UNIVERSIDADE FEDERAL DO PAMPA - UNIPAMPA

Curso Bacharelado em Engenharia de Computação

Disciplina de Engenharia de Software

Professor: Carlos Michel Betemps

Documento de Requisitos e Relatório de Implementação

Aplicativo Bus Tracker Bagé

Jiovana Sousa Gomes

Marina Silva da Silva

Matheus Collares Machado

Versão 1.0 - julho de 2019

1. Introdução

Este documento especifica o sistema Bus Tracker Bagé fornecendo aos desenvolvedores as informações necessárias para o projeto e implementação, assim como para a realização dos testes e homologação do sistema.

1.1. Identificação dos requisitos: os requisitos seguem um padrão de identificação composto por uma sigla seguido por um número: RF000 (requisitos funcionais) e NF000 (requisitos não funcionais).

1.2. Prioridades dos requisitos: para se estabelecer a prioridade dos requisitos foram adotadas as denominações “essencial” (sem esses requisitos o sistema não entra em funcionamento), “importante” (sem eles o sistema funciona, porém não de forma satisfatória) e “desejável” (requisito que não compromete as funcionalidades básicas do sistema).

2. Descrição geral do sistema

O sistema é composto por um aplicativo para dispositivos Android que irá mostrar um mapa com a localização em tempo real dos ônibus que circulam pela cidade de Bagé, bem como calcular suas horas de chegada em paradas ou outros pontos definidos pelo usuário.

O sistema fará uso das APIs do Google, como o Maps, Routes e Transport, para obter os dados dos mapas e rotas dos ônibus na cidade.

Funcionalidades principais:

- Permitir ao usuário a seleção de uma parada de ônibus no mapa, a escolha de uma das linhas que passam por ela e apresentar no mapa a rota e o ônibus se deslocando nela, fornecendo dados como distância e estimativa do horário de chegada na parada selecionada.
- A partir da seleção de um local no mapa, apresentar quais linhas de ônibus o usuário pode pegar para chegar ao local escolhido.
- Após inicialização do aplicativo, o sistema mostrará quais paradas estão próximas da posição do usuário.

A proposta inicial do projeto é fazer uso de dois smartphones contendo o aplicativo, sendo um deles representante de um usuário passageiro e o outro estaria simulando o ônibus. Essa versão só é adequada para testes iniciais das APIs. Uma versão mais adequada ao problema seria o uso de microcontroladores, instalados em cada ônibus com a função específica de envio da localização do veículo, dentre outros sinais que forem importantes. Esse aparelho se comunicaria com um servidor, que enviaria as informações aos aplicativos de passageiros.

Tal sistema seria importante para os usuários do sistema de transporte público, reduzindo o tempo de espera nas paradas, bem como para as próprias

empresas de transporte monitorarem os veículos, se estes estão cumprindo os horários, a velocidade deles, uso de combustível, etc. Há estudos mostrando que quando um usuário tem acesso à dados em tempo real do transporte, isso gera um nível de satisfação maior dos passageiros em relação ao serviço, o que tem benefícios para as empresas.

3. Requisitos funcionais

Requisitos funcionais descrevem a funcionalidade ou serviços do sistema, depende do tipo de software, dos usuários e do tipo de sistema onde o software será utilizado. Podem ser declarações de alto nível sobre o que o sistema deve fazer, descrevendo detalhadamente os serviços do sistema.

Lista de requisitos funcionais:

RF00: Mostrar um mapa da cidade de Bagé

Descrição: Após inicialização do aplicativo, apresentar na tela um mapa da cidade de Bagé centralizado na posição atual do usuário, ou centralizado em alguma parada selecionada previamente como ‘favorita’.

Prioridade: Essencial

RF01: Realizar a seleção da parada

Descrição : Permitir ao usuário a seleção da parada desejada no mapa e mostrar todas as linhas de ônibus que passam nesta linha, por meio de uma lista.

Prioridade: Essencial

RF02: Realizar seleção da linha

Descrição: Selecionar uma das linhas de ônibus da lista, para permitir visualização do ônibus dessa linha que está mais próximo da parada.

Prioridade: Essencial

RF03: Mostrar o ônibus se movendo dentro da rota

Descrição: Possibilitar a visualização do ônibus mais próximo que está em direção à parada selecionada, mostrando o veículo se movendo dentro de sua rota no mapa, bem como informações sobre sua distância e tempo em min para chegada na parada.

Prioridade: Essencial

RF04: Mostrar quanto tempo/ a que distância o ônibus se encontra da parada

Descrição: Após seleção da linha, permitir escolha entre a visualização do ônibus no mapa ou apenas os dados referentes ao tempo e distância do veículo, da parada desejada. Tais dados de tempo estarão em minutos e segundos, e a distância estará em metros ou quilômetros.

Prioridade: Essencial

RF05: Sistema de login de usuário

Descrição: Permitir cadastro e login de usuários no sistema, além de tipos diferentes de usuários: ônibus, passageiro, motorista, cobrador e administrador.

Prioridade: Importante

RF06: Seleção de parada favorita

Descrição: O usuário pode salvar uma parada e linha de ônibus como favorita, que, ao abrir o aplicativo, será a centralizada no mapa e já mostrará as informações referentes à ela.

Prioridade: Desejável

RF07: Uso de cores diferentes para mostrar estado das linhas de ônibus

Descrição: Na lista de linhas, fazer uso de cores diferentes de acordo com o estado das linhas, para evidenciar problemas como atrasos e outros problemas de trânsito. Tais estados podem ser verificados com análise dos dados sobre os veículos, se estes estão apresentando horários de chegada em paradas diferentes dos normais, ou se sua velocidade está abaixo do normal para aquele momento.

Prioridade: Desejável

4. Requisitos não funcionais

Os requisitos não-funcionais definem as propriedades e as restrições do sistema, como confiabilidade, tempo de resposta e ocupação de área. As restrições são capacidade dos dispositivos E/S, representações do sistema, uso de IDE particular, linguagem de programação ou método de desenvolvimento específico. Os requisitos não-funcionais podem ser mais críticos do que os requisitos funcionais, de forma que, se não forem atendidos, o sistema pode ser inútil.

Lista de requisitos não funcionais:

NF00: Restrição no horário de disponibilidade do sistema

Descrição: O aplicativo só poderá apresentar informações em tempo real dos veículos durante o horário de trabalho destes. Fora desse horário, o sistema deve permanecer com essas funções indisponíveis. No lugar, ele pode mostrar o horário previsto para a primeira linha do próximo dia.

Prioridade: Desejável

NF01: Rapidez ao mostrar as informações

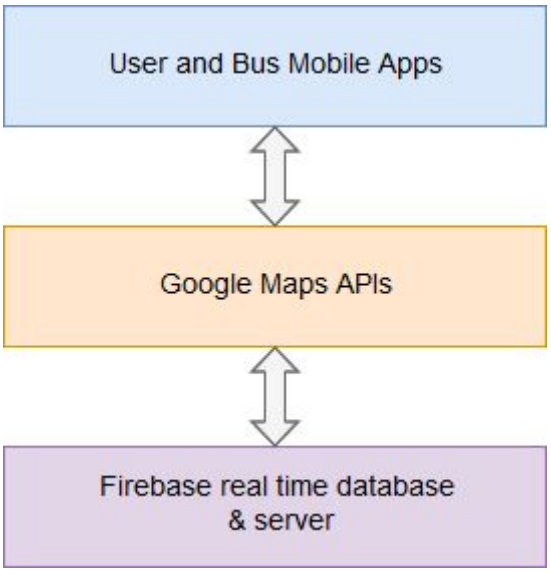
Descrição: O sistema não pode demorar muito tempo para mostrar as informações de distância e tempo dos ônibus. O tempo máximo de processamento não deve exceder 10s. Para deslocamento do veículo no mapa, no máximo 1s, se possível. Se o usuário não tiver uma conexão de internet rápida o suficiente, serão mostrados apenas os dados numéricos, sem mostrar o ônibus se movimentando.

Prioridade: Importante

5. Arquitetura e modelos do sistema

5.1 Arquitetura

Figura 1: Arquitetura do Sistema



Fonte: Autores (2019)

A arquitetura do sistema foi escolhida em camadas, pois foi a arquitetura que mais pareceu adequar-se ao projeto do sistema. Como pode ser visto na figura 1, a arquitetura apresenta uma camada composta pelas aplicações mobile, que possui 2 modos de usuário: aqueles que recebem a localização (os ‘users’) e aquele que envia a localização (o ‘bus’), enviando esses dados para as APIs e para o banco de dados. A camada de APIs faz a implementação das APIs do google, a fim de mostrar as informações necessárias no aplicativo. A camada do banco de dados armazena e gerencia as informações, recebendo dados do aplicativo ‘bus’, processando e enviando estes para o aplicativo ‘user’.

5.2 Diagrama de casos de uso

A versão inicial do sistema foi projetada apresentando 8 casos de uso, com base nos requisitos apresentados previamente. São eles:

Caso 01 - Selecionar parada: Realiza seleção de uma parada qualquer no mapa da cidade.

Caso 02 - Selecionar linha: Seleciona uma linha de ônibus dentre aquelas que passam pela parada selecionada no caso 01.

Caso 03 - Ver rota e ônibus: Apresenta no mapa a rota e o ônibus que está mais próximo da parada, em movimento dentro da rota.

Caso 04 - Ver paradas próximas: Mostra quais as paradas mais próximas da posição atual do usuário.

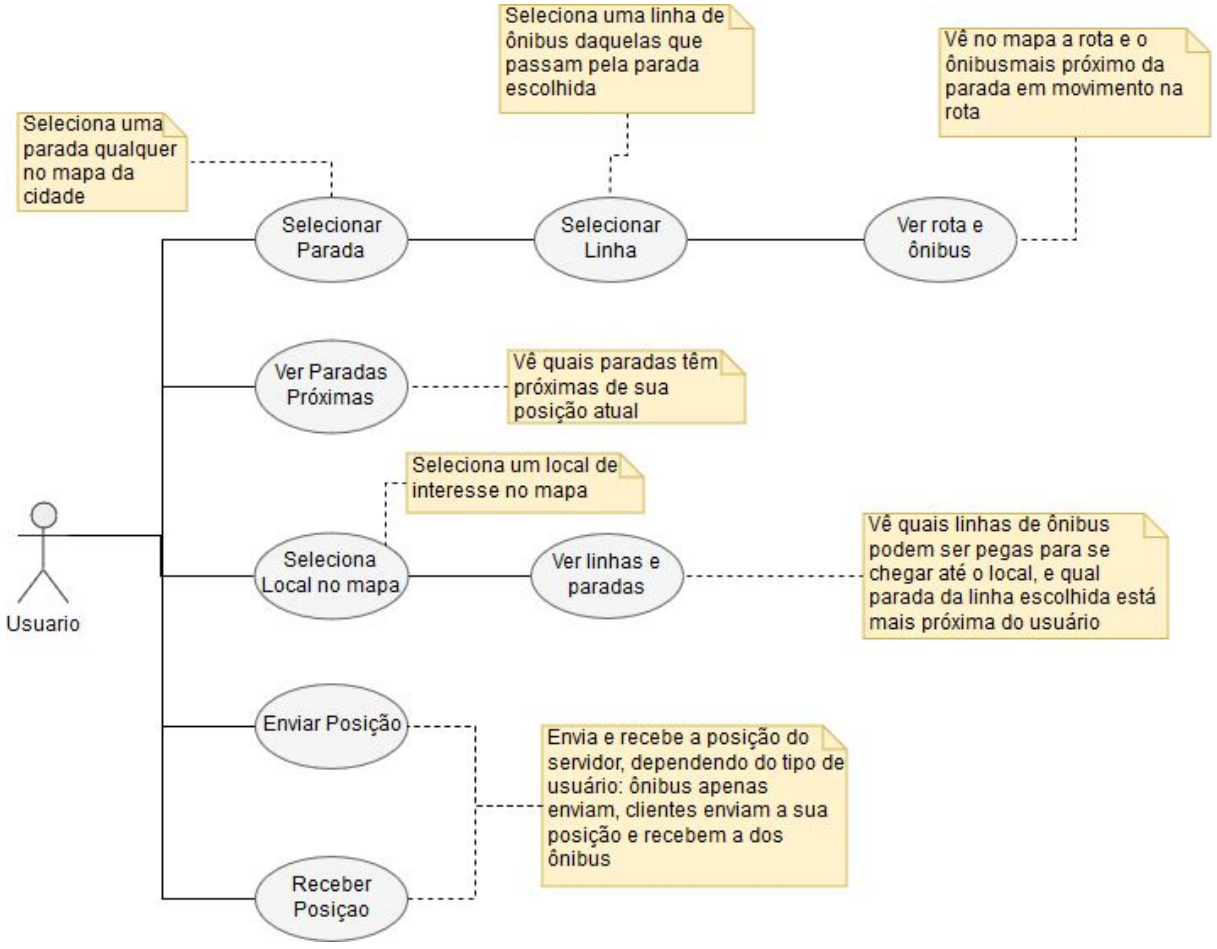
Caso 05 - Seleciona local no mapa: Seleciona um local de interesse, como uma escola ou hospital, no mapa.

Caso 06 - Ver linhas e paradas: O sistema apresenta para o usuário quais linhas de ônibus podem ser usadas para se chegar ao local selecionado, e qual parada onde se pode embarcar está mais próxima do usuário.

Caso 07 - Envia posição: Usuário do tipo ‘ônibus’ enviam constantemente sua posição para o servidor/banco de dados.

Caso 08 - Receber posição: Usuários ‘passageiros’ recebem as posições do ônibus para serem apresentadas no mapa do aplicativo.

Figura 2 - Casos de uso do sistema

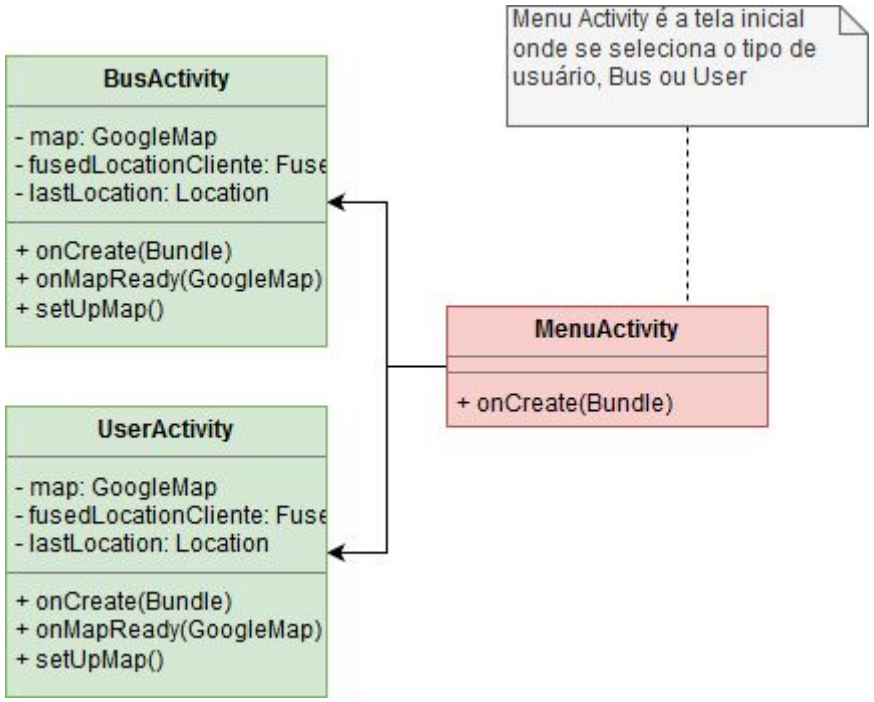


Fonte: Autores (2019)

5.2 Diagrama de classe

O diagrama de classes presente na figura 3 foi o modelado para a versão inicial do aplicativo, possuindo três classes: MenuActivity, BusActivty e UserActivity. A classe Menu é a responsável pela seleção do tipo de usuário, se o aplicativo irá funcionar como ‘user’ ou como ‘bus’, ou seja, se estarão disponíveis as opções para usuários ou para o ônibus. As classes Bus e User são responsáveis pelo envio e recebimento de posições, sendo que a classe Bus apenas envia sua posição, enquanto a classe User recebe posições do tipo Bus e envia a sua.

Figura 3 - Diagrama de classes

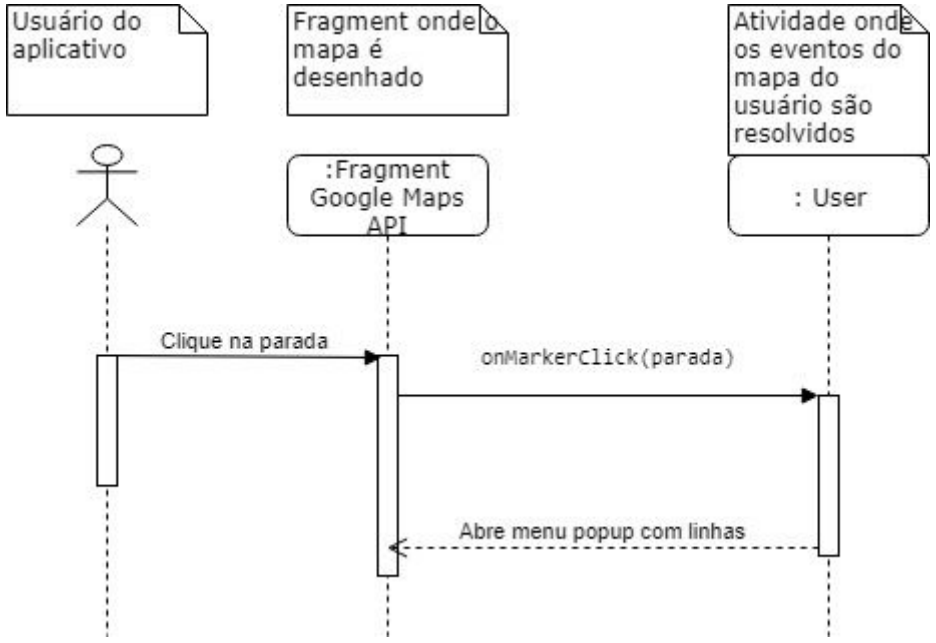


Fonte: Autores (2019)

5.3 Diagramas de sequência

A seguir estão os diagramas de sequência dos casos de uso planejados para implementação no protótipo:

Figura 4 - Diagrama de sequência “Selecionar Parada”

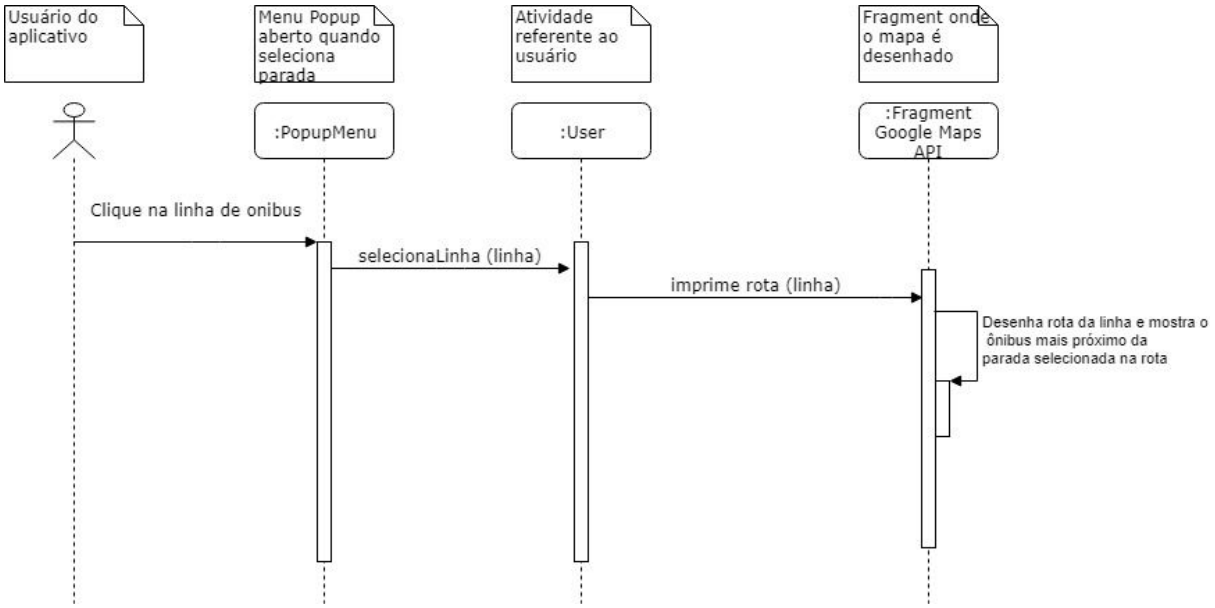


Fonte: Autores (2019)

Este diagrama descreve o clique do usuário na parada de ônibus desejada, esta ação desencadeia um evento que é tratado na classe “User” pelo método “onMarkerClick()” que leva como parâmetro o marcador clicado. O tratamento feito

neste método deve criar um menu popup para seleção das linhas e mostrar para o usuário.

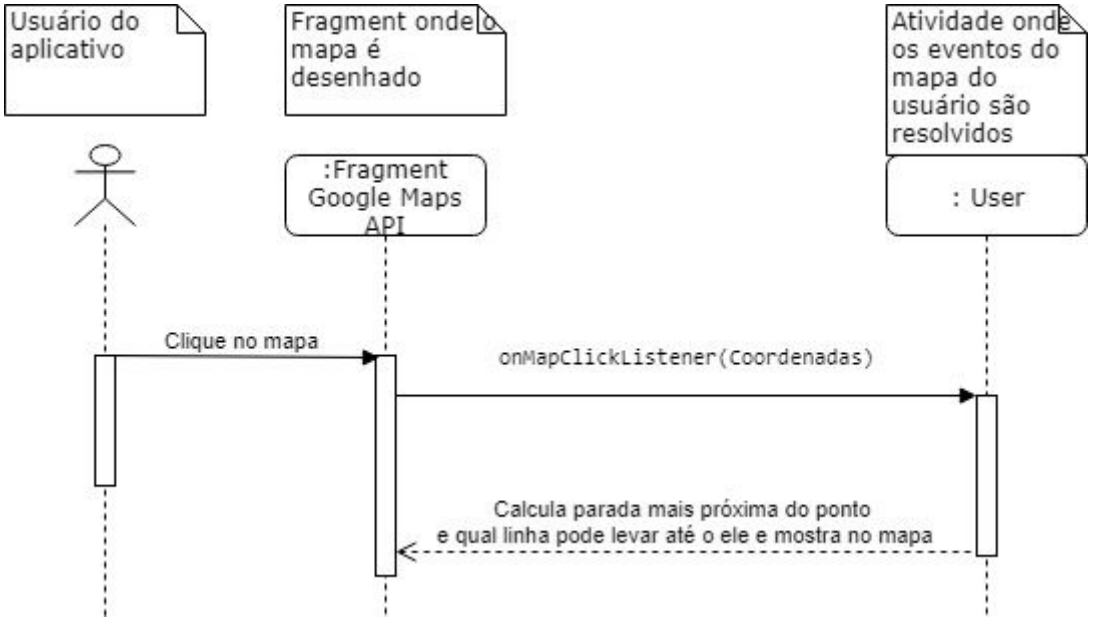
Figura 5 - Diagrama de sequência “Selecionar Linha”



Fonte: Autores (2019)

Após o clique na parada o popup menu estará visível ao usuário, o diagrama acima mostra o comportamento do programa após a seleção da linha. Esta seleção é enviada a classe user pela chamada do método “selecionaLinha()” com o parâmetro “linha” e então enviado para o fragment para sua impressão na tela.

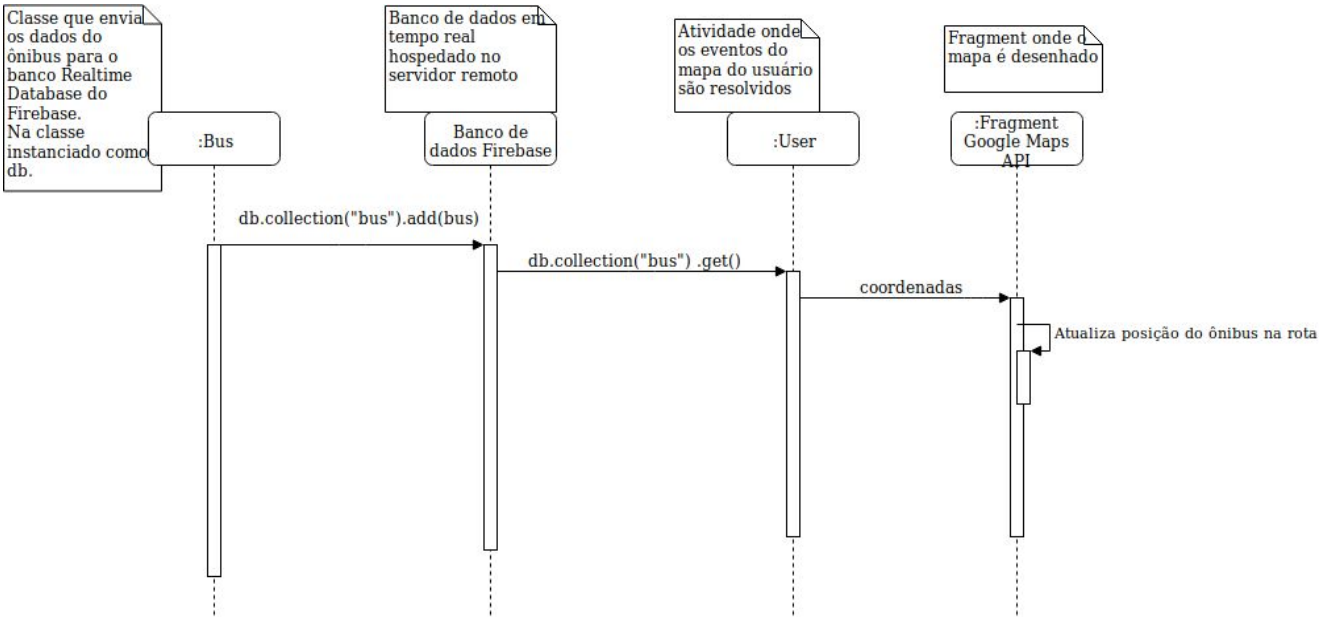
Figura 6 - Diagrama de sequência “Selecionar lugar no mapa e ver linhas e paradas”



Fonte: Autores (2019)

É possível observar no diagrama acima outra funcionalidade proposta para o aplicativo, que seria a funcionalidade de selecionar um ponto no mapa onde você deseja chegar e calcular os ônibus e paradas que você precisaria acessar para chegar lá. O clique no mapa dado pelo usuário deve passar as coordenadas para a classe “User” que fará o cálculo do trajeto necessário para o usuário chegar na parada mais próxima e qual ônibus ele deve usar para chegar na localização.

Figura 7 : Diagrama de sequência “Envia e recebe posição”



Fonte: Autores (2019)

Nesse diagrama está ilustrado como as classes acessam as localizações necessárias. A classe Bus detecta a localização atual do ônibus e envia para o banco de dados como um objeto bus através do método add(), salvando na coleção bus. Essa coleção é a mesma utilizada pela classe User para buscar a localização do ônibus desejado, através do método get(). Por fim, quando essas coordenadas são recuperadas o Fragment com o mapa da Google Maps API marca a posição do ônibus.

6. Ambiente de Desenvolvimento e Linguagens Utilizadas

O projeto foi desenvolvido com o uso da IDE Android Studio, utilizando a linguagem Kotlin. Foi planejado o uso da ferramenta Trello para gerência do projeto. O controle de versão ocorreu através do Github. O processo de software escolhido foi o desenvolvimento incremental e orientado à reúso, pois as APIs são componentes facilmente reutilizáveis, e o projeto foi pensado para ser executado em incrementos, com acréscimo de funcionalidades.

7. Imagens do protótipo

Figura 8: Tela inicial do protótipo

Fonte: Autores (2019)

A figura 8 mostra a tela inicial do protótipo, apresentando os botões de seleção do tipo de usuário. Ao clicar no botão usuário, a tela é alterada para um mapa que apresenta a posição do usuário e a do ônibus. O botão do ônibus altera a posição do veículo no servidor, e mostra a posição atual dele no mapa.