Texto

Descrição gerada automaticamente

Instituto Federal de Educação, Ciência e Tecnologia de São Paulo

Câmpus Birigui

ALINE BERTOLAZO DOS SANTOS

ANDRE LUIZ DA SILVA CONDE

ANDREY MATHEUS BRAMBILLA

ISADORA BUENO DOS SANTOS

VINICIUS DE SOUZA SANTOS

Projeto: BusSpot

**Análise de Projeto e Sistemas**

Birigui 2022

ALINE BERTOLAZO DOS SANTOS

ANDRE LUIZ DA SILVA CONDE

ANDREY MATHEUS BRAMBILLA

ISADORA D BUENO DOS SANTOS

VINICIUS DE SOUZA SANTOS

**Projeto: BusSpot**

Análise de Projeto e Sistemas

Trabalho apresentado ao Instituto Federal de Educação, Ciência e Tecnologia de São Paulo, como requisito parcial para conclusão do curso de Engenharia da Computação.

Área de Concentração: Área de Concentração do Trabalho

Orientador: Prof(a) Helen Freitas dos Santos

Birigui 2022

**FICHA CATALOGRÁFICA (obrigatória; impressa no verso da folha de rosto, não conta na paginação)**

Rissi, Viviane Renata Ventura.

A importância da biblioteca : análise dos últimos 7 anos / Viviane R. V. Rissi, 2019

90 f. : il.

Orientador: Nome direto

Monografia (Graduação)– Instituto Federal de Educação, Ciência e Tecnologia de São Paulo, Birigui, 2019.

1. Leitura (Ensino superior). 2. Livros (Investimento). 3. Leitura – Meios auxiliares.

I. Instituto Federal de Educação, Ciência e Tecnologia. II. Título.

Ficha catalográfica elaborada pela Biblioteca do IFSP – Campus Birigui

# **ERRATA**

ERRATA (OPCIONAL). Aponta erros ortográficos contidos no trabalho, que não foram corrigidos antes da impressão, da seguinte forma: constrói-se uma pequena tabela composta pelas colunas “Folha”, “Linha”, “Onde se lê”, e “Leia-se” e aponta-se nessas colunas a localização do erro, o erro propriamente dito e sua respectiva correção. Encabeçando a errata, deve-se especificar a referência do trabalho.

**FOLHA DE APROVAÇÃO (obrigatório)**

ALINE BERTOLAZO DOS SANTOS

ANDRE LUIZ DA SILVA CONDE

ANDREY MATHEUS BRAMBILLA

ISADORA D BUENO DOS SANTOS

VINICIUS DE SOUZA SANTOS

**PROJETO: BUSSPOT**

Análise de Projeto e Sistemas

Trabalho apresentado ao Instituto Federal de Educação, Ciência e Tecnologia de São Paulo, Câmpus Birigui, como requisito parcial para conclusão do curso de XXXX.

Orientador(a): Prof(a) Helen Freitas dos

Santos

**Banca examinadora**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Membro 1, titulação e instituição

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Membro 1, titulação e instituição

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Membro 1, titulação e instituição

Birigui, de de .

*Dedico este trabalho a Deus; sem ele eu não teria capacidade para desenvolver este trabalho*

# **AGRADECIMENTOS**

A Deus, por ter permitido que eu tivesse saúde е determinação para não desanimar durante a realização deste trabalho.

Aos meus pais e irmãos, que me incentivaram nos momentos difíceis e compreenderam a minha ausência enquanto eu me dedicava à realização deste trabalho.

Aos professores, por todos os conselhos, pela ajuda e pela paciência com a qual guiaram o meu aprendizado.

“*A competitividade de um país não começa nas indústrias ou nos laboratórios de engenharia. Ela começa na sala de aula*”.

Lee Iacocca

# **RESUMO**

Este trabalho apresenta um estudo interdisciplinar acerca do desenvolvimento de produtos, focado no desenvolvimento de um projeto de software que auxilia o uso do transporte coletivo, com o foco em cidades pequenas onde o transporte público usufrui de uma frota menor. Seu objetivo é expor esse mercado de software, referente ao transporte público urbano, que na atualidade se encontra defasado, visando projetar um software que utiliza de técnicas de desenvolvimento como: Levantamento de requisitos e das necessidades do cliente, diagramas na linguagem UML, para desenhar a implantação do produto, prototipação do produto para melhor apresentação do produto ao cliente e alguns paradigmas da análise de projeto. Todas essas funcionalidades estarão disponíveis em um ambiente mobile com acesso individual através da interação com o usuário.

**Palavras-chave:** Projeto. Diagramas. Desenvolvimento.

# **ABSTRACT OU RÉSUMÉ OU RESUMEN**

This work presents an interdisciplinary study on product development, focused on the development of a software project that helps the use of public transport, with a focus on small cities where public transport has a smaller fleet. Its objective is to expose this software market, referring to urban public transport, which is currently outdated, aiming to design a software that uses development techniques such as: Survey of requirements and customer needs, diagrams in the UML language, to design product implementation, product prototyping to better present the product to the customer and some paradigms of project analysis. All these features will be available in a mobile environment with individual access through user interaction.

**Keywords**: Project. diagrams. Development.

# **LISTA DE ILUSTRAÇÕES**

[**Figura 1** Diagrama de Casos de Uso da BusSpot 34](#_Toc103534161)

[**Figura 2** Modelo de Domínio da BusSpot 42](#_Toc103534162)

[**Figura 3** Diagrama de Objeto da BusSpot 43](#_Toc103534163)

[**Figura 4** Diagrama de Classe Análise da BusSpot 44](#_Toc103534164)

[**Figura 5** Diagrama de Atividade da BusSpot 46](#_Toc103534165)

[**Figura 6** Diagrama de Estados da BusSpot 47](#_Toc103534166)

[Figura 7 Tela de Login da BusSpot 33](#_Toc103534167)

[Figura 8 Menu Inicial da BusSpot 34](#_Toc103534168)

[Figura 9 : Tela de Cadastro de Estado da BusSpot 34](#_Toc103534169)

[Figura 10 Tela de Cadastro de Cidade da BusSpot 35](#_Toc103534170)

[Figura 11 Tela de Cadastro de Bairro da BusSpot 35](#_Toc103534171)

[Figura 12 Tela de Cadastro de Tipo de Logradouro da BusSpot 36](#_Toc103534172)

[Figura 13 Tela de Cadastro de Logradouro da BusSpot 36](#_Toc103534173)

[Figura 14 Tela de Cadastro de CEP da BusSpot 37](#_Toc103534174)

[Figura 15 Tela de Cadastro de Empresa da BusSpot 37](#_Toc103534175)

[Figura 16 Tela de Cadastro de Ônibus da BusSpot 38](#_Toc103534176)

[Figura 17 Tela de Cadastro de Parada da BusSpot 39](#_Toc103534177)

[Figura 18 Tela de Cadastro de Ponto da BusSpot 40](#_Toc103534178)

[Figura 19 Tela de Cadastro de Linha da BusSpot 41](#_Toc103534179)

[Figura 20 Tela de Registro de Operação da BusSpot 42](#_Toc103534180)

[Figura 21 Modelo Objeto feito pela IDE SQL SERVER 44](#_Toc103534181)

[Figura 22 Diagrama de Pacote Projeto da BusSpot 37](#_Toc103534182)

[Figura 23 Diagrama de Classe Análise Projeto da BusSpot 39](#_Toc103534183)

[Figura 24 Modelo Projeto Procedimental da BusSpot 40](#_Toc103534184)

[Figura 25 Diagrama de Sequência da BusSpot 41](#_Toc103534185)

[Figura 26 Modelo Projeto Arquitetural da BusSpot 42](#_Toc103534186)

[Figura 27 Visual Studio 44](#_Toc103534187)

[Figura 28 C# 44](#_Toc103534188)

[Figura 29 Adobe Photoshop 45](#_Toc103534189)

[Figura 30 Sql Server 45](#_Toc103534190)

[Figura 31 Word 46](#_Toc103534191)

[Figura 32 Diagrama de Componente da BusSpot 46](#_Toc103534192)

[Figura 33 Diagrama de Implantação da BusSpot 53](#_Toc103534193)

[Figura 34 Exemplo de Caso de Teste de Estado Elaborado na Ferramenta Testlink 54](#_Toc103534194)

[Figura 35 Caso de Teste feito pelo TestLink para a validação de um Estado 55](#_Toc103534195)

[Figura 36 Detalhamento do Caso de Teste de Estado feito pelo TestLink 55](#_Toc103534196)

[Figura 37 Exemplo de Caso de Teste de Cidade Elaborado na Ferramenta Testlink 56](#_Toc103534197)

# LISTA DE QUADROS

[Quadro 1 – Exemplo de quadro](#_heading=h.23ckvvd) 19

# **LISTA DE TABELAS**

[Tabela 1 Requisitos de Funcionais da BusSpot 27](#_Toc103353675)

[Tabela 2 Requisitos Não Funcionais 32](#_Toc103353676)

[Tabela 3 Especificação do Caso de Uso de Cadastra Empresa 35](#_Toc103353677)

[Tabela 4 Especificação do Caso de Uso de login 36](#_Toc103353678)

[Tabela 5 Especificação do Caso de Uso de Cadastra de Ônibus 37](#_Toc103353679)

[Tabela 6 Especificação do Caso de Uso de Localização 38](#_Toc103353680)

[Tabela 7 Especificação do Caso de Uso de Tempo Real 39](#_Toc103353681)

[Tabela 8 Especificação do Caso de Uso de Registro de Reclamação 40](#_Toc103353682)

[Tabela 9 Especificação do Caso de Uso de Consulta de Reclamação 41](#_Toc103353683)

[Tabela 10 Estrutura da tabela Estado da BusSpot 46](#_Toc103353684)

[Tabela 11 Estrutura da tabela Cidade da BusSpot 46](#_Toc103353685)

[Tabela 12 Estrutura da tabela Bairro da BusSpot 46](#_Toc103353686)

[Tabela 13 Estrutura da tabela Logradouro da BusSpot 47](#_Toc103353687)

[Tabela 14 Estrutura da tabela Tipo de Logradouro da BusSpot 47](#_Toc103353688)

[Tabela 15 Estrutura da tabela Endereço da BusSpot 47](#_Toc103353689)

[Tabela 16 Estrutura da tabela Empresa da BusSpot 48](#_Toc103353690)

[Tabela 17 Estrutura da tabela Reclamação da BusSpot 48](#_Toc103353691)

[Tabela 18 Estrutura da tabela Parada da BusSpot 49](#_Toc103353692)

[Tabela 19 Estrutura da tabela Ponto da BusSpot 49](#_Toc103353693)

[Tabela 20 Estrutura da tabela Linha da BusSpot 49](#_Toc103353694)

[Tabela 21 Estrutura da tabela Embarque da BusSpot 50](#_Toc103353695)

[Tabela 22 Estrutura da tabela Comodidade da BusSpot 50](#_Toc103353696)

[Tabela 23 Estrutura da tabela Acessibilidade da BusSpot 51](#_Toc103353697)

[Tabela 24 Estrutura da tabela Pagamento da BusSpot 51](#_Toc103353698)

[Tabela 25 Estrutura da tabela Ônibus da BusSpot 51](#_Toc103353699)

[Tabela 26 Estrutura da tabela Localização da BusSpot 52](#_Toc103353700)

[Tabela 27 Estrutura da tabela Operação da BusSpot 52](#_Toc103353701)

# **LISTA DE ABREVIATURAS E SIGLAS**

UML *Unified Modeling Language* - Linguagem de Modelagem Unificada.

RF/RNF Requisito Funcional / Requisito Não Funcional.

3D Três dimensões.

# **LISTA DE SÍMBOLOS**

*α* Letra grega minúscula Alfa

*β* Letra grega minúscula Beta

**Sumário**

[ERRATA 4](#_Toc104743687)

[AGRADECIMENTOS 7](#_Toc104743688)

[RESUMO 9](#_Toc104743689)

[ABSTRACT OU RÉSUMÉ OU RESUMEN 10](#_Toc104743690)

[LISTA DE ILUSTRAÇÕES 11](#_Toc104743691)

[LISTA DE QUADROS 12](#_Toc104743692)

[LISTA DE TABELAS 13](#_Toc104743693)

[LISTA DE ABREVIATURAS E SIGLAS 14](#_Toc104743694)

[LISTA DE SÍMBOLOS 15](#_Toc104743695)

[1 INTRODUÇÃO 18](#_Toc104743696)

[**1.1** **Justificativa** 18](#_Toc104743697)

[**1.2** **Objetivos** 19](#_Toc104743698)

[1.2.1 Objetivo Geral 19](#_Toc104743699)

[1.2.2 Objetivos Específicos 19](#_Toc104743700)

[**1.3** **Organização Deste Trabalho** 19](#_Toc104743701)

[2 REVISÃO DA LITERATURA 20](#_Toc104743702)

[3 MATERIAIS E MÉTODOS 23](#_Toc104743703)

[4 DESCRIÇÃO GERAL DO PRODUTO 24](#_Toc104743704)

[5 ELICITAÇÃO DE REQUISITOS E ANÁLISE 25](#_Toc104743705)

[**5.1** **Requisitos do Usuário** 25](#_Toc104743706)

[**5.2** **Requisitos do Sistema** 27](#_Toc104743707)

[5.2.1 Requisitos Funcionais 27](#_Toc104743708)

[5.2.2 Requisitos Não-Funcionais 33](#_Toc104743709)

[5.2.3 Restrições, Suposições e Dependências 33](#_Toc104743710)

[5.2.4 Requisitos Adiados 33](#_Toc104743711)

[**5.3** **Casos de Uso** 34](#_Toc104743712)

[5.3.1 Diagrama de Casos de Uso 34](#_Toc104743713)

[5.3.2 Especificação dos Casos de Uso 35](#_Toc104743714)

[**5.4** **Modelo de Domínio** 43](#_Toc104743715)

[**5.5** **Diagrama de Objetos** 44](#_Toc104743716)

[**5.6** **Diagrama de Classes de Análise** 45](#_Toc104743717)

[**5.7** **Diagrama de Atividades** 46](#_Toc104743718)

[**5.8** **Diagrama de Estados** 48](#_Toc104743719)

[6 PROJETO DE SOFTWARE 32](#_Toc104743720)

[**6.1** **Projeto de Interface** 32](#_Toc104743721)

[6.1.2 Interface da Empresa 32](#_Toc104743722)

[1.1.3 Interface Usuário 42](#_Toc104743723)

[**6.2** **Projeto de Dados** 47](#_Toc104743724)

[6.2.1 Mapeamento Objeto-Relacional 48](#_Toc104743725)

[6.2.2 Estrutura das Tabelas no Banco de Dados 49](#_Toc104743726)

[6.2.3 Diagrama de Pacotes 37](#_Toc104743727)

[6.2.4 Diagrama de Classes de Projeto 38](#_Toc104743728)

[**6.3** **Projeto Procedimental** 39](#_Toc104743729)

[6.3.1 Diagrama de Sequência 40](#_Toc104743730)

[**6.4** **Projeto Arquitetural** 41](#_Toc104743731)

[7 IMPLEMENTAÇÃO 42](#_Toc104743732)

[7.1 Ambiente de Desenvolvimento. 42](#_Toc104743733)

[7.1.1 Ferramentas e Linguagens utilizadas 42](#_Toc104743734)

[7.1.2 Diagrama de Componente 45](#_Toc104743735)

[7.2 Coleta de Dados 46](#_Toc104743736)

[7.2.1 Coleta de Dados dos Terminais e paradas de ônibus 46](#_Toc104743737)

[7.2.2 Coleta de Dados e amostra dos ônibus 48](#_Toc104743738)

[7.3 Desafios Enfrentados 51](#_Toc104743739)

[7.3.1 Terminais 52](#_Toc104743740)

[7.3.2 Feriado 52](#_Toc104743741)

[7.3.3 Meia Viagem 52](#_Toc104743742)

[7.4 Diagrama de Implantação 52](#_Toc104743743)

[8 TESTE 54](#_Toc104743744)

[8.1 Plano de Teste de Cadastros 54](#_Toc104743745)

[8.1.1 Plano de Teste de Estado 54](#_Toc104743746)

[8.1.3 Plano de Teste de Reclamação 57](#_Toc104743747)

[09 RESULTADOS E DISCUSSÃO 59](#_Toc104743748)

[REFERÊNCIAS 60](#_Toc104743749)

# **INTRODUÇÃO**

Dada a situação atual com a adesão de empresas de transporte coletivo em cidades pequenas, é necessário implementar o maior número possível de práticas para aumentar produtividade e melhorar o aproveitamento do serviço de transporte, aumentando a vantagem competitiva das empresas no mercado.

No cenário atual utilizamos como base a cidade de Birigui localizada no Estado de São Paulo, que atualmente vem enfrentando problemas na gestão de transporte urbano devido algumas quebras de contrato de prestação de serviço conforme artigo da SBT interior, pensando em melhorar o sistema de transporte e proporcionar para a população uma utilização mais clara e prazerosa do uso de coletivos que é dever do Estado disponibilizar para os habitantes da cidade, foi pensado no desenvolvimento de um software primeiramente no modo mobile para uma interação mais aberta entre as empresas de transporte e seus usuários.

O Projeto B.S tem como principal objetivo usar ferramentas para os usuários do sistema de transporte coletivo possa ter um melhor aproveitamento do sistema público podendo monitorar sua rota e utilizar rotas mais rápidas para seus destinos, poderá também acompanhar os horários programados da sua rota para não perder seu ônibus e poderá ajudar as empresas de ônibus a identificar falhas em seus serviços com a função de feedback ou reclamação.

## **Justificativa**

Existem duas questões determinantes na escolha do referido tema para o Trabalho de Conclusão de Curso.

Em primeiro lugar, o interesse dos autores em aprofundar seus conhecimentos de práticas de desenvolvimento de sistemas computacionais voltados para ambiente mobile. O campo está se expandindo em toda a linha e está mostrando uma grande promessa aqueles que o dominam.

Além disso foi chegado a conclusão que, devido aos problemas na gestão do transporte público que a cidade de Birigui tem passado, a população da cidade vai passar a ter mais problema ao utilizar seu coletivo e com isso a proposta do projeto é desenvolver uma aplicação para tornar a utilização desse serviço mais ágil e simplificada, sendo que também será possível usar as informações de horário de viagens das rotas para determinar se há atraso ou adiantamento nas rotas podendo assim usar isso para cobrar melhor serviço ao órgão fiscalizador.

## **Objetivos**

A seguir serão apresentados os objetivos gerais e específicos do trabalho.

### Objetivo Geral

Desenvolver um software que permita gerenciar a localização dos coletivos e os seus respectivos horários, possibilitando fiscalizar fisicamente o serviço de transporte urbano.

### Objetivos Específicos

Dentre os objetivos específicos do trabalho, destacam-se:

* Realizar a análise da documentação de requisitos, identificando as características essenciais para desenvolver a estrutura básica do software.
* Aplicar as técnicas e linguagem UML para desenhar melhor como será a estrutura da aplicação.
* Protótipo do projeto para ter uma melhor visualização de como será a parte visual do projeto.
* Aplicar o software em ambiente real para verificação do comportamento.

## **Organização Deste Trabalho**

Esse trabalho será divido da seguinte forma: capítulo 1 a introdução do trabalho, capítulo 2 a Revisão da literatura, capítulo 3 onde será apresentado os materiais e os métodos deste trabalho, capítulo 4 será feita a descrição do produto a ser desenvolvido, no capítulo 5 será proposto toda a análise do projeto desde os diagramas de comportamento a engenharia de requisitos, no capítulo 6 será mostrado o projeto em si com seu protótipo e seus diagramas estruturais, no capítulo 7 será apresentado a implementação do software, capítulo 8 será a fase de teste, no capítulo 9 será a implantação da aplicação, capítulo 10 será discutido os resultados obtidos, no capítulo 11 será abordado trabalho relacionados, no capítulo.

# **REVISÃO DA LITERATURA**

Texto da revisão da literatura, dividido em seções e subseções.

Este é um exemplo de como usar figuras. Referência cruzada: Figura [1](#_heading=h.147n2zr)

Figura 1 – Exemplo de figura

Texto

Descrição gerada automaticamente

**Fonte:** Elaborada pelo autor

Este é um exemplo de como usar tabelas. Referência cruzada: Tabela [1](#_heading=h.3o7alnk) Tabela 1 – Exemplo de tabela de 2 colunas

|  |  |
| --- | --- |
| **Coluna 1** | **Coluna 2** |
| Dado 1a | Dado 2a |
| Dado 1b | Dado 2b |
| Dado 1c | Dado 2c |
| Dado 1d | Dado 2d |

**Fonte:** Elaborada pelo autor

Este é um exemplo de como usar quadros. Referência cruzada: Quadro [1](#_heading=h.23ckvvd) Quadro 1 – Exemplo de quadro

Tabela

Descrição gerada automaticamente

**Fonte:** Elaborada pelo autor

Este é um exemplo de como usar equações. Referência cruzada: Equação [2.1](#_heading=h.ihv636)

Exemplo de inserção de lista de código fonte (**não use acentos no código!**):

2 \* C lasse de ex em p lo .

3 \*

4 \* @ au th o r David B u zatto

5 \*/

6 p u b lic class Grafo {

7

8 p u b lic static void main ( S trin g [] args ) {

9

1. S y stem . out . p rin tln ( " E x em p lo de co d ig o fonte !" );
2. S y stem . out . p rin tln ( " Nao use acen to s !" );

12

13 }

14

15 }

Este é um exemplo de como inserir texto sem formatação (ambiente verbatim):

Texto sem formatação, como espaçamento igual.

Exemplo de lista de itens:

* + **Item 1:** texto...;
  + **Item 2:** texto...;
    - **Subitem:** texto...;
    - **Subitem:** texto...;
    - **Subitem:** texto...;
  + **Item 3:** texto...;
  + **Item n:** texto....

Exemplo de lista numerada:

1. **Item:** texto...;
2. **Item:** texto...;
   1. **Subitem:** texto...;
   2. **Subitem:** texto...;
3. **Item:** texto...;
4. **Item:** texto....

Exemplos de comandos para texto e referências:

* + Para iniciar um novo parágrafo, basta deixar uma linha em branco no código fonte;
  + Não force o compilador a pular mais de uma linha, pois terá influência negativa na composição do documento;
  + Sempre deixe o LATEX realizar a formatação de parágrafos e posicionamento de elementos;
  + Utilização de aspas simples (abertura ‘, fechamento ’): ‘Texto entre aspas simples’;
  + Utilização de aspas duplas (abertura ‘‘, fechamento ’’): “Texto entre aspas duplas”;
  + Negrito (comando \textbf): **texto em negrito**;
  + Itálico (comando \textit): *texto em itálico*;
  + Sublinhado (comando \underline): texto sublinhado;
  + Negrito e itálico (usar comandos juntos): ***texto em negrito e itálico***;
  + Alterar cor do texto (comando \textcolor{cor}{texto}):
    - Exemplo \textcolor{red}{texto}: texto vermelho;
    - Exemplo \textcolor[RGB]{255, 102, 0}: texto laranja;
    - Exemplo \textcolor[HTML]{006AD7}: texto azul;
  + Ambiente matemático inline (comando $ expressão $): *s* = *x*2 *−* 2*x* + 1;
  + Referência normal (comando \cite):
    - ([AGAISSE; LERECLUS](#_heading=h.14ykbeg), [1995](#_heading=h.14ykbeg));
    - ([ABEDI et al.](#_heading=h.2pta16n), [2014](#_heading=h.2pta16n));
    - ([CRICKMORE et al.](#_heading=h.3ls5o66), [2016](#_heading=h.3ls5o66));
  + Referência normal com mais de uma obra (comando \cite):
    - ([ABEDI et al.](#_heading=h.2pta16n), [2014](#_heading=h.2pta16n); [AGAISSE; LERECLUS](#_heading=h.14ykbeg), [1995](#_heading=h.14ykbeg));
    - ([AGAPITO-TENFEN et al.](#_heading=h.20xfydz), [2014](#_heading=h.20xfydz); [CRICKMORE et al.](#_heading=h.3ls5o66), [2016](#_heading=h.3ls5o66); [NELSON; COX](#_heading=h.4kx3h1s), [2014](#_heading=h.4kx3h1s));
  + Referência nome e ano (comando \citeauthorandyear):
    - [Agaisse e Lereclus](#_heading=h.14ykbeg) ([1995](#_heading=h.14ykbeg));
    - [Abedi et al.](#_heading=h.2pta16n) ([2014](#_heading=h.2pta16n));
    - [Crickmore et al.](#_heading=h.3ls5o66) ([2016](#_heading=h.3ls5o66)); Exemplo 1 de citação direta:

Os 20 aminoácidos usualmente encontrados como resíduos em proteínas contém um grupo *α*-carboxil, um grupo *α*-amino e um grupo R distinto substituído no átomo de carbono *α*. O átomo de carbono *α* de todos os aminoácidos, com exceção da glicina, é assimétrico e, portanto, os aminoácidos podem existir em pelo menos duas formas estereoisoméricas. Somente os estereoisômeros L, com uma configuração relacionada à configuração absoluta da molécula de referência L-gliceraldeído, são encontrados em proteínas ([NELSON; COX](#_heading=h.4kx3h1s), [2014](#_heading=h.4kx3h1s), p. 81).

Exemplo 2 de citação direta:

*These various insecticidal proteins are synthesized during the stationary phase and accumulate in the mother cell as a crystal inclusion which can account for up to 25% of the dry weight of the sporulated cells. The amount of crystal protein produced by a B. thuringiensis culture in laboratory conditions (about 0.5 mg of protein per ml) and the size of the crystals (24) indicate that each cell has to synthesize* 106 *to* 2 106 *δ-endotoxin molecules during the stationary phase to form a crystal* ([AGAISSE; LERECLUS](#_heading=h.14ykbeg), [1995](#_heading=h.14ykbeg), p. 1).

*×*

Exemplo de nota de rodapé1.

1 Essa é uma nota de rodapé!

# 3 **MATERIAIS E MÉTODOS**

Os materiais propostos para a elaboração desse projeto consistem em uma plataforma com a linguagem de programação em C#, pois a aplicação será feita nessa linguagem será usado como base o OpenStreeMap em questão de custo ele foi escolhido para compor a parte de mapeamento da plataforma.   
 O método para desenvolver esse projeto será a orientação a objeto pois como serão elaborados diagramas UML’s no projeto, esses diagramas auxiliaram no desenvolvimento, a gestão do projeto vai feita com a documentação padrão como TAP que nele já e colocado o escopo e o custo do projeto e o software será desenvolvido com a metodologia Scrum e com 5 ciclos de vida (Análise, Desenho, Implementação, Teste e Implantação).

# 4 **DESCRIÇÃO GERAL DO PRODUTO**

O BusSpot seria um projeto em aplicação que propõem trazer na tela do seu smartphone a localização dos coletivos da linha requisitada em tempo real para que o usuário tenha em mente quando e onde seu transporte mais próximo está, e não seria só isso, o aplicativo também trará informações importantes do coletivo como seu prefixo (Caso precise fazer uma denúncia, elogio ou sugestão para as empresas de ônibus referente aquele coletivo que estava usando), suas informações básica como: Acessibilidade, se possui tecnologias como wi fi e ar-condicionado e também te disponibilizará a possibilidade de informar no aplicativo se o transporte que está utilizando está nas devidas condições básica de uso, tendo itens para avaliação como limpeza, funcionamento do sistema de ar-condicionado ou falha no sistema de acessibilidade, com isso também terá funções extras como tabela de horário das linhas, disponibilidade em dia útil e finais de semana e horários previsto de saída e chegada nos destinos e manual do passageiro caso o usuário se depare com alguma situação inconveniente e não saiba o que fazer, como: tarifa abusiva, má conduta do motorista ou cobrador, assédio e várias outras situações.

A fim de atender o maior número possível de dispositivos, foi adotada a estratégia de desenvolvimento multiplataforma baseada em tecnologias mobile, que permite utilizar o mesmo código para diferentes sistemas operacionais, como Android, iOS, e Windows Phone.

Este assunto vem ganhando a atenção gradual dos desenvolvedores nos últimos anos, de forma que novas práticas e tecnologias surgem e são aperfeiçoadas com frequência.

O protótipo é orientado pela simplicidade e facilidade de utilização. A ideia é fornecer a informação certa com a menor quantidade de interação possível. Para isso, não se optou pela visualização padrão deste tipo de aplicativo que é o mapa com as rotas traçadas. Em vez disso, o aplicativo identifica a localização do usuário e mostra as linhas do ponto de ônibus mais próximo.

# **ELICITAÇÃO DE REQUISITOS E ANÁLISE**

Nessa parte do trabalho será tratado a Engenharia de Requisitos. A engenharia de requisitos é um processo de composição Por meio de atividades de alto nível, considere uma revisão do processo de requisitos, Revise as necessidades do usuário e esclareça as necessidades do usuário usuários, atribuições de tarefas, documentação e revisão de requisitos, Revise os requisitos do usuário e vá ao ar de acordo com. Com base nisso, a engenharia de requisitos é considerada um documento requisito para sua manutenção (Sommerville, 2001).

## **Requisitos do Usuário**

Os requisitos do usuário descrevem necessidades de um grupo específico de partes interessadas em relação à solução proposta. Eles podem ser usados para descrever como um conjunto particular de usuários de uma solução irão interagir com ela e como um produto irá atender as necessidades de diferentes grupos de clientes.

Com o problema do transporte coletivo nas cidades pequenas, presenciamos a falta de comprometimento das empresas em cumprirem seus horários tabelados e servir seus clientes.

Seria necessário implementar o maior número possível de práticas para melhor produtividade e aproveitamento do serviço de transporte coletivo, o qual muitas vezes se encontra fora de seus horários e necessita de um meio para administrá-los, visando aumentar as vantagens competitivas das empresas no mercado.

Utilizamos como base a cidade de Birigui, localizada no Estado de São Paulo, que atualmente vem enfrentando problemas na gestão de transporte urbano, devido a algumas quebras de contrato de prestação de serviço, conforme o artigo publicado no SBT interior ². Pensando em facilitar o acesso ao transporte público pela população, foi proposto o desenvolvimento de um software que seja de benefício para as empresas de transporte urbano e para o público usuário de transporte coletivo. Esse software tem como seu principal objetivo tornar o controle das operações de transporte feita pelas empresas de ônibus, com ferramentas para monitorar a frota e fiscalizar o cumprimento de partidas e facilitar o usuário a saber onde seu transporte público mais próximo se encontra e ajudar o usuário a se sentir mais seguro e confiante com o uso da nossa ferramenta de reclamação. Entre outras funções que foi proposta para desenvolvimento do projeto segue a lista de requisitos descrita pelo usuário.

Os requisitos identificados para esse sistema são:

1. Cadastrar empresa, sendo necessários os seguintes dados: nome, CNPJ, endereço, usuário e senha.

2. Cadastrar parada de ônibus, sendo necessários os seguintes dados: o número do ônibus, acessibilidade, comodidade, embarque e pagamento.

3. Cadastrar trajeto de ônibus, sendo necessários os seguintes dados: comprimento, o ponto inicial, as paradas e o ponto final.

4. Cadastrar linhas de ônibus, sendo necessários os seguintes dados: número da linha, horários das viagens, o ponto inicial, as paradas e o ponto final.

5. Cadastrar ônibus das empresas, sendo necessários os seguintes dados: o número do ônibus, acessibilidade, comodidade, embarque e pagamento.

6. Registrar destino desejado através das paradas requeridas pelo usuário.

7. Apresentar um mapa contendo a localização em tempo real dos ônibus rodando na cidade. (O usuário terá que selecionar previamente a linha desejada e o sistema apontarão todos os ônibus que esteja circulando apenas nessa linha, terá também uma opção de ele definir uma rota que dependendo da distância o sistema mostrar várias linhas de várias empresas, mas somente mostrará os ônibus e as linhas que compõem o trajeto que o usuário pretende percorrer);

8. Visualizar a quantidade de ônibus em cada linha, devendo ser apresentados os seguintes dados: ( Linha dos ônibus e o seu total, isso terá mais serventia para as empresas de ônibus saber se realmente ela está cumprindo com toda a programação dela.);

9. Visualizar o tempo médio das viagens (O sistema irá utilizar as viagens programadas pela empresa e comparar com as viagens registadas pelo rastreador e contabilizada pelo busspot);

10. Visualizar as Linhas possíveis de seus trajetos (O sistema irá mostrar todas as linhas possíveis que o usuário poderá utilizar em seu trajeto, como pré-requisito terá que informar ao sistema a origem eo destino);

11. Cadastrar reclamação, sendo necessários os seguintes dados: descrição da ocorrência, número do ônibus, número da linha, data da ocorrência e hora da ocorrência.

12. O sistema também proporciona às empresas: quantidade de reclamações de usuários, cumprimento real das viagens programadas, e um monitoramento dos seus veículos em caso de ocorrências como roubo/assalto entre outros.

13. O Usuário poderá informar sua localização através do uso da sua geolocalização vindo de seu smartphone e com isso não precisa inserir seu destino de origem manualmente e terá a opção de visualizar-se no mapa junto com os coletivos.

14. O sistema permitirá registro de operação sendo possível relacionar o ônibus a uma linha diariamente para controle de viagens.

## **Requisitos do Sistema**

Requisitos de Software são as ações que o software deve executar, possuindo características e condições próprias, de forma a automatizar uma tarefa de um processo de negócio. Aqui definimos os requisitos funcionais e não funcionais, e, conforme o método IRON, Requisitos de Dados e Regras de Execução. ( Eduardo Castro, 2017).

### Requisitos Funcionais

Segundo Benardi, Fontoura e Cordenonsi (2008), a análise de necessidades e definir o que o sistema deve fazer, suas propriedades desejáveis ​​emergentes e fundamentos e limitações da operação e processos do sistema desenvolvimento de software. Os requisitos podem ser divididos de acordo com as notas os detalhes estão em:

Requisitos Funcionais: Corresponde a uma lista de tudo o que o sistema deve fazer. São declarações do que o sistema deve fornecer, como deve responder uma entrada específica e como ela deve se comportar em determinadas situações. eles podem, Incluindo declarar o que o sistema não deve fazer. Abaixo estão os requisitos Funções do sistema.

Os requisitos funcionais identificados para este sistema estão apresentados nas Tabelas a seguir

**Tabela 1** Requisitos de Funcionais da BusSpot

|  |  |  |
| --- | --- | --- |
| **Nº** | **Descrição** | **Ator** |
| RF 1 | O sistema deve permitir o cadastro, a alteração e a remoção de um **estado.** Tendo como atributos   * nome (String), * Sigla (String).   Todos os atributos são obrigatórios, e não pode registrar dois Estado com a mesma sigla e nem com o mesmo nome | Administrador do Sistema |
| RF 2 | O sistema deve permitir o cadastro, a alteração e a remoção de uma **cidade.** Tendo como atributos:   * nome (String); * estado (String - Cadastrado previamente);   Todos os atributos são obrigatórios, e não pode registrar duas cidades com o mesmo nome associado no mesmo Estado. | Administrador do Sistema |
| RF 3 | O sistema deve permitir o cadastro, a alteração e a remoção de um **bairro.** Tendo como atributos:   * nome (String).   Todos os atributos são obrigatórios. | Administrador do Sistema |
| RF 4 | O sistema deve permitir o cadastro, a alteração e a remoção de um **logradouro.** Tendo como atributos:   * nome (String).   Todos os atributos são obrigatórios. | Administrador do Sistema |
| RF 5 | O sistema deve permitir o cadastro, a alteração e a remoção de um **tipo de logradouro**. Tendo como atributos:   * tipo de logradouro (String - EX: Rua, Avenida, Travessa, etc.) .   Todos os atributos são obrigatórios. | Administrador do Sistema |
| RF 6 | O sistema deve permitir o cadastro, a alteração e a remoção de um **endereço.** Tendo como atributos:   * CEP (String); * bairro (Bairro - Cadastrado previamente); * Tipo de Logradouro (TipoLogradouro - Cadastrado previamente); * Logradouro (Logradouro - Cadastrado previamente); * Cidade (Cidade - Cadastrado previamente); * Estado (Estado - Cadastrado previamente).   Todos os atributos são obrigatórios .Caso haja CEP único o usuário terá que inserir os dados manualmente nos sistemas | Empresa |
| RF 7 | O sistema deve permitir o cadastro, a alteração e a remoção de uma **empresa.** Tendo como atributos:   * fantasia (String); * CNPJ (String); * endereço (Endereco - Cadastrado previamente) * número do endereço ( String); * usuário (String); * senha (String).   Todos os atributos são obrigatórios, mas não pode cadastrar duas empresas com o mesmo CNPJ. | Administrador do Sistema |
| RF 8 | O sistema deve permitir o **login** da empresa, verificando se os dados informados:   * usuário (String); * senha (String).   São válidos. Todos os atributos são obrigatórios, mas não poderá cadastrar dois usuários com a mesma Sting. | Empresa |
| RF 9 | O sistema deve permitir o cadastro, a alteração e a remoção das **paradas.** Tendo como atributos:   * nome (String); * endereço (Endereco - Cadastrado previamente) * latitude (String) * longitude (String); * tipo de terminais (String - principal ou secundário).   Todos os atributos são obrigatórios, mas não pode ter duas paradas com o mesmo endereço. | Empresa |
| RF 10 | O sistema deve permitir o cadastro, a alteração e a remoção de um **ponto.** Tendo como atributos:   * Tipo de Ponto (String).   Todos os atributos são obrigatórios. | Empresa |
| RF 11 | O sistema deve permitir o registro, a alteração e a remoção das **comodidades**. Tendo como atributos se possui:   * ar-condicionado (Boolean); * wifi (Boolean); * tomada/USB(Boolean); * banheiro (Boolean); * itens à venda (Boolean); * cadeira reclinável (Boolean).   Todos os atributos são obrigatórios. | Empresa |
| RF 12 | O sistema deve permitir o registro, a alteração e a remoção da **acessibilidade.** Tendo como atributos:   * faixas guias (String); * elevadores para cadeiras (String); * intérprete de libras (String).;   Todos os atributos são obrigatórios. | Empresa |
| RF 13 | O sistema deve permitir o registro, a alteração e a remoção das formas de **embarque.** Tendo como atributo:   * tipo de embarque (String - Catraca, bilhete etc.).   Todos os atributos são obrigatórios. | Empresa |
| RF 14 | O sistema deve permitir o registro, a alteração e a remoção das formas de **pagamento.** Tendo como atributo:   * tipo de pagamento (String - Dinheiro, cartão).   Todos os atributos são obrigatórios. | Empresa |
| RF15 | O sistema deve permitir o cadastro, a alteração e a remoção de um **ônibus.** Tendo como atributos:   * número do ônibus (String); * acessibilidade (Acessibilidade - Cadastrado previamente); * comodidade (Comodidade - Cadastrado previamente); * embarque (Embarque - Cadastrado previamente); * pagamento (Pagamento - Cadastrado previamente).   Todos os atributos são obrigatórios, mas dois ônibus não podem possuir o mesmo número. | Empresa |
| RF16 | O sistema deve permitir o cadastro, a alteração e a remoção de uma **linha.** Tendo como atributos:   * número da linha (String); * descrição da linha (String); * ônibus (String); * horário de partida (Date Time); * horário de chegada (Date Time); * paradas (Parada - Cadastrado previamente); * pontos (Ponto - Cadastrado previamente).   São todos os atributos obrigatórios, mas duas linhas não podem possuir o mesmo número. | Empresa |
| RF17 | O sistema deve ser capaz de reconhecer a **localização** do ônibusa partir dos dados enviados pelo rastreador, sendo eles:   * data e hora (Date Time); * latitude (String); * longitude(String).   Todos os atributos são obrigatórios. | Sistema |
| RF18 | O sistema deve registrar o **tempo real** de viagem de cada ônibus em cada linha, sendo isto um método. Utilizando como parâmetro:   * viagens registradas nas linhas, comparando com as viagens registradas pelo rastreador.   Todos os atributos são obrigatórios. | Sistema |
| RF19 | O sistema deve permitir o registro de **reclamações** pelo usuário. Tendo como atributos:   * nome (String); * e-mail (String); * telefone (String); * descrição da reclamação (String); * número do ônibus (String); * número de linha (String); * Data da reclamação (Date Time); * hora da reclamação (Date Time).   Todos os atributos são obrigatórios. | Usuário |
| RF20 | O sistema deve registrar um **destino**, sendo este um método. Tendo como retorno os atributos:  Neste método o sistema selecionará a parada mais próxima, do endereço de origem, e irá recomendar uma linha para se aproximar do endereço de destino.   * Endereço origem (String); * Endereço destino(String).   Todos os atributos são obrigatórios. | Usuário |
| RF21 | O sistema deve registrar a **localização do usuário.** Tendo como atributos:   * Dados de coordenadas geográficas vindo do celular (String).   Todos os atributos são obrigatórios. | Usuário |
| RF22 | O sistema deve exibir as **reclamações para a empresa.** Tendo como atributos:   * dados da reclamação vindo do registro do usuário (Reclamacao - Registrado previamente).   Todos os atributos são obrigatórios. | Empresa |
| RF23 | O sistema deve permitir o registro de **Operação** **.** Tendo como atributos:   * Número do Ônibus (Onibus - Registrado previamente); * Número da Linha (Linha - Registrado previamente); * Data da Operação (Date)   Todos os atributos são obrigatórios. | Empresa |

**Fonte**: Elaborada pelo autor

### Requisitos Não-Funcionais

Os requisitos não funcionais são as restrições que o sistema possui sobre as os serviços do sistema. (Ian Sommerville, 2007).

**Tabela 2** Requisitos Não Funcionais

|  |  |  |
| --- | --- | --- |
| **Nº** | **Descrição** | **Ator** |
| RNF 1 | **Banco de Dados Relacional,** será utilizado a plataforma SqlServer. | Desenvolvedor |
| RNF 2 | As **Plataformas Disponíveis**, devem ser desenvolvidas para Android e desktop. | Desenvolvedor |
| RNF 3 | O **Layout Simplificado** contém campos específicos e ao mesmo tempo acessíveis para o uso. | Desenvolvedor |
| RNF 4 | A **Linguagem de Programação** utilizada será C# - Estrutural. | Desenvolvedor |

**Fonte**: Elaborada pelo autor

### Restrições, Suposições e Dependências

Forneça uma descrição geral de qualquer item que limitará a decisão do desenvolvedor e uma lista de fatores que afetam os requisitos expressos.

* Utilizaremos como base de dados para o endereço o ViaCEP para alimentar nossa base de dados;
* Usaremos como base o mapa do OpenStreetMap;
* projeto de lei de número 448, de 2015, diz que todos os ônibus são obrigados a possuir GPS e botão de pânico;

### Requisitos Adiados

Forneça uma descrição dos requisitos identificados pelo usuário ou pelo desenvolvedor, mas que não serão desenvolvidos nessa primeira interação, ou seja, serão analisados e desenvolvidos posteriormente.

* Cadastrar os guinchos de socorro dos ônibus.;
* Sistema terá uma sincronização com o botão de pânico do ônibus;
* O sistema deverá permitir cadastrar a tripulação do coletivo nas viagens;

## **Casos de Uso**

Desenvolvido em 1996, com foco em melhorar a prática de criação de software, a Indústria de Tecnologia da Informação iniciou a elaboração de uma metodologia de trabalho, buscando proporcionar maior clareza aos profissionais envolvidos nas equipes de criação. Este método aperfeiçoou a maneira como as empresas enxergavam o processo de desenvolvimento e análise das aplicações.

A linguagem UML (Linguagem Unificada de Modelagem) busca auxiliar, através de conceitos, objetos, símbolos e diagramas, de uma forma simples, objetiva e funcional, o entendimento de um sistema. Sendo uma modelagem orientada a objetos, a distinção completa das entidades envolvidas, expõe com mais facilidade os reais requisitos funcionais do software (Lucidchart, 2018).

Este diagrama tem como objetivo auxiliar na comunicação entre o analista e o cliente. Documenta as funcionalidades do sistema, ou seja, descreve as principais funções e interações entre os usuários e o sistema (Renato do site Devmedia, 2012).

### Diagrama de Casos de Uso

O diagrama de casos de uso é um diagrama da UML cujo objetivo é representar um requisito do sistema que será automatizado. O diagrama de caso de uso mostra como a análise de requisitos molda o sistema. Na figura a seguir, pode-se conferir o diagrama de caso de uso do sistema.

**Figura 1** Diagrama de Casos de Uso da BusSpot

Diagrama, Esquemático

Descrição gerada automaticamente

**Fonte:** Elaborada pelo autor

### Especificação dos Casos de Uso

Elaborar as especificações para caso de uso mais relevante do seu projeto, como por exemplo: A especificação do caso de uso Cadastra Empresa é

**Tabela 3** Especificação do Caso de Uso de Cadastra Empresa

|  |
| --- |
| **Caso de Uso** |
| Cadastra Empresa |
| **Referências** |
| RF07 |
| **Descrição Geral** |
| O caso de uso inicia-se quando o Administrador do Sistema cadastra a empresa de transporte urbano no sistema. |
| **Atores** |
| Administrador do Sistema |
| **Pré-Condições** |
| Empresa está sem restrições governamentais, ter em mãos todos os dados da empresa. |
| **Garantia de Sucesso (Pós-Condições)** |
| Empresa Cadastrada, aguardando serem lançado dados de trajeto,rota e ônibus. |
| **Requisitos Especiais** |
| Empresa tem que estar em dia com a situação na STU e ter documentado todas as informações de trajeto e rota. |
|  |
| **Fluxo Básico** |  |
| 1.   Adm de Sistemas cadastra a empresa e o sistema pede para cadastrar o endereço da empresa informando o CEP; |  |
| 2.   Sistema valida esse CEP através de uma função no banco de dados e puxa o endereço completo daquele CEP; |  |
| 3. Sistema solicita a inscrição estadual da empresa; |  |
| 4.    Sistema faz validação das informações; |  |
| 5.    Sistema gera um identificador interno dessa empresa; |  |
| 6.    Empresa já está habilitada para usufruir do sistema. |  |
| **Fluxo Alternativo** |  |
| 1.  Cliente com restrição na Inscrição Estadual |  |
| 1.1    Sistema deixa o cliente usar o sistema por um período de 15 dias uteis até sua regularização; |  |
| 1.2    Retorne ao passo 5. |  |

**Fonte:** Elaborada pelo autor

**Tabela 4** Especificação do Caso de Uso de login

|  |
| --- |
| **Caso de Uso** |
| Login |
| **Referências** |
| RF02 |
| **Descrição Geral** |
| Caso de uso se inicia quando a empresa já está cadastrada no sistema. |
| **Atores** |
| Empresa |
| **Pré-Condições** |
| Empresa tem que estar com o cadastro valido no sistema. |
| **Garantia de Sucesso (Pós-Condições)** |
| Após o login a empresa já pode usufruir das funcionalidades do sistema. |
| **Requisitos Especiais** |
| Empresa poderá recuperar senha através do E-Mail |
|  |
| **Fluxo Básico** |  |
| 1. Empresa Informa o usuário e senha; |  |
| 2. Sistema valida essas informações; |  |
| 3. Sistema Informa ao usuário que os dados são validos; |  |
| 4. Empresa já pode usar o sistema; |  |
| **Fluxo Alternativo** |  |
| 1.  Cliente que não informa dados validos pra logon |  |
| 1.1    Sistema informa a inconsistências nos dados |  |
| 1.2    Empresa tem que usar a recuperação de senha e retornar ao passo 1 |  |

**Fonte:** Elaborada pelo autor

**Tabela 5** Especificação do Caso de Uso de Cadastra de Ônibus

|  |
| --- |
| **Caso de Uso** |
| Cadastro de Ônibus |
| **Referências** |
| RF15 |
| **Descrição Geral** |
| Caso de uso se inicia quando já tem registrado no sistema: comodidades, acessibilidade, embarque e pagamento. |
| **Atores** |
| Empresa |
| **Pré-Condições** |
| Todos os campos preenchidos corretamente, precisa ter em mãos todos os dados do ônibus. |
| **Garantia de Sucesso (Pós-Condições)** |
| Após o cadastro do ônibus ele já pode ficar disponível no mapa e a empresa já pode usá-lo para controlar a sua frota, receber reclamações e alocá-lo em linhas. |
| **Requisitos Especiais** |
| O ônibus precisa estar com a documentação em dia ( Regularizado ). |
|  |
| **Fluxo Básico** |  |
| 1. Empresa registra os dados do ônibus |  |
| 2. Sistema confere os dados recebidos e compara se estão na tabela de depenica ( Comodidade, acessibilidade, embarque e pagamento ) |  |
| 3. Sistema confere a regularidade do ônibus. |  |
| 4. Sistema registra o ônibus no banco de dados. |  |
| 6. Ônibus disponível para uso. |  |
|  |  |
| **Fluxo Alternativo** |  |
| 1. Empresa não regulariza o ônibus. |  |
| 1.1 Sistema informa a irregularidade |  |
| 1.2 Empresa após ajustar a documentação do ônibus retornar ao passo 3. |  |

**Fonte:** Elaborada pelo autor

**Tabela 6** Especificação do Caso de Uso de Localização

|  |
| --- |
| **Caso de Uso** |
| Localização |
| **Referências** |
| RF17 |
| **Descrição Geral** |
| Caso de uso se inicia quando o ônibus tem como sistema embarcado o Rastreador transmitindo sinal de coordenadas geográficas. |
| **Atores** |
| Sistema |
| **Pré-Condições** |
| O rastreador do ônibus tem que ter o sinal vindo do Chip de telefonia em perfeitas condições caso contrário o ônibus fica sem transmissão de rastreador , fazendo o ônibus ficar invisível no sistema. |
| **Garantia de Sucesso (Pós-Condições)** |
| com a transmissão do ônibus correta pode ser usado para registrar viagens pelo sistema e pode ser mostrado no Mapa para o usuário e a Empresa. |
| **Requisitos Especiais** |
| Ser implantado no rastreador do ônibus um sistema com boa performasse e com um chip de bom sinal de banda larga. |
|  |
| **Fluxo Básico** |  |
| 1. Empresa instala o rastreador nos ônibus. |  |
| 2. Sistema integra com o sistema interno da Empresa para o monitoramento da frota. |  |
| 3. Sistema valida essa integração. |  |
| 4. Empresa pode informar no sistema se prefere localizar um ônibus ou um conjunto de ônibus através de uma linha. |  |
|  |
| 5. Ônibus já e apresentado na tela para o usuário ver. |  |
| **Fluxo Alternativo** |  |
| 1. Ônibus sem sinal do rastreador. |  |
| 1.1 Empresa deve trocar o equipamento |  |
| 1.2 Empresa devera validar as informações novamente e retornar ao fluxo 2. |  |

**Fonte:** Elaborada pelo autor

**Tabela 7** Especificação do Caso de Uso de Tempo Real

|  |
| --- |
| **Caso de Uso** |
| Tempo Real |
| **Referências** |
| RF18 |
| **Descrição Geral** |
| Caso de uso se inicia quando o ônibus tem uma viagem registrada pelo rastreador que e quando ele sai do ponto inicial a um horário e chega ao ponto final. |
|  |
| **Atores** |  |
| Sistema |  |
| **Pré-Condições** |  |
| O ônibus tem que estar em total sintonia com o CHIP para poder ter um bom rastreamento e a viagem ser registrada e a empresa tem que colocar a tabela de horário da linha que o ônibus irá operar |  |
|  |
|  |
| **Garantia de Sucesso (Pós-Condições)** |  |
| com isso podemos ter uma métrica se os coletivos estão realmente realizando viagem ou se a realmente atraso nas partidas do coletivo, com isso podemos desenvolver vários tipos de relatório de desempenho. |  |
|  |
|  |
| **Requisitos Especiais** |  |
| As viagens programadas devem ser registradas corretamente pela empresa. |  |
|  |
| **Fluxo Básico** |  |
| 1. Empresa cadastra os horários. |  |
| 2. Sistema registra o monitoramento do ônibus verificando a sua passagem nos pontos notáveis. |  |
| 3. Sistema confere as viagens registrada pela empresa e as viagens registrada pelo registrador. |  |
| 4. Sistema informar essa diferença na tela. |  |
|  |  |
| **Fluxo Alternativo** |  |
| 1. Sistema não consegue identificar a viagem do registrado: |  |
| 1.1 Empresa deverá verificar se foi falha no sistema ou no registrador do ônibus. |  |
| 1.2 Após reparos a viagem deve ser registrada normalmente. |  |

**Fonte:** Elaborada pelo autor

**Tabela 8** Especificação do Caso de Uso de Registro de Reclamação

|  |
| --- |
| **Caso de Uso** |
| Registro de Reclamação |
| **Referências** |
| RF19 |
| **Descrição Geral** |
| Caso de uso se inicia quando o usuário registra uma reclamação que aconteceu no interior do coletivo |
|  |
| **Atores** |  |
| Usuário |  |
| **Pré-Condições** |  |
| Usuário precisa relatar todos os fatos no formulário, preenchendo todos os campos. |  |
| **Garantia de Sucesso (Pós-Condições)** |  |
| Com isso o sistema encaminha para a empresa e com isso será tomado as providencias. |  |
| **Requisitos Especiais** |  |
| Necessita de informar dados pessoais do usuário sem a necessidade de um login e o retorno da empresa será notificado em seu e-mail. |  |
|  |
| **Fluxo Básico** |  |
| 1. Usuário abre a tela de Registro de Reclamação. |  |
| 2. Usuário preenche todos os campos corretamente. |  |
| 3. Sistema coleta os dados e informa essa reclamação a empresa. |  |
| 4. Empresa deve responder essa reclamação. |  |
|  |  |
| **Fluxo Alternativo** |  |
| 1. Usuário informa dados inválidos |  |
| 1.1 Será apresentado uma mensagem de erro ao usuário. |  |
| 1.2 Usuário terá que arrumar e retornar ao fluxo 2 |  |

**Fonte:** Elaborada pelo autor

**Tabela 9** Especificação do Caso de Uso de Consulta de Reclamação

|  |
| --- |
| **Caso de Uso** |
| Consulta Reclamações |
| **Referências** |
| RF22 |
| **Descrição Geral** |
| Caso de uso se inicia quando há registro de reclamação |
| **Atores** |
| Sistema |
| **Pré-Condições** |
| Registro valido de uma reclamação feita por um usuário. |
| **Garantia de Sucesso (Pós-Condições)** |
| Com esse caso de uso a empresa poderá ter um indicador de satisfação de usuário e poderá saber onde e suas linhas mais críticas. |
|  |
| **Requisitos Especiais** |  |
| Precisara ser mostrado a empresa a tela de registro de reclamação com um campo a mais para que seja informado ao usuário a providência tomada. |  |
|  |
| **Fluxo Básico** |  |
| 1. Usuário faz um registro valido da reclamação. |  |
| 2. Sistema a importa para a tela da empresa. |  |
| 3. Empresa verifica os dados da reclamação. |  |
| 4. Empresa registra uma providência sobre ocorrido. |  |
| 5. esse registro aparecera para o usuário. |  |
|  |  |
| **Fluxo Alternativo** |  |
| 1. Empresa constata que reclamação do usuário e de natureza invalida |  |
| 1.1 nas providências empresa informa ao usuário o motivo de sua reclamação ser invalida. |  |
| 1.2 Usuário terá que rever seu relato e voltar ao fluxo 1 |  |

**Fonte:** Elaborada pelo autor

## **Modelo de Domínio**

Com base nos slides da iniciação científica "Modelo de Domínio: Visualizando Conceitos" publicado pela M. Cecilia C. Baranauskas, um dos artefatos mais importantes em aplicações de análise orientada a objeto é o modelo de domínio, pois demonstra classes conceituais significativas. O intuito do modelo de domínio deve-se à incorporação de conhecimentos sobre o projeto, com os conceitos, tipo de dados, indivíduos e regras, considerando todos os desenvolvimentos em lógica de programação.

**Figura 2** Modelo de Domínio da BusSpot

Diagrama

Descrição gerada automaticamente

**Fonte:** Elaborada pelo autor

## **Diagrama de Objetos**

De acordo com o site Lucidchart em "O que é um diagrama de objetos?", um dos diagramas mais relevantes para um engenheiro de software na diagramação UML é o diagrama de componentes, devido a colaboração da visão geral e entendimento da estrutura do sistema. Bem parecido com o diagrama de classe, o diagrama de componentes demonstra classes conceituais significativas, porém incide sobre os atributos de um conjunto de objetos e há relações entre si.

**Figura 3** Diagrama de Objeto da BusSpot

Diagrama

Descrição gerada automaticamente

**Fonte:** Elaborada pelo autor

## **Diagrama de Classes de Análise**

O diagrama de classe tem por objetivo representar as estruturas dos modelos dos objetos, e identificando as classes, o que auxilia a criação das tabelas no banco de dados (RENATO JOSE GROFFE, 2013).

**Figura 4** Diagrama de Classe Análise da BusSpot

Diagrama

Descrição gerada automaticamente

**Fonte:** Elaborada pelo autor

## **Diagrama de Atividades**

Um diagrama de atividade ilustra a natureza dinâmica de um sistema pela mo- Segundo Ricardo, um diagrama de atividades é um diagrama UML usado para modelar atividades.

Aspectos comportamentais do processo. Neste diagrama, uma atividade é modelada como uma sequência estruturada de ações, possivelmente controladas por nós de decisão temporal. No caso mais simples, um diagrama de atividades pode ser confundido com um fluxograma. Ao contrário dos fluxogramas, no entanto, os diagramas de atividades UML suportam vários outros recursos, como particionamento e bifurcação e fusão de nós, além de suportar a definição de regiões de interrupção modeladas e bem definidas.

Logo abaixo temos o diagrama de atividades desenvolvido para o projeto mostrando a sequência de atividades que pode ser feita dento do nosso projeto.

**Figura 5** Diagrama de Atividade da BusSpot

Diagrama

Descrição gerada automaticamente

**Fonte:** Elaborada pelo autor

## **Diagrama de Estados**

Segundo Ricardo, os diagramas de estado (ou máquinas de estado, como aparecem na versão 2.3 da norma UML) são utilizados para modelar um comportamento discreto em sistemas de transição entre estados finitos. Existem basicamente dois usos para máquinas de estado: máquinas de estado comportamentais e máquinas de estado para protocolos. conforme exemplo apresentado na Figura 6.

**Figura 6** Diagrama de Estados da BusSpot

**Diagrama

Descrição gerada automaticamente**

**Fonte:** Elaborada pelo autor

# PROJETO DE SOFTWARE

No módulo de projeto de software deve ser implementado o protótipo do projeto por completo, deve se usar todas as ferramenta de desenvolvimento junto aos requisitos do sistema e entregar ao cliente uma prototipação das interfaces do seu sistema e do seu banco de dados, o cliente deve ter a ciência de como será a interface do produto e com isso deverá saber também como será a estrutura do banco de dados do mesmo, nessa etapa usamos ferramentas para prototipação, demonstrado nas figuras dos próximos tópicos, e também utilizamos ferramenta para montar o banco de dados sem implementá-lo. Nessa etapa também foi proposto o desenho de alguns diagramas da UML, como o de atividade e o de pacote, porém esses desenhos têm ênfase no projeto em si do que no produto. Os tópicos a seguir demonstraram como foi planejado o projeto da BusSpot.

## **Projeto de Interface**

De acordo com o editorial Aela.io no site Medium, o protótipo de interface é fundamental para identificação do resultado do projeto, considerando toda a pesquisa, dados e desenvolvimentos feitos até então. Para haver uma aprovação e então continuação, é importante haver uma análise prévia para, se necessário, fazer mudanças antes da entrega final.

Primeiramente, a interface é relevante para a representação de como é e será construído, em seguida, precisão. Pois, é importante haver fidelidade ao descrever sobre o projeto e como será a interface. Além disso, há a interatividade e evolução advindas do desenvolvimento do protótipo, assim aprimorando para o resultado desejado.

Um dos pontos mais relevantes para os protótipos é avaliar a viabilidade técnica do produto, reduzir os riscos e desperdício de tempo, referenciar os desenvolvedores e praticidade na criação.

### 6.1.2 Interface da Empresa

#### 6.1.2.1 Tela de Login

Figura 7 Tela de Login da BusSpot

Interface gráfica do usuário, Site

Descrição gerada automaticamente

**Fonte:** Elaborada pelo autor

#### 6.1.2.2 Menu Principal

Figura 8 Menu Inicial da BusSpot

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

**Fonte:** Elaborada pelo autor

#### 6.1.2.3 Telas de Cadastro de Endereço

Figura 9 : Tela de Cadastro de Estado da BusSpot

Tabela

Descrição gerada automaticamente

**Fonte:** Elaborada pelo autor

Figura 10 Tela de Cadastro de Cidade da BusSpot

Tabela

Descrição gerada automaticamente

**Fonte:** Elaborada pelo autor

Figura 11 Tela de Cadastro de Bairro da BusSpot

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

**Fonte:** Elaborada pelo autor

Figura 12 Tela de Cadastro de Tipo de Logradouro da BusSpot

Interface gráfica do usuário

Descrição gerada automaticamente

**Fonte:** Elaborada pelo autor

Figura 13 Tela de Cadastro de Logradouro da BusSpot

Interface gráfica do usuário, Texto

Descrição gerada automaticamente

**Fonte:** Elaborada pelo autor

Figura 14 Tela de Cadastro de CEP da BusSpot

Interface gráfica do usuário

Descrição gerada automaticamente

**Fonte:** Elaborada pelo autor

#### 6.1.2.4 Cadastro de Empresa

Figura 15 Tela de Cadastro de Empresa da BusSpot

Interface gráfica do usuário

Descrição gerada automaticamente

**Fonte:** Elaborada pelo autor

#### 6.1.2.5 Cadastro de Onibus

Figura 16 Tela de Cadastro de Ônibus da BusSpot

Interface gráfica do usuário

Descrição gerada automaticamente

**Fonte:** Elaborada pelo autor

#### 1.1.2.6 Cadastro de Linha / Paradas e Ponto

Figura 17 Tela de Cadastro de Parada da BusSpot

Interface gráfica do usuário

Descrição gerada automaticamente

**Fonte:** Elaborada pelo autor

Figura 18 Tela de Cadastro de Ponto da BusSpot

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

**Fonte:** Elaborada pelo autor

Figura 19 Tela de Cadastro de Linha da BusSpot

Interface gráfica do usuário

Descrição gerada automaticamente

**Fonte:** Elaborada pelo autor

#### 6.1.2.7 Registro de Operação

Figura 20 Tela de Registro de Operação da BusSpot

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

**Fonte:** Elaborada pelo autor

### 1.1.3 Interface Usuário

1.1.3.1 Tela de Inicial

Interface gráfica do usuário, Site

Descrição gerada automaticamente

**Fonte:** Elaborada pelo autor

1.1.3.2 Tela de Busca por Onibus

Uma imagem contendo Interface gráfica do usuário

Descrição gerada automaticamente

**Fonte:** Elaborada pelo autor

1.1.3.3 Tela de Registro de Reclamação

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

**Fonte:** Elaborada pelo autor

1.1.3.4 Verificando as Informações de um Onibus

Interface gráfica do usuário

Descrição gerada automaticamente com confiança média

## **Projeto de Dados**

Informar nesta seção qual o Banco de Dados a ser utilizado, qual a IDE utilizada para manipulação do Banco de Dados, assim como descrever quais os tipos de dados aceitos no Banco de Dados. Nesse projeto iremos utilizar a linguagem SQL, na IDE do SQL SERVER da Microsoft, o objetivo desse banco de dados e armazenar as viagens, os ônibus e as linhas possibilitando consultas ágeis que retornem as informações de maneira rápida e prática, através de emissão de relatórios diversos ou consultas na web.

As entidades necessárias para o controle da empresa de ônibus seria:

* Endereço: para cadastrar o endereço da empresa e das paradas do ônibus.
* Ônibus: para cadastrar e armazenar toda a frota da empresa.
* Linhas: para a empresa cadastrar todas as linhas e as ordens operacional.
* Reclamações: para os usuários do transporte pode reportar a empresa como está sendo a operação dela.
* Localização: para armazenar as informações coletadas do rastreador para controle interno da empresa.
* O modelo do Banco de dados será o Modelo Relacional:

Figura 21 Modelo Objeto feito pela IDE SQL SERVER

Diagrama, Esquemático

Descrição gerada automaticamente

**Fonte:** Elaborada pelo autor

### Mapeamento Objeto-Relacional

Segundo Thiago Nunes, O mapeamento objeto-relacional (ORM ou MOR) é uma técnica de muito utilizada para converter dados entre bancos relacionais e linguagens orientadas a objeto, como é o caso do PHP, C++, Java, Python, Ruby, entre outras.

Apesar das linguagens de programação modernas possuírem recursos da POO (Programação Orientada a Objetos), a maioria dos SGBD (Sistemas de Gerenciamento de Bases de Dados) é baseada no modelo relacional, onde temos tabelas ao invés de objetos.

Guardar instâncias de objetos em registros de tabelas é o objetivo das técnicas de MOR (Mapeamento Objeto Relacional).

A prática de mapear objetos relacionais tem grande importância no desenvolvimento de projetos. Abaixo encontra-se o MOR da BusSpot.

#### 6.2.1.1 estado (id, nome, sigla)

#### 6.2.1.2 cidade (id, nome, id\_estado, CEP)

#### 6.2.1.3 bairro(id, nome)

#### 6.2.1.4 tipo\_logradouro(id, nome)

#### 6.2.1.5 logradouro(id, nome)

#### 6.2.1.6 endereco (id, nome)

#### 6.2.1.7 empresa (id, fantasia, CNPJ, #id\_endereco, numeral, usuario, senha)

#### 6.2.1.8 reclamacao(id, nome, email, telefone, desc\_rec, data\_rec, #id\_empresa)

#### 6.2.1.9 parada(id, latitude, longitude, nome, #id\_endereco)

#### 6.2.1.10 pontos (id, tipo, #id\_parada)

#### 6.2.1.11 linha(id, numero\_linha, descricao\_linha, horario\_partida, horario\_chegada, #id\_empresa, #id\_ponto)

#### 6.2.1.12 embarque(id, tipo\_emb)

#### 6.2.1.13 comodidades(id, arcond, wifi, tomada, banheiro, cadrec, vendas)

#### 6.2.1.14 acessibilidade(id, faixa\_guia, elevadores, libras)

#### 6.2.1.15 localizacao( id, data\_hora, latitude, longitude)

#### 6.2.1.16 pagamento(id, tipo\_pag)

#### 6.2.1.17 onibus(id, numero\_onibus, #id\_comodidade, #id\_pagamento, #id\_localizacao, #id\_acessibilidade, #id\_embarque, #id\_empresa)

#### 6.2.1.18 operacao(id, data\_operacao, #id\_linha, #id\_onibus)

### Estrutura das Tabelas no Banco de Dados

O padrão adotado no nosso banco de dados relacional para as chaves primárias será: PK\_nomeTabela, enquanto para as chaves estrangeiras adotamos o padrão comum de banco de dados que será: FK\_TabelaComID\_TabelacomAChaveEstrangeira, paras as chaves únicas o padrão será: Campo\_UK0n, onde “n” é uma sequência numérica dependendo da quantidade de campos da tabela, o mesmo padrão será utilizado para as verificações: Campo\_CKn.

Campo\_UK0n, onde “n” é uma sequência numérica dependendo da quantidade de campos da tabela, o mesmo padrão será utilizado para as verificações: Campo\_CKn.

#### 1.2.2.1 Estado

Tabela 10 Estrutura da tabela Estado da BusSpot

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Campo** | **Tipo** | **Obrigatório?** | **Chave Primária**  **?** | **Chave Estrangeira** | | **Chave Única** | |
| **Tabela** | **Campo** | **Grupo** | **Ordem** |
| id | int | Sim | Sim |  |  |  |  |
| nome | varchar2(100) | Sim |  |  |  | 1 | 1 |
| sigla | varchar2(2) | Sim |  |  |  | 2 | 1 |

**Fonte:** Elaborada pelo autor

#### 1.2.2.2 Cidade

Tabela 11 Estrutura da tabela Cidade da BusSpot

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Campo** | **Tipo** | **Obrigatório?** | **Chave Primária**  **?** | **Chave Estrangeira** | | **Chave Única** | |
| **Tabela** | **Campo** | **Grupo** | **Ordem** |
| id | int | Sim | Sim |  |  |  |  |
| nome | varchar2(200) | Sim |  |  |  | 1 | 1 |
| cep | varchar2(11) | Não |  |  |  | 2 | 1 |
| idEstado | int | Sim |  | Estado | id |  |  |

**Fonte:** Elaborada pelo autor

#### 1.2.2.3 Bairro

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Campo** | **Tipo** | **Obrigatório?** | **Chave Primária**  **?** | **Chave Estrangeira** | | **Chave Única** | |
| **Tabela** | **Campo** | **Grupo** | **Ordem** |
| id | int | Sim | Sim |  |  |  |  |
| nome | varchar2(200) | Sim |  |  |  | 1 | 1 |

Tabela 12 Estrutura da tabela Bairro da BusSpot

**Fonte:** Elaborada pelo autor

#### 1.2.2.4 Logradouro

Tabela 13 Estrutura da tabela Logradouro da BusSpot

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Campo** | **Tipo** | **Obrigatório?** | **Chave Primária**  **?** | **Chave Estrangeira** | | **Chave Única** | |
| **Tabela** | **Campo** | **Grupo** | **Ordem** |
| id | int | Sim | Sim |  |  |  |  |
| nome | varchar2(200) | Sim |  |  |  | 1 | 1 |

**Fonte:** Elaborada pelo autor

#### 1.2.2.5 Tipo Logradouro

Tabela 14 Estrutura da tabela Tipo de Logradouro da BusSpot

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Campo** | **Tipo** | **Obrigatório?** | **Chave Primária?** | **Chave Estrangeira** | | **Chave Única** | |
| **Tabela** | **Campo** | **Grupo** | **Ordem** |
| id | int | Sim | Sim |  |  |  |  |
| nome | varchar2(200) | Sim |  |  |  | 1 | 1 |

**Fonte:** Elaborada pelo autor

#### 1.2.2.6 Endereço

##### 

Tabela 15 Estrutura da tabela Endereço da BusSpot

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Campo** | **Tipo** | **Obrigatório?** | **Chave Primária**  **?** | **Chave Estrangeira** | | **Chave Única** | |
| **Tabela** | **Campo** | **Grupo** | **Ordem** |
| id | int | Sim | Sim |  |  |  |  |
| cep | varchar2(11) | Não |  |  |  | 1 | 1 |
| idCidade | int | Sim |  | Cidade | id |  |  |
| idBairro | int | Sim |  | Bairro | id |  |  |
| idLogradouro | int | Sim |  | Logradouro | id |  |  |
| idTipoLogradouro | int | Sim |  | Tipo Logradouro | id |  |  |

**Fonte:** Elaborada pelo autor

#### 1.2.2.7 Empresa

##### 

Tabela 16 Estrutura da tabela Empresa da BusSpot

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Campo** | **Tipo** | **Obrigatório?** | **Chave Primária?** | **Chave Estrangeira** | | **Chave Única** | |
| **Tabela** | **Campo** | **Grupo** | **Ordem** |
| id | int | sim | sim |  |  |  |  |
| fantasia | varchar2(50) | sim |  |  |  | 1 | 1 |
| cnpj | varchar2(50) | sim |  |  |  | 2 | 1 |
| id\_endereco | int | sim |  | Endereco | id |  |  |
| numeral | varchar2(10) | sim |  |  |  |  |  |
| usuario | varchar2(50) | sim |  |  |  | 3 | 1 |
| senha | varchar2(50) | sim |  |  |  |  |  |

**Fonte:** Elaborada pelo autor

#### 1.2.2.8 Reclamação

Tabela 17 Estrutura da tabela Reclamação da BusSpot

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Campo** | **Tipo** | **Obrigatório?** | **Chave Primária?** | **Chave Estrangeira** | | **Chave Única** | |
| **Tabela** | **Campo** | **Grupo** | **Ordem** |
| id | int | sim | sim |  |  |  |  |
| nome | varchar2(50) | sim |  |  |  |  |  |
| email | varchar2(50) | sim |  |  |  |  |  |
| telefone | varchar2(50) | sim |  |  |  |  |  |
| desc\_rec | varchar(max) | sim |  |  |  |  |  |
| data\_rec | date | sim |  |  |  |  |  |
| id\_empresa | int | sim |  | Empresa | id |  |  |

**Fonte:** Elaborada pelo autor

#### 1.2.2.9 Paradas

##### 

Tabela 18 Estrutura da tabela Parada da BusSpot

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Campo** | **Tipo** | **Obrigatório?** | **Chave Primária?** | **Chave Estrangeira** | | **Chave Única** | |
| **Tabela** | **Campo** | **Grupo** | **Ordem** |
| id | int | sim | sim |  |  |  |  |
| latitude | varchar2(50) | sim |  |  |  | 1 | 1 |
| longitude | varchar2(50) | sim |  |  |  | 2 | 1 |
| nome | varchar2(50) | sim |  |  |  |  |  |
| id\_endereco | int | sim |  | Endereco | id |  |  |

**Fonte:** Elaborada pelo autor

#### 1.2.2.10 Ponto

Tabela 19 Estrutura da tabela Ponto da BusSpot

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Campo** | **Tipo** | **Obrigatório?** | **Chave Primária?** | **Chave Estrangeira** | | **Chave Única** | |
| **Tabela** | **Campo** | **Grupo** | **Ordem** |
| id | int | sim | sim |  |  |  |  |
| tipo | varchar2(20) | sim |  |  |  | 1 | 1 |
| id\_parada | int | sim |  | Paradas | id |  |  |

**Fonte:** Elaborada pelo autor

#### 1.2.2.11 Linha

Tabela 20 Estrutura da tabela Linha da BusSpot

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Campo** | **Tipo** | **Obrigatório?** | **Chave Primária?** | **Chave Estrangeira** | | **Chave Única** | |
| **Tabela** | **Campo** | **Grupo** | **Ordem** |
| id | int | sim | sim |  |  |  |  |
| numero\_linha | varchar2(50) | sim |  |  |  | 1 | 1 |
| descricao\_linha | varchar2(50) | não |  |  |  |  |  |
| horario\_partida | varchar2(50) | sim |  |  |  |  |  |
| horario\_chegada | varchar2(50) | sim |  |  |  |  |  |
| id\_empresa | int | sim |  | Empresa | id |  |  |
| id\_ponto | int | sim |  | Ponto | id |  |  |

**Fonte:** Elaborada pelo autor

#### 1.2.2.12 Embarque

Tabela 21 Estrutura da tabela Embarque da BusSpot

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Campo** | **Tipo** | **Obrigatório?** | **Chave Primária?** | **Chave Estrangeira** | | **Chave Única** | |
| **Tabela** | **Campo** | **Grupo** | **Ordem** |
| id | int | sim | sim |  |  |  |  |
| tipo\_embarque | varchar2(50) | sim |  |  |  | 1 | 1 |

**Fonte:** Elaborada pelo autor

#### 1.2.2.13 Comodidades

Tabela 22 Estrutura da tabela Comodidade da BusSpot

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Campo** | **Tipo** | **Obrigatório?** | **Chave Primária?** | **Chave Estrangeira** | | **Chave Única** | |
| **Tabela** | **Campo** | **Grupo** | **Ordem** |
| id | int | sim | sim |  |  |  |  |
| arCond | boolean | sim |  |  |  |  |  |
| wifi | boolean | sim |  |  |  |  |  |
| tomada | boolean | sim |  |  |  |  |  |
| banheiro | boolean | sim |  |  |  |  |  |
| cadRec | boolean | sim |  |  |  |  |  |
| vendas | boolean | sim |  |  |  |  |  |

**Fonte:** Elaborada pelo autor

#### 1.2.2.14 Acessibilidade

Tabela 23 Estrutura da tabela Acessibilidade da BusSpot

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Campo** | **Tipo** | **Obrigatório?** | **Chave Primária?** | **Chave Estrangeira** | | **Chave Única** | |
| **Tabela** | **Campo** | **Grupo** | **Ordem** |
| id | int | sim | sim |  |  |  |  |
| faixa\_guia | boolean | sim |  |  |  |  |  |
| elevadores | boolean | sim |  |  |  |  |  |
| libras | boolean | sim |  |  |  |  |  |

**Fonte:** Elaborada pelo autor

#### 1.2.2.15 Pagamento

##### 

Tabela 24 Estrutura da tabela Pagamento da BusSpot

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Campo** | **Tipo** | **Obrigatório?** | **Chave Primária?** | **Chave Estrangeira** | | **Chave Única** | |
| **Tabela** | **Campo** | **Grupo** | **Ordem** |
| id | int | sim | sim |  |  |  |  |
| tipo\_pag | varchar2(50) | sim |  |  |  |  |  |

**Fonte:** Elaborada pelo autor

#### 1.2.2.16 Ônibus

Tabela 25 Estrutura da tabela Ônibus da BusSpot

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Campo** | **Tipo** | **Obrigatório?** | **Chave Primária?** | **Chave Estrangeira** | | **Chave Única** | |
| **Tabela** | **Campo** | **Grupo** | **Ordem** |
| id | int | sim | sim |  |  |  |  |
| numero\_onibus | varchar2(50) | sim |  |  |  | 1 | 1 |
| id\_comodidade | int | sim |  | Comodidades | id |  |  |
| id\_pagamento | int | sim |  | Pagamento | id |  |  |
| id\_acessibilidade | int | sim |  | Acessibilidade | id |  |  |
| id\_embarque | int | sim |  | Embarque | id |  |  |
| id\_empresa | int | sim |  | Empresa | id |  |  |

**Fonte:** Elaborada pelo autor

#### 1.2.2.17 Localização

##### 

Tabela 26 Estrutura da tabela Localização da BusSpot

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Campo** | **Tipo** | **Obrigatório?** | **Chave Primária?** | **Chave Estrangeira** | | **Chave Única** | |
| **Tabela** | **Campo** | **Grupo** | **Ordem** |
| id | int | sim | sim |  |  |  |  |
| data\_hora | date | sim |  |  |  | 1 | 1 |
| latitude | varchar2(50) | sim |  |  |  |  |  |
| longitude | varchar2(50 | sim |  |  |  |  |  |
| id\_onibus | int | sim |  | Onibus | id |  |  |

**Fonte:** Elaborada pelo autor

#### 1.2.2.18 Operação

Tabela 27 Estrutura da tabela Operação da BusSpot

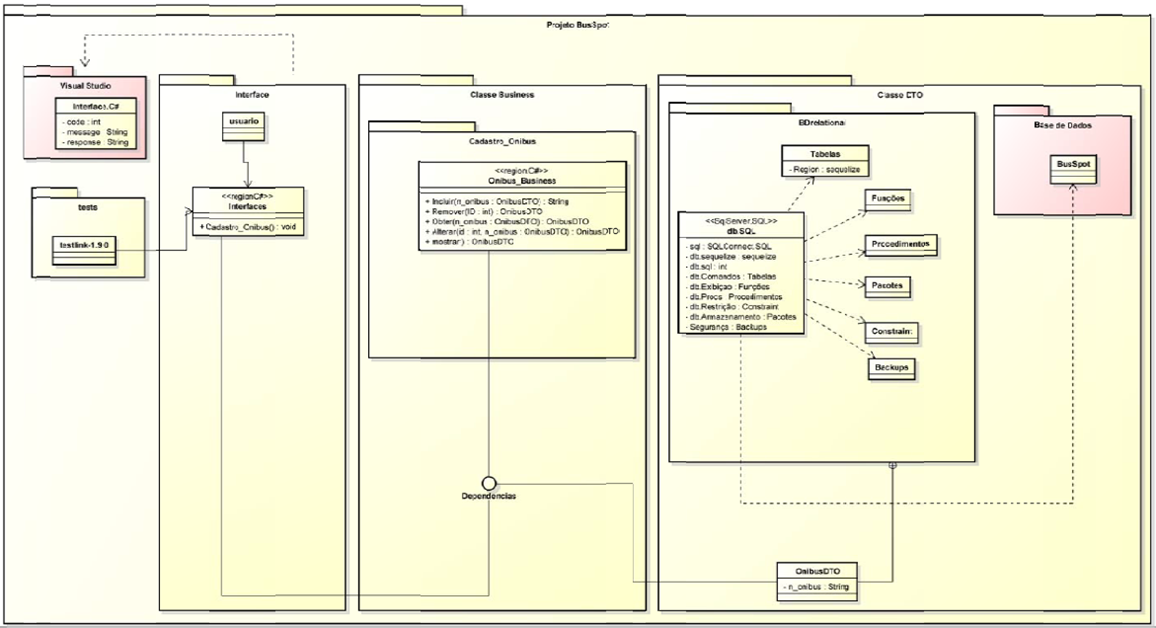
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Campo** | **Tipo** | **Obrigatório?** | **Chave Primária?** | **Chave Estrangeira** | | **Chave Única** | |
| **Tabela** | **Campo** | **Grupo** | **Ordem** |
| id | int | sim | sim |  |  |  |  |
| data\_op | date | sim |  |  |  |  |  |
| id\_onibus | int | sim |  | Onibus | id |  |  |
| id\_linha | int | sim |  | Linha | id |  |  |

**Fonte:** Elaborada pelo autor

### Diagrama de Pacotes

O diagrama de pacote consiste em mostrar a representação interna do projeto, como é feita a transição internamente entre as classes do banco de dados, e como é feita a conexão da camada de negócio com a camada de interfaces e por último como é feita a compilação e os testes.

Figura 22 Diagrama de Pacote Projeto da BusSpot



**Fonte:** Elaborada pelo autor

### Diagrama de Classes de Projeto

Esse Diagrama de Classe tem o conceito de se enxergar as relações das classes no padrão DTO (Data Transfer Object).

Segundo Luiz do site stackOverflow, Data Transfer Object (DTO) ou simplesmente Transfer Object é um padrão de projetos bastante usado em Java para o transporte de dados entre diferentes componentes de um sistema, diferentes instâncias ou processos de um sistema distribuído ou diferentes sistemas via serialização.

A ideia consiste basicamente em agrupar um conjunto de atributos em uma classe simples de forma a otimizar a comunicação.

Numa chamada remota, seria ineficiente passar cada atributo individualmente. Da mesma forma seria ineficiente e poderia até causar erros passando uma entidade mais complexa.

Figura 23 Diagrama de Classe Análise Projeto da BusSpot

Diagrama, Esquemático

Descrição gerada automaticamente

**Fonte:** Elaborada pelo autor

## **Projeto Procedimental**

Projetar Software é o processo de aplicar várias técnicas e princípios com o propósito de se definir um dispositivo, processo ou sistema, com detalhes suficientes para permitir sua realização física (Taylor-59).

O Projeto de software é o núcleo técnico da Engenharia de Software. É a única maneira de se traduzir "com precisão", os requisitos do usuário para um produto ou sistema acabado. Meta: Traduzir requisitos numa representação de software (Portella).

Figura 24 Modelo Projeto Procedimental da BusSpot

Diagrama

Descrição gerada automaticamente

Fonte: Elaborada pelo autor

### Diagrama de Sequência

Um diagrama de sequência é uma espécie de diagrama de interação, pois descreve como, e em qual ordem, um grupo de objetos trabalha em conjunto. Estes diagramas são usados por desenvolvedores de software e profissionais de negócios para entender as necessidades de um novo sistema ou para documentar um processo existente. Diagramas de sequência são conhecidos como diagramas de eventos ou cenários de eventos. O que é um diagrama de sequência, Lucidchart, 2018

Figura 25 Diagrama de Sequência da BusSpot

Diagrama

Descrição gerada automaticamente

**Fonte:** Elaborada pelo autor

## **Projeto Arquitetural**

O projeto arquitetural precede a etapa de construção da obra. O projeto arquitetural determina as partes de uma construção e como estas devem interagir. A arquitetura garante a unidade da obra, ou seja, a consistência entre as suas partes (Vergilio).

Ver algumas definições em (Silva), sendo que um exemplo está apresentado na Figura abaixo.

Figura 26 Modelo Projeto Arquitetural da BusSpot

Diagrama

Descrição gerada automaticamente

**Fonte:** Elaborada pelo autor

# 7 IMPLEMENTAÇÃO

Nesse capítulo vamos descrever as características do ambiente de desenvolvimento, servidor de aplicação, servidor de dados, pacotes utilizados etc.

## 7.1 Ambiente de Desenvolvimento.

Nessa etapa será abordado o ambiente de desenvolvimento do projeto. Um ambiente de desenvolvimento é uma coleção de procedimentos e ferramentas para desenvolver, testar e depurar um aplicativo ou programa.

Nesse projeto utilizamos uma IDE ambiente de desenvolvimento integrado que por definição de (RedHart 2019) é um software para criar aplicações que combina ferramentas comuns de desenvolvimento em uma única interface gráfica do usuário (GUI). Um IDE geralmente consiste em:

* Editor de código-fonte: é um editor de texto que auxilia na criação de código de software por meio de funcionalidades como destaque da sintaxe com indicadores visuais, recurso de preenchimento automático específico da linguagem e verificação de bugs durante a criação.
* Automação de compilação local: são utilitários que automatizam tarefas simples e repetíveis durante a criação de uma compilação local do software usada pelo desenvolvedor. São tarefas como compilação de código-fonte em código binário, criação de pacotes de código binário e execução de testes automatizados.
* Debugger: é um programa usado para testar outros programas e mostrar graficamente a localização do bug no código original.

A IDE utilizada foi o Visual Studio e para o Banco de Dados o SQL Server:

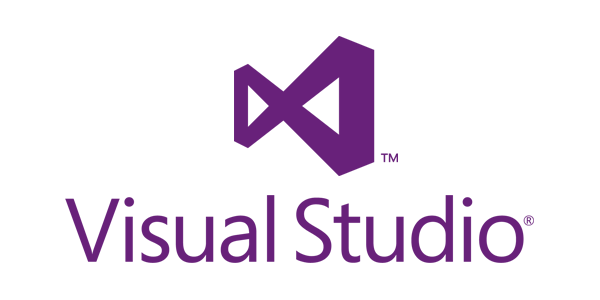
### 7.1.1 **Ferramentas e Linguagens utilizadas**

As ferramentas e linguagem utilizadas para o desenvolvimento do projeto serão:

#### 7.1.1.1 Visual Studio

A Microsoft Visual Studio é um pacote de programas da Microsoft que contém diversas ferramentas para o desenvolvimento de softwares.

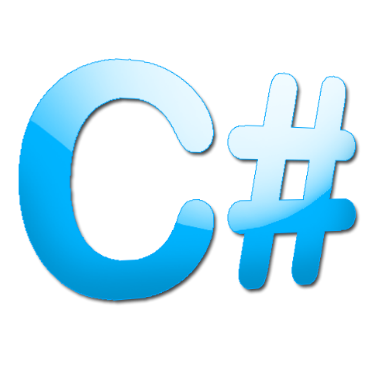
Figura 27 Visual Studio



**7.1.1.2 Linguagem de Programação**

C# é uma linguagem de programação, multiparadigma, de tipagem forte, desenvolvida pela Microsoft.

Figura 28 C#



**7.1.1.3 Adobe Photoshop**

Adobe Photoshop é um programa para edição de imagens e seu uso é fácil com resultados muito eficientes. Ele é líder em edição de fotos.

Figura 29 Adobe Photoshop



**7.1.1.4 Microsoft SQL Server**

SQL Server é um software para o gerenciamento de banco de dados, ele é bem fácil de se utilizar.

Figura 4 – Microsoft SQL Server

Figura 30 Sql Server



**7.1.1.5 Microsoft Word**

O Word é um editor de texto produzido pela Microsoft, suas funções variam das mais simples como escrever um texto comum, até produzir grandes documentos e formulários.

Figura 31 Word



**7.1.1.5 Astah Professional**

Astah – Utilizada nos diagramas dinâmicos, essa ferramenta já é bastante consolidada, voltada para a modelagem de sistemas utilizando a UML, utiliza como recurso adicional a modelagem.

Figura 32 Astah



### 7.1.2 Diagrama de Componente

O diagrama de componentes mostra o relacionamento entre diferentes componentes de um sistema. Para fins de UML 2.0, o termo "componente" refere-se a um módulo de classes que representa sistemas ou subsistemas independentes com capacidade de interagir com o restante do sistema.

Para isso, existe uma abordagem de desenvolvimento em torno de componentes: o desenvolvimento baseado em componentes (CBD). Nela, o diagrama de componentes identifica os diferentes componentes para que todo o sistema funcione corretamente.

Veja o exemplo de um diagrama de componente feito com base no projeto BusSpot.

Figura 32 Diagrama de Componente da BusSpot

Diagrama

Descrição gerada automaticamente

**Fonte:** Elaborada pelo autor

## 7.2 Coleta de Dados

Uma das maiores preocupações com relação ao projeto era da quantidade de interferência necessária para avaliar inconsistências e alterações nos dados das linhas, logo o processo de aquisição desses dados foi feito da forma mais generalista e será feita de forma manual pela empresa.

### 7.2.1 Coleta de Dados dos Terminais e paradas de ônibus

O projeto em si possui uma interface de cadastro de paradas de Onibus onde será armazenado o nome da parada as latitude e longitude junto com o CEP da parada atras do código abaixo:

#### 7.2.1.1 Função para Inserir uma Parada no Sistema

public string Incluir\_Parada(ParadasDTO parada)

{

string Resp;

SqlConnection Sqlcon = new SqlConnection();

try

{

Sqlcon.ConnectionString = ConexaoDB.cn;

Sqlcon.Open();

SqlCommand SqlCmd = new SqlCommand();

SqlCmd.Connection = Sqlcon;

SqlCmd.CommandText = "Proc\_Inserir\_Parada";

SqlCmd.CommandType = CommandType.StoredProcedure;

SqlParameter ParLatitude\_Parada = new SqlParameter();

ParLatitude\_Parada.ParameterName = "@Latitude";

ParLatitude\_Parada.SqlDbType = SqlDbType.VarChar;

ParLatitude\_Parada.Size = 50;

ParLatitude\_Parada.Value = Latitude;

SqlCmd.Parameters.Add(ParLatitude\_Parada);

SqlParameter ParLongitude\_Parada = new SqlParameter();

ParLongitude\_Parada.ParameterName = "@Longitude";

ParLongitude\_Parada.SqlDbType = SqlDbType.VarChar;

ParLongitude\_Parada.Size = 50;

ParLongitude\_Parada.Value = Longitude;

SqlCmd.Parameters.Add(ParLongitude\_Parada);

SqlParameter ParNome\_Parada = new SqlParameter();

ParNome\_Parada.ParameterName = "@NOME";

ParNome\_Parada.SqlDbType = SqlDbType.VarChar;

ParNome\_Parada.Size = 50;

ParNome\_Parada.Value = Nome;

SqlCmd.Parameters.Add(ParNome\_Parada);

SqlParameter ParID\_Endereco = new SqlParameter();

ParID\_Endereco.ParameterName = "@ID\_ENDERECO";

ParID\_Endereco.SqlDbType = SqlDbType.Int;

ParID\_Endereco.Value = ID\_ENDERECO;

SqlCmd.Parameters.Add(ParID\_Endereco);

Resp = SqlCmd.ExecuteNonQuery() == 1 ? "OK" : "Parada Não Inserido";

}

catch (Exception error)

{

Resp = error.Message;

}

finally { if (Sqlcon.State == ConnectionState.Open) Sqlcon.Close(); }

return Resp;

}

O código acima utiliza um procedimento de Banco de Dados para a inserção nas tabelas da DataBase, o procedimento segue descrito abaixo.

#### 7.2.1.1 Procedimento para Inserir uma Parada no Sistema

USE [BusSpot]

GO

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[Proc\_Inserir\_Parada] Script Date: 15/05/2022 14:23:46 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

ALTER Procedure [dbo].[Proc\_Inserir\_Parada]

@Latitude varchar(50),

@Longitude varchar(50),

@NOME varchar(50),

@ID\_ENDERECO int

as

insert Paradas(Latitude,Longitude,Nome,ID\_ENDERECO) values (@Latitude,@Longitude,@NOME,@ID\_ENDERECO).

### 7.2.2 Coleta de Dados e amostra dos ônibus

O projeto também não consome uma API interna para coletar dados dos coletivos isso será feito manualmente pela empresa atras do comando de “Proc\_Inserir\_Onibus”, porém para demostra ele de forma mais compreensível para o usuário também foi implementado uma função de mostrar todos os ônibus: “Proc\_Mostrar\_Onibus”, ambas funções seguem demostrada abaixo.

#### 7.2.2.1 Função para Inserir um Onibus no Sistema

public string Incluir\_Onibus(ColetivoDTO coletivo)

{

string Resp;

SqlConnection Sqlcon = new SqlConnection();

try

{

Sqlcon.ConnectionString = ConexaoDB.cn;

Sqlcon.Open();

SqlCommand SqlCmd = new SqlCommand();

SqlCmd.Connection = Sqlcon;

SqlCmd.CommandText = "Proc\_Inserir\_Onibus";

SqlCmd.CommandType = CommandType.StoredProcedure;

SqlParameter ParNumeroOnibus\_Onibus = new SqlParameter();

ParNumeroOnibus\_Onibus.ParameterName = "@Numero\_Onibus";

ParNumeroOnibus\_Onibus.SqlDbType = SqlDbType.VarChar;

ParNumeroOnibus\_Onibus.Size = 50;

ParNumeroOnibus\_Onibus.Value = N\_Onibus;

SqlCmd.Parameters.Add(ParNumeroOnibus\_Onibus);

SqlParameter ParPlaca\_Onibus = new SqlParameter();

ParPlaca\_Onibus.ParameterName = "@Placa";

ParPlaca\_Onibus.SqlDbType = SqlDbType.VarChar;

ParPlaca\_Onibus.Size = 50;

ParPlaca\_Onibus.Value = Placa;

SqlCmd.Parameters.Add(ParPlaca\_Onibus);

SqlParameter ParIDComodidade\_Onibus = new SqlParameter();

ParIDComodidade\_Onibus.ParameterName = "@ID\_Comodidade";

ParIDComodidade\_Onibus.SqlDbType = SqlDbType.Int;

ParIDComodidade\_Onibus.Value = ID\_Acessibilidade;

SqlCmd.Parameters.Add(ParIDComodidade\_Onibus);

SqlParameter ParIDPAGAMENTO\_Onibus = new SqlParameter();

ParIDPAGAMENTO\_Onibus.ParameterName = "@ID\_Pagamento";

ParIDPAGAMENTO\_Onibus.SqlDbType = SqlDbType.VarChar;

ParIDPAGAMENTO\_Onibus.Size = 50;

ParIDPAGAMENTO\_Onibus.Value = ID\_Pagamento;

SqlCmd.Parameters.Add(ParIDPAGAMENTO\_Onibus);

SqlParameter ParAcessibilidade\_Onibus = new SqlParameter();

ParAcessibilidade\_Onibus.ParameterName = "@ID\_Acessibilidade";

ParAcessibilidade\_Onibus.SqlDbType = SqlDbType.VarChar;

ParAcessibilidade\_Onibus.Size = 50;

ParAcessibilidade\_Onibus.Value = ID\_Acessibilidade;

SqlCmd.Parameters.Add(ParAcessibilidade\_Onibus);

SqlParameter ParIdEmbarque\_Onibus = new SqlParameter();

ParIdEmbarque\_Onibus.ParameterName = "@ID\_Embarque";

ParIdEmbarque\_Onibus.SqlDbType = SqlDbType.VarChar;

ParIdEmbarque\_Onibus.Size = 50;

ParIdEmbarque\_Onibus.Value = ID\_Embarque;

SqlCmd.Parameters.Add(ParIdEmbarque\_Onibus);

SqlParameter ParIdEmpresa\_Onibus = new SqlParameter();

ParIdEmpresa\_Onibus.ParameterName = "@ID\_Empresa";

ParIdEmpresa\_Onibus.SqlDbType = SqlDbType.VarChar;

ParIdEmpresa\_Onibus.Size = 50;

ParIdEmpresa\_Onibus.Value = ID\_Empresa;

SqlCmd.Parameters.Add(ParIdEmpresa\_Onibus);

Resp = SqlCmd.ExecuteNonQuery() == 1 ? "OK" : "Onibus Não Inserido";

}

catch (Exception error)

{

Resp = error.Message;

}

finally { if (Sqlcon.State == ConnectionState.Open) Sqlcon.Close(); }

return Resp;

}

#### 7.2.2.2 Função para Mostrar um Onibus no Sistema

public DataTable Mostrar\_Onibus()

{

DataTable DtResultado = new DataTable("Onibus");

SqlConnection Sqlcon = new SqlConnection();

try

{

Sqlcon.ConnectionString = ConexaoDB.cn;

SqlCommand SqlCmd = new SqlCommand();

SqlCmd.Connection = Sqlcon;

SqlCmd.CommandText = "Proc\_Mostrar\_Onibus";

SqlCmd.CommandType = CommandType.StoredProcedure;

SqlDataAdapter SqlDat = new SqlDataAdapter(SqlCmd);

SqlDat.Fill(DtResultado);

}

catch (Exception error)

{

DtResultado = null;

}

finally { if (Sqlcon.State == ConnectionState.Open) Sqlcon.Close(); }

return DtResultado;

}

A função de mostra e em formato DataTable pois os dados e mostrado num dataGridView na própria IDE do Visual Studio.

Os Procedimento no Banco de dados para essas duas Funções segam abaixo:

#### 7.2.2.3 Procedimento para Inserir um Onibus no Sistema

USE [BusSpot]

GO

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[Proc\_Inserir\_Onibus] Script Date: 15/05/2022 14:47:30 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

ALTER Procedure [dbo].[Proc\_Inserir\_Onibus]

@Numero\_Onibus varchar(50),

@Placa varchar(50),

@ID\_Comodidade int,

@ID\_Pagamento int,

@ID\_Acessibilidade int,

@ID\_Embarque int,

@ID\_Empresa int

as

insert Onibus(Numero\_Onibus,Placa,ID\_COMODIDADE,ID\_PAGAMENTO,ID\_ACESSIBILIDADE,ID\_EMBARQUE,ID\_EMPRESA) values (@Numero\_Onibus,@Placa,@ID\_Comodidade,@ID\_Pagamento,@ID\_Acessibilidade,@ID\_Embarque,@ID\_Empresa)

#### 7.2.2.4 Procedimento para Mostrar um Onibus no Sistema

USE [BusSpot]

GO

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[Proc\_Mostrar\_Onibus] Script Date: 15/05/2022 14:48:49 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

ALTER Procedure [dbo].[Proc\_Mostrar\_Onibus]

as

select Onibus. ID, Onibus.Numero\_Onibus as 'Numero', Onibus.Placa as 'Placa', Comodidades.Tipo\_Comodidade as 'Comodidade', Pagamento.Tipo\_Pag as 'Pagamento', Acessibilidade.Tipo\_Acessibilidade as 'Acessibilidade', Embarque.Tipo\_Emb 'Embarque',Empresa.Fantasia as 'Empresa'

from Onibus, Empresa, Acessibilidade,Comodidades,Embarque,Pagamento

where Onibus.ID\_COMODIDADE = Comodidades.ID and Onibus.ID\_PAGAMENTO = Pagamento.ID and Onibus.ID\_ACESSIBILIDADE = Acessibilidade.ID and Onibus.ID\_EMBARQUE = Embarque.ID and Onibus.ID\_EMPRESA = Empresa.ID

## 7.3 Desafios Enfrentados

### 7.3.1 Terminais

Da forma como a solução foi modelada, os terminais são considerados como pontos de ônibus, de forma que, sem tratamento, apenas os ônibus que passam pelo terminal com origem de outro lugar seriam mostrados.

### 7.3.2 Feriado

Em caso de feriados, os horários considerados para as estimativas devem ser os praticados no domingo. Para tanto, a aplicação precisa ter conhecimento dos feriados nacionais e municipais para aplicar as estimativas de forma correta. A melhor solução para essa questão seria, provavelmente, a criação de um serviço Web responsável por fazer a atualização dos feriados quando o usuário estivesse conectado à internet.

### 7.3.3 Meia Viagem

As linhas de ônibus são categorizadas geralmente como circular (horários apenas de ida) ou radial (horários de ida e de volta). Algumas linhas, entretanto, possuem os dois comportamentos: possuem horários de ida e de volta, mas alguns horários de ida realizam a volta completa. Estas linhas são categorizadas como “circular-radial”.

A forma como o aplicativo lida com isso é considerando o trajeto do ônibus de forma convencional, porém para linhas radiais quando ele chegar no ponto final e iniciar outra meia viagem o ícone do ônibus muda de cor, uma cor categoriza o sentido bairro-centro e outra cor indica centro-bairro, para o usuário saber com exatidão o sentido do ônibus terá que clicar em seu ícone para saber as informações.

## 7.4 Diagrama de Implantação

No contexto da Linguagem de modelagem unificada (UML), um diagrama de implementação faz parte da família de diagramação estrutural pois descreve um aspecto do sistema em si. Neste caso, o diagrama de implementação descreve a implementação física de informações geradas pelo programa de software em componentes de hardware. A informação gerada pelo software é chamada de artefato. Não se deve confundir este com o uso do mesmo termo em outras abordagens de modelagem, como BPMN.

Os diagramas de implementação são feitos de várias formas de UML. As caixas tridimensionais conhecidas como nódulos, representam os elementos básicos de software ou hardware, ou nódulos no sistema. As linhas de nódulo a nódulo indicam relacionamentos e as formas menores contidas dentro das caixas representam os artefatos de software empregados.

Figura 33 Diagrama de Implantação da BusSpot

Diagrama

Descrição gerada automaticamente

**Fonte:** Elaborada pelo autor

# 8 TESTE

Os testes podem ser apresentados na forma de casos de teste. Um caso de teste consiste em conjunto de detalhes necessários para se realizar um teste de software.

Os testes foram feitos pelo Software **TesteLink** é uma ferramenta open source de gerenciamento de testes de software. A ferramenta, possibilita que equipes de teste trabalhem de forma sincronizada, seja no mesmo espaço de trabalho ou remotamente, uma vez que o sistema é apresentado em uma interface web e permite adicionar colaboradores com diversos níveis de acesso (Administrador, Líder de Testes, Projetista de Testes, Testador e Convidado).

Funcionalmente, a ferramenta permite que sejam cadastrados Projetos de Teste, bem como seus respectivos Planos e Casos de testes a serem validados. Então, a partir da definição do escopo do projeto de teste, é possível controlar e armazenar a execução dos casos de teste especificados.

Outra característica marcante da ferramenta, é a possibilidade de controlar a execução dos testes designando, por exemplo, um testador para ser responsável pela execução de um conjunto de testes pré-definido. Também é possível gerar um relatório dos testes que já foram executados .

## 8.1 Plano de Teste de Cadastros

### 8.1.1 Plano de Teste de Estado

Figura 34 Exemplo de Caso de Teste de Estado Elaborado na Ferramenta Testlink

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

**Fonte:** Elaborada pelo autor

#### 8.1.1.1 Caso de Teste de Estado

Figura 35 Caso de Teste feito pelo TestLink para a validação de um Estado

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

**Fonte:** Elaborada pelo autor

Figura 36 Detalhamento do Caso de Teste de Estado feito pelo TestLink

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

**Fonte:** Elaborada pelo autor

8.1.2 Plano de Teste de Cidade

Figura 37 Exemplo de Caso de Teste de Cidade Elaborado na Ferramenta Testlink

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

**Fonte:** Elaborada pelo autor

#### 8.1.2.1 Caso de Teste de Cidade

Figura 38 Exemplo do caso de Teste de Cidade feito pelo TestLink

Uma imagem contendo Aplicativo

Descrição gerada automaticamente

**Fonte:** Elaborada pelo autor

Figura 39 Detalhamento do Caso de Teste de Cidade feito pelo TestLink

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

**Fonte:** Elaborada pelo autor

### 8.1.3 Plano de Teste de Reclamação

Figura 40 Plano de Teste de Reclamação feito pelo TestLink

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

**Fonte:** Elaborada pelo autor

#### 8.1.3.1 Caso de Teste de Reclamação

Figura 41 Exemplo do caso de Teste de Reclamação feito pelo TestLink

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

**Fonte:** Elaborada pelo autor

Figura 42 Detalhamento do Caso de Teste de Reclamação feito pelo TestLink

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

**Fonte:** Elaborada pelo autor

# 09 RESULTADOS E DISCUSSÃO

A partir dos diagramas e dos requisitos elencados, torna-se possível a criação da solução para o problema apresentado no capítulo 1 – O problema enfrentado pela gestão pública no quesito dos ônibus municipais, popularmente chamados de circulares.

Conclui-se que com a aplicação sugerida é possível mitigar tal problema, uma vez que as rotas e horários dos ônibus seriam facilmente localizados na aplicação, tornando a utilização das linhas pelos usuários mais fácil, prática e eficiente.

O aplicativo ainda possui uma interface desktop, para ser utilizado pelas empresas, podendo ser utilizado para o monitoramento e obtenção de informações acerca dos ônibus e das linhas em que estes estão.

Tais ferramentas permitem a melhoria da gestão dos transportes municipais de forma mais eficiente, solucionando o problema recentemente enfrentado pelos usuários do sistema de transporte público da cidade de Birigui.

# REFERÊNCIAS

A Guide to the Business Analysis Body of Knowledge[r], Release 1.6 [c]2006, International Institute of Business Analysis, p. 192.Citado 1 vez na página 25.

População de Birigui amanhece sem transporte coletivo na cidade. sbtinterior.com, 2022. Disponível em <https://sbtinterior.com/noticia/populacao-de-birigui-amanhece-sem-transporte-coletivo-na-cidade,4385666144813.html > acesso em 31 de março 2022. Citado 1 vez na página 25.

‌

CASTRO, E. O que são requisitos? E requisitos de software? Disponível em: <http://rederequisitos.com.br/o-que-sao-requisitos-e-requisitos-de-software/>. Acesso em: 2 abr. 2022. Citado 1 vez na página 27

BERNARDI, G.; FONTOURA, L. M.; CORDENONSI, A. Z. Elicitação: Ferramenta de Apoio ao Ensino de Elicitação de Requisitos de Software baseada em Instituições Eletrônicas. II Workshop Escola de Sistemas de Agentes para Ambientes Colaborativos. Anais do…, Rio Grande do Sul, Santa Cruz do Sul, 2008. Citado 1 vezes na página 27

MENDES, Antonio. Requisitos Não Funcionais e Arquitetura de Software. DevMedia. Disponível em: <https://www.devmedia.com.br/artigo-engenharia-de-software-3-requisitos-nao-funcionais/9525>. Acesso em: 24 Mar. 2022. Citado 1 vezes na página 32

BUSVISION. Legislação para ônibus: capitais onde DVR Veicular é exigido. Disponível em: <https://www.busvision.com.br/blog/legislacao-para-onibus-confira-as-leis-de-todas-as-capitais-brasileiras-onde-dvr-veicular-e-obrigatorio-no-transporte-urbano/>. Acesso em: 3 abr. 2022. Citado 1 vezes na página 32

RENATO JOSE GROFFE. Modelagem de sistemas com UML: Principais tipos de diagramas. Disponível em: <https://www.devmedia.com.br/modelagem-de-sistemas-atraves-de-uml-uma-visao-geral/27913>. Acesso em: 10 abr. 2022. Citado 1 vezes na página 33

O que é um diagrama UML? Disponível em: <https://www.lucidchart.com/pages/pt/o-que-e-uml>. Acesso em: 10 abr. 2022.Citado uma vez na página 33

CECILIA, M. “Modelo de Domínio: Visualizando Conceitos” . M. CECILIA C.

BARANAUSKAS, 2009. Disponível em: <https://www.ic.unicamp.br/~ariadne/mc436/1s2017/Lar10ModDom.pdf>. Acesso

em: 16 de junho de 2020. Acesso em: 10 abr. 2022. Citado uma vez na página 42.

‌ O que é um diagrama de objetos? Disponível em: <https://www.lucidchart.com/pages/pt/o-que-e-diagrama-de-objetos-uml>. Acesso em: 10 abr. 2022. Citado uma vez na página 43.

RENATO JOSE GROFFE. Modelagem de sistemas com UML: Principais tipos de diagramas. Disponível em: <https://www.devmedia.com.br/modelagem-de-sistemas-atraves-de-uml-uma-visao-geral/27913>. Acesso em: 11 abr. 2022. Citado uma vez na página 2 e página 44.

O que é diagrama de atividades UML? Disponível em: <https://www.lucidchart.com/pages/pt/o-que-e-diagrama-de-atividades-uml#section\_0>. Acesso em: 16 abr. 2022. Citado uma vez na página 45.

O que é um diagrama de máquina de estados? Disponível em: <https://www.lucidchart.com/pages/pt/o-que-e-diagrama-de-maquina-de-estados-uml>. Acesso em: 23 abr. 2022. Citado uma vez na Página 47.

‌ EDITORIAL AELA. Protótipos: O Que Você Precisa Saber Para Seus Projetos de UX/UI. Disponível em: <https://medium.com/aela/prot%C3%B3tipos-o-que-voc%C3%AA-precisa-saber-para-seus-projetos-de-ux-ui-5bfc00485e82>. Acesso em: 11 abr. 2022. Citado 1 vez na página 48.

NUNES, T. Mapeamento objeto-relacional: como funciona e técnicas - Geek Blog |. Disponível em: <https://blog.geekhunter.com.br/mapeamento-objeto-relacional/>. Acesso em: 7 maio. 2022. Citado 1 vez na página 59.

QUE, O. O que é um DTO? Disponível em: <https://pt.stackoverflow.com/questions/31362/o-que-%C3%A9-um-dto>. Acesso em: 7 maio. 2022. Citado 1 vez na página 68.

O que é um diagrama de sequência UML? Disponível em: <https://www.lucidchart.com/pages/pt/o-que-e-diagrama-de-sequencia-uml>. Acesso em: 7 maio. 2022. Citado 1 vez na página 70.

O que é IDE? Ambiente de Desenvolvimento Integrado. Disponível em: <https://www.redhat.com/pt-br/topics/middleware/what-is-ide>. Acesso em: 14 maio. 2022. Citado 1 vez na página 72.

Diagrama de componentes UML: o que é, como fazer e exemplos. Disponível em: <https://www.lucidchart.com/pages/pt/diagrama-de-componentes-uml>. Acesso em: 14 maio. 2022. Citado 1 vez na página 73.

‌

‌