

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DO  
RIO GRANDE DO SUL  
CAMPUS CANOAS

JOÃO MARCOS BUENO SILVA

**FLANELINHA - SEU VEÍCULO BEM CUIDADO: Aplicativo  
móvel para anúncio e busca de vagas de estacionamento em  
propriedades não comerciais**

CANOAS  
2022

JOÃO MARCOS BUENO SILVA

**Flanelinha - Seu veículo bem cuidado: Aplicativo móvel para anúncio e busca de vagas de estacionamento em propriedades não comerciais**

Trabalho de Conclusão de Curso apresentado como requisito parcial para obtenção do grau de Tecnólogo em Análise e Desenvolvimento de Sistemas pelo Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Sul – Campus Canoas.

Orientadora: Prof(a). Dr(a). Carla Odete Balestro Silva

Canoas

2022



**Ministério da Educação**  
**Secretaria de Educação Profissional, Científica e Tecnológica**  
**Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Sul**  
**Campus Canoas**

**ATA DE DEFESA PÚBLICA DO TRABALHO DE CONCLUSÃO DE CURSO**

Aos dois dias do mês de agosto do ano de 2022, às 10 horas, em sessão pública na sala virtual do Google Meet (<https://meet.google.com/esw-qhqb-iiu>) do Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Sul, Campus Canoas, na presença da Banca Examinadora presidida pelo(a) Professor(a):

Dra. Carla Odete Balestro Silva e composta pelos examinadores:

1. Prof. Dr. Rafael Coimbra Pinto;
2. Prof. Dr. Leonardo Filipe Batista S. de Carvalho;

o(a) aluno(a) JOÃO MARCOS BUENO SILVA apresentou o Trabalho de Conclusão de Curso intitulado: “FLANELINHA - SEU VEÍCULO BEM CUIDADO: Aplicativo móvel para anúncio e busca de vagas de estacionamento em propriedades não comerciais” como requisito curricular indispensável para a integralização do Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas. Após reunião em sessão reservada, a Banca Examinadora deliberou e decidiu pela APROVAÇÃO do referido trabalho, divulgando o resultado formalmente ao aluno e demais presentes e eu, na qualidade de Presidente da Banca, lavrei a presente ata que será assinada por mim, pelos demais examinadores e pelo aluno.

Carla Odete Balestro Silva

Rafael Coimbra Pinto

Leonardo Filipe Batista S. de Carvalho

João Marcos Bueno Silva

## **AGRADECIMENTOS**

Agradeço em primeiro lugar a Deus, que me manteve e cuidou em todas as situações que me fizeram chegar até aqui.

Aos meus pais que estiveram presente e me deram todo o suporte durante toda a minha vida, me dando apoio, conselhos e correções quando necessário.

Ao meu irmão por estar sempre disposto a conversar e ajudar no que fosse preciso, entendendo as dificuldades que existiam em cursar uma faculdade por ter passado por essa fase também.

Aos meus amigos que sempre me incentivaram a continuar, permanecer até a conclusão do curso, entendendo a ausência em diversos momentos para focar nos estudos e ter uma colheita no futuro dos frutos vindos do esforço no presente.

À minha orientadora por me auxiliar em todas as questões, dúvidas, por acreditar no meu potencial e conseguir extrair o melhor de mim.

Aos professores que fizeram parte da minha jornada acadêmica no IFRS, sempre dispostos a ensinar, mostrar o melhor caminho, partilhar experiências e me poupar de diversas dificuldades que eu certamente enfrentaria sem os conhecimentos deles.

Aos colegas que estiveram ao longo de todo o curso e foram companheiros em trabalhos, dúvidas, indagações e compartilhamento de sofrimentos.

A todos que direta ou indiretamente fizeram parte e contribuíram para a conclusão da minha formação acadêmica, muito obrigado.

## RESUMO

O presente trabalho tem como objetivo construir uma aplicação que supra a necessidade de segurança e baixo preço nas soluções atuais para estacionamento de veículos automotores. Com isso foi constatado que existe uma alternativa viável para suprir essa demanda relatada. Há a possibilidade de propriedades não comerciais disponibilizarem seus espaços com o intuito de obter um menor preço do que aquele praticado por estabelecimentos exclusivamente destinados a estacionamento de veículos e, com isso, proporcionar maior segurança às pessoas que deixam seus carros em vias públicas. Tendo isso em vista, foi desenvolvido o aplicativo móvel Flanelinha que possibilita a busca e anúncio dessas vagas alternativas aos modelos convencionais, o aplicativo é destinado aos sistemas operacionais Android e iOS por ter sido construído com a tecnologia React Native, proporcionando uma maior abrangência de dispositivos. Também foram utilizados serviços no modelo *serverless* que auxiliam no desenvolvimento de aplicações com maior possibilidade de escalabilidade e menor esforço com desenvolvimento no lado servidor do sistema.

**Palavras-chave:** React Native. Estacionamento. Android. iOS. Veículos.

## **ABSTRACT**

The present work has the purpose of build an application that meets the need for security and low price in current solutions for parking automotive vehicles. With this, it was found that there is a viable alternative to meet this reported demand, there is the possibility of non-commercial properties to make their spaces available in order to obtain a lower price than that practiced by establishments exclusively intended for parking vehicles and provide greater security to people who leave their cars on public roads. With this in mind, a mobile application was developed that makes it possible to search and advertise these alternative vacancies to conventional models, it is intended for Android and iOS operating systems because it was built with React Native technology, providing a greater range of devices, as well services in the serverless model were used that help in the development of applications with greater possibility of scalability and less effort with development on the server side of the system.

**Keywords:** React Native. Parking. Android. iOS. Vehicles.

## LISTA DE FIGURAS

Figura 1 - Funcionamento da comunicação na arquitetura cliente servidor.....	15
Figura 2 - Ilustração da arquitetura do sistema operacional Android.....	20
Figura 3 - Ilustração das camadas do sistema operacional IOS.....	21
Figura 4 - Tela inicial do aplicativo BlaBlaCar – Caronas e Ônibus.....	24
Figura 5 - Tela inicial do aplicativo Uber – Peça uma viagem.....	25
Figura 6 - Tela inicial do aplicativo AirBnb.....	26
Figura 7 - Diagrama de casos de uso.....	33
Figura 8 - Tabelas no DynamoDB.....	34
Figura 9 - Estrutura das tabelas.....	34
Figura 10 - Estrutura de pastas do armazenamento de imagens.....	35
Figura 11 - Estrutura da função de inclusão.....	35
Figura 12 - Demonstração da função de inserção.....	36
Figura 13 - Exemplo de uso da biblioteca React Native AWS3.....	37
Figura 14 - Arquivo de conexão do Firebase.....	38
Figura 15 - Arquivo de conexão do AWS.....	38
Figura 16 - Telas de autenticação.....	39
Figura 17 - Tela inicial do aplicativo.....	40
Figura 18 - Tela de perfil do usuário.....	41
Figura 19 - Telas para inserção de endereço.....	42
Figura 20 - Tela de resultados da busca de vaga.....	43
Figura 21 – Tela de detalhes sobre a vaga.....	44
Figura 22 – Telas para anúncio de vaga.....	45
Figura 23 – Telas para gerenciamento de vagas.....	46
Figura 24 – Relação de dispositivos testados por robô.....	47
Figura 25 – Painel principal do teste automatizado.....	48
Figura 26 – Painel de desempenho do teste automatizado.....	48
Figura 27 – Gráfico das respostas sobre utilidade da aplicação.....	49
Figura 28 – Gráfico das respostas sobre facilidade de utilização da aplicação.....	49
Figura 29 – Gráfico das respostas sobre o design da aplicação.....	50
Figura 30 – Gráfico das respostas sobre as funcionalidades da aplicação.....	50
Figura 31 – Gráfico das respostas sobre a utilização comercial da aplicação.....	51
Figura 32 – Respostas da pergunta sobre sugestões para a aplicação.....	51

## LISTA DE QUADROS

Quadro 1 - Exemplo de equivalência dos componentes.....	19
Quadro 2 - Publicações relevantes ao tema do presente trabalho.....	22
Quadro 3 - Comparativo entre funcionalidades dos aplicativos.....	27
Quadro 4 - Descrição do problema.....	31
Quadro 5 - Descrição do produto.....	32



## LISTA DE SIGLAS

**AOSP** Android Open-Source Project

**API** Application Programming Interface

**AWS** Amazon Web Services

**AWS S3** Simple Storage Service

**ECMA Internacional** European Computer Manufacturers Association

**ECSI** European Customer Satisfaction Index

**HTML** HyperText Markup Language

**JSX** JavaScript XML

**PLS-PM** Partial Least Squares-Path Modeling

**SQL** Structured Query Language

**UML** Unified Modeling Language

**WWDC** Worldwide Developers Conference

## SUMÁRIO

<b>1</b>	<b>INTRODUÇÃO.....</b>	<b>11</b>
1.1	MOTIVAÇÃO.....	12
1.1	OBJETIVOS.....	13
<b>1.1.1</b>	<b>Objetivo Geral.....</b>	<b>13</b>
<b>1.1.2</b>	<b>Objetivos Específicos.....</b>	<b>13</b>
<b>2</b>	<b>REVISÃO BIBLIOGRÁFICA.....</b>	<b>14</b>
2.1	ARQUITETURA CLIENTE-SERVIDOR.....	14
2.2	SERVIDOR.....	15
<b>2.2.1</b>	<b>Modelo Serverless.....</b>	<b>15</b>
<b>2.2.2</b>	<b>AWS.....</b>	<b>16</b>
<b>2.2.3</b>	<b>GraphQL.....</b>	<b>16</b>
<b>2.2.4</b>	<b>Firebase.....</b>	<b>17</b>
2.3	DYNAMODB.....	17
2.4	CLIENTE.....	18
<b>2.4.1</b>	<b>Linguagem de programação JavaScript.....</b>	<b>18</b>
<b>2.4.2</b>	<b>React Native.....</b>	<b>18</b>
<b>2.4.3</b>	<b>Android.....</b>	<b>19</b>
<b>2.4.4</b>	<b>iOS.....</b>	<b>20</b>
<b>3</b>	<b>ESTADO DA ARTE.....</b>	<b>22</b>
3.1	APLICAÇÕES RELEVANTES.....	23
<b>3.1.1</b>	<b>BlaBlaCar – Caronas e Ônibus.....</b>	<b>24</b>
<b>3.1.2</b>	<b>Uber – Peça uma viagem.....</b>	<b>25</b>
<b>3.1.3</b>	<b>AirBnb.....</b>	<b>26</b>
<b>3.1.4</b>	<b>Comparativo entre aplicações semelhantes.....</b>	<b>27</b>
<b>4</b>	<b>METODOLOGIA.....</b>	<b>28</b>
<b>5</b>	<b>O APLICATIVO FLANELINHA.....</b>	<b>30</b>
5.1	LEVANTAMENTO DE REQUISITOS.....	30
5.2	ARMAZENAMENTO DE DADOS E ARQUIVOS.....	33
5.3	DESENVOLVIMENTO DO FLANELINHA.....	37
<b>5.3.1</b>	<b>Telas de Autenticação.....</b>	<b>39</b>
<b>5.3.2</b>	<b>Tela Inicial.....</b>	<b>40</b>
<b>5.3.3</b>	<b>Tela de Perfil.....</b>	<b>41</b>
<b>5.3.4</b>	<b>Telas para Busca de Vagas.....</b>	<b>42</b>
<b>5.3.5</b>	<b>Tela de Detalhes Sobre a Vaga.....</b>	<b>44</b>
<b>5.3.6</b>	<b>Telas de Anúncio de Vagas.....</b>	<b>45</b>
<b>5.3.7</b>	<b>Telas de Gerenciamento de Anúncios.....</b>	<b>46</b>
5.4	TESTES.....	47
<b>6</b>	<b>CONSIDERAÇÕES FINAIS.....</b>	<b>52</b>
<b>7</b>	<b>REFERÊNCIAS.....</b>	<b>52</b>

## 1 INTRODUÇÃO

O automóvel, qualquer veículo que, movido a motor, se destina ao transporte de pessoas ou cargas (RISCO, 2021), tornou-se o principal veículo da população mundial para este fim desde que inventado. No Brasil, a década de 2010 a 2020 observou um crescimento na frota automotiva superior a 66%, com o número passando de 64.817.974 (IBGE, 2010) para 107.948.371 de veículos (IBGE, 2020). Com esse avanço, o espaço para alocação dos mesmos foi escasseando. Outro ponto a ser destacado é a segurança em relação aos automóveis e seus respectivos proprietários.

Nos grandes centros urbanos, há uma grande concentração de automóveis nas ruas, muitos deles em estado estacionário devido a seus proprietários estarem desempenhando atividades cotidianas. Tendo em vista a grande quantidade de veículos, os espaços disponíveis para estacionamentos acabam sendo poucos e muitas vezes os espaços remanescentes tendem a não ser em locais seguros.

Identificada essa deficiência no processo diário, é possível constatar a possibilidade de informatização do processo de procura de vagas de estacionamento, tal como o anúncio dessas vagas por parte de quem possua um espaço disponível em sua propriedade. Podendo ser suas garagens que ficam desocupadas durante o dia ou até mesmo um espaço aberto dentro de sua propriedade que acomode as dimensões do veículo de quem procura a vaga. Essa busca e comunicação ocorrerá por meio de um aplicativo móvel, onde o foco será para entre pessoas físicas onde elas possam anunciar e procurar vagas, realizando o contato entre si.

Para isso foi realizada uma pesquisa qualitativa a fim de saber o padrão comportamental e as características do público-alvo do aplicativo, tendo uma natureza aplicada e objetivos exploratórios. Foram distribuídos questionários ao público-alvo para poder estabelecer os requisitos do sistema e assim elaborar o diagrama de casos de uso baseado nos padrões da UML (Unified Modeling Language) para uma melhor compreensão do que seria necessário para realizar o desenvolvimento do aplicativo.

No desenvolvimento foi utilizada a linguagem de programação Javascript com o framework React Native que possibilita a adaptação do código para as plataformas predominantes no mercado atual de dispositivos móveis, Android e iOS. O armazenamento de arquivos foi feito utilizando os serviços da Amazon Web Services e de comunicação com o banco de dados por meio da tecnologia GraphQL. O serviço de autenticação de usuários foi feito por meio da plataforma digital Firebase desenvolvida pelo Google.

A metodologia de desenvolvimento utilizada seguiu o padrão ágil, mais especificamente o Scrum Solo, um framework para gerenciamento de projetos com modelo iterativo e incremental (MARTIN, 2009).

Após o desenvolvimento do aplicativo, testes automatizados e de aceitação foram aplicados para validar se os requisitos foram atendidos e se a usabilidade satisfaz a maior parte dos usuários.

O presente trabalho está dividido da seguinte forma em introdução, contendo a motivação e os objetivos, a revisão bibliográfica, o estado da arte, a metodologia, o desenvolvimento, a conclusão e as referências utilizadas.

## 1.1 MOTIVAÇÃO

Tendo em vista o aumento exponencial de automóveis nos últimos anos e o acúmulo de pessoas nos grandes centros urbanos, o espaço ocupado por veículos tem aumentado e conseqüentemente diminuído o espaço disponível para sua utilização. Em 2010, a frota veicular no Brasil possuía 64.817.974 (IBGE, 2010), ao passo que, em 2020, esse número foi para 107.948.371 (IBGE, 2020), um aumento de 66,5% na última década.

Outro aspecto importante a ser levantado é a segurança pública relacionada aos veículos, no estado do Rio Grande do Sul desde 2010 até 2020, foram registrados 322.984 casos de furto ou roubo de veículos (Secretaria de Segurança Pública, 2021). Isso mostra a grande vulnerabilidade dos proprietários de veículos aos criminosos, no estado.

Diante do exposto, o presente trabalho busca aumentar a comodidade, facilidade e segurança aos motoristas, fazendo a conexão entre pessoas que buscam um lugar seguro para estacionarem seus veículos enquanto estão em atividade nesses grandes centros, onde vagas de estacionamento são escassas, e pessoas que possuem algum tipo de espaço em suas residências ou qualquer outro tipo de lote lindeiro que possam destinar para esse fim.

A informatização de processos foi feita para auxiliar na resolução dos problemas de negociação, logística e segurança relatados nos parágrafos anteriores, por meio de um aplicativo desenvolvido, foi possível estabelecer contato de quem demanda desse serviço com quem oferece. O diferencial do aplicativo é realizar a exposição das ofertas, principalmente, de pessoas físicas com suas vagas de garagem que ficam disponíveis por longos períodos durante o dia às pessoas que precisam exercer alguma atividade naquela localidade.

Assim, a elaboração deste aplicativo, de busca e compartilhamento de vagas de estacionamento, visa principalmente aumentar a eficiência e efetividade dos motoristas em suas rotas e afazeres diários.

## 1.1 OBJETIVOS

Nesta seção do presente trabalho estão descritos os objetivos a serem realizados durante o desenvolvimento do aplicativo Flanelinha.

### 1.1.1 Objetivo Geral

Desenvolver um aplicativo para dispositivos móveis onde os usuários possam procurar e anunciar vagas de estacionamento.

### 1.1.2 Objetivos Específicos

- Realizar a engenharia de requisitos necessários para desenvolvimento do aplicativo;
- Analisar aplicativos com função similar para explorar áreas não atendidas;
- Estabelecer a arquitetura de acordo com os requisitos apontados;
- Definir as tecnologias empregadas no desenvolvimento;
- Desenvolver os requisitos funcionais;
- Testar as funcionalidades implementadas.

## 2 REVISÃO BIBLIOGRÁFICA

Neste capítulo é apresentada a base que mantém o presente trabalho. Procura-se realizar a ligação entre os princípios e conceitos abordados às ideias prioritárias da pesquisa. As temáticas trabalhadas estão divididas em duas áreas diferentes ligadas ao desenvolvimento da aplicação. A primeira é o conceito de servidor e as tecnologias utilizadas para seu desenvolvimento e a segunda é o conceito de cliente e as tecnologias utilizadas para seu desenvolvimento. Em relação ao servidor, foram utilizados serviços no modelo de computação *Serverless*, os quais fornecem recursos de *web services* e banco de dados. Em relação ao cliente, foi utilizada a linguagem Javascript e o *framework* React Native, que é baseado em Javascript e proporciona desenvolvimento de aplicações móveis em dispositivos com sistemas Android e iOS.

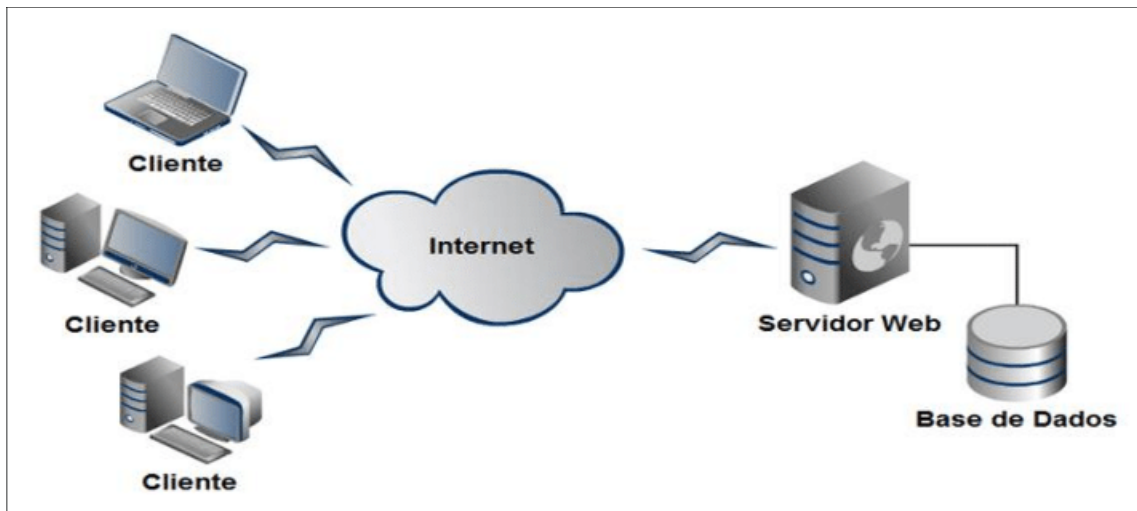
### 2.1 ARQUITETURA CLIENTE-SERVIDOR

Na arquitetura cliente-servidor, a execução do programa se deve à comunicação entre dois objetos, cliente e servidor, ambos os objetos realizam processamento de dados para realizar a atividade fim que o usuário executou. Onde no lado do cliente acontece o envio de requisições para o servidor e no lado do servidor acontece o processamento e execução das requisições enviadas pelo cliente. Ao terminar as operações solicitadas pelo cliente, o servidor retorna a resposta às requisições (COULOURIS, 2013).

Battisti (2001) caracteriza a arquitetura cliente-servidor com três elementos fundamentais, são eles: banco de dados, rede e software. Respectivamente, encarregados de armazenar os dados, trafegar os dados e utilizar os dados.

Na Figura 1 é demonstrado como ocorre a comunicação entre o cliente e o servidor no presente trabalho. O cliente realiza uma requisição que é levada, por meio da internet, até o servidor, que se comunica com a base de dados utilizada para obter os dados requisitados pelo cliente. Ao concluir o processamento da requisição, o servidor retorna uma resposta compatível ao cliente, que recebe por meio da internet os dados para realizar o tratamento pelo software utilizado pelo usuário.

Figura 1 - Funcionamento da comunicação na arquitetura cliente-servidor



Fonte: LIMA, 2013.

## 2.2 SERVIDOR

Nesta seção do presente trabalho, serão descritos os conceitos e as tecnologias que fizeram parte do desenvolvimento de servidor da aplicação.

### 2.2.1 Modelo Serverless

O modelo de computação *Serverless* ou “Sem servidor”, possui essa nomenclatura pelo fato de que o desenvolvedor da aplicação que se utilizará desse modelo não precisará realizar toda a configuração e manutenção do servidor da aplicação, e não por ser um modelo sem a necessidade de máquinas físicas ou virtuais e containers que são utilizados para a disponibilização da aplicação, restando ao desenvolvedor somente a preocupação de desenvolver a aplicação em si (HENDRICKSON *et al.*, 2016).

Ao passo de que ainda são necessários os servidores para o uso da aplicação, existem provedores que fornecem esse tipo de serviço, os quais ficam responsáveis pela disponibilização, manutenção e escalabilidade do serviço contratado. Desse modo, o desenvolvedor da aplicação não tem acesso às máquinas utilizadas nesse tipo de serviço, somente acessa a sua aplicação hospedada no provedor (AWS, 2022e).

## 2.2.2 AWS

A AWS (Amazon Web Services) é uma plataforma que oferta o modelo de arquitetura *serverless*, disponibilizando mais de 200 serviços e funcionalidades a pessoas e empresas, tais como máquinas para hospedar aplicações, infraestrutura, banco de dados, autenticação, entre outros (AWS, 2022d). Com a proposta de oferecer seus serviços, que são majoritariamente baseados em computação em nuvem, com flexibilidade, escalabilidade e segurança (VERAS, 2013).

### 2.2.2.1 AWS AppSync

O AWS AppSync é um serviço oferecido pela plataforma AWS que visa fornecer uma interface para a comunicação entre o cliente e o servidor que seja robusta e escalável para desenvolvedores conciliarem dados de fontes diferentes incluindo o Amazon DynamoDB que foi o banco de dados utilizado no presente trabalho, utilizando a tecnologia de API GraphQL (AWS, 2022a).

### 2.2.2.2 AWS S3

O AWS S3 (Simple Storage Service) é um serviço oferecido pela plataforma AWS que visa oferecer armazenamento de objetos em nuvem, podendo ser utilizado para qualquer volume de dados em qualquer lugar do mundo, assim como os outros serviços da Amazon Web Services, também oferece segurança, flexibilidade e escalabilidade sem a necessidade de configurações e gerenciamento de equipamentos para alterar capacidade de acesso e manutenções de *hardware* (AWS, 2022c).

## 2.2.3 GraphQL

GraphQL é uma linguagem de consulta para API (Application Programming Interface) que faz comunicação com banco de dados por meio de esquemas criados pelo usuário, onde cada elemento do esquema tem um tipo para especificar o dado que será comunicado ao banco de dados, criando uma representação mais acessível ao entendimento do operador do código (GRAPHQL, 2022).



## 2.2.4 Firebase

Semelhante ao Amazon Web Services, o Firebase é uma plataforma criada pela Google Inc. onde disponibiliza serviços de computação no modelo *serverless* com diversas funcionalidades, como banco de dados, armazenamento de arquivos, testes, hospedagem, autenticação, entre outros, estão disponíveis para serem usadas sem a necessidade de configuração de servidores e infraestrutura, tem gerenciamento automático da disponibilidade e escalabilidade dos serviços prestados (FIREBASE, 2022).

### 2.2.4.1 Firebase Authentication

A ferramenta para gerenciamento de autenticação de usuários utilizada no presente trabalho foi a Firebase Authentication, um dos recursos disponibilizados pelo Firebase, que tem por objetivo facilitar o desenvolvimento do sistema de autenticação proporcionando uma solução de identificação completa que é compatível com diversas formas de login, como e-mail e senha, telefone, Google, Twitter, Facebook, entre outros. A forma escolhida para a aplicação atual, foi a de e-mail e senha.

## 2.3 DYNAMODB

O DynamoDB é um serviço de banco de dados NoSQL que utiliza a lógica de chave-valor. Possui um alto desempenho e escalabilidade, oferece criptografia em repouso, eliminando a complexidade e a carga operacional. É possível criar tabelas que armazenam e recuperam qualquer quantidade de dados. Ele faz automaticamente a distribuição dos dados e do tráfego para as tabelas de modo que seus servidores possam suprir todas as requisições mantendo um alto desempenho e consistência. Sua infraestrutura conta com 100% dos dispositivos, que armazenam os dados, sendo discos de estado sólido, deste modo fornecendo durabilidade e disponibilidade (AWS, 2022b). Esta tecnologia foi utilizada no Flanelinha por estar disponível nos serviços da AWS e por atender as demandas da aplicação.

## 2.4 CLIENTE

Nesta seção do presente trabalho, serão descritos os conceitos e as tecnologias que fizeram parte do desenvolvimento de cliente da aplicação.

### 2.4.1 Linguagem de programação JavaScript

JavaScript é uma linguagem de programação leve, interpretada, orientada a objetos e multiplataforma, popularmente conhecida como *script* para web, apesar de ter sido criada para ser utilizada no lado do cliente, também é usada no lado do servidor, por meio de *frameworks* criados a partir do script original. Ela é padronizada pela ECMA Internacional (European Computer Manufacturers Association), que foi criada em 17 de maio de 1961, a última versão de referência para padronização do JavaScript é o ECMAScript 2018. (MOZZILA, 2022).

### 2.4.2 React Native

O React Native é um *framework* JavaScript utilizado para criação de aplicativos móveis com componentes renderizados de forma nativa pelos sistemas Android e iOS. O React Native é baseado no *framework* React que também utiliza JavaScript para criação de aplicações web, ambos foram criados pela empresa Meta Platforms, Inc, antigo Facebook Inc, porém React Native tem por objetivo ser utilizado em dispositivos móveis. Todos os códigos desenvolvidos são feitos em JavaScript, porém são renderizados pelos dispositivos como sendo componentes nativos feitos individualmente para cada sistema, no Quadro 1 é mostrado um exemplo da equivalência entre componentes utilizados na escrita do código em React Native e os componentes utilizados na escrita em código nativo de cada plataforma. React Native foi escolhido para desenvolvimento do presente trabalho para ter uma maior abrangência de dispositivos capazes de rodar a aplicação criada (EISENMAN, 2016).

Quadro 1 - Exemplo de equivalência dos componentes

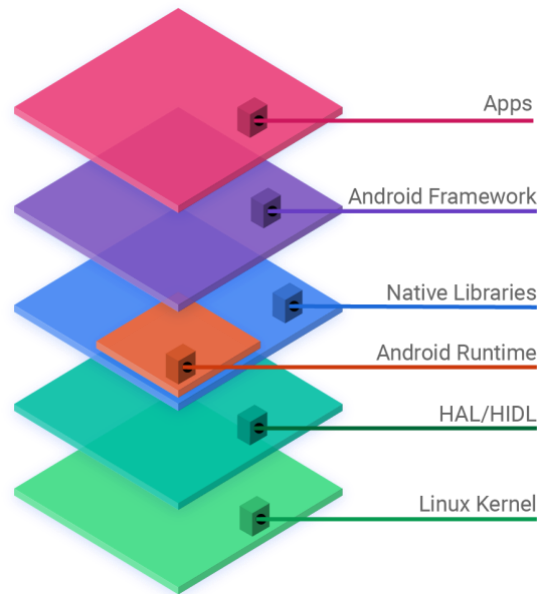
Componente React Native	Componente Android	Componente iOS
<View>	<ViewGroup>	<UIView>
<Text>	<TextView>	<UITextView>
<Image>	<ImageView>	<UIImageView>
<ScrollView>	<ScrollView>	<UIScrollView>
<TextInput>	<EditText>	<UITextField>

Fonte: Elaborado pelo autor (2022).

### 2.4.3 Android

O Android é um sistema operacional, para dispositivos móveis, de código aberto desenvolvido pela Google Inc. denominado AOSP (Android Open Source Project). O Android foi desenvolvido utilizando o mesmo kernel usado no sistema operacional Linux e conta com 2,5 bilhões de dispositivos ativos atualmente, mais de 24.000 dispositivos diferentes e mais de 1 milhão de aplicativos exclusivos na Google Play Store, loja oficial de aplicativos para Android. Sua primeira versão disponibilizada ao público foi a 1.6 e a mais atual é a versão 12 (ANDROID, 2022). Na figura 2 é ilustrado como se dá a arquitetura do sistema operacional em questão.

Figura 2 - Ilustração da arquitetura do sistema operacional Android

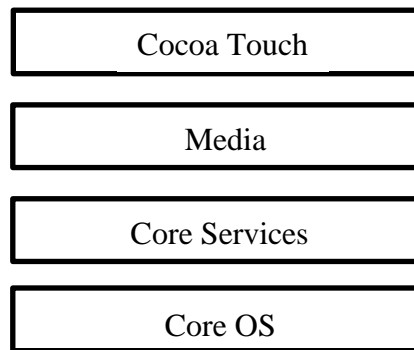


Fonte: ANDROID, 2022.

#### 2.4.4 iOS

O iOS é um sistema operacional feito pela Apple Inc. desenvolvido inicialmente para iPhone, porém foi utilizado posteriormente em iPods e iPads. Se tornou uma das principais plataformas mobile do mundo, seu lançamento em 2007 se deu junto com o lançamento do primeiro iPhone durante a WWDC (Worldwide Developers Conference) realizada pela Apple, naquela ocasião ainda era denominado iPhone OS. Uma das principais características desse sistema operacional é a compatibilidade apenas com dispositivos feitos pela empresa criadora, não sendo possível encontrá-lo em dispositivos de outras marcas. Sua primeira versão foi a 1 e atualmente está na versão 15 (LECHETA, 2014). Na Figura 3 está a ilustração das camadas do iOS.

Figura 3 - Ilustração das camadas do sistema operacional iOS



Fonte: Elaborado pelo autor (2022).

### 3 ESTADO DA ARTE

Foi realizada uma pesquisa na biblioteca online SciELO, Google Acadêmico e ResearchGate com o objetivo de encontrar trabalhos com pertinência ao tema estacionamentos. Os termos empregados na busca foram “estacionamento”, “vagas”, “aplicativo”, “garagem” e “aluguel” individualmente e em combinações entre eles. Somente foram consideradas as publicações posteriores a 2017, com o objetivo de manter um conteúdo com fontes recentes e atualizadas. O quadro 1 consta as publicações onde houve contribuições para o Flanelinha.

Quadro 2 - Publicações relevantes ao tema do presente trabalho

<b>Autor</b>	<b>Título</b>	<b>Ano</b>
Franciane Cougo da Cruz Anderson Cougo da Cruz Paulo Sergio Ceretta	Mensuração da satisfação dos usuários do sistema municipal de estacionamento rotativo pago	2017
Marcos dos Santos João Pedro França de Almeida Caio Gregório de Souza Bruno Garcia da Silva	Uma nova maneira de estacionar veículos de passeio em grandes centros urbanos: proposta do aplicativo “minha vaga”	2018
Leonardo Rodrigues Resende	Projeto conceitual de um equipamento mecanizado para estacionar veículos leves de baixo custo	2020

Fonte: Elaborado pelo autor (2022).

No artigo “Mensuração da satisfação dos usuários do sistema municipal de estacionamento rotativo pago”, os autores Cruz *et al.* (2017) trazem à tona a qualidade do serviço oferecido pelo sistema municipal de estacionamento rotativo de Bagé – RS, baseado nos dados obtidos e aplicados ao *European Customer Satisfaction Index* (ECSI), utilizando o método *Partial Least Squares-Path Modeling* (PLS-PM). Nele foi concluído que os usuários aprovam o sistema e o consideram satisfatório, também foi constatado que havia diferença no nível de satisfação conforme idade e renda, quanto maiores esses fatores, maior era o nível de aprovação do sistema.

No artigo “Uma nova maneira de estacionar veículos de passeio em grandes centros urbanos: proposta do aplicativo “minha vaga”” os autores Santos *et al.* (2018) visam a criação de um modelo conceitual de sistema similar ao do presente trabalho. Consiste em uma ferramenta para alugar uma vaga de estacionamento por um certo período, onde os interessados buscam vagas disponíveis mediante agendamento e quem possui a vaga pode aceitar ou não a solicitação de aluguel. O modelo do conceito foi idealizado no formato de aplicativo em

dispositivos móveis, sendo preciso desenvolver interfaces e regras de negócios para tornar possível a conclusão do aplicativo. O modelo não chegou a ser implementado.

O trabalho de conclusão de curso “Projeto conceitual de um equipamento mecanizado para estacionar veículos leves de baixo custo” de Resende (2020) buscou uma solução de baixo custo para otimizar o espaço ocupado por veículos quando estacionados, visto que a demanda por vagas de estacionamento é maior do que a quantidade ofertada. A proposta trabalhada resultou em um produto apto a suportar 6 veículos de maneira que ficariam suspensos em sentido vertical similar a um mecanismo de roda gigante, entretanto, o valor estimado obtido do mecanismo foi 11% mais caro em relação aos disponíveis no mercado, concluindo então que o valor alto foi devido ao preço superdimensionado dos materiais ou ao valor referir a fabricação de apenas um exemplar da solução.

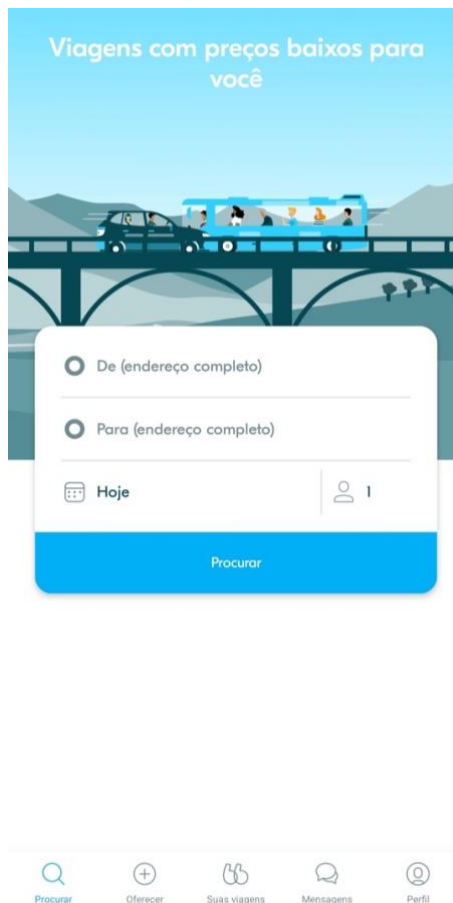
Os artigos, relevantes para o presente trabalho, encontrados nas bibliotecas digitais citadas no início do capítulo, tornaram mais clara a percepção do autor sobre os últimos acontecimentos nesta área de estudo, levando a uma compreensão mais abrangente do que já havia sido trabalhado, e às deficiências ainda abertas a possíveis soluções, deficiências essas que, ao menos em parte, o presente trabalho tentou criar uma alternativa de solução viável de ser executada.

### 3.1 APLICAÇÕES RELEVANTES

Foi realizada uma busca de aplicações similares à desenvolvida com o objetivo de obter possíveis referências para funcionalidades e descoberta de deficiências funcionais que possam vir a ser supridas. Não há nenhuma aplicação semelhante ao Flanelinha disponível nas lojas de aplicativos Google Play e App Store, contudo há aplicações com funcionalidades específicas e princípios similares ao do presente trabalho.

### 3.1.1 BlaBlaCar – Caronas e Ônibus

Figura 4 - Tela inicial do aplicativo BlaBlaCar – Caronas e Ônibus.



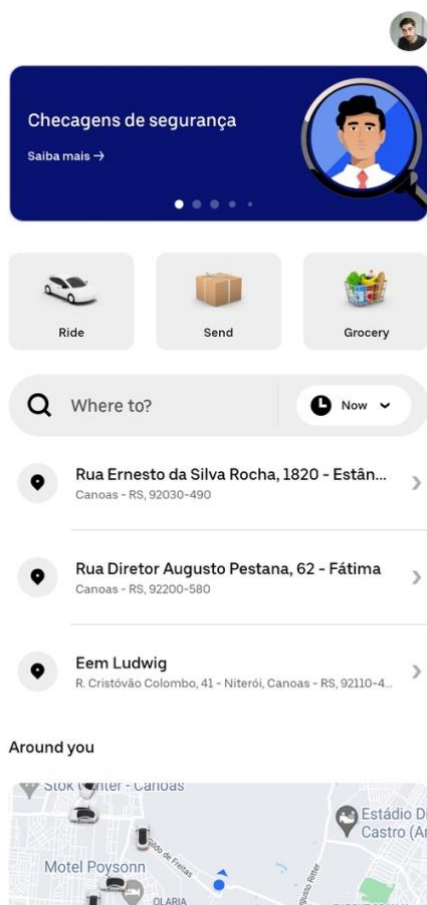
Fonte: Aplicativo BlaBlaCar – Caronas e Ônibus (2022).

O BlaBlaCar possui duas principais funcionalidades, comprar passagens de ônibus e encontrar pessoas com o mesmo local de destino e interessadas em dividir os custos de uma viagem. No aplicativo, o motorista cadastra a sua viagem com as informações necessárias e o passageiro, ao inserir o seu ponto de destino, vê todas as opções de motoristas com rota compatível e o valor da viagem. Ao escolher um motorista, combina-se o ponto de encontro e seguem para o destino. Esse aplicativo é semelhante ao do presente trabalho, conectando pessoas que tem um serviço a oferecer com outras que procuram uma solução mais barata do que a convencional, porém se diferem no tipo de serviço oferecido.



### 3.1.2 Uber – Peça uma viagem

Figura 5 - Tela inicial do aplicativo Uber – Peça uma viagem

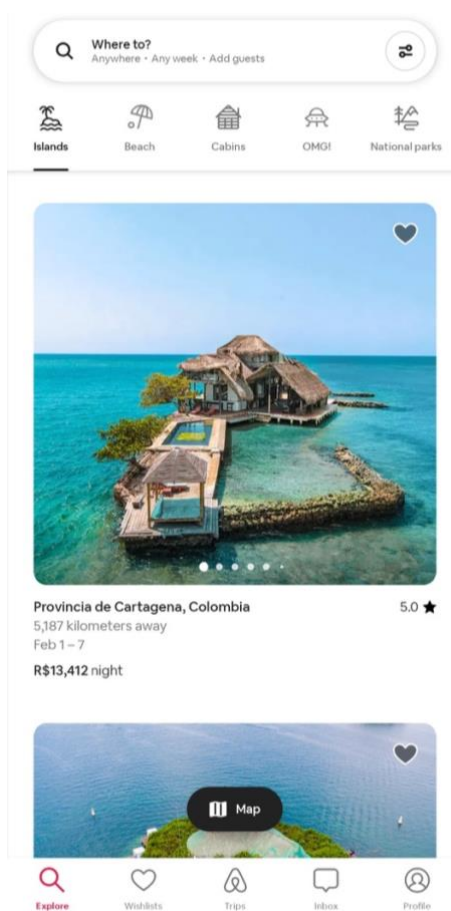


Fonte: Aplicativo Uber – Peça uma viagem (2022).

O Uber é uma aplicação onde o usuário pode solicitar transporte até o destino solicitado no mapa, o aplicativo então procura os motoristas disponíveis para a viagem, um deles, então, aceita e se desloca até a localização do usuário e o leva ao endereço desejado. Durante o processo, o usuário pode escolher a modalidade do veículo que o transportará, contatar o motorista por mensagens dentro do aplicativo e por ligações. Tanto o Flanelinha quanto o Uber se preocupam com a mobilidade urbana e tem por objetivo facilitar a rotina de quem precisa se locomover com veículos automotores, semelhante ao BlaBlaCar, porém diferente da aplicação do presente trabalho, o Uber tem foco no trajeto do usuário enquanto o Flanelinha visa auxiliar no ponto final da trajetória do usuário.

### 3.1.3 AirBnb

Figura 6 - Tela inicial do aplicativo AirBnb



Fonte: Aplicativo AirBnb (2022).

O AirBnb é um aplicativo móvel que tem por objetivo principal a busca e anúncio de espaços residenciais para locação por período pré-determinado. O usuário pode pesquisar cidade e áreas que lhe forem interessantes, com isso serão exibidos anúncios relevantes e dentro dos parâmetros que foram adicionados ao filtro de busca. Pode-se fazer a comunicação diretamente com o proprietário para obter mais informações e esclarecer eventuais dúvidas sobre o imóvel, efetuando pagamento pela plataforma interna de pagamentos. Semelhante ao aplicativo do presente trabalho, o AirBnb possui por característica o modelo de negócio no segmento de aluguéis, porém de propriedades para fins do usuário pernoitar no espaço oferecido, enquanto o Flanelinha se destina ao estacionamento veicular.

### 3.1.4 Comparativo entre aplicações semelhantes

No Quadro 3 é demonstrado um comparativo entre as funcionalidades das aplicações relatadas anteriormente, onde é possível ver as funções de cada aplicação em relação ao aplicativo Flanelinha, o qual é o único que atende a necessidade de disponibilizar as vagas de estacionamento em propriedades não comerciais, nos outros aplicativos analisados, identificou-se a carência dessa funcionalidade, tornando clara a utilidade do presente trabalho. A análise das aplicações foi benéfica à construção do aplicativo Flanelinha pois foi possível mapear o que cada um proporcionava ao usuário e o que ainda havia a necessidade de ser desenvolvido e que fosse compatível com o objetivo do presente trabalho. A análise também ajudou a compreender a atual situação no mercado de aplicativos com funções semelhantes.

Quadro 3 - Comparativo entre funcionalidades dos aplicativos

<b>Funcionalidades</b>	<b>BlaBlaCar</b>	<b>Uber</b>	<b>AirBnb</b>	<b>Flanelinha</b>
Pesquisa de endereços	x	x	x	x
Aluguel de imóveis			x	
Visualização do mapa		x	x	x
Inserção de anúncios	x		x	x
Busca de vagas de estacionamento				x

Fonte: Elaborado pelo autor (2022).

## 4 METODOLOGIA

O presente trabalho tem como classificação, a respeito de sua abordagem, qualitativa, segundo Polit e Beck (2016), "tende a enfatizar o raciocínio dedutivo, as regras da lógica e os atributos mensuráveis da experiência humana". A respeito de sua natureza, a classificação se determina como aplicada, onde os dados levantados foram transformados em informações que possibilitaram o desenvolvimento da aplicação de forma prática, segundo Gerhardt e Silveira (2009) "este tipo de pesquisa tem como objetivo proporcionar maior familiaridade com o problema, com vistas a torná-lo mais explícito ou a construir hipóteses". A respeito dos objetivos, a classificação se determina como exploratória, onde as informações foram extraídas dos dados coletados nos questionários feitos.

A engenharia de requisitos foi o ponto de partida para o desenvolvimento da aplicação. Os requisitos foram definidos por meio dos resultados dos questionários aplicados, e possibilitaram a descoberta dos pensamentos e necessidades do público-alvo. Levantados os principais requisitos da aplicação, na fase de modelagem foi elaborado o diagrama de casos de uso da UML (Unified Modeling Language), o qual foi relevante ao trabalho em questão para ampliar o esclarecimento do leitor. Foi de grande importância para obter um maior entendimento do funcionamento e dos requisitos da aplicação, o diagrama de casos de uso visa demonstrar, em linguagem mais acessível, as principais funcionalidades do sistema e quais personas podem executá-las.

Passada a etapa de modelagem, teve início a elaboração das telas da aplicação, as quais foram elaboradas por meio da extensão de sintaxe Javascript, chamada JSX (JavaScript XML) (REACT, 2022), que possui estrutura semelhante ao HTML (HyperText Markup Language). Foi realizada uma curadoria de *designs* para obter a inspiração em outros aplicativos e determinar qual seria a opção ideal que se encaixaria dentro das particularidades do aplicativo Flanelinha. A curadoria foi feita buscando *designs* de aplicativos e protótipos de *designs* de aplicativos que possuíam telas com funções semelhantes às da presente aplicação, as fontes de procura foram as lojas de aplicativos dos dois sistemas alvo e nas plataformas Pinterest e Google Imagens.

Tido em vista a necessidade de haver armazenamento de dados, foram utilizadas duas tecnologias para realizar essa funcionalidade, a linguagem de *query* GraphQL que permite fácil estruturação das coleções do banco de dados escolhido, o DynamoDB, que não utiliza a linguagem SQL (Structured Query Language), tampouco organização dos dados em tabelas.

Tanto a GraphQL quanto DynamoDB se adequam bem ao Javascript. O esquema das coleções foi estruturado a partir do GraphQL, onde foram estabelecidos quais dados cada coleção teria, seus respectivos tipos e se os dados eram ou não obrigatórios para que a inserção ocorresse corretamente. As imagens utilizadas na aplicação, foram armazenadas pelo serviço de armazenamento de arquivos S3.

Após já ter definido as estruturas de dados, armazenamento e telas, foram implementadas as funcionalidades do aplicativo Flanelinha com base em Javascript e React Native com o intuito de deixá-lo funcional para os usuários, os quais foram autenticados na plataforma por meio da tecnologia de autenticação fornecida pelo Firebase.

Durante todo o processo de desenvolvimento do aplicativo em questão, foram realizados testes unitários de todas as funcionalidades para cobrir a maior parte do funcionamento da aplicação, também foram executados testes automatizados por robôs com exploração aleatória da aplicação tendo em vista testar a estabilidade, paralelamente aos testes automatizados, foram realizados testes de aceitação com o usuário.

## 5 O APLICATIVO FLANELINHA

O presente capítulo possui a finalidade de expor ao leitor o processo de criação, desenvolvimento e os resultados obtidos com o produto do trabalho em questão. A engenharia de software torna a construção do sistema mais robusta e precisa, evita muitos retrabalhos que podem vir a ocorrer devido à falta de clareza nos processos e tomadas de decisão durante as fases do desenvolvimento. Tendo em vista esses princípios de alto valor para a criação do aplicativo Flanelinha, o processo de desenvolvimento escolhido teve como base as metodologias ágeis, que se encaixaram mais adequadamente às necessidades da ocasião.

### 5.1 LEVANTAMENTO DE REQUISITOS

Os requisitos e funcionalidade do aplicativo Flanelinha foram construídos em torno de duas premissas principais, os resultados das pesquisas direcionadas ao público-alvo da aplicação, que foram grande contribuição para conhecer as necessidades e padrões de comportamento deles, e as ideias baseadas nas experiências prévias do autor do presente trabalho em construção de sistemas.

O questionário obteve os seguintes resultados, houve 35 respondentes que em sua maioria se mostrou a favor da ideia e objetivo da aplicação. O público da pesquisa eram pessoas maiores de 18 anos que possuíam carteira de habilitação e/ou tivessem veículos automotores. As principais indagações levantadas ao público foram as seguintes:

- se eles usavam estacionamentos fechados ou estacionavam diretamente nas vias públicas. Em sua maioria, o público confirmou que utilizava diretamente as vias públicas para estacionar;
- se eles consideravam seguro estacionar fora de lugares fechados. Em sua maioria, consideravam inseguro usar as vias públicas;
- se eles julgavam caros os preços aplicados por estacionamentos particulares. Em sua maioria, julgavam caro os preços de estacionamentos particulares;
- se eles usariam a garagem de uma pessoa desconhecida para deixarem seus carros. Em sua maioria, usariam a garagem de terceiros;
- se eles disponibilizariam suas garagens para desconhecidos estacionarem seus veículos. Em sua maioria, disponibilizariam a sua garagem a desconhecidos;

- se eles usariam um aplicativo no mesmo intuito do Flanelinha. Em sua maioria também usariam um aplicativo que disponibilizasse esse serviço de busca e anúncio de vagas.

Baseado nas respostas obtidas foi possível elaborar a descrição do problema que afetava o público-alvo da aplicação, podendo assim estabelecer uma definição de características de sistema mais objetiva e concisa. O Quadro 4 contém, de forma didática, a descrição do problema a ser resolvido pelo aplicativo Flanelinha e, também, de maneira acessível, o Quadro 5 contém a descrição do produto resultante do presente trabalho.

Quadro 4 - Descrição do problema

<b>O problema</b>	o país ter muitos veículos automotores e centros urbanos com pouco espaço físico para armazená-los, aliado aos altos índices de roubo e furto de veículos
<b>Afeta</b>	pessoas que possuem carteira nacional de habilitação e as que possuem ou não veículos automotores
<b>E seu impacto</b>	aumento dos preços de locais disponíveis para estacionamentos desses veículos de maneira segura
<b>E uma solução de sucesso seria</b>	a criação de uma aplicação que fizesse a conexão entre as pessoas que têm o desejo de deixar seus veículos em local seguro e pessoas que têm essa garagem ou esse local disponível sem uso durante o período que o dono do veículo necessita.

Fonte: Elaborado pelo autor (2022).

Quadro 5 - Descrição do produto.

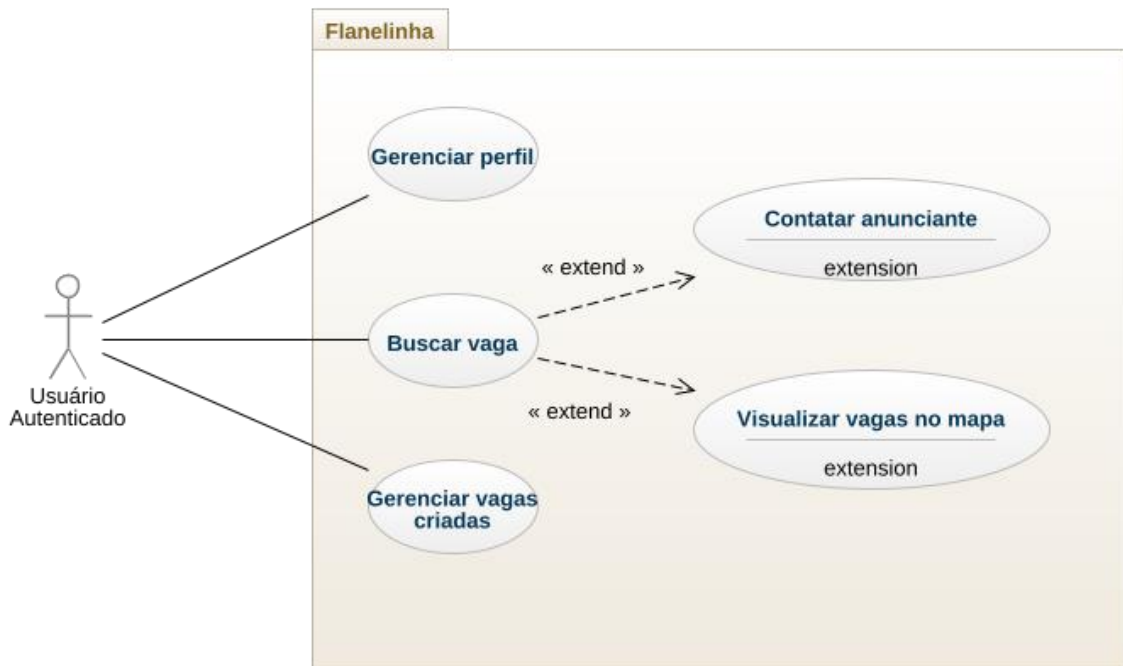
<b>Para</b>	os motoristas de veículos automotores ou proprietários de garagens ou locais não comerciais
<b>Que</b>	querem encontrar local seguro e com baixo custo para estacionamento ou disponibilizar sua propriedade para aluguel
<b>O</b>	Flanelinha
<b>É um</b>	aplicativo móvel
<b>Que</b>	viabiliza a buscar e anunciar vagas de estacionamento em locais não comerciais
<b>Ao contrário</b>	das opções de aplicações disponíveis nas lojas que não possibilitam realizar buscas tampouco realizar anúncios de vagas de estacionamento
<b>Nosso produto</b>	dispõe de todas as funcionalidades para suprir a carência gerada pela necessidade do público-alvo.

Fonte: Elaborado pelo autor (2022).

Após a elaboração dos quadros acima, de acordo com os resultados da pesquisa e com a experiências do autor, foi possível elaborar o diagrama de casos de uso da UML, onde há a definição das funcionalidades do sistema e quem pode vir a usá-las, a figura 7 ilustra quais são essas funcionalidades. O aplicativo Flanelinha tem dois atores, o visitante e o usuário autenticado, o visitante não pode desempenhar nenhuma função além de se autenticar no sistema ou criar um usuário para utilizar o aplicativo, o usuário autenticado, por sua vez, possui todas as permissões para usufruir das funcionalidades que estão disponíveis dentro da aplicação, como gerenciar buscar vagas, anunciar vagas, gerenciar vagas anunciadas e gerenciar perfil.



Figura 7 - Diagrama de casos de uso

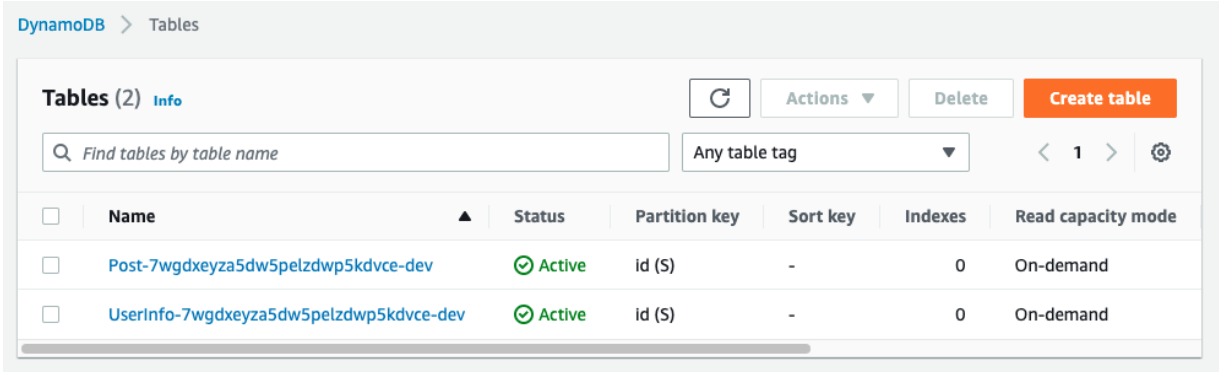


Fonte: Elaborado pelo autor (2022).

## 5.2 ARMAZENAMENTO DE DADOS E ARQUIVOS

Concluída a definição dos requisitos concernentes à aplicação do presente trabalho, notou-se que havia necessidade de armazenamento de dados, tendo isso em vista, foram escolhidas as tecnologias da AWS por oferecer estabilidade e disponibilidade satisfatórias. Para armazenar os dados do aplicativo, escolheu-se o DynamoDB, banco de dados não relacional e que não utiliza a linguagem SQL, sua estruturação foi definida por meio do GraphQL. Na figura 8 é mostrado o console onde estão as tabelas criadas. Na figura 9 está a estrutura definida de cada tabela na sintaxe própria do GraphQL. Para o armazenamento das imagens usadas na aplicação, a tecnologia usada foi o *storage* S3, que guarda as imagens transferidas para ele e gera um link para acesso das mesmas, o link é adicionado na tabela do DynamoDB referente à vaga adicionada, a figura 10 mostra o console onde está a estrutura de pastas no S3.

Figura 8 - Tabelas no DynamoDB



<input type="checkbox"/>	Name	Status	Partition key	Sort key	Indexes	Read capacity mode
<input type="checkbox"/>	Post-7wgdkeyza5dw5pelzdpw5kdvce-dev	Active	id (S)	-	0	On-demand
<input type="checkbox"/>	UserInfo-7wgdkeyza5dw5pelzdpw5kdvce-dev	Active	id (S)	-	0	On-demand

Fonte: Elaborado pelo autor (2022).

Figura 9 - Estrutura das tabelas

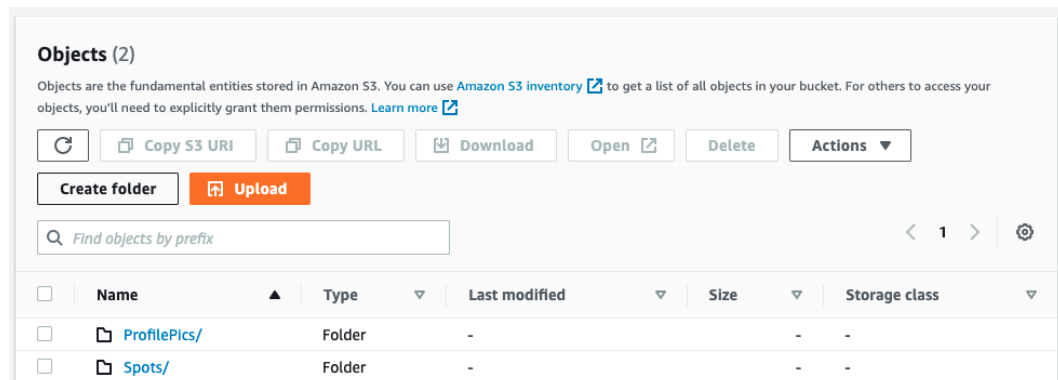
```

4
5 type Post @model {
6   id: ID!
7   title: String!
8   description: String!
9   price: Float!
10  open: String!
11  close: String!
12  isCovered: Boolean!
13  spots_available: Int!
14  type: String!
15  latitude: Float!
16  longitude: Float!
17  image: String
18  phone: String!
19  user_name: String!
20  user_image: String
21  user_id: String!
22 }
23
24 type UserInfo @model {
25   id: ID!
26   phone: String
27   profilePic: String
28   user_name: String
29   user_id: String
30 }
31

```

Fonte: Elaborado pelo autor (2022).

Figura 10 - Estrutura de pastas do armazenamento de imagens



Fonte: Elaborado pelo autor (2022).

Por não ser um banco de dados que se utiliza de SQL, a comunicação é feita por funções em GraphQL, que são geradas pela plataforma AWS AppSync e são o meio utilizado em código para realizar as transações. Nas figuras 11 e 12 está um exemplar de modelo das funções de operações e uma demonstração da mesma em uso.

Figura 11 - Estrutura da função de inclusão

```

4  export const createPost = /* GraphQL */ `
5      mutation CreatePost(
6          $input: CreatePostInput!
7          $condition: ModelPostConditionInput
8      ) {
9          createPost(input: $input, condition: $condition) {
10             id
11             title
12             description
13             price
14             open
15             close
16             isCovered
17             spots_available
18             type
19             latitude
20             longitude
21             image
22             phone
23             user_name
24             user_image
25             user_id
26             createdAt
27             updatedAt
28         }
29     }
30 `;

```

Fonte: Elaborado pelo autor (2022).

Figura 12 - Demonstração da função de inserção

```
103     try {
104         const unsub = API.graphql({
105             query: createPost,
106             variables:
107                 { input: {
108                     title: title,
109                     description: description,
110                     price: parseFloat(price),
111                     open: date,
112                     close: date2,
113                     isCovered: isCovered,
114                     spots_available: spots_available,
115                     type: type,
116                     latitude: latitude,
117                     longitude: longitude,
118                     image: response.body.postResponse.location,
119                     user_id: user_id,
120                     phone: phone,
121                     user_name: user?.displayName,
122                     user_image: user_image
123                 }
124             }
125         });
126     }
```

Fonte: Elaborado pelo autor (2022).

Para o armazenamento das imagens utilizadas, foi feito o uso da biblioteca que visa facilitar a inserção de arquivos ao S3, a React Native AWS3, o comando `npm install react-native-aws3` é um exemplo de como são adicionadas bibliotecas feitas pela comunidade de desenvolvedores da tecnologia principal do presente trabalho. Na figura 13 está um exemplo de um dos seus usos no aplicativo Flanelinha.

Figura 13 - Exemplo de uso da biblioteca React Native AWS3

```
77     const file = {
78         uri: imageURI,
79         name: user?.uid,
80         type: mime.getType(imageURI)
81     }
82
83     const config = {
84         keyPrefix: "ProfilePics/",
85         bucket: 'flanelinhacorp',
86         region: 'sa-east-1',
87         accessKey: '#ACCESS KEY HERE',
88         secretKey: '#SECRET ACCESS KEY HERE',
89         successActionStatus: 201
90     }
91
92     try {
93         const result = await RNS3.put(file, config).then((response) => {
94             console.log(response.body.postResponse.location);
95             setDownloadURL(response.body.postResponse.location)
```

Fonte: Elaborado pelo autor (2022).

### 5.3 DESENVOLVIMENTO DO FLANELINHA

Na etapa de desenvolvimento do aplicativo, foi usada a linguagem JavaScript com *framework* React Native, ao optar pela arquitetura *serverless*, os códigos necessários para que as funcionalidades do aplicativo funcionem, estão juntamente com a parte visual do Flanelinha. Foi usado a tecnologia para autenticação de usuários do Google Firebase, plataforma onde se cria projetos de acordo com o sistema alvo para usar as soluções oferecidas. Na figura 14 está o arquivo que faz a conexão com o Firebase e define as funcionalidades utilizadas na aplicação. Na figura 15 está o arquivo de conexão com os serviços do AWS.

Figura 14 - Arquivo de conexão do Firebase

```

1 // Import the functions you need from the SDKs you need
2 import { initializeApp } from "firebase/app";
3 import { getFirestore } from "firebase/firestore";
4 import { getStorage } from "firebase/storage";
5 import { getAuth } from "firebase/auth";
6 import { getPerformance } from "firebase/performance";
7
8 const firebaseConfig = {
9   apiKey: "#API KEY HERE",
10  authDomain: "#####.firebaseapp.com",
11  projectId: "#####",
12  storageBucket: "#####.appspot.com",
13  messagingSenderId: "#####",
14  appId: "#####"
15 };
16
17 // Initialize Firebase
18 const app = initializeApp(firebaseConfig);
19
20
21 export const storage = getStorage(app);
22 export const db = getFirestore(app);
23 export const auth = getAuth(app);

```

Fonte: Elaborado pelo autor (2022).

Figura 15 - Arquivo de conexão do AWS

```

4 const awsmobile = {
5   "aws_project_region": "sa-east-1",
6   "aws_appsync_graphqlEndpoint": "https://inmbmciznze6jlvnbicqztmm7u.appsync-api.sa-east-1.amazonaws.com/graphql",
7   "aws_appsync_region": "sa-east-1",
8   "aws_appsync_authenticationType": "API_KEY",
9   "aws_appsync_apiKey": "#####",
10  "aws_cognito_identity_pool_id": "sa-east-1:e586fcce-d164-46ce-a136-aa859464b6db",
11  "aws_cognito_region": "sa-east-1",
12  "aws_user_pools_id": "sa-east-1_ZaeU4CFk3",
13  "aws_user_pools_web_client_id": "jclipagru0ln3g450glucgdj5",
14  "oauth": {
15    "domain": "flanelacorpaaf18870-aaf18870-dev.auth.sa-east-1.amazoncognito.com",
16    "scope": [
17      "phone",
18      "email",
19      "openid",
20      "profile",
21      "aws.cognito.signin.user.admin"
22    ],
23    "redirectSignIn": "myapp://signIn/",
24    "redirectSignOut": "myapp://signOut/",
25    "responseType": "code"
26  }
27 };
28
29
30 export default awsmobile;

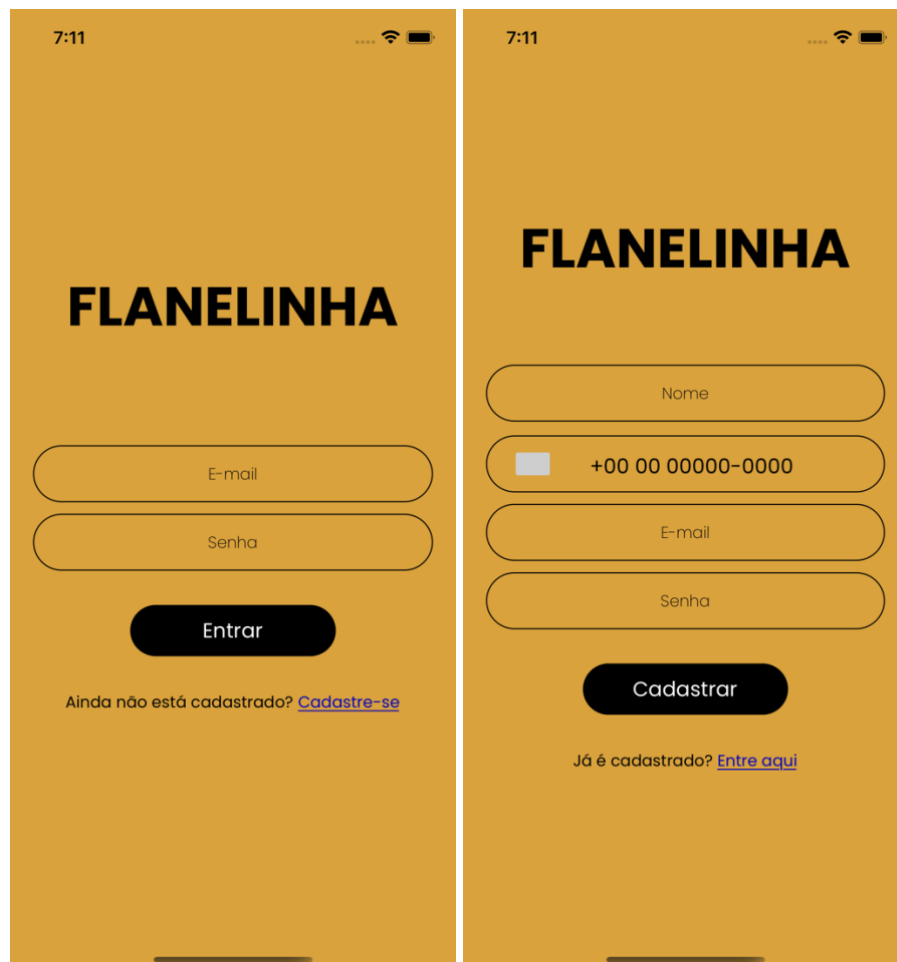
```

Fonte: Elaborado pelo autor (2022).

### 5.3.1 Telas de Autenticação

As telas de autenticação, na figura 16, do aplicativo oferecem a funcionalidade de autenticação de usuários, que possibilita maior segurança para a aplicação, nesta tela é possível inserir credenciais para realizar a autenticação ou inserir as informações necessárias para criação de usuário. Quando o usuário efetua a autenticação, o aplicativo envia as informações inseridas nos campos e-mail e senha para o serviço do Firebase, que retorna a sessão do usuário ou um erro caso as informações inseridas estejam incorretas. Se for um cadastro de usuário, ele envia as informações inseridas nos campos e em seguida as envia para o Firebase, que cadastra o usuário e retorna uma sessão ativa ou retorna um erro se as informações estiverem incorretas.

Figura 16 - Telas de autenticação



Fonte: Elaborado pelo autor (2022).

### 5.3.2 Tela Inicial

A tela inicial do aplicativo, na figura 17, expõe o nome do usuário autenticado ao Firebase, onde é realizada a busca do nome do usuário naquela sessão e todas as funcionalidades principais que o aplicativo Flanelinha possui. Dela é possível ir para as telas referentes a busca de vagas, anúncio de vagas, vagas anunciadas pelo usuário e a sua tela de perfil.

Figura 17 - Tela inicial do aplicativo



Fonte: Elaborado pelo autor (2022).



### 5.3.3 Tela de Perfil

A tela de perfil do usuário, na figura 18, contém as informações inseridas por ele no momento do cadastro, o nome e o e-mail são buscados do serviço de autenticação do Firebase, o número de telefone é buscado do banco de dados, também é possível inserir ou alterar sua foto de perfil e é a tela onde está disponível a função de desconectar a sessão daquele usuário autenticado no momento.

Figura 18 - Tela de perfil do usuário

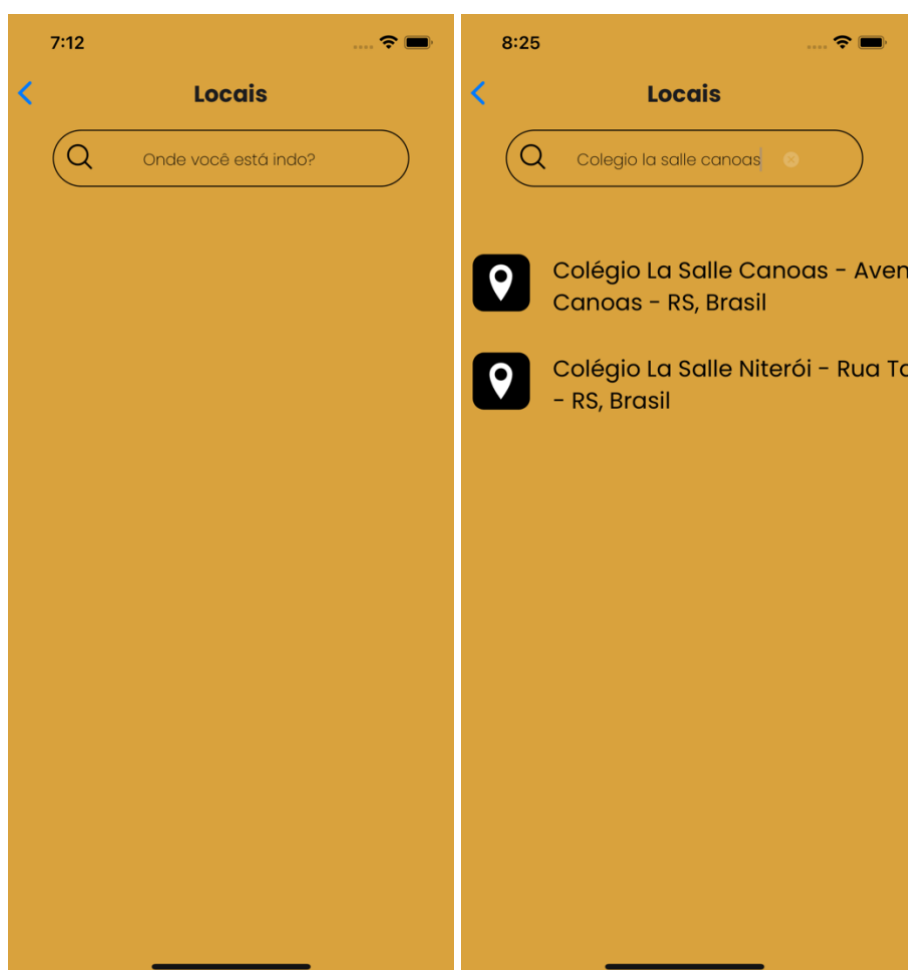


Fonte: Elaborado pelo autor (2022).

### 5.3.4 Telas para Busca de Vagas

O processo de realizar a busca de uma vaga, demonstrado na figura 19, se decorre em inserir o local de seu destino na barra de endereços denominada “Onde você está indo?”, com isso será ativada a biblioteca auxiliar do Google Autocomplete, que realiza a conexão com o banco de dados do Google por meio da chave de API necessária para utilização dos recursos do Google, a biblioteca passa a autocompletar o endereço que está sendo inserido, após achar o endereço nas sugestões, o usuário toca nela e o aplicativo inicia a busca pelas vagas anunciadas em torno de 200 metros de distância. As vagas são achadas por meio da latitude e longitude do endereço inserido pelo usuário.

Figura 19 - Telas para inserção de endereço

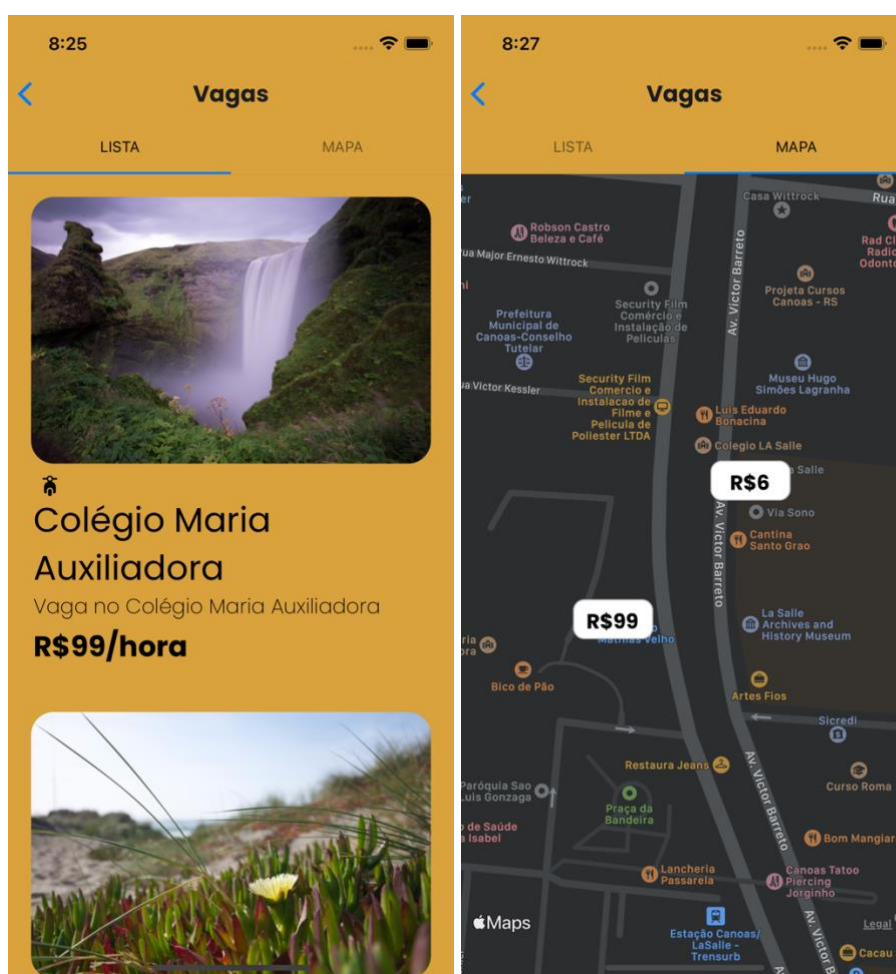


Fonte: Elaborado pelo autor (2022).

Após executar a função para listar as vagas cadastradas no banco de dados, ela retorna os registros filtrados pelos dados geográficos atribuídos àquela vaga, a figura 20 mostra as duas

abas da tela, a primeira aba possui o resultado com os locais encontrados naquela região que foi inserida na barra de endereços na tela anterior em forma de lista, na segunda aba o aplicativo Flanelinha oferece a possibilidade de ver em mapa onde estão localizadas as vagas, bem como o seu preço, para utilizar o mapa, foi adicionada a biblioteca auxiliar React Native Maps components for iOS + Android que possibilita trazer diferentes plataformas de mapas para dentro da aplicação, no sistema Android foi utilizado os recursos de mapa do Google Maps e no sistema iOS foi utilizado os recursos do Apple Maps.

Figura 20 - Tela de resultados da busca de vaga

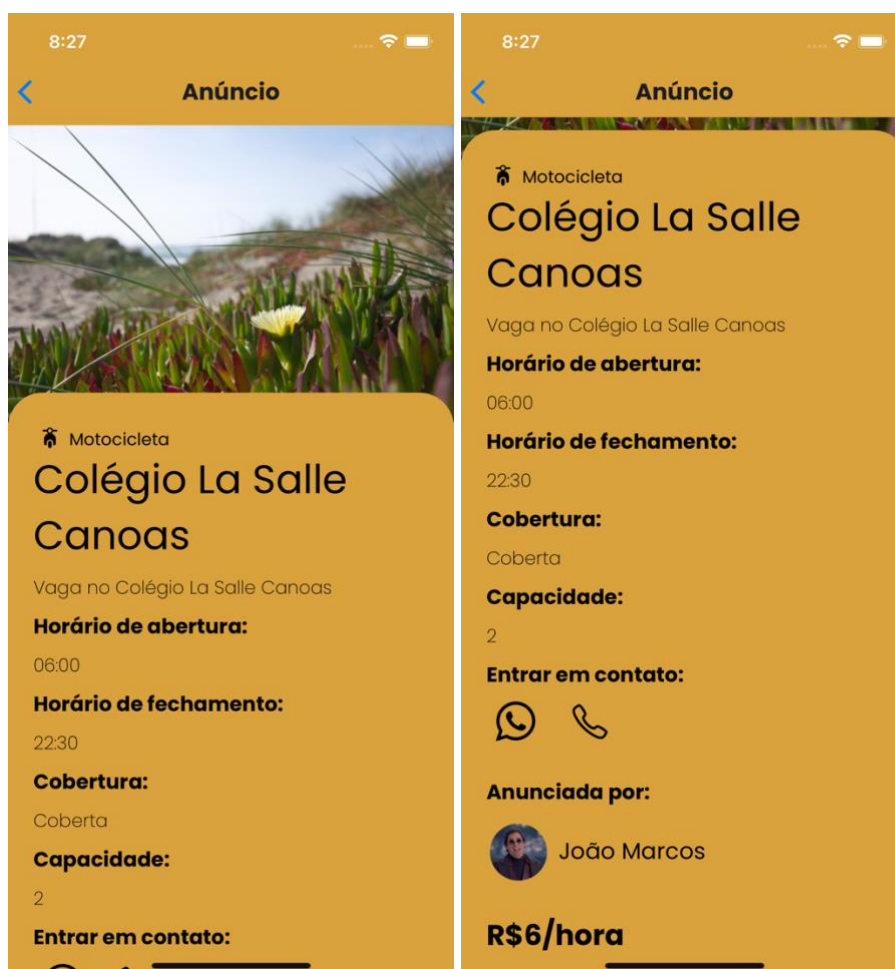


Fonte: Elaborado pelo autor (2022).

### 5.3.5 Tela de Detalhes Sobre a Vaga

Quando o resultado da busca pelas vagas retorna a lista com todas as vagas encontradas, é possível obter mais detalhes da vaga desejada. Ao clicar na vaga, é aberta uma nova tela com mais detalhes e informações sobre a vaga. Na figura 21 é mostrado quais informações estão disponíveis para acesso do usuário. Há também um atalho que abre o número do anunciante direto ao discador do celular para fazer uma ligação e, ao lado, um atalho que direciona o usuário para uma conversa diretamente com o anunciante da vaga no aplicativo WhatsApp, na qual há um texto inicial com o nome do anunciante e título da vaga, como no exemplo a seguir: “Olá [nome do anunciante], vi sua vaga [título da vaga] no aplicativo Flanelinha, vamos conversar sobre ela?”.

Figura 21 – Tela de detalhes sobre a vaga



Fonte: Elaborado pelo autor (2022).

### 5.3.6 Telas de Anúncio de Vagas

Similar ao processo de busca, o anúncio de vagas se inicia pela inserção do endereço onde a vaga se localiza, na barra de endereços. Após o preenchimento do endereço, o aplicativo direciona o usuário para a tela de preenchimento das informações pertinentes à vaga. A figura 24 mostra a tela para inserir o endereço e quais informações são necessárias para fazer o anúncio de alguma vaga, as demais informações como, nome, telefone e foto de perfil do usuário, são coletadas no banco de dados e inseridas no novo registro da vaga anunciada.

Figura 22 – Telas para anúncio de vaga

The figure displays two screenshots of a mobile application interface for job posting, titled "Anunciar Vaga".

The left screenshot (3:54) shows the initial step where the user is prompted to enter the address: "Qual é o endereço da vaga?".

The right screenshot (7:29) shows the subsequent form for providing job details:

- Input field for "título do anúncio"
- Input field for "descrição do anúncio"
- Question "a vaga é coberta?" with radio buttons for "coberta" (selected) and "descoberta"
- Question "para qual tipo de veículo?" with radio buttons for "carro" (selected), "moto", and "caminhão"
- Input field for "preço por hora"
- Input field for "lugares disponíveis"
- Two input fields for "horário inicial" and "horário final", both set to "00:00"
- Input field for "carregar foto"
- A prominent black button labeled "anunciar" at the bottom.

Fonte: Elaborado pelo autor (2022).

### 5.3.7 Telas de Gerenciamento de Anúncios

O usuário também tem acesso a gerência das vagas que ele mesmo anunciou, na figura 23 é mostrado o exemplo da lista de anúncios de usuário, cada anúncio, quando inserido, recebe o número de identificação do usuário que é atribuído pelo Firebase quando o usuário é cadastrado. A busca de vagas do usuário é feita a partir desse número no banco de dados. Todos os registros que possuem esse mesmo número são retornados nesta tela, onde o usuário pode abrir a vaga e excluí-la.

Figura 23 – Telas para gerenciamento de vagas



Fonte: Elaborado pelo autor (2022).

## 5.4 TESTES

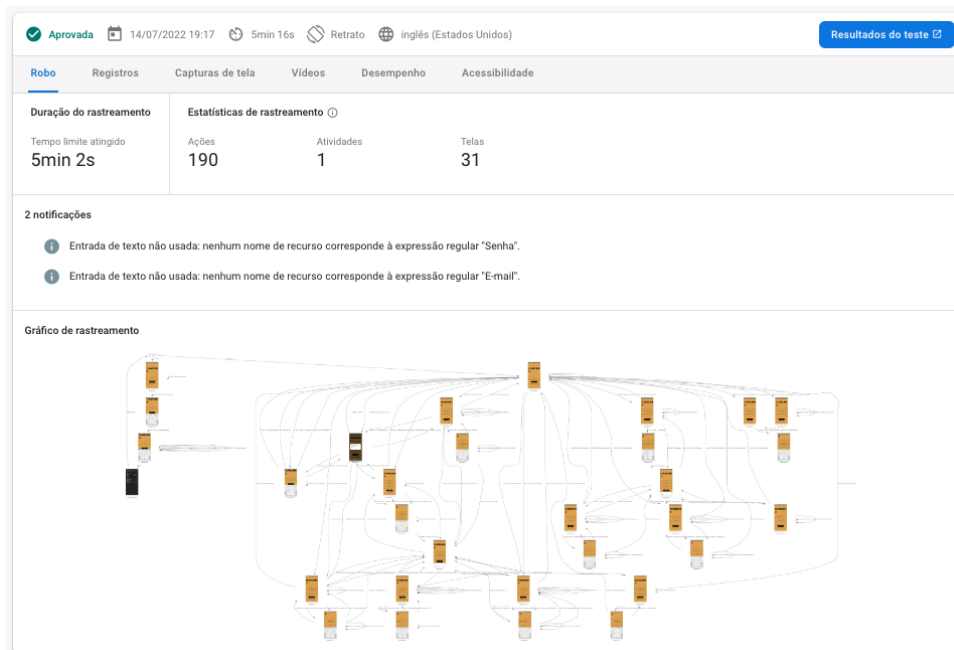
Durante todo o desenvolvimento do aplicativo Flanelinha, foram realizados testes unitários para garantir o funcionamento da aplicação na maior cobertura possível, também foram realizados alguns testes de exploração aleatória por robôs em dispositivos físicos para testar a estabilidade e velocidade da aplicação em diversos dispositivos. Esse recurso de teste automatizado está dentro das opções disponibilizadas pela plataforma do Firebase, onde há celulares físicos para realizar a instalação dos aplicativos a serem testados, após o robô concluir o teste realizado, são geradas algumas estatísticas de desempenho. Na figura 24 está a relação de modelos que foram testados de modo automatizado e as versões do sistema de cada um deles, na figura 25 está o painel principal do teste no dispositivo disponibilizado pela plataforma, contendo o tempo testado, as ações e a quantidade de telas abertas. Na figura 26 mostra o exemplo de algumas informações de desempenho do teste automatizado nos dispositivos físicos.

Figura 24 – Relação de dispositivos testados por robô

Teste Robo ① 14/07/2022 19:17		
<span style="color: green;">✔</span> Aprovada	Falha <span style="color: red;">⚠</span> 0	Estáveis <span style="color: green;">✔</span> 5
		Total de dispositivos 5
Detalhes do dispositivo		
<span style="color: green;">✔</span> SM-G950F, API nível 28 Dispositivo	inglês (Estados Unidos) Localidade	Orientação
<span style="color: green;">✔</span> SM-G981U1, API nível 29 Dispositivo	inglês (Estados Unidos) Localidade	Orientação
<span style="color: green;">✔</span> Pixel 5, API nível 30 Dispositivo	inglês (Estados Unidos) Localidade	Orientação
<span style="color: green;">✔</span> XT1650, API nível 24 Dispositivo	inglês (Estados Unidos) Localidade	Orientação
<span style="color: green;">✔</span> LG-H932, API nível 26 Dispositivo	inglês (Estados Unidos) Localidade	Orientação

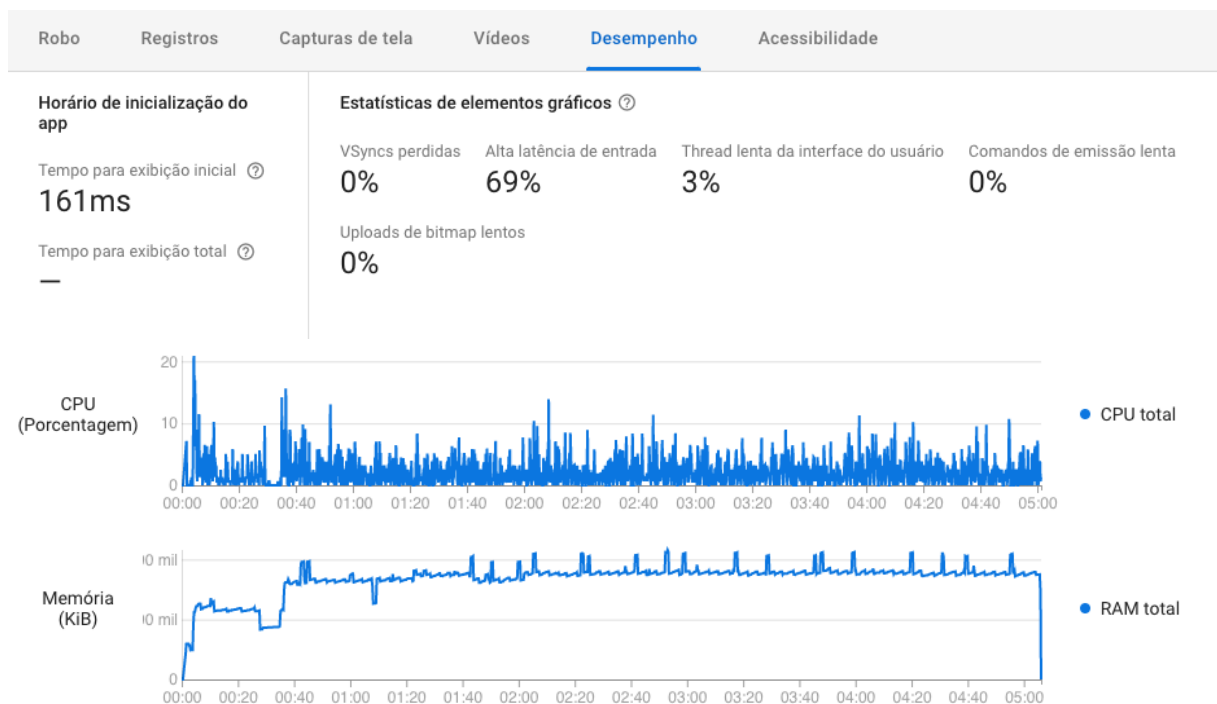
Fonte: Elaborado pelo autor (2022).

Figura 25 – Painel principal do teste automatizado



Fonte: Elaborado pelo autor (2022).

Figura 26 – Painel de desempenho do teste automatizado



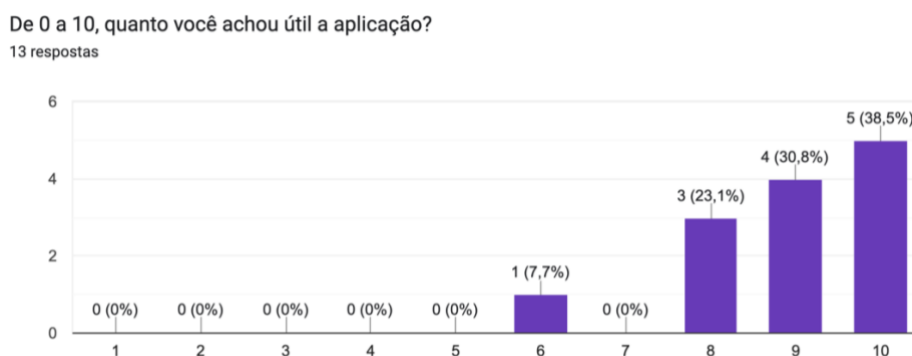
Fonte: Elaborado pelo autor (2022).



Também foram realizados testes de aceitação com usuários reais, sendo disponibilizado para testes a versão do sistema Android. Esses resultados serão apresentados a seguir, baseados no questionário de avaliação enviado a cada testador.

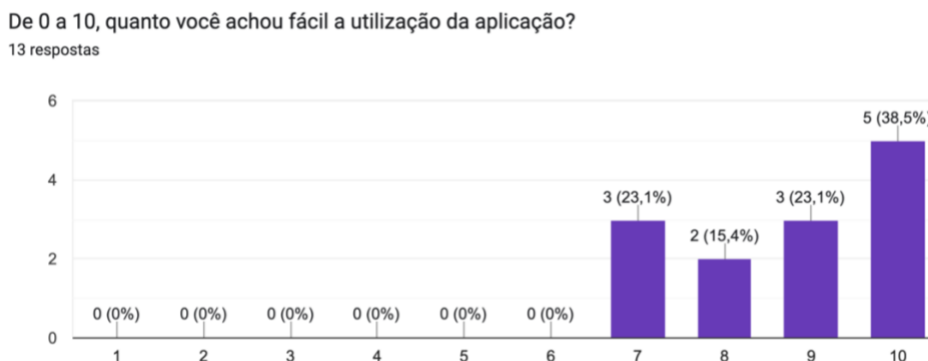
Dentre as pessoas que testaram o aplicativo, foram obtidas 13 respostas. Havia 3 perguntas cuja resposta era em uma escala de 0 a 10, a primeira era relacionada à utilidade da aplicação, onde 0 representava pouca utilidade e 10 muita utilidade, na figura 27 é demonstrado o gráfico com as respostas obtidas nessa questão, a média aritmética resultante das respostas foi 8,92. A segunda pergunta foi relacionada à facilidade de utilização do aplicativo, seguindo a mesma lógica da primeira pergunta, a figura 28 representa o gráfico gerado pelas respostas, a média aritmética delas foi 8,76. A última pergunta com resposta escalar, foi de *design*, se foi agradável ao usuário, na figura 29 está representado o gráfico relativo às respostas da última pergunta, a média aritmética foi 9.

Figura 27 – Gráfico das respostas sobre utilidade da aplicação

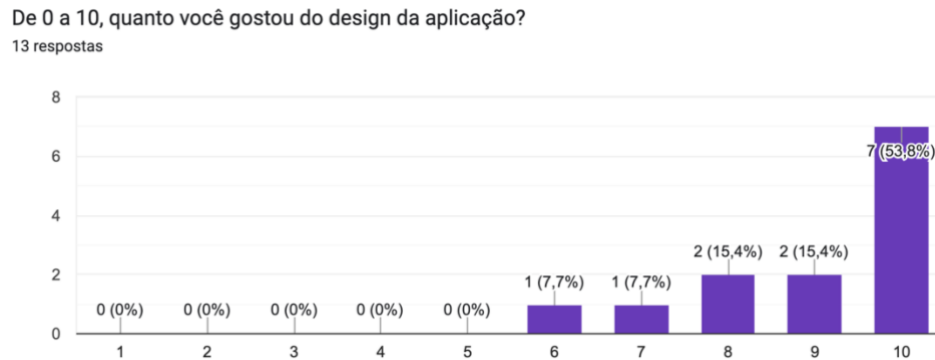


Fonte: Elaborado pelo autor (2022).

Figura 28 – Gráfico das respostas sobre facilidade de utilização da aplicação



Fonte: Elaborado pelo autor (2022).

Figura 29 – Gráfico das respostas sobre o *design* da aplicação

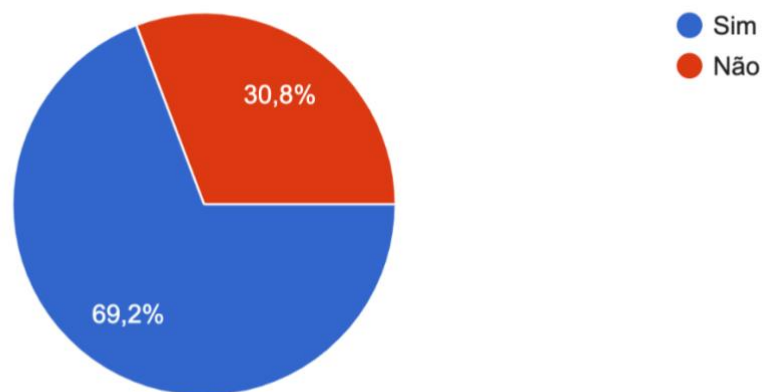
Fonte: Elaborado pelo autor (2022).

O questionário contava com mais duas perguntas de múltipla escolha, se o usuário conseguiu utilizar todas as funcionalidades disponibilizadas para teste e se usaria o aplicativo Flanelinha se fosse lançado comercialmente, nas figuras 30 e 31 estão os gráficos em porcentagem das respostas de cada questão e, respectivamente, em numerais são 9 testadores sim e 4 testadores não na primeira pergunta e 12 testadores sim e um testador não na segunda pergunta.

Figura 30 – Gráfico das respostas sobre as funcionalidades da aplicação

Você conseguiu utilizar todas as funcionalidades disponíveis no aplicativo?

13 respostas

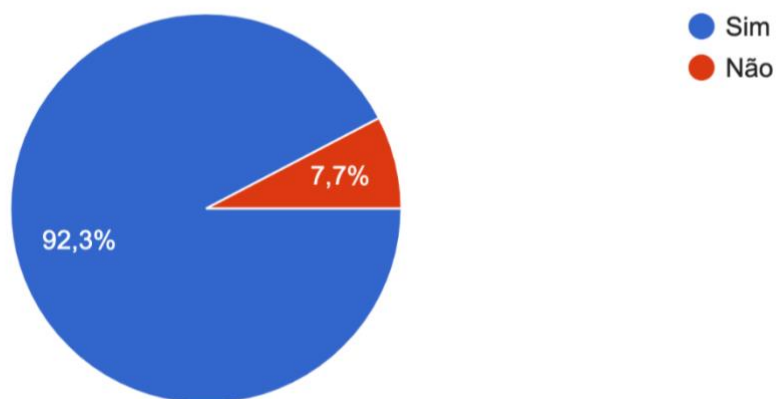


Fonte: Elaborado pelo autor (2022).

Figura 31 – Gráfico das respostas sobre a utilização comercial da aplicação

### Você usaria o aplicativo Flanelinha se fosse lançado comercialmente?

13 respostas



Fonte: Elaborado pelo autor (2022).

A última pergunta do questionário foi uma abertura às sugestões para a aplicação, o resultado está demonstrado na figura 32.

Figura 32 – Respostas da pergunta sobre sugestões para a aplicação

Caso o aplicativo futuramente venha ser lançado no mercado, poderia conter estilo um manual, que explique suas funções, para assim ficar ainda mais fácil o manuseio.
Não ,está perfeito !
Nas transições, a tela tem certo flash. Creio que se isso fosse amenizado ficaria mais agradável a utilização.
- Barra de pesquisa por vagas na tela inicial. - Historico de locações. - Seria interessante se o checkin e o checkout fossem feitos pelo app e conseqüentemente o calculo total de horas e custo.
A ideia do aplicativo é extremamente interessante, o design ficou excelente também. Porém, algumas funções ficaram um pouco difíceis de serem usadas no aplicativo - acredito que isso ocorreu pelo fato de ainda estar em desenvolvimento. De qualquer forma, o app tem um potencial enorme e acredito que é uma proposta necessária para o dia a dia de todos.
Nenhuma
Introdução de um mecanismo de ranqueamento dos usuarios, seja o proprietário da vaga ou o dono do veículo. A fim de aumentar o nivel de confiança dos usuários e fazer com que eles deixem o seu carro [ou abram sua garagem] com tranquilidade. Equivalente às 'estrelas' do Uber.

Fonte: Elaborado pelo autor (2022).

## 6 CONSIDERAÇÕES FINAIS

O presente trabalho demonstrou o desenvolvimento de uma aplicação móvel para os sistemas Android e iOS que possibilita a busca e divulgação de vagas de estacionamento em propriedades particulares que passem inutilizadas por grandes períodos durante o dia.

A finalidade da aplicação era facilitar o uso dos seus veículos e a movimentação das pessoas durante suas rotinas diárias, dando a possibilidade de encontrarem um lugar seguro para estacionarem seus veículos por um preço que fosse considerado acessível, visto que o resultado do questionário aplicado apontou uma insegurança por parte dos usuários de estacionarem em vias públicas, porém, estacionamentos particulares eram uma alternativa inviável financeiramente para a maior parte do público questionado.

Por esta razão foi iniciado o desenvolvimento do aplicativo para tentar criar uma solução para o problema relatado. Baseado nas respostas do questionário, decidiu-se as funcionalidades da aplicação e após começou o desenvolvimento do aplicativo em si, para isso foi utilizada a tecnologia React Native que é um *framework* Javascript conectada a um banco de dados para armazenar as informações da aplicação.

Após desenvolver a aplicação, foram feitos testes automatizados e de aceitação para avaliar o desempenho, funcionalidades, utilidade e *design* do aplicativo, foi possível concluir que para a maioria do público que realizou o teste, o resultado atingiu os objetivos de qualidade desejados para a finalidade do aplicativo.

Para o trabalho futuro com a aplicação, serão acrescentados conforme necessidade e coerência com a proposta do sistema, as sugestões inseridas no questionário de avaliação respondido pelos usuários que realizaram os testes e sugestões feitas pela banca avaliadora dos presente trabalho, tais como gerenciamento de tempo do aluguel da vaga, sistema de classificação dos usuários interno da plataforma, mudança da cor no fundo dos campos de texto, regulagem do raio de busca das vagas e formatação dos valores anunciados.

## 7 REFERÊNCIAS

ANDROID. **A Plataforma que Redefine o Impossível**. [S. l.], 2022. Disponível em: [https://www.android.com/intl/pt-BR\\_br/](https://www.android.com/intl/pt-BR_br/). Acesso em: 3 jul. 2022.

AWS. AppSync. 2022a. Disponível em: <https://aws.amazon.com/pt/appsync/>. Acesso em: 02 de jul. 2022.

AWS. DynamoDB. 2022b. Disponível em: <https://aws.amazon.com/pt/dynamodb/>. Acesso em: 02 de jul. 2022.

AWS. Amazon S3. 2022c. Disponível em: <https://aws.amazon.com/pt/s3/>. Acesso em: 02 de jul. 2022.

AWS. Visão geral da Amazon Web Services. 2022d. Disponível em: [https://docs.aws.amazon.com/pt\\_br/whitepapers/latest/aws-overview/introduction.html](https://docs.aws.amazon.com/pt_br/whitepapers/latest/aws-overview/introduction.html). Acesso em: 02 de jul. 2022.

AWS. Computação sem servidor. 2022e. Disponível em: <https://aws.amazon.com/pt/serverless/>. Acesso em 02 de jul. 2022.

BATTISTI, Júlio. **SQL Server 2000: Administração e Desenvolvimento – Curso Completo**. 2. Ed. Rio de Janeiro: Axcell Books, 2001.

COULOURIS, G. et al. **Sistemas Distribuídos – 5ed: Conceitos e Projeto**. [s.l.] Bookman Editora, 2013.

CRUZ, Franciane *et al.* **Mensuração da satisfação dos usuários do sistema municipal de estacionamento rotativo pago**. [S. l.], 2015. Disponível em: <https://www.scielo.br/j/urbe/a/zf5fxQQm3H3Gpt3xHT46q9x/abstract/?lang=pt>. Acesso em: 4 abr. 2022.

EISENMAN, Bonnie. **Learning React Native: Building Native Mobile Apps with JavaScript**. [S. l.: s. n.], 2016.

FIREBASE. Google. 2022. Disponível em: <https://firebase.google.com/> Acesso em: 02 de jul. 2022.

GERHARDT, Tatiana; SILVEIRA, Denise. **Métodos de Pesquisa**. [S. l.: s. n.], 2009.

GRAPHQL. A query language for your API. 2022. Disponível em: <https://graphql.org>. Acesso em: 02 de jul. 2022.

HENDRICKSON, Scott; STURDEVANT, Stephen; HARTER, Tyler; VENKATARAMANI, Venkateshwaran; ARPACI-DUSSEAU, Andrea C.; ARPACI-DUSSEAU, Remzi H. **Serverless Computation with OpenLambda. Serverless Computation with OpenLambda**, [s. l.], 2016. Disponível em:

[https://www.usenix.org/system/files/conference/hotcloud16/hotcloud16\\_hendrickson.pdf](https://www.usenix.org/system/files/conference/hotcloud16/hotcloud16_hendrickson.pdf). Acesso em: 2 jul. 2022.

IBGE - INSTITUTO BRASILEIRO DE GEOGRAFIA E ESTATÍSTICA (Brasil). Frota Veicular. Brasil: DENATRAN, 2010. Disponível em: <https://cidades.ibge.gov.br/brasil/pesquisa/22/28120?ano=2010>. Acesso em: 28 mar. 2022.

IBGE - INSTITUTO BRASILEIRO DE GEOGRAFIA E ESTATÍSTICA (Brasil). Frota Veicular. Brasil: DENATRAN, 2020. Disponível em: <https://cidades.ibge.gov.br/brasil/pesquisa/22/28120?ano=2020>. Acesso em: 28 mar. 2022.

LECHETA, Ricardo R. **Desenvolvendo para iPhone e iPad**: Aprenda a Desenvolver Aplicações Utilizando o iOS SDK. [S. l.: s. n.], 2014.

LIMA, Aline Francielle dos Anjos; GARCIA, Fabrício Wickey da Silva; SERUFFO, Marcos César da Rocha; SOARES, Zilda Ramalho. Desenvolvimento e Avaliação de um Módulo Educacional para o Ensino da Língua Portuguesa: Um Estudo Baseado em Variações Linguísticas. *In: Desenvolvimento e Avaliação de um Módulo Educacional para o Ensino da Língua Portuguesa: Um Estudo Baseado em Variações Linguísticas*. [S. l.], 2013. Disponível em: [https://www.researchgate.net/publication/337991059\\_II\\_Congresso\\_Brasileiro\\_de\\_Recursos\\_Digitais\\_na\\_Educacao/link/5eae242f45851592d6b4acd0/download](https://www.researchgate.net/publication/337991059_II_Congresso_Brasileiro_de_Recursos_Digitais_na_Educacao/link/5eae242f45851592d6b4acd0/download). Acesso em: 2 jul. 2022.

MARTIN, J. Agile Project Management with Scrum. 2009.

MOZILLA CORPORATION. JavaScript. *In: JavaScript*. [S. l.], 2022. Disponível em: <https://developer.mozilla.org/pt-BR/docs/Web/JavaScript>. Acesso em: 3 jul. 2022.

OBJECT MANAGEMENT GROUP (Estados Unidos). Unified Modeling Language. 2.5.1. [S. l.], 2017. Disponível em: <https://www.omg.org/spec/UML/2.5.1/PDF>. Acesso em: 28 mar. 2022.

POLIT, Denise; BECK, Cheryl. **Fundamentos de Pesquisa em Enfermagem**. 2. ed. [S. l.: s. n.], 2016.

REACT JSX. [S. l.], 2022. Disponível em: [https://www.w3schools.com/react/react\\_jsx.asp](https://www.w3schools.com/react/react_jsx.asp). Acesso em: 16 jul. 2022.

RESENDE, Leonardo. **Projeto conceitual de um equipamento mecanizado para estacionar veículos leves de baixo custo**. [S. l.], 2020. Disponível em: <https://repositorio.ufu.br/handle/123456789/30212>. Acesso em: 5 abr. 2022.

RISCO. *In: DICIO*, Dicionário Online de Português. Significado de Automóvel. Brasil, 2021. Disponível em: <https://www.dicio.com.br/automovel/>. Acesso em: 28 mar. 2022.

SANTOS, Marcos *et al.* **Uma nova maneira de estacionar veículos de passeio em grandes centros urbanos: proposta do aplicativo "minha vaga"**. [S. l.], 2018. Disponível em: [https://www.researchgate.net/publication/325619021\\_Uma\\_nova\\_maneira\\_de\\_estacionar\\_vei](https://www.researchgate.net/publication/325619021_Uma_nova_maneira_de_estacionar_vei)

culos\_de\_passeio\_em\_grandes\_centros\_urbanos\_proposta\_do\_aplicativo\_minha\_vaga.  
Acesso em: 5 abr. 2022.

SECRETARIA DE SEGURANÇA PÚBLICA (Rio Grande do Sul). Indicadores Criminais.  
Rio Grande do Sul: Secretaria de Segurança Pública, 2021. Disponível em:  
<https://ssp.rs.gov.br/indicadores-criminais>. Acesso em: 28 mar. 2022.

VERAS, Manoel. **Arquitetura de Nuvem Amazon Web Services (AWS)**. [S. l.: s. n.], 2013.