UNIVERSIDADE FEDERAL DO CEARÁ

CAMPUS QUIXADÁ

CURSO DE GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

CARLOS EDUARDO DA SILVA FERREIRA

BANCO DE TRÁS: UM APLICATIVO MOBILE PARA CARONAS

QUIXADÁ

2022

CARLOS EDUARDO DA SILVA FERREIRA

BANCO DE TRÁS: UM APLICATIVO MOBILE PARA CARONAS

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

Orientador: Prof. Dr. Marcio Espíndola Freire Maia

QUIXADÁ

2022

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

F44b Ferreira, Carlos Eduardo da Silva.
Banco de Trás: Um Aplicativo Mobile Para Caronas / Carlos Eduardo da Silva Ferreira. – 2022.
85 f.

Trabalho de Conclusão de Curso (graduação) – Universidade Federal do Ceará, Campus de Quixadá, Curso de Ciência da Computação, Quixadá, 2022.
Orientação: Prof. Dr. Marcio Espíndola Freire Maia.

1. Android (Programa de computador). 2. Firebase. 3. Google Maps. I. Título.

CDD 004

CARLOS EDUARDO DA SILVA FERREIRA

BANCO DE TRÁS: UM APLICATIVO MOBILE PARA CARONAS

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

Aprovada em: __/__/_____

BANCA EXAMINADORA

Prof. Dr. Marcio Espíndola Freire Maia (Orientador)
Universidade Federal do Ceará (UFC)

Prof. Dr. Michel Sales Bonfim
Universidade Federal do Ceará - UFC

Prof. Me. Camilo Almendra
Universidade Federal do Ceará - UFC

À minha família que sempre esteve ao meu lado em todos os momentos, e a todos os amigos que encontrei no caminho.

AGRADECIMENTOS

Agradeço à minha família, pelo apoio e amor que sempre recebi. Agradeço também aos colegas e amigos que colhi durante esses anos de luta. Ao Alessandro e ao Claro, que me ajudaram diversas vezes no caminho, ao Diego por ser um amigo incrível, ao Lucas e o Zé Gabriel pelas várias risadas, e ao Alcides, pelos abraços fortes e acolhedores e pelo todo o carinho que compartilhamos. Desejo sucesso e felicidade a todos.

“We are never sad ’cause we are not allowed to be.”

(Breaking Benjamin)

RESUMO

O transporte para as instituições de ensino dos estudantes do ensino superior da cidade de Quixadá se dá, normalmente, por apenas alguns ônibus escolares que realizam o trajeto do Campus para o centro da cidade, além disso, a quantidade de alunos cresce a cada semestre, tornando tal transporte cada vez menos acessível para os estudantes. Como uma forma de amenizar o problema do deslocamento, este trabalho propõe a implementação de um aplicativo que permita a criação e gerenciamento de viagens entre os estudantes e possivelmente professores. Para isso, a aplicação utiliza do ambiente Android, bem como plataformas externas, como os serviços do *Firebase* e da *Google* para implementação das funcionalidades do sistema.

Palavras-chave: Android (Programa de computador). Firebase. Google Maps.

ABSTRACT

The transport to educational institutions for students in the city of Quixadá is usually done by just a few school buses that travel from the university campus to the urban area, furthermore, the number of students grows every semester, making such transport less and less accessible for then. As a way to ease this commuting issue, this work aimed the implementation of an application that allows the creation and management of carpools between students and possibly teachers. For this, the application uses the Android environment, as well as external platforms, such as *Firebase* and *Google* services to implement the system's features.

Keywords: Android (software). Firebase. Google Maps.

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

Um dos maiores desafios da educação brasileira, principalmente na zona rural, é a acessibilidade e a permanência do aluno no ambiente escolar. A dificuldade que os estudantes da zona rural têm para chegar à escola ou faculdade, normalmente é consequência das grandes distâncias dos trajetos e pela carência na disponibilidade dos transportes garantidos pelo estado, este que por sua vez, arca com elevados custos de manutenção dos veículos (NASCIMENTO *et al*., 2016).

G1 (2019) cita que na cidade de Quixadá, o transporte destinado ao ensino técnico e universitário da cidade tem sido precário nos últimos anos, com poucos ônibus realizando as rotas. Ademais, Santos (2007) consta que, segundo o Ministério da Educação (MEC), cerca de 23 mil estudantes no Brasil podem estar fora da sala de aula por falta de transporte.

Nesse contexto, destaca-se a importância da otimização das viagens realizadas pelo sistema de transporte, seja pela revisão da infraestrutura das rotas ou pela criação de sistemas paralelos que auxiliem os estudantes, como aplicativos ou *softwares* diversos.

Uma maneira de amenizar a problemática do transporte universitário e da evasão é através das caronas. Com as caronas, os alunos podem compartilhar seus veículos próprios com outros estudantes da mesma instituição de ensino ou de instituições próximas, facilitando assim, a chegada na faculdade e aliviando de certa forma o transporte público. Além disso, Silva (2014) mostra que, com a crescente evolução da internet e mídias sociais, o uso de dispositivos móveis, *smartphones* e *tablets* tem proporcionado uma revolução de maior impacto nos últimos tempos, onde quase todos possuem algum tipo de dispositivo móvel. Portanto, uma aplicação voltada para a criação de caronas seria acessível para a grande maioria das pessoas.

Diversos impactos sociais, ambientais e comportamentais são atribuídos ao sistema de caronas. Uma crescente massa de evidências indicam que as caronas fornecem benefícios como a redução no consumo de energia e na emissão de gases poluentes, redução do tráfego local e congestionamentos, além da redução na demanda de estacionamentos (SHAHEEN *et al*., 2018).

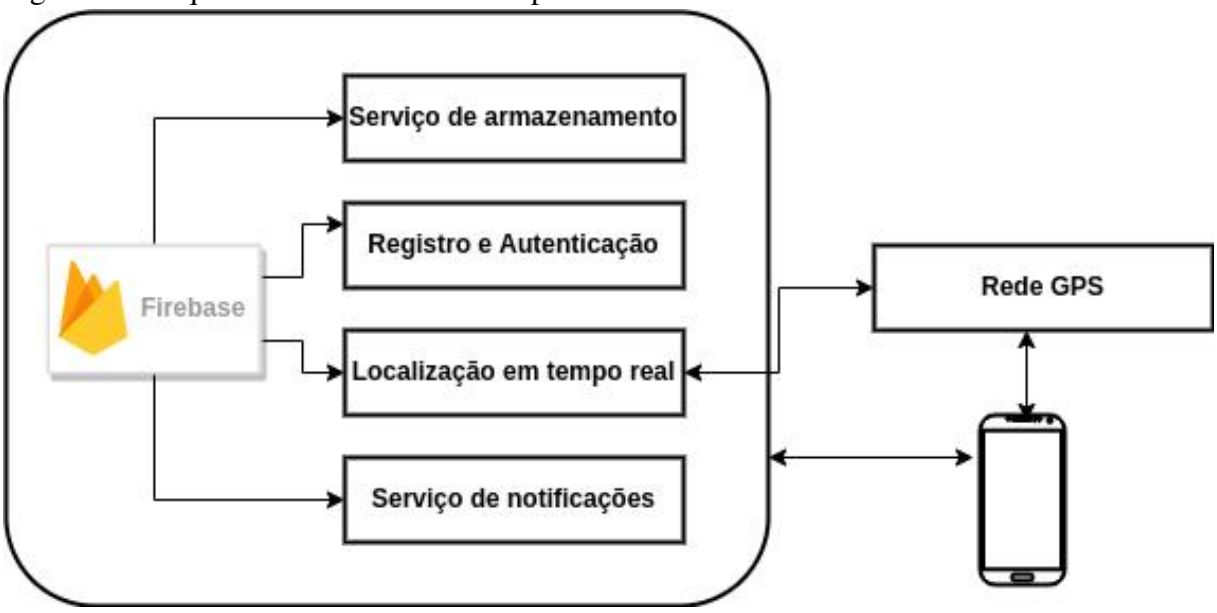
Outras questões como segurança, proteção e taxação são levantadas no contexto de uma carona (LEE *et al*., 2017). Segundo Toluna (2019), empresa especializada em coleta de dados e informações, após uma pesquisa realizada com 760 pessoas que buscava saber os motivos que levavam ou não a utilização de aplicativos de carona, concluiu que 61% das pessoas buscam facilidade em chegar ao seu destino, 56% delas buscam preços mais baixos que aplicativos de

transporte e 45% não gostam de transporte público. Além disso, a pesquisa revela que 65% das pessoas não utilizariam tais sistemas por medo ou falta de segurança e 26% por desconforto em compartilhar veículos com alguém desconhecido.

Além das questões de segurança e proteção, a problemática da saúde se torna um fator fundamental que influencia na decisão das pessoas participarem de uma viagem compartilhada. Desde a tomada de medidas sanitárias para combater a transmissão do Coronavírus (COVID-19), como distanciamento social e uso de máscaras, a população tem evitado ficar em lugares fechados, principalmente com desconhecidos. Segundo Claudia Woods (2020), diretora-geral da Uber no Brasil, as medidas de prevenção recomendadas pela Organização Mundial da Saúde passariam a ser obrigatórias para quem usa a plataforma.

Tais dados mostram que a principal preocupação com esse tipo de aplicação é a saúde e segurança. Assim, o sistema proposto neste trabalho tem como intuito criar um caminho seguro e eficaz entre quem pode fornecer uma carona, com os alunos do ensino superior de Quixadá, visando a realidade dos estudantes. A Figura 1 a seguir mostra simplificadamente a estrutura da aplicação proposta, no qual as funcionalidades como a localização em tempo real, autenticação e armazenamento são mostradas, e como tais funções do sistemas se relacionam diretamente com a plataforma *Firebase* e com o usuário.

Figura 1 – Arquitetura em alto nível do aplicativo



Fonte: Elaborado pelo autor.

1.1 Objetivos

1.1.1 Objetivo Geral

O objetivo geral deste trabalho é o desenvolvimento de um aplicativo de caronas para a plataforma Android para os alunos do ensino superior de Quixadá, visando amenizar o problema dos transportes e da grande lotação dos ônibus.

1.1.2 Objetivos Específicos

- Definições de regras de negócio e histórias de usuário, bem como a arquitetura e padrões de projeto;
- Integrações com sistemas externos e implementações das funcionalidades;
- Testes e validações

2 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo serão apresentados e analisados os principais conceitos que serão utilizados neste trabalho. Os itens que serão apresentados refletem diretamente na construção da aplicação proposta.

2.1 Carona

Muitas pessoas precisam de um meio de transporte para a vida cotidiana, mas têm crescentes preocupações relacionadas ao transporte, como segurança, poluição, tráfego e estacionamento (LEE *et al*., 2017).

A carona é um meio de compartilhamento de veículos pelo qual os motoristas compartilham seus carros com um ou mais passageiros cujos itinerários de viagem são semelhantes aos seus. Como tal, as caronas podem ser uma maneira eficaz de aliviar o congestionamento do tráfego, diminuir a quantidade de poluentes no ar e ainda aproximar pessoas com interesses em comum (HUANG *et al*., 2015).

As caronas, como um modo de viagem, ainda não receberam atenção suficiente, apesar de seus méritos de conveniência, custo-benefício e sustentabilidade. Os estudos sobre caronas, especialmente sobre os fatores que as influenciam, são poucos. Com base em uma pesquisa qualitativa, foi mostrado que o tempo de viagem, o custo da viagem, a segurança, a pontualidade, o conforto, o contato social e a promoção da identidade são fatores potenciais de influência na escolha de uma carona (LIU *et al*., 2019).

Ademais, os veículos influenciam significativamente no meio ambiente. Embora as pessoas não consigam parar de usar carros completamente, elas ainda podem contribuir para a melhoria do ambiente usando os serviços de carona. Hyrecar (2020) identifica os seguintes benefícios ambientais e ecológicos das caronas:

- O compartilhamento de carro ajuda o público a usar menos carros e reduzir o congestionamento, porque as pessoas consideram mais barato alugar um carro;
- O compartilhamento de carro ajuda a reduzir a emissão de poluentes no ar porque menos carros na estrada significam que menos emissões são produzidas;
- O compartilhamento de carro ajuda a conservar fontes de energia não renováveis, porque menos carros exigem menos combustível.

Todos os aspectos citados, desde segurança até conforto serão tratados pelas funcionalidades do trabalho proposto, uma vez que a ideia pode ser expandida para caronas em geral, e não somente para os estudante da cidade de Quixadá.

2.2 Serviços de aplicação

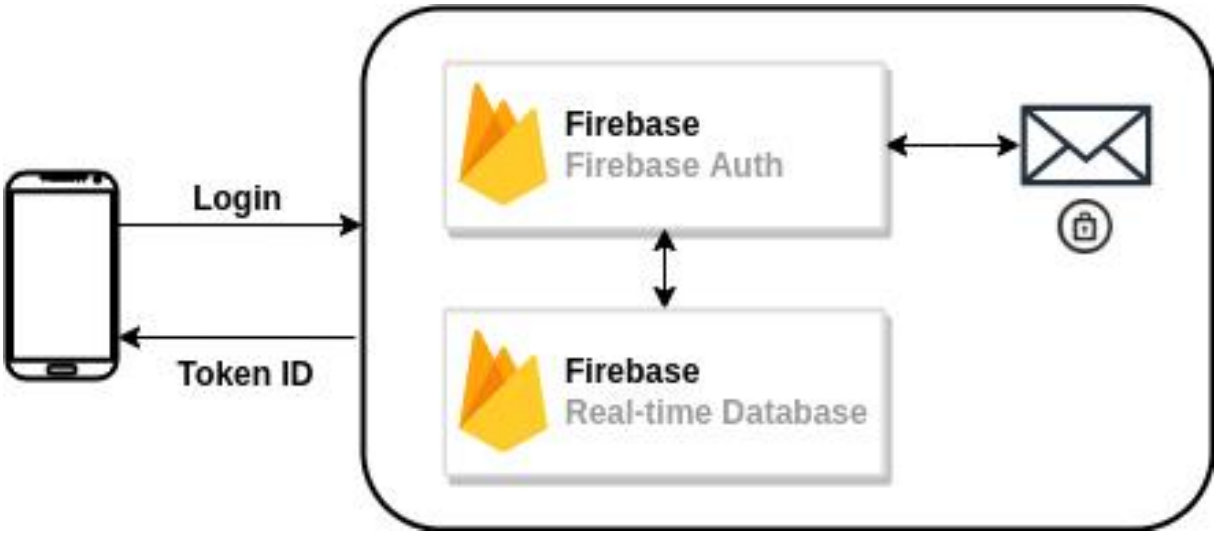
2.2.1 Registro e Autenticação

A maioria dos aplicativos móveis implementa algum tipo de autenticação de usuário. Embora parte da lógica de autenticação e gerenciamento de estado seja realizada pelo serviço de back-end. A autenticação é parte integrante da maioria das arquiteturas de aplicativos móveis, sendo importante entender suas implementações comuns (SWEN, 2020).

Existem diversas formas de registro e autenticação, as mais comuns em aplicativos mobile são a autenticação por e-mail e senha, autenticação por número de telefone e autenticação por redes sociais.

No momento do registro, isto é, no primeiro acesso do usuário, a aplicação sugere alguma das formas de autenticação para ser capaz de gerar um id único, nomeado de “ID Token” para o novo usuário. Assim, em todos os acessos, o sistema é capaz de distinguir cada usuário unicamente. A figura 2 a seguir mostra o processo de autenticação, no qual o usuário tem a opção de se identificar por meio do seu número de celular, e-mail ou redes sociais.

Figura 2 – Autenticação de usuário no Firebase



Fonte: Elaborado pelo autor.

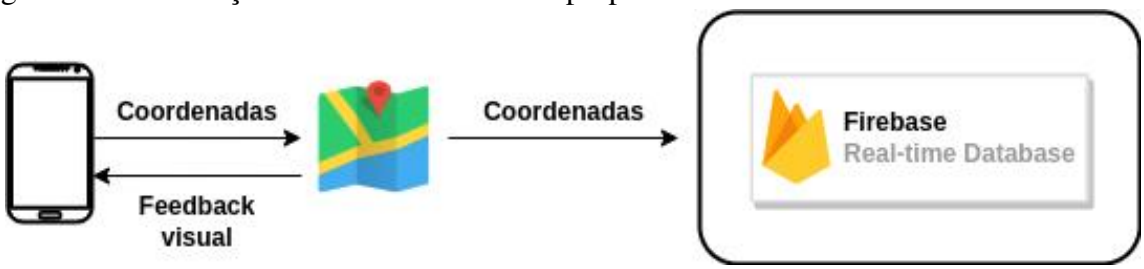
2.2.2 Localização em tempo real

Os sistemas de rastreamento e localização (TLS) estão espalhados em diferentes domínios, como gerenciamento de armazéns, gerenciamento de pátios, gerenciamento de frotas, etc. O TLS usa diferentes tecnologias, como códigos de barras ou GPS (PUŞCAŞIU, 2016).

Por causa de suas características, os *smartphones* são adequados para serem usados como terminais para um TLS. Os *smartphones* podem ser facilmente localizados usando métodos de triangulação ou atrasos nos sinais do transmissor / receptor, mas geralmente eles são equipados com um receptor GPS, portanto, são capazes de transmitir as coordenadas reais de sua localização. Além disso, os smartphones estão equipados com um acelerômetro de três eixos e um giroscópio, para que possam transmitir informações sobre o movimento e o status da pessoa monitorada.

Além disso, os *smartphones* estão usando sistemas operacionais como Android, iOS e Windows Mobile, o que permite a implementação de aplicativos internos capazes de usar as coordenadas de GSP livremente (PUŞCAŞIU, 2016). Em aplicativo internos no sistema Android, o GPS é normalmente gerenciado pela API do Google Maps, por ser uma interface completa e de fácil uso. É mais viável usar os serviços da Google do que implementar um gerenciador de GPS pessoal. A figura 3 a seguir mostra o uso do GPS e do Google Maps no sistema proposto neste trabalho, no qual as coordenadas do usuário autenticado podem ser enviadas diretamente para alguém de confiança, para que esta pessoa tenha noção da localização do usuário durante a viagem.

Figura 3 – Localização do usuário no sistema proposto



Fonte: Elaborado pelo autor.

2.2.3 Push Notifications

Nos smartphones, *Push notifications* são normalmente mensagens de alerta enviadas ao dispositivo do usuário para notificá-lo diretamente na tela do aparelho. Push Notifications têm sido amplamente usadas em plataformas mobile para fornecer todos os tipos de informações

aos usuários de aplicativos (Wang, *et al.*).

A plataforma de *Push notification* permite que os usuários recebam mensagens e anúncios importantes dos servidores para os clientes sem a necessidade de o cliente permanecer sempre conectado ao sistema. As notificações geralmente ocorrem quando a ocasião exige (por exemplo, emergências, novo e-mail). Os dispositivos móveis, em particular os smartphones, tornaram-se uma engrenagem importante na interação com os usuários e na garantia do fluxo de dados entre o usuário e a aplicação.

No contexto do trabalho proposto, o sistema de *Push notification* propõe anunciar o usuário de ações como eventos de viagem e de chat.

2.2.4 Sistema de rating

O sistema de *rating* e classificação foi desenvolvido para ajudar os leitores a saber mais sobre os produtos, serviços ou uma empresa. É um processo em que os clientes compartilham seus comentários honestos. Isso ajuda outras pessoas a entender as verdadeiras características de um produto, bem como seus diferentes aspectos. O *feedback* dos usuários desempenha um papel importante no processo de tomada de decisão dos outros que também usam o produto, ou pensam em usar alguma hora. O sistema de rating e a classificação são um canal aberto através do qual as pessoas podem compartilhar suas opiniões sobre um produto ou empresa (WILLIAM, 2012).

Em sistemas mobile, o sistema de reputação se configura, geralmente, por votações que vão de uma escala que vai de 1 a 5, conhecida também como Escala Likert. Trata-se de uma das metodologias mais populares e, consequentemente, mais indicadas para realizar pesquisas de opinião (FRANKENTHAL, 2017). A coleta dessas informações auxilia a aplicação a melhorar seus serviços.

No contexto do trabalho proposto, o sistema de *rating* propõe auxiliar os usuários a expor suas opiniões sobre as viagens, podendo avaliar tanto o motorista quanto outros passageiros.

2.3 Padrões de projeto

Na Engenharia de Software, um padrão de projeto é uma solução geral para um problema comum no desenvolvimento de software. Um padrão de projeto não é o design final do sistema, que pode ser transformado diretamente em código. É uma descrição ou modelo de

como resolver um problema que pode ser usado em muitas situações diferentes.

Os padrões de projeto podem acelerar o processo de desenvolvimento, fornecendo paradigmas de desenvolvimento testados e comprovados. O design eficaz do software requer a consideração de problemas que podem não se tornar visíveis até mais tarde na implementação. A reutilização de padrões de projeto ajuda a evitar problemas sutis que podem causar grandes problemas e melhora a legibilidade do código para programadores e arquitetos familiarizados com os padrões (SHVETS, 2018).

Shvets (2018) também argumenta que, frequentemente, as pessoas só entendem como aplicar certas técnicas de design de software a certos problemas. É difícil aplicar essas técnicas a uma gama mais ampla de problemas. Os padrões de projeto fornecem soluções gerais, documentadas em um formato que não requer detalhes vinculados a um problema específico.

Um dos benefícios mais notáveis dos padrões de projeto é a facilidade de entendimento e manutenção do código fonte. A padronização contribui diretamente para a redução do acoplamento e para o aumento da coesão de nossas classes, proporcionando uma redução de custo e tempo em nossas futuras manutenções (NASCIMENTO, 2018). Ademais, como outros benefícios, podemos destacar:

- Padronização da estrutura das classes;
- Soluções reusáveis;
- Facilidade na comunicação da equipe.

2.3.1 Padrões MVC e MVP

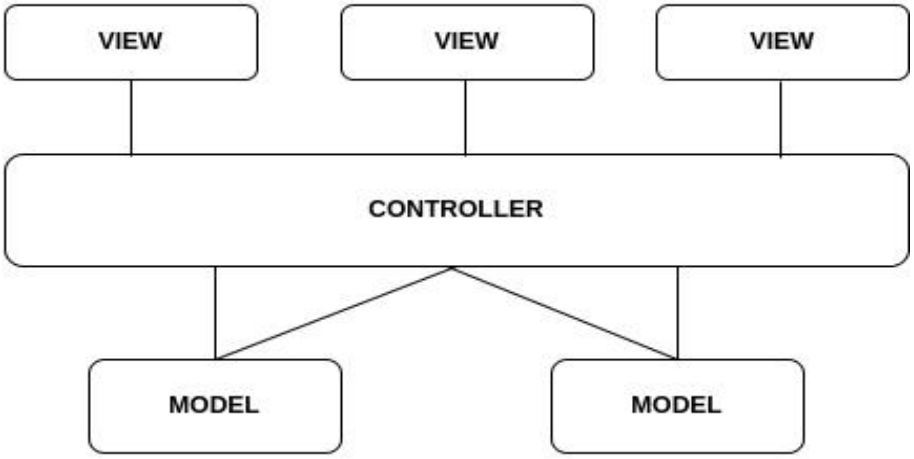
MVC *(Model-View-Controller)* é um padrão de design que se concentra em separar a interface da aplicação (*View*) de sua camada de negócios (*Model*), para isso, o padrão define três componentes:

- **Model:** Models contêm as definições do dados da aplicação, como os atributos de cada dado, além dos procedimento que podem alterar tais atributos;
- **View:** Views são o que é exibido para o usuário. As views também lidam com qualquer interação que um usuário possa ter com a aplicação, como eventos de clique, por exemplo. Uma view deve ser responsável apenas por exibir coisas e não deve conter nenhuma lógica de negócio.
- **Controller:** Controllers são uma maneira de conectar models e views. O controller é que

fica responsável por qualquer funcionalidade lógica que manipula os models.

O padrão MVC se tornou muito utilizado em aplicações orientadas a objeto, como explicado por (Deacon, 2009), pela sua capacidade de separação de tarefas, o que torna a refatoração do código menos trabalhosa de se fazer. A figura 4 a seguir mostra como os componentes do padrão MVC se relacionam.

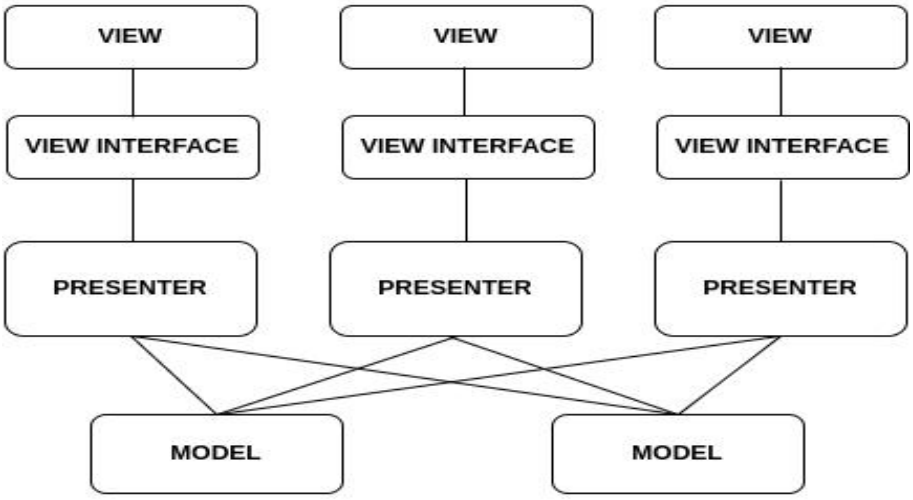
Figura 4 – Relação dos componentes no padrão MVC



Fonte: Elaborado pelo autor.

O MVP (*Model-View-Presenter*) é um padrão de projeto que deriva do MVC, os componentes de *model* e *view* continuam presentes, com a adição de uma nova camada: o presenter. O presenter trata das atualizações da interface do usuário com base nas alterações no modelo de dados e também processa as entradas dos usuários. Além disso, a comunicação entre as *views* e os *presenters* da aplicação normalmente se dá via interfaces, no qual os componentes da view não tem acesso direto à implementação dos métodos no *presenter*, e vice-versa. A vantagem disso é desacoplamento das chamadas das função com suas respectivas implementações, já que temos um presenter para cada view, o que torna, por exemplo, os testes e grandes refatorações mais simples. A figura 5 a seguir mostra as relações entre os componentes em um padrão MVP.

Figura 5 – Relação dos componentes no padrão MVP



Fonte: Elaborado pelo autor.

2.3.2 Data Binding

Data binding, ou vinculação de dados, é o processo que estabelece uma conexão entre a interface do usuário do aplicativo e os dados que ela exibe. Se a associação tiver as configurações corretas e os dados fornecerem as notificações adequadas, quando os dados alterarem seu valor, os elementos vinculados aos dados refletirão as alterações automaticamente. Isso também pode significar que, se uma representação externa dos dados em um elemento for alterada, os dados subjacentes poderão ser atualizados automaticamente para refletir a alteração.

Na plataforma Android, a vinculação de dados é o processo de integração de exibições em um layout *XML* com objetos de dados. A *Data Binding Library* é responsável por gerar as classes necessárias para este procedimento.

Ao contrário de outros tipos de arquivos *XML* de layout, os arquivos *XML* de layout de vinculação de dados começam com uma marca de layout raiz, que é seguida por um elemento de dados. Cada arquivo de layout é então associado a uma classe *Data Binding* que foi gerada pela Biblioteca (Pandya, 2021).

A combinação de *Data binding* com o padrão MVP pode resultar em uma estrutura muito limpa e um projeto de fácil manutenção. A vinculação de dados economiza linhas de código. (Fatoye, 2017).

3 TRABALHOS RELACIONADOS

Os trabalhos abortam a temática do desenvolvimento de aplicações voltadas para caronas, tais trabalhos se diferenciam normalmente pelas funcionalidades implementadas. Entre eles estão o trabalho de Binu e Viswaraj (2016), Kamaruddin e Rozlis (2019) , Zainab *et al*. (2015) e Nale *et al*. (2016).

3.1 *Android based application for efficient carpooling with user tracking facility*

O trabalho de Binu e Viswaraj (2016) propôs uma aplicação eficiente para a plataforma Android buscando a implementação de duas principais funcionalidades: localização de usuário e detecção de anomalias no tráfego.

A função de localização de usuário está inclusa como parte dos métodos de segurança do aplicativo, já a detecção de anomalias é usada como uma forma de gerar a melhor rota para o motorista. Para tal, os autores sugerem um algoritmo recursivo chamado de *Recursive Expectation-Maximization Algorithm* ou algoritmo EM recursivo, além de provar sua corretude.

EM recursivo é um algoritmo usado no campo da estatística como uma forma de encontrar a probabilidade máxima de um parâmetro qualquer, dada uma distribuição de probabilidades. (GUPTA, M. R.; CHEN, Y. 2011, p. 224).

No trabalho, o EM recursivo recebe constantemente os dados dos usuários, que informam sobre qualquer obstrução no tráfego, e analisa a anomalia para determinar sua veracidade.

3.2 *Carpooling Application for Android Focusing on Authentication and Traffic Analysis*

O trabalho de Zainab *et al*. (2015), assim como Binu e Viswaraj (2016), propôs uma aplicação Android para caronas. O trabalho se difere por buscar uma aplicação mais rígida no quesito segurança. Os autores realizaram uma pesquisa com perguntas sobre usabilidade, escalabilidade e até mesmo design, e concluem que a autenticação com imagem e comunicação entre usuários são as formas preferidas pela maioria dos participantes da pesquisa. Assim, foi sugerido um sistema com autenticação eficiente e com um chat interno onde os membros possam discutir qualquer detalhe da carona.

A autenticação sugerida no trabalho de Zainab *et al* (2015). além de requisitar uma imagem do usuário no momento da criação da conta, também sugere a integração da identidade do usuário com suas redes sociais, para garantir mais segurança para quem procura caronas.

Segundo a pesquisa realizada no trabalho, os participantes se sentem mais seguros quando podem verificar a existência do motorista pelas redes e mídias sociais diversas, como Facebook, principalmente.

Já o chat incluso na aplicação entra como uma forma de comunicação entre os participantes da carona e o motorista, onde eles podem discutir detalhes da carona, como o local de encontro, possíveis pagamentos pelo combustível até peso de bagagens, ou até mesmo garantir lugar para algum outro colega que também vai para o mesmo destino.

3.3 *UiTM Share Ride: Requirements Validation, Design and Development of a Campus Ride-Sharing Mobile Application*

O trabalho de Kamaruddin e Rozlis (2019) também discute sobre a criação de uma aplicação Android para caronas. O trabalho se difere por ter sido pensado exclusivamente para os discentes da Universidade Tecnológica de Mara (UiTM), na Malásia, que lidam com o problema de lotação de veículos, no qual existem poucas vagas para quem se desloca com veículos próprios.

Aqui, os autores desenvolveram o sistema usando os serviços do *Google Maps*, e do banco de dados não relacional *Firebase*, mostrando que tais tecnologias são eficientes na criação de um aplicativo voltado para caronas. Além disso, toda a aplicação foi escrita na linguagem de programação Java.

Firebase é uma plataforma de serviços criada pela empresa Firebase, Inc. em 2011 e comprada pela *Google* em 2014 (TAMPLIN, 2014). Dentre os vários serviços disponibilizados pela plataforma, um deles é o banco de dados orientado a documentos, conhecido como *Firebase Realtime Database*, serviço usado no desenvolvimento da aplicação de Kamaruddin e Rozlis (2019).

3.4 *Real-Time Carpooling Application for Android Platform*

A aplicação de Nale *et al* (2016). tem como objetivo criar um sistema amigável ao usuário e que ofereça uma oportunidade para compartilhar carros. O foco é criar um sistema que ajudasse os usuários a fazer upload, visualizar e registrar viagens de curta e longa distância na mesma cidade. O aplicativo se concentra principalmente na segurança do usuário.

O trabalho se difere dos demais trabalhos citados por propor um sistema que busca

ser mais escalável. Para tal, a aplicação tem seu *back-end* implementado com tecnologias voltadas comumente para o desenvolvimento Web. O *software* proposto possui um servidor com um banco de dados relacional (SQL), no qual todos os dados dos usuários são transferidos no formato JSON para o servidor, e vice-versa, com uma conexão PHP.

No sistema de Nale *et al*. (2016), basicamente existem dois módulos que são o módulo do motorista e o módulo do passageiro, ambos precisam registrar seus dispositivos para que o servidor possa obter seus detalhes de localização usando GPS e outros detalhes de contato, como nome, número de celular, tipo de veículo, capacidade do veículo e etc.

3.5 Trabalho Proposto

Estre trabalho propõe, assim como os trabalho descritos, o desenvolvimento de uma aplicação Android voltada para caronas. E, do mesmo modo que Binu e Viswaraj (2016), desenvolver a funcionalidade de localização em tempo real dos usuários. E similarmente a Zainab *et al*. (2015), incluir na aplicação um sistema eficiente de autenticação dos usuários, além da implementação de um chat incluído no aplicativo para assegurar a comunicação dos participantes de uma carona.

Tabela 1 – Comparação entre os trabalhos relacionados e o trabalho proposto

Trabalho	**Localização em tempo real**	**Autenticação**	**Chat**	**Uso do Google Maps**	**Uso do Firebase**
Binu e Viswaraj (2016)	Sim	Não	Não	Não	Não
Zainab et al. (2015)	Não	Sim	Sim	Não	Não
Kamaruddin e Rozlis (2019)	Não	Sim	Não	Sim	Sim
Nale et al. (2016)	Sim	Sim	Não	Sim	Não
Este trabalho	**Sim**	**Sim**	**Sim**	**Sim**	**Sim**

Fonte: Elaborado pelo autor

4 METODOLOGIA

Para que os objetivos propostos neste trabalho fossem alcançados, um conjunto de etapas foram adotadas, conforme ilustrado na Figura 6, descrito nas próximas seções e enumeradas a seguir.

1. Definições de regras de negócio e histórias de usuário, bem como arquitetura e padrões de projeto
    a) Definição do padrão de projeto;
    b) Definição de *assets*, família de ícones e fontes;
    c) Atualização da documentação de *software*.
2. Integrações com sistemas externos e implementações
    a) Conexão com sistema *Firebase*;
    b) Integração e consumo de API's de localização;
    c) Implementação de funcionalidades.
3. Testes e validações

Figura 6 – Passos metodológicos adotados

Fonte: Elaborado pelo autor.

4.1 Definição dos padrões de projeto

O padrão de projeto escolhido para o desenvolvimento foi o MVP, ou *Model-View-Presenter*, que é um padrão de projeto que deriva do MVC. Além disso, também foi usado o conceito de *Data Binding* para a relação entre os dados e a parte visual da aplicação.

4.2 Definição de *assets*, família de ícones e fontes

Os *assets* escolhidos, como família de fontes e ícones, derivam do *Material Design* da Google, que define um padrão baseado em boas práticas de *design* de interface, que define um padrão consistente e conciso para todos os componentes da aplicação. Os ícones tem seu desenho e escala baseados nessas boas práticas, assim como a fonte do sistema escolhida, a fonte *Poppins*.

4.3 Atualização da documentação

A documentação encontrada do Apêndice A do trabalho descreve as especificações do *software* proposto, como os diagramas que relacionam as entidades do sistema, como tais entidades se comportam em determinados cenários e quais desses cenários são mais críticos que outros. Assim, com a evolução no desenvolvimento da aplicação, tais histórias, cenários e entidades podem mudar na forma como se comportam e interagem. Portanto, atualizar a documentação, para que o desenvolvimento esteja mais organizado é de grande importância.

4.4 Conexão com o *Firebase*

Hoje em dia, desenvolvedores de aplicativos independentes e *startups* não conseguem ter especialistas *back-end* dedicados para manter tudo funcionando. Há uma incessante necessidade de desenvolver e testar rapidamente os *apps* para se adequarem ao mercado de produtos antes de investir em suas próprias infra-estruturas e serviços. Para solucionar da melhor forma este obstáculo, surgiu o *MBaaS* (Mobile Back-End as a Service), uma plataforma fácil e rápida que provê serviços, os quais a maioria dos apps necessitam. (BRAZIL, 2017). Atualmente, existem vários *MBaaS* disponíveis, cada um oferecendo diversos serviços, o *MBaaS* escolhido para o projeto foi o *Firebase*, por fornecer os serviços necessários para o trabalho proposto, e por ter uma documentação ampla e disponível, o que facilita com sua integração. Os serviços do *Firebase* que foram consumidos estão listados a seguir.

1. Autenticação;
2. Banco de dados em tempo real;
3. Serviço de notificações;
4. Laboratório de testes.

4.5 Conexão com API's necessárias

APIs são ferramentas vitais para o desenvolvimento, o consumo de uma API permite que recursos de uma plataforma sejam utilizadas por outros, geralmente por meio de *endpoints* que determinam a comunicação com as plataformas fornecedoras dos serviços. Além do consumo dos serviços da plataforma *Firebase* que foram citados, o consumo de outras API's foram necessárias, é o caso do serviço de localização para o mapa, ou *Maps SDK*, que permite o uso do *Google Maps* na aplicação para a localização em tempo real.

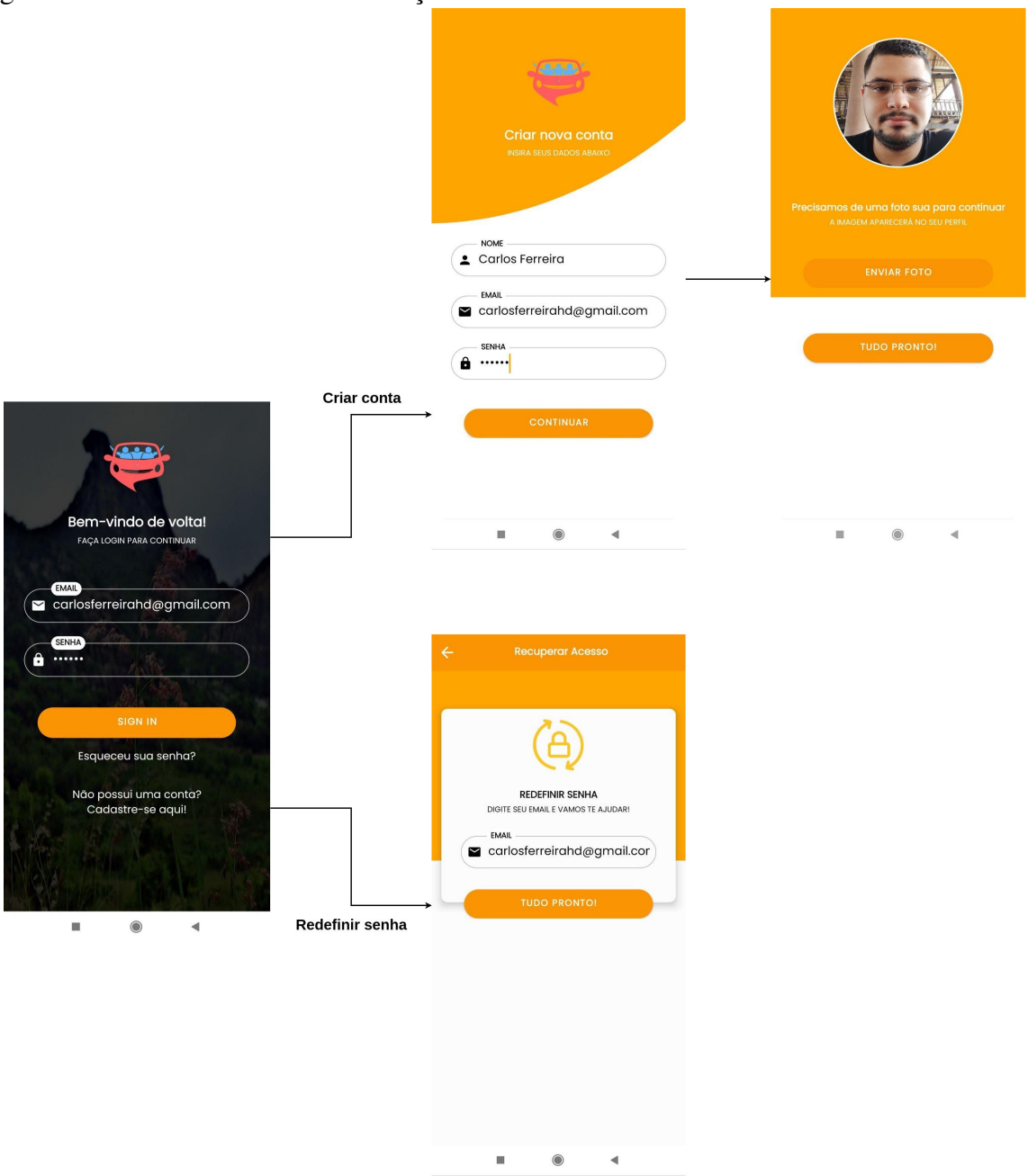
4.6 Implementeção das funcionalidades

Com as conexões aos serviços que serão usados na aplicação, as funcionalidades começam a ser implementadas, o serviço de autenticação foi o primeiro a ser consumido, com a autenticação por e-email e senha sendo escolhida como forma de registro, logo após, o serviço de sistema de banco de dados para salvar as informações que o usuário fornece foi integrado, com os formulários para preenchimento de informações pessoais. Após o de banco de dados, foi consumido o de notificações, para a implementação do *chat* que fica disponível durante viagens, e por fim o serviço de localização é integrado ao sistema.

4.6.1 Telas voltadas à autenticação

Como descrito anteriormente, as primeiras telas criadas foram as voltadas ao fluxo de autenticação, aqui foram implementadas as funcionalidades de *login*, que recebe e-mail e senha como *input*, logo após, a tela de criação de conta, que recebe as informações pessoais, como nome, e-mail e senha de cadastro, bem como uma imagem do usuário que é obrigatória para a criação de conta. A partir deste ponto, o usuário recebe um e-mail para que ele possa confirmá-lo. Logo após isso, a funcionalidade de redefinição de senha foi criada, que envia um e-mail para o usuário para que ele possa recuperar seu acesso. Com o *login* concluído, a integração com o *logout* pôde ser desenvolvida, para que o usuário encerre sua sessão. A figura 7 a seguir mostra o fluxo de autenticação do sistema.

Figura 7 – Telas do fluxo de autenticação



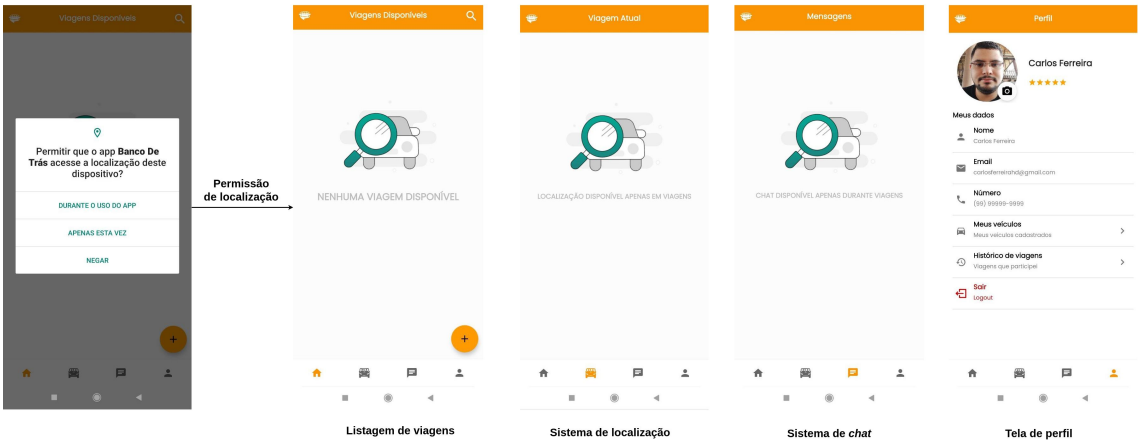
Fonte: Elaborado pelo autor.

4.6.2 Telas principais da aplicação

Após realizar a autenticação, temos o consumo do banco de dados em tempo real, para a persistência dos dados dos usuários, como seus veículos e suas viagens que dependem do sistema de localização do celular, e para tal, é necessária a implementação das requisições de permissão de acesso à localização, que é usada sempre durante as viagens. Em um cenário sem viagens disponíveis, o usuário vê a implementação do *Empty State* da tela de listagem de viagens, informando que não há viagens disponíveis no momento, bem como os *Empty States* das telas de

chat e viagem atual, que só retornam dados do banco quando o usuário está participando de uma viagem no momento. Também temos o perfil pessoal, com as informações dadas na criação de conta. A figura a 8 seguir mostra essas quatro telas principais da aplicação

Figura 8 – Telas principais



Fonte: Elaborado pelo autor.

4.6.3 Telas voltadas às informações do usuário

Apartir da tela de perfil, o usuário pode visualizar e editar algumas informações pessoais, como número de telefone, seus veículos e seu histório de viagens, logo tais telas foram implementadas. Na visão de um banco de dados não relacional, que é o caso do *Firebase*, todas as informações são referentes ao nó de usuário e suas ramificações. A figura 9 mostra o fluxo entre essas telas.

4.6.4 Telas voltadas a criação de nova viagem

Com o *CRUD* de veículos de usuário pronto, a funcionalidade de criação de nova viagem começa a ser implementada, que é disparada pelo botão flutuante na tela de listagem. As informações para a criação de uma nova viagem são descritas a seguir:

1. Ponto de origem;
2. Ponto de destino;
3. Quantidade de assentos vagos;
4. Veículo da viagem;
5. Horário de partida.

Além disso, algumas regras foram definidas para a criação de viagem, tais regras estão descritas abaixo, bem como no documento de especificação de *software* no apêndice A.

1. Usuário deve ter um veículo cadastrado;
2. Usuário não pode estar participando de outra viagem;
3. A viagem deve conter assentos disponíveis.

Uma vez criada uma nova viagem, os detalhes que aparecem na tela de listagem podem variar, dependendo se o usuário é o motorista ou não. na visão do motorista, o sistema foi implementado tal que ele pode cancelar sua própria viagem, senão, ele pode pedir para participar. Apartir da criação de uma nova viagem, o chat fica disponível para os participantes, mas a localização só se inicia quando o horário informado pelo usuário durante a criação chegar. A figura 10 mostra o fluxo descrito acima.

4.6.5 Telas voltadas ao envio de mensagens e notificações

Como descrito na seção anterior, uma vez que uma viagem é cadastrada, o chat fica disponível para os participantes de uma viagem, assim qualquer um que esteja participando pode enviar mensagens, permitindo a comunicação entre todos. Além disso, uma vez que uma mensagem é enviada, a aplicação, por meio do sistema de *tokens* que o *Firebase* oferece para o envio de notificações permite que os usuários sejam notificados das últimas mensagens enviadas no chat. A figura 11 mostra o chat e as notificações entre participantes de uma viagem.

4.6.6 Telas voltadas ao serviço de localização e finalização de viagem

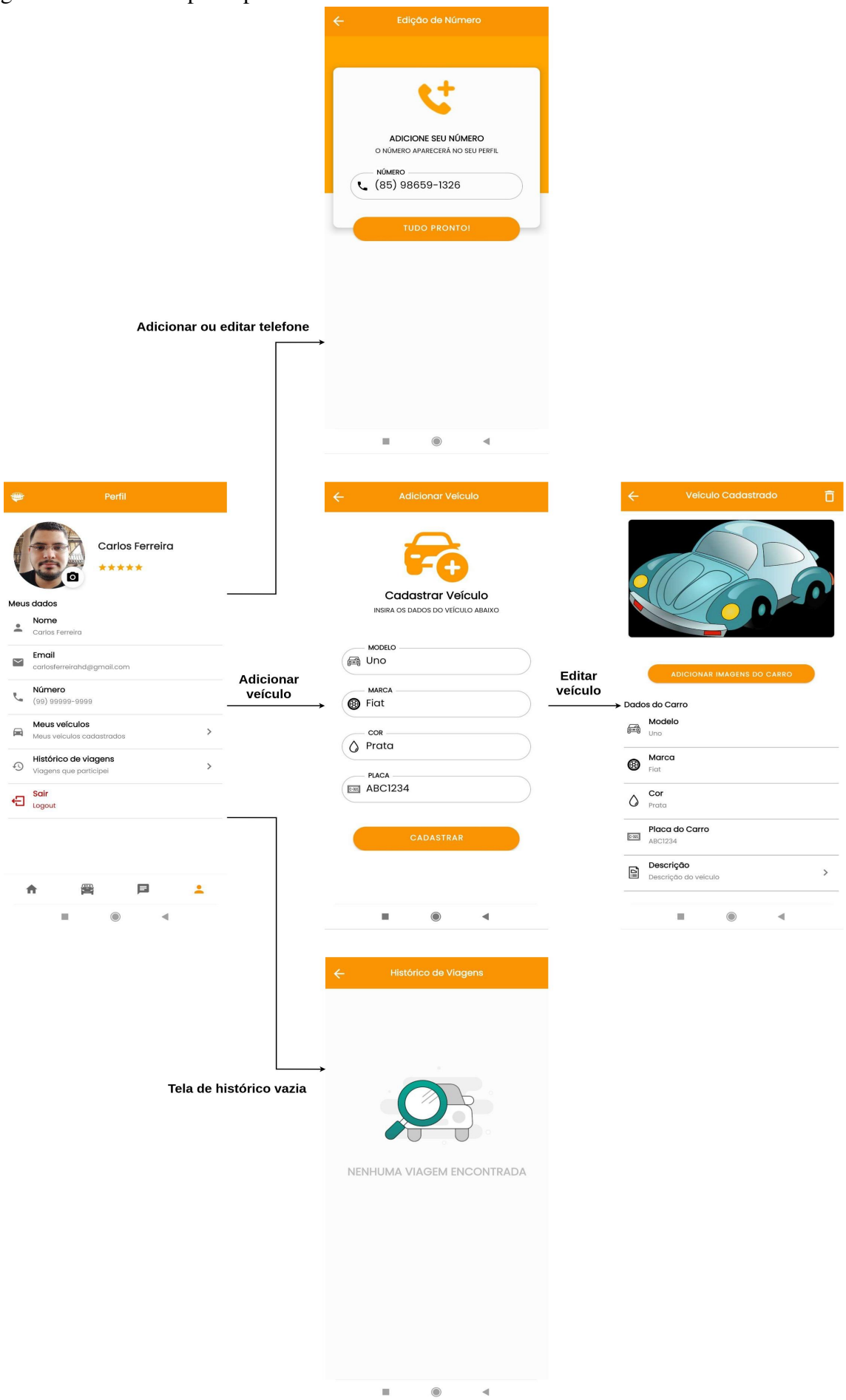
Como dito em seções anteriores, quando uma viagem é criada, o serviço de localização só é disparado no momento que o horário definido na criação da viagem chega, portanto a tela de localização foi implementada tal que o seu *Empty State* mostra aos participantes o horário de início. O serviço de localização foi implementado usando a API do *Google Maps*, e todo o ambiente que ele provê para a função de *GPS*. Assim, quando uma viagem começa, a localização de todos os participantes é atualizada a cada 500 milissegundos, e a cada 5 minutos, o sistema deve enviar as informações de latitude e longitude capturadas pelo *GPS* para o banco de dados. Além disso, o motorista pode finalizar a viagem quando chegar ao seu destino por meio da interface, no qual um botão na tela de localização pode ser clicado para finalizar a viagem, e uma vez finalizada, as coordenadas de latitude e longitude são atualizadas mais uma vez no banco, bem como os detalhes da viagem no histórico de cada participante. A figura 12 mostra o

serviço de localização, bem como o *layout* do mapa.

4.6.7 Telas voltadas ao histórico de viagens e avaliação dos participantes

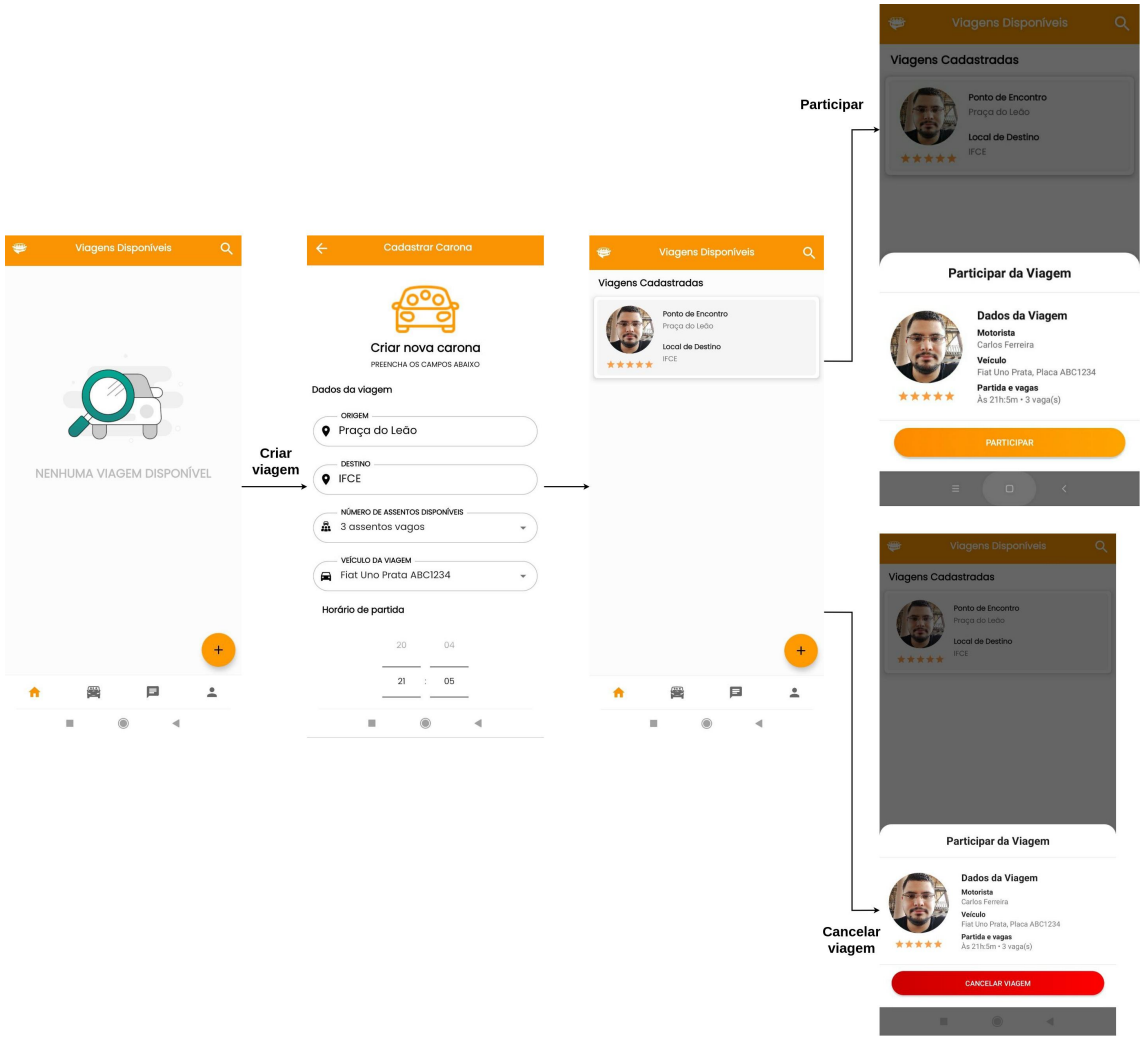
Com uma viagem finalizada, chega a implementação da listagem do histórico e do serviço de avaliação de usuários. Para o serviço de avaliação ou *rating*, a implementação foi feita tal que todos os participantes pudessem votar em todos os outros, em uma escala de 1 a 5, também conhecida como escala de *Likert*. Um usuário começa com cinco estrelas, a pontuação máxima, e a cada novo voto, uma média é feita usando esses novos valores, e então a nova nota do usuário é definida. A figura 13 mostra a tela de listagem do histórico bem como o sistema de avaliações.

Figura 9 – Fluxos do perfil pessoal



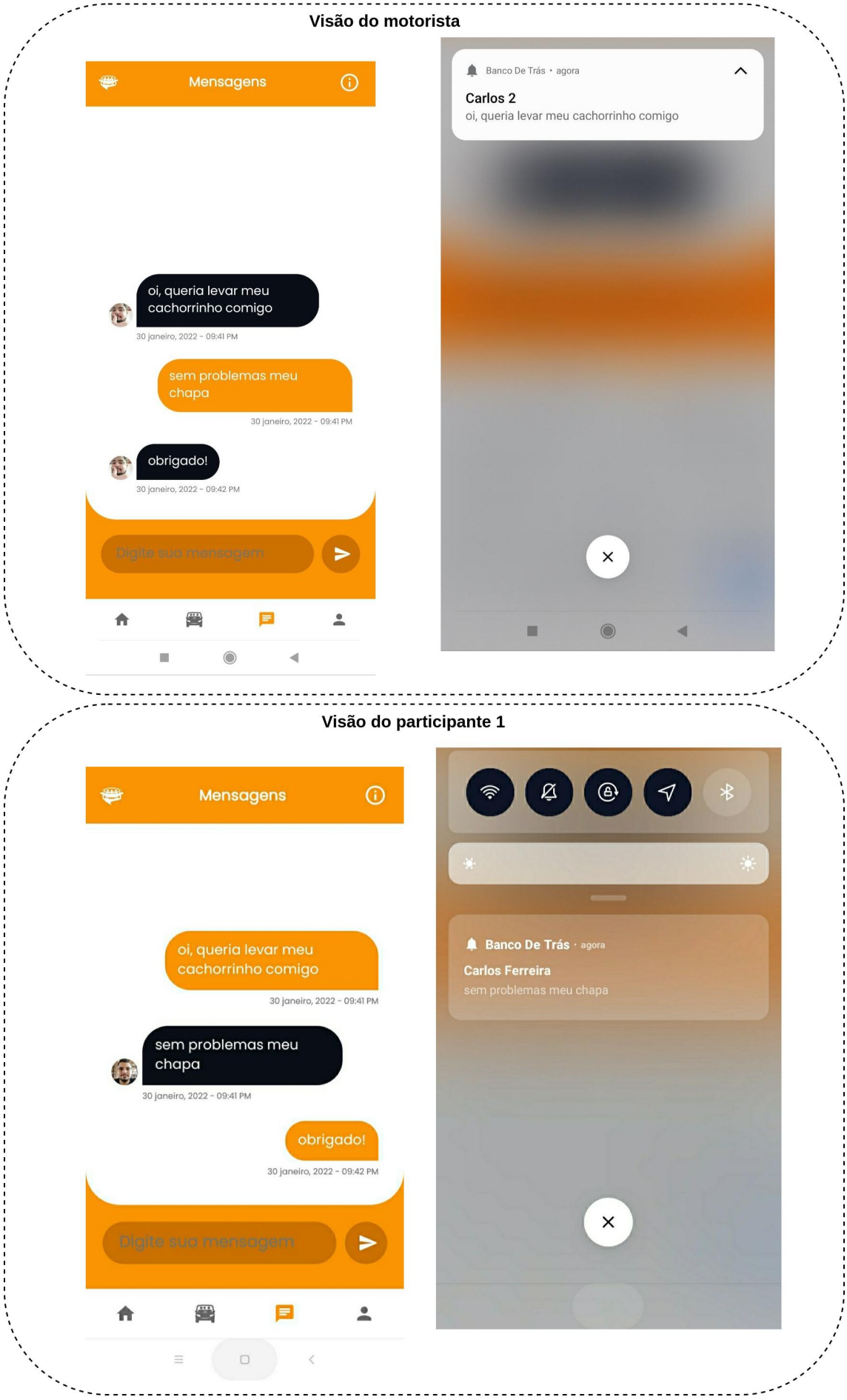
Fonte: Elaborado pelo autor.

Figura 10 – Fluxo de criação, listagem e participação de nova viagem



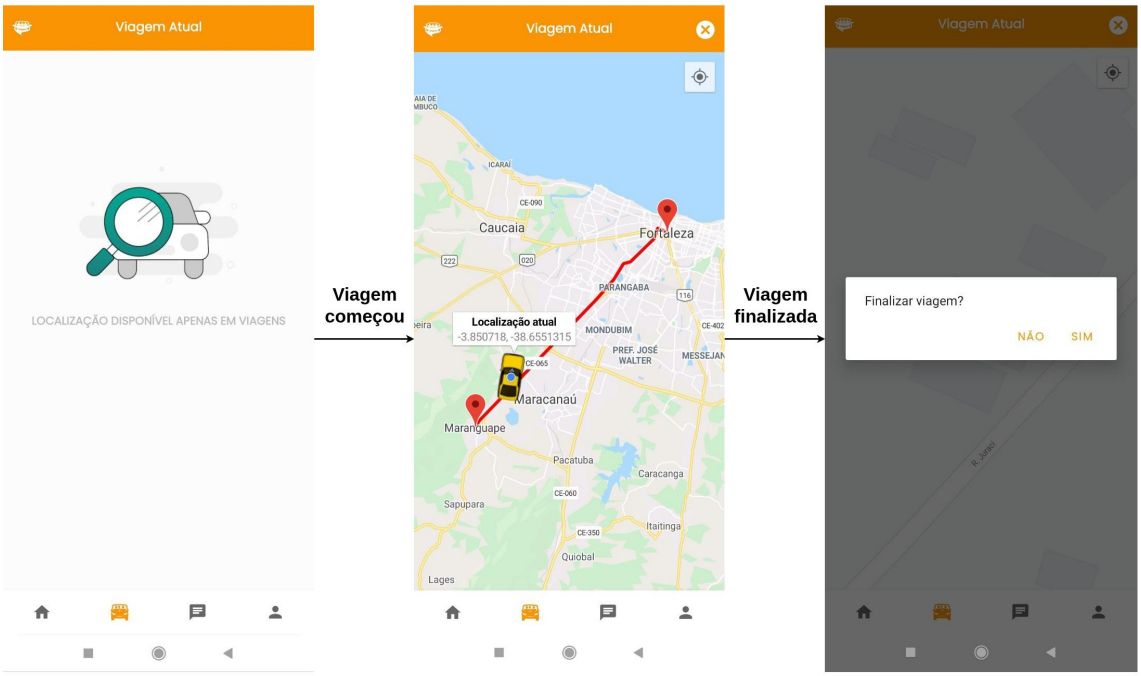
Fonte: Elaborado pelo autor.

Figura 11 – Fluxo de mensagens e notificações



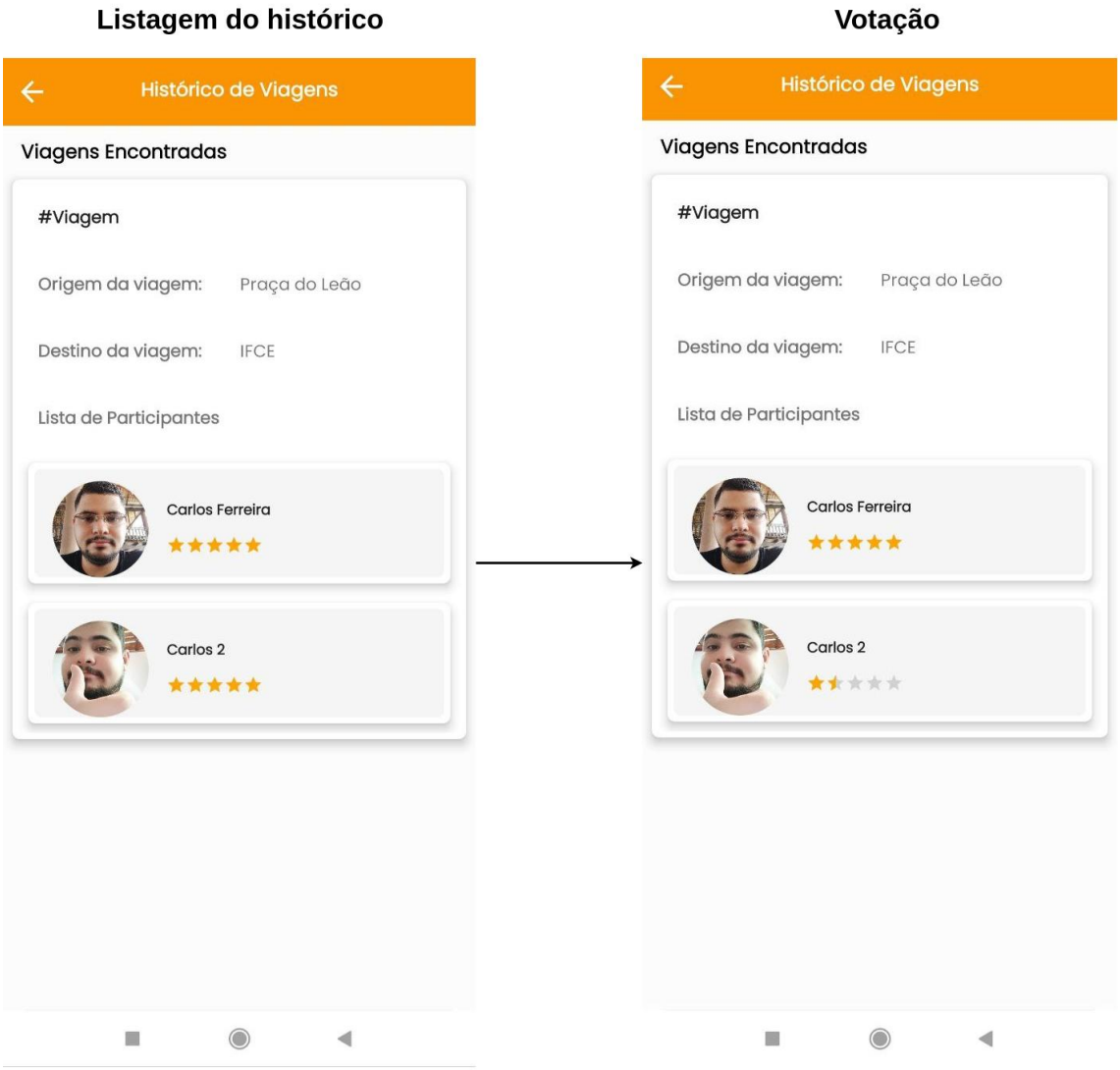
Fonte: Elaborado pelo autor.

Figura 12 – Fluxo de localização de viagem



Fonte: Elaborado pelo autor.

Figura 13 – Fluxo de histórico e *rating*



Fonte: Elaborado pelo autor.

5 EXPERIMENTOS E RESULTADOS

Neste capítulo, serão apresentados a descrição do ambiente de execução, as ferramentas utilizadas e os resultados obtidos.

5.1 Ambiente de execução

O ambiente de testes é constituído de um notebook para desenvolvimento e coleta de dados, como a visualização dos tempos de execução coletados, e dois celulares para instalação e execução do aplicativo, com as respectivas configurações mostradas na tabela 2 a seguir.

Tabela 2 – Especificações técnicas dos aparelhos usados nos testes

Aparelhos	**CPU**	**Memória**	**Sistema operacional**	**GPU**
Notebook	Intel Core i5-10210U 4.2 GHz	8GB	Ubuntu 20.04 LTS	Intel UHD Graphics
Xiaomi Redmi 5	Snapdragon 450 1.8 GHz 8 Core	2GB	Android 8.1 Oreo	Adreno 506
Xiaomi Redmi Note 10 Pro	Snapdragon 732G 2.3 GHz Kryo	6GB	Android 11	Adreno 618

Fonte: Elaborado pelo autor.

5.2 Ferramentas utilizadas

As ferramentas utilizadas para os testes foram a biblioteca *TimingLogger* para Android, responsável pelo coleta de dados referentes ao tempo de execução dos métodos testados, e os serviços *Performance* e *Test Lab* do *Firebase* para monitoramento dos serviços como o de notificações e tempo de carregamento da aplicação.

5.3 Resultados encontrados

Nesta seção serão apresentados os resultados para os testes de cada fluxo de forma isolada. Cada teste foi executado trinta vezes, pela significância estatística.

5.3.1 Fluxos relacionados à autenticação da aplicação

Para esta seção, os tempos de execução para os fluxos de autenticação da aplicação foram testados. As funcionalidades de criação de conta e *login* apresentaram um valor de tempo mediano, como mostrado na figura 14.

5.3.2 Fluxos relacionados ao carregamento e edição de perfil

Os tempos de execução voltados ao carregamento e edição de perfil mostram que a edição dos dados é uma operação mais custosa, enquanto o carregamento consegue ser mais eficiente, como mostrado na figura 15.

5.3.3 Fluxos relacionados ao CRUD de veículo

Os tempos de execução relacionados ao CRUD de veículos mostram que as operações relacionadas à criação de veículo são mais custosas que as outras operações testadas, como mostrado na figura 16.

5.3.4 Fluxos relacionados ao CRUD de viagem

Os tempos de execução voltados ao CRUD de viagens mostram que as operações para o cancelamento de viagem são eficientes, enquanto as operações para a listagem são mais custosas, como mostrado na figura 17.

5.3.5 Fluxos relacionados ao Chat, Mapa e Serviço de notificações

Nesta seção, os tempos de execução das funções relacionadas aos fluxos das funcionalidades de chat, mapa e o serviço de notificação são testados. A plataforma *Test Lab* do *Firebase* foi usada para a coleta dos tempos referentes ao serviço de notificações. As funções do mapa possuem tempo de carregamento mais alto que as outras funções, em ambos aparelhos, seguidos dos tempos para as notificações, como mostrado na figura 18.

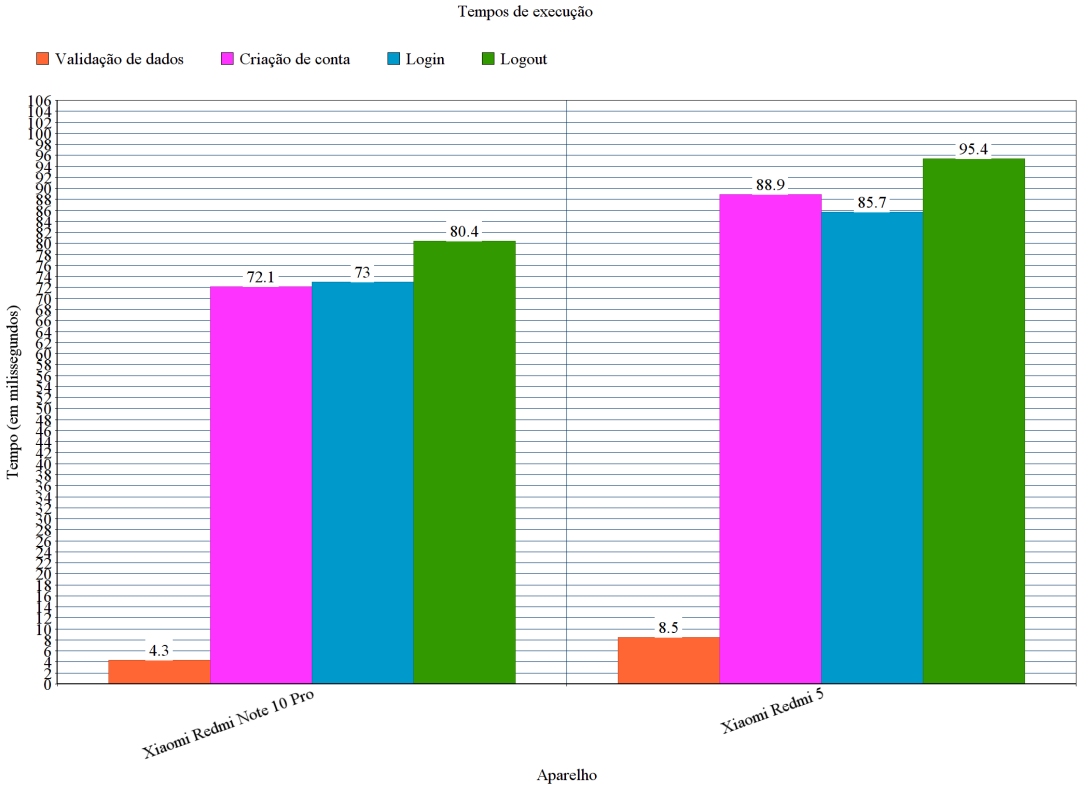
5.3.6 Fluxos relacionados ao histórico e avaliações de viagens

Os tempos de execução voltados para os fluxos de listagem de histório e avaliação dos participantes de uma viagem mostram que as funções voltadas para a listagem do histórico são mais custosas, seguidas das funções relacionadas ao carregamento das avaliações, como mostrado na figura 19.

5.4 Interpretação dos dados

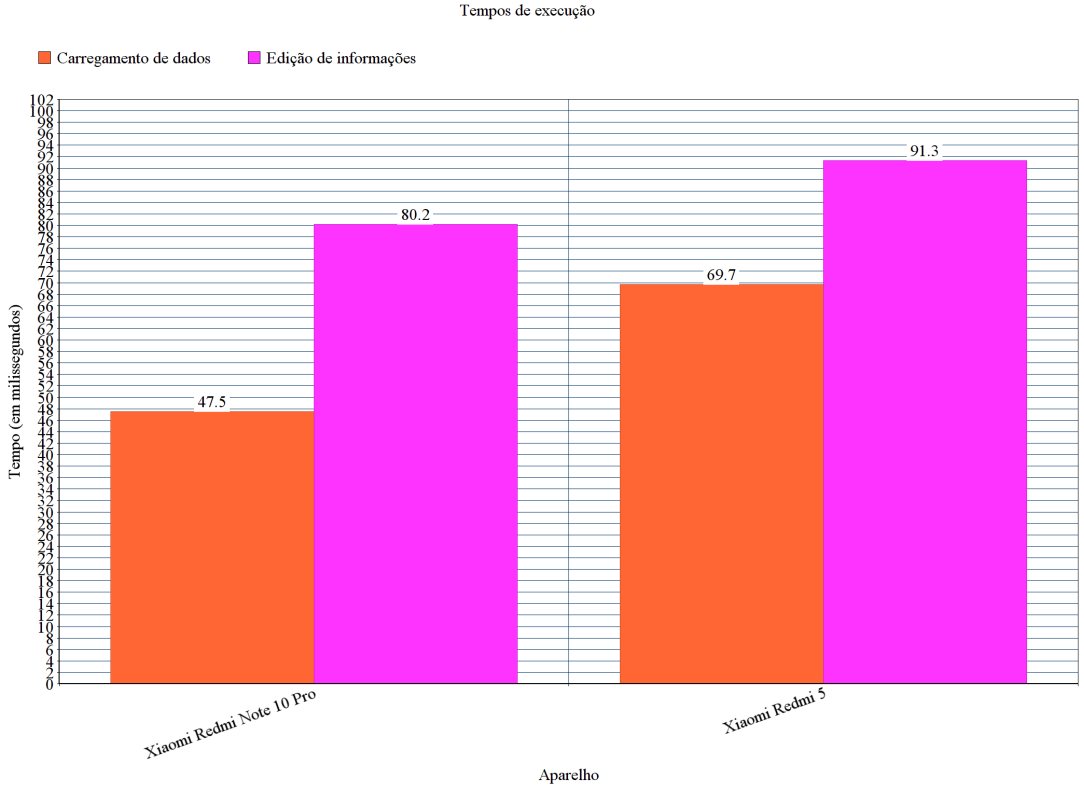
A coleta de dados dos testes foi focada no tempo de execução das funcionalidades. Como o padrão de projeto adotado foi o *MVP*, a implementação dos testes ocorreu de maneira linear, onde cada método dos *presenters* representam uma tarefa a ser considerada. A primeira observação é de que o tempo de execução de um fluxo é normalmente menor no aparelho com melhores especificações técnicas, além disso, métodos em que a aplicação deve carregar os dados do banco normalmente demoram mais para ser finalizados por completo, seguidos dos métodos de edição de dados. Também é importante notar o tempo de execução do mapa, que em média, é de cinco segundos, que é o maior tempo de carregamento dos testes.

Figura 14 – Tempos de execução dos fluxos de autenticação



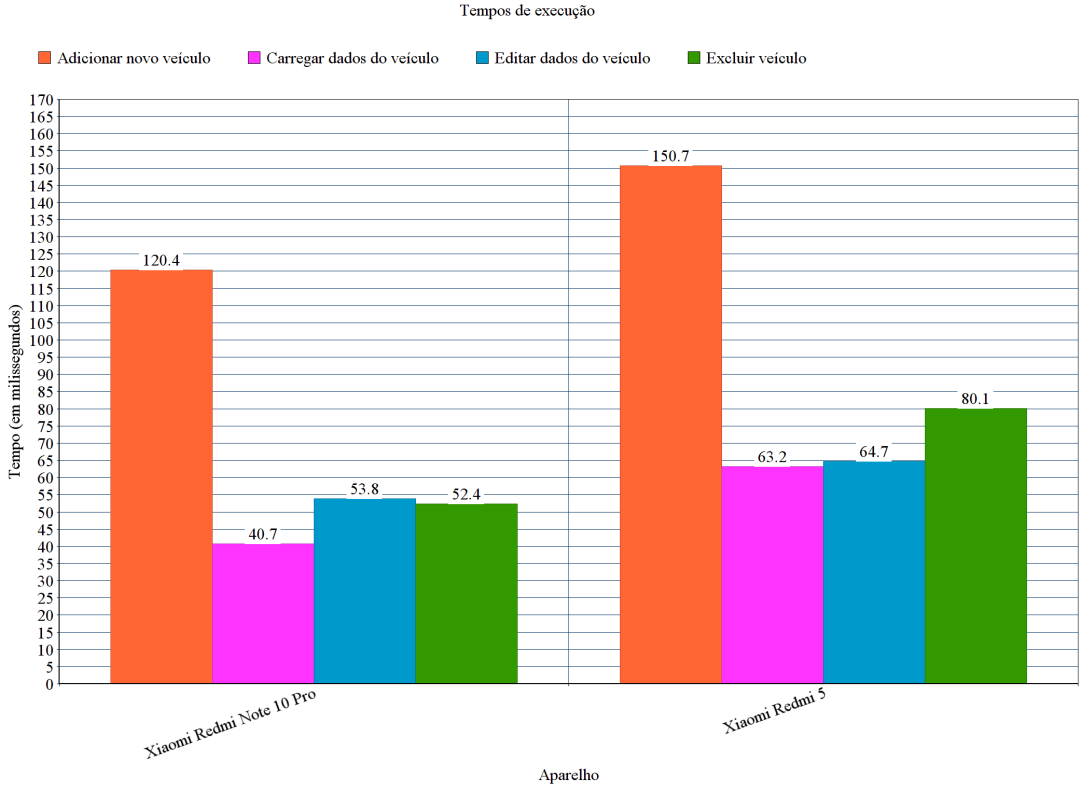
Fonte: Elaborado pelo autor.

Figura 15 – Tempos de execução dos fluxos relacionados ao perfil pessoal



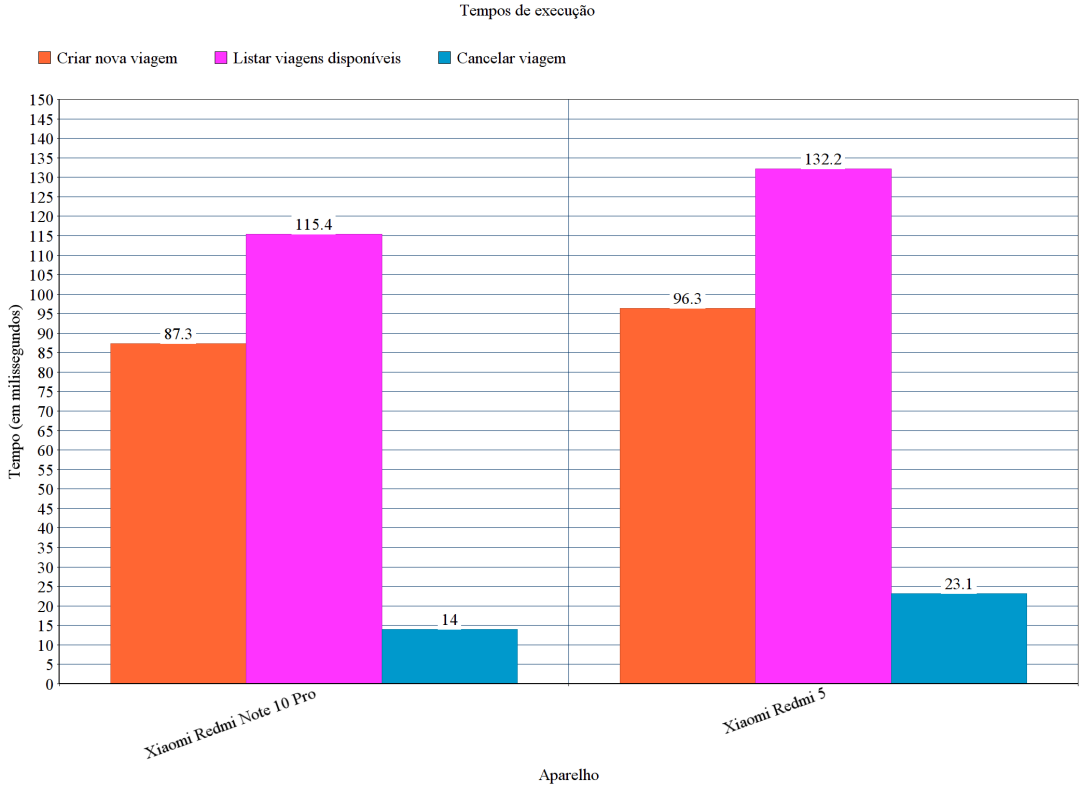
Fonte: Elaborado pelo autor.

Figura 16 – Tempos de execução relacionados ao *CRUD* de veículos



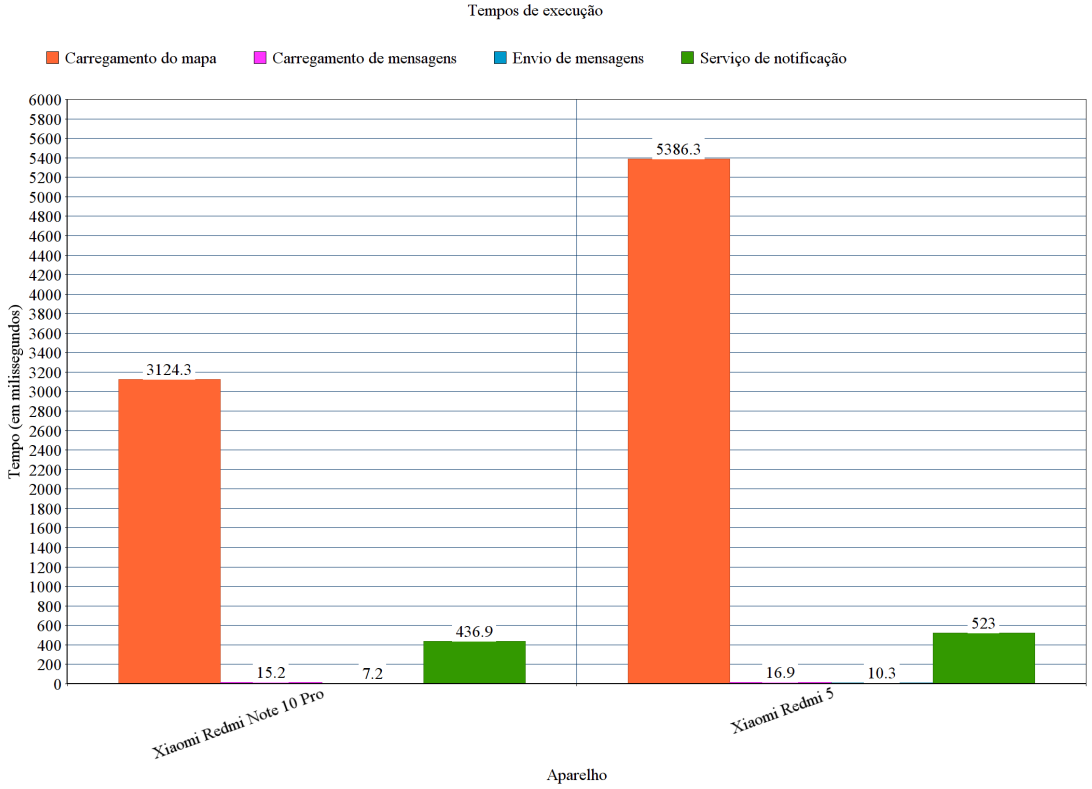
Fonte: Elaborado pelo autor.

Figura 17 – Tempos de execução relacionados ao *CRUD* de viagens



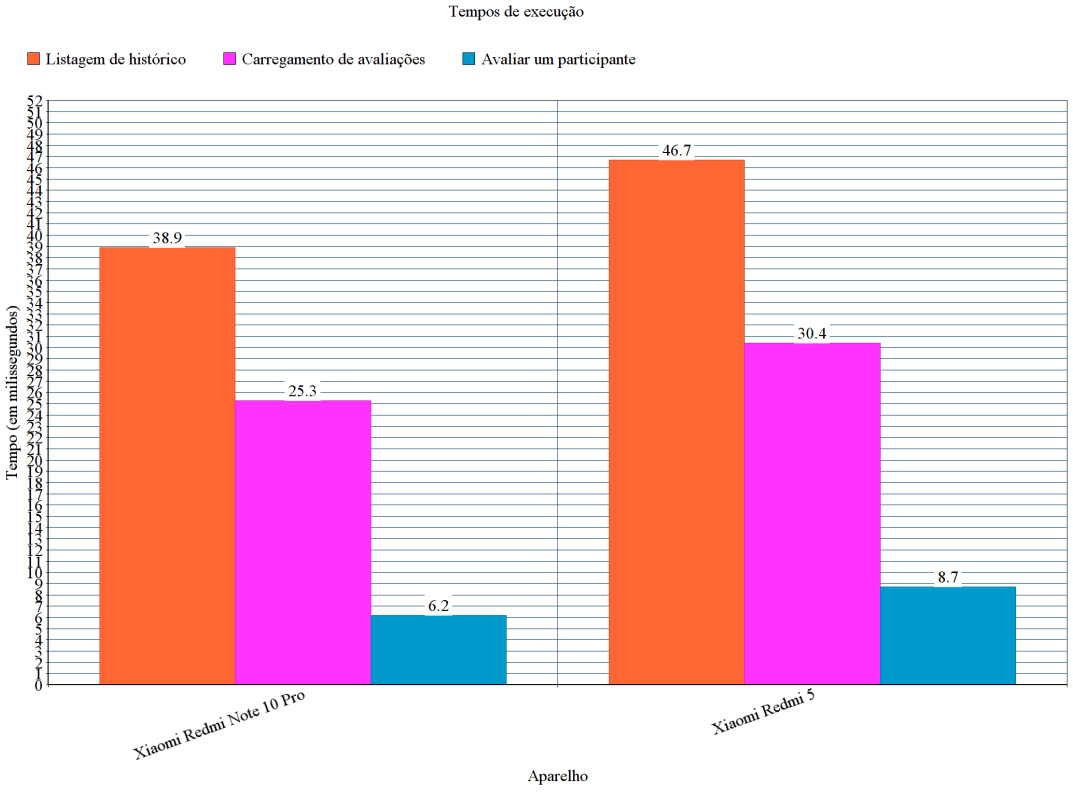
Fonte: Elaborado pelo autor.

Figura 18 – Tempos de execução das funções de viagem atual



Fonte: Elaborado pelo autor.

Figura 19 – Tempos de execução das funções de histórico e avaliação



Fonte: Elaborado pelo autor.

6 CONCLUSÕES E TRABALHOS FUTUROS

Neste trabalho, foi realizada a implementação e testes de tempo de execução de um aplicativo para gerência de viagens e caronas entre estudantes do ensino superior da cidade de Quixadá, voltado para a plataforma Android. A tecnologia principal usada foi a plataforma *Firebase*, que trouxe a implementação de várias das funcionalidades do sistema.

Os testes de tempo de execução mostraram que a plataforma *Firebase* consegue ser performática, trazendo um bom tempo de resposta para as requisições que a aplicação realiza para prover os serviços e objetivos descritos neste trabalho. A localização em tempo real, provida pelos serviços da *Google*, apesar de ter um tempo razoavelmente alto de carregamento na interface, também se mostrou performática durante os testes, já que a alta taxa de atualização da localização (500 ms) foi sempre obedecida, não havendo *bugs* ou *crashs* durante o uso do aplicativo.

Para trabalhos futuros, é pretendida a criação de testes e implementação de novas funcionalidades mais voltadas à experiência do usuário, como por exemplo, testes de usabilidade. Além disso, também é pretendido testes em campo mais amplos, envolvendo mais usuários e a interação entre eles, para observar como o sistema vai se comportar nesses cenários.

REFERÊNCIAS

BINU, P. K.; VISWARAJ, V. S. Android based application for efficient carpooling with user tracking facility. In: **2016 IEEE Conference on Computational Intelligence and Computing Research (ICCIC)**. Chennai, 2016. p. 1-4.

BRAZIL, J. Mobile Back-End as a Service (MBaaS), uma comparação entre CloudKit e Firebase. **MEDIUM**. [S.l], 2017. Disponível em: https://medium.com/mackmobile/mobile-back-end-as-a-service-mbaas-9329e4423f3c. Acesso em: 29 jan. 2022.

EPLER, P. L. **Green Computing** – History, Methodologies, Benefits and Barriers. [S.l]: George Mason, 2010.

FRANKENTHAL, R. Entenda o que é a escala Likert e como aplicá-la em sua pesquisa. **Mindminers**. [S.l], 2017. Disponível em: https://mindminers.com/blog/entenda-o-que-e-escala-likert/. Acesso em: 8 maio. 2020.

G1. Ônibus universitário quebra, e alunos descem para empurrar veículo no Ceará. **Globo**. [S.l], 2019. Disponível em: https://g1.globo.com/ce/ceara/noticia/2019/05/03/onibus-universitario-quebra-no-meio-do-transito-e-passageiros-saem-para-empurrar-o-veiculo-em-quixada-no-ceara.ghtml. Acesso em: 08 maio 2020.

GUEDES, G.; T.; A. **UML 2:** Uma abordagem prática. 3. ed. São Paulo: Novatec Editora LTDA, 2011.

GUPTA, M. R.; CHEN, Y. Theory and Use of the EM Algorithm. In: **2010 Foundations and Trends in Signal Processing (FTSP)**. [S.I.: s.n.], v. 4, n. 3, p. 224-228. 2010.

HUANG, S.; JIAU, M.; LIN, C. Optimization of the Carpool Service Problem via a Fuzzy-Controlled Genetic Algorithm. In: **IEEE Transactions on Fuzzy Systems**. [S.I.: s.n.], 2015. v. 23, n. 5, p. 1698-1712.

HYRECAR. The Environmental Benefits of Ridesharing. **Hyrecar**. [S.l], 2020. Disponível em: https://www.hyrecar.com/blog/environmental-benefits-ridesharing/. Acesso em: 26 maio. 2020.

KAMARUDDIN, K. A.; ROZLIS, N. R. M. UiTM Share Ride: Requirements Validation, Design and Development of a Campus Ride-Sharing Mobile Application. In: **2019 6th International Conference on Research and Innovation in Information Systems (ICRIIS)**. Johor Bahru, 2019. p. 1-6.

LEE, C.; RAHAFROOZ, M.; LEE, O. D. What Are The Concerns of Using a Ride-Sharing Service?: An Investigation of Uber. In: **Twenty-third Americas Conference on Information Systems**. Boston, 2017.

LIU, Z.; HUANG, K.; KIM, I.; ZHANG, Y.; ZHU, T. Analysis of the Influencing Factors of Carpooling Schemes. In: **IEEE Intelligent Transportation Systems Magazine**. [S.I.: s.n.], 2019. v. 11, n. 3, p. 200-208.

MICROSOFT. Data binding overview (WPF .NET). **Microsoft**. [S.l], 2021. Disponível em: https://docs.microsoft.com/en-us/dotnet/desktop/wpf/data/?view=netdesktop-6.0. Acesso em: 26 jan. 2022.

NALE, N. M.; LANDGE, S. R.; DAREKAR, S. A.; GADHAVE, S. B.; JORWEKAR, Y. S. Real-Time Carpooling Application for Android Platform. In: **2016 International Journal Of Engineering And Computer Science (IFECS)**. [S.I.: s.n.], 2016. p. 15900-15903.

NASCIMENTO, J. P. B. Uma introdução aos Padrões de Projeto. **IGTI**. [S.l], 2018. Disponível em: https://www.igti.com.br/blog/uma-introducao-aos-padroes-de-projeto/. Acesso em: 26 maio. 2020.

NASCIMENTO, M. S. L. A.; DOURADO, A. B. F.; ANDRADE, M. O. Avaliação do Transporte Escolar Rural pelos Usuários em Pequena Cidade do Agreste de Pernambuco. **Anais do XXX Congresso de Pesquisa e Ensino em Transportes, ANPET**. Rio de Janeiro, 2016.

OITO em cada dez pessoas aprovam app de carona. **Novo Momento**. [S.l], 2019. Disponível em: https://www.novomomento.com.br/Tecnologia/77556/oito-em-cada-dez-pessoas-aprovam-app-de-carona. Acesso em 19 abr. 2020.

PINTO, P. S. Universidades federais têm evasão de 15% em 2018. **Poder 360**. [S.l], 2019. Disponível em: https://www.poder360.com.br/governo/universidades-federais-tem-evasao-de-15-em-2018/. Acesso em: 19 abr. 2020.

PUŞCAŞIU, A.; FANCA, A.; VĂLEAN, H. Tracking and localization system using Android mobile phones. In: **2016 IEEE International Conference on Automation, Quality and Testing, Robotics (AQTR)**. Cluj-Napoca, 2016. p. 1-6.

ROCKETLOG. Data Binding in Android: A tutorial with examples. **RocketLog**. [S.l], 2021. Disponível em: https://blog.logrocket.com/data-binding-android-tutorial-with-examples/. Acesso em: 26 jan. 2022.

SANTO, S. Falta de transporte dificulta acesso à escola. **Ministério da Educação**. [S.l], 2007. Disponível em: http://portal.mec.gov.br/ultimas-noticias/201-266094987/2567-sp-787759183. Acesso em: 08 maio. 2020.

SHAHEEN, S.; COHEN, A.; BAYER, A. **The Benefits of Carpooling**. Berkeley: [s.n], 2018.

SHVETS, A. **Dive Into Design Patterns**. [S.I.: s.n.], 2018.

SILVA, M. M. DA; SANTOS, M. T. P. Os Paradigmas de Desenvolvimento de Aplicativos para Aparelhos Celulares. **T.I.S - Tecnologias, Infraestrutura e Software - UFSCar**, v. 3, n. 2, p. 162–170, 2014.

SINGH, S. Green computing strategies challenges. In: **2015 International Conference on Green Computing and Internet of Things (ICGCIoT)**. Noida, 2015. p. 758-760.

SWEN, S. Mobile App Authentication Architectures. **Gitbook**. [S.l], 2020. Disponível em: https://mobile-security.gitbook.io/mobile-security-testing-guide/general-mobile-app-testing-guide/0x04e-testing-authentication-and-session-management. Acesso em: 7 maio. 2020.

TAMPLIN, J. Firebase is Joining Google!. **Firebase, Inc**. [S.l], 2014. Disponível em: https://firebase.googleblog.com/2014/10/firebase-is-joining-google.html. Acesso em 5 maio. 2020.

UBER. COVID-19: Uber traz ao Brasil ferramenta que identifica o uso de máscara pelos motoristas parceiros. **Uber**. [S.l], 2020. Disponível em: https://www.uber.com/pt-BR/newsroom/covid-19-uber-traz-ao-brasil-ferramenta-que-identifica-o-uso-de-mascara-pelos-motoristas-parceiros/. Acesso em: 28 jan. 2022.

VIBLO. MVP Pattern with Data binding Android. **Viblo**. [S.l], 2017. Disponível em: https://viblo.asia/p/mvp-pattern-with-data-binding-android-RnB5p1edKPG. Acesso em: 26 jan. 2022.

VIKRAM, S. Green computing. In: **2015 International Conference on Green Computing and Internet of Things (ICGCIoT)**. Noida, 2015. p. 767-772.

WILLIAM, H. Advantages of Review and Rating System. **REVU**. [S.l], 2012. Disponível em: https://pt.slideshare.net/henrywilliams12/advantages-of-review-and-rating-system. Acesso em: 8 maio. 2020.

ZAINAB, P.; DESHMUKH, R.; NISCHAL, D.; FAHAD, F. Carpooling Application for Android Focusing on Authentication and Traffic Analysis. In: **2015 International Journal of Computer Applications (IJCA)**. [S.l.: s.n.], 2015. p. 12-16.

APÊNDICE A – DOCUMENTO DE ESPECIFICAÇÃO DE SOFTWARE

A.1 HISTÓRICO DE REVISÃO

Data	Versão	Descrição	Autor
25/MAI/2020	1.0	Introdução, visão geral e escopo do projeto	Carlos Ferreira
26/MAI/2020	1.1	Elicitação de requisitos funcionais e não funcionais e regras de negócio	Carlos Ferreira
27/MAI/2020	1.2	Definição das prioridades dos requisitos	Carlos Ferreira
29/MAI/2020	1.3	Diagrama de caso de uso	Carlos Ferreira
30/MAI/2020	1.4	Definição das prioridades dos casos de uso	Carlos Ferreira
02/JUN/2020	1.5	Descrição detalhada dos casos de uso	Carlos Ferreira
28/JUN/2020	1.6	Diagramas de atividade	Carlos Ferreira
01/JUL/2020	1.7	Diagramas de sequência	Carlos Ferreira
06/JUL/2020	1.8	Diagrama de classes	Carlos Ferreira
27/JAN/2022	1.9	Atualização da documentação	Carlos Ferreira
27/JAN/2022	2.0	Atualização do diagrama de classes	Carlos Ferreira

A.2 INTRODUÇÃO

Este documento especifica os requisitos, fornece os diagramas e os demais detalhes referentes a um sistema para gerência de viagens e caronas, fornecendo ao desenvolvedor as informações necessárias para o projeto e implementação, assim como para a realização dos testes e homologação do sistema.

A.3 VISÃO GERAL

O aplicativo Banco De Trás tem como propósito amenizar o problema da lotação de transportes no ensino superior de Quixadá – CE. O sistema é implementado para a plataforma Android e gerencia caronas entre os estudantes da cidade.

A.4 ESCOPO DO PROJETO

Nesta sessão está descrito o escopo do projeto, com o nome do produto, sua finalidade e suas limitações.

A.4.1 Nome do produto

Banco de Trás.

A.4.2 Finalidade do produto

Implantar um sistema de caronas entre os alunos do ensino superior de Quixadá, buscando amenizar o problema dos transportes públicos, fornecendo uma forma organizada de gerir as viagens.

A.4.3 Limites do produto

Acessível apenas para *smartphones* com o sistema Android e com uma conexão de internet.

A.5 DESCRIÇÃO DOS REQUISITOS E REGRAS DE NEGÓCIO

Os seguintes requisitos funcionais foram encontrados após o levantamento inicial do Sistema Banco de Trás:

A.5.1 Requisitos Funcionais

Identificador: RF01
Descrição: Salvar as informações de um usuário específico
Prioridade: Essencial
Requisitos associados: Nenhum

Identificador: RF02
Descrição: O usuário cadastra seu(s) veículo(s), fornecendo informações como imagens e descrição
Prioridade: Essencial
Requisitos associados: Nenhum

Identificador: RF03
Descrição: O usuário edita as informações de um veículo específico, como adicio-

nar/remover imagens e editar a descrição

Prioridade: Essencial

Requisitos associados: RF02

Identificador: RF04

Descrição: O usuário remove um veículo específico

Prioridade: Essencial

Requisitos associados: RF02

Identificador: RF05

Descrição: O usuário cria uma nova viagem, fornecendo informações como veículo, quantidade de vagas, local de partida e local de destino

Prioridade: Essencial

Requisitos associados: RF02

Identificador: RF06

Descrição: O usuário que criou uma viagem pode cancelá-la

Prioridade: Essencial

Requisitos associados: RF05

Identificador: RF07

Descrição: O usuário deve ser capaz de visualizar no mapa ou na forma de lista todas as viagens disponíveis

Prioridade: Essencial

Requisitos associados: Nenhum

Identificador: RF08

Descrição: O usuário deve ser capaz de solicitar a participação em uma viagem

Prioridade: Essencial

Requisitos associados: RF07

Identificador: RF09

Descrição: O usuário deve ser capaz de desistir de uma viagem que ele está participando
Prioridade: Essencial
Requisitos associados: RF08

Identificador: RF10
Descrição: O usuário deve ser capaz de ver os detalhes de uma carona que ele deseja participar, como perfil dos participantes e os detalhes do veículo
Prioridade: Essencial
Requisitos associados: RF08

Identificador: RF11
Descrição: O usuário pode editar seu perfil
Prioridade: Importante
Requisitos associados: Nenhum

Identificador: RF12
Descrição: O usuário deve ser capaz de reportar o perfil de outros usuários
Prioridade: Essencial
Requisitos associados: Nenhum

Identificador: RF13
Descrição: O usuário deve ser capaz de avaliar em uma escala de 1 a 5 os participantes da mesma viagem
Prioridade: Essencial
Requisitos associados: RF08, RF12

Identificador: RF14
Descrição: O sistema deve permitir a comunicação entre os participantes de uma viagem via chat
Prioridade: Essencial
Requisitos associados: RF08

Identificador: RF15

Descrição: O sistema deve ser capaz de localizar os participantes de uma viagem em tempo real

Prioridade: Essencial

Requisitos associados: RF08

Identificador: RF16

Descrição: O usuário pode acessar e visualizar suas viagens anteriores

Prioridade: Importante

Requisitos associados: RF08

Identificador: RF17

Descrição: O usuário deve ser capaz de realizar logout no sistema

Prioridade: Importante

Requisitos associados: RF01

A.5.2 Requisitos Não Funcionais

Identificador: RNF01

Descrição: O tempo de resposta deve ser inferior a 10 segundos

Prioridade: Desejável

Categoria: Velocidade

Identificador: RNF02

Descrição: O sistema deve ter acesso restrito a usuários cadastrados e autenticados

Prioridade: Essencial

Categoria: Segurança

Identificador: RNF03

Descrição: O banco de dados do sistema deve ser acessível apenas para os desenvolvedores

Prioridade: Essencial

Categoria: Segurança

Identificador: RNF04
Descrição: O sistema deve ser intuitivo para o usuário
Prioridade: Essencial
Categoria: Usabilidade

Identificador: RNF05
Descrição: O sistema deve ser capaz de informar ao usuário sobre o *status* da viagem
Prioridade: Essencial
Categoria: Usabilidade

Identificador: RNF06
Descrição: O sistema deve estar sempre disponível
Prioridade: Importante
Categoria: Disponibilidade

A.5.3 Requisitos Suplementares

Identificador: RS01
Categoria: Compatibilidade
Descrição: O sistema deve ser compatível com a versão Android 4.4 *KitKat* e superior

Identificador: RS02
Categoria: Software
Descrição: O sistema deve ser implementado na linguagem de programação JAVA

Identificador: RS03
Categoria: Software
Descrição: O banco de dados do sistema deve ser o Firebase

Identificador: RS04

Categoria: Interface

Descrição: As telas do sistema devem ser implementadas em XML

A.5.4 Regras de Negócio

Identificador: RN01

Prioridade: Essencial

Descrição: Estar autenticado como usuário para acessar o sistema

Identificador: RN02

Prioridade: Essencial

Descrição: Ter um veículo cadastrado para criar uma nova viagem

Identificador: RN03

Prioridade: Importante

Descrição: O usuário só deve ser capaz de participar de viagens que estão disponíveis, ou seja, que ainda possuem vaga

Identificador: RN04

Prioridade: Essencial

Descrição: Um usuário só pode reportar outro dada uma descrição do motivo

Identificador: RN05

Prioridade: Importante

Descrição: Um usuário só pode avaliar outro após a conclusão da viagem

A.6 MODELO DE CASOS DE USO

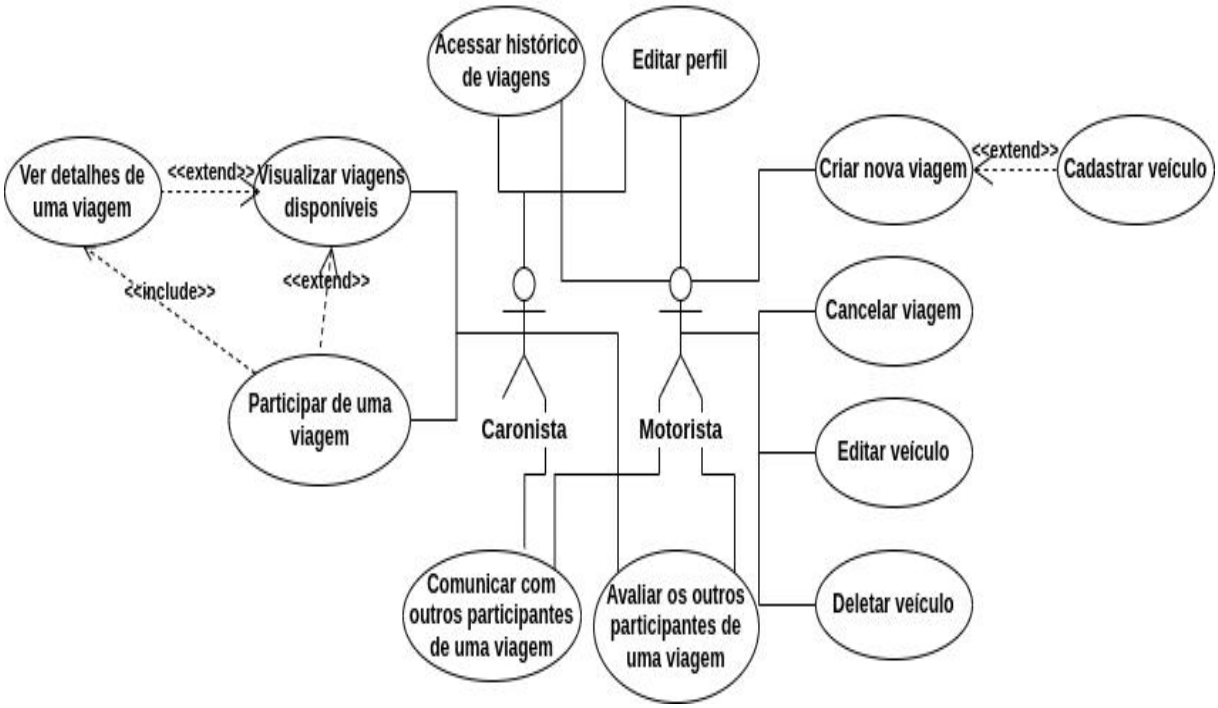
A.6.1 Identificação dos atores e suas responsabilidades

A seguir são apresentados os atores identificados para o Sistema Banco de Trás, bem como suas responsabilidades:

Usuário: Interage diretamente com o sistema, fornece os dados necessários para execução das funções do sistema.

A.6.2 Diagrama de Casos de Uso

As funcionalidades do sistema, a interação entre os atores e o sistema estão representados no Diagrama de Casos de Uso abaixo:



A.6.3 Definição de Prioridade de Desenvolvimento dos Casos de Uso

Casos de Uso			
Número	**Nome**	**Prioridade**	**Justificativa**

UC01	Acessar perfil de outros usuários	Média	Para que um usuário possa visualizar informações sobre outros, como rating
UC02	Desistir de participar de uma viagem	Alta	Para que um usuário possa desistir de uma viagem
UC03	Visualizar viagens disponíveis	Alta	Para que um usuário possa escolher a viagem que deseja
UC04	Criar nova viagem	Alta	Para que o usuário possa cadastrar uma nova viagem
UC05	Cadastrar veículo	Alta	Para que o usuário possa cadastrar seu veículo
UC06	Reportar perfil	Alta	Para que um usuário possa denunciar outro
UC07	Escrever motivo da denúncia	Média	Para que o usuário possa descrever o porquê da denuncia
UC08	Ver detalhes de uma viagem	Alta	Para que o usuário possa saber dos detalhes de uma viagem, como local de saída, destino e participantes
UC09	Participar de uma viagem	Alta	Para que o usuário expresse a vontade de participar de uma viagem
UC10	Editar veículo	Média	Para que o usuário possa atualizar as informações sobre seu veículo cadastrado
UC11	Deletar veículo	Média	Para que o usuário possa deletar um veículos da sua lista de veículos

UC12	Acessar histórico de viagens	Média	Para que o usuário possa visualizar suas viagens anteriores
UC13	Comunicar com outros participantes de uma viagem	Alta	Para que o usuário possa conversar com os outros participantes da viagem
UC14	Avaliar os outros participantes de uma viagem	Alta	Para que o usuário possa expressar de forma quantitativa o comportamento dos outros usuário de uma viagem
UC15	Enviar resenha sobre a viagem	Média	Para que o usuário possa descrever como foi a viagem
UC16	Editar perfil	Média	Para que o usuário possa atualizar as informações do seu próprio perfil
UC17	Cancelar viagem	Alta	Para que o usuário possa cancelar uma viagem

A.6.4 Descrição Detalhada dos Casos de Uso

Caso de Uso: Acessar perfil de outros usuários (UC01)

Descrição Resumida: O usuário visita o perfil de outro usuário

Ator Primário: Usuário

Pré-Condições: Estar logado e autenticado

Fluxo Principal:

1. O usuário clica na imagem ou nome de outro usuário;
2. O sistema abre uma nova tela com o perfil deste outro usuário

Caso de Uso: Desistir de participar de uma viagem (UC02)

Descrição Resumida: O usuário decide desistir de uma viagem antes dela acontecer

Ator Primário: Usuário

Pré-Condições: Estar logado e autenticado; Estar confirmado para uma viagem

Fluxo Principal:

1. O usuário clica no botão de desistir da viagem;
2. O sistema mostra um pop-up perguntando ao usuário se ele tem certeza que quer desistir;
3. O usuário clica em OK

Fluxo Alternativo:

1. O usuário clica em Cancelar

Caso de Uso: Visualizar viagens disponíveis (UC03)

Descrição Resumida: O sistema exibe uma lista de viagens disponíveis

Ator Primário: Usuário

Pré-Condições: Estar logado e autenticado

Fluxo Principal:

1. O usuário entra na página inicial do aplicativo, onde está o mapa disponível;
2. O usuário visualiza todas as viagens no mapa

Fluxo Alternativo:

1. O usuário clica em na aba "viagens";
2. Na aba estão todas as viagens disponíveis em formato de lista

Caso de Uso: Criar nova viagem (UC04)

Descrição Resumida: O usuário cria uma nova viagem com horários marcados, detalhes do veículos e locais de origem e destino

Ator Primário: Usuário

Pré-Condições: Estar logado e autenticado; Ter um veículo cadastrado

Fluxo Principal:

1. O usuário clica no botão de criar nova viagem;
2. O sistema vai para uma tela onde o usuário deve preencher os dados da nova viagem;

3. O usuário clica em finalizar

Fluxo Alternativo:

1. O usuário clica em cancelar

Fluxo de Exceção:

1. O sistema não permite a criação de uma nova viagem se o usuário já tem uma viagem pendente

Caso de Uso: Cadastrar veículo (UC05)

Descrição Resumida: O usuário cadastra um novo veículos informando imagens, número da placa, quantidade de vagas e uma descrição do veículo

Ator Primário: Usuário

Pré-Condições: Estar logado e autenticado

Fluxo Principal:

1. O usuário clica no botão de cadastrar veículo;
2. O usuário fornece os detalhes do veículo;
3. O usuário clica em concluir

Fluxo Alternativo:

1. O usuário clica em cancelar

Fluxo de Exceção:

1. O sistema não permite o cadastra de um novo veículo caso a placa fornecida já esteja no banco de dados

Caso de Uso: Reportar perfil (UC06)

Descrição Resumida: O usuário cria uma denuncia contra outro usuário

Ator Primário: Usuário

Pré-Condições: Estar logado e autenticado; Ter participado de alguma viagem com o outro usuário

Fluxo Principal:

1. O usuário entra no perfil do outro usuário;
2. O usuário escreve a denuncia;
3. O usuário clica em enviar

Fluxo Alternativo:

1. O usuário clica em cancelar

Caso de Uso: Escrever motivo da denuncia (UC07)

Descrição Resumida: O usuário descreve a denuncia em forma de texto

Ator Primário: Usuário

Pré-Condições: Estar logado e autenticado; Ter participado de alguma viagem com o usuário que está sendo denunciado

Fluxo Principal:

1. O usuário escreve a denuncia;
2. O usuário clica em enviar

Fluxo Alternativo:

1. O usuário clica em cancelar

Caso de Uso: Ver detalhes de uma viagem (UC08)

Descrição Resumida: O usuário visualiza os detalhes de uma viagem

Ator Primário: Usuário

Pré-Condições: Estar logado e autenticado

Fluxo Principal:

1. O usuário visualiza as viagens disponíveis;
2. O usuário clica em uma das viagens disponíveis;
3. O sistema mostra os detalhes desta viagem

Caso de Uso: Participar de uma viagem (UC09)

Descrição Resumida: O usuário decide participar de uma viagem

Ator Primário: Usuário

Pré-Condições: Estar logado e autenticado

Fluxo Principal:

1. O usuário clica para ver os detalhes de uma viagem;
2. O usuário clica em participar da viagem;
3. O sistema mostra um pop-up perguntado por confirmação;
4. O usuário clica em OK;
5. O criador da viagem aceita o novo participante

Fluxo Alternativo:

1. O usuário clica em cancelar;
2. O usuário é rejeitado pelo criador da viagem

Fluxo de Exceção:

1. O sistema não permite o usuário participar de uma nova viagem caso ele tenha uma viagem pendente

Caso de Uso: Editar veículo (UC10)

Descrição Resumida: O usuário edita um veículo da sua lista de veículos

Ator Primário: Usuário

Pré-Condições: Estar logado e autenticado; Ter algum veículo cadastrado

Fluxo Principal:

1. O usuário clica na sua lista de veículos;
2. O usuário clica no veículo que ele quer editar;
3. O usuário atualiza as informações do veículos, exceto placa;
4. O usuário clica em concluir

Fluxo Alternativo:

1. O usuário clica em cancelar

Caso de Uso: Deletar veículo (UC11)

Descrição Resumida: O usuário deleta um veículo da sua lista de veículos

Ator Primário: Usuário

Pré-Condições: Estar logado e autenticado; Ter algum veículo cadastrado

Fluxo Principal:

1. O usuário clica na sua lista de veículos;
2. O usuário clica no ícone de deletar no veículo que ele deseja apagar;
3. O sistema mostra um pop-up de confirmação;
4. O usuário clica em OK

Fluxo Alternativo:

1. O usuário clica em cancelar

Caso de Uso: Acessar histórico de viagens (UC12)

Descrição Resumida: O usuário acessa uma lista com todas as viagens que já participou

Ator Primário: Usuário

Pré-Condições: Estar logado e autenticado

Fluxo Principal:

1. O usuário clica na aba de histórico;
2. O sistema mostra uma lista com detalhes de todas as caronas que o usuário já participou

Caso de Uso: Comunicar com outros participantes de uma viagem (UC13)

Descrição Resumida: O usuário manda mensagens para os outros integrantes da viagem

Ator Primário: Usuário

Pré-Condições: Estar logado e autenticado; Estar confirmado para uma viagem

Fluxo Principal:

1. O sistema mostra na tela inicial todos os participantes da viagem com todos os detalhes da mesma, como localização;
2. O usuário clica na imagem ou nome de um dos participantes;
3. O usuário manda uma mensagem para este participante

Caso de Uso: Avaliar os outros participantes de uma viagem (UC14)

Descrição Resumida: O usuário avalia em uma escala de 1 a 5 os outros participantes da viagem

Ator Primário: Usuário

Pré-Condições: Estar logado e autenticado; Ter participado da viagem

Fluxo Principal:

1. Após a conclusão da viagem, o sistema mostra para o usuário uma tela onde o mesmo pode avaliar os outros usuários

Caso de Uso: Enviar resenha sobre a viagem (UC15)

Descrição Resumida: O usuário escreve em forma de texto o que ele achou da viagem

Ator Primário: Usuário

Pré-Condições: Estar logado e autenticado; Ter participado da viagem

Fluxo Principal:

1. Após a conclusão da viagem, o sistema mostra para o usuário uma tela onde o mesmo pode escrever um texto sobre a viagem

Caso de Uso: Editar perfil (UC16)

Descrição Resumida: O usuário edita seu próprio perfil

Ator Primário: Usuário

Pré-Condições: Estar logado e autenticado

Fluxo Principal:

1. O usuário entra na aba de perfil;
2. O usuário atualiza suas informações;
3. O usuário clica em finalizar

Fluxo Alternativo:

1. O usuário clica em cancelar

Caso de Uso: Cancelar viagem (UC17)

Descrição Resumida: O usuário cancela uma viagem que ele criou

Ator Primário: Usuário

Pré-Condições: Estar logado e autenticado; Ter criado alguma viagem

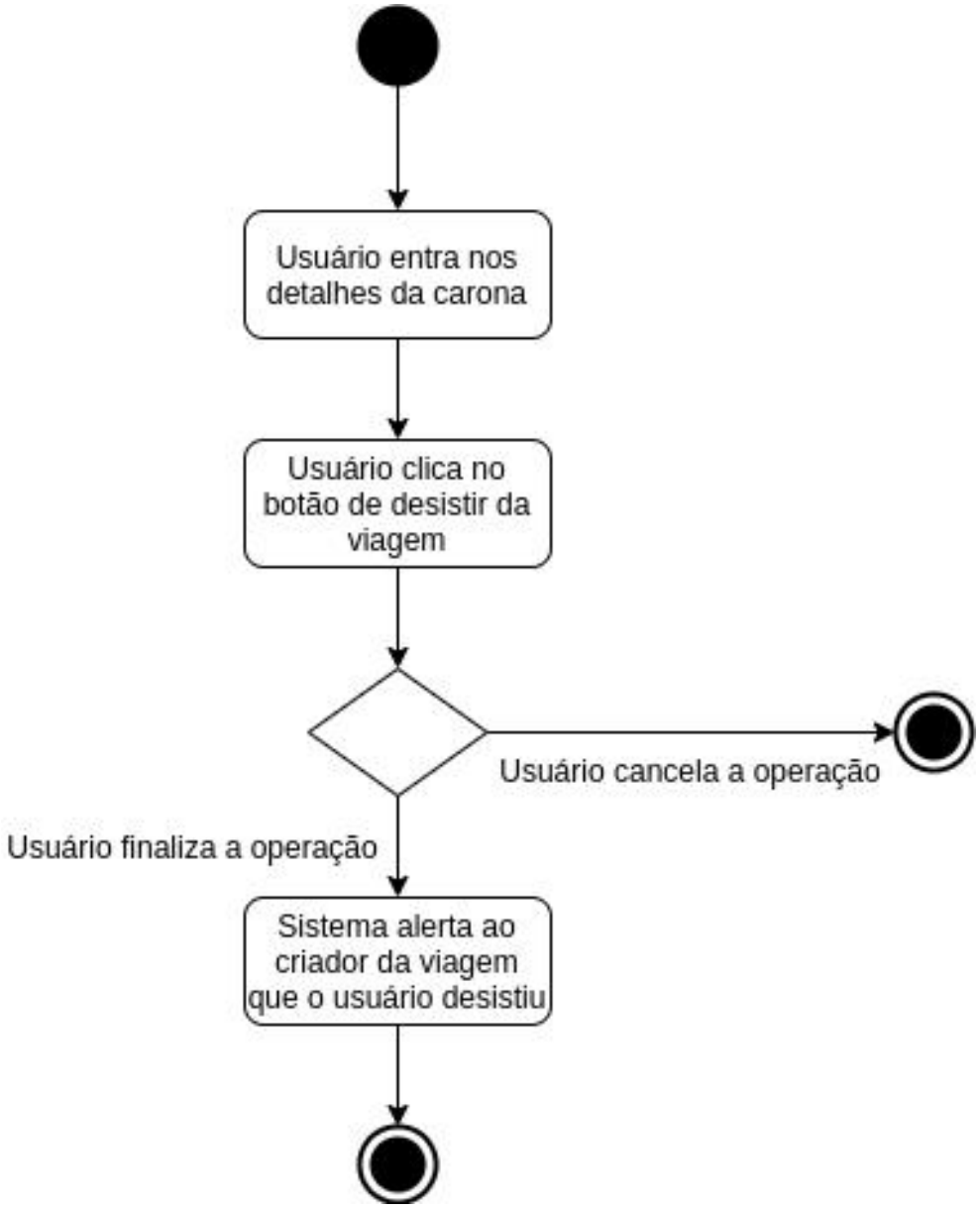
Fluxo Principal:

1. O usuário clica na opção de cancelar viagem;
2. O sistema mostra um pop-up de confirmação;
3. O usuário clica em OK;
4. O sistema avisa para os participantes que estavam confirmados que a viagem acaba de ser cancelada

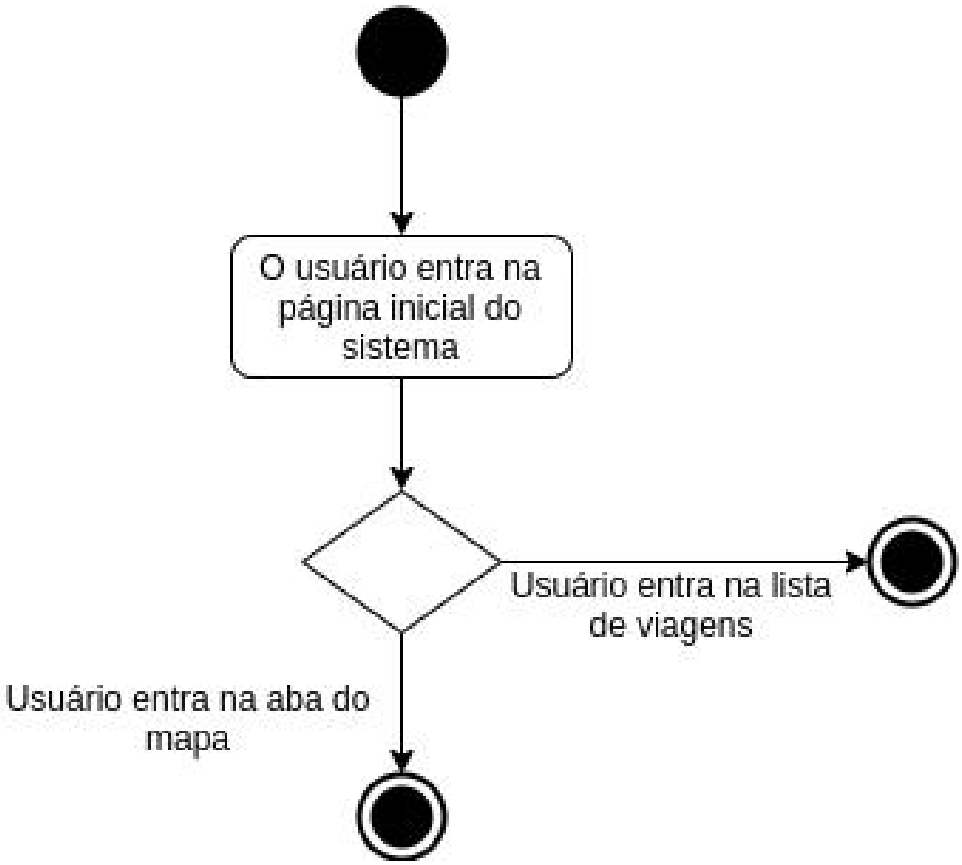
A.7 DIAGRAMAS DE ATIVIDADE

Diagramas de atividade dos casos de uso de alta prioridade.

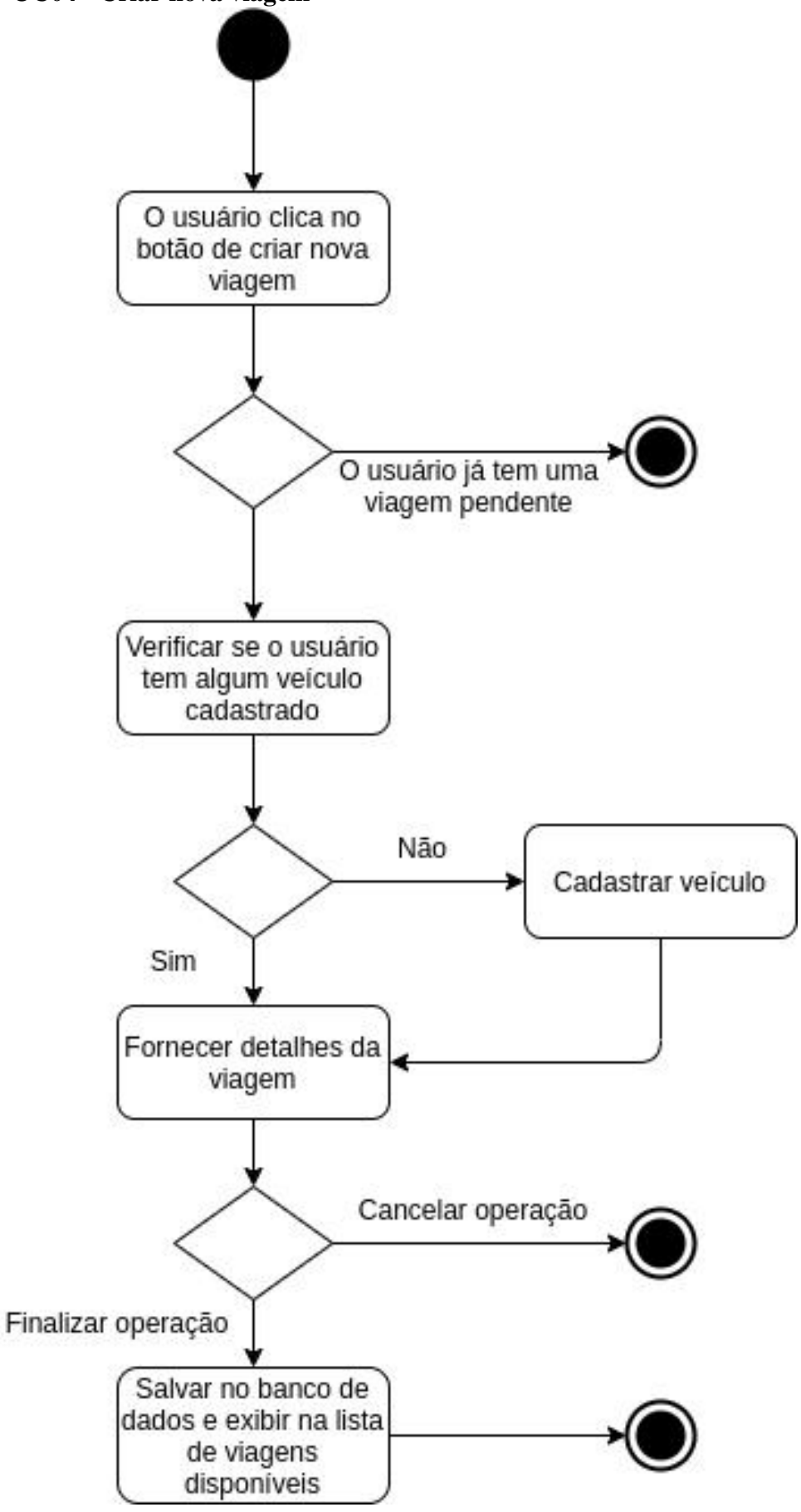
UC02 - Desistir de participar de uma viagem



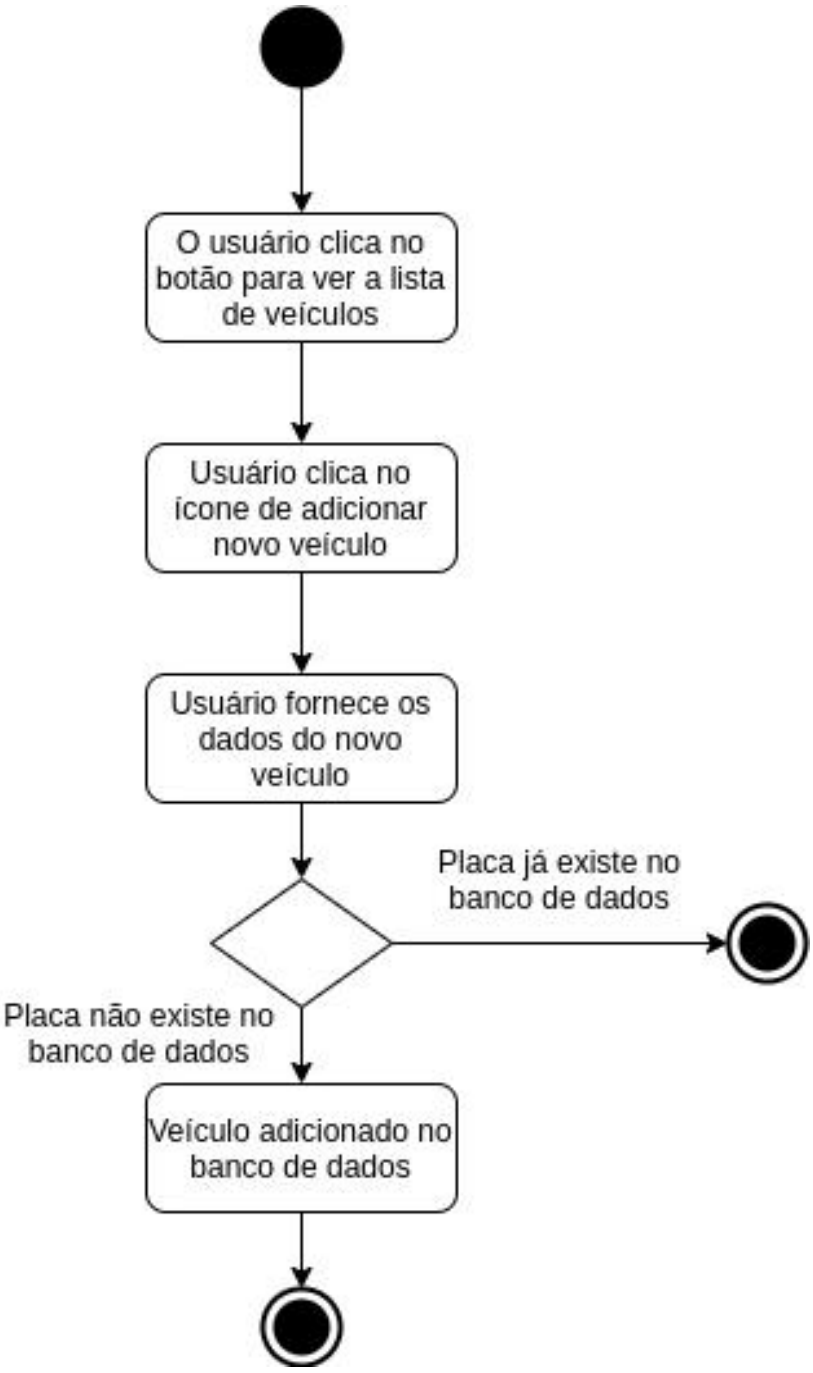
UC03 - Visualizar viagens disponíveis



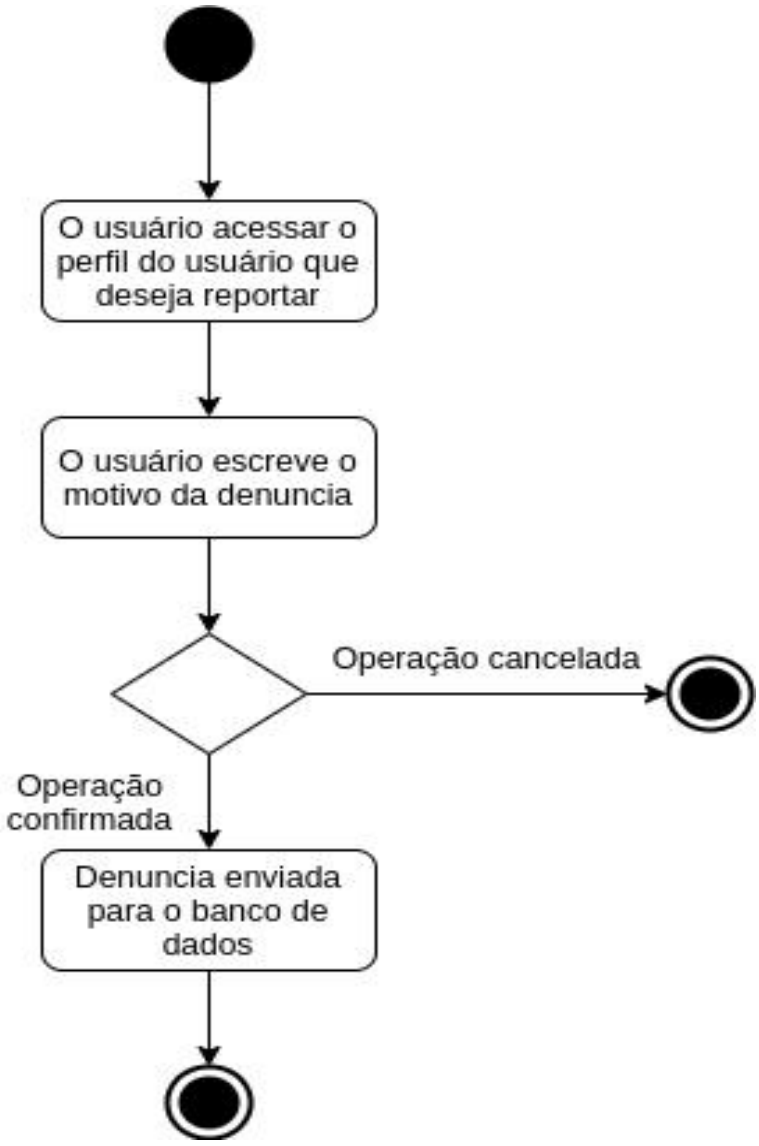
UC04 - Criar nova viagem



UC05 - Cadastrar veículo

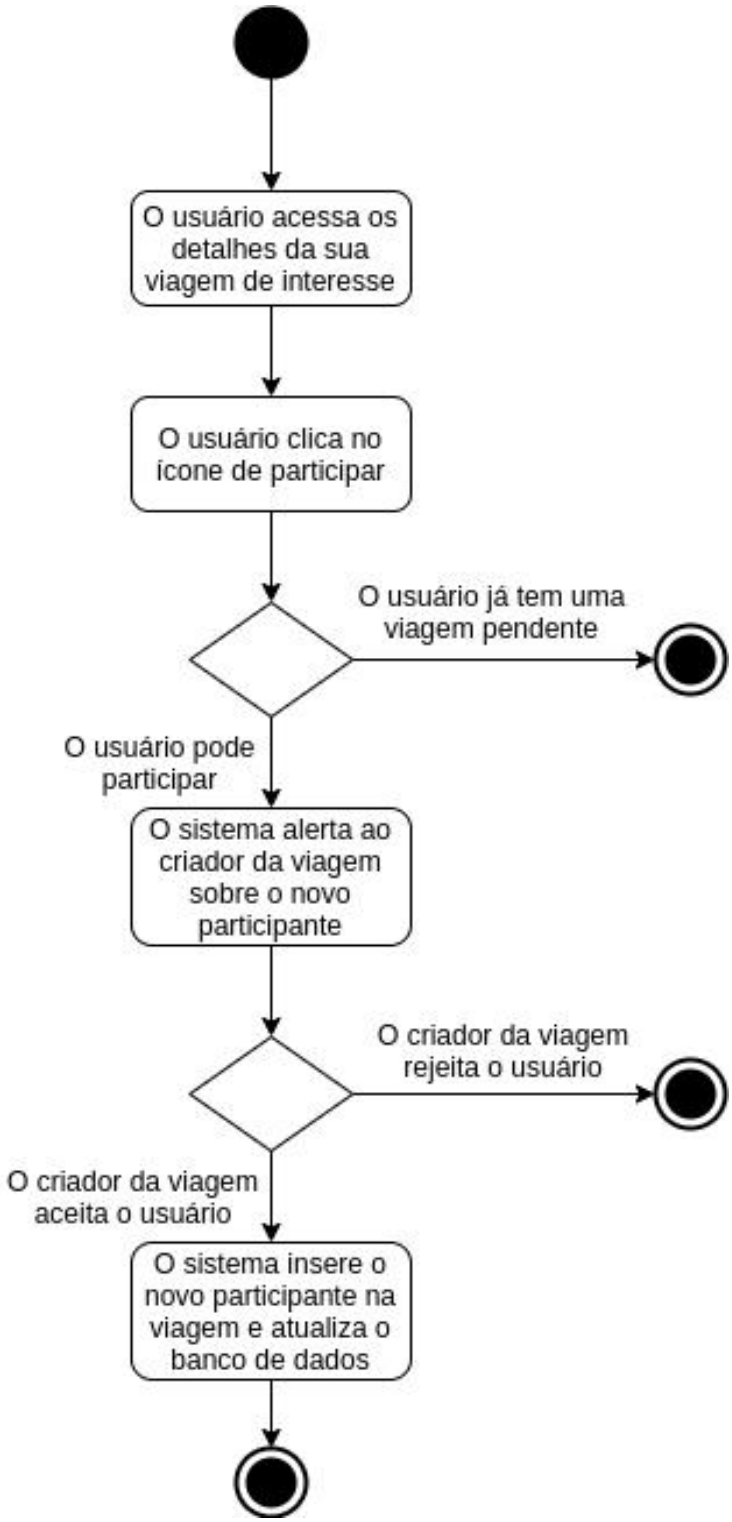


UC06 - Reportar perfil

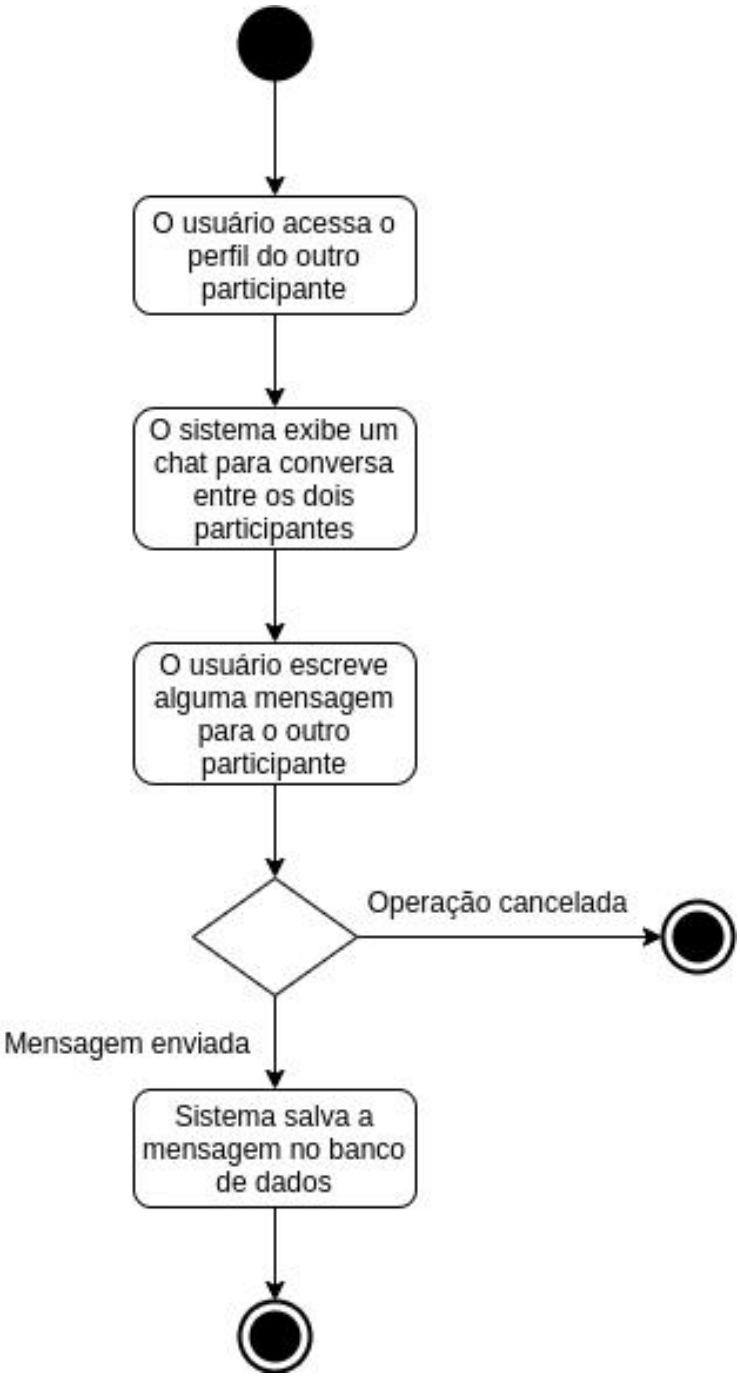


UC08 - Ver detalhes de uma viagem

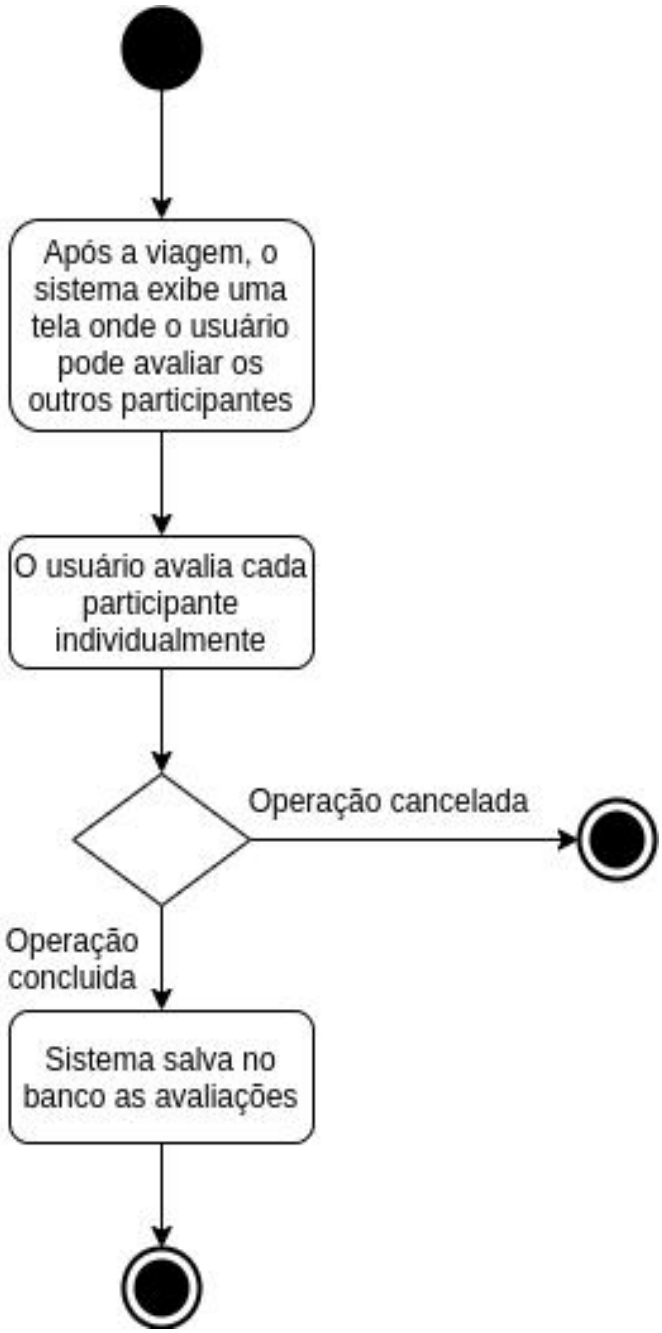
UC09 - Participar de uma viagem



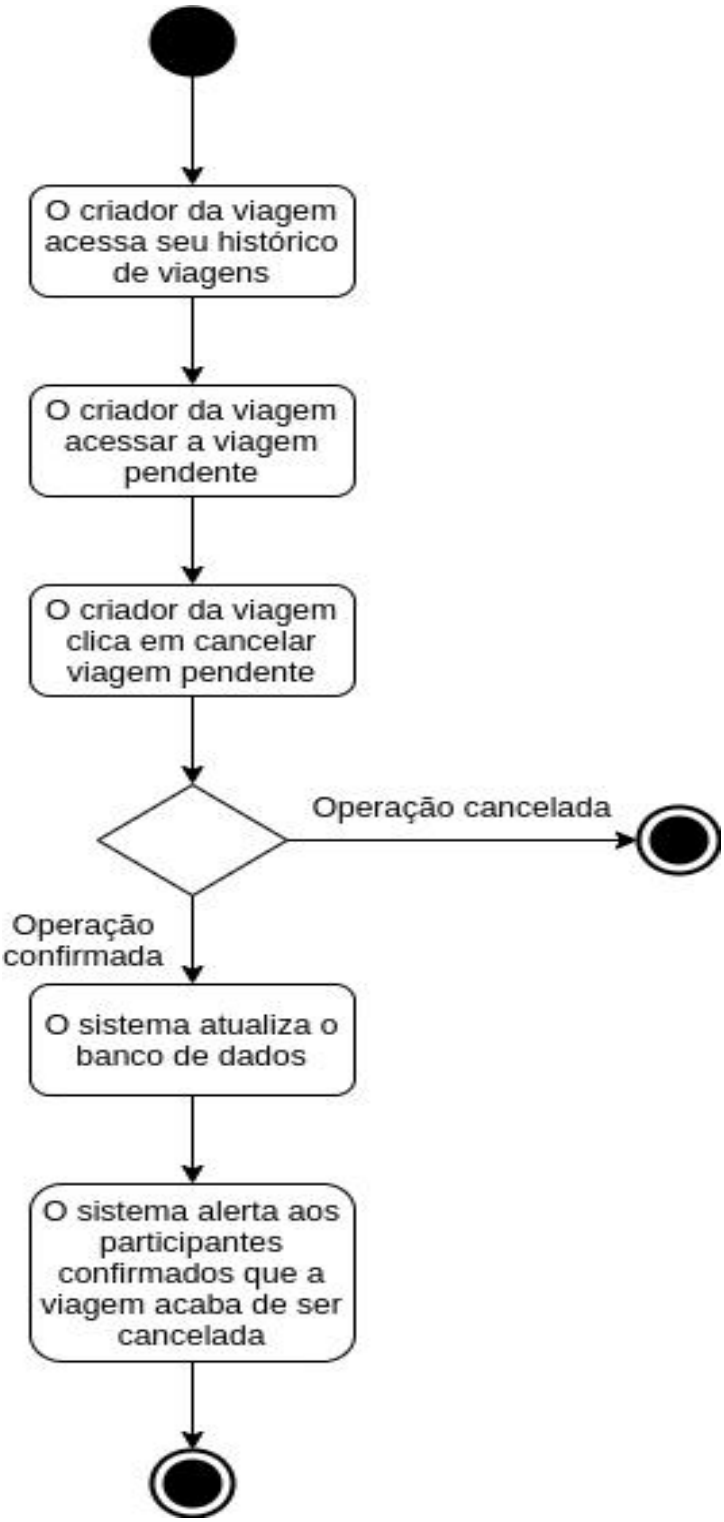
UC13 - Comunicar com outros participantes de uma viagem



UC14 - Avaliar os outros participantes de uma viagem



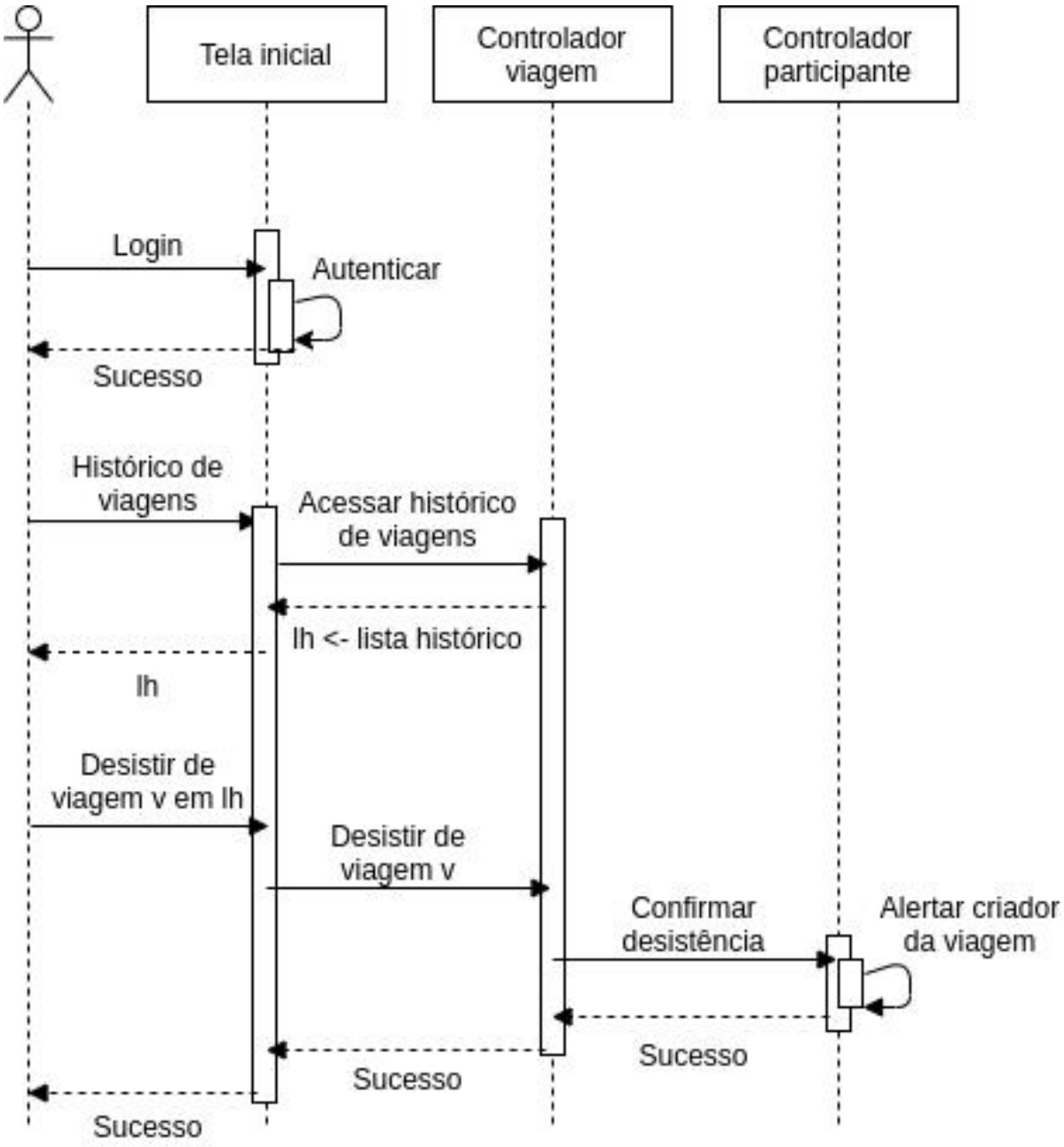
UC17 - Cancelar viagem



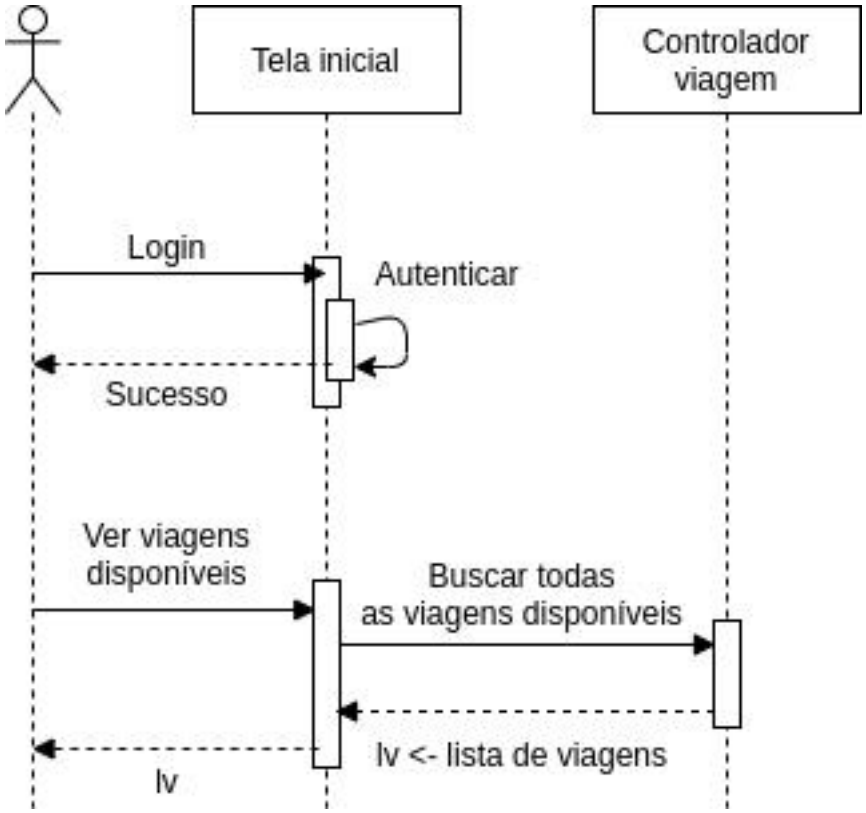
A.8 DIAGRAMAS DE SEQUÊNCIA

Diagramas de sequência dos casos de uso de alta prioridade.

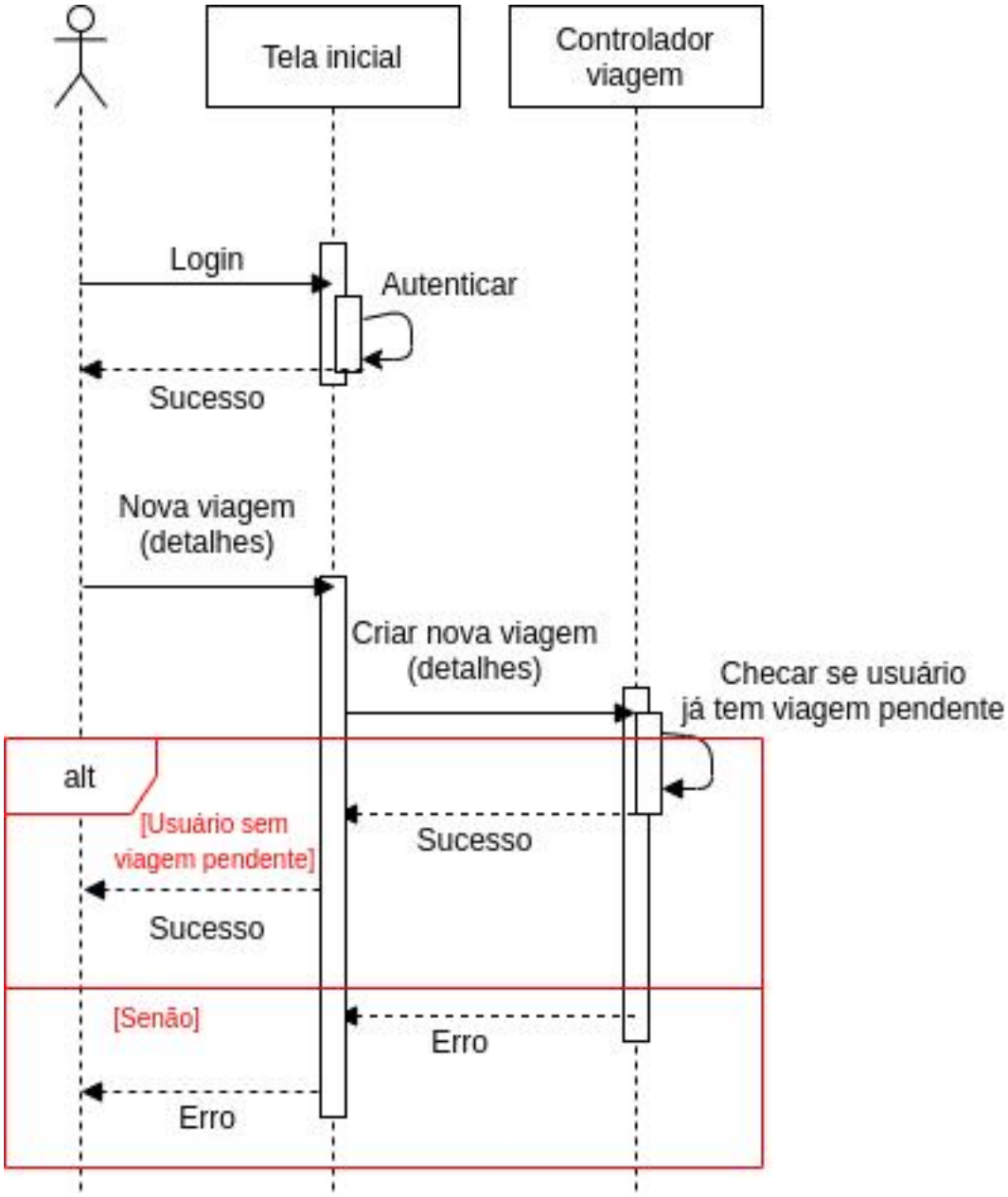
UC02 - Desistir de participar de uma viagem



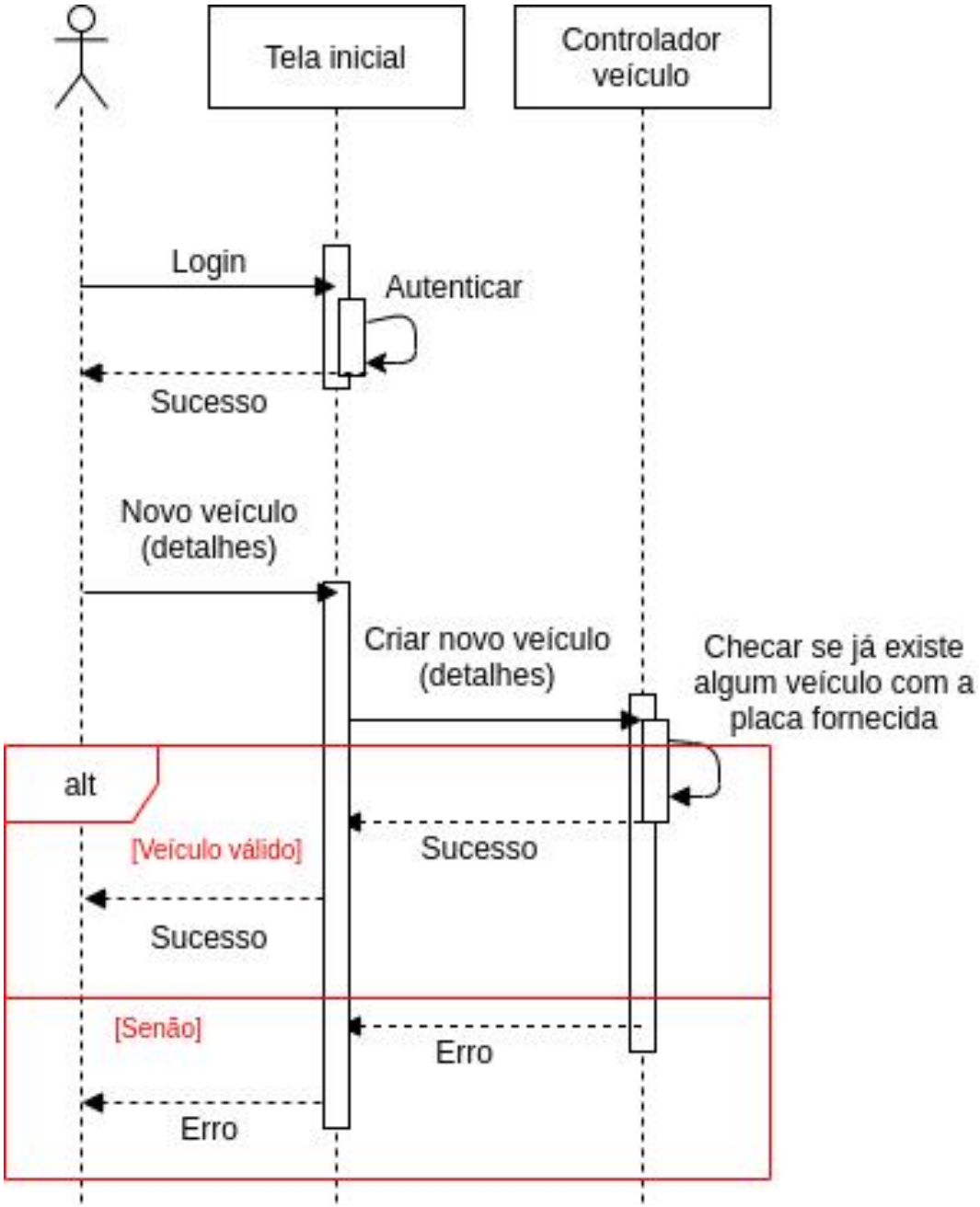
UC03 - Visualizar viagens disponíveis



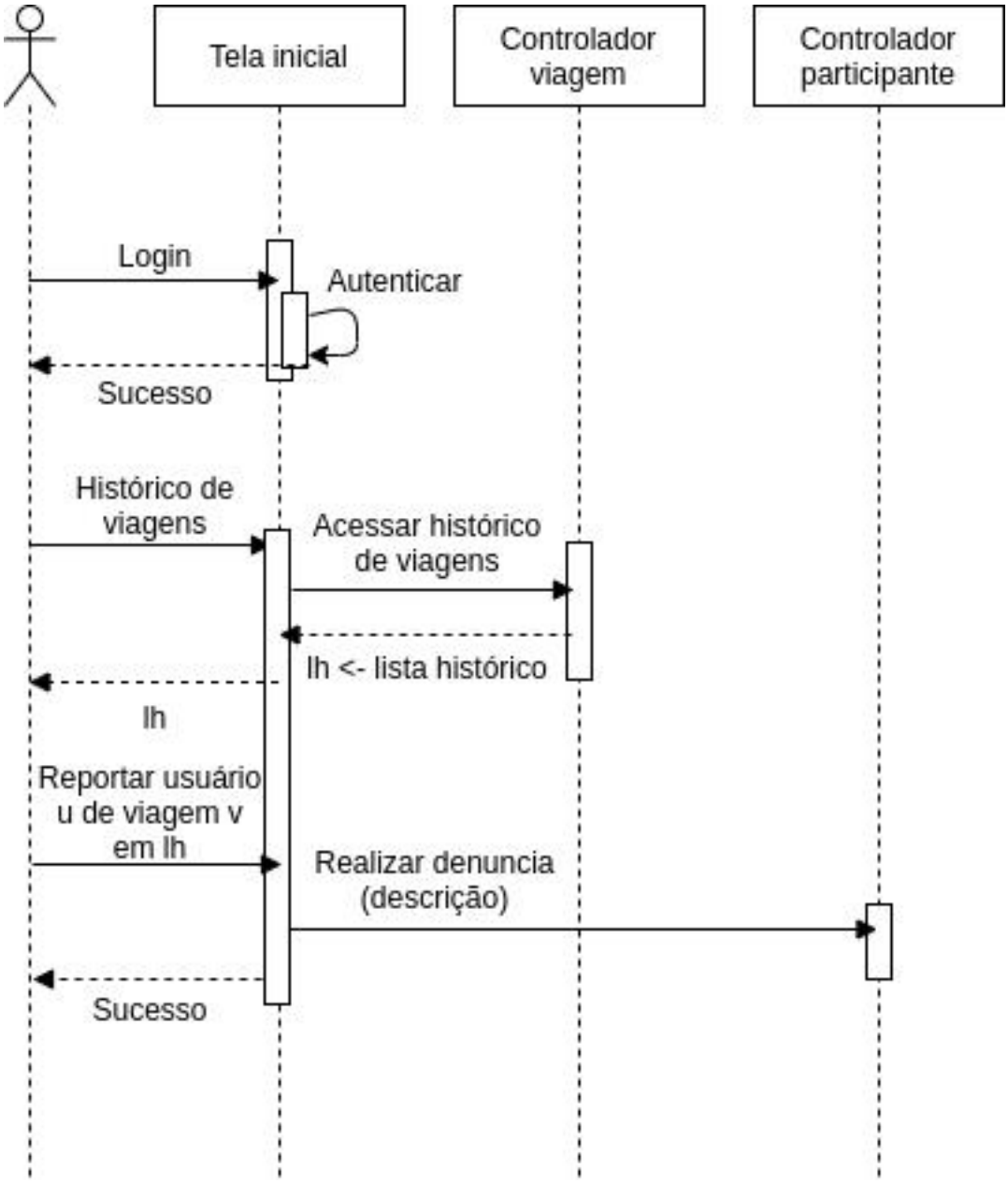
UC04 - Criar nova viagem



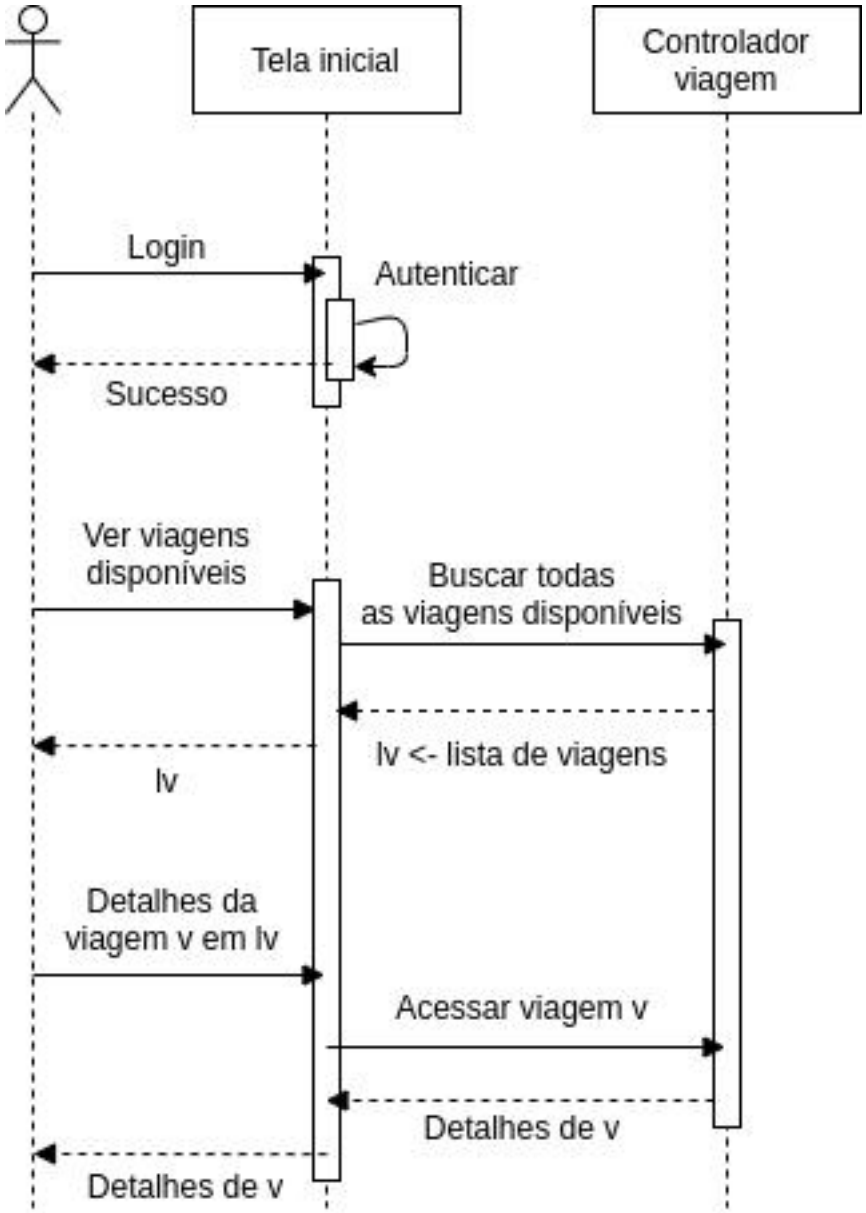
UC05 - Cadastrar veículo



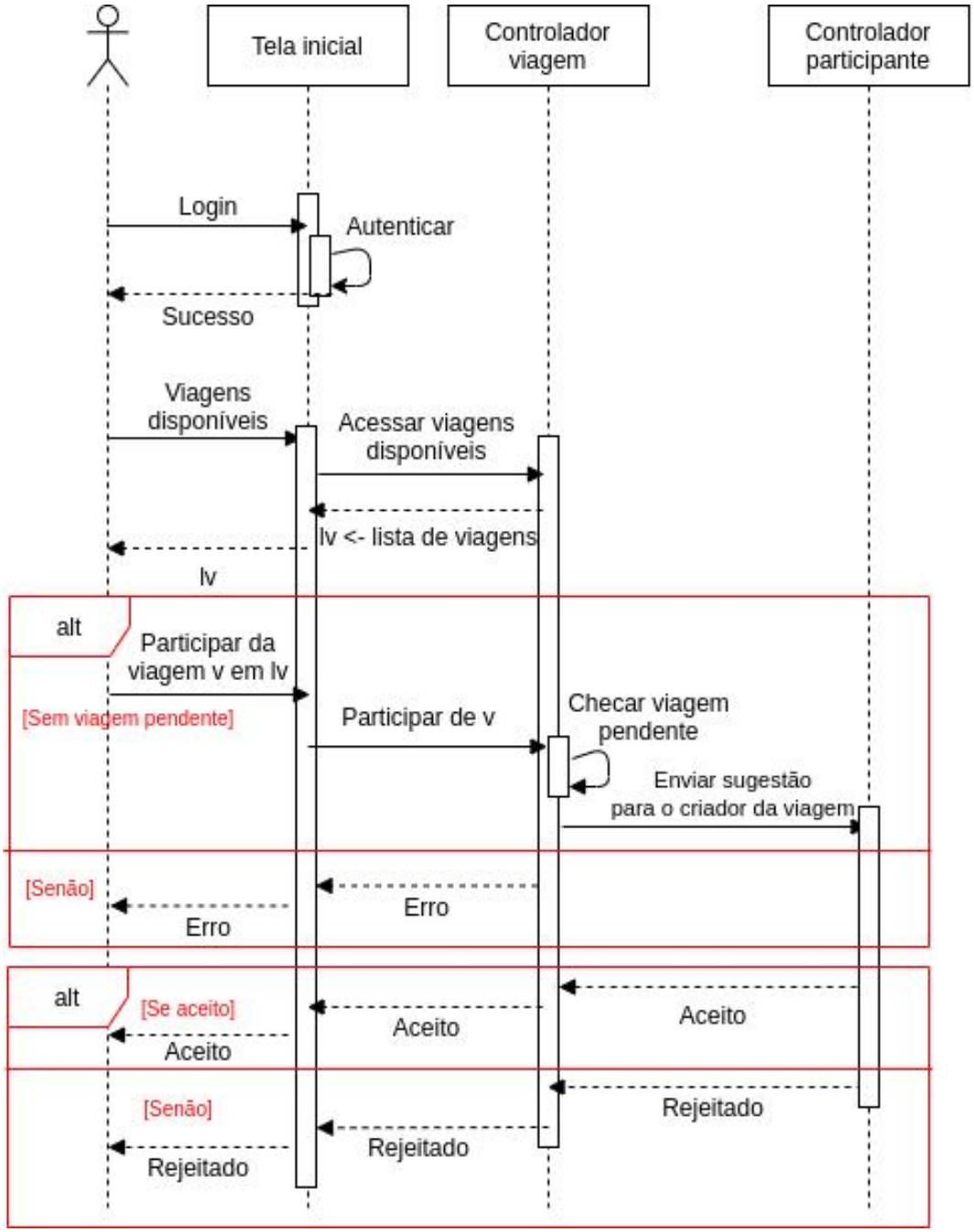
UC06 - Reportar perfil



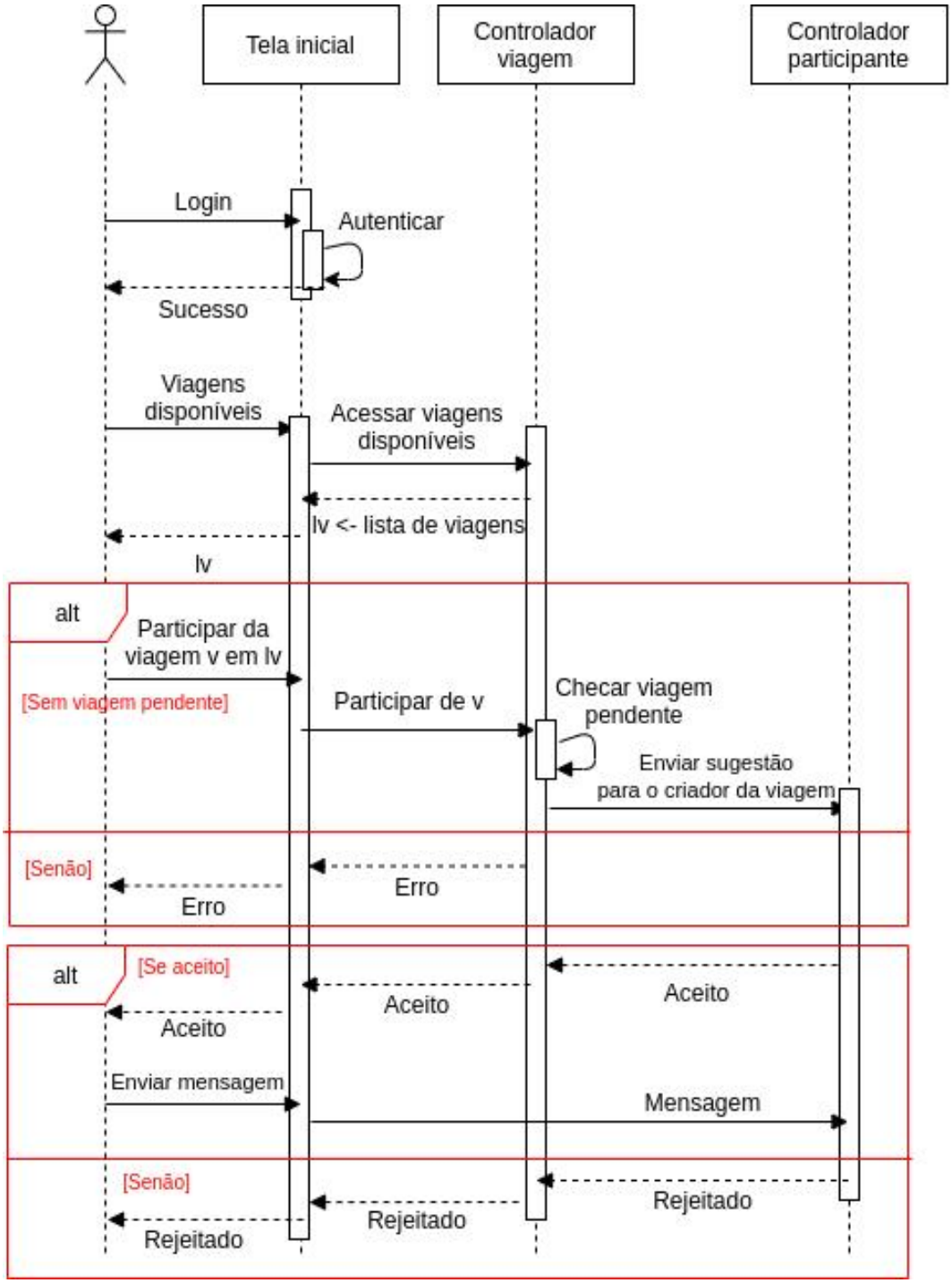
UC08 - Ver detalhes da viagem



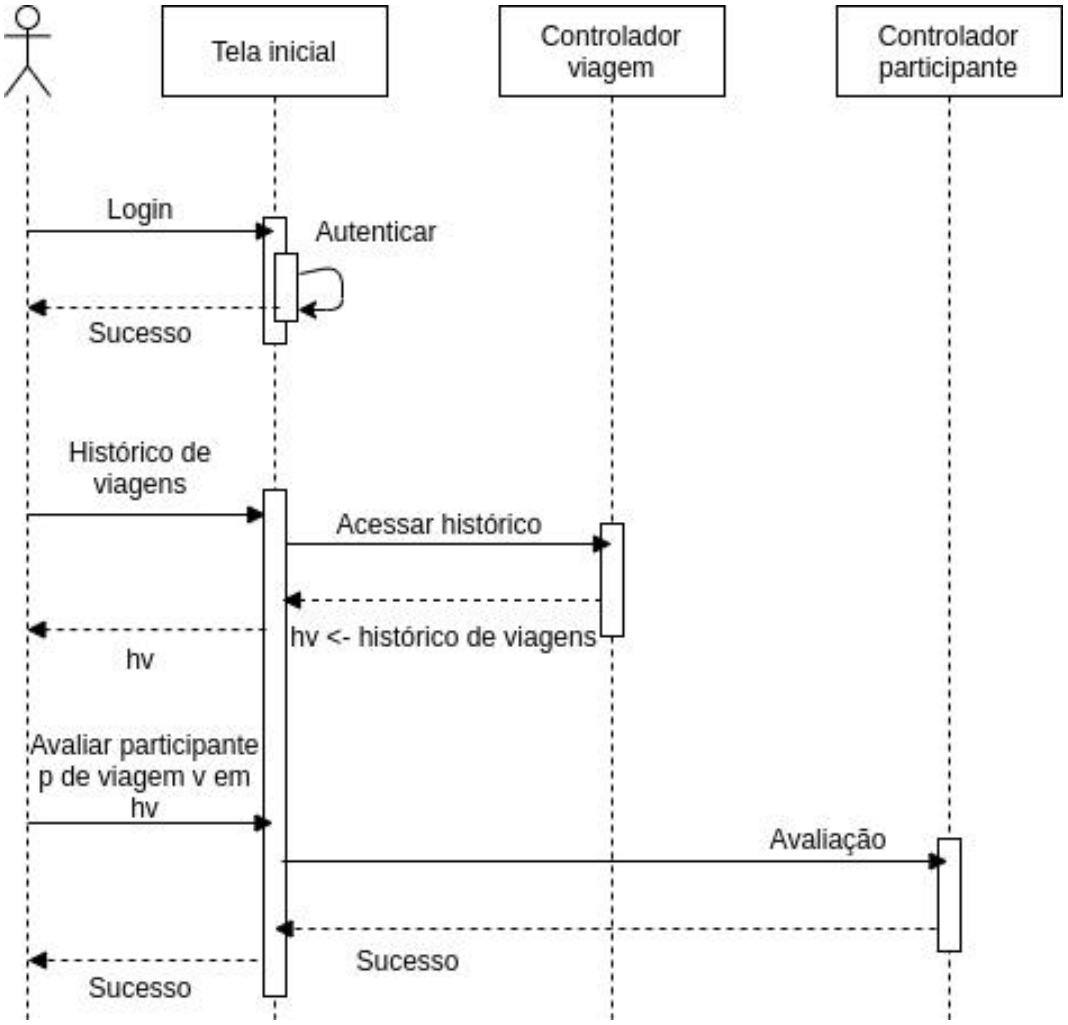
UC09 - Participar de uma viagem



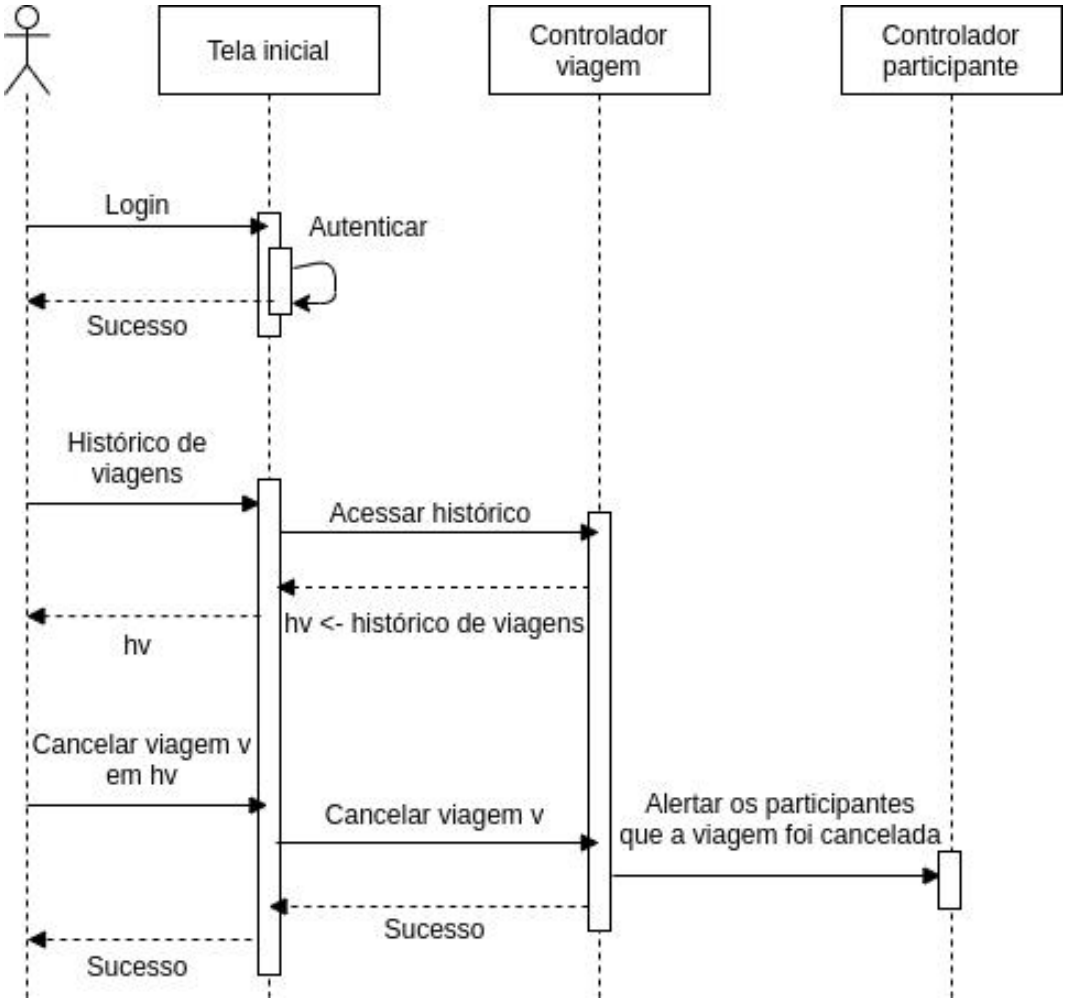
UC13 - Comunicar com outros participantes de uma viagem



UC14 - Avaliar os outros participantes de uma viagem



UC17 - Cancelar viagem



A.9 DIAGRAMA DE CLASSES