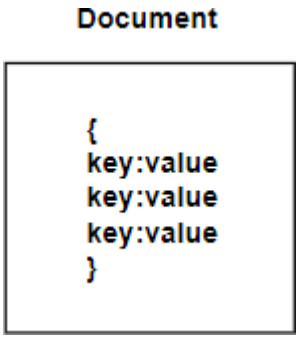
Figura 2 - Armazenar dados como um par chave/valor



Fonte: Elaborado pelo autor

Esses dados pares chave/valor podem ser agrupados, formando um documento, como representado na Figura 3.

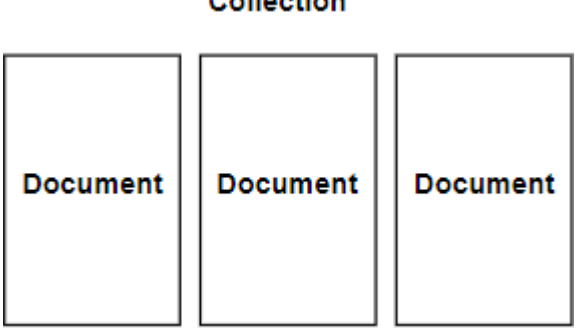
Figura 3 - Vários pares de chave/valor formam um documento



Fonte: Elaborado pelo autor

Vários documentos são chamados de coleção. Uma coleção não contém nada além de documentos. Não pode conter diretamente campos brutos com valores e não pode conter outras coleções (FIREBASE, 2022).

Figura 4 - Vários documentos formam uma coleção



Fonte: Elaborado pelo autor

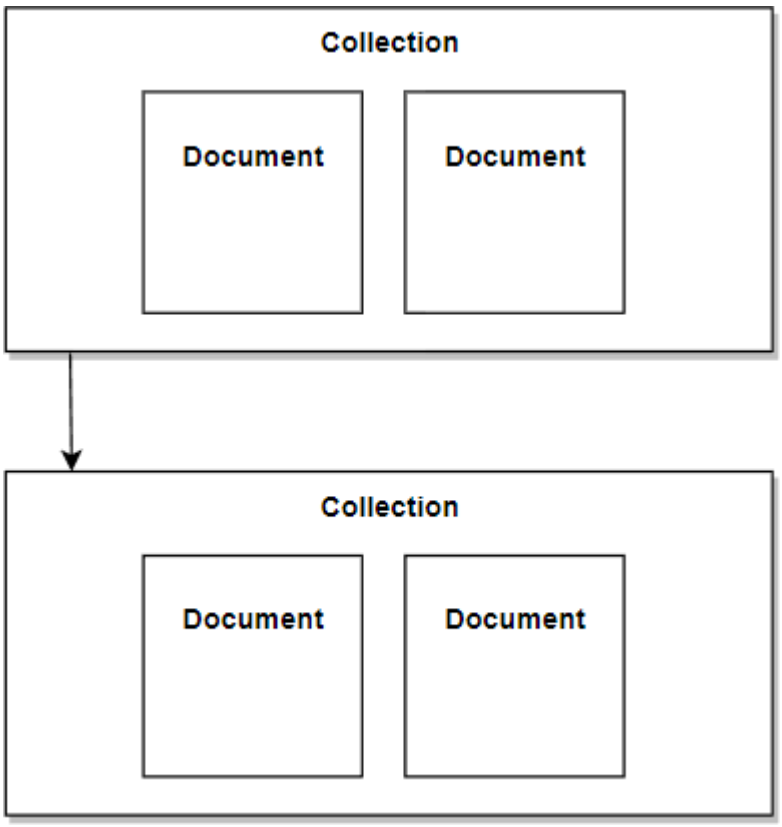
Um banco de documentos pode ter uma coleção de documentos, cada um exclusiva com sua estrutura, não tendo a necessidade de pré-definir a estrutura. A Figura 5 mostra os documentos, cada um com sua estrutura.

Figura 5 - Três documentos, cada com sua estrutura

Fonte: Elaborado pelo autor

Uma coleção de documentos não precisa ter as mesmas estruturas. Além disso, é possível adicionar campos a um documento à medida que avança, proporcionando flexibilidade máxima na maneira como os dados são armazenados. Além de armazenar os dados em um documento, o próprio documento pode referenciar outro documento ou coleção. A Figura 6 mostra isso visualmente.

Figura 6 - Um documento pode referenciar outro documento ou coleção



Fonte: Elaborado pelo autor

2.5 API Google Maps

A API do Google Maps é uma API REST (abreviação de “interface de programação de aplicativos”) que permite que os desenvolvedores acessem dados e funcionalidades do Google Maps para suas aplicações.

O Google Maps é a principal plataforma de visualização e dados de localização para a web, é um conjunto de interfaces de programação de aplicativos que nos permite conversar com seus serviços. Com isso é possível desenvolver aplicativos baseados em localização muito sofisticados para Web, iOS, Android ou qualquer outro sistema.

Figura 7 - Localização através do Google Maps

Fonte: Google Maps Platform, 2022.

A plataforma de geocodificação do Google Maps oferece diversos recursos como, coordenadas geográficas de um endereço, obter um endereço através de coordenadas de latitude/longitude e também localizar um local por meio de um endereço de ID (GOOGLE, 2022).

O preço das APIs é pago de acordo com a utilização, ou seja, você paga apenas pelo que usar, os valores dependem de cada API e são cobrados com base em solicitações. Além disso, você recebe um crédito mensal de US$ 200 para utilização das APIs, esse valor pode ser utilizado a seu critério, é equivalente a 100.000 solicitações de mapas estáticos ou cerca de 28.000 solicitações de mapas dinâmicos por mês. A cobrança do gasto mensal é somente realizada a partir do

momento em que o uso ultrapassa o crédito. Para controlar o consumo mensal das APIs, é possível definir limites de uso e evitar cobranças surpresas.

A Figura 8 representa a tabela de preço de uso da API.

Figura 8 - Tabela de preço de uso da API

	FREE MONTHLY USAGE ($200 VALUE)	PRICE PER THOUSAND CALLS	
Monthly volume range		0–100,000	100,001+
Native Static Maps	Unlimited loads	$0.00	$0.00
Native Dynamic Maps	Unlimited loads	$0.00	$0.00
Embed	Unlimited loads	$0.00	$0.00
Embed Advanced	Up to 14,000 loads	$14.00	$11.20
Static Maps	Up to 100,000 loads	$2.00	$1.60
Dynamic Maps	Up to 28,000 loads	$7.00	$5.60
Static Street View	Up to 28,000 panos	$7.00	$5.60
Dynamic Street View	Up to 14,000 panos	$14.00	$11.20

Fonte: Google Maps Platform, 2022.

Essas APIs do Google maps estão divididas em três produtos: Maps, Routes e Places.

O Maps permite aos usuários visualizar o mundo real, por meio de mapas interativos ou estáticos e que podem ser incorporados ou personalizados nos aplicativos.

O Routes auxilia os usuários encontrarem o melhor trajeto até seu destino, com a sugestão de atualizações e rotas de trânsito em tempo real, além disso, permite também criar itinerários para até 25 pontos de referência, essa funcionalidade é ideal para um planejamento de rotas de entrega mais eficiente.

O Places ajuda os usuários explorar e conhecer o mundo ao seu redor. Com vários registros de lugares, possibilitando encontrar locais específicos, usando endereços, números de telefone ou até mesmo nomes de estabelecimentos. É uma boa solução para facilitar que consumidores encontrem algum lugar especifico contendo acesso a informações detalhadas sobre endereço, nome do local, avaliações e dados de contato.

Para utilizar as funcionalidades na aplicação é necessário tem uma API KEY para a Google Maps API que pode ser feita na Google Cloud Plataform, para isso é necessário ter uma conta Google e fazer o login, em seguida criar um novo projeto.

A escolha dessa API no desenvolvimento da aplicação foi devida as diversas funcionalidades que oferece como: uma cobertura global de localização, dados enriquecidos e precisos.

3 TRABALHOS RELACIONADOS

Neste capítulo são apresentados alguns projetos relacionados já existentes no mercado que foram utilizados como referência para catalogar as principais funcionalidades da solução proposta.

3.1 GetNinjas

A GetNinjas é um software que foi desenvolvido em 2011, e que gerencia uma plataforma online que conecta clientes e prestadores de serviços por todo país. A aplicação oferece serviços das mais diversas categorias, que abrange desde serviços domésticos, reformas e reparos, designers e fotógrafos (GETNINJAS, 2022) .

O GetNinjas trabalha com duas formas diferentes. Uma atende quem está atrás de profissionais para executar um serviço e uma aos prestadores de serviços.

Os prestadores de serviços devem, previamente, efetuar um cadastro na plataforma do GetNinjas, através do aplicativo ou do site, e escolher o tipo de serviço que deseja prestar. Após o cadastro, é possível visualizar uma lista de pedidos de clientes que estão atrás de um determinado serviço de acordo com a categoria que o prestador se cadastrou.

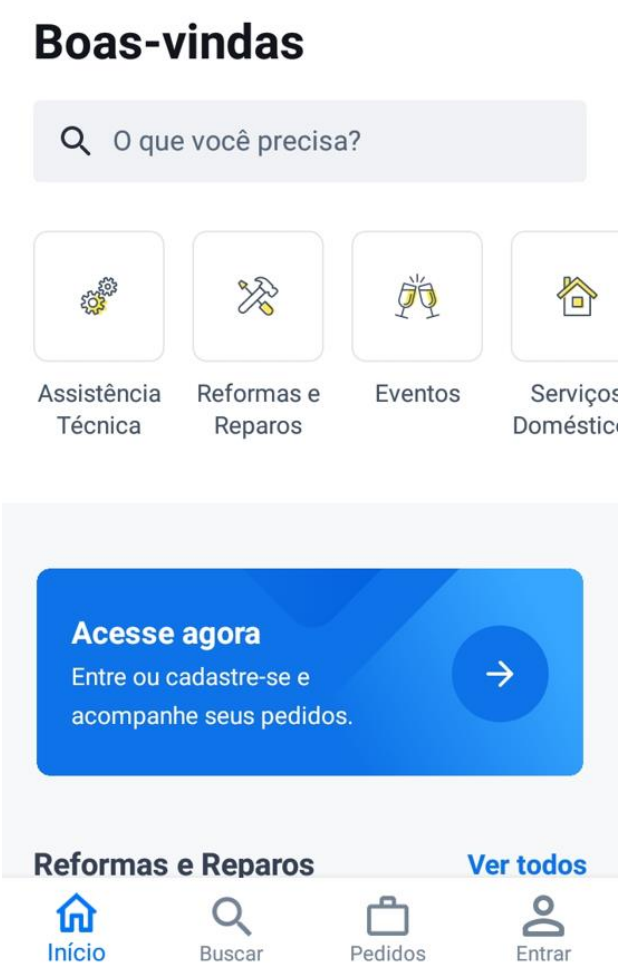
O prestador de serviço consegue verificar gratuitamente os pedidos, mas, caso tenha interesse por algum, é necessário comprar moedas GetNinjas para investir no pedido desejado.

O GetNinjas também possui uma afiliação “Indique e ganhe moedas”, na qual cada indicado que se cadastra na plataforma o afiliado é recompensado com moedas para liberar pedidos.

A figura 9 representa a tela inicial do aplicativo.

.

Figura 9 - Tela inicial



Fonte: captura de tela do aplicativo da GetNinjas para Android (acervo pessoal)

3.2 Triider

O Triider foi desenvolvido em Porto Alegre em 2016 com o intuito de ser uma plataforma para conectar clientes e profissionais de serviços domésticos das mais diferentes áreas. Atualmente a aplicação conta com mais de 50 tipos de serviços disponíveis divididos em grupos como, elétricos, hidráulicos, limpeza, ar-condicionado e assistência técnica (INTELIGÊNCIA & INOVAÇÃO, 2020).

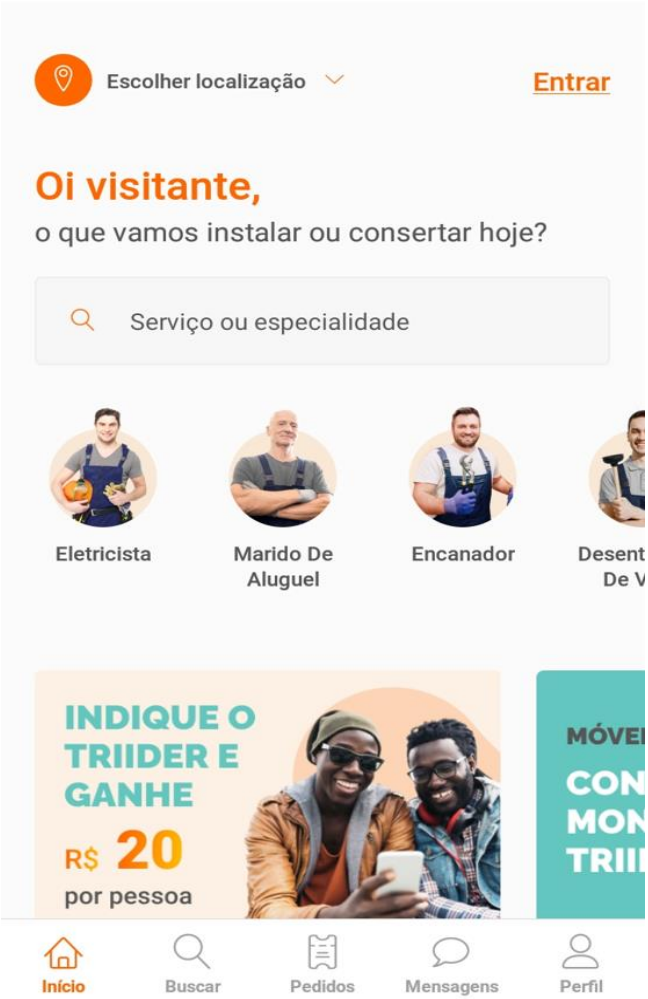
Para o profissional prestador de serviço que deseja se inscrever no app, precisa inserir seus dados, fotografar os documentos pedidos e encaminhá-los através da plataforma. Em seguida, é feita uma averiguação de seus antecedentes criminais. O algoritmo da aplicação faz checagem para buscar possíveis mandados de prisão, e se há processos relacionados à violência entre outros.

Para os prestadores de serviços, a ideia é que a plataforma disponibilize cerca de 10 a 30 pedidos de serviço por mês e a Triider cobra uma taxa de 16% por cada serviço fechado pelo app.

Para a parte do cliente, ao se cadastrar na plataforma, ele precisa completar um formulário a respeito do seu pedido de serviço a ser realizado. O Algoritmo da aplicação distribui a profissionais levando em conta a geolocalização, avaliação do prestador de serviço e se já prestou algum trabalho anteriormente.

A Figura 10 representa a tela inicial do aplicativo Triider.

Figura 10 - Pedidos disponíveis



Fonte: captura de tela do aplicativo da Triider para Android (acervo pessoal).

3.3 Considerações Finais

Este capítulo abordou a fundamentação teórica necessária para dar embasamento para o projeto. A plataforma Android, o framework React Native, a plataforma do Firebase e seus recursos, o sistema de API Google Maps.

4 METODOLOGIA

Este capítulo mostra como o projeto foi realizado, sua estrutura e aplicação, quais documentos foram gerados e as etapas do desenvolvimento. A Figura 11 representa a cronologia dos processos para a implementação da aplicação.

Figura 11 - Desenvolvimento

Fonte: Elaborado pelo autor

4.1 Etapas

As etapas para o desenvolvimento da aplicação do projeto são:

1. **Catalogação de informações**: Foram levantadas as informações relevantes para o desenvolvimento da aplicação, viabilidade e necessidades do meio em que irá funcionar.
2. **Estudar a plataforma e ferramentas de desenvolvimento (Android, React Native, API, Firebase):** Foram aprofundados os conhecimentos das tecnologias mencionadas para o desenvolvimento da aplicação.
3. **Levantamento de requisitos:** Foram levantados os requisitos funcionais e não funcionais, e quais funcionalidades a aplicação deveria atender.
4. **Modelagem:** Foram elaboradas a modelagem do sistema para o desenvolvimento.

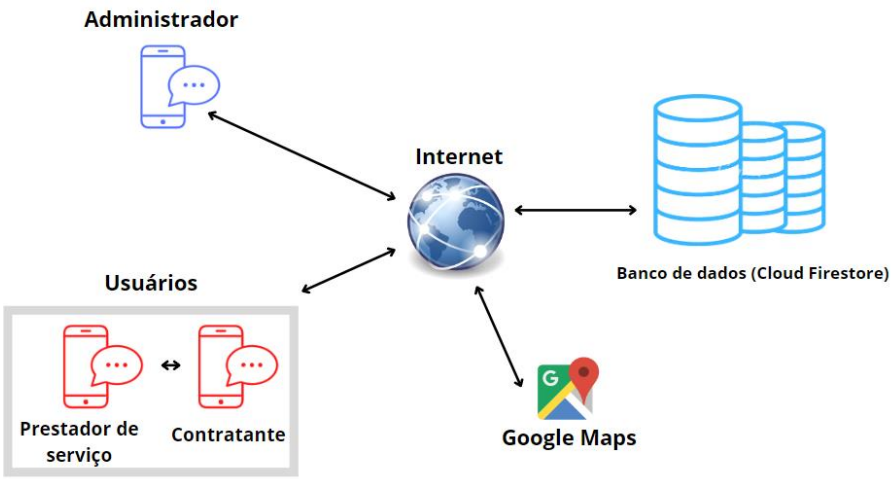
5. **Prototipação:** Foi elaborado um protótipo com as informações coletadas.
6. **Testes:** Os testes foram feitos no protótipo para correção de possíveis falhas e melhorias.

4.2 Arquitetura do Sistema

Para o desenvolvimento uma arquitetura do sistema teve que ser projetada para que entendesse os requisitos mínimos que atendesse as necessidades do mercado. Essa arquitetura é similar as diversas tecnologias necessárias que se comunicam para que o sistema entregue o que é prometido. A arquitetura do sistema é apresentada na Figura 12.

Essa arquitetura foi desenvolvida com as ferramentas para o desenvolvimento de software, sendo elas o React Native para a criação do front-end, a parte que cuida do back-end é o Firebase e Cloud Firestore e as funções de localização o Google Maps API.

Figura 12 - Arquitetura do sistema



Fonte: Elaborado pelo autor

Trata-se de uma aplicação que é composta por 3 contextos:

● Administrador: Usuário capaz de ter acesso a todas as funcionalidades do sistema

● Usuário - Contratante: Usuário capaz de se cadastrar e consultar serviços.

● Usuário – Prestador de serviço: Usuário capaz de se cadastrar, consultar e divulgar serviços.

4.3 Protótipos Inicias

A Figura 13 mostra como seria a tela de login da aplicação

Figura 13 – Tela de login

Fonte: Elaborado pelo autor

A Figura 14 mostra como seria projetado a tela de busca de serviços

Figura 14 – Tela de busca de serviços

Fonte: Elaborado pelo autor

A Figura 15 representa como seria a tela inicial da aplicação

Figura 15 – Tela inicial

Fonte: Elaborado pelo autor

A Figura 16 mostra a prototipação de como seria a tela de cadastro de serviço.

Figura 16 – Tela de cadastro de serviço

Fonte: Elaborado pelo autor

5 MODELAGEM DO SISTEMA

Neste capítulo será apresentado a modelagem da aplicação utilizando a linguagem UML (Unified Modeling Language). A aplicação será representada utilizando diversos diagramas.

5.1 Requisitos

Os requisitos expressam as características e restrições do software do ponto de vista de satisfação das necessidades do usuário e, em geral, independem da tecnologia empregada na construção da solução, sendo a parte mais crítica e propensa a erros no desenvolvimento de software.

5.2 Requisitos Funcionais

Tabela 1 - Requisitos funcionais

	Requisito	**Descrição**
RF01	Manter prestador de serviço	Cadastro de prestador de serviço. Esse cadastro será utilizado para gravar os dados do prestador. O cadastro permitirá a inclusão de dados pessoais e imagens.
RF02	Manter contratante	Cadastro de contratante. Esse cadastro permitirá gravar os dados do contratante (cadastro opcional). Também permitirá a inclusão de dados pessoais e imagens.

RF03	Inserir serviço	Cadastrar os serviços oferecidos pelos prestadores de serviço, separados por categorias prédefinidas.
RF04	Buscar serviço	Realizar a busca de serviços, considerando a localização geográfica.
RF05	Avaliar prestador de serviço	Realizar a avaliação do prestador de serviço de acordo como tempo, qualidade e preço.
RF06	Realizar direcionamento para aplicativo de mensagem instantânea	Realizar o direcionamento do cliente com o prestador de serviço para um aplicativo de mensagem instantânea.
RF07	Manter categoria	O administrador pode gerenciar as categorias, adicionar, alterar ou excluir.

Fonte: Autoria própria

5.3 Requisitos Não Funcionais

Na Tabela 2 estão os requisitos não funcionais levantados para a aplicação. Nesse quadro, RNF significa Requisito não Funcional.

Tabela 2 - Requisitos não funcionais

	Requisito	**Descrição**
RNF01	Realizar login	Para cadastrar demanda ou serviços, o usuário deverá realizar login na aplicação.
RNF02	Cadastro de serviço	Será obrigatória a seleção de categoria e descrição de serviço ao realizar um cadastro de anúncio.
RNF03	Cadastro de demanda	Será obrigatório a descrição da demanda do serviço.

RNF04	Avaliação de prestador de serviço	Para efetuar a avaliação do prestador de serviço deveráestar logado na aplicação.
RNF05	Cadastro do prestador de serviço	No cadastro do prestador de serviço há campo de preenchimento obrigatório, como o telefone para contato.

Fonte: Autoria própria

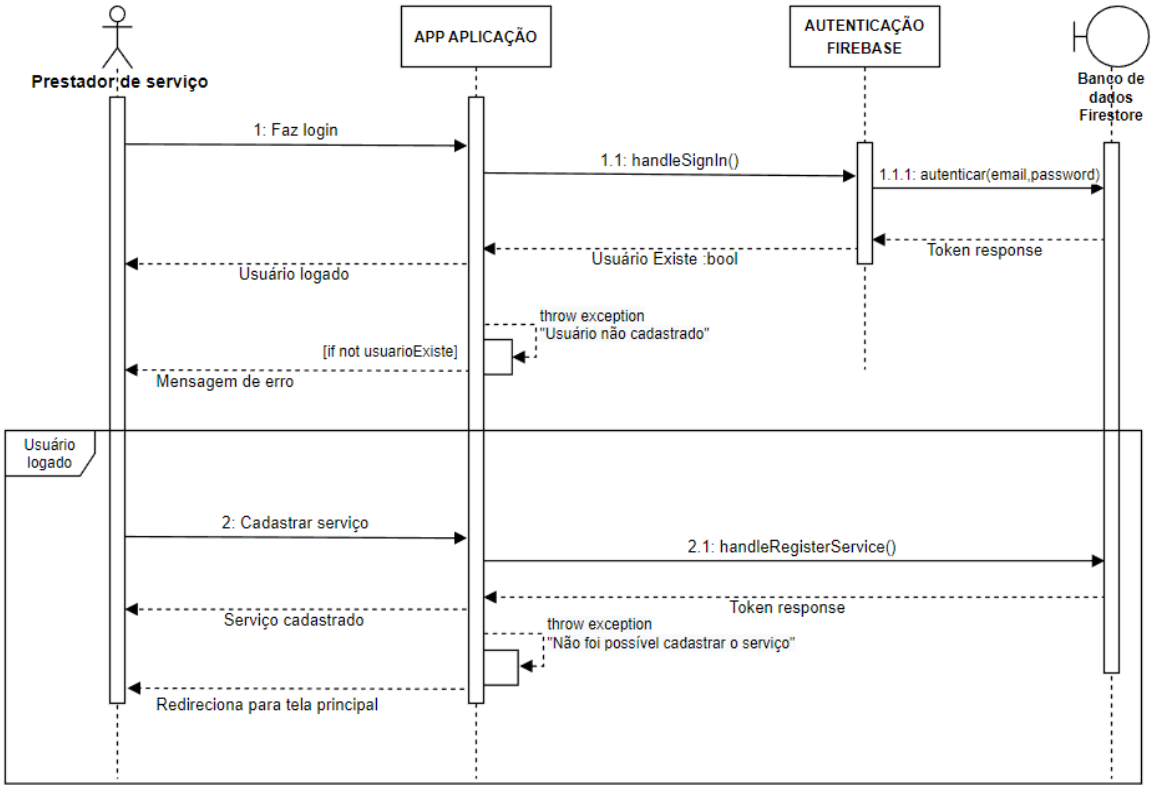
5.4 Diagrama de Casos De Uso

Diagrama 1 - Casos de uso

Fonte: Elaborado pelo autor

5.5 Diagrama De Sequência

Diagrama 2 - Sequência cadastrar serviço



Fonte: Elaborado pelo autor

No diagrama 2 acima temos a representação da sequência para um prestador de serviço cadastrar o serviço, o objetivo deste diagrama é mostrar o procedimento para realizar o cadastro de serviço na aplicação, as demais funcionalidades seguem basicamente a mesma lógica.

5.6 Armazenamento De Dados

Os dados serão armazenados na aplicação através documentos. A Tabela 3 descreve os campos e os tipos de dados do usuário que serão armazenados no Firebase Cloud Firestore.

Tabela 3 - **Descrição dos dados armazenados**

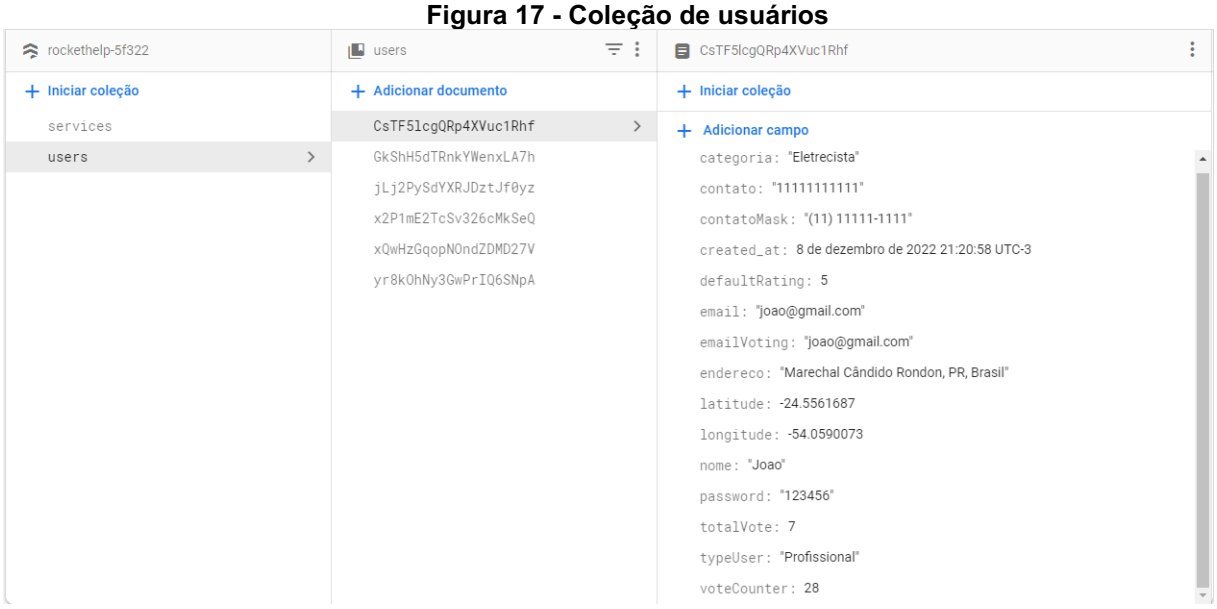
Campo	**Tipo**	**Descrição**
date	timestamp	Data do armazenamento
first_name	string	Primeiro nome
last_name	string	Sobrenome
phone	string	Número de telefone celular
email	string	E-mail do usuário
password	string	Senha do e-mail

Fonte: Elaborado pelo autor

A estrutura criada utiliza duas coleções, a de usuários (users) e serviços (services), onde a primeira guarda os dados do contratante e prestador de serviço e a segunda as informações de serviços.

A Figura 17 representa a coleção de usuários.

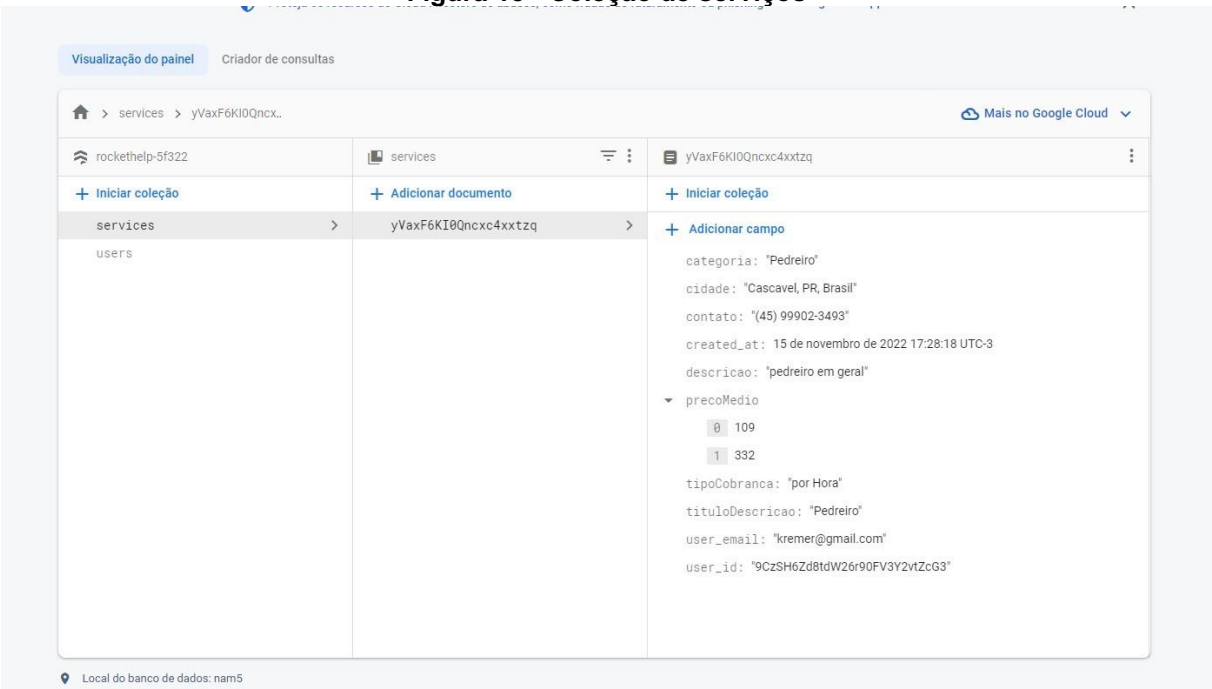
Figura 17 - Coleção de usuários



Fonte: Elaborado pelo autor

A Figura 18 representa a coleção de serviços.

Figura 18 - Coleção de serviços



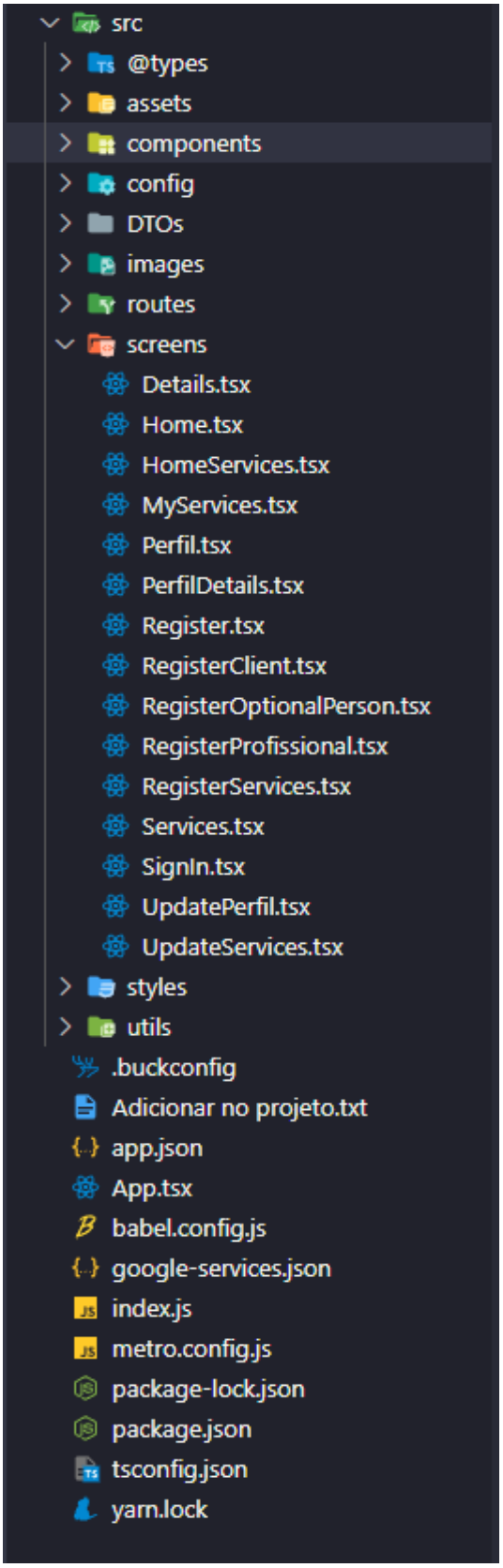
Fonte: Elaborado pelo autor

6 DESENVOLVIMENTO

6.1 Estrutura e código

A estrutura do projeto está na pasta src está dividida da seguinte maneira com as pastas @types, assets, components, config, DTOs, images, routes, screens e styles. Na pasta @types estão algumas propriedades de imagens svg, na pasta assets as logos, dentro da pasta components possui alguns elementos como botões e cards, na pasta config está o token da Google API, dentro da pasta DTOs tem algumas funcionalidades do Firebase, na pasta images possui algumas imagens de botões e ícones, dentro da pasta routes está configurado as navegações das páginas, na screens estão as telas da aplicação e dentro da pasta styles possui a configuração de cores dos temas.

Figura 19 - Estrutura do projeto



Fonte: Elaborado pelo autor

A seguir é mostrado um trecho do código que atende ao RNF01 – Realizar login, o método de login com a função handleSignIn() com email e senha que utiliza o Firebase, o método é do tipo futuro bool assíncrono que dentro da função recebe email e senha e é feita a conexão com o Firebase chamando o método signInWithEmailAndPassword do auth onde é feito o login se email e senha estiverem corretos, enquanto é feito a consulta dos dados o método setIsLoading executa como se fosse um loading até retornar a requisição. Já se ocorrer algum erro o catch é chamado e é feito o tratamento de erro, para poder mostrar ao usuário o que está acontecendo.

Figura 20 - Função de login

```
function handleSignIn(){

  if(!email || !password){
    return Alert.alert('Entrar', 'Informe e-mail e senha.');
  }

  setIsLoading(true);

  auth()
  .signInWithEmailAndPassword(email,password)
  .catch((error) =>{
    console.log(error.code);
    setIsLoading(false);

    if(error.code === 'auth/invalid-email'){
      return Alert.alert('Entrar', 'E-mail inválido');
    }

    if(error.code === 'auth/wrong-password'){
      return Alert.alert('Entrar', 'E-mail ou senha inválida');
    }

    if(error.code === 'auth/user-not-found'){
      return Alert.alert('Entrar', 'E-mail ou senha inválida');
    }

    return Alert.alert('Entrar', 'Não  foi possível acessar');
  });

}
```

Fonte: Elaborado pelo autor

No código a seguir atende ao RF03 – Inserir serviço, onde na tela de serviços quando o usuário com perfil do tipo profissional clica no botão “+” para adicionar um serviço ele é direcionado para uma tela onde irá inserir os dados, após campos preenchidos e clicar em salvar a função handleRegisterService() que é responsável por cadastrar no banco de dados Firestore Cloud, a conexão é feita com o Firebase e utiliza o método add para inserir os campos, caso não ocorra nenhum erro o método then retorna um mensagem de serviço cadastrado com sucesso, caso ocorra algum erro o método catch exibe uma mensagem que não foi possível cadastrar o serviço e direciona para tela inicial de serviços no método navigation.GoBack().

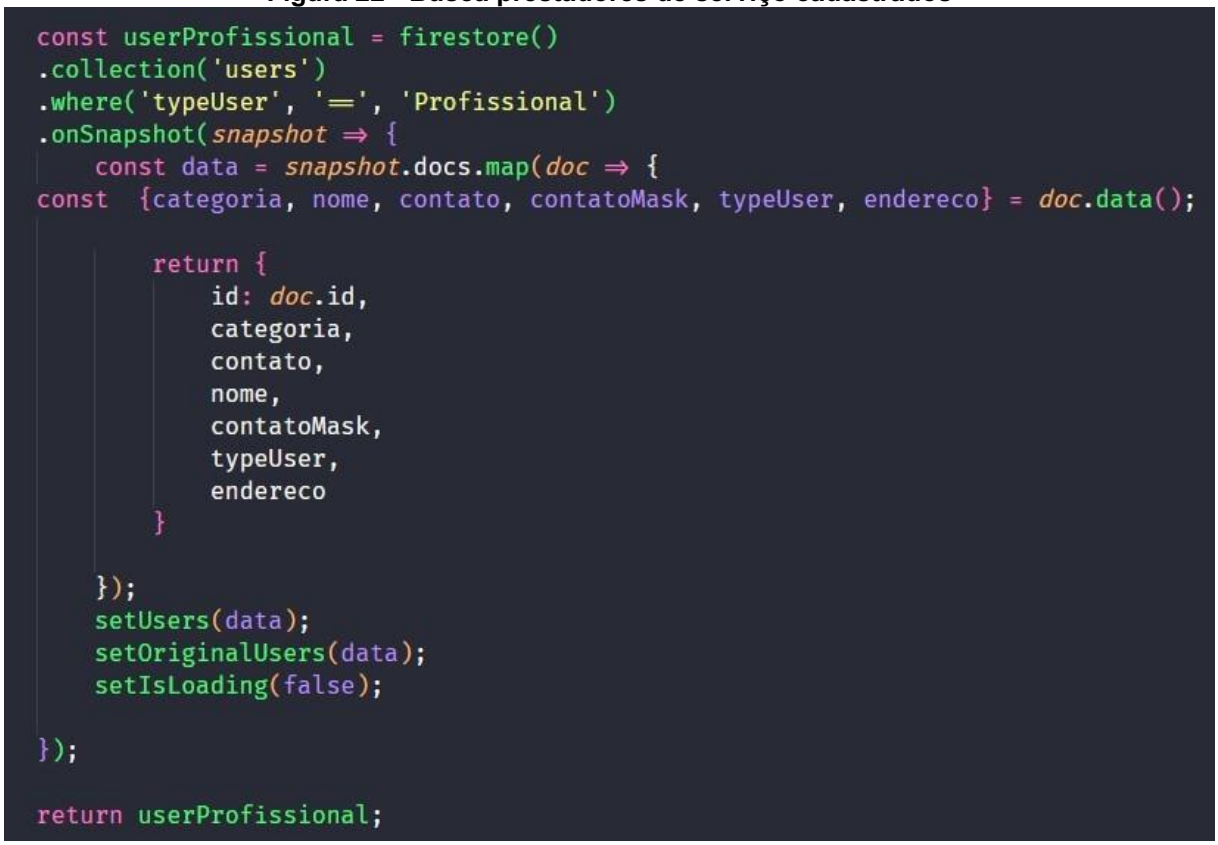
Figura 21 - Função cadastrar serviço

```
function handleRegisterService(){

    firestore()
    .collection('services')
    .add({
            user_id,
            user_email,
            tituloDescricao,
            descricao,
            categoria,
            tipoCobranca,
            precoMedio,
            created_at: firestore.FieldValue.serverTimestamp(),
            contato,
            cidade,

    })
    .then(() => {
        Alert.alert('Serviço', 'Serviço cadastrado com sucesso!')
    })
    .catch((error =>{
        console.log(error)
        return Alert.alert('Serviço', 'Não foi possível cadastrar o serviço!')
    }))

    navigation.goBack();
}
```

Fonte: Elaborado pelo autor

O trecho de código da Figura 22 é executado toda vez que o usuário se direcionar para a página inicial da aplicação. Utilizando o método where é feita uma consulta no banco de dados Firestore Cloud na coleção users e ele retorna todos os profissionais que possuem o typeUser profissional. O método onSnapshot é utilizado para manter os dados sempre atualizados em tempo real, caso ocorra alguma alteração ele atualiza imediatamente. Após feito a consulta ele salva as informações no objeto Users. Caso não tenha nenhum profissional cadastrado ele não retorna nada e aparece uma mensagem que não possui profissionais cadastrados.

Figura 22 - Busca prestadores de serviço cadastrados



```
const userProfissional = firestore()
.collection('users')
.where('typeUser', '==', 'Profissional')
.onSnapshot(snapshot => {
    const data = snapshot.docs.map(doc => {
const  {categoria, nome, contato, contatoMask, typeUser, endereco} = doc.data();

        return {
            id: doc.id,
            categoria,
            contato,
            nome,
            contatoMask,
            typeUser,
            endereco
        }

    });
    setUsers(data);
    setOriginalUsers(data);
    setIsLoading(false);

});

return userProfissional;
```

Fonte: Elaborado pelo autor

O código da Figura 23 faz a busca de todos os serviços cadastrados quando o usuário navega na página de serviços. Após conexão feita com o Firebase é feito uma consulta no banco de dados Firestore Cloud dentro da coleção services, utiliza o memso método da função anterior, o OnSnapshot que mantem os dados sempre atualizados, após a consulta realizada é armazenado na constante data e salvos no objeto setService, o objeto setOriginalService é utilizado na realização de filtros e salva os mesmos dados do setService e o setIsLoading é setado como false pois a requisição já finalizou e por final o retorno do serviço.

Figura 23 - Busca serviços cadastrados

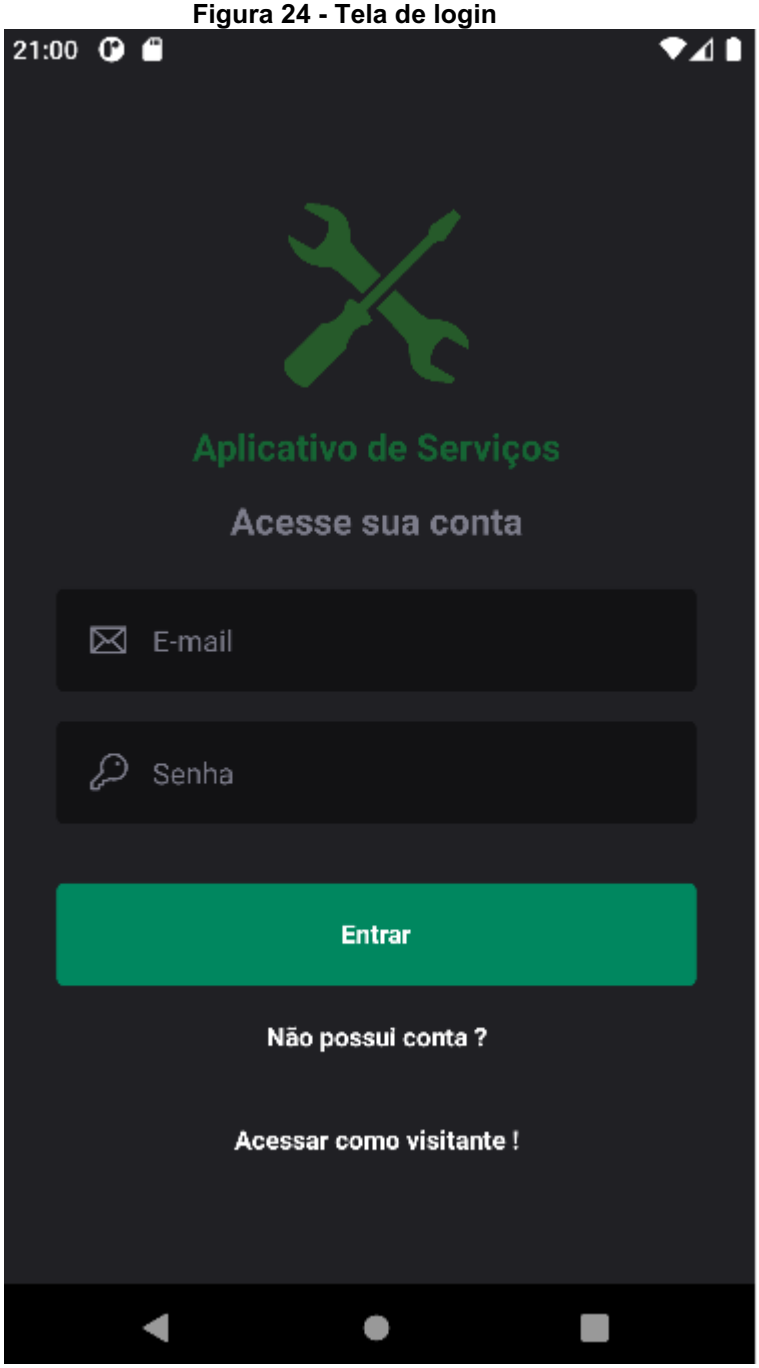
```
const service = firestore()
.collection('services')
.onSnapshot(snapshot => {
    const data = snapshot.docs.map(doc =>{
        const { descricao,
                tituloDescricao,
                precoMedio,
                categoria,
                contato,
                contatoMask,
                user_email,
                tipoCobranca,
                cidade
            } = doc.data();

        return{
            id: doc.id,
            descricao,
            tituloDescricao,
            precoMedio,
            categoria,
            contato,
            contatoMask,
            tipoCobranca,
            user_email,
            cidade
        }
    });

    setServices(data);
    setOriginalServices(data);
    setIsLoading(false);

});

return service;
```

Fonte: Elaborado pelo autor

6.2 Protótipo Final

A Figura 24, representa o RNF01 - Realizar login contendo os campos de preenchimento de e-mail e senha e três botões, um para efetuar o login, criação de nova conta e acessar como visitante.

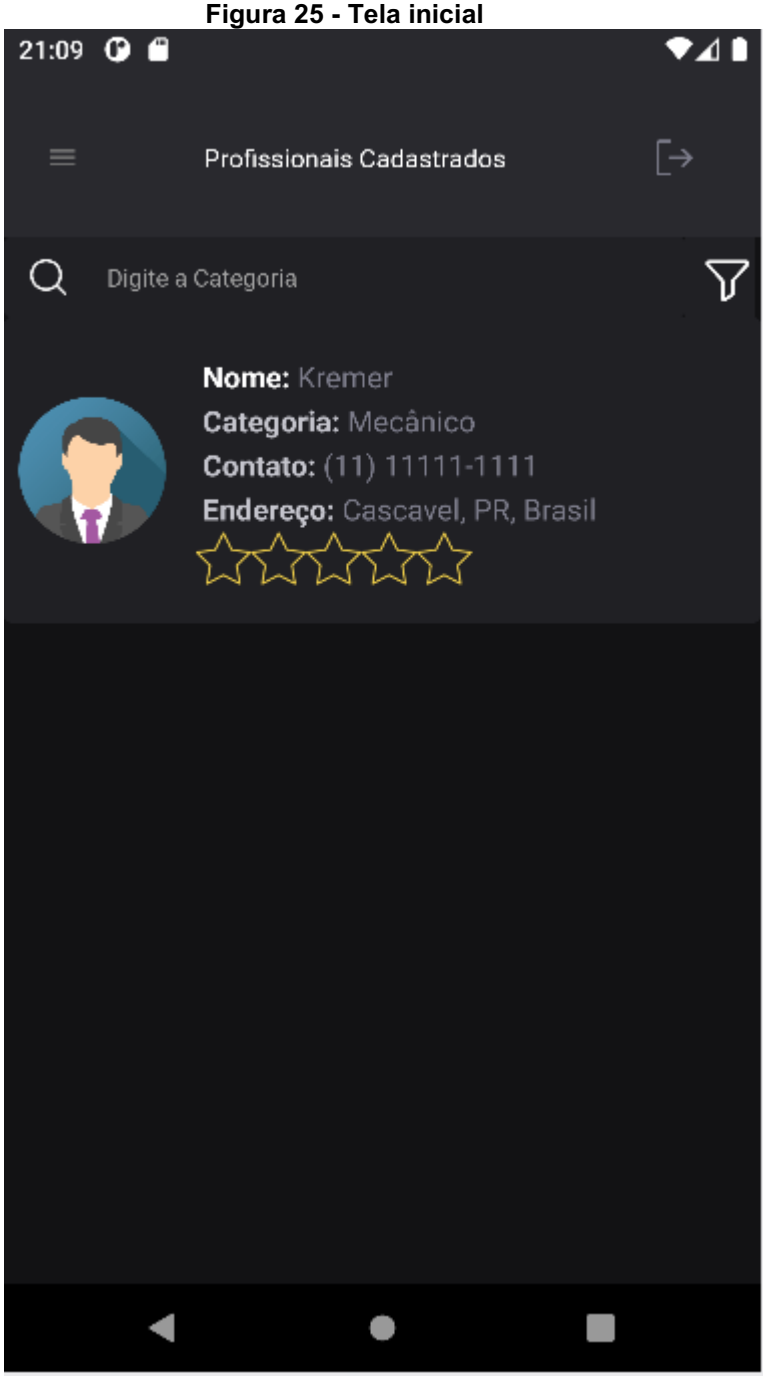
Figura 24 - Tela de login



Fonte: Elaborado pelo autor

A Figura 25, é a tela inicial da aplicação onde aparecem os profissionais cadastrados no sistema, onde é possível o usuário aplicar um filtro de busca por categoria ou cidade e podendo entrar no perfil do prestador de serviço para entrar em contato.

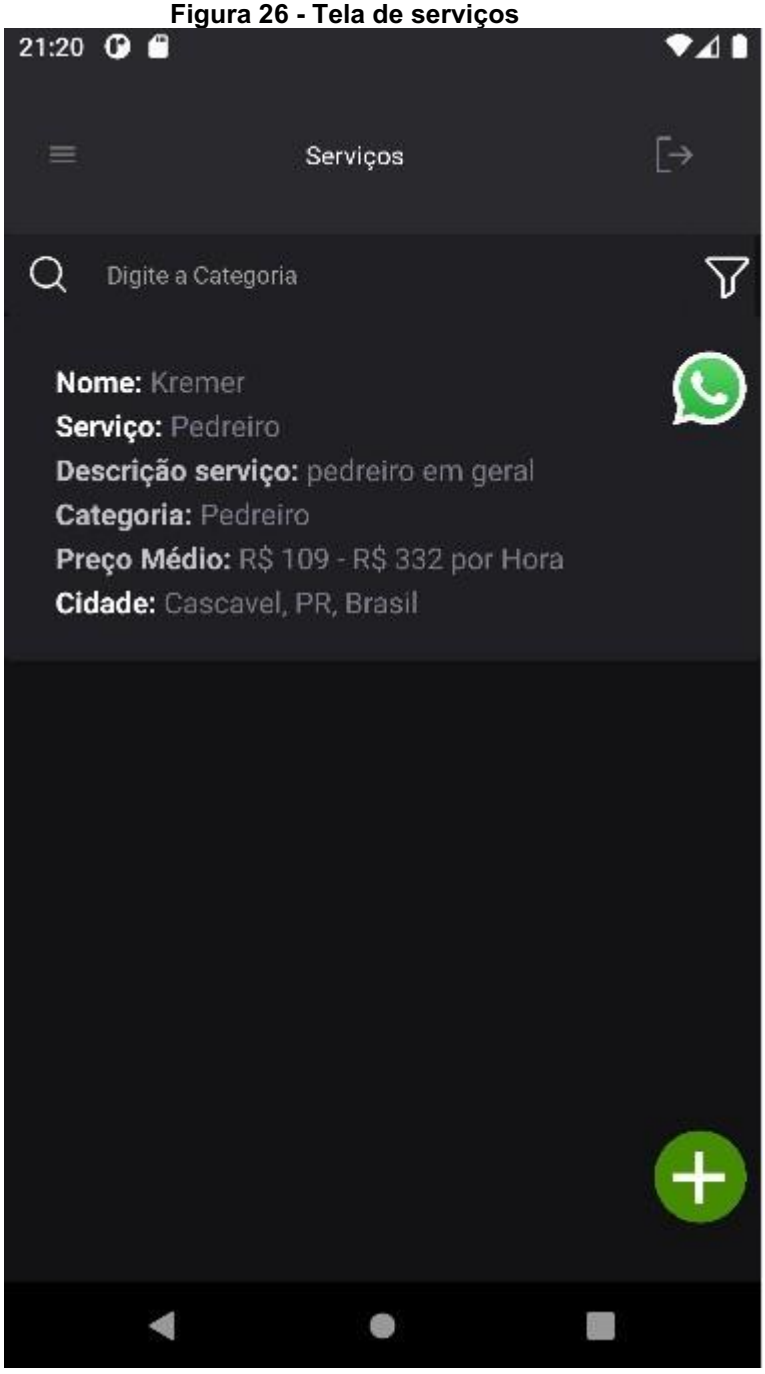
Figura 25 - Tela inicial



Fonte: Elaborado pelo autor

A Figura 26 representa a tela de serviços que atende ao RF04 – Buscar serviços cadastrados pelos prestadores de serviço com detalhes dos serviços, como categoria, preço, localização e entrar em contato pelo Whatsapp que atende ao RF06 – Realizar direcionamento para aplicativo de mensagem instantânea, nessa tela é possível realizar filtro também e se for um usuário com perfil prestador de serviço logado ele pode inserir novos serviços no botão +.

Figura 26 - Tela de serviços



Fonte: Elaborado pelo autor

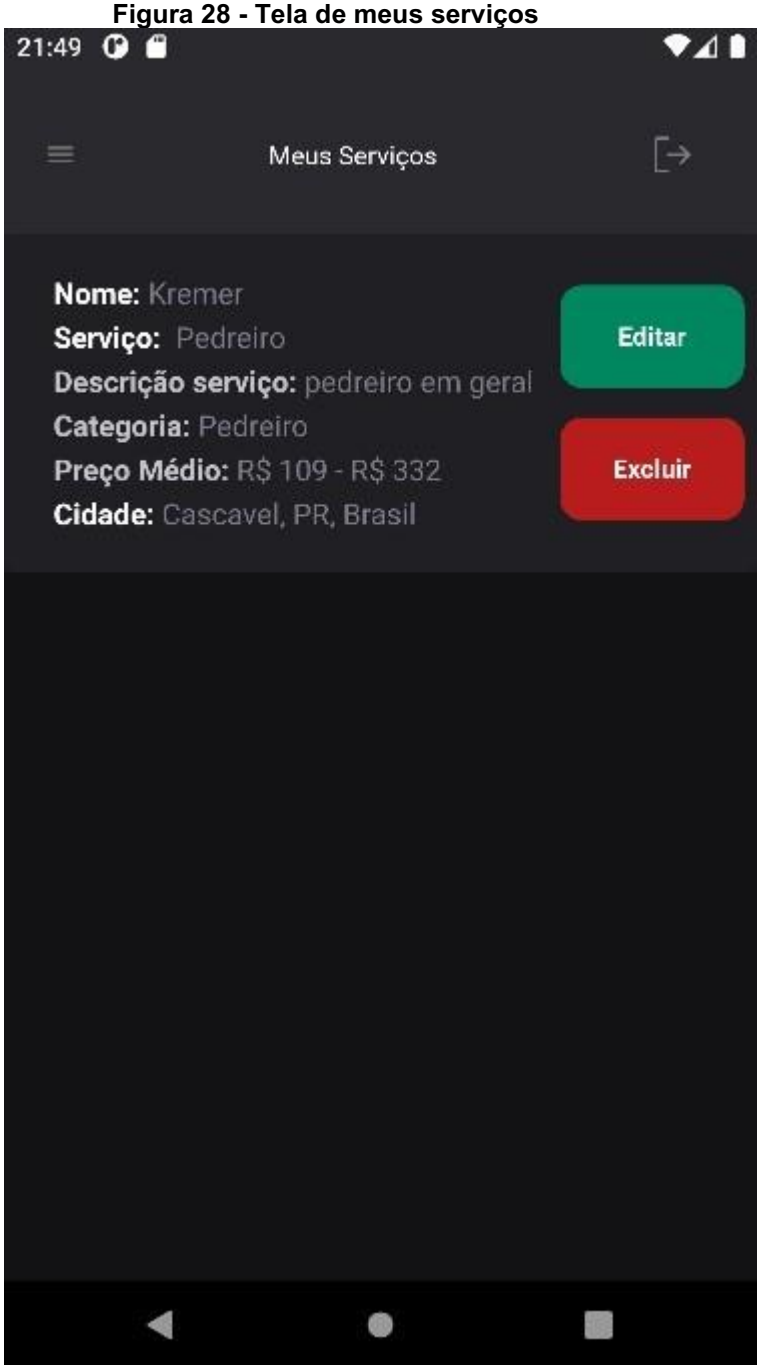
A Figura 27 representa a tela de cadastro de serviço que atende ao RF03 – Inserir serviço, é possível adicionar o título do serviço, descrição, contato, categoria, tipo de cobrança e preço médio.

Figura 27 - Tela de cadastro de serviços

Fonte: Elaborado pelo autor

A Figura 28 representa a tela de meus serviços, onde o prestador de serviço pode editar ou excluir algum serviço cadastrado.

Figura 28 - Tela de meus serviços



Fonte: Elaborado pelo autor

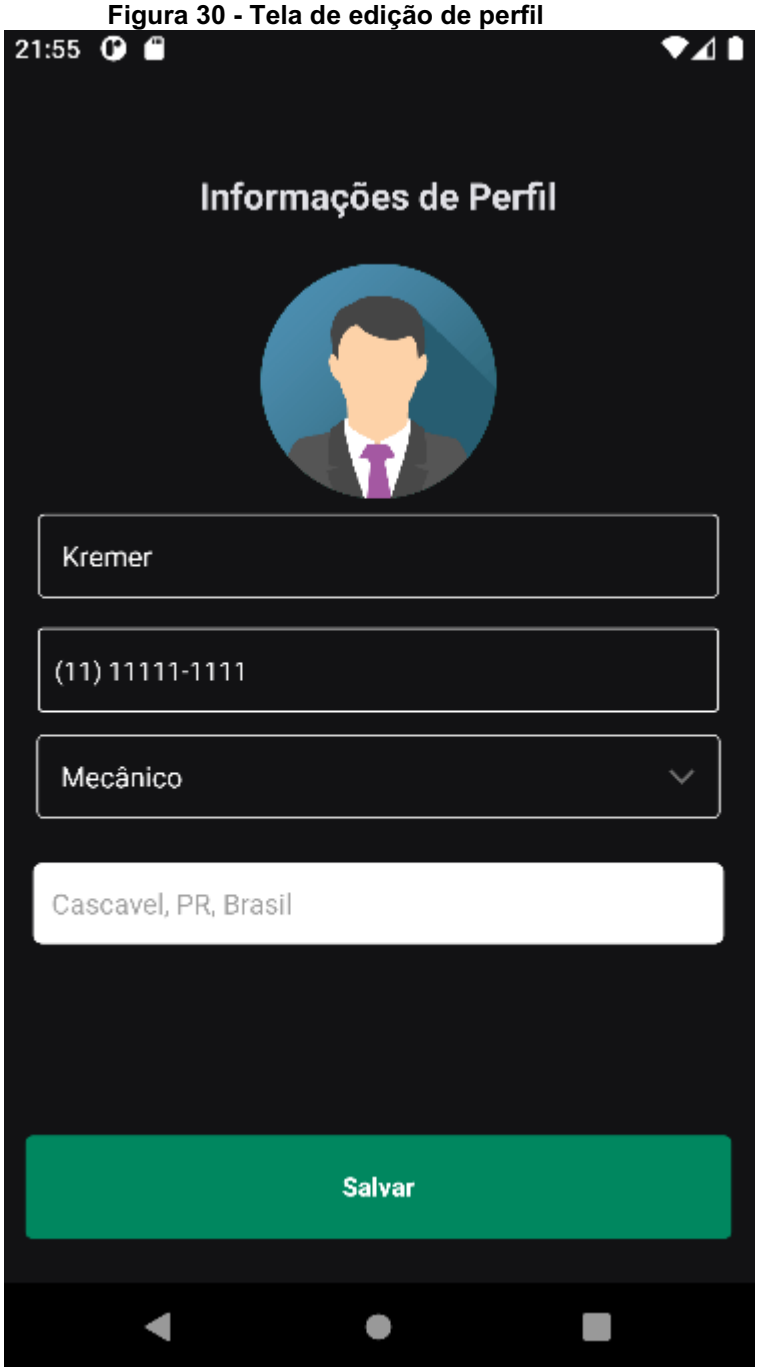
A Figura 29 representa a tela de edição de serviço. Os dados são buscados no banco de dados e o usuário escolhe as informações que deseja alterar.

Figura 29 - Tela de edição de serviço

Fonte: Elaborado pelo autor

A Figura 30 representa a tela de edição de perfil. Onde os dados são pré setados e o usuário pode alterar a informação que desejar.

Figura 30 - Tela de edição de perfil



Fonte: Elaborado pelo autor

7 RESULTADOS

O objetivo deste trabalho foi desenvolver um aplicativo para dispositivo móvel que conecte prestadores de serviços e contratantes. Visando, assim, facilitar as pessoas a encontrarem profissionais que forneçam os mais diversos tipos de serviços. O aplicativo tem como principal função agilizar essa ponte de comunicação entre ambas as partes.

Uma das funcionalidades principais da aplicação é realizar a ponte de comunicação entre o contratante e o prestador de serviço que através da divulgação do serviço o contratante possa entrar em contato pelo WhatsApp com o prestador.

A solução escolhida para o desenvolvimento do aplicativo foi o React Native que possui diversas funcionalidades que aumentam a produtividade do desenvolvimento. Para o armazenamento de dados e autenticação foi utilizado o Firebase que disponibiliza dados em tempo real, segurança, API pronta entre outras funcionalidades. A API do Google Maps também foi implementada para localização das cidades.

As telas que foram desenvolvidas são as telas de cadastros de serviços gerais, profissionais e clientes, podendo ser feito edições no cadastro dos serviços e perfis e também a exclusão de serviços. As telas foram desenvolvidas possuindo as interações e um fluxo de navegação entre elas.

8 TRABALHOS FUTUROS

Nessa aplicação ainda existe bastante espaço para melhorias. Também é vislumbrada a possibilidade de criar novos recursos que possam proporcionar uma experiência mais amigável e moderna para os usuários.

Algumas melhorias/mudanças/ são:

- Implementação de sistemas de pagamento.
- Criação de um chat no próprio aplicativo.
- Adicionar melhorias que possam aperfeiçoar e performar melhor a aplicação.
- Melhorar o layout, para o aplicativo ficar mais bonito.

9 CONSIDERAÇÕES FINAIS

Analisando a problemática de encontrar profissionais qualificados nos mais diversos serviços, o presente trabalho de conclusão de curso teve como objetivo desenvolver uma aplicação para que as pessoas tenham acesso a uma ferramenta prática e fácil de utilizar para a busca e divulgação de serviços.

De acordo com os resultados, foi possível perceber que o aplicativo oferece recursos básicos para atender as demandas de busca e divulgação de serviços. Foram utilizadas tecnologias atuais para desenvolvimento de aplicativos como React Native, Firebase e Google Maps API.

O desenvolvimento deste trabalho no ponto de vista acadêmico contribui para a formação no desenvolvimento de aplicações mobile. Além disso, o desenvolvimento dessa aplicação contribuiu para o autor adquirir novos conhecimentos e competências e reforçar habilidades adquiridas durante a graduação.

REFERÊNCIAS BIBLIOGRÁFICAS

TOKARNIA, Mariana. **Celular é o principal meio de acesso à internet no país**, 2020. Disponível em:<https://agenciabrasil.ebc.com.br/economia/noticia/2020-04/celular-e-o- principal-meio-de-acesso-internet-no-pais>. Acesso em: 08 mai. 2022.

TURBINE, Digital. **Uso do aparelho celular aumentou na pandemia**, 2020. Disponível em: <https://www.abcdoabc.com.br/brasil-mundo/noticia/uso- aparelho-celular-aumentou-pandemia-134451>. Acesso em: 08 mai. 2022.

FIX. **O que é marido de aluguel**, 2019. Disponível em: <https://fix.com.br/dicas/marido-de-aluguel-entenda-mais-sobre-o-servico/>. Acesso em: 08 mai. 2022.

CHAUHAN, Shailendra. **What is Android and Why to use it?**, 2022. Disponível em: <https://www.dotnettricks.com/learn/android/what-is-android- and-why-to-use-it>. Acesso em: 08 mai. 2022.

SPENCER, William. **Top 10 Reasons to Choose Android OS Platform For Mobile Application Develipment**, 2022. Disponível em: <https://www.hokuapps.com/blogs/10-reasons-choose-android-mobile- application-development/>. Acesso em: 08 mai. 2022.

SINGH. Vrijraj. **Introduction to Firebase**, 2022. Disponível em: <https://medium.com/codingurukul/introduction-to-firebase-f9f6ccc8a785>. Acesso em: 10 mai. 2022.

FIREBASE. **Cloud Firestore**, 2022. Disponível em: <https://firebase.google.com/docs/firestore>. Acesso em: 10 mai. 2022.

FIREBASE. **Firebase Authentication**, 2022. Disponível em: <https://firebase.google.com/docs/auth>. Acesso em: 10 mai. 2022.

BUDZINSKI. Maciej. **React Native - one framework to rule them all**, 2022. Disponível em: <https://www.netguru.com/glossary/react-native>. Acesso em: 08 mai. 2022.

PATERSKA. Patrycja. **What is React Native And When to Use It For Your App? (Updated)**, 2021. Disponível em: <https://www.elpassion.com/blog/what-is-react-native-and-when-to-use-it>. Acesso em: 08 mai. 2022.

MIRANDA. Julius. **Firebase Básico - O que é BasS ?**, 2021. Disponível em: <https://juliusmiranda.com/2016/09/13/firebase-basico-o-que-e-bass/>. Acesso em: 08 mai. 2022.

ESCUDELARIO, B.;PINHO, D. **React Native: Desenvolvimento de aplicativos mobile com React**. São Paulo: Casa do Código, 2021.

GOOGLE. **Crie apps incríveis com as informações do Google sobre o mundo real ?**, 2022. Disponível em: <https://developers.google.com/maps?hl=pt-br#top-topics>. Acesso em: 10 mai. 2022.

GOOGLE. **Geocoding API**, 2022. Disponível em: <https://developers.google.com/maps/documentation/geocoding >. Acesso em: 10 mai. 2022.

AMAZON. **O que é um banco de dados de documentos?**, 2022. Disponível em: <https://aws.amazon.com/pt/nosql/document/>. Acesso em: 13 mai. 2022.

BATSCHINSKI. George. **What is Firebase ? All the secrets unlocked**, 2022. Disponível em: <https://blog.back4app.com/firebase/>. Acesso em: 13 mai. 2022.

VISUAL PARADIGM. **What is Unified Modeling Language (UML)?** 2022. Disponível em: <https://www.visual-paradigm.com/guide/uml-unified-modeling- language/what-is-uml/>. Acesso em: 16 mai. 2022

THAKUR. Dinesh. **What is Software Requirement? Types of Requirements**. 2022. Disponível em: <https://ecomputernotes.com/software-engineering/softwarerequirement>. Acesso em: 16 mai. 2022

GORBACHENKO. **Pavel. What are Functional and Non-Functional Requirements and How to Document These**. 2022. Disponível em: <https://enkonix.com/blog/functional-requirements-vs-non-functional/>. Acesso em: 16 mai. 2022

DEVMEDIA. **Artigo Engenharia de Software 3 - Requisitos Não Funcionais**. 2008. Disponível em: <https://www.devmedia.com.br/artigo- engenharia-de-software-3-requisitos-nao-funcionais/9525>. Acesso em: 18 mai. 2022

DEVMEDIA. **O que é UML e Diagramas de Caso de Uso: Introdução Prática à UML**. 2012. Disponível em: <https://www.devmedia.com.br/o-que-e- uml-e-diagramas-de-caso-de-uso-introducao-pratica-a-uml/23408>. Acesso em: 18 mai. 2022

INTELIGÊNCIA E INOVAÇÃO. **Triider: "Uber"dos serviços domésticos chega a São Paulo**. 2020. Disponível em: <https://inteligenciaeinovacao.com/triider-uber-dos-servicos-domesticos- chega-a-sao-paulo/>. Acesso em: 23 mai. 2022

GETNINJAS. **Quem somos nós?**. 2022. Disponível em: <https://www.getninjas.com.br/sobre-nos>. Acesso em: 23 mai. 2022

MEDIUM. **Utilizando Mapas no React Native**. 2022. Disponível em: < https://medium.com/@tadeumx1/utilizando-mapas-no-react-native-817c4f4de6f7>. Acesso em: 13 dez. 2022

SUPPORT GOOGLE. **Ajuda do Firebase**. 2022. Disponível em: <https://support.google.com/firebase/answer/7000104?hl=pt-BR#zippy=%2Cneste-artigo>. Acesso em: 13 dez. 2022