Direto ao Ponto – App colaborativo do transporte coletivo usando o Firebase

Antoni Sganzerla[1], Ramon Lummertz[2]

[1]Acadêmico do Curso de Análise e Desenvolvimento de Sistemas – Universidade Luterana do Brasil (ULBRA) Campus Torres - RS

[2]Professor dos Cursos de Computação – Universidade Luterana do Brasil (ULBRA) Campus Torres – RS

antoni.sganzerla.dev@gmail.com, ramonsl@gmail.com

Abstract. *This article presents the Direto ao Ponto a collaborative application developed for Android platform that aims to provide information quickly and accurately about public transport services based on the GPS location of the device and extensive use of the services of Firebase. The application was developed using Java and follows the concepts of the SCRUM methodology.*

Resumo. *Este artigo apresenta o Direto ao Ponto, um aplicativo colaborativo desenvolvido para plataforma Android que tem como objetivo fornecer informações de maneira rápida e precisa sobre os serviços de transporte público, baseando-se na localização do GPS do dispositivo e com amplo uso dos serviços do Firebase. O aplicativo foi desenvolvido utilizando Java, e segue os conceitos da metodologia SCRUM.*

1. Introdução

A utilização do transporte coletivo ou público é uma das principais soluções para pôr fim aos congestionamentos no trânsito urbano. Além disso, apresenta-se como um modo sustentável de substituir o automóvel, proporcionando uma alternativa de locomoção mais econômica em relação ao uso do transporte individual. Segundo uma pesquisa sobre transporte público, encomendada pela Confederação Nacional da Indústria (CNI), realizada em 2014, cerca de 25% da população do Brasil usa esse meio diariamente para se locomover para escola ou trabalho (MOREIRA, 2015).

Na tentativa de facilitar a utilização desse tipo de serviço, foram desenvolvidos alguns aplicativos. Tais ferramentas, porém, propõem soluções focadas em grandes metrópoles, deixando cidades menores sem uma opção de *software*. É nesse terreno que o aplicativo Direto ao Ponto tem seu alvo, objetivando, primeiramente, facilitar a utilização dos serviços de transporte público por fornecer aos seus usuários informações práticas e de fácil acesso, tendo como referência a localização do passageiro, o que poderá evitar transtornos e perdas de tempo.

Por se tratar de uma aplicação colaborativa, as informações do sistema serão mantidas pela própria comunidade. Desse modo, a base de dados estará sempre

atualizada, tendo como principal trunfo o *Firebase*[1] e a indenpendência das prestadoras de serviço de transporte público.

Este trabalho foi desenvolvido visando a cidades que não são atualmente servidas pelo *Google now*[2], ou similar. Para aplicar a solução proposta neste artigo, foi tomado como base o município de Torres, Rio Grande do Sul. Este artigo foi dividido em 5 seções, a seção 1 traz uma breve introdução sobre o tema, a seção 2 aborda sobre a mobilidade urbana, a importância do *Firebase* e o *software* proposto, a seção 3 explana sobre as tecnologias que foram utilizadas para construir o projeto, a seção 4 detalha como foi feita a aplicação da metologia e o desenvolvidometno dos *sprints* e por fim a seção 5 traz as conclusões deste artigo.

2. Fundamentação Teórica

A mobilidade urbana é um atributo associado às cidades. Refere-se à capacidade de deslocamento de indivíduos a fim de suprir suas necessidades para a realização das atividades cotidianas, tais como: trabalho, educação, saúde, lazer e cultura. Nos últimos vinte e cinco anos, grande parte das cidades brasileiras têm vivenciado uma crise de mobilidade urbana decorrente da alta taxa de crescimento populacional, aliada ao baixo investimento no setor de transportes (ICETRAN, 2017).

Esse fator tem contribuído para que a qualidade do serviço de transporte coletivo urbano venha diminuindo sensivelmente ao longo dos anos. Isso tem dificultado o cotidiano de quem depende de tal serviço para se locomover, pois, na atual conjuntura, o tempo é um fator essencial ao desenvolvimento e as pessoas não podem mais desperdiçá-lo procurando uma informação cuja fonte não seja confiável, precisa e de fácil acesso.

O cenário descrito é a realidade de quem utiliza o transporte coletivo urbano em diversas cidades do Brasil, principalmente em centros urbanos com o número de habitantes relativamente pequeno, como é o caso da cidade de Torres-RS. Em tais localidades, caso uma pessoa precise utilizar o transporte coletivo do município, logo surgem questões muito corriqueiras como: Qual ônibus pegar? Onde pegar? Em qual horário ele irá passar em determinada parada?

O fato de o usuário não conseguir essas informações de maneira rápida e precisa, torna-se um problema porque, geralmente, não há disponibilidade de fontes de informações confiáveis ou de fácil acesso. Para obtê-las, é comum o passageiro recorrer a pessoas que, na maioria das vezes, não conseguem fornecer a informação adequada. Outra possibilidade é o usuário do transporte coletivo valer-se de cartazes expostos em pontos de parada os quais nem sempre estão atualizados ou contêm as informações necessárias.

Os fatos apresentados geram transtornos e perda de tempo, provocando irritabilidade e insatisfação aos usuários que, devido à falta de informações corretas, muitas vezes ficam à mercê da própria sorte para utilizar um serviço tão essencial. Tendo em vista solucionar esse problema, o aplicativo proposto disponibilizará um mecanismo simples e objetivo de fornecimento de informações, a fim de permitr que os usuários saibam exatamente o que fazer para chegar ao seu destino.

[1] O Firebase usa a infraestrutura do Google e é dimensionado automaticamente, para atender à demanda dos usuários

[2] Google Now é um assistente pessoal inteligente, disponível para o sistema operacional Android

Para tornar essa solução possível será feito uso da tecnologia *Firebase*, uma plataforma *BaaS* (*Backend as a Service*) adquirida pela *Google* em 2014. Com ela é possível desenvolver para *iOS*, *Android* ou *Web* já que todo o *Backend* será configurado e gerenciado pelo *Firebase* (MORIBE, 2016).

BaaS é um serviço de computação em nuvem que serve como *middleware*[3]. Ele fornece aos desenvolvedores uma forma para conectar suas aplicações *mobile* e *web* a serviços de nuvem a partir de *API*s[4] e *SDK*s[5]. Esse tipo de serviço auxilia os desenvolvedores a acelerar a criação de aplicações *web* e *mobile*. Em vez de codificar o *backend* inteiro, o desenvolvedor usa o *BaaS* para criar as *API*s e conectá-las às aplicações, desta forma a infraestrutura do lado do servidor é abstraída completamente permitindo ao desenvolvedor se concentrar à experiência de usuário, o *frontend* (BATSCHINSKI, 2016).

O *Firebase* possui diversos recursos, dos quais destacamos os seguintes:

- Autenticação: permite implementar um sistema de autenticação segura e de melhor experiência de *login* para os usuários finais. Oferece uma solução de identidade completa, compatível com contas de *e-mail* e senha como *login* do *Google*, *Twitte*r, *Facebook* e *GitHub*.
- *Realtime Database*: banco de dados *NoSQL* hospedado na nuvem. A partir dele os dados armazenados são sincronizados entre os usuários em tempo real. Os dados são armazenados em uma espécie de árvore *JSON*.
- *Cloud Messaging*: recurso que oferece uma conexão confiável entre servidor e dispositivos para enviar e receber mensagens e notificações no Android, *iOS* e *Web,* com baixo consumo de bateria. As notificações podem ser enviadas instantaneamente ou em horários pré-definidos.
- Monitoramento e Desempenho: mecanismo que fornece *insights* sobre o desempenho do *app*, como tempo de inicialização e latência da rede à qual o usuário está conectado.
- *Crash Reporting:* recurso que ajuda a diagnosticar e corrigir problemas no *app;* os erros são agrupados por categorias, as versões do aplicativo com falhas e modelo de dispositivo afetado são detalhados.
- *AdMob*: plataforma de publicidade móvel que pode ser usada para gerar receita com sua aplicação. Os anúncios podem ser exibidos como *banners*, anúncios intersticiais ou em vídeo (FIREBASE, 2017)**.**

Além disso, existem algumas vantagens em utilizar um serviço *BaaS*, como o *Firebase*. Entre os aspectos positivos aponta-se: a isenção de responsabilidade, pois toda a segurança dos dados fica a cargo da *Google*; a alta disponibilidade, pois os servidores da *Google* possuem extrema confiabilidade; o ganho de performance, pois com o *realtime database* a conexão entre aplicação e servidor mantém-se constante (*three-way bindig);* a redução de custos, pois não é necessário manter uma estrutura física com servidores e, por fim, a economia de tempo, pois são disponibilizadas diversas *API'*s que facilitam, por exemplo, a implementação de um sistema de autenticação.

[3] Programa computacional que faz a mediação entre outros softwares.

[4] Conjunto de rotinas e padrões estabelecidos por um software para utilização das suas funcionalidades por aplicativos sem necessidade de conhecer detalhes da implementação.

[5] Conjunto de ferramentas de desenvolvimento de software que permite a criação de aplicativos.

Com base nesses dados, afirma-se haver diversos fatores que tornam o *Firebase* uma solução melhor se comparada a um *Webservice* construído a partir de uma *API* própria. Isso ocorre porque este costuma tomar bastante tempo para seu desenvolvimento e, caso não seja bem planejado, poderá tornar-se instável e inseguro, opondo-se ao serviço da *Google* que já está consolidado no mercado.

Existem diversas aplicações sólidas que utilizam o Firebase, entre elas pode-se citar o *Duolingo*, importante aplicativo para quem quer aprender novos idiomas, Trivago que permite comparar preços de hotéis e fazer reservas, Shazam aplicativo muito utilizado para identificar músicas e obter suas letras, dentre muitas outras.

O *Firebase* apresenta alguns planos para utilização, como, por exemplo, o *Spark* que é gratuito e permite até 100 conexões simultâneas ao *Realtime Database*, 1GB de armazenamento e envio de 125 mil notificações por mês. Existe também o plano *Flame* que custa US$ 25 por mês e aumenta consideravelmente os limites de uso. Há, ainda, o plano *Blaze,* cuja mensalidade é cobrada de acordo com a utilização dos recursos (FIREBASE, 2017).

2.1. *Softwares* existentes

Há certa diversidade de aplicações disponíveis que se ajustam a alguns dos problemas gerados pela falta de informação, porém nenhum deles atende às pequenas cidades como é o caso de Torres. Foram pesquisados os aplicativos que possuem maior relevância no Brasil a fim de viabilizar uma análise e, a partir dela, formar-se uma base para o desenvolvimento de uma aplicação com foco para pequenos municipios. Para isso, foram selecionados os aplicativos *Moovit*, *CittaMmobi* e Trafi.

2.1.1. Moovit

De acordo com o site, o *Moovit* é o aplicativo mais utilizado no mundo quando se trata de transporte público, contando com mais de 55 milhões de usuários. Ele atende a mais de 1.200 cidades em cerca de 75 países. O serviço abrange ônibus, estações de trem e metrô. Na *Google Play,* seu serviço está avaliado com a nota 4,3 num *ranking* cuja nota máxima é 5. Esse *app* está disponível de forma gratuita para *download* (MOOVIT, 2017).

Sua *interface* é semelhante ao *Google Maps* e para acesso não é necessário possuir cadastro. Para utilização, o usuário deve informar uma origem e um destino. A origem pode ser a sua localização ou um ponto selecionado no mapa, assim como o destino. A partir desses dados, o *app* verifica se existe alguma rota disponível e mostra no mapa o caminho até o ponto de parada ou estação. Um requisito para funcionamento desse aplicativo é a constante conexão com a internet.

2.1.2. CittaMobi

O *CittaMobi* é um aplicativo desenvolvido para uso apenas no Brasil e atende, atualmentea, a cerca de 17 cidades em 10 estados. O serviço é exclusivo para ônibus e seu foco é auxiliar os usuários no deslocamento urbano. No momento está avaliado com a nota 4,3 no *Google Play,* num *ranking* cuja nota máxima é 5. Tal sistema também está disponível gratuitamente para *download* (CITTAMOBI, 2017).

Além do aplicativo, o serviço também pode ser utilizado na versão *web* e em painéis instalados nos terminais. Para utilização, o usuário precisa estar conectado à internet e informar o destino; o *app* se encarrega de verificar linhas ou ônibus que fazem o trajeto desejado e informar, em tempo real, a previsão de chegada ao ponto de parada. Um de seus diferenciais é o fato de possuir um filtro que permite mostrar no mapa apenas

ônibus adaptados para cadeirantes. Seu principal problema é o fato de não possuir todas as linhas disponíveis cadastradas.

2.1.3. Trafi

Trafi é uma solução para usuário de ônibus, trem e metrô. No momento ele atende apenas em duas cidades brasileiras: São Paulo e Rio de Janeiro. Sua principal proposta é permitir a sua utilização de maneira *offline*. O aplicativo encontra-se disponível para *download* de maneira gratuita na *Google Play* e recebeu nota 4,3 na avaliação dos usuários (SALUTES, 2015).

Esse *app* não solicita cadastro para sua utilização. Também permite ao usuário definir uma origem e um destino inserindo o endereço ou marcando um local no mapa. Para funcionamento do modo *offline* é preciso habilitar a opção dentro do aplicativo para que ele armazene os dados no dispositivo.

2.3. *Software* proposto

O *so*ftware proposto neste artigo tem a finalidade de permitir que as informações sobre os ônibus e táxis sejam disponibilizadas a todas as pessoas de forma colaborativa, ou seja, a própria comunidade irá abastecer e manter os dados atualizados. Desta forma o *app* será um mecanismo simples e objetivo para se obter informações sobre o transporte coletivo municipal, de modo a permitir aos usuários saberem exatamente o que devem fazer para chegar ao seu destino. Esse aplicativo será distribuído de forma gratuita na plataforma *Android*.

A tela inicial mostrará um mapa com a localização do usuário e com as paradas de ônibus e pontos de táxis cadastrados. Também estará disponível uma barra de pesquisa para que seja inserido um possível local de destino e um botão para cadastrar um ponto de táxi ou parada de ônibus.

O usuário poderá obter informações selecionando uma parada de ônibus ou ponto de táxi no mapa ou, então, fazendo uma pesquisa ao inserir um local para onde deseja ir.

Nas paradas de ônibus, o usuário obterá uma lista de ônibus cadastrados e terá a possibilidade de cadastrar um novo coletivo. Cada ônibus deverá dispor de uma lista de horários contendo o dia da semana e o horário em que ele passa na parada. O usuário poderá verificar todos os horários e adicionar novos. Os horários têm um sistema de *feedback*, que são os *likes*, ou seja, caso o usuário perceba que o horário está correto ele pode validá-lo com um *like*, no entanto, se estiver incorreto, ele poderá reportar erro, assim o sistema irá removê-lo após muitos usuários reportarem o mesmo erro. O ônibus também possui uma lista de locais que fazem parte de seu itinerário, o usuário pode adicionar ou remover locais deste itinerário.

Nos pontos de táxi o usuário poderá obter uma lista de taxistas cadastrados, contendo o nome e telefone do condutor; terá ainda a possibilidade de adicionar novos taxistas. Em caso de inconsistência nos dados cadastrados, o usuário poderá reportar erro.

O retorno da pesquisa será uma lista em que constem os ônibus que percorrem esse itinerário, os dados sobre os pontos de suas paradas, o horário mais próximo em que irão passar em determinada parada e o roteiro contendo todos os locais que fazem parte de seu trajeto. Os resultados serão ordenados pela menor distância entre a localização do usuário e os pontos de paradas do ônibus em questão.

Após a pesquisa, caso deseje, o usuário poderá visualizar no mapa o percurso que ele deverá percorrer até chegar ao ponto de ônibus onde irá ocorrer o embarque. Para

melhor uso dessa funcionalidade foi feita uma integração precisa com o uso do GPS, para que seja possível acompanhar o usuário no mapa conforme ele se locomove.

Para acesso ao sistema é necessário fazer *login* usando a conta do *Google Plus*, para que não haja perda de tempo com formulários. Desta forma é possível identificar o usuário e manter seus dados, caso ele troque de dispositivo.

3. Tecnologias e Ferramentas

Neste capítulo são descritas as tecnologias e ferramentas utilizadas no planejamento e desenvolvimento do *software* proposto.

3.1. Metodologia Scrum

O Scrum, figura 1, é um *framework* para gerenciamento de projetos ágeis que pode ser utilizado para o planejamento, gerenciamento e desenvolvimento de qualquer produto, principalmente por se tratar de um *framework* interativo e incremental. Baseia-se na flexibilidade dos resultados e prazos, equipes pequenas, revisões frequentes e colaboração entre a equipe. O propósito principal é controlar processos, mantendo o foco na entrega do valor de um negócio no menor tempo possível. Nessa metodologia os projetos são divididos em ciclos repetitivos e curtos, para que possam ser modificados e adaptados para corrigir os desvios. Esses ciclos são chamados de *Sprints* CRUZ, 2013).

O *Sprint* pode ser considerado o principal evento do *Scrum* e deve ser executado em um determinado período de tempo, geralmente entre 2 e 4 semanas, porém, antes de executá-lo, é necessário definir quais são os requisitos e funcionalidades (itens) do produto a ser desenvolvido. O resultado desse levantamento é um artefato conhecido como *Product Backlog.* É a partir desse documento que são determinados quais itens serão atendidos pela *Sprint,* gerando um novo artefato chamado de *Sprint Backlog*. (BUILDER, 2016)

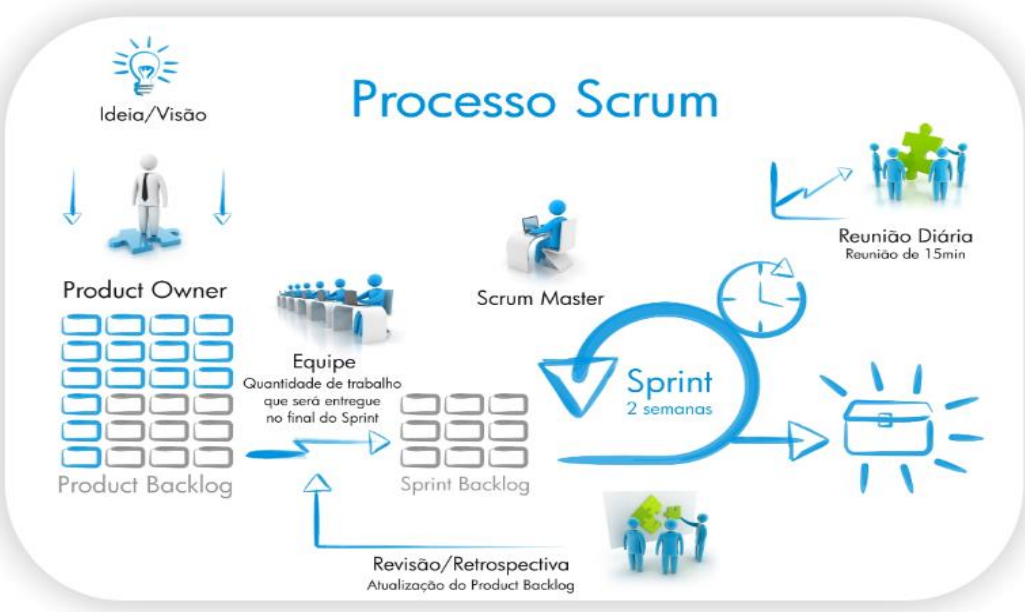


Figura 1. Demonstração do funcionamento da Metodologia Scrum

Segundo Cruz (2013), os papéis e suas respectivas responsabilidades que compõem o *Time Scrum* são:

- ***Scrummaster*** – responsável por garantir que o *time* aplique as práticas e valores do *Scrum,* mantendo o processo em pleno funcionamento e eliminando quaisquer impedimentos que possam interferir no objetivo.
- ***Product Owner (PO)*** – responsável por entender o negócio do produto, entregar valor ao cliente e determinar as prioridades a serem desenvolvidas no *Product Backlog*. É conhecido como o Dono do Produto.
- **Time** – time de desenvolvedores responsáveis por transformar o *Backlog* do Produto em incrementos e funcionalidades que possam ser entregues ao cliente.

Pelo fato de se tratar de uma metodologia flexível a mudanças de requisitos, por focar na melhoria contínua do desenvolvimento de software e possibilitar o acompanhamento do seu desenvolvimento, o *Scrum* foi a metodologia escolhida para ser aplicada neste Projeto de Desenvolvimento de *Software*. Para gerenciamento deste projeto, foram utilizados apenas alguns conceitos e artefatos desta metodologia, como o *Product Backlog*, *Sprint Backlog* e as *Sprints*, portanto os papéis serão desempenhados pelo autor do projeto.

3.2. UML

Ramos (2006, p. 8) afirma que "a UML (Linguagem de Modelagem Unificada) é uma linguagem que serve para especificar, construir, visualizar e documentar os artefatos de um sistema de *software*".

A UML, como linguagem, provê um vocabulário e um conjunto de regras para combinar os elementos dessa linguagem, focando nos elementos conceituais e físicos que representam um sistema. Essa linguagem define uma gramática padrão que auxilia as pessoas a criar e ler documentos UML (MARTINS, 2010).

Para melhor compreensão das ações do *software* desenvolvido, foi utilizado o Diagrama de Caso de Uso que, segundo Martins (2010), é uma documentação que representa a interação entre o sistema e o usuário a qual compreende um conjunto de ações que provêm um resultado de valor. Também foi feito uso das *User Stories*, uma técnica que auxilia a validar os requisitos sob o ponto de vista do usuário.

3.3. Java

A linguagem de programação Java representa uma linguagem simples, orientada a objetos, *multithread*, interpretada, neutra de arquitetura, portável, robusta e segura. A tecnologia *Java* é composta de uma linguagem de programação e de uma plataforma (API e máquina virtual) (MENDES, 2009).

3.4. Android

O *Android* é um sistema operacional baseado no *Linux*, teve seu desenvolvimento iniciado em 2003 pela empresa *Android Inc*. e, em 2005, foi adquirida pela *Google* (MONTEIRO, 2013). De acordo com o site *NetMarketShare.com* o *Android* detém 63,66% do *market share* mundial na categoria de dispositivos móveis.

3.5. Android Studio

O *Android Studio* é o ambiente de desenvolvimento integrado (IDE) oficial para o desenvolvimento de aplicativos *Android* e é baseado no *IntelliJ IDEA* (ANDROID STUDIO, 2017).

3.6. Bitbucket

O *Bitbucket* é um sistema de controle de versão distribuída que facilita o desenvolvimento em equipe. Nessa ferramenta é possível criar repositórios, públicos e privados, de maneira ilimitada para equipes de até 5 colaboradores (BITBUCKET, 2017).

3.7. Balsamiq Mockups 3

O *Software Balsamiq Mockups* tem como objetivo ajudar as pessoas a criarem *softwares*, tornando possível a criação de *Storyboards*, que são os desenhos de projeto para o desenvolvimento das telas que serão utilizadas pelos usuários (BALSAMIQ, 2017).

4. Aplicação da Metodologia

A seguir serão descritas as fases executadas na aplicação da metodologia.

4.1. Pré-planejamento

No início do projeto, foi elaborada uma lista com as funcionalidades do sistema e a descrição dos requisitos por meio do *Product Backlog*, o que pode ser visto na Tabela 1.

Com esses requisitos, foram identificadas a ordem de prioridade baseada na importância e no pré-requisito para o desenvolvimento das demais funcionalidades e a quantidade de horas necessárias para a execução de cada funcionalidade, baseando-se na complexidade da programação.

Tabela 1. *Product Backlog*

ID	DESCRIÇÃO	ESTIMATIVA
1	Conectar aplicação ao Serviços *Firebase*	5h
2	Permitir *Login* com *Google Plus* e estrutura *BD*	13h
3	Mostrar mapa na tela inicial e *Menu Hambúrguuer*	10h
4	Cadastrar paradas de ônibus	10h
5	Cadastrar pontos de táxi	5h
6	Cadastrar ônibus	10h
7	Cadastrar horários	20h
8	Cadastrar locais	20h
9	Cadastrar taxistas	5h

10	Reportar erros	15h
11	*Feedback Like* horários	20h
12	Pesquisar por ônibus por local de destino	30h
13	Visualizar caminho até a parada de ônibus	10h
14	Refinamentos UI com *Material Design* e validações	30h

4.2. Desenvolvimento

Nesta etapa, serão demonstrados o desenvolvimento das funcionalidades do *Product Backlog* e também as tarefas realizadas até a sua conclusão. A implementação do Projeto foi dividida em 3 (três) *Sprints*, organizadas por ordem de prioridade e tempo gasto, em horas, para a realização de cada funcionalidade.

O Diagrama de Caso de Uso foi desenvolvido de forma incremental, porém demonstrado somente em sua fase final. Para uma melhor visualização e compreensão geral da aplicação, ele está disponível no Anexo 1 deste artigo.

Para melhor entendimento dos requisitos e funcionalidades, foram desenvolvidas *User Stories*, que estão disponíveis no Anexo 2 deste artigo.

4.2.1. Sprint 1 - Login, Mapa e Cadastro de Paradas de Ônibus e Pontos de Táxi.

Na *Sprint* 1, foram realizadas as implementações das funcionalidades descritas no *Sprint Backlog*, Tabela 2, relacionadas à possibilidade de *login* usando a conta do *Google Plus*, conexão com o *Firebase*, suporte ao uso do *Google Maps* dentro da aplicação, navegação entre telas por meio do *Menu Hambúrguer* e possibilidade de cadastro de uma parada de ônibus e ponto de táxi. Nessa *sprint* os *Mockups* foram realizados conforme a figura 2.

Tabela 2. *Sprint Backlog* da *Sprint* 1

ITEM	DESCRIÇÃO	PRIORIDADE	ESTIMATIVA
1	Conectar aplicativo ao serviço autenticação *Firebase* e *Realtime Database*	100	5h
2	Criar tela de *Login* e permitir login com a conta *Google Plus*	100	8h
3	Definir estrutura das árvores de dados do Banco de Dados	100	5h
4	Criar tela Inicial com *Menu Hambúrguer* e mostrar dados perfil do	90	10h

	usuário		
5	Integração com *Api Google Maps* para abrir mapa na tela Inicial	90	20h
6	Criar função para acompanhar movimentação do usuário no mapa	85	10h
7	Cadastrar paradas de ônibus	80	10h
8	Cadastrar pontos de táxi	80	5h
9	Listar paradas de ônibus e pontos de táxi na tela inicial	75	3h

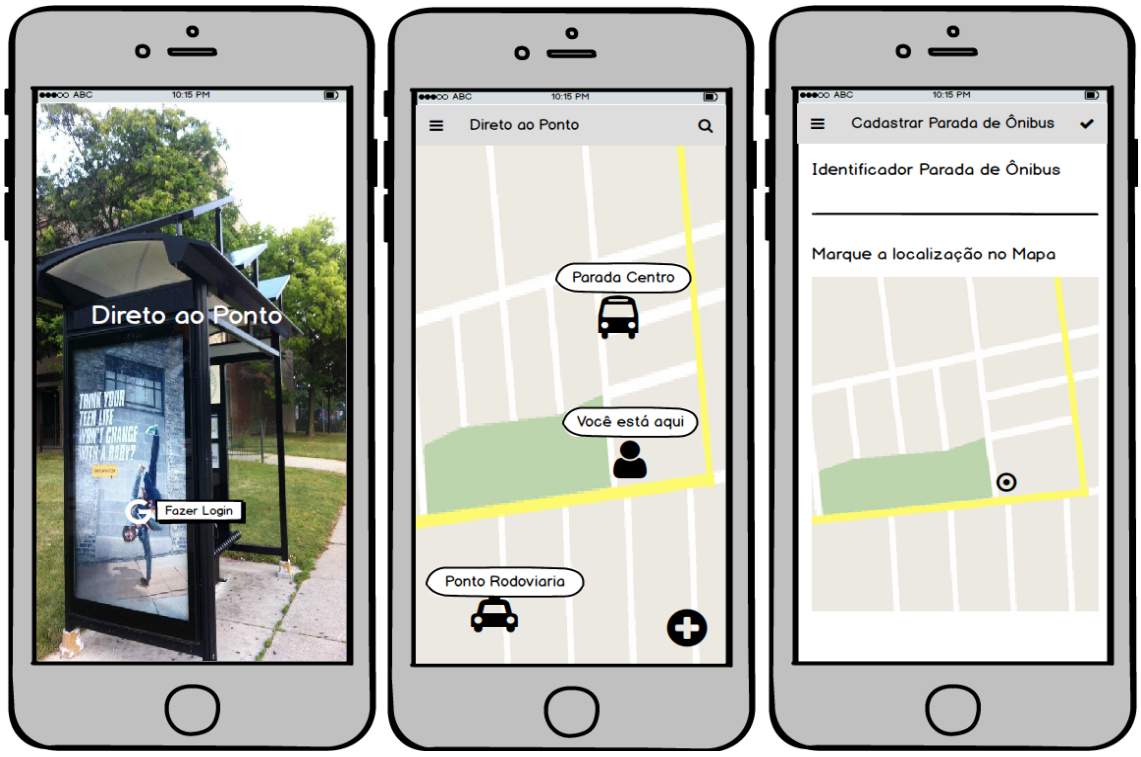


Figura 2. Mockups Tela de Login, Tela Inicial e Tela de Cadastro de Parada de Ônibus

Ao concluir as tarefas desta *Sprint*, os usuários poderão criar um perfil e fazer *login* e *logoff* no aplicativo, conforme a figura 3. Podem visualizar sua localização no mapa por meio de um marcador e acompanhar a sua movimentação conforme caminham. Podem, também, alternar entre as telas por meio do menu e cadastrar uma parada de ônibus ou ponto de táxi informando um nome (identificador) e marcando no mapa a localização dele. Desta forma, a pessoa disponibiliza a todos os usuários a possibilidade de visualizarem no mapa os pontos de parada ou pontos de táxi quando abrirem a aplicação. Os dados são gravados utilizando o recurso *Realtime Database* do *Firebase*.

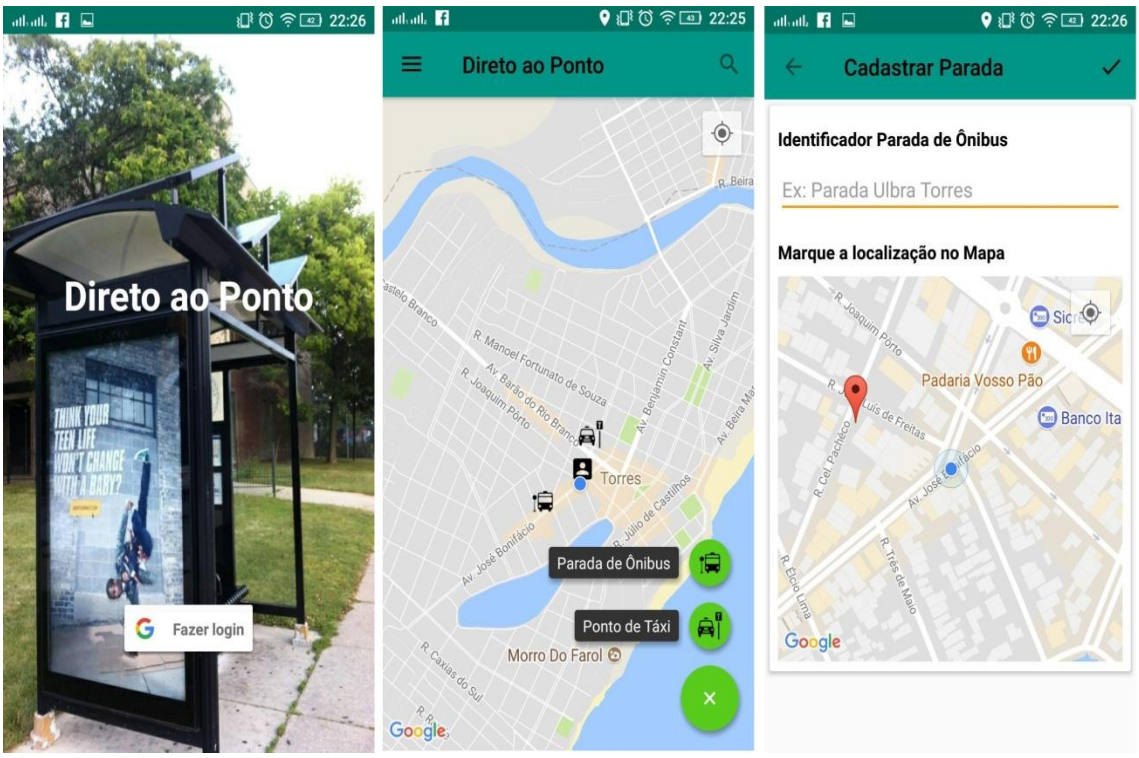


Figura 3. Tela de Login, Tela Inicial e Tela de Cadastro de Parada de Ônibus do app

4.2.2. *Sprint* 2 - Cadastro de Ônibus, Horários, Locais e Taxistas

Na *Sprint* 2, foram realizadas as implementações das funcionalidades descritas no *Sprint Backlog*, Tabela 3, relacionadas à possibilidade de cadastro de ônibus, taxistas, horários e locais.

Tabela 3. *Sprint Backlog* da *Sprint* 2

ITEM	DESCRIÇÃO	PRIORIDADE	ESTIMATIVA
1	Criar tela para informações sobre a parada de ônibus	100	15h
2	Cadastro de ônibus	100	10h
3	Cadastro de horários	100	10h
4	Adição/remoção de locais	90	15h
5	Criar tela para informações sobre a ponto de táxi	90	4h
6	Cadastro de taxistas	90	5h

7	Criar função para saber o próximo horário em que o ônibus irá passar na parada	**85**	10h
8	Validações de horários e locais duplicados	80	10h

Ao concluir as tarefas desta *Sprint*, os usuários poderão adicionar um ônibus a uma parada de ônibus, informando o nome da linha, um horário e os locais do trajeto, adicionar mais horários ao ônibus informando o dia da semana e horário e adicionar ou remover locais do roteiro do ônibus. Também será possível adicionar um taxista a um ponto de táxi, informando um nome e telefone.

Aqui está exemplificado como realizar um cadastro na árvore de horários no *Firebase* por meio desse código:

```
// gera um uuid para o horário que será cadastrado
String uuidHorario = validacao.geraUUID();
//seta o horário informado pelo usuário no nó horário na árvore do horário
firebaseDB.getRef().child("horarios").child(uuidHorario).child("horario").setValue(
editTextHorario.getText().toString());
//seta o dia da semana informado pelo usuário no nó dia na árvore do horário
firebaseDB.getRef().child("horarios").child(uuidHorario).child("dia").setValue(list
DiasDaSemana.get(posicaoSelecionada));
//seta o uuid do ônibus a qual pertence o horário no nó onibus na árvore do horário
firebaseDB.getRef().child("horarios").child(uuidHorario).child("onibus").setValue(u
uidOnibus);
//seta a qtd de likes como zero no nó likes na árvore do horário
firebaseDB.getRef().child("horarios").child(uuidHorario).child("likes").setValue("0
");
//seta a qtd de erros como zero no nó qtd_erros_reportados na árvore do horário
firebaseDB.getRef().child("horarios").child(uuidHorario).child("qtd_erros_reportado
s").setValue("0");
```

A cada linha é inserido um registro na subárvore do novo horário, gerada a partir do *uuid*. Essa subárvore foi cadastrada na árvore 'horários'. A figura 4 mostra como é estruturada a árvore "horários" onde foi feita a inserção do horário:

Figura 4. Árvore que representa os horários no banco de dados do Firebase

No cadastro de horários foi realizada a validação para que a hora não ultrapasse o limite de 23, e os minutos, o limite de 59. Já para a adição ou remoção de locais, foi

utilizada uma biblioteca chamada *TagEditView*[6] disponível no *GitHub* para o componente *EditText*, onde os locais se tornam *tags*. Desta forma, o usuário poderá adicioná-los ou removê-los facilmente. O *Mockup* e o resultado final podem ser vistos no Anexo 3.

4.2.3. Sprint 3 - Reportar Erros, Feedback com Likes, Pesquisa de Ônibus por Local, Caminho a percorrer no Mapa, Refinamentos UI e Validações.

Na *Sprint* 3, foram realizadas as implementações das funcionalidades descritas no *Sprint Backlog*, Tabela 4, relacionadas a reportar erro nos dados cadastrados, o *feedback* com *like* nos horários, pesquisar ônibus informando um destino, caminho a percorrer até a parada de ônibus e os refinamentos da *UI* com *Material Design* e validações.

Tabela 4. *Sprint Backlog* da *Sprint* 3

ITEM	DESCRIÇÃO	PRIORIDADE	ESTIMATIVA
1	Função para reportar erro	100	10h
2	Função para dar e remover like nos horários	100	15h
3	Criar Searchview com locais de sugestão	100	30h
4	Criar tela para mostrar resultados da pesquisa	90	20h
5	Criar função mostrar no mapa o caminho a ser percorrido para chegar a uma parada de ônibus ou ponto de táxi	85	15h
6	Aplicar conceitos do Material Design	80	30h
7	Validações de localização ativa, Intenet, Loading para carregar dados	75	15h

Ao concluir as tarefas desta *Sprint*, os usuários poderão reportar erro nas paradas de ônibus, pontos de táxi, ônibus, taxistas e horários, dar *like* e remover *like* nos horários, identificar qual ônibus pegar informando um destino, descobrir como chegar a uma parada de ônibus ou ponto de táxi, e aproveitar toda a usabilidade proporcionada pelo *Material Design.*

Na funcionalidade de reportar erro é mostrada uma mensagem de confirmação, explicando que, se muitos usuários reportem erro, o dado em si é removido do sistema. Cada usuário pode reportar erro apenas uma vez. A função de *like* nos horários serve para que ele seja validado pela comunidade, desta forma o usuário saberá se o horário cadastrado está de acordo com a realidade.

[6] Disponível em: https://github.com/mabbas007/TagsEditText

O grande trunfo da aplicação é a funcionalidade de pesquisa. O usuário, ao utilizar a barra de pesquisa localizada no topo da tela inicial, informa o seu destino. Então, ao iniciar a digitação, é mostrada, como sugestão, uma lista com locais cadastrados. Ao confirmar a pesquisa, será carregada uma lista contendo os ônibus cujo itinerário contempla o lugar solicitado, mostrando a parada onde esse coletivo passa e o horário mais aproximado ao momento da consulta. Além disso, essa lista é ordenada de acordo com a menor distância entre a localização do usuário e a referida parada de ônibus.

Nessa *Sprint* também foi desenvolvida uma função que permita ao usuário verificar no mapa o caminho que deverá percorrer até chegar a uma parada de ônibus ou ponto de táxi. Por fim, foram realizados os refinamentos na *UI* seguindo os conceitos do *Material Design*, uma metodologia de *design* desenvolvida pela *Google,* com o objetivo de tornar a aplicação fluida, natural, intuitiva e de simples compreensão. Nessa *sprint* os *Mockups* foram realizados conforme a figura 5 e figura 6, enquanto as telas da aplicação podem ser vistas nas figuras 7 e 8.

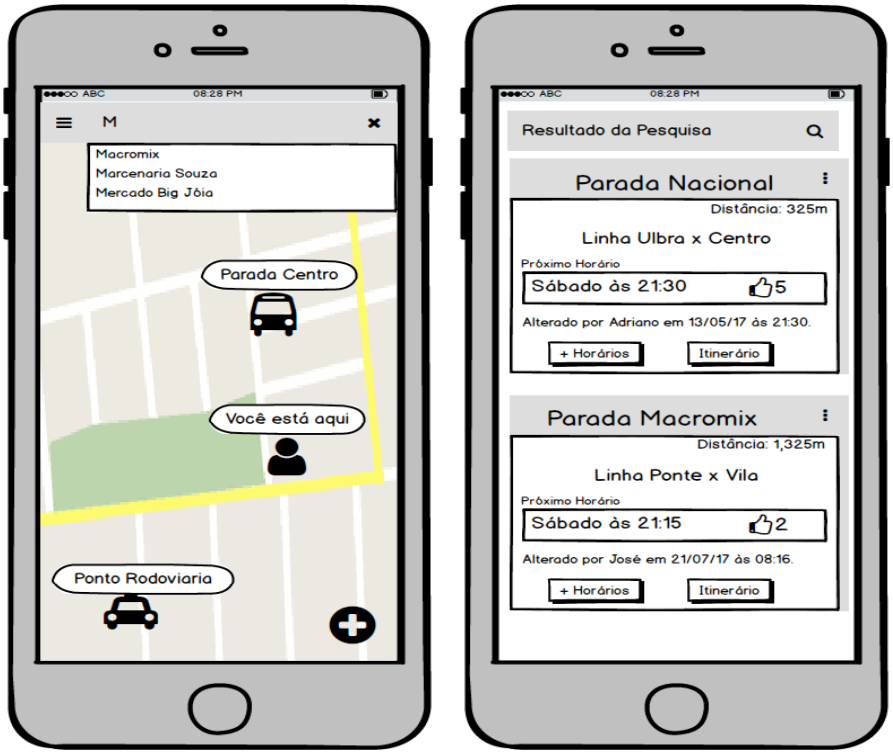


Figura 5. Mockups Tela Inicial com sugestões da barra de pesquisa e Tela de resultado da pesquisa.

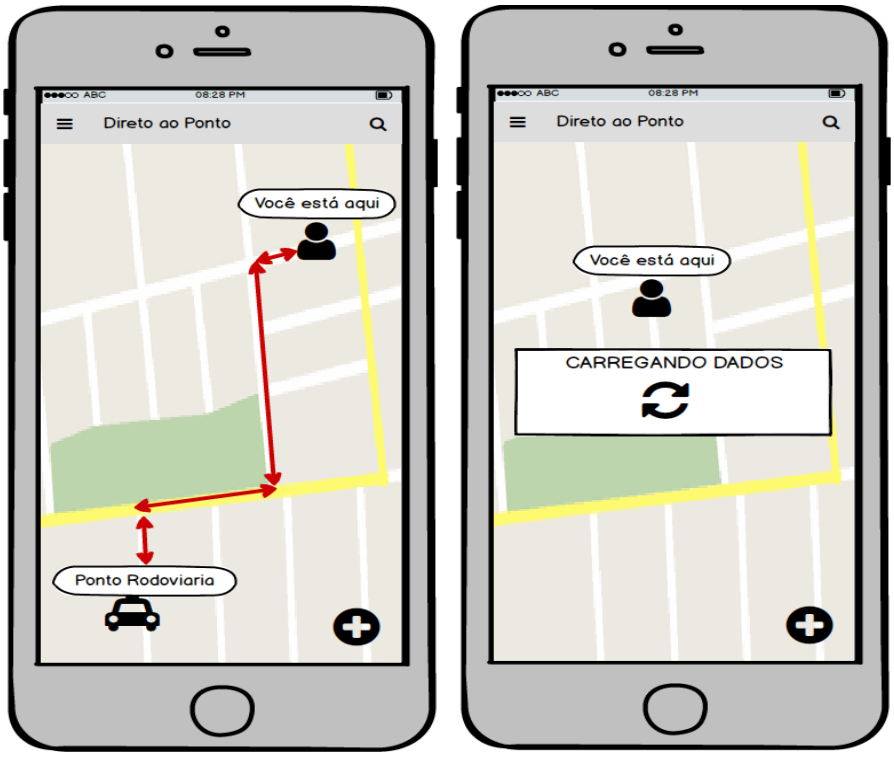


Figura 6. Mockups Tela Inicial com mostrando caminho a ser percorrido até o destino e a tela de loading.

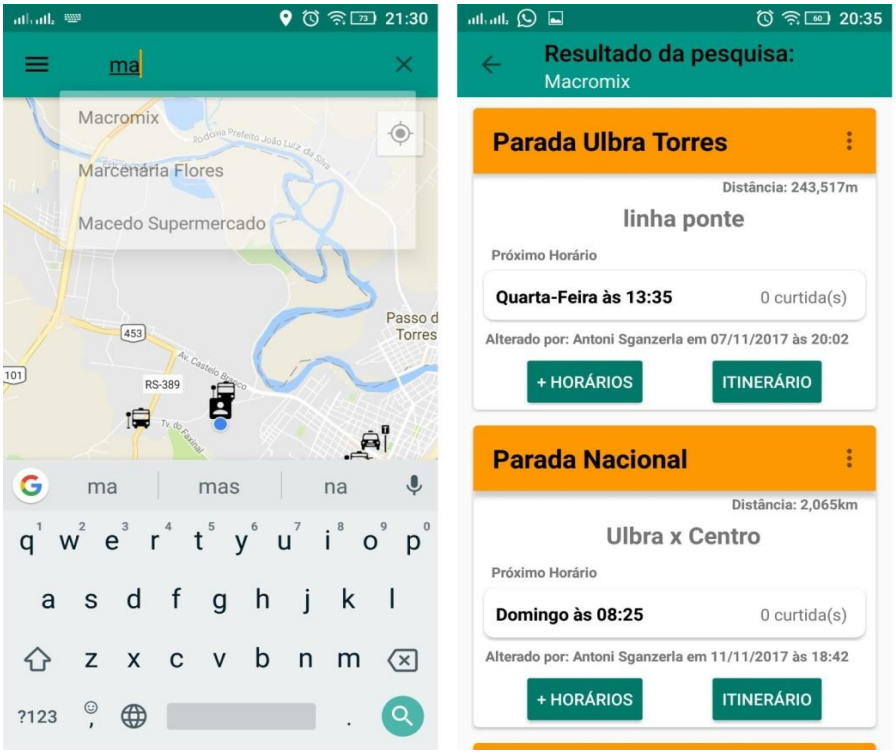


Figura 7. Tela Inicial com sugestões da barra de pesquisa e Tela de resultado da pesquisa do *app*.

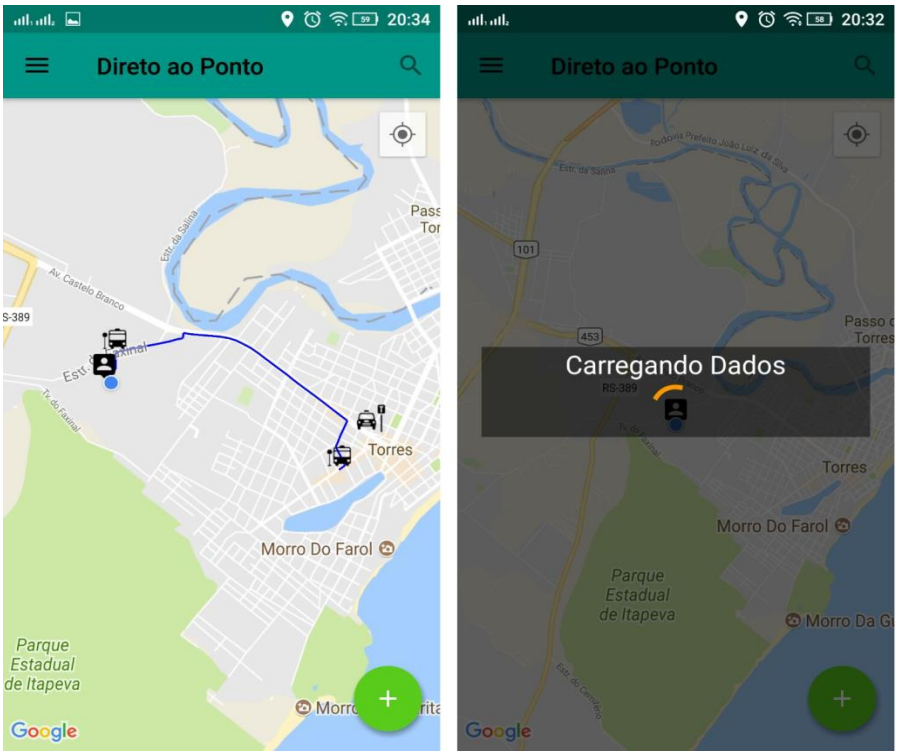


Figura 8. Tela Inicial com mostrando caminho a ser percorrido até o destino e a tela de loading do *app*.

4.3. Pós-planejamento

Nesta etapa, todo desenvolvimento do *software* planejado foi finalizado. Além disso, não só as *Sprints* foram entregues nas datas estipuladas, integrando por completo as funcionalidades, como também foram rodados os testes que validaram as *User Stories* descritas inicialmente.

5. Conclusão

O Direto ao Ponto foi desenvolvido para beneficiar qualquer cidadão que faça uso do transporte público. A aplicação busca fornecer, de forma rápida e prática, as informações sobre ônibus e táxis, para pessoas que possuam um dispositivo com sistema operacional *Android* e acesso à internet. Como o Direto ao Ponto tem a característica de ser colaborativo, ele pode ser utilizado em qualquer lugar do mundo, visto que a própria comunidade abastece o sistema e valida os dados.

Durante o desenvolvimento do sistema, alguns obstáculos foram encontrados, começando pela utilização do serviço *BaaS Firebase*, que ainda não era conhecido ou dominado por completo pelo autor deste trabalho. Ao descobrir as funcionalidades de forma mais detalhada, foi constatado que a recuperação de dados (leitura) é baseada em eventos, desta forma não é possível efetuar consultas com *JOINS* nem determinar exatamente quando os dados terminaram de ser buscados. Também foi constatado que não existe uma função similar ao *LIKE,* como nos bancos de dados relacionais; para possíveis comparações, o termo deve ser exatamente igual. É importante salientar que o banco de dados do *Firebase* é organizado em forma de árvore, o que anula a possibilidade de gerar um diagrama *ER*.

Destaca-se o real desejo em dar continuidade ao projeto, visando à implementação de novas funcionalidades, tais como favoritar um ônibus, definir uma origem diferente da localização atual, implementar o modo *offline* e um filtro de palavras impróprias.

Com a finalização deste projeto, ficou constatado que se tornou real o objetivo inicial do trabalho, permitindo que o Direto ao Ponto funcione como uma solução palpável para facilitar o acesso à informação sobre o transporte coletivo.

Referências Bibliográficas

ANDROID STUDIO. Disponível em: <https://developer.android.com/studio/intro/index.html>. Acesso em: 15 Set. 2017.

BATSCHINSKI, George. Backend as a Service: Prós e Contras. 2016. Disponível em: <https://www.infoq.com/br/news/2016/07/backend-pros-e-contras>. Acesso em: 17 Set. 2017.

BALSAMIQ. Disponível em: <https://balsamiq.com>. Acesso em 30 Set. 2017

BITBUCKET. Disponível em: <https://br.atlassian.com/software/bitbucket>. Acesso em: 17 Set. 2017.

BUILDER, Project. Scrum: o que é sprint e como executá-lo? 2016. Disponível em: <http://www.projectbuilder.com.br/blog-home/entry/conhecimentos/scrum-o-que-e-sprint-e-como-executa-lo>. Acesso em: 15 Set. 2017.

CITTAMOBI. Disponível em: <https://www.cittamobi.com.br>. Acesso em: 17 Set. 2017.

CRUZ, Fábio. Scrum e PMBOK unidos no Gerenciamento de Projetos. 1 ed. Rio de Janeiro: Brasport, 2013.

ICETRAN. O que você precisa saber sobre mobilidade urbana. 2017. Disponível em: <https://icetran.org.br/blog/mais-sobre-mobilidade-urbana-no-brasil/>. Acesso em: 16 Set. 2017.

FIREBASE. Disponível em: <https://firebase.google.com/?hl=pt-br>. Acesso em: 17 Set. 2017.

GOOGLE PLAY. Disponível em: <https://play.google.com/store/apps?hl=pt>. Acesso em: 17 Set. 2017.

MARTINS, José Carlos Cordeiro. Gerenciando Projetos de Desenvolvimento de Software com PMI, RUP e UML. 5 ed. São Paulo: Brasport, 2010.

MENDES, Douglas Rocha. Programação Java com Ênfase em Orientação a Objetos. 1 ed. São Paulo: Novatec, 2009.

MOOVIT. Disponível em: <https://www.company.moovitapp.com/pt>. Acesso em: 17 Set. 2017.

MONTEIRO, João Bosco. Google Android: crie aplicações para celulares e tablets. 1 ed. Casa do Código, 2013.

MOREIRA, Marli. Um em cada quatro brasileiros usa o ônibus como principal meio de transporte. 2015. Disponível em: <http://agenciabrasil.ebc.com.br/geral/noticia/2015-10/um-em-cada-quatro-brasileiros-usa-o-onibus-como-principal-meio-de-transporte>. Acesso em: 23 Set. 2017.

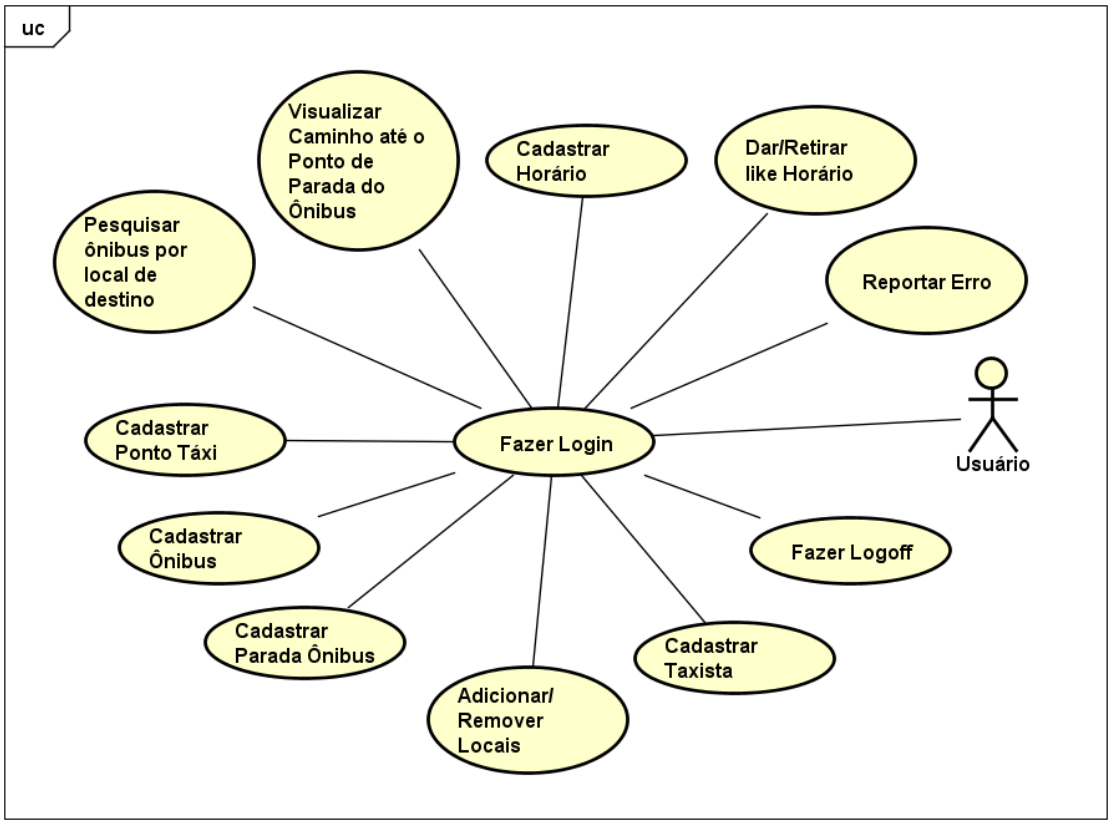
MORIBE, Fernando. Firebase – Vantagens de um BaaS para sua Startup. 2016. Disponível em: <https://medium.com/@fgmoribe/firebase-vantagens-de-um-baas-para-sua-startup-38fd3891329a>. Acesso em: 17 Set. 2017.

RAMOS, Ricardo Argenton. Treinamento Prático em UML. 1 ed. São Paulo: Brasport, 2006.

SALUTES, Bruno. Conheça o Trafi: O aplicativo de transporte público que funciona em modo offline. 2015. Disponível em: <http://www.androidpit.com.br/trafi-aplicativo-transporte-offline>. Acesso em: 17 Set. 2017.

Anexos

Anexo 1: Diagrama de Caso de Uso.



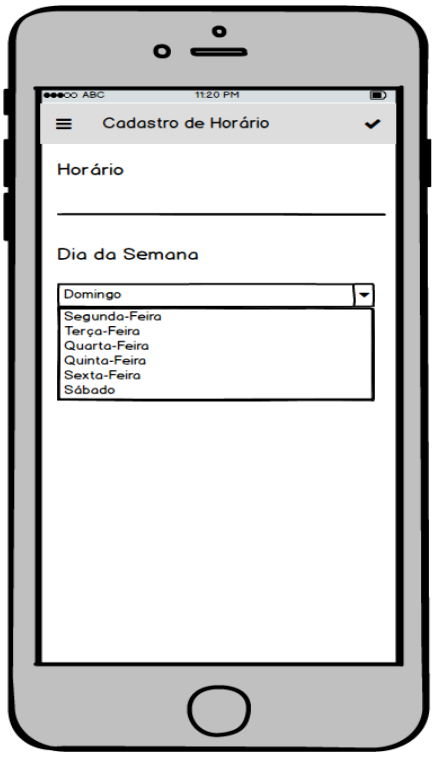
Anexo 2: User Stories.

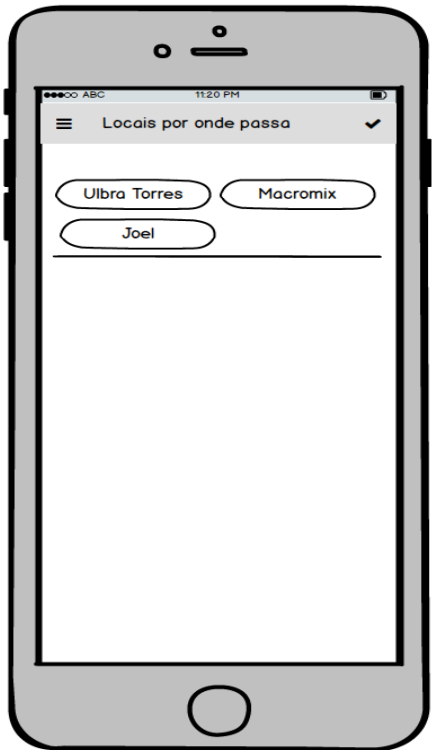
SENDO um usuário sem cadastro
POSSO me cadastrar/fazer login usando a conta do Google Plus
PARA QUE possa ter um perfil e utilizar o aplicativo.

SENDO um usuário do aplicativo **POSSO** cadastrar um ponto de ônibus **PARA QUE** todos os usuários saibam que no local marcado existe um ponto de parada.
SENDO um usuário do aplicativo **POSSO** cadastrar um ônibus num determinado ponto de parada **PARA QUE** todos os usuários saibam que no naquela parada de ônibus passa o ônibus que irei cadastrar.
SENDO um usuário do aplicativo **POSSO** cadastrar um horário na lista de horários de um determinado ônibus **PARA QUE** todos os usuários saibam que esse ônibus possui mais horários.
SENDO um usuário do aplicativo **POSSO** adicionar ou remover locais do itinerário de um determinado ônibus **PARA QUE** todos os usuários saibam que esse ônibus passa nos locais que salvei.
SENDO um usuário do aplicativo **POSSO** cadastrar um ponto de táxi **PARA QUE** todos os usuários saibam que no local marcado existe um ponto de táxi.
SENDO um usuário do aplicativo **POSSO** cadastrar um taxista num determinado ponto de táxi **PARA QUE** todos os usuários saibam que naquele ponto de táxi existe o taxista que irei cadastrar.
SENDO um usuário do aplicativo **POSSO** reportar erro nas paradas de ônibus, pontos de táxi, ônibus, taxistas e horários **PARA QUE** o sistema elimine as inconsistências.

SENDO um usuário do aplicativo **POSSO** consultar os ônibus que vão para um determinado local **PARA QUE** saiba em qual parada de ônibus pegá-lo e em qual horário.
SENDO um usuário do aplicativo **POSSO** dar like em um horário **PARA QUE** todos os usuários tenham o meu feedback sobre o horário.
SENDO um usuário do aplicativo **POSSO** ver o caminho a percorrer no mapa até o ponto de parada ou ponto de táxi **PARA QUE** saiba como chegar até lá.

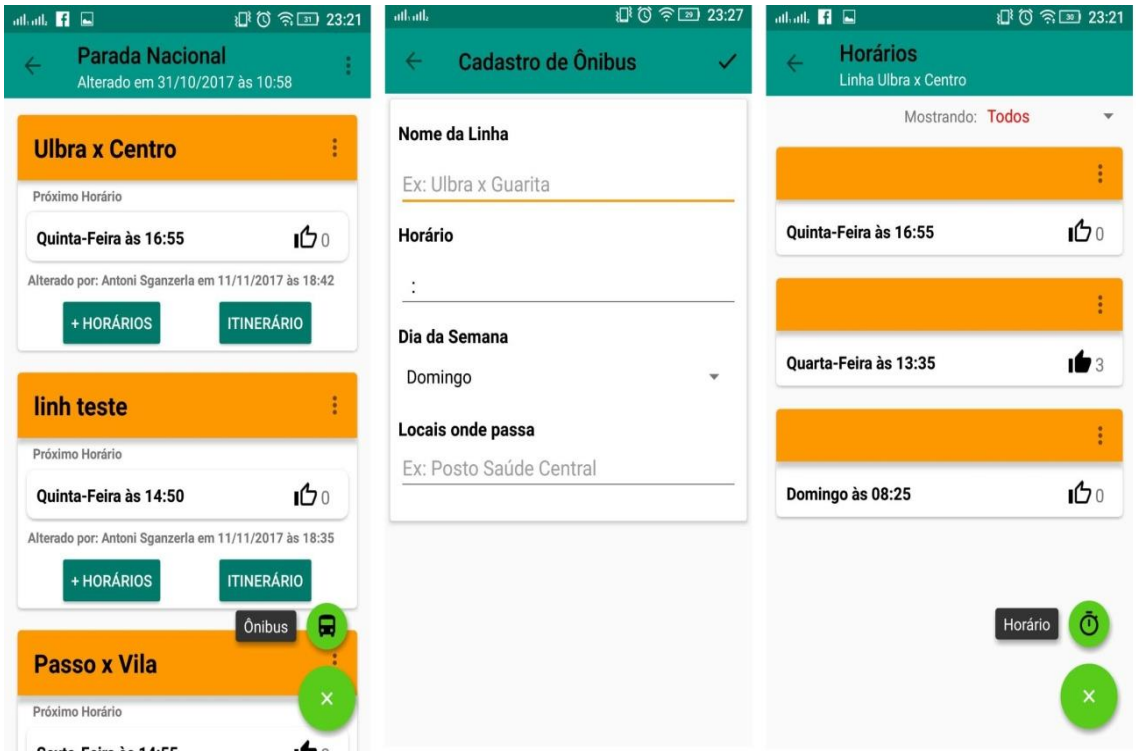
Anexo 3: *Mockups* e Telas do *App Sprint* 2.

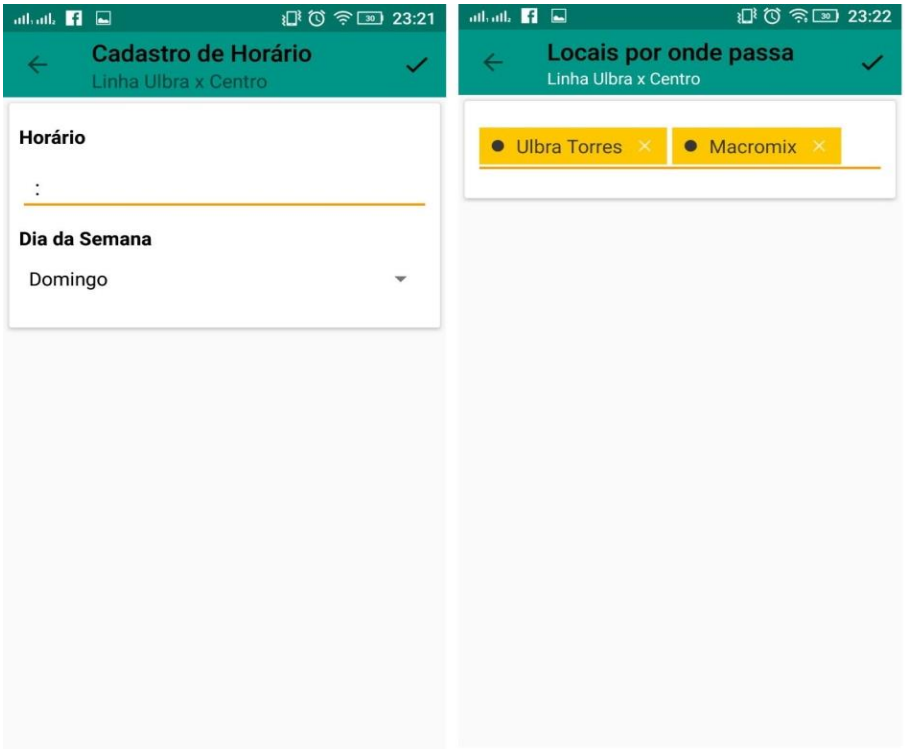

ABC
11:20 PM
Cadastro de Horário
Horário
Dia da Semana
Domingo
Segunda-Feira
Terça-Feira
Quarta-Feira
Quinta-Feira
Sexta-Feira
Sábado

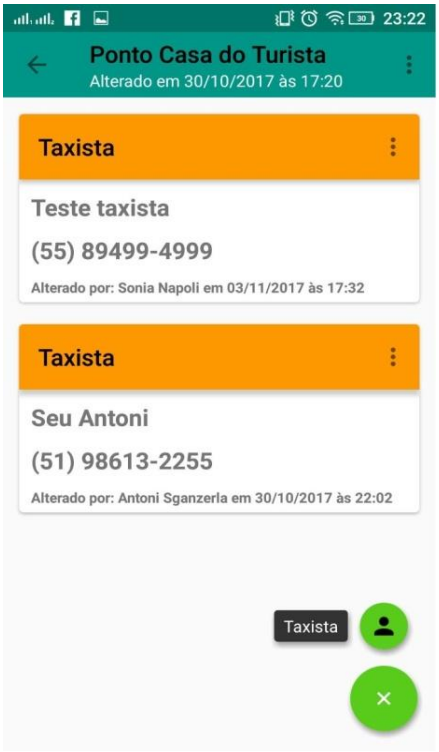

ABC
11:20 PM
Locais por onde passa
Ulbra Torres
Macromix
Joel


ABC
11:20 PM
Cadastro de Taxista
Nome Taxista
Telefone


ABC
11:20 PM
Ponto Casa do Turista
Taxista
João do Táxi
(55) 98033-3929
Alterado por Adriano em 13/05/17 às 21:30.
Taxista
Josefino de
(55) 91339-3299
Alterado por Silveira em 10/09/17 às 08:11.


23:21
Parada Nacional
Alterado em 31/10/2017 às 10:58
Ulbra x Centro
Próximo Horário
Quinta-Feira às 16:55
0
Alterado por: Antoni Sganzerla em 11/11/2017 às 18:42
+ HORÁRIOS
ITINERÁRIO
linh teste
Próximo Horário
Quinta-Feira às 14:50
0
Alterado por: Antoni Sganzerla em 11/11/2017 às 18:35
+ HORÁRIOS
ITINERÁRIO
Ônibus
Passo x Vila
Próximo Horário
23:27
Cadastro de Ônibus
Nome da Linha
Ex: Ulbra x Guarita
Horário
:
Dia da Semana
Domingo
Locais onde passa
Ex: Posto Saúde Central
23:21
Horários
Linha Ulbra x Centro
Mostrando: Todos
Quinta-Feira às 16:55
0
Quarta-Feira às 13:35
3
Domingo às 08:25
0
Horário


23:21
Cadastro de Horário
Linha Ulbra x Centro
Horário
:
Dia da Semana
Domingo
23:22
Locais por onde passa
Linha Ulbra x Centro
Ulbra Torres
Macromix


23:22
Ponto Casa do Turista
Alterado em 30/10/2017 às 17:20
Taxista
Teste taxista
(55) 89499-4999
Alterado por: Sonia Napoli em 03/11/2017 às 17:32
Taxista
Seu Antoni
(51) 98613-2255
Alterado por: Antoni Sganzerla em 30/10/2017 às 22:02
Taxista


23:22
Cadastro de Taxista
Nome Taxista
Ex: João do Táxi
Telefone
( ) -