

UNIVERSIDADE FEDERAL DO PARANÁ

LUCAS DE MELLO CESAR

LUCAS SCHIP DO NASCIMENTO

MATEUS DE MEDEIROS GARCIA

DASHBOARD URBS:

UM SISTEMA DE MONITORAMENTO DO TRANSPORTE PÚBLICO DE CURITIBA

CURITIBA

2024

LUCAS DE MELLO CESAR

LUCAS SCHIP DO NASCIMENTO

MATEUS DE MEDEIROS GARCIA

DASHBOARD URBS:

UM SISTEMA DE MONITORAMENTO DO TRANSPORTE PÚBLICO DE CURITIBA

Trabalho de conclusão de curso apresentado ao curso de Tecnologia em Análise e desenvolvimento de Sistemas, Setor de Educação Profissional e tecnológica, Universidade Federal do Paraná, como requisito parcial à obtenção do título de Tecnólogo em Análise e Desenvolvimento de Sistemas.

Orientadora: Profa. Dra. Rafaela Mantovani Fontana

CURITIBA

2024



UNIVERSIDADE FEDERAL DO PARANÁ
SETOR DE EDUCAÇÃO PROFISSIONAL E TECNOLÓGICA
Rua Alcides Vieira Arcoverde 1225, - - Bairro Jardim das Américas, Curitiba/PR,
CEP 81520-260
Telefone: 3360-5000 - <http://www.ufpr.br/>

Ata de Reunião

TERMO DE APROVAÇÃO

LUCAS DE MELLO CESAR
LUCAS SCHIP DO NASCIMENTO
MATEUS DE MEDEIROS GARCIA

DASHBOARD URBS: UM SISTEMA DE MONITORAMENTO DO TRANSPORTE PÚBLICO DE CURITIBA

Monografia aprovada como requisito parcial à obtenção do título de Tecnólogo em Análise e Desenvolvimento de Sistemas, do Setor de Educação Profissional e Tecnológica da Universidade Federal do Paraná.

Prof.a Dra. Rafaela Mantovani Fontana
Orientadora – SEPT/UFPR

Prof. Dr. Jaime Wojciechowski
SEPT/UFPR

Prof. Dr. Paulo Eduardo Sobreira Moraes
SEPT/UFPR

Curitiba, 26 de junho de 2024.



Documento assinado eletronicamente por **RAFAELA MANTOVANI FONTANA, PROFESSOR ENS BASICO TECN TECNOLOGICO**, em 26/06/2024, às 20:03, conforme art. 1º, III, "b", da Lei 11.419/2006.



Documento assinado eletronicamente por **PAULO EDUARDO SOBREIRA MORAES, PROFESSOR DO MAGISTERIO SUPERIOR**, em 26/06/2024, às 22:25, conforme art. 1º, III, "b", da Lei 11.419/2006.



Documento assinado eletronicamente por **JAIME WOJCIECHOWSKI, PROFESSOR ENSINO BAS/TEC/TECNOL**, em 27/06/2024, às 04:45, conforme art. 1º, III, "b", da Lei 11.419/2006.



A autenticidade do documento pode ser conferida [aqui](#) informando o código verificador **6773411** e o código CRC **633D4040**.

AGRADECIMENTOS

Gostaria de agradecer a Professora Rafaela, pela paciência com o grupo. Também dedicamos esse trabalho às nossas famílias, pois sem o apoio delas não estaríamos aqui.

RESUMO

Este trabalho de conclusão de curso surgiu devido à uma demanda da Companhia de Urbanização de Curitiba (URBS), que entrou em contato com a UFPR solicitando uma forma de saber a quantidade de passageiros por ônibus, onde embarcam, se é uma estação tubo, terminal ou ponto de ônibus, e saber onde esse passageiro desembarca. Esses dados podem melhorar a tomada de decisão para a gestão do transporte urbano de Curitiba, a curto e longo prazo, assim oferecendo um serviço de melhor qualidade para o usuário. Com isso, o objetivo deste TCC é desenvolver um aplicativo que compartilhe a localização do usuário; e um *Dashboard* web, que apresenta a lotação dos veículos. O *dashboard* apresenta a localização de todos os veículos em operação da URBS, podendo visualizar todos, ou os veículos por linha. Ao selecionar uma linha é destacado no mapa o seu itinerário, os pontos de ônibus pertencentes a essa linha, e os veículos operando nela no momento. Também é possível filtrar os veículos por situação, tipo do veículo e se é adaptado. O sistema também possui indicadores e relatórios com dados relevantes para a administração do transporte público. O aplicativo possui a visualização de linha de ônibus da cidade, podendo favoritar para fácil acesso, visualização do itinerário da linha, pontos de ônibus e veículos em operação, além disso ele fornece a localização do usuário. Para as tecnologias utilizou-se o Angular para o frontend do dashboard, Java para o aplicativo, e Python para o backend. Nos casos em que foram necessárias análises mais complexas, foi utilizado o Elasticsearch. O banco de dados utilizado foi o MongoDB e um cache do Redis.

Palavras-chave: URBS. Transporte Público Curitiba. Geolocalização. Internet das Coisas. Mobilidade Urbana.

ABSTRACT

This project came about through a demand from URBS. They contacted UFPR asking for a way to find out the number of passengers per bus, find out where they board whether it is a tube station, terminal or bus stop, and know where that passenger disembarks. This data can improve decision-making in the short and long term, thus offering a better quality service to the user. With that, the objective of this TCC is to develop an application that shares the user's location with a Dashboard, thus showing the occupancy of the vehicles. The dashboard displays the location of all vehicles in operation, allowing the view of all, or the vehicles by line. When selecting a bus line, its route, the bus stops belonging to that line, and the vehicles currently operating on that line are highlighted on the map. It is also possible to filter vehicles by situation, vehicle type and if it is adapted. The system also has indicators and reports with data relevant to public transport administration. The app displays the city's bus lines, allowing you to favorite them for easy access, viewing the line's itinerary, bus stops and vehicles in operation, and also provides the user's location. For technologies, Angular was used for the frontend on the dashboard, Java for the application, and Python for the backend. In cases where more complex analyzes were required, elasticsearch was used. The database used was MongoDB and a Redis cache.

Keywords: URBS. Public Transport of Curitiba. Geolocation. Internet of Things. Urban Mobility.

LISTA DE FIGURAS

FIGURA 1 - CENTRO DE OPERAÇÕES (COR).....	19
FIGURA 2 - QUADRO DE VEÍCULOS DA URBS.....	21
FIGURA 3 - DETALHES DA AERONAVE PLANEFINDER.....	23
FIGURA 4 - VISÃO GERAL FLIGHTRADAR24.....	24
FIGURA 5 - MOOVIT.....	25
FIGURA 6 - CURITIBA156.....	26
FIGURA 7 - METROCARD.....	27
FIGURA 8 - MÉTODO SCRUM.....	30
FIGURA 9 - CRONOGRAMA DAS SPRINTS 1.....	39
FIGURA 10 - ROTEIRO SEGUNDA PARTE PROJETO.....	41
FIGURA 11 - ARQUITETURA DO SISTEMA.....	44
FIGURA 12 - LOGIN.....	46
FIGURA 13 - DASHBOARD.....	47
FIGURA 14 - FILTROS DASHBOARD.....	47
FIGURA 15 - LINHAS.....	48
FIGURA 16 - LINHA SELECIONADA.....	49
FIGURA 17 - INFORMAÇÕES DO VEÍCULO.....	49
FIGURA 18 - DETALHE DO PONTO.....	50
FIGURA 19 - FILTROS DE VEÍCULOS.....	50
FIGURA 20 - INDICADORES.....	51
FIGURA 21 - CADASTRO DE ADMINISTRADOR.....	52
FIGURA 22 - ADMINISTRADOR CADASTRADO.....	52
FIGURA 23 - LOGIN APPLICATIVO.....	53
FIGURA 24 - CADASTRO USUÁRIO.....	54
FIGURA 25 - LISTA DE LINHAS.....	55
FIGURA 26 - LINHAS FAVORITAS.....	55
FIGURA 27 - LINHAS SELECIONADA.....	56
FIGURA 28 - DETALHES DO VEÍCULO.....	57
FIGURA 29 - DETALHE DO PONTO.....	57
FIGURA 30 - TABELA DE HORÁRIO.....	58
FIGURA 31 - DIAGRAMA CASO DE USO DASHBOARD.....	63
FIGURA 32 - DIAGRAMA CASO DE USO APPLICATIVO.....	64
FIGURA 33 - DIAGRAMA DE CLASSE.....	100
FIGURA 34 - MODELO FÍSICO BANCO DE DADOS.....	101
FIGURA 35 - DIAGRAMA DE SEQUÊNCIA - UC01 REALIZAR LOGIN.....	102
FIGURA 36 - DIAGRAMA DE SEQUÊNCIA - UC02 EXIBIR VEÍCULOS EM OPERAÇÃO.....	102
FIGURA 37 - DIAGRAMA DE SEQUÊNCIA - UC03 EXIBIR ROTAS DAS LINHAS.....	103
FIGURA 38 - DIAGRAMA DE SEQUÊNCIA - UC04 EXIBIR PARADAS DA LINHA.....	103
FIGURA 39 - DIAGRAMA DE SEQUÊNCIA - UC05 EXIBIR DETALHES DO ÔNIBUS.....	104
FIGURA 40 - DIAGRAMA DE SEQUÊNCIA - UC06 FILTRAR LINHAS.....	104
FIGURA 41 - DIAGRAMA DE SEQUÊNCIA - UC07 FILTRAR ÔNIBUS.....	105
FIGURA 42 - DIAGRAMA DE SEQUÊNCIA - UC08 EXIBIR QUANTIDADE DE PESSOAS	

POR ÔNIBUS.....	105
FIGURA 43 - DIAGRAMA DE SEQUÊNCIA - UC09 GERAR RELATÓRIO DE MOVIMENTAÇÃO DAS LINHAS.....	106
FIGURA 44 - DIAGRAMA DE SEQUÊNCIA - UC10 GERAR INDICADORES DE LOTAÇÃO.....	106
FIGURA 45 - DIAGRAMA DE SEQUÊNCIA - UC10 GERAR INDICADORES DO VEÍCULO	107
FIGURA 46 - DIAGRAMA DE SEQUÊNCIA APP - UC01 REALIZAR LOGIN.....	107
FIGURA 47 - DIAGRAMA DE SEQUÊNCIA APP - UC02 REALIZAR CADASTRO NO SISTEMA.....	108
FIGURA 48 - DIAGRAMA DE SEQUÊNCIA APP - UC03 LISTAR LINHAS.....	108
FIGURA 49 - DIAGRAMA DE SEQUÊNCIA - UC04 FAVORITAR LINHAS.....	109
FIGURA 50 - DIAGRAMA DE SEQUÊNCIA APP - UC05 EXIBIR ITINERÁRIO DA LINHA..	109
FIGURA 51 - DIAGRAMA DE SEQUÊNCIA APP - UC06 EXIBIR VEÍCULOS EM OPERAÇÃO.	
110	

LISTA DE QUADROS

QUADRO 1 - DIVISÃO DE RESPONSABILIDADES.....	38
--	----

LISTA DE ABREVIATURAS OU SIGLAS

ANTT	- Agência Nacional dos Transportes Terrestres
API	- Application Programmer Interface
COMECA	- Coordenação da Região Metropolitana de Curitiba
CSS	- Cascading Style Sheets
GPS	- Global Positioning System
HTTP	- Hypertext Transfer Protocol
IDE	- Integrated Development Environment
IoT	- Internet of Things
JDK	- Java Development Kit
JEE	- Java Enterprise Edition
JRE	- Java Runtime Environment
JSP	- Java Server Pages
JVM	- Java Virtual Machine
JWT	- JSON Web Token
MVC	- Model View Controller
NOSQL	- Not Only Structured Query Language
PlanMob	- Plano de Mobilidade Urbana e Transporte Integrado de Curitiba
REST	- Representational State Transfer
RIT	- Rede integrada de Transporte Público de Curitiba
SQL	- Structured Query Language
UFPR	- Universidade Federal do Paraná
UML	- Unified Modeling Language
URBS	- Urbanização de Curitiba

SUMÁRIO

1. INTRODUÇÃO.....	13
1.1 PROBLEMA.....	13
1.2 OBJETIVOS.....	15
1.2.1 Objetivos específicos.....	15
1.3 JUSTIFICATIVA.....	15
1.4 ORGANIZAÇÃO DO DOCUMENTO.....	16
2. FUNDAMENTAÇÃO TEÓRICA.....	17
2.1 MOBILIDADE URBANA.....	17
2.2 MOBILIDADE URBANA E SMART CITIES.....	18
2.3 GESTÃO DA MOBILIDADE URBANA EM CURITIBA.....	20
2.4 ANÁLISE DOS SOFTWARES SEMELHANTES.....	22
2.4.1 Plane Finder.....	23
2.4.2 Flightradar24.....	24
2.4.3 Moovit.....	24
2.4.4 Curitiba 156.....	26
2.4.5 Metrocard.....	27
3. MATERIAIS E MÉTODOS.....	29
3.1 MODELO DE PROCESSO DE ENGENHARIA DE SOFTWARE.....	29
3.1.1 Metodologia ágil.....	29
3.1.2 Adaptação do Scrum as necessidades da equipe.....	30
3.1.3 Modelagem do software.....	31
3.1.3.1 Diagrama de Caso de Uso.....	31
3.1.3.2 Histórias de Usuario.....	31
3.1.3.3 Diagrama de Classes.....	32
3.1.3.4 Modelo Físico do Banco de Dados.....	32
3.1.3.5 Diagrama de Sequência.....	32
3.2 FERRAMENTAS DE DESENVOLVIMENTO.....	32
3.2.1 Angular.....	32
3.2.2 Python.....	33
3.2.3 Java.....	33
3.2.4 Redis.....	33
3.2.5 MongoDB.....	33
3.2.6 Elasticsearch.....	34
3.2.7 JSON Web Token - JWT.....	34
3.2.8 API Google Maps.....	34
3.2.9 Google Markers.....	34
3.2.10 FastAPI.....	35
3.2.11 Geopy.....	35
3.2.12 Pandas.....	35
3.2.13 Docker.....	35
3.2.14 Portainer.....	36
3.2.15 Cloudflare DNS.....	36
3.2.16 Outras Ferramentas.....	36
3.2.16.1 Discord.....	36
3.2.16.2 Microsoft Office.....	36
3.2.16.3 BrModelo.....	36

3.2.16.4 Astah.....	37
3.2.16.5 Lucidchart.....	37
3.2.16.6 Git.....	37
3.3 HARDWARE.....	37
3.4 DESENVOLVIMENTO DO PROJETO.....	38
3.4.1 Plano de Atividades.....	38
3.4.2 Cronograma das Sprints.....	38
4. APRESENTAÇÃO DO SISTEMA.....	43
4.1 ARQUITETURA DO SISTEMA.....	43
4.2 APRESENTAÇÃO DO SISTEMA.....	45
4.2.1 Login.....	45
4.2.2 Página inicial Dashboard.....	46
4.2.3 Linhas.....	48
4.2.4 Indicadores.....	51
4.2.5 Cadastro de administrador.....	51
4.3 APLICATIVO.....	53
4.3.1 Login.....	53
4.3.2 Cadastro.....	54
4.3.3 Página inicial.....	54
4.3.4 Seleção de linha.....	56
5. CONSIDERAÇÕES FINAIS.....	59
REFERÊNCIAS.....	60
APÊNDICE A - DIAGRAMA DE CASO DE USO.....	63
APÊNDICE B - HISTÓRIAS DE USUÁRIO.....	65
APÊNDICE C - DIAGRAMA DE CLASSE.....	100
APÊNDICE D - MODELO FÍSICO DO BANCO DE DADOS.....	101
APÊNDICE E - DIAGRAMAS DE SEQUÊNCIA.....	102

1. INTRODUÇÃO

Criada em 1974, a Rede integrada de Transporte Público de Curitiba (RIT) é considerada referência no transporte público no país. No ano de 2022, uma média diária de quase 1,1 milhões de passageiros (URBS, 2022), utilizaram a Rede Integrada de Transportes, em um total de 14 cidades, com uma frota de 1226 ônibus, em 250 diferentes linhas. Foi um total de 11.425 viagens diárias, utilizando os 21 terminais da cidade e 333 estações tubo (URBS, 2022).

Os terminais são localizados em pontos estratégicos, permitindo à população acessar os bairros e cidades vizinhas, e assim diminuindo o gasto da população com o transporte público.

Segundo Vasconcellos (2019), o transporte público de Curitiba foi inovador na década de 70, chamando atenção até de especialistas internacionais e sendo reproduzido por outros países. Porém, nos anos 90 começou a dar sinais de saturação, que ficaram mais visíveis nos anos 2000 (VASCONCELLOS, 2019).

Um dos principais problemas para essa saturação foi o crescimento desordenado de Curitiba e cidades da região metropolitana. Por isso são necessárias políticas públicas e planos que possam contribuir com o crescimento das cidades em conjunto com a gestão da mobilidade urbana (FARIA e VASCONCELOS, 2021).

Para auxiliar nisso foi criada, no ano de 2012, a Política Nacional de Mobilidade, que se aplica aos municípios com população acima de 20 mil habitantes. Esse plano apresenta diretrizes para o desenvolvimento urbano, incluindo os transportes, com o intuito de planejar o crescimento das cidades de forma ordenada e sustentável, priorizando o transporte público coletivo, assim como meios de transporte não motorizados (BRASIL, 2012).

Em um cenário local, em 2008, foi criado o Plano de Mobilidade Urbana e Transporte Integrado de Curitiba, de acordo com a Lei Municipal nº 11.266, de 16 de dezembro de 2004. Esse plano tem o objetivo de estabelecer políticas, diretrizes e planos de ação relativos à mobilidade urbana por meio de ações integradas de desenvolvimento urbano, de mobilidade e de proteção ao meio ambiente, promovendo a cidadania, inclusão social, aperfeiçoamento institucional, regulatório e da gestão (PREFEITURA MUNICIPAL DE CURITIBA, 2008).

1.1 PROBLEMA

O transporte público é essencial para diversas áreas econômicas nos grandes centros urbanos, e tem como papel promover acessibilidade, mobilidade e qualidade de vida para a população.

O transporte é um importante instrumento de direcionamento do desenvolvimento urbano das cidades. A mobilidade urbana bem planejada, com sistemas integrados e sustentáveis, garante o acesso dos cidadãos às

cidades e proporciona qualidade de vida e desenvolvimento econômico (Política Nacional de Mobilidade Urbana, 2013).

Quando um transporte público não possui qualidade a um preço acessível, a população tende a procurar meios de locomoção privados. Segundo Vasconcellos (2019): “a geração de congestionamento pelo uso excessivo do automóvel reduz a velocidade dos ônibus e, consequentemente, aumenta seu custo de operação e o valor da tarifa cobrada dos usuários”. Dados da Agência Nacional dos Transportes Terrestres (ANTT, 2016) revelam que, nos anos de 2014 e 2015, houve uma perda de 9% de usuários do transporte público nas maiores capitais do país, com isso gerando mais congestionamentos, entre outros problemas, como poluição do ar, sonora e acidentes de trânsito.

Segundo Silva e Silva (2018), os principais critérios de qualidade para o transporte público são: terminais e pontos, conforto, atendimento, segurança, preço/custo, confiabilidade e acessibilidade. Caso o valor da tarifa não seja condizente com os critérios desse serviço oferecido, a população inclina-se a procurar outros meios de transporte. Uma evidência disso é que pessoas que possuem automóvel estariam dispostas a deixar o carro em casa se a qualidade dos ônibus fosse melhor, segundo afirma Vasconcellos (2019).

Atualmente, a URBS utiliza o seguinte cálculo para determinar o número de ônibus necessários para uma determinada linha em um determinado período: número de passageiros por hora dividido pela capacidade do ônibus. O valor resultante é o número de viagens necessárias de uma determinada linha (URBS, 2022).

Como o número de passageiros por hora vem de uma média de períodos anteriores, ele pode conter uma margem considerável de erro, e seria mais eficiente poder determinar esse dado em tempo real, para caso uma linha estiver sobrecarregada, poder despachar mais veículos para essa linha.

Hoje a URBS possui a informação da frota em tempo real, consegue visualizar todos os veículos em operação com sua devida localização e possui câmeras nos terminais e principais estações tubo na cidade. Porém, eles não possuem a informação de quantos passageiros estão presentes em um ônibus, não sabem onde ele embarca e desembarca, e também não sabem para onde esse passageiro quer ir. Do lado do usuário, ele consegue visualizar o horário de uma linha, mas não sabe se o veículo está atrasado, ou o ônibus quebrou, assim como também não tem a informação sobre a lotação de um ônibus, podendo saber antecipadamente se vai conseguir viajar em um assento.

Com a dificuldade de manusear esses números, surgiu o tema desse trabalho de conclusão de curso, que propõe criar uma ferramenta de monitoramento da frota em uma mapa, em tempo real, que possibilite a visualização de todos os ônibus em circulação no momento, facilitando a visualização de veículos que estão atrasados, adiantados ou quebrados assim como a lotação de cada veículo de forma dinâmica. Esse sistema também possui indicadores e relatórios, para facilitar a comparação

dos dados obtidos em relação ao tempo, e com isso ajudar no planejamento de melhorias para o transporte público da cidade.

Para saber a quantidade de passageiros por veículo, esse trabalho também propõe a criação de um aplicativo para o usuário, que com a sua localização será determinado se ele está utilizando a rede pública de transporte no momento.

1.2 OBJETIVOS

O objetivo deste trabalho é desenvolver um software web que possibilite o monitoramento da frota da URBS em tempo real, e um aplicativo para os usuários que utilizam a Rede integrada de Transporte, que permita a verificação do itinerário das linhas e horários, além do monitoramento de ocupação dos veículos.

1.2.1 Objetivos específicos

Os objetivos específicos deste trabalho são:

- a) Apresentar ao usuário um *dashboard* dos ônibus em operação no momento, em que demonstre a localização um mapa;
- b) Permitir a visualização da lotação aproximada de cada veículo em tempo real;
- c) Permitir visualizar o itinerário de uma linha no mapa, com a localização dos veículos operando no momento;
- d) Permitir visualizar os pontos de ônibus, estações tubo e terminais no mapa;
- e) Permitir visualizar a situação atualizada dos ônibus, ou seja: atrasado, adiantado ou quebrado;
- f) Possibilitar a geração de relatórios e indicadores de gestão de mobilidade urbana;
- g) Disponibilizar um aplicativo para o usuário, em que ele possa visualizar os horários e itinerário de uma linha.

1.3 JUSTIFICATIVA

Conforme discutido brevemente, o transporte público de Curitiba apresenta sinais de saturação, e vêm perdendo passageiros para os aplicativos de viagem. Uma tendência das últimas décadas para lidar com esse problema é o conceito de *Smart City*, que são cidades que utilizam a tecnologia, muitas vezes em tempo real, para mitigar ou tratar problemas vislumbrando o crescimento sustentável, contando com a participação da população em sua construção e desenvolvimento (ANDRADE e GALVÃO, 2016).

Como exemplo de uma cidade inteligente temos a cidade do Rio de Janeiro, que no ano de 2013 ganhou o *Smart City World Award*, prêmio que reconhece iniciativas inovadoras no desenvolvimento urbano em áreas como mobilidade urbana, segurança e saúde. Um projeto elogiado foi o Centro de operações (COR), que por meio de câmeras ajuda na tomada de decisões rápidas, identificando por

exemplo, congestionamentos e acidentes de trânsito, e através de um parceria com o aplicativo *Waze* instrui a população sobre as melhores opções de tráfego (SMART CITY EXPO, 2013).

Segundo Kon e Santana (2017), outro exemplo de *Smart City* é a cidade de Madrid, na Espanha, que utiliza os aparelhos celulares dos passageiros de um ônibus para estimar a sua lotação, e essa informação como ser vista por outros usuários que estão esperando o ônibus, que tem a opção de esperar por um veículo mais vazio.

Diferente do primeiro exemplo, o segundo não necessita de grandes investimentos como câmeras de alta qualidade ou sensores modernos, ele utiliza apenas a localização dos usuários da rede pública de transporte que possuem o aplicativo instalado, que além de poder visualizar o horário e localização dos veículos, também conseguem verificar a sua lotação.

Um outro caso que utiliza dispositivos móveis para melhorar a qualidade de vida dos cidadãos é a cidade de São Paulo, que com o API Olho Vivo compartilha a localização de todos os ônibus da cidade em tempo real, e com isso permitiu o desenvolvimento de vários aplicativos oferecendo informações sobre o transporte público da cidade (KON e SANTANA, 2017).

O presente Trabalho de Conclusão de Curso propõe o desenvolvimento de um sistema dividido entre um aplicativo para o usuário e um *Dashboard* de monitoramento para a URBS. O aplicativo possibilita ao usuário visualizar as linhas do transporte público da cidade, seus horários e os veículos em operação por meio do API da URBS. A localização do usuário é enviada para o *Dashboard*, que compara as coordenadas do usuário com as coordenadas da frota operante, e com isso apresenta a lotação dos veículos. O *Dashboard* por meio da API do Google Maps, apresenta todos os ônibus, e pontos em um mapa em tempo real. O sistema também tem relatórios e indicadores, para ajudar na tomada de decisão.

1.4 ORGANIZAÇÃO DO DOCUMENTO

Este documento está organizado em cinco capítulos. O Capítulo 2, a seguir, contém a fundamentação teórica do projeto, com o objetivo de apresentar ao leitor os assuntos relevantes para a construção desse *software*. O Capítulo 3 tem o foco nos materiais e métodos planejados para o desenvolvimento desse projeto, apresentando as metodologias e processos no desenvolvimento do *software*, assim como as ferramentas que serão usadas. Já o quarto capítulo apresenta a arquitetura do sistema e suas funcionalidades. Por fim, o Capítulo 5 apresenta as considerações finais.

2. FUNDAMENTAÇÃO TEÓRICA

O objetivo deste capítulo é apresentar as pesquisas feitas a respeito da problemática tratada pelo *software* a ser desenvolvido. Apresentam-se, aqui, os conceitos de mobilidade urbana, *Smart Cities* e gestão da mobilidade urbana, além dos *softwares* semelhantes ao proposto.

2.1 MOBILIDADE URBANA

Em termos gerais, o significado de mobilidade é o deslocamento de um ponto para outro. Esse deslocamento pode ser por meios não motorizados como a pé ou bicicletas, ou meios motorizados. Os motorizados podem ser particulares como carros particulares, táxis ou aplicativos de viagem, ou públicos como o ônibus, que é o foco deste trabalho (MINISTÉRIO DAS CIDADES, 2013).

O transporte público é uma das principais medidas para dimensionar o desenvolvimento de uma cidade, portanto quando a mobilidade urbana é bem planejada com sistemas integrados, sustentáveis e que os cidadãos tenham acesso a uma tarifa justa, isso proporciona qualidade de vida e desenvolvimento econômico para o município (MINISTÉRIO DAS CIDADES, 2013).

Segundo a Lei 12.587/2012 (MINISTÉRIO DAS CIDADES, 2013), a Política Nacional de Mobilidade Urbana está fundamentada nos seguintes princípios:

- Acessibilidade universal;
- Desenvolvimento sustentável das cidades, nas dimensões socioeconômicas e ambientais;
- Equidade no acesso dos cidadãos ao transporte público;
- Eficiência, eficácia e efetividade na prestação dos serviços de transporte urbano;
- Gestão democrática e controle social do planejamento e avaliação da Política Nacional de Mobilidade Urbana;
- Segurança nos deslocamentos das pessoas;
- Justa distribuição dos benefícios e ônus decorrentes do uso dos diferentes modos e serviços;
- Equidade no uso do espaço público de circulação, vias e logradouros; e
- Eficiência, eficácia e efetividade na circulação urbana.

Apesar das políticas e avanços na área, na última década vem se destacando a utilização cada vez maior de meios particulares para o deslocamento urbano, principalmente o crescimento da utilização dos aplicativos de viagem, como o *Uber*. Apesar dos números de viagens e motoristas da *Uber* não serem divulgados, pode-se observar o impacto com a diminuição de táxis na cidade de Curitiba, que em 2017 possuía uma frota de 2954 veículos e em 2021 caiu para 2503 veículos, uma diminuição de mais de 15% (URBS, 2022).

Outro fator importante da região curitibana é a quantidade de ciclovias. Curitiba é uma das capitais com a maior quantidade de ciclovias, possuindo um total de 208,5 km de ciclovias (CURITIBA, 2022), algumas localizadas em ruas movimentadas, ligando distantes bairros ao centro da cidade.

Por fim, o último ponto a ser citado com relação à mobilidade urbana é a frota particular de veículos. Curitiba, no ano 2000, possuía 508.995 veículos particulares. Esse número subiu para 1.536.860 no ano de 2022 (DETRAN PR, 2022). Segundo Silva e Silva (2018), o aumento desse número gera uma série de efeitos colaterais tais como: poluição do ar, poluição sonora, congestionamentos, saturação viária e acidentes.

Apesar desse aumento no número de veículos nas ruas da cidade, Curitiba apresentou uma diminuição no número de acidentes e óbitos nos últimos anos (DETRAN PR 2022). Esse fator pode ser atribuído à políticas públicas, como diminuição da velocidade das vias. No entanto, deve-se considerar que a pandemia também pode ter tido algum impacto na redução da circulação de carros nas ruas desde o ano de 2020, e consequente diminuição dos acidentes (DETRAN PR, 2022).

2.2 MOBILIDADE URBANA E SMART CITIES

Uma *Smart City* é uma cidade que utiliza a Internet das coisas (IoT, do inglês *Internet of Things*) e análise de dados para otimizar a eficiência de seu funcionamento e serviços, e com isso melhorar a qualidade de vida de seus habitantes, partindo da perspectiva de que a tecnologia é o fator fundamental e indispensável para fornecer uma melhor infraestrutura para a população e gerenciamento dos recursos (ANDRADE E GALVÃO, 2016).

Aplicar estes conceitos na mobilidade urbana não é uma tarefa fácil. Segundo Silva e Silva (2018), um dos indicadores de um Transporte Público inteligente é ter uma boa área de cobertura, possibilitando o passageiro ir a qualquer lugar que necessite. O serviço deve ter uma frequência que não prejudique a rotina do usuário, a um preço acessível, com conforto para o passageiro e aproveitar as últimas tecnologias para o transporte. As rotas utilizadas devem ser mais diretas que o possível, sem que o passageiro necessite trocar de veículo.

Esses objetivos parecem comuns para qualquer transporte público, porém, *Smart Cities* utilizam tecnologia para inovar nas soluções, usando Ciência de Dados e a IoT para melhorar a tomada de decisão a curto e longo prazo, melhorando os modelos para o transporte público (KON e SANTANA, 2017).

Segundo Vasconcelos e Faria (2021), pesquisas com dados como origem e destino ainda não são periódicas, com isso não permite um planejamento técnico com uma margem menor de erros. Porém, nos anos atuais, a disseminação do uso de *smartphones* e da Internet criou um novo relacionamento do cidadão com o governo, viabilizando a realização de pesquisas de maneira mais simples, sem a necessidade da proximidade física.

Sistemas integrados e a rápida transmissão de dados - em conjunto com tecnologias GPS (*Global Positioning System*) em carros e vias - permitem o controle e monitoramento do tráfego das cidades em tempo real, favorecendo decisões rápidas e cada vez mais significativas.

Porém, segundo Andrade e Galvão (2016), uma *Smart City*, não se resume somente à tecnologia. É necessário, também, a governança da infraestrutura, assim como o capital humano e social, visando o desenvolvimento sustentável e econômico juntos. Como exemplo disso, vê-se cidades europeias como Amsterdã, Barcelona e Estocolmo, que possuem projetos de urbanização inteligentes, e contam com a população na participação da definição das políticas públicas e tomada de decisão.

No Brasil, destaca-se a cidade do Rio de Janeiro, que em 2015 foi considerada a cidade mais inteligente do Brasil pela Revista Exame (2015), um dos motivos dessa decisão é o COR, apresentado na Figura 1. Nesse mesmo ranking Curitiba ficou com a quinta posição, atrás de São Paulo, Belo Horizonte e Brasília.

FIGURA 1 - CENTRO DE OPERAÇÕES (COR)



FONTE: PREFEITURA DO RIO (2023)

Algumas dessas cidades compõem a Rede Brasileira de Cidades Inteligentes e Humanas (RBCIH), uma organização colaborativa que tem como objetivo implantar projetos de cidades inteligentes. A associação divulga e compartilha tecnologias entre os municípios membros, conectando agentes interessados em contribuir e adaptando-se às necessidades brasileiras (RBCIH, 2015).

Outro fator decisivo para um *Smart City*, é o alcance da população à Internet. Segundo o site Statista (2022), o Brasil possui 167,7 milhões de pessoas com acesso a internet e sua população no mesmo ano é de 207,7 milhões, isto é, mais

de 80% dos brasileiros têm acesso à internet de alguma forma. Na maioria das vezes o acesso a população é por meio de um celular. Segundo Andrade e Galvão (2018), esse é um fato imprescindível, pois o GPS desses aparelhos permitem proporcionar rotas e estimativas de tempo para o trânsito, entre outras informações importantes para o usuário.

2.3 GESTÃO DA MOBILIDADE URBANA EM CURITIBA

Criada no ano de 2012, a Lei 12.587 estabelece as diretrizes para a Política Nacional de Mobilidade Urbana, que foca no desenvolvimento sustentável das cidades e na priorização dos investimentos federais em meios não motorizados, e em modos coletivos. O Plano de Mobilidade Urbana tornou-se o instrumento de efetivação da Política Nacional de Mobilidade (MINISTÉRIO DAS CIDADES, 2013).

Diante desta determinação, os municípios com mais de vinte mil habitantes estão obrigados a entregar o Plano de Mobilidade Urbana, como condição para receber recursos federais destinados a projetos de mobilidade urbana. Também é de dever federal promover aos municípios capacitação contínua, apoiar ações entre estados e municípios, além de disponibilizar informações em um sistema nacional sobre mobilidade urbana.

Segundo a mesma lei, cabe ao estado gerir e integrar aglomerados urbanos e as regiões metropolitanas, além de ser responsável pelo transporte intermunicipal urbano. Aos municípios, cabe o importante papel de planejar e executar a política de mobilidade urbana e organizar e prestar os serviços de transporte público coletivo dentro do município.

Para a cidade de Curitiba e região metropolitana, os órgãos responsáveis pela mobilidade urbana são a URBS, pelo governo municipal, e a Coordenação da Região Metropolitana de Curitiba (COMEC), pelo governo estadual. A URBS é a responsável por Curitiba e 14 cidades da região metropolitana: Almirante Tamandaré, Araucária, Bocaiúva do Sul, Campo Largo, Campo Magro, Colombo, Contenda, Fazenda Rio Grande, Itaperuçu, Rio Branco do Sul, São José dos Pinhais, Pinhais e Piraquara. Esses municípios compõem a Rede Integrada de Transporte de Curitiba:

A Rede Integrada de Transporte Coletivo de Curitiba (RIT) permite ao usuário a utilização de mais de uma linha de ônibus com o pagamento de apenas uma tarifa. O processo de integração ocorre a partir de terminais de integração onde o cidadão pode desembarcar de uma linha e embarcar em qualquer outra dentro daquele espaço sem um novo pagamento. Assim, o usuário pode compor o seu próprio trajeto para se deslocar por diversos bairros de Curitiba (URBS, 2022).

A rede é responsável por 1.226 veículos em 250 linhas diferentes, como apresentado na Figura 2, percorrendo diariamente mais de 240 mil quilômetros em 11.425 viagens, utilizando as 333 estações tubo e 21 terminais disponíveis na

cidade, e usando os mais de 83 quilômetros de canaletas, com uma média diária de mais de um milhão de usuários (URBS, 2022).

FIGURA 2 - QUADRO DE VEÍCULOS DA URBS

Categoria de Linhas	Tipos de Veículo	Capacidade dos Veículos	Frota Operante Subtotal	Frota Operante Total	Quantidade de Linhas
EXPRESSO LIGEIRÃO	BIARTICulado	250	44	44	03
EXPRESSO	BIARTICulado	230/250	97	128	05
	ARTICulado	170	31		
LINHA DIRETA	ARTICulado	150	38	219	15
	PADRON	110	181		
INTERBAIRROS	ARTICulado	140	91	102	08
	PADRON	100	1		
	HÍBRIDO	79	10		
ALIMENTADOR	ARTICulado	140	71	426	129
	COMUM	85	326		
	MICRO ESPECIAL	70	29		
TRONCAL	ARTICulado	140	5	78	15
	COMUM	85	60		
	HÍBRIDO	79	10		
	MICRO ESPECIAL	70	3		
CONVENCIONAL	COMUM	85	102	217	74
	HÍBRIDO	79	10		
	MICRO ESPECIAL	70	102		
	MICRO	40	3		
TURISMO	DOUBLE-DECK	65	12	12	01
TOTAL			1.226	250	

(Fonte: URBS, 2023)

Já a COMEC é a responsável entre os municípios citados, por mais 15 cidades da região metropolitana de Curitiba incluindo, Doutor Ulysses, Cerro Azul, Adrianópolis, Tunas do Paraná, Campina Grande do Sul, Quatro Barras, Bolsa Nova, Lapa, Quitandinha, Mandirituba, Tijucas do Sul, Agudos do Sul, Piên, Campo do Tenente e Rio Negro. Somando todas as cidades de Curitiba e da região

metropolitana, tem-se uma população de 3.223.836 habitantes, correspondendo a mais de 30% da população do estado, também é a segunda maior região metropolitana do país por área, com 16.581,21km² (GOVERNO DO PARANÁ, 2023). Diante desse número de habitantes e área de atuação, em 2008 foi desenvolvido o PlanMob (PREFEITURA MUNICIPAL DE CURITIBA, 2008), que tem como objetivo auxiliar o desenvolvimento e aperfeiçoamento de um sistema de mobilidade urbana integrado de Curitiba com as conexões metropolitanas. O plano atende aos interesses e necessidades da população local, estendendo a acessibilidade com segurança e rapidez, especialmente para pessoas com deficiência física, idosas ou com mobilidade reduzida, tudo isso minimizando os impactos ambientais com relação a emissão de poluentes no ar e poluição sonora.

A URBS possui hoje alguns aplicativos próprios, por exemplo, com o Itibus o usuário pode visualizar o horário das linhas, rotas dos ônibus e localização dos pontos de ônibus; o Boletim do Transporte apresenta informações como desvios em rotas devido a eventos, como corridas de rua ou feriados como o carnaval, porém, em nenhum desses aplicativos é apresentada a localização dos veículos, por exemplo. Os passageiros não conseguem visualizar se um ônibus está atrasado, adiantado ou se ocorreu algum tipo de problema mecânico com ele. E também não conseguem saber a lotação de um veículo, podendo tomar a decisão de esperar um próximo ônibus ou optar por outra linha que esteja mais vazia.

Portanto, como apresentado até agora, se os critérios de qualidade do serviço de transporte público como conforto, pontualidade e serviço não foram condizentes com o preço da tarifa cobrada, o usuário tende a procurar outros meios de transporte, como aplicativos, táxis ou particular, aumentando o congestionamento das vias, e causando mais acidentes de trânsito, o que consequentemente diminui a velocidade das vias.

Na próxima seção será detalhado o estudo dos softwares que mais se assemelham ao produto final desejado.

2.4 ANÁLISE DOS SOFTWARES SEMELHANTES

Para a análise dos softwares semelhantes, no dashboard o tipo de sistema que mais se aproximou do projeto foram programas de visualização de voos em tempo real, como *Plane Finder* e o *Flightradar*. Esses sites apresentam a localização de voos comerciais, apresentando a localização atualizada do avião em tempo real, assim como informações como, código do voo, origem, destino, número dos voos da aeronave, altitude, velocidade, informações da aeronave como a idade do veículo, primeiro voo, mecânica do avião entre outras.

Outras características desses sites, como o filtro de busca, e a diferenciação do tipo de aeronave e a cor que é apresentada no dashboard, foram fatores que influenciaram neste trabalho também.

Para o aplicativo foram analisados os apps Curitiba 156, Metrocard e Moovit. Os três apresentam características semelhantes, apresentando informações como:

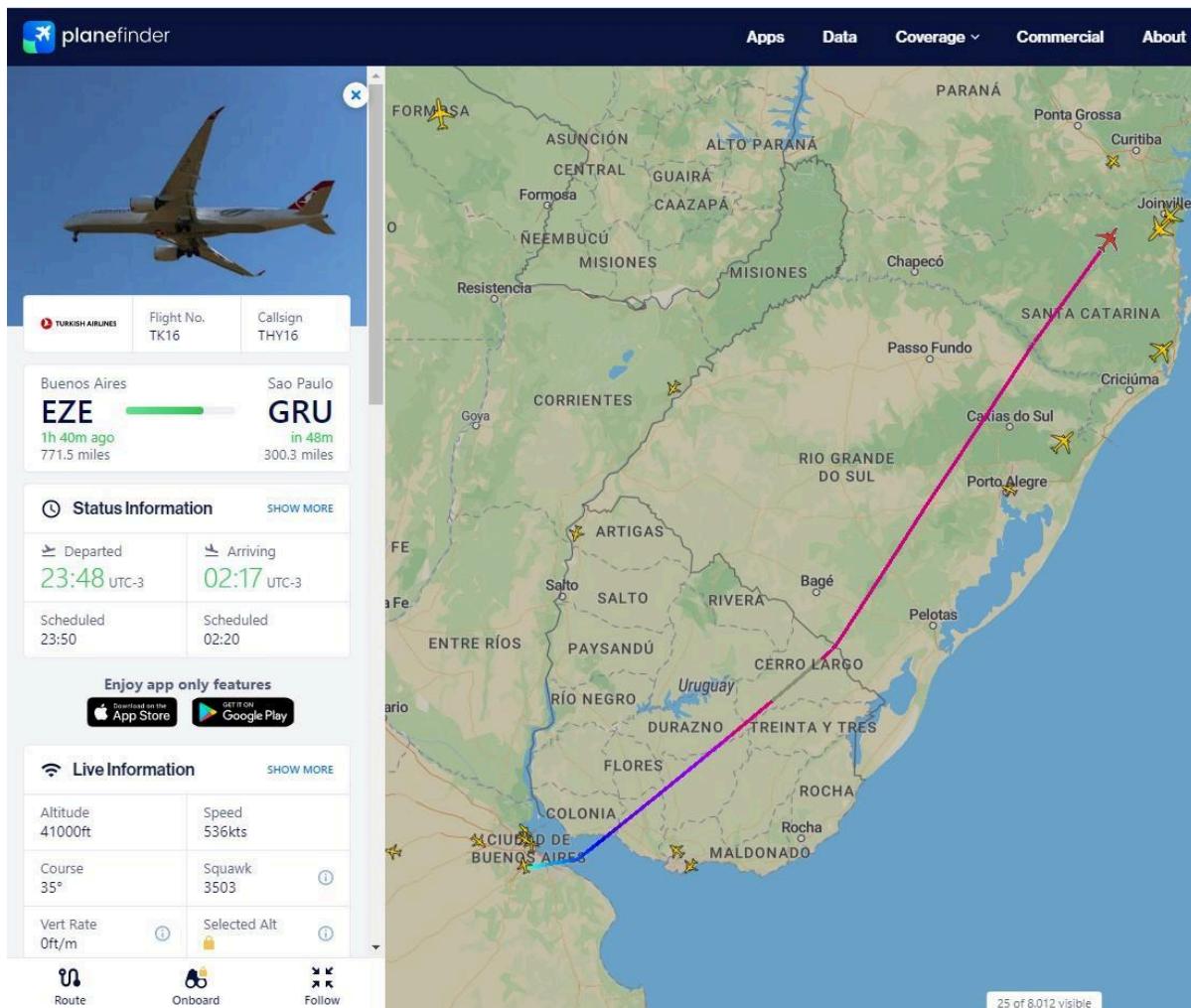
localização dos pontos de ônibus, localização dos veículos, itinerário entre outras funcionalidades que serão apresentadas a seguir.

2.4.1 Plane Finder

Apresenta os aviões em voo em tempo real, atualizando sua posição a cada certo intervalo de tempo, não tem distinção na apresentação das aeronaves, como apresentado na Figura 3. Ao colocar o *mouse* em cima do avião são apresentadas as informações do voo, código e companhia aérea, e ao clicar é destacado o trajeto além de algumas informações como horário de decolagem e pouso, origem, altitude, velocidade atual, entre outros. O site também possui uma série de filtros como aeronaves, aeroportos, companhias aéreas, altitude e velocidade (planefinder.net).

O software também oferece ferramentas mais complexas e filtros, como verificação de condições climáticas, sob pagamento de mensalidade.

FIGURA 3 - DETALHES DA AERONAVE PLANEFINDER



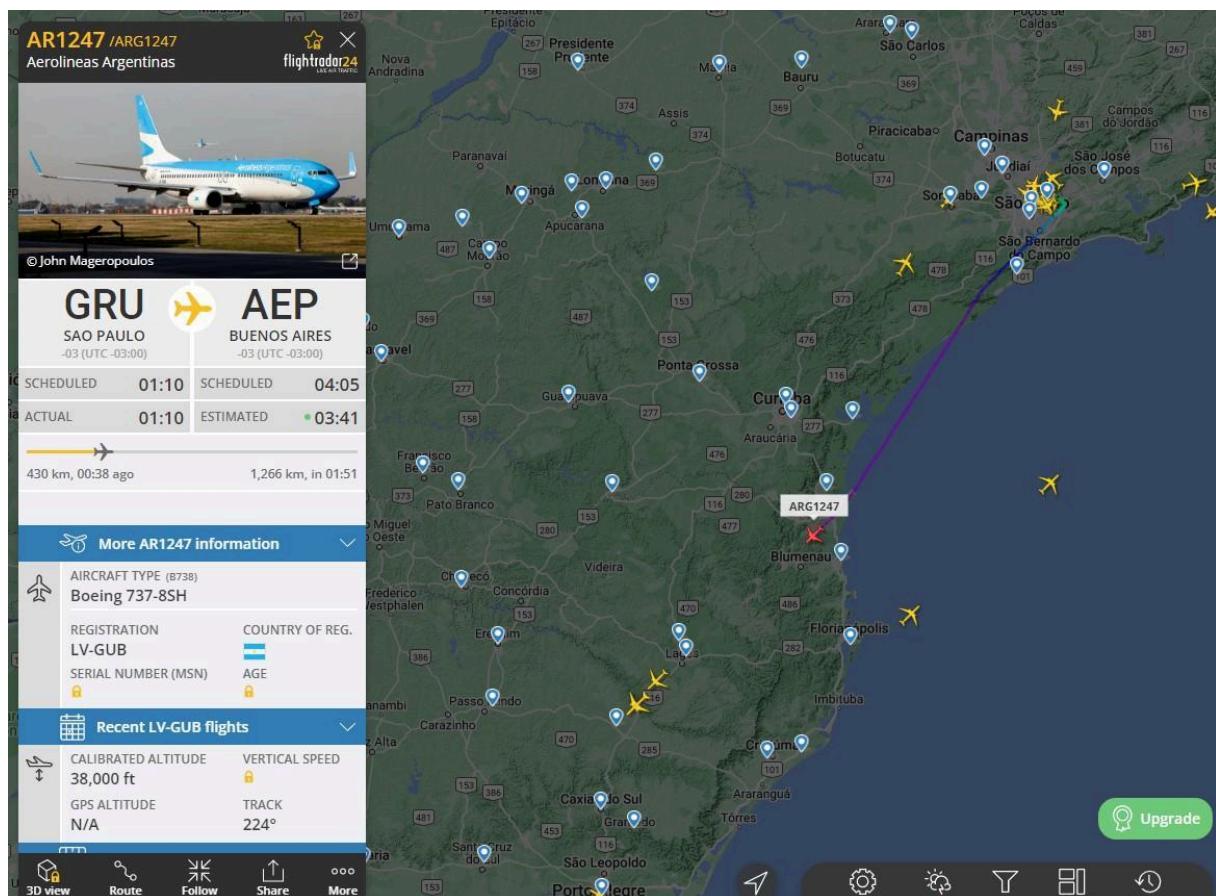
FONTE: planefinder.net (2023)

2.4.2 Flightradar24

Apresenta as aeronaves em tempo real, com atualização da localização em um intervalo de tempo (flightradar24.com). Os aviões são apresentados de formas diferentes, caso a aeronave esteja sendo rastreada por meios terrestres ela é apresentada em amarelo, se for por satélite então ela vai aparecer em azul, como apresentado na Figura 4. Ao reposar o *mouse* em cima de uma aeronave são apresentadas suas informações, e ao clicar é aberto uma caixa com algumas informações adicionais e é destacado o trajeto no mapa.

O site possui uma série de filtros, como rota, linha aérea, aeroportos e aeronaves próximas. Também possui algumas funcionalidades pagas, que alteram a visualização do terreno e luminosidade, que por não possuírem um período de teste a equipe não conseguiu visualizar.

FIGURA 4 - VISÃO GERAL FLIGHTRADAR24



FONTE: flightradar24.com (2023)

2.4.3 Moovit

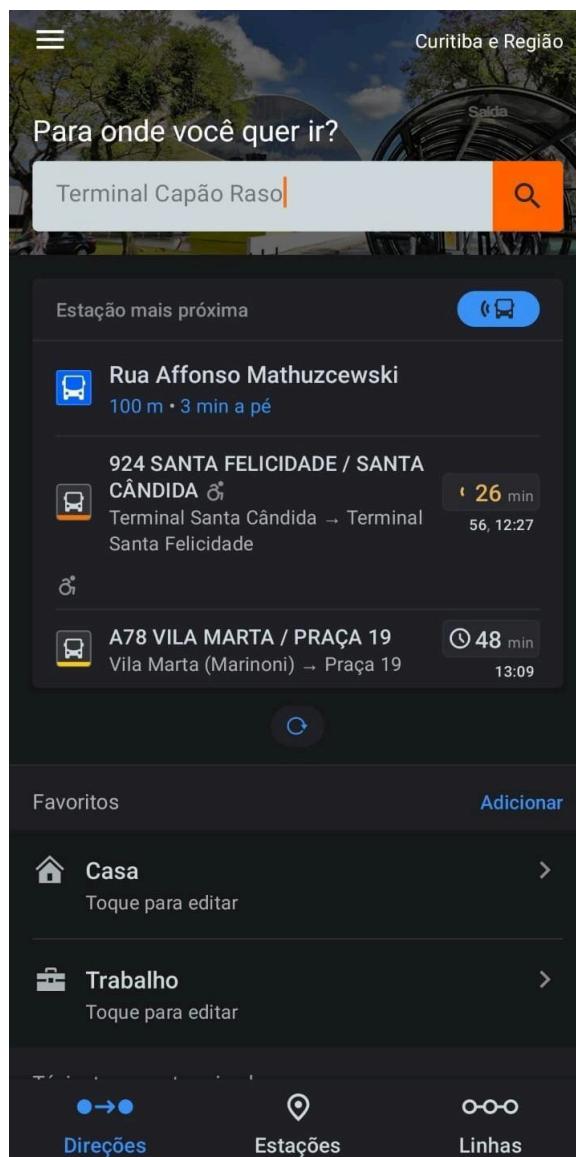
É um dos aplicativos mais utilizados de mobilidade urbana, pois oferece uma série de funcionalidades para os usuários. Entre suas principais características estão a integração de informações sobre transporte público em tempo real,

permitindo aos usuários planejar suas rotas de ônibus e outros meios de transporte coletivo (Figura 5), incluindo também serviços de bicicletas compartilhadas, táxis e serviços de carona (moovitapp.com/curitiba-942).

O aplicativo fornece alertas como atrasos em linhas, mudanças de itinerário e interrupção do serviço. Outra funcionalidade disponível é a compartilhamento da localização em tempo real, que pode ser feita entre os usuários do aplicativo. Porém algumas dessas funcionalidades só podem ser usadas com a assinatura do Moovit+ por meio de pagamento anual.

O aplicativo tem uma funcionalidade que permite que o usuário que acabou de entrar em um ônibus informe a quantidade de assentos disponíveis no veículo. Porém esse procedimento não é automático, e o usuário deve navegar até o menu para informar a quantidade de assentos. Nos testes realizados, não foi possível encontrar um ônibus que tivesse essa informação.

FIGURA 5 - MOOVIT



FONTE: Moovit 2023

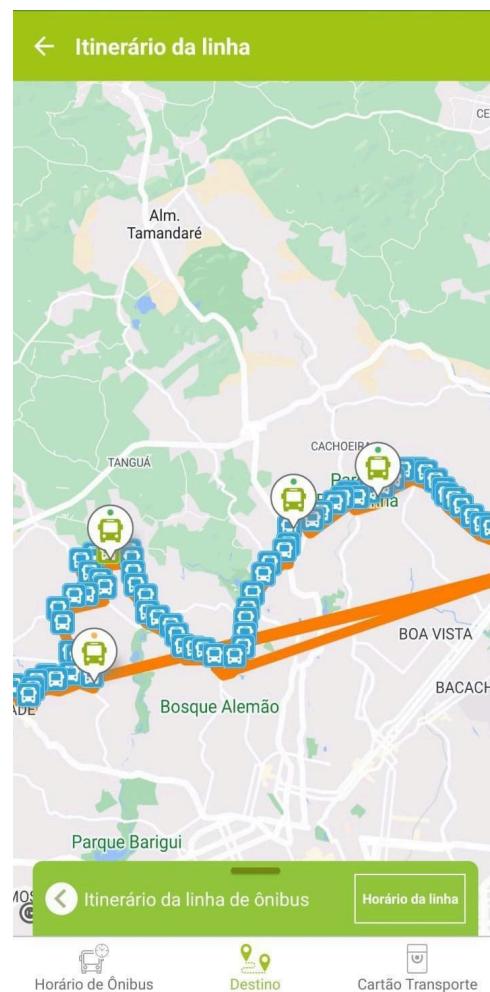
2.4.4 Curitiba 156

Outro software analisado foi o Curitiba 156 (156.curitiba.pr.gov.br/), que apresenta funcionalidades como solicitações e reclamações sobre diversos assuntos, como: iluminação pública, limpeza urbana, manutenção de vias e áreas verdes, agendamentos de consultas médicas, serviços de educação, segurança, e assistência social.

Além das funcionalidades citadas acima o aplicativo possui uma área dedicada à mobilidade urbana e linhas de ônibus da cidade, sendo possível visualizar em tempo real a localização dos ônibus e dos horários das linhas. É possível, assim, saber o tempo de chegada de cada veículo em cada ponto. O aplicativo também permite verificar mudanças de itinerário e horários, caso evento como corridas estejam acontecendo, e o trajeto das linhas seja alterado.

A visualização de trajeto das linhas no aplicativo, no entanto, apresenta alguns problemas, como apresentado na Figura 6, mostrando linhas ligando pontos que não fazem parte do itinerário, assim dificultando a visualização do trajeto da linha e seleção de pontos de ônibus no aplicativo.

FIGURA 6 - CURITIBA156



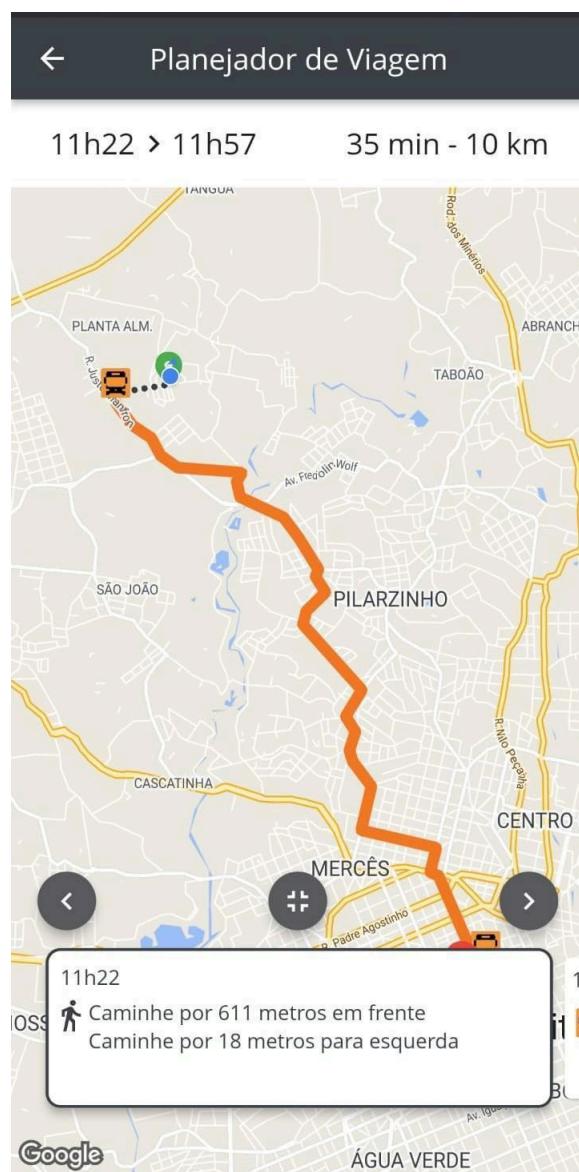
Fonte: Curitiba 156 2023

2.4.5 Metrocard

O último software que foi analisado foi o Metrocard, que além das funcionalidade de visualização de itinerários, visualização dos pontos de ônibus e planejamento de viagens, como apresentado na Figura 7. Também possui opção de recarga online do cartão Metrocard, permitindo que os usuários adicionem créditos sem a necessidade de deslocamento até pontos físicos de recarga. O aplicativo também oferece consulta de saldo e extrato de uso, proporcionando um controle dos gastos com transporte (cartaometrocard.com.br).

O usuário do aplicativo tem acesso a linhas apenas da região metropolitana de Curitiba, controlados pela Coordenação da Região Metropolitana de Curitiba (COMEC), que atende os municípios ao redor da capital, com isso o pagamento e recarga do cartão somente pode ser utilizados nas linhas intermunicipais.

FIGURA 7 - METROCARD



Fonte: Metrocard 2023

Portanto, como apresentado neste capítulo, conceitos como *smart cities*, sistemas integrados e a tecnologia em geral podem ajudar na melhoria contínua da gestão da mobilidade urbana, e cidades que aplicam esses conceitos obtiveram resultados tangíveis na questão da melhoria do transporte público.

Na próxima seção são descritas as metodologias e ferramentas utilizadas no desenvolvimento desse projeto, assim como o cronograma das sprints realizadas e as atividades exercidas por cada membro da equipe.

3. MATERIAIS E MÉTODOS

Este capítulo descreve os métodos e ferramentas utilizados para o desenvolvimento deste trabalho.

3.1 MODELO DE PROCESSO DE ENGENHARIA DE SOFTWARE

Engenharia de Software é o processo de estudar, criar e otimizar os processos de trabalho para o desenvolvimento de um software (WAZLAWICK, 2013). Isso não engloba somente a fase de desenvolvimento, mas também toda a análise e projeto necessários para o desenvolvimento do software.

3.1.1 Metodologia ágil

A metodologia ágil é uma combinação de filosofia e um conjunto de princípios de desenvolvimento, defendendo a satisfação do cliente e a entrega incremental antecipada com equipes pequenas, métodos informais, artefatos de engenharia de software mínimos e simplicidade no desenvolvimento geral (PRESSMAN; MAXIM, 2016).

Segundo Booch, Rumbaugh e Jacobson (2005), a metodologia ágil é uma abordagem iterativa e incremental para o desenvolvimento de software que enfatiza a flexibilidade, a colaboração constante com os clientes e a rápida adaptação a mudanças. Esta metodologia valoriza a entrega contínua de software funcional, permitindo ajustes frequentes e incorporando feedback ao longo do ciclo de desenvolvimento. As metodologias ágeis, como SCRUM e Extreme Programming (XP), promovem equipes auto-organizadas e multifuncionais, que trabalham em ciclos curtos e focam na melhoria contínua e na comunicação eficaz. Esta abordagem contrasta com os métodos tradicionais de desenvolvimento, proporcionando maior capacidade de resposta às necessidades do negócio e mudanças tecnológicas (BOOCH; RUMBAUGH; JACOBSON, 2005).

Uma das metodologias mais utilizadas é o *Scrum*. É um framework ágil de gerenciamento de projetos que tem como objetivo aumentar a eficiência e a produtividade dos times de desenvolvimento. Ele é muito utilizado na área de tecnologia, mas também pode ser aplicado em outras áreas.

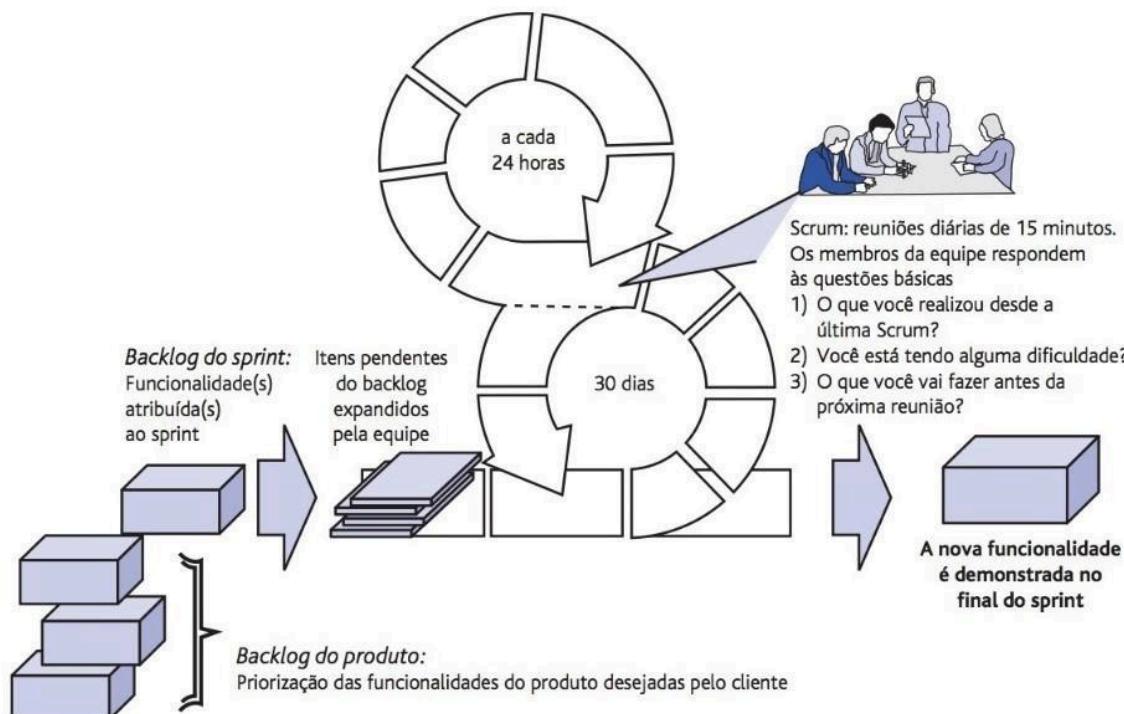
Segundo Pressman e Maxim (2016), os princípios do *Scrum* são coerentes com a metodologia ágil, incorporando atividades como requisitos, análise, projeto, evolução e entrega. Essas atividades são desenvolvidas em uma janela de tempo chamada *Sprint*. A quantidade de trabalho em cada *Sprint* é adaptada à questão atual do projeto, e definido e modificado em tempo real, como apresentado na Figura 8. Dessa maneira o número de *Sprints* necessárias para cada uma das atividades varia conforme a complexidade do produto, e cada *Sprint* geralmente tem duração de 2 a 4 semanas.

A equipe *Scrum* é interdisciplinar, e composta por três papéis: o *Product Owner*, o *Scrum Master* e o Time de Desenvolvimento. O *Product Owner* é

responsável por definir as prioridades do projeto, gerenciar o *backlog* do produto e garantir que o time esteja construindo algo de valor. O *Scrum Master* é responsável por garantir que o processo do *Scrum* seja seguido corretamente e remover quaisquer impedimentos que estejam atrapalhando o time. O Time de Desenvolvimento é responsável por desenvolver o produto em si, seguindo as prioridades definidas pelo *Product Owner* (PRESSMAN; MAXIM, 2016).

Segundo Pressman e Maxim (2016), os principais artefatos do *Scrum* são o *Backlog* do produto, *Backlog da Sprint* e o incremento do produto. O *Backlog* do produto é uma lista de requisitos ou características do artefato que agregam valor de negócio para o cliente, ordenados geralmente por prioridade, podendo ser adicionado itens ao *Backlog* a qualquer momento, com a aprovação do *product owner*, e o consentimento da equipe. O *Backlog da Sprint* é um subconjunto de itens do *Backlog* do produto, selecionado pela equipe para ser completado, assim gerando incremento do produto durante a *Sprint* atual. Por último, o incremento representa a união de todos os itens do *Backlog* do produto completados em *Sprint* anteriores, e todos os itens do *Backlog* a serem completados na *Sprint* atual.

FIGURA 8 - MÉTODO SCRUM



FONTE: PRESSMAN; MAXIM (2016)

3.1.2 Adaptação do Scrum as necessidades da equipe

Para a metodologia de desenvolvimento, optou-se por usar o Scrum, porém adaptado quando necessário para atender as necessidades da equipe. No Scrum, geralmente são realizadas reuniões diárias. Porém, para melhor atender a equipe

decidiu-se por realizar reuniões uma ou duas vezes por semana, dependendo da complexidade da sprint. As reuniões podiam ser *on-line* pelo aplicativo Discord ou presenciais, dependendo da disponibilidade dos participantes da equipe.

Outra adaptação feita foi com relação à duração das *sprints*. Como observado no item anterior, geralmente a duração varia de duas a quatro semanas. No entanto, para tornar a comunicação e execução do projeto mais dinâmica, a duração das *sprints* foi de uma semana.

3.1.3 Modelagem do software

A *Unified Modeling Language* (UML) ou, em português, Linguagem de Modelagem Unificada é uma linguagem visual para especificar, construir e documentar os artefatos de um sistema (LARMAN, 2007). Ela se baseia nas boas práticas de engenharia de software e no paradigma de orientação a objetos. Para esse projeto, foi utilizada a UML como linguagem de especificação do software, desenvolvendo diagramas detalhados como diretrizes para o desenvolvimento, que será realizado na segunda etapa do projeto.

Já segundo Booch, Rumbaugh e Jacobson (2005), a *Unified Modeling Language* (UML) é uma linguagem de modelagem padronizada que proporciona uma forma de visualizar, especificar, construir e documentar artefatos de sistemas de software. A UML combina as melhores práticas de engenharia de software em uma linguagem visual, permitindo a criação de diagramas que representam diferentes aspectos de um sistema, como diagramas de classes, casos de uso, atividades e sequências. Essa linguagem facilita a comunicação entre os membros da equipe de desenvolvimento e outras partes interessadas, promovendo um entendimento claro e comum do sistema projetado (BOOCH; RUMBAUGH; JACOBSON, 2005).

3.1.3.1 Diagrama de Caso de Uso

Segundo Pressman e Maxim (2016), em um caso de uso é descrito uma jornada estilizada sobre as interações de um usuário podendo desempenhar uma série de papéis, com o sistema, sobre um conjunto de circunstâncias específicas. Como o sistema foi dividido em duas partes, Dashboard e Aplicativo, ele possui dois diagramas diferentes. O diagrama de caso de uso do *Dashboard* pode ser encontrado no Apêndice A, e o diagrama do aplicativo consta no Apêndice A.

3.1.3.2 Histórias de Usuário

Histórias de usuário são descrições breves e simples de uma funcionalidade ou característica do sistema escritas do ponto de vista do usuário final. Elas servem como um método eficaz de capturar requisitos de software, promovendo a comunicação entre desenvolvedores e usuários. Conforme Larman (2007), histórias de usuário ajudam a focar no valor do negócio e na usabilidade, fornecendo uma

linguagem comum que todos os stakeholders podem entender. Além disso, Pressman e Maxim (2016) destacam que essas histórias facilitam a priorização de funcionalidades e a adaptação contínua do sistema às necessidades do usuário, promovendo um desenvolvimento iterativo e incremental. As histórias de usuário estão presentes no Apêndice B.

3.1.3.3 Diagrama de Classes

O diagrama de classes apresenta a estrutura do sistema por meio de classes, com seus respectivos atributos, operações e relações entre os objetos (LARMAN, 2007). O diagrama de classe deste projeto está presente no Apêndice C.

3.1.3.4 Modelo Físico do Banco de Dados

O modelo físico de banco de dados é uma representação detalhada de como os dados são armazenados e organizados no sistema de gerenciamento de banco de dados (SGBD). Esse modelo inclui especificações técnicas como tipos de dados, índices, partições, e outras estruturas de armazenamento que otimizam o desempenho e a eficiência do acesso aos dados. Diferente do modelo lógico, que foca na estrutura e nas relações entre os dados, o modelo físico aborda aspectos práticos de implementação, garantindo que o banco de dados funcione de maneira eficaz no ambiente físico onde será executado (LARMAN, 2007). O modelo físico do banco de dados está presente no Apêndice D.

3.1.3.5 Diagrama de Sequência

O diagrama de sequência mostra o processo de interações entre objetos em uma linha sequencial, ou seja, a sequência de mensagens trocadas entre os objetos necessários para realizar as funcionalidades do sistema, assim, apresentando a invocação dos métodos (LARMAN, 2007). Os diagramas de sequência são apresentados no Apêndice E.

3.2 FERRAMENTAS DE DESENVOLVIMENTO

Os tópicos a seguir apresentam as ferramentas que se planeja utilizar para o desenvolvimento deste trabalho de conclusão de curso.

3.2.1 Angular

Angular é um Framework de código aberto e *front-end* para aplicações web baseado em TypeScript desenvolvido pela Google, lançado no ano de 2016. Segundo o site oficial, o Angular foi desenvolvido para suportar desde projetos pequenos até aplicações corporativas robustas, de forma direta e eficiente (ANGULAR, 2023).

3.2.2 Python

Python é uma linguagem de programação de alto nível, interpretada e de propósito geral, conhecida por sua sintaxe clara e legível, que facilita a aprendizagem e a utilização. Criada por Guido van Rossum e lançada em 1991, Python suporta múltiplos paradigmas de programação, incluindo programação orientada a objetos, imperativa, funcional e procedural. Ela é amplamente utilizada em diversas áreas, como desenvolvimento web, ciência de dados, automação, inteligência artificial e muitos outros campos. A vasta biblioteca padrão e a comunidade ativa de desenvolvedores tornam Python uma escolha popular tanto para iniciantes quanto para programadores experientes (PYTHON, 2024).

3.2.3 Java

Java é uma linguagem de programação de propósito geral, orientada a objetos, que foi projetada para ter o mínimo de dependências de implementação, o que a torna ideal para desenvolvimento de aplicativos portáteis e escaláveis. Desenvolvida originalmente pela Sun Microsystems e mantida pela Oracle, Java é amplamente utilizada. Java é utilizada em uma variedade de aplicações, desde desenvolvimento de aplicativos Android, sistemas corporativos, desenvolvimento web até soluções de big data e computação em nuvem (ORACLE, 2024).

3.2.4 Redis

O Remote Dictionary Server é um software de código aberto lançado em 2009, e é uma estrutura de dados em memória, usado como banco de dados em memória distribuído de chave-valor ou cache. Ele suporta diferentes estruturas de dados abstratas como strings, listas, conjuntos, conjuntos classificados, hyperlogs, e índices especiais (REDIS, 2023).

3.2.5 MongoDB

MongoDB é um banco de dados NoSQL de código aberto que utiliza um modelo de dados orientado a documentos, permitindo a armazenagem de dados em formato BSON (uma extensão binária de JSON). Desenvolvido pela MongoDB Inc., ele foi projetado para lidar com grandes volumes de dados e alta demanda de leitura e escrita, oferecendo escalabilidade horizontal e flexibilidade na modelagem de dados. O MongoDB é amplamente utilizado em aplicações modernas, incluindo desenvolvimento web, *big data* e análise em tempo real, devido à sua capacidade de lidar com estruturas de dados complexas e não estruturadas de maneira eficiente (MONGODB, 2024).

3.2.6 Elasticsearch

O Elasticsearch é uma ferramenta de busca e análise distribuída, de código aberto, capaz de processar grandes volumes de dados em tempo real. Ele oferece funcionalidades avançadas de busca, agregação e análise de dados, permitindo consultas rápidas e detalhadas em grandes conjuntos de dados. Outro diferencial é oferecer uma interface RESTful, assim se torna uma escolha popular para análise de dados complexos e em tempo real (ELASTIC, 2024).

3.2.7 JSON Web Token - JWT

O JSON Web Token (JWT) é um método de autenticação baseado em *token*. É um padrão aberto com base no RFC 7519, criando uma assinatura criptografada através de um objeto JSON.

A vantagem de utilizar o JWT - em vez de abordagens como *cookies* e de sessões no servidor - é que, para utilização destas, é necessário manter e gerir as informações de cada usuário no servidor; o que não ocorre com o JWT que apenas verifica a consistência da assinatura (CONCEIÇÃO, 2015).

3.2.8 API Google Maps

A API Google Maps é uma ferramenta fornecida pelo Google que permite aos desenvolvedores integrar mapas interativos e recursos avançados de geolocalização em aplicações web e móveis. Com essa API, é possível adicionar mapas personalizados, geocodificação, direções, visualização de trânsito, e muitos outros recursos de geolocalização. A API oferece uma ampla gama de funcionalidades, incluindo a capacidade de desenhar rotas, calcular distâncias, adicionar marcadores, criar camadas de dados personalizados e integrar serviços como Street View e Places (GOOGLE, 2024).

3.4.9 Google Markers

A biblioteca Google Markers é uma ferramenta do Google Maps JavaScript API que permite a criação e manipulação de marcadores em mapas, facilitando a visualização de pontos de interesse personalizados. Com essa biblioteca, desenvolvedores podem adicionar marcadores interativos a um mapa, definir suas propriedades, como ícones personalizados, e gerenciar eventos associados, como cliques ou movimentos do mouse. Além disso, o Google Markers suporta funcionalidades avançadas, como clusters de marcadores para melhor performance em mapas com muitos pontos. Essa biblioteca possibilita a criação de mapas dinâmicos e interativos em aplicações web, melhorando significativamente a experiência do usuário (GOOGLE DEVELOPERS, 2024).

3.2.10 FastAPI

FastAPI é um *framework web* de código aberto para a construção de APIs com Python. Projetado para oferecer alta performance, ele utiliza tipagem assíncrona e validação automática de dados proporcionando desenvolvimento rápido e eficiente. Ele suporta recursos como autenticação, roteamento, documentação automática com *Swagger* e *Redoc*, e é totalmente compatível com ASGI, tornando-o ideal para aplicações que requerem alta concorrência e tempo de resposta rápido. Sua capacidade de gerar documentação interativa e executar validações de dados com base nos tipos anotados melhora significativamente a produtividade e a manutenção do código (FASTAPI, 2024).

3.2.11 Geopy

O *Geopy* é uma biblioteca *Python* que facilita a geocodificação, permitindo a conversão de endereços em coordenadas geográficas (latitude e longitude) e vice-versa. Além da geocodificação, o *Geopy* oferece suporte para cálculos de distâncias entre pontos geográficos, bem como outras operações geoespaciais. Ele suporta diversos serviços de geocodificação, como Google Geocoding API, OpenStreetMap Nominatim, Bing Maps API, entre outros (GEOPY, 2024).

3.2.12 Pandas

Pandas é uma biblioteca de código aberto para manipulação e análise de dados em Python. Ela fornece estruturas de dados flexíveis e eficientes, como *Data Frames* e *Séries*, que permitem a manipulação e análise de dados tabulares e heterogêneos. Com Pandas, é possível realizar operações complexas de dados, incluindo limpeza, transformação, agregação e visualização (PANDAS, 2024).

3.2.13 Docker

Docker é uma plataforma de código aberto que automatiza o desenvolvimento, o envio e a execução de aplicações em contêineres. Os containers Docker encapsulam uma aplicação e suas dependências em um ambiente leve e portátil, garantindo que o software funcione de forma consistente em diferentes ambientes, desde a fase de desenvolvimento até a produção. Docker oferece ferramentas para criar, gerenciar e orquestrar contêineres, incluindo o Docker Engine, que é responsável pela execução dos contêineres, e o Docker Hub, um repositório de contêineres que facilita o compartilhamento e a distribuição de imagens. A utilização de Docker permite maior eficiência na utilização de recursos, isolamento de aplicações, e maior flexibilidade no desenvolvimento e implantação de software (DOCKER, 2024).

3.2.14 Portainer

Portainer é uma plataforma de gerenciamento de contêineres de código aberto que simplifica a gestão de ambientes Docker e Kubernetes. Ele fornece uma interface gráfica de usuário (GUI) intuitiva, permitindo que administradores e desenvolvedores gerenciem contêineres, imagens, redes e volumes. Com Portainer, é possível visualizar e monitorar o status dos contêineres, implantar aplicativos, gerenciar stacks e serviços, além de configurar políticas de segurança e autenticação de usuários. Portainer é compatível com diversas tecnologias de orquestração de contêineres, facilitando a administração tanto de ambientes locais quanto de nuvem, e proporcionando uma experiência unificada para gerenciamento de infraestrutura de contêineres (PORTAINER, 2024).

3.2.15 Cloudflare DNS

Cloudflare DNS é um serviço de resolução de nomes de domínio (DNS) fornecido pela Cloudflare. Projetado para melhorar a performance e a segurança da navegação na internet, o Cloudflare DNS oferece uma resolução de nomes de domínio, garantindo que os usuários possam acessar websites com o menor tempo de resposta possível. O serviço inclui recursos de segurança, como proteção contra ataques DDoS e mitigação de ameaças, ajudando a proteger tanto os usuários finais quanto os servidores dos sites (CLOUDFLARE, 2024).

3.2.16 Outras Ferramentas

Aqui serão descritas outras ferramentas utilizadas como apoio para o desenvolvimento deste trabalho de conclusão de curso.

3.2.16.1 Discord

Para as reuniões *on-line* foi utilizado o aplicativo Discord. Ele possibilita criação de canais com salas separadas para conversa, e também possibilita uma maior facilidade para o compartilhamento de códigos.

3.2.16.2 Microsoft Office

Para o documento escrito foi utilizado o Microsoft Word. Além disso, também foi utilizado o OneDrive para compartilhamento e armazenamento da documentação do trabalho.

3.2.16.3 BrModelo

Por ser uma ferramenta gratuita, utilizou-se o BrModelo para criação dos modelos conceituais e lógicos do banco de dados. A ferramenta foi selecionada, também, pela familiaridade da equipe com seu funcionamento.

3.2.16.4 Astah

Para os diagramas foi utilizado o Astah, uma ferramenta de fácil utilização e que atende todas as necessidades da equipe para a criação da documentação UML. Além disso, é possível obter uma licença gratuita para estudantes.

3.2.16.5 Lucidchart

Para os diagramas também foi utilizado o Lucidchart que é uma aplicação web gratuita que facilita a criação de diagramas e modelos visuais colaborativos.

3.2.16.6 Git

Git é um sistema de código aberto para versionamento de códigos e de distribuição livre que possibilita desenvolvimento colaborativo de forma paralela. Além disso, oferece suporte para diversos sistemas operacionais, demonstrando versatilidade (GIT, 2024). Para o processo de desenvolvimento, foi utilizada a distribuição GitHub devido a sua simplicidade e familiaridade oriunda de sua aplicação em outros trabalhos.

3.3 HARDWARE

- Notebook 1
 - Nome da máquina: mateus
 - Proprietário: Mateus Medeiros Garcia
 - Fabricante: Lenovo
 - Sistema Operacional: Arch Linux
 - Processador: Intel(R) Core(TM) i7-1065G7 CPU @ 1.30GHz
 - Memória RAM: 8Gb
 - Espaço de armazenamento: 256Gb SSD
- Notebook 2
 - Nome da máquina: lucas
 - Proprietário: Lucas de Mello Cesar
 - Fabricante: Apple
 - Sistema Operacional: macOS Versão 11.7.3
 - Processador: Intel(R) Core(TM) i5 Dual-Core 2.8GHz
 - Memória RAM: 8Gb
 - Espaço de armazenamento: 500Gb HD
- Notebook 3
 - Nome da máquina: DESKTOP-JJFUD3U

- Proprietário: Lucas Schip do Nascimento
- Fabricante: Dell
- Sistema Operacional: Ubuntu
- Processador: Intel(R) Core(TM) i7-8750H CPU @ 2.20GHz
- Memória RAM: 16Gb
- Espaço de armazenamento: 240Gb SSD

3.4 DESENVOLVIMENTO DO PROJETO

Esta seção apresenta a organização da equipe e das atividades para o desenvolvimento do projeto..

3.4.1 Plano de Atividades

Segundo Pressman e Maxim, o cronograma de projeto de *software* é uma ação de distribuir todo o esforço estimado do projeto em uma duração estimada, alocando esse esforço em tarefas específicas da engenharia de *software*.

O Quadro 1 apresenta como as atividades foram distribuídas entre os membros da equipe.

QUADRO 1 - DIVISÃO DE RESPONSABILIDADES

Integrante	Atividade
Lucas de Mello Cesar	Análise dos Softwares Semelhantes Aplicativo Diagrama de Sequência Protótipo das Telas
Lucas Schip do Nascimento	Diagrama de Caso de Uso Documentação Sprints Documento Escrito Frontend Histórias de Usuário
Mateus Medeiros Garcia	BackEnd Banco de Dados (Mongodb e Redis) Diagrama de Classe Modelo do Banco de Dados Prova de Conceito

FONTE: Os Autores(2023)

3.4.2 Cronograma das *Sprints*

Este projeto foi dividido em duas partes, uma focada na análise e modelagem e outra focada no desenvolvimento. A Figura 9 mostra as datas das sprints da

primeira parte. Já a Figura 10, mostra as datas das sprints da segunda parte, ambas explicadas em seguida.

FIGURA 9 - CRONOGRAMA DAS SPRINTS 1

Nome	Recursos Entregues
Sprint 1	Softwares semelhantes, Sugestão das funcionalidades
Sprint 2	Estudo das ferramentas para o desenvolvimento
Sprint 3	Estudo da API da URBS, Sugestão das funcionalidades
Sprint 4	Diagrama de caso de uso, protótipo das telas, Diagrama de classe, Histórias de usuário
Sprint 5	Diagrama de classe, Diagrama de objetos, Capítulo 1
Sprint 6	Diagrama de classe, Capítulo 2
Sprint 7	Modelo Físico do BD, Protótipo das telas, Prova de conceito, Capítulo 2
Sprint 8	Diagrama de sequência, Capítulo 3; Especificação das funcionalidades da prova de conceito
Sprint 9	Implementação da prova de conceito, Capítulo 1, Capítulo 2, Capítulo 3, Diagrama de sequência

FONTE: OS AUTORES (2023)

Sprint 1

A primeira *Sprint* teve duração de duas semanas, ela foi inicializada no dia 21/10 e finalizada no dia 04/11. A *Sprint* teve início com uma reunião presencial para discussão e levantamento de ideias para o projeto de TCC.

Após decidido que prosseguiremos com o projeto da URBS, foi feito uma pesquisa com softwares semelhantes e proposta funcionalidades. No final da sprint foi entregue o resultado do levantamento de softwares semelhantes

Sprint 2

Na sprint 2 realizou-se um estudo sobre quais ferramentas poderiam ser utilizadas para a exibição do mapa e como é feito a movimentação de objetos nelas. E também foi feito um estudo das tecnologias que têm sido utilizadas mais recentemente com performance de tempo real para leitura dos dados do banco e atualização do mapa.

A sprint teve duração de uma semana, com início no dia 4/11 e final no dia 11/11. No final foi entregue um documento com um software escolhido para exibição dos mapas com exemplos de implementação.

Sprint 3

A *Sprint* 3 teve duração de uma semana, iniciando no dia 11/11 e finalizando no dia 18/11. Nesta Sprint foi realizado um estudo sobre a API que a URBS

disponibilizou e evoluiu-se as funcionalidades com base no estudo de softwares semelhantes.

No final da sprint, foi entregue um documento com o resultado da análise de softwares semelhantes e sugestões de funcionalidades.

Sprint 4

A *Sprint 4* teve duração de uma semana, começou no dia 18/11 e terminou no dia 25/11. Nesta Sprint foi feito o diagrama de caso de uso, protótipos e histórias de usuários. Foi iniciado o diagrama de classes.

Sprint 5

A *Sprint 5* teve duração de uma semana, com início no dia 25/11 e foi finalizada no dia 19/12. Nesta Sprint foram realizados ajustes no diagrama de classes, e também foi feito um diagrama de objetos, com o intuito de melhor entender o sistema, e foi feita a proposta do capítulo de introdução.

Sprint 6

A *Sprint 6* teve duração de três semanas, com início no dia 19/12 e foi finalizada no dia 20/01. Diferentemente das demais, essa sprint teve uma duração maior por conta do recesso de final de ano. Durante a sprint foram realizados ajustes no diagrama de classes.

Sprint 7

A *Sprint 7* teve duração de uma semana, com início no dia 20/01 e foi finalizada no dia 27/01. Nesta Sprint foi especificado o modelo físico do banco de dados, ajustes nos protótipos de tela, a proposta da prova de conceito e proposta do Capítulo 2 completo.

Sprint 8

A *Sprint 8* teve duração de uma semana, com início no dia 27/01 e foi finalizada no dia 03/02. Nesta Sprint foi feito os diagramas de sequência, a proposta do Capítulo 3 e especificação das funcionalidades da prova de conceito.

Sprint 9

A *Sprint 9* teve duração de uma semana, com início no dia 03/02 e foi finalizada no dia 16/02. Nesta Sprint foram realizados ajustes nos diagramas e documentação, e início da implementação da prova de conceito.

Na Figura 10, é apresentado o cronograma utilizado no desenvolvimento da segunda parte do projeto.

FIGURA 10 - ROTEIRO SEGUNDA PARTE PROJETO

Nome	Recursos Entregues
Sprint 1	Planejamento das Entregas
Sprint 2	Dashboard: Entrega: HU001 - Login, HU002 - Cadastrar Administrador HU003 - Exibir veículos em operação, HU004 - Exibir rotas da linha, HU005 - Exibir paradas da linha, HU006 - Exibir detalhes do ônibus, HU007 - Filtrar linhas
Sprint 3	Dashboard: HU008 - Filtrar ônibus Aplicativo: HU001 - Realizar login, HU002 - Realizar cadastro no sistema, HU004 - Favoritar linhas
Sprint 4	Dashboard: HU009 - Exibir quantidade de pessoas por ônibus, HU010 - Gerar relatório de movimentação das linhas, HU011 - Gerar indicadores Aplicativo: HU003 - Listar Linhas, HU005 - Exibir itinerário da linha, HU006 - Visualizar veículos em operação
Sprint 5	Entrega Documento Final

FONTE: OS AUTORES (2023)

Na segunda parte do projeto as reuniões foram alteradas para ocorrerem a cada duas semanas, e segue um resumo das sprints.

Sprint 1

A primeiro *sprint* ocorreu no dia 11/03, e foi decidido que seria intercalado entregas e reuniões de acompanhamento, com as entregas sendo nos dias 08/04, 06/05, 03/06 e 17/06, e as reuniões de acompanhamento a cada duas semanas no intervalo das entregas.

Sprint 2

A terceira *sprint* durou do dia 12/03 a 08/04, e no final foram entregues os casos de uso HU001 - Realizar Login; HU002 - Realizar cadastro no sistema; do aplicativo, assim como a prévia da apresentação das linhas no mapa. No *dashboard* acabamos tendo alguns problemas, pois a versão do Angular usada na primeira fase do projeto estava defasada, então foi decidido que todo o *frontend* seria refeito utilizando a versão mais recente do Angular.

Sprint 3

A terceira *sprint* teve início em 09/04 até 06/05, e foram entregues os casos de uso HU001 - Realizar Login; HU002 - Realizar cadastro; HU003 - Exibir veículo em operação, HU004 - Exibir rotas das linhas, HU005 - Exibir paradas das linhas; HU007 - Filtrar Linhas; do *dashboard*, assim como os casos de uso HU003 - Listar linhas; HU005 - Exibir itinerário das linhas; e HU006 - Exibir veículos em operação; do aplicativo.

Sprint 4

A quarta *sprint* teve início em 07/05 e durou até 03/06, nela foi entregue o HU004 - Favoritar linhas, do aplicativo, assim como HU007 - Exibir detalhes das

linhas e HU008 - Filtrar ônibus, assim como uma série de melhorias e validações na parte do *dashboard*, como validação de login e cadastro apresentando mensagem mais coesas sobre erros como CPF inálido ou usuário já cadastrado.

Sprint 5

A quinta e última sprint teve início em 04/06 e foi até dia 17/06, nesta data foram entregues às HU009 - Exibir quantidades de pessoas no ônibus; HU010 - Gerar relatórios e HU011 - Gerar indicadores. Além disso, todo o documento gerado na primeira parte do projeto foi revisado, devido a algumas alterações na arquitetura, decididas na segunda fase do projeto.

Neste capítulo foi apresentado a metodologia, ferramentas e o cronograma de desenvolvimento desse projeto, assim como as atividades feitas por cada integrante da equipe e o *Hardware* utilizado. No capítulo seguinte será apresentado a arquitetura do sistema, assim como o seu funcionamento detalhado.

4. APRESENTAÇÃO DO SISTEMA

Este capítulo tem como objetivo apresentar a arquitetura do sistema, as telas e funcionalidades desenvolvidas.

Em linhas gerais, o sistema foi desenvolvido consiste de um *Dashboard* e um aplicativo mobile. O *Dashboard* permite o controle em tempo real da frota, permitindo visualizar os veículos em operação da frota da URBS. No *Dashboard* foram implementadas uma série de filtros para facilitar a visualização de informações como veículos atrasados, adiantados e no horário. Outros filtros também foram implementados, como por tipo de veículo, para linhas que utilizam ligeirinhos, somente veículos bí-articulados, veículos elétricos, híbridos, etc.

Além disso, também é possível selecionar qualquer linha ativa da rede de transporte de Curitiba. Ao selecionar uma linha o seu itinerário é destacado no mapa, e todos os pontos de ônibus que fazem parte da linha são apresentados no mapa. Assim sendo possível visualizar facilmente veículos que estão fora da rota.

Ao selecionar uma linha também é possível visualizar as informações de cada ponto de ônibus individualmente, como informações como endereço do ponto, sentido e tipo do ponto.

Também é possível visualizar as informações dos veículos em operação, obtendo informações como linha, código do veículo, quantidade de pessoas no veículo, sentido, situação e última alteração.

Por último o *Dashboard*, possui uma série de indicadores e relatórios para ajudar no controle da frota, identificando gargalos. Os indicadores disponíveis são: quantidade de veículos em operação por linha; veículos adiantados de uma linha; veículos atrasados; veículos no horário; quantidade de pessoas por veículo; e quantidade de pessoas por linha.

Quanto ao aplicativo móvel, sua principal função é enviar ao servidor a localização do usuário para que seja determinado se ele está em uma linha. Para incentivar a utilização do aplicativo, foram desenvolvidas funcionalidades como visualização de linhas de ônibus, favoritar linhas, horário de ônibus de uma linha, visualização do itinerário de uma linha, visualização dos veículos em operação, informações do ponto de ônibus e informações do veículo.

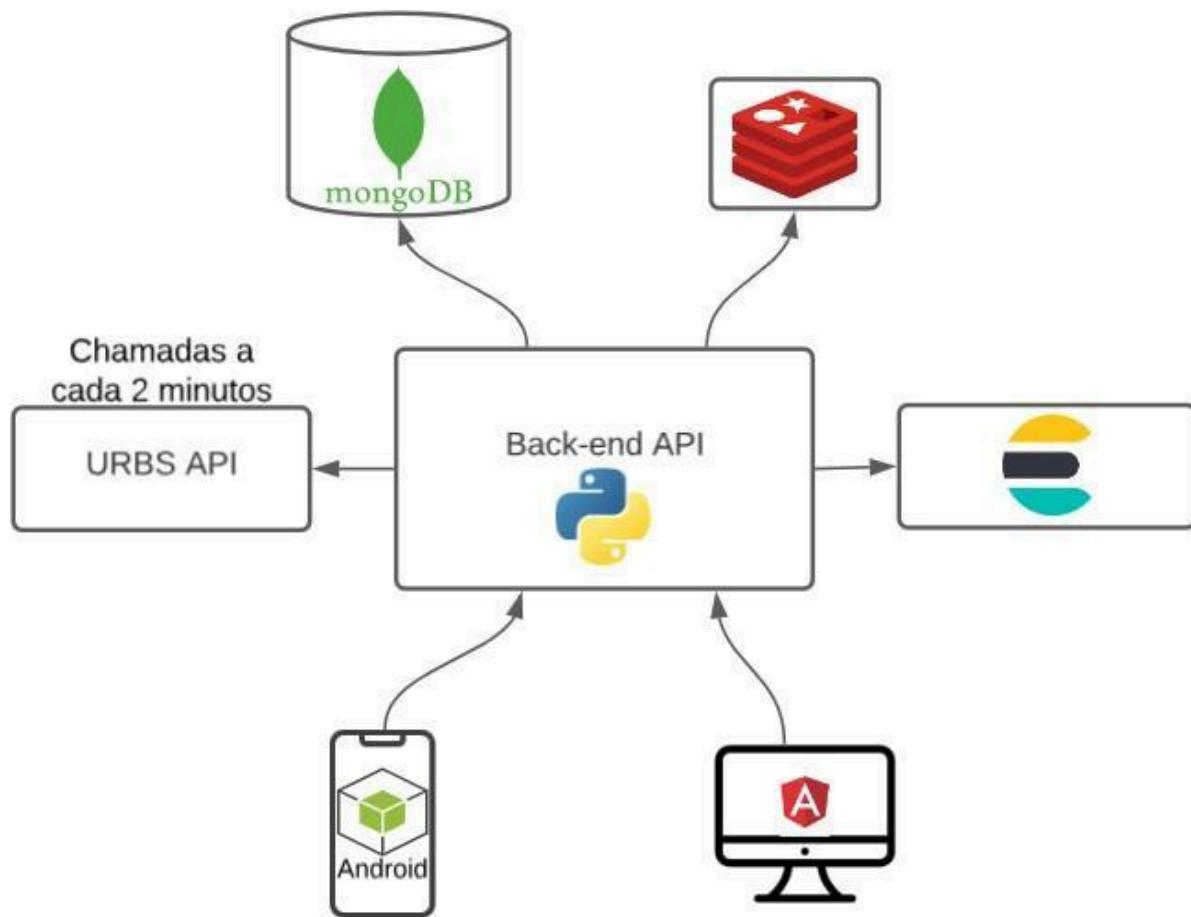
4.1 ARQUITETURA DO SISTEMA

A arquitetura do sistema foi dividida em três camadas distintas conforme apresentado na Figura 11, sendo que cada uma é responsável por um aspecto específico do sistema. A primeira camada é a camada de apresentação, composta por um aplicativo em Android e um *dashboard* em Angular. A segunda camada é a camada de integração, responsável pela conexão aos bancos de dados e pela lógica de negócios, como o cálculo da lotação dos ônibus, chamada para a API da URBS e correção do *shape* das linhas, essa camada foi desenvolvida em *Python*, que possui bibliotecas que ajudam com georreferenciamento e geoposição, além disso foi usado o *Elasticsearch*, para comparar se um usuário está em um veículo.

Por fim, a terceira camada é o acesso aos dados, que utiliza o MongoDB, um cache em Redis.

O sistema também utiliza alguns protocolos de interação entre camadas. A conexão da camada de apresentação com a de integração é feita através de uma arquitetura RESTful que favorece a fácil compatibilidade entre os sistemas. Já a conexão da camada de integração para o acesso a dados é realizada com protocolos como o TCP/IP, utilizado pelo MongoDB, RESP (REdis Serialization Protocol) utilizado pelo Redis, e REST que é utilizado para as chamadas para a API pública da URBS.

FIGURA 11 - ARQUITETURA DO SISTEMA



FONTE: OS AUTORES (2024)

Para saber se um usuário do aplicativo está em um ônibus ou não, foi utilizado o *Elasticsearch*. Com o auxílio do *Elastic*, foi comparado se as coordenadas de um usuário estão em um shape, que basicamente é um série de centenas ou milhares coordenadas que definem o itinerário de uma linha, caso as coordenadas estejam dentro de um shape começa-se a comparar sua localização com a dos veículos da linha que possui aquele shape, como somente pode-se obter a localização dos veículos a cada dois minutos pelo API da URBS, se em mais de duas chamadas as posições coincidem, então significa que o usuário está naquele ônibus.

Alguns *shapes* tinham pontos que desviam muito do trajeto dos demais, as vezes um ponto no meio do shape tinha uma coordenada quilômetros de distância dos demais, como os pontos de um *shape* são sempre sequenciais consideramos isso um erro, como existem centenas de linhas de ônibus na cidade, e o *shape* de cada uma dessas linhas pode possuir até milhares de pontos, fazer a correção manualmente era inviável, então decidimos usar um método chamado *Three Sigma Rule*, essa regra fala que se um ponto está fora de três desvios padrões, indica um valor atípico. Então para cada um dos shapes foi calculado a média da distância entre os pontos vizinhos, e com isso foi calculado o desvio padrão, depois aplicando a *Three Sigma Rule*, caso a distância entre um ponto e seus vizinhos era maior que três vezes o desvio padrão, ele era desconsiderado do *shape*. Depois que aplicada, a rota de cada uma das linhas da URBS foram corrigidas.

Para determinar se um usuário está em um ônibus, o sistema utiliza um método de clusterização. Primeiramente, as coordenadas de latitude e longitude dos usuários são convertidas para radianos. Em seguida, o algoritmo DBSCAN (*Density-Based Spatial Clustering of Applications with Noise*), que é um algoritmo que identifica grupos de pontos densos separados por regiões de baixa densidade, é aplicado para agrupar as coordenadas em clusters.

Após a clusterização, o sistema obtém um conjunto de *clusters*, onde cada *cluster* representa um grupo de usuários que estão próximos uns dos outros. Em seguida, o sistema calcula a média dos *timestamps* das coordenadas em cada *cluster*, que representa o tempo médio em que os usuários estavam naquela localização. Com base nos *clusters* e nos *timestamps* médios, o sistema pode determinar quais usuários estavam em um ônibus em um determinado momento. Para isso, o sistema compara as coordenadas dos *clusters* com as coordenadas dos ônibus em um determinado intervalo de tempo. Se as coordenadas de um *cluster* estiverem próximas das coordenadas de um ônibus em um determinado momento, o sistema infere que os usuários naquele cluster estavam naquele ônibus.

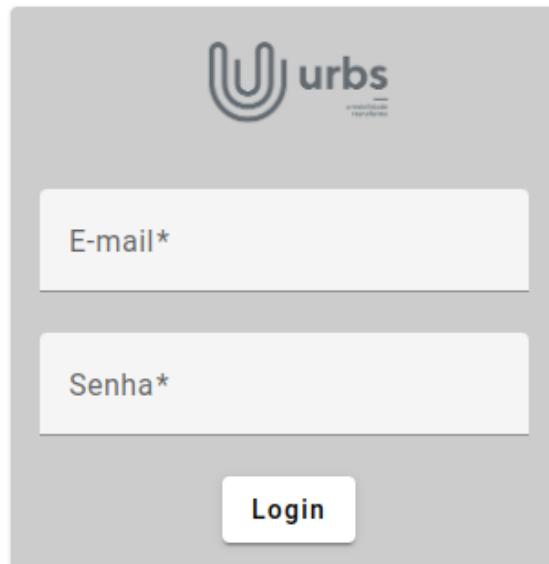
4.2 APRESENTAÇÃO DO SISTEMA

Esta seção tem como objetivo apresentar o sistema resultante deste projeto, descrevendo suas funcionalidades e interação com o usuário na camada da visão. Nessa primeira seção serão apresentadas as telas e funcionalidades do Dashboard, de acesso administrativo, na seção seguinte será apresentado o aplicativo.

4.2.1 Login

Como mostra na Figura 12, a primeira página que é apresentada no sistema é a tela de login, na qual o usuário deve inserir um e-mail e senha pré cadastrados. Como o acesso ao *Dashboard* foi feito somente para administradores terem acesso, não existe opção de cadastro nesta página.

FIGURA 12 - LOGIN



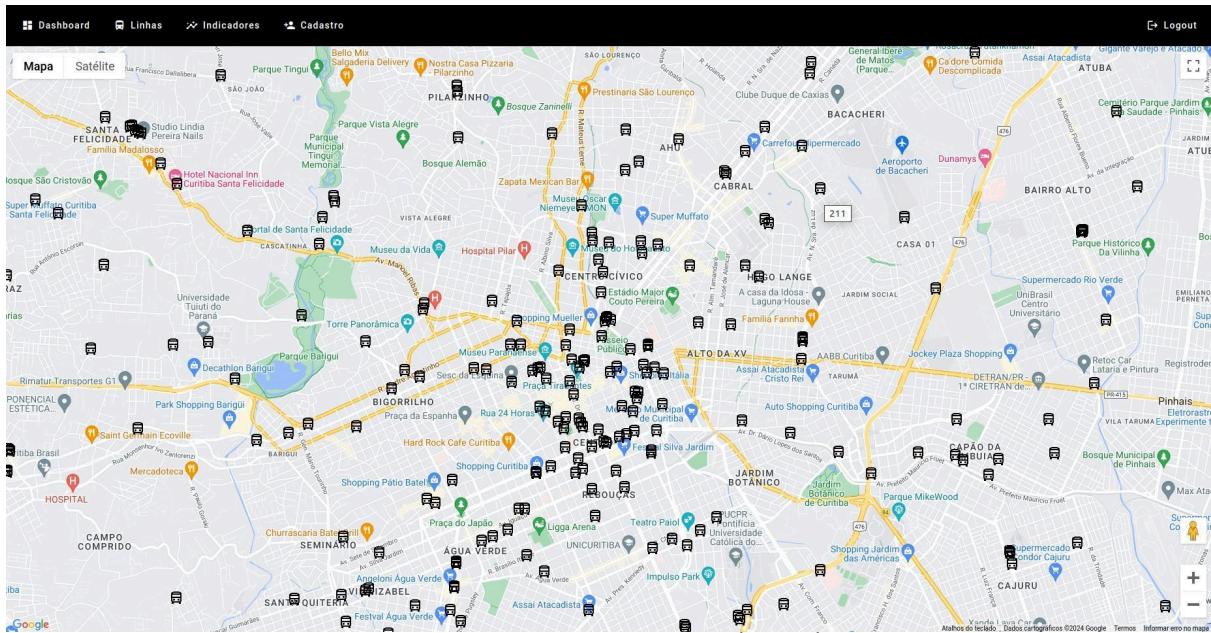
FONTE: OS AUTORES (2024)

4.2.2 Página inicial Dashboard

Após realizar o login o usuário é redirecionado para o página inicial do Dashboard, como apresentado na Figura 13. Aqui ele tem a visualização do mapa e de todas as linhas em operação, assim como filtros disponíveis para melhor interação.

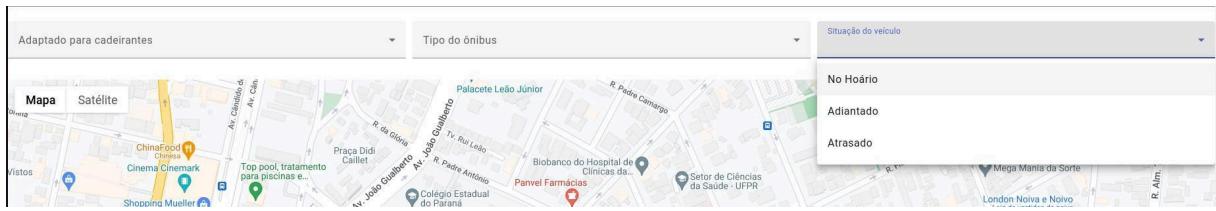
Para melhor visualização, também é possível filtrar os veículos sendo apresentados na tela, sendo estes: Veículos adaptados; Tipo do ônibus (Comum, Articulado, Biarticulado, Micro, Micro especial, Biarticulado biodiesel, Articulado Biodiesel, Híbrido e Elétrico); Situação do veículo (No horário, Adiantado ou Atrasado). Conforme mostra a Figura 14.

FIGURA 13 - DASHBOARD



FONTE: OS AUTORES (2024)

FIGURA 14 - FILTROS DASHBOARD



FONTE: OS AUTORES (2024)

4.2.3 Linhas

Conforme mostra a Figura 15, nesta página o usuário tem a visualização de todas as linhas operantes do transporte público coletivo da cidade, podendo pesquisar a linha digitando o código na linha ou nome na barra de pesquisa.

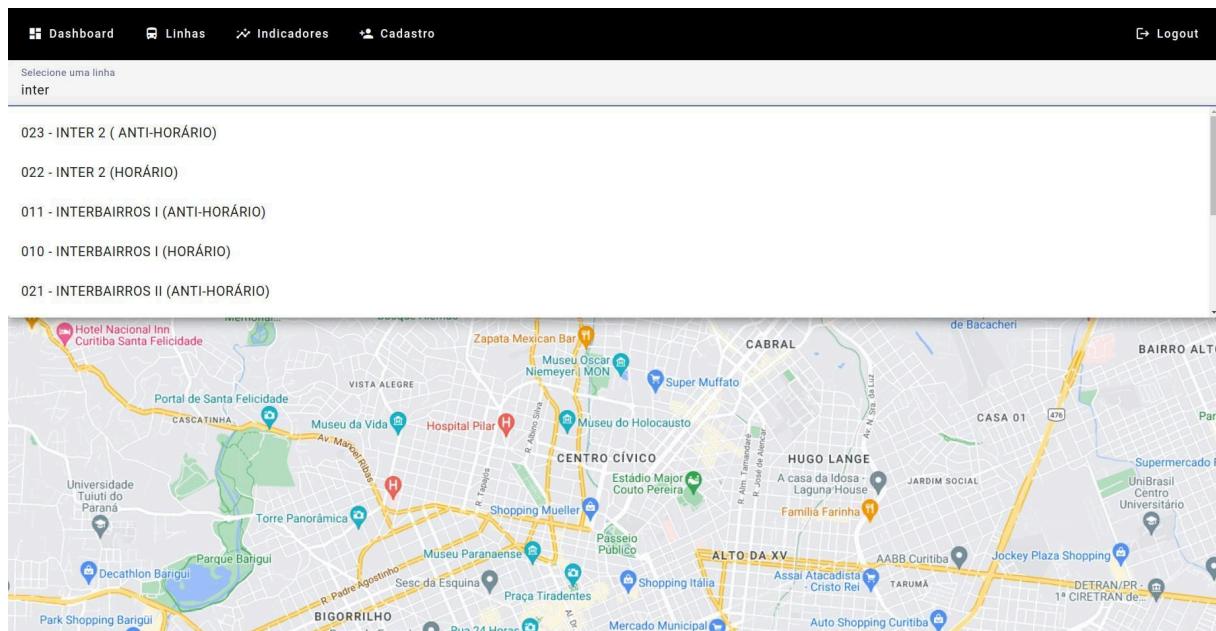
Ao selecionar uma linha são destacados no mapa o seu itinerário conforme mostrado na Figura 16, todos os pontos de ônibus que fazem parte dessa linha e os veículos em operação neste momento.

Clicando em um veículo são apresentadas as suas informações, como Código da linha que está operando, código do veículo, sentido, situação atualizada, tipo do veículo e horário da última atualização, conforme mostrado na Figura 17.

Ao clicar em um ponto de ônibus são apresentadas as informações relevantes daquele ponto, como o endereço em que está localizado, tipo do ponto e sentido conforme a linha que foi selecionada, como apresentada na Figura 18.

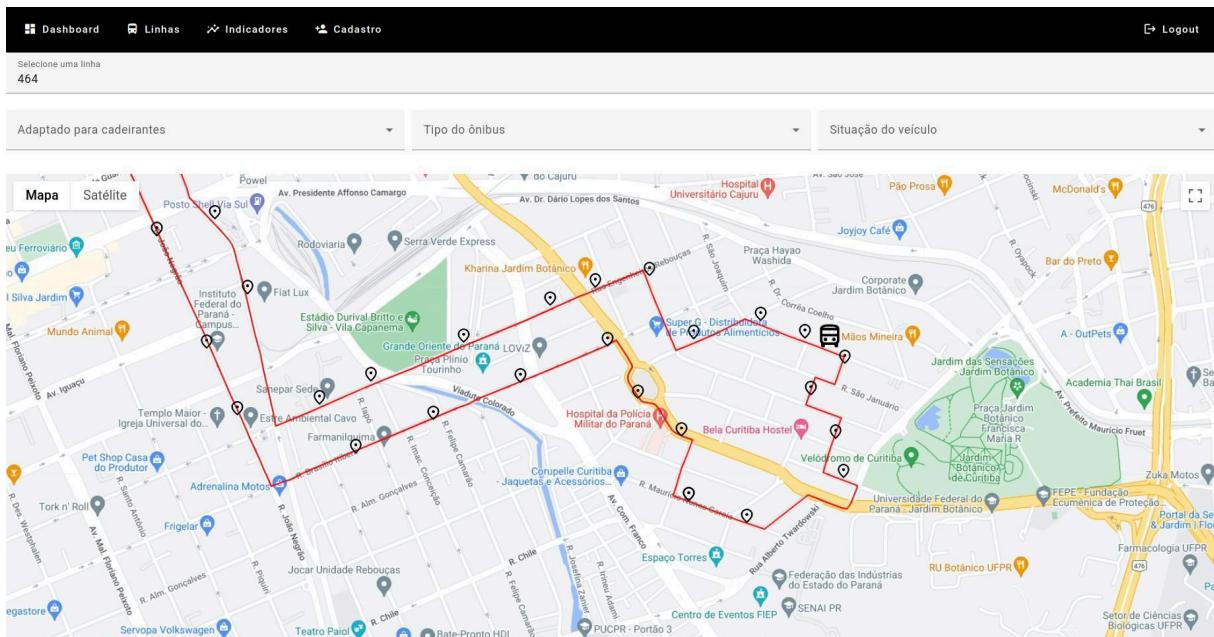
Conforme apresentado na Figura 19, essa página também possibilita filtrar os veículos sendo apresentados, sendo eles: Veículos adaptados; Tipo do ônibus (Comum, Articulado, Biarticulado, Micro, Micro especial, Biarticulado biodiesel, Articulado Biodiesel, Híbrido e Elétrico); Situação do veículo (No horário, Adiantado ou Atrasado)

FIGURA 15 - LINHAS



FONTE: OS AUTORES (2024)

FIGURA 16 - LINHA SELECIONADA



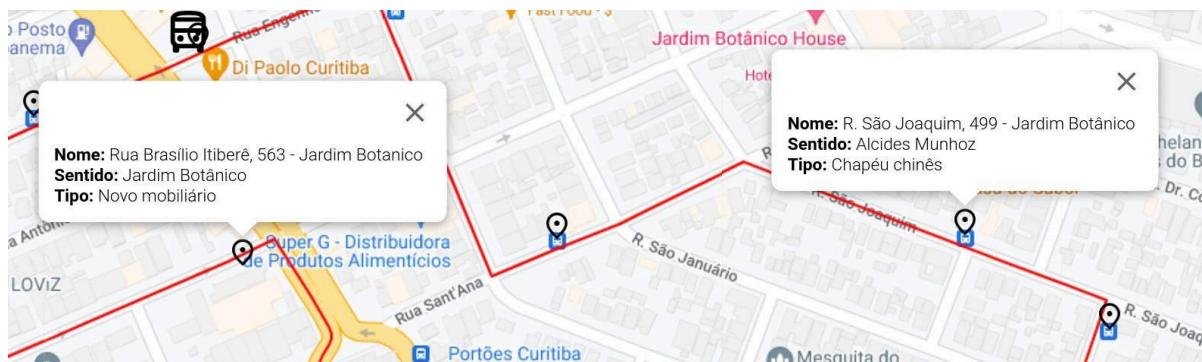
FONTE: OS AUTORES (2024)

FIGURA 17 - INFORMAÇÕES DO VEÍCULO



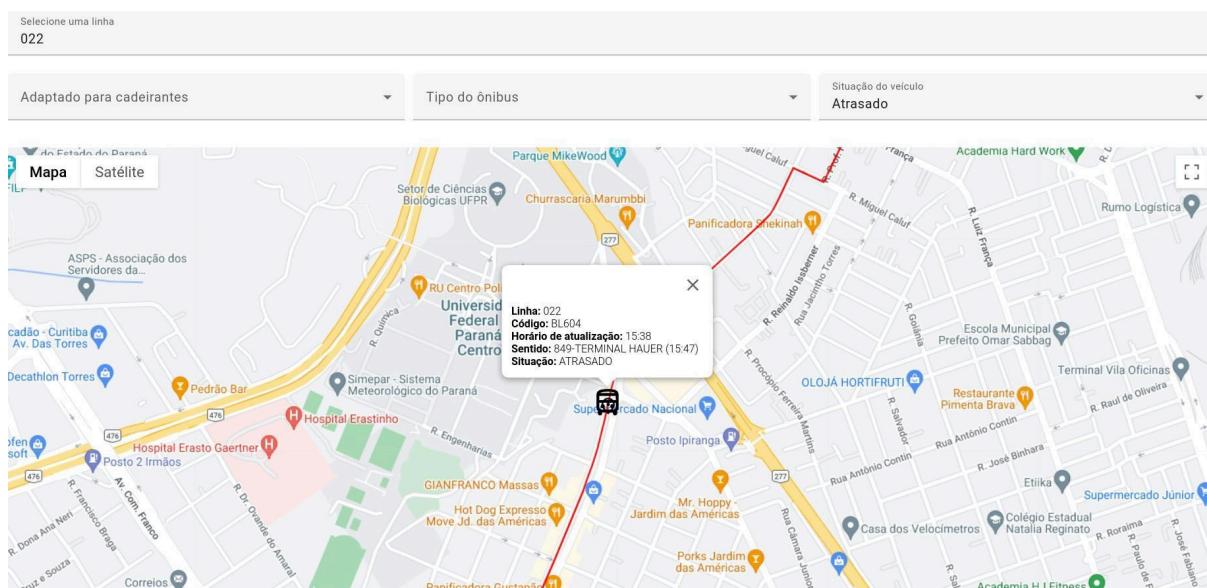
FONTE: OS AUTORES (2024)

FIGURA 18 - DETALHE DO PONTO



FONTE: OS AUTORES (2024)

FIGURA 19 - FILTROS DE VEÍCULOS



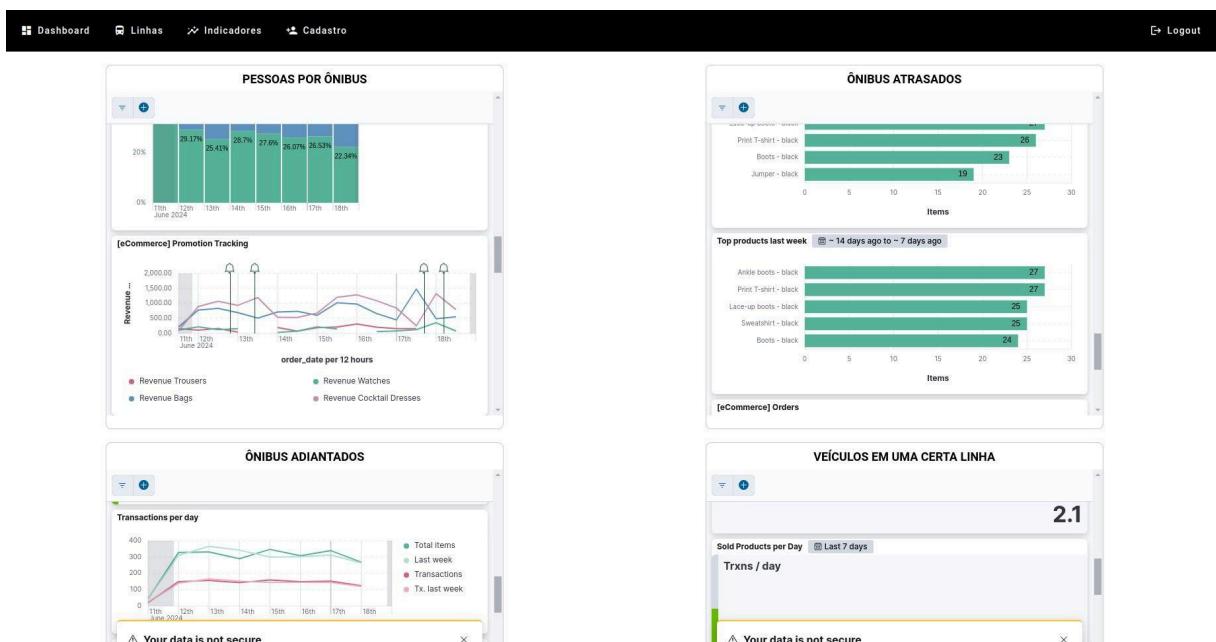
FONTE: OS AUTORES (2024)

4.2.4 Indicadores

No menu “Indicadores” estão disponíveis relatórios e indicadores, sendo possível obter informações como: Quantidade de veículos em operação por linha (média por hora), Veículos adiantados de uma linha (Quantidade por hora), Veículos atrasados de uma linha (Quantidade por hora), Veículos no horário de uma linha (Quantidade por hora), Quantidade de pessoas por veículo (média de cada linha por hora), Quantidade de pessoas por linha (hora, dia, semana, mês). Conforme apresentado na Figura 20.

Também é possível realizar a exportação desses indicadores em formato PDF, caso o usuário ache necessário.

FIGURA 20 - INDICADORES



FONTE: OS AUTORES (2024)

4.2.5 Cadastro de administrador

Como o Dashboard foi feito pensando que o acesso será feito apenas por administradores e funcionários da URBS, o cadastro de um novo usuário para o *dashboard* pode ser feito somente por um usuário já cadastrado. Os usuários cadastrados no Dashboard podem acessar o aplicativo com suas credenciais, porém um usuário cadastrado no aplicativo não consegue acessar o Dashboard.

A tela de cadastro de usuário é mostrada na Figura 21. Nela, o usuário deve inserir o nome, CPF, e-mail, data de nascimento, telefone e uma senha para acesso. O sistema faz a validação do CPF digitado, e caso seja válido, e não exista nenhum usuário cadastrado com o mesmo e-mail ou CPF, será apresentada uma mensagem de usuário cadastrado com sucesso, como mostra a Figura 22. Caso contrário será

apresentada uma tela informando o erro que ocorreu na hora do cadastro do novo usuário.

FIGURA 21 - CADASTRO DE ADMINISTRADOR

Nome*

CPF*

Data de Nascimento*
dd/mm/aaaa

Telefone*

E-mail*

Senha*

Cadastrar

FONTE: OS AUTORES (2024)

FIGURA 22 - ADMINISTRADOR CADASTRADO

Nome*
Guilherme

CPF*
84410730070

Data de Nascimento*
10/10/2000

Telefone*
(41) 99999-9999

E-mail*
guilherme@ufpr.br

Senha*

Sucesso

Usuário cadastrado com sucesso!

Fechar

Cadastrar

FONTE: OS AUTORES (2024)

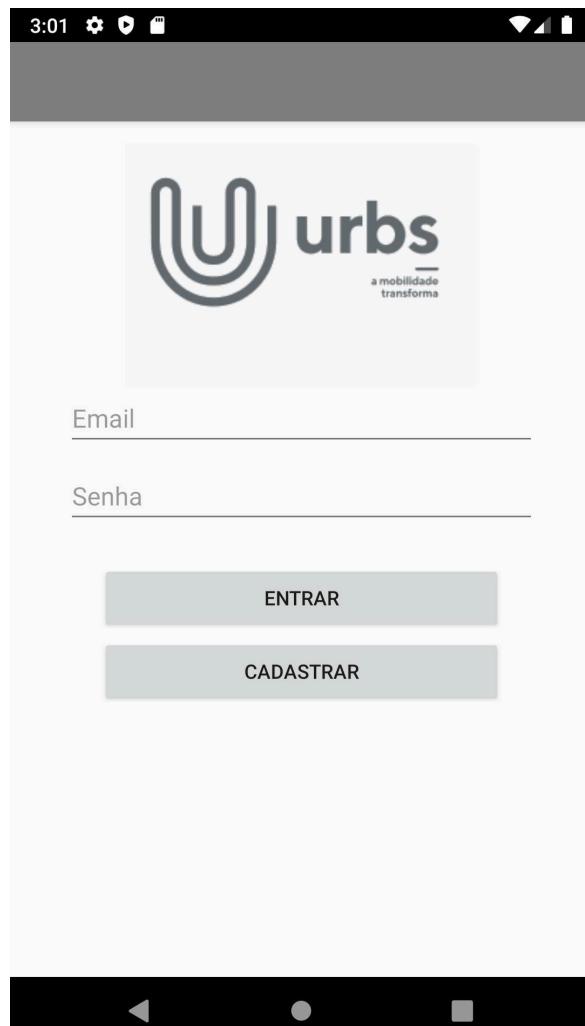
4.3 APLICATIVO

Nessa seção será apresentada a utilização do aplicativo móvel, que será utilizado pelos usuários da rede pública de transporte de Curitiba.

4.3.1 Login

Conforme mostra a Figura 23, a primeira interação com o usuário é feita na tela de login, na qual deve-se inserir e-mail e senha previamente cadastrados para acessar o sistema. Além disso, para usuários não cadastrados, é apresentada a opção de realização de cadastro, presente na seção seguinte.

FIGURA 23 - LOGIN APLICATIVO



FONTE: OS AUTORES (2024)

4.3.2 Cadastro

Na tela de cadastro devem ser inseridas as informações pessoais do usuário, conforme apresentado na Figura 24, sendo elas: Nome, Email, Data de Nascimento, Telefone, CPF e Senha. O sistema faz a validação dos dados preenchidos, e caso todos estejam corretos, o novo usuário é cadastrado.

FIGURA 24 - CADASTRO USUÁRIO



FONTE: OS AUTORES (2024)

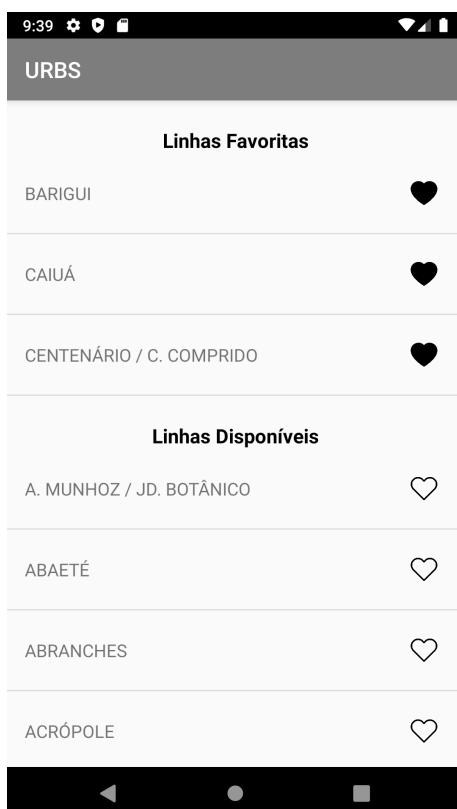
4.3.3 Página inicial

Conforme mostra a Figura 25, após realizar o login o usuário é redirecionado para a página de seleção de linhas, na qual todas as linhas de ônibus de Curitiba são apresentadas. O usuário pode selecionar uma linha para obter suas informações.

Nessa página também é possível favoritar as linhas mais utilizadas, dessa maneira ficando em destaque no topo da página, conforme apresentado na Figura 26.

FIGURA 25 - LISTA DE LINHAS

FONTE: OS AUTORES (2024)

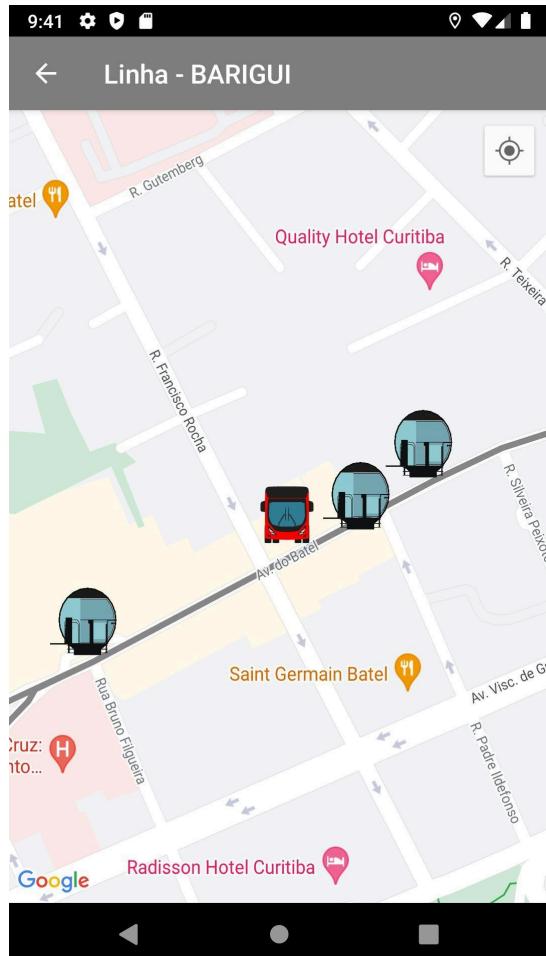
FIGURA 26 - LINHAS FAVORITAS

FONTE: OS AUTORES (2024)

4.3.4 Seleção de linha

Conforme mostra a Figura 27, ao selecionar uma linha da lista, o seu itinerário é destacado no mapa, e são apresentados todos os veículos em operação, assim como todos os pontos pertencentes a essa linha.

FIGURA 27 - LINHAS SELECIONADA



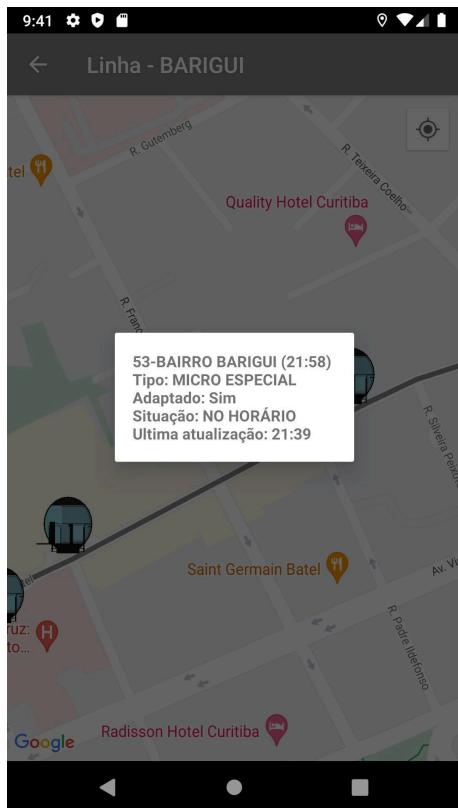
FONTE: OS AUTORES (2024)

Clicando em um veículo, são apresentadas suas informações, como tipo de ônibus, sentido, situação, se é adaptado, e o horário que ele vai chegar no próximo ponto, como apresentado na Figura 28.

Conforme mostra a Figura 29, ao clicar em um ponto de ônibus, serão apresentadas as informações daquele ponto, como nome da rua, as linhas que passam por aquele ponto e o sentido do ponto.

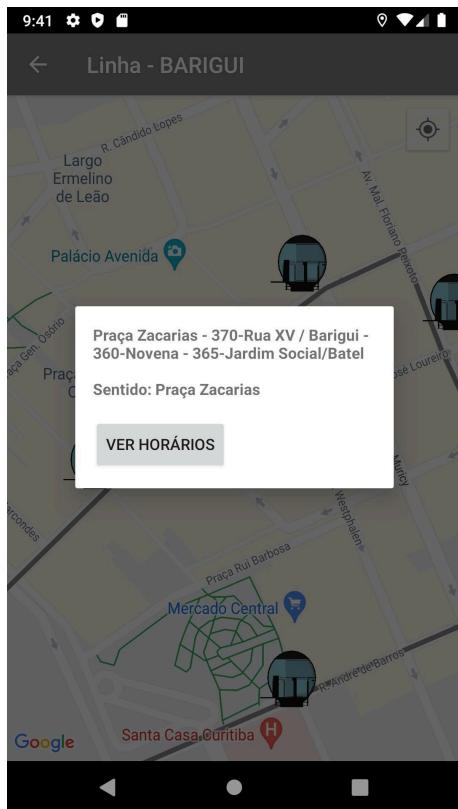
Ao clicar no botão ver horário, o aplicativo apresenta a tabela de horários daquele ponto para a linha que foi selecionada. Pontos de ônibus, como estações tubo, terminais, praças e ruas principais possuem tabelas de horários, conforme mostra a Figura 30.

FIGURA 28 - DETALHES DO VEÍCULO



FONTE: OS AUTORES (2024)

FIGURA 29 - DETALHE DO PONTO



FONTE: OS AUTORES (2024)

FIGURA 30 - TABELA DE HORÁRIO

The screenshot shows a mobile application interface titled "URBS - Horários". The screen displays a table of time intervals (06:31 to 23:57) across various days (Dias Úteis, Saturday, Sunday, and Holidays). The table is organized into columns for each hour from 06:31 to 23:57. The data is as follows:

	06:31	07:02	07:37	08:10	08:42	09:15	10:17	11:19
Dias Úteis	06:31	07:02	07:37	08:10	08:42	09:15	10:17	11:19
	12:22	13:26	14:30	15:35	16:08	16:43	17:18	18:02
	18:47	19:31	20:32	21:30	22:28	23:25	00:22	06:30
Sábado	06:31	07:02	07:37	08:10	08:42	09:15	10:17	11:19
	12:22	13:26	14:30	15:35	16:08	16:43	17:18	18:02
	18:47	19:31	20:32	21:30	22:28	23:25	00:22	06:30
	07:23	08:16	09:09	10:02	10:55	11:47	12:39	13:32
	14:25	15:18	16:11	17:04	17:57	18:50	19:43	20:36
	21:27	22:17	23:07	23:57	06:27	07:17	08:07	08:57
Domingo	06:31	07:02	07:37	08:10	08:42	09:15	10:17	11:19
	12:22	13:26	14:30	15:35	16:08	16:43	17:18	18:02
	18:47	19:31	20:32	21:30	22:28	23:25	00:22	06:30
	07:23	08:16	09:09	10:02	10:55	11:47	12:39	13:32
	14:25	15:18	16:11	17:04	17:57	18:50	19:43	20:36
	21:27	22:17	23:07	23:57	06:27	07:17	08:07	08:57
	09:47	10:37	11:27	12:17	13:07	13:57	14:47	15:37
	16:27	17:17	18:07	18:57	19:47	20:37	21:27	22:17
	23:07	23:57						
Feriados	06:31	07:02	07:37	08:10	08:42	09:15	10:17	11:19
	12:22	13:26	14:30	15:35	16:08	16:43	17:18	18:02
	18:47	19:31	20:32	21:30	22:28	23:25	00:22	06:30
	07:23	08:16	09:09	10:02	10:55	11:47	12:39	13:32
	14:25	15:18	16:11	17:04	17:57	18:50	19:43	20:36
	21:27	22:17	23:07	23:57	06:27	07:17	08:07	08:57
	09:47	10:37	11:27	12:17	13:07	13:57	14:47	15:37
	16:27	17:17	18:07	18:57	19:47	20:37	21:27	22:17
	23:07	23:57						

FONTE: OS AUTORES (2024)

Este capítulo apresentou o funcionamento do sistema desenvolvido, tanto do *dashboard* como do aplicativo, assim o detalhamento da arquitetura do sistema. Na Próxima seção será apresentado as considerações finais da equipe, assim como ideias de futuras implementações.

5. CONSIDERAÇÕES FINAIS

A fim de aprimorar os serviços prestados à população, a URBS identificou a necessidade de coletar dados a respeito da utilização e demanda dos usuários do transporte público de Curitiba, sendo estes dados a lotação dos ônibus, horários de pico e a movimentação nesses horários.

A partir deste cenário, foi definido como objetivo deste TCC a criação de uma aplicação que possui um *dashboard* e um aplicativo móvel. No *dashboard*, apresenta a posição do ônibus, a quantidade de pessoas em um determinado ônibus e as informações específicas de uma linha. Também estão disponíveis relatórios e indicadores de controle que facilitam o planejamento estratégico da URBS. O aplicativo oferece funcionalidades de visualização de linhas pelo usuário, com os pontos pertencentes a linha, registro de linhas favoritas, itinerário e visualização dos ônibus circulando em tempo real, com informações como a situação do veículo e uma estimativa de tempo para chegar em determinados pontos. Todas essas funcionalidades são para incentivar o usuário a usar o aplicativo, para que com sua localização possa se determinar se está ou não em um ônibus.

Durante a etapa inicial do trabalho, foi possível definir conceitualmente: as funcionalidades e estrutura do aplicativo; produzir a especificação do software; estabelecer as tecnologias de desenvolvimento do programa; e o modo de produção fazendo uso do modelo incremental e o da metodologia Scrum. Já a segunda fase do projeto aconteceu por meio de entregas incrementais, e ao final desse ciclo de desenvolvimento, o sistema e sua arquitetura foram descritos e apresentados no capítulo quatro deste documento.

Embora o projeto tenha atingido seu objetivo, a solução final possui limitações. A primeira delas é proveniente da API da URBS, pois as requisições não podem ser feitas com um intervalo menor do que dois minutos, o que pode acarretar uma margem de erro. A diminuição desse tempo de resposta melhoraria significativamente o algoritmo desenvolvido, podendo realizar comparações da coordenada dos usuários com a frota da URBS.

Como implementação futura, sugere-se uma funcionalidade de recarga do cartão transporte por meio do aplicativo, como realizado pelo Metrocard da COMEC. Isso potencializaria a utilização do aplicativo entre os usuários, permitindo aprimorar a funcionalidade de identificação de lotação dos ônibus.

REFERÊNCIAS

ANDRADE, J. N.; GALVÃO, D. C.; **O conceito de Smart Cities Aliado à Mobilidade Urbana.** 2016.

ANGULAR. **Angular.** Disponível em: <angular.io/guide/what-is-angular>. Acessado em: Janeiro 2023.

CLOUDFLARE. **cloudflare.** Disponível em: <<https://www.cloudflare.com/dns/>>. Acessado em: Junho 2024

CONCEIÇÃO, H. M. R. S. d. **Plataforma de gestão de importação.** In: **Plataforma de Gestão de Importação.** [S.l.: s.n.], 2015.

CURITIBA. **Curta Curitiba Pedalando: mais de 200 km de ciclovias para explorar.** Disponível em:
<<https://www.curitiba.pr.gov.br/noticias/curta-curitiba-pedalando-mais-de-200-km-de-ciclovias-para-explorar/64052>>. Acessado em: Fevereiro 2023.

DETRAN PR. **Dimensionamento da Frota, Dezembro 2022.** Disponível em:
<https://www.detran.pr.gov.br/sites/default/arquivos_restritos/files/documento/2023-01/frota_dezembro_de_2022_c.pdf>. Acessado em: Fevereiro 2023.

DETRAN PR. **Estatísticas de Trânsito.** Disponível em:
<<https://www.detran.pr.gov.br/Pagina/Estatisticas-de-transito>>. Acessado em: Fevereiro 2023.

DETRAN PR. **Ocorrências entre 2018 e 2022.** Disponível em:
<https://www.detran.pr.gov.br/sites/default/arquivos_restritos/files/documento/2023-01/ocorrencias_2018_a_2022.pdf>. Acessando em: Fevereiro 2023.

DOCKER. **docker.** Disponível em: <<https://docs.docker.com>>. Acessado em: Junho 2024

ELASTICSEARCH. **elasticSearch.** Disponível em:
<<https://www.elastic.co/elasticsearch>>. Acessado em: Junho 2024

FASTAPI. **fastAPI.** Disponível em: <<https://fastapi.tiangolo.com>>. Acessado em: Junho 2024

FLUTTER. **Flutter.** Disponível em: <<https://docs.flutter.dev/resources/faq>>. Acessado em: Janeiro 2023.

FOWLER, M. **UML Essencial: um breve guia para a linguagem-padrão de modelagem de objetos.** 3. ed. Porto Alegre: Bookman, 2005.

GEOPY. **geopy.** Disponível em: <<https://geopy.readthedocs.io>>. Acessado em: Junho 2024

GOVERNO DO PARANÁ. **Região metropolitana de Curitiba.** Disponível em:
<<https://www.amep.pr.gov.br/Pagina/Regiao-Metropolitana-de-Curitiba>> Acessado em: Fevereiro 2023

JAVA. **java.** Disponível em: <<https://www.oracle.com/java/>>. Acessado em: Junho

2024

KON, F.; SANTANA, E. F. Z. **Computação aplicada a Cidades Inteligentes: Como dados, serviços e aplicações podem melhorar a qualidade de vida nas cidades.** Capítulo 4, 2017.

LARMAN, G. **Utilizando UML e Padrões**. 3. Edição. Bookman, 2007. MAZUTTI, J. H.; Gestão de Obras. 1. Edição. Porto Alegre: SAGAH, 2018.

MINISTÉRIO DAS CIDADES. **Cartilha Lei 12.587 - Política Nacional da Mobilidade Urbana. República Federativa do Brasil**, 2013.

MONGODB. **mongoDB**. Disponível em: <<https://www.mongodb.com>>. Acessado em: Junho 2024

NODEJS. **nodeJS**. Disponível em: <nodejs.org/en/about/>. Acessado em: Janeiro 2023.

PANDAS. **pandas**. Disponível em: <<https://pandas.pydata.org>>. Acessado em: Junho 2024

PORTAINER. **portainer**. Disponível em: <<https://www.portainer.io/documentation>>. Acessado em: Junho 2024

POSTGRESQL. **postgreSQL**. Disponível em: <postgresql.org/about/>. Acessado em: Janeiro 2023

PREFEITURA MUNICIPAL DE CURITIBA, **Plano Municipal de Mobilidade Urbana e Transporte Integrado**. Prefeitura Municipal de Curitiba, 2008.

PRESSMAN, R. S.; MAXIM, B. R. **Engenharia de software: uma abordagem profissional**. 8. ed. Porto Alegre: AMGH, 2016.

PYTHON. **python**. Disponível em: <<https://www.python.org>>. Acessado em: Junho 2024

REDIS. **Redis**. Disponível em: <redis.io/docs/about/>. Acessado em: Janeiro 2023.

SILVA, A. B. S.; SILVA, S. **Critérios na Qualidade em Serviços de Transporte Público Urbano: uma Contribuição Teórica**. 2018.

SMART CITY EXPO. **2013 Edition Smart City Expo Bogotá**. Disponível em: <<https://www.smartcityexpo.com/the-event/editions-abroad/>> . Acessado em: Fevereiro 2023

URBS. **Estatísticas do Transporte**. Disponível em: <<https://www.urbs.curitiba.pr.gov.br/transporte/estatisticas>>. Acessado em: Dezembro 2022.

URBS. **Táxis**. Disponível em: <<https://www.urbs.curitiba.pr.gov.br/transporte/taxis>> Acessado em: Fevereiro 2023.

URBS. **URBS - Versão Mobile**. Disponível em: <<https://www.urbs.curitiba.pr.gov.br/mobile>>. Acessado em: Fevereiro 2023.

VASCONCELLOS, E. A.; **Mobilidade urbana em Curitiba – os limites do sonho.** Revista dos Transportes Públicos - ANTP, 2019 - 1º quadrimestre, p. 7-24. Disponível em: <http://files.antp.org.br/2019/4/16/rtp151-2.pdf>. Acessado em: dez. 2022

VASCONCELOS, E. M. A.; FARIA, H. M.; **Transformações na mobilidade urbana em Curitiba sob a perspectiva metropolitana.** 2021.

WAZLAWICK, R. S.; **Engenharia de Software: Conceitos e Práticas.** 1. Edição. Rio de Janeiro: Elsevier Editora Ltda, 2013

APÊNDICE A - DIAGRAMA DE CASO DE USO

FIGURA 31 - DIAGRAMA CASO DE USO DASHBOARD

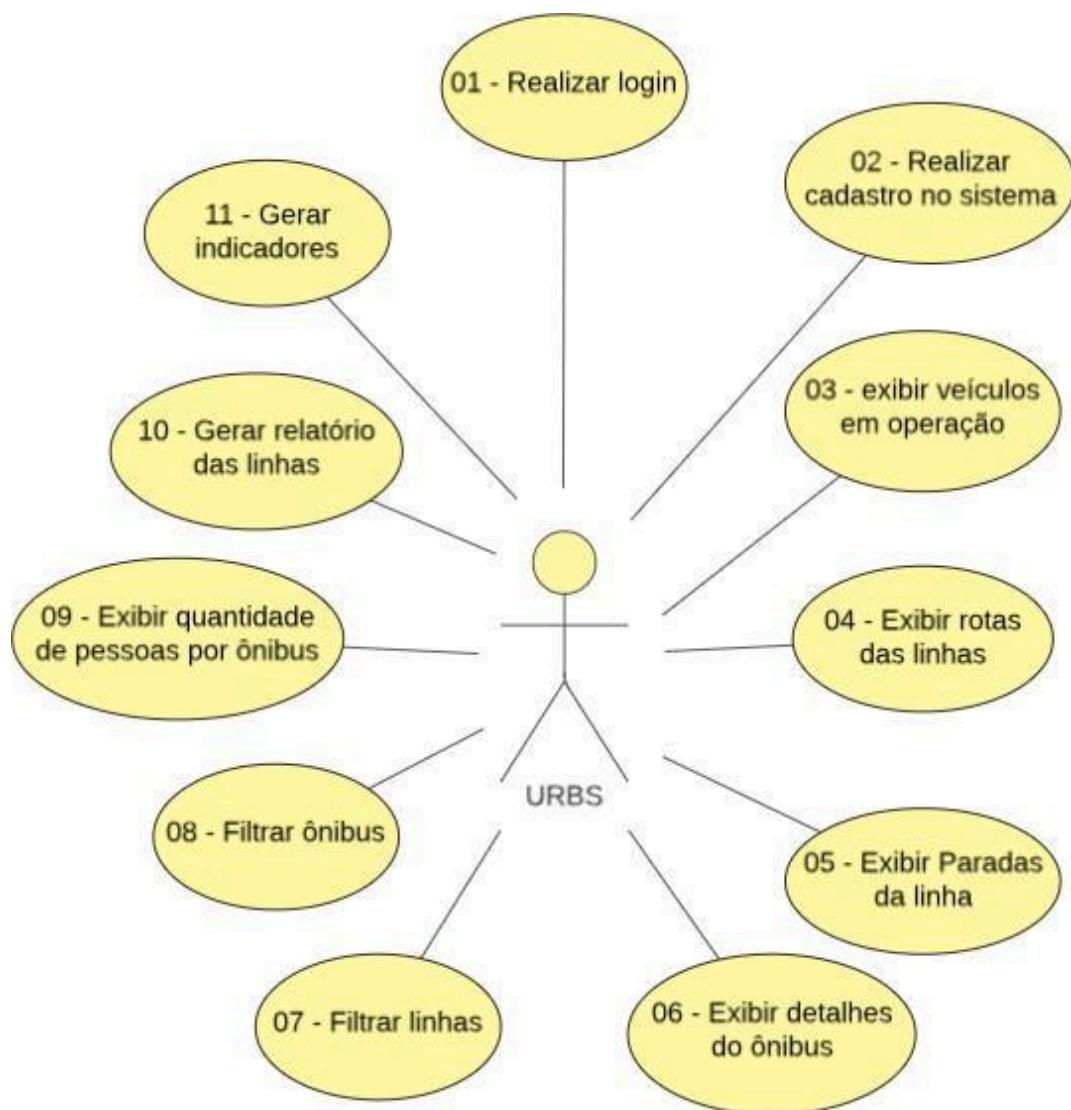
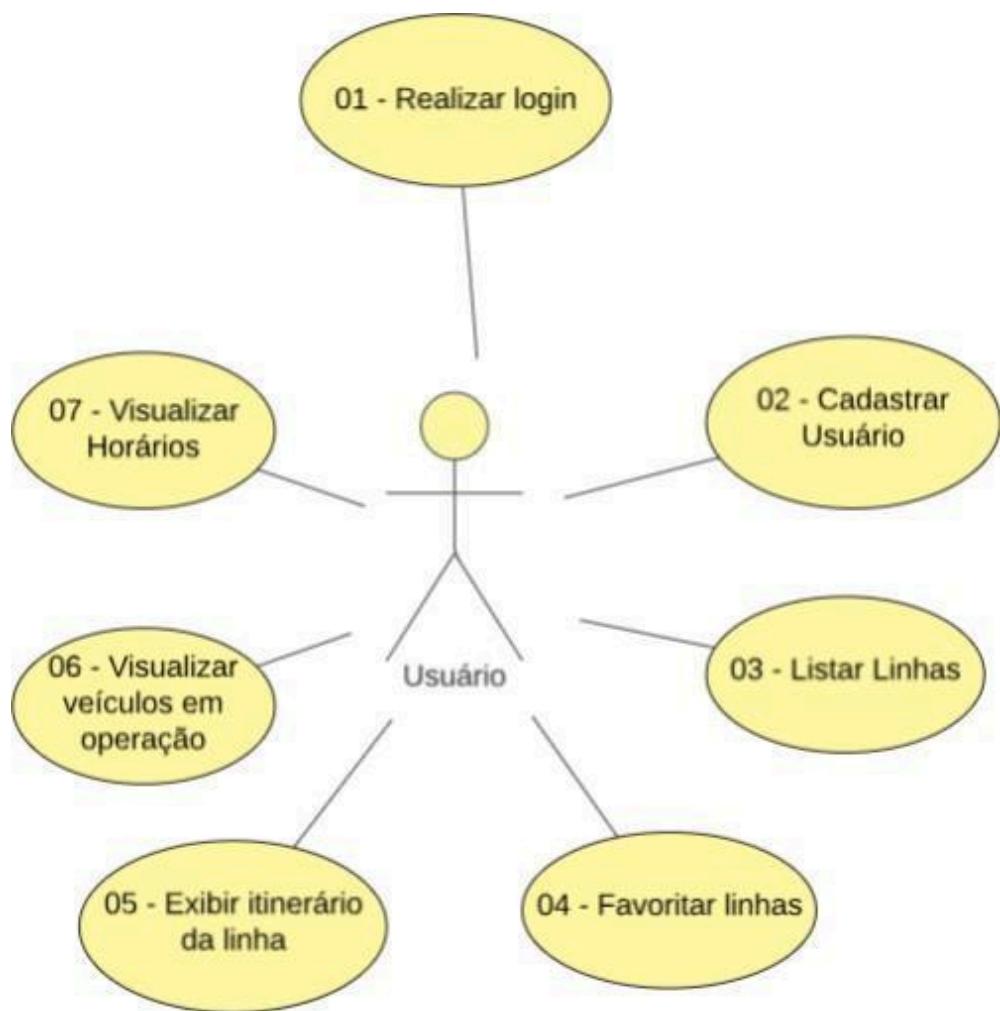


FIGURA 32 - DIAGRAMA CASO DE USO APLICATIVO



APÊNDICE B - HISTÓRIAS DE USUÁRIO

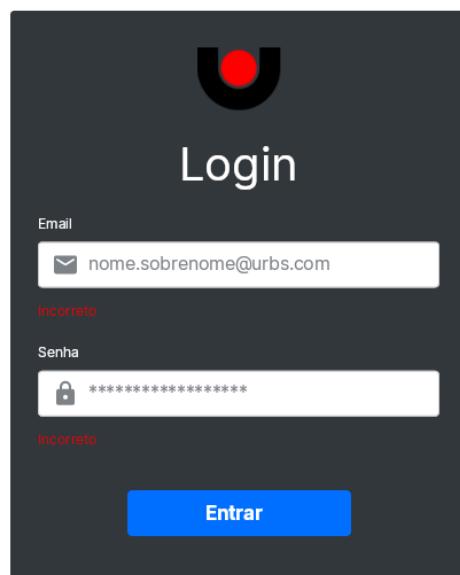
HU001 - Login

SENDO URBS

QUERO Efetuar Login

PARA Poder acessar o sistema

DESENHO DA TELA



CRITÉRIO DE ACEITAÇÃO

- 1) Deve logar o usuário

CRITÉRIO DE ACEITAÇÃO - DETALHAMENTO

Critério de contexto (Válido como premissa para todos os critérios):

Dado que Estou na página de login

1.1) Deve logar o usuário

Dado que Os dados não foram preenchidos

Quando Clicar no botão Login

Então	Mostrar que os dados não foram preenchidos
1.2)	
Dado que	A senha ou e-mail estão incorretos
Quando	Clicar no botão login
Então	Mostrar um aviso que os dados estão incorretos
1.3)	
Dado que	Todos os dados estão preenchidos corretamente
Quando	Clicar no botão de login
Então	Logar o usuário correspondente
E	Redirecionar para o Dashboard

HU002 - Realizar cadastro de administrador

Sendo URBS

Quero Cadastrar um novo administrador

Para Acessar o dashboard

DESENHO DA TELA

The screenshot shows a user registration form titled "Cadastro". The form fields are as follows:

- Primeiro Nome: Text input field.
- Último Nome: Text input field.
- Data de Nascimento: Text input field containing "2020/12/30".
- E-mail: Text input field.
- Confirmar E-mail: Text input field.
- Telefone: Text input field containing "(99) 99999-9999".
- Senha: Text input field.
- Confirmar Senha: Text input field.
- CPF: Text input field.

A blue "Confirmar" button is located at the bottom right of the form area.

CRITÉRIOS DE ACEITAÇÃO:

- 1) Deve cadastrar um administrador
- 2) O sistema deve validar o CPF
- 3) O sistema deve validar se já existe uma usuário cadastrado com o mesmo CPF

CRITÉRIOS DE ACEITAÇÃO – DETALHAMENTO:

Dado que estou na página de cadastro

1) Deve cadastrar o administrador

1.1)

Dado que algum campo está vazio

Quando	clicar no botão “Confirmar”
Então	mostrar aviso campos vazios
1.2) Dado	que e-mail inserido está invalido
Quando	clicar no botão “Confirmar”
Então	mostrar aviso de e-mail invalido
1.3) Dado	que senhas não correspondem
Quando	clicar no botão “Confirmar”
Então	mostrar aviso de senhas diferentes
1.4) Dado	que o telefone está invalido
Quando	clicar no botão “Confirmar”
Então	mostrar aviso de telefone invalido
1.5) Dado	que o cpf está invalido
Quando	clicar no botão “Confirmar”
Então	mostrar aviso de cpf invalido
1.6) Dado	que todos os campos estão corretamente preenchidos
Quando	clicar no botão “Confirmar”
Então	criar usuário no banco de dados
E	apresenta mensagem de administrador cadastrado

2) Deve redirecionar para a página inicial

Quando	Clicar no ícone da URBS
Então	Redirecionar para a página de login

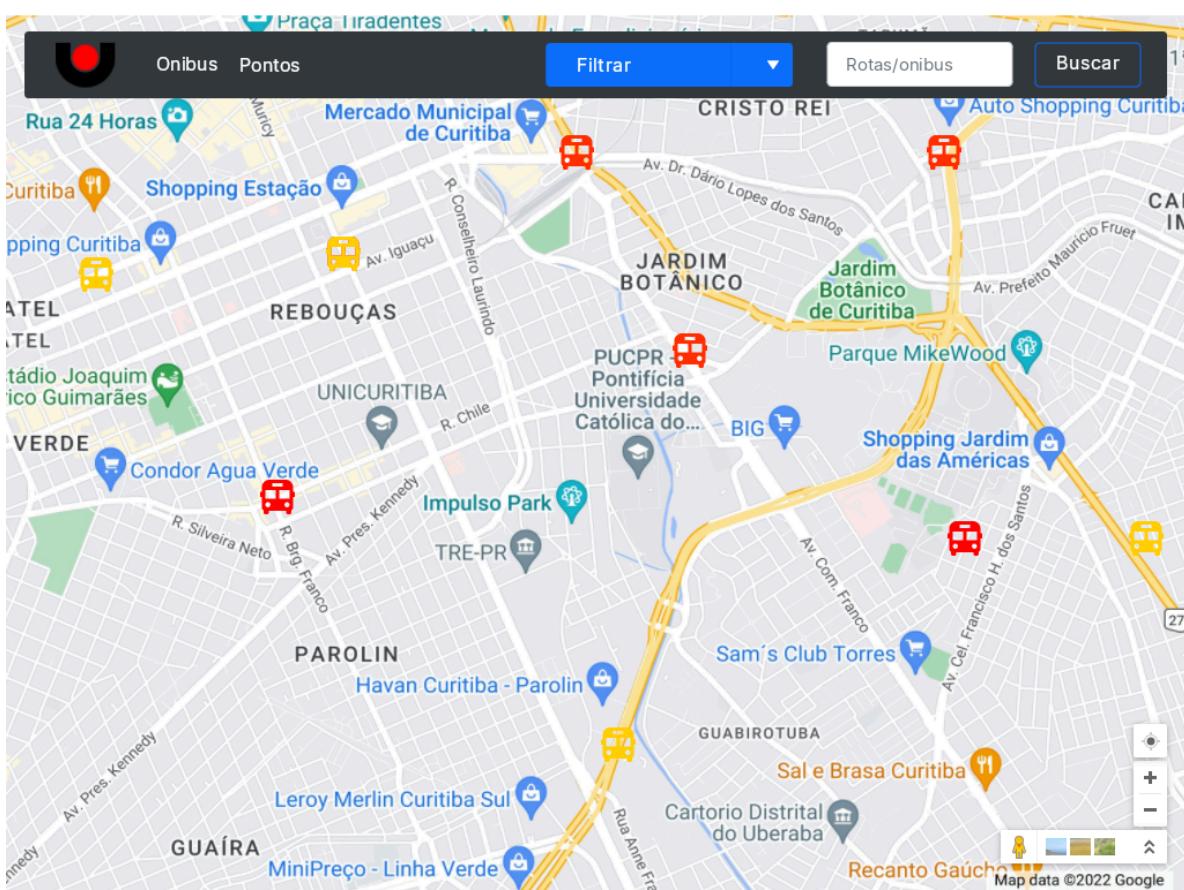
HU003 - Exibir veículos em operação

SENDO URBS

QUERO Visualizar veículos em operação em um mapa

PARA Saber os veículos que estão sendo utilizados no momento e sua localização

DESENHO DA TELA



CRITÉRIO DE ACEITAÇÃO

- 1) Visualizar um mapa com todos os veículos sendo usados no momento
- 2) Apresentar sua localização atualizada no mapa utilizando a API da URBS
- 3) Cada tipo de ônibus deve ter uma coloração diferente

CRITÉRIO DE ACEITAÇÃO - DETALHAMENTO:

Dado que Estou no dashboard

E Estou logado

- | | |
|--------------|---|
| Então | Mostrar todos os veículos em operação no momento |
| E | Apresentar a localização atualizada do veículo no mapa |
| E | Apresentar cada tipo de ônibus com uma cor correspondente ao tipo |

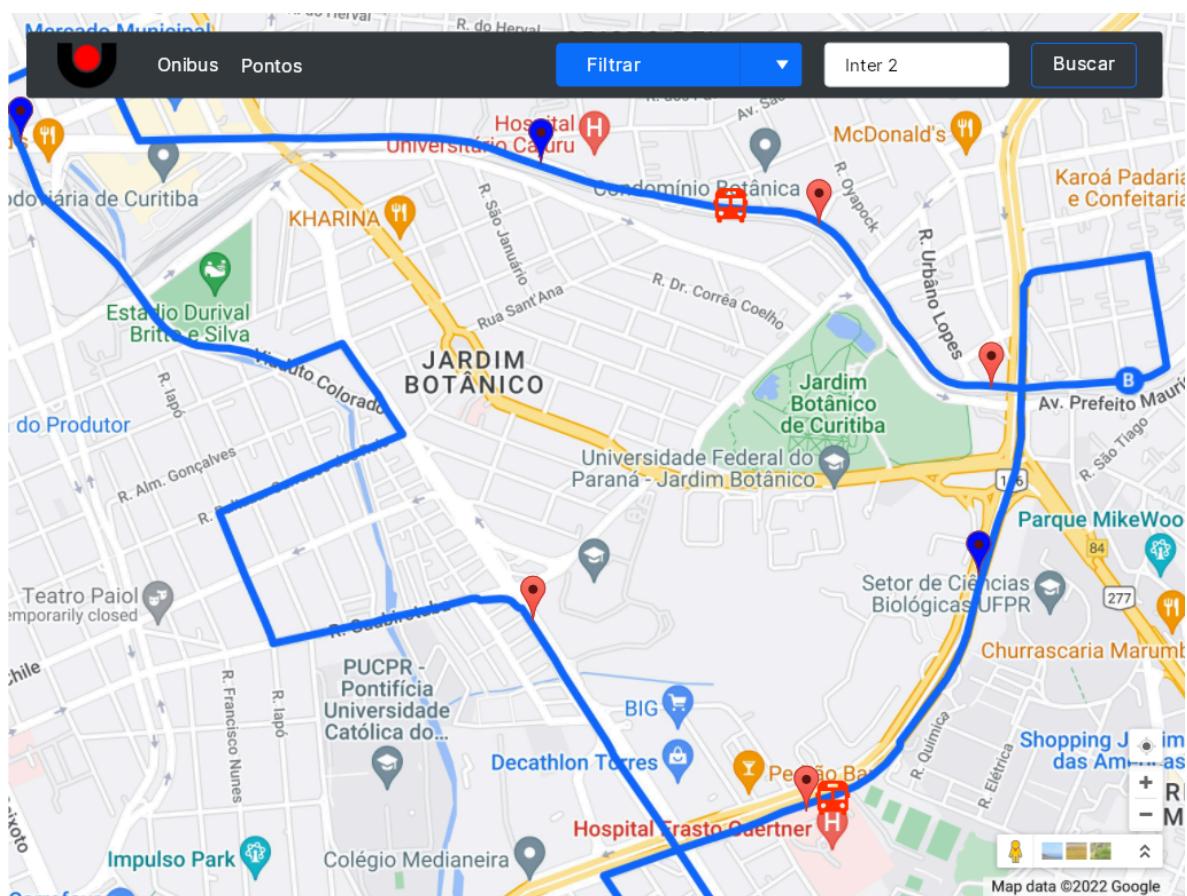
HU004 - Exibir rotas da linha

SENDO URBS

QUERO Buscar uma linha

PARA Ver o trajeto desta linha

DESENHO DA TELA.



Critérios de aceitação

- 1) Poder buscar uma linha pelo código ou nome
- 2) Quando uma linha for selecionada, destacar o seu itinerário no mapa
- 3) Apresentar no mapa todas as paradas dessa linha
- 4) Apresentar no mapa somente os veículos que estão operando nessa linha no momento

CRITÉRIO DE ACEITAÇÃO - DETALHAMENTO:

Dado que Estou no Dashboard

E Estou logado

1) Buscar linha por código.

Quando For digitado um código de linha no campo de busca

Então Apresentar somente os veículos em operação nesta linha

E Mostrar a rota destacada da linha no mapa

E Mostrar todas as paradas dessa linha

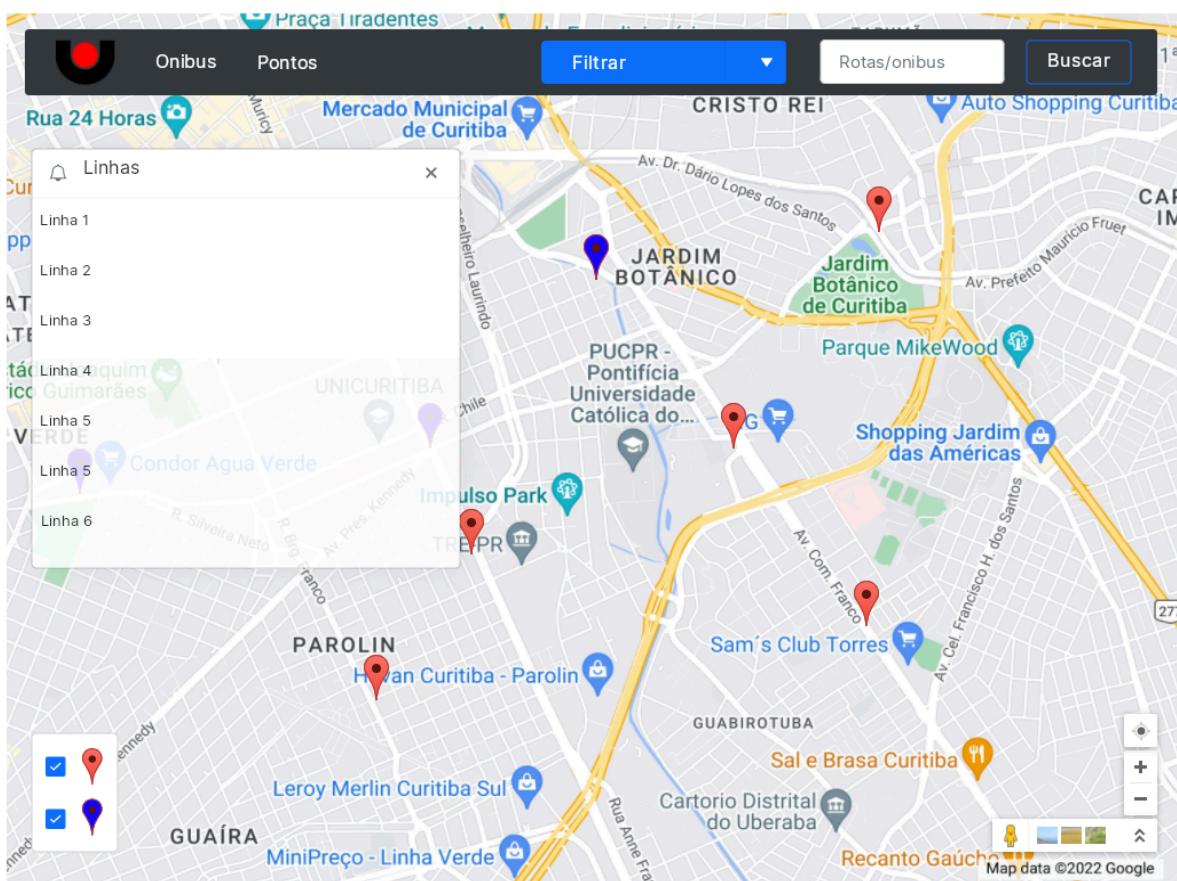
2) Buscar linha por nome

Quando For digitado o nome da linha no campo de busca

Então Apresentar somente os veículos em operação nesta linha

E Mostrar a rota destacada da linha no mapa

E Mostrar todas as paradas desse linha

HU005 - Exibir paradas da linha**SENDO** URBS**QUERO** Visualizar os pontos de ônibus**PARA** Ver o sua localização**DESENHO DA TELA**


Ônibus
Pontos

Filtrar

	Terminal Santa Candida Av. Paraná, 4818 - Santa Cândida, Curitiba - PR, 82620-360
	Terminal Santa Candida Av. Paraná, 4818 - Santa Cândida, Curitiba - PR, 82620-360
	Terminal Santa Candida Av. Paraná, 4818 - Santa Cândida, Curitiba - PR, 82620-360
	Terminal Santa Candida Av. Paraná, 4818 - Santa Cândida, Curitiba - PR, 82620-360
	Terminal Santa Candida Av. Paraná, 4818 - Santa Cândida, Curitiba - PR, 82620-360
	Terminal Santa Candida Av. Paraná, 4818 - Santa Cândida, Curitiba - PR, 82620-360

Visualizar no mapa

Critérios de aceitação:

- 1) Apresentar no mapa todos os pontos de ônibus, estações tubo e terminais pertencentes à rede pública de transporte
- 2) Cada tipo de parada deve ser apresentado de uma forma diferente
- 3) Ter a opção de esconder qualquer um dos tipos de parada no mapa
- 4) Selecionando uma parada, será apresentado todas as linhas que utilizam essa parada, podendo selecionar uma das linhas

Dados que Foi selecionada a opção Pontos no Dashboard

1) Apresentar pontos no mapa

Então Mostrar todos os pontos, estações tubo e terminais no mapa

E Cada tipo deve ser apresentado em uma forma diferente

2) Apresentar linhas que utilizam essa parada

2.1)

Dado que Foi selecionada a opção pontos no Dashboard

E Foi clicado em um determinado ponto no mapa

Então Apresentar uma janela com todas as linhas que utilizam essa parada

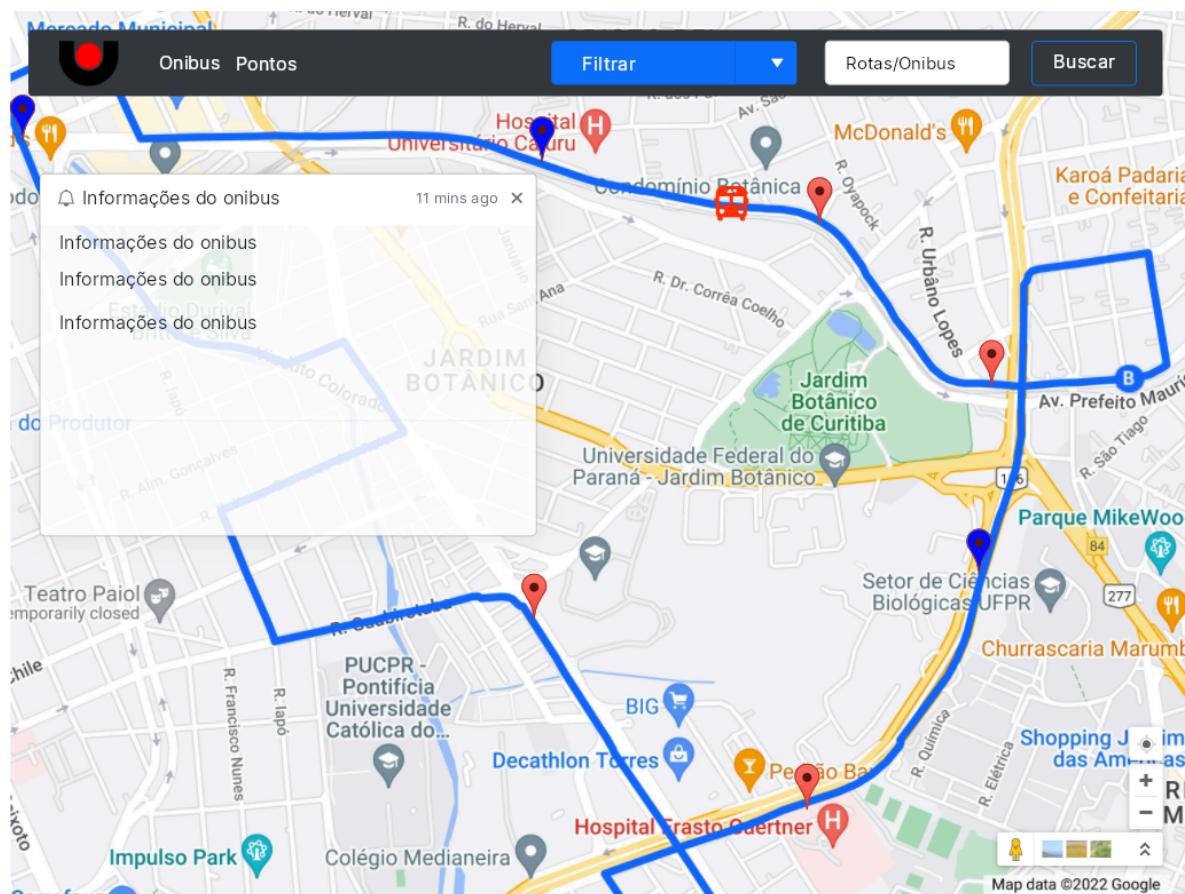
HU006 - Exibir detalhes do ônibus

SENDO URBS

QUERO Visualizar os detalhes de um veículo

PARA Saber os seus dados e situação atual

DESENHO DA TELA



Critérios de aceitação:

- 1) Poder selecionar um ônibus diretamente no mapa
- 2) Selezionando um ônibus será destacado no mapa a rota da linha
- 3) Ao selecionar um ônibus, seja apresentada uma janela com as seguintes informações

CRITÉRIOS DE ACEITAÇÃO - DETALHAMENTO:

Dado que Estou no dashboard

E Foi clicado em veículo no mapa

Então
momento

Apresentar o itinerário da linha que o veículo está operando no momento

E Apresentar uma janela com as informações do veículo (Código linha, nome linha, código veículo, tipo veículo, número de passageiros, situação atualizada, adaptado para cadeirantes e empresa do ônibus)

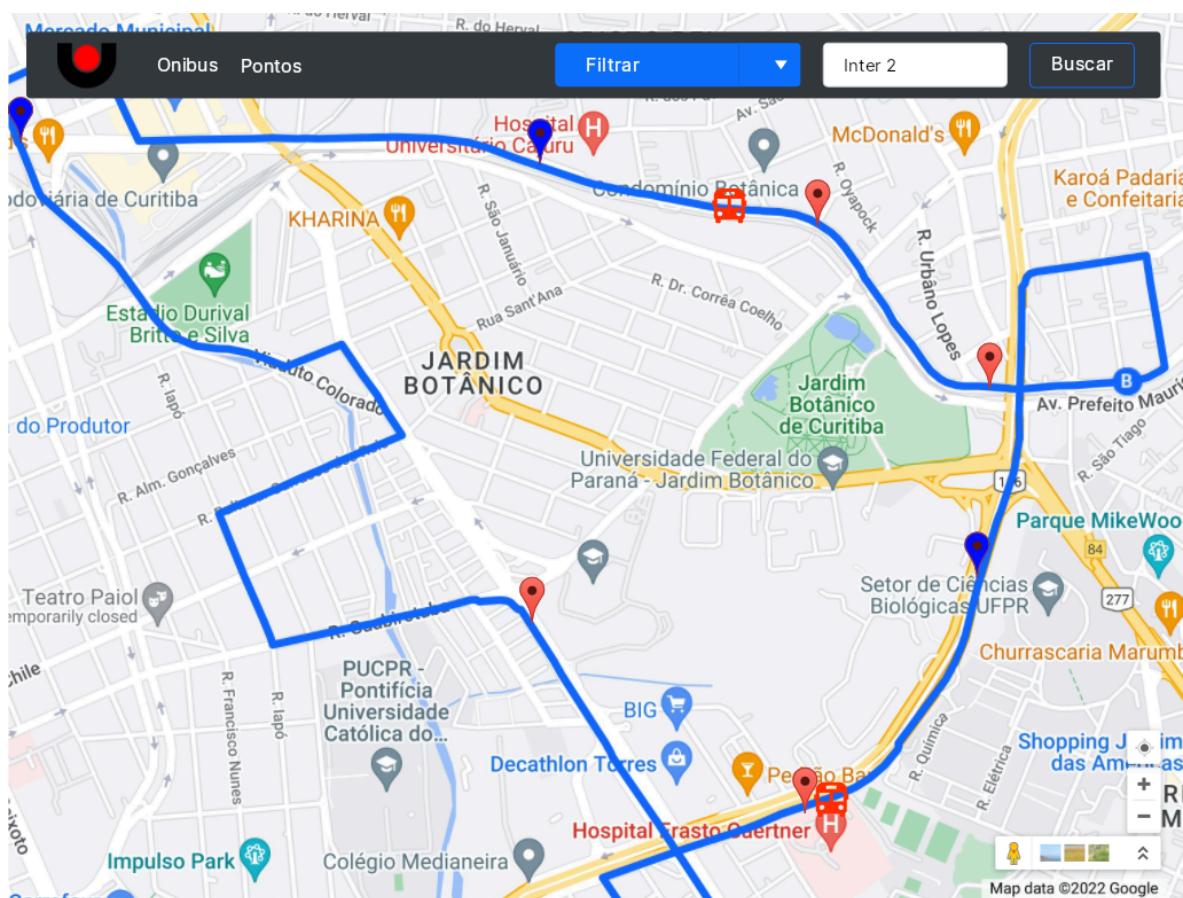
HU007 - Filtrar linhas

SENDO URBS

QUERO Pesquisar uma determinada região

PARA Saber as linhas que passa por essa região

DESENHO DA TELA



Critério de aceitação

1) Poder pesquisar as linhas que passam por uma determinado bairro ou cidade

CRITÉRIO DE ACEITAÇÃO - DETALHADO

Dado que Estou no Dashboard

Quando Na barra de pesquisa foi digitado um endereço, bairro ou cidade

Então Apresentar a região destacada no mapa

E Apresentar somente os veículos que operam na região

- E** Destacar o itinerário das linhas que passam pela região no mapa
- E** Apresentar uma listagem com todas as linhas que operam na região

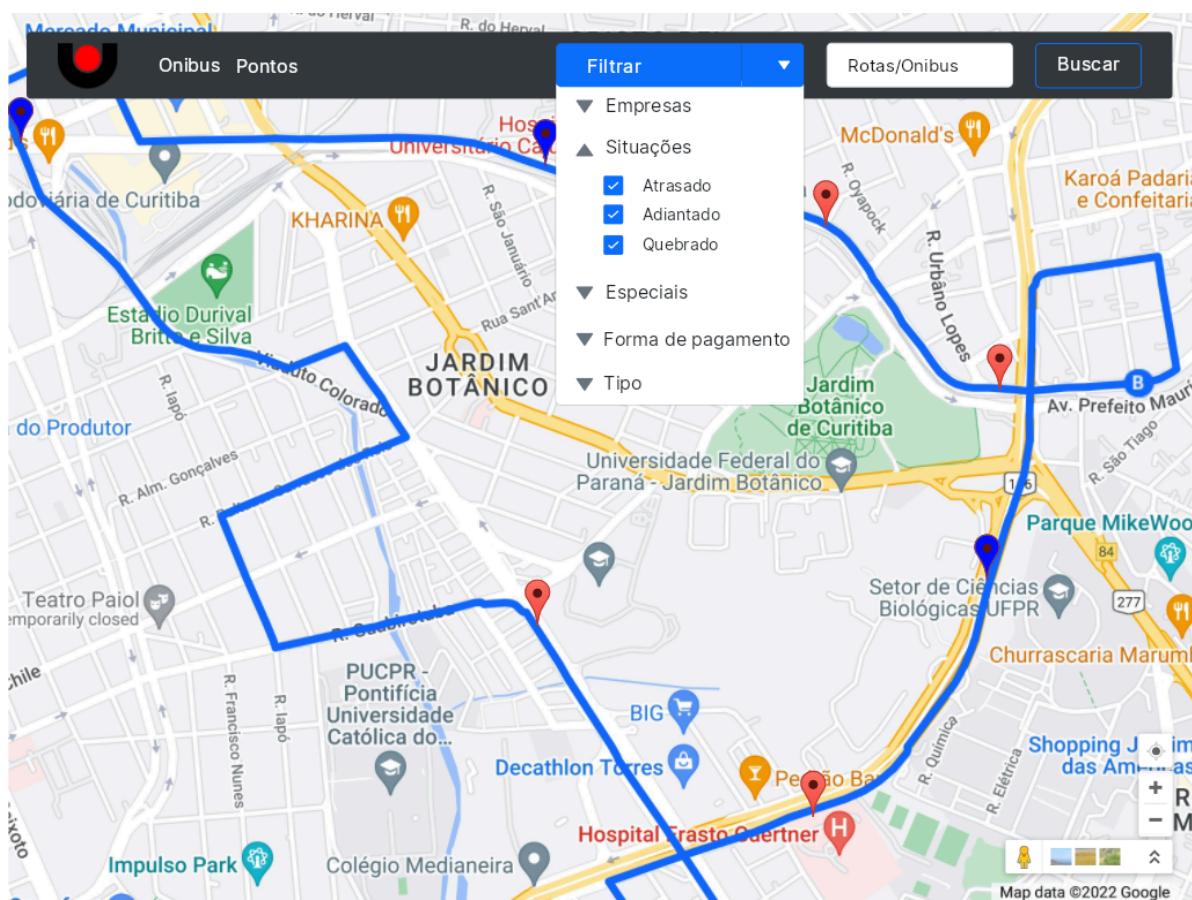
HU008 - Filtrar ônibus

SENDO URBS

QUERO Poder filtrar ônibus

PARA Identificar os veículos por determinados parâmetros

.DESENHO DA TELA



Map data ©2022 Google

Onibus Pontos

Filtrar

- Empresas
- Situações
- Especiais
 - Suporte a cadeirantes
- Forma de pagamento
- Tipo

Rotas/Onibus

Buscar

Mercado Municipal

Onibus Pontos

Filtrar

- Empresas
- Situações
- Especiais
 - Dinheiro
 - Cartão
- Forma de pagamento
- Tipo

Rotas/Onibus

Buscar

Map data ©2022 Google

Onibus Pontos

Filtrar

- Empresas
- Situações
- Especiais
 - Suporte a cadeirantes
- Forma de pagamento
- Tipo

Rotas/Onibus

Buscar

Map data ©2022 Google

Critério de aceitação

- 1) Poder filtrar os ônibus pelas situações: Atrasado, adiantado ou quebrado
- 2) Poder filtrar os ônibus que são adaptados
- 3) Poder filtrar os veículos que operam somente com cartão
- 4) Poder filtrar os veículos por tipo

CRITÉRIOS DE ACEITAÇÃO - DETALHAMENTO:

Dado que Estou no Dashboard

1) Filtrar por situação

Quando For selecionado o filtro situação

Então Apresentar no mapa apenas os veículos da situação selecionada

2) Filtrar por adaptados

Quando For selecionado veículos adaptados

Então Apresentar somente os veículos que são adaptados para cadeirantes

3) Filtrar por pagamento

- | | |
|---------------|--|
| Quando | For selecionado o filtro forma de pagamento |
| Então | Apresentar somente os veículos que tem aquela forma de pagamento |

4) Filtrar veículos por tipo

- | | |
|---------------|--|
| Quando | For selecionado o filtro tipo do veículo |
| Então | Apresentar somente os veículos do tipo selecionado no mapa |

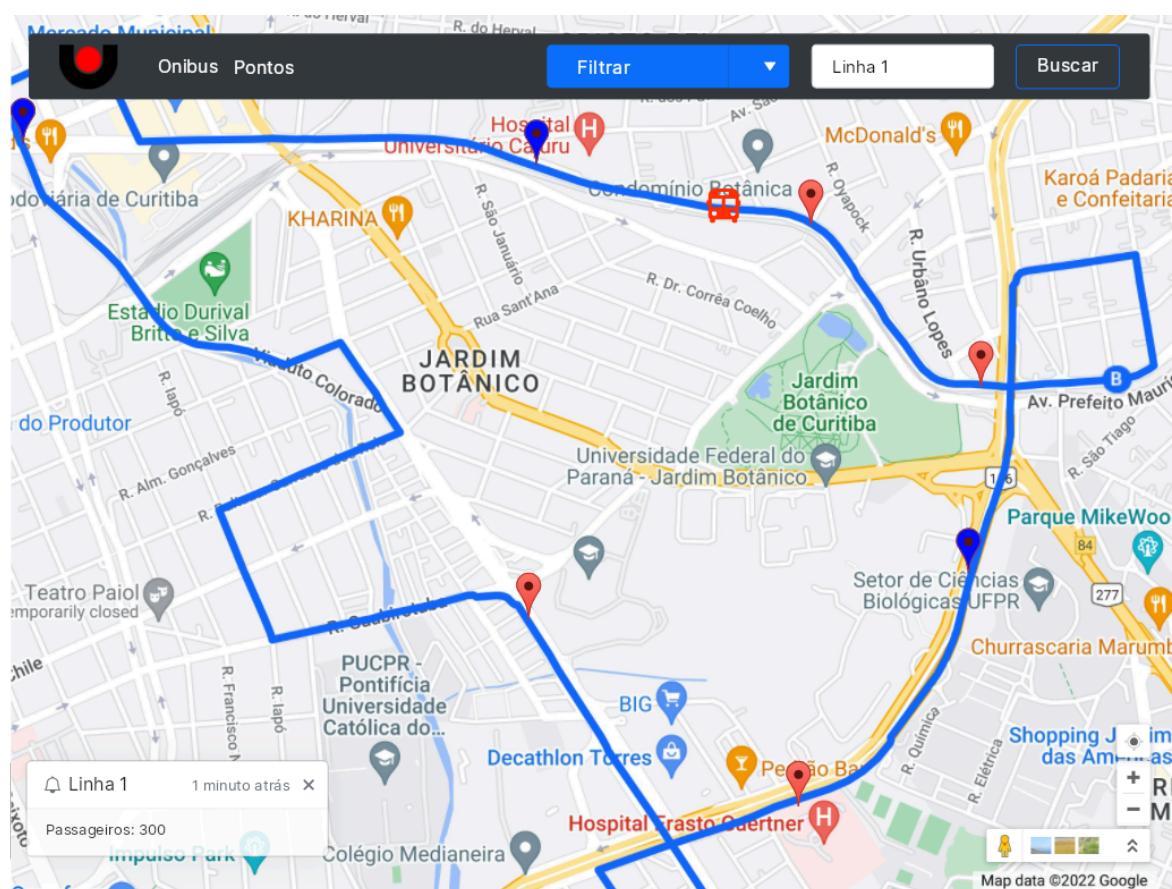
HU009 - Exibir quantidade de pessoas por ônibus

SENDO URBS

QUERO Quantidade de pessoas em um veículo

PARA Controlar a lotação dos ônibus

DESENHO DA TELA



Critérios de aceitação

1) Saber quantos usuários estão em uma determinada linha

CRITÉRIO DE ACEITAÇÃO - DETALHAMENTO

Dado que Estou no dashboard

1) Apresentar locação do veículo

Quando Clicar em um ônibus no mapa

Então Apresentar uma janela com suas informações, incluindo a quantidade de pessoas presente no veículo

HU010 - Gerar relatório das linhas

SENDO URBS

QUERO Poder gerar relatórios

PARA Realizar a análise da frota

DESENHO DA TELA

Linhas: Linha 1 ▾ Adicionar
Linha 2

Periodo: 2020/12/30 - 2021/01/12

Ordenar: Data
 Quantidade de veiculos

Gerar relatorio

Critério de aceitação

- 1) Poder gerar um relatório selecionando um determinada intervalo de tempo
- 2) Poder selecionar o número de linhas que quero verificar ou selecionar todas

CRITÉRIO DE ACEITAÇÃO - DETALHAMENTO

Dado que Estou na tela de relatórios

1) Gerar o relatório de N linhas

Dado que Foi selecionado as linhas

E Foi selecionado o período

Quando For clicado no botão gerar relatório

Então Gerar o relatório das linhas selecionadas com o número máximo, mínimo e média de passageiros por ônibus e quantidade de veículos da linha

2) Gerar relatório de todas as linhas

Dado que Foi selecionado a box de todas as linhas

E O período foi selecionado

Quando Foi clicado no botão gerar relatório

Então Gerar o relatório de todas as linhas, com o número de passageiros e a quantidade de veículos que operam na linha

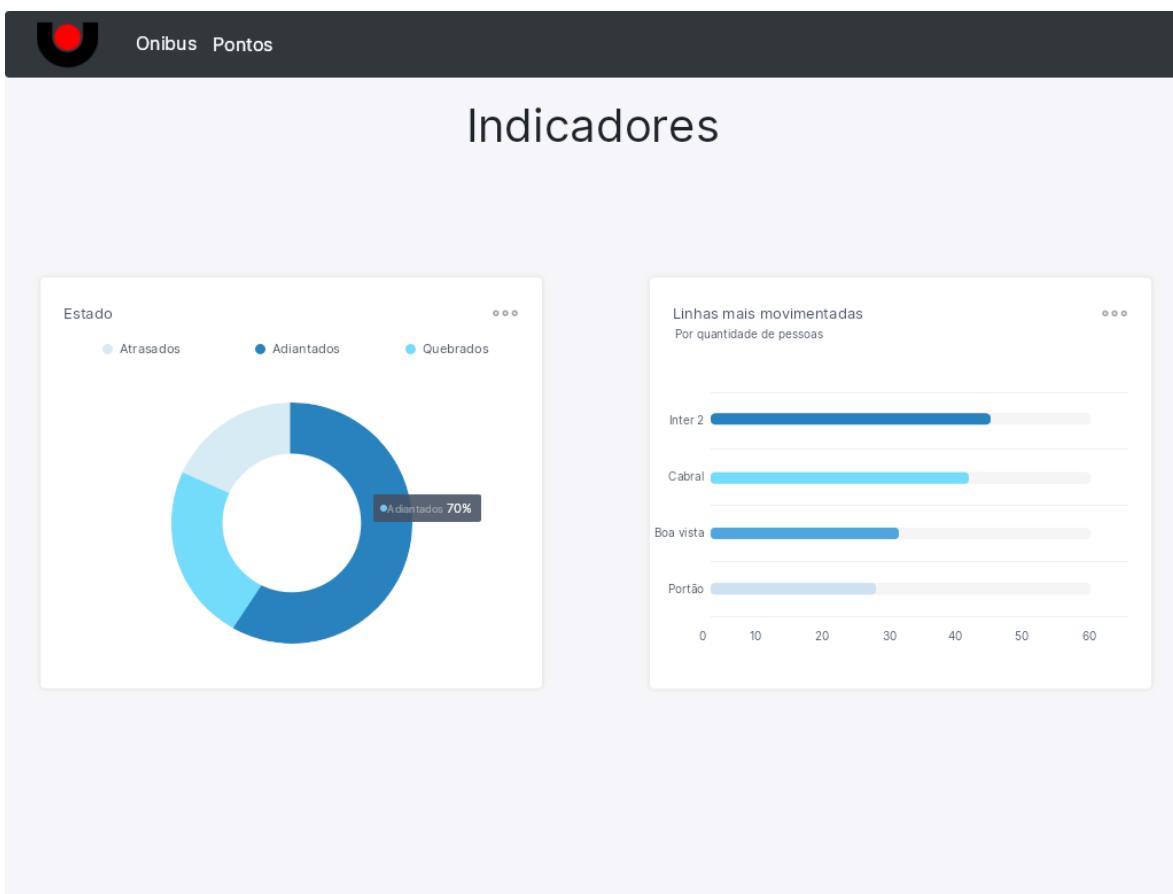
HU011 - Gerar indicadores

SENDO URBS

QUERO Poder gerar indicadores

PARA Administrar os dados gerados com o Dashboard

DESENHO DA TELA



CRITÉRIO DE ACEITAÇÃO

- 1) Gerar indicadores de quantidade de veículos em operação por linha
- 2) Gerar indicadores de veículos adiantados de uma linha
- 3) Gerar indicadores de veículos atrasados
- 4) Gerar indicadores de veículos no horário
- 5) Gerar indicadores de quantidade de pessoas por veículo
- 6) Gerar indicadores de quantidade de pessoas por linha

CRITÉRIOS DE ACEITAÇÃO - DETALHAMENTO

Dado que Estou na página de indicadores

- 1) Visualizar indicadores**

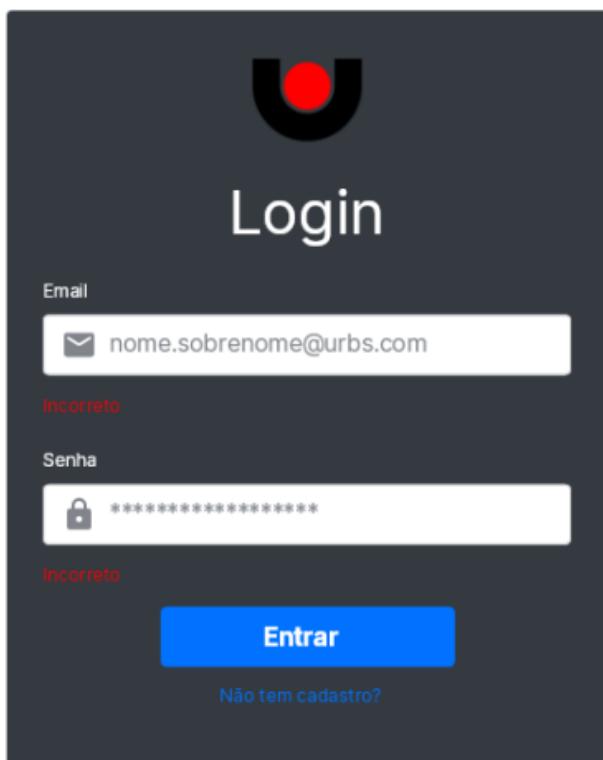
Dado que	Os indicadores foram atualizados com uma tarefa diária
Então	Será apresentado um indicador gráfico veículos por linha
E	Será apresentado um indicador gráfico de ônibus adiantados
E	Será apresentado um indicador gráfico de ônibus atrasados
E	Será apresentado um indicador gráfico de ônibus no horário
E	Será apresentado um indicador gráfico pessoas por veículo
E	Será apresentado um indicador gráfico de pessoas por linhas

HU001 Aplicativo - Realizar login

SENDO Usuário da URBS

QUERO Fazer Login

DESENHO DA TELA



CRITÉRIOS DE ACEITAÇÃO:

- 1) Deve logar o usuário
- 2) Deve redirecionar para a página de esqueci senha
- 3) Deve redirecionar para a página de cadastro

CRITÉRIOS DE ACEITAÇÃO – DETALHAMENTO:

Critério de contexto (Válido como premissa para todos os critérios):

Dado que estou na tela de login

1) Deve logar o usuário.

1.1)

Dado que Os dados não foram preenchidos

Quando Clicar no botão de login

Então Mostrar um aviso que os dados não foram preenchidos

1.2)

Dado que A senha ou e-mail estão incorretos.

Quando Clicar no botão de login

Então Mostrar um aviso que os dados estão incorretos

1.3)

Dado que Todos os dados estão preenchidos corretamente

Quando Clicar no botão de login

Então Logar o usuário correspondente

E Redirecionar para a página de principal

2) Deve redirecionar para a página de esqueci senha.

Quando Clicar no botão “esqueci senha”

Então Redirecionar para a página de esqueci senha

3) Deve redirecionar para a página de cadastro.

Quando Clicar no botão “cadastro”

Então Redirecionar para a página de cadastro

HU002 Aplicativo - Realizar cadastro no sistema

Sendo Usuário

Quero Me cadastrar

Para Acessar o sistema

DESENHO DA TELA

The wireframe shows a registration form with the following fields:

- First Name: Text input field
- Last Name: Text input field
- Date of Birth: Text input field containing "2020/12/30"
- E-mail: Text input field
- Confirm E-mail: Text input field
- Phone: Text input field containing "(99) 99999-9999"
- Password: Text input field
- Confirm Password: Text input field
- CPF: Text input field
- Confirm Button: A blue rectangular button labeled "Confirmar".

CRITÉRIOS DE ACEITAÇÃO:

- 1) Deve cadastrar o usuário
- 2) Deve redirecionar para a página inicial

CRITÉRIOS DE ACEITAÇÃO – DETALHAMENTO:

Dado que estou na página de cadastro

1) Deve cadastrar o usuário

1.1)

Dado que algum campo está vazio

Quando	clicar no botão “Confirmar”
Então	mostrar aviso campos vazios
1.2) Dado	que e-mail inserido está invalido
Quando	clicar no botão “Confirmar”
Então	mostrar aviso de e-mail invalido
1.3) Dado	que senhas não correspondem
Quando	clicar no botão “Confirmar”
Então	mostrar aviso de senhas diferentes
1.4) Dado	que o telefone está invalido
Quando	clicar no botão “Confirmar”
Então	mostrar aviso de telefone invalido
1.5) Dado	que o cpf está invalido
Quando	clicar no botão “Confirmar”
Então	mostrar aviso de cpf invalido
1.6) Dado	que todos os campos estão corretamente preenchidos
Quando	clicar no botão “Confirmar”
Então	criar conta no banco de dados
E	redirecionar para a página de login
2) Deve redirecionar para a página inicial	
Quando	Clicar no ícone da URBS
Então	Redirecionar para a página de login

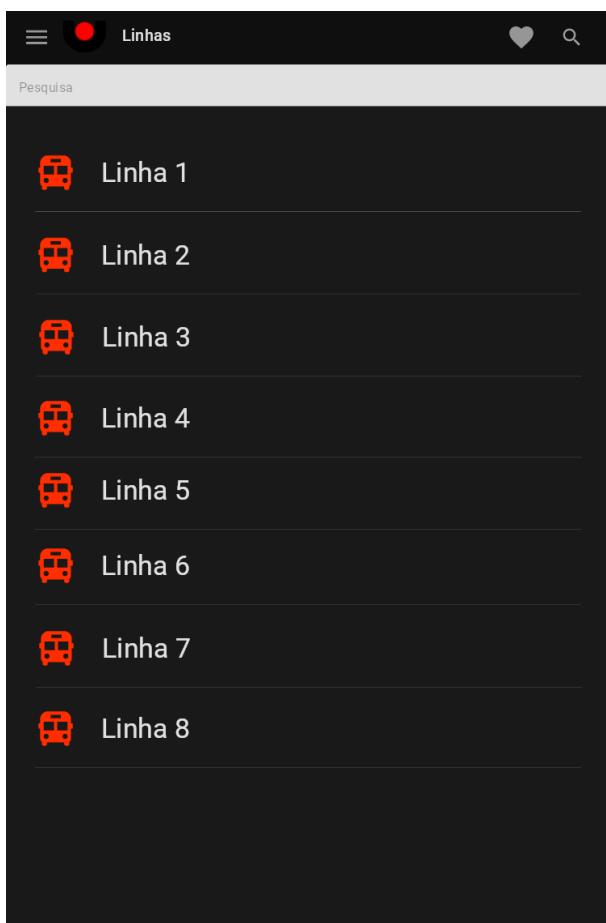
HU003 Aplicativo - Listar Linhas

SENDO Usuário

QUERO Listar as linhas da URBS

PARA Visualizar as linhas de ônibus

DESENHO DA TELA



CRITÉRIOS DE ACEITAÇÃO:

- 1) Deve listar todas as linhas de ônibus
- 2) Deve permitir pesquisar por nome ou código da linha.

CRITÉRIOS DE ACEITAÇÃO – DETALHAMENTO:

Dado que estou na página onde é listada as linhas de ônibus.

1) Deve listar todas as linhas de ônibus

1.1)

Dado que uma linha foi selecionada

Quando clicar em cima da linha

Então mostrar o itinerário da linha

2) Deve permitir pesquisar por nome ou código da linha.

Dado que uma linha específica foi buscada.

Quando foi digitado o nome ou número de uma linha no campo de pesquisa

Então reduzir a lista a somente as linhas que sejam compatíveis com a busca

HU004 Aplicativo - Favoritar linhas

SENDO

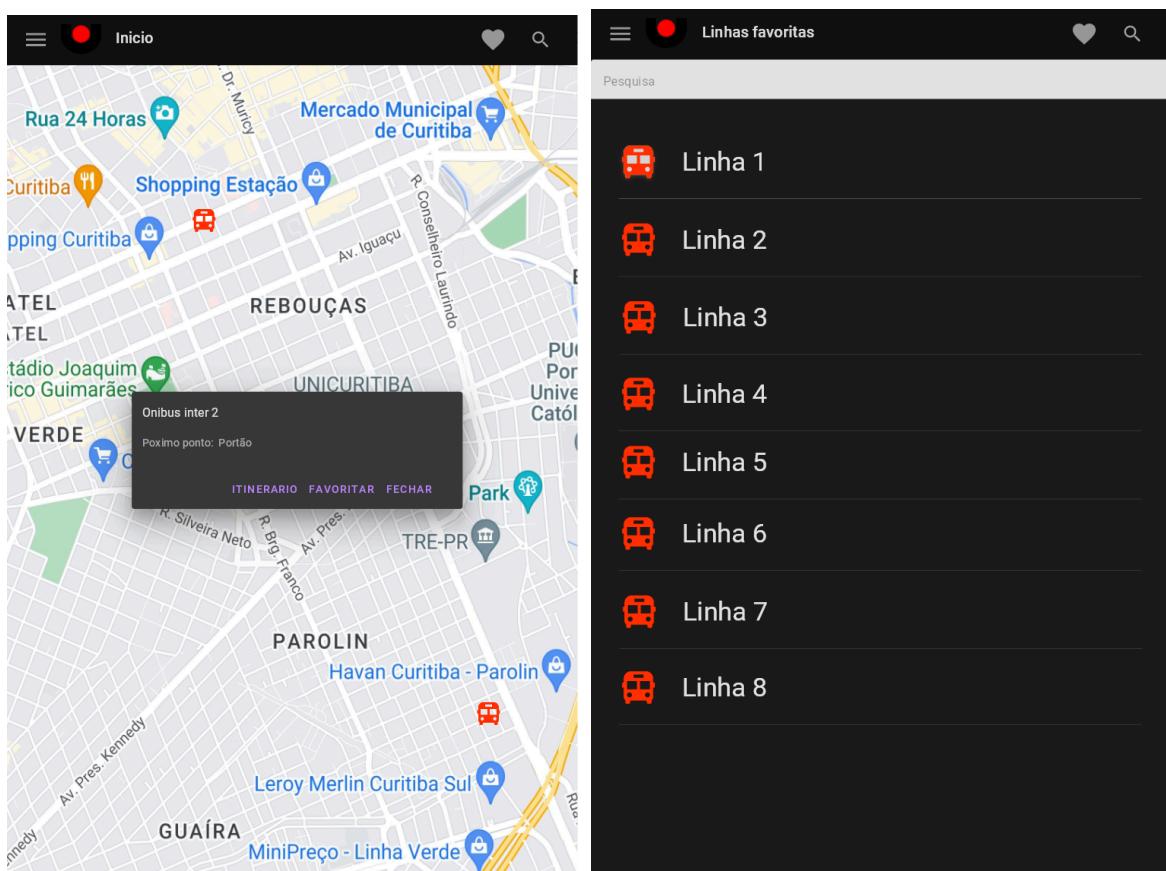
Usuário

QUERO

Favoritar uma linha

PARA

Facilitar e agilizar minhas linhas de preferência



CRITÉRIOS DE ACEITAÇÃO:

1) Adicionar à lista de favoritas a linha selecionada

CRITÉRIOS DE ACEITAÇÃO – DETALHAMENTO:

Dado que estou na visualização do itinerário da linha de ônibus.

1) Adicionar à lista de favoritas a linha selecionada

1.1)

Dado que uma linha foi favoritada

Quando selecionar a linha para favoritar

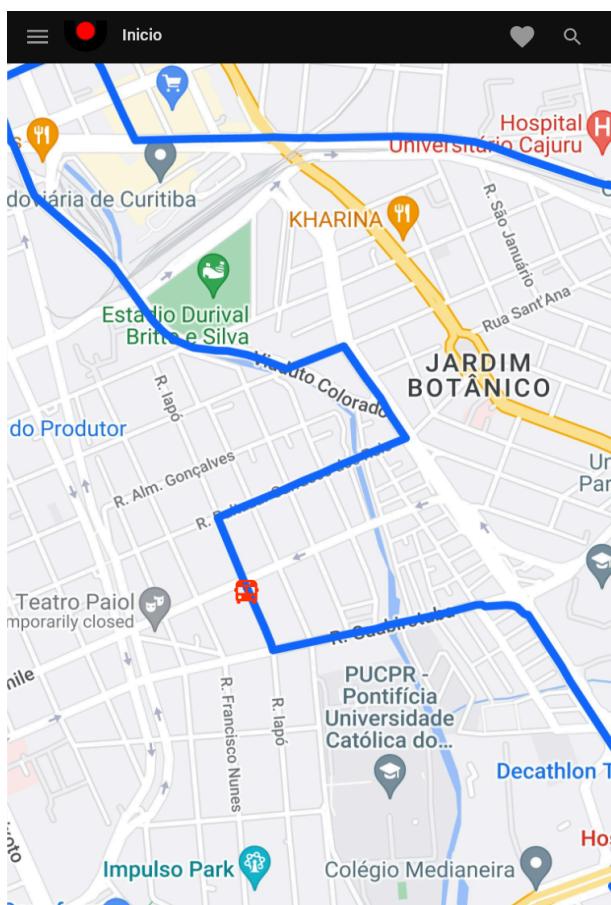
Então adicionar a linha para a lista de favoritas

HU005 Aplicativo - Exibir itinerário da linha

SENDO Usuário

QUERO Visualizar o itinerário da linha

PARA Acessar as informações do trajeto



CRITÉRIOS DE ACEITAÇÃO:

- 1) Exibir adequadamente o itinerário da linha
- 2) Exibir as paradas da linha
- 3) Exibir os detalhes da linha

CRITÉRIOS DE ACEITAÇÃO – DETALHAMENTO:

Dado que estou na visualização da listagem das linhas.

- 1) Exibir adequadamente o itinerário da linha**

1.1)

Dado que uma linha foi selecionada

Quando clicar na linha na listagem de todas linhas

Então destacar seu trajeto no mapa

2) Exibir adequadamente as paradas da linha

2.1)

Dado que uma linha foi selecionada

Quando clicar na linha na listagem de todas linhas

Então sobre o trajeto destacado exibir as paradas da linha

3) Exibir adequadamente o itinerário da linha

3.1)

Dado que uma linha foi selecionada

Quando clicar na linha na listagem de todas linhas

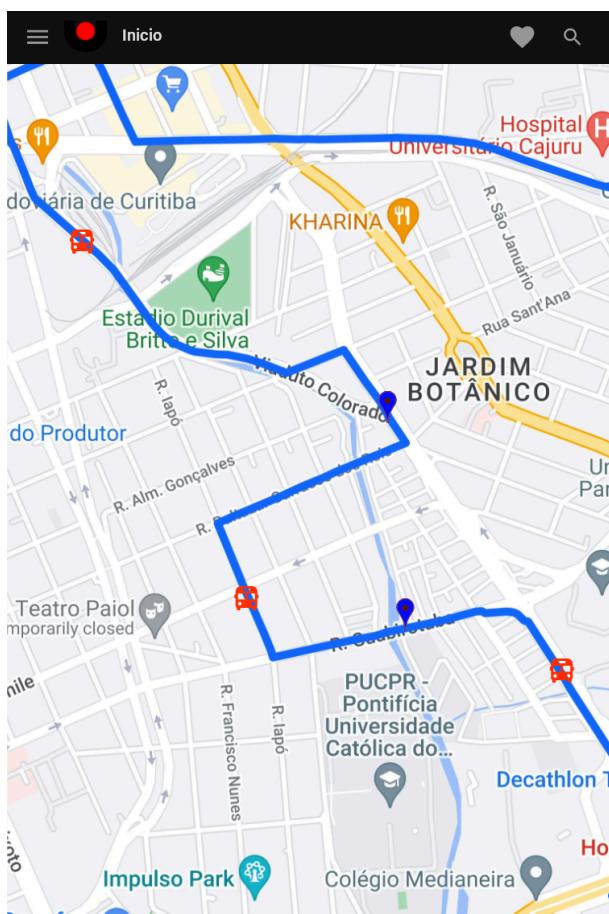
Então exibir os detalhes da linha

HU006 Aplicativo - Visualizar veículos em operação

SENDO Usuário

QUERO Visualizar os veículos em operação

PARA Analisar a posição do ônibus



CRITÉRIOS DE ACEITAÇÃO:

- 1) Exibir adequadamente os veículos em operação

CRITÉRIOS DE ACEITAÇÃO – DETALHAMENTO:

Dado que estou na visualização do itinerário da linha no mapa.

1) Exibir adequadamente o itinerário da linha

1.1)

Dado que uma linha foi selecionada

Quando clicar na linha na listagem de todas linhas

Então destacar os veículos em operação da linha selecionada

HU007 Aplicativo - Visualizar Horários

SENDO Usuário

QUERO Visualizar o horário das linhas em um ponto

PARA Saber o horário dos ônibus



CRITÉRIOS DE ACEITAÇÃO:

- 1) Exibir a tabela de horários da linha que selecionei
- 2) Os horários apresentados deve ser do ponto de ônibus que estou

CRITÉRIOS DE ACEITAÇÃO – DETALHAMENTO:

Dado que estou nas informações detalhadas de um ponto de ônibus

1) Exibir adequadamente o horário da linha

1.1)

Dado que uma linha foi selecionada

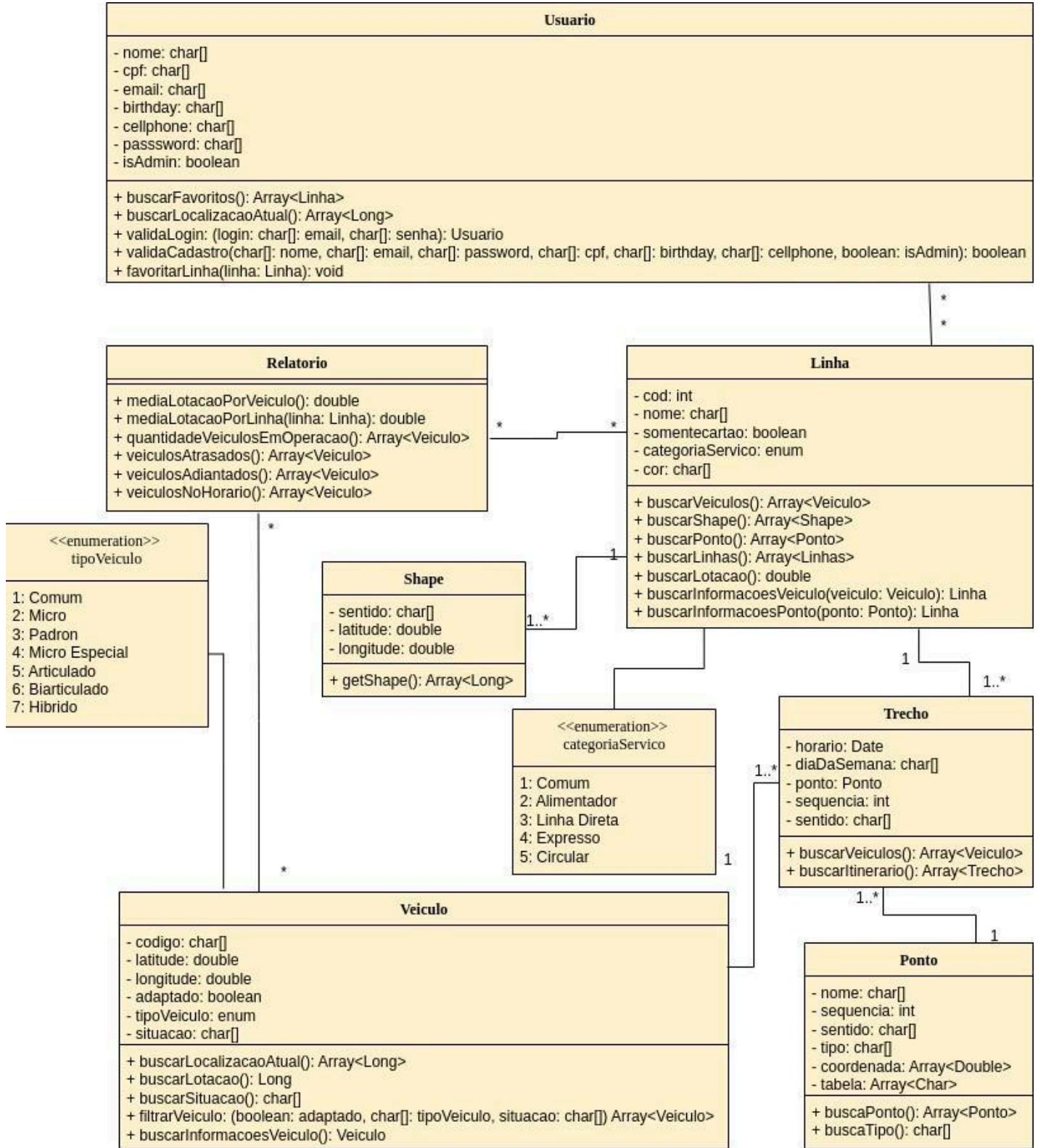
E foi clicado em um ponto de ônibus no mapa

Quando for clicado no botão exibir horários

Então apresentar a tabela de horários do ponto selecionado

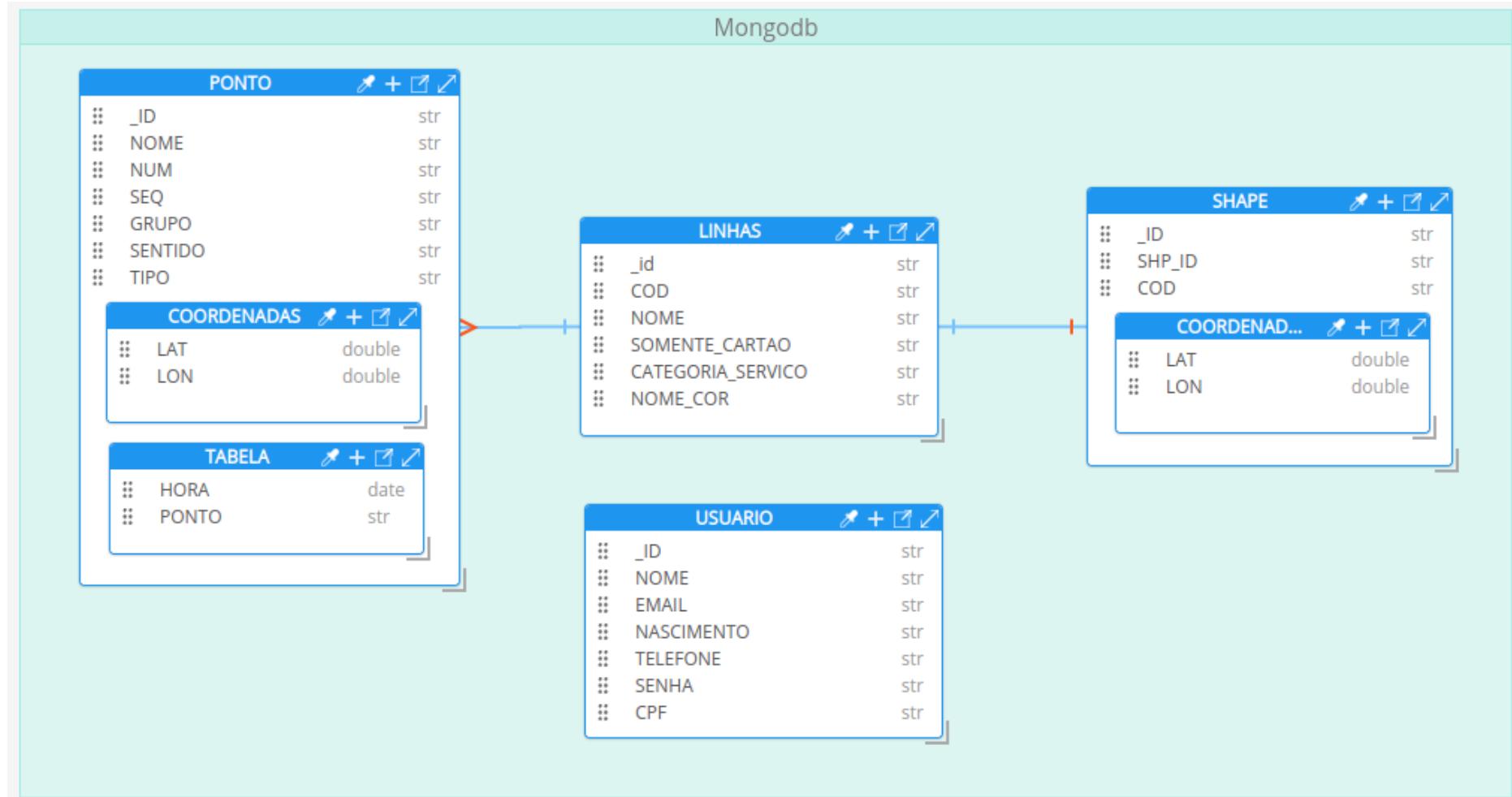
APÊNDICE C - DIAGRAMA DE CLASSE

FIGURA 33 - DIAGRAMA DE CLASSE



APÊNDICE D - MODELO FÍSICO DO BANCO DE DADOS

FIGURA 34 - MODELO FÍSICO BANCO DE DADOS



APÊNDICE E - DIAGRAMAS DE SEQUÊNCIA

FIGURA 35 - DIAGRAMA DE SEQUÊNCIA - UC01 REALIZAR LOGIN

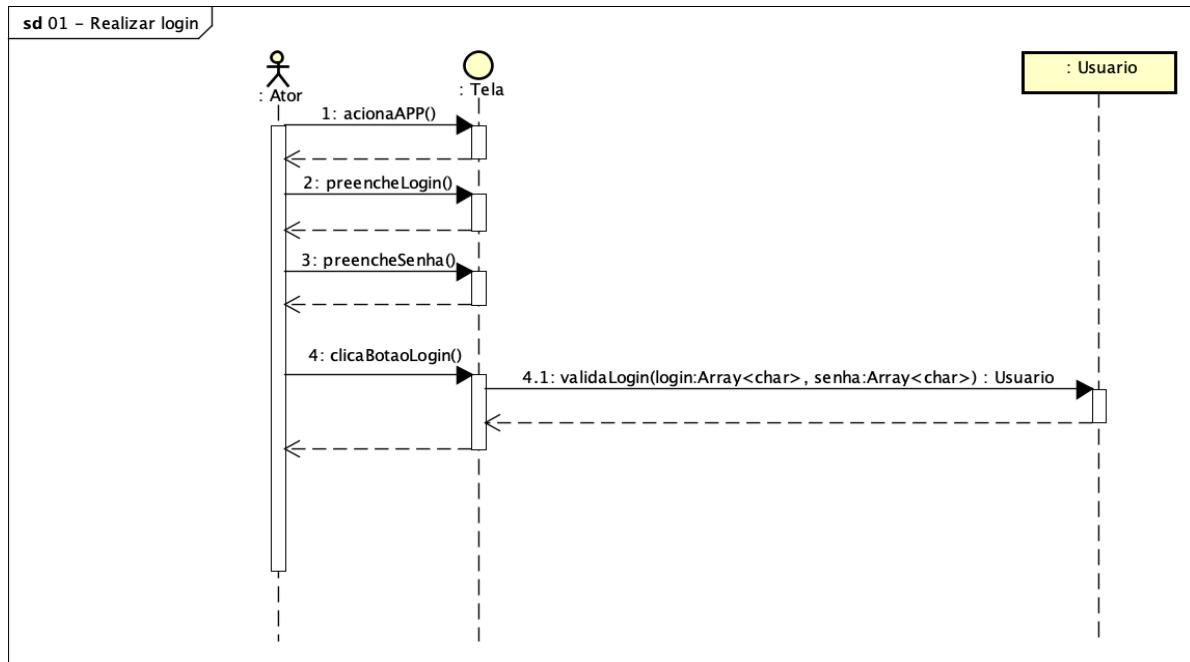


FIGURA 36 - DIAGRAMA DE SEQUÊNCIA - UC02 EXIBIR VEÍCULOS EM OPERAÇÃO

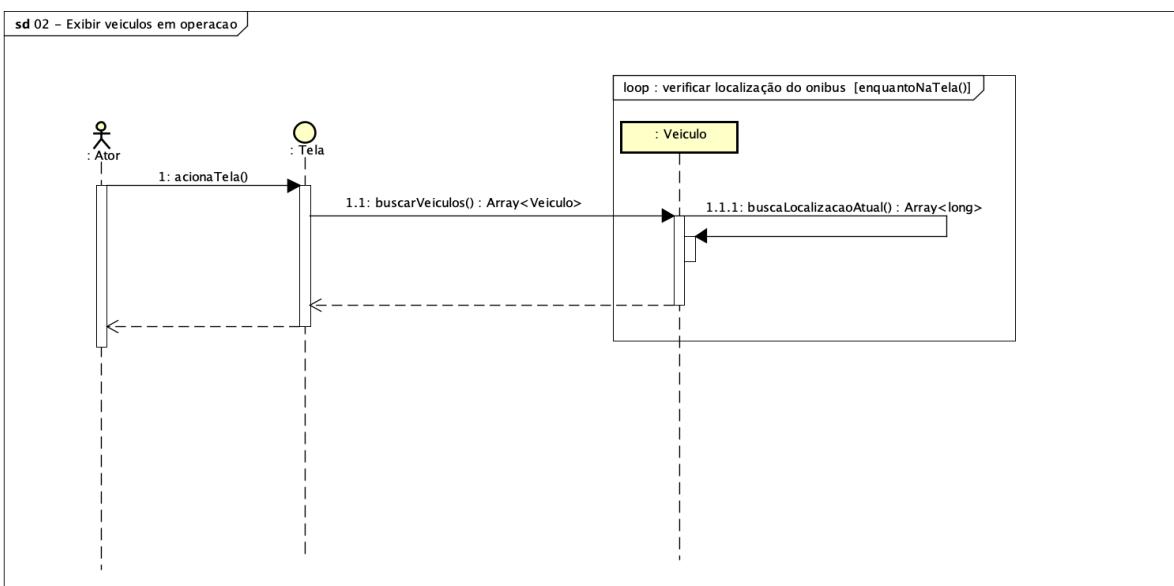


FIGURA 37 - DIAGRAMA DE SEQUÊNCIA - UC03 EXIBIR ROTAS DAS LINHAS

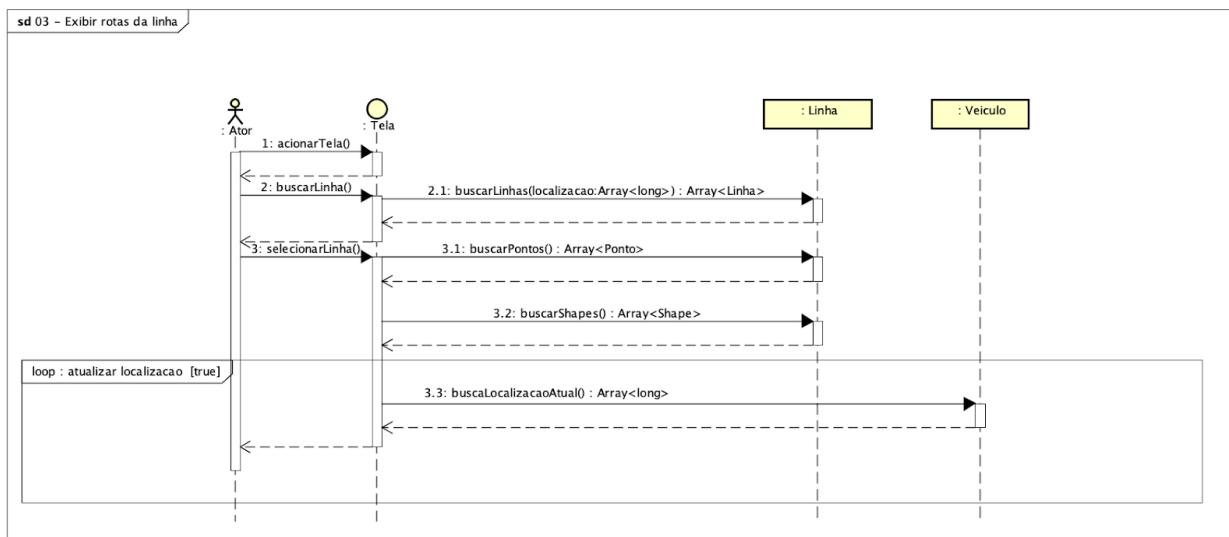


FIGURA 38 - DIAGRAMA DE SEQUÊNCIA - UC04 EXIBIR PARADAS DA LINHA

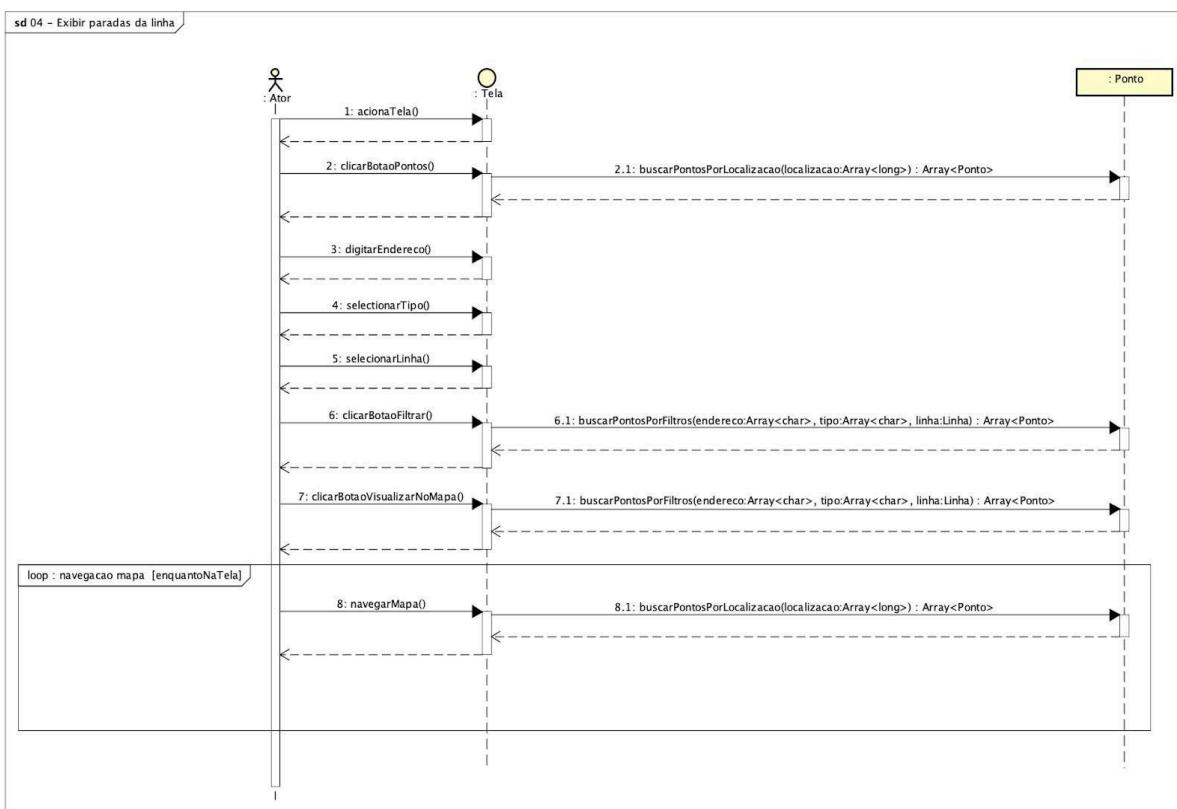


FIGURA 39 - DIAGRAMA DE SEQUÊNCIA - UC05 EXIBIR DETALHES DO ÔNIBUS

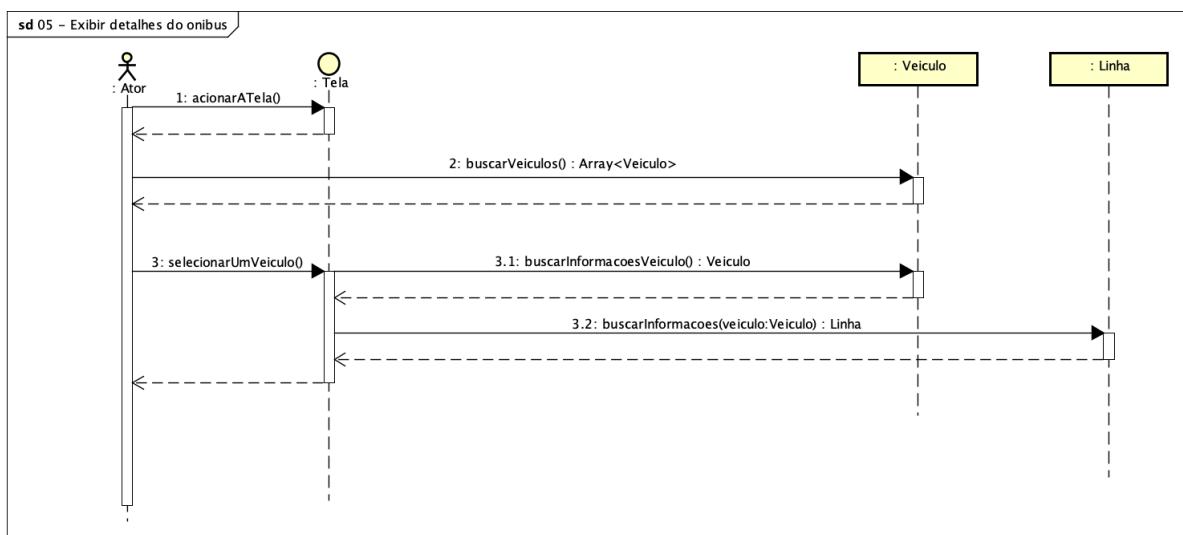


FIGURA 40 - DIAGRAMA DE SEQUÊNCIA - UC06 FILTRAR LINHAS

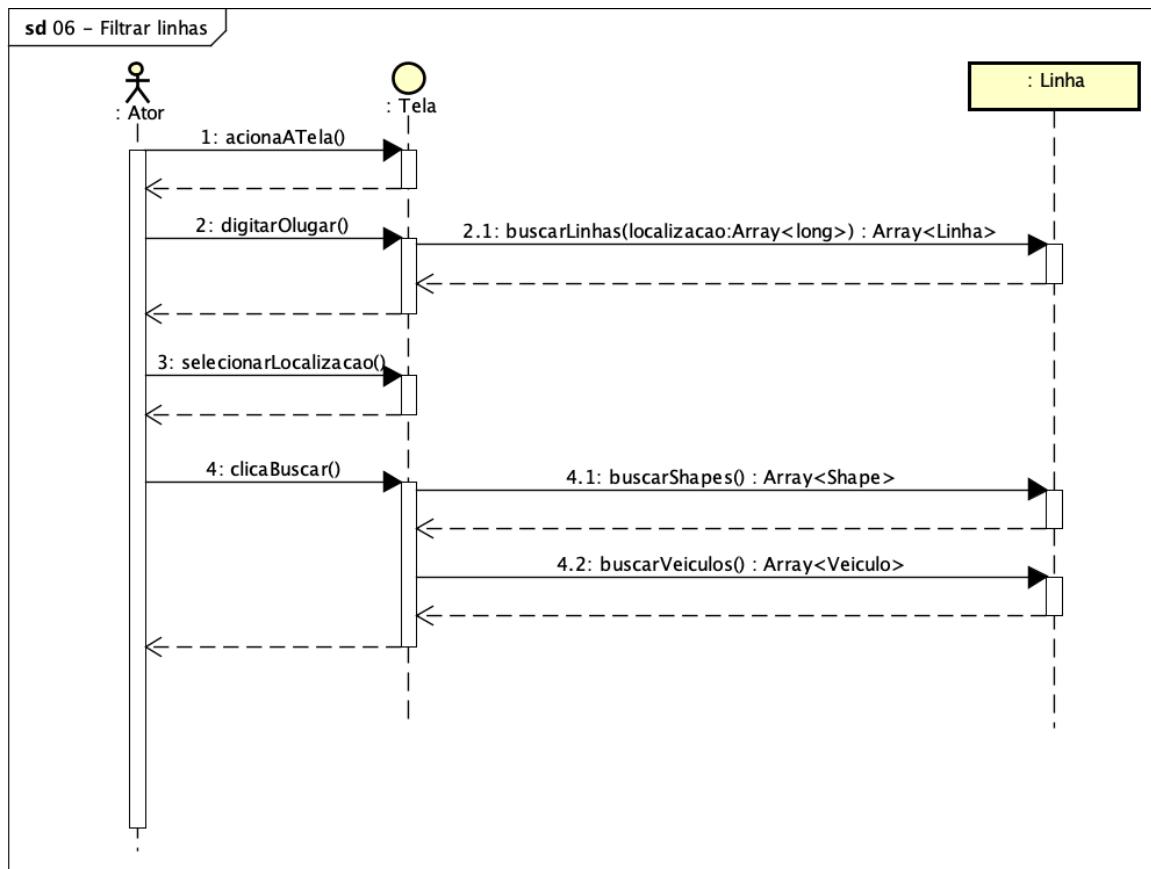


FIGURA 41 - DIAGRAMA DE SEQUÊNCIA - UC07 FILTRAR ÔNIBUS

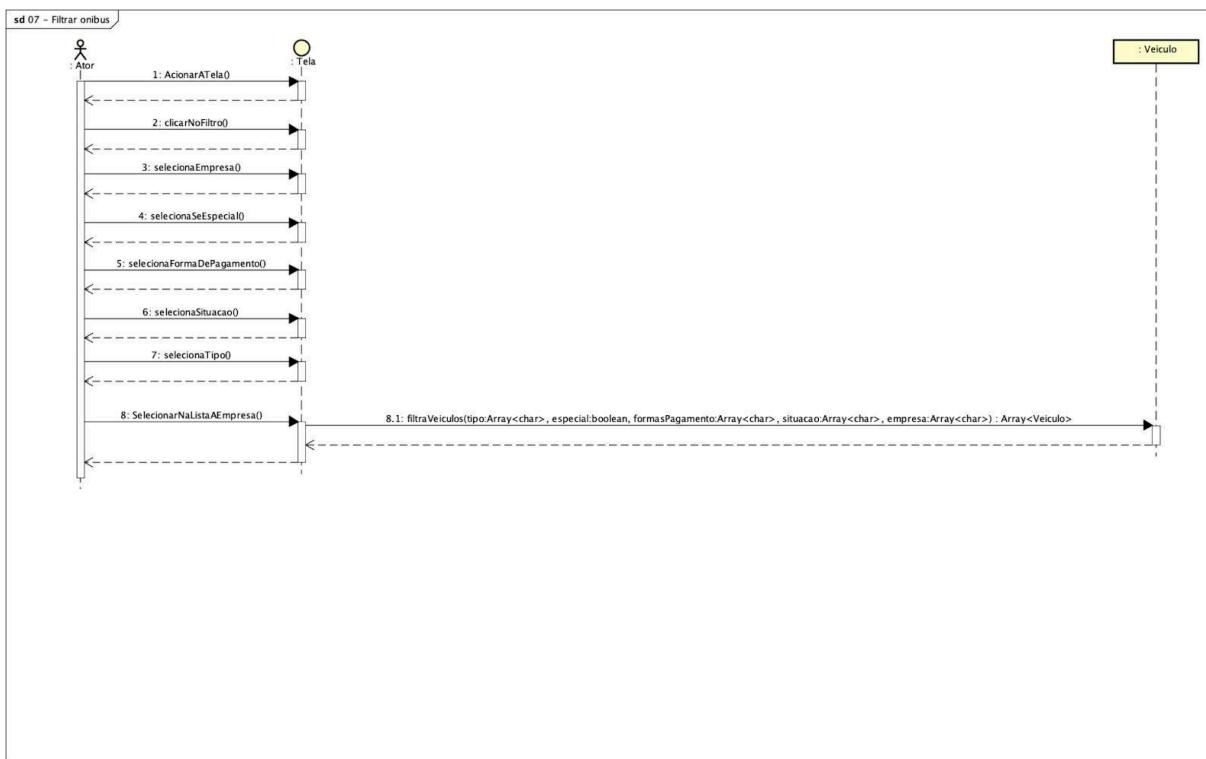


FIGURA 42 - DIAGRAMA DE SEQUÊNCIA - UC08 EXIBIR QUANTIDADE DE PESSOAS POR ÔNIBUS

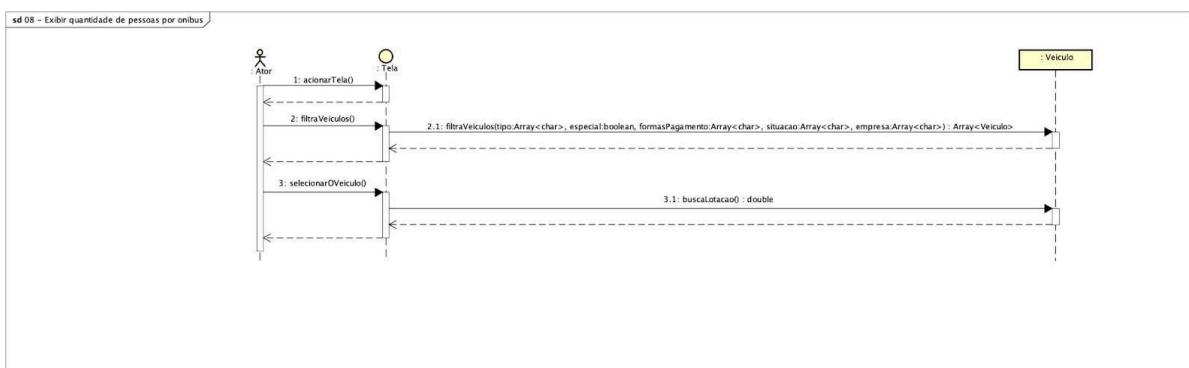


FIGURA 43 - DIAGRAMA DE SEQUÊNCIA - UC09 GERAR RELATÓRIO DE MOVIMENTAÇÃO DAS LINHAS

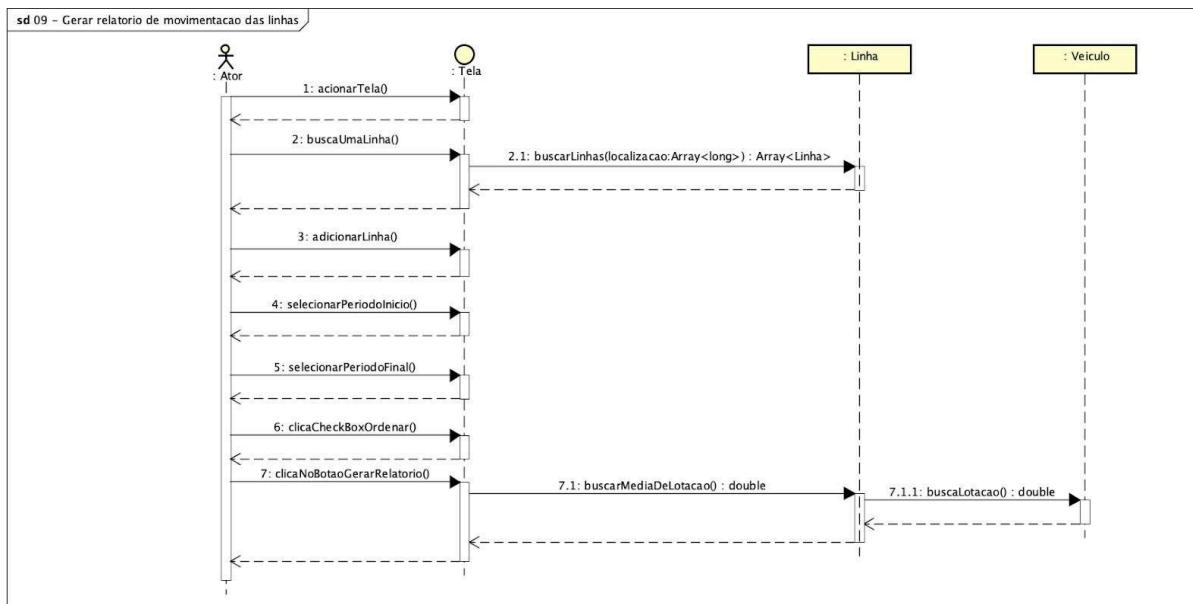


FIGURA 44 - DIAGRAMA DE SEQUÊNCIA - UC10 GERAR INDICADORES DE LOTAÇÃO

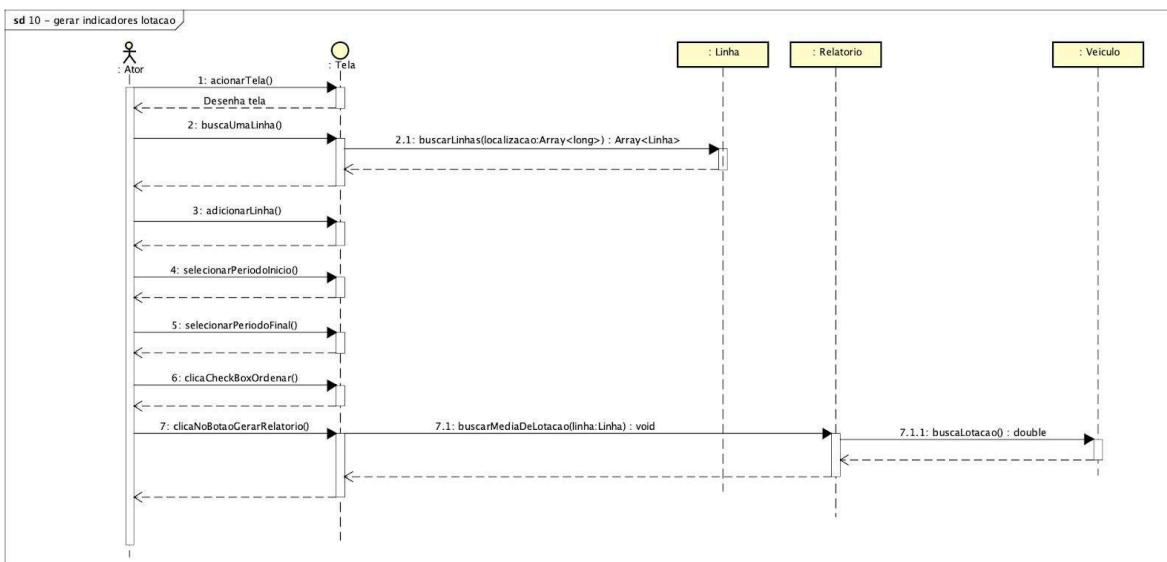


FIGURA 45 - DIAGRAMA DE SEQUÊNCIA - UC10 GERAR INDICADORES DO VEÍCULO

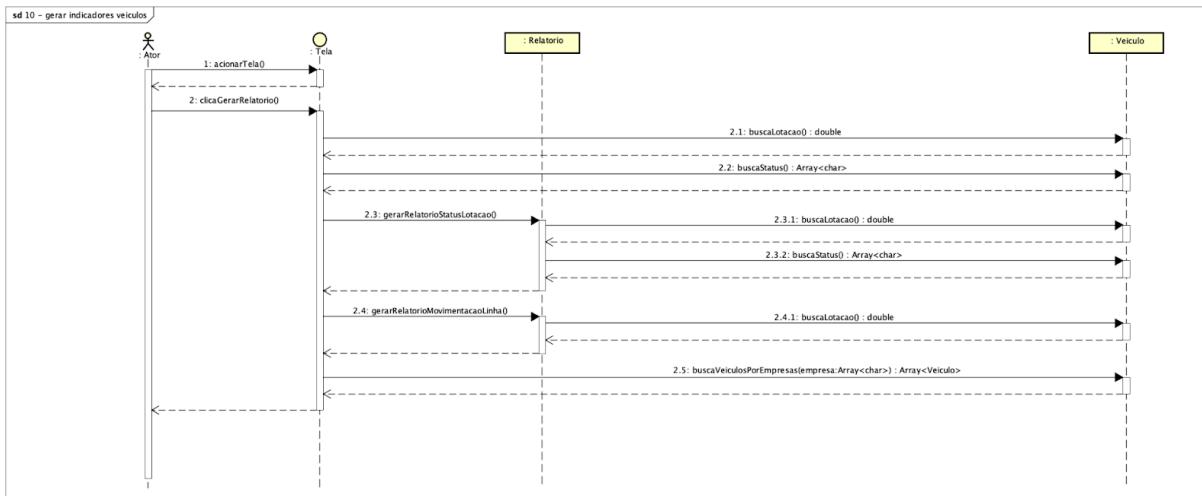


FIGURA 46 - DIAGRAMA DE SEQUÊNCIA APP - UC01 REALIZAR LOGIN

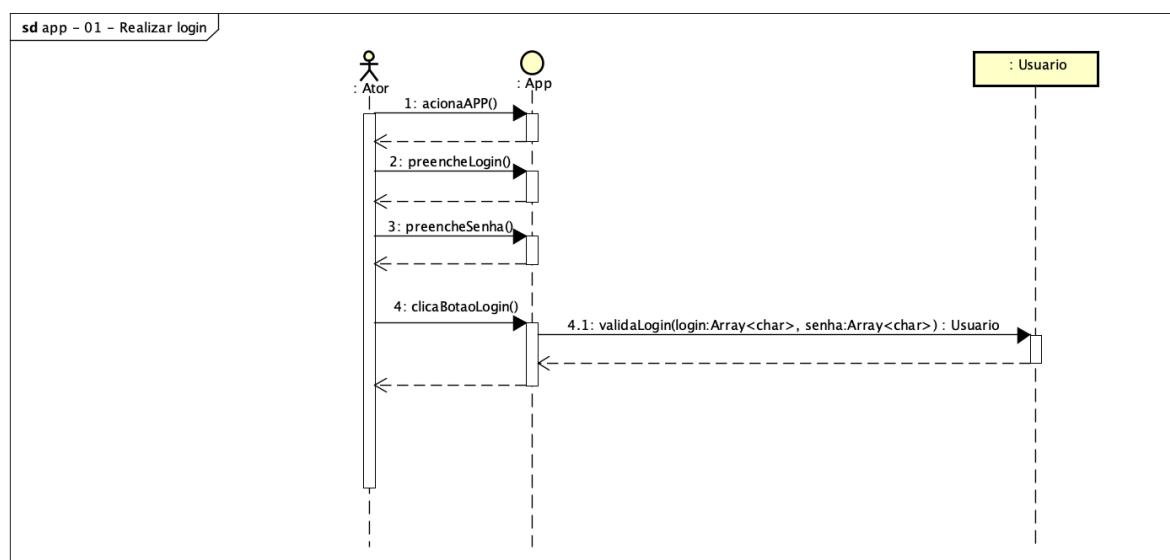


FIGURA 47 - DIAGRAMA DE SEQUÊNCIA APP - UC02 REALIZAR CADASTRO NO SISTEMA

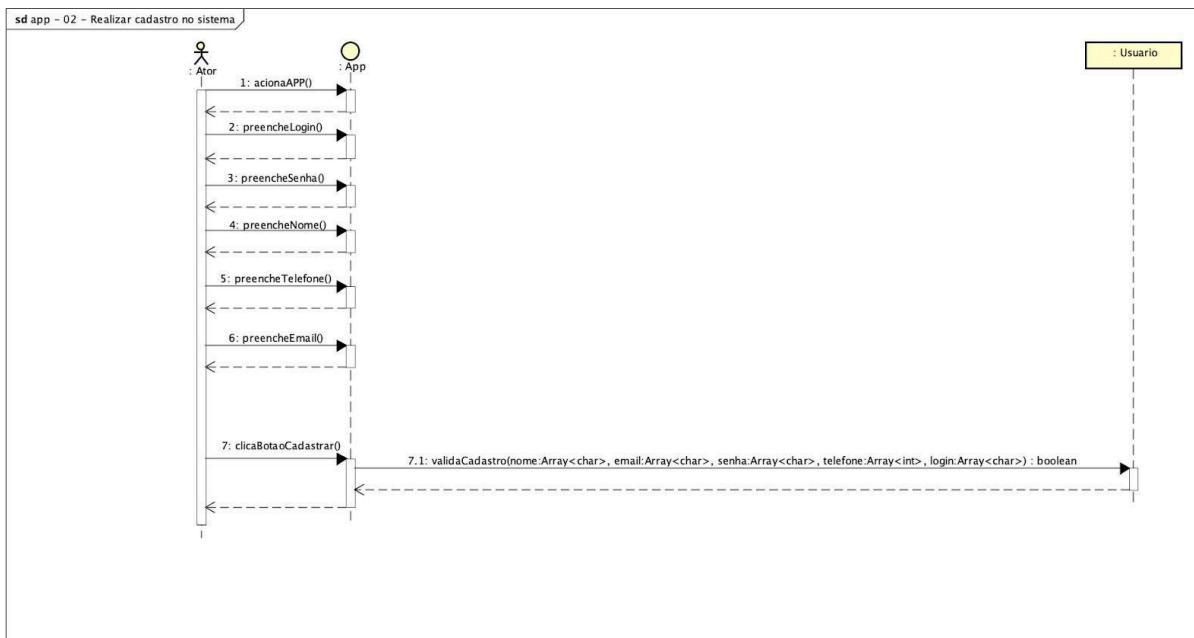


FIGURA 48 - DIAGRAMA DE SEQUÊNCIA APP - UC03 LISTAR LINHAS

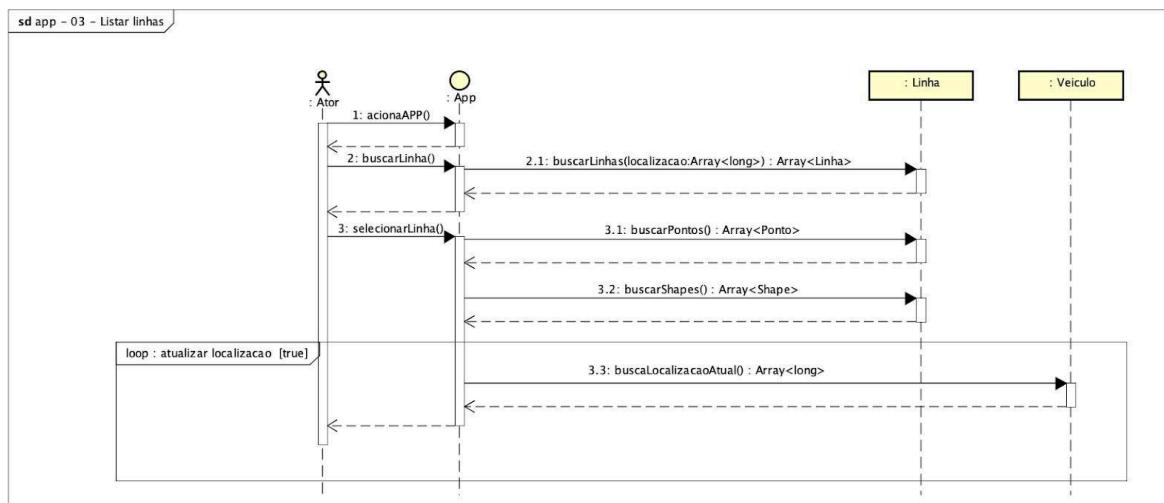


FIGURA 49 - DIAGRAMA DE SEQUÊNCIA - UC04 FAVORITAR LINHAS

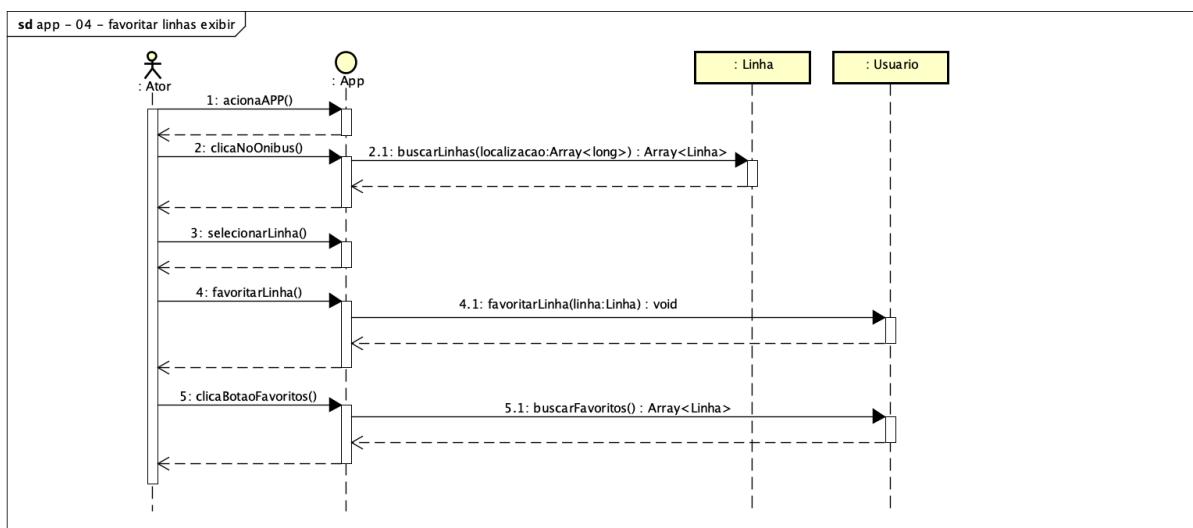


FIGURA 50 - DIAGRAMA DE SEQUÊNCIA APP - UC05 EXIBIR ITINERÁRIO DA LINHA

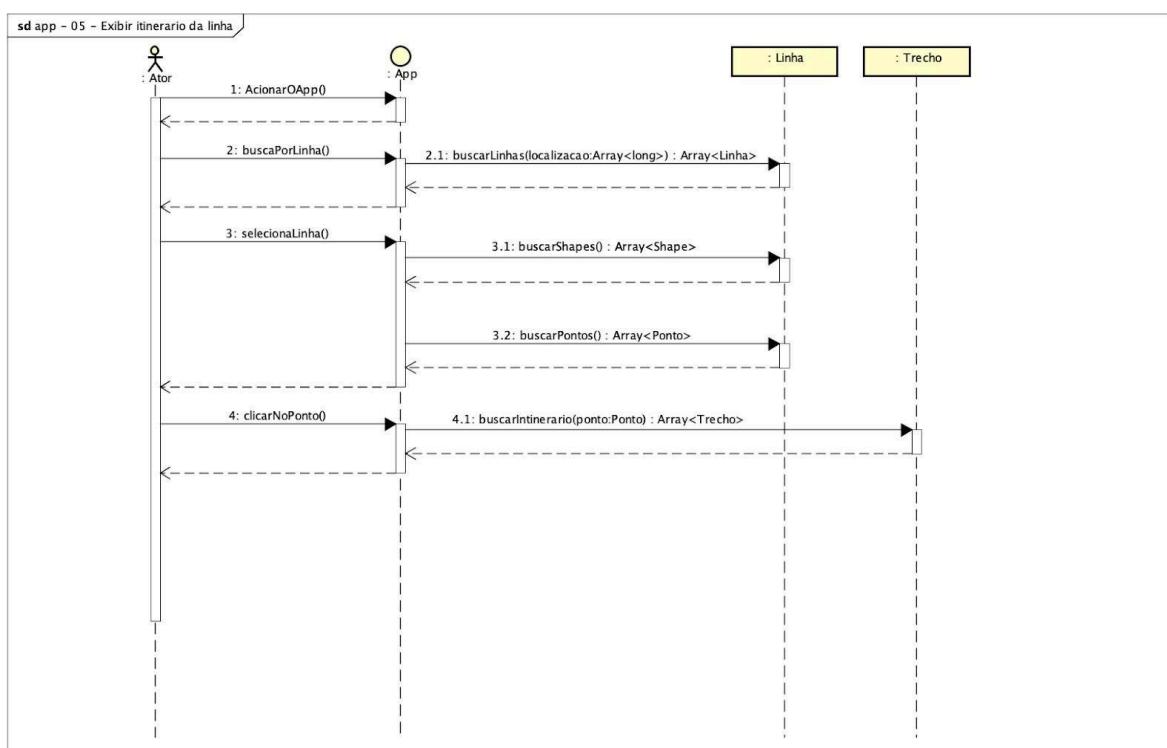


FIGURA 51 - DIAGRAMA DE SEQUÊNCIA APP - UC06 EXIBIR VEÍCULOS EM OPERAÇÃO

